Autenticação de Dois Fatores (2FA)

Segurança Aprimorada com Senhas de Uso Único Baseadas em Tempo

OmniCRM suporta **autenticação de dois fatores (2FA)** usando senhas de uso único baseadas em tempo (TOTP). Isso adiciona uma camada extra de segurança ao exigir que os usuários forneçam tanto sua senha quanto um código sensível ao tempo de um aplicativo autenticador.

Veja também: `RBAC <rbac>` para permissões de gerenciamento de 2FA, `Authentication Flows <authentication_flows>` para detalhes do processo de login.

Propósito

2FA fornece:

1. **Segurança Aprimorada** — Protege contas mesmo que as senhas sejam comprometidas.
2. **Conformidade** — Atende aos requisitos de segurança para indústrias regulamentadas.
3. **Escolha do Usuário** — Opcional para usuários, pode ser imposto por função ou globalmente.
4. **Padrão da Indústria** — Usa o protocolo TOTP compatível com Google Authenticator, Authy, Microsoft Authenticator e outros aplicativos padrão.

Como 2FA Funciona

Quando 2FA é ativado para um usuário:

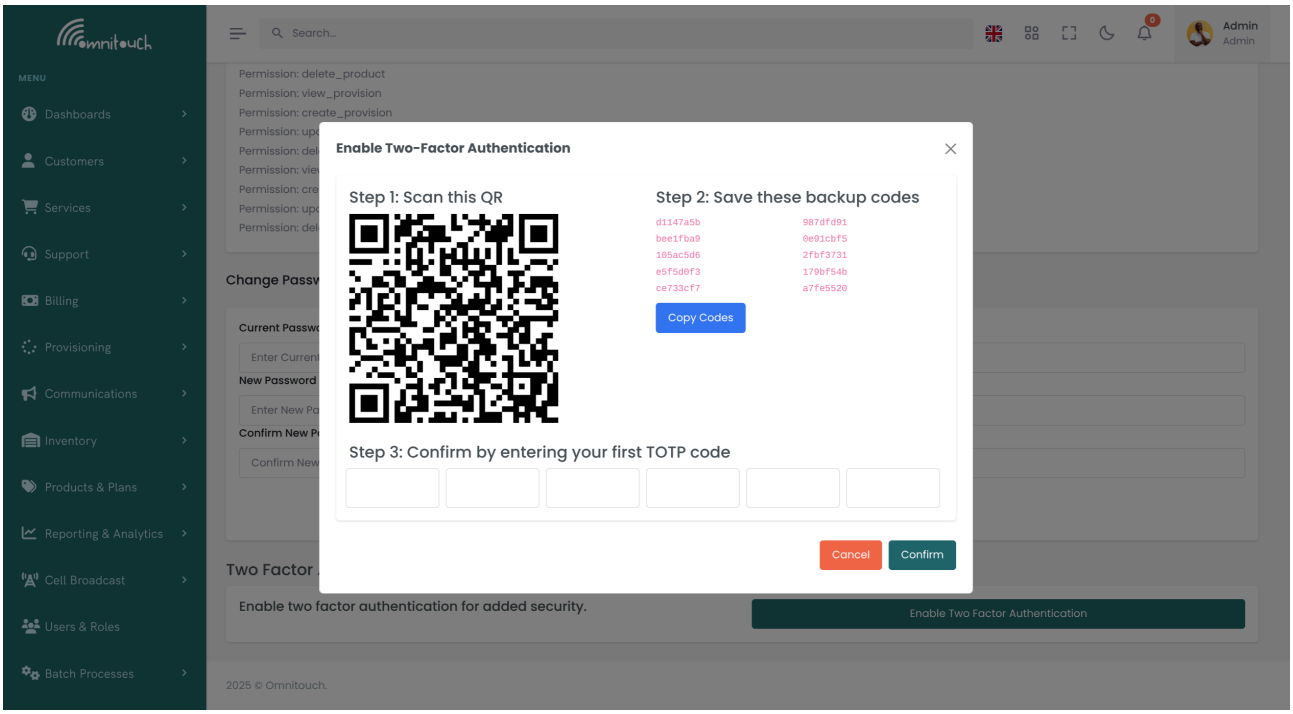
1. **Configuração** — O usuário escaneia um código QR com seu aplicativo autenticador durante o registro.
2. **Login** — Após inserir nome de usuário/senha, o usuário fornece o código de 6 dígitos de seu aplicativo.
3. **Verificação** — O sistema valida se o código baseado em tempo corresponde ao valor esperado.
4. **Acesso** — O usuário ganha acesso somente após ambos os fatores serem verificados.

Habilitando 2FA

Para Usuários Individuais

Os usuários podem habilitar 2FA para sua própria conta:

1. Navegue até **Configurações do Usuário** ou **Perfil**
2. Selecione **Habilitar Autenticação de Dois Fatores**
3. Escaneie o código QR com um aplicativo autenticador
4. Insira o código de verificação para confirmar a configuração
5. Salve os códigos de backup em um local seguro



Backup & Recuperação

Códigos de Backup

Ao habilitar 2FA, os usuários recebem códigos de backup que podem ser usados se seu dispositivo estiver indisponível:

- Cada código é de uso único
- Armazene os códigos de forma segura offline
- Gere novos códigos se todos forem usados

Redefinição do Admin

Se um usuário perder o acesso ao seu autenticador e todos os códigos de backup forem esgotados, um administrador com acesso ao banco de dados pode redefinir manualmente o 2FA do usuário limpando o campo totp_secret no banco de dados. O usuário pode então reabilitar o 2FA.

Integração de API

Habilitar 2FA para um usuário

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

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

A resposta inclui a URI de provisionamento (para código QR) e códigos de backup.

Verificar configuração de 2FA

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

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

Verificar 2FA durante o login

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

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

Retorna o token de acesso, token de atualização e dados do usuário após a verificação bem-sucedida.

Regenerar códigos de backup

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

Requer autenticação. Retorna um novo conjunto de códigos de backup.

Melhores Práticas

- **Códigos de backup primeiro.** Sempre salve os códigos de backup antes de concluir a configuração do 2FA.
- **Eduque os usuários.** Forneça instruções claras para configuração e recuperação.
- **Processo de redefinição seguro.** Verifique a identidade do usuário antes de redefinir manualmente o 2FA no banco de dados.

FAQ

Quais aplicativos autenticadores são suportados? Qualquer aplicativo compatível com TOTP (Google Authenticator, Authy, Microsoft Authenticator, 1Password, etc.).

E se eu perder meu telefone? Use um código de backup ou entre em contato com um administrador para redefinir o 2FA.

Posso usar SMS em vez de um aplicativo? Atualmente, apenas aplicativos autenticadores TOTP são suportados.

O 2FA é obrigatório? Depende da política da sua organização. O 2FA é tipicamente exigido para funções administrativas e de suporte, mas é opcional para usuários clientes. O sistema não impõe 2FA para contas de clientes (usuários com função de cliente não veem prompts de registro de 2FA).

Qual é a validade dos códigos TOTP? Os códigos se renovam a cada 30 segundos e têm uma pequena janela de tempo para validação (normalmente aceita códigos da janela atual de 30 segundos mais janelas anteriores/próximas para tolerância de desvio de relógio).

Gerenciamento de Chaves de API

A interface de Gerenciamento de Chaves de API fornece uma **UI baseada na web** para criar, monitorar e gerenciar chaves de API usadas para acesso programático à API do OmniCRM.

Nota

Para conceitos gerais de autenticação de API e exemplos de uso, consulte `concepts_api`.

Visão Geral

As chaves de API permitem **autenticação segura e de longa duração** para:

- Integrações de servidor para servidor
- Scripts de automação
- Aplicativos de terceiros
- Tarefas agendadas e jobs cron
- Sistemas de monitoramento externos

Ao contrário dos tokens JWT (que expiram após minutos/horas), as chaves de API permanecem válidas até serem revogadas manualmente ou até a data de expiração.

Acessando o Gerenciamento de Chaves de API

Navegue para:

Ou diretamente:

Permissão Necessária: `MANAGE_API_KEYS` (papel de administrador)

Visualização da Lista de Chaves de API

A página principal exibe todas as chaves de API em formato de tabela:

Colunas:

- **Nome** - Rótulo descritivo para a chave de API (por exemplo, "Sistema de Provisionamento", "Ferramenta de Monitoramento")
- **Criado Por** - Nome de usuário da pessoa que criou a chave
- **Chave de API** - A string da chave real (parcialmente mascarada por segurança)
- **Status** - Ativa, Expirada ou Revogada
- **Data de Criação** - Quando a chave foi gerada
- **Data de Expiração** - Quando a chave expirará automaticamente
- **Ações** - Botões Editar, Excluir, Regenerar

Exemplo de Exibição:

Widgets do Painel

No topo da página, estatísticas resumidas são exibidas:

- **Total de Chaves de API** - Contagem de todas as chaves de API (ativas e inativas)
- **Chaves Ativas** - Chaves atualmente válidas
- **Expirando em Breve** - Chaves que expiram nos próximos 30 dias
- **Chaves Expiradas** - Chaves que passaram da data de expiração

Criando uma Chave de API

Passo 1: Clique em "Adicionar Chave de API"

Clique no botão **+ Adicionar** no canto superior direito da lista de Chaves de API.

Passo 2: Preencha os Detalhes

Um formulário modal aparece solicitando:

> Nome: _____
> (por exemplo, "Sistema de Provisionamento")
>
> Descrição: _____
> (Opcional - finalidade desta chave)
>
> Data de Expiração: [Seletor de Data]
> (Opcional - deixe em branco para sem expiração)
>
> Permissões: ☐ Visualizar Clientes ☐ Criar Clientes ☐ Visualizar Serviços ☐ Criar Serviços ☐ Provisionamento ☐ Visualizar Inventário ☑ Admin (todas as permissões)
>
> [Cancelar] [Gerar Chave]

Diretrizes para os Campos:

Nome (obrigatório)

- Identificador curto e descritivo
- Exemplos: "Sistema de Provisionamento", "Integração de Faturamento", "Monitoramento"
- Usado em logs de auditoria e exibido na lista

Descrição (opcional)

- Explicação mais detalhada
- Exemplos: "Usado pelo servidor de provisionamento Ansible", "Sincronização de faturamento de terceiros"
- Ajuda futuros administradores a entenderem a finalidade da chave

Data de Expiração (opcional)

- Se em branco: A chave nunca expira (não recomendado)
- Se definida: A chave se torna automaticamente inválida após esta data
- Recomendado: Definir expiração por segurança (90 dias a 1 ano)

Permissões

- Selecione permissões específicas ou marque "Admin" para acesso total
- Segue o mesmo modelo de permissão baseado em função que as contas de usuário
- **Melhor Prática:** Atribuir as permissões mínimas necessárias

Passo 3: Gerar e Copiar a Chave

Após clicar em **"Gerar Chave"**, o sistema exibe a nova chave de API criada:

> ⚠ Copie esta chave agora - ela não será mostrada novamente!
>
> sk_live_a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6q7r8s9t0
>
> [Copiar para a Área de Transferência]
>
> [Fechar]

Aviso

Salve a chave de API imediatamente!

Uma vez que você feche este diálogo, a chave completa não poderá ser recuperada novamente. Você só verá uma versão mascarada (`sk_live_...XYZ`) na visualização da lista.

Se você perder a chave, deverá **regenerá-la**, o que invalida a chave antiga e pode quebrar integrações existentes.

Passo 4: Configure Seu Aplicativo

Use a chave de API nas solicitações do seu aplicativo:

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

Ou em variáveis de ambiente:

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

Gerenciando Chaves Existentes

Visualizando Detalhes da Chave

Clique em qualquer nome de chave de API para ver os detalhes completos:

- Nome completo da chave e descrição
- Timestamp de criação
- Nome de usuário do criador
- Permissões associadas
- Estatísticas de uso (se implementadas)
- Logs de acesso recentes

Regenerando uma Chave de API

Se uma chave de API for comprometida ou perdida, regenere-a:

1. Clique nos **⋮ (três pontos)** na coluna Ações
2. Selecione **"Regenerar Chave"**
3. Confirme a ação

Aviso

Regenerar invalida a chave antiga imediatamente.

Quaisquer aplicativos que utilizem a chave antiga deixarão de funcionar. Atualize todas as integrações com a nova chave antes de regenerar.

O Que Acontece:

- A chave antiga é revogada
- Uma nova chave com as mesmas permissões é gerada
- A nova chave é exibida (copie-a imediatamente)
- Nome, descrição e permissões permanecem inalterados

Revogando (Excluindo) uma Chave de API

Para remover permanentemente uma chave de API:

1. Clique nos ⋮ **(três pontos)** na coluna Ações
2. Selecione **"Excluir"**
3. Confirme a exclusão

O Que Acontece:

- A chave é imediatamente revogada
- Todas as solicitações usando esta chave retornam `401 Unauthorized`
- A chave é removida do banco de dados
- **Não pode ser desfeita** - a chave não pode ser recuperada

Quando Revogar:

- A integração não é mais necessária
- A chave foi comprometida
- O sistema que usa a chave foi desativado
- Substituindo por uma nova chave com permissões diferentes

Editando Detalhes da Chave de API

Para modificar os detalhes de uma chave de API:

1. Clique nos ⋮ **(três pontos)** na coluna Ações
2. Selecione **"Editar"**
3. Atualize nome, descrição, expiração ou permissões
4. Clique em **"Salvar Alterações"**

Campos Editáveis:

- Nome
- Descrição
- Data de expiração
- Permissões

Não Editável:

- O valor da chave em si (use Regenerar para mudar)
- Data de criação
- Criado por usuário

Status da Chave de API

As chaves de API podem ter vários status:

Ativa

- A chave é válida e pode ser usada
- Dentro da data de expiração (ou sem expiração definida)
- Não revogada manualmente
- Exibida com um distintivo verde

Expirando em Breve

- Ativa, mas expirará nos próximos 30 dias
- Exibida com um distintivo laranja/aviso
- Considere rotacionar antes da expiração

Expirada

- Passou da data de expiração
- Não aceita mais autenticação
- Exibida com um distintivo vermelho
- Pode ser excluída ou a expiração estendida

Revogada

- Excluída/desativada manualmente
- Permanentemente inválida
- Não é mais exibida na lista ativa

Filtragem e Pesquisa

A lista de Chaves de API suporta:

Pesquisa:

Pesquise por nome, descrição ou chave parcial:

Filtrar por Status:

Filtro suspenso para mostrar:

- Todas as Chaves
- Apenas Ativas
- Expirando em Breve (próximos 30 dias)
- Expiradas

Ordenar:

Clique nos cabeçalhos das colunas para ordenar por:

- Nome
- Data de Criação
- Data de Expiração
- Criado Por

Melhores Práticas de Segurança

Geração de Chaves de API

- **Comprimento:** As chaves devem ter pelo menos 32 caracteres (o sistema impõe isso)
- **Aleatoriedade:** Geradas usando geradores de números aleatórios criptograficamente seguros
- **Formato:** Normalmente prefixadas (por exemplo, `sk_live_`) para identificação

Armazenamento de Chaves de API

No CRM:

- As chaves são hash antes do armazenamento (como senhas)
- A chave completa é mostrada apenas uma vez durante a criação
- O banco de dados armazena o hash para verificação
- Mesmo administradores não podem recuperar a chave completa depois

No Seu Aplicativo:

- Armazene em variáveis de ambiente, não no código
- Use sistemas de gerenciamento de segredos (AWS Secrets Manager, HashiCorp Vault)
- Nunca comite chaves no controle de versão
- Rotacione chaves periodicamente (90-365 dias)

Gerenciamento de Permissões

- **Princípio do Menor Privilégio** - Conceda apenas as permissões necessárias
- Evite criar chaves de administrador, a menos que absolutamente necessário
- Use chaves separadas para diferentes sistemas/finalidades
- Revise permissões regularmente

Monitoramento e Auditoria

- Monitore o uso da chave de API por meio de logs de atividade
- Configure alertas para padrões de acesso incomuns
- Revise timestamps de "último uso" regularmente
- Remova chaves não utilizadas

Rotação de Chaves

Estabeleça uma política de rotação de chaves:

1. **Crie uma nova chave** com as mesmas permissões
2. **Atualize os aplicativos** para usar a nova chave
3. **Monitore** para garantir que a chave antiga não seja mais usada
4. **Revogue a chave antiga** após o período de carência

Solução de Problemas

"401 Unauthorized" ao usar a chave de API

- **Causa:** Chave inválida, expirada ou incorreta
- **Solução:**
  - Verifique se a chave foi copiada corretamente (sem espaços extras)
  - Verifique o status da chave (Ativa vs. Expirada)
  - Confirme se a chave tem as permissões necessárias
  - Certifique-se de usar o cabeçalho `X-API-KEY` (não `Authorization`)

"Chave de API não encontrada" após a criação

- **Causa:** A chave pode ter sido criada, mas não armazenada corretamente
- **Solução:**
  - Verifique a lista de chaves de API para a nova entrada
  - Se estiver faltando, crie uma nova chave
  - Relate o problema ao administrador

Chave de API expirando em breve

- **Causa:** Data de expiração se aproximando (dentro de 30 dias)
- **Solução:**
  - Crie uma nova chave com expiração estendida
  - Atualize os aplicativos para usar a nova chave
  - Revogue a chave antiga após a migração

Não é possível excluir a chave de API

- **Causa:** Pode estar protegida ou em uso
- **Solução:**
  - Certifique-se de que você tem permissões de administrador
  - Verifique se a chave está bloqueada/protegida
  - Entre em contato com o administrador se o problema persistir

Endpoints da API (para Gerenciamento Programático)

As chaves de API também podem ser gerenciadas via API (requer permissões de administrador):

Listar Chaves de API

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

Criar Chave de API

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

{
  "name": "Nova Integração",
  "description": "Sincronização de faturamento de terceiros",
  "expiry_date": "2026-01-10",
  "permissions": ["view_customer", "view_service"]
}
```

Resposta:

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

Revogar Chave de API

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

Atualizar Chave de API

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

{
  "name": "Nome Atualizado",
  "expiry_date": "2026-12-31"
}
```

Casos de Uso Comuns

Caso de Uso 1: Integração do Sistema de Provisionamento

Crie uma chave de API para seu servidor de provisionamento Ansible:

1. Navegue para Chaves de API → Adicionar
2. Nome: "Servidor de Provisionamento Ansible"
3. Descrição: "Usado pela automação de provisionamento"
4. Permissões: Provisionamento, Visualizar/Criar Serviços, Visualizar/Atualizar Inventário
5. Expiração: 1 ano
6. Copie a chave e adicione ao `crm_config.yaml` do Ansible

Caso de Uso 2: Integração de Faturamento de Terceiros

Crie uma chave somente leitura para exportação de faturamento:

1. Nome: "Sincronização de Faturamento - QuickBooks"
2. Permissões: Visualizar Clientes, Visualizar Transações, Visualizar Faturas
3. Expiração: 90 dias (rotacione trimestralmente)
4. Use no script de exportação agendada

Caso de Uso 3: Monitoramento e Alerta

Crie uma chave para coleta de métricas do Prometheus/Grafana:

1. Nome: "Monitoramento - Grafana"
2. Permissões: Visualizar Serviços, Visualizar Provisionamento
3. Expiração: Nunca (monitoramento precisa de acesso contínuo)
4. Configure na fonte de dados do Grafana

Caso de Uso 4: API do Portal do Cliente

Crie uma chave para o portal de autoatendimento do cliente:

1. Nome: "Backend do Portal do Cliente"

2. Permissões: Visualizar Próprio Cliente, Visualizar Próprios Serviços, Criar Pagamentos
3. Expiração: 180 dias
4. Use nas chamadas da API do backend do portal

Documentação Relacionada

- `concepts_api` - Conceitos e exemplos de autenticação de API
- `rbac` - Controle de acesso baseado em função e permissões
- `2fa` - Autenticação de dois fatores para segurança adicional

Atributos do Cliente

Atributos do Cliente são pares de chave-valor flexíveis que podem ser anexados a qualquer registro de cliente para armazenar metadados personalizados, configurações de configuração ou informações específicas de negócios que não se encaixam nos campos padrão do cliente.

Para categorização visual de clientes e links clicáveis, veja `Customer Tags <administration_tags>`. Para informações básicas sobre clientes, veja `Customers, Contacts, Sites & Services <basics_customers>`.

Ao contrário dos campos de banco de dados fixos, os atributos permitem que você estenda dinamicamente os registros de clientes sem modificar o esquema do banco de dados. Isso os torna ideais para armazenar dados específicos de implantação, parâmetros de integração ou sinalizadores de lógica de negócios personalizados.

Propósito e Casos de Uso

Casos de uso comuns para Atributos do Cliente incluem:

1. Dados de Integração

Armazene identificadores de sistemas externos ou chaves de API específicas para este cliente:

- `external_crm_id` = "SF-12345" (ID do cliente Salesforce)
- `legacy_system_id` = "OLD-CRM-789" (Referência de migração)
- `hubspot_contact_id` = "12345678" (Integração HubSpot)

2. Lógica de Negócios Personalizada

Armazene sinalizadores ou configurações que controlam o comportamento específico do cliente:

- `billing_method` = "quarterly" (Substituir a cobrança mensal padrão)

- `auto_provision` = "true" (Habilitar provisionamento automático de serviços)
- `support_tier` = "premium" (Nível de suporte personalizado)
- `credit_limit` = "10000" (Limite de crédito específico do cliente)

3. Dados de Conformidade e Regulatórios

Rastreie metadados relacionados à conformidade:

- `gdpr_consent_date` = "2025-01-01" (Consentimento para processamento de dados)
- `tax_exempt` = "true" (Status de isenção fiscal)
- `regulatory_entity` = "FCC-123456" (Identificador regulatório)

4. Metadados Operacionais

Armazene informações operacionais:

- `preferred_contact_method` = "email" (Preferência de comunicação)
- `account_manager` = "<john.smith@company.com>" (Gerente de conta designado)
- `onboarding_date` = "2025-01-15" (Rastreamento do ciclo de vida do cliente)
- `churn_risk_score` = "0.23" (Análise preditiva)

5. Parâmetros de Provisionamento

Armazene configurações específicas de provisionamento:

- `radius_username_format` = "email" (Formato RADIUS personalizado)
- `vlan_id` = "100" (Configuração de rede)
- `ipv6_enabled` = "true" (Sinalizadores de recurso)

Atributos vs. Campos Padrão

Use Atributos Quando:

- Os dados são específicos da implantação ou variam por instalação
- Os requisitos mudam com frequência
- Armazenar metadados específicos de integração
- Prototipar novos recursos antes de adicionar campos ao banco de dados
- Os dados não precisam de consultas complexas ou junções

Use Campos Padrão Quando:

- Os dados são essenciais para o modelo de cliente (nome, email, endereço)
- Pesquisa, filtragem ou relatórios frequentes são necessários
- Os dados têm restrições de integridade referencial
- O desempenho é crítico para consultas em grande escala

Gerenciando Atributos via a UI

Visualizando Atributos do Cliente

Para visualizar atributos de um cliente:

1. Navegue até a página de visão geral do cliente
2. Clique na aba **Attributes**
3. Você verá uma tabela de todos os atributos para este cliente, mostrando:
    - Nome do Atributo (chave)
    - Valor do Atributo
    - Data de Criação
    - Data da Última Modificação

Criando um Novo Atributo

Para criar um novo atributo para um cliente:

1. Navegue até a página de visão geral do cliente
2. Clique na aba **Attributes**
3. Clique no botão **Add Attribute**
4. Preencha os campos obrigatórios:

- **Nome do Atributo** (obrigatório): A chave/nome para este atributo (por exemplo, `external_crm_id`)
- **Valor do Atributo** (obrigatório): O valor a ser armazenado (por exemplo, `SF-12345`)

5. Clique em **Create Attribute**

Convenções de Nomenclatura:

- Use letras minúsculas com sublinhados: `external_system_id` ✓
- Evite espaços: `external system id` ✗
- Mantenha os nomes descritivos, mas concisos
- Use nomenclatura consistente entre clientes para os mesmos tipos de atributos

Editando um Atributo

Para editar um atributo existente:

1. Navegue até a página de visão geral do cliente
2. Clique na aba **Attributes**
3. Encontre o atributo que você deseja editar na tabela
4. Clique no botão **Edit** (lápis)
5. Modifique o nome ou valor do atributo
6. Clique em **Update Attribute**

Nota

Alterar o nome de um atributo cria um novo par de chave-valor. Certifique-se de que isso não quebre integrações que dependem do nome original do atributo.

Deletando um Atributo

Para deletar um atributo:

1. Navegue até a página de visão geral do cliente
2. Clique na aba **Attributes**

3. Encontre o atributo que você deseja deletar na tabela
4. Clique no botão **Delete** (lixeira)
5. Confirme a exclusão na janela pop-up

Aviso

Deletar atributos usados por integrações, fluxos de trabalho de provisionamento ou lógica de cobrança pode causar falhas. Verifique as dependências antes da exclusão.

Referência de Campo de Atributo

Integração API

Os atributos podem ser gerenciados programaticamente via API:

Criar ou Atualizar um Atributo

Endpoint: `PUT /crm/attribute/`

Permissão Necessária: `create_customer_attribute`

Corpo da Solicitação:

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

Resposta:

```
{
  "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"
}
```


Atualizar um Atributo Existente

Endpoint: `PATCH /crm/attribute/attribute_id/{attribute_id}`

Permissão Necessária: `update_customer_attribute`

Corpo da Solicitação:


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


Obter Atributo por ID

Endpoint: `GET /crm/attribute/attribute_id/{attribute_id}`

Permissão Necessária: `view_customer_attribute`

Resposta:


```
{
  "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"
}
```

Obter Todos os Atributos por ID do Cliente

Endpoint: `GET /crm/attribute/customer_id/{customer_id}`

Permissão Necessária: `view_customer_attribute`

Resposta:


```
[
  {
    "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"
  }
]
```


Deletar um Atributo

Endpoint: `DELETE /crm/attribute/attribute_id/{attribute_id}`

Permissão Necessária: `delete_customer_attribute`

Resposta:


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

Operações em Lote de Atributos

Gerenciando Múltiplos Atributos

Para definir múltiplos atributos para um cliente de uma vez (por exemplo, durante o onboarding ou sincronização de integração):

```
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"}
    )
```

Consultando Clientes por Atributo

Embora os atributos não tenham endpoints de pesquisa integrados, você pode filtrar clientes por atributo usando a API de pesquisa de clientes com filtragem personalizada:

```
# Obtenha todos os clientes e, em seguida, filtre por atributo no cód
aplicativo
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()

    # Encontre clientes com atributo específico
    for attr in attributes:
        if attr['attribute_name'] == 'support_tier' and
attr['attribute_value'] == 'premium':
            print(f"Cliente premium: {customer['customer_name']}")
```

Nota

Para consultas frequentes baseadas em atributos, considere adicionar campos de banco de dados indexados ou implementar um endpoint de pesquisa dedicado.

Melhores Práticas

1. Convenções de Nomenclatura

- Use snake_case: `external_system_id` ✓
- Seja descritivo: `billing_method` ✓ vs `method` ✗
- Evite palavras-chave reservadas ou caracteres especiais
- Documente os significados dos atributos em seu guia de implantação

2. Tipos de Dados

- Atributos armazenam valores como strings (máx. 150 caracteres)
- Para booleanos, use "true"/"false" (minúsculas)
- Para datas, use o formato ISO 8601: "2025-01-04 10:30:00"
- Para grandes dados JSON, considere campos de banco de dados dedicados em vez disso

3. Validação

- Valide os valores dos atributos no código do aplicativo antes de salvar
- Use formatos de valor consistentes entre clientes
- Documente os valores esperados para cada nome de atributo

4. Documentação

- Mantenha um registro dos nomes e propósitos dos atributos
- Documente quais sistemas/integrações dependem de atributos específicos
- Inclua exemplos de valores válidos

5. Migração e Limpeza

- Audite regularmente atributos não utilizados
- Remova atributos obsoletos após migrações de sistema
- Versione os nomes dos atributos ao alterar esquemas (por exemplo, `api_key_v2`)

Exemplos de Fluxos de Trabalho

Integração de Onboarding

Ao migrar clientes de um sistema legado:

```
# Armazene a referência do sistema legado para depuração
PUT /crm/attribute/
{
  "customer_id": 123,
  "attribute_name": "legacy_crm_id",
  "attribute_value": "OLD-12345"
}

# Rastreie a data de migração
PUT /crm/attribute/
{
  "customer_id": 123,
  "attribute_name": "migrated_date",
  "attribute_value": "2025-01-04"
}
```

Regras de Cobrança Personalizadas

Substitua o ciclo de cobrança padrão para um cliente específico:

```
# Defina a cobrança trimestral
PUT /crm/attribute/
{
  "customer_id": 123,
  "attribute_name": "billing_cycle",
  "attribute_value": "quarterly"
}

# Em seguida, no código de cobrança, verifique o atributo antes de
processar
attributes = GET /crm/attribute/customer_id/123
billing_cycle = next(
    (a['attribute_value'] for a in attributes if
a['attribute_name'] == 'billing_cycle'),
    'monthly'  # padrão
)
```

Sinalizadores de Recursos

Habilite recursos beta para clientes específicos:

```
# Habilite o provisionamento IPv6
PUT /crm/attribute/
{
  "customer_id": 123,
  "attribute_name": "feature_ipv6_enabled",
  "attribute_value": "true"
}
```

Permissões

As operações de atributo requerem as seguintes permissões:

- `view_customer_attribute` - Visualizar atributos
- `create_customer_attribute` - Criar novos atributos
- `update_customer_attribute` - Modificar atributos existentes
- `delete_customer_attribute` - Remover atributos

Veja `rbac` para configuração de controle de acesso baseado em função.

Solução de Problemas

Atributo Não Aparecendo na UI

- Verifique se o atributo foi criado (verifique a resposta da API)
- Atualize a página para recarregar os dados do cliente
- Verifique se o usuário tem a permissão `view_customer_attribute`

Não é Possível Atualizar Atributo

- Certifique-se de que você tem a permissão `update_customer_attribute`
- Verifique se o attribute_id está correto
- Verifique se o atributo pertence ao cliente especificado

Integração Falhando Após Exclusão de Atributo

- Restaure o atributo com o valor anterior
- Atualize o código de integração para lidar com atributos ausentes de forma adequada
- Audite as dependências de atributos antes da exclusão

Valor do Atributo Truncado

- Os valores dos atributos têm um limite de 150 caracteres
- Para dados mais longos, divida em múltiplos atributos ou use o campo de notas do cliente
- Considere armazenar grandes dados em campos de banco de dados dedicados

Guia de Configuração e Personalização

Este guia abrangente cobre todas as opções de configuração e personalização para o OmniCRM, incluindo configurações de backend, branding de frontend, personalização visual e melhores práticas de implantação.

Visão Geral

O OmniCRM utiliza dois sistemas principais de configuração:

Configuração de Backend:

- **Arquivo:** `OmniCRM-API/crm_config.yaml`
- **Formato:** YAML
- **Requer:** Reinício da API após alterações
- **Usado para:** Banco de dados, integrações, segurança, provisionamento

Configuração de Frontend:

- **Arquivo:** `OmniCRM-UI/.env`
- **Formato:** Variáveis de ambiente
- **Requer:** Recompilação da UI após alterações
- **Usado para:** Branding, recursos, serviços externos

Configuração de Backend (crm_config.yaml)

O arquivo `crm_config.yaml` contém todas as configurações do sistema de backend.

Localização: `/OmniCRM/OmniCRM-API/crm_config.yaml`

Configuração do Banco de Dados

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

Campos:

- `username` - Nome de usuário do banco de dados MySQL
- `password` - Senha do banco de dados MySQL
- `server` - Nome do host ou IP do servidor de banco de dados (padrão: localhost)

Conexão com o Banco de Dados:

- O nome do banco de dados é codificado como `omnicrm`
- Porta padrão: 3306 (padrão do MySQL)
- String de conexão: `mysql+pymysql://username:password@server/omnicrm`

Nota de Segurança: Nunca comite este arquivo com credenciais reais no controle de versão. Use configuração específica para o ambiente ou gerenciamento de segredos.

Nota: Isso deve corresponder às suas credenciais de banco de dados no `.env`. Em implantações em contêiner, o servidor é tipicamente `db` (nome do serviço Docker).

Tipos de Serviço

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

Propósito: Define tipos de serviço válidos para sua implantação.

Estes são usados em todo o sistema para:

- Categorização de produtos
- Filtragem de addons (addons correspondem a tipos de serviço)
- Fluxos de trabalho de provisionamento
- Relatórios e análises

Adicione tipos de serviço personalizados aqui para seus casos de uso específicos.

Integração HSS (Home Subscriber Server)

Para operadores de rede móvel com integração HSS:

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

Configuração:

- `hss_peers` - Lista de endpoints HSS para provisionamento de assinantes
- `apn_list` - Lista de IDs de APN disponíveis para provisionamento, separada por vírgulas

Usado para: Provisionamento de rede móvel e autenticação de assinantes.

Configuração do Modelo de Email Mailjet

O OmniCRM usa o Mailjet para emails transacionais. Cada tipo de email tem sua própria configuração de modelo:

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

  # Email de Boas-Vindas ao Cliente
  api_crmCommunicationCustomerWelcome:
    from_email: "support@yourcompany.com"
    from_name: "Suporte da Sua Empresa"
    template_id: 5977509
    subject: "Bem-vindo à SuaEmpresa"

  # Email de Fatura do Cliente
  api_crmCommunicationCustomerInvoice:
    from_email: "billing@yourcompany.com"
    from_name: "Cobrança da Sua Empresa"
    template_id: 6759851
    subject: "Sua Fatura - "

  # Lembrete de Fatura
  api_crmCommunicationCustomerInvoiceReminder:
    from_email: "billing@yourcompany.com"
    from_name: "Cobrança da Sua Empresa"
    template_id: 5977570
    subject: "Lembrete de Pagamento de Fatura"

  # Email de Boas-Vindas ao Usuário (Funcionário/Admin)
  api_crmCommunicationUserWelcome:
    from_email: "admin@yourcompany.com"
    from_name: "Admin da Sua Empresa"
    template_id: 6118112
    subject: "Bem-vindo à Equipe"

  # Solicitação de Redefinição de Senha
  api_crmCommunicationUserPasswordReset:
    from_email: "security@yourcompany.com"
    from_name: "Segurança da Sua Empresa"
    template_id: 6735666
    subject: "Solicitação de Redefinição de Senha"

  # Confirmação de Sucesso na Redefinição de Senha
  api_crmCommunicationUserPasswordResetSuccess:
    from_email: "security@yourcompany.com"
    from_name: "Segurança da Sua Empresa"
```

```
  template_id: 6118378
  subject: "Redefinição de Senha Bem-Sucedida"

# Notificação de Mudança de Senha
api_crmCommunicationUserPasswordChange:
  from_email: "security@yourcompany.com"
  from_name: "Segurança da Sua Empresa"
  template_id: 6118423
  subject: "Senha Alterada"

# Verificação de Email
api_crmCommunicationEmailVerification:
  from_email: "verify@yourcompany.com"
  from_name: "Verificação da Sua Empresa"
  template_id: 6267350
  subject: "Verifique Seu Endereço de Email"

# Notificação de Saldo Expirado
api_crmCommunicationsBalanceExpired:
  from_email: "alerts@yourcompany.com"
  from_name: "Alertas da Sua Empresa"
  template_id: 7238252
  subject: "Saldo de Serviço Expirado"

# Aviso de Saldo Baixo
api_crmCommunicationsBalanceLow:
  from_email: "alerts@yourcompany.com"
  from_name: "Alertas da Sua Empresa"
  template_id: 7238263
  subject: "Aviso de Saldo Baixo"
```

Criando Modelos Mailjet:

1. Faça login no painel do Mailjet (<https://app.mailjet.com>)
2. Navegue até **Transacional → Modelos**
3. Crie um novo modelo ou clone um existente
4. Anote o **ID do Modelo** (valor numérico)
5. Adicione variáveis de modelo correspondentes à estrutura de dados do OmniCRM
6. Atualize `crm_config.yaml` com o ID do modelo

Variáveis de Modelo Disponíveis:

Cada tipo de email recebe variáveis específicas. Exemplos comuns:

- `{{customer_name}}` - Nome do cliente ou usuário
- `{{service_name}}` - Nome do serviço ou produto
- `{{invoice_id}}` - Número da fatura
- `{{invoice_amount}}` - Valor total da fatura
- `{{due_date}}` - Data de vencimento do pagamento
- `{{reset_link}}` - URL de redefinição de senha
- `{{verification_link}}` - URL de verificação de email
- `{{balance}}` - Saldo atual da conta
- `{{expiry_date}}` - Data de expiração do saldo ou serviço

Configuração de Provisionamento

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

Propósito:

- `failure_list` - Endereços de email notificados quando o provisionamento do Ansible falha
- As notificações incluem o nome do playbook, detalhes do erro e informações do cliente
- Permite que a equipe de operações responda rapidamente a problemas de provisionamento

Configuração de Fatura

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

Propósito:

Especifica qual modelo HTML Jinja2 usar para a geração de faturas em PDF.

Localização do Modelo: `/OmniCRM-API/invoice_templates/`

Consulte a seção **Geração de PDF de Fatura** abaixo para detalhes sobre como criar modelos personalizados.

URL Base do CRM

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

Propósito:

- Usado pelos playbooks do Ansible para fazer callbacks da API
- Usado em modelos de email para gerar links para o CRM
- Deve ser a URL publicamente acessível da sua API (não nomes de contêiner internos)

Exemplos:

- Desenvolvimento: `http://localhost:5000`
- Produção: `https://api.yourcompany.com`
- Docker: `http://omnicrm-api:5000` (comunicação interna entre contêineres)

Importante:

- Não inclua barra final
- Use URL publicamente acessível se os playbooks forem executados em servidores diferentes
- Atualize ao implantar em produção

Configuração OCS e CGRates

```
ocs:
  ocsApi: 'http://10.179.2.142:8080/api'
  ocsTenant: 'mnc380.mcc313.3gppnetwork.org'
  cgrates: 'localhost:2080'
```

Configuração:

- `ocsApi` - Endpoint da API OCS para gerenciamento de assinantes
- `ocsTenant` - Identificador do locatário para implantações OCS multi-locatário
- `cgrates` - Endpoint da API JSON-RPC do CGRates (host:porta)

Usado para: Cobranca em tempo real, gerenciamento de saldo, rastreamento de uso.

Configuração SMSC (Gateway SMS)

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

Propósito:

- Enviar notificações SMS para clientes (saldo baixo, alertas de serviço, códigos 2FA)
- `source_msisdn` - ID do remetente exibido para os destinatários (alfanumérico ou número de telefone)
- `smsc_url` - Endpoint da API do gateway SMSC
- `api_key` - Autenticação para a API SMSC

Configuração de Broadcast de Células

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

Propósito: Endpoint da API do Centro de Broadcast de Células (CBC) para alertas de emergência.

Consulte `features_cell_broadcast` para detalhes de uso.

Chave Secreta JWT

```
jwt_secret: 'CHANGE_ME_ON_DEPLOYMENT'  # Gerado automaticamente pelo Ansible
```

Segurança:

- Usado para assinar e verificar tokens de autenticação (JWT)
- **Crítico para a segurança da autenticação** - trate como uma senha
- Cada implantação/ambiente DEVE ter um segredo único
- Nunca compartilhe ou comite no controle de versão

Gerenciamento Automatizado de Segredos (Implantação Ansible):

Ao implantar com Ansible, o segredo JWT é gerenciado automaticamente:

1. **Primeira implantação:** O Ansible gera uma string hexadecimal aleatória de 64 caracteres (256 bits de entropia)
2. **Implantações subsequentes:** O Ansible preserva o segredo existente do servidor
3. **Cada servidor recebe seu próprio segredo único** - sessões não funcionam entre ambientes

Isso garante:

- Nenhum segredo codificado em repositórios git
- Cada ambiente é criptograficamente isolado

- Segredos persistem entre implantações (sessões não são invalidadas na nova implantação)
- Reconstruir um servidor gera um novo segredo (sessões são corretamente invalidadas)

Geração Manual de Segredo (Implantações Não-Ansible):

Se implantar sem Ansible, você DEVE gerar manualmente um segredo único:

```
# Gere uma chave aleatória criptograficamente segura
python3 -c "import secrets; print(secrets.token_hex(32))"
# ou
openssl rand -hex 32
```

Então atualize `crm_config.yaml`:

```
jwt_secret: 'your_generated_secret_here'
```

Importante:

- **Nunca use o valor padrão de espaço reservado** `CHANGE_ME_ON_DEPLOYMENT` em produção
- **Nunca compartilhe publicamente** - qualquer pessoa com o segredo pode forjar tokens de autenticação
- Alterar isso invalida todas as sessões de usuário existentes (os usuários devem fazer login novamente)
- Cada ambiente (dev, staging, prod) DEVE ter segredos diferentes

Configuração de Pagamento Stripe

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

Configuração:

- `secret_key` - Chave secreta da API Stripe (lado do servidor, mantenha confidencial)
- `publishable_key` - Chave publicável da Stripe (lado do cliente, seguro para expor)
- `currency` - Código de moeda ISO 4217 (aud, usd, gbp, eur, etc.)
- `statement_descriptor_suffix` - Aparece nos extratos de cartão de crédito dos clientes

Tipos de Chaves:

- Chaves de teste: `sk_test_...` e `pk_test_...` (para desenvolvimento)
- Chaves ao vivo: `sk_live_...` e `pk_live_...` (para produção)

Uso do Descritor de Extrato:

- Exibido nos extratos bancários dos clientes como "YOURCOMPANY"
- Máximo de 22 caracteres
- Ajuda os clientes a identificar cobranças
- Também usado nos nomes de arquivos PDF de faturas (por exemplo, `YOURCOMPANY_12345.pdf`)

Consulte `Integrações de Fornecedores de Pagamento <integrations_payment_vendors>` para detalhes de configuração.

Chaves de API e Whitelisting de IP

Defina chaves de API com acesso baseado em funções e restrições de 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"]

# Whitelist de IP (autônomo, sem chave de API)
ip_whitelist:
  "10.179.2.142":
    roles: ["admin"]
```

Propósito:

- Permitir que sistemas externos se autentiquem via chave de API
- Restringir acesso por endereço IP
- Conceder funções específicas aos consumidores da API
- Útil para integrações (sistemas de cobrança, monitoramento, automação)

Gerando Chaves de API:

```
openssl rand -base64 48
```

Funções:

- `admin` - Acesso total a todos os endpoints
- Funções personalizadas definidas no sistema RBAC

Melhores Práticas de Segurança:

- Use chaves de API longas e aleatórias (mínimo 32 caracteres)
- Restringir IPs a fontes conhecidas apenas
- Conceder as funções mínimas necessárias
- Rotacione chaves de API regularmente
- Monitore o uso de chaves de API nos logs

Aviso de Segurança:

- Use a whitelist de IP apenas para redes internas confiáveis
- NÃO deve usar IPs localhost (127.0.0.1, ::1)
- Use chaves de API em vez disso para acesso externo
- Considere regras de firewall como proteção adicional

Consulte `administration_api_keys` e `concepts_api` para detalhes.

Configuração de Frontend (.env)

A UI React é configurada através de variáveis de ambiente em `OmniCRM-UI/.env`.

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

Chaves de API e Serviços Externos

```
# Google Maps API (para autocomplete de endereço e geocodificação)
VITE_GOOGLE_API_KEY=your_google_api_key

# Gateway de Pagamento Stripe
VITE_STRIPE_PUBLISHABLE_KEY=pk_test_xxxxx

# Desativar auto-lançamento do navegador ao iniciar npm
BROWSER=none
```

Deve corresponder:

`VITE_STRIPE_PUBLISHABLE_KEY` deve corresponder a `stripe.publishable_key` em `crm_config.yaml`.

Branding e Informações da Empresa

```
# Branding da Empresa
VITE_COMPANY_NAME="ShellFone"
VITE_PORTAL_NAME="ShellManager"
VITE_SELF_CARE_NAME="ShellCare"
VITE_COMPANY_TAGLINE="Phones with Shells"
```

Esses valores aparecem em toda a UI:

- `COMPANY_NAME` - Exibido em títulos de página, emails e comunicações com clientes
- `PORTAL_NAME` - Nome do portal de admin/funcionários (por exemplo, "ShellManager")
- `SELF_CARE_NAME` - Nome do portal de autoatendimento do cliente (por exemplo, "ShellCare")
- `COMPANY_TAGLINE` - Aparece nas telas de login e materiais de marketing

Localização e Configurações Regionais

```
# Idioma e Localidade
# Idiomas suportados: ar, ch, en, fr, gr, it, ru, sp
VITE_DEFAULT_LANGUAGE=en
VITE_LOCALE="en-GB"

# Localização Padrão (para autocomplete de endereço)
VITE_DEFAULT_LOCATION="Sydney, Australia"
VITE_DEFAULT_COUNTRY="Australia"

# Configurações de Moeda
VITE_CURRENCY_CODE="GBP"
VITE_CURRENCY_SYMBOL="£"
```

Idiomas Suportados:

- `ar` - Árabe
- `ch` - Chinês

- `en` - Inglês (padrão)
- `fr` - Francês
- `gr` - Grego
- `it` - Italiano
- `ru` - Russo
- `sp` - Espanhol

Moedas Comuns:

- USD - $ (Dólar Americano)
- GBP - £ (Libra Esterlina)
- EUR - € (Euro)
- AUD - $ (Dólar Australiano)
- CAD - $ (Dólar Canadense)

Nota: Deve corresponder a `stripe.currency` em `crm_config.yaml`.

Configuração do Tema de Cores

```
# Cor Primária (cor principal da marca)
VITE_PRIMARY_COLOR=#405189

# Opções de Cores Adicionais (exemplos comentados)
# VITE_SECONDARY_COLOR=#2bFFcf
# VITE_TERTIARY_COLOR=#1a9fbf
# VITE_SUCCESS_COLOR=#28a745
# VITE_INFO_COLOR=#17a2b8
# VITE_WARNING_COLOR=#ffc107
# VITE_DANGER_COLOR=#dc3545
```

Cores Disponíveis:

- `VITE_PRIMARY_COLOR` - Cor principal da marca (botões, links, destaques)
- `VITE_SECONDARY_COLOR` - Cor de destaque
- `VITE_TERTIARY_COLOR` - Destaque adicional
- `VITE_SUCCESS_COLOR` - Mensagens de sucesso (#28a745)
- `VITE_INFO_COLOR` - Mensagens informativas (#17a2b8)

- `VITE_WARNING_COLOR` - Avisos (#ffc107)
- `VITE_DANGER_COLOR` - Erros (#dc3545)
- `VITE_LIGHT_COLOR` - Fundos claros (#f8f9fa)
- `VITE_DARK_COLOR` - Texto escuro (#343a40)
- `VITE_PRIMARY_DARK_COLOR` - Variante escura da cor primária

Formato: Códigos de cores hexadecimais (#RRGGBB)

A cor primária é aplicada a:

- Cabeçalhos de navegação
- Botões de ação
- Links e destaques
- Estados ativos
- Elementos de marca

Configuração de Fonte

```
# Configuração da Família de Fontes
# 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
# Sistema: Sistema (fontes nativas do dispositivo)
# Padrão: Quicksand
VITE_FONT_FAMILY=Quicksand
```

Importante: Todas as fontes são **auto-hospedadas localmente** dentro da aplicação OmniCRM-UI. Isso significa:

- **Nenhum carregamento de fonte externo** - As fontes são agrupadas com a aplicação
- **Compatível com jardim murado** - Nenhum acesso à internet é necessário para que as fontes funcionem
- **Operação offline** - Funcionalidade completa em ambientes sem conexão ou com rede restrita

- **Privacidade** - Nenhuma solicitação externa para Google Fonts, Adobe Fonts ou outros CDNs
- **Desempenho** - Carregamento mais rápido sem dependências externas
- **Segurança** - Nenhum rastreamento de terceiros ou vazamento de dados através de solicitações de fonte

Opções Disponíveis:

Fontes Sans-Serif:

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

Fontes Serif:

- Merriweather, Lora, Playfair Display

Fontes do Sistema:

- Sistema - Usa fontes nativas do dispositivo para melhor desempenho e menor tamanho de pacote

Adicionando Fontes Personalizadas:

Sim, você pode adicionar fontes adicionais! Todas as fontes são armazenadas localmente na aplicação.

Para adicionar uma nova fonte personalizada:

1. **Adicione arquivos de fonte** a `OmniCRM-UI/src/assets/fonts/your-font-name/`
    - Use o formato WOFF2 para melhor compressão e suporte a navegadores
    - Inclua múltiplos pesos (300, 400, 500, 600, 700) para renderização adequada
    - Nomeie os arquivos: `your-font-name-300.woff2`, `your-font-name-400.woff2`, etc.

2. **Defina regras @font-face** em `OmniCRM-UI/src/assets/scss/fonts/_google-fonts.scss`

```
//
// Sua Fonte Personalizada - Descrição
//

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

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

3. **Defina no arquivo .env:**

```
VITE_FONT_FAMILY=Your Font Name
```

Diretrizes de Peso de Fonte:

- 300 - Leve (opcional, para títulos sutis)
- 400 - Regular (obrigatório, texto padrão)
- 500 - Médio (opcional, ênfase)
- 600 - Semi-Negrito (opcional, subtítulos)
- 700 - Negrito (obrigatório, títulos e texto forte)

Links Rápidos para Aplicativos Web

Configure até 6 aplicativos web de acesso rápido que aparecem no painel de administração:

```
# Aplicativo 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"

# Aplicativo 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"

# Aplicativo Web 3-6: Integrações adicionais
# (Configure de forma semelhante com NAME, URL e ICON_PATH)
```

Padrão:

- `VITE_WEB_APP_N_NAME` - Nome exibido
- `VITE_WEB_APP_N_URL` - URL de destino
- `VITE_WEB_APP_N_ICON_PATH` - Caminho do arquivo do ícone (relativo a public/)

Exemplos de Ícones: GitHub, Xero, Monday.com, Gmail, MailJet, Slack

Integração do Dashboard Grafana

O OmniCRM se integra ao Grafana para fornecer dashboards de análises. Usuários com a permissão `grafana_access` podem acessar o Grafana em `/grafana` para criar dashboards personalizados, e esses dashboards podem ser incorporados na interface do OmniCRM.

