Guide des opérations OmniCRM

OmniCRM est la solution BSS/OSS complète d'Omnitouch pour les fournisseurs de services mobiles et de ligne fixe. Une plateforme complète qui gère tout, de l'intégration des clients à la facturation, à l'approvisionnement et au 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** - Approvisionnez votre premier service
4. **Traiter un paiement** - Gérer les paiements des clients
5. **Recharger des services** - Ajouter du crédit aux comptes clients

Tâches quotidiennes :

- **Gestion des services** - Gérer les services clients
- **Voir l'utilisation** - Vérifier les soldes et l'utilisation
- **Modifier les services** - Changer les configurations de service
- **Générer des factures** - Créer et envoyer 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** - Automatiser l'approvisionnement
5. **Configurer des utilisateurs** - Créer des comptes et attribuer des rôles

Sujets avancés :

- **Gestion des stocks** - Gérer les actifs
- **Intégration API** - Connecter des systèmes externes
- **Configuration de la sécurité** - Configurer la 2FA et les autorisations

Pour les clients

Utilisation du portail de self-care :

- **Accéder à votre compte** - Connectez-vous et naviguez
- **Voir les services** - Consultez vos services actifs
- **Vérifier l'utilisation** - Surveiller les données et les soldes
- **Payer des factures** - Effectuer des paiements en ligne
- **Recharger des services** - Ajouter 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 des services et les soldes	Utilisation des services
Traiter un paiement	Traiter un paiement
Générer une facture	Gestion des factures
Recharger un service	Recharger et Recharge
Gérer l'inventaire	Système d'inventaire
Créer un produit	Cycle de vie des produits
Écrire un playbook d'approvisionnement	Playbooks Ansible
Configurer des comptes utilisateurs	RBAC
Activer la 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 des produits de bout en bout
- **Guide des playbooks Ansible** - Maîtriser l'automatisation de l'approvisionnement

Gestion des clients

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

Opérations de service

- Gestion des services - Vue d'ensemble
- Ajout de services - Workflow d'approvisionnement
- Attribution de plans - Attribution de plan
- Modification des services - Apporter des modifications
- Utilisation des services & Soldes - Surveiller l'utilisation
- Recharger & Recharge - Ajouter des crédits

Facturation & Finance

- Vue d'ensemble de la facturation - Guide complet de facturation
- 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 - Configurer OmniCRM
- Gestion des stocks - Suivi des actifs
- Clés API - Gestion de l'accès à l'API

Sécurité & Accès

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

Intégrations

- Intégration Mailjet - Service de messagerie
- Documentation API - Référence API REST

Orienté client

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

Référence

- Journal des modifications - Historique des versions

Qu'est-ce qui rend OmniCRM spécial ?

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

Approvisionnement de service automatisé

Oubliez la configuration manuelle - OmniCRM utilise **l'automatisation Ansible** pour approvisionner des services en quelques secondes. Que vous activiez une carte SIM, configuriez un équipement client ou mettiez en place un forfait complexe, le système s'en charge automatiquement.

Ce qu'il fait :

- Approvisionne des services d'un simple clic
- Configure automatiquement l'équipement réseau
- Crée des comptes de facturation en temps réel
- Envoie des e-mails de bienvenue et des notifications SMS
- Rétablit automatiquement si quelque chose échoue

En savoir plus sur l'approvisionnement | **Voir le guide des playbooks Ansible**

Facturation et paiements intelligents

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

Ce qu'il fait :

- Suivi et tarification des utilisations 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

Explorer 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 :

- Suivre les clients, les contacts et les emplacements de service
- Gérer plusieurs services par client
- Stocker des attributs et des métadonnées personnalisées
- Étiqueter les clients pour l'organisation
- Historique d'activité complet et journaux d'audit
- Intégration Google Maps pour la géocodage des sites

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

- Suivre les cartes SIM, l'équipement et les accessoires
- Gérer les numéros de téléphone et les blocs d'adresses IP
- Attribution automatisée lors de l'approvisionnement
- Capacités d'importation et d'exportation en masse
- Stockage de configuration d'équipement
- Modèles d'inventaire personnalisés

Documentation du système d'inventaire

Portail de self-care pour les clients

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-care | **Fonctionnalités de service client**

Sécurité d'entreprise

Construit 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 e-mail pour les changements de compte
- Journalisation complète des audits via le journal d'activité
- Gestion des sessions et contrôles de délai d'expiration

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

Intégrations puissantes

OmniCRM s'intègre parfaitement aux services leaders du secteur :

- **CGRateS** - Moteur de facturation et de tarification en temps réel pour la facturation de qualité télécom
- **Stripe** - Traitement des paiements sécurisé et stockage de cartes
- **Mailjet** - Livraison d'e-mails professionnels avec modèles

- **Google Maps** - Validation d'adresse et géocodage pour des emplacements de site précis
- **API RESTful** - Intégrer avec vos systèmes existants

Documentation API | **Intégration Mailjet**

Catalogue de produits flexible

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

Types de produits pris en charge :

- **Autonome** - Services uniques (forfaits mobiles, forfaits Internet)
- **Forfaits** - Offres combinées (Internet + TV + téléphone)
- **Suppléments** - Services supplémentaires (recharges de données, appels internationaux)
- **Promotions** - Offres spéciales et réductions

Chaque produit peut avoir sa propre automatisation d'approvisionnement, 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 de diffusion cellulaire

Recherche et navigation puissantes

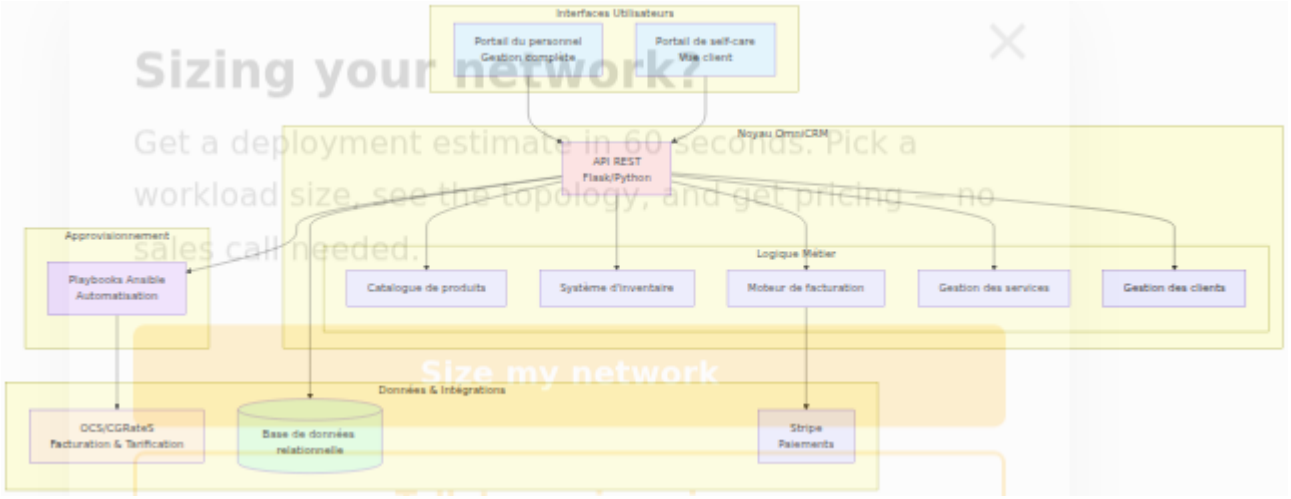
Trouvez n'importe quoi instantanément avec **la recherche globale** à travers les clients, les services, les factures, et plus encore.

Fonctionnalités de recherche :

- Rechercher 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 approvisionnez 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 de l'approvisionnement

Le secret d'OmniCRM est son **système d'approvisionnement automatisé** :



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

Plongée approfondie : Playbooks Ansible | **Système d'approvisionnement**

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 de branche)
└── Services
    ├── Internet - Siège
    ├── Forfait mobile - John Smith
    └── VoIP - Bureau de branche
```

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

L'approvisionnement 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 d'approvisionnement en passant par les attributs personnalisés - adaptez OmniCRM pour correspondre à vos processus commerciaux exacts.

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, la 2FA, le 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 créer 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 supplémentaire de sécurité 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 <rbac>` pour les permissions de gestion de la 2FA, `Authentication Flows <authentication_flows>` 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 avoir saisi le nom d'utilisateur/mot de passe, l'utilisateur 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 sûr

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 est indisponible :

- Chaque code est à usage unique
- Conservez 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 alors 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"
}
```

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

Régénérer les codes de secours

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

Nécessite une authentification. Retourne 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 requise ? 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 actuelle de 30 secondes ainsi que des fenêtres précédentes/suivantes 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`.

Vue d'ensemble

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 valides 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 :

Autorisation requise : `MANAGE_API_KEYS` (rôle admin)

Vue de la 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 Provisioning", "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 des raisons de 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 de Tableau de Bord

En haut de la page, des statistiques résumées 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 suivants
- **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 Provisioning")
>
> Description : _____
> (Optionnel - but de cette clé)
>
> Date d'expiration : [Sélecteur de Date]
> (Optionnel - laissez vide pour aucune expiration)
>
> Permissions : ☐ Voir les Clients ☐ Créer des Clients ☐ Voir les Services ☐ Créer des Services ☐ Provisioning ☐ Voir l'Inventaire ☑ Admin (toutes les permissions)
>
> [Annuler] [Générer la Clé]

Directives pour les Champs :

Nom (obligatoire)

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

Description (optionnelle)

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

Date d'expiration (optionnelle)

- 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 :** Attribuez 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]

Avertissement

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 la 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 le nom de n'importe quelle 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

Régénérer 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

Avertissement

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

Révoquer (Supprimer) 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 retournent `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

Modifier les 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 **"Enregistrer 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 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 aucune expiration définie)
- Pas révoquée manuellement
- Affichée avec un badge vert

Expirant Bientôt

- Actif mais expirera dans les 30 jours suivants
- 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 invalidé
- 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 pour afficher :

- Toutes les Clés
- Actives Uniquement
- 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 Clés 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 Clés API

Dans le CRM :

- Les clés sont hachées avant le 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** - Accordez uniquement les permissions nécessaires
- Évitez de créer des clés admin sauf si absolument nécessaire
- Utilisez des clés distinctes pour différents systèmes/buts
- Révisez régulièrement les permissions

Surveillance et Audit

- Surveillez l'utilisation des clés API via les journaux d'activité
- Configurez des alertes pour des modèles d'accès inhabituels
- Révisez régulièrement les horodatages "dernier utilisé"
- 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 utilisée
- **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 la Clé API

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

Mettre à Jour la 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 Provisioning

Créez une clé API pour votre serveur de provisioning Ansible :

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

5. Expiration : 1 an
6. Copiez la clé et ajoutez-la à `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 les Clients, Voir les Transactions, Voir les 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 les Services, Voir le Provisioning
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 d'auto-service des clients :

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, voir `Customer Tags <administration_tags>`. Pour des informations de base sur les clients, voir `Customers, Contacts, Sites & Services <basics_customers>`.

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 fonctionnalité)

Attributs vs. Champs Standards

Utilisez des Attributs Lorsque :

- 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 Lorsque :

- 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/nombre 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 un 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** (poubelle)
5. Confirmez la suppression dans la fenêtre contextuelle

Avertissement

La suppression d'attributs utilisés par des intégrations, des workflows 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 programme via l'API :

Créer ou Mettre à Jour un Attribut

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

Permission 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}`

Permission 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}`

Permission 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}`

Permission 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}`

Permission Requise : `delete_customer_attribute`

Réponse :

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

Opérations d'Attributs en Masse

Gestion de Plusieurs Attributs

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

```
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 des 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/").jso

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 des 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 d'attributs et de leurs objectifs
- Documentez quels systèmes/intégrations dépendent d'attributs spécifiques
- Incluez des exemples de valeurs valides

5. Migration et Nettoyage

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

Exemples de Flux de Travail

Intégration d'Onboarding

Lors de la migration des clients d'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 Fonctionnalité

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 du 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 du 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-les en plusieurs attributs ou utilisez le champ de 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.

Vue d'ensemble

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, provisionnement

Configuration Frontend :

- **Fichier :** `OmniCRM-UI/.env`
- **Format :** Variables d'environnement
- **Nécessite :** Reconstruction de l'UI après modifications
- **Utilisé pour :** Branding, fonctionnalités, services externes

---

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 des identifiants réels dans le contrôle de version. Utilisez une configuration spécifique à l'environnement ou une gestion des secrets.

Remarque : Cela doit correspondre à vos identifiants de base de données dans `.env`.

Types de Services

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

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

Ces types sont utilisés dans tout le système pour :

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

Ajoutez ici des types de services personnalisés 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 le provisionnement des abonnés
- `apn_list` - Liste séparée par des virgules des ID APN disponibles pour le provisionnement

Utilisé pour : Provisionnement de réseau mobile et authentification des abonnés.

Configuration des Modèles 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 Réinitialisation de Mot de Passe
  api_crmCommunicationUserPasswordResetSuccess:
    from_email: "security@yourcompany.com"
    from_name: "Sécurité de Votre Entreprise"
```

```yaml
    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"
```

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 du 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 de Provisionnement

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

But :

- `failure_list` - Adresses email notifiées lorsque le provisionnement 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 de provisionnement

Configuration de Facture

```
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 du 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

```yaml
ocs:
  ocsApi: 'http://10.179.2.142:8080/api'
  ocsTenant: 'mnc380.mcc313.3gppnetwork.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 : Chargement 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 des détails d'utilisation.

Clé Secrète JWT

```
jwt_secret: 'CHANGE_ME_ON_DEPLOYMENT'  # Généré automatiquement 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 à travers 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 cryptographiquement sécurisée
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 a le secret peut forger 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, garder confidentiel)
- `publishable_key` - Clé publique 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 <integrations_payment_vendors>` pour des détails de configuration.

Clés API et Liste Blanche d'IP

Définir des clés API avec accès basé sur les rôles et 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 d'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 :

- Utiliser 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'utiliser la liste blanche d'IP que pour des réseaux internes de confiance
- Ne PAS utiliser les IP localhost (127.0.0.1, ::1)
- Utiliser des clés API à la place pour l'accès externe
- Envisager des règles de pare-feu comme protection supplémentaire

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

Configuration Frontend (.env)

L'UI React est configurée via des variables d'environnement dans `OmniCRM-UI/.env`.

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

Clés API et Services Externes

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

# Désactiver le lancement automatique du navigateur lors du
démarrage npm
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 publique 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 : Ce sont des variables **de temps de construction**. Changer une valeur `VITE_*` nécessite de reconstruire et de redémarrer l'UI. Voir Guide du Système de Paiement - Configuration des Fournisseurs 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="Téléphones avec Coquilles"
```

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 en libre-service (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 restreints ou isolés

- **Confidentialité** - Pas de requêtes externes vers Google Fonts, Adobe Fonts ou d'autres CDN
- **Performance** - Chargement plus rapide sans dépendances externes
- **Sécurité** - Pas de suivi tiers ou 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 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: 'Votre Nom de Police';
  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: 'Votre Nom de Police';
  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=Votre Nom de Police
```

Directives de Poids de Police :

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

Liens Rapides vers les 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 d'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 analytiques. 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 d'OmniCRM.

```
# Intégration du Tableau de Bord Grafana
# IDs de Tableau de Bord (séparés par des virgules) - obtenez-les
à partir des URLs /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 d'OmniCRM sous le menu "Tableaux de Bord". Les utilisateurs peuvent cliquer pour voir les tableaux de bord intégrés tout en gardant la navigation d'OmniCRM visible.

Exemple :

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

- ID du tableau de bord : `system-overview-123`
- Étiquette du 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 affichées 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 dans **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 des 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 dans toute 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'
  });
  // Continuer 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** - Divulguez 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
Les vues de pages ne sont pas suivies	UI non reconstruite après un changement .env	Exécutez `npm run build` et redémarrez l'UI
Événements non apparaissant	Bloqueur de publicité activé	Testez avec le bloqueur de publicité 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 sur la performance minimal (< 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 libre-service.

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 des prix backend. Voir `features_topup_recharge` pour des détails complets de configuration.

Branding et Personnalisation Visuelle

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

Fichiers de Logo et Système de Repli

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

Logos Par Défaut (toujours présents) :

- `DefaultLogoDark.png` - Logo thème sombre (utilisé sur des arrière-plans clairs)
- `DefaultLogoLight.png` - Logo thème clair (utilisé sur des arrière-plans 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 Repli de Logo :

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

```
// Depuis Header.js
const tryImport = (filename) => {
    try {
        return require(`../assets/images/omnitouch/${filename}`);
    } catch (err) {
        return null; // Retourne au 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, arrière-plans sombres, carrousel d'authentification

Remplacement des 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 du mode clair (doit afficher `logoDark.png`)
   - Vérifiez l'en-tête du mode sombre (doit afficher `logoLight.png`)
   - Vérifiez la barre latérale réduite (doit afficher `logoSm.png`)
   - Vérifiez l'écran de connexion (doit afficher `logoLight.png`)

Meilleures Pratiques de Conception de Logo :

- **Contraste** - Assurez-vous que les logos sont visibles sur des arrière-plans clairs et sombres
- **Simplicité** - Les logos doivent être reconnaissables à des tailles réduites
- **Format** - Utilisez PNG avec des arrière-plans transparents
- **Retina** - Fournir une résolution 2x pour les écrans 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" />
```

Personnalisation des 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 :

```css
/* 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 du Modèle :**
    - Le nom du fichier modèle est spécifié dans `crm_config.yaml` sous `invoice.template_filename`
    - Le modèle est chargé à partir du 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
    - Des variables comme `{{ invoice_number }}`, `{{ total_amount }}`, etc. sont remplacées
    - Le HTML rendu est enregistré dans `invoice_templates/rendered/` pour débogage

4. **Génération de PDF :**
    - WeasyPrint convertit le HTML rendu en PDF
    - Le PDF prend en charge 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 de l'intégrité
    - Les demandes ultérieures 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 Morceau 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 la 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 norfone_invoice_template.html
yourcompany_invoice_template.html
```

2. **Éditez 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 la taxe, instructions de paiement, etc.)
3. **Mettez à Jour la Configuration :**

    Éditez `crm_config.yaml` :

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

4. **Testez la Génération de Facture :**
    - Créez une facture de 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 :

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

Système de Mise en Cache de PDF

Pour améliorer les performances, OmniCRM met en cache les PDF 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 la vérification de l'intégrité
- Inclut le nom de fichier, l'horodatage de création, l'horodatage de dernier accès

Gestion Manuelle du Cache :

```
# Dans l'API OmniCRM ou le 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 de plus de 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 le 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. Éditez `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. Éditez `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 identifiants
- **Utilisez des mots de passe forts** - Minimum 16 caractères avec des majuscules, des chiffres et des symboles
- **Faites tourner régulièrement les identifiants** - Surtout pour les déploiements en production
- **Restreignez l'accès à la base de données** - Utilisez des listes blanches d'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** - Attribuez les rôles minimaux nécessaires
- **Utilisez la liste blanche d'IP avec parcimonie** - Préférez les clés API pour une meilleure sécurité

Maintenance

- **Documentez les changements** - Tenez un changelog des modifications de configuration
- **Sauvegardez les configs** - 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 une 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 des arrière-plans colorés
- **Tests mobiles** - Vérifiez le branding sur des 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
- **Fix :** Redémarrez les services API/UI après les changements de configuration

Erreurs de syntaxe YAML

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

Échec de la connexion à la base de données

- **Cause :** Identifiants incorrects ou serveur injoignable

- **Fix :** Vérifiez que la base de données fonctionne, que les identifiants sont corrects

Les paiements Stripe ne fonctionnent pas

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

Les emails ne sont pas envoyés

- **Cause :** Identifiants Mailjet invalides ou IDs de modèle
- **Fix :** Vérifiez la clé/secrète API Mailjet, vérifiez que les IDs de modèle existent

Échec de la 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 n'apparaissant pas dans le PDF :

- Utilisez des chemins de fichiers absolus : `file:///full/path/to/image.png`
- Assurez-vous que les fichiers d'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 :

- Intégrez tout le CSS (les feuilles de style externes ne sont pas prises en charge)
- 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'invalidant 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 identifiants 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'��chec de provisionnement
  - [ ] Nom de fichier du modèle de facture
  - [ ] URL de base du CRM (accessible publiquement)
  - [ ] Points de terminaison OCS/CGRates
  - [ ] Configuration SMSC
  - [ ] Générer une nouvelle clé secrète JWT
  - [ ] Clés Stripe (en direct, pas de test)
  - [ ] Clés API et liste blanche d'IP

Configuration Frontend

- [ ] Copiez `OmniCRM-UI/.env.example` à `OmniCRM-UI/.env`
- [ ] Définissez la clé API Google Maps
- [ ] Définissez la clé publique 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 et symbole de devise
- [ ] Définissez la couleur principale de la marque
- [ ] 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 un 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 de production (pas de clés de test)
- [ ] Faites tourner les clés API Mailjet
- [ ] Activez les règles de pare-feu
- [ ] Configurez la liste blanche d'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 le provisionnement de 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 Plan d'Action
- [ ] 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 </rbac>`
- `Produits et Services </concepts_products_and_services>`
- `Provisionnement Ansible </concepts_ansible>`
- `Gestion des Inventaires </administration_inventory>`
- `Factures Clients </payments_invoices>`
- `Authentification à Deux Facteurs </2fa>`
- `Service Client et Imitation </customer_care>`
- `Intégrations de Fournisseurs de Paiement <integrations_payment_vendors>`
- `administration_api_keys` - Gestion des clés API
- `integrations_mailjet` - Intégration des emails
- `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 Client <administration_attributes>` pour stocker des métadonnées personnalisées, et `Tags Client <administration_tags>` 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.

Note

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 <guide_product_lifecycle>`.

Objectif

L'inventaire OmniCRM sert plusieurs objectifs clés :

1. **Provisionnement de 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 le stock physique, tel 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 à 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 cours d'utilisation.

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

Exemple de maisons desservies 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 sont pas encore desservies.

Lorsqu'un client s'inscrit à 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 en utilisant **InventoryTemplates**. Ces modèles permettent de catégoriser et de gérer les articles de manière structurée, garantissant la cohérence dans la façon dont les articles sont suivis.

Liaison aux 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: "['Carte SIM', 'Numéro de mobile']"` nécessite que deux modèles d'inventaire nommés exactement "Carte SIM" et "Numéro de mobile" 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 <guide_product_lifecycle>`.

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

   Note

   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 de texte de l'article 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 de texte de l'article 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 de texte de l'article 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** : Permettre au personnel de sélectionner ce type d'inventaire dans les menus déroulants
- **Autoriser le client à sélectionner dans le menu déroulant** : Permettre aux clients de 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**

Avertissement

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 afficheront les nouveaux noms d'étiquettes.

Précaution

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 de texte de l'article 1 : "ICCID"
- Étiquette de texte de l'article 2 : "IMSI"
- Étiquette de texte de l'article 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 de texte de l'article 1 : "Numéro de téléphone"
- Étiquette de texte de l'article 2 : "Région géographique"
- Étiquette de texte de l'article 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 de texte de l'article 1 : "Adresse MAC"
- Étiquette de texte de l'article 2 : "Numéro de série"
- Étiquette de texte de l'article 3 : "Version du firmware"
- Étiquette de texte de l'article 4 : "Fabricant"
- Étiquette de texte de l'article 5 : "Modèle"

Modèle ONT GPON :

- Article : "ONT GPON"
- Coût de gros : 45,00
- Coût de détail : 0,00 (inclus avec le service)
- Étiquette de texte de l'article 1 : "Numéro de série"
- Étiquette de texte de l'article 2 : "Adresse MAC"
- Étiquette de texte de l'article 3 : "Emplacement PON"
- Étiquette de texte de l'article 4 : "Modèle"

Création et gestion des articles d'inventaire

Une fois qu'un **InventoryTemplate** est défini, des articles **Inventory** 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.
- **Vendu ou décommissionné** : Une fois vendu, 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 le statut et l'historique de chaque article.

Les services peuvent être liés à un article **Inventory** donné, permettant un suivi facile des articles associés aux clients ou services.

Une fois qu'un article d'inventaire a été attribué à un client, les playbooks 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 des articles en cours d'utilisation et de ceux 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 de 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 par le 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 de l'article (à partir 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 équipements et sites réseau)

Utilisé pour suivre les emplacements de déploiement physique, en particulier pour les équipements réseau fixes (CPE, ONT, modems) ou 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 la saisie semi-automatique d'adresse, ces champs sont automatiquement remplis :

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

Ces champs permettent :

- De cartographier les emplacements d'inventaire sur une vue de carte
- De calculer la proximité pour la qualification de service

- D'analyser la couverture pour la planification du réseau
- D'optimiser les itinéraires pour l'envoi de techniciens sur le terrain

Gestion des appareils et URLs d'accès

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

- **management_url** - URL d'accès pour l'article d'inventaire
  - **Équipements 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 détaillée de l'article d'inventaire** : QR code 128x128 affiché à côté de l'URL
  - **Tableau d'inventaire de service** : 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 de l'appareil sans taper d'adresses IP
  - **Activation 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 de l'appareil ou aux portails clients
- **management_username** - Nom d'utilisateur admin pour l'accès à l'appareil

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

Gestion de la configuration

Pour les appareils avec des fichiers de configuration :

- **config_content** - Contenu complet du fichier de configuration (stocké en tant que texte)
  - Utile pour la sauvegarde, la gestion des versions et la récup��ration 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 sur 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": "89612345678901234 56",
  "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": "89612345678901234 57",
  "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 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 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"
}
```


Note

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 URLs web)

Cas d'utilisation :

- **Activation eSIM** (Exemple 2) : Les clients scannent le QR code pour installer le profil eSIM sur leur appareil
- **Accès aux équipements 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 par le 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)
- **Endommagé/Utilisé** → **Nouveau** - Après remise à neuf ou réparation

Filtrer et rechercher l'inventaire par état permet 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 en clé-valeur, voir `Attributs Client <administration_attributes>`.

Les étiquettes sont affichées sous forme de pastilles sur la page de profil du client, et la couleur de la pastille est personnalisable avec le lien.

Un cas d'utilisation courant est de marquer 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, afin qu'elles puissent 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 vue d'ensemble 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 vue d'ensemble du client
2. Cliquez sur l'onglet **Étiquettes**
3. Cliquez sur le bouton **Ajouter 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** (optionnel) : URL qui s'ouvrira lorsque l'étiquette est 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 2099-01-01)
5. Prévisualisez votre étiquette dans la section de prévisualisation

6. Cliquez sur **Créer Étiquette**

Édition d'une Étiquette

Pour modifier une étiquette existante :

1. Accédez à la page de vue d'ensemble 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 si nécessaire
6. Cliquez sur **Mettre à Jour l'Étiquette**

Suppression d'une Étiquette

Pour supprimer une étiquette :

1. Accédez à la page de vue d'ensemble 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 par programmation 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 :

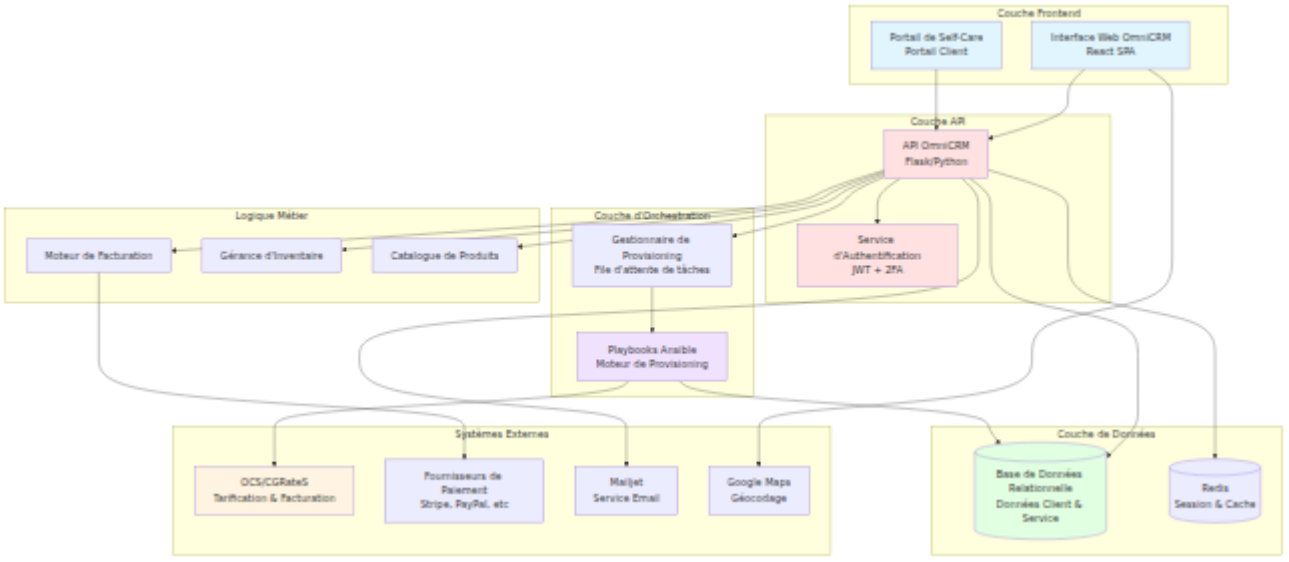
```
DELETE /crm/tag/tag_id/{tag_id}
```

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 un 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 de services, la facturation
- Mises à jour du statut de provisioning en temps réel
- Contrôle d'accès basé sur les rôles

Portail de Self-Care

- Portail orienté client
- Gestion des services et suivi de l'utilisation
- Consultation et paiement des factures
- Base de code partagée avec l'interface du personnel, vues différentes

2. Couche API

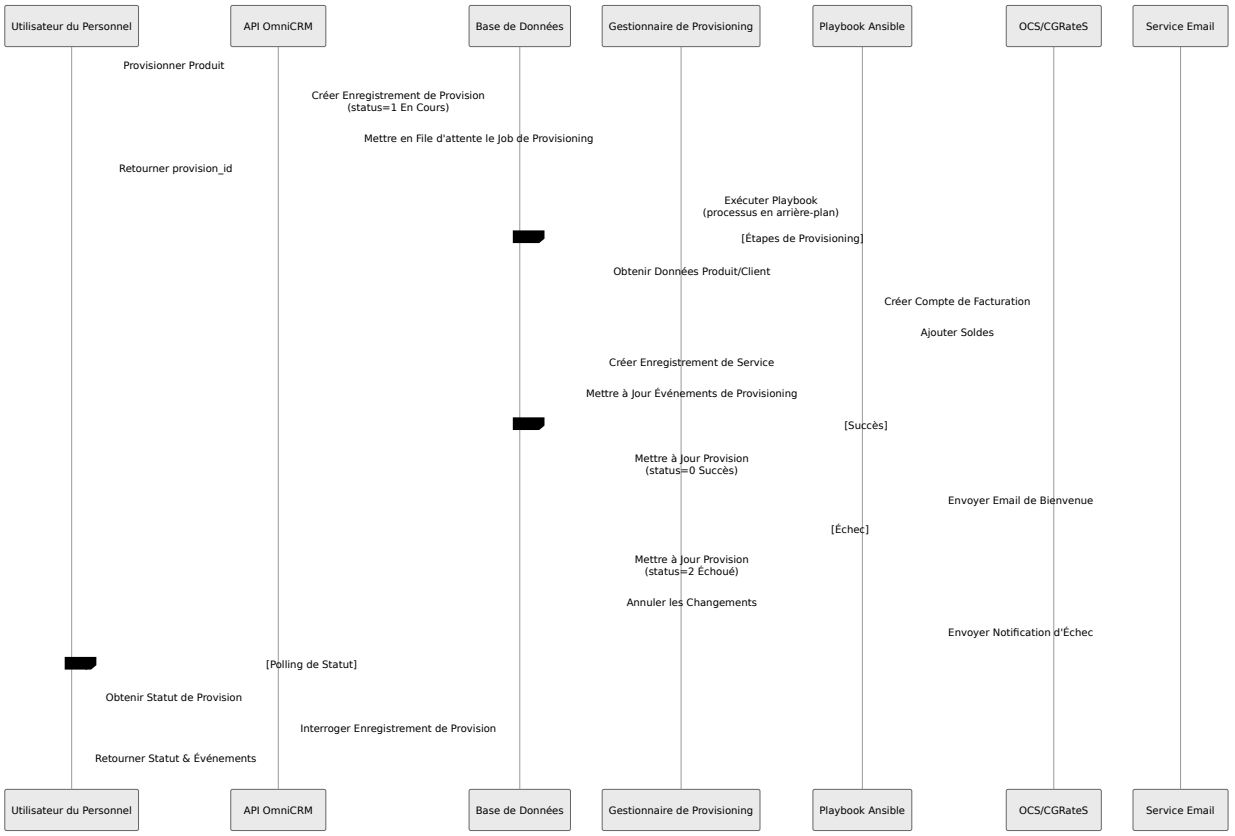
API OmniCRM (Flask/Python)

- 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 Clés de l'API :

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

3. Système de Provisioning

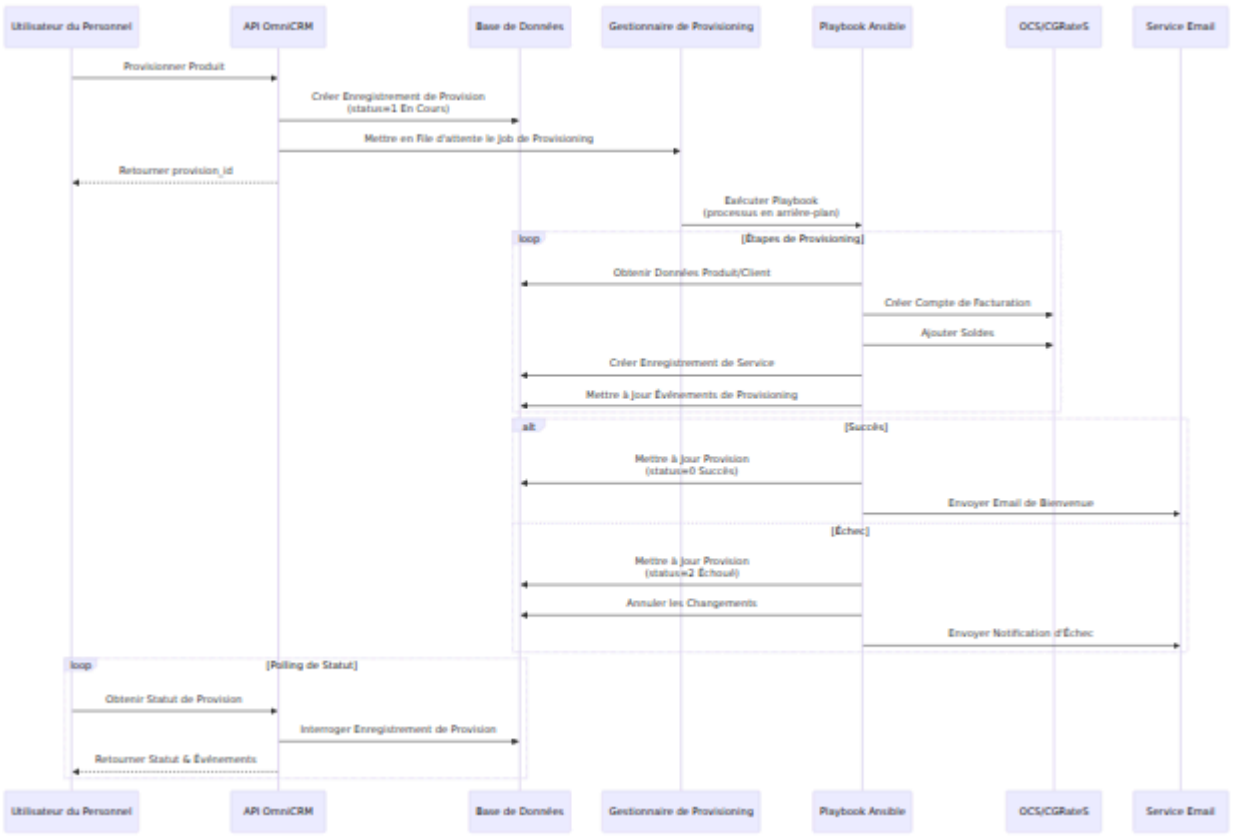


4. Facturation & Tarification

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 principaux 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 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 de Localisation Ouverte

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)	Lieu de 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 service

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 associé
title	String	Description de la transaction
retail_cost	Float	Montant
tax_percentage	Float	Taux de taxe
tax_amount	Float	Taxe calculée
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 travaux 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)	Tâche de provisioning 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 (plusieurs à plusieurs)
- Les rôles ont plusieurs permissions (plusieurs à plusieurs)
- Les utilisateurs peuvent se lier 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 des livraisons

