# Guide des opérations OmniCRM

**OmniCRM** est la solution BSS/OSS complète d'Omnitouch pour les fournisseurs de services mobiles et fixes. Une plateforme complète qui gère tout, de l'intégration des clients à la facturation, en passant par la fourniture et le support - le tout dans un système intégré.

---

# Commencer

## Pour le personnel du service client

**Vos premières étapes :**

1. **Apprendre l'interface** - Familiarisez-vous avec la navigation et la recherche
2. **Créer un client** - Création de client étape par étape
3. **Ajouter un service** - Fournissez votre premier service
4. **Traiter un paiement** - Gérez les paiements des clients
5. **Recharger des services** - Ajoutez du crédit aux comptes clients

**Tâches quotidiennes :**

- **Gestion des services** - Gérez les services clients
- **Voir l'utilisation** - Vérifiez les soldes et l'utilisation
- **Voir les relevés d'appels** - Inspectez les enregistrements d'appels, SMS et données (CDRs)
- **Modifier les services** - Changez les configurations de service
- **Générer des factures** - Créez et envoyez des factures

# Pour les administrateurs système

**Configuration et paramétrage :**

1. **Comprendre l'architecture** - Vue d'ensemble du système
2. **Configurer le système** - Paramètres du système
3. **Créer des produits** - Construisez votre catalogue
4. **Écrire des playbooks** - Automatisez la fourniture
5. **Configurer des utilisateurs** - Créez des comptes et assignez des rôles

**Sujets avancés :**

- **Gestion des stocks** - Gérez les actifs
- **Intégration API** - Connectez des systèmes externes
- **Configuration de la sécurité** - Configurez 2FA et les autorisations

# Pour les clients

**Utilisation du portail de self-service :**

- **Accéder à votre compte** - Connectez-vous et naviguez
- **Voir les services** - Consultez vos services actifs
- **Vérifier l'utilisation** - Surveillez les données et les soldes
- **Payer des factures** - Effectuez des paiements en ligne
- **Recharger des services** - Ajoutez du crédit instantanément

---

# Référence rapide des tâches

| Je veux... | Documentation |
|---|---|
| Créer un nouveau client | Créer un client |
| Ajouter un service à un client | Ajouter un service |
| Voir l'utilisation et les soldes des services | Utilisation des services |
| Voir les relevés d'appels/SMS/données d'un service | Relevés d'appels |
| Traiter un paiement | Traiter un paiement |
| Générer une facture | Gestion des factures |
| Recharger un service | Recharger |
| Gérer les stocks | Système de gestion des stocks |
| Créer un produit | Cycle de vie des produits |
| Écrire un playbook de provisionnement | Playbooks Ansible |
| Configurer des comptes utilisateurs | RBAC |
| Activer 2FA | Authentification à deux facteurs |
| Rechercher quoi que ce soit | Recherche globale |
| Comprendre le système | Architecture du système |
| Utiliser l'API | Documentation API |

# Bibliothèque de documentation

## Lecture essentielle

- **Architecture du système** - Vue technique complète avec diagrammes
- **Guide du cycle de vie des produits** - Gestion de produit de bout en bout
- **Guide des playbooks Ansible** - Maîtriser l'automatisation du provisionnement

## Gestion des clients

- Clients, Contacts & Sites - Modèle de données et relations
- Création de clients - Guide étape par étape
- Étiquettes de clients - Organisez avec des étiquettes
- Attributs de clients - Métadonnées personnalisées
- Journal d'activité - Suivez tous les changements

## Support

- Système de tickets de support - Gérez les tickets de support client
- Base de connaissances FAQ - Réponses en libre-service pour les problèmes courants

## Gestion du travail

- Tableaux - Gestion du travail de style Kanban

## Opérations de service

- Gestion des services - Vue d'ensemble
- Ajout de services - Workflow de provisionnement
- Attribution de plans - Attribution de plan
- Modification des services - Apporter des modifications
- Utilisation et soldes des services - Surveillez l'utilisation

- Relevés d'appels (CDRs) - Voir les relevés d'appels, SMS et données
- Recharger - Ajouter des crédits

## Facturation & Finance

- Vue d'ensemble de la facturation - Guide complet de facturation
- Guide du système de paiement - Architecture du système de paiement
- Méthodes de paiement - Gestion des méthodes de paiement
- Transactions - Charges et crédits
- Traitement des paiements - Workflows de paiement
- Gestion des factures - Générer et gérer des factures

## Administration

- Configuration du système - Configurez OmniCRM
- Gestion des stocks - Suivi des actifs
- Provisionnement de cartes SIM - Gestion du cycle de vie des cartes SIM
- Gestion des stocks eSIM - Suivi et attribution des eSIM
- Clés API - Gestion de l'accès API
- Surveillance & Métriques - Santé et observabilité du système

## Sécurité & Accès

- Flux d'authentification - Comment fonctionne l'authentification
- Authentification à deux facteurs - Configurez 2FA
- Contrôle d'accès basé sur les rôles - Utilisateurs, rôles et autorisations

## Intégrations

- Campagnes Email & SMS (EDM) - Campagnes de communication en masse
- Intégration Mailjet - Service d'email
- API de notification - Intégration de notifications sortantes
- Documentation API - Référence API REST

## Orienté client

- Portail de self-service - Guide du portail client
- Service client - Fonctionnalités en libre-service
- Système de diffusion cellulaire - Alertes d'urgence

## Référence

- Journal des modifications - Historique des versions
- Glossaire des termes - Référence terminologique

---

# Qu'est-ce qui rend OmniCRM spécial ?

OmniCRM regroupe tous les outils nécessaires pour gérer une entreprise de télécommunications moderne :

## Provisionnement de service automatisé

OmniCRM élimine la configuration manuelle. Il utilise **l'automatisation Ansible** pour provisionner des services en quelques secondes. Le système gère le travail automatiquement lorsque vous activez une carte SIM, configurez l'équipement client ou mettez en place un bundle complexe.

**Ce qu'il fait :**

- Provisionne des services d'un simple clic
- Configure automatiquement l'équipement réseau
- Crée des comptes de facturation en temps réel
- Envoie des emails de bienvenue et des notifications SMS
- Rétrograde automatiquement en cas d'échec

**En savoir plus sur le provisionnement** | **Voir le guide des playbooks Ansible**

## Facturation et paiements intelligents

L'intégration intégrée avec **CGRateS** fournit un tarif et une facturation en temps réel, tandis que Stripe gère le traitement des paiements de manière transparente.

**Ce qu'il fait :**

- Suivi et tarification en temps réel
- Génération automatique de factures
- Traitement des cartes de crédit via Stripe
- Modèles de facturation prépayés et postpayés
- Tarification flexible et codes promotionnels
- Calcul et reporting des taxes

**Explorez les fonctionnalités de facturation** | **Traitement des paiements** | **Gestion des factures**

## Gestion complète des clients

Gérez les clients, les contacts, plusieurs sites et services avec un modèle de relation complet.

**Ce qu'il fait :**

- Suivez les clients, les contacts et les emplacements de services
- Gérez plusieurs services par client
- Stockez des attributs personnalisés et des métadonnées
- Étiquetez les clients pour l'organisation
- Historique complet des activités et journaux d'audit
- Intégration Google Maps pour la géocodification des sites

**Guide de gestion des clients** | **Créer un client** | **Étiquettes de clients**

## Gestion des stocks

Suivez et gérez tous vos actifs physiques et virtuels - des cartes SIM aux routeurs en passant par les blocs d'adresses IP.

**Ce qu'il fait :**

- Suivez les cartes SIM, l'équipement et les accessoires
- Gérez les numéros de téléphone et les blocs d'adresses IP
- Attribution automatisée lors du provisionnement
- Capacités d'importation et d'exportation en masse
- Stockage de la configuration de l'équipement
- Modèles d'inventaire personnalisés

**Documentation du système de gestion des stocks**

## Portail de self-service client

Donnez à vos clients un **portail de self-service** où ils peuvent gérer leurs propres services.

**Ce que les clients peuvent faire :**

- Voir les services et l'utilisation
- Vérifier les soldes et les allocations de données
- Payer des factures en ligne
- Télécharger des factures au format PDF
- Mettre à jour les informations de contact
- **Recharger leurs services** instantanément

**Guide du portail de self-service** | **Fonctionnalités du service client**

## Sécurité d'entreprise

Conçu avec la sécurité au cœur, avec une authentification et une autorisation complètes.

**Fonctionnalités de sécurité :**

- **Authentification basée sur JWT**
- **Authentification à deux facteurs (2FA)** avec TOTP et codes de secours
- **Contrôle d'accès basé sur les rôles (RBAC)** avec des autorisations granulaires
- Vérification par email pour les changements de compte
- Journalisation complète des audits via le journal d'activité
- Gestion des sessions et contrôles de timeout

**Guide d'authentification** | **Documentation RBAC** | **Configuration de 2FA**

## Intégrations puissantes

OmniCRM s'intègre parfaitement avec des services de premier plan dans l'industrie :

- **CGRateS** - Moteur de facturation et de tarification en temps réel pour la facturation de niveau télécom
- **Stripe** - Traitement des paiements sécurisé et stockage de cartes
- **Mailjet** - Livraison d'emails professionnels avec modèles
- **Google Maps** - Validation d'adresse et géocodage pour des emplacements de site précis
- **API RESTful** - Intégrez avec vos systèmes existants

**Documentation API** | **Intégration Mailjet**

## Catalogue de produits flexible

Créez tout type d'offre de produit - des services simples aux bundles complexes avec plusieurs composants.

**Types de produits pris en charge :**

- **Autonome** - Services uniques (forfaits mobiles, paquets Internet)
- **Bundles** - Offres combinées (internet + TV + téléphone)

- **Addons** - Services supplémentaires (recharges de données, appels internationaux)
- **Promotions** - Offres spéciales et remises

Chaque produit peut avoir sa propre automatisation de provisionnement, ses règles de tarification et sa logique commerciale.

**Guide du cycle de vie des produits** | **Concepts de produits et services**

## Système de diffusion d'urgence

Pour les opérateurs mobiles, OmniCRM inclut un **système de diffusion cellulaire** pour les alertes de sécurité publique.

**Ce qu'il fait :**

- Envoyer des alertes d'urgence à des zones géographiques
- Support multilingue des messages
- Ciblage par opérateur de réseau mobile
- Conformité aux normes d'alerte gouvernementales

**Documentation sur la diffusion cellulaire**

## Recherche et navigation puissantes

Trouvez n'importe quoi instantanément avec **la recherche globale** à travers les clients, les services, les factures, et plus encore.

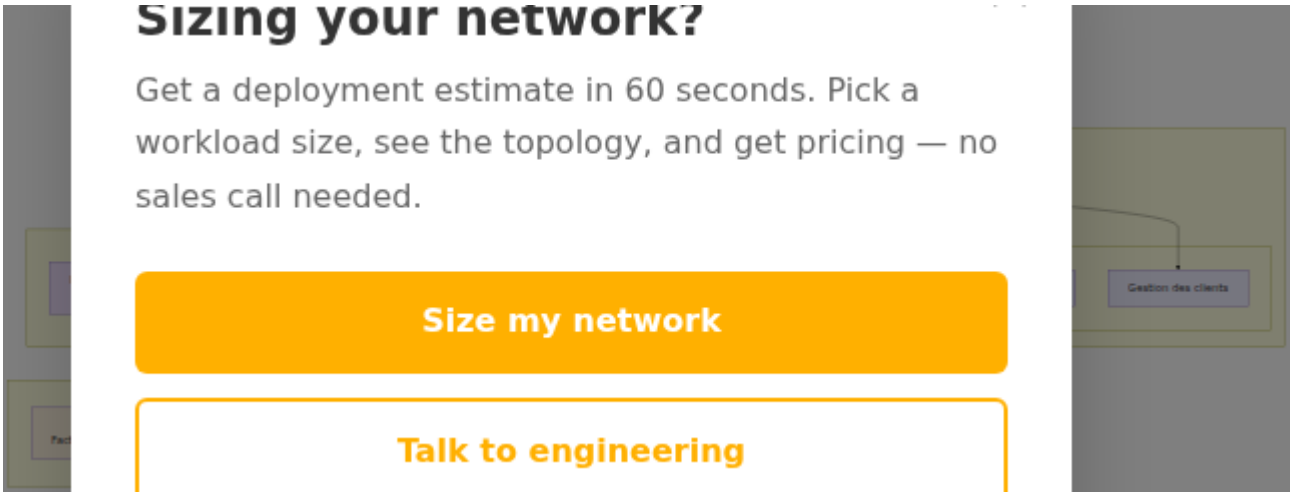
**Fonctionnalités de recherche :**

- Recherche à travers toutes les entités
- Filtrer et trier les résultats
- Raccourcis de navigation rapides
- Suggestions intelligentes

**Guide de navigation** | **Recherche globale**

---

# Architecture du système




**Voir la documentation complète de l'architecture**

---

# Concepts clés

## Comprendre les produits vs services

C'est l'un des concepts les plus importants dans OmniCRM :

- **Produit** = Un modèle ou une offre dans votre catalogue (par exemple, "Forfait mobile illimité")
- **Service** = Une instance active d'un produit pour un client spécifique (par exemple, "Forfait mobile illimité de John Smith")

Lorsque vous provisionnez un produit, le système utilise **l'automatisation Ansible** pour créer le(s) service(s) réel(s). Un produit peut créer plusieurs services, aucun service (configuration uniquement), ou modifier des services existants.

**En savoir plus sur les produits et services**

## La magie du provisionnement

Le secret d'OmniCRM est son **système de provisionnement automatisé** :




Le playbook gère tout - de la création de comptes dans CGRateS à la configuration des routeurs en passant par l'envoi d'emails de bienvenue. Et si quelque chose échoue, il se rétrograde automatiquement.

**Plongée approfondie : Playbooks Ansible** | **Système de provisionnement**

## Facturation simplifiée



Chaque service génère des **transactions** (charges ou crédits). Celles-ci sont regroupées en **factures** et envoyées aux clients. Avec l'intégration de Stripe, les paiements peuvent être traités automatiquement.

**Vue d'ensemble de la facturation** | **Traitement des paiements**

## Hiérarchie des clients

```
Client (ABC Company)
├── Contacts (John Smith - Facturation, Jane Doe - Technique)
├── Sites (Siège, Bureau)
└── Services
    ├── Internet - Siège
    ├── Forfait mobile - John Smith
    └── VoIP - Bureau
```

Tout est organisé de manière hiérarchique, ce qui facilite la gestion des clients avec plusieurs emplacements et services.

**Clients, Contacts & Sites**

# Pourquoi choisir OmniCRM ?

## Plateforme tout-en-un

Tout ce dont vous avez besoin dans un système intégré unique - pas besoin de jongler avec plusieurs outils ou fournisseurs.

## Automatisation d'abord

Le provisionnement alimenté par Ansible signifie que les services sont déployés en quelques secondes, et non en heures. Réduisez les erreurs, gagnez du temps, augmentez la satisfaction client.

## Flexible et personnalisable

Des définitions de produits aux playbooks de provisionnement en passant par les attributs personnalisés - adaptez OmniCRM pour correspondre exactement à vos processus commerciaux.

## Conçu pour les télécoms

Conçu spécifiquement pour les fournisseurs de services avec des fonctionnalités telles que l'intégration CGRateS, la gestion des stocks et la tarification en temps réel.

## Sécurité d'entreprise

L'authentification JWT, 2FA, RBAC et la journalisation complète des audits garantissent que vos données sont sécurisées et conformes.

## API-Driven

Une API REST complète signifie que vous pouvez intégrer OmniCRM avec n'importe quel système existant ou construire des outils personnalisés.

---

# Obtenez de l'aide

## Ressources de documentation

- Commencez par **l'architecture du système** pour une vue technique
- Consultez les **guides de démarrage** pour l'intégration spécifique aux rôles
- Vérifiez la **référence rapide des tâches** pour les opérations courantes
- Consultez le glossaire pour la terminologie

## Besoin de plus d'aide ?

Parcourez la bibliothèque de documentation complète ci-dessus ou utilisez la **recherche globale** pour trouver des sujets spécifiques.

---

*OmniCRM - BSS/OSS complet pour les fournisseurs de services modernes*

*Dernière mise à jour : 2025-12-23*

# Authentification à Deux Facteurs (2FA)

## Sécurité Renforcée avec des Mots de Passe à Usage Unique Basés sur le Temps

OmniCRM prend en charge **l'authentification à deux facteurs (2FA)** utilisant des mots de passe à usage unique basés sur le temps (TOTP). Cela ajoute une couche de sécurité supplémentaire en exigeant que les utilisateurs fournissent à la fois leur mot de passe et un code sensible au temps provenant d'une application d'authentification.

Voir aussi : RBAC pour les autorisations de gestion de la 2FA, Flux d'Authentification pour les détails du processus de connexion.

# Objectif

La 2FA fournit :

1. **Sécurité Renforcée** : Protège les comptes même si les mots de passe sont compromis.
2. **Conformité** : Répond aux exigences de sécurité pour les secteurs réglementés.
3. **Choix de l'Utilisateur** : Optionnel pour les utilisateurs, peut être appliqué par rôle ou globalement.
4. **Norme de l'Industrie** : Utilise le protocole TOTP compatible avec Google Authenticator, Authy, Microsoft Authenticator et d'autres applications standard.

# Comment Fonctionne la 2FA

Lorsque la 2FA est activée pour un utilisateur :

1. **Configuration** : L'utilisateur scanne un code QR avec son application d'authentification lors de l'inscription.
2. **Connexion** : Après que l'utilisateur a saisi son nom d'utilisateur et son mot de passe, il fournit le code à 6 chiffres de son application.
3. **Vérification** : Le système valide que le code basé sur le temps correspond à la valeur attendue.
4. **Accès** : L'utilisateur obtient l'accès uniquement après que les deux facteurs ont été vérifiés.

# Activation de la 2FA

## Pour les Utilisateurs Individuels

Les utilisateurs peuvent activer la 2FA pour leur propre compte :

1. Accédez à **Paramètres de l'Utilisateur** ou **Profil**
2. Sélectionnez **Activer l'Authentification à Deux Facteurs**
3. Scannez le code QR avec une application d'authentification
4. Entrez le code de vérification pour confirmer la configuration
5. Enregistrez les codes de secours dans un endroit sécurisé

# Sauvegarde & Récupération

## Codes de Secours

Lors de l'activation de la 2FA, les utilisateurs reçoivent des codes de secours qui peuvent être utilisés si leur appareil n'est pas disponible :

- Chaque code est à usage unique
- Stockez les codes en toute sécurité hors ligne
- Générez de nouveaux codes si tous sont utilisés

## Réinitialisation par l'Administrateur

Si un utilisateur perd l'accès à son authentificateur et que tous les codes de secours sont épuisés, un administrateur ayant accès à la base de données peut réinitialiser manuellement la 2FA de l'utilisateur en effaçant le champ *totp_secret* dans la base de données. L'utilisateur peut ensuite réactiver la 2FA.

# Intégration API

## Activer la 2FA pour un utilisateur

POST /2fa/enable/user/{user_id}

```
{
  "password": "current_password"
}
```

La réponse inclut l'URI de provisionnement (pour le code QR) et les codes de secours.

## Vérifier la configuration de la 2FA

POST /2fa/verify-setup/user/{user_id}

```
{
  "code": "123456"
}
```

## Vérifier la 2FA lors de la connexion

POST /2fa/verify/user/{user_id}

```
{
  "code": "123456"
}
```

Renvoie un jeton d'accès, un jeton de rafraîchissement et des données utilisateur après une vérification réussie.

## Régénérer des codes de secours

POST /2fa/backup-codes/regenerate/user/{user_id}

Nécessite une authentification. Renvoie un nouvel ensemble de codes de secours.

# Meilleures Pratiques

- **Codes de secours d'abord.** Toujours sauvegarder les codes de secours avant de terminer la configuration de la 2FA.
- **Éduquer les utilisateurs.** Fournir des instructions claires pour la configuration et la récupération.
- **Processus de réinitialisation sécurisé.** Vérifiez l'identité de l'utilisateur avant de réinitialiser manuellement la 2FA dans la base de données.

# FAQ

**Quelles applications d'authentification sont prises en charge ?** Toute application compatible TOTP (Google Authenticator, Authy, Microsoft

Authenticator, 1Password, etc.).

**Que faire si je perds mon téléphone ?** Utilisez un code de secours ou contactez un administrateur pour réinitialiser la 2FA.

**Puis-je utiliser SMS au lieu d'une application ?** Actuellement, seules les applications d'authentification TOTP sont prises en charge.

**La 2FA est-elle obligatoire ?** Cela dépend de la politique de votre organisation. La 2FA est généralement requise pour les rôles administratifs et de support, mais est optionnelle pour les utilisateurs clients. Le système n'impose pas la 2FA pour les comptes clients (les utilisateurs de rôle client ne voient pas les invites d'inscription à la 2FA).

**Combien de temps les codes TOTP sont-ils valides ?** Les codes se rafraîchissent toutes les 30 secondes et ont une petite fenêtre de temps pour la validation (accepte généralement les codes de la fenêtre de 30 secondes actuelle plus les fenêtres précédente/suivante pour la tolérance au décalage horaire).

# Gestion des Clés API

L'interface de gestion des clés API fournit une **interface utilisateur basée sur le web** pour créer, surveiller et gérer les clés API utilisées pour l'accès programmatique à l'API OmniCRM.

> ⓘ **REMARQUE**
>
> Pour des concepts d'authentification API généraux et des exemples d'utilisation, voir `concepts_api`.

# Aperçu

Les clés API permettent une **authentification sécurisée et de longue durée** pour :

- Intégrations serveur à serveur
- Scripts d'automatisation
- Applications tierces
- Tâches planifiées et travaux cron
- Systèmes de surveillance externes

Contrairement aux tokens JWT (qui expirent après quelques minutes/heures), les clés API restent valables jusqu'à ce qu'elles soient révoquées manuellement ou jusqu'à leur date d'expiration.

# Accéder à la Gestion des Clés API

Naviguez vers :

Ou directement :

**Permission requise :** `MANAGE_API_KEYS` (rôle admin)

# Vue Liste des Clés API

La page principale affiche toutes les clés API dans un format de tableau :

**Colonnes :**

- **Nom** - Étiquette descriptive pour la clé API (par exemple, "Système de Provisionnement", "Outil de Surveillance")
- **Créé par** - Nom d'utilisateur de la personne qui a créé la clé
- **Clé API** - La chaîne de clé réelle (partiellement masquée pour la sécurité)
- **Statut** - Actif, Expiré ou Révoqué
- **Date de création** - Quand la clé a été générée
- **Date d'expiration** - Quand la clé expirera automatiquement
- **Actions** - Boutons Modifier, Supprimer, Régénérer

**Exemple d'affichage :**

## Widgets du Tableau de Bord

En haut de la page, des statistiques récapitulatives sont affichées :

- **Total des Clés API** - Nombre total de clés API (actives et inactives)
- **Clés Actives** - Clés actuellement valides
- **Expirant Bientôt** - Clés expirant dans les 30 jours
- **Clés Expirées** - Clés dépassant leur date d'expiration

# Création d'une Clé API

**Étape 1 : Cliquez sur "Ajouter une Clé API"**

Cliquez sur le bouton **+ Ajouter** en haut à droite de la liste des clés API.

**Étape 2 : Remplissez les Détails**

Un formulaire modal apparaît demandant :

> Nom : _____
> (par exemple, "Système de Provisionnement")
>
> Description : _____
> (Optionnel - but de cette clé)
>
> Date d'expiration : [Sélecteur de Date]
> (Optionnel - laissez vide pour pas d'expiration)
>
> Permissions : ☐ Voir Clients ☐ Créer Clients ☐ Voir Services ☐ Créer Services ☐ Provisionnement ☐ Voir Inventaire ☑ Admin (toutes les permissions)
>
> [Annuler] [Générer la Clé]

**Directives pour les Champs :**

**Nom** (obligatoire)

- Identifiant court et descriptif
- Exemples : "Système de Provisionnement", "Intégration de Facturation", "Surveillance"
- Utilisé dans les journaux d'audit et affiché dans la liste

**Description** (optionnel)

- Explication plus détaillée
- Exemples : "Utilisé par le serveur de provisionnement Ansible", "Synchronisation de facturation tierce"
- Aide les futurs administrateurs à comprendre le but de la clé

**Date d'expiration** (optionnel)

- Si vide : La clé n'expire jamais (non recommandé)
- Si définie : La clé devient automatiquement invalide après cette date
- Recommandé : Définir une expiration pour la sécurité (90 jours à 1 an)

**Permissions**

- Sélectionnez des permissions spécifiques ou cochez "Admin" pour un accès complet
- Suit le même modèle de permissions basé sur les rôles que les comptes utilisateurs
- **Meilleure Pratique :** Attribuer le minimum de permissions nécessaires

**Étape 3 : Générer et Copier la Clé**

Après avoir cliqué sur **"Générer la Clé"**, le système affiche la clé API nouvellement créée :

> ⚠ Copiez cette clé maintenant - elle ne sera plus affichée !
>
> sk_live_a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6q7r8s9t0
>
> [Copier dans le Presse-papiers]
>
> [Fermer]

> ⚠ **ATTENTION**
>
> **Sauvegardez la clé API immédiatement !**
>
> Une fois que vous fermez cette boîte de dialogue, la clé complète ne peut plus être récupérée. Vous ne verrez qu'une version masquée (`sk_live_...XYZ`) dans la vue de liste.
>
> Si vous perdez la clé, vous devez **la régénérer**, ce qui invalide l'ancienne clé et peut casser les intégrations existantes.

**Étape 4 : Configurez Votre Application**

Utilisez la clé API dans les requêtes de votre application :

```
curl -X GET "https://yourcrm.com/crm/customers" \
  -H "X-API-KEY: sk_live_a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6q7r8s9t0"
```

Ou dans des variables d'environnement :

```
export
CRM_API_KEY="sk_live_a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6q7r8s9t0"
```

# Gestion des Clés Existantes

## Affichage des Détails de la Clé

Cliquez sur n'importe quel nom de clé API pour voir les détails complets :

- Nom complet de la clé et description
- Horodatage de création
- Nom d'utilisateur du créateur
- Permissions associées
- Statistiques d'utilisation (si mises en œuvre)
- Journaux d'accès récents

## Régénération d'une Clé API

Si une clé API est compromise ou perdue, régénérez-la :

1. Cliquez sur le ⋮ **(trois points)** dans la colonne Actions
2. Sélectionnez **"Régénérer la Clé"**
3. Confirmez l'action

> ⚠ **ATTENTION**
>
> **La régénération invalide immédiatement l'ancienne clé.**
>
> Toutes les applications utilisant l'ancienne clé cesseront de fonctionner. Mettez à jour toutes les intégrations avec la nouvelle clé avant de régénérer.

**Ce qui se passe :**

- L'ancienne clé est révoquée

- Une nouvelle clé avec les mêmes permissions est générée
- La nouvelle clé est affichée (copiez-la immédiatement)
- Le nom, la description et les permissions restent inchangés

# Révocation (Suppression) d'une Clé API

Pour supprimer définitivement une clé API :

1. Cliquez sur le **⋮ (trois points)** dans la colonne Actions
2. Sélectionnez **"Supprimer"**
3. Confirmez la suppression

**Ce qui se passe :**

- La clé est immédiatement révoquée
- Toutes les requêtes utilisant cette clé renvoient `401 Unauthorized`
- La clé est supprimée de la base de données
- **Ne peut pas être annulé** - la clé ne peut pas être récupérée

**Quand révoquer :**

- L'intégration n'est plus nécessaire
- La clé a été compromise
- Le système utilisant la clé a été décommissionné
- Remplacement par une nouvelle clé avec des permissions différentes

# Modification des Détails de la Clé API

Pour modifier les détails d'une clé API :

1. Cliquez sur le **⋮ (trois points)** dans la colonne Actions
2. Sélectionnez **"Modifier"**
3. Mettez à jour le nom, la description, l'expiration ou les permissions
4. Cliquez sur **"Sauvegarder les Modifications"**

**Champs Modifiables :**

- Nom
- Description
- Date d'expiration
- Permissions

**Non Modifiable :**

- La valeur de la clé elle-même (utilisez Régénérer pour changer)
- Date de création
- Créé par l'utilisateur

# Statut de la Clé API

Les clés API peuvent avoir plusieurs statuts :

**Actif**

- La clé est valide et peut être utilisée
- Dans la date d'expiration (ou pas d'expiration définie)
- Pas révoquée manuellement
- Affichée avec un badge vert

**Expirant Bientôt**

- Actif mais expirera dans les 30 jours
- Affichée avec un badge orange/d'avertissement
- Envisagez de faire une rotation avant l'expiration

**Expiré**

- Passé la date d'expiration
- N'accepte plus l'authentification
- Affichée avec un badge rouge
- Peut être supprimée ou l'expiration prolongée

**Révoqué**

- Supprimé/désactivé manuellement
- Permanent invalide
- N'est plus affiché dans la liste active

# Filtrage et Recherche

La liste des Clés API prend en charge :

**Recherche :**

Recherchez par nom, description ou clé partielle :

**Filtrer par Statut :**

Menu déroulant de filtrage pour afficher :

- Toutes les Clés
- Actives Seulement
- Expirant Bientôt (dans les 30 jours)
- Expirées

**Trier :**

Cliquez sur les en-têtes de colonne pour trier par :

- Nom
- Date de Création
- Date d'Expiration
- Créé par

# Meilleures Pratiques de Sécurité

## Génération de Clé API

- **Longueur :** Les clés doivent avoir au moins 32 caractères (le système impose cela)

- **Aléatoire :** Générées à l'aide de générateurs de nombres aléatoires cryptographiquement sécurisés
- **Format :** Typiquement préfixé (par exemple, `sk_live_`) pour identification

# Stockage de Clé API

**Dans le CRM :**

- Les clés sont hachées avant stockage (comme les mots de passe)
- La clé complète n'est affichée qu'une seule fois lors de la création
- La base de données stocke le hachage pour vérification
- Même les administrateurs ne peuvent pas récupérer la clé complète plus tard

**Dans Votre Application :**

- Stockez dans des variables d'environnement, pas dans le code
- Utilisez des systèmes de gestion de secrets (AWS Secrets Manager, HashiCorp Vault)
- Ne jamais commettre les clés dans le contrôle de version
- Faites une rotation des clés périodiquement (90-365 jours)

# Gestion des Permissions

- **Principe du Moindre Privilège** - N'accorder que les permissions nécessaires
- Évitez de créer des clés admin à moins que cela ne soit absolument nécessaire
- Utilisez des clés séparées pour différents systèmes/buts
- Passez en revue les permissions régulièrement

# Surveillance et Audit

- Surveillez l'utilisation des clés API via les journaux d'activité
- Configurez des alertes pour des modèles d'accès inhabituels
- Passez en revue les horodatages "dernier utilisé" régulièrement

- Supprimez les clés inutilisées

## Rotation des Clés

Établissez une politique de rotation des clés :

1. **Créez une nouvelle clé** avec les mêmes permissions
2. **Mettez à jour les applications** pour utiliser la nouvelle clé
3. **Surveillez** pour vous assurer que l'ancienne clé n'est plus utilisée
4. **Révoquez l'ancienne clé** après la période de grâce

# Dépannage

**"401 Unauthorized" lors de l'utilisation de la clé API**

- **Cause :** Clé invalide, expirée ou incorrecte
- **Solution :**
  - Vérifiez que la clé est copiée correctement (pas d'espaces supplémentaires)
  - Vérifiez le statut de la clé (Actif vs. Expiré)
  - Confirmez que la clé a les permissions requises
  - Assurez-vous d'utiliser l'en-tête `X-API-KEY` (pas `Authorization`)

**"Clé API non trouvée" après création**

- **Cause :** La clé peut avoir été créée mais pas correctement stockée
- **Solution :**
  - Vérifiez la liste des clés API pour la nouvelle entrée
  - Si manquante, créez une nouvelle clé
  - Signalez le problème à l'administrateur

**Clé API expirant bientôt**

- **Cause :** Date d'expiration approchant (dans 30 jours)
- **Solution :**
  - Créez une nouvelle clé avec une expiration prolongée

- Mettez à jour les applications pour utiliser la nouvelle clé
- Révoquez l'ancienne clé après la migration

**Impossible de supprimer la clé API**

- **Cause :** Peut être protégée ou en cours d'utilisation
- **Solution :**
  - Assurez-vous d'avoir les permissions admin
  - Vérifiez si la clé est verrouillée/protégée
  - Contactez l'administrateur si le problème persiste

# Points de Terminaison API (pour la Gestion Programmatique)

Les clés API peuvent également être gérées via l'API (nécessite des permissions admin) :

**Lister les Clés API**

```
GET /crm/api-keys
Authorization: Bearer <admin-token>
```

**Créer une Clé API**

```
POST /crm/api-keys
Authorization: Bearer <admin-token>
Content-Type: application/json

{
  "name": "Nouvelle Intégration",
  "description": "Synchronisation de facturation tierce",
  "expiry_date": "2026-01-10",
  "permissions": ["view_customer", "view_service"]
}
```

**Réponse :**

```
{
  "api_key_id": 123,
  "name": "Nouvelle Intégration",
  "api_key": "sk_live_a1b2c3d4e5f6g7h8i9j0",
  "status": "active",
  "created": "2025-01-10T10:00:00Z",
  "expiry_date": "2026-01-10T23:59:59Z"
}
```

**Révoquer une Clé API**

```
DELETE /crm/api-keys/{api_key_id}
Authorization: Bearer <admin-token>
```

**Mettre à Jour une Clé API**

```
PATCH /crm/api-keys/{api_key_id}
Authorization: Bearer <admin-token>
Content-Type: application/json

{
  "name": "Nom Mis à Jour",
  "expiry_date": "2026-12-31"
}
```

# Cas d'Utilisation Courants

**Cas d'Utilisation 1 : Intégration du Système de Provisionnement**

Créez une clé API pour votre serveur de provisionnement Ansible :

1. Naviguez vers Clés API → Ajouter
2. Nom : "Serveur de Provisionnement Ansible"
3. Description : "Utilisé par l'automatisation de provisionnement"
4. Permissions : Provisionnement, Voir/Créer Services, Voir/Mettre à Jour Inventaire

5. Expiration : 1 an
6. Copiez la clé et ajoutez-la au `crm_config.yaml` d'Ansible

**Cas d'Utilisation 2 : Intégration de Facturation Tierce**

Créez une clé en lecture seule pour l'exportation de facturation :

1. Nom : "Synchronisation de Facturation - QuickBooks"
2. Permissions : Voir Clients, Voir Transactions, Voir Factures
3. Expiration : 90 jours (rotation trimestrielle)
4. Utilisez dans le script d'exportation planifié

**Cas d'Utilisation 3 : Surveillance et Alerte**

Créez une clé pour la collecte de métriques Prometheus/Grafana :

1. Nom : "Surveillance - Grafana"
2. Permissions : Voir Services, Voir Provisionnement
3. Expiration : Jamais (la surveillance nécessite un accès continu)
4. Configurez dans la source de données Grafana

**Cas d'Utilisation 4 : API du Portail Client**

Créez une clé pour le portail de service client :

1. Nom : "Backend du Portail Client"
2. Permissions : Voir Son Propre Client, Voir Ses Propres Services, Créer des Paiements
3. Expiration : 180 jours
4. Utilisez dans les appels API du backend du portail

# Documentation Connexe

- `concepts_api` - Concepts et exemples d'authentification API
- `rbac` - Contrôle d'accès basé sur les rôles et permissions
- `2fa` - Authentification à deux facteurs pour une sécurité supplémentaire

# Attributs Client

Les Attributs Client sont des paires clé-valeur flexibles qui peuvent être attachées à n'importe quel enregistrement client pour stocker des métadonnées personnalisées, des paramètres de configuration ou des informations spécifiques à l'entreprise qui ne s'intègrent pas dans les champs clients standard.

Pour une catégorisation visuelle des clients et des liens cliquables, consultez Tags Clients. Pour des informations de base sur les clients, voir Clients, Contacts, Sites & Services.

Contrairement aux champs de base de données fixes, les attributs vous permettent d'étendre dynamiquement les enregistrements clients sans modifier le schéma de la base de données. Cela les rend idéaux pour stocker des données spécifiques au déploiement, des paramètres d'intégration ou des indicateurs de logique métier personnalisée.

# Objectif et Cas d'Utilisation

Les cas d'utilisation courants pour les Attributs Client incluent :

**1. Données d'Intégration**

Stockez des identifiants de systèmes externes ou des clés API spécifiques à ce client :

- `external_crm_id` = "SF-12345" (ID client Salesforce)
- `legacy_system_id` = "OLD-CRM-789" (Référence de migration)
- `hubspot_contact_id` = "12345678" (Intégration HubSpot)

**2. Logique Métier Personnalisée**

Stockez des indicateurs ou des paramètres qui contrôlent le comportement spécifique au client :

- `billing_method` = "quarterly" (Remplacer la facturation mensuelle par défaut)
- `auto_provision` = "true" (Activer la provision automatique des services)
- `support_tier` = "premium" (Niveau de support personnalisé)
- `credit_limit` = "10000" (Limite de crédit spécifique au client)

**3. Données de Conformité et Réglementaires**

Suivez les métadonnées liées à la conformité :

- `gdpr_consent_date` = "2025-01-01" (Consentement au traitement des données)
- `tax_exempt` = "true" (Statut d'exemption fiscale)
- `regulatory_entity` = "FCC-123456" (Identifiant réglementaire)

**4. Métadonnées Opérationnelles**

Stockez des informations opérationnelles :

- `preferred_contact_method` = "email" (Préférence de communication)
- `account_manager` = "<john.smith@company.com>" (Responsable de compte assigné)
- `onboarding_date` = "2025-01-15" (Suivi du cycle de vie du client)
- `churn_risk_score` = "0.23" (Analytique prédictive)

**5. Paramètres de Provisioning**

Stockez des configurations spécifiques au provisioning :

- `radius_username_format` = "email" (Format RADIUS personnalisé)
- `vlan_id` = "100" (Configuration réseau)
- `ipv6_enabled` = "true" (Indicateurs de fonctionnalités)

# Attributs vs. Champs Standards

**Utilisez des Attributs Quand :**

- Les données sont spécifiques au déploiement ou varient selon l'installation
- Les exigences changent fréquemment
- Stockage de métadonnées spécifiques à l'intégration
- Prototypage de nouvelles fonctionnalités avant d'ajouter des champs de base de données
- Les données n'ont pas besoin de requêtes complexes ou de jointures

**Utilisez des Champs Standards Quand :**

- Les données sont essentielles au modèle client (nom, email, adresse)
- Recherche, filtrage ou reporting fréquents requis
- Les données ont des contraintes d'intégrité référentielle
- La performance est critique pour des requêtes à grande échelle

# Gestion des Attributs via l'UI

## Affichage des Attributs Client

Pour afficher les attributs d'un client :

1. Accédez à la page d'aperçu du client
2. Cliquez sur l'onglet **Attributs**
3. Vous verrez un tableau de tous les attributs pour ce client, montrant :
    - Nom de l'attribut (clé)
    - Valeur de l'attribut
    - Date de création
    - Date de dernière modification

## Création d'un Nouvel Attribut

Pour créer un nouvel attribut pour un client :

1. Accédez à la page d'aperçu du client
2. Cliquez sur l'onglet **Attributs**
3. Cliquez sur le bouton **Ajouter un Attribut**

4. Remplissez les champs requis :
    - **Nom de l'Attribut** (requis) : La clé/nome pour cet attribut (par exemple, `external_crm_id`)
    - **Valeur de l'Attribut** (requis) : La valeur à stocker (par exemple, `SF-12345`)
5. Cliquez sur **Créer l'Attribut**

**Conventions de Nommage :**

- Utilisez des minuscules avec des underscores : `external_system_id` ✓
- Évitez les espaces : `external system id` ✗
- Gardez les noms descriptifs mais concis
- Utilisez une nomenclature cohérente entre les clients pour les mêmes types d'attributs

# Édition d'un Attribut

Pour modifier un attribut existant :

1. Accédez à la page d'aperçu du client
2. Cliquez sur l'onglet **Attributs**
3. Trouvez l'attribut que vous souhaitez modifier dans le tableau
4. Cliquez sur le bouton **Modifier** (crayon)
5. Modifiez le nom ou la valeur de l'attribut
6. Cliquez sur **Mettre à Jour l'Attribut**

> ⓘ **REMARQUE**
>
> Changer le nom d'un attribut crée une nouvelle paire clé-valeur. Assurez-vous que cela ne casse pas les intégrations qui dépendent du nom d'attribut original.

# Suppression d'un Attribut

Pour supprimer un attribut :

1. Accédez à la page d'aperçu du client

2. Cliquez sur l'onglet **Attributs**
3. Trouvez l'attribut que vous souhaitez supprimer dans le tableau
4. Cliquez sur le bouton **Supprimer** (corbeille)
5. Confirmez la suppression dans la fenêtre contextuelle

> ⚠ **ATTENTION**
>
> La suppression d'attributs utilisés par des intégrations, des flux de travail de provisioning ou de la logique de facturation peut entraîner des échecs. Vérifiez les dépendances avant la suppression.

# Référence des Champs d'Attribut

# Intégration API

Les attributs peuvent être gérés par programmation via l'API :

## Créer ou Mettre à Jour un Attribut

**Point de terminaison :** `PUT /crm/attribute/`

**Autorisation Requise :** `create_customer_attribute`

**Corps de la Demande :**

```
{
  "customer_id": 123,
  "attribute_name": "external_crm_id",
  "attribute_value": "SF-12345"
}
```

**Réponse :**

```
{
  "attribute_id": 456,
  "customer_id": 123,
  "attribute_name": "external_crm_id",
  "attribute_value": "SF-12345",
  "created": "2025-01-04 10:30:00",
  "last_modified": "2025-01-04 10:30:00"
}
```


## Mettre à Jour un Attribut Existant

**Point de terminaison :** `PATCH /crm/attribute/attribute_id/{attribute_id}`

**Autorisation Requise :** `update_customer_attribute`

**Corps de la Demande :**


```
{
  "attribute_value": "SF-54321"
}
```


## Obtenir un Attribut par ID

**Point de terminaison :** `GET /crm/attribute/attribute_id/{attribute_id}`

**Autorisation Requise :** `view_customer_attribute`

**Réponse :**

```
{
  "attribute_id": 456,
  "customer_id": 123,
  "attribute_name": "external_crm_id",
  "attribute_value": "SF-12345",
  "created": "2025-01-04 10:30:00",
  "last_modified": "2025-01-04 10:30:00"
}
```


## Obtenir Tous les Attributs par ID Client

**Point de terminaison :** `GET /crm/attribute/customer_id/{customer_id}`

**Autorisation Requise :** `view_customer_attribute`

**Réponse :**


```
[
  {
    "attribute_id": 456,
    "customer_id": 123,
    "attribute_name": "external_crm_id",
    "attribute_value": "SF-12345",
    "created": "2025-01-04 10:30:00",
    "last_modified": "2025-01-04 10:30:00"
  },
  {
    "attribute_id": 457,
    "customer_id": 123,
    "attribute_name": "billing_method",
    "attribute_value": "quarterly",
    "created": "2025-01-04 10:35:00",
    "last_modified": "2025-01-04 10:35:00"
  }
]
```

## Supprimer un Attribut

**Point de terminaison :** `DELETE /crm/attribute/attribute_id/{attribute_id}`

**Autorisation Requise :** `delete_customer_attribute`

**Réponse :**

```
{
  "result": "success"
}
```

# Opérations d'Attributs en Masse

## Gestion de Plusieurs Attributs

Pour définir plusieurs attributs pour un client en une seule fois (par exemple, lors de l'intégration ou de la synchronisation) :

```python
import requests

customer_id = 123
attributes = [
    {"attribute_name": "external_crm_id", "attribute_value": "SF-
12345"},
    {"attribute_name": "billing_method", "attribute_value":
"quarterly"},
    {"attribute_name": "support_tier", "attribute_value":
"premium"}
]

for attr in attributes:
    attr["customer_id"] = customer_id
    requests.put(
        "https://api.example.com/crm/attribute/",
        json=attr,
        headers={"Authorization": "Bearer YOUR_TOKEN"}
    )
```

## Interroger les Clients par Attribut

Bien que les attributs n'aient pas de points de terminaison de recherche intégrés, vous pouvez filtrer les clients par attribut en utilisant l'API de recherche de clients avec un filtrage personnalisé :

```
# Obtenez tous les clients, puis filtrez par attribut dans le code de
l'application
customers = requests.get("https://api.example.com/crm/customer/").jsc

for customer in customers:
    attributes = requests.get(

f"https://api.example.com/crm/attribute/customer_id/{customer['custom
    ).json()

    # Trouvez les clients avec un attribut spécifique
    for attr in attributes:
        if attr['attribute_name'] == 'support_tier' and
attr['attribute_value'] == 'premium':
            print(f"Client Premium : {customer['customer_name']}")
```

> **REMARQUE**
>
> Pour des requêtes fréquentes basées sur les attributs, envisagez d'ajouter des champs de base de données indexés ou de mettre en œuvre un point de terminaison de recherche dédié.

# Meilleures Pratiques

### 1. Conventions de Nommage

- Utilisez snake_case : `external_system_id` ✓
- Soyez descriptif : `billing_method` ✓ vs `method` ✗
- Évitez les mots-clés réservés ou les caractères spéciaux
- Documentez les significations des attributs dans votre guide de déploiement

### 2. Types de Données

- Les attributs stockent des valeurs sous forme de chaînes (max 150 caractères)
- Pour les booléens, utilisez "true"/"false" (minuscules)

- Pour les dates, utilisez le format ISO 8601 : "2025-01-04 10:30:00"
- Pour de grandes données JSON, envisagez d'utiliser des champs de base de données dédiés à la place

**3. Validation**

- Validez les valeurs des attributs dans le code de l'application avant de les enregistrer
- Utilisez des formats de valeur cohérents entre les clients
- Documentez les valeurs attendues pour chaque nom d'attribut

**4. Documentation**

- Maintenez un registre des noms et des objectifs des attributs
- Documentez quels systèmes/intégrations dépendent d'attributs spécifiques
- Incluez des exemples de valeurs valides

**5. Migration et Nettoyage**

- Auditez régulièrement les attributs inutilisés
- Supprimez les attributs obsolètes après les migrations système
- Versionnez les noms d'attributs lors des changements de schémas (par exemple, `api_key_v2`)

# Exemples de Flux de Travail

## Intégration d'Onboarding

Lors de la migration de clients depuis un système hérité :

```
# Stockez la référence du système hérité pour le débogage
PUT /crm/attribute/
{
  "customer_id": 123,
  "attribute_name": "legacy_crm_id",
  "attribute_value": "OLD-12345"
}

# Suivez la date de migration
PUT /crm/attribute/
{
  "customer_id": 123,
  "attribute_name": "migrated_date",
  "attribute_value": "2025-01-04"
}
```

## Règles de Facturation Personnalisées

Remplacez le cycle de facturation par défaut pour un client spécifique :

```
# Définir la facturation trimestrielle
PUT /crm/attribute/
{
  "customer_id": 123,
  "attribute_name": "billing_cycle",
  "attribute_value": "quarterly"
}

# Puis dans le code de facturation, vérifiez l'attribut avant de
traiter
attributes = GET /crm/attribute/customer_id/123
billing_cycle = next(
    (a['attribute_value'] for a in attributes if
a['attribute_name'] == 'billing_cycle'),
    'monthly'  # par défaut
)
```

## Indicateurs de Fonctionnalités

Activez les fonctionnalités bêta pour des clients spécifiques :

```
# Activer le provisioning IPv6
PUT /crm/attribute/
{
  "customer_id": 123,
  "attribute_name": "feature_ipv6_enabled",
  "attribute_value": "true"
}
```

# Permissions

Les opérations sur les attributs nécessitent les permissions suivantes :

- `view_customer_attribute` - Voir les attributs
- `create_customer_attribute` - Créer de nouveaux attributs
- `update_customer_attribute` - Modifier les attributs existants
- `delete_customer_attribute` - Supprimer des attributs

Voir `rbac` pour la configuration de contrôle d'accès basé sur les rôles.

# Dépannage

#### Attribut Non Apparaissant dans l'UI

- Vérifiez que l'attribut a été créé (vérifiez la réponse de l'API)
- Rafraîchissez la page pour recharger les données client
- Vérifiez que l'utilisateur a la permission `view_customer_attribute`

#### Impossible de Mettre à Jour l'Attribut

- Assurez-vous que vous avez la permission `update_customer_attribute`
- Vérifiez que l'attribute_id est correct
- Vérifiez que l'attribut appartient au client spécifié

### Échec de l'Intégration Après la Suppression d'un Attribut

- Restaurez l'attribut avec la valeur précédente
- Mettez à jour le code d'intégration pour gérer les attributs manquants de manière élégante
- Auditez les dépendances des attributs avant la suppression

### Valeur d'Attribut Troncée

- Les valeurs des attributs ont une limite de 150 caractères
- Pour des données plus longues, divisez en plusieurs attributs ou utilisez le champ des notes client
- Envisagez de stocker de grandes données dans des champs de base de données dédiés

# Guide de Configuration et de Personnalisation

Ce guide complet couvre toutes les options de configuration et de personnalisation pour OmniCRM, y compris les paramètres backend, le branding frontend, la personnalisation visuelle et les meilleures pratiques de déploiement.

# Aperçu

OmniCRM utilise deux systèmes de configuration principaux :

**Configuration Backend :**

- **Fichier :** `OmniCRM-API/crm_config.yaml`
- **Format :** YAML
- **Nécessite :** Redémarrage de l'API après modifications
- **Utilisé pour :** Base de données, intégrations, sécurité, approvisionnement

**Configuration Frontend (runtime, production) :**

- **Source :** `crm_config.yaml` → bloc `ui:` servi par `GET /crm/public/ui-config`
- **Format :** JSON à l'exécution (YAML dans la configuration de déploiement)
- **Nécessite :** Rechargement de l'API / mise à jour du montage de configuration uniquement : **pas de reconstruction d'image UI**
- **Utilisé pour :** Branding, thème, locale, tableaux de bord Grafana, raccourcis d'application web, drapeaux de fonctionnalités

**Configuration Frontend (fallback de développement local) :**

- **Fichier :** `OmniCRM-UI/.env` (voir `.env.example`)
- **Format :** variables d'environnement `VITE_*`
- **Nécessite :** Redémarrage du serveur de développement UI

- **Utilisé lorsque :** `/crm/public/ui-config` est indisponible ; `runtimeConfig.js` se rabat sur ces valeurs

L'image Docker générique `omnicrm-ui` lit le branding à l'exécution. Les images UI par client et les arguments de construction `VITE_*` sont obsolètes pour les déploiements en production.

---

# Configuration Backend (crm_config.yaml)

Le fichier `crm_config.yaml` contient tous les paramètres du système backend.

**Emplacement :** `/OmniCRM/OmniCRM-API/crm_config.yaml`

## Configuration de la Base de Données

```yaml
database:
  username: omnitouch
  password: omnitouch2024
  server: localhost
```

**Champs :**

- `username` - Nom d'utilisateur de la base de données MySQL
- `password` - Mot de passe de la base de données MySQL
- `server` - Nom d'hôte ou IP du serveur de base de données (par défaut : localhost)

**Connexion à la Base de Données :**

- Le nom de la base de données est codé en dur comme `omnicrm`
- Port par défaut : 3306 (par défaut MySQL)
- Chaîne de connexion : `mysql+pymysql://username:password@server/omnicrm`

**Remarque de Sécurité :** Ne jamais commettre ce fichier avec de vraies informations d'identification dans le contrôle de version. Utilisez une configuration spécifique à l'environnement ou une gestion des secrets.

**Remarque :** Cela doit correspondre à vos informations d'identification de base de données `.env`.

## Types de Services

```
service_types:
  - omnicharge
  - mobile
  - fixed
  - fixed-voice
  - hotspot
  - dongle
```

**But :** Définit les types de services valides pour votre déploiement.

Ceux-ci sont utilisés dans tout le système pour :

- Catégorisation des produits
- Filtrage des addons (les addons correspondent aux types de services)
- Flux de travail d'approvisionnement
- Reporting et analyses

Ajoutez des types de services personnalisés ici pour vos cas d'utilisation spécifiques.

## Intégration HSS (Home Subscriber Server)

Pour les opérateurs de réseau mobile avec intégration HSS :

```
hss:
  hss_peers:
    - 'http://10.179.2.140:8080'
  apn_list: "1,2,3,4,5,6"
```

**Configuration :**

- `hss_peers` - Liste des points de terminaison HSS pour l'approvisionnement des abonnés
- `apn_list` - Liste d'IDs APN séparés par des virgules disponibles pour l'approvisionnement

**Utilisé pour :** Approvisionnement mobile et authentification des abonnés.

# Configuration du Modèle d'Email Mailjet

OmniCRM utilise Mailjet pour les emails transactionnels. Chaque type d'email a sa propre configuration de modèle :

```yaml
mailjet:
  api_key: your_mailjet_api_key
  api_secret: your_mailjet_api_secret

  # Email de Bienvenue Client
  api_crmCommunicationCustomerWelcome:
    from_email: "support@yourcompany.com"
    from_name: "Support de Votre Entreprise"
    template_id: 5977509
    subject: "Bienvenue chez YourCompany"

  # Email de Facture Client
  api_crmCommunicationCustomerInvoice:
    from_email: "billing@yourcompany.com"
    from_name: "Facturation de Votre Entreprise"
    template_id: 6759851
    subject: "Votre Facture - "

  # Rappel de Facture
  api_crmCommunicationCustomerInvoiceReminder:
    from_email: "billing@yourcompany.com"
    from_name: "Facturation de Votre Entreprise"
    template_id: 5977570
    subject: "Rappel de Paiement de Facture"

  # Email de Bienvenue Utilisateur (Personnel/Admin)
  api_crmCommunicationUserWelcome:
    from_email: "admin@yourcompany.com"
    from_name: "Admin de Votre Entreprise"
    template_id: 6118112
    subject: "Bienvenue dans l'Équipe"

  # Demande de Réinitialisation de Mot de Passe
  api_crmCommunicationUserPasswordReset:
    from_email: "security@yourcompany.com"
    from_name: "Sécurité de Votre Entreprise"
    template_id: 6735666
    subject: "Demande de Réinitialisation de Mot de Passe"

  # Confirmation de Succès de Réinitialisation de Mot de Passe
  api_crmCommunicationUserPasswordResetSuccess:
    from_email: "security@yourcompany.com"
    from_name: "Sécurité de Votre Entreprise"
```

```
    template_id: 6118378
    subject: "Réinitialisation de Mot de Passe Réussie"

  # Notification de Changement de Mot de Passe
  api_crmCommunicationUserPasswordChange:
    from_email: "security@yourcompany.com"
    from_name: "Sécurité de Votre Entreprise"
    template_id: 6118423
    subject: "Mot de Passe Changé"

  # Vérification de l'Email
  api_crmCommunicationEmailVerification:
    from_email: "verify@yourcompany.com"
    from_name: "Vérification de Votre Entreprise"
    template_id: 6267350
    subject: "Vérifiez Votre Adresse Email"

  # Notification d'Équilibre Expiré
  api_crmCommunicationsBalanceExpired:
    from_email: "alerts@yourcompany.com"
    from_name: "Alertes de Votre Entreprise"
    template_id: 7238252
    subject: "Équilibre de Service Expiré"

  # Alerte de Faible Équilibre
  api_crmCommunicationsBalanceLow:
    from_email: "alerts@yourcompany.com"
    from_name: "Alertes de Votre Entreprise"
    template_id: 7238263
    subject: "Alerte de Faible Équilibre"

  # Email de Livraison eSIM
  # Définissez template_id sur un modèle Mailjet valide pour
utiliser un modèle personnalisé,
  # ou laissez à 0 pour utiliser l'email HTML de marque intégré
  api_crmCommunicationEsimDelivery:
    from_email: "support@yourcompany.com"
    from_name: "Support de Votre Entreprise"
    template_id: 0
    subject: "Votre eSIM est Prête"
```

**Email de Livraison eSIM**

Envoyé lorsqu'une eSIM est approvisionnée et prête à être activée.

```
api_crmCommunicationEsimDelivery:
  from_email: "support@yourcompany.com"
  from_name: "Support de Votre Entreprise"
  template_id: 0
  subject: "Votre eSIM est Prête"
```

**Quand Envoyé :**

- eSIM approvisionnée lors de l'auto-inscription
- eSIM assignée à un service client

**Variables de Modèle Disponibles (lors de l'utilisation d'un modèle Mailjet) :**

- `{{ var:customer_name }}` - Nom complet du client
- `{{ var:iccid }}` - ICCID de l'eSIM
- `{{ var:activation_code }}` - Chaîne de code d'activation LPA
- `{{ var:has_qr_code }}` - Booléen, vrai si l'image du code QR est jointe
- `{{ var:has_activation_code }}` - Booléen, vrai si le code d'activation est disponible

**Fallback Intégré :**

Lorsque `template_id` est défini sur `0`, OmniCRM envoie un email HTML de marque intégré avec le code QR de l'eSIM en ligne. Le branding de cet email est contrôlé par la section `branding` (voir ci-dessous).

# Branding (Modèles d'Email Intégrés)

Lorsque OmniCRM envoie des emails en utilisant ses modèles HTML intégrés (plutôt que des modèles Mailjet), la section `branding` contrôle l'apparence visuelle.

```yaml
branding:
  brand_name: 'YourCompany'
  primary_dark: '#594712'
  dark_color: '#32280B'
  accent_bg: '#f5f0e0'
```

**Champs :**

| Clé | Description | Par défaut | Exemple |
|---|---|---|---|
| `brand_name` | Nom de l'entreprise affiché dans le sujet de l'email, la salutation et la signature | Dérivé de `from_name` | `'Example Mobile'` |
| `primary_dark` | Couleur principale de la marque utilisée pour les titres et les bordures d'accent | `#594712` | `'#1a73e8'` |
| `dark_color` | Couleur sombre utilisée dans le dégradé d'en-tête de l'email | `#32280B` | `'#0d47a1'` |
| `accent_bg` | Couleur de fond claire pour les boîtes d'appel d'information | `#f5f0e0` | `'#e3f2fd'` |

**Actuellement Utilisé Par :**

- Email de livraison eSIM (lorsque `api_crmCommunicationEsimDelivery.template_id` est `0`)

**Création de Modèles Mailjet :**

1. Connectez-vous au tableau de bord Mailjet (<https://app.mailjet.com>)
2. Accédez à **Transactionnel → Modèles**
3. Créez un nouveau modèle ou clonez un modèle existant

4. Notez l'**ID du Modèle** (valeur numérique)
5. Ajoutez des variables de modèle correspondant à la structure de données d'OmniCRM
6. Mettez à jour `crm_config.yaml` avec l'ID du modèle

**Variables de Modèle Disponibles :**

Chaque type d'email reçoit des variables spécifiques. Exemples courants :

- `{{customer_name}}` - Nom du client ou de l'utilisateur
- `{{service_name}}` - Nom du service ou du produit
- `{{invoice_id}}` - Numéro de la facture
- `{{invoice_amount}}` - Montant total de la facture
- `{{due_date}}` - Date d'échéance du paiement
- `{{reset_link}}` - URL de réinitialisation de mot de passe
- `{{verification_link}}` - URL de vérification de l'email
- `{{balance}}` - Solde actuel du compte
- `{{expiry_date}}` - Date d'expiration du solde ou du service

# Configuration d'Approvisionnement

```yaml
provisioning:
  failure_list: ['admin@yourcompany.com', 'ops@yourcompany.com']
```

**But :**

- `failure_list` - Adresses email notifiées lorsque l'approvisionnement Ansible échoue
- Les notifications incluent le nom du playbook, les détails de l'erreur et les informations client
- Permet à l'équipe des opérations de répondre rapidement aux problèmes d'approvisionnement

## Configuration de Facture

```yaml
invoice:
  template_filename: 'yourcompany_invoice_template.html'
```

**But :**

Spécifie quel modèle HTML Jinja2 utiliser pour la génération de factures PDF.

**Emplacement du Modèle :** `/OmniCRM-API/invoice_templates/`

Voir la section **Génération de PDF de Facture** ci-dessous pour des détails sur la création de modèles personnalisés.

## URL de Base CRM

```yaml
crm:
  base_url: 'http://localhost:5000'
```

**But :**

- Utilisé par les playbooks Ansible pour effectuer des rappels API
- Utilisé dans les modèles d'email pour générer des liens vers le CRM
- Doit être l'URL accessible publiquement de votre API

**Exemples :**

- Développement : `http://localhost:5000`
- Production : `https://api.yourcompany.com`

**Important :**

- Ne pas inclure de barre oblique à la fin
- Utilisez une URL accessible publiquement si les playbooks s'exécutent sur des serveurs différents
- Mettez à jour lors du déploiement en production

## Configuration OCS et CGRates

```
ocs:
  ocsApi: 'http://10.179.2.142:8080/api'
  ocsTenant: 'mnc001.mcc001.3gppnetwork.org'
  cgrates: 'localhost:2080'
```

**Configuration :**

- `ocsApi` - Point de terminaison API OCS pour la gestion des abonnés
- `ocsTenant` - Identifiant de locataire pour les déploiements OCS multi-locataires
- `cgrates` - Point de terminaison API JSON-RPC CGRates (hôte:port)

**Utilisé pour :** Facturation en temps réel, gestion des soldes, suivi de l'utilisation.

## Configuration SMSC (Passerelle SMS)

```
smsc:
  source_msisdn: 'YourCompany'
  smsc_url: 'http://10.179.2.216/SMSc/'
  api_key: 'your_smsc_api_key'
```

**But :**

- Envoyer des notifications SMS aux clients (faible solde, alertes de service, codes 2FA)
- `source_msisdn` - ID de l'expéditeur affiché aux destinataires (alphanumérique ou numéro de téléphone)
- `smsc_url` - Point de terminaison API de la passerelle SMSC
- `api_key` - Authentification pour l'API SMSC

## Configuration de Diffusion Cellulaire

```
cbc_url: 'http://10.179.1.113:8080'
```

**But :** Point de terminaison API du Centre de Diffusion Cellulaire (CBC) pour les alertes d'urgence.

Voir `features_cell_broadcast` pour les détails d'utilisation.

## Clé Secrète JWT

```
jwt_secret: 'CHANGE_ME_ON_DEPLOYMENT'  # Auto-généré par Ansible
```

**Sécurité :**

- Utilisé pour signer et vérifier les jetons d'authentification (JWT)
- **Critique pour la sécurité de l'authentification** - traiter comme un mot de passe
- Chaque déploiement/environnement DOIT avoir un secret unique
- Ne jamais partager ou commettre dans le contrôle de version

**Gestion Automatisée des Secrets (Déploiement Ansible) :**

Lors du déploiement avec Ansible, le secret JWT est géré automatiquement :

1. **Premier déploiement :** Ansible génère une chaîne hexadécimale aléatoire de 64 caractères (256 bits d'entropie)
2. **Déploiements suivants :** Ansible préserve le secret existant du serveur
3. **Chaque serveur obtient son propre secret unique** - les sessions ne fonctionnent pas entre les environnements

Cela garantit :

- Pas de secrets codés en dur dans les dépôts git
- Chaque environnement est cryptographiquement isolé

- Les secrets persistent entre les déploiements (les sessions ne sont pas invalidées lors du redéploiement)
- La reconstruction d'un serveur génère un nouveau secret (les sessions sont correctement invalidées)

**Génération Manuelle de Secrets (Déploiements Non-Ansible) :**

Si vous déployez sans Ansible, vous DEVEZ générer manuellement un secret unique :

```
# Générer une clé aléatoire sécurisée sur le plan cryptographique
python3 -c "import secrets; print(secrets.token_hex(32))"
# ou
openssl rand -hex 32
```

Puis mettez à jour `crm_config.yaml` :

```
jwt_secret: 'your_generated_secret_here'
```

**Important :**

- **Ne jamais utiliser la valeur par défaut** `CHANGE_ME_ON_DEPLOYMENT` en production
- **Ne jamais partager publiquement** - quiconque possède le secret peut falsifier des jetons d'authentification
- Changer cela invalide toutes les sessions utilisateur existantes (les utilisateurs doivent se reconnecter)
- Chaque environnement (dev, staging, prod) DOIT avoir des secrets différents

# Configuration de Paiement Stripe

```
stripe:
  secret_key: 'sk_live_xxxxxxxxxx'
  publishable_key: 'pk_live_xxxxxxxxxx'
  currency: 'aud'
  statement_descriptor_suffix: 'YOURCOMPANY'
```

**Configuration :**

- `secret_key` - Clé API secrète Stripe (côté serveur, à garder confidentielle)
- `publishable_key` - Clé publiable Stripe (côté client, sûr à exposer)
- `currency` - Code de devise ISO 4217 (aud, usd, gbp, eur, etc.)
- `statement_descriptor_suffix` - Apparaît sur les relevés de carte de crédit des clients

**Types de Clés :**

- Clés de test : `sk_test_...` et `pk_test_...` (pour le développement)
- Clés en direct : `sk_live_...` et `pk_live_...` (pour la production)

**Utilisation du Descripteur de Relevé :**

- Affiché sur les relevés bancaires des clients comme "YOURCOMPANY"
- Maximum 22 caractères
- Aide les clients à identifier les frais
- Utilisé également dans les noms de fichiers PDF de factures (par exemple, `YOURCOMPANY_12345.pdf`)

Voir *Intégrations de Fournisseurs de Paiement* pour les détails de configuration.

# Clés API et Liste Blanche IP

Définir des clés API avec accès basé sur les rôles et des restrictions IP :

```yaml
api_keys:
  "YOUR_API_KEY_1":
    roles: ["admin"]
    ips: ["127.0.0.1", "::1"]
  "YOUR_API_KEY_2":
    roles: ["customer_service_agent_1"]
    ips: ["127.0.0.1", "::1", "10.0.1.0/24"]

# Liste Blanche IP (autonome, sans clé API)
ip_whitelist:
  "10.179.2.142":
    roles: ["admin"]
```

**But :**

- Permettre aux systèmes externes de s'authentifier via la clé API
- Restreindre l'accès par adresse IP
- Accorder des rôles spécifiques aux consommateurs d'API
- Utile pour les intégrations (systèmes de facturation, surveillance, automatisation)

**Génération de Clés API :**

```
openssl rand -base64 48
```

**Rôles :**

- `admin` - Accès complet à tous les points de terminaison
- Rôles personnalisés définis dans le système RBAC

**Meilleures Pratiques de Sécurité :**

- Utilisez des clés API longues et aléatoires (minimum 32 caractères)
- Restreindre les IP à des sources connues uniquement
- Accorder les rôles minimaux nécessaires
- Faire tourner les clés API régulièrement
- Surveiller l'utilisation des clés API dans les journaux

**Avertissement de Sécurité :**

- N'utilisez la liste blanche IP que pour des réseaux internes de confiance
- Ne pas utiliser d'IP localhost (127.0.0.1, ::1)
- Utilisez plutôt des clés API pour l'accès externe
- Envisagez des règles de pare-feu comme protection supplémentaire

Voir `administration_api_keys` et `concepts_api` pour des détails.

---

# Configuration UI à l'exécution (`ui:` dans crm_config.yaml)

Les déploiements en production utilisent une image `omnicrm-ui` **générique**. Le branding, le thème et les bascules de fonctionnalités sont chargés à l'exécution depuis l'API :

- **Point de terminaison :** `GET /crm/public/ui-config` (supporté par ETag ; l'UI met en cache dans `localStorage`)
- **Actifs :** logos, favicons et icônes d'application web sous `ui.assets_dir`, servis depuis `/crm/public/assets/...`
- **Bloc `branding:` hérité :** toujours utilisé pour les modèles d'email ; `ui.branding` et `ui.theme` sont la source de vérité de l'UI

Exemple de bloc `ui:` (voir aussi `OmniCRM-API/crm_config.yaml`) :

```yaml
ui:
  version: '1'
  assets_dir: '/etc/omnicrm/deployment_assets'
  branding:
    client_name: 'Example Mobile'
    portal_name: 'Omnitouch'
    self_care_name: 'Example Mobile'
    logos:
      sm: logos/logoSm.png
      dark: logos/logoDark.png
      light: logos/logoLight.png
      favicon: logos/favicon.ico
  theme:
    primary: '#0D0629'
    secondary: '#CBB42B'
    font_family: Quicksand
  locale:
    default_language: en
    locale: en-GB
    default_country: Australia
  integrations:
    google_maps_api_key: ''
    google_tag_id: ''
    grafana:
      dashboard_ids: 'id1,id2'
      dashboard_labels: 'Ops,Finance'
  features:
    cbc_mode: no
    simplified_addresses: no
  links:
    faqs_url: 'https://example.com/faqs'
    support_url: 'https://example.com/support'
  web_apps:
    - name: GitHub
      url: https://github.com
      icon_path: webapp_icons/github.png
```

Les logos Omnitouch par défaut sont **livrés avec le package** `omnicrm-api` sous `/usr/share/omnicrm/deployment_assets/logos/`. Les déploiements sans superposition `deployment_assets` de client servent toujours le branding à partir de ces valeurs par défaut. Les remplacements par client : synchronisez vers `/etc/omnicrm/deployment_assets` (Ansible copie

`group_vars/deployment_assets/` et/ou `logo_*`) et référencez les chemins dans `ui.branding.logos`. Définissez `ui.assets_dir: '/etc/omnicrm/deployment_assets'`, jamais `/etc/omnicrm` lui-même. Si ce chemin est manquant (Docker), l'API se rabat sur `UI_ASSETS_DIR` puis `/app/deployment_assets`. `GET /crm/public/assets/` ne sert que des fichiers image sous `logos/`, `login/`, et `webapp_icons/`.

**Migration depuis `VITE_*` :** mappez chaque ancien argument de construction au champ `ui:` correspondant (voir les commentaires de `.env.example`). Local `npm run dev` lit toujours `VITE_*` via `runtimeConfig.js` lorsque le point de terminaison API est inaccessible.

---

# Configuration Frontend (.env) : fallback de développement local

**Emplacement :** `/OmniCRM/OmniCRM-UI/.env`

## Clés API et Services Externes

```
# Google Maps API (pour l'autocomplétion d'adresse et la
géocodage)
VITE_GOOGLE_API_KEY=your_google_api_key

# Désactiver le lancement automatique du navigateur sur npm start
BROWSER=none
```

## Configuration des Fournisseurs de Paiement

```
# Fournisseurs de paiement activés (séparés par des virgules) :
stripe, paypal, ou les deux
VITE_PAYMENT_VENDORS="stripe,paypal"

# Clé publiable Stripe (requise si Stripe est activé)
VITE_STRIPE_PUBLISHABLE_KEY=pk_test_xxxxx

# ID client PayPal (requise si PayPal est activé)
VITE_PAYPAL_CLIENT_ID=YOUR_PAYPAL_CLIENT_ID
```

**Doit Correspondre au Backend :**

- `VITE_STRIPE_PUBLISHABLE_KEY` doit correspondre à `stripe.publishable_key` dans `crm_config.yaml`
- `VITE_PAYPAL_CLIENT_ID` doit correspondre à `paypal.client_id` dans `crm_config.yaml`

**Important :** Ces variables sont **à l'heure de la construction**. Changer une valeur `VITE_*` nécessite de reconstruire et de redémarrer l'UI. Voir Guide du Système de Paiement - Configuration du Fournisseur pour la procédure complète de configuration y compris la configuration backend.

## Branding et Informations sur l'Entreprise

```
# Branding de l'Entreprise
VITE_COMPANY_NAME="ShellFone"
VITE_PORTAL_NAME="ShellManager"
VITE_SELF_CARE_NAME="ShellCare"
VITE_COMPANY_TAGLINE="Phones with Shells"
```

Ces valeurs apparaissent dans toute l'UI :

- `COMPANY_NAME` - Affiché dans les titres de page, les emails et les communications avec les clients
- `PORTAL_NAME` - Nom du portail admin/personnel (par exemple, "ShellManager")

- `SELF_CARE_NAME` - Nom du portail de service client (par exemple, "ShellCare")
- `COMPANY_TAGLINE` - Apparaît dans les écrans de connexion et les supports marketing

## Localisation et Paramètres Régionaux

```
# Langue et Locale
# Langues supportées : ar, ch, en, fr, gr, it, ru, sp
VITE_DEFAULT_LANGUAGE=en
VITE_LOCALE="en-GB"

# Emplacement par Défaut (pour l'autocomplétion d'adresse)
VITE_DEFAULT_LOCATION="Sydney, Australia"
VITE_DEFAULT_COUNTRY="Australia"

# Paramètres de Devise
VITE_CURRENCY_CODE="GBP"
VITE_CURRENCY_SYMBOL="£"
```

**Langues Supportées :**

- `ar` - Arabe
- `ch` - Chinois
- `en` - Anglais (par défaut)
- `fr` - Français
- `gr` - Grec
- `it` - Italien
- `ru` - Russe
- `sp` - Espagnol

**Devises Courantes :**

- USD - $ (Dollar US)
- GBP - £ (Livre Sterling)
- EUR - € (Euro)
- AUD - $ (Dollar Australien)

- CAD - $ (Dollar Canadien)

**Remarque :** Doit correspondre à `stripe.currency` dans `crm_config.yaml`.

# Configuration du Thème de Couleur

```
# Couleur Principale (couleur principale de la marque)
VITE_PRIMARY_COLOR=#405189

# Options de Couleur Supplémentaires (exemples commentés)
# VITE_SECONDARY_COLOR=#2bFFcf
# VITE_TERTIARY_COLOR=#1a9fbf
# VITE_SUCCESS_COLOR=#28a745
# VITE_INFO_COLOR=#17a2b8
# VITE_WARNING_COLOR=#ffc107
# VITE_DANGER_COLOR=#dc3545
```

**Couleurs Disponibles :**

- `VITE_PRIMARY_COLOR` - Couleur principale de la marque (boutons, liens, surlignages)
- `VITE_SECONDARY_COLOR` - Couleur d'accent
- `VITE_TERTIARY_COLOR` - Accent supplémentaire
- `VITE_SUCCESS_COLOR` - Messages de succès (#28a745)
- `VITE_INFO_COLOR` - Messages d'information (#17a2b8)
- `VITE_WARNING_COLOR` - Avertissements (#ffc107)
- `VITE_DANGER_COLOR` - Erreurs (#dc3545)
- `VITE_LIGHT_COLOR` - Arrière-plans clairs (#f8f9fa)
- `VITE_DARK_COLOR` - Texte sombre (#343a40)
- `VITE_PRIMARY_DARK_COLOR` - Variante sombre de la couleur primaire

**Format :** Codes de couleur hexadécimaux (#RRGGBB)

La couleur primaire est appliquée à :

- En-têtes de navigation
- Boutons d'action

- Liens et surlignages
- États actifs
- Éléments de marque

## Configuration de la Police

```
# Configuration de la Famille de Polices
# Sans-Serif : Inter, Roboto, Open Sans, Lato, Quicksand, Poppins,
Nunito, Montserrat, Work Sans, Source Sans Pro, Raleway, Ubuntu,
Josefin Sans, HKGrotesk
# Serif : Merriweather, Lora, Playfair Display
# Système : Système (polices natives de l'appareil)
# Par défaut : Quicksand
VITE_FONT_FAMILY=Quicksand
```

**Important :** Toutes les polices sont **auto-hébergées localement** au sein de l'application OmniCRM-UI. Cela signifie :

- **Pas de chargement de polices externes** - Les polices sont regroupées avec l'application
- **Compatible jardin clos** - Aucun accès Internet requis pour que les polices fonctionnent
- **Fonctionnement hors ligne** - Fonctionnalité complète dans des environnements réseau isolés ou restreints
- **Confidentialité** - Pas de requêtes externes vers Google Fonts, Adobe Fonts ou d'autres CDNs
- **Performance** - Chargement plus rapide sans dépendances externes
- **Sécurité** - Pas de suivi tiers ni de fuite de données via des requêtes de polices

**Options Disponibles :**

**Polices Sans-Serif :**

- Inter, Roboto, Open Sans, Lato, Quicksand (par défaut), Poppins, Nunito, Montserrat, Work Sans, Source Sans Pro, Raleway, Ubuntu, Josefin Sans, HKGrotesk

**Polices Serif :**

- Merriweather, Lora, Playfair Display

**Polices Système :**

- Système - Utilise les polices natives de l'appareil pour la meilleure performance et la plus petite taille de bundle

**Ajout de Polices Personnalisées :**

Oui, vous pouvez ajouter des polices supplémentaires ! Toutes les polices sont stockées localement dans l'application.

Pour ajouter une nouvelle police personnalisée :

1. **Ajoutez les fichiers de police** dans `OmniCRM-UI/src/assets/fonts/your-font-name/`
    - Utilisez le format WOFF2 pour la meilleure compression et le meilleur support des navigateurs
    - Incluez plusieurs poids (300, 400, 500, 600, 700) pour un rendu approprié
    - Nommez les fichiers : `your-font-name-300.woff2`, `your-font-name-400.woff2`, etc.
2. **Définissez les règles @font-face** dans `OmniCRM-UI/src/assets/scss/fonts/_google-fonts.scss`

```
//
// Votre Police Personnalisée - Description
//

@font-face {
  font-family: 'Your Font Name';
  font-style: normal;
  font-weight: 400;
  font-display: swap;
  src: url("../../fonts/your-font-name/your-font-name-
400.woff2") format('woff2');
}

@font-face {
  font-family: 'Your Font Name';
  font-style: normal;
  font-weight: 700;
  font-display: swap;
  src: url("../../fonts/your-font-name/your-font-name-
700.woff2") format('woff2');
}
```

3. **Définissez dans le fichier .env :**

```
VITE_FONT_FAMILY=Your Font Name
```

**Directives de Poids de Police :**

- 300 - Léger (optionnel, pour des titres subtils)
- 400 - Régulier (obligatoire, texte par défaut)
- 500 - Moyen (optionnel, accentuation)
- 600 - Semi-Gras (optionnel, sous-titres)
- 700 - Gras (obligatoire, titres et texte fort)

# Liens Rapides vers des Applications Web

Configurez jusqu'à 6 applications web d'accès rapide qui apparaissent dans le tableau de bord admin :

```
# Application Web 1 : GitHub
VITE_WEB_APP_1_NAME="GitHub"
VITE_WEB_APP_1_URL="https://github.com"
VITE_WEB_APP_1_ICON_PATH="resources/webapp_icons/github.png"

# Application Web 2 : Xero
VITE_WEB_APP_2_NAME="Xero"
VITE_WEB_APP_2_URL="https://go.xero.com/"
VITE_WEB_APP_2_ICON_PATH="resources/webapp_icons/xero.png"

# Applications Web 3-6 : Intégrations supplémentaires
# (Configurer de manière similaire avec NAME, URL, et ICON_PATH)
```

**Modèle :**

- `VITE_WEB_APP_N_NAME` - Nom affiché
- `VITE_WEB_APP_N_URL` - URL cible
- `VITE_WEB_APP_N_ICON_PATH` - Chemin du fichier icône (relatif à public/)

**Icônes Exemples :** GitHub, Xero, Monday.com, Gmail, MailJet, Slack

# Intégration du Tableau de Bord Grafana

OmniCRM s'intègre à Grafana pour fournir des tableaux de bord d'analytique. Les utilisateurs ayant la permission `grafana_access` peuvent accéder à Grafana à `/grafana` pour créer des tableaux de bord personnalisés, et ces tableaux de bord peuvent être intégrés dans l'interface OmniCRM.

```
# Intégration du Tableau de Bord Grafana
# IDs de Tableau de Bord (séparés par des virgules) - obtenez-les
à partir des URL /grafana/d/ID/name
VITE_GRAFANA_DASHBOARD_IDS=abc123,def456,ghi789

# Étiquettes de Tableau de Bord (séparées par des virgules,
doivent correspondre à l'ordre des IDs)
VITE_GRAFANA_DASHBOARD_LABELS=System Metrics,User
Analytics,Network Performance
```

**Champs :**

- `VITE_GRAFANA_DASHBOARD_IDS` - Liste séparée par des virgules des IDs de tableau de bord Grafana provenant du chemin URL
- `VITE_GRAFANA_DASHBOARD_LABELS` - Liste séparée par des virgules des noms d'affichage des tableaux de bord (doit correspondre à l'ordre des IDs)

**Comment Trouver les IDs de Tableau de Bord :**

Lorsque vous visualisez un tableau de bord dans Grafana, l'URL ressemblera à `/grafana/d/abc123/dashboard-name`. L'ID du tableau de bord est la partie après `/d/` et avant le nom (dans cet exemple, `abc123`).

**Utilisation :**

Une fois configurés, les tableaux de bord apparaissent automatiquement dans la barre latérale OmniCRM sous le menu "Tableaux de Bord". Les utilisateurs peuvent cliquer pour voir les tableaux de bord intégrés tout en gardant la navigation OmniCRM visible.

**Exemple :**

Si l'URL de votre tableau de bord est `http://your-server/grafana/d/system-overview-123/system-metrics`, alors :

- ID de Tableau de Bord : `system-overview-123`
- Étiquette de Tableau de Bord : Choisissez un nom convivial comme "Aperçu du Système"

**Important :**

- Le nombre d'IDs doit correspondre au nombre d'étiquettes
- Les tableaux de bord apparaissent dans la barre latérale dans l'ordre où ils sont définis
- Seuls les utilisateurs ayant la permission `grafana_access` peuvent voir les tableaux de bord
- Les modifications nécessitent de reconstruire l'UI frontend

# URLs de Support et de Documentation

```
# URLs FAQ et Support
VITE_FAQS_URL=https://support.yourcompany.com/faqs
VITE_SUPPORT_URL=https://support.yourcompany.com
```

**But :** Liens vers des ressources de support externes affichés dans l'UI.

# Connexions Sociales

```
# Autoriser les Connexions Sociales (oui|non)
VITE_ALLOW_SOCIAL_LOGINS=yes
```

**Options :**

- `yes` - Activer les boutons de connexion sociale (Google, Facebook, etc.)
- `no` - Désactiver les connexions sociales

**Remarque :** Les fournisseurs de connexion sociale doivent être configurés séparément.

# Analytique et Suivi

```
# Google Analytics 4 ou Google Tag Manager (optionnel)
# Pour GA4, utilisez le format : G-XXXXXXXXXX
# Pour GTM, utilisez le format : GTM-XXXXXXX
VITE_GOOGLE_TAG_ID=G-XXXXXXXXXX
```

**But :** Activer le suivi Google Analytics 4 (GA4) ou Google Tag Manager (GTM) pour surveiller le comportement des utilisateurs et l'utilisation de l'application.

**Formats Supportés :**

- **ID de Mesure GA4 :** `G-XXXXXXXXXX` - Pour le suivi Google Analytics 4
- **ID de Conteneur GTM :** `GTM-XXXXXXX` - Pour Google Tag Manager
- **Désactivé :** Laissez vide ou omettez pour désactiver le suivi

**Comment Cela Fonctionne :**

Le système détecte automatiquement le service de suivi en fonction du format de l'ID :

- Les IDs commençant par `G-` initialisent Google Analytics 4
- Les IDs commençant par `GTM-` initialisent Google Tag Manager
- Une valeur vide ou manquante désactive tout suivi

**Fonctionnalités :**

- **Suivi Automatique des Pages** - Chaque changement de route est suivi comme une vue de page
- **Suivi des Événements Personnalisés** - Suivez les interactions des utilisateurs, les clics sur les boutons, les soumissions de formulaires
- **Aucun Changement de Code Requis** - Il suffit de définir la variable d'environnement et de reconstruire l'UI

**Instructions de Configuration :**

1. **Obtenez Votre ID de Suivi :**

   *Pour Google Analytics 4 :*

   - Connectez-vous à Google Analytics
   - Créez ou sélectionnez une propriété GA4
   - Allez à **Admin** → **Flux de Données**
   - Sélectionnez votre flux web
   - Copiez l'**ID de Mesure** (format : `G-XXXXXXXXXX`)

   *Pour Google Tag Manager :*

   - Connectez-vous à Google Tag Manager
   - Créez ou sélectionnez un conteneur
   - Copiez l'**ID de Conteneur** (format : `GTM-XXXXXXX`)

2. **Ajoutez au Fichier .env :**

```
VITE_GOOGLE_TAG_ID=G-XXXXXXXXXX
```

3. **Reconstruisez l'UI :**

```
cd OmniCRM-UI
npm run build
```

4. **Vérifiez le Suivi :**

   - Ouvrez votre application dans un navigateur
   - Vérifiez la console du navigateur pour les messages d'initialisation
   - Utilisez les rapports en temps réel de Google Analytics pour voir les utilisateurs actifs

**Utilisation Avancée - Suivi des Événements Personnalisés :**

L'intégration fournit des fonctions utilitaires pour suivre des événements personnalisés tout au long de votre application.

**Exemple - Suivre les Clics sur les Boutons :**

```
import { trackEvent } from './utils/googleAnalytics';

// Dans votre composant
const handleCheckout = () => {
  trackEvent('checkout_initiated', {
    cart_value: 99.99,
    item_count: 3,
    page: '/cart'
  });
  // Continuez avec la logique de paiement
};
```

**Exemple - Suivre les Soumissions de Formulaires :**

```
import { trackEvent } from './utils/googleAnalytics';

const handleFormSubmit = (formType) => {
  trackEvent('form_submission', {
    form_type: formType,
    timestamp: new Date().toISOString()
  });
};
```

**Exemple - Suivre les Vues de Pages Personnalisées :**

```
import { trackPageView } from './utils/googleAnalytics';

// Suivre manuellement une vue de page (utile pour les dialogues modaux, les onglets, etc.)
trackPageView('/virtual/page-path', 'Titre de la Page');
```

**Fonctions de Suivi Disponibles :**

| Fonction | But | Paramètres |
|---|---|---|
| `initializeGoogleTracking()` | Initialiser le suivi au démarrage de l'application | Aucun (appelé automatiquement) |
| `trackEvent(eventName, eventParams)` | Suivre des événements personnalisés | Nom de l'événement (chaîne), paramètres (objet) |
| `trackPageView(path, title)` | Suivre manuellement les vues de pages | Chemin (chaîne), titre (chaîne) |

**Confidentialité et Conformité :**

Le suivi Google Analytics collecte des données sur le comportement des utilisateurs. Assurez-vous de respecter les réglementations sur la protection des données :

- **GDPR (Europe)** - Obtenez le consentement des utilisateurs avant le suivi, fournissez un mécanisme de désinscription
- **CCPA (Californie)** - Permettez aux utilisateurs de se désinscrire de la collecte de données
- **Politique de Confidentialité** - Divulguer l'utilisation de Google Analytics dans votre politique de confidentialité
- **Consentement aux Cookies** - Implémentez une bannière de consentement aux cookies si requis par les lois locales
- **Conservation des Données** - Configurez des périodes de conservation des données appropriées dans les paramètres de Google Analytics

**Dépannage :**

| Problème | Cause Possible | Solution |
|---|---|---|
| Pas de données dans le tableau de bord GA | ID de suivi incorrect | Vérifiez que le format de l'ID correspond à G-XXXXXXXXXX ou GTM-XXXXXXX |
| Erreurs de console au chargement de la page | ID de suivi invalide | Vérifiez les fautes de frappe dans le fichier .env |
| Vues de pages non suivies | UI non reconstruite après changement de .env | Exécutez `npm run build` et redémarrez l'UI |
| Événements non apparaissant | Bloqueur de publicités activé | Testez avec le bloqueur de publicités désactivé |
| Données retardées | Comportement normal de GA | Utilisez les rapports en temps réel pour une vérification immédiate |

**Considérations de Performance :**

- Les scripts de suivi se chargent de manière asynchrone et ne bloquent pas le rendu de la page
- Impact minimal sur la performance (< 50 Ko de charge utile supplémentaire)
- Les scripts sont mis en cache par les navigateurs après le premier chargement
- Aucun suivi n'a lieu si `VITE_GOOGLE_TAG_ID` est vide

**Détails d'Implémentation :**

- Code de suivi : `OmniCRM-UI/src/utils/googleAnalytics.js`
- Hook de suivi de page : `OmniCRM-UI/src/utils/usePageTracking.js`

- Initialisation : Automatique au démarrage de l'application dans `App.js` ou `index.js`

## Configuration de Recharge et de Top-Up

```
VITE_TOPUP_PRICE_PER_DAY=10
```

**But :** Définit le prix par jour pour les services de recharge/top-up dans le portail de service client.

**Champs :**

- `VITE_TOPUP_PRICE_PER_DAY` - Prix quotidien pour les services de recharge (valeur numérique)

**Exemple :** Si défini à `10` et que la devise est USD, les clients paient 10 $ par jour de service.

**Remarque :** Cette valeur doit correspondre à la configuration tarifaire du backend. Voir `features_topup_recharge` pour des détails complets de configuration.

---

# Branding et Personnalisation Visuelle

OmniCRM vous permet de remplacer les images de branding par défaut par le logo et les écrans de démarrage de votre entreprise sans modifier le code.

## Fichiers de Logo et Système de Fallback

Les logos sont stockés dans `/OmniCRM-UI/src/assets/images/omnitouch/` et utilisent un système de fallback :

**Logos par Défaut** (toujours présents) :

- `DefaultLogoDark.png` - Logo thème sombre (utilisé sur fonds clairs)
- `DefaultLogoLight.png` - Logo thème clair (utilisé sur fonds sombres)

**Logos Personnalisés** (optionnels, prennent la priorité lorsqu'ils sont présents) :

- `logoSm.png` - Petit logo pour la barre latérale réduite (recommandé : 100x100px)
- `logoDark.png` - Logo sombre de taille normale pour les en-têtes (recommandé : 200x50px)
- `logoLight.png` - Logo clair de taille normale pour les écrans d'authentification (recommandé : 200x50px)

**Comment Fonctionne le Fallback de Logo :**

Le système tente de charger d'abord les logos personnalisés. Si un fichier de logo personnalisé n'existe pas, il se rabat sur le logo par défaut :

```
// Depuis Header.js
const tryImport = (filename) => {
    try {
        return require(`../assets/images/omnitouch/${filename}`);
    } catch (err) {
        return null; // Se rabat sur le défaut
    }
};

const userLogoSm = tryImport("logoSm.png");
const userLogoDark = tryImport("logoDark.png");
const userLogoLight = tryImport("logoLight.png");
```

**Où les Logos Apparaissent :**

- **logoSm.png** - Barre latérale réduite, navigation mobile, affichages d'en-tête petits
- **logoDark.png** - Barre d'en-tête principale (mode clair), en-tête du tableau de bord admin
- **logoLight.png** - Écrans de connexion/enregistrement, fonds sombres, carrousel d'authentification

**Remplacer les Logos :**

1. **Créez Vos Fichiers de Logo :**
   - Utilisez le format PNG pour le support de la transparence
   - Correspondre aux dimensions recommandées ci-dessus
   - Assurez-vous que les logos sont clairs à la fois en résolutions régulières et retina
2. **Ajoutez à OmniCRM :**

```
# Copiez vos fichiers de logo dans le répertoire d'images
omnitouch
cp /path/to/your/logoSm.png OmniCRM-
UI/src/assets/images/omnitouch/
cp /path/to/your/logoDark.png OmniCRM-
UI/src/assets/images/omnitouch/
cp /path/to/your/logoLight.png OmniCRM-
UI/src/assets/images/omnitouch/
```

3. **Reconstruisez l'UI :**

```
cd OmniCRM-UI
npm run build
```

4. **Vérifiez les Changements :**
   - Vérifiez l'en-tête en mode clair (devrait afficher `logoDark.png`)
   - Vérifiez l'en-tête en mode sombre (devrait afficher `logoLight.png`)
   - Vérifiez la barre latérale réduite (devrait afficher `logoSm.png`)
   - Vérifiez l'écran de connexion (devrait afficher `logoLight.png`)

**Meilleures Pratiques de Conception de Logo :**

- **Contraste** - Assurez-vous que les logos sont visibles sur les fonds clairs et sombres
- **Simplicité** - Les logos doivent être reconnaissables à des tailles réduites
- **Format** - Utilisez PNG avec des arrière-plans transparents

- **Retina** - Fournissez une résolution 2x pour les affichages haute définition
- **Cohérence** - Utilisez les mêmes couleurs de marque sur toutes les variantes de logo

## Écrans de Démarrage et Arrière-plans d'Authentification

Les écrans d'authentification (connexion, enregistrement, réinitialisation de mot de passe) utilisent un arrière-plan de carrousel avec des images personnalisables.

**Emplacement :** `/OmniCRM-UI/src/pages/AuthenticationInner/authCarousel.js`

**Configuration par Défaut :**

```
import logoLight from "../../assets/images/logo-light.png";

// Logo affiché sur les écrans d'authentification
<img src={logoLight} alt="" height="18" />
```

**Personnaliser les Écrans d'Authentification :**

1. Remplacez `logo-light.png` dans `/OmniCRM-UI/src/assets/images/`
2. Ajoutez un CSS d'arrière-plan personnalisé à la classe `.auth-one-bg`
3. Modifiez les citations du carrousel dans `authCarousel.js` pour correspondre à votre voix de marque

**Exemple de Personnalisation :**

```
/* Ajoutez à votre CSS personnalisé */
.auth-one-bg {
    background-image: url('/assets/images/custom-auth-bg.jpg');
    background-size: cover;
    background-position: center;
}
```

---

# Génération de PDF de Facture

OmniCRM génère des factures PDF professionnelles en utilisant des modèles HTML Jinja2 et le rendu PDF WeasyPrint.

## Comment Fonctionne la Génération de PDF

1. **Sélection de Modèle :**
    - Le nom de fichier du modèle est spécifié dans `crm_config.yaml` sous `invoice.template_filename`
    - Le modèle est chargé depuis le répertoire `/OmniCRM-API/invoice_templates/`
2. **Préparation des Données :**
    - Les données de la facture (ID, dates, montants, statut) sont récupérées de la base de données
    - Les informations client (nom, adresse) sont récupérées
    - La liste des transactions est compilée (tous les frais/crédits sur la facture)
3. **Rendu du Modèle :**
    - Jinja2 rend le modèle HTML avec les données de la facture
    - Les variables comme `{{ invoice_number }}`, `{{ total_amount }}`, etc. sont remplacées
    - Le HTML rendu est enregistré dans `invoice_templates/rendered/` pour le débogage
4. **Génération de PDF :**
    - WeasyPrint convertit le HTML rendu en PDF
    - Le PDF supporte le style CSS, les images, les sauts de page, les en-têtes/pieds de page
    - Les données binaires PDF sont générées en mémoire
5. **Mise en Cache :**

- Le PDF est encodé en Base64 et stocké dans la table `Invoice_PDF_Cache`
- Un hachage SHA256 est calculé pour la vérification d'intégrité
- Les demandes suivantes retournent le PDF mis en cache (livraison instantanée)

6. **Invalidation du Cache :**

   - Le cache est invalidé lorsque la facture est modifiée, annulée ou remboursée
   - Les modifications de modèle n'invalident pas automatiquement les caches existants

## Structure du Modèle de Facture

Les modèles de facture sont des fichiers HTML Jinja2 avec des variables et de la logique intégrées.

**Emplacement du Modèle :** `/OmniCRM-API/invoice_templates/yourcompany_invoice_template.html`

**Variables Disponibles :**

```
{
    'invoice_number': 12345,
    'date': '2025-01-04',
    'client': {
        'name': 'John Smith',
        'address': {
            'address_line_1': '123 Main St',
            'city': 'Sydney',
            'state': 'NSW',
            'zip_code': '2000',
            'country': 'Australia'
        }
    },
    'transaction_list': [
        [
            {
                'transaction_id': 1,
                'title': 'Mobile Service - Monthly Fee',
                'retail_cost': 30.00,
                'wholesale_cost': 10.00,
                'created': '2025-01-01'
            },
            {
                'transaction_id': 2,
                'title': 'Data Addon - 5GB',
                'retail_cost': 15.00,
                'wholesale_cost': 5.00,
                'created': '2025-01-15'
            }
        ]
    ],
    'total_amount': 45.00,
    'due_date': '2025-01-31',
    'start_date': '2025-01-01',
    'end_date': '2025-01-31',
    'paid': False,
    'void': False
}
```

**Exemple de Fragment de Modèle :**

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Facture {{ invoice_number }}</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 40px;
        }
        .header {
            text-align: center;
            margin-bottom: 30px;
        }
        .invoice-details {
            margin-bottom: 20px;
        }
        table {
            width: 100%;
            border-collapse: collapse;
        }
        th, td {
            border: 1px solid #ddd;
            padding: 12px;
            text-align: left;
        }
        th {
            background-color: #405189;
            color: white;
        }
        .total {
            text-align: right;
            font-size: 18px;
            font-weight: bold;
            margin-top: 20px;
        }
    </style>
</head>
<body>
    <div class="header">
        <h1>FACTURE</h1>
    </div>
```

```
    <div class="invoice-details">
        <p><strong>Numéro de Facture :</strong> {{ invoice_number
}}</p>
        <p><strong>Date :</strong> {{ date }}</p>
        <p><strong>Date d'Échéance :</strong> {{ due_date }}</p>
        <p><strong>Période de Facturation :</strong> {{ start_date
}} à {{ end_date }}</p>
    </div>

    <div class="customer-details">
        <h3>Facturer À :</h3>
        <p>{{ client.name }}</p>
        <p>{{ client.address.address_line_1 }}</p>
        <p>{{ client.address.city }}, {{ client.address.state }}
{{ client.address.zip_code }}</p>
        <p>{{ client.address.country }}</p>
    </div>

    <table>
        <thead>
            <tr>
                <th>Description</th>
                <th>Date</th>
                <th>Montant</th>
            </tr>
        </thead>
        <tbody>
            {% for transaction in transaction_list[0] %}
            <tr>
                <td>{{ transaction.title }}</td>
                <td>{{ transaction.created }}</td>
                <td>${{ "%.2f"|format(transaction.retail_cost) }}
</td>
            </tr>
            {% endfor %}
        </tbody>
    </table>

    <div class="total">
        <p>Montant Total Dû : ${{ "%.2f"|format(total_amount) }}
</p>
    </div>

    {% if paid %}
```

```
    <div style="text-align: center; color: green; font-weight:
bold;">
        PAYÉ
    </div>
    {% endif %}

    {% if void %}
    <div style="text-align: center; color: red; font-weight:
bold;">
        ANNULÉ
    </div>
    {% endif %}
</body>
</html>
```

**Meilleures Pratiques pour le Modèle :**

- **Utilisez des chemins absolus pour les images** - `file:///absolute/path/to/image.png`
- **CSS en ligne** - WeasyPrint ne charge pas de manière fiable les feuilles de style externes
- **Testez avec des données d'exemple** - Utilisez `invoice_templates/rendered/` pour inspecter le HTML
- **Sauts de page** - Utilisez `<div style="page-break-after: always;"></div>` pour les factures multi-pages
- **En-têtes et pieds de page** - Utilisez des règles CSS `@page` pour les éléments répétitifs
- **Formatage de devise** - Utilisez des filtres Jinja2 : `{{ "%.2f"|format(amount) }}`

# Création d'un Modèle de Facture Personnalisé

1. **Copiez le Modèle Exemple :**

```
cd /OmniCRM/OmniCRM-API/invoice_templates
cp sample_invoice_template.html
yourcompany_invoice_template.html
```

2. **Modifiez le Modèle :**
    - Remplacez le nom de l'entreprise, le logo, les informations de contact
    - Ajustez le style (couleurs, polices, mise en page) pour correspondre à la marque
    - Ajoutez ou supprimez des sections si nécessaire (détails de taxe, instructions de paiement, etc.)
3. **Mettez à Jour la Configuration :**

    Modifiez `crm_config.yaml` :

    ```
    invoice:
      template_filename: 'yourcompany_invoice_template.html'
    ```

4. **Testez la Génération de Facture :**
    - Créez une facture test dans le CRM
    - Téléchargez le PDF et vérifiez le formatage
    - Vérifiez `invoice_templates/rendered/{invoice_id}.html` pour les erreurs de débogage
5. **Invalidation des Anciens Caches (si nécessaire) :**

    Si vous avez changé le modèle et souhaitez régénérer les factures existantes :

    ```
    -- Effacer tous les PDFs mis en cache (force la régénération)
    DELETE FROM Invoice_PDF_Cache;
    ```

# Système de Mise en Cache des PDF

Pour améliorer les performances, OmniCRM met en cache les PDFs générés :

**Comportement du Cache :**

- **Première Demande** - Le PDF est généré, mis en cache et retourné

- **Demandes Suivantes** - Le PDF mis en cache est retourné immédiatement (pas de régénération)
- **Invalidation du Cache** - Se produit lorsque la facture est modifiée, annulée ou remboursée
- **Nettoyage du Cache** - Les anciens caches sont automatiquement purgés après 30 jours d'inactivité

**Stockage du Cache :**

- PDF encodé en Base64 stocké dans la table `Invoice_PDF_Cache`
- Hachage de contenu SHA256 pour vérification d'intégrité
- Inclut le nom de fichier, l'horodatage de création, l'horodatage du dernier accès

**Gestion Manuelle du Cache :**

```
# Dans l'API OmniCRM ou shell Python
from services.invoice_service import cleanup_old_pdf_cache,
invalidate_invoice_cache
from utils.db_helpers import get_db_session

session = get_db_session()

# Nettoyer les caches plus anciens que 30 jours
result = cleanup_old_pdf_cache(session, days_old=30)
print(result)  # {'status': 'success', 'deleted_count': 15}

# Invalider le cache d'une facture spécifique
invalidate_invoice_cache(session, invoice_id='12345')
```

**Points de Terminaison API :**

Générer/télécharger le PDF de facture :

GET /invoice/pdf/{invoice_id}

**Réponse :** Téléchargement du fichier PDF avec nom de fichier provenant du descripteur de relevé Stripe

**En-têtes de Cache :**

- Première demande : Réponse plus lente (temps de génération)
- Demandes mises en cache : Réponse instantanée
- Le cache hit/miss est transparent pour l'utilisateur

---

# Application des Changements de Configuration

## Backend (crm_config.yaml)

1. Modifiez `OmniCRM-API/crm_config.yaml`
2. Enregistrez les modifications
3. Redémarrez le service API :

```
cd OmniCRM-API
sudo systemctl restart omnicrm-api
# ou
./restart_api.sh
```

**Les changements prennent effet immédiatement** après le redémarrage.

## Frontend (.env)

1. Modifiez `OmniCRM-UI/.env`
2. Enregistrez les modifications
3. Reconstruisez l'UI :

```
cd OmniCRM-UI
npm run build
```

4. Redémarrez le service UI ou le serveur web

**Mode Développement :**

Lors du développement avec `npm start`, redémarrez le serveur de développement pour appliquer les changements.

---

# Meilleures Pratiques de Configuration

## Sécurité

- **Ne jamais commettre de secrets** - Utilisez `.gitignore` pour les fichiers de configuration avec des informations d'identification
- **Utilisez des mots de passe forts** - Minimum 16 caractères avec des majuscules, des chiffres et des symboles
- **Faites tourner les informations d'identification régulièrement** - Surtout pour les déploiements en production
- **Restreignez l'accès à la base de données** - Utilisez des listes blanches IP et des règles de pare-feu
- **Utilisez des configurations spécifiques à l'environnement** - Configurations séparées pour dev/staging/production
- **Limitez les permissions des clés API** - Assignez les rôles minimaux nécessaires
- **Utilisez la liste blanche IP avec parcimonie** - Préférez les clés API pour une meilleure sécurité

## Maintenance

- **Documentez les changements** - Tenez un changelog des modifications de configuration
- **Sauvegardez les configurations** - Conservez des copies avant les changements majeurs
- **Testez en staging** - Vérifiez les changements de configuration avant le déploiement en production
- **Contrôle de version** - Suivez les modèles de configuration (sans secrets) dans git

## Performance

- **Utilisez une base de données locale** - Évitez la base de données distante pour de meilleures performances
- **Configurez la mise en cache** - Activez la mise en cache OCS si disponible
- **Optimisez Grafana** - Limitez le nombre de tableaux de bord intégrés

## Branding

- **Correspondre aux couleurs** - Assurez-vous que les couleurs de l'UI complètent votre logo
- **Testez le contraste** - Vérifiez la lisibilité du texte sur les fonds colorés
- **Testez sur mobile** - Vérifiez le branding sur les appareils mobiles
- **Placement du logo** - Utilisez des tailles de logo appropriées pour différents contextes

---

# Dépannage

## Problèmes Courants

### Changements non appliqués

- **Cause :** Service non redémarré ou UI non reconstruite
- **Solution :** Redémarrez les services API/UI après les changements de configuration

### Erreurs de syntaxe YAML

- **Cause :** Format YAML invalide (indentation, guillemets, etc.)
- **Solution :** Validez le YAML en ligne ou utilisez `yamllint crm_config.yaml`

### Échec de la connexion à la base de données

- **Cause :** Mauvaises informations d'identification ou serveur inaccessible

- **Solution :** Vérifiez que la base de données fonctionne, que les informations d'identification sont correctes

**Paiements Stripe ne fonctionnent pas**

- **Cause :** Clés non correspondantes entre le backend et le frontend
- **Solution :** Assurez-vous que `publishable_key` correspond dans les deux fichiers

**Emails non envoyés**

- **Cause :** Informations d'identification Mailjet invalides ou IDs de modèle
- **Solution :** Vérifiez la clé/secrète API Mailjet, vérifiez que les IDs de modèle existent

**Échec de Génération de PDF :**

- Vérifiez que WeasyPrint est installé : `pip install weasyprint`
- Vérifiez que le nom du modèle correspond à `crm_config.yaml`
- Vérifiez `invoice_templates/rendered/` pour les erreurs de rendu HTML
- Consultez les journaux API pour les erreurs de modèle Jinja2

**Images Non Apparaissant dans le PDF :**

- Utilisez des chemins de fichiers absolus : `file:///full/path/to/image.png`
- Assurez-vous que les fichiers image existent et sont lisibles
- Vérifiez le format d'image (PNG et JPEG fonctionnent le mieux)
- Vérifiez que les chemins d'image ne contiennent pas de caractères spéciaux

**Problèmes de Style :**

- Inlinez tout le CSS (les feuilles de style externes ne sont pas supportées)
- Évitez les fonctionnalités CSS complexes (flexbox, grid peuvent ne pas se rendre correctement)
- Testez d'abord avec des mises en page simples, ajoutez de la complexité progressivement

- Utilisez des tables pour la mise en page au lieu de divs lorsque cela est possible

**Cache Ne S'invalide Pas :**

- Vérifiez que `invalidate_invoice_cache()` est appelé lorsque la facture est modifiée
- Vérifiez que les mises à jour de transaction déclenchent l'invalidation du cache
- Supprimez manuellement de la table `Invoice_PDF_Cache` si nécessaire

---

# Liste de Vérification de Configuration

Utilisez cette liste de vérification lors du déploiement d'OmniCRM :

## Configuration Backend

- [ ] Copiez `.env.example` à `.env`
- [ ] Définissez des mots de passe de base de données forts
- [ ] Configurez les informations d'identification CGRates
- [ ] Mettez à jour `crm_config.yaml` avec vos paramètres :
  - [ ] Connexion à la base de données
  - [ ] Types de services
  - [ ] Clés API Mailjet et IDs de modèle
  - [ ] Emails de notification d'échec d'approvisionnement
  - [ ] Nom de fichier du modèle de facture
  - [ ] URL de base CRM (accessible publiquement)
  - [ ] Points de terminaison OCS/CGRates
  - [ ] Configuration SMSC
  - [ ] Générez une nouvelle clé secrète JWT
  - [ ] Clés Stripe (live, pas test)
  - [ ] Clés API et liste blanche IP

## Configuration Frontend

- [ ] Copiez `OmniCRM-UI/.env.example` à `OmniCRM-UI/.env`
- [ ] Définissez la clé API Google Maps
- [ ] Définissez la clé publiable Stripe
- [ ] Mettez à jour le branding de l'entreprise :
  - [ ] Nom de l'entreprise
  - [ ] Nom du portail
  - [ ] Nom du service client
  - [ ] Slogan de l'entreprise
- [ ] Configurez la localisation :
  - [ ] Langue par défaut
  - [ ] Locale
  - [ ] Emplacement et pays par défaut
  - [ ] Code de devise et symbole
- [ ] Définissez la couleur de marque principale
- [ ] Configurez les intégrations d'applications web (optionnel)
- [ ] Ajoutez des URLs de support et FAQ (optionnel)
- [ ] Définissez l'ID de suivi Google Analytics/Tag Manager (optionnel)

## Actifs de Branding

- [ ] Créez des fichiers de logo (logoSm.png, logoDark.png, logoLight.png)
- [ ] Téléchargez les logos dans `OmniCRM-UI/src/assets/images/omnitouch/`
- [ ] Créez le modèle HTML de facture personnalisé
- [ ] Téléchargez le modèle de facture dans `OmniCRM-API/invoice_templates/`
- [ ] Mettez à jour `crm_config.yaml` avec le nom de fichier du modèle de facture
- [ ] Testez la génération de PDF de facture
- [ ] Reconstruisez l'UI : `npm run build`

## Sécurité

- [ ] Changez tous les mots de passe par défaut
- [ ] Générez un secret JWT unique

- [ ] Utilisez des clés Stripe en production (pas de clés de test)
- [ ] Faites tourner les clés API Mailjet
- [ ] Activez les règles de pare-feu
- [ ] Configurez la liste blanche IP pour l'accès API
- [ ] Configurez des certificats SSL/TLS
- [ ] Activez HTTPS pour tous les points de terminaison
- [ ] Passez en revue les paramètres CORS
- [ ] Implémentez une limitation de taux
- [ ] Configurez des procédures de sauvegarde et de récupération

## Tests

- [ ] Testez le flux d'enregistrement client
- [ ] Testez l'approvisionnement du service de bout en bout
- [ ] Vérifiez que les notifications par email sont envoyées correctement
- [ ] Testez la génération de factures et le téléchargement de PDF
- [ ] Vérifiez le traitement des paiements (Stripe)
- [ ] Vérifiez l'authentification des utilisateurs et 2FA
- [ ] Testez l'imitation et la journalisation des audits
- [ ] Vérifiez que les données d'utilisation se synchronisent depuis OCS
- [ ] Testez la création et le renouvellement de l'ActionPlan
- [ ] Confirmez que l'allocation d'inventaire fonctionne correctement

## Déploiement

- [ ] Démarrez la base de données (MySQL)
- [ ] Démarrez CGRates
- [ ] Démarrez l'API OmniCRM
- [ ] Démarrez l'UI OmniCRM
- [ ] Configurez le proxy inverse (nginx, traefik)
- [ ] Configurez la surveillance (Grafana, Prometheus)
- [ ] Configurez l'agrégation des journaux
- [ ] Configurez des sauvegardes automatisées

- [ ] Formez le personnel à l'utilisation du système

---

# Documentation Connexe

- RBAC et Gestion des Utilisateurs
- Produits et Services
- Approvisionnement Ansible
- Gestion des Inventaires
- Factures Clients
- Authentification à Deux Facteurs
- Service Client et Imitation
- *Intégrations de Fournisseurs de Paiement*
- `administration_api_keys` - Gestion des clés API
- `integrations_mailjet` - Intégration d'email
- `concepts_api` - Authentification API

# Vue d'ensemble de l'inventaire dans OmniCRM

Le système **Inventaire** dans OmniCRM est conçu pour gérer et suivre à la fois les articles physiques et virtuels utilisés par les opérateurs de réseau et les clients.

Cela signifie que nous pouvons suivre toutes sortes d'articles, tels que des modems, des numéros de téléphone, des blocs d'adresses IP, ou même du matériel physique comme des ONT GPON ou des CPE sans fil fixes.

Voir aussi : Attributs des clients pour stocker des métadonnées personnalisées, et Étiquettes des clients pour une catégorisation visuelle.

Pour soutenir les clients avec une empreinte réseau fixe, le système d'inventaire peut également suivre les maisons desservies pour un service donné, permettant aux opérateurs de faire une qualification de service à distance, et pour ceux qui exploitent un réseau sans fil fixe, nous pouvons suivre les CPE déployés sur le terrain avec leurs emplacements.

> **REMARQUE**
>
> Les articles d'inventaire sont liés aux produits lors de la provisionnement via le champ `inventory_items_list`. Pour un guide complet sur la façon dont l'inventaire s'intègre au provisionnement des produits, y compris l'interface utilisateur du sélecteur d'inventaire et l'intégration du playbook Ansible, voir Guide complet du cycle de vie des produits - Exigences en matière d'inventaire.

# Objectif

L'inventaire OmniCRM sert plusieurs objectifs clés :

1. **Provisionnement des services** : Lorsqu'un client s'inscrit à un service, des articles comme des modems, des cartes SIM ou des numéros de téléphone peuvent devoir être alloués. Le système d'inventaire suit ces articles et les associe aux clients.
2. **Gestion des stocks** : Pour les stocks physiques, tels que le matériel ou d'autres équipements, l'inventaire aide les opérateurs à maintenir une visibilité sur ce qui est disponible, où il est stocké, et ce qui a été alloué ou vendu aux clients.
3. **Allocation aux clients** : Le système permet d'allouer des articles aux clients, que ce soit pour une utilisation dans un service (par exemple, attribuer un modem au compte Internet d'un client) ou pour une vente directe.
4. **Qualification de service / Empreinte réseau** : En stockant des informations sur l'empreinte réseau, telles que chaque maison desservie pour un service GPON, ou chaque CPE sans fil fixe déployé, cela permet au personnel de faire une qualification de service à distance et de voir s'il y a une empreinte réseau dans une zone spécifique.

# Exemple de cycle de vie de l'inventaire

Pour illustrer comment fonctionne le système d'inventaire, considérons les exemples suivants :

## Exemple de carte SIM

Un lot de 1000 cartes SIM est commandé auprès d'Omnitouch. Tout d'abord, un modèle d'inventaire est créé pour les cartes SIM (s'il n'existe pas déjà) et les SIM commandées sont chargées dans l'inventaire dans l'état *En transit*.

Une fois les SIM reçues, elles sont marquées comme *En stock*, et peuvent être allouées à différents magasins de détail, avec l'emplacement de l'article d'inventaire mis à jour pour refléter dans quel magasin de détail chaque carte SIM se trouve - Cela est utile pour suivre les niveaux de stock dans chaque magasin et s'assurer que chaque magasin a suffisamment de stock pour répondre à la demande des clients.

Lorsqu'un client s'inscrit pour un service mobile en magasin, une carte SIM est allouée au client et le statut est changé en *Alloué*. L'article d'inventaire de la carte SIM est alors attribué au client, et le statut est mis à jour en *En utilisation*.

Si le client annule le service ou si le service devient inactif, la carte SIM est marquée comme *Décommissionnée*.

## Exemple de maisons desservies par GPON

Pour un réseau GPON, le système d'inventaire peut suivre chaque maison desservie pour un service donné.

Lorsqu'une nouvelle zone est construite, chaque adresse desservie peut être ajoutée à l'inventaire.

Cela permet aux opérateurs de voir quelles maisons sont desservies pour un service donné, et quelles maisons ne le sont pas encore.

Lorsqu'un client s'inscrit pour un service, OmniCRM peut automatiquement exécuter une qualification de service contre l'adresse du client, pour voir si l'adresse se trouve dans l'inventaire des maisons desservies et quels services peuvent être offerts.

# Modèles d'inventaire

La fonctionnalité **InventoryTemplate** permet la création de n'importe quel nombre de types d'articles avec des champs prédéfinis. Ces modèles agissent comme des plans qui définissent les caractéristiques essentielles de différents articles, tels que :

- **Modems** avec une adresse MAC (`itemtext1`) et un numéro de série (`itemtext2`).
- **Maisons desservies** pour un service donné, avec un emplacement et un statut (par exemple, desservi ou non desservi).
- **Numéros de téléphone** avec un numéro principal (`itemtext1`) et un emplacement géographique (`itemtext2`).

- **Ressources virtuelles** comme des blocs d'adresses IP, avec des identifiants mappés via le modèle.

Chaque modèle d'inventaire définit jusqu'à 20 champs de texte personnalisables (`itemtext1` à `itemtext20`) avec des étiquettes correspondantes (`itemtext1_label` à `itemtext20_label`) qui décrivent ce que chaque champ représente. Par exemple, un modèle de modem pourrait définir `itemtext1_label` sur "Adresse MAC" et `itemtext2_label` sur "Numéro de série".

Les opérateurs peuvent personnaliser les champs pour chaque type d'article à l'aide des **InventoryTemplates**. Ces modèles permettent de catégoriser et de gérer les articles de manière structurée, garantissant la cohérence dans la manière dont les articles sont suivis.

**Lien vers les produits :**

Les noms des modèles d'inventaire sont référencés dans les définitions de produits via le champ `inventory_items_list`. Lors du provisionnement d'un produit, le système affiche un sélecteur d'inventaire montrant uniquement les articles correspondant aux types de modèles requis.

**Exemple :** Un produit avec `inventory_items_list: "['SIM Card', 'Mobile Number']"` nécessite que deux modèles d'inventaire nommés exactement "SIM Card" et "Mobile Number" existent. Les noms des modèles sont sensibles à la casse.

Pour des détails complets sur la façon dont les modèles d'inventaire se connectent au provisionnement des produits, voir Cycle de vie des produits - Exigences en matière d'inventaire.

**Création de modèles d'inventaire via l'interface utilisateur**

Pour créer un nouveau modèle d'inventaire :

1. Accédez à **Inventaire** → **Modèles** depuis le menu principal
2. Cliquez sur le bouton **Ajouter un modèle**
3. Remplissez les champs requis :

   **Informations de base :**

   - **Icône** (facultatif) : Nom de classe d'icône pour identification visuelle (par exemple, `fa-solid fa-sim-card`)
   - **Article** (requis) : Le nom du modèle (doit correspondre exactement à ce qui est utilisé dans `inventory_items_list` pour les produits)

   **Informations sur le coût (Requis) :**

   - **Coût de gros** (requis) : Votre coût pour acheter ou provisionner ce type d'article
   - **Coût de détail** (requis) : Prix de détail standard s'il est vendu séparément aux clients

   > ⓘ **REMARQUE**
   >
   > Les coûts de gros et de détail définis ici servent de valeurs par défaut lors de la création de nouveaux articles d'inventaire à partir de ce

> modèle. Les articles d'inventaire individuels peuvent avoir des coûts différents si nécessaire.

**Étiquettes de champ :**

- **Étiquette du champ de texte 1** (requis) : Étiquette pour le premier champ personnalisable (par défaut "Numéro de modèle")
  - Exemples courants : "ICCID" pour les cartes SIM, "Adresse MAC" pour les modems, "Numéro de téléphone" pour les numéros
- **Étiquette du champ de texte 2** (requis) : Étiquette pour le deuxième champ personnalisable (par défaut "Numéro de série")
  - Exemples courants : "IMSI" pour les cartes SIM, "Numéro de série" pour le matériel, "Région géographique" pour les numéros
- **Étiquettes des champs de texte 3-20** (facultatif) : Étiquettes de champ supplémentaires selon les besoins
  - Cliquez sur **Ajouter un champ** pour ajouter d'autres champs personnalisés
  - N'ajoutez que les champs que vous utiliserez réellement pour ce type d'article

**Paramètres de visibilité :**

- **Autoriser le personnel à sélectionner dans le menu déroulant** : Autoriser le personnel à sélectionner ce type d'inventaire dans les menus déroulants
- **Autoriser le client à sélectionner dans le menu déroulant** : Autoriser les clients à voir/sélectionner ce type d'inventaire (portail client)

4. Cliquez sur **Enregistrer** pour créer le modèle

### Modification des modèles d'inventaire

Pour modifier un modèle existant :

1. Accédez à **Inventaire → Modèles**
2. Trouvez le modèle dans la liste
3. Cliquez sur le bouton **Modifier**

4. Modifiez les champs selon les besoins
5. Cliquez sur **Enregistrer**

> **⚠ ATTENTION**
>
> Changer les étiquettes de champ (par exemple, `itemtext1_label`) n'affecte que les nouveaux articles créés après le changement. Les articles d'inventaire existants conservent leurs données mais s'afficheront avec les nouveaux noms d'étiquettes.

> **⚠ ATTENTION**
>
> Les noms des modèles référencés dans les champs `inventory_items_list` des produits sont sensibles à la casse. Renommer un modèle rompra le lien avec les produits utilisant l'ancien nom.

**Exemples de modèles courants**

**Modèle de carte SIM :**

- Article : "Carte SIM"
- Coût de gros : 2.50
- Coût de détail : 10.00
- Étiquette du champ de texte 1 : "ICCID"
- Étiquette du champ de texte 2 : "IMSI"
- Étiquette du champ de texte 3 : "Type de SIM" (Physique/eSIM)

**Modèle de numéro de mobile :**

- Article : "Numéro de mobile"
- Coût de gros : 1.00
- Coût de détail : 0.00
- Étiquette du champ de texte 1 : "Numéro de téléphone"
- Étiquette du champ de texte 2 : "Région géographique"
- Étiquette du champ de texte 3 : "Type de numéro" (Mobile/Fixe)

**Modèle de CPE sans fil fixe :**

- Article : "CPE sans fil fixe"
- Coût de gros : 250.00
- Coût de détail : 450.00
- Étiquette du champ de texte 1 : "Adresse MAC"
- Étiquette du champ de texte 2 : "Numéro de série"
- Étiquette du champ de texte 3 : "Version du firmware"
- Étiquette du champ de texte 4 : "Fabricant"
- Étiquette du champ de texte 5 : "Modèle"

**Modèle d'ONT GPON :**

- Article : "ONT GPON"
- Coût de gros : 45.00
- Coût de détail : 0.00 (inclus avec le service)
- Étiquette du champ de texte 1 : "Numéro de série"
- Étiquette du champ de texte 2 : "Adresse MAC"
- Étiquette du champ de texte 3 : "Emplacement PON"
- Étiquette du champ de texte 4 : "Modèle"

# Création et gestion des articles d'inventaire

Une fois qu'un **InventoryTemplate** est défini, des articles d'**Inventaire** individuels peuvent être créés. Chaque article d'inventaire représente une instance spécifique d'un type d'article (par exemple, un modem ou un numéro de téléphone spécifique) qui peut être :

- **Alloué aux clients** : Les articles sont liés aux clients pour le provisionnement de services (par exemple, attribuer du matériel pour une connexion Internet).
- **Suivi pour le stock** : Les opérateurs peuvent surveiller l'inventaire disponible, tel que les articles non vendus ou non attribués.
- **Vendus ou décommissionnés** : Une fois vendus, les articles sont marqués avec des horodatages pertinents (par exemple, `sold_date`) et ne peuvent plus être considérés comme du stock disponible.

Grâce à ce système, OmniCRM facilite la gestion efficace des stocks, aide à allouer des ressources aux clients et fournit une visibilité détaillée sur l'état et l'historique de chaque article.

**Les services** peuvent être liés à un article d'**Inventaire** donné, permettant un suivi facile des articles associés à quels clients ou services.

Une fois qu'un article d'inventaire a été attribué à un client, les jeux Ansible peuvent mettre à jour le statut et l'historique de l'article pour refléter l'allocation. Cela garantit que les opérateurs ont un enregistrement précis de quels articles sont en cours d'utilisation et lesquels sont disponibles pour allocation, ainsi que de savoir quel client utilise quel article.

Nous pouvons voir les articles alloués à un client depuis la page de profil du client dans l'onglet **Inventaire**.

Pour les articles d'inventaire liés à un **Service**, nous pouvons le voir en modifiant le service, pour voir les articles d'inventaire liés.

# Champs des articles d'inventaire

Chaque article d'inventaire contient des informations complètes organisées en plusieurs catégories :

**Informations de base sur l'article**

- **inventory_id** - Identifiant unique pour l'article d'inventaire (généré automatiquement)
- **item** - Type d'article (correspond au nom du modèle d'inventaire, par exemple, "Carte SIM", "Modem", "Numéro de téléphone")
- **inventory_template_id** - Lien vers le modèle d'inventaire qui définit ce type d'article
- **customer_id** - Si attribué à un client, l'ID du client (nullable)
- **service_id** - Si lié à un service spécifique, l'ID du service (nullable)
- **item_location** - Emplacement physique ou logique de l'article :
  - Pour les articles physiques : bâtiment, entrepôt, emplacement d'étagère, nom du magasin, etc.
  - Pour les articles virtuels : emplacement géographique, emplacement du bloc IP, région de plage de numéros, etc.
- **item_state** - État actuel de l'article d'inventaire (valeurs énumérées) :
  - **Nouveau** - Article tout neuf, inutilisé
  - **Utilisé** - Précédemment utilisé mais fonctionnel
  - **Utilisation interne** - Alloué pour des tests internes ou un usage du personnel
  - **Attribué** - Actuellement attribué à un client ou à un service
  - **Endommagé** - Non fonctionnel, nécessite une réparation ou une élimination
  - **Hors service** - Temporairement indisponible
  - **Perdu** - L'article ne peut pas être localisé
  - **Volé** - L'article a été volé

**Champs personnalisables (du modèle)**

Le système d'inventaire prend en charge jusqu'à 20 champs de texte personnalisables dont la signification est définie par le modèle d'inventaire :

- **itemtext1** - Premier champ personnalisable (requis, étiquette définie par `itemtext1_label` du modèle)
  - Exemple : Pour les modems, cela pourrait être "Adresse MAC"
  - Exemple : Pour les cartes SIM, cela pourrait être "ICCID"
  - Exemple : Pour les numéros de téléphone, cela pourrait être "Numéro de téléphone"
- **itemtext2** à **itemtext20** - Champs personnalisables supplémentaires (facultatif, étiquettes définies par le modèle)
  - Exemple : itemtext2 pour les modems pourrait être "Numéro de série"
  - Exemple : itemtext2 pour les cartes SIM pourrait être "IMSI"
  - Exemple : itemtext3 pour les modems pourrait être "Version du firmware"

Chaque modèle d'inventaire définit quels champs sont utilisés et ce qu'ils représentent via les champs d'étiquette correspondants (`itemtext1_label`, `itemtext2_label`, etc.).

**Informations sur le coût**

- **wholesale_cost** - Votre coût pour acheter/provisionner cet article (float)
- **retail_cost** - Prix facturé au client s'il est vendu séparément (float)
- **sold_date** - Horodatage lorsque l'article a été vendu ou attribué au client

**Adresse physique (pour l'équipement et les sites réseau)**

Utilisé pour suivre les emplacements de déploiement physique, en particulier pour l'équipement réseau fixe (CPE, ONT, modems) ou les maisons desservies :

- **address_line_1** - Adresse de rue, numéro de bâtiment, numéro d'unité
- **address_line_2** - Informations d'adresse supplémentaires (suite, appartement, étage)
- **city** - Ville ou commune
- **state** - État, province ou région
- **zip_code** - Code postal
- **country** - Nom du pays

**Emplacement géographique (rempli automatiquement depuis l'interface Web)**

Lors de la création d'articles d'inventaire via l'interface Web avec l'auto-complétion d'adresse, ces champs sont automatiquement remplis :

- **google_maps_place_id** - ID de lieu Google Maps pour l'adresse
- **plus_code** - Code Plus Google Maps (Open Location Code) pour un emplacement précis
- **latitude** - Latitude géographique (degrés décimaux)
- **longitude** - Longitude géographique (degrés décimaux)

Ces champs permettent :

- La cartographie des emplacements d'inventaire sur une vue de carte
- Les calculs de proximité pour la qualification de service
- L'analyse de couverture pour la planification du réseau
- L'optimisation des itinéraires pour l'envoi de techniciens sur le terrain

**Gestion des appareils et URL d'accès**

Le champ `management_url` fournit un accès rapide aux interfaces des appareils et aux URL de provisionnement :

- **management_url** - URL d'accès pour l'article d'inventaire
  - **Équipement réseau** : URL de l'interface Web (par exemple, `https://192.168.1.1` pour les routeurs, commutateurs, ONT, CPE)
  - **Profils eSIM** : adresse LPA (Local Profile Assistant) pour l'activation de l'eSIM (par exemple, `LPA:1$smdp.example.com$ACTIVATION-CODE-HERE`)
  - **Autres cas d'utilisation** : Toute URL qui doit être facilement accessible via un appareil mobile

  **Génération de QR Code**

  Lors de la visualisation des articles d'inventaire avec un `management_url`, le système génère automatiquement un **QR code scannable** :

- **Vue des détails de l'article d'inventaire** : QR code 128x128 affiché à côté de l'URL
- **Tableau d'inventaire des services** : QR code 64x64 affiché pour les articles attribués
- **Format** : QR code et lien cliquable affichés ensemble

**Cas d'utilisation courants**

- **Techniciens réseau** : Scanner le QR code pour accéder à l'interface de gestion des appareils sans taper d'adresses IP
- **Activation de l'eSIM** : Les clients scannent le QR code depuis le CRM pour installer le profil eSIM sur leur appareil
- **Service client autonome** : Fournir un accès facile à la configuration des appareils ou aux portails clients

- **management_username** - Nom d'utilisateur administrateur pour l'accès à l'appareil

- **management_password** - Mot de passe administrateur pour l'accès à l'appareil (chiffré au repos)

**Gestion de la configuration**

Pour les appareils avec des fichiers de configuration :

- **config_content** - Contenu complet du fichier de configuration (stocké sous forme de texte)
  - Utile pour la sauvegarde, la gestion des versions et la récupération après sinistre
  - Peut stocker des configurations de routeurs, de commutateurs, de CPE, etc.
- **config_file_path** - Chemin vers le fichier de configuration externe s'il est stocké séparément
  - Alternative au stockage de la configuration complète dans la base de données
  - Chemin vers un partage réseau, un dépôt de contrôle de version ou un système de gestion de configuration

**Notes et métadonnées**

- **inventory_notes** - Notes libres concernant l'article d'inventaire
  - Notes d'installation
  - Historique de maintenance
  - Particularités ou exigences de manipulation spéciales
  - Informations sur le fournisseur
  - Détails de garantie
- **created** - Horodatage lorsque l'article d'inventaire a été créé dans le système (défini automatiquement)
- **last_modified** - Horodatage de la dernière mise à jour de l'article d'inventaire (mis à jour automatiquement)

#### Exemples d'utilisation des champs

### Exemple 1 : Carte SIM mobile

```
{
  "inventory_id": 1001,
  "item": "Carte SIM",
  "inventory_template_id": 5,
  "itemtext1": "89612345678901234567",
  "itemtext2": "310120123456789",
  "itemtext3": "Physique",
  "item_location": "Entrepôt A, Étagère 3",
  "item_state": "Attribué",
  "customer_id": 456,
  "service_id": 789,
  "wholesale_cost": 2.50,
  "retail_cost": 10.00,
  "sold_date": "2025-01-15T10:30:00Z",
  "inventory_notes": "Activé le 2025-01-15"
}
```

### Exemple 2 : Profil eSIM mobile

```
{
  "inventory_id": 1002,
  "item": "eSIM",
  "inventory_template_id": 6,
  "itemtext1": "89612345678901234567",
  "itemtext2": "310120123456790",
  "itemtext3": "eSIM",
  "item_location": "Inventaire virtuel",
  "item_state": "Attribué",
  "customer_id": 457,
  "service_id": 790,
  "management_url": "LPA:1$smdp.example.com$ACTIVATION-CODE-
ABC123XYZ",
  "wholesale_cost": 0.00,
  "retail_cost": 0.00,
  "sold_date": "2025-01-16T14:20:00Z",
  "inventory_notes": "Profil eSIM prêt pour activation"
}
```


Lors de la visualisation de cet article d'inventaire eSIM, l'interface utilisateur affiche un QR code contenant l'adresse LPA. Les clients scannent ce QR code avec leur appareil mobile pour installer le profil eSIM.

**Exemple 3 : Équipement de client (CPE) - Sans fil fixe**

```
{
  "inventory_id": 2001,
  "item": "CPE sans fil fixe",
  "inventory_template_id": 10,
  "itemtext1": "AA:BB:CC:DD:EE:FF",
  "itemtext2": "FW2024-12345",
  "itemtext3": "v2.4.1",
  "itemtext4": "Ubiquiti",
  "itemtext5": "LiteBeam AC Gen2",
  "item_location": "Site du client",
  "item_state": "Attribué",
  "customer_id": 789,
  "service_id": 1234,
  "address_line_1": "123 Main Street",
  "address_line_2": "Apt 4B",
  "city": "Sydney",
  "state": "NSW",
  "zip_code": "2000",
  "country": "Australie",
  "latitude": "-33.8688",
  "longitude": "151.2093",
  "management_url": "https://192.168.100.1",
  "management_username": "admin",
  "management_password": "encrypted_password_here",
  "config_file_path": "/configs/cpe/fw2024-12345.conf",
  "inventory_notes": "Installé le 2025-01-10. Le client signale un
excellent signal. Orientation : Azimut 45°, Élévation 15°"
}
```


**Exemple 4 : ONT GPON avec adresse complète**

```
{
  "inventory_id": 3001,
  "item": "ONT GPON",
  "inventory_template_id": 15,
  "itemtext1": "ALCL12345678",
  "itemtext2": "AA:BB:CC:DD:EE:FF",
  "itemtext3": "OLT-1, PON 3, ONT 42",
  "itemtext4": "Nokia G-010G-A",
  "item_location": "Locaux du client",
  "item_state": "Attribué",
  "customer_id": 321,
  "service_id": 654,
  "address_line_1": "456 Fiber Lane",
  "city": "Melbourne",
  "state": "VIC",
  "zip_code": "3000",
  "country": "Australie",
  "google_maps_place_id": "ChIJ1234567890",
  "plus_code": "4RRH+2C Melbourne VIC",
  "latitude": "-37.8136",
  "longitude": "144.9631",
  "management_url": "https://192.168.1.1",
  "management_username": "admin",
  "config_content": "# Configuration de l'ONT\nwlan-ssid:
HomeNetwork\nwlan-password: encrypted...",
  "wholesale_cost": 45.00,
  "retail_cost": 0.00,
  "inventory_notes": "Provisionné le 2025-01-20. Puissance optique
: -22dBm"
}
```


**REMARQUE**

Lors de la visualisation des articles d'inventaire avec un `management_url` (comme dans les exemples 2, 3 et 4 ci-dessus), l'interface utilisateur affiche automatiquement :

- Un QR code scannable contenant l'URL ou l'adresse LPA
- Un lien cliquable (pour les URL web)

**Cas d'utilisation :**

- **Activation de l'eSIM** (Exemple 2) : Les clients scannent le QR code pour installer le profil eSIM sur leur appareil
- **Accès à l'équipement réseau** (Exemples 3 et 4) : Les techniciens scannent pour accéder aux interfaces de gestion des appareils sans taper manuellement les adresses IP

**Exemple 5 : Numéro de téléphone (Inventaire virtuel)**


```
{
  "inventory_id": 4001,
  "item": "Numéro de téléphone",
  "inventory_template_id": 20,
  "itemtext1": "+61412345678",
  "itemtext2": "Melbourne",
  "itemtext3": "Mobile",
  "item_location": "Australie - VIC",
  "item_state": "Attribué",
  "customer_id": 555,
  "service_id": 888,
  "wholesale_cost": 1.00,
  "retail_cost": 0.00,
  "inventory_notes": "Porté de Telstra le 2025-01-05"
}
```


#### États des articles d'inventaire expliqués

Le champ `item_state` suit le cycle de vie des articles d'inventaire :

- **Nouveau** → **Attribué** - Flux normal lors du provisionnement à un client
- **Attribué** → **Utilisé** - Après la désactivation du service, l'article peut être réutilisé
- **Nouveau** → **Utilisation interne** - Alloué pour des tests, des démonstrations ou un usage du personnel
- **Attribué** → **Endommagé** - L'article a échoué, nécessite un RMA ou une élimination
- **Tout état** → **Perdu** - L'article ne peut pas être localisé (déclenche un audit)
- **Tout état** → **Volé** - L'article a été volé (déclenche un rapport de sécurité)

Le filtrage et la recherche d'inventaire par état permettent aux opérateurs de :

- Suivre le stock disponible (articles neufs)
- Identifier les articles attribués aux clients (Attribué)
- Trouver les articles disponibles pour réutilisation (Utilisé)
- Surveiller les problèmes d'équipement (Endommagé, Hors service)
- Auditer les articles manquants (Perdu, Volé)

# Étiquettes Client

Les étiquettes sont des liens codés par couleur pratiques qui peuvent être ajoutés à un client pour aider à les catégoriser. Par exemple, un client pourrait avoir une étiquette pour "Ticket de Support Ouvert" ou "Facture Très En Retard" ou "Imbécile".

Pour stocker des métadonnées structurées et des données personnalisées clé-valeur, consultez Attributs Client.

Les étiquettes sont affichées sous forme de pilules sur la page de profil du client, et la couleur de la pilule est personnalisable ainsi que le lien.

Un cas d'utilisation courant est de taguer les clients qui ont un ticket de support ouvert, afin que l'équipe de support puisse facilement accéder au ticket ouvert depuis la page de profil du client.

Les étiquettes peuvent être créées dans le système par un administrateur via l'interface utilisateur ou par des systèmes tiers via l'API et peuvent avoir des dates de début et de fin, de sorte qu'elles peuvent être automatiquement supprimées après une certaine période.

# Gestion des Étiquettes via l'UI

## Affichage des Étiquettes Client

Pour afficher les étiquettes d'un client :

1. Accédez à la page de synthèse du client
2. Cliquez sur l'onglet **Étiquettes**
3. Vous verrez une liste de toutes les étiquettes actives pour le client, montrant :
   - Aperçu de l'étiquette avec la couleur configurée
   - Texte de l'étiquette
   - Date d'activation (quand l'étiquette devient visible)
   - Date de désactivation (quand l'étiquette sera cachée)
   - Lien (si configuré)

## Création d'une Nouvelle Étiquette

Pour créer une nouvelle étiquette pour un client :

1. Accédez à la page de synthèse du client
2. Cliquez sur l'onglet **Étiquettes**
3. Cliquez sur le bouton **Ajouter une Étiquette**
4. Remplissez les champs requis :
   - **Texte de l'Étiquette** (obligatoire) : Le texte qui sera affiché sur l'étiquette
   - **Couleur de l'Étiquette** (obligatoire) : Choisissez une couleur à l'aide du sélecteur de couleur ou entrez un code hexadécimal
   - **Lien de l'Étiquette** (facultatif) : URL qui s'ouvrira lorsque l'étiquette sera cliquée
   - **Date d'Activation** (obligatoire) : Date à laquelle l'étiquette doit commencer à être affichée
   - **Date de Désactivation** (obligatoire) : Date à laquelle l'étiquette doit cesser d'être affichée (par défaut au 2099-01-01)
5. Prévisualisez votre étiquette dans la section de prévisualisation

6. Cliquez sur **Créer une Étiquette**

## Édition d'une Étiquette

Pour modifier une étiquette existante :

1. Accédez à la page de synthèse du client
2. Cliquez sur l'onglet **Étiquettes**
3. Trouvez l'étiquette que vous souhaitez modifier dans la liste
4. Cliquez sur le bouton **Modifier** (crayon)
5. Modifiez les champs selon vos besoins
6. Cliquez sur **Mettre à Jour l'Étiquette**

## Suppression d'une Étiquette

Pour supprimer une étiquette :

1. Accédez à la page de synthèse du client
2. Cliquez sur l'onglet **Étiquettes**
3. Trouvez l'étiquette que vous souhaitez supprimer dans la liste
4. Cliquez sur le bouton **Supprimer** (poubelle)
5. Confirmez la suppression dans la fenêtre contextuelle

# Référence des Champs d'Étiquette

# Intégration API

Les étiquettes peuvent également être gérées de manière programmatique via l'API :

**Créer une Étiquette :**

```
PUT /crm/tag/
{
  "tag_text": "Client VIP",
  "tag_hex_color": "FFD700",
  "tag_link": "https://example.com/vip",
  "tag_active_date": "2025-01-01 00:00:00",
  "tag_deactivate_date": "2099-12-31 23:59:59",
  "customer_id": 12
}
```

**Mettre à Jour une Étiquette :**

```
PATCH /crm/tag/tag_id/{tag_id}
{
  "tag_text": "Texte de l'Étiquette Mis à Jour",
  "tag_hex_color": "FF0000"
}
```

**Obtenir des Étiquettes par Client :**