```
# Integração do Dashboard Grafana
# IDs dos Dashboards (separados por vírgula) - obtenha esses IDs a
partir das URLs /grafana/d/ID/name
VITE_GRAFANA_DASHBOARD_IDS=abc123,def456,ghi789

# Rótulos dos Dashboards (separados por vírgula, devem
corresponder à ordem dos IDs)
VITE_GRAFANA_DASHBOARD_LABELS=Métricas do Sistema,Análises de
Usuários,Desempenho da Rede
```

Campos:

- `VITE_GRAFANA_DASHBOARD_IDS` - Lista separada por vírgulas dos IDs dos dashboards do Grafana a partir do caminho da URL
- `VITE_GRAFANA_DASHBOARD_LABELS` - Lista separada por vírgulas dos nomes de exibição dos dashboards (devem corresponder à ordem dos IDs)

Como Encontrar IDs de Dashboards:

Ao visualizar um dashboard no Grafana, a URL será semelhante a `/grafana/d/abc123/dashboard-name`. O ID do dashboard é a parte após `/d/` e antes do nome (neste exemplo, `abc123`).

Uso:

Uma vez configurados, os dashboards aparecem automaticamente na barra lateral do OmniCRM sob o menu "Dashboards". Os usuários podem clicar para visualizar dashboards incorporados enquanto mantêm a navegação do OmniCRM visível.

Exemplo:

Se sua URL de dashboard for `http://your-server/grafana/d/system-overview-123/system-metrics`, então:

- ID do Dashboard: `system-overview-123`
- Rótulo do Dashboard: Escolha qualquer nome amigável como "Visão Geral do Sistema"

Importante:

- O número de IDs deve corresponder ao número de rótulos
- Os dashboards aparecem na barra lateral na ordem em que são definidos
- Apenas usuários com a permissão `grafana_access` podem visualizar dashboards
- Mudanças requerem recompilação da UI frontend

URLs de Suporte e Documentação

```
# URLs de FAQs e Suporte
VITE_FAQS_URL=https://support.yourcompany.com/faqs
VITE_SUPPORT_URL=https://support.yourcompany.com
```

Propósito: Links para recursos de suporte externos mostrados na UI.

Logins Sociais

```
# Permitir Logins Sociais (sim|não)
VITE_ALLOW_SOCIAL_LOGINS=yes
```

Opções:

- `yes` - Habilitar botões de login social (Google, Facebook, etc.)
- `no` - Desabilitar logins sociais

Nota: Os provedores de login social devem ser configurados separadamente.

Análises e Rastreamento

```
# Google Analytics 4 ou Google Tag Manager (opcional)
# Para GA4, use o formato: G-XXXXXXXXXX
# Para GTM, use o formato: GTM-XXXXXXX
VITE_GOOGLE_TAG_ID=G-XXXXXXXXXX
```

Propósito: Habilitar o rastreamento do Google Analytics 4 (GA4) ou Google Tag Manager (GTM) para monitorar o comportamento do usuário e o uso da aplicação.

Formatos Suportados:

- **ID de Medição GA4:** `G-XXXXXXXXXX` - Para rastreamento do Google Analytics 4
- **ID do Contêiner GTM:** `GTM-XXXXXXX` - Para Google Tag Manager
- **Desativado:** Deixe vazio ou omita para desabilitar o rastreamento

Como Funciona:

O sistema detecta automaticamente o serviço de rastreamento com base no formato do ID:

- IDs que começam com `G-` inicializam o Google Analytics 4
- IDs que começam com `GTM-` inicializam o Google Tag Manager
- Valor vazio ou ausente desabilita todo o rastreamento

Recursos:

- **Rastreamento Automático de Páginas** - Cada mudança de rota é rastreada como uma visualização de página
- **Rastreamento de Eventos Personalizados** - Rastreie interações do usuário, cliques em botões, envios de formulários
- **Nenhuma Mudança de Código Necessária** - Basta definir a variável de ambiente e recompilar a UI

Instruções de Configuração:

1. **Obtenha Seu ID de Rastreamento:**

   Para Google Analytics 4:

   - Faça login no Google Analytics
   - Crie ou selecione uma propriedade GA4
   - Vá para **Admin** → **Fluxos de Dados**
   - Selecione seu fluxo da web

- Copie o **ID de Medição** (formato: `G-XXXXXXXXXX`)

Para Google Tag Manager:

- Faça login no Google Tag Manager
- Crie ou selecione um contêiner
- Copie o **ID do Contêiner** (formato: `GTM-XXXXXXX`)

2. **Adicione ao Arquivo .env:**

```
VITE_GOOGLE_TAG_ID=G-XXXXXXXXXX
```

3. **Recompile a UI:**

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

4. **Verifique o Rastreamento:**

- Abra sua aplicação em um navegador
- Verifique o console do navegador para mensagens de inicialização
- Use os relatórios em tempo real do Google Analytics para ver usuários ativos

Uso Avançado - Rastreamento de Eventos Personalizados:

A integração fornece funções utilitárias para rastrear eventos personalizados em toda a sua aplicação.

Exemplo - Rastrear Cliques em Botões:

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

// No seu componente
const handleCheckout = () => {
  trackEvent('checkout_initiated', {
    cart_value: 99.99,
    item_count: 3,
    page: '/cart'
  });
  // Continue com a lógica de checkout
};
```

Exemplo - Rastrear Envios de Formulários:

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

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

Exemplo - Rastrear Visualizações de Páginas Personalizadas:

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

// Rastrear manualmente uma visualização de página (útil para
diálogos modais, abas, etc.)
trackPageView('/virtual/page-path', 'Título da Página');
```

Funções de Rastreamento Disponíveis:

Função	Propósito	Parâmetros
`initializeGoogleTracking()`	Inicializar rastreamento no in��cio do app	Nenhum (chamado automaticamente)
`trackEvent(eventName, eventParams)`	Rastrear eventos personalizados	Nome do evento (string), parâmetros (objeto)
`trackPageView(path, title)`	Rastrear visualizações de página manualmente	Caminho (string), título (string)

Privacidade e Conformidade:

O rastreamento do Google Analytics coleta dados de comportamento do usuário. Certifique-se de estar em conformidade com as regulamentações de proteção de dados:

- **GDPR (Europa)** - Obtenha consentimento do usuário antes do rastreamento, forneça mecanismo de opt-out
- **CCPA (Califórnia)** - Permita que os usuários optem por não participar da coleta de dados
- **Política de Privacidade** - Divulgue o uso do Google Analytics em sua política de privacidade
- **Consentimento de Cookies** - Implemente banner de consentimento de cookies se exigido por leis locais
- **Retenção de Dados** - Configure períodos de retenção de dados apropriados nas configurações do Google Analytics

Solução de Problemas:

Problema	Causa Possível	Solução
Sem dados no painel GA	ID de rastreamento incorreto	Verifique se o formato do ID corresponde a G-XXXXXXXXXX ou GTM-XXXXXXX
Erros no console ao carregar a página	ID de rastreamento inválido	Verifique se há erros de digitação no arquivo .env
Visualizações de página não rastreadas	UI não recompilada após alteração no .env	Execute `npm run build` e reinicie a UI
Eventos não aparecendo	Bloqueador de anúncios ativado	Teste com o bloqueador de anúncios desativado
Dados atrasados	Comportamento normal do GA	Use relatórios em tempo real para verificação imediata

Considerações de Desempenho:

- Scripts de rastreamento são carregados de forma assíncrona e não bloqueiam a renderização da página
- Impacto mínimo no desempenho (< 50KB de carga adicional)
- Scripts são armazenados em cache pelos navegadores após o primeiro carregamento
- Nenhum rastreamento ocorre se `VITE_GOOGLE_TAG_ID` estiver vazio

Detalhes de Implementação:

- Código de rastreamento: `OmniCRM-UI/src/utils/googleAnalytics.js`
- Hook de rastreamento de página: `OmniCRM-UI/src/utils/usePageTracking.js`
- Inicialização: Automática na inicialização do app em `App.js` ou `index.js`

Configuração de Recarga e Recarregamento

```
VITE_TOPUP_PRICE_PER_DAY=10
```

Propósito: Define o preço por dia para serviços de recarga/recarregamento no portal de autoatendimento.

Campos:

- `VITE_TOPUP_PRICE_PER_DAY` - Preço diário para serviços de recarga (valor numérico)

Exemplo: Se definido como `10` e a moeda é USD, os clientes pagam $10 por dia de serviço.

Nota: Este valor deve corresponder à configuração de preços do backend. Consulte `features_topup_recharge` para detalhes completos de configuração.

Branding e Personalização Visual

O OmniCRM permite que você substitua imagens de branding padrão pelo logotipo e telas de abertura da sua empresa sem modificar o código.

Arquivos de Logotipo e Sistema de Fallback

Os logotipos são armazenados em `/OmniCRM-UI/src/assets/images/omnitouch/` e usam um sistema de fallback:

Logotipos Padrão (sempre presentes):

- `DefaultLogoDark.png` - Logotipo do tema escuro (usado em fundos claros)
- `DefaultLogoLight.png` - Logotipo do tema claro (usado em fundos escuros)

Logotipos Personalizados (opcionais, têm precedência quando presentes):

- `logoSm.png` - Logotipo pequeno para barra lateral colapsada (recomendado: 100x100px)
- `logoDark.png` - Logotipo escuro em tamanho completo para cabeçalhos (recomendado: 200x50px)
- `logoLight.png` - Logotipo claro em tamanho completo para telas de autenticação (recomendado: 200x50px)

Como Funciona o Fallback do Logotipo:

O sistema tenta carregar logotipos personalizados primeiro. Se um arquivo de logotipo personalizado não existir, ele recai para o padrão:

```
// De Header.js
const tryImport = (filename) => {
    try {
        return require(`../assets/images/omnitouch/${filename}`);
    } catch (err) {
        return null; // Volta para o padrão
    }
};

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

Onde os Logotipos Aparecem:

- **logoSm.png** - Barra lateral colapsada, navegação móvel, exibições de cabeçalho pequenas
- **logoDark.png** - Barra de cabeçalho principal (modo claro), cabeçalho do painel de administração
- **logoLight.png** - Telas de login/registro, fundos escuros, carrossel de autenticação

Substituindo Logotipos:

1. **Crie Seus Arquivos de Logotipo:**
    - Use o formato PNG para suporte a transparência

- Correspondam às dimensões recomendadas acima
- Certifique-se de que os logotipos estejam claros em resoluções regulares e retina

2. **Adicione ao OmniCRM:**

```
# Copie seus arquivos de logotipo para o diretório de imagens
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. **Recompile a UI:**

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

4. **Verifique as Alterações:**

- Verifique o cabeçalho do modo claro (deve mostrar `logoDark.png`)
- Verifique o cabeçalho do modo escuro (deve mostrar `logoLight.png`)
- Verifique a barra lateral colapsada (deve mostrar `logoSm.png`)
- Verifique a tela de login (deve mostrar `logoLight.png`)

Melhores Práticas de Design de Logotipo:

- **Contraste** - Certifique-se de que os logotipos sejam visíveis em fundos claros e escuros
- **Simplicidade** - Logotipos devem ser reconhecíveis em tamanhos pequenos
- **Formato** - Use PNG com fundos transparentes
- **Retina** - Forneça resolução 2x para displays de alta DPI
- **Consistência** - Use as mesmas cores da marca em todas as variantes do logotipo

Telas de Abertura e Fundos de Autenticação

As telas de autenticação (login, registro, redefinição de senha) usam um fundo de carrossel com imagens personalizáveis.

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

Configuração Padrão:

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

// Logotipo exibido nas telas de autenticação
<!-- ![Logotipo da Empresa](../absolute/path/to/your-logo.png) -->
        <h1>FATURA</h1>
    </div>

    <div class="invoice-details">
        <p><strong>Número da Fatura:</strong> {{ invoice_number }}
</p>
        <p><strong>Data:</strong> {{ date }}</p>
        <p><strong>Data de Vencimento:</strong> {{ due_date }}</p>
        <p><strong>Período de Cobrança:</strong> {{ start_date }}
a {{ end_date }}</p>
    </div>

    <div class="customer-details">
        <h3>Faturar Para:</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>Descrição</th>
                <th>Data</th>
                <th>Valor</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>Valor Total Devido: ${{ "%.2f"|format(total_amount) }}
</p>
    </div>

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

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

Melhores Práticas para Modelos:

- **Use caminhos absolutos para imagens** - `file:///absolute/path/to/image.png`
- **CSS Inline** - WeasyPrint não carrega folhas de estilo externas de forma confiável
- **Teste com dados de exemplo** - Use `invoice_templates/rendered/` para inspecionar HTML
- **Quebras de página** - Use `<div style="page-break-after: always;"></div>` para faturas de várias páginas
- **Cabeçalhos e rodapés** - Use regras CSS `@page` para elementos repetidos
- **Formatação de moeda** - Use filtros Jinja2: `{{ "%.2f"|format(amount) }}`

Criando um Modelo de Fatura Personalizado

1. **Copie o Modelo de Exemplo:**

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

2. **Edite o Modelo:**
   - Substitua o nome da empresa, logotipo, informações de contato
   - Ajuste o estilo (cores, fontes, layout) para corresponder à marca
   - Adicione ou remova seções conforme necessário (quebra de impostos, instruções de pagamento, etc.)
3. **Atualize a Configuração:**

   Edite `crm_config.yaml`:

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

4. **Teste a Geração de Faturas:**
   - Crie uma fatura de teste no CRM
   - Baixe o PDF e verifique a formatação
   - Verifique `invoice_templates/rendered/{invoice_id}.html` para erros de depuração
5. **Invalidar Caches Antigos (se necessário):**

   Se você alterou o modelo e deseja regenerar faturas existentes:

```sql
-- Limpar todos os PDFs em cache (força a regeneração)
DELETE FROM Invoice_PDF_Cache;
```

Sistema de Cache de PDF

Para melhorar o desempenho, o OmniCRM armazena em cache PDFs gerados:

Comportamento do Cache:

- **Primeira Solicitação** - O PDF é gerado, armazenado em cache e retornado
- **Solicitações Subsequentes** - O PDF em cache é retornado imediatamente (sem regeneração)
- **Invalidação de Cache** - Ocorre quando a fatura é modificada, anulada ou reembolsada
- **Limpeza de Cache** - Caches antigos são automaticamente purgados após 30 dias de inatividade

Armazenamento de Cache:

- PDF codificado em base64 armazenado na tabela `Invoice_PDF_Cache`
- Hash de conteúdo SHA256 para verificação de integridade
- Inclui nome do arquivo, timestamp de criação, timestamp da última acessibilidade

Gerenciamento Manual de Cache:

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

session = get_db_session()

# Limpar caches mais antigos que 30 dias
result = cleanup_old_pdf_cache(session, days_old=30)
print(result)  # {'status': 'success', 'deleted_count': 15}

# Invalidar cache de fatura específica
invalidate_invoice_cache(session, invoice_id='12345')
```

Endpoints da API:

Gerar/download de PDF de fatura:

GET /invoice/pdf/{invoice_id}

Resposta: Download do arquivo PDF com nome do arquivo a partir do descritor de extrato do Stripe

Cabeçalhos de Cache:

- Primeira solicitação: Resposta mais lenta (tempo de geração)
- Solicitações em cache: Resposta instantânea
- A correspondência de cache/falha é transparente para o usuário

---

Aplicando Alterações de Configuração

Backend (crm_config.yaml)

1. Edite `OmniCRM-API/crm_config.yaml`
2. Salve as alterações
3. Reinicie o serviço da API:

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

As alterações entram em vigor imediatamente após o reinício.

Frontend (.env)

1. Edite `OmniCRM-UI/.env`
2. Salve as alterações
3. Recompile a UI:

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

4. Reinicie o serviço da UI ou servidor web

Modo de Desenvolvimento:

Durante o desenvolvimento com `npm start`, reinicie o servidor de desenvolvimento para aplicar as alterações.

Melhores Práticas de Configuração

Segurança

- **Nunca comite segredos** - Use `.gitignore` para arquivos de configuração com credenciais
- **Use senhas fortes** - Mínimo de 16 caracteres com letras maiúsculas, minúsculas, números e símbolos
- **Rotacione credenciais regularmente** - Especialmente para implantações em produção
- **Restringir acesso ao banco de dados** - Use whitelisting de IP e regras de firewall
- **Use configurações específicas para o ambiente** - Separe configurações de dev/staging/produção
- **Limite permissões de chave de API** - Atribua funções mínimas necessárias
- **Use whitelisting de IP com moderação** - Prefira chaves de API para melhor segurança

Manutenção

- **Documente alterações** - Mantenha um changelog de modificações de configuração
- **Faça backup das configurações** - Armazene cópias antes de grandes alterações
- **Teste em staging** - Verifique alterações de configuração antes da implantação em produção

- **Controle de versão** - Rastreie modelos de configuração (sem segredos) no git

Desempenho

- **Use banco de dados local** - Evite banco de dados remoto para melhor desempenho
- **Configure caching** - Habilite caching OCS se disponível
- **Otimize Grafana** - Limite o número de dashboards incorporados

Branding

- **Combine cores** - Certifique-se de que as cores da UI complementem seu logotipo
- **Teste contraste** - Verifique a legibilidade do texto em fundos coloridos
- **Teste em dispositivos móveis** - Verifique o branding em dispositivos móveis
- **Posicionamento do logotipo** - Use tamanhos de logotipo apropriados para diferentes contextos

---

Solução de Problemas

Problemas Comuns

Alterações não aplicadas

- **Causa:** Serviço não reiniciado ou UI não recompilada
- **Solução:** Reinicie os serviços da API/UI após alterações de configuração

Erros de sintaxe YAML

- **Causa:** Formatação YAML inválida (indentação, aspas, etc.)
- **Solução:** Valide YAML online ou use `yamllint crm_config.yaml`

Falha na conexão com o banco de dados

- **Causa:** Credenciais erradas ou servidor inacessível
- **Solução:** Verifique se o banco de dados está em execução, se as credenciais estão corretas

Pagamentos Stripe não funcionando

- **Causa:** Chaves incompatíveis entre backend e frontend
- **Solução:** Certifique-se de que `publishable_key` corresponda em ambos os arquivos

Emails não enviados

- **Causa:** Credenciais do Mailjet inválidas ou IDs de modelo
- **Solução:** Verifique a chave/secreto da API Mailjet, verifique se os IDs de modelo existem

Falha na Geração de PDF:

- Verifique se o WeasyPrint está instalado: `pip install weasyprint`
- Verifique se o nome do modelo corresponde a `crm_config.yaml`
- Verifique `invoice_templates/rendered/` para erros de renderização HTML
- Revise os logs da API para erros de modelo Jinja2

Imagens não aparecem no PDF:

- Use caminhos de arquivo absolutos: `file:///full/path/to/image.png`
- Certifique-se de que os arquivos de imagem existam e sejam legíveis
- Verifique o formato da imagem (PNG e JPEG funcionam melhor)
- Verifique se os caminhos das imagens não contêm caracteres especiais

Problemas de Estilo:

- Inline todo o CSS (folhas de estilo externas não são suportadas)
- Evite recursos CSS complexos (flexbox, grid podem não renderizar corretamente)
- Teste com layouts simples primeiro, adicione complexidade gradualmente
- Use tabelas para layout em vez de divs sempre que possível

Cache não invalidando:

- Verifique se `invalidate_invoice_cache()` é chamado quando a fatura é modificada
- Verifique se as atualizações de transação acionam a invalidação do cache
- Exclua manualmente da tabela `Invoice_PDF_Cache` se necessário

Checklist de Configuração

Use esta lista de verificação ao implantar o OmniCRM:

Configuração de Backend

- [ ] Copie `.env.example` para `.env`
- [ ] Defina senhas fortes para o banco de dados
- [ ] Configure credenciais do CGRates
- [ ] Atualize `crm_config.yaml` com suas configurações:
  - [ ] Conexão com o banco de dados
  - [ ] Tipos de serviço
  - [ ] Chaves da API Mailjet e IDs de modelo
  - [ ] Emails de notificação de falha de provisionamento
  - [ ] Nome do arquivo do modelo de fatura
  - [ ] URL base do CRM (acessível publicamente)
  - [ ] Endpoints OCS/CGRates
  - [ ] Configuração do SMSC
  - [ ] Gere uma nova chave secreta JWT
  - [ ] Chaves Stripe (ao vivo, não teste)
  - [ ] Chaves de API e whitelisting de IP

Configuração de Frontend

- [ ] Copie `OmniCRM-UI/.env.example` para `OmniCRM-UI/.env`
- [ ] Defina a chave da API do Google Maps

- [ ] Defina a chave publicável do Stripe
- [ ] Atualize o branding da empresa:
  - [ ] Nome da empresa
  - [ ] Nome do portal
  - [ ] Nome do autoatendimento
  - [ ] Slogan da empresa
- [ ] Configure a localização:
  - [ ] Idioma padrão
  - [ ] Localidade
  - [ ] Localizaç��o e país padrão
  - [ ] Código e símbolo da moeda
- [ ] Defina a cor principal da marca
- [ ] Configure integrações de aplicativos web (opcional)
- [ ] Adicione URLs de suporte e FAQ (opcional)
- [ ] Defina o ID de rastreamento do Google Analytics/Tag Manager (opcional)

Ativos de Branding

- [ ] Crie arquivos de logotipo (logoSm.png, logoDark.png, logoLight.png)
- [ ] Faça upload dos logotipos para `OmniCRM-UI/src/assets/images/omnitouch/`
- [ ] Crie modelo HTML de fatura personalizado
- [ ] Faça upload do modelo de fatura para `OmniCRM-API/invoice_templates/`
- [ ] Atualize `crm_config.yaml` com o nome do arquivo do modelo de fatura
- [ ] Teste a geração de PDF de fatura
- [ ] Recompile a UI: `npm run build`

Segurança

- [ ] Altere todas as senhas padrão
- [ ] Gere um segredo JWT exclusivo
- [ ] Use chaves Stripe de produção (não chaves de teste)
- [ ] Rotacione chaves da API Mailjet
- [ ] Habilite regras de firewall
- [ ] Configure whitelisting de IP para acesso à API

- [ ] Configure certificados SSL/TLS
- [ ] Habilite HTTPS para todos os endpoints
- [ ] Revise configurações CORS
- [ ] Implemente limitação de taxa
- [ ] Configure procedimentos de backup e recuperação

Testes

- [ ] Teste o fluxo de registro de cliente
- [ ] Teste o provisionamento de serviços de ponta a ponta
- [ ] Verifique se as notificações por email são enviadas corretamente
- [ ] Teste a geração de faturas e download de PDF
- [ ] Verifique o processamento de pagamentos (Stripe)
- [ ] Verifique a autenticação de usuários e 2FA
- [ ] Teste a impersonação e registro de auditoria
- [ ] Verifique se os dados de uso sincronizam do OCS
- [ ] Teste a criação e renovação de ActionPlan
- [ ] Confirme se a alocação de inventário funciona corretamente

Implantação

- [ ] Construa imagens Docker ou implante em servidores
- [ ] Inicie contêineres de banco de dados (MySQL, PostgreSQL)
- [ ] Inicie o CGRates
- [ ] Inicie a API OmniCRM
- [ ] Inicie a UI OmniCRM
- [ ] Configure proxy reverso (nginx, traefik)
- [ ] Configure monitoramento (Grafana, Prometheus)
- [ ] Configure agregação de logs
- [ ] Configure backups automatizados
- [ ] Documente a arquitetura de implantação
- [ ] Treine a equipe sobre o uso do sistema

Documentação Relacionada

- `RBAC e Gerenciamento de Usuários </rbac>`
- `Produtos e Serviços </concepts_products_and_services>`
- `Provisionamento Ansible </concepts_ansible>`
- `Gerenciamento de Inventário </administration_inventory>`
- `Faturas de Clientes </payments_invoices>`
- `Autenticação de Dois Fatores </2fa>`
- `Atendimento ao Cliente e Impersonação </customer_care>`
- `Integrações de Fornecedores de Pagamento <integrations_payment_vendors>`
- `administration_api_keys` - gerenciamento de chaves de API
- `integrations_mailjet` - integração de email
- `concepts_api` - autenticação da API

Visão Geral do Inventário no OmniCRM

O sistema de **Inventário** no OmniCRM é projetado para gerenciar e rastrear tanto itens físicos quanto virtuais usados por operadores de rede e clientes.

Isso significa que podemos rastrear todos os tipos de itens, como modems, números de telefone, blocos de endereços IP ou até mesmo hardware físico, como ONTs GPON ou CPEs de Wireless Fixo.

Veja também: `Customer Attributes <administration_attributes>` para armazenar metadados personalizados e `Customer Tags <administration_tags>` para categorização visual.

Para apoiar clientes com uma pegada de rede fixa, o sistema de inventário também pode rastrear residências atendidas para um determinado serviço, permitindo que os operadores façam qualificação de serviço remotamente, e para aqueles que operam uma rede wireless fixa, podemos rastrear os CPEs implantados em campo com suas localizações.

Nota

Os itens de inventário estão vinculados a produtos durante a provisão através do campo `inventory_items_list`. Para um guia completo de como o inventário se integra com a provisão de produtos, incluindo a interface do usuário do seletor de inventário e a integração do playbook Ansible, veja `Complete Product Lifecycle Guide - Inventory Requirements <guide_product_lifecycle>`.

Propósito

O Inventário do OmniCRM serve a vários propósitos principais:

1. **Provisionamento de Serviços**: Quando um cliente se inscreve para um serviço, itens como modems, cartões SIM ou números de telefone podem

precisar ser alocados. O sistema de inventário rastreia esses itens e os associa aos clientes.

2. **Gerenciamento de Estoque**: Para estoque físico, como hardware ou outros equipamentos, o inventário ajuda os operadores a manter visibilidade sobre o que está disponível, onde está armazenado e o que foi alocado ou vendido aos clientes.
3. **Alocação de Clientes**: O sistema permite que itens sejam alocados a clientes, seja para uso em um serviço (por exemplo, atribuindo um modem à conta de internet de um cliente) ou para venda direta.
4. **Qualificação de Serviço / Pegada de Rede**: Ao armazenar informações sobre a pegada da rede, como cada residência atendida para um serviço GPON, ou cada CPE Wireless Fixo implantado, permite que a equipe faça qualificação de serviço remotamente e veja se há uma pegada de rede em uma área específica.

Exemplo de Ciclo de Vida do Inventário

Para ilustrar como o sistema de Inventário funciona, considere os seguintes exemplos.

Exemplo de Cartão SIM

Um lote de 1000 cartões SIM é encomendado da Omnitouch. Primeiramente, um Modelo de Inventário é criado para cartões SIM (se já não existir) e os SIMs encomendados são carregados no Inventário no estado Em Trânsito.

Uma vez que os SIMs são recebidos, eles são marcados como Em Estoque, e podem ser alocados para diferentes lojas de varejo, com a Localização do item de inventário atualizada para refletir em qual loja de varejo cada cartão SIM está - Isso é útil para rastrear níveis de estoque em cada loja e garantir que cada loja tenha estoque suficiente para atender à demanda dos clientes.

Quando um cliente se inscreve para um serviço móvel na loja, um cartão SIM é alocado ao cliente e o status é alterado para Alocado. O item de inventário do cartão SIM é então atribuído ao cliente, e o status é atualizado para Em Uso.

Se o cliente cancelar o serviço ou o serviço ficar inativo, o cartão SIM é marcado como Descomissionado.

Exemplo de Residências Atendidas GPON

Para uma rede GPON, o sistema de inventário pode rastrear cada residência atendida para um determinado serviço.

Quando uma nova área é construída, cada endereço atendido pode ser adicionado ao inventário.

Isso permite que os operadores vejam quais residências estão atendidas para um determinado serviço e quais residências ainda não estão atendidas.

Quando um cliente se inscreve para um serviço, o OmniCRM pode executar automaticamente uma qualificação de serviço contra o endereço do cliente, para ver se o endereço está no inventário de residências atendidas e quais serviços podem ser oferecidos.

Modelos de Inventário

O recurso **InventoryTemplate** permite a criação de qualquer número de tipos de itens com campos predefinidos. Esses modelos atuam como plantas que definem as características essenciais de diferentes itens, como:

- **Modems** com um endereço MAC (`itemtext1`) e um número de série (`itemtext2`).
- **Residências Atendidas** para um determinado serviço, com uma localização e status (por exemplo, atendida ou não atendida).
- **Números de Telefone** com um número principal (`itemtext1`) e uma localização geográfica (`itemtext2`).
- **Recursos Virtuais** como blocos de endereços IP, com identificadores mapeados através do modelo.

Cada modelo de inventário define até 20 campos de texto personalizáveis (`itemtext1` até `itemtext20`) com rótulos correspondentes (`itemtext1_label` até `itemtext20_label`) que descrevem o que cada campo representa. Por

exemplo, um modelo de Modem pode definir `itemtext1_label` como "Endereço MAC" e `itemtext2_label` como "Número de Série".

Os operadores podem personalizar os campos para cada tipo de item usando **InventoryTemplates**. Esses modelos permitem que os itens sejam categorizados e gerenciados de maneira estruturada, garantindo consistência em como os itens são rastreados.

Vinculando a Produtos:

Os nomes dos modelos de inventário são referenciados nas definições de produtos através do campo `inventory_items_list`. Ao provisionar um produto, o sistema exibe um seletor de inventário mostrando apenas itens que correspondem aos tipos de modelo exigidos.

Exemplo: Um produto com `inventory_items_list: "['SIM Card', 'Mobile Number']"` requer que dois modelos de inventário nomeados exatamente "SIM Card" e "Mobile Number" existam. Os nomes dos modelos são sensíveis a maiúsculas e minúsculas.

Para detalhes completos sobre como os modelos de inventário se conectam à provisão de produtos, veja `Product Lifecycle - Inventory Requirements <guide_product_lifecycle>`.

Criando Modelos de Inventário via a UI

Para criar um novo modelo de inventário:

1. Navegue até **Inventário** → **Modelos** no menu principal

2. Clique no botão **Adicionar Modelo**

3. Preencha os campos obrigatórios:

   Informações Básicas:

   - **Ícone** (opcional): Nome da classe do ícone para identificação visual (por exemplo, `fa-solid fa-sim-card`)
   - **Item** (obrigatório): O nome do modelo (deve corresponder exatamente ao que é usado em `inventory_items_list` para produtos)

   Informações de Custo (Obrigatório):

   - **Custo Atacadista** (obrigatório): Seu custo para comprar ou provisionar este tipo de item
   - **Custo Varejo** (obrigatório): Preço de varejo padrão se vendido separadamente para clientes

   Nota

   Os custos atacadista e varejo definidos aqui servem como valores padrão ao criar novos itens de inventário a partir deste modelo. Itens de inventário individuais podem ter custos diferentes, se necessário.

   Rótulos de Campo:

   - **Rótulo do Texto do Item 1** (obrigatório): Rótulo para o primeiro campo personalizável (padrão para "Número do Modelo")
     - Exemplos comuns: "ICCID" para cartões SIM, "Endere��o MAC" para modems, "Número de Telefone" para números
   - **Rótulo do Texto do Item 2** (obrigatório): Rótulo para o segundo campo personalizável (padrão para "Número de Série")
     - Exemplos comuns: "IMSI" para cartões SIM, "Número de Série" para hardware, "Região Geográfica" para números
   - **Rótulos do Texto do Item 3-20** (opcional): Rótulos de campo adicionais conforme necessário
     - Clique em **Adicionar Campo** para adicionar mais campos personalizados

        - Adicione apenas campos que você realmente usará para este tipo de item

    Configurações de Visibilidade:

    - **Permitir Dropdown para Funcionários**: Permitir que funcionários selecionem este tipo de inventário em dropdowns
    - **Permitir Dropdown para Clientes**: Permitir que clientes vejam/selecionem este tipo de inventário (portal do cliente)

4. Clique em **Salvar** para criar o modelo

Editando Modelos de Inventário

Para editar um modelo existente:

1. Navegue até **Inventário** → **Modelos**
2. Encontre o modelo na lista
3. Clique no botão **Editar**
4. Modifique os campos conforme necessário
5. Clique em **Salvar**

Aviso

Alterar rótulos de campo (por exemplo, `itemtext1_label`) afeta apenas novos itens criados após a alteração. Itens de inventário existentes mantêm seus dados, mas serão exibidos com os novos nomes de rótulo.

Cuidado

Os nomes dos modelos referenciados nos campos `inventory_items_list` de produtos são sensíveis a maiúsculas e minúsculas. Renomear um modelo quebrará o vínculo com produtos que usam o nome antigo.

Exemplos Comuns de Modelos

Modelo de Cartão SIM:

- Item: "SIM Card"
- Custo Atacadista: 2.50

- Custo Varejo: 10.00
- Rótulo do Texto do Item 1: "ICCID"
- Rótulo do Texto do Item 2: "IMSI"
- Rótulo do Texto do Item 3: "Tipo de SIM" (Físico/eSIM)

Modelo de Número Móvel:

- Item: "Mobile Number"
- Custo Atacadista: 1.00
- Custo Varejo: 0.00
- Rótulo do Texto do Item 1: "Número de Telefone"
- Rótulo do Texto do Item 2: "Região Geográfica"
- Rótulo do Texto do Item 3: "Tipo de Número" (Móvel/Fixo)

Modelo de CPE Wireless Fixo:

- Item: "Fixed Wireless CPE"
- Custo Atacadista: 250.00
- Custo Varejo: 450.00
- Rótulo do Texto do Item 1: "Endereço MAC"
- Rótulo do Texto do Item 2: "Número de Série"
- Rótulo do Texto do Item 3: "Versão do Firmware"
- Rótulo do Texto do Item 4: "Fabricante"
- Rótulo do Texto do Item 5: "Modelo"

Modelo de ONT GPON:

- Item: "GPON ONT"
- Custo Atacadista: 45.00
- Custo Varejo: 0.00 (incluído com o serviço)
- Rótulo do Texto do Item 1: "Número de Série"
- Rótulo do Texto do Item 2: "Endereço MAC"
- Rótulo do Texto do Item 3: "Localização PON"
- Rótulo do Texto do Item 4: "Modelo"

Criando e Gerenciando Itens de Inventário

Uma vez que um **InventoryTemplate** é definido, itens individuais de **Inventário** podem ser criados. Cada item de inventário representa uma instância específica de um tipo de item (por exemplo, um modem ou número de telefone específico) que pode ser:

- **Alocado a Clientes**: Itens são vinculados a clientes para provisão de serviços (por exemplo, atribuindo hardware para uma conexão de internet).
- **Rastreados para Estoque**: Os operadores podem monitorar o inventário disponível, como itens não vendidos ou não alocados.
- **Vendidos ou Descomissionados**: Uma vez vendidos, os itens são marcados com timestamps relevantes (por exemplo, `sold_date`) e não podem mais ser considerados estoque disponível.

Através deste sistema, o OmniCRM facilita o gerenciamento eficiente de estoque, ajuda a alocar recursos para clientes e fornece visibilidade detalhada sobre o status e a história de cada item.

Serviços podem ser vinculados a um determinado item de **Inventário**, permitindo fácil rastreamento de quais itens estão associados a quais clientes ou serviços.

Uma vez que um item de inventário foi atribuído a um cliente, os plays do Ansible podem atualizar o status e a história do item para refletir a alocação. Isso garante que os operadores tenham um registro preciso de quais itens

estão em uso e quais estão disponíveis para alocação, além de saber qual cliente está usando qual item.

Podemos visualizar itens alocados a um cliente a partir da página de perfil do cliente na aba **Inventário**.

Para itens de Inventário vinculados a um **Serviço**, podemos ver isso editando o serviço, para ver os itens de Inventário vinculados.

Campos do Item de Inventário

Cada item de inventário contém informações abrangentes organizadas em várias categorias:

Informações Básicas do Item

- **inventory_id** - Identificador único para o item de inventário (gerado automaticamente)
- **item** - Tipo de item (corresponde ao nome do Modelo de Inventário, por exemplo, "SIM Card", "Modem", "Phone Number")
- **inventory_template_id** - Link para o Modelo de Inventário que define este tipo de item
- **customer_id** - Se atribuído a um cliente, o ID do cliente (nulo)
- **service_id** - Se vinculado a um serviço específico, o ID do serviço (nulo)
- **item_location** - Localização física ou lógica do item:

    - Para itens físicos: prédio, armazém, localização da prateleira, nome da loja, etc.
    - Para itens virtuais: localização geográfica, localização do bloco IP, região de faixa de números, etc.
- **item_state** - Estado atual do item de inventário (valores enumerados):
    - **Novo** - Item novo, não utilizado
    - **Usado** - Anteriormente utilizado, mas funcional
    - **Uso Interno** - Alocado para testes internos ou uso de funcionários
    - **Atribuído** - Atualmente atribuído a um cliente ou serviço
    - **Danificado** - Não funcional, requer reparo ou descarte
    - **Fora de Serviço** - Temporariamente indisponível
    - **Perdido** - Item não pode ser localizado
    - **Roubado** - Item foi roubado

Campos Personalizáveis do Item (do Modelo)

O sistema de inventário suporta até 20 campos de texto personalizáveis cujo significado é definido pelo Modelo de Inventário:

- **itemtext1** - Primeiro campo personalizável (obrigatório, rótulo definido pelo `itemtext1_label` do modelo)
    - Exemplo: Para modems, pode ser "Endereço MAC"
    - Exemplo: Para cartões SIM, pode ser "ICCID"
    - Exemplo: Para números de telefone, pode ser "Número de Telefone"
- **itemtext2** até **itemtext20** - Campos personalizáveis adicionais (opcional, rótulos definidos pelo modelo)
    - Exemplo: itemtext2 para modems pode ser "Número de Série"
    - Exemplo: itemtext2 para cartões SIM pode ser "IMSI"
    - Exemplo: itemtext3 para modems pode ser "Versão do Firmware"

Cada Modelo de Inventário define quais desses campos são usados e o que representam através dos campos de rótulo correspondentes (`itemtext1_label`, `itemtext2_label`, etc.).

Informações de Custo

- **wholesale_cost** - Seu custo para comprar/provisionar este item (float)

- **retail_cost** - Preço cobrado ao cliente se vendido separadamente (float)
- **sold_date** - Timestamp quando o item foi vendido ou atribuído ao cliente

Endereço Físico (para Equipamentos e Locais de Rede)

Usado para rastrear locais de implantação física, particularmente para equipamentos de rede fixa (CPEs, ONTs, modems) ou residências atendidas:

- **address_line_1** - Endereço da rua, número do prédio, número da unidade
- **address_line_2** - Informações adicionais do endereço (suíte, apartamento, andar)
- **city** - Cidade ou município
- **state** - Estado, província ou região
- **zip_code** - Código postal/ZIP
- **country** - Nome do país

Localização Geográfica (Preenchido Automaticamente pela Web UI)

Ao criar itens de inventário via a Web UI com autocompletar de endereço, esses campos são preenchidos automaticamente:

- **google_maps_place_id** - Google Maps Place ID para o endereço
- **plus_code** - Google Maps Plus Code (Open Location Code) para localização precisa
- **latitude** - Latitude geográfica (graus decimais)
- **longitude** - Longitude geográfica (graus decimais)

Esses campos permitem:

- Mapeamento de locais de inventário em uma visualização de mapa
- Cálculos de proximidade para qualificação de serviço
- Análise de cobertura para planejamento de rede
- Otimização de rotas para despacho de técnicos de campo

Gerenciamento de Dispositivos e URLs de Acesso

O campo `management_url` fornece acesso rápido às interfaces de dispositivos e URLs de provisão:

- **management_url** - URL de acesso para o item de inventário

  - **Equipamento de Rede**: URL da interface web (por exemplo, `https://192.168.1.1` para roteadores, switches, ONTs, CPEs)
  - **Perfis eSIM**: Endereço LPA (Local Profile Assistant) para ativação de eSIM (por exemplo, `LPA:1$smdp.example.com$ACTIVATION-CODE-HERE`)
  - **Outros Casos de Uso**: Qualquer URL que precise ser facilmente acessível via dispositivo móvel

  Geração de QR Code

  Ao visualizar itens de inventário com um `management_url`, o sistema gera automaticamente um **QR code escaneável**:

  - **Visualização de detalhes do item de inventário**: QR code de 128x128 exibido ao lado da URL
  - **Tabela de inventário de serviço**: QR code de 64x64 mostrado para itens atribuídos
  - **Formato**: Tanto o QR code quanto o hyperlink clicável exibidos juntos

  Casos de Uso Comuns

  - **Técnicos de Rede**: Escaneie o QR code para acessar a interface de gerenciamento do dispositivo sem digitar endereços IP
  - **Ativação de eSIM**: Clientes escaneiam o QR code do CRM para instalar o perfil eSIM em seu dispositivo
  - **Autoatendimento do Cliente**: Fornecer acesso fácil à configuração do dispositivo ou portais de clientes

- **management_username** - Nome de usuário administrador para acesso ao dispositivo

- **management_password** - Senha de administrador para acesso ao dispositivo (criptografada em repouso)

Gerenciamento de Configuração

Para dispositivos com arquivos de configuração:

- **config_content** - Conteúdo completo do arquivo de configuração (armazenado como texto)
  - Útil para backup, versionamento e recuperação de desastres
  - Pode armazenar configurações de roteadores, configurações de switches, configurações de CPEs, etc.
- **config_file_path** - Caminho para o arquivo de configuração externo, se armazenado separadamente
  - Alternativa ao armazenamento da configuração completa no banco de dados
  - Caminho para compartilhamento de rede, repositório de controle de versão ou sistema de gerenciamento de configuração

Notas e Metadados

- **inventory_notes** - Notas livres sobre o item de inventário
  - Notas de instalação
  - Histórico de manutenção
  - Particularidades ou requisitos especiais de manuseio
  - Informações do fornecedor
  - Detalhes da garantia
- **created** - Timestamp quando o item de inventário foi criado no sistema (definido automaticamente)
- **last_modified** - Timestamp da última atualização do item de inventário (atualizado automaticamente)

Exemplos de Uso de Campos

Exemplo 1: Cartão SIM Móvel

```
{
  "inventory_id": 1001,
  "item": "SIM Card",
  "inventory_template_id": 5,
  "itemtext1": "89612345678901234 56",
  "itemtext2": "310120123456789",
  "itemtext3": "Physical",
  "item_location": "Warehouse A, Shelf 3",
  "item_state": "Assigned",
  "customer_id": 456,
  "service_id": 789,
  "wholesale_cost": 2.50,
  "retail_cost": 10.00,
  "sold_date": "2025-01-15T10:30:00Z",
  "inventory_notes": "Activated on 2025-01-15"
}
```


Exemplo 2: Perfil eSIM Móvel


```
{
  "inventory_id": 1002,
  "item": "eSIM",
  "inventory_template_id": 6,
  "itemtext1": "89612345678901234 57",
  "itemtext2": "310120123456790",
  "itemtext3": "eSIM",
  "item_location": "Virtual Inventory",
  "item_state": "Assigned",
  "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": "eSIM profile ready for activation"
}
```


Ao visualizar este item de inventário eSIM, a interface do usuário exibe um QR code contendo o endereço LPA. Os clientes escaneiam este QR code com seu

dispositivo móvel para instalar o perfil eSIM.

Exemplo 3: Equipamento de Cliente (CPE) - Wireless Fixo


```
{
  "inventory_id": 2001,
  "item": "Fixed Wireless CPE",
  "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": "Customer Site",
  "item_state": "Assigned",
  "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": "Australia",
  "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": "Installed 2025-01-10. Customer reports
excellent signal. Pointing: Azimuth 45°, Elevation 15°"
}
```


Exemplo 4: ONT GPON com Endereço Completo

```
{
  "inventory_id": 3001,
  "item": "GPON ONT",
  "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": "Customer Premises",
  "item_state": "Assigned",
  "customer_id": 321,
  "service_id": 654,
  "address_line_1": "456 Fiber Lane",
  "city": "Melbourne",
  "state": "VIC",
  "zip_code": "3000",
  "country": "Australia",
  "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": "# ONT Configuration\nwlan-ssid:
HomeNetwork\nwlan-password: encrypted...",
  "wholesale_cost": 45.00,
  "retail_cost": 0.00,
  "inventory_notes": "Provisioned 2025-01-20. Optical power:
-22dBm"
}
```


Nota

Ao visualizar itens de inventário com um `management_url` (como Exemplos 2, 3 e 4 acima), a interface do usuário exibe automaticamente:

- Um QR code escaneável contendo a URL ou endereço LPA
- Um hyperlink clicável (para URLs web)

Casos de Uso:

- **Ativação de eSIM** (Exemplo 2): Clientes escaneiam o QR code para instalar o perfil eSIM em seu dispositivo
- **Acesso a Equipamentos de Rede** (Exemplos 3 e 4): Técnicos escaneiam para acessar interfaces de gerenciamento de dispositivos sem digitar manualmente endereços IP

Exemplo 5: Número de Telefone (Inventário Virtual)


```
{
  "inventory_id": 4001,
  "item": "Phone Number",
  "inventory_template_id": 20,
  "itemtext1": "+61412345678",
  "itemtext2": "Melbourne",
  "itemtext3": "Mobile",
  "item_location": "Australia - VIC",
  "item_state": "Assigned",
  "customer_id": 555,
  "service_id": 888,
  "wholesale_cost": 1.00,
  "retail_cost": 0.00,
  "inventory_notes": "Ported from Telstra on 2025-01-05"
}
```


Estados do Item de Inventário Explicados

O campo `item_state` rastreia o ciclo de vida dos itens de inventário:

- **Novo** → **Atribuído** - Fluxo normal ao provisionar para um cliente
- **Atribuído** → **Usado** - Após a desativação do serviço, o item pode ser reutilizado
- **Novo** → **Uso Interno** - Alocado para testes, demonstrações ou uso de funcionários
- **Atribuído** → **Danificado** - Item falhou, requer RMA ou descarte
- **Qualquer Estado** → **Perdido** - Item não pode ser localizado (aciona auditoria)
- **Qualquer Estado** → **Roubado** - Item foi roubado (aciona relatório de segurança)
- **Danificado/Usado** → **Novo** - Após reforma ou reparo

Filtrar e pesquisar inventário por estado permite que os operadores:

- Rastrear estoque disponível (itens Novos)
- Identificar itens atribuídos a clientes (Atribuído)
- Encontrar itens disponíveis para reutilização (Usado)
- Monitorar problemas de equipamentos (Danificado, Fora de Serviço)
- Auditar itens faltantes (Perdido, Roubado)

Tags de Clientes

Tags são links codificados por cores que podem ser adicionados a um cliente para ajudar a categorizá-los, por exemplo, um cliente pode ter uma tag para "Chamado de Suporte Aberto" ou "Fatura Super Atrasada" ou "Chato".

Para armazenar metadados estruturados e dados personalizados de chave-valor, veja `Atributos do Cliente <administration_attributes>`.

As tags são exibidas como pílulas na página de perfil do cliente, e a cor da pílula é personalizável junto com o link.

Um caso de uso comum é marcar clientes que têm um chamado de suporte aberto, para que a equipe de suporte possa facilmente acessar o chamado aberto a partir da página de perfil do cliente.

As tags podem ser criadas no sistema por um administrador através da interface do usuário ou por sistemas de terceiros via API e podem ter datas de início e término, para que possam ser removidas automaticamente após um certo período.

Gerenciando Tags via a UI

Visualizando Tags de Clientes

Para visualizar as tags de um cliente:

1. Navegue até a página de visão geral do cliente
2. Clique na aba **Tags**
3. Você verá uma lista de todas as tags ativas para o cliente, mostrando:
    - Pré-visualização da tag com a cor configurada
    - Texto da tag
    - Data de ativação (quando a tag se torna visível)
    - Data de desativação (quando a tag será ocultada)
    - Link (se configurado)

Criando uma Nova Tag

Para criar uma nova tag para um cliente:

1. Navegue até a página de visão geral do cliente
2. Clique na aba **Tags**
3. Clique no botão **Adicionar Tag**
4. Preencha os campos obrigatórios:
    - **Texto da Tag** (obrigatório): O texto que será exibido na tag
    - **Cor da Tag** (obrigatório): Escolha uma cor usando o seletor de cores ou insira um código hex
    - **Link da Tag** (opcional): URL que será aberta quando a tag for clicada
    - **Data de Ativação** (obrigatório): Data quando a tag deve começar a ser exibida
    - **Data de Desativação** (obrigatório): Data quando a tag deve parar de ser exibida (padrão é 2099-01-01)
5. Visualize sua tag na seção de pré-visualização
6. Clique em **Criar Tag**

Editando uma Tag

Para editar uma tag existente:

1. Navegue até a página de visão geral do cliente
2. Clique na aba **Tags**
3. Encontre a tag que você deseja editar na lista
4. Clique no botão **Editar** (lápis)
5. Modifique os campos conforme necessário
6. Clique em **Atualizar Tag**

Excluindo uma Tag

Para excluir uma tag:

1. Navegue até a página de visão geral do cliente
2. Clique na aba **Tags**
3. Encontre a tag que você deseja excluir na lista
4. Clique no botão **Excluir** (lixeira)
5. Confirme a exclusão na janela pop-up

Referência de Campo de Tag

Integração com API

As tags também podem ser gerenciadas programaticamente via API:

Criar uma Tag:

```
PUT /crm/tag/
{
  "tag_text": "Cliente 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
}
```

Atualizar uma Tag:

```
PATCH /crm/tag/tag_id/{tag_id}
{
  "tag_text": "Texto da Tag Atualizado",
  "tag_hex_color": "FF0000"
}
```

Obter Tags por Cliente:

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

Excluir uma Tag:

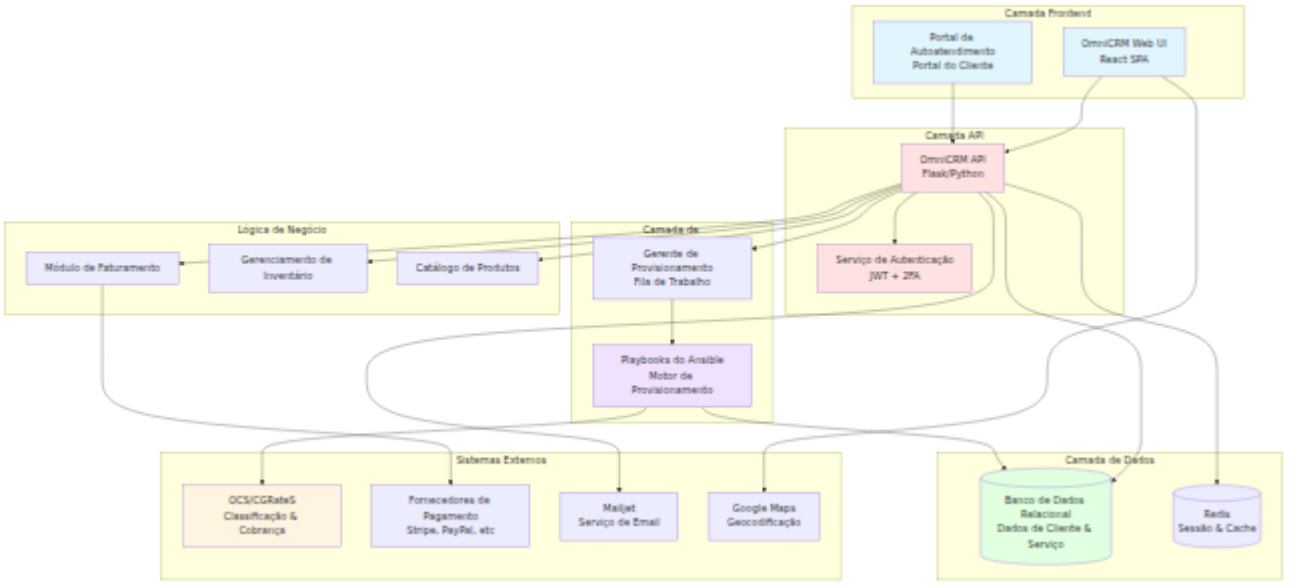
```
DELETE /crm/tag/tag_id/{tag_id}
```

Arquitetura do Sistema OmniCRM

Este documento fornece uma visão geral da arquitetura do sistema OmniCRM, incluindo relacionamentos entre componentes e fluxo de dados.

Visão Geral do Sistema em Alto Nível

OmniCRM é uma plataforma abrangente BSS/OSS que integra vários componentes-chave para fornecer gerenciamento completo de serviços para provedores de telecomunicações.