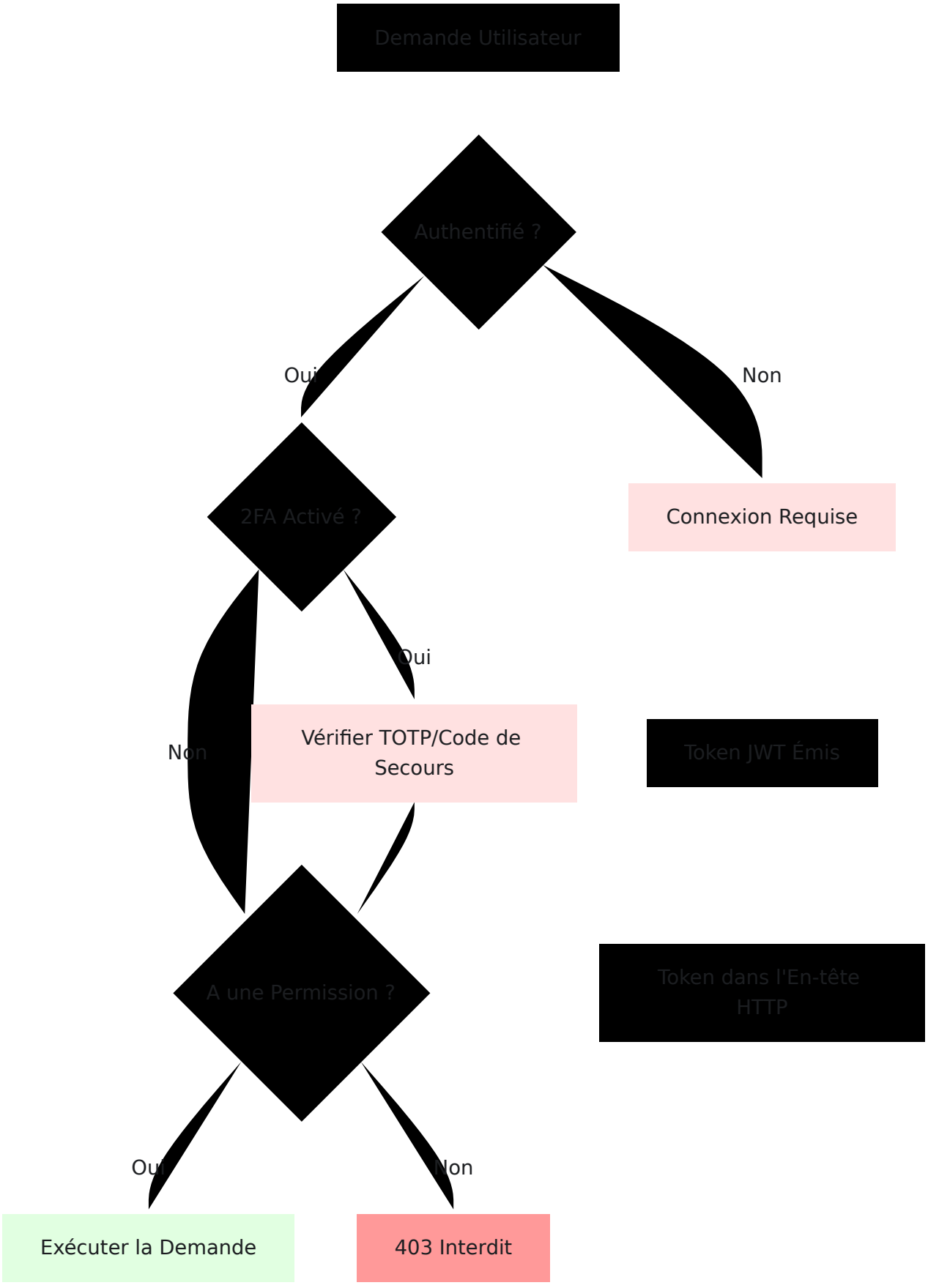
Google Maps

- Autocomplétion d'adresse
- 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 calendriers

Architecture de Sécurité

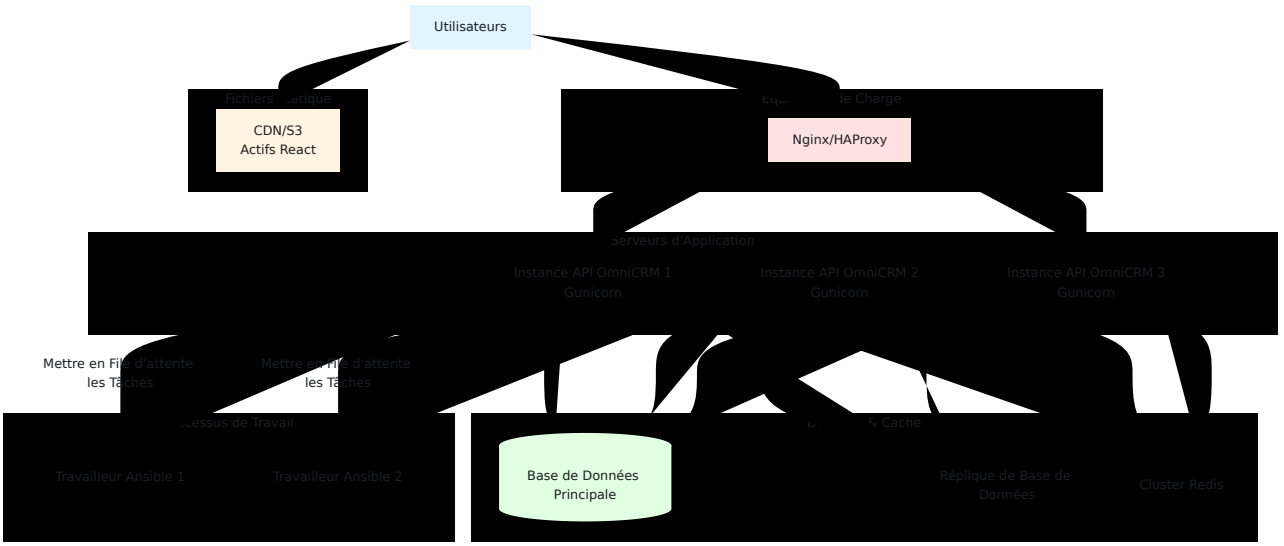


Fonctionnalités de Sécurité :

- 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 des emails
- 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 demandes
- 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 tâches en arrière-plan
- Plusieurs travailleurs Ansible
- Gestion de la file d'attente des tâches
- 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 demandes
- Taux de succès/échec de provisioning et durée des tâches
- Performance des requêtes de base de données et utilisation du pool de connexions
- Santé des intégrations externes (OCS, Stripe, Mailjet)
- Exécution des tâches 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 récupération, 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 de l'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 des 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 : `Self-Care Portal <self_care_portal>` pour la connexion des clients et l'accès au portail, `RBAC <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 optionnel
- **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 optionnelle avec Google, Apple, Facebook (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. Saisir **l'adresse e-mail** (e-mail du personnel ou du client)
2. Saisir **le mot de passe**
3. Optionnel : Cocher **"Se souvenir de moi pendant 30 jours"** pour une session prolongée
4. Cliquer sur **"Connexion"**

Ce qui se passe ensuite :

- **Sans 2FA :** Utilisateur connecté immédiatement, navigué en fonction du rôle :
  - **Clients** → Portail de libre-service (`/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 d'œil (□)** pour basculer la visibilité du mot de passe.

Connexion avec 2FA

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

Utilisation du code d'authentificateur :

1. Ouvrir l'application d'authentificateur (Google Authenticator, Authy, etc.)
2. Trouver l'entrée OmniCRM
3. Saisir le code à 6 chiffres
4. Le code se soumet automatiquement lorsque les 6 chiffres sont saisis
5. Si valide, l'utilisateur est connecté et navigué vers le tableau de bord approprié

Utilisation du code de récupération :

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

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

Annuler :

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

Connexions sociales (optionnel)

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. Actuellement, un message "bientôt disponible" s'affiche (mise en œuvre de la connexion sociale en cours).

Lien de réinitialisation de mot de passe

Cliquez sur le lien **"Mot de passe oublié ?"** pour initier le flux de réinitialisation de 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'authentificateur standard.

Accéder à la configuration de 2FA

Depuis le profil utilisateur ou les paramètres :

Remarque pour les clients :

Les utilisateurs ayant le rôle de 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]

Saisissez 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. **Télécharger l'application d'authentificateur** (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 :**
   - Ouvrir l'application d'authentificateur
   - Appuyer sur "+" ou "Ajouter un compte"
   - Choisir "Scanner le code QR"
   - Pointer la caméra sur 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'authentificateur
- Cliquez sur **"Copier les codes"** pour copier tous les codes dans le presse-papiers

4. **Vérifier la configuration :**
   - Saisir le code à 6 chiffres actuel de l'application d'authentificateur
   - 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. Saisir **l'adresse e-mail**
2. Cliquer 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 verrou)
>
> Mot de passe
>
> Confirmer le mot de passe
>
> [Réinitialiser le mot de passe]

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

2. Saisir à nouveau dans le champ **Confirmer le mot de passe**
3. Cliquer sur **"Réinitialiser le mot de passe"**

Exigences de mot de passe :

- Longueur minimale (généralement 8+ caractères)
- 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 disposant des 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 : Cliquer 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 devra 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 la 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'authentificateur et à tous les codes de sauvegarde :

Étape 1 : Cliquer 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 volontairement 2FA depuis ses paramètres

Remarque de sécurité importante :

Avant de réinitialiser 2FA, l'administrateur doit :

1. Vérifier l'identité de l'utilisateur par d'autres moyens :
    - Vérification d'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 récupéré 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 :

Lorsque 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 de l'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 (si applicable) • Informations de contact du 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 le 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 le 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 récupéré l'accès
- Envisagez d'exiger 2FA pour tous les utilisateurs administratifs

Gestion des utilisateurs :

- Examinez régulièrement 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
  - Vérifiez que la touche de verrouillage des majuscules est désactivée
  - Essayez de réinitialiser le 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 prochain code qui apparaît
  - Utilisez un code de sauvegarde si l'authentificateur ne fonctionne pas
  - Contactez l'administrateur pour réinitialiser 2FA si tout le reste 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 lors de la fermeture du 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 sur le lien rapidement
  - 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'authentificateur 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 :** Autorisations 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 d'autorisations
- **Solution :**
    - Actualisez la page et réessayez
    - Vérifiez les autorisations administratives
    - Vérifiez les journaux API pour les 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 par 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 piste d'audit
- Les actions administratives nécessitent des autorisations 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-mail
- `customer_care` - Portail de libre-service pour les clients

Créer un client

0fT52ZvoZBE

Clients, Contacts, Sites & Services

Nous avons un modèle simple de **Client** qui 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, c'est probablement la même personne que le client lui-même, 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 lieu. 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 <csa_activity_log>`, qui est un enregistrement de tous les changements effectués, `Tags <administration_tags>`, `Attributs <administration_attributes>` pour stocker des métadonnées personnalisées, `Éléments d'Inventaire <administration_inventory>` et des informations financières comme `Transactions <payments_transaction>`, `Factures <payments_invoices>` & `Méthodes de Paiement <basics_payment>`.

Une fois que nous avons créé un client, nous pouvons ensuite `ajouter un service <csa_add_service>` à ce client, qui est la chose pour laquelle nous les

facturons.

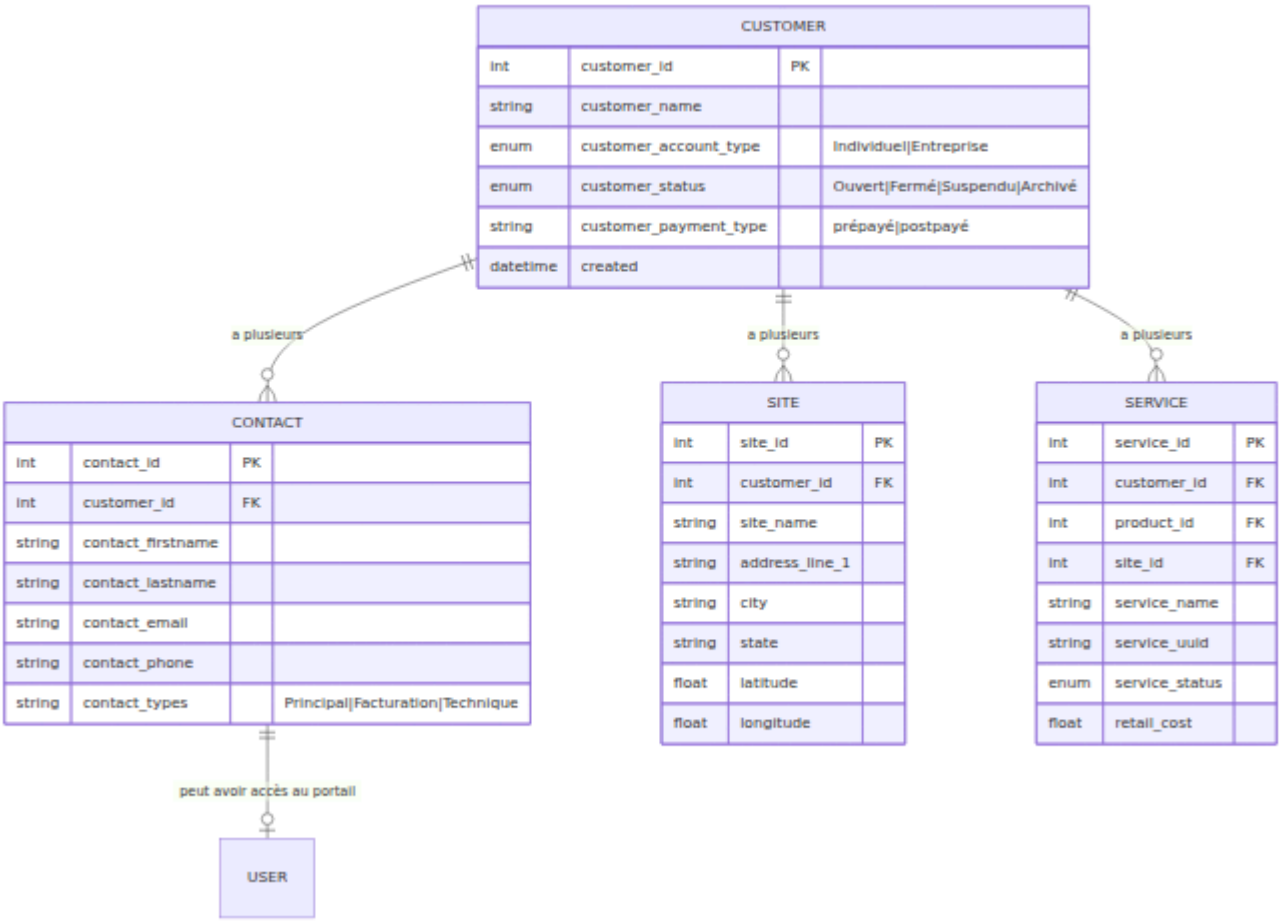
Pour des informations sur la création d'un client, voir Créer un Client.

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 décomposé 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.



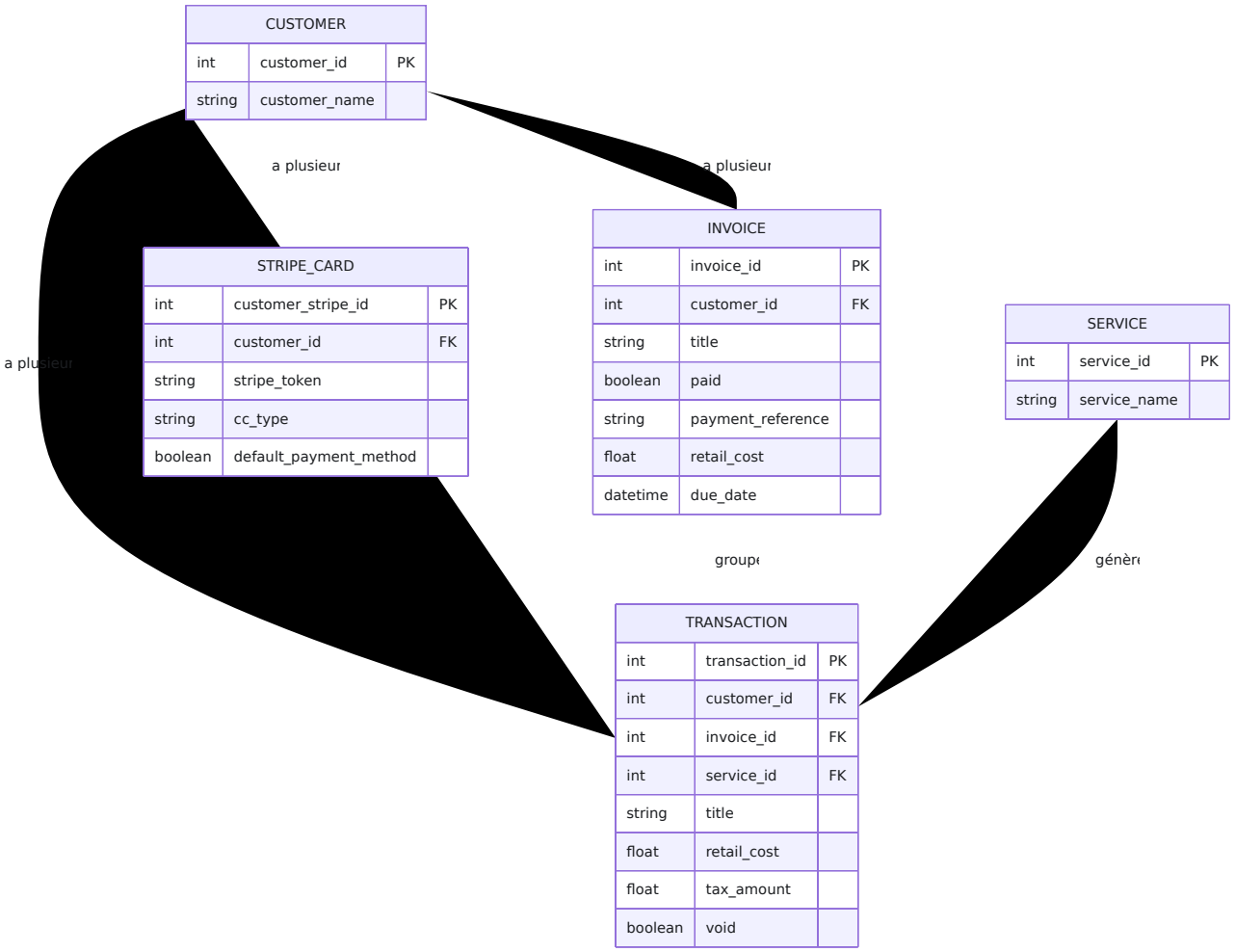
Points Clés :

- 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.

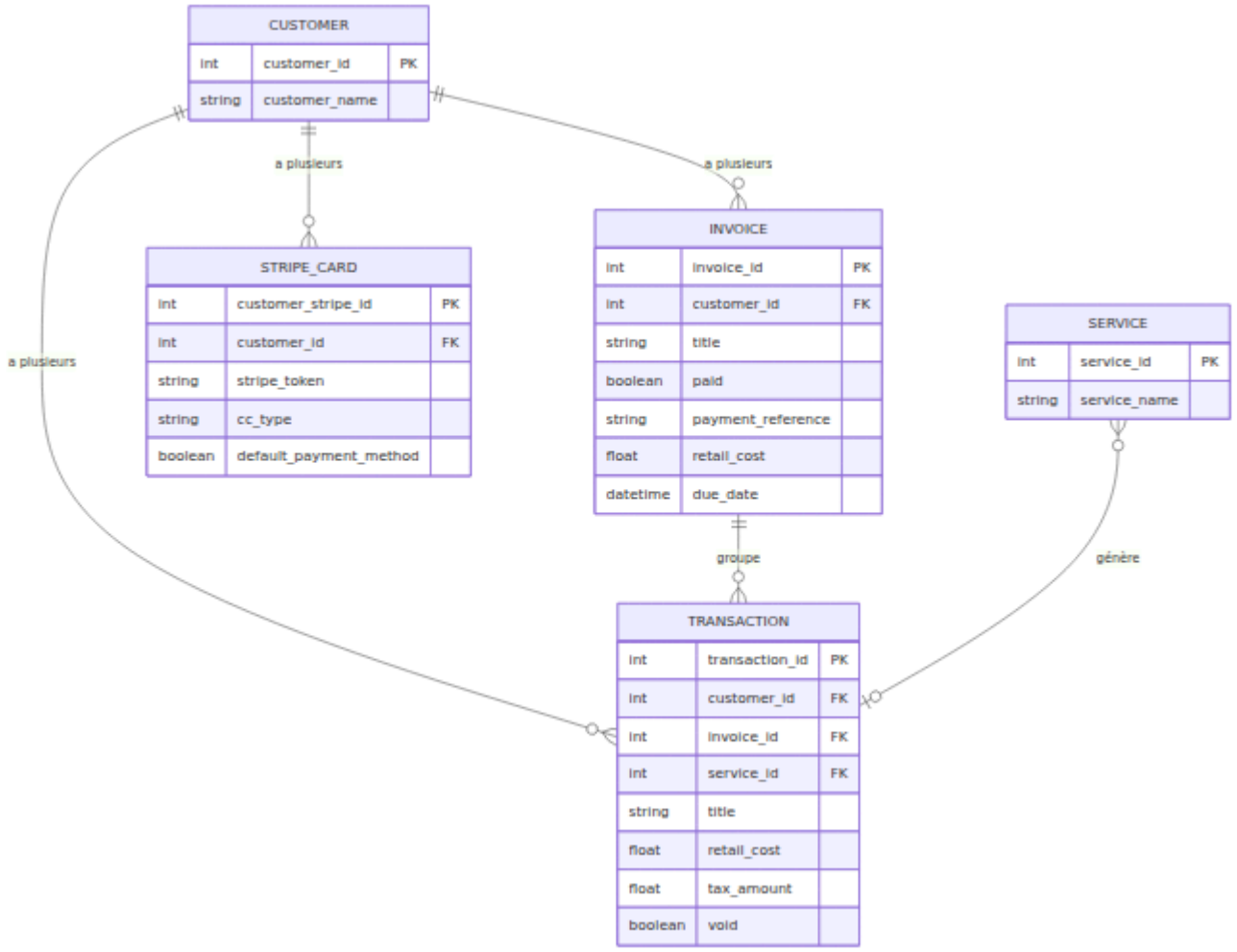


Points Clés :

- 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

Produit & Provisionnement

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

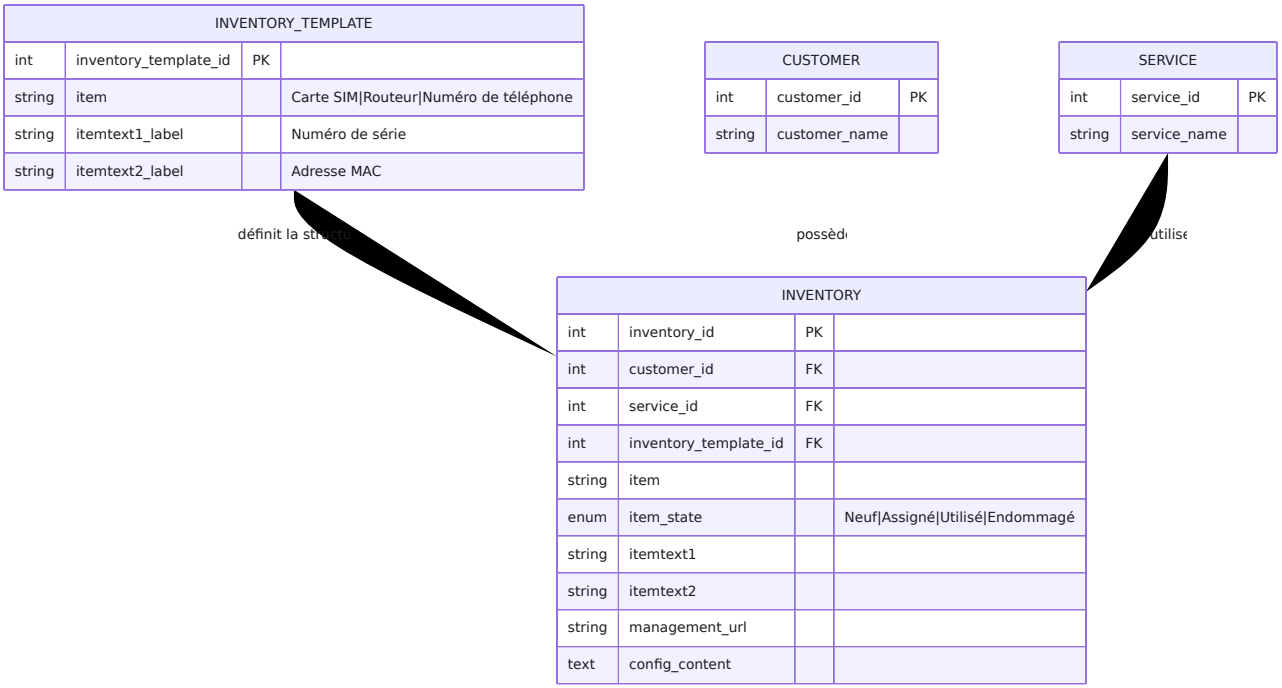


Points Clés :

- 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 (packs)

Système d'Inventaire

Suivez les actifs physiques et virtuels assignés aux clients.

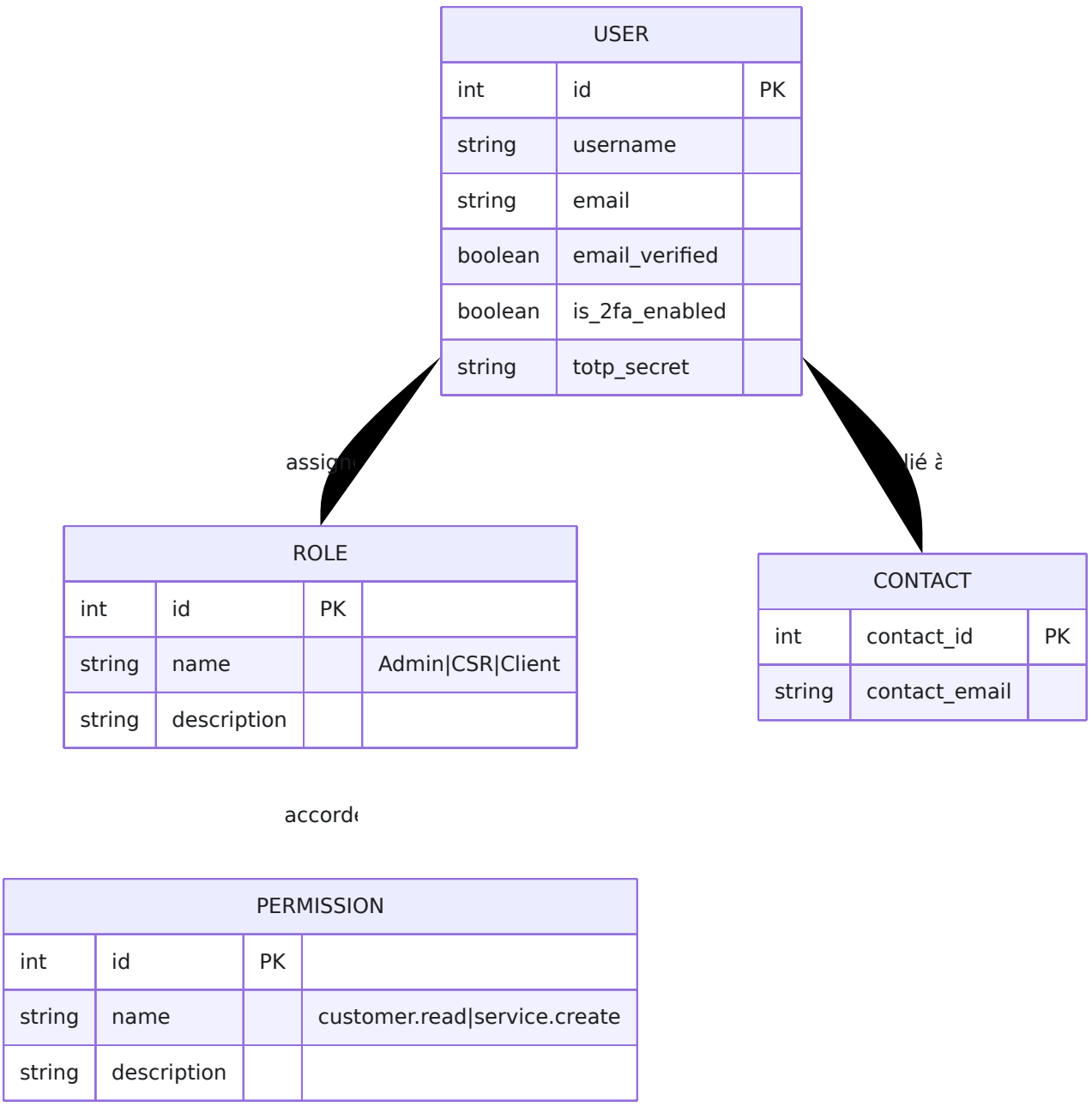


Points Clés :

- Les modèles d'inventaire définissent la structure (champs) pour chaque type d'élément
- Les champs flexibles itemtext1-20 s'adaptent à différents types d'inventaire
- Les éléments peuvent être assignés aux clients et liés aux services
- Stocke les configurations d'équipement et les informations d'identification 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 consultable et filtrable 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, c'est juste un nom et 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 se trouve 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 la manière dont ce client se compare 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 peuvent ensuite être restaurées 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 bon service 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 utilise automatiquement le géocodage des 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 (Recommandé)**
    - 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-prémisse (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 (par exemple, `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)
- **Pays** doit être renseigné (par défaut à `VITE_DEFAULT_COUNTRY` si non défini)
- **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 affectés via `Mailjet <integrations_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 mal 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 Distants

Pour les clients dans les zones rurales où les adresses de rue 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 (par exemple, `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 : "Tournez à gauche au grenier 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 effectuera un géocodage inverse de 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 pouvons vouloir envoyer des notifications de panne uniquement 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 contacts 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 <integrations_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 et les sous-sections de chacune.

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 Clients

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 de configuration, et lorsque le 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 des `Factures <payments_invoices>`, 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 le 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 (tous les 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 : Affichage avec suffixe "k" (par exemple, $1.5k)
- Valeurs supérieures à 1 000 000 : Affichage avec suffixe "M" (par exemple, $2.3M)
- Valeurs supérieures à 1 000 000 000 : Affichage avec 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 les options :

- **Voir les Détails** - Ouvre le modal de détails de la transaction
- **Télécharger la Facture PDF** - Télécharger le PDF (uniquement si facturé)
- **Annuler la Transaction** - Marquer la transaction comme annulée (uniquement si pas facturée)

Types de Transactions

Les transactions se divisent en deux catégories principales :

Transactions de Débit (Frais)

Montants positifs qui augmentent le solde dû par le client :

- **Frais de Configuration 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 de 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 le modal d'ajout de transaction.

Transaction de Débit (Frais) :

Transaction de 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 la 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 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. La transaction est créée dans la base de données
2. Elle apparaît dans la liste des transactions du client
3. Incluse dans le compte des "Transactions Non Facturées"
4. Disponible pour inclusion dans la prochaine génération de factures
5. Une entrée de journal d'activité est créée

Recherche et Filtrage des Transactions

Recherche

Utilisez la barre de recherche pour trouver des transactions. Recherche à travers :

- ID de la transaction
- Titre
- Description
- ID de la 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 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).

Annulation 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 :

- 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"

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 de Débit (Frais)** - La taxe est appliquée aux frais en fonction de :
  - **Pourcentage de Taxe 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 : `coût_de_détail × (pourcentage_de_taxe / 100)`
    - **Format d'Affichage** - Affiché comme : `$10.00 (10%)` dans les listes de transactions
- **Transactions de 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 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 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)
- L'ID de la facture est 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 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 Manuels

1. Le technicien de terrain installe le service

2. Le personnel doit facturer 75 £ de frais d'installation
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 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 : Annulation d'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 de facturation. Si un remboursement est nécessaire, créez une transaction de Crédit à la place.

Transactions dupliquées apparaissant

- **Cause :** Service facturé plusieurs fois ou erreur de provisionnement
- **Solution :** Annulez la ou les transactions dupliquées si non facturées. Si facturées, émettez un crédit.

Transaction n'apparaît 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 l'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 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 Clients

Les `Transactions </payments_transaction>` 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 couvre la facture, 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, lorsqu'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 <integrations_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 sous forme de PDF.