```
GET /crm/tag/customer_id/{customer_id}
```

**Supprimer une Étiquette :**

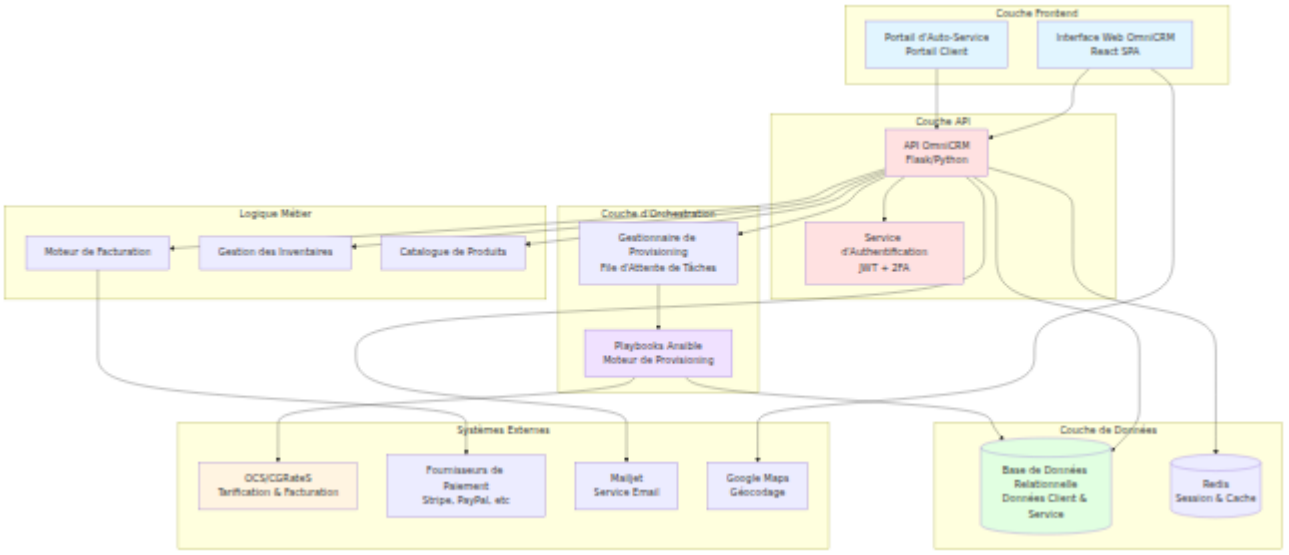
```
DELETE /crm/tag/tag_id/{tag_id}
```

# Architecture du Système OmniCRM

Ce document fournit un aperçu de l'architecture du système OmniCRM, y compris les relations entre les composants et le flux de données.

## Vue d'ensemble du système à haut niveau

OmniCRM est une plateforme BSS/OSS complète qui intègre plusieurs composants clés pour fournir une gestion complète des services pour les fournisseurs de télécommunications.




## Composants Principaux

### 1. Applications Frontend

#### Interface Web OmniCRM

- Application monopage React

- Interface pour le personnel pour la gestion des clients, le provisioning des services, la facturation
- Mises à jour en temps réel du statut de provisioning
- Contrôle d'accès basé sur les rôles

**Portail d'Auto-Service**

- Portail orienté client
- Gestion des services et suivi de l'utilisation
- Consultation et paiement des factures
- Code source partagé avec l'interface du personnel, vues différentes

# 2. Couche API

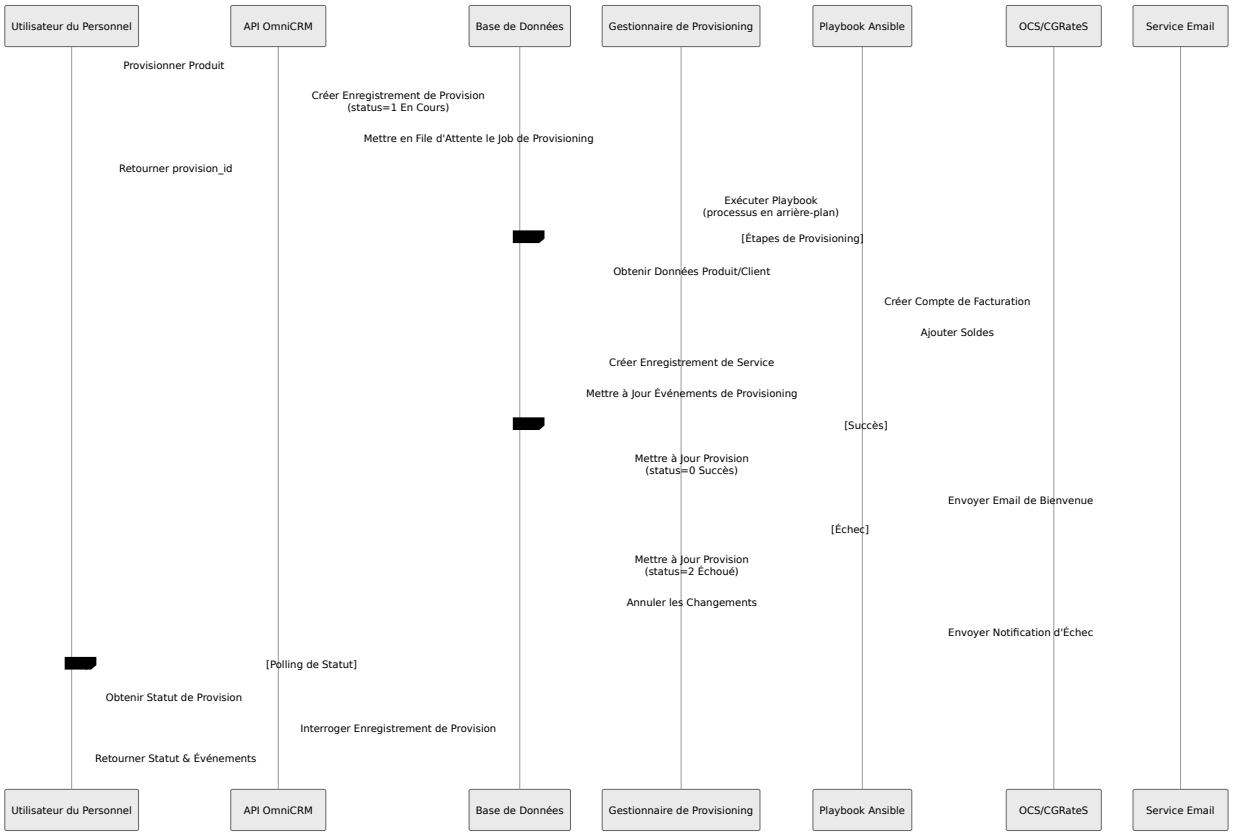
**API OmniCRM (Flask/Python)**

- API RESTful pour toutes les opérations
- Documentation OpenAPI/Swagger
- Authentification basée sur JWT
- Limitation de débit et mise en cache
- Support WebSocket pour les mises à jour en temps réel

**Routes API Clés :**

- `/crm/customer/*` - Gestion des clients
- `/crm/service/*` - Opérations sur les services
- `/crm/product/*` - Catalogue de produits
- `/crm/provision/*` - Opérations de provisioning
- `/crm/transaction/*` - Transactions de facturation
- `/crm/invoice/*` - Gestion des factures

# 3. Système de Provisioning

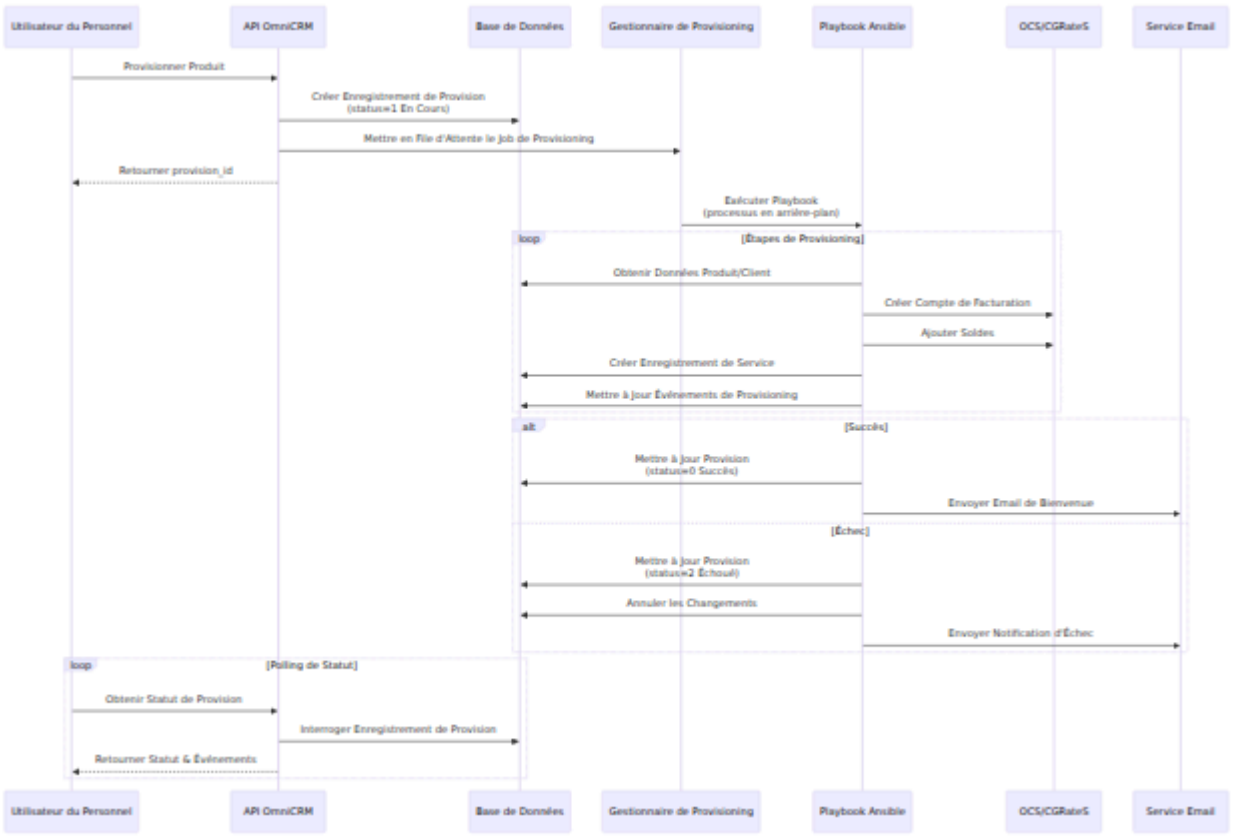



# 4. Facturation & Tarification

## Intégration OCS/CGRateS

- Tarification et facturation en temps réel
- Gestion des comptes
- Suivi des soldes (monétaire, données, voix, SMS)
- Plans d'action pour les charges récurrentes
- Notifications basées sur des seuils

## Flux de Travail de Facturation :




# 5. Modèle de Données

OmniCRM utilise une base de données relationnelle avec les modèles centraux suivants. Pour des diagrammes de relation d'entité visuels, voir Clients, Contacts, Sites & Services.

### Client & Entités Associées

**Client** - Entité centrale représentant une entreprise ou un individu

| Champ | Type | Description |
|---|---|---|
| customer_id | Integer (PK) | Identifiant unique |
| customer_name | String | Nom de l'entreprise ou de l'individu |
| customer_account_type | Enum | 'Individuel' ou 'Entreprise' |
| customer_status | Enum | 'Ouvert', 'Fermé', 'Suspendu', 'Archivé' |
| customer_payment_type | String | 'prépayé' ou 'postpayé' |
| customer_enabled | Boolean | Le compte est-il actif |
| tax_identifier | String | Numéro de TVA/GST |
| contract_start_date | DateTime | Début du contrat |
| contract_end_date | DateTime | Fin du contrat |

**Contact** - Personnes associées à un client

| Champ | Type | Description |
|---|---|---|
| contact_id | Integer (PK) | Identifiant unique |
| customer_id | Integer (FK) | Client parent |
| contact_firstname | String | Prénom |
| contact_lastname | String | Nom de famille |
| contact_email | String | Adresse email |
| contact_phone | String | Numéro de téléphone |
| contact_types | String | 'Principal', 'Facturation', 'Technique' |

**Site** - Lieux physiques de livraison de services

| Champ | Type | Description |
|---|---|---|
| site_id | Integer (PK) | Identifiant unique |
| customer_id | Integer (FK) | Client parent |
| site_name | String | Nom de l'emplacement |
| address_line_1 | String | Adresse de la rue |
| city, state, zip_code | String | Détails de l'emplacement |
| latitude, longitude | Float | Coordonnées GPS |
| google_maps_place_id | String | Référence Google Maps |
| plus_code | String | Code d'Emplacement Ouvert |

**Modèles de Service & Produit**

**Service** - Instances de service actives

| Champ | Type | Description |
|---|---|---|
| service_id | Integer (PK) | Identifiant unique |
| customer_id | Integer (FK) | Client parent |
| product_id | Integer (FK) | Modèle de produit |
| site_id | Integer (FK) | Emplacement du service |
| service_name | String | Nom d'affichage |
| service_uuid | String | Identifiant du système de facturation |
| service_status | Enum | Statut actuel |
| service_billed | Boolean | Générer des transactions |
| wholesale_cost | Float | Coût fournisseur |
| retail_cost | Float | Prix client |
| bundled_parent | Integer (FK) | Service parent pour les bundles |

**Produit** - Modèles d'offres de services

| Champ | Type | Description |
| --- | --- | --- |
| product_id | Integer (PK) | Identifiant unique |
| product_name | String | Nom d'affichage |
| product_slug | String | Nom adapté aux URL |
| category | Enum | 'autonome', 'bundle', 'addon', 'promo' |
| provisioning_play | String | Nom du playbook Ansible |
| provisioning_json_vars | JSON | Variables du playbook |
| inventory_items_list | String | Inventaire requis |
| retail_cost | Float | Prix mensuel |
| retail_setup_cost | Float | Frais uniques |
| enabled | Boolean | Disponible à la vente |

**Modèles de Facturation**

**Transaction** - Charges/crédits individuels

| Champ | Type | Description |
|---|---|---|
| transaction_id | Integer (PK) | Identifiant unique |
| customer_id | Integer (FK) | Client parent |
| invoice_id | Integer (FK) | Facture parent (optionnel) |
| service_id | Integer (FK) | Service lié |
| title | String | Description de la transaction |
| retail_cost | Float | Montant (négatif = crédit/paiement) |
| payment_type | String | par ex. `wallet_topup_card`, `wallet_invoice`, `postpaid_deferred_charge` |
| wallet_ledger_id | Integer (FK) | Lien vers wallet_ledger lorsque la ligne reflète le mouvement du portefeuille |
| void | Boolean | Transaction annulée |

**Facture** - Transactions groupées pour la facturation

| Champ | Type | Description |
|---|---|---|
| invoice_id | Integer (PK) | Identifiant unique |
| customer_id | Integer (FK) | Client parent |
| paid | Boolean | Paiement reçu |
| void | Boolean | Facture annulée |
| payment_reference | String | ID de transaction Stripe |
| start_date, end_date | Date | Période de facturation |
| due_date | Date | Date limite de paiement |
| retail_cost | Float | Montant total |

**Modèles d'Inventaire**

**Inventaire** - Actifs physiques et virtuels

| Champ | Type | Description |
|---|---|---|
| inventory_id | Integer (PK) | Identifiant unique |
| customer_id | Integer (FK) | Client assigné |
| service_id | Integer (FK) | Service lié |
| inventory_template_id | Integer (FK) | Modèle de type d'élément |
| item | String | Type d'élément (Carte SIM, Routeur, etc.) |
| item_state | Enum | 'Nouveau', 'Assigné', 'Utilisé', etc. |
| itemtext1-20 | String | Champs flexibles |
| management_url | String | URL d'administration de l'équipement |
| config_content | Text | Fichier de configuration |

**Inventory_Template** - Définit la structure des éléments d'inventaire

| Champ | Type | Description |
|---|---|---|
| inventory_template_id | Integer (PK) | Identifiant unique |
| item | String | Nom du modèle |
| itemtext1_label | String | Étiquette pour le champ itemtext1 |
| itemtext2_label | String | Étiquette pour le champ itemtext2 |

**Modèles de Provisioning**

**Provision** - Suivi des jobs de provisioning

| Champ | Type | Description |
|---|---|---|
| provision_id | Integer (PK) | Identifiant unique |
| product_id | Integer (FK) | Produit en cours de provisioning |
| customer_id | Integer (FK) | Client cible |
| service_id | Integer (FK) | Service créé/modifié |
| provisioning_play | String | Nom du playbook Ansible |
| provisioning_status | Integer | 0=Succès, 1=En Cours, 2=Échoué |

**Provision_Event** - Étapes de provisioning individuelles

| Champ | Type | Description |
|---|---|---|
| provision_event_id | Integer (PK) | Identifiant unique |
| provision_id | Integer (FK) | Job de provision parent |
| event_name | String | Nom de la tâche |
| event_number | Integer | Numéro de séquence |
| provisioning_status | Integer | 0=Succès, 1=En Cours, 2=Échoué |
| provisioning_result_json | JSON | Sortie complète de la tâche |

**Modèles Utilisateur & Sécurité**

**Utilisateur** - Comptes utilisateurs

| Champ | Type | Description |
|---|---|---|
| id | Integer (PK) | Identifiant unique |
| username | String | Nom d'utilisateur de connexion |
| email | String | Adresse email |
| email_verified | Boolean | Email confirmé |
| is_2fa_enabled | Boolean | Authentification à deux facteurs activée |
| totp_secret | String | Clé secrète TOTP |

**Rôle** - Rôles utilisateurs

| Champ | Type | Description |
|---|---|---|
| id | Integer (PK) | Identifiant unique |
| name | String | Nom du rôle |
| description | String | Description du rôle |

**Permission** - Permissions granulaires

| Champ | Type | Description |
|---|---|---|
| id | Integer (PK) | Identifiant unique |
| name | String | Nom de la permission (ressource.action) |
| description | String | Description de la permission |

**Relations :**

- Les utilisateurs ont plusieurs rôles (many-to-many)
- Les rôles ont plusieurs permissions (many-to-many)
- Les utilisateurs peuvent être liés à un contact (pour l'accès au portail client)

# Points d'Intégration

## Passerelle de Paiement Stripe

- Méthodes de paiement tokenisées
- Stockage de cartes conforme PCI
- Paiement automatisé des factures
- Traitement des remboursements
- Notifications de cartes expirantes

## Service Email Mailjet

- Emails transactionnels (factures, bienvenue, notifications)
- Synchronisation des contacts
- Gestion des modèles
- Suivi de livraison

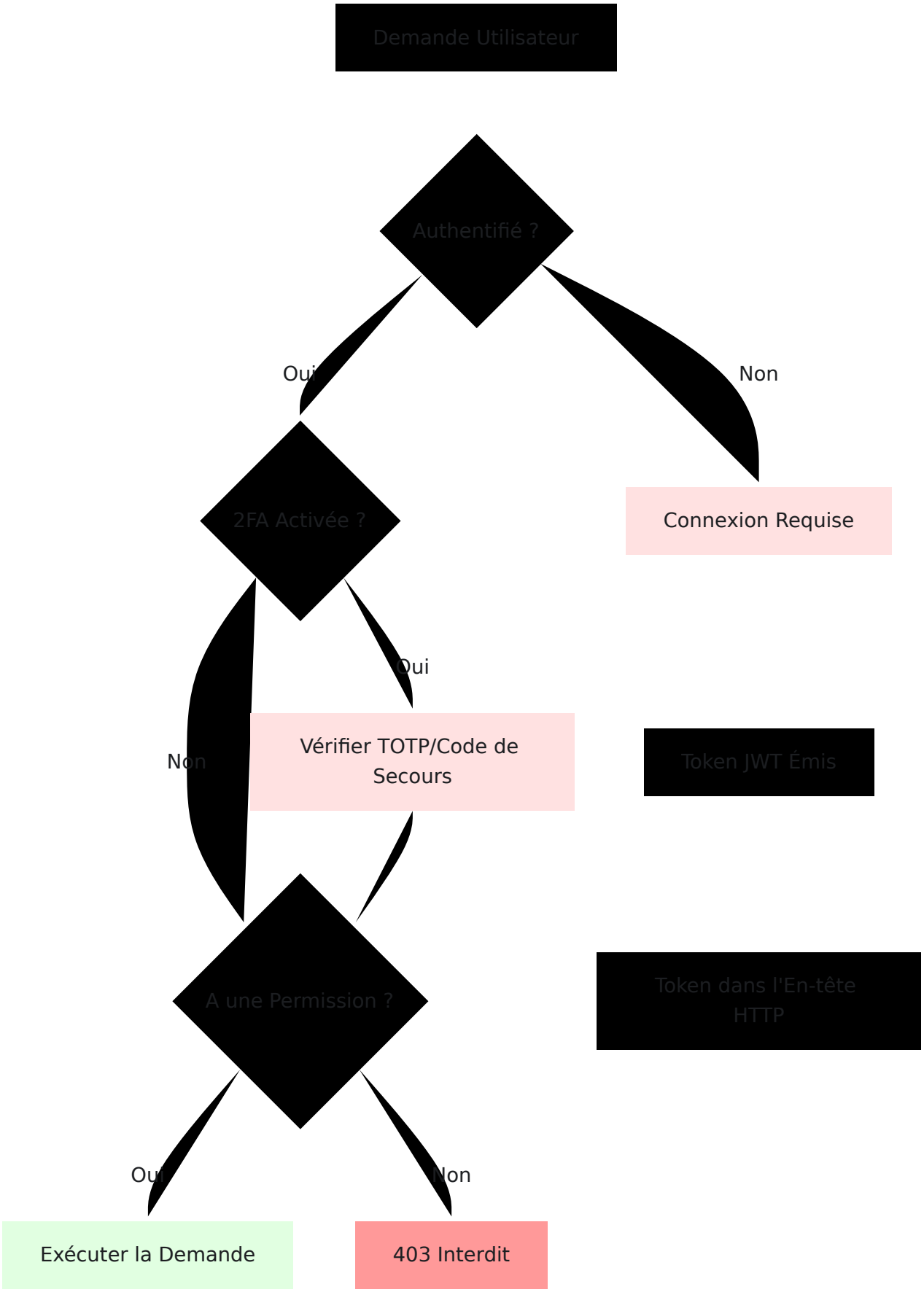
## Google Maps

- Autocomplétion d'adresses
- Géocodage et géocodage inverse
- Génération de Plus Code
- Cartographie des emplacements des sites

## OCS/CGRateS

- Provisioning de comptes
- Tarification en temps réel
- Gestion des soldes
- Traitement des CDR
- Plans d'action et horaires

# Architecture de Sécurité

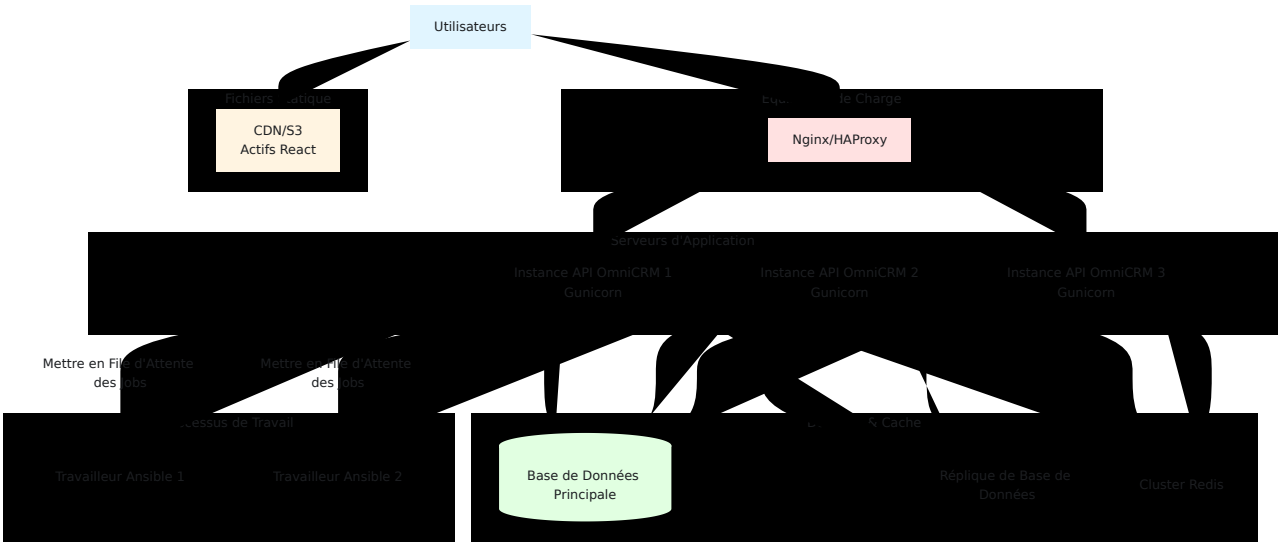


**Fonctionnalités de Sécurité :**

- Authentification basée sur JWT
- Authentification à deux facteurs (TOTP)
- Contrôle d'accès basé sur les rôles (RBAC)
- Accès aux ressources basé sur les permissions
- Gestion des sessions avec Redis
- Hachage des mots de passe (bcrypt)
- Vérification par email
- Journalisation des audits via le Journal d'Activité

# Architecture de Déploiement

**Configuration de Production Recommandée :**




# Pile Technologique

**Backend :**

- Python 3.x
- Framework web Flask
- ORM SQLAlchemy
- Migrations Alembic

- Ansible pour l'automatisation

**Frontend :**

- React
- Gestion d'état
- React Router
- Axios pour les appels API

**Base de Données :**

- Base de données relationnelle (stockage de données principal)
- Redis (cache & sessions)

**Services Externes :**

- CGRateS (facturation/tarification)
- Stripe (paiements)
- Mailjet (email)
- Google Maps (géocodage)

# Considérations de Scalabilité

**Scalabilité Horizontale :**

- Conception d'API sans état permettant plusieurs instances
- L'équilibreur de charge distribue les requêtes
- Redis pour l'état de session partagé

**Scalabilité de la Base de Données :**

- Répliques de lecture pour les requêtes de reporting
- Pooling de connexions
- Optimisation des requêtes et indexation

**Provisioning à Grande Échelle :**

- Traitement des jobs en arrière-plan
- Plusieurs travailleurs Ansible
- Gestion de la file d'attente des jobs
- Logique de réessai pour les provisions échouées

# Surveillance & Observabilité

OmniCRM fournit des métriques complètes basées sur Prometheus pour surveiller tous les aspects du système. Pour des détails complets, voir Surveillance & Métriques.

**Métriques Clés :**

- Temps de réponse de l'API et taux de requêtes
- Taux de succès/échec de provisioning et durée des jobs
- Performance des requêtes de base de données et utilisation du pool de connexions
- Santé des intégrations externes (OCS, Stripe, Mailjet)
- Exécution des jobs en arrière-plan et erreurs

**Journalisation :**

- Journaux d'application (Flask)
- Journaux de provisioning (sortie Ansible)
- Journal d'activité (trace d'audit)
- Suivi des erreurs et alertes

**Point de Terminaison des Métriques :** Toutes les métriques sont exposées à `/crm/metrics` au format Prometheus. Voir Surveillance & Métriques pour la configuration de scraping, les règles d'alerte et des exemples de tableau de bord.

# Documentation Connexe

- Playbooks Ansible - Automatisation du provisioning

- Système de Provisioning - Détails du flux de travail
- Produits et Services - Architecture des produits
- Documentation API - Référence API
- RBAC - Sécurité et permissions

# Flux d'authentification et contrôles administratifs

OmniCRM fournit des fonctionnalités d'authentification complètes, y compris la connexion, l'authentification à deux facteurs (2FA), la gestion des mots de passe et les contrôles administratifs pour gérer la sécurité des utilisateurs. Ce guide se concentre sur les flux de travail de l'interface utilisateur pour les utilisateurs finaux et les administrateurs.

Voir aussi : Portail de Self-Care pour la connexion des clients et l'accès au portail, RBAC pour les autorisations du personnel.

## Aperçu

L'authentification OmniCRM comprend :

- **Connexion par e-mail/mot de passe** - Authentification standard basée sur des identifiants
- **Authentification à deux facteurs (2FA)** - Deuxième facteur basé sur TOTP facultatif
- **Se souvenir de moi** - Session prolongée jusqu'à 30 jours
- **Réinitialisation de mot de passe** - Récupération de mot de passe en libre-service par e-mail
- **Contrôles administratifs** - Outils administratifs pour réinitialiser 2FA et mots de passe
- **Connexions sociales** - Intégration Google, Apple, Facebook facultative (si activée)
- **Navigation basée sur les rôles** - Routage automatique basé sur les rôles des utilisateurs

# Flux de connexion

La page de connexion est le point d'entrée pour tous les utilisateurs (personnel et clients).

## Connexion standard

**Processus de connexion :**

1. Entrez **l'adresse e-mail** (e-mail du personnel ou du client)
2. Entrez **le mot de passe**
3. Facultatif : Cochez **"Se souvenir de moi pendant 30 jours"** pour une session prolongée
4. Cliquez sur **"Connexion"**

**Ce qui se passe ensuite :**

- **Sans 2FA :** Utilisateur connecté immédiatement, navigué en fonction du rôle :
  - **Clients** → Portail de Self-Care (`/self-care`)
  - **Personnel/Admins** → Tableau de bord des clients (`/customers`)
  - **Mode CBC** → Interface de diffusion cellulaire (`/create-cell-broadcast`)
- **Avec 2FA activé :** Redirigé vers l'écran de vérification 2FA

**Fonction "Se souvenir de moi" :**

Lorsqu'elle est activée, la session persiste pendant **30 jours** au lieu d'expirer lorsque le navigateur se ferme. Utilise des cookies sécurisés uniquement HTTP.

**Afficher/Masquer le mot de passe :**

Cliquez sur l'**icône de l'œil (□)** pour basculer la visibilité du mot de passe.

# Connexion avec 2FA

Si l'utilisateur a 2FA activé, après avoir entré l'e-mail/le mot de passe, l'écran de défi 2FA apparaît :

**Utilisation du code d'authentification :**

1. Ouvrez l'application d'authentification (Google Authenticator, Authy, etc.)
2. Trouvez l'entrée OmniCRM
3. Entrez le code à 6 chiffres
4. Le code est automatiquement soumis lorsque les 6 chiffres sont entrés
5. Si valide, l'utilisateur est connecté et navigué vers le tableau de bord approprié

**Utilisation du code de récupération :**

Si l'application d'authentification n'est pas disponible :

1. Cliquez sur l'onglet **"Code de récupération"**
2. Entrez l'un de vos codes de sauvegarde enregistrés (par exemple, `3fa5b9c2`)
3. Cliquez sur **"Vérifier"**
4. Le code est consommé (ne peut être utilisé qu'une seule fois)

**Annuler :**

Cliquez sur **"Annuler"** pour revenir à la page de connexion.

## Connexions sociales (facultatif)

Si activé (`VITE_ALLOW_SOCIAL_LOGINS=yes`), des boutons de connexion sociale apparaissent :

> [ Se connecter avec Google ] [• Se connecter avec Apple ] [ Se connecter avec Facebook]

Cliquez sur n'importe quel bouton pour vous authentifier via ce fournisseur. Affiche actuellement un message "bientôt disponible" (mise en œuvre de la connexion sociale en cours).

## Lien "Mot de passe oublié"

Cliquez sur le lien **"Mot de passe oublié ?"** pour initier le flux de réinitialisation du mot de passe.

# Configuration de l'authentification à deux facteurs (2FA)

Les utilisateurs peuvent activer 2FA pour une sécurité accrue de leur compte. 2FA utilise TOTP (Mot de passe à usage unique basé sur le temps) compatible avec les applications d'authentification standard.

# Accéder à la configuration de 2FA

Depuis le profil utilisateur ou les paramètres :

**Remarque pour les clients :**

Les utilisateurs de rôle client ne voient pas les invites 2FA. La 2FA est généralement requise uniquement pour le personnel et les utilisateurs administratifs.

# Étape 1 : Confirmer le mot de passe

> Mot de passe actuel
>
> [Annuler] [Continuer]

Entrez votre mot de passe actuel pour continuer. Cela vérifie votre identité avant d'activer 2FA.

# Étape 2 : Scanner le code QR

> [Annuler] [Confirmer]

**Instructions de configuration :**

1. **Téléchargez l'application d'authentification** (si vous n'en avez pas) :
   - iOS : Apple App Store → "Google Authenticator"
   - Android : Google Play → "Google Authenticator"
   - Alternatives : Authy, Microsoft Authenticator, 1Password
2. **Scanner le code QR :**
   - Ouvrez l'application d'authentification
   - Appuyez sur "+" ou "Ajouter un compte"
   - Choisissez "Scanner le code QR"
   - Pointez la caméra vers le code QR à l'écran
   - L'application ajoute l'entrée "OmniCRM" avec le code à 6 chiffres
3. **Sauvegarder les codes de sauvegarde :**
   - **CRITIQUE :** Notez ou copiez ces 8 codes

- Conservez-les dans un endroit sécurisé (gestionnaire de mots de passe, coffre-fort, etc.)
- Chaque code est à usage unique
- Utilisé si vous perdez l'accès à l'application d'authentification
- Cliquez sur **"Copier les codes"** pour copier tous les codes dans le presse-papiers

4. **Vérifier la configuration :**
   - Entrez le code à 6 chiffres actuel de l'application d'authentification
   - Cliquez sur **"Confirmer"**
   - Si valide, 2FA est maintenant activé

# Étape 3 : 2FA activé

Un message de succès apparaît :

Désormais, la connexion nécessite à la fois le mot de passe et le code 2FA.

# Flux de réinitialisation de mot de passe (auto-service)

Les utilisateurs qui oublient leur mot de passe peuvent le réinitialiser par e-mail.

## Étape 1 : Demander un lien de réinitialisation

Depuis la page de connexion, cliquez sur **"Mot de passe oublié ?"**

1. Entrez **l'adresse e-mail**
2. Cliquez sur **"Envoyer le lien de réinitialisation"**

**Ce qui se passe :**

- Le système vérifie si l'e-mail existe dans la base de données
- Si trouvé, envoie un e-mail de réinitialisation de mot de passe via Mailjet

- L'e-mail contient un lien de réinitialisation à durée limitée (généralement 1 heure d'expiration)
- Un message de succès apparaît : "Les instructions de réinitialisation ont été envoyées à votre e-mail"

**Si l'e-mail n'est pas trouvé :**

Message d'erreur : "Aucun compte trouvé avec cette adresse e-mail"

## Étape 2 : Vérifier l'e-mail

L'utilisateur reçoit un e-mail avec un sujet comme :

> Bonjour [Nom],
>
> Vous avez demandé une réinitialisation de mot de passe pour votre compte OmniCRM.
>
> Cliquez sur le lien ci-dessous pour réinitialiser votre mot de passe : <https://yourcompany.com/reset-password/abc123token456>
>
> Ce lien expire dans 1 heure.
>
> Si vous n'avez pas demandé cela, ignorez cet e-mail.

Cliquez sur le lien de réinitialisation pour continuer.

## Étape 3 : Définir un nouveau mot de passe

Le lien de réinitialisation ouvre la page de création de mot de passe :

> □ (icône de cadenas)
>
> Mot de passe
>
> Confirmer le mot de passe
>
> [Réinitialiser le mot de passe]

1. Entrez **le nouveau mot de passe**

2. Réentrez dans le champ **Confirmer le mot de passe**
3. Cliquez sur **"Réinitialiser le mot de passe"**

**Exigences de mot de passe :**

- Longueur minimale (généralement 8 caractères ou plus)
- Les mots de passe doivent correspondre

**Succès :**

- Message de succès : "Le mot de passe a été réinitialisé avec succès"
- Redirection automatique vers la page de connexion
- L'utilisateur peut maintenant se connecter avec le nouveau mot de passe

**Jeton expiré/invalide :**

Si le lien de réinitialisation est expiré ou invalide :

> [Demander un nouveau lien de réinitialisation]

# Contrôles administratifs pour la gestion des utilisateurs

Les administrateurs ayant les autorisations appropriées peuvent gérer les paramètres d'authentification des utilisateurs depuis l'interface de gestion des utilisateurs.

## Accéder à la gestion des utilisateurs

Affiche un tableau de tous les utilisateurs avec des boutons d'action.

> Nom E-mail Téléphone Actions John Smith <john@example.com> +44 123... ✏ □ □ □ Jane Doe <jane@example.com> +44 456... ✏ □ □ Bob Wilson <bob@example.com> +44 789... ✏ □ □ □ ✉

**Icônes d'action :**

- ✏ **Modifier** - Modifier les détails de l'utilisateur, les rôles, les autorisations

- **Supprimer** - Supprimer le compte utilisateur
- **Réinitialiser le mot de passe** - Générer un mot de passe temporaire
- **Réinitialiser 2FA** - Désactiver 2FA pour l'utilisateur (affiché uniquement si 2FA activé)
- ✉ **Envoyer un e-mail de bienvenue** - Renvoyer l'e-mail de bienvenue (affiché uniquement si l'utilisateur ne s'est jamais connecté)

# Admin : Réinitialiser le mot de passe de l'utilisateur

Lorsque l'utilisateur oublie son mot de passe et que l'administrateur doit aider :

**Étape 1 : Cliquez sur l'icône de réinitialisation du mot de passe ()**

Une fenêtre de confirmation apparaît :

> Êtes-vous sûr de vouloir réinitialiser le mot de passe pour :
>
> Utilisateur : John Smith (<john@example.com>)
>
> Un mot de passe temporaire sera généré et affiché. L'utilisateur doit changer ce mot de passe lors de sa prochaine connexion.
>
> [Annuler] [Réinitialiser le mot de passe]

**Étape 2 : Confirmer la réinitialisation**

Cliquez sur **"Réinitialiser le mot de passe"**. Le système génère un mot de passe temporaire sécurisé.

**Étape 3 : Mot de passe temporaire affiché**

> Mot de passe temporaire pour John Smith :
>
> [ Copier le mot de passe]
>
> ⚠ IMPORTANT : • Envoyez ce mot de passe à l'utilisateur via un canal sécurisé • Ne pas envoyer par e-mail ou messagerie non sécurisée • L'utilisateur sera contraint de changer le mot de passe lors de sa prochaine connexion

> [Fermer]

**Action de l'administrateur :**

- Copier le mot de passe temporaire
- Appeler l'utilisateur ou communiquer par méthode sécurisée
- Fournir le mot de passe temporaire verbalement
- Indiquer à l'utilisateur de se connecter et de changer le mot de passe

**Expérience utilisateur :**

Lorsque l'utilisateur se connecte avec le mot de passe temporaire :

1. La connexion réussit
2. Redirection immédiate vers l'écran "Changer le mot de passe"
3. Doit définir un nouveau mot de passe avant d'accéder au système
4. Ne peut pas sauter le changement de mot de passe

# Admin : Réinitialiser 2FA de l'utilisateur

Lorsque l'utilisateur perd l'accès à l'application d'authentification et à tous les codes de sauvegarde :

**Étape 1 : Cliquez sur l'icône de réinitialisation 2FA (□)**

N'apparaît que pour les utilisateurs ayant 2FA actuellement activé.

Une fenêtre de confirmation apparaît :

**Étape 2 : Confirmer la réinitialisation**

Cliquez sur **"Réinitialiser 2FA"**

**Étape 3 : Confirmation**

Message de succès :

> John Smith peut maintenant se connecter uniquement avec son mot de passe. Il peut réactiver 2FA depuis ses paramètres utilisateur.

**Expérience utilisateur :**

- L'utilisateur peut maintenant se connecter uniquement avec le mot de passe (aucun code 2FA requis)
- L'icône de bouclier 2FA (□) disparaît de la ligne de l'utilisateur dans le tableau d'administration
- L'utilisateur peut réactiver 2FA volontairement depuis ses paramètres

**Remarque de sécurité importante :**

Avant de réinitialiser 2FA, l'administrateur doit :

1. Vérifier l'identité de l'utilisateur par des moyens alternatifs :
    - Vérification de l'identité gouvernementale
    - Questions de sécurité

    - Vérification de transaction récente
    - Vérification en personne (si applicable)
2. Documenter la réinitialisation dans les notes du client
3. Informer l'utilisateur de réactiver 2FA après avoir retrouvé l'accès

# Admin : Envoyer un e-mail de bienvenue

Pour les utilisateurs qui n'ont pas reçu ou ont perdu leur e-mail de bienvenue :

**Lorsqu'il est disponible :**

L'icône de l'avion en papier (✉) n'apparaît que pour les utilisateurs qui **ne se sont jamais connectés** (`login_count = 0`).

**Cliquez sur l'icône d'envoi d'e-mail de bienvenue (✉)**

> Envoyer un e-mail de bienvenue à :
>
> Utilisateur : Bob Wilson (<bob@example.com>)
>
> L'e-mail inclura : • Message de bienvenue • Instructions de connexion • Lien pour définir le mot de passe initial (le cas échéant) • Informations de contact pour le support
>
> [Annuler] [Envoyer l'e-mail]

Cliquez sur **"Envoyer l'e-mail"**

Message de succès :

**E-mail envoyé via Mailjet :**

Utilise le modèle : `api_crmCommunicationUserWelcome`

# Admin : Modifier l'utilisateur

Cliquez sur **l'icône Modifier (✏)** pour modifier les détails de l'utilisateur :

> Prénom

> Nom de famille
>
> E-mail
>
> Numéro de téléphone
>
> Rôles ☑ admin ☐ customer_service_agent_1 ☐ client
>
> [Annuler] [Enregistrer les modifications]

**Champs modifiables :**

- Nom, e-mail, téléphone
- **Rôles** - Attribuer/retirer des rôles (affecte les autorisations)
- Statut actif/inactif

# Admin : Supprimer l'utilisateur

Cliquez sur **l'icône Supprimer (🗑)** pour retirer l'utilisateur :

> Êtes-vous sûr de vouloir supprimer :
>
> Utilisateur : John Smith (<john@example.com>)
>
> ⚠ AVERTISSEMENT : Cette action ne peut pas être annulée.
>
> Cela supprimera définitivement : • Compte utilisateur et identifiants • Paramètres 2FA • Historique des sessions
>
> Les données et transactions des clients NE seront PAS supprimées.
>
> [Annuler] [Supprimer l'utilisateur]

Cliquez sur **"Supprimer l'utilisateur"** pour confirmer.

Message de succès :

# Meilleures pratiques

## Pour les utilisateurs finaux

**Sécurité de la connexion :**

- Utilisez des mots de passe forts et uniques
- Activez "Se souvenir de moi" uniquement sur des appareils personnels
- Déconnectez-vous toujours sur des ordinateurs partagés
- Activez 2FA pour une sécurité supplémentaire

**Gestion de 2FA :**

- Sauvegardez les codes de sauvegarde immédiatement après avoir activé 2FA
- Conservez les codes dans un gestionnaire de mots de passe ou un endroit sécurisé
- Testez un code de sauvegarde pour vous assurer qu'il fonctionne
- Régénérez les codes de sauvegarde si vous en utilisez plusieurs
- Contactez l'administrateur si vous perdez à la fois l'authentificateur et les codes de sauvegarde

**Gestion des mots de passe :**

- Utilisez un gestionnaire de mots de passe pour générer et stocker des mots de passe
- Ne partagez jamais de mots de passe par e-mail ou messagerie
- Changez de mot de passe si vous soupçonnez un compromis
- Utilisez un mot de passe unique pour OmniCRM (ne pas réutiliser les mots de passe)

## Pour les administrateurs

**Gestion de la sécurité des utilisateurs :**

- Vérifiez l'identité de l'utilisateur avant de réinitialiser 2FA ou les mots de passe
- Ne jamais envoyer de mots de passe temporaires par e-mail
- Documentez toutes les réinitialisations de sécurité dans les notes des utilisateurs
- Encouragez le personnel à activer 2FA
- Surveillez les modèles de connexion inhabituels

**Réinitialisations de mot de passe :**

- Communiquez les mots de passe temporaires uniquement par téléphone ou en personne
- Générez des mots de passe temporaires forts (le système le fait automatiquement)
- Assurez-vous que l'utilisateur change de mot de passe lors de la première connexion
- Ne réinitialisez pas les mots de passe inutilement - utilisez le flux de réinitialisation par e-mail lorsque cela est possible

**Réinitialisations de 2FA :**

- Traitez les réinitialisations de 2FA comme des actions de haute sécurité
- Vérifiez l'identité par plusieurs canaux avant de réinitialiser
- Documentez la raison de la réinitialisation
- Encouragez l'utilisateur à réactiver 2FA immédiatement après avoir retrouvé l'accès
- Envisagez d'exiger 2FA pour tous les utilisateurs administratifs

**Gestion des utilisateurs :**

- Passez régulièrement en revue la liste des utilisateurs pour les comptes inactifs
- Supprimez les utilisateurs qui ont quitté l'organisation
- Assurez-vous que les attributions de rôles sont appropriées
- Surveillez les utilisateurs qui ne se sont jamais connectés
- Auditez les autorisations des utilisateurs trimestriellement

# Dépannage

**Erreur "E-mail ou mot de passe invalide"**

- **Cause :** Identifiants incorrects
- **Solution :**
  - Vérifiez que l'adresse e-mail est correcte
  - Assurez-vous que la touche majuscule est désactivée
  - Essayez la réinitialisation du mot de passe si oublié
  - Contactez l'administrateur si le compte est verrouillé

**Code 2FA non accepté**

- **Cause :** Problème de synchronisation horaire ou code incorrect
- **Solution :**
  - Assurez-vous que l'heure de l'appareil est correcte (Paramètres → Date & Heure → Automatique)
  - Attendez que le code se rafraîchisse (les codes changent toutes les 30 secondes)
  - Essayez le code suivant qui apparaît
  - Utilisez le code de sauvegarde si l'authentificateur ne fonctionne pas
  - Contactez l'administrateur pour réinitialiser 2FA si tout échoue

**"Se souvenir de moi" ne fonctionne pas**

- **Cause :** Cookies désactivés ou effacés
- **Solution :**
  - Activez les cookies dans les paramètres du navigateur
  - Ne pas effacer les cookies lorsque vous fermez le navigateur
  - Désactivez les extensions de confidentialité pour le domaine OmniCRM
  - Essayez un autre navigateur

**E-mail de réinitialisation de mot de passe non reçu**

- **Cause :** E-mail non envoyé, filtre anti-spam ou mauvaise adresse e-mail
- **Solution :**
  - Vérifiez le dossier spam/junk

    - Vérifiez que l'adresse e-mail est correcte
    - Attendez 5-10 minutes (la livraison des e-mails peut être retardée)
    - Vérifiez que l'intégration Mailjet fonctionne (administrateur)
    - Contactez l'administrateur pour une réinitialisation manuelle du mot de passe

**Lien de réinitialisation de mot de passe expiré**

- **Cause :** Jeton expiré (généralement 1 heure)
- **Solution :**
    - Demandez une nouvelle réinitialisation de mot de passe
    - Vérifiez l'e-mail et cliquez rapidement sur le lien
    - Contactez l'administrateur en cas de problèmes répétés

**Impossible d'activer 2FA (mot de passe incorrect)**

- **Cause :** Mot de passe actuel saisi incorrectement
- **Solution :**
    - Vérifiez le mot de passe actuel
    - Réinitialisez le mot de passe d'abord si incertain
    - Contactez l'administrateur pour obtenir de l'aide

**Application d'authentification et codes de sauvegarde perdus**

- **Cause :** Téléphone perdu/réinitialisé, codes de sauvegarde non sauvegardés
- **Solution :**
    - Contactez immédiatement l'administrateur
    - L'administrateur vérifiera l'identité et réinitialisera 2FA
    - Connectez-vous uniquement avec le mot de passe après la réinitialisation
    - Réactivez 2FA et SAUVEGARDEZ les codes de sauvegarde cette fois

**Admin : Erreur "Échec de la réinitialisation de 2FA"**

- **Cause :** Permissions insuffisantes
- **Solution :**

- Assurez-vous d'avoir le rôle d'administrateur
- Vérifiez les autorisations API
- Contactez l'administrateur système

**Admin : Mot de passe temporaire non généré**

- **Cause :** Erreur API ou problème de permissions
- **Solution :**
  - Actualisez la page et réessayez
  - Vérifiez les permissions administratives
  - Vérifiez les journaux API pour des erreurs
  - Assurez-vous que la base de données est accessible

# Considérations de sécurité

**Gestion des sessions :**

- Les sessions expirent après une période d'inactivité
- "Se souvenir de moi" prolonge la session à 30 jours
- Les sessions sont stockées sous forme de cookies HTTP uniquement (non accessibles par JavaScript)
- Le drapeau sécurisé garantit que les cookies ne sont envoyés que sur HTTPS

**Sécurité des mots de passe :**

- Les mots de passe sont hachés à l'aide d'algorithmes standard de l'industrie
- Les mots de passe en texte clair ne sont jamais stockés
- Les mots de passe temporaires expirent automatiquement après la première utilisation
- Les tentatives de connexion échouées sont suivies (limitation de taux potentielle)

**Sécurité de 2FA :**

- Les secrets TOTP sont chiffrés dans la base de données
- Les codes QR sont générés côté client lorsque cela est possible
- Les codes de sauvegarde sont hachés avant stockage
- Chaque code de sauvegarde est à usage unique

**Actions administratives :**

- Les réinitialisations de 2FA sont enregistrées dans le journal d'activité
- Les réinitialisations de mot de passe créent une trace d'audit
- Les actions administratives nécessitent des permissions de rôle appropriées
- Les adresses IP sont enregistrées pour les événements de sécurité

# Documentation connexe

- `2fa` - Référence API 2FA détaillée (axée sur l'API)
- `rbac` - Contrôle d'accès basé sur les rôles et autorisations
- `administration_configuration` - Configuration des e-mails Mailjet pour la réinitialisation de mot de passe
- `integrations_mailjet` - Configuration des modèles d'e-mails
- `customer_care` - Portail de Self-Care pour les clients

# Créer un client

0fT52ZvoZBE

# Clients, Contacts, Sites & Services

Nous avons un modèle simple de **Client** sous lequel un client peut avoir plusieurs **Contacts** et plusieurs **Sites**, **Services**, etc.

Un **Client** est une entreprise ou un individu qui a une relation avec nous, à qui nous envoyons une facture.

Un **Contact** est une personne qui travaille avec le client, pour un individu, il s'agit probablement du client lui-même, une seule personne, mais nous pourrions avoir des membres de la famille ou d'autres contacts, et chaque contact a un type, par exemple un contact de facturation, un contact technique, etc., ce qui influence la manière dont nous gérons le contact.

Un **Site** est un emplacement physique où nous livrons des services, cela peut être une maison, un bureau ou un autre emplacement. Cela nous permet d'avoir plusieurs sites pour un seul client, par exemple, un client avec plusieurs bureaux, et de savoir quels services sont associés à quel site.

Un **Service** est quelque chose pour lequel nous facturons un client, cela peut être un service Internet à domicile, un service mobile, ou même des services abstraits comme la location d'un sous-réseau ou la fourniture d'électricité mesurée à un rack. Chaque service est lié à un client et à un site, et peut avoir plusieurs frais associés.

Les clients ont également un Journal d'Activité, qui est un enregistrement de tous les changements effectués, Tags, Attributs pour stocker des métadonnées personnalisées, Articles d'Inventaire et des informations financières comme Transactions, Factures & Méthodes de Paiement.

Une fois que nous avons créé un client, nous pouvons ensuite ajouter un service à ce client, qui est la chose pour laquelle nous les facturons.

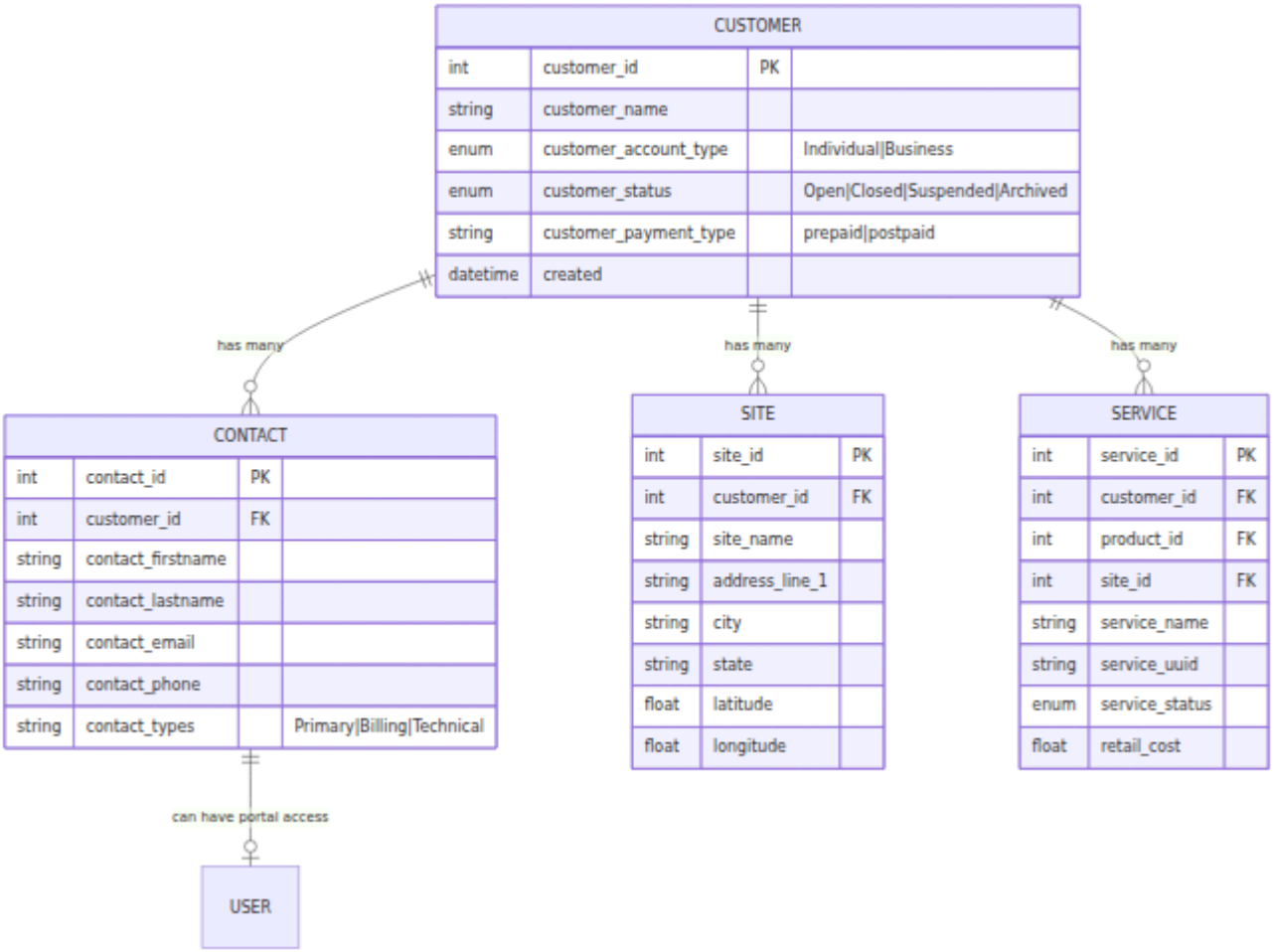
Pour des informations sur la création d'un client, voir Créer un Client.

# Vue d'ensemble du Modèle de Données

OmniCRM utilise un modèle de données relationnel organisé autour des clients et de leurs services. Le modèle est divisé en sections ciblées ci-dessous.

## Relations de Base du Client

Le client est l'entité centrale, avec des contacts, des sites et des services associés.



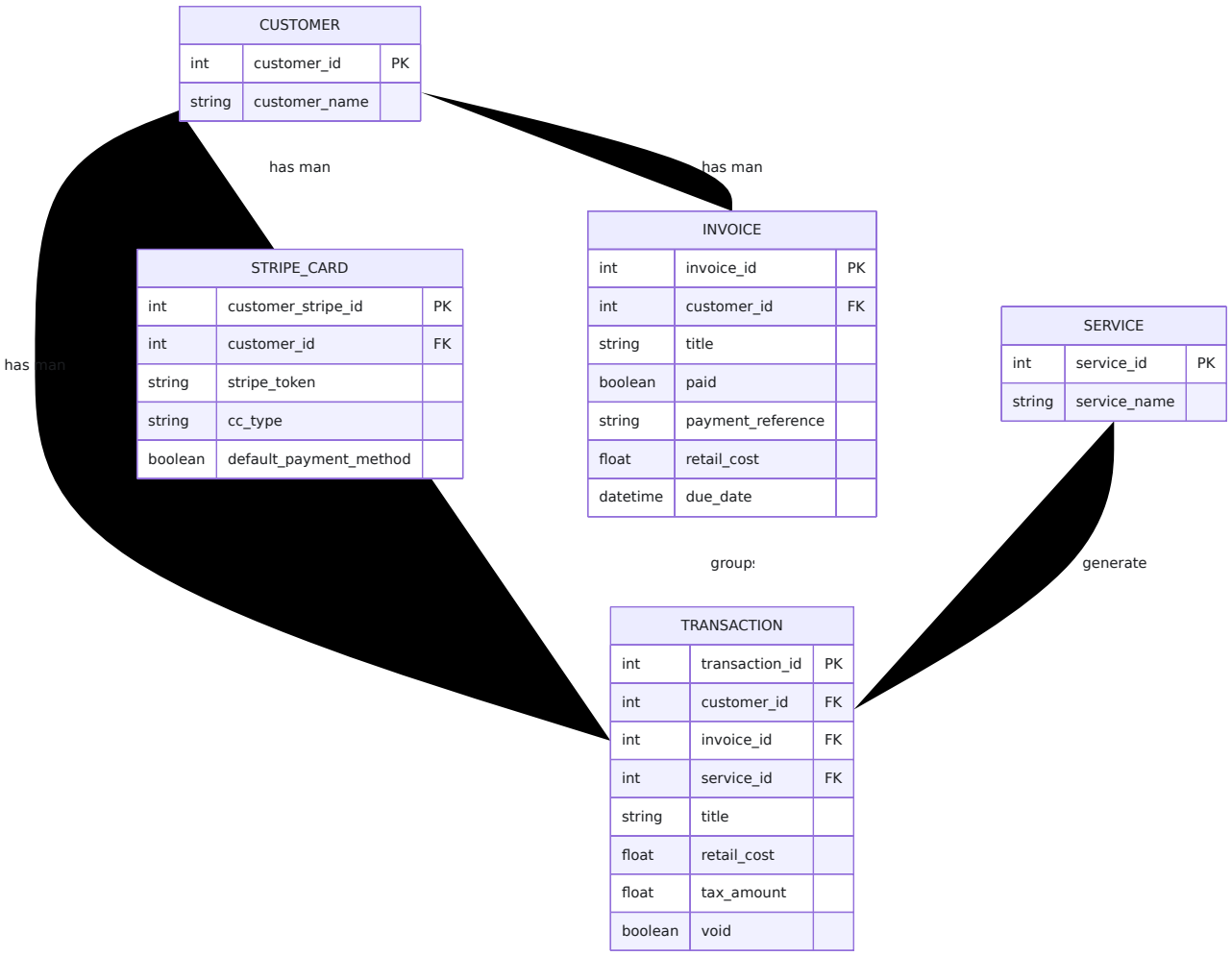

**Points Clés :**

- Un client peut avoir plusieurs contacts (facturation, technique, etc.)
- Un client peut avoir plusieurs sites (bureaux, emplacements)
- Les services sont livrés aux sites

- Les contacts peuvent avoir un accès au portail via des comptes utilisateurs liés

## Données de Facturation & Financières

Les transactions et les factures suivent toute l'activité financière.

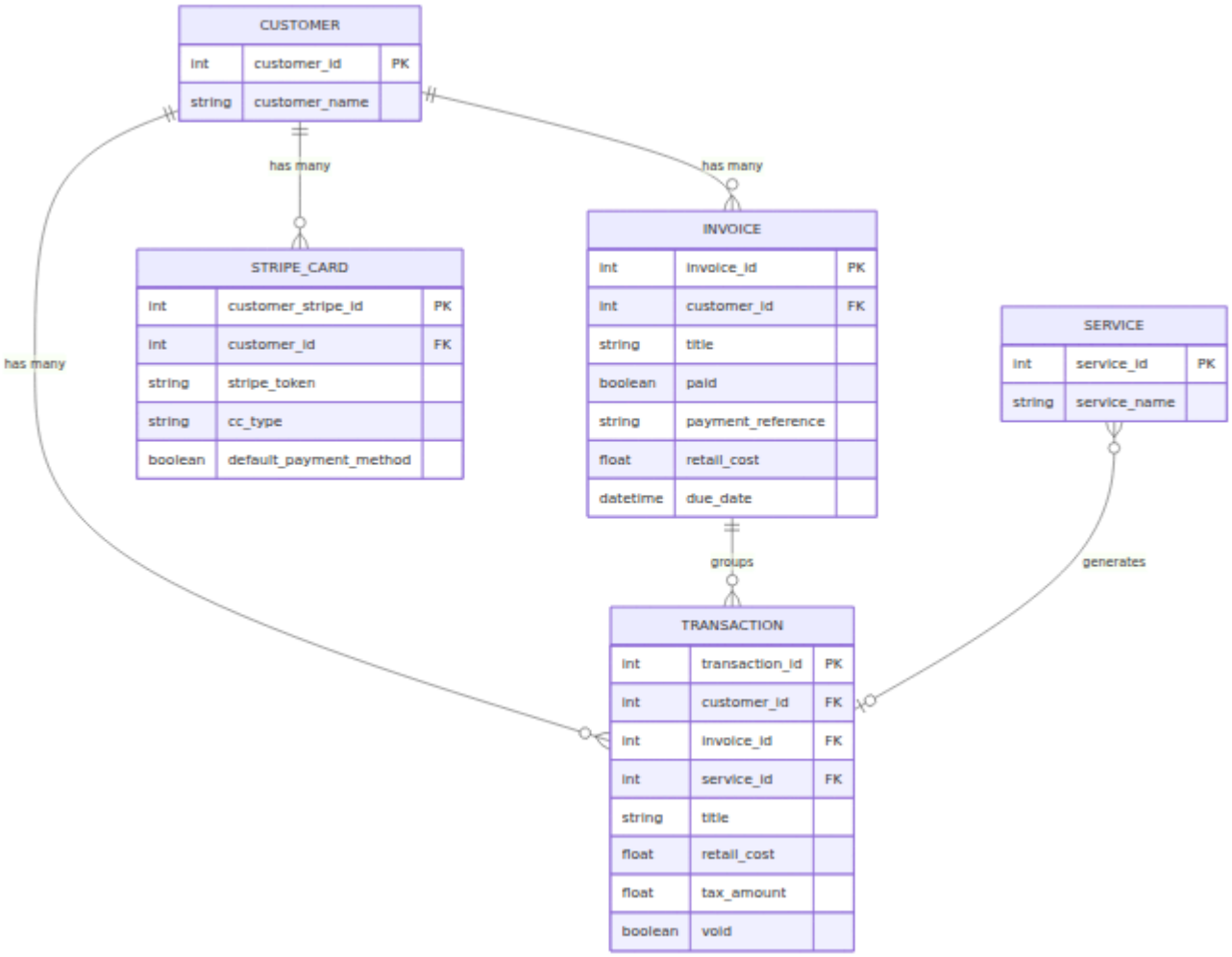



**Points Clés :**

- Les transactions peuvent être autonomes ou regroupées en factures
- Les services génèrent automatiquement des transactions mensuelles
- Les cartes Stripe sont tokenisées et stockées en toute sécurité
- Les factures regroupent plusieurs transactions pour la facturation

# Produits & Provisionnement

Les produits définissent les offres de services ; le provisionnement crée des services réels.

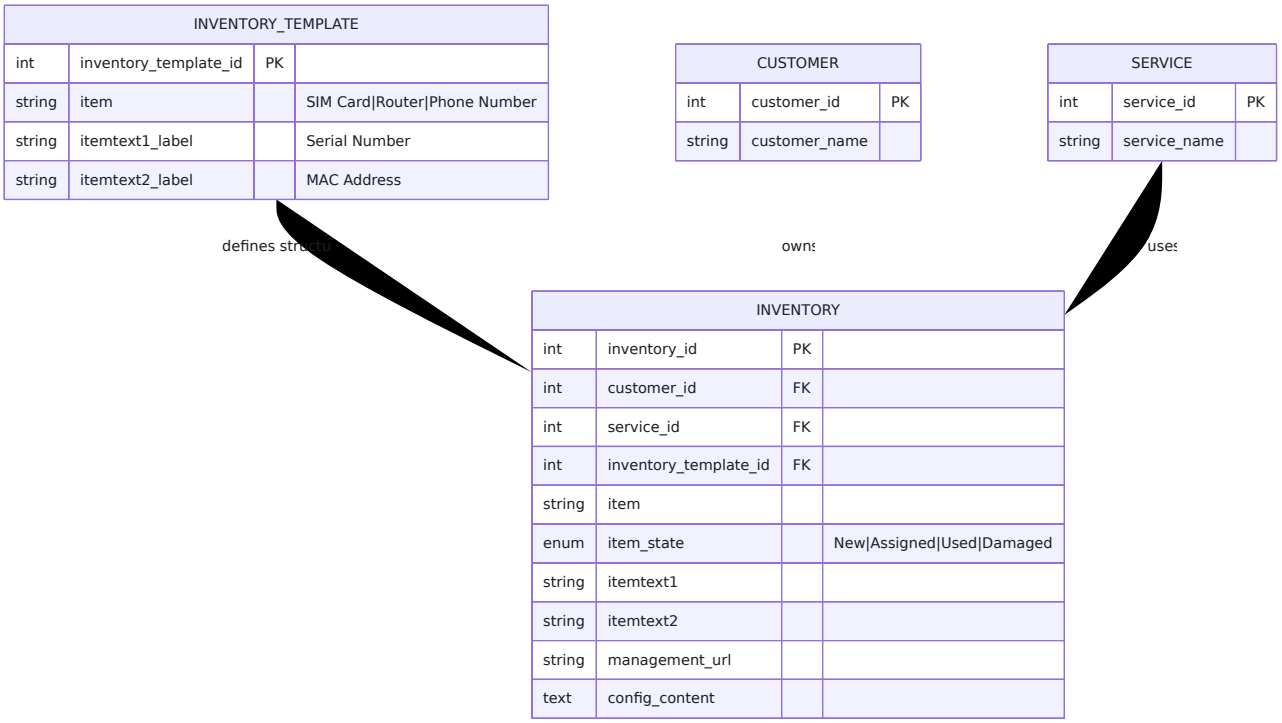



**Points Clés :**

- Les produits sont des modèles ; les services sont des instances actives
- Le provisionnement crée ou modifie des services via des playbooks Ansible
- Chaque tâche de provisionnement a plusieurs événements pour suivre les progrès
- Un provisionnement de produit peut créer plusieurs services (bundles)

# Système d'Inventaire

Suivez les actifs physiques et virtuels assignés aux clients.

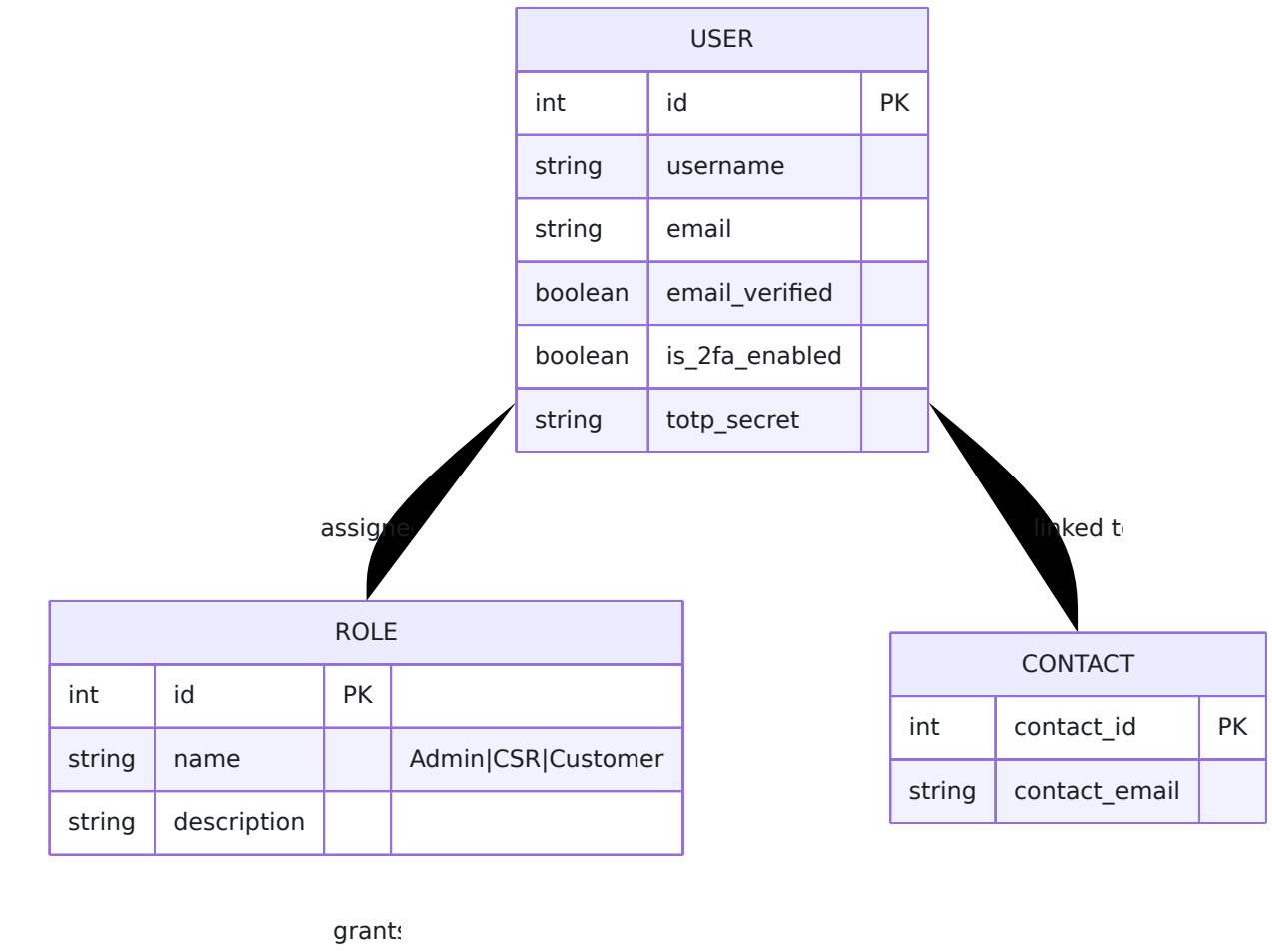



**Points Clés :**

- Les modèles d'inventaire définissent la structure (champs) pour chaque type d'article
- Les champs flexibles itemtext1-20 s'adaptent à différents types d'inventaire
- Les articles peuvent être assignés à des clients et liés à des services
- Stocke les configurations d'équipement et les identifiants de gestion

# Utilisateur & Sécurité

Comptes utilisateurs avec contrôle d'accès basé sur les rôles.




**Points Clés :**

- Les utilisateurs peuvent avoir plusieurs rôles ; les rôles ont plusieurs permissions
- L'authentification à deux facteurs (2FA) est optionnelle par utilisateur
- Les utilisateurs du personnel sont autonomes ; les utilisateurs clients sont liés aux contacts
- Des permissions granulaires contrôlent l'accès aux ressources

# Liste des Clients

La liste des clients fournit un tableau filtrable et consultable de tous les clients dans le système.

**Fonctionnalités :**

- **Recherche** - Filtrer les clients par nom ou ID
- **Actions Groupées** - Sélectionner plusieurs clients pour des opérations en lot
- **Pagination** - Naviguer à travers de grandes listes de clients
- **Actions Rapides** - Voir ou supprimer des clients directement depuis la liste

# Client - Détails

L'objet client lui-même ne contient pas beaucoup d'informations, il s'agit juste d'un nom et d'une référence aux contacts et aux sites.

Votre déploiement spécifique peut inclure des champs supplémentaires ou des personnalisations, mais l'objet client de base est très simple.

Sur la page d'aperçu, il y a également un graphique montrant le Revenu Moyen Par Utilisateur (`ARPU`) pour le client, qui est le revenu total divisé par le nombre

de services, et une comparaison de ce client par rapport à l'ARPU moyen de tous les clients dans le système.

Les options de Statut du Client sont adaptées en fonction des besoins spécifiques de votre entreprise, mais incluent généralement des options comme Actif, Inactif, En Attente, etc., avec différentes règles dans chacune qui contrôlent le comportement du client dans le système dans cet état.

La suppression d'un client ne peut être effectuée que si le client n'a pas de services actifs, de factures impayées ou de transactions non facturées. Si le client a l'un de ces éléments, vous devrez fermer les services actifs et vous assurer que les paiements sont effectués avant de pouvoir supprimer le client, ce qui à son tour archivera le client et toutes les données associées, qui pourront être restaurées ultérieurement si nécessaire.

# Site - Détails

Les sites sont des emplacements physiques où les services sont livrés, et peuvent avoir plusieurs services associés.

Ils sont principalement utilisés pour les clients professionnels, où un seul client pourrait avoir plusieurs sites, par exemple, un client avec plusieurs bureaux.

Avoir plusieurs sites nous permet de suivre quels services sont associés à quel site, par exemple, si un client commande un nouveau service pour un nouveau bureau, nous devons nous assurer que nous livrons le service correct au bon emplacement. Cela nous permet de suivre les services par site et de les facturer séparément si nécessaire.

# Intégration Google Maps et Géocodage

Chaque site est intégré à **Google Maps** pour garantir des données d'adresse précises et une géolocalisation. L'interface utilisateur géocode automatiquement les adresses et génère des données de localisation.

**Comment fonctionne le Géocodage d'Adresse :**

Lors de l'ajout ou de l'édition d'un site, l'interface propose deux méthodes pour définir l'emplacement :

1. **Recherche d'Adresse (Recommandée)**
    - Utilisez la barre de recherche en haut du formulaire
    - Tapez une adresse et Google Maps Autocomplete suggère des correspondances
    - Sélectionnez la bonne adresse dans le menu déroulant
    - Le système remplit automatiquement :
        - **Nom du Site** - Nom du lieu provenant de Google Maps
        - **Adresse Ligne 1** - Numéro et nom de la rue
        - **Adresse Ligne 2** - Sous-lieu (numéro d'unité/suite)
        - **Ville** - Localité
        - **État/Région** - Zone administrative
        - **Code Postal** - Code postal
        - **Pays** - Nom du pays
        - **Latitude & Longitude** - Coordonnées précises
        - **Plus Code** - Code d'Emplacement Ouvert de 11 caractères (ex. `8C3MFJV8+2F`)
        - **Google Maps Place ID** - Identifiant unique du lieu
2. **Marqueur de Carte Déplaçable (Manuel)**
    - Faites glisser le pin sur la carte jusqu'à l'emplacement exact

- Le système effectue un **géocodage inverse** pour obtenir l'adresse à partir des coordonnées
- Les mêmes champs se remplissent automatiquement en fonction de l'emplacement du pin
- Utile pour les zones rurales ou lorsque l'adresse est imprécise

**Génération de Plus Code :**

Les Plus Codes sont générés automatiquement à partir de la latitude/longitude en utilisant la bibliothèque Open Location Code. Un Plus Code est un code court (11 caractères) qui représente un emplacement précis n'importe où dans le monde.

**Exemple :**

> Champs auto-remplis : ✓ Nom du Site : "123 Main Street" ✓ Adresse Ligne 1 : "123 Main Street" ✓ Ville : "Londres" ✓ État : "Grand Londres" ✓ Pays : "Royaume-Uni" ✓ Code Postal : "SW1A 1AA" ✓ Latitude : 51.5074 ✓ Longitude : -0.1278 ✓ Plus Code : "9C3XGPHC+3Q" ✓ Google Place ID : "ChIJdd4hrwug2EcRmSrV3Vo6llI"

**Exigences de Validation :**

Avant de sauvegarder un site, le système valide :

- **Latitude & Longitude** doivent être définies (soit via la recherche soit en déplaçant le marqueur)
- **Le Pays** doit être renseigné (par défaut à `VITE_DEFAULT_COUNTRY` s'il n'est pas défini)
- **Le Plus Code** doit comporter 12 caractères (11 + 1 pour le remplissage)

Si la validation échoue, vous verrez une erreur :

**Retour Visuel :**

L'interface montre un retour en temps réel :

ou

# Comment les Données de Localisation sont Utilisées

Les données de localisation géocodées (latitude, longitude, Plus Code) sont utilisées dans tout OmniCRM pour :

### 1. Livraison et Installation de Services

- **Techniciens de Terrain** - Accéder au Plus Code pour naviguer vers l'emplacement exact du site
- **Planification d'Installation** - Assigner des techniciens en fonction de la proximité géographique
- **Déploiement d'Équipement** - S'assurer que le bon équipement est livré au bon site

### 2. Notifications de Panne

- **Alertes Géofencées** - Si une panne de réseau dans une zone spécifique, interroger les sites par rayon lat/long
- **Communications Ciblées** - Envoyer des notifications de panne uniquement aux clients concernés via Mailjet
- **Pages de Statut** - Afficher une carte des pannes avec les sites affectés

**Exemple :**

> Requête : SELECT * FROM Customer_Site
> WHERE distance(latitude, longitude, 51.5074, -0.1278) < 5
>
> Résultat : 47 sites affectés Action : Envoyer une notification de panne à 47 clients

### 3. Reporting et Analytique

- **Revenu Géographique** - Revenu par ville, état, région
- **Cartes de Densité de Service** - Carte thermique des emplacements de service
- **Planification d'Expansion** - Identifier les zones sous-desservies

### 4. Clients Professionnels Multi-Sites

- **Gestion des Sites** - Suivre quels services sont à quels emplacements
- **Facturation Séparée** - Facturer par site si nécessaire
- **Assignation de Services** - Lier des services à des sites spécifiques lors du provisionnement

## Sites Ruraux et Éloignés

Pour les clients dans les zones rurales où les adresses peuvent ne pas exister ou être inexactes :

1. **Utiliser le Glissement de Carte**
   - Zoomer dans la zone approximative
   - Faire glisser le pin jusqu'à la propriété/bâtiment exact
   - Le système génère un Plus Code pour cet emplacement précis
2. **Partage de Plus Code**
   - Partager le Plus Code avec le client (ex. `8C3MFJV8+2F`)
   - Le client peut entrer cela dans Google Maps pour voir l'emplacement exact
   - Les techniciens de terrain utilisent le Plus Code pour la navigation
3. **Notes d'Adresse**
   - Utiliser le champ "Note d'Adresse" pour des directions supplémentaires
   - Exemple : "Tourner à gauche au barn rouge, 500m après la grille à bétail"
   - Notes visibles pour les équipes d'installation

**ASTUCE**

Vous pouvez faire glisser le pin sur la carte jusqu'à l'emplacement correct si l'adresse n'est pas précise. Le système géocodera l'emplacement et remplira automatiquement tous les champs d'adresse.

# Contact - Détails

Les Contacts sont des personnes associées au client. Ils peuvent être des contacts de facturation, des contacts techniques, ou d'autres types, et chaque contact a un type qui influence la manière dont nous gérons le contact.

Nous pouvons avoir plusieurs contacts pour un seul client, par exemple, un client avec plusieurs contacts de facturation, ou un client avec plusieurs contacts techniques.

Un bon exemple serait une entreprise avec un fournisseur de services gérés, qui gère le côté technique des choses, et un contact de facturation séparé qui gère le côté financier, ou une famille où chaque membre a son propre contact mais tous ne sont pas autorisés à apporter des modifications.

De même, nous ne voudrions peut-être envoyer des notifications de panne qu'au contact technique, ou envoyer des factures uniquement au contact de facturation, et le type de contact nous permet de contrôler cela.

La logique exacte de la manière dont les types de contact sont utilisés dépend de votre entreprise, mais l'idée de base est que chaque contact a un type qui influence la manière dont nous les gérons, et que chaque personne associée au client est un contact.

Les contacts sont synchronisés avec l'intégration Mailjet, ce qui nous permet d'envoyer des campagnes email ciblées en fonction du type de contact, de la localisation, des dépenses mensuelles ou des services achetés, et de gérer tous les modèles d'email utilisés pour les communications transactionnelles.

# Navigation

OmniCRM est conçu dès le départ pour être réactif et intuitif.

Ce guide vous aidera à naviguer dans le système et à trouver les informations dont vous avez besoin.

Une fois connecté, la barre de navigation sur le côté gauche de l'écran vous montrera les sections principales du système, ainsi que les sous-sections de chaque section.

En haut à droite de l'écran, vous verrez le menu utilisateur, qui vous permet de vous déconnecter, de changer votre mot de passe ou d'accéder à vos

paramètres utilisateur.

Vous avez des liens vers toutes les applications web les plus couramment utilisées de votre organisation dans la barre WebApp (cela peut être adapté aux besoins de votre organisation).

Tous les alertes que vous avez en attente sont visibles en haut à droite de l'écran, et vous pouvez cliquer sur l'alerte pour voir plus d'informations.

Nous pouvons changer la langue du système en cliquant sur le menu déroulant de langue en haut à droite de l'écran.

Si vous êtes un prince des ténèbres, vous pouvez passer en mode sombre en cliquant sur l'icône de lune en haut à droite de l'écran.

# Paiements, Factures et Transactions

## Transactions Client

Tout ce qui coûte de l'argent dans le système est enregistré comme une transaction sous le client.

Chaque transaction a un montant monétaire pour le coût de gros et le coût de détail, ainsi qu'une description de ce à quoi sert la transaction.

Les transactions peuvent être générées automatiquement par le système, par exemple, lorsqu'un service est provisionné, une transaction est créée pour le coût d'installation, et lorsqu'un service est facturé, une transaction est créée pour le coût de détail.

Les transactions peuvent également être créées manuellement, par exemple, si un client reçoit un crédit, une transaction est créée pour le montant du crédit, ou si des frais d'installation sont facturés, une transaction est créée pour les frais d'installation.

Les transactions sont regroupées pour former Factures, qui sont envoyées au client pour paiement.

# Accéder aux Transactions

Les transactions peuvent être consultées au niveau du système ou par client :

**Vue par Client :**

1. Naviguez vers **Clients → [Sélectionner Client]**
2. Cliquez sur l'onglet **Facturation**
3. Consultez la liste des transactions dans la première carte

**Vue Système :**

1. Naviguez vers **Facturation → Transactions** (dans le menu principal)
2. Consultez toutes les transactions de tous les clients

# Widgets de Statistiques de Transactions

En haut de la page des transactions, quatre cartes statistiques affichent des résumés financiers :

**Descriptions des Widgets :**

- **Total des Transactions** - Somme de tous les coûts de détail des transactions (toute période)
- **Total des Transactions Non Facturées** - Somme des transactions non encore incluses dans une facture
- **Total des Transactions Ce Mois** - Somme des transactions créées ce mois calendaire
- **Total des Transactions Le Mois Dernier** - Somme des transactions créées le mois dernier

**Formatage des Valeurs :**

- Valeurs supérieures à 1 000 : Afficher avec le suffixe "k" (par exemple, $1.5k)
- Valeurs supérieures à 1 000 000 : Afficher avec le suffixe "M" (par exemple, $2.3M)
- Valeurs supérieures à 1 000 000 000 : Afficher avec le suffixe "B" (par exemple, $1.1B)

# Liste des Transactions

Le tableau des transactions affiche toutes les transactions avec les colonnes suivantes :

**Descriptions des Colonnes :**

- **ID** - ID de transaction unique
- **Date** - Date de création de la transaction
- **Titre** - Nom court de la transaction
- **Description** - Description détaillée de ce à quoi sert la transaction
- **Montant** - Coût de détail (positif pour les frais, négatif pour les crédits)
- **Facture** - ID de la facture si la transaction a été facturée (lien cliquable)
- **Statut** - Coche si facturée, tiret si pas encore facturée

**Actions par Ligne :**

Chaque ligne a un menu d'actions ( ⋮ ) avec les options :

- **Voir Détails** - Ouvre la modal de détail de la transaction
- **Télécharger PDF de la Facture** - Télécharger PDF (uniquement si facturée)
- **Annuler la Transaction** - Marquer la transaction comme annulée (uniquement si pas facturée)

# Types de Transactions

Les transactions se divisent en deux catégories principales :

## Transactions Débitées (Frais)

Montants positifs qui augmentent le solde dû par le client :

- **Frais d'Installation de Service** - Frais uniques lors de la provision du service
- **Frais de Service Mensuels** - Frais récurrents pour les services
- **Frais d'Installation** - Frais pour les visites de techniciens sur le terrain
- **Frais d'Équipement** - Frais pour modems, routeurs, cartes SIM
- **Frais de Retard de Paiement** - Pénalités pour factures en retard
- **Frais Manuels** - Frais personnalisés ajoutés par le personnel

## Transactions Crédit (Paiements/Remboursements)

Montants négatifs qui diminuent le solde dû par le client :

- **Paiements en Espèces** - Client a payé en espèces
- **Paiements par Carte** - Client a payé par carte de crédit/débit
- **Paiements par Virement Bancaire** - Client a payé par virement bancaire
- **Crédits de Compte** - Crédits de bonne volonté, compensation
- **Remboursements** - Argent retourné au client
- **Remises** - Remises promotionnelles ou de fidélité

# Ajouter une Transaction Manuellement

Cliquez sur **"+ Ajouter une Transaction"** pour ouvrir la modal d'ajout de transaction.

**Transaction Débité (Frais) :**

**Transaction Crédit (Paiement/Remboursement) :**

**Descriptions des Champs :**

- **Type de Transaction** - Sélectionnez Débit (frais) ou Crédit (paiement/remboursement)
- **Type de Crédit** - Si Crédit sélectionné, choisissez le mode de paiement (Espèces, Carte, Virement Bancaire)
- **Titre** - Nom court pour la transaction (obligatoire)
- **Description** - Explication détaillée (optionnelle)
- **Coût de Détail** - Montant que le client paie (obligatoire, nombre positif)
- **Coût de Gros** - Votre coût (optionnel, pour le suivi de marge)
- **Pourcentage de Taxe** - Taux de taxe appliqué à cette transaction (optionnel, par défaut à la taxe produit ou 0%)
- **Service** - Lier la transaction à un service spécifique (optionnel)
- **Site** - Lier la transaction à un site spécifique (optionnel)
- **Date de la Transaction** - Date de la transaction (par défaut à aujourd'hui)

**Validation :**

- Le titre et le coût de détail sont obligatoires
- Le coût de détail doit être un nombre positif
- Si le type de Crédit est sélectionné, un type de crédit doit être choisi

**Ce Qui Se Passe :**

1. Transaction créée dans la base de données
2. Apparait dans la liste des transactions du client
3. Inclus dans le compte des "Transactions Non Facturées"
4. Disponible pour inclusion dans la prochaine génération de factures
5. Entrée de journal d'activité créée

# Recherche et Filtrage des Transactions

## Recherche

Utilisez la barre de recherche pour trouver des transactions. Recherche dans :

- ID de transaction
- Titre
- Description
- ID de facture

## Filtres

Appliquez des filtres pour affiner la liste des transactions :

**Filtres Disponibles :**

- **Statut d'Annulation** - Tous, Annulé, Non Annulé
- **Statut de Facture** - Tous, Facturé, Non Facturé

**Actions de Filtre :**

- **Appliquer les Filtres** - Appliquer les filtres sélectionnés à la liste
- **Réinitialiser les Filtres** - Effacer tous les filtres et afficher toutes les transactions

## Tri

Cliquez sur n'importe quel en-tête de colonne pour trier :

- **ID** - Trier par ID de transaction (le plus récent/le plus ancien)
- **Date** - Trier par date de transaction
- **Titre** - Trier par ordre alphabétique
- **Montant** - Trier par coût de détail (le plus élevé/le plus bas)
- **Facture** - Trier par ID de facture

Cliquez à nouveau pour inverser la direction du tri (ascendant ↔ descendant).

# Annuler des Transactions

Les transactions ajoutées par erreur peuvent être **annulées** (marquées comme supprimées).

**Exigences :**

- La transaction ne doit PAS être facturée
- Une fois facturées, les transactions ne peuvent pas être annulées (doivent être remboursées à la place)

**Comment Annuler :**

1. Localisez la transaction dans la liste
2. Cliquez sur le menu d'actions ( ⋮ )
3. Sélectionnez "**Annuler la Transaction**"
4. Confirmez dans la modal

**Ce Qui Se Passe :**

- La transaction est marquée comme `void = true`
- N'apparaît plus dans la liste des transactions par défaut
- Exclue de la génération de factures
- Peut être vue en filtrant pour les transactions "Annulées"
- Déduite du total des "Transactions Non Facturées"

**Transactions liées au portefeuille** (`wallet_ledger_id` défini) :

- Le solde du portefeuille est inversé avant l'annulation (débit pour annuler un crédit, crédit pour annuler un débit)

- **Carte de recharge de fonds** (`payment_type=wallet_topup_card`) : portefeuille débité et carte remboursée
- **Recharge en espèces / paiement de facture de portefeuille / autres lignes de portefeuille :** portefeuille ajusté uniquement ; **pas de remboursement de carte**
- **Portefeuille insuffisant :** l'annulation échoue si les fonds rechargés ont déjà été dépensés (la transaction reste active)

**Frais de service / de provisionnement** de capture hybride n'ont généralement **pas** de `wallet_ledger_id` sur la ligne de facturation. L'annulation de ces lignes ne définit que `void=true` aujourd'hui ; un crédit complet du portefeuille pour les annulations de service est prévu sur la branche des documents financiers.

**API :** `DELETE /crm/transaction/transaction_id/{transaction_id}`. Voir guide_système_de_paiement.md : Annuler la Transaction.

**Remarque :** Annuler n'est PAS la même chose que rembourser. Annuler signifie "cette transaction n'aurait jamais dû exister." Rembourser signifie "inverser une transaction valide."

# Taxe sur les Transactions

Les transactions peuvent inclure des taxes, qui sont automatiquement calculées en fonction de la configuration fiscale du produit ou spécifiées manuellement par transaction.

**Comportement de la Taxe :**

- **Transactions Débitées (Frais)** - La taxe est appliquée aux frais en fonction de :
  - **Pourcentage de Taxe du Produit** - Si la transaction est liée à un produit, le pourcentage de taxe du produit est automatiquement appliqué
  - **Surcharge Manuelle** - Le personnel peut remplacer le pourcentage de taxe lors de la création d'une transaction
  - **Montant de la Taxe** - Calculé comme : `retail_cost × (tax_percentage / 100)`
  - **Format d'Affichage** - Affiché comme : `$10.00 (10%)` dans les listes de transactions
- **Transactions Crédit (Paiements/Remboursements)** - Aucune taxe n'est appliquée aux crédits
  - Le champ de pourcentage de taxe est masqué pour les transactions de crédit
  - La taxe est automatiquement fixée à 0 % pour tous les paiements et remboursements
  - Les crédits réduisent le solde impayé du client sans implications fiscales

**Exemple de Calcul de Taxe :**

- Produit : Forfait Mobile avec 10 % de taxe, $50.00 coût de détail
- Calcul de Taxe Automatique : $50.00 × 0.10 = $5.00
- Affichage : $5.00 (10 %)

**Taxe Nulle (NIL/Exempt) :**

- Les produits peuvent être exonérés de taxe en fixant le pourcentage de taxe à 0
- La taxe par défaut est de 0 % si non spécifiée
- Les transactions exonérées de taxe affichent "-" dans la colonne Taxe

# Vue des Détails de la Transaction

Cliquez sur une transaction pour voir les détails complets :

# Transactions Facturées vs Non Facturées

**Transactions Non Facturées :**

- Pas encore incluses dans une facture
- Disponibles pour la prochaine génération de factures

- Peuvent être annulées
- Comptent pour le total des "Transactions Non Facturées"
- Le statut affiche un tiret (-)

**Transactions Facturées :**

- Incluses dans une facture
- Ne peuvent pas être annulées (doivent être remboursées si nécessaire)
- ID de facture cliquable (lien vers les détails de la facture)
- Le statut affiche une coche (✓)
- Ne peut pas être modifié

**Génération de Facture :**

Lorsque vous générez une facture pour un client :

1. Le système trouve toutes les transactions non facturées pour ce client
2. Optionnellement filtrer par plage de dates
3. Transactions incluses dans la nouvelle facture
4. Champ `invoice_id` de la transaction peuplé
5. Transaction maintenant marquée comme "facturée"

Voir `payments_invoices` pour les détails de génération de factures.

# Flux de Travail Courants

## Flux de Travail 1 : Crédit Manuel pour Panne de Service

1. Le client appelle : "Le service était hors ligne pendant 2 jours"
2. Le personnel décide de créditer 10 £
3. Naviguez vers l'onglet **Facturation** du client
4. Cliquez sur **"+ Ajouter une Transaction"**
5. Sélectionnez le type de transaction **Crédit**
6. Sélectionnez le type de crédit **Paiement en Espèces**

7. Entrez le titre : "Crédit pour Panne de Service"
8. Entrez la description : "Compensation pour la panne de 2 jours 8-9 Jan"
9. Entrez le coût de détail : 10.00
10. Sélectionnez le service affecté dans le menu déroulant
11. Cliquez sur **"Ajouter une Transaction"**
12. La transaction apparaît avec un montant de -10,00 £
13. Sera incluse dans la prochaine facture en tant que crédit

# Flux de Travail 2 : Frais d'Installation Manuel

1. Le technicien de terrain installe le service
2. Le personnel doit facturer des frais d'installation de 75 £
3. Naviguez vers l'onglet **Facturation** du client
4. Cliquez sur **"+ Ajouter une Transaction"**
5. Sélectionnez le type de transaction **Débit**
6. Entrez le titre : "Frais d'Installation"
7. Entrez la description : "Visite du technicien de terrain pour installation de fibre"
8. Entrez le coût de détail : 75.00
9. Entrez le coût de gros : 45.00 (optionnel, pour le suivi de marge)
10. Sélectionnez le service installé
11. Sélectionnez le site où installé
12. Cliquez sur **"Ajouter une Transaction"**
13. La transaction apparaît dans la liste non facturée
14. Sera incluse dans la prochaine facture

# Flux de Travail 3 : Annulation de Transaction Dupliquée

1. Le personnel remarque une transaction dupliquée
2. Vérifiez que la transaction n'est PAS encore facturée
3. Cliquez sur le menu d'actions ( ⋮ ) sur la transaction dupliquée
4. Sélectionnez **"Annuler la Transaction"**
5. Confirmez dans la modal

6. La transaction est supprimée de la liste
7. Le total non facturé diminue en conséquence

## Flux de Travail 4 : Trouver des Transactions pour Facture

1. Besoin de générer une facture mensuelle
2. Cliquez sur **Filtre de Facture : "Non Facturé"**
3. Cliquez sur **Appliquer les Filtres**
4. Consultez toutes les transactions non facturées
5. Notez le montant total des widgets
6. Naviguez pour générer la facture
7. Sélectionnez la plage de dates (par exemple, 1-31 Jan)
8. Transactions dans la plage incluses dans la facture

# Dépannage

**Impossible d'annuler la transaction**

- **Cause :** Transaction déjà facturée
- **Solution :** La transaction fait partie de l'historique des factures. Si un remboursement est nécessaire, créez une transaction de Crédit à la place.
- **Cause :** Échec de l'annulation de la recharge de portefeuille (solde de portefeuille insuffisant)
- **Solution :** Le client a déjà dépensé les fonds rechargés. Utilisez un débit/ajustement manuel de portefeuille ou des flux de renversement de documents financiers ; vous ne pouvez pas annuler la ligne de recharge tant que le portefeuille ne couvre pas le montant.
- **Cause :** Échec du remboursement par carte (recharge de fonds uniquement)

- **Solution :** Le renversement du portefeuille est annulé automatiquement. Résolvez le problème du fournisseur de paiement et réessayez d'annuler.

**Transactions dupliquées apparaissant**

- **Cause :** Service facturé plusieurs fois ou erreur de provisionnement
- **Solution :** Annulez la ou les transactions dupliquées si elles ne sont pas facturées. Si facturées, émettez un crédit.

**Transaction n'apparaissant pas dans la liste**

- **Cause :** Filtres appliqués ou transaction annulée
- **Solution :** Cliquez sur "Réinitialiser les Filtres" pour afficher toutes les transactions. Pour voir les transactions annulées, filtrez par "Annulé : Annulé".

**Le total non facturé ne correspond pas à ce qui est attendu**

- **Cause :** Certaines transactions déjà facturées, ou transactions annulées exclues
- **Solution :** Appliquez le filtre "Facture : Non Facturée" pour voir uniquement les non facturées. Vérifiez les transactions annulées séparément.

**Impossible d'ajouter une transaction (champ client désactivé)**

- **Cause :** Consultation de la page des transactions spécifiques au client
- **Solution :** Le client est pré-sélectionné. Si vous devez ajouter une transaction pour un autre client, allez à la page des Transactions au niveau du système.

# Documentation Connexe

- `payments_invoices` - Génération et gestion des factures
- `payments_process` - Traitement des paiements contre les factures
- `basics_payment` - Vue d'ensemble des méthodes de paiement
- `csa_activity_log` - Consultation de l'historique des transactions dans le journal d'activité

# Factures Client

Transactions sont regroupées pour former une facture, qui est envoyée au client pour paiement.

Les factures ont une date de début et une date de fin, qui est la période que la facture couvre, et une date d'échéance, qui est la date à laquelle la facture est due pour paiement.

Les factures peuvent être générées automatiquement par le système, par exemple, lorsque un service est facturé, une facture est créée pour le coût de détail, ou elles peuvent être créées manuellement, par exemple, si un client demande une copie d'une facture, ou si un client est facturé pour un frais unique.

Les factures clients sont entièrement modélisées avec Mailjet et peuvent être personnalisées pour inclure le logo de l'entreprise, l'adresse et les détails de paiement, et peuvent être envoyées au client par e-mail, ou téléchargées en tant que PDF.

# Personnaliser les Modèles de Facture

OmniCRM utilise des modèles HTML avec Jinja2 pour générer des factures. Vous pouvez entièrement personnaliser le design de la facture, la marque, les couleurs et la mise en page.

## Emplacement du Modèle de Facture

Les modèles de factures sont stockés dans `OmniCRM-API/invoice_templates/`

**Modèles Par Défaut :**

- `sample_invoice_template.html` - Modèle de facture d'exemple
- `cifi_invoice_template.html` - Exemple de modèle alternatif

**Configuration :**

Le modèle de facture actif est spécifié dans `OmniCRM-API/crm_config.yaml` :

```yaml
invoice:
  template_filename: 'sample_invoice_template.html'
```

# Variables de Modèle Disponibles

Les modèles de facture ont accès aux variables Jinja2 suivantes :

**Informations sur la Facture :**

- `{{ invoice_number }}` - ID de facture unique (par exemple, `INV-2025-001234`)
- `{{ date }}` - Date d'émission de la facture (format ISO : `2025-01-10T12:00:00`)
- `{{ due_date }}` - Date d'échéance de paiement (par exemple, `2025-02-10`)
- `{{ start_date }}` - Date de début de la période de facturation
- `{{ end_date }}` - Date de fin de la période de facturation
- `{{ total_amount }}` - Montant total de la facture avant taxe (numérique)
- `{{ total_tax }}` - Montant total de la taxe calculé à partir de toutes les transactions (numérique)

**Informations sur le Client :**

- `{{ client.name }}` - Nom complet du client ou nom de l'entreprise
- `{{ client.address.address_line_1 }}` - Ligne d'adresse 1
- `{{ client.address.address_line_2 }}` - Ligne d'adresse 2
- `{{ client.address.city }}` - Ville
- `{{ client.address.state }}` - État/province
- `{{ client.address.zip_code }}` - Code postal
- `{{ client.address.country }}` - Pays

**Articles de Ligne de Transaction :**

Boucle à travers les transactions en utilisant :

```
{% for sub_transaction in transactions %}
    <tr>
        <td>{{ sub_transaction.transaction_id }}</td>
        <td>{{ sub_transaction.created.split("T")[0] }}</td>
        <td>{{ sub_transaction.title }}</td>
        <td>{{ sub_transaction.description }}</td>
        <td>${{ "%.2f"|format(sub_transaction.retail_cost) }}</td>
    </tr>
{% endfor %}
```

**Champs de Transaction :**

- `sub_transaction.transaction_id` - ID de la transaction
- `sub_transaction.created` - Date/heure de la transaction
- `sub_transaction.title` - Titre de la transaction
- `sub_transaction.description` - Description détaillée
- `sub_transaction.retail_cost` - Montant de l'article
- `sub_transaction.tax_percentage` - Pourcentage de taxe appliqué (par exemple, 10 pour 10 %)
- `sub_transaction.tax_amount` - Montant de la taxe calculé en dollars

**Affichage de la Taxe dans les Modèles :**

```
<td>
    {% if sub_transaction.tax_amount and
sub_transaction.tax_amount > 0 %}
        ${{ "%.2f"|format(sub_transaction.tax_amount) }} ({{
sub_transaction.tax_percentage }}%)
    {% else %}
        -
    {% endif %}
</td>
```

# Création d'un Modèle de Facture Personnalisé

### Étape 1 : Copier un Modèle Existant

```
cd OmniCRM-API/invoice_templates/
cp sample_invoice_template.html your_company_invoice_template.html
```

**Étape 2 : Personnaliser HTML/CSS**

Modifiez `your_company_invoice_template.html` pour correspondre à votre marque :

**Zones de Personnalisation Clés :**

1. **Logo et Marque de l'Entreprise**

```
<!-- Remplacez par l'URL de votre logo -->
![Votre Entreprise](https://yourcompany.com/logo.png)

<!-- Mettez à jour le nom de l'entreprise -->
<h1>Nom de Votre Entreprise</h1>
```

2. **Schéma de Couleurs**

```
<style>
    /* Couleur de marque principale */
    .navbar {
        background: linear-gradient(to bottom right, #your-
color-1, #your-color-2);
    }

    /* En-têtes de tableau */
    .table thead th {
        background-color: #your-brand-color !important;
        color: white !important;
    }

    /* Boutons et liens */
    .btn-primary {
        background-color: #your-brand-color;
    }
</style>
```

3. **Informations de Pied de Page de l'Entreprise**

```
<footer>
    <p>Nom de Votre Entreprise</p>
    <p>123 Business Street, Ville, Pays</p>
    <p>Téléphone : +1-555-123-4567 | Email :
billing@yourcompany.com</p>
    <p>ABN/ID Fiscal : 12345678900</p>
</footer>
```

4. **Instructions de Paiement**

```
<div class="payment-info">
    <h3>Méthodes de Paiement</h3>
    <p><strong>En Ligne :</strong> Payez sur
https://yourcompany.com/pay</p>
    <p><strong>Virement Bancaire :</strong></p>
    <ul>
        <li>Nom du Compte : Votre Entreprise Ltd</li>
        <li>BSB : 123-456</li>
        <li>Numéro de Compte : 987654321</li>
        <li>Référence : {{ invoice_number }}</li>
    </ul>
</div>
```

5. **Conditions Générales**

```
<div class="terms">
    <h4>Conditions de Paiement</h4>
    <p>Paiement dû dans les 30 jours suivant la date de la
facture.</p>
    <p>Frais de retard de paiement : 2 % par mois sur les
soldes en retard.</p>
    <p>Pour les questions de facturation :
billing@yourcompany.com</p>
</div>
```

**Étape 3 : Mettre à Jour la Configuration**

Modifiez `OmniCRM-API/crm_config.yaml` :

```
invoice:
  template_filename: 'your_company_invoice_template.html'
```

**Étape 4 : Redémarrer l'API**

```
cd OmniCRM-API
sudo systemctl restart omnicrm-api
```

**Étape 5 : Tester la Génération de Facture**

1. Naviguez vers un client avec des transactions
2. Générez une facture de test
3. Téléchargez le PDF pour vérifier le formatage
4. Envoyez la facture par e-mail pour tester la livraison par e-mail

# Personnalisation Avancée

**Contenu Conditionnel :**

Utilisez des conditionnels Jinja2 pour afficher/masquer du contenu :

```
{% if total_amount > 1000 %}
    <div class="high-value-notice">
        <p><strong>Remarque :</strong> Solde élevé - Plan de
paiement disponible sur demande.</p>
    </div>
{% endif %}

{% if client.address.country == "Australia" %}
    <p>GST Inclus : ${{ "%.2f"|format(total_amount * 0.10) }}</p>
{% endif %}
```

**Support Multilingue :**

Créez des modèles spécifiques à la langue :

```
invoice_template_en.html
invoice_template_es.html
invoice_template_fr.html
```

Configurez en fonction de la préférence linguistique du client.

**Calculs Personnalisés :**

```
<!-- Afficher le sous-total et la répartition de la taxe -->
<tr>
    <td colspan="4" class="text-right"><strong>Sous-total :
</strong></td>
    <td>${{ "%.2f"|format(total_amount) }}</td>
</tr>
<tr>
    <td colspan="4" class="text-right"><strong>Taxe :</strong>
</td>
    <td>${{ "%.2f"|format(total_tax) }}</td>
</tr>
<tr>
    <td colspan="4" class="text-right"><strong>Total :</strong>
</td>
    <td>${{ "%.2f"|format(total_amount + total_tax) }}</td>
</tr>
```

**Remarque :** La variable `total_tax` est calculée automatiquement en sommant le `tax_amount` de toutes les transactions de la facture. La taxe de chaque transaction est calculée en fonction de son champ `tax_percentage`, qui par défaut est celui du produit ou 0 % si non spécifié.

**Code QR pour le Paiement :**

Générez un code QR pour le paiement mobile :

```
<div class="qr-payment">
    ![Scanner pour Payer](https://api.qrserver.com/v1/create-qr-
code/?size=150x150&data={{ payment_url }})
    <p>Scannez avec votre téléphone pour payer instantanément</p>
</div>
```

# Meilleures Pratiques de Style PDF

OmniCRM utilise **WeasyPrint** pour convertir HTML en PDF. Suivez ces directives :

**CSS Supporté :**

- La plupart des propriétés CSS 2.1
- CSS3 limité (flexbox, certaines transformations)
- Polices Web via `@font-face`

**Non Supporté :**

- JavaScript
- CSS Grid (utilisez des tableaux à la place)
- Animations complexes
- Certaines propriétés CSS modernes

**Taille de Page et Marges :**

```
@page {
    size: A4;
    margin: 1cm;
}

body {
    font-family: Arial, sans-serif;
    font-size: 10pt;
}
```

**Style Spécifique à l'Impression :**

```
@media print {
    .no-print {
        display: none;
    }

    .page-break {
        page-break-after: always;
    }
}
```

**Disposition des Tableaux :**

```
.table {
    table-layout: fixed;
    width: 100%;
}

.table th, .table td {
    word-wrap: break-word;
    padding: 4px;
}
```

**Intégration de Polices :**

Pour des polices personnalisées, utilisez des polices sécurisées pour le web ou intégrez :

```
@font-face {
    font-family: 'YourFont';
    src: url('https://yourcompany.com/fonts/yourfont.woff2')
format('woff2');
}

body {
    font-family: 'YourFont', Arial, sans-serif;
}
```

# Tester les Modèles de Factures

**Liste de Vérification des Tests :**

1. **Inspection Visuelle :**
   - Le logo s'affiche correctement
   - Les couleurs correspondent aux directives de la marque
   - Le texte est lisible (pas trop petit)
   - Les tableaux sont correctement alignés
   - Toutes les sections sont présentes
2. **Exactitude des Données :**
   - Détails du client corrects
   - Montants des transactions additionnés correctement
   - Dates formatées correctement
   - Toutes les variables substituées correctement
3. **Qualité du PDF :**
   - Taille de fichier raisonnable (<5MB)
   - Images nettes et claires
   - Aucun texte coupé ou débordement
   - Sauts de page aux endroits appropriés
4. **Factures Multi-Page :**
   - En-têtes répétés sur chaque page
   - Numéros de page affichés
   - Longues listes de transactions paginées correctement
5. **Livraison par Email :**
   - PDF attaché à l'email
   - Taille de fichier inférieure à la limite de Mailjet (15MB)
   - S'affiche dans Gmail, Outlook, Apple Mail

**Commande de Test (Génération Manuelle) :**

Vous pouvez tester la génération de factures via l'API :

```
curl -X GET "http://localhost:5000/crm/invoice/{invoice_id}/pdf" \
     -H "Authorization: Bearer YOUR_TOKEN" \
     --output test_invoice.pdf
```

# Problèmes Courants de Modèle

**Variables non substituées :**

- **Cause :** Fautes de frappe dans le nom de la variable ou données manquantes
- **Solution :** Vérifiez l'orthographe exactement (sensible à la casse), vérifiez que les données existent dans la base de données

**Style PDF cassé :**

- **Cause :** Propriété CSS non supportée
- **Solution :** Utilisez des propriétés CSS 2.1, testez avec CSS compatible WeasyPrint

**Images ne s'affichent pas :**

- **Cause :** URLs relatives ou ressources externes bloquées
- **Solution :** Utilisez des URLs HTTPS absolues, assurez-vous que les images sont accessibles publiquement

**Tableaux débordant de la page :**

- **Cause :** Largeurs de colonnes fixes trop larges
- **Solution :** Utilisez des largeurs en pourcentage, `table-layout: fixed`

**Polices ne s'affichant pas :**

- **Cause :** Police non intégrée ou indisponible
- **Solution :** Utilisez des polices sécurisées pour le web (Arial, Times New Roman, etc.) ou intégrez correctement des polices personnalisées

**Échec de la génération de PDF :**

- **Cause :** Erreurs de syntaxe HTML ou crash de WeasyPrint
- **Solution :** Validez le HTML, vérifiez les journaux de WeasyPrint, simplifiez les mises en page complexes

# Mise en Cache des PDF de Factures

Pour améliorer les performances et réduire la génération redondante de PDF, OmniCRM inclut un système de mise en cache des PDF de factures. Lorsqu'un PDF de facture est généré pour la première fois, il est mis en cache dans la base de données pour les demandes suivantes.

**Comment Fonctionne la Mise en Cache des PDF :**

1. **Première Demande** - Lorsqu'un PDF de facture est demandé (téléchargement ou email), le système :
   - Génère le PDF à partir du modèle de facture
   - Encode le PDF en Base64
   - Calcule un hachage SHA256 du contenu du PDF
   - Stocke dans la table `Invoice_PDF_Cache` avec :
     - Référence d'ID de facture
     - Données PDF (encodées en Base64)
     - Nom de fichier
     - Hachage de contenu (pour vérification d'intégrité)
     - Horodatage de création
2. **Demandes Suivantes** - Lorsque la même facture est demandée à nouveau :
   - Le système vérifie le PDF mis en cache par invoice_id
   - Si le cache existe et est valide, retourne immédiatement le PDF mis en cache
   - Met à jour l'horodatage `last_accessed` pour suivre l'utilisation du cache
3. **Invalidation du Cache** - Les PDF mis en cache sont invalidés lorsque :
   - La facture est modifiée (transactions ajoutées/enlevées, détails changés)
   - Le modèle de facture est mis à jour

    - Le nettoyage manuel du cache est déclenché

**Avantages :**

- **Performance** - Livraison instantanée de PDF pour les demandes répétées (pas de délai de régénération)
- **Cohérence** - Même PDF pour tous les téléchargements d'une facture (sauf si la facture est modifiée)
- **Charge Serveur** - Réduit l'utilisation du CPU pour la génération de PDF
- **Expérience Utilisateur** - Un indicateur de chargement apparaît lors de la génération initiale, les demandes suivantes sont instantanées

**Gestion du Cache :**

Le Cache PDF de Facture est géré automatiquement par le système. Les anciennes ou entrées de cache inutilisées peuvent être purgées périodiquement en fonction de :

- Âge (par exemple, supprimer les entrées de cache de plus de 90 jours)
- Modèles d'accès (supprimer les entrées non accédées depuis 30 jours)
- Limites de stockage (mettre en œuvre des limites de taille de cache si nécessaire)

**Comportement de l'API :**

Lors du téléchargement d'une facture via l'API ou l'UI :

- Première demande : Affiche un indicateur de chargement pendant que le PDF se génère, puis met en cache
- Demandes suivantes : Téléchargement immédiat depuis le cache
- Le succès ou l'échec du cache est transparent pour l'utilisateur

**Important :** Lorsque vous mettez à jour votre modèle de facture, videz le cache pour vous assurer que les nouvelles factures utilisent le design mis à jour :

```sql
-- Vider tous les PDF de factures mis en cache (exécuter dans MySQL)
DELETE FROM Invoice_PDF_Cache;
```

Ou mettez à jour `crm_config.yaml` pour invalider automatiquement le cache lors du changement de modèle.

# Accéder aux Factures

Les factures peuvent être consultées au niveau du système ou par client :

**Vue par Client :**

1. Naviguez vers **Clients → [Sélectionner Client]**
2. Cliquez sur l'onglet **Facturation**
3. Consultez la liste des factures dans la troisième carte

**Vue Système :**

1. Naviguez vers **Facturation → Factures** (depuis le menu principal)
2. Consultez toutes les factures de tous les clients

# Widgets de Statistiques de Factures

En haut de la page des factures, quatre cartes statistiques affichent des résumés financiers.

**Descriptions des Widgets :**

- **Total des Factures** - Somme de tous les coûts de détail des factures (tous les temps) et nombre de factures envoyées
- **Factures Impayées** - Somme des factures non encore payées et nombre de factures impayées
- **Factures Ce Mois** - Somme des factures créées ce mois calendaire avec le compte
- **Factures Le Mois Dernier** - Somme des factures créées le mois calendaire dernier avec le compte

**Formatage des Valeurs :**

- Valeurs supérieures à 1 000 : Afficher avec suffixe "k" (par exemple, $1.5k)
- Valeurs supérieures à 1 000 000 : Afficher avec suffixe "M" (par exemple, $2.3M)
- Valeurs supérieures à 1 000 000 000 : Afficher avec suffixe "B" (par exemple, $1.1B)

**Indicateurs de Tendance :**

- Widgets pour "Ce Mois" et "Le Mois Dernier" montrent le changement en pourcentage
- Flèche verte vers le haut : Augmentation par rapport à la période précédente

- Flèche rouge vers le bas : Diminution par rapport à la période précédente
- Flèche grise vers la droite : Pas de changement

# Liste des Factures

Le tableau des factures affiche toutes les factures avec les colonnes suivantes :

**Descriptions des Colonnes :**

- **ID** - ID de facture unique
- **Titre** - Titre/description de la facture
- **Période** - Période de facturation (date de début - date de fin) ou "N/A" pour les factures uniques
- **Date d'Échéance** - Date d'échéance du paiement
- **Créé** - Date de création de la facture
- **Montant** - Montant total de la facture (coût de détail)
- **Statut** - Payé, Impayé ou Remboursé
- **Actions** - Actions disponibles (varie selon le statut)

**Icônes d'Action :**

- ⬇ **(Télécharger)** - Télécharger le PDF de la facture
- 🗑 **(Supprimer)** - Annuler la facture (uniquement si non payée)
- 💳 **(Payer)** - Payer la facture en ligne (uniquement si impayée)
- ✉ **(Email)** - Envoyer l'email de la facture au client
- 💸 **(Rembourser)** - Rembourser le paiement Stripe (uniquement pour les factures Stripe payées)

# Générer une Facture

Cliquez sur **"+ Générer une Facture Proforma"** pour créer une nouvelle facture.

**Descriptions des Champs :**

- **Rechercher des Clients** - Sélectionnez un client (uniquement affiché dans la vue système, pré-rempli dans la vue client)
- **Titre** - Titre/nom de la facture (optionnel, par défaut "Facture pour [Période]")
- **Date de Début** - Début de la période de facturation (par défaut 14 jours auparavant)
- **Date de Fin** - Fin de la période de facturation (par défaut aujourd'hui)
- **Date d'Échéance** - Date limite de paiement (par défaut aujourd'hui)

- **Aperçu de la Transaction** - Affiche toutes les transactions non facturées dans la plage de dates avec la possibilité d'inclure/exclure des transactions spécifiques

**Sélection de Transactions :**

- ✓ **(Plus Vert)** - Cliquez pour exclure une transaction de la facture
- × **(X Rouge)** - Cliquez pour inclure une transaction précédemment exclue
- **Tout Sélectionner** - Inclure toutes les transactions affichées
- **Tout Effacer** - Exclure toutes les transactions
- Les transactions exclues apparaissent grises avec un texte barré
- Les totaux en temps réel se mettent à jour au fur et à mesure que vous sélectionnez/désélectionnez des transactions

**Ce Qui Se Passe :**

1. Le système trouve toutes les transactions non facturées pour le client dans la plage de dates
2. Affiche l'aperçu des transactions avec la possibilité d'inclure/exclure des transactions individuelles
3. Montre le calcul en temps réel du sous-total, de la taxe et du total basé sur les transactions sélectionnées
4. Seules les transactions sélectionnées (incluses) sont ajoutées à la facture
5. Génère le PDF de la facture et le met en cache
6. Marque les transactions sélectionnées comme facturées (`invoice_id` champ peuplé)
7. Les transactions exclues restent non facturées et disponibles pour de futures factures
8. La facture apparaît dans la liste avec le statut "Impayé"

**Exemples de Cas d'Utilisation :**

**Facturation Mensuelle :** Définissez la date de début au premier du mois, la date de fin au dernier jour du mois, l'aperçu montre toutes les transactions non facturées de cette période. Sélectionnez tout ou excluez manuellement des transactions spécifiques.

**Facture Spécifique à un Service :** Utilisez la même plage de dates, puis excluez manuellement les transactions non désirées (par exemple, exclure les transactions non mobiles pour créer une facture uniquement mobile).

**Facture Unique :** Définissez à la fois la date de début et la date de fin au même jour, l'aperçu montre uniquement les transactions de cette date. Excluez les frais non pertinents pour cette facture spécifique.

# Voir les Détails de la Facture

Cliquez sur n'importe quelle ligne de facture dans le tableau pour voir les détails complets de la facture y compris toutes les transactions, totaux et actions disponibles.

**Modal des Détails de la Facture :**

- **Informations sur la Facture** - Affiche l'ID de la facture, le titre, les dates, le statut de paiement et le statut d'annulation
- **Liste des Transactions** - Affiche toutes les transactions incluses dans la facture avec :
  - Date de transaction
  - Titre et description
  - Coût de détail
  - Montant de la taxe et pourcentage (formaté comme `$10.00 (10%)`)

    - Les transactions exonérées de taxe affichent "-" dans la colonne Taxe
- **Résumé des Totaux** - Calcul en temps réel montrant :
    - Nombre de transactions
    - Sous-total (somme de tous les coûts de détail)
    - Taxe (somme de tous les montants de taxe)
    - Total de la Facture (sous-total + taxe)
- **Boutons d'Action** - Les mêmes actions disponibles que dans le tableau :
    - **Télécharger PDF** - Télécharger le PDF de la facture (toujours disponible)
    - **Envoyer un Email** - Email de la facture au client (factures non annulées)
    - **Payer la Facture** - Traiter le paiement (factures impayées, non annulées uniquement)
    - **Rembourser** - Rembourser le paiement Stripe (uniquement pour les factures Stripe payées)
    - **Supprimer** - Annuler la facture (factures impayées, non annulées uniquement)

# Télécharger les PDF de Factures

Cliquez sur l'**icône de téléchargement (⬇)** dans le tableau ou sur le bouton **"Télécharger PDF"** dans le modal des détails de la facture pour télécharger une facture au format PDF.

**Processus de Téléchargement :**

1. Cliquez sur l'icône de téléchargement à côté de la facture
2. Un indicateur de chargement apparaît pendant la génération (uniquement la première fois)
3. Le navigateur demande de sauvegarder le fichier : `Invoice_01234.pdf`
4. Le PDF s'ouvre ou se sauvegarde dans le dossier Téléchargements

**Comportement de Mise en Cache des PDF :**

- **Premier Téléchargement** - PDF généré à partir du modèle, mis en cache dans la base de données (peut prendre 2-3 secondes)

- **Téléchargements Suivants** - Téléchargement instantané depuis le cache
- **Invalidation du Cache** - Cache vidé si la facture est modifiée ou si le modèle est mis à jour

**Dépannage des Problèmes de Téléchargement :**

- **L'indicateur ne s'arrête jamais** - Vérifiez la console du navigateur, l'API peut être hors service
- **PDF vide ou corrompu** - Vérifiez le modèle de facture pour des erreurs de syntaxe
- **Le téléchargement échoue** - Vérifiez les paramètres du bloqueur de fenêtres contextuelles, essayez un autre navigateur

# Payer les Factures

Cliquez sur l'**icône de paiement (□)** pour payer une facture en ligne.

**Processus de Paiement :**

1. Cliquez sur l'icône de paiement sur la facture impayée
2. Le modal de paiement s'ouvre montrant les détails de la facture
3. Sélectionnez le mode de paiement :
    - **Transaction Stripe** - Facturer la carte de crédit enregistrée (disponible pour tous les utilisateurs)

    - **Espèces** - Paiement en espèces manuel (personnel uniquement)
    - **Remboursement** - Appliquer un remboursement comme paiement (personnel uniquement)
    - **Transaction POS** - Terminal de point de vente (personnel uniquement)
    - **Virement Bancaire** - Virement bancaire manuel (personnel uniquement)
4. Si Stripe est sélectionné :
    - Sélectionnez la carte parmi les méthodes de paiement enregistrées
    - La carte par défaut est pré-sélectionnée
    - Cliquez pour sélectionner une carte différente
5. Si un autre mode est sélectionné :
    - Entrez le numéro de référence (optionnel)
6. Cliquez sur **"Payer la Facture"** pour traiter
7. Le système traite le paiement :
    - **Stripe** - Charge la carte via l'API Stripe
    - **Autres méthodes** - Crée une transaction négative pour le montant du paiement
8. Le statut de la facture change en "Payé"
9. Notification de succès affichée

**Self-Care vs Paiement du Personnel :**

**:doc:`Portail de Self-Care <self_care_portal>` (Clients) :**

- Seulement le paiement Stripe disponible
- Doit avoir une méthode de paiement enregistrée
- Avertissement affiché si aucune méthode de paiement n'existe
- Lien pour ajouter une méthode de paiement fourni

**Portail du Personnel (Administrateurs) :**

- Tous les modes de paiement disponibles
- Peut marquer la facture comme payée manuellement (espèces, POS, virement bancaire)
- Peut entrer des numéros de référence pour le suivi

**Avertissement sur le Mode de Paiement :**

Si le client n'a pas de méthodes de paiement enregistrées, un avertissement est affiché l'invitant à ajouter une méthode de paiement avant de pouvoir payer les factures.

# Envoyer des Factures par Email

Cliquez sur l'**icône d'email (✉)** pour envoyer la facture au client.

**Ce Qui Se Passe :**

1. Cliquez sur l'icône d'email à côté de la facture
2. Le système récupère le PDF de la facture depuis le cache (ou génère s'il n'est pas mis en cache)
3. Envoie l'email via Mailjet en utilisant le modèle `api_crmCommunicationCustomerInvoice`
4. L'email comprend :
   - PDF de la facture en pièce jointe
   - Nom du client
   - Numéro de facture et date d'échéance
   - Montant total dû
   - Lien pour payer la facture en ligne
   - Lien pour voir/télécharger la facture
5. Notification de succès : "Email de facture envoyé avec succès"

**Destinataires de l'Email :**

L'email est envoyé à tous les contacts clients de type "facturation" ou au contact principal si aucun contact de facturation n'existe.

**Variables de Modèle d'Email :**

- `{{ var:customer_name }}` - Nom complet du client
- `{{ var:invoice_number }}` - ID de la facture
- `{{ var:invoice_date }}` - Date d'émission de la facture
- `{{ var:due_date }}` - Date d'échéance du paiement
- `{{ var:total_amount }}` - Montant total dû
- `{{ var:invoice_url }}` - Lien pour voir/télécharger le PDF
- `{{ var:pay_url }}` - Lien pour payer la facture en ligne

**Dépannage des Problèmes d'Email :**

- **Email non envoyé** - Vérifiez les identifiants API Mailjet dans `crm_config.yaml`
- **Client ne reçoit pas** - Vérifiez les adresses email de contact du client
- **PDF non attaché** - Vérifiez que la génération du PDF a réussi (essayez de télécharger d'abord)

# Annuler des Factures

Cliquez sur l'**icône de suppression (□)** pour annuler une facture.

**Exigences :**

- La facture doit être **Impayée**
- Les factures payées ne peuvent pas être annulées (doivent être remboursées à la place)

**Comment Annuler :**

1. Localisez la facture impayée dans la liste
2. Cliquez sur l'icône de suppression (□)

3. Confirmez dans le modal :

**Ce Qui Se Passe :**

- La facture est marquée comme `void = true`
- Toutes les transactions sont dissociées de la facture (`invoice_id` défini sur null)
- Les transactions redeviennent "non facturées"
- Les transactions peuvent être incluses dans une nouvelle facture
- La facture apparaît dans la liste avec le préfixe "Annulé :" dans le titre
- Les actions de la facture sont désactivées (pas de téléchargement, de paiement ou d'email)
- Peut être vue en filtrant pour les factures "Annulées"

**Notes Importantes :**

- Annuler n'est PAS la même chose que rembourser
- **Annuler** = "Cette facture n'aurait jamais dû exister" (erreur de facturation, doublon)
- **Rembourser** = "Inverser une facture payée valide" (retourner de l'argent au client)

# Rembourser des Factures

Cliquez sur l'**icône de remboursement (□)** pour rembourser une facture payée.

**Exigences :**

- La facture doit être **Payée**
- La facture doit être payée via **Stripe**
- La facture doit avoir une `payment_reference` valide (ID d'intention de paiement Stripe)
- Disponible uniquement pour les utilisateurs du personnel (pas de Self-Care)

**Comment Rembourser :**

1. Localisez la facture Stripe payée
2. Cliquez sur l'icône de remboursement (□)
3. Le modal de confirmation de remboursement s'ouvre :

4. Cliquez sur **"Confirmer le Remboursement"**
5. Le système traite le remboursement Stripe :
    - Appelle l'API Stripe pour rembourser le paiement
    - Crée une transaction de remboursement dans Stripe
    - Met à jour la facture avec `refund_reference`
6. Le statut de la facture change en "Remboursé"

7. Notification de succès affichée

**Ce Qui Se Passe Après le Remboursement :**

- La facture reste dans le système (non annulée)
- Le statut affiche "Remboursé"
- Les transactions restent liées à la facture
- Le client reçoit un remboursement sur le mode de paiement d'origine (3-7 jours ouvrables)
- Le tableau de bord Stripe affiche la transaction de remboursement

**Restrictions de Remboursement :**

- Ne peut pas rembourser les factures payées par espèces, POS ou virement bancaire (inversion manuelle requise)
- Ne peut pas rembourser partiellement (montant total de la facture uniquement)
- Ne peut pas rembourser deux fois

# Recherche et Filtrage des Factures

## Recherche

Utilisez la barre de recherche pour trouver des factures. Recherche dans :

- ID de la facture
- Titre de la facture
- Nom du client (vue système uniquement)

## Filtres

Appliquez des filtres pour affiner la liste des factures :

**Filtres Disponibles :**

- **Statut d'Annulation** - Tout, Annulé, Non Annulé

- **Statut de Paiement** - Tout, Payé, Pas encore Payé

**Actions de Filtre :**

- **Appliquer les Filtres** - Appliquer les filtres sélectionnés à la liste
- **Réinitialiser les Filtres** - Effacer tous les filtres et afficher toutes les factures

## Tri

Cliquez sur n'importe quel en-tête de colonne pour trier :

- **ID** - Trier par ID de facture (le plus récent/le plus ancien)
- **Titre** - Trier par ordre alphabétique
- **Date d'Échéance** - Trier par date d'échéance
- **Créé** - Trier par date de création
- **Montant** - Trier par coût de détail (le plus élevé/le plus bas)
- **Statut** - Trier par statut de paiement (payé en premier ou impayé en premier)

Cliquez à nouveau pour inverser la direction du tri (ascendant ↔ descendant).

## Pagination

Naviguez à travers de grandes listes de factures avec des contrôles de page affichant la page actuelle, le nombre total de pages, et un sélecteur d'éléments par page (10, 25, 50, ou 100 éléments).

# Flux de Travail de Facture Courants

## Flux de Travail 1 : Facturation Mensuelle avec Aperçu de Transaction

1. La fin du mois arrive (par exemple, 31 janvier)
2. Naviguez vers **Facturation → Factures**
3. Cliquez sur **"+ Générer une Facture Proforma"**

4. Sélectionnez le client (ou faites-le par client si vous facturez de nombreux clients)
5. Définissez les dates :
    - Date de Début : 2025-01-01
    - Date de Fin : 2025-01-31
    - Date d'Échéance : 2025-02-15 (15 jours à partir de maintenant)
    - Titre : "Services de Janvier 2025" (optionnel)
6. La section **Aperçu de la Transaction** apparaît montrant toutes les transactions non facturées de janvier
7. Examinez l'aperçu :
    - Toutes les transactions sont incluses par défaut
    - Vérifiez les totaux : Sous-total, Taxe et Total de la Facture
    - Vérifiez que tous les frais sont corrects
8. Cliquez sur **"Générer la Facture"** (le bouton affiche le nombre de transactions, par exemple, "Générer la Facture (15)")
9. Facture créée avec toutes les transactions sélectionnées
10. Cliquez sur la ligne de facture pour voir les détails et vérifier
11. Cliquez sur le bouton **"Envoyer un Email"** dans le modal de détails ou sur l'icône d'email dans le tableau
12. Le client reçoit un email de facture avec le PDF et le lien de paiement

# Workflow 2: Facturation Sélective des Transactions

1. Le client a plusieurs services (Mobile + Internet) et des frais divers
2. Souhaite des factures séparées pour chaque service
3. **Générer la première facture (Services Mobiles) :**
    - Cliquez sur **"+ Générer la Facture Proforma"**
    - Titre : "Services Mobiles - Janvier 2025"
    - Début/Fin : 1-31 Jan
    - Date d'échéance : 15 Févr
    - Dans l'aperçu des transactions, **exclure** toutes les transactions non mobiles :
        - Cliquez sur le bouton **X** à côté des transactions Internet

        - Cliquez sur le bouton **X** à côté des frais divers
        - Seules les transactions de services mobiles restent sélectionnées
    - Vérifiez que les totaux ne reflètent que les services mobiles
    - Cliquez sur **"Générer la Facture"** (affiche le nombre de transactions mobiles)
4. **Générer la deuxième facture (Services Internet) :**
    - Cliquez à nouveau sur **"+ Générer la Facture Proforma"**
    - Titre : "Services Internet - Janvier 2025"
    - Début/Fin : 1-31 Jan (même période)
    - Dans l'aperçu des transactions :
        - Les transactions mobiles déjà facturées (n'apparaissent pas)
        - Exclure les frais divers en utilisant le bouton **X**
        - Seules les transactions de services Internet restent
    - Cliquez sur **"Générer la Facture"**
5. **Générer la troisième facture (Frais Supplémentaires) :**
    - Cliquez à nouveau sur **"+ Générer la Facture Proforma"**
    - Titre : "Frais Supplémentaires - Janvier 2025"
    - Seuls les frais divers non facturés apparaissent dans l'aperçu
    - Cliquez sur **"Tout Sélectionner"** pour inclure tout
    - Cliquez sur **"Générer la Facture"**
6. Envoyez les trois factures au client

# Workflow 3: Exclusion des Transactions Contestées ou en Attente

1. La fin de la période de facturation arrive
2. Naviguez vers l'onglet **Facturation** du client
3. Cliquez sur **"+ Générer la Facture Proforma"**
4. Définissez les dates de la période de facturation
5. L'aperçu des transactions affiche 20 transactions
6. Le client a contesté un frais et un autre est en attente d'enquête
7. Dans l'aperçu des transactions :
    - Localisez la transaction contestée (par exemple, "Frais de dépassement de données")

    - Cliquez sur le bouton **X** pour l'exclure
    - Localisez la transaction en attente (par exemple, "Frais d'installation")
    - Cliquez sur le bouton **X** pour l'exclure
    - Le nombre de transactions se met à jour : "18 Transactions sélectionnées"
    - Les totaux se recalculent automatiquement
8. Passez en revue les totaux mis à jour (exclut les montants contestés)
9. Cliquez sur **"Générer la Facture (18)"**
10. La facture est générée avec uniquement les transactions approuvées
11. Les transactions contestées/en attente restent non facturées pour le prochain cycle de facturation

## Workflow 4: Revue Rapide de la Facture et Ajustement

1. Le personnel génère la facture mensuelle
2. L'aperçu des transactions montre un total élevé inattendu
3. Passez en revue chaque transaction dans l'aperçu :
    - Remarquez un frais dupliqué pour le même service
    - Cliquez sur **X** pour exclure le duplicata
    - Remarquez une transaction de test qui ne devrait pas être facturée
    - Cliquez sur **X** pour exclure la transaction de test
4. Les totaux se mettent à jour en temps réel
5. Vérifiez que le nouveau total correspond au montant attendu
6. Cliquez sur **"Générer la Facture"** avec les transactions corrigées
7. Revenez en arrière et annulez/supprimez les transactions exclues si nécessaire
8. Envoyez la facture au client en toute confiance

## Workflow 5: Facture d'Installation Unique

1. Le technicien sur le terrain termine l'installation
2. Le personnel ajoute manuellement la transaction d'installation
3. Naviguez vers l'onglet **Facturation** du client

4. Cliquez sur **"+ Générer la Facture Proforma"**
5. Définissez les dates :
    - Date de début : aujourd'hui
    - Date de fin : aujourd'hui
    - Date d'échéance : aujourd'hui + 7 jours
    - Titre : "Services d'Installation"
6. L'aperçu des transactions montre uniquement les transactions d'aujourd'hui
7. Vérifiez que le frais d'installation apparaît
8. Excluez tous les frais récurrents en utilisant le bouton **X** (s'ils sont présents)
9. Cliquez sur **"Générer la Facture"**
10. Envoyez au client immédiatement
11. Le client paie en ligne via Stripe

# Workflow 6: Revue de la Facture Avant Contact Client

1. Le client appelle avec une question de facturation
2. Le personnel navigue vers la liste des factures du client
3. **Cliquez sur la ligne de la facture** pour ouvrir la modal des Détails de la Facture
4. Passez en revue les informations de la facture :
    - ID de la facture, dates, statut
    - Toutes les transactions incluses avec descriptions
    - Détail des taxes par transaction
    - Montants sous-total, taxe et total
5. Répondez aux questions du client avec des détails exacts
6. Si le client demande un PDF, cliquez sur le bouton **"Télécharger PDF"** dans la modal
7. Si le client demande un renvoi d'email, cliquez sur le bouton **"Envoyer Email"**
8. Fermez la modal lorsque vous avez terminé

# Workflow 7: Correction d'une Erreur de Facturation

1. Le client signale un frais incorrect
2. Le personnel clique sur la ligne de la facture pour voir les détails
3. Passe en revue la liste des transactions dans la modal des Détails de la Facture
4. Identifie la transaction incorrecte
5. La facture est impayée, donc peut être annulée
6. Cliquez sur le bouton **"Supprimer"** dans le pied de modal
7. Confirmez l'annulation
8. Les transactions redeviennent non facturées
9. Le personnel modifie ou supprime la transaction incorrecte de la liste des transactions
10. Générez une nouvelle facture avec les transactions corrigées :
    - Utilisez l'aperçu des transactions pour exclure la transaction corrigée si nécessaire
    - Incluez uniquement les frais valides
11. Envoyez la facture corrigée au client

# Workflow 8: Traitement de Plusieurs Paiements

1. Le client apporte de l'argent pour payer plusieurs factures
2. Naviguez vers l'onglet **Facturation** du client
3. Affichez les factures impayées
4. Cliquez sur la première ligne de facture pour voir les détails
5. Vérifiez le montant et les transactions
6. Cliquez sur le bouton **"Payer la Facture"** dans le pied de modal
7. Sélectionnez le mode de paiement **"Espèces"**
8. Entrez la référence : "Espèces payées 2025-01-15"
9. Cliquez sur **"Payer la Facture"**
10. La modal se ferme, la facture est marquée comme "Payée"
11. Répétez pour les factures restantes
12. Toutes les factures sont maintenant marquées comme "Payées"

## Workflow 9: Traitement d'une Demande de Remboursement

1. Le client demande un remboursement pour un trop-perçu
2. Le personnel vérifie que la facture a été payée via Stripe
3. Naviguez vers la facture dans la liste
4. Cliquez sur la ligne de la facture pour voir les détails
5. Vérifiez les informations de paiement et le montant
6. Cliquez sur le bouton **"Rembourser"** dans le pied de modal (n'apparaît que pour les factures Stripe)
7. Confirmez le remboursement
8. Le système traite le remboursement Stripe
9. Le statut de la facture change en "Remboursée"
10. Le client reçoit le remboursement dans 3 à 7 jours ouvrables
11. Le personnel fait un suivi avec le client pour confirmer la réception

# Dépannage

### Impossible de générer la facture - Aucune transaction trouvée

- **Cause :** Aucune transaction non facturée dans la plage de dates spécifiée
- **Solution :** Vérifiez la liste des transactions, vérifiez que les transactions existent et ne sont pas déjà facturées. Ajustez la plage de dates ou retirez le filtre.

### Échec de la génération du PDF de la facture

- **Cause :** Erreur de syntaxe du modèle, plantage de WeasyPrint ou données client manquantes
- **Solution :** Vérifiez le HTML du modèle de facture pour des erreurs, vérifiez que les champs d'adresse du client sont remplis, consultez les journaux de l'API.

### Échec du paiement avec une erreur Stripe

- **Cause :** Carte refusée, fonds insuffisants, carte expirée ou problème d'API Stripe
- **Solution :** Essayez un autre mode de paiement, vérifiez que la carte est valide, consultez le tableau de bord Stripe pour connaître la raison du refus.

**Impossible d'annuler la facture**

- **Cause :** Facture déjà payée
- **Solution :** Les factures payées ne peuvent pas être annulées. Si un remboursement est nécessaire, utilisez la fonction de remboursement pour les factures Stripe ou créez manuellement une transaction de crédit.

**L'email de la facture ne s'envoie pas**

- **Cause :** Identifiants API Mailjet invalides, le client n'a pas de contact de facturation ou le modèle d'email est manquant
- **Solution :** Vérifiez la configuration de Mailjet dans `crm_config.yaml`, vérifiez les contacts du client, vérifiez que le modèle d'email de la facture existe.

**Le bouton de remboursement n'apparaît pas**

- **Cause :** Facture payée par espèces/POS/transfer bancaire (pas Stripe), ou facture non payée
- **Solution :** Le bouton de remboursement n'apparaît que pour les paiements Stripe. Pour d'autres modes de paiement, créez une transaction de crédit manuelle.

**Le téléchargement du PDF montre l'ancien design du modèle**

- **Cause :** PDF mis en cache avant la mise à jour du modèle
- **Solution :** Effacez le cache PDF de la facture : `DELETE FROM Invoice_PDF_Cache WHERE invoice_id = X;`

**Le client ne peut pas payer la facture (aucun mode de paiement)**

- **Cause :** Aucun mode de paiement enregistré dans le portail de Self-Care
- **Solution :** Le client doit ajouter une carte de crédit à la page **Modes de Paiement** avant de payer les factures.

**Plusieurs factures générées pour la même période**

- **Cause :** Le personnel a généré la facture deux fois, ou les plages de dates se chevauchent
- **Solution :** Annulez la facture en double. Ajustez les plages de dates pour éviter le chevauchement. Utilisez l'aperçu des transactions pour garantir des ensembles de transactions uniques.

**L'aperçu des transactions ne montre aucune transaction**

- **Cause :** Toutes les transactions dans la plage de dates ont déjà été facturées ou aucune transaction n'existe
- **Solution :** Vérifiez que la plage de dates est correcte. Vérifiez la liste des transactions pour confirmer que des transactions existent. Filtrez les factures pour voir quelle facture contient les transactions.

**Impossible d'exclure une transaction de la génération de la facture**

- **Cause :** Transaction déjà facturée ou problème de navigateur
- **Solution :** Vérifiez que la transaction apparaît dans l'aperçu avec une coche. Rafraîchissez la page et réessayez. Effacez le cache du navigateur si le problème persiste.

**Le total de la facture ne correspond pas au montant attendu**

- **Cause :** Transactions inattendues incluses, taxe non calculée ou transactions exclues toujours comptées
- **Solution :** Passez en revue l'aperçu des transactions avec soin. Vérifiez le coût de détail et la taxe de chaque transaction. Vérifiez que les transactions exclues sont grises. Vérifiez l'insigne de compte de transactions sur le bouton Générer la Facture.

**Le bouton Générer la Facture est désactivé**

- **Cause :** Aucune transaction sélectionnée ou plage de dates invalide
- **Solution :** Assurez-vous qu'au moins une transaction est incluse (pas exclue). Vérifiez que la Date de Début est avant la Date de Fin. Vérifiez que la Date d'Échéance est définie.

**La modal des Détails de la Facture ne s'ouvre pas**

- **Cause :** Erreur JavaScript ou page pas complètement chargée
- **Solution :** Rafraîchissez la page. Vérifiez la console du navigateur pour des erreurs. Essayez un autre navigateur. Vérifiez la connexion Internet.

**La taxe de la transaction ne s'affiche pas dans les Détails de la Facture**

- **Cause :** La transaction a un taux de taxe de 0 % ou tax_amount est nul
- **Solution :** Vérifiez que la transaction a un tax_percentage défini. Vérifiez que tax_amount a été calculé lors de la création de la transaction. Mettez à jour la transaction si nécessaire.

**Les boutons d'action manquent dans la modal des Détails de la Facture**

- **Cause :** La facture est annulée ou l'utilisateur n'a pas les autorisations
- **Solution :** Les factures annulées n'affichent que le bouton Télécharger PDF. Vérifiez le statut de la facture. Vérifiez le rôle et les autorisations de l'utilisateur.

# Documentation Connexe

- `integrations_mailjet` - Livraison d'email de facture et modèles
- `administration_configuration` - Configuration du modèle de facture
- `payments_transaction` - Création de transactions apparaissant sur les factures
- `payments_process` - Traitement des paiements de factures
- `payment_system_guide` - Référence API de paiement et configuration des fournisseurs

# Gestion des Méthodes de Paiement

Le système de Méthodes de Paiement d'OmniCRM permet aux clients et au personnel de gérer en toute sécurité les cartes de paiement en utilisant le **traitement de paiement multi-fournisseurs** (Stripe, PayPal, etc.). Les méthodes de paiement permettent la facturation automatique des services, les paiements uniques et les frais récurrents sans stocker de données de carte sensibles dans OmniCRM.

Voir aussi : Guide du Système de Paiement, Vue d'Ensemble de la Facturation, Traitement des Paiements, Factures.

# Vue d'Ensemble

Le système de méthodes de paiement fournit :

- **Stockage Sécurisé des Cartes** - Cartes tokenisées par les fournisseurs de paiement (Stripe, PayPal), jamais stockées dans OmniCRM
- **Support Multi-Fournisseurs** - Méthodes de paiement Stripe et PayPal prises en charge
- **Plusieurs Cartes** - Les clients peuvent stocker plusieurs méthodes de paiement
- **Sélection par Défaut** - Désignez la méthode de paiement préférée pour les frais automatiques
- **Suivi des Expirations** - Surveillez et mettez à jour les cartes expirantes
- **Service Autonome** - Les clients peuvent gérer leurs propres cartes via le Portail de Self-Care
- **Gestion du Personnel** - Le personnel de support peut ajouter/supprimer des cartes au nom des clients

**Méthodes de Paiement Supportées :**

- Cartes de Crédit (Visa, Mastercard, American Express, Discover)
- Cartes de Débit
- Cartes Prépayées (si prises en charge par le réseau de cartes)

**Non Stocké dans OmniCRM :**

Les détails de la carte sont tokenisés par les fournisseurs de paiement et stockés en toute sécurité. OmniCRM ne stocke que :

- Fournisseur de paiement (stripe, paypal)
- Marque de carte (Visa, Mastercard, etc.)
- Derniers 4 chiffres
- Mois/année d'expiration
- Nom/surnom du titulaire de la carte
- Jeton de méthode de paiement spécifique au fournisseur

# Accéder aux Méthodes de Paiement

**Depuis la Page Client :**

1. Naviguez vers **Clients → [Sélectionner le Client]**
2. Cliquez sur l'onglet **Facturation**
3. Faites défiler jusqu'à la section **Méthodes de Paiement**

Ou directement :

**Depuis le Tableau de Bord des Cartes Expirantes :**

Affichez tous les clients avec des cartes expirantes :

Cela montre une liste globale des cartes expirant dans les 60 jours suivants.

# Liste des Méthodes de Paiement

Le tableau des méthodes de paiement affiche toutes les cartes stockées pour un client :

**Descriptions des Colonnes :**

- **Surnom** - Nom amical pour la carte (par exemple, "Carte Personnelle", "Visa de Travail")
- **Émetteur** - Marque de carte et derniers 4 chiffres
- **Expiration** - Mois/année d'expiration (format MM/YYYY)
- **Ajouté** - Date à laquelle la carte a été ajoutée au compte
- **Par Défaut** - La coche indique la méthode de paiement par défaut pour les frais automatiques

**Actions par Carte :**

Chaque ligne a un menu d'actions ( ⋮ ) avec des options :

- **Définir comme Par Défaut** - Faites-en la méthode de paiement par défaut
- **Supprimer** - Retirer la carte du compte

# Ajouter une Méthode de Paiement

Cliquez sur **"Ajouter une Méthode de Paiement"** pour ouvrir la modal de paiement sécurisé.

# Étape 1 : Entrer les Détails de la Carte

Le formulaire de paiement sécurisé apparaît (propulsé par Stripe Elements ou PayPal SDK) :

**Champs Obligatoires :**

- **Informations sur la Carte** - Numéro de carte, expiration, CVC (validé par Stripe)
- **Nom du Titulaire de la Carte** - Nom sur la carte
- **Pays/Région** - Pays de facturation

**Champs Optionnels :**

- **Surnom de la Carte** - Étiquette amicale pour distinguer les cartes

**Sécurité :**

- Les détails de la carte sont saisis directement dans un iframe sécurisé hébergé par le fournisseur (Stripe Elements / PayPal SDK)
- OmniCRM ne voit jamais ni ne stocke les numéros de carte complets
- La conformité PCI DSS est gérée par le fournisseur de paiement
- La validation en temps réel empêche les numéros de carte invalides

# Étape 2 : Soumettre et Tokeniser

Lorsque vous cliquez sur **"Ajouter une Méthode de Paiement"** :

1. **Validation Côté Client :**
   - Le fournisseur de paiement valide le format du numéro de carte
   - Vérifie que la date d'expiration est dans le futur
   - Vérifie le format du CVC
2. **Tokenisation :**
   - Les détails de la carte sont envoyés directement au fournisseur de paiement (pas à OmniCRM)
   - Le fournisseur crée un jeton sécurisé (par exemple, `pm_1A2B3C4D` pour Stripe)
   - Le jeton est renvoyé à OmniCRM
3. **Traitement Serveur :**
   - OmniCRM enregistre le jeton dans le dossier client avec l'identifiant du fournisseur
   - Stocke les derniers 4 chiffres, la marque, l'expiration et le nom du fournisseur pour affichage
   - Aucun numéro de carte complet ne touche jamais les serveurs d'OmniCRM

# Étape 3 : Confirmation

Un message de succès apparaît :

> Votre Visa se terminant par 1234 a été ajoutée à votre compte.

La nouvelle carte apparaît dans le tableau des méthodes de paiement.

**Sélection Automatique par Défaut :**

- Si c'est la première carte du client, elle est automatiquement définie comme par défaut
- Si le client a déjà des cartes, la nouvelle carte est ajoutée comme non par défaut
- Le client peut changer le défaut après l'ajout

# Définir la Méthode de Paiement par Défaut

La méthode de paiement par défaut est utilisée pour :

- Les frais de service récurrents automatiques
- Les paiements de factures
- Les recharges et recharges
- Les transactions uniques (sauf indication contraire)

**Pour Changer le Défaut :**

1. Localisez la carte que vous souhaitez définir comme par défaut dans le tableau des méthodes de paiement
2. Cliquez sur le **menu d'actions ( ⋮ )** à côté de la carte
3. Sélectionnez **"Définir comme Par Défaut"**
4. Une confirmation apparaît

   La Visa se terminant par 5678 est maintenant votre méthode de paiement par défaut.

La coche se déplace vers la carte nouvellement sélectionnée.

**Indicateur Visuel :**

Les cartes par défaut montrent :

dans la colonne Par Défaut, généralement avec un badge de coche verte.

# Supprimer une Méthode de Paiement

Retirez les cartes qui sont expirées, perdues ou dont vous n'avez plus besoin.

# Étape 1 : Initier la Suppression

1. Trouvez la carte à supprimer dans le tableau des méthodes de paiement
2. Cliquez sur l'icône **corbeille** sur cette ligne

Si le client n'a qu'une seule méthode de paiement active, l'icône de corbeille est **désactivée**. En la survolant, un tooltip s'affiche :

> Impossible de supprimer votre seule méthode de paiement. Ajoutez d'abord une autre méthode de paiement.

Ajoutez une deuxième méthode de paiement avant de pouvoir retirer l'existante.

# Étape 2 : Confirmer la Suppression

Lorsque la suppression est autorisée, une modal de confirmation apparaît :

> Êtes-vous sûr de vouloir supprimer cette méthode de paiement ?
>
> [Annuler] [Supprimer la Méthode de Paiement]

Si la méthode à supprimer est la par défaut actuelle et que d'autres méthodes actives existent, la modal note qu'une autre méthode active sera promue par défaut après la suppression.

Cliquez sur **"Supprimer la Méthode de Paiement"** pour confirmer.

# Étape 3 : Suppression Terminée

Message de succès :

La carte est retirée du tableau et supprimée du fournisseur de paiement.

**Restrictions Importantes :**

- **Impossible de supprimer la seule méthode de paiement active** - Le contrôle de suppression est désactivé (avec tooltip). L'API rejette également la demande. Ajoutez une autre méthode de paiement d'abord.

- **Suppression de la par défaut lorsque d'autres existent** - Une autre méthode de paiement active utilisable est automatiquement promue par défaut.
- **Pas de retour en arrière** - La suppression est permanente ; le client doit réajouter la méthode si nécessaire.

# Gestion des Cartes Expirantes

OmniCRM suit les dates d'expiration des cartes et fournit des outils pour mettre à jour proactivement les cartes expirantes.

## Tableau de Bord des Cartes Expirantes

Naviguez vers **Facturation → Cartes Expirantes** pour voir une liste globale :

> Jours avant action d'expiration de la carte du client John Smith Visa ****1234 02/2025 12 jours Mettre à jour Acme Corp MC5678 03/2025 45 jours Mettre à jour Jane Doe Amex**9012 01/2025 EXPIRÉE Mettre à jour

**Filtres :**

- **Plage d'Expiration** - Prochains 30/60/90 jours ou déjà expiré
- **Type de Client** - Individuel vs Entreprise
- **Type de Service** - Filtrer par service nécessitant une méthode de paiement

**Actions :**

- **Mettre à jour** - Ouvre la page des méthodes de paiement du client pour ajouter une nouvelle carte
- **Notifier** - Envoyer un rappel par email au client (si Mailjet configuré)

# Notifications d'Expiration

Si Mailjet est configuré, des emails automatiques sont envoyés :

- **60 jours avant l'expiration** - Premier rappel
- **30 jours avant l'expiration** - Deuxième rappel
- **7 jours avant l'expiration** - Avertissement final
- **À l'expiration** - Avis de carte expirée

Les clients peuvent cliquer sur un lien dans l'email pour mettre à jour leur méthode de paiement via le portail de Self-Care.

**Variables de Modèle d'Email :**

Les modèles Mailjet reçoivent :

- Nom du client
- Marque de carte et derniers 4 chiffres
- Date d'expiration
- Lien vers la page des méthodes de paiement de Self-Care

Voir `integrations_mailjet` pour la configuration du modèle d'email.

# Mise à Jour d'une Carte Expirante

**Workflow Recommandé :**

1. Le client reçoit un email de notification d'expiration
2. Le client se connecte au portail de Self-Care
3. Navigue vers **Facturation → Méthodes de Paiement**
4. Clique sur **"Ajouter une Méthode de Paiement"**
5. Saisit les nouveaux détails de la carte (même carte avec expiration mise à jour, ou carte de remplacement)
6. Définit la nouvelle carte comme par défaut
7. Supprime l'ancienne carte expirée

**Workflow du Personnel :**

Si le client appelle le support :

1. Le personnel ouvre le compte du client
2. Navigue vers **Facturation → Méthodes de Paiement**
3. Ajoute une nouvelle carte au nom du client (le client fournit les détails par téléphone)
4. Définit la nouvelle carte comme par défaut
5. Supprime la carte expirée
6. Confirme avec le client

> ⚠ **ATTENTION**
>
> Ne demandez jamais aux clients d'envoyer par email ou par texto les détails de la carte. Utilisez toujours :
>
> - Le portail sécurisé de Self-Care pour l'auto-service
> - Le téléphone avec le personnel saisissant directement les détails dans le système
> - En personne dans un point de vente

## Que se passe-t-il lorsque les cartes expirent

Lorsqu'une carte de paiement atteint sa date d'expiration et n'est pas mise à jour :

**Effets Immédiats :**

1. **Les Paiements Automatiques Échouent**
    - Le fournisseur de paiement rejette les transactions avec des cartes expirées
    - Les renouvellements de services mensuels échouent à traiter
    - Les recharges automatiques échouent
    - Les paiements automatiques de factures échouent
2. **Notifications au Client**
    - Le système tente de facturer la carte
    - Notification d'échec de paiement envoyée
    - Email "Mettre à jour la Méthode de Paiement" envoyé avec un lien vers le portail de Self-Care
3. **Changements de Statut de Service**
    - **Services Postpayés** - Peuvent continuer temporairement avec un solde impayé
    - **Services Prépayés** - Suspension de service lorsque le solde s'épuise
    - **Services de Renouvellement Automatique** - Le renouvellement échoue, le service peut expirer

**Actions Suivantes :**

**Jour 1-3 (Période de Grâce) :**

- Le service continue normalement
- Le client reçoit le premier avis d'échec de paiement
- Le système tente une nouvelle tentative (en fonction de la configuration)

**Jour 4-7 :**

- Deuxième tentative de paiement (si configuré)
- Email d'avertissement envoyé
- Le service client peut contacter le client

**Jour 8-14 :**

- Le service peut être suspendu en cas de non-paiement
- Le statut suspendu empêche l'utilisation mais préserve le compte

- Le client peut restaurer en mettant à jour le mode de paiement et en payant le solde impayé

**Jour 15+:**

- Le service peut être résilié en cas de non-paiement
- Inventaire (cartes SIM, équipements) marqué pour retour
- Avis final envoyé
- Compte référé à des agences de recouvrement (si applicable)

**Prévenir l'interruption de service :**

Pour éviter une interruption de service :

- Mettre à jour les cartes **30 jours avant l'expiration**
- Ajouter plusieurs modes de paiement pour la redondance
- Activer les alertes de défaillance de paiement
- Surveiller le tableau de bord des cartes expirantes chaque semaine

**Restaurer le service après expiration :**

Si le service est suspendu en raison d'une carte expirée :

1. Ajouter un nouveau mode de paiement valide
2. Définir comme par défaut
3. Payer le solde impayé (le cas échéant)
4. Contacter le support pour réactiver le service
5. Service restauré en quelques minutes à quelques heures

# Sécurité des modes de paiement

## Tokenisation

OmniCRM utilise la tokenisation des fournisseurs pour garantir la sécurité :

1. **Le client saisit la carte** → Envoyée directement aux serveurs du fournisseur de paiement

2. **Le fournisseur valide et tokenise** → Crée un jeton unique
3. **Jeton stocké dans OmniCRM** → Le numéro de carte complet n'est jamais stocké
4. **Traitement du paiement** → Jeton envoyé au fournisseur, le fournisseur facture la carte

**Ce que stocke OmniCRM :**

```
{
  "vendor": "stripe",
  "vendor_payment_method_id": "pm_1A2B3C4D5E6F",
  "payment_type": "card",
  "brand": "visa",
  "last4": "1234",
  "exp_month": 12,
  "exp_year": 2026,
  "name": "John Smith",
  "nickname": "Personal Card",
  "is_default": true
}
```

**Ce que OmniCRM NE STOCKE PAS :**

- Numéro de carte complet
- Code CVV/CVC
- Données de bande magnétique
- Numéros PIN

## Conformité PCI

En utilisant des formulaires de paiement hébergés par le fournisseur :

- **Réduction de la portée PCI** - Les données de carte ne touchent jamais les serveurs d'OmniCRM
- **Champs hébergés par le fournisseur** - La saisie de la carte se fait dans l'iframe sécurisé du fournisseur
- **Pas de stockage de carte** - Des jetons sont utilisés au lieu de données de carte brutes

- **Transmission sécurisée** - Toute communication se fait via HTTPS/TLS

Voir Guide du système de paiement pour les détails de sécurité du fournisseur de paiement.

# Flux de travail communs

## Flux de travail 1 : Le client ajoute son premier mode de paiement

**Scénario :** Nouveau client s'inscrivant pour un service

1. Le client crée un compte
2. Sélectionne un plan de service
3. Invite à ajouter un mode de paiement lors du passage à la caisse
4. Saisit les détails de la carte dans la fenêtre modale Stripe
5. Carte tokenisée et enregistrée
6. Définie automatiquement comme par défaut
7. Service provisionné
8. Premier prélèvement traité

## Flux de travail 2 : Le client met à jour une carte expirante

**Scénario :** Carte de crédit sur le point d'expirer

1. Le client reçoit une notification par e-mail (60 jours avant l'expiration)
2. Se connecte au portail Self-Care
3. Navigue vers **Facturation → Modes de paiement**
4. Examine la carte actuelle expirant 12/2025
5. Clique sur **"Ajouter un mode de paiement"**
6. Saisit la carte de remplacement avec expiration 12/2028
7. Définit la nouvelle carte comme par défaut
8. Supprime l'ancienne carte

9. E-mail de confirmation envoyé

# Flux de travail 3 : Le personnel aide le client par téléphone

**Scénario :** Le client appelle : "Ma carte a été refusée"

1. Le client appelle le support
2. Le personnel vérifie l'identité (questions de sécurité)
3. Le personnel vérifie les modes de paiement : Carte expirée 01/2025
4. Personnel : "Votre carte a expiré. Avez-vous une nouvelle carte ?"
5. Le client fournit les détails de la nouvelle carte par téléphone
6. Le personnel navigue vers **Clients → [Client] → Facturation**
7. Clique sur **"Ajouter un mode de paiement"**
8. Saisit les détails de la carte pendant que le client les lit
9. Définit la nouvelle carte comme par défaut
10. Supprime la carte expirée
11. Réessaie le paiement échoué
12. Confirme avec le client : "Paiement réussi, service restauré"

# Flux de travail 4 : Client professionnel avec plusieurs cartes

**Scénario :** L'entreprise souhaite différentes cartes pour différents usages

1. Le client professionnel ajoute une carte principale (Visa se terminant par 1111)
2. Définit comme par défaut pour les frais de service mensuels
3. Ajoute une carte de secours (Mastercard se terminant par 2222) pour les recharges
4. Ajoute une carte d'achat (Amex se terminant par 3333) pour les achats d'équipement
5. Lors de la recharge, sélectionne manuellement Mastercard lors du passage à la caisse

6. La Visa par défaut est toujours utilisée pour la facturation mensuelle automatique

## Flux de travail 5 : Gestion des cartes expirantes (Admin)

**Scénario :** Gestion proactive des expirations

1. L'admin navigue vers **Facturation → Cartes expirantes**
2. Filtres : "Prochaines 30 jours"
3. Voit 15 clients avec des cartes expirantes
4. Sélectionne tout → **"Envoyer des e-mails de rappel"**
5. Mailjet envoie des e-mails personnalisés à chaque client
6. Les clients mettent à jour leurs cartes via Self-Care
7. L'admin examine la liste 1 semaine plus tard
8. Appelle les clients restants qui n'ont pas mis à jour
9. Aide à la mise à jour des cartes par téléphone

# Dépannage

**"Carte refusée" lors de l'ajout d'un mode de paiement**

- **Cause :** Stripe a rejeté la carte (fonds insuffisants, prévention de la fraude, refus de l'émetteur)
- **Solution :**
  - Essayer une autre carte
  - Contacter l'émetteur de la carte pour autoriser la transaction
  - S'assurer que la carte prend en charge les achats en ligne
  - Vérifier que l'adresse de facturation correspond à celle enregistrée

**"Erreur lors de l'ajout d'un mode de paiement" (erreur générique)**

- **Cause :** Erreur de l'API du fournisseur de paiement ou problème de réseau
- **Solution :**
  - Actualiser la page et réessayer

- Vérifier la connexion Internet
- Vérifier que la configuration du fournisseur de paiement est correcte dans les paramètres du système
- Vérifier la console du navigateur pour un message d'erreur spécifique
- Essayer un autre navigateur (désactiver les bloqueurs de publicité)

**Impossible de supprimer le mode de paiement (bouton désactivé)**

- **Cause :** C'est le seul mode de paiement actif du client
- **Solution :**
  - Ajouter d'abord un nouveau mode de paiement, puis supprimer l'ancien
  - Survoler l'icône de poubelle désactivée pour l'explication de l'info-bulle

**La carte apparaît comme expirée mais pas dans la liste des "Cartes expirantes"**

- **Cause :** La carte a expiré récemment, le cache n'a pas été actualisé
- **Solution :**
  - Actualiser la page
  - Vérifier les filtres sur le tableau de bord des cartes expirantes
  - Les cartes expirées peuvent passer à une vue différente

**La nouvelle carte n'apparaît pas immédiatement**

- **Cause :** La page ne s'est pas actualisée après l'ajout de la carte
- **Solution :**
  - Le tableau des modes de paiement doit se rafraîchir automatiquement
  - Si ce n'est pas le cas, actualiser manuellement le navigateur
  - Vérifier si une erreur s'est produite lors du processus d'ajout

**Le modal de paiement ne se charge pas**

- **Cause :** Le SDK du fournisseur de paiement ne se charge pas, problème de clé API ou extension de navigateur bloquant
- **Solution :**
  - Vérifier la console du navigateur pour des erreurs
  - Désactiver les bloqueurs de publicité et la protection contre le suivi

- Vérifier la configuration du fournisseur de paiement dans les paramètres du système
- S'assurer que le script SDK du fournisseur se charge (vérifier l'onglet Réseau)
- Essayer le mode navigation privée/incognito

**Le client ne reçoit pas de notifications d'expiration**

- **Cause :** Mailjet non configuré ou modèle d'e-mail manquant
- **Solution :**
  - Vérifier les identifiants Mailjet dans *crm_config.yaml*
  - Vérifier que le modèle d'e-mail existe pour l'expiration de la carte
  - Confirmer que l'adresse e-mail du client est valide
  - Vérifier les journaux Mailjet pour les échecs de livraison

# Meilleures pratiques

**Pour les clients :**

- Ajouter un mode de paiement avant l'activation du service pour éviter les retards
- Conserver au moins 2 cartes sur le dossier pour la redondance
- Mettre à jour les cartes expirantes 30 jours ou plus avant l'expiration
- Supprimer les anciennes cartes expirées pour éviter la confusion
- Utiliser des surnoms descriptifs ("Visa personnelle", "Amex de travail")
- Vérifier que le mode de paiement par défaut est correct pour la facturation automatique

**Pour le personnel de support :**

- Vérifier l'identité du client avant d'accéder aux modes de paiement
- Ne jamais demander aux clients d'envoyer les détails de la carte par e-mail/SMS/chat
- Traiter les ajouts de carte immédiatement lors des appels (ne pas différer)
- Confirmer que la nouvelle carte est définie comme par défaut après l'ajout

- Supprimer les anciennes cartes uniquement après avoir confirmé que la nouvelle carte fonctionne
- Tester le paiement après la mise à jour d'une carte expirée (traiter une autorisation de £0.01)

**Pour les administrateurs :**

- Surveiller le tableau de bord des cartes expirantes chaque semaine
- Envoyer des e-mails de rappel 60/30/7 jours avant l'expiration
- Garder les clés de test/vie du fournisseur de paiement séparées pour le développement et la production
- S'assurer que les modèles Mailjet sont configurés pour les notifications d'expiration
- Examiner les rapports de paiement échoué pour identifier les cartes expirées
- Former le personnel sur les procédures de gestion sécurisée des cartes

**Meilleures pratiques de sécurité :**

- Utiliser uniquement les clés publiables du fournisseur (jamais de clés secrètes dans le frontend)
- S'assurer que toutes les pages de paiement se chargent via HTTPS
- Examiner régulièrement le tableau de bord du fournisseur de paiement pour détecter une activité suspecte
- Activer les règles de prévention de la fraude du fournisseur
- Exiger le CVC pour toutes les transactions sans carte présente
- Enregistrer les modifications de mode de paiement dans le journal d'activité

# Documentation connexe

- Guide du système de paiement - Configuration et intégration du fournisseur de paiement
- `payments_process` - Traitement des paiements avec des modes de paiement enregistrés

- `payments_invoices` - Paiement automatique des factures en utilisant la carte par défaut
- `features_topup_recharge` - Système de recharge utilisant des modes de paiement
- `basics_payment` - Concepts généraux de paiement et de facturation
- `customer_care` - Portail Self-Care pour que les clients gèrent leurs propres cartes

# Traitement des paiements

La majorité des paiements seront traités automatiquement par le système, mais il y a des moments où vous devrez peut-être traiter un paiement manuellement.

Pour payer une facture, sélectionnez la facture impayée et cliquez sur le bouton "Payer".

Cela ouvrira un formulaire de paiement, où vous pourrez entrer le mode de paiement et cliquer sur "Soumettre" pour traiter le paiement.

Le client recevra automatiquement un reçu pour le paiement, et la facture sera marquée comme payée.

Pour les virements bancaires, vous pouvez entrer la référence de paiement et la date à laquelle le paiement a été effectué (si différente de la date actuelle).

# Aperçu de l'onglet Facturation

L'onglet Facturation fournit une vue unifiée de toutes les informations financières pour un client, combinant les méthodes de paiement, les transactions et les factures dans une seule interface pour une gestion efficace de la facturation.

Documentation connexe : Méthodes de paiement, Transactions, Factures, Traitement des paiements, Portefeuille vs grand livre des transactions clients.

## Accéder à l'onglet Facturation

**Vue par client :**

1. Accédez à **Clients → [Sélectionner le client]**
2. Cliquez sur l'onglet **Facturation**
3. Consultez les trois sections : Méthodes de paiement, Transactions et Factures

**Vues système :**

Les données de facturation à l'échelle du système peuvent être consultées séparément :

- **Facturation → Transactions** - Toutes les transactions de tous les clients
- **Facturation → Factures** - Toutes les factures de tous les clients

**Portail d'auto-assistance :**

Les clients accédant au Portail d'auto-assistance voient la même structure d'onglet Facturation :

- Voir et gérer leurs méthodes de paiement
- Voir l'historique des transactions

- Voir et payer les factures en ligne

# Structure de l'onglet Facturation

L'onglet Facturation est organisé en trois sections principales, affichées sous forme de cartes :

## Section 1 : Méthodes de paiement

**Objectif :** Gérer comment le client paie pour les services

**Fonctionnalités clés :**

- Voir toutes les cartes de crédit enregistrées
- Définir la méthode de paiement par défaut
- Ajouter de nouvelles méthodes de paiement (via Stripe)
- Supprimer les cartes expirées ou inutilisées

**Documentation :** `basics_payment`

## Section 2 : Transactions

**Objectif :** Suivre tous les frais et crédits pour le client

**Fonctionnalités clés :**

- Voir les statistiques des transactions (Total, Non facturé, Ce mois-ci, Le mois dernier)
- Lister toutes les transactions avec filtrage par statut d'annulation/facturation
- Ajouter des transactions manuelles (frais ou crédits)
- Annuler des transactions incorrectes
- Voir quelles transactions sont facturées vs non facturées

**Documentation :** `payments_transaction`

## Section 3 : Factures

**Objectif :** Regrouper les transactions en factures à payer par le client

**Fonctionnalités clés :**

- Voir les statistiques des factures (Total, Impayées, Ce mois-ci, Le mois dernier)

- Lister toutes les factures avec filtrage par statut payé/annulé
- Générer de nouvelles factures à partir de transactions non facturées
- Télécharger des PDF de factures
- Envoyer des factures par e-mail aux clients
- Payer des factures en ligne (Stripe ou méthodes de paiement manuelles)

- Annuler ou rembourser des factures

**Documentation :** `payments_invoices`

# Flux de données entre les sections

Comprendre comment les données circulent entre les trois sections est crucial pour une gestion efficace de la facturation.

## Diagramme de flux

## Relation Transaction → Facture

**1. Création de transaction :**

Lorsqu'un service est provisionné ou qu'un frais manuel est ajouté :

- Transaction créée dans la section **Transactions**
- Statut de la transaction : **Non facturé**
- Champ `invoice_id` de la transaction est `null`

**Exemple :**

**2. Génération de facture :**

Lorsque le personnel génère une facture :

- Facture créée dans la section **Factures**
- Toutes les transactions non facturées dans la plage de dates regroupées dans la facture
- Champ `invoice_id` de la transaction peuplé
- Statut de la transaction change en : **Facturé**

**Exemple :**

**3. Mise à jour des statistiques de transaction :**

- Le total des **Transactions Non facturées** diminue
- La statistique des **Total des Factures** augmente
- Le total des **Factures Impayées** augmente

# Relation Facture → Paiement

**1. Paiement de la facture :**

Lorsque le client paie la facture :

- Paiement traité en utilisant la **Méthode de Paiement** enregistrée (carte Stripe)
- Ou méthode de paiement manuelle sélectionnée (espèces, POS, virement bancaire)
- Statut de la facture change en : **Payée**

**2. Transaction de paiement créée :**

Pour les paiements manuels (non-Stripe) :

- Transaction négative créée automatiquement
- Titre de la transaction : "Paiement pour la Facture #1234"
- Montant de la transaction : -$45.00 (négatif, crédite le client)
- Champ `invoice_id` de la transaction : Lien vers la facture payée

**Exemple :**

**3. Mise à jour des statistiques :**

- Le total des **Factures Impayées** diminue
- Le **Total des Factures Ce Mois-ci** reste inchangé (la facture existait déjà)

# Relation Méthode de Paiement → Facture

**Flux de Paiement Stripe :**

1. Le client ajoute une carte de crédit dans **Méthodes de paiement**
2. Carte tokenisée via Stripe, stockée en toute sécurité
3. Lors du paiement de la facture, le client sélectionne la carte enregistrée
4. Stripe facture la carte
5. Facture marquée comme payée
6. Champ `payment_reference` peuplé avec l'ID d'intention de paiement Stripe

**Flux de Paiement Manuel :**

1. Le client paie par espèces/POS/virement bancaire (aucune méthode de paiement nécessaire)
2. Le personnel sélectionne la méthode de paiement dans la modal de Paiement de Facture
3. Le personnel entre le numéro de référence (optionnel)
4. Transaction négative créée pour le montant du paiement
5. Facture marquée comme payée

# Flux de travail de facturation complet

Ces flux de travail démontrent comment les trois sections fonctionnent ensemble pour accomplir des tâches courantes.

## Flux de travail 1 : Configuration d'un nouveau client et première facture

**Objectif :** Configurer la facturation pour un nouveau client et collecter le premier paiement

1. **Ajouter une méthode de paiement :**
   - Accédez au client → onglet **Facturation**
   - Section **Méthodes de paiement** → Cliquez sur **"Ajouter une méthode de paiement"**
   - Le client ajoute une carte de crédit via Stripe
   - Carte enregistrée comme méthode de paiement par défaut
2. **Vérifier les transactions :**
   - La section **Transactions** montre les transactions non facturées :
     - Frais de configuration de service : $50.00
     - Service du premier mois : $45.00
     - Total Non facturé : $95.00
3. **Générer une facture :**
   - Section **Factures** → Cliquez sur **"Générer une facture proforma"**
   - Définir la plage de dates pour inclure la configuration et le premier mois
   - Cliquez sur **"Générer une facture"**
   - Facture #INV-2025-001234 créée pour $95.00
4. **Mise à jour des transactions :**
   - Les deux transactions montrent maintenant : Facture #INV-2025-001234
   - Le total des **Transactions Non facturées** est maintenant $0.00
5. **Envoyer la facture par e-mail :**
   - Cliquez sur l'icône d'e-mail à côté de la facture

- Le client reçoit un e-mail de facture avec PDF et lien de paiement

6. **Le client paie en ligne :**
   - Le client clique sur le lien de paiement dans l'e-mail
   - Redirigé vers le portail d'auto-assistance
   - Cliquez sur le bouton **"Payer la facture"**
   - Sélectionnez la méthode de paiement par défaut
   - Cliquez sur **"Payer la facture"**
   - Stripe facture la carte
7. **Mise à jour de la facture :**
   - Le statut de la facture change en **"Payée"**
   - Le total des **Factures Impayées** diminue de $95.00

**Résultat :** Client entièrement configuré avec méthode de paiement, première facture payée.

# Flux de travail 2 : Facturation récurrente mensuelle

**Objectif :** Facturer tous les clients pour le service mensuel à la fin du mois

1. **Services auto-facturés :**
   - La fin du mois arrive (31 janvier)
   - Le système de facturation crée automatiquement des transactions pour tous les services récurrents
   - La section **Transactions** montre de nouvelles transactions non facturées
2. **Vérifier les transactions non facturées :**
   - Accédez à la section **Transactions**
   - Filtrer : **Statut de la Facture : Non facturé**
   - Vérifiez la liste de toutes les transactions prêtes pour la facturation
   - Vérifiez que les montants et descriptions sont corrects
3. **Générer des factures :**
   - Accédez à **Facturation → Factures** (à l'échelle du système)
   - Pour chaque client (ou utilisez le processus par lot) :
     - Cliquez sur **"Générer une facture proforma"**

        - Sélectionnez le client
        - Date de début : 2025-01-01
        - Date de fin : 2025-01-31
        - Date d'échéance : 2025-02-15
        - Cliquez sur **"Générer une facture"**
4. **Mise à jour des transactions :**
    - Toutes les transactions sont maintenant liées aux factures
    - Les totaux des **Transactions Non facturées** sont réinitialisés à $0.00
5. **Envoyer toutes les factures par e-mail :**
    - Pour chaque facture, cliquez sur l'icône d'e-mail
    - Tous les clients reçoivent des factures mensuelles
6. **Les clients paient :**
    - Les clients avec des méthodes de paiement enregistrées paient en ligne via l'auto-assistance
    - Le personnel traite les paiements en espèces/POS pour les clients qui paient en personne
    - Le total des **Factures Impayées** diminue à mesure que les paiements sont reçus

**Résultat :** Tous les clients facturés pour janvier, factures envoyées, paiements traités.

# Flux de travail 3 : Gestion du crédit pour problème de service

**Objectif :** Créditez le client pour une panne de service, appliquez-le à une facture impayée

1. **Le client signale un problème :**
    - Le service était hors ligne pendant 2 jours
    - Le client mérite un crédit de $10
2. **Ajouter une transaction de crédit :**
    - Accédez au client → onglet **Facturation** → section **Transactions**
    - Cliquez sur **"+ Ajouter une transaction"**
    - Type de transaction : **Crédit**

- Type de crédit : **Paiement en espèces** (ou type approprié)
- Titre : "Crédit pour panne de service"
- Description : "Compensation pour panne de 2 jours 8-9 janv."
- Coût de détail : 10.00
- Cliquez sur **"Ajouter une transaction"**

3. **Transaction créée :**
   - La transaction apparaît dans la liste avec le montant : **-$10.00**
   - Statut de la transaction : **Non facturé**
   - Le total des **Transactions Non facturées** inclut maintenant -$10.00
4. **Appliquer à la facture :**
   - Si le client a déjà une facture impayée :
     - La facture reste impayée avec le montant d'origine
     - Le crédit sera appliqué à la prochaine génération de facture
   - Si vous générez une nouvelle facture :
     - Section **Factures** → Cliquez sur **"Générer une facture proforma"**
     - Inclure la plage de dates avec la transaction de crédit
     - Facture générée avec crédit appliqué :
5. **Le client paie :**
   - Le client paie le montant réduit : $35.00
   - La facture est marquée comme payée

**Résultat :** Client crédité pour la panne, cr��dit appliqué à la prochaine facture, paiement réduit collecté.

# Flux de travail 4 : Méthode de paiement expirée - Mise à jour et réessai

**Objectif :** La carte du client a expiré, provoquant un échec de paiement - mettre à jour la carte et réessayer le paiement

1. **Notification d'échec de paiement :**
   - Le client tente de payer la facture
   - Stripe renvoie une erreur : "Carte expirée"
   - Le paiement échoue, la facture reste impayée

2. **Mettre à jour la méthode de paiement :**
    - Le client accède à l'onglet **Facturation**
    - Section **Méthodes de paiement** → Cliquez sur **"Ajouter une méthode de paiement"**
    - Entrez les nouvelles informations de carte (date d'expiration mise à jour)
    - Nouvelle carte enregistrée
3. **Définir comme par défaut :**
    - Le client clique sur **"Définir comme par défaut"** sur la nouvelle carte
    - Ancienne carte supprimée automatiquement (si désiré)
4. **Réessayer le paiement :**
    - Accédez à la section **Factures**
    - Localisez la facture impayée
    - Cliquez sur l'icône **"Payer"**
    - La modal de paiement s'ouvre avec la nouvelle carte par défaut pré-sélectionnée
    - Cliquez sur **"Payer la facture"**
    - Stripe facture la nouvelle carte avec succès
5. **Mise à jour de la facture :**
    - Le statut de la facture change en **"Payée"**
    - Champ `payment_reference` peuplé avec la nouvelle intention de paiement Stripe ID

**Résultat :** Client a mis à jour sa méthode de paiement, facture payée avec succès avec la nouvelle carte.

# Flux de travail 5 : Annuler une facture incorrecte et refacturer

**Objectif :** Facture générée par le personnel avec de mauvaises transactions - annuler et régénérer correctement

1. **Erreur découverte :**
    - Facture #INV-2025-001234 générée avec une plage de dates incorrecte
    - Transactions incluses du mauvais mois
    - Facture est impayée

2. **Annuler la facture :**
    - Accédez à l'onglet **Facturation** → section **Factures**
    - Localisez la facture incorrecte
    - Cliquez sur l'icône de suppression (🗑)
    - Confirmer l'annulation
    - Facture annulée
3. **Transactions libérées :**
    - Accédez à la section **Transactions**
    - Toutes les transactions de la facture annulée montrent maintenant : **Non facturé**
    - Le total des **Transactions Non facturées** augmente
    - Transactions disponibles pour une nouvelle facture
4. **Générer la facture correcte :**
    - Section **Factures** → Cliquez sur **"Générer une facture proforma"**
    - Définir la plage de dates correcte
    - Appliquer un filtre si nécessaire (par exemple, "Mobile" pour une facture uniquement mobile)
    - Cliquez sur **"Générer une facture"**
    - Nouvelle facture créée avec les transactions correctes
5. **Vérifier et envoyer par e-mail :**
    - Vérifiez les détails de la nouvelle facture
    - Vérifiez que les transactions correctes sont incluses
    - Cliquez sur l'icône d'e-mail pour l'envoyer au client

**Résultat :** Facture incorrecte annulée, transactions refacturées correctement, le client reçoit la facture corrigée.

# Flux de travail 6 : Traitement d'un paiement en espèces pour plusieurs factures

**Objectif :** Le client paie plusieurs factures impayées avec un seul paiement en espèces

1. **Le client arrive avec des espèces :**
    - Le client apporte 300 $ en espèces pour payer les factures en attente

    - Accédez au client → onglet **Facturation**
2. **Vérifier les factures impayées :**
    - Section **Factures** → Filtrer : **Payé : Pas encore payé**
    - Voir les factures impayées :
3. **Payer la première facture :**
    - Cliquez sur l'icône de paiement sur la facture #1234
    - La modal de paiement s'ouvre
    - Sélectionnez la méthode de paiement **"Espèces"**
    - Entrez la référence : "Espèces payées 2025-02-10 - Reçu #001"
    - Cliquez sur **"Payer la facture"**
    - La facture #1234 est marquée comme **"Payée"**
4. **Payer les factures restantes :**
    - Répétez le processus pour la facture #1235 :
        - Référence : "Espèces payées 2025-02-10 - Reçu #001"
    - Répétez pour la facture #1236 :
        - Référence : "Espèces payées 2025-02-10 - Reçu #001"
5. **Vérifier les transactions :**
    - Accédez à la section **Transactions**
    - Trois nouvelles transactions de paiement créées :
    - Toutes liées aux factures respectives
6. **Mise à jour des statistiques :**
    - Section **Factures** → Le total des **Factures Impayées** a diminué de 300,00 $.
    - Toutes les factures montrent maintenant le statut **"Payé"**

**Résultat :** Le client a payé toutes les factures en attente avec des espèces, les transactions de paiement enregistrées avec référence de reçu.

# Meilleures pratiques

## Pour les utilisateurs du personnel

**Gestion des transactions :**

- Ajouter des transactions manuelles immédiatement (ne pas retarder)

- Utiliser des titres et descriptions descriptifs pour plus de clarté
- Lier les transactions aux services et sites lorsque cela est applicable
- Annuler les transactions incorrectes avant qu'elles ne soient facturées

**Génération de factures :**

- Générer des factures à des intervalles cohérents (par exemple, mensuellement le 1er du mois)
- Utiliser les plages de dates avec soin pour éviter les chevauchements ou les lacunes
- Utiliser des filtres pour créer des factures spécifiques aux services lorsque cela est nécessaire
- Envoyer les factures par e-mail immédiatement après leur génération
- Vérifier les PDF de factures avant de les envoyer aux clients

**Traitement des paiements :**

- Vérifier que la méthode de paiement est valide avant d'essayer de facturer
- Toujours entrer des numéros de référence pour les paiements manuels (espèces, POS, virement bancaire)
- Marquer les factures comme payées immédiatement après réception du paiement
- Rembourser uniquement via Stripe (créer un crédit manuel pour d'autres méthodes de paiement)

**Hygiène des données :**

- Vérifier régulièrement les transactions non facturées
- Enquêter sur les transactions annulées pour comprendre les erreurs de facturation
- Surveiller les factures impayées et faire un suivi avec les clients
- Garder les méthodes de paiement à jour (supprimer les cartes expirées)

# Pour les clients (Portail d'auto-assistance)

**Méthodes de paiement :**

- Garder au moins une méthode de paiement valide sur le dossier
- Mettre à jour les méthodes de paiement avant que les cartes n'expirent
- Définir votre carte préférée comme par défaut

**Paiements de factures :**

- Payer les factures avant la date d'échéance pour éviter les frais de retard
- Vérifier les détails de la facture et les transactions avant de payer
- Télécharger les PDF de factures pour vos dossiers
- Contacter le support immédiatement si la facture semble incorrecte

**Examen des transactions :**

- Vérifier régulièrement l'historique des transactions
- Signaler immédiatement toute charge inattendue
- Comprendre quelles transactions sont facturées vs non facturées

# Pour les administrateurs

**Configuration du système :**

- Configurer les modèles d'e-mail Mailjet pour une livraison professionnelle des factures
- Personnaliser les modèles PDF de factures pour correspondre à la marque
- Configurer l'intégration des fournisseurs de paiement (Stripe, PayPal) pour un traitement sécurisé des paiements
- Configurer les conditions de paiement et les dates d'échéance

**Surveillance et reporting :**

- Utiliser des widgets statistiques pour surveiller la santé de la facturation
- Suivre le total des **Transactions Non facturées** - devrait diminuer après le cycle de facturation
- Surveiller le total des **Factures Impayées** - faire un suivi sur les paiements en retard
- Examiner les statistiques **Ce mois-ci** vs **Le mois dernier** pour les tendances

**Automatisation :**

- Automatiser les frais de services récurrents via la configuration des produits
- Configurer la génération automatique de factures pour la facturation récurrente (si disponible)
- Configurer des rappels par e-mail pour les factures en retard

# Problèmes courants et solutions

## Problème : Le client ne peut pas payer la facture

**Symptômes :**

- Le client clique sur le bouton de paiement mais rien ne se passe
- Message d'erreur : "Aucune méthode de paiement trouvée"

**Diagnostic :**

1. Accédez au client → onglet **Facturation** → section **Méthodes de paiement**
2. Vérifiez si le client a des méthodes de paiement enregistrées
3. Vérifiez si les cartes enregistrées sont expirées

**Solution :**

- Le client doit ajouter une méthode de paiement valide avant de payer les factures
- Guider le client vers la page **Méthodes de paiement** pour ajouter une carte de crédit
- Vérifier que la carte est acceptée (Visa, Mastercard, Amex, etc.)
- Réessayer le paiement après l'ajout de la carte

# Problème : Facture générée avec de mauvaises transactions

**Symptômes :**

- La facture inclut des transactions de la mauvaise période
- Facture manquant des transactions attendues
- Total de la facture incorrect

**Diagnostic :**

1. Ouvrir la facture dans la section **Factures**
2. Vérifiez les transactions incluses dans la facture
3. Vérifiez les dates des transactions par rapport à la plage de dates de la facture
4. Vérifiez si un filtre a été appliqué lors de la génération

**Solution :**

- **Si la facture est impayée :** Annuler la facture, vérifier que les transactions sont non facturées, régénérer avec la plage de dates correcte
- **Si la facture est payée :** Ne peut pas être annulée - créer une transaction de crédit pour le montant incorrect, générer une facture corrigée
- **Prévention :** Toujours vérifier la section **Transactions** avant de générer une facture pour vérifier que les transactions correctes seront incluses

# Problème : Le total des Transactions Non facturées ne diminue pas

**Symptômes :**

- Le widget **Transactions Non facturées** montre un montant élevé
- La liste des transactions montre de nombreuses transactions non facturées
- Des factures mensuelles ont déjà été générées

**Diagnostic :**

1. Filtrer les transactions par **Statut de la Facture : Non facturé**

2. Vérifiez la liste des transactions non facturées
3. Vérifiez les dates des transactions - peuvent être des frais récents après la dernière génération de facture
4. Vérifiez si certaines transactions sont annulées (ne devraient pas compter dans le total non facturé)

**Solution :**

- **Comportement attendu :** Les transactions créées après la dernière génération de facture restent non facturées jusqu'au prochain cycle de facturation
- **Si de vieilles transactions sont non facturées :** Générer une facture pour ces transactions avec la plage de dates appropriée
- **Si des transactions annulées sont comptées :** Le système devrait exclure automatiquement les transactions annulées - signaler un bug si ce n'est pas le cas

# Problème : La méthode de paiement continue de décliner

**Symptômes :**

- Le paiement Stripe échoue avec "Carte refusée"
- Le client signale que la carte devrait fonctionner
- Plusieurs tentatives de réessai échouent

**Diagnostic :**

1. Vérifiez le tableau de bord Stripe pour la raison du refus
2. Raisons courantes de refus :
    - Fonds insuffisants
    - Carte expirée
    - Carte signalée comme perdue/volée
    - Prévention de fraude bancaire
    - Carte internationale bloquée

**Solution :**

- Demandez au client de contacter sa banque pour autoriser le paiement
- Essayer une méthode de paiement différente (carte différente)
- Pour le personnel : Traiter comme paiement manuel (espèces, POS) et marquer la facture comme payée
- Vérifiez que l'adresse de facturation correspond à l'adresse de facturation de la carte

# Raccourcis clavier et conseils

**Navigation :**

- Accéder rapidement à l'onglet de facturation : Page client → Touche **B** (si les raccourcis clavier sont activés)

**Filtrage :**

- **Transactions :** Cliquez sur le filtre **"Non facturé"** avant de générer des factures pour voir exactement ce qui sera inclus
- **Factures :** Cliquez sur le filtre **"Pas encore payé"** pour voir toutes les factures en attente nécessitant un suivi

**Opérations en masse :**

- Sélectionnez plusieurs factures (case à cocher) pour supprimer/annuler plusieurs à la fois (personnel uniquement)
- Utilisez la barre de recherche pour trouver rapidement une transaction ou une facture spécifique par ID

**Actions rapides :**

- Cliquez sur le titre de la transaction/facture pour voir tous les détails dans une modal
- Menu d'actions clic droit ( ⋮ ) pour un accès rapide aux options de téléchargement/envoi par e-mail/paiement

# Documentation connexe

- `basics_payment` - Méthodes de paiement et intégration de paiement multi-fournisseurs
- `payment_system_guide` - Référence API de paiement et configuration des fournisseurs
- `payments_transaction` - Transactions en détail
- `payments_invoices` - Factures en détail
- `integrations_mailjet` - Livraison de factures par e-mail
- `csa_activity_log` - Consultation de l'historique des activités de facturation
- Surveillance & Métriques - Métriques et surveillance du système de facturation OCS/CGRateS

# Mouvements de portefeuille et transactions clients (alignement du grand livre)

## Vue d'ensemble

Les lignes `customer_transaction` constituent le grand livre de facturation / GL pour le CRM. Le `wallet_ledger` reste l'historique opérationnel du portefeuille. Pour le **rechargement de portefeuille par carte** et le **crédit manuel en espèces / portefeuille admin**, l'API crée désormais une `customer_transaction` correspondante avec un `retail_cost` **négatif** (fonds reçus), `payment_type` défini comme suit :

- `wallet_topup_card` : rechargement via méthode de paiement chargée (`POST /crm/wallet/topup`)
- `wallet_topup_cash` : crédit manuel (`POST /crm/wallet/credit`, par ex. espèces / banque)
- `wallet_adjustment_debit` : débit du portefeuille admin (`POST /crm/wallet/debit`) publie une ligne **positive** (frais d'ajustement)

Chaque ligne stocke `wallet_ledger_id` pointant vers la ligne correspondante du `wallet_ledger` pour réconciliation.

## Convention de signe vs GAAP

Dans ce sous-grand livre, un **`retail_cost`** **négatif** signifie une ligne qui **réduit** les frais nets du point de vue du client (par ex. rechargement de portefeuille /

crédits de type paiement) ; **positif** signifie une ligne de type **frais** (par ex. facture, débit d'ajustement de portefeuille). C'est une **convention de facturation CRM**, et non un substitut à la présentation formelle GAAP.

**US GAAP** ne mandate pas ce signe particulier pour les tables internes. La conformité dépend de **la comptabilité en partie double dans le grand livre général**, de **l'attribution des comptes** (espèces, AR, revenus non gagnés, etc.) et de la préparation des **états financiers**, et non de l'utilisation de plus/moins dans `customer_transaction.retail_cost`. Pour les opérateurs, il est préférable d'utiliser une interface utilisateur qui affiche **le montant absolu + l'étiquette** et éventuellement un **solde courant**, plutôt que de se fier uniquement au signe brut. Le tableau de facturation **Toutes les Transactions** formate `retail_cost` avec un **montant en gras** et un **exposant Cr** / **Dr** (frais de crédit / débit ; les infobulles l'expliquent) plus vert/rouge sous la colonne **Dû / crédit**, donc le type n'est pas transmis par la couleur seule. **Dr** ici signifie un **frais pour le client** (montant dû), et non "votre carte a été débitée".

# Payer une facture depuis le portefeuille

**Payer la facture** inclut **Uniquement le portefeuille** lorsque le client a un solde positif dans son portefeuille ; cela appelle `POST /crm/payments/invoice` avec `routing_mode: wallet_only` (le montant total doit être couvert). La **carte enregistrée** utilise toujours un routage **hybride** : le portefeuille est appliqué en premier, puis la carte pour tout reste.

# Déduire automatiquement le portefeuille lors de la création d'une facture ?

Non implémenté par défaut. **Prépayé vs postpayé** est une règle commerciale : annuler silencieusement le portefeuille lorsqu'une facture est émise peut surprendre les clients et complique les paiements partiels, les crédits et les

litiges. Modèle plus sûr : **émettre la facture**, puis **payer explicitement** (uniquement portefeuille, carte, hybride ou types manuels du personnel). Si le produit nécessite plus tard une application automatique, ajoutez un paramètre clair (par ex. par client ou par facture) et documentez l'impact sur le GL.

# Postpayé

Les crédits de portefeuille réduisent `customer.outstanding_balance` jusqu'au montant crédité (pour `customer_payment_type = postpaid`).

# Facturation

Les lignes de synchronisation de portefeuille non facturées sont des **prépaiements clients**. L'aperçu **Générer une Facture** sur la liste des factures **exclut** les lignes dont le `payment_type` commence par `wallet_` afin que les factures de routine ne soient pas accidentellement réduites par des crédits de prépaiement. Pour appliquer un prépaiement à une facture spécifique, utilisez un flux de travail qui sélectionne des `transaction_ids` explicites (si votre outil le prend en charge), ou gérez les ajustements en dehors de cet aperçu par défaut.

**Taxe :** Les nouvelles factures provenant de transactions en attente stockent **`customer_invoice.retail_cost` = somme(ligne `retail_cost`) + somme(ligne `tax_amount`)** (montant dû). Les réponses API de liste/détail **recalculent** également ce total à partir des lignes liées, de sorte que les anciennes factures sans taxe intégrée dans `retail_cost` affichent et facturent toujours le montant correct. Le sous-total/taxe/pied de page PDF utilise **les lignes d'articles**, et non uniquement l'en-tête brut de la facture.

# Permissions

- `POST /crm/wallet/topup`: utilisateurs authentifiés pouvant utiliser l'application (les clients utilisent le rechargement par carte).
- `POST /crm/wallet/credit` et `POST /crm/wallet/debit`: nécessitent **`manage_customer_wallet`** (accordé à `admin` et `customer_service_agent_2`

dans les semences de rôle par défaut). Relancez `populate_roles_permissions` (ou équivalent) après le déploiement afin que la permission existe et soit attribuée.

# "Crédit" manuel dans Ajouter une Transaction

L'interface utilisateur n'offre plus l'entrée de transaction de **crédit** manuel, car cela ne créditait pas le portefeuille. Utilisez **Facturation → Portefeuille → Espèces / manuel** (personnel) ou paiements de factures à la place.

# Tableaux (Gestion du Travail)

OmniCRM **Tableaux** fournit un système de gestion du travail flexible de type kanban pour suivre les tâches, les installations, les demandes de fonctionnalités et tout autre travail que votre équipe doit coordonner. Les tableaux se trouvent aux côtés du Système de Tickets de Support. Les tickets sont motivés par des problèmes clients. Les tableaux sont motivés par le travail et les projets. Les tableaux offrent aux équipes un moyen visuel d'organiser les éléments en groupes, de les déplacer à travers les statuts, d'assigner des propriétaires et de collaborer.

Voir aussi : Tickets de Support pour le suivi des problèmes clients, Utilisateurs et Rôles pour les permissions, Clients, Contacts et Sites pour les enregistrements auxquels vous pouvez lier des éléments.

## Aperçu

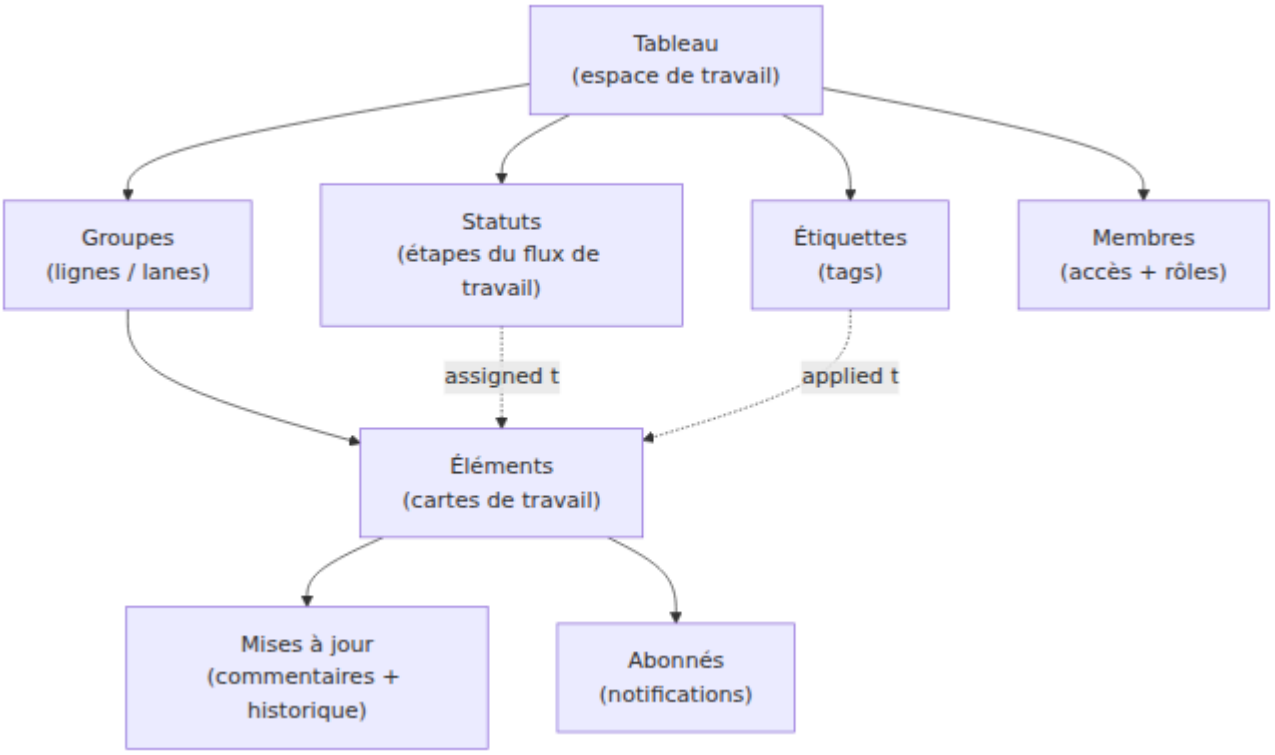
Les tableaux permettent :

1. **Suivi Visuel du Travail** : Organiser le travail en groupes (lignes) et statuts (étapes du flux de travail)
2. **Modèles** : Commencer rapidement avec des mises en page préconstruites pour les listes de tâches, les installations et les demandes de fonctionnalités
3. **Propriété** : Assigner des éléments au personnel, enregistrer qui les a demandés et suivre les dates d'échéance
4. **Liens** : Connecter des éléments à des clients, des services et des tickets de support
5. **Collaboration** : Commentaires, historique des modifications, @mentions et abonnements
6. **Contrôle d'Accès** : Adhésion par tableau avec des permissions basées sur les rôles

7. **Partage Public** : Exposer optionnellement un tableau en lecture seule via un lien sécurisé pour l'intégration

# Concepts de Base

Un tableau est construit à partir d'un petit ensemble de blocs de construction. Comprendre comment ils s'assemblent rend le reste de ce guide simple.

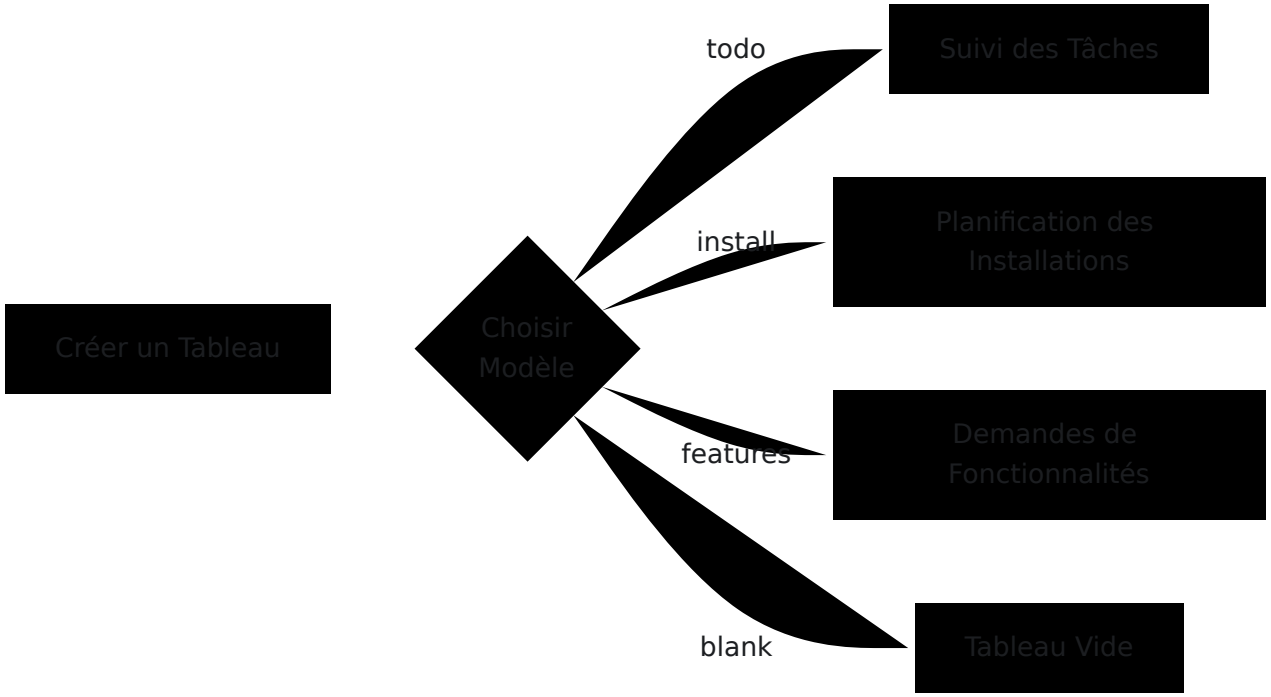


| Concept | Ce que c'est |
| --- | --- |
| **Tableau** | L'espace de travail de niveau supérieur. Contient ses propres groupes, statuts, étiquettes, membres et éléments. |
| **Groupe** | Une ligne horizontale (lane) qui organise les éléments en sections, par exemple *Cette Semaine*, *Semaine Prochaine*, *Plus Tard*. Les groupes peuvent être repliés. |
| **Statut** | L'étape du flux de travail d'un élément, par exemple *À Faire*, *En Cours*, *Fait*. Un statut est le défaut pour les nouveaux éléments. Tout statut peut être marqué comme un état "fait". |
| **Élément** | Une unité de travail unique (une carte). A un titre, une description, une priorité, un assigné, une date d'échéance, des étiquettes et un numéro d'élément unique. |
| **Étiquette** | Un tag codé par couleur appliqué aux éléments pour la catégorisation (par exemple *Bug*, *Fonctionnalité*). Un élément peut porter plusieurs étiquettes. |
| **Membre** | Un utilisateur qui a été accordé l'accès au tableau, avec un rôle qui contrôle ce qu'il peut faire. |

**Comment la mise en page fonctionne :** Les groupes sont les lignes du tableau. Au sein de chaque groupe, les éléments sont listés et montrent leur statut actuel, assigné, date d'échéance et priorité. Les éléments se déplacent *entre les groupes* pour les réorganiser, et *entre les statuts* pour les faire progresser à travers le flux de travail.

# Modèles de Tableau

Lors de la création d'un tableau, vous choisissez un modèle. Le modèle préremplit les statuts et groupes du tableau afin que vous puissiez commencer à travailler immédiatement. Vous pouvez ajouter, renommer, recolorer ou supprimer des statuts et des groupes par la suite.




| Modèle | But | Statuts (par défaut en premier) | Groupes |
|---|---|---|---|
| **Suivi des Tâches** (`todo`) | Listes de tâches générales | À Faire · En Cours · **Fait** | Principal |
| **Planification des Installations** (`install`) | Planification des installations sur le terrain/réseau | Prévu · En Cours · **Terminé** · En Attente | Cette Semaine · Semaine Prochaine · Plus Tard |
| **Demandes de Fonctionnalités** (`features`) | Pipeline produit/fonctionnalité | Nouveau · En Révision · Prévu · En Développement · **Publié** | Demandes de Fonctionnalités · En Cours · Terminé |
| **Tableau Vide** (`blank`) | Point de départ minimal | À Faire · **Fait** | Éléments |

Les statuts affichés en **gras** sont marqués comme états "faits". Les éléments dans ces statuts comptent comme complétés. Si aucun modèle n'est

sélectionné, la mise en page **Tableau Vide** est utilisée.

# Numérotation des Éléments

Chaque élément se voit attribuer un numéro unique, lisible par l'homme lorsqu'il est créé, afin qu'il puisse être référencé dans des conversations, des tickets et des e-mails.

Le format est :