Componentes Principais

1. Aplicações Frontend

OmniCRM Web UI

- Aplicação de página única em React

- Interface para funcionários para gerenciamento de clientes, provisionamento de serviços, faturamento
- Atualizações de status de provisionamento em tempo real
- Controle de acesso baseado em funções

Portal de Autoatendimento

- Portal voltado para o cliente
- Gerenciamento de serviços e rastreamento de uso
- Visualização e pagamento de faturas
- Código compartilhado com a interface do funcionário, diferentes visualizações

2. Camada API

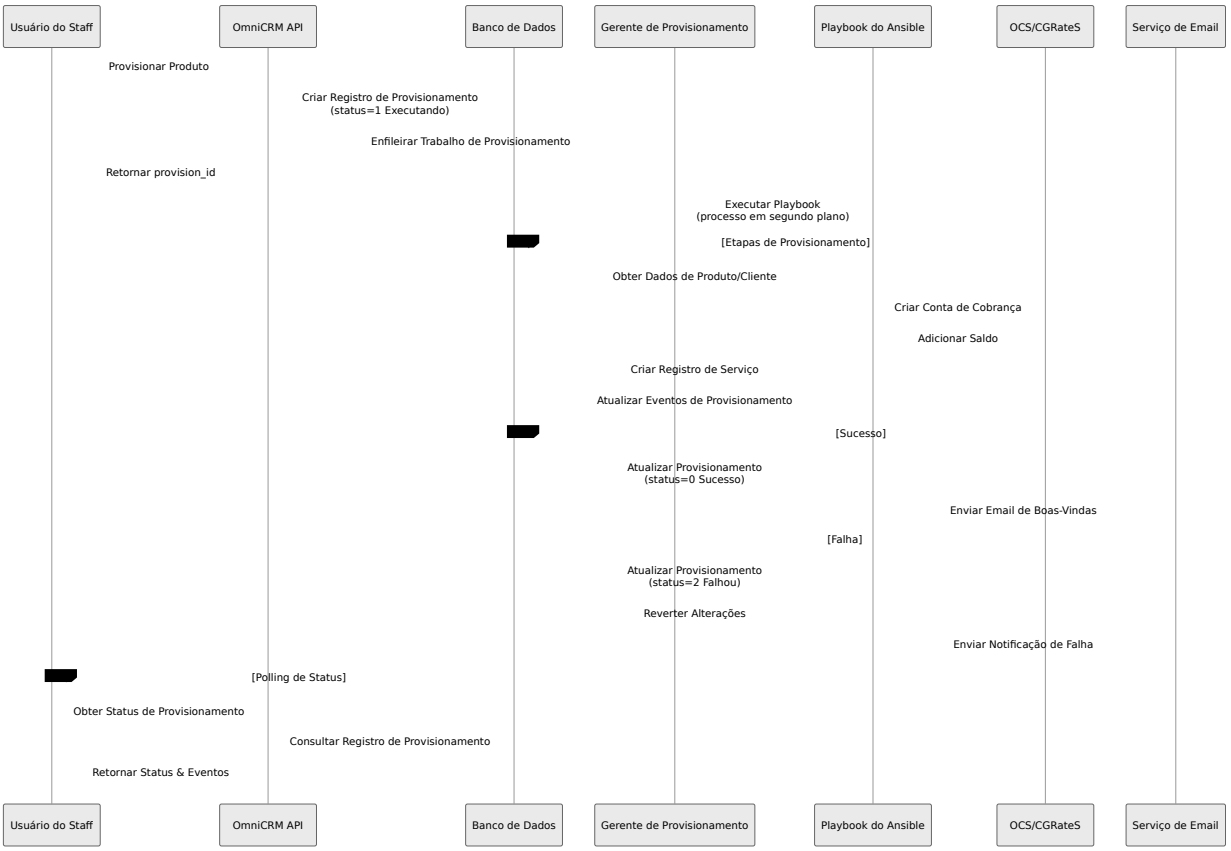
OmniCRM API (Flask/Python)

- API RESTful para todas as operações
- Documentação OpenAPI/Swagger
- Autenticação baseada em JWT
- Limitação de taxa e cache
- Suporte a WebSocket para atualizações em tempo real

Principais Rotas da API:

- `/crm/customer/*` - Gerenciamento de clientes
- `/crm/service/*` - Operações de serviço
- `/crm/product/*` - Catálogo de produtos
- `/crm/provision/*` - Operações de provisionamento
- `/crm/transaction/*` - Transações de faturamento
- `/crm/invoice/*` - Gerenciamento de faturas

3. Sistema de Provisionamento

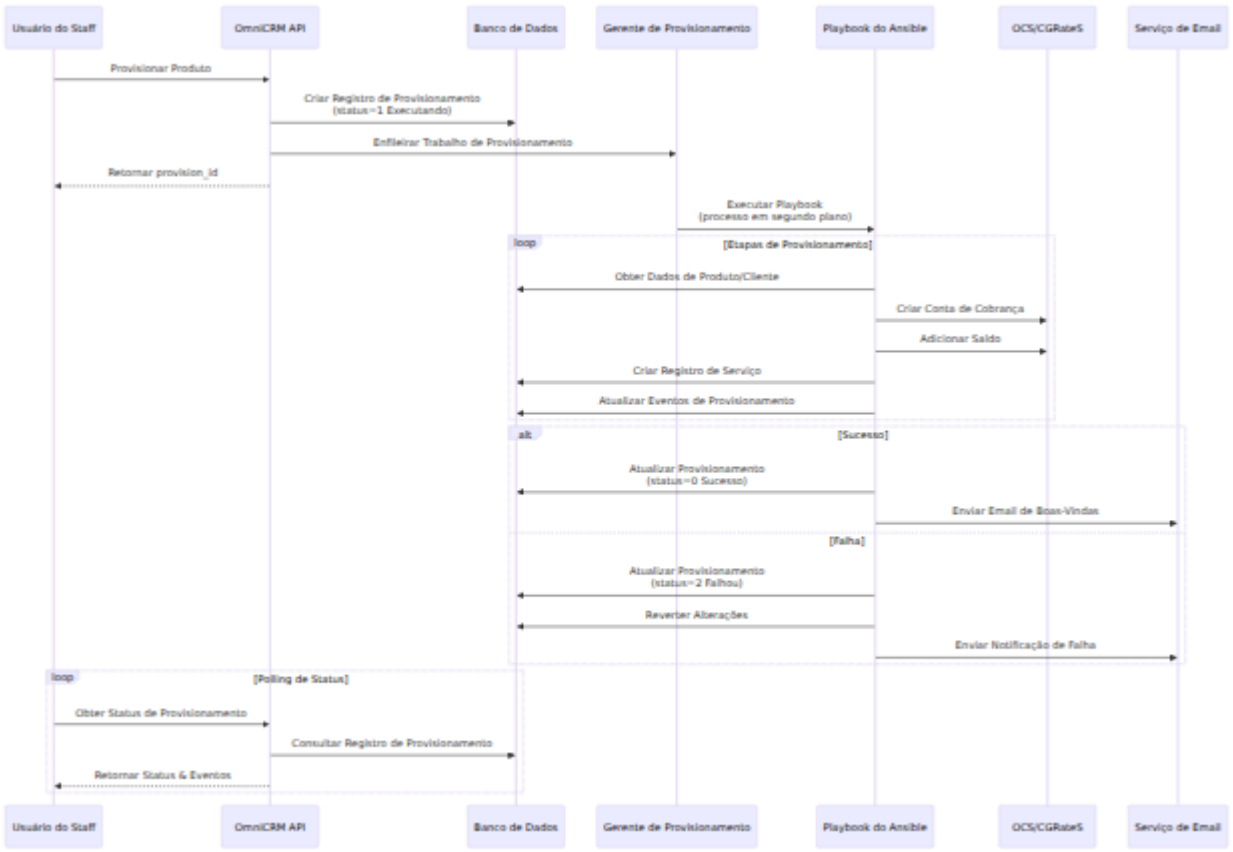


4. Faturamento & Classificação

Integração OCS/CGRateS

- Cobrança e classificação em tempo real
- Gerenciamento de contas
- Rastreamento de saldo (monetário, dados, voz, SMS)
- Planos de ação para cobranças recorrentes
- Notificações baseadas em limites

Fluxo de Trabalho de Faturamento:



5. Modelo de Dados

OmniCRM utiliza um banco de dados relacional com os seguintes modelos principais. Para diagramas visuais de relacionamento de entidades, veja Clientes, Contatos, Locais & Serviços.

Cliente & Entidades Relacionadas

Cliente - Entidade central representando uma empresa ou indivíduo

Campo	Tipo	Descrição
customer_id	Inteiro (PK)	Identificador único
customer_name	String	Nome da empresa ou indivíduo
customer_account_type	Enum	'Individual' ou 'Business'
customer_status	Enum	'Open', 'Closed', 'Suspended', 'Archived'
customer_payment_type	String	'prepaid' ou 'postpaid'
customer_enabled	Boolean	A conta está ativa
tax_identifier	String	Número VAT/GST
contract_start_date	DateTime	Início do contrato
contract_end_date	DateTime	Fim do contrato

Contato - Pessoas associadas a um cliente

Campo	Tipo	Descrição
contact_id	Inteiro (PK)	Identificador único
customer_id	Inteiro (FK)	Cliente pai
contact_firstname	String	Primeiro nome
contact_lastname	String	Último nome
contact_email	String	Endereço de email
contact_phone	String	Número de telefone
contact_types	String	'Primary', 'Billing', 'Technical'

Local - Locais físicos de entrega de serviços

Campo	Tipo	Descrição
site_id	Inteiro (PK)	Identificador único
customer_id	Inteiro (FK)	Cliente pai
site_name	String	Nome da localização
address_line_1	String	Endereço da rua
city, state, zip_code	String	Detalhes da localização
latitude, longitude	Float	Coordenadas GPS
google_maps_place_id	String	Referência do Google Maps
plus_code	String	Código de Localização Aberta

Modelos de Serviço & Produto

Serviço - Instâncias de serviço ativas

Campo	Tipo	Descrição
service_id	Inteiro (PK)	Identificador único
customer_id	Inteiro (FK)	Cliente pai
product_id	Inteiro (FK)	Modelo de produto
site_id	Inteiro (FK)	Local do serviço
service_name	String	Nome de exibição
service_uuid	String	Identificador do sistema de faturamento
service_status	Enum	Status atual
service_billed	Boolean	Gerar transações
wholesale_cost	Float	Custo do provedor
retail_cost	Float	Preço do cliente
bundled_parent	Inteiro (FK)	Serviço pai para pacotes

Produto - Modelos de ofertas de serviço

Campo	Tipo	Descrição
product_id	Inteiro (PK)	Identificador único
product_name	String	Nome de exibição
product_slug	String	Nome amigável para URL
category	Enum	'standalone', 'bundle', 'addon', 'promo'
provisioning_play	String	Nome do playbook do Ansible
provisioning_json_vars	JSON	Variáveis do playbook
inventory_items_list	String	Inventário necessário
retail_cost	Float	Preço mensal
retail_setup_cost	Float	Taxa única
enabled	Boolean	Disponível para venda

Modelos de Faturamento

Transação - Cobranças/créditos individuais

Campo	Tipo	Descrição
transaction_id	Inteiro (PK)	Identificador único
customer_id	Inteiro (FK)	Cliente pai
invoice_id	Inteiro (FK)	Fatura pai (opcional)
service_id	Inteiro (FK)	Serviço relacionado
title	String	Descrição da transação
retail_cost	Float	Montante
tax_percentage	Float	Taxa de imposto
tax_amount	Float	Imposto calculado
void	Boolean	Transação cancelada

Fatura - Transações agrupadas para faturamento

Campo	Tipo	Descrição
invoice_id	Inteiro (PK)	Identificador único
customer_id	Inteiro (FK)	Cliente pai
paid	Boolean	Pagamento recebido
void	Boolean	Fatura cancelada
payment_reference	String	ID da transação Stripe
start_date, end_date	Date	Período de faturamento
due_date	Date	Prazo de pagamento
retail_cost	Float	Montante total

Modelos de Inventário

Inventário - Ativos físicos e virtuais

Campo	Tipo	Descrição
inventory_id	Inteiro (PK)	Identificador único
customer_id	Inteiro (FK)	Cliente atribuído
service_id	Inteiro (FK)	Serviço vinculado
inventory_template_id	Inteiro (FK)	Modelo de tipo de item
item	String	Tipo de item (Cartão SIM, Roteador, etc.)
item_state	Enum	'Novo', 'Atribuído', 'Usado', etc.
itemtext1-20	String	Campos flexíveis
management_url	String	URL de administração do equipamento
config_content	Texto	Arquivo de configuração

Inventory_Template - Define a estrutura do item de inventário

Campo	Tipo	Descrição
inventory_template_id	Inteiro (PK)	Identificador único
item	String	Nome do modelo
itemtext1_label	String	Rótulo para o campo itemtext1
itemtext2_label	String	Rótulo para o campo itemtext2

Modelos de Provisionamento

Provision - Rastreamento de trabalhos de provisionamento

Campo	Tipo	Descrição
provision_id	Inteiro (PK)	Identificador único
product_id	Inteiro (FK)	Produto sendo provisionado
customer_id	Inteiro (FK)	Cliente alvo
service_id	Inteiro (FK)	Serviço criado/modificado
provisioning_play	String	Nome do playbook do Ansible
provisioning_status	Inteiro	0=Sucesso, 1=Executando, 2=Falhou

Provision_Event - Etapas individuais de provisionamento

Campo	Tipo	Descrição
provision_event_id	Inteiro (PK)	Identificador único
provision_id	Inteiro (FK)	Trabalho de provisionamento pai
event_name	String	Nome da tarefa
event_number	Inteiro	Número da sequência
provisioning_status	Inteiro	0=Sucesso, 1=Executando, 2=Falhou
provisioning_result_json	JSON	Saída completa da tarefa

Modelos de Usuário & Segurança

Usuário - Contas de usuário

Campo	Tipo	Descrição
id	Inteiro (PK)	Identificador único
username	String	Nome de usuário para login
email	String	Endereço de email
email_verified	Boolean	Email confirmado
is_2fa_enabled	Boolean	Autenticação de dois fatores habilitada
totp_secret	String	Chave secreta TOTP

Função - Funções de usuário

Campo	Tipo	Descrição
id	Inteiro (PK)	Identificador único
name	String	Nome da função
description	String	Descrição da função

Permissão - Permissões granulares

Campo	Tipo	Descrição
id	Inteiro (PK)	Identificador único
name	String	Nome da permissão (recurso.ação)
description	String	Descrição da permissão

Relacionamentos:

- Usuários têm muitas Funções (muitos-para-muitos)
- Funções têm muitas Permissões (muitos-para-muitos)
- Usuários podem vincular a um Contato (para acesso ao portal do cliente)

Pontos de Integração

Gateway de Pagamento Stripe

- Métodos de pagamento tokenizados
- Armazenamento de cartões em conformidade com PCI
- Pagamento automático de faturas
- Processamento de reembolsos
- Notificações de cartões expirando

Serviço de Email Mailjet

- Emails transacionais (faturas, boas-vindas, notificações)
- Sincronização de contatos
- Gerenciamento de modelos
- Rastreamento de entrega

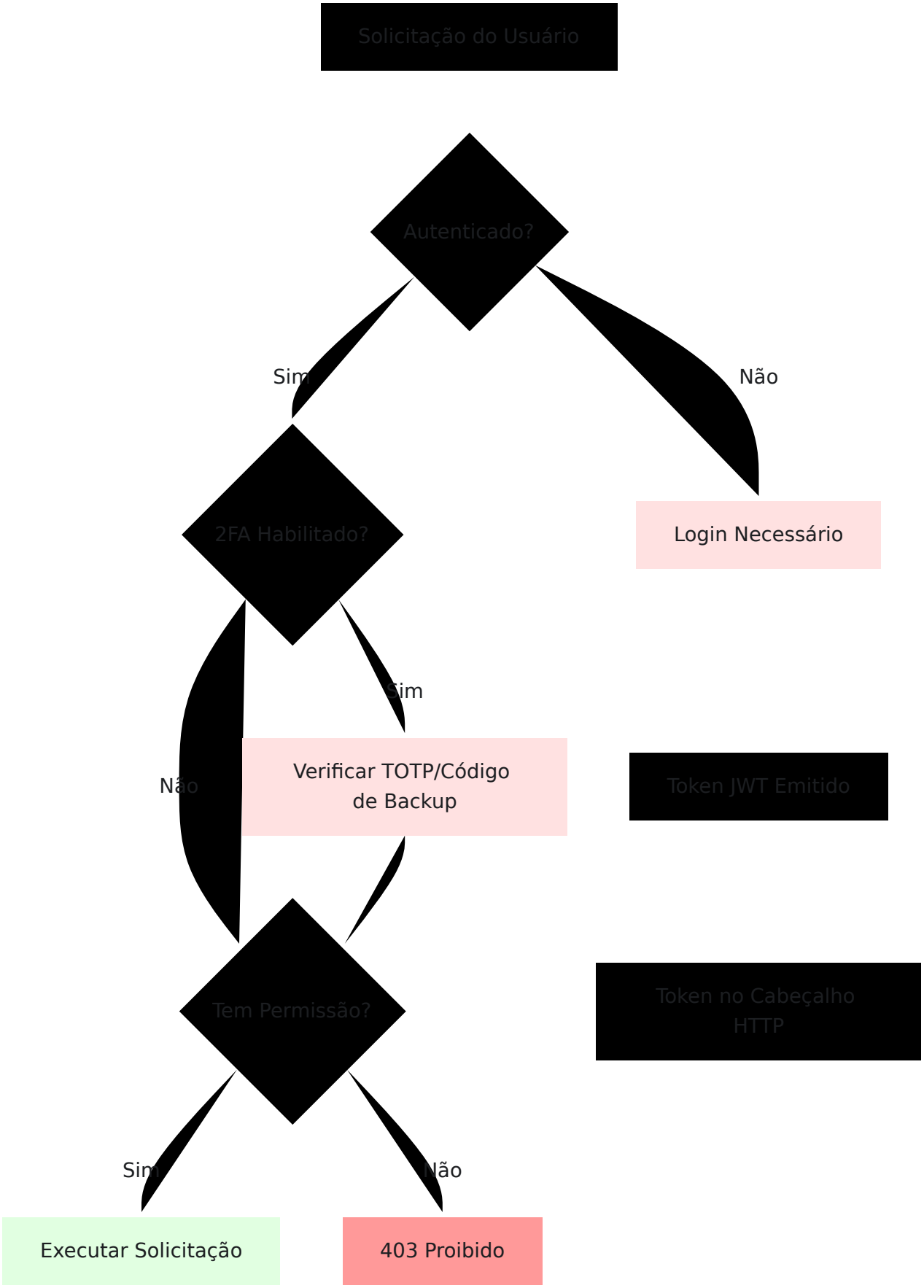
Google Maps

- Autocompletar endereço
- Geocodificação e geocodificação reversa
- Geração de Código Plus
- Mapeamento de localização do site

OCS/CGRateS

- Provisionamento de contas
- Classificação em tempo real
- Gerenciamento de saldo
- Processamento de CDR
- Planos de ação e cronogramas

Arquitetura de Segurança

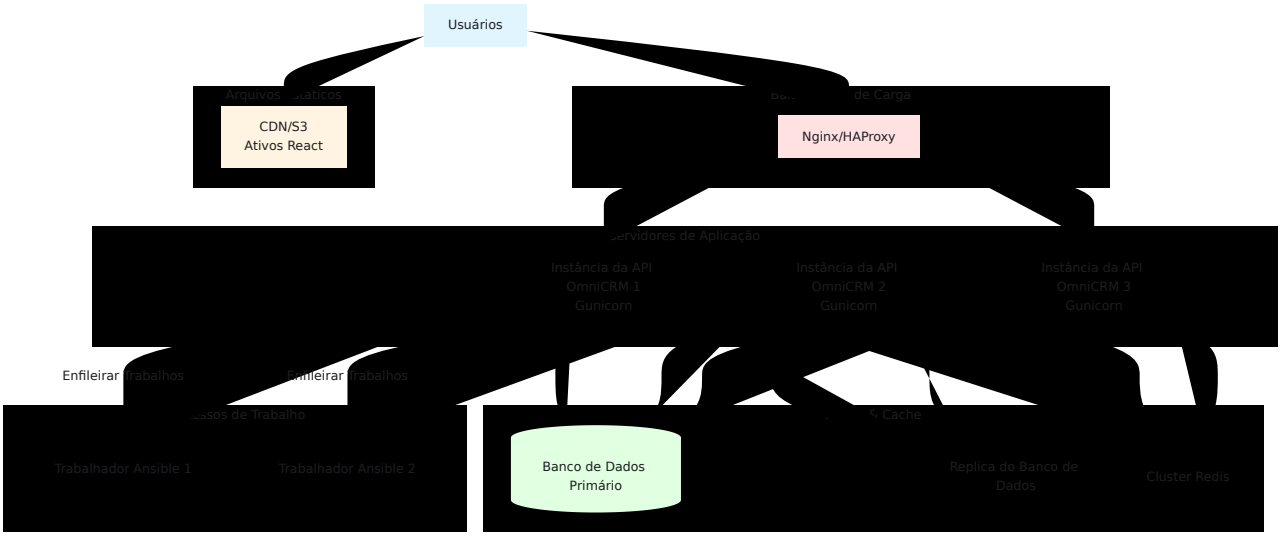


Recursos de Segurança:

- Autenticação baseada em JWT
- Autenticação de dois fatores (TOTP)
- Controle de acesso baseado em funções (RBAC)
- Acesso a recursos baseado em permissões
- Gerenciamento de sessão com Redis
- Hashing de senhas (bcrypt)
- Verificação de email
- Registro de auditoria via Registro de Atividades

Arquitetura de Implantação

Configuração de Produção Recomendada:



Stack Tecnológico

Backend:

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

- Ansible para automação

Frontend:

- React
- Gerenciamento de estado
- React Router
- Axios para chamadas de API

Banco de Dados:

- Banco de dados relacional (armazenamento de dados primário)
- Redis (cache & sessões)

Serviços Externos:

- CGRateS (faturamento/classificação)
- Stripe (pagamentos)
- Mailjet (email)
- Google Maps (geocodificação)

Considerações de Escalabilidade

Escalonamento Horizontal:

- Design de API sem estado permite múltiplas instâncias
- Balanceador de carga distribui solicitações
- Redis para estado de sessão compartilhado

Escalonamento de Banco de Dados:

- Réplicas de leitura para consultas de relatório
- Pooling de conexões
- Otimização de consultas e indexação

Provisionamento em Escala:

- Processamento de trabalhos em segundo plano
- Múltiplos trabalhadores Ansible
- Gerenciamento de fila de trabalhos
- Lógica de repetição para provisionamentos falhados

Monitoramento & Observabilidade

OmniCRM fornece métricas abrangentes baseadas em Prometheus para monitorar todos os aspectos do sistema. Para detalhes completos, veja Monitoramento & Métricas.

Métricas Principais:

- Tempos de resposta da API e taxas de solicitação
- Taxas de sucesso/falha de provisionamento e duração do trabalho
- Desempenho de consulta do banco de dados e uso do pool de conexões
- Saúde da integração externa (OCS, Stripe, Mailjet)
- Execução de trabalhos em segundo plano e erros

Registro:

- Logs de aplicação (Flask)
- Logs de provisionamento (saída do Ansible)
- Registro de atividades (trilha de auditoria)
- Rastreamento de erros e alertas

Endpoint de Métricas: Todas as métricas são expostas em `/crm/metrics` no formato Prometheus. Veja Monitoramento & Métricas para configuração de coleta, regras de alerta e exemplos de painel.

Documentação Relacionada

- Playbooks do Ansible - Automação de provisionamento
- Sistema de Provisionamento - Detalhes do fluxo de trabalho
- Produtos e Serviços - Arquitetura do produto

- Documentação da API - Referência da API
- RBAC - Segurança e permissões

Fluxos de Autenticação e Controles de Administração

OmniCRM fornece recursos de autenticação abrangentes, incluindo login, autenticação de dois fatores (2FA), gerenciamento de senhas e controles administrativos para gerenciar a segurança do usuário. Este guia se concentra nos fluxos de trabalho da interface do usuário para usuários finais e administradores.

Veja também: `Self-Care Portal <self_care_portal>` para login de clientes e acesso ao portal, `RBAC <rbac>` para permissões de funcionários.

Visão Geral

A autenticação do OmniCRM inclui:

- **Login com Email/Senha** - Autenticação padrão baseada em credenciais
- **Autenticação de Dois Fatores (2FA)** - Segundo fator baseado em TOTP opcional
- **Lembrar-me** - Sessão estendida por até 30 dias
- **Redefinição de Senha** - Recuperação de senha autoatendida via email
- **Controles Administrativos** - Ferramentas administrativas para redefinir 2FA e senhas
- **Logins Sociais** - Integração opcional com Google, Apple, Facebook (se habilitado)
- **Navegação Baseada em Funções** - Roteamento automático com base nas funções dos usuários

Fluxo de Login

A página de login é o ponto de entrada para todos os usuários (funcionários e clientes).

Login Padrão

Processo de Login:

1. Insira o **endereço de email** (email do funcionário ou cliente)
2. Insira a **senha**
3. Opcional: Marque **"Lembrar-me por 30 dias"** para sessão estendida
4. Clique em **"Login"**

O que acontece a seguir:

- **Sem 2FA:** Usuário logado imediatamente, navegado com base na função:
  - **Clientes** → Portal de Autoatendimento (`/self-care`)
  - **Funcionários/Admins** → Painel de clientes (`/customers`)
  - **Modo CBC** → Interface de Transmissão de Célula (`/create-cell-broadcast`)
- **Com 2FA Habilitado:** Redirecionado para a tela de verificação de 2FA

Recurso Lembrar-me:

Quando habilitado, a sessão persiste por **30 dias** em vez de expirar quando o navegador é fechado. Usa cookies seguros apenas para HTTP.

Mostrar/Ocultar Senha:

Clique no **ícone de olho (□)** para alternar a visibilidade da senha.

Login com 2FA

Se o usuário tiver 2FA habilitado, após inserir email/senha, a tela de desafio 2FA aparece:

Usando Código do Autenticador:

1. Abra o aplicativo autenticador (Google Authenticator, Authy, etc.)
2. Encontre a entrada do OmniCRM
3. Insira o código de 6 dígitos
4. O código é enviado automaticamente quando todos os 6 dígitos são inseridos
5. Se válido, o usuário é logado e navegado para o painel apropriado

Usando Código de Recuperação:

Se o aplicativo autenticador não estiver disponível:

1. Clique na aba **"Código de Recuperação"**

2. Insira um dos seus códigos de backup salvos (por exemplo, `3fa5b9c2`)
3. Clique em **"Verificar"**
4. O código é consumido (pode ser usado apenas uma vez)

Cancelar:

Clique em **"Cancelar"** para retornar à página de login.

Logins Sociais (Opcional)

Se habilitado (`REACT_APP_ALLOW_SOCIAL_LOGINS=yes`), botões de login social aparecem:

> [Entrar com Google] [• Entrar com Apple] [Entrar com Facebook]

Clique em qualquer botão para autenticar via esse provedor. Atualmente exibe a mensagem "em breve" (implementação de login social em andamento).

Link de Esqueci a Senha

Clique no link **"Esqueci a senha?"** para iniciar o fluxo de redefinição de senha.

Configuração da Autenticação de Dois Fatores (2FA)

Os usuários podem habilitar 2FA para aumentar a segurança da conta. O 2FA usa TOTP (Senha Única Baseada em Tempo) compatível com aplicativos autenticadores padrão.

Acessando a Configuração de 2FA

A partir do perfil do usuário ou configurações:

Nota para Clientes:

Usuários com função de cliente não veem os prompts de 2FA. O 2FA é tipicamente exigido apenas para funcionários e usuários administrativos.

Passo 1: Confirmar Senha

> Senha Atual
>
> [Cancelar] [Continuar]

Insira sua senha atual para prosseguir. Isso verifica sua identidade antes de habilitar o 2FA.

Passo 2: Escanear Código QR

> [Cancelar] [Confirmar]

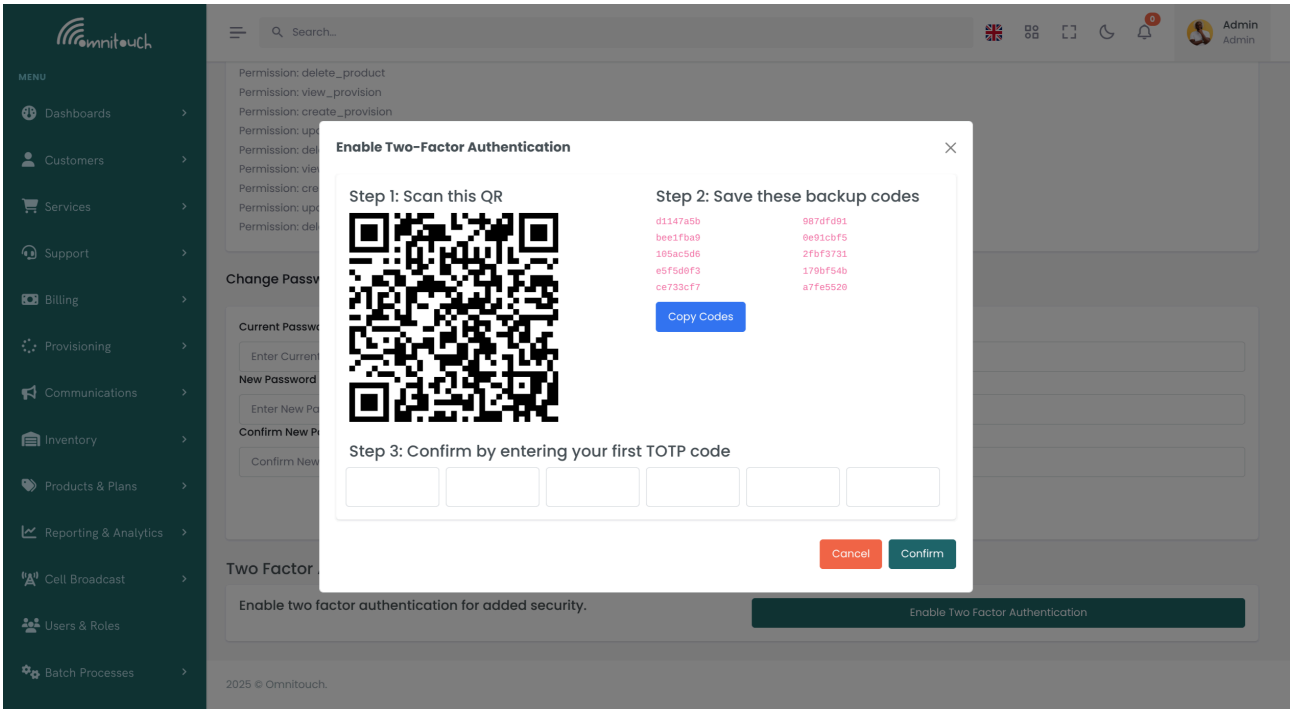
Instruções de Configuração:

1. **Baixar Aplicativo Autenticador** (se você não tiver um):
   - iOS: Apple App Store → "Google Authenticator"
   - Android: Google Play → "Google Authenticator"
   - Alternativas: Authy, Microsoft Authenticator, 1Password
2. **Escanear Código QR:**
   - Abra o aplicativo autenticador
   - Toque em "+" ou "Adicionar conta"
   - Escolha "Escanear código QR"
   - Aponte a câmera para o código QR na tela
   - O aplicativo adiciona a entrada "OmniCRM" com o código de 6 dígitos
3. **Salvar Códigos de Backup:**
   - **CRÍTICO:** Anote ou copie esses 8 códigos
   - Armazene em local seguro (gerenciador de senhas, cofre, etc.)
   - Cada código é de uso único
   - Usado se você perder o acesso ao aplicativo autenticador
   - Clique em **"Copiar Códigos"** para copiar todos os códigos para a área de transferência
4. **Verificar Configuração:**
   - Insira o código de 6 dígitos atual do aplicativo autenticador
   - Clique em **"Confirmar"**
   - Se válido, o 2FA agora está habilitado

Passo 3: 2FA Habilitado

Uma mensagem de sucesso aparece:

A partir de agora, o login requer tanto a senha quanto o código de 2FA.



Fluxo de Redefinição de Senha (Autoatendimento)

Usuários que esquecem sua senha podem redefini-la via email.

Passo 1: Solicitar Link de Redefinição

A partir da página de login, clique em **"Esqueci a senha?"**

1. Insira o **endereço de email**
2. Clique em **"Enviar Link de Redefinição"**

O que acontece:

- O sistema verifica se o email existe no banco de dados
- Se encontrado, envia um email de redefinição de senha via Mailjet
- O email contém um link de redefinição com tempo limitado (tipicamente 1 hora de expiração)
- Mensagem de sucesso aparece: "Instruções de redefinição foram enviadas para seu email"

Se o Email Não For Encontrado:

Mensagem de erro: "Nenhuma conta encontrada com esse endereço de email"

Passo 2: Verificar Email

O usuário recebe um email com assunto como:

> Olá [Nome],
>
> Você solicitou uma redefinição de senha para sua conta do OmniCRM.
>
> Clique no link abaixo para redefinir sua senha:
> <https://yourcompany.com/reset-password/abc123token456>
>
> Este link expira em 1 hora.
>
> Se você não solicitou isso, ignore este email.

Clique no link de redefinição para prosseguir.

Passo 3: Definir Nova Senha

O link de redefinição abre a página de criação de senha:

> □ (ícone de cadeado)
>
> Senha
>
> Confirmar Senha
>
> [Redefinir Senha]

1. Insira a **nova senha**
2. Reinsira no campo **Confirmar Senha**
3. Clique em **"Redefinir Senha"**

Requisitos da Senha:

- Comprimento mínimo (tipicamente 8+ caracteres)
- As senhas devem corresponder

Sucesso:

- Mensagem de sucesso: "A senha foi redefinida com sucesso"
- Redirecionado automaticamente para a página de login
- O usuário pode agora fazer login com a nova senha

Token Expirado/Inválido:

Se o link de redefinição estiver expirado ou inválido:

> [Solicitar Novo Link de Redefinição]

Controles Administrativos para Gerenciamento de Usuários

Administradores com permissões apropriadas podem gerenciar as configurações de autenticação do usuário a partir da interface de Gerenciamento de Usuários.

Acessando o Gerenciamento de Usuários

Exibe uma tabela de todos os usuários com botões de ação.

> Nome Email Telefone Ações John Smith <john@example.com> +44 123... ✏ □ □ □ Jane Doe <jane@example.com> +44 456... ✏ □ □ Bob Wilson <bob@example.com> +44 789... ✏ □ □ □ ✉

Ícones de Ação:

- ✏ **Editar** - Modificar detalhes do usuário, funções, permissões
- □ **Excluir** - Remover conta do usuário
- □ **Redefinir Senha** - Gerar senha temporária
- □ **Redefinir 2FA** - Desativar 2FA para o usuário (apenas mostrado se 2FA habilitado)
- ✉ **Enviar Email de Boas-Vindas** - Reenviar email de boas-vindas (apenas mostrado se o usuário nunca fez login)

Admin: Redefinir Senha do Usuário

Quando o usuário esquece a senha e o administrador precisa ajudar:

Passo 1: Clique no Ícone de Redefinição de Senha (□)

A modal de confirmação aparece:

> Você tem certeza de que deseja redefinir a senha de:
>
> Usuário: John Smith (<john@example.com>)
>
> Uma senha temporária será gerada e exibida. O usuário deve alterar essa senha no próximo login.
>
> [Cancelar] [Redefinir Senha]

Passo 2: Confirmar Redefinição

Clique em **"Redefinir Senha"**. O sistema gera uma senha temporária segura.

Passo 3: Senha Temporária Exibida

> Senha temporária para John Smith:
>
> [Copiar Senha]
>
> ⚠ IMPORTANTE: • Envie esta senha ao usuário via canal seguro • Não envie por email ou mensagem não segura • O usuário será forçado a mudar a senha no próximo login
>
> [Fechar]

Ação do Administrador:

- Copie a senha temporária
- Ligue para o usuário ou comunique-se por meio seguro
- Forneça a senha temporária verbalmente
- Instrua o usuário a fazer login e alterar a senha

Experiência do Usuário:

Quando o usuário faz login com a senha temporária:

1. O login é bem-sucedido
2. Redirecionado imediatamente para a tela "Alterar Senha"
3. Deve definir uma nova senha antes de acessar o sistema

4. Não pode pular a alteração da senha

Admin: Redefinir 2FA do Usuário

Quando o usuário perde o acesso ao aplicativo autenticador e todos os códigos de backup:

Passo 1: Clique no Ícone de Redefinição de 2FA (□)

Apenas aparece para usuários com 2FA atualmente habilitado.

A modal de confirmação aparece:

Passo 2: Confirmar Redefinição

Clique em **"Redefinir 2FA"**

Passo 3: Confirmação

Mensagem de sucesso:

> John Smith agora pode fazer login apenas com sua senha. Ele pode reabilitar 2FA em suas configurações de usuário.

Experiência do Usuário:

- O usuário agora pode fazer login apenas com a senha (sem código 2FA necessário)
- O ícone de escudo 2FA (□) desaparece da linha do usuário na tabela do administrador
- O usuário pode reabilitar voluntariamente o 2FA em suas configurações

Nota de Segurança Importante:

Antes de redefinir o 2FA, o administrador deve:

1. Verificar a identidade do usuário por meios alternativos:
    - Verificação de ID do governo
    - Perguntas de segurança
    - Verificação de transação recente
    - Verificação em pessoa (se aplicável)
2. Documentar a redefinição nas notas do cliente
3. Informar o usuário para reabilitar o 2FA após recuperar o acesso

Admin: Enviar Email de Boas-Vindas

Para usuários que não receberam ou perderam seu email de boas-vindas:

Quando Disponível:

O ícone de avião de papel (✉) aparece apenas para usuários que **nunca fizeram login** (`login_count = 0`).

Clique no Ícone de Enviar Email de Boas-Vindas (✉)

> Enviar email de boas-vindas para:
>
> Usuário: Bob Wilson (<bob@example.com>)
>
> O email incluirá: • Mensagem de boas-vindas • Instruções de login • Link para definir a senha inicial (se aplicável) • Informações de contato de suporte
>
> [Cancelar] [Enviar Email]

Clique em **"Enviar Email"**

Mensagem de sucesso:

Email Enviado via Mailjet:

Usa o modelo: `api_crmCommunicationUserWelcome`

Admin: Editar Usuário

Clique no **ícone Editar (✏)** para modificar os detalhes do usuário:

> Primeiro Nome
>
> Último Nome
>
> Email
>
> Número de Telefone
>
> Funções ☑ admin ☐ customer_service_agent_1 ☐ customer
>
> [Cancelar] [Salvar Alterações]

Campos Editáveis:

- Nome, email, telefone
- **Funções** - Atribuir/remover funções (afeta permissões)
- Status ativo/inativo

Admin: Excluir Usuário

Clique no **ícone Excluir (🗑)** para remover o usuário:

> Você tem certeza de que deseja excluir:
>
> Usuário: John Smith (<john@example.com>)
>
> ⚠ AVISO: Esta ação não pode ser desfeita.

> Isso excluirá permanentemente: • Conta e credenciais do usuário • Configurações de 2FA • Histórico de sessão
>
> Dados e transações do cliente NÃO serão excluídos.
>
> [Cancelar] [Excluir Usuário]

Clique em **"Excluir Usuário"** para confirmar.

Mensagem de sucesso:

Melhores Práticas

Para Usuários Finais

Segurança do Login:

- Use senhas fortes e únicas
- Habilite "Lembrar-me" apenas em dispositivos pessoais
- Sempre faça logout em computadores compartilhados
- Habilite 2FA para segurança adicional

Gerenciamento de 2FA:

- Salve os códigos de backup imediatamente após habilitar o 2FA
- Armazene os códigos em um gerenciador de senhas ou local seguro
- Teste um código de backup para garantir que funcionem
- Regere os códigos de backup se você usar vários
- Contate o administrador se perder tanto o autenticador quanto os códigos de backup

Gerenciamento de Senhas:

- Use um gerenciador de senhas para gerar e armazenar senhas
- Nunca compartilhe senhas por email ou mensagem
- Altere a senha se suspeitar de comprometimento
- Use uma senha única para o OmniCRM (não reutilize senhas)

Para Administradores

Gerenciamento de Segurança do Usuário:

- Verifique a identidade do usuário antes de redefinir 2FA ou senhas
- Nunca envie senhas temporárias por email
- Documente todas as redefinições de segurança nas notas do usuário
- Incentive a equipe a habilitar 2FA
- Monitore padrões de login incomuns

Redefinições de Senha:

- Comunique senhas temporárias apenas por telefone ou pessoalmente
- Gere senhas temporárias fortes (o sistema faz isso automaticamente)
- Certifique-se de que o usuário altere a senha no primeiro login
- Não redefina senhas desnecessariamente - use o fluxo de redefinição por email quando possível

Redefinições de 2FA:

- Trate as redefinições de 2FA como ações de alta segurança
- Verifique a identidade através de múltiplos canais antes de redefinir
- Documente o motivo da redefinição
- Incentive o usuário a reabilitar 2FA imediatamente após recuperar o acesso
- Considere exigir 2FA para todos os usuários administrativos

Gerenciamento de Usuários:

- Revise regularmente a lista de usuários em busca de contas inativas
- Remova usuários que deixaram a organização
- Garanta atribuições de função apropriadas
- Monitore usuários que nunca fizeram login
- Audite permissões de usuários trimestralmente

Resolução de Problemas

Erro "Email ou senha inválidos"

- **Causa:** Credenciais incorretas
- **Solução:**
  - Verifique se o endereço de email está correto
  - Verifique se o caps lock está desligado
  - Tente redefinir a senha se esquecida
  - Contate o administrador se a conta estiver bloqueada

Código 2FA não aceito

- **Causa:** Problema de sincronização de tempo ou código incorreto
- **Solução:**
  - Certifique-se de que o horário do dispositivo está correto (Configurações → Data e Hora → Automático)
  - Aguarde o código atualizar (os códigos mudam a cada 30 segundos)
  - Tente o próximo código que aparecer
  - Use um código de backup se o autenticador não estiver funcionando
  - Contate o administrador para redefinir 2FA se tudo mais falhar

"Lembrar-me" não funcionando

- **Causa:** Cookies desativados ou limpos
- **Solução:**
  - Ative cookies nas configurações do navegador
  - Não limpe cookies ao fechar o navegador
  - Desative extensões de privacidade para o domínio do OmniCRM
  - Tente um navegador diferente

Email de redefinição de senha não recebido

- **Causa:** Email não enviado, filtro de spam ou email incorreto
- **Solução:**
  - Verifique a pasta de spam/lixo eletrônico

    - Verifique se o endereço de email está correto
    - Aguarde 5-10 minutos (a entrega de email pode ser atrasada)
    - Verifique se a integração com Mailjet está funcionando (administrador)
    - Contate o administrador para redefinição manual da senha

Link de redefinição de senha expirado

- **Causa:** Token expirado (tipicamente 1 hora)
- **Solução:**
    - Solicite nova redefinição de senha
    - Verifique o email e clique no link prontamente
    - Contate o administrador se houver problemas repetidos

Não é possível habilitar 2FA (senha incorreta)

- **Causa:** Senha atual inserida incorretamente
- **Solução:**
    - Verifique a senha atual
    - Redefina a senha primeiro se estiver incerto
    - Contate o administrador para assistência

Perdeu o aplicativo autenticador e os códigos de backup

- **Causa:** Telefone perdido/resetado, códigos de backup não salvos
- **Solução:**
    - Contate o administrador imediatamente
    - O administrador verificará a identidade e redefinirá o 2FA
    - Faça login apenas com a senha após a redefinição
    - Rehabilite o 2FA e SALVE os códigos de backup desta vez

Admin: "Falha ao redefinir 2FA" erro

- **Causa:** Permissões insuficientes
- **Solução:**
    - Certifique-se de que você tem a função de administrador
    - Verifique as permissões da API
    - Contate o administrador do sistema

Admin: Senha temporária não gerada

- **Causa:** Erro de API ou problema de permissões
- **Solução:**
  - Atualize a página e tente novamente
  - Verifique as permissões do administrador
  - Verifique os logs da API em busca de erros
  - Certifique-se de que o banco de dados está acessível

Considerações de Segurança

Gerenciamento de Sessões:

- Sessões expiram após um período de inatividade
- "Lembrar-me" estende a sessão para 30 dias
- Sessões armazenadas como cookies apenas para HTTP (não acessíveis via JavaScript)
- A flag segura garante que os cookies sejam enviados apenas via HTTPS

Segurança da Senha:

- Senhas criptografadas usando algoritmos padrão da indústria
- Senhas em texto simples nunca armazenadas
- Senhas temporárias expiram automaticamente após o primeiro uso
- Tentativas de login falhadas são rastreadas (potencial limitação de taxa)

Segurança do 2FA:

- Segredos TOTP criptografados no banco de dados
- Códigos QR gerados do lado do cliente quando possível
- Códigos de backup criptografados antes do armazenamento
- Cada código de backup é de uso único

Ações do Administrador:

- Redefinições de 2FA registradas no log de atividades

- Redefinições de senha criam trilha de auditoria
- Ações administrativas requerem permissões de função apropriadas
- Endereços IP registrados para eventos de segurança

Documentação Relacionada

- `2fa` - Referência detalhada da API de 2FA (focada em API)
- `rbac` - Controle de acesso baseado em funções e permissões
- `administration_configuration` - Configuração de email do Mailjet para redefinição de senha
- `integrations_mailjet` - Configuração de modelo de email
- `customer_care` - Portal de Autoatendimento para clientes

Criar um Cliente

0fT52ZvoZBE

Clientes, Contatos, Sites e Serviços

Temos um modelo simples de um **Cliente** que pode ter múltiplos **Contatos** e múltiplos **Sites**, **Serviços**, etc.

Um **Cliente** é uma empresa ou indivíduo que tem um relacionamento conosco, para quem enviamos uma fatura / conta.

Um **Contato** é uma pessoa que trabalha com o cliente, para um indivíduo, provavelmente é a mesma pessoa que o próprio cliente, uma única pessoa, mas podemos ter membros da família ou outros contatos, e cada contato tem um tipo, por exemplo, um contato de cobrança, um contato técnico, etc., que influencia como lidamos com o contato.

Um **Site** é um local físico onde entregamos serviços, pode ser uma casa, escritório ou outro local. Isso nos permite ter múltiplos sites para um único cliente, por exemplo, um cliente com múltiplos escritórios, e saber quais serviços estão associados a qual site.

Um **Serviço** é algo pelo qual cobramos um cliente, pode ser um serviço de internet residencial, serviço móvel, ou até mesmo serviços abstratos como locação de uma sub-rede ou fornecimento de eletricidade medida para um rack. Cada serviço está vinculado a um cliente e a um site, e pode ter múltiplas cobranças associadas a ele.

Os clientes também têm um `Registro de Atividades <csa_activity_log>`, que é um registro de todas as mudanças feitas, `Tags <administration_tags>`, `Atributos <administration_attributes>` para armazenar metadados personalizados, `Itens de Inventário <administration_inventory>` e informações financeiras como `Transações <payments_transaction>`, `Faturas <payments_invoices>` e `Métodos de Pagamento <basics_payment>`.

Uma vez que criamos um cliente, podemos então `adicionar um serviço <csa_add_service>` a esse cliente, que é a coisa pela qual cobramos.

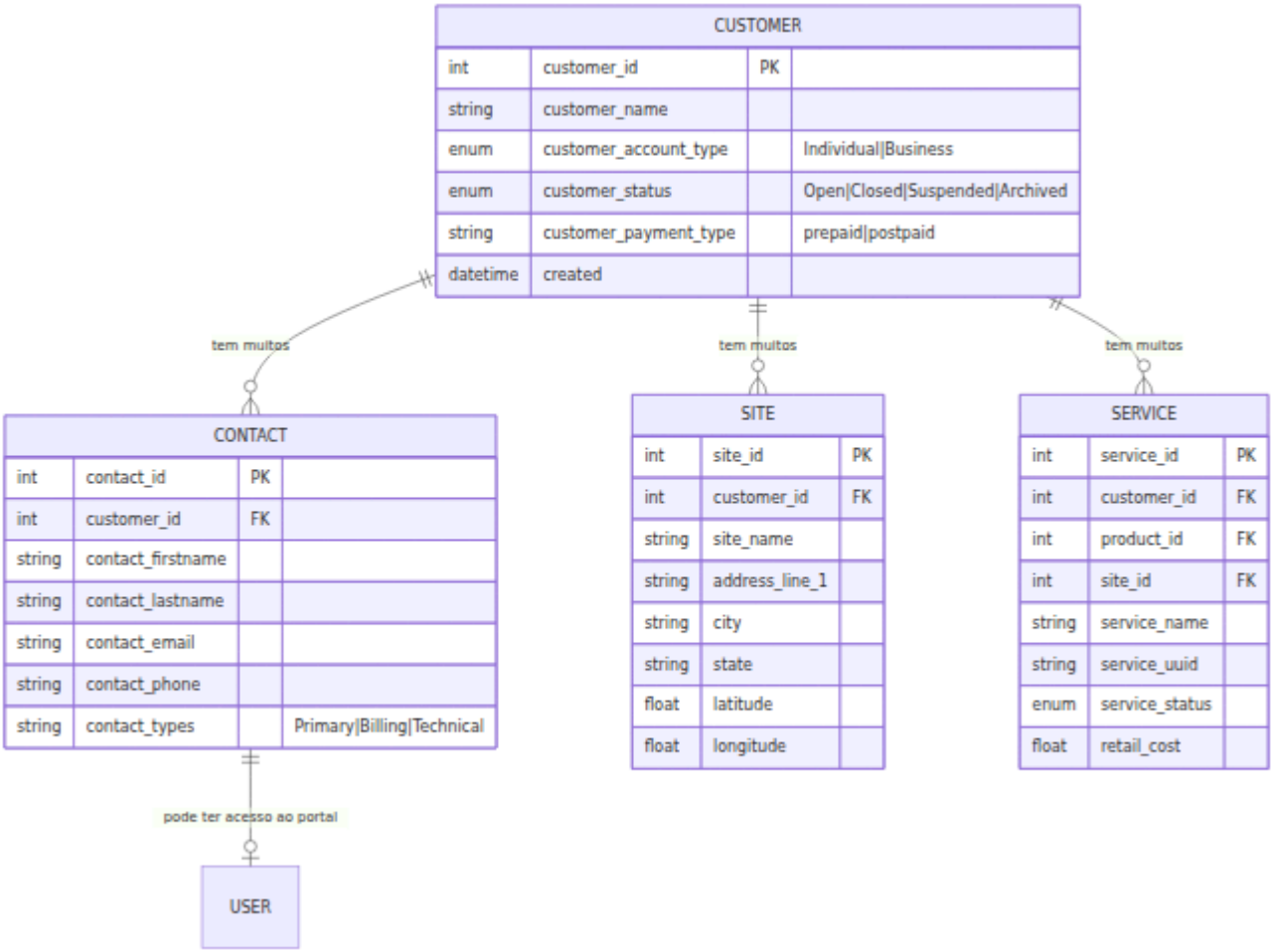
Para informações sobre como criar um cliente, veja Criar um Cliente.

Visão Geral do Modelo de Dados

OmniCRM usa um modelo de dados relacional organizado em torno de clientes e seus serviços. O modelo é dividido em seções focadas abaixo.

Relações Centrais do Cliente

O cliente é a entidade central, com contatos, sites e serviços relacionados.



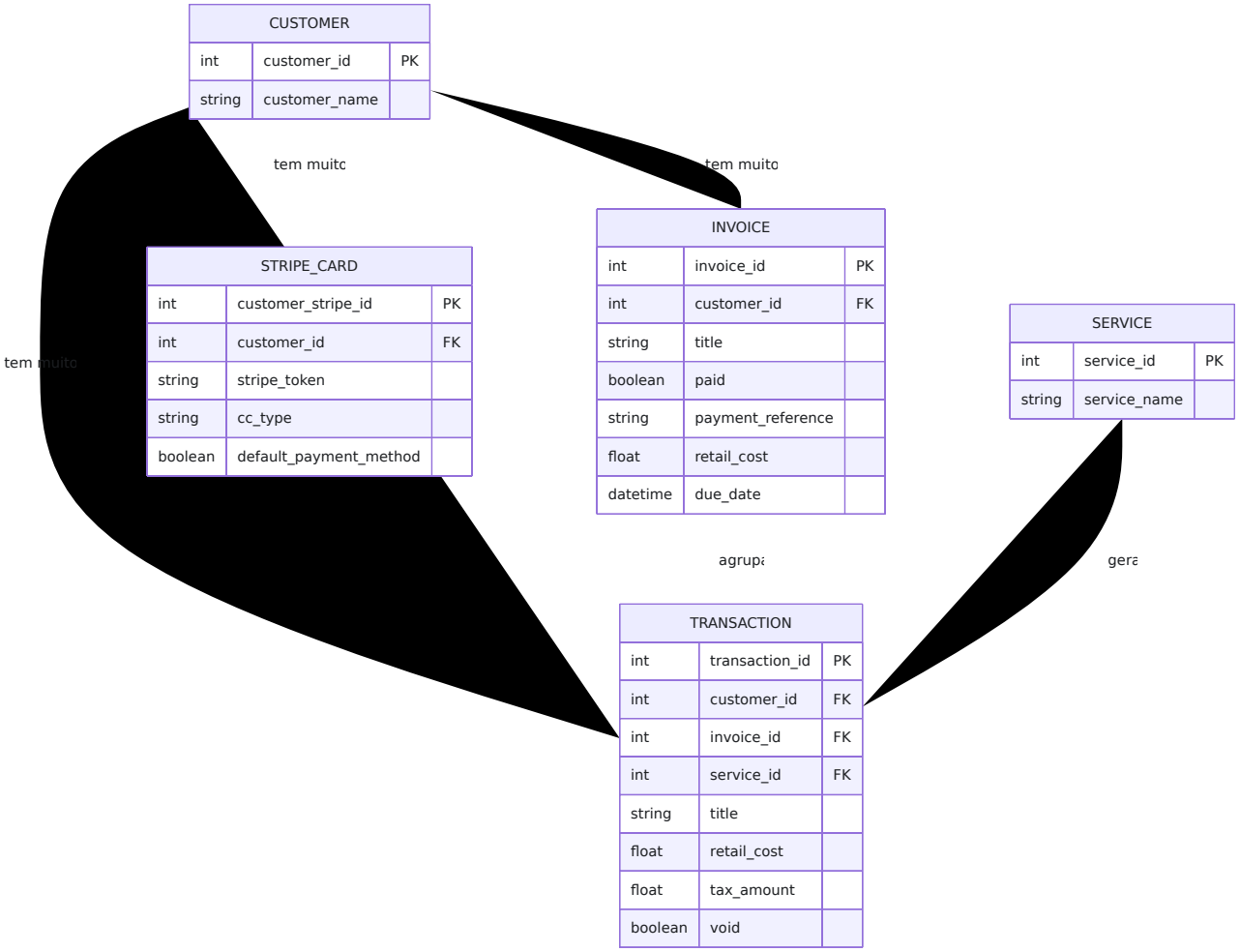
Pontos Chave:

- Um cliente pode ter múltiplos contatos (cobrança, técnico, etc.)
- Um cliente pode ter múltiplos sites (filiais, locais)
- Os serviços são entregues aos sites

- Os contatos podem ter acesso ao portal através de contas de usuário vinculadas

Dados de Cobrança e Financeiros

Transações e faturas rastreiam toda a atividade financeira.