Personnalisation des Modèles de Factures

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 :

- `norfone_invoice_template.html` - Modèle de facture exemple
- `cifi_invoice_template.html` - Exemple de modèle alternatif

Configuration :

Le modèle de facture actif est spécifié dans `OmniCRM-API/crm_config.yaml` :

```
invoice:
  template_filename: 'norfone_invoice_template.html'
```

Variables de Modèle Disponibles

Les modèles de factures ont accès aux variables Jinja2 suivantes :

Informations sur la Facture :

- `{{ invoice_number }}` - ID unique de la facture (par exemple, `INV-2025-001234`)
- `{{ date }}` - Date d'émission de la facture (format ISO : `2025-01-10T12:00:00`)
- `{{ due_date }}` - Date d'échéance du 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

Lignes d'Articles de Transaction :

Bouclez 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 norfone_invoice_template.html
your_company_invoice_template.html
```

Étape 2 : Personnaliser HTML/CSS

Modifiez `your_company_invoice_template.html` pour correspondre à votre marque :

Domaines 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/Tax ID : 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. **Termes et Conditions**

```
<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 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` :

```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 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 Multi-Langue :

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 automatiquement calculée en additionnant le `tax_amount` de toutes les transactions dans la facture. La taxe de chaque 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 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 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. **Précision des Données :**
   - Détails du client corrects
   - Montants des transactions s'additionnent 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 s'affichent
   - Longs listes de transactions paginées correctement
5. **Livraison par E-mail :**
   - PDF attaché 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
- **Solution :** Vérifiez l'orthographe exactement (sensible à la casse), vérifiez que les données existent dans la base de données

Le style PDF est cassé :

- **Cause :** Propriété CSS non prise en charge
- **Solution :** Utilisez des propriétés CSS 2.1, testez avec CSS compatible WeasyPrint

Les images ne s'affichent pas :

- **Cause :** URLs relatives ou ressources externes bloquées
- **Solution :** Utilisez des URLs absolues HTTPS, assurez-vous que les images sont accessibles publiquement

Les tableaux débordent de la page :

- **Cause :** Largeurs de colonnes fixes trop larges
- **Solution :** Utilisez des largeurs en pourcentage, `table-layout: fixed`

Les polices ne se rendent 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 plantage de WeasyPrint
- **Solution :** 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 la 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/retirées, 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 (à moins que la facture ne soit 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 entrées de cache ou 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 dans les 30 jours)
- Limites de stockage (implémenter 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
- Hit/miss de 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. Naviguez vers **Clients → [Sélectionner le Client]**
2. Cliquez sur l'onglet **Facturation**
3. Consultez la liste des factures dans la troisième carte

Vue Système :

1. Naviguez vers **Facturation → Factures** (dans le menu principal)
2. Consultez toutes les factures de tous les clients

Widgets de Statistiques des 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 : Affichage avec suffixe "k" (par exemple, $1.5k)
- Valeurs supérieures à 1 000 000 : Affichage avec suffixe "M" (par exemple, $2.3M)
- Valeurs supérieures à 1 000 000 000 : Affichage avec suffixe "B" (par exemple, $1.1B)

Indicateurs de Tendance :

- Les 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 unique de la facture
- **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)
- ✉ **(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 le client (uniquement affiché en vue système, pré-rempli en 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
- **Sélectionner Tout** - Inclure toutes les transactions affichées

- **Effacer Tout** - Exclure toutes les transactions
- Les transactions exclues apparaissent grises avec un texte barré
- Totaux en temps réel mis 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 (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 : Fixez 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 non désirées (par exemple, excluez les transactions non mobiles pour créer une facture uniquement mobile).

Facture Unique : Fixez la date de début et de fin au même jour, l'aperçu montre uniquement les transactions de cette date. Excluez tous 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 tous les détails de la facture, y compris toutes les transactions, les totaux et les actions disponibles.

Modal de 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 E-mail** - E-mail 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 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 indicateur de chargement apparaît pendant la génération (uniquement la première fois)
3. Le navigateur vous invite à enregistrer le fichier : `Invoice_01234.pdf`
4. Le PDF s'ouvre ou est enregistré 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 effacé si la facture est modifiée ou si le modèle est mis à jour

Dépannage des Problèmes de Téléchargement :

- **L'indicateur de chargement ne s'arrête jamais** - Vérifiez la console du navigateur, l'API peut être hors ligne

- **PDF vide ou corrompu** - Vérifiez le modèle de facture pour des erreurs de syntaxe
- **Le téléchargement échoue** - Vérifiez les paramètres de 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 affichant 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 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** - 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 "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 (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 des factures.

E-mail des Factures

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 <integrations_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 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'E-mail :

- **E-mail non envoyé** - Vérifiez les identifiants API Mailjet dans `crm_config.yaml`
- **Client ne reçoit pas** - Vérifiez les adresses e-mail de contact du client
- **PDF non attaché** - Vérifiez que la génération du PDF a réussi (essayez de télécharger d'abord)

Annulation 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 pour 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, 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 avoir été 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 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 par espèces, POS ou virement bancaire (inversion manuelle requise)
- Impossible de rembourser partiellement (montant total de la facture uniquement)
- Impossible de rembourser deux fois

Recherche et Filtrage des 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 (uniquement en vue système)

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 de Factures

Flux de Travail 1 : Facturation Mensuelle avec Aperçu des Transactions

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 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. Vérifiez 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 la 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 : Jan 1-31
    - Date d'Échéance : Feb 15

- 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 : Jan 1-31 (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"
   - Seules les frais divers non facturés apparaissent dans l'aperçu
   - Cliquez sur **"Sélectionner Tout"** pour inclure tout
   - Cliquez sur **"Générer la Facture"**
6. Envoyez les trois factures au client

Flux de Travail 3 : Exclure 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 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'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. Vérifiez 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 : Révision Rapide de la Facture et Ajustement

1. Le personnel génère la facture mensuelle
2. L'aperçu des transactions montre un total élevé inattendu
3. Vérifiez 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 avec confiance

Flux de Travail 5 : Facture d'Installation Unique

1. Le technicien de 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 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 tous les frais récurrents en utilisant le bouton **X** (si présent)
9. Cliquez sur **"Générer la Facture"**
10. Envoyez au client immédiatement
11. Le client paie en ligne via Stripe

Flux de Travail 6 : Révision 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 le modal des Détails de la Facture
4. Vérifiez les informations de la facture :
   - ID de la facture, dates, statut
   - Toutes les transactions incluses avec descriptions
   - Répartition 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 de renvoyer l'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 la facture pour voir les détails
3. Vérifie la liste des transactions dans le 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 page du 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

Flux de Travail 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. 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 page du modal
7. Sélectionnez le mode 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. 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 page du 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

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 peuplés, consultez les journaux de l'API.

Échec de paiement avec 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 une transaction de crédit manuellement.

L'e-mail de la facture n'est pas envoyé

- **Cause :** Identifiants API Mailjet invalides, le client n'a pas de contact de facturation ou modèle d'e-mail manquant
- **Solution :** Vérifiez la configuration Mailjet dans `crm_config.yaml`, vérifiez les contacts du client, vérifiez que le modèle d'e-mail de facture existe.

Le bouton de remboursement n'apparaît pas

- **Cause :** Facture payée par espèces/POS/virement 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 :** Videz 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 :** Aucune méthode de paiement enregistrée dans le portail Self-Care
- **Solution :** Le client doit ajouter une carte de crédit à la page **Méthodes de Paiement** avant de payer des 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 dupliquée. 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 facture

- **Cause :** Transaction déjà facturée ou problème de navigateur
- **Solution :** Vérifiez que la transaction s'affiche dans l'aperçu avec une coche. Actualisez 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
- **Solution :** Vérifiez soigneusement l'aperçu des transactions. Vérifiez le coût de détail et la taxe de chaque 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
- **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.

Le modal des Détails de la Facture ne s'ouvre pas

- **Cause :** Erreur JavaScript ou page non complètement chargée
- **Solution :** Actualisez 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 0 % de taxe 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 le 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 par e-mail des factures et modèles
- `administration_configuration` - Configuration du modèle de facture
- `payments_transaction` - Création de transactions qui apparaissent sur les factures
- `payments_process` - Traitement des paiements des 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 <payment_system_guide>`, `Vue d'Ensemble de la Facturation <billing_overview>`, `Traitement des Paiements <payments_process>`, `Factures <payments_invoices>`.

Vue d'Ensemble

Le système de méthodes de paiement fournit :

- **Stockage Sécurisé des Cartes** - Cartes tokenisées par des 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 Self-Care <self_care_portal>`
- **Gestion par le Personnel** - Le personnel de support peut ajouter/retirer des cartes au nom des clients

Méthodes de Paiement Prises en Charge :

- 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 des fournisseurs de paiement et stockés en toute sécurité. OmniCRM ne stocke que :

- Fournisseur de paiement (stripe, paypal)
- Marque de carte (Visa, Mastercard, etc.)
- 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 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 :

Voir tous les clients avec des cartes expirantes :

Cela montre une liste au niveau système des cartes expirant dans les 60 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 carte et 4 derniers chiffres
- **Expiration** - Mois/année d'expiration (format MM/YYYY)
- **Ajouté** - Date à laquelle

Vue d'ensemble 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 : `Payment Methods <payment_methods>`, `Transactions <payments_transaction>`, `Invoices <payments_invoices>`, `Payment Processing <payments_process>`, `Wallet vs customer_transaction ledger <billing_wallet_ledger>`.

Accéder à l'onglet Facturation

Vue par client :

1. Naviguez vers **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 de libre-service :

Les clients accédant au `Self-Care Portal <self_care_portal>` 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, Impayé, 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 **Factures Totales** augmente
- Le total des **Factures Impayées** augmente

Relation Facture → Paiement

1. Paiement de 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 de Ce Mois** 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 complets de facturation

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 :**
   - Naviguez vers le 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 du 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 la 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 de libre-service
   - 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 :**
   - Naviguez vers la section **Transactions**
   - Filtrer : **Statut de 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 :**
   - Naviguez vers **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 la 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 le portail de libre-service
    - Le personnel traite les paiements en espèces/POS pour les clients qui paient en personne
    - Le total des **Factures Impayées** diminue au fur et 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 :**
    - Naviguez vers le 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 Jan"
- 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 une nouvelle facture est générée :
     - 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, crédit 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 navigue vers 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
    - L'ancienne carte est automatiquement supprimée (si désiré)
4. **Réessayer le paiement :**
    - Naviguez vers 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 le nouvel ID d'intention de paiement Stripe

Résultat : Client a mis à jour sa méthode de paiement, facture payée avec succès avec la nouvelle carte.

Flux de travail 5 : Annulation d'une facture incorrecte et nouvelle facturation

Objectif : Facture générée par le personnel avec des transactions incorrectes - annuler et régénérer correctement

1. **Erreur découverte :**
    - Facture #INV-2025-001234 générée avec une plage de dates incorrecte
    - Comprenait des transactions du mauvais mois
    - La facture est impayée

2. **Annuler la facture :**
    - Naviguez vers l'onglet **Facturation** → section **Factures**
    - Localisez la facture incorrecte
    - Cliquez sur l'icône de suppression (🗑)
    - Confirmer l'annulation
    - Facture annulée
3. **Transactions libérées :**
    - Naviguez vers 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 la 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 re-facturées 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 des factures impayées

    - Naviguez vers le 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 de 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 :**
    - Naviguez vers la section **Transactions**
    - Trois nouvelles transactions de paiement créées :
    - Toutes liées aux factures respectives
6. **Mettre à jour les 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 impayées en 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 la 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 si nécessaire
- Envoyer les factures par e-mail immédiatement après leur génération
- Vérifier les PDF des 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 :

- Examiner 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 de libre-service)

Méthodes de paiement :

- Garder au moins une méthode de paiement valide dans 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 des 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 service 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. Naviguer vers le client → **Onglet Facturation** → section **Méthodes de paiement**
2. Vérifier si le client a des méthodes de paiement enregistrées
3. Vérifier 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 des transactions incorrectes

Symptômes :

- La facture comprend des transactions de la mauvaise période
- La facture manque des transactions attendues
- Le total de la facture est incorrect

Diagnostic :

1. Ouvrir la facture dans la section **Factures**
2. Vérifier les transactions incluses dans la facture
3. Vérifier les dates des transactions par rapport à la plage de dates de la facture
4. Vérifier 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 la 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
- Les factures mensuelles ont déjà été générées

Diagnostic :

1. Filtrer les transactions par **Statut de Facture : Non facturé**

2. Vérifier la liste des transactions non facturées
3. Vérifier les dates des transactions - peuvent être des frais récents après la dernière génération de facture
4. Vérifier 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 d'être refusée

Symptômes :

- Le paiement Stripe échoue avec "Carte refusée"
- Le client signale que la carte devrait fonctionner
- Plusieurs tentatives de réessai échouent

Diagnostic :

1. Vérifier le tableau de bord Stripe pour la raison du refus
2. Raisons de refus courantes :
    - Fonds insuffisants
    - Carte expirée
    - Carte signalée comme perdue/volée
    - Prévention de fraude bancaire
    - Carte internationale bloquée

Solution :

- Demander 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
- Vérifier que l'adresse de facturation correspond à l'adresse de facturation de la carte

Raccourcis clavier et astuces

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 impayées nécessitant un suivi

Opérations en masse :

- Sélectionnez plusieurs factures (case à cocher) pour supprimer/annuler plusieurs à la fois (réservé au personnel)
- 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 la 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` - Visualisation de l'historique d'activité de facturation
- Surveillance & Métriques - Métriques et surveillance du système de facturation OCS/CGRateS

Actions et Comportements de Recharge de CGRateS

Ce guide explique comment fonctionnent les Actions de 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 Facturation en Ligne (CGRateS) d'OmniCRM, **Actions** est 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 manuellement des soldes aux comptes 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 un temps d'expiration et un poids

ID de Solde - Un identifiant unique pour un type de solde (par exemple, "Forfait de Données", "Minutes de Voix")

Poids - Priorité pour la consommation du solde (le poids le plus élevé est consommé en premier)

Temps d'Expiration - Quand le solde expire (date absolue ou relative comme "+5 jours")

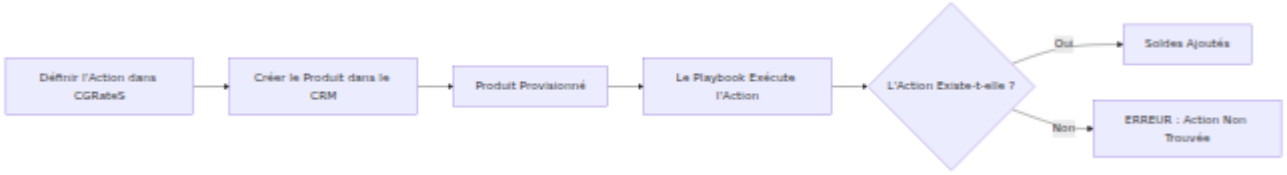
Bloqueur - Un drapeau de solde spécial qui bloque toute utilisation lorsque le solde atteint zéro, même si d'autres soldes existent (voir Bloqueurs de Solde)

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 un prérequis critique qui est souvent négligé.

Quand les Actions Sont Définies

Les Actions sont généralement définies lors de la **configuration initiale du système** ou de **la configuration du produit**, 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

```python
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 son 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 façon 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 un ID correspondant
- Ajoute la nouvelle valeur à la valeur existante
- Met à jour l'expiration au nouveau temps d'expiration
- Préserve le solde existant (report)

Exemple :

```
# État initial :
Solde :
  ID: "Data_Package__5368709120"
  Valeur: 1073741824  # 1GB restant
  Expiration: 2024-12-24 (1 jour restant)

# Exécuter l'action *topup avec :
Action :
  Identifier: "*topup"
  Solde:
    ID: "Data_Package__5368709120"  # Même ID - déclenche le
report
    Valeur: 5368709120  # 5GB
    ExpiryTime: "+5d"

# Résultat après *topup :
Solde :
  ID: "Data_Package__5368709120"
  Valeur: 6442450944  # 6GB (1GB + 5GB reportés)
  Valeur_Originale: 5368709120  # Affiche toujours l'original 5GB
  Valeur_hr: "6 GB"
  Valeur_Originale_hr: "5 GB"
  Restant_hr: "6 GB de 5 GB (1 GB reporté)"
  Expiration: 2024-12-29 (5 jours à partir de maintenant)
```

Cas d'Utilisation :

- Récompenses de fidélité (ajouter des données supplémentaires au forfait existant)
- Crédits de compensation (ajouter au solde du client)
- Forfaits de données reportés
- 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 un ID correspondant
- Jette l'ancienne valeur (pas de report)
- Définit le solde à la nouvelle valeur uniquement
- Met à jour l'expiration au nouveau temps d'expiration

Exemple :

```
# État initial :
Solde :
  ID: "Data_Package__5368709120"
  Valeur: 1073741824  # 1GB restant
  Expiration: 2024-12-24 (1 jour restant)

# Exécuter 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 :

- Forfaits mensuels récurrents (réinitialiser au montant total chaque mois)
- Recharges de taille fixe (recevoir toujours le 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 originale. 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 : N'incluez pas 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 des 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 le 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. **Formater l'Affichage Restant :**

```
remaining_hr = f"{current_value_hr} de {original_value_hr}"
# "50 GB de 100 GB"
```

Affichage de l'UI Web :

L'interface 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 Report** - Lors de l'utilisation de `*topup` (report), si un client a 70 GB restants et recharge 100 GB :
   - L'ID de Solde reste : `"AU_Data_Domestic__107374182400"` (original 100 GB)
   - La valeur actuelle devient : 170 GB
   - L'UI affiche : "170 GB (70 GB reportés + 100 GB nouveaux)"

Meilleure Pratique - Création d'IDs de Solde :

Incluez toujours la taille originale après `__` pour un affichage correct de l'UI. Ne dupliquez pas 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'information 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 car 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 de Mise en Œuvre :

```
- 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 à travers 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 (Report)

Utilisez le même ID de Solde avec l'action `*topup` pour permettre le report de solde et l'extension d'expiration.

Exemple de Mise en Œuvre :

```
- name: Définir l'ID de solde fixe
  set_fact:
    balance_id: "Data_5days__5368709120"

- name: Ajouter un solde avec *topup (report)
  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 suivants 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 à nouveau le même
addon :
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 reportés + 5 GB nouveaux)"
  PourcentageUtilisé: -80  # Négatif indique un report
  Expiration: 2024-12-27 (nouvelle +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 de Mise en Œuvre :

```
- name: Définir l'ID de solde fixe
  set_fact:
    balance_id: "Monthly_Plan__32212254720"

- name: Ajouter un solde avec *topup_reset (pas de report)
  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 : Le Plan d'Action s'exécute *topup_reset
Solde :
  ID: "Monthly_Plan__32212254720"
  Valeur: 32212254720  # Réinitialisé au total, les 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 limiter l'utilisation** même lorsque des soldes existent. Un solde bloqueur arrête la consommation lorsque son montant 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 soldes : `*voice`, `*data`, `*sms`, `*monetary`

Cas d'Utilisation pour les Bloqueurs

1. Suspension de Compte

Bloquer toute utilisation lorsque le 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
- vérifiée 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 solde disponible.

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": [
            # 10GB 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 $ (bloqueur)
            {
                "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,  # Arrêter lorsque 50 $ dépensés
                "Weight": 90
            }
        ]
    }]
}
```

Flux :

1. Le client utilise 10GB inclus → GRATUIT (à partir de Included_Data_10GB)
2. Le client utilise 5GB supplémentaires → Déduit de Overage_Cap (tarifs PAYG)

3. Lorsque Overage_Cap atteint 0 $ → **Toute utilisation bloquée** (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 (bloqueur - 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 aux Destinations

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 zéro
                "BalanceWeight": 2000,  # Poids plus élevé -
vérifié en premier pour premium
                "Blocker": True,
                "Weight": 10
            }
        ]
    }]
}
```

Flux :

- Appels domestiques → Utilise le solde Regular_Usage
- Appels vers des numéros premium → Correspond à Premium_Blocker (poids 2000 > 1000) → **BLOQUÉ**

Bloqueur vs Soldes Désactivés

Ne confondez pas `Blocker` avec `Disabled` :

Fonctionnalité	Bloqueur	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

```
# Bloqueur : Autorise 10GB, puis bloque tout
{
    "BalanceId": f"Data_Cap__{10 * 1024 * 1024 * 1024}",
    "Units": 10 * 1024 * 1024 * 1024,
    "Blocker": True  # Arrête l'utilisation à 10GB
}

# 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

Combinez des soldes unitaires, un débordement monétaire et un plafond bloqueur :

```
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 domestiques 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 limite de dépassement
            {
                "Identifier": "*topup_reset",
                "BalanceId": "Overage_Allowance",
                "BalanceType": "*monetary",
                "DestinationIDs": "*any",
                "Units": 2000,  # 20,00 $
                "ExpiryTime": "*month",
                "BalanceWeight": 1000,
                "Weight": 90
            },
            # 50 $ PLAFOND DUR (bloqueur)
            {
                "Identifier": "*topup_reset",
                "BalanceId": "Hard_Spending_Cap",
                "BalanceType": "*monetary",
                "DestinationIDs": "*any",
                "Units": 5000,  # 50,00 $ maximum absolu
                "ExpiryTime": "*month",
```

```
                "BalanceWeight": 500,  # Plus bas que le
dépassement - utilisé en dernier
                "Blocker": True,  # STOP 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. **À 700-1200 minutes :** Utilise Hard_Spending_Cap à 0,10 $/min = 50 $ (500 min)
4. **À 1200 minutes :** Hard_Spending_Cap épuisé → **TOUTE UTILISATION BLOQUÉE**

Le client obtient 1200 minutes au total, avec un plafond de dépenses maximum de 50 $.

Meilleures Pratiques avec les Bloqueurs

1. **Utilisez un Poids Élevé pour les Bloqueurs**

   ```
   "BalanceWeight": 9999  # Assurez-vous que le bloqueur est
   vérifié en premier
   ```

2. **Bloqueurs à Valeur Zéro pour un Blocage Immédiat**

   ```
   "Units": 0,  # Bloquer immédiatement
   "Blocker": True
   ```

3. **Notifier les Clients Avant l'Épuisement du Bloqueur**

- Utilisez ActionTriggers pour envoyer des notifications à 80 %, 90 %, 100 % de l'utilisation du bloqueur
- Donnez aux clients l'option d'augmenter le plafond avant que le blocage ne se produise

4. **Retirer les Bloqueurs Lors de la Réouverture**

```
# Utilisez *remove_balance pour supprimer le bloqueur
{
    "Identifier": "*remove_balance",
    "BalanceId": "Suspension_Blocker"
}
```

5. **Tester le Comportement des Bloqueurs**

   - Vérifiez que le bloqueur retourne l'erreur `INSUFFICIENT_CREDIT_BALANCE_BLOCKER`
   - Confirmez que les CDR montrent coût = -1.0 lorsqu'il est bloqué
   - Testez que d'autres soldes NE SONT PAS utilisés après l'épuisement du bloqueur

Dépannage des Bloqueurs

Problème : Utilisation non bloquée malgré le bloqueur à zéro

Causes Possibles :

1. Poids du bloqueur trop bas (d'autres soldes vérifiés en premier)
2. DestinationIDs ne correspondent pas à la destination d'utilisation
3. Champ Bloqueur non défini sur `True`

Solution :

```
# Vérifiez la configuration du bloqueur
OCS_Obj.SendData({
    'method': 'ApierV2.GetAccount',
    'params': [{"Tenant": tenant, "Account": "service_uuid"}]
})

# Vérifiez :
# - Blocker : true
# - 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 bloqueur créé involontairement avec une faible valeur

Solution : V��rifiez tous les soldes pour `Blocker: true` et vérifiez 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 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

# Le Solde 1 consommé en premier (précision 4 > précision 2)
# La 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: 5GB
  Poids: 20

Solde 2:
  ID: "Standard_Data"
  Valeur: 10GB
  Poids: 10

# Le 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 : Plus Ancien en Premier

Lorsque le poids correspond, le plus ancien solde est utilisé en premier.

```
Solde 1:
  ID: "Data_Package_A"
  Valeur: 5GB
  Expiration: 2024-12-25
  Poids: 10

Solde 2:
  ID: "Data_Package_B"
  Valeur: 10GB
  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: 5GB
  Expiration: 2024-12-25
  Poids: 11  # Poids plus élevé = consommé en premier

Solde 2:
  ID: "Data_Package_B"
  Valeur: 10GB
  Expiration: 2025-01-15
  Poids: 10  # Poids plus bas = consommé en second

# Le Solde 1 consommé en premier (poids 11 > poids 10)
```

Meilleure Pratique : Si vous souhaitez garantir que les soldes bientôt expirés soient consommés en premier, assignez-leur des poids plus élevés lors de leur

création.

Exemples Pratiques

Exemple 1 : Simple Addon de Données (Indépendant)

Scénario : Le client peut acheter plusieurs fois l'addon "5GB 5 jours", chacun créant un solde séparé.

Mise en Œuvre :

```
- name: Générer un UUID pour un solde unique
  set_fact:
    uuid: "{{ 99999999 | random | to_uuid }}"

- name: Ajouter un solde indépendant de 5GB
  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 l'addon le 24 décembre → Obtient 5GB expirant le 29 décembre
- Achète l'addon le 25 décembre → Obtient 5GB expirant le 30 décembre
- Les deux soldes coexistent ; l'ordre de consommation dépend des poids de solde (les deux ont un poids de 10, donc l'ordre n'est pas garanti)

Exemple 2 : Forfait de Données avec Report

Scénario : "Plan Mensuel 50GB" où les données non utilisées sont reportées lorsque le client recharge tôt.

Mise en Œuvre :

```
- name: Ajouter un solde de données reporté
  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,  # 50GB
            "ExpiryTime": "+720h",  # 30 jours
            "Weight": 10
          }
        }]
      }
```

Type d'Action : Utilise le comportement par défaut `*topup` (report activé)

Expérience Client :

- Jour 1 : Obtient 50GB expirant dans 30 jours
- Jour 20 : Utilise 30GB, a 20GB restants

- Jour 20 : Recharge à nouveau → Obtient 70GB au total (20GB + 50GB), l'expiration est prolongée à +30 jours à partir du Jour 20

Exemple 3 : Plan Mensuel Fixe (Pas de Report)

Scénario : Plan "Illimité 100GB Mensuel" qui se réinitialise exactement à 100GB chaque mois, sans report.

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,  # 100GB
            "ExpiryTime": "*monthly",
            "BalanceId": "Monthly_Plan__107374182400",
            "Weight": 10
          }]
        }]
      }

- 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_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 100GB, utilise 95GB, a 5GB restants
- Mois 2 : Le solde se réinitialise à 100GB (5GB de données non utilisées perdus)
- Mois 2 : Utilise 20GB, a 80GB restants
- Mois 3 : Le solde se réinitialise à 100GB (80GB de données non utilisées perdus)

Exemple 4 : Soldes Multi-Niveaux avec Priorité de Poids

Scénario : Le client a "Données Bonus" (haute priorité) et "Données Standard" (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,  # 5GB
            "ExpiryTime": "+240h",  # 10 jours
            "Weight": 20  # Priorité plus élevée
          }
        }]
      }

# Ajouter des données standard avec un poids normal
- name: Ajouter un solde de données standard
  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": "Standard_Data",
            "Value": 53687091200,  # 50GB
            "ExpiryTime": "+720h",  # 30 jours
            "Weight": 10  # Priorité normale
          }
        }]
      }
```

Expérience Client :

- A 5GB de données bonus (poids 20) + 50GB de données standard (poids 10)
- Consomme d'abord toutes les données bonus
- Puis consomme à partir du pool de 50GB de données standard
- Les données standard sont préservées plus longtemps

Identifiants d'Action Courants

CGRateS prend en charge plusieurs identifiants d'action pour différentes opérations :

Manipulation de Solde

***topup** - Ajouter au solde existant (report) ***topup_reset** - Réinitialiser le solde à une nouvelle valeur (pas de report) ***debit** - Soustraire du solde ***debit_reset** - Définir le solde à une valeur négative ***reset_account** - Supprimer tous les soldes

Conception de Produits Additionnels

Lors de la conception de produits additionnels dans OmniCRM, considérez ces questions :

Question 1 : Les soldes doivent-ils s'accumuler ?

Oui (Indépendant) → Utilisez des IDs de Solde uniques (avec UUID) **Non (Remplacer)** → Utilisez un ID de Solde fixe avec `*topup_reset` **Oui (Report)** → Utilisez un ID de Solde fixe avec `*topup`

Question 2 : Que se passe-t-il avec le solde non utilisé ?

Report → Utilisez l'action `*topup` **Perdu** → Utilisez l'action `*topup_reset` **Pools séparés** → Utilisez des IDs de Solde uniques

Question 3 : Comment les soldes doivent-ils être consommés ?

Le plus ancien en premier → Utilisez des poids différents (attribuez un poids plus élevé aux soldes plus anciens/bientôt expirés) **Premium d'abord** → Poids différents (poids plus élevé = priorité plus élevée) **Ordre spécifique** → Utilisez des poids : 30 (premium), 20 (bonus), 10 (régulier) **Aléatoire/Pas de préférence** → Utilisez le même poids (ordre de consommation non garanti)

Question 4 : Quelle est la stratégie d'expiration ?

Durée fixe → Utilisez une expiration relative (`+720h` pour 30 jours) **Fin de mois** → Utilisez `*monthly` dans le Plan d'Action **N'expire jamais** → Utilisez `*unlimited` ou une durée très longue

Structure des Actions 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": "Nom_Action_Ici",
          "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": "Nom_Solde",  # 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
en tant que CDR
              "BalanceType": "*generic",
              "ExtraParameters": "
{\"Category\":\"^activation\",\"Destination\":\"Nom de l'Addon\"}"
            }
          ]
        }]
      }
```

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 (compte)
- `*monetary` - Soldes monétaires (unités monétaires)
- `*generic` - Soldes génériques

Units - Montant à ajouter/déduire (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

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": "Nom_Action_Ici"
        }]
      }
```

Cela exécute toutes les opérations définies dans l'Action sur le compte spécifié.

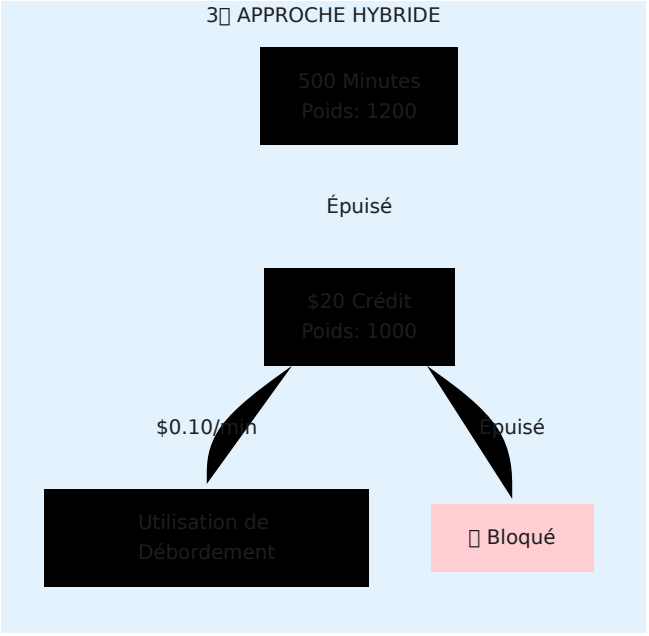
Approches de Gestion des Soldes

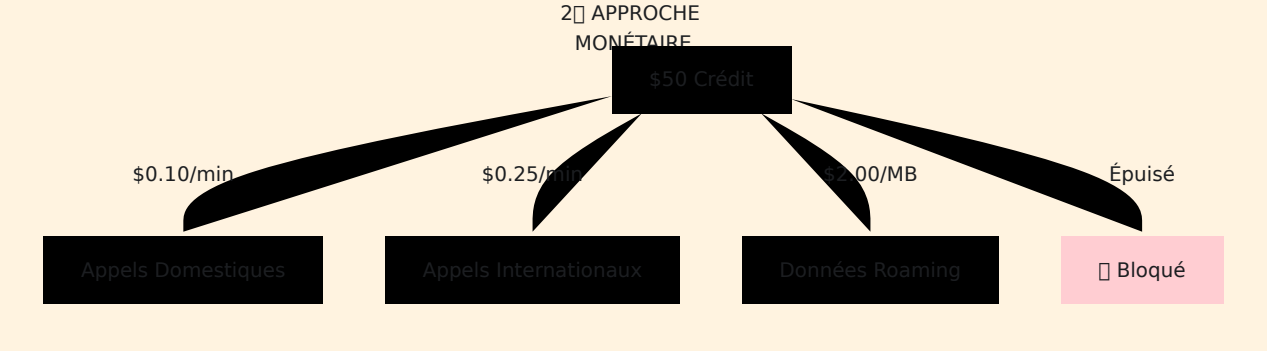
Il existe trois approches principales pour gérer les soldes dans CGRateS, chacune ayant des compromis différents pour l'expérience client, la prévisibilité de la facturation et la complexité opérationnelle.

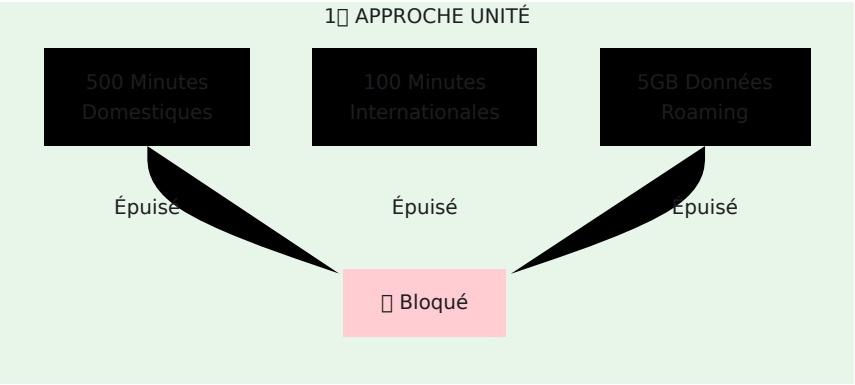
Comparaison : Unitaire vs Monétaire vs Hybride

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 Facture Surprise	□ Faible (plafonds durs)	□ Élevé (facturation illimitée)	⚠ Modéré (plafond de débordement)
Complexité de Configuration	⚠ Modérée (beaucoup de soldes)	□ Simple (un solde)	□ Complexe (les deux)
Optimisation des Revenus	⚠ ARPU plus bas	□ ARPU plus élevé pour les utilisateurs lourds	□ ARPU équilibré
Satisfaction Client	□ Élevée (pas de surprises)	□ Faible (facture surprise)	□ 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, compte 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 Domestique 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 domestiques
            {
                "Identifier": "*topup_reset",
                "BalanceId": f"Domestic_Voice__{500 * 60 *
1000000000}",
                "BalanceType": "*voice",
                "DestinationIDs": "Dest_Domestic_All",  #
UNIQUEMENT domestique
                "Units": 500 * 60 * 1000000000,  # 500 minutes en
nanosecondes
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 90
            },
            # 1000 SMS domestiques
            {
                "Identifier": "*topup_reset",
                "BalanceId": "Domestic_SMS__1000",
                "BalanceType": "*sms",
                "DestinationIDs": "Dest_Domestic_All",
                "Units": 1000,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 89
            },
            # 10GB de données domestiques
            {
                "Identifier": "*topup_reset",
                "BalanceId": f"Domestic_Data__{10 * 1024 * 1024 *
1024}",
                "BalanceType": "*data",
```

```
                "DestinationIDs": "Dest_PLMN_OnNet",  # Votre PLMN
sur le réseau
                "Units": 10 * 1024 * 1024 * 1024,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 88
            },
            {
                "Identifier": "*cdrlog",
                "BalanceType": "*generic",
                "ExtraParameters": "
{\"Category\":\"^activation\",\"Destination\":\"Domestic Plan\"}",
                "Weight": 80
            }
        ]
    }]
}

result = OCS_Obj.SendData(Action_Domestic_Plan)
assert result['error'] is None or result['error'] == "EXISTS"
```

Comment Cela Fonctionne :

- Appels domestiques (1-555-1234) → Utilise 500 minutes
- Appels internationaux (44-20-xxx) → AUCUN solde disponible, bloqué OU utilise le solde monétaire si disponible
- SMS domestiques → Utilise 1000 SMS
- Utilisation de données à domicile → Utilise 10GB
- Données en roaming → AUCUN solde disponible (nécessiterait un solde de données en roaming séparé)

Avantages :

- Coûts prévisibles pour les clients
- Pas de surprise sur la 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 à la destination. 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 Tarification pour chaque destination afin de définir le montant en dollars par unité. Voir Profils de Tarification 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\":\"$50 PAYG
Credit\"}",
                "Weight": 80
            }
        ]
    }]
}

result = OCS_Obj.SendData(Action_PAYG_Credit)
```

Comment PAYG Fonctionne :

Scénario 1 : Appel domestique, 10 minutes

- Tarif : 0,10 $/min
- Charge : 10 × 0,10 $ = 1,00 $

- Restant : 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 $
- Restant : 47,70 $

Scénario 3 : Roaming sur Verizon, 100 Mo de données

- Tarif : 2,00 $/Mo
- Charge : 100 × 2,00 $ = 200,00 $
- Résultat : Fonds insuffisants → Session bloquée à ~47 $ (environ 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 surprise sur la facture (surtout en roaming)
- Peut être coûteux pour les utilisateurs lourds

Approche 3 : Hybride (Le Meilleur des Deux)

Concept : Combiner des soldes unitaires avec un secours monétaire. Utiliser des soldes unitaires de haute priorité pour l'utilisation incluse, avec un solde monétaire de faible priorité comme débordement. Meilleure expérience client avec un coût de base prévisible et des dépassements flexibles.

Exemple : Plan Flexible Hybride

```
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 domestiques
            {
                "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 domestiques
                "Weight": 95
            },
            # 100 minutes de voix internationales
            {
                "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
l'international
                "Weight": 94
            },
            # 1000 SMS domestiques
            {
```

```
                "Identifier": "*topup_reset",
                "BalanceId": "Domestic_SMS__1000",
                "BalanceType": "*sms",
                "DestinationIDs": "Dest_Domestic_All",
                "Units": 1000,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 93
            },
            # 15GB de données domestiques
            {
                "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
            },
            # 2GB de données en roaming (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
            },

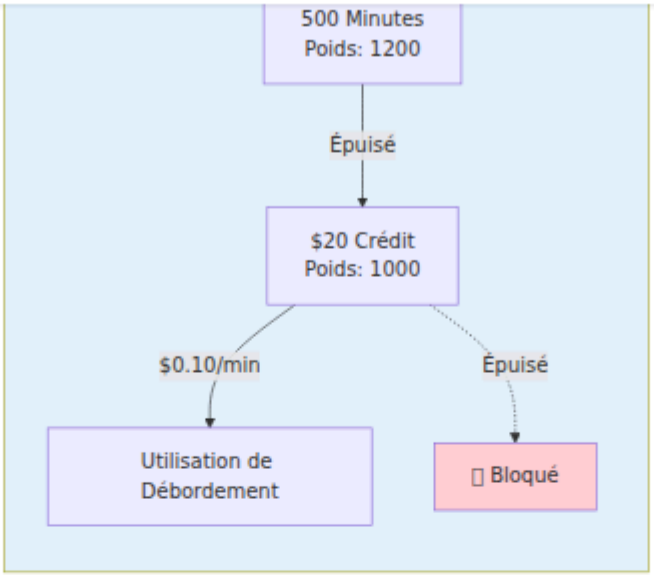
            # ============= SOLDE MONÉTAIRE (Débordement/PAYG)
=============
            # 20 $ pour les dépassements
            {
                "Identifier": "*topup",
                "BalanceId": "PAYG_Overflow_Balance",
                "BalanceType": "*monetary",
                "DestinationIDs": "*any",
                "Units": 2000,  # 20,00 $
                "ExpiryTime": "*month",
```

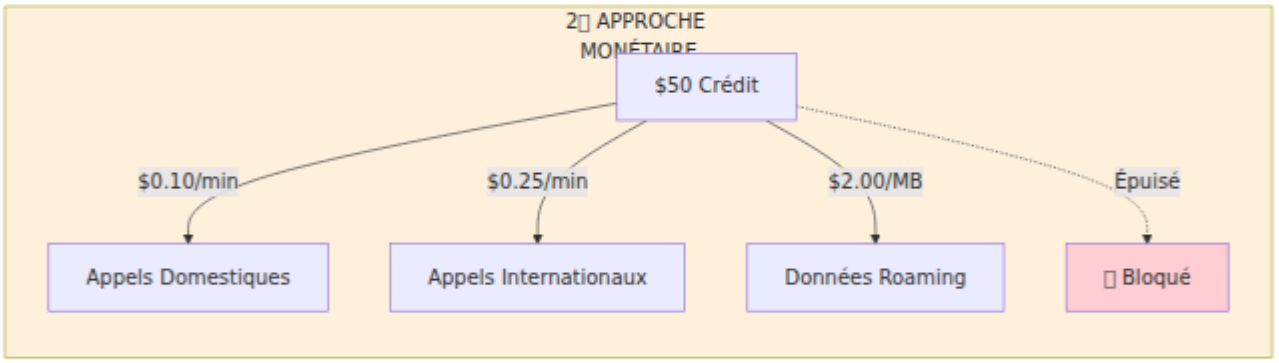
```
                "BalanceWeight": 1000,  # Consommé EN DERNIER
(secours)
                "Weight": 90
            },