```
{PREFIX}-{BOARD}-{SEQUENCE}
```

Par exemple : **`BRD-001-00001`**

| Segment | Signification |
|---|---|
| `PREFIX` | Le préfixe d'élément du tableau. Par défaut `BRD` et configurable par tableau (jusqu'à 10 caractères, par exemple `INST` pour un tableau d'installations). |
| `BOARD` | L'ID du tableau, complété avec des zéros pour atteindre trois chiffres. |
| `SEQUENCE` | La séquence de l'élément au sein du tableau, complétée avec des zéros pour atteindre cinq chiffres. |

Vous pouvez ouvrir n'importe quel élément directement par son numéro d'élément depuis la vue du tableau ou via l'API.

# Création et Gestion des Tableaux

## Création d'un Tableau

1. Naviguer vers **Gestion du Travail → Tableaux**

2. Cliquer sur **Créer un Tableau**
3. Compléter les détails du tableau :
    - **Nom du Tableau** (obligatoire)
    - **Description** (optionnelle)
    - **Modèle** : *Suivi des Tâches*, *Planification des Installations*, *Demandes de Fonctionnalités*, ou *Tableau Vide*
    - **Préfixe d'Élément** : utilisé dans les numéros d'éléments (par défaut `BRD`, jusqu'à 10 caractères)
    - **Visibilité** : *Personnel Seulement* ou *Public (Intégration API)*
4. Cliquer sur **Créer**

L'utilisateur qui crée un tableau devient son **propriétaire**.

# Paramètres du Tableau

Les propriétaires et administrateurs de tableau peuvent modifier un tableau pour changer son nom, sa description, son icône, sa couleur, le client lié, le préfixe d'élément et la visibilité. Les tableaux peuvent également être **archivés** pour les cacher de la liste active sans supprimer leur contenu, et **supprimés** (une suppression douce qui retire le tableau de la vue tout en conservant les enregistrements sous-jacents).

# Personnalisation des Statuts, Groupes et Étiquettes

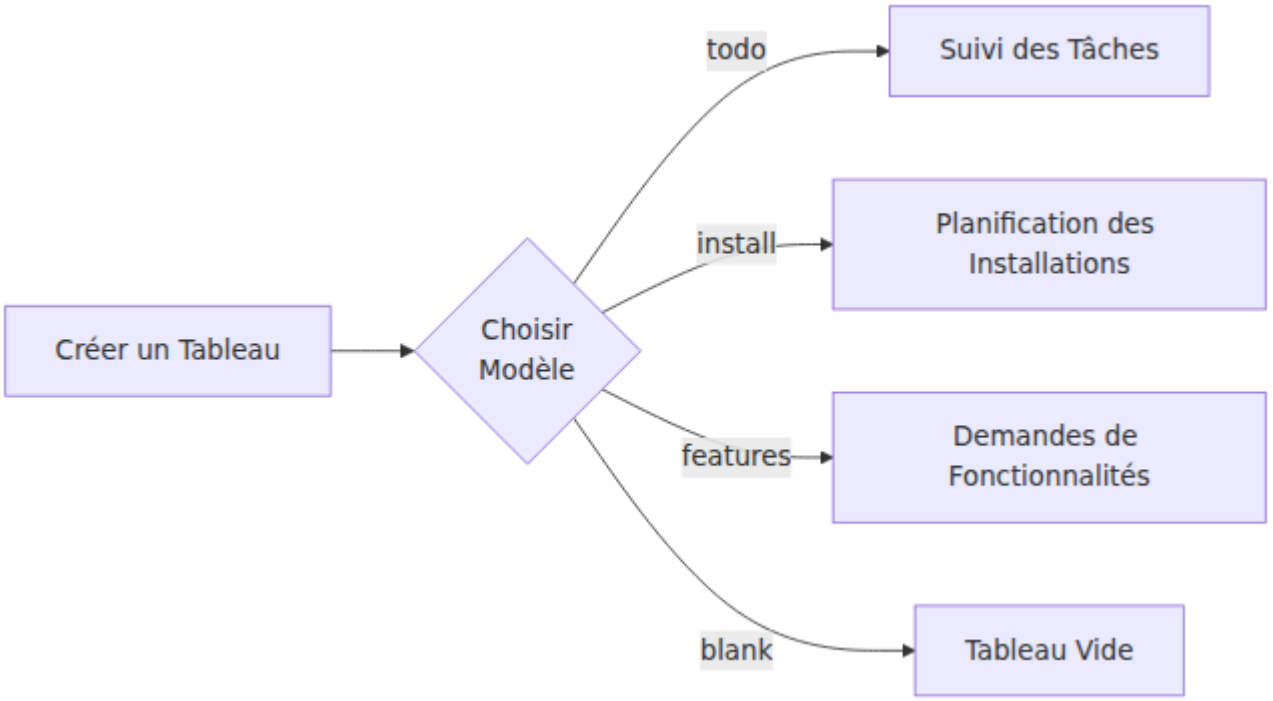
Depuis un tableau, vous pouvez :

- **Statuts** : ajouter, renommer, recolorer, réorganiser, définir le statut par défaut pour les nouveaux éléments, et marquer les statuts comme "faits". Si vous supprimez un statut, ses éléments sont laissés sans statut. Ils ne sont pas supprimés.
- **Groupes** : ajouter, renommer, recolorer, réorganiser et replier. Si vous supprimez un groupe, ses éléments sont laissés non groupés. Ils ne sont pas supprimés.
- **Étiquettes** : ajouter, renommer et recolorer. Si vous supprimez une étiquette, le système la retire de tous les éléments.

# Travailler avec les Éléments

## La Vue du Tableau

La vue du tableau montre chaque groupe comme une section avec un en-tête coloré affichant le nom du groupe, le nombre d'éléments et le progrès d'achèvement (combien d'éléments sont dans un statut "fait"). Au sein de chaque groupe, les éléments sont listés avec leur titre et numéro, assigné, statut, date d'échéance et priorité. Une ligne d'ajout rapide vous permet de créer des éléments en ligne.




*Les noms de statut ci-dessus sont illustratifs. Chaque tableau définit ses propres statuts.*

## Création d'un Élément

Les éléments peuvent être créés depuis la ligne d'ajout rapide dans n'importe quel groupe, ou via le bouton **Créer un Élément**. Un élément a :

| Champ | Description |
| --- | --- |
| **Titre** | Obligatoire. Résumé court du travail. |
| **Description** | Détail optionnel. |
| **Groupe** | La ligne à laquelle l'élément appartient. Par défaut, c'est le premier groupe si aucun n'est choisi. |
| **Statut** | L'étape du flux de travail. Par défaut, c'est le statut par défaut du tableau. |
| **Priorité** | *Faible*, *Moyenne*, *Élevée* ou *Urgente* (par défaut *Moyenne*). |
| **Date d'Échéance** | Date limite optionnelle. |
| **Assigné** | Le membre du personnel responsable. |
| **Demandeur** | Qui a demandé le travail : un utilisateur du personnel ou un contact client. |
| **Client / Service / Ticket** | Liens optionnels vers des enregistrements connexes. |
| **Étiquettes** | Un ou plusieurs tags de catégorie. |

## Déplacement des Éléments

Les éléments sont déplacés pour les réorganiser ou les faire progresser :

- **Changer de groupe** : déplacer l'élément vers une autre ligne/lane
- **Changer de statut** : faire progresser l'élément à travers le flux de travail
- **Réorganiser** : changer la position de l'élément au sein de son groupe

Chaque mouvement est enregistré dans l'historique de l'élément.

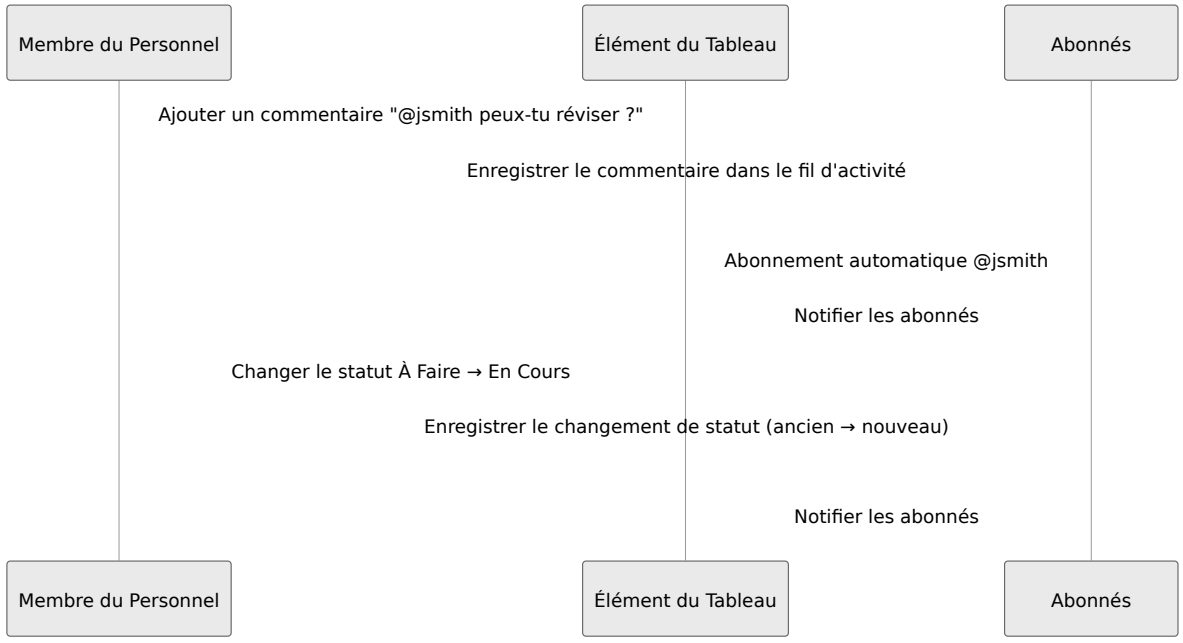
## Niveaux de Priorité

| Priorité | Utilisation typique |
|---|---|
| **Urgente** | Travail critique nécessitant une attention immédiate |
| **Élevée** | Travail important, à traiter bientôt |
| **Moyenne** | Travail standard (par défaut) |
| **Faible** | Travail mineur ou agréable à avoir |

# Collaboration

## Commentaires et Historique d'Activité

Chaque élément a un fil d'activité qui combine des **commentaires** manuels avec un **historique des modifications** automatique. Le système enregistre les changements de statut, d'assigné, de titre, de priorité, de date d'échéance et de groupe. Il capture les valeurs précédentes et nouvelles, de sorte que l'histoire complète d'un élément soit toujours visible.



## Abonnés et Notifications

Les abonnés reçoivent des notifications lorsqu'un élément change. Les abonnements sont gérés automatiquement et manuellement :

- **Automatique** : le système abonne automatiquement le créateur et l'assigné de l'élément. Lorsque vous réassignez un élément, le système abonne le nouvel assigné.
- **@Mentions** : mentionnez un membre du personnel avec `@username` dans un commentaire pour les abonner automatiquement.
- **Manuel** : abonnez directement le personnel ou les contacts clients à un élément, ou désabonnez-les.

Les abonnés ajoutés automatiquement peuvent toujours se désabonner eux-mêmes.

# Contrôle d'Accès et Rôles

Les tableaux sont privés pour leurs membres. Chaque membre se voit attribuer un **rôle** qui régit ce qu'il peut faire sur ce tableau.

| Rôle | Vue | Créer / Modifier des Éléments | Gérer les Paramètres (statuts, étiquettes) | Gérer les Membres | Supprir le Tabl |
|---|---|---|---|---|---|
| **Propriétaire** | ✓ | ✓ | ✓ | ✓ | ✓ |
| **Administrateur** | ✓ | ✓ | ✓ | ✗ | ✗ |
| **Membre** | ✓ | ✓ | ✗ | ✗ | ✗ |
| **Spectateur** | ✓ | ✗ | ✗ | ✗ | ✗ |

Les administrateurs système peuvent accéder et gérer tous les tableaux, quel que soit leur statut de membre. Les utilisateurs non administrateurs ne voient

que les tableaux dont ils sont membres.

> **Remarque :** Les actions sur les tableaux sont également régies par des permissions au niveau système (voir Utilisateurs et Rôles). Un utilisateur a besoin à la fois de la permission système pertinente *et* d'un rôle de tableau approprié. Un tableau doit toujours conserver au moins un propriétaire.

## Permissions Liées aux Tableaux

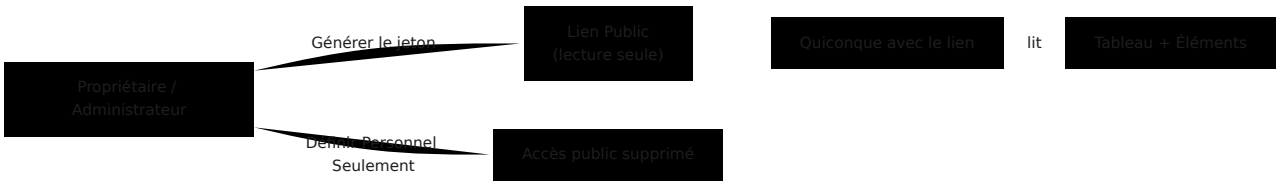
| Permission | Accorde la capacité de |
|---|---|
| `VIEW_BOARD` | Lister et voir les tableaux |
| `CREATE_BOARD` | Créer de nouveaux tableaux |
| `UPDATE_BOARD` | Modifier les détails du tableau et gérer les groupes |
| `DELETE_BOARD` | Supprimer un tableau |
| `MANAGE_BOARD_SETTINGS` | Créer, modifier et supprimer des statuts et des étiquettes |
| `MANAGE_BOARD_MEMBERS` | Ajouter, mettre à jour et supprimer des membres du tableau |
| `GENERATE_PUBLIC_TOKEN` | Générer un lien de partage public |
| `VIEW_BOARD_ITEM` | Voir les éléments, commentaires et abonnés |
| `CREATE_BOARD_ITEM` | Créer des éléments |
| `UPDATE_BOARD_ITEM` | Modifier des éléments, déplacer des éléments et ajouter des commentaires |
| `DELETE_BOARD_ITEM` | Supprimer des éléments |

# Visibilité et Partage Public

Chaque tableau a un paramètre de visibilité :

| Visibilité | Comportement |
|---|---|
| **Personnel Seulement** | Par défaut. Seuls les membres du tableau (et les administrateurs système) peuvent accéder au tableau. |
| **Public** | En plus de l'accès des membres, le tableau peut être lu via un lien public sécurisé sans connexion. |

Générer un **jeton d'accès public** produit un lien unique et imprévisible et change la visibilité du tableau en *Public*. Le lien public est **en lecture seule**. Utilisez-le pour intégrer une vue de tableau, par exemple sur une page de statut interne ou un tableau d'affichage. Régénérer le jeton invalide le lien précédent. Pour arrêter l'accès public, changez la visibilité du tableau en *Personnel Seulement*.




# Trouver des Éléments

## Filtrage et Recherche

La vue du tableau et les listes d'éléments prennent en charge le filtrage et la recherche pour se concentrer sur ce qui est important :

| Filtre | Description |
|---|---|
| **Recherche** | Correspond au titre de l'élément, à la description et au numéro d'élément |
| **Statut** | Un ou plusieurs statuts |
| **Assigné** | Un membre du personnel spécifique |
| **Priorité** | Un ou plusieurs niveaux de priorité |
| **Groupe** | Un groupe spécifique |
| **Client** | Éléments liés à un client spécifique |
| **Étiquettes** | Une ou plusieurs étiquettes |
| **En Retard** | Éléments dont la date d'échéance est passée |

### Tri

Les éléments peuvent être triés par position (l'ordre manuel du tableau), date de création, date d'échéance, priorité, titre ou date de dernière modification, dans un ordre croissant ou décroissant. Les longs tableaux sont paginés.

# Meilleures Pratiques

## Configuration du Tableau

1. **Choisissez le modèle le plus proche** : commencez par *Planification des Installations* ou *Demandes de Fonctionnalités* plutôt que *Vide* lorsqu'ils conviennent. Vous pouvez toujours personnaliser par la suite.
2. **Utilisez un préfixe d'élément significatif** : par exemple `INST` pour les installations, afin que les numéros d'éléments soient auto-descriptifs.

3. **Gardez les statuts simples** : un petit ensemble clair d'étapes de flux de travail est plus facile à gérer que de nombreuses étapes qui se chevauchent.
4. **Marquez les statuts "faits"** : afin que le progrès d'achèvement s'affiche correctement dans les en-têtes de groupe.

## Utilisation Quotidienne

1. **Assignez chaque élément** : une propriété claire empêche le travail de stagner.
2. **Définissez des dates d'échéance** : afin que le filtre *En Retard* soit significatif.
3. **Utilisez @mentions** : pour impliquer les bonnes personnes dans un élément sans gérer les abonnements manuellement.
4. **Liez des enregistrements connexes** : connectez des éléments au client, service ou ticket pertinent pour le contexte.
5. **Déplacez, ne recréez pas** : faites progresser les éléments à travers les statuts afin que l'historique des modifications raconte l'histoire complète.

# Résolution de Problèmes

**Problème : Un utilisateur ne peut pas voir un tableau**

- Confirmez que l'utilisateur a été ajouté en tant que membre du tableau.
- Confirmez que l'utilisateur a la permission `VIEW_BOARD` pour son rôle.
- Rappelez-vous que les non-administrateurs ne voient que les tableaux auxquels ils appartiennent.

**Problème : Un utilisateur ne peut pas changer de statuts ou d'étiquettes**

- Gérer les statuts et les étiquettes nécessite le rôle de tableau **Propriétaire** ou **Administrateur** ainsi que la permission `MANAGE_BOARD_SETTINGS`.

**Problème : Les éléments ont disparu après la suppression d'un statut ou d'un groupe**

- Les éléments ne sont pas supprimés. Supprimer un statut laisse ses éléments sans statut ; supprimer un groupe laisse ses éléments non groupés. Assignez un nouveau statut ou groupe pour les ramener à la vue.

**Problème : Le lien public ne montre pas le tableau**

- Confirmez que la visibilité du tableau est définie sur *Public*.
- Confirmez que le lien est actuel. Régénérer le jeton invalide le lien précédent.

**Problème : Impossible de supprimer un membre**

- Un tableau doit toujours avoir au moins un propriétaire. Assignez un autre propriétaire avant de supprimer le dernier.

# Comportements des Actions et des Recharges CGRateS

Ce guide explique comment fonctionnent les Actions CGRateS au sein d'OmniCRM, en se concentrant spécifiquement sur la gestion des soldes, les comportements de recharge et comment différents types d'actions affectent les produits additionnels.

# Aperçu

Dans le Système de Chargement en Ligne (CGRateS) d'OmniCRM, les **Actions** sont le mécanisme permettant d'ajouter, de modifier ou de supprimer des soldes sur les comptes clients.

Lorsque vous provisionnez un produit additionnel ou de recharge, vous exécutez en réalité une Action CGRateS qui manipule les soldes du compte. (Vous pouvez également adopter d'autres approches, comme ajouter des soldes aux comptes manuellement à partir des Playbooks. C'est juste un modèle courant que nous utilisons pour garder les choses claires.)

## Concepts Clés

**Action** - Un ensemble d'opérations à effectuer sur un compte (ajouter un solde, déduire un solde, enregistrer un CDR, etc.)

**Solde** - Une quantité d'une ressource (données, voix, SMS, monétaire) avec une date d'expiration et un poids

**ID de Solde** - Un identifiant unique pour un type de solde (par exemple, "Paquet de Données", "Minutes de Voix")

**Poids** - Priorité pour la consommation du solde (poids plus élevé consommé en premier)

**Date d'Expiration** - Quand le solde expire (date absolue ou relative comme "+5 jours")

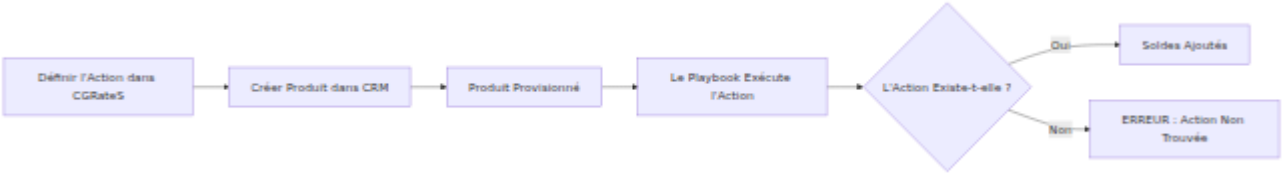
**Bloqueur** - Un drapeau de solde spécial qui bloque toute utilisation lorsque le solde atteint zéro, même si d'autres soldes existent (voir Bloqueurs de Solde)

# Critique : Les Actions Doivent Être Définies D'abord

**Avant de pouvoir exécuter une Action dans un playbook en utilisant `ExecuteAction`**, cette Action doit déjà être définie dans CGRateS. C'est une condition préalable critique souvent négligée.

## Quand les Actions Sont Définies

Les Actions sont généralement définies lors de **la configuration initiale du système** ou **de la configuration des produits**, PAS lors du provisionnement. Elles sont généralement créées via des scripts Python qui configurent à la fois le CRM et l'OCS simultanément.




## Comment les Actions Sont Liées aux Produits

Les Actions sont liées aux produits via une convention de nommage :

- **Action CGRateS :** `ActionsId = "Action_50gb-data-pack"`
- **Produit CRM :** `product_slug = "50gb-data-pack"`
- **Dans le Playbook :** `cgr_action_name = "Action_" + product_slug`

Lorsque le playbook s'exécute, il construit le nom de l'Action à partir du product_slug et appelle `ExecuteAction`. Si cette Action n'existe pas dans CGRateS, le provisionnement échoue.

## Où Définir les Actions

Les Actions doivent être définies dans vos scripts de configuration de produit, typiquement :

1. **Lors de la Configuration Initiale** - Lors de la première configuration de votre système
2. **Lors de la Création de Nouveaux Produits** - Définir l'Action avant de créer le produit
3. **Via des Scripts de Configuration** - Scripts Python qui configurent à la fois l'OCS et le CRM

**Exemple : Définir une Action avant de créer le produit**

```
import cgrateshttpapi

OCS_Obj = cgrateshttpapi.CGRateS("ocs.example.com", "2080")
tenant = "your_tenant"

# Étape 1 : Définir l'Action dans CGRateS D'ABORD
Action_50GB_Data_Pack = {
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_50gb-data-pack",
        "Tenant": tenant,
        "Actions": [
            {
                "Identifier": "*topup",
                "BalanceType": "*data",
                "Units": 50 * 1024 * 1024 * 1024,
                "ExpiryTime": "+720h",
                "Weight": 10
            }
        ]
    }]
}

result = OCS_Obj.SendData(Action_50GB_Data_Pack)
assert result['error'] is None or result['error'] == "EXISTS"

# Étape 2 : Maintenant créer le produit dans le CRM
# (product_slug = "50gb-data-pack" sera lié à Action_50gb-data-
pack)
```

**Que Se Passe-t-il Si l'Action N'existe Pas :**

```
# Dans le playbook
- name: Exécuter l'Action
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body:
      {
        "method": "APIerSv1.ExecuteAction",
        "params": [{
          "ActionsId": "Action_50gb-data-pack"
        }]
      }
  register: response

# Résultat si l'Action n'est pas définie :
# response.json.error = "SERVER_ERROR: Action not found"
# Le provisionnement échoue, le client ne reçoit aucun solde
```

Pour des détails complets sur la définition des Actions et leur liaison aux produits, voir Définir des Produits.

# Indépendance des Soldes

Par défaut, les produits additionnels créent des **soldes indépendants** qui fonctionnent séparément les uns des autres. Cela signifie :

- Vous pouvez avoir plusieurs addons actifs simultanément
- Chaque addon maintient son propre solde et sa propre expiration
- Les soldes sont consommés en fonction des règles de poids et d'expiration

### Exemple : Plusieurs Addons Indépendants

```
# Le client a ces soldes actifs :
Solde 1:
  ID: "Data_5GB_5days_uuid_abc123"
  Type: *data
  Valeur: 5368709120  # 5GB en octets
  Expiration: 2024-12-29
  Poids: 10

Solde 2:
  ID: "Data_10GB_30days_uuid_def456"
  Type: *data
  Valeur: 10737418240  # 10GB en octets
  Expiration: 2025-01-24
  Poids: 10

# Les deux soldes coexistent indépendamment
# Le système consomme en fonction du poids ; si poids égal,
l'ordre n'est pas garanti
```

Lorsque les soldes ont le même poids et correspondent à la même destination, CGRateS ne garantit pas l'ordre de consommation basé sur l'expiration - l'ordre dépend de la manière dont les soldes sont stockés et récupérés. **Utilisez des poids différents pour contrôler l'ordre de consommation**.

# Types d'Actions : *topup vs *topup_reset

Le type d'action détermine comment les nouveaux soldes interagissent avec les soldes existants du même ID.

### *topup - Comportement Additif

L'action `*topup` **ajoute aux soldes existants** avec le même ID de Solde et **prolonge le temps d'expiration**.

**Comportement :**

- Trouve le solde existant avec l'ID correspondant
- Ajoute la nouvelle valeur à la valeur existante
- Met à jour l'expiration à la nouvelle date d'expiration
- Préserve le solde existant (roulement)

**Exemple :**

```
# État initial :
Solde :
  ID: "Data_Package__5368709120"
  Valeur: 1073741824  # 1GB restant
  Expiration: 2024-12-24 (1 jour restant)

# Exécutez l'action *topup avec :
Action :
  Identifier: "*topup"
  Solde :
    ID: "Data_Package__5368709120"  # Même ID - déclenche le
roulement
    Valeur: 5368709120  # 5GB
    ExpiryTime: "+5d"

# Résultat après *topup :
Solde :
  ID: "Data_Package__5368709120"
  Valeur: 6442450944  # 6GB (1GB + 5GB roulés)
  Valeur_originale: 5368709120  # Affiche toujours les 5GB
d'origine
  Valeur_hr: "6 GB"
  Valeur_originale_hr: "5 GB"
  Restant_hr: "6 GB de 5 GB (1 GB roulé)"
  Expiration: 2024-12-29 (5 jours à partir de maintenant)
```

**Cas d'utilisation :**

- Récompenses de fidélité (ajouter des données supplémentaires au paquet existant)
- Crédits de compensation (ajouter au solde du client)
- Paquets de données roulés
- Extensions de période de grâce

# *topup_reset - Comportement de Réinitialisation

L'action `*topup_reset` **remplace les soldes existants** avec le même ID de Solde.

**Comportement :**

- Trouve le solde existant avec l'ID correspondant
- Jette l'ancienne valeur (pas de roulement)
- Définit le solde à la nouvelle valeur uniquement
- Met à jour l'expiration à la nouvelle date d'expiration

**Exemple :**

```
# État initial :
Solde :
  ID: "Data_Package__5368709120"
  Valeur: 1073741824  # 1GB restant
  Expiration: 2024-12-24 (1 jour restant)

# Exécutez l'action *topup_reset avec :
Action :
  Identifier: "*topup_reset"
  Solde :
    ID: "Data_Package__5368709120"  # Même ID - déclenche la
réinitialisation
    Valeur: 5368709120  # 5GB
    ExpiryTime: "+5d"

# Résultat après *topup_reset :
Solde :
  ID: "Data_Package__5368709120"
  Valeur: 5368709120  # 5GB (ancien 1GB jeté)
  Valeur_originale: 5368709120
  Valeur_hr: "5 GB"
  Restant_hr: "5 GB de 5 GB"
  PourcentageUtilisé: 0
  Expiration: 2024-12-29 (5 jours à partir de maintenant)
```

**Cas d'utilisation :**

- Paquets récurrents mensuels (réinitialiser à un montant complet chaque mois)
- Recharges de taille fixe (recevoir toujours un montant exact)
- Changements de plan (remplacer le solde de l'ancien plan par le nouveau plan)
- Prévenir les abus (ne peut pas empiler des addons illimités)

# Contrôle du Comportement des Soldes avec les IDs de Solde

L'**ID de Solde** est crucial pour déterminer si les soldes sont indépendants ou interagissent les uns avec les autres.

## Convention de Nommage des IDs de Solde et Vues Lisibles par l'Homme

OmniCRM utilise une convention de nommage spécifique pour les IDs de Solde qui encode à la fois le nom descriptif et la taille d'origine. Cela permet à l'API de générer automatiquement des champs lisibles par l'homme pour l'interface web.

**Modèle d'ID de Solde :**

```
{NomDescriptif}__{TailleOriginaleEnUnitésDeBase}
```

**Exemples d'IDs de Solde :**

```
# Solde de données : 100GB
"AU_Data_Domestic__107374182400"
# Se décompose en :
# - Partie descriptive : "AU_Data_Domestic" (CE QUE c'est -
type/destination)
# - Séparateur : "__" (double underscore)
# - Taille originale : "107374182400" (100GB en octets)
# - L'UI affiche : "AU Data Domestic - 100 GB"

# Solde de voix : 3000 minutes
"AU_Voice_Domestic__180000000000000"
# - Partie descriptive : "AU_Voice_Domestic" (PAS
"AU_Voice_Domestic_3000min")
# - Taille originale : "180000000000000" (3000 minutes en
nanosecondes)
# - L'UI affiche : "AU Voice Domestic - 3000 min"

# Solde de SMS : 3000 messages
"AU_SMS_Domestic__3000"
# - Partie descriptive : "AU_SMS_Domestic"
# - Taille originale : "3000" (compte)
# - L'UI affiche : "AU SMS Domestic - 3000 msgs"
```

**Important :** Ne pas inclure d'informations de taille dans la partie descriptive. C'est redondant puisque la taille est encodée après `__` et l'API la convertit automatiquement en format lisible par l'homme.

**Comment l'API Crée des Vues Lisibles par l'Homme :**

Lorsque l'API OmniCRM récupère les données de solde de CGRateS, elle analyse automatiquement l'ID de Solde et génère des champs `_hr` (lisibles par l'homme) :

```
{
  "BalanceMap": {
    "*data": [
      {
        "ID": "AU_Data_Domestic__107374182400",
        "Value": 53687091200,
        "ExpiryTime": "2025-01-25T23:59:59Z",
        "Weight": 1200,

        // Champs lisibles par l'homme générés automatiquement :
        "ID_hr": "AU Data Domestic",
        "OriginalValue": 107374182400,
        "OriginalValue_hr": "100 GB",
        "Value_hr": "50 GB",
        "Remaining_hr": "50 GB de 100 GB",
        "PercentUsed": 50,
        "ExpiryTime_hr": "25 Jan 2025 (22 jours)"
      }
    ]
  }
}
```

**Logique de Traitement de l'API :**

1. **Analyser l'ID de Solde :**

```
balance_id = "AU_Data_Domestic__107374182400"
parts = balance_id.split("__")

descriptive_name = parts[0]  # "AU_Data_Domestic"
original_size = int(parts[1]) if len(parts) > 1 else None  # 107374182400
```

2. **Générer un Nom Descriptif Lisible par l'Homme :**

```
# Remplacer les underscores par des espaces
id_hr = descriptive_name.replace("_", " ")  # "AU Data Domestic"
```

3. **Convertir la Taille Originale en Unités Humaines :**

```
# Pour les soldes de données (octets)
if balance_type == "*data":
    original_value_hr = convert_bytes_to_gb(original_size)  #
"100 GB"

# Pour les soldes de voix (nanosecondes)
elif balance_type == "*voice":
    original_value_hr = convert_ns_to_minutes(original_size)  #
"3000 min"

# Pour les soldes de SMS (compte)
elif balance_type == "*sms":
    original_value_hr = f"{original_size} msgs"  # "3000 msgs"
```

4. **Calculer le Pourcentage d'Utilisation :**

```
if original_size and original_size > 0:
    percent_used = ((original_size - current_value) /
original_size) * 100
```

5. **Format d'Affichage Restant :**

```
remaining_hr = f"{current_value_hr} de {original_value_hr}"
# "50 GB de 100 GB"
```

**Affichage de l'UI Web :**

Le frontend utilise ces champs `_hr` pour afficher des informations de solde conviviales :

```
// Au lieu d'afficher des valeurs brutes :
// ID: "AU_Data_Domestic__107374182400"
// Valeur: 53687091200

// Afficher lisiblement :
<BalanceCard>
  <Title>{balance.ID_hr}</Title>           {/* "AU Data Domestic" */}
  <Progress value={balance.PercentUsed}>   {/* 50% */}
    {balance.Remaining_hr}                 {/* "50 GB de 100 GB" */}
  </Progress>
  <Expiry>{balance.ExpiryTime_hr}</Expiry> {/* "25 Jan 2025 (22 jours)" */}
</BalanceCard>
```

**Pourquoi Cela Compte :**

1. **Suivi de la Taille Originale** - Même lorsque les soldes sont partiellement consommés, l'UI peut afficher "50 GB de 100 GB" au lieu de simplement "50 GB restant"

2. **Visualisation de la Progression** - Les calculs de pourcentage permettent des barres de progression précises

3. **Nommage Cohérent** - Les noms descriptifs extraits des IDs de Solde garantissent la cohérence entre le backend et le frontend

4. **Affichage de Rollover** - Lors de l'utilisation de `*topup` (roulement), si un client a 70 GB restants et recharge 100 GB :
   - L'ID de Solde reste : `"AU_Data_Domestic__107374182400"` (100 GB d'origine)
   - La valeur actuelle devient : 170 GB
   - L'UI affiche : "170 GB (70 GB roulés + 100 GB nouveaux)"

**Meilleure Pratique - Création d'IDs de Solde :**

Incluez toujours la taille originale après `__` pour un affichage correct de l'UI. Ne pas dupliquer les informations de taille dans le nom descriptif :

```
# Bon - nom descriptif + taille en unités de base
Action_Data_100GB = {
    "Actions": [
        {
            "BalanceId": f"AU_Data_Domestic__{100 * 1024 * 1024 *
1024}",
            "Units": 100 * 1024 * 1024 * 1024
        }
    ]
}

# Mauvais - taille redondante dans le nom descriptif
Action_Data_100GB = {
    "Actions": [
        {
            "BalanceId": f"AU_Data_Domestic_100GB__{100 * 1024 *
1024 * 1024}",  # Redondant !
            "Units": 100 * 1024 * 1024 * 1024
        }
    ]
}

# Mauvais - pas d'informations de taille (l'UI ne peut pas
calculer le pourcentage)
Action_Data_100GB = {
    "Actions": [
        {
            "BalanceId": "AU_Data_Domestic",  # Taille manquante
__size
            "Units": 100 * 1024 * 1024 * 1024
        }
    ]
}
```

**Cas Spécial - Soldes Monétaires :**

Les soldes monétaires n'incluent généralement pas de taille originale puisqu'ils peuvent être rechargés à n'importe quel montant :

```
# Solde monétaire sans encodage de taille
{
    "BalanceId": "PAYG_Monetary_Balance",
    "BalanceType": "*monetary",
    "Units": 5000  # 50,00 $
}

# L'UI affiche simplement le solde actuel sans pourcentage
# "Solde : 50,00 $"
```

## Stratégie 1 : IDs Uniques (Soldes Indépendants)

Utilisez des IDs uniques (par exemple, avec des UUID) pour créer des soldes complètement indépendants qui n'interagissent jamais.

**Exemple d'Implémentation :**

```
- name: Générer un identifiant de solde unique
  set_fact:
    uuid: "{{ 99999999 | random | to_uuid }}"
    balance_id: "Data_5days__5368709120_{{ uuid[0:8] }}"

- name: Ajouter un solde indépendant avec *topup
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.AddBalance",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Account": "{{ service_uuid }}",
          "BalanceType": "*data",
          "Balance": {
            "ID": "{{ balance_id }}",
            "Value": 5368709120,
            "ExpiryTime": "+5d",
            "Weight": 10
          }
        }]
      }
```

**Résultat :** Chaque addon crée un nouveau solde séparé même si le client achète le même addon plusieurs fois.

```yaml
# Le client achète "addon de données de 5 jours" trois fois :
Solde 1:
  ID: "Data_5days__5368709120_a1b2c3d4"
  Valeur: 5368709120
  Valeur_hr: "5 GB"
  Valeur_originale_hr: "5 GB"
  Restant_hr: "5 GB de 5 GB"
  Expiration: 2024-12-29

Solde 2:
  ID: "Data_5days__5368709120_e5f6g7h8"
  Valeur: 5368709120
  Valeur_hr: "5 GB"
  Restant_hr: "5 GB de 5 GB"
  Expiration: 2024-12-30

Solde 3:
  ID: "Data_5days__5368709120_i9j0k1l2"
  Valeur: 5368709120
  Valeur_hr: "5 GB"
  Restant_hr: "5 GB de 5 GB"
  Expiration: 2024-12-31

# Total disponible : 15GB répartis sur trois soldes séparés
# Chacun s'affiche individuellement dans l'UI comme "Données 5
jours - 5 GB de 5 GB"
```

## Stratégie 2 : IDs Partagés avec *topup (Rollover)

Utilisez le même ID de Solde avec l'action `*topup` pour permettre le roulement de solde et l'extension de l'expiration.

**Exemple d'Implémentation :**

```yaml
- name: Définir l'ID de solde fixe
  set_fact:
    balance_id: "Data_5days__5368709120"

- name: Ajouter un solde avec *topup (roulement)
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.AddBalance",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Account": "{{ service_uuid }}",
          "BalanceType": "*data",
          "Balance": {
            "ID": "Data_5days__5368709120",
            "Value": 5368709120,
            "ExpiryTime": "+5d",
            "Weight": 10
          }
        }]
      }
```

**Résultat :** Les achats ultérieurs s'ajoutent au solde existant et prolongent l'expiration.

```
# Jour 1 : Le client achète "addon de données de 5 jours" :
Solde :
  ID: "Data_5days__5368709120"
  Valeur: 5368709120
  Valeur_hr: "5 GB"
  Restant_hr: "5 GB de 5 GB"
  PourcentageUtilisé: 0
  Expiration: 2024-12-29

# Jour 3 : Le client utilise 1GB, puis achète le même addon à
nouveau :
Solde :
  ID: "Data_5days__5368709120"
  Valeur: 9663676416  # 4GB restants + 5GB nouveaux
  Valeur_hr: "9 GB"
  Valeur_originale_hr: "5 GB"
  Restant_hr: "9 GB (4 GB roulés + 5 GB nouveaux)"
  PourcentageUtilisé: -80  # Négatif indique un roulement
  Expiration: 2024-12-27 (nouveaux +5 jours à partir
d'aujourd'hui)
```

## Stratégie 3 : IDs Partagés avec *topup_reset (Montant Fixe)

Utilisez le même ID de Solde avec l'action `*topup_reset` pour toujours réinitialiser à un montant fixe.

**Exemple d'Implémentation :**

```
- name: Définir l'ID de solde fixe
  set_fact:
    balance_id: "Monthly_Plan__32212254720"

- name: Ajouter un solde avec *topup_reset (pas de roulement)
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.SetActions",
        "params": [{
          "ActionsId": "Action_Reset_Monthly_Plan",
          "Actions": [{
            "Identifier": "*topup_reset",
            "BalanceType": "*data",
            "Units": 32212254720,  # 30GB
            "ExpiryTime": "*monthly",
            "DestinationIds": "*any",
            "BalanceId": "Monthly_Plan__32212254720",
            "Weight": 10
          }]
        }]
      }
```

**Résultat :** Chaque mois, le solde se réinitialise à exactement 30GB, peu importe combien a été utilisé.

```
# Mois 1, Jour 1 :
Solde :
  ID: "Monthly_Plan__32212254720"
  Valeur: 32212254720
  Valeur_hr: "30 GB"
  Restant_hr: "30 GB de 30 GB"
  PourcentageUtilisé: 0
  Expiration: 2024-12-31

# Mois 1, Jour 25 : Le client a utilisé 28GB
Solde :
  ID: "Monthly_Plan__32212254720"
  Valeur: 2147483648
  Valeur_hr: "2 GB"
  Restant_hr: "2 GB de 30 GB"
  PourcentageUtilisé: 93
  Expiration: 2024-12-31

# Mois 2, Jour 1 : ActionPlan exécute *topup_reset
Solde :
  ID: "Monthly_Plan__32212254720"
  Valeur: 32212254720  # Réinitialisé à plein, 2GB non utilisés
perdus
  Valeur_hr: "30 GB"
  Restant_hr: "30 GB de 30 GB"
  PourcentageUtilisé: 0
  Expiration: 2025-01-31
```

# Bloqueurs de Solde

Les **Bloqueurs de Solde** sont une fonctionnalité puissante de CGRateS qui vous permet de **bloquer ou de limiter l'utilisation** même lorsque des soldes existent. Un solde bloqueur arrête la consommation lorsqu'il atteint zéro, empêchant toute utilisation, quel que soit le solde disponible.

## Comment Fonctionnent les Bloqueurs

Lorsqu'un solde a `Blocker: true` :

1. **Tant que le bloqueur a de la valeur** → L'utilisation est autorisée jusqu'au montant du bloqueur
2. **Lorsque le bloqueur atteint zéro** → Toute utilisation s'arrête, même si d'autres soldes existent
3. **Erreur retournée** → `INSUFFICIENT_CREDIT_BALANCE_BLOCKER` empêche la session

**Caractéristiques Clés :**

- Les soldes bloqueurs sont **vérifiés même lorsque la valeur est zéro** (contrairement aux soldes normaux qui sont ignorés)
- Lorsqu'un bloqueur est rencontré avec une utilisation restante demandée, CGRateS **arrête le traitement** et retourne une erreur
- Les bloqueurs fonctionnent avec tous les types de solde : `*voice`, `*data`, `*sms`, `*monetary`

# Cas d'Utilisation pour les Bloqueurs

### 1. Suspension de Compte

Bloquer toute utilisation lorsqu'un compte est suspendu (par exemple, échec de paiement) :

```
Action_Suspend_Account = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_suspend-account",
        "Overwrite": True,
        "Tenant": tenant,
        "Actions": [
            # Ajouter un bloqueur à valeur zéro pour empêcher
toute utilisation
            {
                "Identifier": "*topup",
                "BalanceId": "Suspension_Blocker",
                "BalanceType": "*monetary",
                "DestinationIDs": "*any",
                "Units": 0,  # Valeur zéro
                "BalanceWeight": 9999,  # Priorité la plus élevée
- vérifié en premier
                "Blocker": True,  # Bloquer toute utilisation
                "Weight": 10
            }
        ]
    }]
}

result = OCS_Obj.SendData(Action_Suspend_Account)
```

**Résultat :** Tous les appels/données/SMS bloqués, quel que soit le reste des soldes.

### 2. Limites de Dépenses

Limiter les dépenses maximales pour éviter les surprises sur la facture :

```
Action_Monthly_Plan_With_Cap = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_monthly-with-cap",
        "Overwrite": True,
        "Tenant": tenant,
        "Actions": [
            # 10 Go de données incluses
            {
                "Identifier": "*topup_reset",
                "BalanceId": f"Included_Data__{10 * 1024 * 1024 *
1024}",
                "BalanceType": "*data",
                "DestinationIDs": "Dest_PLMN_OnNet",
                "Units": 10 * 1024 * 1024 * 1024,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,  # Consommé en premier
                "Weight": 95
            },
            # Limite de dépassement de 50 $ (bloquant)
            {
                "Identifier": "*topup_reset",
                "BalanceId": "Overage_Cap",
                "BalanceType": "*monetary",
                "DestinationIDs": "*any",
                "Units": 5000,  # 50,00 $ maximum de dépassement
                "ExpiryTime": "*month",
                "BalanceWeight": 1000,  # Consommé après inclus
                "Blocker": True,  # Arrête lorsque 50 $ dépensés
                "Weight": 90
            }
        ]
    }]
}
```

**Flux :**

1. Le client utilise 10 Go inclus → GRATUIT (à partir de Included_Data_10GB)
2. Le client utilise 5 Go supplémentaires → Facturé à partir de Overage_Cap (tarifs PAYG)

3. Lorsque Overage_Cap atteint 0 $ → **Tous les usages bloqués** (limite de dépenses atteinte)

### 3. Essai Gratuit Limité dans le Temps

Fournir une utilisation gratuite limitée pour les comptes d'essai :

```
Action_Trial_Account = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_trial-100-minutes",
        "Overwrite": True,
        "Tenant": tenant,
        "Actions": [
            # 100 minutes gratuites (bloquant - s'arrête
lorsqu'épuisé)
            {
                "Identifier": "*topup",
                "BalanceId": f"Trial_Voice__{100 * 60 *
1000000000}",
                "BalanceType": "*voice",
                "DestinationIDs": "Dest_Domestic_All",
                "Units": 100 * 60 * 1000000000,  # 100 minutes
                "ExpiryTime": "+720h",  # 30 jours
                "BalanceWeight": 1200,
                "Blocker": True,  # Pas d'utilisation après 100
minutes
                "Weight": 10
            }
        ]
    }]
}
```

**Résultat :** Le client obtient exactement 100 minutes gratuites. Après cela, tous les appels sont bloqués (pas de frais automatiques).

### 4. Blocage Spécifique à la Destination

Bloquer des destinations spécifiques tout en permettant d'autres :

```
Action_Block_Premium_Numbers = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_block-premium",
        "Overwrite": True,
        "Tenant": tenant,
        "Actions": [
            # Utilisation régulière autorisée
            {
                "Identifier": "*topup_reset",
                "BalanceId": "Regular_Usage",
                "BalanceType": "*monetary",
                "DestinationIDs": "Dest_Domestic_All",
                "Units": 10000,  # 100 $
                "ExpiryTime": "*month",
                "BalanceWeight": 1000,
                "Weight": 20
            },
            # Bloquer les numéros premium (0900, etc.)
            {
                "Identifier": "*topup",
                "BalanceId": "Premium_Blocker",
                "BalanceType": "*monetary",
                "DestinationIDs": "Dest_Domestic_Premium",
                "Units": 0,  # Valeur nulle
                "BalanceWeight": 2000,  # Poids plus élevé -
vérifié en premier pour premium
                "Blocker": True,
                "Weight": 10
            }
        ]
    }]
}
```

**Flux :**

- Appels nationaux → Utilise le solde Regular_Usage
- Appels vers des numéros premium → Correspond à Premium_Blocker (poids 2000 > 1000) → **BLOQUÉ**

## Bloquant vs Soldes Désactivés

Ne pas confondre `Blocker` avec `Disabled` :

| Fonctionnalité | Bloquant | Désactivé |
|---|---|---|
| **But** | Arrêter l'utilisation lorsque le solde est épuisé | Suspendre temporairement un solde |
| **Lorsque la valeur > 0** | Le solde est utilisable normalement | Le solde est ignoré |
| **Lorsque la valeur = 0** | **Bloque toute utilisation ultérieure** | Le solde est ignoré (le solde suivant est essayé) |
| **Cas d'utilisation** | Limites de dépenses, plafonds, limites d'essai | Suspendre temporairement un solde spécifique |

```
# Bloquant : Permet 10 Go, puis bloque tout
{
    "BalanceId": f"Data_Cap__{10 * 1024 * 1024 * 1024}",
    "Units": 10 * 1024 * 1024 * 1024,
    "Blocker": True  # Arrête l'utilisation à 10 Go
}

# Désactivé : Ignore complètement ce solde (suspendu
temporairement)
{
    "BalanceId": f"Bonus_Data__{5 * 1024 * 1024 * 1024}",
    "Units": 5 * 1024 * 1024 * 1024,
    "Disabled": True  # Ce solde ne sera pas utilisé du tout
}
```

# Exemple Pratique : Plan Hybride avec Plafond de Sécurité

Combiner des soldes unitaires, un dépassement monétaire et un plafond bloquant :

```
Action_Safe_Hybrid_Plan = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_safe-hybrid-plan",
        "Overwrite": True,
        "Tenant": tenant,
        "Actions": [
            {
                "Identifier": "*reset_account",
                "Weight": 700
            },
            # 500 minutes nationales incluses
            {
                "Identifier": "*topup_reset",
                "BalanceId": f"Domestic_Voice__{500 * 60 *
1000000000}",
                "BalanceType": "*voice",
                "DestinationIDs": "Dest_Domestic_All",
                "Units": 500 * 60 * 1000000000,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 95
            },
            # 20 $ de dépassement autorisé
            {
                "Identifier": "*topup_reset",
                "BalanceId": "Overage_Allowance",
                "BalanceType": "*monetary",
                "DestinationIDs": "*any",
                "Units": 2000,  # 20,00 $
                "ExpiryTime": "*month",
                "BalanceWeight": 1000,
                "Weight": 90
            },
            # LIMITE ABSOLUE DE 50 $ (bloquant)
            {
                "Identifier": "*topup_reset",
                "BalanceId": "Hard_Spending_Cap",
                "BalanceType": "*monetary",
                "DestinationIDs": "*any",
                "Units": 5000,  # 50,00 $ maximum absolu
                "ExpiryTime": "*month",
```

```
                "BalanceWeight": 500,  # Inférieur au dépassement
- utilisé en dernier
                "Blocker": True,  # ARRÊTER lorsque le plafond est
atteint
                "Weight": 85
            }
        ]
    }]
}
```

**Parcours Client :**

1. **0-500 minutes :** Utilise Domestic_Voice_500min (GRATUIT)
2. **500-700 minutes :** Utilise Overage_Allowance à 0,10 $/min = 20 $ (200 min)
3. **700-1200 minutes :** Utilise Hard_Spending_Cap à 0,10 $/min = 50 $ (500 min)
4. **À 1200 minutes :** Hard_Spending_Cap épuisé → **TOUS LES USAGES BLOQUÉS**

Le client obtient un total de 1200 minutes, avec des dépenses plafonnées à un maximum de 50 $ de dépassement.

## Meilleures Pratiques avec les Bloquants

1. **Utiliser un Poids Élevé pour les Bloquants**

   ```
   "BalanceWeight": 9999  # Assurez-vous que le bloquant est
   vérifié en premier
   ```

2. **Bloquants à Valeur Nulle pour un Blocage Immédiat**

   ```
   "Units": 0,  # Bloquer immédiatement
   "Blocker": True
   ```

3. **Notifier les Clients Avant l'Épuisement du Bloquant**

- Utiliser ActionTriggers pour envoyer des notifications à 80 %, 90 %, 100 % de l'utilisation du bloquant
- Donner aux clients la possibilité d'augmenter le plafond avant que le blocage ne se produise

4. **Supprimer les Bloquants Lors de la Réactivation**

```
# Utiliser *remove_balance pour supprimer le bloquant
{
    "Identifier": "*remove_balance",
    "BalanceId": "Suspension_Blocker"
}
```

5. **Tester le Comportement du Bloquant**

   - Vérifier que le bloquant renvoie l'erreur `INSUFFICIENT_CREDIT_BALANCE_BLOCKER`
   - Confirmer que les CDR montrent un coût = -1.0 lorsqu'il est bloqué
   - Tester que d'autres soldes NE SONT PAS utilisés après l'épuisement du bloquant

# Dépannage des Bloquants

**Problème : Utilisation non bloquée malgré un bloquant à zéro**

**Causes Possibles :**

1. Poids du bloquant trop faible (d'autres soldes vérifiés en premier)
2. DestinationIDs ne correspondent pas à la destination d'utilisation
3. Champ Bloquant non défini sur `True`

**Solution :**

```
# Vérifier la configuration du bloquant
OCS_Obj.SendData({
    'method': 'ApierV2.GetAccount',
    'params': [{"Tenant": tenant, "Account": "service_uuid"}]
})

# Vérifier :
# - Bloquant : vrai
# - BalanceWeight est le plus élevé (par exemple, 9999)
# - DestinationIDs inclut la destination d'utilisation
# - Valeur = 0
```

**Problème : Utilisation bloquée de manière inattendue**

**Cause Possible :** Solde bloquant créé involontairement avec une faible valeur

**Solution :** Vérifier tous les soldes pour `Blocker: true` et vérifier que leurs valeurs sont appropriées pour votre cas d'utilisation.

# Règles de Consommation des Soldes

## Règle 1 : Priorité de Précision de Destination

Les soldes avec **une précision de destination plus élevée** (correspondance de destination plus spécifique) sont consommés en premier. Cela est déterminé par la longueur de la correspondance de préfixe.

```
# Le client appelle +44-20-1234-5678 (Londres, Royaume-Uni)

Solde 1 :
  DestinationIDs: "Dest_UK_London"  # Préfixe : "4420" (précision
: 4)
  Valeur: 100 minutes
  Poids: 10

Solde 2 :
  DestinationIDs: "Dest_UK_All"     # Préfixe : "44" (précision :
2)
  Valeur: 200 minutes
  Poids: 10

# Solde 1 consommé en premier (précision 4 > précision 2)
# Correspondance de destination plus spécifique l'emporte
```

**Cas d'utilisation :** Les soldes spécifiques à une ville ou à une région ont la priorité sur les soldes à l'échelle nationale.

## Règle 2 : Priorité de Poids (Même Précision)

Lorsque la précision de destination est égale, les soldes avec **un poids plus élevé** sont consommés en premier.

```
Solde 1 :
  ID: "Premium_Data"
  Valeur: 5 Go
  Poids: 20

Solde 2 :
  ID: "Standard_Data"
  Valeur: 10 Go
  Poids: 10

# Solde 1 consommé en premier (poids 20 > poids 10)
# Même si le Solde 2 a plus de données
```

**Cas d'utilisation :** Les soldes prioritaires (données bonus consommées avant les données régulières).

## Règle 3 : Le Plus Ancien en Premier

Lorsque le poids est identique, le plus ancien solde est utilisé en premier.

```
Solde 1 :
  ID: "Data_Package_A"
  Valeur: 5 Go
  Expiration: 2024-12-25
  Poids: 10

Solde 2 :
  ID: "Data_Package_B"
  Valeur: 10 Go
  Expiration: 2025-01-15
  Poids: 10
```

**Pour contrôler l'ordre de consommation, utilisez des poids différents :**

```
# Approche correcte : Utilisez le poids pour prioriser les soldes
bientôt expirés
Solde 1 :
  ID: "Data_Package_A"
  Valeur: 5 Go
  Expiration: 2024-12-25
  Poids: 11  # Poids plus élevé = consommé en premier

Solde 2 :
  ID: "Data_Package_B"
  Valeur: 10 Go
  Expiration: 2025-01-15
  Poids: 10  # Poids inférieur = consommé en second

# Solde 1 consommé en premier (poids 11 > poids 10)
```

**Meilleure Pratique :** Si vous souhaitez vous assurer que les soldes bientôt expirés sont consommés en premier, assignez-leur des poids plus élevés lors

de leur création.

# Exemples Pratiques

## Exemple 1 : Addon de Données Simple (Indépendant)

**Scénario :** Le client peut acheter un addon "5 Go 5 jours" plusieurs fois, chacun créant un solde séparé.

**Mise en œuvre :**

```
- name: Générer UUID pour un solde unique
  set_fact:
    uuid: "{{ 99999999 | random | to_uuid }}"

- name: Ajouter un solde indépendant de 5 Go
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.AddBalance",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Account": "{{ service_uuid }}",
          "BalanceType": "*data",
          "Categories": "*any",
          "Balance": {
            "ID": "Data_5GB_{{ uuid[0:8] }}",
            "Value": 5368709120,
            "ExpiryTime": "+120h",  # 5 jours
            "Weight": 10
          }
        }]
      }
```

**Expérience Client :**

- Achète un addon le 24 décembre → Obtient 5 Go expirant le 29 décembre
- Achète un addon le 25 décembre → Obtient 5 Go expirant le 30 décembre
- Les deux soldes coexistent ; l'ordre de consommation dépend des poids des soldes (les deux ont un poids de 10, donc l'ordre n'est pas garanti)

## Exemple 2 : Forfait de Données Rollover

**Scénario :** "Plan Mensuel de 50 Go" où les données non utilisées sont reportées lorsque le client recharge tôt.

**Mise en œuvre :**

```
- name: Ajouter un solde de données rollover
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.AddBalance",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Account": "{{ service_uuid }}",
          "BalanceType": "*data",
          "Balance": {
            "ID": "Rollover_Monthly_50GB",
            "Value": 53687091200,  # 50 Go
            "ExpiryTime": "+720h",  # 30 jours
            "Weight": 10
          }
        }]
      }
```

**Type d'Action :** Utilise le comportement par défaut `*topup` (report activé)

**Expérience Client :**

- Jour 1 : Obtient 50 Go expirant dans 30 jours
- Jour 20 : Utilise 30 Go, a 20 Go restants

- Jour 20 : Recharge à nouveau → Obtient 70 Go au total (20 Go + 50 Go), l'expiration est prolongée à +30 jours à partir du Jour 20

## Exemple 3 : Plan Mensuel Fixe (Pas de Rollover)

**Scénario :** Plan "Mensuel Illimité de 100 Go" qui se réinitialise exactement à 100 Go chaque mois, sans rollover.

**Mise en œuvre :**

```
- name: Créer une action de réinitialisation mensuelle
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.SetActions",
        "params": [{
          "ActionsId": "Action_Monthly_100GB_Reset",
          "Overwrite": true,
          "Actions": [{
            "Identifier": "*topup_reset",
            "BalanceType": "*data",
            "Units": 107374182400,  # 100 Go
            "ExpiryTime": "*monthly",
            "BalanceId": "Monthly_Plan__107374182400",
            "Weight": 10
          }]
        }]
      }

- name: Créer le Plan d'Action Mensuel
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.SetActionPlan",
        "params": [{
          "Id": "ActionPlan_Monthly_100GB",
          "ActionPlan": [{
            "ActionsId": "Action_Monthly_100GB_Reset",
            "Time": "*monthly",
            "Weight": 10
          }],
          "Overwrite": true,
          "ReloadScheduler": true
        }]
      }

- name: Assigner le Plan d'Action au compte
```

```
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV2.SetAccount",
        "params": [{
          "Account": "{{ service_uuid }}",
          "ActionPlanIds": ["ActionPlan_Monthly_100GB"],
          "ReloadScheduler": true
        }]
      }
```

**Expérience Client :**

- Mois 1 : Obtient 100 Go, utilise 95 Go, a 5 Go restants
- Mois 2 : Le solde se réinitialise à 100 Go (5 Go de données non utilisées perdus)
- Mois 2 : Utilise 20 Go, a 80 Go restants
- Mois 3 : Le solde se réinitialise à 100 Go (80 Go de données non utilisées perdus)

# Exemple 4 : Soldes Multi-Tiers avec Priorité de Poids

**Scénario :** Le client a "Données Bonus" (haute priorité) et "Données Régulières" (basse priorité). Les données bonus sont consommées en premier.

**Mise en œuvre :**

```
# Ajouter des données bonus avec un poids élevé
- name: Ajouter un solde de données bonus
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.AddBalance",
        "params": [{
          "Account": "{{ service_uuid }}",
          "BalanceType": "*data",
          "Balance": {
            "ID": "Bonus_Data",
            "Value": 5368709120,  # 5 Go
            "ExpiryTime": "+240h",  # 10 jours
            "Weight": 20  # Priorité plus élevée
          }
        }]
      }

# Ajouter des données régulières avec un poids normal
- name: Ajouter un solde de données régulières
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.AddBalance",
        "params": [{
          "Account": "{{ service_uuid }}",
          "BalanceType": "*data",
          "Balance": {
            "ID": "Regular_Data",
            "Value": 53687091200,  # 50 Go
            "ExpiryTime": "+720h",  # 30 jours
            "Weight": 10  # Priorité normale
          }
        }]
      }
```

**Expérience Client :**

- A 5 Go de données bonus (poids 20) + 50 Go de données régulières (poids 10)
- Consomme d'abord toutes les 5 Go de données bonus
- Puis consomme du pool de 50 Go de données régulières
- Les données régulières sont préservées plus longtemps

# Identifiants d'Action Communs

CGRateS prend en charge plusieurs identifiants d'action pour différentes opérations :

## Manipulation de Solde

***topup** - Ajouter au solde existant (rollover) ***topup_reset** - Réinitialiser le solde à une nouvelle valeur (pas de rollover) ***debit** - Soustraire du solde ***debit_reset** - Définir le solde à une valeur négative ***reset_account** - Supprimer tous les soldes

# Conception de Produits Addon

Lors de la conception de produits addon dans OmniCRM, considérez ces questions :

## Question 1 : Les soldes doivent-ils s'accumuler ?

**Oui (Indépendant)** → Utiliser des ID de Solde uniques (avec UUID) **Non (Remplacer)** → Utiliser un ID de Solde fixe avec `*topup_reset` **Oui (Rollover)** → Utiliser un ID de Solde fixe avec `*topup`

## Question 2 : Que se passe-t-il avec le solde non utilisé ?

**Rollover** → Utiliser l'action `*topup` **Perdu** → Utiliser l'action `*topup_reset` **Pools séparés** → Utiliser des ID de Solde uniques

## Question 3 : Comment les soldes doivent-ils être consommés ?

**Le plus ancien en premier** → Utiliser des poids différents (attribuer un poids plus élevé aux soldes plus anciens/bientôt expirés) **Premium en premier** → Poids différents (poids plus élevé = priorité plus élevée) **Ordre spécifique** → Utiliser des poids : 30 (premium), 20 (bonus), 10 (régulier) **Aléatoire/Pas de préférence** → Utiliser le même poids (ordre de consommation non garanti)

## Question 4 : Quelle est la stratégie d'expiration ?

**Durée fixe** → Utiliser l'expiration relative (`+720h` pour 30 jours) **Fin de mois** → Utiliser `*monthly` dans le Plan d'Action **N'expire jamais** → Utiliser `*unlimited` ou une durée très longue

# Structure d'Action CGRateS dans les Playbooks

Voici la structure complète pour créer une Action :

```
- name: Créer une Action CGRateS
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.SetActions",
        "params": [{
          "ActionsId": "Action_Name_Here",
          "Overwrite": true,  # Remplacer si existe
          "Actions": [
            {
              "Identifier": "*topup",  # ou *topup_reset
              "BalanceType": "*data",  # *data, *voice, *sms,
*monetary
              "Units": 5368709120,  # Montant à ajouter
              "ExpiryTime": "+120h",  # +Xh, *unlimited, *monthly
              "DestinationIds": "*any",  # Généralement *any
              "BalanceId": "Balance_Name",  # Unique ou partagé
              "Weight": 10,  # Priorité (plus élevé = consommé en
premier)
              "Blocker": false,  # Si vrai, empêche le solde
négatif
              "Disabled": false,  # Si vrai, le solde est
désactivé
              "SharedGroups": ""  # Groupe de solde partagé
(optionnel)
            },
            {
              "Identifier": "*cdrlog",  # Enregistrer cette action
comme CDR
              "BalanceType": "*generic",
              "ExtraParameters": "
{\"Category\":\"^activation\",\"Destination\":\"Addon Name\"}"
            }
          ]
        }]
      }
```

# Descriptions des Champs

**ActionsId** - Identifiant unique pour cet ensemble d'actions

**Identifier** - Le type d'opération (`*topup`, `*topup_reset`, `*cdrlog`, etc.)

**BalanceType** - Type de solde :

- `*data` - Soldes de données (octets)
- `*voice` - Soldes de voix (secondes)
- `*sms` - Soldes de SMS (nombre)
- `*monetary` - Soldes monétaires (unités de monnaie)
- `*generic` - Soldes génériques

**Units** - Montant à ajouter/soustraire (en unités de base : octets pour les données, secondes pour la voix)

**ExpiryTime** - Quand le solde expire :

- `+Xh` - Relatif (par exemple, `+720h` = 30 jours)
- `*unlimited` - N'expire jamais
- `*monthly` - Fin de mois
- `2024-12-31T23:59:59Z` - Horodatage absolu

**BalanceId** - Identifiant pour ce solde (ID partagé = interagit, ID unique = indépendant)

**Weight** - Priorité (numéro plus élevé = priorité plus élevée, consommé en premier)

**Blocker** - Si vrai, empêche le compte de devenir négatif

**Disabled** - Si vrai, le solde existe mais ne peut pas être utilisé

# Définir des Actions via Python (Configuration Initiale)

Les actions sont généralement définies lors de la configuration initiale du système à l'aide de scripts Python avec la bibliothèque `cgrateshttpapi`. Ces exemples montrent comment définir des Actions en utilisant `OCS_Obj.SendData()`.

## Prérequis

```
import cgrateshttpapi
import time

OCS_Obj = cgrateshttpapi.CGRateS("ocs.example.com", "2080")
tenant = "your_tenant_name"
tpid = str(tenant) + "_" + str(int(time.time()))
```

## Définir les Destinations

**Avant de créer des Actions**, vous devez définir des destinations qui spécifient OÙ les soldes peuvent être utilisés.

**Destinations** se déclinent en deux types :

- **Destinations Géographiques** - Préfixes de numéros pour les appels/SMS vers des lieux (par exemple, `Dest_International_UK`)
- **Destinations PLMN** - Codes de réseau pour les données provenant de lieux (par exemple, `Dest_PLMN_OnNet`, `Dest_PLMN_US_Verizon`)

**Règle Critique :**

- **Soldes Voix/SMS** → Utilisez des destinations géographiques (le numéro appelé)
- **Soldes de données** → Utilisez des destinations PLMN (le réseau auquel le client est connecté)

Pour une configuration complète des destinations incluant :

- Destinations géographiques (nationales, internationales, gratuites, premium)
- Destinations PLMN (sur réseau, réseaux en itinérance, zones)
- Règles de format PLMN et meilleures pratiques
- Dépannage des problèmes de destination

Voir : **Configuration des Destinations CGRateS**

## Calculs d'Unités

Comprendre les conversions d'unités est essentiel pour définir correctement les soldes :

```
# Soldes de données (en octets)
1_GB = 1 * 1024 * 1024 * 1024  # 1073741824 octets
100_GB = 100 * 1024 * 1024 * 1024

# Soldes de voix (en nanosecondes)
1_minute = 60 * 1000000000  # 60 milliards de nanosecondes
3000_minutes = 3000 * 60 * 1000000000

# Soldes SMS (en nombre)
3000_sms = 3000
```

## Exemple 1 : Plan Mensuel Multi-Soldes (Python)

Un plan mensuel complet avec des soldes de données, de voix, de SMS et d'itinérance qui se réinitialisent chaque mois.

```
Action_AU_Premium_Plan_1 = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_au-premium-plan-1",
        "Overwrite": True,
        "Tenant": str(tenant),
        "Actions": [
            # D'abord, réinitialisez le compte pour effacer les
anciens soldes
            {
                "Identifier": "*reset_account",
                "Weight": 700
            },
            # Ajouter un solde de données de 100 Go
            # IMPORTANT : Les soldes de données utilisent des
destinations PLMN (réseau auquel le client est connecté)
            # PAS des destinations géographiques. Utilisez VOTRE
PLMN sur réseau.
            {
                "Identifier": "*topup_reset",
                "BalanceId": "AU_Data_Domestic__" + str(100 * 1024
* 1024 * 1024),
                "BalanceType": "*data",
                "DestinationIDs": "Dest_PLMN_OnNet",  # Votre PLMN
sur réseau (mcc505.mnc057)
                "Units": 100 * 1024 * 1024 * 1024,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 90
            },
            # Ajouter un solde de voix de 3000 minutes
            {
                "Identifier": "*topup_reset",
                "BalanceId": "AU_Voice_Domestic__" + str(3000 * 60
* 1000000000),
                "BalanceType": "*voice",
                "DestinationIDs":
"Dest_AU_Mobile;Dest_AU_Fixed;Dest_AU_TollFree;",
                "Units": 3000 * 60 * 1000000000,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 89
```

```
            },
            # Ajouter un solde de 3000 SMS
            {
                "Identifier": "*topup_reset",
                "BalanceId": "AU_SMS_Domestic__" + str(3000),
                "BalanceType": "*sms",
                "DestinationIDs": "Dest_AU_Mobile;",
                "Units": 3000,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 88
            },
            # Ajouter 6 Go de données en itinérance
            {
                "Identifier": "*topup_reset",
                "BalanceId": "AU_Roaming_Data__" + str(6 * 1024 *
1024 * 1024),
                "BalanceType": "*data",
                "DestinationIDs": "Dest_Roaming_All",
                "Units": 6 * 1024 * 1024 * 1024,
                "ExpiryTime": "*month",
                "BalanceWeight": 1100,
                "Weight": 87
            },
            # Enregistrer cette action comme un CDR
            {
                "Identifier": "*cdrlog",
                "BalanceId": "",
                "BalanceUuid": "",
                "BalanceType": "*generic",
                "Directions": "*out",
                "Units": 0,
                "ExpiryTime": "",
                "Filter": "",
                "TimingTags": "",
                "DestinationIds": "",
                "RatingSubject": "",
                "Categories": "",
                "SharedGroups": "",
                "BalanceWeight": 0,
                "ExtraParameters": "
{\"Category\":\"^activation\",\"Destination\":\"AU Premium Plan
1\"}",
                "BalanceBlocker": "false",
```

```
                "BalanceDisabled": "false",
                "Weight": 80
            }
        ]
    }]
}

# Envoyer la définition de l'action à CGRateS
result = OCS_Obj.SendData(Action_AU_Premium_Plan_1)
assert result['error'] is None or result['error'] == "EXISTS"
print("Action créée : Action_au-premium-plan-1")
```

**Points Clés :**

- Utilise `*reset_account` pour effacer d'abord les anciens soldes
- Utilise `*topup_reset` pour des allocations mensuelles fixes (pas de report)
- BalanceWeight détermine l'ordre de consommation (domestique 1200 > itinérance 1100)
- Weight détermine l'ordre d'exécution au sein de l'Action
- Inclut `*cdrlog` pour le suivi des activations

## Exemple 2 : Addon de Données Simple (Python)

Un addon de données simple de 20 Go avec report désactivé.

```
Action_AU_Data_Addon_20GB = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_au-data-addon-20gb",
        "Overwrite": True,
        "Tenant": str(tenant),
        "Actions": [
            # Réinitialiser le compte d'abord
            {
                "Identifier": "*reset_account",
                "Weight": 700
            },
            # Ajouter 20 Go de données
            # Les soldes de données utilisent des destinations
PLMN (sur quel réseau le client est)
            {
                "Identifier": "*topup_reset",
                "BalanceId": "AU_Data_Domestic__" + str(20 * 1024
* 1024 * 1024),
                "BalanceType": "*data",
                "DestinationIDs": "Dest_PLMN_OnNet",  # Votre PLMN
sur réseau (mcc505.mnc057)
                "Units": 20 * 1024 * 1024 * 1024,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 90
            }
        ]
    }]
}

result = OCS_Obj.SendData(Action_AU_Data_Addon_20GB)
assert result['error'] is None or result['error'] == "EXISTS"
print("Action créée : Action_au-data-addon-20gb")
```

## Exemple 3 : Addon de Voix Internationale (Python)

Un addon pour des minutes d'appels internationaux.

```
Action_AU_International_Voice_100min = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_au-international-voice-100min",
        "Overwrite": True,
        "Tenant": str(tenant),
        "Actions": [
            {
                "Identifier": "*reset_account",
                "Weight": 700
            },
            # Ajouter 100 minutes pour des appels internationaux
            {
                "Identifier": "*topup_reset",
                "BalanceId": "AU_Voice_International__" + str(100
* 60 * 1000000000),
                "BalanceType": "*voice",
                "DestinationIDs": "Dest_International_All",
                "Units": 100 * 60 * 1000000000,
                "ExpiryTime": "*month",
                "BalanceWeight": 1000,
                "Weight": 90
            }
        ]
    }]
}

result = OCS_Obj.SendData(Action_AU_International_Voice_100min)
assert result['error'] is None or result['error'] == "EXISTS"
print("Action créée : Action_au-international-voice-100min")
```

**Remarque :** Les soldes Voix/SMS utilisent des destinations géographiques (numéro appelé), tandis que les données utilisent des destinations PLMN (réseau auquel le client est connecté). Voir Définir des Produits pour la configuration des destinations.

## Référence des Champs d'Action Python

**Champs de Définition de l'Action :**

- **ActionsId** (obligatoire) - Identifiant unique pour cet ensemble d'actions (doit correspondre à la convention product_slug dans le CRM)
- **Overwrite** - Si True, remplace l'action existante avec le même ID
- **Tenant** - Nom du locataire CGRateS
- **Actions** - Tableau des actions individuelles à exécuter

**Champs d'Action Individuels :**

- **Identifier** - Type d'action (`*topup`, `*topup_reset`, `*reset_account`, `*cdrlog`, etc.)
- **BalanceId** - Identifiant unique pour ce solde (doit correspondre entre les recharges pour que le report fonctionne)
- **BalanceType** - Type de solde (`*data`, `*voice`, `*sms`, `*monetary`)
- **DestinationIDs** - Contrôle OÙ le solde peut être utilisé :
  - Pour **voix/SMS** : Utilisez des destinations géographiques (par exemple, `"Dest_AU_Mobile"`, `"Dest_International_UK"`)
  - Pour **données** : Utilisez des destinations PLMN (par exemple, `"Dest_PLMN_OnNet"`, `"Dest_PLMN_US_Verizon"`)
- **Units** - Montant à ajouter (octets pour les données, nanosecondes pour la voix, nombre pour les SMS)
- **ExpiryTime** - Quand le solde expire (`*month`, `+720h`, `2024-12-31`, etc.)
- **BalanceWeight** - Priorité de consommation (plus élevé = consommé en premier)
- **Weight** - Ordre d'exécution au sein de l'ensemble d'actions (plus élevé = s'exécute en premier)
- **Blocker** (optionnel) - Drapeau booléen ; si `True`, bloque toute utilisation lorsque ce solde atteint zéro (voir Bloqueurs de Solde)
- **Disabled** (optionnel) - Drapeau booléen ; si `True`, ce solde est ignoré/omis lors de la consommation

# Exécution des Actions

Une fois qu'une Action est créée, exécutez-la sur un compte :

```
- name: Exécuter l'Action sur le Compte
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "APIerSv1.ExecuteAction",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Account": "{{ service_uuid }}",
          "ActionsId": "Action_Name_Here"
        }]
      }
```

Cela exécute toutes les opérations définies dans l'Action sur le compte spécifié.

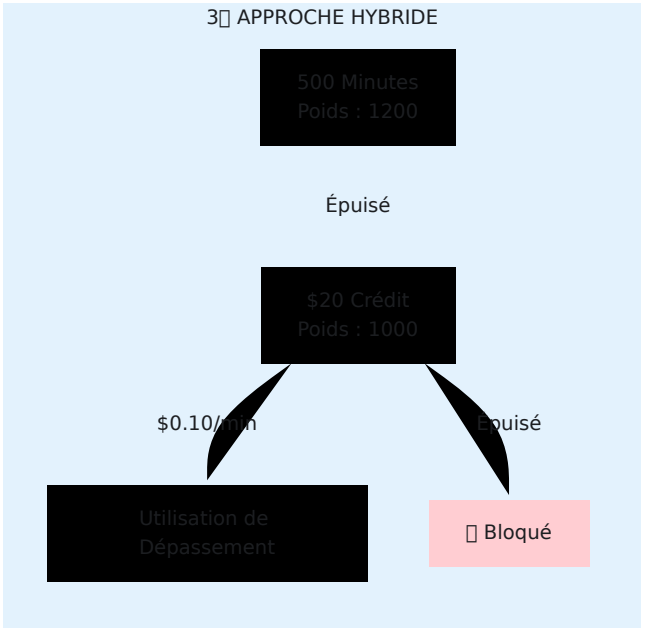
# Approches de Gestion des Soldes

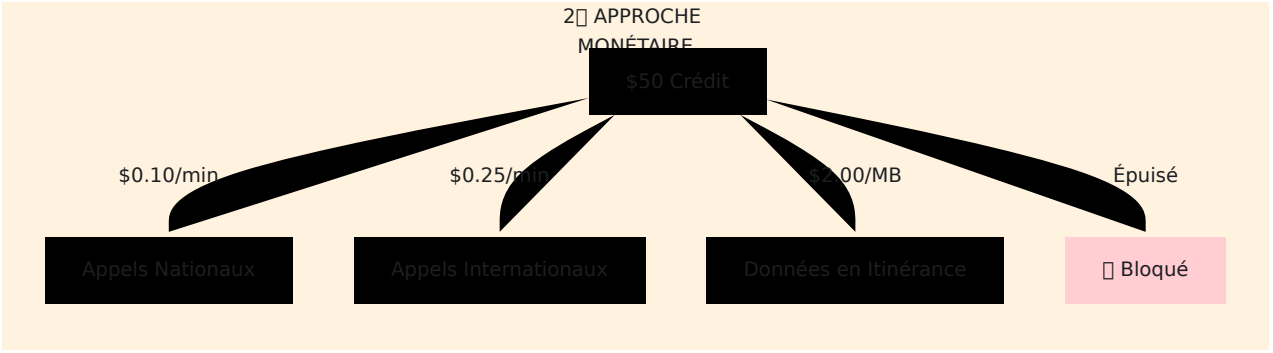
Il existe trois approches principales pour gérer les soldes dans CGRateS, chacune ayant des compromis différents pour l'expérience client, la prévisibilité de la facturation et la complexité opérationnelle.

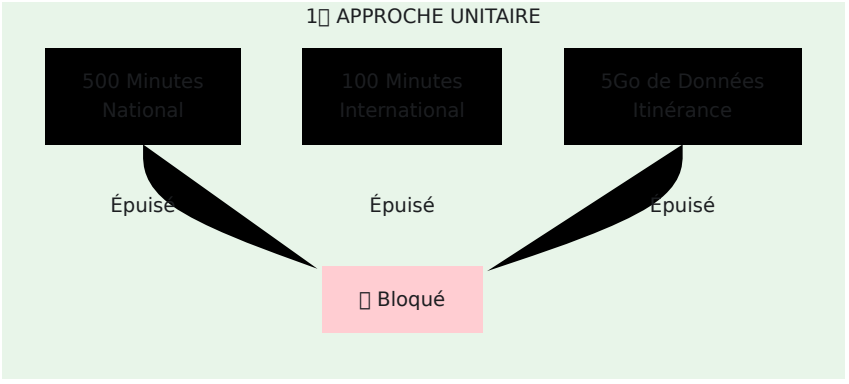
## Comparaison : Unitaire vs Monétaire vs Hybride

Ce tableau résume les compromis entre les trois approches de solde :

| Fonctionnalité | Unitaire | Monétaire (PAYG) | Hybride |
|---|---|---|---|
| **Prévisibilité Client** | □ Coût fixe mensuel | □ Coûts variables | ⚠□□□ Principalement prévisible |
| **Flexibilité de Destination** | □ Limité aux destinations incluses | □ Appel/utilisation partout | □ Inclus + partout |
| **Gestion des Dépassements** | □ Coupure nette | □ Utilisation automatique | □ Débordement automatique |
| **Risque de Choc de Facture** | □ Faible (plafonds durs) | □ Élevé (facturation illimitée) | ⚠ Modéré (débordement plafonné) |
| **Complexité de Configuration** | ⚠ Modérée (beaucoup de soldes) | □ Simple (un solde) | □ Complexe (les deux) |
| **Optimisation des Revenus** | ⚠ ARPU inférieur | □ ARPU plus élevé des utilisateurs intensifs | □ ARPU équilibré |
| **Satisfaction Client** | □ Élevée (pas de surprises) | □ Faible (choc de facture) | □ Élevée (le meilleur des deux) |
| **Meilleur Pour** | Utilisateurs prévisibles | Utilisateurs occasionnels | La plupart des clients |




# Approche 1 : Soldes Unitaires

**Concept :** Fournir des quantités spécifiques pour des destinations spécifiques. Chaque solde a un montant fixe (minutes, Go, nombre de SMS) lié à des destinations spécifiques. Lorsqu'il est épuisé, l'utilisation est bloquée à moins qu'il n'y ait un secours monétaire.

## Exemple : Plan National avec Plusieurs Soldes

```
Action_Domestic_Plan = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_domestic-plan",
        "Overwrite": True,
        "Tenant": tenant,
        "Actions": [
            {
                "Identifier": "*reset_account",
                "Weight": 700
            },
            # 500 minutes d'appels nationaux
            {
                "Identifier": "*topup_reset",
                "BalanceId": f"Domestic_Voice__{500 * 60 *
1000000000}",
                "BalanceType": "*voice",
                "DestinationIDs": "Dest_Domestic_All",  #
UNIQUEMENT national
                "Units": 500 * 60 * 1000000000,  # 500 minutes en
nanosecondes
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 90
            },
            # 1000 SMS nationaux
            {
                "Identifier": "*topup_reset",
                "BalanceId": "Domestic_SMS__1000",
                "BalanceType": "*sms",
                "DestinationIDs": "Dest_Domestic_All",
                "Units": 1000,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 89
            },
            # 10 Go de données nationales
            {
                "Identifier": "*topup_reset",
                "BalanceId": f"Domestic_Data__{10 * 1024 * 1024 *
1024}",
                "BalanceType": "*data",
```

```
                "DestinationIDs": "Dest_PLMN_OnNet",  # Votre PLMN
sur réseau
                "Units": 10 * 1024 * 1024 * 1024,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 88
            },
            {
                "Identifier": "*cdrlog",
                "BalanceType": "*generic",
                "ExtraParameters": "
{\"Category\":\"^activation\",\"Destination\":\"Plan National\"}",
                "Weight": 80
            }
        ]
    }]
}

result = OCS_Obj.SendData(Action_Domestic_Plan)
assert result['error'] is None or result['error'] == "EXISTS"
```

**Comment Cela Fonctionne :**

- Appels nationaux (1-555-1234) → Utilise le solde de 500 minutes
- Appels internationaux (44-20-xxx) → AUCUN solde disponible, bloqué OU utilise le solde monétaire si disponible
- SMS nationaux → Utilise le solde de 1000 SMS
- Utilisation des données domestiques → Utilise le solde de 10 Go
- Données en itinérance → AUCUN solde disponible (aurait besoin d'un solde de données en itinérance séparé)

**Avantages :**

- Coûts prévisibles pour les clients
- Pas de choc de facture
- Limites claires

**Inconvénients :**

- Moins flexible - ne peut pas utiliser le service en dehors des destinations incluses
- Nécessite plusieurs soldes pour différents cas d'utilisation
- Le client peut se sentir restreint

## Approche 2 : Monétaire (PAYG)

**Concept :** Fournir un crédit monétaire facturé à des tarifs spécifiques aux destinations. Un seul solde monétaire utilisé pour TOUS les types d'utilisation. CGRateS consulte le tarif pour chaque destination et déduit le coût du crédit.

**Remarque :** PAYG nécessite la définition de Profils de Tarifs pour chaque destination afin de définir le montant en dollars par unité. Voir Profils de Tarifs pour Soldes PAYG/Monétaires pour une configuration complète.

**Exemple : Crédit PAYG de 50 $**

```
Action_PAYG_Credit = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_payg-50-credit",
        "Overwrite": True,
        "Tenant": tenant,
        "Actions": [
            # Solde monétaire de 50 $
            {
                "Identifier": "*topup",
                "BalanceId": "PAYG_Monetary_Balance",
                "BalanceType": "*monetary",
                "DestinationIDs": "*any",  # Fonctionne pour
TOUTES les destinations
                "Units": 5000,  # 50,00 $ (en cents)
                "ExpiryTime": "+2160h",  # 90 jours
                "BalanceWeight": 1000,  # Plus bas que les soldes
unitaires
                "Weight": 90
            },
            {
                "Identifier": "*cdrlog",
                "BalanceType": "*generic",
                "ExtraParameters": "
{\"Category\":\"^activation\",\"Destination\":\"Crédit PAYG de 50
$\"}",
                "Weight": 80
            }
        ]
    }]
}

result = OCS_Obj.SendData(Action_PAYG_Credit)
```

**Comment fonctionne le PAYG :**

**Scénario 1 : Appel national, 10 minutes**

- Tarif : 0,10 $/min
- Charge : 10 × 0,10 $ = 1,00 $

- Reste : 49,00 $

**Scénario 2 : Appel vers le Royaume-Uni, 5 minutes**

- Tarif : 0,25 $/min + 0,05 $ de frais de connexion
- Charge : (5 × 0,25 $) + 0,05 $ = 1,30 $
- Reste : 47,70 $

**Scénario 3 : Itinérance sur Verizon, 100 Mo de données**

- Tarif : 2,00 $/Mo
- Charge : 100 × 2,00 $ = 200,00 $
- Résultat : Fonds insuffisants → Session bloquée à ~47 $ (~23 Mo)

**Avantages :**

- Un solde pour tout - très flexible
- Le client peut utiliser le service partout où les tarifs sont définis
- Configuration simple

**Inconvénients :**

- Coûts imprévisibles pour les clients
- Risque de choc de facture (surtout en itinérance)
- Peut être coûteux pour les utilisateurs intensifs

# Approche 3 : Hybride (Le Meilleur des Deux)

**Concept :** Combiner des soldes unitaires avec un secours monétaire. Utiliser des soldes unitaires à poids élevé pour l'utilisation incluse, avec un solde monétaire à faible poids comme débordement. Meilleure expérience client avec un coût de base prévisible et des dépassements flexibles.

**Exemple : Plan Hybride Flexible**

```
Action_Hybrid_Plan = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_hybrid-flex-plan",
        "Overwrite": True,
        "Tenant": tenant,
        "Actions": [
            {
                "Identifier": "*reset_account",
                "Weight": 700
            },

            # ============= SOLDES UNITAIRES (Inclus)
=============
            # 500 minutes de voix nationale
            {
                "Identifier": "*topup_reset",
                "BalanceId": f"Domestic_Voice__{500 * 60 *
1000000000}",
                "BalanceType": "*voice",
                "DestinationIDs": "Dest_Domestic_All",
                "Units": 500 * 60 * 1000000000,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,  # Consommé EN PREMIER pour
les appels nationaux
                "Weight": 95
            },
            # 100 minutes de voix internationale
            {
                "Identifier": "*topup_reset",
                "BalanceId": f"International_Voice__{100 * 60 *
1000000000}",
                "BalanceType": "*voice",
                "DestinationIDs": "Dest_International_All",
                "Units": 100 * 60 * 1000000000,
                "ExpiryTime": "*month",
                "BalanceWeight": 1150,  # Consommé EN PREMIER pour
international
                "Weight": 94
            },
            # 1000 SMS nationaux
            {
```

```
                "Identifier": "*topup_reset",
                "BalanceId": "Domestic_SMS__1000",
                "BalanceType": "*sms",
                "DestinationIDs": "Dest_Domestic_All",
                "Units": 1000,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 93
            },
            # 15 Go de données nationales
            {
                "Identifier": "*topup_reset",
                "BalanceId": f"Domestic_Data__{15 * 1024 * 1024 *
1024}",
                "BalanceType": "*data",
                "DestinationIDs": "Dest_PLMN_OnNet",
                "Units": 15 * 1024 * 1024 * 1024,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 92
            },
            # 2 Go de données en itinérance (Zone 1)
            {
                "Identifier": "*topup_reset",
                "BalanceId": f"Roaming_Zone1_Data__{2 * 1024 *
1024 * 1024}",
                "BalanceType": "*data",
                "DestinationIDs": "Dest_PLMN_Zone_NorthAmerica",
                "Units": 2 * 1024 * 1024 * 1024,
                "ExpiryTime": "*month",
                "BalanceWeight": 1100,
                "Weight": 91
            },

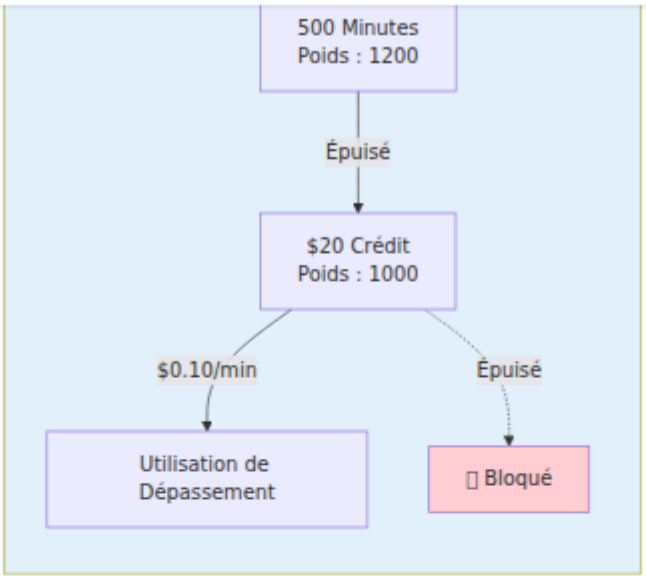
            # ============= SOLDE MONÉTAIRE (Dépassement/PAYG)
=============
            # 20 $ pour les dépassements
            {
                "Identifier": "*topup",
                "BalanceId": "PAYG_Overflow_Balance",
                "BalanceType": "*monetary",
                "DestinationIDs": "*any",
                "Units": 2000,  # 20,00 $
                "ExpiryTime": "*month",
```

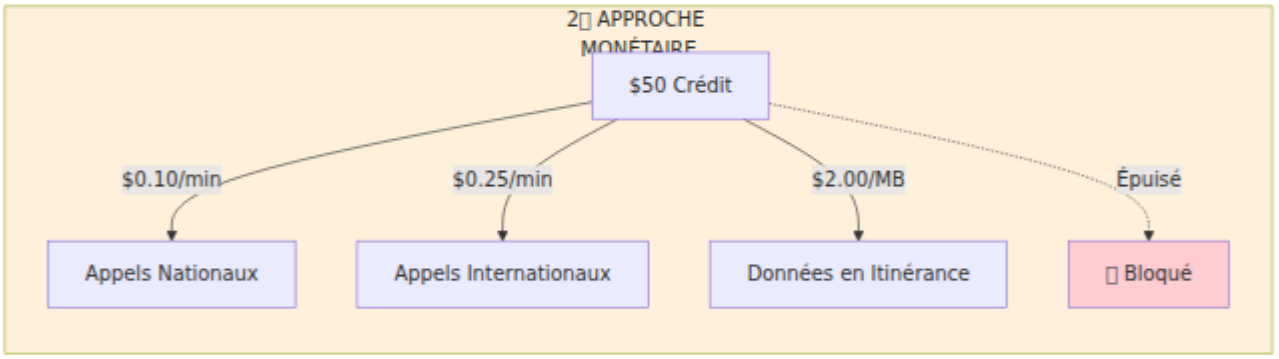
```
                "BalanceWeight": 1000,  # Consommé EN DERNIER
(secours)
                "Weight": 90
            },

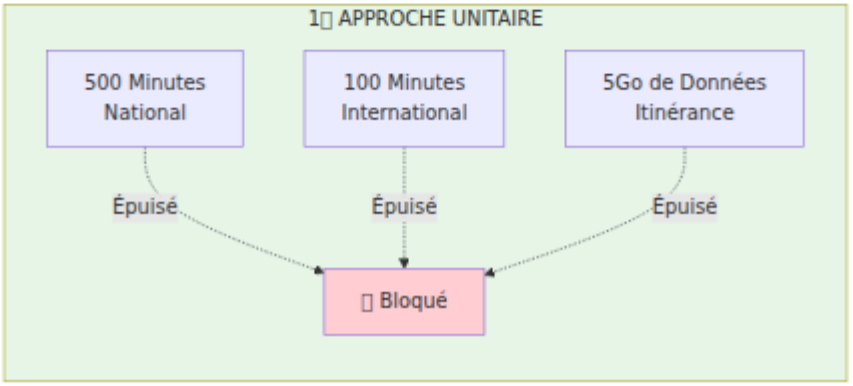
            {
                "Identifier": "*cdrlog",
                "BalanceType": "*generic",
                "ExtraParameters": "
{\"Category\":\"^activation\",\"Destination\":\"Plan Hybride
Flexible\"}",
                "Weight": 80
            }
        ]
    }]
}

result = OCS_Obj.SendData(Action_Hybrid_Plan)
```

**Flux de Consommation de Solde :**




**Scénarios Exemples :**

**Scénario 1 : 600 minutes domestiques utilisées**

1. Les 500 premières minutes → Utilise "Domestic_Voice_500min" (poids 1200) - GRATUIT
2. Les 100 prochaines minutes → "Domestic_Voice_500min" épuisé, passe à "PAYG_Overflow_Balance" (poids 1000)

3. Facturé : 100 min × 0,10 $ = 10,00 $ du solde monétaire
4. Restant : 10,00 $ monétaires

**Scénario 2 : 150 minutes internationales (vers le Royaume-Uni)**

1. Les 100 premières minutes → Utilise "International_Voice_100min" (poids 1150) - GRATUIT
2. Les 50 prochaines minutes → Passe à "PAYG_Overflow_Balance"
3. Facturé : (50 × 0,25 $) + 0,05 $ de connexion = 12,55 $
4. Restant : 7,45 $ monétaires (si départ avec 20 $)

**Avantages :**

- Coût de base prévisible
- Flexibilité pour les dépassements occasionnels
- Meilleure expérience client
- Pas de coupure brutale - le service continue

**Inconvénients :**

- Plus complexe à configurer
- Plus complexe à expliquer aux clients
- Nécessite une gestion soigneuse du poids des soldes

# Scénarios d'Utilisation dans le Monde Réel

Ces scénarios démontrent comment les concepts se réunissent en pratique. Chacun montre le flux complet depuis l'événement d'utilisation jusqu'à la déduction ou le blocage du solde.

## Scénario 1 : Client à Domicile Appelle le Royaume-Uni

**Le client a :** Plan Domestique (500 min domestiques, 1000 SMS, 10 Go de données)

**Action :** Appelle un numéro au Royaume-Uni (44-20-7946-0958) pendant 10 minutes

**Vérification du Solde :**

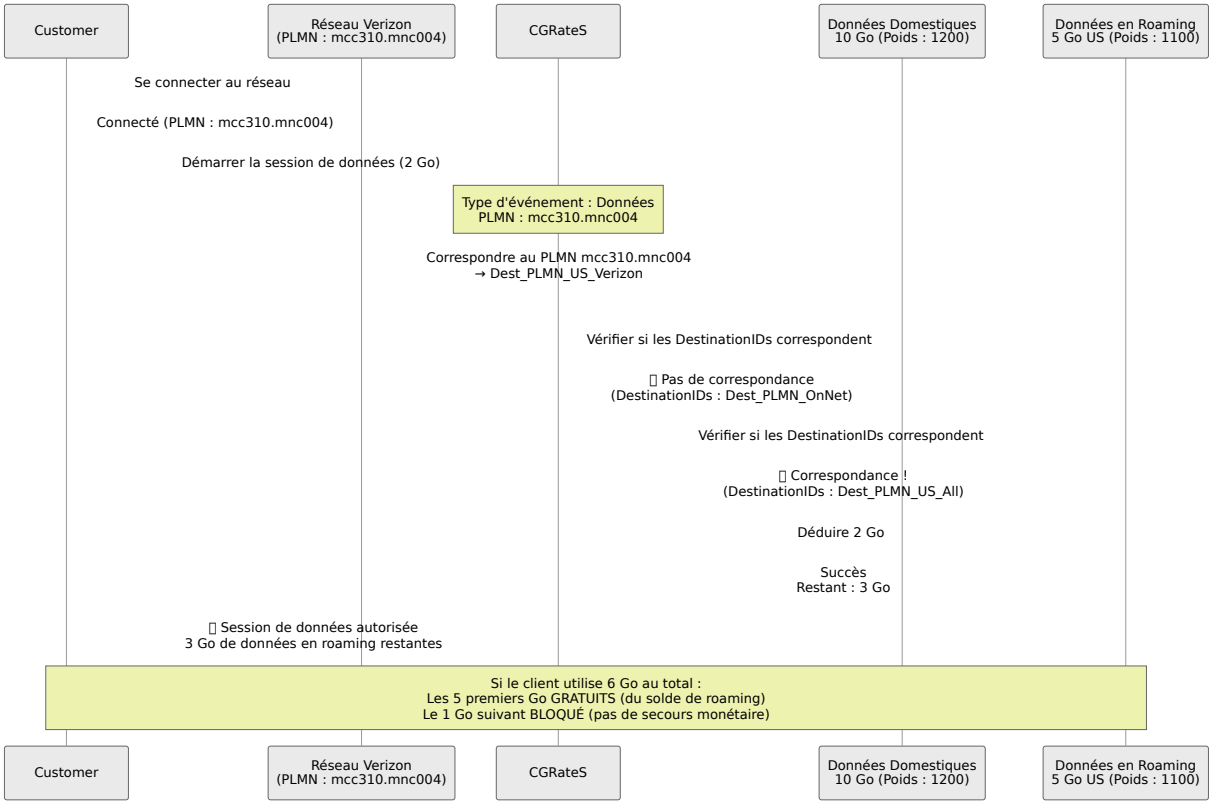
1. CGRateS reçoit l'appel au `44207946...`
2. Correspond à la destination : `Dest_International_UK`
3. Vérifie le solde avec `DestinationIDs: "Dest_International_UK"`
4. AUCUN solde correspondant trouvé
5. Vérifie le solde monétaire
6. AUCUN solde monétaire trouvé
7. **Résultat :** Appel BLOQUÉ (solde insuffisant)

**Solution :** Le client a besoin d'un Plan International OU d'un crédit PAYG pour passer des appels vers le Royaume-Uni

# Scénario 2 : Client en Roaming aux États-Unis Utilise des Données

**Le client a :** Plan Domestique + addon Roaming 5 Go aux États-Unis

**Action :** Roaming sur Verizon (PLMN mcc310.mnc004), utilise 2 Go de données




### Vérification du Solde :

1. CGRateS reçoit la session de données sur PLMN `mcc310.mnc004`
2. Correspond à la destination : `Dest_PLMN_US_Verizon`
3. Vérifie le solde de données avec `DestinationIDs: "Dest_PLMN_US_All"` (correspondance plus large)
4. Trouve : solde "Roaming_US_Data_5GB" (BalanceWeight : 1100)
5. Déduit 2 Go
6. **Restant :** 3 Go de données en roaming

### Que se passe-t-il s'ils utilisent 6 Go ?

1. Les 5 premiers Go → Utilise le solde de roaming (épuisé)
2. Le 1 Go suivant → Vérifie le solde monétaire
3. AUCUN solde monétaire → **Session bloquée** à 5 Go

# Scénario 3 : Client avec Plan Hybride Dépasse les Minutes Domestiques

**Le client a :** Plan Hybride (500 min domestiques + 20 $ de dépassement)

**Action :** Passe 600 minutes d'appels domestiques

**Vérification du Solde :**

1. Les 500 premières minutes :
    - Utilise "Domestic_Voice_500min" (BalanceWeight : 1200)
    - Pas de frais
2. "Domestic_Voice_500min" épuisé
3. Les 100 prochaines minutes :
    - Passe à "PAYG_Overflow_Balance" (BalanceWeight : 1000)
    - Tarif : 0,10 $/min
    - Frais : 100 × 0,10 $ = 10,00 $
    - Déduit de 20 $ de solde monétaire
    - **Restant :** 10,00 $

**Facturation :**

- Le client voit : 500 minutes incluses + 100 minutes de dépassement (10,00 $)

# Scénario 4 : Utilisateur PAYG Roaming Inattendu

**Le client a :** 50 $ de crédit PAYG

**Action :** Voyage aux États-Unis, roam sur Verizon, utilise 100 Mo de données (sans le savoir)

**Calcul des Frais :**

1. 100 Mo sur Verizon
2. Tarif : 2,00 $/Mo (des tarifs PAYG)
3. Frais totaux : 100 × 2,00 $ = 200,00 $
4. Disponible : 50,00 $
5. **Résultat :** Le client utilise ~25 Mo avant que le crédit ne soit épuisé, session bloquée

6. **Choc de facturation !** Le client a consommé tout le 50 $ de manière inattendue

**Meilleure approche :** Recommander un addon de roaming pour éviter le choc de facturation

# Meilleures Pratiques

## 1. Utiliser des IDs de Solde Descriptifs

De bons IDs de solde sont auto-documentés :

```
# Bon
"Data_5GB_5days_{{ uuid }}"
"Voice_100min_Monthly"
"Bonus_Data_Loyalty"

# Mauvais
"balance1"
"data"
"temp"
```

## 2. Documenter Votre Stratégie de Poids

Définir un schéma de poids cohérent à travers tous les produits :

```
# Schéma de poids :
# 30 = Soldes Premium/Promotionnels
# 20 = Soldes Bonus/Fidélité
# 10 = Soldes Réguliers/Achétés
# 5 = Soldes de Sauvegarde/Secours
```

## 3. Inclure la Journalisation CDR

Toujours enregistrer les ajouts de solde pour les pistes d'audit :

```
{
  "Identifier": "*cdrlog",
  "BalanceType": "*generic",
  "ExtraParameters": "
{\"Category\":\"^activation\",\"Destination\":\"{{ package_name
}}\"}"
}
```

## 4. Utiliser des Plans d'Action pour les Opérations Récurrentes

Pour les réinitialisations mensuelles, ne pas exécuter l'Action manuellement. Utiliser des Plans d'Action :

```
# Créer un Plan d'Action qui s'exécute mensuellement
- name: Créer un Plan d'Action mensuel
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.SetActionPlan",
        "params": [{
          "Id": "ActionPlan_Monthly_Reset",
          "ActionPlan": [{
            "ActionsId": "Action_Monthly_Reset",
            "Time": "*monthly",
            "Weight": 10
          }],
          "ReloadScheduler": true
        }]
      }
```

## 5. Réinitialiser les ActionTriggers Après un Rechargement

Après l'exécution d'une action, réinitialiser les déclencheurs pour éviter les notifications en double :

```
- name: Réinitialiser les ActionTriggers
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "APIerSv1.ResetAccountActionTriggers",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Account": "{{ service_uuid }}",
          "Executed": false
        }]
      }
```

## 6. Définir des Poids de Solde Appropriés

Utiliser une stratégie de poids cohérente à travers tous les produits :

```
# Priorité de consommation (de la plus élevée à la plus basse)
BalanceWeight: 1200  # Inclus domestique (utilisé en premier à domicile)
BalanceWeight: 1150  # Inclus international (utilisé en premier pour intl)
BalanceWeight: 1100  # Inclus roaming (utilisé en premier lors du roaming)
BalanceWeight: 1000  # Secours monétaire (utilisé en dernier)
```

Cela garantit un ordre de consommation prévisible : soldes domestiques d'abord, puis internationaux, puis en roaming, et enfin monétaires PAYG comme secours.

## 7. Contrôler l'Utilisation du Solde avec les DestinationIDs

Utiliser toujours des DestinationIDs spécifiques pour éviter une utilisation non intentionnelle :

```
# Bon - contrôle de destination explicite
{
    "BalanceType": "*voice",
    "DestinationIDs": "Dest_International_UK",  # UNIQUEMENT UK
    "Units": 200 * 60 * 1000000000
}

# Mauvais - utilisation non intentionnelle
{
    "BalanceType": "*voice",
    "DestinationIDs": "*any",  # Pourrait être utilisé pour des numéros premium !
    "Units": 200 * 60 * 1000000000
}
```

## 8. Grouper les Destinations de Manière Logique

Créer des groupes de destinations qui ont du sens pour votre offre de produit :

```
# Bon - regroupement logique
"Dest_International_Europe" = Tous les pays de l'UE (pool partagé de 500 min)

# Mauvais - trop granulaire
"Dest_France", "Dest_Allemagne", "Dest_Italie" (petits pools séparés)
```

Voir Destinations CGRateS pour des exemples de regroupement de destinations.

## 9. Inclure un Secours Monétaire dans les Plans

Pour la meilleure expérience client, inclure un petit solde monétaire comme dépassement :

```
# Recommandé : Toujours inclure un petit solde monétaire
{
    "BalanceType": "*monetary",
    "Units": 1000,  # Protection de dépassement de 10 $
    "BalanceWeight": 1000  # Plus bas que les soldes unitaires
}
```

Cela empêche les coupures de service brutales et offre de la flexibilité pour les dépassements occasionnels.

## 10. Utiliser des Incréments de Tarifs de Manière Stratégique

Choisir des incréments de tarif appropriés pour différents types de services :

```
# Domestique : Amical pour le client par seconde
"RateIncrement": "1s"

# International : Blocs standard de 6 secondes
"RateIncrement": "6s"

# Premium : Blocs de 30 secondes pour décourager les abus
"RateIncrement": "30s"

# Données : Par Ko pour précision
"RateIncrement": "1024"
```

## 11. Définir des Temps d'Expiration Raisonnables

Faire correspondre les temps d'expiration aux types de produits :

```
# Plans d'abonnement mensuels
"ExpiryTime": "*month"

# Recharges/addons ponctuels
"ExpiryTime": "+720h"  # 30 jours

# Crédit PAYG (validité plus longue)
"ExpiryTime": "+2160h"  # 90 jours

# Addons de roaming (basés sur le voyage)
"ExpiryTime": "+360h"  # 15 jours
```

# Dépannage

## Problème : Solde Non Ajouté

**Symptômes :** L'action s'exécute avec succès mais le solde n'apparaît pas

**Causes Possibles :**

- UUID de compte incorrect
- Mismatch de BalanceType
- Expiration déjà passée

**Solution :** Vérifiez que le compte existe et vérifiez le temps d'expiration du solde

## Problème : Mauvais Solde Consommé

**Symptômes :** Le système consomme à partir d'un solde inattendu

**Causes Possibles :**

- Configuration de poids incorrecte
- Plusieurs soldes avec le même ID

**Solution :** Vérifiez les valeurs de poids et assurez-vous que les IDs de solde sont uniques si l'indépendance est souhaitée

# Problème : Rollover Non Fonctionnel

**Symptômes :** Utilisation de `*topup` mais ancien solde ne roule pas

**Causes Possibles :**

- Utilisation de différents IDs de solde (besoin du même ID)
- Action utilisant `*topup_reset` au lieu de `*topup`

**Solution :** Vérifiez la cohérence des IDs de solde et le type d'action

# Problème : Solde Expire Immédiatement

**Symptômes :** Solde ajouté mais affiché comme expiré

**Causes Possibles :**

- ExpiryTime dans le passé
- Utilisation d'un timestamp absolu au lieu d'un relatif

**Solution :** Utilisez une expiration relative (`+Xh`) au lieu de timestamps absolus

# Problème : Solde Non Consommé

**Symptôme :** Le client a un solde mais est toujours bloqué ou facturé PAYG

**Causes Possibles :**

1. Mismatch de DestinationIDs - Les destinations de solde ne correspondent pas à la destination d'utilisation
2. Solde expiré
3. Mauvais type de solde pour l'utilisation

**Étapes de Débogage :**

```
# Vérifiez les soldes de compte
OCS_Obj.SendData({
    'method': 'ApierV2.GetAccount',
    'params': [{"Tenant": tenant, "Account": "service_uuid"}]
})

# Vérifiez ce qui est retourné :
# - Le solde existe-t-il ?
# - Les DestinationIDs sont-ils corrects ?
# - ExpiryTime dans le futur ?
# - Units > 0 ?
```

**Solution :** Vérifiez que la destination correspond et que le solde est actif. Voir Scénario 1 : Client à Domicile Appelle le Royaume-Uni pour un exemple.

## Problème : Mauvais Tarif Appliqué

**Symptôme :** Client facturé d'un montant incorrect pour l'utilisation PAYG

**Causes Possibles :**

1. Chevauchement de destination (mauvaise priorité dans le RatingPlan)
2. Poids de liaison du plan de tarification incorrect
3. Définition de tarif avec des valeurs incorrectes

**Solution :**

- Assurez-vous que les destinations spécifiques ont un poids plus élevé dans les liaisons du RatingPlan
- Vérifiez que la correspondance du préfixe le plus long fonctionne correctement
- Vérifiez que les valeurs et les incréments de tarif sont corrects

Voir Profils de Tarification pour Soldes PAYG/Monétaires pour une configuration appropriée.

# Problème : Solde de Roaming Non Utilisé

**Symptôme :** Client en roaming, a un solde de roaming, mais facturé PAYG ou bloqué

**Causes Possibles :**

1. Mismatch de destination PLMN - Le solde n'inclut pas le PLMN visité
2. Client connecté à un réseau incorrect/inattendu
3. DestinationIDs ne correspondent pas au PLMN réel

**Solution :**

```
# Vérifiez sur quel PLMN le client se trouve (à partir des CDR ou des événements d'utilisation)
# Vérifiez que le solde inclut ce PLMN :
"DestinationIDs": "Dest_PLMN_US_All"  # Devrait inclure mcc310.mnc004
```

Voir Destinations CGRateS pour la configuration de destination PLMN et Scénario 2 : Client en Roaming aux États-Unis Utilise des Données pour un exemple fonctionnel.

# Problème : Plan Hybride Ne Passe Pas au Monétaire

**Symptôme :** Solde unitaire épuisé mais le solde mon��taire n'est pas utilisé

**Causes Possibles :**

1. Le solde monétaire a une restriction de DestinationIDs (devrait être `*any`)
2. Aucun tarif PAYG défini pour cette destination
3. Problème de poids de solde - le poids du solde monétaire n'est pas inférieur à celui du solde unitaire

**Solution :**

- Assurez-vous que le solde monétaire a `DestinationIDs: "*any"`

- Vérifiez que les tarifs PAYG sont définis pour toutes les destinations que le client pourrait utiliser
- Vérifiez les poids de solde : monétaire devrait être le plus bas (par exemple, 1000)

Voir Approche 3 : Hybride pour une configuration hybride appropriée.

## Problème : Frais Internationaux Inattendus

**Symptôme :** Client a appelé un numéro domestique, facturé à des tarifs internationaux

**Causes Possibles :**

1. Numéro en réalité international (par exemple, numéro canadien +1 traité comme international)
2. Chevauchement de préfixe dans les définitions de destination
3. Client a mal composé (ajouté des chiffres supplémentaires)

**Solution :**

- Vérifiez les CDR pour le numéro réellement composé
- Vérifiez que les définitions de préfixe de destination ne se chevauchent pas incorrectement
- Envisagez une destination séparée pour le Canada si les pays NANP nécessitent des tarifs différents

Voir Destinations CGRateS pour la configuration de destination géographique.

# Documentation Connexe

## Concepts de Base

- Définir des Produits - Flux de travail complet pour créer des produits, définir des Actions CGRateS et les lier ensemble
- Destinations CGRateS - Comment définir des destinations géographiques et PLMN pour les services vocaux, SMS et de données

# Mise en Œuvre & Opérations

- Facturation et Paiements à partir des Playbooks - Flux de paiement à deux phases, calculs au prorata et facturation des clients
- Guide des Playbooks Ansible - Structure des playbooks, flux de provisionnement et déprovisionnement avec des blocs de secours
- Aperçu du Système de Provisionnement - Comment fonctionne le provisionnement dans OmniCRM

# Configuration des Destinations CGRateS

Ce guide explique comment configurer les destinations dans CGRateS pour OmniCRM. Les destinations définissent OÙ les soldes peuvent être utilisés, que ce soit pour appeler des numéros spécifiques (géographiques) ou pour utiliser des données/voix sur des réseaux spécifiques (PLMN).

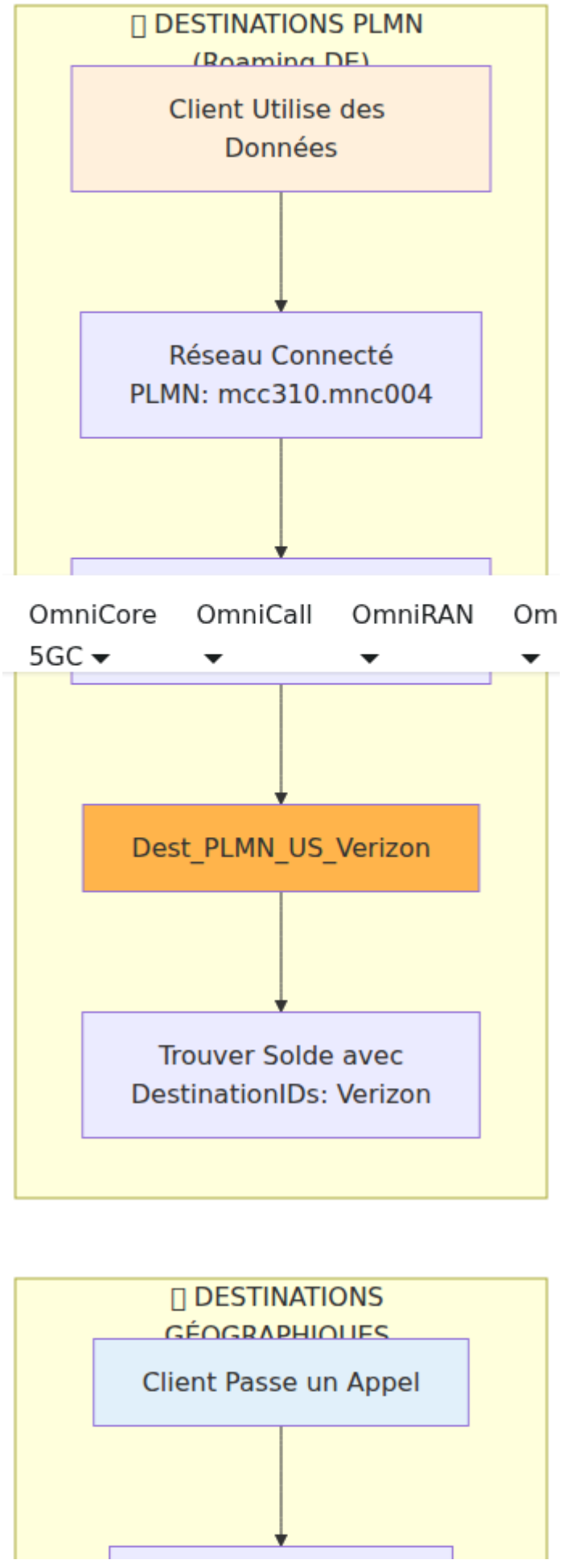
## Aperçu

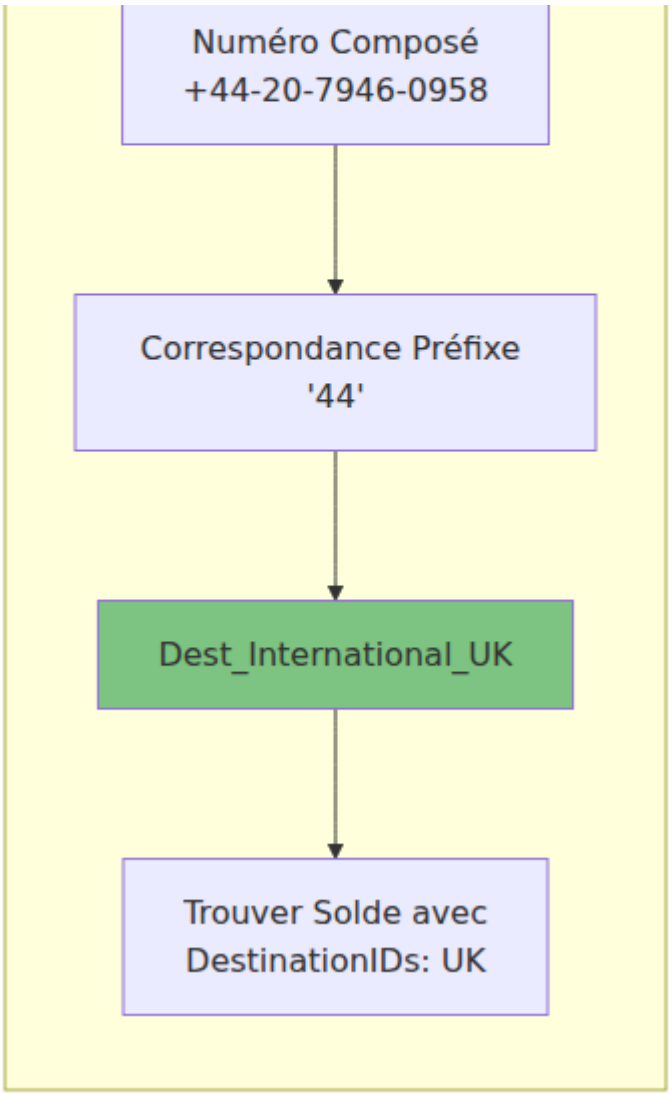
CGRateS utilise des **destinations** pour contrôler où les clients peuvent utiliser leurs soldes :

- **Destinations Géographiques** - Préfixes de numéros pour les appels/SMS VERS des emplacements spécifiques (par exemple, numéros du Royaume-Uni, numéros gratuits aux États-Unis)
- **Destinations PLMN** - Codes de réseau pour l'utilisation des données et le roaming DE réseaux spécifiques (par exemple, Verizon, Vodafone UK)

**Lorsque les clients utilisent des services**, CGRateS fait correspondre :

- **Voix/SMS** : Le préfixe du numéro composé → Destination géographique
- **Données** : Le code PLMN du réseau → Destination PLMN


DESTINATIONS PLMN
Client Utilise des Données
Réseau Connecté PLMN: mcc310.mnc004
OmniCore 5GC
OmniCall
OmniRAN
Om
Dest_PLMN_US_Verizon
Trouver Solde avec DestinationIDs: Verizon
DESTINATIONS
Client Passe un Appel




# Règle Critique

- **Soldes Voix/SMS** → Utiliser des destinations géographiques (le numéro appelé VERS)
- **Soldes de données** → Utiliser des destinations PLMN (le réseau auquel le client est connecté DE)

# Prérequis

```
import cgrateshttpapi
import time

OCS_Obj = cgrateshttpapi.CGRateS("ocs.example.com", "2080")
tenant = "your_tenant_name"
tpid = str(tenant) + "_" + str(int(time.time()))
```

# Partie 1 : Destinations Géographiques (Appels VERS des Lieux)

Les destinations géographiques définissent les préfixes de numéros pour les appels vocaux et les SMS vers des emplacements spécifiques.

## Destinations Domestiques

```
#
# ======================================================================
# DESTINATIONS DOMESTIQUES (États-Unis/Canada - NANP)
#
# ======================================================================

# Mobile & Fixe aux États-Unis/Canada (partagent les indicatifs
régionaux)
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_Domestic_All",
        "Prefixes": ["1"]  # NANP (États-Unis/Canada)
    }]
})

# Numéros gratuits aux États-Unis
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_Domestic_TollFree",
        "Prefixes": ["1800", "1888", "1877", "1866", "1855",
"1844", "1833"]
    }]
})

# Tarifs Premium aux États-Unis
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_Domestic_Premium",
        "Prefixes": ["1900"]
    }]
})
```

## Destinations Internationales

```
#
# ======================================================================
# DESTINATIONS INTERNATIONALES (Par Pays)
#
# ======================================================================

# Royaume-Uni
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_International_UK",
        "Prefixes": ["44"]
    }]
})

# Chine
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_International_China",
        "Prefixes": ["86"]
    }]
})

# Australie - Mobile
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_AU_Mobile",
        "Prefixes": ["614"]  # Mobiles australiens (04xx composé
comme 614xx)
    }]
})

# Australie - Ligne Fixe
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
```

```
        "TPid": tpid,
        "ID": "Dest_AU_Fixed",
        "Prefixes": [
            "612",  # NSW (Sydney)
            "613",  # VIC (Melbourne)
            "617",  # TAS (Hobart)
            "618"   # SA, WA, NT (Adélaïde, Perth, Darwin)
        ]
    }]
})

# Australie - Numéros Gratuits
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_AU_TollFree",
        "Prefixes": [
            "611800",  # Numéro gratuit
            "611300"   # Tarif local
        ]
    }]
})

# Australie - Tarifs Premium
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_AU_Premium",
        "Prefixes": ["6119"]  # Services à tarif premium
    }]
})

# Europe (groupé)
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_International_Europe",
        "Prefixes": [
            "33",  # France
            "49",  # Allemagne
            "39",  # Italie
```

```
            "34",  # Espagne
            "31",  # Pays-Bas
            "32",  # Belgique
            "41"   # Suisse
        ]
    }]
})

# International - Tous (attrape-tout, exclut le domestique)
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_International_All",
        "Prefixes": ["2", "3", "4", "5", "6", "7", "8", "9"]  # Tous sauf "1" (NANP)
    }]
})
```

**Cas d'utilisation :**

- **Plans domestiques** : "Appels illimités vers les numéros des États-Unis/Canada" → `Dest_Domestic_All`
- **Option d'appel vers le Royaume-Uni** : "100 minutes vers le Royaume-Uni" → `Dest_International_UK`
- **Plan australien** : "3000 minutes vers les mobiles et lignes fixes australiennes" → `Dest_AU_Mobile`, `Dest_AU_Fixed`, `Dest_AU_TollFree`
- **Forfait international** : "50 minutes vers n'importe quel numéro international" → `Dest_International_All`

# Partie 2 : Destinations PLMN (Utilisation DE Lieux - Roaming / Utilisation de Données)

Les destinations PLMN définissent des codes de réseau pour les scénarios d'utilisation des données et de roaming. Utilisez le format `mccXXX.mncYYY` où :

- **MCC** = Code de Pays Mobile (3 chiffres)

- **MNC** = Code de Réseau Mobile (2-3 chiffres)

## Règles de Format PLMN

- **Réseau spécifique** : `"mcc310.mnc004"` → Seulement le réseau Verizon 004
- **Tous les réseaux du pays** : `"mcc310"` → N'importe quel réseau américain
- **Exemples** : `mcc310.mnc004`, `mcc234.mnc015`, `mcc505.mnc057`

## Sur le Réseau (Votre Réseau Domicile)

**IMPORTANT** : Définissez VOTRE réseau spécifique où VOS cartes SIM sont provisionnées.

```
# Sur le Réseau (Votre Réseau Domicile)
# C'est le PLMN de VOTRE opérateur - le réseau où les cartes SIM de VOS clients sont actives
# Tous les soldes de données domestiques sur le réseau doivent utiliser cette destination

OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_OnNet",
        "Prefixes": ["mcc505.mnc057"]  # Exemple : MCC 505, MNC 57 si c'est votre réseau domicile
    }]
})
# Remplacez mcc505.mnc057 par le code PLMN réel de VOTRE opérateur
# Tous les soldes de données domestiques sur le réseau doivent utiliser les préfixes définis ici
# Sinon, ils sont considérés comme en roaming
```

# Destinations PLMN de Roaming aux États-Unis

```
#
# ============================================================================
# DESTINATIONS PLMN DE ROAMING AUX ÉTATS-UNIS
#
# ============================================================================

# Verizon Wireless
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_US_Verizon",
        "Prefixes": [
            "mcc310.mnc004", "mcc310.mnc010", "mcc310.mnc012",
"mcc310.mnc013",
            "mcc311.mnc480", "mcc311.mnc481", "mcc311.mnc482",
"mcc311.mnc483"
        ]
    }]
})

# Tous les PLMN américains (attrape-tout)
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_US_All",
        "Prefixes": ["mcc310", "mcc311", "mcc312", "mcc313",
"mcc316"]
    }]
})
# Remarque : Cela correspond à N'IMPORTE quel réseau avec ces MCC
(310, 311, etc.)
# Un client sur mcc310.mnc004 (Verizon), mcc310.mnc410 (AT&T),
mcc311.mnc580 (US Cellular)
# correspondrait TOUS à cette destination
```

# Destinations PLMN de Roaming au Royaume-Uni

```
#
# ============================================================================
# DESTINATIONS PLMN DE ROAMING AU ROYAUME-UNI
#
# ============================================================================

# Vodafone UK
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_UK_Vodafone",
        "Prefixes": ["mcc234.mnc015"]
    }]
})

# EE (Everything Everywhere)
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_UK_EE",
        "Prefixes": ["mcc234.mnc030", "mcc234.mnc033"]
    }]
})

# Tous les PLMN du Royaume-Uni
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_UK_All",
        "Prefixes": ["mcc234"]
    }]
})
```

# Zones de Roaming (Groupes Multi-Pays)

Idéal pour des forfaits de roaming régionaux comme "Roaming Europe" ou "Roaming Asie-Pacifique".

```
#
================================================================
# ZONES DE ROAMING (Groupes Multi-Pays)
#
================================================================

# Zone 1 : Amérique du Nord
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_Zone_NorthAmerica",
        "Prefixes": [
            "mcc310", "mcc311", "mcc312", "mcc313", "mcc316",  #
États-Unis
            "mcc302", "mcc334"  # Canada
        ]
    }]
})
# Utiliser des préfixes uniquement MCC fournit une couverture sur
N'IMPORTE quel réseau dans ces pays
# Idéal pour les forfaits de "roaming Amérique du Nord" qui
fonctionnent sur tous les opérateurs

# Zone 2 : Europe
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_Zone_Europe",
        "Prefixes": [
            "mcc234",  # Royaume-Uni
            "mcc208",  # France
            "mcc262",  # Allemagne
            "mcc222",  # Italie
            "mcc214",  # Espagne
            "mcc228"   # Suisse
        ]
    }]
})

# Zone 3 : Asie-Pacifique
OCS_Obj.SendData({
```

```
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_Zone_AsiaPacific",
        "Prefixes": [
            "mcc505",  # Australie
            "mcc460",  # Chine
            "mcc454",  # Hong Kong
            "mcc440",  # Japon
            "mcc520",  # Thaïlande
            "mcc525",  # Singapour
            "mcc530"   # Nouvelle-Zélande
        ]
    }]
})

# Attrape-tout pour tout roaming (à utiliser avec précaution)
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_Roaming_All",
        "Prefixes": ["mcc"]  # Correspond à N'IMPORTE quel PLMN
(très large)
    }]
})
```

**Cas d'utilisation :**

- **Roaming spécifique au réseau** : "5 Go uniquement sur Verizon" → `Dest_PLMN_US_Verizon`
- **Roaming à l'échelle du pays** : "10 Go aux États-Unis (n'importe quel réseau)" → `Dest_PLMN_US_All`
- **Roaming régional** : "15 Go de roaming en Europe" → `Dest_PLMN_Zone_Europe`
- **Roaming mondial** : "1 Go n'importe où" → `Dest_Roaming_All`

# Quand Utiliser Chaque Approche

| Type de Destination | Cas d'utilisation | Exemple | Types de Soldes |
|---|---|---|---|
| **Géographique** | Voix/SMS vers des pays spécifiques | "100 minutes vers des numéros du Royaume-Uni" | Voix, SMS |
| **PLMN sur le réseau** | Données domestiques sur VOTRE réseau | "100 Go de données sur le réseau" | Données |
| **PLMN spécifique** | Roaming premium sur des réseaux préférés | "5 Go uniquement sur Verizon" | Données (et voix/SMS en roaming) |
| **PLMN à l'échelle du pays** | Roaming sur n'importe quel réseau du pays | "5 Go aux États-Unis (n'importe quel réseau)" | Données (et voix/SMS en roaming) |
| **PLMN de zone** | Forfaits de roaming régionaux | "10 Go de roaming en Europe" | Données (et voix/SMS en roaming) |

# Mises à Jour Dynamiques des Destinations et Accès aux Services

## Comment Fonctionnent les Mises à Jour des Destinations

**Concept Critique :** Les destinations dans CGRateS sont **référencées par ID**, pas copiées. Lorsque vous mettez à jour les préfixes ou les codes PLMN d'une

destination et **chargez le Plan Tarifaire dans le runtime**, **tous les soldes existants référencant cette destination sont immédiatement affectés**.

Cela signifie :

- □ **Pas besoin de mettre à jour les soldes individuels** lors de l'expansion ou de la restriction de la couverture de service
- □ **Changements à l'échelle du réseau** pour tous les clients utilisant cette destination une fois le plan tarifaire chargé
- ⚠ **Les changements prennent effet lorsque vous chargez le Plan Tarifaire** - vous pouvez modifier les destinations plusieurs fois avant de charger, et les changements ne deviennent actifs qu'après `LoadTariffPlanFromStorDb`

# Exemple 1 : Expansion de la Couverture

Vous avez une destination pour le roaming européen :

```
# Configuration initiale - Seulement 3 pays
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_Zone_Europe",
        "Prefixes": [
            "mcc262",  # Allemagne
            "mcc208",  # France
            "mcc222"   # Italie
        ]
    }]
})
```

**Les clients avec des soldes** référencant `Dest_PLMN_Zone_Europe` peuvent se déplacer en Allemagne, France et Italie.

**Plus tard, vous ajoutez l'Espagne et le Royaume-Uni :**

```
# Configuration mise à jour - Maintenant 5 pays
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_Zone_Europe",
        "Prefixes": [
            "mcc262",  # Allemagne
            "mcc208",  # France
            "mcc222",  # Italie
            "mcc214",  # Espagne (NOUVEAU)
            "mcc234"   # Royaume-Uni (NOUVEAU)
        ]
    }]
})

# Chargez le plan tarifaire pour rendre les changements actifs
OCS_Obj.SendData({
    'method': 'ApierV1.LoadTariffPlanFromStorDb',
    'params': [{
        "TPid": tpid,
        "DryRun": False,
        "Validate": True
    }]
})

# Rechargez le cache des destinations
OCS_Obj.SendData({
    'method': 'CacheSv1.ReloadCache',
    'params': [{
        "DestinationIDs": ["Dest_PLMN_Zone_Europe"]
    }]
})
```

**Résultat :** Après avoir chargé le plan tarifaire, tous les clients existants avec des soldes pour `Dest_PLMN_Zone_Europe` obtiennent accès au roaming en Espagne et au Royaume-Uni - aucune mise à jour de solde individuelle requise.

## Exemple 2 : Restriction de la Couverture

**Supprimer un pays** d'une destination bloque l'accès une fois le plan tarifaire chargé :

```
# Supprimer l'Allemagne de la zone Europe
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_Zone_Europe",
        "Prefixes": [
            "mcc208",  # France
            "mcc222",  # Italie
            "mcc214",  # Espagne
            "mcc234"   # Royaume-Uni
            # Allemagne (mcc262) SUPPRIMÉ
        ]
    }]
})

# Chargez le plan tarifaire pour rendre les changements actifs
OCS_Obj.SendData({
    'method': 'ApierV1.LoadTariffPlanFromStorDb',
    'params': [{
        "TPid": tpid,
        "DryRun": False,
        "Validate": True
    }]
})

# Rechargez le cache
OCS_Obj.SendData({
    'method': 'CacheSv1.ReloadCache',
    'params': [{
        "DestinationIDs": ["Dest_PLMN_Zone_Europe"]
    }]
})
```

**Résultat :** Après avoir chargé le plan tarifaire, les clients tentant de se déplacer en Allemagne seront **bloqués**, même s'ils ont encore un solde pour

`Dest_PLMN_Zone_Europe`. Le solde existe mais ne correspond plus à la destination PLMN.

## Exemple 3 : Destinations Géographiques Voix/SMS

Le même principe s'applique aux destinations géographiques pour la voix et les SMS :

```
# Ajouter le Canada à l'appel "domestique"
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_Domestic_All",
        "Prefixes": [
            "1",  # Inclut déjà les États-Unis (et précédemment le Canada)
            # Aucun changement nécessaire - le préfixe "1" couvre déjà les deux
        ]
    }]
})

# Ou créer une destination séparée pour l'Amérique du Nord
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_North_America",
        "Prefixes": [
            "1"  # États-Unis + Canada
        ]
    }]
})
```

**Impact :** Tous les soldes avec `DestinationIDs: "Dest_Domestic_All"` fonctionnent maintenant pour les appels vers les États-Unis et le Canada (si utilisant le préfixe "1").

# Règles d'Accès aux Services

**Critique :** Si un client tente d'utiliser un service (voix/SMS/données) et **aucune destination ne correspond**, le service sera **refusé**, même s'il a des soldes.

**Scénario 1 : Aucune Destination Correspondante**

```
# Le client a un solde :
{
    "BalanceId": "Europe_Data_5GB",
    "BalanceType": "*data",
    "DestinationIDs": "Dest_PLMN_Zone_Europe",  # Seulement
France, Allemagne, Italie
    "Units": 5 * 1024 * 1024 * 1024
}

# Le client se déplace sur un réseau en Espagne (mcc214.mnc001)
# L'Espagne n'est PAS dans Dest_PLMN_Zone_Europe
```

**Résultat :**

- □ Session de données **bloquée** (aucune destination correspondante)
- Le solde reste inutilisé
- Le client reçoit une erreur "Crédit Insuffisant" ou similaire

**Correction :** Ajoutez l'Espagne à `Dest_PLMN_Zone_Europe` OU assurez-vous que le client a un solde monétaire avec `DestinationIDs: "*any"` en tant que solution de secours.

**Scénario 2 : Pas de Solution de Secours Wildcard**

```
# Le client a deux soldes :
{
    "BalanceId": "UK_Voice_100min",
    "BalanceType": "*voice",
    "DestinationIDs": "Dest_International_UK",  # Seulement
préfixe "44"
    "Units": 100 * 60 * 1000000000
}
# AUCUN solde monétaire avec destination "*any"

# Le client appelle l'Australie (+61)
```

**Résultat :**

- □ Appel **bloqué** (aucune destination ne correspond au préfixe "61")
- Le solde de minutes au Royaume-Uni reste inutilisé (ne correspond qu'à "44")
- Pas de solution de secours monétaire pour couvrir l'appel

**Correction :** Ajoutez un solde monétaire avec `DestinationIDs: "*any"` OU ajoutez une destination spécifique pour l'Australie à leur solde.

**Scénario 3 : La Solution de Secours Wildcard Fonctionne**

```
# Le client a :
# Solde 1 : Minutes UK
{
    "BalanceId": "UK_Voice_100min",
    "BalanceType": "*voice",
    "DestinationIDs": "Dest_International_UK",
    "Units": 100 * 60 * 1000000000,
    "BalanceWeight": 1200
}

# Solde 2 : Monétaire avec wildcard
{
    "BalanceId": "PAYG_Credit",
    "BalanceType": "*monetary",
    "DestinationIDs": "*any",  # Correspond à TOUT
    "Units": 5000,  # 50 $
    "BalanceWeight": 1000  # Priorité inférieure
}

# Le client appelle l'Australie (+61)
```

**Résultat :**

- □ Appel **autorisé**
- Le solde de minutes au Royaume-Uni n'est PAS utilisé (ne correspond pas à "61")
- Le solde de crédit PAYG est utilisé (correspond à "*any")
- Facturé au tarif PAYG pour l'Australie

## Tests et Simulation

Les changements de destinations peuvent être testés et simulés dans des environnements non productifs avant d'être appliqués à la production. Cela vous permet de :

- Vérifier que les nouveaux codes PLMN sont corrects
- Tester l'impact de l'expansion/restriction de la couverture
- Valider que les soldes existants se comportent comme prévu avec les destinations mises à jour

- S'assurer que les mécanismes de secours fonctionnent correctement

**Recommandation :** Testez toujours les mises à jour de destinations dans un environnement de staging avec des comptes de test avant de les appliquer à la production.

## Points Clés à Retenir

1. **Les destinations sont dynamiques** - Les changements affectent tous les soldes référencant ces destinations une fois le plan tarifaire chargé
2. **Charger le Plan Tarifaire pour activer** - Utilisez `LoadTariffPlanFromStorDb` pour rendre les changements de destination actifs dans le runtime
3. **Aucune correspondance de destination = Aucun service** - Le système bloque les clients si leur utilisation ne correspond à aucune destination
4. **Wildcard (`*any`) est puissant** - Utilisez-le pour une solution de secours monétaire afin d'éviter des blocages inattendus
5. **Recharge du cache requise** - Rechargez toujours le cache des destinations après avoir chargé le plan tarifaire
6. **Tester avant la production** - Les changements de destinations impactent tous les clients une fois chargés

# Profils de Tarification pour Soldes PAYG/Monétaires

Lors de l'utilisation de soldes monétaires (PAYG - Pay As You Go), vous devez définir des **Profils de Tarification** qui spécifient combien facturer par unité pour différentes destinations.

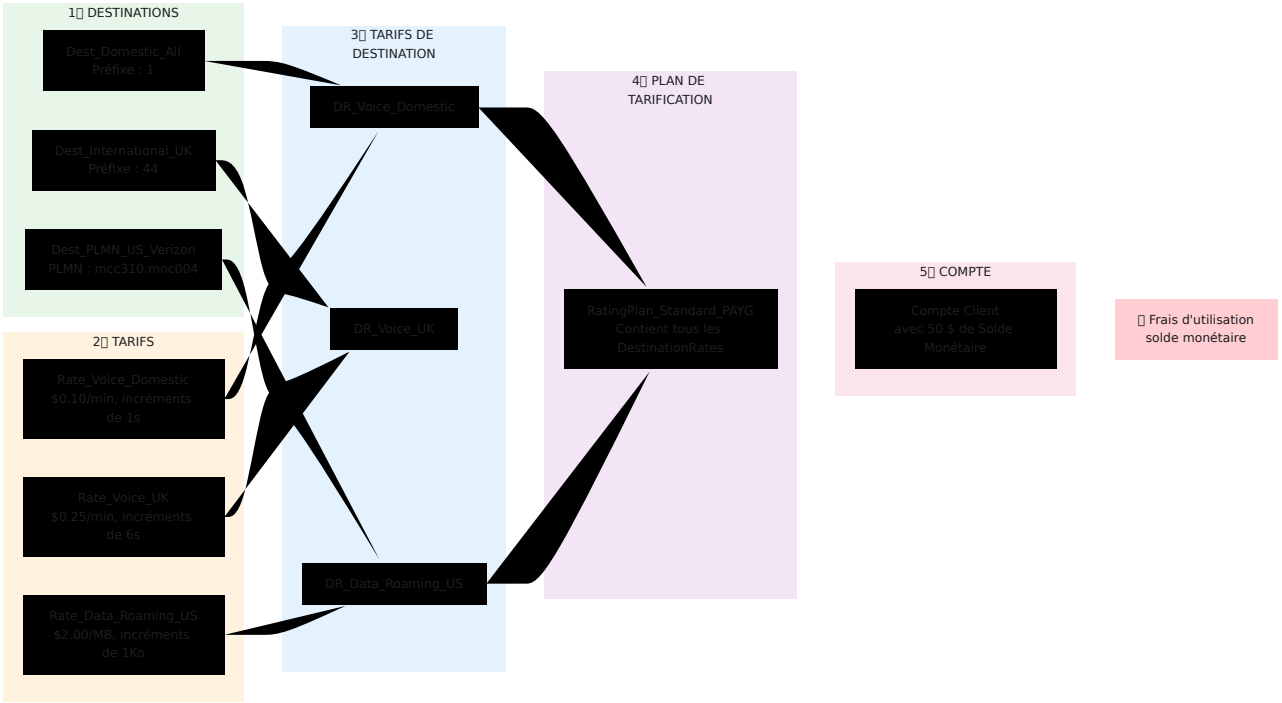
**Remarque Importante sur la Flexibilité de CGRateS :** CGRateS est extrêmement flexible et peut appliquer des tarifs et des plans différents en fonction de nombreux facteurs, y compris :

- D'où le client appelle (PLMN - couvert ici)
- Ce qu'il appelle (destination géographique - couvert ici)
- L'heure de la journée (tarification de pointe/hors pointe)

- Niveau du client (VIP, standard, etc.)
- Niveau de qualité de service
- Et de nombreux autres scénarios de tarification complexes

Ce guide se concentre sur la **configuration au niveau CRM** pour le roaming basé sur PLMN et la tarification basée sur les destinations géographiques, qui couvre les cas d'utilisation les plus courants. Des scénarios de tarification avancés (heure de la journée, niveaux de client, routage basé sur la qualité) sont possibles dans CGRateS mais sont en dehors du champ de ce document axé sur le CRM.

## Comment Fonctionnent les Profils de Tarification




## Définition des Profils de Tarification

Les profils de tarification définissent le coût par unité pour différents types d'utilisation. Voici des exemples pour la voix, les SMS et les données :

```
#
================================================================
# TARIFS VOIX (Par Minute)
#
================================================================

# Voix domestique : 0,10 $/minute
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPRate',
    'params': [{
        "TPid": tpid,
        "ID": "Rate_Voice_Domestic",
        "RateSlots": [{
            "ConnectFee": 0,
            "Rate": 0.10,
            "RateUnit": "60s",         # 1 minute
            "RateIncrement": "1s",     # Facturation par seconde
            "GroupIntervalStart": "0s"
        }]
    }]
})

# Voix au Royaume-Uni : 0,25 $/minute
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPRate',
    'params': [{
        "TPid": tpid,
        "ID": "Rate_Voice_UK",
        "RateSlots": [{
            "ConnectFee": 0.05,        # 0,05 $ de frais de
connexion
            "Rate": 0.25,
            "RateUnit": "60s",
            "RateIncrement": "6s",     # Blocs de 6 secondes
            "GroupIntervalStart": "0s"
        }]
    }]
})

#
================================================================
# TARIFS SMS (Par Message)
#
```

```
# ============================================================================

# SMS domestique : 0,05 $/message
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPRate',
    'params': [{
        "TPid": tpid,
        "ID": "Rate_SMS_Domestic",
        "RateSlots": [{
            "ConnectFee": 0,
            "Rate": 0.05,
            "RateUnit": "1",          # 1 SMS
            "RateIncrement": "1",     # Par message
            "GroupIntervalStart": "0s"
        }]
    }]
})

#
# ============================================================================
# TARIFS DE DONNÉES (Par Mo) - Pour le Roaming
#
# ============================================================================

# Roaming sur les réseaux premium américains (Verizon, AT&T) : 2,00
$/Mo
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPRate',
    'params': [{
        "TPid": tpid,
        "ID": "Rate_Data_Roaming_US_Premium",
        "RateSlots": [{
            "ConnectFee": 0,
            "Rate": 2.00,
            "RateUnit": "1048576",    # 1 Mo en octets
            "RateIncrement": "1024",  # Par Ko
            "GroupIntervalStart": "0s"
        }]
    }]
})
```

**Champs des Slots de Tarification :**

- **ConnectFee** - Frais unique par session
- **Rate** - Coût par RateUnit
- **RateUnit** - Taille de l'unité de facturation (60s, 1Mo, 1 message)
- **RateIncrement** - Unité minimum facturable
- **GroupIntervalStart** - Quand ce niveau de tarif commence

## Lier les Destinations aux Tarifs

Après avoir défini les tarifs, liez-les aux destinations en utilisant les DestinationRates :

```
# Lier la destination domestique au tarif voix domestique
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestinationRate',
    'params': [{
        "TPid": tpid,
        "ID": "DR_Voice_Domestic",
        "DestinationRates": [{
            "DestinationId": "Dest_Domestic_All",
            "RateId": "Rate_Voice_Domestic",
            "RoundingMethod": "*up",
            "RoundingDecimals": 4
        }]
    }]
})

# Lier la destination du Royaume-Uni au tarif voix du Royaume-Uni
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestinationRate',
    'params': [{
        "TPid": tpid,
        "ID": "DR_Voice_UK",
        "DestinationRates": [{
            "DestinationId": "Dest_International_UK",
            "RateId": "Rate_Voice_UK",
            "RoundingMethod": "*up",
            "RoundingDecimals": 4
        }]
    }]
})

# Lier le PLMN Verizon au tarif de données de roaming
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestinationRate',
    'params': [{
        "TPid": tpid,
        "ID": "DR_Data_Roaming_US_Premium",
        "DestinationRates": [{
            "DestinationId": "Dest_PLMN_US_Verizon",
            "RateId": "Rate_Data_Roaming_US_Premium",
            "RoundingMethod": "*up",
            "RoundingDecimals": 4
        }]
```

```
    }]
})
```

## Création du Plan de Tarification

Combinez tous les DestinationRates dans un RatingPlan :

```
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPRatingPlan',
    'params': [{
        "TPid": tpid,
        "ID": "RatingPlan_Standard_PAYG",
        "RatingPlanBindings": [
            # Tarifs voix (poids plus élevé = plus spécifique)
            {"DestinationRatesId": "DR_Voice_UK", "TimingId":
"*any", "Weight": 40},
            {"DestinationRatesId": "DR_Voice_Domestic",
"TimingId": "*any", "Weight": 20},

            # Tarifs SMS
            {"DestinationRatesId": "DR_SMS_Domestic", "TimingId":
"*any", "Weight": 20},

            # Tarifs de données de roaming
            {"DestinationRatesId": "DR_Data_Roaming_US_Premium",
"TimingId": "*any", "Weight": 50}
        ]
    }]
})
```

## Chargement du Plan Tarifaire

Enfin, chargez le plan tarifaire dans CGRateS :

```
# Charger le plan tarifaire
OCS_Obj.SendData({
    'method': 'ApierV1.LoadTariffPlanFromStorDb',
    'params': [{
        "TPid": tpid,
        "DryRun": False,
        "Validate": True
    }]
})

# Rechargez le cache
OCS_Obj.SendData({
    'method': 'CacheSv1.ReloadCache',
    'params': [{
        "Tenant": tenant,
        "DestinationIDs": ["*all"]
    }]
})
```

Voir aussi :

- Définir des Produits - Workflow complet pour créer des produits
- Actions CGRateS et Comportements de Recharge - Comment définir des Actions et des approches de gestion des soldes

# Meilleures Pratiques

## 1. Soyez Spécifique avec le Sur-Réseau

```
# Bien - réseau domicile spécifique
"DestinationIDs": "Dest_PLMN_OnNet",  # mcc505.mnc057

# Mal - trop large, correspond aux concurrents
"DestinationIDs": "Dest_PLMN_AU_All",  # mcc505 (tous les
opérateurs AU)
```

## 2. Utilisez des Noms de Destination Significatifs

```
# Bien - auto-documentant
Dest_PLMN_US_Verizon
Dest_International_UK
Dest_PLMN_Zone_Europe

# Mal - peu clair
Dest_Data_1
Dest_Voice_Package
```

## 3. Regroupez Logiquement les Destinations Connexes

```
# Bien - regroupement logique
Dest_Domestic_All
Dest_Domestic_TollFree
Dest_Domestic_Premium

# Mal - trop granulaire
Dest_Area_Code_212
Dest_Area_Code_213
Dest_Area_Code_214
```

## 4. Documentez les Sources PLMN

Lors de la définition des destinations PLMN, ajoutez des commentaires indiquant d'où proviennent les codes MCC/MNC :

```
# Verizon Wireless
# Source : https://www.mcc-mnc.com/ (vérifié 2024-12)
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_US_Verizon",
        "Prefixes": [
            "mcc310.mnc004",  # Verizon - Nationwide
            "mcc310.mnc010",  # Verizon - East
            ...
        ]
    }]
})
```

# Dépannage

## Problème : Solde Ne Correspond Pas à la Destination

**Symptôme :** Le client a un solde mais ne peut pas l'utiliser pour une destination spécifique

**Vérifiez :**

```
# 1. Vérifiez que la destination existe
OCS_Obj.SendData({
    'method': 'ApierV2.GetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_US_Verizon"
    }]
})

# 2. Vérifiez le champ DestinationIDs du solde
# Assurez-vous qu'il inclut la destination ou "*any"

# 3. Pour PLMN : Vérifiez que le client est réellement sur ce
r��seau
# Vérifiez les messages CCR Diameter pour l'AVP Visited-PLMN-Id
```

## Problème : Utilisation des Données Comptée Comme Roaming

**Symptôme :** L'utilisation domestique des données consomme le solde de roaming au lieu du solde sur le réseau

**Cause :** Le PLMN du client n'est pas défini dans `Dest_PLMN_OnNet`

**Solution :**

```
# Vérifiez sur quel PLMN le client se trouve (à partir de CCR Diameter)
# Vérifiez que le solde inclut ce PLMN :

OCS_Obj.SendData({
    'method': 'ApierV2.GetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_OnNet"
    }]
})
# Assurez-vous que les Préfixes incluent le réseau PLMN réel du client
# Exemple : ["mcc505.mnc057"] doit correspondre au réseau de la carte SIM du client
```

## Problème : Appels Internationaux Ne Fonctionnent Pas

**Symptôme :** Le client a des "minutes internationales" mais les appels échouent

**Vérifiez :**

1. **Préfixe de destination** : Le numéro composé correspond-il au préfixe de destination ?
   - Appelant +44-20-xxxx → Nécessite une destination avec le préfixe "44"
2. **DestinationIDs du solde** : Le solde inclut-il cette destination ?
   - Vérifiez que le solde a `"DestinationIDs": "Dest_International_UK"` ou `"Dest_International_All"`
3. **Solde non expiré** : Vérifiez `ExpiryTime`
4. **Solde a des unités** : Vérifiez `Units > 0`

# Documentation Connexe

- Actions CGRateS et Comportements de Recharge - Comment utiliser les destinations dans les Actions, approches de solde et stratégies de

tarification

- Définir des Produits - Workflow complet de création de produits

# Ansible Playbooks : Guide détaillé

Les produits OmniCRM sont provisionnés à l'aide d'**Ansible**, permettant une gestion automatisée des services en fonction des exigences spécifiques de chaque produit et de son inventaire associé.

Voir aussi : Provisionnement de carte SIM pour un exemple complet de provisionnement basé sur Ansible pour les services mobiles, y compris les SIM physiques et les eSIM.

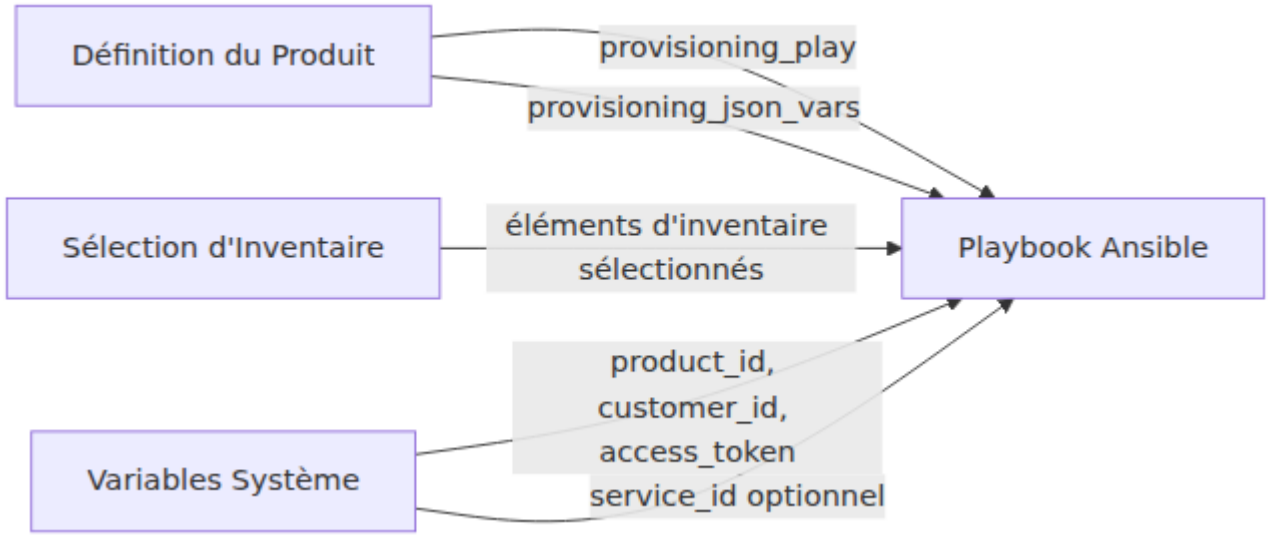
## Comment les Playbooks et les Produits Fonctionnent Ensemble

**Concept Critique :** Les playbooks sont ce qui crée réellement des services dans OmniCRM. Lorsque vous assignez un playbook à un produit, vous définissez **ce qui se passe** lorsque ce produit est provisionné. Cela peut signifier différentes choses pour différents produits.

### Les Produits Déclenchent des Playbooks

Lorsqu'un produit est provisionné dans OmniCRM :

1. La définition du produit spécifie quel playbook exécuter (via le champ `provisioning_play`)
2. Le produit passe des variables au playbook (via `provisioning_json_vars` et les sélections d'inventaire)
3. Le playbook s'exécute et fait ce pour quoi il est programmé
4. Le playbook détermine ce qui est créé (le cas échéant)




# Ce Que les Playbooks Peuvent Faire

Un seul playbook de provisionnement peut :

**Créer Plusieurs Services**
Un playbook de produit groupé pourrait créer :

- Un enregistrement principal de service Internet
- Un enregistrement de service d'addon IPTV
- Un enregistrement de service VoIP
- Le tout avec une seule action de provisionnement de produit

**Créer Aucun Service**
Certains playbooks ne créent pas d'enregistrements de service du tout :

- Un playbook qui configure simplement l'équipement CPE
- Un playbook qui envoie la configuration à l'équipement réseau
- Un playbook qui met à jour des systèmes externes

**Créer Un Service**
Le modèle le plus courant :

- Créer un seul enregistrement de service pour le client
- Lier l'inventaire à ce service
- Configurer la facturation pour ce service

**Modifier des Services Existants**
Les playbooks de recharge et d'addon :

- Ne créent pas de nouveaux services
- Mettent à jour les enregistrements de service existants (ajout de données, prolongation de l'expiration, etc.)
- Ajoutent des soldes aux comptes de facturation existants

**Effectuer des Actions Sans Enregistrements de Service**
Certains playbooks sont purement opérationnels :

- Réinitialiser les soldes de compte
- Échanger des éléments d'inventaire entre clients
- Générer des rapports ou des configurations

## Exemple : Comportements Différents des Playbooks

```
# Produit 1 : Service SIM Mobile (crée 1 service)
# provisioning_play: play_simple_service
- Crée un enregistrement de service dans le CRM
- Crée un compte de facturation dans l'OCS
- Assigne l'inventaire de la carte SIM et du numéro de téléphone
- Envoie un e-mail de bienvenue

# Produit 2 : Pack Internet (crée 3 services)
# provisioning_play: play_bundle_internet_tv_voice
- Crée un enregistrement de service Internet
- Crée un enregistrement de service IPTV
- Crée un enregistrement de service VoIP
- Lie tous au même client
- Compte de facturation unique pour le pack

# Produit 3 : Recharge de Données (crée 0 services)
# provisioning_play: play_topup_no_charge
- Trouve le service existant par service_id
- Ajoute un solde de données au compte OCS existant
- Met à jour la date d'expiration du service
- AUCUN nouveau service créé

# Produit 4 : Configuration CPE (crée 0 services)
# provisioning_play: play_prov_cpe_mikrotik
- Génère la configuration du routeur
- Met à jour l'enregistrement d'inventaire avec la config
- Envoie la config à l'équipe de support
- AUCUN service créé (juste configuration de l'équipement)
```

Le point clé : **Le playbook définit le comportement, le produit est juste un déclencheur.**

# Plays vs Tâches

Comprendre la distinction entre Plays et Tâches est fondamental pour travailler avec les playbooks OmniCRM.

**Play** (Playbook)
Un flux de travail de provisionnement complet qui orchestre plusieurs tâches pour atteindre un objectif commercial. Les Plays sont les playbooks de haut niveau stockés dans `OmniCRM-API/Provisioners/plays/` et sont référencés dans les définitions de produits.

Exemples :

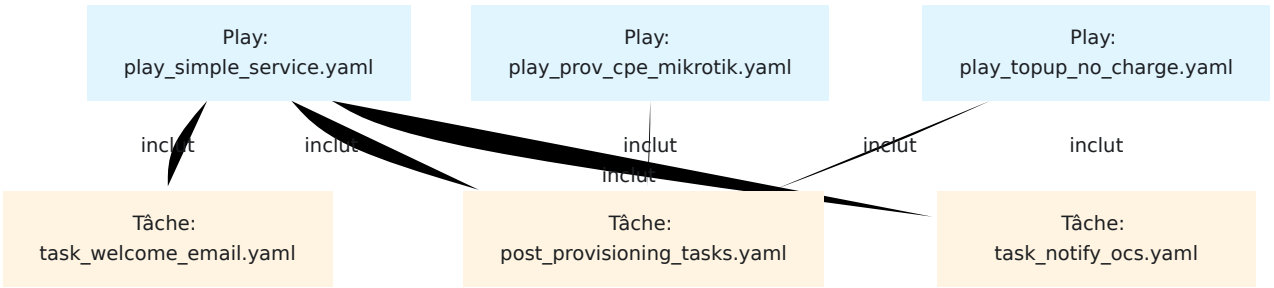
- `play_simple_service.yaml` - Provisionner un service de base
- `play_topup_no_charge.yaml` - Appliquer une recharge gratuite à un service
- `play_prov_cpe_mikrotik.yaml` - Configurer l'équipement de locaux clients

**Tâche** (Composant Réutilisable)
Un ensemble d'opérations autonome et réutilisable qui peut être inclus par plusieurs plays. Les Tâches sont préfixées par `task_` et se trouvent dans le même répertoire.

Exemples :

- `task_welcome_email.yaml` - Envoyer un e-mail de bienvenue à un client
- `task_activate_olt.yaml` - Activer l'équipement OLT
- `task_notify_ocs.yaml` - Envoyer des notifications au système de facturation

La relation entre eux :



```yaml
# play_simple_service.yaml (Un Play)
- name: Play de Provisionnement Simple
  hosts: localhost
  tasks:
    - name: Bloc principal de provisionnement
      block:
        - name: Créer le service
          uri: ...

        - name: Configurer la facturation
          uri: ...

        # Inclure la tâche réutilisable
        - include_tasks: task_welcome_email.yaml

        # Inclure les tâches post-provisionnement
        - include_tasks: post_provisioning_tasks.yaml
```

# Structure et Anatomie du Playbook

Tous les playbooks OmniCRM suivent une structure cohérente. Comprendre cette structure est essentiel pour créer et maintenir des playbooks.

## Structure de Base

Chaque playbook commence par ces en-têtes standard :

```yaml
- name: Nom Descriptif du Playbook
  hosts: localhost           # Toujours localhost pour OmniCRM
  gather_facts: no           # Désactivé pour la performance
  become: False              # Ne pas élever les privilèges

  tasks:
    - name: Bloc principal
      block:
        # Les tâches de provisionnement vont ici

      rescue:
        # Les tâches de rollback/nettoyage vont ici
```

## Explication des En-têtes

**name**
Nom descriptif affiché dans les journaux de provisionnement et l'interface utilisateur. Cela apparaît comme `playbook_description` dans l'enregistrement de provision.

**hosts: localhost**
Tous les playbooks OmniCRM s'exécutent sur localhost car ils interagissent avec des systèmes distants via des API, pas SSH.

**gather_facts: no**
La collecte de faits d'Ansible est désactivée car :

- Nous n'avons pas besoin d'informations système
- Cela ajoute une surcharge inutile
- Peut faire planter les navigateurs s'il est affiché dans la sortie de débogage

**become: False**
Aucune élévation de privilèges n'est nécessaire puisque nous faisons des appels API, pas de modifications de fichiers système.

## Chargement de Configuration

Chaque playbook doit charger le fichier de configuration central :

```
tasks:
  - name: Inclure les vars de crm_config
    ansible.builtin.include_vars:
      file: "../../crm_config.yaml"
      name: crm_config
```

Cela rend la configuration disponible sous `crm_config.ocs.cgrates`, `crm_config.crm.base_url`, etc.

Le `crm_config.yaml` contient généralement :

```
ocs:
  cgrates: "10.0.1.100:2080"
  ocsTenant: "default_tenant"
crm:
  base_url: "https://crm.example.com"
```

## Modèles d'Accès aux Variables

Les variables peuvent provenir de plusieurs sources :

**De la Définition du Produit :**

```
- name: Accéder au product_id passé par OmniCRM
  debug:
    msg: "Provisionnement du produit {{ product_id }}"
```

**De la Sélection d'Inventaire :**

```
- name: Obtenir l'ID d'inventaire pour la carte SIM
  set_fact:
    sim_card_id: "{{ hostvars[inventory_hostname]['SIM Card'] |
int }}"
  when: "'SIM Card' in hostvars[inventory_hostname]"
```

**Des Réponses API :**

```
- name: Obtenir les informations du produit depuis l'API CRM
  uri:
    url: "http://localhost:5000/crm/product/product_id/{{
product_id }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
    return_content: yes
  register: api_response_product

- name: Utiliser le nom du produit
  debug:
    msg: "Le nom du produit est {{
api_response_product.json.product_name }}"
```

# Modèles de Playbook Courants

## Modèle de Provisionnement de Service

C'est le modèle le plus courant pour créer de nouveaux services.

```yaml
- name: Playbook de Provisionnement de Service
  hosts: localhost
  gather_facts: no
  become: False

  tasks:
    - name: Bloc principal
      block:

      # 1. Charger la configuration
      - name: Inclure les vars de crm_config
        ansible.builtin.include_vars:
          file: "../../crm_config.yaml"
          name: crm_config

      # 2. Obtenir les informations du produit
      - name: Obtenir les informations du produit depuis l'API CRM
        uri:
          url: "http://localhost:5000/crm/product/product_id/{{
product_id }}"
          method: GET
          headers:
            Authorization: "Bearer {{ access_token }}"
          return_content: yes
          validate_certs: no
        register: api_response_product

      # 3. Obtenir les informations du client
      - name: Obtenir les informations du client depuis l'API CRM
        uri:
          url: "http://localhost:5000/crm/customer/customer_id/{{
customer_id }}"
          method: GET
          headers:
            Authorization: "Bearer {{ access_token }}"
          return_content: yes
        register: api_response_customer

      # 4. Définir des faits à partir des données récupérées
      - name: Définir les faits du package
        set_fact:
          package_name: "{{ api_response_product.json.product_name
}}"
```

```
          package_comment: "{{ api_response_product.json.comment
}}"
          setup_cost: "{{
api_response_product.json.retail_setup_cost }}"
          monthly_cost: "{{ api_response_product.json.retail_cost
}}"

      # 5. Générer des identifiants uniques
      - name: Générer UUID
        set_fact:
          uuid: "{{ 99999999 | random | to_uuid }}"

      - name: Générer UUID de Service
        set_fact:
          service_uuid: "Service_{{ uuid[0:8] }}"

      # 6. Créer un compte dans le système de facturation
      - name: Créer un compte dans OCS/CGRateS
        uri:
          url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
          method: POST
          body_format: json
          headers:
            Content-Type: "application/json"
          body:
            {
              "method": "ApierV2.SetAccount",
              "params": [{
                  "Tenant": "{{ crm_config.ocs.ocsTenant }}",
                  "Account": "{{ service_uuid }}",
                  "ActionPlanIds": [],
                  "ActionPlansOverwrite": true,
                  "ExtraOptions": {
                    "AllowNegative": false,
                    "Disabled": false
                  },
                  "ReloadScheduler": true
              }]
            }
          status_code: 200
        register: ocs_response

      - name: Vérifier la création du compte OCS
        assert:
```

```
          that:
            - ocs_response.status == 200
            - ocs_response.json.result == "OK"

      # 7. Ajouter un solde initial
      - name: Ajouter 0 Solde Monétaire
        uri:
          url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
          method: POST
          body_format: json
          body:
            {
              "method": "ApierV1.AddBalance",
              "params": [{
                  "Tenant": "{{ crm_config.ocs.ocsTenant }}",
                  "Account": "{{ service_uuid }}",
                  "BalanceType": "*monetary",
                  "Categories": "*any",
                  "Balance": {
                      "ID": "Solde Initial",
                      "Value": 0,
                      "ExpiryTime": "+4320h",
                      "Weight": 1,
                      "Blocker": true
                  }
              }]
            }
          status_code: 200
        register: balance_response

      # 8. Créer un enregistrement de service dans le CRM
      - name: Obtenir la date et l'heure actuelles au format ISO
8601
        command: date --utc +%Y-%m-%dT%H:%M:%S%z
        register: current_date_time

      - name: Ajouter le Service via l'API
        uri:
          url: "http://localhost:5000/crm/service/"
          method: PUT
          body_format: json
          headers:
            Content-Type: "application/json"
            Authorization: "Bearer {{ access_token }}"
```

```
          body:
            {
              "customer_id": "{{ customer_id }}",
              "product_id": "{{ product_id }}",
              "service_name": "{{ package_name }} - {{
service_uuid }}",
              "service_type": "generic",
              "service_uuid": "{{ service_uuid }}",
              "service_billed": true,
              "service_taxable": true,
              "service_provisioned_date": "{{
current_date_time.stdout }}",
              "service_status": "Active",
              "wholesale_cost": "{{
api_response_product.json.wholesale_cost | float }}",
              "retail_cost": "{{ monthly_cost | float }}"
            }
          status_code: 200
        register: service_creation_response

      # 9. Ajouter une transaction de coût de mise en place
      - name: Ajouter une Transaction de Coût de Mise en Place via
l'API
        uri:
          url: "http://localhost:5000/crm/transaction/"
          method: PUT
          headers:
            Content-Type: "application/json"
            Authorization: "Bearer {{ access_token }}"
          body_format: json
          body:
            {
              "customer_id": {{ customer_id | int }},
              "service_id": {{
service_creation_response.json.service_id | int }},
              "title": "{{ package_name }} - Coûts de Mise en
Place",
              "description": "Coûts de mise en place pour {{
package_comment }}",
              "invoice_id": null,
              "retail_cost": "{{ setup_cost | float }}"
            }
          return_content: yes
        register: transaction_response
```

```
      # 10. Inclure les tâches post-provisionnement
      - include_tasks: post_provisioning_tasks.yaml

      rescue:

      # Section de rollback/nettoyage
      - name: Imprimer toutes les vars pour le débogage
        debug:
          var: hostvars[inventory_hostname]

      - name: Supprimer le compte dans l'OCS
        uri:
          url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
          method: POST
          body_format: json
          body:
            {
              "method": "ApierV2.RemoveAccount",
              "params": [{
                  "Tenant": "{{ crm_config.ocs.ocsTenant }}",
                  "Account": "{{ service_uuid }}",
                  "ReloadScheduler": true
              }]
            }
          status_code: 200
        ignore_errors: True
        when: service_uuid is defined

      - name: Supprimer le Service du CRM s'il a été créé
        uri:
          url: "http://localhost:5000/crm/service/service_id/{{
service_creation_response.json.service_id }}"
          method: DELETE
          headers:
            Authorization: "Bearer {{ access_token }}"
          status_code: 200
        ignore_errors: True
        when: service_creation_response is defined

      - name: Échouer si ce n'est pas un déprovisionnement
intentionnel
        assert:
```

```
            that:
              - action == "deprovision"
```

## Modèle de Recharge/Topup

Utilisé pour ajouter des crédits, des données ou du temps à des services existants.

```yaml
- name: Playbook de Topup de Service
  hosts: localhost
  gather_facts: no
  become: False

  tasks:
    - name: Inclure les vars de crm_config
      ansible.builtin.include_vars:
        file: "../../crm_config.yaml"
        name: crm_config

    # 1. Obtenir les informations du service
    - name: Obtenir les informations du Service depuis l'API CRM
      uri:
        url: "http://localhost:5000/crm/service/service_id/{{
service_id }}"
        method: GET
        headers:
          Authorization: "Bearer {{ access_token }}"
        return_content: yes
      register: api_response_service

    # 2. Obtenir les informations du produit (ce qu'il faut
recharger)
    - name: Obtenir les informations du Produit depuis l'API CRM
      uri:
        url: "http://localhost:5000/crm/product/product_id/{{
product_id }}"
        method: GET
        headers:
          Authorization: "Bearer {{ access_token }}"
        return_content: yes
      register: api_response_product

    # 3. Extraire les détails du service
    - name: Définir les faits du service
      set_fact:
        service_uuid: "{{ api_response_service.json.service_uuid
}}"
        customer_id: "{{ api_response_service.json.customer_id }}"
        package_name: "{{ api_response_product.json.product_name
}}"
        topup_value: "{{ api_response_product.json.retail_cost }}"
```

```
    # 4. Exécuter l'action dans le système de facturation
(recharge gratuite)
    - name: Exécuter l'Action pour ajouter des crédits
      uri:
        url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
        method: POST
        body_format: json
        body:
          {
            "method": "APIerSv1.ExecuteAction",
            "params": [{
                "Tenant": "{{ crm_config.ocs.ocsTenant }}",
                "Account": "{{ service_uuid }}",
                "ActionsId": "Action_Topup_Standard"
            }]
          }
        status_code: 200
      register: action_response

    - name: Vérifier que l'action a été exécutée avec succès
      assert:
        that:
          - action_response.status == 200
          - action_response.json.result == "OK"

    # 5. Réinitialiser toutes les limites déclenchées
    - name: Réinitialiser ActionTriggers
      uri:
        url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
        method: POST
        body_format: json
        body:
          {
            "method": "APIerSv1.ResetAccountActionTriggers",
            "params": [{
                "Tenant": "{{ crm_config.ocs.ocsTenant }}",
                "Account": "{{ service_uuid }}",
                "Executed": false
            }]
          }
        status_code: 200

    # 6. Mettre à jour les dates de service
```

```
    - name: Calculer la nouvelle date d'expiration
      command: "date --utc +%Y-%m-%dT%H:%M:%S%z -d '+30 days'"
      register: new_expiry_date

    - name: Mettre à jour le Service avec la nouvelle expiration
      uri:
        url: "http://localhost:5000/crm/service/{{ service_id }}"
        method: PATCH
        headers:
          Authorization: "Bearer {{ access_token }}"
          Content-Type: "application/json"
        body_format: json
        body:
          {
            "service_deactivate_date": "{{ new_expiry_date.stdout
}}",
            "service_status": "Active"
          }

    # 7. Optionnel : Envoyer une notification
    - name: Envoyer un SMS de Notification
      uri:
        url: "http://sms-gateway/api/send"
        method: POST
        body_format: json
        body:
          {
            "source": "CompanyName",
            "destination": "{{ customer_phone }}",
            "message": "Votre service a été rechargé. Nouvelle
expiration : {{ new_expiry_date.stdout }}"
          }
        status_code: 201
      ignore_errors: True
```

## Modèle de Provisionnement CPE

Utilisé pour configurer l'équipement de locaux clients (routeurs, modems, ONTs).

```yaml
- name: Playbook de Provisionnement CPE
  hosts: localhost
  gather_facts: no
  become: False

  tasks:
    - name: Inclure les vars de crm_config
      ansible.builtin.include_vars:
        file: "../../crm_config.yaml"
        name: crm_config

    # 1. Obtenir l'élément d'inventaire pour CPE
    - name: Définir l'ID d'inventaire CPE à partir des hostvars
      set_fact:
        cpe_inventory_id: "{{ hostvars[inventory_hostname]['WiFi
Router CPE'] | int }}"
      when: "'WiFi Router CPE' in hostvars[inventory_hostname]"

    # 2. Obtenir les détails CPE de l'inventaire
    - name: Obtenir les données d'inventaire pour CPE
      uri:
        url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ cpe_inventory_id }}"
        method: GET
        headers:
          Authorization: "Bearer {{ access_token }}"
        return_content: yes
      register: api_response_cpe

    # 3. Obtenir les informations du site client
    - name: Obtenir les informations du site depuis l'API
      uri:
        url: "{{ crm_config.crm.base_url
}}/crm/site/customer_id/{{ customer_id }}"
        method: GET
        headers:
          Authorization: "Bearer {{ access_token }}"
        return_content: yes
      register: api_response_site

    # 4. Mettre à jour l'inventaire CPE avec l'emplacement
    - name: Patch l'élément d'inventaire CPE avec l'emplacement
      uri:
```

```
        url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ cpe_inventory_id }}"
        method: PATCH
        body_format: json
        headers:
          Authorization: "Bearer {{ access_token }}"
        body:
          {
            "address_line_1": "{{
api_response_site.json.0.address_line_1 }}",
            "city": "{{ api_response_site.json.0.city }}",
            "state": "{{ api_response_site.json.0.state }}",
            "latitude": "{{ api_response_site.json.0.latitude }}",
            "longitude": "{{ api_response_site.json.0.longitude
}}"
          }
        status_code: 200

    # 5. Générer des identifiants
    - name: Définir le nom d'hôte CPE
      set_fact:
        cpe_hostname: "CPE_{{ cpe_inventory_id }}"
        cpe_username: "admin_{{ cpe_inventory_id }}"

    - name: Générer un mot de passe aléatoire
      set_fact:
        cpe_password: "{{ lookup('pipe', 'cat /dev/urandom | tr -
dc a-zA-Z0-9 | head -c 16') }}"

    # 6. Générer des identifiants WiFi
    - name: Définir le SSID WiFi
      set_fact:
        wifi_ssid: "Network_{{ cpe_inventory_id }}"

    - name: Générer un mot de passe WiFi
      set_fact:
        word_list:
          - apple
          - cloud
          - river
          - mountain
          - ocean

    - name: Créer le PSK WiFi
```

```
      set_fact:
        random_word: "{{ word_list | random }}"
        random_number: "{{ 99999 | random(start=10000) }}"

    - name: Combiner le PSK WiFi
      set_fact:
        wifi_psk: "{{ random_word }}{{ random_number }}"

    # 7. Générer le fichier de configuration
    - name: Définir le nom du fichier de config
      set_fact:
        config_name: "{{ cpe_hostname }}_{{ lookup('pipe', 'date
+%Y%m%d%H%M%S') }}.cfg"
        config_dest: "/tmp/{{ cpe_hostname }}_{{ lookup('pipe',
'date +%Y%m%d%H%M%S') }}.cfg"

    - name: Créer la config à partir du modèle
      template:
        src: "templates/cpe_router_config.j2"
        dest: "{{ config_dest }}"

    # 8. Lire la config générée
    - name: Lire le fichier de config
      ansible.builtin.slurp:
        src: "{{ config_dest }}"
      register: config_content

    # 9. Mettre à jour l'inventaire avec les informations de
provisionnement
    - name: Patch l'inventaire CPE avec la config
      uri:
        url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ cpe_inventory_id }}"
        method: PATCH
        body_format: json
        headers:
          Authorization: "Bearer {{ access_token }}"
        body:
          {
            "itemtext3": "{{ wifi_ssid }}",
            "itemtext4": "{{ wifi_psk }}",
            "management_url": "{{ cpe_hostname }}",
            "management_username": "{{ cpe_username }}",
            "management_password": "{{ cpe_password }}",
```

```
            "config_content": "{{ config_content.content |
b64decode }}",
            "inventory_notes": "Provisionné : {{ lookup('pipe',
'date +%Y-%m-%d') }}"
          }
        status_code: 200

    # 10. Envoyer la config à l'équipe de support
    - name: Envoyer la configuration par e-mail au support
      uri:
        url: "https://api.mailjet.com/v3.1/send"
        method: POST
        body_format: json
        headers:
          Content-Type: "application/json"
        body:
          {
            "Messages": [{
              "From": {
                "Email": "provisioning@example.com",
                "Name": "Système de Provisionnement"
              },
              "To": [{
                "Email": "support@example.com",
                "Name": "Équipe de Support"
              }],
              "Subject": "Config CPE - {{ cpe_hostname }}",
              "Attachments": [{
                "ContentType": "text/plain",
                "Filename": "{{ config_name }}",
                "Base64Content": "{{ config_content.content }}"
              }]
            }]
          }
        user: "{{ mailjet_api_key }}"
        password: "{{ mailjet_api_secret }}"
        force_basic_auth: true
        status_code: 200
```

# Modèle de Renouvellement Automatique

Configurer des charges récurrentes ou des renouvellements automatiques à l'aide des ActionPlans CGRateS.

```
# Partie d'un playbook de recharge qui configure le renouvellement
automatique

# 1. Normaliser le paramètre auto_renew
- name: Normaliser auto_renew en booléen
  set_fact:
    auto_renew_bool: "{{ (auto_renew | string | lower) in ['true',
'1', 'yes'] }}"

# 2. Créer une action pour le renouvellement automatique
- name: Créer Action pour AutoRenouvellement
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.SetActions",
        "params": [{
          "ActionsId": "Action_AutoTopup_{{ service_uuid }}_{{
product_id }}",
          "Overwrite": true,
          "Actions": [
            {
              "Identifier": "*http_post",
              "ExtraParameters": "{{ crm_config.crm.base_url
}}/crm/provision/simple_provision_addon/service_id/{{ service_id
}}/product_id/{{ product_id }}"
            },
            {
              "Identifier": "*cdrlog",
              "BalanceType": "*generic",
              "ExtraParameters": "
{\"Category\":\"^activation\",\"Destination\":\"Auto Renewal\"}"
            }
          ]
        }]
      }
    status_code: 200
  register: action_response
  when: auto_renew_bool

# 3. Créer ActionPlan mensuel
```

```
- name: Créer ActionPlan pour Renouvellement Mensuel
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.SetActionPlan",
        "params": [{
          "Id": "ActionPlan_Monthly_{{ service_uuid }}_{{
product_id }}",
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "ActionPlan": [{
            "ActionsId": "Action_AutoTopup_{{ service_uuid }}_{{
product_id }}",
            "Years": "*any",
            "Months": "*any",
            "MonthDays": "*any",
            "WeekDays": "*any",
            "Time": "*monthly",
            "StartTime": "*now",
            "Weight": 10
          }],
          "Overwrite": true,
          "ReloadScheduler": true
        }]
      }
    status_code: 200
  when: auto_renew_bool

# 4. Assigner ActionPlan au compte
- name: Assigner ActionPlan au compte
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV2.SetAccount",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Account": "{{ service_uuid }}",
          "ActionPlanIds": ["ActionPlan_Monthly_{{ service_uuid
}}_{{ product_id }}"],
```

```
          "ActionPlansOverwrite": true,
          "ReloadScheduler": true
        }]
      }
    status_code: 200
  when: auto_renew_bool

# 5. Supprimer ActionPlan si le renouvellement automatique est
désactivé
- name: Supprimer ActionPlan du compte
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.RemoveActionPlan",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Id": "ActionPlan_Monthly_{{ service_uuid }}_{{
product_id }}"
        }]
      }
    status_code: 200
  ignore_errors: true
  when: not auto_renew_bool
```

# Tâches Réutilisables

Les tâches réutilisables sont de petits playbooks autonomes qui peuvent être inclus par plusieurs plays. Elles favorisent la réutilisation du code et la cohérence.

## Tâche d'E-mail de Bienvenue

`task_welcome_email.yaml` - Envoie un e-mail de bienvenue aux nouveaux clients.

```
# Cette tâche s'attend à ce que ces variables soient définies par
le play parent :
# - api_response_customer (détails du client)
# - package_name (nom du produit)
# - monthly_cost (coût récurrent)
# - setup_cost (coût unique)

- name: Définir la configuration de l'e-mail
  set_fact:
    mailjet_api_key: "{{ lookup('env', 'MAILJET_API_KEY') }}"
    mailjet_api_secret: "{{ lookup('env', 'MAILJET_SECRET') }}"
    email_from: "noreply@example.com"
    recipients: []

- name: Définir le sujet de l'e-mail et le nom de l'expéditeur
  set_fact:
    email_subject: "Bienvenue dans notre service !"
    email_from_name: "Équipe du Service Client"

- name: Préparer la liste des destinataires à partir des contacts
du client
  loop: "{{ api_response_customer.json.contacts }}"
  set_fact:
    recipients: "{{ recipients + [{'Email': item.contact_email,
'Name': item.contact_firstname ~ ' ' ~ item.contact_lastname}] }}"

- name: Obtenir le nom du premier contact
  set_fact:
    first_contact: "{{
api_response_customer.json.contacts[0].contact_firstname }}"

- name: Envoyer l'e-mail de bienvenue
  uri:
    url: "https://api.mailjet.com/v3.1/send"
    method: POST
    body_format: json
    headers:
      Content-Type: "application/json"
    body:
      {
        "Messages": [{
          "From": {
            "Email": "{{ email_from }}",
```

```
            "Name": "{{ email_from_name }}"
          },
          "To": "{{ recipients }}",
          "Subject": "{{ email_subject }}",
          "TextPart": "Cher {{ first_contact }}, bienvenue ! Votre
service est prêt.",
          "HTMLPart": "Cher {{ first_contact }},<br/><h3>Bienvenue
!</h3><br/>Votre service {{ package_name }} est maintenant actif.
<br/>Coût mensuel : ${{ monthly_cost }}<br/>Frais de mise en place
: ${{ setup_cost }}<br/>Si vous avez des questions, contactez
support@example.com"
        }]
      }
    user: "{{ mailjet_api_key }}"
    password: "{{ mailjet_api_secret }}"
    force_basic_auth: true
    status_code: 200
  register: email_response
```

## Tâches Post-Provisionnement

`post_provisioning_tasks.yaml` - Nettoyage standard et notifications exécutées après chaque provision.