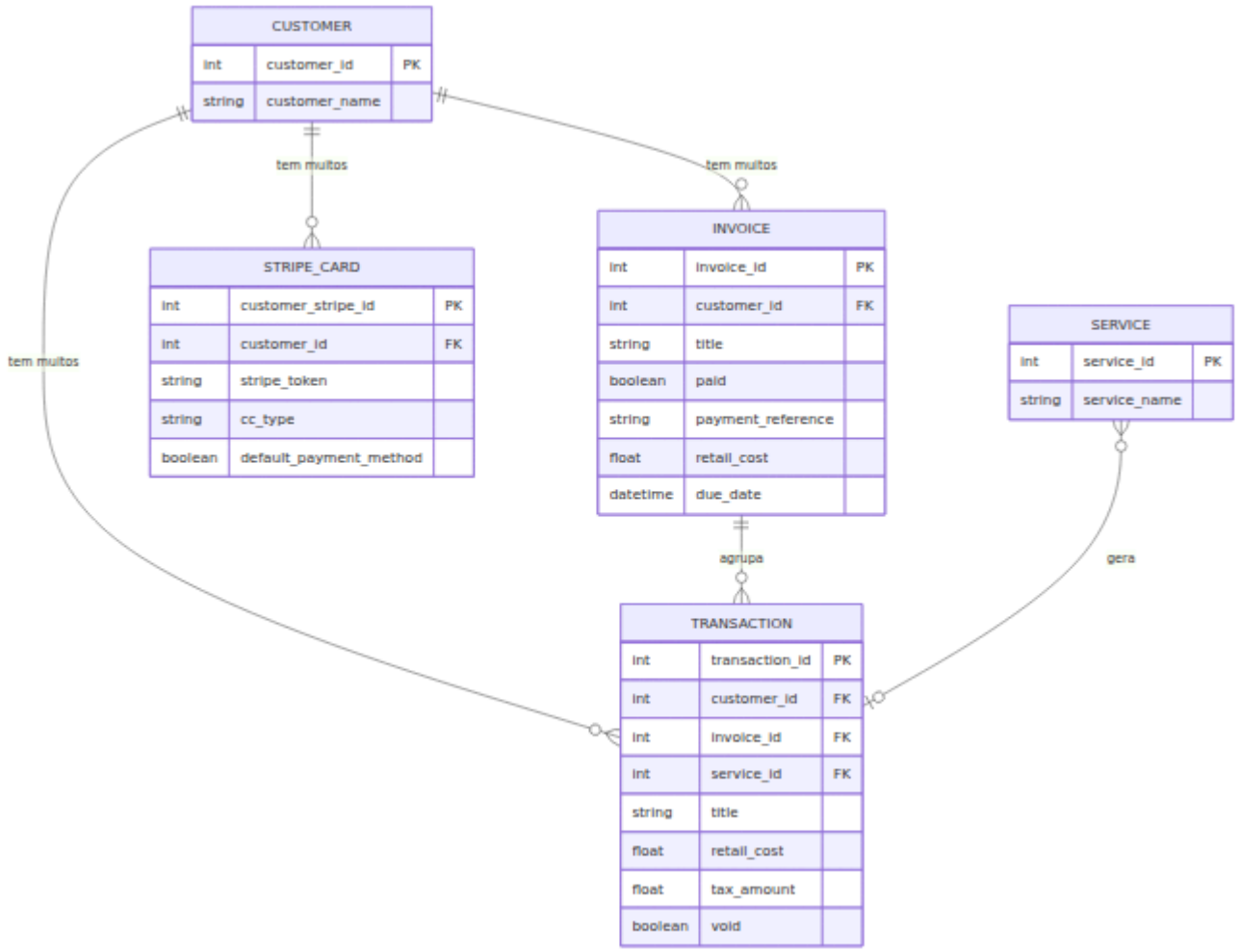


Pontos Chave:

- Transações podem ser independentes ou agrupadas em faturas
- Serviços geram automaticamente transações mensais
- Cartões Stripe são tokenizados e armazenados de forma segura
- Faturas agrupam múltiplas transações para cobrança

Produto e Provisionamento

Produtos definem ofertas de serviços; o provisionamento cria serviços reais.

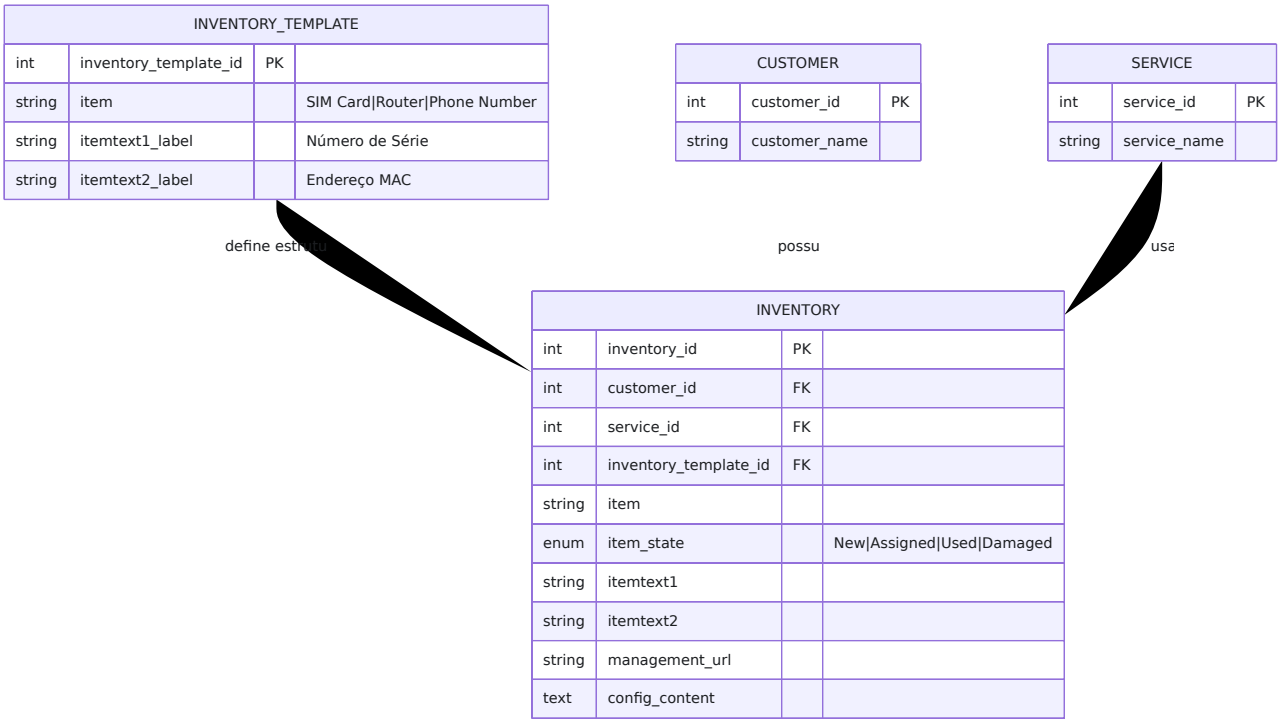


Pontos Chave:

- Produtos são templates; serviços são instâncias ativas
- O provisionamento cria ou modifica serviços via playbooks Ansible
- Cada trabalho de provisionamento tem múltiplos eventos para rastrear o progresso
- Um provisionamento de produto pode criar múltiplos serviços (pacotes)

Sistema de Inventário

Rastreie ativos físicos e virtuais atribuídos a clientes.

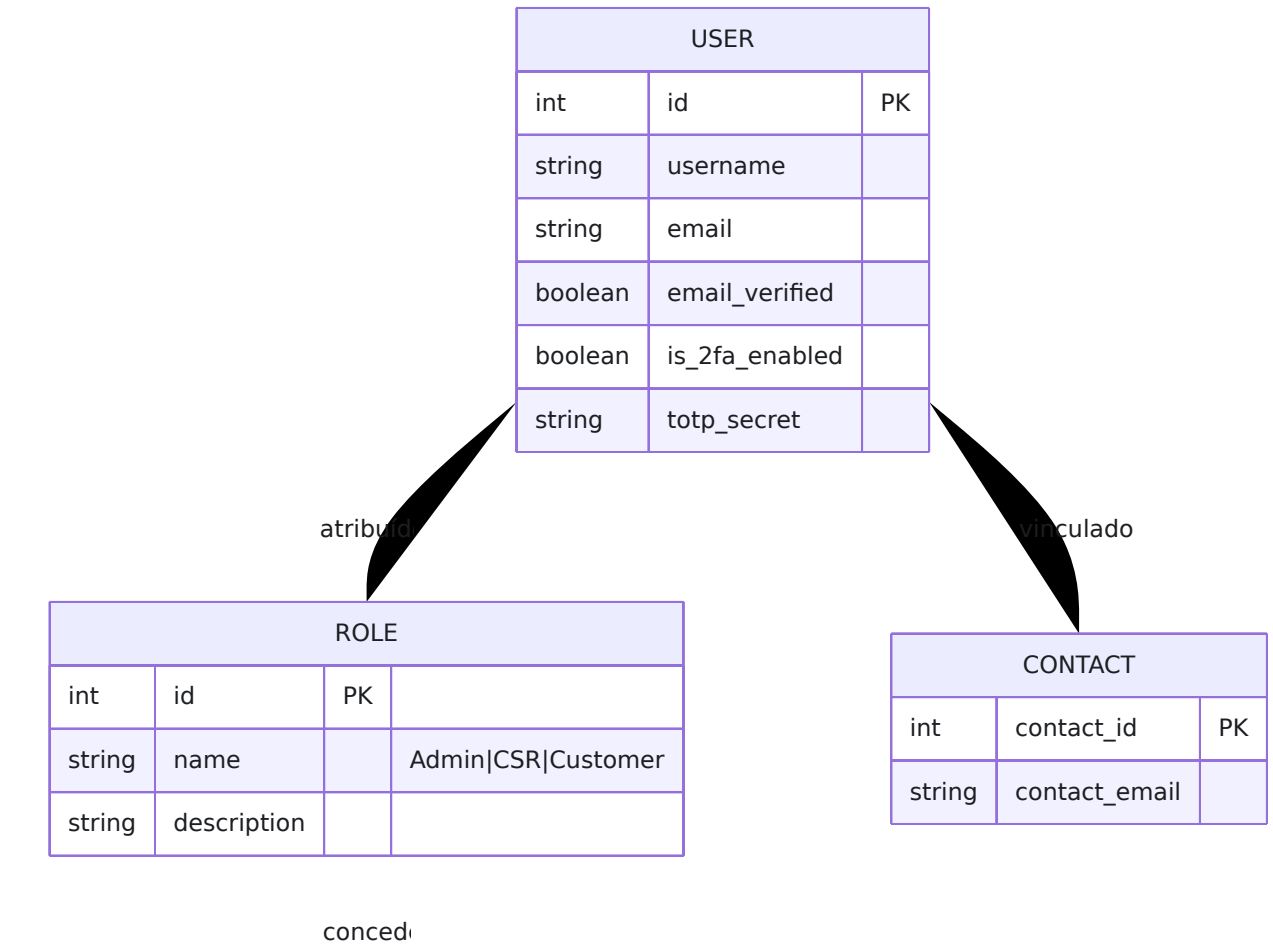


Pontos Chave:

- Templates de inventário definem a estrutura (campos) para cada tipo de item
- Campos flexíveis itemtext1-20 se adaptam a diferentes tipos de inventário
- Itens podem ser atribuídos a clientes e vinculados a serviços
- Armazena configurações de equipamentos e credenciais de gerenciamento

Usuário e Segurança

Contas de usuário com controle de acesso baseado em funções.



Pontos Chave:

- Usuários podem ter múltiplos papéis; papéis têm múltiplas permissões
- A autenticação de dois fatores (2FA) é opcional por usuário
- Usuários da equipe são independentes; usuários clientes estão vinculados a contatos
- Permissões granulares controlam o acesso a recursos

Lista de Clientes

A lista de clientes fornece uma tabela pesquisável e filtrável de todos os clientes no sistema.

Recursos:

- **Pesquisa** - Filtrar clientes por nome ou ID
- **Ações em Lote** - Selecionar múltiplos clientes para operações em lote
- **Paginação** - Navegar por grandes listas de clientes
- **Ações Rápidas** - Visualizar ou excluir clientes diretamente da lista

Cliente - Detalhe

O objeto cliente em si não contém muitas informações, é apenas um nome e uma referência aos contatos e sites.

Sua implementação específica pode incluir campos adicionais ou personalizações, mas o objeto cliente básico é muito simples.

Na página de visão geral também há um gráfico mostrando a Receita Média por Usuário (`ARPU`) para o cliente, que é a receita total dividida pelo número de serviços, e uma comparação de como esse cliente se compara à média de ARPU para todos os clientes no sistema.

As opções de Status do Cliente são adaptadas com base nas necessidades específicas do seu negócio, mas normalmente incluem opções como Ativo, Inativo, Pendente, etc., com diferentes regras em cada uma que controlam o comportamento do cliente no sistema nesse estado.

Excluir um cliente só pode ser feito se o cliente não tiver serviços ativos, faturas não pagas ou transações não faturadas. Se o cliente tiver algum desses, você precisará fechar os serviços ativos e garantir que os pagamentos sejam feitos antes de poder excluir o cliente, o que, por sua vez, arquivará o cliente e todos os dados associados, que podem ser restaurados posteriormente, se necessário.

Site - Detalhe

Os sites são locais físicos onde os serviços são entregues e podem ter múltiplos serviços associados a eles.

Eles são predominantemente usados para clientes empresariais, onde um único cliente pode ter múltiplos sites, por exemplo, um cliente com múltiplos escritórios.

Ter múltiplos sites nos permite rastrear quais serviços estão associados a qual site, por exemplo, se um cliente solicita um novo serviço para um novo escritório, precisamos garantir que entregamos o serviço correto ao local

correto. Isso nos permite rastrear serviços por site e faturá-los separadamente, se necessário.

Integração com Google Maps e Geocodificação

Cada site está integrado com **Google Maps** para garantir dados de endereço precisos e geolocalização. A interface automaticamente geocodifica endereços e gera dados de localização.

Como Funciona a Geocodificação de Endereço:

Ao adicionar ou editar um site, a interface fornece dois métodos para definir a localização:

1. **Pesquisa de Endereço (Recomendado)**
    - Use a barra de pesquisa na parte superior do formulário
    - Digite um endereço e o Google Maps Autocomplete sugere correspondências
    - Selecione o endereço correto no dropdown
    - O sistema automaticamente preenche:
        - **Nome do Site** - Nome do local do Google Maps
        - **Endereço Linha 1** - Número e nome da rua
        - **Endereço Linha 2** - Subprédio (número da unidade/suíte)
        - **Cidade** - Localidade
        - **Estado/Região** - Área administrativa
        - **Código Postal** - Código postal
        - **País** - Nome do país
        - **Latitude & Longitude** - Coordenadas precisas
        - **Código Plus** - Código de Localização Aberta de 11 caracteres (ex., `8C3MFJV8+2F`)
        - **ID do Local do Google Maps** - Identificador único do local
2. **Marcador de Mapa Arrastável (Manual)**
    - Arraste o pino no mapa para a localização exata
    - O sistema realiza **geocodificação reversa** para obter o endereço a partir das coordenadas

    - Os mesmos campos são preenchidos automaticamente com base na localização do pino
    - Útil para áreas rurais ou quando o endereço é impreciso

Geração de Código Plus:

Os Códigos Plus são gerados automaticamente a partir de latitude/longitude usando a biblioteca Open Location Code. Um Código Plus é um código curto (11 caracteres) que representa uma localização precisa em qualquer lugar do mundo.

Exemplo:

> Campos auto-preenchidos: ✓ Nome do Site: "123 Main Street" ✓ Endereço Linha 1: "123 Main Street" ✓ Cidade: "Londres" ✓ Estado: "Grande Londres" ✓ País: "Reino Unido" ✓ Código Postal: "SW1A 1AA" ✓ Latitude: 51.5074 ✓ Longitude: -0.1278 ✓ Código Plus: "9C3XGPHC+3Q" ✓ ID do Local do Google: "ChIJdd4hrwug2EcRmSrV3Vo6lll"

Requisitos de Validação:

Antes de salvar um site, o sistema valida:

- **Latitude & Longitude** devem ser definidos (seja via pesquisa ou arrastando o marcador)
- **País** deve ser preenchido (padrão para `REACT_APP_DEFAULT_COUNTRY` se não definido)
- **Código Plus** deve ter 12 caracteres (11 + 1 para preenchimento)

Se a validação falhar, você verá um erro:

Feedback Visual:

A interface mostra feedback em tempo real:

ou

Como os Dados de Localização são Usados

Os dados de localização geocodificados (latitude, longitude, Código Plus) são usados em todo o OmniCRM para:

1. Entrega e Instalação de Serviços

- **Técnicos de Campo** - Acessar Código Plus para navegar até a localização exata do site
- **Agendamento de Instalação** - Atribuir técnicos com base na proximidade geográfica
- **Implantação de Equipamentos** - Garantir que o equipamento correto seja entregue ao site correto

2. Notificações de Interrupção

- **Alertas Geofenced** - Se houver interrupção de rede em uma área específica, consultar sites por raio de lat/long
- **Comunicações Direcionadas** - Enviar notificações de interrupção apenas para clientes afetados via `Mailjet <integrations_mailjet>`
- **Páginas de Status** - Exibir mapa de interrupção com sites afetados

Exemplo:

> Consulta: SELECT * FROM Customer_Site
> WHERE distance(latitude, longitude, 51.5074, -0.1278) < 5
>
> Resultado: 47 sites afetados Ação: Enviar notificação de interrupção para 47 clientes

3. Relatórios e Análises

- **Receita Geográfica** - Receita por cidade, estado, região
- **Mapas de Densidade de Serviços** - Mapa de calor das localizações de serviços
- **Planejamento de Expansão** - Identificar áreas carentes

4. Clientes Empresariais com Múltiplos Sites

- **Gerenciamento de Sites** - Rastrear quais serviços em quais locais
- **Cobrança Separada** - Faturar por site, se necessário
- **Atribuição de Serviços** - Vincular serviços a sites específicos durante o provisionamento

Sites Rurais e Remotos

Para clientes em áreas rurais onde endereços de rua podem não existir ou serem imprecisos:

1. **Usar Arraste no Mapa**
   - Aproxime-se da área aproximada
   - Arraste o pino para a propriedade/edifício exato
   - O sistema gera Código Plus para essa localização precisa
2. **Compartilhamento de Código Plus**
   - Compartilhe o Código Plus com o cliente (ex., `8C3MFJV8+2F`)
   - O cliente pode inserir isso no Google Maps para ver a localização exata
   - Técnicos de campo usam Código Plus para navegação
3. **Notas de Endereço**
   - Use o campo "Nota de Endereço" para direções adicionais
   - Exemplo: "Vire à esquerda no celeiro vermelho, 500m após a grade de gado"
   - Notas visíveis para equipes de instalação

Dica

Você pode arrastar o pino no mapa para a localização correta se o endereço não for preciso. O sistema fará a geocodificação reversa da localização e preencherá automaticamente todos os campos de endereço.

Contato - Detalhe

Os Contatos são pessoas associadas ao cliente. Eles podem ser contatos de cobrança, contatos técnicos ou outros tipos, e cada contato tem um tipo que influencia como lidamos com o contato.

Podemos ter múltiplos contatos para um único cliente, por exemplo, um cliente com múltiplos contatos de cobrança, ou um cliente com múltiplos contatos técnicos.

Um bom exemplo seria uma empresa com um provedor de serviços gerenciados, que lida com a parte técnica das coisas, e um contato de cobrança separado que lida com a parte financeira, ou uma família onde cada membro tem seu próprio contato, mas nem todos estão autorizados a fazer alterações.

Da mesma forma, podemos querer enviar notificações de interrupção apenas para o contato técnico, ou enviar faturas apenas para o contato de cobrança, e o tipo de contato nos permite controlar isso.

A lógica exata de como os tipos de contato são usados depende do seu negócio, mas a ideia básica é que cada contato tem um tipo que influencia como lidamos com eles, e que cada pessoa associada ao cliente é um contato.

Os contatos são sincronizados com a integração `Mailjet <integrations_mailjet>`, permitindo-nos enviar campanhas de e-mail direcionadas com base no tipo de contato, localização, gastos mensais ou serviços adquiridos, e gerenciar todos os modelos de e-mail usados para comunicações transacionais.

Navegação

OmniCRM foi projetado desde o início para ser responsivo e intuitivo.

Este guia o ajudará a navegar pelo sistema e encontrar as informações de que você precisa.

Uma vez logado, a barra de navegação no lado esquerdo da tela mostrará as principais seções do sistema e as subseções dentro de cada uma.

No canto superior direito da tela, você verá o menu do usuário, que permite sair, alterar sua senha ou acessar suas configurações de usuário.

Você tem links para todos os webapps mais comumente usados da sua organização na barra de WebApp (isso pode ser adaptado às necessidades da sua organização).

Quaisquer alertas pendentes são visíveis no canto superior direito da tela, e você pode clicar no alerta para ver mais informações.

Podemos mudar o idioma do sistema clicando no menu suspenso de idiomas no canto superior direito da tela.

Se você é um príncipe das trevas, pode alternar para o modo escuro clicando no ícone da lua no canto superior direito da tela.

Pagamentos, Faturas e Transações

Transações do Cliente

Qualquer coisa que custe dinheiro no sistema é registrada como uma transação sob o cliente.

Cada transação tem um valor monetário para custo de atacado e custo de varejo, e uma descrição do que a transação é.

As transações podem ser geradas automaticamente pelo sistema, por exemplo, quando um serviço é provisionado, uma transação é criada para o custo de instalação, e quando um serviço é faturado, uma transação é criada para o custo de varejo.

As transações também podem ser criadas manualmente, por exemplo, se um cliente receber um crédito, uma transação é criada para o valor do crédito, ou se uma taxa de instalação for cobrada, uma transação é criada para a taxa de instalação.

As transações são agrupadas para formar `Faturas <payments_invoices>`, que são enviadas ao cliente para pagamento.

Acessando Transações

As transações podem ser visualizadas no nível do sistema ou por cliente:

Visualização por Cliente:

1. Navegue até **Clientes → [Selecionar Cliente]**
2. Clique na aba **Faturamento**
3. Veja a lista de transações no primeiro cartão

Visualização em Todo o Sistema:

1. Navegue até **Faturamento → Transações** (no menu principal)
2. Veja todas as transações de todos os clientes

Widgets de Estatísticas de Transações

No topo da página de transações, quatro cartões de estatísticas exibem resumos financeiros:

Descrições dos Widgets:

- **Total de Transações** - Soma de todos os custos de varejo das transações (todos os tempos)
- **Total de Transações Não Faturadas** - Soma de transações ainda não incluídas em uma fatura
- **Total de Transações Este Mês** - Soma de transações criadas neste mês calendário
- **Total de Transações Mês Passado** - Soma de transações criadas no mês calendário passado

Formatação de Valores:

- Valores acima de 1.000: Exibir como sufixo "k" (por exemplo, $1.5k)
- Valores acima de 1.000.000: Exibir como sufixo "M" (por exemplo, $2.3M)
- Valores acima de 1.000.000.000: Exibir como sufixo "B" (por exemplo, $1.1B)

Lista de Transações

A tabela de transações exibe todas as transações com as seguintes colunas:

Descrições das Colunas:

- **ID** - ID único da transação
- **Data** - Data de criação da transação
- **Título** - Nome curto da transação
- **Descrição** - Descrição detalhada do que a transação é
- **Valor** - Custo de varejo (positivo para cobranças, negativo para créditos)
- **Fatura** - ID da fatura se a transação foi faturada (link clicável)
- **Status** - Marca de verificação se faturada, traço se ainda não faturada

Ações Por Linha:

Cada linha tem um menu de ações (⋮) com opções:

- **Ver Detalhes** - Abre o modal de detalhes da transação
- **Baixar PDF da Fatura** - Baixar PDF (somente se faturada)

- **Anular Transação** - Marcar transação como anulada (somente se não faturada)

Tipos de Transações

As transações se enquadram em duas categorias principais:

Transações de Débito (Cobranças)

Valores positivos que aumentam o saldo devedor do cliente:

- **Taxas de Configuração de Serviço** - Cobranças únicas quando o serviço é provisionado
- **Taxas Mensais de Serviço** - Cobranças recorrentes por serviços
- **Taxas de Instalação** - Cobranças por visitas de técnicos de campo
- **Cobranças de Equipamentos** - Cobranças por modems, roteadores, cartões SIM
- **Taxas de Pagamento Atrasado** - Penalidades por faturas em atraso
- **Cobranças Manuais** - Cobranças personalizadas adicionadas pela equipe

Transações de Crédito (Pagamentos/Reembolsos)

Valores negativos que diminuem o saldo devedor do cliente:

- **Pagamentos em Dinheiro** - Cliente pagou em dinheiro
- **Pagamentos com Cartão** - Cliente pagou com cartão de crédito/débito
- **Pagamentos por Transferência Bancária** - Cliente pagou via transferência bancária
- **Créditos em Conta** - Créditos de boa vontade, compensação
- **Reembolsos** - Dinheiro devolvido ao cliente
- **Descontos** - Descontos promocionais ou de fidelidade

Adicionando uma Transação Manualmente

Clique em **"+ Adicionar Transação"** para abrir o modal de adicionar transação.

Transação de Débito (Cobrança):

Transação de Crédito (Pagamento/Reembolso):

Descrições dos Campos:

- **Tipo de Transação** - Selecione Débito (cobrança) ou Crédito (pagamento/reembolso)
- **Tipo de Crédito** - Se Crédito selecionado, escolha o método de pagamento (Dinheiro, Cartão, Transferência Bancária)
- **Título** - Nome curto para a transação (obrigatório)
- **Descrição** - Explicação detalhada (opcional)
- **Custo de Varejo** - Valor que o cliente paga (obrigatório, número positivo)
- **Custo de Atacado** - Seu custo (opcional, para rastreamento de margem)
- **Porcentagem de Imposto** - Taxa de imposto aplicada a esta transação (opcional, padrão para imposto do produto ou 0%)
- **Serviço** - Vincular transação a um serviço específico (opcional)
- **Site** - Vincular transação a um site específico (opcional)
- **Data da Transação** - Data da transação (padrão para hoje)

Validação:

- Título e custo de varejo são obrigatórios
- O custo de varejo deve ser um número positivo
- Se o tipo de Crédito for selecionado, um tipo de crédito deve ser escolhido

O Que Acontece:

1. Transação criada no banco de dados
2. Aparece na lista de transações do cliente
3. Incluída na contagem de "Transações Não Faturadas"
4. Disponível para inclusão na próxima geração de fatura
5. Entrada de log de atividade criada

Pesquisando e Filtrando Transações

Pesquisa

Use a barra de pesquisa para encontrar transações. Pesquisas em:

- ID da Transação
- Título
- Descrição
- ID da Fatura

Filtros

Aplique filtros para restringir a lista de transações:

Filtros Disponíveis:

- **Status de Anulação** - Todos, Anulada, Não Anulada
- **Status da Fatura** - Todos, Faturada, Não Faturada

Ações de Filtro:

- **Aplicar Filtros** - Aplicar filtros selecionados à lista
- **Redefinir Filtros** - Limpar todos os filtros e mostrar todas as transações

Ordenação

Clique em qualquer cabeçalho de coluna para ordenar:

- **ID** - Ordenar por ID da transação (mais recente/mais antiga)
- **Data** - Ordenar por data da transação
- **Título** - Ordenar alfabeticamente
- **Valor** - Ordenar por custo de varejo (mais alto/mais baixo)
- **Fatura** - Ordenar por ID da fatura

Clique novamente para inverter a direção da ordenação (crescente ↔ decrescente).

Anulando Transações

Transações adicionadas por erro podem ser **anuladas** (marcadas como excluídas).

Requisitos:

- A transação NÃO deve estar faturada
- Uma vez faturadas, as transações não podem ser anuladas (devem ser reembolsadas em vez disso)

Como Anular:

1. Localize a transação na lista
2. Clique no menu de ações (⋮)
3. Selecione **"Anular Transação"**
4. Confirme no modal

O Que Acontece:

- Transação marcada como `void = true`
- Não aparece mais na lista padrão de transações
- Excluída da geração de faturas
- Pode ser visualizada filtrando por transações "Anuladas"
- Deduza do total de "Transações Não Faturadas"

Nota: Anular NÃO é o mesmo que reembolsar. Anular significa "esta transação nunca deveria ter existido." Reembolsar significa "reverter uma transação válida."

Imposto sobre Transações

As transações podem incluir imposto, que é calculado automaticamente com base na configuração de imposto do produto ou especificado manualmente por transação.

Comportamento do Imposto:

- **Transações de Débito (Cobranças)** - O imposto é aplicado às cobranças com base em:
  - **Porcentagem de Imposto do Produto** - Se a transação estiver vinculada a um produto, a porcentagem de imposto do produto é aplicada automaticamente
  - **Substituição Manual** - A equipe pode substituir a porcentagem de imposto ao criar uma transação
  - **Valor do Imposto** - Calculado como: `retail_cost × (tax_percentage / 100)`
  - **Formato de Exibição** - Exibido como: `$10.00 (10%)` nas listas de transações
- **Transações de Crédito (Pagamentos/Reembolsos)** - Nenhum imposto é aplicado a créditos
  - O campo de porcentagem de imposto é oculto para transações de crédito

    - O imposto é automaticamente definido como 0% para todos os pagamentos e reembolsos
    - Créditos reduzem o saldo devedor do cliente sem implicações fiscais

Exemplo de Cálculo de Imposto:

- Produto: Plano Móvel com 10% de imposto, $50.00 custo de varejo
- Cálculo Automático de Imposto: $50.00 × 0.10 = $5.00
- Exibição: $5.00 (10%)

Imposto Zero (NIL/Isento):

- Produtos podem ser isentos de imposto definindo a porcentagem de imposto como 0
- O imposto padrão é 0% se não especificado
- Transações isentas de imposto mostram "-" na coluna de Imposto

Visualização de Detalhes da Transação

Clique em uma transação para ver todos os detalhes:

Transações Faturadas vs Não Faturadas

Transações Não Faturadas:

- Ainda não incluídas em nenhuma fatura
- Disponíveis para a próxima geração de fatura
- Podem ser anuladas
- Contam para o total de "Transações Não Faturadas"
- Status mostra traço (-)

Transações Faturadas:

- Incluídas em uma fatura
- Não podem ser anuladas (devem ser reembolsadas se necessário)
- ID da fatura clicável (link para detalhes da fatura)
- Status mostra marca de verificação (✓)
- Não podem ser modificadas

Geração de Fatura:

Quando você gera uma fatura para um cliente:

1. O sistema encontra todas as transações n��o faturadas para aquele cliente
2. Opcionalmente, filtre por intervalo de datas
3. Transações incluídas na nova fatura
4. Campo `invoice_id` da transação populado
5. Transação agora marcada como "faturada"

Veja `payments_invoices` para detalhes sobre geração de faturas.

Fluxos de Trabalho Comuns

Fluxo de Trabalho 1: Crédito Manual por Interrupção de Serviço

1. Cliente liga: "O serviço ficou fora do ar por 2 dias"
2. A equipe decide creditar £10
3. Navegue até a aba **Faturamento** do cliente
4. Clique em **"+ Adicionar Transação"**
5. Selecione o tipo de transação **Crédito**
6. Selecione o tipo de crédito **Pagamento em Dinheiro**
7. Insira o título: "Crédito por Interrupção de Serviço"
8. Insira a descrição: "Compensação por interrupção de 2 dias 8-9 Jan"
9. Insira o custo de varejo: 10.00
10. Selecione o serviço afetado no dropdown
11. Clique em **"Adicionar Transação"**
12. A transação aparece com valor de -£10.00
13. Será incluída na próxima fatura como crédito

Fluxo de Trabalho 2: Taxa de Instalação Manual

1. O técnico de campo instala o serviço
2. A equipe precisa cobrar £75 de taxa de instalação
3. Navegue até a aba **Faturamento** do cliente
4. Clique em **"+ Adicionar Transação"**
5. Selecione o tipo de transação **Débito**
6. Insira o título: "Taxa de Instalação"
7. Insira a descrição: "Visita do técnico de campo para instalação de fibra"
8. Insira o custo de varejo: 75.00
9. Insira o custo de atacado: 45.00 (opcional, para rastreamento de margem)
10. Selecione o serviço instalado
11. Selecione o site onde foi instalado
12. Clique em **"Adicionar Transação"**

13. A transação aparece na lista de não faturadas
14. Será incluída na próxima fatura

Fluxo de Trabalho 3: Anulando Transação Duplicada

1. A equipe nota uma transação duplicada
2. Verifique se a transação NÃO foi faturada
3. Clique no menu de ações (⋮) na transação duplicada
4. Selecione **"Anular Transação"**
5. Confirme no modal
6. Transação removida da lista
7. Total não faturado diminui de acordo

Fluxo de Trabalho 4: Encontrando Transações para Fatura

1. Necessidade de gerar fatura mensal
2. Clique no filtro **Fatura: "Não Faturada"**
3. Clique em **Aplicar Filtros**
4. Veja todas as transações não faturadas
5. Anote o valor total dos widgets
6. Navegue para gerar fatura
7. Selecione intervalo de datas (por exemplo, 1-31 Jan)
8. Transações no intervalo incluídas na fatura

Solução de Problemas

Não é possível anular a transação

- **Causa:** Transação já faturada
- **Correção:** A transação faz parte do histórico de faturas. Se o reembolso for necessário, crie uma transação de Crédito em vez disso.

Transações duplicadas aparecendo

- **Causa:** Serviço cobrado várias vezes ou erro de provisionamento
- **Correção:** Anule a(s) transação(ões) duplicadas se não faturadas. Se faturadas, emita crédito.

Transação não aparece na lista

- **Causa:** Filtros aplicados ou transação anulada
- **Correção:** Clique em "Redefinir Filtros" para mostrar todas as transações. Para ver transações anuladas, filtre por "Anular: Anulada".

Total não faturado não corresponde ao esperado

- **Causa:** Algumas transações já faturadas ou transações anuladas excluídas
- **Correção:** Aplique o filtro "Fatura: Não Faturada" para ver apenas as não faturadas. Verifique transações anuladas separadamente.

Não é possível adicionar transação (campo cliente desativado)

- **Causa:** Visualizando a página de transações específicas do cliente
- **Correção:** O cliente está pré-selecionado. Se você precisar adicionar uma transação para um cliente diferente, vá para a página de Transações em todo o sistema.

Documentação Relacionada

- `payments_invoices` - Geração e gerenciamento de faturas
- `payments_process` - Processamento de pagamentos contra faturas
- `basics_payment` - Visão geral dos métodos de pagamento
- `csa_activity_log` - Visualizando o histórico de transações no log de atividades

Faturas do Cliente

`Transações </payments_transaction>` são agrupadas para formar uma fatura, que é enviada ao cliente para pagamento.

As faturas têm uma data de início e uma data de término, que é o período que a fatura cobre, e uma data de vencimento, que é a data em que a fatura deve ser paga.

As faturas podem ser geradas automaticamente pelo sistema, por exemplo, quando um serviço é faturado, uma fatura é criada para o custo de varejo, ou podem ser criadas manualmente, por exemplo, se um cliente solicitar uma cópia de uma fatura, ou se um cliente for cobrado por uma cobrança única.

As faturas dos clientes são totalmente modeladas com `Mailjet <integrations_mailjet>` e podem ser personalizadas para incluir o logotipo da empresa, endereço e detalhes de pagamento, e podem ser enviadas ao cliente por e-mail ou baixadas como PDF.

Personalizando Modelos de Fatura

OmniCRM usa modelos HTML com Jinja2 para gerar faturas. Você pode personalizar completamente o design da fatura, branding, cores e layout.

Localização do Modelo de Fatura

Os modelos de fatura são armazenados em `OmniCRM-API/invoice_templates/`

Modelos Padrão:

- `norfone_invoice_template.html` - Modelo de fatura de exemplo
- `cifi_invoice_template.html` - Exemplo de modelo alternativo

Configuração:

O modelo de fatura ativo é especificado em `OmniCRM-API/crm_config.yaml`:

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

Variáveis de Modelo Disponíveis

Os modelos de fatura têm acesso às seguintes variáveis Jinja2:

Informações da Fatura:

- `{{ invoice_number }}` - ID único da fatura (por exemplo, `INV-2025-001234`)
- `{{ date }}` - Data de emissão da fatura (formato ISO: `2025-01-10T12:00:00`)
- `{{ due_date }}` - Data de vencimento do pagamento (por exemplo, `2025-02-10`)
- `{{ start_date }}` - Data de início do período de faturamento
- `{{ end_date }}` - Data de término do período de faturamento
- `{{ total_amount }}` - Valor total da fatura antes do imposto (numérico)
- `{{ total_tax }}` - Valor total do imposto calculado a partir de todas as transações (numérico)

Informações do Cliente:

- `{{ client.name }}` - Nome completo do cliente ou nome da empresa
- `{{ client.address.address_line_1 }}` - Linha de endereço 1
- `{{ client.address.address_line_2 }}` - Linha de endereço 2
- `{{ client.address.city }}` - Cidade
- `{{ client.address.state }}` - Estado/província
- `{{ client.address.zip_code }}` - Código postal/ZIP
- `{{ client.address.country }}` - País

Itens de Linha de Transação:

Percorra as transações usando:

```
{% 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 %}
```

Campos de Transação:

- `sub_transaction.transaction_id` - ID da transação
- `sub_transaction.created` - Data/hora da transação
- `sub_transaction.title` - Título da transação
- `sub_transaction.description` - Descrição detalhada
- `sub_transaction.retail_cost` - Valor do item
- `sub_transaction.tax_percentage` - Porcentagem de imposto aplicada (por exemplo, 10 para 10%)
- `sub_transaction.tax_amount` - Valor do imposto calculado em dólares

Exibindo Imposto em Modelos:

```
<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>
```

Criando um Modelo de Fatura Personalizado

Passo 1: Copiar Modelo Existente

```
cd OmniCRM-API/invoice_templates/
cp norfone_invoice_template.html
your_company_invoice_template.html
```

Passo 2: Personalizar HTML/CSS

Edite `your_company_invoice_template.html` para corresponder à sua marca:

Áreas de Personalização Chave:

1. **Logotipo da Empresa e Branding**

```
<!-- Substitua pela URL do seu logotipo -->
![Sua Empresa](https://yourcompany.com/logo.png)

<!-- Atualize o nome da empresa -->
<h1>Nome da Sua Empresa</h1>
```

2. **Esquema de Cores**

```
<style>
    /* Cor da marca principal */
    .navbar {
        background: linear-gradient(to bottom right, #sua-cor-
1, #sua-cor-2);
    }

    /* Cabeçalhos da tabela */
    .table thead th {
        background-color: #sua-cor-da-marca !important;
        color: white !important;
    }

    /* Botões e links */
    .btn-primary {
        background-color: #sua-cor-da-marca;
    }
</style>
```

3. **Informações da Empresa no Rodapé**

```
<footer>
    <p>Nome da Sua Empresa</p>
    <p>123 Business Street, Cidade, País</p>
    <p>Telefone: +1-555-123-4567 | Email:
billing@yourcompany.com</p>
    <p>ABN/ID Fiscal: 12345678900</p>
</footer>
```

4. **Instruções de Pagamento**

```
<div class="payment-info">
    <h3>Métodos de Pagamento</h3>
    <p><strong>Online:</strong> Pague em
https://yourcompany.com/pay</p>
    <p><strong>Transferência Bancária:</strong></p>
    <ul>
        <li>Nome da Conta: Sua Empresa Ltd</li>
        <li>BSB: 123-456</li>
        <li>Número da Conta: 987654321</li>
        <li>Referência: {{ invoice_number }}</li>
    </ul>
</div>
```

5. **Termos e Condições**

```
<div class="terms">
    <h4>Termos de Pagamento</h4>
    <p>Pagamento devido dentro de 30 dias a partir da data da
fatura.</p>
    <p>Taxas de pagamento atrasado: 2% ao mês sobre saldos em
atraso.</p>
    <p>Para consultas de faturamento:
billing@yourcompany.com</p>
</div>
```

Passo 3: Atualizar Configuração

Edite `OmniCRM-API/crm_config.yaml`:

```
invoice:
  template_filename: 'your_company_invoice_template.html'
```

Passo 4: Reiniciar API

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

Passo 5: Testar Geração de Fatura

1. Navegue até um cliente com transações
2. Gere uma fatura de teste
3. Baixe o PDF para verificar a formatação
4. Envie a fatura para você mesmo para testar a entrega do e-mail

Personalização Avançada

Conteúdo Condicional:

Use condicionais Jinja2 para mostrar/ocultar conteúdo:

```
{% if total_amount > 1000 %}
    <div class="high-value-notice">
        <p><strong>Nota:</strong> Saldo alto - Plano de pagamento
disponível mediante solicitação.</p>
    </div>
{% endif %}

{% if client.address.country == "Australia" %}
    <p>GST Incluído: ${{ "%.2f"|format(total_amount * 0.10) }}</p>
{% endif %}
```

Suporte a Múltiplos Idiomas:

Crie modelos específicos para cada idioma:

```
invoice_template_en.html
invoice_template_es.html
invoice_template_fr.html
```

Configure com base na preferência de idioma do cliente.

Cálculos Personalizados:

```
<!-- Exibir subtotal e detalhamento do imposto -->
<tr>
    <td colspan="4" class="text-right"><strong>Subtotal:</strong>
</td>
    <td>${{ "%.2f"|format(total_amount) }}</td>
</tr>
<tr>
    <td colspan="4" class="text-right"><strong>Imposto:</strong>
</td>
    <td>${{ "%.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>
```

Nota: A variável `total_tax` é calculada automaticamente somando o `tax_amount` de todas as transações na fatura. O imposto de cada transação é calculado com base em seu campo `tax_percentage`, que padrão é a `tax_percentage` do produto ou 0% se não especificado.

QR Code para Pagamento:

Gere um QR code para pagamento móvel:

```
<div class="qr-payment">
    ![Escanear para Pagar](https://api.qrserver.com/v1/create-qr-
code/?size=150x150&data={{ payment_url }})
    <p>Escaneie com seu telefone para pagar instantaneamente</p>
</div>
```

Melhores Práticas de Estilo para PDF

OmniCRM usa **WeasyPrint** para converter HTML em PDF. Siga estas diretrizes:

CSS Suportado:

- A maioria das propriedades CSS 2.1
- CSS3 limitado (flexbox, algumas transformações)
- Fontes da web via `@font-face`

Não Suportado:

- JavaScript
- CSS Grid (use tabelas em vez disso)
- Animações complexas
- Algumas propriedades CSS modernas

Tamanho da Página e Margens:

```
@page {
    size: A4;
    margin: 1cm;
}

body {
    font-family: Arial, sans-serif;
    font-size: 10pt;
}
```

Estilo Específico para Impressão:

```
@media print {
    .no-print {
        display: none;
    }

    .page-break {
        page-break-after: always;
    }
}
```

Layout da Tabela:

```
.table {
    table-layout: fixed;
    width: 100%;
}

.table th, .table td {
    word-wrap: break-word;
    padding: 4px;
}
```

Incorporação de Fonte:

Para fontes personalizadas, use fontes seguras para a web ou incorpore:

```
@font-face {
    font-family: 'SuaFonte';
    src: url('https://yourcompany.com/fonts/yourfont.woff2')
format('woff2');
}

body {
    font-family: 'SuaFonte', Arial, sans-serif;
}
```

Testando Modelos de Fatura

Checklist de Teste:

1. **Inspeção Visual:**
   - Logotipo é exibido corretamente
   - Cores correspondem às diretrizes da marca
   - Texto é legível (não muito pequeno)
   - Tabelas estão alinhadas corretamente
   - Todas as seções estão presentes
2. **Precisão dos Dados:**
   - Detalhes do cliente estão corretos
   - Valores das transações somam corretamente
   - Datas estão formatadas corretamente
   - Todas as variáveis estão substituindo corretamente
3. **Qualidade do PDF:**
   - Tamanho do arquivo razoável (<5MB)
   - Imagens nítidas e claras
   - Sem corte ou transbordamento de texto
   - Quebras de página em locais apropriados
4. **Faturas de Múltiplas Páginas:**
   - Cabeçalhos se repetem em cada página
   - Números de página são exibidos
   - Listas longas de transações paginam corretamente
5. **Entrega de E-mail:**
   - PDF anexa ao e-mail
   - Tamanho do arquivo abaixo do limite do Mailjet (15MB)
   - Renderiza no Gmail, Outlook, Apple Mail

Comando de Teste (Geração Manual):

Você pode testar a geração de faturas via API:

```
curl -X GET "http://localhost:5000/crm/invoice/{invoice_id}/pdf" \
     -H "Authorization: Bearer YOUR_TOKEN" \
     --output test_invoice.pdf
```

Problemas Comuns de Modelo

Variáveis não substituindo:

- **Causa:** Erro de digitação no nome da variável ou dados ausentes
- **Correção:** Verifique a ortografia exatamente (sensível a maiúsculas e minúsculas), verifique se os dados existem no banco de dados

Estilo do PDF quebrado:

- **Causa:** Propriedade CSS não suportada
- **Correção:** Use propriedades CSS 2.1, teste com CSS compatível com WeasyPrint

Imagens não aparecem:

- **Causa:** URLs relativas ou recursos externos bloqueados
- **Correção:** Use URLs absolutas HTTPS, certifique-se de que as imagens sejam acessíveis publicamente

Tabelas transbordando a página:

- **Causa:** Larguras de coluna fixas muito largas
- **Correção:** Use larguras percentuais, `table-layout: fixed`

Fontes não renderizando:

- **Causa:** Fonte não incorporada ou indisponível
- **Correção:** Use fontes seguras para a web (Arial, Times New Roman, etc.) ou incorpore corretamente fontes personalizadas

Falha na geração de PDF:

- **Causa:** Erros de sintaxe HTML ou falha do WeasyPrint
- **Correção:** Valide o HTML, verifique os logs do WeasyPrint, simplifique layouts complexos

Cache de PDF de Fatura

Para melhorar o desempenho e reduzir a geração redundante de PDFs, o OmniCRM inclui um sistema de cache de PDF de fatura. Quando um PDF de fatura é gerado pela primeira vez, ele é armazenado em cache no banco de dados para solicitações subsequentes.

Como Funciona o Cache de PDF:

1. **Primeira Solicitação** - Quando um PDF de fatura é solicitado (download ou e-mail), o sistema:
   - Gera o PDF a partir do modelo de fatura
   - Codifica o PDF como Base64
   - Calcula um hash SHA256 do conteúdo do PDF
   - Armazena na tabela `Invoice_PDF_Cache` com:
     - Referência do ID da fatura
     - Dados do PDF (codificados em Base64)
     - Nome do arquivo
     - Hash de conteúdo (para verificação de integridade)
     - Timestamp de criação
2. **Solicitações Subsequentes** - Quando a mesma fatura é solicitada novamente:
   - O sistema verifica se há PDF em cache pelo invoice_id
   - Se o cache existir e for válido, retorna o PDF em cache imediatamente
   - Atualiza o timestamp `last_accessed` para rastrear o uso do cache
3. **Invalidando o Cache** - PDFs em cache são invalidados quando:
   - A fatura é modificada (transações adicionadas/removidas, detalhes alterados)
   - O modelo de fatura é atualizado
   - A limpeza manual do cache é acionada

Benefícios:

- **Desempenho** - Entrega instantânea de PDF para solicitações repetidas (sem atraso de regeneração)

- **Consistência** - Mesmo PDF para todos os downloads de uma fatura (a menos que a fatura seja modificada)
- **Carga do Servidor** - Reduz o uso da CPU na geração de PDFs
- **Experiência do Usuário** - Indicador de carregamento aparece durante a geração inicial, solicitações subsequentes são instantâneas

Gerenciamento de Cache:

O Cache de PDF de Fatura é gerenciado automaticamente pelo sistema. Entradas de cache antigas ou não utilizadas podem ser purgadas periodicamente com base em:

- Idade (por exemplo, remover entradas de cache com mais de 90 dias)
- Padrões de acesso (remover entradas não acessadas em 30 dias)
- Limites de armazenamento (implementar limites de tamanho de cache, se necessário)

Comportamento da API:

Ao baixar uma fatura via API ou UI:

- Primeira solicitação: Mostra indicador de carregamento enquanto o PDF é gerado, depois armazena em cache
- Solicitações subsequentes: Download imediato do cache
- Acerto de cache é transparente para o usuário

Importante: Quando você atualizar seu modelo de fatura, limpe o cache para garantir que novas faturas usem o design atualizado:

```
-- Limpar todos os PDFs de fatura em cache (executar no MySQL)
DELETE FROM Invoice_PDF_Cache;
```

Ou atualize `crm_config.yaml` para invalidar automaticamente o cache na alteração do modelo.

Acessando Faturas

As faturas podem ser visualizadas no nível do sistema ou por cliente:

Visualização por Cliente:

1. Navegue até **Clientes → [Selecionar Cliente]**
2. Clique na aba **Faturamento**
3. Veja a lista de faturas no terceiro cartão

Visualização em Todo o Sistema:

1. Navegue até **Faturamento → Faturas** (no menu principal)
2. Veja todas as faturas de todos os clientes

Widgets de Estatísticas de Faturas

No topo da página de faturas, quatro cartões de estatísticas exibem resumos financeiros.