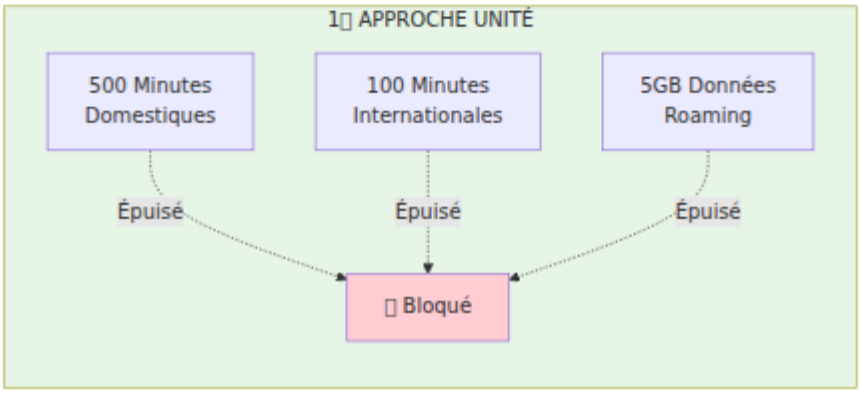
            {
                "Identifier": "*cdrlog",
                "BalanceType": "*generic",
                "ExtraParameters": "
{\"Category\":\"^activation\",\"Destination\":\"Hybrid Flex
Plan\"}",
                "Weight": 80
            }
        ]
    }]
}

result = OCS_Obj.SendData(Action_Hybrid_Plan)
```

Flux de Consommation de Solde :







Exemples de Scénarios :

Scénario 1 : 600 minutes utilisées

1. Les 500 premières minutes → Utilise "Domestic_Voice_500min" (poids 1200) - GRATUIT
2. Les 100 prochaines minutes → "Domestic_Voice_500min" épuisé, se tourne vers "PAYG_Overflow_Balance" (poids 1000)

3. Facturé : 100 min × 0,10 $ = 10,00 $ du solde monétaire

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).

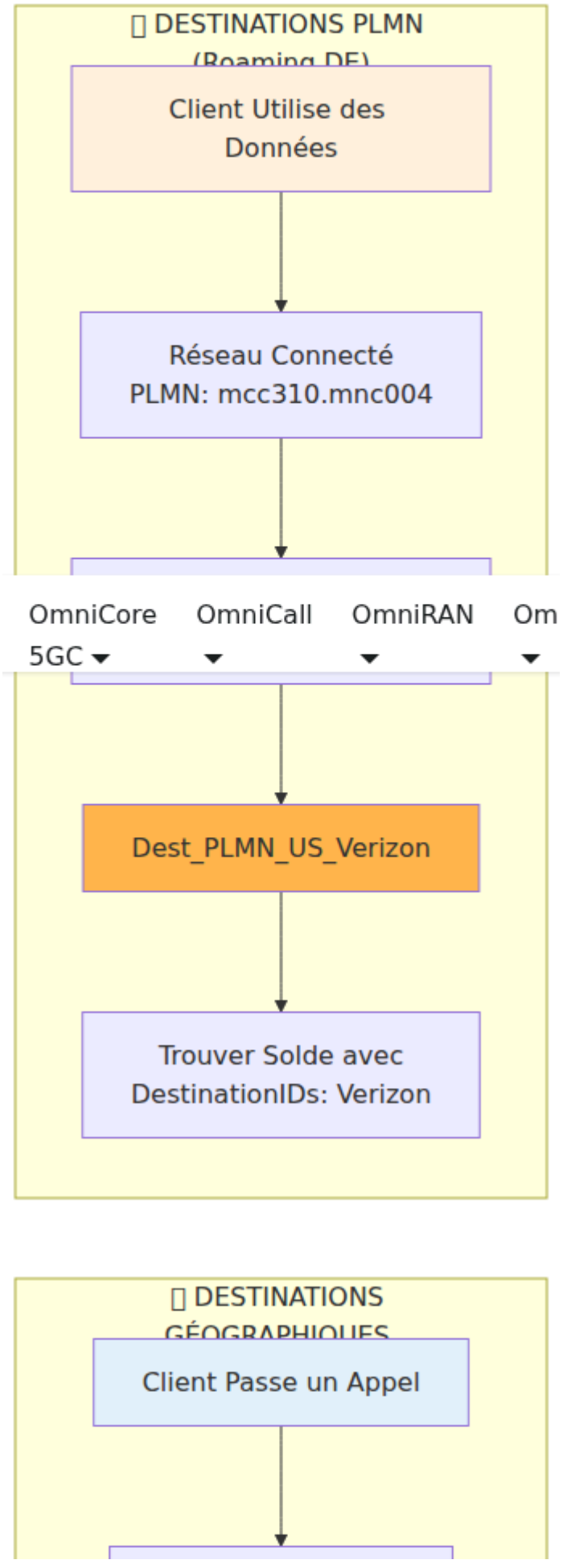
Aperçu

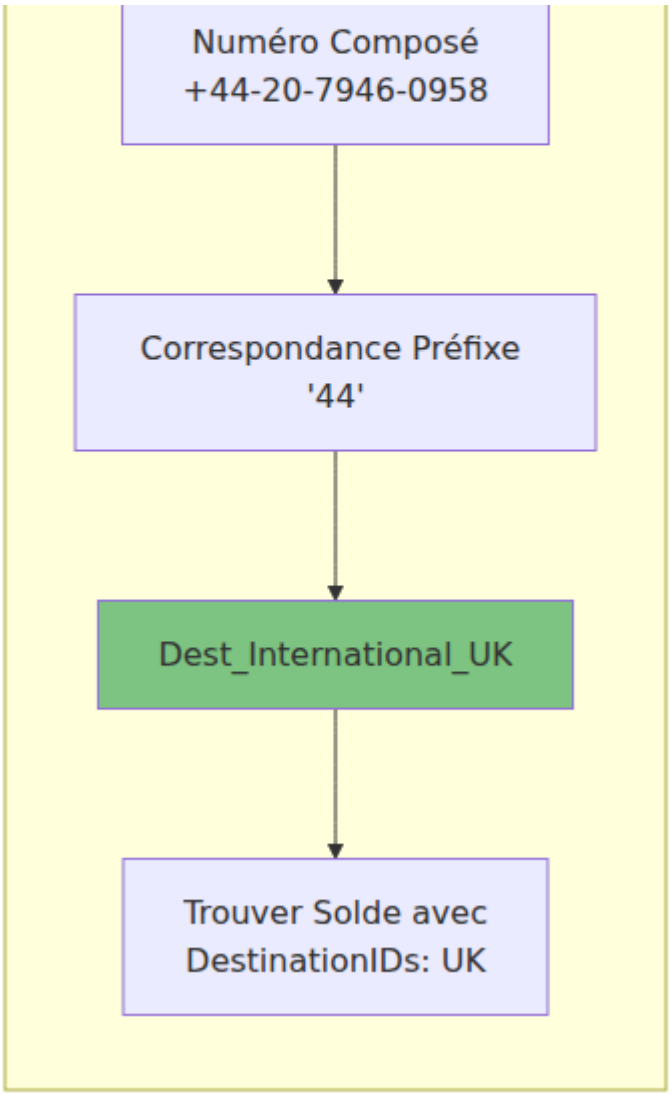
CGRateS utilise des **destinations** pour contrôler où les clients peuvent utiliser leurs soldes :

- **Destinations Géographiques** - Préfixes de numéros pour les appels/SMS VERS des emplacements spécifiques (par exemple, numéros UK, numéro gratuit US)
- **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 étant appelé VERS)
- **Soldes de données** → Utiliser des destinations PLMN (le réseau sur lequel 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 des préfixes de numéros pour les appels vocaux et les SMS vers des emplacements spécifiques.

Destinations Domestiques

```
#
# ======================================================================
# DESTINATIONS DOMESTIQUES (US/Canada - NANP)
#
# ======================================================================

# Mobile & Fixe US/Canada (partagent les indicatifs régionaux)
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_Domestic_All",
        "Prefixes": ["1"]  # NANP (US/Canada)
    }]
})

# Numéro Gratuit US
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_Domestic_TollFree",
        "Prefixes": ["1800", "1888", "1877", "1866", "1855",
"1844", "1833"]
    }]
})

# Taux Premium US
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 composés
en 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 (Adelaide, Perth, Darwin)
        ]
    }]
})

# Australie - Numéro Gratuit
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_AU_TollFree",
        "Prefixes": [
            "611800",  # Numéro gratuit
            "611300"   # Tarif local
        ]
    }]
})

# Australie - Taux Premium
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_AU_Premium",
        "Prefixes": ["6119"]  # Services à tarif premium
    }]
})

# Europe (groupées)
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 des numéros US/Canada" → `Dest_Domestic_All`
- **Addon d'appel vers le UK** : "100 minutes vers le UK" → `Dest_International_UK`
- **Plan australien** : "3000 minutes vers les mobiles et lignes fixes australiens" → `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 l'utilisation des données et les scénarios 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 dans le pays** : `"mcc310"` → N'importe quel réseau US
- **Exemples** : `mcc310.mnc004`, `mcc234.mnc015`, `mcc505.mnc057`

Sur le Réseau (Votre Réseau Domicile)

IMPORTANT : Définissez VOTRE réseau spécifique où VOS SIMs sont provisionnées.

```
# Sur le Réseau (Votre Réseau Domicile)
# C'est le PLMN de VOTRE opérateur - le réseau où les SIMs 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 US

```
#
# ======================================================================
# DESTINATIONS PLMN DE ROAMING US
#
# ======================================================================

# 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 PLMNs US (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 MCCs
(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 UK

```
#
# ============================================================================
# DESTINATIONS PLMN DE ROAMING UK
#
# ============================================================================

# 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 PLMNs UK
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",  #
USA
            "mcc302", "mcc334"  # Canada
        ]
    }]
})
# Utiliser des préfixes uniquement MCC fournit une couverture sur
N'IMPORTE quel réseau dans ces pays
# Idéal pour des 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 à un réseau** : "5 Go uniquement sur Verizon" → `Dest_PLMN_US_Verizon`
- **Roaming à l'échelle du pays** : "10 Go aux USA (tous réseaux)" → `Dest_PLMN_US_All`
- **Roaming régional** : "15 Go de roaming en Europe" → `Dest_PLMN_Zone_Europe`
- **Roaming global** : "1 Go partout" → `Dest_Roaming_All`

Quand Utiliser Chaque Approche

Type de Destination	Cas d'utilisation	Exemple	Types de Solde
Géographique	Voix/SMS vers des pays spécifiques	"100 minutes vers des numéros UK"	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 dans le pays	"5 Go aux USA (tous réseaux)"	Données (et voix/SMS en roaming)
Zone PLMN	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 l'exécution**, **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 roamer 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)
        ]
    }]
})

# Charger le plan tarifaire pour rendre les changements actifs
OCS_Obj.SendData({
    'method': 'ApierV1.LoadTariffPlanFromStorDb',
    'params': [{
        "TPid": tpid,
        "DryRun": False,
        "Validate": True
    }]
})

# Recharger le cache des destinations
OCS_Obj.SendData({
    'method': 'CacheSv1.ReloadCache',
    'params': [{
        "DestinationIDs": ["Dest_PLMN_Zone_Europe"]
    }]
})
```

Résultat : Après le chargement du 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

Retirer un pays d'une destination bloque l'accès une fois le plan tarifaire chargé :

```
# Retirer 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) RETIRÉE
        ]
    }]
})

# Charger le plan tarifaire pour rendre les changements actifs
OCS_Obj.SendData({
    'method': 'ApierV1.LoadTariffPlanFromStorDb',
    'params': [{
        "TPid": tpid,
        "DryRun": False,
        "Validate": True
    }]
})

# Recharger le cache
OCS_Obj.SendData({
    'method': 'CacheSv1.ReloadCache',
    'params': [{
        "DestinationIDs": ["Dest_PLMN_Zone_Europe"]
    }]
})
```

Résultat : Après le chargement du plan tarifaire, les clients tentant de roamer en Allemagne seront **bloqués**, même s'ils ont encore un solde restant 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 USA (et précédemment le
Canada)
            # Aucun changement nécessaire - le préfixe "1" couvre
déjà les deux
        ]
    }]
})

# Ou créer une destination distincte pour l'Amérique du Nord
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_North_America",
        "Prefixes": [
            "1"  # USA + Canada
        ]
    }]
})
```

Impact : Tous les soldes avec `DestinationIDs: "Dest_Domestic_All"` fonctionnent maintenant pour les appels vers les USA 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 roame 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 : Ajouter l'Espagne à `Dest_PLMN_Zone_Europe` OU s'assurer que le client a un solde monétaire avec `DestinationIDs: "*any"` comme 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 UK reste inutilisé (ne correspond qu'à "44")
- Pas de solution de secours monétaire pour couvrir l'appel

Correction : Ajouter un solde monétaire avec `DestinationIDs: "*any"` OU ajouter 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 UK 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 modifications de destination peuvent être testées et simulées dans des environnements non productifs avant d'être appliquées 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 : Toujours tester les mises à jour de destination 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 les référencant une fois le plan tarifaire chargé
2. **Charger le Plan Tarifaire pour activer** - Utilisez `LoadTariffPlanFromStorDb` pour rendre les changements de destination actifs dans l'exécution
3. **Aucune correspondance de destination = Aucun service** - Les clients seront bloqués 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. **Rechargement du cache requis** - Rechargez toujours le cache des destinations après le chargement du plan tarifaire
6. **Tester avant la production** - Les modifications de destination impactent tous les clients une fois chargées

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.

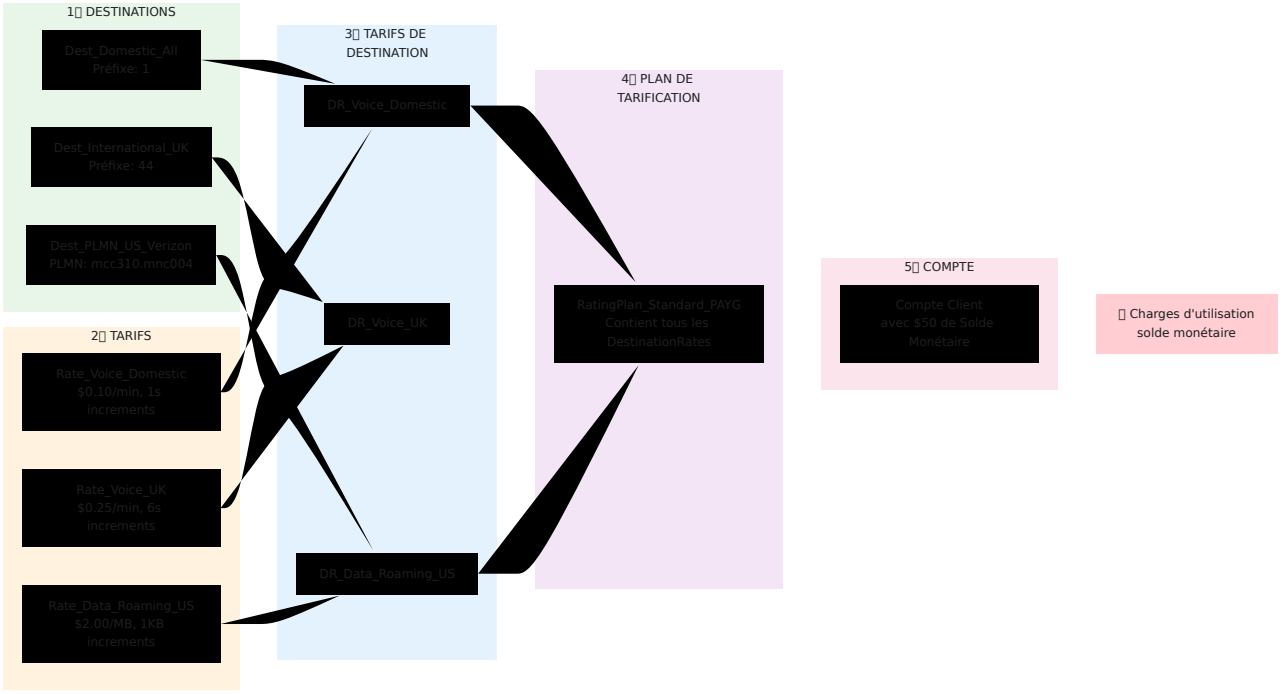
Remarque Importante sur la Flexibilité de CGRateS : CGRateS est extrêmement flexible et peut appliquer des tarifs et des plans différents en fonction de nombreux facteurs, y compris :

- D'où le client appelle (PLMN - couvert ici)
- Vers quoi il appelle (destination géographique - couvert ici)
- Heure de la journée (tarification de pointe/hors pointe)
- Niveau de client (VIP, standard, etc.)

- Niveau de qualité de service
- Et de nombreux scénarios de tarification plus complexes

Ce guide se concentre sur la **configuration au niveau CRM** pour le roaming basé sur PLMN et la tarification basée sur des 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 hors du cadre de cette documentation axée 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 UK : $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 US (Verizon, AT&T) : $2.00/MB
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 uniques par session

- **Rate** - Coût par RateUnit
- **RateUnit** - Taille de l'unité de facturation (60s, 1Mo, 1 message)
- **RateIncrement** - Unité minimale facturable
- **GroupIntervalStart** - Quand ce niveau de tarif commence

Lien entre Destinations et Tarifs

Après avoir défini les tarifs, liez-les aux destinations en utilisant les DestinationRates :

```
# Lier la destination domestique au tarif de 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 UK au tarif de voix UK
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 de voix (poids plus élevé = plus spécifique)
            {"DestinationRatesId": "DR_Voice_UK", "TimingId":
"*any", "Weight": 40},
            {"DestinationRatesId": "DR_Voice_Domestic",
"TimingId": "*any", "Weight": 20},

            # Tarifs de 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
    }]
})

# Recharger le cache
OCS_Obj.SendData({
    'method': 'CacheSv1.ReloadCache',
    'params': [{
        "Tenant": tenant,
        "DestinationIDs": ["*all"]
    }]
})
```

Voir aussi :

- Définir des Produits - Flux de travail 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

```
# Bon - réseau domicile spécifique
"DestinationIDs": "Dest_PLMN_OnNet",  # mcc505.mnc057

# Mauvais - trop large, correspond aux concurrents
"DestinationIDs": "Dest_PLMN_AU_All",  # mcc505 (tous les
opérateurs AU)
```

2. Utilisez des Noms de Destination Significatifs

```
# Bon - auto-documentant
Dest_PLMN_US_Verizon
Dest_International_UK
Dest_PLMN_Zone_Europe

# Mauvais - peu clair
Dest_Data_1
Dest_Voice_Package
```

3. Regroupez Logiquement les Destinations Associées

```
# Bon - regroupement logique
Dest_Domestic_All
Dest_Domestic_TollFree
Dest_Domestic_Premium

# Mauvais - 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
réseau
# Vérifiez les messages CCR Diameter pour l'AVP Visited-PLMN-Id
```

Problème : Utilisation des Données Comptée comme Roaming

Symptôme : L'utilisation des données domestiques 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 est (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
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, les approches de solde et les stratégies de tarification

- Définir des Produits - Flux de travail complet pour la création de produits

Ansible Playbooks : Guide détaillé

Les produits OmniCRM sont provisionnés à l'aide de **Ansible**, permettant une gestion automatisée des services basée sur les exigences spécifiques de chaque produit et de son inventaire associé.

Voir aussi : `SIM Card Provisioning <concepts_sim_provisioning>` pour un exemple complet de provisioning basé sur Ansible pour les services mobiles, y compris les SIM physiques et les eSIM.

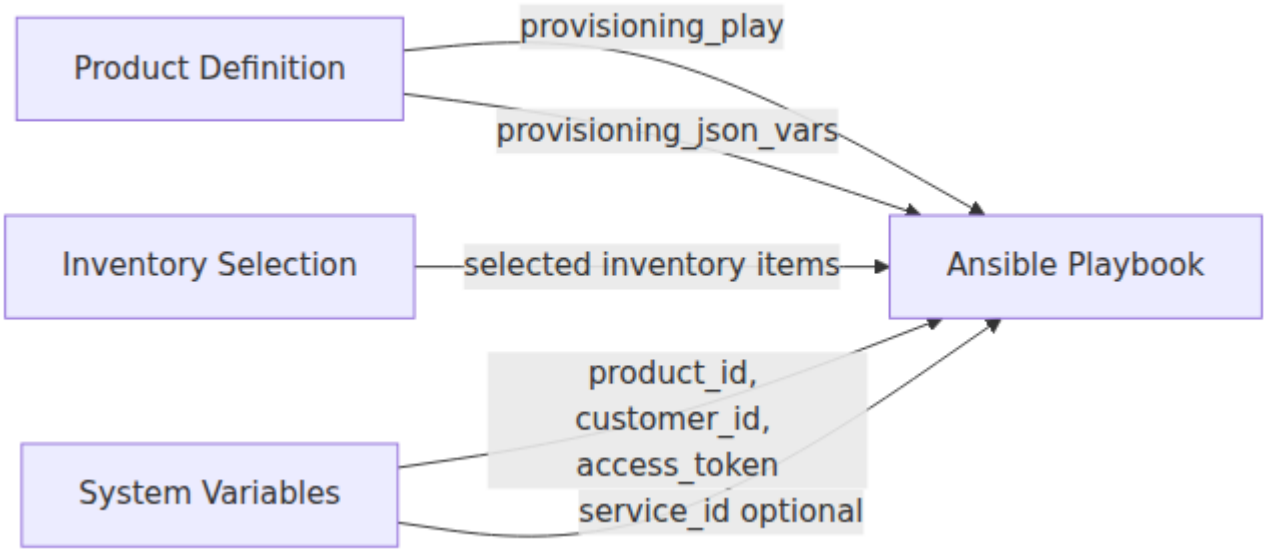
Comment les Playbooks et les Produits Fonctionnent Ensemble

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 - mais cela peut signifier des choses différentes 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 provisioning peut :

Créer Plusieurs Services
Un playbook de produit groupé pourrait créer :

- Un enregistrement de service Internet principal
- 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 du tout d'enregistrements de service :

- Un playbook qui configure simplement l'équipement CPE
- Un playbook qui envoie des configurations 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 des comptes
- 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
- Un seul compte de facturation 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 la 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 provisioning complet qui orchestre plusieurs tâches pour atteindre un objectif commercial. Les Plays sont les playbooks de niveau supérieur stockés dans `OmniCRM-API/Provisioners/plays/` et sont référencés dans les définitions de produits.

Exemples :

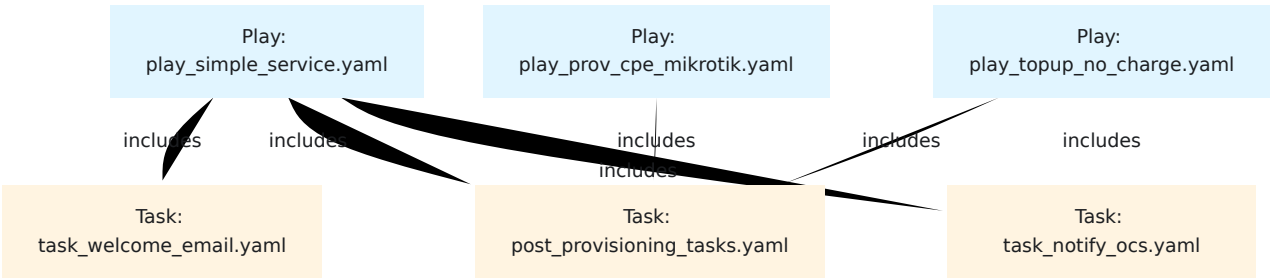
- `play_simple_service.yaml` - Provisionner un service de base
- `play_topup_no_charge.yaml` - Appliquer une recharge gratuite à un service
- `play_prov_cpe_mikrotik.yaml` - Configurer l'équipement sur site du client

Tâche (Composant Réutilisable)
Un ensemble d'opérations autonome et réutilisable qui peut être inclus par plusieurs plays. Les Tâches sont préfixées par `task_` et vivent 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 un 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 des 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 forme de `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 à product_id passé par OmniCRM
  debug:
    msg: "Provisionnement du produit {{ product_id }}"
```

De la Sélection de l'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 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 configuration
      - name: Ajouter une Transaction de Coût de Configuration via
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
Configuration",
              "description": "Coûts de configuration 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

Utilisé pour ajouter des crédits, des données ou du temps aux services existants.

```yaml
- name: Playbook de Recharge 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 qui doit être
rechargé)
    - 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 sur site du client (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 PSK WiFi
```

```
      set_fact:
        random_word: "{{ word_list | random }}"
        random_number: "{{ 99999 | random(start=10000) }}"

    - name: Combiner PSK WiFi
      set_fact:
        wifi_psk: "{{ random_word }}{{ random_number }}"

    # 7. Générer le fichier de configuration
    - name: Définir le nom de fichier de configuration
      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 configuration
      ansible.builtin.slurp:
        src: "{{ config_dest }}"
      register: config_content

    # 9. Mettre à jour l'inventaire avec les informations de
provisioning
    - 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: Email de configuration 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 frais récurrents 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 AutoRenew
  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 attend 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
clients
  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 configuration
: ${{ 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 post-provisionnement courantes

- include_tasks: task_notify_ocs.yaml
```

Le `task_notify_ocs.yaml` pourrait contenir :

```
- name: Notifier 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 30 jours à partir de
maintenant
  command: "date --utc +%Y-%m-%dT%H:%M:%S%z -d '+30 days'"
  register: expiry_date

- name: Calculer la date 90 jours dans le futur
  command: "date --utc +%Y-%m-%d -d '+{{ days }} days'"
  register: future_date
  vars:
    days: 90
```

Générer des 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": "Data Package",
            "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 des 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 de l'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 déprovision
  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 }
```

Boucler 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 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

Utilisez 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 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 si cela 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 }}"
```

Utilisez 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 la vérification de santé
      assert:
        that:
          - health_check.status == 200
```

Pièges Courants

Conversions de type manquantes :

```yaml
# Mauvais - peut être une chaîne
customer_id: "{{ customer_id }}"

# Correct - assurer un entier
customer_id: {{ customer_id | int }}
```

Ne pas gérer les variables indéfinies :

```
# Mauvais - échoue si non défini
service_uuid: "{{ service_uuid }}"

# Correct - fournir une valeur par défaut
service_uuid: "{{ service_uuid | default('') }}"
```

Oublier la validation :

```
# 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 s'assurant 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 pour le 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 principales façons 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:
        # Les 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 de `play_simple_service.yaml` :

```
rescue:
  # 1. Informations de débogage
  - name: Imprimer toutes les vars pour Déprovision/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 Déprovision
    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 c'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 ne nettoyer que 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 les ressources dépendantes en premier (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 de retour 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 (Frais Récurrents) :

```
- 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 Déprovisionnement Manuel

Pour déprovisionner un service manuellement :

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 la 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 largement

Le nettoyage doit être tolérant - ne pas échouer si une ressource n'existe pas.

2. Vérifier l'existence des ressources avec des clauses when

Tenter de nettoyer uniquement 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 frais récurrents
- 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
- Non recommandé pour la production

Rollback et Gestion des Erreurs

La fonctionnalité **block/rescue** d'Ansible est utilisée pendant le provisionnement et le 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 si une tâche génère une erreur :
    # 1. L'exécution passe 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 :

```
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 de facturation des services / produits CRM

Note

Pour un guide complet de bout en bout couvrant la définition des produits, la fourniture, les addons et la déprovisionnement avec des exemples Ansible détaillés et une stratégie de tarification, voir `Complete Product Lifecycle Guide <guide_product_lifecycle>`.

Pour des informations détaillées sur la fourniture de services mobiles avec des cartes SIM, voir `SIM Card Provisioning <concepts_sim_provisioning>`.

Aperçu des Produits et Services

Produit (Élément de Menu) :

Un Produit est comme un plat spécifique sur un menu de restaurant, tel que 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 — fonctionnalités, 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 personnalisations ou modifications d'un service dépend de vous (l'opérateur) à définir.

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 se reflète 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 dans 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 un client commande un article du menu, le plat est préparé dans la cuisine. Cela est similaire à la création d'un Service à partir d'un Produit.

Dans OmniCRM, lorsque un client sélectionne un Produit, une instance de ce Produit est créée et livrée en tant que 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 provisionnement "cuisine" 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 bundles, 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 "Pack Essentiels à Domicile" qui inclut les services Internet, câble et téléphone à un tarif réduit.

Une fois sélectionné, ce bundle est transformé en plusieurs Services adaptés au client.

Définitions de Produits

Un produit est un modèle utilisé pour créer un service / addon / remise / ajout, etc.

À l'intérieur de 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 autour de qui peut acheter le produit (*Commercial* ou *Résidentiel*), s'il dépend d'un service parent provisionné (comme les addons mobiles uniquement disponibles pour les clients avec un service mobile), s'il peut être commandé directement par un client via l'auto-assistance ou seulement par un agent du service client, et quand le produit

peut être acheté (permettant à un produit d'être disponible uniquement pendant une période déterminée).

- 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 :

  `['SIM Card', 'Phone Number']` Ceux-ci correspondent aux `Inventory Items <administration_inventory>` définis dans le CRM.

- Référence à un Playbook Ansible pour provisionner le service `Provisioning Play <concepts_ansible>` ainsi que les variables à passer à Ansible. Ces variables à passer 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 passés lors de l'attribution de l'inventaire. Le bundling 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 addon à des services existants)
- **promo** - Offres promotionnelles spéciales

Ces catégories sont purement organisationnelles et ne dictent pas ce qui est provisionné. Le comportement de provisionnement réel est déterminé entièrement 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 addon, 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 câblés
- **fixed-voice** - Services de voix fixe (VoIP, ligne terrestre)
- **hotspot** - Dispositifs de point d'accès mobile ou de location
- **dongle** - Services de modem USB ou 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 addons 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 addons mobiles, tandis qu'un client avec un service `fixed` voit des addons 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 améliorée avec des onglets qui organisent 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, conditions)
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 uniques pour le détail et le gros
- **Taxe** - Configuration du pourcentage de taxe avec calcul automatique

Fonctionnalités du Mode Édition :

- **Sélecteur d'Icône** - 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'Icône :

Lors de l'édition du champ d'icône, une interface de sélection d'icône recherchable apparaît, vous permettant de parcourir visuellement et de sélectionner 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 la 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'addon (par exemple, les addons 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 passées 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 de l'inventaire disponible lors de la commande
    - Les identifiants d'inventaire sélectionnés sont passés 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 Vue :**
    - Fermer - Dismiss modal
    - 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 (gros bouton pour souligner)

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, pas d'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 addons s'appliquent à quels services.
- **comment** - Notes internes sur le produit pour référence 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 (définir 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 de configuration uniques facturés au client
- **wholesale_setup_cost** - Votre coût unique pour la mise en place du service
- **tax_percentage** - Pourcentage de taxe appliqué à ce produit (par exemple, 10 pour 10%, 12.5 pour 12.5%). Définir 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 → Transaction a $5.00 de taxe
- Produit avec 0% de taxe (exonéré de taxe) → Transaction a $0.00 de taxe
- Remplacement manuel de la 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 (définir à false 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 (true) ou si seuls le personnel peut l'ajouter (false)
- **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 à 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 (définir à false pour les achats uniques)

Contenu Orienté Client

- **terms** - Termes et conditions affichés aux clients avant l'achat (inclure les limitations, règles d'expiration, conditions d'utilisation)
- **features_list** - Liste des fonctionnalités et inclusions affichées aux clients (format de liste Python : `['Feature 1', 'Feature 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 passées 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, `['SIM Card', 'Mobile Number']`). Lorsqu'un client commande, il sera invité à sélectionner des éléments spécifiques de l'inventaire disponible. Les identifiants d'inventaire sélectionnés sont passés au playbook de provisionnement avec le type d'inventaire comme nom de variable.
- **relies_on_list** - Liste des identifiants de produit ou types de service qui doivent exister avant que ce produit puisse être ajouté. Utilisé pour les dépendances d'addon (par exemple, les addons 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 Produits

Produit Autonome (Carte SIM Mobile)

```
{
  "product_id": 1,
  "product_slug": "Mobile-SIM",
  "product_name": "Mobile SIM Only",
  "category": "standalone",
  "service_type": "mobile",
  "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,
  "contract_days": 0,
  "residential": true,
  "business": true,
  "enabled": true,
  "customer_can_purchase": true,
  "icon": "fa-solid fa-sim-card",
  "features_list": "['Australian Phone Number (04xxx)', 'Fastest speeds', 'Best coverage', 'Roaming on the Mainland']",
  "terms": "Must be activated within 6 months. All credit lost if service is not used for 12 months.",
  "comment": "Physical SIM card for use with Mobile Phones"
}
```


Ce produit autonome nécessite deux éléments d'inventaire (Carte SIM et Numéro de Mobile) et crée un nouveau service lorsqu'il est provisionné.

Produit Addon (Plan de Données Mensuel)


```
{
  "product_slug": "norfone-mobile-prepaid-mini",
  "product_name": "Norfone Mini Plan",
  "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": "['8GB of Ultra fast data', 'Unlimited Calls & Texts to Norfone users', '100 Minutes of Talk to Australia', '100 SMS to Australia', '30 Day Expiry']",
  "terms": "Credit expires after 30 days. Once data, voice or sms is used up, you will need to top up to continue using the service.",
  "comment": "Our smallest plan for light users"
}
```


Ce produit addon 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 Bundle (Pack Seniors)


```
{
  "product_slug": "Bundle-Seniors",
  "product_name": "Seniors Bundle",
  "category": "bundle",
  "service_type": "fixed",
  "provisioning_play": "play_seniors_package",
  "provisioning_json_vars": "{\"IPTV_Service_ID\":
\"SeniorBundle\"}",
  "inventory_items_list": "['Modem Router']",
  "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": "['20Mbps Download', '5Mbps Upload', 'Unlimited
Data', 'Home Voice', 'TV: Extra +£5 per month', '£60 Installation
Fee']",
  "terms": "6 Month Contract, must show senior citizen's card to
qualify",
  "comment": "20Mbps/2Mbps GPON Service + IPTV + Phone"
}
```


Ce produit bundle provisionne plusieurs services (Internet + IPTV + Téléphone) en utilisant un seul playbook. Il nécessite un élément d'inventaire (Modem Routeur).

Produit Addon (Recharge Simple)

```
{
  "product_slug": "Mobile-Topup-5",
  "product_name": "PAYG £5 Topup",
  "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": "['£5 credit', 'Valid for 180 days']",
  "terms": "Valid for 180 days or until all credit is used. See
our website for full rates",
  "comment": "£5 to use for Calls, SMS & Data"
}
```

Cet addon 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 Passées à Ansible

Comprendre comment les variables circulent de la définition du produit à travers l'API vers le playbook Ansible est essentiel 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 passées 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 passées à Ansible seront :

```
{
    "monthly_cost": 45,         # Remplacé par la demande
    "data_gb": 100,             # De provisioning_json_vars
    "product_id": 10,           # De la demande
    "customer_id": 456,         # De la demande
    "custom_param": "value",    # 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/clé API, ou généré à partir de `refresh_token` pour l'authentification utilisateur)
- `initiating_user` - L'ID de l'utilisateur qui a déclenché le provisionnement (ou premier administrateur pour les systèmes automatisés)
- Tous les champs du corps de la demande (`product_id`, `customer_id`, `service_id`, etc.)

Variables d'Inventaire

Lorsqu'un produit nécessite des éléments d'inventaire (par exemple, `inventory_items_list: "['SIM Card', 'Mobile Number']"`), 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. **Identifiants d'inventaire 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 identifiants d'inventaire** - Le playbook de provisionnement peut alors récupérer les détails complets de l'inventaire à partir de l'API CRM

Par exemple, si un utilisateur sélectionne : - Carte SIM avec inventory_id : 789 - Numéro de Mobile avec inventory_id : 101

Les variables passées au playbook incluent : - `SIM Card`: 789 - `Mobile Number`: 101

Le playbook peut alors utiliser ces identifiants pour récupérer les enregistrements d'inventaire complets (ICCID, IMSI, MSISDN, etc.) à partir de l'API CRM et utiliser ces informations pour provisionner le service sur l'équipement réseau.

Comment Ansible Reçoit les Variables

Le système de provisionnement passe toutes les variables fusionnées au playbook Ansible sous forme de `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 Passées aux Produits Addon

Lorsqu'un produit addon est provisionné, le système passe automatiquement :

- `product_id` - L'ID du produit addon étant provisionné
- `customer_id` - Le client qui possède le service
- `service_id` - L'ID du service auquel cet addon est ajouté (critique pour les addons)
- `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'Addon

Lorsqu'un client ajoute l'addon "Recharge de £5" à son service mobile (service_id : 123), le playbook reçoit des variables incluant :

- `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** à partir de 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'addon d'identifier exactement quel service modifier et d'appliquer les changements de manière appropriée.

Différence entre les Variables de Produits Autonomes et Addon

Produits Autonomes reçoivent :

- `product_id` - Le produit étant provisionné
- `customer_id` - Le client commandant le service
- Identifiants d'éléments d'inventaire (par exemple, `SIM Card`, `Mobile Number`) si le produit les nécessite
- `access_token` - Pour l'authentification API

Produits Addon reçoivent :

- `product_id` - Le produit addon étant provisionné
- `customer_id` - Le client qui possède le service
- `service_id` - **L'ID du service existant à modifier** (c'est la différence clé)
- `access_token` - Pour l'authentification API

La différence clé est `service_id` - cela indique au playbook quel service existant modifier ou ajouter.

Produits Groupés

Les produits groupés sont provisionnés comme des addons mais leur playbook peut créer plusieurs enregistrements de service. Ils reçoivent les mêmes variables que les addons, y compris :

- `product_id` - Le produit bundle
- `customer_id` - Le client
- `service_id` - Service parent (le cas échéant)
- Identifiants d'éléments d'inventaire (par exemple, `Modem Router`) si requis
- `access_token` - Pour l'authentification API

Le playbook de bundle (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. L'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.