```
# Ce fichier est inclus à la fin de la plupart des playbooks de
provisionnement
# Il gère les opérations courantes de post-provisionnement

- include_tasks: task_notify_ocs.yaml
```

Le `task_notify_ocs.yaml` pourrait contenir :

```
- name: Notifier l'OCS de l'achèvement du provisionnement
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "APIerSv1.ReloadCache",
        "params": [{
          "ArgsCache": "*all"
        }]
      }
    status_code: 200
  ignore_errors: true
```

# Opérations Courantes

## Travailler avec l'Inventaire

Récupérer les détails de l'inventaire :

```
- name: Obtenir l'ID d'inventaire de la carte SIM
  set_fact:
    sim_inventory_id: "{{ hostvars[inventory_hostname]['SIM Card']
| int }}"
  when: "'SIM Card' in hostvars[inventory_hostname]"

- name: Obtenir les détails de la carte SIM
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ sim_inventory_id }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
    return_content: yes
  register: sim_response

- name: Extraire les détails de la SIM
  set_fact:
    iccid: "{{ sim_response.json.iccid }}"
    imsi: "{{ sim_response.json.imsi }}"
    ki: "{{ sim_response.json.ki }}"
```

Assigner l'inventaire au client :

```
- name: Assigner la SIM au client
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ sim_inventory_id }}"
    method: PATCH
    headers:
      Authorization: "Bearer {{ access_token }}"
    body_format: json
    body:
      {
        "customer_id": {{ customer_id }},
        "service_id": {{ service_id }},
        "item_state": "Assigned"
      }
    status_code: 200
```

## Opérations de Date et Heure

Obtenir la date/heure actuelle :

```
- name: Obtenir la date et l'heure actuelles au format ISO 8601
  command: date --utc +%Y-%m-%dT%H:%M:%S%z
  register: current_date_time

- name: Obtenir uniquement la date d'aujourd'hui
  set_fact:
    today: "{{ lookup('pipe', 'date +%Y-%m-%d') }}"
```

Calculer des dates futures :

```
- name: Calculer la date d'expiration dans 30 jours
  command: "date --utc +%Y-%m-%dT%H:%M:%S%z -d '+30 days'"
  register: expiry_date

- name: Calculer la date dans 90 jours
  command: "date --utc +%Y-%m-%d -d '+{{ days }} days'"
  register: future_date
  vars:
    days: 90
```

## Génération de Valeurs Aléatoires

UUIDs et identifiants :

```
- name: Générer UUID
  set_fact:
    uuid: "{{ 99999999 | random | to_uuid }}"

- name: Générer identifiant de service
  set_fact:
    service_uuid: "SVC_{{ uuid[0:8] }}"
```

Mots de passe aléatoires :

```
- name: Générer un mot de passe sécurisé
  set_fact:
    password: "{{ lookup('pipe', 'cat /dev/urandom | tr -dc a-zA-
Z0-9 | head -c 16') }}"
```

Phrases de passe mémorables :

```
- name: Définir la liste de mots
  set_fact:
    words:
      - alpha
      - bravo
      - charlie
      - delta
      - echo

- name: Générer une phrase de passe
  set_fact:
    word: "{{ words | random }}"
    number: "{{ 99999 | random(start=10000) }}"

- name: Combiner en phrase de passe
  set_fact:
    passphrase: "{{ word }}{{ number }}"
```

## Travailler avec CGRateS/OCS

Créer des comptes :

```
- name: Créer un compte de facturation
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV2.SetAccount",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Account": "{{ service_uuid }}",
          "ActionPlanIds": [],
          "ActionPlansOverwrite": true,
          "ExtraOptions": {
            "AllowNegative": false,
            "Disabled": false
          },
          "ReloadScheduler": true
        }]
      }
    status_code: 200
  register: account_response
```

Ajouter des soldes :

```
- name: Ajouter un solde de données
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.AddBalance",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Account": "{{ service_uuid }}",
          "BalanceType": "*data",
          "Categories": "*any",
          "Balance": {
            "ID": "Paquet de Données",
            "Value": 10737418240,
            "ExpiryTime": "+720h",
            "Weight": 10
          }
        }]
      }
    status_code: 200
```

Exécuter des actions :

```
- name: Exécuter une action de facturation
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "APIerSv1.ExecuteAction",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Account": "{{ service_uuid }}",
          "ActionsId": "Action_Standard_Charge"
        }]
      }
    status_code: 200
```

Obtenir des informations sur le compte :

```
- name: Obtenir les détails du compte
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV2.GetAccount",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Account": "{{ service_uuid }}"
        }]
      }
    status_code: 200
  register: account_info
```

Travailler avec les Profils d'Attributs :

```
- name: Obtenir le Profil d'Attribut
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "APIerSv1.GetAttributeProfile",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "ID": "ATTR_{{ service_uuid }}"
        }]
      }
    return_content: yes
    status_code: 200
  register: attr_response
  ignore_errors: true

- name: Extraire la valeur d'attribut
  set_fact:
    phone_number: "{{ attr_response.json.result.Attributes |
json_query(\"[?Path=='*req.PhoneNumber'].Value[0].Rules\") | first
}}"
  when: attr_response is defined
```

## Logique Conditionnelle

Vérifier si les variables existent :

```
- name: Utiliser une valeur personnalisée ou par défaut
  set_fact:
    monthly_cost: "{{ custom_cost | default(50.00) }}"

- name: Exécuter uniquement si la variable est définie
  debug:
    msg: "Le UUID du service est {{ service_uuid }}"
  when: service_uuid is defined
```

Conditions booléennes :

```yaml
- name: Provisionner l'équipement
  include_tasks: configure_cpe.yaml
  when: provision_cpe | default(false) | bool

- name: Passer si déprovisionnement
  assert:
    that:
      - action != "deprovision"
  when: action is defined
```

Conditions multiples :

```yaml
- name: Tâche conditionnelle complexe
  uri:
    url: "{{ endpoint }}"
    method: POST
  when:
    - service_uuid is defined
    - customer_id is defined
    - action != "deprovision"
    - enable_feature | default(true) | bool
```

## Boucles et Itération

Boucles simples :

```yaml
- name: Créer plusieurs soldes
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.AddBalance",
        "params": [{
          "Account": "{{ service_uuid }}",
          "BalanceType": "{{ item.type }}",
          "Balance": {
            "Value": "{{ item.value }}"
          }
        }]
      }
  loop:
    - { type: "*voice", value: 3600 }
    - { type: "*data", value: 10737418240 }
    - { type: "*sms", value: 100 }
```

Boucle sur les réponses API :

```yaml
- name: Obtenir tous les sites clients
  uri:
    url: "{{ crm_config.crm.base_url }}/crm/site/customer_id/{{
customer_id }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
  register: sites_response

- name: Configurer l'équipement à chaque site
  debug:
    msg: "Configuration du site à {{ item.address_line_1 }}"
  loop: "{{ sites_response.json }}"
```

## Gestion des Erreurs

Utiliser ignore_errors :

```yaml
- name: Notification SMS optionnelle
  uri:
    url: "http://sms-gateway/send"
    method: POST
    body: {...}
  ignore_errors: true
```

Assertions pour validation :

```yaml
- name: Vérifier la réponse de l'API
  assert:
    that:
      - response.status == 200
      - response.json.result == "OK"
    fail_msg: "L'appel API a échoué : {{ response.json }}"
```

Gestion conditionnelle des erreurs :

```yaml
- name: Essayer d'obtenir le service existant
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/service/service_uuid/{{ service_uuid }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
  register: service_lookup
  failed_when: false

- name: Créer le service s'il n'existe pas
  uri:
    url: "{{ crm_config.crm.base_url }}/crm/service/"
    method: PUT
    body: {...}
  when: service_lookup.status == 404
```

# Meilleures Pratiques

## Nommage des Variables

Utiliser des noms descriptifs et cohérents :

```yaml
# Bon
service_uuid: "SVC_12345"
customer_name: "John Smith"
monthly_cost: 49.99

# Mauvais
svc: "SVC_12345"
name: "John Smith"
cost: 49.99
```

Préfixer les variables par source :

```yaml
api_response_customer: {...}
api_response_product: {...}
cgr_account_info: {...}
```

## Débogage

Imprimer des variables pour le dépannage :

```yaml
- name: Imprimer toutes les variables
  debug:
    var: hostvars[inventory_hostname]

- name: Imprimer une variable spécifique
  debug:
    msg: "UUID du service : {{ service_uuid }}"

- name: Imprimer la réponse de l'API
  debug:
    var: api_response_product.json
```

## Validation

Toujours valider les réponses API critiques :

```
- name: Créer un compte
  uri:
    url: "{{ billing_endpoint }}"
    method: POST
    body: {...}
  register: response

- name: Vérifier la création du compte
  assert:
    that:
      - response.status == 200
      - response.json.result == "OK"
    fail_msg: "Échec de la création du compte : {{ response.json
}}"
```

## Idempotence

Concevoir des tâches pour être réexécutables en toute sécurité :

```
# Vérifier si la ressource existe d'abord
- name: Vérifier si le compte existe
  uri:
    url: "{{ ocs_endpoint }}/get_account"
    method: POST
    body: {"Account": "{{ service_uuid }}"}
  register: account_check
  failed_when: false

# Créer uniquement s'il n'existe pas
- name: Créer un compte
  uri:
    url: "{{ ocs_endpoint }}/create_account"
    method: POST
    body: {...}
  when: account_check.status == 404
```

## Sécurité

Ne jamais coder en dur les identifiants :

```
# Mauvais
mailjet_api_key: "abc123def456"

# Bon - utiliser des variables d'environnement
mailjet_api_key: "{{ lookup('env', 'MAILJET_API_KEY') }}"

# Bon - utiliser un fichier de configuration
mailjet_api_key: "{{ crm_config.email.api_key }}"
```

Utiliser toujours HTTPS et l'authentification :

```
- name: Appeler une API externe
  uri:
    url: "https://api.example.com/endpoint"
    method: POST
    headers:
      Authorization: "Bearer {{ access_token }}"
    validate_certs: yes
```

## Documentation

Documenter la logique complexe :

```
# Calculer le coût au prorata pour un mois partiel
# Si le client s'inscrit le 15 et que la facturation est le 1er,
# facturer 50% du coût mensuel pour les jours restants
- name: Calculer les jours jusqu'à la fin du mois
  command: "date -d 'last day of this month' +%d"
  register: days_in_month

- name: Obtenir le jour actuel
  command: "date +%d"
  register: current_day

- name: Calculer le montant au prorata
  set_fact:
    days_remaining: "{{ (days_in_month.stdout | int) -
(current_day.stdout | int) }}"
    pro_rata_cost: "{{ (monthly_cost | float) * (days_remaining |
float) / (days_in_month.stdout | float) }}"
```

# Tester les Playbooks

## Approche de Test

1. **Exécution à Sec d'abord** : Tester avec des systèmes non productifs
2. **Vérifier les Variables** : Utiliser des tâches de débogage pour confirmer que toutes les variables requises sont présentes
3. **Vérifier les Réponses** : Valider les réponses API avant de continuer
4. **Test de Rollback** : Échouer intentionnellement des tâches pour vérifier que les blocs de secours fonctionnent
5. **Test de Déprovisionnement** : Tester avec `action: "deprovision"` pour vérifier le nettoyage

Exemple de playbook de test :

```yaml
- name: Tester le Provisionnement de Service
  hosts: localhost
  gather_facts: no

  tasks:
    - name: Vérifier les variables requises
      assert:
        that:
          - product_id is defined
          - customer_id is defined
          - access_token is defined
        fail_msg: "Variables requises manquantes"

    - name: Tester la connectivité API
      uri:
        url: "http://localhost:5000/crm/health"
        method: GET
      register: health_check

    - name: Vérifier le contrôle de santé
      assert:
        that:
          - health_check.status == 200
```

## Pièges Communs

**Conversions de type manquantes :**

```yaml
# Mauvais - peut être une chaîne
customer_id: "{{ customer_id }}"

# Correct - s'assurer qu'il est entier
customer_id: {{ customer_id | int }}
```

**Ne pas gérer les variables indéfinies :**

```yaml
# Mauvais - échoue si non défini
service_uuid: "{{ service_uuid }}"

# Correct - fournir une valeur par défaut
service_uuid: "{{ service_uuid | default('') }}"
```

**Oublier la validation :**

```yaml
# Mauvais - ne vérifie pas la réponse
- name: Créer un compte
  uri: ...
  register: response

# Correct - valider la réponse
- name: Créer un compte
  uri: ...
  register: response

- name: Vérifier la création
  assert:
    that:
      - response.json.result == "OK"
```

# Flux de Travail de Provisionnement

En général, le personnel d'Omnitouch travaillera avec le client pour :

1. Définir les exigences du produit
2. Développer les playbooks Ansible nécessaires pour automatiser le processus de provisionnement
3. Tester les playbooks dans un environnement de staging
4. Déployer en production

Cela garantit que chaque service est déployé de manière cohérente et fiable, réduisant le risque d'erreurs et garantissant que toutes les étapes nécessaires sont complétées dans le bon ordre.

# Variables Ansible

Les variables passées aux playbooks Ansible incluent :

**Variables de Produit**
Dérivées des configurations de produit OmniCRM et définissent comment le service doit être configuré.

**Variables d'Inventaire**
Sélectionnées dans l'inventaire, elles incluent des éléments tels que des modems, des cartes SIM, des blocs d'adresses IP ou des numéros de téléphone nécessaires au provisionnement.

**Variables Système**
Ajoutées automatiquement par OmniCRM :

- `product_id` - Le produit étant provisionné
- `customer_id` - Le client recevant le service
- `service_id` - Le service étant modifié (pour les recharges/changements)
- `access_token` - JWT pour l'authentification API

# Déprovisionnement

Lorsqu'un service n'est plus nécessaire, les **Playbooks Ansible** sont également utilisés pour déprovisionner le service en utilisant le modèle de bloc `rescue`. Cela :

- Supprime toutes les configurations
- Libère l'inventaire de retour dans le pool
- Supprime les comptes de facturation
- Assure que le système reste propre

## Méthodes de Déprovisionnement

Il existe deux manières principales de déclencher le déprovisionnement :

**1. Rollback Automatique (Échec de Provisionnement)**
Lorsque toute tâche dans le bloc principal de provisionnement échoue, la section de secours s'exécute automatiquement pour nettoyer les modifications partielles.

**2. Déprovisionnement Manuel**
Définir `action: "deprovision"` lors de l'exécution d'un playbook déclenche le bloc de secours intentionnellement pour supprimer un service.

# Modèle de Déprovisionnement

Tous les playbooks OmniCRM suivent cette structure pour un nettoyage sûr :

```yaml
- name: Playbook de Provisionnement de Service
  hosts: localhost
  gather_facts: no
  become: False

  tasks:
    - name: Bloc principal
      block:
        # Toutes les tâches de provisionnement vont ici
        - name: Créer un compte OCS
          uri: ...

        - name: Créer un enregistrement de service
          uri: ...
          register: service_creation_response

        - name: Ajouter une transaction
          uri: ...

      rescue:
        # Tâches de déprovisionnement/rollback vont ici
        - name: Imprimer toutes les vars pour le débogage
          debug:
            var: hostvars[inventory_hostname]

        # Les tâches de nettoyage s'exécutent dans l'ordre inverse
        - name: Supprimer le compte dans l'OCS
          uri: ...

        - name: Supprimer le Service du CRM
          uri: ...

        - name: Échouer si ce n'est pas un déprovisionnement
intentionnel
          assert:
            that:
              - action == "deprovision"
```

## Exemple Complet de Déprovisionnement

Voici un exemple complet du `play_simple_service.yaml` :

```
rescue:
  # 1. Informations de débogage
  - name: Imprimer toutes les vars pour Déprovisionnement/Rollback
    debug:
      var: hostvars[inventory_hostname]

  - name: Obtenir la date d'aujourd'hui
    set_fact:
      today: "{{ lookup('pipe', 'date +%Y-%m-%d') }}"

  # 2. Obtenir les informations du service si disponibles
  - name: Essayer d'obtenir les informations du Service depuis
l'API CRM pour Déprovisionner
    uri:
      url: "http://localhost:5000/crm/service/service_id/{{
service_creation_response.json.service_id }}"
      method: GET
      headers:
        Authorization: "Bearer {{ access_token }}"
      return_content: yes
    register: api_response_service
    ignore_errors: True
    when: service_creation_response is defined and
service_creation_response.json is defined and
service_creation_response.json.service_id is defined

  - name: Imprimer api_response_service
    debug:
      var: api_response_service
    when: api_response_service is defined

  # 3. Définir service_uuid pour le nettoyage
  - name: Définir les faits service_uuid pour Déprovisionnement
    set_fact:
      service_uuid: "{{ api_response_service.json.service_uuid }}"
    when:
      - service_uuid is not defined
      - api_response_service is defined
      - api_response_service.json is defined

  # 4. Supprimer le compte OCS
  - name: Supprimer le compte dans l'OCS
    uri:
```

```
      url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
      method: POST
      body_format: json
      headers:
        Content-Type: "application/json"
        Authorization: "Bearer {{ access_token }}"
      body:
        {
          "method": "ApierV2.RemoveAccount",
          "params": [{
            "Tenant": "{{ crm_config.ocs.ocsTenant }}",
            "Account": "{{ service_uuid }}",
            "ReloadScheduler": true
          }]
        }
      status_code: 200
    register: response
    ignore_errors: True
    when: service_uuid is defined

  - name: Imprimer la réponse de la suppression du compte OCS
    debug:
      var: response
    when: response is defined

  # 5. Supprimer l'enregistrement de service du CRM
  - name: Supprimer le Service du CRM s'il a été créé
    uri:
      url: "http://localhost:5000/crm/service/service_id/{{
service_creation_response.json.service_id }}"
      method: DELETE
      headers:
        Authorization: "Bearer {{ access_token }}"
      status_code: 200
    register: delete_service_response
    ignore_errors: True
    when: service_creation_response is defined and
service_creation_response.json is defined and
service_creation_response.json.service_id is defined

  - name: Imprimer la réponse de suppression du service
    debug:
      var: delete_service_response
    when: delete_service_response is defined
```

```
  # 6. Déterminer si cela était intentionnel ou un échec
  - name: Définir le statut sur "Succès" si Déprovisionnement
Manuel / Échouer si échec de provisionnement
    assert:
      that:
        - action == "deprovision"
      fail_msg: "Le provisionnement a échoué et a été annulé"
      success_msg: "Service déprovisionné avec succès"
```

## Concepts Clés de Déprovisionnement

**ignore_errors: True**
La plupart des tâches de nettoyage utilisent `ignore_errors: True` car les ressources peuvent ne pas exister (par exemple, si le provisionnement a échoué avant leur création).

**Exécution Conditionnelle**
Utilisez des clauses `when` pour nettoyer uniquement les ressources qui ont été créées :

```
when: service_creation_response is defined
```

**Ordre Inversé**
Le nettoyage se fait dans l'ordre inverse de la création :

1. Supprimer d'abord les ressources dépendantes (transactions, affectations d'inventaire)
2. Supprimer l'enregistrement de service
3. Supprimer le compte OCS/facturation
4. Libérer toutes les ressources retenues

**Assertion Finale**
L'assertion finale distingue entre :

- **Déprovisionnement Intentionnel** (`action == "deprovision"`) → Succès
- **Échec de Provisionnement** (action non définie) → Échec

# Déprovisionnement de Différents Types de Ressources

**Comptes OCS/CGRateS :**

```
- name: Supprimer le compte dans l'OCS
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV2.RemoveAccount",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Account": "{{ service_uuid }}",
          "ReloadScheduler": true
        }]
      }
    status_code: 200
  ignore_errors: True
  when: service_uuid is defined
```

**Enregistrements de Service CRM :**

```
- name: Supprimer le Service du CRM
  uri:
    url: "http://localhost:5000/crm/service/service_id/{{
service_id }}"
    method: DELETE
    headers:
      Authorization: "Bearer {{ access_token }}"
    status_code: 200
  ignore_errors: True
  when: service_id is defined
```

**Éléments d'Inventaire (Retour au Pool) :**

```
- name: Libérer la carte SIM dans le pool d'inventaire
  uri:
    url: "http://localhost:5000/crm/inventory/inventory_id/{{
sim_inventory_id }}"
    method: PATCH
    headers:
      Authorization: "Bearer {{ access_token }}"
    body_format: json
    body:
      {
        "customer_id": null,
        "service_id": null,
        "item_state": "Available"
      }
    status_code: 200
  ignore_errors: True
  when: sim_inventory_id is defined
```

**ActionPlans (Charges Récurrentes) :**

```
- name: Supprimer ActionPlan du compte
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.RemoveActionPlan",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Id": "ActionPlan_Monthly_{{ service_uuid }}_{{
product_id }}"
        }]
      }
    status_code: 200
  ignore_errors: True
```

**Autorisations de Paiement :**

```
- name: Libérer l'autorisation de paiement
  uri:
    url: "http://localhost:5000/crm/payments/release/{{
authorization_id }}"
    method: POST
    headers:
      Authorization: "Bearer {{ access_token }}"
    body_format: json
    body:
      {
        "metadata": {
          "release_reason": "provisioning_failed"
        }
      }
    return_content: yes
  ignore_errors: True
  when: authorization_id is defined
```

## Flux de Travail de Déprovisionnement Manuel

Pour déprovisionner manuellement un service :

1. Identifier le service à déprovisionner
2. Exécuter le playbook de provisionnement d'origine avec `action: "deprovision"`
3. Le playbook entre immédiatement dans le bloc de secours
4. Toutes les tâches de nettoyage s'exécutent
5. Le service est supprimé proprement

**Exemple d'appel API :**

```
POST /crm/provision/run_playbook
{
  "product_id": 123,
  "customer_id": 456,
  "service_id": 789,
  "action": "deprovision"
}
```

# Scénarios de Nettoyage Partiel

### Scénario 1 : Compte OCS Créé, Échec de Création de Service

- Le compte OCS existe
- L'enregistrement de service n'existe PAS
- Le bloc de secours supprime le compte OCS
- Aucun service à supprimer (sauté en toute sécurité avec une condition `when`)

### Scénario 2 : Service Créé, Échec de Transaction

- Le compte OCS existe
- L'enregistrement de service existe
- La transaction n'existe PAS
- Le bloc de secours supprime à la fois le compte OCS et le service
- Aucune transaction à annuler (jamais créée)

### Scénario 3 : Provisionnement Complet, Échec de Capture de Paiement

- Le compte OCS existe
- L'enregistrement de service existe
- Paiement autorisé mais PAS capturé
- Le bloc de secours :
  - Libère l'autorisation de paiement (le client n'est pas facturé)
  - Supprime l'enregistrement de service
  - Supprime le compte OCS

# Meilleures Pratiques pour le Déprovisionnement

### 1. Utiliser ignore_errors généreusement

Le nettoyage doit être tolérant. Ne pas échouer si une ressource n'existe pas.

### 2. Vérifier l'existence des ressources avec des clauses when

N'essayez de nettoyer que si la ressource a été créée :

```
when: service_creation_response is defined
```

**3. Imprimer des informations de débogage**
Inclure toujours des tâches de débogage pour aider à résoudre les problèmes de provisions échouées :

```
- name: Imprimer toutes les vars pour le débogage
  debug:
    var: hostvars[inventory_hostname]
```

**4. Nettoyer dans l'ordre inverse des dépendances**
Supprimer les enfants avant les parents :

- Transactions avant services
- Services avant comptes OCS
- Affectations d'inventaire avant libérations d'inventaire

**5. Gérer les autorisations de paiement**
Toujours libérer les autorisations de paiement dans les blocs de secours pour éviter de facturer les clients pour des provisions échouées.

**6. Recharger les planificateurs après nettoyage**
Lors de la suppression des ressources OCS, inclure `ReloadScheduler: true` pour garantir que CGRateS se met à jour immédiatement.

# Déprovisionnement vs Suppression

**Déprovisionnement** (via bloc de secours) :

- Supprime le service de tous les systèmes
- Libère l'inventaire
- Annule les charges récurrentes
- Laisse une trace d'audit
- Approche recommandée

**Suppression Directe** (via API) :

- Supprime uniquement l'enregistrement CRM
- NE nettoie PAS le compte OCS
- NE libère PAS l'inventaire
- Peut laisser des ressources orphelines
- Pas recommandé pour la production

# Rollback et Gestion des Erreurs

La fonction **block/rescue** d'Ansible est utilisée lors du provisionnement et du déprovisionnement pour gérer les erreurs de manière élégante. Si une tâche échoue à tout moment pendant le provisionnement, la section de secours annule automatiquement les modifications pour revenir à un état cohérent. Cela garantit la fiabilité et réduit le risque de déploiements partiels ou échoués.

## Exemple de Gestion des Erreurs

```
- name: Bloc principal
  block:
    - name: Étape 1 : Créer un compte
      uri: ...
      register: response

    - name: Vérifier l'étape 1
      assert:
        that:
          - response.status == 200

    - name: Étape 2 : Créer un service
      uri: ...

  rescue:
    # Si une assertion échoue ou qu'une tâche échoue :
    # 1. L'exécution saute au bloc de secours
    # 2. Les tâches de nettoyage s'exécutent
    # 3. Le playbook échoue avec un message d'erreur
    - name: Nettoyer les modifications partielles
      uri: ...
```

Pour des détails complets sur le système de provisionnement, les flux de travail et l'authentification, voir `concepts_provisioning`.

# OmniCRM API

Toutes les fonctions d'OmniCRM sont accessibles via l'API - Il n'y a pas de fonctionnalité qui soit uniquement disponible dans l'interface utilisateur.

Cela vous permet d'intégrer OmniCRM avec d'autres systèmes ou d'automatiser des tâches.

L'API est une API RESTful, et est sécurisée en utilisant plusieurs méthodes d'authentification, y compris les tokens JWT, les clés API et le filtrage par adresse IP.

L'API est documentée à l'aide de Swagger, un outil qui permet une lecture, une compréhension et un test faciles des fonctionnalités de l'API.

La documentation de l'API est disponible à l'URL suivante :

> <https://yourcrm/crm/docs/>

# Méthodes d'Authentification

OmniCRM prend en charge trois méthodes d'authentification, chacune conçue pour différents cas d'utilisation :

1. **Tokens JWT Bearer** - Pour les sessions utilisateur interactives (UI Web, applications mobiles)
2. **Clés API** - Pour les intégrations serveur à serveur et les scripts d'automatisation
3. **Filtrage par IP** - Pour les systèmes internes de confiance (serveurs de provisionnement, outils de surveillance)

## Authentification par Token JWT Bearer

C'est la méthode d'authentification principale pour les sessions utilisateur. Les utilisateurs se connectent avec un e-mail et un mot de passe, reçoivent un token JWT, et l'utilisent pour les requêtes suivantes.

**Cas d'utilisation :**

- Authentification de l'UI Web
- Authentification d'application mobile
- Accès programmatique à court terme

**Comment s'authentifier :**

Pour se connecter, envoyez un corps JSON avec la structure suivante à `/crm/auth/login` en tant que requête POST :

```
{
    "email": "youruser@yourdomain.com",
    "password": "yourpassword"
}
```

L'API renverra un objet JSON contenant un champ `token`, qui est utilisé pour authentifier toutes les requêtes futures. De plus, la réponse inclut un `refresh_token` qui peut être utilisé pour rafraîchir le token lorsqu'il expire, ainsi que les permissions et rôles de l'utilisateur.

Vous pouvez tester cela depuis la page Swagger en sélectionnant le point de terminaison `/auth/login`, en remplissant votre nom d'utilisateur et votre mot de passe, et en cliquant sur le bouton `Try it out`.

Pour autoriser la session, copiez la valeur du token et cliquez sur le bouton "Authorize" en haut à droite de la page Swagger. Collez le token dans le champ "Value", préfixé par `Bearer` et cliquez sur "Authorize".

Maintenant, toutes les requêtes suivantes seront authentifiées avec ce token.

## Authentification par Clé API

Les clés API fournissent une authentification sécurisée et à long terme pour les intégrations serveur à serveur et les scripts d'automatisation sans nécessiter de mots de passe utilisateur.

**Cas d'utilisation :**

- Systèmes de provisionnement automatisés
- Outils de surveillance et d'alerte
- Intégration avec des systèmes externes
- Tâches planifiées et jobs cron

**Comment fonctionnent les Clés API :**

Les clés API sont configurées dans le fichier `crm_config.yaml` et sont associées à des rôles et permissions spécifiques. Chaque clé API est une chaîne aléatoire sécurisée (minimum 32 caractères) qui authentifie les requêtes lorsqu'elle est passée dans l'en-tête `X-API-KEY`.

**Configuration des Clés API :**

Les clés API doivent être ajoutées à `crm_config.yaml` par un administrateur ayant un accès serveur :

```
api_keys:
  your-secure-api-key-here-minimum-32-chars:
    roles:
      - admin
    description: "Système d'automatisation de provisionnement"
  another-api-key-for-monitoring-system:
    roles:
      - view_customer
      - view_service
    description: "Surveillance et alerte"
```

**Utilisation des Clés API :**

Incluez la clé API dans l'en-tête `X-API-KEY` de vos requêtes :

```
curl -X GET "https://yourcrm.com/crm/customers" \
  -H "X-API-KEY: your-secure-api-key-here-minimum-32-chars"
```

**Exemple Python :**

```
import requests

crm_url = 'https://yourcrm.com'
api_key = 'your-secure-api-key-here-minimum-32-chars'

headers = {
    "Content-Type": "application/json",
    "X-API-KEY": api_key
}

# Obtenir les Clients
response = requests.get(crm_url + '/crm/customers',
headers=headers)
for customer in response.json()['data']:
    print(customer)
```

**Meilleures Pratiques :**

- Générer des clés API en utilisant des générateurs aléatoires sécurisés sur le plan cryptographique (`openssl rand -base64 48`)
- Utiliser des clés API différentes pour différents systèmes
- Documenter l'objectif de chaque clé API dans le champ `description`
- Faire tourner les clés API périodiquement
- Ne jamais commettre les clés API dans le contrôle de version
- Assigner les permissions minimales nécessaires à chaque clé API

# Authentification par Filtrage IP

Le filtrage IP permet à des adresses IP spécifiques d'accéder à l'API sans authentification. Cela est utile pour les systèmes internes de confiance sur des réseaux privés.

**Cas d'utilisation :**

- Serveurs de provisionnement internes
- Systèmes de surveillance réseau sur des VLAN de gestion
- Playbooks Ansible exécutés sur une infrastructure contrôlée

**Configuration de la Liste Blanche IP :**

Ajoutez des adresses IP de confiance à `crm_config.yaml` :

```
ip_whitelist:
  - 192.168.1.100
  - 10.0.0.0/24
  - 172.16.50.10
```

**Considérations de Sécurité :**

- N'utiliser le filtrage IP que sur des réseaux privés et sécurisés
- Ne jamais mettre sur liste blanche des adresses IP publiques
- Utiliser les plages IP les plus spécifiques possibles
- Documenter pourquoi chaque IP est sur liste blanche

- Auditer régulièrement les IP sur liste blanche

# Exemples d'Appels API avec Python

Voici un exemple de la façon de se connecter et de récupérer une liste de clients en utilisant l'authentification par token JWT :

```python
import requests

crm_url = 'https://yourcrm.com'
session = requests.Session()

print("Provisioning data to server: " + str(crm_url))

headers = {
    "Content-Type": "application/json"
}

# Obtenir le Token d'Auth
response = session.post(crm_url + '/crm/auth/login', json={
    "email": "youruser@yourdomain.com",
    "password": "yourpassword"
}, headers=headers)

print(response.status_code)
print(response.json())
assert response.status_code == 200

headers['Authorization'] = 'Bearer ' + response.json()['token']
print("Authentifié avec succès au CRM")

# Obtenir les Clients
response = session.get(crm_url + '/crm/customers',
headers=headers)
for customer in response.json()['data']:
    print(customer)
```

# Surveillance de la Performance de l'API

Toutes les requêtes API sont automatiquement suivies avec des métriques complètes. Pour des détails sur les métriques de performance de l'API, y compris les taux de requêtes, les temps de réponse, les taux d'erreur et les statistiques spécifiques aux points de terminaison, voir Monitoring & Metrics.

Le point de terminaison des métriques est disponible à `/crm/metrics` au format Prometheus pour la surveillance et l'alerte.

# Documentation Connexe

- Architecture du Système - Conception globale du système et relations entre les composants
- Monitoring & Metrics - Référence complète des métriques incluant les métriques API, base de données et d'intégration
- Flux d'Authentification - Flux d'authentification et d'autorisation détaillés
- RBAC - Contrôle d'accès basé sur les rôles et permissions

# Notes sur le Service / Produit CRM

> **REMARQUE**
>
> Pour un guide complet de bout en bout couvrant la définition des produits, la fourniture, les options supplémentaires et la déprovisionnement avec des exemples Ansible détaillés et une stratégie de tarification, consultez le Guide complet du cycle de vie des produits.
>
> Pour des informations détaillées sur la fourniture de services mobiles avec des cartes SIM, voir Fourniture de cartes SIM.

## Aperçu des Produits et Services

## Produit (Élément de Menu) :

Un Produit est comme un plat spécifique sur un menu de restaurant, tel qu'un "Spaghetti Carbonara."

Il a une description claire, une liste d'ingrédients (comme des pâtes, de la crème, des œufs, du fromage et du bacon), et un prix.

Dans OmniCRM, un Produit contient également les détails de ce qui est inclus : caractéristiques, spécifications et tarification.

Souvent, les clients peuvent vouloir des modifications, comme "sans oignons" ou "ajouter du fromage supplémentaire" à leur repas. Dans OmniCRM, cela correspond à la personnalisation d'un service avant sa création. Le niveau de personnalisation ou de modifications d'un service est à votre (l'opérateur) définition.

Dans OmniCRM, les clients ou le personnel peuvent modifier un Produit pour mieux répondre aux besoins d'un client spécifique, comme améliorer leur vitesse Internet ou ajouter des fonctionnalités spécifiques. Cette personnalisation est reflétée dans le Service spécifique fourni.

Un produit est essentiellement une offre que les clients peuvent choisir de commander comme lire et choisir un plat sur le menu.

# Catalogue de Produits (Menu du Restaurant) :

Le Catalogue de Produits est comme l'ensemble du menu d'un restaurant, qui liste tous les plats disponibles, des entrées aux desserts.

C'est la collection complète de tout ce que le restaurant (ou dans votre cas, le Fournisseur de Services) a à offrir.

Dans le contexte commercial, le Catalogue de Produits fournit aux clients tous les Produits disponibles, afin qu'ils puissent choisir celui qui répond le mieux à leurs besoins.

# Service (Plat Préparé) :

Lorsque qu'un client commande un article du menu, le plat est préparé dans la cuisine. Cela équivaut à créer un Service à partir d'un Produit.

Dans OmniCRM, lorsque qu'un client sélectionne un Produit, une instance de ce Produit est créée et livrée sous forme de Service.

Il est personnalisé et préparé spécifiquement pour ce client, tout comme un repas préparé pour un client.

Par exemple, lorsque quelqu'un sélectionne le "Plan Internet Bronze" dans le Catalogue de Produits, le système de fourniture "prépare" une instance de ce plan à partir des ingrédients (adresses IP, modems et ports) : c'est-à-dire, active le plan et le livre au client spécifique.

# Produits Groupés (Repas Combinés) :

Le Catalogue de Produits peut également proposer des forfaits, comme un repas combiné qui comprend une entrée, un plat principal et un dessert ensemble pour un prix spécial.

Dans OmniCRM, les Produits groupés combinent plusieurs Produits individuels en un seul package pratique, comme un "Forfait Essentiels à Domicile" qui inclut les services Internet, câble et téléphone à un tarif réduit.

Une fois sélectionné, ce forfait est transformé en plusieurs Services adaptés au client.

# Définitions de Produit

Un produit est un modèle qui est utilisé pour créer un service / option supplémentaire / remise / ajout, etc.

Dans la définition, nous incluons :

- Informations sur le produit (caractéristiques, inclusions, T&Cs, durée du contrat, icône, etc.) qui sont affichées à l'utilisateur du CRM (Client ou *Personnel*).

- La logique commerciale concernant qui peut acheter le produit (*Commercial* ou *Résidentiel*), s'il dépend d'un service parent provisionné (comme les options supplémentaires mobiles uniquement disponibles pour les clients ayant un service mobile), s'il peut être commandé directement par un client via l'auto-service ou uniquement par un agent de service

client, et quand le produit peut être acheté (permettant à un produit d'être disponible uniquement pour une période définie).

- Lorsque des éléments d'inventaire doivent être inclus (comme des modems ou des cartes SIM), ceux-ci sont spécifiés comme Liste d'Éléments d'Inventaire, par exemple le service ci-dessous nécessite une carte SIM et un numéro de téléphone à attribuer :

  `['Carte SIM', 'Numéro de Téléphone']` Ceux-ci correspondent aux Éléments d'Inventaire définis dans le CRM.

- Référence à un Playbook Ansible pour provisionner le service Play de Provisionnement ainsi que les variables à transmettre à Ansible. Ces variables à transmettre sont magiques, en ce sens qu'elles peuvent être des variables comme `service_id` qui sont définies par le produit auquel nous l'ajoutons, ou elles peuvent être comme `ICCID` & `MSISDN` où nous avons sélectionné des éléments d'inventaire qui sont transmis lors de l'attribution de l'inventaire. Le regroupement est géré dans le play de provisionnement pour contenir plusieurs services, par exemple un produit groupé Internet à domicile, TV & Voix peut provisionner un service pour chacun.

# Catégories de Produits et Types de Services

Les produits utilisent deux champs de classification pour aider à organiser et filtrer les offres :

## Catégories de Produits

Le champ `category` contrôle où les produits sont affichés dans l'interface utilisateur. Les valeurs courantes incluent :

- **standalone** - Affiché comme une option de service de base lors de la création d'un nouveau service
- **addon** - Affiché lors de l'ajout à un service existant
- **bundle** - Affiché comme une option de service groupé (provisionné comme un ajout aux services existants)
- **promo** - Offres promotionnelles spéciales

Ces catégories sont purement organisationnelles et ne dictent pas ce qui est provisionné. Le comportement de provisionnement r��el est entièrement déterminé par le playbook Ansible référencé dans `provisioning_play`.

Par exemple : - Un produit `standalone` crée généralement un nouvel objet de service

- Un produit `addon` ou `bundle` est généralement ajouté à un service existant - Mais cela dépend de l'implémenteur écrivant le playbook - vous pourriez créer plusieurs objets de service à partir d'un ajout, ou modifier des services existants à partir d'un produit autonome si nécessaire

La catégorie contrôle simplement le flux de l'interface utilisateur et où les clients/le personnel voient l'option de produit.

## Types de Services

Le champ `service_type` catégorise quel type de service est fourni.

Ceux-ci sont entièrement définis par l'utilisateur, mais les valeurs courantes incluent :

- **mobile** - Services de téléphonie mobile avec voix, SMS et données
- **fixed** - Services Internet fixes sans fil ou filaires
- **fixed-voice** - Services de voix fixe (VoIP, ligne fixe)
- **hotspot** - Dispositifs de point d'accès mobile ou de location
- **dongle** - Services de modem USB ou de dongle
- **voice** - Services uniquement vocaux
- **data** - Services uniquement de données

Comme les catégories, les types de services sont personnalisables en fonction de vos offres. Ils aident à :

- Filtrer quels ajouts s'appliquent à quels services de base
- Organiser les produits dans le portail client
- Correspondre aux exigences d'inventaire
- Déterminer les flux de travail de provisionnement

Exemple : Un client avec un service `mobile` peut voir des ajouts mobiles, tandis qu'un client avec un service `fixed` voit des ajouts de ligne fixe.

# Gestion des Produits

Les produits sont gérés via la page de Gestion des Produits, où vous pouvez voir, rechercher, filtrer et éditer tous les produits disponibles.

**Interface Modale de Produit**

Cliquer sur n'importe quel produit ouvre une interface modale à onglets améliorée qui organise tous les paramètres du produit en groupes logiques pour une navigation et une édition plus faciles.

La modale de gestion des produits comporte cinq onglets organisés :

1. □ **Informations de Base** - Informations de base sur le produit (nom, slug, catégorie, icône, caractéristiques, termes)
2. □ **Tarification** - Tous les champs liés aux coûts, y compris les coûts récurrents, les coûts d'installation et le pourcentage de taxe
3. ⚙ **Configuration** - Paramètres de renouvellement, types de clients et dépendances
4. □ **Provisionnement** - Configuration du playbook Ansible et exigences d'inventaire
5. □ **Disponibilité** - Plages de dates et horodatages système

**Organisation de l'Onglet Tarification :**

L'onglet Tarification regroupe les champs de coût en sections logiques :

- **Coûts Récurrents** - Coûts de détail et de gros mensuels côte à côte
- **Coûts d'Installation** - Frais d'activation ou d'installation uniques pour le détail et le gros
- **Taxe** - Configuration du pourcentage de taxe avec calcul automatique

**Fonctionnalités du Mode Édition :**

- **Sélecteur d'Icônes** - Rechercher et sélectionner visuellement des icônes FontAwesome
- **Sélecteur d'Éléments d'Inventaire** - Sélectionner parmi les types d'éléments d'inventaire disponibles
- **Sélecteur de Date/Heure** - Sélection facile des fenêtres de disponibilité
- **Formatage de Devise** - Préfixe automatique $ pour les champs de coût
- **Sélecteurs Déroulants** - Options prédéfinies pour les catégories et les champs booléens

**Sélecteur d'Icônes :**

Lors de l'édition du champ d'icône, une interface de sélection d'icônes recherchable apparaît vous permettant de parcourir et de sélectionner visuellement parmi des milliers d'icônes FontAwesome.

Fonctionnalités : * Rechercher des icônes par mot-clé (par exemple, "clé", "mobile", "wifi") * Prévisualiser l'apparence de l'icône en temps réel * Affiche le nom de classe de l'icône pour référence * Sélection déroulante pour un accès rapide

**Onglet Configuration :**

L'onglet Configuration organise les paramètres de comportement du produit en groupes logiques.

**Sections de Configuration :**

- **Paramètres de Renouvellement :**

    - Renouvellement Automatique - Comportement par défaut de renouvellement (Demander/Oui/Non)
    - Autoriser le Renouvellement Automatique - Si les clients peuvent activer le renouvellement automatique
    - Jours de Contrat - Durée minimale du contrat (par exemple, 30 pour mensuel, 365 pour annuel)
- **Types de Clients :**
    - Résidentiel - Disponible pour les clients consommateurs
    - Commercial - Disponible pour les clients commerciaux
- **Dépendances :**
    - Dépend de la Liste - Identifiants de produit ou types de service requis avant que ce produit puisse être ajouté
    - Utilisé pour les dépendances d'ajout (par exemple, les ajouts mobiles nécessitent un service mobile actif)

**Onglet Provisionnement :**

L'onglet Provisionnement gère l'automatisation Ansible et les exigences d'inventaire.

**Champs de Provisionnement :**

- **Play de Provisionnement :**
    - Nom du playbook Ansible (sans extension .yaml)
    - Doit exister dans le répertoire OmniCRM-API/Provisioners/plays/

    - Appelé lorsque le service est créé, mis à jour ou déprovisionné
- **Variables JSON de Provisionnement :**
    - Variables par défaut transmises au playbook Ansible sous forme de JSON
    - Peuvent être remplacées lors du provisionnement
    - Le playbook reçoit celles-ci plus customer_id, product_id, service_id, access_token
- **Liste d'Éléments d'Inventaire :**
    - Sélecteur multi-choix montrant les types d'éléments d'inventaire disponibles
    - Exemples : Carte SIM, Numéro de Téléphone, Modem Routeur, Adresse IPv4
    - Le client/le personnel sélectionne des éléments spécifiques parmi l'inventaire disponible lors de la commande
    - Les identifiants d'inventaire sélectionnés sont transmis au playbook avec le type d'inventaire comme nom de variable

**Onglet Disponibilité :**

L'onglet Disponibilité contrôle quand le produit peut être acheté et affiche les métadonnées système.

**Paramètres de Disponibilité :**

- **Disponible À Partir de :**

    - Date/heure à laquelle le produit devient disponible à l'achat
    - Laissez vide pour une disponibilité immédiate
    - Utile pour préannoncer de nouveaux produits
- **Disponible Jusqu'à :**
    - Date/heure à laquelle le produit n'est plus disponible à l'achat
    - Laissez vide pour une disponibilité indéfinie
    - Parfait pour des promotions à durée limitée ou des produits en fin de vie
- **Métadonnées Système (Lecture Seule) :**
    - Créé - Horodatage lorsque le produit a été créé pour la première fois
    - Dernière Modification - Horodatage de la mise à jour la plus récente
    - Maintenu automatiquement par le système

**Actions Modales :**

- **Mode Affichage :**
    - Fermer - Rejeter la modale
    - Cloner le Produit - Créer une copie avec le suffixe "_clone"
    - Éditer le Produit - Passer en mode édition
- **Mode Édition/Création :**
    - Annuler - Jeter les modifications et fermer
    - Enregistrer les Modifications - Créer ou mettre à jour le produit (grand bouton pour accentuer)

# Champs de Produit

Le modèle de Produit contient toutes les informations nécessaires pour définir une offre et comment elle doit être provisionnée. Ces champs sont gérés via l'interface modale de Gestion des Produits décrite ci-dessus.

**Informations de Base**

- **product_id** - Identifiant unique automatiquement attribué par le système
- **product_name** - Nom affiché aux clients et au personnel dans l'UI

- **product_slug** - Identifiant unique utilisé dans les URL et les appels API (minuscules, sans espaces, utiliser des tirets)
- **category** - Contrôle où ce produit apparaît dans l'UI :
  - `standalone` - Affiché comme une option de service de base lors de la création d'un nouveau service
  - `addon` - Affiché lors de l'ajout à un service existant
  - `bundle` - Affiché comme une option de service groupé
  - `promo` - Offres promotionnelles spéciales
- **service_type** - Type de service fourni (par exemple, mobile, fixe, fixe-voix, point d'accès, dongle, voix, données). Utilisé pour filtrer quels ajouts s'appliquent à quels services.
- **comment** - Notes internes sur le produit à l'attention du personnel uniquement (non affichées aux clients)
- **icon** - Classe d'icône FontAwesome affichée dans l'UI (par exemple, `fa-solid fa-sim-card`)

**Champs de Tarification**

- **retail_cost** - Charge récurrente mensuelle facturée au client (définie à 0 pour les achats uniques ou les produits prépayés)
- **wholesale_cost** - Votre coût mensuel pour fournir ce service (utilisé pour les calculs de marge)
- **retail_setup_cost** - Frais d'activation ou d'installation uniques facturés au client
- **wholesale_setup_cost** - Votre coût unique pour mettre en place le service
- **tax_percentage** - Pourcentage de taxe appliqué à ce produit (par exemple, 10 pour 10 %, 12,5 pour 12,5 %). Défini à 0 pour les produits exonérés de taxe. Ce taux de taxe est automatiquement appliqué aux transactions créées à partir de ce produit.

**Application de la Taxe :**

Lorsqu'une transaction est créée à partir de ce produit, le pourcentage de taxe est automatiquement copié dans la transaction et le montant de la taxe est calculé. Par exemple :

- Produit avec 10 % de taxe, $50.00 coût de détail → La transaction a $5.00 de taxe
- Produit avec 0 % de taxe (exonéré de taxe) → La transaction a $0.00 de taxe
- Remplacement manuel de transaction → Le personnel peut changer le pourcentage de taxe par transaction

**Visibilité et Accès Client**

- **enabled** - Si ce produit est actif et disponible à l'achat (défini à faux pour cacher sans supprimer)
- **residential** - Si les clients résidentiels (consommateurs) peuvent acheter ce produit
- **business** - Si les clients commerciaux peuvent acheter ce produit
- **customer_can_purchase** - Si les clients peuvent s'auto-acheter via le portail (vrai) ou si seul le personnel peut l'ajouter (faux)
- **available_from** - Date/heure à laquelle ce produit devient disponible à l'achat (optionnel)

- **available_until** - Date/heure à laquelle ce produit n'est plus disponible à l'achat (optionnel, utile pour les offres à durée limitée)

**Contrat et Renouvellement**

- **contract_days** - Durée minimale du contrat en jours (par exemple, 30 pour mensuel, 365 pour annuel, 0 pour aucun contrat minimum)
- **auto_renew** - Comportement par défaut de renouvellement :
  - `prompt` - Demande au client chaque fois s'il souhaite renouveler
  - `true` - Renouvelle automatiquement sans demander
  - `false` - Nécessite un renouvellement manuel
- **allow_auto_renew** - Si les clients peuvent activer le renouvellement automatique (défini à faux pour les achats uniques)

**Contenu Vis-à-Vis du Client**

- **terms** - Termes et conditions affichés aux clients avant l'achat (inclure les limitations, règles d'expiration, conditions d'utilisation)
- **features_list** - Liste des caractéristiques et inclusions montrées aux clients (format liste Python : `['Caractéristique 1', 'Caractéristique 2']`)

**Configuration de Provisionnement**

- **provisioning_play** - Nom du playbook Ansible qui provisionne ce service (sans extension .yaml). Doit exister dans `OmniCRM-API/Provisioners/plays/`.
- **provisioning_json_vars** - Variables par défaut transmises au playbook Ansible sous forme de JSON. Celles-ci peuvent être remplacées lors du provisionnement. Le playbook reçoit celles-ci avec `customer_id`, `product_id`, `service_id`, et `access_token`.
- **inventory_items_list** - Liste des éléments d'inventaire requis pour ce produit (par exemple, `['Carte SIM', 'Numéro de Téléphone']`). Lorsqu'un client commande, il sera invité à sélectionner des éléments spécifiques parmi l'inventaire disponible. Les identifiants d'inventaire sélectionnés sont transmis au playbook avec le type d'inventaire comme nom de variable.
- **relies_on_list** - Liste des identifiants de produit ou des types de service qui doivent exister avant que ce produit puisse être ajouté. Utilisé pour les

dépendances d'ajout (par exemple, les ajouts mobiles nécessitent un service mobile actif).

**Métadonnées Système**

- **created** - Horodatage lorsque le produit a été créé (défini automatiquement)
- **last_modified** - Horodatage de la dernière mise à jour (mis à jour automatiquement)

# Exemples de Définitions de Produit

## Produit Autonome (Carte SIM Mobile)

```
{
  "product_id": 1,
  "product_slug": "Mobile-SIM",
  "product_name": "Carte SIM Mobile Seulement",
  "category": "standalone",
  "service_type": "mobile",
  "provisioning_play": "play_psim_only",
  "provisioning_json_vars": "{\"iccid\": \"\", \"msisdn\": \"\"}",
  "inventory_items_list": "['Carte SIM', 'Numéro de Mobile']",
  "retail_cost": 0,
  "retail_setup_cost": 0,
  "wholesale_cost": 3,
  "wholesale_setup_cost": 1,
  "contract_days": 0,
  "residential": true,
  "business": true,
  "enabled": true,
  "customer_can_purchase": true,
  "icon": "fa-solid fa-sim-card",
  "features_list": "['Numéro de Téléphone Australien (04xxx)',
'Vitesses les plus rapides', 'Meilleure couverture', 'Roaming sur
le Continent']",
  "terms": "Doit être activé dans les 6 mois. Tout crédit perdu si
le service n'est pas utilisé pendant 12 mois.",
  "comment": "Carte SIM physique à utiliser avec des Téléphones
Mobiles"
}
```


Ce produit autonome nécessite deux éléments d'inventaire (Carte SIM et Numéro de Mobile) et crée un nouveau service lorsqu'il est provisionné.

## Produit d'Ajout (Forfait de Données Mensuel)


```
{
  "product_slug": "example-mobile-prepaid-mini",
  "product_name": "Forfait Mobile Exemple Mini",
  "category": "addon",
  "service_type": "mobile",
  "provisioning_play": "play_topup_charge_then_action",
  "provisioning_json_vars": "",
  "inventory_items_list": "[]",
  "retail_cost": 30,
  "retail_setup_cost": 0,
  "wholesale_cost": 5.84,
  "contract_days": 30,
  "residential": true,
  "business": false,
  "enabled": true,
  "customer_can_purchase": true,
  "auto_renew": "prompt",
  "icon": "fa-solid fa-sim-card",
  "features_list": "['8 Go de données ultra-rapides', 'Appels et SMS illimités vers les utilisateurs de Exemple Mobile', '100 Minutes d'Appel vers l'Australie', '100 SMS vers l'Australie', 'Expiration de 30 jours']",
  "terms": "Le crédit expire après 30 jours. Une fois les données, la voix ou les SMS épuisés, vous devrez recharger pour continuer à utiliser le service.",
  "comment": "Notre plus petit forfait pour les utilisateurs légers"
}
```


Ce produit d'ajout ne nécessite pas d'inventaire et est appliqué à un service existant. Il facture le client et ajoute des crédits/soldes à leur service.

## Produit Groupé (Forfait Seniors)


```
{
  "product_slug": "Bundle-Seniors",
  "product_name": "Forfait Seniors",
  "category": "bundle",
  "service_type": "fixed",
  "provisioning_play": "play_seniors_package",
  "provisioning_json_vars": "{\"IPTV_Service_ID\":
\"SeniorBundle\"}",
  "inventory_items_list": "['Modem Routeur']",
  "retail_cost": 30,
  "retail_setup_cost": 0,
  "wholesale_cost": 10,
  "wholesale_setup_cost": 11,
  "contract_days": 180,
  "residential": true,
  "business": false,
  "enabled": true,
  "icon": "fa-solid fa-person-walking-with-cane",
  "features_list": "['Téléchargement à 20 Mbps', 'Téléchargement à
5 Mbps', 'Données Illimitées', 'Voix à Domicile', 'TV : Supplément
de +£5 par mois', 'Frais d'Installation de £60']",
  "terms": "Contrat de 6 mois, doit montrer la carte de citoyen
senior pour se qualifier",
  "comment": "Service GPON 20Mbps/2Mbps + IPTV + Téléphone"
}
```


Ce produit groupé provisionne plusieurs services (Internet + IPTV + Téléphone) en utilisant un seul playbook. Il nécessite un élément d'inventaire (Modem Routeur).

### Produit d'Ajout (Recharge Simple)

```
{
  "product_slug": "Mobile-Topup-5",
  "product_name": "Recharge PAYG £5",
  "category": "addon",
  "service_type": "mobile",
  "provisioning_play": "play_topup_monetary",
  "provisioning_json_vars": "{\"service_id\": \"\"}",
  "inventory_items_list": "[]",
  "retail_cost": 5,
  "retail_setup_cost": 0,
  "wholesale_cost": 0,
  "contract_days": 0,
  "residential": true,
  "business": false,
  "enabled": true,
  "customer_can_purchase": true,
  "icon": "fa-solid fa-coins",
  "features_list": "['Crédit de £5', 'Valide pendant 180 jours']",
  "terms": "Valide pendant 180 jours ou jusqu'à ce que tout le
crédit soit utilisé. Consultez notre site Web pour les tarifs
complets",
  "comment": "£5 à utiliser pour les Appels, SMS & Données"
}
```

Cet ajout ajoute simplement un crédit monétaire à un service existant. Aucun inventaire requis, et il utilise `service_id` pour identifier quel service recharger.

# Comment les Variables sont Transmises à Ansible

Comprendre comment les variables circulent de la définition du produit à travers l'API jusqu'au playbook Ansible est crucial pour écrire des playbooks de provisionnement efficaces.

# Sources de Variables et Fusion

Lorsqu'un travail de provisionnement est créé, les variables proviennent de plusieurs sources et sont fusionnées dans cet ordre (les sources ultérieures remplacent les précédentes) :

1. **Variables de provisioning_json_vars du Produit** - Variables par défaut de la définition du produit
2. **Corps de la Demande** - Variables transmises dans l'appel API (peuvent remplacer les valeurs par défaut du produit)
3. **Variables ajoutées par le Système** - Ajoutées automatiquement par le système de provisionnement
4. **Sélections d'Inventaire** - Identifiants des éléments d'inventaire sélectionnés (si `inventory_items_list` n'est pas vide)

# Processus de Fusion des Variables

Le système fusionne les variables de toutes les sources, les sources ultérieures remplaçant les précédentes. Cela permet une personnalisation flexible au moment du provisionnement.

Par exemple, si votre produit a :

```
"provisioning_json_vars": "{\"monthly_cost\": 50, \"data_gb\": 100}"
```

Et votre demande API inclut :

```
{
  "product_id": 10,
  "customer_id": 456,
  "monthly_cost": 45,
  "custom_param": "value"
}
```

Les `extra_vars` finales transmises à Ansible seront :

```
{
    "monthly_cost": 45,         # Remplacé par la demande
    "data_gb": 100,             # Provenant de
provisioning_json_vars
    "product_id": 10,           # Provenant de la demande
    "customer_id": 456,         # Provenant de la demande
    "custom_param": "value",    # Provenant de la demande
    "access_token": "eyJ..."    # Ajouté par le système
}
```

## Variables Ajoutées par le Système

Le système de provisionnement ajoute automatiquement :

- `access_token` - Jeton JWT pour authentifier les appels API vers le CRM (fourni directement pour l'authentification IP/clés API, ou généré à partir de `refresh_token` pour l'authentification utilisateur)
- `initiating_user` - L'identifiant de l'utilisateur qui a déclenché le provisionnement (ou le premier administrateur pour les systèmes automatisés)
- Tous les champs du corps de la demande (`product_id`, `customer_id`, `service_id`, etc.)

## Variables d'Inventaire

Lorsque un produit nécessite des éléments d'inventaire (par exemple, `inventory_items_list: "['Carte SIM', 'Numéro de Mobile']"`), le processus fonctionne comme suit :

1. **UI/API invite à la sélection** - L'utilisateur sélectionne des éléments d'inventaire spécifiques parmi le stock disponible
2. **Les ID d'inventaire sont ajoutés aux variables** - Les identifiants des éléments d'inventaire sélectionnés sont ajoutés avec le type d'inventaire comme nom de variable
3. **Le playbook accède aux ID d'inventaire** - Le playbook de provisionnement peut alors récupérer les détails complets de l'inventaire via l'API CRM

Par exemple, si un utilisateur sélectionne : - Carte SIM avec inventory_id : 789 - Numéro de Mobile avec inventory_id : 101

Les variables transmises au playbook incluent : - `Carte SIM`: 789 - `Numéro de Mobile`: 101

Le playbook peut alors utiliser ces ID pour récupérer les enregistrements d'inventaire complets (ICCID, IMSI, MSISDN, etc.) depuis l'API CRM et utiliser ces informations pour provisionner le service sur l'équipement réseau.

## Comment Ansible Reçoit les Variables

Le système de provisionnement transmet toutes les variables fusionnées au playbook Ansible en tant que `extravars`. À l'intérieur du playbook, ces variables sont disponibles via le système de variables standard d'Ansible et peuvent être utilisées dans les tâches.

Les variables peuvent être référencées directement dans les tâches du playbook en utilisant la syntaxe `{{ variable_name }}`. Par exemple, `{{ product_id }}`, `{{ customer_id }}`, `{{ monthly_cost }}`, etc.

## Variables Transmises aux Produits d'Ajout

Lorsqu'un produit d'ajout est provisionné, le système transmet automatiquement :

- `product_id` - L'ID du produit d'ajout en cours de provisionnement
- `customer_id` - Le client qui possède le service
- `service_id` - L'ID du service auquel cet ajout est ajouté (critique pour les ajouts)
- `access_token` - Jeton d'authentification pour les appels API
- Toutes les variables de `provisioning_json_vars`
- Toutes les variables supplémentaires de la demande API

# Exemple de Flux de Provisionnement d'Ajout

Lorsqu'un client ajoute l'ajout "Recharge de £5" à son service mobile (service_id : 123), le playbook reçoit des variables comprenant :

- `product_id`: 45 (le produit de recharge)
- `customer_id`: 456 (le client)
- `service_id`: 123 (le service auquel ajouter du crédit)
- `access_token`: Jeton d'authentification
- Plus toutes les variables de `provisioning_json_vars` du produit

Le playbook utilise ensuite ces variables pour :

1. **Récupérer les détails du service** depuis l'API CRM en utilisant le `service_id`
2. **Extraire l'UUID du service** et d'autres informations de l'enregistrement du service
3. **Ajouter du crédit au système de facturation** (OCS) en utilisant l'UUID du service
4. **Enregistrer la transaction** dans le CRM à des fins de facturation

Ce flux permet à l'ajout d'identifier exactement quel service modifier et d'appliquer les changements de manière appropriée.

# Différence entre les Variables de Produits Autonomes et d'Ajouts

**Produits Autonomes** reçoivent :

- `product_id` - Le produit en cours de provisionnement
- `customer_id` - Le client commandant le service
- Identifiants d'éléments d'inventaire (par exemple, `Carte SIM`, `Numéro de Mobile`) si le produit les nécessite
- `access_token` - Pour l'authentification API

**Produits d'Ajout** reçoivent :

- `product_id` - Le produit d'ajout en cours de provisionnement
- `customer_id` - Le client qui possède le service
- `service_id` - **L'ID du service existant à modifier** (c'est la clé de la différence)
- `access_token` - Pour l'authentification API

La clé de la différence est `service_id` - cela indique au playbook quel service existant modifier ou ajouter.

## Produits Groupés

Les produits groupés sont provisionnés comme des ajouts mais leur playbook peut créer plusieurs enregistrements de service. Ils reçoivent les mêmes variables que les ajouts, y compris :

- `product_id` - Le produit groupé
- `customer_id` - Le client
- `service_id` - Service parent (le cas échéant)
- Identifiants d'éléments d'inventaire (par exemple, `Modem Routeur`) si requis
- `access_token` - Pour l'authentification API

Le playbook groupé (par exemple, `play_seniors_package`) crée alors plusieurs services liés (Internet, IPTV, Téléphone) et les relie ensemble.

# Services

Un service est une instance d'un produit qui appartient à un client, pour lequel il est facturé.

C'est essentiellement un lien vers un compte `OCS` (Système de Facturation En Ligne) qui gère la génération de charges et les soldes et usages réels pour le compte. Le OCS est alimenté par CGRateS et gère les soldes monétaires, les soldes unitaires (données, voix, SMS), les Plans d'Action pour le renouvellement automatique, et les Seuils pour les limites de dépenses.

# Ajout d'un Service : Sélection et Filtrage de Produit

Lors de l'ajout d'un service à un client (soit un nouveau service autonome ou un ajout à un service existant), le système affiche les produits disponibles dans une interface de carrousel. Les produits affichés sont filtrés en fonction de plusieurs critères :

## Filtrage de Produits pour Services Autonomes

Lors de la création d'un nouveau service pour un client, l'UI affiche les produits filtrés par :

1. **Type de Client** - Les produits sont catégorisés comme :
   - **Individuel (Résidentiel)** : Produits où `residential = true` ou `business = false`
   - **Commercial** : Produits où `business = true`
2. **Catégorie** - Les produits sont séparés en :
   - **Plans de Service** : Produits avec `category` = `standalone` ou `bundle`
   - **Ajouts** : Produits avec `category` = `addon` (affichés dans un carrousel séparé)
3. **Disponibilité** - Les produits ne sont affichés que si :
   - `enabled = true` - Le produit est actif et non désactivé
   - La date actuelle est entre `available_from` et `available_until` - Le produit est dans sa fenêtre de disponibilité
   - `customer_can_purchase = true` (si le client s'auto-achetant) - Le produit permet l'achat direct par le client

> ⓘ **REMARQUE**
>
> **Filtrage au Niveau de l'API** : L'API filtre automatiquement les produits par statut activé et dates de disponibilité à deux niveaux :
>
> - **Points de Terminaison d'Achat/Sélection** (`/crm/product/`) - Utilisé par le modal d'Ajouts et la Liste de Plans pour la sélection de produits. Filtre automatiquement pour afficher UNIQUEMENT les produits activés dans leur plage de dates de disponibilité. Cela garantit que les clients

> et le personnel ne peuvent sélectionner que des produits qui sont actuellement disponibles à l'achat.
> - **Points de Terminaison de Gestion** (`/crm/product/paginated`) - Utilisé par la page de Gestion des Produits. Affiche TOUS les produits, y compris ceux désactivés et hors des dates de disponibilité, permettant aux administrateurs de gérer l'ensemble du catalogue de produits, y compris les produits inactifs.
>
> Passez `include_disabled=true` au point de terminaison de produit de base pour contourner le filtrage (uniquement pour un usage administratif).

L'UI affiche des carrousels séparés pour :

- **Plans de Service Individuels** - Produits résidentiels pour clients consommateurs
- **Plans de Service Commercial** - Produits commerciaux pour clients commerciaux
- **Ajouts Individuels** - Packs d'ajouts résidentiels
- **Ajouts Commerciaux** - Packs d'ajouts commerciaux

# Filtrage de Produits pour Services d'Ajout

Lors de l'ajout d'un ajout à un **service existant**, un filtrage supplémentaire est appliqué :

1. **Correspondance de Type de Service** - Seuls les ajouts avec `service_type` correspondant sont affichés :
   - Si le service existant a `service_type = "mobile"`, seuls les ajouts avec `service_type = "mobile"` sont affichés
   - Cela garantit que les clients mobiles ne voient que des ajouts mobiles, que les clients Internet ne voient que des ajouts Internet, etc.
2. **Vérification des Dépendances** - Si un ajout a une `relies_on_list` :
   - Le système vérifie si le client a les produits/services requis
   - Seuls les ajouts dont les dépendances sont satisfaites sont affichés
3. **Filtre de Type de Client Identique** - Les ajouts sont toujours filtrés par `residential` contre `business` pour correspondre au type de client

## Scénario de Filtrage Exemple

Pour un client commercial avec un service mobile existant (`service_type = "mobile"`):

- **Produits Autonomes Affichés** : Tous les produits autonomes/groupés commerciaux (`business = true`, `category != "addon"`)
- **Produits d'Ajout Affichés** : Seulement les ajouts mobiles commerciaux (`business = true`, `category = "addon"`, `service_type = "mobile"`)
- **Produits Cachés** : Produits résidentiels, ajouts pour d'autres types de services (internet, voix, etc.), produits désactivés

# Champs de Service

Le modèle de Service contient des champs qui suivent l'instance de service provisionnée et sa relation avec le client, le produit et le système de facturation.

#### Informations de Base sur le Service

- **service_id** - Identifiant unique automatiquement attribué par le système (lecture seule)
- **customer_id** - Lien vers le client qui possède ce service (lecture seule après création)
- **product_id** - Lien vers le produit à partir duquel ce service a été créé (lecture seule après création)
- **service_name** - Nom affiché aux clients (modifiable)
- **service_type** - Type de service : mobile, internet, voip, iptv, bundle, etc. (modifiable)
- **service_uuid** - Identifiant unique utilisé dans OCS/CGRateS pour la facturation (lecture seule, auto-généré)
- **icon** - Classe d'icône FontAwesome pour affichage dans le portail d'auto-service (modifiable)

#### Statut et Dates du Service

- **service_status** - Statut actuel : Actif, Inactif, Suspendu, etc. (modifiable)

- **service_provisioned_date** - Date à laquelle le service a été provisionné pour la première fois (défini automatiquement, lecture seule)
- **service_active_date** - Date à laquelle le service est devenu actif (modifiable)
- **service_deactivate_date** - Date à laquelle le service expire ou sera désactivé (modifiable)
- **contract_end_date** - Date de fin de l'engagement contractuel (modifiable)

**Facturation et Tarification**

- **retail_cost** - Charge récurrente mensuelle au client (modifiable)
- **wholesale_cost** - Votre coût pour fournir le service (modifiable)
- **service_billed** - Si ce service apparaît sur les factures (modifiable, par défaut : vrai)
- **service_taxable** - Si les taxes s'appliquent à ce service (modifiable, par défaut : vrai)
- **invoiced** - Si le service a été facturé (défini automatiquement par le système de facturation)
- **promo_code** - Code promotionnel utilisé lors de la création du service (modifiable)

**Visibilité Client**

- **service_visible_to_customer** - Si le client peut voir ce service dans le portail d'auto-service (modifiable, par défaut : vrai)
- **service_usage_visible_to_customer** - Si le client peut voir les détails d'utilisation/soldes (modifiable, par défaut : vrai)

**Configuration de Provisionnement**

- **provisioning_play** - Playbook Ansible utilisé pour provisionner ce service (hérité du produit, lecture seule)
- **provisioning_json_vars** - Variables transmises au playbook de provisionnement (héritées du produit, lecture seule)
- **deprovisioning_play** - Playbook Ansible à exécuter lorsque le service est déprovisionné (lecture seule)

- **deprovisioning_json_vars** - Variables pour le playbook de déprovisionnement (lecture seule)

**Relations de Service**

- **bundled_parent** - Si ce service fait partie d'un bundle, le service_id du service parent (lecture seule)
- **site_id** - Lien vers le site/local physique où le service est délivré (modifiable)

**Notes et Métadonnées**

- **service_notes** - Notes internes sur le service pour référence du personnel (modifiable)
- **created** - Horodatage lorsque le service a été créé (défini automatiquement, lecture seule)
- **last_modified** - Horodatage de la dernière mise à jour (mis à jour automatiquement, lecture seule)

# Champs Modifiables vs Lecture Seule

**Modifiables via API/UI :**

Les services peuvent être mis à jour via `PATCH /crm/service/{service_id}` avec ces champs :

- service_name, service_type, service_status
- service_notes
- retail_cost, wholesale_cost
- service_billed, service_taxable
- service_visible_to_customer, service_usage_visible_to_customer
- service_active_date, service_deactivate_date, contract_end_date
- icon, promo_code, site_id

**Lecture Seule (Définie Automatiquement) :**

Ces champs ne peuvent pas être modifiés directement après création :

- service_id, customer_id, product_id
- service_uuid (généré lors du provisionnement)
- service_provisioned_date
- provisioning_play, provisioning_json_vars
- deprovisioning_play, deprovisioning_json_vars
- bundled_parent
- invoiced (géré par le système de facturation)
- created, last_modified (géré automatiquement)

# Provisionnement de Produits en Services

Le processus de provisionnement convertit un Produit (modèle) en un Service (instance spécifique au client) à travers une série d'étapes coordonnées impliquant l'UI Web, l'API et les playbooks Ansible.

## Flux de Provisionnement de Haut Niveau

1. **Configuration Pré-Provisionnement** - Produit créé dans l'API avec configuration de provisionnement, et playbooks Ansible correspondants écrits et testés
2. **Sélection de Service** - Depuis la Page Client, le personnel ou le client sélectionne "Ajouter un Service"
3. **Filtrage de Produits** - Produits affichés filtrés en fonction de :
   - Type de client (résidentiel/commercial)
   - Services existants (pour les dépendances d'ajout dans `relies_on_list`)
   - Dates de disponibilité (`available_from`/`available_until`)
   - Drapeaux `enabled` et `customer_can_purchase`
4. **Personnalisation** - Option de remplacer les variables de provisionnement (pour ajustements de prix, configurations personnalisées, etc.)
5. **Sélection d'Inventaire** - Si le produit nécessite un inventaire (`inventory_items_list` n'est pas vide), l'utilisateur sélectionne des

éléments spécifiques (par exemple, quelle carte SIM, quel numéro de téléphone)

6. **Initiation de la Provision** - Lorsque le bouton "Provisionner" est cliqué, l'API crée un travail de provisionnement

# Flux d'Intégration Détail API et Ansible

Lorsqu'un service est provisionné, la séquence suivante se produit :

**Étape 1 : Création de Travail de Provisionnement**

L'API reçoit la demande de provisionnement et crée un travail de provisionnement avec :

- `provisioning_play` - Nom du playbook Ansible (par exemple, `play_psim_only`)
- `provisioning_json_vars` - Chaîne JSON de variables du produit ou remplacées par la demande
- `customer_id` - ID du client commandant le service
- `product_id` - ID du produit en cours de provisionnement
- `service_id` - (Optionnel) ID du service existant pour les ajouts
- Sélections d'inventaire - Identifiants des éléments d'inventaire sélectionnés

**Étape 2 : Assemblage des Variables**

Le service de provisionnement fusionne les variables provenant de plusieurs sources dans cet ordre :

1. Variables `provisioning_json_vars` du Produit (valeurs par défaut de la définition du produit)
2. Paramètres du corps de la demande (peuvent remplacer les valeurs par défaut du produit)
3. Variables ajoutées par le Système :
   - `access_token` - Jeton JWT pour l'authentification API vers le CRM
   - `initiating_user` - ID de l'utilisateur qui a déclenché le provisionnement
   - `customer_id`, `product_id`, `service_id`

4. Sélections d'inventaire - Ajoutées sous forme de paires `{inventory_type: inventory_id}`

Exemple de variables fusionnées :

```
{
    "customer_id": 123,
    "product_id": 456,
    "service_id": 789,          # Seulement pour les ajouts
    "SIM Card": 1001,           # Provenant de la sélection
d'inventaire
    "Mobile Number": 1002,      # Provenant de la sélection
d'inventaire
    "monthly_cost": 30,         # Provenant de
provisioning_json_vars
    "data_gb": 50,              # Provenant de
provisioning_json_vars
    "access_token": "eyJ...",   # Ajouté par le système
    "initiating_user": 5        # Ajouté par le système
}
```

**Étape 3 : Création d'Enregistrement de Provision**

Un enregistrement `Provision` est créé dans la base de données avec :

- `provision_id` - Identifiant unique pour le suivi
- `provisioning_play` - Nom du fichier du playbook
- `provisioning_json_vars` - Variables fusionnées sous forme de chaîne JSON
- `task_count` - Nombre de tâches dans le playbook (extrait du YAML)
- `provisioning_status` - Code de statut (initialement défini à 1 = en cours, puis mis à jour à 0 = succès, 2 = échec, ou peut rester 1 si toujours en cours)
- `product_id`, `customer_id`, `service_id` - Références contextuelles

**Étape 4 : Exécution de Playbook en Arrière-Plan**

L'API lance un thread en arrière-plan qui :

1. Charge le YAML du playbook depuis `OmniCRM-API/Provisioners/plays/{playbook_name}.yaml`
2. Appelle `ansible_runner.run()` avec :
   - `playbook` - Chemin vers le fichier YAML chargé
   - `extravars` - Toutes les variables fusionnées (transmises à Ansible)
   - `inventory` - Défini sur `'localhost,'` (exécution locale)
   - `event_handler` - Gestionnaire personnalisé pour capturer les événements d'exécution des tâches
3. Surveille l'exécution du playbook en temps réel

**Étape 5 : Capture d'Événements et Journalisation** (`ProvisioningEventHandler`)

À mesure que chaque tâche Ansible s'exécute, les événements sont capturés et stockés en tant que enregistrements `Provision_Event` :

- `event_name` - Nom de la tâche provenant du playbook
- `event_number` - Numéro de séquence
- `provisioning_status` - Code de statut indiquant le résultat de la tâche :
  - **0** = Succès - Tâche terminée avec succès
  - **1** = En cours - Tâche en cours d'exécution
  - **2** = Échec - Échec critique qui arrête le provisionnement
  - **3** = Échec (ignoré) - La tâche a échoué mais les erreurs ont été ignorées (`ignore_errors: true` dans le playbook)
- `provisioning_result_json` - Résultats de la tâche avec des données sensibles masquées

Le gestionnaire d'événements supprime automatiquement les mots de passe, clés, secrets, et autres données sensibles des journaux.

**Étape 6 : Exécution du Playbook Ansible** (`Provisioners/plays/*.yaml`)

Le playbook Ansible s'exécute localement et effectue généralement ces actions :

1. **Récupérer la Définition du Produit** - Requête GET à `/crm/product/product_id/{{ product_id }}` en utilisant `{{ access_token`

`}}`

2. **Récupérer les Informations Client** - Requête GET à `/crm/customer/customer_id/{{ customer_id }}`

3. **Traiter les Éléments d'Inventaire** (si nécessaire) - Requête GET à `/crm/inventory/inventory_id/{{ inventory_id }}` pour chaque élément sélectionné afin de récupérer les détails complets (ICCID, MSISDN, numéros de série, etc.)

4. **Configurer les Systèmes Externes** - Effectuer des appels API à :

    - HSS (Serveur d'Abonnés à Domicile) pour le provisionnement des abonnés
    - IMS (Système Multimédia IP) pour l'enregistrement vocal
    - CGRateS/OCS pour la création de compte, configuration de facturation, plans tarifaires
    - Serveurs ENUM pour le mappage des numéros de téléphone
    - Équipement réseau (routeurs, commutateurs, etc.)

5. **Ajouter des Coûts d'Installation** (si applicable) - POST à `/crm/transaction/` pour enregistrer des frais uniques

6. **Facturer le Client** - POST à OCS/CGRateS pour facturer retail_setup_cost si configuré

7. **Créer un Compte OCS** - POST à OCS/CGRateS pour créer un compte de facturation avec UUID

8. **Configurer les Charges Récurrentes** - Créer des Actions et des Plans d'Action dans OCS/CGRateS pour les charges récurrentes mensuelles

9. **Créer un Enregistrement de Service** - PUT/POST à `/crm/service/` pour créer l'enregistrement de service dans le CRM :

```
{
  "customer_id": 123,
  "product_id": 456,
  "service_name": "Carte SIM Mobile - 0412345678",
  "service_uuid": "generated-uuid-for-ocs",
  "service_status": "Active",
  "service_type": "mobile",
  "retail_cost": 30,
  "wholesale_cost": 5,
  "provisioning_play": "play_psim_only",
  "provisioning_json_vars": "{...}"
}
```

10. **Attribuer l'Inventaire** - PATCH à `/crm/inventory/inventory_id/{{ inventory_id }}` pour marquer l'inventaire comme "Attribué" au service

11. **Envoyer des Notifications** (optionnel) - Email ou SMS au client avec les détails du service

**Étape 7 : Achèvement et Mise à Jour du Statut**

Lorsque le playbook se termine :

- **Succès** : `Provision.provisioning_status` mis à jour à **0** (Succès)
- **Échec Critique** : `Provision.provisioning_status` mis à jour à **2** (Échoué), et un email d'échec envoyé à `crm_config.provisioning.failure_list`
- **Échecs Non-Critiques** : Les tâches qui échouent avec `ignore_errors: true` sont marquées avec le statut **3** (Échoué mais ignoré) et ne stoppent pas le provisionnement

Le service provisionné est maintenant visible dans le CRM et actif pour le client (si le provisionnement a réussi).

## Différences Clés : Provisionnement Autonome vs Ajout vs Produit Groupé

**Produits Autonomes** (`category: standalone`) :

- Reçoivent `customer_id` et `product_id`
- Nécessitent généralement des éléments d'inventaire (cartes SIM, numéros de téléphone, modems)
- Créent un nouvel enregistrement de service via l'API PUT `/crm/service/`
- Provisionnent de nouvelles ressources sur des systèmes externes (HSS, OCS, équipement réseau)
- Exemple : Activation d'une nouvelle carte SIM mobile, nouvelle connexion Internet

**Produits d'Ajout** (`category: addon`) :

- Reçoivent `customer_id`, `product_id`, et **`service_id`** (service existant à modifier)
- Nécessitent généralement PAS d'inventaire (ou un inventaire minimal)
- **Modifient un service existant** ou ajoutent des charges au compte OCS existant
- Peuvent exécuter des actions sur OCS (ajouter un pack de données, ajouter du crédit, activer une fonctionnalité)
- Ne créent pas de nouveaux enregistrements de service (ou créent des enregistrements de service enfants liés au parent)
- Exemple : Recharge de plan de données mensuel, pack d'itinérance internationale, crédit supplémentaire

**Produits Groupés** (`category: bundle`) :

- Semblable aux ajouts en termes de variables reçues
- Peuvent nécessiter certains éléments d'inventaire (par exemple, modem pour le forfait à domicile)
- Créent **plusieurs** enregistrements de service liés (Internet + TV + Téléphone)
- Provisionnent plusieurs ressources à travers différents systèmes
- Lient les services ensemble dans le CRM pour une facturation/gestion unifiée
- Exemple : Forfait à domicile (Internet + IPTV + téléphone)

# Exigences du Playbook de Provisionnement

Pour qu'un playbook fonctionne correctement, il doit :

1. **Être situé à** `OmniCRM-API/Provisioners/plays/{playbook_name}.yaml`
2. **Accepter des variables** via `extravars` d'Ansible (accessibles comme `{{ variable_name }}`)
3. **Authentifier les appels API** en utilisant `Authorization: Bearer {{ access_token }}` en-tête
4. **Gérer les échecs gracieusement** en utilisant des blocs `rescue` et `ignore_errors` lorsque cela est approprié
5. **Créer un enregistrement de service** pour les produits autonomes, ou modifier le service existant pour les ajouts
6. **Attribuer l'inventaire** si des éléments d'inventaire ont été sélectionnés
7. **Retourner des messages d'erreur significatifs** via le module `fail` lorsque des erreurs critiques se produisent

# Variables Courantes Disponibles dans les Playbooks

Chaque playbook reçoit ces variables :

- `customer_id` - Entier, client commandant le service
- `product_id` - Entier, produit en cours de provisionnement
- `service_id` - Entier (ajouts/produits groupés uniquement), service existant à modifier
- `access_token` - Chaîne, jeton JWT pour l'authentification API CRM
- `initiating_user` - Entier, utilisateur qui a déclenché le provisionnement
- Plus tous les ID d'éléments d'inventaire : `{{ inventory_type }}: inventory_id`
- Plus toutes les variables de `provisioning_json_vars`
- Plus toutes les variables transmises dans la demande de provisionnement

Les playbooks peuvent utiliser celles-ci pour :

- Récupérer les détails complets du produit : `GET /crm/product/product_id/{{ product_id }}`
- Récupérer les détails du client : `GET /crm/customer/customer_id/{{ customer_id }}`
- Récupérer les détails de l'inventaire : `GET /crm/inventory/inventory_id/{{ SIM_Card }}`
- Créer des transactions : `POST /crm/transaction/`
- Créer des services : `PUT /crm/service/`
- Mettre à jour des services : `PATCH /crm/service/{{ service_id }}`
- Attribuer l'inventaire : `PATCH /crm/inventory/inventory_id/{{ inventory_id }}`

# Exemple : Flux de Playbook d'Ajout Simple

Pour un ajout de recharge de données mobiles :

1. Le playbook reçoit : `customer_id`, `product_id`, `service_id`, `access_token`
2. Récupérer les détails du service : `GET /crm/service/{{ service_id }}` pour obtenir `service_uuid`
3. Récupérer les détails du produit : `GET /crm/product/product_id/{{ product_id }}` pour obtenir le prix et le montant de données
4. Facturer le client dans OCS : `POST à CGRateS` pour déduire retail_cost du solde
5. Ajouter du crédit de données dans OCS : `POST à CGRateS` pour ajouter un solde de données avec expiration
6. Enregistrer la transaction dans le CRM : `POST /crm/transaction/` avec les détails de la charge
7. Terminer avec succès

L'ensemble du processus est suivi dans les tables `Provision` et `Provision_Event` pour le débogage et les audits.

# Implication de l'OCS

`OCS` (Système de Facturation En Ligne), mis en œuvre via CGRateS, gère tous les frais en temps réel et le suivi de l'utilisation des services. L'enregistrement de service CRM agit comme un pointeur vers le compte OCS, qui gère :

- **Charges récurrentes** - Frais mensuels, location de DID, frais d'abonnement
- **Facturation basée sur l'utilisation** - Appels vocaux par minute, données par Mo, frais par SMS
- **Gestion des soldes** - Soldes monétaires (crédit prépayé) et soldes unitaires (Go de données, minutes de voix, nombre de SMS)
- **Conversions de solde** - Conversion de soldes monétaires en soldes unitaires (par exemple, dépenser $30 pour obtenir un pack de données de 10 Go)
- **État du compte** - Actif, suspendu, désactivé en fonction des limites de crédit et des seuils

L'enregistrement de service CRM contient des métadonnées et une configuration (client, produit, tarification, visibilité), tandis que l'OCS contient l'état de facturation en direct (soldes, utilisation, charges).

# Récupération de l'Utilisation et des Soldes de Service

Les informations d'utilisation du service sont récupérées depuis OCS/CGRateS et affichées aux clients et au personnel en temps réel.

## Comment l'Utilisation est Récupérée

Lorsqu'une utilisation de service est demandée (via l'UI ou l'API), le flux suivant se produit :

1. **Demande API** - Le frontend appelle `GET /crm/service/{service_id}` ou consulte les détails du service dans l'UI

2. **Recherche de Service** - L'API récupère l'enregistrement de service dans la base de données, extrait `service_uuid`

3. **Appels API CGRateS** - Le système effectue deux appels à CGRateS :

    i. **Get_Balance(service_uuid)** - Récupère le solde du compte avec `BalanceMap`
        - Renvoie des soldes organisés par type : DONNÉES, VOIX, SMS, MONÉTAIRE, DONNÉES_DONGLE
        - Chaque solde comprend : ID, Valeur, Date d'Expiration, Poids, DestinationIDs
        - Le système ajoute des champs lisibles par l'homme : `custom_Name_hr`, `custom_Expiration`, `custom_Description_String`

    ii. **Get_ActionPlans(service_uuid)** - Récupère les plans d'action de renouvellement automatique actifs (couverts dans la section suivante)

4. **Fusion de Réponse** - Les données CGRateS sont fusionnées dans la réponse de service :

```
{
  "service_id": 123,
  "service_name": "Service Mobile",
  "service_uuid": "abc-123-def",
  "cgrates": {
    "BalanceMap": {
      "DATA": [{
        "ID": "DATA_10GB",
        "Value": 5368709120,
        "ExpirationDate": "2025-02-01T00:00:00Z",
        "custom_Name_hr": "Pack de Données de 10 Go",
        "custom_Expiration": "1er Février 2025",
        "custom_Description_String": "5 Go restants"
      }],
      "VOICE": [{
        "ID": "VOICE_UNLIMITED",
        "Value": 999999999,
        "custom_Name_hr": "Appels Illimités",
        "custom_Description_String": "Minutes illimitées"
      }],
      "MONETARY": [{
        "ID": "PREPAID_CREDIT",
        "Value": 25.50,
        "custom_Description_String": "Crédit de 25.50 $"
      }]
    },
    "ActionPlans": [...]
  }
}
```


5. **Affichage UI** - Les composants frontend affichent les données d'utilisation :

   - **ServiceUsage.js** - Composant principal d'affichage d'utilisation avec actualisation automatique toutes les 3 secondes
   - **UsageCard.js** - Cartes de résumé pour chaque type de solde
   - **UsageProgress.js** - Barres de progression montrant le pourcentage utilisé/restant
   - Les soldes sont codés par couleur et formatés pour une meilleure lisibilité

# Structure des Données d'Utilisation

Chaque solde dans le `BalanceMap` contient :

**Champs Natifs CGRateS :**

- `ID` - Identifiant unique pour le solde (par exemple, "DATA_10GB_2025_01")
- `Value` - Montant du solde :
  - Pour DONNÉES : octets (5368709120 = 5 Go)
  - Pour VOIX : secondes (3600 = 1 heure)
  - Pour SMS : compte (100 = 100 messages)
  - Pour MONÉTAIRE : unités monétaires (25.50 = $25.50)
- `ExpirationDate` - Horodatage ISO 8601 lorsque le solde expire
- `Weight` - Priorité pour la consommation du solde (poids plus élevé consommé en premier)
- `DestinationIDs` - Destinations auxquelles ce solde s'applique (par exemple, ["AU", "INTERNATIONAL"])

**Champs Lisibles par l'Homme (ajoutés par le CRM) :**

- `custom_Name_hr` - Nom lisible par l'homme extrait de l'ID
- `custom_Expiration` - Date d'expiration formatée (par exemple, "15 Janv 2025" ou "dans 11 jours")
- `custom_Description_String` - Description de solde lisible par l'homme :
  - DONNÉES : "5 Go restants" ou "10 Go au total"
  - VOIX : "60 minutes restantes" ou "Illimité"
  - SMS : "50 SMS restants"
  - MONÉTAIRE : "Crédit de 25.50 $"

# Contrôle de Visibilité de l'Utilisation

La visibilité de l'utilisation du service est contrôlée par deux champs :

- **service_visible_to_customer** - Si faux, le service est entièrement caché du portail d'auto-service du client
- **service_usage_visible_to_customer** - Si faux, le service est visible mais les détails d'utilisation/solde sont cachés (le client peut voir qu'il a le

service, mais pas combien il a utilisé)

Cela permet aux opérateurs de :

- Cacher les services internes/test aux clients
- Montrer que le service existe sans révéler l'utilisation (utile pour les plans illimités ou les services sensibles)
- Affichage totalement transparent de l'utilisation (par défaut)

## Mises à Jour en Temps Réel de l'Utilisation

L'UI Web actualise automatiquement les données d'utilisation :

- **Intervalle** : Toutes les 3 secondes (configurable dans le composant ServiceUsage)
- **Méthode** : Interroge `GET /crm/service/{service_id}` qui récupère des données en direct depuis CGRateS
- **Efficacité** : Seules les vues de services actifs se rafraîchissent ; les vues de liste utilisent des données mises en cache

Cela garantit que les clients et le personnel voient des mises à jour de solde presque en temps réel à mesure que l'utilisation se produit.

# Charges Récurrentes / Renouvellement Automatique

Les charges récurrentes, telles qu'une Charge de Service Mensuelle, ou une Charge par DID, sont d'abord créées en tant qu'Actions à l'intérieur de l'OCS et prennent le format `Action_ServiceUUID_ServiceName_WhatitDoes`.

Pour un service GPON de $60 par mois qui inclut 1 To d'utilisation, l'Action ressemblerait à ceci :

`Action_kj49-adsf-1234-9742_60_GPON_1TB_MonthlyExpiry`

1. Réinitialiser le Solde Monétaire à $0

2. Envoyer un POST HTTP à `/simple_provision` sur le CRM pour provisionner quelque chose
3. Ajouter un Crédit pour 1 To d'Utilisation expirant dans 1 mois

Si nous voulons que le MRC soit récurrent (ce que nous faisons), nous créerions un `ActionPlan` nommé `ActionPlan_{{ service_uuid }}_Monthly_Charge` qui aurait le temps défini sur *mensuel* pour se déclencher chaque mois, et assigner le `ActionPlan` au compte.

Nous pouvons définir en fonction du paramètre *Année / Mois / Jours* une date d'expiration pour quand le MRC s'arrêtera également, par exemple pour un service fixe de 12 mois qui s'arrête après ce point.

Parce que les Actions et `ActionPlans` sont tous deux uniques au service, ils ne partagent rien avec d'autres services.

Cela signifie que nous pouvons les provisionner avec des valeurs ajustées, et cela n'impactera aucun autre service.

# Ajouts & Options Supplémentaires

Les Ajouts / Options Supplémentaires tels que l'achat de données supplémentaires, de packs d'itinérance, de minutes internationales, etc., sont gérés de la même manière. Une Action est créée pour faire ce qui est nécessaire, comme facturer une valeur monétaire et puis accorder un solde unitaire avec une expiration définie.

Plutôt que d'utiliser des `ActionPlans` pour ajouter automatiquement cela à se reproduire sur le compte, nous déclenchons simplement `ExecuteAction` pour l'Action que nous venons de créer une fois depuis Ansible.

# Ajout de Soldes Monétaires Prépayés

Pour les soldes monétaires prépayés, tels qu'un plan PAYG de $10, cela est ajouté comme un solde monétaire, mais avec une priorité plus élevée.

La limite de crédit sur ces services pour le solde par défaut serait de $0.

# Limites de Crédit / Prévention des Dépenses Excessives

Les `ThresholdS` sont utilisés sur chaque compte pour définir le maximum de dépenses pour une période donnée.

Pour les clients PAYG / Prépayés, cela est de $0.

# Interaction avec l'OCS via le CRM

Pour chaque Service, vous pouvez voir les `Balances` et `ActionPlans`, `Actions` et `ThresholdS` de l'OCS depuis l'API CRM.

Les `ActionPlans` peuvent être supprimés si nécessaire depuis l'API CRM, actionnés via des Playbooks Ansible. Les `ActionPlans` peuvent être ajoutés si nécessaire, depuis le CRM, en ajoutant un Ajout/Service et actionnés via des Playbooks Ansible.

Les comptes `OCS` peuvent être désactivés, ce qui arrêtera l'exécution des `ActionPlans` et des services pouvant être consommés.

Pour les Limites de Crédit, une valeur de `ThresholdS` est définie selon la politique pour le produit.

# Visualisation et Gestion des ActionPlans dans le CRM

Les ActionPlans (configurations de renouvellement automatique) sont affichés et gérés via l'interface CRM, permettant au personnel et aux clients de voir les renouvellements automatiques à venir et de les gérer.

# Comment les ActionPlans sont Récupérés et Affichés

Lors de la visualisation d'un service dans le CRM, les ActionPlans sont automatiquement récupérés et affichés :

1. **Appel API** - Lorsque `GET /crm/service/{service_id}` est appelé, l'API :
   - Récupère l'enregistrement de service de la base de données
   - Extrait le `service_uuid` (identifiant de compte OCS)
   - Appelle l'API CGRateS pour récupérer les ActionPlans par UUID de service
   - Cela appelle en interne `Get_ActionPlans(service_uuid)` sur CGRateS
2. **Structure de Données ActionPlan** - Chaque ActionPlan retourné contient :

```
{
    "ActionPlanId": "ServiceID_abc-123-
def__ProductID_456__...",
    "PlanName": "Plan de Renouvellement Mensuel",
    "NextExecTime": "2025-02-01T00:00:00Z",
    "custom_NextExecTime_hr": "dans 11 jours",
    "ActionPlanId_split_dict": {
        "ServiceID": "abc-123-def",
        "ProductID": 456,
        "CustomerID": 789,
        ...
    }
}
```

   - `ActionPlanId` - Identifiant unique contenant des informations encodées sur le service/produit/client
   - `PlanName` - Nom du plan d'action (typiquement le nom du playbook de renouvellement)
   - `NextExecTime` - Horodatage ISO lorsque le plan d'action s'exécutera prochainement

- `custom_NextExecTime_hr` - Temps lisible par l'homme jusqu'à l'exécution (par exemple, "dans 11 jours", "demain", "1er Février 2025")
- `ActionPlanId_split_dict` - Dictionnaire avec des composants analysés du ActionPlanId

3. **Affichage UI** - Les ActionPlans sont affichés dans le composant **ActionPlansTable** :

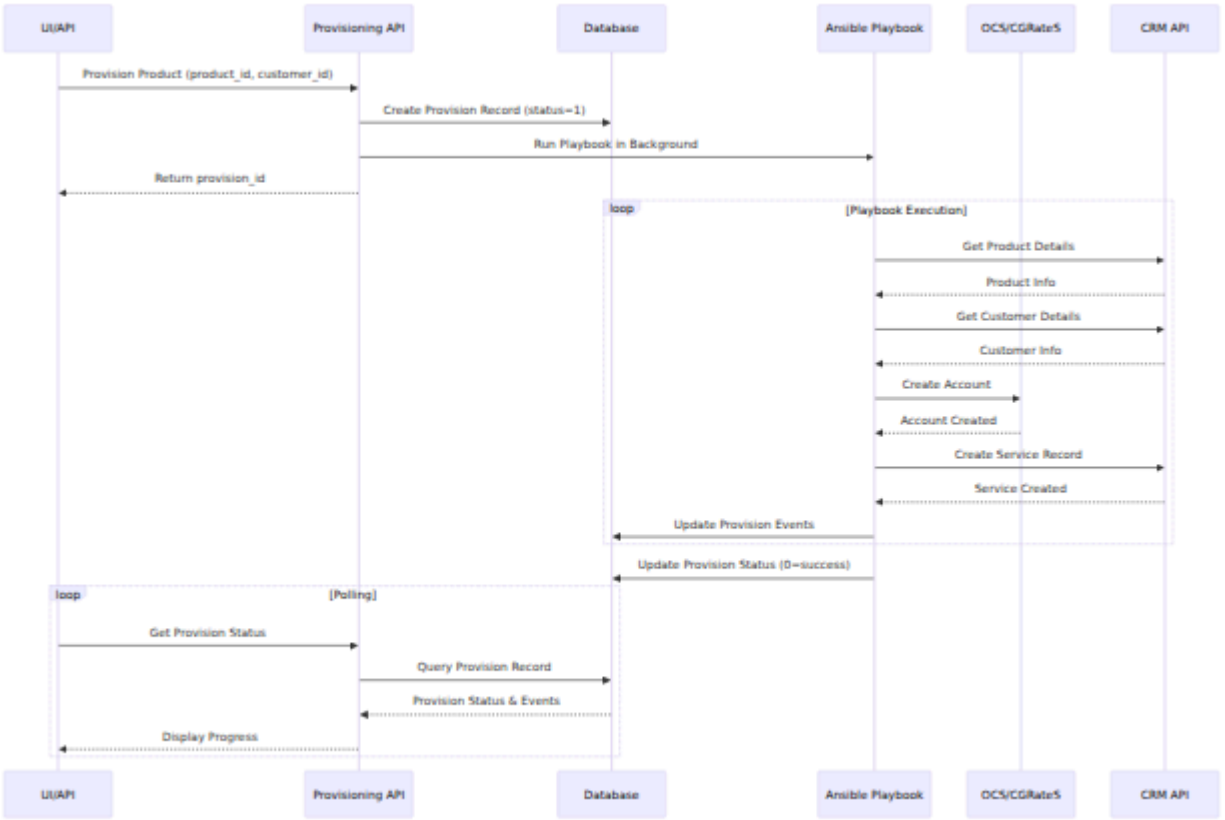
   **Colonnes du Tableau :**

   - **Nom du Produit** - Récupéré en consultant `ProductID` depuis l'ActionPlanId
   - **Coût** - Affiche `retail_cost` depuis la définition du produit
   - **Date de Renouvellement** - Affiche `custom_NextExecTime_hr` (lisible par l'homme)
   - **Actions** - Deux boutons :
     - **Renouveler Maintenant** - Provisionner immédiatement l'ajout/renouvellement (contourne l'attente de

# Système de Provisionnement

OmniCRM utilise **Ansible** pour automatiser le provisionnement, la configuration et le déprovisionnement des services clients. Le système de provisionnement est conçu pour être flexible, permettant des flux de travail complexes tout en maintenant la cohérence et la fiabilité.

Voir aussi : Provisionnement de Carte SIM pour des informations détaillées sur le provisionnement des cartes SIM mobiles, y compris les SIM physiques et les eSIM.



**REMARQUE**

Pour un guide complet du parcours produit-à-service avec des exemples détaillés de playbook Ansible, des stratégies de tarification et des scénarios du monde réel, voir Guide Complet du Cycle de Vie des Produits.

# Aperçu

Lorsqu'un produit est commandé ou qu'un service doit être configuré, OmniCRM crée un **Travail de Provisionnement** qui exécute un ou plusieurs playbooks Ansible. Ces playbooks interagissent avec divers systèmes backend (OCS/CGRateS, équipements réseau, API, etc.) pour provisionner complètement le service.

Le système de provisionnement prend en charge deux flux de travail principaux :

1. **Provisionnement Standard** - Déclenché par le personnel ou les clients via l'UI/API
2. **Provisionnement Simple** - Déclenché par des systèmes externes comme OCS pour des opérations automatisées

# Valeurs de Statut de Provisionnement

Les travaux de provisionnement et les tâches individuelles peuvent avoir les statuts suivants :

- **Statut 0 (Succès)** - Le travail de provisionnement s'est terminé avec succès
- **Statut 1 (En cours)** - Le travail de provisionnement ou la tâche est actuellement en cours d'exécution
- **Statut 2 (Échoué - Critique)** - Un échec critique s'est produit, ce qui a entraîné l'échec du provisionnement
- **Statut 3 (Échoué - Ignoré)** - Une tâche a échoué mais avait `ignore_errors: true`, donc le provisionnement a continué

Lorsqu'un travail de provisionnement échoue, OmniCRM envoie des notifications par e-mail à la liste de notification de défaillance configurée avec des informations d'erreur détaillées.

# Comment les Produits Pilotent le Provisionnement

La définition du **Produit** est le plan pour ce qui est provisionné et comment. Lorsqu'un utilisateur sélectionne un produit à provisionner, le système lit plusieurs champs clés de la définition du produit pour déterminer quoi faire.

## Champs de Produit Utilisés dans le Provisionnement

Une définition de produit contient :

- `provisioning_play` - Le nom du playbook Ansible à exécuter (sans l'extension .yaml)
- `provisioning_json_vars` - Chaîne JSON contenant des variables par défaut à passer à Ansible

- `inventory_items_list` - Liste des types d'inventaire qui doivent être assignés (par exemple, `['SIM Card', 'Mobile Number']`)
- `product_id`, `product_name`, champs de tarification - Passés automatiquement au playbook

## Exemple de Définition de Produit

```
{
  "product_id": 1,
  "product_slug": "Mobile-SIM",
  "product_name": "Mobile SIM Only",
  "provisioning_play": "play_psim_only",
  "provisioning_json_vars": "{\"iccid\": \"\", \"msisdn\": \"\"}",
  "inventory_items_list": "['SIM Card', 'Mobile Number']",
  "retail_cost": 0,
  "retail_setup_cost": 0,
  "wholesale_cost": 3,
  "wholesale_setup_cost": 1
}
```

## Du Produit au Travail de Provisionnement

Lorsque le provisionnement est initié, le système :

1. **Charge le playbook** spécifié dans `provisioning_play`

   Le système recherche `OmniCRM-API/Provisioners/plays/play_psim_only.yaml`

2. **Fusionne les variables** de plusieurs sources dans `extra_vars` :

   i. **À partir de provisioning_json_vars** : `{"iccid": "", "msisdn": ""}`
   ii. **À partir du corps de la requête** : Toute variable supplémentaire fournie par l'utilisateur/API
   iii. **À partir des champs de produit** : `product_id`, `customer_id`, etc.
   iv. **À partir de l'authentification** : `access_token` ou configuration pour `refresh_token`

3. **Assigne l'inventaire** basé sur `inventory_items_list`

   Avant d'exécuter le playbook, l'UI/API demande la sélection de l'inventaire :

   - **SIM Card** - L'utilisateur sélectionne une SIM disponible dans l'inventaire
   - **Mobile Number** - L'utilisateur sélectionne un numéro de téléphone disponible

   Les IDs d'inventaire sélectionnés sont ajoutés à `extra_vars` avec le type d'inventaire comme clé :

```
extra_vars = {
    "product_id": 1,
    "customer_id": 456,
    "SIM Card": 789,        # inventory_id de la SIM sélectionnée
    "Mobile Number": 101,   # inventory_id du numéro de téléphone sélectionné
    "iccid": "",            # À partir de provisioning_json_vars
    "msisdn": "",           # À partir de provisioning_json_vars
    "access_token": "eyJ..."
}
```

4. **Passe tout à Ansible** via `hostvars[inventory_hostname]`

   À l'intérieur du playbook, les variables sont accessibles comme :

```
- name: Get inventory_id for SIM Card
  set_fact:
    inventory_id_sim_card: "{{ hostvars[inventory_hostname]['SIM Card'] | int }}"
  when: "'SIM Card' in hostvars[inventory_hostname]"
```

## Comment les Playbooks Utilisent les Variables d'Inventaire

Une fois que le playbook a les IDs d'inventaire, il récupère les détails complets de l'inventaire à partir de l'API :

```
- name: Get SIM Card Details from Inventory
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ inventory_id_sim_card }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
    return_content: yes
  register: sim_card_response

- name: Extract ICCID and IMSI from inventory
  set_fact:
    iccid: "{{ sim_card_response.json.iccid }}"
    imsi: "{{ sim_card_response.json.imsi }}"

- name: Get Phone Number Details from Inventory
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ inventory_id_phone_number }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
    return_content: yes
  register: phone_number_response

- name: Extract MSISDN
  set_fact:
    msisdn: "{{ phone_number_response.json.msisdn }}"
```

Le playbook peut ensuite utiliser ces valeurs pour :

- Provisionner la carte SIM sur le HSS avec l'IMSI
- Configurer le numéro de téléphone dans le système de facturation
- Assigner les éléments d'inventaire au client

- Créer l'enregistrement de service avec ces détails

## Exemple du Monde Réel : Provisionnement de Carte SIM Mobile

À partir de `play_psim_only.yaml`, voici comment il utilise les données de produit et d'inventaire :

```
- name: Get Product information from CRM API
  uri:
    url: "{{ crm_config.crm.base_url }}/crm/product/product_id/{{
product_id }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
    return_content: yes
  register: api_response_product

- name: Set package facts from product
  set_fact:
    package_name: "{{ api_response_product.json.product_name }}"
    package_comment: "{{ api_response_product.json.comment }}"
    setup_cost: "{{ api_response_product.json.retail_setup_cost
}}"
    monthly_cost: "{{ api_response_product.json.retail_cost }}"

- name: Set inventory_id_sim_card if SIM Card was selected
  set_fact:
    inventory_id_sim_card: "{{ hostvars[inventory_hostname]['SIM
Card'] | int }}"
  when: "'SIM Card' in hostvars[inventory_hostname]"

- name: Set inventory_id_phone_number if Mobile Number was
selected
  set_fact:
    inventory_id_phone_number: "{{ hostvars[inventory_hostname]
['Mobile Number'] | int }}"
  when: "'Mobile Number' in hostvars[inventory_hostname]"

- name: Get SIM Card details from inventory
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ inventory_id_sim_card }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
    return_content: yes
  register: sim_inventory_response

- name: Get Phone Number details from inventory
  uri:
```

```
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ inventory_id_phone_number }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
    return_content: yes
  register: phone_inventory_response

- name: Extract values from inventory
  set_fact:
    iccid: "{{ sim_inventory_response.json.iccid }}"
    imsi: "{{ sim_inventory_response.json.imsi }}"
    msisdn: "{{ phone_inventory_response.json.msisdn }}"
    ki: "{{ sim_inventory_response.json.ki }}"
    opc: "{{ sim_inventory_response.json.opc }}"

- name: Provision subscriber on HSS
  uri:
    url: "http://{{ hss_server }}/subscriber/{{ imsi }}"
    method: PUT
    body_format: json
    body:
      {
        "imsi": "{{ imsi }}",
        "msisdn": "{{ msisdn }}",
        "ki": "{{ ki }}",
        "opc": "{{ opc }}",
        "enabled": true
      }
    status_code: 200

- name: Assign inventory to customer
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ inventory_id_sim_card }}"
    method: PATCH
    headers:
      Authorization: "Bearer {{ access_token }}"
    body_format: json
    body:
      {
        "customer_id": {{ customer_id }},
        "item_state": "Assigned"
```

```
    }
  status_code: 200
```

Cela démontre le flux complet :

1. La définition du produit spécifie `provisioning_play: "play_psim_only"`
2. Le produit nécessite `inventory_items_list: ['SIM Card', 'Mobile Number']`
3. L'utilisateur sélectionne des éléments d'inventaire lors du provisionnement
4. Les IDs d'inventaire sont passés au playbook comme `extra_vars`
5. Le playbook récupère les détails complets de l'inventaire à partir de l'API
6. Le playbook utilise les données d'inventaire pour configurer l'équipement réseau
7. Le playbook marque l'inventaire comme assigné au client

## Rollback et Nettoyage : Modèle de Meilleure Pratique

**Meilleure Pratique Critique** : Le même playbook doit gérer à la fois le rollback de provisionnement échoué et le déprovisionnement intentionnel en utilisant la structure `block` et `rescue` d'Ansible.

### Structure du Playbook

À partir de `play_psim_only.yaml` :

```yaml
- name: OmniCore Service Provisioning 2024
  hosts: localhost
  gather_facts: no
  become: False

  tasks:
    - name: Main block
      block:
        # --- TÂCHES DE PROVISIONNEMENT ---
        - name: Get Product information
          uri: ...

        - name: Create account in OCS
          uri: ...

        - name: Provision subscriber on HSS
          uri: ...

        - name: Create service record
          uri: ...

        # ... de nombreuses autres tâches de provisionnement ...

      rescue:
        # --- TÂCHES DE NETTOYAGE ---
        # Cette section s'exécute lorsque :
        # 1. Toute tâche dans le bloc échoue (rollback)
        # 2. action == "deprovision" (nettoyage intentionnel)

        - name: Get Inventory items linked to this service
          uri:
            url: "{{ crm_config.crm.base_url
}}/crm/inventory/customer_id/{{ customer_id }}"
            method: GET
          register: inventory_api_response
          ignore_errors: True

        - name: Return inventory to pool
          uri:
            url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ item.inventory_id }}"
            method: PATCH
            body_format: json
```

```
      body:
        service_id: null
        customer_id: null
        item_state: "Used"
    with_items: "{{ inventory_api_response.json.data }}"
    ignore_errors: True

  - name: Delete Account from Charging
    uri:
      url: "http://{{ crm_config.ocs.OCS }}/jsonrpc"
      method: POST
      body:
        {
          "method": "ApierV1.RemoveAccount",
          "params": [{
              "Tenant": "{{ crm_config.ocs.ocsTenant }}",
              "Account": "{{ service_uuid }}"
          }]
        }
    ignore_errors: True

  - name: Delete Attribute Profile
    uri:
      url: "http://{{ crm_config.ocs.OCS }}/jsonrpc"
      method: POST
      body:
        {
          "method": "APIerSv1.RemoveAttributeProfile",
          "params": [{
              "ID": "ATTR_ACCOUNT_{{ service_uuid }}"
          }]
        }
    ignore_errors: True

  - name: Remove Resource Profile
    uri: ...
    ignore_errors: True

  - name: Remove Filters
    uri: ...
    ignore_errors: True

  - name: Deprovision Subscriber from HSS
    uri:
```

```
            url: "{{ item.key }}/subscriber/{{
item.value.subscriber_id }}"
            method: DELETE
          loop: "{{ hss_subscriber_data | dict2items }}"
          ignore_errors: True
          when:
            - deprovision_subscriber | bool == true

        - name: Patch Subscriber to Dormant State
          uri:
            url: "{{ item.key }}/subscriber/{{
item.value.subscriber_id }}"
            method: PATCH
            body:
              {
                "enabled": true,
                "msisdn": "9999{{ imsi[-10:] }}",  # Numéro fictif
                "ue_ambr_dl": 9999999,              # Inutilisable
                "ue_ambr_ul": 9999999
              }
          loop: "{{ hss_subscriber_data | dict2items }}"
          when:
            - deprovision_subscriber | default(false) | bool ==
false

        # L'assertion finale détermine le succès ou l'échec
        - name: Set status to "Success" if Manual deprovision /
Fail if failed provision
          assert:
            that:
              - action == "deprovision"
```

**Pourquoi Ce Modèle Est une Meilleure Pratique**

**1. Pas de Duplication de Code**

Les mêmes tâches de nettoyage gèrent les deux scénarios :

- **Échec de Provisionnement (Rollback)** : Si une tâche dans le `block` échoue, la section `rescue` s'exécute automatiquement
- **Déprovisionnement Intentionnel** : Lorsqu'il est appelé avec `action: "deprovision"`, le playbook passe immédiatement à `rescue`

**2. Nettoyage Complet Garanti**

Lorsqu'un provisionnement échoue en cours de route, la section de secours garantit :

- Tous les comptes OCS créés sont supprimés
- Toutes les entrées d'équipement réseau configurées sont supprimées
- L'inventaire assigné est retourné à la piscine
- Les abonnés HSS sont supprimés ou mis en état dormant
- Aucun provisionnement partiel ne reste dans aucun système

Cela empêche les ressources "orphelines" qui :

- Consomment de l'inventaire sans être suivies
- Créent des comptes de facturation qui ne sont pas liés aux services
- Provoquent de la confusion lors du dépannage
- Gaspillent des ressources réseau

**3. Gestion des Échecs Gracieuse avec ignore_errors**

Remarquez que chaque tâche de nettoyage utilise `ignore_errors: True`. C'est intentionnel car :

- Lors du rollback, certaines ressources peuvent ne pas avoir encore été créées
- Nous voulons tenter toutes les tâches de nettoyage même si certaines échouent
- L'assertion finale détermine le succès/échec global

Par exemple, si le provisionnement échoue à "Créer un compte dans OCS", le nettoyage tentera de :

- Supprimer le compte OCS (échouera, mais sera ignoré)
- Supprimer les profils d'attributs (échouera, mais sera ignoré)
- Retourner l'inventaire (réussit)
- Supprimer l'abonné HSS (peut ne pas exister, ignoré)

**4. Distinction entre Déprovisionnement et Rollback**

L'assertion finale à la fin de `rescue` est astucieuse :

```
- name: Set status to "Success" if Manual deprovision / Fail if
failed provision
  assert:
    that:
      - action == "deprovision"
```

Cela signifie :

- **Si ``action == "deprovision"``** : L'assertion passe, le playbook réussit (statut 0)
- **Si ``action`` n'est pas défini ou != "deprovision"** : L'assertion échoue, le playbook échoue (statut 2)

Ainsi, le même code de nettoyage entraîne différents statuts de travail de provisionnement en fonction de l'intention.

**5. Nettoyage Conditionnel Basé sur le Type de Service**

Certaines tâches de nettoyage utilisent des conditionnels pour gérer différents scénarios :

```
- name: Deprovision Subscriber from HSS
  uri: ...
  when:
    - deprovision_subscriber | bool == true

- name: Patch Subscriber to Dormant State
  uri: ...
  when:
    - deprovision_subscriber | default(false) | bool == false
```

Cela permet un nettoyage flexible :

- **Suppression complète** : Lorsque les SIM sont dédiées aux clients (`deprovision_subscriber: true`)
- **État dormant** : Lorsque les SIM sont réutilisables et doivent rester dans le HSS (`deprovision_subscriber: false`)

**Comment Utiliser Ce Modèle**

**Pour le Provisionnement :**

```
{
  "product_id": 1,
  "customer_id": 456,
  "provisioning_play": "play_psim_only"
}
```

Si le provisionnement échoue, un rollback automatique se produit via `rescue`.

**Pour le Déprovisionnement :**

```
{
  "service_id": 123,
  "service_uuid": "Service_abc123",
  "action": "deprovision",
  "provisioning_play": "play_psim_only"
}
```

Le playbook passe directement à la section `rescue`, exécute tout le nettoyage et réussit.

**Résumé des Avantages**

 **Source unique de vérité** : Un playbook gère le provisionnement et le déprovisionnement  **Opérations atomiques** : Soit entièrement provisionné, soit entièrement nettoyé  **Pas de ressources orphelines** : Les provisions échouées ne laissent aucune trace  **Maintenance facilitée** : Les modifications de la logique de provisionnement s'appliquent automatiquement au nettoyage  **Réduction des erreurs** : Aucune chance que le code de provisionnement et de déprovisionnement soit désynchronisé  **Testable** : Peut tester la logique de déprovisionnement en exécutant avec `action: "deprovision"`

Ce modèle doit être suivi dans tous les playbooks de provisionnement pour garantir fiabilité et cohérence.

## Surcharge des Variables de Produit

Les `provisioning_json_vars` peuvent être surchargées au moment du provisionnement. Par exemple, un produit pourrait définir :

```
{
  "provisioning_json_vars": "{\"monthly_cost\": 50,
\"data_limit_gb\": 100}"
}
```

Mais lors du provisionnement, vous pouvez les surcharger :

```
{
  "product_id": 1,
  "customer_id": 456,
  "monthly_cost": 45,
  "data_limit_gb": 150
}
```

Les `extra_vars` fusionnés utiliseront les valeurs surchargées. Cela permet :

- Tarification personnalisée pour des clients spécifiques
- Différents limites de données basées sur des promotions
- Tests avec différents paramètres sans modifier le produit

## Produits Sans Inventaire

Tous les produits ne nécessitent pas d'inventaire. Par exemple, un addon de données ou un basculement de fonctionnalité pourrait avoir :

```
{
  "product_id": 10,
  "product_name": "Extra 10GB Data",
  "provisioning_play": "play_local_data_addon",
  "provisioning_json_vars": "{\"data_gb\": 10}",
  "inventory_items_list": "[]"
}
```

Dans ce cas, le playbook reçoit :

```
extra_vars = {
    "product_id": 10,
    "customer_id": 456,
    "service_id": 123,  # Service auquel ajouter des données
    "data_gb": 10,
    "access_token": "eyJ..."
}
```

Le playbook ajoute simplement les données au service existant sans avoir besoin d'éléments d'inventaire.

# Flux de Travail de Provisionnement Standard

Le provisionnement standard est initié lorsque :

- Un membre du personnel ajoute un service à un client depuis l'UI
- Un client commande un service via le portail d'auto-assistance
- L'API est appelée directement avec `PUT /crm/provision/`

## Lorsque Vous Cliquez sur "Provisionner"

Voici le flux complet qui se produit lorsqu'un utilisateur clique sur le bouton "Provisionner" :

### 1. L'UI Affiche la Sélection de Produit

L'utilisateur sélectionne un produit dans le catalogue de produits. Le produit contient :

- `provisioning_play` - Quel playbook Ansible exécuter
- `inventory_items_list` - Inventaire requis (par exemple, `['SIM Card', 'Mobile Number']`)
- `provisioning_json_vars` - Variables par défaut

### 2. Sélecteur d'Inventaire (Si Nécessaire)

Si `inventory_items_list` n'est pas vide, une fenêtre modale apparaît montrant des listes déroulantes pour chaque type d'inventaire. L'utilisateur doit sélectionner des éléments d'inventaire disponibles avant de continuer.

### 3. Bouton de Provisionnement Cliqué

JavaScript envoie une requête `PUT /crm/provision/` :

```
PUT /crm/provision/
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...
Content-Type: application/json

{
  "product_id": 42,
  "customer_id": 123,
  "SIM Card": 5001,
  "Mobile Number": 5002
}
```

### 4. L'API Reçoit la Requête

Le point de terminaison de provisionnement :

- Valide l'authentification (jeton Bearer, clé API ou liste blanche d'IP)
- Vérifie que l'utilisateur a la permission `CREATE_PROVISION`
- Extrait `initiating_user` du jeton
- Charge la définition du produit depuis la base de données
- Récupère le chemin du playbook : `OmniCRM-API/Provisioners/plays/play_psim_only.yaml`

### 5. Variables Fusionnées

Le système combine les variables de plusieurs sources :

```
# À partir du produit
product_vars = json.loads(product['provisioning_json_vars'])
# À partir du corps de la requête
request_vars = request_body_json
# Ajouté par le système
system_vars = {
    'product_id': 42,
    'customer_id': 123,
    'access_token': auth_token,  # Voir la section
d'authentification ci-dessous
    'initiating_user': 7
}
# Final fusionné
extra_vars = {**product_vars, **request_vars, **system_vars}
```

### 6. Enregistrement de Provision Créé

Enregistrement de base de données créé avec statut 1 (En cours) :

```
provision = {
    'provision_id': 456,
    'customer_id': 123,
    'product_id': 42,
    'provisioning_play': 'play_psim_only',
    'provisioning_json_vars': json.dumps(extra_vars),
    'provisioning_status': 1,  # En cours
    'task_count': 85,
    'initiating_user': 7,
    'created': '2025-01-10T14:30:00Z'
}
```

### 7. Fil de Fond Généré

Le système génère un fil de fond pour exécuter le playbook de manière asynchrone :

- Playbook : `play_psim_only.yaml`
- Variables : `extra_vars` fusionnées
- ID de Provision : 456 (pour le suivi de statut)

- Jeton de rafraîchissement : passé pour le rafraîchissement de jeton pendant l'exécution

**8. L'API Retourne Immédiatement**

Réponse retournée à l'UI avec `provision_id` :

```
{
  "provision_id": 456,
  "provisioning_status": 1,
  "message": "Provisioning job created"
}
```

**9. L'UI Interroge pour des Mises à Jour**

L'UI commence à interroger `GET /crm/provision/provision_id/456` toutes les 3 secondes pour vérifier le statut. La réponse inclut :

```
{
  "provision_id": 456,
  "provisioning_status": 1,
  "task_count": 12,
  "provisioning_result_json": [
    {
      "event_number": 1,
      "event_name": "Get Product information from CRM API",
      "provisioning_status": 0,
      "timestamp": "2024-01-15T10:30:05"
    },
    {
      "event_number": 2,
      "event_name": "Assign SIM Card from inventory",
      "provisioning_status": 1,
      "timestamp": "2024-01-15T10:30:07"
    }
  ]
}
```

**10. Ansible S'exécute en Arrière-Plan**

Le playbook exécute les tâches séquentiellement :

- Chaque tâche terminée crée un enregistrement `Provision_Event` dans la base de données
- L'événement inclut : nom de la tâche, statut (0=succès, 2=échec, 3=échec mais ignoré), résultat JSON
- L'UI affiche la progression en temps réel montrant les tâches terminées et la tâche actuellement en cours d'exécution
- Les tâches échouées affichent des messages d'erreur dans les détails de l'événement

**Suivi dans l'UI :**

Pendant que le provisionnement est en cours (statut 1), les utilisateurs peuvent voir :

- **Page Détails du Service** - Affiche le badge de statut de provisionnement (En cours/Succès/Échoué)
- **Journal d'Activité** - Liste tous les événements de provisionnement avec des horodatages
- **Vue Détails de Provision** - Montre la progression tâche par tâche avec des options d'expansion/repli pour les détails

Exemple d'affichage :

Statut de Provisionnement : En cours (8 sur 12 tâches complétées)

✓ Get Product information from CRM API ✓ Fetch Customer details ✓ Assign SIM Card from inventory (ICCID: 8991101200003204510) ✓ Assign Mobile Number (555-0123) ⟳ Create account in OCS/CGRateS (en cours...) □ Configure network policies □ Create service record ...

### 11. Le Provisionnement se Termine

Statut final défini :

- `provisioning_status: 0` - Succès
- `provisioning_status: 2` - Échec (erreur critique)

L'UI arrête l'interrogation et affiche le résultat :

- **Succès** : Coche verte, service marqué Actif, l'utilisateur peut voir les détails du service
- **Échec** : X rouge, message d'erreur affiché, option de réessayer ou de contacter le support
- **Notification par e-mail** : En cas d'échec, un e-mail est envoyé à `provisioning.failure_list` dans la configuration

# Authentification et Autorisation

### Suivi des Utilisateurs

Chaque travail de provisionnement suit quel utilisateur l'a initié :

- **Initié par l'utilisateur** : Le champ `initiating_user` est défini sur l'ID de l'utilisateur à partir de son jeton JWT
- **Authentification par clé API** : Utilise l'ID du premier utilisateur administrateur
- **Authentification par liste blanche d'IP** : Utilise l'ID du premier utilisateur administrateur

### Vérifications de Permission

Le système vérifie les permissions avant de permettre le provisionnement :

- Le personnel a besoin de la permission `CREATE_PROVISION`
- Les clients ne peuvent provisionner des services que pour leur propre compte (`VIEW_OWN_PROVISION` permission)

### Comment Ansible S'authentifie avec l'API CRM

Les playbooks Ansible doivent effectuer des appels API authentifiés vers le CRM (pour récupérer les détails du produit, créer des services, mettre à jour l'inventaire, etc.). L'authentification est gérée par des **jetons Bearer** passés au playbook.

La source du `access_token` dépend de la méthode d'authentification utilisée pour appeler l'API de provisionnement :

**Méthode 1 : Connexion Utilisateur (Jeton Bearer)**

Lorsqu'un utilisateur se connecte via l'UI web :

1. L'utilisateur s'authentifie : `POST /crm/auth/login`
2. Reçoit un `access_token` JWT (à durée de vie courte, 15-30 min) et un `refresh_token` (à longue durée de vie)
3. Fait une demande de provisionnement avec le jeton Bearer dans l'en-tête
4. L'API de provisionnement extrait le jeton de l'en-tête `Authorization: Bearer ...`
5. Stocke le jeton pour une utilisation pendant le provisionnement
6. Passe à Ansible comme variable `access_token`

**Flux d'Implémentation :**

Le système extrait le jeton Bearer de l'en-tête d'autorisation, le valide et le décode, puis le stocke pour une utilisation pendant le provisionnement. Le jeton est ensuite passé au playbook comme une variable supplémentaire afin qu'il puisse authentifier les appels API suivants.

**Méthode 2 : Clé API (En-tête X-API-KEY)**

Pour les systèmes automatisés utilisant des clés API :

1. Le système fait une demande : `PUT /crm/provision/` avec l'en-tête `X-API-KEY: your-api-key...`
2. L'API de provisionnement valide la clé API par rapport à `crm_config.yaml`
3. **Génère un nouveau jeton JWT à la volée** pour le premier utilisateur administrateur
4. Stocke le jeton pour le provisionnement
5. Passe à Ansible

**Pourquoi Générer un Jeton ?**

Les clés API sont des chaînes, pas des JWT. Les playbooks appellent des points de terminaison API s'attendant à une authentification JWT. Donc :

- Valider la clé API

- Si valide et a le rôle `admin`, générer un JWT temporaire
- Utiliser l'ID du premier utilisateur administrateur comme sujet du JWT
- Le jeton permet au playbook de faire des appels API authentifiés

**Flux d'Implémentation :**

Le système valide la clé API par rapport aux clés configurées. Si valide et a le rôle administrateur, il génère un JWT temporaire en utilisant l'identité d'un utilisateur administrateur, puis le stocke pour une utilisation lors du provisionnement.

**Méthode 3 : Liste Blanche d'IP**

Pour les systèmes internes de confiance sur des réseaux privés :

1. Le système fait une demande depuis une IP sur liste blanche (par exemple, 192.168.1.100)
2. L'API de provisionnement vérifie l'IP du client par rapport à `ip_whitelist` dans `crm_config.yaml`
3. Si sur liste blanche, **génère un nouveau jeton JWT** pour le premier utilisateur administrateur
4. Stocke le jeton pour le provisionnement
5. Passe à Ansible

**Flux d'Implémentation :**

Le système vérifie l'IP du client par rapport à la liste blanche. Si autorisé, il génère un JWT pour un utilisateur administrateur et le stocke pour une utilisation lors du provisionnement.

**Utilisation du Jeton dans les Playbooks**

Chaque appel API dans le playbook inclut le jeton :

```
- name: Get Product Details
  uri:
    url: "http://localhost:5000/crm/product/product_id/{{
product_id }}"
    headers:
      Authorization: "Bearer {{ access_token }}"

- name: Create Service Record
  uri:
    url: "http://localhost:5000/crm/service/"
    method: PUT
    headers:
      Authorization: "Bearer {{ access_token }}"
    body:
      customer_id: "{{ customer_id }}"
      service_name: "Mobile Service"
```

**Expiration et Rafraîchissement du Jeton**

Les playbooks de longue durée (5-10 minutes) peuvent dépasser la durée de vie de l'`access_token` (expiration de 15-30 min). Pour les provisions initi��es par l'utilisateur, le système passe à la fois `access_token` et `refresh_token` :

```
refresh_token = request.cookies.get('refresh_token')
run_playbook(playbook_path, extra_vars, provision_id,
refresh_token=refresh_token)
```

Si `access_token` expire, le coureur de playbook peut :

1. Détecter une réponse 401 Non autorisée
2. Appeler `POST /crm/auth/refresh` avec `refresh_token`
3. Recevoir un nouvel `access_token`
4. Réessayer la demande échouée

Pour l'authentification par clé API/liste blanche d'IP, les jetons générés peuvent avoir une expiration plus longue (1-2 heures) car ce sont des systèmes automatisés de confiance.

# Le Processus de Provisionnement

1. **Création de Travail**

   Lorsqu'une demande de provisionnement est reçue, le système :

   - Valide la demande et vérifie les permissions
   - Charge le playbook Ansible spécifié dans la définition du produit
   - Crée un enregistrement `Provision` dans la base de données avec le statut 1 (En cours)
   - Extrait les variables de la définition du produit et du corps de la demande
   - Capture les jetons d'authentification pour l'accès API

2. **Gestion des Jetons**

   Les playbooks Ansible doivent s'authentifier avec l'API CRM pour récupérer des données et effectuer des modifications. Le système de provisionnement gère cela de deux manières :

   - **Jeton Bearer (JWT)** : Pour le provisionnement initié par l'utilisateur, le `refresh_token` de la demande est utilisé pour générer des jetons d'accès frais pendant l'exécution du playbook
   - **Authentification par Clé API/IP** : Pour les systèmes automatisés, un `access_token` est passé directement au playbook

3. **Exécution en Arrière-Plan**

   Le playbook s'exécute dans un fil de fond. Cela permet à l'API de retourner immédiatement pendant que le provisionnement continue de manière asynchrone.

   Pendant l'exécution :

   - Chaque tâche terminée/échouée crée un enregistrement `Provision_Event`
   - Le gestionnaire d'événements surveille les échecs critiques par rapport aux échecs ignorés

- Les mises à jour de statut en temps réel sont écrites dans la base de données
- L'UI peut interroger pour des mises à jour via `GET /crm/provision/provision_id/<id>`

4. **Exécution du Playbook**

   Le playbook Ansible effectue généralement ces opérations :

   - Récupère les informations sur le produit à partir de l'API
   - Récupère les informations sur le client à partir de l'API
   - Assigne des éléments d'inventaire (cartes SIM, adresses IP, numéros de téléphone, etc.)
   - Crée des comptes dans OCS/OCS
   - Configure l'équipement réseau
   - Crée l'enregistrement de service dans l'API CRM
   - Ajoute des transactions de coût de configuration
   - Envoie des e-mails/SMS de bienvenue aux clients

5. **Gestion des Erreurs**

   Les playbooks Ansible utilisent des sections `block` et `rescue` pour le rollback :

   - Si une tâche critique échoue, la section de secours supprime le provisionnement partiel
   - Les tâches avec `ignore_errors: true` sont marquées comme statut 3 et ne font pas échouer le travail
   - Les erreurs fatales (erreurs de syntaxe YAML, échecs de connexion) créent un événement d'erreur spécial avec des informations de débogage

# Exemple : Playbook de Provisionnement Standard

Voici un exemple à partir de `play_simple_service.yaml` :

```
- name: Simple Provisioning Play
  hosts: localhost
  gather_facts: no
  become: False

  tasks:
    - name: Main block
      block:
      - name: Get Product information from CRM API
        uri:
          url: "http://localhost:5000/crm/product/product_id/{{
product_id }}"
          method: GET
          headers:
            Authorization: "Bearer {{ access_token }}"
          return_content: yes
          validate_certs: no
        register: api_response_product

      - name: Set package facts
        set_fact:
          package_name: "{{ api_response_product.json.product_name
}}"
          setup_cost: "{{
api_response_product.json.retail_setup_cost }}"
          monthly_cost: "{{ api_response_product.json.retail_cost
}}"

      - name: Generate Service UUID
        set_fact:
          service_uuid: "Service_{{ 99999999 | random | to_uuid
}}"

      - name: Create account in OCS
        uri:
          url: "http://{{ crm_config.ocs.OCS }}/jsonrpc"
          method: POST
          body_format: json
          body:
            {
              "method": "ApierV2.SetAccount",
              "params": [{
                  "Tenant": "{{ crm_config.ocs.ocsTenant }}",
```

```
                  "Account": "{{ service_uuid }}",
                  "ActionPlanIds": [],
                  "ExtraOptions": { "AllowNegative": false,
"Disabled": false }
                }]
              }
            status_code: 200
          register: response

      - name: Add Service via API
          uri:
            url: "http://localhost:5000/crm/service/"
            method: PUT
            body_format: json
            headers:
              Authorization: "Bearer {{ access_token }}"
            body:
              {
                "customer_id": "{{ customer_id }}",
                "product_id": "{{ product_id }}",
                "service_name": "Service: {{ service_uuid }}",
                "service_uuid": "{{ service_uuid }}",
                "service_status": "Active",
                "retail_cost": "{{ monthly_cost | float }}"
              }
            status_code: 200
          register: service_creation_response

      - name: Add Setup Cost Transaction
          uri:
            url: "http://localhost:5000/crm/transaction/"
            method: PUT
            headers:
              Authorization: "Bearer {{ access_token }}"
            body_format: json
            body:
              {
                "customer_id": {{ customer_id | int }},
                "service_id": {{
service_creation_response.json.service_id | int }},
                "title": "{{ package_name }} - Setup Costs",
                "retail_cost": "{{ setup_cost | float }}"
              }
          register: api_response_transaction
```

```
    rescue:
      - name: Remove account in OCS
        uri:
          url: "http://{{ crm_config.ocs.OCS }}/jsonrpc"
          method: POST
          body_format: json
          body:
            {
              "method": "ApierV2.RemoveAccount",
              "params": [{
                  "Tenant": "{{ crm_config.ocs.ocsTenant }}",
                  "Account": "{{ service_uuid }}"
              }]
            }
          status_code: 200

      - name: Fail the provision
        assert:
          that:
            - false
```

Ce playbook démontre le flux typique :

1. Récupérer les détails du produit à partir de l'API CRM
2. Générer un UUID de service unique
3. Créer le compte de facturation dans OCS
4. Créer l'enregistrement de service via l'API CRM
5. Ajouter des transactions de coût de configuration
6. Si quelque chose échoue, la section `rescue` supprime le compte OCS

# Flux de Travail de Provisionnement Simple

Le provisionnement simple est conçu pour les systèmes automatisés qui ont besoin de déclencher le provisionnement sans interaction utilisateur. Le cas d'utilisation le plus courant est le déclenchement de provisionnement d'addon par OCS via des Plans d'Action.

# Points de Terminaison de Provisionnement Simple

OmniCRM fournit deux points de terminaison de provisionnement simple :

- `POST /crm/provision/simple_provision_addon/service_id/<id>/product_id/<id>`

  Pour le provisionnement automatisé d'addon (par exemple, frais récurrents, recharges automatiques)

- `POST /crm/provision/simple_provision_addon_recharge/service_id/<id>/product_id/<id>`

  Pour des opérations de recharge rapides qui nécessitent un retour immédiat

# Authentification pour le Provisionnement Simple

Les points de terminaison de provisionnement simple utilisent **la liste blanche d'IP** ou **les clés API** pour l'authentification :

- L'IP source de la demande est vérifiée par rapport à `ip_whitelist` dans `crm_config.yaml`
- Ou une clé API provenant de `api_keys` dans `crm_config.yaml` peut être fournie
- Un jeton d'accès est généré et passé au playbook

# Exemple : Callback de Plan d'Action OCS

OCS peut être configuré pour appeler le point de terminaison de provisionnement simple lors de l'exécution d'actions récurrentes :

```
{
  "method": "ApierV1.SetActionPlan",
  "params": [{
    "Id": "ActionPlan_Service123_Monthly_Charge",
    "ActionsId": "Action_Service123_Add_Monthly_Data",
    "Timing": {
      "Years": [],
      "Months": [],
      "MonthDays": [1],
      "Time": "00:00:00Z"
    },
    "Weight": 10,
    "ActionTriggers": [
      {
        "ThresholdType": "*min_event_counter",
        "ThresholdValue": 1,
        "ActionsID": "Action_Service123_HTTP_Callback"
      }
    ]
  }]
}
```

L'action effectue un POST HTTP vers :

Cela déclenche le playbook associé (par exemple, `play_topup_no_charge.yaml`) qui ajoute des données/crédits au service.

## Exemple : Playbook de Recharge Simple

À partir de `play_topup_monetary.yaml` :

```yaml
- name: Mobile Topup Monetary - 2024
  hosts: localhost
  gather_facts: no
  become: False

  tasks:
    - name: Get Product information from CRM API
      uri:
        url: "http://localhost:5000/crm/product/product_id/{{
product_id }}"
        method: GET
        headers:
          Authorization: "Bearer {{ access_token }}"
        return_content: yes
      register: api_response_product

    - name: Get Service information from CRM API
      uri:
        url: "http://localhost:5000/crm/service/service_id/{{
service_id }}"
        method: GET
        headers:
          Authorization: "Bearer {{ access_token }}"
        return_content: yes
      register: api_response_service

    - name: Set service facts
      set_fact:
        service_uuid: "{{ api_response_service.json.service_uuid
}}"
        customer_id: "{{ api_response_service.json.customer_id }}"
        package_name: "{{ api_response_product.json.product_name
}}"
        monthly_cost: "{{ api_response_product.json.retail_cost
}}"

    - name: Get Customer Payment Method
      uri:
        url: "http://localhost:5000/api/payments/methods/default?
customer_id={{ customer_id }}"
        method: GET
        headers:
          Authorization: "Bearer {{ access_token }}"
```

```
      return_content: yes
    register: api_response_payment_method

- name: Charge customer
  uri:
    url: "http://localhost:5000/api/payments/charge"
    method: POST
    headers:
      Authorization: "Bearer {{ access_token }}"
    body_format: json
    body:
      {
        "customer_id": "{{ customer_id | int }}",
        "amount": "{{ monthly_cost }}",
        "currency": "USD",
        "metadata": {
          "description": "{{ package_name }} topup",
          "service_id": "{{ service_id | int }}"
        }
      }
  register: api_response_payment

- name: Add monetary balance to OCS
  uri:
    url: "http://{{ crm_config.ocs.OCS }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.AddBalance",
        "params": [{
            "Tenant": "{{ crm_config.ocs.ocsTenant }}",
            "Account": "{{ service_uuid }}",
            "BalanceType": "*monetary",
            "Balance": {
                "Value": "{{ monthly_cost | float * 100 }}",
                "ExpiryTime": "+4320h"
            }
        }]
      }
    status_code: 200

- name: Add Transaction to CRM
  uri:
```

```
        url: "http://localhost:5000/crm/transaction/"
        method: PUT
        headers:
          Authorization: "Bearer {{ access_token }}"
        body_format: json
        body:
          {
            "customer_id": {{ customer_id | int }},
            "service_id": {{ service_id | int }},
            "title": "{{ package_name }}",
            "retail_cost": "{{ monthly_cost | float }}"
          }

    - name: Send Notification SMS
      uri:
        url: "http://sms-gateway/SMS/plaintext/{{ api_key }}"
        method: POST
        body_format: json
        body:
          {
            "source_msisdn": "YourCompany",
            "destinatination_msisdn": "{{ customer_phone }}",
            "message_body": "Thanks for topping up {{ monthly_cost
}}!"
          }
        status_code: 201
      ignore_errors: True
```

Ce playbook :

1. Récupère les détails du service et du produit à partir de l'API
2. Récupère la méthode de paiement du client
3. Charge le client via l'API Stripe
4. Ajoute un solde monétaire à OCS
5. Enregistre la transaction dans le CRM
6. Envoie un SMS de confirmation (avec `ignore_errors: True` afin que les échecs ne fassent pas échouer le travail)

# Chaînes de Provisionnement

Pour des produits complexes qui nécessitent plusieurs étapes de provisionnement, OmniCRM prend en charge les **chaînes de provisionnement**. Une chaîne exécute plusieurs playbooks séquentiellement, passant le contexte entre eux.

Cas d'utilisation exemple : Un service groupé qui provisionne :

1. Service Internet de base (crée l'enregistrement de service principal)
2. Addon IPTV (utilise le service_id de l'étape 1)
3. Addon IP statique (utilise le service_id de l'étape 1)

Le service de provisionnement :

- Interroge la base de données pour le `service_id` créé par le premier playbook
- L'injecte dans les `extra_vars` pour les playbooks suivants
- Suit chaque playbook comme un enregistrement `Provision` séparé

# Raisons d'Échec et Débogage

Lorsque le provisionnement échoue, le système capture des informations détaillées pour aider à diagnostiquer le problème.

## Scénarios d'Échec Courants

### Échecs de Tâches Critiques (Statut 2)

Ceux-ci entraînent l'échec de l'ensemble du travail de provisionnement :

- Appels API retournant des codes d'état inattendus
- Échecs d'assertions (par exemple, `assert: that: response.status == 200`)
- Éléments d'inventaire requis manquants
- Équipement réseau injoignable

- Identifiants invalides ou jetons expirés
- Erreurs OCS/OCS

**Échecs Ignorés (Statut 3)**

Ceux-ci sont enregistrés mais ne font pas échouer le travail :

- Notifications SMS/e-mail optionnelles échouant
- Recherches de données non critiques (marquées avec `ignore_errors: True`)
- Opérations de nettoyage lors du déprovisionnement

**Erreurs Fatales**

Celles-ci empêchent le playbook de s'exécuter :

- Erreurs de syntaxe YAML dans le playbook
- Variables non définies dans le playbook
- Fichiers de playbook manquants
- Échecs de connexion au contrôleur Ansible

Lorsqu'une erreur fatale se produit, le système crée un événement d'erreur spécial contenant :

- Le code de sortie Ansible
- La sortie complète (contient des détails sur les erreurs de syntaxe)
- La sortie d'erreur complète (contient des erreurs d'exécution)
- Une liste des causes courantes pour ce type d'échec
- Toutes les variables passées au playbook

# E-mails de Notification d'Erreur

Lorsque le provisionnement échoue (statut 2), un e-mail est automatiquement envoyé à la liste de notification de défaillance configurée (`provisioning.failure_list` dans `crm_config.yaml`).

L'e-mail inclut :

- Informations sur le client
- Détails du produit/service
- Résultats des tâches codés par couleur :
  - **Vert** : Tâches réussies
  - **Orange** : T��ches échouées mais ignorées
  - **Rouge** : Échecs critiques
- Pour les échecs critiques : Sortie de débogage complète incluant les corps de demande/réponse
- Pour les erreurs fatales : Sortie Ansible, messages d'erreur et causes courantes

# Surveillance des Travaux de Provisionnement

OmniCRM fournit des métriques complètes pour surveiller les performances de provisionnement. Pour des détails complets sur les métriques de provisionnement, y compris la durée des travaux, les taux de succès, les comptes de tâches et le suivi des erreurs, voir Surveillance & Métriques.

## API de Statut de Provisionnement

Pour vérifier le statut d'un travail de provisionnement :

```
GET /crm/provision/provision_id/<id>
Authorization: Bearer <token>
```

La réponse inclut :

```
{
  "provision_id": 123,
  "customer_id": 456,
  "customer_name": "John Smith",
  "product_id": 10,
  "provisioning_status": 0,
  "provisioning_play": "play_psim_only",
  "playbook_description": "OmniCore Service Provisioning 2024",
  "task_count": 85,
  "provisioning_result_json": [
    {
      "event_number": 1,
      "event_name": "Get Product information from CRM API",
      "provisioning_status": 1,
      "provisioning_result_json": "{...}"
    },
    {
      "event_number": 2,
      "event_name": "Create account in OCS",
      "provisioning_status": 1,
      "provisioning_result_json": "{...}"
    }
  ]
}
```


## Listing des Travaux de Provisionnement

Pour obtenir une liste paginée de tous les travaux de provisionnement :

```
GET /crm/provision/?
page=1&per_page=20&sort=provision_id&order=desc
Authorization: Bearer <token>
```

Prend en charge le filtrage :

```
GET /crm/provision/?filters={"provisioning_status":
[2]}&search=Mobile
Authorization: Bearer <token>
```

Cela renvoie uniquement les travaux échoués (statut 2) où la description contient "Mobile".

# Meilleures Pratiques

## Conception de Playbook

- **Utilisez toujours block/rescue** : Assurez-vous que le provisionnement partiel peut être annulé
- **Utilisez ignore_errors judicieusement** : Uniquement pour des opérations vraiment optionnelles
- **Journalisez les variables importantes** : Utilisez des tâches `debug` pour journaliser les valeurs clés pour le dépannage
- **Validez les réponses** : Utilisez `assert` pour vérifier que les réponses API sont conformes aux attentes
- **Idempotence** : Concevez les playbooks pour être réexécutables en toute sécurité

## Authentification

- **Provisionnement initié par l'utilisateur** : Utilisez toujours refresh_token pour les playbooks de longue durée
- **Provisionnement automatisé** : Utilisez la liste blanche d'IP ou les clés API avec des jetons d'accès générés
- **Expiration des jetons** : Le refresh_token garantit que les jetons d'accès sont régénérés au besoin

## Gestion des Erreurs

- **Fournir un contexte** : Inclure customer_id, service_id et détails de l'opération dans les messages d'erreur
- **Notifier de manière appropriée** : Les échecs critiques déclenchent des e-mails, mais ne spammez pas pour les échecs attendus
- **Informations de débogage** : Capturer les corps de demande/réponse complets dans les enregistrements Provision_Event

## Sécurité

- **Valider les entrées** : Vérifiez customer_id, product_id, service_id avant le provisionnement
- **Vérifications de permission** : Vérifiez que les utilisateurs ne peuvent provisionner que pour des clients autorisés
- **Données sensibles** : Utilisez le système de redaction pour supprimer les mots de passe/clés des journaux
- **Liste blanche d'IP** : Restreindre les points de terminaison simple_provision aux systèmes de confiance uniquement

## Performance

- **Exécution en arrière-plan** : Ne bloquez jamais les réponses API en attendant le provisionnement
- **Intervalles d'interrogation** : L'UI doit interroger pour des mises à jour de statut toutes les 2-5 secondes
- **Tâches parallèles** : Utilisez le parallélisme natif d'Ansible pour les opérations indépendantes
- **Mises à jour de la base de données** : Le gestionnaire d'événements met à jour la base de données en temps réel, pas besoin d'interroger pendant l'exécution

# Documentation Connexe

- `concepts_ansible` - Concepts généraux de provisionnement Ansible
- `concepts_sim_provisioning` - Provisionnement de carte SIM (physique et eSIM)
- `concepts_api` - Authentification et utilisation de l'API CRM
- `concepts_products_and_services` - Définitions de produits et de services
- `administration_inventory` - Gestion de l'inventaire pour le provisionnement
- Surveillance & Métriques - Métriques de performance et de surveillance du provisionnement

# Provisionnement de carte SIM

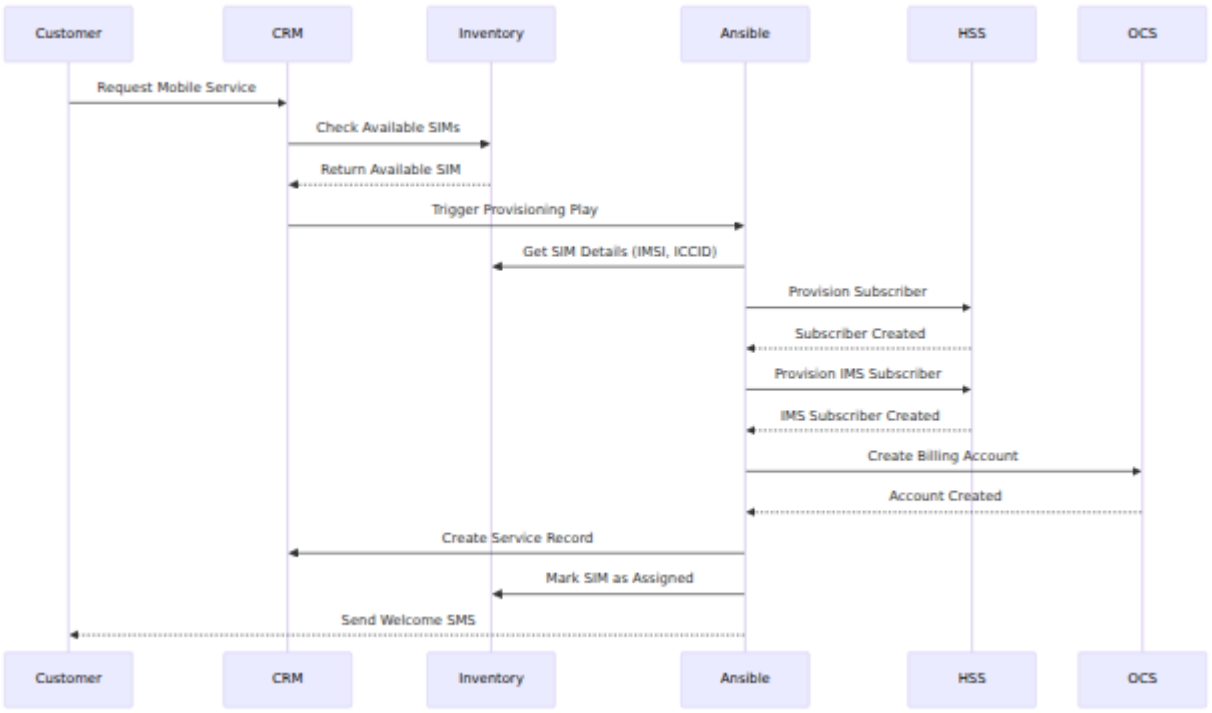
OmniCRM fournit un support complet pour le provisionnement à la fois des **cartes SIM physiques** et des **eSIM** (cartes SIM intégrées) pour les opérateurs de réseau mobile et les MVNO. Le système gère l'ensemble du cycle de vie, de la gestion des stocks à l'activation, l'attribution et le déprovisionnement.

Voir aussi : Système de Provisionnement pour les concepts généraux de provisionnement, Inventaire pour la gestion des stocks, Ansible Playbooks pour l'automatisation du provisionnement.

## Aperçu

Le provisionnement de SIM dans OmniCRM implique plusieurs systèmes intégrés travaillant ensemble :

1. **Gestion des Stocks** - Suit les cartes SIM disponibles (cartes physiques et profils eSIM)
2. **Intégration HSS/IMS** - Provisionne les identifiants d'abonné et les services vocaux
3. **Intégration OCS** - Configure la facturation et le chargement pour le service
4. **Automatisation Ansible** - Orchestre le flux de travail de provisionnement
5. **Support d'Auto-Inscription** - Permet aux clients d'activer leurs propres SIM




# Provisionnement de Carte SIM Physique

Les cartes SIM physiques sont des cartes SIM amovibles traditionnelles que les clients insèrent dans leurs appareils. OmniCRM gère celles-ci via le système d'inventaire et les provisionne au HSS/IMS pour l'accès au réseau.

## Flux de Travail de la Carte SIM Physique

### 1. Configuration de l'Inventaire

Les cartes SIM doivent d'abord être chargées dans le système d'inventaire :

- **ICCID** (Identifiant de Carte à Circuit Intégré) - Identifiant unique de la carte SIM
- **IMSI** (Identité d'Abonné Mobile Internationale) - Identité de l'abonné pour le réseau

Ceux-ci sont stockés dans les champs d'éléments d'inventaire :

- `itemtext1`: ICCID

- `itemtext2`: IMSI
- `itemtext3`: Type de SIM (Physique/eSIM) - optionnel

**Exemple d'Élément d'Inventaire de Carte SIM Physique :**

```
{
  "inventory_id": 1001,
  "item": "SIM Card",
  "itemtext1": "89612345678901234567",  // ICCID
  "itemtext2": "310120123456789",       // IMSI
  "itemtext3": "Physical",               // Type de SIM
  "item_location": "Warehouse A, Shelf 3",
  "item_state": "New",
  "wholesale_cost": 2.50,
  "retail_cost": 10.00
}
```

Pour une explication complète de tous les champs d'inventaire (itemtext1-20, valeurs d'état d'élément, champs d'adresse, management_url, etc.), voir Aperçu de l'Inventaire - Champs d'Éléments d'Inventaire.

**Stockage des Identifiants d'Authentification**

Les identifiants d'authentification pour les cartes SIM sont stockés dans le **HSS AuC (Centre d'Authentification)**, pas dans l'inventaire CRM :

- **Ki** (Clé d'Authentification) - Clé secrète pour l'authentification (stockée dans HSS)
- **OPC** (Code Opérateur) - Paramètre d'authentification spécifique à l'opérateur (stocké dans HSS)
- **PIN1/PIN2** - Codes PIN utilisateur (stockés dans HSS)
- **PUK1/PUK2** - Codes de déverrouillage PIN (stockés dans HSS)

L'inventaire CRM suit quelle SIM (par ICCID/IMSI) est attribuée à quel client, tandis que le HSS gère l'authentification réseau réelle en utilisant les identifiants Ki/OPC qui correspondent à la carte SIM physique.

**2. Attribution de Service**

Lorsqu'un client commande un service mobile :

1. Le personnel ou le client sélectionne une carte SIM disponible dans l'inventaire
2. Un numéro de mobile (MSISDN) est sélectionné dans l'inventaire des numéros de téléphone
3. Le playbook de provisionnement du produit est déclenché (par exemple, `play_psim_only.yaml`)

**3. Provisionnement HSS**

Le playbook Ansible provisionne l'abonné au Serveur Abonné Domicile (HSS) :

Tout d'abord, il récupère l'**auc_id** (référence aux identifiants d'authentification déjà stockés dans HSS) :

```
- name: Get AuC ID for IMSI
  uri:
    url: "{{ item }}/auc/imsi/{{ imsi }}"
    method: GET
  loop: "{{ hss_peers }}"
  register: auc_lookup

- name: Extract auc_id
  set_fact:
    auc_id: "{{ auc_lookup.results[0].json.auc_id }}"
```

Puis crée l'enregistrement d'abonné en faisant référence à cet auc_id :

```yaml
- name: Provision subscriber on HSS
  uri:
    url: "{{ item }}/subscriber/"
    method: PUT
    body_format: json
    body:
      enabled: true
      roaming_enabled: true
      auc_id: "{{ auc_id }}"
      msisdn: "{{ phone_number }}"
      imsi: "{{ imsi }}"
      ue_ambr_dl: 9999999
      ue_ambr_ul: 9999999
      apn_list: "1,2,3"
      default_apn: 1
  loop: "{{ hss_peers }}"
```

Les identifiants d'authentification (Ki, OPC) restent dans le HSS AuC et ne sont jamais exposés au CRM ou aux playbooks de provisionnement.

**4. Provisionnement IMS (pour les Services Vocaux)**

Pour la capacité d'appel vocal, un abonné IMS est créé :

```yaml
- name: Create IMS subscriber for voice services
  uri:
    url: "{{ item }}/ims_subscriber/"
    method: PUT
    body_format: json
    body:
      imsi: "{{ imsi }}"
      msisdn: "{{ phone_number }}"
      msisdn_list: "{{ phone_number }}"
      ifc_path: "default_ifc.xml"
      sh_profile: "{{ sh_profile_xml }}"
  loop: "{{ hss_peers }}"
```

**5. Configuration du Système de Facturation**

Le OCS (Système de Facturation en Ligne) est configuré avec :

- Création de compte
- Mapping IMSI/MSISDN via des profils d'attributs
- Règles de filtrage pour identifier l'abonné
- Limites de ressources (sessions concurrentes)
- Solde initial

**6. Création de Service et Attribution d'Inventaire**

Enfin :

- Un enregistrement de service est créé dans le CRM
- L'élément d'inventaire de la carte SIM est attribué au client (`customer_id` défini)
- La carte SIM est liée au service (`service_id` défini)
- L'état de l'inventaire est mis à jour à `"Assigned"`

## Authentification de la Carte SIM Physique

Les SIM physiques utilisent les identifiants AuC (Centre d'Authentification) stockés dans le HSS :

- **Ki** et **OPC** sont utilisés pour l'authentification 4G/5G (algorithme MILENAGE)
- Ces identifiants sont préchargés dans la base de données AuC du HSS lors de l'importation de la SIM
- La carte SIM physique contient des valeurs Ki/OPC correspondantes
- Lors du provisionnement, le playbook fait référence à l'**auc_id** pour lier l'abonné à ces identifiants
- L'authentification réseau se fait via un défi-réponse entre la SIM et le HSS
- Le CRM ne voit jamais ni ne gère les valeurs Ki/OPC réelles

# Provisionnement eSIM

Les eSIM (cartes SIM intégrées) sont des profils SIM basés sur des logiciels qui peuvent être téléchargés sur des appareils compatibles sans cartes SIM

physiques. OmniCRM prend en charge le provisionnement eSIM avec des codes d'activation LPA (Local Profile Assistant).

# Fonctionnalités de l'eSIM

### Codes d'Activation LPA

Les eSIM utilisent des codes LPA pour l'activation :

```
LPA:1$smdp.example.com$ACTIVATION-CODE-ABC123XYZ
```

Où :

- `LPA:1` - Identifiant de version LPA
- `smdp.example.com` - Adresse du serveur SM-DP+ (Gestionnaire de Données de Préparation d'Abonnement)
- `ACTIVATION-CODE-ABC123XYZ` - Code d'activation unique pour ce profil eSIM

### Génération de QR Code

OmniCRM génère automatiquement des QR codes à partir des codes d'activation LPA :

- Stockés dans le champ `management_url` de l'inventaire
- Affichés dans l'interface utilisateur sous forme de QR code scannable
- Les clients scannent avec la caméra de leur appareil pour installer le profil eSIM
- Aucune saisie manuelle de longs codes d'activation requise

### Exemple d'Élément d'Inventaire eSIM :

```
{
  "inventory_id": 1002,
  "item": "eSIM",
  "itemtext1": "89612345678901234567",
  "itemtext2": "310120123456790",
  "itemtext3": "eSIM",
  "management_url": "LPA:1$smdp.example.com$ACTIVATION-CODE-
ABC123XYZ",
  "item_location": "Virtual Inventory",
  "item_state": "New",
  "wholesale_cost": 0.00,
  "retail_cost": 0.00
}
```

Lors de la visualisation de cet élément d'inventaire dans l'interface utilisateur, un QR code est automatiquement généré et affiché pour un scan facile.

## Flux de Travail de Provisionnement eSIM

Le flux de travail de provisionnement eSIM est presque identique à celui du provisionnement de carte SIM physique, avec quelques différences clés :

### 1. Attribution Automatique d'eSIM

Si un produit nécessite une SIM mais qu'aucune SIM physique n'est sélectionnée, le système peut automatiquement attribuer une eSIM disponible :

```
# Recherche automatique d'eSIM
search_filters = {
    "customer_id": [None],
    "item_state": ["New"],
    "itemtext3": ["eSIM"]  # Filtrer par type eSIM
}
available_esim = search_inventory("SIM Card",
filters=search_filters)
```

### 2. Provisionnement HSS/IMS

Les eSIM sont provisionnées au HSS/IMS exactement comme les SIM physiques :

- Même provisionnement IMSI
- Même création d'abonné IMS
- Même identifiants d'authentification (Ki, OPC)

**3. Activation Client**

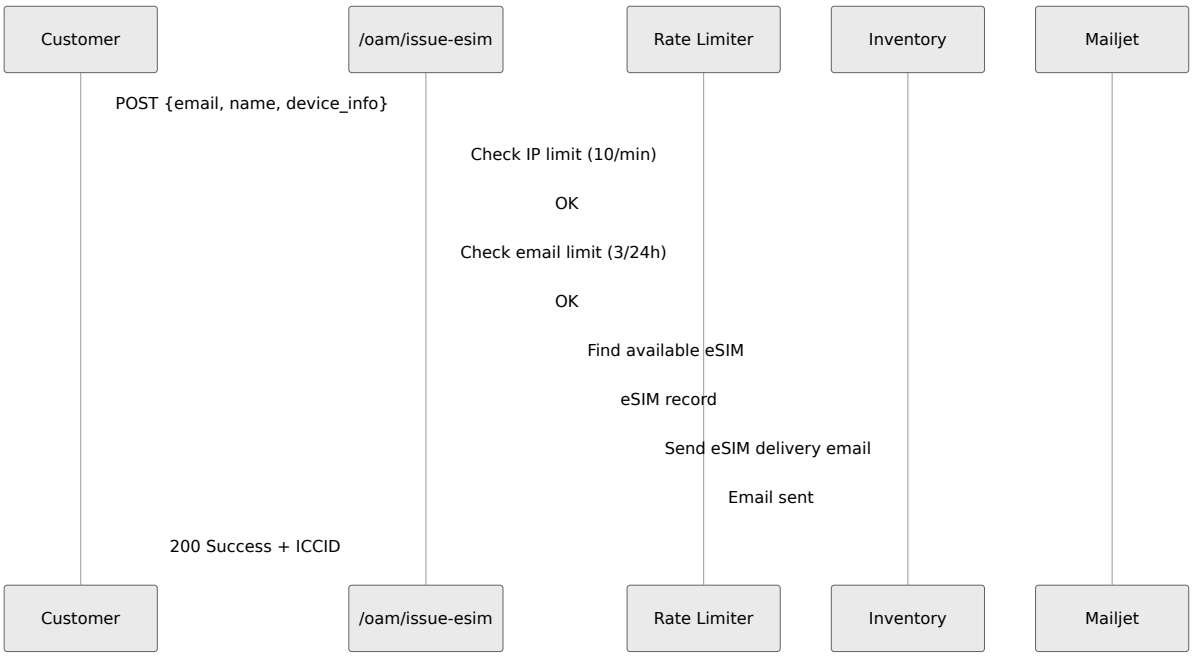
Après le provisionnement :

1. Le client reçoit les détails d'activation de l'eSIM par e-mail ou dans le portail client
2. Un QR code est affiché pour le scan
3. Le client scanne le QR code avec son appareil
4. L'appareil télécharge le profil eSIM depuis le serveur SM-DP+
5. L'eSIM est installée et prête à l'emploi

# API d'Émetteur eSIM Publique

OmniCRM fournit un point de terminaison API public pour les demandes d'eSIM en libre-service, permettant aux clients de demander une eSIM directement depuis un site Web ou une application mobile sans intervention du personnel.

**Point de terminaison :** `POST /crm/oam/issue-esim`




## Paramètres de Demande

| Paramètre | Type | Requis | Par Défaut | D |
|---|---|---|---|---|
| `email` | Chaîne | Oui | - | Adr clie livra l'eS au f vali |
| `name` | Chaîne | Non | Dérivé de l'e-mail | Nor pou pers de l n'es la p l'ad est |
| `customer_id` | Entier | Non | `null` | ID d exis est un enr d'ac clie suiv |
| `device_info` | Objet | Non | `{}` | Info l'ap l'an suiv com |
| `device_info.compatibility_status` | Chaîne | Non | `"unknown"` | Éta com l'eS |

| Paramètre | Type | Requis | Par Défaut | D |
|---|---|---|---|---|
| | | | | `sup` `sup` `unk` pou |
| `device_info.user_agent` | Chaîne | Non | En-tête de demande | Cha utili Cap aut à pa tête den fou |

**Limitation de Taux**

Le point de terminaison met en œuvre une double limitation de taux pour prévenir les abus :

| Limite | Fenêtre | Portée | Description |
|---|---|---|---|
| 10 demandes | 1 minute | Par adresse IP | Limite de taux générale appliquée via un décorateur |
| 3 demandes | 24 heures | Par adresse e-mail | Empêche les demandes excessives d'eSIM à la même adresse e-mail |

Lorsque les limites de taux sont dépassées, le point de terminaison renvoie HTTP 429 avec `error_type: "RateLimitExceeded"`.

**Format de Réponse**

**Réponse de Succès (HTTP 200) :**

```
{
  "result": "Success",
  "message": "eSIM request received. You will receive your eSIM QR code via email shortly.",
  "esim_id": 8472,
  "iccid": "89442000000001234567",
  "email_sent": true
}
```


| Champ | Type | Description |
|---|---|---|
| `result` | Chaîne | Toujours `"Success"` pour les réponses HTTP 200 |
| `message` | Chaîne | Message de confirmation lisible par l'homme |
| `esim_id` | Entier | L'ID d'inventaire de l'eSIM attribuée |
| `iccid` | Chaîne | L'ICCID de l'eSIM attribuée |
| `email_sent` | Booléen | Indique si l'e-mail de livraison a été envoyé avec succès |

**Réponses d'Erreur**

| Code HTTP | Type d'Erreur | Description |
|---|---|---|
| 400 | - | Adresse e-mail invalide ou manquante |
| 429 | `RateLimitExceeded` | Limite de taux par e-mail ou par IP dépassée |
| 503 | `NoInventory` | Aucune eSIM disponible dans l'inventaire |
| 500 | - | Erreur interne du serveur |

**Format de Réponse d'Erreur :**

```
{
  "result": "Failed",
  "reason": "Error description",
  "error_type": "ErrorType"
}
```

**Livraison par E-mail**

Le point de terminaison envoie un e-mail de livraison d'eSIM via Mailjet contenant :

- ICCID
- QR code (si disponible dans `itemtext2`)
- Code d'activation/chaîne LPA (si disponible dans `itemtext3` ou `management_url`)
- Instructions d'activation étape par étape

Configurez le modèle Mailjet dans `crm_config.yaml` :

```
mailjet:
  api_crmCommunicationEsimDelivery:
    from_email: "support@aimobile.ac"
    from_name: "Example Mobile"
    template_id: 1234567
    subject: "Your eSIM is Ready"
```

| Paramètre | Type | Requis | Description |
|---|---|---|---|
| `from_email` | Chaîne | Oui | Adresse e-mail de l'expéditeur |
| `from_name` | Chaîne | Oui | Nom d'affichage de l'expéditeur |
| `template_id` | Entier | Non | ID de modèle Mailjet. S'il n'est pas fourni, un e-mail HTML par défaut est généré. |
| `subject` | Chaîne | Oui | Ligne de sujet de l'e-mail |

S'il n'y a pas de modèle configuré, le système génère un e-mail HTML de marque avec :

- Marque et logo de l'entreprise
- QR code pour l'activation de l'eSIM (si disponible)
- Instructions manuelles pour le code d'activation
- Instructions de configuration spécifiques à l'appareil pour iOS et Android

**Journalisation des Activités**

Chaque demande d'eSIM crée un enregistrement d'activité avec :

- Type d'activité : `esim_request`
- Lien avec l'ID client (si fourni)
- État de compatibilité de l'appareil
- Informations sur l'agent utilisateur

Cela permet de suivre les demandes d'eSIM et d'analyser les conversions.

**4. Structure des Données eSIM dans HSS**

Les eSIM dans le HSS AuC (Centre d'Authentification) sont marquées avec un drapeau `esim: true` :

```
{
  "esim": true,
  "lpa": "LPA:1$smdp.example.com$ACTIVATION-CODE-ABC123XYZ",
  "iccid": "89612345678901234 57",
  "imsi": "310120123456790",
  "ki": "00112233445566778899AABBCCDDEEFF",
  "opc": "FFEEDDCCBBAA99887766554433221100",
  "pin1": "1234",
  "pin2": "5678",
  "puk1": "12345678",
  "puk2": "87654321",
  "batch_name": "eSIM_Batch_2024_Q1",
  "sim_vendor": "Thales"
}
```


## Processus d'Importation d'eSIM

Les eSIM sont généralement importées en masse depuis le fournisseur d'eSIM :

**Script d'Importation :** `/OmniCRM-API/Provisioners/customer_product_creation/import_keys_esim.py`

Le processus d'importation :

1. Lit les données eSIM à partir de fichiers Excel fournis par le fournisseur
2. Charge les codes d'activation LPA à partir de fichiers CSV
3. Met à jour le HSS avec les identifiants d'eSIM
4. Crée des éléments d'inventaire correspondants dans le CRM
5. Lie les profils eSIM au système d'inventaire

Cela garantit que les profils eSIM sont disponibles pour attribution tout en maintenant un suivi approprié des stocks.

# Intégration de l'Inventaire

Le système d'inventaire est central au provisionnement de SIM, suivant à la fois les ressources physiques et virtuelles nécessaires pour les services mobiles.

## Modèle d'Inventaire de Carte SIM

Un modèle d'inventaire typique de carte SIM définit :

```
{
  "item": "SIM Card",
  "itemtext1_label": "ICCID",
  "itemtext2_label": "IMSI",
  "itemtext3_label": "SIM Type",
  "wholesale_cost": 2.50,
  "retail_cost": 10.00
}
```

Remarque : Les identifiants d'authentification (Ki, OPC, PIN, PUK) sont stockés dans le HSS AuC, pas dans l'inventaire CRM.

Pour des informations détaillées sur la création et la gestion des modèles d'inventaire, y compris comment définir des champs personnalisés (itemtext1-20) et lier des modèles à des produits, voir Gestion de l'Inventaire - Modèles d'Inventaire section.

## Inventaire de Numéros Mobiles

Les numéros de téléphone sont gérés comme des éléments d'inventaire séparés :

```
{
  "inventory_id": 4001,
  "item": "Mobile Number",
  "itemtext1": "+61412345678",
  "itemtext2": "Melbourne",
  "itemtext3": "Mobile",
  "item_location": "Australia - VIC",
  "item_state": "New",
  "wholesale_cost": 1.00,
  "retail_cost": 0.00
}
```

## États d'Inventaire pour les SIM

Les cartes SIM progressent à travers plusieurs états d'inventaire :

- **New** - SIM inutilisée, disponible pour attribution
- **Assigned** - Actuellement active avec un client
- **Used** - Précédemment attribuée, retournée à l'inventaire (peut être réutilisée)
- **Internal Use** - Pour tests ou usage du personnel
- **Damaged** - Non fonctionnelle, nécessite un remplacement
- **Lost** - Ne peut pas être localisée
- **Stolen** - Signalée comme volée

## Intégration de Produits

Les produits spécifient les stocks requis via `inventory_items_list` :

```
{
  "product_id": 1,
  "product_slug": "Mobile-SIM",
  "product_name": "Mobile SIM Only",
  "provisioning_play": "play_psim_only",
  "inventory_items_list": "['SIM Card', 'Mobile Number']"
}
```

Lors du provisionnement de ce produit :

1. L'interface utilisateur affiche un sélecteur d'inventaire avec deux listes déroulantes
2. L'utilisateur sélectionne une carte SIM disponible (physique ou eSIM)
3. L'utilisateur sélectionne un numéro mobile disponible
4. Les ID d'inventaire sont passés au playbook Ansible
5. Le playbook récupère tous les détails d'inventaire via l'API
6. Le provisionnement se poursuit avec les ressources sélectionnées

Pour une explication complète sur la façon dont les produits pilotent le provisionnement, comment les exigences d'inventaire sont définies, et

comment les variables sont passées aux playbooks, voir Système de Provisionnement - Comment les Produits Pilotent le Provisionnement.

## Flux d'Attribution d'Inventaire

Le playbook Ansible gère l'attribution d'inventaire.

Pour plus de détails sur la façon dont les playbooks s'authentifient avec l'API CRM (tokens Bearer, clés API, tokens de rafraîchissement), voir Système de Provisionnement - Authentification et Autorisation.

```yaml
- name: Get SIM Card details from inventory
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ inventory_id_sim_card }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
  register: sim_inventory_response

- name: Extract SIM identifiers
  set_fact:
    iccid: "{{ sim_inventory_response.json.itemtext1 }}"
    imsi: "{{ sim_inventory_response.json.itemtext2 }}"

- name: Get AuC ID from HSS (reference to authentication
credentials)
  uri:
    url: "{{ item }}/auc/imsi/{{ imsi }}"
    method: GET
  loop: "{{ hss_peers }}"
  register: auc_response

- name: Extract AuC ID
  set_fact:
    auc_id: "{{ auc_response.results[0].json.auc_id }}"

# ... provision to HSS/IMS/OCS ...

- name: Assign SIM to customer
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ inventory_id_sim_card }}"
    method: PATCH
    headers:
      Authorization: "Bearer {{ access_token }}"
    body_format: json
    body:
      customer_id: "{{ customer_id }}"
      service_id: "{{ service_id }}"
      item_state: "Assigned"
```

Cela garantit :

- La SIM est marquée comme attribuée au client
- L'ID de service est lié pour le suivi
- L'état de l'inventaire reflète le statut actuel
- D'autres systèmes peuvent interroger l'inventaire pour trouver la SIM du client

# Auto-Inscription avec Provisionnement de SIM

OmniCRM prend en charge l'auto-inscription en libre-service des clients où les clients peuvent activer leurs propres cartes SIM ou eSIM sans intervention du personnel.

## Flux d'Auto-Inscription

### 1. Le Client Initie l'Inscription

Le client visite la page d'auto-inscription et commence la création de compte :

- Fournit des informations personnelles
- Sélectionne un plan de service (produit)
- Entre les détails de paiement

### 2. Validation du Numéro SIM

Le client peut fournir son propre numéro SIM (optionnel) :

```
GET /auth/self-signup/validate-itemtext1/{sim_number}
```

Ce point de terminaison vérifie si :

- La SIM existe dans l'inventaire
- La SIM est disponible (pas déjà attribuée)
- La SIM est dans l'état correct ("New" ou "Used")

### 3. Attribution Automatique de SIM (Si Aucune SIM Fournie)

Si le client n'a pas de SIM, le système attribue automatiquement une :

```
def _populate_base_plan_inventory(sim_number, plan):
    if sim_number:
        # SIM fournie par le client - vérifiez qu'elle existe
        sim_inventory = get_inventory_by_itemtext1(session,
sim_number, None)
        if not sim_inventory:
            # Vérifiez s'il existe un service inactif avec ce
numéro
            service = get_inactive_service_by_sim(sim_number)
            if service:
                sim_inventory =
get_sim_for_service(service.service_id)
    else:
        # Attribution automatique d'une SIM disponible de
l'inventaire
        sim_inventory =
get_first_available_inventory_by_item_name(
            session,
            "SIM Card",
            filters={"item_state": ["New"], "customer_id": [None]}
        )

    # Obtenez ICCID et MSISDN
    iccid = sim_inventory.itemtext1

    # Attribution automatique du numéro de téléphone
    number_inventory = get_first_available_inventory_by_item_name(
        session,
        "Mobile Number"
    )
    msisdn = number_inventory.itemtext1

    return {
        "SIM Card": sim_inventory.inventory_id,
        "Mobile Number": number_inventory.inventory_id
    }
```

**4. Création de Compte et Provisionnement**

Une fois le paiement autorisé :

```
def _post_signup_tasks(customer_id, plan, payment_method):
    # 1. Enregistrer le mode de paiement
    save_stripe_payment_method(customer_id, payment_method)

    # 2. Facturer la carte du client
    charge_customer_card(customer_id, plan.retail_setup_cost)

    # 3. Connecter l'utilisateur (obtenir des tokens JWT)
    tokens = login_customer(customer_id)

    # 4. Remplir l'inventaire (SIM et numéro de téléphone)
    inventory_vars = _populate_base_plan_inventory(sim_number,
plan)

    # 5. Créer un travail de provisionnement
    provision_vars = {
        "product_id": plan.product_id,
        "customer_id": customer_id,
        **inventory_vars  # Inclure les IDs de la carte SIM et du
numéro mobile
    }

    # Optionnel : Chaîner des produits supplémentaires
    if plan.addon_chain:
        provision_vars["addon_chain"] = plan.addon_chain

    provision_id = create_provisioning_job(provision_vars)

    # 6. Retourner le succès avec les tokens
    return {
        "access_token": tokens.access_token,
        "refresh_token": tokens.refresh_token,
        "customer_id": customer_id,
        "provision_id": provision_id
    }
```

**5. Le Client Reçoit les Détails d'Activation**

Après que le provisionnement soit terminé :

- **SIM Physique** : E-mail de bienvenue avec numéro de téléphone et instructions d'activation

- **eSIM** : L'e-mail inclut un QR code pour le téléchargement du profil eSIM, ou le client peut voir le QR code dans le portail client

**6. Permissions Spéciales pour l'Auto-Inscription**

Lors de l'inscription, le token JWT inclut une revendication spéciale :

```
{
  "sub": "customer_123",
  "signup_process": true,
  "permissions": ["VIEW_OWN_PROVISION", "UPDATE_OWN_INVENTORY"]
}
```

Cela permet :

- Au client de lire/mettre à jour l'inventaire pendant l'inscription
- Restreint à son propre compte client uniquement
- Empêche l'attribution de SIM appartenant à d'autres clients

## Sécurité de l'Auto-Inscription

**Vérifications de Validation :**

- La SIM doit exister dans l'inventaire
- La SIM ne doit pas être attribuée à un autre client
- La SIM doit être dans un état acceptable ("New", "Used")
- Le client ne peut attribuer des SIM qu'à son propre compte
- Le paiement doit être autorisé avant le provisionnement

**Mesures de Sécurité de l'Inventaire :**

- Les transactions de base de données garantissent une attribution atomique
- Les tentatives d'inscription simultanées sont empêchées via des verrous de base de données
- Les changements d'état d'inventaire sont enregistrés pour audit

# Provisionnement Basé sur Ansible

Tout le provisionnement de SIM est orchestré via des playbooks Ansible, fournissant un processus cohérent, auditable et répétable.

Pour une documentation complète sur la structure des playbooks Ansible, le modèle bloc/ressort, comment les playbooks interagissent avec les produits, et les meilleures pratiques, voir Playbooks Ansible : Guide Détaille.

## Playbook Principal de Provisionnement de SIM

**Fichier :** `/OmniCRM-API/Provisioners/plays/play_psim_only.yaml`

C'est le playbook principal pour le provisionnement de SIM mobiles (à la fois physiques et eSIM). Il gère :

- Services vocaux et données mobiles
- Services uniquement de données
- Abonnés IMS uniquement vocaux

**Structure du Playbook (1 561 lignes) :**

Pour une explication détaillée des en-têtes de playbook, des directives hosts/gather_facts/become, et de la structure globale, voir Playbooks Ansible - Structure et Anatomie du Playbook.