Descrições dos Widgets:

- **Total de Faturas** - Soma de todos os custos de varejo das faturas (todos os tempos) e contagem de faturas enviadas

- **Faturas Não Pagas** - Soma de faturas ainda não pagas e contagem de faturas não pagas
- **Faturas Este Mês** - Soma de faturas criadas neste mês calendário com contagem
- **Faturas Mês Passado** - Soma de faturas criadas no mês calendário passado com contagem

Formatação de Valores:

- Valores acima de 1.000: Exibir como sufixo "k" (por exemplo, $1.5k)
- Valores acima de 1.000.000: Exibir como sufixo "M" (por exemplo, $2.3M)
- Valores acima de 1.000.000.000: Exibir como sufixo "B" (por exemplo, $1.1B)

Indicadores de Tendência:

- Widgets para "Este Mês" e "Mês Passado" mostram mudança percentual
- Seta verde para cima: Aumento em relação ao período anterior
- Seta vermelha para baixo: Diminuição em relação ao período anterior
- Seta cinza para a direita: Sem mudança

Lista de Faturas

A tabela de faturas exibe todas as faturas com as seguintes colunas:

Descrições das Colunas:

- **ID** - ID único da fatura
- **Título** - Título/descrição da fatura
- **Período** - Período de faturamento (data de início - data de término) ou "N/A" para faturas únicas
- **Data de Vencimento** - Data de vencimento do pagamento
- **Criado** - Data de criação da fatura
- **Valor** - Valor total da fatura (custo de varejo)
- **Status** - Pago, Não Pago ou Reembolsado
- **Ações** - Ações disponíveis (varia conforme o status)

Ícones de Ação:

- ⬇ **(Baixar)** - Baixar PDF da fatura
- □ **(Excluir)** - Anular fatura (somente se não paga)
- □ **(Pagar)** - Pagar fatura online (somente se não paga)
- ✉ **(E-mail)** - Enviar e-mail da fatura para o cliente
- □ **(Reembolsar)** - Reembolsar pagamento Stripe (somente para faturas Stripe pagas)

Gerando uma Fatura

Clique em **"+ Gerar Fatura Proforma"** para criar uma nova fatura.

Descrições dos Campos:

- **Pesquisar Clientes** - Selecionar cliente (apenas mostrado na visualização em todo o sistema, pré-preenchido na visualização do cliente)
- **Título** - Título/nome da fatura (opcional, padrão para "Fatura para [Período]")
- **Data de Início** - Início do período de faturamento (padrão para 14 dias atrás)
- **Data de Término** - Término do período de faturamento (padrão para hoje)
- **Data de Vencimento** - Prazo de pagamento (padrão para hoje)
- **Pré-visualização de Transação** - Mostra todas as transações não faturadas no intervalo de datas com a capacidade de incluir/excluir transações específicas

Seleção de Transações:

- ✓ **(Sinal Verde)** - Clique para excluir uma transação da fatura
- × **(Sinal Vermelho)** - Clique para incluir uma transação anteriormente excluída
- **Selecionar Tudo** - Incluir todas as transações exibidas

- **Limpar Tudo** - Excluir todas as transações
- Transações excluídas aparecem esmaecidas com texto riscado
- Totais em tempo real atualizam conforme você seleciona/deseleciona transações

O Que Acontece:

1. O sistema encontra todas as transações não faturadas para o cliente dentro do intervalo de datas
2. Exibe a pré-visualização das transações com a capacidade de incluir/excluir transações individuais
3. Mostra o cálculo em tempo real do subtotal, imposto e total com base nas transações selecionadas
4. Apenas transações selecionadas (incluídas) são adicionadas à fatura
5. Gera o PDF da fatura e o armazena em cache
6. Marca as transações selecionadas como faturadas (`invoice_id` campo populado)
7. Transações excluídas permanecem não faturadas e disponíveis para futuras faturas
8. A fatura aparece na lista com status "Não Pago"

Exemplos de Casos de Uso:

Faturamento Mensal: Defina a data de início para o primeiro dia do mês, a data de término para o último dia do mês, a pré-visualização mostra todas as transações não faturadas desse período. Selecione todas ou exclua manualmente as específicas.

Fatura Específica de Serviço: Use o mesmo intervalo de datas, depois exclua manualmente as transações indesejadas (por exemplo, exclua transações não móveis para criar uma fatura apenas para móveis).

Fatura Única: Defina tanto a data de início quanto a data de término para o mesmo dia, a pré-visualização mostra apenas transações dessa data. Exclua quaisquer cobranças não relevantes para esta fatura específica.

Visualizando Detalhes da Fatura

Clique em qualquer linha de fatura na tabela para ver todos os detalhes da fatura, incluindo todas as transações, totais e ações disponíveis.

Modal de Detalhes da Fatura:

- **Informações da Fatura** - Mostra ID da fatura, título, datas, status de pagamento e status de anulação
- **Lista de Transações** - Exibe todas as transações incluídas na fatura com:
  - Data da transação
  - Título e descrição
  - Custo de varejo
  - Valor do imposto e porcentagem (formatado como `$10.00 (10%)`)
  - Transações isentas de imposto mostram "-" na coluna de Imposto
- **Resumo de Totais** - Cálculo em tempo real mostrando:
  - Contagem de transações
  - Subtotal (soma de todos os custos de varejo)
  - Imposto (soma de todos os valores de imposto)
  - Total da Fatura (subtotal + imposto)
- **Botões de Ação** - Mesmas ações disponíveis como na tabela:
  - **Baixar PDF** - Baixar PDF da fatura (sempre disponível)
  - **Enviar E-mail** - Enviar fatura para o cliente (faturas não anuladas)

- **Pagar Fatura** - Processar pagamento (faturas não pagas, não anuladas apenas)
- **Reembolsar** - Reembolsar pagamento Stripe (apenas faturas Stripe pagas)
- **Excluir** - Anular fatura (faturas não pagas, não anuladas apenas)

Baixando PDFs de Faturas

Clique no **ícone de download (⬇)** na tabela ou no botão **"Baixar PDF"** no modal de detalhes da fatura para baixar uma fatura como PDF.

Processo de Download:

1. Clique no ícone de download ao lado da fatura
2. O indicador de carregamento aparece durante a geração (apenas na primeira vez)
3. O navegador solicita salvar o arquivo: `Invoice_01234.pdf`
4. O PDF abre ou é salvo na pasta Downloads

Comportamento de Cache de PDF:

- **Primeiro Download** - PDF gerado a partir do modelo, armazenado em cache no banco de dados (pode levar 2-3 segundos)
- **Downloads Subsequentes** - Download instantâneo do cache
- **Invalidando o Cache** - Cache limpo se a fatura for modificada ou o modelo for atualizado

Solução de Problemas para Questões de Download:

- **O indicador de carregamento nunca para** - Verifique o console do navegador, a API pode estar fora do ar
- **PDF em branco ou corrompido** - Verifique o modelo de fatura para erros de sintaxe
- **Download falha** - Verifique as configurações do bloqueador de pop-ups, tente um navegador diferente

Pagando Faturas

Clique no **ícone de pagamento (□)** para pagar uma fatura online.

Processo de Pagamento:

1. Clique no ícone de pagamento na fatura não paga
2. O modal de pagamento se abre mostrando os detalhes da fatura
3. Selecione o método de pagamento:
    - **Transação Stripe** - Cobrar cartão de crédito salvo (disponível para todos os usuários)
    - **Dinheiro** - Pagamento manual em dinheiro (somente para a equipe)
    - **Reembolso** - Aplicar reembolso como pagamento (somente para a equipe)
    - **Transação POS** - Terminal de ponto de venda (somente para a equipe)
    - **Transferência Bancária** - Transferência bancária manual (somente para a equipe)
4. Se Stripe for selecionado:
    - Selecione o cartão dos métodos de pagamento salvos
    - O cartão padrão é pré-selecionado
    - Clique para selecionar um cartão diferente
5. Se outro método for selecionado:
    - Insira o número de referência (opcional)

6. Clique em **"Pagar Fatura"** para processar
7. O sistema processa o pagamento:
    - **Stripe** - Cobra o cartão via API Stripe
    - **Outros métodos** - Cria uma transação negativa para o valor do pagamento
8. O status da fatura muda para "Pago"
9. Notificação de sucesso exibida

Portal de Autoatendimento vs Pagamento da Equipe:

:doc:`Portal de Autoatendimento <self_care_portal>` (Clientes):

- Apenas pagamento Stripe disponível
- Deve ter método de pagamento salvo
- Aviso exibido se não existirem métodos de pagamento
- Link para adicionar método de pagamento fornecido

Portal da Equipe (Administradores):

- Todos os métodos de pagamento disponíveis
- Pode marcar a fatura como paga manualmente (dinheiro, POS, transferência bancária)
- Pode inserir números de referência para rastreamento

Aviso de Método de Pagamento:

Se o cliente não tiver métodos de pagamento salvos, um aviso é exibido solicitando que ele adicione um método de pagamento antes que possa pagar faturas.

Enviando Faturas por E-mail

Clique no **ícone de e-mail (✉)** para enviar a fatura ao cliente.

O Que Acontece:

1. Clique no ícone de e-mail ao lado da fatura
2. O sistema recupera o PDF da fatura do cache (ou gera se não estiver em cache)
3. Envia e-mail via `Mailjet <integrations_mailjet>` usando o modelo `api_crmCommunicationCustomerInvoice`
4. O e-mail inclui:
   - PDF da fatura como anexo
   - Nome do cliente
   - ID da fatura e data de vencimento
   - Valor total devido
   - Link para pagar a fatura online
   - Link para visualizar/baixar a fatura
5. Notificação de sucesso: "E-mail da fatura enviado com sucesso"

Destinatários do E-mail:

O e-mail é enviado para todos os contatos do cliente com tipo "faturamento" ou contato principal se não existir contato de faturamento.

Variáveis do Modelo de E-mail:

- `{{ var:customer_name }}` - Nome completo do cliente
- `{{ var:invoice_number }}` - ID da fatura
- `{{ var:invoice_date }}` - Data de emissão da fatura
- `{{ var:due_date }}` - Data de vencimento do pagamento
- `{{ var:total_amount }}` - Valor total devido
- `{{ var:invoice_url }}` - Link para visualizar/baixar PDF
- `{{ var:pay_url }}` - Link para pagar a fatura online

Solução de Problemas para Questões de E-mail:

- **E-mail não enviado** - Verifique as credenciais da API Mailjet em `crm_config.yaml`
- **Cliente não recebendo** - Verifique os endereços de e-mail dos contatos do cliente
- **PDF não anexado** - Verifique se a geração do PDF foi bem-sucedida (tente baixar primeiro)

Anulando Faturas

Clique no **ícone de exclusão (🗑)** para anular uma fatura.

Requisitos:

- A fatura deve estar **Não Paga**
- Faturas pagas não podem ser anuladas (devem ser reembolsadas em vez disso)

Como Anular:

1. Localize a fatura não paga na lista
2. Clique no ícone de exclusão (🗑)
3. Confirme no modal:

O Que Acontece:

- Fatura marcada como `void = true`
- Todas as transações desvinculadas da fatura (`invoice_id` definido como nulo)
- Transações tornam-se "não faturadas" novamente
- Transações podem ser incluídas em nova fatura
- Fatura aparece na lista com prefixo "Anulada:" no título
- Ações da fatura desativadas (sem download, pagamento ou e-mail)
- Pode ser visualizada filtrando por faturas "Anuladas"

Notas Importantes:

- Anular NÃO é o mesmo que reembolsar
- **Anular** = "Esta fatura nunca deveria ter existido" (erro de faturamento, duplicado)
- **Reembolsar** = "Reverter uma fatura paga válida" (devolver dinheiro ao cliente)

Reembolsando Faturas

Clique no **ícone de reembolso (□)** para reembolsar uma fatura paga.

Requisitos:

- A fatura deve estar **Paga**
- A fatura deve ter sido paga via **Stripe**
- A fatura deve ter um `payment_reference` válido (ID de intenção de pagamento Stripe)
- Disponível apenas para usuários da equipe (n��o para Autoatendimento)

Como Reembolsar:

1. Localize a fatura paga da Stripe
2. Clique no ícone de reembolso (□)
3. O modal de confirmação de reembolso se abre:

4. Clique em **"Confirmar Reembolso"**
5. O sistema processa o reembolso da Stripe:
    - Chama a API da Stripe para reembolsar o pagamento
    - Cria uma transação de reembolso na Stripe
    - Atualiza a fatura com `refund_reference`
6. O status da fatura muda para "Reembolsada"
7. Notificação de sucesso exibida

O Que Acontece Após o Reembolso:

- A fatura permanece no sistema (não anulada)
- O status mostra "Reembolsada"
- As transações permanecem vinculadas à fatura

- O cliente recebe o reembolso no método de pagamento original (3-7 dias úteis)
- O painel da Stripe mostra a transação de reembolso

Restrições de Reembolso:

- Não é possível reembolsar faturas pagas em dinheiro, POS ou transferência bancária (reversão manual necessária)
- Não é possível reembolsar parcialmente (apenas valor total da fatura)
- Não é possível reembolsar duas vezes

Pesquisando e Filtrando Faturas

Pesquisa

Use a barra de pesquisa para encontrar faturas. Pesquisas em:

- ID da Fatura
- Título da Fatura
- Nome do Cliente (apenas visualização em todo o sistema)

Filtros

Aplique filtros para restringir a lista de faturas:

Filtros Disponíveis:

- **Status de Anulação** - Todos, Anulada, Não Anulada
- **Status de Pagamento** - Todos, Pago, Não Pago

Ações de Filtro:

- **Aplicar Filtros** - Aplicar filtros selecionados à lista
- **Redefinir Filtros** - Limpar todos os filtros e mostrar todas as faturas

Ordenação

Clique em qualquer cabeçalho de coluna para ordenar:

- **ID** - Ordenar por ID da fatura (mais recente/mais antiga)
- **Título** - Ordenar alfabeticamente
- **Data de Vencimento** - Ordenar por data de vencimento
- **Criado** - Ordenar por data de criação
- **Valor** - Ordenar por custo de varejo (mais alto/mais baixo)
- **Status** - Ordenar por status de pagamento (pago primeiro ou não pago primeiro)

Clique novamente para inverter a direção da ordenação (crescente ↔ decrescente).

Paginação

Navegue por listas grandes de faturas com controles de página mostrando a página atual, total de páginas e seletor de itens por página (10, 25, 50 ou 100 itens).

Fluxos de Trabalho Comuns de Faturas

Fluxo de Trabalho 1: Faturamento Mensal com Pré-visualização de Transação

1. O final do mês chega (por exemplo, 31 de janeiro)
2. Navegue até **Faturamento ⟩⟩ Faturas**
3. Clique em **"+ Gerar Fatura Proforma"**
4. Selecione o cliente (ou faça por cliente se faturando muitos clientes)
5. Defina as datas:
    - Data de Início: 2025-01-01
    - Data de Término: 2025-01-31

    - Data de Vencimento: 2025-02-15 (15 dias a partir de agora)
    - Título: "Serviços de Janeiro de 2025" (opcional)
6. A seção **Pré-visualização de Transação** aparece mostrando todas as transações não faturadas de janeiro
7. Revise a pré-visualização:
    - Todas as transações estão incluídas por padrão
    - Verifique os totais: Subtotal, Imposto e Total da Fatura
    - Verifique se todas as cobranças estão corretas
8. Clique em **"Gerar Fatura"** (o botão mostra a contagem de transações, por exemplo, "Gerar Fatura (15)")
9. Fatura criada com todas as transações selecionadas
10. Clique na linha da fatura para ver detalhes e verificar
11. Clique no botão **"Enviar E-mail"** no modal de detalhes ou no ícone de e-mail na tabela
12. O cliente recebe o e-mail da fatura com PDF e link de pagamento

Fluxo de Trabalho 2: Faturamento Seletivo de Transações

1. O cliente tem vários serviços (Móvel + Internet) e cobranças diversas
2. Deseja faturas separadas para cada serviço
3. **Gerar a primeira fatura (Serviços Móveis):**
    - Clique em **"+ Gerar Fatura Proforma"**
    - Título: "Serviços Móveis - Janeiro de 2025"
    - Data de Início/Fim: Jan 1-31
    - Data de Vencimento: Feb 15
    - Na pré-visualização de transações, **exclua** todas as transações não móveis:
        - Clique no botão **X** ao lado das transações de Internet
        - Clique no botão **X** ao lado das cobranças diversas
        - Apenas as transações de serviço móvel permanecem selecionadas
    - Verifique se os totais refletem apenas serviços móveis
    - Clique em **"Gerar Fatura"** (mostra a contagem de transações móveis)
4. **Gerar a segunda fatura (Serviços de Internet):**

- Clique em **"+ Gerar Fatura Proforma"** novamente
- Título: "Serviços de Internet - Janeiro de 2025"
- Data de Início/Fim: Jan 1-31 (mesmo período)
- Na pré-visualização de transações:
  - Transações móveis já faturadas (não aparecem)
  - Exclua cobranças diversas usando o botão **X**
  - Apenas as transações de serviço de Internet permanecem
- Clique em **"Gerar Fatura"**

5. **Gerar a terceira fatura (Cobranças Adicionais):**
   - Clique em **"+ Gerar Fatura Proforma"** novamente
   - Título: "Cobranças Adicionais - Janeiro de 2025"
   - Apenas cobranças diversas não faturadas aparecem na pré-visualização
   - Clique em **"Selecionar Tudo"** para incluir todas
   - Clique em **"Gerar Fatura"**
6. Envie todas as três faturas para o cliente

Fluxo de Trabalho 3: Excluindo Transações Disputadas ou Pendentes

1. O final do período de faturamento chega
2. Navegue até a aba **Faturamento** do cliente
3. Clique em **"+ Gerar Fatura Proforma"**
4. Defina as datas do período de faturamento
5. A pré-visualização de transações mostra 20 transações
6. O cliente disputou uma cobrança e outra está pendente de investigação
7. Na pré-visualização de transações:
   - Localize a transação disputada (por exemplo, "Cobrança por excesso de dados")
   - Clique no botão **X** para excluí-la
   - Localize a transação pendente (por exemplo, "Taxa de instalação")
   - Clique no botão **X** para excluí-la
   - A contagem de transações atualiza: "18 Transações selecionadas"
   - Os totais recalculam automaticamente

8. Revise os totais atualizados (exclui valores disputados)
9. Clique em **"Gerar Fatura (18)"**
10. Fatura gerada com apenas transações aprovadas
11. Transações disputadas/pendentes permanecem não faturadas para o próximo ciclo de faturamento

Fluxo de Trabalho 4: Revisão Rápida da Fatura e Ajuste

1. A equipe gera a fatura mensal
2. A pré-visualização de transa��ões mostra um total inesperadamente alto
3. Revise cada transação na pré-visualização:
   - Note a cobrança duplicada pelo mesmo serviço
   - Clique em **X** para excluir a duplicata
   - Note a transação de teste que não deveria ser cobrada
   - Clique em **X** para excluir a transação de teste
4. Os totais atualizam em tempo real
5. Verifique se o novo total corresponde ao valor esperado
6. Clique em **"Gerar Fatura"** com as transações corrigidas
7. Volte e anule/exclua as transações excluídas, se necessário
8. Envie a fatura para o cliente com confiança

Fluxo de Trabalho 5: Fatura de Instalação Única

1. O técnico de campo completa a instalação
2. A equipe adiciona a transação de instalação manualmente
3. Navegue até a aba **Faturamento** do cliente
4. Clique em **"+ Gerar Fatura Proforma"**
5. Defina as datas:
   - Data de Início: hoje
   - Data de Término: hoje
   - Data de Vencimento: hoje + 7 dias
   - Título: "Serviços de Instalação"

6. A pré-visualização de transações mostra apenas as transações de hoje
7. Verifique se a cobrança de instalação aparece
8. Exclua quaisquer cobranças recorrentes usando o botão **X** (se presente)
9. Clique em **"Gerar Fatura"**
10. Envie para o cliente imediatamente
11. O cliente paga online via Stripe

Fluxo de Trabalho 6: Revisando a Fatura Antes do Contato com o Cliente

1. O cliente liga com uma pergunta de faturamento
2. A equipe navega até a lista de faturas do cliente
3. **Clique na linha da fatura** para abrir o modal de Detalhes da Fatura
4. Revise as informações da fatura:
    - ID da fatura, datas, status
    - Todas as transações incluídas com descrições
    - Detalhamento do imposto por transação
    - Subtotal, Imposto e Valores Totais
5. Responda às perguntas do cliente com detalhes exatos
6. Se o cliente solicitar o PDF, clique no botão **"Baixar PDF"** no modal
7. Se o cliente solicitar reenvio do e-mail, clique no botão **"Enviar E-mail"**
8. Feche o modal quando terminar

Fluxo de Trabalho 7: Corrigindo Erro de Faturamento

1. O cliente relata uma cobrança incorreta
2. A equipe clica na linha da fatura para ver os detalhes
3. Revê a lista de transações no modal de Detalhes da Fatura
4. Identifica a transação incorreta
5. A fatura está não paga, então pode ser anulada
6. Clique no botão **"Excluir"** no rodapé do modal
7. Confirme a anulação
8. As transações tornam-se não faturadas novamente

9. A equipe modifica ou remove a transação incorreta da lista de transações
10. Gere uma nova fatura com as transações corrigidas:
    - Use a pré-visualização de transações para excluir a transação corrigida, se necessário
    - Inclua apenas cobranças válidas
11. Envie a fatura corrigida para o cliente

Fluxo de Trabalho 8: Processando Múltiplos Pagamentos

1. O cliente traz dinheiro para pagar várias faturas
2. Navegue até a aba **Faturamento** do cliente
3. Veja as faturas não pagas
4. Clique na primeira linha da fatura para ver os detalhes
5. Verifique o valor e as transações
6. Clique no botão **"Pagar Fatura"** no rodapé do modal
7. Selecione o método de pagamento **"Dinheiro"**
8. Insira a referência: "Dinheiro pago em 2025-01-15"
9. Clique em **"Pagar Fatura"**
10. O modal se fecha, a fatura é marcada como "Paga"
11. Repita para as faturas restantes
12. Todas as faturas agora estão marcadas como "Pagas"

Fluxo de Trabalho 9: Lidando com Solicitação de Reembolso

1. O cliente solicita reembolso por pagamento em excesso
2. A equipe verifica se a fatura foi paga via Stripe
3. Navegue até a fatura na lista
4. Clique na linha da fatura para ver os detalhes
5. Verifique as informações de pagamento e o valor
6. Clique no botão **"Reembolsar"** no rodapé do modal (apenas aparece para faturas Stripe)
7. Confirme o reembolso

8. O sistema processa o reembolso da Stripe
9. O status da fatura muda para "Reembolsada"
10. O cliente recebe o reembolso em 3-7 dias úteis
11. A equipe acompanha com o cliente para confirmar o recebimento

Solução de Problemas

Não é possível gerar fatura - Nenhuma transação encontrada

- **Causa:** Nenhuma transação não faturada no intervalo de datas especificado
- **Correção:** Verifique a lista de transações, verifique se as transações existem e não estão já faturadas. Ajuste o intervalo de datas ou remova o filtro.

Falha na geração de PDF da fatura

- **Causa:** Erro de sintaxe no modelo, falha do WeasyPrint ou dados do cliente ausentes
- **Correção:** Verifique o HTML do modelo de fatura em busca de erros, verifique se os campos de endereço do cliente estão preenchidos, revise os logs da API.

Pagamento falha com erro da Stripe

- **Causa:** Cartão recusado, fundos insuficientes, cartão expirado ou problema na API da Stripe
- **Correção:** Tente um método de pagamento diferente, verifique se o cartão é válido, verifique o painel da Stripe para a razão da recusa.

Não é possível anular a fatura

- **Causa:** Fatura já paga
- **Correção:** Faturas pagas não podem ser anuladas. Se o reembolso for necessário, use a função de reembolso para faturas Stripe ou crie uma transação de crédito manualmente.

E-mail da fatura não enviado

- **Causa:** Credenciais da API Mailjet inválidas, cliente sem contato de faturamento ou modelo de e-mail ausente
- **Correção:** Verifique a configuração do Mailjet em `crm_config.yaml`, verifique os contatos do cliente, verifique se o modelo de e-mail da fatura existe.

Botão de reembolso não aparecendo

- **Causa:** Fatura paga em dinheiro/POS/transferência bancária (não Stripe), ou fatura não paga
- **Correção:** O botão de reembolso aparece apenas para pagamentos Stripe. Para outros métodos de pagamento, crie uma transação de crédito manual.

Download de PDF mostra design antigo do modelo

- **Causa:** PDF armazenado em cache antes da atualização do modelo
- **Correção:** Limpe o cache de PDF da fatura: `DELETE FROM Invoice_PDF_Cache WHERE invoice_id = X;`

Cliente não pode pagar fatura (sem métodos de pagamento)

- **Causa:** Nenhum método de pagamento salvo no portal de Autoatendimento
- **Correção:** O cliente deve adicionar um cartão de crédito na página **Métodos de Pagamento** antes de pagar faturas.

Múltiplas faturas geradas para o mesmo período

- **Causa:** A equipe gerou a fatura duas vezes, ou os intervalos de datas se sobrepõem
- **Correção:** Anule a fatura duplicada. Ajuste os intervalos de datas para evitar sobreposição. Use a pré-visualização de transações para garantir conjuntos de transações exclusivos.

Pré-visualização de transações não mostra transações

- **Causa:** Todas as transações no intervalo de datas já estão faturadas ou não existem transações
- **Correção:** Verifique se o intervalo de datas está correto. Verifique a lista de transações para confirmar se as transações existem. Filtre as faturas para ver qual fatura contém as transações.

Não é possível excluir transação da geração de fatura

- **Causa:** Transação já faturada ou problema no navegador
- **Correção:** Verifique se a transação aparece na pré-visualização com marca de verificação. Atualize a página e tente novamente. Limpe o cache do navegador se o problema persistir.

Total da fatura não corresponde ao valor esperado

- **Causa:** Transações inesperadas incluídas, imposto não calculado ou transações excluídas ainda contadas
- **Correção:** Revise a pré-visualização de transações com cuidado. Verifique o custo de varejo e o imposto de cada transação. Verifique se as transações excluídas estão esmaecidas. Verifique o distintivo de contagem de transações no botão Gerar Fatura.

Botão Gerar Fatura está desativado

- **Causa:** Nenhuma transação selecionada ou intervalo de datas inválido
- **Correção:** Certifique-se de que pelo menos uma transação esteja incluída (não excluída). Verifique se a Data de Início é anterior à Data de Término. Verifique se a Data de Vencimento está definida.

Modal de Detalhes da Fatura não abrindo

- **Causa:** Erro de JavaScript ou página não totalmente carregada
- **Correção:** Atualize a página. Verifique o console do navegador em busca de erros. Tente um navegador diferente. Verifique a conexão com a internet.

Imposto da transação não exibido nos Detalhes da Fatura

- **Causa:** Transação tem imposto de 0% ou tax_amount é nulo

- **Correção:** Verifique se a transação tem tax_percentage definida. Verifique se tax_amount foi calculado quando a transação foi criada. Atualize a transação se necessário.

Botões de ação ausentes no modal de Detalhes da Fatura

- **Causa:** A fatura está anulada ou o usuário não tem permissões
- **Correção:** Faturas anuladas mostram apenas o botão Baixar PDF. Verifique o status da fatura. Verifique o papel e as permissões do usuário.

Documentação Relacionada

- `integrations_mailjet` - Entrega de e-mail de fatura e modelos
- `administration_configuration` - Configuração do modelo de fatura
- `payments_transaction` - Criando transações que aparecem nas faturas
- `payments_process` - Processando pagamentos com métodos de pagamento armazenados
- `payment_system_guide` - Referência da API de pagamento e configuração de fornecedores

Gerenciamento de Métodos de Pagamento

O sistema de Métodos de Pagamento do OmniCRM permite que clientes e funcionários gerenciem cartões de pagamento de forma segura usando **processamento de pagamento de múltiplos fornecedores** (Stripe, PayPal, etc.). Os métodos de pagamento permitem faturamento automático para serviços, pagamentos únicos e cobranças recorrentes sem armazenar dados sensíveis do cartão no OmniCRM.

Veja também: `Guia do Sistema de Pagamento <payment_system_guide>`, `Visão Geral de Faturamento <billing_overview>`, `Processamento de Pagamento <payments_process>`, `Faturas <payments_invoices>`.

Visão Geral

O sistema de métodos de pagamento fornece:

- **Armazenamento Seguro de Cartões** - Cartões tokenizados por fornecedores de pagamento (Stripe, PayPal), nunca armazenados no OmniCRM
- **Suporte a Múltiplos Fornecedores** - Métodos de pagamento Stripe e PayPal suportados
- **Múltiplos Cartões** - Clientes podem armazenar múltiplos métodos de pagamento
- **Seleção Padrão** - Designar método de pagamento preferido para cobranças automáticas
- **Rastreamento de Expiração** - Monitorar e atualizar cartões que estão expirando
- **Autoatendimento** - Clientes podem gerenciar seus próprios cartões via `Portal de Autoatendimento <self_care_portal>`
- **Gerenciamento pela Equipe** - A equipe de suporte pode adicionar/remover cartões em nome dos clientes

Métodos de Pagamento Suportados:

- Cartões de Crédito (Visa, Mastercard, American Express, Discover)
- Cartões de Débito
- Cartões Pré-pagos (se suportados pela rede de cartões)

Não Armazenados no OmniCRM:

Os detalhes do cartão são tokenizados pelos fornecedores de pagamento e armazenados de forma segura. O OmniCRM armazena apenas:

- Fornecedor de pagamento (stripe, paypal)
- Marca do cartão (Visa, Mastercard, etc.)
- Últimos 4 dígitos
- Mês/Ano de expiração
- Nome/nome de apelido do titular do cartão
- Token de método de pagamento específico do fornecedor

Acessando Métodos de Pagamento

Da Página do Cliente:

1. Navegue até **Clientes → [Selecionar Cliente]**
2. Clique na aba **Faturamento**
3. Role até a seção **Métodos de Pagamento**

Ou diretamente:

Do Painel de Cartões que Estão Expirando:

Veja todos os clientes com cartões que estão expirando:

Isso mostra uma lista em todo o sistema de cartões que expiram dentro dos próximos 60 dias.

Lista de Métodos de Pagamento

A tabela de métodos de pagamento exibe todos os cartões armazenados para um cliente:

Descrições das Colunas:

- **Apelido** - Nome amigável para o cartão (por exemplo, "Cartão Pessoal", "Visa do Trabalho")
- **Emissor** - Marca do cartão e últimos 4 dígitos
- **Expiração** - Mês/ano de expiração (formato MM/AAAA)
- **Adicionado** - Data em que o cartão foi adicionado à conta
- **Padrão** - Marca de verificação indica o método de pagamento padrão para cobranças automáticas

Ações Por Cartão:

Cada linha tem um menu de ações (⋮) com opções:

- **Definir como Padrão** - Tornar este o método de pagamento padrão
- **Excluir** - Remover cartão da conta

Adicionando um Método de Pagamento

Clique em **"Adicionar Método de Pagamento"** para abrir o modal de pagamento seguro.

Passo 1: Inserir Detalhes do Cartão

O formulário de pagamento seguro aparece (alimentado por Stripe Elements ou PayPal SDK):

Campos Obrigatórios:

- **Informações do Cartão** - Número do cartão, expiração, CVC (validado pela Stripe)
- **Nome do Titular do Cartão** - Nome no cartão
- **País/Região** - País de cobrança

Campos Opcionais:

- **Apelido do Cartão** - Rótulo amigável para distinguir entre cartões

Segurança:

- Detalhes do cartão inseridos diretamente em iframe seguro hospedado pelo fornecedor (Stripe Elements / PayPal SDK)
- O OmniCRM nunca vê ou armazena números de cartão completos
- Conformidade com PCI DSS tratada pelo fornecedor de pagamento
- Validação em tempo real evita números de cartão inválidos

Passo 2: Enviar e Tokenizar

Quando você clicar em **"Adicionar Método de Pagamento"**:

1. **Validação do Lado do Cliente:**
    - O fornecedor de pagamento valida o formato do

Visão Geral da Aba de Faturamento

A aba de Faturamento fornece uma visão unificada de todas as informações financeiras de um cliente, combinando métodos de pagamento, transações e faturas em uma única interface para uma gestão de faturamento eficiente.

Documentação relacionada: `Payment Methods <payment_methods>`, `Transactions <payments_transaction>`, `Invoices <payments_invoices>`, `Payment Processing <payments_process>`.

Acessando a Aba de Faturamento

Visão por Cliente:

1. Navegue até **Customers → [Select Customer]**
2. Clique na aba **Billing**
3. Veja todas as três seções: Métodos de Pagamento, Transações e Faturas

Visões do Sistema:

Os dados de faturamento do sistema podem ser acessados separadamente:

- **Billing → Transactions** - Todas as transações de todos os clientes
- **Billing → Invoices** - Todas as faturas de todos os clientes

Portal de Autoatendimento:

Clientes que acessam o `Self-Care Portal <self_care_portal>` veem a mesma estrutura da aba de Faturamento:

- Visualizar e gerenciar seus métodos de pagamento
- Visualizar histórico de transações
- Visualizar e pagar faturas online

Estrutura da Aba de Faturamento

A aba de Faturamento está organizada em três seções principais, exibidas como cartões:

Seção 1: Métodos de Pagamento

Objetivo: Gerenciar como o cliente paga pelos serviços

Principais Recursos:

- Visualizar todos os cartões de crédito salvos
- Definir método de pagamento padrão
- Adicionar novos métodos de pagamento (via Stripe)
- Remover cartões expirados ou não utilizados

Documentação: `basics_payment`

Seção 2: Transações

Objetivo: Rastrear todas as cobranças e créditos do cliente

Principais Recursos:

- Visualizar estatísticas de transações (Total, Não Faturado, Este Mês, Último Mês)
- Listar todas as transações com filtragem por status de anulação/faturamento
- Adicionar transações manuais (cobranças ou créditos)
- Anular transações incorretas
- Ver quais transações estão faturadas vs não faturadas

Documentação: `payments_transaction`

Seção 3: Faturas

Objetivo: Agrupar transações em faturas para o cliente pagar

Principais Recursos:

- Visualizar estatísticas de faturas (Total, Não Pagas, Este Mês, Último Mês)

- Listar todas as faturas com filtragem por status de pagas/anuladas
- Gerar novas faturas a partir de transações não faturadas
- Baixar PDFs de faturas
- Enviar faturas por e-mail para os clientes
- Pagar faturas online (métodos de pagamento Stripe ou manuais)

- Anular ou reembolsar faturas

Documentação: `payments_invoices`

Fluxo de Dados Entre Seções

Entender como os dados fluem entre as três seções é crucial para uma gestão de faturamento eficaz.

Diagrama de Fluxo

Relação Transação → Fatura

1. Criação de Transação:

Quando um serviço é provisionado ou uma cobrança manual é adicionada:

- Transação criada na seção **Transactions**
- Status da transação: **Não Faturado**
- Campo `invoice_id` da transação é `null`

Exemplo:

2. Geração de Fatura:

Quando a equipe gera uma fatura:

- Fatura criada na seção **Invoices**
- Todas as transações não faturadas dentro do intervalo de datas agrupadas em uma fatura
- Campo `invoice_id` da transação populado
- Status da transação muda para: **Faturado**

Exemplo:

3. Atualização de Estatísticas de Transação:

- Total de **Transações Não Faturadas** diminui
- Estatística de **Total de Faturas** aumenta
- Total de **Faturas Não Pagas** aumenta

Relação Fatura → Pagamento

1. Pagamento de Fatura:

Quando o cliente paga a fatura:

- Pagamento processado usando **Método de Pagamento** salvo (cartão Stripe)
- Ou método de pagamento manual selecionado (dinheiro, POS, transferência bancária)
- Status da fatura muda para: **Paga**

2. Transação de Pagamento Criada:

Para pagamentos manuais (não Stripe):

- Transação negativa criada automaticamente
- Título da transação: "Pagamento pela Fatura #1234"
- Valor da transação: -$45.00 (negativo, credita o cliente)
- Campo `invoice_id` da transação: Link para a fatura paga

Exemplo:

3. Atualização de Estatísticas:

- Total de **Faturas Não Pagas** diminui
- **Total de Faturas Este Mês** inalterado (fatura já existia)

Relação Método de Pagamento → Fatura

Fluxo de Pagamento Stripe:

1. Cliente adiciona cartão de crédito em **Payment Methods**
2. Cartão tokenizado via Stripe, armazenado com segurança
3. Ao pagar a fatura, o cliente seleciona o cartão salvo
4. Stripe cobra o cartão
5. Fatura marcada como paga
6. Campo `payment_reference` populado com o ID da intenção de pagamento do Stripe

Fluxo de Pagamento Manual:

1. Cliente paga via dinheiro/POS/transferência bancária (nenhum método de pagamento necessário)
2. A equipe seleciona o método de pagamento no modal Pagar Fatura
3. A equipe insere o número de referência (opcional)
4. Transação negativa criada para o valor do pagamento
5. Fatura marcada como paga

Fluxos de Trabalho de Faturamento Completo

Esses fluxos de trabalho demonstram como as três seções trabalham juntas para realizar tarefas comuns.

Fluxo de Trabalho 1: Configuração de Novo Cliente e Primeira Fatura

Objetivo: Configurar faturamento para novo cliente e coletar o primeiro pagamento

1. **Adicionar Método de Pagamento:**
   - Navegue até o cliente → aba **Billing**
   - Seção **Payment Methods** → Clique em **"Add Payment Method"**
   - Cliente adiciona cartão de crédito via Stripe
   - Cartão salvo como método de pagamento padrão
2. **Verificar Transações:**
   - Seção **Transactions** mostra transações não faturadas:
     - Taxa de configuração de serviço: $50.00
     - Serviço do primeiro mês: $45.00
     - Total Não Faturado: $95.00
3. **Gerar Fatura:**
   - Seção **Invoices** → Clique em **"Generate Proforma Invoice"**
   - Defina o intervalo de datas para incluir configuração e primeiro mês
   - Clique em **"Generate Invoice"**
   - Fatura #INV-2025-001234 criada por $95.00
4. **Atualização de Transações:**
   - Ambas as transações agora mostram: Fatura #INV-2025-001234
   - Total de **Transações Não Faturadas** agora $0.00
5. **Enviar Fatura por E-mail:**
   - Clique no ícone de e-mail ao lado da fatura
   - Cliente recebe e-mail da fatura com PDF e link para pagamento
6. **Cliente Paga Online:**

- Cliente clica no link de pagamento no e-mail
- Redirecionado para o portal de Autoatendimento
- Clique no botão **"Pay Invoice"**
- Selecione o método de pagamento padrão
- Clique em **"Pay Invoice"**
- Stripe cobra o cartão

7. **Atualização da Fatura:**
   - Status da fatura muda para **"Paid"**
   - Total de **Faturas Não Pagas** diminui em $95.00

Resultado: Cliente totalmente configurado com método de pagamento, primeira fatura paga.

Fluxo de Trabalho 2: Faturamento Recorrente Mensal

Objetivo: Faturar todos os clientes pelo serviço mensal no final do mês

1. **Serviços Auto-Cobrados:**
   - O final do mês chega (31 de janeiro)
   - O sistema de faturamento cria automaticamente transações para todos os serviços recorrentes
   - Seção **Transactions** mostra novas transações não faturadas
2. **Revisar Transações Não Faturadas:**
   - Navegue até a seção **Transactions**
   - Filtrar: **Invoice Status: Not Invoiced**
   - Revisar lista de todas as transações prontas para faturamento
   - Verificar se os valores e descrições estão corretos
3. **Gerar Faturas:**
   - Navegue até **Billing → Invoices** (em todo o sistema)
   - Para cada cliente (ou use o processo em lote):
     - Clique em **"Generate Proforma Invoice"**
     - Selecione o cliente
     - Data de Início: 2025-01-01
     - Data de Fim: 2025-01-31

        - Data de Vencimento: 2025-02-15
        - Clique em **"Generate Invoice"**
4. **Atualização de Transações:**
    - Todas as transações agora vinculadas a faturas
    - Totais de **Transações Não Faturadas** redefinidos para $0.00
5. **Enviar Todas as Faturas por E-mail:**
    - Para cada fatura, clique no ícone de e-mail
    - Todos os clientes recebem faturas mensais
6. **Clientes Pagam:**
    - Clientes com métodos de pagamento salvos pagam online via Autoatendimento
    - A equipe processa pagamentos em dinheiro/POS para clientes que pagam pessoalmente
    - Total de **Faturas Não Pagas** diminui à medida que os pagamentos são recebidos

Resultado: Todos os clientes faturados por janeiro, faturas enviadas, pagamentos processados.

Fluxo de Trabalho 3: Tratamento de Crédito por Problema de Serviço

Objetivo: Creditar o cliente por interrupção de serviço, aplicar a fatura não paga

1. **Cliente Relata Problema:**
    - O serviço ficou fora do ar por 2 dias
    - Cliente merece crédito de $10
2. **Adicionar Transação de Crédito:**
    - Navegue até o cliente → aba **Billing** → seção **Transactions**
    - Clique em **"+ Add Transaction"**
    - Tipo de Transação: **Credit**
    - Tipo de Crédito: **Cash Payment** (ou tipo apropriado)
    - Título: "Crédito por Interrupção de Serviço"
    - Descrição: "Compensação por interrupção de 2 dias 8-9 Jan"

- Custo de Varejo: 10.00
- Clique em **"Add Transaction"**

3. **Transação Criada:**
   - Transação aparece na lista com valor: **-$10.00**
   - Status da transação: **Não Faturado**
   - Total de **Transações Não Faturadas** agora inclui -$10.00
4. **Aplicar à Fatura:**
   - Se o cliente já tiver uma fatura não paga:
     - Fatura permanece não paga com o valor original
     - Crédito será aplicado à próxima geração de fatura
   - Se gerando nova fatura:
     - Seção **Invoices** → Clique em **"Generate Proforma Invoice"**
     - Incluir intervalo de datas com a transação de crédito
     - Fatura gerada com crédito aplicado:
5. **Cliente Paga:**
   - Cliente paga valor reduzido: $35.00
   - Fatura marcada como paga

Resultado: Cliente creditado pela interrupção, crédito aplicado à próxima fatura, pagamento menor coletado.

Fluxo de Trabalho 4: Método de Pagamento Expirado - Atualizar e Tentar Novamente

Objetivo: O cartão do cliente expirou, causando falha no pagamento - atualizar cartão e tentar pagamento novamente

1. **Notificação de Falha de Pagamento:**
   - Cliente tenta pagar a fatura
   - Stripe retorna erro: "Cartão expirado"
   - Pagamento falha, fatura permanece não paga
2. **Atualizar Método de Pagamento:**
   - Cliente navega até a aba **Billing**
   - Seção **Payment Methods** → Clique em **"Add Payment Method"**
   - Insira os novos dados do cartão (data de expiração atualizada)

- Novo cartão salvo

3. **Definir como Padrão:**
   - Cliente clica em **"Set as Default"** no novo cartão
   - Cartão antigo removido automaticamente (se desejado)
4. **Tentar Pagamento Novamente:**
   - Navegue até a seção **Invoices**
   - Localize a fatura não paga
   - Clique no ícone **"Pay"**
   - Modal de pagamento abre com o novo cartão padrão pré-selecionado
   - Clique em **"Pay Invoice"**
   - Stripe cobra o novo cartão com sucesso
5. **Atualização da Fatura:**
   - Status da fatura muda para **"Paid"**
   - Campo `payment_reference` populado com o novo ID da intenção de pagamento do Stripe

Resultado: Cliente atualizou o método de pagamento, fatura paga com sucesso com o novo cartão.

Fluxo de Trabalho 5: Anulando Fatura Incorreta e Re-Faturamento

Objetivo: Fatura gerada pela equipe com transações erradas - anular e regenerar corretamente

1. **Erro Descoberto:**
   - Fatura #INV-2025-001234 gerada com intervalo de datas errado
   - Incluiu transações do mês errado
   - Fatura está não paga
2. **Anular Fatura:**
   - Navegue até a aba **Billing** → seção **Invoices**
   - Localize a fatura incorreta
   - Clique no ícone de deletar (🗑)
   - Confirme a anulação
   - Fatura anulada

3. **Transações Liberadas:**
    - Navegue até a seção **Transactions**
    - Todas as transações da fatura anulada agora mostram: **Não Faturadas**
    - Total de **Transações Não Faturadas** aumenta
    - Transações disponíveis para nova fatura
4. **Gerar Fatura Correta:**
    - Seção **Invoices** → Clique em **"Generate Proforma Invoice"**
    - Defina o intervalo de datas correto
    - Aplique filtro se necessário (por exemplo, "Mobile" para fatura apenas móvel)
    - Clique em **"Generate Invoice"**
    - Nova fatura criada com transações corretas
5. **Verificar e Enviar por E-mail:**
    - Revise os detalhes da nova fatura
    - Verifique se as transações corretas estão incluídas
    - Clique no ícone de e-mail para enviar ao cliente

Resultado: Fatura incorreta anulada, transações re-faturadas corretamente, cliente recebe fatura corrigida.

Fluxo de Trabalho 6: Processando Pagamento em Dinheiro para Múltiplas Faturas

Objetivo: Cliente paga múltiplas faturas não pagas com um único pagamento em dinheiro

1. **Cliente Chega com Dinheiro:**
    - Cliente traz $300 em dinheiro para pagar faturas pendentes
    - Navegue até o cliente → aba **Billing**
2. **Revisar Faturas Não Pagas:**
    - Seção **Invoices** → Filtrar: **Paid: Not yet Paid**
    - Visualizar faturas não pagas:
3. **Pagar Primeira Fatura:**
    - Clique no ícone de pagamento na fatura #1234
    - Modal de pagamento abre

- Selecione o método de pagamento **"Cash"**
- Insira a referência: "Dinheiro pago 2025-02-10 - Recibo #001"
- Clique em **"Pay Invoice"**
- Fatura #1234 marcada como **"Paid"**

4. **Pagar Faturas Restantes:**
   - Repita o processo para a fatura #1235:
     - Referência: "Dinheiro pago 2025-02-10 - Recibo #001"
   - Repita para a fatura #1236:
     - Referência: "Dinheiro pago 2025-02-10 - Recibo #001"
5. **Verificar Transações:**
   - Navegue até a seção **Transactions**
   - Três novas transações de pagamento criadas:
   - Todas vinculadas às respectivas faturas
6. **Atualizar Estatísticas:**
   - Seção **Invoices** → Total de **Faturas Não Pagas** diminuído em $300.00
   - Todas as faturas agora mostram status **"Paid"**

Resultado: Cliente pagou todas as faturas pendentes em dinheiro, transações de pagamento registradas com referência de recibo.

Melhores Práticas

Para Usuários da Equipe

Gerenciamento de Transações:

- Adicione transações manuais imediatamente (não atrase)
- Use títulos e descrições descritivas para clareza
- Vincule transações a serviços e locais quando aplicável
- Anule transações incorretas antes que sejam faturadas

Geração de Faturas:

- Gere faturas em intervalos consistentes (por exemplo, mensalmente no 1º do mês)

- Use intervalos de datas com cuidado para evitar sobreposições ou lacunas
- Use filtros para criar faturas específicas de serviços quando necessário
- Envie faturas por e-mail imediatamente após a geração
- Revise PDFs de faturas antes de enviar aos clientes

Processamento de Pagamentos:

- Verifique se o método de pagamento é válido antes de tentar a cobrança
- Sempre insira números de referência para pagamentos manuais (dinheiro, POS, transferência bancária)
- Marque faturas como pagas imediatamente após receber o pagamento
- Reembolse apenas via Stripe (crie crédito manual para outros métodos de pagamento)

Higiene de Dados:

- Revise regularmente transações não faturadas
- Investigue transações anuladas para entender erros de faturamento
- Monitore faturas não pagas e siga com os clientes
- Mantenha métodos de pagamento atualizados (remova cartões expirados)

Para Clientes (Portal de Autoatendimento)

Métodos de Pagamento:

- Mantenha pelo menos um método de pagamento válido em arquivo
- Atualize métodos de pagamento antes que os cartões expirem
- Defina seu cartão preferido como padrão

Pagamentos de Faturas:

- Pague faturas antes da data de vencimento para evitar taxas de atraso
- Revise os detalhes da fatura e transações antes de pagar
- Baixe PDFs de faturas para seus registros
- Entre em contato com o suporte imediatamente se a fatura parecer incorreta

Revisão de Transações:

- Revise regularmente o histórico de transações
- Relate qualquer cobrança inesperada imediatamente
- Entenda quais transações estão faturadas vs não faturadas

Para Administradores

Configuração do Sistema:

- Configure modelos de e-mail Mailjet para entrega profissional de faturas
- Personalize modelos de PDF de faturas para corresponder à marca
- Configure integração de fornecedores de pagamento (Stripe, PayPal) para processamento seguro de pagamentos
- Configure termos de pagamento e datas de vencimento

Monitoramento e Relatórios:

- Use widgets de estatísticas para monitorar a saúde do faturamento
- Acompanhe o total de **Transações Não Faturadas** - deve diminuir após o ciclo de faturamento
- Monitore o total de **Faturas Não Pagas** - siga os pagamentos em atraso
- Revise estatísticas de **Este Mês** vs **Último Mês** para tendências

Automação:

- Automatize cobranças de serviços recorrentes via configuração de produtos
- Configure geração automática de faturas para faturamento recorrente (se disponível)
- Configure lembretes por e-mail para faturas em atraso

Problemas Comuns e Soluções

Problema: Cliente Não Consegue Pagar a Fatura

Sintomas:

- Cliente clica no botão de pagamento, mas nada acontece
- Mensagem de erro: "Nenhum método de pagamento encontrado"

Diagnóstico:

1. Navegue até o cliente → aba **Billing** → seção **Payment Methods**
2. Verifique se o cliente tem algum método de pagamento salvo
3. Verifique se os cartões salvos estão expirados

Solução:

- O cliente deve adicionar um método de pagamento válido antes de pagar faturas
- Oriente o cliente a ir para a página **Payment Methods** para adicionar um cartão de crédito
- Verifique se o cartão é aceito (Visa, Mastercard, Amex, etc.)
- Tente o pagamento novamente após o cartão ser adicionado

Problema: Fatura Gerada com Transações Erradas

Sintomas:

- Fatura inclui transações do período errado
- Fatura faltando transações esperadas
- Total da fatura incorreto

Diagnóstico:

1. Abra a fatura na seção **Invoices**

2. Revise as transações incluídas na fatura
3. Verifique as datas das transações vs intervalo de datas da fatura
4. Verifique se um filtro foi aplicado durante a geração

Solução:

- **Se a fatura estiver não paga:** Anule a fatura, verifique se as transações estão não faturadas, regenere com o intervalo de datas correto
- **Se a fatura estiver paga:** Não é possível anular - crie uma transação de crédito para o valor incorreto, gere uma fatura corrigida
- **Prevenção:** Sempre revise a seção **Transactions** antes de gerar a fatura para verificar se as transações corretas serão incluídas

Problema: Total de Transações Não Faturadas Não Diminui

Sintomas:

- Widget de **Transações Não Faturadas** mostra um valor alto
- Lista de transações mostra muitas transações não faturadas
- Faturas mensais já foram geradas

Diagnóstico:

1. Filtrar transações por **Invoice Status: Not Invoiced**
2. Revisar lista de transações não faturadas
3. Verifique as datas das transações - podem ser cobranças recentes após a última geração de fatura
4. Verifique se algumas transações foram anuladas (não devem contar para o total não faturado)

Solução:

- **Comportamento esperado:** Transações criadas após a última geração de fatura permanecem não faturadas até o próximo ciclo de faturamento
- **Se transações antigas estiverem não faturadas:** Gere uma fatura para essas transações com o intervalo de datas apropriado

- **Se transações anuladas contadas:** O sistema deve excluir transações anuladas automaticamente - relate um bug se não o fizer

Problema: Método de Pagamento Continua Sendo Recusado

Sintomas:

- Pagamento Stripe falha com "Cartão recusado"
- Cliente relata que o cartão deve funcionar
- Múltiplas tentativas de reembolso falham

Diagnóstico:

1. Verifique o painel do Stripe para a razão da recusa
2. Razões comuns para recusa:
    - Fundos insuficientes
    - Cartão expirado
    - Cart��o relatado como perdido/roubado
    - Prevenção de fraude bancária
    - Cartão internacional bloqueado

Solução:

- Peça ao cliente para entrar em contato com o banco para autorizar o pagamento
- Tente um método de pagamento diferente (cartão diferente)
- Para a equipe: Processar como pagamento manual (dinheiro, POS) e marcar fatura como paga
- Verifique se o endereço de cobrança corresponde ao endereço de cobrança do cartão

Atalhos de Teclado e Dicas

Navegação:

- Acesse rapidamente a aba de faturamento: Página do cliente → tecla **B** (se os atalhos de teclado estiverem habilitados)

Filtragem:

- **Transactions:** Clique no filtro **"Not Invoiced"** antes de gerar faturas para ver exatamente o que será incluído
- **Invoices:** Clique no filtro **"Not yet Paid"** para ver todas as faturas pendentes que requerem acompanhamento

Operações em Lote:

- Selecione várias faturas (caixa de seleção) para excluir/anular várias de uma vez (apenas para a equipe)
- Use a barra de pesquisa para encontrar rapidamente uma transação ou fatura específica pelo ID

Ações Rápidas:

- Clique no título da transação/fatura para ver todos os detalhes em um modal
- Menu de ações de clique direito (⋮) para acesso rápido a opções de download/enviar por e-mail/pagar

Documentação Relacionada

- `basics_payment` - Métodos de pagamento e integração de pagamento multi-fornecedor
- `payment_system_guide` - Referência da API de pagamento e configuração de fornecedor
- `payments_transaction` - Transações em detalhes
- `payments_invoices` - Faturas em detalhes
- `integrations_mailjet` - Entrega de faturas por e-mail
- `csa_activity_log` - Visualizando o histórico de atividades de faturamento
- Monitoring & Metrics - Métricas e monitoramento do sistema de faturamento OCS/CGRateS

Ações e Comportamentos de Recarga do CGRateS

Este guia explica como as Ações do CGRateS funcionam dentro do OmniCRM, focando especificamente na gestão de saldo, comportamentos de recarga e como diferentes tipos de ações afetam produtos adicionais.