Ajouter un Service : Sélection et Filtrage de Produits

Lors de l'ajout d'un service à un client (soit un nouveau service autonome soit un addon à un service existant), le système affiche les produits disponibles dans une interface en carrousel. Les produits affichés sont filtrés en fonction de plusieurs critères :

Filtrage des Produits pour les 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`
   - **Addons** : 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

Note

Filtrage au Niveau de l'API : L'API filtre automatiquement les produits par statut activé et dates de disponibilité à deux niveaux :

- **Points de Fin d'Achat/Sélection** (`/crm/product/`) - Utilisé par le modal Addons et PlanList 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 Fin de Gestion** (`/crm/product/paginated`) - Utilisé par la page de Gestion des Produits. Affiche TOUS les produits y compris les désactivés et hors dates de disponibilité, permettant aux administrateurs de gérer l'intégralité 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 les clients consommateurs
- **Plans de Service Commercial** - Produits commerciaux pour les clients commerciaux
- **Addons Individuels** - Packs d'addons résidentiels
- **Addons Commerciaux** - Packs d'addons commerciaux

Filtrage des Produits pour les Services Addon

Lors de l'ajout d'un addon à un **service existant**, un filtrage supplémentaire est appliqué :

1. **Correspondance du Type de Service** - Seuls les addons avec un `service_type` correspondant sont affichés :
   - Si le service existant a `service_type = "mobile"`, seuls les addons avec `service_type = "mobile"` sont affichés
   - Cela garantit que les clients mobiles ne voient que des addons mobiles, que les clients Internet ne voient que des addons Internet, etc.
2. **Vérification des Dépendances** - Si un addon a une `relies_on_list` :
   - Le système vérifie si le client a les produits/services requis
   - Seuls les addons dont les dépendances sont satisfaites sont affichés
3. **Filtre de Type de Client Identique** - Les addons sont toujours filtrés par `residential` contre `business` pour correspondre au type de client

Scénario d'Exemple de Filtrage

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 Addon Affichés** : Seuls les addons mobiles commerciaux (`business = true`, `category = "addon"`, `service_type = "mobile"`)
- **Produits Cachés** : Produits résidentiels, addons 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-assistance (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 des 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-assistance (modifiable, par défaut : vrai)
- **service_usage_visible_to_customer** - Si le client peut voir les détails d'utilisation/de solde (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 passées 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 (Défini 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 des 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 de 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 des Produits** - Produits affichés filtrés en fonction de :
   - Type de client (résidentiel/commercial)
   - Services existants (pour les dépendances d'addon 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 API et Ansible Détaillé

Lorsqu'un service est provisionné, la séquence suivante se produit :

Étape 1 : Création du 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 étant provisionné
- `service_id` - (Optionnel) ID du service existant pour les addons
- Sélections d'inventaire - Identifiants des éléments d'inventaire sélectionnés

Étape 2 : Assemblage des Variables

Le service de provisionnement fusionne les variables de plusieurs sources dans cet ordre :

1. `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 addons
    "SIM Card": 1001,           # De la sélection d'inventaire
    "Mobile Number": 1002,      # De la sélection d'inventaire
    "monthly_cost": 30,         # De provisioning_json_vars
    "data_gb": 50,              # De provisioning_json_vars
    "access_token": "eyJ...",   # Ajouté par le système
    "initiating_user": 5        # Ajouté par le système
}
```

Étape 3 : Création de l'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 d'état (initialement défini à 1 = en cours, puis mis à jour à 0 = succès, 2 = échoué, ou peut rester 1 si toujours en cours)
- `product_id`, `customer_id`, `service_id` - Références de contexte

Étape 4 : Exécution en Arrière-plan du Playbook

L'API lance un thread en arrière-plan qui :

1. Charge le YAML du playbook à partir de `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 (passées 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 et Journalisation des Événements (`ProvisioningEventHandler`)

À mesure que chaque tâche Ansible s'exécute, les événements sont capturés et stockés en tant qu'enregistrements `Provision_Event` :

- `event_name` - Nom de la tâche du playbook
- `event_number` - Numéro de séquence
- `provisioning_status` - Code d'état indiquant le résultat de la tâche :
  - **0** = Succès - Tâche terminée avec succès
  - **1** = En Cours - Tâche en cours d'exécution
  - **2** = Échoué - Échec critique qui arrête le provisionnement
  - **3** = Échoué (ignoré) - Tâche échouée mais les erreurs ont été ignorées (`ignore_errors: true` dans le playbook)
- `provisioning_result_json` - Résultats de la tâche avec des données sensibles expurgées

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 requis) - 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 (Home Subscriber Server) pour le provisionnement des abonnés
    - IMS (IP Multimedia Subsystem) pour l'enregistrement vocal
    - CGRateS/OCS pour la création de compte, la configuration de facturation, les plans tarifaires
    - Serveurs ENUM pour la cartographie 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": "Mobile SIM - 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 de l'État

Lorsque le playbook est terminé :

- **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 Addon vs Bundle

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 Addon (`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 de roaming international, crédit supplémentaire

Produits Groupés (`category: bundle`) :

- Semblables aux addons en termes de variables reçues
- Peuvent nécessiter certains éléments d'inventaire (par exemple, modem pour le bundle 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 : Bundle à domicile (Internet + IPTV + téléphone VoIP)

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 de manière élégante** 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 addons
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 étant provisionné
- `service_id` - Entier (uniquement pour les addons/groupes), service existant à modifier
- `access_token` - Chaîne, jeton JWT pour l'authentification API vers le CRM
- `initiating_user` - Entier, utilisateur qui a déclenché le provisionnement
- Plus tous les identifiants d'éléments d'inventaire : `{{ inventory_type }}: inventory_id`
- Plus toutes les variables de `provisioning_json_vars`
- Plus toutes les variables passées 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'Addon Simple

Pour un addon 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 besoins d'audit.

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 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 (données Go, minutes de voix, nombre de SMS)
- **Conversions de solde** - Conversion des 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 des configurations (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 à partir de OCS/CGRateS et affichées aux clients et au personnel en temps réel.

Comment l'Utilisation est Récupérée

Lorsque l'utilisation d'un 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 du 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`
        - Retourne les soldes organisés par type : DONNÉES, VOIX, SMS, MONÉTAIRE, DONNÉES_DONGLE
        - Chaque solde inclut : 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 d'auto-renouvellement 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": "Mobile Service",
  "service_uuid": "abc-123-def",
  "cgrates": {
    "BalanceMap": {
      "DATA": [{
        "ID": "DATA_10GB",
        "Value": 5368709120,
        "ExpirationDate": "2025-02-01T00:00:00Z",
        "custom_Name_hr": "10GB Data Pack",
        "custom_Expiration": "Feb 1, 2025",
        "custom_Description_String": "5 GB remaining"
      }],
      "VOICE": [{
        "ID": "VOICE_UNLIMITED",
        "Value": 999999999,
        "custom_Name_hr": "Unlimited Calls",
        "custom_Description_String": "Unlimited minutes"
      }],
      "MONETARY": [{
        "ID": "PREPAID_CREDIT",
        "Value": 25.50,
        "custom_Description_String": "$25.50 credit"
      }]
    },
    "ActionPlans": [...]
  }
}
```


5. **Affichage UI** - Les composants frontend affichent les données d'utilisation :

   - **ServiceUsage.js** - Composant principal d'affichage de l'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 la 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 de 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 : "$25.50 de crédit"

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 complètement caché du portail d'auto-assistance du client
- **service_usage_visible_to_customer** - Si faux, le service est visible mais les détails d'utilisation/de 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 transparent complet de l'utilisation (par défaut)

Mises à Jour d'Utilisation en Temps Réel

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 de CGRateS
- **Efficacité** : Seules les vues de service actives 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 quasi en temps réel au fur et à mesure que l'utilisation se produit.

Charges Récurrentes / AutoRenouvellement

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 par exemple un service GPON à $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 les `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 ne n'affectera aucun autre service.

Addons & Options Supplémentaires

Les Addons / Options Supplémentaires comme l'achat de données supplémentaires, des packs de roaming, des 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 ensuite 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, comme 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 $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 $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 selon les besoins depuis l'API CRM, actionnés via des Playbooks Ansible. Les `ActionPlans` peuvent être ajoutés selon les besoins, depuis le CRM, en ajoutant un Addon/Service et actionnés via des Playbooks Ansible.

Les comptes `OCS` peuvent être désactivés, ce qui arrêtera l'exécution des `ActionPlans` et empêchera les services d'être consommés.

Pour les Limites de Crédit, une valeur `ThresholdS` est définie selon la politique pour le produit.

Visualisation et Gestion des ActionPlans dans le CRM

Les ActionPlans (configurations d'auto-renouvellement) 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 du service dans 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 des Données d'ActionPlan** - Chaque ActionPlan retourné contient :

```
{
    "ActionPlanId": "ServiceID_abc-123-
def__ProductID_456__...",
    "PlanName": "Monthly_Renewal_Plan",
    "NextExecTime": "2025-02-01T00:00:00Z",
    "custom_NextExecTime_hr": "in 11 days",
    "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 l'ActionPlan s'exécutera prochainement

- `custom_NextExecTime_hr` - Temps lisible par l'homme jusqu'à l'exécution (par exemple, "dans 11 jours", "demain", "1er févr. 2025")
- `ActionPlanId_split_dict` - Dictionnaire avec des composants analysés de l'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` de la définition du produit
   - **Date de Renouvellement** - Affiche `custom_NextExecTime_hr` (lisible par l'homme)
   - **Actions** - Deux boutons :
     - **Renouveler Maintenant** - Provisionner immédiatement l'addon/renouvellement (contourne l'attente de l'exécution programmée)
     - **Supprimer l'Auto Renouvellement** - Annule le renouvellement automatique

   Lorsque Aucun ActionPlan n'Existe :

   - Le tableau affiche le message : "Aucun auto-renouvellement activé pour ce service"
   - Le client peut ajouter des addons renouvelables automatiquement pour activer le renouvellement automatique

Gestion des ActionPlans

Le personnel et les clients peuvent gérer les ActionPlans via l'UI :

Suppression d'un ActionPlan (Annulation de l'Auto-Renouvellement) :

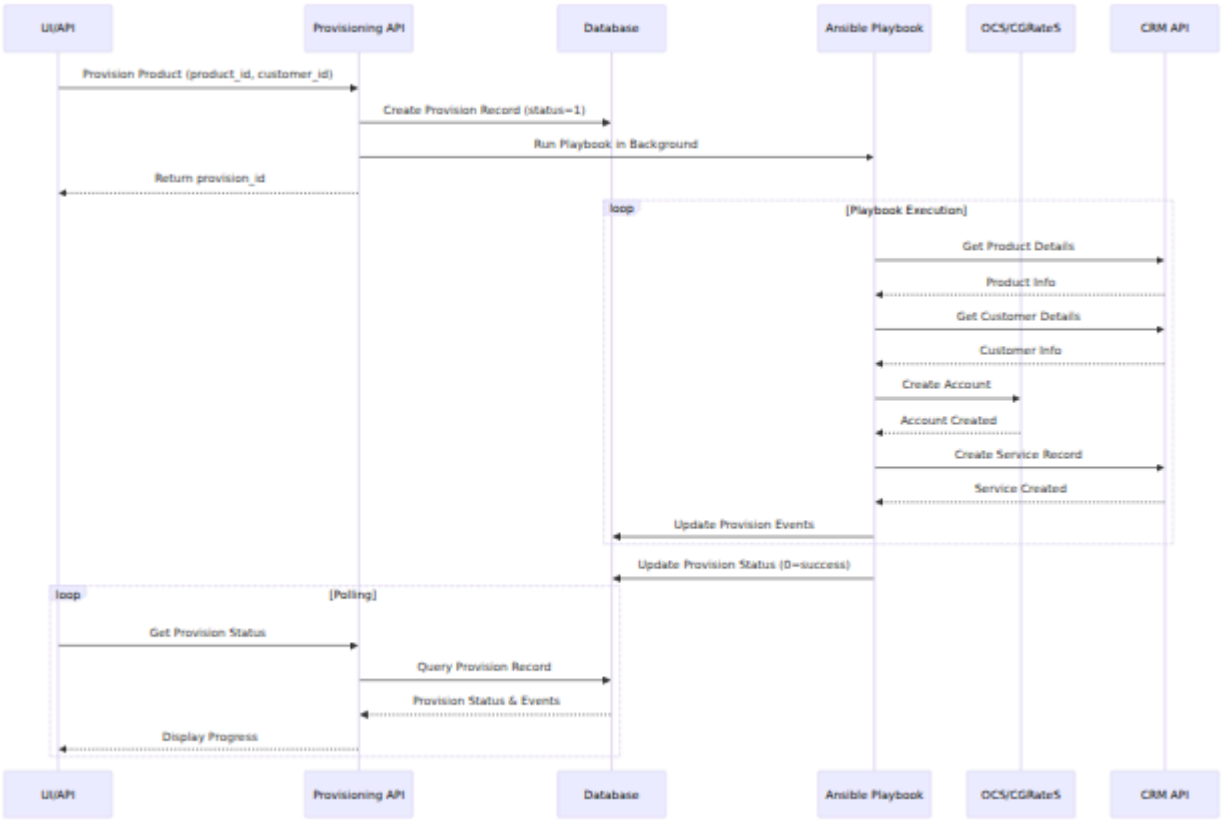
1. Cliquez sur le bouton "Supprimer l'Auto Renouvellement" dans ActionPlansTable

2. Une modale de confirmation apparaît : "Êtes-vous sûr de vouloir supprimer cet auto-renouvellement ?"
3. En cas de confirmation, le frontend appelle : `DELETE /crm/oam/remove_action_plan/{action_plan_id}`
4. L'API supprime l'Action

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 <concepts_sim_provisioning>` pour des informations détaillées sur le provisionnement des cartes SIM mobiles, y compris les cartes SIM physiques et les eSIM.



Note

Pour un guide complet du parcours produit-service avec des exemples détaillés de playbooks Ansible, des stratégies de tarification et des scénarios du monde réel, voir `Guide complet du cycle de vie des produits <guide_product_lifecycle>`.

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)** - Une défaillance critique s'est produite, 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 Déclenchent 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 les 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** : Toutes les variables supplémentaires fournies 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 ID 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 suit :

```
- 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 ID d'inventaire, il récupère les détails complets de l'inventaire depuis 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 ID d'inventaire sont passés au playbook comme `extra_vars`
5. Le playbook récupère les détails complets de l'inventaire depuis 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: ...

        # ... beaucoup d'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 :

- **Provision échouée (Rollback)** : Si une tâche dans le `block` échoue, la section `rescue` s'exécute automatiquement
- **Déprovisionnement Intentionnel** : Lorsqu'elle est appelée 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 au pool
- Les abonnés HSS sont supprimés ou mis en état dormant
- Aucun provisionnement partiel ne reste dans aucun système

Cela empêche des 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/l'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 selon l'intention.

5. Nettoyage Conditionnel Basé sur le Type de Service

Certaines tâches de nettoyage utilisent des conditionnelles 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 plus facile** : Les changements dans la logique de provisionnement s'appliquent automatiquement au nettoyage **Réduction des erreurs** : Pas de risque 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 surcharger celles-ci :

```
{
  "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 plafonds de données en fonction 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 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 le 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 d'Exécution en Arrière-Plan Généré

Le système génère un fil d'exécution en arrière-plan pour exécuter le playbook de manière asynchrone :

- Playbook : `play_psim_only.yaml`
- Variables : `extra_vars` fusionnées
- ID de provision : 456 (pour le suivi du statut)

- Jeton de rafraîchissement : passé pour le rafraîchissement du jeton pendant l'exécution

8. L'API Retourne Immédiatement

Réponse renvoyée 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 complétée 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 complétées et la tâche actuellement en cours
- 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 de 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 des Détails de Provision** - Montre la progression tâche par tâche avec des options d'extension/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 d'interroger 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
- **Auth par clé API** : Utilise l'ID du premier utilisateur admin
- **Auth par liste blanche IP** : Utilise l'ID du premier utilisateur admin

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 des détails sur les produits, créer des services, mettre à jour l'inventaire, etc.). L'authentification est gérée via 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 jeton JWT `access_token` (à durée limitée, 15-30 min) et un `refresh_token` (à longue durée)
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 en tant que 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 contre `crm_config.yaml`
3. **Génère un nouveau jeton JWT à la volée** pour le premier utilisateur admin
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 admin comme sujet du JWT

- Le jeton permet au playbook de faire des appels API authentifiés

Flux d'Implémentation :

Le système valide la clé API contre les clés configurées. Si valide et a le rôle admin, il génère un JWT temporaire en utilisant l'identité d'un utilisateur admin, puis le stocke pour une utilisation pendant le provisionnement.

Méthode 3 : Liste Blanche 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 contre `ip_whitelist` dans `crm_config.yaml`
3. Si sur liste blanche, **génère un nouveau jeton JWT** pour le premier utilisateur admin
4. Stocke le jeton pour le provisionnement
5. Passe à Ansible

Flux d'Implémentation :

Le système vérifie l'IP du client contre la liste blanche. Si autorisé, il génère un JWT pour un utilisateur admin et le stocke pour une utilisation pendant le 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 du `access_token` (expiration de 15-30 min). Pour les provisions initiées 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 nouveau `access_token`
4. Réessayer la demande échouée

Pour l'authentification par clé API/liste blanche 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 Job**

   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 requête
   - 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
   - **Auth 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 d'arrière-plan. Cela permet à l'API de retourner immédiatement pendant que le provisionnement continue de manière asynchrone.

   Pendant l'exécution :

   - Chaque tâche complétée/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 des informations sur le produit depuis l'API
   - Récupère des informations sur le client depuis 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 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 depuis 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 OCS déclenchant le provisionnement d'addon via des ActionPlans.

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 rapide qui nécessitent un retour immédiat

Authentification pour le Provisionnement Simple

Les points de terminaison de provisionnement simple utilisent **la liste blanche IP** ou **les clés API** pour l'authentification :

- L'IP source de la demande est vérifiée contre `ip_whitelist` dans `crm_config.yaml`
- Ou une clé API de `api_keys` dans `crm_config.yaml` peut être fournie
- Un jeton d'accès est généré et passé au playbook

Exemple : Callback d'ActionPlan 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 à :

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 depuis 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` pour 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 provoquent l'échec de l'ensemble du travail de provisionnement :

- Appels API retournant des codes de statut inattendus
- Échecs d'assertions (par exemple, `assert: that: response.status == 200`)
- Éléments d'inventaire requis manquants
- Équipement réseau inaccessible

- 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
- L'intégralité de stdout (contient les détails de l'erreur de syntaxe)
- L'intégralité de stderr (contient les erreurs d'exécution)
- Une liste des causes courantes de 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** : Tâches échouées mais ignorées
  - **Rouge** : Échecs critiques
- Pour les échecs critiques : Sortie de débogage complète incluant les corps de requête/réponse
- Pour les erreurs fatales : Sortie Ansible, messages d'erreur et causes courantes

Suivi des Travaux de Provisionnement

OmniCRM fournit des métriques complètes pour surveiller la performance du 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": "{...}"
    }
  ]
}
```


Liste 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 retourne 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 avec discernement** : Seulement pour des opérations vraiment optionnelles
- **Journalisez les variables importantes** : Utilisez des tâches `debug` pour enregistrer des valeurs clés pour le dépannage
- **Validez les réponses** : Utilisez `assert` pour vérifier que les réponses API sont conformes aux attentes
- **Idempotence** : Concevez des 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 IP ou des 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
- **Infos de débogage** : Capturer l'intégralité des corps de requête/réponse 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 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 de sondage** : L'UI doit interroger les 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 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

Journal d'Activité Client

Chaque changement apporté à un client, un contact, un site, un service et des paramètres financiers, comme les Transactions, les Factures et les Méthodes de Paiement, est enregistré dans le Journal d'Activité.

Cela nous permet de suivre les modifications apportées au système, qui les a effectuées, et quand elles ont été effectuées, et est utile pour auditer les changements et résoudre des 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 elle a été envoyée, ou 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, avec les changements les plus récents en haut, et les changements plus anciens plus bas dans la liste, qui peuvent être filtrés à partir des onglets.

Les enregistrements du journal d'activité ne peuvent pas être supprimés, mais ils peuvent être filtrés, et les détails peuvent être consultés 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 est simplement une instance d'un `Product <concepts_products_and_services>` pour un client donné, qui est sélectionnée dans le catalogue de produits et provisionnée pour le client.

Si vous n'avez pas encore `créé un client <basics_create_customer>`, vous devrez d'abord le faire, car les services sont liés aux clients, vous devrez 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), ou le personnel du Service Client peut provisionner des services pour le client.

En fonction de la définition du produit, il existe des règles concernant qui peut acheter un produit, comme n'autoriser que les clients professionnels à acheter un produit professionnel, ou n'autoriser que les clients ayant un service mobile à 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, et peuvent avoir plusieurs frais associés à eux, tels que des frais mensuels, des frais uniques, ou des frais d'utilisation, nous pouvons voir cela 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, et un service Internet à domicile pourrait avoir une adresse IP statique ou des données supplémentaires disponibles en tant qu'addon, encore une fois tout cela est défini dans le `product catalog<concepts_products_and_services>`.

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 Add-ons :

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 • Add-ons

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, vous serez invité 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.

Avertissement

Assurez-vous d'avoir sélectionné le bon client avant de continuer. Attribuer un plan au mauvais client nécessite une intervention manuelle pour corriger.

Étape 3 : Configurer l'Inventaire (si nécessaire)

Si le produit nécessite des articles d'inventaire (définis dans `inventory_items_list`), des sélecteurs d'inventaire apparaissent.

Exemple de Sélecteur d'Inventaire :

> Articles d'Inventaire Requis :
>
> Carte SIM *

> Cartes SIM Disponibles : • SIM-00123 - ICCID : 8944...0001 (Nouveau) • SIM-00124 - ICCID : 8944...0002 (Nouveau) • SIM-00125 - ICCID : 8944...0003 (Nouveau)
>
> 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 articles d'inventaire disponibles sont affichés (statut : "En Stock" ou "Nouveau")
- Les menus déroulants se chargent dynamiquement en fonction des noms de modèles d'inventaire
- Une fois sélectionnés, les articles sont temporairement réservés

Ce Qui Se Passe :

- Les articles d'inventaire sélectionnés sont passés comme variables au playbook de provisionnement
- Pendant le provisionnement, les articles sont attribués au service et au client
- Le statut de l'article 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, choisissez "Oui" pour les services aux consommateurs, "Non" pour les services uniques ou lorsque le client demande un contrôle manuel.

Étape 5 : Réviser et Confirmer

L'écran de révision 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 de 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 sur 6 tâches complétées

La fenêtre modale interroge l'API de provisionnement toutes les 0,2 secondes pour des mises à jour de statut.

Indicateurs de Progrès :

- ☑ 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 de 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 Provision : 456
>
> [Voir les Journaux] [Réessayer] [Fermer]

- **Voir les Journaux** : Ouvre les détails du provisionnement avec les 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 Particuliers

Ajout d'Add-ons à un Service Existant

Lors de l'attribution d'un add-on (catégorie : "addon") à un client qui a déjà un service :

1. **Détection Automatique de Service :**
    - Le système trouve les services existants du client
    - Filtre par service_type (l'add-on mobile ne s'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'add-on 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 un équipement au compte existant

Exemple :

> Services Mobiles Existants : • Mobile - +44 7700 900123 (ID : 789) • Mobile - +44 7700 900456 (ID : 790)
>
> Quel service devrait recevoir cet add-on ? [Mobile - +44 7700 900123 ▼]
>
> [Annuler] [Continuer]

Provisionnement pour 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 coût, 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 Bon 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. Un client entre dans le magasin en voulant un service mobile
2. Le personnel ouvre **Produits → Plans**
3. Bascule 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 la fin
13. Remet la SIM au client avec le paquet de bienvenue

Flux de Travail 2 : Ajout d'Internet à un Client Existant

1. Naviguer vers 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 le 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'un Add-on de Rechargement 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 "Add-ons" à côté du service mobile
5. Sélectionne "5 Go de Boost de Données"
6. Paiement autorisé
7. L'add-on 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 :** Articles d'inventaire en rupture de stock ou tous attribués
- **Solution :**
  - Ajouter plus d'articles d'inventaire au système
  - Vérifier le statut des articles (doit être "Nouveau" ou "En Stock")
  - Vérifier que les noms de modèles d'inventaire correspondent aux exigences du produit

"Client introuvable"

- **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 reste 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 OCS a le compte
  - Vérifier que la SIM est activée dans HLR/HSS
  - Vérifier le provisionnement réseau (RADIUS, DPI, etc.)
  - Réviser 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 termes et la tarification du produit avec le client

Pendant l'Attribution :

- Vérifiez deux fois la sélection du client avant de confirmer
- Sélectionnez les articles d'inventaire corrects (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 la fin

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 Add-ons :

- Confirmez que l'add-on est compatible avec le service existant
- Expliquez la facturation (unique vs récurrente)
- Vérifiez le paiement avant le provisionnement
- Vérifiez que le solde est mis à jour immédiatement après le provisionnement de l'add-on

Documentation Connexe

- `csa_add_service` - Aperçu des services
- `concepts_products_and_services` - Concepts du catalogue de produits
- `administration_inventory` - Gestion des articles d'inventaire
- `concepts_provisioning` - Détails du système de provisionnement
- `guide_product_lifecycle` - Cycle de vie complet du produit, y compris le provisionnement

Modification d'un Service

Les services peuvent être modifiés par le client final via le `Self-Care Portal <self_care_portal>`, 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 les 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 de Service
- Paramètres de Renouvellement Automatique
- Articles d'inventaire liés
- Champs personnalisés

Les modifications sont enregistrées immédiatement et reflétées dans le portail de self-care du client.

Ajout d'Options de 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 verrez 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 puces)
- 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 désirée, 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 (optionnel)

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 chaque période de facturation
- **Non** - Achat unique, le client doit renouveler manuellement

Étape 4 : Confirmer et Provisionner

Vérifiez les détails de l'option et cliquez sur **"Confirmer"**

La fenêtre 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 `Complete Product Lifecycle Guide - Adding Addons <guide_product_lifecycle>`.

Visualisation 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 **"Supprimer"** ou **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

Avertissement

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 une recharge de données instantanée

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 depuis le pool disponible (inventaire)

5. Configure le routage dans l'équipement réseau
6. L'entreprise reçoit les détails de configuration de l'IP

Scénario 3 : Location de Matériel pour Service Fibre

1. Le client s'inscrit pour internet fibre
2. Lors du 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. Modem expédié au client
6. Charge récurrente de £10/mois ajoutée 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 a 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 :** Drapeau de renouvellement automatique non défini ou méthode de paiement expirée
- **Solution :**
    - Vérifiez que le service a `auto_renew: true`
    - Vérifiez que le client a une méthode de paiement valide enregistrée
    - Consultez les travaux programmés 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 : `Adding Services <csa_add_service>`, `Service Usage <csa_service_usage>`, `Modifying Services <csa_modify>`, `Products & Services <concepts_products_and_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

Filtrez par type de service :

Sélectionnez un ou plusieurs types pour afficher uniquement les services correspondants.

Cas d'Utilisation :

- **Mobile** - Voir tous les services de SIM mobile
- **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

Recherchez et filtrez 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 d'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 numéros de téléphone contenant "0770".

Trier les Services

Cliquez sur l'en-tête de n'importe quelle 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 montre 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 montre 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

Avertissement

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 les comptes clients fermés
- Supprimer les 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 garde l'enregistrement
- **Suspendu → Actif :** Réactive le service, reprend la facturation

- **Inactif → Actif :** Peut déclencher le reprovisionnement (utilisez avec prudence)

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 de Données** - Utilisé vs. alloué (barre de progression)
- **Utilisation de Voix** - Minutes utilisées vs. allocation du plan
- **Utilisation de 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 renouvellement suivante.

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 un 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 programmée, 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 CGRateS directement 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 programmée :

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 Méga Coût : $89.99
>
> Prochain renouvellement programmé : 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 de 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 programmé

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 programmé pour :
>
> Client : Acme Corp Service : FixedWireless_75628fa5 Prochain Renouvellement : 11 nov. 2025 10:45 AM
>
> Le service ne se renouvellera pas automatiquement. Vous devrez 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

Avertissement

La suppression du renouvellement automatique signifie que le service expirera à moins d'être renouvelé manuellement. Les clients ne recevront pas de continuité de service automatique.

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 de 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 les détails des erreurs
  - Vérifiez que les playbooks Ansible sont configurés correctement
  - Vérifiez la connectivité OCS
  - Consultez 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 ►
>
> Articles par page : [10 ▼]
>
> - 10
> - 25
> - 50
> - 100

Contrôles :

- **Précédent/Suivant** - Naviguer entre les pages
- **Numéros de Page** - Passer à une page spécifique
- **Articles par Page** - Ajuster le nombre de services affichés par page

Astuce de Performance : Utilisez des filtres pour réduire le nombre total de r��sultats plutôt que d'augmenter le nombre d'articles 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 montrent :

Expiration Proche :

Les services expirant dans les 7 jours montrent :

En Retard :

Les services avec un solde impayé montrent :

Intégration CGRateS (Avancé)

Pour les services intégrés avec 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

Dans le 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.)
- Établissent des filtres pour garantir une tarification correcte
- Appliquent les plans de tarification appropriés

Cependant, il existe des cas où vous pourriez avoir besoin de modifier ces configurations après que le service a été provisionné :

- **Changements de Profil de Vitesse** - Le client améliore/réduit 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. Ceci est particulièrement utile pour :

- **Changements Rapides** - Modifier un seul attribut sans attendre le provisionnement
- **Tests** - Expérimenter avec différentes configurations
- **Support Client** - Résoudre des problèmes à la volée lors d'appels de support
- **Configurations Personnalisées** - Appliquer des paramètres spécifiques au service non couverts par des modèles

Avertissement

Les modifications manuelles à la configuration CGRateS contournent le flux de travail de provisionnement standard. Assurez-vous de comprendre l'impact de vos changements, car des configurations incorrectes peuvent affecter la

facturation et la fonctionnalité du service. Tous les changements sont enregistrés dans le fil d'activité du client à des fins d'audit.

Accéder aux Fonctionnalités CGRateS

Lors de la visualisation ou de l'édition d'un service provisionné dans CGRateS, trois sections rétractables 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/éditer la configuration.

Les sections repliées montrent 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 et des transformations personnalisés qui s'appliquent 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. Ajouter/modifier/supprimer 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 à partir des champs d'événement à l'aide de RSRParser
    - `*composed` - Ajoute la valeur au lieu d'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 politiques (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 changements effectués.

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. Ajouter/modifier/supprimer 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 :

- **Heure 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)
  - Heure 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
- Dépanner des problèmes de facturation

- Vérifier que la session est correctement tarifiée
- 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 des 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 ne s'enregistrant pas

- **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
  - Consultez les journaux CGRateS

Aucune session active affichée

- **Cause :** Aucune session actuellement active pour ce service
- **Solution :**
  - C'est normal si le service n'est pas utilisé
  - 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 ne montrant 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 par Type de Service** (optionnel)
2. Cliquez sur le **filtre par 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. Voir 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 intensifs 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 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'articles par page
  - Utilisez la recherche pour affiner les résultats

Documentation Connexe

- `csa_add_service` - Ajouter de nouveaux services
- `csa_modify` - Modifier des services et ajouter des options supplémentaires
- `concepts_products_and_services` - Concepts de produits et services
- `basics_customers` - Gestion des clients

Suivi de l'utilisation du service et du solde

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 le OCS (Système de Facturation en Ligne) pour afficher l'utilisation actuelle, les allocations restantes et les informations d'expiration du solde aux clients (via le `Self-Care Portal <self_care_portal>`) et au personnel.

Aperçu

Le suivi de l'utilisation permet :

- **Affichage du solde en temps réel** - Voir l'utilisation actuelle et les allocations restantes
- **Types de solde multiples** - Suivre simultanément les soldes de données, de voix, de SMS et monétaires
- **Surveillance des expirations** - Voir quand les soldes expirent
- **Répartition du solde** - 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

Le modal ou la page d'utilisation s'ouvre en montrant les données de consommation en temps réel.

Affichage de l'utilisation

L'interface d'utilisation montre des cartes récapitulatives et des barres de progression détaillées pour chaque type de solde.

Cartes récapitulatives

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
- **Heure d'expiration** - Jours/heures jusqu'à l'expiration du solde
- **Bouton Plus d'infos** - Cliquez pour développer la répartition détaillée

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 la répartition détaillée

Types de solde

Solde de données

Suit la consommation de données Internet.

Unités : Gigaoctets (Go) ou Mégaoctets (Mo)

Format d'affichage :

> Progression : 12.5 Go / 20 Go (62%)

Scénarios courants :

- **Forfaits de données mobiles** - Allocations mensuelles de 5 Go, 10 Go, 20 Go
- **Sans fil fixe** - Illimité ou plafonné à des montants élevés (500 Go, 1 To)
- **Recharges** - Données supplémentaires achetées en cours de cycle
- **Services de dongle** - Données prépayées pour des dispositifs 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 de l'appel 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'ils sont suivis 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

Répartition détaillée du solde

Cliquez sur **"Plus d'infos"** sur n'importe quelle carte ou cliquez sur une barre de progression pour développer la répartition détaillée.

Vue étendue :

> □ Allocation mensuelle 20 Go
> Restant : 12.5 Go Utilisé : 7.5 Go Expire : 25 Jan 2025 (15 jours) Poids : 10
>
> □ Données bonus 5 Go
> Restant : 5.0 Go Utilisé : 0 Go Expire : 31 Jan 2025 (21 jours) Poids : 20
>
> □ Données de recharge 3 Go
> Restant : 0 Go Utilisé : 3.0 Go Expire : 18 Jan 2025 (EXPIRÉ) 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 de solde, les stratégies de poids et comment CGRateS priorise les soldes, voir CGRateS Actions et Comportements de Recharge.

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
- Minuteries d'expiration
- Détails de seau individuels

Expérience utilisateur :

- Aucun rechargement de page requis
- Mises à jour fluides sans clignotement
- 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 : 12.5 Go restant VOIX : 125 min restant SMS : 45 msgs restant MONÉTAIRE : £15.50 restant

Sans fil fixe / Internet

Affiche généralement seulement :

- Données (Go ou To)
- Monétaire (si prépayé)

Exemple :

> DONNÉES : 450 Go / 500 Go 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, montrant uniquement les soldes de données et monétaires pertinents.

Dépannage

Utilisation affichée comme 0 / 0

- **Cause :** Service non intégré avec OCS ou CGRateS
- **Correction :**
  - Vérifiez que le service est provisionné dans OCS
  - Vérifiez la connectivité de l'API OCS
  - Vérifiez le mappage de l'UUID du service

Utilisation non mise à jour

- **Cause :** Polling arrêté ou OCS inaccessible
- **Correction :**
  - 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 des données OCS ou problème de cache
- **Correction :**
  - Forcer le rafraîchissement du solde OCS
  - Vérifiez les transactions en attente
  - Vérifiez la configuration OCS

Dates d'expiration manquantes

- **Cause :** Le solde n'a pas de date d'expiration définie

- **Correction :**
  - Certains soldes sont définis pour ne jamais expirer (validité illimitée)
  - Vérifiez la configuration du solde dans OCS

Soldes multiples déroutants

- **Cause :** Plusieurs recharges ou données bonus ajoutées
- **Correction :**
  - Utilisez la vue de répartition détaillée pour voir tous les seaux
  - Triez par poids pour voir l'ordre de consommation
  - Vérifiez les dates d'expiration individuelles

Intégration avec OCS/CGRateS

Les données d'utilisation proviennent du **OCS (Système de Facturation en Ligne)**, généralement CGRateS. Pour des informations complètes sur le fonctionnement des soldes, y compris les types de solde, les règles de consommation, le comportement de rollover et les configurations d'Action, voir CGRateS Actions et Comportements de Recharge.

Flux de données :

1. L'utilisateur ouvre la vue d'utilisation
2. OmniCRM appelle `GET /crm/service/{service_id}`
3. L'API interroge OCS via l'UUID du service
4. 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'interface utilisateur convertit les valeurs en unités d'affichage (octets → Go, nanosecondes → minutes)
6. Barres de progression et cartes 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'interface utilisateur :

- `*data` → **CARTE DE DONNÉES** (utilisation 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 de répartition détaillée.

Conversions de valeurs de solde :

- **Données :** Octets → Go (diviser par 1024^3)
- **Voix :** Nanosecondes → Minutes (diviser par 60×10^9)
- **SMS :** Compte (pas de conversion)
- **Monétaire :** Cents → Monnaie (diviser par 100)

Auto-renouvellement et Plans d'Action

Les services avec auto-renouvellement 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 CGRateS Actions et Comportements de Recharge.

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, du 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, 20 Go 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
   - Les Plans d'Action sont listés avec l'heure d'exécution suivante
   - Affiche : 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": "20 Go 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 auto-renouvelé)
- **Modifier** - Changer le montant ou la fréquence de renouvellement via la modification du service

Renouvellement manuel :

Si l'auto-renouvellement est désactivé, le client doit manuellement :

- Recharger avant l'expiration du solde
- Ou le service est suspendu lorsque le solde s'épuise

Visualisation dans l'UI :

L'onglet des services montre l'état de l'auto-renouvellement :

> Prochain renouvellement : 1er fév 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 la répartition détaillée pour identifier les seaux expirés
- Surveillez les clients à forte utilisation pour éviter les dépassements

Pour les clients (Self-Care) :

- 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 des rapports précis
- Vérifiez que les configurations de solde correspondent aux offres de produits

Documentation connexe

- CGRateS Actions et Comportements de Recharge - Guide complet sur les types de solde, les règles de consommation, les poids, le comportement de rollover et les configurations d'Action
- CGRateS Destinations - 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 supplémentaires 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 pour résoudre des problèmes, vérifier des configurations ou voir exactement ce que l'utilisateur expérimente. Cette fonctionnalité est essentielle pour le support client mais nécessite des autorisations appropriées et est entièrement audité.