```yaml
- name: OmniCore Service Provisioning 2024
  hosts: localhost
  gather_facts: no
  become: False

  tasks:
    - name: Main provisioning block
      block:
        # --- PRÉ-PROVISIONNEMENT (Lignes 240-396) ---
        - name: Get SIM information from inventory
          # Retrieves IMSI, ICCID, etc.

        - name: Validate IMSI length
          # Must be exactly 15 digits

        - name: Check if IMSI exists in HSS
          # Prevents duplicate provisioning

        - name: Get AuC ID for IMSI
          # Retrieves auc_id reference from HSS

        # --- PROVISIONNEMENT HSS (Lignes 398-534) ---
        - name: Create/Update HSS Subscriber
          # Provisions to all HSS peers

        - name: Create IMS Subscriber
          # Enables voice calling

        # --- CONFIGURATION OCS (Lignes 536-837) ---
        - name: Create ENUM entry
          # E.164 number mapping

        - name: Create FilterS rule
          # Account identification

        - name: Create AttributeS profile
          # IMSI/MSISDN mapping

        - name: Create ResourceS profile
          # Concurrent session limits

        - name: Create OCS account
          # Billing account creation
```

```
    # --- SERVICE & INVENTAIRE (Lignes 865-946) ---
    - name: Create service record
      # In CRM database

    - name: Assign SIM to service
      # Update inventory

    - name: Send welcome SMS
      # Customer notification

  rescue:
    # --- NETTOYAGE/DEPROVISIONNEMENT (Lignes 975-1562) ---
    - name: Return inventory to pool
      # Reset customer_id, service_id to null

    - name: Delete OCS account
      # Remove billing account

    - name: Remove HSS subscribers
      # Or set to dormant state

    - name: Determine success/failure
      assert:
        that:
          - action == "deprovision"
```

# Étapes Clés de Provisionnement

### 1. Validation Préalable au Provisionnement

```yaml
- name: Get SIM information from CRM inventory
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ inventory_id_sim_card }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
  register: api_response_sim

- name: Extract and validate IMSI
  set_fact:
    imsi: "{{ api_response_sim.json.itemtext2 }}"

- name: Validate IMSI format
  assert:
    that:
      - imsi | length == 15
      - imsi is match('^[0-9]+$')
    fail_msg: "IMSI must be exactly 15 digits"
```

**2. Récupérer l'ID AuC depuis HSS**

```yaml
- name: Get AuC ID for IMSI from HSS
  uri:
    url: "{{ item }}/auc/imsi/{{ imsi }}"
    method: GET
  loop: "{{ crm_config.hss.hss_peers }}"
  register: auc_lookup

- name: Extract auc_id from first HSS response
  set_fact:
    auc_id: "{{ auc_lookup.results[0].json.auc_id }}"
```

**3. Provisionnement de l'Abonné HSS**

```yaml
- name: Create subscriber in each HSS peer
  uri:
    url: "{{ item }}/subscriber/"
    method: PUT
    body_format: json
    body:
      enabled: true
      roaming_enabled: true
      auc_id: "{{ auc_id }}"
      msisdn: "{{ phone_number }}"
      imsi: "{{ imsi }}"
      ue_ambr_dl: 9999999  # Download speed limit
      ue_ambr_ul: 9999999  # Upload speed limit
      apn_list: "1,2,3"
      default_apn: 1
    status_code: [200, 201]
  loop: "{{ crm_config.hss.hss_peers }}"
  register: hss_responses
```

**4. Provisionnement de l'Abonné IMS (Vocal)**

```
- name: Create IMS subscriber for voice services
  uri:
    url: "{{ item }}/ims_subscriber/"
    method: PUT
    body_format: json
    body:
      imsi: "{{ imsi }}"
      msisdn: "{{ phone_number }}"
      msisdn_list: "{{ phone_number }}"
      ifc_path: "default_ifc.xml"
      sh_profile: |
        <?xml version="1.0"?>
        <IMSSubscription>
          <PrivateID>{{ imsi }}@ims.mnc{{ mnc }}.mcc{{ mcc
}}.3gppnetwork.org</PrivateID>
          <ServiceProfile>
            <PublicIdentity>
              <Identity>sip:{{ phone_number }}@ims.mnc{{ mnc
}}.mcc{{ mcc }}.3gppnetwork.org</Identity>
            </PublicIdentity>
          </ServiceProfile>
        </IMSSubscription>
    status_code: [200, 201]
  loop: "{{ crm_config.hss.hss_peers }}"
```

### 5. Configuration de la Facturation OCS

```yaml
# Créer des filtres de compte (identifier l'abonné par
IMSI/MSISDN)
- name: Create FilterS for IMSI identification
  uri:
    url: "{{ crm_config.ocs.ocsApi }}/jsonrpc"
    method: POST
    body_format: json
    body:
      method: "ApierV1.SetFilter"
      params:
        - ID: "FILTER_IMSI_{{ service_uuid }}"
          Type: "*string"
          Element: "~*req.OriginHost"
          Values: ["{{ imsi }}"]

# Créer un profil d'attribut (mapper l'IMSI au compte)
- name: Create AttributeS profile
  uri:
    url: "{{ crm_config.ocs.ocsApi }}/jsonrpc"
    method: POST
    body_format: json
    body:
      method: "ApierV1.SetAttributeProfile"
      params:
        - ID: "ATTR_ACCOUNT_{{ service_uuid }}"
          FilterIDs: ["FILTER_IMSI_{{ service_uuid }}"]
          Attributes:
            - Path: "*req.Account"
              Value: "{{ service_uuid }}"

# Créer un compte OCS
- name: Create OCS account
  uri:
    url: "{{ crm_config.ocs.ocsApi }}/jsonrpc"
    method: POST
    body_format: json
    body:
      method: "ApierV2.SetAccount"
      params:
        - Tenant: "{{ crm_config.ocs.ocsTenant }}"
          Account: "{{ service_uuid }}"
          ActionPlanIds: []
          ExtraOptions:
```

```
            AllowNegative: false
            Disabled: false

# Ajouter un solde initial
- name: Add initial monetary balance
  uri:
    url: "{{ crm_config.ocs.ocsApi }}/jsonrpc"
    method: POST
    body_format: json
    body:
      method: "ApierV1.AddBalance"
      params:
        - Tenant: "{{ crm_config.ocs.ocsTenant }}"
          Account: "{{ service_uuid }}"
          BalanceType: "*monetary"
          Balance:
            Value: 0
            ExpiryTime: "+8760h"  # 1 year
```

**6. Création de Service**

```
- name: Create service record in CRM
  uri:
    url: "{{ crm_config.crm.base_url }}/crm/service/"
    method: PUT
    headers:
      Authorization: "Bearer {{ access_token }}"
    body_format: json
    body:
      customer_id: "{{ customer_id }}"
      product_id: "{{ product_id }}"
      service_name: "Mobile Service - {{ phone_number }}"
      service_uuid: "{{ service_uuid }}"
      service_status: "Active"
      retail_cost: "{{ monthly_cost }}"
    status_code: 200
  register: service_creation_response

- name: Extract service_id
  set_fact:
    service_id: "{{ service_creation_response.json.service_id }}"
```

**7. Attribution de l'Inventaire**

```
- name: Assign SIM card to customer and service
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ inventory_id_sim_card }}"
    method: PATCH
    headers:
      Authorization: "Bearer {{ access_token }}"
    body_format: json
    body:
      customer_id: "{{ customer_id }}"
      service_id: "{{ service_id }}"
      item_state: "Assigned"
      sold_date: "{{ ansible_date_time.iso8601 }}"

- name: Assign phone number to customer and service
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ inventory_id_phone_number }}"
    method: PATCH
    headers:
      Authorization: "Bearer {{ access_token }}"
    body_format: json
    body:
      customer_id: "{{ customer_id }}"
      service_id: "{{ service_id }}"
      item_state: "Assigned"
      sold_date: "{{ ansible_date_time.iso8601 }}"
```

## Déprovisionnement et Rétrogradation

Le même playbook gère à la fois le retour en arrière des échecs de provisionnement et le déprovisionnement intentionnel en utilisant le bloc `rescue` d'Ansible.

Pour une explication détaillée du modèle bloc/ressort et pourquoi c'est une meilleure pratique, voir Système de Provisionnement - Rétrogradation et Nettoyage : Modèle de Meilleure Pratique.

```yaml
rescue:
  # Cette section s'exécute lorsque :
  # 1. Toute tâche dans le bloc échoue (retour automatique en
arrière)
  # 2. action == "deprovision" (nettoyage intentionnel)

  - name: Get inventory items linked to service
    uri:
      url: "{{ crm_config.crm.base_url
}}/crm/inventory/customer_id/{{ customer_id }}"
      method: GET
      headers:
        Authorization: "Bearer {{ access_token }}"
    register: inventory_items
    ignore_errors: true

  - name: Return inventory to available pool
    uri:
      url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ item.inventory_id }}"
      method: PATCH
      headers:
        Authorization: "Bearer {{ access_token }}"
      body_format: json
      body:
        customer_id: null
        service_id: null
        item_state: "Used"
    loop: "{{ inventory_items.json.data }}"
    when: inventory_items.json.data is defined
    ignore_errors: true

  - name: Delete OCS account
    uri:
      url: "{{ crm_config.ocs.ocsApi }}/jsonrpc"
      method: POST
      body_format: json
      body:
        method: "ApierV1.RemoveAccount"
        params:
          - Tenant: "{{ crm_config.ocs.ocsTenant }}"
            Account: "{{ service_uuid }}"
    ignore_errors: true
```

```
  - name: Delete or disable HSS subscriber
    uri:
      url: "{{ item.key }}/subscriber/{{ item.value.subscriber_id
}}"
      method: DELETE
    loop: "{{ hss_subscriber_data | dict2items }}"
    when: deprovision_subscriber | default(false) | bool
    ignore_errors: true

  # L'assertion finale détermine le succès ou l'échec
  - name: Determine if this was intentional deprovision or failed
provision
    assert:
      that:
        - action == "deprovision"
      fail_msg: "Provisioning failed and rollback completed"
      success_msg: "Deprovisioning completed successfully"
```

**Points Clés :**

- `ignore_errors: true` sur toutes les tâches de nettoyage garantit que toutes les tentatives de nettoyage sont effectuées
- Si `action == "deprovision"`, l'assertion réussit (statut 0 = succès)
- Si `action` n'est pas défini, l'assertion échoue (statut 2 = échec du provisionnement)
- Cela garantit des opérations atomiques : soit entièrement provisionné, soit entièrement nettoyé

## Autres Playbooks Liés aux SIM

**Playbook de Développement Local :** `play_local_mobile_sim.yaml` - Version de développement qui ignore les dépendances externes HSS/OCS

**Échange de MSISDN :** `play_swap_msisdn.yaml` - Échange de numéros de téléphone entre services existants

**Playbooks de Recharge :**

- `play_topup_monetary.yaml` - Ajouter un solde monétaire

- `play_topup_no_charge.yaml` - Ajouter des données/minutes gratuites
- `play_topup_dongle.yaml` - Recharge pour appareils IoT/données

# Intégration HSS et IMS

OmniCRM s'intègre à un Serveur Abonné Domicile (HSS) externe pour la gestion et l'authentification des abonnés.

## Fonctions HSS

**Fichier :** `/OmniCRM-API/Provisioners/hss.py`

**Fonctions Clés :**

```
def ProvisionMobileSIM(json_data, hss_urls):
    """
    Provisionne une carte SIM mobile au HSS et à l'IMS

    Args:
        json_data: Dict avec IMSI, MSISDN, identifiants
d'authentification
        hss_urls: Liste des URL des pairs HSS

    Returns:
        Dict avec les IDs d'abonnés de chaque pair HSS
    """
    # Créer un abonné HSS (pour les données)
    # Créer un abonné IMS (pour la voix)
    # Retourner les IDs d'abonnés pour le suivi

def get_subscriber_info_msisdn(msisdn, hss_urls):
    """
    Récupère les détails de l'abonné par numéro de téléphone

    Args:
        msisdn: Numéro de téléphone à rechercher
        hss_urls: Liste des URL des pairs HSS

    Returns:
        Informations sur l'abonné provenant du HSS
    """
```

## Support Multi-HSS

OmniCRM prend en charge le provisionnement vers plusieurs pairs HSS pour la redondance :

```
# crm_config.yaml
hss:
  hss_peers:
    - "http://10.12.64.140:8080"
    - "http://10.12.64.141:8080"
  apn_list: "1,2,3"
  default_apn: 1
```

Le provisionnement se fait sur toutes les instances HSS configurées pour garantir :

- Redondance géographique
- Répartition de la charge
- Capacité de basculement
- Données d'abonnés synchronisées

## Centre d'Authentification (AuC)

Le AuC stocke les identifiants d'authentification des abonnés. Lors du provisionnement, seul l'**auc_id** est récupéré :

**Obtenir l'ID AuC pour IMSI :**

```
GET /auc/imsi/{imsi}
```

**Réponse :**

```
{
  "auc_id": 12345,
  "imsi": "310120123456789",
  "iccid": "89612345678901234 56",
  "esim": false,
  "lpa": null
}
```

Pour les eSIM, le drapeau `esim` est `true` et `lpa` contient le code d'activation à afficher aux clients.

**Remarque :** Les identifiants d'authentification réels (Ki, OPC, PIN, PUK) restent dans le HSS et ne sont jamais exposés via l'API. Le système de provisionnement a seulement besoin de l'`auc_id` pour faire référence à ces identifiants lors de la création d'abonnés.

# Intégration OCS

Le Système de Facturation en Ligne (OCS) gère la facturation et le tarification en temps réel pour les services mobiles. OmniCRM utilise CGRateS comme plateforme OCS.

## Configuration OCS

**Fichier :** `/OmniCRM-API/Provisioners/ocs.py`

**Configuration :**

```yaml
# crm_config.yaml
ocs:
  cgrates: "10.64.12.160:2080"
  ocsTenant: "mnc001.mcc001.3gppnetwork.org"
  ocsApi: "http://10.64.12.160:2080"
```

## Composants OCS pour les Services SIM

### 1. Filtres - Identification de l'Abonné

Les filtres identifient l'abonné à partir des demandes réseau :

```json
{
  "ID": "FILTER_IMSI_Service_abc123",
  "Type": "*string",
  "Element": "~*req.OriginHost",
  "Values": ["310120123456789"]
}
```

### 2. Attributs - Mapping de Compte

Les attributs mappent l'IMSI au compte de facturation :

```
{
  "ID": "ATTR_ACCOUNT_Service_abc123",
  "FilterIDs": ["FILTER_IMSI_Service_abc123"],
  "Attributes": [
    {
      "Path": "*req.Account",
      "Value": "Service_abc123"
    },
    {
      "Path": "*req.Bandwidth",
      "Value": "100000000"  // 100 Mbps
    }
  ]
}
```

**3. Ressources - Limites de Session**

Les ressources contrôlent les limites de sessions concurrentes :

```
{
  "ID": "RES_SESSIONS_Service_abc123",
  "FilterIDs": ["FILTER_IMSI_Service_abc123"],
  "Limit": 5,  // Max 5 sessions concurrentes
  "UsageTTL": "-1"
}
```

**4. Statistiques - Suivi de l'Utilisation**

Les files d'attente de statistiques suivent l'utilisation pour l'analyse :

```
{
  "ID": "STATS_Service_abc123",
  "FilterIDs": ["FILTER_IMSI_Service_abc123"],
  "Metrics": [
    "*sum#~*req.Usage",
    "*tcc"  // Total des appels
  ]
}
```

**5. Plans d'Action - Charges Récurrentes**

Les plans d'action gèrent les opérations récurrentes comme les frais mensuels :

```
{
  "Id": "ActionPlan_Service_abc123_Monthly",
  "ActionsId": "Action_Add_Monthly_Data",
  "Timing": {
    "MonthDays": [1],  // 1er de chaque mois
    "Time": "00:00:00Z"
  }
}
```

## Fonctions API OCS

**Fonctions Clés de ocs.py :**

```
async def async_get_balance(account, server, tenant):
    """Obtenir le solde actuel du compte"""

async def async_topup_dongle(tenant, account, server, days):
    """Ajouter une allocation de données basée sur le temps"""

def Get_Account_Status(account, server, tenant):
    """Obtenir des informations complètes sur le compte"""

def CGRateS_API_Call(method, params, server, tenant):
    """Appel API JSON-RPC générique CGRateS"""
```

# Réconciliation de l'Inventaire SIM

OmniCRM comprend des outils pour garantir la cohérence entre l'inventaire CRM et le HSS.

## Script de Réconciliation

**Fichier :** `/OmniCRM-API/Provisioners/hss_reconcile.py`

**Objectif :**

- Identifie les SIM dans le HSS mais pas dans l'inventaire (orphelines)
- Trouve les SIM attribuées non provisionnées dans le HSS
- Valide les formats MSISDN
- Génère des rapports HTML

**Vérifications Effectuées :**

1. **SIM Vendues dans le HSS :**
    - Interroge toutes les SIM attribuées de l'inventaire
    - Vérifie si chaque IMSI existe dans le HSS
    - Rapport sur les SIM attribuées aux clients mais pas dans le HSS
2. **SIM Provisionnées dans l'Inventaire :**
    - Interroge tous les abonnés du HSS
    - Vérifie si chaque IMSI existe dans l'inventaire
    - Rapport sur les entrées HSS orphelines
3. **Validation MSISDN :**
    - Valide les formats de numéros de téléphone
    - Vérifie la conformité des numéros australiens
    - Rapport sur les numéros mal formés

**Exécution de la Réconciliation :**

```
python hss_reconcile.py
```

**Sortie :**

- Rapport HTML avec résultats codés par couleur
- Listes de discordances pour enquête
- Recommandations pour nettoyage

**Meilleures Pratiques :**

- Exécuter la réconciliation chaque semaine
- Enquêter rapidement sur les discordances
- Utiliser les rapports pour identifier les échecs de provisionnement
- Nettoyer les entrées orphelines pour libérer des ressources

# Dépannage du Provisionnement de SIM

## Problèmes Courants et Solutions

**Problème : Le provisionnement de la SIM échoue avec "IMSI existe déjà dans le HSS"**

- **Cause :** L'IMSI est déjà provisionné à partir d'une tentative précédente
- **Solution :** Vérifiez le HSS pour un abonné existant, soit :
  - Supprimez l'abonné existant s'il est orphelin
  - Mettez à jour l'abonné existant au lieu de créer un nouveau
  - Exécutez la réconciliation pour identifier le conflit

**Problème : QR code eSIM non généré**

- **Cause :** Le champ `management_url` n'est pas peuplé avec le code LPA
- **Solution :** Assurez-vous que le script d'importation eSIM définit correctement `management_url`
- **Format :** Doit être `LPA:1$server$activation-code`

**Problème : L'élément d'inventaire apparaît comme "Disponible" mais est attribué**

- **Cause :** Le provisionnement a échoué avant l'attribution de l'inventaire
- **Solution :** Vérifiez les événements de provisionnement pour la raison de l'échec
- **Correction :** Complétez le provisionnement ou retournez l'inventaire dans la piscine

**Problème : Le client ne peut pas passer d'appels (les données fonctionnent)**

- **Cause :** Abonné IMS non provisionné
- **Solution :** Vérifiez si l'étape de provisionnement IMS a été complétée
- **Correction :** Provisionnez manuellement l'abonné IMS ou réexécutez le playbook

**Problème : Échecs d'authentification (la SIM ne peut pas se connecter)**

- **Cause :** Mismatch Ki/OPC entre la carte SIM et le HSS
- **Solution :** Vérifiez que les identifiants AuC correspondent à la carte SIM physique
- **Correction :** Mettez à jour le HSS AuC avec les identifiants corrects

**Problème : L'auto-inscription échoue à attribuer une SIM**

- **Cause :** Aucune SIM disponible dans l'inventaire
- **Solution :** Vérifiez l'inventaire pour les SIM avec l'état "New" et customer_id NULL
- **Correction :** Importez un nouveau lot de SIM ou marquez les SIM retournées comme "Used"

# Conseils de Débogage

**1. Vérifiez les Événements de Provisionnement :**

```
GET /crm/provision/provision_id/{id}
```

Recherchez les tâches échouées (statut 2) et examinez les messages d'erreur.

**2. Vérifiez l'Abonné HSS :**

```
GET {hss_url}/subscriber/msisdn/{phone_number}
```

Vérifiez si l'abonné existe et a la configuration correcte.

**3. Vérifiez le Compte OCS :**

```
{
  "method": "ApierV2.GetAccount",
  "params": [
    {
      "Tenant": "mnc001.mcc001.3gppnetwork.org",
      "Account": "Service_abc123"
    }
  ]
}
```

**4. Interrogez l'État de l'Inventaire :**

```
GET /crm/inventory/?filters={"itemtext2":["310120123456789"]}
```

Recherchez par IMSI pour trouver l'élément d'inventaire et vérifiez son état.

**5. Consultez les Journaux Ansible :**

Vérifiez `/var/log/ansible/` ou les détails des événements de provisionnement pour la sortie complète du playbook.

# Meilleures Pratiques

## Gestion de l'Inventaire SIM

1. **Importation en Masse :** Utilisez des scripts d'importation pour de grands lots de SIM
2. **Réconciliation Régulière :** Exécutez la réconciliation hebdomadaire HSS/inventaire
3. **Gestion des États :** Gardez les états d'inventaire précis (New, Assigned, Used)
4. **Suivi des Fournisseurs :** Enregistrez le fournisseur de SIM et les informations de lot
5. **Suivi des Coûts :** Maintenez des coûts de gros/détail précis

## Sécurité

1. **Séparation des Identifiants :** Les identifiants d'authentification (Ki/OPC) restent uniquement dans le HSS, jamais dans le CRM
2. **Contrôle d'Accès :** Limitez qui peut attribuer/modifier l'inventaire SIM
3. **Journalisation d'Audit :** Suivez toutes les attributions et modifications de SIM
4. **Gestion des PIN/PUK :** Les codes PIN/PUK sont stockés dans le HSS, fournis aux clients via des canaux sécurisés
5. **Processus de Perte/Vol :** Suspendre immédiatement le service dans le HSS et marquer l'état de l'inventaire

## Provisionnement

1. **Validation :** Validez toujours le format et l'unicité de l'IMSI
2. **Retour en Arrière :** Utilisez le bloc/ressort pour un nettoyage automatique en cas d'échec
3. **Notifications :** Envoyez des instructions d'activation claires aux clients
4. **Tests :** Testez le provisionnement en développement avant la production
5. **Surveillance :** Suivez les taux de succès de provisionnement et les raisons d'échec

## Spécifique à l'eSIM

1. **Format LPA :** Validez le format du code LPA avant de le stocker
2. **Tests de QR :** Testez les QR codes sur plusieurs appareils
3. **Instructions d'Activation :** Fournissez des guides étape par étape clairs
4. **Gestion des Profils :** Suivez les téléchargements et installations de profils eSIM
5. **Codes de Sauvegarde :** Fournissez des codes LPA manuels en cas d'échec de scan de QR

# Documentation Connexe

- Système de Provisionnement - Détails complets sur le flux de travail de provisionnement
- Ansible Playbooks - Structure des playbooks et meilleures pratiques
- Gestion de l'Inventaire - Aperçu du système d'inventaire
- Produits et Services - Configuration des produits
- Service Client - Soutenir les clients avec des problèmes de SIM
- Recharge et Top-Up - Ajouter un solde aux services mobiles

# Journal d'Activité Client

Le Journal d'Activité enregistre chaque changement apporté à un client, un contact, un site, un service et des paramètres financiers tels que les Transactions, les Factures et les Méthodes de Paiement.

Cela nous permet de suivre les changements effectués dans le système, qui les a effectués et quand. Le journal est utile pour auditer les changements et pour identifier les problèmes. Par exemple, si un client dit qu'il n'a jamais reçu une facture, nous pouvons vérifier le Journal d'Activité pour voir si le système l'a envoyée. Si un contact a été supprimé, nous pouvons voir qui l'a supprimé et quand.

Le Journal d'Activité est une liste chronologique de changements. Les changements les plus récents apparaissent en haut, et les changements plus anciens apparaissent plus bas dans la liste. Vous pouvez filtrer la liste à partir des onglets.

Vous ne pouvez pas supprimer les enregistrements du journal d'activité. Vous pouvez les filtrer, et vous pouvez consulter les détails pour voir ce qui a été changé et par qui.

# Ajout d'un Service

Les services sont les éléments pour lesquels nous facturons le client. Ils peuvent être des services Internet à domicile, des services mobiles, ou même des services abstraits comme la location d'un sous-réseau ou la fourniture d'électricité mesurée à un rack.

Un Service n'est qu'une instance d'un Produit pour un client donné. Le système le sélectionne dans le catalogue de produits et le provisionne pour le client.

Si vous n'avez pas encore créé un client, vous devez d'abord le faire, car les services sont liés aux clients. Vous devez également définir un mode de paiement pour le client, car les services génèrent des frais qui doivent être payés.

Les clients peuvent provisionner leurs propres services (si nous leur permettons). Le personnel du Service Client peut également provisionner des services pour le client.

La définition du produit établit des règles concernant qui peut acheter un produit. Par exemple, une règle peut permettre uniquement aux clients professionnels d'acheter un produit professionnel. Une règle peut également permettre uniquement aux clients ayant un service mobile d'acheter un addon mobile.

Les services peuvent également avoir des composants d'utilisation, comme l'utilisation de données, les minutes d'appel, ou d'autres frais basés sur l'utilisation. Un service peut avoir plusieurs frais, tels que des frais mensuels, des frais ponctuels ou des frais d'utilisation. Nous les consultons depuis le bouton "Utilisation".

De nombreux services prennent en charge des Addons. Par exemple, un service mobile a tous les recharges disponibles dans le menu des Addons. Un service Internet à domicile peut avoir une adresse IP statique ou des données supplémentaires disponibles en tant qu'addon. Le catalogue de produits définit tout cela.

# Flux de travail d'attribution de plans

La fonctionnalité d'attribution de plan permet au personnel de **provisionner des services pour les clients** en sélectionnant des produits dans le catalogue et en initiant le processus de provisionnement. C'est la méthode principale pour créer de nouveaux services lorsque les clients ne se provisionnent pas eux-mêmes.

## Aperçu

L'attribution d'un plan implique :

1. Sélectionner un client
2. Choisir un produit dans le catalogue
3. Configurer les exigences en matière d'inventaire (cartes SIM, équipements, etc.)
4. Définir les paramètres de service (renouvellement automatique, champs personnalisés)
5. Initier le provisionnement
6. Surveiller l'avancement du provisionnement

Ce flux de travail est utilisé pour tous les types de services : mobile, internet, IPTV et VoIP.

## Accéder à l'attribution de plan

**Depuis le catalogue de produits :**

**Depuis la page client :**

**Depuis la page des options supplémentaires :**

L'interface d'attribution de plan s'ouvre dans une fenêtre modale ou une page dédiée.

# Flux de travail étape par étape

## Étape 1 : Parcourir le catalogue de produits

Le catalogue de produits affiche les produits disponibles regroupés par catégorie et type de client.

**Bascules de vue :**

> Catégories : • Plans de service • Options supplémentaires

Basculer entre les filtres Individuel et Entreprise filtre les produits pour n'afficher que ceux disponibles pour le type de client sélectionné.

**Cartes de produits :**

Les produits sont affichés dans un carrousel ou une grille :

Cliquez sur **"Attribuer au client"** pour continuer.

# Étape 2 : Sélectionner un client

Si vous n'êtes pas déjà sur une page client, le système vous invite à sélectionner un client.

**Recherche de client :**

> Rechercher des clients : [John ▼]
>
> Clients correspondants : • John Smith (ID : 123) • John Doe (ID : 456) • Johnson Enterprises (ID : 789)

Tapez pour rechercher par :

- Nom du client
- ID du client
- Adresse e-mail
- Numéro de téléphone

Sélectionnez le client dans le menu déroulant.

> ⚠ **ATTENTION**
>
> Assurez-vous d'avoir sélectionné le bon client avant de continuer. Si vous attribuez un plan au mauvais client, cela nécessite une intervention manuelle pour corriger.

## Étape 3 : Configurer l'inventaire (si nécessaire)

Si le produit nécessite des éléments d'inventaire (définis dans `inventory_items_list`), des sélecteurs d'inventaire apparaissent.

**Exemple de sélecteur d'inventaire :**

> Éléments d'inventaire requis :
>
> Carte SIM *

> Cartes SIM disponibles : • SIM-00123 - ICCID : 8944...0001 (Neuve) • SIM-00124 - ICCID : 8944...0002 (Neuve) • SIM-00125 - ICCID : 8944...0003 (Neuve)
>
> Numéro de mobile *
>
> Numéros disponibles : • +44 7700 900123 (Réservé) • +44 7700 900124 (Disponible) • +44 7700 900125 (Disponible)

**Règles de sélection d'inventaire :**

- L'astérisque (*) indique un champ requis
- Seuls les éléments d'inventaire disponibles sont affichés (statut : "En stock" ou "Neuf")
- Les menus déroulants se chargent dynamiquement en fonction des noms de modèles d'inventaire
- Une fois sélectionnés, les éléments sont temporairement réservés

**Ce qui se passe :**

- Les éléments d'inventaire sélectionnés sont passés en tant que variables au playbook de provisionnement
- Pendant le provisionnement, les éléments sont attribués au service et au client
- Le statut de l'élément passe de "En stock" à "Attribué"

## Étape 4 : Configurer le renouvellement automatique (optionnel)

Pour les services récurrents, vous pouvez être invité à définir le renouvellement automatique :

> Souhaitez-vous activer le renouvellement automatique pour ce service ?
>
> Lorsqu'il est activé, ce service sera automatiquement renouvelé à la fin de chaque période de facturation et facturera le mode de paiement par défaut du client.
>
> [ Non ] [ Oui ]

**Comportement du renouvellement automatique :**

- **Oui** : Le service se renouvelle automatiquement, le client est facturé mensuellement
- **Non** : Le service expire à la fin de la période de contrat, un renouvellement manuel est requis

**Meilleure pratique :** Par défaut, optez pour "Oui" pour les services destinés aux consommateurs, "Non" pour les services ponctuels ou lorsque le client demande un contrôle manuel.

# Étape 5 : Vérifier et confirmer

L'écran de vérification montre toutes les sélections avant le provisionnement :

> Client : John Smith (ID : 123) Produit : Mobile - Plan 20 Go
>
> Inventaire : • Carte SIM : SIM-00123 (ICCID : 8944...0001) • Numéro de mobile : +44 7700 900123
>
> Tarification : • Frais d'installation : £0.00 • Coût mensuel : £15.00
>
> Renouvellement automatique : Oui
>
> [Annuler] [Confirmer & Provisionner]

Cliquez sur **"Confirmer & Provisionner"** pour initier le processus de provisionnement.

# Étape 6 : Avancement du provisionnement

La fenêtre modale de provisionnement affiche l'avancement en temps réel :

> ✓ Validation du compte client ✓ Attribution de la carte SIM (ICCID : 8944...0001) ✓ Attribution du numéro mobile (+44 7700 900123) ⟳ Configuration du compte OCS (en cours...) □ Création de l'enregistrement de service □ Envoi de l'e-mail de bienvenue
>
> Avancement : 3 des 6 tâches complétées

La fenêtre interroge l'API de provisionnement toutes les 0,2 secondes pour des mises à jour de statut.

**Indicateurs de progression :**

- ☑ Terminé avec succès
- ⟳ En cours d'exécution
- □ En attente (non commencé)
- ✗ Échoué (en cas d'erreurs)

## Étape 7 : Achèvement

**Succès :**

> Service provisionné avec succès pour John Smith
>
> ID du service : 789 Nom du service : Mobile - +44 7700 900123 Statut : Actif
>
> [Voir le service] [Fermer]

Cliquez sur **"Voir le service"** pour ouvrir la page des détails du service.

**Échec :**

Si le provisionnement échoue :

> Erreur : Impossible de se connecter à OCS
>
> L'enregistrement de service a été créé mais le provisionnement n'a pas été complété. Veuillez examiner l'erreur et réessayer.
>
> ID de provisionnement : 456
>
> [Voir les journaux] [Réessayer] [Fermer]

- **Voir les journaux** : Ouvre les détails du provisionnement avec des messages d'erreur
- **Réessayer** : Tente à nouveau le provisionnement
- **Fermer** : Quitte la fenêtre modale (l'enregistrement de service reste mais n'est pas activé)

# Cas spéciaux

## Ajout d'options supplémentaires à un service existant

Lors de l'attribution d'une option supplémentaire (catégorie : "addon") à un client qui a déjà un service :

1. **Détection automatique des services :**
    - Le système trouve les services existants du client
    - Filtre par service_type (l'option supplémentaire mobile n'affiche que pour les services mobiles)
    - Si le client a plusieurs services correspondants, il est invité à sélectionner lequel
2. **Aucun nouveau service créé :**
    - L'option supplémentaire se provisionne contre l'ID de service existant
    - Utilise le compte OCS du service existant
    - L'inventaire (le cas échéant) est attribué au service existant
3. **Playbook de provisionnement :**
    - Playbook différent de celui des services autonomes
    - Ajoute généralement un solde, des fonctionnalités ou des équipements au compte existant

**Exemple :**

> Services mobiles existants : • Mobile - +44 7700 900123 (ID : 789) • Mobile - +44 7700 900456 (ID : 790)
>
> Quel service doit recevoir cette option supplémentaire ? [Mobile - +44 7700 900123 ▼]
>
> [Annuler] [Continuer]

# Provisionnement pour les clients professionnels

Les clients professionnels peuvent avoir des exigences supplémentaires :

- **Sélection de site** - Choisir quel emplacement commercial reçoit le service
- **Attribution de contact** - Désigner des contacts de facturation/techniques
- **Champs personnalisés** - Numéros de compte, centres de coûts, numéros de commande

**Exemple de flux commercial :**

> Sélectionner le site d'installation : [Bureau de Londres - 123 Main St ▼]

> Contact de facturation : [Jane Doe - <jane@acme.com> ▼]
>
> Contact technique : [Bob Smith - <bob@acme.com> ▼]
>
> Numéro de commande : [PO-2025-001234_____]
>
> [Annuler] [Continuer]

### Attribution de service en masse

Pour attribuer le même plan à plusieurs clients (par exemple, migrations massives) :

1. Utilisez l'importation CSV (si disponible)
2. Ou attribuez individuellement avec des paramètres de modèle
3. L'inventaire doit être disponible en masse
4. Surveillez la file d'attente de provisionnement pour éviter une surcharge

## Flux de travail courants

### Flux de travail 1 : Nouveau service mobile

1. Le client entre dans le magasin en demandant un service mobile
2. Le personnel ouvre **Produits → Plans**
3. Basculer vers le type de client **Individuel**
4. Sélectionne **"Mobile - Plan 20 Go"**
5. Clique sur **"Attribuer au client"**
6. Recherche le client par téléphone : "+1234567890"
7. Sélectionne **John Smith** dans les résultats
8. Choisit une carte SIM dans le sélecteur d'inventaire
9. Choisit un numéro mobile disponible
10. Active le **renouvellement automatique**
11. Confirme et provisionne
12. Surveille l'avancement jusqu'à ce qu'il soit complet
13. Remet la SIM au client avec un paquet de bienvenue

## Flux de travail 2 : Ajout d'internet à un client existant

1. Accédez à la page client : **Clients → John Smith**
2. Cliquez sur l'onglet **Services**
3. Cliquez sur le bouton **"Ajouter un service"**
4. Parcourez les plans internet
5. Sélectionnez **"Fibre - 100Mbps"**
6. Sélectionnez le site d'installation (si entreprise/multiples sites)
7. Choisissez un modem CPE dans l'inventaire
8. Définissez la date d'installation (si nécessaire)
9. Provisionnez le service
10. Créez un ticket d'installation

## Flux de travail 3 : Attribution d'une option supplémentaire de recharge de données

1. Le client appelle : "J'ai besoin de plus de données"
2. Le personnel recherche le client dans la recherche globale
3. Ouvre l'onglet des services du client
4. Clique sur "Options supplémentaires" à côté du service mobile
5. Sélectionne "5 Go de boost de données"
6. Paiement autorisé
7. L'option supplémentaire se provisionne immédiatement
8. Le client reçoit un boost de données instantané

# Dépannage

**"Aucun produit disponible"**

- **Cause :** Les filtres excluent tous les produits
- **Solution :**
  - Basculer le type de client (Individuel vs Entreprise)
  - Vérifier que le catalogue de produits contient des produits activés

- Vérifier que les produits correspondent à l'éligibilité du client

**"Aucun inventaire disponible"**

- **Cause :** Les éléments d'inventaire sont en rupture de stock ou tous attribués
- **Solution :**
  - Ajouter plus d'éléments d'inventaire au système
  - Vérifier le statut des éléments (doit être "Neuf" ou "En stock")
  - Vérifier que les noms des modèles d'inventaire correspondent aux exigences des produits

**"Client non trouvé"**

- **Cause :** Le client n'existe pas ou le terme de recherche est incorrect
- **Solution :**
  - Créer le client d'abord
  - Essayer différents termes de recherche (ID, e-mail, téléphone)
  - Vérifier les fautes de frappe

**Le provisionnement est bloqué**

- **Cause :** Erreur de playbook ou système externe inaccessible
- **Solution :**
  - Attendre le délai d'attente (généralement 5 minutes)
  - Vérifier les journaux de provisionnement pour une erreur spécifique
  - Vérifier que les systèmes OCS et réseau sont en ligne
  - Réessayer le provisionnement après avoir corrigé le problème

**Le provisionnement réussit mais le service ne fonctionne pas**

- **Cause :** Compte OCS créé mais réseau non mis à jour
- **Solution :**
  - Vérifier que l'OCS a un compte
  - Vérifier que la SIM est activée dans le HLR/HSS
  - Vérifier le provisionnement réseau (RADIUS, DPI, etc.)
  - Revoir les tâches du playbook pour des étapes manquées

# Meilleures pratiques

**Avant d'attribuer :**

- Vérifiez que le client a un mode de paiement valide enregistré
- Confirmez l'éligibilité du client pour le produit (résidentiel vs entreprise)
- Assurez-vous que l'inventaire requis est disponible
- Passez en revue les conditions et les prix du produit avec le client

**Pendant l'attribution :**

- Vérifiez deux fois la sélection du client avant de confirmer
- Sélectionnez les bons éléments d'inventaire (vérifiez les numéros de série)
- Activez le renouvellement automatique pour plus de commodité (sauf si le client s'y oppose)
- Surveillez l'avancement du provisionnement jusqu'à l'achèvement

**Après l'attribution :**

- Vérifiez que le service apparaît dans la liste des services du client
- Vérifiez que le statut du service est "Actif"
- Confirmez que l'inventaire est attribué correctement
- Envoyez un e-mail de bienvenue ou des instructions au client
- Testez le service si possible (effectuez un appel test, vérifiez les données)

**Pour les options supplémentaires :**

- Confirmez que l'option supplémentaire est compatible avec le service existant
- Expliquez la facturation (ponctuelle vs récurrente)
- Vérifiez le paiement avant le provisionnement
- Vérifiez que le solde est mis à jour immédiatement après le provisionnement de l'option supplémentaire

# Documentation connexe

- `csa_add_service` - Aperçu des services
- `concepts_products_and_services` - Concepts du catalogue de produits
- `administration_inventory` - Gestion des éléments d'inventaire
- `concepts_provisioning` - Détails du système de provisionnement
- `guide_product_lifecycle` - Cycle de vie complet du produit, y compris le provisionnement

# Enregistrements de Détails d'Appels (CDRs)

Les Enregistrements de Détails d'Appels montrent les **appels, messages SMS et sessions de données** individuels pour un service. Alors que Utilisation du Service montre combien de solde est *resté*, les Enregistrements d'Appels montrent les *événements* qui l'ont utilisé : qui a été appelé, quand, pendant combien de temps, et combien cela a coûté.

## Ouverture des Enregistrements d'Appels

Les Enregistrements d'Appels se trouvent dans la fenêtre **Utilisation du Service**, à côté de la vue du Solde.

**Depuis un service :**

1. Cliquez sur le bouton **Solde** (ou **Utilisation**) à côté du service
2. Sélectionnez l'onglet **Enregistrements d'Appels (CDRs)**

Vous pouvez également l'ouvrir depuis le bouton **CDRs** sur l'écran Modifier le Service. Pour les clients, ouvrez-le depuis **Voir les Enregistrements d'Appels** sur la page d'utilisation du Portail d'Auto-Service.

## Lecture de la Liste

Les enregistrements sont listés avec les **plus récents en premier**. Chaque ligne montre :

| Colonne | Ce qu'elle montre |
| --- | --- |
| **Heure de Configuration** | Quand l'appel, le message ou la session a commencé |
| **Destination** | Le numéro appelé, le destinataire du SMS, ou la destination des données |
| **Catégorie** | S'il s'agissait d'un événement **Appel**, **SMS**, ou **Données** |
| **Utilisation** | Durée de l'appel (`HH:MM:SS`), montant des données (Mo/Go), ou nombre de SMS |
| **Coût** | Ce que l'événement a coûté |

Utilisez **Précédent** et **Suivant** en bas pour naviguer à travers les enregistrements plus anciens.

**Cliquez sur n'importe quelle ligne** pour ouvrir les détails complets de cet enregistrement, y compris le temps de réponse et les informations supplémentaires enregistrées à son sujet.

# Trouver des Enregistrements Spécifiques

Utilisez la barre de filtre en haut pour affiner la liste, puis cliquez sur **Appliquer** :

- **Boutons rapides (1h / 24h / 7j / 30j)** : montrent les enregistrements de la dernière heure, jour, semaine, ou mois.
- **De / À** : choisissez votre propre date et heure de début et de fin.
- **La destination commence par** : tapez les premiers chiffres d'un numéro pour montrer uniquement les appels correspondants (par exemple, `44` pour les numéros du Royaume-Uni).

- **Catégorie** : montrez uniquement les **Appels**, **SMS**, ou **Données** (ou **Tous**).

**Réinitialiser** efface les filtres et montre à nouveau les enregistrements les plus récents.

> **Vous ne voyez rien ?** La vue s'ouvre sur les enregistrements les plus récents, mais si vous avez choisi une plage de dates, le service n'a peut-être tout simplement eu aucune activité pendant cette période. Essayez **Réinitialiser**, ou élargissez les dates **De / À**.

# Téléchargement des Enregistrements

Cliquez sur **Exporter CSV** pour télécharger les enregistrements actuellement affichés (après vos filtres) sous forme de fichier tableur. Cela est pratique pour le partage avec un client ou pour garder une trace d'une question de facturation.

# Qui Peut Voir les Enregistrements d'Appels

- **Le personnel** a besoin de la permission des enregistrements d'appels. Si vous ne voyez pas l'onglet **Enregistrements d'Appels**, demandez à votre administrateur de l'activer pour votre compte.
- **Les clients** peuvent voir les enregistrements pour **leurs propres services uniquement** via le Portail d'Auto-Service.

Les Enregistrements d'Appels sont **en lecture seule**. Ils ne peuvent pas être modifiés ou supprimés depuis ici.

# Modification d'un Service

Les services peuvent être modifiés par le client final via le Portail de Self-Service, ou par un administrateur via le portail administrateur.

Une fois qu'un service est provisionné, vous pouvez modifier ses paramètres, ajouter des améliorations ou changer des paramètres.

## Modification des Paramètres du Service

Les paramètres de base du service peuvent être modifiés en cliquant sur le bouton **Modifier** sur la page de détails du service.

**Champs Modifiables :**

- Nom du Service
- Statut du Service (Actif, Inactif, Suspendu)
- Notes du Service
- Paramètres de Renouvellement Automatique
- Articles d'inventaire liés
- Champs personnalisés

Les modifications sont enregistrées immédiatement et reflétées dans le portail de self-service du client.

# Ajout d'Options au Service

Les options améliorent les services existants avec des fonctionnalités supplémentaires, des allocations de données ou du matériel. Les types d'options courants incluent :

- **Recharges de données** - Allocation de données supplémentaire (par exemple, "5 Go de Boost de Données")
- **Mises à niveau de fonctionnalités** - Capacités supplémentaires (par exemple, "Appels Internationaux")
- **Locations de matériel** - Ajouts d'équipement (par exemple, "Modem WiFi 6")
- **Services Premium** - Fonctionnalités améliorées (par exemple, "Adresse IP Statique")

## Accès au Catalogue d'Options

Depuis une page de détails de service :

1. Accédez au service que vous souhaitez améliorer
2. Cliquez sur le bouton **"Ajouter une Option"** ou **"Parcourir les Options"**
3. Le catalogue d'options s'ouvre, filtré pour afficher uniquement les options compatibles

**Filtrage Automatique :**

Le système filtre automatiquement les options en fonction de :

- **Type de Service** - Affiche uniquement les options correspondant au type de service (mobile, internet, voip, etc.)
- **Type de Client** - Filtre par client résidentiel ou professionnel
- **Compatibilité du Service** - Vérifie si le service répond aux exigences de l'option
- **Disponibilité** - Affiche uniquement les produits activés

Par exemple, si vous visualisez un service mobile résidentiel, vous ne voyez que les options mobiles marquées pour les clients résidentiels.

## Affichage des Options

Les options sont affichées dans un carrousel interactif montrant :

**Affichage de la Carte Produit :**

**Informations Affichées :**

- Icône du produit

- Nom du produit
- Liste des fonctionnalités (points de balle)
- Coût d'installation
- Coût mensuel/récurrent
- Lien vers les termes et conditions
- Bouton "Ajouter au Service"

# Provisionnement d'une Option

**Étape 1 : Sélectionner l'Option**

Cliquez sur la carte d'option souhaitée, puis cliquez sur **"Ajouter au Service"**

**Étape 2 : Sélectionner l'Inventaire (si nécessaire)**

Si l'option nécessite un inventaire physique (par exemple, location de matériel), un sélecteur d'inventaire apparaît :

> Modems Disponibles : • Modem-12345 - TP-Link AX1800 (Neuf) • Modem-12346 - TP-Link AX1800 (Neuf) • Modem-12347 - Netgear RAX40 (Neuf)

Sélectionnez l'article d'inventaire spécifique à attribuer à ce service.

**Étape 3 : Configurer le Renouvellement Automatique (facultatif)**

Pour les options récurrentes, vous pouvez être invité :

> Souhaitez-vous activer le renouvellement automatique pour cette option ?
>
> [ Non ] [ Oui ]

- **Oui** - L'option se renouvelle automatiquement à chaque période de facturation
- **Non** - Achat unique, le client doit renouveler manuellement

**Étape 4 : Confirmer et Provisionner**

Examinez les détails de l'option et cliquez sur **"Confirmer"**

Le modal de provisionnement apparaît montrant la progression en temps réel :

> ✓ Validation du paiement ✓ Attribution de l'inventaire ⟳ Configuration du service (en cours...) □ Création de la transaction □ Envoi de l'email de confirmation

Une fois terminé, l'option apparaît dans la liste des options du service.

**Provisionnement en Coulisses :**

Lorsque vous ajoutez une option, le système :

1. Valide que le client peut acheter l'option
2. Exécute le playbook Ansible de l'option (`provisioning_play`)
3. Attribue les articles d'inventaire requis au service
4. Crée un enregistrement de transaction pour la facturation
5. Met à jour la configuration du service (OCS, systèmes réseau, etc.)
6. Envoie une notification de confirmation au client

Pour des détails techniques sur le provisionnement des options, voir Guide Complet du Cycle de Vie du Produit - Ajout d'Options.

## Affichage des Options Actives

Les options actives apparaissent sur la page de détails du service dans la section **"Options"** :

**Affichage Exemple :**

> □ 5 Go de Boost de Données
> Ajouté : 10 Jan 2025 Expire : 17 Jan 2025 Statut : Actif Coût : 5,00 £
>
> □ Location de Modem WiFi 6
> Ajouté : 01 Déc 2024 Inventaire : Modem-12345 Renouvellement Automatique : Oui Mensuel : 10,00 £

## Suppression d'Options

Pour supprimer une option d'un service :

1. Accédez à la page de détails du service

2. Localisez l'option dans la section **"Options Actives"**
3. Cliquez sur le bouton **"Supprimer"** ou l'**icône de poubelle** à côté de l'option
4. Confirmez la suppression

**Ce Qui Se Passe :**

- **Options matérielles** - Le playbook de déprovisionnement s'exécute, l'inventaire est marqué pour retour
- **Options virtuelles** - Les avantages sont supprimés immédiatement
- **Options à renouvellement automatique** - Annule les renouvellements futurs
- **Transaction créée** - Crédite tout montant au prorata si applicable

> ⚠ **ATTENTION**
>
> La suppression d'options matérielles (modems, CPE, etc.) nécessite généralement le retour de l'équipement. Le système marquera l'inventaire pour retour et peut envoyer des instructions de retour au client.

# Scénarios Courants d'Options

**Scénario 1 : Client à Court de Données**

1. Le client contacte le support : "J'ai utilisé mon allocation mensuelle"
2. Le personnel navigue vers le service mobile du client
3. Clique sur "Ajouter une Option"
4. Sélectionne "5 Go de Boost de Données"
5. Provisionne immédiatement
6. Le client reçoit un top-up de données instantané

**Scénario 2 : Besoin d'une IP Statique pour une Entreprise**

1. Le client professionnel demande une IP statique pour VPN
2. Le personnel ouvre le service internet du client
3. Parcourt les options, sélectionne "Adresse IP Statique"

4. Le système provisionne l'IP à partir du pool disponible (inventaire)
5. Configure le routage dans l'équipement réseau
6. L'entreprise reçoit les détails de configuration IP

**Scénario 3 : Location d'Équipement pour un Service Fibre**

1. Le client s'inscrit pour internet fibre
2. Pendant le provisionnement, le client opte pour la location de modem
3. Le personnel ajoute l'option "Modem WiFi 6"
4. Sélectionne le modem disponible dans l'inventaire
5. Le modem est expédié au client
6. Un frais récurrent de 10,00 £/mois est ajouté au compte

# Dépannage

**"Aucune option compatible disponible"**

- **Cause :** Aucune option ne correspond au type de service ou au type de client
- **Solution :** Vérifiez que le catalogue de produits contient des options avec des paramètres `service_type` et `residential`/`business` correspondants

**Échec du provisionnement de l'option**

- **Cause :** Erreur de playbook de provisionnement ou inventaire non disponible
- **Solution :**
  - Vérifiez les journaux de provisionnement pour une erreur spécifique
  - Vérifiez que les articles d'inventaire sont en stock (si nécessaire)
  - Consultez les journaux de playbook dans le système de provisionnement

**Le sélecteur d'inventaire ne montre aucun article**

- **Cause :** Aucun article d'inventaire disponible du type requis
- **Solution :**

    - Ajoutez des articles d'inventaire au système
    - Vérifiez que les articles existants ne sont pas tous attribués ou endommagés
    - Vérifiez que le nom du modèle d'inventaire correspond exactement à `inventory_items_list`

**Le renouvellement automatique ne fonctionne pas**

- **Cause :** Le drapeau de renouvellement automatique n'est pas défini ou le mode de paiement a expiré
- **Solution :**
    - Vérifiez que le service a `auto_renew: true`
    - Vérifiez que le client a un mode de paiement valide enregistré
    - Consultez les tâches planifiées dans le système de facturation

# Documentation Connexe

- `guide_product_lifecycle` - Flux complet de provisionnement des options
- `csa_add_service` - Création de nouveaux services
- `concepts_products_and_services` - Concepts de produits et de services
- `administration_inventory` - Gestion de l'inventaire pour les options matérielles

# Gestion des Services

L'interface de Gestion des Services fournit des outils complets pour visualiser, filtrer et gérer les services clients au sein de votre organisation.

Documentation connexe : Ajout de Services, Utilisation des Services, Modification des Services, Produits & Services.

## Accéder à la Liste des Services

Naviguez vers :

Ou directement :

Cela affiche tous les services avec des capacités de filtrage, de tri et de gestion en masse.

## Aperçu de la Liste des Services

La liste des services affiche les services dans un tableau avec :

**Colonnes :**

- **ID du Service** - Identifiant unique
- **Nom du Service** - Nom descriptif (par exemple, "Mobile - +44 7700 900123")
- **Nom du Client** - Client associé (lien cliquable)
- **Type de Service** - mobile, iptv, internet, voip
- **Statut** - Actif, Inactif, Suspendu
- **Date de Début** - Date de début du service
- **Date de Fin** - Date d'expiration/fin du service
- **Coût Mensuel** - Prix de détail
- **Actions** - Voir, Modifier, Options supplémentaires, Utilisation, Supprimer

# Filtrer les Services

## Onglets de Statut

Filtrez rapidement les services par statut en utilisant la barre d'onglets :

- **Tous les Résultats** - Affiche tous les services, quel que soit le statut
- **Actif** - Services actuellement actifs (vue par défaut)
- **Inactif** - Services annulés ou expirés
- **Suspendu** - Services temporairement désactivés (non-paiement, fraude, etc.)

Cliquer sur un onglet met à jour la liste instantanément. L'onglet actif est mis en surbrillance.

## Filtre par Type de Service

Filtrer par type de service :

Sélectionnez un ou plusieurs types pour afficher uniquement les services correspondants.

**Cas d'Utilisation :**

- **Mobile** - Voir tous les services SIM mobiles

- **Internet** - Afficher fibre, DSL et sans fil fixe
- **IPTV** - Afficher les services d'abonnement TV
- **VoIP** - Lister les services de voix sur IP

## Filtre par Client

Rechercher et filtrer par client :

Commencez à taper un nom de client pour voir les résultats correspondants :

Sélectionnez un client pour afficher uniquement ses services.

**Cas d'Utilisation :** Voir rapidement tous les services pour un client spécifique.

## Recherche

Recherche globale dans tous les champs de service :

Recherche :

- Nom du service
- ID du service
- Nom du client
- Numéros de téléphone (pour les services mobiles)
- UUID du service

**Exemple :** Recherchez "0770" pour trouver tous les services avec des num��ros de téléphone contenant "0770".

# Trier les Services

Cliquez sur n'importe quel en-tête de colonne pour trier par ce champ :

**Colonnes Triables :**

- ID du Service (par défaut : le plus récent en premier)
- Nom du Service (alphabétique)

- Nom du Client (alphabétique)
- Type de Service (alphabétique)
- Date de Début (chronologique)
- Date de Fin (chronologique)
- Coût Mensuel (numérique)

**Direction de Tri :**

- Cliquez une fois : Ascendant (A-Z, le plus ancien-le plus récent, le plus bas-le plus élevé)
- Cliquez deux fois : Descendant (Z-A, le plus récent-le plus ancien, le plus élevé-le plus bas)
- La colonne de tri active affiche un indicateur ▲ ou ▼

**Menu Déroulant de Tri :**

Le menu déroulant en haut à droite fournit des préréglages de tri rapides :

# Voir les Détails du Service

Cliquez sur n'importe quel nom de service pour ouvrir la vue détaillée du service.

**Onglets de Détails du Service :**

1. **Aperçu** - Résumé du service, statut, dates, tarification
2. **Inventaire** - Équipement attribué (cartes SIM, modems, etc.)
3. **Transactions** - Charges, crédits, paiements
4. **Utilisation** - Statistiques d'utilisation des données/voix/SMS
5. **Options supplémentaires** - Options supplémentaires actives et disponibles
6. **Journal d'Activité** - Historique des modifications

## Actions Rapides

Depuis la liste des services, cliquez sur le menu d'action ( ⋮ ) pour des opérations rapides :

- **Voir** - Ouvre les détails du service
- **Modifier** - Modifier les paramètres du service
- **Options supplémentaires** - Parcourir et ajouter des améliorations de service
- **Utilisation** - Voir l'utilisation actuelle et le solde
- **Supprimer** - Annuler/retirer le service

# Opérations en Masse

Sélectionnez plusieurs services à l'aide des cases à cocher pour effectuer des actions en masse.

## Sélectionner des Services

**Sélection Individuelle :**

Cliquez sur la case à cocher à côté de chaque service que vous souhaitez sélectionner.

**Sélectionner Tout :**

Cliquez sur la case à cocher dans l'en-tête du tableau pour sélectionner tous les services visibles.

> ☑ Service 1 ☑ Service 2 ☑ Service 3

**Compteur de Sélection :**

L'interface affiche combien de services sont sélectionnés :

## Actions en Masse

Une fois les services sélectionnés, des boutons d'action en masse apparaissent :

**Supprimer Plusieurs Services :**

1. Sélectionnez les services à supprimer
2. Cliquez sur le bouton **"Supprimer Sélectionnés"**
3. Confirmez la suppression dans la fenêtre modale

> ⚠ **ATTENTION**
>
> La suppression en masse est permanente et va :
>
> - Annuler tous les services sélectionnés
> - Marquer l'inventaire comme non attribué
> - Arrêter toutes les charges récurrentes
> - Créer des entrées dans le journal d'activité

**Cas d'Utilisation :**

- Nettoyer les services de test
- Annuler les services pour des comptes clients fermés
- Supprimer des entrées en double ou erronées

**Meilleure Pratique :** Utilisez des filtres pour réduire la liste avant les opérations en masse afin d'éviter des suppressions accidentelles.

# Gestion du Statut des Services

## Types de Statut

Les services peuvent avoir trois statuts :

**Actif**

- Le service est opérationnel
- Des charges s'appliquent
- Le client peut utiliser le service
- Affiché avec un badge vert

**Inactif**

- Le service a été annulé ou a expiré
- Aucune charge appliquée
- Le client ne peut pas utiliser le service
- Affiché avec un badge gris
- Inventaire marqué pour retour/réparation

**Suspendu**

- Service temporairement désactivé
- Peut ou non facturer (configurable)
- Le client ne peut pas utiliser le service
- Affiché avec un badge orange/jaune
- Raisons courantes : Non-paiement, enquête sur la fraude, demande du client

# Changer le Statut d'un Service

Pour changer le statut d'un service :

1. Ouvrez les détails du service
2. Cliquez sur le bouton **"Modifier"**
3. Changez le menu déroulant **"Statut du Service"**
4. Cliquez sur **"Enregistrer"**

**Ce Qui Se Passe :**

- **Actif → Inactif :** Déclenche le playbook de déprovisionnement (si configuré)
- **Actif → Suspendu :** Désactive le service dans OCS/réseau mais conserve l'enregistrement

- **Suspendu → Actif :** Réactive le service, reprend la facturation
- **Inactif → Actif :** Peut déclencher le reprovisionnement (utiliser avec précaution)

# Vue d'Utilisation du Service

Cliquez sur **"Utilisation"** dans le menu d'actions pour ouvrir la fenêtre modale d'utilisation.

**Informations Affichées :**

- **Solde** - Crédit restant ou valeur prépayée
- **Utilisation des Données** - Utilisé vs. alloué (barre de progression)
- **Utilisation de la Voix** - Minutes utilisées vs. allocation du plan
- **Utilisation des SMS** - Messages envoyés vs. allocation
- **Date d'Expiration** - Quand le solde/plan actuel expire

**Actions :**

- **Recharger** - Ajouter du crédit ou des données (pour les services prépayés)
- **Voir Détails** - Voir la répartition détaillée de l'utilisation
- **Exporter** - Télécharger le rapport d'utilisation (si disponible)

# Renouvellements Automatiques à Venir

La vue des Renouvellements Automatiques à Venir fournit une interface centralisée pour surveiller et gérer tous les renouvellements de services programmés au sein de votre organisation.

## Accéder aux Renouvellements Automatiques

Naviguez vers :

Ou directement :

Cela affiche tous les services avec renouvellement automatique programmé, triés par date de prochain renouvellement.

## Aperçu des Renouvellements Automatiques

La liste des renouvellements automatiques affiche les renouvellements programmés dans un tableau avec :

**Colonnes :**

- **Client** - Nom du client (lien cliquable vers l'aperçu du client)
- **Service** - Nom du service (lien cliquable vers l'aperçu du client)
- **Produit** - Nom du produit/plan renouvelé
- **Coût** - Coût de renouvellement (à partir du prix de détail du produit)
- **Renouvelle Le** - Date et heure du prochain renouvellement avec format lisible par l'homme
- **Statut** - Statut du service (Actif, Suspendu, etc.)
- **Actions** - Renouveler maintenant ou supprimer le renouvellement automatique

**Exemple d'Affichage :**

## Comment Fonctionne le Renouvellement Automatique

Le renouvellement automatique est programmé dans CGRateS (le système de facturation) lorsqu'un service est créé ou modifié. Le système :

1. **Programme l'Action** - Crée un ActionPlan dans CGRateS avec la date de renouvellement
2. **Surveille le Solde** - Vérifie si le client a un solde suffisant avant le renouvellement
3. **Exécute le Renouvellement** - À la date prévue, renouvelle automatiquement le service
4. **Met à Jour les Enregistrements** - Crée des transactions, met à jour les dates de service et enregistre l'activité

**Source de Données :**

La vue des Renouvellements Automatiques à Venir interroge directement CGRateS en utilisant l'API `ApierV1.GetScheduledActions` et enrichit les données avec des informations sur le client et le service à partir de la base de données CRM.

## Renouveler Maintenant

Pour déclencher manuellement un renouvellement avant la date prévue :

1. Cliquez sur le bouton  **(Renouveler Maintenant)** pour le service

2. Examinez les détails du renouvellement dans la fenêtre de confirmation :

> Client : Acme Corp Service : FixedWireless_75628fa5 Produit : Internet à Domicile Mega Coût : $89.99
>
> Prochain renouvellement prévu : 11 nov. 2025 10:45 AM
>
> [Annuler] [Confirmer le Renouvellement]

3. Cliquez sur **"Confirmer le Renouvellement"** pour traiter immédiatement
4. Un travail de provisionnement est créé et exécuté
5. Surveillez l'avancement dans la fenêtre modale de statut de provisionnement

**Ce Qui Se Passe :**

- Le service est rechargé avec les allocations du produit
- Le solde est mis à jour dans CGRateS
- Un enregistrement de transaction est créé
- La date de fin du service est prolongée
- Une entrée dans le journal d'activité est créée

**Cas d'Utilisation :**

- Le client demande un renouvellement anticipé
- Résoudre des problèmes de service en réinitialisant les allocations
- Tester les processus de renouvellement
- Le client souhaite utiliser le service avant le renouvellement prévu

# Supprimer le Renouvellement Automatique

Pour annuler le renouvellement automatique d'un service :

1. Cliquez sur le bouton  **(Supprimer le Renouvellement Automatique)** pour le service
2. Confirmez la suppression dans la fenêtre modale :

> Cela supprimera le renouvellement automatique prévu pour :
>
> Client : Acme Corp Service : FixedWireless_75628fa5 Prochain Renouvellement : 11 nov. 2025 10:45 AM
>
> Le service ne se renouvellera pas automatiquement. Vous devrez le renouveler manuellement ou le service expirera à la date de fin.
>
> [Annuler] [Supprimer le Renouvellement Automatique]

3. Cliquez sur **"Supprimer le Renouvellement Automatique"** pour confirmer

4. L'ActionPlan est supprimé de CGRateS

5. Le service expirera naturellement à moins d'être renouvelé manuellement

**Ce Qui Se Passe :**

- L'ActionPlan est supprimé de CGRateS
- Le statut du service reste inchangé
- Le service expirera à sa date de fin actuelle
- Le client doit renouveler manuellement ou le service s'arrêtera

**Cas d'Utilisation :**

- Le client annule le service
- Passer à un processus de renouvellement manuel
- Le plan de service change
- Le client a demandé d'arrêter les charges automatiques

> ⚠ **ATTENTION**
>
> Supprimer le renouvellement automatique signifie que le service expirera à moins d'être renouvelé manuellement. Les clients ne recevront pas de continuation automatique du service.

# Comprendre l'Affichage

**Colonne Client :**

- Affiche le nom du client sous forme de lien cliquable
- Lien vers la page d'aperçu du client
- Affiche "N/A" si le service n'est pas lié à un client (service orphelin)

**Colonne Service :**

- Affiche le nom/UUID du service
- Lien vers la page d'aperçu du client
- Affiche l'UUID du service si le nom n'est pas défini
- Affiche "N/A" si le service ne peut pas être trouvé dans la base de données

**Colonne Produit :**

- Nom du produit à partir de l'ActionPlan CGRateS
- Extrait des métadonnées de l'action programmée
- Affiche toujours le nom du produit même si le service n'est pas trouvé

**Colonne Coût :**

- Coût de détail à partir de la table Produit dans le CRM
- Montant qui sera facturé lors du renouvellement
- Affiche "N/A" si le produit n'est pas trouvé dans la base de données

**Colonne Renouvelle Le :**

- Date et heure exactes du prochain renouvellement
- Temps relatif lisible par l'homme (par exemple, "dans 3 heures", "dans 2 jours")
- Sensible au fuseau horaire (utilise le fuseau horaire du serveur CGRateS)

**Colonne Statut :**

- Statut actuel du service à partir de la base de données CRM
- **Actif** (Vert) - Le service est opérationnel

- **Suspendu** (Orange) - Le service est temporairement désactivé
- **Inconnu** (Gris) - Service non trouvé dans la base de données ou statut non défini

# Dépannage

### Services affichant "N/A" pour Client/Service

- **Cause :** L'UUID du service dans CGRateS ne correspond pas à service_uuid dans la base de données CRM
- **Solution :**
  - Vérifiez que le service existe dans la base de données
  - Vérifiez que le format service_uuid correspond : `ServiceType_UUID` (par exemple, `FixedWireless_75628fa5`)
  - Le service peut avoir été supprimé du CRM mais est toujours programmé dans CGRateS

### Coût affichant "N/A"

- **Cause :** L'ID du produit dans CGRateS n'existe pas dans la table Produit du CRM
- **Solution :**
  - Vérifiez que le produit existe dans la base de données
  - Vérifiez que product_id dans l'ActionPlan correspond à un produit dans le CRM
  - Le produit peut avoir été supprimé

### "Renouvelle Le" n'affiche pas de date

- **Cause :** NextRunTime non présent dans la réponse CGRateS
- **Solution :**
  - Vérifiez la configuration de l'ActionPlan CGRateS
  - Vérifiez que ActionTiming est correctement configuré
  - Vérifiez les journaux CGRateS pour des erreurs

### "Renouveler Maintenant" échoue

- **Cause :** Diverses erreurs de provisionnement
- **Dépannage :**
  - Vérifiez la fenêtre modale de statut de provisionnement pour des détails d'erreur
  - Vérifiez que les playbooks Ansible sont configurés correctement
  - Vérifiez la connectivité OCS
  - Examinez les journaux de provisionnement

**Renouvellement automatique supprimé mais toujours affiché**

- **Cause :** Délai de cache ou problème de synchronisation CGRateS
- **Solution :**
  - Cliquez sur le bouton **"Rafraîchir"** pour recharger les données
  - Vérifiez que l'ActionPlan a réellement été supprimé dans CGRateS
  - Vérifiez la connectivité API CGRateS

# Pagination

Les services sont affichés en pages pour des performances optimales :

> ◄ Précédent 1 [2] 3 4 5 Suivant ►
>
> Éléments par page : [10 ▼]
>
> - 10
> - 25
> - 50
> - 100

**Contrôles :**

- **Précédent/Suivant** - Naviguer entre les pages
- **Numéros de Page** - Passer à une page spécifique
- **Éléments par Page** - Ajuster le nombre de services affichés par page

**Astuce de Performance :** Utilisez des filtres pour réduire le nombre total de résultats plutôt que d'augmenter le nombre d'éléments par page.

# Badges et Indicateurs de Service

Des indicateurs visuels aident à identifier rapidement les états des services :

**Badges de Statut :**

**Indicateur de Renouvellement Automatique :**

Les services avec renouvellement automatique activé affichent :

**Expiration Proche :**

Les services expirant dans les 7 jours affichent :

**En Retard :**

Les services avec un solde impayé affichent :

# Intégration CGRateS (Avancé)

Pour les services intégrés à CGRateS (le moteur de facturation et de tarification), les administrateurs peuvent gérer des configurations avancées directement depuis la vue du service. Cela inclut des attributs, des filtres et la visualisation des sessions actives.

> **REMARQUE**
>
> Les fonctionnalités d'intégration CGRateS nécessitent la permission **cgrates_api_access**. Seuls les administrateurs ont accès à ces fonctionnalités par défaut. Voir `rbac` pour la configuration des permissions.

**Provisionnement Automatique vs. Gestion Manuelle**

En fonctionnement normal, les attributs et filtres CGRateS sont automatiquement provisionnés par Ansible lors du flux de travail de provisionnement initial du service. Lorsqu'un nouveau service est créé, les playbooks de provisionnement :

- Créent le compte de service dans CGRateS
- Configurent les attributs (IMSI, MSISDN, identifiants de compte, profils de vitesse, etc.)
- Configurent des filtres pour garantir une tarification correcte
- Appliquent les plans de tarification appropriés

Cependant, il existe des cas où vous devrez peut-être modifier ces configurations après que le service a été provisionné :

- **Changements de Profil de Vitesse** - Le client met à niveau/diminue la bande passante (MaxBitrateDL/UL)
- **Ajustements de Politique** - Changer les politiques QoS ou les règles de gestion du trafic (PcefPolicyName)
- **Changements de Numéro de Téléphone** - Mettre à jour MSISDN ou d'autres identifiants
- **Dépannage** - Corriger des erreurs de configuration ou tester différents paramètres
- **Configurations Spéciales** - Appliquer des attributs personnalisés non inclus dans le provisionnement standard

L'interface de gestion manuelle permet aux administrateurs de faire ces modifications directement sans relancer l'ensemble du flux de travail de provisionnement. Cela est particulièrement utile pour :

- **Changements Rapides** - Modifier un seul attribut sans attendre le provisionnement
- **Tests** - Expérimenter avec différentes configurations
- **Support Client** - Résoudre des problèmes en temps réel lors des appels de support
- **Configurations Personnalisées** - Appliquer des paramètres spécifiques au service non couverts par des modèles

> ⚠ **ATTENTION**
>
> Les modifications manuelles de la configuration CGRateS contournent le flux de travail de provisionnement standard. Assurez-vous de comprendre l'impact de vos modifications, car des configurations incorrectes peuvent

> affecter la facturation et la fonctionnalité du service. Toutes les modifications sont enregistrées dans le fil d'activité du client à des fins d'audit.

## Accéder aux Fonctionnalités CGRateS

Lors de la visualisation ou de la modification d'un service qui est provisionné dans CGRateS, trois sections repliables apparaissent en bas du formulaire de service :

- **Attributs CGRateS** - Configurer des attributs spécifiques au service
- **Filtres CGRateS** - Définir des règles de filtrage pour le service
- **Sessions Actives** - Voir les sessions actives en temps réel

Chaque section est repliée par défaut pour garder l'interface propre. Cliquez sur l'en-tête de la section pour développer et voir/modifier la configuration.

Les sections repliées affichent des badges de compte indiquant combien d'attributs, de filtres ou de sessions actives existent pour le service.

# Attributs CGRateS

Les attributs vous permettent de définir des champs personnalisés et des transformations qui sont appliqués aux événements de tarification pour ce service spécifique.

**Format de l'ID d'Attribut :** `ATTR_ACCOUNT_{service_uuid}`

**Exemple :** Pour un service avec UUID `Mobile_SIM_c2880638`, l'ID de profil d'attribut est `ATTR_ACCOUNT_Mobile_SIM_c2880638`

**Gestion des Attributs :**

1. Ouvrez la vue d'édition du service
2. Développez la section **"Attributs CGRateS"**
3. Cliquez sur le bouton **"Modifier les Attributs"**
4. Ajoutez/modifiez/supprimez des attributs selon les besoins
5. Cliquez sur **"Enregistrer les Attributs"**

**Champs d'Attribut :**

- **Path** - Le champ à modifier (par exemple, `*req.Account`, `*req.IMSI`)
- **Type** - Comment la valeur est définie :

    - `*constant` - Définit une valeur fixe
    - `*variable` - Capture la valeur des champs d'événement en utilisant RSRParser
    - `*composed` - Ajoute la valeur au lieu de l'écraser
    - `*usage_difference` - Calcule la durée entre deux champs
    - `*sum` - Somme plusieurs valeurs
    - `*value_exponent` - Calcule l'exposant d'un champ
- **Rules** - La ou les valeurs à appliquer (peut avoir plusieurs règles par attribut)

**Configuration d'Attribut Exemple :**

```
{
  "Path": "*req.Account",
  "Type": "*constant",
  "Value": [{"Rules": "Mobile_SIM_474a380a"}]
}
```

**Cas d'Utilisation Courants :**

- Définir l'identifiant de compte pour la tarification
- Mapper IMSI/MSISDN au service
- Configurer des limites de bande passante (MaxBitrateDL/UL)
- Définir des noms de politique (PcefPolicyName)
- Transformer ou enrichir les événements de tarification

**Journal d'Activité :**

Toutes les modifications d'attributs sont enregistrées dans le fil d'activité du client avec tous les détails des modifications effectuées.

# Filtres CGRateS

Les filtres définissent des règles de correspondance qui déterminent quand la configuration de ce service doit être appliquée lors de la tarification.

**Format de l'ID de Filtre :** `FLTR_ACCOUNT_{service_uuid}`

**Exemple :** Pour un service avec UUID `Mobile_SIM_c2880638`, l'ID de filtre est `FLTR_ACCOUNT_Mobile_SIM_c2880638`

**Gestion des Filtres :**

1. Ouvrez la vue d'édition du service
2. Développez la section **"Filtres CGRateS"**
3. Cliquez sur le bouton **"Modifier les Filtres"**
4. Ajoutez/modifiez/supprimez des règles de filtre
5. Cliquez sur **"Enregistrer les Filtres"**

**Champs de Règle de Filtre :**

- **Element** - Le champ à faire correspondre (par exemple, `~*req.Account`, `~*req.Destination`)
- **Type** - Type de correspondance :
  - `*string` - Correspondance exacte de chaîne
  - `*prefix` - Commence par la valeur spécifiée
  - `*suffix` - Se termine par la valeur spécifiée
  - `*empty` - Le champ est vide
  - `*exists` - Le champ existe
  - `*notexists` - Le champ n'existe pas
  - `*timings` - Correspondre aux motifs de temps/date
  - `*destinations` - Correspondre aux motifs de destination
  - `*rsr` - Correspondance de champ RSR
  - `*gt` / `*gte` / `*lt` / `*lte` - Comparaisons numériques
- **Values** - La ou les valeurs à faire correspondre (peut avoir plusieurs valeurs par règle)

**Configuration de Filtre Exemple :**

```
{
  "Element": "~*req.Account",
  "Type": "*string",
  "Values": ["Mobile_SIM_474a380a"]
}
```

**Cas d'Utilisation Courants :**

- Assurer que les attributs ne s'appliquent qu'à un compte spécifique
- Filtrer par destination (nationale vs. internationale)
- Filtrage basé sur le temps (heures de pointe vs. heures creuses)
- Filtrer par type ou catégorie de service

**Journal d'Activité :**

Toutes les modifications de filtres sont enregistrées dans le fil d'activité du client.

# Sessions Actives

Voir les sessions actives en temps réel pour ce service. Cela montre les appels en cours, les sessions de données ou d'autres événements facturables actuellement en cours.

**Voir les Sessions Actives :**

1. Ouvrez la vue d'édition du service
2. Développez la section **"Sessions Actives"**
3. Voir la liste des sessions actives
4. Cliquez sur **"Voir Détails"** sur n'importe quelle session pour voir les données complètes de la session
5. Cliquez sur **"Rafraîchir"** pour recharger la liste des sessions

**Informations de Session Affichées :**

- **Temps de Configuration** - Quand la session a commencé
- **Utilisation** - Durée actuelle de la session (en secondes)
- **Destination** - Numéro appelé ou destination

**Fenêtre Modale des Détails de Session :**

Cliquer sur "Voir Détails" ouvre une fenêtre modale affichant :

- **Informations de Base :**
  - CGRID (ID de session)
  - Compte (UUID du service)
  - Temps de configuration
  - Utilisation/durée actuelle
  - Destination
  - Catégorie
- **Données Complètes de la Session :**
  - Représentation JSON complète de la session
  - Tous les champs de session CGRateS
  - État de session en temps réel
  - Visionneuse JSON défilable pour inspection

**Cas d'Utilisation :**

- Surveiller les appels actifs ou les sessions de données
- Résoudre des problèmes de facturation

- Vérifier que la session est correctement tarifée
- Vérifier les attributs et valeurs de session
- Auditer l'utilisation active du service

**Taux de Rafraîchissement :**

Les sessions sont récupérées à la demande lorsque vous développez la section. Cliquez sur "Rafraîchir" pour obtenir les dernières données de session.

> **REMARQUE**
>
> Seules les sessions correspondant au compte de ce service (UUID du service) sont affichées. Le filtre `*string:~*req.Account:{service_uuid}` est automatiquement appliqué.

# Proxy API CGRateS

Toutes les opérations CGRateS (attributs, filtres, sessions) utilisent le point de terminaison proxy API OmniCRM :

**Point de terminaison :** `POST /crm/ocs/proxy`

**Champs Requis :**

- `method` - Méthode API CGRateS (par exemple, `APIerSv1.GetAttributeProfile`)
- `params` - Tableau de paramètres pour la méthode
- `customer_id` - ID du client (pour l'enregistrement d'activité)
- `service_id` - ID du service (pour l'enregistrement d'activité)

**Champs Optionnels :**

- `tenant` - Locataire CGRateS (par défaut à la valeur de configuration)

**Exemple de Requête :**

```
{
  "method": "APIerSv1.GetAttributeProfile",
  "params": [{"ID": "ATTR_ACCOUNT_Mobile_SIM_c2880638"}],
  "customer_id": 123,
  "service_id": 456
}
```

**Configuration du Locataire :**

Le locataire est automatiquement défini à partir du fichier de configuration OmniCRM (`crm_config.yaml`) sous `ocs.ocsTenant`. Cela garantit que toutes les opérations CGRateS utilisent le bon locataire sans coder en dur les valeurs dans le frontend.

**Exigence de Permission :**

La permission `cgrates_api_access` est requise. Cette permission est accordée au rôle `admin` par défaut.

**Journal d'Activité :**

Toutes les opérations API CGRateS non-GET sont automatiquement enregistrées dans le fil d'activité du client, y compris :

- Méthode API appelée
- Locataire utilisé
- Tous les paramètres envoyés
- ID du service sur lequel l'opération a été effectuée
- Utilisateur ayant effectué l'opération
- Horodatage

Cela crée une piste d'audit complète de tous les changements de configuration CGRateS.

# Dépannage de l'Intégration CGRateS

**"Permission Refusée" lors de l'accès aux fonctionnalités CGRateS**

- **Cause :** L'utilisateur n'a pas la permission `cgrates_api_access`
- **Solution :** Accorder la permission au rôle de l'utilisateur (fonctionnalité généralement réservée aux administrateurs)

**Attributs ou Filtres ne se chargeant pas**

- **Cause :** Problème de connectivité CGRateS ou le profil n'existe pas
- **Solution :**
  - Vérifiez la connectivité du serveur CGRateS dans la configuration
  - Vérifiez que la configuration du locataire est correcte
  - Vérifiez la console du navigateur pour des erreurs API
  - Le profil peut ne pas encore exister (affichera un formulaire vide)

**Modifications non enregistrées**

- **Cause :** Erreur de validation ou erreur API CGRateS
- **Solution :**
  - Vérifiez les champs requis (Path, Type, Element)
  - Vérifiez que le format JSON est correct
  - Vérifiez le journal d'activité pour des détails d'erreur
  - Examinez les journaux CGRateS

**Aucune session active affichée**

- **Cause :** Aucune session actuellement active pour ce service
- **Solution :**
  - Cela est normal si le service n'est pas utilis��
  - Essayez de rafraîchir après avoir initié une session (appel, données, etc.)
  - Vérifiez que l'UUID du service correspond au compte dans CGRateS

**Détails de session ne se mettant pas à jour en temps réel**

- **Cause :** Les données de session sont récupérées à la demande, pas en direct
- **Solution :** Cliquez sur le bouton "Rafraîchir" pour obtenir les dernières données de session

**Journal d'activité n'affichant pas les changements CGRateS**

- **Cause :** Seules les opérations non-GET sont enregistrées (les lectures ne sont pas enregistrées)
- **Solution :** Cela est intentionnel - seules les écritures/modifications créent des entrées d'activité

# Flux de Travail Courants

## Flux de Travail 1 : Trouver les Services d'un Client

1. Cliquez sur le **filtre de Type de Service** (optionnel)
2. Cliquez sur le **filtre de Client**
3. Tapez le nom du client
4. Sélectionnez le client dans le menu déroulant
5. Examinez les services du client

## Flux de Travail 2 : Identifier les Services Expirants

1. Cliquez sur l'onglet **"Actif"**
2. Triez par **"Date de Fin"** (ascendant)
3. Les services expirant le plus tôt apparaissent en premier
4. Contactez les clients pour le renouvellement

## Flux de Travail 3 : Nettoyer les Services de Test

1. Recherchez "test" dans la boîte de recherche
2. Examinez les résultats pour confirmer qu'il s'agit de données de test
3. Sélectionnez tous les services de test
4. Cliquez sur **"Supprimer Sélectionnés"**
5. Confirmez la suppression

## Flux de Travail 4 : Suspendre un Client Non-Payant

1. Naviguez vers le compte client
2. Affichez l'onglet des services
3. Sélectionnez tous les services actifs
4. Changez le statut en "Suspendu"
5. Enregistrez les modifications

## Flux de Travail 5 : Voir l'Utilisation du Service Mobile

1. Filtrez par **Type de Service : Mobile**
2. Cliquez sur le nom du service pour ouvrir les détails
3. Cliquez sur l'onglet **"Utilisation"**
4. Examinez la consommation de données/voix/SMS
5. Identifiez les utilisateurs lourds ou les dépassements

# Dépannage

### Services n'apparaissant pas dans la liste

- **Cause :** Filtre de statut cachant les résultats
- **Solution :** Cliquez sur l'onglet "Tous les Résultats" pour afficher tous les statuts

### Impossible de trouver le service par recherche

- **Cause :** Le terme de recherche ne correspond pas aux données stockées
- **Solution :**
  - Essayez une recherche partielle (par exemple, "0770" au lieu du numéro complet)
  - Utilisez le filtre de client à la place
  - Vérifiez les fautes de frappe

**Bouton de suppression en masse désactivé**

- **Cause :** Aucun service sélectionné
- **Solution :** Cochez les cases à côté des services que vous souhaitez supprimer

**Le tri ne fonctionne pas**

- **Cause :** Colonne non triable
- **Solution :** Seules les colonnes avec des icônes ▲▼ sont triables

**La page se charge lentement**

- **Cause :** Trop de services à afficher
- **Solution :**
  - Appliquez des filtres pour réduire l'ensemble des résultats
  - Réduisez le nombre d'éléments par page
  - Utilisez la recherche pour affiner les résultats

# Documentation Connexe

- `csa_add_service` - Ajout de nouveaux services
- `csa_modify` - Modification des services et ajout d'options supplémentaires
- `concepts_products_and_services` - Concepts de produits et de services
- `basics_customers` - Gestion des clients

# Suivi de l'utilisation du service et des soldes

Le système de suivi de l'utilisation du service fournit une **surveillance en temps réel** de la consommation des clients pour les données, la voix, les SMS et les soldes monétaires. Cette fonctionnalité s'intègre avec l'OCS (Système de Facturation en Ligne) pour afficher l'utilisation actuelle, les allocations restantes et les informations sur l'expiration des soldes aux clients (via le Portail d'Auto-Soins) et au personnel.

# Aperçu

Le suivi de l'utilisation permet :

- **Affichage en temps réel du solde** - Voir l'utilisation actuelle et les allocations restantes
- **Types de soldes multiples** - Suivre simultanément les soldes de données, de voix, de SMS et monétaires
- **Surveillance des expirations** - Voir quand les soldes expirent
- **Détail des soldes** - Vue détaillée des seaux de solde individuels
- **Actualisation automatique** - Mises à jour de l'utilisation toutes les 3 secondes automatiquement

# Accéder à l'utilisation du service

**Depuis la liste des services :**

1. Naviguez vers **Services → Liste des services**
2. Cliquez sur le **menu d'actions ( ⋮ )** à côté d'un service
3. Sélectionnez **"Utilisation"**

**Depuis les détails du service :**

1. Ouvrez la page de détails d'un service
2. Cliquez sur l'onglet **"Utilisation"**

**Depuis la page client :**

1. Ouvrez l'aperçu du client
2. Naviguez vers l'onglet **Services**
3. Cliquez sur **"Voir l'utilisation"** à côté de n'importe quel service

La modal ou la page d'utilisation s'ouvre en affichant les données de consommation en temps réel. Pour inspecter les appels individuels, les SMS et les sessions de données derrière ces soldes, passez à l'onglet **Registres d'appels (CDRs)**. Voir Registres de Détails des Appels.

# Affichage de l'utilisation

L'interface d'utilisation montre des cartes de résumé et des barres de progression détaillées pour chaque type de solde.

## Cartes de résumé

La rangée supérieure affiche des cartes de vue rapide pour chaque type de solde :

**Informations sur la carte :**

- **Type de solde** - Icône et étiquette (Données, Voix, SMS, Monétaire)
- **Montant restant** - Solde actuel dans les unités appropriées
- **Temps d'expiration** - Jours/heures jusqu'à l'expiration du solde
- **Bouton Plus d'info** - Cliquez pour développer le détail

## Barres de progression

Sous les cartes, les barres de progression montrent la consommation visuellement avec des portions remplies indiquant le solde restant.

**Fonctionnalités de la barre de progression :**

- **Indicateur visuel** - La portion remplie montre le solde restant
- **Pourcentage** - Pourcentage numérique du solde restant
- **Valeurs absolues** - Montre utilisé vs total (par exemple, "12.5GB / 20GB")
- **Codage couleur** :
  - Vert : >50% restant
  - Jaune : 20-50% restant

    - Rouge : <20% restant
- **Cliquable** - Cliquez pour développer le détail

# Types de soldes

## Solde de données

Suit la consommation de données Internet.

**Unités :** Gigaoctets (Go) ou Mégaoctets (Mo)

**Format d'affichage :**

> Progression : 12.5Go / 20Go (62%)

**Scénarios courants :**

- **Forfaits de données mobiles** - 5Go, 10Go, 20Go d'allocations mensuelles
- **Sans fil fixe** - Illimité ou plafonné à des montants élevés (500Go, 1To)
- **Recharges** - Données supplémentaires achetées en cours de cycle
- **Services de dongle** - Données prépayées pour appareils hotspot

**Seaux multiples :**

Les services ont souvent plusieurs soldes de données :

- Allocation mensuelle (expire mensuellement)
- Données bonus (expire après la période de campagne)
- Données de recharge (expiration plus courte, consommées en premier)

## Solde de voix

Suit les minutes d'appels téléphoniques.

**Unités :** Minutes (min)

**Format d'affichage :**

> Progression : 125 min / 500 min (25%)

**Suivi de la durée des appels :**

- Appels entrants (s'ils sont facturés)
- Appels sortants
- Appels internationaux (seau séparé si applicable)
- Numéros premium

**Calcul :**

L'utilisation de la voix est calculée par la durée des appels en nanosecondes en interne, convertie en minutes pour l'affichage.

# Solde de SMS

Suit l'utilisation des messages texte.

**Unités :** Messages (msgs)

**Format d'affichage :**

> Progression : 45 / 250 (18%)

**Types de messages :**

- SMS standard (160 caractères)
- Long SMS (plusieurs segments)
- MMS (s'il est suivi séparément)

# Solde monétaire

Suit le crédit prépayé ou le solde du compte.

**Unités :** Monnaie (£, $, €, etc.)

**Format d'affichage :**

> Progression : £15.50 / £20.00 (77%)

**Utilisation :**

- Les comptes prépayés utilisent le solde monétaire pour payer l'utilisation
- Le crédit diminue à mesure que le client utilise des services
- Peut être rechargé via paiement ou bon
- Peut expirer s'il n'est pas utilisé dans la période de validité

# Détail du solde

Cliquez sur **"Plus d'info"** sur n'importe quelle carte ou cliquez sur une barre de progression pour développer le détail.

**Vue développée :**

> □ Allocation mensuelle 20Go
> Restant : 12.5 Go Utilisé : 7.5 Go Expire : 25 Jan 2025 (15 jours) Poids : 10
>
> □ Données bonus 5Go
> Restant : 5.0 Go Utilisé : 0 Go Expire : 31 Jan 2025 (21 jours) Poids : 20
>
> □ Données de recharge 3Go
> Restant : 0 Go Utilisé : 3.0 Go Expire : 18 Jan 2025 (EXPIRE) Poids : 30
>
> Total restant : 17.5 Go

**Champs du seau de solde :**

- **ID/Nom** - Identifiant pour le seau de solde
- **Restant** - Montant restant dans ce seau spécifique
- **Utilisé** - Montant consommé de ce seau
- **Date d'expiration** - Quand ce solde expire
- **Poids** - Ordre de priorité (poids plus élevé consommé en premier)

## Système de poids

Les soldes ont une valeur de **poids** qui détermine l'ordre de consommation :

- **Poids plus élevé = consommé en premier**

- **Poids plus bas = consommé en dernier**

**Exemples de poids :**

- Données de recharge : Poids 30 (consommé en premier, expiration plus courte)
- Données bonus : Poids 20 (consommé en second)
- Allocation mensuelle : Poids 10 (consommé en dernier, expiration la plus longue)

Cela garantit que les soldes expirants sont utilisés avant ceux qui durent plus longtemps.

Pour des détails complets sur les règles de consommation des soldes, les stratégies de poids et comment CGRateS priorise les soldes, voir Actions et Comportements de Recharge CGRateS.

# Mises à jour en temps réel

Les données d'utilisation se rafraîchissent automatiquement toutes les **3 secondes** via polling.

**Qu'est-ce qui se met à jour :**

- Montants de solde actuels
- Barres de progression d'utilisation
- Minuteurs d'expiration
- Détails des seaux individuels

**Expérience utilisateur :**

- Aucun rechargement de page requis
- Mises à jour fluides sans scintillement
- Superposition de chargement pendant le rafraîchissement
- Badge d'état montre l'état actuel du service

**Cas d'utilisation :**

- Surveiller l'utilisation du client pendant l'appel
- Observer la diminution du solde en temps réel à mesure que le client utilise le service
- Vérifier la recharge immédiatement après l'achat

# Utilisation dans différents types de services

## Services mobiles

Affiche les quatre types de solde :

- Données (Go)
- Voix (minutes)
- SMS (messages)
- Monétaire (monnaie)

**Exemple :**

> DONNÉES : 12.5Go restant VOIX : 125 min restant SMS : 45 msgs restant MONÉTAIRE : £15.50 restant

## Sans fil fixe / Internet

Affiche généralement uniquement :

- Données (Go ou To)
- Monétaire (si prépayé)

**Exemple :**

> DONNÉES : 450Go / 500Go restant MONÉTAIRE : £45.00 (crédit prépayé)

## Services de point d'accès / Dongle

Montre le suivi des données spécifiques au dongle :

- Données (consommées vs. prépayées)
- Monétaire (solde prépayé)

**Mode d'affichage :**

Lorsque `dongle=true`, le composant cache la voix et les SMS, affichant uniquement les soldes de données et monétaires pertinents.

# Dépannage

**Utilisation affichée comme 0 / 0**

- **Cause :** Service non intégré avec OCS ou CGRateS
- **Solution :**
  - Vérifiez que le service est provisionné dans l'OCS
  - Vérifiez la connectivité de l'API OCS
  - Examinez le mappage UUID du service

**Utilisation non mise à jour**

- **Cause :** Polling arrêté ou OCS inaccessible
- **Solution :**
  - Rafraîchissez la page
  - Vérifiez la console du navigateur pour des erreurs
  - Vérifiez que l'API OCS est en ligne

**Les soldes affichent des montants incorrects**

- **Cause :** Mismatch de données OCS ou problème de mise en cache
- **Solution :**
  - Forcez le rafraîchissement du solde OCS
  - Vérifiez les transactions en attente
  - Vérifiez la configuration OCS

**Dates d'expiration manquantes**

- **Cause :** Le solde n'a pas d'expiration définie

- **Solution :**
  - Certains soldes sont configurés pour ne jamais expirer (validité illimitée)
  - Vérifiez la configuration du solde dans l'OCS

**Soldes multiples déroutants**

- **Cause :** Plusieurs recharges ou données bonus ajoutées
- **Solution :**
  - Utilisez la vue détaillée pour voir tous les seaux
  - Triez par poids pour voir l'ordre de consommation
  - Examinez les dates d'expiration individuelles

# Intégration avec OCS/CGRateS

Les données d'utilisation proviennent de l'**OCS (Système de Facturation en Ligne)**, généralement CGRateS. Pour des informations complètes sur le fonctionnement des soldes, y compris les types de soldes, les règles de consommation, le comportement de report et les configurations d'Action, voir Actions et Comportements de Recharge CGRateS.

**Flux de données :**

1. L'utilisateur ouvre la vue d'utilisation
2. OmniCRM appelle `GET /crm/service/{service_id}`
3. L'API interroge l'OCS via l'UUID du service
4. L'OCS renvoie la carte des soldes :

```
{
  "BalanceMap": {
    "*data": [
      {
        "ID": "monthly_data_20GB",
        "Value": 13421772800,
        "ExpiryTime": "2025-01-25T23:59:59Z",
        "Weight": 10
      }
    ],
    "*voice": [
      {
        "ID": "monthly_voice_500min",
        "Value": 7500000000000,
        "ExpiryTime": "2025-01-25T23:59:59Z",
        "Weight": 10
      }
    ],
    "*sms": [
      {
        "ID": "monthly_sms_250",
        "Value": 250,
        "ExpiryTime": "2025-01-25T23:59:59Z",
        "Weight": 10
      }
    ],
    "*monetary": [
      {
        "ID": "prepaid_credit",
        "Value": 1550,
        "ExpiryTime": "2025-02-25T23:59:59Z",
        "Weight": 10
      }
    ]
  }
}
```


5. L'UI convertit les valeurs en unités d'affichage (octets → Go, nanosecondes → minutes)
6. Les barres de progression et les cartes sont rendues
7. Le polling continue toutes les 3 secondes

**Mapping des types de solde OCS :**

L'OCS renvoie les données de solde avec des préfixes de type qui se mappent à l'affichage de l'UI :

- `*data` → **CARTE DE DONNÉES** (utilisation d'Internet)
- `*voice` → **CARTE DE VOIX** (minutes d'appel)
- `*sms` → **CARTE SMS** (messages texte)
- `*monetary` → **CARTE MONÉTAIRE** (crédit prépayé)

Chaque type de solde peut avoir plusieurs seaux (par exemple, allocation mensuelle + données bonus + données de recharge), tous affichés dans la vue détaillée.

**Conversions de valeurs de solde :**

- **Données :** Octets → Go (diviser par $1024^3$)
- **Voix :** Nanosecondes → Minutes (diviser par $60 \times 10^9$)
- **SMS :** Compte (pas de conversion)
- **Monétaire :** Cents → Monnaie (diviser par 100)

# Auto-Renouvellement et Plans d'Action

Les services avec renouvellement automatique activé ont des **Plans d'Action** programmés dans l'OCS. Pour des informations détaillées sur le fonctionnement des Actions, la manipulation des soldes et la configuration des Plans d'Action, voir Actions et Comportements de Recharge CGRateS.

**Qu'est-ce que les Plans d'Action ?**

Les Plans d'Action sont des tâches programmées dans CGRateS qui s'exécutent automatiquement à des moments spécifiques pour :

- Ajouter du solde à un compte (recharge automatique)
- Renouveler les allocations mensuelles
- Appliquer des frais récurrents
- Expirer les anciens soldes

**Comment fonctionne l'Auto-Renouvellement :**

1. **Provisionnement du service :**
   - Lorsque le service est créé avec `auto_renew = true`
   - Le playbook de provisionnement crée un Plan d'Action dans l'OCS
   - Plan d'Action configuré pour s'exécuter mensuellement (ou par cycle de facturation)
2. **Configuration du Plan d'Action :**

   Le Plan d'Action contient :

   - **ID de compte** - UUID du service
   - **Actions** - Que faire (ajouter des données, de la voix, des SMS, un solde monétaire)
   - **Calendrier** - Quand exécuter (par exemple, mensuellement le 1er à 00:00 UTC)
   - **Montant** - Combien de solde ajouter
3. **Exécution automatique :**
   - L'OCS exécute le Plan d'Action à l'heure programmée
   - Ajoute du solde au compte (par exemple, 20Go de données, 500 minutes de voix)
   - Définit la date d'expiration pour le nouveau solde (par exemple, 30 jours)
   - Le client est facturé via le mode de paiement enregistré
4. **Visualisation des Plans d'Action :**
   - Naviguez vers les détails du service dans la vue OCS
   - Plans d'Action listés avec l'heure d'exécution suivante
   - Montre : Nom du plan, date de prochaine exécution, détails de l'action

**Exemple de Plan d'Action :**

```
{
  "ActionPlanId":
"ProductID_MonthlyPlan__ProductName_20GB_Mobile__ActionPlan_Monthly_F
  "NextExecTime": "2025-02-01T00:00:00+00:00",
  "ActionName_hr": "Renouvellement Mensuel",
  "PlanName": "20Go Mobile",
  "ActionFrequency_hr": "Chaque Plan Mensuel",
  "custom_NextExecTime_hr": "dans 22 jours"
}
```


**Gestion de l'Auto-Renouvellement :**

- **Activer** - Définir lors de la création ou de la modification du service
- **Désactiver** - Supprimer le Plan d'Action de l'OCS (le service conserve le solde existant mais ne sera pas renouvelé automatiquement)
- **Modifier** - Changer le montant ou la fréquence de renouvellement via la modification du service

**Renouvellement manuel :**

Si le renouvellement automatique est désactivé, le client doit manuellement :

- Recharger avant l'expiration du solde
- Ou le service est suspendu lorsque le solde est épuisé

**Visualisation dans l'UI :**

L'onglet des services montre l'état du renouvellement automatique :

> Prochain renouvellement : 1er Février 2025 (dans 22 jours) Montant de renouvellement : £15.00

# Meilleures pratiques

**Pour le personnel de support :**

- Vérifiez l'utilisation avant de répondre aux appels "Pourquoi mon service est-il lent ?"

- Vérifiez le solde après les recharges pour confirmer le succès
- Utilisez le détail pour identifier les seaux expirés
- Surveillez les clients à forte consommation pour éviter les dépassements

**Pour les clients (Auto-Soins) :**

- Vérifiez régulièrement l'utilisation pour éviter de manquer
- Rechargez avant l'expiration du solde
- Comprenez le système de poids pour savoir quel solde est consommé en premier
- Contactez le support si l'utilisation semble incorrecte

**Pour les administrateurs :**

- Configurez des expirations de solde appropriées
- Définissez des valeurs de poids pour prioriser les soldes expirants
- Surveillez la connectivité OCS pour un reporting précis
- Vérifiez que les configurations de solde correspondent aux offres de produits

# Documentation connexe

- Actions et Comportements de Recharge CGRateS - Guide complet sur les types de soldes, les règles de consommation, les poids, le comportement de report et les configurations d'Action
- Destinations CGRateS - Configuration des destinations pour les restrictions de solde géographiques et basées sur le PLMN
- `features_topup_recharge` - Système de recharge pour ajouter du solde
- `csa_service_management` - Gestion des services
- `csa_modify` - Ajout d'options pour augmenter les allocations
- `concepts_products_and_services` - Configuration des produits

# Service Client

## Usurpation d'Identité pour le Support et le Dépannage

L'usurpation d'identité permet au personnel autorisé de se connecter temporairement en tant qu'autre utilisateur. Le personnel peut résoudre des problèmes, vérifier des configurations ou voir exactement ce que l'utilisateur expérimente. Cette fonctionnalité est essentielle pour le support client. Elle nécessite des autorisations appropriées et est entièrement audité.

Lorsque vous usurpez l'identité d'un client, vous accédez au Portail d'Auto-Service exactement comme le client le voit. Cela permet un dépannage et un support précis.

Voir aussi : RBAC pour la configuration des autorisations, Clients pour la gestion des clients, Portail d'Auto-Service pour les fonctionnalités du portail client.

# Objectif

L'usurpation d'identité fournit :

1. **Dépannage** : Voir exactement ce que le client voit pour diagnostiquer des problèmes
2. **Vérification** : Confirmer que les configurations de service et les autorisations fonctionnent correctement
3. **Formation** : Démontrer les fonctionnalités du point de vue du client
4. **Support** : Aider les clients à naviguer dans le système sans partage d'écran
5. **Trace d'Audit** : Le système enregistre toutes les sessions d'usurpation d'identité pour la sécurité et la conformité

# Autorisations Requises

Pour usurper l'identité des utilisateurs, vous devez avoir l'une des autorisations suivantes :

- `can_impersonate` : Autorisation d'usurpation dédiée pour le personnel de support
- `admin` : Accès administratif complet (inclut les droits d'usurpation)

Les utilisateurs sans ces autorisations ne peuvent pas accéder à la fonctionnalité d'usurpation.

# Comment Usurper l'Identité d'un Utilisateur

**Via l'Interface Web :**

1. **Naviguer vers le Client** : Trouver le client dans le CRM
2. **Sélectionner les Contacts** : Voir la liste de contacts du client
3. **Cliquer sur "Se Connecter en tant qu'Utilisateur"** : Le bouton apparaît à côté de chaque contact qui a un compte utilisateur
4. **Confirmer l'Usurpation** : Le système peut demander une confirmation
5. **La Session Commence** : Vous êtes maintenant connecté en tant que cet utilisateur

**Via l'API :**

Démarrer la session d'usurpation :

**Point de terminaison :** `POST /auth/impersonate`

**Autorisation Requise :** `can_impersonate` ou `admin`

**Demande :**

```
{
  "user_id": 42
}
```

**Réponse :**

```
{
  "success": true,
  "impersonating_user_id": 1,
  "target_user_id": 42,
  "impersonation_start": "2025-01-04T15:30:00Z",
  "access_token": "new_token_for_impersonated_user",
  "refresh_token": "new_refresh_token"
}
```

Les jetons retournés sont pour la session de l'utilisateur usurpé.

# Que Se Passe-t-il Pendant l'Usurpation

Lorsque vous usurpez l'identité d'un utilisateur :

- **Contexte Complet** : Vous voyez le système exactement comme le voit l'utilisateur cible :
  - Leur tableau de bord et navigation
  - Leurs données client (si utilisateur client)
  - Leurs autorisations et contrôles d'accès
  - Leurs services, factures et utilisation
- **Suivi de Session** : Le système suit les deux identités :
  - `impersonating_user_id` : Votre véritable ID utilisateur
  - `target_user_id` : L'utilisateur que vous usurpez
  - `impersonation_start` : Quand l'usurpation a commencé
- **Indicateur** : L'interface utilisateur affiche une bannière indiquant que vous usurpez l'identité de quelqu'un :
  - "Vous êtes actuellement connecté en tant que [Nom d'utilisateur]"
  - "Cliquez ici pour arrêter l'usurpation"
- **Journalisation d'Audit** : Le système enregistre toutes les actions avec les deux ID utilisateurs :
  - Les actions apparaissent comme effectuées par l'utilisateur cible
  - Les journaux d'audit enregistrent qui les a réellement effectuées (utilisateur usurpant)
  - Trace d'audit complète maintenue dans la table `ImpersonationLog`

# Arrêter l'Usurpation

**Via l'Interface Web :**

1. **Cliquer sur la Bannière** : Cliquez sur la bannière d'usurpation en haut de la page

2. **Ou Naviguer** : Allez dans le menu utilisateur et sélectionnez "Arrêter l'Usurpation"
3. **Confirmation** : La session se termine et vous revenez à votre propre compte

**Via l'API :**

**Point de terminaison :** `POST /auth/stop_impersonation`

**Demande :** Aucun corps requis (demande authentifiée)

**Réponse :**

```
{
  "success": true,
  "impersonation_end": "2025-01-04T15:45:00Z",
  "duration_seconds": 900,
  "access_token": "your_original_token",
  "refresh_token": "your_original_refresh_token"
}
```

Votre session originale est restaurée.

# Journalisation d'Audit de l'Usurpation

Toutes les sessions d'usurpation sont enregistrées dans la table `ImpersonationLog` avec :

- **impersonating_user_id** : Membre du personnel qui a effectué l'usurpation
- **target_user_id** : Client ou utilisateur qui a été usurpé
- **impersonation_start** : Horodatage de début
- **impersonation_end** : Horodatage de fin (lorsque la session a été arrêtée)
- **impersonation_duration** : Durée en secondes

Cela fournit une responsabilité complète pour toutes les sessions d'usurpation et permet :

- **Audits de sécurité** : Examiner qui a usurpé qui et quand
- **Rapports de conformité** : Démontrer l'utilisation appropriée des accès élevés
- **Enquête** : Suivre les actions pendant les sessions d'usurpation
- **Surveillance** : Identifier des modèles d'usurpation inhabituels

**Visualisation des Journaux d'Usurpation :**

Les administrateurs peuvent interroger les journaux d'usurpation via l'API :

`GET /auth/impersonation_logs?user_id={user_id}&start_date={date}&end_date={date}`

Filtrer par :

- Utilisateur usurpant (qui a effectué l'usurpation)
- Utilisateur cible (qui a été usurpé)
- Plage de dates
- Durée

# Meilleures Pratiques

1. **Minimiser la Durée** : Usurpez uniquement le temps nécessaire pour résoudre le problème
2. **Documenter l'Objectif** : Notez pourquoi l'usurpation était nécessaire dans le journal d'activité du client
3. **Informer les Clients** : Faites savoir aux clients que vous pourriez avoir besoin de consulter leur compte (politique de confidentialité)
4. **Vérifier l'Identité** : Confirmez l'identité de l'utilisateur avant d'usurper leur compte
5. **Revoir les Journaux** : Auditez régulièrement les journaux d'usurpation pour des modèles inhabituels
6. **Limiter les Autorisations** : Accordez `can_impersonate` uniquement au personnel de support qui en a besoin
7. **Formation** : Assurez-vous que le personnel comprend la responsabilité et les implications d'audit

# Considérations de Sécurité

- **Accès Complet** : L'usurpation accorde un accès complet au compte de l'utilisateur cible
- **Aucun Mot de Passe Requis** : L'usurpation contourne l'authentification (basée uniquement sur les autorisations)
- **Actions Journalisées** : Les journaux d'application attribuent toutes les actions pendant l'usurpation à l'utilisateur cible (mais les journaux d'audit montrent le véritable acteur)
- **Isolation de Session** : L'usurpation crée une nouvelle session. Elle n'affecte pas les sessions actives de l'utilisateur cible
- **Limites de Temps** : Les sessions d'usurpation doivent être limitées dans le temps (configurables)
- **Contournement de la MFA** : L'usurpation contourne les exigences de 2FA (utilise l'authentification de l'usurpateur)

# Restrictions

- **Impossible d'Usurper les Administrateurs** : Selon la configuration, vous ne pourrez peut-être pas usurper d'autres administrateurs
- **Mêmes Autorisations** : Vous obtenez les autorisations de l'utilisateur cible, pas l'union des autorisations des deux utilisateurs
- **Limites de Session** : Une seule session d'usurpation par membre du personnel à la fois
- **Exigences d'Audit** : Vous ne pouvez pas désactiver ou cacher la journalisation d'usurpation

# Dépannage des Problèmes Courants

**Problème : Le bouton "Se Connecter en tant qu'Utilisateur" n'apparaît pas**

- Solution : Vérifiez que vous avez l'autorisation `can_impersonate` ou `admin`

- Solution : Confirmez que le contact a un compte utilisateur lié

**Problème : L'usurpation échoue avec une erreur de permission**

- Solution : Vérifiez si l'utilisateur cible est un administrateur (peut être restreint)
- Solution : Vérifiez que votre autorisation d'usurpation est active

**Problème : Impossible d'arrêter l'usurpation**

- Solution : Utilisez le point de terminaison de l'API pour arrêter l'usurpation : `POST /auth/stop_impersonation`
- Solution : Effacez les cookies du navigateur et reconnectez-vous avec vos identifiants

**Problème : Les actions ne se journalisent pas correctement**

- Solution : Vérifiez que la session d'usurpation est active (vérifiez la bannière)
- Solution : Consultez les journaux d'audit - les actions sont enregistrées avec les deux ID utilisateurs

# Définir les Produits et la Configuration CGRateS

Ce guide explique comment définir des produits dans OmniCRM ainsi que leur configuration de facturation CGRateS associée.

Les produits et leurs Actions CGRateS sont généralement définis ensemble dans des scripts Python qui configurent à la fois le CRM et le Système de Facturation en Ligne (OCS).

# Aperçu

Définir un produit complet implique une configuration dans **deux systèmes** :

1. **CGRateS/OCS** - Définir le comportement de facturation (Actions, Tarifs, Destinations)
2. **OmniCRM** - Définir les métadonnées du produit (prix, nom, caractéristiques)

Ces systèmes sont intentionnellement liés de manière lâche, vous permettant d'interagir avec CGRateS via les Ansible Plays que vous définissez.

## Important : Capacités de CGRateS

**CGRateS est un moteur de tarification extrêmement puissant et flexible.** Ce guide montre **un modèle de configuration courant** pour les produits OmniCRM, mais CGRateS peut faire beaucoup plus :

- **Scénarios de tarification avancés** : Tarifs selon l'heure de la journée, niveaux de clients, routage basé sur la qualité des appels, facturation par pic

- **Logique de roaming complexe** : Tarifs différents pour les appels/SMS MO vs MT lors du roaming, tarification spécifique au réseau
- **Support des protocoles hérités** : CAMEL pour le roaming 2G/3G (SMS MO-MT, USSD), Diameter pour LTE/5G
- **Facturation sophistiquée** : Facturation basée sur la réservation, contrôle de crédit, fallback multi-niveaux
- **Routage dynamique** : Routage basé sur le coût, sélection de fournisseur, LCR (Routage à Moindre Coût)

**Ceci est juste le guide CRM** - il se concentre sur des modèles de définition de produits simples qui fonctionnent bien avec le système de provisionnement d'OmniCRM. Pour des configurations CGRateS avancées, consultez la documentation CGRateS ou voyez Actions CGRateS et Comportements de Recharge pour des approches de solde et des stratégies de tarification.

# Flux de Travail Complet pour la Définition de Produits

## Étape 1 : Authentifier à l'API CRM

```
import requests
import json

crm_url = 'https://crm.example.com/'
session = requests.Session()

headers = {
    "Content-Type": "application/json"
}

# Obtenir le jeton d'authentification
response = session.post(crm_url + '/crm/auth/login', json={
    "email": "admin@example.com",
    "password": "your_password"
}, headers=headers)

assert response.status_code == 200
headers['Authorization'] = 'Bearer ' + response.json()['token']
print("Auth to CRM OK")
```

## Étape 2 : Définir les Modèles d'Inventaire (Optionnel)

Les modèles d'inventaire définissent les types d'articles physiques (cartes SIM, routeurs, etc.) qui peuvent être attribués lors du provisionnement.

```
inventory_list_new = []

# Définir le modèle d'inventaire de carte SIM
inventory_list_new.append({
    "item": "SIM Card",
    "itemtext1_label": "ICCID",
    "itemtext2_label": "IMSI",
    "wholesale_cost": 0.2,
    "retail_cost": 1,
    "allow_dropdown_staff": True,
    "allow_dropdown_customer": True,
    "icon": "fa-solid fa-sim-card"
})

# Définir le modèle d'inventaire de numéro mobile
inventory_list_new.append({
    "item": "Mobile Number",
    "itemtext1_label": "E.164 Mobile Number",
    "itemtext2_label": "Type",
    "wholesale_cost": 0.2,
    "retail_cost": 3,
    "icon": "fa-solid fa-phone"
})

# Vérifier si le modèle d'inventaire existe, créer ou mettre à
jour
for inventory_template in inventory_list_new:
    # Obtenir les modèles existants
    inventory_list_existing = session.get(
        crm_url + '/crm/inventory/item_template/',
        headers=headers
    )

    # Vérifier si le modèle existe déjà
    if inventory_template['item'] in [x['item'] for x in
inventory_list_existing.json()]:
        # Mettre à jour le modèle existant
        inventory_template_id = [
            x['inventory_template_id']
            for x in inventory_list_existing.json()
            if x['item'] == inventory_template['item']
        ][0]
```

```
        response = session.patch(
            crm_url + '/crm/inventory/item_template/' +
str(inventory_template_id),
            json=inventory_template,
            headers=headers
        )
        assert response.status_code == 200
        print(f"Modèle d'inventaire mis à jour :
{inventory_template['item']}")
    else:
        # Créer un nouveau modèle
        response = session.put(
            crm_url + '/crm/inventory/item_template',
            json=inventory_template,
            headers=headers
        )
        assert response.status_code == 200
        print(f"Modèle d'inventaire créé :
{inventory_template['item']}")
```

**Champs du Modèle d'Inventaire :**

- **item** (requis) - Nom du modèle
- **itemtext1_label** - Étiquette pour le premier champ personnalisé
- **itemtext2_label** - Étiquette pour le deuxième champ personnalisé
- **wholesale_cost** - Votre coût pour cet article
- **retail_cost** - Coût pour le client
- **allow_dropdown_staff** - Afficher dans les menus déroulants du personnel
- **allow_dropdown_customer** - Afficher dans les menus déroulants des clients
- **icon** - Classe d'icône Font Awesome

## Étape 3 : Se Connecter à CGRateS

```
import cgrateshttpapi
import time

OCS_Obj = cgrateshttpapi.CGRateS("ocs.example.com", "2080")

tenant = "your_tenant_name"
tpid = str(tenant) + "_" + str(int(time.time()))
```

## Étape 4 : Définir les Destinations dans CGRateS

**Les Destinations** définissent OÙ les soldes peuvent être utilisés. Elles doivent être définies avant de créer des Actions qui les référencent.

Il existe deux types :

1. **Destinations Géographiques** - Préfixes de numéros pour la voix/SMS VERS des endroits spécifiques (par exemple, `Dest_AU_Mobile`, `Dest_International_UK`)
2. **Destinations PLMN** - Codes de réseau pour l'utilisation des données et le roaming (par exemple, `Dest_PLMN_OnNet`, `Dest_PLMN_US_Verizon`)

**Règle Critique :**

- **Soldes Voix/SMS** → Utilisez des destinations géographiques (le numéro appelé VERS)
- **Soldes de données** → Utilisez des destinations PLMN (le réseau auquel le client est connecté DE)

Pour des exemples complets de définition de destinations incluant :

- Destinations géographiques (mobile, ligne fixe, sans frais, international)
- Destinations PLMN (sur le réseau, roaming, zones)
- Règles de format PLMN (`mccXXX.mncYYY`)
- Quand utiliser chaque approche

Voir : Actions CGRateS - Définir des Destinations

# Étape 5 : Définir les Actions CGRateS

**Les Actions** sont le mécanisme pour ajouter des soldes aux comptes clients. Avant de pouvoir lier un produit CRM à une Action, l'Action doit être définie dans CGRateS.

**Critique :** Les Actions sont définies lors de la configuration initiale du système (pas lors du provisionnement). Elles spécifient quels soldes ajouter, les temps d'expiration, le comportement de rollover, etc.

Voir Actions CGRateS et Comportements de Recharge pour la documentation complète sur :

- Comment définir des Actions en utilisant Python (`OCS_Obj.SendData`)
- Comprendre `*topup` vs `*topup_reset` (comportement de rollover)
- Calculs d'unités (octets pour les données, nanosecondes pour la voix, compte pour les SMS)
- Référence de champ (BalanceId, BalanceType, DestinationIDs, Units, ExpiryTime, Weight)
- Exemples de travail complets (plans multi-soldes, addons de données, addons de voix)
- Règles et priorités de consommation de solde

**Référence Rapide sur le Rollover :**

- **`*topup`** - Ajoute au solde existant (active le rollover). Pour : addons prépayés, packs de données
- **`*topup_reset`** - Remplace le solde existant (pas de rollover). Pour : plans mensuels avec allocations fixes

# Étape 6 : Créer des Produits dans le CRM

Maintenant que les Actions sont définies dans CGRateS, créez les produits correspondants dans le CRM.

**Exemple : Service SIM Autonome**

```
product_list_new = []

product_list_new.append({
    "category": "standalone",
    "provisioning_play": "play_psim_only",
    "relies_on_list": "",
    "contract_days": 0,
    "retail_cost": 0,
    "retail_setup_cost": 0,
    "product_slug": "Mobile-SIM",
    "product_name": "Mobile SIM Only",
    "comment": "Carte SIM physique à utiliser avec des téléphones
mobiles",
    "provisioning_json_vars": "{\"iccid\" : \"\", \"msisdn\" :
\"\"}",
    "terms": "Doit être activé dans les 6 mois. Tout crédit perdu
si le service n'est pas utilisé pendant 12 mois.",
    "service_type": "mobile",
    "residential": True,
    "business": True,
    "enabled": True,
    "inventory_items_list": "['SIM Card', 'Mobile Number']",
    "icon": "fa-solid fa-sim-card",
    "features_list": "['Numéro de téléphone australien (61xxx)',
'Vitesses les plus rapides', 'Meilleure couverture']",
    "wholesale_cost": 3,
    "wholesale_setup_cost": 1,
    "customer_can_purchase": True
})
```

**Exemple : Addon de Plan Mensuel**

```
product_list_new.append({
    "category": "addon",
    "provisioning_play": "play_topup_charge_then_action",
    "relies_on_list": "",
    "contract_days": 30,
    "retail_cost": 59.00,
    "retail_setup_cost": 0,
    "product_slug": "au-premium-plan-1",  # Lien vers Action_au-premium-plan-1
    "product_name": "AU Premium Plan 1",
    "comment": "100 Go de données en Australie, 3000 minutes en Australie, 3000 SMS en Australie, 6 Go de données en roaming",
    "provisioning_json_vars": "",
    "terms": "Plan postpayé. Renouvellement automatique tous les 30 jours.",
    "service_type": "mobile",
    "residential": True,
    "business": True,
    "enabled": True,
    "icon": "fa-solid fa-mobile",
    "features_list": "['100 Go de données en Australie', '3000 minutes en Australie', '3000 SMS en Australie', '6 Go de données en roaming']",
    "wholesale_cost": 45.00,
    "wholesale_setup_cost": 0,
    "auto_renew": "True",
    "customer_can_purchase": True
})
```

**Exemple : Addon de Recharge Unique**

```
product_list_new.append({
    "category": "addon",
    "provisioning_play": "play_topup_charge_then_action",
    "relies_on_list": "",
    "contract_days": 0,
    "retail_cost": 30,
    "retail_setup_cost": 0,
    "product_slug": "au-data-addon-20gb",  # Lien vers Action_au-data-addon-20gb
    "product_name": "AU Data Addon 20GB",
    "comment": "20 Go de données en Australie",
    "provisioning_json_vars": "",
    "terms": "Recharge prépayée. Charge immédiate.",
    "service_type": "mobile",
    "residential": True,
    "business": True,
    "enabled": True,
    "icon": "fa-solid fa-mobile",
    "features_list": "['20 Go de données en Australie']",
    "wholesale_cost": 22,
    "wholesale_setup_cost": 0,
    "auto_renew": "False",
    "customer_can_purchase": True
})
```

**Descriptions des Champs de Produit :**

- **category** (requis) - Catégorie du produit :
  - `standalone` - Service complet (crée un nouvel enregistrement de service)
  - `addon` - Ajouté à un service existant
  - `bundle` - Pack de plusieurs services
- **provisioning_play** (requis) - Playbook Ansible à exécuter (par exemple, `play_topup_charge_then_action`, `play_simple_service`)
- **relies_on_list** - Liste séparée par des virgules des product_slugs dont ce produit dépend
- **contract_days** - Durée du contrat en jours (0 = pas de contrat, 30 = mensuel)
- **retail_cost** (requis) - Coût mensuel/récurrent pour le client

- **retail_setup_cost** - Frais d'installation unique facturés au client
- **product_slug** (requis) - **Identifiant unique qui lie à l'Action CGRateS** (convention : `Action_{product_slug}`)
- **product_name** (requis) - Nom d'affichage
- **comment** - Brève description
- **provisioning_json_vars** - Chaîne JSON de variables passées au playbook
- **terms** - Texte des termes et conditions
- **service_type** - Classification du service (mobile, dongle, fixe, etc.)
- **residential** - Disponible pour les clients résidentiels
- **business** - Disponible pour les clients professionnels
- **enabled** - Le produit est actif et achetable
- **inventory_items_list** - Chaîne de liste Python des articles d'inventaire requis
- **icon** - Classe d'icône Font Awesome
- **features_list** - Chaîne de liste Python des caractéristiques
- **wholesale_cost** - Votre coût récurrent
- **wholesale_setup_cost** - Votre coût unique
- **auto_renew** - Chaîne "True" ou "False" pour le renouvellement automatique
- **customer_can_purchase** - Le client peut acheter via le portail en libre-service

Voir aussi : Cycle de Vie du Produit

# Étape 7 : Créer ou Mettre à Jour des Produits

## via l'API

```
# Obtenir les produits existants
product_list_existing = session.get(crm_url + '/crm/product',
headers=headers)
existing_products = product_list_existing.json()

# Traiter chaque produit
for product in product_list_new:
    print(f"Traitement du produit : {product['product_slug']}")

    # Vérifier si le produit existe déjà
    if product['product_slug'] in [x['product_slug'] for x in
existing_products]:
        print(f"Le produit existe déjà :
{product['product_slug']}")

        # Obtenir product_id du produit existant
        product_id = [
            x['product_id']
            for x in existing_products
            if x['product_slug'] == product['product_slug']
        ][0]

        product['product_id'] = product_id

        # Mettre à jour le produit existant
        response = session.patch(
            crm_url + '/crm/product/' + str(product_id),
            json=product,
            headers=headers
        )
        print(f"Statut : {response.status_code}")
        assert response.status_code == 200
        print(f"Produit mis à jour : {product['product_slug']}")
    else:
        # Créer un nouveau produit
        print(f"Création d'un nouveau produit :
{product['product_slug']}")
        response = session.put(
            crm_url + '/crm/product',
            json=product,
            headers=headers
```

```
        )
        print(f"Statut : {response.status_code}")
        assert response.status_code == 200
        print(f"Produit créé : {product['product_slug']}")
```

# Lier les Produits aux Actions

Le lien critique entre les produits CRM et les Actions CGRateS est la **convention de nommage** :

```
# Dans CGRateS
ActionsId = "Action_au-premium-plan-1"

# Dans le CRM
product_slug = "au-premium-plan-1"

# Dans le playbook (construit automatiquement)
cgr_action_name = "Action_" + product_slug
# Résultat : "Action_au-premium-plan-1"
```

## Comment les Playbooks Exécutent les Actions

**Remarque :** Utiliser `ExecuteAction` est juste un modèle pour ajouter des soldes. Vous pourriez également appeler directement `SetBalance` ou d'autres API CGRateS à l'intérieur d'un playbook. Cependant, `OmniCRM-API/Provisioners/plays/play_topup_charge_then_action.yaml` utilise le modèle ExecuteAction car c'est une approche commune et réutilisable qui sépare la logique de solde (Actions) de la logique de provisionnement (Playbooks).

Lorsqu'un produit est provisionné en utilisant ce modèle, le playbook construit le nom de l'Action à partir du product_slug :

```
# Dans play_topup_charge_then_action.yaml
- name: Set cgr_action_name fact
  set_fact:
    cgr_action_name: "Action_{{
api_response_product.json.product_slug }}"

# Exécuter l'action
- name: Exécuter l'action CGRateS
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "APIerSv1.ExecuteAction",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Account": "{{ service_uuid }}",
          "ActionsId": "{{ cgr_action_name }}"
        }]
      }
```

**Lors de l'utilisation de ce modèle :** L'Action **doit exister dans CGRateS** avant que le playbook ne s'exécute, sinon l'exécution échouera avec une erreur "Action non trouvée".

Voir aussi : Guide des Playbooks Ansible

# Exemple Complet : Définir un Nouveau Produit

Cette section décrit la création d'un nouveau produit complet "Pack de Données de 50 Go" :

## 1. Définir l'Action dans CGRateS

Voir l'exemple Addon de Données Simple à l'étape 5 ci-dessus pour savoir comment définir une Action d'addon de données. Pour notre exemple de 50 Go,

l'Action suivrait le même modèle mais avec des unités différentes :

- **ActionsId** : `"Action_50gb-data-pack"` (correspond au product_slug ci-dessous)
- **BalanceType** : `"*data"`
- **DestinationIDs** : `"Dest_PLMN_OnNet"` (pour l'utilisation des données sur le réseau)
- **Units** : `50 * 1024 * 1024 * 1024` (50 Go en octets)
- **Identifier** : `"*topup"` (active le rollover)

## 2. Créer le Produit dans le CRM

```
product_50gb_data = {
    "category": "addon",
    "provisioning_play": "play_topup_charge_then_action",
    "relies_on_list": "",
    "contract_days": 0,
    "retail_cost": 25.00,
    "retail_setup_cost": 0,
    "product_slug": "50gb-data-pack",  # Doit correspondre à
ActionsId sans "Action_"
    "product_name": "50GB Data Pack",
    "comment": "50 Go de données à haute vitesse valables pendant
30 jours",
    "provisioning_json_vars": "",
    "terms": "Les données expirent après 30 jours. Les données non
utilisées se reportent si rechargées avant l'expiration.",
    "service_type": "mobile",
    "residential": True,
    "business": True,
    "enabled": True,
    "icon": "fa-solid fa-database",
    "features_list": "['50 Go de données à haute vitesse',
'Validité de 30 jours', 'Rollover activé']",
    "wholesale_cost": 20.00,
    "wholesale_setup_cost": 0,
    "auto_renew": "False",
    "customer_can_purchase": True
}

# Créer dans le CRM
response = session.put(
    crm_url + '/crm/product',
    json=product_50gb_data,
    headers=headers
)
assert response.status_code == 200
print("Produit créé : 50gb-data-pack")
```

## 3. Vérifier le Lien

Le produit peut maintenant être provisionné :

1. Le client achète "Pack de Données de 50 Go" (product_slug : `50gb-data-pack`)
2. Le playbook `play_topup_charge_then_action` s'exécute
3. Le playbook construit le nom de l'Action : `Action_50gb-data-pack`
4. Le playbook appelle `APIerSv1.ExecuteAction` avec ActionsId `Action_50gb-data-pack`
5. CGRateS trouve l'Action et l'exécute
6. Le client reçoit un solde de 50 Go

Voir aussi : Facturation et Paiements depuis les Playbooks

# Modèles de Produits Courants

Ces modèles montrent des configurations typiques de produits CRM et comment elles se lient aux Actions CGRateS. Pour les définitions d'Action correspondantes, voir Actions CGRateS et Comportements de Recharge.

## Modèle 1 : Plan Mensuel Multi-Solde

**Cas d'utilisation :** Plan mensuel complet avec données, voix, SMS, roaming

**Produit CRM :**

```
{
    "product_slug": "premium-monthly-plan",  # Lien vers
Action_premium-monthly-plan
    "product_name": "Premium Monthly Plan",
    "category": "addon",
    "retail_cost": 59.00,
    "auto_renew": "True",
    "contract_days": 30,
    "provisioning_play": "play_topup_charge_then_action",
    "features_list": "['100 Go de données', '5000 minutes', '5000
SMS', 'Roaming inclus']"
}
```

**Action :** Utilise `*topup_reset` avec plusieurs types de soldes. Voir Exemple 1 : Plan Mensuel Multi-Solde (Python).

## Modèle 2 : Addon de Données Simple (Rollover)

**Cas d'utilisation :** Achat de données unique qui se reporte si rechargé à nouveau

**Produit CRM :**

```
{
    "product_slug": "10gb-addon",  # Lien vers Action_10gb-addon
    "product_name": "10GB Data Addon",
    "category": "addon",
    "retail_cost": 15.00,
    "auto_renew": "False",  # Achat unique
    "contract_days": 0,
    "provisioning_play": "play_topup_charge_then_action",
    "features_list": "['10 Go de données avec rollover', 'Validité
de 30 jours']"
}
```

**Action :** Utilise `*topup` pour le comportement de rollover. Voir Actions CGRateS - Rollover.

## Modèle 3 : Plan Mensuel Fixe (Pas de Rollover)

**Cas d'utilisation :** Plan mensuel qui se réinitialise toujours à exactement le même montant

**Produit CRM :**

```
{
    "product_slug": "fixed-50gb-monthly",  # Lien vers Action_fixed-50gb-monthly
    "product_name": "Fixed 50GB Monthly",
    "category": "addon",
    "retail_cost": 35.00,
    "auto_renew": "True",
    "contract_days": 30,
    "provisioning_play": "play_topup_charge_then_action",
    "features_list": "['50 Go de données', 'Se réinitialise chaque mois', 'Pas de rollover']"
}
```

**Action :** Utilise `*topup_reset` pour empêcher le rollover. Voir Actions CGRateS - Comportement de Réinitialisation.

# Avancé : Chargeurs et Plans de Tarification

**Important :** Cette section couvre la tarification de base pay-per-use. CGRateS prend en charge des scénarios de tarification incroyablement sophistiqués, y compris :

- **Tarifs de roaming différentiels** : Prix différents pour les appels/SMS MO vs MT lors du roaming
- **Facturation spécifique aux protocoles** : CAMEL pour 2G/3G (SMS MO-MT, USSD), Diameter pour 4G/5G
- **Tarification contextuelle** : Tarifs basés sur le temps, l'emplacement, le segment de client, la qualité du réseau

- **Tarification multidimensionnelle** : Combinaison du type de réseau, du statut de roaming, de la destination et de l'heure de la journée

Les exemples ci-dessous montrent des configurations PAYG simples et à tarif fixe adaptées à la plupart des déploiements OmniCRM.

Pour la facturation à la consommation (où l'utilisation au-delà des soldes inclus coûte de l'argent), vous devez définir des Plans de Tarification qui spécifient des montants en dollars à facturer.

## Comment Fonctionne la Tarification

Lorsqu'un client utilise un service (données, voix, SMS) :

1. CGRateS vérifie les soldes unitaires (par exemple, Go de données inclus)
2. Si un solde unitaire existe, l'utilisation est déduite de celui-ci
3. Si aucun solde unitaire ou solde épuisé, CGRateS revient au solde monétaire
4. **Plan de Tarification** détermine le prix (par exemple, 0,10 $ par Mo)
5. Le coût calculé est déduit du solde monétaire du client

## Exemple : Configuration des Charges de

## Données Pay-Per-Use

```
# 1. Définir un Tarif de Destination (le prix)
OCS_Obj.SendData({
    "method": "ApierV2.SetTPDestinationRate",
    "params": [{
        "TPid": tpid,
        "ID": "DR_DATA_PAYG",
        "DestinationRates": [{
            "DestinationId": "Dest_PLMN_OnNet",
            "RateId": "RT_DATA_$0_10_PER_MB",
            "RoundingMethod": "*up",
            "RoundingDecimals": 4,
            "MaxCost": 0,
            "MaxCostStrategy": ""
        }]
    }]
})

# 2. Définir le tarif réel
OCS_Obj.SendData({
    "method": "ApierV2.SetTPRate",
    "params": [{
        "TPid": tpid,
        "ID": "RT_DATA_$0_10_PER_MB",
        "RateSlots": [{
            "ConnectFee": 0,          # Pas de frais de connexion
            "Rate": 0.10,             # 0,10 $ par unité
            "RateUnit": "1048576",    # 1 Mo (1024 * 1024 octets)
            "RateIncrement": "1048576", # Facturer par Mo
            "GroupIntervalStart": "0s"
        }]
    }]
})

# 3. Définir un Plan de Tarification qui utilise ce tarif
OCS_Obj.SendData({
    "method": "ApierV2.SetTPRatingPlan",
    "params": [{
        "TPid": tpid,
        "ID": "RP_PAYG_DATA",
        "RatingPlanBindings": [{
            "DestinationRatesId": "DR_DATA_PAYG",
```

```
            "TimingId": "*any",
            "Weight": 10
        }]
    }]
})

# 4. Lier le Plan de Tarification à un Profil de Tarification
(pour des comptes spécifiques)
OCS_Obj.SendData({
    "method": "ApierV2.SetTPRatingProfile",
    "params": [{
        "TPid": tpid,
        "Tenant": tenant,
        "Category": "data",
        "Subject": "*any",  # Ou identifiant de compte spécifique
        "RatingPlanActivations": [{
            "ActivationTime": "2024-01-01T00:00:00Z",
            "RatingPlanId": "RP_PAYG_DATA",
            "FallbackSubjects": ""
        }]
    }]
})

# 5. Charger le plan tarifaire
OCS_Obj.SendData({
    "method": "APIerSv1.LoadTariffPlanFromStorDb",
    "params": [{
        "TPid": tpid,
        "DryRun": False,
        "Validate": True,
        "APIOpts": {}
    }]
})
```

**Ce que cela fait :**

- Lorsqu'un client n'a pas de solde de données (ou en manque), l'utilisation des données coûte **0,10 $ par Mo**
- La charge est déduite de leur solde monétaire (`*monetary`)
- Les incréments de facturation sont de 1 Mo (arrondis)

## Exemple : Tarifs des Appels Vocaux

```
# Définir différents tarifs pour différentes destinations
OCS_Obj.SendData({
    "method": "ApierV2.SetTPRate",
    "params": [{
        "TPid": tpid,
        "ID": "RT_VOICE_DOMESTIC_$0_30_PER_MIN",
        "RateSlots": [{
            "ConnectFee": 0.15,           # 0,15 $ de frais de
connexion
            "Rate": 0.30,                 # 0,30 $ par minute
            "RateUnit": "60s",            # Facturer par minute
            "RateIncrement": "60s",       # Arrondir à des minutes
pleines
            "GroupIntervalStart": "0s"
        }]
    }]
})

OCS_Obj.SendData({
    "method": "ApierV2.SetTPRate",
    "params": [{
        "TPid": tpid,
        "ID": "RT_VOICE_INTL_$1_50_PER_MIN",
        "RateSlots": [{
            "ConnectFee": 0.25,           # 0,25 $ de frais de
connexion
            "Rate": 1.50,                 # 1,50 $ par minute
            "RateUnit": "60s",
            "RateIncrement": "1s",        # Facturer par seconde
(plus précis)
            "GroupIntervalStart": "0s"
        }]
    }]
})
```

## Profils de Chargeurs (Optionnel)

Les chargeurs appliquent la tarification à des types d'événements spécifiques. Pour la plupart des cas d'utilisation, le chargeur par défaut est suffisant :

```
OCS_Obj.SendData({
    "method": "APIerSv1.SetChargerProfile",
    "params": [{
        "Tenant": tenant,
        "ID": "Charger_Default",
        "FilterIDs": [],
        "AttributeIDs": ["*constant:*req.RequestType:*none"],
        "Weight": 999
    }]
})
```

Voir aussi :

- Actions CGRateS et Comportements de Recharge - Approches de gestion des soldes (unitaire, monétaire, hybride) et configuration de tarification
- Destinations CGRateS - Comment définir des destinations géographiques et PLMN

# Meilleures Pratiques

## 1. Utiliser des Conventions de Nommage Cohérentes

```
# Bon - relation claire
ActionsId = "Action_premium-plan-100gb"
product_slug = "premium-plan-100gb"

# Mauvais - pas de relation claire
ActionsId = "Action_Plan_A"
product_slug = "premium_100"
```

## 2. Définir les Actions Avant les Produits

Créez toujours les Actions CGRateS avant de créer les produits CRM. Le playbook de provisionnement échouera s'il essaie d'exécuter une Action inexistante.

## 3. Inclure la Journalisation CDR

Ajoutez toujours l'action `*cdrlog` pour suivre les ajouts de solde :

```
{
    "Identifier": "*cdrlog",
    "BalanceType": "*generic",
    "ExtraParameters": "
{\"Category\":\"^activation\",\"Destination\":\"Nom du
Produit\"}",
    "Weight": 80
}
```

## 4. Utiliser des Identifiants de Solde Descriptifs

```
# Bon
"BalanceId": "Premium_Data_100GB"
"BalanceId": "AU_Voice_Domestic__" + str(units)

# Mauvais
"BalanceId": "data1"
"BalanceId": "balance"
```

## 5. Documenter les Unités Clairement

```
# Bon - montre le calcul
Units = 100 * 1024 * 1024 * 1024  # 100 Go en octets
Units = 3000 * 60 * 1000000000    # 3000 minutes en nanosecondes

# Mauvais - nombre magique
Units = 107374182400
```

## 6. Utiliser le Poids de Manière Stratégique

Poids plus élevé = priorité plus élevée pour l'exécution et la consommation :

```
# Ordre d'exécution (poids plus élevé = s'exécute en premier)
{"Identifier": "*reset_account", "Weight": 700}
{"Identifier": "*topup_reset", "Weight": 90}
{"Identifier": "*cdrlog", "Weight": 80}

# Consommation de solde (poids plus élevé = consommé en premier)
"BalanceWeight": 1200  # Soldes premium/domestiques
"BalanceWeight": 1100  # Soldes de roaming
"BalanceWeight": 1000  # Soldes internationaux
```

## 7. Tester d'Abord avec de Petites Valeurs

Lors de la définition de nouveaux produits, testez d'abord avec de petits soldes :

```
# Test
Units = 100 * 1024 * 1024  # 100 Mo pour le test

# Production
Units = 100 * 1024 * 1024 * 1024  # 100 Go
```

# Dépannage

## Erreur : "Action non trouvée"

**Symptôme :** Le playbook échoue avec "Action non trouvée" ou "ActionsId n'existe pas"

**Cause :** Action non définie dans CGRateS, ou incohérence de nommage

**Solution :**

1. Vérifiez que l'Action existe : interrogez CGRateS pour l'Action
2. Vérifiez le nommage : assurez-vous que `ActionsId = "Action_" + product_slug`
3. Définissez l'Action si elle est manquante

## Erreur : "Destination non trouvée"

**Symptôme :** L'Action s'exécute mais les soldes ne sont pas créés

**Cause :** DestinationIDs référence une destination inexistante

**Solution :**

1. Définissez la destination dans CGRateS d'abord
2. Utilisez `*any` pour des destinations universelles
3. Vérifiez l'orthographe des IDs de destination

## Produit Créé Mais Aucun Solde Ajouté

**Symptôme :** Le produit est provisionné avec succès mais le client n'a pas de solde

**Cause :** L'Action existe mais n'a pas d'opérations d'ajout de solde

**Solution :**

1. Vérifiez que l'Action a des opérations `*topup` ou `*topup_reset`
2. Vérifiez que la valeur des Units est correcte (pas 0)
3. Vérifiez que l'ExpiryTime n'est pas déjà passé

## Soldes Ne Se Reportent Pas

**Symptôme :** Utilisation du rollover mais le solde inutilisé disparaît

**Cause :** Utilisation de `*topup_reset` au lieu de `*topup`

**Solution :**

- Changez l'Identifier de `*topup_reset` à `*topup`
- Assurez-vous que le BalanceId est cohérent à travers les recharges

Voir aussi : Actions CGRateS et Comportements de Recharge

# Résumé

Ce guide démontre une approche courante pour configurer des produits dans OmniCRM avec CGRateS. Les modèles présentés ici fonctionnent bien pour des cas d'utilisation typiques d'opérateurs de réseau mobile virtuel (MVNO) avec des plans prépayés/postpayés, des paquets de données et du roaming.

**Rappelez-vous :** CGRateS est capable de scénarios beaucoup plus sophistiqués que ceux couverts ici. Si vous avez besoin de fonctionnalités avancées telles que :

- Tarification différentielle MO/MT pour la voix et les SMS en roaming
- Facturation basée sur CAMEL pour les appareils hérités 2G/3G
- Variations de tarifs selon l'heure de la journée ou la saison
- Tarification basée sur les niveaux de clients (bronze/argent/or)
- Tarification basée sur la qualité du réseau ou QoS
- Routage à moindre coût avec plusieurs fournisseurs

... consultez directement la documentation CGRateS et travaillez avec votre configuration de plan de tarification indépendamment des définitions de produits CRM.

# Documentation Connexe

- Actions CGRateS et Comportements de Recharge - Guide détaillé sur les types d'Action et le comportement des soldes
- Guide des Playbooks Ansible - Comment les playbooks exécutent les Actions
- Facturation et Paiements depuis les Playbooks - Traitement des paiements lors du provisionnement
- Cycle de Vie du Produit - Gestion des produits à travers leur cycle de vie
- Concepts : Produits et Services - Fondamentaux des produits
- Documentation Officielle CGRateS - Référence complète de CGRateS

# Campagnes par Email & SMS (EDM)

Le module de campagnes en masse d'OmniCRM (EDM : Email Direct Électronique) permet aux opérateurs de créer des segments d'audience, de concevoir des modèles et d'envoyer des campagnes en masse aux clients via **deux canaux : email (via SMTP) et SMS (via OmniMessage)**. Une campagne cible un segment avec un modèle choisi sur un canal sélectionné, avec personnalisation par destinataire, gestion des désinscriptions et suivi. Ce guide couvre les deux canaux.

Voir aussi : Mailjet pour l'intégration transactionnelle Mailjet, Notification API pour les notifications système, Configuration et Personnalisation pour la configuration générale.

> **Remarque sur le champ d'application :** Ceci décrit uniquement le chemin d'envoi de la **campagne en masse** (email et SMS). Les emails transactionnels (bienvenue, factures, réinitialisations de mot de passe, livraison d'eSIM, mises à jour de billets) sont livrés séparément via l'API REST de Mailjet et ne sont pas affectés par les paramètres ci-dessous.

## Aperçu

Le module a ces éléments constitutifs :

| Concept | Description |
|---|---|
| **Segment** | Une audience ciblée, définie par des critères de filtre (type de facturation, type de contact, statut, solde, …) ou une liste d'adresses statiques. |
| **Modèle** | Contenu réutilisable pour un canal : un **corps HTML téléchargé par l'opérateur** (email) ou un court **message SMS**, tous deux avec personnalisation par destinataire. |
| **Campagne** | Un envoi : un modèle à un segment sur un **canal** choisi, avec suivi par destinataire. |
| **Canal** | Comment une campagne est livrée : **email** (relais SMTP) ou **SMS** (OmniMessage). |

## Canaux

Le `canal` d'une campagne décide de tout en aval : transport de livraison, format de modèle et quelle désinscription s'applique :

| Canal | Livré via | Personnalisation | Désinscription | Détail |
|---|---|---|---|---|
| **Email** | Relais SMTP (Mailjet, Mailgun, SES, …) | Modèles HTML Jinja2 | Préférences de catégorie `email` + désinscription + liste de suppression | Livraison p email |
| **SMS** | OmniMessage | Substitution `var1`–`var4` | Préférences de catégorie `sms` + OmniMessage STOP | Livraison S (OmniMess |

## Modèles d'email et personnalisation

Les modèles stockent du HTML brut que vous **téléchargez ou collez** dans l'éditeur de modèles. La personnalisation utilise la syntaxe **Jinja2**, donc les modèles peuvent contenir des variables et une logique simple :

- `{{ first_name }}`, `{{ last_name }}`, `{{ customer_name }}`, `{{ email }}`
- conditionnels et boucles, par exemple `{% if first_name %}Salut {{ first_name }}{% else %}Bonjour{% endif %}`

Tout champ supplémentaire fourni par le segment pour un destinataire est également disponible dans le modèle par son nom. L'email de chaque destinataire est rendu individuellement avant l'envoi.

# Utilisation des campagnes en masse

Ce guide couvre la création d'un segment, la création d'une campagne email ou SMS, son envoi et son suivi, ainsi que ce que voient les destinataires. Tout se trouve dans la section **Marketing** de la barre latérale (**Segments**, **Modèles**, **Campagnes**). (Destiné aux opérateurs ; rédigé à partir de l'implémentation actuelle.)

## Créer un segment

Un **segment** est l'audience ciblée par une campagne. Depuis **Marketing → Segments**, vous pouvez rechercher, actualiser et créer des segments. Chaque ligne montre les critères du segment, le nombre actuel de destinataires et la dernière actualisation ; cliquez sur une ligne pour l'ouvrir.

*La liste des Segments. Les comptes de destinataires se rafraîchissent à la demande ; cliquer sur une ligne ouvre le segment.*

**Créer un Segment** (ou cliquer sur une ligne pour modifier) ouvre l'éditeur de segment, où vous définissez l'audience à partir des attributs des clients : *Clients sans services*, *Clients à solde nul*, *Types de facturation*, *Types de contact* et *Statut du client*. Enregistrez d'abord le segment. **Générer le Segment** calcule ensuite la liste des destinataires, et **Exporter CSV** la télécharge.

*L'éditeur de segment : critères à gauche, un compte de destinataires en direct et une exportation CSV en bas.*

> Les segments sont agnostiques au canal : le même segment peut être utilisé pour une campagne email ou SMS. Les campagnes par email ciblent les contacts avec une adresse email. Les campagnes SMS ciblent les contacts avec un numéro de téléphone.

## Télécharger un modèle

Les campagnes par email envoient un modèle HTML. Depuis **Marketing → Modèles**, créez ou modifiez un modèle et utilisez **Télécharger .html** pour charger un fichier HTML directement dans le corps (ou le coller). Les modèles supportent la personnalisation Jinja2, et le panneau **Variables Disponibles** insère `{{customer_name}}`, `{{contact_firstname}}`, `{{contact_lastname}}` et `{{contact_email}}` au curseur.

Les campagnes SMS n'utilisent pas de modèle. Vous écrivez le message en ligne lorsque vous composez la campagne.

## Créer une campagne

Depuis **Marketing → Campagnes → Créer Campagne**, un court assistant vous guide à travers la campagne. Le commutateur **Canal** (Email / SMS) en haut de l'assistant détermine quelles étapes et champs apparaissent.

1. **Sélectionner le Segment** : choisissez l'audience. Le segment sélectionné montre son compte de destinataires.

*Étape 1 : choisissez le segment dont les contacts recevront la campagne. Le commutateur de canal se trouve au-dessus des étapes.*

2. **Modèle / Message** : pour une campagne **email**, choisissez un modèle HTML (cliquez sur une carte de modèle pour le sélectionner ; **Aperçu** montre le HTML rendu). Pour une campagne **SMS**, l'assistant passe cette étape. Il affiche un avis, et vous écrivez le message à l'étape suivante.

*Pour le SMS, un avis remplace l'étape du modèle. Vous continuez à écrire le message.*

3. **Détails de la Campagne** : la configuration spécifique au canal :
    - **Catégorie** : la catégorie de communication à laquelle appartient la campagne (chargée depuis `edm.communication_categories`). Le système exclut les destinataires qui se sont désinscrits de cette catégorie pour ce canal.
    - **Email** : *Nom de la Campagne*, *Objet*, *Email de l'Expéditeur*, *Nom de l'Expéditeur*.
    - **SMS** : *Nom de la Campagne*, *ID de l'Expéditeur* et le *Message SMS* (personnalisez avec `{{ first_name }}`, `{{ last_name }}`, `{{ customer_name }}`).

*Détails de la campagne par email : champs d'objet et d'expéditeur, plus le sélecteur de catégorie.*

*Détails de la campagne SMS : le formulaire passe à l'ID de l'expéditeur et au corps du message SMS.*

4. **Réviser & Enregistrer** : confirmez le canal, la catégorie, le segment et le contenu, puis enregistrez comme **brouillon**. **Voir les destinataires** liste les contacts du segment (nom, email ou téléphone, client) avant de vous engager.

*Révision pour une campagne par email.*

*Révision pour une campagne SMS, avec le lien **Voir les destinataires** et la note de sauvegarde en tant que brouillon.*

La campagne est toujours enregistrée d'abord comme un **brouillon**. Vous l'envoyez depuis la page de détails de la campagne (ci-dessous).

## Liste des Campagnes

**Marketing → Campagnes** liste toutes les campagnes avec leur **Canal** (Email / SMS), statut, compte de destinataires et, une fois envoyées, taux d'ouverture et de clics. Filtrez par **Toutes / Brouillon / Envoyées**, et cliquez sur n'importe quelle ligne pour ouvrir la campagne.

*La liste des Campagnes. La colonne Canal distingue l'email du SMS d'un coup d'œil. Les campagnes envoyées montrent leurs taux d'ouverture et de clics en ligne.*

## Envoyer et suivre une campagne

Cliquez sur une campagne pour ouvrir sa page de détails :

- Un **brouillon** montre un bouton **Envoyer la Campagne**. Le bouton est activé une fois que la campagne a un segment et, pour l'email, un modèle, ou pour le SMS, un message. Si quelque chose manque, la page liste ce qui est requis.
- Une fois **envoyée**, la page de détails montre :
  - **Statistiques de la Campagne** : pour *email*, livrée, taux d'ouverture, taux de clic, ouvert/cliqué/rebondi. Pour *SMS*, livré / échoué, avec un bouton **Synchroniser** qui tire les derniers comptes de livraison d'OmniMessage.
  - **Destinataires** : une liste par destinataire avec statut (`envoyé`, `livré`, `ouvert`, `cliqué`, `rebondi`, `échoué`). La colonne d'adresse montre l'adresse email (email) ou le numéro de téléphone (SMS).

*La page de détails de la campagne pour une campagne SMS brouillon, prête à être envoyée. Une fois envoyée, cette page montre les statistiques de livraison/engagement et le statut par destinataire ; pour le SMS,* ***Synchroniser*** *rafraîchit les comptes d'OmniMessage.*

Une fois envoyée, la même page fait apparaître le panneau **Statistiques de la Campagne** et la liste par destinataire :

*Une campagne email envoyée : compte de livraisons, taux d'ouverture/clic et totaux ouverts/cliqués/rebondis, aux côtés de la liste par destinataire et du statut de chaque destinataire.*

> Étant donné que le suivi est agnostique au fournisseur, **Livré** est dérivé de l'engagement. Une ouverture ou un clic prouve que le message est arrivé. Le taux d'ouverture est donc mesuré par rapport aux destinataires livrés (engagés) plutôt qu'à l'ensemble des envois.

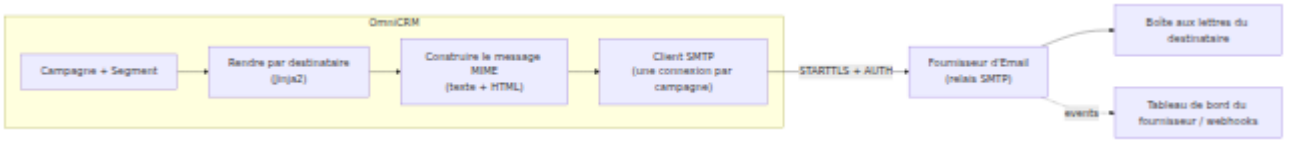
## Ce que voient les destinataires (désinscription)

Chaque email contient un lien de désinscription qui ouvre un **centre de préférences** : les catégories d'email du contact avec leur état d'opt-in actuel, une action **Enregistrer les préférences**, et une option **Se désinscrire de tous les emails**. La désinscription en un clic d'un client de messagerie désinscrit le contact uniquement de la catégorie de cette campagne.

> Les fichiers image ci-dessus se trouvent dans `docs/images/` (noms de fichiers préfixés `edm-`).

# Livraison par Email

Les emails de campagne sont livrés via **SMTP** à un relais configuré. Cela rend le chemin d'envoi indépendant du fournisseur : la même configuration d'OmniCRM peut pointer vers Mailjet, Mailgun, Amazon SES, ou le propre serveur SMTP d'un client.




Pour chaque destinataire, OmniCRM :

1. Rendre les parties HTML et texte du modèle, et l'objet, avec les variables du destinataire.
2. Construire un message MIME `multipart/alternative` (fallback texte + HTML).
3. L'envoyer via une seule connexion SMTP réutilisée pour toute la campagne.
4. Enregistrer un résultat par destinataire (`envoyé` lorsque le relais accepte le message, `échoué` en cas de rejet).

Chaque message porte également un en-tête de corrélation, `X-MJ-CustomID`, sous la forme `campaign_{campaign_id}_recipient_{recipient_id}`. Mailjet renvoie cela dans ses webhooks d'événements ; d'autres relais ignorent simplement l'en-tête inconnu.

# Configuration du Relais SMTP

Le relais est configuré sous le bloc `mailjet.smtp` dans `crm_config.yaml`. Malgré le nom de la section, il s'agit du relais SMTP générique utilisé pour toutes les campagnes EDM et peut pointer vers n'importe quel fournisseur.