Visão Geral

No Sistema de Cobrança Online (CGRateS) do OmniCRM, **Ações** são o mecanismo para adicionar, modificar ou remover saldos nas contas dos clientes.

Quando você provisiona um produto adicional ou de recarga, na verdade, está executando uma Ação do CGRateS que manipula os saldos da conta. (Você pode adotar outras abordagens também, como adicionar manualmente saldos às contas a partir de Playbooks - este é apenas um padrão comum que usamos para manter as coisas organizadas)

Conceitos Chave

Ação - Um conjunto de operações a serem realizadas em uma conta (adicionar saldo, deduzir saldo, registrar CDR, etc.)

Saldo - Uma quantidade de um recurso (dados, voz, SMS, monetário) com um tempo de expiração e peso

ID do Saldo - Um identificador único para um tipo de saldo (por exemplo, "Pacote de Dados", "Minutos de Voz")

Peso - Prioridade para consumo de saldo (peso maior consumido primeiro)

Tempo de Expiração - Quando o saldo expira (data absoluta ou relativa como "+5 dias")

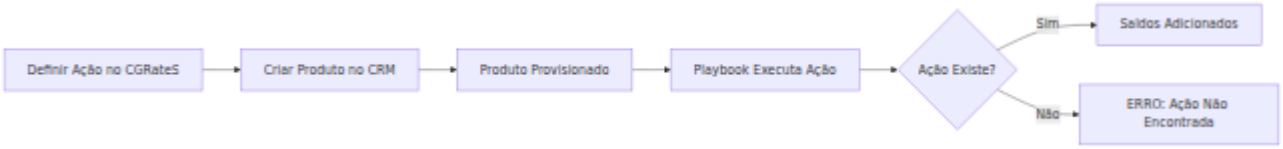
Bloqueador - Uma bandeira especial de saldo que bloqueia todo o uso quando o saldo chega a zero, mesmo que outros saldos existam (veja Bloqueadores de Saldo)

Crítico: Ações Devem Ser Definidas Primeiro

Antes que você possa executar uma Ação em um playbook usando `ExecuteAction`, essa Ação deve já estar definida no CGRateS. Este é um pré-requisito crítico que muitas vezes é negligenciado.

Quando as Ações São Definidas

As Ações são tipicamente definidas durante a **configuração inicial do sistema** ou **configuração do produto**, NÃO durante o provisionamento. Elas são geralmente criadas através de scripts Python que configuram simultaneamente o CRM e o OCS.



Como as Ações Estão Ligadas aos Produtos

As Ações estão ligadas aos produtos por meio de uma convenção de nomenclatura:

- **Ação CGRateS:** `ActionsId = "Action_50gb-data-pack"`
- **Produto CRM:** `product_slug = "50gb-data-pack"`
- **No Playbook:** `cgr_action_name = "Action_" + product_slug`

Quando o playbook é executado, ele constrói o nome da Ação a partir do product_slug e chama `ExecuteAction`. Se essa Ação não existir no CGRateS, o

provisionamento falha.

Onde Definir Ações

As Ações devem ser definidas em seus scripts de configuração de produtos, tipicamente:

1. **Durante a Configuração Inicial** - Ao configurar seu sistema pela primeira vez
2. **Ao Criar Novos Produtos** - Defina a Ação antes de criar o produto
3. **Via Scripts de Configuração** - Scripts Python que configuram tanto o OCS quanto o CRM

Exemplo: Definindo uma Ação antes de criar o produto

```python
import cgrateshttpapi

OCS_Obj = cgrateshttpapi.CGRateS("ocs.example.com", "2080")
tenant = "your_tenant"

# Passo 1: Defina a Ação no CGRateS PRIMEIRO
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"

# Passo 2: Agora crie o produto no CRM
# (product_slug = "50gb-data-pack" estará vinculado a Action_50gb-data-pack)
```

O Que Acontece Se a Ação Não Existir:

```
# No playbook
- name: Executar Ação
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body:
      {
        "method": "APIerSv1.ExecuteAction",
        "params": [{
          "ActionsId": "Action_50gb-data-pack"
        }]
      }
  register: response

# Resultado se a Ação não estiver definida:
# response.json.error = "SERVER_ERROR: Ação não encontrada"
# O provisionamento falha, o cliente não recebe saldo
```

Para detalhes completos sobre como definir Ações e vinculá-las a produtos, veja Definindo Produtos.

Independência de Saldo

Por padrão, produtos adicionais criam **saldos independentes** que funcionam separadamente uns dos outros. Isso significa:

- Você pode ter vários addons ativos simultaneamente
- Cada addon mantém seu próprio saldo e expiração
- Saldos são consumidos com base em regras de peso e expiração

Exemplo: Múltiplos Addons Independentes

```
# O cliente tem estes saldos ativos:
Saldo 1:
  ID: "Data_5GB_5days_uuid_abc123"
  Tipo: *data
  Valor: 5368709120  # 5GB em bytes
  Expiração: 2024-12-29
  Peso: 10

Saldo 2:
  ID: "Data_10GB_30days_uuid_def456"
  Tipo: *data
  Valor: 10737418240  # 10GB em bytes
  Expiração: 2025-01-24
  Peso: 10

# Ambos os saldos coexistem de forma independente
# O sistema consome com base no peso; se o peso for igual, a ordem não é garantida
```

Quando os saldos têm o mesmo peso e correspondem ao mesmo destino, o CGRateS não garante a ordem de consumo com base na expiração - a ordem depende de como os saldos são armazenados e recuperados. **Use pesos diferentes para controlar a ordem de consumo**.

Tipos de Ação: *topup vs *topup_reset

O tipo de ação determina como novos saldos interagem com saldos existentes do mesmo ID.

*topup - Comportamento Aditivo

A ação `*topup` **adiciona aos saldos existentes** com o mesmo ID de Saldo e **estende o tempo de expiração**.

Comportamento:

- Encontra saldo existente com ID correspondente
- Adiciona novo valor ao valor existente
- Atualiza a expiração para o novo tempo de expiração
- Preserva o saldo existente (rollover)

Exemplo:

```yaml
# Estado inicial:
Saldo:
  ID: "Data_Package__5368709120"
  Valor: 1073741824  # 1GB restante
  Expiração: 2024-12-24 (1 dia restante)

# Executa a ação *topup com:
Ação:
  Identificador: "*topup"
  Saldo:
    ID: "Data_Package__5368709120"  # Mesmo ID - aciona rollover
    Valor: 5368709120  # 5GB
    ExpiryTime: "+5d"

# Resultado após *topup:
Saldo:
  ID: "Data_Package__5368709120"
  Valor: 6442450944  # 6GB (1GB + 5GB rolados)
  Valor_Original: 5368709120  # Ainda mostra os originais 5GB
  Valor_hr: "6 GB"
  Valor_Original_hr: "5 GB"
  Restante_hr: "6 GB de 5 GB (1 GB rolado)"
  Expiração: 2024-12-29 (5 dias a partir de agora)
```

Casos de Uso:

- Recompensas de fidelidade (adicionar dados extras ao pacote existente)
- Créditos de compensação (adicionar ao saldo do cliente)
- Pacotes de dados rollover
- Extensões de período de carência

*topup_reset - Comportamento de Redefinição

A ação `*topup_reset` **substitui saldos existentes** com o mesmo ID de Saldo.

Comportamento:

- Encontra saldo existente com ID correspondente
- Descartar valor antigo (sem rollover)
- Define saldo para novo valor apenas
- Atualiza expiração para novo tempo de expiração

Exemplo:

```
# Estado inicial:
Saldo:
  ID: "Data_Package__5368709120"
  Valor: 1073741824  # 1GB restante
  Expiração: 2024-12-24 (1 dia restante)

# Executa a ação *topup_reset com:
Ação:
  Identificador: "*topup_reset"
  Saldo:
    ID: "Data_Package__5368709120"  # Mesmo ID - aciona
redefinição
    Valor: 5368709120  # 5GB
    ExpiryTime: "+5d"

# Resultado após *topup_reset:
Saldo:
  ID: "Data_Package__5368709120"
  Valor: 5368709120  # 5GB (antigo 1GB descartado)
  Valor_Original: 5368709120
  Valor_hr: "5 GB"
  Restante_hr: "5 GB de 5 GB"
  PercentualUsado: 0
  Expiração: 2024-12-29 (5 dias a partir de agora)
```

Casos de Uso:

- Pacotes mensais recorrentes (redefinir para o valor total a cada mês)

- Recargas de tamanho fixo (sempre receber valor exato)
- Mudanças de plano (substituir saldo do plano antigo pelo novo plano)
- Prevenir abuso (não pode empilhar addons ilimitados)

Controlando o Comportamento do Saldo com IDs de Saldo

O **ID do Saldo** é crucial para determinar se os saldos são independentes ou interagem entre si.

Convenção de Nomenclatura do ID do Saldo e Visualizações Legíveis por Humanos

O OmniCRM usa uma convenção de nomenclatura específica para IDs de Saldo que codifica tanto o nome descritivo quanto o tamanho original. Isso permite que a API gere automaticamente campos legíveis por humanos para a interface web.

Padrão do ID do Saldo:

```
{NomeDescritivo}__{TamanhoOriginalEmUnidadesBase}
```

Exemplos de IDs de Saldo:

```
# Saldo de dados: 100GB
"AU_Data_Domestic__107374182400"
# Desagrega para:
# - Parte descritiva: "AU_Data_Domestic" (O QUE é - tipo/destino)
# - Separador: "__" (duplo sublinhado)
# - Tamanho original: "107374182400" (100GB em bytes)
# - UI mostra: "AU Data Domestic - 100 GB"

# Saldo de voz: 3000 minutos
"AU_Voice_Domestic__1800000000000000"
# - Parte descritiva: "AU_Voice_Domestic" (NÃO
"AU_Voice_Domestic_3000min")
# - Tamanho original: "1800000000000000" (3000 minutos em
nanossegundos)
# - UI mostra: "AU Voice Domestic - 3000 min"

# Saldo de SMS: 3000 mensagens
"AU_SMS_Domestic__3000"
# - Parte descritiva: "AU_SMS_Domestic"
# - Tamanho original: "3000" (contagem)
# - UI mostra: "AU SMS Domestic - 3000 msgs"
```

Importante: Não inclua informações de tamanho na parte descritiva - é redundante, uma vez que o tamanho está codificado após `__` e a API o converte automaticamente para o formato legível por humanos.

Como a API Cria Visualizações Legíveis por Humanos:

Quando a API do OmniCRM recupera dados de saldo do CGRateS, ela automaticamente analisa o ID do Saldo e gera campos `_hr` (legíveis por humanos):

```
{
  "BalanceMap": {
    "*data": [
      {
        "ID": "AU_Data_Domestic__107374182400",
        "Value": 53687091200,
        "ExpiryTime": "2025-01-25T23:59:59Z",
        "Weight": 1200,

        // Campos legíveis por humanos gerados automaticamente:
        "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 dias)"
      }
    ]
  }
}
```

Lógica de Processamento da API:

1. **Analisar ID do Saldo:**

```
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. **Gerar Nome Descritivo Legível por Humanos:**

```
# Substituir sublinhados por espaços
id_hr = descriptive_name.replace("_", " ")  # "AU Data Domestic"
```

3. **Converter Tamanho Original para Unidades Humanas:**

```
# Para saldos de dados (bytes)
if balance_type == "*data":
    original_value_hr = convert_bytes_to_gb(original_size)  #
"100 GB"

# Para saldos de voz (nanossegundos)
elif balance_type == "*voice":
    original_value_hr = convert_ns_to_minutes(original_size)  #
"3000 min"

# Para saldos de SMS (contagem)
elif balance_type == "*sms":
    original_value_hr = f"{original_size} msgs"  # "3000 msgs"
```

4. **Calcular Percentual de Uso:**

```
if original_size and original_size > 0:
    percent_used = ((original_size - current_value) /
original_size) * 100
```

5. **Formatar Exibição Restante:**

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

Exibição na UI Web:

O frontend usa esses campos `_hr` para exibir informações de saldo amigáveis ao usuário:

```
// Em vez de mostrar valores brutos:
// ID: "AU_Data_Domestic__107374182400"
// Valor: 53687091200

// Mostrar legível por humanos:
<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 dias)" */}
</BalanceCard>
```

Por Que Isso Importa:

1. **Rastreamento de Tamanho Original** - Mesmo quando os saldos são parcialmente consumidos, a UI pode mostrar "50 GB de 100 GB" em vez de apenas "50 GB restantes"

2. **Visualização de Progresso** - Cálculos de percentual permitem barras de progresso precisas

3. **Nomenclatura Consistente** - Nomes descritivos extraídos de IDs de Saldo garantem consistência entre backend e frontend

4. **Exibição de Rollover** - Ao usar `*topup` (rollover), se um cliente tiver 70 GB restantes e recarregar 100 GB:
   - O ID do Saldo permanece: `"AU_Data_Domestic__107374182400"` (originais 100 GB)
   - O valor atual se torna: 170 GB
   - A UI mostra: "170 GB (70 GB rolados + 100 GB novos)"

Melhores Práticas - Criando IDs de Saldo:

Sempre inclua o tamanho original após `__` para exibição adequada na UI. Não duplique informações de tamanho no nome descritivo:

```
# Bom - nome descritivo + tamanho em unidades base
Action_Data_100GB = {
    "Actions": [
        {
            "BalanceId": f"AU_Data_Domestic__{100 * 1024 * 1024 *
1024}",
            "Units": 100 * 1024 * 1024 * 1024
        }
    ]
}

# Ruim - tamanho redundante no nome descritivo
Action_Data_100GB = {
    "Actions": [
        {
            "BalanceId": f"AU_Data_Domestic_100GB__{100 * 1024 *
1024 * 1024}",  # Redundante!
            "Units": 100 * 1024 * 1024 * 1024
        }
    ]
}

# Ruim - sem informação de tamanho (UI não pode calcular
percentual)
Action_Data_100GB = {
    "Actions": [
        {
            "BalanceId": "AU_Data_Domestic",  # Faltando __tamanho
            "Units": 100 * 1024 * 1024 * 1024
        }
    ]
}
```

Caso Especial - Saldos Monetários:

Saldos monetários geralmente não incluem tamanho original, pois podem ser recarregados para qualquer valor:

```
# Saldo monetário sem codificação de tamanho
{
    "BalanceId": "PAYG_Monetary_Balance",
    "BalanceType": "*monetary",
    "Units": 5000  # $50.00
}

# UI simplesmente mostra saldo atual sem percentual
# "Saldo: $50.00"
```

Estratégia 1: IDs Únicos (Saldos Independentes)

Use IDs únicos (por exemplo, com UUIDs) para criar saldos completamente independentes que nunca interagem.

Exemplo de Implementação:

```
- name: Gerar identificador de saldo único
  set_fact:
    uuid: "{{ 99999999 | random | to_uuid }}"
    balance_id: "Data_5days__5368709120_{{ uuid[0:8] }}"

- name: Adicionar saldo independente com *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
          }
        }]
      }
```

Resultado: Cada addon cria um novo saldo separado, mesmo que o cliente compre o mesmo addon várias vezes.

```yaml
# O cliente compra "addon de dados de 5 dias" três vezes:
Saldo 1:
  ID: "Data_5days__5368709120_a1b2c3d4"
  Valor: 5368709120
  Valor_hr: "5 GB"
  Valor_Original_hr: "5 GB"
  Restante_hr: "5 GB de 5 GB"
  Expiração: 2024-12-29

Saldo 2:
  ID: "Data_5days__5368709120_e5f6g7h8"
  Valor: 5368709120
  Valor_hr: "5 GB"
  Restante_hr: "5 GB de 5 GB"
  Expiração: 2024-12-30

Saldo 3:
  ID: "Data_5days__5368709120_i9j0k1l2"
  Valor: 5368709120
  Valor_hr: "5 GB"
  Restante_hr: "5 GB de 5 GB"
  Expiração: 2024-12-31

# Total disponível: 15GB em três saldos separados
# Cada um mostra individualmente na UI como "Dados 5 dias - 5 GB
de 5 GB"
```

Estratégia 2: IDs Compartilhados com *topup (Rollover)

Use o mesmo ID de Saldo com a ação `*topup` para permitir rollover de saldo e extensão de expiração.

Exemplo de Implementação:

```
- name: Definir ID de saldo fixo
  set_fact:
    balance_id: "Data_5days__5368709120"

- name: Adicionar saldo com *topup (rollover)
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.AddBalance",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Account": "{{ service_uuid }}",
          "BalanceType": "*data",
          "Balance": {
            "ID": "Data_5days__5368709120",
            "Value": 5368709120,
            "ExpiryTime": "+5d",
            "Weight": 10
          }
        }]
      }
```

Resultado: Compras subsequentes adicionam ao saldo existente e estendem a expiração.

```yaml
# Dia 1: O cliente compra "addon de dados de 5 dias":
Saldo:
  ID: "Data_5days__5368709120"
  Valor: 5368709120
  Valor_hr: "5 GB"
  Restante_hr: "5 GB de 5 GB"
  PercentualUsado: 0
  Expiração: 2024-12-29

# Dia 3: O cliente usa 1GB e depois compra o mesmo addon
novamente:
Saldo:
  ID: "Data_5days__5368709120"
  Valor: 9663676416  # 4GB restantes + 5GB novos
  Valor_hr: "9 GB"
  Valor_Original_hr: "5 GB"
  Restante_hr: "9 GB (4 GB rolados + 5 GB novos)"
  PercentualUsado: -80  # Negativo indica rollover
  Expiração: 2024-12-27 (novos +5 dias a partir de hoje)
```

Estratégia 3: IDs Compartilhados com *topup_reset (Valor Fixo)

Use o mesmo ID de Saldo com a ação `*topup_reset` para sempre redefinir para um valor fixo.

Exemplo de Implementação:

```
- name: Definir ID de saldo fixo
  set_fact:
    balance_id: "Monthly_Plan__32212254720"

- name: Adicionar saldo com *topup_reset (sem rollover)
  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
          }]
        }]
      }
```

Resultado: A cada mês, o saldo é redefinido para exatamente 30GB, independentemente de quanto foi usado.

```
# Mês 1, Dia 1:
Saldo:
  ID: "Monthly_Plan__32212254720"
  Valor: 32212254720
  Valor_hr: "30 GB"
  Restante_hr: "30 GB de 30 GB"
  PercentualUsado: 0
  Expiração: 2024-12-31

# Mês 1, Dia 25: O cliente usou 28GB
Saldo:
  ID: "Monthly_Plan__32212254720"
  Valor: 2147483648
  Valor_hr: "2 GB"
  Restante_hr: "2 GB de 30 GB"
  PercentualUsado: 93
  Expiração: 2024-12-31

# Mês 2, Dia 1: O ActionPlan executa *topup_reset
Saldo:
  ID: "Monthly_Plan__32212254720"
  Valor: 32212254720  # Redefinido para o total, os 2GB não
utilizados são perdidos
  Valor_hr: "30 GB"
  Restante_hr: "30 GB de 30 GB"
  PercentualUsado: 0
  Expiração: 2025-01-31
```

Bloqueadores de Saldo

Bloqueadores de Saldo são um recurso poderoso do CGRateS que permitem **bloquear ou limitar o uso** mesmo quando saldos existem. Um saldo bloqueador interrompe o consumo quando chega a zero, impedindo mais uso, independentemente de outros saldos disponíveis.

Como Funcionam os Bloqueadores

Quando um saldo tem `Blocker: true`:

1. **Enquanto o bloqueador tiver valor** → O uso é permitido até o valor do bloqueador
2. **Quando o bloqueador chega a zero** → Todo o uso para, mesmo que outros saldos existam
3. **Erro retornado** → `INSUFFICIENT_CREDIT_BALANCE_BLOCKER` impede a sessão

Características Chave:

- Saldos bloqueadores são **verificados mesmo quando o valor é zero** (diferente dos saldos normais que são ignorados)
- Quando um bloqueador é encontrado com uso restante solicitado, o CGRateS **para o processamento** e retorna um erro
- Bloqueadores funcionam com todos os tipos de saldo: `*voice`, `*data`, `*sms`, `*monetary`

Casos de Uso para Bloqueadores

1. Suspensão de Conta

Bloquear todo o uso quando uma conta é suspensa (por exemplo, falha de pagamento):

```
Action_Suspend_Account = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_suspend-account",
        "Overwrite": True,
        "Tenant": tenant,
        "Actions": [
            # Adicionar bloqueador de valor zero para impedir todo
uso
            {
                "Identifier": "*topup",
                "BalanceId": "Suspension_Blocker",
                "BalanceType": "*monetary",
                "DestinationIDs": "*any",
                "Units": 0,  # Valor zero
                "BalanceWeight": 9999,  # Maior prioridade -
verificado primeiro
                "Blocker": True,  # Bloquear todo uso
                "Weight": 10
            }
        ]
    }]
}

result = OCS_Obj.SendData(Action_Suspend_Account)
```

Resultado: Todas as chamadas/dados/SMS bloqueados, independentemente de outros saldos.

2. Limites de Gastos

Limitar o gasto máximo para evitar surpresas na conta:

```
Action_Monthly_Plan_With_Cap = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_monthly-with-cap",
        "Overwrite": True,
        "Tenant": tenant,
        "Actions": [
            # 10GB de dados incluídos
            {
                "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,  # Consumido primeiro
                "Weight": 95
            },
            # Limite de $50 (bloqueador)
            {
                "Identifier": "*topup_reset",
                "BalanceId": "Overage_Cap",
                "BalanceType": "*monetary",
                "DestinationIDs": "*any",
                "Units": 5000,  # $50.00 máximo de sobrecarga
                "ExpiryTime": "*month",
                "BalanceWeight": 1000,  # Consumido após o
incluído
                "Blocker": True,  # Para quando $50 gastos
                "Weight": 90
            }
        ]
    }]
}
```

Fluxo:

1. O cliente usa 10GB incluídos → GRÁTIS (do Included_Data_10GB)
2. O cliente usa 5GB adicionais → Cobrado do Overage_Cap (taxas PAYG)

3. Quando Overage_Cap chega a $0 → **Todo uso bloqueado** (limite de gastos atingido)

3. Período de Teste Limitado no Tempo

Fornecer uso gratuito limitado para contas de teste:

```
Action_Trial_Account = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_trial-100-minutes",
        "Overwrite": True,
        "Tenant": tenant,
        "Actions": [
            # 100 minutos gratuitos (bloqueador - para quando esgotados)
            {
                "Identifier": "*topup",
                "BalanceId": f"Trial_Voice__{100 * 60 * 1000000000}",
                "BalanceType": "*voice",
                "DestinationIDs": "Dest_Domestic_All",
                "Units": 100 * 60 * 1000000000,  # 100 minutos
                "ExpiryTime": "+720h",  # 30 dias
                "BalanceWeight": 1200,
                "Blocker": True,  # Sem uso após 100 minutos
                "Weight": 10
            }
        ]
    }]
}
```

Resultado: O cliente recebe exatamente 100 minutos gratuitos. Após isso, todas as chamadas são bloqueadas (sem cobranças automáticas).

4. Bloqueio Específico de Destinos

Bloquear destinos específicos enquanto permite outros:

```
Action_Block_Premium_Numbers = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_block-premium",
        "Overwrite": True,
        "Tenant": tenant,
        "Actions": [
            # Uso regular permitido
            {
                "Identifier": "*topup_reset",
                "BalanceId": "Regular_Usage",
                "BalanceType": "*monetary",
                "DestinationIDs": "Dest_Domestic_All",
                "Units": 10000,  # $100
                "ExpiryTime": "*month",
                "BalanceWeight": 1000,
                "Weight": 20
            },
            # Bloquear números premium (0900, etc.)
            {
                "Identifier": "*topup",
                "BalanceId": "Premium_Blocker",
                "BalanceType": "*monetary",
                "DestinationIDs": "Dest_Domestic_Premium",
                "Units": 0,  # Valor zero
                "BalanceWeight": 2000,  # Peso maior - verificado
primeiro para premium
                "Blocker": True,
                "Weight": 10
            }
        ]
    }]
}
```

Fluxo:

- Chamadas domésticas → Usa saldo Regular_Usage
- Chamadas para números premium → Corresponde ao Premium_Blocker (peso 2000 > 1000) → **BLOQUEADO**

Bloqueador vs Saldos Desativados

Não confunda `Blocker` com `Disabled`:

Recurso	Bloqueador	Desativado
Propósito	Parar uso quando saldo esgotado	Pausar temporariamente um saldo
Quando valor > 0	Saldo é utilizável normalmente	Saldo é ignorado
Quando valor = 0	**Bloqueia todo uso posterior**	Saldo é ignorado (próximo saldo tentado)
Caso de Uso	Limites de gastos, tetos, limites de teste	Suspender temporariamente um saldo específico

```
# Bloqueador: Permite 10GB, depois bloqueia tudo
{
    "BalanceId": f"Data_Cap__{10 * 1024 * 1024 * 1024}",
    "Units": 10 * 1024 * 1024 * 1024,
    "Blocker": True  # Para de usar ao atingir 10GB
}

# Desativado: Ignora este saldo completamente (pausado
temporariamente)
{
    "BalanceId": f"Bonus_Data__{5 * 1024 * 1024 * 1024}",
    "Units": 5 * 1024 * 1024 * 1024,
    "Disabled": True  # Este saldo não será usado
}
```

Exemplo Prático: Plano Híbrido com Teto de Segurança

Combine saldos unitários, excesso monetário e um teto bloqueador:

```
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 minutos domésticos incluídos
            {
                "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 excesso
            {
                "Identifier": "*topup_reset",
                "BalanceId": "Overage_Allowance",
                "BalanceType": "*monetary",
                "DestinationIDs": "*any",
                "Units": 2000,  # $20.00
                "ExpiryTime": "*month",
                "BalanceWeight": 1000,
                "Weight": 90
            },
            # $50 LIMITE RÍGIDO (bloqueador)
            {
                "Identifier": "*topup_reset",
                "BalanceId": "Hard_Spending_Cap",
                "BalanceType": "*monetary",
                "DestinationIDs": "*any",
                "Units": 5000,  # $50.00 máximo absoluto
                "ExpiryTime": "*month",
```

```
                "BalanceWeight": 500,  # Menor que o excesso - usado por último
                "Blocker": True,  # PARA quando o teto for atingido
                "Weight": 85
            }
        ]
    }]
}
```

Jornada do Cliente:

1. **0-500 minutos:** Usa Domestic_Voice_500min (GRÁTIS)
2. **500-700 minutos:** Usa Overage_Allowance a $0.10/min = $20 (200 min)
3. **700-1200 minutos:** Usa Hard_Spending_Cap a $0.10/min = $50 (500 min)
4. **Aos 1200 minutos:** Hard_Spending_Cap esgotado → **TODO USO BLOQUEADO**

O cliente recebe 1200 minutos no total, com um teto de gastos máximo de $50.

Melhores Práticas com Bloqueadores

1. **Use Peso Alto para Bloqueadores**

   ```
   "BalanceWeight": 9999  # Garantir que o bloqueador seja verificado primeiro
   ```

2. **Bloqueadores de Valor Zero para Bloqueio Imediato**

   ```
   "Units": 0,  # Bloquear imediatamente
   "Blocker": True
   ```

3. **Notifique Clientes Antes da Exaustão do Bloqueador**

   - Use ActionTriggers para enviar notificações em 80%, 90%, 100% do uso do bloqueador

- Dê aos clientes a opção de aumentar o teto antes que o bloqueio ocorra

4. **Remova Bloqueadores Ao Desuspender**

```
# Use *remove_balance para deletar o bloqueador
{
    "Identifier": "*remove_balance",
    "BalanceId": "Suspension_Blocker"
}
```

5. **Teste o Comportamento do Bloqueador**

   - Verifique se o bloqueador retorna o erro `INSUFFICIENT_CREDIT_BALANCE_BLOCKER`
   - Confirme que os CDRs mostram custo = -1.0 quando bloqueados
   - Teste se outros saldos NÃO são usados após a exaustão do bloqueador

Resolução de Problemas com Bloqueadores

Problema: Uso não bloqueado apesar do bloqueador em zero

Causas Possíveis:

1. Peso do bloqueador muito baixo (outros saldos verificados primeiro)
2. DestinationIDs não correspondem ao destino de uso
3. Campo Blocker não definido como `True`

Solução:

```
# Verifique a configuração do bloqueador
OCS_Obj.SendData({
    'method': 'ApierV2.GetAccount',
    'params': [{"Tenant": tenant, "Account": "service_uuid"}]
})

# Verifique:
# - Blocker: true
# - BalanceWeight é o mais alto (por exemplo, 9999)
# - DestinationIDs inclui o destino de uso
# - Valor = 0
```

Problema: Uso bloqueado inesperadamente

Causa Possível: Saldo bloqueador criado involuntariamente com valor baixo

Solução: Verifique todos os saldos para `Blocker: true` e verifique se seus valores são apropriados para seu caso de uso.

Regras de Consumo de Saldo

Regra 1: Prioridade de Precisão de Destino

Saldos com **maior precisão de destino** (correspondência de destino mais específica) são consumidos primeiro. Isso é determinado pelo comprimento da correspondência de prefixo.

```
# O cliente chama +44-20-1234-5678 (Londres, Reino Unido)

Saldo 1:
  DestinationIDs: "Dest_UK_London"  # Prefixo: "4420" (precisão:
4)
  Valor: 100 minutos
  Peso: 10

Saldo 2:
  DestinationIDs: "Dest_UK_All"     # Prefixo: "44" (precisão: 2)
  Valor: 200 minutos
  Peso: 10

# O Saldo 1 é consumido primeiro (precisão 4 > precisão 2)
# Correspondência de destino mais específica vence
```

Caso de Uso: Saldos específicos de cidade ou região têm prioridade sobre saldos de cobertura nacional.

Regra 2: Prioridade de Peso (Mesma Precisão)

Quando a precisão de destino é igual, saldos com **peso maior** são consumidos primeiro.

```
Saldo 1:
  ID: "Premium_Data"
  Valor: 5GB
  Peso: 20

Saldo 2:
  ID: "Standard_Data"
  Valor: 10GB
  Peso: 10

# O Saldo 1 é consumido primeiro (peso 20 > peso 10)
# Mesmo que o Saldo 2 tenha mais dados
```

Caso de Uso: Saldos prioritários (dados bônus consumidos antes de dados regulares).

Regra 3: Primeiro o Mais Antigo

Quando o peso coincide, o saldo mais antigo é usado primeiro.

```
Saldo 1:
  ID: "Data_Package_A"
  Valor: 5GB
  Expiração: 2024-12-25
  Peso: 10

Saldo 2:
  ID: "Data_Package_B"
  Valor: 10GB
  Expiração: 2025-01-15
  Peso: 10
```

Para controlar a ordem de consumo, use pesos diferentes:

```
# Abordagem correta: Use peso para priorizar saldos prestes a
expirar
Saldo 1:
  ID: "Data_Package_A"
  Valor: 5GB
  Expiração: 2024-12-25
  Peso: 11  # Peso maior = consumido primeiro

Saldo 2:
  ID: "Data_Package_B"
  Valor: 10GB
  Expiração: 2025-01-15
  Peso: 10  # Peso menor = consumido segundo

# O Saldo 1 é consumido primeiro (peso 11 > peso 10)
```

Melhor Prática: Se você deseja garantir que saldos prestes a expirar sejam consumidos primeiro, atribua a eles pesos mais altos ao criá-los.

Exemplos Práticos

Exemplo 1: Addon Simples de Dados (Independente)

Cenário: O cliente pode comprar o addon "5GB 5 dias" várias vezes, cada uma criando um saldo separado.

Implementação:

```
- name: Gerar UUID para saldo único
  set_fact:
    uuid: "{{ 99999999 | random | to_uuid }}"

- name: Adicionar saldo independente 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 dias
            "Weight": 10
          }
        }]
      }
```

Experiência do Cliente:

- Compra o addon em 24 de dezembro → Recebe 5GB com expiração em 29 de dezembro

- Compra o addon em 25 de dezembro → Recebe 5GB com expiração em 30 de dezembro
- Ambos os saldos coexistem; a ordem de consumo depende dos pesos dos saldos (ambos têm peso 10, então a ordem não é garantida)

Exemplo 2: Pacote de Dados Rollover

Cenário: "Plano Mensal de 50GB" onde dados não utilizados rolam quando o cliente recarrega cedo.

Implementação:

```
- name: Adicionar saldo de dados rollover
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.AddBalance",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Account": "{{ service_uuid }}",
          "BalanceType": "*data",
          "Balance": {
            "ID": "Rollover_Monthly_50GB",
            "Value": 53687091200,  # 50GB
            "ExpiryTime": "+720h",  # 30 dias
            "Weight": 10
          }
        }]
      }
```

Tipo de Ação: Usa o comportamento padrão `*topup` (rollover habilitado)

Experiência do Cliente:

- Dia 1: Recebe 50GB com expiração em 30 dias
- Dia 20: Usou 30GB, tem 20GB restantes

- Dia 20: Recarrega novamente → Recebe 70GB no total (20GB + 50GB), expiração se estende para +30 dias a partir do Dia 20

Exemplo 3: Plano Mensal Fixo (Sem Rollover)

Cenário: Plano "Mensal Ilimitado de 100GB" que é redefinido para exatamente 100GB a cada mês, sem rollover.

Implementação:

```
- name: Criar ação de redefinição mensal
  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: Criar ActionPlan mensal
  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: Atribuir ActionPlan à conta
```

```
  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
        }]
      }
```

Experiência do Cliente:

- Mês 1: Recebe 100GB, usa 95GB, tem 5GB restantes
- Mês 2: Saldo é redefinido para 100GB (5GB não utilizados são perdidos)
- Mês 2: Usa 20GB, tem 80GB restantes
- Mês 3: Saldo é redefinido para 100GB (80GB não utilizados são perdidos)

Exemplo 4: Saldos Multi-Tier com Prioridade de Peso

Cenário: O cliente tem "Dados Bônus" (alta prioridade) e "Dados Regulares" (baixa prioridade). Dados bônus consumidos primeiro.

Implementação:

```
# Adicionar dados bônus com peso alto
- name: Adicionar saldo de dados bônus
  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 dias
            "Weight": 20  # Maior prioridade
          }
        }]
      }

# Adicionar dados regulares com peso normal
- name: Adicionar saldo de dados regulares
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.AddBalance",
        "params": [{
          "Account": "{{ service_uuid }}",
          "BalanceType": "*data",
          "Balance": {
            "ID": "Regular_Data",
            "Value": 53687091200,  # 50GB
            "ExpiryTime": "+720h",  # 30 dias
            "Weight": 10  # Prioridade normal
          }
        }]
      }
```

Experiência do Cliente:

- Tem 5GB de dados bônus (peso 20) + 50GB de dados regulares (peso 10)
- Consome todos os 5GB de dados bônus primeiro
- Depois consome do pool de 50GB de dados regulares
- Dados regulares preservados por mais tempo

Ações Comuns do CGRateS

O CGRateS suporta múltiplos identificadores de ação para diferentes operações:

Manipulação de Saldo

***topup** - Adicionar ao saldo existente (rollover)
***topup_reset** - Redefinir saldo para novo valor (sem rollover)
***debit** - Subtrair do saldo
***debit_reset** - Definir saldo para valor negativo
***reset_account** - Remover todos os saldos

Projetando Produtos Adicionais

Ao projetar produtos adicionais no OmniCRM, considere estas perguntas:

Pergunta 1: Os saldos devem empilhar?

Sim (Independentes) → Use IDs de Saldo únicos (com UUID)
Não (Substituir) → Use ID de Saldo fixo com `*topup_reset`
Sim (Rollover) → Use ID de Saldo fixo com `*topup`

Pergunta 2: O que acontece com o saldo não utilizado?

Rollover → Use ação `*topup`
Perdido → Use ação `*topup_reset`

Pools separados → Use IDs de Saldo únicos

Pergunta 3: Como os saldos devem ser consumidos?

Mais antigo primeiro → Use pesos diferentes (atribua peso maior a saldos mais antigos/prestes a expirar)
Premium primeiro → Pesos diferentes (peso maior = prioridade maior)
Ordem específica → Use pesos: 30 (premium), 20 (bônus), 10 (regular)
Aleatório/Sem preferência → Use mesmo peso (ordem de consumo não garantida)

Pergunta 4: Qual é a estratégia de expiração?

Duração fixa → Use expiração relativa (`+720h` para 30 dias)
Final do mês → Use `*monthly` no ActionPlan
Nunca expira → Use `*unlimited` ou duração muito longa

Estrutura da Ação do CGRateS em Playbooks

Aqui está a estrutura completa para criar uma Ação:

```
- name: Criar Ação CGRateS
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "ApierV1.SetActions",
        "params": [{
          "ActionsId": "Action_Name_Here",
          "Overwrite": true,  # Substituir se existir
          "Actions": [
            {
              "Identifier": "*topup",  # ou *topup_reset
              "BalanceType": "*data",  # *data, *voice, *sms,
*monetary
              "Units": 5368709120,  # Quantidade a adicionar
              "ExpiryTime": "+120h",  # +Xh, *unlimited, *monthly
              "DestinationIds": "*any",  # Geralmente *any
              "BalanceId": "Balance_Name",  # Único ou
compartilhado
              "Weight": 10,  # Prioridade (maior = consumido
primeiro)
              "Blocker": false,  # Se true, impede saldo negativo
              "Disabled": false,  # Se true, saldo está desativado
              "SharedGroups": ""  # Grupo de saldo compartilhado
(opcional)
            },
            {
              "Identifier": "*cdrlog",  # Registrar esta ação como
CDR
              "BalanceType": "*generic",
              "ExtraParameters": "
{\"Category\":\"^activation\",\"Destination\":\"Addon Name\"}"
            }
          ]
        }]
      }
```

Descrições dos Campos

ActionsId - Identificador único para este conjunto de ações

Identifier - O tipo de operação (`*topup`, `*topup_reset`, `*cdrlog`, etc.)

BalanceType - Tipo de saldo:

- `*data` - Saldos de dados (bytes)
- `*voice` - Saldos de voz (segundos)
- `*sms` - Saldos de SMS (contagem)
- `*monetary` - Saldos monetários (unidades de moeda)
- `*generic` - Saldos genéricos

Units - Quantidade a adicionar/deduzir (em unidades base: bytes para dados, segundos para voz)

ExpiryTime - Quando o saldo expira:

- `+Xh` - Relativo (por exemplo, `+720h` = 30 dias)
- `*unlimited` - Nunca expira
- `*monthly` - Final do m��s
- `2024-12-31T23:59:59Z` - Timestamp absoluto

BalanceId - Identificador para este saldo (ID compartilhado = interage, ID único = independente)

Weight - Prioridade (número maior = maior prioridade, consumido primeiro)

Blocker - Se true, impede a conta de ficar negativa

Disabled - Se true, saldo existe mas não pode ser usado

Definindo Ações via Python (Configuração Inicial)

As Ações são tipicamente definidas durante a configuração inicial do sistema usando scripts Python com a biblioteca `cgrateshttpapi`. Estes exemplos mostram como definir Ações usando `OCS_Obj.SendData()`.

Pré-requisitos

```
import cgrateshttpapi
import time

OCS_Obj = cgrateshttpapi.CGRateS("ocs.example.com", "2080")
tenant = "your_tenant_name"
tpid = str(tenant) + "_" + str(int(time.time()))
```

Definindo Destinos

Antes de criar Ações, você deve definir destinos que especificam ONDE os saldos podem ser usados.

Destinos vêm em dois tipos:

- **Destinos Geográficos** - Prefixos de números para voz/SMS PARA lugares (por exemplo, `Dest_International_UK`)
- **Destinos PLMN** - Códigos de rede para dados DE lugares (por exemplo, `Dest_PLMN_OnNet`, `Dest_PLMN_US_Verizon`)

Regra Crítica:

- **Saldos de voz/SMS** → Use destinos geográficos (o número que está sendo chamado PARA)
- **Saldos de dados** → Use destinos PLMN (a rede à qual o cliente está conectado DE)

Para configuração completa de destinos incluindo:

- Destinos geográficos (domésticos, internacionais, gratuitos, premium)
- Destinos PLMN (redes on-net, redes de roaming, zonas)
- Regras de formato PLMN e melhores práticas
- Resolução de problemas de questões de destino

Veja: **Configuração de Destinos do CGRateS**

Cálculos de Unidades

Entender conversões de unidades é crítico para definir saldos corretamente:

```
# Saldos de dados (em bytes)
1_GB = 1 * 1024 * 1024 * 1024  # 1073741824 bytes
100_GB = 100 * 1024 * 1024 * 1024

# Saldos de voz (em nanossegundos)
1_minuto = 60 * 1000000000  # 60 bilhões de nanossegundos
3000_minutos = 3000 * 60 * 1000000000

# Saldos de SMS (em contagem)
3000_sms = 3000
```

Exemplo 1: Plano Mensal Multi-Saldo (Python)

Um plano mensal abrangente com saldos de dados, voz, SMS e roaming que se redefinem a cada mês.

```
Action_AU_Premium_Plan_1 = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_au-premium-plan-1",
        "Overwrite": True,
        "Tenant": str(tenant),
        "Actions": [
            # Primeiro, redefina a conta para limpar saldos
antigos
            {
                "Identifier": "*reset_account",
                "Weight": 700
            },
            # Adicione saldo de dados de 100GB
            # IMPORTANTE: Saldos de dados usam destinos PLMN (rede
à qual o cliente está conectado)
            # NÃO destinos geográficos. Use SEU on-net PLMN.
            {
                "Identifier": "*topup_reset",
                "BalanceId": "AU_Data_Domestic__" + str(100 * 1024
* 1024 * 1024),
                "BalanceType": "*data",
                "DestinationIDs": "Dest_PLMN_OnNet",  # Seu on-net
PLMN (mcc505.mnc057)
                "Units": 100 * 1024 * 1024 * 1024,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 90
            },
            # Adicione saldo de voz de 3000 minutos
            {
                "Identifier": "*topup_reset",
                "BalanceId": "AU_Voice_Domestic__" + str(3000 * 60
* 1000000000),
                "BalanceType": "*voice",
                "DestinationIDs":
"Dest_AU_Mobile;Dest_AU_Fixed;Dest_AU_TollFree;",
                "Units": 3000 * 60 * 1000000000,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 89
            },
```

```
            # Adicione saldo de SMS de 3000
            {
                "Identifier": "*topup_reset",
                "BalanceId": "AU_SMS_Domestic__" + str(3000),
                "BalanceType": "*sms",
                "DestinationIDs": "Dest_AU_Mobile;",
                "Units": 3000,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 88
            },
            # Adicione 6GB de dados de roaming
            {
                "Identifier": "*topup_reset",
                "BalanceId": "AU_Roaming_Data__" + str(6 * 1024 *
1024 * 1024),
                "BalanceType": "*data",
                "DestinationIDs": "Dest_Roaming_All",
                "Units": 6 * 1024 * 1024 * 1024,
                "ExpiryTime": "*month",
                "BalanceWeight": 1100,
                "Weight": 87
            },
            # Registre esta ação como um CDR
            {
                "Identifier": "*cdrlog",
                "BalanceType": "*generic",
                "ExtraParameters": "
{\"Category\":\"^activation\",\"Destination\":\"Addon Name\"}"
            }
        ]
    }]
}

# Envie a definição da ação para o CGRateS
result = OCS_Obj.SendData(Action_AU_Premium_Plan_1)
assert result['error'] is None or result['error'] == "EXISTS"
print("Criada Ação: Action_au-premium-plan-1")
```

Pontos Chave:

- Usa `*reset_account` para limpar saldos antigos primeiro

- Usa `*topup_reset` para alocações mensais fixas (sem rollover)
- BalanceWeight determina a ordem de consumo (doméstico 1200 > roaming 1100)
- Peso determina a ordem de execução dentro da Ação
- Inclui `*cdrlog` para rastreamento de ativações

Exemplo 2: Addon Simples de Dados (Python)

Um simples addon de 20GB de dados com rollover desativado.

```
Action_AU_Data_Addon_20GB = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_au-data-addon-20gb",
        "Overwrite": True,
        "Tenant": str(tenant),
        "Actions": [
            # Redefina a conta primeiro
            {
                "Identifier": "*reset_account",
                "Weight": 700
            },
            # Adicione 20GB de dados
            # Saldos de dados usam destinos PLMN (rede à qual o
cliente está conectado)
            {
                "Identifier": "*topup_reset",
                "BalanceId": "AU_Data_Domestic__" + str(20 * 1024
* 1024 * 1024),
                "BalanceType": "*data",
                "DestinationIDs": "Dest_PLMN_OnNet",  # Seu on-net
PLMN (mcc505.mnc057)
                "Units": 20 * 1024 * 1024 * 1024,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 90
            }
        ]
    }]
}

result = OCS_Obj.SendData(Action_AU_Data_Addon_20GB)
assert result['error'] is None or result['error'] == "EXISTS"
print("Criada Ação: Action_au-data-addon-20gb")
```

Exemplo 3: Addon de Voz Internacional (Python)

Um addon para minutos de chamadas internacionais.