Lors de l'usurpation d'un client, le personnel accède au `Self-Care Portal <self_care_portal>` exactement comme le client le voit, permettant un dépannage et un support précis.

Voir aussi : `RBAC <rbac>` pour la configuration des autorisations, `Customers <basics_customers>` pour la gestion des clients, `Self-Care Portal <self_care_portal>` 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 nécessiter de partage d'écran
5. **Piste de Vérification** — Toutes les sessions d'usurpation sont enregistrées pour la sécurité et la conformité

Autorisations Requises

Pour usurper 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 un Utilisateur

Via l'Interface Web :

1. **Naviguer vers le Client** — Trouver le client dans le CRM
2. **Sélectionner les Contacts** — Voir la liste des contacts du client
3. **Cliquer sur "Se connecter en tant qu'utilisateur"** — Un 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 un utilisateur :

- **Contexte Complet** — Vous voyez le système exactement comme l'utilisateur cible le voit :
  - 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 quelqu'un :
    - "Vous êtes actuellement connecté en tant que [Nom d'utilisateur]"
    - "Cliquez ici pour arrêter l'usurpation"
- **Journalisation d'Audit** — Toutes les actions sont enregistrées 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 usurpateur)
    - Piste 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 : Pas de 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 d'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 de l'accès élevé
- **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 usurpateur (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 voir leur compte (politique de confidentialité)
4. **Vérifier l'Identité** — Confirmez l'identité de l'utilisateur avant d'usurper via 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 sur les autorisations uniquement)

- **Actions Enregistrées** — Toutes les actions pendant l'usurpation sont attribuées à l'utilisateur cible dans les journaux d'application (mais les journaux d'audit montrent le véritable acteur)
- **Isolation de Session** — L'usurpation crée une nouvelle session ; n'affecte pas les sessions actives de l'utilisateur cible
- **Limites de Temps** — Les sessions d'usurpation doivent être limitées dans le temps (configurable)
- **Contournement MFA** — L'usurpation contourne les exigences 2FA (utilise l'authentification de l'usurpateur)

Restrictions

- **Impossible d'Usurper les Administrateurs** — Selon la configuration, il peut ne pas être possible d'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** — Impossible de désactiver ou de masquer 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 d'autorisation

- 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 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 sont pas correctement enregistrées

- 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 l'OCS (Système de Facturation en Ligne).

Vue d'ensemble

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 en fonction de 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 au Moindre Coût)

Ceci est juste le guide CRM - il se concentre sur des modèles de définition de produit simples qui fonctionnent bien avec le système de provisionnement d'OmniCRM. Pour des configurations CGRateS avancées, consultez la documentation CGRateS ou consultez Actions CGRateS et Comportements de Recharge pour des approches de solde et des stratégies de tarification.

Flux de Travail Complet pour la Définition de Produit

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 assignés 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** (obligatoire) - 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 numéro pour la voix/SMS VERS des endroits spécifiques (par exemple, `Dest_AU_Mobile`, `Dest_International_UK`)
2. **Destinations PLMN** - Codes réseau pour l'utilisation des données et le roaming (par exemple, `Dest_PLMN_OnNet`, `Dest_PLMN_US_Verizon`)

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 DEPUIS)

Pour des exemples complets de définition de destination 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 des champs (BalanceId, BalanceType, DestinationIDs, Units, ExpiryTime, Weight)
- Exemples de travail complets (plans multi-soldes, addons de données, addons de voix)
- Règles de consommation de solde et priorité

Référence Rapide au 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 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** (obligatoire) - Catégorie de produit :
  - `standalone` - Service complet (crée un nouvel enregistrement de service)
  - `addon` - Ajouté à un service existant
  - `bundle` - Paquet de plusieurs services
- **provisioning_play** (obligatoire) - 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 dépend ce produit
- **contract_days** - Durée du contrat en jours (0 = pas de contrat, 30 = mensuel)
- **retail_cost** (obligatoire) - Coût mensuel/récurrent pour le client

- **retail_setup_cost** - Frais d'installation unique facturés au client
- **product_slug** (obligatoire) - **Identifiant unique qui lie à l'Action CGRateS** (convention : `Action_{product_slug}`)
- **product_name** (obligatoire) - Nom affiché
- **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 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 courante 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 }}"
        }]
      }
```

Lorsque vous utilisez 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

Passons en revue la création d'un nouveau produit complet "Pack de Données de 50 Go" :

1. Définir l'Action dans CGRateS

Voir l'exemple Simple Data Addon 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 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 solde. Voir Exemple 1 : Plan Mensuel Multi-Solde (Python).

Modèle 2 : Addon de Données Simple (Rollover)

Cas d'utilisation : Achat unique de données 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', 'Réinitialisation
mensuelle', '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 à l'utilisation. 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, la localisation, le segment de client, la qualité du réseau

- **Tarification multidimensionnelle** : Combinaison de type de réseau, statut de roaming, destination et 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 l'utilisation (où l'utilisation au-delà des soldes inclus coûte de l'argent), vous devez définir des Plans de Tarification qui spécifient les 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 : Configurer des Charges de Données

à l'Usage

```
# 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", # Facturation 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 son 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",            # Facturation par minute
            "RateIncrement": "60s",       # Arrondi à la minute
pleine
            "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",        # Facturation par
seconde (plus précise)
            "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 les tests

# 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 discordance 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 fait référence à 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 non utilisé disparaît

Cause : Utilisation de `*topup_reset` au lieu de `*topup`

Solution :

- Changez `Identifier` de `*topup_reset` à `*topup`
- Assurez-vous que 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 forfaits 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
- Chargement basé sur CAMEL pour les appareils hérités 2G/3G
- Variations tarifaires en fonction de l'heure de la journée ou saisonnières
- Tarification basée sur des niveaux de clients (bronze/argent/or)
- Tarification basée sur la qualité du réseau ou QoS
- Routage au moindre coût avec plusieurs fournisseurs

... consultez directement la documentation CGRateS et travaillez sur la configuration de votre 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

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 situés dans des zones géographiques spécifiques. La Diffusion Cellulaire est une fonctionnalité critique de sécurité publique 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 de manière audible les téléphones qui sont en mode silencieux, à court de crédit ou en itinérance. Étant donné qu'il s'agit d'un message diffusé, il est possible d'envoyer une alerte à chaque membre de la population possédant un téléphone mobile en quelques secondes.

Lien d'Alerte Omnitouch (OWL)

La plateforme Lien d'Alerte 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'alerte 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), tel que des ordinateurs, des ordinateurs portables, des tablettes ou des téléphones mobiles.

Aperçu

La Diffusion Cellulaire (également connue sous le nom de Système d'Alerte Publique 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 dans 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 à des Systèmes Externes** - Se connecter à 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 conditions météorologiques sévères, alertes de tornade
- **Sécurité Publique** - Menaces terroristes, déversements chimiques, évacuations
- **Messages de Test** - Tests du système et campagnes de sensibilisation publique (Les Messages de Test Mensuels n'alerteront pas les utilisateurs mais seront reçus)

Le ciblage géographique est souvent d'une importance primordiale dans les situations d'urgence. Par exemple, les alertes de tsunami conseillant aux personnes 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'alerte d'urgence, la portée de la transmission peut être limitée à la zone géographique appropriée.

Cycle de Vie des Messages d'Alerte d'Urgence

La transmission d'un message d'Alerte 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 de la seconde personne (si nécessaire)
4. **Révision du Message** - Confirmation finale avant la 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'alerte d'urgence à diffuser :

Identifiant du Message
Différents types de messages ont différents identifiants, 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 de 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. Rappelez-vous 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 au besoin 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 n'affichera le message à l'utilisateur qu'une seule fois. Cependant, les sites cellulaires continueront à transmettre les 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'alerte 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 un 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'Alerte 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 codes avant les urgences

Pour des instructions détaillées sur la configuration de 2FA, voir `Authentification à Deux Facteurs <2fa>`.

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 puisse continuer. Cela fournit une supervision et minimise le risque d'abus.

Rôles Utilisateurs Granulaires
Des rôles d'utilisateur 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 réviser 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 du 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 du 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 deviennent à nouveau disponibles pendant que l'Alerte d'Urgence est toujours active, 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'Alerte** - Durée en secondes pendant laquelle l'alerte 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 visualiser 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 à partir de la navigation principale
2. **Cliquez sur "Créer une 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 un nombre spécifique)
    - Période d'Alerte (durée en secondes)
    - Indicateur de Canal (généralement "de base")

4. **Ajouter des Messages Localisés :**
    - Cliquez sur "Ajouter une 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 une 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 une 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` champ)
2. **Utilisateur Approuvant** - Responsable qui approuve l'alerte avant diffusion (`approving_user` champ)

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'Alerte de Tremblement de Terre et de Tsunami (ETWS) :

- **4370** - Alerte de Tremblement de Terre ETWS
- **4371** - Alerte de Tsunami ETWS
- **4372** - Alerte Combinée 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 des 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

(CBC) d'OWL se connecte à chaque réseau cellulaire pour envoyer des messages d'Alerte 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'Alerte d'Urgence via les autres réseaux disponibles.

Instances CBC par Opérateur
Pour garantir qu'il n'y a pas de connexion entre des 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 diffusion de messages 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 serveur selon le fournisseur de BSC
- Une connexion doit être établie entre tous les BSC du 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 du 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 faible nombre de réseaux 5G SA commercialement disponibles en 2025
- Sera entièrement prise en charge 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 séparée logiquement
- 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 d'Omnitouch périodiquement par e-mail.

Gestion des Utilisateurs et Contrôle d'Accès

Contrôle d'Accès Basé sur les Rôles (RBAC)

Le système OWL utilise un 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 attribués.

Composants RBAC :

- **Utilisateurs** - Personnes réelles qui se connectent au système OWL
- **Permissions** - Micro capacités (par exemple, approuver un message brouillon, créer un message, visualiser 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 <rbac>`.

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 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 Admin <authentication_flows>`.

Réinitialisation du 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 du Mot de Passe Admin :

Les administrateurs système peuvent réinitialiser les mots de passe des 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'admin
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 <rbac>`.

Ajout de Nouveaux Utilisateurs Système

Les administrateurs système peuvent ajouter de nouveaux utilisateurs en accédant à **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 sur 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 des utilisateurs existants :

1. Accédez à **Utilisateurs et Rôles → Utilisateurs**
2. Sélectionnez l'utilisateur à modifier
3. Modifiez les rôles attribués selon les besoins
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. Soyez très prudent 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

accédant à **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 la révision opérationnelle.

Accéder à l'Historique des Messages :

Accédez à **Gérer les Diffusions Cellulaires** pour voir :

- Tous les messages précédemment envoyés
- Statut du message (actif, expiré, annulé)
- Horodatage de transmission
- Utilisateurs initiateurs et approuvants
- Détails de ciblage géographique
- Contenu et paramètres du message

Cela aide à :

- Auditer le système
- Réviser 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 des zones de couverture géographique et identifier quelles tours cellulaires diffuseront des alertes.

Accéder à la Carte

Accédez à :

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 un 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 sous la carte
- Montre les noms et coordonnées des sites
- Fournit le compte des tours affectées

Exemple de Tableau 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 de 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 alertes de tsunami/surge de tempête
- **Zones de Feu 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
- Dossiers d'installation de tours
- Enquêtes de terrain avec GPS
- Configuration du contrôleur de réseau radio (RNC)

Style de la Carte :

Des styles de carte personnalisés peuvent être configurés dans `mapStyles.js` pour :

- Mettre en évidence les services d'urgence
- Afficher les caractéristiques topographiques
- Mettre en valeur les centres de population
- Correspondre à l'image de marque de l'organisation

Cas d'Utilisation

Planification d'Urgence :

- Pré-planifier des zones de couverture pour des zones de danger connues
- Tester différentes stratégies de ciblage
- Estimer la portée de la population
- Coordonner avec les services d'urgence

Vérification des Alertes :

- Confirmer que le message atteindra la zone prévue
- Identifier les lacunes de couverture
- Éviter de suralerter les régions adjacentes
- Valider les codes de zone de suivi

Analyse du Réseau :

- Visualiser la distribution des tours cellulaires
- Identifier les chevauchements de couverture
- Planifier les déploiements de tours pour une meilleure couverture d'alerte
- Optimiser 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 page)
- Utilisez des lettres majuscules pour les alertes d'urgence (améliore la lisibilité)
- Incluez des actions 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

- Considérez 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 aux bâtiments
- Signalez immédiatement les jetons perdus ou volés
- Testez régulièrement la 2FA pour vous assurer qu'elle fonctionne
- Enregistrez les codes de sauvegarde lors de la configuration de la 2FA et conservez-les en toute sécurité hors ligne
- Contactez l'administrateur système si vous avez besoin d'une réinitialisation de la 2FA

Pour plus d'informations sur la configuration et la récupération de la 2FA, voir `Authentification à Deux Facteurs <2fa>`.

Maintenance des Données

Zones Ciblées Prédéfinies :

À mesure que les frontières géographiques changent, que le développement se produit et que les zones à risque se déplacent, il est nécessaire de revoir les Zones Ciblées Prédéfinies. Omnitouch suggère que ces données soient examinées **annuellement par le NDMO**, avec le soutien d'autres agences d'alerte si nécessaire.

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 d'Omnitouch qui effectuera les modifications dans le système.

Modèles de Message Prédéfinis :

Les modèles de message prédéfinis doivent être examinés **au moins annuellement par le NDMO**, avec le soutien d'autres agences d'alerte si nécessaire, 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 d'Omnitouch qui effectuera les modifications dans le système.

Données des Sites Cellulaires :

- Examinez 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 niveau 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 direct sur matériel

Architecture Distribuée

L'architecture distribuée permet :

- Aux Bureaux Locaux de Gestion des Catastrophes d'accéder à OWL CBE et de diffuser 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 pour leur région même si le principal NDMO (Bureau National de Gestion des Catastrophes) perd l'accès

Ceci est particulièrement précieux lorsque les MNO ont distribué leur réseau cellulaire avec des ressources locales BSC/MME.

Mise en Réseau CBE et CBC

Pour garantir qu'il n'y a pas de connexion entre des réseaux concurrents, chaque MNO a une instance CBC séparée (non partagée).

Exigences de mise en réseau entre CBE et CBC :

- Tout le trafic est crypté 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 pour 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 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** - Mettre en place 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 de caractères strictes 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 limit�� 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 approuvants
- 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 une congestion du réseau lorsque les SMS échouent

Expérience Utilisateur :

- L'alerte s'affiche comme une notification plein écran
- Un ton d'alerte unique joue
- L'alerte persiste jusqu'à ce qu'elle soit reconnue
- Aucune souscription utilisateur requise
- Ne peut pas être bloquée par les utilisateurs pour des 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 est 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 diffusion de messages de Diffusion Cellulaire :

SMS de Masse

Envoi de SMS réguliers à des individus pour des notifications supplémentaires.

- Souvent utilisé pour des alertes déclassées 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 Ligne Fixe

Appeler automatiquement des numéros de ligne fixe et jouer des messages d'alerte d'urgence.

- Lecture de message préenregistré
- Rendu de texte à la parole du message d'alerte 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'Alerte d'Urgence sur des canaux de médias sociaux officiels.

- Étend la portée au-delà du réseau cellulaire
- Fournit une référence pour ceux qui ont manqué l'alerte initiale
- Permet un messaging prolongé au-delà des limites de caractères

Transmission Automatique Radio/TV

Diffuser automatiquement des messages d'urgence via la radio et la télévision.

- Lecture de message préenregistré
- Rendu de texte à la parole du message d'alerte d'urgence
- Atteint des populations pendant les pannes de réseau
- Complète la Diffusion Cellulaire pour une couverture complète

Dispositifs d'Alerte Externes

Le CBC OWL peut se connecter à une variété de sources externes :

- Plateformes de médias sociaux
- APIs 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 doivent 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'Alerte 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'alerte publique 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 d'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 d'Omnitouch périodiquement par e-mail dans divers formats.

Fréquence de Mise à Jour : Examinez et mettez à jour les données des sites cellulaires chaque fois que des changements de réseau se produisent, généralement mensuellement ou trimestriellement.

Zones Ciblées Prédéfinies

À mesure que les frontières géographiques changent, que le développement se produit et que les zones à risque se déplacent, il est nécessaire de revoir les Zones Ciblées Prédéfinies utilisées dans l'étape de Ciblage.

Révision Annuelle : Les zones cibles prédéfinies doivent être examinées 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 d'Omnitouch.

Modèles de Message Prédéfinis

Révision Annuelle : Les modèles de message prédéfinis doivent être examinés 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 d'Omnitouch.

Flux d'Approbation des Messages

Différentes régions ont différentes exigences concernant l'émission de 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 d'utilisateur 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 d'Omnitouch peut aider avec :

- Réinitialisation des jetons 2FA
- Réémission de jetons perdus/endommagés/expirés
- Audit de sécurité et gestion des jetons

Procédures de Sécurité des Jetons :

Lors de l'émission des jetons, 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. Un examen de sécurité est effectué
4. Un nouveau jeton est émis après vérification de la 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 :

- **Rapport/Suivi** - Vérifier le statut et la portée des messages transmis
- **Création de Message** - Créer et diffuser de nouveaux messages de manière programmatique
- **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 la 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 détaillées de l'API.

Recherche Globale

La fonctionnalité de Recherche Globale fournit une **interface de recherche unifiée** pour trouver rapidement des clients, contacts, services, inventaire et sites dans l'ensemble de la base de données OmniCRM.

Voir aussi : `Customers <basics_customers>`, `Inventory <administration_inventory>`, `Service Management <csa_service_management>`.

Accéder à la Recherche Globale

De 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-entities** 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 aux clients clôturés

Cas d'utilisation pour la recherche de comptes clôturés :

- Trouver des dossiers clients historiques
- Localiser des équipements provenant 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 défilable 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, défilant 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 renvoie 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 depuis les 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 du 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 et 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 possède 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
- **Inclure 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 de transaction
- Détails de facturation
- Journaux de provisionnement
- Entrées de 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 comptes clôturés pour réduire la portée de la recherche
- Envisagez d'ajouter des index de base de données sur les champs souvent 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 (décocher les comptes clôturés)
  - Contactez l'administrateur concernant l'indexation de la base de données

Les résultats renvoient 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 :** La case "Inclure les Comptes Clôturés" n'est pas 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 recharge OmniCRM fournit un **portail de recharge prépayé en libre-service** pour que les clients ajoutent du crédit ou prolongent la validité du service via le `Self-Care Portal <self_care_portal>`. 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 : `SIM Card Provisioning <concepts_sim_provisioning>` pour des informations sur la fourniture de services mobiles et la gestion de l'inventaire des SIM.

Aperçu

Le système de recharge permet aux clients d'acheter des jours supplémentaires de service via un processus de paiement simplifié en plusieurs étapes avec traitement de paiement Stripe intégré.

Caractéristiques Clés :

- Portail client en libre-service (aucune intervention du personnel requise)
- Sélection de durée flexible (1-30 jours)
- Affichage de l'utilisation en temps réel avant l'achat
- Traitement de paiement sécurisé alimenté par Stripe
- Remboursements automatiques si la recharge échoue
- Génération de factures et de transactions
- Intégration du système de provisionnement pour l'activation du service

Accéder au Portail de Recharge

Le portail de recharge 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 supports 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 de demande ou de son IMSI.

Processus de Recharge

Le flux de recharge 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** - Montre le coût total basé sur la sélection
- **Affichage de la date d'expiration** - Calcule et montre quand le service expirera
- **Affichage de l'utilisation actuelle** - Montre le solde restant/l'expiration avant la recharge

Affichage d'Exemple :

Configuration des Prix :

- 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 information 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 de la recharge échoue (par exemple, erreur réseau, OCS inaccessible), le système initie automatiquement un **remboursement complet** au moyen de paiement du client.

Étape 4 : Achèvement

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 la recharge échoue, le système affiche une erreur et traite automatiquement un remboursement :

> Nous n'avons pas pu compléter votre recharge. Votre paiement a été remboursé.
>
> Erreur : Impossible de se connecter au système de facturation
>
> Veuillez réessayer ou contacter le support.

Traitement Backend

Lorsqu'un 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 (prévention de double recharge)

2. Opération de Recharge

```
- API endpoint: POST /oam/topup_dongle
- Valide service_uuid et IMSI
- Appelle OCS/CGRateS pour ajouter du solde
- Crée un travail de provisionnement (play_topup_hotspot)
```

3. Création d'Enregistrement

Le système crée plusieurs enregistrements dans la base de données :

- **Enregistrement HotspotTopup** - Suit la transaction de recharge
  - payment_intent_id
  - service_uuid
  - imsi
  - jours achetés
  - topup_amount
  - statut (Succès/Échoué/Remboursé)
- **Enregistrement de Transaction** - Transaction financière
  - Titre : "Recharge Hotspot - 7 Jours"
  - Montant : topup_amount (positif)
  - Lié à service_id et customer_id

- **Enregistrement de Facture** - Facture de paiement
  - Contient la transaction de recharge
  - 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)
  - Lie le paiement de la facture au compte client

4. Travail de Provisionnement

Un travail 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 0,2 s, 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 de la Recharge. Remboursement du
paiement..."
```

Le remboursement est traité automatiquement et le client est notifié à l'écran.

Points de Terminaison API

Point de Terminaison de Recharge

```
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 à days : `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 ceci dans le fichier `.env` de l'OmniCRM-UI :

```
# Configuration du Portail de Recharge
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 recharge 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 sur l'état du 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

Structure d'exemple du playbook :

```
- name: Top up hotspot service
  hosts: localhost
  tasks:
    - name: Add balance to 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 du portail de recharge pour un accès facile aux clients.

Dépannage

Problèmes Courants

"Système de paiement indisponible"

- **Cause :** La bibliothèque Stripe a échoué à se charger ou clé publiable 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

"Échec de la recharge. Remboursement du paiement..."

- **Cause :** Le travail de provisionnement a échoué (OCS inaccessible, erreur de playbook, etc.)
- **Solution :**
  - Vérifiez les journaux de provisionnement : `GET /crm/provision/provision_id/<id>`
  - Vérifiez que l'API OCS/CGRateS est accessible
  - Passez en revue 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, actualiser après succès)

- **Solution :** Cela fonctionne comme prévu pour prévenir la double facturation. Le client doit commencer une nouvelle recharge si nécessaire.

"Le montant de l'intent de paiement ne correspond pas"

- **Cause :** Incohérence entre le calcul de l'UI et la validation backend
- **Solution :**
  - Vérifiez que `VITE_TOPUP_PRICE_PER_DAY` correspond à l'attente du backend (par défaut 10 $)
  - Vérifiez que la configuration de la devise est cohérente
  - Effacez le cache du navigateur et réessayez

Surveillance des Recharges

Voir les Enregistrements de Recharge :

Interrogez la table `HotspotTopup` pour voir toutes les tentatives de recharge :

```
SELECT
  hotspot_topup_id,
  service_uuid,
  days,
  topup_amount,
  status,
  payment_intent_id,
  created
FROM hotspot_topup
WHERE status = 'Failed'
ORDER BY created DESC;
```

Vérifier le Statut de Provisionnement :

```
GET /crm/provision/provision_id/<provision_id>
```

Montre le statut détaillé du travail de provisionnement de recharge.

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 sont gérées 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 garantit que la recharge va au bon service
- Les remboursements automatiques limitent l'exposition financière

Contrôle d'Accès :

- Le portail de recharge 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 recharges de service arbitraires
- L'administrateur peut voir tous les enregistrements de recharge via l'interface CRM

Meilleures Pratiques

Pour les Opérateurs :

1. **Tester le flux de remboursement** - Testez régulièrement les scénarios d'échec pour garantir que les remboursements fonctionnent
2. **Surveiller les recharges échouées** - Configurez des alertes pour des taux d'échec élevés
3. **Garder les playbooks simples** - Les playbooks de recharge 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 le lien de recharge aux favoris** - Accès rapide en cas de besoin
2. **Conserver les notifications de solde bas** - Le SMS contient 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** - Rechargez 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 recharge nécessitent des informations de dépannage détaillées
4. **Tester les chemins d'erreur** - V��rifiez 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` - Vue d'ensemble du système de provisionnement

- `Payment System Guide <payment_system_guide>` - Détails sur l'intégration du fournisseur de paiement
- `payments_transaction` - Gestion des transactions
- `payments_invoices` - Gestion des factures

Guide complet du cycle de vie du produit

Ce guide fournit un aperçu complet du cycle de vie du produit dans OmniCRM, de la création d'une définition de produit à la fourniture de services, en passant par l'ajout d'addons 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 addons/renflouements, 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 de la fourniture du 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 configuration

Les coûts de gros et de détail ont des variantes de coûts de configuration 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

Créons un produit de carte SIM mobile prépayée 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": "Mobile Prépayé 20 Go",
  "product_slug": "mobile-prepayé-20go",
  "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 addon/pack)
- `service_type` - Regroupe les produits connexes, utilisé pour le filtrage des addons
- `enabled` - Doit être vrai pour que le produit soit commandable
- `icon` - Icône FontAwesome affichée dans l'UI
- `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
}
```


Détails de la tarification :

- Revenu mensuel par client : £15.00
- Coût mensuel de livraison : £5.00
- Marge bénéficiaire mensuelle : £10.00 (200 % de majoration, 67 % de marge)
- Bénéfice de configuration : -£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 (renflouement manuel uniquement)
- `allow_auto_renew: true` - Le client peut activer/désactiver le renouvellement automatique plus tard

Étape 6 : Spécifier les exigences en matière d'inventaire

Les exigences en matière d'inventaire définissent quels 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 Inventaires <administration_inventory>`.

```
{
  "inventory_items_list": "['Carte SIM', 'Numéro de mobile']"
}
```

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, Désaffecté, etc.
- **Emplacement** - Emplacement physique ou logique

Comment fonctionnent les exigences en matière 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 existant dans le `Modèle d'Inventaire <administration_inventory>`.

Exemples d'exigences en matière d'inventaire :

```
# Produit de carte SIM mobile
inventory_items_list: "['Carte SIM', 'Numéro de mobile']"

# Service Internet fixe
inventory_items_list: "['Modem Routeur', 'Adresse IP Statique']"

# Service numérique (pas d'articles physiques)
inventory_items_list: "[]"

# Sans fil fixe avec CPE
inventory_items_list: "['CPE sans fil fixe', 'Adresse IPv4',
'Préfixe IPv6']"
```

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. Modal de sélection 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 "Nouveau" ou "En Stock" (pas "Assigné" ou "Endommagé")
- **Non Assigné** - `service_id` et `customer_id` sont NULL
- **En Stock à l'Emplacement** - Filtré éventuellement par emplacement d'entrepôt/magasin

Exemple 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 la SIM, numéro pour le 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 passés 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 passées 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: Obtenir les détails de la carte SIM depuis l'inventaire
  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: Extraire ICCID et 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: Assigner la carte SIM au 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, implémentez un verrouillage d'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 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 que l'inventaire a été attribué, le bloc de secours doit libérer l'inventaire :

```
rescue:
  - name: Libérer l'inventaire en cas d'échec
    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 entièrement ignoré et le provisionnement se poursuit immédiatement. Cela est courant pour :

- **Produits numériques** - Licences logicielles, comptes VPN
- **Addons de service** - Renflouements de données qui n'ont pas besoin 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, voir `Gestion des Inventaires <administration_inventory>`.

Plusieurs articles du même type

Bien que la `inventory_items_list` soit un tableau, avoir des types en double (par exemple, `"['Carte SIM', 'Carte SIM']"`) est **non recommandé** car cela peut causer de la confusion dans l'UI et le nommage des variables du playbook.

Pour des scénarios nécessitant plusieurs articles similaires :

Option 1 : Créer des noms de modèles d'inventaire distincts

```
# Service de téléphone à double SIM
inventory_items_list: "['Carte SIM Principale', 'Carte SIM
Secondaire', 'Numéro de mobile']"
```

Créez des modèles séparés : "Carte SIM Principale" et "Carte SIM Secondaire" avec les mêmes champs mais des noms différents.

Option 2 : Utiliser un seul article d'inventaire groupé

```
# Kit à double SIM
inventory_items_list: "['Kit à Double SIM', 'Numéro de mobile']"
```

Où le modèle d'inventaire "Kit à Double SIM" a des champs pour les deux SIM (`itemtext1`: ICCID principal, `itemtext2`: ICCID secondaire, etc.).

Scénarios d'inventaire courants

Service mobile :

```
inventory_items_list: "['Carte SIM', 'Numéro de mobile']"
```

- Carte SIM : Physique ou eSIM avec ICCID/IMSI
- Numéro de mobile : Numéro de téléphone (MSISDN)

Internet fixe :

```
inventory_items_list: "['Modem Routeur', 'Adresse IP Statique']"
```

- Modem Routeur : Dispositif CPE avec adresse MAC
- Adresse IP Statique : IPv4 du pool d'adresses

Sans fil fixe :

```
inventory_items_list: "['CPE sans fil fixe', 'Adresse IPv4',
'Préfixe IPv6']"
```

- CPE : Équipement sur site 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: "['Numéro DID']"
```

- 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: "['Appareil ONT', 'Port GPON', 'Adresse IPv4', 'Câble de Fibre']"
```

- Appareil 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: "['Modem de location']"
```

- Suit quel modem est avec quel client
- Important pour récupérer le matériel en cas d'annulation

Pourquoi les exigences en matière d'inventaire sont-elles importantes ?

1. Prévenir la double attribution

Sans suivi d'inventaire, vous pourriez accidentellement :

- Assigner 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 de vérification

L'attribution d'inventaire crée une piste de vérification 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
- Planifier les horaires des techniciens en fonction de la disponibilité du CPE
- Gérer 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 la dépréciation 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)

Résolution des problèmes liés au 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 "Nouveau" ou "En 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 que la réponse API pour le produit inclut inventory_items_list

Problème : Le playbook échoue avec "inventaire non trouvé"

Causes :

- Le playbook fait référence au mauvais nom de variable
- L'ID d'inventaire n'est pas passé correctement
- L'inventaire a été supprimé entre la sélection et le provisionnement

Solution :

- Vérifiez que le playbook utilise la bonne variable : `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

Voir `Gestion des Inventaires <administration_inventory>` pour des détails complets sur la création de modèles, l'ajout d'articles et la gestion des niveaux de stock.

Étape 7 : Définir les caractéristiques et les conditions

Les caractéristiques et les conditions sont des contenus marketing et juridiques visibles par le client qui aident les clients à comprendre ce qu'ils achètent et les obligations impliquées.

```
{
  "features_list": "Données haut débit 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, les
appels et les 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 caractéristiques - Marketing et ventes :

La liste des caractéristiques remplit plusieurs fonctions commerciales critiques :

1. **Différenciation des produits** - Aide les clients à comparer rapidement les produits et à choisir le bon
   - "Mobile Prépayé 20 Go" vs "Mobile Prépayé 50 Go" - les caractéristiques montrent clairement la différence
   - Sans caractéristiques, 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 caractéristiques 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 caractéristiques, 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 caractéristiques 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 - Juridique et conformité :

Les conditions servent des objectifs juridiques et opérationnels :

1. **Protection juridique** - 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" - empêche les abus (tethering de tout un bureau sur un plan mobile)
    - Crée un accord contraignant
2. **Gestion des attentes** - Prévenir l'insatisfaction des clients
    - "Valable uniquement pendant la période d'expiration" - le client connaît la date limite d'utilisation
    - "Ne peut pas être remboursé" (pour les addons) - empêche 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 la 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 de vérification** - 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 :

Le 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 caractéristiques :

Comprendre le format correct est critique car **un formatage incorrect casse l'affichage de l'UI**. Les caractéristiques peuvent apparaître comme une seule longue chaîne au lieu de points de puces, 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 caractéristiques sont séparées par un point et un espace (". "). L'UI se divise sur ce délimiteur et rend chaque caractéristique sous forme de point de puce.

Pourquoi ce format ?

- Simple à éditer - il suffit de taper les caractéristiques avec des points entre elles
- Aucun caractère spécial à échapper
- Fonctionne de manière fiable dans tous les composants de l'UI
- Facile à mettre à jour sans casser la syntaxe JSON

Correct vs Incorrect :

Option 2 : Chaîne de tableau JSON

```
"['Données haut débit de 20 Go', 'Appels et SMS illimités',
'Roaming UE inclus']"
```

L'UI 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 caractéristiques 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 à partir 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'UI (plus simple, moins sujet aux erreurs)
- **Tableau JSON** - Pour la création de produits basée sur API/script (plus robuste pour des caractéristiques complexes)

Les deux formats produisent un affichage identique dans l'UI - ils n'affectent que la manière dont vous saisissez les données.

Où apparaissent les caractéristiques dans l'UI :

1. Catalogue de produits (vue client)

Lorsque les clients parcourent les produits disponibles, les caractéristiques sont affichées sous forme de points de puces 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 configuration)
- Liste complète des caractéristiques (points de puces)
- Conditions générales (voir ci-dessous)
- Disponibilité et éligibilité

3. Confirmation de provisionnement

Lors du provisionnement, les caractéristiques sont affichées pour que l'utilisateur les examine avant de confirmer :

> Caractéristiques : • Données haut débit de 20 Go • Appels et SMS illimités • Roaming UE inclus • Pas de contrat • Expiration de 30 jours
>
> Coût : £15.00/mois Configuration : £0.00
>
> [Annuler] [Confirmer & Provisionner]

4. Détails du service (après provisionnement)

Une fois le service actif, les caractéristiques sont affichées sur la page de détails du service pour référence du client.

Format des conditions générales :

Le champ `terms` est un texte brut qui peut inclure des sauts de ligne :

Où apparaissent les conditions dans l'UI :

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 jusqu'à ce qu'il soit 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 :

- **Caractéristiques :** Gardez-les concises (moins de 50 caractères chacune), concentrez-vous sur les avantages clés
- **Conditions :** Inclure les exigences juridiques 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\": \"illimité\",
    \"sms_count\": \"illimité\"
  }"
}
```

- `provisioning_play` - Nom du playbook Ansible (sans l'extension .yaml)
- `provisioning_json_vars` - Variables par défaut passées au playbook

- Le playbook doit exister à : `OmniCRM-API/Provisioners/plays/play_local_mobile_sim.yaml`

Définition complète du produit

```
{
  "product_name": "Mobile Prépayé 20 Go",
  "product_slug": "mobile-prepayé-20go",
  "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": "['Carte SIM', 'Numéro de mobile']",

  "features_list": "[
    'Données haut débit 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, les
appels et les 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\": \"illimité\",
```

```
    \"sms_count\": \"illimité\"
  }"
}
```

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 de l'inventaire).

Exemple 1 : Addon virtuel (5 Go de données supplémentaires)

Un addon numérique qui ajoute des données à un service mobile existant :

```
{
  "product_name": "5 Go de données supplémentaires",
  "product_slug": "5go-de-données-supplémentaires",
  "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 haut débit 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 fibre existant :

```
{
  "product_name": "Location de modem WiFi 6",
  "product_slug": "location-modem-wifi6",
  "category": "addon",
  "service_type": "internet",
  "enabled": true,
  "icon": "fa-solid fa-router",
  "comment": "Ajoutez un modem WiFi 6 au service 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": "['Modem de location']",
  "relies_on_list": "",

  "features_list": "WiFi 6 (802.11ax). Double bande 2.4 GHz + 5
GHz. Jusqu'à 40 appareils. Contrôles parentaux",
  "terms": "Location de matériel. Doit être retourné en cas
d'annulation du service ou des frais de remplacement de 150 £
s'appliquent. Le matériel 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 `"['Modem de location']"` (matériel) - Ressources physiques
- `auto_renew: "false"` (unique) ou `"true"` (location) - Comportement récurrent
- `relies_on_list: ""` - Vide signifie s'applique à tout service de type `service_type` 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** - Impossible de provisionner l'addon si aucun modem n'est 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. **Dépréciation** - 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 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 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 ce que l'addon soit annulé

Déprovisionnement des addons matériels :

Lorsque le client annule la location du modem :

1. **Annulation initiée** - Le client clique sur "Supprimer l'addon"
2. **Processus de retour démarré** :
   - 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` = "En Stock" (après remise à neuf)
   - Ou `item_state` = "Endommagé" (si défectueux)
   - Lié au prochain client une fois remis à neuf
4. **Aucun retour** :
   - Après 14 jours, des frais de remplacement de 150 £ sont facturés
   - Inventaire marqué : `item_state` = "Perdu"
   - 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 configuration
- Les addons matériels peuvent avoir des frais de configuration 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 d'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 d'inventaire

Le système demande l'inventaire requis :

- **Carte SIM** - Le menu déroulant montre les cartes SIM disponibles en stock
  - Exemple : "SIM-00123 - ICCID : 8944..."
- **Numéro de 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 configuration : £0.00 (activation gratuite)
- Coût mensuel : £15.00
- À payer aujourd'hui : £15.00 (premier mois)
- Date de renouvellement : 30 jours à partir d'aujourd'hui

Si l'invite de renouvellement automatique est activée, 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 fil d'exécution en arrière-plan pour exécuter le playbook Ansible
- Renvoie le provision_id à l'UI 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": "illimité",
  "sms_count": "illimité"
}
```

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 passées à Ansible :


```
{
    "product_id": 42,
    "customer_id": 123,
    "SIM Card": 5001,
    "Mobile Number": 5002,
    "days": 30,
    "data_gb": 20,
    "voice_minutes": "illimité",
    "sms_count": "illimité",
    "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

Examinons un playbook de provisionnement réel 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. Assigner 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
```

Détails 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 du système, y compris :

- URL et identifiants OCS/CGRateS
- URL de base du CRM
- Configuration des locataires

Tâche 2 : Récupérer les détails du produit

```
- name: Obtenir les informations du 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 caractéristiques 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 des 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 d'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 de 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

```yaml
- 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 l'endroit où toutes les facturations se produisent
- Les soldes (données, voix, SMS, argent) y sont stockés
- 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 voix et SMS illimités

```
- name: Ajouter 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 : Assigner l'inventaire au service

```
- name: Assigner la carte SIM au 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 "En Stock" à "Assigné"
- Répète pour l'inventaire du numéro de mobile

Pourquoi :

- Suit quelle SIM est assignée 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 configuration

```
- name: Ajouter la transaction de coût de configuration
  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 }} - Configuration",
        "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 configuration 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 "En Stock"
- Échoue le job de provisionnement avec un message d'erreur

Pourquoi :

- Empêche les comptes orphelins dans OCS
- Assure 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)
- Bénéfice : £10.00/mois

3. Attributions d'inventaire :

- Carte SIM 5001 : Assignée au service 1234, client 123
- Numéro de mobile 5002 : Assigné au service 1234, client 123

4. Enregistrements de transaction :

- Transaction de coût de configuration créée
- Charge du premier mois enregistrée

5. Le client peut maintenant :

- Voir le service dans le portail en libre-service
- Voir le solde de 20 Go de données
- Passer des appels et envoyer des SMS
- Renflouer ou ajouter des addons
- Voir l'utilisation en temps réel

Étape 3 : Ajouter des addons et des renflouements

Après qu'un service soit actif, les clients peuvent acheter des addons pour améliorer leur service.

Flux de provisionnement des addons

Disons que le client a 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 "Renflouer"

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 à autonome :

- `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

- 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: Définir 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 le 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 jusqu'à ce que le paiement soit confirmé
- Prévenir la fraude
- Faire correspondre le paiement à la provision de l'addon

Tâche 3 : Ajouter un solde de données au compte OCS

```
- name: Ajouter un solde de 5 Go de données
  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 £
- Lie la transaction au service pour le 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 par le 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 d'un 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 demande : "Êtes-vous sûr ? Tout solde restant sera perdu."
- Le client confirme

2. Période de grâce (optionnelle)

Certains opérateurs mettent en œuvre une période de grâce :

- Service marqué "Annulation en attente"
- Reste actif pendant 7 à 30 jours
- Le client peut inverser 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 d'origine.