```yaml
mailjet:
  api_key: "your-api-key"          # utilisé par l'email
transactionnel + comme nom d'utilisateur SMTP par défaut
  api_secret: "your-api-secret"    # utilisé par l'email
transactionnel + comme mot de passe SMTP par défaut
  smtp:
    host: in-v3.mailjet.com        # nom d'hôte du relais SMTP
    port: 587                      # port de soumission
    use_tls: true                  # mise à niveau de la connexion
avec STARTTLS
    use_auth: true                 # authentifier avec nom
d'utilisateur/mot de passe
    username: ''                   # vide -> utiliser mailjet
api_key
    password: ''                   # vide -> utiliser mailjet
api_secret
    timeout: 30                    # délai d'attente de
connexion/commande en secondes
```

## Paramètres

| Paramètre | Type | Requis | Par défaut | Description |
|---|---|---|---|---|
| `host` | Chaîne | Oui | `in-v3.mailjet.com` | Nom d'hôte du relais SMTP. |
| `port` | Entier | Non | `587` | Port de soumission. Utilisez `587` pour STARTTLS, `465` pour TLS implicite ; de nombreux fournisseurs offrent également `2525`/`25`. |
| `use_tls` | Booléen | Non | `true` | Lorsque `true`, la connexion est mise à niveau avec STARTTLS après connexion. Définissez `false` uniquement pour un serveur de test local. |
| `use_auth` | Booléen | Non | `true` | Lorsque `true`, OmniCRM s'authentifie avec `username`/`password`. Définissez `false` uniquement pour un serveur de test non authentifié. |
| `username` | Chaîne | Non | *(vide)* | Nom d'utilisateur SMTP. **Lorsqu'il est vide, OmniCRM** |

| Paramètre | Type | Requis | Par défaut | Description |
|---|---|---|---|---|
| | | | | **revient à la `api_key` de Mailjet.** Pour tout relais non-Mailjet, vous devez le définir explicitement. |
| `password` | Chaîne | Non | *(vide)* | Mot de passe SMTP. **Lorsqu'il est vide, OmniCRM revient à la `api_secret` de Mailjet.** Pour tout relais non-Mailjet, vous devez le définir explicitement. |
| `timeout` | Entier | Non | `30` | Délai d'attente de connexion et de commande, en secondes. |

> **Important :** `username`/`password` par défaut aux identifiants API de Mailjet uniquement par commodité. Lorsque vous pointez vers un autre fournisseur, **définissez toujours les deux explicitement**. Sinon, OmniCRM essaie de s'authentifier auprès de ce fournisseur avec les identifiants Mailjet et échoue.

# Référence de Configuration du Fournisseur

Les tableaux ci-dessous donnent les paramètres de connexion SMTP et le mécanisme de suivi natif pour les fournisseurs populaires. Les noms d'hôtes, ports et conventions d'identifiants peuvent changer ou varier selon la région/le plan. Confirmez-les par rapport à la documentation actuelle de votre

fournisseur. Les notes **Suivi** sont *informatives* : elles décrivent ce que chaque relais offre dans son propre tableau de bord si vous l'utilisez. OmniCRM envoie uniquement SMTP et ne détecte pas automatiquement ou n'affiche pas les statistiques natives du fournisseur.

## Mailjet

| Paramètre | Valeur |
|---|---|
| Hôte | `in-v3.mailjet.com` |
| Ports | `587` (STARTTLS), `465` (TLS), `25` / `2525` |
| Nom d'utilisateur | Clé API (Compte → Gestion des Clés API) |
| Mot de passe | Clé Secrète |

**Suivi :** Le suivi des ouvertures et des clics est appliqué par Mailjet (paramètre par défaut du compte, ou par message via les en-têtes `X-Mailjet-TrackOpen` / `X-Mailjet-TrackClick`). Les événements d'engagement sont disponibles dans le **tableau de bord Mailjet**, via l'**API Statistiques**, et en tant que **webhooks API d'événements**. OmniCRM inclut déjà un gestionnaire de webhook Mailjet ; corrélez les événements aux destinataires avec la valeur `X-MJ-CustomID` que OmniCRM envoie.

# Mailgun

| Paramètre | Valeur |
|---|---|
| Hôte | `smtp.mailgun.org` (US) · `smtp.eu.mailgun.org` (EU) |
| Ports | `587` / `465` / `25` / `2525` |
| Nom d'utilisateur | `postmaster@your-sending-domain` (ou une connexion SMTP personnalisée créée sur le domaine) |
| Mot de passe | Le mot de passe d'identifiant SMTP pour ce domaine |

**Suivi :** Activez le suivi des ouvertures/clics par domaine, ou par message via `X-Mailgun-Track`, `X-Mailgun-Track-Opens`, `X-Mailgun-Track-Clicks`. Attachez une étiquette de campagne avec `X-Mailgun-Tag` et des données de corrélation personnalisées avec `X-Mailgun-Variables` (JSON). Les statistiques apparaissent dans le **tableau de bord Mailgun** et l'**API Événements/Statistiques** ; l'engagement peut être poussé vers OmniCRM via **webhooks Mailgun**.

# Amazon SES

| Paramètre | Valeur |
|---|---|
| Hôte | `email-smtp.<region>.amazonaws.com` (par exemple `email-smtp.us-east-1.amazonaws.com`) |
| Ports | `587` / `2587` (STARTTLS), `465` / `2465` (TLS), `25` |
| Nom d'utilisateur | Nom d'utilisateur **SMTP** SES (généré dans la console SES) |
| Mot de passe | Mot de passe **SMTP** SES (distinct de vos clés d'accès AWS) |

**Suivi :** Le suivi des ouvertures et des clics est activé via un **ensemble de configuration**, attaché par message avec l'en-tête `X-SES-CONFIGURATION-SET` ; ajoutez des dimensions de campagne avec `X-SES-MESSAGE-TAGS`. Les événements (envoi, livraison, ouverture, clic, rebond, plainte) sont livrés à une destination d'événements (SNS, Kinesis Firehose ou EventBridge) et affichés dans **CloudWatch**. Il n'y a pas de tableau de bord d'engagement intégré, donc prévoyez une destination d'événements si vous souhaitez des statistiques.

## SendGrid (Twilio)

| Paramètre | Valeur |
|---|---|
| Hôte | `smtp.sendgrid.net` |
| Ports | `587` / `465` / `25` / `2525` |
| Nom d'utilisateur | la chaîne littérale `apikey` |
| Mot de passe | une clé API SendGrid avec permission **Mail Send** |

**Suivi :** Le suivi des ouvertures/clics est défini dans les **Paramètres de Suivi** (compte) ou par message via l'en-tête `X-SMTPAPI` (JSON `filters` tels que `opentrack`/`clicktrack`, plus `category` et `unique_args` pour corrélation). Les statistiques se trouvent dans le **tableau de bord SendGrid** et l'**API Statistiques** ; l'engagement atteint OmniCRM via le **Webhook d'Événements**.

## Postmark

| Paramètre | Valeur |
|---|---|
| Hôte | `smtp.postmarkapp.com` |
| Ports | `587` / `2525` / `25` |
| Nom d'utilisateur | votre **Jeton API de Serveur** |
| Mot de passe | votre **Jeton API de Serveur** (même valeur que le nom d'utilisateur) |

**Suivi :** Activez les ouvertures avec `X-PM-Track-Opens: true` et les liens avec `X-PM-Track-Links: HtmlAndText`. Étiquetez les campagnes avec `X-PM-Tag` et ajoutez des corrélations avec les en-têtes `X-PM-Metadata-<key>`. Utilisez un **flux de messages de diffusion** pour l'envoi en masse/EDM. Les statistiques se trouvent dans le **tableau de bord Postmark** et l'**API Statistiques** ; l'engagement atteint OmniCRM via des **webhooks**.

## Mailchimp Transactional (Mandrill)

| Paramètre | Valeur |
|---|---|
| Hôte | `smtp.mandrillapp.com` |
| Ports | `587` / `465` / `25` |
| Nom d'utilisateur | votre nom d'utilisateur Mailchimp/Mandrill (généralement l'email du compte) |
| Mot de passe | toute clé API Mandrill valide |

**Suivi :** Par défaut du compte, ou par message via `X-MC-Track` (par exemple `opens,clicks_all`). Étiquetez avec `X-MC-Tags` et ajoutez des corrélations avec

`X-MC-Metadata` (JSON). Les statistiques se trouvent dans le **tableau de bord Mandrill** et l'API ; l'engagement atteint OmniCRM via des **webhooks**.

> **Remarque :** Le produit standard **Mailchimp Marketing** n'est *pas* un relais SMTP. Seul **Mailchimp Transactional (Mandrill)** offre l'envoi SMTP.

## Brevo (anciennement Sendinblue)

| Paramètre | Valeur |
|---|---|
| Hôte | `smtp-relay.brevo.com` |
| Ports | `587` / `465` / `25` / `2525` |
| Nom d'utilisateur | votre identifiant de compte Brevo (email) |
| Mot de passe | votre **clé SMTP** (Brevo → SMTP & API → SMTP) |

**Suivi :** Suivi des ouvertures/clics au niveau du compte ; statistiques dans le **tableau de bord Brevo** ; l'engagement atteint OmniCRM via des **webhooks**.

## SMTP générique / auto-hébergé

| Paramètre | Valeur |
|---|---|
| Hôte | Comme fourni (par exemple un relais Postfix, ou un relais Microsoft 365 / Google Workspace d'un client) |
| Ports | Typiquement `587` (STARTTLS) ou `465` (TLS) |
| Nom d'utilisateur / Mot de passe | Comme fourni par l'opérateur |

**Suivi :** Un serveur SMTP simple ne fournit **aucun suivi d'engagement natif**. Utilisez le suivi natif d'OmniCRM (ci-dessous) pour les ouvertures et les clics, et le traitement des rebonds (DSN) pour les échecs de livraison.

# Livraison SMS (OmniMessage)

Les campagnes en masse peuvent également être **SMS**, livrées via le service séparé OmniMessage. Le `canal` d'une campagne est `email` (par défaut) ou `sms` ; pour le SMS, définissez `channel: sms` et un `sms_message`.

**Destinataires** sont les contacts du segment qui ont un **numéro de téléphone**, filtrés par la catégorie de la campagne sur le canal **`sms`**.

**L'envoi** suit le flux de campagne en masse d'OmniMessage (OmniCRM effectue les trois étapes) :

1. `POST /api/campaign_lists` : téléchargez les destinataires sous forme de liste nommée (`msisdn` + `var1`–`var4`).
2. `POST /api/campaigns` : créez la campagne (`message`, `list_id`, `source_msisdn`, `source_smsc`, `drip_tps`).
3. `POST /api/campaigns/control` `{action: "start"}` : démarrez le goutte-à-goutte.

**Personnalisation :** Les modèles OmniMessage ne supportent que `{{ var1 }}`–`{{ var4 }}` (plus `{{ msisdn }}`). Écrivez les espaces réservés nommés familiers et OmniCRM les mappe automatiquement : `first_name`→`var1`, `last_name`→`var2`, `customer_name`→`var3`, `email`→`var4`.

**Statut :** OmniMessage n'a **pas de webhook de livraison**. Interrogez `GET /api/campaigns/:id` pour ses `stats`. `POST /crm/email-campaign/campaigns/<id>/sms-sync` tire les dernières statistiques dans la campagne (`total_delivered` / `total_bounced` / `total_sent`). Exécutez-le sur un calendrier pour des chiffres en direct.

Configuration (`smsc` bloc dans `crm_config.yaml`) :

```yaml
smsc:
  campaigns_url: 'https://smsc:8443/api'   # base de l'API des
campagnes en masse
  source_msisdn: 'YourBrand'                 # expéditeur par
défaut
  source_smsc: 'IMS_SMSC'                    # route SMSC
  drip_tps: 50                               # taux d'envoi par
campagne
  verify_tls: false                          # OmniMessage utilise
un certificat auto-signé
  timeout: 30
```

> OmniMessage maintient sa propre suppression STOP/d'opt-out ; OmniCRM applique également les préférences de catégorie `sms` du contact lors de la création de la liste des destinataires.

# Statistiques et Suivi

Il existe deux façons indépendantes de mesurer une campagne. Elles ne sont pas mutuellement exclusives.






## Ce que chaque modèle vous donne

| Métrique | Enregistré à l'envoi (chaque relais) | Dans la console propre de votre fournisseur [1] | Suivi natif d'OmniCRM (tout relais) |
|---|---|---|---|
| Envoyé / en file d'attente | ✓ | ✓ | ✓ |
| Échecs d'envoi (rejetés au relais) | ✓ | ✓ | ✓ |
| Livré | – | ✓ | ✓ (prouvé par une ouverture/un clic ; sinon *supposé* = envoyé − rebondi) |
| Ouvertures | – | ✓ | ✓ (pixel de suivi) |
| Clics | – | ✓ | ✓ (redirection de lien) |
| Rebonds | – | ✓ | ✓ (traitement DSN) |
| Plaintes pour spam | – | ✓ | boucle de retour d'information du fournisseur uniquement |
| Désinscriptions | – | ✓ | ✓ (lien de désinscription + `List-Unsubscribe`) |

[1] **OmniCRM ne détecte pas quel fournisseur vous utilisez.** Il est configuré uniquement avec un hôte SMTP et des identifiants, qui semblent identiques pour chaque relais. La colonne du milieu est ce que votre relais offre, vu dans **la console propre de ce fournisseur** (pas dans OmniCRM), à moins que vous ne configuriez son webhook (Modèle 1) ou n'utilisiez le suivi natif d'OmniCRM (Modèle 2).

# Modèle 1 : Suivi natif du fournisseur

La plupart des relais peuvent appliquer leur propre pixel d'ouverture et suivi des liens et enregistrer les événements. Étant donné qu'OmniCRM ne détient qu'un hôte SMTP et des identifiants, **il ne peut pas dire quel fournisseur c'est**, et il ne fait pas apparaître ces chiffres. Vous les lisez en vous connectant vous-même à **la console propre de ce fournisseur**. Cela ne nécessite aucune configuration d'OmniCRM. C'est simplement ce que votre relais choisi offre.

Pour tirer ces événements **dans les analyses d'OmniCRM**, OmniCRM doit ingérer le **webhook** du fournisseur et faire correspondre chaque événement à un destinataire en utilisant le champ de corrélation par message :

| Fournisseur | Champ de corrélation qu'OmniCRM définit / utilise |
|---|---|
| Mailjet | `X-MJ-CustomID` (gestionnaire déjà présent) |
| Mailgun | `X-Mailgun-Variables` |
| Amazon SES | `X-SES-MESSAGE-TAGS` |
| SendGrid | `unique_args` (via `X-SMTPAPI`) |
| Postmark | `X-PM-Metadata-*` |
| Mandrill | `X-MC-Metadata` |

Chaque format d'événement de fournisseur diffère, donc ingérer les événements d'un nouveau fournisseur nécessite un gestionnaire de webhook

spécifique au fournisseur.

## Modèle 2 : Suivi natif d'OmniCRM (portable)

Étant donné que le suivi natif du fournisseur n'existe que tant que vous restez sur ce fournisseur, OmniCRM mesure l'engagement lui-même, de manière identique sur **tout** relais. Ceci est **implémenté et activé par défaut** :

- **Ouvertures** : OmniCRM injecte un pixel de suivi 1×1 dans le HTML. La requête enregistre l'ouverture contre le destinataire.
- **Clics** : OmniCRM réécrit les liens pour rediriger à travers lui-même. Il enregistre le clic et redirige vers la destination.
- **Livré** : une ouverture ou un clic est une preuve de livraison. OmniCRM marque le destinataire **livré** lors de la première frappe de pixel/redirection, sans webhook de fournisseur requis. Les destinataires qui ont été `envoyés` et n'ont jamais rebondi sont comptés comme *supposés livrés*.
- **Rebonds** : les campagnes utilisent un chemin de retour qui achemine les notifications de rebond (DSNs) vers le serveur de messagerie entrant d'OmniCRM pour traitement.
- **Désinscriptions** : un en-tête `List-Unsubscribe` et un lien de désinscription dans le corps alimentent la liste de suppression de désinscription existante.

Avec ce modèle, toutes les statistiques vivent dans OmniCRM et survivent à un changement de fournisseur d'email.

## État de support actuel

- **Actif (par défaut) :** Envoi de campagne SMTP et **suivi natif d'OmniCRM sur tout relais**. Cela fournit `envoyé`/`échoué` à l'envoi, plus **livré, ouvert, cliqué, rebondi** détecté par OmniCRM lui-même (pixel, redirection de clic, rebond VERP). Aucun webhook de fournisseur requis. Voir Configuration du suivi.
- **Optionnel :** si vous utilisez Mailjet, son webhook d'événements peut également alimenter l'engagement (corrélé via `X-MJ-CustomID`). OmniCRM ne dépend pas de cela.

## Configuration du suivi

Le suivi natif d'OmniCRM est configuré sous le bloc `edm` dans `crm_config.yaml` :

```
edm:
  enable_open_tracking: true     # injecter le pixel d'ouverture dans le HTML de la campagne
  enable_click_tracking: true    # réécrire les liens pour rediriger à travers OmniCRM
  enable_unsubscribe: true       # ajouter l'en-tête List-Unsubscribe + lien de désinscription dans le corps
  tracking_base_url: ''          # URL publique que les destinataires peuvent atteindre ; revient à crm.base_url
  tracking_secret: ''            # secret qui signe les jetons de suivi ; revient à mailjet.api_secret
  bounce_domain: ''              # domaine dont le MX pointe vers le serveur de messagerie entrant (active les rebonds VERP)
  bounce_prefix: 'bounce'        # préfixe de la partie locale VERP -> bounce+<token>@<bounce_domain>
```

| Paramètre | Type | Par défaut | Description |
| --- | --- | --- | --- |
| `enable_open_tracking` | Booléen | `true` | Injecter un pixel d suivi d'ouverture 1×1 dans le corps HTML. |
| `enable_click_tracking` | Booléen | `true` | Réécrire les liens `http(s)` absolus pour rediriger à travers OmniCRM afin que les clics soient enregistrés |
| `enable_unsubscribe` | Booléen | `true` | Ajouter un en-tête `List-Unsubscrib` (+ un clic RFC 80 et un lien de désinscription dan le corps. |
| `tracking_base_url` | Chaîne | *(crm.base_url)* | URL publique pou les liens de suivi. **Les clients de messagerie des destinataires doivent pouvoi accéder, donc n'utilisez pas `localhost`.** Revie à `crm.base_url` lorsqu'il est vide. |
| `tracking_secret` | Chaîne | *(mailjet.api_secret)* | Secret utilisé pou signer les jetons d suivi HMAC. Revie à |

| Paramètre | Type | Par défaut | Description |
| --- | --- | --- | --- |
| | | | `mailjet.api_sec` lorsqu'il est vide. |
| `bounce_domain` | Chaîne | *(vide)* | Domaine pour les adresses de rebo VERP (`bounce+ <token>@<domain>`. Son MX doit point vers le serveur de messagerie entra d'OmniCRM. Vide désactive le chem de retour VERP. |
| `bounce_prefix` | Chaîne | `bounce` | Préfixe de la parti locale pour les adresses de rebo VERP. |

Les jetons sont des paires `(campaign_id, recipient_id)` signées par HMAC, donc le suivi n'a pas besoin de colonne de base de données supplémentaire et les liens ne peuvent pas être falsifiés ou énumérés sans le secret.

> **Remarque sur les jetons :** le jeton est les ID de campagne/destinataire *signés*, pas cryptés. Les ID sont encodés en base64, donc un destinataire qui décode une URL de suivi peut lire les numéros de campagne et de destinataire (séquentiels). Les jetons sont donc à l'épreuve des falsifications mais pas opaques. Si les ID ne doivent pas être exposés, passez à des jetons aléatoires par destinataire stockés sur `EmailCampaignRecipient` (ajoute une colonne) ou un payload de jeton crypté.

Les points de terminaison de suivi publics (sans authentification) sont servis par l'API :

| Point de terminaison | But |
| --- | --- |
| `GET /crm/email-campaign/edm/open/<token>` | Enregistre une ouverture ; retourne un GIF transparent 1×1. |
| `GET /crm/email-campaign/edm/click/<token>?u=<url>` | Enregistre un clic, puis redirige 302 vers `<url>` (http/https uniquement). |
| `GET` ou `POST /crm/email-campaign/edm/unsubscribe/<token>` | Ajoute le destinataire à la liste de suppression **et enregistre un opt-out `marketing`/`email` dans les préférences de communication du contact** (honoré lors des futurs envois). |

> **Important :** `tracking_base_url` doit être une URL que les destinataires peuvent atteindre. La capture des rebonds nécessite également `bounce_domain` avec un enregistrement MX pointant vers le serveur de messagerie entrant d'OmniCRM, qui reconnaît les destinataires VERP et enregistre le rebond au lieu de créer un ticket.

## Préférences de communication

Les opt-outs par contact sont stockés dans un champ JSON `communication_preferences` sur le contact (pas de table supplémentaire), structuré comme `{category: {channel: opted_in}}`. Les catégories sont `marketing`, `newsletter`, `product_updates`, `surveys`, et les canaux sont `email`, `sms`. Il utilise un modèle de désinscription : une catégorie/canal manquant signifie que le contact est opté in, donc de nouvelles catégories ou canaux n'ont pas besoin de changement de schéma.

Une désinscription de campagne enregistre un opt-out `marketing`/`email`, et la sélection des destinataires de la campagne exclut tout contact désinscrit de `marketing`/`email` (en plus de la liste de suppression globale). Les campagnes EDM sont traitées comme la catégorie `marketing`. Remarque : le filtre de

désinscription utilise `JSON_EXTRACT` et n'est pas indexé, ce qui est acceptable à des volumes de contacts typiques.

## Catégories de communication

Les catégories sont **définies par l'opérateur** dans `crm_config.yaml` sous `edm.communication_categories` (pas de table), sous forme de liste de `{key, name, channel, default_opt_in, active}` :

```yaml
edm:
  communication_categories:
    # Types de messages marketing par email (désinscription : default_opt_in true = abonné jusqu'à ce qu'il se désinscrive)
    - {key: marketing,       name: Marketing,           channel: email, default_opt_in: true}
    - {key: newsletter,      name: Newsletter,          channel: email, default_opt_in: true}
    - {key: promos,          name: Promotions & Offres, channel: email, default_opt_in: true}
    - {key: product_updates, name: Mises à Jour de Produits, channel: email, default_opt_in: true}
    - {key: surveys,         name: Enquêtes & Retours,  channel: email, default_opt_in: false}
    - {key: events,          name: Événements & Webinaires, channel: email, default_opt_in: false}
    # Types de messages marketing par SMS (opt-in : default_opt_in false = exclu jusqu'à ce qu'il opte in)
    - {key: marketing,       name: Marketing SMS,       channel: sms,   default_opt_in: false}
    - {key: promos,          name: Promotions SMS,      channel: sms,   default_opt_in: false}
    - {key: service_alerts,  name: Alertes de Service,  channel: sms,   default_opt_in: true}
```

Chaque catégorie appartient à un **canal** (`email` ou `sms`) et porte son propre **opt-in par défaut**, de sorte qu'un opérateur puisse exécuter, par exemple, une catégorie de désinscription *Promotions (email)* aux côtés d'une catégorie *Alertes SMS (sms)* nécessitant un opt-in.

L'opt-in effectif pour un contact se résout comme suit : une préférence de contact **explicite** l'emporte ; sinon, le **`default_opt_in`** de la catégorie s'applique.

| Point de terminaison | But |
|---|---|
| `GET /crm/email-campaign/communication-categories` | Liste le catalogue de catégories (lecture seule ; défini dans `crm_config.yaml`). |
| `GET /crm/email-campaign/contacts/<id>/communication-preferences` | Opt-in résolu par catégorie active pour un contact. |
| `PUT /crm/email-campaign/contacts/<id>/communication-preferences` | Définir une préférence (corps `{category, channel, opted_in}`). |

Chaque **campagne spécifie une** **`category`** (par défaut `marketing`). La sélection des destinataires exclut les contacts désinscrits de cette catégorie/canal, appliquant le **`default_opt_in`** de la catégorie pour les contacts sans préférence explicite.

Le **lien de désinscription est un centre de préférences** : il montre les catégories d'email du contact avec leur état d'opt-in actuel (la catégorie de la campagne mise en évidence), leur permet d'enregistrer des modifications, et offre *Se désinscrire de tous les emails* (une suppression globale). Une désinscription en un clic (RFC 8058 `List-Unsubscribe-Post`) désinscrit le contact uniquement de la catégorie de cette campagne.

# Livraison

Quel que soit le fournisseur, configurez l'authentification pour votre domaine d'envoi afin que les campagnes atteignent la boîte de réception :

| Mécanisme | But |
| --- | --- |
| **SPF** | Autorise les serveurs du fournisseur à envoyer pour votre domaine. |
| **DKIM** | Signe cryptographiquement les messages ; chaque fournisseur fournit l'enregistrement DKIM à publier. |
| **DMARC** | Aligne SPF/DKIM et définit une politique pour les échecs ; améliore le placement et le reporting. |
| **Return-Path** | Défini afin que les notifications de rebond (DSNs) soient acheminées là où OmniCRM (ou le fournisseur) peut les traiter. |

# Tests

Pour le développement et les tests, pointez le relais vers un serveur SMTP de débogage local qui capture les messages au lieu de les livrer :

```
mailjet:
  smtp:
    host: 127.0.0.1
    port: 1025
    use_tls: false
    use_auth: false
```

Un serveur de débogage est fourni à cet effet ; il imprime les destinataires, l'objet et l'ID de corrélation de chaque message (et, en mode verbeux, le corps rendu complet), afin que vous puissiez confirmer le rendu Jinja2 et la structure du message sans envoyer d'email en direct.

# Dépannage

**Campagne bloquée ou destinataires marqués `échoués` immédiatement**

- Vérifiez `host`/`port` et que `username`/`password` sont définis pour le fournisseur choisi (ils par défaut aux identifiants Mailjet lorsqu'ils sont vides).
- Confirmez que l'adresse de l'expéditeur est un expéditeur vérifié/autorisé sur le fournisseur.
- Vérifiez que l'authentification du domaine du fournisseur (SPF/DKIM) est complète.

**L'email est envoyé mais la personnalisation apparaît vide ou littérale `{{ ... }}`**

- Confirmez que le nom de la variable existe pour le segment (par exemple `first_name`).
- Vérifiez que le modèle utilise une syntaxe Jinja2 valide ; une erreur de modèle échoue ce destinataire et est enregistrée dans son message d'erreur.

**Le tableau de bord du fournisseur montre des ouvertures/clics mais les analyses d'OmniCRM ne le font pas**

- Cela est attendu à moins qu'OmniCRM n'ingère le webhook de ce fournisseur (Modèle 1) ou que le suivi natif d'OmniCRM soit activé (Modèle 2). L'ingestion d'événements Mailjet est disponible aujourd'hui ; d'autres fournisseurs nécessitent un gestionnaire spécifique au fournisseur.

**Aucune donnée d'ouverture/clic nulle part**

- Le suivi des ouvertures/clics doit être activé sur le fournisseur (paramètre de compte ou en-têtes par message). Sur un relais simple/auto-hébergé, il n'y a pas de suivi natif, donc utilisez le suivi natif d'OmniCRM.

# Gestion de l'inventaire eSIM

Ce guide explique comment configurer l'inventaire eSIM pour différents canaux de distribution : numérique (en ligne) contre imprimé (cartes physiques).

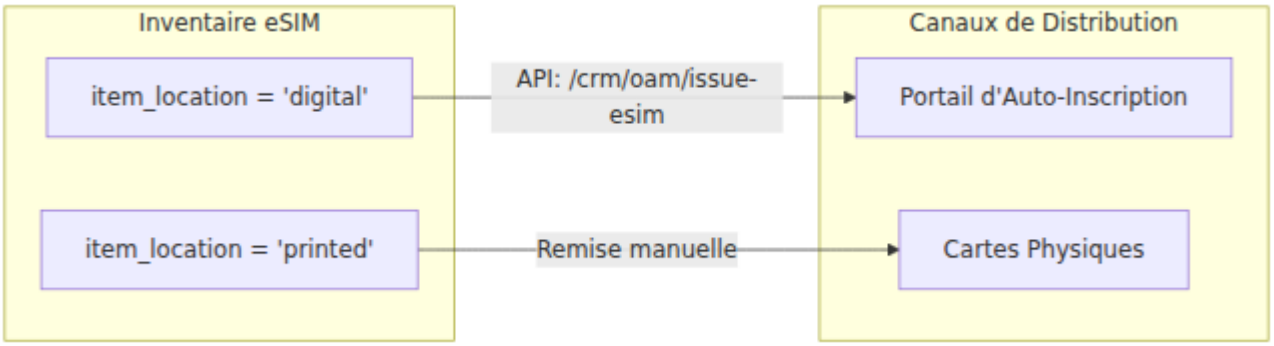
Voir aussi : Inventaire pour la gestion générale de l'inventaire, Provisionnement de carte SIM pour les flux de travail de provisionnement et l'API de l'émetteur eSIM.

## Vue d'ensemble

Les eSIM peuvent être distribuées par deux canaux :

1. **Distribution Numérique** : Émis instantanément via le portail d'auto-inscription, livré par email avec un code QR
2. **Distribution Imprimée** : Pré-imprimé sur des cartes en carton pour remise physique ou vente au détail

Le champ `item_location` dans le tableau d'inventaire contrôle quel canal une eSIM est disponible.



# Configuration du Canal de Distribution

Le champ `item_location` détermine comment une eSIM peut être distribuée :

| Valeur | Description | Disponible Via |
|---|---|---|
| `digital` | Disponible pour émission en ligne/instantanée | Portail d'auto-inscription, API |
| `printed` | Réservé à la distribution de cartes physiques | Attribution manuelle uniquement |
| Autres valeurs | Non disponible pour émission automatique | Attribution manuelle uniquement |

## Configuration des eSIM Numériques

Pour rendre les eSIM disponibles pour la distribution en ligne :

1. Accédez à **Inventaire** dans OmniCRM
2. Sélectionnez les éléments d'inventaire eSIM destinés à la distribution numérique
3. Réglez `item_location` sur `digital`
4. Enregistrez les modifications

## Configuration des eSIM Imprimées

Pour les eSIM qui ont été imprimées sur des cartes physiques :

1. Accédez à **Inventaire** dans OmniCRM
2. Sélectionnez les éléments d'inventaire eSIM qui sont imprimés
3. Réglez `item_location` sur `printed` (ou toute valeur autre que `digital`)
4. Enregistrez les modifications

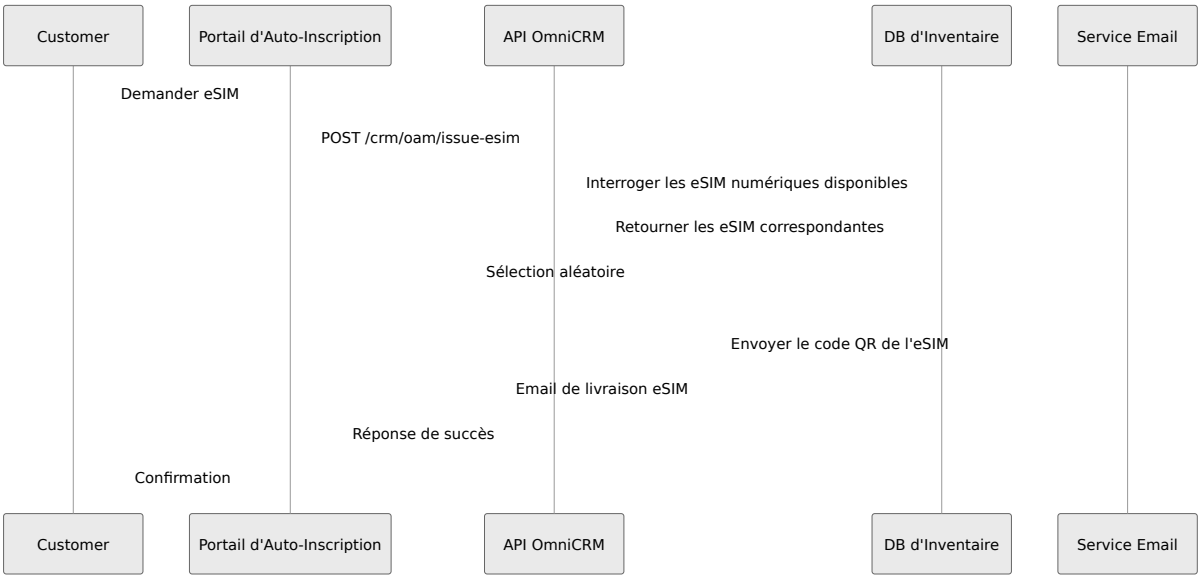
# Fonctionnement de l'Émission Numérique

Lorsqu'un client demande une eSIM via le portail d'auto-inscription via `POST /crm/oam/issue-esim` :

1. L'API interroge l'inventaire pour les eSIM disponibles
2. Seules les eSIM correspondant à **tous** les critères ci-dessous sont prises en compte
3. Une eSIM est sélectionnée au hasard dans le pool disponible
4. Les détails de l'eSIM sont envoyés au client par email

## Critères de Sélection de l'Inventaire

| Champ | Condition | Objectif |
|---|---|---|
| `item` | Contient "esim" (insensible à la casse) | Identifie les éléments d'inventaire eSIM |
| `item_location` | Égal à `digital` | Restriction au canal de distribution numérique |
| `customer_id` | Est nul | Pas encore attribué à un client |
| `service_id` | Est nul | Pas encore attribué à un service |




Pour des détails complets sur le point de terminaison issue-esim, y compris les paramètres de requête, la limitation de débit et les réponses d'erreur, voir Provisionnement de carte SIM - API publique de l'émetteur eSIM.

# Gestion des Erreurs

Si aucune eSIM numérique n'est disponible, l'API retourne :

| Statut HTTP | Type d'Erreur | Message |
|---|---|---|
| 503 | `NoInventory` | "Aucune eSIM actuellement disponible. Veuillez réessayer plus tard ou choisir une SIM physique." |

# Champs Clés de l'Inventaire pour les eSIM

| Champ | Type | Description |
|---|---|---|
| `item` | Chaîne | Identifiant de type d'élément (doit contenir "esim" pour les éléments eSIM) |
| `item_location` | Chaîne | Canal de distribution : `digital` pour en ligne, autres valeurs pour physique |
| `itemtext1` | Chaîne | ICCID : l'identifiant unique de l'eSIM |
| `management_url` | Chaîne | Chaîne d'activation LPA (par exemple, `LPA:1$rsp-eu.simlessly.com$...`) |
| `customer_id` | Entier | Client attribué (nul si non attribué) |
| `service_id` | Entier | Service attribué (nul si non attribué) |

Pour une référence complète de tous les champs d'éléments d'inventaire, voir Inventaire - Champs des Éléments d'Inventaire.

# Importation Automatique depuis HSS

Lorsque des SIM sont importées depuis le HSS à l'aide du script de provisionnement, deux champs sont définis automatiquement en fonction de l'enregistrement HSS :

**Canal de Distribution (`item_location`)** :

- Si le nom du lot contient **"digital"** (insensible à la casse), l'eSIM est marquée comme `item_location = 'digital'` pour la distribution en ligne

- Sinon, l'eSIM est définie sur l'emplacement physique par défaut (par exemple, "Siège Social")

**Chaîne d'Activation LPA (`management_url`)** :

- Le champ `lpa` de l'enregistrement SIM HSS est automatiquement importé comme `management_url`
- Cela permet la génération de codes QR dans le CRM pour l'activation de l'eSIM
- Si l'enregistrement HSS n'a pas de valeur `lpa`, l'eSIM est importée sans chaîne d'activation

Cela signifie que le canal de distribution et les données LPA peuvent être contrôlés au niveau HSS :

- Nommez les lots de manière appropriée (par exemple, "eSIM Numériques T1 2026", "Lot de Cartes Imprimées 3")
- Assurez-vous que les eSIM ont leur champ `lpa` rempli dans le HSS avant l'importation

# Meilleures Pratiques

## Ségrégation de l'Inventaire

- Gardez les pools d'eSIM numériques et imprimées séparés en utilisant `item_location`
- Surveillez les niveaux du pool numérique pour garantir la disponibilité pour les clients d'auto-inscription
- Envisagez de définir des alertes lorsque l'inventaire d'eSIM numériques est faible

## Mises à Jour Manuelles par Lot

Lors de l'importation ou de la mise à jour manuelle des eSIM :

1. **eSIM Numériques** : Réglez `item_location = 'digital'` lors de l'importation
2. **eSIM Imprimées** : Réglez `item_location = 'printed'` ou l'emplacement physique (par exemple, "Bureau de Georgetown")

## Conversion entre Canaux

Pour déplacer des eSIM entre les canaux :

- **Imprimé à Numérique** : Mettez à jour `item_location` de `printed` à `digital`
- **Numérique à Imprimé** : Mettez à jour `item_location` de `digital` à `printed` (uniquement si pas encore émis)

# Dépannage

## Aucune eSIM Disponible pour Émission en Ligne

**Symptômes** : Les clients voient "Aucune eSIM actuellement disponible" lorsqu'ils demandent une eSIM en ligne.

**Causes possibles** :

- Aucune eSIM n'a `item_location` défini sur `digital`
- Toutes les eSIM numériques sont déjà attribuées (à la fois `customer_id` et `service_id` sont non nuls)
- Les éléments eSIM n'ont pas "esim" dans le nom de l'élément

**Résolution** :

1. Vérifiez l'inventaire pour les eSIM avec `item_location = 'digital'`
2. Vérifiez qu'il y a des eSIM non attribuées (à la fois `customer_id` et `service_id` sont nuls)
3. Ajoutez plus d'eSIM au pool numérique ou convertissez celles imprimées si disponibles

# eSIM Imprimées Émises en Ligne

**Symptômes** : Les eSIM destinées aux cartes physiques sont émises via le portail.

**Causes possibles** :

- `item_location` non défini correctement sur les eSIM imprimées

**Résolution** :

1. Mettez à jour toutes les eSIM imprimées pour avoir `item_location = 'printed'`
2. Assurez-vous que les nouvelles importations d'eSIM imprimées définissent la valeur `item_location` correcte

# Système de Diffusion Cellulaire

Le Système de Diffusion Cellulaire dans OmniCRM permet aux opérateurs de réseaux mobiles d'envoyer des alertes d'urgence et des avertissements publics aux appareils mobiles dans des zones géographiques spécifiques. La Diffusion Cellulaire est une fonctionnalité de sécurité publique essentielle utilisée pour les alertes AMBER, les avertissements météorologiques, les alertes de tsunami et d'autres notifications d'urgence.

**Avantage Clé :** Contrairement aux messages SMS standard, les messages de Diffusion Cellulaire alerteront audiblement les téléphones qui sont en mode silencieux, à court de crédit ou en itinérance. Comme il s'agit d'un message diffusé, il est possible d'envoyer une alerte à chaque membre de la population portant un téléphone mobile en quelques secondes.

# Lien d'Avertissement Omnitouch (OWL)

La plateforme Lien d'Avertissement Omnitouch (OWL) fournit une solution complète pour les professionnels de la gestion des catastrophes et les opérateurs de réseaux mobiles :

- **Entité de Diffusion Cellulaire (CBE)** - Application web sécurisée pour les utilisateurs autorisés afin de créer et diffuser des messages d'avertissement d'urgence
- **Centre de Diffusion Cellulaire (CBC)** - Composant d'intégration réseau conforme aux normes qui se connecte aux réseaux cellulaires (2G/3G/4G/5G) pour distribuer des messages

OWL est conçu pour être utilisé sur tout appareil disposant d'un navigateur web (Chrome/Firefox/Safari/Edge), tels que des ordinateurs, des ordinateurs portables, des tablettes ou des téléphones mobiles.

# Aperçu

La Diffusion Cellulaire (également connue sous le nom de Système d'Avertissement Public ou PWS) permet aux opérateurs de :

- **Envoyer des Alertes d'Urgence** - Distribuer des informations critiques de sécurité à tous les appareils dans une zone
- **Cibler des Régions Géographiques** - Diffuser à des zones de suivi spécifiques ou des cellules réseau en contrôlant quelles tours cellulaires diffusent des messages
- **Supporter Plusieurs Langues** - Fournir des messages d'alerte en plusieurs langues simultanément (jusqu'à 500 caractères par langue)
- **Gérer le Cycle de Vie des Alertes** - Créer, mettre à jour, approuver, surveiller et supprimer des messages diffusés
- **S'intégrer avec des Systèmes Externes** - Se connecter avec l'infrastructure du CBC (Centre de Diffusion Cellulaire) via plusieurs interfaces de réseau cellulaire
- **Authentification à Deux Facteurs** - Processus d'approbation sécurisé utilisant des mots de passe à usage unique basés sur le temps (TOTP)
- **Règle des Deux Personnes** - Exigence optionnelle d'approbation d'une seconde personne avant la transmission du message

Contrairement aux SMS, la Diffusion Cellulaire ne nécessite pas de listes d'abonnés et peut atteindre tous les appareils capables dans une zone géographique instantanément, ce qui la rend idéale pour les alertes de sécurité publique critiques en termes de temps. Dans la plupart des réseaux, une diffusion à tous les appareils prend moins de 10 secondes.

# Cas d'Utilisation

La Diffusion Cellulaire est utilisée pour :

- **Avertissements d'Urgence** - Catastrophes naturelles (tremblements de terre, tsunamis, inondations, incendies)
- **Alertes AMBER** - Notifications d'enlèvement d'enfants

- **Alertes Météorologiques** - Avertissements de temps sévère, alertes de tornade
- **Sécurité Publique** - Menaces terroristes, déversements chimiques, évacuations
- **Messages de Test** - Tests de système et campagnes de sensibilisation publique (Les Messages de Test Mensuels n'alertent pas les utilisateurs mais seront reçus)

Le ciblage géographique est souvent d'une importance capitale dans les situations d'urgence. Par exemple, les alertes de tsunami conseillant à ceux près des zones côtières de chercher un terrain plus élevé ne devraient pas être envoyées aux personnes dans des zones intérieures éloignées du danger. En contrôlant quelles tours cellulaires diffusent des messages d'avertissement d'urgence, la portée de la transmission peut être limitée à la zone géographique appropriée.

# Cycle de Vie des Messages d'Avertissement d'Urgence

La transmission d'un message d'Avertissement d'Urgence nécessite une action rapide tout en fournissant des informations précises et une authentification pour garantir la validité du message.

Le cycle de vie du message se compose de quatre étapes :

1. **Définition du Message** - Type de message, contenu du message, paramètres d'expiration
2. **Ciblage du Message** - Zones géographiques et tours cellulaires à diffuser
3. **Approbation du Message** - Autorisation/vérification de l'identité de l'opérateur et approbation d'une seconde personne (si nécessaire)
4. **Révision du Message** - Confirmation finale avant transmission

Après diffusion, les messages peuvent être surveillés, mis à jour au fur et à mesure que les situations évoluent, et arrêtés lorsque le danger immédiat diminue.

# Étape 1 : Définition du Message

Cette étape définit les paramètres de base du message d'avertissement d'urgence à diffuser :

**Identifiant du Message**
Différents types de messages ont des identifiants différents, qui sont traités différemment par les téléphones récepteurs. Par exemple, un Message de Test Mensuel ne devrait pas alerter les utilisateurs réels mais ils recevront tout de même le message.

**Remarque :** Chaque modèle de message disponible dans le système OWL a déjà l'Identifiant de Message (MI) approprié intégré, donc les utilisateurs du système n'ont pas besoin de le sélectionner manuellement lors de la création d'un message d'alerte.

**Texte du Message**
Corps du texte limité à 500 caractères contenant le message à afficher aux utilisateurs finaux. Les messages peuvent être fournis en **plusieurs langues** en ajoutant le texte de la deuxième langue sous le premier. N'oubliez pas que la limite de 500 caractères s'applique au message total, y compris toutes les langues.

**Modèles de Message**
Les messages d'alerte de danger peuvent être prédéfinis à l'avance sous forme de "modèles" pour différents scénarios envisagés tels que :

- Inondations
- Tsunamis
- Tremblements de terre
- Tests périodiques
- Autres scénarios de catastrophe

Les modèles font gagner du temps lors des urgences. Ces modèles peuvent être modifiés si nécessaire lors de la définition du message, ou les messages peuvent être scriptés depuis zéro.

**Expiration et Répétitions du Message**
Les messages d'urgence ont une durée de vie finie pendant laquelle ils sont

pertinents. Lors de la définition du message :

- **Expire (minutes)** - Combien de temps le message continuera à être diffusé
- **Répétitions du Message** - Combien de fois il sera retransmis

Chaque téléphone ne montrera le message à l'utilisateur qu'une seule fois. Cependant, les sites cellulaires continueront à transmettre des messages jusqu'à ce que le temps d'expiration soit atteint pour s'assurer que les personnes entrant dans la zone de couverture depuis l'extérieur recevront le message.

# Étape 2 : Ciblage

Les messages de Diffusion Cellulaire sont envoyés au niveau de la tour cellulaire, et la portée géographique peut être limitée en sélectionnant quelles tours diffusent le message.

**Ciblage Optionnel**
Cette étape est optionnelle. Ne pas entrer d'informations de ciblage signifie que toutes les tours cellulaires transmettront le message d'avertissement d'urgence.

**Zones Ciblées Prédéfinies**
Le système OWL dispose d'une base de données de toutes les tours cellulaires et peut définir des zones cibles sur une carte. Les zones peuvent être ciblées en utilisant des zones prédéfinies (déterminées à l'avance pour une sélection rapide) ou en dessinant des zones personnalisées sur la carte.

**Outils de Dessin de Carte**
Des zones cibles personnalisées peuvent être créées en utilisant :

- Outil Polygone - Dessiner des limites de couverture précises
- Outil Cercle - Alertes rapides basées sur le rayon
- Outil Rectangle - Couverture alignée sur une grille

La fonctionnalité "Ajouter une Nouvelle Zone" permet de définir des zones cibles personnalisées qui peuvent être enregistrées pour une utilisation future.

# Étape 3 : Approbation

Un processus d'approbation valide que la personne émettant le message d'Avertissement d'Urgence est autorisée à le faire.

**Authentification à Deux Facteurs**
Utilise des mots de passe à usage unique basés sur le temps (TOTP) via :

- **Jeton physique** (comme un RSA SecurID)
- **Solution basée sur une application** (Google Authenticator, Authy, Microsoft Authenticator ou d'autres applications compatibles TOTP)

Les utilisateurs qui créeront ou approuveront des alertes de message brouillon doivent avoir une application d'authentification sur leur smartphone pour générer le code d'autorisation que le système demandera.

**Configuration de 2FA**
Lors de la première configuration de 2FA :

1. Installez une application d'authentification sur votre smartphone (Google Authenticator, Authy, Microsoft Authenticator, etc.)
2. Accédez aux paramètres de votre compte OWL et scannez le code QR avec votre application d'authentification
3. Entrez le code de vérification pour confirmer la configuration
4. Enregistrez les codes de sauvegarde dans un endroit sécurisé
5. Testez la génération de code avant les urgences

Pour des instructions détaillées sur la configuration de 2FA, voir Authentification à Deux Facteurs.

Si vous changez d'appareil mobile ou si l'application cesse de se synchroniser avec votre compte OWL, contactez votre Administrateur Système pour obtenir de l'aide. Les administrateurs peuvent réinitialiser les jetons 2FA depuis la page **Utilisateurs et Rôles → Utilisateurs**.

**Règle des Deux Personnes**
Lorsque le processus nécessite l'approbation d'une seconde personne, la personne émettant l'alerte doit saisir le mot de passe à usage unique basé sur

le temps de l'autre personne avant que le processus ne puisse se poursuivre. Cela fournit une supervision et minimise le risque d'abus.

**Rôles Utilisateurs Granulaires**
Des rôles utilisateurs individuels peuvent être configurés pour :

- Autoriser uniquement certains utilisateurs à envoyer des messages prédéfinis
- Restreindre le ciblage des messages à des régions spécifiques
- Exiger des flux de travail d'approbation supplémentaires

# Étape 4 : Révision

Une fois les étapes de Définition du Message, de Ciblage et d'Approbation terminées, l'opérateur doit examiner le message avant la diffusion finale. Une fois satisfait des détails du message, il peut transmettre le message.

**Vitesse de Transmission :** Dans la plupart des réseaux, une diffusion à tous les appareils dans le réseau prend moins de 10 secondes.

# Étape 5 : Surveillance et Mises à Jour

Une fois la diffusion du message commencée, les opérateurs peuvent surveiller et gérer les messages transmis.

**Retour d'Information Réseau**
Les réseaux cellulaires renvoient des informations sur les sites cellulaires qui ont diffusé le message. Si un site cellulaire est hors ligne ou indisponible, cela sera signalé à l'opérateur.

**Retransmission Automatique**
Si des sites cellulaires hors ligne redeviennent disponibles pendant que l'Avertissement d'Urgence est toujours actif, tous les téléphones connectés à cette cellule recevront le message.

**Mises à Jour en Cours de Diffusion**
Une fois diffusé, le message peut être :

- Mis à jour au fur et à mesure que la situation évolue

- Modifié avec un nouveau contenu de corps de message
- Rappelé/arrêté à tout moment

**Archives Historiques**
Toutes les informations sur les messages historiques peuvent être consultées et examinées à des fins d'audit.

# Structure du Message

Chaque message de Diffusion Cellulaire se compose de :

**Configuration du Message**

- **Identifiant du Message** - Identifiant unique pour le type d'alerte (par exemple, 4370 pour l'alerte de tremblement de terre ETWS, 4371 pour l'alerte de tsunami ETWS)
- **Catégorie** - Catégorie d'alerte (normale, urgence, élevée, extrême)
- **Période de Répétition** - Secondes entre les répétitions de diffusion
- **Nombre de Diffusions** - Combien de fois diffuser le message
- **Période d'Avertissement** - Durée en secondes pendant laquelle l'avertissement est valide
- **Indicateur de Canal** - Type de canal utilisé pour la diffusion

**Messages Localisés**

Chaque message CBC peut inclure plusieurs variantes linguistiques :

- **Langue** - Code de langue ISO (en, es, fr, zh, etc.)
- **Corps du Message** - Texte d'alerte dans cette langue (jusqu'à 1395 caractères)

Le système diffuse automatiquement toutes les variantes linguistiques, permettant aux destinataires de voir les alertes dans leur langue préférée.

**Zones de Suivi**

Définit le ciblage géographique pour l'alerte :

- **Zone de Suivi** - Identifiant géographique (ID de cellule, code de zone de suivi)
- **Opérateur** - Code de l'opérateur de réseau mobile (MCC-MNC)
- **Type de RAT** - Technologie d'Accès Radio (LTE, 5G, UMTS, GSM)

Plusieurs zones de suivi peuvent être spécifiées pour couvrir de plus grandes régions ou plusieurs opérateurs.

# Création d'un Message de Diffusion Cellulaire

**Via l'Interface Web :**

1. **Accéder à la Diffusion Cellulaire** - Accédez à l'interface de gestion du CBC depuis la navigation principale
2. **Cliquez sur "Créer Alerte"** - Ouvre le formulaire de création de message
3. **Configurer les Paramètres du Message :**
    - Identifiant du Message (par exemple, 4370 pour les alertes de tremblement de terre)
    - Catégorie (normale, élevée, extrême)
    - Période de Répétition (généralement 5-60 secondes)
    - Nombre de Diffusions (999 pour continu, ou nombre spécifique)
    - Période d'Avertissement (durée en secondes)
    - Indicateur de Canal (généralement "de base")

4. **Ajouter des Messages Localisés :**
    - Cliquez sur "Ajouter Langue"
    - Sélectionnez la langue dans le menu déroulant
    - Entrez le texte du message (max 1395 caractères pour GSM7, moins pour Unicode)
    - Répétez pour d'autres langues
5. **Définir les Zones de Suivi :**
    - Cliquez sur "Ajouter Zone de Suivi"
    - Entrez le code de zone de suivi
    - Sélectionnez l'opérateur (combinaison MCC-MNC)
    - Choisissez le type de RAT (LTE, 5G, etc.)
    - Répétez pour d'autres zones géographiques

6. **Réviser et Créer** - Vérifiez tous les détails et cliquez sur "Créer Alerte"

**Via l'API :**

**Point de Terminaison :** `PUT /crm/cbc/`

**Autorisation Requise :** `CREATE_CBC_MESSAGE`

**Corps de la Demande :**

```
{
  "messageIdentifier": "4370",
  "category": "emergency",
  "repetitionPeriod": 10,
  "numberOfBroadcasts": 999,
  "warningPeriodSec": 3600,
  "channelIndicator": "basic",
  "localized_messages": [
    {
      "language": "en",
      "messageBody": "EARTHQUAKE WARNING: Magnitude 6.5 earthquake
detected. Take cover immediately. Drop, Cover, Hold On."
    },
    {
      "language": "es",
      "messageBody": "ADVERTENCIA DE TERREMOTO: Terremoto de
magnitud 6.5 detectado. Cúbrase inmediatamente. Agáchese, Cúbrase,
Agárrese."
    }
  ],
  "tracking_areas": [
    {
      "tracking_area": "12345",
      "operator": "310-410",
      "rat_type": "LTE"
    },
    {
      "tracking_area": "12346",
      "operator": "310-410",
      "rat_type": "5G"
    }
  ]
}
```


**Réponse :**

```
{
  "cbc_message_id": 123,
  "cbc_unique_id": "550e8400-e29b-41d4-a716-446655440000",
  "messageIdentifier": "4370",
  "category": "emergency",
  "repetitionPeriod": 10,
  "numberOfBroadcasts": 999,
  "warningPeriodSec": 3600,
  "channelIndicator": "basic",
  "initiating_user": 5,
  "approving_user": null,
  "created": "2025-01-10T14:30:00Z",
  "localized_messages": [...],
  "tracking_areas": [...]
}
```


Le message est immédiatement envoyé au Centre de Diffusion Cellulaire pour transmission.

# Gestion des Messages Existants

**Voir Tous les Messages**

**Point de Terminaison :** `GET /crm/cbc/`

**Autorisation Requise :** `VIEW_CBC_MESSAGE`

Renvoie la liste de tous les messages CBC avec leur statut, horodatages et configuration.

**Mettre à Jour un Message**

**Point de Terminaison :** `PATCH /crm/cbc/{cbc_message_id}`

**Autorisation Requise :** `UPDATE_CBC_MESSAGE`

Met à jour le contenu du message, les zones de suivi ou les paramètres de diffusion. Les messages mis à jour sont renvoyés au CBC.

**Corps de la Demande :**

```
{
  "cbc_message_id": 123,
  "numberOfBroadcasts": 500,
  "localized_messages": [
    {
      "language": "en",
      "messageBody": "UPDATED: Earthquake warning still in effect.
Aftershocks possible."
    }
  ]
}
```

**Supprimer un Message**

**Point de Terminaison :** `DELETE /crm/cbc/{cbc_message_id}`

**Autorisation Requise :** `DELETE_CBC_MESSAGE`

Supprime le message de la base de données et tente de l'annuler sur le CBC.

# Flux de Travail d'Approbation

Les messages de Diffusion Cellulaire supportent un flux de travail d'approbation optionnel pour les alertes à enjeux élevés :

1. **Utilisateur Initiateur** - Membre du personnel qui crée l'alerte (`initiating_user` field)
2. **Utilisateur Approuvant** - Responsable qui approuve l'alerte avant la diffusion (`approving_user` field)

Si l'approbation est requise :

- Le message est créé avec `approving_user = null`
- Le message est maintenu dans un état "en attente d'approbation"
- L'utilisateur approuvant examine le message et l'approuve ou le rejette
- En cas d'approbation, `approving_user` est défini et le message est diffusé

Ce flux de travail est configurable en fonction de la politique de l'organisation.

# Identifiants de Message

Les identifiants de message standard suivent 3GPP TS 23.041 :

**Système d'Avertissement de Tremblement de Terre et de Tsunami (ETWS) :**

- **4370** - Avertissement de Tremblement de Terre ETWS
- **4371** - Avertissement de Tsunami ETWS
- **4372** - Avertissement Combiné de Tremblement de Terre et de Tsunami ETWS
- **4373-4378** - Autres Types d'Urgence ETWS
- **4379** - Message de Test ETWS

**Système d'Alerte Mobile Commercial (CMAS) / Alertes d'Urgence Sans Fil (WEA) :**

- **4352** - Alerte Présidentielle
- **4353-4355** - Alertes Extrêmes
- **4356-4359** - Alertes Sévères
- **4360-4363** - Alertes AMBER
- **4364-4367** - Messages de Sécurité Publique
- **4368-4369** - Tests d'État/Local
- **4380-4381** - Messages de Test

**Plages Personnalisées :**

- **0-999** - Réservé pour les alertes spécifiques à l'opérateur
- **1000-4095** - Types de messages personnalisés

# Intégration avec le Centre de Diffusion Cellulaire

L'Entité de Diffusion Cellulaire (CBE) a besoin d'un mécanisme pour livrer des messages à des réseaux cellulaires individuels. Le Centre de Diffusion Cellulaire OWL (CBC) se connecte à chaque réseau cellulaire pour envoyer des messages d'Avertissement d'Urgence au public.

**Redondance Multi-Réseau**
Dans le cas où un opérateur de réseau cellulaire est indisponible (panne ou pas de couverture), si un autre réseau cellulaire opérationnel est disponible, les utilisateurs recevront toujours des messages d'Avertissement d'Urgence via les autres réseaux disponibles.

**Instances CBC par Opérateur**
Pour garantir qu'il n'y a pas de connexion entre les réseaux concurrents, OWL exécute une instance CBC séparée pour chaque MNO ; elle n'est pas partagée entre les opérateurs.

## Configuration OWL CBC

L'URL du CBC est configurée dans `crm_config.yaml` :

```
cbc_url: "http://cbc.example.com:8080"
```

**Transmission de Message :**

Lorsqu'un message CBC est créé ou mis à jour :

1. OmniCRM CBE stocke le message dans sa base de données
2. Le message est formaté pour l'API CBC
3. Une requête HTTP POST est envoyée à `{cbc_url}/alerts/send`
4. Le CBC se connecte au réseau cellulaire via l'interface appropriée (CBSP, SBc-AP ou N50)
5. Le CBC accuse réception et commence la diffusion
6. Les appareils dans les zones de suivi cibles reçoivent l'alerte

**Suppression de Message :**

Lorsqu'un message est supprimé, le CBE tente de l'annuler sur le CBC pour arrêter les diffusions en cours.

# Points d'Intégration du Réseau Cellulaire

Différentes générations de réseaux cellulaires (2G/3G/4G/5G) ont chacune des interfaces uniques pour se connecter au Centre de Diffusion Cellulaire. En fonction des technologies utilisées par chaque réseau cellulaire, l'interface correcte doit être configurée.

Le Centre de Diffusion Cellulaire OWL prend en charge les interfaces de Diffusion Cellulaire 2G, 3G, 4G et 5G et s'est intégré à de nombreux composants de réseau cellulaire couramment utilisés.

**CBSP - 2G/3G - Contrôleur de Station de Base (BSC)**

L'interface du Protocole de Service de Diffusion Cellulaire (CBSP) relie le CBC au Contrôleur de Station de Base (BSC) contrôlant les stations de base 2G (GSM).

- Utilisé pour la messagerie de Diffusion Cellulaire 2G et 3G avec des déploiements combinés de Contrôleur de Réseau Radio/Contrôleur de Station de Base
- Peut être configuré soit comme un arrangement client, soit comme un serveur en fonction du fournisseur BSC
- Une connexion doit être établie entre tous les BSC dans le réseau et le CBC OWL
- Les interfaces sont continuellement surveillées avec des alertes pour indiquer si un lien CBSP est tombé

Remarque : 3GPP a défini le Protocole de Diffusion de Zone de Service (SABP) pour une utilisation dans des déploiements RNC autonomes. Cela peut être utilisé si nécessaire lorsque le CBSP n'est pas pris en charge pour les cellules 3G sur un RNC/BSC combiné, cependant, des tests supplémentaires et un support du fournisseur RNC peuvent être nécessaires.

**SBc-AP - 4G/5G Non-Standalone - MME/IWF**

L'interface SBc-AP relie le CBC OWL au MME (Entité de Gestion de Mobilité) servant les eNodeB/gNodeBs 4G et 5G.

- Utilisé pour les réseaux LTE 4G
- Également utilisé pour le 5G Non-Standalone (la majorité des déploiements en 2025)
- Une connexion doit être établie entre tous les MMEs dans le réseau et le CBC OWL
- Les interfaces sont continuellement surveillées avec des alertes pour indiquer si un lien SBc-AP est tombé

**N50 - 5G Standalone - AMF**

Pour les réseaux 5G autonomes, l'interface N50 connecte le CBC OWL à l'AMF (Fonction de Gestion d'Accès et de Mobilité) servant les gNodeBs 5G.

- L'interface est présente dans le CBC OWL
- N'a pas été largement testée avec des AMF tiers en raison du petit nombre de réseaux 5G SA commercialement disponibles en 2025
- Sera entièrement supportée à mesure que les déploiements 5G SA deviendront plus courants

**Exigences Réseau MNO**

Un réseau doit être en place entre le CBC OWL et le réseau de l'Opérateur de Réseau Mobile pour atteindre les interfaces décrites ci-dessus.

Cela est géré au cas par cas, mais nécessite généralement :

- Connexion croisée/fibre dédiée entre le CBC et le réseau MNO
- Chaque interface logiquement séparée
- Connectivité à chaque point d'intégration (MME, RNC, BSC) dans le réseau cellulaire

# Équipement Réseau Supporté

Le CBC OWL a été testé et s'intègre à des composants de réseau cellulaire couramment utilisés provenant de grands fournisseurs :

**Intégration des Données des Sites Cellulaires**

OWL prend en charge le scraping automatique des données à partir de :

- **Nokia NetAct**
- **Huawei U2000 / U2020**
- **ZTE NetNumen / ZXPOS**
- **Ericsson ENM**

Alternativement, les données des sites cellulaires peuvent être fournies à l'équipe des opérations Omnitouch périodiquement par e-mail.

# Gestion des Utilisateurs et Contrôle d'Accès

## Contrôle d'Accès Basé sur les Rôles (RBAC)

Le système OWL utilise le contrôle d'accès basé sur les rôles (RBAC) : les personnes (Utilisateurs Enregistrés) se voient attribuer un ou plusieurs Rôles, et chaque Rôle est un ensemble de Permissions. Les Permissions sont la plus petite unité d'accès (par exemple, créer une alerte de message brouillon). L'accès effectif d'un Utilisateur Enregistré est l'union des Permissions de tous les Rôles assignés.

**Composants RBAC :**

- **Utilisateurs** - Personnes réelles qui se connectent au système OWL
- **Permissions** - Micro capacités (par exemple, approuver un message brouillon, créer un message, voir des rapports)
- **Rôles** - Ensembles nommés de permissions (par exemple, Approuveurs de Messages, Créateurs de Messages)
- **Attribution** - Les utilisateurs reçoivent un ou plusieurs Rôles ; les permissions s'agrègent

**Avantages du RBAC :**

1. **Protection des Données** - Les utilisateurs ne voient et ne font que ce qu'ils sont autorisés à faire
2. **Adaptation Opérationnelle** - Les rôles reflètent les fonctions de travail (Admin, Créateur de Message, Approuveur de Message)
3. **Administration Simple** - Accorder l'accès en attribuant des rôles ; éviter la microgestion par utilisateur

# Permissions Système

Les permissions système suivent généralement des modèles CRUD avec quatre options :

- **Voir** - Lire ou parcourir des messages et des rapports
- **Créer** - Créer ou ajouter une alerte de message
- **Mettre à Jour** - Éditer ou modifier une alerte de message brouillon
- **Supprimer** - Supprimer ou retirer une alerte de message brouillon

**Permissions CBC de Base :**

- `CREATE_CBC_MESSAGE` - Créer de nouveaux messages de diffusion
- `VIEW_CBC_MESSAGE` - Voir les messages existants et leur statut
- `UPDATE_CBC_MESSAGE` - Modifier le contenu du message ou les paramètres de diffusion
- `DELETE_CBC_MESSAGE` - Supprimer des messages et annuler des diffusions

Attribuez ces permissions aux rôles en fonction des responsabilités de sécurité publique de votre organisation.

Pour des informations complètes sur la gestion des rôles et des permissions des utilisateurs, voir Utilisateurs et Rôles.

# Gestion des Comptes Utilisateurs

### Première Connexion

Lorsqu'un nouvel utilisateur est configuré dans le système, il reçoit des identifiants de connexion par l'Administrateur Système :

1. Accédez à l'URL de connexion OWL fournie par votre Administrateur Système
2. Entrez votre nom d'utilisateur et votre mot de passe
3. Si 2FA est activé pour votre compte, vous serez invité à le configurer (voir la section 2FA ci-dessus)

**Meilleure Pratique :** Les utilisateurs devraient changer leur mot de passe initial immédiatement après la première connexion pour des raisons de sécurité. Accédez aux paramètres de votre profil utilisateur pour changer votre mot de passe.

Pour plus de détails sur les flux d'authentification, voir Flux d'Authentification et Contrôles Administratifs.

**Réinitialisation de Mot de Passe**

Les utilisateurs peuvent réinitialiser leurs propres mots de passe :

1. Depuis la page de connexion du système, sélectionnez "Mot de Passe Oublié"
2. Suivez le processus de vérification par e-mail
3. Définissez un nouveau mot de passe

**Réinitialisation de Mot de Passe Administrateur :**

Les Administrateurs Système peuvent réinitialiser les mots de passe pour les utilisateurs depuis la page **Utilisateurs et Rôles → Utilisateurs** :

1. Accédez à **Utilisateurs et Rôles → Utilisateurs**
2. Sélectionnez l'utilisateur nécessitant une réinitialisation de mot de passe
3. Cliquez sur **Réinitialiser le Mot de Passe**
4. Le système génère un mot de passe temporaire sécurisé aléatoire (12 caractères)
5. Le mot de passe temporaire est affiché à l'administrateur
6. Fournissez le mot de passe temporaire à l'utilisateur via un canal sécurisé (téléphone ou en personne, jamais par e-mail)
7. L'utilisateur peut se connecter avec le mot de passe temporaire

8. **Important :** L'utilisateur doit changer ce mot de passe immédiatement après s'être connecté

Pour plus de détails, voir Utilisateurs et Rôles.

**Ajout de Nouveaux Utilisateurs Système**

Les Administrateurs Système peuvent ajouter de nouveaux utilisateurs en naviguant vers **Utilisateurs et Rôles → Utilisateurs** :

Informations requises pour chaque nouvel Utilisateur Enregistré :

- Prénom
- Nom de Famille
- E-mail
- Mot de Passe (temporaire)
- Numéro de Mobile
- Rôle(s)

**Meilleure Pratique :** Mettez en œuvre un Formulaire de Demande de Nouvel Utilisateur Système qui s'aligne avec les politiques et procédures existantes de votre organisation. N'ajoutez des utilisateurs qu'après approbation appropriée.

**Modification des Profils Utilisateurs**

Les Administrateurs Système peuvent changer les permissions et l'accès pour les utilisateurs existants :

1. Accédez à **Utilisateurs et Rôles → Utilisateurs**
2. Sélectionnez l'utilisateur à modifier
3. Modifiez les rôles assignés si nécessaire
4. Enregistrez les modifications

**Suppression des Utilisateurs Système**

Les Administrateurs Système peuvent supprimer des utilisateurs depuis **Utilisateurs et Rôles → Utilisateurs**.

**Avertissement :** Une fois qu'un utilisateur a été supprimé du système OWL, l'action ne peut pas être annulée. Faites très attention avant d'exécuter cette tâche.

**Visualisation de Tous les Utilisateurs Système**

À tout moment, l'Administrateur Système peut voir une liste de tous les Utilisateurs Enregistrés et voir quelles permissions et rôles chacun a en naviguant vers **Utilisateurs et Rôles → Utilisateurs**.

# Historique des Messages d'Alerte

Le système OWL conserve un enregistrement de tous les messages d'alerte envoyés à des fins d'audit et pour soutenir l'examen opérationnel.

**Accéder à l'Historique des Messages :**

Accédez à **Gérer les Diffusions Cellulaires** pour voir :

- Tous les messages précédemment envoyés
- Statut des messages (actif, expiré, annulé)
- Horodatage de la transmission
- Utilisateurs initiateurs et approbateurs
- Détails de ciblage géographique
- Contenu et paramètres du message

Cela aide à :

- Auditer le système
- Examiner les procédures opérationnelles utilisées par le NDMO et d'autres agences d'alerte
- Comprendre l'efficacité des messages
- Exigences de conformité et de reporting

# Visualisation de la Carte de Diffusion Cellulaire

La Carte de Diffusion Cellulaire fournit une **interface visuelle** pour planifier les zones de couverture géographique et identifier quelles tours cellulaires diffuseront des alertes.

## Accéder à la Carte

Naviguez vers :

Ou directement :

**Autorisation Requise :** `VIEW_CBC_MESSAGE` ou `CREATE_CBC_MESSAGE`

## Fonctionnalités de la Carte

La carte affiche :

**Emplacements des Tours Cellulaires :**

- **Icônes de Tours** - Positions des sites cellulaires affichées sous forme d'icônes de tours cellulaires
- **Indicateurs de Secteur** - Flèches directionnelles montrant l'orientation du secteur de l'antenne
- **Noms des Tours** - Identifiants et noms des sites
- **Données de Couverture** - Chargées à partir du fichier `cellSites.txt`

**Outils de Dessin :**

- **Outil Polygone** - Dessiner des zones de couverture personnalisées
- **Outil Cercle** - Créer des zones de diffusion circulaires
- **Outil Rectangle** - Définir des zones de couverture rectangulaires

**Analyse de Couverture :**

- **Liste des Sites** - Tableau montrant tous les sites cellulaires dans la zone dessinée
- **Gestion des Zones** - Enregistrer et charger des zones de couverture prédéfinies
- **Nombre de Sites** - Compte en temps réel des tours dans la zone de couverture

# Utilisation de la Carte

### Étape 1 : Charger les Données des Sites Cellulaires

La carte charge automatiquement les emplacements des tours cellulaires à partir de `/cellSites.txt` lors du chargement de la page. Ce fichier contient :

- **lat/lng** - Coordonnées de la tour
- **site_name** - Identifiant
- **sectors** - Directions des antennes en degrés

### Étape 2 : Dessiner la Zone de Couverture

Sélectionnez un outil de dessin :

1. Cliquez sur **Polygone** pour dessiner des formes personnalisées
    - Cliquez pour ajouter des points
    - Double-cliquez pour terminer
    - Crée des limites de couverture précises
2. Cliquez sur **Cercle** pour dessiner des zones circulaires
    - Cliquez sur le point central
    - Faites glisser pour définir le rayon
    - Utile pour des alertes rapides basées sur le rayon
3. Cliquez sur **Rectangle** pour dessiner des zones rectangulaires
    - Cliquez sur un coin
    - Faites glisser vers le coin opposé
    - Efficace pour une couverture alignée sur une grille

### Étape 3 : Réviser les Sites Couvert

Après le dessin, le système :

- Calcule quels sites cellulaires se trouvent dans la zone
- Affiche la liste dans le tableau en dessous de la carte
- Montre les noms et coordonnées des sites
- Fournit le compte des tours affectées

**Tableau d'Exemple de Couverture :**

**Étape 4 : Exporter les Données de Couverture**

Utilisez les données de couverture pour :

- **Identifier les codes de zone de suivi** pour la diffusion
- **Estimer la portée de l'alerte** en fonction du nombre de tours
- **Planifier la réponse d'urgence** en comprenant les zones affectées
- **Enregistrer les définitions de zone** pour une utilisation future

# Intégration du Flux de Travail

La visualisation de la carte s'intègre à la création de diffusion :

**Planification d'une Alerte :**

1. **Ouvrir la Carte CBC** pour visualiser la zone affectée
2. **Dessiner la zone de couverture** en utilisant l'outil polygone
3. **Réviser les sites couverts** et ajuster les limites
4. **Noter les codes de zone de suivi** pour les sites cellulaires affectés
5. **Créer un message CBC** avec un ciblage approprié

**Exemple : Alerte de Tsunami Côtière**

# Gestion des Zones

**Enregistrement des Zones :**

Des zones de couverture prédéfinies peuvent être enregistrées pour des scénarios courants :

- **Région Côtière** - Pour les avertissements de tsunami/onde de tempête
- **Zones d'Incendie de Forêt** - Pour les alertes d'évacuation
- **Centres Urbains** - Pour les alertes AMBER dans les zones peuplées
- **Corridors Autoroutiers** - Pour les alertes de trafic/météo

**Format des Données de Zone :**

Les zones sont stockées dans `site_data.json` :

```
[
  {
    "area": "Coastal Region",
    "polygon": [
      {
        "coord": [
          {"lat": -33.8688, "lng": 151.2093},
          {"lat": -33.8650, "lng": 151.2070},
          {"lat": -33.8600, "lng": 151.2150},
          {"lat": -33.8688, "lng": 151.2093}
        ]
      }
    ]
  }
]
```

**Chargement des Zones Enregistrées :**

1. Sélectionnez la zone dans le menu déroulant
2. Le polygone s'affiche automatiquement sur la carte
3. La liste des sites se met à jour pour montrer les tours couvertes
4. Modifiez le polygone si nécessaire

# Configuration de la Carte

**API Google Maps :**

La carte nécessite une clé API Google Maps configurée dans l'environnement :

```
VITE_GOOGLE_API_KEY=your_google_maps_api_key_here
```

**Données des Sites Cellulaires :**

Mettez à jour `public/cellSites.txt` avec les emplacements des tours cellulaires de votre réseau :

```
# Format : lat,lng,name,sector1,sector2,sector3
-33.8688,151.2093,SYD_CENTRAL,0,120,240
-33.8650,151.2070,SYD_HARBOUR,45,165,285
```

Obtenez les coordonnées des sites cellulaires à partir de :

- Outils de planification de réseau
- Registres d'installation de tours
- Enquêtes sur le terrain avec GPS
- Configuration du contrôleur de réseau radio (RNC)

**Style de Carte :**

Des styles de carte personnalisés peuvent être configurés dans `mapStyles.js` pour :

- Mettre en évidence les services d'urgence
- Afficher les caractéristiques topographiques
- Émettre des centres de population
- Correspondre à la marque de l'organisation

# Cas d'Utilisation

**Planification d'Urgence :**

- Prévoyez des zones de couverture pour des zones de danger connues
- Testez différentes stratégies de ciblage
- Estimez la portée de la population
- Coordonnez-vous avec les services d'urgence

**Vérification des Alertes :**

- Confirmez que le message atteindra la zone prévue
- Identifiez les lacunes de couverture
- Évitez de suralerter les régions adjacentes
- Validez les codes de zone de suivi

**Analyse du Réseau :**

- Visualisez la distribution des tours cellulaires
- Identifiez les chevauchements de couverture
- Planifiez les déploiements de tours pour une meilleure couverture d'alerte
- Optimisez l'efficacité de la diffusion

# Meilleures Pratiques

Basé sur l'expérience avec des clients du monde entier, Omnitouch recommande les meilleures pratiques suivantes pour tous les déploiements OWL.

**Contenu du Message :**

- Gardez les messages concis et clairs (moins de 360 caractères pour les alertes d'une seule page)
- Utilisez des lettres majuscules pour les alertes d'urgence (améliore la lisibilité)
- Incluez des éléments d'action spécifiques ("Évacuez immédiatement", "Cherchez un abri")
- Évitez le jargon technique
- Testez les messages avec des appareils réels avant les urgences

**Support Linguistique :**

- Fournissez toujours des messages dans la langue principale de la région
- Incluez des langues supplémentaires pour les zones multiculturelles
- Assurez-vous que les traductions sont culturellement appropriées

- Testez les caractères spéciaux et le support Unicode

**Ciblage Géographique :**

- Utilisez les plus petites zones de suivi nécessaires pour éviter la fatigue d'alerte
- Tenez compte de la densité de population lors de la définition de la période de répétition
- Testez le ciblage géographique avant les urgences
- Maintenez une documentation précise des zones de suivi

**Tests :**

- Utilisez des identifiants de message de test (4379, 4380-4381) pour les exercices
- Planifiez des tests réguliers du système
- Vérifiez que l'intégration du CBC fonctionne
- Formez le personnel sur les procédures d'urgence

**Fatigue d'Alerte :**

- Utilisez uniquement pour de véritables urgences
- Évitez de surdiffuser
- Définissez des périodes de répétition appropriées
- Utilisez les niveaux de gravité de manière appropriée

# Sécurité des Comptes Utilisateurs

**Sécurité des Jetons 2FA :**

- Protégez les jetons 2FA physiques comme les cartes d'accès au bâtiment
- Signalez immédiatement les jetons perdus ou volés
- Testez régulièrement 2FA pour vous assurer qu'il fonctionne
- Enregistrez les codes de sauvegarde lors de la configuration de 2FA et conservez-les en toute sécurité hors ligne
- Contactez l'Administrateur Système si vous avez besoin d'une réinitialisation de 2FA

Pour plus d'informations sur la configuration et la récupération de 2FA, voir Authentification à Deux Facteurs.

# Maintenance des Données

**Zones Ciblées Prédéfinies :**

À mesure que les frontières géographiques changent, que le développement se produit et que les zones de risque se déplacent, il est nécessaire de réviser les Zones Ciblées Prédéfinies. Omnitouch suggère que ces données soient révisées **annuellement par le NDMO**, avec le soutien d'autres agences d'alerte lorsque cela est applicable.

Les mises à jour des frontières peuvent être définies en utilisant plusieurs plateformes SIG courantes ou Google Earth, puis fournies par e-mail à l'équipe des opérations Omnitouch qui effectuera les modifications dans le système.

**Modèles de Message Prédéfinis :**

Les modèles de message prédéfinis doivent être révisés **au moins annuellement par le NDMO**, avec le soutien d'autres agences d'alerte lorsque cela est applicable, pour s'assurer que :

- Le contenu du message reflète toujours avec précision le danger et l'appel à l'action
- Les informations de contact et les instructions sont à jour
- Les traductions linguistiques restent précises
- Le ton et l'urgence du message sont appropriés
- Les références aux services d'urgence ou aux procédures sont à jour

Les mises à jour peuvent être fournies par e-mail à l'équipe des opérations Omnitouch qui effectuera les modifications dans le système.

**Données des Sites Cellulaires :**

- Révisez et mettez à jour les données des sites cellulaires chaque fois que des changements de réseau se produisent
- Fréquence de mise à jour typique : mensuelle ou trimestrielle

- Coordonnez-vous avec les équipes de planification de réseau MNO
- Vérifiez l'exactitude après des mises à jour ou des expansions majeures du réseau

# Architecture du Système

Tous les produits Omnitouch sont conçus pour prendre en charge des déploiements géographiquement distribués.

**Options de Déploiement**

Tous les composants peuvent fonctionner comme :

- **Conteneurs (K8s)** - Déploiements conteneurisés orchestrés par Kubernetes
- **Machines Virtuelles** - VMware, Proxmox, HyperV
- **Cloud Privé** - Infrastructure cloud sur site
- **Cloud Public** - AWS, GCP
- **Matériel Dédié** - Déploiement matériel direct

**Architecture Distribuée**

L'architecture distribuée permet :

- Aux Bureaux Locaux de Gestion des Catastrophes d'accéder au CBE OWL et de distribuer des messages même si une région devient isolée du réseau national
- Plusieurs Entités de Diffusion Cellulaire et plusieurs Centres de Diffusion Cellulaire par opérateur/par pays
- Aux agences locales de réponse aux catastrophes (Gouvernement Municipal, Police, Pompiers, etc.) d'émettre des alertes à leur région même si le principal NDMO (Bureau National de Gestion des Catastrophes) perd l'accès

Cela est particulièrement précieux lorsque les MNO ont distribué leur réseau cellulaire avec des ressources BSC/MME locales.

**Mise en Réseau CBE et CBC**

Pour garantir qu'il n'y a pas de connexion entre les réseaux concurrents, chaque MNO a une instance CBC séparée (non partagée).

Exigences de mise en réseau entre CBE et CBC :

- Tout le trafic est chiffré entre CBE & CBC
- Authentification basée sur des certificats mutuels
- Connectivité de CBE à CBC via TLS sur le port TCP 443
- Coordination requise entre les MNO et le NDMO ou l'agence hébergeant le CBE

**Considérations d'Accès**

L'agence hébergeant le CBE devra définir des procédures d'accès pour les utilisateurs finaux (c'est-à-dire Citrix, VPN, etc.), en gardant à l'esprit que le système doit être accessible dans des scénarios non idéaux tels que :

- Pannes à grande échelle des réseaux d'alimentation publics
- Pannes de réseau de télécommunications
- Catastrophes naturelles affectant l'infrastructure

# Exigences de Déploiement

## Exigences VM CBE (NDMO / Gouvernement)

**3x Machines Virtuelles :**

- **2x VM CBE** (Idéalement dans différents centres de données/zones de disponibilité)
- **1x VM de Surveillance**

Chaque VM nécessite :

- **Stockage :** 50 Go
- **CPU :** 2x CPU Virtuel

- **RAM :** 8 Go
- **OS :** OS de base fourni par Omnitouch
- **Mise en Réseau :** Autoriser le trafic vers les VM CBC sur le port TCP 443 pour le trafic TLS afin de contrôler les CBC

# Exigences VM CBC (MNO)

**3x Machines Virtuelles :**

- **2x VM CBC** (Idéalement dans différents centres de données/zones de disponibilité)
- **1x VM de Surveillance**

Chaque VM nécessite :

- **Stockage :** 50 Go
- **CPU :** 2x CPU Virtuel
- **RAM :** 8 Go
- **OS :** OS de base fourni par Omnitouch
- **Connectivité :** À chacun des Points d'Intégration dans le réseau cellulaire (MME, RNC, BSC)
- **Mise en Réseau :** Autoriser le trafic des VM CBE sur le port TCP 443 pour le trafic TLS à être contrôlé par les CBE

# Étapes d'Intégration

Le déploiement d'OWL implique les étapes suivantes :

1. **Source de nouveaux opérateurs** - Identifier les opérateurs de réseau mobile participants
2. **Admin de projet (configuration)** - Établir la gouvernance et l'administration du projet
3. **Sélectionner l'emplacement d'hébergement pour le CBE** - Déterminer où le CBE sera hébergé
4. **Définir les Utilisateurs & Flux/Procédures de Message** - Établir des flux de travail d'approbation et des rôles d'utilisateur

5. **Définir des Polygones pour les Zones Cibles et les Modèles de Message** - Préconfigurer des scénarios courants
6. **Allocations d'Adresse IP** - Allouer des adresses IP pour le NDMO et les MNO
7. **Configurer un VPN site-à-site** - Établir une connexion sécurisée à l'équipe Omnitouch
8. **Déployer les VM CBE au NDMO** - Installer l'Entité de Diffusion Cellulaire
9. **Déployer les VM CBC au MNO** - Installer le Centre de Diffusion Cellulaire chez chaque opérateur
10. **Configurer les Éléments Réseau** - Établir la connectivité au CBC à partir de l'équipement de réseau cellulaire
11. **Mise en Réseau entre les VM CBE et CBC** - Établir une communication sécurisée
12. **Mise en Réseau entre les VM CBC et les Éléments Réseau** - Connecter au BSC/MME/AMF
13. **Configurer l'accès API au NMS** - Configurer l'intégration des données des sites cellulaires
14. **Configuration de la surveillance & testée** - Vérifier la surveillance et l'alerte
15. **Vérification/Test des alertes de test** - Effectuer des tests système
16. **Test d'alerte publique** - Effectuer un test public de bout en bout

# Limites de Caractères

Les messages de Diffusion Cellulaire ont des limites strictes de caractères basées sur l'encodage :

**Encodage GSM 7 bits (anglais, caractères latins de base) :**

- Page unique : 93 caractères
- Multi-page : 15 pages × 93 = 1395 caractères maximum

**Encodage Unicode UCS-2 (scripts non latins, emojis) :**

- Page unique : 41 caractères

- Multi-page : 15 pages × 41 = 615 caractères maximum

**Plateforme OWL :**

- Texte du message limité à 500 caractères
- L'interface Web affiche le nombre de caractères restants et avertit lorsqu'elle approche des limites

# Surveillance et Journaux

L'activité de Diffusion Cellulaire est enregistrée à des fins d'audit :

- Événements de création, mise à jour et suppression de messages
- Utilisateurs initiateurs et approbateurs
- Horodatages et identifiants de message
- Réponses et erreurs de l'API CBC
- Détails de ciblage géographique

Accédez aux journaux via le Journal d'Activité ou des requêtes de base de données :

```
SELECT * FROM cbc
WHERE created >= '2025-01-01'
ORDER BY created DESC;
```

# Intégration avec les Appareils Mobiles

Les messages de Diffusion Cellulaire sont reçus par des appareils mobiles compatibles :

**Support des Appareils :**

- La plupart des smartphones depuis 2015 prennent en charge la Diffusion Cellulaire

- Les téléphones basiques peuvent avoir un support limité
- L'appareil doit être connecté au réseau (aucun crédit de données/SMS requis)
- Fonctionne même pendant la congestion du réseau lorsque les SMS échouent

**Expérience Utilisateur :**

- L'alerte s'affiche comme une notification en plein écran
- Un ton d'alerte unique se fait entendre
- L'alerte persiste jusqu'à ce qu'elle soit reconnue
- Aucun abonnement utilisateur requis
- Ne peut pas être bloquée par les utilisateurs pour les alertes présidentielles/extrêmes

**Tester la Réception des Appareils :**

Pour vérifier que les appareils peuvent recevoir des alertes :

1. Envoyez un message de test (identifiant 4379 ou 4380-4381)
2. Assurez-vous que l'appareil se trouve dans la zone de suivi cible
3. Vérifiez que l'appareil a la Diffusion Cellulaire activée dans les paramètres
4. Vérifiez avec plusieurs modèles d'appareils et versions d'OS

# Fonctionnalités Supplémentaires

La plateforme OWL peut être étendue avec des fonctionnalités optionnelles pour compléter la messagerie de Diffusion Cellulaire :

## SMS de Masse

Envoi de SMS réguliers à des individus pour des notifications supplémentaires.

- Souvent utilisé pour des alertes dégradées pour informer les gens que la menace immédiate est passée, de manière moins intrusive

- Beaucoup plus lent que la Diffusion Cellulaire mais peut inclure un mécanisme de confirmation pour vérifier que le message a été reçu
- Utile pour des communications de suivi ciblées

## Appels Vocaux pour Lignes Fixes

Appeler automatiquement des numéros de lignes fixes et jouer des messages d'avertissement d'urgence.

- Lecture de messages préenregistrés
- Rendu texte-à-parole du message d'avertissement d'urgence
- Atteint des populations sans téléphones mobiles
- Peut vérifier la livraison du message par l'achèvement de l'appel

## Publication Croisée sur les Réseaux Sociaux

Publier automatiquement des messages d'Avertissement d'Urgence sur des canaux de réseaux sociaux officiels.

- Étend la portée au-delà du réseau cellulaire
- Fournit une référence pour ceux qui ont manqué l'alerte initiale
- Permet un message prolongé au-delà des limites de caractères

## Transmission Automatique Radio/Télévision

Diffuser automatiquement des messages d'urgence via la radio et la télévision.

- Lecture de messages préenregistrés
- Rendu texte-à-parole du message d'avertissement d'urgence
- Atteint des populations pendant les pannes de réseau
- Complète la Diffusion Cellulaire pour une couverture complète

## Dispositifs d'Avertissement Externes

Le CBC OWL peut se connecter à une variété de sources externes :

- Plateformes de réseaux sociaux

- API publiques
- Systèmes d'appels vocaux
- Systèmes de diffusion radio
- Alarmes physiques et sirènes
- Signalisation électronique

Les options de personnalisation peuvent être explorées dans le cadre de la phase de conception.

# Personnalisation et Maintenance

## Tests Périodiques

Des tests périodiques de la solution devraient être effectués à intervalles réguliers pour garantir :

- La solution et tous les composants fonctionnent correctement
- Tout le personnel est familier avec les processus et procédures nécessaires pour émettre des Messages d'Avertissement d'Urgence
- Les points d'intégration restent opérationnels
- Les modèles de message sont actuels et efficaces

**Tests Recommandés :**

- Messages de test mensuels utilisant les identifiants 4379, 4380-4381
- Tests complets du système trimestriels incluant des flux de travail d'approbation
- Tests d'avertissement public annuels avec notification préalable
- Sessions de formation régulières pour les utilisateurs autorisés

## Maintenance des Données des Sites Cellulaires

Lorsque les opérateurs ajoutent ou suppriment des sites cellulaires, ou changent les Zones/Identifiants de Suivi des sites cellulaires, ces informations

doivent être partagées avec l'équipe Omnitouch pour garantir que les données de l'outil de cartographie restent précises.

**Intégration Automatique des Données**

OWL prend en charge le scraping automatique des données à partir de :

- Nokia NetAct
- Huawei U2000 / U2020
- ZTE NetNumen / ZXPOS
- Ericsson ENM

**Mises à Jour Manuelles des Données**

Alternativement, les données des sites cellulaires peuvent être fournies à l'équipe des opérations Omnitouch périodiquement par e-mail dans divers formats.

**Fréquence de Mise à Jour :** Révisez et mettez à jour les données des sites cellulaires chaque fois que des changements de réseau se produisent, typiquement mensuellement ou trimestriellement.

# Zones Ciblées Prédéfinies

À mesure que les frontières géographiques changent, que le développement se produit et que les zones de risque se déplacent, il est nécessaire de réviser les Zones Ciblées Prédéfinies utilisées dans l'étape de Ciblage.

**Révision Annuelle :** Les zones cibles prédéfinies doivent être révisées annuellement par le NDMO (Bureau National de Gestion des Catastrophes).

**Processus de Mise à Jour :** Les mises à jour des frontières peuvent être définies en utilisant plusieurs plateformes SIG courantes ou Google Earth, puis fournies à l'équipe des opérations Omnitouch.

# Modèles de Message Prédéfinis

**Révision Annuelle :** Les modèles de message prédéfinis doivent être révisés annuellement par le NDMO pour garantir :

- Le contenu du message reflète les procédures d'urgence actuelles
- Les traductions linguistiques sont précises
- Les identifiants de message sont appropriés
- Les informations de contact et les instructions sont à jour

**Processus de Mise à Jour :** Les mises à jour peuvent être fournies par e-mail à l'équipe des opérations Omnitouch.

# Flux d'Approbation des Messages

Différentes régions ont des exigences différentes concernant la manière dont elles émettent des messages et les flux d'approbation.

**Règle des Deux Personnes :** L'utilisation de la règle des deux personnes est préconisée chaque fois que cela est pratique pour garantir une supervision dans la soumission des messages.

**Rôles Utilisateurs Granulaires :** Des rôles utilisateurs individuels peuvent être configurés pour :

- Autoriser uniquement certains utilisateurs à envoyer des messages prédéfinis
- Restreindre le ciblage à des régions spécifiques
- Exiger des étapes d'approbation supplémentaires
- Minimiser le risque d'abus

# Maintenance de la Sécurité 2FA

L'équipe des opérations Omnitouch peut soutenir :

- Réinitialisation des jetons 2FA
- Réémission de jetons perdus/endommagés/expirés
- Audit de sécurité et gestion des jetons

**Procédures de Sécurité des Jetons :**

Lorsque des jetons sont émis, une procédure est détaillée pour les étapes à suivre si un jeton devient perdu ou non comptabilisé. La personne acceptant le

jeton doit suivre ces procédures pour garantir que le système n'est pas abusé.

**Actions Immédiates pour les Jetons Perdus :**

1. Signalez immédiatement le jeton perdu au personnel autorisé
2. Le jeton est désactivé dans le système
3. Une révision de sécurité est effectuée
4. Un nouveau jeton est émis après vérification de sécurité

## Intégration API Externe

Une suite complète d'APIs est disponible pour l'Entité de Diffusion Cellulaire afin de permettre aux systèmes tiers de s'intégrer et d'interagir avec le CBE.

**Capacités de l'API :**

- **Rapports/Surveillance** - Vérifier le statut et la portée des messages transmis
- **Création de Messages** - Créer et diffuser de nouveaux messages par programmation
- **Vérifications de Santé** - Vérifier périodiquement la santé du système avec la génération de trafic de message de test
- **Requêtes de Statut** - Récupérer le statut des messages, les statistiques de livraison et les métriques du système

Voir la section Référence API ci-dessous pour une documentation détaillée des points de terminaison.

# Référence API

Tous les points de terminaison CBC nécessitent une authentification et des permissions appropriées.

**Créer un Message :**

`PUT /crm/cbc/`

**Obtenir Tous les Messages :**

`GET /crm/cbc/`

**Mettre à Jour un Message :**

`PATCH /crm/cbc/{cbc_message_id}`

**Supprimer un Message :**

`DELETE /crm/cbc/{cbc_message_id}`

Voir la documentation Swagger à `/crm/docs/` pour des spécifications API détaillées.

# Recherche Globale

La fonctionnalité de Recherche Globale fournit une **interface de recherche unifiée** pour trouver rapidement des clients, des contacts, des services, de l'inventaire et des sites dans l'ensemble de la base de données OmniCRM.

Voir aussi : Clients, Inventaire, Gestion des Services.

## Accéder à la Recherche Globale

**Depuis n'importe où dans le CRM :**

Cliquez sur l'icône de recherche dans la barre de navigation supérieure ou naviguez vers :

La page de recherche globale apparaît avec une grande zone de recherche et des options de filtre.

## Comment Ça Fonctionne

La recherche globale effectue une **recherche inter-entités** à travers cinq types de données :

**Ce qui est recherché :**

1. **Clients** - Nom du client
2. **Contacts** - Prénom, nom de famille, adresse e-mail, numéro de téléphone
3. **Sites** - Nom du site
4. **Inventaire** - Numéros de série, ICCIDs, identifiants (itemtext1, itemtext2)
5. **Services** - Nom du service, UUID du service

**Comportement de Recherche :**

- **Correspondance partielle** - Recherche des termes contenant votre requête (par exemple, "Smith" correspond à "John Smith" et "Smithson")

- **Insensible à la casse** - "john" correspond à "John", "JOHN" et "john"
- **Multiples entités** - Une seule recherche renvoie des résultats de tous les types d'entités
- **Résultats paginés** - Affiche 10 résultats par page par défaut

# Effectuer une Recherche

## Recherche de Base

1. Entrez votre terme de recherche dans la zone de recherche
2. Cliquez sur **"Rechercher"** ou appuyez sur Entrée

**Exemples de termes de recherche :**

- Nom du client : `"Acme Corp"`
- Numéro de téléphone : `"+1234567890"` ou `"1234567890"`
- E-mail : `"john@example.com"` ou `"john"`
- Numéro de série : `"ICCID8944"` ou juste `"8944"`
- UUID du service : `"123e4567-e89b"`

## Inclure les Comptes Clôturés

Par défaut, la recherche ne renvoie que des résultats des comptes clients **Ouverts**.

Pour rechercher dans tous les comptes, y compris ceux clôturés :

1. Cochez la case **"Inclure les Comptes Clôturés"**
2. Cliquez à nouveau sur **"Rechercher"**

Cela recherchera :

- Clients avec `customer_status = "Closed"`
- Contacts, services, sites et inventaire liés à des clients clôturés

**Cas d'utilisation pour la recherche de comptes clôturés :**

- Trouver des dossiers clients historiques
- Localiser des équipements de services désactivés
- Rechercher d'anciens numéros de téléphone ou services
- Récupérer des données clients pour réactivation

# Comprendre les Résultats de Recherche

## Format d'Affichage des Résultats

Les résultats sont affichés dans une liste déroulante montrant :

> John Smith ID Client : 123 Type : client
>
> John Smith (Contact) ID Client : 123 Type : contact
>
> Mobile - +44 7700 900123 ID Client : 123 Type : service

**Chaque résultat montre :**

- **Nom/Titre** - L'identifiant principal (lien cliquable)
- **ID Client** - Le client parent auquel cela appartient
- **Type** - Le type d'entité (client, contact, site, inventaire, service)

## Types de Résultats Expliqués

**Résultats Clients :**

Cliquer ouvre la page de vue d'ensemble du client montrant tous les détails, services, contacts, etc.

**Résultats Contacts :**

Cliquer ouvre la page du client avec l'onglet Contacts actif, faisant défiler vers le contact spécifique.

**Résultats Sites :**

Cliquer ouvre la page du client avec l'onglet Sites actif.

**Résultats Inventaire :**

Cliquer ouvre la page du client avec l'onglet Inventaire actif. Si l'inventaire n'est pas attribué (pas de customer_id), il se lie à la liste principale de l'inventaire.

**Résultats Services :**

Cliquer ouvre la page du client avec l'onglet Services actif, mettant en évidence le service spécifique.

# Navigation à partir des Résultats

Tous les résultats de recherche sont des **liens cliquables** qui naviguent directement vers la page pertinente :

**Modèle de Lien :**

- `/customers/{customer_id}` - Dossiers clients
- `/customers/{customer_id}#4` - Contacts (onglet 4)
- `/customers/{customer_id}#2` - Sites (onglet 2)
- `/customers/{customer_id}#8` - Inventaire (onglet 8)
- `/customers/{customer_id}#3` - Services (onglet 3)
- `/inventory-items-list` - Inventaire non attribué

Le fragment de hachage (`#`) sélectionne automatiquement l'onglet correct lorsque la page client se charge.

## Pagination

Les résultats sont paginés avec 10 éléments par page :

> Affichage des résultats 11-20 sur 47

Naviguez à travers les pages en utilisant :

- Boutons **Précédent/Suivant**
- **Numéros de page** - Cliquez sur une page spécifique
- **Clavier** - Flèches gauche/droite (si implémenté)

# Scénarios de Recherche Courants

**Scénario 1 : Trouver un Client par Téléphone**

L'utilisateur appelle, fournit un numéro de téléphone.

> Résultats : • John Smith (Contact) - ID Client : 123 • Mobile - 555-0123 (Service) - ID Client : 123

Cliquez sur l'un ou l'autre résultat pour accéder au compte client.

**Scénario 2 : Localiser une Carte SIM**

Le technicien doit trouver quel client a une carte SIM spécifique.

> Résultats : • 8944538000000001234 (Inventaire) - ID Client : 456

Cliquez sur le résultat pour voir l'attribution de la SIM, les détails du client.

**Scénario 3 : Trouver un Client Inactif**

Besoin de localiser un client qui a clôturé son compte il y a 6 mois.

> ☑ Inclure les Comptes Clôturés
>
> Résultats : • Acme Corporation (client) - ID Client : 789

**Scénario 4 : Recherche par E-mail**

Le client envoie un e-mail au support, le personnel doit trouver son compte.

> Résultats : • John Smith (Contact) - ID Client : 123

**Scénario 5 : Trouver un Service par UUID**

Le journal de provisionnement montre l'UUID du service, besoin de trouver quel client.

> Résultats : • Mobile - +44 7700 900123 (Service) - ID Client : 456

# Conseils de Recherche

**Pour de Meilleurs Résultats :**

- **Utilisez des termes partiels** - "Smith" est mieux que "John Smith" pour des résultats plus larges
- **Essayez des variations** - Si "John" ne fonctionne pas, essayez le téléphone ou l'e-mail
- **Incluez les comptes clôturés** - Lors de la recherche de données historiques
- **Soyez spécifique pour l'équipement** - Utilisez des numéros de série complets pour l'inventaire
- **Recherchez l'UUID du service** - Lorsque d'autres identifiants ne sont pas connus

**Ce qui est recherché (par Entité) :**

**Clients :**

- Nom du client uniquement (pas d'adresse, notes ou autres champs)

**Contacts :**

- Prénom
- Nom de famille
- Adresse e-mail
- Numéro de téléphone

**Sites :**

- Nom du site uniquement

**Inventaire :**

- itemtext1 (généralement ICCID, numéro de série, adresse MAC)
- itemtext2 (généralement IMSI, identifiant secondaire)
- *Remarque : Ne recherche pas itemtext3-20 ou notes d'inventaire*

**Services :**

- Nom du service
- UUID du service

**Ce qui n'est pas recherché :**

- Adresses des clients
- Notes des clients
- Descriptions des transactions
- Détails des factures
- Journaux de provisionnement
- Entrées du journal d'activité
- Notes d'inventaire (au-delà de itemtext1/2)

# Référence API

## Point de Terminaison de Recherche Globale

```
GET /utilities/search_everything?
search=Smith&page=1&per_page=10&search_closed_records=false
Authorization: Bearer <token>
```

**Paramètres de Requête :**

- `search` (obligatoire) - Le terme de recherche

- `page` (optionnel) - Numéro de page (par défaut : 1)
- `per_page` (optionnel) - Résultats par page (par défaut : 10)
- `search_closed_records` (optionnel) - Inclure les comptes clôturés (par défaut : false)

**Réponse :**


```
{
  "data": [
    {
      "id": 123,
      "name": "John Smith",
      "customer_id": 123,
      "type": "customer"
    },
    {
      "id": 456,
      "name": "John Smith",
      "customer_id": 123,
      "type": "contact"
    },
    {
      "id": 789,
      "name": "Mobile - +44 7700 900123",
      "customer_id": 123,
      "type": "service"
    }
  ],
  "pagination": {
    "current_page": 1,
    "per_page": 10,
    "total_pages": 5,
    "total_items": 47
  }
}
```


**Logique de Recherche (Backend) :**

Le backend effectue une UNION SQL à travers toutes les tables d'entités :

```sql
-- Clients
SELECT customer_id AS id,
       customer_name AS name,
       customer_id,
       'customer' AS type
FROM customer
WHERE customer_name LIKE '%Smith%'
  AND customer_status = 'Open'

UNION ALL

-- Contacts
SELECT contact_id AS id,
       CONCAT(contact_firstname, ' ', contact_lastname) AS name,
       customer_id,
       'contact' AS type
FROM customer_contact
WHERE (contact_firstname LIKE '%Smith%' OR
       contact_lastname LIKE '%Smith%' OR
       contact_email LIKE '%Smith%' OR
       contact_phone LIKE '%Smith%')

UNION ALL

-- Sites
SELECT site_id AS id,
       site_name AS name,
       customer_id,
       'site' AS type
FROM customer_site
WHERE site_name LIKE '%Smith%'

UNION ALL

-- Inventaire
SELECT inventory_id AS id,
       itemtext1 AS name,
       customer_id,
       'inventory' AS type
FROM inventory
WHERE itemtext1 LIKE '%Smith%' OR
      itemtext2 LIKE '%Smith%'
```

```
UNION ALL

-- Services
SELECT service_id AS id,
       service_name AS name,
       customer_id,
       'service' AS type
FROM customer_service
WHERE service_name LIKE '%Smith%' OR
      service_uuid LIKE '%Smith%'
```

Les résultats sont ensuite paginés et renvoyés.

# Considérations de Performance

**Performance de Recherche :**

- Les recherches utilisent des requêtes LIKE avec des jokers (`%term%`)
- Aucun index de texte intégral actuellement mis en œuvre
- Les grandes bases de données (>100k clients) peuvent connaître des recherches plus lentes
- Résultats limités à 10 par page pour des raisons de performance

**Conseils d'Optimisation :**

- Soyez spécifique avec les termes de recherche pour réduire l'ensemble des résultats
- Utilisez le filtre de compte clôturé pour réduire la portée de la recherche
- Envisagez d'ajouter des index de base de données sur les champs fréquemment recherchés

# Dépannage

**Aucun résultat trouvé (mais l'enregistrement existe)**

- **Cause :** Le terme de recherche ne correspond pas au format de données stockées

- **Exemples :**
  - Téléphone stocké sous "+44 7700 900123", rechercher "07700900123" ne correspondra pas
  - E-mail stocké sous "<john.smith@example.com>", rechercher "john" ne correspondra pas
- **Solution :** Essayez des variations, utilisez des correspondances partielles qui existent définitivement

**Recherche trop lente**

- **Cause :** Grande base de données, requête complexe à travers plusieurs tables
- **Solution :**
  - Utilisez des termes de recherche plus spécifiques
  - Limitez-vous aux comptes ouverts uniquement (décochez les comptes clôturés)
  - Contactez l'administrateur concernant l'indexation de la base de données

**Les résultats lient au mauvais client**

- **Cause :** Plusieurs clients/contacts avec le même nom
- **Solution :** Utilisez l'ID Client pour différencier, ou recherchez par identifiant unique (e-mail, téléphone)

**Les comptes clôturés n'apparaissent pas**

- **Cause :** Case ☐ cocher "Inclure les Comptes Clôturés" non cochée
- **Solution :** Cochez la case et recherchez à nouveau

# Documentation Connexe

- `basics_customers` - Gestion des clients
- `basics_navigation` - Navigation générale
- `administration_inventory` - Recherches d'inventaire

# Système de Recharge et de Ravitaillement

Le système de Ravitaillement d'OmniCRM fournit un **portail de recharge prépayée en libre-service** pour que les clients ajoutent du crédit ou prolongent la validité du service via le Portail de Self-Care. Cette fonctionnalité est couramment utilisée pour :

- **Services de données mobiles** - Cartes SIM prépayées et services uniquement de données
- **Services de point d'accès** - Dongles WiFi et dispositifs Internet portables
- **Services sans fil fixes** - Accès Internet prépayé

Voir aussi : Provisionnement de Carte SIM pour des informations sur le provisionnement des services mobiles et la gestion de l'inventaire des SIM.

## Aperçu

Le système de ravitaillement permet aux clients d'acheter des jours supplémentaires de service à travers un processus de paiement simplifié et multi-étapes avec un traitement de paiement intégré via Stripe.

**Caractéristiques Clés :**

- Portail client en libre-service (aucune intervention du personnel requise)
- Sélection de durée flexible (1 à 30 jours)
- Affichage de l'utilisation en temps réel avant l'achat
- Traitement de paiement sécurisé alimenté par Stripe
- Remboursements automatiques si le ravitaillement échoue
- Génération de factures et de transactions
- Intégration du système de provisionnement pour l'activation du service

# Accéder au Portail de Ravitaillement

Le portail de ravitaillement est accessible via une **URL publique** que les clients peuvent visiter sans se connecter au CRM :

**Comment les Clients y Accèdent :**

- Lien direct envoyé par SMS lorsque le solde est bas
- Code QR sur les documents imprimés
- Lien sur le portail de self-care
- Partagé via le support client

Le portail détecte automatiquement le service du client en fonction de son adresse IP ou de son IMSI.

# Processus de Ravitaillement

Le flux de ravitaillement se compose de **4 étapes** :

## Étape 1 : Sélection des Données

Les clients sélectionnent combien de jours de service ils souhaitent acheter.

**Interface :**

- **Contrôle de curseur** - Sélectionnez de 1 à 30 jours
- **Calcul de prix en direct** - Affiche le coût total en fonction de la sélection
- **Affichage de la date d'expiration** - Calcule et affiche quand le service expirera
- **Affichage de l'utilisation actuelle** - Affiche le solde restant/l'expiration avant le ravitaillement

**Exemple d'Affichage :**

**Configuration des Tarifs :**

- Le prix par jour est configuré via la variable d'environnement `VITE_TOPUP_PRICE_PER_DAY`
- Par défaut : 10 $ USD par jour
- La devise est définie via `VITE_CURRENCY_CODE`

# Étape 2 : Informations de Facturation

Les clients fournissent leurs coordonnées pour la transaction :

- **Prénom**
- **Nom**
- **Adresse E-mail**

Ces informations sont utilisées pour :

- Génération de factures
- E-mail de reçu de paiement
- Enregistrements de transactions
- Traitement des remboursements (si nécessaire)

# Étape 3 : Paiement

Traitement de paiement sécurisé via **formulaires de paiement hébergés par le fournisseur** (Stripe Elements, PayPal SDK).

**Méthodes de Paiement Supportées :**

- Cartes de crédit (Visa, Mastercard, Amex)
- Cartes de débit
- Portefeuilles numériques (Apple Pay, Google Pay, PayPal) *si activé par le fournisseur de paiement*

**Fonctionnalités de Sécurité :**

- Intégration de fournisseur de paiement conforme PCI
- Aucune donnée de carte stockée dans OmniCRM
- Support d'authentification 3D Secure

- Transmission de paiement chiffrée

**Flux de Paiement :**

1. Formulaire de paiement sécurisé affiché avec saisie de carte
2. Le client entre les détails de paiement
3. Paiement traité via le fournisseur configuré
4. Carte débitée immédiatement
5. Succès/échec du paiement géré

> **REMARQUE**
>
> Si le paiement réussit mais que le provisionnement du ravitaillement échoue (par exemple, erreur réseau, OCS injoignable), le système initie automatiquement un **remboursement complet** vers le mode de paiement du client.

# Étape 4 : Finalisation

**Écran de Succès :**

> Votre service a été prolongé. Nouvelle date d'expiration : 17 Jan 2025
>
> Reçu envoyé à : <customer@example.com> ID de Transaction : TXN-123456

**Écran d'Échec :**

Si le ravitaillement échoue, le système affiche une erreur et traite automatiquement un remboursement :

> Nous n'avons pas pu compléter votre ravitaillement. Votre paiement a été remboursé.
>
> Erreur : Impossible de se connecter au système de facturation
>
> Veuillez réessayer ou contacter le support.

# Traitement Backend

Lorsque le client termine le paiement, les actions suivantes se produisent automatiquement :

### 1. Validation du Paiement

Le système valide :

- Le statut de l'Intent de Paiement est `succeeded`
- Le montant du paiement correspond aux jours sélectionnés (`days × price_per_day`)
- L'Intent de Paiement n'a pas été traité auparavant (évite le double ravitaillement)

### 2. Opération de Ravitaillement

```
- API endpoint: POST /oam/topup_dongle
- Valide service_uuid et IMSI
- Appelle OCS/CGRateS pour ajouter du solde
- Crée un job de provisionnement (play_topup_hotspot)
```

### 3. Création d'Enregistrements

Le système crée plusieurs enregistrements dans la base de données :

- **Enregistrement HotspotTopup** - Suit la transaction de ravitaillement
  - payment_intent_id
  - service_uuid
  - imsi
  - jours achetés
  - topup_amount
  - statut (Succès/Échoué/Remboursé)
- **Enregistrement de Transaction** - Transaction financière
  - Titre : "Ravitaillement Hotspot - 7 Jours"
  - Montant : topup_amount (positif)
  - Lié à service_id et customer_id

- **Enregistrement de Facture** - Facture de paiement
  - Contient la transaction de ravitaillement
  - Marqué comme payé immédiatement
  - Référence de paiement : Stripe payment_intent_id
- **Transaction de Paiement** - Transaction de crédit compensatoire
  - Titre : "Paiement pour [Titre de la Facture]"
  - Montant : topup_amount (négatif - crédit)
  - Lien entre le paiement de la facture et le compte client

**4. Job de Provisionnement**

Un job de provisionnement est créé avec le playbook `play_topup_hotspot` qui :

- Se connecte à l'API OCS/CGRateS
- Ajoute du solde au compte
- Prolonge la date d'expiration
- Crée une entrée dans le journal d'activité
- Envoie une notification de confirmation (si configuré)

L'API attend que le provisionnement soit terminé (polling avec des intervalles de 0,2s, max 25 itérations) avant de retourner le succès au client.

**5. Remboursement Automatique en Cas d'Échec**

Si une étape échoue après le paiement :

```
if topup_provisioning_failed:
    refund = stripe.Refund.create(
        payment_intent=payment_intent_id,
        reason='requested_by_customer'  # Remboursement
automatique du système
    )
    status_message = "Échec du Ravitaillement. Remboursement du
paiement..."
```

Le remboursement est traité automatiquement et le client est informé à l'écran.

# Points de Terminaison API

## Point de Terminaison de Ravitaillement

```
POST /oam/topup_dongle
Content-Type: application/json

{
  "service_uuid": "123e4567-e89b-12d3-a456-426614174000",
  "imsi": "310120123456789",
  "days": 7,
  "payment_intent_id": "pi_1234567890abcdef",
  "topup_amount": 70.00
}
```

**Réponse (Succès) :**

```
{
  "result": "OK",
  "status": 200,
  "provision_id": 456,
  "payment_intent_id": "pi_1234567890abcdef",
  "service_uuid": "123e4567-e89b-12d3-a456-426614174000",
  "invoice_id": 789
}
```

**Réponse (Échec) :**

```
{
  "result": "Failed",
  "Reason": "Délai de connexion OCS",
  "status": 500
}
```

**Vérifications de Validation :**

- Tous les champs requis présents (service_uuid, imsi, days, payment_intent_id, topup_amount)

- topup_amount correspond aux jours : `topup_amount × 100 == days × 1000` (en cents)
- L'Intent de Paiement existe dans Stripe
- Le montant de l'Intent de Paiement correspond : `payment_intent.amount == topup_amount × 100`
- Le statut de l'Intent de Paiement est `succeeded`
- L'Intent de Paiement n'a pas déjà été traité (vérifie la table `HotspotTopup`)

## Point de Terminaison d'Utilisation

Récupère l'utilisation actuelle et les informations de service pour le client :

```
GET /oam/usage
```

**Réponse :**

```
{
  "imsi": "310120123456789",
  "service": {
    "service_uuid": "123e4567-e89b-12d3-a456-426614174000",
    "service_name": "Données Mobiles - 0412345678",
    "service_status": "Actif"
  },
  "balance": {
    "expiry": "2025-01-10T23:59:59Z",
    "unlimited": true
  },
  "requestingIp": "203.0.113.45"
}
```

Ce point de terminaison utilise l'adresse IP de demande pour identifier automatiquement le service du client.

# Configuration

## Variables d'Environnement

Configurez-les dans le fichier `.env` d'OmniCRM-UI :

```
# Configuration du Portail de Ravitaillement
VITE_TOPUP_PRICE_PER_DAY=10
VITE_CURRENCY_CODE=AUD
VITE_SELF_CARE_NAME="YourCompany"

# Configuration de Paiement
VITE_PAYMENT_VENDORS="stripe"
VITE_STRIPE_PUBLISHABLE_KEY=pk_live_...
```

## Configuration de Stripe

Le système de ravitaillement utilise les Intents de Paiement Stripe :

1. **Activer les Intents de Paiement** dans votre tableau de bord Stripe
2. **Configurer le Webhook** pour recevoir des mises à jour de statut de paiement (optionnel mais recommandé)
3. **Configurer les méthodes de paiement** (cartes, portefeuilles, etc.)
4. **Mode test** - Utilisez des clés de test pour le développement

```
# Développement
VITE_STRIPE_PUBLISHABLE_KEY=pk_test_...

# Production
VITE_STRIPE_PUBLISHABLE_KEY=pk_live_...
```

## Configuration du Playbook

Le playbook de provisionnement `play_topup_hotspot.yaml` doit être configuré pour :

- Accepter la variable `days`
- Se connecter à l'API OCS/CGRateS
- Ajouter du solde au compte
- Mettre à jour la date d'expiration du service

Exemple de structure de playbook :

```
- name: Ravitaillez le service hotspot
  hosts: localhost
  tasks:
    - name: Ajouter du solde à OCS
      uri:
        url: "{{ ocs_api_url }}/add_balance"
        method: POST
        body:
          imsi: "{{ imsi }}"
          days: "{{ days }}"
          service_uuid: "{{ service_uuid }}"
```

# Notifications de Solde Bas

Le système peut envoyer des notifications automatiques lorsque le solde du client est bas :

**Notifications SMS :**

Lorsqu'elles sont déclenchées par des événements OCS (`Action_Balance_Low`, `Action_Balance_Out`, `Action_Balance_Expired`) :

**Notifications par E-mail :**

Configurées dans les plans d'action OCS/CGRateS pour envoyer des alertes de solde.

**Déclencheurs de Notification :**

- `Action_Balance_Low` - Solde en dessous du seuil (par exemple, 2 jours restants)

- `Action_Balance_Out` - Solde épuisé
- `Action_Balance_Expired` - Service expiré

Chaque notification inclut le lien vers le portail de ravitaillement pour un accès facile par le client.

# Dépannage

## Problèmes Courants

**"Système de paiement indisponible"**

- **Cause :** La bibliothèque Stripe a échoué à se charger ou clé publique invalide
- **Solution :**
  - Vérifiez que `VITE_STRIPE_PUBLISHABLE_KEY` est correctement défini
  - Vérifiez que le compte Stripe est actif
  - Vérifiez la console du navigateur pour des erreurs JavaScript

**"Ravitaillement échoué. Remboursement du paiement..."**

- **Cause :** Le job de provisionnement a échoué (OCS injoignable, erreur de playbook, etc.)
- **Solution :**
  - Vérifiez les journaux de provisionnement : `GET /crm/provision/provision_id/<id>`
  - Vérifiez que l'API OCS/CGRateS est accessible
  - Examinez le playbook `play_topup_hotspot.yaml` pour des erreurs
  - Vérifiez les journaux Ansible

**"Intent de paiement déjà traité"**

- **Cause :** Le client tente de réutiliser le même paiement (par exemple, rafraîchir après succès)
- **Solution :** Cela fonctionne comme prévu pour éviter la double facturation. Le client doit commencer un nouveau ravitaillement si nécessaire.

**"Le montant de l'intent de paiement ne correspond pas"**

- **Cause :** Mismatch entre le calcul de l'UI et la validation backend
- **Solution :**
  - Vérifiez que `VITE_TOPUP_PRICE_PER_DAY` correspond aux attentes du backend (par défaut 10 $)
  - Vérifiez que la configuration de la devise est cohérente
  - Effacez le cache du navigateur et réessayez

## Surveillance des Ravitaillements

**Voir les Enregistrements de Ravitaillement :**

Interrogez la table `HotspotTopup` pour voir toutes les tentatives de ravitaillement :

```
SELECT
  hotspot_topup_id,
  service_uuid,
  days,
  topup_amount,
  status,
  payment_intent_id,
  created
FROM hotspot_topup
WHERE status = 'Failed'
ORDER BY created DESC;
```

**Vérifiez le Statut de Provisionnement :**

```
GET /crm/provision/provision_id/<provision_id>
```

Affiche le statut détaillé du job de provisionnement de ravitaillement.

**Tableau de Bord Stripe :**

Surveillez les paiements, remboursements et transactions échouées dans votre tableau de bord Stripe à <https://dashboard.stripe.com>

# Considérations de Sécurité

**Sécurité des Paiements :**

- Toutes les données de carte traitées par Stripe (conforme PCI Niveau 1)
- Aucune donnée de paiement sensible stockée dans la base de données OmniCRM
- Les Intents de Paiement empêchent les charges non autorisées
- Validation des montants côté client et serveur

**Prévention de Fraude :**

- La détection des Intents de paiement en double empêche la double facturation
- Suivi des adresses IP pour la corrélation d'utilisation
- Validation de l'IMSI pour s'assurer que le ravitaillement va au bon service
- Les remboursements automatiques limitent l'exposition financière

**Contrôle d'Accès :**

- Le portail de ravitaillement est public (par conception - les clients ont besoin d'accès)
- Le point de terminaison d'utilisation nécessite une identification de service valide (IP ou IMSI)
- La validation backend empêche les ravitaillements de service arbitraires
- L'administrateur peut voir tous les enregistrements de ravitaillement via l'interface CRM

# Meilleures Pratiques

**Pour les Opérateurs :**

1. **Tester le flux de remboursement** - Testez régulièrement les scénarios d'échec pour vous assurer que les remboursements fonctionnent
2. **Surveiller les ravitaillements échoués** - Configurez des alertes pour des taux d'échec élevés

3. **Garder les playbooks simples** - Les playbooks de ravitaillement doivent être rapides et fiables
4. **Vérifier la connectivité OCS** - Assurez-vous que l'API OCS est toujours accessible
5. **Réviser les prix** - Mettez à jour `VITE_TOPUP_PRICE_PER_DAY` si nécessaire

**Pour les Clients :**

1. **Ajouter l'URL de ravitaillement aux favoris** - Accès rapide en cas de besoin
2. **Conserver les notifications de solde bas** - Les SMS contiennent un lien direct
3. **Maintenir l'e-mail à jour** - Les reçus sont envoyés à l'e-mail enregistré
4. **Vérifier l'expiration avant de voyager** - Ravitaillez avant de quitter la zone de couverture

**Pour les Développeurs :**

1. **Gérer les webhooks Stripe** - Implémentez des gestionnaires de webhook pour les mises à jour de statut de paiement
2. **Implémenter l'idempotence** - Vérifiez toujours payment_intent_id avant de traiter
3. **Logger de manière extensive** - Les échecs de ravitaillement nécessitent des informations de dépannage détaillées
4. **Tester les chemins d'erreur** - Vérifiez que l'automatisation des remboursements fonctionne correctement
5. **Surveiller les performances** - Le polling de provisionnement doit se terminer en <5 secondes

# Documentation Connexe

- `payments_process` - Traitement général des paiements
- `concepts_provisioning` - Aperçu du système de provisionnement
- Guide du Système de Paiement - Détails sur l'intégration des fournisseurs de paiement
- `payments_transaction` - Gestion des transactions

- `payments_invoices` - Gestion des factures

# Glossaire des Termes

Définitions de la terminologie utilisée dans toute la documentation d'OmniCRM. Les termes sont listés par ordre alphabétique. Lorsqu'un concept a son propre guide, un lien vers la page pertinente est inclus.

## A

**Addon** - Un extra optionnel attaché à un service existant, tel qu'un amplificateur de données ou une fonctionnalité à valeur ajoutée. Les addons sont facturés avec le service parent plutôt que vendus comme un produit autonome.

**Ansible** - Le moteur d'automatisation qu'OmniCRM utilise pour provisionner des services. Les définitions de produits déclenchent des playbooks Ansible qui effectuent la configuration sous-jacente. Voir Provisionnement Basé sur Ansible.

**AuC (Centre d'Authentification)** - La fonction qui stocke la clé secrète matérielle pour une carte SIM et calcule les vecteurs d'authentification utilisés pour vérifier un abonné sur le réseau mobile. Dans OmniCRM, l'AuC est peuplé dans le cadre du provisionnement de la SIM vers le HSS.

## C

**CGRateS** - Le moteur de tarification et de facturation en temps réel intégré à OmniCRM. Il évalue l'utilisation (voix, SMS, données) par rapport aux destinations et applique des soldes, et est configuré via des profils d'attributs, des filtres de compte et des tarifs de destination.

**Cell Broadcast** - Une capacité de messagerie un-à-plusieurs utilisée pour envoyer des alertes d'urgence et des informations publiques à tous les appareils dans une zone géographique définie. Voir Système de Diffusion Cellulaire.

# E

**eSIM** - Une SIM intégrée qui est provisionnée par voie aérienne plutôt qu'en insérant une carte physique. OmniCRM prend en charge l'émission et l'importation d'eSIM. Voir Provisionnement de Carte SIM.

# H

**HSS (Serveur d'Abonnés Principal)** - La base de données centrale des abonnés d'un réseau mobile. Elle contient des données d'abonnement et d'authentification et répond aux demandes du réseau lorsqu'un abonné se connecte. OmniCRM écrit des enregistrements d'abonnés dans le HSS lors du provisionnement de la SIM et prend en charge plusieurs backends HSS.

# I

**IMS (Système Multimédia IP)** - Le cadre du réseau central qui fournit des services voix (VoLTE) et multimédias basés sur IP. Les abonnés sont provisionnés dans l'IMS avec le HSS afin que les services vocaux fonctionnent.

**IMSI (Identité Internationale d'Abonné Mobile)** - L'identifiant unique associé à une SIM que le réseau utilise pour identifier un abonné. OmniCRM associe l'IMSI à un compte de facturation lors du provisionnement.

**Inventaire** - Le pool de ressources physiques et logiques (cartes SIM, numéros de téléphone mobile et équipements) qui peuvent être attribuées aux clients. Les éléments passent par des états définis alors qu'ils sont stockés, attribués et activés. Voir Gestion de l'Inventaire.

# M

**MSISDN (Numéro ISDN de Station Mobile)** - Le numéro de téléphone appelable associé à un abonné. L'IMSI d'une SIM est l'identité réseau, tandis que le MSISDN est le numéro que d'autres personnes composent.

# O

**OCS (Système de Facturation en Ligne)** - Le système responsable du contrôle de crédit en temps réel, autorisant et facturant l'utilisation au fur et à mesure qu'elle se produit. L'intégration OCS permet à OmniCRM d'appliquer des soldes et d'éviter les dépassements de dépenses sur les services prépayés.

# P

**Playbook** - Une routine d'automatisation Ansible nommée qui effectue une tâche de provisionnement, telle que l'activation d'une SIM ou la configuration d'équipements clients. Le playbook de provisionnement d'un produit est exécuté lorsque le service associé est créé. Voir Playbooks Ansible.

**PLMN (Réseau Mobile Terrestre Public)** - Le réseau d'un opérateur de réseau mobile, identifié par un code. La facturation des données utilise des destinations PLMN (le réseau auquel le client est connecté), tandis que la voix et les SMS utilisent des destinations géographiques.

**Produit** - Une définition vendable dans le catalogue. Un produit spécifie les prix, le comportement de provisionnement, et le(s) service(s) qu'il crée lors de l'achat. Voir Produits et Services.

**Facture Proforma** - Une facture préliminaire émise avant que le paiement ne soit dû ou avant qu'une vente ne soit finalisée. Elle indique ce qui sera facturé mais n'est pas en elle-même une demande de paiement. Voir Factures.

**Provisionnement** - Le processus de transformation d'un produit acheté en un service fonctionnel en effectuant la configuration réseau et système requise, généralement via des playbooks Ansible. Voir Provisionnement.

# R

**RBAC (Contrôle d'Accès Basé sur les Rôles)** - Le modèle de permission qui régit ce que chaque utilisateur peut voir et faire dans OmniCRM. Les

permissions sont regroupées en rôles, et les rôles sont attribués aux utilisateurs. Voir Contrôle d'Accès Basé sur les Rôles.

# S

**Portail de Self-Care** - Le portail destiné aux clients où les abonnés peuvent consulter leur compte, leur utilisation, leurs factures et leurs services sans contacter le support. Voir Portail de Self-Care.

**Auto-Inscription** - Un flux qui permet à un nouveau client de s'inscrire et de provisionner un service par lui-même, y compris le provisionnement de la SIM, sans intervention du personnel.

**Service** - Une instance active et provisionnée d'un produit sur le compte d'un client. Acheter un produit crée un ou plusieurs services, chacun pouvant être géré, modifié ou suspendu. Voir Produits et Services.

**SIM (Module d'Identité de l'Abonné)** - La carte ou le module intégré qui identifie un abonné au réseau mobile et stocke la clé d'authentification. Voir Provisionnement de Carte SIM.

# T

**Tag** - Une étiquette appliquée aux clients ou aux enregistrements pour les regrouper et les filtrer, par exemple par segment ou campagne. Voir Tags.

**Transaction** - Un mouvement financier unique sur un compte, tel qu'une charge, un paiement, un crédit ou un remboursement. Les transactions constituent l'historique de facturation du compte. Voir Transactions.

# Nombres

**2FA (Authentification à Deux Facteurs)** - Une étape de connexion supplémentaire qui requiert un code à usage unique en plus du mot de passe, protégeant les comptes du personnel contre les identifiants compromis. Voir Authentification à Deux Facteurs.

# Guide complet du cycle de vie du produit

Ce guide fournit un aperçu complet du cycle de vie du produit dans OmniCRM, depuis la création d'une définition de produit jusqu'à la fourniture de services, l'ajout d'options supplémentaires et la déprovisionnement. Nous aborderons la stratégie de tarification, l'intégration Ansible et fournirons des exemples concrets tout au long du document.

## Aperçu : Le parcours du produit au service

Le cycle de vie d'un produit dans OmniCRM suit ces étapes :

1. **Définition du produit** - L'administrateur crée un modèle de produit avec des règles de tarification et de provisionnement.
2. **Création de service** - Le client commande le produit, le système provisionne l'instance de service.
3. **Cycle de vie du service** - Le client utilise le service, ajoute des options supplémentaires/crédits, modifie le service.
4. **Déprovisionnement** - Le service est résilié, les ressources sont libérées.

## Comprendre la tarification : Gros vs Détail

Chaque produit et service dans OmniCRM a deux dimensions de tarification : **gros** et **détail**.

**Coût de gros**

Le coût de gros représente le coût réel pour fournir le service :

- Coûts d'infrastructure et de bande passante
- Frais de licence
- Coûts d'équipement
- Dépenses opérationnelles

**Coût de détail**

Le coût de détail est le montant facturé au client.

**Coûts de mise en place**

Les coûts de mise en place en gros et au détail ont des variantes pour les frais de provisionnement uniques :

- `wholesale_setup_cost` - Votre coût de provisionnement
- `retail_setup_cost` - Montant facturé au client pour l'activation

**Exemple :**

```
{
  "retail_cost": 15.00,
  "wholesale_cost": 5.00,
  "retail_setup_cost": 0.00,
  "wholesale_setup_cost": 1.00
}
```

# Étape 1 : Création d'une définition de produit

Les produits sont des modèles qui définissent ce qui est provisionné et comment les clients sont facturés.

## Création d'un produit de carte SIM mobile

Créez un produit de carte SIM mobile prépayé avec 20 Go de données par mois.

**Étape 1 : Accéder à la gestion des produits**

Depuis l'interface admin, allez à **Produits** → **Créer un produit**.

**Étape 2 : Définir les informations de base**

```
{
  "product_name": "Prepaid Mobile 20GB",
  "product_slug": "prepaid-mobile-20gb",
  "category": "standalone",
  "service_type": "mobile",
  "enabled": true,
  "icon": "fa-solid fa-sim-card",
  "comment": "Carte SIM mobile prépayée avec 20 Go de données,
appels et SMS illimités"
}
```

**Explications des champs :**

- `product_name` - Nom visible par le client affiché dans le catalogue
- `product_slug` - Identifiant sécurisé pour l'URL utilisé dans les appels API et les liens
- `category` - "standalone" signifie que cela crée un nouveau service (vs option supplémentaire/pack)
- `service_type` - Regroupe des produits connexes, utilisé pour le filtrage des options supplémentaires
- `enabled` - Doit être vrai pour que le produit soit commandable
- `icon` - Icône FontAwesome affichée dans l'interface utilisateur
- `comment` - Notes internes pour référence du personnel

**Étape 3 : Définir les prix**

```
{
  "retail_cost": 15.00,
  "wholesale_cost": 5.00,
  "retail_setup_cost": 0.00,
  "wholesale_setup_cost": 1.00,
  "contract_days": 30
}
```

**Répartition des prix :**

- Revenu mensuel par client : 15,00 £
- Coût mensuel de livraison : 5,00 £
- Marge bénéficiaire mensuelle : 10,00 £ (200 % de majoration, 67 % de marge)
- Profit de mise en place : -1,00 £ (subventionné pour attirer les clients)
- Durée du contrat : 30 jours (renouvellement mensuel)

**Étape 4 : Définir l'éligibilité des clients**

```
{
  "residential": true,
  "business": false,
  "customer_can_purchase": true,
  "available_from": "2025-01-01T00:00:00Z",
  "available_until": null
}
```

- Les clients résidentiels peuvent commander
- Les clients professionnels ne peuvent pas (ligne de produit différente)
- Achat en libre-service activé
- Disponible à partir du 1er janvier 2025
- Pas de date de fin (offre continue)

**Étape 5 : Configurer le renouvellement automatique**

```
{
  "auto_renew": "prompt",
  "allow_auto_renew": true
}
```

- `"prompt"` - Demander au client s'il souhaite un renouvellement automatique lors de l'achat
- `"true"` - Renouveler automatiquement sans demander
- `"false"` - Ne jamais renouveler automatiquement (recharge manuelle uniquement)

- `allow_auto_renew: true` - Le client peut activer/désactiver le renouvellement automatique plus tard

**Étape 6 : Spécifier les exigences d'inventaire**

Les exigences d'inventaire définissent quelles ressources physiques ou virtuelles doivent être allouées lors du provisionnement de ce produit. C'est une étape critique qui relie votre catalogue de produits à votre Système de gestion des stocks.

```
{
  "inventory_items_list": "['SIM Card', 'Mobile Number']"
}
```

**Qu'est-ce que les articles d'inventaire ?**

Les articles d'inventaire sont des ressources traçables stockées dans le système d'inventaire d'OmniCRM. Chaque article a :

- **Type** - Défini par le modèle d'inventaire (par exemple, "Carte SIM", "Numéro de mobile", "Modem")
- **Attributs uniques** - Numéros de série, adresses MAC, numéros de téléphone, etc.
- **État** - En stock, Assigné, Déclassé, etc.
- **Emplacement** - Emplacement physique ou logique

**Comment fonctionnent les exigences d'inventaire :**

La `inventory_items_list` est une liste Python (sous forme de chaîne) contenant les noms des types d'inventaire. Chaque nom doit correspondre exactement à un nom de Modèle d'inventaire existant.

**Exemples d'exigences d'inventaire :**

```
# Produit de carte SIM mobile
inventory_items_list: "['SIM Card', 'Mobile Number']"

# Service Internet fixe
inventory_items_list: "['Modem Router', 'Static IP Address']"

# Service numérique (pas d'articles physiques)
inventory_items_list: "[]"

# Sans fil fixe avec CPE
inventory_items_list: "['Fixed Wireless CPE', 'IPv4 Address',
'IPv6 Prefix']"
```

**Le processus de sélection d'inventaire**

Lorsqu'un utilisateur provisionne un produit avec des exigences d'inventaire, le système impose un processus de sélection obligatoire :

**1. Bouton de provisionnement cliqué**

Après avoir sélectionné le produit, l'utilisateur clique sur "Provisionner". Au lieu de provisionner immédiatement, le système vérifie `inventory_items_list`.

**2. La fenêtre modale du sélecteur d'inventaire apparaît**

Si l'inventaire est requis, une boîte de dialogue modale apparaît avec un menu déroulant séparé pour chaque type d'inventaire :

**3. Filtrage de l'inventaire disponible**

Le menu déroulant pour chaque type d'inventaire ne montre que les articles qui sont :

- **Type Correct** - Correspond exactement au nom du modèle d'inventaire
- **État Disponible** - `item_state` est "New" ou "In Stock" (pas "Assigned" ou "Damaged")
- **Non Assigné** - `service_id` et `customer_id` sont NULL
- **En Stock à l'Emplacement** - Filtré éventuellement par emplacement d'entrepôt/magasin

**Exemples d'options de menu déroulant :**

Pour le type d'inventaire "Carte SIM", le menu déroulant pourrait afficher :

Chaque option affiche :

- ID d'inventaire ou numéro de référence
- Identifiant principal (`itemtext1` - par exemple, ICCID pour SIM, numéro pour téléphone)
- Emplacement actuel (`item_location`)

**4. Sélection requise pour continuer**

**Règle critique :** Le provisionnement NE PEUT PAS se poursuivre sans sélectionner tous les articles d'inventaire requis.

- Le bouton "Continuer" est désactivé jusqu'à ce que tous les menus déroulants aient des sélections
- L'utilisateur doit sélectionner un article pour chaque type d'inventaire
- Le système valide les sélections avant de continuer

**5. Inventaire sélectionné transmis à Ansible**

Une fois que l'utilisateur clique sur "Continuer", les ID d'inventaire sélectionnés sont transmis au playbook Ansible en tant que variables :

```
# L'utilisateur a sélectionné :
# - ID d'inventaire de la carte SIM : 5001
# - ID d'inventaire du numéro de mobile : 5002

# Variables transmises à Ansible :
{
    "product_id": 42,
    "customer_id": 123,
    "SIM Card": 5001,           # ID d'inventaire
    "Mobile Number": 5002,       # ID d'inventaire
    "access_token": "eyJ..."
}
```

**Remarque :** Le nom de la variable correspond exactement au type d'inventaire. Le playbook utilise `hostvars[inventory_hostname]['SIM Card']` pour accéder à l'ID d'inventaire.

**6. Le playbook récupère les détails complets de l'inventaire**

Le playbook Ansible utilise l'ID d'inventaire pour récupérer les détails complets :

```
- name: Get SIM Card details from inventory
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ hostvars[inventory_hostname]['SIM
Card'] }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
  register: api_response_sim

- name: Extract ICCID and IMSI
  set_fact:
    iccid: "{{ api_response_sim.json.itemtext1 }}"
    imsi: "{{ api_response_sim.json.itemtext2 }}"
```

Maintenant, le playbook a tous les détails de la SIM (ICCID, IMSI, etc.) pour provisionner l'abonné dans le HSS.

**7. État de l'inventaire changé en "Assigné"**

Après la création de l'enregistrement de service, le playbook met à jour l'inventaire pour le lier au service :

```
- name: Assign SIM Card to Service
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ hostvars[inventory_hostname]['SIM
Card'] }}"
    method: PATCH
    body:
      {
        "service_id": "{{
service_creation_response.json.service_id }}",
        "customer_id": "{{ customer_id }}",
        "item_state": "Assigned"
      }
```

**Important :** L'attribution de l'inventaire se produit **pendant l'exécution du playbook** en tant que tâche spécifique, PAS lorsque le bouton de provisionnement est cliqué. Cela signifie :

- **Risque de double attribution** : Entre le clic sur "Provisionner" et l'attribution de l'inventaire, un autre utilisateur pourrait théoriquement sélectionner le même article d'inventaire.
- **Meilleure pratique** : Pour les opérations à volume élevé, mettez en œuvre un verrouillage de l'inventaire ou utilisez des transactions de base de données.
- **Rollback en cas d'échec** : Si le playbook échoue avant l'attribution de l'inventaire, l'inventaire reste non attribué et disponible pour une réutilisation.

**Pourquoi ne pas attribuer plus tôt ?**

L'inventaire n'est pas attribué lorsque "Provisionner" est cliqué parce que :

1. **ID de service requis** : L'`service_id` n'existe pas tant que le service n'est pas créé dans le playbook.
2. **Simplicité de rollback** : Si le provisionnement échoue tôt (par exemple, l'échec de la création de compte OCS), l'inventaire n'a pas besoin d'être nettoyé.
3. **Flexibilité** : Le playbook peut décider de ne pas attribuer l'inventaire en fonction de la logique conditionnelle.

**Gestion des échecs de provisionnement :**

Lorsqu'un provisionnement échoue après l'attribution de l'inventaire, le bloc de secours doit libérer l'inventaire :

```
rescue:
  - name: Release inventory on failure
    uri:
      url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ hostvars[inventory_hostname]['SIM
Card'] }}"
      method: PATCH
      body:
        {
          "service_id": null,
          "customer_id": null,
          "item_state": "In Stock"
        }
    when: service_id is defined  # Seulement si le service a été
créé
```

Cela garantit que l'inventaire n'est pas laissé dans un état "Assigné" pour un service inexistant ou échoué.

**Lorsque la liste d'inventaire est vide**

Si `inventory_items_list: "[]"` (liste vide), le sélecteur d'inventaire est complètement ignoré et le provisionnement se poursuit immédiatement. Cela est courant pour :

- **Produits numériques** - Licences logicielles, comptes VPN
- **Options supplémentaires de service** - Recharges de données qui ne nécessitent pas de nouveau matériel
- **Services virtuels** - Qui ne consomment pas de ressources traçables

**Exemple :** Un addon "5 Go de données supplémentaires" a `inventory_items_list: "[]"` car il ajoute simplement un solde à un service existant sans nécessiter de nouveau matériel.

**Configuration du modèle d'inventaire**

Avant d'utiliser un type d'inventaire dans `inventory_items_list`, vous devez créer le modèle d'inventaire :

1. Accédez à **Administration → Inventaire → Modèles**
2. Créez un modèle avec le nom exact (par exemple, "Carte SIM")
3. Définissez les champs :
   - `itemtext1_label`: "ICCID"
   - `itemtext2_label`: "IMSI"
   - `itemtext3_label`: "Code PUK"
4. Ajoutez des articles d'inventaire de ce type en stock

Pour des détails complets sur la création et la gestion des modèles d'inventaire, consultez Gestion des stocks.

**Plusieurs articles du même type**

Bien que la `inventory_items_list` soit un tableau, avoir des types en double (par exemple, `"['SIM Card', 'SIM Card']"`) est **non recommandé** car cela peut causer de la confusion dans l'interface utilisateur et le nommage des variables du playbook.

Pour les scénarios nécessitant plusieurs articles similaires :

**Option 1 : Créer des noms de modèles d'inventaire distincts**

```
# Service de téléphone à double SIM
inventory_items_list: "['Primary SIM Card', 'Secondary SIM Card',
'Mobile Number']"
```

Créez des modèles séparés : "Primary SIM Card" et "Secondary SIM Card" avec les mêmes champs mais des noms différents.

**Option 2 : Utiliser un seul article d'inventaire groupé**

```
# Kit double SIM
inventory_items_list: "['Dual SIM Kit', 'Mobile Number']"
```

Où le modèle d'inventaire "Dual SIM Kit" a des champs pour les deux SIM (`itemtext1`: ICCID principal, `itemtext2`: ICCID secondaire, etc.).

**Scénarios d'inventaire courants**

**Service mobile :**

```
inventory_items_list: "['SIM Card', 'Mobile Number']"
```

- Carte SIM : Physique ou eSIM avec ICCID/IMSI
- Numéro mobile : Numéro de téléphone (MSISDN)

**Internet fixe :**

```
inventory_items_list: "['Modem Router', 'Static IP Address']"
```

- Modem Routeur : Dispositif CPE avec adresse MAC
- Adresse IP statique : IPv4 du pool d'adresses

**Sans fil fixe :**

```
inventory_items_list: "['Fixed Wireless CPE', 'IPv4 Address',
'IPv6 Prefix']"
```

- CPE : Équipement de locaux du client (antenne, modem)
- IPv4 : Adresse IP publique
- Préfixe IPv6 : préfixe /56 ou /64

**Remarque :** Les rendez-vous et la planification ne sont **pas** des articles d'inventaire. Utilisez des systèmes de planification/calendrier séparés pour les rendez-vous d'installation.

**Service VoIP :**

```
inventory_items_list: "['DID Number']"
```

- Numéro DID : Numéro de téléphone de composition directe

**Remarque :** Les noms d'utilisateur SIP, les mots de passe et les configurations de compte sont **générés par programme** par le playbook de provisionnement, pas sélectionnés dans l'inventaire.

**GPON/Fibre :**

```
inventory_items_list: "['ONT Device', 'GPON Port', 'IPv4 Address',
'Fiber Drop Cable']"
```

- Dispositif ONT : Terminal de réseau optique avec numéro de série
- Port GPON : Port spécifique sur OLT avec connexion fibre
- Adresse IPv4 : IP publique ou privée
- Câble de fibre : Câble de fibre physique de la rue aux locaux (suivi pour la gestion des actifs)

**Location de matériel :**

```
inventory_items_list: "['Rental Modem']"
```

- Suit quel modem est avec quel client
- Important pour récupérer le matériel en cas d'annulation

**Pourquoi les exigences d'inventaire sont importantes**

**1. Prévenir la double attribution**

Sans suivi de l'inventaire, vous pourriez accidentellement :

- Attribuer la même carte SIM à deux clients
- Allouer la même adresse IP à plusieurs services
- Expédier le même numéro de série d'équipement à différents emplacements

Le sélecteur d'inventaire garantit que chaque article est attribué à un seul service.

**2. Piste d'audit**

L'attribution de l'inventaire crée une piste d'audit complète :

- Quelle carte SIM est avec quel client
- Quand a-t-elle été attribuée
- Quel service utilise quel numéro de téléphone
- Historique de l'équipement (qui l'a eu, quand, pour quel service)

**3. Planification des ressources**

Suivez les niveaux d'inventaire :

- Alerte lorsque les cartes SIM sont en faible quantité
- Réapprovisionner avant rupture de stock
- Planifiez les horaires des techniciens en fonction de la disponibilité du CPE
- Gérez l'allocation de l'espace d'adresses IP

**4. Suivi des coûts**

Liez le coût de gros à un article spécifique :

- Suivez le coût de chaque carte SIM
- Calculez l'amortissement de l'équipement
- Identifiez les articles perdus ou volés
- Coût des biens vendus (COGS) précis

**5. Déprovisionnement**

Lorsque le service est annulé, l'inventaire peut être :

- Libéré en stock (cartes SIM, modems)
- Retiré (équipement endommagé)
- Renvoyé au fournisseur (équipement de location)
- Conservé pour une période de grâce (numéros de téléphone avant libération)

**Dépannage des problèmes de sélecteur d'inventaire**

**Problème :** Message "Aucun inventaire disponible" apparaît

**Causes :**

- Aucun article d'inventaire du type requis n'existe dans la base de données
- Tous les articles sont déjà "Assignés" à d'autres services
- Les articles sont marqués comme "Endommagés" ou "Hors service"
- Le nom du modèle d'inventaire ne correspond pas exactement (sensible à la casse)

**Solution :**

1. Vérifiez que le modèle d'inventaire existe : **Administration** → **Inventaire** → **Modèles**
2. Vérifiez que le nom du modèle correspond exactement (y compris les espaces, la casse)
3. Ajoutez des articles d'inventaire de ce type : **Administration** → **Inventaire** → **Ajouter un article**
4. Vérifiez que les articles sont dans l'état "New" ou "In Stock"
5. Vérifiez que les articles ne sont pas déjà assignés (`service_id` doit être NULL)

**Problème :** Le sélecteur d'inventaire n'apparaît pas

**Causes :**

- `inventory_items_list` est vide : `"[]"`
- `inventory_items_list` est NULL ou non défini
- La catégorie de produit est "addon" et hérite de l'inventaire du service parent

**Solution :**

- Si l'inventaire est nécessaire, définissez `inventory_items_list: "['Type1', 'Type2']"`
- Vérifiez que la définition du produit est correctement enregistrée
- Vérifiez la réponse API pour que le produit inclue inventory_items_list

**Problème :** Le playbook échoue avec "inventaire non trouvé"

**Causes :**

- Le playbook fait référence à un mauvais nom de variable
- L'ID d'inventaire n'est pas transmis correctement
- L'inventaire a été supprimé entre la sélection et le provisionnement

**Solution :**

- Vérifiez que le playbook utilise la variable correcte : `hostvars[inventory_hostname]['SIM Card']`
- Vérifiez que la variable est un entier : `{{ hostvars[inventory_hostname]['SIM Card'] | int }}`
- Ajoutez une gestion des erreurs dans le playbook pour l'inventaire manquant

Consultez Gestion des stocks pour des détails complets sur la création de modèles, l'ajout d'articles et la gestion des niveaux de stock.

**Étape 7 : Définir les fonctionnalités et les conditions**

Les fonctionnalités et les conditions sont du contenu marketing et légal visible par le client qui aide les clients à comprendre ce qu'ils achètent et les obligations impliquées.

```
{
  "features_list": "Données haute vitesse de 20 Go. Appels et SMS illimités. Roaming UE inclus. Pas de contrat. Expiration de 30 jours.",
  "terms": "Le crédit expire après 30 jours. Les données, appels et SMS ne sont valables que pendant la période d'expiration. La politique d'utilisation équitable s'applique. Voir le site Web pour les conditions complètes."
}
```

**Objectif et valeur commerciale**

**Liste des fonctionnalités - Marketing et ventes :**

La liste des fonctionnalités remplit plusieurs fonctions commerciales critiques :

1. **Différenciation du produit** - Aide les clients à comparer rapidement les produits et à choisir le bon
    - "Mobile prépayé 20 Go" vs "Mobile prépayé 50 Go" - les fonctionnalités montrent clairement la différence
    - Sans fonctionnalités, les clients ne voient que le prix, manquant la proposition de valeur
2. **Communication marketing** - Points de vente clés affichés en évidence
    - "Roaming UE inclus" attire les voyageurs internationaux
    - "Pas de contrat" attire les clients réticents à l'engagement
    - Les fonctionnalités influencent les décisions d'achat
3. **Attentes des clients** - Définit des attentes claires sur ce qui est inclus
    - Réduit les appels de support ("Cela inclut-il des appels ?" → clairement listé)
    - Prévenir les malentendus et les demandes de remboursement
    - Renforce la confiance par la transparence
4. **Libre-service** - Permet aux clients de sélectionner eux-mêmes les produits appropriés
    - Le client lit les fonctionnalités, comprend l'offre, fait un choix éclairé
    - Réduit le besoin d'explications de la part du personnel de vente
    - Accélère le processus d'achat
5. **SEO et découvrabilité** - Les fonctionnalités peuvent être indexées pour la recherche
    - Le client recherche "plan mobile appels illimités" → le produit apparaît
    - Améliore la recherche dans le catalogue de produits

**Conditions générales - Légal et conformité :**

Les conditions servent des objectifs légaux et opérationnels :

1. **Protection légale** - Protège l'entreprise des litiges et de la responsabilité
    - "Le crédit expire après 30 jours" - le client ne peut pas demander de remboursement après 31 jours
    - "La politique d'utilisation équitable s'applique" - prévient les abus (tethering de tout un bureau sur un plan mobile)

    - Crée un accord contraignant
2. **Gestion des attentes** - Prévenir l'insatisfaction du client
    - "Valable uniquement pendant la période d'expiration" - le client connaît la date limite d'utilisation
    - "Ne peut pas être remboursé" (pour les options supplémentaires) - prévient les achats frauduleux
    - Réduit les rétrofacturations et les plaintes
3. **Conformité réglementaire** - Répond aux exigences légales
    - Les lois sur la protection des consommateurs exigent des conditions claires
    - Les réglementations en matière de télécommunications imposent une divulgation
    - Les termes de confidentialité/ RGPD peuvent être référencés
4. **Limites opérationnelles** - Définit la portée et les limitations du service
    - "Sous réserve de la couverture réseau" - non responsable des zones mortes
    - "La vitesse peut varier" - gère les attentes sur les vitesses "jusqu'à"
    - "L'équipement doit être retourné" - garantit la récupération de l'équipement de location
5. **Piste d'audit** - Prouve que le client a été informé
    - Le client a accepté les conditions lors de l'achat
    - Le système enregistre l'horodatage d'acceptation
    - Défendable en cas de litiges ou de procédures judiciaires

**Exemple concret :**

Un client achète un plan "Appels et SMS illimités", puis l'utilise pour le télémarketing (10 000 appels/jour). Sans conditions :

- Client : "Vous avez dit illimité !"
- Fournisseur : "Nous voulions dire usage personnel..."
- Client : "Ce n'est pas ce que vous avez annoncé !"
- Résultat : Litige, plainte potentielle auprès du régulateur, dommages à la marque

Avec des conditions : "La politique d'utilisation équitable s'applique. Le service est uniquement pour un usage personnel. L'utilisation commerciale est interdite."

- Fournisseur : Montre les conditions acceptées par le client
- Le client ne peut pas prétendre à l'ignorance
- Base légale pour suspendre le service
- Litige résolu en faveur du fournisseur

**Format de la liste des fonctionnalités :**

Comprendre le format correct est crucial car **un formatage incorrect casse l'affichage de l'interface utilisateur**. Les fonctionnalités peuvent apparaître comme une longue chaîne au lieu de points de balle, ou ne pas s'afficher du tout.

Le champ `features_list` peut être formaté de deux manières :

**Option 1 : Chaîne séparée par des points (recommandée)**

Les fonctionnalités sont séparées par un point et un espace (". "). L'interface utilisateur se divise sur ce délimiteur et rend chaque fonctionnalité comme un point de balle.

**Pourquoi ce format ?**

- Simple à éditer - il suffit de taper les fonctionnalités avec des points entre elles
- Aucun caractère spécial à échapper
- Fonctionne de manière fiable sur tous les composants de l'interface utilisateur
- Facile à mettre à jour sans casser la syntaxe JSON

**Correct vs Incorrect :**

**Option 2 : Chaîne de tableau JSON**

```
"['20GB High-Speed Data', 'Unlimited Calls & Texts', 'EU Roaming Included']"
```

L'interface utilisateur peut également analyser les tableaux JSON. Notez qu'il s'agit d'une **chaîne contenant du JSON**, pas d'un tableau JSON réel dans la base de données.

**Pourquoi ce format existe-t-il ?**

- Permet des fonctionnalités avec des points en elles (par exemple, "Jusqu'à 100 Mbps. Sous réserve de disponibilité.")
- La génération programmatique à partir de scripts/API est plus facile
- Importé de catalogues de produits externes qui utilisent des tableaux

**Important :** Cela doit être une syntaxe de liste Python valide sous forme de chaîne. Guillemets simples autour de chaque élément, guillemets doubles autour de toute la chaîne.

**Quel format utiliser ?**

- **Séparé par des points** - Pour la création manuelle de produits dans l'interface utilisateur (plus simple, moins sujet aux erreurs)
- **Tableau JSON** - Pour la création de produits basée sur API/script (plus robuste pour des fonctionnalités complexes)

Les deux formats produisent une sortie identique dans l'interface utilisateur - ils n'affectent que la manière dont vous saisissez les données.

**Où les fonctionnalités apparaissent dans l'interface utilisateur :**

**1. Catalogue de produits (vue client)**

Lorsque les clients parcourent les produits disponibles, les fonctionnalités sont affichées sous forme de points de balle sur chaque carte de produit :

**2. Page de détails du produit**

En cliquant sur "Voir les détails", toutes les informations sur le produit sont affichées, y compris :

- Nom du produit et icône
- Tarification (coût mensuel, coût de mise en place)
- Liste complète des fonctionnalités (points de balle)
- Conditions générales (voir ci-dessous)
- Disponibilité et éligibilité

**3. Confirmation de provisionnement**

Lors du provisionnement, les fonctionnalités sont affichées pour que l'utilisateur les examine avant de confirmer :

> Fonctionnalités : • Données haute vitesse de 20 Go • Appels et SMS illimités • Roaming UE inclus • Pas de contrat • Expiration de 30 jours
>
> Coût : 15,00 £/mois Mise en place : 0,00 £
>
> [Annuler] [Confirmer & Provisionner]

**4. Détails du service (après provisionnement)**

Une fois le service actif, les fonctionnalités sont affichées sur la page de détails du service pour référence du client.

**Format des conditions générales :**

Le champ `terms` est du texte brut qui peut inclure des sauts de ligne :

**Où apparaissent les conditions dans l'interface utilisateur :**

**1. Page de détails du produit**

Les conditions sont affichées dans une section repliée qui s'agrandit lorsqu'on clique :

**2. Confirmation de commande**

Lors du provisionnement, une case à cocher exige que l'utilisateur accepte les conditions :

> Le bouton [Provisionner] est désactivé tant qu'il n'est pas coché

### 3. Factures

Les conditions de service peuvent être incluses sur les factures en tant que notes de bas de page pour plus de clarté.

**Meilleures pratiques :**

- **Fonctionnalités :** Gardez concis (moins de 50 caractères chacun), concentrez-vous sur les avantages clés
- **Conditions :** Incluez les exigences légales critiques, les politiques d'expiration, les politiques d'utilisation équitable
- **Les deux :** Mettez à jour lorsque le produit change pour tenir les clients informés

**Étape 8 : Lier le playbook de provisionnement Ansible**

```
{
  "provisioning_play": "play_local_mobile_sim",
  "provisioning_json_vars": "{
    \"days\": 30,
    \"data_gb\": 20,
    \"voice_minutes\": \"unlimited\",
    \"sms_count\": \"unlimited\"
  }"
}
```

- `provisioning_play` - Nom du playbook Ansible (sans extension .yaml)
- `provisioning_json_vars` - Variables par défaut transmises au playbook
- Le playbook doit exister à : `OmniCRM-API/Provisioners/plays/play_local_mobile_sim.yaml`

**Définition complète du produit**

```
{
  "product_name": "Prepaid Mobile 20GB",
  "product_slug": "prepaid-mobile-20gb",
  "category": "standalone",
  "service_type": "mobile",
  "enabled": true,
  "icon": "fa-solid fa-sim-card",
  "comment": "Carte SIM mobile prépayée avec 20 Go de données,
appels et SMS illimités",

  "retail_cost": 15.00,
  "wholesale_cost": 5.00,
  "retail_setup_cost": 0.00,
  "wholesale_setup_cost": 1.00,
  "contract_days": 30,

  "residential": true,
  "business": false,
  "customer_can_purchase": true,
  "available_from": "2025-01-01T00:00:00Z",
  "available_until": null,

  "auto_renew": "prompt",
  "allow_auto_renew": true,

  "inventory_items_list": "['SIM Card', 'Mobile Number']",

  "features_list": "[
    'Données haute vitesse de 20 Go',
    'Appels et SMS illimités',
    'Roaming UE inclus',
    'Pas de contrat',
    'Expiration de 30 jours'
  ]",
  "terms": "Le crédit expire après 30 jours. Les données, appels
et SMS ne sont valables que pendant la période d'expiration. La
politique d'utilisation équitable s'applique.",

  "provisioning_play": "play_local_mobile_sim",
  "provisioning_json_vars": "{
    \"days\": 30,
    \"data_gb\": 20,
    \"voice_minutes\": \"unlimited\",
```

```
    \"sms_count\": \"unlimited\"
  }"
}
```

## Création d'un produit addon

Les addons améliorent ou modifient les services existants. Ils se présentent sous deux types : **addons virtuels** (pas de ressources physiques) et **addons matériels** (nécessitent un inventaire).

**Exemple 1 : Addon virtuel (5 Go de données supplémentaires)**

Un addon numérique qui ajoute des données à un service mobile existant :

```
{
  "product_name": "5GB Data Boost",
  "product_slug": "5gb-data-boost",
  "category": "addon",
  "service_type": "mobile",
  "enabled": true,
  "icon": "fa-solid fa-plus",
  "comment": "Ajoutez 5 Go de données supplémentaires à un service
mobile existant",

  "retail_cost": 5.00,
  "wholesale_cost": 1.50,
  "retail_setup_cost": 0.00,
  "wholesale_setup_cost": 0.00,
  "contract_days": 0,

  "residential": true,
  "business": true,
  "customer_can_purchase": true,

  "auto_renew": "false",
  "allow_auto_renew": false,

  "inventory_items_list": "[]",
  "relies_on_list": "",

  "features_list": "Données haute vitesse de 5 Go. Valable pendant
7 jours.",
  "terms": "Les données expirent après 7 jours ou lorsqu'elles
sont épuisées. Ne peut pas être remboursé.",

  "provisioning_play": "play_topup_charge_then_action",
  "provisioning_json_vars": "{
    \"data_gb\": 5,
    \"days\": 7
  }"
}
```

**Exemple 2 : Addon matériel (Location de modem)**

Un addon qui fournit un équipement physique pour un service de fibre existant :

```
{
  "product_name": "Location de Modem WiFi 6",
  "product_slug": "wifi6-modem-rental",
  "category": "addon",
  "service_type": "internet",
  "enabled": true,
  "icon": "fa-solid fa-router",
  "comment": "Ajoutez un modem WiFi 6 au service de fibre -
location",

  "retail_cost": 10.00,
  "wholesale_cost": 3.00,
  "retail_setup_cost": 0.00,
  "wholesale_setup_cost": 45.00,
  "contract_days": 30,

  "residential": true,
  "business": true,
  "customer_can_purchase": true,

  "auto_renew": "true",
  "allow_auto_renew": true,

  "inventory_items_list": "['Rental Modem']",
  "relies_on_list": "",

  "features_list": "WiFi 6 (802.11ax). Double bande 2.4GHz + 5GHz.
Jusqu'à 40 appareils. Contrôles parentaux.",
  "terms": "Location d'équipement. Doit être retourné lors de
l'annulation du service ou des frais de remplacement de 150 £
s'appliquent. L'équipement reste la propriété du fournisseur.",

  "provisioning_play": "play_addon_assign_modem",
  "provisioning_json_vars": "{
    \"device_type\": \"modem_router\",
    \"requires_configuration\": true
  }"
}
```


**Différences clés pour les addons :**

- `category: "addon"` - Appliqué à un service existant, pas autonome

- `contract_days: 0` (virtuel) ou `30` (location récurrente) - Fréquence de facturation
- `inventory_items_list: "[]"` (virtuel) ou `"['Rental Modem']"` (matériel) - Ressources physiques
- `auto_renew: "false"` (unique) ou `"true"` (location) - Comportement récurrent
- `relies_on_list: ""` - Vide signifie s'applique à tout service de type de service correspondant

**Pourquoi les addons matériels ont besoin d'inventaire :**

Les addons matériels nécessitent `inventory_items_list` car :

1. **Suivre l'équipement** - Savoir quel modem est avec quel client
2. **Prévenir les ruptures de stock** - Ne peut pas provisionner l'addon s'il n'y a pas de modems en stock
3. **Récupération** - Lorsque le client annule, savoir quel équipement récupérer
4. **Suivi des coûts** - Lier le coût de gros à un numéro de série spécifique
5. **Amortissement** - Suivre la valeur de l'équipement pendant la période de location
6. **Garantie** - Identifier les unités défectueuses par numéro de série

**Flux de provisionnement des addons avec inventaire :**

Lorsqu'un client ajoute "Location de Modem WiFi 6" à son service de fibre :

1. **Addon sélectionné** - Le client clique sur "Ajouter au service"
2. **Sélecteur d'inventaire apparaît** - Tout comme les services autonomes :
3. **Paiement traité** - 10,00 £ de location mensuelle facturée
4. **Modem attribué** - Inventaire mis à jour :
   - `service_id` : Lié au service de fibre
   - `customer_id` : Lié au client
   - `item_state` : "Assigné"
5. **Expédition déclenchée** - Système de réalisation notifié pour expédier le modem
6. **Installation** - Le client reçoit le modem, le branche dans l'ONT

7. **Facturation récurrente** - 10 £/mois facturés jusqu'à annulation de l'addon

**Déprovisionnement des addons matériels :**

Lorsque le client annule la location du modem :

1. **Annulation initiée** - Le client clique sur "Retirer l'addon"
2. **Processus de retour commencé** :
   - Email envoyé avec des instructions de retour
   - Étiquette d'expédition prépayée générée
   - Période de grâce de 14 jours avant pénalité
3. **Équipement retourné** :
   - Inventaire mis à jour : `item_state` = "In Stock" (après remise à neuf)
   - Ou `item_state` = "Damaged" (s'il est défectueux)
   - Lié au prochain client une fois remis à neuf
4. **Pas de retour** :
   - Après 14 jours, des frais de remplacement de 150 £ sont facturés
   - Inventaire marqué : `item_state` = "Lost"
   - Coût de gros (45 £) + valeur de remplacement récupérée

**Tarification des addons :**

Les addons peuvent être tarifés différemment des services autonomes :

- Les addons virtuels n'ont généralement pas de frais de mise en place
- Les addons matériels peuvent avoir des frais de mise en place de gros pour l'équipement
- Les addons de location récurrents utilisent `contract_days` pour la fréquence de facturation

# Étape 2 : Le processus de provisionnement

Lorsqu'un client commande le produit "Mobile prépayé 20 Go", OmniCRM orchestre le provisionnement via Ansible.

# Diagramme de flux de provisionnement

Commandes du client → Sélection de l'inventaire → Job de provisionnement créé ↓ ↓ Paiement autorisé ← Variables assemblées ← Playbook Ansible exécuté ↓ ↓ Enregistrement de service créé → Configuration du compte OCS → Inventaire attribué → Service actif

# Flux de provisionnement étape par étape

**1. Le client initie la commande**

Depuis la page client :

- Le personnel clique sur "Ajouter un service"
- Sélectionne "Mobile prépayé 20 Go" dans le carrousel de produits
- Détails du produit et tarification affichés

**2. Sélection de l'inventaire**

Le système invite à sélectionner l'inventaire requis :

- **Carte SIM** - Le menu déroulant montre les cartes SIM disponibles en stock
  - Exemple : "SIM-00123 - ICCID : 8944..."
- **Numéro mobile** - Le menu déroulant montre les numéros de téléphone disponibles
  - Exemple : "+44 7700 900123"

Le personnel ou le client sélectionne des articles dans l'inventaire disponible.

**3. Confirmation des prix**

Le système affiche le prix final :

- Coût de mise en place : 0,00 £ (activation gratuite)
- Coût mensuel : 15,00 £
- Montant dû aujourd'hui : 15,00 £ (premier mois)
- Date de renouvellement : 30 jours à partir d'aujourd'hui

Si le prompt de renouvellement automatique est activé, le client choisit :

☐ Renouveler automatiquement ce service tous les 30 jours

**4. Bouton de provisionnement cliqué**

Lorsque "Provisionner" est cliqué, l'API :

- Crée un enregistrement `Provision` avec le statut "En cours" (status=1)
- Fusionne les variables du produit + demande + sélections d'inventaire
- Génère un thread en arrière-plan pour exécuter le playbook Ansible
- Renvoie `provision_id` à l'interface utilisateur pour le suivi du statut

**5. Variables assemblées**

Le système fusionne les variables de plusieurs sources :

**Du produit :**

```
{
  "days": 30,
  "data_gb": 20,
  "voice_minutes": "unlimited",
  "sms_count": "unlimited"
}
```

**De la demande :**

```
{
  "product_id": 42,
  "customer_id": 123,
  "SIM Card": 5001,
  "Mobile Number": 5002
}
```

**Ajouté par le système :**

```
{
  "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "initiating_user": 7
}
```


**Variables finales transmises à Ansible :**


```
{
    "product_id": 42,
    "customer_id": 123,
    "SIM Card": 5001,
    "Mobile Number": 5002,
    "days": 30,
    "data_gb": 20,
    "voice_minutes": "unlimited",
    "sms_count": "unlimited",
    "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
    "initiating_user": 7
}
```


**6. Exécution du playbook Ansible**

Le playbook `play_local_mobile_sim.yaml` s'exécute avec ces variables.

# Comprendre le playbook de provisionnement Ansible

Examinez un vrai playbook de provisionnement pour comprendre ce qui se passe en coulisses.

## Exemple de playbook de provisionnement de carte SIM mobile

**Emplacement :** `OmniCRM-API/Provisioners/plays/play_local_mobile_sim.yaml`

**Structure de haut niveau :**

```yaml
- name: Provisionnement de carte SIM mobile
  hosts: localhost
  gather_facts: no
  become: False

  tasks:
    - name: Bloc principal
      block:
        # 1. Charger la configuration
        # 2. Récupérer les détails du produit depuis l'API
        # 3. Récupérer les détails du client depuis l'API
        # 4. Récupérer les détails de l'inventaire depuis l'API
        # 5. Créer un compte dans OCS (CGRateS)
        # 6. Ajouter des soldes et des allocations à OCS
        # 7. Créer un enregistrement de service dans le CRM
        # 8. Attribuer l'inventaire au service
        # 9. Enregistrer les transactions
        # 10. Envoyer des notifications de bienvenue

      rescue:
        # Rollback en cas d'échec
        # - Supprimer le compte OCS
        # - Libérer l'inventaire
        # - Enregistrer l'erreur
```

**Détail du walkthrough du playbook :**

**Tâche 1 : Charger la configuration**

```yaml
- name: Inclure les vars de crm_config
  ansible.builtin.include_vars:
    file: "../../crm_config.yaml"
    name: crm_config
```

Charge la configuration système, y compris :

- URL et identifiants OCS/CGRateS
- URL de base du CRM
- Configuration du locataire

**Tâche 2 : Récupérer les détails du produit**

```
- name: Obtenir les informations sur le produit depuis l'API CRM
  uri:
    url: "{{ crm_config.crm.base_url }}/crm/product/product_id/{{
product_id }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
    return_content: yes
  register: api_response_product
```

**Ce que cela fait :**

- Appelle `GET /crm/product/product_id/42`
- Récupère la définition complète du produit
- Stocke dans la variable `api_response_product`

**Pourquoi :** Même si nous avons `provisioning_json_vars` du produit, nous récupérons le produit complet pour obtenir :

- Tarification la plus récente (peut avoir changé depuis le début de la commande)
- Nom du produit pour le nommage du service
- Liste des fonctionnalités pour la documentation
- Coûts de gros pour le suivi des marges

**Tâche 3 : Définir les faits du package**

```
- name: Définir les faits du package
  set_fact:
    package_name: "{{ api_response_product.json.product_name }}"
    monthly_cost: "{{ api_response_product.json.retail_cost }}"
    setup_cost: "{{ api_response_product.json.retail_setup_cost
}}"
```

Extrait les valeurs couramment utilisées dans des variables simples pour la lisibilité.

**Tâche 4 : Récupérer les détails de l'inventaire**

```
- name: Obtenir les informations de la carte SIM depuis l'API CRM
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ hostvars[inventory_hostname]['SIM
Card'] }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
  register: api_response_sim

- name: Définir l'IMSI à partir de la réponse de l'inventaire
  set_fact:
    imsi: "{{ api_response_sim.json.itemtext2 }}"
    iccid: "{{ api_response_sim.json.itemtext1 }}"
```

**Ce que cela fait :**

- Recherche l'ID d'inventaire de la carte SIM 5001
- Récupère les détails de la SIM :
  - `itemtext1` = ICCID (numéro de carte SIM)
  - `itemtext2` = IMSI (identité de l'abonné)
- Fait de même pour l'inventaire du numéro mobile (récupère le numéro de téléphone)

**Pourquoi cela compte :**

- L'IMSI est nécessaire pour provisionner l'abonné dans le HSS (Home Subscriber Server)
- L'ICCID est enregistré dans les notes de service pour le dépannage
- Le numéro de téléphone (MSISDN) est affiché au client et utilisé pour le routage

**Tâche 5 : Générer l'UUID du service**

```yaml
- name: Générer le fait UUID
  set_fact:
    uuid: "{{ 99999999 | random | to_uuid }}"

- name: Définir l'UUID du service
  set_fact:
    service_uuid: "Local_Mobile_SIM_{{ uuid[0:8] }}"
```

**Ce que cela fait :**

- Génère un UUID aléatoire
- Crée un service_uuid comme `Local_Mobile_SIM_a3f2c1d8`

**Pourquoi :**

- L'UUID du service est l'identifiant unique dans OCS/CGRateS
- Utilisé pour toutes les opérations de facturation
- Doit être globalement unique à travers tous les services

**Tâche 6 : Créer un compte OCS**

```
- name: Créer un compte dans OCS
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    headers:
      Content-Type: "application/json"
    body:
      {
        "method": "ApierV2.SetAccount",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Account": "{{ service_uuid }}",
          "ActionPlanIds": [],
          "ExtraOptions": {
            "AllowNegative": false,
            "Disabled": false
          },
          "ReloadScheduler": true
        }]
      }
  register: ocs_create_response
```

**Ce que cela fait :**

- Appelle l'API JSON-RPC de CGRateS
- Crée un nouveau compte avec service_uuid
- Définit le compte comme actif (non désactivé)
- Empêche le solde négatif (mode prépayé)

**Pourquoi :**

- Le compte OCS est où toutes les opérations de facturation se produisent
- Les soldes (données, voix, SMS, argent) sont stockés ici
- L'utilisation est suivie et évaluée en temps réel

**Tâche 7 : Ajouter un solde de données**

```
- name: Ajouter un solde de données de 20 Go
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.AddBalance",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Account": "{{ service_uuid }}",
          "BalanceType": "*data",
          "Balance": {
            "ID": "DATA_20GB_Monthly",
            "Value": 21474836480,
            "ExpiryTime": "+720h",
            "Weight": 10,
            "DestinationIDs": "*any"
          }
        }]
      }
```

**Ce que cela fait :**

- Ajoute un solde de données de 20 Go au compte
- Valeur : 21474836480 octets (20 * 1024 * 1024 * 1024)
- Expire dans 720 heures (30 jours)
- Poids 10 (poids plus élevé consommé en premier)

**Tâche 8 : Ajouter des voix illimitées et des SMS**

```
- name: Ajouter une voix illimitée
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.AddBalance",
        "params": [{
          "Account": "{{ service_uuid }}",
          "BalanceType": "*voice",
          "Balance": {
            "ID": "VOICE_Unlimited",
            "Value": 999999999,
            "ExpiryTime": "+720h"
          }
        }]
      }
```

- Ajoute 999 999 999 secondes de voix (essentiellement illimité)
- Expire dans 30 jours

**Tâche 9 : Créer un enregistrement de service dans le CRM**

```
- name: Ajouter le service via l'API
  uri:
    url: "{{ crm_config.crm.base_url }}/crm/service/"
    method: PUT
    body_format: json
    headers:
      Authorization: "Bearer {{ access_token }}"
    body:
      {
        "customer_id": "{{ customer_id }}",
        "product_id": "{{ product_id }}",
        "service_name": "Mobile - {{ phone_number }}",
        "service_type": "mobile",
        "service_uuid": "{{ service_uuid }}",
        "service_status": "Active",
        "service_provisioned_date": "{{ provision_datetime }}",
        "retail_cost": "{{ monthly_cost }}",
        "wholesale_cost": "{{
api_response_product.json.wholesale_cost }}",
        "icon": "fa-solid fa-sim-card"
      }
  register: service_creation_response
```

**Ce que cela crée :**

- Enregistrement de service lié au client
- Lien vers OCS via `service_uuid`
- Stocke les coûts de détail et de gros
- Définit le statut sur "Actif"
- Renvoie `service_id` pour les opérations ultérieures

**Tâche 10 : Attribuer l'inventaire au service**

```
- name: Assign SIM Card to Service
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ hostvars[inventory_hostname]['SIM
Card'] }}"
    method: PATCH
    body_format: json
    headers:
      Authorization: "Bearer {{ access_token }}"
    body:
      {
        "service_id": "{{
service_creation_response.json.service_id }}",
        "customer_id": "{{ customer_id }}",
        "item_state": "Assigned"
      }
```

**Ce que cela fait :**

- Met à jour l'enregistrement d'inventaire de la carte SIM
- Définit `service_id` pour lier la SIM au service
- Change l'état de "In Stock" à "Assigned"
- Répète pour l'inventaire du numéro mobile

**Pourquoi :**

- Suit quelle SIM est attribuée à quel client
- Empêche la double attribution de l'inventaire
- Permet le reporting et l'audit de l'inventaire

**Tâche 11 : Enregistrer la transaction de coût de mise en place**

```
- name: Ajouter la transaction de coût de mise en place
  uri:
    url: "{{ crm_config.crm.base_url }}/crm/transaction/"
    method: PUT
    body_format: json
    headers:
      Authorization: "Bearer {{ access_token }}"
    body:
      {
        "customer_id": "{{ customer_id }}",
        "service_id": "{{
service_creation_response.json.service_id }}",
        "title": "{{ package_name }} - Setup",
        "description": "Frais d'activation",
        "retail_cost": "{{ setup_cost }}",
        "wholesale_cost": "{{
api_response_product.json.wholesale_setup_cost }}"
      }
```

**Ce que cela fait :**

- Enregistre des frais de mise en place de 0,00 £ au client (détail)
- Enregistre un coût de gros de 1,00 £
- Crée un enregistrement de transaction pour la facturation

**Tâche 12 : Bloc de secours (Gestion des erreurs)**

```
rescue:
  - name: Supprimer le compte dans OCS en cas d'échec
    uri:
      url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
      method: POST
      body:
        {
          "method": "ApierV2.RemoveAccount",
          "params": [{
            "Account": "{{ service_uuid }}"
          }]
        }

  - name: Échouer le provisionnement
    fail:
      msg: "Le provisionnement a échoué, le compte OCS a été
annulé"
```

**Ce que cela fait :**

- Si une tâche échoue, le bloc de secours s'exécute
- Supprime le compte OCS qui a été partiellement créé
- Libère l'inventaire en retour à "In Stock"
- Échoue le job de provisionnement avec un message d'erreur

**Pourquoi :**

- Empêche les comptes orphelins dans OCS
- Garantit un rollback propre en cas d'erreurs
- Maintient la cohérence des données

# Provisionnement complet : Ce qui a été créé

Après un provisionnement réussi, le système a :

**1. Compte OCS (CGRateS) :**

- ID de compte : `Local_Mobile_SIM_a3f2c1d8`
- Soldes :

    - 20 Go de données (expire dans 30 jours)
    - Voix illimitée (999M secondes, expire dans 30 jours)
    - SMS illimités (999M messages, expire dans 30 jours)

**2. Enregistrement de service CRM :**

- ID de service : 1234
- Client : John Doe (customer_id : 123)
- Produit : Mobile prépayé 20 Go (product_id : 42)
- Nom du service : "Mobile - +44 7700 900123"
- UUID du service : `Local_Mobile_SIM_a3f2c1d8`
- Statut : Actif
- Coût mensuel : 15,00 £ (détail), 5,00 £ (gros)
- Profit : 10,00 £/mois

**3. Attributions d'inventaire :**

- Carte SIM 5001 : Assignée au service 1234, client 123
- Numéro mobile 5002 : Assigné au service 1234, client 123

**4. Enregistrements de transaction :**

- Transaction de coût de mise en place créée
- Charge du premier mois enregistrée

**5. Le client peut maintenant :**

- Voir le service dans le portail de libre-service
- Voir le solde de 20 Go de données
- Passer des appels et envoyer des SMS
- Recharger ou ajouter des addons
- Voir l'utilisation en temps réel

# Étape 3 : Ajout d'addons et de recharges

Après qu'un service soit actif, les clients peuvent acheter des addons pour améliorer leur service.

## Flux de provisionnement des addons

Supposons que le client ait utilisé 18 Go de son allocation de 20 Go et souhaite acheter l'addon "5 Go de données supplémentaires".

**1. Le client navigue vers le service**

- Ouvre la page "Mobile - +44 7700 900123"
- Voit l'utilisation actuelle : 18 Go de 20 Go utilisés (90 %)
- Clique sur "Ajouter un addon" ou "Recharger"

**2. Le système filtre les addons disponibles**

N'affiche que les addons où :

- `category = "addon"`
- `service_type = "mobile"` (correspond au type de service)
- `residential = true` (si le client est résident)
- `enabled = true`

Le client voit : "5 Go de données supplémentaires - 5,00 £"

**3. Le client sélectionne l'addon**

- Clique sur "5 Go de données supplémentaires"
- Confirme l'achat pour 5,00 £
- Le système capture l'autorisation de paiement

**4. Provisionnement de l'addon initié**

Le système appelle `play_topup_charge_then_action.yaml` avec des variables :

```
{
    "product_id": 43,  # Produit 5 Go de données supplémentaires
    "customer_id": 123,
    "service_id": 1234,  # Service existant
    "access_token": "eyJ...",
    "data_gb": 5,  # À partir de provisioning_json_vars
    "days": 7      # À partir de provisioning_json_vars
}
```

**Différence clé par rapport aux autonomes :**

- `service_id` est inclus (service existant à modifier)
- Aucun inventaire requis
- Pas de création de service (modifie l'existant)

# Walkthrough du playbook de provisionnement des addons

### Tâche 1 : Récupérer les détails du service

```
- name: Obtenir les informations du service depuis l'API CRM
  uri:
    url: "http://localhost:5000/crm/service/service_id/{{
service_id }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
  register: api_response_service

- name: Définir les faits du service
  set_fact:
    service_uuid: "{{ api_response_service.json.service_uuid }}"
    customer_id: "{{ api_response_service.json.customer_id }}"
```

**Pourquoi :**

- Besoin de `service_uuid` pour ajouter un solde au bon compte OCS
- Vérifie que le service existe et est actif

- S'assure que le service appartient au client

**Tâche 2 : Facturer le client**

```yaml
- name: Obtenir le mode de paiement par défaut du client
  uri:
    url: "http://localhost:5000/api/payments/methods/default?
customer_id={{ customer_id }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
  register: api_response_payment_method

- name: Obtenir l'ID du mode de paiement par défaut
  set_fact:
    payment_method_id: "{{
api_response_payment_method.json.data.payment_method_id }}"

- name: Facturer le client
  uri:
    url: "http://localhost:5000/api/payments/charge"
    method: POST
    body_format: json
    headers:
      Authorization: "Bearer {{ access_token }}"
    body:
      {
        "customer_id": "{{ customer_id | int }}",
        "amount": 5.00,
        "currency": "USD",
        "payment_method_id": "{{ payment_method_id }}",
        "metadata": {
          "description": "5 Go de données supplémentaires",
          "service_id": "{{ service_id | int }}",
          "product_id": "{{ product_id | int }}",
          "invoice": true
        }
      }
  register: charge_response

- name: Vérifier que le paiement a réussi
  assert:
    that:
      - charge_response.json.success == true
```

**Ce que cela fait :**

- Trouve le mode de paiement par défaut du client sur Stripe
- Facture 5,00 £ à la carte
- Enregistre un coût de gros de 1,50 £ pour le suivi des marges
- Crée une transaction liée au service
- Ajoute à la prochaine facture
- Échoue le provisionnement si le paiement échoue

**Pourquoi facturer d'abord :**

- Aucun crédit livré tant que le paiement n'est pas confirmé
- Prévenir la fraude
- Faire correspondre le paiement au provisionnement de l'addon

**Tâche 3 : Ajouter un solde de données à OCS**

```
- name: Ajouter un solde de données de 5 Go
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.AddBalance",
        "params": [{
          "Account": "{{ service_uuid }}",
          "BalanceType": "*data",
          "Balance": {
            "ID": "DATA_5GB_Boost_{{ uuid }}",
            "Value": 5368709120,
            "ExpiryTime": "+168h",
            "Weight": 20
          }
        }]
      }
```

**Ce que cela fait :**

- Ajoute 5 Go (5368709120 octets) au compte
- Expire dans 168 heures (7 jours)

- Poids 20 (poids plus élevé consommé en premier - boost avant l'allocation mensuelle)

**Solde du client après l'addon :**

- Mensuel d'origine : 2 Go restants (expire dans 25 jours)
- Nouveau boost : 5 Go (expire dans 7 jours)
- Total disponible : 7 Go
- Ordre d'utilisation : Boost consommé en premier, puis mensuel

**Tâche 4 : Enregistrer la transaction**

```
- name: Ajouter la transaction d'addon
  uri:
    url: "http://localhost:5000/crm/transaction/"
    method: PUT
    body_format: json
    headers:
      Authorization: "Bearer {{ access_token }}"
    body:
      {
        "customer_id": "{{ customer_id }}",
        "service_id": "{{ service_id }}",
        "title": "5 Go de données supplémentaires",
        "description": "Données supplémentaires de 5 Go valables
pendant 7 jours",
        "retail_cost": 5.00,
        "wholesale_cost": 1.50
      }
```

**Ce que cela fait :**

- Enregistre une charge de 5,00 £ au client
- Enregistre un coût de gros de 1,50 £
- Lien de la transaction au service pour reporting

## Résumé complet du flux d'addon

1. Le client sélectionne l'addon dans la liste filtrée

2. Paiement autorisé et facturé
3. Solde de données ajouté au compte OCS
4. Transaction enregistrée dans le CRM
5. Le client voit immédiatement le solde mis à jour : 7 Go disponibles

**Suivi financier :**

- Charge mensuelle du service : 15 £ détail, 5 £ gros
- Achat d'addon : 5 £ détail, 1,50 £ gros

## Renouvellement automatique : Addons récurrents

Certains addons peuvent être configurés pour se renouveler automatiquement (plans de données mensuels, abonnements, etc.).

**Configuration du produit :**

```
{
  "product_name": "Plan de données mensuel de 10 Go",
  "category": "addon",
  "retail_cost": 10.00,
  "contract_days": 30,
  "auto_renew": "true",
  "provisioning_play": "play_topup_charge_then_action"
}
```

**Le provisionnement crée un ActionPlan :**

```
- name: Créer un ActionPlan pour le renouvellement automatique
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body:
      {
        "method": "ApierV1.SetActionPlan",
        "params": [{
          "Id": "ServiceID_{{ service_uuid }}__ProductID_{{
product_id }}__MonthlyRenewal",
          "ActionPlan": [{
            "ActionsId": "Action_{{ product_slug }}",
            "Years": "*any",
            "Months": "*any",
            "MonthDays": "*any",
            "WeekDays": "*any",
            "Time": "00:00:00",
            "Weight": 10
          }],
          "Overwrite": false,
          "ReloadScheduler": true
        }]
      }
```

**Ce que cela fait :**

- Crée une tâche planifiée dans OCS
- Exécute `Action_{{ product_slug }}` tous les 30 jours
- L'action facture le client et réapplique le solde de données
- Continue jusqu'à ce que le client annule

**Gestion des clients :**

- Le client voit "Prochain renouvellement : 1er février 2025 - 10,00 £" dans la vue du service
- Peut cliquer sur "Annuler le renouvellement automatique" pour arrêter les frais futurs
- Peut cliquer sur "Renouveler maintenant" pour appliquer immédiatement l'allocation du mois suivant

# Étape 4 : Déprovisionnement des services

Lorsqu'un client annule un service, le système doit supprimer proprement toutes les ressources.

## Déclencheurs de déprovisionnement

Le déprovisionnement peut être déclenché par :

1. **Annulation du client** - Le client clique sur "Annuler le service"
2. **Action administrative** - Le personnel marque le service pour désactivation
3. **Non-paiement** - Le service expire en raison du manque de renouvellement
4. **Fin de contrat** - Le contrat à durée fixe atteint sa date de fin

## Flux de déprovisionnement

### 1. Le client initie l'annulation

- Navigue vers le service
- Clique sur "Annuler le service"
- Le système invite : "Êtes-vous sûr ? Tout solde restant sera perdu."
- Le client confirme

### 2. Période de grâce (facultatif)

Certains opérateurs mettent en œuvre une période de grâce :

- Service marqué "Annulation en attente"
- Reste actif pendant 7 à 30 jours
- Le client peut annuler l'annulation pendant la période de grâce
- Déprovisionnement automatique après la période de grâce

### 3. Job de déprovisionnement créé

Le système crée un job de provisionnement avec :

```
{
    "action": "deprovision",
    "service_id": 1234,
    "customer_id": 123,
    "service_uuid": "Local_Mobile_SIM_a3f2c1d8"
}
```

Appelle le playbook spécifié dans `service.deprovisioning_play` ou le bloc de secours du playbook original.

**4. Playbook de déprovisionnement Ansible**

```yaml
- name: Déprovisionner le service mobile
  hosts: localhost
  tasks:
    - name: Désactiver le compte OCS
      uri:
        url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
        method: POST
        body:
          {
            "method": "ApierV2.SetAccount",
            "params": [{
              "Account": "{{ service_uuid }}",
              "ExtraOptions": { "Disabled": true }
            }]
          }

    - name: Supprimer les ActionPlans (arrêter les renouvellements
automatiques)
      uri:
        url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
        method: POST
        body:
          {
            "method": "ApierV1.RemoveActionPlan",
            "params": [{
              "Id": "ServiceID_{{ service_uuid }}__*"
            }]
          }

    - name: Mettre à jour le statut du service dans le CRM
      uri:
        url: "http://localhost:5000/crm/service/{{ service_id }}"
        method: PATCH
        body:
          {
            "service_status": "Deactivated",
            "service_deactivate_date": "{{ current_datetime }}"
          }

    - name: Libérer l'inventaire en stock
      uri:
        url: "http://localhost:5000/crm/inventory/inventory_id/{{
sim_card_id }}"
```

```
        method: PATCH
        body:
          {
            "service_id": null,
            "customer_id": null,
            "item_state": "Decommissioned"
          }
```

**Ce que cela fait :**

1. **Désactive le compte OCS** - Arrête toute facturation, utilisation bloquée
2. **Supprime les ActionPlans** - Annule les renouvellements automatiques
3. **Met à jour le service CRM** - Statut "Désactivé", date enregistrée
4. **Libère l'inventaire** - SIM marquée "Déclassée", disponible pour r��utilisation (après remise à neuf)

**5. Post-déprovisionnement**

Le système effectue un nettoyage :

- Le client ne voit plus le service dans le portail de libre-service
- Le service reste dans le CRM pour un reporting historique
- Les transactions et factures sont préservées pour la comptabilité
- L'inventaire peut être remis à neuf et réutilisé
- Le compte OCS peut être archivé après la période de conservation

# Déprovisionnement partiel vs complet

**Déprovisionnement partiel (suspension) :**

- Utilisé pour non-paiement ou suspension temporaire
- Compte OCS désactivé mais pas supprimé
- Soldes préservés
- Peut être réactivé lorsque le paiement est reçu

```
- name: Suspendre le service
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body:
      {
        "method": "ApierV2.SetAccount",
        "params": [{
          "Account": "{{ service_uuid }}",
          "ExtraOptions": { "Disabled": true }
        }]
      }
```

**Déprovisionnement complet (annulation permanente) :**

- Utilisé pour une annulation permanente
- Compte OCS supprimé entièrement
- Soldes perdus
- Ne peut pas être réactivé

```
- name: Supprimer le compte OCS
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body:
      {
        "method": "ApierV2.RemoveAccount",
        "params": [{
          "Account": "{{ service_uuid }}"
        }]
      }
```

# Meilleures pratiques pour la gestion des produits

## Gestion du cycle de vie des produits

**États des produits :**

- `enabled: true` - Produit disponible pour de nouvelles commandes
- `enabled: false` - Produit désactivé, les services existants continuent

**D

# Intégration de Mailjet avec OmniCRM

OmniCRM s'intègre avec **Mailjet** pour gérer toute la communication par e-mail avec les clients et le personnel, garantissant une livraison d'e-mails professionnelle, de marque et fiable pour les e-mails transactionnels et les campagnes marketing.

## Aperçu

L'intégration de Mailjet fournit :

- **E-mails transactionnels automatisés** - Réinitialisations de mot de passe, factures, e-mails de bienvenue, notifications
- **Consentement marketing (double opt-in)** - Opt-in/opt-out auditable pour les e-mails marketing, y compris les flux de confirmation et de ré-autorisation
- **Modèles d'e-mails** - Plus de 10 types d'e-mails préconfigurés avec des modèles Mailjet personnalisables
- **Campagnes marketing** - Campagnes par e-mail segmentées basées sur les données clients (les segments peuvent être construits à partir des données CRM ou d'un téléchargement CSV)
- **Livraison fiable** - Infrastructure d'e-mail professionnelle avec suivi et analyses

OmniCRM **ne** maintient **pas** une synchronisation en direct de la liste de contacts Mailjet. Utilisez **Clients → Tous les Contacts** dans l'interface d'administration (ou `GET /crm/contact/export`) pour exporter des contacts pour une gestion externe de la liste si nécessaire.

# Configuration

Mailjet est configuré dans `OmniCRM-API/crm_config.yaml` sous la section `mailjet`.

## Configuration de base

```
mailjet:
  api_key: your_mailjet_api_key
  api_secret: your_mailjet_api_secret
```

**Obtention des identifiants API :**

1. Créez un compte sur <https://www.mailjet.com>
2. Accédez à **Paramètres du compte → Clés API**
3. Copiez **Clé API** et **Clé secrète**
4. Collez dans `crm_config.yaml`

# Configuration des modèles d'e-mails

OmniCRM utilise **10 types distincts de modèles d'e-mails** pour les communications automatisées. Chaque modèle est configuré avec :

- **from_email** - Adresse e-mail de l'expéditeur
- **from_name** - Nom d'affichage de l'expéditeur
- **template_id** - ID du modèle Mailjet (numérique)
- **subject** - Ligne d'objet de l'e-mail

## Types de modèles et configuration

### E-mail de bienvenue client

Envoyé lorsqu'un nouveau compte client est créé.

```
api_crmCommunicationCustomerWelcome:
  from_email: "support@yourcompany.com"
  from_name: "Support de Votre Entreprise"
  template_id: 5977509
  subject: "Bienvenue chez Votre Entreprise"
```

**Quand envoyé :**

- Un nouveau client s'inscrit via le portail d'auto-assistance
- Le personnel crée un nouveau compte client
- Le client active le service pour la première fois

**Variables de modèle disponibles :**

- `{{ var:customer_name }}` - Nom complet du client
- `{{ var:email }}` - Adresse e-mail du client
- `{{ var:company_name }}` - Nom de votre entreprise
- `{{ var:login_url }}` - Lien vers le portail d'auto-assistance
- `{{ var:support_url }}` - Lien vers la page de support

**E-mail de facture client**

Envoyé lorsqu'une facture est générée et prête à être payée.

```
api_crmCommunicationCustomerInvoice:
  from_email: "billing@yourcompany.com"
  from_name: "Facturation de Votre Entreprise"
  template_id: 6759851
  subject: "Votre Facture - "
```

**Quand envoyé :**

- Facture générée automatiquement pour la période de facturation
- Facture manuelle créée par le personnel
- Le client demande une copie de la facture

**Variables de modèle disponibles :**

- `{{ var:customer_name }}` - Nom complet du client
- `{{ var:invoice_number }}` - ID/numéro de la facture
- `{{ var:invoice_date }}` - Date d'émission de la facture
- `{{ var:due_date }}` - Date d'échéance du paiement
- `{{ var:total_amount }}` - Montant total dû
- `{{ var:invoice_url }}` - Lien pour voir/télécharger le PDF de la facture
- `{{ var:pay_url }}` - Lien pour payer la facture en ligne

**Pièce jointe de la facture :**

Le PDF de la facture est automatiquement joint à l'e-mail.

**Rappel de facture client**

Envoyé pour rappeler aux clients les factures en retard.

```
api_crmCommunicationCustomerInvoiceReminder:
  from_email: "billing@yourcompany.com"
  from_name: "Facturation de Votre Entreprise"
  template_id: 6759852
  subject: "Rappel de Paiement - Facture En Retard"
```

**Quand envoyé :**

- La facture est en retard de X jours (configurable)
- Rappel manuel déclenché par le personnel
- Flux de rappel automatisé (si configuré)

**Variables de modèle disponibles :**

- `{{ var:customer_name }}`
- `{{ var:invoice_number }}`
- `{{ var:due_date }}`
- `{{ var:days_overdue }}`
- `{{ var:total_amount }}`
- `{{ var:pay_url }}`

### E-mail de bienvenue utilisateur du personnel

Envoyé lorsqu'un nouveau compte utilisateur du personnel est créé.

```
api_crmCommunicationUserWelcome:
  from_email: "admin@yourcompany.com"
  from_name: "Admin de Votre Entreprise"
  template_id: 5977510
  subject: "Bienvenue dans l'Équipe"
```

**Quand envoyé :**

- L'administrateur crée un nouvel utilisateur du personnel
- Le bouton "Envoyer un e-mail de bienvenue" est cliqué dans la gestion des utilisateurs

**Variables de modèle disponibles :**

- `{{ var:user_name }}` - Nom complet de l'utilisateur du personnel
- `{{ var:email }}` - E-mail de l'utilisateur du personnel
- `{{ var:role }}` - Rôle(s) assigné(s)
- `{{ var:login_url }}` - Lien vers la connexion du portail administrateur
- `{{ var:temp_password }}` - Mot de passe temporaire (le cas échéant)
- `{{ var:support_email }}` - Contact de support informatique

### Réinitialisation de mot de passe utilisateur

Envoyé lorsqu'un utilisateur demande à réinitialiser son mot de passe.

```
api_crmCommunicationUserPasswordReset:
  from_email: "noreply@yourcompany.com"
  from_name: "Sécurité de Votre Entreprise"
  template_id: 5977511
  subject: "Demande de Réinitialisation de Mot de Passe"
```

**Quand envoyé :**

- L'utilisateur clique sur "Mot de passe oublié" sur la page de connexion

- L'utilisateur soumet une demande de réinitialisation de mot de passe

**Variables de modèle disponibles :**

- `{{ var:user_name }}`
- `{{ var:reset_url }}` - Lien de réinitialisation de mot de passe limité dans le temps (typiquement 1 heure)
- `{{ var:expiry_time }}` - Quand le lien de réinitialisation expire

**Note de sécurité :**

Les liens de réinitialisation expirent après la période de temps configurée (par défaut 1 heure).

**Succès de réinitialisation de mot de passe utilisateur**

Envoyé pour confirmer que le mot de passe a été changé avec succès.

```
api_crmCommunicationUserPasswordResetSuccess:
  from_email: "noreply@yourcompany.com"
  from_name: "Sécurité de Votre Entreprise"
  template_id: 5977512
  subject: "Mot de Passe Changé Avec Succès"
```

**Quand envoyé :**

- L'utilisateur termine avec succès la réinitialisation du mot de passe
- Immédiatement après que le nouveau mot de passe soit défini

**Variables de modèle disponibles :**

- `{{ var:user_name }}`
- `{{ var:change_date }}` - Date/heure à laquelle le mot de passe a été changé
- `{{ var:ip_address }}` - Adresse IP du changement (facultatif)
- `{{ var:support_email }}` - Contact si le changement était non autorisé

**Changement de mot de passe utilisateur**

Envoyé lorsqu'un utilisateur change son mot de passe depuis les paramètres.

```
api_crmCommunicationUserPasswordChange:
  from_email: "noreply@yourcompany.com"
  from_name: "Sécurité de Votre Entreprise"
  template_id: 5977513
  subject: "Notification de Changement de Mot de Passe"
```

**Quand envoyé :**

- L'utilisateur change le mot de passe depuis son profil/paramètres
- L'administrateur réinitialise le mot de passe de l'utilisateur

**Variables de modèle disponibles :**

- `{{ var:user_name }}`
- `{{ var:change_date }}`
- `{{ var:changed_by }}` - "Soit" ou nom de l'administrateur
- `{{ var:support_email }}`

**Vérification de l'e-mail**

Envoyé pour vérifier l'adresse e-mail d'un utilisateur.

```
api_crmCommunicationEmailVerification:
  from_email: "noreply@yourcompany.com"
  from_name: "Votre Entreprise"
  template_id: 5977514
  subject: "Vérifiez Votre Adresse E-mail"
```

**Quand envoyé :**

- Nouveau compte créé (client ou personnel)
- L'utilisateur change d'adresse e-mail
- Vérification de l'e-mail requise pour des raisons de sécurité

**Variables de modèle disponibles :**

- `{{ var:user_name }}`
- `{{ var:verification_url }}` - Lien pour vérifier l'e-mail
- `{{ var:verification_code }}` - Code à entrer manuellement (alternative au lien)

**Notification d'expiration de solde**

Envoyé lorsque le solde/limite de service d'un client a expiré.