```
Action_AU_International_Voice_100min = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_au-international-voice-100min",
        "Overwrite": True,
        "Tenant": str(tenant),
        "Actions": [
            {
                "Identifier": "*reset_account",
                "Weight": 700
            },
            # Adicione 100 minutos para chamadas internacionais
            {
                "Identifier": "*topup_reset",
                "BalanceId": "AU_Voice_International__" + str(100
* 60 * 1000000000),
                "BalanceType": "*voice",
                "DestinationIDs": "Dest_International_All",
                "Units": 100 * 60 * 1000000000,
                "ExpiryTime": "*month",
                "BalanceWeight": 1000,
                "Weight": 90
            }
        ]
    }]
}

result = OCS_Obj.SendData(Action_AU_International_Voice_100min)
assert result['error'] is None or result['error'] == "EXISTS"
print("Criada Ação: Action_au-international-voice-100min")
```

Nota: Saldos de voz/SMS usam destinos geográficos (número sendo chamado), enquanto dados usam destinos PLMN (rede à qual o cliente está conectado). Veja Definindo Produtos para configuração de destinos.

Referência de Campos de Ação em Python

Campos de Definição de Ação:

- **ActionsId** (obrigatório) - Identificador único para este conjunto de ações (deve corresponder à convenção product_slug no CRM)
- **Overwrite** - Se True, substitui a ação existente com o mesmo ID
- **Tenant** - Nome do inquilino do CGRateS
- **Actions** - Array de ações individuais a serem executadas

Campos de Ação Individual:

- **Identifier** - Tipo de ação (`*topup`, `*topup_reset`, `*cdrlog`, etc.)
- **BalanceId** - Identificador único para este saldo (deve corresponder entre recargas para que o rollover funcione)
- **BalanceType** - Tipo de saldo (`*data`, `*voice`, `*sms`, `*monetary`)
- **DestinationIDs** - Controla ONDE o saldo pode ser usado:
  - Para **voz/SMS**: Use destinos geográficos (por exemplo, `"Dest_AU_Mobile"`, `"Dest_International_UK"`)
  - Para **dados**: Use destinos PLMN (por exemplo, `"Dest_PLMN_OnNet"`, `"Dest_PLMN_US_Verizon"`)
- **Units** - Quantidade a adicionar (bytes para dados, nanossegundos para voz)
- **ExpiryTime** - Quando o saldo expira:
  - `+Xh` - Relativo (por exemplo, `+720h` = 30 dias)
  - `*unlimited` - Nunca expira
  - `*monthly` - Final do mês
  - `2024-12-31T23:59:59Z` - Timestamp absoluto
- **BalanceWeight** - Prioridade de consumo (número maior = maior prioridade, consumido primeiro)
- **Weight** - Ordem de execução dentro do conjunto de ações (número maior = executa primeiro)
- **Blocker** (opcional) - Se `True`, impede a conta de ficar negativa

- **Disabled** (opcional) - Se `True`, este saldo é ignorado/pulado durante o consumo

Executando Ações

Uma vez que uma Ação é criada, execute-a em uma conta:

```
- name: Executar Ação na Conta
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body_format: json
    body:
      {
        "method": "APIerSv1.ExecuteAction",
        "params": [{
          "Tenant": "{{ crm_config.ocs.ocsTenant }}",
          "Account": "{{ service_uuid }}",
          "ActionsId": "Action_Name_Here"
        }]
      }
```

Isso executa todas as operações definidas na Ação na conta especificada.

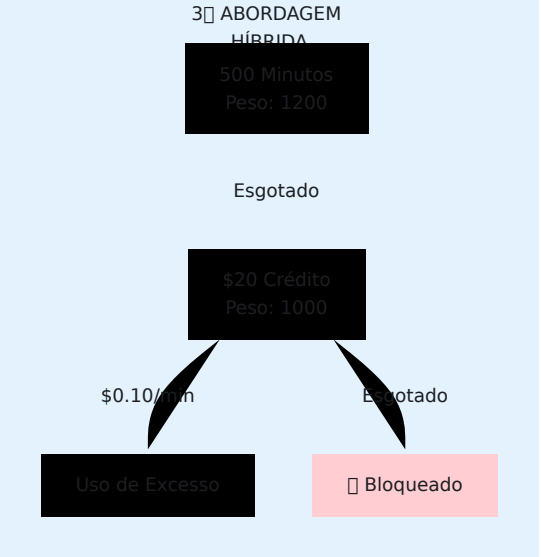
Abordagens de Gestão de Saldo

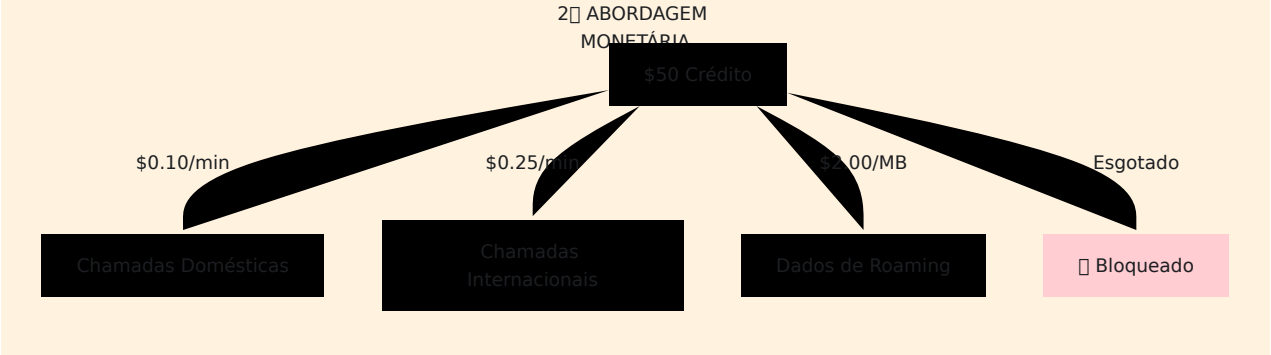
Existem três abordagens principais para gerenciar saldos no CGRateS, cada uma com diferentes trade-offs para a experiência do cliente, previsibilidade de faturamento e complexidade operacional.

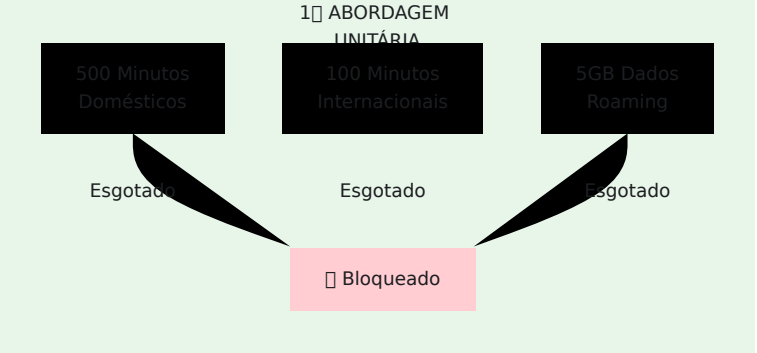
Comparação: Unitário vs Monetário vs Híbrido

Esta tabela resume os trade-offs entre as três abordagens de saldo:

Recurso	Unitário	Monetário (PAYG)	Híbrido
Previsibilidade do Cliente	□ Custo mensal fixo	□ Custos variáveis	⚠ Principalmente previsível
Flexibilidade de Destino	□ Limitado a destinos incluídos	□ Chame/use em qualquer lugar	□ Incluído + em qualquer lugar
Tratamento de Excesso	□ Corte rígido	□ Uso automático	□ Transbordo automático
Risco de Surpresa na Conta	□ Baixo (teto rígido)	□ Alto (faturamento ilimitado)	⚠ Moderado (transbordo com teto)
Complexidade de Configuração	⚠ Moderada (muitos saldos)	□ Simples (um saldo)	□ Complexo (ambos)
Otimização de Receita	⚠ Menor ARPU	□ Maior ARPU de usuários pesados	□ ARPU equilibrado
Satisfação do Cliente	□ Alta (sem surpresas)	□ Baixa (surpresa na conta)	□ Alta (o melhor dos dois)
Melhor Para	Usuários previsíveis	Usuários ocasionais	A maioria dos clientes







Abordagem 1: Saldos Unitários

Conceito: Fornecer quantidades específicas para destinos específicos. Cada saldo tem um valor fixo (minutos, GB, contagem de SMS) vinculado a destinos específicos. Quando esgotado, o uso é bloqueado, a menos que haja um fallback monetário.

Exemplo: Plano Doméstico com Múltiplos Saldos

```
Action_Domestic_Plan = {
    "id": "0",
    "method": "ApierV1.SetActions",
    "params": [{
        "ActionsId": "Action_domestic-plan",
        "Overwrite": True,
        "Tenant": tenant,
        "Actions": [
            {
                "Identifier": "*reset_account",
                "Weight": 700
            },
            # 500 minutos para chamadas domésticas
            {
                "Identifier": "*topup_reset",
                "BalanceId": f"Domestic_Voice__{500 * 60 *
1000000000}",
                "BalanceType": "*voice",
                "DestinationIDs": "Dest_Domestic_All",  # SOMENTE
doméstico
                "Units": 500 * 60 * 1000000000,  # 500 minutos em
nanossegundos
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 90
            },
            # 1000 SMS domésticos
            {
                "Identifier": "*topup_reset",
                "BalanceId": "Domestic_SMS__1000",
                "BalanceType": "*sms",
                "DestinationIDs": "Dest_Domestic_All",
                "Units": 1000,
                "ExpiryTime": "*month",
                "BalanceWeight": 1200,
                "Weight": 89
            },
            # 10GB de dados domésticos
            {
                "Identifier": "*topup_reset",
                "BalanceId": f"Domestic_Data__{10 * 1024 * 1024 *
1024}",
                "BalanceType": "*data",
```

```
                "DestinationIDs": "Dest_PLMN_OnNet",  # Seu on-net
PLMN
                "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"
```

Como Isso Funciona:

- Chamadas domésticas (1-555-1234) → Usa saldo de 500 minutos
- Chamadas internacionais (44-20-xxx) → NÃO saldo disponível, bloqueado OU usa saldo monetário se disponível
- SMS doméstico → Usa saldo de 1000 SMS
- Uso de dados em casa → Usa saldo de 10GB
- Uso de dados em roaming → NÃO saldo disponível (precisaria de saldo de dados de roaming separado)

Prós:

- Custos previsíveis para os clientes
- Sem surpresas na conta
- Limites claros

Contras:

- Menos flexível - não pode usar serviço fora dos destinos incluídos

- Requer múltiplos saldos para diferentes casos de uso
- O cliente pode se sentir restrito

Abordagem 2: Monetário (PAYG)

Conceito: Fornecer crédito monetário cobrado a taxas específicas de destino. Um único saldo monetário usado para TODOS os tipos de uso. O CGRateS procura a taxa para cada destino e deduz o custo do crédito.

Nota: PAYG requer a definição de Perfis de Taxa para cada destino para definir o valor em dólar por unidade. Veja Perfis de Taxa para Saldos PAYG/Monetários para configuração completa.

**Ex

Configuração de Destinos do CGRateS

Este guia explica como configurar destinos no CGRateS para o OmniCRM. Os destinos definem ONDE os saldos podem ser usados - seja chamando números específicos (geográficos) ou usando dados/vóz em redes específicas (PLMN).

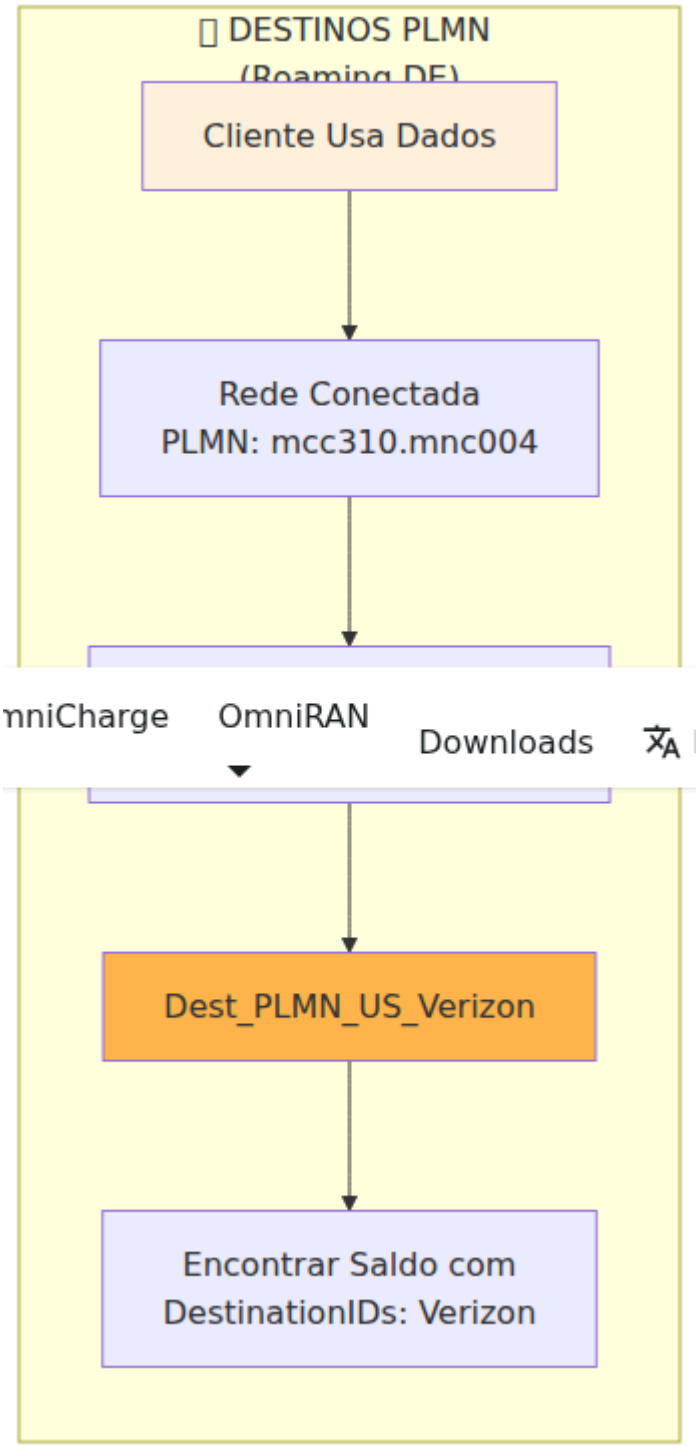
Visão Geral

O CGRateS usa **destinos** para controlar onde os clientes podem usar seus saldos:

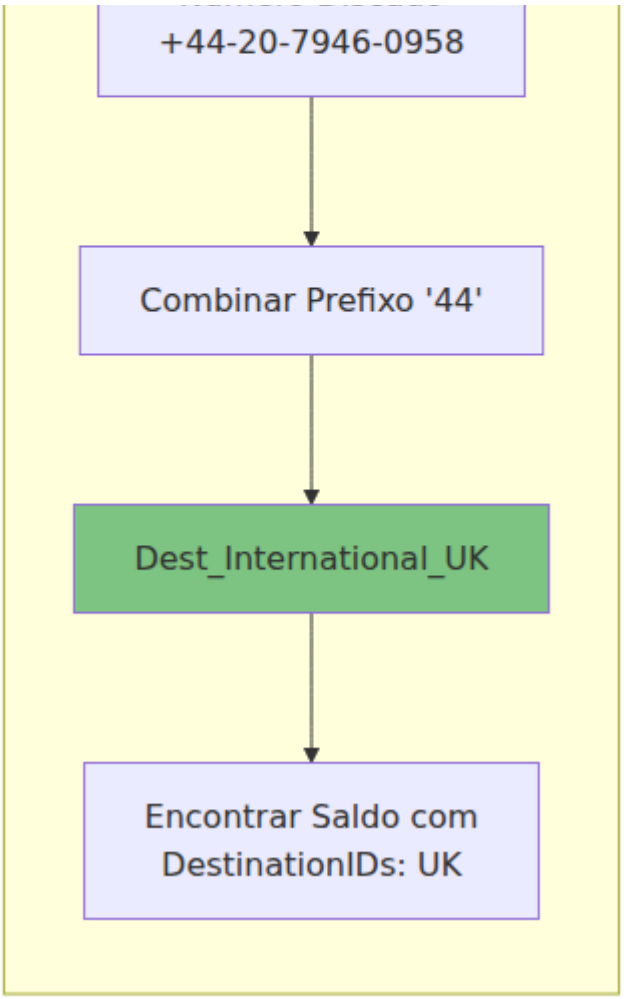
- **Destinos Geográficos** - Prefixos de números para chamadas/SMS PARA locais específicos (por exemplo, números do Reino Unido, números gratuitos dos EUA)
- **Destinos PLMN** - Códigos de rede para uso de dados e roaming DE redes específicas (por exemplo, Verizon, Vodafone UK)

Quando os clientes usam serviços, o CGRateS combina:

- **Voz/SMS**: O prefixo do número discado → Destino geográfico
- **Dados**: O código PLMN da rede → Destino PLMN



nniCharge OmniRAN Downloads



Regra Crítica

- **Saldos de Voz/SMS** → Use destinos geográficos (o número sendo chamado PARA)
- **Saldos de Dados** → Use destinos PLMN (a rede à qual o cliente está conectado DE)

Pré-requisitos

```
import cgrateshttpapi
import time

OCS_Obj = cgrateshttpapi.CGRateS("ocs.example.com", "2080")
tenant = "seu_nome_de_tenant"
tpid = str(tenant) + "_" + str(int(time.time()))
```

Parte 1: Destinos Geográficos (Chamadas PARA Locais)

Os destinos geográficos definem prefixos de números para chamadas de voz e SMS para locais específicos.

Destinos Domésticos

```
#
# ============================================================================
# DESTINOS DOMÉSTICOS (EUA/Canadá - NANP)
#
# ============================================================================

# Móveis e Fixos dos EUA/Canadá (compartilham códigos de área)
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_Domestic_All",
        "Prefixes": ["1"]  # NANP (EUA/Canadá)
    }]
})

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

# Tarifas Premium dos EUA
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_Domestic_Premium",
        "Prefixes": ["1900"]
    }]
})
```

Destinos Internacionais

```
#
# ============================================================================
# DESTINOS INTERNACIONAIS (Por País)
#
# ============================================================================

# Reino Unido
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_International_UK",
        "Prefixes": ["44"]
    }]
})

# China
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_International_China",
        "Prefixes": ["86"]
    }]
})

# Austrália - Móvel
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_AU_Mobile",
        "Prefixes": ["614"]  # Móveis australianos (04xx discados
como 614xx)
    }]
})

# Austrália - Linha Fixa
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)
        ]
    }]
})

# Austrália - Números Gratuitos
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_AU_TollFree",
        "Prefixes": [
            "611800",  # Números gratuitos
            "611300"   # Tarifa local
        ]
    }]
})

# Austrália - Tarifas Premium
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_AU_Premium",
        "Prefixes": ["6119"]  # Serviços de tarifas premium
    }]
})

# Europa (agrupados)
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_International_Europe",
        "Prefixes": [
            "33",  # França
            "49",  # Alemanha
            "39",  # Itália
```

```
            "34",  # Espanha
            "31",  # Países Baixos
            "32",  # Bélgica
            "41"   # Suíça
        ]
    }]
})

# Internacional - Todos (catch-all, exclui doméstico)
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_International_All",
        "Prefixes": ["2", "3", "4", "5", "6", "7", "8", "9"]  # Todos exceto "1" (NANP)
    }]
})
```

Casos de Uso:

- **Planos domésticos**: "Chamadas ilimitadas para números dos EUA/Canadá" → `Dest_Domestic_All`
- **Adicional de chamadas para o Reino Unido**: "100 minutos para o Reino Unido" → `Dest_International_UK`
- **Plano australiano**: "3000 minutos para móveis e linhas fixas australianas" → `Dest_AU_Mobile`, `Dest_AU_Fixed`, `Dest_AU_TollFree`
- **Pacote internacional**: "50 minutos para qualquer número internacional" → `Dest_International_All`

Parte 2: Destinos PLMN (Uso DE Locais - Roaming / Uso de Dados)

Os destinos PLMN definem códigos de rede para uso de dados e cenários de roaming. Use o formato `mccXXX.mncYYY` onde:

- **MCC** = Código do País Móvel (3 dígitos)
- **MNC** = Código da Rede Móvel (2-3 dígitos)

Regras de Formato PLMN

- **Rede específica**: `"mcc310.mnc004"` → Apenas rede Verizon 004
- **Todas as redes no país**: `"mcc310"` → Qualquer rede dos EUA
- **Exemplos**: `mcc310.mnc004`, `mcc234.mnc015`, `mcc505.mnc057`

On-Net (Sua Rede Doméstica)

IMPORTANTE: Defina SUA rede específica onde SEUS SIMs estão provisionados.

```
# On-Net (Sua Rede Doméstica)
# Este é o PLMN do SEU operador - a rede onde os SIMs dos SEUS clientes estão ativos
# Todos os saldos de dados domésticos on-net devem usar este destino

OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_OnNet",
        "Prefixes": ["mcc505.mnc057"]  # Exemplo: MCC 505, MNC 57 se esta for sua rede doméstica
    }]
})
# Substitua mcc505.mnc057 pelo código PLMN real do SEU operador
# Todos os saldos de dados domésticos on-net devem usar os prefixos definidos aqui
# Caso contrário, serão considerados roaming
```

Destinos PLMN de Roaming nos EUA

```
#
# ============================================================================
# DESTINOS PLMN DE ROAMING NOS EUA
#
# ============================================================================

# 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"
        ]
    }]
})

# Todos os PLMNs dos EUA (catch-all)
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_US_All",
        "Prefixes": ["mcc310", "mcc311", "mcc312", "mcc313",
"mcc316"]
    }]
})
# Nota: Isso combina QUALQUER rede com esses MCCs (310, 311, etc.)
# Cliente na mcc310.mnc004 (Verizon), mcc310.mnc410 (AT&T),
mcc311.mnc580 (US Cellular)
# todos combinariam com este destino
```

Destinos PLMN de Roaming no Reino Unido

```
#
# ============================================================================
# DESTINOS PLMN DE ROAMING NO REINO UNIDO
#
# ============================================================================

# 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"]
    }]
})

# Todos os PLMNs do Reino Unido
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_UK_All",
        "Prefixes": ["mcc234"]
    }]
})
```

Zonas de Roaming (Grupos de Múltiplos Países)

Ótimo para pacotes de roaming regionais como "Roaming na Europa" ou "Roaming na Ásia-Pacífico".

```
#
================================================================
# ZONAS DE ROAMING (Grupos de Múltiplos Países)
#
================================================================

# Zona 1: América do Norte
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_Zone_NorthAmerica",
        "Prefixes": [
            "mcc310", "mcc311", "mcc312", "mcc313", "mcc316",  #
EUA
            "mcc302", "mcc334"  # Canadá
        ]
    }]
})
# Usar prefixos apenas com MCC fornece cobertura em QUALQUER rede
nesses países
# Ótimo para pacotes de "roaming na América do Norte" que funcionam
em todas as operadoras

# Zona 2: Europa
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_Zone_Europe",
        "Prefixes": [
            "mcc234",  # Reino Unido
            "mcc208",  # França
            "mcc262",  # Alemanha
            "mcc222",  # Itália
            "mcc214",  # Espanha
            "mcc228"   # Suíça
        ]
    }]
})

# Zona 3: Ásia-Pacífico
OCS_Obj.SendData({
```

```
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_Zone_AsiaPacific",
        "Prefixes": [
            "mcc505",  # Austrália
            "mcc460",  # China
            "mcc454",  # Hong Kong
            "mcc440",  # Japão
            "mcc520",  # Tailândia
            "mcc525",  # Cingapura
            "mcc530"   # Nova Zelândia
        ]
    }]
})

# Catch-all para qualquer roaming (usar com cautela)
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_Roaming_All",
        "Prefixes": ["mcc"]  # Combina QUALQUER PLMN (muito amplo)
    }]
})
```

Casos de Uso:

- **Roaming específico de rede**: "5GB apenas na Verizon" → `Dest_PLMN_US_Verizon`
- **Roaming em todo o país**: "10GB nos EUA (qualquer rede)" → `Dest_PLMN_US_All`
- **Roaming regional**: "15GB de roaming na Europa" → `Dest_PLMN_Zone_Europe`
- **Roaming global**: "1GB em qualquer lugar" → `Dest_Roaming_All`

Quando Usar Cada Abordagem

Tipo de Destino	Caso de Uso	Exemplo	Tipos de Saldo
Geográfico	Voz/SMS para países específicos	"100 minutos para números do Reino Unido"	Voz, SMS
PLMN On-net	Dados domésticos na SUA rede	"100GB de dados on-net"	Dados
PLMN Específico	Roaming premium em redes preferidas	"5GB apenas na Verizon"	Dados (e voz/SMS de roaming)
PLMN de País Inteiro	Roaming em qualquer rede no país	"5GB nos EUA (qualquer rede)"	Dados (e voz/SMS de roaming)
PLMN de Zona	Pacotes de roaming regionais	"10GB de roaming na Europa"	Dados (e voz/SMS de roaming)

Atualizações Dinâmicas de Destinos e Acesso ao Serviço

Como Funcionam as Atualizações de Destinos

Conceito Crítico: Os destinos no CGRateS são **referenciados por ID**, não copiados. Quando você atualiza os prefixos ou códigos PLMN de um destino e **carrega o TariffPlan no tempo de execução**, **todos os saldos existentes que referenciam esse destino são imediatamente afetados**.

Isso significa:

- **Não há necessidade de atualizar saldos individuais de clientes** ao expandir ou restringir a cobertura do serviço
- **Mudanças em toda a rede** para todos os clientes que usam esse destino uma vez que o plano tarifário é carregado
- ⚠ **Mudanças entram em vigor quando você carrega o TariffPlan** - você pode modificar destinos várias vezes antes de carregar, e as mudanças só se tornam ativas após `LoadTariffPlanFromStorDb`

Exemplo 1: Expandindo a Cobertura

Você tem um destino para roaming europeu:

```
# Configuração inicial - Apenas 3 países
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_Zone_Europe",
        "Prefixes": [
            "mcc262",  # Alemanha
            "mcc208",  # França
            "mcc222"   # Itália
        ]
    }]
})
```

Clientes com saldos referenciando `Dest_PLMN_Zone_Europe` podem fazer roaming na Alemanha, França e Itália.

Mais tarde, você adiciona Espanha e Reino Unido:

```
# Configuração atualizada - Agora 5 países
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_Zone_Europe",
        "Prefixes": [
            "mcc262",  # Alemanha
            "mcc208",  # França
            "mcc222",  # Itália
            "mcc214",  # Espanha (NOVO)
            "mcc234"   # Reino Unido (NOVO)
        ]
    }]
})

# Carregue o plano tarifário para tornar as mudanças ativas
OCS_Obj.SendData({
    'method': 'ApierV1.LoadTariffPlanFromStorDb',
    'params': [{
        "TPid": tpid,
        "DryRun": False,
        "Validate": True
    }]
})

# Recarregue o cache de destinos
OCS_Obj.SendData({
    'method': 'CacheSv1.ReloadCache',
    'params': [{
        "DestinationIDs": ["Dest_PLMN_Zone_Europe"]
    }]
})
```

Resultado: Após carregar o plano tarifário, todos os clientes existentes com saldos para `Dest_PLMN_Zone_Europe` ganham acesso para fazer roaming na Espanha e no Reino Unido - nenhuma atualização de saldo individual é necessária.

Exemplo 2: Restringindo a Cobertura

Remover um país de um destino bloqueia o acesso uma vez que o plano tarifário é carregado:

```
# Remover a Alemanha da zona da Europa
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_Zone_Europe",
        "Prefixes": [
            "mcc208",  # França
            "mcc222",  # Itália
            "mcc214",  # Espanha
            "mcc234"   # Reino Unido
            # Alemanha (mcc262) REMOVIDA
        ]
    }]
})

# Carregue o plano tarifário para tornar as mudanças ativas
OCS_Obj.SendData({
    'method': 'ApierV1.LoadTariffPlanFromStorDb',
    'params': [{
        "TPid": tpid,
        "DryRun": False,
        "Validate": True
    }]
})

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

Resultado: Após carregar o plano tarifário, clientes que tentarem fazer roaming na Alemanha serão **bloqueados**, mesmo que ainda tenham saldo

restante para `Dest_PLMN_Zone_Europe`. O saldo existe, mas não corresponde mais ao destino PLMN.

Exemplo 3: Destinos Geográficos de Voz/SMS

O mesmo princípio se aplica a destinos geográficos para voz e SMS:

```
# Adicionar Canadá à chamada "Doméstica"
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_Domestic_All",
        "Prefixes": [
            "1",  # Já inclui os EUA (e anteriormente o Canadá)
            # Nenhuma mudança necessária - o prefixo "1" já cobre
ambos
        ]
    }]
})

# Ou crie um destino separado para a América do Norte
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_North_America",
        "Prefixes": [
            "1"  # EUA + Canadá
        ]
    }]
})
```

Impacto: Quaisquer saldos com `DestinationIDs: "Dest_Domestic_All"` agora funcionam para chamadas tanto dos EUA quanto do Canadá (se usando o prefixo "1").

Regras de Acesso ao Serviço

Crítico: Se um cliente tentar usar um serviço (voz/SMS/dados) e **nenhum destino corresponder**, o serviço será **negado**, mesmo que tenham saldos.

Cenário 1: Nenhum Destino Correspondente

```
# Cliente tem saldo:
{
    "BalanceId": "Europe_Data_5GB",
    "BalanceType": "*data",
    "DestinationIDs": "Dest_PLMN_Zone_Europe",  # Apenas França,
Alemanha, Itália
    "Units": 5 * 1024 * 1024 * 1024
}

# Cliente faz roaming em rede na Espanha (mcc214.mnc001)
# Espanha NÃO está em Dest_PLMN_Zone_Europe
```

Resultado:

- □ Sessão de dados **bloqueada** (nenhum destino correspondente)
- Saldo permanece não utilizado
- Cliente recebe erro "Crédito Insuficiente" ou similar

Correção: Adicione a Espanha a `Dest_PLMN_Zone_Europe` OU garanta que o cliente tenha um saldo monetário com `DestinationIDs: "*any"` como fallback.

Cenário 2: Sem Fallback Coringa

```
# Cliente tem dois saldos:
{
    "BalanceId": "UK_Voice_100min",
    "BalanceType": "*voice",
    "DestinationIDs": "Dest_International_UK",  # Apenas prefixo
"44"
    "Units": 100 * 60 * 1000000000
}
# SEM saldo monetário com destino "*any"

# Cliente chama a Austrália (+61)
```

Resultado:

- ❌ Chamada **bloqueada** (nenhum destino corresponde ao prefixo "61")
- Saldo de minutos do Reino Unido não utilizado (apenas corresponde a "44")
- Sem fallback monetário para cobrir a chamada

Correção: Adicione um saldo monetário com `DestinationIDs: "*any"` OU adicione um destino específico da Austrália ao saldo deles.

Cenário 3: Fallback Coringa Funciona

```
# Cliente tem:
# Saldo 1: minutos do Reino Unido
{
    "BalanceId": "UK_Voice_100min",
    "BalanceType": "*voice",
    "DestinationIDs": "Dest_International_UK",
    "Units": 100 * 60 * 1000000000,
    "BalanceWeight": 1200
}

# Saldo 2: Monetário com coringa
{
    "BalanceId": "PAYG_Credit",
    "BalanceType": "*monetary",
    "DestinationIDs": "*any",  # Combina com QUALQUER COISA
    "Units": 5000,  # $50
    "BalanceWeight": 1000  # Prioridade mais baixa
}

# Cliente chama a Austrália (+61)
```

Resultado:

- □ Chamada **permitida**
- Saldo de minutos do Reino Unido NÃO é utilizado (não corresponde a "61")
- Saldo de Crédito PAYG é utilizado (corresponde a "*any")
- Cobrado na tarifa PAYG para a Austrália

Testes e Simulações

Mudanças de destinos podem ser testadas e simuladas em ambientes não produtivos antes de serem aplicadas à produção. Isso permite que você:

- Verifique se os novos códigos PLMN estão corretos
- Teste o impacto da expansão/restrição da cobertura
- Valide que os saldos existentes se comportam como esperado com os destinos atualizados
- Garanta que os mecanismos de fallback funcionem corretamente

Recomendação: Sempre teste atualizações de destinos em um ambiente de staging com contas de teste antes de aplicar à produção.

Principais Conclusões

1. **Destinos são dinâmicos** - Mudanças afetam todos os saldos que os referenciam uma vez que o plano tarifário é carregado
2. **Carregue o TariffPlan para ativar** - Use `LoadTariffPlanFromStorDb` para tornar as mudanças de destino ativas no tempo de execução
3. **Nenhuma correspondência de destino = Nenhum serviço** - Clientes serão bloqueados se seu uso não corresponder a nenhum destino
4. **Coringa (`*any`) é poderoso** - Use para fallback monetário para evitar bloqueios inesperados
5. **Recarregamento de cache necessário** - Sempre recarregue o cache de destinos após carregar o plano tarifário
6. **Teste antes da produção** - Mudanças de destinos impactam todos os clientes uma vez carregadas

Perfis de Tarifas para Saldos PAYG/Monetários

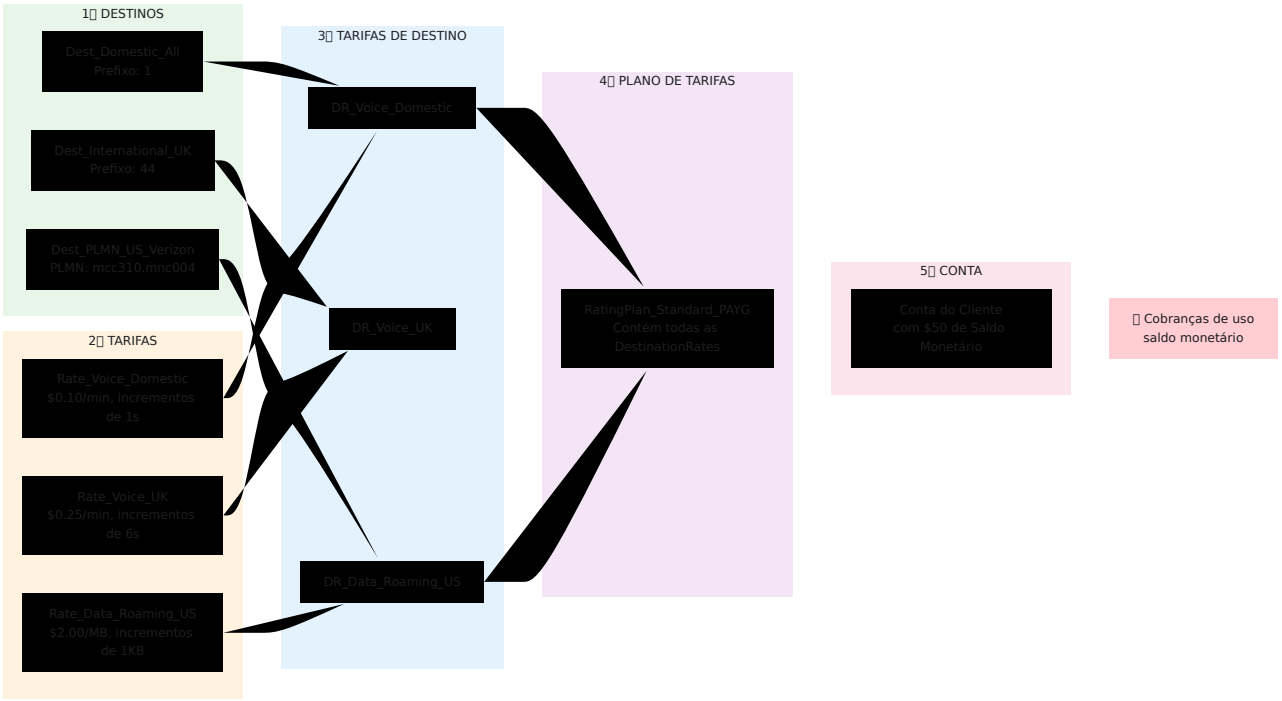
Ao usar saldos monetários (PAYG - Pague Conforme Usa), você precisa definir **Perfis de Tarifas** que especificam quanto cobrar por unidade para diferentes destinos.

Nota Importante sobre a Flexibilidade do CGRateS: O CGRateS é extremamente flexível e pode aplicar diferentes tarifas e planos com base em muitos fatores, incluindo:

- De onde o cliente está chamando (PLMN - coberto aqui)
- Para onde estão chamando (destino geográfico - coberto aqui)
- Hora do dia (tarifação em horário de pico/fora de pico)
- Nível do cliente (VIP, padrão, etc.)
- Nível de qualidade do serviço
- E muitos outros cenários de tarifação complexos

Este guia foca na **configuração em nível de CRM** para roaming baseado em PLMN e tarifação baseada em destinos geográficos, que cobre os casos de uso mais comuns. Cenários de tarifação avançados (hora do dia, níveis de clientes, roteamento baseado em qualidade) são possíveis no CGRateS, mas estão fora do escopo desta documentação focada em CRM.

Como Funcionam os Perfis de Tarifas



Definindo Perfis de Tarifas

Os perfis de tarifas definem o custo por unidade para diferentes tipos de uso. Aqui estão exemplos para voz, SMS e dados:

```
#
# ============================================================================
# TARIFAS DE VOZ (Por Minuto)
#
# ============================================================================

# Voz doméstica: $0.10/minuto
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPRate',
    'params': [{
        "TPid": tpid,
        "ID": "Rate_Voice_Domestic",
        "RateSlots": [{
            "ConnectFee": 0,
            "Rate": 0.10,
            "RateUnit": "60s",         # 1 minuto
            "RateIncrement": "1s",     # Cobrança por segundo
            "GroupIntervalStart": "0s"
        }]
    }]
})

# Voz no Reino Unido: $0.25/minuto
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPRate',
    'params': [{
        "TPid": tpid,
        "ID": "Rate_Voice_UK",
        "RateSlots": [{
            "ConnectFee": 0.05,        # $0.05 de taxa de conexão
            "Rate": 0.25,
            "RateUnit": "60s",
            "RateIncrement": "6s",     # Blocos de 6 segundos
            "GroupIntervalStart": "0s"
        }]
    }]
})

#
# ============================================================================
# TARIFAS DE SMS (Por Mensagem)
#
# ============================================================================
```

```
# SMS doméstico: $0.05/mensagem
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",     # Por mensagem
            "GroupIntervalStart": "0s"
        }]
    }]
})

#
# ============================================================================
# TARIFAS DE DADOS (Por MB) - Para Roaming
#
# ============================================================================

# Roaming em Redes Premium dos EUA (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 MB em bytes
            "RateIncrement": "1024",  # Por KB
            "GroupIntervalStart": "0s"
        }]
    }]
})
```

Campos de Slot de Tarifa:

- **ConnectFee** - Cobrança única por sessão
- **Rate** - Custo por RateUnit

- **RateUnit** - Tamanho da unidade de cobrança (60s, 1MB, 1 mensagem)
- **RateIncrement** - Unidade mínima cobrável
- **GroupIntervalStart** - Quando este nível de tarifa começa

Vinculando Destinos a Tarifas

Após definir as tarifas, vincule-as aos destinos usando DestinationRates:

```
# Vincular destino doméstico à tarifa de voz doméstica
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
        }]
    }]
})

# Vincular destino do Reino Unido à tarifa de voz do Reino Unido
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
        }]
    }]
})

# Vincular PLMN da Verizon à tarifa de dados 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
        }]
```

```
    }]
})
```

Criando o Plano de Tarifas

Combine todas as DestinationRates em um RatingPlan:

```
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPRatingPlan',
    'params': [{
        "TPid": tpid,
        "ID": "RatingPlan_Standard_PAYG",
        "RatingPlanBindings": [
            # Tarifas de voz (peso maior = mais específico)
            {"DestinationRatesId": "DR_Voice_UK", "TimingId":
"*any", "Weight": 40},
            {"DestinationRatesId": "DR_Voice_Domestic",
"TimingId": "*any", "Weight": 20},

            # Tarifas de SMS
            {"DestinationRatesId": "DR_SMS_Domestic", "TimingId":
"*any", "Weight": 20},

            # Tarifas de dados de roaming
            {"DestinationRatesId": "DR_Data_Roaming_US_Premium",
"TimingId": "*any", "Weight": 50}
        ]
    }]
})
```

Carregando o Plano Tarifário

Finalmente, carregue o plano tarifário no CGRateS:

```
# Carregar plano tarifário
OCS_Obj.SendData({
    'method': 'ApierV1.LoadTariffPlanFromStorDb',
    'params': [{
        "TPid": tpid,
        "DryRun": False,
        "Validate": True
    }]
})

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

Veja também:

- Definindo Produtos - Fluxo de trabalho completo para criar produtos
- Ações do CGRateS e Comportamentos de Recarga - Como definir Ações e abordagens de gerenciamento de saldo

Melhores Práticas

1. Seja Específico com On-Net

```
# Bom - rede doméstica específica
"DestinationIDs": "Dest_PLMN_OnNet",  # mcc505.mnc057

# Ruim - muito amplo, combina com concorrentes
"DestinationIDs": "Dest_PLMN_AU_All",  # mcc505 (todas as
operadoras da AU)
```

2. Use Nomes de Destinos Significativos

```
# Bom - auto-documentação
Dest_PLMN_US_Verizon
Dest_International_UK
Dest_PLMN_Zone_Europe

# Ruim - pouco claro
Dest_Data_1
Dest_Voice_Package
```

3. Agrupe Destinos Relacionados Logicamente

```
# Bom - agrupamento lógico
Dest_Domestic_All
Dest_Domestic_TollFree
Dest_Domestic_Premium

# Ruim - muito granular
Dest_Area_Code_212
Dest_Area_Code_213
Dest_Area_Code_214
```

4. Documente Fontes de PLMN

Ao definir destinos PLMN, adicione comentários indicando de onde os códigos MCC/MNC vieram:

```
# Verizon Wireless
# Fonte: https://www.mcc-mnc.com/ (verificado 2024-12)
OCS_Obj.SendData({
    'method': 'ApierV2.SetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_US_Verizon",
        "Prefixes": [
            "mcc310.mnc004",  # Verizon - Nacional
            "mcc310.mnc010",  # Verizon - Leste
            ...
        ]
    }]
})
```

Resolução de Problemas

Problema: Saldo Não Correspondendo ao Destino

Sintoma: O cliente tem saldo, mas não consegue usá-lo para um destino específico

Verifique:

```
# 1. Verifique se o destino existe
OCS_Obj.SendData({
    'method': 'ApierV2.GetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_US_Verizon"
    }]
})

# 2. Verifique o campo DestinationIDs do saldo
# Certifique-se de que inclui o destino ou "*any"

# 3. Para PLMN: Verifique se o cliente está realmente nessa rede
# Verifique mensagens CCR de Diâmetro para o AVP Visited-PLMN-Id
```

Problema: Uso de Dados Contado como Roaming

Sintoma: Uso de dados domésticos consumindo saldo de roaming em vez de saldo on-net

Causa: PLMN do cliente não definido em `Dest_PLMN_OnNet`

Solução:

```
# Verifique em qual PLMN o cliente está (a partir do CCR de Diâmetro)
# Verifique se o saldo inclui esse PLMN:

OCS_Obj.SendData({
    'method': 'ApierV2.GetTPDestination',
    'params': [{
        "TPid": tpid,
        "ID": "Dest_PLMN_OnNet"
    }]
})
# Certifique-se de que os Prefixes incluam a rede PLMN real do cliente
# Exemplo: ["mcc505.mnc057"] deve corresponder à rede do SIM do cliente
```

Problema: Chamadas Internacionais Não Funcionando

Sintoma: O cliente tem "minutos internacionais", mas as chamadas falham

Verifique:

1. **Prefixo do destino**: O número discado corresponde ao prefixo do destino?
   - Chamando +44-20-xxxx → Precisa de destino com prefixo "44"
2. **DestinationIDs do saldo**: O saldo inclui esse destino?
   - Verifique se o saldo tem `"DestinationIDs": "Dest_International_UK"` ou `"Dest_International_All"`
3. **Saldo não expirado**: Verifique `ExpiryTime`
4. **Saldo tem unidades**: Verifique `Units > 0`

Documentação Relacionada

- Ações do CGRateS e Comportamentos de Recarga - Como usar destinos em Ações, abordagens de saldo e estratégias de tarifação
- Definindo Produtos - Fluxo de trabalho completo para criação de produtos

Ansible Playbooks: Guia Detalhado

Os produtos OmniCRM são provisionados usando **Ansible**, permitindo a gestão automatizada de serviços com base nos requisitos específicos de cada produto e seu inventário associado.

Veja também: `Provisionamento de Cartão SIM <concepts_sim_provisioning>` para um exemplo completo de provisionamento baseado em Ansible para serviços móveis, incluindo SIMs físicos e eSIMs.

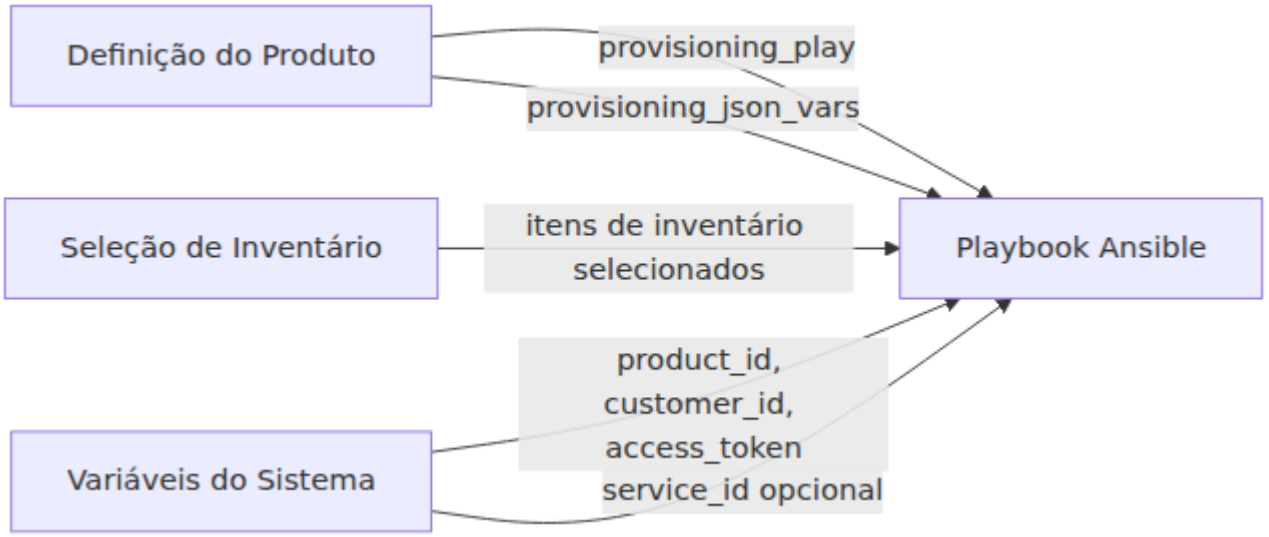
Como Playbooks e Produtos Trabalham Juntos

Conceito Crítico: Playbooks são o que realmente cria serviços em OmniCRM. Quando você atribui um playbook a um produto, está definindo **o que acontece** quando esse produto é provisionado - mas isso pode significar coisas diferentes para produtos diferentes.

Produtos Disparam Playbooks

Quando um produto é provisionado no OmniCRM:

1. A definição do produto especifica qual playbook executar (via campo `provisioning_play`)
2. O produto passa variáveis para o playbook (via `provisioning_json_vars` e seleções de inventário)
3. O playbook é executado e faz o que foi programado para fazer
4. O playbook determina o que será criado (se houver algo)



O Que os Playbooks Podem Fazer

Um único playbook de provisionamento pode:

Criar Múltiplos Serviços
Um playbook de produto agrupado pode criar:

- Um registro principal de serviço de internet
- Um registro de serviço de addon IPTV
- Um registro de serviço VoIP
- Tudo com uma única ação de provisionamento de produto

Não Criar Serviços
Alguns playbooks não criam registros de serviço:

- Um playbook que apenas configura equipamentos CPE
- Um playbook que envia configuração para equipamentos de rede
- Um playbook que atualiza sistemas externos

Criar Um Serviço
O padrão mais comum:

- Criar um único registro de serviço para o cliente
- Vincular inventário a esse serviço
- Configurar a cobrança para esse serviço

Modificar Serviços Existentes
Playbooks de recarga e addon:

- Não criam novos serviços
- Atualizam registros de serviços existentes (adicionam dados, estendem a expiração, etc.)
- Adicionam saldos a contas de cobrança existentes

Executar Ações Sem Registros de Serviço
Alguns playbooks são puramente operacionais:

- Redefinir saldos de conta
- Trocar itens de inventário entre clientes
- Gerar relatórios ou configurações

Exemplo: Comportamentos Diferentes de Playbook

```
# Produto 1: Serviço de SIM Móvel (cria 1 serviço)
# provisioning_play: play_simple_service
- Cria registro de serviço no CRM
- Cria conta de cobrança no OCS
- Atribui inventário de cartão SIM e número de telefone
- Envia e-mail de boas-vindas

# Produto 2: Pacote de Internet (cria 3 serviços)
# provisioning_play: play_bundle_internet_tv_voice
- Cria registro de serviço de internet
- Cria registro de serviço IPTV
- Cria registro de serviço VoIP
- Vincula todos ao mesmo cliente
- Conta de cobrança única para o pacote

# Produto 3: Recarga de Dados (cria 0 serviços)
# provisioning_play: play_topup_no_charge
- Encontra serviço existente por service_id
- Adiciona saldo de dados à conta OCS existente
- Atualiza data de expiração do serviço
- NENHUM novo serviço criado

# Produto 4: Configuração CPE (cria 0 serviços)
# provisioning_play: play_prov_cpe_mikrotik
- Gera configuração do roteador
- Atualiza registro de inventário com configuração
- Envia configuração para a equipe de suporte
- NENHUM serviço criado (apenas configuração do equipamento)
```

O ponto chave: **O playbook define o comportamento, o produto é apenas um gatilho.**

Plays vs Tarefas

Entender a distinção entre Plays e Tarefas é fundamental para trabalhar com playbooks do OmniCRM.

Play (Playbook)
Um fluxo de trabalho de provisionamento completo que orquestra várias tarefas para alcançar um objetivo de negócios. Plays são os playbooks de nível superior armazenados em `OmniCRM-API/Provisioners/plays/` e são referenciados nas definições de produtos.

Exemplos:

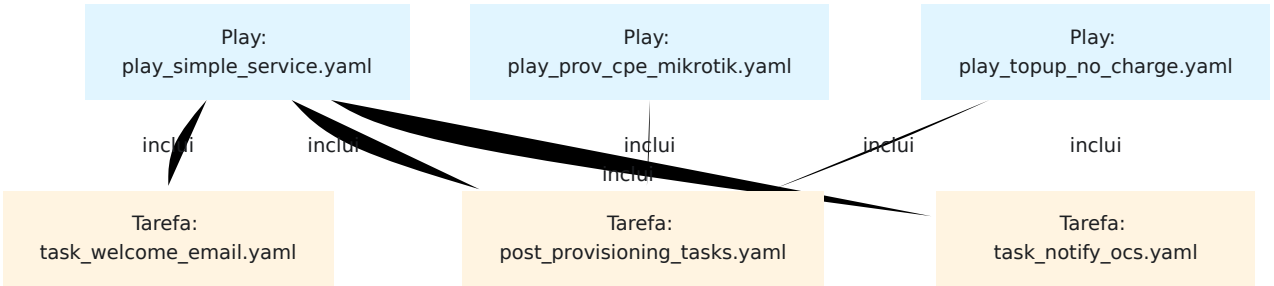
- `play_simple_service.yaml` - Provisionar um serviço básico
- `play_topup_no_charge.yaml` - Aplicar uma recarga gratuita a um serviço
- `play_prov_cpe_mikrotik.yaml` - Configurar equipamentos de cliente

Tarefa (Componente Reutilizável)
Um conjunto de operações autônomas e reutilizáveis que podem ser incluídas por múltiplas plays. Tarefas são prefixadas com `task_` e vivem no mesmo diretório.

Exemplos:

- `task_welcome_email.yaml` - Enviar um e-mail de boas-vindas a um cliente
- `task_activate_olt.yaml` - Ativar equipamentos OLT
- `task_notify_ocs.yaml` - Enviar notificações para o sistema de cobrança

A relação entre eles:



```yaml
# play_simple_service.yaml (Uma Play)
- name: Play de Provisionamento Simples
  hosts: localhost
  tasks:
    - name: Bloco de provisionamento principal
      block:
        - name: Criar serviço
          uri: ...

        - name: Configurar cobrança
          uri: ...

        # Incluir tarefa reutilizável
        - include_tasks: task_welcome_email.yaml

        # Incluir tarefas pós-provisionamento
        - include_tasks: post_provisioning_tasks.yaml
```

Estrutura e Anatomia do Playbook

Todos os playbooks do OmniCRM seguem uma estrutura consistente. Compreender essa estrutura é essencial para criar e manter playbooks.

Estrutura Básica

Todo playbook começa com esses cabeçalhos padrão:

```yaml
- name: Nome Descritivo do Playbook
  hosts: localhost           # Sempre localhost para OmniCRM
  gather_facts: no           # Desabilitado para desempenho
  become: False              # Não elevar privilégios

  tasks:
    - name: Bloco principal
      block:
        # Tarefas de provisionamento vão aqui

      rescue:
        # Tarefas de rollback/limpeza vão aqui
```

Explicação do Cabeçalho

name
Nome descritivo mostrado nos logs de provisionamento e na interface do usuário. Isso aparece como `playbook_description` no registro de provisionamento.

hosts: localhost
Todos os playbooks do OmniCRM são executados em localhost, pois interagem com sistemas remotos via APIs, não SSH.

gather_facts: no
A coleta de fatos do Ansible está desabilitada porque:

- Não precisamos de informações do sistema
- Adiciona sobrecarga desnecessária
- Pode travar navegadores se exibido na saída de depuração

become: False
Nenhuma elevação de privilégio é necessária, pois estamos fazendo chamadas de API, não modificando arquivos de sistema.

Carregamento de Configuração

Todo playbook deve carregar o arquivo de configuração central:

```
tasks:
  - name: Incluir vars do crm_config
    ansible.builtin.include_vars:
      file: "../../crm_config.yaml"
      name: crm_config
```

Isso torna a configuração disponível como `crm_config.ocs.cgrates`, `crm_config.crm.base_url`, etc.

O `crm_config.yaml` normalmente contém:

```
ocs:
  cgrates: "10.0.1.100:2080"
  ocsTenant: "default_tenant"
crm:
  base_url: "https://crm.example.com"
```

Padrões de Acesso a Variáveis

As variáveis podem vir de várias fontes:

Da Definição do Produto:

```
- name: Acessar product_id passado pelo OmniCRM
  debug:
    msg: "Provisionando produto {{ product_id }}"
```

Da Seleção de Inventário:

```
- name: Obter ID de inventário para Cartão SIM
  set_fact:
    sim_card_id: "{{ hostvars[inventory_hostname]['SIM Card'] |
int }}"
  when: "'SIM Card' in hostvars[inventory_hostname]"
```

Das Respostas da API:

```
- name: Obter informações do Produto da API do 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: Usar o nome do produto
  debug:
    msg: "O nome do produto é {{
api_response_product.json.product_name }}"
```

Padrões Comuns de Playbook

Padrão de Provisionamento de Serviço

Este é o padrão mais comum para criar novos serviços.

```yaml
- name: Playbook de Provisionamento de Serviço
  hosts: localhost
  gather_facts: no
  become: False

  tasks:
    - name: Bloco principal
      block:

      # 1. Carregar configuração
      - name: Incluir vars do crm_config
        ansible.builtin.include_vars:
          file: "../../crm_config.yaml"
          name: crm_config

      # 2. Obter informações do produto
      - name: Obter informações do Produto da API do 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. Obter informações do cliente
      - name: Obter informações do Cliente da API do 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. Definir fatos a partir dos dados recuperados
      - name: Definir fatos do pacote
        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. Gerar identificadores únicos
      - name: Gerar UUID
        set_fact:
          uuid: "{{ 99999999 | random | to_uuid }}"

      - name: Gerar UUID do Serviço
        set_fact:
          service_uuid: "Service_{{ uuid[0:8] }}"

      # 6. Criar conta no sistema de cobrança
      - name: Criar conta no 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: Verificar criação da conta OCS
        assert:
```

```
      that:
        - ocs_response.status == 200
        - ocs_response.json.result == "OK"

    # 7. Adicionar saldo inicial
    - name: Adicionar 0 Saldo Monetário
      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": "Saldo Inicial",
                    "Value": 0,
                    "ExpiryTime": "+4320h",
                    "Weight": 1,
                    "Blocker": true
                }
            }]
          }
        status_code: 200
      register: balance_response

    # 8. Criar registro de serviço no CRM
    - name: Obter data e hora atuais em formato ISO 8601
      command: date --utc +%Y-%m-%dT%H:%M:%S%z
      register: current_date_time

    - name: Adicionar Serviço 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. Adicionar transação de custo de configuração
      - name: Adicionar Transação de Custo de Configuração 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 }} - Custos de
Configuração",
              "description": "Custos de configuração para {{
package_comment }}",
              "invoice_id": null,
              "retail_cost": "{{ setup_cost | float }}"
            }
          return_content: yes
        register: transaction_response

      # 10. Incluir tarefas pós-provisionamento
```

```
      - include_tasks: post_provisioning_tasks.yaml

      rescue:

      # Seção de rollback/limpeza
      - name: Imprimir todas as variáveis para depuração
        debug:
          var: hostvars[inventory_hostname]

      - name: Remover conta no 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: Deletar Serviço do CRM se foi criado
        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: Falhar se não for desprovisionamento intencional
        assert:
          that:
            - action == "deprovision"
```

Padrão de Recarga

Usado para adicionar créditos, dados ou tempo a serviços existentes.