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": "Désactivé",
            "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": "Désaffecté"
          }
```

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 "Désaffectée", disponible pour réutilisation (après remise à neuf)

5. Post-déprovisionnement

Le système effectue un nettoyage :

- Le client ne voit plus le service dans le portail en 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` -

Intégration de Mailjet avec OmniCRM

OmniCRM s'intègre à **Mailjet** pour gérer toute la communication par email avec les clients et le personnel, garantissant une livraison d'emails professionnelle, de marque et fiable tant pour les emails transactionnels que pour les campagnes marketing.

Aperçu

L'intégration de Mailjet fournit :

- **Emails Transactionnels Automatisés** - Réinitialisations de mot de passe, factures, emails de bienvenue, notifications
- **Synchronisation des Contacts** - Contacts clients automatiquement synchronisés avec Mailjet pour les campagnes
- **Modèles d'Email** - Plus de 10 types d'emails préconfigurés avec des modèles Mailjet personnalisables
- **Campagnes Marketing** - Campagnes email segmentées basées sur les données clients
- **Livraison Fiable** - Infrastructure email professionnelle avec suivi et analyses

Configuration

Mailjet est configuré dans `OmniCRM-API/crm_config.yaml` sous la section `mailjet`.

Configuration de Base

```yaml
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'Email

OmniCRM utilise **10 types distincts de modèles d'email** pour les communications automatisées. Chaque modèle est configuré avec :

- **from_email** - Adresse email de l'expéditeur
- **from_name** - Nom d'affichage de l'expéditeur
- **template_id** - ID du modèle Mailjet (numérique)
- **subject** - Ligne de sujet de l'email

Types de Modèles et Configuration

Email de Bienvenue Client

Envoyé lorsqu'un nouveau compte client est créé.

```yaml
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 email 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

Email 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'email.

Rappel de Facture Client

Envoyé pour rappeler aux clients les factures impayées.

```
api_crmCommunicationCustomerInvoiceReminder:
  from_email: "billing@yourcompany.com"
  from_name: "Facturation de Votre Entreprise"
  template_id: 6759852
  subject: "Rappel de Paiement - Facture Échue"
```

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 }}`

Email de Bienvenue Utilisateur Staff

Envoyé lorsqu'un nouveau compte utilisateur staff 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 staff
- Le bouton "Envoyer l'Email de Bienvenue" est cliqué dans la gestion des utilisateurs

Variables de Modèle Disponibles :

- `{{ var:user_name }}` - Nom complet de l'utilisateur staff
- `{{ var:email }}` - Email de l'utilisateur staff
- `{{ var:role }}` - Rôle(s) assigné(s)
- `{{ var:login_url }}` - Lien vers la connexion du portail admin
- `{{ var:temp_password }}` - Mot de passe temporaire (si applicable)
- `{{ var:support_email }}` - Contact support IT

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 (généralement 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).

Réinitialisation de Mot de Passe Utilisateur Réussie

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 de son 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 (optionnel)
- `{{ 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 }}` - "Soi-même" ou nom de l'administrateur
- `{{ var:support_email }}`

Vérification de l'Email

Envoyé pour vérifier l'adresse email d'un utilisateur.

```
api_crmCommunicationEmailVerification:
  from_email: "noreply@yourcompany.com"
  from_name: "Votre Entreprise"
  template_id: 5977514
  subject: "Vérifiez Votre Adresse Email"
```

Quand Envoyé :

- Nouveau compte créé (client ou personnel)
- L'utilisateur change d'adresse email
- Vérification de l'email requise pour des raisons de sécurité

Variables de Modèle Disponibles :

- `{{ var:user_name }}`
- `{{ var:verification_url }}` - Lien pour vérifier l'email
- `{{ 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/ajouter du solde

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

Création de Modèles d'Email Mailjet

Pour chaque type d'email, 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'Email → 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 email.

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 de 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 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
email.</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 fonctionnalité **Email de Test**
2. Fournissez des valeurs d'exemple pour toutes les variables
3. Envoyez un email 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 changements prennent effet :

```
cd OmniCRM-API
sudo systemctl restart omnicrm-api
```

Synchronisation des Contacts

Tous les contacts clients dans OmniCRM sont automatiquement synchronisés avec Mailjet.

Ce qui est Synchronisé :

- Nom du contact
- Adresse email
- Type de contact (facturation, technique, etc.)
- Localisation du client
- Champs personnalisés

Fréquence de Synchronisation :

Les contacts se synchronisent en temps réel lorsque :

- Un nouveau client est créé
- Un contact est ajouté/mis à jour
- Les détails du client sont modifiés

Dépannage

Email 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 de modèle existe dans Mailjet
  - Consultez les journaux API pour les erreurs

Les variables de modèle ne se substituent pas

- **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 variable dans l'appel API
  - Testez avec des données d'exemple dans Mailjet

Le PDF de la facture ne s'attache pas

- **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 s'affiche correctement
  - Assurez-vous que le PDF fait moins de 15 Mo (limite Mailjet)
  - Testez la génération de PDF de facture séparément

Les contacts ne se synchronisent pas avec Mailjet

- **Cause :** Limite de taux API dépassée ou service de synchronisation non exécuté
- **Solution :**
  - Vérifiez les limites de taux API Mailjet (200 appels/minute)
  - Vérifiez que le service OmniCRM-API est en cours d'exécution
  - Consultez les journaux de synchronisation pour les erreurs
  - Déclenchez manuellement la synchronisation pour tester

Documentation Connexe

- `administration_configuration` - Référence complète de configuration de Mailjet
- `payments_invoices` - Génération de factures et livraison d'emails
- `authentication_flows` - Emails de réinitialisation de mot de passe et de vérification

- `customer_care` - Emails de bienvenue du portail d'auto-assistance
- Surveillance & Métriques - Métriques de l'API Mailjet, suivi de la livraison d'emails et surveillance de la performance

Lectures Complémentaires

- Documentation Mailjet : <https://dev.mailjet.com/>
- Référence API Mailjet : <https://dev.mailjet.com/email/reference/>

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.

Aperçu

Le système de métriques suit :

- **Opérations de Provisionnement** - Exécution de tâches, durée, taux de succès/échec
- **Performance de la Base de Données** - Temps de requête, santé du pool de connexions
- **Intégrations Externes** - OCS/CGRateS, appels API Stripe, Mailjet
- **Tâches en Arrière-plan** - Exécution de tâches asynchrones et performance
- **Requêtes HTTP** - Utilisation des endpoints API et temps de réponse (généré automatiquement)

Endpoint des Métriques

URL : `http://your-omnicrm-api:5000/crm/metrics`

Format : Format d'exposition Prometheus

Authentification : L'endpoint des métriques est accessible publiquement pour le scraping par Prometheus. En production, il est recommandé de restreindre l'accès l'aide de règles de pare-feu ou d'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 tâches 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 la 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 empruntées 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 empruntées au pool 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 ayant 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 Aperçu 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 à 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 de plans d'action CGRateS pour des charges récurrentes.

---

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 de 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 en plusieurs 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 vers 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 d'état 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 d'état HTTP

Description : Nombre total de requêtes HTTP par endpoint, méthode et code d'état.

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 d'état 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 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

```
- 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 les 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 des paiements Stripe
2. **Suivre les Métriques Commerciales** - Surveiller les KPI opérationnels :
    - Revenus traités via Stripe
    - Débit de provisionnement
    - Taux de livraison des emails aux clients
3. **Surveillance de la Performance** - Surveiller la dégradation :
    - Percentiles du temps de réponse de l'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

Rétention des Métriques

Recommandations de Rétention 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 des métriques :

```
curl http://localhost:5000/crm/metrics
```

Vérifiez que Prometheus peut scraper :

```
# 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 Cardinalité Élevée

Si Prometheus est lent ou consomme une mémoire excessive, vérifiez les étiquettes à 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
- Aperçu 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 un **traitement de paiement multi-fournisseurs** (Stripe, PayPal, etc.). Les méthodes de paiement permettent la facturation automatique pour les services, les paiements uniques et les frais récurrents sans stocker de données sensibles de carte dans OmniCRM.

Voir aussi : `Payment System Guide <payment_system_guide>`, `Billing Overview <billing_overview>`, `Payment Processing <payments_process>`, `Invoices <payments_invoices>`.

Aperçu

Le système de méthodes de paiement fournit :

- **Stockage Sécurisé des Cartes** - Cartes tokenisées par des 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** - Désigner la méthode de paiement préférée pour les frais automatiques
- **Suivi des Expirations** - Surveiller et mettre à jour les cartes expirantes
- **Auto-Service** - Les clients peuvent gérer leurs propres cartes via le `Self-Care Portal <self_care_portal>`
- **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 des fournisseurs de paiement et stockés en toute sécurité. OmniCRM ne stocke que :

- Fournisseur de paiement (stripe, paypal)
- Marque de 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 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 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 frais 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 fenêtre 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 retourné 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 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 les recharges
- Les 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 passe à 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

Retirer 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 le **menu d'actions (⋮)**
3. Sélectionnez **"Supprimer"**

Étape 2 : Confirmer la Suppression

Une fenêtre de confirmation apparaît :

> Êtes-vous sûr de vouloir supprimer cette méthode de paiement ?
>
> Carte : Visa se terminant par 1234 Expiration : 12/2026
>
> ⚠ Avertissement : Si c'est votre seule méthode de paiement, vous devrez en ajouter une nouvelle pour continuer à utiliser les services nécessitant une facturation automatique.
>
> [Annuler] [Supprimer la Méthode de Paiement]

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 par défaut si d'autres cartes existent** - Définissez d'abord une autre carte comme par d��faut

- **Avertissement si la dernière carte est supprimée** - Les services nécessitant un paiement peuvent être suspendus
- **Pas de retour en arrière** - La suppression est permanente ; le client doit réajouter la carte 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 Self-Care.

Variables de Modèle d'E-mail :

Les modèles Mailjet reçoivent :

- Nom du client
- Marque de carte et les 4 derniers chiffres
- Date d'expiration
- Lien vers la page des méthodes de paiement Self-Care

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 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 une 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 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

Avertissement

Ne demandez jamais aux clients d'envoyer par e-mail ou par texto les détails de leur carte. Utilisez toujours :

- Portail Self-Care sécurisé pour l'auto-service
- Téléphone avec le personnel saisissant les détails directement 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 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 est épuisé
    - **Services Auto-Renouvelables** - 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) marqué pour retour
- Avis final envoyé
- Compte référé à des agences de recouvrement (si applicable)

Prévenir les Interruptions de Service :

Pour éviter les interruptions 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 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 du fournisseur 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. **Jeton stocké dans OmniCRM** → Le numéro de carte complet n'est jamais stocké
4. **Traitement des paiements** → 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 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** - 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 `Payment System Guide <payment_system_guide>` 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 paiement
4. Saisit les détails de la carte dans la fenêtre Stripe
5. La carte est tokenisée et enregistrée
6. Définie automatiquement comme par défaut
7. Service provisionné
8. Premier frais 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 → Méthodes de Paiement**
4. Examine la carte actuelle expirant 12/2025
5. Clique sur **"Ajouter une Méthode de Paiement"**
6. Saisit une carte de remplacement avec une 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 la carte par téléphone
6. Le personnel navigue vers **Clients → [Client] → Facturation**
7. Clique sur **"Ajouter une Méthode de Paiement"**
8. Saisit les détails de la carte au fur et à mesure 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 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 la Mastercard au paiement
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 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'une méthode de paiement

- **Cause :** Le fournisseur de paiement a rejeté la carte (fonds insuffisants, prévention de la fraude, refus de l'émetteur)
- **Solution :**
  - Essayez une carte différente
  - 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

"Erreur lors de l'ajout de la méthode de paiement" (erreur générique)

- **Cause :** Erreur de l'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 publicités)

Impossible de supprimer la méthode de paiement (bouton désactivé)

- **Cause :** Tentative de suppression de la carte par défaut, ou c'est la seule carte
- **Solution :**
  - Définissez d'abord une autre carte comme par défaut
  - Si c'est la seule carte, ajoutez une nouvelle carte avant de supprimer

La carte apparaît comme expirée mais n'est pas dans la liste "Cartes Expirantes"

- **Cause :** Carte expirée récemment, cache non 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 ne s'est pas 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 pendant le processus d'ajout

La fenêtre 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 publicités et la protection du 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 en mode navigation privée/incognito

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 des échecs de livraison

Meilleures Pratiques

Pour les Clients :

- Ajoutez une méthode de paiement avant l'activation du service pour éviter des retards
- Gardez au moins 2 cartes en fichier pour la redondance
- Mettez à jour les cartes expirantes 30 jours ou plus 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 les détails de leur 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 uniquement après avoir confirmé que la nouvelle carte fonctionne
- Testez le paiement après avoir mis à jour une carte expirée (processus d'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/vie 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 manipulation sécurisée des cartes

Documentation Connexe

- `Payment System Guide <payment_system_guide>` - 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 Self-Care pour que les clients gèrent leurs propres cartes

Guide de l'API du Système de Paiement OmniCRM

Aperçu

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 de Méthodes de Paiement
5. API de Flux de Paiement
6. API de Portefeuille
7. Résumé de la Référence API
8. Cas d'Utilisation Courants

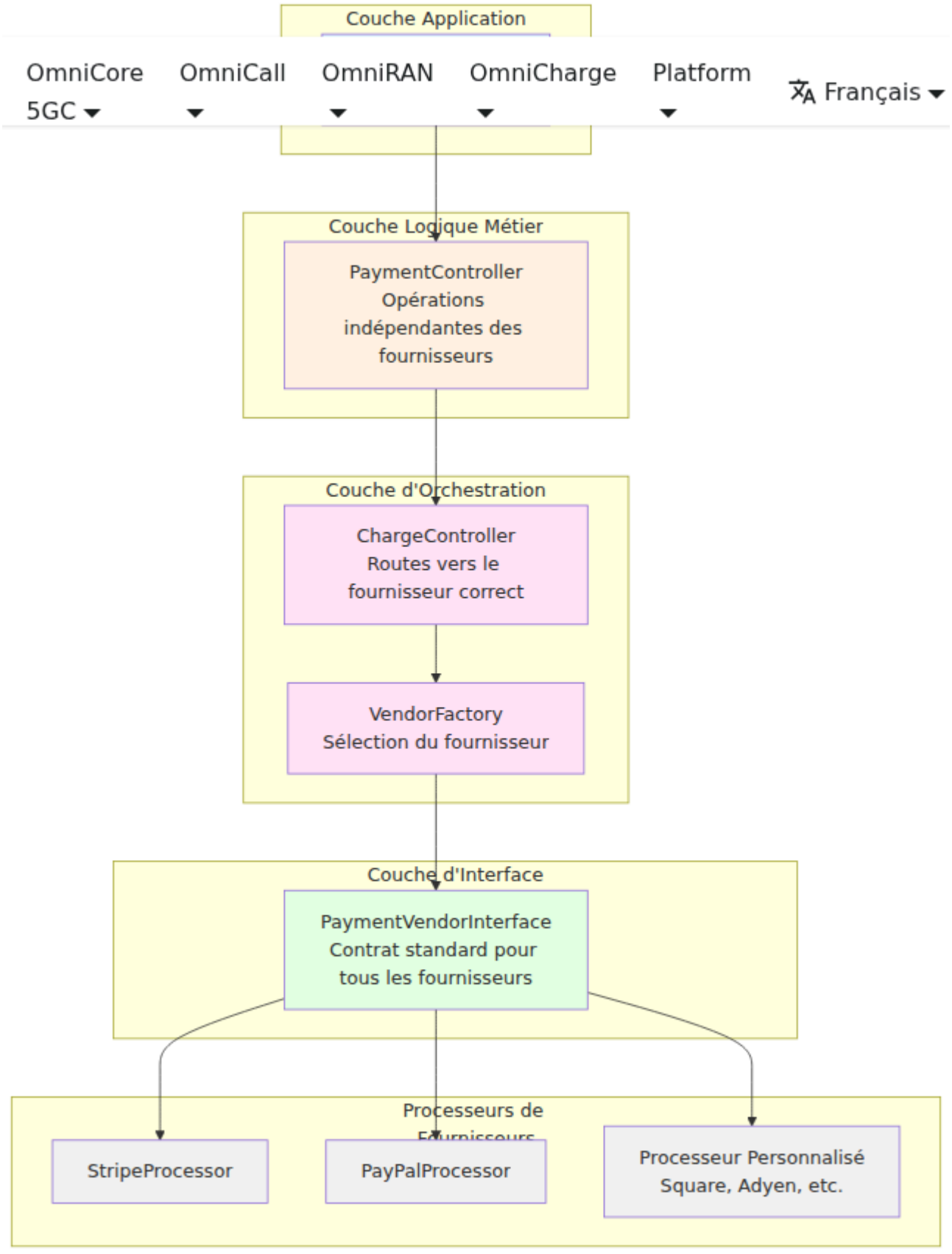
Architecture Modulaire

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
- □ Supporter plusieurs fournisseurs simultanément
- □ API cohérente indépendamment du 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 des Fournisseurs**.

Aperçu 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 APIs existantes supportent automatiquement le nouveau fournisseur

Résultat : Une fois déployé, le nouveau fournisseur fonctionne sans problème :

```
# Ajouter la 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 APIs de paiement supportent automatiquement le nouveau fournisseur sans nécessiter de modifications du code de l'application.

Temps d'Intégration : Typiquement 1 à 2 jours pour un nouveau processeur de fournisseur.

Configuration des Fournisseurs

Activation des Fournisseurs de Paiement

L'interface utilisateur contrôle quels fournisseurs de paiement sont disponibles pour les clients. Cela est défini 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 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 à partir de `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 mise 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 via compte PayPal.

1. Obtenir les Identifiants API PayPal

- Créez une application REST API 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 à partir de `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`
Mode PayPal	`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 / client ID), puis reconstruisez l'interface utilisateur et redémarrez l'API.

Conformité PCI

Comment OmniCRM Maintient la Conformité PCI :

- Les données de carte 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
- Réduction de la portée de conformité PCI 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 - Sauvegarder une carte Square :

```
{
  "vendor": "square",
  "vendor_payment_method_id": "sq_card_abc123",
  "payment_type": "card"
}
```

Pas de modifications de code. Pas de migrations. 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": "card",                // 'card', '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": "active"
}
```

PaymentAuthorization

Enregistrements d'autorisation de commit en deux phases (retenue de 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/complets.

```
{
  "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 du 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 Client**.

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",   // Généré automatiquement
: 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",      // Dernier type 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 générés automatiquement au format `INV-YYYY-NNNNNN`, séquentiellement 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)
- Supporter plusieurs paiements par facture
- Nettoyer contre les débits pour calculer le solde de la facture

Statut de la Facture :

- **PAYÉE** : Les paiements totaux (crédits) égalent ou dépassent les débits totaux
- **PARTIELLEMENT PAYÉE** : Certains paiements appliqués mais un solde reste
- **SURPAYÉE** : Pas actuellement géré - nécessite un fractionnement de crédit (fonctionnalité future)

Comptabilité en 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 **DEBITS et CREDITS** des transactions d'un client sur une période spécifiée. C'est le seul type de document où les débits et crédits sont affichés comme éléments de ligne ensemble, comme un relevé bancaire.

Notes de Crédit

But : Appliqué uniquement aux factures avec des 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 alors être :
    - Appliqué à une autre facture comme PAIEMENT, OU
    - REMBOURSÉ au client

Logique de Remboursement : Si un 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 de 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 optionnel
}
```

Réponse (201 Créé) :

```
{
  "success": true,
  "message": "Méthode de paiement ajoutée avec succès",
  "data": {
    "payment_method_id": 789,
    "customer_id": 123,
    "vendor": "stripe",
    "payment_type": "card",
    "card_brand": "visa",
    "card_last4": "4242",
    "card_exp_month": 12,
    "card_exp_year": 2025,
    "card_nickname": "Ma Carte Visa",
    "is_default": false,
    "status": "active"
  }
}
```

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": "card",
      "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": "card",
    "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 de Flux de Paiement

Le système prend en charge plusieurs flux de paiement selon 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 (le cas échéant)
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 flux de paiement en deux phases dans les playbooks Ansible, 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 : vrai)
  "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 du
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, échec de la provision, 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 le paiement pour une recharge de service qui nécessite un provisionnement (par exemple, activation de hotspot/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 Location/Hotspot Anonyme (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 une 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 : pas de
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": "customer_request"
  }'
```

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": "customer_request"           // 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 du 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
  },
  "message": "Portefeuille rechargé avec 100.00 USD"
}
```

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` (obligatoire) : 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"
    }
  ]
}
```

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 (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/tiers
POST	`/api/payments/refund`	Rembourser un paiement

Points de Terminaison du Portefeuille

M��thode	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
GET	`/api/wallet/transactions?customer_id={id}`	Obtenir l'historique des transactions

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 le 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 Hotspot

Scénario : Un utilisateur anonyme paie pour l'accès WiFi du hotspot. Aucun enregistrement client, aucune méthode de paiement sauvegardée.

```
# Paiement anonyme pour hotspot
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": "Café Downtown"
    }
  }'
```

Résultat :

- Paiement traité □
- Hotspot activé □
- Méthode de paiement NON sauvegardée □
- Aucun enregistrement client créé □

Cas d'Utilisation 4 : Réservation d'Hôtel (Retenue d'Autorisation + 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 avec 100 $ en utilisant la 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 du 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** selon vos besoins commerciaux :

1. Remboursement à la Source de Paiement (Stripe/PayPal)

Cas d'Utilisation : Le client demande un remboursement complet 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": "customer_request"
  }'
```

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 :

- □ Client a annulé la commande
- □ Service non livré
- □ Erreur de facturation
- □ 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 administratifs.

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`

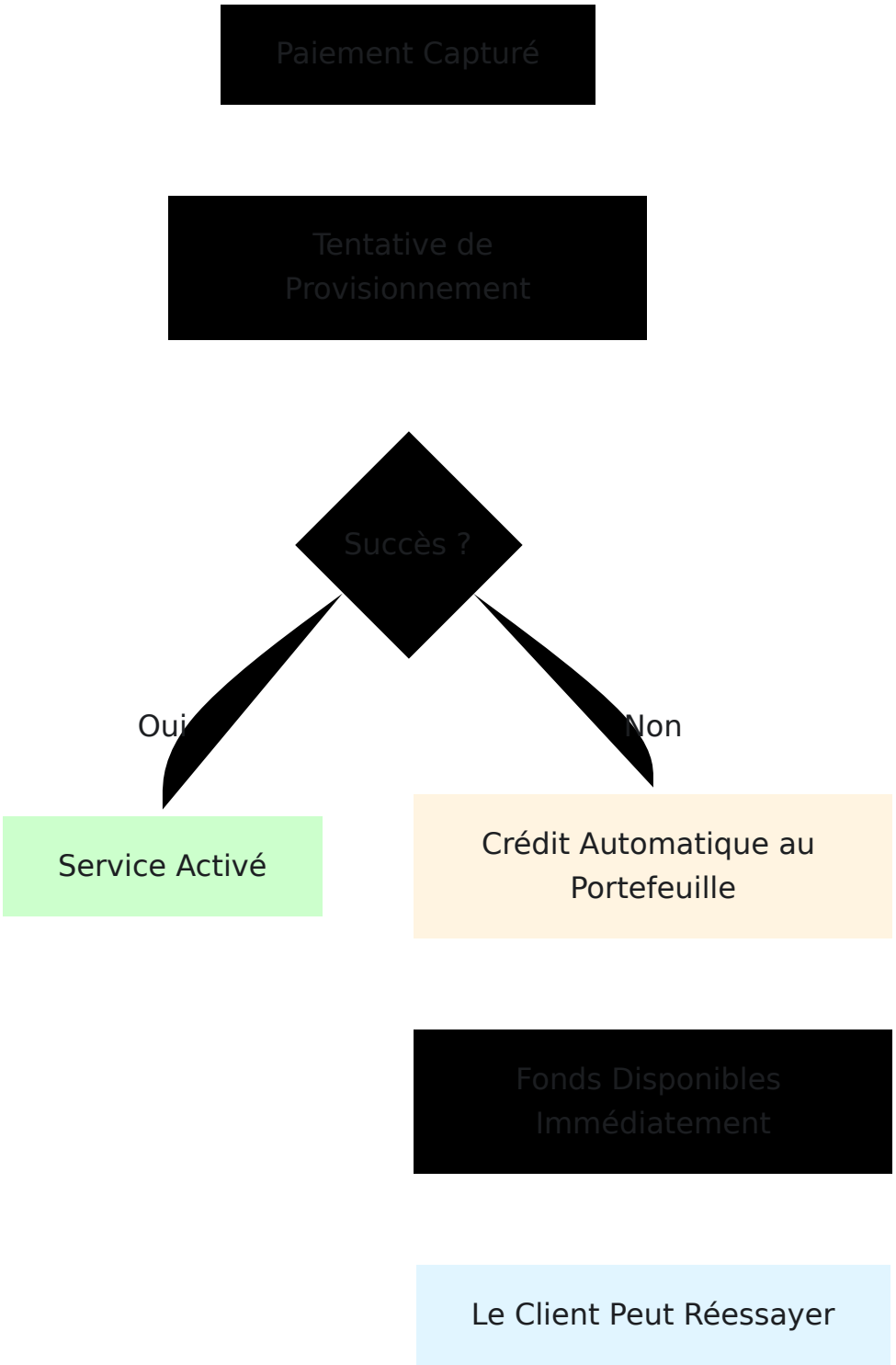
Quand Utiliser :

- □ Crédits de service (par exemple, compensation pour 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 des Erreurs :



Scénario 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 $ sur 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.

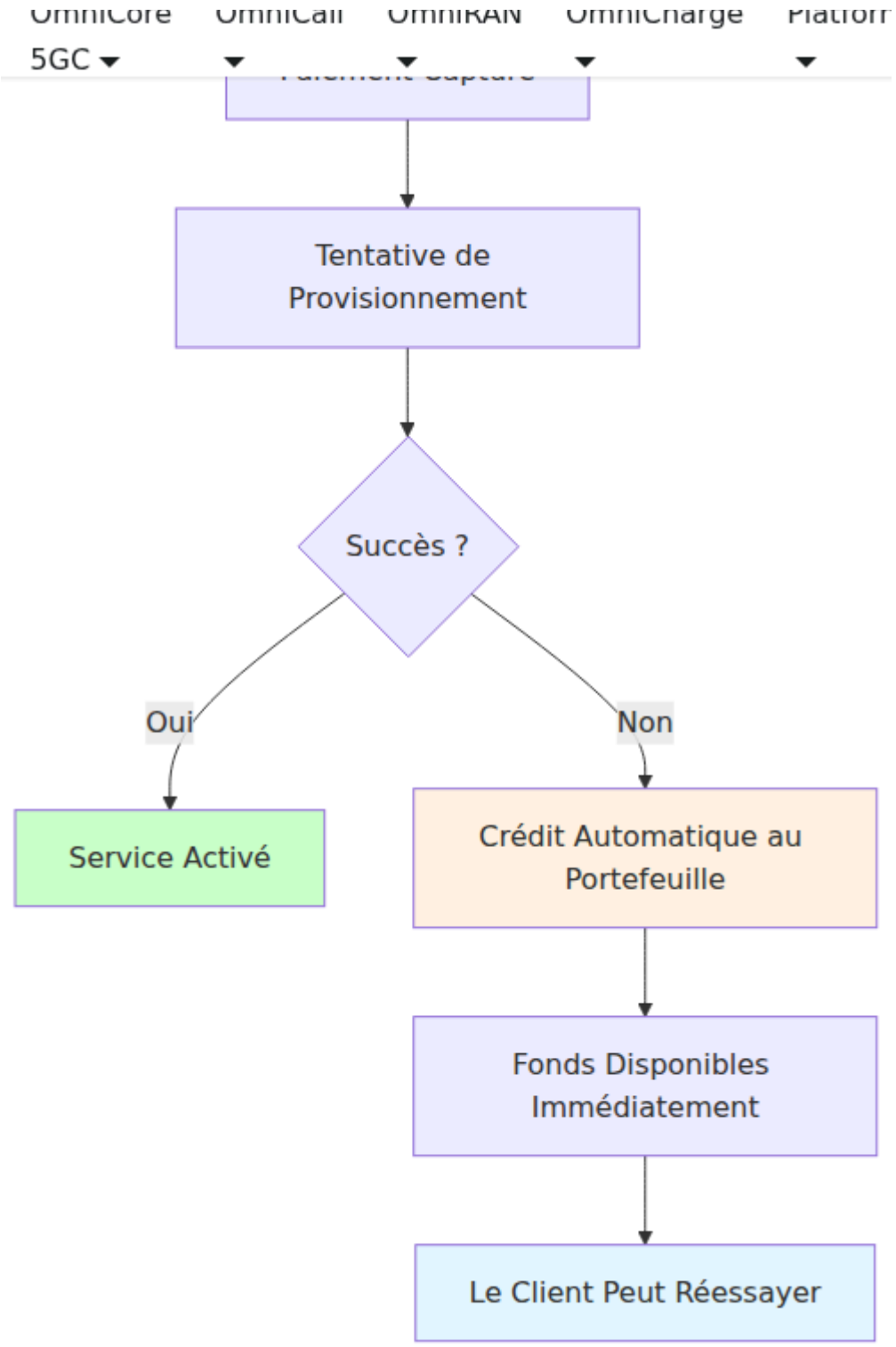
Exemple 1 : Solde de 1 $ au Portefeuille + Achat de 10 $

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 :


OmniCore 5GC
OmniCall
OmniRAN
OmniCharge
Platform
Tentative de Provisionnement
Succès ?
Oui
Non
Service Activé
Crédit Automatique 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
    }
  }'
```

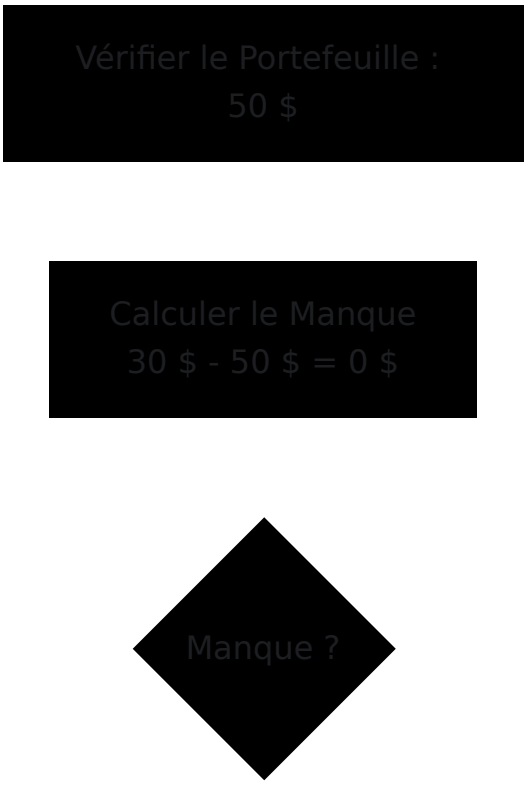
Réponse API :

```
{
  "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 facturé 9 $ de
manque
    "charged_amount": 9.00,          // ← Charge de la carte
    "wallet_credited": 9.00,         // ← Recharge du portefeuille
    "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 de l'achat est de 30 $.

Flux Prioritaire au Portefeuille :


Vérifier le Portefeuille :
50 $
Calculer le Manque
30 $ - 50 $ = 0 $
Manque ?
Non
Ignorer la Charge de
Carte
Débiter le Portefeuille :
-30 $
Solde Final : 20 $
Carte Facturée : 0 $
Utilisé : 30 $

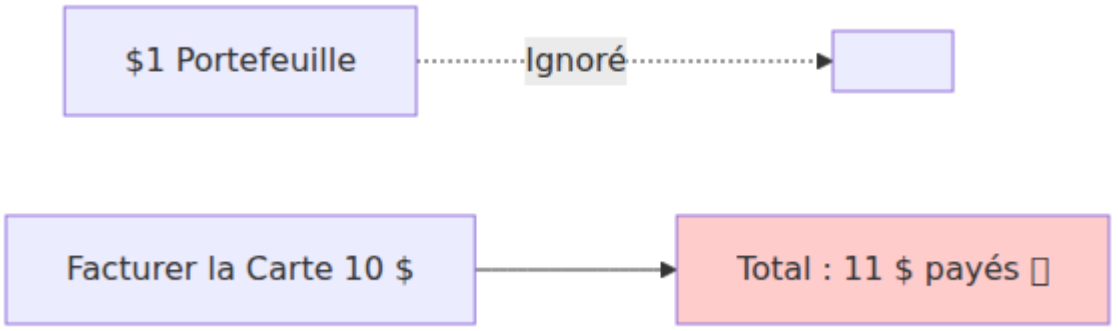
Réponse API :

```
{
  "success": true,
  "data": {
    "service_amount": 30.00,
    "initial_balance": 50.00,
    "charged_amount": 0,              // ← Pas de 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 du Portefeuille