```
api_crmCommunicationsBalanceExpired:
  from_email: "support@yourcompany.com"
  from_name: "Support de Votre Entreprise"
  template_id: 5977515
  subject: "Votre Solde de Service a Expiré"
```

**Quand envoyé :**

- Le solde prépayé expire
- La date de renouvellement de l'allocation mensuelle est passée
- La date d'expiration du service est atteinte

**Variables de modèle disponibles :**

- `{{ var:customer_name }}`
- `{{ var:service_name }}` - Nom du service expiré
- `{{ var:expiry_date }}`
- `{{ var:balance_type }}` - "Données", "Voix", "Monétaire", etc.
- `{{ var:renewal_url }}` - Lien pour renouveler/approvisionner

**Alerte de solde bas**

Envoyé lorsque le solde d'un client tombe en dessous du seuil configuré.

```
api_crmCommunicationsBalanceLow:
  from_email: "support@yourcompany.com"
  from_name: "Support de Votre Entreprise"
  template_id: 5977516
  subject: "Alerte de Solde Bas"
```

**Quand envoyé :**

- Le solde tombe en dessous du seuil (par exemple, 20 % restant)
- Configuré dans le plan de service ou OCS
- La surveillance en temps réel déclenche l'alerte

**Variables de modèle disponibles :**

- `{{ var:customer_name }}`
- `{{ var:service_name }}`
- `{{ var:current_balance }}`
- `{{ var:threshold }}`
- `{{ var:balance_type }}`
- `{{ var:topup_url }}` - Lien pour ajouter du solde

**E-mail de livraison eSIM**

Envoyé lorsqu'une eSIM est provisionnée et prête à être activée.

```
api_crmCommunicationEsimDelivery:
  from_email: "support@yourcompany.com"
  from_name: "Support de Votre Entreprise"
  template_id: 0
  subject: "Votre eSIM est Prête"
```

**Quand envoyé :**

- eSIM provisionnée lors de l'auto-inscription
- eSIM assignée à un service client

**Variables de modèle disponibles :**

- `{{ var:customer_name }}` - Nom complet du client
- `{{ var:iccid }}` - eSIM ICCID
- `{{ var:activation_code }}` - Chaîne de code d'activation LPA
- `{{ var:has_qr_code }}` - Booléen, vrai si l'image du code QR est jointe
- `{{ var:has_activation_code }}` - Booléen, vrai si le code d'activation est disponible

**Fallback intégré :**

Lorsque `template_id` est `0` (ou que la clé du modèle est manquante dans la configuration), OmniCRM utilise un e-mail HTML de marque intégré avec le code QR eSIM rendu en ligne. Les couleurs et le nom de la marque sont contrôlés par la section `branding` dans `crm_config.yaml`. Voir Guide de configuration pour plus de détails.

# Création de modèles d'e-mails Mailjet

Pour chaque type d'e-mail, vous devez créer un modèle correspondant dans Mailjet.

## Étape 1 : Créer un modèle dans Mailjet

1. Connectez-vous au tableau de bord Mailjet
2. Accédez à **Modèles d'e-mails → Modèles transactionnels**
3. Cliquez sur **Créer un nouveau modèle**
4. Choisissez **Codez votre propre modèle** (pour les utilisateurs avancés) ou **Utilisez le constructeur de modèles**

## Étape 2 : Concevoir le modèle

Utilisez le constructeur par glisser-déposer de Mailjet ou l'éditeur HTML pour concevoir votre e-mail.

**Éléments essentiels :**

- **En-tête** - Logo de l'entreprise et branding
- **Salutation** - Personnalisée avec `{{ var:customer_name }}` ou `{{ var:user_name }}`
- **Contenu** - Corps principal du message
- **Variables** - Insérez les variables du modèle de la liste ci-dessus
- **Appel à l'action** - Boutons/liens pour les actions des utilisateurs
- **Pied de page** - Lien de désinscription, adresse de l'entreprise, contact de support

**Exemple de modèle (Réinitialisation de mot de passe) :**

```
<!DOCTYPE html>
<html>
<head>
    <style>
        body { font-family: Arial, sans-serif; }
        .button { background-color: #4CAF50; color: white;
padding: 14px 28px; }
    </style>
</head>
<body>
    ![Logo](https://yourcompany.com/logo.png)

    <h2>Demande de Réinitialisation de Mot de Passe</h2>

    <p>Bonjour {{ var:user_name }},</p>

    <p>Nous avons reçu une demande pour réinitialiser votre mot de
passe. Cliquez sur le bouton ci-dessous pour créer un nouveau mot
de passe :</p>

    <a href="{{ var:reset_url }}" class="button">Réinitialiser le
Mot de Passe</a>

    <p>Ce lien expire dans {{ var:expiry_time }}.</p>

    <p>Si vous n'avez pas demandé cela, veuillez ignorer cet e-
mail.</p>

    <hr>
    <p style="font-size: 12px; color: #666;">
        Votre Entreprise | support@yourcompany.com<br/>
        123 Rue des Affaires, Ville, Pays
    </p>
</body>
</html>
```

## Étape 3 : Obtenir l'ID du modèle

1. Enregistrez le modèle dans Mailjet
2. Notez l'**ID du modèle** (numérique, par exemple, `5977509`)
3. Copiez cet ID dans `crm_config.yaml`

## Étape 4 : Tester le modèle

1. Dans Mailjet, utilisez la fonction **E-mail de test**
2. Fournissez des valeurs d'exemple pour toutes les variables
3. Envoyez un e-mail de test à vous-même
4. Vérifiez le formatage, les liens et le branding

## Étape 5 : Configurer dans OmniCRM

Ajoutez la configuration du modèle à `crm_config.yaml` :

```
mailjet:
  api_key: your_api_key
  api_secret: your_secret

  api_crmCommunicationUserPasswordReset:
    from_email: "noreply@yourcompany.com"
    from_name: "Sécurité de Votre Entreprise"
    template_id: 5977511
    subject: "Demande de Réinitialisation de Mot de Passe"
```

Redémarrez l'API OmniCRM pour que les modifications prennent effet :

```
cd OmniCRM-API
sudo systemctl restart omnicrm-api
```

# Exportation de contacts (CRM)

Pour le reporting, la conformité ou l'importation manuelle dans Mailjet (ou un autre ESP), les administrateurs peuvent parcourir et exporter des contacts depuis le CRM sans synchronisation en direct de la liste Mailjet.

## Interface d'administration

**Clients → Tous les Contacts** (`/contacts`) fournit un tableau paginé avec recherche, filtres (`marketing_consent_status`, `customer_status`), et **Exporter**

**CSV**. L'exportation utilise les mêmes filtres et recherches que le tableau.

## API

Les deux points de terminaison nécessitent la permission `VIEW_CUSTOMER_CONTACT`.

| Méthode | Chemin | Objectif |
|---|---|---|
| `GET` | `/crm/contact/search` | Liste de contacts paginée avec `search`, `page`, `per_page`, `sort`, `order`, `filters` (JSON) |
| `GET` | `/crm/contact/export` | Téléchargement CSV avec les mêmes paramètres de requête (max 50 000 lignes) |

**Colonnes CSV :** `contact_id`, `customer_id`, `customer_name`, `customer_status`, nom, e-mail, téléphone, titre du poste, `contact_types`, `marketing_consent_status`, `marketing_consent_granted_at`, `created`, `last_modified`.

Filtrez sur `marketing_consent_status=granted` lors de la préparation d'une liste pour un usage marketing. Importez le CSV dans Mailjet (ou votre ESP) en utilisant leur flux d'importation de contacts.

Implémentation : `OmniCRM-API/services/contact_service.py` (`search_contacts`, `export_contacts_csv`).

# Consentement marketing (double opt-in)

Mailjet refuse d'approuver un import de liste de contacts à moins que l'expéditeur ne puisse démontrer que chaque destinataire a explicitement consenti à recevoir des e-mails marketing. Pour répondre à cette exigence, OmniCRM stocke le consentement marketing par contact et expose un flux de

double opt-in ("e-mail de confirmation") qui produit une trace de consentement auditable.

L'implémentation se trouve dans `OmniCRM-API/services/marketing_consent_service.py`. Seuls les contacts avec `marketing_consent_status == 'granted'` doivent être utilisés pour les envois marketing ou les imports de listes externes.

## Modèle de données

Deux tables sont impliquées :

- `contact.marketing_consent_status` (`Enum: none | pending | granted | revoked`) et `contact.marketing_consent_granted_at` - l'état actuel dénormalisé utilisé pour un filtrage rapide dans la table des contacts et l'exportation.
- `marketing_consent` - un **journal d'audit en ajout uniquement**. Chaque transition d'état (coche d'inscription, clic de confirmation, basculement d'auto-assistance, révocation par webhook, ré-autorisation par un administrateur) écrit une nouvelle ligne capturant l'ID de contact, l'e-mail au moment de l'événement, la source, la version du texte de consentement, l'adresse IP, l'agent utilisateur, le jeton signé et les horodatages de confirmation / révocation. Le journal est la source de vérité si Mailjet demande un preuve de consentement pour un destinataire donné.

La "version du texte de consentement" est configurée sous `marketing_consent.consent_text_version` dans `crm_config.yaml` et est estampillée sur chaque ligne d'audit, donc changer le libellé du texte de consentement à l'avenir est non destructif : les lignes existantes continuent de pointer vers la version que l'utilisateur a réellement vue.

## Configuration

Ajoutez le bloc `marketing_consent` à `crm_config.yaml` :

```yaml
marketing_consent:
  consent_text_version: "v1-2026-05"
  confirmation_url_path: "/marketing/confirm"
  token_expiry_days: 30
  re_permission_batch_size: 200
  re_permission_rate_per_second: 5
  re_permission_skip_pending_within_hours: 72
```

Configurez également un modèle Mailjet pour l'e-mail de confirmation (sinon, un corps HTML/texte codé en dur est utilisé) :

```yaml
mailjet:
  api_crmCommunicationMarketingConfirm:
    from_email: "support@example.com"
    from_name: "Support Mobile Exemple"
    template_id: 0       # 0 = utiliser le corps HTML intégré
    subject: "Veuillez confirmer votre abonnement"
```

Variables de modèle Mailjet exposées à un modèle personnalisé : `firstname`, `confirmation_link`, `consent_text_version`.

## Flux A : Nouvelles inscriptions

Le formulaire d'auto-inscription affiche une case à cocher "Inscrivez-vous à l'e-News de Exemple Mobile pour les dernières nouvelles, événements et promotions". Lorsqu'elle est cochée :

1. `routes/auth.py /self-signup-partial` extrait `customer.subscribe_to_newsletter` du corps de la requête.
2. Après la création du contact, `marketing_consent_service.request_consent` est appelé avec `source='signup'`, l'IP de la requête et l'agent utilisateur.
3. Une ligne d'audit `marketing_consent` est écrite avec `granted=True`, `confirmed_at=NULL`, et un jeton signé URL-safe. Le `marketing_consent_status` du contact est défini sur `pending`.
4. Un e-mail transactionnel Mailjet est envoyé contenant `${public_url}${confirmation_url_path}?token=...`.

5. Lorsque l'utilisateur clique sur le lien, `GET /crm/marketing-consent/confirm` échange le jeton : `confirmed_at` est estampillé sur la ligne d'audit et `marketing_consent_status` passe à `granted`.

## Flux B : Auto-assistance (utilisateurs connectés)

L'auto-assistance affiche une page "Préférences" (`/self-care/preferences`) qui appelle :

| Méthode | Chemin | Objectif |
|---|---|---|
| `GET` | `/crm/marketing-consent/me` | État actuel + historique d'audit complet |
| `POST` | `/crm/marketing-consent/me/subscribe` | (Re)commencer le flux de double opt-in |
| `POST` | `/crm/marketing-consent/me/unsubscribe` | Révoquer le consentement marketing immédiatement |

La révocation écrit une ligne d'audit `granted=False` et définit le statut sur `revoked`. Le contact doit être retiré de toutes les listes marketing externes maintenues en dehors d'OmniCRM.

## Flux C : Ré-autorisation de base existante (administrateur)

Les contacts hérités créés avant le déploiement de cette fonctionnalité n'ont pas d'enregistrement de consentement et sont donc dans `marketing_consent_status='none'`. Les administrateurs peuvent leur demander de s'inscrire via la campagne de ré-autorisation :

| Méthode | Chemin | Objectif |
|---|---|---|
| `GET` | `/crm/oam/marketing-consent/stats` | Comptes par statut + liste de candidats + échantillons |
| `POST` | `/crm/oam/marketing-consent/re-permission?dry_run=true` | Plan (aucun e-mail envoyé) |
| `POST` | `/crm/oam/marketing-consent/re-permission?dry_run=false` | Exécuter. Rythmé et limité par la configuration. |

Le travail scanne chaque contact dont le statut est `none`, `revoked`, ou `pending` plus vieux que `re_permission_skip_pending_within_hours`, a un e-mail valide, n'est pas désinscrit, et dont le statut client n'est pas `Closed`. Pour chaque candidat, il appelle le même `request_consent` utilisé par le flux d'inscription, donc le format de ligne d'audit et la sémantique du jeton sont identiques, quelle que soit l'origine.

L'envoi est limité par un seau de jetons de taille `re_permission_rate_per_second`, nous maintenant confortablement en dessous du plafond de débit transactionnel de Mailjet. Un paramètre de requête `limit` optionnel limite le nombre d'envois par exécution afin que la première exécution puisse être un petit test de validation.

## Déploiement recommandé pour la première fois

1. Confirmez que la migration de base de données pour `marketing_consent` a été effectuée.
2. Visitez **Marketing par e-mail → Consentement marketing** dans l'interface d'administration (`/email-marketing/marketing-consent`). Examinez la répartition par statut. La plupart des contacts hérités seront `none`.
3. Exécutez **Ré-autorisation en mode test** et inspectez les comptes.

4. **Envoyez des e-mails de confirmation** avec le paramètre de requête `limit` défini sur un petit lot (par exemple, 50) pour valider la délivrabilité et soutenir le volume de la boîte de réception.
5. Exécutez à nouveau sans `limit` pour couvrir le reste de la base. Le travail est idempotent : les contacts avec une demande en attente récente sont ignorés.
6. Après que les opt-ins se stabilisent, utilisez **Clients → Tous les Contacts**, filtrez `marketing_consent_status = granted`, et **Exporter CSV** pour importation dans Mailjet (ou votre ESP) si vous maintenez une liste marketing là-bas.

## Métriques Prometheus

| Métrique | Étiquettes | Rema |
|---|---|---|
| `omnicrm_marketing_consent_events_total` | `event` | `confirmation_se` `confirmation_fa` `revoked`, `token_` `token_invalid`, |
| `omnicrm_marketing_consent_status_total` | `status` | Jauge par `none\|pending\|gr` Rafraîchi par `get` |

Les métriques de l'API transactionnelle Mailjet (`omnicrm_mailjet_api_*`, `omnicrm_mailjet_emails_sent_*`) sont documentées dans Surveillance & Métriques.

# Dépannage

**E-mail non envoyé**

- **Cause :** Identifiants API invalides, compte Mailjet suspendu, ou ID de modèle incorrect
- **Solution :**
  - Vérifiez `api_key` et `api_secret` dans `crm_config.yaml`

- Vérifiez l'état du compte Mailjet et la facturation
- Vérifiez que l'ID du modèle existe dans Mailjet
- Vérifiez les journaux API pour les erreurs

**Variables de modèle non substituées**

- **Cause :** Mismatch de nom de variable ou données manquantes dans OmniCRM
- **Solution :**
  - Vérifiez que les noms de variables correspondent exactement (sensible à la casse)
  - Utilisez le format `{{ var:variable_name }}`
  - Vérifiez qu'OmniCRM passe les données de variables dans l'appel API
  - Testez avec des données d'exemple dans Mailjet

**PDF de facture non attaché**

- **Cause :** Échec de la génération du PDF ou taille de fichier trop grande
- **Solution :**
  - Vérifiez les journaux de génération de factures
  - Vérifiez que le modèle de facture se rend correctement
  - Assurez-vous que le PDF fait moins de 15 Mo (limite Mailjet)
  - Testez la génération du PDF de facture séparément

**Besoin d'une liste de contacts marketing dans Mailjet**

- **Cause :** OmniCRM ne pousse plus automatiquement les contacts vers Mailjet
- **Solution :**
  - Collectez les opt-ins via le flux de consentement marketing (`granted` status)
  - Exportez depuis **Clients → Tous les Contacts** avec filtre de consentement, ou `GET /crm/contact/export?filters={"marketing_consent_status":["granted"]}`
  - Importez le CSV dans Mailjet en utilisant leur interface d'importation de contacts

# Documentation Connexe

- `administration_configuration` - Référence complète de configuration Mailjet
- `payments_invoices` - Génération de factures et livraison d'e-mails
- `authentication_flows` - E-mails de réinitialisation de mot de passe et de vérification
- `customer_care` - E-mails de bienvenue du portail d'auto-assistance
- Surveillance & Métriques - Métriques de l'API Mailjet, suivi de la livraison d'e-mails et surveillance des performances

# Lectures Complémentaires

- Documentation Mailjet : <https://dev.mailjet.com/>
- Référence API Mailjet : <https://dev.mailjet.com/email/reference/>

# API de Notification pour Systèmes Externes

L'API de Notification OmniCRM permet aux systèmes externes et aux intégrations tierces d'envoyer des notifications dans l'application aux utilisateurs. Cela permet d'envoyer des alertes automatisées, des mises à jour de statut et des informations importantes directement aux utilisateurs dans l'interface OmniCRM.

Voir aussi : Utilisation de l'API pour les méthodes d'authentification, Système de Tickets de Support pour les notifications liées aux tickets, Surveillance & Métriques pour la surveillance du système.

## Aperçu

L'API de Notification fournit un moyen flexible pour tout système d'envoyer des notifications aux utilisateurs d'OmniCRM. Les notifications apparaissent dans le menu déroulant de notifications de l'utilisateur dans l'interface OmniCRM et peuvent éventuellement lier à des pages ou ressources spécifiques.

### Fonctionnalités Clés

1. **Intégration Flexible** : Tout système peut envoyer des notifications via l'API
2. **Multiples Types de Notifications** : Support pour divers types d'événements (provisionnement, facturation, tickets, personnalisé)
3. **Métadonnées Riches** : Inclure des données structurées avec les notifications
4. **Navigation par Lien** : Les notifications peuvent lier à des pages spécifiques
5. **Authentification par Clé API** : Intégration sécurisée de serveur à serveur

6. **Livraison en Temps Réel** : Les notifications apparaissent immédiatement dans l'interface utilisateur
7. **Stockage Persistant** : Le système stocke les notifications et les rend récupérables

## Cas d'Utilisation Courants

**Systèmes de Provisionnement :**

- Notifier les utilisateurs lorsque l'activation du service est terminée
- Alerter sur les échecs de provisionnement
- Mettre à jour les utilisateurs sur les tâches de provisionnement de longue durée

**Systèmes de Facturation et de Paiement :**

- Alerter les utilisateurs lorsque de nouvelles factures sont générées
- Confirmer le traitement des paiements
- Avertir des expirations de service à venir

**Surveillance et Alerte :**

- Envoyer des alertes de santé du système aux administrateurs
- Notifier sur les violations de seuil
- Alerter sur des événements critiques

**Intégrations Personnalisées :**

- Systèmes de flux de travail externes
- Intégrations de services tiers
- Systèmes de reporting automatisés

# Authentification

L'API de Notification prend en charge deux méthodes d'authentification :

## Authentification par Clé API (Recommandée pour les Systèmes Externes)

Les clés API fournissent une authentification sécurisée et de longue durée pour les intégrations de serveur à serveur. C'est la méthode recommandée pour les systèmes externes.

**Configurer la Clé API :**

Ajoutez votre clé API à `crm_config.yaml` :

```yaml
api_keys:
  your-secure-api-key-minimum-32-characters:
    roles:
      - admin
    description: "Système de notification de provisionnement"
```

**Utiliser la Clé API :**

Incluez la clé API dans l'en-tête `X-API-KEY` :

```
curl -X POST "https://yourcrm.com/crm/notification/send" \
  -H "X-API-KEY: your-secure-api-key-minimum-32-characters" \
  -H "Content-Type: application/json" \
  -d '{
    "user_id": 123,
    "type": "provision_complete",
    "title": "Service Activé",
    "message": "Votre service mobile a été activé avec succès",
    "link": "/services/167"
  }'
```

## Authentification par Token Bearer

Pour les systèmes interactifs ou les notifications initiées par l'utilisateur, utilisez l'authentification par token Bearer.

Voir Utilisation de l'API pour des détails sur l'obtention de tokens Bearer.

# Point de Terminaison de l'API

## Envoyer une Notification

**Point de Terminaison :** `POST /crm/notification/send`

**Authentification :** Clé API (X-API-KEY) ou Token Bearer

**Description :** Envoyer une notification à un utilisateur spécifique

### Corps de la Requête

```
{
  "user_id": 123,
  "type": "provision_complete",
  "title": "Service Activé",
  "message": "Votre service mobile a été activé avec succès",
  "link": "/services/167",
  "metadata": {
    "service_id": 167,
    "provision_job_id": "prov-2026-00042",
    "duration_seconds": 45
  },
  "related_ticket_id": 42,
  "related_user_id": 5
}
```

### Paramètres de la Requête

| Champ | Type | Requis | Description |
|---|---|---|---|
| `user_id` | entier | **Oui** | ID de l'utilisateur à notifier |
| `type` | chaîne | **Oui** | Type de notification (voir types ci-dessous) |
| `title` | chaîne | **Oui** | Titre de la notification (max 255 caractères) |
| `message` | chaîne | Non | Message de notification détaillé |
| `link` | chaîne | Non | URL/chemin à naviguer lors du clic |
| `metadata` | objet | Non | Données structurées supplémentaires (JSON) |
| `related_ticket_id` | entier | Non | ID du ticket associé |
| `related_user_id` | entier | Non | ID de l'utilisateur lié (par exemple, celui qui a causé l'événement) |

**Types de Notifications**

| Type | Description | Cas d'Utilisation Commun |
|---|---|---|
| `provision_complete` | Provisionnement terminé avec succès | Playbook Ansible terminé |
| `provision_failed` | Échec de provisionnement | Erreur d'activation de service |
| `ticket_mention` | Utilisateur @mentionné dans un ticket | Alerte de collaboration |
| `ticket_assigned` | Ticket assigné à l'utilisateur | Flux de travail de support |
| `ticket_updated` | Ticket mis à jour | Notification d'abonné au ticket |
| `customer_reply` | Le client a répondu au ticket | Alerte de support |
| `system_alert` | Alerte à l'échelle du système | Événement système critique |
| `billing_invoice` | Nouvelle facture générée | Intégration du système de facturation |
| `billing_payment` | Paiement traité | Webhook de passerelle de paiement |
| `service_expiring` | Service expirant bientôt | Surveillance automatisée |
| `custom` | Type de notification personnalisé | Tout autre but |

**Réponse de Succès**

**Statut :** `201 Created`


```
{
  "result": "Success",
  "message": "Notification envoyée avec succès",
  "notification": {
    "notification_id": 1234,
    "user_id": 123,
    "type": "provision_complete",
    "title": "Service Activé",
    "message": "Votre service mobile a été activé avec succès",
    "link": "/services/167",
    "metadata": {
      "service_id": 167,
      "provision_job_id": "prov-2026-00042",
      "duration_seconds": 45
    },
    "is_read": false,
    "created_at": "2026-01-11T14:30:00Z"
  }
}
```


**Réponses d'Erreur**

**Champ Requis Manquant (400 Bad Request) :**


```
{
  "result": "Failed",
  "reason": "user_id est requis"
}
```


**Utilisateur Non Trouvé (404 Not Found) :**


```
{
  "result": "Failed",
  "reason": "Utilisateur 999 non trouvé"
}
```


**Échec de l'Authentification (401 Unauthorized) :**

```
{
  "message": "Clé API invalide"
}
```

# Exemples de Code

## Exemple Python

```python
import requests

# Configuration
CRM_URL = 'https://yourcrm.com'
API_KEY = 'your-secure-api-key-minimum-32-characters'

def send_notification(user_id, notification_type, title,
message=None, link=None, metadata=None):
    """Envoyer une notification à un utilisateur via l'API
OmniCRM"""

    headers = {
        "Content-Type": "application/json",
        "X-API-KEY": API_KEY
    }

    payload = {
        "user_id": user_id,
        "type": notification_type,
        "title": title
    }

    # Ajouter des champs optionnels
    if message:
        payload["message"] = message
    if link:
        payload["link"] = link
    if metadata:
        payload["metadata"] = metadata

    response = requests.post(
        f"{CRM_URL}/crm/notification/send",
        headers=headers,
        json=payload
    )

    if response.status_code == 201:
        print(f"✓ Notification envoyée avec succès :
```

```
{response.json()}")
        return response.json()
    else:
        print(f"✗ Échec de l'envoi de la notification :
{response.text}")
        response.raise_for_status()

# Exemple d'utilisation
if __name__ == "__main__":
    # Notifier l'utilisateur de l'achèvement du provisionnement
    send_notification(
        user_id=123,
        notification_type="provision_complete",
        title="Service Mobile Activé",
        message="L'ID de service 167 a été provisionné et activé
avec succès",
        link="/services/167",
        metadata={
            "service_id": 167,
            "provision_job_id": "prov-2026-00042",
            "duration_seconds": 45,
            "playbook": "activation_service_mobile"
        }
    )
```

## Exemple Bash/cURL

```
#!/bin/bash

# Configuration
CRM_URL="https://yourcrm.com"
API_KEY="your-secure-api-key-minimum-32-characters"

# Envoyer une notification
curl -X POST "${CRM_URL}/crm/notification/send" \
  -H "X-API-KEY: ${API_KEY}" \
  -H "Content-Type: application/json" \
  -d '{
    "user_id": 123,
    "type": "provision_complete",
    "title": "Service Activé",
    "message": "Votre service mobile a été activé avec succès",
    "link": "/services/167",
    "metadata": {
      "service_id": 167,
      "provision_job_id": "prov-2026-00042",
      "duration_seconds": 45
    }
  }'
```

## Exemple Node.js

```
const axios = require('axios');

const CRM_URL = 'https://yourcrm.com';
const API_KEY = 'your-secure-api-key-minimum-32-characters';

async function sendNotification(userId, type, title, options = {})
{
    try {
        const response = await axios.post(
            `${CRM_URL}/crm/notification/send`,
            {
                user_id: userId,
                type: type,
                title: title,
                message: options.message,
                link: options.link,
                metadata: options.metadata,
                related_ticket_id: options.relatedTicketId,
                related_user_id: options.relatedUserId
            },
            {
                headers: {
                    'X-API-KEY': API_KEY,
                    'Content-Type': 'application/json'
                }
            }
        );

        console.log('✓ Notification envoyée :', response.data);
        return response.data;
    } catch (error) {
        console.error('✗ Échec de l'envoi de la notification :',
error.response?.data || error.message);
        throw error;
    }
}

// Exemple d'utilisation
sendNotification(
    123,
    'provision_complete',
```

```
    'Service Activé',
    {
        message: 'Votre service mobile a été activé avec succès',
        link: '/services/167',
        metadata: {
            service_id: 167,
            provision_job_id: 'prov-2026-00042',
            duration_seconds: 45
        }
    }
);
```

# Scénarios d'Intégration

## Playbook de Provisionnement Ansible

Notifier les utilisateurs lorsque les playbooks Ansible se terminent :

```python
# À la fin de votre plugin de rappel Ansible ou script post-playbook
import requests

def notify_provisioning_complete(service_id, user_id, success=True):
    """Notifier l'utilisateur du résultat de provisionnement"""

    notification_type = "provision_complete" if success else "provision_failed"
    title = "Service Activé" if success else "Échec de l'Activation du Service"
    message = f"Le service {service_id} a été {'provisionné avec succès' if success else 'rencontré une erreur'}"

    requests.post(
        "https://yourcrm.com/crm/notification/send",
        headers={"X-API-KEY": "your-api-key"},
        json={
            "user_id": user_id,
            "type": notification_type,
            "title": title,
            "message": message,
            "link": f"/services/{service_id}",
            "metadata": {
                "service_id": service_id,
                "playbook": "activation_service_mobile",
                "success": success
            }
        }
    )
```

## Intégration du Système de Facturation

Envoyer des notifications de factures depuis le système de facturation :

```
def notify_invoice_generated(customer_id, invoice_id, amount,
due_date):
    """Notifier le contact client de la nouvelle facture"""

    # Trouver l'utilisateur associé au client
    user_id = get_customer_primary_contact_user_id(customer_id)

    if user_id:
        requests.post(
            "https://yourcrm.com/crm/notification/send",
            headers={"X-API-KEY": "your-billing-api-key"},
            json={
                "user_id": user_id,
                "type": "billing_invoice",
                "title": f"Nouvelle Facture #{invoice_id}",
                "message": f"La facture de ${amount} est
maintenant disponible. Date d'échéance : {due_date}",
                "link": f"/invoices/{invoice_id}",
                "metadata": {
                    "invoice_id": invoice_id,
                    "amount": amount,
                    "due_date": due_date,
                    "customer_id": customer_id
                }
            }
        )
```

## Intégration du Système de Surveillance

Envoyer des alertes depuis les systèmes de surveillance :

```python
def send_monitoring_alert(admin_user_ids, alert_title,
alert_message, severity="warning"):
    """Envoyer une alerte système aux administrateurs"""

    for user_id in admin_user_ids:
        requests.post(
            "https://yourcrm.com/crm/notification/send",
            headers={"X-API-KEY": "your-monitoring-api-key"},
            json={
                "user_id": user_id,
                "type": "system_alert",
                "title": alert_title,
                "message": alert_message,
                "metadata": {
                    "severity": severity,
                    "alert_time": datetime.now().isoformat(),
                    "source": "prometheus_alertmanager"
                }
            }
        )

# Exemple : Alerte d'espace disque
send_monitoring_alert(
    admin_user_ids=[1, 2, 3],
    alert_title="⚠ Alerte d'Espace Disque",
    alert_message="L'utilisation du disque du serveur de base de
données a dépassé 85%",
    severity="warning"
)
```

## Webhook de Passerelle de Paiement

Notifier les utilisateurs lorsque des paiements sont traités :

```
from flask import Flask, request

app = Flask(__name__)

@app.route('/webhook/payment', methods=['POST'])
def payment_webhook():
    """Gérer le webhook de la passerelle de paiement"""

    payment_data = request.json

    if payment_data['status'] == 'succeeded':
        # Notifier l'utilisateur du paiement réussi
        requests.post(
            "https://yourcrm.com/crm/notification/send",
            headers={"X-API-KEY": "your-payment-webhook-api-key"},
            json={
                "user_id": payment_data['user_id'],
                "type": "billing_payment",
                "title": "Paiement Reçu",
                "message": f"Le paiement de
${payment_data['amount']} a été traité avec succès",
                "link": f"/payments/{payment_data['payment_id']}",
                "metadata": {
                    "payment_id": payment_data['payment_id'],
                    "amount": payment_data['amount'],
                    "payment_method":
payment_data['payment_method']
                }
            }
        )

    return {"status": "ok"}
```

## Moniteur d'Expiration de Service

Script automatisé pour notifier les utilisateurs des services expirants :

```
import datetime
from sqlalchemy import create_engine, text

def check_expiring_services():
    """Vérifier les services expirant dans les 7 prochains jours
et notifier les utilisateurs"""

    engine = create_engine("postgresql://...")

    # Interroger les services expirant bientôt
    query = text("""
        SELECT s.service_id, s.end_date, u.id as user_id,
s.service_name
        FROM services s
        JOIN customer_contacts cc ON s.customer_id =
cc.customer_id
        JOIN users u ON cc.contact_id = u.contact_id
        WHERE s.end_date BETWEEN CURRENT_DATE AND CURRENT_DATE +
INTERVAL '7 days'
        AND s.deleted = false
    """)

    with engine.connect() as conn:
        results = conn.execute(query)

        for row in results:
            days_until_expiry = (row.end_date -
datetime.date.today()).days

            requests.post(
                "https://yourcrm.com/crm/notification/send",
                headers={"X-API-KEY": "your-automation-api-key"},
                json={
                    "user_id": row.user_id,
                    "type": "service_expiring",
                    "title": f"Service Expirant dans
{days_until_expiry} Jours",
                    "message": f"Votre {row.service_name} expirera
le {row.end_date}",
                    "link": f"/services/{row.service_id}",
                    "metadata": {
                        "service_id": row.service_id,
                        "days_remaining": days_until_expiry,
```

```
                        "expiry_date": str(row.end_date)
                    }
                }
            )

# Exécuter quotidiennement via cron
if __name__ == "__main__":
    check_expiring_services()
```

# Meilleures Pratiques

## Sécurité de la Clé API

1. **Générer des Clés Fortes** : Utilisez des générateurs aléatoires cryptographiquement sécurisés :

   ```
   openssl rand -base64 48
   ```

2. **Clés Séparées par Système** : Utilisez des clés API différentes pour différents systèmes externes

3. **Documenter l'Usage des Clés** : Inclure toujours une description dans `crm_config.yaml`

4. **Faire Périodiquement des Rotations de Clés** : Changer les clés API périodiquement (par exemple, trimestriellement)

5. **Ne Jamais Commiter les Clés** : Gardez les clés API hors du contrôle de version

6. **Permissions Minimales** : Accorder uniquement les rôles nécessaires à chaque clé API

## Conception de Notifications

1. **Titres Clairs** : Rendre les titres descriptifs et actionnables (par exemple, "Service Activé" plutôt que "Mise à Jour")

2. **Messages Concis** : Garder les messages brefs mais informatifs

3. **Inclure des Liens** : Toujours fournir un lien pour que les utilisateurs puissent obtenir plus de détails

4. **Utiliser des Métadonnées** : Stocker des données structurées pour référence future et débogage

5. **Choisir le Type Correct** : Utiliser des types de notification appropriés pour la catégorisation

6. **Tester les Notifications** : Tester avec de vrais utilisateurs avant de déployer en production

## Gestion des Erreurs

1. **Gérer les Échecs avec Grâce** : Ne pas laisser les échecs de notification casser les flux de travail principaux

2. **Journaliser Tous les Essais** : Journaliser les tentatives de notification pour le débogage

3. **Réessayer en Cas d'Erreurs Transitoires** : Implémenter une logique de réessai pour les problèmes de réseau

4. **Valider les IDs d'Utilisateur** : Vérifier que l'utilisateur existe avant d'envoyer

5. **Vérifier les Réponses de l'API** : Toujours vérifier les codes de statut de la réponse

## Performance

1. **Batcher Quand Possible** : Pour plusieurs utilisateurs, envisager d'envoyer en parallèle

2. **Traitement Asynchrone** : Envoyer des notifications de manière asynchrone pour éviter de bloquer

3. **Limitation de Taux** : Implémenter une limitation de taux pour prévenir les abus de l'API

4. **Pool de Connexion** : Réutiliser les connexions HTTP pour de meilleures performances

# Tests

## Tester la Configuration de la Clé API

Avant de déployer, vérifiez que votre clé API fonctionne :

```
# Tester l'authentification
curl -X POST "https://yourcrm.com/crm/notification/send" \
  -H "X-API-KEY: your-api-key" \
  -H "Content-Type: application/json" \
  -d '{
    "user_id": 1,
    "type": "custom",
    "title": "Notification de Test",
    "message": "Ceci est une notification de test pour vérifier
l'authentification de la clé API"
  }'
```

Réponse attendue : `201 Created` avec les détails de la notification

## Tester en Développement

Utilisez un ID d'utilisateur de test pour vérifier que les notifications apparaissent :

```
# Envoyer une notification de test
response = send_notification(
    user_id=1,  # Votre utilisateur de test
    notification_type="custom",
    title="Test d'Intégration",
    message="Test de l'intégration de l'API de notification"
)

print(f"ID de Notification : {response['notification']
['notification_id']}")
```

Ensuite, connectez-vous à OmniCRM en tant que cet utilisateur et vérifiez que la notification apparaît dans le menu déroulant des notifications.

# Dépannage

**Problème : 401 Unauthorized**

- **Cause :** Clé API invalide ou manquante
- **Solution :** Vérifiez la clé API dans `crm_config.yaml` et l'en-tête de requête
- **Vérifiez :** Assurez-vous que la clé API a au moins 32 caractères

**Problème : 404 Utilisateur Non Trouvé**

- **Cause :** `user_id` invalide
- **Solution :** Vérifiez que l'utilisateur existe dans la base de données
- **Vérifiez :** Interrogez la table des utilisateurs : `SELECT id FROM users WHERE id = ?`

**Problème : 400 Bad Request - "user_id est requis"**

- **Cause :** Champ requis manquant dans la requête
- **Solution :** Inclure tous les champs requis : user_id, type, title
- **Vérifiez :** Validez la structure de la charge utile JSON

**Problème : Notification envoyée mais n'apparaît pas**

- **Cause :** Utilisateur non connecté ou cache du navigateur

- **Solution :** Rafraîchir la page ou vider le cache
- **Vérifiez :** Interrogez la base de données : `SELECT * FROM notifications WHERE user_id = ? ORDER BY created_at DESC`

**Problème : 500 Internal Server Error**

- **Cause :** Erreur côté serveur
- **Solution :** Vérifiez les journaux du serveur CRM pour plus de détails
- **Vérifiez :** Consultez `/var/log/omnicrm/api.log` pour les erreurs

# Résumé de la Référence API

## Points de Terminaison

| Méthode | Point de Terminaison | Description | Auth |
|---|---|---|---|
| POST | `/crm/notification/send` | Envoyer une notification à un utilisateur | Clé API ou Bearer |
| GET | `/crm/notification/` | Obtenir les notifications de l'utilisateur actuel | Bearer |
| GET | `/crm/notification/unread-count` | Obtenir le compte des non lus | Bearer |
| POST | `/crm/notification/{id}/mark-read` | Marquer la notification comme lue | Bearer |
| POST | `/crm/notification/mark-all-read` | Marquer toutes comme lues | Bearer |
| DELETE | `/crm/notification/{id}` | Supprimer la notification | Bearer |

## Permissions Requises

- **Envoyer Notification :** Nécessite une clé API valide ou un rôle admin
- **Voir Notifications :** L'utilisateur ne peut voir que ses propres notifications
- **Gérer Notifications :** L'utilisateur ne peut gérer que ses propres notifications

# Documentation Connexe

- Utilisation de l'API : Guide complet d'authentification de l'API
- Système de Tickets de Support : Notifications de tickets et @mentions
- Système de Provisionnement : Intégration Ansible
- Surveillance & Métriques : Surveillance et alertes du système
- RBAC : Rôles et permissions

# Journal des modifications

Cela contient les 50 dernières modifications apportées à la pile logicielle OmniCRM ou à ses dépendances.

Remarque : Cela ne suit pas les modifications apportées aux configurations individuelles des clients.

# Surveillance & Métriques

L'API OmniCRM fournit des métriques complètes basées sur Prometheus pour surveiller la performance des applications, les opérations commerciales et les intégrations externes. Toutes les métriques sont exposées à l'endpoint `/crm/metrics` au format d'exposition Prometheus.

## Vue d'ensemble

Le système de métriques suit :

- **Opérations de Provisionnement** - Exécution des tâches, durée, taux de succès/échec
- **Performance de la Base de Données** - Temps de requête, santé du pool de connexions
- **Intégrations Externes** - Appels API OCS/CGRateS, Stripe, Mailjet
- **Tâches en Arrière-plan** - Exécution et performance des tâches asynchrones
- **Requêtes HTTP** - Utilisation des endpoints API et temps de réponse (généré automatiquement)

## Endpoint de Métriques

**URL :** `http://your-omnicrm-api:5000/crm/metrics`

**Format :** Format d'exposition Prometheus

**Authentification :** Prometheus peut extraire les métriques de l'endpoint, qui est accessible publiquement. En production, restreignez l'accès avec des règles de pare-feu ou une authentification par proxy inverse.

# Catégories de Métriques

## 1. Métriques de Provisionnement

Les métriques de provisionnement suivent l'exécution des playbooks Ansible qui provisionnent des services, gèrent l'inventaire et configurent des systèmes externes. Voir Système de Provisionnement et Playbooks Ansible pour plus de détails.

**omnicrm_provision_jobs_total**

**Type :** Compteur

**Étiquettes :**

- `status` - Statut d'achèvement de la tâche : `success`, `failed`, `running`

**Description :** Nombre total de tâches de provisionnement créées. Incrémenté lorsque les tâches de provisionnement se terminent avec leur statut final.

**omnicrm_provision_job_duration_seconds**

**Type :** Histogramme

**Étiquettes :**

- `playbook` - Nom du playbook Ansible exécuté
- `status` - Statut d'achèvement de la tâche : `success`, `failed`

**Seaux :** `[1, 5, 10, 30, 60, 120, 180, 300, 600]` secondes (1s à 10min)

**Description :** Temps pris pour que les tâches de provisionnement se terminent. Enregistre la durée de l'exécution complète du playbook du début à la fin.

**omnicrm_provision_jobs_active**

**Type :** Jauge

**Description :** Nombre de t��ches de provisionnement actuellement en cours d'exécution. Incrémenté lorsqu'une tâche commence, décrémenté lorsqu'elle se termine.

**omnicrm_provision_tasks_total**

**Type :** Compteur

**Étiquettes :**

- `playbook` - Nom du playbook Ansible
- `status` - Résultat de la tâche : `ok`, `failed`, `ignored`

**Description :** Nombre total de tâches Ansible exécutées dans les playbooks. Incrémenté pour chaque tâche individuelle qui se termine (succès ou échec).

**omnicrm_provision_errors_total**

**Type :** Compteur

**Étiquettes :**

- `error_type` - Type d'erreur : `fatal`, `task_failed`, `timeout`
- `playbook` - Nom du playbook Ansible

**Description :** Nombre total d'erreurs de provisionnement par type. Incrémenté lorsque les tâches de provisionnement échouent ou que des erreurs fatales se produisent pendant l'exécution.

---

# 2. Métriques de Base de Données

Les métriques de base de données surveillent la performance des requêtes et la santé du pool de connexions. OmniCRM utilise SQLAlchemy avec instrumentation automatique. Voir Architecture du Système pour les détails du modèle de données.

**omnicrm_db_query_duration_seconds**

**Type :** Histogramme

**Étiquettes :**

- `operation` - Type d'opération SQL : `SELECT`, `INSERT`, `UPDATE`, `DELETE`
- `table` - Nom de la table de base de données

**Seaux :** `[0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1.0, 5.0, 10.0]` secondes

**Description :** Temps d'exécution des requêtes de base de données. Suivi automatiquement via des écouteurs d'événements SQLAlchemy sur chaque requête.

**omnicrm_db_pool_size**

**Type :** Jauge

**Description :** Taille actuelle du pool de connexions de la base de données. Le nombre total de connexions dans le pool (à la fois vérifiées et disponibles).

**omnicrm_db_pool_overflow**

**Type :** Jauge

**Description :** Débordement actuel du pool de connexions de la base de données. Le nombre de connexions créées au-delà de la limite normale de taille du pool.

**omnicrm_db_pool_connections_checked_out**

**Type :** Jauge

**Description :** Nombre de connexions actuellement vérifiées et utilisées par le code de l'application.

**omnicrm_db_errors_total**

**Type :** Compteur

**Étiquettes :**

- `error_type` - Type d'erreur de base de données

- `operation` - Opération qui a causé l'erreur : `connection_error`, `query_error`, etc.

**Description :** Nombre total d'erreurs de base de données.

**Statut :** Défini mais pas activement utilisé dans le code actuel (réservé pour une utilisation future)

---

# 3. Métriques OCS/CGRateS

Les métriques OCS (Système de Facturation en Ligne) suivent les interactions avec le moteur de facturation CGRateS. Voir Vue d'ensemble de la Facturation et Intégration CGRateS pour plus de détails.

**omnicrm_ocs_api_calls_total**

**Type :** Compteur

**Étiquettes :**

- `method` - Méthode API OCS : `GetBalance`, `SetBalance`, `SetAccount`, `RemoveAccount`, etc.
- `status` - Résultat de l'appel : `success`, `failed`

**Description :** Nombre total d'appels API OCS/CGRateS. Incrémenté pour chaque appel API asynchrone au système OCS.

**omnicrm_ocs_api_duration_seconds**

**Type :** Histogramme

**Étiquettes :**

- `method` - Méthode API OCS : `GetBalance`, `SetBalance`, `GetAccount`, etc.

**Seaux :** `[0.1, 0.25, 0.5, 1.0, 2.0, 5.0, 10.0, 30.0]` secondes

**Description :** Durée des appels API OCS/CGRateS. Enregistre le temps pris pour chaque appel API, y compris la latence réseau.

**omnicrm_ocs_api_errors_total**

**Type :** Compteur

**Étiquettes :**

- `method` - Méthode API OCS qui a échoué
- `error_type` - Catégorie d'erreur : `timeout`, `connection_error`, `json_error`, `http_error`, etc.

**Description :** Nombre total d'erreurs API OCS/CGRateS. Incrémenté lorsque les appels API OCS échouent avec des types d'erreurs spécifiques.

**omnicrm_ocs_balance_queries_total**

**Type :** Compteur

**Étiquettes :**

- `query_type` - Type de requête de solde : `single_account`, `multiple_accounts`

**Description :** Nombre total de requêtes de solde vers OCS. Utilisé pour suivre l'utilisation des opérations de demande de solde.

**omnicrm_ocs_action_plan_operations_total**

**Type :** Compteur

**Étiquettes :**

- `operation` - Opération de plan d'action : `create`, `remove`, `query`

**Description :** Nombre total d'opérations de plan d'action. Suit la création, la suppression et la requête des plans d'action CGRateS pour les frais récurrents.

---

# 4. Métriques de Paiement Stripe

Les métriques Stripe suivent les opérations de traitement des paiements. Voir Guide du Système de Paiement et Méthodes de Paiement pour les détails

d'intégration.

**omnicrm_stripe_api_calls_total**

**Type :** Compteur

**Étiquettes :**

- `operation` - Opération Stripe : `create_customer`, `charge`, `refund`, `update_payment_method`, etc.
- `status` - Résultat de l'opération : `success`, `failed`

**Description :** Nombre total d'appels API Stripe. Incrémenté pour chaque opération de traitement de paiement.

**omnicrm_stripe_api_duration_seconds**

**Type :** Histogramme

**Étiquettes :**

- `operation` - Type d'opération Stripe

**Seaux :** `[0.1, 0.5, 1.0, 2.0, 5.0, 10.0, 30.0]` secondes

**Description :** Durée des appels API Stripe, y compris la latence réseau.

**omnicrm_stripe_api_errors_total**

**Type :** Compteur

**Étiquettes :**

- `operation` - Opération Stripe qui a échoué
- `error_type` - Catégorie d'erreur : `card_declined`, `network_error`, `api_error`, etc.

**Description :** Nombre total d'erreurs API Stripe. Incrémenté lorsque les opérations de paiement échouent.

**omnicrm_stripe_payment_amount_cents**

**Type :** Compteur

**Étiquettes :**

- `payment_type` - Direction du paiement : `charge`, `refund`

**Description :** Montant total des paiements traités via Stripe en cents. Utile pour suivre le volume des transactions et les revenus.

**omnicrm_stripe_payment_failures_total**

**Type :** Compteur

**Étiquettes :**

- `reason` - Raison de l'échec : `card_declined`, `insufficient_funds`, `expired_card`, etc.

**Description :** Nombre total d'échecs de paiement Stripe classés par code de refus.

---

# 5. Métriques d'Email Mailjet

Les métriques Mailjet suivent les opérations de livraison d'emails. Voir Intégration Mailjet pour les détails de configuration.

**omnicrm_mailjet_api_calls_total**

**Type :** Compteur

**Étiquettes :**

- `email_type` - Type de modèle d'email : `welcome`, `user_welcome`, `invoice`, `notification`
- `status` - Résultat de la livraison : `success`, `failed`

**Description :** Nombre total d'appels API Mailjet. Suivi via le décorateur `@track_mailjet_call`.

**omnicrm_mailjet_api_duration_seconds**

**Type :** Histogramme

**Étiquettes :**

- `email_type` - Type de modèle d'email

**Seaux :** `[0.1, 0.5, 1.0, 2.0, 5.0, 10.0, 30.0]` secondes

**Description :** Durée des appels API Mailjet. Temps pris pour soumettre un email via l'API Mailjet (pas le temps de livraison).

**omnicrm_mailjet_api_errors_total**

**Type :** Compteur

**Étiquettes :**

- `email_type` - Type de modèle d'email
- `error_type` - Catégorie d'erreur

**Description :** Nombre total d'erreurs API Mailjet. Incrémenté lorsque l'envoi d'emails échoue.

**omnicrm_mailjet_emails_sent_total**

**Type :** Compteur

**Étiquettes :**

- `email_type` - Type de modèle d'email
- `template_id` - ID du modèle Mailjet

**Description :** Nombre total d'emails envoyés avec succès via Mailjet. Distinct des appels API car un appel peut envoyer à plusieurs destinataires.

**omnicrm_mailjet_email_recipients_total**

**Type :** Compteur

**Étiquettes :**

- `email_type` - Type de modèle d'email

**Description :** Nombre total de destinataires d'emails à travers tous les emails envoyés.

---

## 6. Métriques de Tâches en Arrière-plan

Les métriques de tâches en arrière-plan suivent les opérations asynchrones comme les chaînes de playbooks et les tâches planifiées. Voir Système de Provisionnement pour les détails des tâches en arrière-plan.

**omnicrm_background_jobs_total**

**Type :** Compteur

**Étiquettes :**

- `job_type` - Catégorie de tâche : `playbook_single`, `playbook_chain`, `periodic_task`

**Description :** Nombre total de tâches en arrière-plan démarrées. Suivi via le gestionnaire de contexte `BackgroundJobTimer`.

**omnicrm_background_jobs_active**

**Type :** Jauge

**Étiquettes :**

- `job_type` - Catégorie de tâche

**Description :** Nombre de tâches en arrière-plan actuellement en cours d'exécution. Incrémenté au démarrage de la tâche, décrémenté à l'achèvement de la tâche.

**omnicrm_background_job_duration_seconds**

**Type :** Histogramme

**Étiquettes :**

- `job_type` - Catégorie de tâche
- `status` - Résultat de la tâche : `success`, `failed`

**Seaux :** `[1, 5, 10, 30, 60, 120, 180, 300, 600, 1800, 3600]` secondes (1s à 1h)

**Description :** Temps d'exécution des tâches en arrière-plan. Inclut la durée complète des opérations multi-étapes.

**omnicrm_background_job_errors_total**

**Type :** Compteur

**Étiquettes :**

- `job_type` - Catégorie de tâche
- `error_type` - Catégorie d'erreur

**Description :** Nombre total d'erreurs de tâches en arrière-plan. Incrémenté lorsque les tâches en arrière-plan échouent avec des exceptions.

---

# 7. Métriques HTTP Flask (Générées automatiquement)

Ces métriques sont générées automatiquement par la bibliothèque `prometheus-flask-exporter` et suivent toutes les requêtes HTTP à l'API.

**flask_http_request_duration_seconds**

**Type :** Histogramme

**Étiquettes :**

- `method` - Méthode HTTP : `GET`, `POST`, `PUT`, `DELETE`, etc.
- `endpoint` - Nom de la route Flask
- `status` - Code de statut HTTP

**Description :** Durée des requêtes HTTP pour tous les endpoints API. Instrumenté automatiquement.

**flask_http_request_total**

**Type :** Compteur

**Étiquettes :**

- `method` - Méthode HTTP
- `endpoint` - Nom de la route Flask
- `status` - Code de statut HTTP

**Description :** Nombre total de requêtes HTTP par endpoint, méthode et code de statut.

**flask_http_request_exceptions_total**

**Type :** Compteur

**Étiquettes :**

- `method` - Méthode HTTP
- `endpoint` - Nom de la route Flask

**Description :** Nombre total d'exceptions non gérées dans les requêtes HTTP. Indique des bogues ou des erreurs inattendues.

---

# 8. Métriques d'Erreur API (Réservées)

Ces métriques sont définies mais pas actuellement instrumentées. Elles sont réservées pour une utilisation future.

**omnicrm_api_errors_total**

**Type :** Compteur

**Étiquettes :**

- `endpoint` - Endpoint API
- `error_type` - Catégorie d'erreur
- `status_code` - Code de statut HTTP

**Statut :** Défini mais pas activement utilisé

**omnicrm_api_auth_failures_total**

**Type :** Compteur

**Étiquettes :**

- `auth_method` - Méthode d'authentification : `jwt`, `api_key`, `ip_whitelist`
- `reason` - Raison de l'échec

**Statut :** Défini mais pas activement utilisé

---

# 9. Métrique d'Info de l'Application

**omnicrm_api**

**Type :** Info

**Étiquettes :**

- `version` - Chaîne de version de l'API
- `environment` - Nom de l'environnement : `production`, `staging`, `development`

**Description :** Métrique d'information de l'API OmniCRM. Définie une fois au démarrage de l'application avec des informations sur la version et l'environnement.

---

# Mises à Jour Périodiques

**update_db_pool_metrics(engine)**

Appelée automatiquement toutes les 30 secondes pour mettre à jour les jauges du pool de connexions de la base de données.

---

# Configuration de Prometheus

## Configuration de Scraping

Ajoutez OmniCRM à votre `prometheus.yml` :

```yaml
scrape_configs:
  - job_name: 'omnicrm-api'
    scrape_interval: 15s
    scrape_timeout: 10s
    static_configs:
      - targets: ['omnicrm-api:5000']
    metrics_path: '/crm/metrics'
```

## Exemples d'Alerte

### Taux d'Échec de Provisionnement Élevé

```yaml
- alert: HighProvisionFailureRate
  expr: |
    (
      rate(omnicrm_provision_jobs_total{status="failed"}[5m]) /
      rate(omnicrm_provision_jobs_total[5m])
    ) > 0.1
  for: 5m
  labels:
    severity: warning
  annotations:
    summary: "Taux d'échec des tâches de provisionnement élevé"
    description: "{{ $value | humanizePercentage }} des tâches de
provisionnement échouent"
```

### Pool de Connexion de Base de Données Épuisé

```yaml
- alert: DatabasePoolExhausted
  expr: omnicrm_db_pool_overflow > 0
  for: 2m
  labels:
    severity: critical
  annotations:
    summary: "Débordement du pool de connexions de la base de données détecté"
    description: "Le pool de connexions utilise des connexions de débordement, ce qui peut indiquer que la taille du pool est trop petite"
```

**Requêtes de Base de Données Lentes**

```yaml
- alert: SlowDatabaseQueries
  expr: |
    histogram_quantile(0.99,
      rate(omnicrm_db_query_duration_seconds_bucket[5m])
    ) > 1.0
  for: 5m
  labels:
    severity: warning
  annotations:
    summary: "Requêtes de base de données lentes détectées"
    description: "Le temps de requête au 99e percentile est {{ $value }}s"
```

**API OCS Hors Service**

```
- alert: OCSAPIDown
  expr: |
    (
      rate(omnicrm_ocs_api_calls_total{status="failed"}[5m]) /
      rate(omnicrm_ocs_api_calls_total[5m])
    ) > 0.5
  for: 2m
  labels:
    severity: critical
  annotations:
    summary: "Taux d'échec de l'API OCS critique"
    description: "{{ $value | humanizePercentage }} des appels API
OCS échouent"
```

**Problèmes de Paiement Stripe**

```
- alert: StripePaymentFailures
  expr: rate(omnicrm_stripe_payment_failures_total[5m]) > 5
  for: 5m
  labels:
    severity: warning
  annotations:
    summary: "Échecs de paiement Stripe élevés"
    description: "{{ $value }} échecs de paiement par seconde"
```

---

# Meilleures Pratiques

## Stratégie de Surveillance

1. **Configurer des Alertes de Base** - Configurer des alertes pour des métriques critiques :
    - Taux d'échec de provisionnement > 10%
    - Épuisement du pool de connexions de la base de données
    - Échecs de l'API OCS/CGRateS
    - Erreurs de traitement de paiement Stripe

2. **Suivre les Métriques Commerciales** - Surveiller les KPI opérationnels :
    - Revenus traités via Stripe
    - Débit de provisionnement
    - Taux de livraison d'emails aux clients
3. **Surveillance de la Performance** - Surveiller la dégradation :
    - Percentiles de temps de réponse API
    - Performance des requêtes de base de données
    - Latence des API externes
4. **Planification de Capacité** - Utiliser les métriques pour prévoir :
    - Dimensionnement du pool de connexions de la base de données
    - Mise à l'échelle des travailleurs de tâches en arrière-plan
    - Capacité du serveur API

## Conservation des Métriques

**Recommandations de Conservation de Prometheus :**

- **15 jours** - Métriques haute résolution (intervalle de scraping de 15s)
- **90 jours** - Métriques sous-échantillonnées (agrégation de 5m)
- **1 an** - Métriques agrégées à long terme (agrégation de 1h)

Utilisez Thanos, Cortex ou VictoriaMetrics pour le stockage à long terme et les requêtes globales.

# Dépannage

## Métriques Non Apparentes

Par défaut, l'endpoint `/metrics` est uniquement exposé aux sources internes (non publiques). Vous pouvez modifier cela dans votre configuration nginx si nécessaire.

**Vérifiez l'endpoint de métriques :**

```
curl http://localhost:5000/crm/metrics
```

**Vérifiez que Prometheus peut extraire :**

```
# Vérifiez la page des cibles de Prometheus
http://prometheus:9090/targets
```

## Métriques Spécifiques Manquantes

Certaines métriques ne sont créées qu'après leur première utilisation :

- Les métriques de provisionnement apparaissent après la première tâche de provisionnement
- Les métriques Stripe apparaissent après le premier paiement
- Les métriques OCS apparaissent après la première opération de facturation

## Problèmes de Haute Cardinalité

Si Prometheus est lent ou consomme une mémoire excessive, vérifiez les étiquettes de haute cardinalité :

```
# Comptez les séries temporelles uniques par métrique
count by (__name__) ({__name__=~".+"})
```

Les métriques avec >10 000 séries peuvent indiquer un problème de cardinalité.

---

# Résumé des Métriques

**Total des Métriques :** 34 (31 personnalisées + 3 métriques Flask générées automatiquement)

**Par Type :**

- **Compteurs :** 20
- **Jauges :** 5
- **Histogrammes :** 8
- **Info :** 1

**Par Catégorie :**

- Provisionnement : 5 métriques
- Base de Données : 5 métriques
- OCS/CGRateS : 5 métriques
- Stripe : 5 métriques
- Mailjet : 5 métriques
- Tâches en Arrière-plan : 4 métriques
- HTTP/Flask : 3 métriques
- Info de l'Application : 1 métrique
- Réservé (utilisation future) : 2 métriques

---

# Documentation Connexe

- Architecture du Système - Conception globale du système et relations entre les composants
- Système de Provisionnement - Flux de travail de provisionnement et exécution des tâches
- Playbooks Ansible - Structure et exécution des playbooks
- Documentation API - Endpoints API et authentification
- Vue d'ensemble de la Facturation - Système de facturation et de tarification
- Guide du Système de Paiement - Intégration Stripe et traitement des paiements
- Intégration Mailjet - Configuration du service email
- Configuration d'Administration - Options de configuration du système

# Gestion des Méthodes de Paiement

Le système de Méthodes de Paiement d'OmniCRM permet aux clients et au personnel de gérer en toute sécurité les cartes de paiement en utilisant le **traitement de paiement multi-fournisseurs** (Stripe, PayPal, etc.). Les méthodes de paiement permettent la facturation automatique pour les services, les paiements uniques et les charges récurrentes sans stocker de données sensibles de carte dans OmniCRM.

Voir aussi : Guide du Système de Paiement, Vue d'Ensemble de la Facturation, Traitement des Paiements, Factures.

## Vue d'Ensemble

Le système de méthodes de paiement fournit :

- **Stockage Sécurisé des Cartes** - Cartes tokenisées par les fournisseurs de paiement (Stripe, PayPal), jamais stockées dans OmniCRM
- **Support Multi-Fournisseurs** - Méthodes de paiement Stripe et PayPal prises en charge
- **Plusieurs Cartes** - Les clients peuvent stocker plusieurs méthodes de paiement
- **Sélection par Défaut** - Désignez la méthode de paiement préférée pour les charges automatiques
- **Suivi des Expirations** - Surveillez et mettez à jour les cartes expirantes
- **Auto-Service** - Les clients peuvent gérer leurs propres cartes via le Portail d'Auto-Service
- **Gestion du Personnel** - Le personnel de support peut ajouter/retirer des cartes au nom des clients

**Méthodes de Paiement Prises en Charge :**

- **Cartes** (via Stripe ou PayPal)

    - Cartes de Crédit (Visa, Mastercard, American Express, Discover)
    - Cartes de Débit
    - Cartes Prépayées (si prises en charge par le réseau de cartes)
- **Comptes PayPal** (via l'intégration PayPal)

**Non Stocké dans OmniCRM :**

Les détails de la carte sont tokenisés par les fournisseurs de paiement et stockés en toute sécurité. OmniCRM ne stocke que :

- Fournisseur de paiement (stripe, paypal)
- Marque de la carte (Visa, Mastercard, etc.)
- Les 4 derniers chiffres
- Mois/année d'expiration
- Nom/surnom du titulaire de la carte
- Jeton de méthode de paiement spécifique au fournisseur

# Accéder aux Méthodes de Paiement

**Depuis la Page Client :**

1. Naviguez vers **Clients → [Sélectionner Client]**
2. Cliquez sur l'onglet **Facturation**
3. Faites défiler jusqu'à la section **Méthodes de Paiement**

Ou directement :

**Depuis le Tableau de Bord des Cartes Expirantes :**

Voir tous les clients avec des cartes expirantes :

Cela montre une liste à l'échelle du système des cartes expirant dans les 60 prochains jours.

# Liste des Méthodes de Paiement

Le tableau des méthodes de paiement affiche toutes les cartes stockées pour un client :

**Descriptions des Colonnes :**

- **Surnom** - Nom amical pour la carte (par exemple, "Carte Personnelle", "Visa de Travail")
- **Émetteur** - Marque de la carte et les 4 derniers chiffres
- **Expiration** - Mois/année d'expiration (format MM/YYYY)
- **Ajouté** - Date à laquelle la carte a été ajoutée au compte
- **Par Défaut** - La coche indique la méthode de paiement par défaut pour les charges automatiques

**Actions par Carte :**

Chaque ligne a un menu d'actions ( ⋮ ) avec des options :

- **Définir comme Par Défaut** - Faire de cette carte la méthode de paiement par défaut
- **Supprimer** - Retirer la carte du compte

# Ajouter une Méthode de Paiement

Cliquez sur **"Ajouter une Méthode de Paiement"** pour ouvrir la modal de paiement sécurisée.

## Étape 1 : Entrer les Détails de la Carte

Le formulaire de paiement sécurisé apparaît (alimenté par Stripe Elements ou PayPal SDK) :

**Champs Obligatoires :**

- **Informations sur la Carte** - Numéro de carte, expiration, CVC (validé par Stripe)
- **Nom du Titulaire de la Carte** - Nom sur la carte
- **Pays/Région** - Pays de facturation

**Champs Optionnels :**

- **Surnom de la Carte** - Étiquette amicale pour distinguer les cartes

**Sécurité :**

- Les détails de la carte sont saisis directement dans un iframe sécurisé hébergé par le fournisseur (Stripe Elements / PayPal SDK)

- OmniCRM ne voit jamais ni ne stocke les numéros de carte complets
- La conformité PCI DSS est gérée par le fournisseur de paiement
- La validation en temps réel empêche les numéros de carte invalides

# Étape 2 : Soumettre et Tokeniser

Lorsque vous cliquez sur **"Ajouter une Méthode de Paiement"** :

1. **Validation Côté Client :**
   - Le fournisseur de paiement valide le format du numéro de carte
   - Vérifie que la date d'expiration est dans le futur
   - Vérifie le format CVC
2. **Tokenisation :**
   - Les détails de la carte sont envoyés directement au fournisseur de paiement (pas à OmniCRM)
   - Le fournisseur crée un jeton sécurisé (par exemple, `pm_1A2B3C4D` pour Stripe)
   - Le jeton est renvoyé à OmniCRM
3. **Traitement Serveur :**
   - OmniCRM enregistre le jeton dans le dossier client avec l'identifiant du fournisseur
   - Stocke les 4 derniers chiffres, la marque, l'expiration et le nom du fournisseur pour l'affichage
   - Aucun numéro de carte complet ne touche jamais les serveurs d'OmniCRM

# Étape 3 : Confirmation

Un message de succès apparaît :

> Votre Visa se terminant par 1234 a été ajoutée à votre compte.

La nouvelle carte apparaît dans le tableau des méthodes de paiement.

**Sélection Automatique par Défaut :**

- Si c'est la première carte du client, le système la définit automatiquement comme par défaut
- Si le client a déjà des cartes, la nouvelle carte est ajoutée comme non par défaut
- Le client peut changer la méthode par défaut après l'ajout

# Définir la Méthode de Paiement par Défaut

La méthode de paiement par défaut est utilisée pour :

- Charges de service récurrentes automatiques
- Paiements de factures
- Recharges et recharges
- Transactions uniques (sauf indication contraire)

**Pour Changer le Par Défaut :**

1. Localisez la carte que vous souhaitez définir comme par défaut dans le tableau des méthodes de paiement

2. Cliquez sur le **menu d'actions ( ⋮ )** à côté de la carte

3. Sélectionnez **"Définir comme Par Défaut"**

4. Une confirmation apparaît

   La Visa se terminant par 5678 est maintenant votre méthode de paiement par défaut.

La coche se déplace vers la carte nouvellement sélectionnée.

**Indicateur Visuel :**

Les cartes par défaut affichent :

dans la colonne Par Défaut, généralement avec un badge de coche verte.

# Supprimer une Méthode de Paiement

Retirez les cartes qui sont expirées, perdues ou dont vous n'avez plus besoin.

## Étape 1 : Initier la Suppression

1. Trouvez la carte à supprimer dans le tableau des méthodes de paiement
2. Cliquez sur l'icône **corbeille** sur cette ligne

Si le client n'a qu'une seule méthode de paiement active, l'icône de corbeille est **désactivée**. En survolant, un tooltip s'affiche :

> Impossible de supprimer votre seule méthode de paiement. Ajoutez une autre méthode de paiement d'abord.

Ajoutez une deuxième méthode de paiement avant de pouvoir retirer l'existante.

## Étape 2 : Confirmer la Suppression

Lorsque la suppression est autorisée, une modal de confirmation apparaît :

> Êtes-vous sûr de vouloir supprimer cette méthode de paiement ?
>
> [Annuler] [Supprimer la Méthode de Paiement]

Si la méthode à supprimer est la par défaut actuelle et qu'il existe d'autres méthodes actives, la modal note qu'une autre méthode active sera promue par défaut après la suppression.

Cliquez sur **"Supprimer la Méthode de Paiement"** pour confirmer.

## Étape 3 : Suppression Complète

Message de succès :

La carte est retirée du tableau et supprimée du fournisseur de paiement.

**Restrictions Importantes :**

- **Impossible de supprimer la seule méthode de paiement active** - Le contrôle de suppression est désactivé (avec tooltip). L'API rejette également la demande. Ajoutez une autre méthode de paiement d'abord.
- **Supprimer la méthode par défaut lorsque d'autres existent** - Une autre méthode de paiement active utilisable est automatiquement promue par défaut.
- **Pas de retour en arrière** - La suppression est permanente ; le client doit réajouter la méthode si nécessaire

# Gestion des Cartes Expirantes

OmniCRM suit les dates d'expiration des cartes et fournit des outils pour mettre à jour proactivement les cartes expirantes.

## Tableau de Bord des Cartes Expirantes

Naviguez vers **Facturation → Cartes Expirantes** pour voir une liste à l'échelle du système :

> Client Carte Expiration Jours Avant Action John Smith Visa ****1234 02/2025 12 jours Mettre à Jour Acme Corp MC5678 03/2025 45 jours Mettre à Jour Jane Doe Amex**9012 01/2025 EXPIRÉE Mettre à Jour

**Filtres :**

- **Plage d'Expiration** - Prochains 30/60/90 jours ou déjà expiré
- **Type de Client** - Individuel vs Entreprise
- **Type de Service** - Filtrer par service nécessitant une méthode de paiement

**Actions :**

- **Mettre à Jour** - Ouvre la page des méthodes de paiement du client pour ajouter une nouvelle carte
- **Notifier** - Envoyer un rappel par e-mail au client (si Mailjet configuré)

## Notifications d'Expiration

Si Mailjet est configuré, des e-mails automatiques sont envoyés :

- **60 jours avant l'expiration** - Premier rappel
- **30 jours avant l'expiration** - Deuxième rappel
- **7 jours avant l'expiration** - Avertissement final
- **À l'expiration** - Avis de carte expirée

Les clients peuvent cliquer sur un lien dans l'e-mail pour mettre à jour leur méthode de paiement via le portail d'auto-service.

**Variables de Modèle d'E-mail :**

Les modèles Mailjet reçoivent :

- Nom du client
- Marque de la carte et les 4 derniers chiffres
- Date d'expiration
- Lien vers la page des méthodes de paiement d'auto-service

Voir `integrations_mailjet` pour la configuration du modèle d'e-mail.

# Mise à Jour d'une Carte Expirante

**Flux de Travail Recommandé :**

1. Le client reçoit un e-mail de notification d'expiration
2. Le client se connecte au portail d'auto-service
3. Navigue vers **Facturation → Méthodes de Paiement**
4. Clique sur **"Ajouter une Méthode de Paiement"**
5. Entre les nouveaux détails de la carte (même carte avec expiration mise à jour, ou carte de remplacement)
6. Définit la nouvelle carte comme par défaut
7. Supprime l'ancienne carte expirée

**Flux de Travail du Personnel :**

Si le client appelle le support :

1. Le personnel ouvre le compte du client
2. Navigue vers **Facturation → Méthodes de Paiement**
3. Ajoute une nouvelle carte au nom du client (le client fournit les détails par téléphone)
4. Définit la nouvelle carte comme par défaut
5. Supprime la carte expirée

6. Confirme avec le client

> ⚠ **ATTENTION**
>
> Ne demandez jamais aux clients d'envoyer des détails de carte par e-mail ou par SMS. Utilisez toujours :
>
> - Le portail d'auto-service sécurisé pour l'auto-service
> - Le téléphone avec le personnel saisissant directement les détails dans le système
> - En personne dans un point de vente

## Que Se Passe-t-il Lorsque les Cartes Expirent

Lorsqu'une carte de paiement atteint sa date d'expiration et n'est pas mise à jour :

**Effets Immédiats :**

1. **Les Paiements Automatiques Échouent**
   - Le fournisseur de paiement rejette les transactions avec des cartes expirées
   - Les renouvellements de services mensuels échouent à traiter
   - Les recharges automatiques échouent
   - Les paiements automatiques de factures échouent
2. **Notifications aux Clients**
   - Le système tente de facturer la carte
   - Notification d'échec de paiement envoyée
   - E-mail "Mettre à Jour la Méthode de Paiement" envoyé avec un lien vers le portail d'auto-service
3. **Changements de Statut de Service**
   - **Services Postpayés** - Peuvent continuer temporairement avec un solde impayé
   - **Services Prépayés** - Suspension du service lorsque le solde s'épuise
   - **Services de Renouvellement Automatique** - Le renouvellement échoue, le service peut expirer

**Actions Subséquentes :**

**Jour 1-3 (Période de Grâce) :**

- Le service continue normalement
- Le client reçoit le premier avis d'échec de paiement
- Le système tente une nouvelle tentative (selon la configuration)

**Jour 4-7 :**

- Deuxième tentative de paiement (si configuré)
- E-mail d'avertissement envoyé
- Le service client peut contacter le client

**Jour 8-14 :**

- Le service peut être suspendu pour non-paiement
- Le statut suspendu empêche l'utilisation mais préserve le compte
- Le client peut restaurer en mettant à jour la méthode de paiement et en payant le solde impayé

**Jour 15+ :**

- Le service peut être résilié pour non-paiement
- L'inventaire (cartes SIM, équipements) est marqué pour retour
- Avis final envoyé
- Compte référé à des recouvrements (si applicable)

**Prévenir l'Interruption de Service :**

Pour éviter une interruption de service :

- Mettez à jour les cartes **30 jours avant l'expiration**
- Ajoutez plusieurs méthodes de paiement pour la redondance
- Activez les alertes d'échec de paiement
- Surveillez le tableau de bord des Cartes Expirantes chaque semaine

**Restaurer le Service Après Expiration :**

Si le service est suspendu en raison d'une carte expirée :

1. Ajoutez une nouvelle méthode de paiement valide
2. Définissez-la comme par défaut
3. Payez le solde impayé (le cas échéant)
4. Contactez le support pour réactiver le service
5. Service restauré en quelques minutes à quelques heures

# Sécurité des Méthodes de Paiement

## Tokenisation

OmniCRM utilise la tokenisation des fournisseurs pour garantir la sécurité :

1. **Le client entre la carte** → Envoyée directement aux serveurs du fournisseur de paiement
2. **Le fournisseur valide et tokenise** → Crée un jeton unique
3. **Le jeton est stocké dans OmniCRM** → Le numéro de carte complet n'est jamais stocké
4. **Traitement des paiements** → Le jeton est envoyé au fournisseur, le fournisseur facture la carte

**Ce que Stocke OmniCRM :**

```
{
  "vendor": "stripe",
  "vendor_payment_method_id": "pm_1A2B3C4D5E6F",
  "payment_type": "card",
  "brand": "visa",
  "last4": "1234",
  "exp_month": 12,
  "exp_year": 2026,
  "name": "John Smith",
  "nickname": "Carte Personnelle",
  "is_default": true
}
```

**Ce que OmniCRM NE Stocke PAS :**

- Numéro de carte complet
- Code CVV/CVC
- Données de bande magnétique
- Numéros de PIN

## Conformité PCI

En utilisant des formulaires de paiement hébergés par le fournisseur :

- **Réduction de la portée PCI** - Les données de carte ne touchent jamais les serveurs d'OmniCRM
- **Champs hébergés par le fournisseur** - L'entrée de carte se fait dans l'iframe sécurisé du fournisseur
- **Pas de stockage de carte** - Des jetons sont utilisés au lieu de données de carte brutes
- **Transmission sécurisée** - Toute communication se fait via HTTPS/TLS

Voir Guide du Système de Paiement pour les détails de sécurité du fournisseur de paiement.

# Flux de Travail Courants

## Flux de Travail 1 : Le Client Ajoute sa Première Méthode de Paiement

**Scénario :** Nouveau client s'inscrivant pour un service

1. Le client crée un compte
2. Sélectionne un plan de service
3. Invite à ajouter une méthode de paiement lors du passage à la caisse
4. Entre les détails de la carte dans la modal Stripe
5. La carte est tokenisée et enregistrée
6. Automatiquement définie comme par défaut
7. Service provisionné
8. Première charge traitée

## Flux de Travail 2 : Le Client Met à Jour une Carte Expirante

**Scénario :** Carte de crédit sur le point d'expirer

1. Le client reçoit une notification par e-mail (60 jours avant l'expiration)
2. Se connecte au portail d'auto-service
3. Navigue vers **Facturation → Méthodes de Paiement**
4. Examine la carte actuelle expirant 12/2025
5. Clique sur **"Ajouter une Méthode de Paiement"**
6. Entre une carte de remplacement avec expiration 12/2028
7. Définit la nouvelle carte comme par défaut
8. Supprime l'ancienne carte
9. E-mail de confirmation envoyé

# Flux de Travail 3 : Le Personnel Aide le Client par Téléphone

**Scénario :** Le client appelle : "Ma carte a été refusée"

1. Le client appelle le support
2. Le personnel vérifie l'identité (questions de sécurité)
3. Le personnel vérifie les méthodes de paiement : Carte expirée 01/2025
4. Le personnel : "Votre carte a expiré. Avez-vous une nouvelle carte ?"
5. Le client fournit les nouveaux détails de carte par téléphone
6. Le personnel navigue vers **Clients → [Client] → Facturation**
7. Clique sur **"Ajouter une Méthode de Paiement"**
8. Entre les détails de la carte pendant que le client les lit
9. Définit la nouvelle carte comme par défaut
10. Supprime la carte expirée
11. Réessaie le paiement échoué
12. Confirme avec le client : "Paiement réussi, service restauré"

# Flux de Travail 4 : Client Entreprise avec Plusieurs Cartes

**Scénario :** L'entreprise souhaite différentes cartes pour différents usages

1. Le client entreprise ajoute la carte principale (Visa se terminant par 1111)
2. La définit comme par défaut pour les charges de service mensuelles
3. Ajoute une carte de secours (Mastercard se terminant par 2222) pour les recharges
4. Ajoute une carte d'achat (Amex se terminant par 3333) pour les achats d'équipement
5. Lors de la recharge, sélectionne manuellement la Mastercard lors du passage à la caisse
6. La Visa par défaut est toujours utilisée pour la facturation mensuelle automatique

## Flux de Travail 5 : Gestion des Cartes Expirantes (Admin)

**Scénario :** Gestion proactive des expirations

1. L'admin navigue vers **Facturation → Cartes Expirantes**
2. Filtres : "Prochains 30 jours"
3. Voit 15 clients avec des cartes expirantes
4. Sélectionne tout → **"Envoyer des E-mails de Rappel"**
5. Mailjet envoie des e-mails personnalisés à chaque client
6. Les clients mettent à jour les cartes via l'auto-service
7. L'admin examine la liste 1 semaine plus tard
8. Appelle les clients restants qui n'ont pas mis à jour
9. Aide à la mise à jour des cartes par téléphone

# Dépannage

**"Carte refusée" lors de l'ajout de la méthode de paiement**

- **Cause :** Le fournisseur de paiement a rejeté la carte (fonds insuffisants, prévention de fraude, refus de l'émetteur)
- **Solution :**
  - Essayez une autre carte
  - Contactez l'émetteur de la carte pour autoriser la transaction
  - Assurez-vous que la carte prend en charge les achats en ligne
  - Vérifiez que l'adresse de facturation correspond à celle enregistrée sur la carte

**"Erreur lors de l'ajout de la méthode de paiement" (erreur générique)**

- **Cause :** Erreur API du fournisseur de paiement ou problème de réseau
- **Solution :**
  - Actualisez la page et réessayez
  - Vérifiez la connexion Internet

    - Vérifiez que la configuration du fournisseur de paiement est correcte dans les paramètres du système
    - Vérifiez la console du navigateur pour un message d'erreur spécifique
    - Essayez un autre navigateur (désactivez les bloqueurs de publicité)

**Impossible de supprimer la méthode de paiement (bouton désactivé)**

- **Cause :** C'est la seule méthode de paiement active du client
- **Solution :**
    - Ajoutez une nouvelle méthode de paiement d'abord, puis supprimez l'ancienne
    - Survolez l'icône de corbeille désactivée pour l'explication du tooltip

**La carte apparaît comme expirée mais pas dans la liste "Cartes Expirantes"**

- **Cause :** La carte a expiré récemment, le cache n'a pas été actualisé
- **Solution :**
    - Actualisez la page
    - Vérifiez les filtres sur le tableau de bord des Cartes Expirantes
    - Les cartes expirées peuvent passer à une vue différente

**Nouvelle carte n'apparaissant pas immédiatement**

- **Cause :** La page n'a pas été actualisée après l'ajout de la carte
- **Solution :**
    - Le tableau des méthodes de paiement devrait se rafraîchir automatiquement
    - Si ce n'est pas le cas, actualisez manuellement le navigateur
    - Vérifiez si une erreur s'est produite lors du processus d'ajout

**La modal de paiement ne se charge pas**

- **Cause :** SDK du fournisseur de paiement ne se charge pas, problème de clé API, ou extension de navigateur bloquant
- **Solution :**
    - Vérifiez la console du navigateur pour des erreurs

    - Désactivez les bloqueurs de publicité et la protection contre le suivi
    - Vérifiez la configuration du fournisseur de paiement dans les paramètres du système
    - Assurez-vous que le script SDK du fournisseur se charge (vérifiez l'onglet Réseau)
    - Essayez le mode incognito/navigation privée

**Le client ne reçoit pas de notifications d'expiration**

- **Cause :** Mailjet non configuré ou modèle d'e-mail manquant
- **Solution :**
    - Vérifiez les identifiants Mailjet dans *crm_config.yaml*
    - Vérifiez que le modèle d'e-mail existe pour l'expiration de la carte
    - Confirmez que l'adresse e-mail du client est valide
    - Vérifiez les journaux Mailjet pour les échecs de livraison

# Meilleures Pratiques

**Pour les Clients :**

- Ajoutez une méthode de paiement avant l'activation du service pour éviter les retards
- Gardez au moins 2 cartes sur fichier pour la redondance
- Mettez à jour les cartes expirantes 30+ jours avant l'expiration
- Supprimez les anciennes/cartes expirées pour éviter la confusion
- Utilisez des surnoms descriptifs ("Visa Personnelle", "Amex de Travail")
- Vérifiez que la méthode de paiement par défaut est correcte pour la facturation automatique

**Pour le Personnel de Support :**

- Vérifiez l'identité du client avant d'accéder aux méthodes de paiement
- Ne demandez jamais aux clients d'envoyer des détails de carte par e-mail/SMS/chat
- Traitez les ajouts de cartes immédiatement lors des appels (ne pas différer)

- Confirmez que la nouvelle carte est définie comme par défaut après l'ajout
- Supprimez les anciennes cartes seulement après avoir confirmé que la nouvelle carte fonctionne
- Testez le paiement après la mise à jour d'une carte expirée (traitez une autorisation de £0.01)

**Pour les Administrateurs :**

- Surveillez le tableau de bord des Cartes Expirantes chaque semaine
- Envoyez des e-mails de rappel 60/30/7 jours avant l'expiration
- Gardez les clés de test/live du fournisseur de paiement séparées pour le développement et la production
- Assurez-vous que les modèles Mailjet sont configurés pour les notifications d'expiration
- Examinez les rapports de paiement échoué pour identifier les cartes expirées
- Formez le personnel sur les procédures de gestion sécurisée des cartes

# Documentation Connexe

- Guide du Système de Paiement - Référence complète de l'API de paiement et architecture
- `payments_process` - Traitement des paiements avec des méthodes de paiement stockées
- `payments_invoices` - Paiement automatique des factures en utilisant la carte par défaut
- `features_topup_recharge` - Système de recharge utilisant des méthodes de paiement
- `basics_payment` - Concepts généraux de paiement et de facturation
- `customer_care` - Portail d'auto-service pour les clients afin de gérer leurs propres cartes

# Guide de l'API du Système de Paiement OmniCRM

## Vue d'ensemble

Le système de paiement OmniCRM fournit une infrastructure de traitement des paiements **indépendante des fournisseurs**. Aujourd'hui, il prend en charge Stripe et PayPal, mais l'architecture modulaire permet l'intégration avec n'importe quel fournisseur de paiement (Square, Adyen, Braintree, etc.) sans modifier le code de l'application.

Ce document couvre toutes les API de paiement et les flux de travail disponibles dans le système.

> **Intégration du Playbook Ansible** : Pour implémenter ces API de paiement dans les playbooks de provisionnement, voir **Facturation et Paiements depuis les Playbooks**

---

## Table des matières


1. Architecture Modulaire
2. Concepts de Base
3. Documents Financiers
4. API des Méthodes de Paiement
5. API des Flux de Paiement
6. API de Portefeuille
7. Résumé de la Référence API
8. Cas d'utilisation courants


---

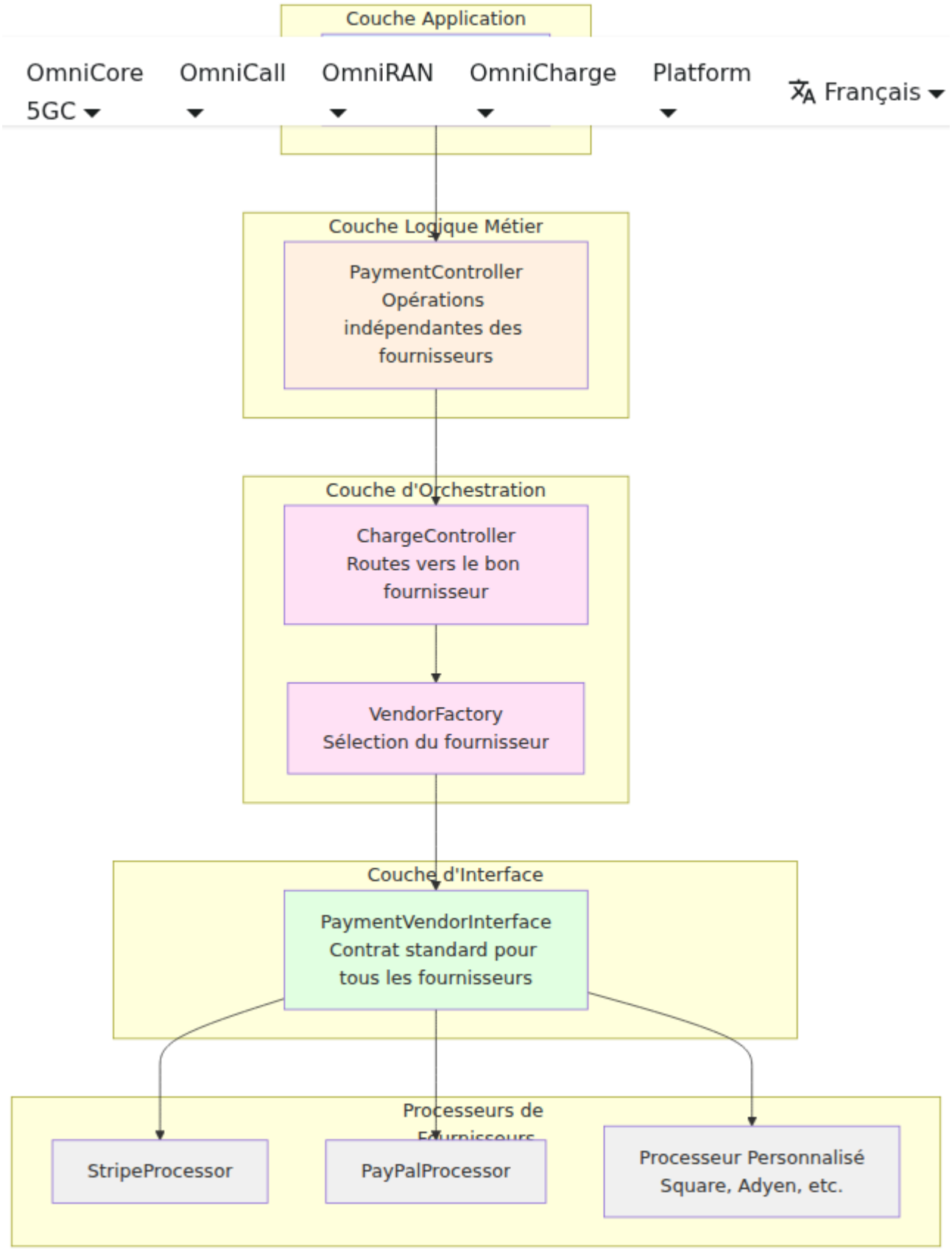
# Architecture Modulaire

## Pourquoi Indépendant des Fournisseurs ?

Le système utilise des **couches d'abstraction** pour séparer la logique métier des spécificités des fournisseurs de paiement. Cela signifie :

- □ Ajouter de nouveaux fournisseurs de paiement sans toucher au code de l'application
- □ Changer de fournisseur sans migrations de base de données
- □ Prendre en charge plusieurs fournisseurs simultanément
- □ API cohérente, quel que soit le fournisseur backend

# Couches d'Architecture



# Ajouter un Nouveau Fournisseur de Paiement

Pour ajouter un nouveau fournisseur (par exemple, Square, Adyen, Braintree), contactez votre équipe technique OmniCRM.

Pour des détails de configuration sur les fournisseurs existants (Stripe, PayPal), voir **Configuration du Fournisseur**.

**Vue d'ensemble du Processus** :




**Étapes** :

1. **Implémenter la classe de processeur** avec l'interface standard `PaymentVendorInterface` (autoriser, capturer, facturer, rembourser, libérer)
2. **Enregistrer le processeur** dans VendorFactory avec le nom du fournisseur
3. **Tester l'intégration** avec l'environnement sandbox du fournisseur
4. **Déployer** - Toutes les API existantes prennent automatiquement en charge le nouveau fournisseur

**Résultat** : Une fois déployé, le nouveau fournisseur fonctionne sans problème :

```
# Ajouter une méthode de paiement Square
curl -X POST /api/payments/methods \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "customer_id": 123,
    "vendor": "square",
    "payment_token": "sq_xxxxx"
  }'
```

Toutes les API de paiement prennent automatiquement en charge le nouveau fournisseur sans modifications du code de l'application.

**Temps d'Intégration** : Typiquement 1 à 2 jours pour un nouveau processeur de fournisseur.

---

# Configuration du Fournisseur

## Activation des Fournisseurs de Paiement

L'interface utilisateur contrôle quels fournisseurs de paiement sont disponibles pour les clients. Cela se règle via la variable `VITE_PAYMENT_VENDORS` dans le fichier `.env` de l'interface utilisateur :

```
# Activer uniquement Stripe (par défaut)
VITE_PAYMENT_VENDORS="stripe"

# Activer à la fois Stripe et PayPal
VITE_PAYMENT_VENDORS="stripe,paypal"

# Activer uniquement PayPal
VITE_PAYMENT_VENDORS="paypal"
```

**Important :** `VITE_PAYMENT_VENDORS` est une variable **de temps de construction**. La modification nécessite de reconstruire et de redémarrer l'interface utilisateur.

## Configuration de Stripe

Stripe est le fournisseur de paiement principal, fournissant le traitement des cartes avec le support 3D Secure.

### 1. Obtenir les Clés API Stripe

- Inscrivez-vous sur https://stripe.com
- Accédez à Développeurs → Clés API
- Copiez votre **Clé Publiable** (pk_live_... ou pk_test_...)
- Copiez votre **Clé Secrète** (sk_live_... ou sk_test_...)

**Important :**

- Utilisez des **clés de test** (pk_test_/sk_test_) pour le développement
- Utilisez des **clés en direct** (pk_live_/sk_live_) uniquement pour la production
- **Ne jamais** commettre les clés API dans le contrôle de version

**2. Configurer le Backend**

Ajoutez à `crm_config.yaml` (niveau racine, pas imbriqué sous `payment_vendors`) :

```
stripe:
  secret_key: 'sk_live_YOUR_SECRET_KEY_HERE'
  publishable_key: 'pk_live_YOUR_PUBLISHABLE_KEY_HERE'
  currency: 'aud'
  statement_descriptor_suffix: 'YOUR_COMPANY'
```

L'API lit les identifiants Stripe depuis `crm_config.yaml` au démarrage. Modifier ces valeurs nécessite de redémarrer l'API.

**3. Configurer le Frontend**

Ajoutez au fichier `.env` de l'interface utilisateur :

```
VITE_STRIPE_PUBLISHABLE_KEY=pk_live_YOUR_PUBLISHABLE_KEY_HERE
```

**Note de Sécurité :** Seule la clé publiable doit être dans le frontend. La clé secrète ne doit **jamais** être exposée au navigateur.

# Configuration de PayPal

PayPal fournit le stockage de cartes et les paiements par compte PayPal.

**1. Obtenir les Identifiants API PayPal**

- Créez une application API REST sur https://developer.paypal.com/dashboard/

- Copiez votre **Client ID** et **Client Secret**
- Pour les tests, utilisez les identifiants **Sandbox** ; pour la production, utilisez les identifiants **Live**

### 2. Configurer le Backend

Ajoutez à `crm_config.yaml` (niveau racine) :

```yaml
paypal:
  client_id: 'YOUR_PAYPAL_CLIENT_ID'
  client_secret: 'YOUR_PAYPAL_CLIENT_SECRET'
  mode: 'sandbox'  # 'sandbox' pour les tests, 'live' pour la production
```

L'API lit les identifiants PayPal depuis `crm_config.yaml` au démarrage. Modifier ces valeurs nécessite de redémarrer l'API.

### 3. Configurer le Frontend

Ajoutez au fichier `.env` de l'interface utilisateur :

```
VITE_PAYMENT_VENDORS="stripe,paypal"
VITE_PAYPAL_CLIENT_ID=YOUR_PAYPAL_CLIENT_ID
```

`VITE_PAYPAL_CLIENT_ID` doit correspondre à la valeur `paypal.client_id` dans `crm_config.yaml`.

## Sandbox vs Live

| | Sandbox | Live |
|---|---|---|
| **API Stripe** | `sk_test_` / `pk_test_` | `sk_live_` / `pk_live_` |
| **API PayPal** | `api-m.sandbox.paypal.com` | `api-m.paypal.com` |
| **PayPal `mode`** | `sandbox` | `live` |
| **Argent** | Fictif | Réel |

Lors du passage de sandbox à live, mettez à jour **les deux** `crm_config.yaml` (identifiants et mode) et le fichier `.env` de l'interface utilisateur (clé publiable / ID client), puis reconstruisez l'interface utilisateur et redémarrez l'API.

# Conformité PCI

**Comment OmniCRM Maintient la Conformité PCI :**

- Les données de carte sont saisies directement dans des iframes hébergées par le fournisseur (Stripe Elements, PayPal Card Fields)
- OmniCRM **ne voit jamais ni ne stocke** les numéros de carte complets
- Seules les méthodes de paiement tokenisées sont stockées dans la base de données
- Portée de conformité PCI réduite pour votre entreprise

# Métriques de Traitement des Paiements

OmniCRM fournit des métriques complètes pour surveiller les opérations de traitement des paiements. Pour des détails complets sur les métriques de paiement Stripe, y compris le suivi des appels API, les volumes de paiement, les taux d'échec et les temps de réponse, voir Surveillance & Métriques.

---

## Le Schéma de Base de Données est Indépendant des Fournisseurs

Le schéma de base de données prend en charge n'importe quel fournisseur de paiement sans migrations :




**Exemple** - Sauvegarde d'une carte Square :

```
{
  "vendor": "square",
  "vendor_payment_method_id": "sq_card_abc123",
  "payment_type": "carte"
}
```

Aucun changement de code. Aucune migration. Ça fonctionne simplement.

---

# Concepts de Base

## Modèles de Données

### PaymentMethod

Stockage indépendant des fournisseurs des méthodes de paiement des clients.

```
{
  "payment_method_id": 789,
  "customer_id": 123,
  "vendor": "stripe",                    // 'stripe', 'paypal', ou
tout fournisseur ajouté
  "vendor_payment_method_id": "pm_xxx", // ID interne du
fournisseur
  "payment_type": "carte",               // 'carte', 'paypal',
'ach', etc.
  "is_default": true,
  "card_brand": "visa",
  "card_last4": "4242",
  "card_exp_month": 12,
  "card_exp_year": 2025,
  "card_nickname": "Ma Carte Visa",
  "status": "actif"
}
```

**PaymentAuthorization**

Enregistrements d'autorisation de commit en deux phases (retenir des fonds).

```
{
  "authorization_id": 301,
  "customer_id": 123,
  "payment_method_id": 789,
  "vendor": "stripe",                    // Quel fournisseur a
autorisé
  "vendor_authorization_id": "auth_xxx", // ID d'autorisation du
fournisseur
  "amount": 200.00,
  "currency": "USD",
  "status": "authorized",                // 'authorized',
'captured', 'released'
  "authorized_at": "2025-12-27T10:00:00Z",
  "expires_at": "2026-01-03T10:00:00Z",
  "meta": {}
}
```

**PaymentCapture**

Paiements capturés/complétés.

```
{
  "capture_id": 103,
  "authorization_id": 301,
  "customer_id": 123,
  "payment_method_id": 789,
  "vendor": "stripe",
  "vendor_transaction_id": "ch_xxx",     // ID de transaction du
fournisseur
  "amount": 200.00,
  "currency": "USD",
  "status": "succeeded",                 // 'succeeded', 'failed',
'refunded'
  "captured_at": "2025-12-27T10:30:00Z",
  "vendor_response": {},                 // Réponse complète du
fournisseur
  "meta": {}
}
```

**WalletAccount**

Portefeuille du client avec suivi de solde (1-à-1 avec le client).

```
{
  "wallet_account_id": 456,
  "customer_id": 123,
  "balance": 150.50,
  "currency": "USD",
  "auto_recharge_enabled": true,
  "auto_recharge_amount": 100.00,
  "auto_recharge_threshold": 10.00,
  "low_balance_warning_threshold": 10.00
}
```

**WalletLedger**

Piste d'audit complète de toutes les transactions de portefeuille.

```
{
  "ledger_id": 501,
  "customer_id": 123,
  "wallet_account_id": 456,
  "transaction_type": "credit",          // 'credit', 'debit',
'refund', 'adjustment'
  "amount": 100.00,
  "balance_before": 150.50,
  "balance_after": 250.50,
  "currency": "USD",
  "description": "Recharge de carte",
  "reference_type": "payment_capture",   // Lien vers l'objet
associé
  "reference_id": 103,
  "meta": {},
  "created_at": "2025-12-27T10:35:00Z"
}
```

---

# Documents Financiers

Pour le modèle et la personnalisation des factures, voir **Factures Clients**.

## Factures

**Définition** : Une facture est un document qui contient une liste de **transactions DEBIT** (charges). Lorsque l'API est appelée pour créer une facture, un tableau d'IDs de transactions de débit doit être fourni. Ces transactions sont "liées" à la facture en définissant leur champ `invoice_id`.

Pour des détails sur la gestion des transactions, voir **Transactions**.

**Champs Clés** :

```
{
  "invoice_id": 12345,
  "invoice_number": "INV-2025-000001",   // Auto-généré : INV-YYYY-NNNNNN
  "customer_id": 123,
  "title": "Facture des Services Mensuels",
  "paid": true,                          // Statut de paiement
  "void": false,                         // Statut d'annulation
  "payment_reference": "ch_xxxxx",       // Dernier/ID de paiement principal
  "payment_type": "stripe_capture",      // Dernière méthode de paiement
  "payment_time": "2025-12-27T10:30:00",
  "start_date": "2025-12-01",
  "end_date": "2025-12-31",
  "due_date": "2026-01-15",
  "retail_cost": 500.00,                 // Montant total de la facture
  "wholesale_cost": 250.00
}
```

**Génération de Factures** : Les numéros de facture sont automatiquement générés au format `INV-YYYY-NNNNNN`, séquentiels au sein de l'année civile, se réinitialisant chaque 1er janvier.

## Paiements sur Factures

**Comment Fonctionnent les Paiements** : Une fois qu'une facture est créée à partir de transactions de débit, un **PAIEMENT** peut être appliqué en créant une **transaction de crédit** (coût de vente négatif) liée à la facture.

Les paiements doivent :

- Être listés sur la facture clairement marqués comme "PAIEMENTS"
- Afficher la date de paiement pertinente (qui peut différer de la date de création de la facture)
- Prendre en charge plusieurs paiements par facture
- Compter contre les débits pour calculer le solde de la facture

**Statut de la Facture** :

- **PAYÉE** : Les paiements totaux (crédits) sont égaux ou supérieurs aux débits totaux
- **PARTIELLEMENT PAYÉE** : Certains paiements appliqués mais un solde reste
- **SURPAYÉE** : Actuellement non géré - nécessite un fractionnement de crédit (fonctionnalité future)

**Comptabilité en Partie Double** : Le système met en œuvre une comptabilité appropriée où chaque charge a un paiement compensatoire :

```
// 1. Transaction de débit (charge)
{
  "transaction_id": 7001,
  "invoice_id": 12345,
  "retail_cost": 100.00,                // Positif = le client
doit
  "title": "Frais de Service"
}

// 2. Transaction de crédit (paiement)
{
  "transaction_id": 7002,
  "invoice_id": 12345,
  "retail_cost": -100.00,               // Négatif = paiement
reçu
  "title": "Paiement pour la Facture : Frais de Service (12345)",
  "payment_type": "stripe_capture",
  "payment_reference": "ch_xxxxx"
}

// Résultat net : solde de 0 $ → facture marquée comme PAYÉE
```

**Métadonnées Supplémentaires de Facture** : En plus des transactions de crédit, la facture stocke également des champs de résumé (`payment_reference`, `payment_type`, `payment_time`) pour des recherches rapides. Cependant :

- **Méthode principale** : Transactions de crédit liées via `invoice_id` (spécification de Hayden)
- **Métadonnées secondaires** : Les champs de facture stockent un résumé du dernier/paiement principal
- **Pour plusieurs paiements** : Interroger les transactions de crédit pour l'historique complet

# Relevés

**Définition** : Un relevé montre tous les **DÉBITS et CRÉDITS** des transactions d'un client sur une période spécifiée. C'est le seul type de document où les débits et crédits sont affichés en tant qu'éléments de ligne ensemble, comme un relevé bancaire.

# Notes de Crédit

**Objectif** : Appliquées uniquement aux factures avec paiements déjà appliqués. Doit être fait dans le cadre du processus d'annulation de la facture.

**Processus** :

1. Une facture est annulée (avec ses transactions DEBIT associées)
2. Tous les PAIEMENTS déjà appliqués sont liés à la NOTE DE CRÉDIT créée
3. Le solde du client passe en crédit par le montant équivalent
4. Le solde de la note de crédit peut ensuite être :
    - Appliqué à une autre facture en tant que PAIEMENT, OU
    - REMBOURSÉ au client

**Logique de Remboursement** : Si le remboursement est choisi, il est invoqué en fonction des transactions de CRÉDIT associées et de la manière dont elles ont été initialement payées (par exemple, remboursement Stripe si le type de transaction de crédit était Stripe).

---

# API des Méthodes de Paiement

Tous les points de terminaison utilisent l'URL de base : `/api/payments/`

Pour la gestion détaillée des méthodes de paiement et le traitement des cartes, voir **Méthodes de Paiement**.

## Ajouter une Méthode de Paiement

Sauvegarder une nouvelle méthode de paiement pour un client.

**Point de terminaison** : `POST /api/payments/methods`

**Demande** :

```
curl -X POST https://your-domain.com/api/payments/methods \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "customer_id": 123,
    "vendor": "stripe",
    "payment_token": "pm_xxxxx",
    "is_default": false,
    "card_nickname": "Ma Carte Visa"
  }'
```

**Corps de la Demande** :

```
{
  "customer_id": 123,
  "vendor": "stripe",                    // 'stripe', 'paypal', ou
tout fournisseur ajouté
  "payment_token": "pm_xxxxx",           // Jeton unique du SDK
frontend
  "is_default": false,                   // Définir comme méthode
de paiement par défaut ?
  "card_nickname": "Ma Carte Visa"        // Nom amical facultatif
}
```

**Réponse** (201 Créé) :

```
{
  "success": true,
  "message": "Méthode de paiement ajoutée avec succès",
  "data": {
    "payment_method_id": 789,
    "customer_id": 123,
    "vendor": "stripe",
    "payment_type": "carte",
    "card_brand": "visa",
    "card_last4": "4242",
    "card_exp_month": 12,
    "card_exp_year": 2025,
    "card_nickname": "Ma Carte Visa",
    "is_default": false,
    "status": "actif"
  }
}
```

## Obtenir les Méthodes de Paiement

Récupérer toutes les méthodes de paiement pour un client.

**Point de terminaison** : `GET /api/payments/methods?customer_id={id}`

**Demande** :

```
curl -X GET "https://your-domain.com/api/payments/methods?
customer_id=123" \
  -H "Authorization: Bearer YOUR_API_KEY"
```

**Réponse** (200 OK) :

```
{
  "success": true,
  "data": [
    {
      "payment_method_id": 789,
      "customer_id": 123,
      "vendor": "stripe",
      "payment_type": "carte",
      "card_brand": "visa",
      "card_last4": "4242",
      "is_default": true
    },
    {
      "payment_method_id": 790,
      "customer_id": 123,
      "vendor": "paypal",
      "payment_type": "paypal",
      "paypal_email": "user@example.com",
      "is_default": false
    }
  ]
}
```


## Obtenir la Méthode de Paiement par Défaut

Obtenir la méthode de paiement par défaut pour un client.

**Point de terminaison** : `GET /api/payments/methods/default?customer_id={id}`

**Demande** :

```
curl -X GET "https://your-domain.com/api/payments/methods/default?customer_id=123" \
  -H "Authorization: Bearer YOUR_API_KEY"
```

**Réponse** (200 OK) :

```
{
  "success": true,
  "data": {
    "payment_method_id": 789,
    "vendor": "stripe",
    "payment_type": "carte",
    "card_brand": "visa",
    "card_last4": "4242",
    "is_default": true
  }
}
```


## Définir la Méthode de Paiement par Défaut

Définir une méthode de paiement comme par défaut.

**Point de terminaison** : `PUT /api/payments/methods/set-default`

**Demande** :

```
curl -X PUT https://your-domain.com/api/payments/methods/set-
default \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "customer_id": 123,
    "payment_method_id": 790
  }'
```

**Réponse** (200 OK) :


```
{
  "success": true,
  "message": "Méthode de paiement par défaut mise à jour",
  "data": {
    "payment_method_id": 790,
    "is_default": true
  }
}
```

## Supprimer une Méthode de Paiement

Supprimer une méthode de paiement sauvegardée.

**Point de terminaison** : `DELETE /api/payments/methods/{payment_method_id}?customer_id={id}`

**Demande** :

```
curl -X DELETE "https://your-domain.com/api/payments/methods/789?
customer_id=123" \
  -H "Authorization: Bearer YOUR_API_KEY"
```

**Réponse** (200 OK) :

```
{
  "success": true,
  "message": "Méthode de paiement supprimée avec succès"
}
```

---

# API des Flux de Paiement

Le système prend en charge plusieurs flux de paiement en fonction de votre cas d'utilisation.

## 1. Paiement Direct (Charge Simple)

**Cas d'utilisation** : Paiement simple en une étape sans provisionnement de service.

**Point de terminaison** : `POST /api/payments/charge`

**Demande** :

```
curl -X POST https://your-domain.com/api/payments/charge \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "customer_id": 123,
    "amount": 50.00,
    "currency": "USD",
    "payment_method_id": 789,
    "metadata": {
      "order_id": "12345"
    }
  }'
```

**Corps de la Demande** :

```
{
  "customer_id": 123,
  "amount": 50.00,
  "currency": "USD",                    // Par défaut : USD
  "payment_method_id": 789,             // Optionnel - utilise
par défaut si omis
  "vendor": "stripe",                   // Requis si utilisation
de payment_token
  "payment_token": "pm_xxxxx",          // Optionnel - jeton
unique
  "save_method": false,                 // Sauvegarder la méthode
de paiement pour une utilisation future ?
  "metadata": {
    "order_id": "12345"
  }
}
```

**Réponse** (200 OK) :

```
{
  "success": true,
  "message": "Paiement réussi",
  "data": {
    "transaction_id": "ch_xxxxx",
    "capture_id": 101,
    "amount": 50.00,
    "currency": "USD",
    "status": "succeeded"
  }
}
```

## 2. Paiement de Facture (Routage Prioritaire au Portefeuille)

**Cas d'utilisation** : Payer une facture en utilisant d'abord le solde du portefeuille, puis la carte pour le reste.

**Point de terminaison** : `POST /api/payments/invoice`

**Flux** :

1. Vérifier le solde du portefeuille
2. Utiliser d'abord les fonds du portefeuille
3. Facturer la carte pour le manque (si nécessaire)
4. Créditer le portefeuille avec le montant de la carte
5. Débiter le portefeuille pour le montant total de la facture

**Demande** :

```
curl -X POST https://your-domain.com/api/payments/invoice \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "customer_id": 123,
    "amount": 200.00,
    "payment_method_id": 789,
    "metadata": {
      "invoice_id": 12345,
      "description": "Paiement de facture"
    }
  }'
```

**Réponse** (200 OK) :

```
{
  "success": true,
  "message": "Paiement de facture traité",
  "data": {
    "customer_id": 123,
    "service_amount": 200.00,
    "routing_mode": "hybride",
    "initial_balance": 150.00,
    "wallet_portion_used": 150.00,       // Le portefeuille a
couvert cette somme
    "card_portion_used": 50.00,          // Carte facturée pour le
reste
    "charged_amount": 50.00,
    "wallet_credited": 50.00,
    "wallet_debited": 200.00,
    "final_balance": 0.00,
    "payment_method_used": true
  }
}
```

## 3. Autorisation de Retenue (Réserver des Fonds)

**Cas d'utilisation** : Retenir des fonds pour une capture ultérieure (par exemple, réservations d'hôtel, locations).

> **Implémentation du Playbook** : Pour des exemples de playbook Ansible de flux de paiement en deux phases, voir Modèle de Commit en Deux Phases

**Étape 1 : Créer une Autorisation de Retenue**

**Point de terminaison** : `POST /api/payments/authorize/hold`

**Flux Prioritaire au Portefeuille** :

1. Vérifier le solde du portefeuille
2. Calculer le manque (montant - wallet_balance)
3. Autoriser la carte uniquement pour le manque
4. Le débit du portefeuille se produit au moment de la capture, pas ici

**Demande** :

```
curl -X POST https://your-domain.com/api/payments/authorize/hold \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "customer_id": 123,
    "amount": 200.00,
    "payment_method_id": 789,
    "use_wallet": true,
    "metadata": {
      "reservation_id": "RES-001"
    }
  }'
```

**Corps de la Demande** :

```
{
  "customer_id": 123,
  "amount": 200.00,
  "currency": "USD",
  "payment_method_id": 789,              // Optionnel - utilise
par défaut si omis
  "vendor": "stripe",                    // Requis si utilisation
de payment_token
  "payment_token": "pm_xxxxx",           // Optionnel - jeton
unique
  "save_method": false,
  "use_wallet": true,                    // Activer le routage
prioritaire au portefeuille (par défaut : true)
  "metadata": {
    "reservation_id": "RES-001"
  }
}
```

**Réponse** (200 OK) :

```
{
  "success": true,
  "message": "Paiement autorisé (retenue créée)",
  "data": {
    "authorization_id": 301,
    "vendor_authorization_id": "auth_xxxxx",
    "amount": 200.00,
    "currency": "USD",
    "status": "authorized",
    "wallet_balance": 150.00,
    "wallet_to_use": 150.00,             // Le portefeuille
couvrira cette somme
    "card_amount": 50.00,                // Carte autorisée pour
cette somme
    "message": "Carte autorisée pour 50 $ (recharge de
portefeuille). Le débit du portefeuille de 200 $ se produira lors
de la capture."
  }
}
```

**Étape 2 : Capturer l'Autorisation**

**Point de terminaison** : `POST /api/payments/capture/{authorization_id}`

**Flux de Capture Prioritaire au Portefeuille** :

1. Capturer la carte (si autorisée) - recharge le portefeuille
2. Créditer le portefeuille avec le montant capturé de la carte
3. Débiter le portefeuille pour le montant total du service
4. Créer des factures/transactions (si demandé)

**Demande** :

```
curl -X POST https://your-domain.com/api/payments/capture/301 \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "amount": 200.00,
    "metadata": {
      "invoice": true,
      "title": "Réservation d'Hôtel",
      "description": "Séjour de 3 nuits",
      "wholesale_cost": 100.00,
      "contract_days": 3,
      "send_email": true
    }
  }'
```

**Corps de la Demande** :

```
{
  "amount": 200.00,                    // Optionnel - capture le
montant total si omis
  "metadata": {
    "invoice": true,                   // Créer une facture et
des transactions ?
    "create_transaction": true,        // Créer un
enregistrement de transaction ?
    "title": "Réservation d'Hôtel",
    "description": "Séjour de 3 nuits",
    "wholesale_cost": 100.00,
    "contract_days": 3,
    "send_email": true                 // Envoyer un e-mail de
facture ?
  }
}
```

**Réponse** (200 OK) :

```
{
  "success": true,
  "message": "Autorisation capturée",
  "data": {
    "capture_id": 103,
    "transaction_id": "ch_xxxxx",
    "authorization_id": 301,
    "amount": 200.00,
    "currency": "USD",
    "status": "succeeded",
    "wallet_credit": {                  // Carte a rechargé le
portefeuille
      "ledger_id": 401,
      "amount": 50.00
    },
    "wallet_debit": {                   // Service facturé depuis
le portefeuille
      "ledger_id": 402,
      "amount": 200.00
    },
    "transaction": {                    // Créé si invoice=true
      "transaction_id": 7001
    },
    "invoice": {                        // Créé si invoice=true
      "invoice_id": 12345,
      "invoice_number": "INV-2025-000001"
    }
  }
}
```

**Étape 3 : Libérer l'Autorisation (Annuler)**

**Point de terminaison** : `POST /api/payments/release/{authorization_id}`

**Cas d'utilisation** : Annuler la réservation, provision échouée, ou changement d'avis du client.

**Demande** :

```
curl -X POST https://your-domain.com/api/payments/release/301 \
  -H "Authorization: Bearer YOUR_API_KEY"
```

**Réponse** (200 OK) :

```
{
  "success": true,
  "message": "Autorisation libérée",
  "data": {
    "authorization_id": 301,
    "vendor_authorization_id": "auth_xxxxx",
    "status": "released",
    "released_at": "2025-12-27T10:45:00Z"
  }
}
```

**Remarque** : Avec le flux prioritaire au portefeuille, aucun remboursement de portefeuille n'est nécessaire puisque le portefeuille n'est pas débité avant le moment de la capture.

## 4. Paiement de Recharge (Deux Phases avec Provisionnement)

**Cas d'utilisation** : Traiter un paiement pour une recharge de service nécessitant un provisionnement (par exemple, activation de point d'accès/dongle). Si le provisionnement échoue, l'autorisation est libérée.

**Point de terminaison** : `POST /api/payments/topup`

**Flux** : Autoriser → Provisionner le service → Capturer

**Demande** :

```
curl -X POST https://your-domain.com/api/payments/topup \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "customer_id": 123,
    "amount": 30.00,
    "payment_method_id": 789,
    "service_uuid": "svc-uuid-123",
    "imsi": "123456789012345",
    "days": 30,
    "metadata": {
      "is_rental": false
    }
  }'
```

**Pour un Paiement Anonyme de Location/Point d'Accès** (méthode de paiement NON sauvegardée) :

```
curl -X POST https://your-domain.com/api/payments/topup \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "customer_id": 1,
    "amount": 5.00,
    "vendor": "stripe",
    "payment_token": "pm_xxxxx",
    "service_uuid": "hotspot-uuid",
    "imsi": "123456789012345",
    "days": 1,
    "metadata": {
      "is_rental": true,
      "billing_email": "user@example.com"
    }
  }'
```

**Réponse** (200 OK) :

```
{
  "success": true,
  "message": "Paiement de recharge traité avec succès",
  "data": {
    "transaction_id": "ch_xxxxx",
    "authorization_id": 302,
    "capture_id": 104,
    "amount": 30.00,
    "status": "succeeded",
    "provision_result": {
      "success": true,
      "topup_result": {...},
      "service_uuid": "svc-uuid-123",
      "imsi": "123456789012345",
      "days": 30
    },
    "payment_method_saved": false        // Faux pour
anonyme/location
  }
}
```

## 5. Paiement de Location (Tiers)

**Cas d'utilisation** : Facturer la carte d'un client pour payer le service d'un autre client.

**Point de terminaison** : `POST /api/payments/rental`

**Demande** :

```
curl -X POST https://your-domain.com/api/payments/rental \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "beneficiary_customer_id": 456,
    "charge_customer_id": 123,
    "amount": 75.00,
    "payment_method_id": 789,
    "service_description": "Paiement de service de location",
    "metadata": {
      "rental_agreement_id": "RA-001"
    }
  }'
```

**Pour un Paiement de Location Anonyme** (carte du locataire NON sauvegardée) :

```
curl -X POST https://your-domain.com/api/payments/rental \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "beneficiary_customer_id": 456,
    "amount": 75.00,
    "vendor": "stripe",
    "payment_token": "pm_xxxxx",
    "service_description": "Paiement de location anonyme",
    "metadata": {
      "billing_email": "renter@example.com"
    }
  }'
```

**Réponse** (200 OK) :

```
{
  "success": true,
  "message": "Paiement de location traité",
  "data": {
    "transaction_id": "ch_xxxxx",
    "amount": 75.00,
    "payment_method_saved": false        // Anonyme : aucune méthode sauvegardée
  }
}
```

## 6. Rembourser un Paiement

**Point de terminaison** : `POST /api/payments/refund`

**Demande** :

```
curl -X POST https://your-domain.com/api/payments/refund \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "transaction_id": "ch_xxxxx",
    "vendor": "stripe",
    "amount": 50.00,
    "reason": "demande_client"
  }'
```

**Corps de la Demande** :

```
{
  "transaction_id": "ch_xxxxx",          // ID de transaction du fournisseur
  "vendor": "stripe",                    // 'stripe', 'paypal', etc.
  "amount": 50.00,                       // Optionnel - remboursement total si omis
  "reason": "demande_client"             // Raison de remboursement optionnelle
}
```

**Réponse** (200 OK) :

```
{
  "success": true,
  "message": "Remboursement traité",
  "data": {
    "refund_id": "re_xxxxx",
    "amount": 50.00,
    "status": "succeeded"
  }
}
```

---

# API de Portefeuille

Tous les points de terminaison de portefeuille utilisent l'URL de base : `/api/wallet/`

## Obtenir le Solde du Portefeuille

**Point de terminaison** : `GET /api/wallet/balance?customer_id={id}`

**Demande** :

```
curl -X GET "https://your-domain.com/api/wallet/balance?
customer_id=123" \
  -H "Authorization: Bearer YOUR_API_KEY"
```

**Réponse** (200 OK) :

```
{
  "customer_id": 123,
  "balance": 150.50,
  "currency": "USD"
}
```

## Obtenir les Informations sur le Portefeuille

Obtenir des informations complètes sur le portefeuille et le crédit, y compris les paramètres de recharge automatique.

**Point de terminaison** : `GET /api/wallet/info?customer_id={id}`

**Demande** :

```
curl -X GET "https://your-domain.com/api/wallet/info?
customer_id=123" \
  -H "Authorization: Bearer YOUR_API_KEY"
```

**Réponse** (200 OK) :

```
{
  "customer_id": 123,
  "wallet": {
    "wallet_account_id": 456,
    "balance": 150.50,
    "currency": "USD",
    "auto_recharge_enabled": true,
    "auto_recharge_amount": 100.00,
    "auto_recharge_threshold": 10.00,
    "low_balance_warning_threshold": 10.00
  }
}
```

## Recharge du Portefeuille

Recharger le portefeuille en facturant une méthode de paiement.

**Point de terminaison** : `POST /api/wallet/topup`

**Flux** : Facturer la carte/PayPal → Créditer le portefeuille

**Demande** :

```
curl -X POST https://your-domain.com/api/wallet/topup \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "customer_id": 123,
    "amount": 100.00,
    "payment_method_id": 789
  }'
```

**Corps de la Demande** :

```
{
  "customer_id": 123,
  "amount": 100.00,
  "currency": "USD",                    // Par défaut : USD
  "payment_method_id": 789              // Optionnel - utilise
par défaut si omis
}
```

**Réponse** (200 OK) :

```
{
  "success": true,
  "payment": {
    "transaction_id": "ch_xxxxx",
    "amount": 100.00
  },
  "wallet": {
    "ledger_id": 503,
    "balance_after": 250.50,
    "customer_transaction_id": 7001
  },
  "message": "Portefeuille rechargé avec 100.00 USD"
}
```

Crée une **ligne de crédit de facturation** (`customer_transaction` avec `payment_type=wallet_topup_card`, coût de vente négatif, `wallet_ledger_id` défini). Cela est distinct du provisionnement hybride, où les manques de carte

créditent le grand livre du portefeuille comme `card_capture_topup` sans ligne de facturation de recharge.

**Annulation des recharges de fonds** : `DELETE /crm/transaction/transaction_id/{transaction_id}` débite le portefeuille et rembourse la carte. Voir Annuler la Transaction ci-dessous.

## Obtenir les Transactions du Portefeuille

Obtenir l'historique des transactions du portefeuille (grand livre).

**Point de terminaison** : `GET /api/wallet/transactions?customer_id={id}&limit={n}&offset={n}&type={type}`

**Demande** :

```
curl -X GET "https://your-domain.com/api/wallet/transactions?
customer_id=123&limit=50&offset=0&type=credit" \
  -H "Authorization: Bearer YOUR_API_KEY"
```

**Paramètres de Requête** :

- `customer_id` (requis) : ID du client
- `limit` (optionnel) : Nombre d'enregistrements (par défaut : 50)
- `offset` (optionnel) : Décalage de pagination (par défaut : 0)
- `type` (optionnel) : Filtrer par type de transaction ('credit', 'debit', 'refund', 'adjustment')

**Réponse** (200 OK) :

```
{
  "customer_id": 123,
  "count": 2,
  "transactions": [
    {
      "ledger_id": 501,
      "transaction_type": "credit",
      "amount": 100.00,
      "balance_before": 150.50,
      "balance_after": 250.50,
      "description": "Recharge de carte",
      "reference_type": "payment_capture",
      "reference_id": 103,
      "created_at": "2025-12-27T10:35:00Z"
    },
    {
      "ledger_id": 502,
      "transaction_type": "debit",
      "amount": 50.00,
      "balance_before": 250.50,
      "balance_after": 200.50,
      "description": "Frais de Service",
      "reference_type": "service_charge",
      "reference_id": 789,
      "created_at": "2025-12-27T11:00:00Z"
    }
  ]
}
```


---

## Annuler une Transaction

Annuler en douceur une transaction client non facturée. Nécessite `delete_customer_transaction`.

**Point de terminaison** : `DELETE /crm/transaction/transaction_id/{transaction_id}`

**Demande** :

```
curl -X DELETE "https://your-
domain.com/api/transaction/transaction_id/7001" \
  -H "Authorization: Bearer YOUR_API_KEY"
```

**Succès** (200 OK) : charge manuelle simple (pas de lien de portefeuille) :

```
{
  "result": "Succès"
}
```

**Succès** (200 OK) : ligne liée au portefeuille (annulation uniquement) :

```
{
  "result": "Succès",
  "wallet_reversal": {
    "wallet_reversed": true,
    "wallet_amount": 50.00,
    "wallet_ledger_id": 503
  }
}
```

**Succès** (200 OK) : recharge de fonds explicite par carte (`payment_type=wallet_topup_card`) :

```
{
  "result": "Succès",
  "wallet_reversal": {
    "wallet_reversed": true,
    "wallet_amount": 100.00,
    "wallet_ledger_id": 503,
    "refund": {
      "capture_id": 88,
      "vendor": "stripe",
      "vendor_transaction_id": "ch_xxxxx",
      "refund_id": "re_xxxxx",
      "status": "succeeded",
      "amount": 100.00
    }
  }
}
```


**Politique d'Annulation / Remboursement**

| Source de Transaction | `wallet_ledger_id` | Portefeuille à annuler | Remboursement de carte à annule |
|---|---|---|---|
| Charge/crédit manuel | Non | Non | Non |
| Recharge de Fonds (`POST /wallet/topup`) | Oui | Débiter le portefeuille | **Oui** (`wallet_topup_card` |
| Crédit manuel de portefeuille | Oui | Débiter le portefeuille | Non |
| Facture payée depuis le portefeuille | Oui | Créditer le portefeuille | Non |
| Service / provisionnement (capture hybride) | Généralement **non** sur la charge CT | Non* | Non |

*Les flux hybrides routent carte → portefeuille → débit de service en interne (`card_capture_topup` + lignes de grand livre `service_charge`). La ligne de facturation visible est le frais de service, pas une ligne de recharge. Un retour plus riche pour ces charges est prévu sur la branche des documents financiers.

**Erreurs** (500) :

```
{
  "result": "Échoué",
  "Reason": "Solde de portefeuille insuffisant. Disponible : 0.00, Requis : 20.00"
}
```

- **Transaction facturée :** `"Impossible de supprimer une transaction associée à une facture"`

- **Solde de portefeuille insuffisant :** le client a déjà dépensé des fonds rechargés ; l'annulation est rejetée (pas d'annulation partielle)
- **Échec de remboursement de carte :** l'annulation du portefeuille est annulée ; la transaction reste non annulée

**Remarque :** L'annulation marque `void=true` (exclue de la facturation/statistiques). Ce n'est pas la même chose qu'un remboursement de facture.

---

# Résumé de la Référence API

## Points de terminaison des Méthodes de Paiement

| Méthode | Point de terminaison | Description |
|---|---|---|
| POST | `/api/payments/methods` | Ajouter une méthode de paiement |
| GET | `/api/payments/methods?customer_id={id}` | Obtenir toutes les méthodes de paiement |
| GET | `/api/payments/methods/default?customer_id={id}` | Obtenir la méthode de paiement par défaut |
| PUT | `/api/payments/methods/set-default` | Définir la méthode de paiement par défaut |
| DELETE | `/api/payments/methods/{id}?customer_id={id}` | Supprimer une méthode de paiement |

# Points de terminaison des Flux de Paiement

| Méthode | Point de terminaison | Description |
|---|---|---|
| POST | `/api/payments/charge` | Paiement direct (en une étape) |
| POST | `/api/payments/invoice` | Paiement de facture (priorité au portefeuille) |
| POST | `/api/payments/topup` | Recharge avec provisionnement |
| POST | `/api/payments/authorize/hold` | Créer une autorisation de retenue |
| POST | `/api/payments/capture/{id}` | Capturer l'autorisation |
| POST | `/api/payments/release/{id}` | Libérer l'autorisation |
| POST | `/api/payments/rental` | Paiement de location/tier |
| POST | `/api/payments/refund` | Rembourser un paiement |

## Points de terminaison du Portefeuille

| Méthode | Point de terminaison | Description |
|---|---|---|
| GET | `/api/wallet/balance?customer_id={id}` | Obtenir le solde du portefeuille |
| GET | `/api/wallet/info?customer_id={id}` | Obtenir les informations du portefeuille + paramètres |
| POST | `/api/wallet/topup` | Recharger le portefeuille via méthode de paiement (crée une ligne de facturation `wallet_topup_card`) |
| GET | `/api/wallet/transactions?customer_id={id}` | Obtenir l'historique des transactions |

# Points de terminaison des Transactions

| Méthode | Point de terminaison | Description |
|---|---|---|
| PUT | `/api/transaction/` | Créer une transaction client |
| GET | `/api/transaction/` | Lister les transactions (paginé) |
| DELETE | `/api/transaction/transaction_id/{id}` | Annuler une transaction ; annulation de portefeuille ; remboursement de carte uniquement pour les recharges de fonds |

## Points de terminaison Spécifiques à PayPal

| Méthode | Point de terminaison | Description |
|---|---|---|
| POST | `/api/payments/paypal/vault/setup-token` | Créer un jeton de configuration PayPal pour le SDK Card Fields |
| POST | `/api/payments/paypal/vault/finalize` | Finaliser le coffre PayPal et sauvegarder la méthode de paiement |
| POST | `/api/payments/paypal/vault/update-setup-token` | Mettre à jour le jeton de configuration pour la gestion 3DS/SCA |

---

# Cas d'utilisation courants

## Cas d'utilisation 1 : Facturer un Client pour un Service (Prépayé)

**Scénario** : Client payant pour 30 jours de service en utilisant le routage prioritaire au portefeuille.

```
# Paiement de facture (priorité au portefeuille, carte pour le
manque)
curl -X POST https://your-domain.com/api/payments/invoice \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "customer_id": 123,
    "amount": 99.99,
    "metadata": {
      "service_id": 456,
      "contract_days": 30,
      "description": "Service Premium de 30 jours"
    }
  }'
```

**Réponse** :

```
{
  "success": true,
  "message": "Paiement de facture traité",
  "data": {
    "service_amount": 99.99,
    "wallet_portion_used": 50.00,        // Le portefeuille avait
50 $
    "card_portion_used": 49.99,          // Carte facturée 49,99 $
    "final_balance": 0.00
  }
}
```

## Cas d'utilisation 2 : Paiement en Deux Phases avec Provisionnement de Service

**Scénario** : Provisionner un service uniquement après que le paiement soit autorisé. Si le provisionnement échoue, aucun frais n'est appliqué.

```
# Paiement de recharge avec provisionnement automatique
curl -X POST https://your-domain.com/api/payments/topup \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "customer_id": 123,
    "amount": 149.99,
    "payment_method_id": 789,
    "service_uuid": "fiber-svc-uuid",
    "imsi": "123456789012345",
    "days": 30,
    "metadata": {
      "service_type": "fiber_internet"
    }
  }'
```

**Flux** :

1. Autoriser 149,99 $ sur la carte
2. Provisionner le service Internet par fibre
3. Si le provisionnement réussit → capturer le paiement
4. Si le provisionnement échoue → libérer l'autorisation (aucun frais)

## Cas d'utilisation 3 : Paiement Anonyme pour Point d'Accès

**Scénario** : Utilisateur anonyme paie pour l'accès WiFi. Aucun enregistrement client, aucune méthode de paiement sauvegardée.

```
# Paiement anonyme pour point d'accès
curl -X POST https://your-domain.com/api/payments/topup \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "customer_id": 1,
    "amount": 5.00,
    "vendor": "stripe",
    "payment_token": "pm_xxxxx",
    "service_uuid": "hotspot-downtown",
    "imsi": "999999999999999",
    "days": 1,
    "metadata": {
      "is_rental": true,
      "billing_email": "user@example.com",
      "hotspot_location": "Caf�? Downtown"
    }
  }'
```

**Résultat** :

- Paiement traité □
- Point d'accès activé □
- Méthode de paiement NON sauvegardée □
- Aucun enregistrement client créé □

## Cas d'utilisation 4 : Réservation d'Hôtel (Autorisation de Retenue + Capture)

**Scénario** : Retenir des fonds pour une réservation d'hôtel, capturer à l'enregistrement, libérer si annulé.

```
# Étape 1 : Enregistrement - retenir des fonds
curl -X POST https://your-domain.com/api/payments/authorize/hold \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "customer_id": 123,
    "amount": 500.00,
    "use_wallet": true,
    "metadata": {
      "reservation_id": "RES-2025-001"
    }
  }'

# Réponse : {"authorization_id": 301, "status": "authorized"}

# Étape 2a : Le client s'enregistre - capturer
curl -X POST https://your-domain.com/api/payments/capture/301 \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "metadata": {
      "invoice": true,
      "title": "Séjour à l'Hôtel - 3 Nuits",
      "description": "Chambre 302, 27-30 Déc",
      "send_email": true
    }
  }'

# Étape 2b : Le client annule - libérer la retenue
curl -X POST https://your-domain.com/api/payments/release/301 \
  -H "Authorization: Bearer YOUR_API_KEY"
```

## Cas d'utilisation 5 : Ajouter une Méthode de Paiement (Stripe)

**Scénario** : Un client ajoute une nouvelle carte Visa à son compte.

```
# Étape 1 : Le frontend crée un jeton Stripe via Stripe.js
# const {token} = await stripe.createToken(cardElement);

# Étape 2 : Le backend sauvegarde la méthode de paiement
curl -X POST https://your-domain.com/api/payments/methods \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "customer_id": 123,
    "vendor": "stripe",
    "payment_token": "pm_xxxxx",
    "is_default": true,
    "card_nickname": "Visa de Travail"
  }'
```

## Cas d'utilisation 6 : Recharge du Portefeuille

**Scénario** : Un client recharge son portefeuille de 100 $ en utilisant une méthode de paiement sauvegardée.

```
curl -X POST https://your-domain.com/api/wallet/topup \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "customer_id": 123,
    "amount": 100.00
  }'
```

**Réponse** :

```
{
  "success": true,
  "payment": {
    "transaction_id": "ch_xxxxx",
    "amount": 100.00
  },
  "wallet": {
    "balance_after": 250.50
  },
  "message": "Portefeuille rechargé avec 100.00 USD"
}
```


---

# Gestion des Erreurs

## Réponses d'Erreur Courantes

**Fonds Insuffisants** (400 Bad Request):


```
{
  "error": "Solde de portefeuille insuffisant. Disponible : 50.00,
Requis : 200.00"
}
```


**Échec du Paiement** (400 Bad Request):


```
{
  "error": "Le paiement a échoué. Veuillez réessayer."
}
```


**Erreur de Validation** (400 Bad Request):


```
{
  "error": "customer_id et amount sont requis"
}
```

**Non Trouvé** (404 Not Found):

```
{
  "error": "Autorisation 999 non trouvée"
}
```

**Erreur Serveur** (500 Internal Server Error):

```
{
  "error": "Une erreur s'est produite lors du traitement de votre paiement"
}
```

---

# Remboursements et Routage Prioritaire au Portefeuille

## Options de Remboursement

Le système prend en charge **deux types de remboursements** en fonction des besoins de votre entreprise :

**1. Remboursement à la Source de Paiement (Stripe/PayPal)**

**Cas d'utilisation** : Le client demande un remboursement total pour un service annulé ou un produit défectueux.

**Flux** : L'argent retourne à la méthode de paiement d'origine (carte, compte PayPal, etc.)

**Point de terminaison** : `POST /api/payments/refund`

**Exemple** :

```
curl -X POST https://your-domain.com/api/payments/refund \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "transaction_id": "ch_xxxxx",
    "vendor": "stripe",
    "amount": 100.00,
    "reason": "demande_client"
  }'
```

**Résultat** :

- Stripe/PayPal traite le remboursement à la méthode de paiement d'origine
- Le client voit le crédit sur sa carte/compte PayPal dans les 5 à 10 jours ouvrables
- L'argent ne va PAS au portefeuille
- Piste d'audit complète maintenue dans la table `PaymentCapture`

**Quand Utiliser** :

- □ Le client a annulé la commande
- □ Service non livré
- □ Erreur de facturation
- □ Le client demande explicitement un remboursement sur la carte

**2. Crédit au Portefeuille**

**Cas d'utilisation** : Remboursement partiel, crédit de service, ou garder des fonds pour des achats futurs.

**Flux** : L'argent est crédité au solde du portefeuille du client pour une utilisation immédiate.

**Remarque** : Les crédits de portefeuille sont généralement gérés en interne par le système lors de scénarios d'erreur. Pour des crédits de portefeuille manuels, contactez le support ou utilisez les outils d'administration.

**Résultat** :

- Fonds immédiatement disponibles dans le portefeuille
- Pas de période d'attente
- Peut être utilisé pour des achats futurs
- Piste d'audit complète dans la table `WalletLedger`

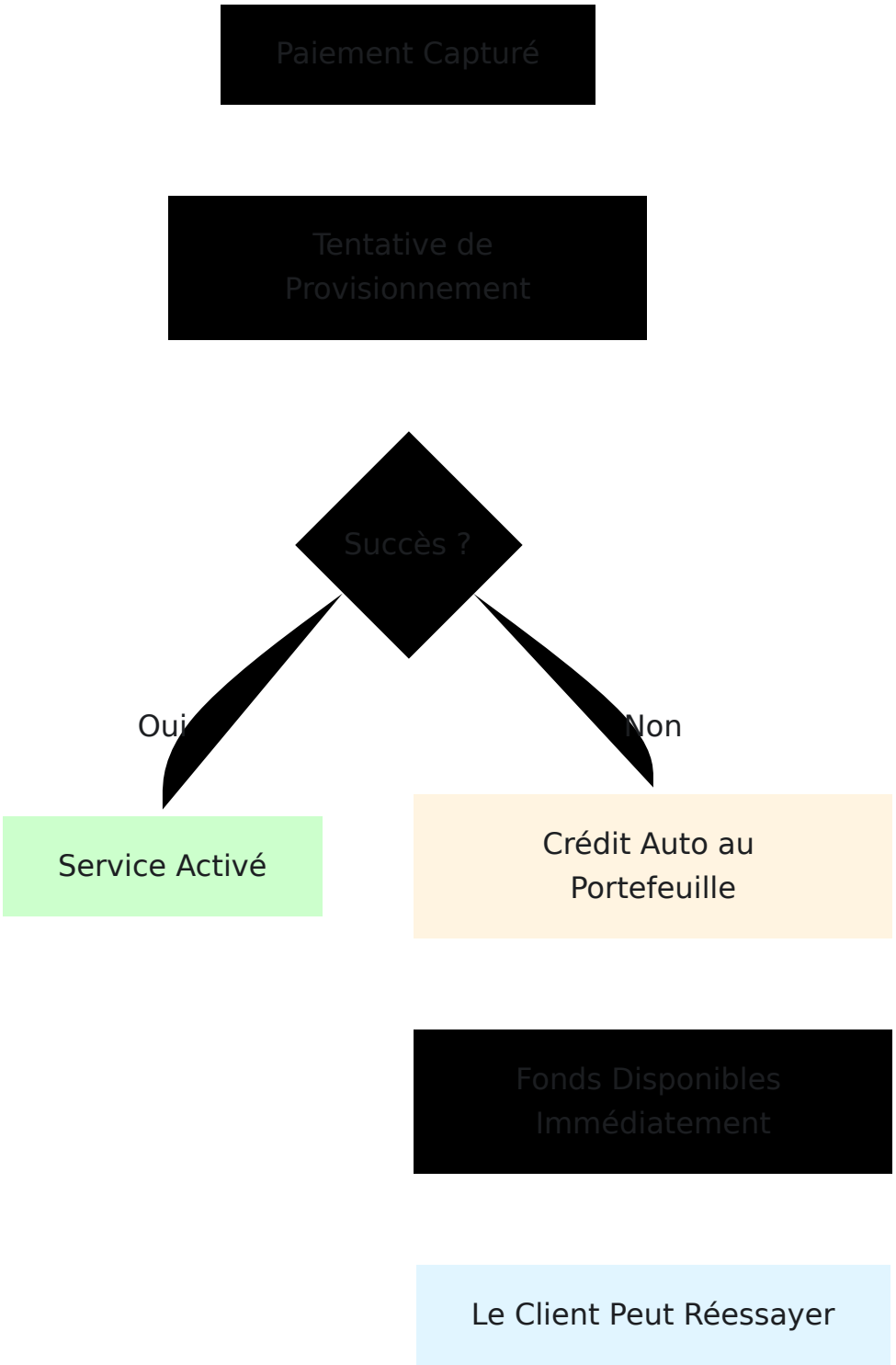
**Quand Utiliser** :

- □ Crédits de service (par exemple, compensation pour un temps d'arrêt)
- □ Remboursements partiels où le client va racheter
- □ Compensation d'erreur (provisionnement échoué)
- □ Crédits promotionnels

## Stratégie de Remboursement Hybride

**Meilleure Pratique** : Pour les scénarios d'erreur pendant le flux de paiement, le système crédite automatiquement le portefeuille au lieu de rembourser la carte.

**Récupération Automatique d'Erreur** :




**Scénario d'Exemple** :

1. Le client paie 100 $ pour un service
2. La carte est facturée avec succès
3. Le provisionnement échoue
4. Au lieu de rembourser 100 $ à la carte (5-10 jours + frais de remboursement) :

- Créditer 100 $ au portefeuille immédiatement
- Le client peut réessayer l'achat instantanément
- Pas de frais de remboursement
- Meilleure expérience utilisateur

## Routage Prioritaire au Portefeuille : Charges de Carte Optimisées

Le système **ne facture votre carte que pour le manque**, pas le montant total, lorsque vous avez un solde de portefeuille.

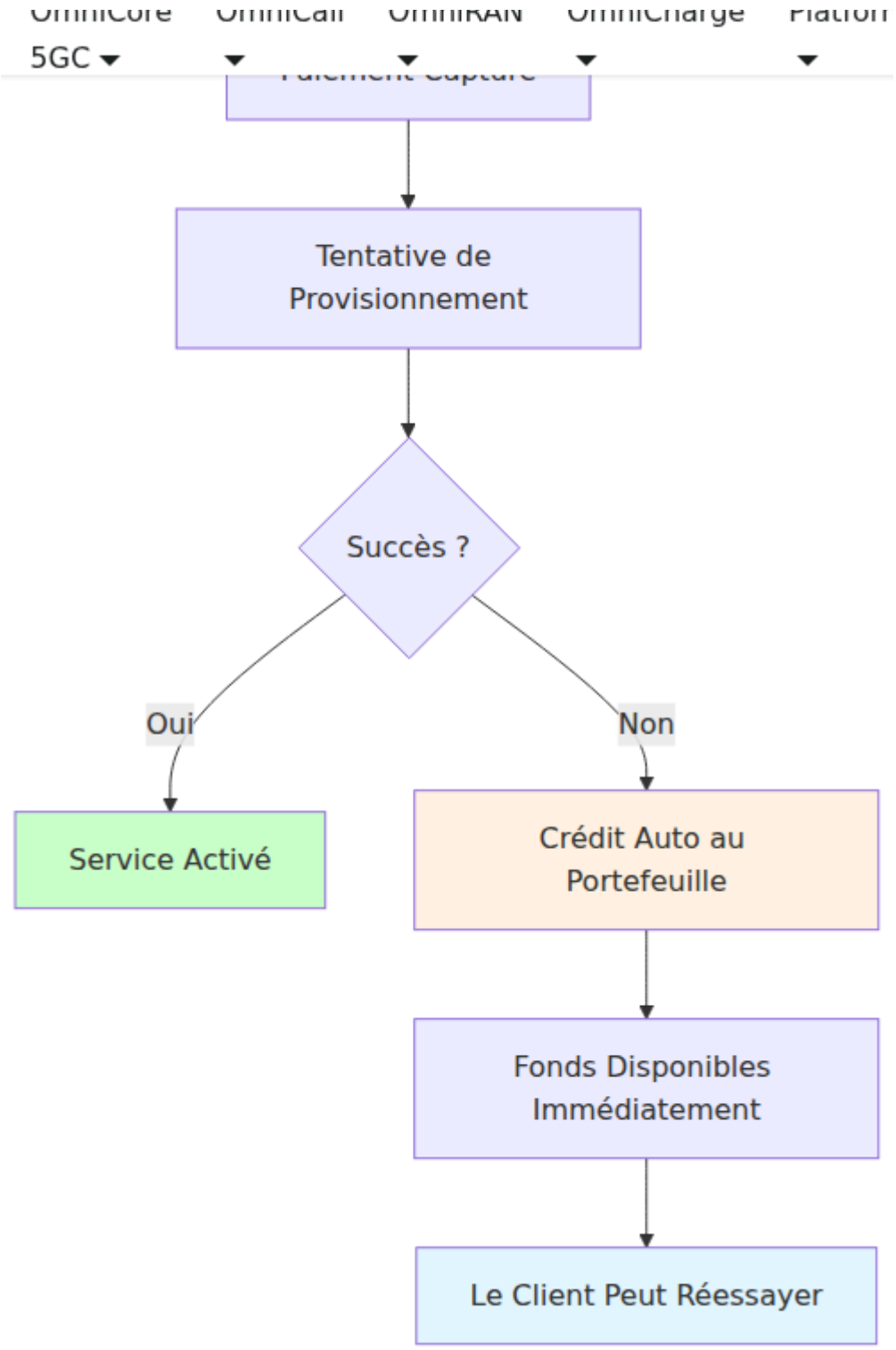
**Exemple 1 : Solde de 1 $ au Portefeuille + Achat de 10 $**

**Scénario** : Vous avez 1 $ dans votre portefeuille et souhaitez acheter un addon de 10 $.

**Flux de Paiement Traditionnel** (PAS comment ce système fonctionne) :




**Flux Prioritaire au Portefeuille OmniCRM** :


5GC
Tentative de Provisionnement
Succès ?
Oui
Non
Service Activé
Crédit Auto au Portefeuille
Fonds Disponibles Immédiatement
Le Client Peut Réessayer

**Demande API** :

```
curl -X POST https://your-domain.com/api/payments/invoice \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "customer_id": 123,
    "amount": 10.00,
    "metadata": {
      "addon_id": 456
    }
  }'
```

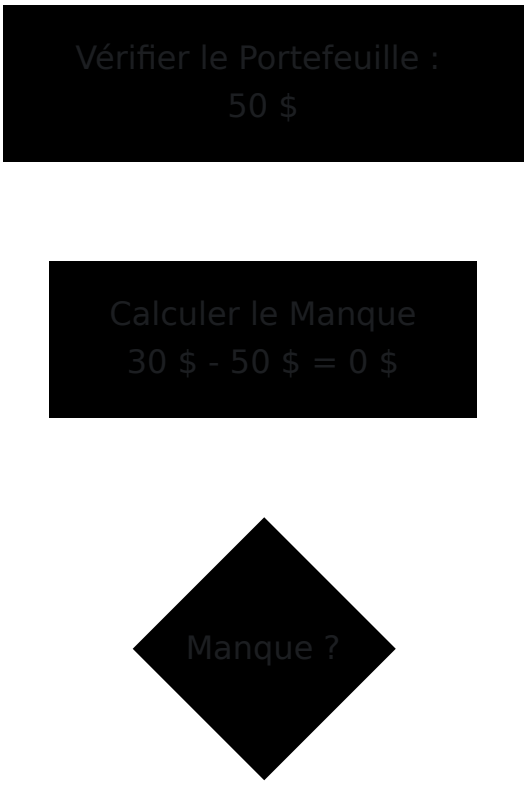
**Réponse API** :

```
{
  "success": true,
  "message": "Paiement de facture traité",
  "data": {
    "customer_id": 123,
    "service_amount": 10.00,
    "routing_mode": "hybride",
    "initial_balance": 1.00,
    "wallet_portion_used": 1.00,     // Utilisé 1 $ existant
    "card_portion_used": 9.00,       // Seulement 9 $ de manque
facturé
    "charged_amount": 9.00,          // ← Charge de carte
    "wallet_credited": 9.00,         // ← Portefeuille rechargé
    "wallet_debited": 10.00,         // ← Frais de service
    "final_balance": 0.00
  }
}
```

**Exemple 2 : Solde de 50 $ au Portefeuille + Achat de 30 $**

**Scénario** : Vous avez 50 $ dans le portefeuille et le coût d'achat est de 30 $.

**Flux Prioritaire au Portefeuille** :


Vérifier le Portefeuille :
50 $
Calculer le Manque
30 $ - 50 $ = 0 $
Manque ?
Pas de manque
Ignorer la Charge de
Carte
Débiter le Portefeuille :
-30 $
Solde Final : 20 $
Carte Facturée : 0 $
Utilisé : 30 $ □

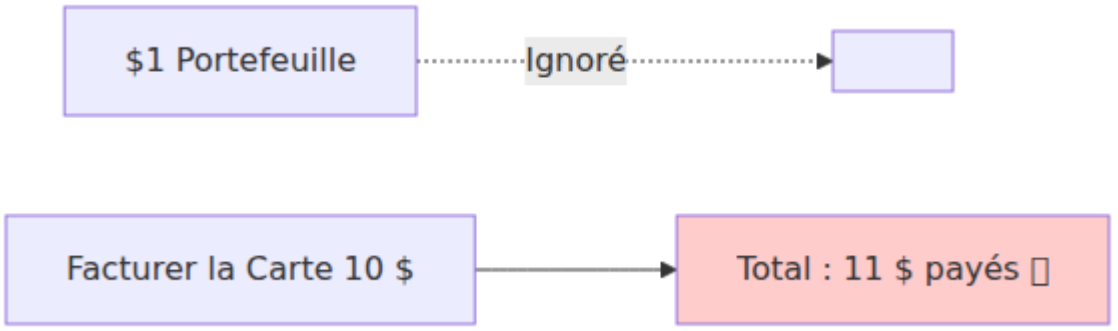
**Réponse API** :

```
{
  "success": true,
  "data": {
    "service_amount": 30.00,
    "initial_balance": 50.00,
    "charged_amount": 0,              // ← Aucune charge de carte
    "wallet_debited": 30.00,
    "final_balance": 20.00,
    "payment_method_used": false      // ← Carte non utilisée
  }
}
```

**Exemple 3 : Autorisation de Retenue avec Solde de Portefeuille**

**Scénario** : Hôtel retient 500 $ d'autorisation avec 150 $ de solde de portefeuille.

**Étape 1 : Autorisation** (`POST /api/payments/authorize/hold`) :




**Réponse API** :

```
{
  "authorization_id": 301,
  "amount": 500.00,
  "status": "authorized",
  "wallet_balance": 150.00,
  "wallet_to_use": 150.00,
  "card_amount": 350.00,
  "message": "Carte autorisée pour 350 $ (recharge de
portefeuille). Le débit du portefeuille de 500 $ se produira lors
de la capture."
}
```

**Étape 2 : Capture** (`POST /api/payments/capture/301`) :

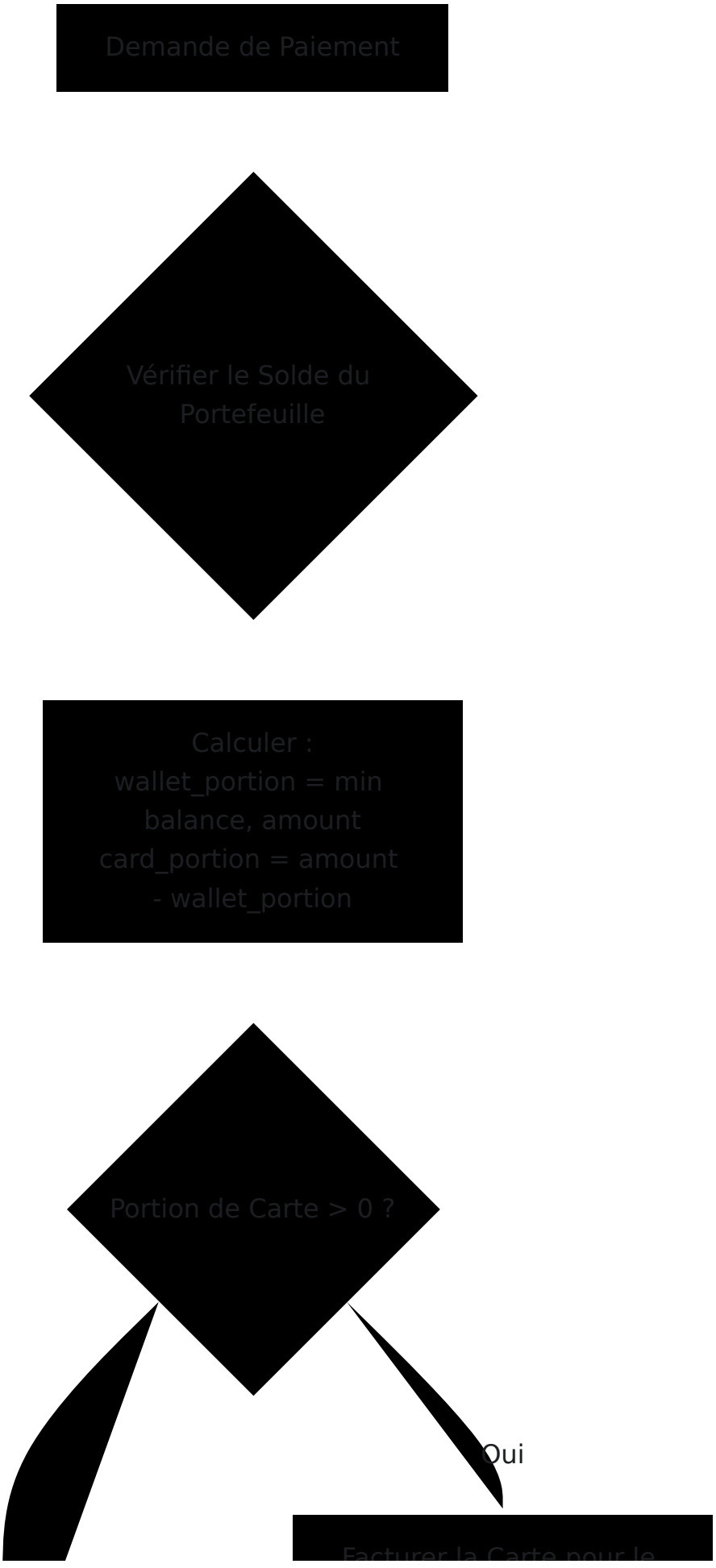


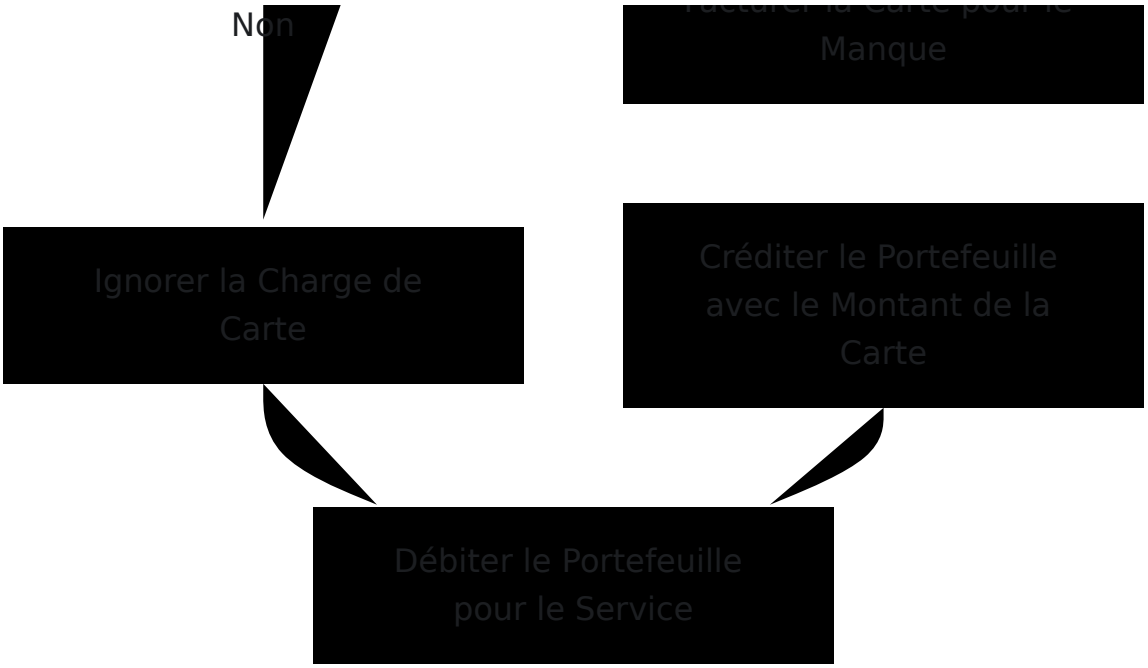

**Pourquoi Cela Compte** :

- □ Le client n'a que 350 $ "retenus" sur sa carte, pas 500 $
- □ Réduit l'impact sur le crédit disponible du client
- □ Montants d'autorisation plus précis
- □ Meilleure expérience client

# Détails de Mise en Œuvre

**Logique de Routage Prioritaire au Portefeuille** :


Demande de Paiement
Vérifier le Solde du Portefeuille
Calculer : wallet_portion = min balance, amount card_portion = amount - wallet_portion
Portion de Carte > 0 ?
Oui
Facturer la Carte pour le




**Comment Cela Fonctionne** :

1. Le système vérifie le solde actuel du portefeuille
2. Calcule le manque : `amount - wallet_balance`
3. Si le manque > 0, facture la carte uniquement pour le manque
4. Crédite le portefeuille avec le paiement de la carte
5. Débite le portefeuille pour le montant total du service
6. Résultat : Le client n'est facturé que pour ce que le portefeuille ne peut pas couvrir

# Mode de Contournement

Vous pouvez contourner le comportement prioritaire au portefeuille si nécessaire :

**Mode de Contournement** - Toujours facturer la carte même si le portefeuille a des fonds :

```
curl -X POST https://your-domain.com/api/payments/charge \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{
    "customer_id": 123,
    "amount": 100.00,
    "metadata": {
      "routing_mode": "bypass"
    }
  }'
```

**Résultat** : Carte facturée 100 $, portefeuille crédité 100 $, portefeuille débité 100 $ (net : 0 $)

**Cas d'utilisation** : Le client souhaite utiliser la carte pour des récompenses/points même avec un solde de portefeuille.

---

# Meilleures Pratiques

1. **Utilisez toujours le routage prioritaire au portefeuille** pour les paiements des clients afin d'activer la fonctionnalité prépayée
2. **Utilisez des paiements en deux phases** (point de terminaison de recharge) lors du provisionnement de services pour éviter de facturer si le provisionnement échoue
3. **Définissez des métadonnées** pour tous les paiements afin de maintenir une piste d'audit
4. **Utilisez des paiements anonymes** uniquement pour les utilisateurs vraiment anonymes (points d'accès, locations)
5. **Gérez les erreurs avec grâce** et fournissez des retours clairs aux utilisateurs
6. **Testez les flux de paiement** dans les scénarios de succès et d'échec
7. **Surveillez les expirations d'autorisation** (typiquement 7 jours pour les cartes)
8. **Implémentez la recharge automatique** pour les services récurrents afin d'améliorer l'expérience client

# Meilleures Pratiques Spécifiques aux Playbooks

Lors de l'implémentation de flux de paiement dans des playbooks Ansible :

1. **Utilisez toujours le modèle bloc/ressort** - Enveloppez le provisionnement dans un essai/attraper pour un retour automatique
2. **Stockez authorization_id** - Sauvegardez pour les opérations de capture/libération
3.

# Factures Client

> **Référence API** : Pour les API de paiement des factures et les flux de travail, voir le **Guide de l'API du Système de Paiement**

Transactions sont regroupées pour former une facture, qui est envoyée au client pour paiement.

Les factures ont une date de début et une date de fin, qui est la période que la facture couvre, et une date d'échéance, qui est la date à laquelle la facture doit être payée.

Les factures peuvent être générées automatiquement par le système, par exemple, lorsque un service est facturé, une facture est créée pour le coût de détail, ou elles peuvent être créées manuellement, par exemple, si un client demande une copie d'une facture, ou si un client est facturé pour un frais unique.

Les factures clients sont entièrement modélisées avec Mailjet et peuvent être personnalisées pour inclure le logo de l'entreprise, l'adresse et les détails de paiement, et peuvent être envoyées au client par e-mail, ou téléchargées au format PDF.

# Personnalisation des Modèles de Facture

OmniCRM utilise des modèles HTML avec Jinja2 pour générer des factures. Vous pouvez personnaliser entièrement le design de la facture, le branding, les couleurs et la mise en page.

## Emplacement des Modèles de Facture

Les modèles de facture sont stockés dans `OmniCRM-API/invoice_templates/`

**Modèles par Défaut :**

- `sample_invoice_template.html` - Modèle de facture d'exemple
- `cifi_invoice_template.html` - Exemple de modèle alternatif

**Configuration :**

Le modèle de facture actif est spécifié dans `OmniCRM-API/crm_config.yaml` :

```yaml
invoice:
  template_filename: 'sample_invoice_template.html'
```

# Variables de Modèle Disponibles

Les modèles de facture ont accès aux variables Jinja2 suivantes :

**Informations sur la Facture :**

- `{{ invoice_number }}` - ID de facture unique (ex. : `INV-2025-001234`)
- `{{ date }}` - Date d'émission de la facture (format ISO : `2025-01-10T12:00:00`)
- `{{ due_date }}` - Date d'échéance de paiement (ex. : `2025-02-10`)
- `{{ start_date }}` - Date de début de la période de facturation
- `{{ end_date }}` - Date de fin de la période de facturation
- `{{ total_amount }}` - Montant total de la facture avant taxes (numérique)
- `{{ total_tax }}` - Montant total des taxes calculé à partir de toutes les transactions (numérique)

**Informations sur le Client :**

- `{{ client.name }}` - Nom complet du client ou nom de l'entreprise
- `{{ client.address.address_line_1 }}` - Ligne d'adresse 1
- `{{ client.address.address_line_2 }}` - Ligne d'adresse 2
- `{{ client.address.city }}` - Ville
- `{{ client.address.state }}` - État/province
- `{{ client.address.zip_code }}` - Code postal
- `{{ client.address.country }}` - Pays

**Lignes de Transactions :**

Boucle à travers les transactions en utilisant :

```
{% for sub_transaction in transactions %}
    <tr>
        <td>{{ sub_transaction.transaction_id }}</td>
        <td>{{ sub_transaction.created.split("T")[0] }}</td>
        <td>{{ sub_transaction.title }}</td>
        <td>{{ sub_transaction.description }}</td>
        <td>${{ "%.2f"|format(sub_transaction.retail_cost) }}</td>
    </tr>
{% endfor %}
```

**Champs de Transaction :**

- `sub_transaction.transaction_id` - ID de la transaction
- `sub_transaction.created` - Date/heure de la transaction
- `sub_transaction.title` - Titre de la transaction
- `sub_transaction.description` - Description détaillée
- `sub_transaction.retail_cost` - Montant de l'article
- `sub_transaction.tax_percentage` - Pourcentage de taxe appliqué (ex. : 10 pour 10%)
- `sub_transaction.tax_amount` - Montant de la taxe calculé en dollars

**Affichage de la Taxe dans les Modèles :**

```
<td>
    {% if sub_transaction.tax_amount and
sub_transaction.tax_amount > 0 %}
        ${{ "%.2f"|format(sub_transaction.tax_amount) }} ({{
sub_transaction.tax_percentage }}%)
    {% else %}
        -
    {% endif %}
</td>
```

# Création d'un Modèle de Facture Personnalisé

**Étape 1 : Copier un Modèle Existant**

```
cd OmniCRM-API/invoice_templates/
cp sample_invoice_template.html your_company_invoice_template.html
```

**Étape 2 : Personnaliser HTML/CSS**

Modifiez `your_company_invoice_template.html` pour correspondre à votre branding :

**Zones de Personnalisation Clés :**

1. **Logo de l'Entreprise et Branding**

```
<!-- Remplacez par l'URL de votre logo -->
![Votre Entreprise](https://yourcompany.com/logo.png)

<!-- Mettez à jour le nom de l'entreprise -->
<h1>Nom de Votre Entreprise</h1>
```

2. **Schéma de Couleurs**

```
<style>
    /* Couleur de marque principale */
    .navbar {
        background: linear-gradient(to bottom right, #your-
color-1, #your-color-2);
    }

    /* En-têtes de tableau */
    .table thead th {
        background-color: #your-brand-color !important;
        color: white !important;
    }

    /* Boutons et liens */
    .btn-primary {
        background-color: #your-brand-color;
    }
</style>
```

3. **Informations sur l'Entreprise dans le Pied de Page**

```
<footer>
    <p>Nom de Votre Entreprise</p>
    <p>123 Rue des Affaires, Ville, Pays</p>
    <p>Téléphone : +1-555-123-4567 | Email :
billing@yourcompany.com</p>
    <p>ABN/ID Fiscal : 12345678900</p>
</footer>
```

4. **Instructions de Paiement**

```
<div class="payment-info">
    <h3>Méthodes de Paiement</h3>
    <p><strong>En Ligne :</strong> Payez à
https://yourcompany.com/pay</p>
    <p><strong>Virement Bancaire :</strong></p>
    <ul>
        <li>Nom du Compte : Your Company Ltd</li>
        <li>BSB : 123-456</li>
        <li>Numéro de Compte : 987654321</li>
        <li>Référence : {{ invoice_number }}</li>
    </ul>
</div>
```

5. **Conditions Générales de Vente**

```
<div class="terms">
    <h4>Conditions de Paiement</h4>
    <p>Le paiement est dû dans les 30 jours suivant la date de
la facture.</p>
    <p>Frais de retard : 2 % par mois sur les soldes en
souffrance.</p>
    <p>Pour les questions de facturation :
billing@yourcompany.com</p>
</div>
```

**Étape 3 : Mettre à Jour la Configuration**

Modifiez `OmniCRM-API/crm_config.yaml` :

```
invoice:
  template_filename: 'your_company_invoice_template.html'
```

**Étape 4 : Redémarrer l'API**

```
cd OmniCRM-API
sudo systemctl restart omnicrm-api
```

**Étape 5 : Tester la Génération de Factures**

1. Accédez à un client avec des transactions
2. Générez une facture de test
3. Téléchargez le PDF pour vérifier le formatage
4. Envoyez la facture par e-mail à vous-même pour tester la livraison par e-mail

# Personnalisation Avancée

**Contenu Conditionnel :**

Utilisez des conditionnels Jinja2 pour afficher/masquer du contenu :

```
{% if total_amount > 1000 %}
    <div class="high-value-notice">
        <p><strong>Remarque :</strong> Solde élevé - Plan de
paiement disponible sur demande.</p>
    </div>
{% endif %}

{% if client.address.country == "Australia" %}
    <p>GST Inclus : ${{ "%.2f"|format(total_amount * 0.10) }}</p>
{% endif %}
```

**Support Multilingue :**

Créez des modèles spécifiques à la langue :

```
invoice_template_en.html
invoice_template_es.html
invoice_template_fr.html
```

Configurez en fonction de la préférence linguistique du client.

**Calculs Personnalisés :**

```
<!-- Afficher le sous-total et le détail de la taxe -->
<tr>
    <td colspan="4" class="text-right"><strong>Sous-total :
</strong></td>
    <td>${{ "%.2f"|format(total_amount) }}</td>
</tr>
<tr>
    <td colspan="4" class="text-right"><strong>Taxe :</strong>
</td>
    <td>${{ "%.2f"|format(total_tax) }}</td>
</tr>
<tr>
    <td colspan="4" class="text-right"><strong>Total :</strong>
</td>
    <td>${{ "%.2f"|format(total_amount + total_tax) }}</td>
</tr>
```

**Remarque :** La variable `total_tax` est automatiquement calculée en sommant le `tax_amount` de toutes les transactions dans la facture. Chaque taxe de transaction est calculée en fonction de son champ `tax_percentage`, qui par défaut est le `tax_percentage` du produit ou 0 % si non spécifié.

**QR Code pour le Paiement :**

Générez un QR code pour le paiement mobile :

```
<div class="qr-payment">
    ![Scanner pour Payer](https://api.qrserver.com/v1/create-qr-code/?size=150x150&data={{ payment_url }})
    <p>Scannez avec votre téléphone pour payer instantanément</p>
</div>
```

## Meilleures Pratiques de Style PDF

OmniCRM utilise **WeasyPrint** pour convertir HTML en PDF. Suivez ces directives :

**CSS Supporté :**

- La plupart des propriétés CSS 2.1
- CSS3 limité (flexbox, certaines transformations)
- Polices Web via `@font-face`

**Non Supporté :**

- JavaScript
- CSS Grid (utilisez des tableaux à la place)
- Animations complexes
- Certaines propriétés CSS modernes

**Taille de Page et Marges :**

```
@page {
    size: A4;
    margin: 1cm;
}

body {
    font-family: Arial, sans-serif;
    font-size: 10pt;
}
```

**Style Spécifique à l'Impression :**

```
@media print {
    .no-print {
        display: none;
    }

    .page-break {
        page-break-after: always;
    }
}
```

**Mise en Page du Tableau :**

```
.table {
    table-layout: fixed;
    width: 100%;
}

.table th, .table td {
    word-wrap: break-word;
    padding: 4px;
}
```

**Intégration de Polices :**

Pour des polices personnalisées, utilisez des polices sécurisées pour le web ou intégrez :

```
@font-face {
    font-family: 'YourFont';
    src: url('https://yourcompany.com/fonts/yourfont.woff2')
format('woff2');
}

body {
    font-family: 'YourFont', Arial, sans-serif;
}
```

# Tester les Modèles de Facture

**Liste de Vérification de Test :**

1. **Inspection Visuelle :**
   - Le logo s'affiche correctement
   - Les couleurs correspondent aux directives de la marque
   - Le texte est lisible (pas trop petit)
   - Les tableaux sont correctement alignés
   - Toutes les sections sont présentes
2. **Exactitude des Données :**
   - Détails du client corrects
   - Montants des transactions totalisés correctement
   - Dates formatées correctement
   - Toutes les variables substituées correctement
3. **Qualité du PDF :**
   - Taille de fichier raisonnable (<5MB)
   - Images nettes et claires
   - Pas de texte coupé ou débordant
   - Sauts de page aux endroits appropriés
4. **Factures Multi-Page :**
   - En-têtes se répètent sur chaque page
   - Numéros de page affichés
   - Longs listes de transactions paginées correctement
5. **Livraison par E-mail :**
   - Le PDF est joint à l'e-mail
   - Taille de fichier sous la limite de Mailjet (15MB)
   - Rendu dans Gmail, Outlook, Apple Mail

**Commande de Test (Génération Manuelle) :**

Vous pouvez tester la génération de factures via l'API :

```
curl -X GET "http://localhost:5000/crm/invoice/{invoice_id}/pdf" \
     -H "Authorization: Bearer YOUR_TOKEN" \
     --output test_invoice.pdf
```

# Problèmes Courants de Modèle

**Variables non substituées :**

- **Cause :** Erreur de frappe dans le nom de la variable ou données manquantes
- **Correction :** Vérifiez l'orthographe exactement (sensible à la casse), vérifiez que les données existent dans la base de données

**Style PDF cassé :**

- **Cause :** Propriété CSS non supportée
- **Correction :** Utilisez des propriétés CSS 2.1, testez avec CSS compatible WeasyPrint

**Images ne s'affichent pas :**

- **Cause :** URLs relatives ou ressources externes bloquées
- **Correction :** Utilisez des URLs HTTPS absolues, assurez-vous que les images sont accessibles publiquement

**Tables débordant de la page :**

- **Cause :** Largeurs de colonnes fixes trop larges
- **Correction :** Utilisez des largeurs en pourcentage, `table-layout: fixed`

**Polices ne se rendant pas :**

- **Cause :** Police non intégrée ou indisponible
- **Correction :** Utilisez des polices sécurisées pour le web (Arial, Times New Roman, etc.) ou intégrez correctement des polices personnalisées

**Échec de génération de PDF :**

- **Cause :** Erreurs de syntaxe HTML ou plantage de WeasyPrint
- **Correction :** Validez le HTML, vérifiez les journaux de WeasyPrint, simplifiez les mises en page complexes

# Mise en Cache des PDF de Facture

Pour améliorer les performances et réduire la génération redondante de PDF, OmniCRM inclut un système de mise en cache des PDF de factures. Lorsqu'un PDF de facture est généré pour la première fois, il est mis en cache dans la base de données pour les demandes ultérieures.

**Comment Fonctionne la Mise en Cache des PDF :**

1. **Première Demande** - Lorsqu'un PDF de facture est demandé (téléchargement ou e-mail), le système :
   - Génère le PDF à partir du modèle de facture
   - Encode le PDF en Base64
   - Calcule un hachage SHA256 du contenu du PDF
   - Stocke dans la table `Invoice_PDF_Cache` avec :
     - Référence ID de facture
     - Données PDF (encodées en Base64)
     - Nom de fichier
     - Hachage de contenu (pour vérification d'intégrité)
     - Horodatage de création
2. **Demandes Suivantes** - Lorsque la même facture est demandée à nouveau :
   - Le système vérifie le PDF mis en cache par invoice_id
   - Si le cache existe et est valide, renvoie immédiatement le PDF mis en cache
   - Met à jour l'horodatage `last_accessed` pour suivre l'utilisation du cache
3. **Invalidation du Cache** - Les PDF mis en cache sont invalidés lorsque :
   - La facture est modifiée (transactions ajoutées/enlevées, détails changés)
   - Le modèle de facture est mis à jour

    - Un nettoyage manuel du cache est déclenché

**Avantages :**

- **Performance** - Livraison instantanée du PDF pour les demandes répétées (pas de délai de régénération)
- **Cohérence** - Même PDF pour tous les téléchargements d'une facture (sauf si la facture est modifiée)
- **Charge Serveur** - Réduit l'utilisation du CPU pour la génération de PDF
- **Expérience Utilisateur** - Un indicateur de chargement apparaît lors de la génération initiale, les demandes suivantes sont instantanées

**Gestion du Cache :**

Le Cache PDF de Facture est géré automatiquement par le système. Les anciennes ou inutilisées entrées de cache peuvent être purgées périodiquement en fonction de :

- Âge (par exemple, supprimer les entrées de cache de plus de 90 jours)
- Modèles d'accès (supprimer les entrées non accédées depuis 30 jours)
- Limites de stockage (mettre en œuvre des limites de taille de cache si nécessaire)

**Comportement de l'API :**

Lors du téléchargement d'une facture via l'API ou l'UI :

- Première demande : Affiche un indicateur de chargement pendant que le PDF se génère, puis le met en cache
- Demandes suivantes : Téléchargement immédiat depuis le cache
- Le hit/miss du cache est transparent pour l'utilisateur

**Important :** Lorsque vous mettez à jour votre modèle de facture, videz le cache pour vous assurer que les nouvelles factures utilisent le design mis à jour :

```
-- Vider tous les PDF de factures mis en cache (exécuter dans MySQL)
DELETE FROM Invoice_PDF_Cache;
```

Ou mettez à jour `crm_config.yaml` pour invalider automatiquement le cache lors du changement de modèle.

# Accéder aux Factures

Les factures peuvent être consultées au niveau du système ou par client :

**Vue par Client :**

1. Accédez à **Clients → [Sélectionner le Client]**
2. Cliquez sur l'onglet **Facturation**
3. Consultez la liste des factures dans la troisième carte

**Vue Système :**

1. Accédez à **Facturation → Factures** (depuis le menu principal)
2. Consultez toutes les factures de tous les clients

# Widgets de Statistiques de Factures

En haut de la page des factures, quatre cartes statistiques affichent des résumés financiers.

**Descriptions des Widgets :**

- **Total des Factures** - Somme de tous les coûts de détail des factures (tous les temps) et nombre de factures envoyées
- **Factures Impayées** - Somme des factures non encore payées et nombre de factures impayées
- **Factures Ce Mois** - Somme des factures créées ce mois calendaire avec le nombre
- **Factures Le Mois Dernier** - Somme des factures créées le mois calendaire dernier avec le nombre

**Formatage des Valeurs :**

- Valeurs supérieures à 1 000 : Afficher avec suffixe "k" (ex. : $1.5k)
- Valeurs supérieures à 1 000 000 : Afficher avec suffixe "M" (ex. : $2.3M)
- Valeurs supérieures à 1 000 000 000 : Afficher avec suffixe "B" (ex. : $1.1B)

**Indicateurs de Tendance :**

- Widgets pour "Ce Mois" et "Le Mois Dernier" montrent le changement en pourcentage
- Flèche verte vers le haut : Augmentation par rapport à la période précédente
- Flèche rouge vers le bas : Diminution par rapport à la période précédente
- Flèche grise vers la droite : Pas de changement

# Liste des Factures

Le tableau des factures affiche toutes les factures avec les colonnes suivantes :

**Descriptions des Colonnes :**

- **ID** - ID de facture unique
- **Titre** - Titre/description de la facture
- **Période** - Période de facturation (date de début - date de fin) ou "N/A" pour les factures uniques
- **Date d'Échéance** - Date d'échéance de paiement
- **Créé** - Date de création de la facture
- **Montant** - Montant total de la facture (coût de détail)
- **Statut** - Payé, Impayé ou Remboursé
- **Actions** - Actions disponibles (varie selon le statut)

**Icônes d'Action :**

- ⬇ **(Télécharger)** - Télécharger le PDF de la facture
- 🗑 **(Supprimer)** - Annuler la facture (uniquement si non payée)
- 💳 **(Payer)** - Payer la facture en ligne (uniquement si impayée)
- ✉ **(E-mail)** - Envoyer l'e-mail de la facture au client

- **(Rembourser)** - Rembourser le paiement Stripe (uniquement pour les factures Stripe payées)

# Générer une Facture

Cliquez sur **"+ Générer une Facture Proforma"** pour créer une nouvelle facture.

**Descriptions des Champs :**

- **Rechercher des Clients** - Sélectionnez un client (uniquement affiché dans la vue système, pré-rempli dans la vue client)
- **Titre** - Titre/nom de la facture (optionnel, par défaut "Facture pour [Période]")
- **Date de Début** - Début de la période de facturation (par défaut 14 jours auparavant)
- **Date de Fin** - Fin de la période de facturation (par défaut aujourd'hui)
- **Date d'Échéance** - Date limite de paiement (par défaut aujourd'hui)
- **Aperçu des Transactions** - Affiche toutes les transactions non facturées dans la plage de dates avec la possibilité d'inclure/exclure des transactions spécifiques

**Sélection de Transactions :**

- ✓ **(Plus Vert)** - Cliquez pour exclure une transaction de la facture
- × **(X Rouge)** - Cliquez pour inclure une transaction précédemment exclue
- **Tout Sélectionner** - Inclure toutes les transactions affichées
- **Tout Effacer** - Exclure toutes les transactions
- Les transactions exclues apparaissent grises avec un texte barré
- Les totaux se mettent à jour en temps réel lorsque vous sélectionnez/désélectionnez des transactions

**Ce Qui Se Passe :**

1. Le système trouve toutes les transactions non facturées pour le client dans la plage de dates
2. Affiche l'aperçu des transactions avec la possibilité d'inclure/exclure des transactions individuelles
3. Montre le calcul en temps réel du sous-total, de la taxe et du total basé sur les transactions sélectionnées
4. Seules les transactions sélectionnées (incluses) sont ajoutées à la facture
5. Génère le PDF de la facture et le met en cache
6. Marque les transactions sélectionnées comme facturées (champ `invoice_id` peuplé)
7. Les transactions exclues restent non facturées et disponibles pour de futures factures
8. La facture apparaît dans la liste avec le statut "Impayé"

**Exemples de Cas d'Utilisation :**

**Facturation Mensuelle :** Définissez la date de début au premier du mois, la date de fin au dernier jour du mois, l'aperçu montre toutes les transactions non facturées de cette période. Sélectionnez tout ou excluez manuellement des transactions spécifiques.

**Facture Spécifique au Service :** Utilisez la même plage de dates, puis excluez manuellement les transactions indésirables (ex. : excluez les transactions non mobiles pour créer une facture uniquement mobile).

**Facture Unique :** Définissez à la fois la date de début et la date de fin au même jour, l'aperçu montre uniquement les transactions de cette date. Excluez

tout frais non pertinent pour cette facture spécifique.

# Voir les Détails de la Facture

Cliquez sur n'importe quelle ligne de facture dans le tableau pour voir les détails complets de la facture y compris toutes les transactions, totaux et actions disponibles.

**Modal des Détails de la Facture :**

- **Informations sur la Facture** - Affiche l'ID de la facture, le titre, les dates, le statut de paiement et le statut d'annulation
- **Liste des Transactions** - Affiche toutes les transactions incluses dans la facture avec :
  - Date de la transaction
  - Titre et description
  - Coût de détail
  - Montant de la taxe et pourcentage (formaté comme `$10.00 (10%)`)
  - Les transactions exonérées de taxe affichent "-" dans la colonne Taxe
- **Résumé des Totaux** - Calcul en temps réel montrant :
  - Nombre de transactions
  - Sous-total (somme de tous les coûts de détail)
  - Taxe (somme de tous les montants de taxe)

    - Total de la Facture (sous-total + taxe)
- **Boutons d'Action** - Les mêmes actions disponibles que dans le tableau :
    - **Télécharger PDF** - Télécharger le PDF de la facture (toujours disponible)
    - **Envoyer un E-mail** - Envoyer la facture au client (factures non annulées)
    - **Payer la Facture** - Traiter le paiement (factures impayées, non annulées uniquement)
    - **Rembourser** - Rembourser le paiement Stripe (uniquement pour les factures Stripe payées)
    - **Supprimer** - Annuler la facture (factures impayées, non annulées uniquement)

# Télécharger les PDFs de Facture

Cliquez sur l'**icône de téléchargement (⬇)** dans le tableau ou sur le bouton **"Télécharger PDF"** dans le modal des détails de la facture pour télécharger une facture au format PDF.

**Processus de Téléchargement :**

1. Cliquez sur l'icône de téléchargement à côté de la facture
2. Un spinner de chargement apparaît pendant la génération (uniquement la première fois)
3. Le navigateur demande de sauvegarder le fichier : `Invoice_01234.pdf`
4. Le PDF s'ouvre ou est enregistré dans le dossier Téléchargements

**Comportement de Mise en Cache du PDF :**

- **Premier Téléchargement** - PDF généré à partir du modèle, mis en cache dans la base de données (peut prendre 2-3 secondes)
- **Téléchargements Suivants** - Téléchargement instantané depuis le cache
- **Invalidation du Cache** - Cache vidé si la facture est modifiée ou si le modèle est mis à jour

**Résolution des Problèmes de Téléchargement :**

- **Le spinner ne s'arrête jamais** - Vérifiez la console du navigateur, l'API peut être hors service
- **PDF vide ou corrompu** - Vérifiez le modèle de facture pour des erreurs de syntaxe
- **Le téléchargement échoue** - Vérifiez les paramètres du bloqueur de fenêtres contextuelles, essayez un autre navigateur

# Payer les Factures

Cliquez sur l'**icône de paiement (□)** pour payer une facture en ligne.

**Processus de Paiement :**

1. Cliquez sur l'icône de paiement sur la facture impayée
2. Le modal de paiement s'ouvre montrant les détails de la facture
3. Sélectionnez la méthode de paiement :
    - **Transaction Stripe** - Facturer la carte de crédit enregistrée (disponible pour tous les utilisateurs)
    - **Espèces** - Paiement en espèces manuel (personnel uniquement)
    - **Remboursement** - Appliquer un remboursement comme paiement (personnel uniquement)
    - **Transaction POS** - Terminal de point de vente (personnel uniquement)

    - **Virement Bancaire** - Virement bancaire manuel (personnel uniquement)
4. Si Stripe est sélectionné :
    - Sélectionnez la carte parmi les méthodes de paiement enregistrées
    - La carte par défaut est pré-sélectionnée
    - Cliquez pour sélectionner une carte différente
5. Si une autre méthode est sélectionnée :
    - Entrez le numéro de référence (optionnel)
6. Cliquez sur **"Payer la Facture"** pour traiter
7. Le système traite le paiement :
    - **Stripe** - Facture la carte via l'API Stripe
    - **Autres méthodes** - Crée une transaction négative pour le montant du paiement
8. Le statut de la facture change en "Payée"
9. Notification de succès affichée

**Self-Care vs Paiement du Personnel :**

**:doc:`Portail Self-Care <self_care_portal>` (Clients) :**

- Seul le paiement Stripe est disponible
- Doit avoir une méthode de paiement enregistrée
- Avertissement affiché si aucune méthode de paiement n'existe
- Lien pour ajouter une méthode de paiement fourni

**Portail du Personnel (Admins) :**

- Toutes les méthodes de paiement disponibles
- Peut marquer la facture comme payée manuellement (espèces, POS, virement bancaire)
- Peut entrer des numéros de référence pour le suivi

**Avertissement de Méthode de Paiement :**

Si le client n'a pas de méthodes de paiement enregistrées, un avertissement est affiché l'invitant à ajouter une méthode de paiement avant de pouvoir payer les factures.

# Envoyer des Factures par E-mail

Cliquez sur l'**icône d'e-mail (✉)** pour envoyer la facture au client.

**Ce Qui Se Passe :**

1. Cliquez sur l'icône d'e-mail à côté de la facture
2. Le système récupère le PDF de la facture depuis le cache (ou génère si non mis en cache)
3. Envoie l'e-mail via Mailjet en utilisant le modèle `api_crmCommunicationCustomerInvoice`
4. L'e-mail inclut :
   - PDF de la facture en pièce jointe
   - Nom du client
   - Numéro de facture et date d'échéance
   - Montant total dû
   - Lien pour payer la facture en ligne
   - Lien pour voir/télécharger la facture
5. Notification de succès : "E-mail de facture envoyé avec succès"

**Destinataires de l'E-mail :**

L'e-mail est envoyé à tous les contacts clients de type "facturation" ou au contact principal si aucun contact de facturation n'existe.

**Variables de Modèle d'E-mail :**

- `{{ var:customer_name }}` - Nom complet du client
- `{{ var:invoice_number }}` - ID de la facture
- `{{ var:invoice_date }}` - Date d'émission de la facture
- `{{ var:due_date }}` - Date d'échéance de paiement
- `{{ var:total_amount }}` - Montant total dû
- `{{ var:invoice_url }}` - Lien pour voir/télécharger le PDF
- `{{ var:pay_url }}` - Lien pour payer la facture en ligne

**Résolution des Problèmes d'E-mail :**

- **E-mail non envoyé** - Vérifiez les identifiants API Mailjet dans `crm_config.yaml`
- **Client ne reçoit pas** - Vérifiez les adresses e-mail des contacts clients
- **PDF non attaché** - Vérifiez que la génération du PDF a réussi (essayez de télécharger d'abord)

# Annuler des Factures

Cliquez sur l'**icône de suppression (🗑)** pour annuler une facture.

**Exigences :**

- La facture doit être **Impayée**
- Les factures payées ne peuvent pas être annulées (doivent être remboursées à la place)

**Comment Annuler :**

1. Localisez la facture impayée dans la liste
2. Cliquez sur l'icône de suppression (🗑)
3. Confirmez dans le modal :

**Ce Qui Se Passe :**

- La facture est marquée comme `void = true`
- Toutes les transactions sont dissociées de la facture (`invoice_id` défini sur null)
- Les transactions redeviennent "non facturées"
- Les transactions peuvent être incluses dans une nouvelle facture
- La facture apparaît dans la liste avec le préfixe "Annulé :" dans le titre
- Les actions de la facture sont désactivées (pas de téléchargement, paiement ou e-mail)
- Peut être vue en filtrant les factures "Annulées"

**Remarques Importantes :**

- Annuler n'est PAS la même chose que rembourser
- **Annuler** = "Cette facture n'aurait jamais dû exister" (erreur de facturation, duplicata)
- **Rembourser** = "Inverser une facture payée valide" (retour d'argent au client)

# Rembourser des Factures

Cliquez sur l'**icône de remboursement (□)** pour rembourser une facture payée.

**Exigences :**

- La facture doit être **Payée**
- La facture doit avoir été payée via **Stripe**
- La facture doit avoir une référence de paiement valide (`payment_reference` ID de l'intention de paiement Stripe)
- Disponible uniquement pour les utilisateurs du personnel (pas de Self-Care)

**Comment Rembourser :**

1. Localisez la facture Stripe payée
2. Cliquez sur l'icône de remboursement (□)
3. Le modal de confirmation de remboursement s'ouvre :

4. Cliquez sur **"Confirmer le Remboursement"**
5. Le système traite le remboursement Stripe :
    - Appelle l'API Stripe pour rembourser le paiement
    - Crée une transaction de remboursement dans Stripe
    - Met à jour la facture avec `refund_reference`
6. Le statut de la facture change en "Remboursé"

7. Notification de succès affichée

**Ce Qui Se Passe Après le Remboursement :**

- La facture reste dans le système (non annulée)
- Le statut affiche "Remboursé"
- Les transactions restent liées à la facture
- Le client reçoit un remboursement sur le mode de paiement d'origine (3-7 jours ouvrables)
- Le tableau de bord Stripe affiche la transaction de remboursement

**Restrictions de Remboursement :**

- Impossible de rembourser les factures payées en espèces, POS ou virement bancaire (inversion manuelle requise)
- Impossible de rembourser partiellement (montant total de la facture uniquement)
- Impossible de rembourser deux fois

# Rechercher et Filtrer les Factures

## Recherche

Utilisez la barre de recherche pour trouver des factures. Recherche à travers :

- ID de la facture
- Titre de la facture
- Nom du client (vue système uniquement)

## Filtres

Appliquez des filtres pour affiner la liste des factures :

**Filtres Disponibles :**

- **Statut d'Annulation** - Tout, Annulé, Non Annulé

- **Statut de Paiement** - Tout, Payé, Pas encore Payé

**Actions de Filtre :**

- **Appliquer les Filtres** - Appliquer les filtres sélectionnés à la liste
- **Réinitialiser les Filtres** - Effacer tous les filtres et afficher toutes les factures

## Tri

Cliquez sur n'importe quel en-tête de colonne pour trier :

- **ID** - Trier par ID de facture (le plus récent/le plus ancien)
- **Titre** - Trier par ordre alphabétique
- **Date d'Échéance** - Trier par date d'échéance
- **Créé** - Trier par date de création
- **Montant** - Trier par coût de détail (le plus élevé/le plus bas)
- **Statut** - Trier par statut de paiement (payé en premier ou impayé en premier)

Cliquez à nouveau pour inverser la direction du tri (ascendant ↔ descendant).

## Pagination

Naviguez à travers de grandes listes de factures avec des contrôles de page affichant la page actuelle, le nombre total de pages et un sélecteur d'articles par page (10, 25, 50 ou 100 articles).

# Flux de Travail Courants pour les Factures

## Flux de Travail 1 : Facturation Mensuelle avec Aperçu des Transactions

1. La fin du mois arrive (ex. : 31 janvier)

2. Accédez à **Facturation → Factures**
3. Cliquez sur **"+ Générer une Facture Proforma"**
4. Sélectionnez un client (ou faites-le par client si vous facturez plusieurs clients)
5. Définissez les dates :
    - Date de Début : 2025-01-01
    - Date de Fin : 2025-01-31
    - Date d'Échéance : 2025-02-15 (15 jours à partir de maintenant)
    - Titre : "Services de Janvier 2025" (optionnel)
6. La section **Aperçu des Transactions** apparaît montrant toutes les transactions non facturées de janvier
7. Passez en revue l'aperçu :
    - Toutes les transactions sont incluses par défaut
    - Vérifiez les totaux : Sous-total, Taxe et Total de la Facture
    - Vérifiez que tous les frais sont corrects
8. Cliquez sur **"Générer la Facture"** (le bouton affiche le nombre de transactions, ex. : "Générer la Facture (15)")
9. Facture créée avec toutes les transactions sélectionnées
10. Cliquez sur la ligne de facture pour voir les détails et vérifier
11. Cliquez sur le bouton **"Envoyer un E-mail"** dans le modal de détails ou sur l'icône d'e-mail dans le tableau
12. Le client reçoit un e-mail de facture avec le PDF et le lien de paiement

# Flux de Travail 2 : Facturation Sélective des Transactions

1. Le client a plusieurs services (Mobile + Internet) et des frais divers
2. Veut des factures séparées pour chaque service
3. **Générer la première facture (Services Mobiles) :**
    - Cliquez sur **"+ Générer une Facture Proforma"**
    - Titre : "Services Mobiles - Janvier 2025"
    - Début/Fin : 1-31 janvier
    - Date d'Échéance : 15 février

- Dans l'aperçu des transactions, **excluez** toutes les transactions non mobiles :
  - Cliquez sur le bouton **X** à côté des transactions Internet
  - Cliquez sur le bouton **X** à côté des frais divers
  - Seules les transactions de service mobile restent sélectionnées
- Vérifiez que les totaux reflètent uniquement les services mobiles
- Cliquez sur **"Générer la Facture"** (affiche le nombre de transactions mobiles)

4. **Générer la deuxième facture (Services Internet) :**
   - Cliquez à nouveau sur **"+ Générer une Facture Proforma"**
   - Titre : "Services Internet - Janvier 2025"
   - Début/Fin : 1-31 janvier (même période)
   - Dans l'aperçu des transactions :
     - Les transactions mobiles ont déjà été facturées (n'apparaissent pas)
     - Excluez les frais divers en utilisant le bouton **X**
     - Seules les transactions de service Internet restent
   - Cliquez sur **"Générer la Facture"**
5. **Générer la troisième facture (Frais Supplémentaires) :**
   - Cliquez à nouveau sur **"+ Générer une Facture Proforma"**
   - Titre : "Frais Supplémentaires - Janvier 2025"
   - Seuls les frais divers non facturés apparaissent dans l'aperçu
   - Cliquez sur **"Tout Sélectionner"** pour inclure tout
   - Cliquez sur **"Générer la Facture"**
6. Envoyez les trois factures au client

# Flux de Travail 3 : Exclusion des Transactions Contestées ou en Attente

1. La fin de la période de facturation arrive
2. Accédez à l'onglet **Facturation** du client
3. Cliquez sur **"+ Générer une Facture Proforma"**
4. Définissez les dates de la période de facturation
5. L'aperçu des transactions montre 20 transactions

6. Le client a contesté un frais et un autre est en attente d'investigation
7. Dans l'aperçu des transactions :
    - Localisez la transaction contestée (ex. : "Frais de dépassement de données")
    - Cliquez sur le bouton **X** pour l'exclure
    - Localisez la transaction en attente (ex. : "Frais d'installation")
    - Cliquez sur le bouton **X** pour l'exclure
    - Le nombre de transactions se met à jour : "18 Transactions sélectionnées"
    - Les totaux se recalculent automatiquement
8. Passez en revue les totaux mis à jour (exclut les montants contestés)
9. Cliquez sur **"Générer la Facture (18)"**
10. Facture générée avec uniquement les transactions approuvées
11. Les transactions contestées/en attente restent non facturées pour le prochain cycle de facturation

# Flux de Travail 4 : Revue Rapide de la Facture et Ajustement

1. Le personnel gén��re la facture mensuelle
2. L'aperçu des transactions montre un total élevé inattendu
3. Passez en revue chaque transaction dans l'aperçu :
    - Remarquez un frais dupliqué pour le même service
    - Cliquez sur **X** pour exclure le duplicata
    - Remarquez une transaction de test qui ne devrait pas être facturée
    - Cliquez sur **X** pour exclure la transaction de test
4. Les totaux se mettent à jour en temps réel
5. Vérifiez que le nouveau total correspond au montant attendu
6. Cliquez sur **"Générer la Facture"** avec les transactions corrigées
7. Revenez en arrière et annulez/supprimez les transactions exclues si nécessaire
8. Envoyez la facture au client en toute confiance

# Flux de Travail 5 : Facture d'Installation Unique

1. Le technicien sur le terrain termine l'installation
2. Le personnel ajoute manuellement la transaction d'installation
3. Accédez à l'onglet **Facturation** du client
4. Cliquez sur **"+ Générer une Facture Proforma"**
5. Définissez les dates :
   - Date de Début : aujourd'hui
   - Date de Fin : aujourd'hui
   - Date d'Échéance : aujourd'hui + 7 jours
   - Titre : "Services d'Installation"
6. L'aperçu des transactions montre uniquement les transactions d'aujourd'hui
7. Vérifiez que le frais d'installation apparaît
8. Excluez tout frais récurrent en utilisant le bouton **X** (s'il est présent)
9. Cliquez sur **"Générer la Facture"**
10. Envoyez immédiatement au client
11. Le client paie en ligne via Stripe

# Flux de Travail 6 : Revue de la Facture Avant le Contact Client

1. Le client appelle avec une question de facturation
2. Le personnel navigue vers la liste des factures du client
3. **Cliquez sur la ligne de facture** pour ouvrir le modal des Détails de la Facture
4. Passez en revue les informations de la facture :
   - ID de la facture, dates, statut
   - Toutes les transactions incluses avec descriptions
   - Détail de la taxe par transaction
   - Sous-total, Taxe et Montants Totaux
5. Répondez aux questions du client avec des détails exacts
6. Si le client demande le PDF, cliquez sur le bouton **"Télécharger PDF"** dans le modal

7. Si le client demande un renvoi d'e-mail, cliquez sur le bouton **"Envoyer un E-mail"**
8. Fermez le modal lorsque vous avez terminé

# Flux de Travail 7 : Correction d'une Erreur de Facturation

1. Le client signale un frais incorrect
2. Le personnel clique sur la ligne de facture pour voir les détails
3. Passe en revue la liste des transactions dans le modal des Détails de la Facture
4. Identifie la transaction incorrecte
5. La facture est impayée, donc elle peut être annulée
6. Cliquez sur le bouton **"Supprimer"** dans le pied de modal
7. Confirmez l'annulation
8. Les transactions redeviennent non facturées
9. Le personnel modifie ou supprime la transaction incorrecte de la liste des transactions
10. Générer une nouvelle facture avec les transactions corrigées :
    - Utilisez l'aperçu des transactions pour exclure la transaction corrigée si nécessaire
    - Incluez uniquement les frais valides
11. Envoyez la facture corrigée au client

# Flux de Travail 8 : Traitement de Plusieurs Paiements

1. Le client apporte des espèces pour payer plusieurs factures
2. Accédez à l'onglet **Facturation** du client
3. Consultez les factures impayées
4. Cliquez sur la première ligne de facture pour voir les détails
5. Vérifiez le montant et les transactions
6. Cliquez sur le bouton **"Payer la Facture"** dans le pied de modal
7. Sélectionnez la méthode de paiement **"Espèces"**

8. Entrez la référence : "Espèces payées 2025-01-15"
9. Cliquez sur **"Payer la Facture"**
10. Le modal se ferme, la facture est marquée comme "Payée"
11. Répétez pour les factures restantes
12. Toutes les factures sont maintenant marquées comme "Payées"

## Flux de Travail 9 : Gestion d'une Demande de Remboursement

1. Le client demande un remboursement pour un trop-perçu
2. Le personnel vérifie que la facture a été payée via Stripe
3. Accédez à la facture dans la liste
4. Cliquez sur la ligne de facture pour voir les détails
5. Vérifiez les informations de paiement et le montant
6. Cliquez sur le bouton **"Rembourser"** dans le pied de modal (n'apparaît que pour les factures Stripe)
7. Confirmez le remboursement
8. Le système traite le remboursement Stripe
9. Le statut de la facture change en "Remboursé"
10. Le client reçoit un remboursement dans les 3-7 jours ouvrables
11. Le personnel suit avec le client pour confirmer la réception

# Résolution des Problèmes

**Impossible de générer la facture - Aucune transaction trouvée**

- **Cause :** Aucune transaction non facturée dans la plage de dates spécifiée
- **Correction :** Vérifiez la liste des transactions, vérifiez que les transactions existent et ne sont pas déjà facturées. Ajustez la plage de dates ou retirez le filtre.

**Échec de la génération du PDF de la facture**

- **Cause :** Erreur de syntaxe dans le modèle, plantage de WeasyPrint ou données client manquantes

- **Correction :** Vérifiez le HTML du modèle de facture pour des erreurs, vérifiez que les champs d'adresse du client sont peuplés, consultez les journaux de l'API.

**Échec de paiement avec erreur Stripe**

- **Cause :** Carte refusée, fonds insuffisants, carte expirée ou problème avec l'API Stripe
- **Correction :** Essayez une autre méthode de paiement, vérifiez que la carte est valide, consultez le tableau de bord Stripe pour connaître la raison du refus.

**Impossible d'annuler la facture**

- **Cause :** La facture est déjà payée
- **Correction :** Les factures payées ne peuvent pas être annulées. Si un remboursement est nécessaire, utilisez la fonction de remboursement pour les factures Stripe ou créez manuellement une transaction de crédit.

**E-mail de facture non envoyé**

- **Cause :** Identifiants API Mailjet invalides, le client n'a pas de contact de facturation, ou modèle d'e-mail manquant
- **Correction :** Vérifiez la configuration de Mailjet dans `crm_config.yaml`, vérifiez les contacts du client, vérifiez que le modèle d'e-mail de facture existe.

**Bouton de remboursement n'apparaît pas**

- **Cause :** Facture payée en espèces/POS/virement bancaire (pas Stripe), ou facture non payée
- **Correction :** Le bouton de remboursement n'apparaît que pour les paiements Stripe. Pour d'autres méthodes de paiement, créez une transaction de crédit manuelle.

**Le téléchargement du PDF montre l'ancien design du modèle**

- **Cause :** PDF mis en cache avant la mise à jour du modèle

- **Correction :** Videz le cache PDF de la facture : `DELETE FROM Invoice_PDF_Cache WHERE invoice_id = X;`

**Le client ne peut pas payer la facture (aucune méthode de paiement)**

- **Cause :** Aucune méthode de paiement enregistrée dans le portail Self-Care
- **Correction :** Le client doit ajouter une carte de crédit à la page **Méthodes de Paiement** avant de payer les factures.

**Plusieurs factures générées pour la même période**

- **Cause :** Le personnel a généré la facture deux fois, ou les plages de dates se chevauchent
- **Correction :** Annulez la facture en double. Ajustez les plages de dates pour éviter le chevauchement. Utilisez l'aperçu des transactions pour vous assurer que les ensembles de transactions sont uniques.

**L'aperçu des transactions ne montre aucune transaction**

- **Cause :** Toutes les transactions dans la plage de dates ont déjà été facturées ou aucune transaction n'existe
- **Correction :** Vérifiez que la plage de dates est correcte. Vérifiez la liste des transactions pour confirmer que des transactions existent. Filtrez les factures pour voir quelle facture contient les transactions.

**Impossible d'exclure une transaction de la génération de facture**

- **Cause :** La transaction a déjà été facturée ou problème de navigateur
- **Correction :** Vérifiez que la transaction s'affiche dans l'aperçu avec une coche. Rafraîchissez la page et réessayez. Effacez le cache du navigateur si le problème persiste.

**Le total de la facture ne correspond pas au montant attendu**

- **Cause :** Transactions inattendues incluses, taxe non calculée, ou transactions exclues comptées
- **Correction :** Passez en revue l'aperçu des transactions avec soin. Vérifiez chaque coût de détail et taxe de transaction. Vérifiez que les transactions

exclues sont grises. Vérifiez le badge de nombre de transactions sur le bouton Générer la Facture.

**Le bouton Générer la Facture est désactivé**

- **Cause :** Aucune transaction sélectionnée ou plage de dates invalide
- **Correction :** Assurez-vous qu'au moins une transaction est incluse (non exclue). Vérifiez que la Date de Début est avant la Date de Fin. Vérifiez que la Date d'Échéance est définie.

**Le modal des Détails de la Facture ne s'ouvre pas**

- **Cause :** Erreur JavaScript ou page non complètement chargée
- **Correction :** Rafraîchissez la page. Vérifiez la console du navigateur pour des erreurs. Essayez un autre navigateur. Vérifiez la connexion Internet.

**La taxe de transaction ne s'affiche pas dans les Détails de la Facture**

- **Cause :** La transaction a une taxe de 0 % ou `tax_amount` est nul
- **Correction :** Vérifiez que la transaction a un `tax_percentage` défini. Vérifiez que `tax_amount` a été calculé lorsque la transaction a été créée. Mettez à jour la transaction si nécessaire.

**Les boutons d'action manquent dans le modal des Détails de la Facture**

- **Cause :** La facture est annulée ou l'utilisateur n'a pas les permissions
- **Correction :** Les factures annulées n'affichent que le bouton Télécharger PDF. Vérifiez le statut de la facture. Vérifiez le rôle et les permissions de l'utilisateur.

# Documentation Connexe

- `integrations_mailjet` - Livraison d'e-mail de facture et modèles
- `administration_configuration` - Configuration du modèle de facture
- `payments_transaction` - Création de transactions qui apparaissent sur les factures

- `payments_process` - Traitement des paiements de factures
- `basics_payment` - Gestion des méthodes de paiement
- `payment_system_guide` - Référence API de paiement et configuration des fournisseurs

# Traitement des paiements

Le système traite la plupart des paiements automatiquement. Parfois, vous devez traiter un paiement manuellement.

Pour payer une facture, sélectionnez la facture impayée et cliquez sur le bouton "Payer".

Cela ouvrira un formulaire de paiement, où vous pouvez entrer le mode de paiement et cliquer sur "Soumettre" pour traiter le paiement.

Le client recevra automatiquement un reçu pour le paiement, et la facture sera marquée comme payée.

Pour les virements bancaires, entrez la référence de paiement. Entrez également la date à laquelle le paiement a été effectué, si elle est différente de la date actuelle.

# Transactions Clients

Tout ce qui coûte de l'argent dans le système est enregistré en tant que transaction sous le client.

Chaque transaction a un montant monétaire pour le coût de gros et le coût de détail, ainsi qu'une description de ce à quoi sert la transaction.

Les transactions peuvent être générées automatiquement par le système, par exemple, lorsqu'un service est provisionné, une transaction est créée pour le coût d'installation, et lorsqu'un service est facturé, une transaction est créée pour le coût de détail.

Les transactions peuvent également être créées manuellement, par exemple, si un client reçoit un crédit, une transaction est créée pour le montant du crédit, ou si des frais d'installation sont facturés, une transaction est créée pour les frais d'installation.

Les transactions sont regroupées pour former Factures, qui sont envoyées au client pour paiement.

# Accéder aux Transactions

Les transactions peuvent être consultées au niveau du système ou par client :

**Vue par Client :**

1. Naviguez vers **Clients → [Sélectionner Client]**
2. Cliquez sur l'onglet **Facturation**
3. Consultez la liste des transactions dans la première carte

**Vue Système :**

1. Naviguez vers **Facturation → Transactions** (dans le menu principal)
2. Consultez toutes les transactions de tous les clients

# Widgets de Statistiques des Transactions

En haut de la page des transactions, quatre cartes statistiques affichent des résumés financiers :

**Descriptions des Widgets :**

- **Total des Transactions** - Somme de tous les coûts de détail des transactions (tout le temps)
- **Total des Transactions Non Facturées** - Somme des transactions non encore incluses dans une facture
- **Total des Transactions Ce Mois** - Somme des transactions créées ce mois calendaire
- **Total des Transactions Le Mois Dernier** - Somme des transactions créées le mois calendaire dernier

**Formatage des Valeurs :**

- Valeurs supérieures à 1 000 : Afficher avec le suffixe "k" (par exemple, $1.5k)
- Valeurs supérieures à 1 000 000 : Afficher avec le suffixe "M" (par exemple, $2.3M)
- Valeurs supérieures à 1 000 000 000 : Afficher avec le suffixe "B" (par exemple, $1.1B)

# Liste des Transactions

Le tableau des transactions affiche toutes les transactions avec les colonnes suivantes :

**Descriptions des Colonnes :**

- **ID** - ID unique de la transaction
- **Date** - Date de création de la transaction
- **Titre** - Nom court de la transaction
- **Description** - Description détaillée de ce à quoi sert la transaction
- **Montant** - Coût de détail (positif pour les frais, négatif pour les crédits)
- **Facture** - ID de la facture si la transaction a été facturée (lien cliquable)
- **Statut** - Coche si facturé, tiret si pas encore facturé

**Actions Par Ligne :**

Chaque ligne a un menu d'actions ( ⋮ ) avec des options :

- **Voir Détails** - Ouvre le modal de détails de la transaction
- **Télécharger PDF de la Facture** - Télécharger le PDF (uniquement si facturé)
- **Annuler la Transaction** - Marquer la transaction comme annulée (uniquement si pas facturée)

# Types de Transactions

Les transactions se classent en deux catégories principales :

## Transactions Débitées (Frais)

Montants positifs qui augmentent le solde dû par le client :

- **Frais d'Installation de Service** - Frais uniques lors de la provision du service
- **Frais de Service Mensuels** - Frais récurrents pour les services
- **Frais d'Installation** - Frais pour les visites de techniciens sur le terrain
- **Frais d'Équipement** - Frais pour modems, routeurs, cartes SIM
- **Frais de Retard** - Pénalités pour factures en retard
- **Frais Manuels** - Frais personnalisés ajoutés par le personnel

## Transactions Crédits (Paiements/Remboursements)

Montants négatifs qui diminuent le solde dû par le client :

- **Paiements en Espèces** - Client a payé en espèces
- **Paiements par Carte** - Client a payé par carte de crédit/débit
- **Paiements par Virement Bancaire** - Client a payé par virement bancaire
- **Crédits de Compte** - Crédits de bonne volonté, compensation
- **Remboursements** - Argent retourné au client
- **Remises** - Remises promotionnelles ou de fidélité

# Ajouter une Transaction Manuellement

Cliquez sur **"+ Ajouter une Transaction"** pour ouvrir le modal d'ajout de transaction.

**Transaction Débitée (Frais) :**

**Transaction Crédit (Paiement/Remboursement) :**

**Descriptions des Champs :**

- **Type de Transaction** - Sélectionnez Débit (frais) ou Crédit (paiement/remboursement)
- **Type de Crédit** - Si Crédit sélectionné, choisissez le mode de paiement (Espèces, Carte, Virement Bancaire)
- **Titre** - Nom court pour la transaction (obligatoire)
- **Description** - Explication détaillée (facultatif)
- **Coût de Détail** - Montant que le client paie (obligatoire, nombre positif)
- **Coût de Gros** - Votre coût (facultatif, pour le suivi de la marge)
- **Pourcentage de Taxe** - Taux de taxe appliqué à cette transaction (facultatif, par défaut à la taxe produit ou 0%)
- **Service** - Lier la transaction à un service spécifique (facultatif)
- **Site** - Lier la transaction à un site spécifique (facultatif)
- **Date de Transaction** - Date de la transaction (par défaut à aujourd'hui)

**Validation :**

- Le titre et le coût de détail sont obligatoires
- Le coût de détail doit être un nombre positif
- Si le type de Crédit est sélectionné, un type de crédit doit être choisi

**Ce Qui Se Passe :**

1. Transaction créée dans la base de données
2. Apparaît dans la liste des transactions du client
3. Inclus dans le compte des "Transactions Non Facturées"
4. Disponible pour inclusion dans la prochaine génération de factures
5. Entrée de journal d'activité créée

# Recherche et Filtrage des Transactions

## Recherche

Utilisez la barre de recherche pour trouver des transactions. Recherche à travers :

- ID de Transaction
- Titre
- Description
- ID de Facture

## Filtres

Appliquez des filtres pour affiner la liste des transactions :

**Filtres Disponibles :**

- **Statut d'Annulation** - Tout, Annulé, Non Annulé
- **Statut de Facture** - Tout, Facturé, Non Facturé

**Actions de Filtrage :**

- **Appliquer les Filtres** - Appliquer les filtres sélectionnés à la liste
- **Réinitialiser les Filtres** - Effacer tous les filtres et afficher toutes les transactions

## Tri

Cliquez sur n'importe quel en-tête de colonne pour trier :

- **ID** - Trier par ID de transaction (le plus récent/le plus ancien)
- **Date** - Trier par date de transaction
- **Titre** - Trier par ordre alphabétique
- **Montant** - Trier par coût de détail (le plus élevé/le plus bas)

- **Facture** - Trier par ID de facture

Cliquez à nouveau pour inverser la direction du tri (ascendant ↔ descendant).

# Annuler des Transactions

Les transactions ajoutées par erreur peuvent être **annulées** (marquées comme supprimées).

**Exigences :**

- La transaction ne doit PAS être facturée
- Une fois facturées, les transactions ne peuvent pas être annulées (doivent être remboursées à la place)

**Comment Annuler :**

1. Localisez la transaction dans la liste
2. Cliquez sur le menu d'actions ( ⋮ )
3. Sélectionnez **"Annuler la Transaction"**
4. Confirmez dans le modal

**Ce Qui Se Passe :**

- Transaction marquée comme `void = true`

- N'apparaît plus dans la liste par défaut des transactions
- Exclue de la génération de factures
- Peut être consultée en filtrant pour les transactions "Annulées"
- Déduite du total des "Transactions Non Facturées"

**Transactions liées au portefeuille** (`wallet_ledger_id` défini) :

- Le solde du portefeuille est inversé avant l'annulation (débit pour annuler un crédit, crédit pour annuler un débit)
- **Ajout de Fonds carte de recharge** (`payment_type=wallet_topup_card`) : portefeuille débité et carte remboursée
- **Recharge en espèces / paiement de facture de portefeuille / autres lignes de portefeuille :** portefeuille ajusté uniquement ; **pas de remboursement de carte**
- **Portefeuille insuffisant :** l'annulation échoue si les fonds rechargés ont déjà été dépensés (la transaction reste active)

**Frais de service / de provisionnement** provenant de la capture hybride n'ont généralement **pas** de `wallet_ledger_id` sur la ligne de facturation (carte → portefeuille → débit se produit en interne). L'annulation de ces lignes ne définit que `void=true` aujourd'hui ; un crédit complet du portefeuille pour les annulations de service est prévu sur la branche des documents financiers.

**API :** `DELETE /crm/transaction/transaction_id/{transaction_id}`. Voir guide_système_de_paiement.md : Annuler la Transaction.

**Remarque :** Annuler n'est PAS la même chose que rembourser. Annuler signifie "cette transaction n'aurait jamais dû exister." Rembourser signifie "inverser une transaction valide."

# Taxe sur les Transactions

Les transactions peuvent inclure une taxe, qui est automatiquement calculée en fonction de la configuration fiscale du produit ou spécifiée manuellement par transaction.

**Comportement de la Taxe :**

- **Transactions Débitées (Frais)** - La taxe est appliquée aux frais en fonction de :
  - **Pourcentage de Taxe du Produit** - Si la transaction est liée à un produit, le pourcentage de taxe du produit est automatiquement appliqué
  - **Surcharge Manuelle** - Le personnel peut remplacer le pourcentage de taxe lors de la création d'une transaction
  - **Montant de la Taxe** - Calculé comme : `retail_cost × (tax_percentage / 100)`
  - **Format d'Affichage** - Affich�� comme : `$10.00 (10%)` dans les listes de transactions
- **Transactions Crédits (Paiements/Remboursements)** - Aucune taxe n'est appliquée aux crédits
  - Le champ de pourcentage de taxe est masqué pour les transactions de crédit
  - La taxe est automatiquement fixée à 0 % pour tous les paiements et remboursements
  - Les crédits réduisent le solde impayé du client sans implications fiscales

**Exemple de Calcul de Taxe :**

- Produit : Forfait Mobile avec 10 % de taxe, $50.00 coût de détail
- Calcul de Taxe Automatique : $50.00 × 0.10 = $5.00
- Affichage : $5.00 (10 %)

**Taxe Nulle (NIL/Exempt) :**

- Les produits peuvent être exonérés de taxe en fixant le pourcentage de taxe à 0
- La taxe par défaut est de 0 % si non spécifiée
- Les transactions exonérées de taxe affichent "-" dans la colonne Taxe

# Vue Détails de la Transaction

Cliquez sur une transaction pour voir tous les détails :

# Transactions Facturées vs Non Facturées

**Transactions Non Facturées :**

- Pas encore incluses dans une facture
- Disponibles pour la prochaine génération de factures

- Peuvent être annulées
- Comptent dans le total des "Transactions Non Facturées"
- Le statut affiche un tiret (-)

**Transactions Facturées :**

- Incluses dans une facture
- Ne peuvent pas être annulées (doivent être remboursées si nécessaire)
- ID de Facture cliquable (lien vers les détails de la facture)
- Le statut affiche une coche (✓)
- Ne peut pas être modifié

**Génération de Facture :**

Lorsque vous générez une facture pour un client :

1. Le système trouve toutes les transactions non facturées pour ce client
2. Filtrer par plage de dates (facultatif)
3. Transactions incluses dans la nouvelle facture
4. Champ `invoice_id` de la transaction peuplé
5. Transaction maintenant marquée comme "facturée"

Voir `payments_invoices` pour les détails de génération de factures.

# Flux de Travail Courants

## Flux de Travail 1 : Crédit Manuel pour Panne de Service

1. Le client appelle : "Le service était hors ligne pendant 2 jours"
2. Le personnel décide de créditer £10
3. Naviguez vers l'onglet **Facturation** du client
4. Cliquez sur **"+ Ajouter une Transaction"**
5. Sélectionnez le type de transaction **Crédit**
6. Sélectionnez le type de crédit **Paiement en Espèces**

7. Entrez le titre : "Crédit pour Panne de Service"
8. Entrez la description : "Compensation pour panne de 2 jours 8-9 Jan"
9. Entrez le coût de détail : 10.00
10. Sélectionnez le service concerné dans le menu déroulant
11. Cliquez sur **"Ajouter une Transaction"**
12. La transaction apparaît avec un montant de -£10.00
13. Sera incluse dans la prochaine facture en tant que crédit

# Flux de Travail 2 : Frais d'Installation Manuel

1. Le technicien de terrain installe le service
2. Le personnel doit facturer des frais d'installation de £75
3. Naviguez vers l'onglet **Facturation** du client
4. Cliquez sur **"+ Ajouter une Transaction"**
5. Sélectionnez le type de transaction **Débit**
6. Entrez le titre : "Frais d'Installation"
7. Entrez la description : "Visite du technicien de terrain pour installation de fibre"
8. Entrez le coût de détail : 75.00
9. Entrez le coût de gros : 45.00 (facultatif, pour le suivi de la marge)
10. Sélectionnez le service installé
11. Sélectionnez le site où installé
12. Cliquez sur **"Ajouter une Transaction"**
13. La transaction apparaît dans la liste non facturée
14. Sera incluse dans la prochaine facture

# Flux de Travail 3 : Annuler une Transaction Dupliquée

1. Le personnel remarque une transaction dupliquée
2. Vérifiez que la transaction n'est PAS encore facturée
3. Cliquez sur le menu d'actions ( ⋮ ) sur la transaction dupliquée
4. Sélectionnez **"Annuler la Transaction"**
5. Confirmez dans le modal

6. La transaction est supprimée de la liste
7. Le total non facturé diminue en conséquence

## Flux de Travail 4 : Trouver des Transactions pour Facture

1. Besoin de générer une facture mensuelle
2. Cliquez sur **Filtre de Facture : "Non Facturé"**
3. Cliquez sur **Appliquer les Filtres**
4. Consultez toutes les transactions non facturées
5. Notez le montant total des widgets
6. Naviguez pour générer la facture
7. Sélectionnez la plage de dates (par exemple, 1-31 Jan)
8. Transactions dans la plage incluses dans la facture

# Dépannage

**Impossible d'annuler la transaction**

- **Cause :** Transaction déjà facturée
- **Solution :** La transaction fait partie de l'historique des factures. Si un remboursement est nécessaire, créez une transaction de Crédit à la place.
- **Cause :** Échec de l'annulation de la recharge du portefeuille (solde de portefeuille insuffisant)
- **Solution :** Le client a déjà dépensé les fonds rechargés. Utilisez un débit/ajustement manuel du portefeuille ou des flux de renversement de documents financiers ; vous ne pouvez pas annuler la ligne de recharge tant que le portefeuille ne couvre pas le montant.
- **Cause :** Échec du remboursement par carte (Ajout de Fonds uniquement)
- **Solution :** Le renversement du portefeuille est annulé automatiquement. Résolvez le problème du fournisseur de paiement et réessayez d'annuler.

**Transactions dupliquées apparaissant**

- **Cause :** Service facturé plusieurs fois ou erreur de provisionnement
- **Solution :** Annulez la ou les transactions dupliquées si elles ne sont pas facturées. Si facturées, émettez un crédit.

**Transaction n'apparaissant pas dans la liste**

- **Cause :** Filtres appliqués ou transaction annulée
- **Solution :** Cliquez sur "Réinitialiser les Filtres" pour afficher toutes les transactions. Pour voir les transactions annulées, filtrez par "Annulé : Annulé".

**Le total non facturé ne correspond pas à ce qui est attendu**

- **Cause :** Certaines transactions déjà facturées, ou transactions annulées exclues
- **Solution :** Appliquez le filtre "Facture : Non Facturée" pour voir uniquement les non facturées. Vérifiez les transactions annulées séparément.

**Impossible d'ajouter une transaction (champ client désactivé)**

- **Cause :** Consultation de la page des transactions spécifiques au client
- **Solution :** Le client est pré-sélectionné. Si vous devez ajouter une transaction pour un client différent, allez à la page des Transactions au niveau du système.

# Documentation Connexe

- `payments_invoices` - Génération et gestion des factures
- `payments_process` - Traitement des paiements contre les factures
- `basics_payment` - Aperçu des méthodes de paiement
- `csa_activity_log` - Consultation de l'historique des transactions dans le journal d'activité

# Chargement et Paiements depuis les Playbooks

Ce guide explique comment mettre en œuvre le chargement et le traitement des paiements dans les playbooks Ansible pour les flux de travail de provisionnement OmniCRM.

> **Référence API Complète** : Pour des détails complets sur les API de paiement, le routage de portefeuille, les remboursements et l'architecture indépendante des fournisseurs, voir le **Guide de l'API du Système de Paiement**

## Vue d'ensemble

Les Playbooks Ansible OmniCRM peuvent gérer le traitement des paiements de plusieurs manières différentes, mais toutes appellent simplement des appels API sur le CRM pour facturer.

1. **Flux de Paiement à Deux Phases** - Pour les services payants nécessitant une autorisation de paiement et une capture immédiates (c'est-à-dire Services Prépayés)
2. **Création de Transaction Directe** - Pour ajouter des frais/crédits sans traitement de paiement immédiat (par exemple, frais d'installation, crédits manuels) qui peuvent être traités ultérieurement (c'est-à-dire services Postpad)

Ces approches garantissent des transactions atomiques où les clients ne sont facturés que si le provisionnement réussit, avec un rollback automatique si une étape échoue.

# Flexibilité des Prix : Playbooks comme Moteur de Script

**Important :** Les valeurs en dollars définies dans les produits et services (par exemple, `retail_cost`, `wholesale_cost`, `retail_setup_cost`) sont simplement des **valeurs par défaut** stockées dans la base de données. Elles ne dictent pas ce que vous devez facturer. Le playbook a un contrôle total sur le prix final.

Le playbook Ansible est essentiellement un **moteur de script** que vous pouvez adapter pour répondre à toutes les exigences de logique commerciale. Vous pouvez :

- **Utiliser les prix stockés tels quels** - Référez-vous simplement à `api_response_product.json.retail_cost` dans votre autorisation
- **Remplacer complètement les prix** - Facturer un montant différent quelle que soit la définition du produit
- **Appliquer des remises dynamiques** - Calculer des pourcentages de réduction en fonction du niveau de client, des promotions ou de la fidélité
- **Mettre en œuvre des prix par paliers** - Facturer des taux différents en fonction de la quantité, des niveaux d'utilisation ou des conditions contractuelles
- **Tarification groupée** - Combiner plusieurs produits avec des remises groupées personnalisées
- **Tarification basée sur le temps** - Ajuster les prix en fonction de l'heure de la journée, de la saison ou des périodes promotionnelles
- **Tarification spécifique au client** - Rechercher des tarifs négociés à partir des contrats clients

## Exemple : Remplacement du Prix du Produit

```
# Le retail_cost du produit est de 99,00 $, mais nous voulons
facturer 79,00 $ pour une promotion
- name: Définir le prix promotionnel (ignorant le retail_cost du
produit)
  set_fact:
    charge_amount: 79.00

- name: Autoriser le paiement promotionnel
  uri:
    url: "http://localhost:5000/crm/payments/authorize/hold"
    method: POST
    body_format: json
    body:
      customer_id: "{{ customer_id | int }}"
      amount: "{{ charge_amount | float }}"  # Utilisant notre
remplacement, pas retail_cost
      # ...
```

## Exemple : Remise par Niveau Client

```
# Appliquer une remise en fonction du niveau client
- name: Obtenir le niveau client
  uri:
    url: "http://localhost:5000/crm/customer/{{ customer_id }}"
    # ...
  register: customer_info

- name: Calculer la remise par niveau
  set_fact:
    base_price: "{{ api_response_product.json.retail_cost | float
}}"
    discount_percent: >-
      {% if customer_info.json.tier == 'gold' %}20
      {% elif customer_info.json.tier == 'silver' %}10
      {% else %}0{% endif %}

- name: Appliquer la remise au prix final
  set_fact:
    final_price: "{{ (base_price | float) * (1 - (discount_percent
| float / 100)) | round(2) }}"
```

## Exemple : Tarification des Partenaires de Gros

```
# Les partenaires de gros paient le wholesale_cost au lieu du
retail_cost
- name: Déterminer le prix en fonction du type de client
  set_fact:
    charge_amount: >-
      {% if customer_info.json.is_wholesale_partner %}
      {{ api_response_product.json.wholesale_cost | float }}
      {% else %}
      {{ api_response_product.json.retail_cost | float }}
      {% endif %}
```

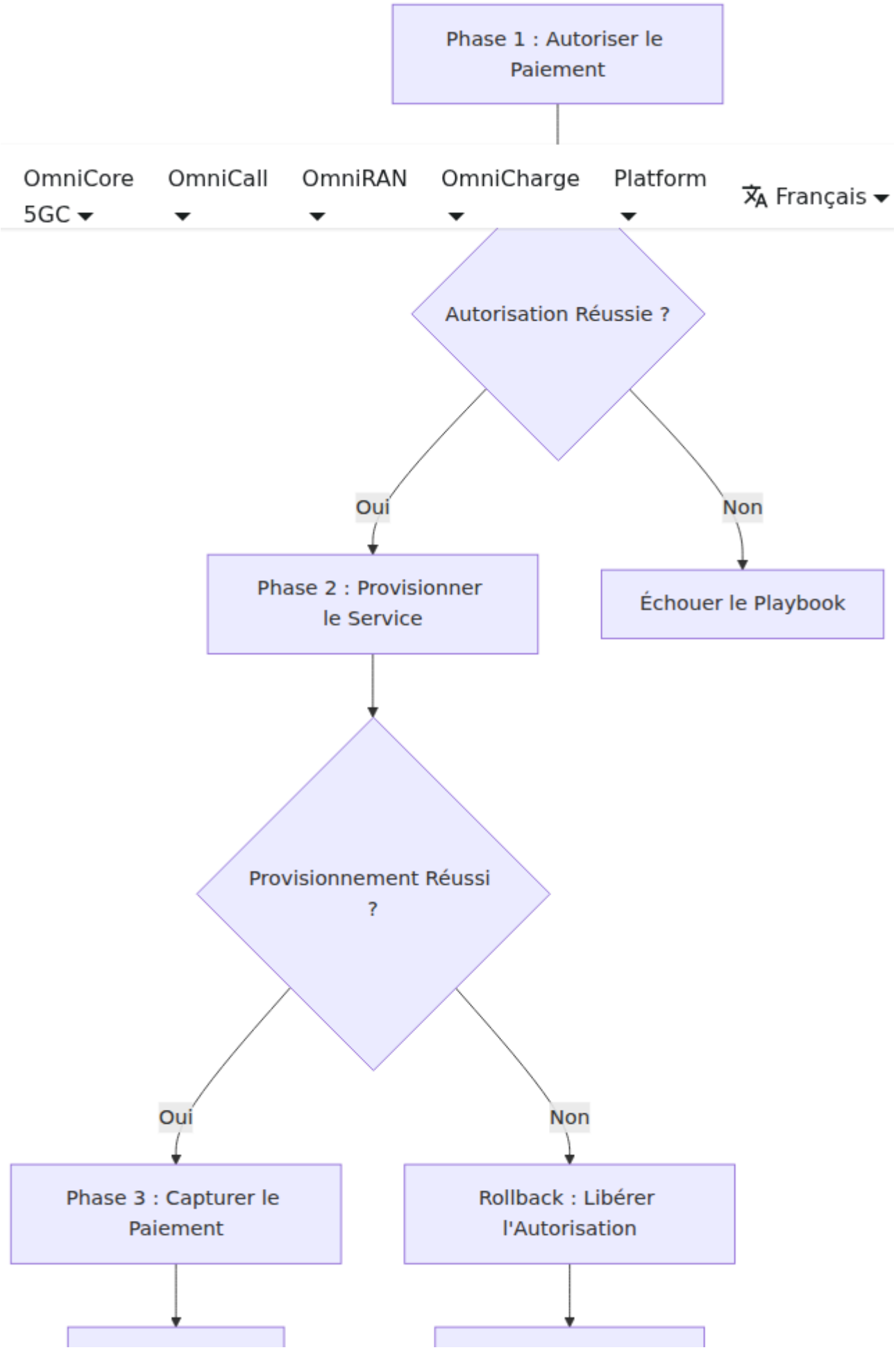
Le point clé est que **vous n'êtes pas enfermé dans un modèle de tarification**. Le playbook vous donne un contrôle programmatique total pour mettre en œuvre toutes les règles commerciales que votre organisation exige.

Les prix des produits/services dans le CRM ne sont que des valeurs par défaut pratiques que les playbooks peuvent utiliser ou ignorer selon les besoins.

# Flux de Paiement à Deux Phases

Le modèle de commit à deux phases est utilisé lorsque vous devez facturer le moyen de paiement d'un client pendant le provisionnement. Cela garantit que le client n'est facturé que si le provisionnement se termine avec succès.

# Vue d'ensemble du Flux


Phase 1 : Autoriser le Paiement
OmniCore 5GC
OmniCall
OmniRAN
OmniCharge
Platform
Français
Autorisation Réussie ?
Oui
Non
Phase 2 : Provisionner le Service
Échouer le Playbook
Provisionnement Réussi ?
Oui
Non
Phase 3 : Capturer le Paiement
Rollback : Libérer l'Autorisation




# Modèle de Mise en Œuvre

Le modèle suit ces phases :

**Phase 1 : Autorisation** - Retenir des fonds sur le moyen de paiement
**Phase 2 : Provisionnement** - Exécuter les mises à jour de solde/action OCS
**Phase 3 : Capture** - Finaliser le paiement après un provisionnement réussi
**Rollback** - Libérer l'autorisation si le provisionnement échoue

# Exemple Complet

Voici un exemple complet du fichier `play_topup_charge_then_action.yaml` :

```yaml
- name: Play Topup - Charger la carte puis actionner le Topup
  hosts: localhost
  gather_facts: no
  become: False

  tasks:
    - name: Inclure les vars de crm_config
      ansible.builtin.include_vars:
        file: "../../crm_config.yaml"
        name: crm_config

    # Obtenir des informations sur le produit et le service
    - name: Obtenir des informations sur le Produit depuis l'API
CRM
      uri:
        url: "http://localhost:5000/crm/product/product_id//{{
product_id }}"
        method: GET
        headers:
          Authorization: "Bearer {{ access_token }}"
        return_content: yes
        validate_certs: no
      register: api_response_product

    - name: Obtenir des informations sur le Service depuis l'API
CRM
      uri:
        url: "http://localhost:5000/crm/service/service_id/{{
service_id }}"
        method: GET
        headers:
          Authorization: "Bearer {{ access_token }}"
        return_content: yes
        validate_certs: no
      register: api_response_service

    - name: Définir les faits de service et de package
      set_fact:
        service_uuid: "{{ api_response_service.json.service_uuid
}}"
        customer_id: "{{ api_response_service.json.customer_id }}"
        package_name: "{{ api_response_product.json.product_name
}}"
```

```
        monthly_cost: "{{ api_response_product.json.retail_cost
}}"
        wholesale_cost: "{{
api_response_product.json.wholesale_cost }}"

    # Obtenir le moyen de paiement par défaut du client
    - name: Obtenir les Méthodes de Paiement du Client depuis le
Contrôleur de Paiement
      uri:
        url: "http://localhost:5000/crm/payments/methods?
customer_id={{ customer_id }}"
        method: GET
        headers:
          Authorization: "Bearer {{ access_token }}"
        return_content: yes
        validate_certs: no
      register: api_response_payment_methods

    - name: Obtenir le payment_method_id par défaut de la réponse
      set_fact:
        payment_method_id: "{{ api_response_payment_methods.json |
json_query(query) }}"
      vars:
        query: "data[?is_default==`true`].payment_method_id | [0]"

    # ============================================================
    # FLUX DE PAIEMENT À DEUX PHASES
    # ============================================================

    - name: "Phase 1 : Autoriser le paiement (retenir des fonds)"
      uri:
        url: "http://localhost:5000/crm/payments/authorize/hold"
        method: POST
        headers:
          Content-Type: "application/json"
          Authorization: "Bearer {{ access_token }}"
        body_format: json
        body:
          {
            "customer_id": "{{ customer_id | int }}",
            "amount": "{{ monthly_cost | float }}",
            "currency": "{{ crm_config.currency | default('AUD')
}}",
            "payment_method_id": "{{ payment_method_id | int }}",
```

```
            "metadata": {
              "description": "{{ package_name }} sur {{
api_response_service.json.service_name }}",
              "service_id": "{{
api_response_service.json.service_id | int }}",
              "site_id": "{{ api_response_service.json.site_id |
int }}",
              "product_id": "{{ product_id }}",
              "user_id": "{{ (initiating_user | int) if
(initiating_user is defined and initiating_user is not none) else
omit }}",
              "title": "{{ package_name }}",
              "wholesale_cost": "{{ wholesale_cost | float }}",
              "invoice": true,
              "contract_days": "{{ days | int }}",
              "send_email": true
            }
          }
        return_content: yes
      register: api_response_authorization

    - name: Vérifier que l'autorisation a réussi
      assert:
        that:
          - api_response_authorization.status == 200
          - api_response_authorization.json.success == true

    - name: Stocker authorization_id pour capture/libération
      set_fact:
        authorization_id: "{{
api_response_authorization.json.data.authorization_id }}"

    # Phase 2 : Provisionnement OCS (enveloppé dans un bloc pour
rollback transactionnel)
    - block:
        - name: "Phase 2 : Exécuter l'action CGRateS"
          uri:
            url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
            method: POST
            body_format: json
            headers:
              Content-Type: "application/json"
              Authorization: "Bearer {{ access_token }}"
            body:
```

```
              {
                "id": "{{ 999999999 | random }}",
                "method": "APIerSv1.ExecuteAction",
                "params": [{
                  "Tenant": "{{ crm_config.ocs.ocsTenant }}",
                  "Account": "{{ service_uuid }}",
                  "ActionsId": "{{ cgr_action_name }}"
                }]
              }
            status_code: 200
          register: action_execute_response

        - name: Vérifier que la réponse de l'action est OK
          assert:
            that:
              - action_execute_response.status == 200
              - action_execute_response.json.result == "OK"

        # Phase 3 : Capture de Paiement - Finaliser la transaction
après un provisionnement réussi
        - name: "Phase 3 : Capturer le paiement autorisé"
          uri:
            url: "http://localhost:5000/crm/payments/capture/{{
authorization_id }}"
            method: POST
            headers:
              Content-Type: "application/json"
              Authorization: "Bearer {{ access_token }}"
            body_format: json
            body:
              {
                "metadata": {
                  "provisioning_status": "success",
                  "cgr_action": "{{ cgr_action_name }}"
                }
              }
            return_content: yes
          register: api_response_capture

        - name: Vérifier que la capture a réussi
          assert:
            that:
              - api_response_capture.status == 200
              - api_response_capture.json.success == true
```

```
      rescue:
        # Rollback de la Transaction : Annuler l'autorisation pour
libérer les fonds retenus
        - name: "Rollback : Libérer l'autorisation de paiement"
          uri:
            url: "http://localhost:5000/crm/payments/release/{{
authorization_id }}"
            method: POST
            headers:
              Content-Type: "application/json"
              Authorization: "Bearer {{ access_token }}"
            body_format: json
            body:
              {
                "metadata": {
                  "release_reason": "provisioning_failed",
                  "cgr_action": "{{ cgr_action_name }}"
                }
              }
            return_content: yes
          register: api_response_release

        - name: Terminer l'exécution du playbook après rollback
          fail:
            msg: "Le provisionnement OCS a échoué. L'autorisation
de paiement {{ authorization_id }} a été annulée. Client non
facturé."
```

# Points de Terminaison de l'API de Paiement

> **Routage Prioritaire du Portefeuille** : Le système de paiement optimise automatiquement les frais par carte en utilisant d'abord le solde du portefeuille. Si un client a 1 $ dans son portefeuille et achète un addon de 10 $, seulement 9 $ sont facturés à sa carte. Voir la section Routage Prioritaire du Portefeuille pour plus de détails.

# 1. Autoriser/Retenir le Paiement

**Point de Terminaison :** `POST /crm/payments/authorize/hold`

Ce point de terminaison place une retenue sur le moyen de paiement du client pour le montant spécifié. Les fonds sont réservés mais pas encore capturés.

**Comportement Prioritaire du Portefeuille** : L'autorisation calcule automatiquement le manque après vérification du solde du portefeuille. Si le solde du portefeuille est de 150 $ et que l'autorisation est de 500 $, seulement 350 $ sont autorisés sur la carte. Le portefeuille est débité au moment de la capture.

**Corps de la Requête :**

```
{
  "customer_id": 123,
  "amount": 49.99,
  "currency": "AUD",
  "payment_method_id": 456,
  "metadata": {
    "description": "Nom et description du package",
    "service_id": 789,
    "site_id": 12,
    "product_id": "34",
    "user_id": 5,
    "title": "Nom du Package",
    "wholesale_cost": 25.00,
    "invoice": true,
    "contract_days": 30,
    "send_email": true
  }
}
```

**Descriptions des Champs :**

- **customer_id** (requis) - ID du client à facturer
- **amount** (requis) - Montant à autoriser au format décimal
- **currency** (optionnel) - Code de la devise (par défaut à la valeur par défaut du système)

- **payment_method_id** (requis) - Moyen de paiement à facturer
- **metadata** (optionnel) - Données supplémentaires pour la création de la transaction :
  - **description** - Description de la transaction
  - **service_id** - Service facturé
  - **site_id** - Site associé au service
  - **product_id** - Produit provisionné
  - **user_id** - Utilisateur initiant la charge (optionnel)
  - **title** - Titre de la transaction
  - **wholesale_cost** - Votre coût (pour le suivi de la marge)
  - **invoice** - Si `true`, créer automatiquement la transaction lors de la capture
  - **contract_days** - Durée du contrat
  - **send_email** - Si `true`, envoyer une notification par e-mail

**Réponse :**


```
{
  "success": true,
  "message": "Paiement autorisé (retenue créée)",
  "data": {
    "authorization_id": 301,
    "vendor_authorization_id": "auth_xxxxx",
    "amount": 500.00,
    "currency": "USD",
    "status": "authorized",
    "wallet_balance": 150.00,
    "wallet_to_use": 150.00,
    "card_amount": 350.00,
    "message": "Carte autorisée pour 350 $ (recharge de
portefeuille). Le débit du portefeuille de 500 $ aura lieu lors de
la capture."
  }
}
```


**Important :**

- Sauvegardez l'`authorization_id` pour une utilisation dans les appels de capture ou de libération
- Notez que le `card_amount` montre que seul le manque a été autorisé
- Le solde du portefeuille est vérifié mais PAS débité jusqu'à la capture

## 2. Capturer le Paiement

**Point de Terminaison :** `POST /crm/payments/capture/{authorization_id}`

Ce point de terminaison finalise l'autorisation de paiement et facture le client. Si `invoice: true` a été défini dans les métadonnées d'autorisation, une transaction est automatiquement créée.

**Corps de la Requête :**

```
{
  "metadata": {
    "provisioning_status": "success",
    "cgr_action": "Action_Topup_Standard",
    "additional_info": "Toute autre donnée pertinente"
  }
}
```

**Réponse :**

```
{
  "success": true,
  "data": {
    "payment_id": "pay_xyz789",
    "transaction_id": 1234
  }
}
```

**Ce qui se passe :**

1. **Carte capturée** pour le montant de manque (si la carte a été autorisée)
2. **Portefeuille crédité** avec le montant capturé de la carte
3. **Portefeuille débité** pour le montant total du service

4. Si `invoice: true` était dans les métadonnées d'autorisation :
    - **Transaction de débit** créée (montant positif = charge)
    - **Facture** créée et liée à la transaction de débit
    - **Transaction de crédit** créée (montant négatif = paiement reçu)
    - Facture marquée comme PAYÉE (les débits et crédits se compensent à zéro)
    - La transaction apparaît dans la facturation du client
5. Un enregistrement de paiement est créé dans la base de données
6. Si `send_email: true`, le client reçoit un e-mail de facture

**Exemple** : Autorisation de 500 $ avec un solde de portefeuille de 150 $ :

- Carte capturée : 350 $
- Portefeuille crédité : +350 $ (portefeuille maintenant 500 $)
- Portefeuille débité : -500 $ (frais de service)
- Portefeuille final : 0 $

Voir Exemples de Routage Prioritaire du Portefeuille pour un flux détaillé.

## 3. Libérer l'Autorisation de Paiement

**Point de Terminaison :** `POST /crm/payments/release/{authorization_id}`

Ce point de terminaison annule une autorisation de paiement et libère les fonds retenus. Utilisez ceci dans des scénarios de secours/rollback lorsque le provisionnement échoue.

**Corps de la Requête :**

```
{
  "metadata": {
    "release_reason": "provisioning_failed",
    "error_details": "Échec de la création du compte OCS"
  }
}
```

**Réponse :**

```
{
  "success": true,
  "message": "Autorisation libérée"
}
```

**Ce qui se passe :**

1. **Autorisation de carte libérée** (si la carte a été autorisée)
2. Les fonds retenus sont retournés au crédit disponible du client
3. **Portefeuille NON débité** (puisque le débit n'a lieu qu'à la capture)
4. Le client n'est PAS facturé
5. Aucune transaction n'est créée

**Remarque** : Avec le routage prioritaire du portefeuille, aucun remboursement de portefeuille n'est nécessaire puisque le portefeuille n'est pas débité avant le moment de la capture.

# Création de Transaction Directe

Pour les frais qui ne nécessitent pas de traitement de paiement (frais d'installation, crédits manuels, ajustements), vous pouvez créer des transactions directement via l'API.

## Ajout d'une Transaction via l'API

**Point de Terminaison :** `PUT /crm/transaction/`

**Exemple du fichier `play_simple_service.yaml` :**

```
- name: Ajouter une Transaction de Coût d'Installation via l'API
  uri:
    url: "http://localhost:5000/crm/transaction/"
    method: PUT
    headers:
      Content-Type: "application/json"
      Authorization: "Bearer {{ access_token }}"
    body_format: json
    body:
      {
        "customer_id": {{ customer_id | int }},
        "service_id": {{ service_creation_response.json.service_id
| int }},
        "title": "{{ package_name }} - Coûts d'Installation",
        "description": "Coûts d'Installation pour {{
package_comment }}",
        "invoice_id": null,
        "wholesale_cost": {{
api_response_product.json.wholesale_setup_cost | float }},
        "retail_cost": "{{
api_response_product.json.retail_setup_cost | float }}"
      }
    return_content: yes
  register: api_response_transaction

- name: Vérifier que la réponse de la transaction est OK
  assert:
    that:
      - api_response_transaction.status == 200
```

**Champs du Corps de la Requête :**

- **customer_id** (requis) - ID du client
- **service_id** (optionnel) - ID du service pour lier la transaction
- **title** (requis) - Nom court de la transaction
- **description** (optionnel) - Description détaillée
- **invoice_id** (optionnel) - ID de la facture si déjà facturée (généralement null)
- **wholesale_cost** (optionnel) - Votre coût
- **retail_cost** (requis) - Coût pour le client

- **site_id** (optionnel) - ID du site pour lier la transaction
- **tax_percentage** (optionnel) - Taux de taxe en pourcentage

**Cas d'utilisation :**

- Frais d'installation lors du provisionnement de service
- Charges d'installation
- Crédits ou ajustements manuels
- Charges d'équipement
- Frais administratifs

**Remarque :** La création directe de transactions ne traite PAS le paiement - elle crée uniquement un enregistrement de facturation. La transaction apparaîtra comme non facturée et pourra être incluse dans de futures factures.

# Calcul des Charges Pro-Rata

Le chargement pro-rata vous permet de facturer les clients proportionnellement en fonction des périodes de facturation partielles. Cela est courant lorsque :

- Le client s'inscrit à mi-mois
- Le service est mis à niveau/déclassé en cours de cycle
- La facturation est calculée en fonction des jours utilisés

## Formule de Calcul Pro-Rata

```
pro_rata_charge = (monthly_cost × days_remaining) / days_in_month
```

## Exemple de Mise en Œuvre

Voici comment calculer une charge pro-rata dans un playbook :

```yaml
# Calculer la charge pro-rata pour un mois partiel
# Si le client s'inscrit le 15 et que la facturation est le 1er,
# facturer proportionnellement pour les jours restants

- name: Obtenir le jour actuel du mois
  command: "date +%d"
  register: current_day

- name: Obtenir le nombre total de jours dans le mois actuel
  command: "date -d 'last day of this month' +%d"
  register: days_in_month

- name: Obtenir le dernier jour du mois
  command: "date -d 'last day of this month' +%Y-%m-%d"
  register: last_day_of_month

- name: Calculer les jours restants dans le mois
  set_fact:
    days_remaining: "{{ (days_in_month.stdout | int) -
(current_day.stdout | int) + 1 }}"

- name: Calculer le coût pro-rata
  set_fact:
    pro_rata_cost: "{{ ((monthly_cost | float) * (days_remaining |
float) / (days_in_month.stdout | float)) | round(2) }}"

- name: Afficher les détails du calcul
  debug:
    msg:
      - "Coût mensuel : ${{ monthly_cost }}"
      - "Jours dans le mois : {{ days_in_month.stdout }}"
      - "Jours restants : {{ days_remaining }}"
      - "Charge pro-rata : ${{ pro_rata_cost }}"

# Utiliser pro_rata_cost dans l'autorisation ou la création de
transaction
- name: "Autoriser le paiement pro-rata"
  uri:
    url: "http://localhost:5000/crm/payments/authorize/hold"
    method: POST
    headers:
      Content-Type: "application/json"
      Authorization: "Bearer {{ access_token }}"
```

```
    body_format: json
    body:
      {
        "customer_id": "{{ customer_id | int }}",
        "amount": "{{ pro_rata_cost | float }}",
        "currency": "{{ crm_config.currency | default('AUD') }}",
        "payment_method_id": "{{ payment_method_id | int }}",
        "metadata": {
          "title": "{{ package_name }} (Pro-rata {{ days_remaining
}} jours)",
          "description": "Charge pro-rata pour {{ days_remaining
}}/{{ days_in_month.stdout }} jours de {{ package_name }}",
          "service_id": "{{ service_id | int }}",
          "invoice": true
        }
      }
```

## Pro-Rata à Partir d'une Date de Début Personnalisée

Si vous devez calculer le pro-rata à partir d'une date de début spécifique jusqu'à une date de facturation :

```
- name: Définir la date de début personnalisée et la date de
facturation
  set_fact:
    service_start_date: "2024-01-15"
    next_billing_date: "2024-02-01"

- name: Calculer les jours entre les dates
  shell: |
    echo $(( ( $(date -d "{{ next_billing_date }}" +%s) - $(date -
d "{{ service_start_date }}" +%s) ) / 86400 ))
  register: days_until_billing

- name: Obtenir les jours dans la période de facturation
(généralement 30)
  set_fact:
    billing_period_days: 30

- name: Calculer le coût pro-rata
  set_fact:
    pro_rata_cost: "{{ ((monthly_cost | float) *
(days_until_billing.stdout | float) / (billing_period_days |
float)) | round(2) }}"

- name: Afficher le calcul
  debug:
    msg:
      - "Date de début : {{ service_start_date }}"
      - "Prochaine facturation : {{ next_billing_date }}"
      - "Jours jusqu'à la facturation : {{
days_until_billing.stdout }}"
      - "Charge pro-rata : ${{ pro_rata_cost }}"
```

## Scénarios d'Exemple Pro-Rata

### Scénario 1 : Inscription à Mi-Mois

- Le client s'inscrit le 15 janvier
- Coût mensuel : 60,00 $
- Jours en janvier : 31
- Jours restants : 17 (15 au 31 inclus)

- Charge pro-rata : `$60.00 × 17 ÷ 31 = $32.90`

**Scénario 2 : Mise à Niveau de Service**

- Le client met à niveau le 10ème jour du cycle de facturation
- Ancien plan : 30 $/mois
- Nouveau plan : 50 $/mois
- Jours dans le cycle : 30
- Jours restants : 21
- Différence : 20 $/mois
- Charge pro-rata : `$20.00 × 21 ÷ 30 = $14.00`

# Meilleures Pratiques

## 1. Utiliser Toujours le Modèle Bloc/Rescue

Enveloppez la capture de paiement et le provisionnement dans un bloc/rescue pour garantir le rollback :

```
- block:
    # Tâches de provisionnement
    - name: Provisionner le service
      uri: ...

    # Capturer le paiement uniquement après succès
    - name: Capturer le paiement
      uri: ...

  rescue:
    # Libérer l'autorisation si quelque chose échoue
    - name: Libérer l'autorisation de paiement
      uri: ...

    - name: Échouer le playbook
      fail:
        msg: "Le provisionnement a échoué, le client n'est pas
facturé"
```

## 2. Valider Toutes les Réponses API

Vérifiez toujours que les opérations critiques ont réussi :

```
- name: Autoriser le paiement
  uri: ...
  register: api_response_authorization

- name: Vérifier que l'autorisation a réussi
  assert:
    that:
      - api_response_authorization.status == 200
      - api_response_authorization.json.success == true
    fail_msg: "L'autorisation de paiement a échoué : {{
api_response_authorization.json }}"
```

## 3. Stocker l'ID d'Autorisation

Sauvegardez toujours l'authorization_id pour une utilisation dans la capture/libération :

```
- name: Stocker authorization_id pour capture/libération
  set_fact:
    authorization_id: "{{
api_response_authorization.json.data.authorization_id }}"
```

## 4. Utiliser Efficacement les Métadonnées

Incluez des métadonnées complètes dans les demandes d'autorisation :

```
metadata:
  description: "Description claire de ce pour quoi le client est
facturé"
  service_id: "{{ service_id | int }}"
  product_id: "{{ product_id }}"
  user_id: "{{ initiating_user | int }}"
  title: "Titre court pour la transaction"
  wholesale_cost: "{{ wholesale_cost | float }}"
  invoice: true  # Créer automatiquement la transaction lors de la
capture
  send_email: true  # Envoyer une notification au client
```

## 5. Arrondir les Valeurs Monétaires

Arrondissez toujours les calculs monétaires à 2 décimales :

```
- name: Calculer le coût avec arrondi
  set_fact:
    final_cost: "{{ (base_cost | float * multiplier | float) |
round(2) }}"
```

## 6. Gérer les Méthodes de Paiement Manquantes

Vérifiez si le client a un moyen de paiement par défaut avant d'essayer d'autoriser :

```yaml
- name: Obtenir le moyen de paiement par défaut
  set_fact:
    payment_method_id: "{{ api_response_payment_methods.json |
json_query(query) }}"
  vars:
    query: "data[?is_default==`true`].payment_method_id | [0]"

- name: Vérifier que le moyen de paiement existe
  assert:
    that:
      - payment_method_id is defined
      - payment_method_id != ""
      - payment_method_id != None
    fail_msg: "Aucun moyen de paiement par défaut trouvé pour le
client {{ customer_id }}"
```

# Modèles Communs

## Modèle 1 : Provisionnement de Service Payant

Pour les services nécessitant un paiement immédiat :

1. Obtenez le moyen de paiement du client
2. Autorisez le paiement
3. Provisionnez le service dans OCS/CGRateS
4. Créez un enregistrement de service dans le CRM
5. Capturez le paiement
6. En cas d'échec : Libérez l'autorisation et faites un rollback

Voir `play_topup_charge_then_action.yaml` pour un exemple complet.

## Modèle 2 : Service Gratuit avec Frais d'Installation

Pour les services qui sont gratuits mais ont un coût d'installation unique :

1. Provisionnez le service

2. Créez un enregistrement de service
3. Ajoutez directement la transaction de frais d'installation (pas de traitement de paiement)
4. Les frais d'installation apparaissent sur la prochaine facture

Voir `play_simple_service.yaml` aux lignes 202-232 pour un exemple complet.

## Modèle 3 : Topup/Addon Gratuit

Pour les topups gratuits qui ne nécessitent pas de paiement :

1. Obtenez des informations sur le service
2. Exécutez l'action CGRateS
3. Mettez à jour les dates de service
4. Aucun paiement ou création de transaction nécessaire

## Modèle 4 : Charges Récurrentes via ActionPlan

Pour les charges récurrentes automatiques :

1. Créez une Action avec `*http_post` vers le point de terminaison de provisionnement
2. Créez un ActionPlan avec un timing `*monthly`
3. Assignez l'ActionPlan au compte
4. CGRateS appellera automatiquement le point de terminaison chaque mois
5. Le playbook du point de terminaison gère le traitement des paiements

# Dépannage

## Échec de l'Autorisation

**Symptôme :** Le point de terminaison d'autorisation retourne une erreur

**Causes Courantes :**

- Le moyen de paiement n'existe pas ou est invalide

- Fonds insuffisants
- Moyen de paiement expiré
- Mismatch d'ID client

**Solution :** Vérifiez l'état du moyen de paiement et le solde du client.

## Échec de la Capture Après Provisionnement Réussi

**Symptôme :** Service provisionné mais échec de la capture de paiement

**Problème :** Il s'agit d'un état d'échec critique - le service est actif mais le client n'est pas facturé

**Solution :**

- L'autorisation peut avoir expiré (généralement 7 jours)
- Vérifiez que l'autorisation est toujours valide avant d'essayer de capturer
- Mettez en œuvre une surveillance pour les captures échouées
- Une intervention manuelle peut être nécessaire

## Transaction Non Créée Après Capture

**Symptôme :** Paiement capturé mais aucune transaction dans la facturation

**Cause :** `invoice: true` n'a pas été défini dans les métadonnées d'autorisation

**Solution :** Soit :

- Définir `invoice: true` dans les métadonnées d'autorisation, OU
- Créer manuellement une transaction après une capture réussie

## Calcul Pro-Rata Incorrect

**Symptôme :** Les charges pro-rata ne correspondent pas aux valeurs attendues

**Problèmes Courants :**

- Erreurs de décalage d'un jour dans le comptage des jours (inclure/exclure les dates de début/fin)
- Mauvais mois utilisé pour le comptage des jours
- Erreurs d'arrondi

**Solution :**

- Utilisez des plages de dates inclusives (incluez à la fois le jour de début et le jour de fin)
- Arrondissez toujours à 2 décimales
- Testez les calculs avec des valeurs connues
- Documentez quelles dates sont incluses dans les calculs

# Remboursements et Gestion des Erreurs

## Options de Remboursement

Le système de paiement prend en charge deux types de remboursements :

**1. Remboursement au Moyen de Paiement** - Argent retourné à la carte/PayPal d'origine

```
- name: Rembourser le paiement à la carte du client
  uri:
    url: "http://localhost:5000/crm/payments/refund"
    method: POST
    headers:
      Content-Type: "application/json"
      Authorization: "Bearer {{ access_token }}"
    body_format: json
    body:
      {
        "transaction_id": "{{ vendor_transaction_id }}",
        "vendor": "stripe",
        "amount": "{{ refund_amount | float }}",
        "reason": "customer_request"
      }
```

**2. Crédit au Portefeuille** - Solde immédiat pour de futurs achats (géré automatiquement pour les scénarios d'erreur)

Pour les échecs de provisionnement, le système crédite automatiquement le portefeuille au lieu de rembourser la carte pour :

- Éviter les frais de remboursement
- Fournir une disponibilité immédiate pour une nouvelle tentative
- Améliorer l'expérience client

Voir Options de Remboursement pour des détails complets.

## Support des Fournisseurs

Le système de paiement est **indépendant des fournisseurs** et prend actuellement en charge :

- □ Stripe (cartes, ACH)
- □ PayPal (comptes PayPal, cartes)

De nouveaux fournisseurs de paiement (Square, Adyen, Braintree, etc.) peuvent être ajoutés sans modifier les playbooks.

**Voir aussi :**

- Architecture Modulaire - Comment fonctionne l'abstraction des fournisseurs
- Configuration des Fournisseurs - Configuration et clés API Stripe/PayPal

# Documentation Connexe

## Guides Spécifiques aux Playbooks

- `concepts_ansible.md` - Modèles et structure générale des playbooks
- `concepts_provisioning.md` - Vue d'ensemble du système de provisionnement

## Documentation du Système de Paiement

- **Guide de l'API du Système de Paiement** - Référence API complète, routage de portefeuille, remboursements
  - API des Méthodes de Paiement
  - API des Flux de Paiement
  - Routage Prioritaire du Portefeuille
  - Options de Remboursement
  - Architecture Modulaire
- `payments_transaction.md` - Gestion des transactions et charges manuelles
- `payments_process.md` - Traitement des paiements et des factures
- `basics_payment.md` - Méthodes de paiement et facturation client

# Contrôle d'Accès Basé sur les Rôles

## Rôles, Permissions et Utilisateurs dans OmniCRM

OmniCRM utilise le **contrôle d'accès basé sur les rôles (RBAC)** : les personnes (Utilisateurs) se voient attribuer un ou plusieurs Rôles, et chaque Rôle est un ensemble de Permissions. Les Permissions sont la plus petite unité d'accès (par exemple, *view_customer*, *create_inventory*). L'accès effectif d'un utilisateur est l'**union** des permissions de tous les rôles attribués.

# Objectif

Le RBAC permet :

1. **Protection des Données** : Les utilisateurs ne voient et ne font que ce qu'ils sont autorisés à faire.
2. **Adaptation Opérationnelle** : Les rôles reflètent les fonctions de travail (Admin, Support, Finance, Administrateur Client).
3. **Administration Simple** : Accorder l'accès en attribuant des rôles. Éviter la microgestion par utilisateur.
4. **Isolation des Locataires** : Les permissions "voir ses propres ..." limitent la visibilité aux données clients/locataires d'un utilisateur.

# Comment les Utilisateurs, Rôles et Permissions S'Intègrent

- **Utilisateurs** : Personnes réelles qui se connectent à OmniCRM.
- **Permissions** : Capacités atomiques (par exemple, *view_customer*, *delete_product*).
- **Rôles** : Ensembles nommés de permissions (par exemple, *Admin*, *Support*, *Finance*).
- **Attribution** : Les utilisateurs reçoivent un ou plusieurs rôles. Les permissions s'agrègent.

L'authentification prouve *qui vous êtes* (JWT, clé API ou IP sur liste blanche). L'autorisation (rôles/permissions) décide *ce que vous pouvez faire*.

# Gestion des Utilisateurs

Le système de gestion des utilisateurs d'OmniCRM permet aux administrateurs de créer et de gérer des utilisateurs du personnel (administrateurs, agents de service client), de visualiser et de modifier les rôles des utilisateurs, de réinitialiser les mots de passe, de gérer l'authentification à deux facteurs et de contrôler l'accès des utilisateurs.

## Types d'Utilisateurs

**Utilisateurs Clients** - Créés via auto-inscription ou par des administrateurs. Attribués automatiquement au rôle "Client". Ces utilisateurs accèdent au portail d'auto-assistance pour gérer leurs services, visualiser leur utilisation, payer des factures, etc.

**Utilisateurs du Personnel** - Créés par des administrateurs avec les permissions appropriées. Peuvent se voir attribuer des rôles comme Admin, Support, Finance, etc. Ces utilisateurs accèdent à l'interface CRM pour gérer les clients, provisionner des services, gérer la facturation, etc.

**Utilisateurs Administratifs** - Utilisateurs avec la permission `admin`. Ont un accès complet au système, y compris la gestion des utilisateurs, la gestion des rôles et tous les points de terminaison protégés.

L'utilisateur administratif initial est créé par l'équipe Omnitouch lors du déploiement du système.

# Ajout de Nouveaux Utilisateurs (Administrateurs et CSA)

Les administrateurs peuvent créer de nouveaux utilisateurs du personnel via l'interface Web ou l'API.

## Via l'Interface Web

1. **Naviguer vers Utilisateurs & Rôles** - Accéder à l'interface de gestion des utilisateurs depuis le menu d'administration.
2. **Cliquer sur "Ajouter un Utilisateur"** - Ouvre le formulaire de création d'utilisateur.

3. **Remplir les Détails de l'Utilisateur :**
    - **Nom d'utilisateur** - Nom d'utilisateur unique pour la connexion (obligatoire)

- **Email** - Adresse email de l'utilisateur (obligatoire, doit être unique)
- **Mot de passe** - Mot de passe temporaire (obligatoire, l'utilisateur doit le changer lors de la première connexion)
- **Prénom** - Prénom de l'utilisateur (obligatoire)
- **Deuxième Prénom** - Deuxième prénom de l'utilisateur (facultatif)
- **Nom de Famille** - Nom de famille de l'utilisateur (obligatoire)
- **Numéro de Téléphone** - Numéro de téléphone de contact (facultatif)
- **Rôle** - Sélectionner un ou plusieurs rôles à attribuer (obligatoire)
- **Contact Client** - Lier éventuellement l'utilisateur à un enregistrement de contact client (pour les utilisateurs clients)

4. **Cliquer sur "Créer un Utilisateur"** - L'utilisateur est créé et peut immédiatement se connecter avec les identifiants fournis.
5. **L'utilisateur reçoit une notification** - Envoyer éventuellement un email de bienvenue avec des instructions de connexion.

**Meilleures Pratiques :**

- Utiliser un mot de passe temporaire comme `TempP@ssw0rd!` et exiger que l'utilisateur le change lors de la première connexion.
- Attribuer des rôles appropriés en fonction de la fonction de travail (voir Conceptions de Rôles Typiques ci-dessous).
- Activer l'authentification à deux facteurs pour tout le personnel administratif et de support.
- Lier les utilisateurs clients à leur enregistrement de contact client pour un bon ciblage des données.

# Via l'API

Créer un utilisateur par programmation :

**Point de terminaison :** `POST /auth/users`

**Permission Requise :** `admin`

**Corps de la Requête :**

```
{
  "username": "john.smith",
  "email": "john.smith@company.com",
  "password": "TempP@ssw0rd!",
  "first_name": "John",
  "middle_name": "D",
  "last_name": "Smith",
  "phone_number": "+61412345678",
  "role": "Support"
}
```

**Réponse :**

```
{
  "id": 123,
  "username": "john.smith",
  "email": "john.smith@company.com",
  "first_name": "John",
  "last_name": "Smith",
  "roles": ["Support"],
  "created": "2025-01-04T10:30:00Z"
}
```

**Attribution de Rôles Multiples :**

Les utilisateurs peuvent avoir plusieurs rôles. Les permissions sont additives (union de toutes les permissions de rôle attribuées).

Pour attribuer plusieurs rôles, incluez-les dans la requête :

```
{
  "username": "jane.doe",
  "email": "jane.doe@company.com",
  "password": "TempP@ssw0rd!",
  "first_name": "Jane",
  "last_name": "Doe",
  "role": "Support,Finance"
}
```

Ou utilisez le point de terminaison d'attribution de rôle après la création de l'utilisateur :

```
POST /auth/roles/{role_id}/users/{user_id}
```

# Visualiser et Rechercher des Utilisateurs

**Lister Tous les Utilisateurs (Admin) :**

`GET /auth/users`

Renvoie une liste paginée de tous les utilisateurs avec leurs rôles et informations de base.

**Rechercher des Utilisateurs :**

`GET /auth/users/search?search={query}&filters={"role":["Support"]}&page=1&per_page=50`

Filtrer par :

- Nom de rôle
- Domaine email
- Statut actif/supprimé
- Statut 2FA activé
- Date de dernière connexion

**Obtenir un Utilisateur Spécifique :**

`GET /auth/users/{user_id}`

Renvoie les détails complets de l'utilisateur, y compris :

- Informations personnelles
- Rôles attribués et permissions effectives

- Statut 2FA
- Dernière connexion et informations de session
- Contact client lié (le cas échéant)

# Création et Gestion des Rôles

Les rôles sont des collections de permissions qui peuvent être attribuées aux utilisateurs. Au lieu d'attribuer des permissions individuellement à chaque utilisateur, vous créez des rôles qui regroupent des permissions connexes et attribuez ces rôles aux utilisateurs.

## Création d'un Nouveau Rôle

**Via l'Interface Web :**

1. Naviguer vers **Utilisateurs & Rôles → Onglet Rôles**
2. Cliquer sur **"Créer un Rôle"**
3. Entrer les détails du rôle :
   - **Nom** - Nom court et descriptif (par exemple, "Tier2_Support")
   - **Description** - Expliquer l'objectif et les responsabilités du rôle
4. Cliquer sur **"Créer"**
5. Le rôle est créé sans permissions ; ajouter des permissions à l'étape suivante.

**Via l'API :**

**Point de terminaison :** `POST /auth/roles`

**Permission Requise :** `admin`

**Requête :**

```
{
  "name": "Tier2_Support",
  "description": "Équipe de support de niveau 2 avec accès de
provisionnement élevé"
}
```

**Réponse :**

```
{
  "id": 45,
  "name": "Tier2_Support",
  "description": "Équipe de support de niveau 2 avec accès de
provisionnement élevé",
  "permissions": [],
  "users": []
}
```

## Ajout de Permissions à un Rôle

Une fois qu'un rôle est créé, attribuez des permissions pour définir ce que les utilisateurs avec ce rôle peuvent faire.

**Via l'Interface Web :**

1. Naviguer vers **Utilisateurs & Rôles → Onglet Rôles**
2. Cliquer sur le nom du rôle pour voir les détails
3. Dans la section **Permissions**, cliquer sur **"Ajouter une Permission"**
4. Sélectionner une ou plusieurs permissions dans la liste
5. Cliquer sur **"Ajouter"** - Les permissions sont immédiatement attribuées.

**Via l'API :**

**Point de terminaison :** `POST /auth/roles/{role_id}/permissions`

**Requête :**

```
{
  "permission_id": 123
}
```

Ou ajouter plusieurs permissions :

```
{
  "permission_ids": [123, 124, 125]
}
```

**Exemple : Création d'un Rôle "Spécialiste en Provisionnement"**

Ce rôle peut voir les clients, gérer les services et provisionner :

1. Créer le rôle :

```
POST /auth/roles
{
  "name": "Provisioning_Specialist",
  "description": "Peut provisionner des services et gérer les
services clients"
}
```

2. Ajouter des permissions :

```
POST /auth/roles/45/permissions
{
  "permission_ids": [
    1,   # view_customer
    20,  # view_customer_service
    21,  # create_customer_service
    22,  # update_customer_service
    30,  # view_provision
    31,  # create_provision
    40,  # view_inventory
    50,  # view_product
  ]
}
```

# Suppression de Permissions d'un Rôle

**Via l'Interface Web :**

1. Naviguer vers les détails du rôle
2. Dans la liste **Permissions**, cliquer sur le **"X"** ou le bouton **"Supprimer"** à côté de la permission
3. Confirmer la suppression

**Via l'API :**

**Point de terminaison :** `DELETE /auth/roles/{role_id}/permissions/{permission_id}`

**Exemple :**

```
DELETE /auth/roles/45/permissions/31
```

Cela supprime la permission `create_provision` du rôle.

# Édition des Détails du Rôle

Mettre à jour le nom ou la description du rôle :

**Via l'Interface Web :**

1. Naviguer vers **Utilisateurs & Rôles → Onglet Rôles**
2. Cliquer sur le rôle à éditer
3. Modifier le nom ou la description du rôle
4. Cliquer sur **"Enregistrer"**

**Via l'API :**

**Point de terminaison :** `PUT /auth/roles/{role_id}`

```
{
  "name": "Senior_Support",
  "description": "Équipe de support senior avec accès complet aux
clients"
}
```

## Suppression d'un Rôle

**Avertissement :** La suppression d'un rôle le retire de tous les utilisateurs attribués. Assurez-vous que les utilisateurs ont des rôles alternatifs ou ils perdront l'accès.

**Via l'API :**

`DELETE /auth/roles/{role_id}`

**Meilleure Pratique :** Au lieu de supprimer, envisagez d'archiver ou de renommer les rôles qui ne sont plus nécessaires.

# Attribution de Rôles aux Utilisateurs

**Lors de la Création d'un Utilisateur :**

Inclure le rôle dans la requête de création d'utilisateur (voir "Ajout de Nouveaux Utilisateurs" ci-dessus).

**Pour les Utilisateurs Existants :**

**Via l'Interface Web :**

1. Naviguer vers **Utilisateurs & Rôles → Onglet Utilisateurs**
2. Cliquer sur l'utilisateur à éditer
3. Dans la section **Rôles**, sélectionner/désélectionner des rôles
4. Cliquer sur **"Enregistrer"**

**Via l'API :**

Mettre à jour les rôles de l'utilisateur :

**Point de terminaison :** `PUT /auth/users/{user_id}`

```
{
  "role": "Support,Finance"
}
```

Ou attribuer un rôle unique à un utilisateur via le point de terminaison de rôle :

**Point de terminaison :** `POST /auth/roles/{role_id}/users/{user_id}`

## Visualiser les Attributions de Rôle

**Tous les utilisateurs dans un rôle :**

`GET /auth/roles/{role_id}/users`

Renvoie la liste de tous les utilisateurs attribués à ce rôle.

**Tous les rôles pour un utilisateur :**

`GET /auth/users/{user_id}`

La réponse inclut un tableau `roles` avec tous les rôles attribués.

# Gestion des Mots de Passe des Utilisateurs

OmniCRM fournit plusieurs méthodes pour la gestion des mots de passe selon le contexte.

## Réinitialisation de Mot de Passe en Libre-Service par l'Utilisateur

Les utilisateurs qui ont oublié leur mot de passe peuvent le réinitialiser eux-mêmes via la page de connexion :

1. **Cliquer sur "Mot de Passe Oublié"** sur la page de connexion
2. **Entrer l'adresse email** - Le système envoie un email de réinitialisation de mot de passe
3. **Vérifier l'email** - L'email contient un lien de réinitialisation sécurisé avec un token (valide pendant 1 heure)
4. **Cliquer sur le lien** - Ouvre le formulaire de réinitialisation de mot de passe

5. **Entrer un nouveau mot de passe** - Doit respecter les exigences de complexité du mot de passe :
   - Minimum 8 caractères
   - Au moins une lettre majuscule
   - Au moins une lettre minuscule
   - Au moins un chiffre
   - Au moins un caractère spécial
6. **Soumettre** - Le mot de passe est immédiatement mis à jour ; l'utilisateur peut se connecter avec le nouveau mot de passe.

**Flux API :**

1. **Demander la réinitialisation :**

   **Point de terminaison :** `POST /auth/forgot_password`

   ```
   {
     "email": "user@example.com"
   }
   ```

   Le système génère un token de réinitialisation et envoie un email.

2. **Réinitialiser avec le token :**

   **Point de terminaison :** `POST /auth/reset_password`

   ```
   {
     "token": "abc123...",
     "new_password": "NewSecureP@ssw0rd!"
   }
   ```

## Réinitialisation de Mot de Passe par l'Administrateur

Les administrateurs peuvent réinitialiser le mot de passe d'un utilisateur sans nécessiter de vérification par email. Cela définit un mot de passe temporaire que l'utilisateur doit changer lors de la prochaine connexion.

**Via l'Interface Web :**

1. Naviguer vers **Utilisateurs & Rôles → Onglet Utilisateurs**
2. Trouver l'utilisateur et cliquer sur le bouton **"Réinitialiser le Mot de Passe"**
3. Entrer un mot de passe temporaire
4. Cliquer sur **"Réinitialiser"**
5. Informer l'utilisateur de son mot de passe temporaire (via un canal sécurisé)
6. L'utilisateur doit changer le mot de passe lors de la prochaine connexion.

**Via l'API :**

**Point de terminaison :** `POST /auth/users/{user_id}/admin_reset_password`

**Permission Requise :** `admin`

**Requête :**

```
{
  "new_password": "TempP@ssw0rd!",
  "force_change": true
}
```

**Paramètres :**

- `new_password` - Mot de passe temporaire à définir
- `force_change` (facultatif) - Si vrai, l'utilisateur doit changer le mot de passe lors de la prochaine connexion.

# Changement de Mot de Passe par l'Utilisateur

Les utilisateurs authentifiés peuvent changer leur propre mot de passe depuis leur profil :

**Point de terminaison :** `POST /auth/change_password`

**Requête :**

```
{
  "current_password": "OldP@ssw0rd!",
  "new_password": "NewSecureP@ssw0rd!"
}
```

Le système valide le mot de passe actuel avant de permettre le changement.

### Sécurité des Mots de Passe

- Les mots de passe sont hachés en utilisant bcrypt (sécurité werkzeug)
- Jamais stockés en texte clair
- Les tokens de réinitialisation expirent après 1 heure
- Les tentatives de connexion échouées peuvent déclencher le verrouillage du compte (configurable)
- Le suivi de l'historique des mots de passe empêche la réutilisation (configurable)
- Exigences de complexité appliquées

## Gestion de l'Authentification à Deux Facteurs (2FA)

OmniCRM prend en charge l'authentification à deux facteurs basée sur TOTP pour une sécurité renforcée. Les administrateurs peuvent activer, désactiver et réinitialiser la 2FA pour les utilisateurs.

# Activation de la 2FA pour un Utilisateur

**Via l'Interface Web :**

1. Naviguer vers **Utilisateurs & Rôles → Onglet Utilisateurs**
2. Cliquer sur l'utilisateur pour voir les détails
3. Dans la section **Sécurité**, cliquer sur **"Activer la 2FA"**
4. Le système génère :
    - Secret TOTP (QR code affiché)
    - 10 codes de sauvegarde (à usage unique)
5. L'utilisateur scanne le QR code avec l'application d'authentification (Google Authenticator, Authy, etc.)
6. L'utilisateur entre le code de vérification de l'application pour confirmer la configuration
7. L'utilisateur sauvegarde les codes de sauvegarde dans un endroit sécurisé
8. La 2FA est maintenant activée ; requise pour toutes les connexions futures.

**Via l'API :**

1. **Générer le secret TOTP :**

   **Point de terminaison :** `POST /2fa/enable/user/{user_id}`

   **Réponse :**

```
{
  "totp_secret": "JBSWY3DPEHPK3PXP",
  "qr_code_url": "otpauth://totp/OmniCRM:user@example.com?
secret=JBSWY3DPEHPK3PXP&issuer=OmniCRM",
  "backup_codes": [
    "12345678",
    "23456789",
    "34567890",
    ...
  ]
}
```

2. **Vérifier la configuration :**

   **Point de terminaison :** `POST /2fa/verify-setup/user/{user_id}`

   ```
   {
     "code": "123456"
   }
   ```

   Renvoie `{"verified": true}` en cas de succès.

# Flux de Connexion 2FA

Une fois la 2FA activée, le processus de connexion change :

1. L'utilisateur entre son nom d'utilisateur et son mot de passe
2. Le système valide les identifiants
3. Si valide, demande le code 2FA
4. L'utilisateur entre le code de l'application d'authentification OU le code de sauvegarde
5. Le système vérifie le code
6. En cas de succès, l'utilisateur est connecté

**Codes de Sauvegarde :**

- 10 codes générés lors de la configuration de la 2FA
- À usage unique seulement (consommés après utilisation)
- Utilisés si l'application d'authentification n'est pas disponible
- Peuvent être régénérés par l'utilisateur ou l'administrateur

## Vérification du Code 2FA

**Point de terminaison :** `POST /2fa/verify/user/{user_id}`

```
{
  "code": "123456"
}
```

Accepte les deux :

- **Code TOTP** (6 chiffres de l'application d'authentification)
- **Code de Sauvegarde** (8 chiffres de la liste de codes de sauvegarde)

# Régénération des Codes de Sauvegarde

Si un utilisateur épuise ses codes de sauvegarde ou les perd, générez-en de nouveaux :

**Via l'Interface Web :**

1. Naviguer vers les détails de l'utilisateur
2. Cliquer sur **"Régénérer les Codes de Sauvegarde"**
3. Afficher ou envoyer de nouveaux codes à l'utilisateur
4. Les anciens codes sont invalidés

**Via l'API :**

**Point de terminaison :** `POST /2fa/backup-codes/regenerate/user/{user_id}`

**Réponse :**

```
{
  "backup_codes": [
    "98765432",
    "87654321",
    "76543210",
    ...
  ]
}
```

# Réinitialisation de la 2FA par l'Administrateur

Si un utilisateur perd l'accès à son application d'authentification et à tous les codes de sauvegarde, un administrateur peut désactiver et réactiver la 2FA.

**Via l'Interface Web :**

1. Naviguer vers **Utilisateurs & Rôles → Onglet Utilisateurs**
2. Cliquer sur l'utilisateur
3. Cliquer sur le bouton **"Réinitialiser la 2FA"**

4. Confirmer la réinitialisation
5. La 2FA est désactivée ; l'utilisateur peut se connecter uniquement avec le mot de passe.
6. Guider l'utilisateur pour configurer à nouveau la 2FA avec un nouveau secret.

**Via l'API :**

**Point de terminaison :** `POST /2fa/admin/disable/user/{user_id}`

**Permission Requise :** `admin`

Cela désactive complètement la 2FA pour l'utilisateur :

- Secret TOTP effacé
- Codes de sauvegarde effacés
- Le drapeau `is_2fa_enabled` est défini sur faux

L'utilisateur peut ensuite réactiver la 2FA pour obtenir un nouveau secret et de nouveaux codes de sauvegarde.

## Réinitialisation de la 2FA en Libre-Service par l'Utilisateur (Nouvel Appareil)

Si un utilisateur obtient un nouvel appareil mais a toujours accès aux codes de sauvegarde :

**Point de terminaison :** `POST /2fa/reset-for-new-device/user/{user_id}`

```
{
  "backup_code": "12345678"
}
```

Le système valide le code de sauvegarde, puis génère un nouveau secret TOTP et de nouveaux codes de sauvegarde. L'utilisateur peut configurer l'application d'authentification sur le nouvel appareil.

## Meilleures Pratiques pour la 2FA

- **Exiger la 2FA pour tout le personnel administratif et de support**
- **Stocker les codes de sauvegarde en toute sécurité** (gestionnaire de mots de passe ou note sécurisée)
- **Régénérer les codes de sauvegarde** après en avoir utilisé plusieurs
- **Utiliser des applications d'authentification réputées** (Google Authenticator, Authy, Microsoft Authenticator)
- **Documenter les procédures de réinitialisation de la 2FA** pour le personnel de support
- **Auditer l'utilisation de la 2FA** - surveiller quels utilisateurs ont la 2FA activée

# Mise à Jour des Informations Utilisateur

Les administrateurs peuvent mettre à jour les détails des utilisateurs à tout moment.

**Via l'Interface Web :**

1. Naviguer vers **Utilisateurs & Rôles → Onglet Utilisateurs**
2. Cliquer sur l'utilisateur à éditer
3. Modifier tous les champs éditables :
    - Prénom, deuxième prénom, nom de famille
    - Adresse email (nécessite vérification)
    - Numéro de téléphone
    - Rôles
    - Lien avec le contact client
4. Cliquer sur **"Enregistrer"**

**Via l'API :**

**Point de terminaison :** `PUT /auth/users/{user_id}`

```
{
  "first_name": "Jane",
  "last_name": "Doe-Smith",
  "email": "jane.doesmith@newcompany.com",
  "phone_number": "+61498765432",
  "role": "Support,Finance"
}
```

**Changements d'Email :**

Lorsque l'email est changé, le nouvel email est marqué comme en attente jusqu'à vérification :

- Le champ `pending_email` stocke le nouvel email
- Un email de vérification est envoyé à la nouvelle adresse
- L'utilisateur clique sur le lien pour vérifier
- Le champ `email` est mis à jour avec la nouvelle valeur
- Le drapeau `email_verified` est défini sur vrai

# Suppression d'Utilisateurs

OmniCRM utilise des **suppressions douces** pour les utilisateurs - ils sont marqués comme supprimés mais ne sont pas retirés de la base de données. Cela préserve les pistes d'audit et les données historiques.

## Suppression d'un Utilisateur

**Via l'Interface Web :**

1. Naviguer vers **Utilisateurs & Rôles → Onglet Utilisateurs**
2. Trouver l'utilisateur à supprimer
3. Cliquer sur le bouton **"Supprimer"**

4. Confirmer la suppression
5. L'utilisateur est immédiatement déconnecté et ne peut plus se reconnecter.

**Via l'API :**

**Point de terminaison :** `DELETE /auth/users/{user_id}`

**Permission Requise :** `admin`

**Que se Passe-t-il :**

- Le drapeau `deleted` est défini sur `True`
- L'horodatage `deleted_at` est enregistré
- L'utilisateur ne peut pas se connecter
- Toutes les sessions actives sont invalidées
- L'utilisateur apparaît toujours dans les journaux d'audit et les enregistrements historiques
- Les données liées (contacts clients, activités) sont préservées

## Visualiser les Utilisateurs Supprimés

**Filtrer pour les utilisateurs supprimés :**

`GET /auth/users/search?filters={"deleted":[true]}`

## Restauration d'un Utilisateur Supprimé

Si un utilisateur a été supprimé par erreur, les administrateurs peuvent le restaurer :

**Point de terminaison :** `PUT /auth/users/{user_id}`

```
{
  "deleted": false
}
```

Cela efface le drapeau `deleted` et permet à l'utilisateur de se reconnecter.

**Remarque :** Le mot de passe de l'utilisateur reste inchangé, donc il peut utiliser son ancien mot de passe.

## Suppression Permanente d'un Utilisateur

**Avertissement :** Cela est irréversible et supprime toutes les données de l'utilisateur de la base de données.

Non exposé via l'UI. Disponible uniquement via un accès direct à la base de données pour des raisons de conformité (par exemple, demandes de suppression de données GDPR).

## Meilleures Pratiques pour la Suppression d'Utilisateurs

- **Suppression douce par défaut** - Préserve les pistes d'audit
- **Documenter les raisons de la suppression** - Ajouter une note dans le journal d'activité avant de supprimer
- **Transférer la propriété** - Réaffecter les tickets ouverts et les tâches de l'utilisateur avant de supprimer
- **Examiner l'accès** - S'assurer qu'aucun processus critique ne dépend de l'utilisateur
- **Archiver les données** - Exporter l'historique de travail de l'utilisateur si nécessaire
- **Notifier les équipes concernées** - Informer les responsables/collègues de la suppression

# Catalogue des Permissions

Les permissions suivent généralement des modèles CRUD :

- *view_** : lire/parcourir
- *create_** : créer/ajouter
- *update_** : éditer/modifier
- *delete_** : supprimer/enlever

Certaines entités incluent également des variantes **"voir ses propres ..."** qui restreignent la visibilité aux données associées au **propre client/locataire** de l'utilisateur.

# Global / Administratif

- `admin` : Accès administratif complet (gérer les utilisateurs, les rôles et les permissions ; accéder à tous les points de terminaison protégés).
- `can_impersonate` : Agir temporairement en tant qu'autre utilisateur (audité ; pour le support/dépannage).
- `grafana_access` : Accès aux tableaux de bord d'analytique Grafana pour créer et visualiser des rapports et des visualisations personnalisés. Voir la section Analytique Grafana ci-dessous pour plus de détails.

# Clients et Enregistrements Connexes

- **Client**
  - *view_customer*, *create_customer*, *update_customer*, *delete_customer*
  - **Portée propre :** *view own customer*
- **Site Client**

    - *view_customer_site*, *create_customer_site*, *update_customer_site*, *delete_customer_site*
    - **Portée propre :** *view own customer site*
- **Contact Client**
    - *view_customer_contact*, *create_customer_contact*, *update_customer_contact*, *delete_customer_contact*
    - **Portée propre :** *view own customer contact*
- **Attribut Client** (voir Attributs Clients)
    - *view_customer_attribute*, *create_customer_attribute*, *update_customer_attribute*, *delete_customer_attribute*
    - **Portée propre :** *view own customer attribute*
- **Tag Client** (voir Tags Clients)
    - *view_customer_tag*, *create_customer_tag*, *update_customer_tag*, *delete_customer_tag*
    - **Portée propre :** *view own customer tag*
- **Service Client**
    - *view_customer_service*, *create_customer_service*, *update_customer_service*, *delete_customer_service*
    - **Portée propre :** *view own customer service*
- **Activité Client**
    - *view_customer_activity*, *create_customer_activity*, *update_customer_activity*, *delete_customer_activity*
    - **Portée propre :** *view own customer activity*

# Facturation

- **Carte Stripe**
    - `view_customer_stripe_card`, `create_customer_stripe_card`, `update_customer_stripe_card`, `delete_customer_stripe_card`
    - **Portée propre :** *view own customer stripe card*
- **Transactions**
    - `view_customer_transaction`, `create_customer_transaction`, `update_customer_transaction`, `delete_customer_transaction`
    - **Portée propre :** *view own customer transaction*

- **Factures**
  - `view_customer_invoice`, `create_customer_invoice`, `update_customer_invoice`, `delete_customer_invoice`
  - **Portée propre :** *view own customer invoice*

# Communications

- *view_communication*, *create_communication*, *update_communication*, *delete_communication*
- **Portée propre :** *view own communication*

# Inventaire et Modèles

- **Inventaire**
  - `view_inventory`, `create_inventory`, `update_inventory`, `delete_inventory`
  - **Portée propre :** *view own inventory*
- **Modèle d'Inventaire**
  - `view_inventory_template`, `create_inventory_template`, `update_inventory_template`, `delete_inventory_template`
  - **Portée propre :** *view own inventory template*

#### Produits

- `view_product`, `create_product`, `update_product`, `delete_product`

#### Diffusion Cellulaire (CBC)

- `view_cbc_message`, `create_cbc_message`, `update_cbc_message`, `delete_cbc_message`

#### Provisionnement

- **Provision**
  - `view_provision`, `create_provision`, `update_provision`, `delete_provision`

    - **Portée propre :** *view own provision*
- **Événement de Provisionnement**
    - `view_provision_event`, `create_provision_event`, `update_provision_event`, `delete_provision_event`

# Accès "Voir Ses Propres"

Les permissions "voir ses propres ..." limitent les lectures (et éventuellement les modifications, où cela est mis en œuvre) aux données associées au **propre client/locataire** de l'utilisateur. Par exemple, un rôle *Administrateur Client* peut gérer les contacts, sites, factures et services de *son* locataire, mais ne peut pas voir d'autres locataires.

# Conceptions de Rôles Typiques

| Rôle | Permissions Typiques | Remarques |
|---|---|---|
| Administrateur Système | `admin`, éventuellement `can_impersonate` ; plus large CRUD selon les besoins | Contrôle total sur les utilisateurs/rôles/permissior |
| Support | `view_customer`, `view_customer_service`, `view_communication`, `view_provision` ; mises à jour optionnelles | Ajouter `can_impersonate` si permis |
| Finance | `view_customer_invoice`, `view_customer_transaction`, `view_product` ; optionnel `create_customer_invoice` | Axé sur la lecture ; écriture limitée |
| Administrateur Client (locataire) | "view own ..." à travers clients, sites, contacts, services, inventaire, factures, transactions, communications, provisionnement | Gestion à portée locataire |
| Auditeur en Lecture Seule | Large `view_*` seulement | Pas de création/mise à jour/suppression |

Exemples de Rôles et Permissions Incluses (résumé)

# Gestion des Rôles et Permissions via l'API

Tous les points de terminaison nécessitent la permission `admin`.

**Lister les permissions**

**Point de terminaison :** `GET /auth/permissions`

**Créer une permission (rare)**

**Point de terminaison :** `POST /auth/permissions`

**Corps de la Requête :**

```
{
    "name": "view_example",
    "description": "Accès en lecture seule aux objets d'exemple"
}
```

**Lister les rôles**

**Point de terminaison :** `GET /auth/roles`

**Créer un rôle**

**Point de terminaison :** `POST /auth/roles`

**Corps de la Requête :**

```
{
"name": "Support",
"description": "Équipe de support de niveau 1"
}
```

**Ajouter une permission à un rôle**

**Point de terminaison :** `POST /auth/roles/{role_id}/permissions`

**Corps de la Requête :**

```
{
  "permission_id": 123
}
```

### Supprimer une permission d'un rôle

**Point de terminaison :** `DELETE /auth/roles/{role_id}/permissions/{permission_id}`

# Attribution de Rôles aux Utilisateurs

### Créer un utilisateur avec rôle

**Point de terminaison :** `POST /auth/users`

**Corps de la Requête :**

```
{
  "username": "sara",
  "email": "sara@example.com",
  "password": "TempP@ssw0rd!",
  "first_name": "Sara",
  "last_name": "Ng",
  "phone_number": "+61...",
  "role": "Support"
}
```

### Mettre à jour le rôle d'un utilisateur

**Point de terminaison :** `PUT /auth/users/{user_id}`

**Corps de la Requête :**

```
{
  "role": "Finance"
}
```

### Lister les utilisateurs (Admin uniquement)

**Point de terminaison :** `GET /auth/users`

**Usurpation d'identité (Contrôlée)**

- **Requis :** `can_impersonate` ou `admin`

**Démarrer l'usurpation d'identité**

**Point de terminaison :** `POST /auth/impersonate`

**Corps de la Requête :**

```
{ "user_id": 42 }
```

**Arrêter l'usurpation d'identité**

**Point de terminaison :** `POST /auth/stop_impersonation`

# Meilleures Pratiques

- **Moins de privilèges d'abord.** Commencer avec des rôles minimaux ; ajouter des permissions au besoin.
- **Préférer "voir ses propres ...".** Utiliser des permissions à portée locataire pour les rôles orientés client.
- **Maintenir la stabilité des rôles.** Mettre à jour les permissions des rôles lorsque les fonctionnalités changent. Ne pas éditer chaque utilisateur.
- **Auditer régulièrement.** Examiner qui a `admin` ou `can_impersonate`.

# Analytique et Tableaux de Bord Grafana

OmniCRM s'intègre à Grafana pour fournir des capacités d'analytique et de visualisation puissantes. Les utilisateurs avec la permission `grafana_access` peuvent accéder à Grafana pour créer des tableaux de bord, des rapports et des visualisations personnalisés.

# Accéder à Grafana

Si votre compte utilisateur a reçu la permission `grafana_access`, vous pouvez accéder à Grafana en naviguant vers `/grafana` dans votre navigateur. Vous serez automatiquement authentifié en utilisant vos identifiants OmniCRM - aucune connexion séparée à Grafana n'est requise.

Depuis l'interface Grafana, vous pouvez :

- Créer des tableaux de bord personnalisés avec des graphiques, des tableaux et des tableaux
- Construire des requêtes complexes pour analyser les données clients, les tendances de revenus et les métriques de service
- Configurer des alertes basées sur des indicateurs de performance clés
- Partager des tableaux de bord avec d'autres membres de l'équipe

# Intégration des Tableaux de Bord dans OmniCRM

Une fois que vous avez créé des tableaux de bord dans Grafana, vous pouvez les intégrer directement dans l'interface d'OmniCRM. Cela vous permet de visualiser vos métriques les plus importantes sans quitter le CRM, tout en maintenant la navigation d'OmniCRM intacte.

Pour intégrer un tableau de bord :

1. **Créez votre tableau de bord dans Grafana** - Naviguez vers `/grafana` et concevez votre tableau de bord avec les visualisations dont vous avez besoin.

2. **Notez l'ID du tableau de bord** - Lorsque vous visualisez votre tableau de bord dans Grafana, vérifiez l'URL. Elle ressemblera à `/grafana/d/abc123/dashboard-name`. L'ID est la partie après `/d/` (dans cet exemple, `abc123`).

3. **Configurez le tableau de bord dans votre environnement** - Contactez votre administrateur système pour ajouter l'ID du tableau de bord et un

nom d'affichage à la configuration du système. Les tableaux de bord sont configurés via des variables d'environnement.

4. **Accéder depuis la barre latérale** - Une fois configuré, votre tableau de bord apparaîtra automatiquement dans la barre latérale d'OmniCRM sous le menu "Tableaux de Bord". Vous pouvez cliquer dessus pour voir le tableau de bord complet tout en gardant la navigation d'OmniCRM visible.

## Configuration du Tableau de Bord

Les tableaux de bord sont configurés par votre administrateur système à l'aide de variables d'environnement. Chaque tableau de bord nécessite :

- **ID du Tableau de Bord** - L'identifiant unique de l'URL Grafana
- **Étiquette du Tableau de Bord** - Le nom convivial affiché dans la barre latérale d'OmniCRM

Plusieurs tableaux de bord peuvent être configurés et apparaîtront comme des éléments de menu dans l'ordre où ils sont définis. Voir la documentation de Configuration d'Administration pour plus de détails sur la configuration des variables d'environnement.

## Qui Peut Accéder à Grafana ?

Seuls les utilisateurs avec la permission `grafana_access` peuvent accéder à Grafana et visualiser les tableaux de bord intégrés. Cette permission est généralement accordée à :

- Administrateurs qui ont besoin d'une visibilité complète sur les métriques du système
- Analystes commerciaux qui créent des rapports et des visualisations
- Équipes financières qui suivent les revenus et les métriques de facturation
- Équipes opérationnelles qui surveillent la performance des services

# FAQ

**Un utilisateur peut-il avoir plusieurs rôles ?** Oui. Les permissions sont additives.

**Ai-je besoin de permissions personnalisées ?** Généralement non. Le catalogue intégré couvre la plupart des besoins.

**Comment les règles "voir ses propres ..." savent-elles ce qui est à moi ?** Elles évaluent le lien entre votre utilisateur/contact et votre client (locataire).

**Puis-je créer mes propres tableaux de bord Grafana ?** Oui, si vous avez la permission `grafana_access`. Naviguez vers `/grafana` pour accéder à l'interface complète de Grafana.

# Portail de Soins Personnels

Le Portail de Soins Personnels est une interface destinée aux clients qui permet aux utilisateurs finaux de gérer leurs propres comptes, de consulter leur utilisation, de payer des factures et de modifier des services sans avoir besoin d'assistance du personnel du service client.

**Méthodes d'Accès :**

- Connexion directe via les identifiants du client
- Impersonation du personnel pour le dépannage (de Clients → Contacts → "Se connecter en tant qu'utilisateur")

Voir aussi : Service Client pour les détails sur l'imitation, Flux d'Authentification pour le processus de connexion.

# Objectif

Le Portail de Soins Personnels fournit aux clients :

1. **Gestion de Compte** - Consulter et mettre à jour les informations personnelles, les contacts et les adresses
2. **Aperçu des Services** - Voir tous les services actifs, l'utilisation et les dates d'expiration
3. **Suivi de l'Utilisation** - Surveiller les données, la voix, les SMS et les soldes monétaires
4. **Accès à la Facturation** - Consulter et payer des factures, gérer les méthodes de paiement
5. **Modifications de Services** - Ajouter des recharges, acheter des options supplémentaires, modifier des services
6. **Disponibilité 24/7** - Accéder aux informations du compte à tout moment sans appeler le support

# Aperçu du Portail

Lorsque les clients se connectent au Portail de Soins Personnels, ils voient un tableau de bord personnalisé avec :

**Sections de Navigation :**

- **Tableau de Bord** - Aperçu rapide des services et des activités récentes
- **Compte** - Détails personnels, contacts, sites
- **Services** - Liste de tous les services avec statut et détails
- **Utilisation** - Informations sur la consommation et l'expiration des soldes
- **Facturation** - Factures, transactions, méthodes de paiement
- **Recharge** - Achat de crédits de données, de voix, de SMS ou de crédits monétaires

# Détails du Compte

La section Compte affiche les informations du client et permet des mises à jour en libre-service limitées.

**Champs Éditables :**

Les clients peuvent mettre à jour :

- Adresse e-mail (nécessite vérification)
- Numéro de téléphone
- Mot de passe
- Préférences de notification

**Informations en Lecture Seule :**

- ID Client
- Date de création du compte
- Type de client (Particulier/Entreprise)
- Sites (adresses)
- Contacts liés

**Mise à Jour des Informations du Compte :**

1. Naviguer vers **Compte** → **Détails**
2. Cliquer sur **"Modifier"** à côté du champ à mettre à jour
3. Entrer les nouvelles informations
4. Cliquer sur **"Enregistrer les Modifications"**
5. Pour les changements d'e-mail, vérifier via le lien envoyé à la nouvelle adresse

**Fonctionnalités de Sécurité :**

- Les changements de mot de passe nécessitent le mot de passe actuel
- Les changements d'e-mail nécessitent une vérification
- Activité enregistrée pour la traçabilité
- Paramètres 2FA (si activés)

# Aperçu des Services

La section Services montre tous les services actifs et inactifs pour le client.

**Affichage de la Carte de Service :**

Chaque service affiche :

- **Nom du Service** - Identifiant lisible par l'homme (par exemple, "Mobile - +44 7700 900123")
- **Produit** - Nom du plan ou du produit
- **Statut** - Actif, Suspendu, Expiré, Annulé
- **Date de Création** - Date d'activation du service
- **Date d'Expiration** - Quand le service expire (le cas échéant)
- **Coût Mensuel** - Charge récurrente
- **Renouvellement Automatique** - Indicateur Activé/Désactivé

**Actions de Service :**

- **Voir l'Utilisation** - Voir la consommation de solde (Données, Voix, SMS, Monétaire)
- **Recharge** - Ajouter du crédit ou des données
- **Options Supplémentaires** - Acheter des fonctionnalités supplémentaires
- **Modifier** - Changer les paramètres du service (si autorisé)
- **Voir les Détails** - Voir la configuration complète du service

**Indicateurs de Statut du Service :**

- ✅ **Actif** - Le service est opérationnel
- ⚠️ **Expiration Proche** - Renouvelle ou expire dans <7 jours
- ⏸️ **Suspendu** - Service temporairement désactivé (problème de paiement, suspension manuelle)
- ● **Expiré** - Service n'est plus actif

# Suivi de l'Utilisation

Les clients peuvent surveiller leur utilisation à travers tous les types de soldes en temps réel.

## Utilisation des Données

Voir la consommation de données avec des détails par compartiment et date d'expiration.

**Affichage de l'Utilisation des Données :**

- **Solde Total** - Tous les compartiments de données combinés

- **Utilisé Cette Période** - Consommation depuis le dernier renouvellement
- **Barre de Progression** - Représentation visuelle de la consommation
- **Informations d'Expiration** - Quand chaque compartiment expire
- **Répartition des Compartiments** - Plusieurs compartiments de données avec ordre de priorité

**Priorité des Compartiments :**

Lorsque plusieurs compartiments de données existent (par exemple, plan de base + recharges), ils sont consommés par ordre de poids :

- **Poids 10** - Consommé en premier (généralement des données promotionnelles/bonus)
- **Poids 20** - Consommé en second (généralement des données de recharge)
- **Poids 30** - Consommé en dernier (généralement des données de plan de base)

# Utilisation de la Voix

Suivre les minutes d'appel restantes à travers tous les compartiments de voix.

**Affichage de l'Utilisation de la Voix :**

- **Minutes Restantes** - Solde total de voix
- **Minutes Utilisées** - Consommation cette période
- **Historique des Appels** - Appels récents (si activé)
- **Dates d'Expiration** - Quand les compartiments de voix expirent
- **Minutes Internationales** - Suivi séparé (le cas échéant)

**Répartition de l'Utilisation :**

- Appels Sur Réseau - Appels au sein du même réseau
- Appels Hors Réseau - Appels vers d'autres réseaux
- Appels Internationaux - Appels en dehors du pays
- Numéros Premium - Numéros à tarif spécial

# Utilisation des SMS

Surveiller les allocations et la consommation de messages SMS.

**Affichage de l'Utilisation des SMS :**

- **Messages Restants** - Solde SMS
- **Utilisés Ce Mois** - Messages envoyés
- **MMS Inclus** - Si les MMS comptent dans le solde
- **SMS Internationaux** - Suivi séparé (le cas échéant)

# Solde Monétaire

Voir le solde de crédit prépayé pour les services à la demande.

**Affichage Monétaire :**

- **Solde Actuel** - Crédit disponible
- **Dernière Recharge** - Montant et date de la recharge la plus récente
- **Date d'Expiration** - Quand le solde expire (le cas échéant)
- **Recharge Automatique** - Statut Activé/Désactivé

# Gestion de la Facturation

Les clients peuvent consulter des factures, des transactions et gérer des méthodes de paiement.

## Factures

Consulter et payer des factures impayées directement depuis le portail.

**Affichage de la Liste des Factures :**

- **Numéro de Facture** - Identifiant unique
- **Date** - Date de création de la facture
- **Date d'Échéance** - Date limite de paiement
- **Montant** - Montant total de la facture
- **Statut** - Payé, Impayé, En Retard
- **Actions** - Télécharger PDF, Payer en Ligne

**Payer une Facture :**

1. Naviguer vers **Facturation** → **Factures**

2. Trouver la facture impayée dans la liste
3. Cliquer sur le bouton **"Payer Maintenant"**
4. Sélectionner la méthode de paiement (carte enregistrée ou nouvelle carte)
5. Confirmer le paiement
6. Recevoir un e-mail de confirmation

**Télécharger des Factures :**

1. Cliquer sur l'icône **"Télécharger"** à côté de la facture
2. PDF téléchargé avec tous les détails de la facture
3. Enregistré pour les dossiers fiscaux et la documentation

# Transactions

Consulter l'historique complet des transactions, y compris les charges, les crédits et les paiements.

**Affichage des Transactions :**

- **Date** - Date de création de la transaction
- **Description** - À quoi correspond la charge/le crédit
- **Montant** - Charge (positive) ou crédit (négatif)
- **Facture** - Quelle facture inclut cette transaction
- **Statut** - Facturé ou Non Facturé

# Méthodes de Paiement

Gérer les cartes de crédit enregistrées pour la facturation automatique et les paiements en ligne.

**Gestion des Méthodes de Paiement :**

- **Ajouter une Carte** - Ajouter en toute sécurité une nouvelle carte de crédit/débit via Stripe
- **Définir par Défaut** - Choisir la méthode de paiement principale
- **Supprimer une Carte** - Supprimer les cartes expirées ou inutilisées

- **Détails de la Carte** - Voir les 4 derniers chiffres, l'expiration, la marque de la carte

Voir Méthodes de Paiement pour la documentation détaillée sur la gestion des paiements.

# Recharge / Rechargement

Acheter instantanément des données supplémentaires, de la voix, des SMS ou du crédit monétaire.

**Processus de Recharge :**

1. Naviguer vers **Services** → Sélectionner le service → **"Recharge"**
2. Choisir le produit de recharge dans le catalogue
3. Sélectionner le montant (options prédéfinies ou personnalisé)
4. Vérifier le coût et les informations d'expiration
5. Sélectionner la méthode de paiement
6. Confirmer l'achat
7. Solde mis à jour immédiatement

**Recharges Disponibles :**

- **Recharges de Données** - Options de 1 Go, 5 Go, 10 Go, 20 Go, 50 Go
- **Recharges de Voix** - Minutes d'appel supplémentaires
- **Forfaits SMS** - Packs de messages
- **Crédit Monétaire** - Solde prépayé (£5, £10, £20, £50, £100)

Voir Recharge & Rechargement pour des flux de travail de recharge détaillés.

# Options Supplémentaires de Service

Acheter des fonctionnalités et des améliorations supplémentaires pour les services existants.

**Options Disponibles :**

- **Roaming International** - Activer le service à l'étranger
- **Adresse IP Statique** - IP fixe pour Internet à domicile
- **Contenu Premium** - Chaînes IPTV, services de streaming
- **Mises à Niveau Matérielles** - Location de modem, décodeurs
- **Augmentations de Vitesse** - Augmentations temporaires de bande passante

**Achat d'Options Supplémentaires :**

1. Naviguer vers **Services** → Sélectionner le service → **"Options Supplémentaires"**
2. Parcourir les options disponibles pour ce type de service
3. Cliquer sur **"Ajouter au Service"** sur l'option désirée
4. Vérifier le coût (unique + récurrent)
5. Confirmer l'achat
6. Option fournie automatiquement

Voir Modification des Services - Options Supplémentaires pour les détails de gestion des options supplémentaires.

# Notifications & Alertes

Les clients reçoivent des notifications automatisées pour des événements importants :

**Notifications par E-mail :**

- Facture générée et prête à être payée
- Confirmation de paiement reçu
- Avertissements d'expiration de service (7 jours, 3 jours, 1 jour)
- Alertes de solde faible (données, voix, monétaire)
- Activation/désactivation de service
- Demandes de réinitialisation de mot de passe
- Alertes de sécurité (connexion depuis un nouvel appareil)

**Alertes dans le Portail :**

- Factures impayées
- Services expirants
- Avertissements de données faibles (10 % restantes)
- Méthode de paiement expirante
- Actions requises (vérifier l'e-mail, mettre à jour la méthode de paiement)

**Préférences de Notification :**

Les clients peuvent configurer :

- Fréquence des notifications par e-mail
- Alertes SMS (si activées)
- Seuils d'alerte (par exemple, notifier lorsque <20 % de données restantes)
- Catégories de notification (facturation, utilisation, service)

# Limitations de Libre-Service

Certaines opérations nécessitent l'assistance du personnel :

**Nécessite le Service Client :**

- Changer le type de client (Particulier ↔ Entreprise)
- Transférer des services entre clients
- Annuler des services (peut être en libre-service si activé)
- Contester des factures
- Demander des remboursements
- Changer le contact principal
- Problèmes de provisionnement complexes

**Restrictions de Sécurité :**

- Ne peut pas voir ou modifier les comptes d'autres utilisateurs
- Limité à ses propres données clients (isolement des locataires)

- Ne peut pas accéder aux fonctions administratives
- Ne peut pas annuler des factures ou des transactions
- Ne peut pas modifier la configuration du service (uniquement les options supplémentaires/recharges)

# Accès à l'Impersonation du Personnel

Le personnel de support peut accéder au Portail de Soins Personnels en tant que client pour le dépannage.

**Processus d'Impersonation :**

1. Naviguer vers **Clients** → Sélectionner le client
2. Aller à l'onglet **Contacts**
3. Trouver le contact lié au compte utilisateur
4. Cliquer sur le bouton **"Se connecter en tant qu'utilisateur"**
5. Un nouvel onglet s'ouvre avec la vue du Portail de Soins Personnels du client
6. Toutes les actions sont enregistrées et attribuées à l'utilisateur impersonné
7. Le personnel voit exactement ce que le client voit

**Cas d'Utilisation :**

- **Dépannage** - Reproduire les problèmes signalés par le client
- **Vérification** - Confirmer que les configurations de service apparaissent correctement
- **Formation** - Démontrer les fonctionnalités du portail
- **Support** - Guider le client à travers le portail tout en voyant son écran

**Sécurité & Audit :**

- Nécessite la permission `can_impersonate` ou `admin`
- Toutes les actions sont enregistrées dans la traçabilité
- Le client voit l'imitation dans le journal d'activité
- Temps d'attente de session après inactivité
- Ne peut pas changer le mot de passe du client pendant l'imitation

Voir Service Client - Impersonation d'Utilisateur pour la documentation complète sur l'imitation.

# Réactivité Mobile

Le Portail de Soins Personnels est entièrement réactif et optimisé pour les appareils mobiles.

**Fonctionnalités Mobiles :**

- Navigation optimisée pour le toucher
- Dispositions simplifiées pour les petits écrans
- Gestes de balayage pour la navigation dans le carrousel
- Formulaires et saisies adaptés aux mobiles
- Scan de codes QR pour le provisionnement eSIM
- Appel téléphonique en un clic
- Intégration GPS pour l'autocomplétion d'adresse

**Application Web Progressive (PWA) :**

- Installer comme application sur l'écran d'accueil
- Consultation hors ligne des données récentes
- Notifications push (si activées)
- Chargement rapide avec des workers de service

# Réinitialisation de Mot de Passe & Récupération de Compte

Les clients peuvent réinitialiser les mots de passe oubliés sans appeler le support.

**Réinitialisation de Mot de Passe en Libre-Service :**

1. Cliquer sur **"Mot de Passe Oublié"** sur la page de connexion
2. Entrer l'adresse e-mail
3. Recevoir un e-mail de réinitialisation de mot de passe (valide 1 heure)
4. Cliquer sur le lien dans l'e-mail
5. Entrer un nouveau mot de passe (doit respecter les exigences de complexité)
6. Soumettre et se connecter avec le nouveau mot de passe

**Exigences de Mot de Passe :**

- Minimum 8 caractères
- Au moins une lettre majuscule
- Au moins une lettre minuscule
- Au moins un chiffre
- Au moins un caractère spécial (!@#$%^&*)

**Verrouillage de Compte :**

Après 5 tentatives de connexion échouées :

- Compte verrouillé pendant 30 minutes
- E-mail de réinitialisation de mot de passe envoyé automatiquement
- Notification de sécurité envoyée à l'e-mail enregistré

# Meilleures Pratiques pour les Clients

**Recommandations de Sécurité :**

1. Activer la 2FA pour une sécurité renforcée
2. Utiliser un mot de passe unique et fort
3. Garder l'adresse e-mail à jour pour les notifications
4. Configurer la méthode de paiement par défaut pour les renouvellements automatiques
5. Surveiller l'utilisation régulièrement pour éviter les frais supplémentaires
6. Enregistrer les codes de secours dans un endroit sûr (si 2FA activée)
7. Se déconnecter après avoir utilisé des ordinateurs partagés/publics

**Gestion de l'Utilisation :**

1. Activer les alertes de solde faible (10-20 % restantes)
2. Recharger avant que les soldes n'expirent pour éviter les interruptions de service
3. Vérifier les factures mensuelles pour des frais inattendus
4. Mettre à jour les méthodes de paiement avant l'expiration des cartes
5. Surveiller l'utilisation des données tout au long du mois pour éviter le throttling

**Escalade de Support :**

Si le libre-service ne résout pas le problème :

1. Vérifier la base de connaissances / articles d'aide (si disponibles)
2. Examiner le journal d'activité pour les changements récents
3. Contacter le service client par téléphone, e-mail ou chat
4. Fournir l'ID client et les détails du service pour une résolution plus rapide

# Accès à l'API

Les clients ayant des exigences techniques peuvent utiliser l'API directement.

**Génération de Clé API :**

Disponible pour les clients professionnels ou sur demande :

1. Naviguer vers **Compte → Accès API**
2. Cliquer sur **"Générer Clé API"**
3. Définir les permissions (lecture seule ou lecture-écriture)
4. Définir la date d'expiration
5. Enregistrer la clé API en toute sécurité (affichée une seule fois)

**Cas d'Utilisation de l'API :**

- Surveillance automatisée de l'utilisation
- Intégration avec des systèmes de facturation internes
- Recharges programmatiques
- Provisionnement de services via des scripts
- Exportations de données pour l'analyse

Voir Documentation de l'API pour les détails des points de terminaison et des exemples.

# Questions Fréquemment Posées

**Q : Pourquoi ne puis-je pas voir tous mes services ?**

R : Assurez-vous que vous êtes connecté avec le bon compte. Si vous avez plusieurs comptes clients, chacun a des services séparés. Contactez le support pour fusionner les comptes si nécessaire.

**Q : Mon paiement a échoué mais j'ai été facturé. Que dois-je faire ?**

R : Vérifiez votre relevé bancaire pour les charges en attente. Si la charge apparaît mais que la facture indique toujours impayée, contactez le support

avec le numéro de référence de la transaction.

**Q : Comment annuler un service ?**

R : Naviguez vers les détails du service et cliquez sur "Annuler le Service" (si le libre-service est activé). Sinon, contactez le service client pour traiter l'annulation.

**Q : Puis-je transférer un service à une autre personne ?**

R : Non, les transferts de service nécessitent l'assistance du service client pour des raisons de sécurité et de conformité.

**Q : Pourquoi mon solde de données diminue-t-il plus vite que prévu ?**

R : Vérifiez les mises à jour d'applications en arrière-plan, la qualité de streaming vidéo et les sauvegardes cloud automatiques. Examinez la répartition de l'utilisation dans le portail pour une consommation détaillée.

**Q : J'ai perdu mon appareil 2FA. Comment puis-je récupérer l'accès ?**

R : Utilisez les codes de secours pour vous connecter, puis désactivez et réactivez la 2FA. Si vous n'avez pas de codes de secours, contactez le support pour une réinitialisation de la 2FA (nécessite une vérification d'identité).

**Q : Puis-je payer une facture sans me connecter ?**

R : Le paiement de facture invité peut être disponible via un lien de facture direct. Sinon, la connexion est requise pour des raisons de sécurité.

**Q : Comment puis-je télécharger toutes mes factures en une seule fois ?**

R : Actuellement, il faut télécharger chaque facture individuellement. Pour des téléchargements en masse, contactez le support ou utilisez l'API si disponible.

# Documentation Connexe

- Service Client - Impersonation d'Utilisateur - Accès de dépannage du personnel
- Méthodes de Paiement - Gestion des cartes et des paiements
- Recharge & Rechargement - Achat de crédits supplémentaires
- Utilisation des Services - Comprendre le suivi des soldes
- Aperçu de la Facturation - Concepts de facturation et factures
- Flux d'Authentification - Connexion et sécurité
- 2FA - Configuration de l'authentification à deux facteurs
- Modifications de Services - Ajout de fonctionnalités et d'options supplémentaires

# FAQ Système de Base de Connaissances

Le système FAQ (Foire Aux Questions) fournit une base de connaissances en libre-service pour les clients et le personnel, réduisant le volume des tickets de support et permettant une résolution plus rapide des problèmes.

Voir aussi : Tickets de Support pour la gestion des tickets, Service Client pour les fonctionnalités de support client.

# Aperçu

Le système FAQ permet :

1. **Libre-Service** : Les clients trouvent des réponses sans créer de tickets
2. **Base de Connaissances** : Répertoire centralisé de solutions
3. **Ressource pour le Personnel** : Référence rapide pour le personnel de support
4. **Amélioration Continue** : Suivre quelles FAQ sont les plus utiles
5. **Multi-Catégorie** : Organiser les FAQ par sujet et tags
6. **Contrôle de Visibilité** : FAQ publiques vs internes
7. **Analytique** : Comptes de vues et évaluations d'utilité

# Fonctionnalités Clés

## Gestion des FAQ

- **Créer des FAQ** : Rédiger des questions et des réponses détaillées
- **Contenu Riche** : Formatage Markdown, images, blocs de code
- **Catégorisation** : Organiser par catégorie et tags
- **Visibilité** : Contrôler l'accès client vs uniquement personnel
- **Recherche** : Recherche en texte intégral à travers les titres et le contenu

- **Versioning** : Suivre les changements et mises à jour au fil du temps

## Analytique

- **Suivi des Vues** : Compter combien de fois la FAQ est vue
- **Utilité** : Les clients évaluent si la FAQ était utile
- **FAQ Populaires** : Questions les plus vues
- **Les Plus Utiles** : Solutions les mieux notées
- **Performances Faibles** : FAQ marquées comme inutiles (besoin d'amélioration)

## Organisation

- **Catégories** : Regrouper les FAQ connexes (Facturation, Technique, Compte, etc.)
- **Tags** : Plusieurs tags par FAQ pour la catégorisation croisée
- **Recherche** : Trouver rapidement par mot-clé
- **Filtrage** : Filtrer par catégorie, tags, visibilité
- **Tri** : Par date, vues, utilité, ordre alphabétique

# Structure des FAQ

## Champs Obligatoires

- **Question** : Question claire et concise (le titre de la FAQ)
- **Réponse** : Réponse détaillée avec des instructions étape par étape
- **Catégorie** : Catégorisation principale
- **Visibilité** : Visible par le client ou uniquement par le personnel

## Champs Optionnels

- **Tags** : Mots-clés de catégorisation supplémentaires
- **FAQ Connexes** : Liens vers des questions similaires (fonctionnalité future)

- **Pièces Jointes** : Captures d'écran, documents, vidéos (fonctionnalité future)
- **Auteur** : Membre du personnel qui a créé la FAQ
- **Dernière Mise à Jour** : Quand la FAQ a été modifiée pour la dernière fois

# Exemple de FAQ

**Question :** Comment vérifier ma consommation de données ?

**Catégorie :** Gestion de Compte

**Tags :** données, consommation, solde, auto-assistance

**Réponse :**

```
Vous pouvez vérifier votre consommation de données actuelle de
plusieurs manières :

## Via le Portail d'Auto-Assistance

1. Connectez-vous à votre compte sur portal.example.com
2. Allez à l'onglet **Services**
3. Cliquez sur votre service mobile
4. Consultez la section **Solde**
5. Votre consommation de données s'affiche comme "Données
Utilisées / Données Totales"

## Via l'Application Mobile

1. Ouvrez l'application mobile
2. Appuyez sur **Mes Services**
3. Sélectionnez votre ligne mobile
4. La consommation de données s'affiche sur l'écran d'aperçu

## Via SMS

Envoyez **BAL** au 123456 pour recevoir votre solde actuel par
SMS.

## Besoin d'Aide ?

Si vous ne pouvez pas voir votre consommation, veuillez contacter
le support à support@example.com
```

**Visibilité :** Visible par le Client

# Consultation et Édition des FAQ

Lors de la consultation d'une FAQ, tout le formatage markdown est rendu en HTML riche avec un style approprié pour les en-têtes, les listes, les blocs de code et les liens.

*Figure 1 : Vue détail de la FAQ montrant le contenu markdown rendu avec des en-têtes formatés, des listes et des blocs de code*

La page de détail de la FAQ fournit :

- **Contenu Rendu** : Markdown automatiquement converti en HTML formaté
- **Statistiques** : Compte de vues et évaluations d'utilité
- **Mode Édition** : Édition en ligne avec support markdown
- **Basculer la Visibilité** : Afficher/masquer aux clients
- **Actions Rapides** : Copier le lien, supprimer la FAQ

# Vue Client

Les clients voient une vue propre et formatée des FAQ sans capacités d'édition :

*Figure 2 : Vue FAQ orientée client avec évaluation d'utilité et formatage propre*

# Création de FAQ

## Via l'Interface Web

1. Naviguez vers **Support → FAQ**
2. Cliquez sur **Créer FAQ**
3. Remplissez les détails de la FAQ :
    - Question (titre)
    - Réponse (contenu avec markdown)
    - Catégorie
    - Tags (séparés par des virgules)
    - Visibilité (case à cocher visible par le client)
4. Cliquez sur **Créer**

## Via l'API

```
POST /crm/faq/
```

**Permission Requise :** `create_faq`

**Requête :**

```
{
  "question": "Comment réinitialiser mon mot de passe ?",
  "answer": "Pour réinitialiser votre mot de passe :\n1. Allez à
la page de connexion\n2. Cliquez sur 'Mot de passe oublié'\n3.
Entrez votre email\n4. Vérifiez votre boîte de réception pour le
lien de réinitialisation\n5. Cliquez sur le lien et définissez un
nouveau mot de passe",
  "category": "Compte",
  "tags": "motdepasse,connexion,compte,réinitialiser",
  "is_visible_to_customer": true
}
```

**Réponse :**

```
{
  "faq_id": 42,
  "question": "Comment réinitialiser mon mot de passe ?",
  "answer": "Pour réinitialiser votre mot de passe :\n1. Allez à
la page de connexion...",
  "category": "Compte",
  "tags": "motdepasse,connexion,compte,réinitialiser",
  "is_visible_to_customer": true,
  "view_count": 0,
  "helpful_count": 0,
  "not_helpful_count": 0,
  "created": "2026-01-07T10:00:00Z",
  "last_modified": "2026-01-07T10:00:00Z",
  "created_by_user": {
    "id": 1,
    "username": "admin",
    "first_name": "Admin",
    "last_name": "User"
  }
}
```


# Édition des FAQ

## Mise à Jour du Contenu

1. Ouvrez la FAQ à partir de la liste
2. Cliquez sur **Éditer**
3. Modifiez la question, la réponse ou les métadonnées
4. Cliquez sur **Enregistrer**

Les changements sont suivis avec :

- Horodatage de la dernière modification
- Dernière modification par l'utilisateur
- Historique des changements (journal d'audit)

## Basculer la Visibilité

Contrôlez si les clients peuvent voir une FAQ :

1. Ouvrez la FAQ
2. Cliquez sur le bouton de basculement **Visibilité**
3. Choisissez :
    - **Visible par les Clients** : Apparaît dans le portail client
    - **Personnel Uniquement** : Visible uniquement par le personnel de support
4. Enregistrez les modifications

**Cas d'Utilisation pour le Personnel Uniquement :**

- Procédures et workflows internes
- Guides de dépannage avec des informations sensibles
- FAQ en brouillon en attente d'approbation
- Solutions obsolètes conservées pour référence

# Catégories de FAQ

## Catégories Standards

Organisez les FAQ en groupes logiques :

- **Gestion de Compte** : Connexion, mots de passe, profils
- **Facturation** : Factures, paiements, frais
- **Support Technique** : Problèmes de service, dépannage
- **Fonctionnalités** : Comment utiliser des fonctionnalités spécifiques
- **Services Mobiles** : Cartes SIM, données, itinérance
- **Services Internet** : Connectivité, vitesses, équipements
- **Services VoIP** : Appels, messagerie vocale, fonctionnalités
- **Services IPTV** : Chaînes, streaming, appareils

## Création de Catégories

Les catégories sont créées automatiquement lorsqu'elles sont utilisées dans une FAQ. Pour standardiser :

1. Utilisez des noms de catégories cohérents
2. Suivez les conventions de nommage
3. Examinez périodiquement la liste des catégories
4. Fusionnez les catégories dupliquées/similaires

## Gestion des Catégories

**Voir Toutes les Catégories :**

```
GET /crm/faq/categories
```

Renvoie la liste de toutes les catégories avec les comptes de FAQ.

**Vue Client :**

```
GET /crm/faq/customer/categories
```

Renvoie uniquement les catégories avec des FAQ visibles.

# Tags

## Objectif

Les tags fournissent une organisation transversale :

- Une seule FAQ peut avoir plusieurs tags
- Les tags complètent la structure de catégorie
- Permettent un filtrage et une recherche flexibles
- Aident les clients à trouver du contenu connexe

## Exemples de Tags

- **Technique :** dépannage, erreur, bug, correction, réinitialisation
- **Compte :** connexion, mot de passe, profil, paramètres, sécurité
- **Facturation :** facture, paiement, frais, remboursement, litige
- **Service :** activation, désactivation, mise à niveau, changement
- **Plateforme :** mobile, web, application, portail, auto-assistance

## Meilleures Pratiques

1. **Nommage Cohérent** : Utilisez des minuscules, forme singulière
2. **Tags Spécifiques** : "réinitialisation-motdepasse" pas juste "motdepasse"
3. **Évitez la Redondance** : Ne pas dupliquer la catégorie dans les tags
4. **Limiter le Nombre** : 3-5 tags par FAQ
5. **Standardiser** : Maintenir un glossaire de tags

# Portail FAQ Client

## Accéder aux FAQ

Les clients accèdent aux FAQ par :

1. **Portail d'Auto-Assistance** : Section FAQ dédiée
2. **Lors de la Création de Ticket** : Suggestions "Vérifiez ces FAQ d'abord"
3. **Recherche** : La recherche globale inclut le contenu des FAQ
4. **Email** : Liens dans les réponses automatisées

## Affichage des FAQ

La vue client montre :

- **Question** : Titre/question de la FAQ
- **Réponse** : Contenu complet formaté
- **Catégorie** : Pour parcourir les FAQ connexes

- **Tags** : Navigation supplémentaire
- **Évaluation d'Utilité** : Retours par pouces en haut/en bas
- **FAQ Connexes** : Questions similaires (futur)

## Recherche et Filtrage

Les clients peuvent :

- **Rechercher** : Recherche en texte intégral à travers toutes les FAQ visibles
- **Filtrer par Catégorie** : Voir les FAQ dans une catégorie spécifique
- **Filtrer par Tag** : Trouver des FAQ avec des tags spécifiques
- **Trier** : Par popularité, récence ou ordre alphabétique
- **Parcourir** : Explorer toutes les FAQ ou les listes de catégories

## Évaluation des FAQ

Les clients fournissent des retours :

1. Lire la FAQ
2. Faire défiler vers le bas
3. Cliquer sur **Cette information était-elle utile ?**
4. Sélectionner **Oui** ou **Non**
5. Évaluation enregistrée

Les évaluations aident à identifier :

- **FAQ Utiles** : Votes oui élevés
- **Besoin d'Amélioration** : Votes non élevés
- **Contenu Obsolète** : Utilité en déclin
- **Sujets Populaires** : Vues élevées + évaluations élevées

# Outils FAQ pour le Personnel

## FAQ Internes

FAQ uniquement pour le personnel pour :

- **Procédures** : Workflows et processus internes
- **Dépannage** : Étapes de diagnostic avancées
- **Escalade** : Quand et comment escalader des problèmes
- **Outils** : Comment utiliser les systèmes internes
- **Politiques** : Politiques et directives de l'entreprise

## Référence Rapide

Le personnel peut :

- **Rechercher Toutes les FAQ** : Y compris le contenu uniquement pour le personnel
- **Voir les Analytique** : Voir quelles FAQ sont les plus utiles
- **Copier des Liens** : Partager des liens FAQ avec les clients
- **Suggérer des Édits** : Proposer des améliorations aux FAQ
- **Créer à partir d'un Ticket** : Convertir la résolution d'un ticket en FAQ

## Analytique des FAQ

Suivre la performance des FAQ :

**Compte de Vues :**

- Vues totales par FAQ
- Vues au fil du temps
- FAQ les plus populaires
- FAQ tendance (pics soudains)

**Utilité :**

- Votes utiles (pouce en haut)
- Votes non utiles (pouce en bas)
- Ratio d'utilité
- FAQ les plus/moins utiles

**Points de Terminaison API :**

```
GET /crm/faq/popular?limit=10
GET /crm/faq/helpful?limit=10
```

## Amélioration Continue

Utilisez les analyses pour :

1. **Identifier les Lacunes** : Recherches courantes sans résultats
2. **Mettre à Jour le Contenu** : Améliorer les FAQ mal notées
3. **Retirer les Obsolètes** : Supprimer ou mettre à jour les FAQ obsolètes
4. **Promouvoir les Meilleures** : Mettre en avant les FAQ très notées
5. **Créer de Nouvelles** : Ajouter des FAQ pour les tickets récurrents

# Formatage Markdown

Les réponses FAQ supportent le markdown pour un formatage riche :

## Formatage de Texte

```
**Texte en gras** pour l'emphase
*Texte en italique* pour une emphase subtile
~~Barr~~ pour les informations obsolètes
`code en ligne` pour les termes techniques
```

## En-têtes

```
## En-tête de Section
### En-tête de Sous-section
#### En-tête Mineur
```

## Listes

```
Liste Ordonnée :
1. Première étape
2. Deuxième étape
3. Troisième étape

Liste Non Ordonnée :
- Point un
- Point deux
- Point trois
```

## Liens

```
[Texte du lien](https://example.com)
[Email support](mailto:support@example.com)
```

## Blocs de Code

````
```bash
commande --avec-drapeaux
```

```json
{
  "exemple": "données"
}
```
````

## Images

```
![Texte alternatif](https://example.com/image.png)
```

## Tables

```
| Colonne 1 | Colonne 2 |
|------------|------------|
| Donnée 1   | Donnée 2   |
```

## Citations

```
> Note ou astuce importante
> Se poursuit sur plusieurs lignes
```

# Référence API

## Obtenir Toutes les FAQ (Personnel)

```
GET /crm/faq/
```

**Permission Requise :** `view_faq`

**Paramètres de Requête :**

- `page` : Numéro de page (par défaut : 1)
- `per_page` : Résultats par page (par défaut : 20)
- `sort` : Champ de tri (créé, compte_de_vues, compte_utiles)
- `order` : Ordre de tri (asc, desc)
- `search` : Requête de recherche
- `category` : Filtrer par catégorie
- `tags` : Filtrer par tags (séparés par des virgules)

- `is_visible_to_customer` : Filtrer par visibilité (true/false)

## Obtenir les FAQ Client

```
GET /crm/faq/customer
```

**Authentification Requise :** Oui **Permission :** Tout utilisateur authentifié

Renvoie uniquement les FAQ visibles par les clients.

## Obtenir une FAQ Unique

```
GET /crm/faq/faq_id/{faq_id}
```

**Vue Personnel :** N'incrémente pas le compte de vues

```
GET /crm/faq/customer/faq_id/{faq_id}
```

**Vue Client :** Incrémente le compte de vues

## Mettre à Jour une FAQ

```
PATCH /crm/faq/faq_id/{faq_id}
```

**Permission Requise :** `update_faq`

## Supprimer une FAQ

```
DELETE /crm/faq/faq_id/{faq_id}
```

**Permission Requise :** `delete_faq`

Effectue une suppression douce (marque comme supprimée, préserve les données).

## Basculer la Visibilité

```
POST /crm/faq/faq_id/{faq_id}/visibility
```

**Permission Requise :** `manage_faq_visibility`

**Requête :**

```
{
  "is_visible_to_customer": true
}
```

## Marquer une FAQ Comme Utile

```
POST /crm/faq/faq_id/{faq_id}/helpful
```

**Authentification Requise :** Oui **Permission :** Tout utilisateur authentifié

**Requête :**

```
{
  "is_helpful": true
}
```

## Obtenir les FAQ Populaires

```
GET /crm/faq/popular?limit=10
```

Renvoie les FAQ triées par compte de vues.

## Obtenir les FAQ les Plus Utiles

```
GET /crm/faq/helpful?limit=10
```

Renvoie les FAQ triées par compte d'utilité.

# Meilleures Pratiques

## Rédaction de Contenu

1. **Questions Claires** : Utilisez un langage naturel que les clients utiliseraient
2. **Réponses Complètes** : Inclure toutes les étapes, aucune supposition
3. **Captures d'Écran** : Ajouter des aides visuelles pour des procédures complexes
4. **Liens** : Référencer des FAQ et des ressources connexes
5. **Tests** : Vérifier que les instructions sont précises

## Organisation

1. **Catégories Cohérentes** : Utilisez des noms de catégories standardisés
2. **Tags Spécifiques** : Utilisez des tags descriptifs et recherchables
3. **Évitez la Duplication** : Fusionnez les FAQ similaires
4. **Cross-Reference** : Liez les FAQ connexes ensemble
5. **Révision Régulière** : Mettez à jour le contenu obsolète trimestriellement

## Maintenance

1. **Surveillez les Analytique** : Suivez les vues et l'utilité
2. **Mettez à Jour Régulièrement** : Gardez le contenu à jour
3. **Retirez les Anciennes FAQ** : Supprimez les informations obsolètes
4. **Incorporez les Retours** : Améliorez en fonction des évaluations
5. **Créez à partir de Tickets** : Transformez les problèmes courants en FAQ

# Dépannage

**Problème : FAQ n'apparaît pas pour les clients**

- Solution : Vérifiez que le paramètre "Visible par les Clients" est activé
- Solution : Vérifiez que la FAQ n'est pas marquée comme supprimée
- Solution : Effacez le cache du portail client

**Problème : La recherche ne trouve pas de FAQ**

- Solution : Vérifiez l'orthographe et essayez différents mots-clés
- Solution : Vérifiez que la FAQ est dans la bonne catégorie
- Solution : Ajoutez des tags plus pertinents à la FAQ

**Problème : Les analyses ne se mettent pas à jour**

- Solution : Attendez le rafraîchissement du cache (peut prendre quelques minutes)
- Solution : Vérifiez que le suivi des vues est activé
- Solution : Vérifiez que le client est authentifié lors de la consultation

**Problème : Impossible d'éditer la FAQ**

- Solution : Vérifiez que vous avez la permission `update_faq`
- Solution : Vérifiez que la FAQ n'est pas verrouillée par un autre utilisateur
- Solution : Confirmez que la FAQ existe et n'est pas supprimée

# Système de Tickets de Support

Le Système de Tickets de Support OmniCRM fournit une plateforme complète pour gérer les demandes de support client, suivre les problèmes et maintenir un historique de communication détaillé avec les clients.

Voir aussi : FAQs pour la base de connaissances en libre-service, Service Client pour l'imitation et le dépannage, Clients pour la gestion des clients.

*Figure 1 : Liste des tickets de support montrant tous les tickets avec filtrage, recherche, statut, priorité et personnel assigné*

## Aperçu

Le système de tickets permet :

1. **Support Centralisé** : Suivre tous les problèmes des clients en un seul endroit

2. **Intégration par Email** : Créer automatiquement des tickets à partir des emails des clients
3. **Collaboration** : Plusieurs membres du personnel peuvent travailler ensemble sur des tickets
4. **Base de Connaissances** : Lier les FAQs aux problèmes courants pour une résolution plus rapide
5. **Suivi des SLA** : Surveiller l'âge des tickets et les temps de réponse
6. **Portail Client** : Les clients peuvent voir et répondre à leurs tickets
7. **Piste de Vérification** : Historique complet de toutes les interactions liées aux tickets

# Fonctionnalités Clés

## Gestion des Tickets

- **Créer des Tickets** : Création manuelle ou automatique à partir d'un email
- **Mettre à Jour les Tickets** : Ajouter des commentaires, changer le statut, réassigner
- **Pièces Jointes** : Télécharger des fichiers (images, documents, journaux)
- **Tags** : Catégoriser les tickets pour les rapports et le filtrage
- **Recherche** : Recherche en texte intégral à travers les tickets, mises à jour et pièces jointes
- **Filtrage** : Filtrer par statut, priorité, personne assignée, client, service
- **Vues en Temps Réel** : Suivre qui consulte actuellement chaque ticket
- **Dernière Activité** : Voir quand les tickets ont été consultés ou mis à jour pour la dernière fois

## Communication

- **Notifications par Email** : Notifications automatiques sur les mises à jour des tickets
- **Notifications In-App** : Notifications en temps réel dans l'interface OmniCRM

- **@Mentions** : Mentionner les membres du personnel avec @nom_utilisateur pour les notifier
- **Abonnés** : Ajouter du personnel et des contacts aux notifications de tickets
- **Visibilité Client** : Contrôler quelles mises à jour les clients peuvent voir
- **Notes Internes** : Ajouter des notes privées non visibles par les clients
- **Suivi des Conversations par Email** : Suivi automatique des fils de discussion par email
- **Visiteurs Actifs** : Voir qui consulte actuellement le ticket en temps réel

## Intégration de Services

- **Lier aux Services** : Associer des tickets à des services clients spécifiques
- **Contexte de Service** : Voir les détails du service, l'utilisation et l'historique depuis le ticket
- **Actions Rapides** : Recharger, voir les options supplémentaires, éditer le service directement depuis le ticket

# Cycle de Vie du Ticket

## 1. Création de Ticket

**Via l'Interface Web :**

1. Naviguer vers **Support → Tickets**
2. Cliquer sur **Créer un Ticket**
3. Remplir les détails du ticket :
   - Titre (obligatoire)
   - Description (obligatoire)
   - Client (facultatif, peut être assigné plus tard)
   - Service (facultatif, lien vers un service client spécifique)
   - Priorité (Basse, Moyenne, Haute, Urgente)
   - Statut (Ouvert, En Cours, En Attente, Résolu, Fermé)
   - Tags (pour la catégorisation)
4. Cliquer sur **Créer**

**Via Email :**

Les clients peuvent créer des tickets en envoyant un email à votre adresse de support. Le système crée automatiquement :

- Un ticket à partir de l'objet et du corps de l'email
- Un lien vers le client si l'email de l'expéditeur correspond à un contact
- Les pièces jointes de l'email au ticket
- Une confirmation au client

**Via API :**

```
POST /crm/ticket/
```

**Demande :**

```
{
  "title": "Impossible d'accéder à la messagerie vocale",
  "description": "Le client signale que la messagerie vocale ne
fonctionne pas depuis hier",
  "customer_id": 290,
  "service_id": 167,
  "priority": "High",
  "status": "Open",
  "tags": "technical,voicemail"
}
```

## 2. Assignation de Ticket

Les tickets peuvent être assignés au personnel de support pour traitement :

1. **Assignation Manuelle** : Cliquer sur "Assigner" et sélectionner un membre du personnel
2. **Assignation Automatique** : Assignation automatique en fonction des règles (fonctionnalité future)
3. **Réassignation** : Transférer des tickets entre membres du personnel

4. **Non Assigné** : Les tickets peuvent rester non assignés dans la file d'attente

Lorsqu'un ticket est assigné :

- Le personnel assigné reçoit une notification par email
- Le personnel est automatiquement abonné aux mises à jour du ticket
- Le ticket apparaît dans leur vue "Mes Tickets"

# 3. Travailler sur les Tickets

**Ajouter des Mises à Jour :**

1. Ouvrir le ticket
2. Cliquer sur **Ajouter une Mise à Jour** ou répondre au ticket
3. Entrer votre message
4. Utiliser **@nom_utilisateur** pour mentionner et notifier des membres spécifiques du personnel (par exemple, "@john peux-tu aider avec ça ?")
5. Choisir la visibilité :
   - **Visible par le Client** : Le client verra cette mise à jour
   - **Interne Seulement** : Note privée pour le personnel
6. Cliquer sur **Soumettre**

**@Mentions :**

Lorsque vous mentionnez un membre du personnel en utilisant @nom_utilisateur dans une mise à jour de ticket :

- Ils reçoivent une notification dans l'application
- Ils sont automatiquement abonnés au ticket
- La notification inclut un lien directement vers le ticket
- Fonctionne dans les mises à jour visibles par le client et internes

**Joindre des Fichiers :**

1. Cliquer sur le bouton **Joindre des Fichiers**
2. Sélectionner des fichiers (max 10 Mo chacun)

3. Les fichiers sont téléchargés et liés au ticket
4. Formats pris en charge : PDF, images, documents, archives

**Changer le Statut :**

- **Ouvert** : Nouveau ticket, en attente de réponse initiale
- **En Cours** : Le personnel travaille activement sur le problème
- **En Attente** : En attente de réponse du client ou d'une dépendance externe
- **Résolu** : Problème corrigé, en attente de confirmation
- **Fermé** : Ticket complet et archivé

**Définir la Priorité :**

- **Urgent** : Problème critique affectant le service (SLA : 1 heure)
- **Élevé** : Problème important, service dégradé (SLA : 4 heures)
- **Moyen** : Problème standard (SLA : 24 heures)
- **Bas** : Problème mineur ou demande d'information (SLA : 48 heures)

## 4. Résolution de Ticket

Lors de la résolution d'un ticket :

1. Ajouter la mise à jour finale expliquant la résolution
2. Changer le statut en **Résolu**
3. Le client reçoit une notification
4. Le client peut rouvrir si le problème persiste
5. Après confirmation, changer en **Fermé**

# Vue Détails du Ticket

Lors de la consultation d'un ticket, vous voyez le fil de conversation complet avec toutes les mises à jour, pièces jointes, et une barre latérale complète affichant les informations sur le client et le service.

*Figure 2 : Page de détails du ticket montrant le fil de conversation, les mises à jour et les métadonnées du ticket*

# Barre Latérale Détails du Ticket

Chaque ticket affiche une barre latérale complète avec des informations sur le client, des détails sur le service, et des actions rapides :

*Figure 3 : Barre latérale du ticket avec informations sur le client, détails du service, personnel assigné, et boutons d'action rapide*

La barre latérale comprend :

## Informations sur le Client

- **Nom du Client** : Avec lien vers le profil du client
- **Email de Contact** : Email principal où les réponses sont envoyées
- **Modifier le Client** : Édition rapide de l'assignation du client

## Informations sur le Service

Lorsqu'un service est lié :

- **ID du Service** : Identifiant unique
- **Nom du Service** : Nom descriptif
- **Dates du Service** : Dates de début et de fin
- **Renouvellement Automatique** : Statut de renouvellement

- **Statut du Service** : Actif, Inactif, Suspendu
- **Type de Service** : Mobile, IPTV, Internet, VoIP
- **Actions Rapides :**
  - **Ajouter** : Ajouter des forfaits/fonctionnalités au service
  - **Supprimer** : Retirer le service (avec confirmation)
  - **Modifier le Service** : Modal de gestion complète du service

## Métadonnées du Ticket

- **Numéro de Ticket** : Référence unique (par exemple, TKT-2026-00042)
- **Date de Création** : Date d'ouverture du ticket
- **Dernière Mise à Jour** : Horodatage de la dernière activité
- **Assigné à** : Propriétaire actuel du ticket
- **Niveau de Priorité** : Avec codage couleur
- **Statut Actuel** : Avec badge de statut
- **Tags** : Étiquettes de catégorisation

## Abonnés

Liste des utilisateurs et contacts recevant des notifications :

- **Auto-abonnés** : Créateur et assigné
- **Abonnements manuels** : Autres membres du personnel ou contacts
- **Ajouter/Retirer** : Gérer la liste des abonnés

# Intégration par Email

## Traitement des Emails Entrants

Le système traite les emails de support entrants :

1. **Email Reçu** : L'email arrive à l'adresse de support
2. **Analyser l'Email** : Extraire l'expéditeur, l'objet, le corps, les pièces jointes

3. **Identifier le Client** : Faire correspondre l'email de l'expéditeur avec un contact
4. **Créer ou Mettre à Jour** : Créer un nouveau ticket ou ajouter à un fil existant
5. **Notifier le Personnel** : Alerter le personnel ou l'équipe assignée
6. **Envoyer une Confirmation** : Confirmer la réception au client

**En-têtes d'Email Utilisés :**

- `From` : Adresse email du client
- `Subject` : Titre du ticket (avec numéro de ticket si réponse)
- `In-Reply-To` : Suivi de fil pour les réponses
- `Message-ID` : Identifiant unique du message

## Notifications Sortantes

Les mises à jour du personnel déclenchent des notifications par email à :

- **Client** : Si la mise à jour est marquée comme visible par le client
- **Abonnés** : Tout le personnel et les contacts abonnés
- **Assigné** : Lorsque le ticket est assigné ou réassigné

**Variables de Modèle d'Email :**

- `{{ticket_number}}` : Référence du ticket
- `{{customer_name}}` : Nom du client
- `{{update_text}}` : Contenu de la mise à jour
- `{{staff_name}}` : Nom du membre du personnel qui a ajouté la mise à jour
- `{{ticket_url}}` : Lien direct vers le ticket

# Accès au Portail Client

Les clients peuvent voir leurs tickets via le portail de libre-service :

1. **Connexion** : Authentification au portail client
2. **Naviguer** : Aller à "Support" ou "Mes Tickets"

3. **Voir les Tickets** : Voir tous leurs tickets et leur statut
4. **Voir les Détails** : Lire l'historique complet du ticket (mises à jour visibles par le client uniquement)
5. **Répondre** : Ajouter des réponses aux tickets ouverts
6. **Pièces Jointes** : Voir et télécharger les pièces jointes
7. **Statut** : Voir le statut actuel du ticket et sa priorité

**Restrictions de Vue Client :**

- Les clients ne voient que leurs propres tickets
- Les mises à jour internes sont cachées
- Les pièces jointes réservées au personnel ne sont pas accessibles
- Les changements système (assignations, statut) sont résumés

# Abonnés et Notifications

## Auto-Abonnement

Abonnement automatique lorsque :

- **Création d'un ticket** : Créateur du ticket
- **Assigné à un ticket** : Actuel assigné
- **Mentionné** : Personnel @mentionné dans les mises à jour en utilisant la syntaxe @nom_utilisateur

## Abonnement Manuel

Ajouter des abonnés :

1. Ouvrir le ticket
2. Cliquer sur la section **Abonnés**
3. Cliquer sur **Ajouter un Abonné**
4. Sélectionner un utilisateur ou un contact
5. Cliquer sur **Ajouter**

Retirer des abonnés :

1. Cliquer sur X à côté du nom de l'abonné
2. Confirmer la suppression
3. Ils ne recevront plus de notifications

## Préférences de Notification

Le personnel reçoit des notifications par plusieurs canaux :

- **Notifications In-App** : Notifications en temps réel dans le menu déroulant de notifications
- **Notifications par Email** : Notifications par email configurables pour les mises à jour des tickets
- **Notifications @Mention** : Alertes immédiates lorsqu'ils sont mentionnés dans les mises à jour de tickets
- **Fréquence** : Immédiate, résumé horaire, résumé quotidien
- **Types** : Quels événements déclenchent des notifications (nouveaux tickets, mises à jour, assignations, mentions)

# Recherche et Filtrage

## Recherche Rapide

La barre de recherche globale recherche :

- Titres et descriptions de tickets
- Numéros de tickets
- Noms de clients
- Contenu des mises à jour
- Noms de pièces jointes

## Filtrage Avancé

Filtrer les tickets par :

- **Statut** : Ouvert, En Cours, Résolu, etc.

- **Priorité** : Urgent, Élevé, Moyen, Bas
- **Assigné à** : Membre spécifique du personnel ou non assigné
- **Client** : Client spécifique
- **Service** : Service spécifique
- **Tags** : Un ou plusieurs tags
- **Plage de Dates** : Créé ou mis à jour dans les dates
- **A des Pièces Jointes** : Tickets avec/sans fichiers

## Filtres Enregistrés

Créer et enregistrer des combinaisons de filtres personnalisés :

1. **Appliquer des Filtres** : Définir vos filtres souhaités
2. **Enregistrer le Filtre** : Cliquer sur le bouton "Enregistrer le Filtre"
3. **Nommer le Filtre** : Lui donner un nom descriptif
4. **Accès Rapide** : Apparaît dans le menu déroulant des filtres

**Filtres Enregistrés Courants :**

- "Mes Tickets Ouverts" : Assigné à moi, statut Ouvert ou En Cours
- "File Urgente" : Priorité Urgente, statut Ouvert
- "Besoin de Réponse" : Statut En Attente, en attente du client
- "Récemment Fermés" : Fermés dans les 7 derniers jours

# Reporting et Analytique

## Métriques des Tickets

Suivre la performance du support :

- **Total des Tickets** : Créés, résolus, fermés
- **Temps Moyen de Résolution** : Temps pour résoudre par priorité
- **Temps de Première Réponse** : Temps jusqu'à la première mise à jour du personnel
- **Tickets par Statut** : Distribution à travers les statuts

- **Tickets par Priorité** : Urgent vs Élevé vs Moyen vs Bas
- **Tickets par Personnel** : Charge de travail par membre du personnel
- **Tickets par Tag** : Catégories de problèmes courants

## Conformité aux SLA

Surveiller le respect des accords de niveau de service :

- **Dans le SLA** : Tickets résolus dans les délais cibles
- **SLA Violé** : Tickets dépassant les délais cibles
- **À Risque** : Tickets approchant la date limite du SLA
- **Réponse Moyenne** : Temps de réponse réel vs temps de réponse cible

## Insights Clients

Analyser les modèles de support client :

- **Principaux Créateurs de Tickets** : Clients avec le plus de tickets
- **Problèmes Répétitifs** : Clients avec des problèmes récurrents
- **Liés au Service** : Tickets par type de service
- **Taux de Résolution** : Taux de résolution au premier contact

# Intégration avec les Services

## Lier des Services aux Tickets

Lorsqu'un ticket est lié à un service spécifique :

1. **Sélectionner le Service** : Choisir parmi les services du client
2. **Contexte de Service** : Voir les détails du service dans la barre latérale
3. **Actions Rapides** : Gérer le service depuis le ticket
4. **Historique du Service** : Voir les tickets et changements liés

## Actions Rapides de Service

Depuis la barre latérale du ticket :

**Ajouter :**

1. Cliquer sur le bouton **Ajouter**
2. La modal des addons s'ouvre
3. Sélectionner les forfaits à ajouter
4. Confirmer et appliquer
5. Service mis à jour automatiquement

**Supprimer le Service :**

1. Cliquer sur le bouton **Supprimer**
2. La boîte de dialogue de confirmation apparaît
3. Confirmer la suppression
4. Le service est retiré
5. Ticket mis à jour avec l'action

**Modifier le Service :**

1. Cliquer sur le bouton **Modifier le Service**
2. La modal complète du service s'ouvre
3. Apporter des modifications à la configuration du service
4. Enregistrer les modifications
5. Retourner au ticket

# Meilleures Pratiques

## Création de Ticket

1. **Titres Descriptifs** : Résumé clair et concis du problème
2. **Description Détailée** : Inclure toutes les informations pertinentes
3. **Priorité Correcte** : Définir en fonction de l'impact réel
4. **Lier le Service** : Toujours lier au service affecté si applicable

5. **Ajouter des Tags** : Utiliser des tags cohérents pour la catégorisation

## Gestion des Tickets

1. **Mises à Jour Opportunes** : Mettre à jour les tickets régulièrement, même si "aucun progrès"
2. **Exactitude du Statut** : Garder le statut à jour (ne pas laisser les tickets résolus ouverts)
3. **Communication avec le Client** : Tenir les clients informés des progrès
4. **Notes Internes** : Documenter les étapes de dépannage en interne
5. **Documentation de Résolution** : Indiquer clairement ce qui a corrigé le problème

## Transfert de Connaissances

1. **Lier les FAQs** : Référencer ou créer des FAQs pour les problèmes courants
2. **Documenter les Solutions** : Ajouter des détails de résolution pour référence future
3. **Taguer Appropriatement** : Facilite la recherche de tickets similaires
4. **Partager les Enseignements** : Discuter des résolutions complexes avec l'équipe

## Expérience Client

1. **Premier Contact** : Répondre rapidement pour accuser réception
2. **Définir les Attentes** : Informer le client du délai de résolution attendu
3. **Ton Professionnel** : Maintenir la courtoisie et le professionnalisme
4. **Langage Clair** : Éviter le jargon technique avec les clients
5. **Confirmer la Résolution** : S'assurer que le client est satisfait avant de fermer

**Intégration du Système de Notification :**

- Ajout du support de notification in-app pour les événements de ticket
- Fonctionnalité @mention pour notifier des membres spécifiques du personnel

- Intégration avec l'API de Notification pour les systèmes externes
- Livraison de notifications en temps réel

**Collaboration en Temps Réel :**

- Suivi des visiteurs actifs - voir qui consulte actuellement chaque ticket
- Horodatages de dernière activité pour les tickets
- Mises à jour en temps réel lorsque plusieurs utilisateurs travaillent sur le même ticket

**Communication Améliorée :**

- Les @mentions abonnent automatiquement les utilisateurs mentionnés
- Amélioration des préférences de notification avec plusieurs canaux
- Meilleure intégration entre les notifications par email et in-app

Voir aussi : API de Notification pour les détails sur l'envoi de notifications depuis des systèmes externes.

# Référence API

## Créer un Ticket

```
POST /crm/ticket/
```

**Permission Requise :** `create_ticket`

## Obtenir un Ticket

```
GET /crm/ticket/ticket_id/{ticket_id}
```

**Permission Requise :** `view_ticket`

## Mettre à Jour un Ticket

```
PATCH /crm/ticket/ticket_id/{ticket_id}
```

**Permission Requise :** `update_ticket`

## Supprimer un Ticket

```
DELETE /crm/ticket/ticket_id/{ticket_id}
```

**Permission Requise :** `delete_ticket`

## Ajouter une Mise à Jour de Ticket

```
POST /crm/ticket/ticket_id/{ticket_id}/updates
```

## Télécharger une Pièce Jointe

```
POST /crm/ticket/ticket_id/{ticket_id}/attachments
```

**Permission Requise :** `upload_attachment`

## S'abonner à un Ticket

```
POST /crm/ticket/ticket_id/{ticket_id}/subscribe
```

**Permission Requise :** `subscribe_ticket`

# Dépannage

**Problème : Les emails ne créent pas de tickets**

- Solution : Vérifier la configuration de l'intégration par email
- Solution : Vérifier que l'email de l'expéditeur correspond à un contact client
- Solution : Examiner les journaux de traitement des emails

**Problème : Le client ne reçoit pas de notifications**

- Solution : Vérifier que l'email de contact est correct
- Solution : Vérifier que la mise à jour est marquée "visible par le client"
- Solution : Confirmer que l'email n'est pas bloqué/spam

**Problème : Impossible de télécharger des pièces jointes**

- Solution : Vérifier la taille du fichier (max 10 Mo)
- Solution : Vérifier que le type de fichier est autorisé
- Solution : Vérifier les permissions du répertoire de téléchargements

**Problème : Actions rapides de service ne fonctionnent pas**

- Solution : Vérifier que le service est correctement lié au ticket
- Solution : Vérifier que l'utilisateur a les permissions de service requises
- Solution : Confirmer que les données du service ont été chargées avec succès