Scénario : Hôtel retient 500 $ d'autorisation avec 150 $ de solde du 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 du
portefeuille). Le débit du portefeuille de 500 $ se produira lors
de la capture."
}
```

Étape 2 : Capture (`POST /api/payments/capture/301`) :

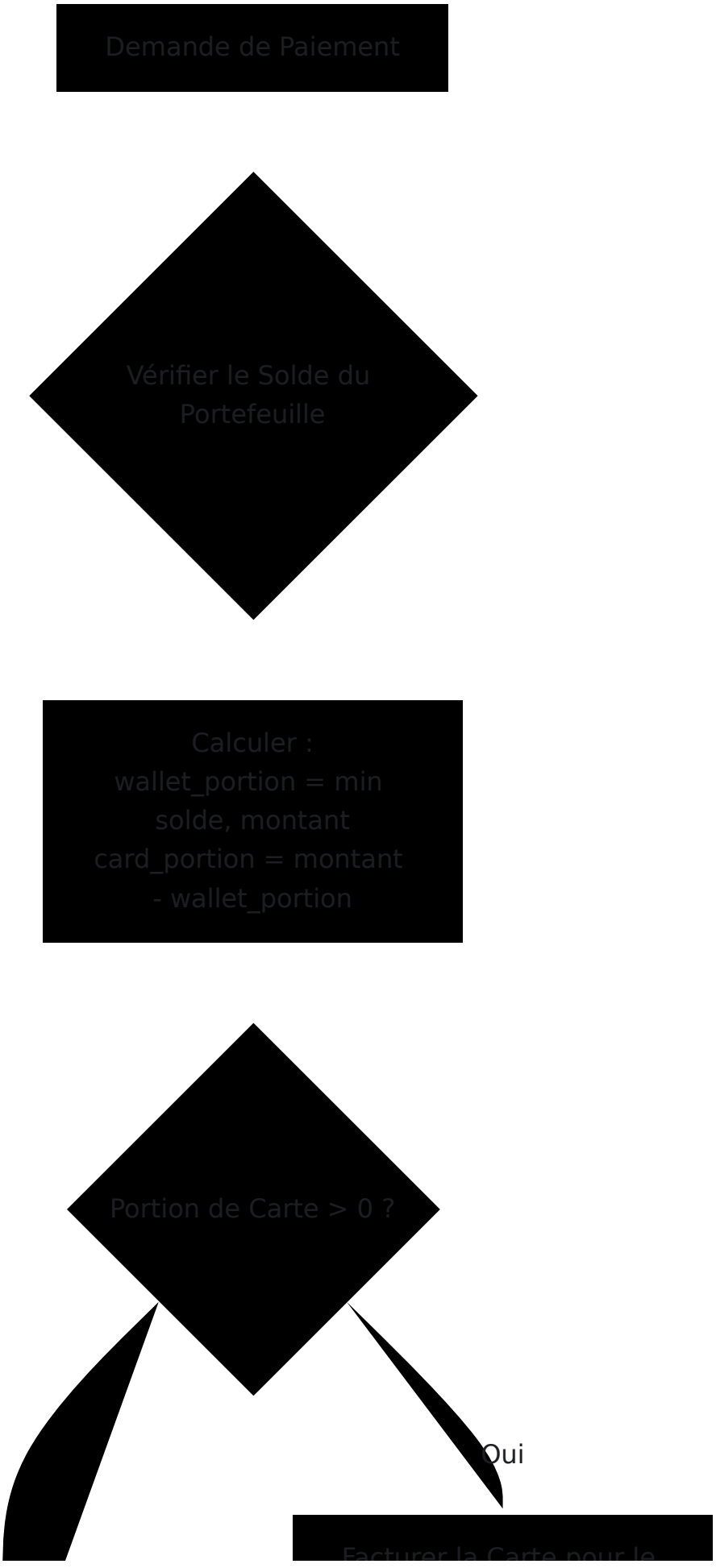


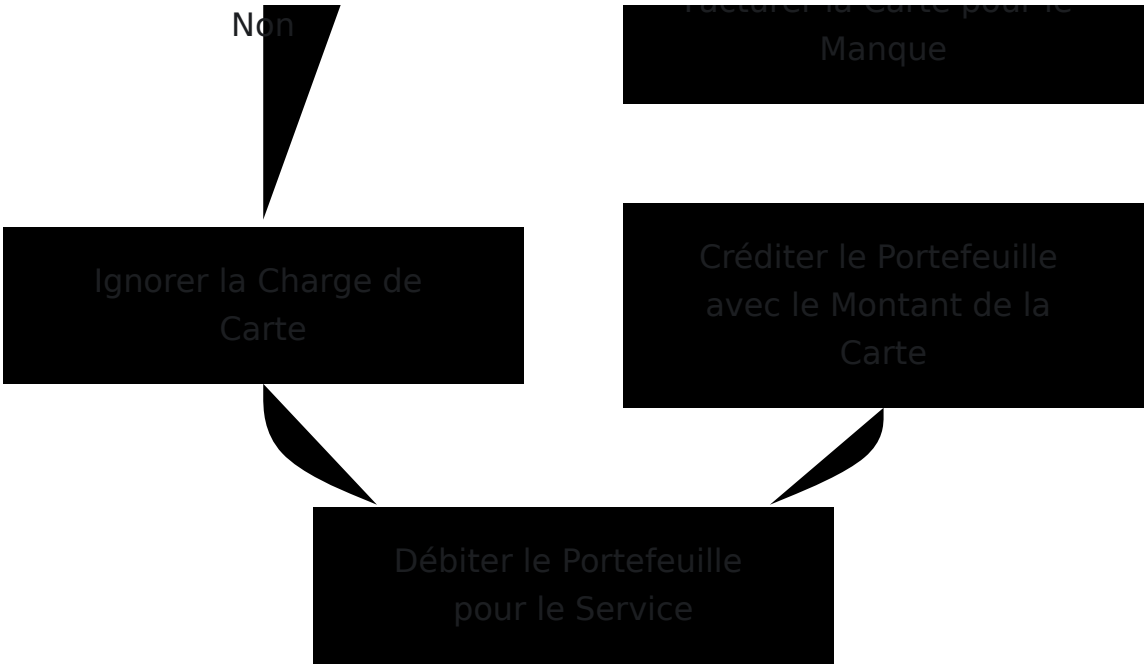
Pourquoi Cela Compte :

- □ 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 d'Implémentation

Logique de Routage Prioritaire au Portefeuille :


Demande de Paiement
Vérifier le Solde du Portefeuille
Calculer : wallet_portion = min solde, montant card_portion = montant - 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 : `montant - wallet_balance`
3. Si le manque > 0, facture la carte uniquement pour le manque
4. Crédite le portefeuille avec le paiement par carte
5. Débite le portefeuille pour le montant total du service
6. Résultat : Le client est uniquement facturé pour ce que le portefeuille ne pouvait pas couvrir

Mode de Contournement de Routage

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 véritablement anonymes (hotspots, locations)
5. **Gérez les erreurs avec grâce** et fournissez un retour d'information clair à l'utilisateur
6. **Testez les flux de paiement** dans des scénarios de succès et d'échec
7. **Surveillez les expirations d'autorisation** (généralement 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 des flux de paiement dans les playbooks Ansible :

1. **Utilisez toujours le modèle bloc/ressort** - Enveloppez le provisionnement dans un essai/catch pour un retour automatique
2. **Stockez l'authorization_id** - Sauvegardez pour les opérations de capture/libération
3. **Validez les réponses API** - Affirmez le succès avant de continuer
4. **Arrondissez les valeurs de devise** - Utilisez toujours 2 décimales
5. **Vérifiez les méthodes de paiement** - Vérifiez que le client a une méthode de paiement par défaut avant l'autorisation

Voir Meilleures Pratiques du Playbook pour des détails et exemples complets.

Authentification

Tous les points de terminaison API nécessitent une authentification via la clé API dans l'en-tête `Authorization` :

```
Authorization: Bearer YOUR_API_KEY
```

Contactez votre administrateur système pour l'approvisionnement de la clé API.

Support des Fournisseurs

Actuellement Supportés

- **Stripe** - Support complet (cartes, ACH)
- **PayPal** - Support complet (comptes PayPal, cartes via le SDK Card Fields)

Ajout de Nouveaux Fournisseurs

L'architecture modulaire rend trivial l'ajout de nouveaux fournisseurs de paiement. Voir la section Architecture Modulaire pour des détails.

Documentation Connexe

Guides d'Implémentation

- **Facturation et Paiements depuis les Playbooks** - Implémentation du playbook Ansible
  - Modèle de Commit en Deux Phases
  - Exemples d'Autorisation/Capture
  - Création de Transaction Directe
  - Calculs Pro-Rata

Guides Spécifiques aux Fonctionnalités

- **Factures Client** - Génération, modélisation et personnalisation des factures
- **Méthodes de Paiement** - Gestion des méthodes de paiement des clients, cartes et comptes PayPal
- **Transactions** - Gestion des transactions, charges manuelles et crédits
- **Traitement des Paiements** - Flux et traitement des paiements
- **Bases de Facturation** - Fondamentaux de la facturation client

Navigation Rapide

- **Ajouter des fournisseurs ?** → Voir Architecture Modulaire
- **Configurer Stripe/PayPal ?** → Voir Configuration des Fournisseurs
- **Personnaliser les factures ?** → Voir Factures Client
- **Provisionner des services ?** → Voir Intégration du Playbook

---

Factures Clients

> **Référence API** : Pour les API et workflows de paiement des factures, consultez le **Guide de l'API du Système de Paiement**

`Transactions </payments_transaction>` 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 couvre la facture, 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, lorsqu'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 <integrations_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 Factures

OmniCRM utilise des modèles HTML avec Jinja2 pour générer des factures. Vous pouvez personnaliser entièrement 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 :

- `norfone_invoice_template.html` - Modèle de facture exemple
- `cifi_invoice_template.html` - Exemple de modèle alternatif

Configuration :

Le modèle de facture actif est spécifié dans `OmniCRM-API/crm_config.yaml` :

```
invoice:
  template_filename: 'norfone_invoice_template.html'
```

Variables de Modèle Disponibles

Les modèles de factures ont accès aux variables Jinja2 suivantes :

Informations sur la Facture :

- `{{ invoice_number }}` - ID unique de la facture (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 taxes (numérique)
- `{{ total_tax }}` - Montant total des taxes calculé à partir de toutes les transactions (numérique)

Informations sur le Client :

- `{{ client.name }}` - Nom complet ou nom de l'entreprise du client
- `{{ 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 :

Parcourez 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 le Modèle Existant

```
cd OmniCRM-API/invoice_templates/
cp norfone_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 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 sur
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**

```
<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 demandes 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 chaque 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. La taxe de chaque 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 web-safe 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 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 totalisent 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 :**
    - PDF attaché à l'e-mail
    - Taille de fichier sous 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 :** Erreur de frappe dans le nom de la variable ou données manquantes
- **Solution :** Vérifiez l'orthographe exactement (sensible à la casse), v��rifiez que les données existent dans la base de données

Le style PDF est cassé :

- **Cause :** Propriété CSS non supportée
- **Solution :** Utilisez des propriétés CSS 2.1, testez avec un CSS compatible WeasyPrint

Les 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

Les tableaux débordent de la page :

- **Cause :** Largeurs de colonnes fixes trop larges
- **Solution :** Utilisez des largeurs en pourcentage, `table-layout: fixed`

Les polices ne s'affichent pas :

- **Cause :** Police non intégrée ou non disponible
- **Solution :** Utilisez des polices web-safe (Arial, Times New Roman, etc.) ou intégrez correctement des polices personnalisées

La génération de PDF échoue :

- **Cause :** Erreurs de syntaxe HTML ou plantage 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 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 la 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/retirées, 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 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 (implémenter 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 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 de la modification du modèle.

Accéder aux Factures

Les factures peuvent être consultées au niveau du système ou par client :

Vue par Client :

1. Accédez à **Clients → [Sélectionner 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 de statistiques affichent des résumés financiers.

Descriptions des Widgets :

- **Total des Factures** - Somme de tous les coûts de détail des factures (de 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 :

- Les 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 unique de la facture
- **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 des 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 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 (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 non désirées (par exemple, excluez les transactions non mobiles pour créer une facture uniquement mobile).

Facture Unique : Définissez 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, les totaux et les 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 et pourcentage de taxe (formaté comme `$10.00 (10%)`)
  - Les transactions exonérées de taxe affichent "-" dans la colonne Taxe
- **Résumé des Totaux** - Calcul en temps réel affichant :
  - 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 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 (première fois uniquement)
3. Le navigateur invite à enregistrer le fichier : `Invoice_01234.pdf`
4. Le PDF s'ouvre ou est enregistré 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

Résolution des Problèmes de Téléchargement :

- **L'indicateur de chargement 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 une 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 "Payé"
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 (Administrateurs) :

- 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 sur la 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 des 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 <integrations_mailjet>` en utilisant le modèle `api_crmCommunicationCustomerInvoice`
4. L'e-mail 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 : "E-mail de la 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 s'il n'existe pas de contact de facturation.

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 consultée en filtrant pour 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, 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 avoir été 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 le 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 par espèces, POS ou virement bancaire (inversion manuelle requise)
- Impossible de rembourser partiellement (montant total de la facture uniquement)
- Impossible de 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** - Tous, Annulé, Non Annulé

- **Statut de Paiement** - Tous, 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 de Factures

Flux de Travail 1 : Facturation Mensuelle avec Aperçu des Transactions

1. La fin du mois arrive (par exemple, 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. 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 la 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 l'e-mail de la 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 Jan
    - Date d'Échéance : 15 Fév

- 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 Jan (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 Additionnels) :**
   - Cliquez à nouveau sur **"+ Générer une Facture Proforma"**
   - Titre : "Frais Additionnels - 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 (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. Examinez 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 : Révision Rapide de la Facture et Ajustement

1. Le personnel génère une facture mensuelle
2. L'aperçu des transactions montre un total élevé inattendu
3. Examinez chaque transaction dans l'aperçu :
    - Remarquez un frais en double pour le même service
    - Cliquez sur **X** pour exclure le double
    - 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** (si présent)
9. Cliquez sur **"Générer la Facture"**
10. Envoyez immédiatement au client
11. Le client paie en ligne via Stripe

Flux de Travail 6 : Révision 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 la facture** pour ouvrir le modal des Détails de la Facture
4. Examinez 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 un PDF, cliquez sur le bouton **"Télécharger PDF"** dans le modal

7. Si le client demande un nouvel envoi 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 la facture pour voir les détails
3. Examine la liste des transactions dans le 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

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. 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 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 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 "Remboursé"
10. Le client reçoit le remboursement dans 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
- **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 dans le 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 client sont peuplés, consultez les journaux de l'API.

Échec du paiement avec erreur Stripe

- **Cause :** Carte refusée, fonds insuffisants, carte expirée ou problème avec l'API Stripe
- **Solution :** 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 :** 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 une transaction de crédit manuellement.

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
- **Solution :** Vérifiez la configuration de Mailjet dans `crm_config.yaml`, vérifiez les contacts clients, vérifiez que le modèle d'e-mail de facture existe.

Le bouton de remboursement n'apparaît pas

- **Cause :** Facture payée par espèces/POS/virement bancaire (pas Stripe), ou facture non payée
- **Solution :** Le bouton de remboursement n'apparaît que pour les paiements Stripe. Pour d'autres méthodes de paiement, créez une transaction de crédit manuelle.

Le PDF téléchargé montre l'ancien design du modèle

- **Cause :** PDF mis en cache avant la mise à jour du modèle

- **Solution :** Videz le cache des PDF de factures : `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
- **Solution :** Le client doit ajouter une carte de crédit à la page **Méthodes de Paiement** avant de payer des 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 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
- **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 facture

- **Cause :** Transaction déjà facturée ou problème de navigateur
- **Solution :** Vérifiez que la transaction s'affiche dans l'aperçu avec une coche. Rafraîchissez la page et réessayez. Videz 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 :** Examinez attentivement l'aperçu des transactions. Vérifiez chaque coût de détail de transaction et taxe. 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
- **Solution :** 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
- **Solution :** 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 null
- **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 le 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'e-mails de factures 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 de l'API de paiement et configuration des fournisseurs

Traitement des paiements

La majorité des paiements seront traités automatiquement par le système, mais il y a des moments où vous devrez 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 elle est différente de la date actuelle).

Transactions Clients

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 de configuration, 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 des `Factures <payments_invoices>`, 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** (à partir du 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 (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 : Affichage avec suffixe "k" (par exemple, $1.5k)
- Valeurs supérieures à 1 000 000 : Affichage avec suffixe "M" (par exemple, $2.3M)
- Valeurs supérieures à 1 000 000 000 : Affichage avec 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 facturé, tiret si pas encore facturé

Actions par Ligne :

Chaque ligne a un menu d'actions (⋮) avec les options :

- **Voir Détails** - Ouvre le modal de détail de la transaction
- **Télécharger PDF de 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 de Débit (Frais)

Montants positifs qui augmentent le solde dû par le client :

- **Frais de Configuration 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 de 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 le modal d'ajout de transaction.

Transaction de Débit (Frais) :

Transaction de 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 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 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 sur :

- 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 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 (croissant ↔ décroissant).

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 des transactions par défaut
- Exclue de la génération de factures
- Peut être consultée en filtrant pour les transactions "Annulées"
- Déduite du total des "Transactions Non Facturées"

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 de Débit (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 de Crédit (Paiements/Remboursements)** - Aucune taxe n'est appliquée aux crédits
  - Le champ de pourcentage de taxe est caché 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 peuvent pas être modifiées

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 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 comme crédit

Flux de Travail 2 : Frais d'Installation Manuel

1. Le technicien de terrain installe le service
2. Le personnel doit facturer £75 de frais d'installation
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 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 d'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. Les transactions dans la plage sont 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.

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 aux attentes

- **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

Chargement et Paiements depuis les Playbooks

Ce guide explique comment mettre en œuvre le chargement et le traitement des paiements au sein des playbooks Ansible pour les flux de travail de provisioning d'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, consultez le **Guide de l'API du Système de Paiement**

Vue d'ensemble

Les Playbooks Ansible d'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 et une capture de paiement 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 provisioning réussit, avec un rollback automatique si une étape échoue.

Flexibilité des Tarifs : Playbooks en tant que 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 indépendamment de ce qui est dans 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 une tarification 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 de groupe 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 wholesale_cost au lieu de
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 %}
```

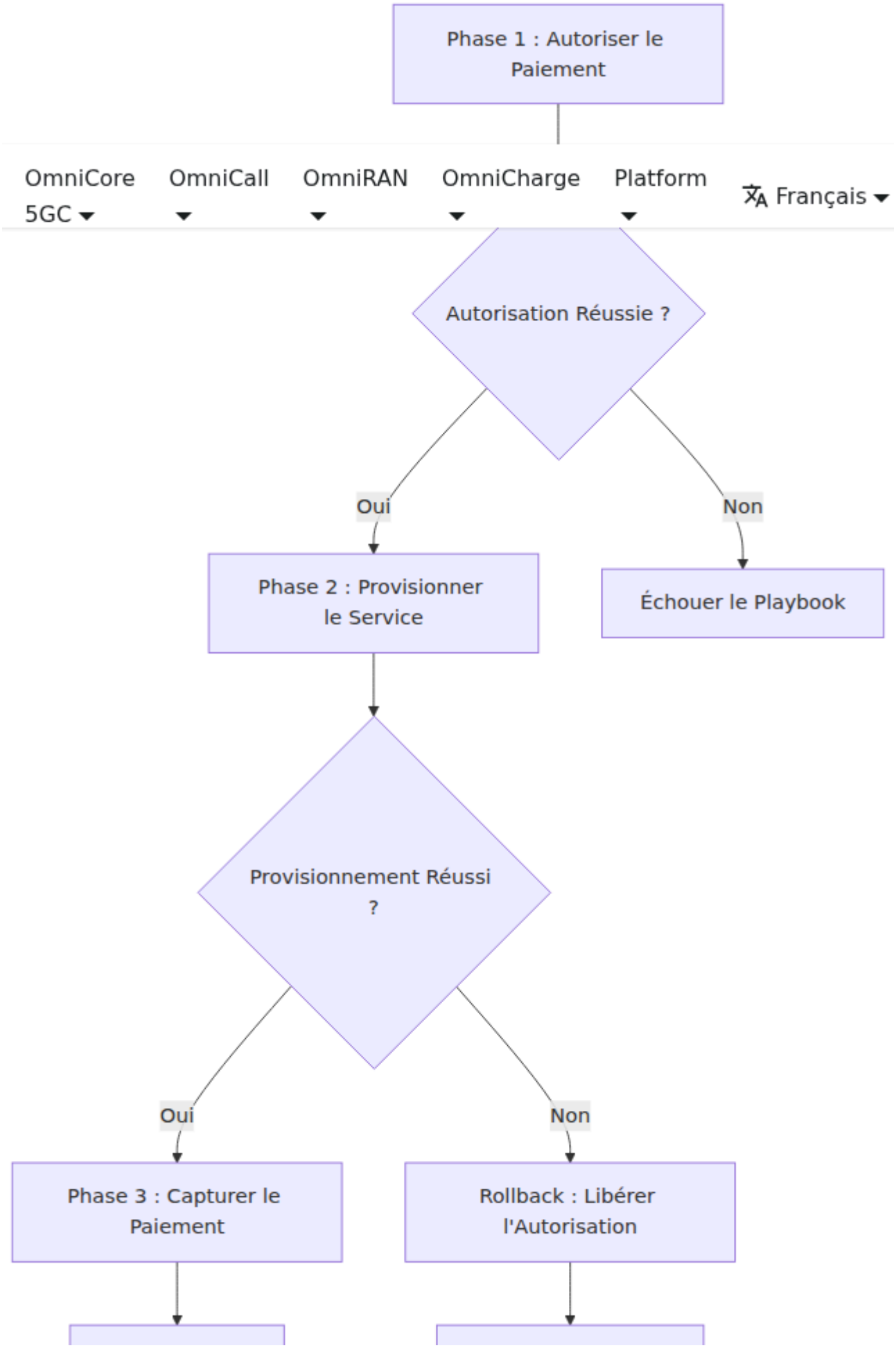
Le point clé est que **vous n'êtes pas enfermé dans un modèle de tarification**. Le playbook vous donne un contrôle programmatique total pour mettre en œuvre toutes les règles commerciales requises par votre

organisation. 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 provisioning. Cela garantit que le client n'est facturé que si le provisioning 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 d'Implémentation

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 provisioning réussi
Rollback - Libérer l'autorisation si le provisioning échoue

Exemple Complet

Voici un exemple complet provenant de `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 provisioning 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 le rollback
          fail:
            msg: "Le provisioning OCS a échoué. L'autorisation de
paiement {{ authorization_id }} a été annulée. Client non
facturé."
```

Points de Terminaison de l'API de Paiement

> **Routage Wallet-First** : Le système de paiement optimise automatiquement les charges par carte en utilisant d'abord le solde du portefeuille. Si un client a 1 $ dans son portefeuille et achète un addon de 10 $, seuls 9 $ sont facturés à sa carte. Consultez la section Routage Wallet-First 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 Wallet-First : L'autorisation calcule automatiquement le manque à gagner après avoir vérifié le solde du portefeuille. Si le solde du portefeuille est de 150 $ et que l'autorisation est de 500 $, seuls 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** (obligatoire) - ID du client à facturer
- **amount** (obligatoire) - Montant à autoriser au format décimal
- **currency** (optionnel) - Code de la devise (par défaut à la valeur par défaut du système)
- **payment_method_id** (obligatoire) - Moyen de paiement à facturer

- **metadata** (optionnel) - Données supplémentaires pour la création de transaction :
  - **description** - Description de la transaction
  - **service_id** - Service pour lequel on facture
  - **site_id** - Site associé au service
  - **product_id** - Produit en cours de provisioning
  - **user_id** - Utilisateur initiant la charge (optionnel)
  - **title** - Titre de la transaction
  - **wholesale_cost** - Votre coût (pour le suivi de 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 le `authorization_id` pour une utilisation dans les appels de capture ou de libération

- Notez que le `card_amount` montre seulement que le manque à gagner 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 du manque à gagner (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 s'annulent)
    - 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 $

Consultez Exemples de Routage Wallet-First 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 cela dans des scénarios de secours/rollback lorsque le provisioning 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 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 wallet-first, 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 des 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 provenant de `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** (obligatoire) - ID du client
- **service_id** (optionnel) - ID du service pour lier la transaction
- **title** (obligatoire) - 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** (obligatoire) - 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 provisioning de service
- Charges d'installation
- Crédits ou ajustements manuels
- Charges d'équipement
- Frais administratifs

Remarque : La création directe de transaction ne traite PAS le paiement - elle crée simplement 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 à mi-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 d'Implémentation

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 :

```yaml
- 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 le nombre de 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. Utilisez Toujours le Modèle Bloc/Rescue

Enveloppez la capture de paiement et le provisioning dans un bloc/rescue pour garantir le rollback :

```
- block:
    # Tâches de provisioning
    - 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 provisioning a échoué, le client n'a pas été
facturé"
```

2. Validez 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. Stockez 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. Utilisez les Métadonnées Efficacement

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. Arrondissez 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. Gérez les Méthodes de Paiement Manquantes

Vérifiez si le client a un moyen de paiement par défaut avant d'essayer l'autorisation :

```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 Courants

Modèle 1 : Provisionnement de Service Payant

Pour les services qui nécessitent un paiement immédiat :

1. Obtenir le moyen de paiement du client
2. Autoriser le paiement
3. Provisionner le service dans OCS/CGRateS
4. Créer un enregistrement de service dans le CRM
5. Capturer le paiement
6. En cas d'échec : Libérer l'autorisation et 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. Provisionner le service

2. Créer un enregistrement de service
3. Ajouter la transaction de frais d'installation directement (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. Obtenir des informations sur le service
2. Exécuter l'action CGRateS
3. Mettre à jour les dates de service
4. Aucun paiement ou création de transaction nécessaire

Modèle 4 : Charges Récurrentes via ActionPlan

Pour des charges récurrentes automatiques :

1. Créer une Action avec `*http_post` vers le point de terminaison de provisioning
2. Créer un ActionPlan avec un timing `*monthly`
3. Assigner 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

L'Autorisation Échoue

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.

La Capture Échoue Après un Provisionnement Réussi

Symptôme : Service provisionné mais la capture de paiement échoue

Problème : C'est un état de défaillance critique - le service est actif mais le client n'est pas facturé

Solution :

- L'autorisation peut avoir expiré (typiquement 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 :

- Définissez `invoice: true` dans les métadonnées d'autorisation, OU
- Créez manuellement la 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)
- Toujours arrondir 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 vers la Source 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 vers le Portefeuille - Solde immédiat pour de futurs achats (géré automatiquement pour les scénarios d'erreur)

Pour les échecs de provisioning, 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 provisioning

Documentation du Système de Paiement

- **Guide de l'API du Système de Paiement** - Référence API complète, routage de portefeuille, remboursements
  - APIs des Méthodes de Paiement
  - APIs des Flux de Paiement
  - Routage Wallet-First
  - 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 client/locataire 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 Administrateurs - 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 (optionnel)
- **Nom de famille** - Nom de famille de l'utilisateur (obligatoire)
- **Numéro de téléphone** - Numéro de téléphone de contact (optionnel)
- **Rôle** - Sélectionner un ou plusieurs rôles à attribuer (obligatoire)
- **Contact Client** - Lier optionnellement l'utilisateur à un enregistrement de contact client (pour les utilisateurs clients)

4. **Cliquer sur "Créer Utilisateur"** - L'utilisateur est créé et peut immédiatement se connecter avec les identifiants fournis
5. **L'utilisateur reçoit une notification** - Envoyer optionnellement 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 2FA pour tout le personnel administratif et de support
- Lier les utilisateurs clients à leur enregistrement de contact client pour un bon encadrement des données

Via l'API

Créer un utilisateur de manière programmatique :

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 Plusieurs Rôles :

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 le but 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 client"
}
```

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 auxquels il est attribué. 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'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 seul rôle 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 en fonction du 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 jeton (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 répondre aux exigences de complexité des mots 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 jeton de réinitialisation et envoie un email.

2. **Réinitialiser avec le jeton :**

   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` (optionnel) - 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 jetons de réinitialisation expirent après 1 heure
- Les tentatives de connexion échouées peuvent déclencher un verrouillage de 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 2FA"**
4. Le système génère :
   - Secret TOTP (QR code affiché)
   - 10 codes de sauvegarde (usage unique)
5. L'utilisateur scanne le QR code avec une 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 valides, 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
- Usage unique seulement (consommés après utilisation)
- Utilisés si l'application d'authentification est indisponible
- 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 des 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 2FA par l'Administrateur

Si un utilisateur perd l'accès à son application d'authentification et à tous ses 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 2FA"**

4. Confirmer la réinitialisation
5. La 2FA est désactivée ; l'utilisateur peut se connecter uniquement avec son 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 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 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 modifiables :
    - Prénom, deuxième prénom, nom de famille
    - Adresse email (nécessite vérification)
    - Numéro de téléphone
    - Rôles
    - Lien de 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 des 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 plus 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]}`

Restaurer 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 : Ceci est irréversible et supprime toutes les données utilisateur de la base de données.

Non exposé via l'interface utilisateur. 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 des 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** - Réattribuer les tickets ouverts et les tâches de l'utilisateur avant de supprimer
- **Vérifier 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
- **Informer les équipes concernées** - Informer les gestionnaires/collègues de la suppression

Catalogue de 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, rôles et permissions ; accéder à tous les points de terminaison protégés).
- `can_impersonate` — Agir temporairement en tant qu'autre utilisateur (audité ; pour le support/le dépannage).
- `grafana_access` — Accès aux tableaux de bord d'analytique Grafana pour créer et visualiser des rapports et visualisations personnalisés. Voir la section Analytique Grafana ci-dessous pour plus de détails.

Clients et Enregistrements Associés

- **Client**
  - view_customer, create_customer, update_customer, delete_customer
  - **Portée propre :** *voir son propre client*
- **Site Client**

    - view_customer_site, create_customer_site, update_customer_site, delete_customer_site
    - **Portée propre :** *voir son propre site client*
- **Contact Client**
    - view_customer_contact, create_customer_contact, update_customer_contact, delete_customer_contact
    - **Portée propre :** *voir son propre contact client*
- **Attribut Client** (voir `Attributs Client <administration_attributes>`)
    - view_customer_attribute, create_customer_attribute, update_customer_attribute, delete_customer_attribute
    - **Portée propre :** *voir son propre attribut client*
- **Tag Client** (voir `Tags Client <administration_tags>`)
    - view_customer_tag, create_customer_tag, update_customer_tag, delete_customer_tag
    - **Portée propre :** *voir son propre tag client*
- **Service Client**
    - view_customer_service, create_customer_service, update_customer_service, delete_customer_service
    - **Portée propre :** *voir son propre service client*
- **Activité Client**
    - view_customer_activity, create_customer_activity, update_customer_activity, delete_customer_activity
    - **Portée propre :** *voir sa propre activité client*

Facturation

- **Carte Stripe**
    - `view_customer_stripe_card`, `create_customer_stripe_card`, `update_customer_stripe_card`, `delete_customer_stripe_card`
    - **Portée propre :** *voir sa propre carte stripe client*
- **Transactions**
    - `view_customer_transaction`, `create_customer_transaction`, `update_customer_transaction`, `delete_customer_transaction`
    - **Portée propre :** *voir sa propre transaction client*

- **Factures**
  - `view_customer_invoice`, `create_customer_invoice`, `update_customer_invoice`, `delete_customer_invoice`
  - **Portée propre :** *voir sa propre facture client*

Communications

- view_communication, create_communication, update_communication, delete_communication
- **Portée propre :** *voir sa propre communication*

Inventaire et Modèles

- **Inventaire**
  - `view_inventory`, `create_inventory`, `update_inventory`, `delete_inventory`
  - **Portée propre :** *voir son propre inventaire*
- **Modèle d'Inventaire**
  - `view_inventory_template`, `create_inventory_template`, `update_inventory_template`, `delete_inventory_template`
  - **Portée propre :** *voir son propre modèle d'inventaire*

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

- **Provisionner**
  - `view_provision`, `create_provision`, `update_provision`, `delete_provision`

    - **Portée propre :** *voir son propre provisionnement*
- **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, lorsque 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	Notes
Administrateur Système	`admin`, éventuellement `can_impersonate` ; plus CRUD large 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` ; `create_customer_invoice` optionnel	Lecture intensive ; écriture limitée
Administrateur Client (locataire)	"voir ses propres ..." à travers clients, sites, contacts, services, inventaire, factures, transactions, communications, provisionnement	Gestion à portée de locatair
Auditeur en Lecture Seule	Large `view_*` uniquement	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`

Commencer 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.** Commencez avec des rôles minimaux ; ajoutez des permissions au besoin.
- **Préférer "voir ses propres ...".** Utilisez des permissions à portée de locataire pour les rôles orientés client.
- **Maintenir les rôles stables.** Mettez à jour les permissions des rôles lorsque les fonctionnalités changent—ne modifiez pas chaque utilisateur.
- **Auditer régulièrement.** Vérifiez 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 de 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 OmniCRM. Cela vous permet de visualiser vos métriques les plus importantes sans quitter le CRM, tout en maintenant la navigation 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 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 ��tre 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 Autonomes

Le Portail de Soins Autonomes 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 nécessiter l'assistance du personnel du service client.

Méthodes d'Accès :

- Connexion directe via les identifiants du client
- `Impersonation du personnel <customer_care>` pour le dépannage (de Clients → Contacts → "Se connecter en tant qu'utilisateur")

Voir aussi : `Service Client <customer_care>` pour les détails sur l'imitation, `Flux d'Authentification <authentication_flows>` pour le processus de connexion.

Objectif

Le Portail de Soins Autonomes fournit aux clients :

1. **Gestion de Compte** - Voir 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** - Voir 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 de compte à tout moment sans appeler le support

Aperçu du Portail

Lorsque les clients se connectent au Portail de Soins Autonomes, 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** - Acheter des crédits de données, de voix, de SMS ou 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 une vérification)
- Numéro de téléphone
- Mot de passe
- Préférences de notification

Informations en Lecture Seule :

- ID client
- Date de création du compte
- Type de client (Individuel/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 de Service :

- ✅ **Actif** - Le service est opérationnel
- ⚠️ **Expirant Bientôt** - 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 solde 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 Progrès** - Représentation visuelle de la consommation
- **Informations sur l'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 dans l'ordre de poids :

- **Poids 10** - Consommé en premier (typiquement des données promotionnelles/bonus)
- **Poids 20** - Consommé en second (typiquement des données de recharge)
- **Poids 30** - Consommé en dernier (typiquement 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 On-Net - Appels au sein du même réseau
- Appels Off-Net - Appels vers d'autres réseaux
- Appels Internationaux - Appels en dehors du pays
- Numéros Premium - Numéros à tarif spécial

Utilisation des SMS

Surveiller les allocations de messages SMS et la consommation.

Affichage de l'Utilisation des SMS :

- **Messages Restants** - Solde SMS
- **Utilisés Ce Mois** - Messages envoyés
- **MMS Inclus** - Si les MMS comptent pour 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 voir des factures, des transactions et gérer des méthodes de paiement.

Factures

Voir et payer les 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

Voir 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 <payment_methods>` pour la documentation détaillée sur la gestion des paiements.

Recharge / Recharge

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 personnalisées)
4. Examiner 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 & Recharge <features_topup_recharge>` pour des flux de travail de recharge détaillés.

Options Supplémentaires de Service

Acheter des fonctionnalités et des améliorations supplémentaires pour les services existants.

Options Supplémentaires 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 supplémentaires disponibles pour ce type de service
3. Cliquer sur **"Ajouter au Service"** sur l'option désirée
4. Examiner le coût (unique + récurrent)
5. Confirmer l'achat
6. Option supplémentaire provisionnée automatiquement

Voir `Modification des Services - Options Supplémentaires <csa_modify>` 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 reçue
- Avertissements d'expiration de service (7 jours, 3 jours, 1 jour)
- Alertes de solde bas (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 basses (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 l'Auto-Service

Certaines opérations nécessitent l'assistance du personnel :

Nécessite le Service Client :

- Changer le type de client (Individuel ↔ 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 client (isolation 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 des options supplémentaires/recharges)

Accès à l'Impersonation du Personnel

Le personnel de support peut accéder au Portail de Soins Autonomes 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 Autonomes du client
6. Toutes les actions sont enregistrées et attribuées à l'utilisateur impersonné
7. Le personnel voit exactement ce que voit le client

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
- Délai d'expiration de session après inactivité
- Ne peut pas changer le mot de passe du client pendant l'imitation

Voir `Service Client - Impersonation d'Utilisateur <customer_care>` pour la documentation complète sur l'imitation.

Réactivité Mobile

Le Portail de Soins Autonomes est entièrement réactif et optimisé pour les appareils mobiles.

Fonctionnalités Mobiles :

- Navigation optimisée pour le tactile
- Dispositions simplifiées pour les petits écrans
- Gestes de balayage pour la navigation dans le carrousel
- Formulaires et entrées adaptés aux mobiles
- Scan de code QR pour la 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
- Visualisation hors ligne des données récentes
- Notifications push (si activées)
- Chargement rapide avec des travailleurs 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 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 de dépassement
6. Sauvegarder les codes de secours dans un endroit sûr (si 2FA activé)
7. Se déconnecter après avoir utilisé des ordinateurs partagés/publics

Gestion de l'Utilisation :

1. Activer les alertes de solde bas (10-20 % restantes)
2. Recharger avant que les soldes n'expirent pour éviter l'interruption du service
3. Examiner 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 l'auto-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 API <concepts_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 l'auto-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 rapidement 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 retrouver l'accès ?

R : Utilisez les codes de secours pour vous connecter, puis désactivez et réactivez 2FA. Si vous n'avez pas de codes de secours, contactez le support pour une réinitialisation de 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 direct de facture. Sinon, la connexion est requise pour des raisons de sécurité.

Q : Comment puis-je télécharger toutes mes factures en une seule fois ?

R : Actuellement, cela nécessite de 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 <customer_care>` - Accès au dépannage du personnel
- `Méthodes de Paiement <payment_methods>` - Gestion des cartes et des paiements
- `Recharge & Recharge <features_topup_recharge>` - Achat de crédit supplémentaire
- `Utilisation des Services <csa_service_usage>` - Comprendre le suivi des soldes
- `Aperçu de la Facturation <billing_overview>` - Concepts de facturation et factures
- `Flux d'Authentification <authentication_flows>` - Connexion et sécurité
- `2FA <2fa>` - Configuration de l'authentification à deux facteurs
- `Modifications de Service <csa_modify>` - Ajout de fonctionnalités et d'options supplémentaires