```yaml
- name: Playbook de Recarga de Serviço
  hosts: localhost
  gather_facts: no
  become: False

  tasks:
    - name: Incluir vars do crm_config
      ansible.builtin.include_vars:
        file: "../../crm_config.yaml"
        name: crm_config

    # 1. Obter informações do serviço
    - name: Obter informações do Serviço da API do 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. Obter informações do produto (o que recarregar)
    - name: Obter informações do Produto da API do 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. Extrair detalhes do serviço
    - name: Definir fatos do serviço
      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. Executar ação no sistema de cobrança (recarga gratuita)
    - name: Executar Ação para adicionar créditos
      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: Verificar se a ação foi executada com sucesso
      assert:
        that:
          - action_response.status == 200
          - action_response.json.result == "OK"

    # 5. Redefinir quaisquer limites acionados
    - name: Redefinir 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. Atualizar datas do serviço
    - name: Calcular nova data de expiração
      command: "date --utc +%Y-%m-%dT%H:%M:%S%z -d '+30 days'"
```

```
      register: new_expiry_date

    - name: Atualizar Serviço com nova expiração
      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. Opcional: Enviar notificação
    - name: Enviar SMS de Notificação
      uri:
        url: "http://sms-gateway/api/send"
        method: POST
        body_format: json
        body:
          {
            "source": "NomeDaEmpresa",
            "destination": "{{ customer_phone }}",
            "message": "Seu serviço foi recarregado. Nova
expiração: {{ new_expiry_date.stdout }}"
          }
        status_code: 201
      ignore_errors: True
```

Padrão de Provisionamento CPE

Usado para configurar equipamentos de cliente (roteadores, modems, ONTs).

```yaml
- name: Playbook de Provisionamento CPE
  hosts: localhost
  gather_facts: no
  become: False

  tasks:
    - name: Incluir vars do crm_config
      ansible.builtin.include_vars:
        file: "../../crm_config.yaml"
        name: crm_config

    # 1. Obter item de inventário para CPE
    - name: Definir ID de inventário CPE a partir de hostvars
      set_fact:
        cpe_inventory_id: "{{ hostvars[inventory_hostname]['WiFi
Router CPE'] | int }}"
      when: "'WiFi Router CPE' in hostvars[inventory_hostname]"

    # 2. Obter detalhes do CPE do inventário
    - name: Obter dados de inventário para 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. Obter informações do site do cliente
    - name: Obter informações do Site da 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. Atualizar inventário CPE com localização
    - name: Patch item de inventário CPE com localização
      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. Gerar credenciais
    - name: Definir nome do host CPE
      set_fact:
        cpe_hostname: "CPE_{{ cpe_inventory_id }}"
        cpe_username: "admin_{{ cpe_inventory_id }}"

    - name: Gerar senha aleatória
      set_fact:
        cpe_password: "{{ lookup('pipe', 'cat /dev/urandom | tr -
dc a-zA-Z0-9 | head -c 16') }}"

    # 6. Gerar credenciais WiFi
    - name: Definir SSID WiFi
      set_fact:
        wifi_ssid: "Rede_{{ cpe_inventory_id }}"

    - name: Gerar senha WiFi
      set_fact:
        word_list:
          - apple
          - cloud
          - river
          - mountain
          - ocean

    - name: Criar PSK WiFi
```

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

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

    # 7. Gerar arquivo de configuração
    - name: Definir nome do arquivo de configuração
      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: Criar configuração a partir do template
      template:
        src: "templates/cpe_router_config.j2"
        dest: "{{ config_dest }}"

    # 8. Ler configuração gerada
    - name: Ler arquivo de configuração
      ansible.builtin.slurp:
        src: "{{ config_dest }}"
      register: config_content

    # 9. Atualizar inventário com informações de provisionamento
    - name: Patch CPE inventário com configuração
      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": "Provisionado: {{ lookup('pipe',
'date +%Y-%m-%d') }}"
          }
        status_code: 200

    # 10. Enviar configuração para a equipe de suporte
    - name: Enviar configuração por e-mail para suporte
      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": "Sistema de Provisionamento"
              },
              "To": [{
                "Email": "support@example.com",
                "Name": "Equipe de Suporte"
              }],
              "Subject": "Configuração 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
```

Padrão de Renovação Automática

Configurar cobranças ou renovações recorrentes automáticas usando CGRateS ActionPlans.

```
# Parte de um playbook de recarga que configura a renovação
automática

# 1. Normalizar parâmetro auto_renew
- name: Normalizar auto_renew para booleano
  set_fact:
    auto_renew_bool: "{{ (auto_renew | string | lower) in ['true',
'1', 'yes'] }}"

# 2. Criar ação para renovação automática
- name: Criar Ação para AutoRenovação
  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. Criar ActionPlan mensal
```

```
- name: Criar ActionPlan para Renovação Mensal
  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. Atribuir ActionPlan à conta
- name: Atribuir ActionPlan à conta
  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. Remover ActionPlan se auto-renovação desativada
- name: Remover ActionPlan da conta
  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
```

Tarefas Reutilizáveis

Tarefas reutilizáveis são pequenos playbooks autônomos que podem ser incluídos por múltiplas plays. Elas promovem a reutilização de código e a consistência.

Tarefa de E-mail de Boas-Vindas

`task_welcome_email.yaml` - Envia um e-mail de boas-vindas para novos clientes.

```
# Esta tarefa espera que essas variáveis sejam definidas pela play pai:
# - api_response_customer (detalhes do cliente)
# - package_name (nome do produto)
# - monthly_cost (custo recorrente)
# - setup_cost (custo único)

- name: Definir configuração de 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: Definir assunto do e-mail e nome do remetente
  set_fact:
    email_subject: "Bem-vindo ao nosso serviço!"
    email_from_name: "Equipe de Atendimento ao Cliente"

- name: Preparar lista de destinatários a partir dos contatos do cliente
  loop: "{{ api_response_customer.json.contacts }}"
  set_fact:
    recipients: "{{ recipients + [{'Email': item.contact_email, 'Name': item.contact_firstname ~ ' ' ~ item.contact_lastname}] }}"

- name: Obter nome do primeiro contato
  set_fact:
    first_contact: "{{ api_response_customer.json.contacts[0].contact_firstname }}"

- name: Enviar e-mail de boas-vindas
  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": "Caro {{ first_contact }}, bem-vindo! Seu
serviço está pronto.",
          "HTMLPart": "Caro {{ first_contact }},<br/><h3>Bem-
vindo!</h3><br/>Seu serviço {{ package_name }} agora está ativo.
<br/>Custo mensal: ${{ monthly_cost }}<br/>Taxa de configuração:
${{ setup_cost }}<br/>Se você tiver alguma dúvida, entre em
contato com support@example.com"
        }]
      }
    user: "{{ mailjet_api_key }}"
    password: "{{ mailjet_api_secret }}"
    force_basic_auth: true
    status_code: 200
  register: email_response
```

Tarefas Pós-Provisionamento

`post_provisioning_tasks.yaml` - Limpeza padrão e notificações executadas após cada provisionamento.

```
# Este arquivo é incluído no final da maioria dos playbooks de
provisionamento
# Ele lida com operações comuns pós-provisionamento

- include_tasks: task_notify_ocs.yaml
```

O `task_notify_ocs.yaml` pode conter:

```
- name: Notificar OCS sobre a conclusão do provisionamento
  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
```

Operações Comuns

Trabalhando com Inventário

Recuperando detalhes do inventário:

```
- name: Obter ID de inventário do Cartão SIM
  set_fact:
    sim_inventory_id: "{{ hostvars[inventory_hostname]['SIM Card']
| int }}"
  when: "'SIM Card' in hostvars[inventory_hostname]"

- name: Obter detalhes do Cartão 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: Extrair detalhes do SIM
  set_fact:
    iccid: "{{ sim_response.json.iccid }}"
    imsi: "{{ sim_response.json.imsi }}"
    ki: "{{ sim_response.json.ki }}"
```

Atribuindo inventário ao cliente:

```
- name: Atribuir SIM ao cliente
  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
```

Operações de Data e Hora

Obtendo data/hora atual:

```
- name: Obter data e hora atuais em formato ISO 8601
  command: date --utc +%Y-%m-%dT%H:%M:%S%z
  register: current_date_time

- name: Obter apenas a data de hoje
  set_fact:
    today: "{{ lookup('pipe', 'date +%Y-%m-%d') }}"
```

Calculando datas futuras:

```
- name: Calcular data de expiração 30 dias a partir de agora
  command: "date --utc +%Y-%m-%dT%H:%M:%S%z -d '+30 days'"
  register: expiry_date

- name: Calcular data 90 dias no futuro
  command: "date --utc +%Y-%m-%d -d '+{{ days }} days'"
  register: future_date
  vars:
    days: 90
```

Gerando Valores Aleatórios

UUIDs e identificadores:

```
- name: Gerar UUID
  set_fact:
    uuid: "{{ 99999999 | random | to_uuid }}"

- name: Gerar identificador de serviço
  set_fact:
    service_uuid: "SVC_{{ uuid[0:8] }}"
```

Senhas aleatórias:

```
- name: Gerar senha segura
  set_fact:
    password: "{{ lookup('pipe', 'cat /dev/urandom | tr -dc a-zA-
Z0-9 | head -c 16') }}"
```

Frases de senha memoráveis:

```
- name: Definir lista de palavras
  set_fact:
    words:
      - alpha
      - bravo
      - charlie
      - delta
      - echo

- name: Gerar frase de senha
  set_fact:
    word: "{{ words | random }}"
    number: "{{ 99999 | random(start=10000) }}"

- name: Combinar em frase de senha
  set_fact:
    passphrase: "{{ word }}{{ number }}"
```

Trabalhando com CGRateS/OCS

Criando contas:

```
- name: Criar conta de cobrança
  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
```

Adicionando saldos:

```
- name: Adicionar saldo de dados
  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": "Pacote de Dados",
            "Value": 10737418240,
            "ExpiryTime": "+720h",
            "Weight": 10
          }
        }]
      }
    status_code: 200
```

Executando ações:

```
- name: Executar ação de cobrança
  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
```

Obtendo informações da conta:

```
- name: Obter detalhes da conta
  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
```

Trabalhando com Perfis de Atributos:

```
- name: Obter Perfil de Atributo
  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: Extrair valor do atributo
  set_fact:
    phone_number: "{{ attr_response.json.result.Attributes |
json_query(\"[?Path=='*req.PhoneNumber'].Value[0].Rules\") | first
}}"
  when: attr_response is defined
```

Lógica Condicional

Verificando se variáveis existem:

```
- name: Usar valor personalizado ou padrão
  set_fact:
    monthly_cost: "{{ custom_cost | default(50.00) }}"

- name: Executar apenas se a variável estiver definida
  debug:
    msg: "UUID do Serviço é {{ service_uuid }}"
  when: service_uuid is defined
```

Condições booleanas:

```yaml
- name: Provisionar equipamento
  include_tasks: configure_cpe.yaml
  when: provision_cpe | default(false) | bool

- name: Pular se desprovisionar
  assert:
    that:
      - action != "deprovision"
  when: action is defined
```

Múltiplas condições:

```yaml
- name: Tarefa condicional complexa
  uri:
    url: "{{ endpoint }}"
    method: POST
  when:
    - service_uuid is defined
    - customer_id is defined
    - action != "deprovision"
    - enable_feature | default(true) | bool
```

Laços e Iteração

Laços simples:

```
- name: Criar múltiplos saldos
  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 }
```

Iterando sobre respostas da API:

```
- name: Obter todos os sites do cliente
  uri:
    url: "{{ crm_config.crm.base_url }}/crm/site/customer_id/{{
customer_id }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
  register: sites_response

- name: Configurar equipamento em cada site
  debug:
    msg: "Configurando site em {{ item.address_line_1 }}"
  loop: "{{ sites_response.json }}"
```

Tratamento de Erros

Usando ignore_errors:

```yaml
- name: Notificação SMS opcional
  uri:
    url: "http://sms-gateway/send"
    method: POST
    body: {...}
  ignore_errors: true
```

Aserções para validação:

```yaml
- name: Verificar resposta da API
  assert:
    that:
      - response.status == 200
      - response.json.result == "OK"
    fail_msg: "A chamada da API falhou: {{ response.json }}"
```

Tratamento de erros condicional:

```yaml
- name: Tentar obter serviço existente
  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: Criar serviço se não existir
  uri:
    url: "{{ crm_config.crm.base_url }}/crm/service/"
    method: PUT
    body: {...}
  when: service_lookup.status == 404
```

Melhores Práticas

Nomeação de Variáveis

Use nomes descritivos e consistentes:

```yaml
# Bom
service_uuid: "SVC_12345"
customer_name: "John Smith"
monthly_cost: 49.99

# Ruim
svc: "SVC_12345"
name: "John Smith"
cost: 49.99
```

Prefixe variáveis por origem:

```yaml
api_response_customer: {...}
api_response_product: {...}
cgr_account_info: {...}
```

Depuração

Imprima variáveis para solução de problemas:

```yaml
- name: Imprimir todas as variáveis
  debug:
    var: hostvars[inventory_hostname]

- name: Imprimir variável específica
  debug:
    msg: "UUID do Serviço: {{ service_uuid }}"

- name: Imprimir resposta da API
  debug:
    var: api_response_product.json
```

Validação

Sempre valide respostas críticas da API:

```yaml
- name: Criar conta
  uri:
    url: "{{ billing_endpoint }}"
    method: POST
    body: {...}
  register: response

- name: Verificar criação da conta
  assert:
    that:
      - response.status == 200
      - response.json.result == "OK"
    fail_msg: "Falha ao criar conta: {{ response.json }}"
```

Idempotência

Projete tarefas para serem executáveis novamente com segurança:

```yaml
# Verifique se o recurso existe primeiro
- name: Verificar se a conta existe
  uri:
    url: "{{ ocs_endpoint }}/get_account"
    method: POST
    body: {"Account": "{{ service_uuid }}"}
  register: account_check
  failed_when: false

# Crie apenas se não existir
- name: Criar conta
  uri:
    url: "{{ ocs_endpoint }}/create_account"
    method: POST
    body: {...}
  when: account_check.status == 404
```

Segurança

Nunca codifique credenciais:

```
# Ruim
mailjet_api_key: "abc123def456"

# Bom - use variáveis de ambiente
mailjet_api_key: "{{ lookup('env', 'MAILJET_API_KEY') }}"

# Bom - use arquivo de configuração
mailjet_api_key: "{{ crm_config.email.api_key }}"
```

Sempre use HTTPS e autenticação:

```
- name: Chamar API externa
  uri:
    url: "https://api.example.com/endpoint"
    method: POST
    headers:
      Authorization: "Bearer {{ access_token }}"
    validate_certs: yes
```

Documentação

Documente lógica complexa:

```
# Calcular cobrança pro-rata para mês parcial
# Se o cliente assinar no dia 15 e a cobrança for no dia 1,
# cobre 50% do custo mensal pelos dias restantes
- name: Calcular dias até o final do mês
  command: "date -d 'last day of this month' +%d"
  register: days_in_month

- name: Obter dia atual
  command: "date +%d"
  register: current_day

- name: Calcular valor pro-rata
  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) }}"
```

Testando Playbooks

Abordagem de Teste

1. **Executar Teste Primeiro**: Teste com sistemas não produtivos
2. **Verificar Variáveis**: Use tarefas de depuração para confirmar que todas as variáveis necessárias estão presentes
3. **Verificar Respostas**: Valide respostas da API antes de prosseguir
4. **Teste de Rollback**: Falhe intencionalmente tarefas para verificar se os blocos de resgate funcionam
5. **Teste de Desprovisionamento**: Teste com `action: "deprovision"` para verificar a limpeza

Exemplo de playbook de teste:

```yaml
- name: Testar Provisionamento de Serviço
  hosts: localhost
  gather_facts: no

  tasks:
    - name: Verificar variáveis necessárias
      assert:
        that:
          - product_id is defined
          - customer_id is defined
          - access_token is defined
        fail_msg: "Variáveis necessárias ausentes"

    - name: Testar conectividade da API
      uri:
        url: "http://localhost:5000/crm/health"
        method: GET
      register: health_check

    - name: Verificar verificação de saúde
      assert:
        that:
          - health_check.status == 200
```

Armadilhas Comuns

Conversões de tipo ausentes:

```yaml
# Errado - pode ser string
customer_id: "{{ customer_id }}"

# Correto - garantir que seja inteiro
customer_id: {{ customer_id | int }}
```

Não lidar com variáveis indefinidas:

```
# Errado - falha se não definido
service_uuid: "{{ service_uuid }}"

# Correto - fornecer padrão
service_uuid: "{{ service_uuid | default('') }}"
```

Esquecer validação:

```
# Errado - não verifica resposta
- name: Criar conta
  uri: ...
  register: response

# Correto - valida resposta
- name: Criar conta
  uri: ...
  register: response

- name: Verificar criação
  assert:
    that:
      - response.json.result == "OK"
```

Fluxo de Trabalho de Provisionamento

Geralmente, a equipe da Omnitouch trabalhará com o cliente para:

1. Definir os requisitos do produto
2. Desenvolver os playbooks Ansible necessários para automatizar o processo de provisionamento
3. Testar os playbooks em um ambiente de homologação
4. Implantar em produção

Isso garante que cada serviço seja implantado de forma consistente e confiável, reduzindo o risco de erros e garantindo que todas as etapas necessárias sejam concluídas na ordem correta.

Variáveis Ansible

As variáveis passadas para os playbooks Ansible incluem:

Variáveis do Produto
Derivadas das configurações de produto do OmniCRM e definem como o serviço deve ser configurado.

Variáveis de Inventário
Selecionadas do inventário, incluem itens como modems, cartões SIM, blocos de endereços IP ou números de telefone que são necessários para o provisionamento.

Variáveis do Sistema
Adicionadas automaticamente pelo OmniCRM:

- `product_id` - O produto sendo provisionado
- `customer_id` - O cliente recebendo o serviço
- `service_id` - O serviço sendo modificado (para recargas/mudanças)
- `access_token` - JWT para autenticação da API

Desprovisionamento

Quando um serviço não é mais necessário, os **Playbooks Ansible** também são usados para desprovisionar o serviço usando o padrão de bloco `rescue`. Isso:

- Remove quaisquer configurações
- Libera o inventário de volta para o pool
- Deleta contas de cobrança
- Garante que o sistema seja mantido limpo

Métodos de Desprovisionamento

Existem duas maneiras principais de acionar o desprovisionamento:

1. Rollback Automático (Falha no Provisionamento) Quando qualquer tarefa no bloco principal de provisionamento falha, a seção de resgate é

executada automaticamente para limpar alterações parciais.

2. Desprovisionamento Manual Definir `action: "deprovision"` ao executar um playbook aciona o bloco de resgate intencionalmente para remover um serviço.

Padrão de Desprovisionamento

Todos os playbooks do OmniCRM seguem esta estrutura para limpeza segura:

```yaml
- name: Playbook de Provisionamento de Serviço
  hosts: localhost
  gather_facts: no
  become: False

  tasks:
    - name: Bloco principal
      block:
        # Todas as tarefas de provisionamento vão aqui
        - name: Criar conta OCS
          uri: ...

        - name: Criar registro de serviço
          uri: ...
          register: service_creation_response

        - name: Adicionar transação
          uri: ...

      rescue:
        # Tarefas de desprovisionamento/rollback vão aqui
        - name: Imprimir todas as variáveis para depuração
          debug:
            var: hostvars[inventory_hostname]

        # Tarefas de limpeza são executadas na ordem inversa
        - name: Remover conta no OCS
          uri: ...

        - name: Deletar Serviço do CRM
          uri: ...

        - name: Falhar se não for desprovisionamento intencional
          assert:
            that:
              - action == "deprovision"
```

Exemplo Completo de Desprovisionamento

Aqui está um exemplo completo do `play_simple_service.yaml`:

```
rescue:
  # 1. Informações de depuração
  - name: Imprimir todas as variáveis para
Desprovisionamento/Rollback
    debug:
      var: hostvars[inventory_hostname]

  - name: Obter data de hoje
    set_fact:
      today: "{{ lookup('pipe', 'date +%Y-%m-%d') }}"

  # 2. Obter informações do serviço, se disponível
  - name: Tentar obter informações do Serviço da API do CRM para
Desprovisionar
    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: Imprimir api_response_service
    debug:
      var: api_response_service
    when: api_response_service is defined

  # 3. Definir service_uuid para limpeza
  - name: Definir fatos de service_uuid para Desprovisionamento
    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. Remover conta OCS
  - name: Remover conta no 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: Imprimir resposta da remoção da conta OCS
    debug:
      var: response
    when: response is defined

  # 5. Deletar registro de serviço do CRM
  - name: Deletar Serviço do CRM se foi criado
    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: Imprimir resposta de deletar serviço
    debug:
      var: delete_service_response
```

```
    when: delete_service_response is defined

  # 6. Determinar se isso foi intencional ou uma falha
  - name: Definir status como "Sucesso" se desprovisionamento
manual / Falhar se falha de provisionamento
    assert:
      that:
        - action == "deprovision"
      fail_msg: "Provisionamento falhou e foi revertido"
      success_msg: "Serviço desprovisionado com sucesso"
```

Conceitos Chave de Desprovisionamento

ignore_errors: True A maioria das tarefas de limpeza usa `ignore_errors: True` porque os recursos podem não existir (por exemplo, se o provisionamento falhou antes de criá-los).

Execução Condicional Use cláusulas `when` para limpar apenas recursos que foram criados:

```
when: service_creation_response is defined
```

Ordem Inversa A limpeza acontece na ordem inversa da criação:

1. Deletar recursos dependentes primeiro (transações, atribuições de inventário)
2. Deletar registro de serviço
3. Deletar conta OCS/cobrança
4. Liberar quaisquer recursos retidos

Aserção Final A asserção final distingue entre:

- **Desprovisionamento Intencional** (`action == "deprovision"`) → Sucesso
- **Falha de Provisionamento** (ação não definida) → Falha

Desprovisionamento de Diferentes Tipos de Recursos

Contas OCS/CGRateS:

```
- name: Remover conta no 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
```

Registros de Serviço do CRM:

```
- name: Deletar Serviço do 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
```

Itens de Inventário (Retornar ao Pool):

```
- name: Liberar cartão SIM de volta ao pool de inventário
  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 (Cobranças Recorrentes):

```
- name: Remover ActionPlan da conta
  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
```

Autorizações de Pagamento:

```
- name: Liberar autorização de pagamento
  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
```

Fluxo de Trabalho de Desprovisionamento Manual

Para desprovisionar um serviço manualmente:

1. Identifique o serviço a ser desprovisionado
2. Execute o playbook de provisionamento original com `action: "deprovision"`
3. O playbook entra no bloco de resgate imediatamente
4. Todas as tarefas de limpeza são executadas
5. O serviço é removido de forma limpa

Exemplo de chamada da API:

```
POST /crm/provision/run_playbook
{
  "product_id": 123,
  "customer_id": 456,
  "service_id": 789,
  "action": "deprovision"
}
```

Cenários de Limpeza Parcial

Cenário 1: Conta OCS Criada, Criação de Serviço Falhou

- A conta OCS existe
- O registro de serviço NÃO existe
- O bloco de resgate remove a conta OCS
- Nenhum serviço para deletar (safamente pulado com condição `when`)

Cenário 2: Serviço Criado, Transação Falhou

- A conta OCS existe
- O registro de serviço existe
- A transação NÃO existe
- O bloco de resgate remove tanto a conta OCS quanto o serviço
- Nenhuma transação para anular (nunca criada)

Cenário 3: Provisionamento Completo, Captura de Pagamento Falhou

- A conta OCS existe
- O registro de serviço existe
- Pagamento autorizado, mas NÃO capturado
- O bloco de resgate:
  - Libera autorização de pagamento (cliente não cobrado)
  - Remove registro de serviço
  - Remove conta OCS

Melhores Práticas para Desprovisionamento

1. Use ignore_errors liberalmente A limpeza deve ser tolerante - não falhe se um recurso não existir.

2. Verifique a existência do recurso com cláusulas when Tente limpar apenas se o recurso foi criado:

```
when: service_creation_response is defined
```

3. Imprima informações de depuração Sempre inclua tarefas de depuração para ajudar a solucionar falhas de provisionamento:

```
- name: Imprimir todas as variáveis para depuração
  debug:
    var: hostvars[inventory_hostname]
```

4. Limpe na ordem inversa da dependência Deletar filhos antes dos pais:

- Transações antes de serviços
- Serviços antes de contas OCS
- Atribuições de inventário antes de liberações de inventário

5. Lide com autorizações de pagamento Sempre libere autorizações de pagamento em blocos de resgate para evitar cobranças de clientes por provisionamentos falhados.

6. Recarregue agendadores após a limpeza Ao remover recursos OCS, inclua `ReloadScheduler: true` para garantir que CGRateS atualize imediatamente.

Desprovisionamento vs Exclusão

Desprovisionamento (via bloco de resgate):

- Remove serviço de todos os sistemas
- Libera inventário

- Cancela cobranças recorrentes
- Deixa trilha de auditoria
- Abordagem recomendada

Exclusão Direta (via API):

- Apenas deleta registro do CRM
- NÃO limpa conta OCS
- NÃO libera inventário
- Pode deixar recursos órfãos
- Não recomendado para produção

Rollback e Tratamento de Erros

O recurso **block/rescue** do Ansible é empregado durante o provisionamento e desprovisionamento para lidar com erros de forma elegante. Se uma tarefa falhar em qualquer ponto durante o provisionamento, a seção de resgate reverte automaticamente as alterações para retornar a um estado consistente. Isso garante confiabilidade e reduz o risco de implantações parciais ou falhadas.

Exemplo de Tratamento de Erros

```
- name: Bloco principal
  block:
    - name: Etapa 1: Criar conta
      uri: ...
      register: response

    - name: Verificar etapa 1
      assert:
        that:
          - response.status == 200

    - name: Etapa 2: Criar serviço
      uri: ...

  rescue:
    # Se qualquer asserção falhar ou qualquer tarefa gerar erro:
    # 1. A execução salta para o bloco de resgate
    # 2. Tarefas de limpeza são executadas
    # 3. O playbook falha com mensagem de erro
    - name: Limpar alterações parciais
      uri: ...
```

Para detalhes completos sobre o sistema de provisionamento, fluxos de trabalho e autenticação, veja `concepts_provisioning`.

OmniCRM API

Todas as funções dentro do OmniCRM estão acessíveis via API - Não há funcionalidade que esteja disponível apenas na UI.

Isso permite que você integre o OmniCRM com outros sistemas ou automatize tarefas.

A API é uma API RESTful e é protegida usando múltiplos métodos de autenticação, incluindo tokens JWT, chaves de API e lista de permissões de IP.

A documentação da API é documentada usando Swagger, uma ferramenta que permite fácil leitura, compreensão e teste da funcionalidade da API.

A documentação da API está disponível no seguinte URL:

> <https://yourcrm/crm/docs/>

Métodos de Autenticação

O OmniCRM suporta três métodos de autenticação, cada um projetado para casos de uso diferentes:

1. **Tokens Bearer JWT** - Para sessões de usuários interativas (UI Web, aplicativos móveis)
2. **Chaves de API** - Para integrações servidor-a-servidor e scripts de automação
3. **Lista de Permissões de IP** - Para sistemas internos confiáveis (servidores de provisionamento, ferramentas de monitoramento)

Autenticação com Token Bearer JWT

Este é o método de autenticação principal para sessões de usuários. Os usuários fazem login com e-mail e senha, recebem um token JWT e o utilizam para solicitações subsequentes.

Casos de Uso:

- Autenticação na UI Web
- Autenticação em aplicativos móveis
- Acesso programático de curta duração

Como se Autenticar:

Para fazer login, envie um corpo JSON com a seguinte estrutura para `/crm/auth/login` como uma solicitação POST:

```
{
    "email": "youruser@yourdomain.com",
    "password": "yourpassword"
}
```

A API retornará um objeto JSON contendo um campo `token`, que é usado para autenticar todas as solicitações futuras. Além disso, a resposta inclui um `refresh_token` que pode ser usado para atualizar o token quando ele expira, juntamente com as permissões e funções do usuário.

Você pode testar isso na página do Swagger selecionando o endpoint `/auth/login`, preenchendo seu Nome de Usuário e Senha, e clicando no botão `Try it out`.

Para autorizar a sessão, copie o valor do token e clique no botão "Authorize" no canto superior direito da página do Swagger. Cole o token no campo "Value", prefixado por `Bearer` e clique em "Authorize".

Agora, todas as solicitações subsequentes serão autenticadas com este token.

Autenticação com Chave de API

As chaves de API fornecem autenticação segura e de longa duração para integrações servidor-a-servidor e scripts de automação sem exigir senhas de usuários.

Casos de Uso:

- Sistemas de provisionamento automatizados
- Ferramentas de monitoramento e alerta
- Integração com sistemas externos
- Tarefas agendadas e cron jobs

Como as Chaves de API Funcionam:

As chaves de API são configuradas no arquivo `crm_config.yaml` e estão associadas a funções e permissões específicas. Cada chave de API é uma string aleatória segura (mínimo de 32 caracteres) que autentica solicitações quando passada no cabeçalho `X-API-KEY`.

Configurando Chaves de API:

As chaves de API devem ser adicionadas ao `crm_config.yaml` por um administrador com acesso ao servidor:

```yaml
api_keys:
  your-secure-api-key-here-minimum-32-chars:
    roles:
      - admin
    description: "Sistema de automação de provisionamento"
  another-api-key-for-monitoring-system:
    roles:
      - view_customer
      - view_service
    description: "Monitoramento e alerta"
```

Usando Chaves de API:

Inclua a chave de API no cabeçalho `X-API-KEY` de suas solicitações:

```
curl -X GET "https://yourcrm.com/crm/customers" \
  -H "X-API-KEY: your-secure-api-key-here-minimum-32-chars"
```

Exemplo em Python:

```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
}

# Obter Clientes
response = requests.get(crm_url + '/crm/customers',
headers=headers)
for customer in response.json()['data']:
    print(customer)
```

Melhores Práticas:

- Gere chaves de API usando geradores aleatórios criptograficamente seguros (`openssl rand -base64 48`)
- Use chaves de API diferentes para sistemas diferentes
- Documente o propósito de cada chave de API no campo `description`
- Rotacione as chaves de API periodicamente
- Nunca comprometa chaves de API no controle de versão
- Atribua permissões mínimas necessárias a cada chave de API

Autenticação com Lista de Permissões de IP

A lista de permissões de IP permite que endereços IP específicos acessem a API sem autenticação. Isso é útil para sistemas internos confiáveis em redes privadas.

Casos de Uso:

- Servidores internos de provisionamento
- Sistemas de monitoramento de rede em VLANs de gerenciamento
- Playbooks do Ansible executando em infraestrutura controlada

Configurando a Lista de Permissões de IP:

Adicione endereços IP confiáveis ao `crm_config.yaml`:

```
ip_whitelist:
  - 192.168.1.100
  - 10.0.0.0/24
  - 172.16.50.10
```

Considerações de Segurança:

- Use a lista de permissões de IP apenas em redes privadas e seguras
- Nunca adicione endereços IP públicos à lista de permissões
- Use os intervalos de IP mais específicos possíveis
- Documente por que cada IP está na lista de permissões

- Audite regularmente os IPs na lista de permissões

Exemplos de Chamadas de API com Python

Aqui está um exemplo de como fazer login e recuperar uma lista de clientes usando autenticação com token JWT:

```python
import requests

crm_url = 'https://yourcrm.com'
session = requests.Session()

print("Provisionando dados para o servidor: " + str(crm_url))

headers = {
    "Content-Type": "application/json"
}

# Obter Token de Autenticação
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("Autenticado no CRM com sucesso")

# Obter Clientes
response = session.get(crm_url + '/crm/customers',
headers=headers)
for customer in response.json()['data']:
    print(customer)
```

Monitoramento de Desempenho da API

Todas as solicitações da API são automaticamente rastreadas com métricas abrangentes. Para detalhes sobre métricas de desempenho da API, incluindo taxas de solicitação, tempos de resposta, taxas de erro e estatísticas específicas de endpoint, consulte Monitoramento & Métricas.

O endpoint de métricas está disponível em `/crm/metrics` no formato Prometheus para monitoramento e alerta.

Documentação Relacionada

- Arquitetura do Sistema - Design geral do sistema e relacionamentos de componentes
- Monitoramento & Métricas - Referência completa de métricas, incluindo métricas de API, banco de dados e integração
- Fluxos de Autenticação - Fluxos detalhados de autenticação e autorização
- RBAC - Controle de acesso baseado em funções e permissões

Notas de Cobrança de Serviço / Produto CRM

Nota

Para um guia completo de ponta a ponta cobrindo definição de produto, provisionamento, complementos e desprovisionamento com exemplos detalhados de Ansible e estratégia de preços, veja `Complete Product Lifecycle Guide <guide_product_lifecycle>`.

Para informações detalhadas sobre provisionamento de serviços móveis com cartões SIM, veja `SIM Card Provisioning <concepts_sim_provisioning>`.

Visão Geral de Produtos e Serviços

Produto (Item do Menu):

Um Produto é como um prato específico em um menu de restaurante, como um "Spaghetti Carbonara."

Ele tem uma descrição clara, uma lista de ingredientes (como massa, creme, ovos, queijo e bacon) e um preço.

No OmniCRM, um Produto contém de forma semelhante os detalhes do que está incluído — recursos, especificações e preços.

Frequentemente, os clientes podem querer modificações, como "sem cebolas" ou "adicionar queijo extra" ao seu prato. Dentro do OmniCRM, isso corresponde a personalizar um serviço antes da criação. O nível de personalizações ou modificações a um serviço fica a seu critério (o operador) para definir.

No OmniCRM, os clientes ou funcionários podem modificar um Produto para melhor atender às necessidades de um cliente específico, como aumentar a

velocidade da Internet ou adicionar recursos específicos. Essa personalização é refletida no Serviço específico fornecido.

Um produto é essencialmente uma oferta que os clientes podem escolher para solicitar, semelhante a ler e escolher um prato no menu.

Catálogo de Produtos (Menu do Restaurante):

O Catálogo de Produtos é como o menu completo em um restaurante, que lista todos os pratos disponíveis — de aperitivos a sobremesas.

É a coleção completa de tudo que o restaurante (ou, no seu caso, o Provedor de Serviços) tem a oferecer.

No contexto empresarial, o Catálogo de Produtos fornece aos clientes todos os Produtos disponíveis, para que possam escolher aquele que melhor atende às suas necessidades.

Serviço (Prato Preparado):

Quando um cliente solicita um item do menu, o prato é preparado na cozinha. Isso é semelhante a criar um Serviço a partir de um Produto.

No OmniCRM, quando um cliente seleciona um Produto, uma instância desse Produto é criada e entregue como um Serviço.

Ele é personalizado e preparado especificamente para aquele cliente, assim como uma refeição preparada para um cliente.

Por exemplo, quando alguém seleciona o "Plano de Internet Bronze" do Catálogo de Produtos, o sistema de provisionamento "prepara" uma instância desse plano a partir dos ingredientes (endereços IP, Modems e Portas) — ou seja, ativa o plano e o entrega ao cliente específico.

Produtos Agrupados (Refeições Combinadas):

O Catálogo de Produtos também pode oferecer pacotes, como uma refeição combinada que inclui um aperitivo, prato principal e sobremesa juntos por um preço especial.

No OmniCRM, Produtos agrupados combinam vários Produtos individuais em um pacote conveniente — como um "Pacote de Essenciais para o Lar" que inclui serviços de Internet, TV e telefone a uma taxa com desconto.

Uma vez selecionado, esse pacote é transformado em vários Serviços adaptados para o cliente.

Definições de Produto

Um produto é um modelo que é usado para criar um serviço / complemento / desconto / adicional, etc.

Dentro da definição incluímos:

- Informações sobre o produto (recursos, inclusões, T&Cs, duração do contrato, ícone, etc.) que são exibidas para o usuário do CRM (Cliente ou *Funcionário*).
- A lógica de negócios sobre quem pode comprar o produto (*Empresarial* ou *Residencial*), se depende de ter um serviço pai provisionado (como complementos móveis disponíveis apenas para clientes com um serviço móvel), se pode ser solicitado diretamente por um cliente via autoatendimento ou apenas por um agente de atendimento ao cliente, e

quando o produto pode ser adquirido (permitindo que um produto esteja disponível apenas por um período definido).

- Quando itens de Inventário devem ser incluídos (como Modems ou Cartões SIM) estes são especificados como Lista de Itens de Inventário, por exemplo, o serviço abaixo requer um Cartão SIM e um Número de Telefone a serem atribuídos:

  `['SIM Card', 'Phone Number']` Estes correlacionam-se com os `Inventory Items <administration_inventory>` definidos no CRM.

- Referenciar um Playbook Ansible para provisionar o serviço `Provisioning Play <concepts_ansible>` assim como as variáveis a passar para o Ansible. Essas variáveis a passar são mágicas, pois podem ser variáveis como `service_id` que são definidas pelo produto ao qual estamos adicionando, ou podem ser como `ICCID` & `MSISDN` onde temos selecionado itens de inventário que são passados ao atribuir o inventário. O agrupamento é tratado no playbook de provisionamento para conter vários serviços, por exemplo, um produto de internet residencial, TV e Voz pode provisionar um serviço para cada.

Categorias de Produto e Tipos de Serviço

Os produtos usam dois campos de classificação para ajudar a organizar e filtrar ofertas:

Categorias de Produto

O campo `category` controla onde os produtos são exibidos na interface do usuário. Valores comuns incluem:

- **standalone** - Exibido como uma opção de serviço base ao criar um novo serviço
- **addon** - Exibido ao adicionar a um serviço existente
- **bundle** - Exibido como uma opção de serviço agrupado (provisionado como um complemento a serviços existentes)
- **promo** - Ofertas promocionais especiais

Essas categorias são puramente organizacionais e não ditam o que é provisionado. O comportamento real de provisionamento é determinado inteiramente pelo playbook Ansible referenciado em `provisioning_play`.

Por exemplo: - Um produto `standalone` tipicamente cria um novo objeto de serviço - Um produto `addon` ou `bundle` é tipicamente adicionado a um serviço existente - Mas isso fica a critério do implementador que escreve o playbook - você pode criar múltiplos objetos de serviço a partir de um complemento, ou modificar serviços existentes a partir de um produto autônomo, se necessário

A categoria simplesmente controla o fluxo da interface do usuário e onde os clientes/funcionários veem a opção de produto.

Tipos de Serviço

O campo `service_type` categoriza que tipo de serviço está sendo fornecido.

Esses são definidos inteiramente pelo usuário, mas valores comuns incluem:

- **mobile** - Serviços de telefone móvel com voz, SMS e dados
- **fixed** - Serviços de internet fixa sem fio ou com fio
- **fixed-voice** - Serviços de voz fixa (VoIP, linha fixa)
- **hotspot** - Dispositivos móveis de hotspot ou de aluguel
- **dongle** - Serviços de modem USB ou dongle
- **voice** - Serviços apenas de voz
- **data** - Serviços apenas de dados

Assim como as categorias, os tipos de serviço são personalizáveis com base em suas ofertas. Eles ajudam em:

- Filtrar quais complementos se aplicam a quais serviços base

- Organizar produtos no portal do cliente
- Correspondência de requisitos de inventário
- Determinar fluxos de trabalho de provisionamento

Exemplo: Um cliente com um serviço `mobile` pode ver complementos móveis, enquanto um cliente com um serviço `fixed` vê complementos de linha fixa.

Gerenciando Produtos

Os produtos são gerenciados através da página de Gerenciamento de Produtos, onde você pode visualizar, pesquisar, filtrar e editar todos os produtos disponíveis.

Interface Modal de Produto

Clicando em qualquer produto, abre uma interface de abas aprimorada que organiza todas as configurações do produto em grupos lógicos para facilitar a navegação e edição.

O modal de gerenciamento de produtos possui cinco abas organizadas:

1. □ **Informações Básicas** - Informações principais do produto (nome, slug, categoria, ícone, recursos, termos)
2. □ **Preços** - Todos os campos relacionados a custos, incluindo custos recorrentes, custos de configuração e porcentagem de imposto
3. ⚙ **Configuração** - Configurações de renovação, tipos de clientes e dependências
4. □ **Provisionamento** - Configuração do playbook Ansible e requisitos de inventário
5. □ **Disponibilidade** - Intervalos de datas e timestamps do sistema

Organização da Aba de Preços:

A aba de Preços agrupa campos de custo em seções lógicas:

- **Custos Recorrentes** - Custos mensais de varejo e atacado lado a lado
- **Custos de Configuração** - Taxas de ativação únicas para varejo e atacado
- **Imposto** - Configuração da porcentagem de imposto com cálculo automático

Recursos do Modo de Edição:

- **Selecionador de Ícone** - Pesquisar e selecionar ícones do FontAwesome visualmente
- **Selecionador de Itens de Inventário** - Selecionar a partir de tipos de itens de inventário disponíveis
- **Selecionador de Data/Hora** - Seleção fácil de janelas de disponibilidade
- **Formatação de Moeda** - Prefixo automático $ para campos de custo
- **Seletores de Dropdown** - Opções pré-definidas para categorias e campos booleanos

Selecionador de Ícone:

Ao editar o campo de ícone, uma interface de seletor de ícone pesquisável aparece permitindo que você navegue visualmente e selecione entre milhares de ícones do FontAwesome.

Recursos: * Pesquisar ícones por palavra-chave (por exemplo, "chave inglesa", "móvel", "wifi") * Visualizar a aparência do ícone em tempo real * Mostra o nome da classe do ícone para referência * Seleção de dropdown para acesso rápido

Aba de Configuração:

A aba de Configuração organiza as configurações de comportamento do produto em grupos lógicos.

Seções de Configuração:

- **Configurações de Renovação:**

    - Auto Renovar - Comportamento padrão de renovação (Prompt/Sim/Não)
    - Permitir Auto Renovar - Se os clientes podem habilitar a renovação automática
    - Dias do Contrato - Duração mínima do contrato (por exemplo, 30 para mensal, 365 para anual)
- **Tipos de Clientes:**
    - Residencial - Disponível para clientes consumidores
    - Empresarial - Disponível para clientes comerciais
- **Dependências:**
    - Lista de Dependências - IDs de produtos ou tipos de serviço necessários antes que este produto possa ser adicionado
    - Usado para dependências de complementos (por exemplo, complementos móveis requerem serviço móvel ativo)

Aba de Provisionamento:

A aba de Provisionamento lida com automação Ansible e requisitos de inventário.

Campos de Provisionamento:

- **Play de Provisionamento:**
    - Nome do playbook Ansible (sem extensão .yaml)
    - Deve existir no diretório OmniCRM-API/Provisioners/plays/

    - Chamado quando o serviço é criado, atualizado ou desprovisionado
- **Variáveis JSON de Provisionamento:**
    - Variáveis padrão passadas para o playbook Ansible como JSON
    - Podem ser substituídas durante o provisionamento
    - O playbook recebe essas além de `customer_id`, `product_id`, `service_id`, `access_token`
- **Lista de Itens de Inventário:**
    - Seletor de múltipla seleção mostrando tipos de itens de inventário disponíveis
    - Exemplos: Cartão SIM, Número de Telefone, Modem Roteador, Endereço IPv4
    - O cliente/funcionário seleciona itens específicos do inventário disponível durante o pedido
    - IDs de inventário selecionados são passados para o playbook com o tipo de inventário como nome da variável

Aba de Disponibilidade:

A aba de Disponibilidade controla quando o produto pode ser adquirido e exibe metadados do sistema.

Configurações de Disponibilidade:

- **Disponível A Partir de:**
    - Data/hora quando o produto se torna disponível para compra

    - Deixe em branco para disponibilidade imediata
    - Útil para pré-anunciar novos produtos
- **Disponível Até:**
    - Data/hora quando o produto não está mais disponível para compra
    - Deixe em branco para disponibilidade indefinida
    - Perfeito para promoções por tempo limitado ou produtos fora de linha
- **Metadados do Sistema (Somente Leitura):**
    - Criado - Timestamp quando o produto foi criado pela primeira vez
    - Última Modificação - Timestamp da atualização mais recente
    - Mantido automaticamente pelo sistema

Ações do Modal:

- **Modo de Visualização:**
    - Fechar - Descartar modal
    - Clonar Produto - Criar uma cópia com sufixo "_clone"
    - Editar Produto - Mudar para modo de edição
- **Modo de Edição/Criar:**
    - Cancelar - Descartar alterações e fechar
    - Salvar Alterações - Criar ou atualizar produto (botão grande para ênfase)

Campos do Produto

O modelo de Produto contém todas as informações necessárias para definir uma oferta e como ela deve ser provisionada. Esses campos são gerenciados através da interface modal de gerenciamento de produtos descrita acima.

Informações Básicas

- **product_id** - Identificador único atribuído automaticamente pelo sistema
- **product_name** - Nome exibido mostrado aos clientes e funcionários na interface do usuário
- **product_slug** - Identificador único usado em URLs e chamadas de API (minúsculas, sem espaços, use hífens)

- **category** - Controla onde este produto aparece na interface do usuário:
  - `standalone` - Exibido como uma opção de serviço base ao criar um novo serviço
  - `addon` - Exibido ao adicionar a um serviço existente
  - `bundle` - Exibido como uma opção de serviço agrupado
  - `promo` - Ofertas promocionais especiais
- **service_type** - Tipo de serviço sendo fornecido (por exemplo, móvel, fixo, voz-fixa, hotspot, dongle, voz, dados). Usado para filtrar quais complementos se aplicam a quais serviços.
- **comment** - Notas internas sobre o produto para referência dos funcionários apenas (não exibidas para os clientes)
- **icon** - Classe de ícone FontAwesome exibida na interface do usuário (por exemplo, `fa-solid fa-sim-card`)

Campos de Preço

- **retail_cost** - Cobrança mensal recorrente faturada ao cliente (defina como 0 para compras únicas ou produtos pré-pagos)
- **wholesale_cost** - Seu custo mensal para fornecer este serviço (usado para cálculos de margem)
- **retail_setup_cost** - Taxa de ativação ou configuração única cobrada ao cliente
- **wholesale_setup_cost** - Seu custo único para configurar o serviço
- **tax_percentage** - Porcentagem de imposto aplicada a este produto (por exemplo, 10 para 10%, 12.5 para 12.5%). Defina como 0 para produtos isentos de impostos. Esta taxa de imposto é aplicada automaticamente a transações criadas a partir deste produto.

Aplicação de Imposto:

Quando uma transação é criada a partir deste produto, a porcentagem de imposto é automaticamente copiada para a transação e o valor do imposto é calculado. Por exemplo:

- Produto com 10% de imposto, $50.00 custo de varejo → Transação tem $5.00 de imposto
- Produto com 0% de imposto (isento de impostos) → Transação tem $0.00 de imposto
- Sobrescrição manual da transação → Funcionários podem alterar a porcentagem de imposto por transação

Visibilidade e Acesso do Cliente

- **enabled** - Se este produto está ativo e disponível para compra (defina como falso para ocultar sem excluir)
- **residential** - Se clientes residenciais (consumidores) podem comprar este produto
- **business** - Se clientes empresariais (comerciais) podem comprar este produto
- **customer_can_purchase** - Se os clientes podem se auto-adquirir via o portal (verdadeiro) ou se apenas os funcionários podem adicioná-lo (falso)
- **available_from** - Data/hora quando este produto se torna disponível para compra (opcional)

- **available_until** - Data/hora quando este produto não está mais disponível para compra (opcional, útil para ofertas por tempo limitado)

Contrato e Renovação

- **contract_days** - Duração mínima do contrato em dias (por exemplo, 30 para mensal, 365 para anual, 0 para sem contrato mínimo)
- **auto_renew** - Comportamento padrão de renovação:
  - `prompt` - Pergunta ao cliente a cada vez se deseja renovar
  - `true` - Renova automaticamente sem perguntar
  - `false` - Requer renovação manual
- **allow_auto_renew** - Se os clientes podem habilitar a renovação automática (defina como falso para compras únicas)

Conteúdo Voltado para o Cliente

- **terms** - Termos e condições exibidos para os clientes antes da compra (inclua limitações, regras de expiração, condições de uso)
- **features_list** - Lista de recursos e inclusões mostradas aos clientes (formato de lista Python: `['Feature 1', 'Feature 2']`)

Configuração de Provisionamento

- **provisioning_play** - Nome do playbook Ansible que provisiona este serviço (sem extensão .yaml). Deve existir em `OmniCRM-API/Provisioners/plays/`.
- **provisioning_json_vars** - Variáveis padrão passadas para o playbook Ansible como JSON. Essas podem ser substituídas durante o provisionamento. O playbook recebe essas junto com `customer_id`, `product_id`, `service_id` e `access_token`.
- **inventory_items_list** - Lista de itens de inventário necessários para este produto (por exemplo, `['SIM Card', 'Mobile Number']`). Quando um cliente faz um pedido, será solicitado a selecionar itens específicos do inventário disponível. IDs de inventário selecionados são passados para o playbook com o tipo de inventário como o nome da variável.
- **relies_on_list** - Lista de IDs de produtos ou tipos de serviço que devem existir antes que este produto possa ser adicionado. Usado para

dependências de complementos (por exemplo, complementos móveis requerem um serviço móvel ativo).

Metadados do Sistema

- **created** - Timestamp quando o produto foi criado (definido automaticamente)
- **last_modified** - Timestamp quando o produto foi atualizado pela última vez (atualizado automaticamente)

Exemplos de Definições de Produto

Produto Autônomo (SIM Móvel)

```
{
  "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"
}
```


Este produto autônomo requer dois itens de inventário (Cartão SIM e Número Móvel) e cria um novo serviço quando provisionado.

Produto Adicional (Plano de Dados Mensal)


```
{
  "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"
}
```


Este produto adicional não requer inventário e é aplicado a um serviço existente. Ele cobra o cliente e adiciona créditos/saldos ao seu serviço.

Produto Agrupado (Pacote para Idosos)


```
{
  "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"
}
```


Este produto agrupado provisiona múltiplos serviços (Internet + IPTV + Telefone) usando um único playbook. Ele requer um item de inventário (Modem Roteador).

Produto Adicional (Recarga Simples)

```
{
  "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"
}
```

Este complemento simplesmente adiciona crédito monetário a um serviço existente. Nenhum inventário é necessário, e ele usa `service_id` para identificar qual serviço deve ser recarregado.

Como as Variáveis são Passadas para o Ansible

Entender como as variáveis fluem da definição do produto através da API para o playbook Ansible é crítico para escrever playbooks de provisionamento eficazes.

Fontes de Variáveis e Mesclagem

Quando um trabalho de provisionamento é criado, as variáveis vêm de várias fontes e são mescladas nesta ordem (fontes posteriores substituem as anteriores):

1. **Variáveis de provisioning_json_vars do Produto** - Variáveis padrão da definição do produto
2. **Corpo da Solicitação** - Variáveis passadas na chamada da API (podem substituir os padrões do produto)
3. **Variáveis adicionadas pelo Sistema** - Adicionadas automaticamente pelo sistema de provisionamento
4. **Seleções de Inventário** - IDs dos itens de inventário selecionados (se `inventory_items_list` não estiver vazio)

Processo de Mesclagem de Variáveis

O sistema mescla variáveis de todas as fontes, com fontes posteriores substituindo as anteriores. Isso permite uma personalização flexível no momento do provisionamento.

Por exemplo, se seu produto tiver:

```
"provisioning_json_vars": "{\"monthly_cost\": 50, \"data_gb\": 100}"
```

E sua solicitação API incluir:

```
{
  "product_id": 10,
  "customer_id": 456,
  "monthly_cost": 45,
  "custom_param": "value"
}
```

As variáveis finais `extra_vars` passadas para o Ansible serão:

```
{
    "monthly_cost": 45,         # Substituído pela solicitação
    "data_gb": 100,             # Da provisioning_json_vars
    "product_id": 10,           # Da solicitação
    "customer_id": 456,         # Da solicitação
    "custom_param": "value",    # Da solicitação
    "access_token": "eyJ..."    # Adicionado pelo sistema
}
```

Variáveis Adicionadas pelo Sistema

O sistema de provisionamento adiciona automaticamente:

- `access_token` - Token JWT para autenticar chamadas de API de volta ao CRM (fornecido diretamente para autenticação IP/API key, ou gerado a partir de `refresh_token` para autenticação de usuário)
- `initiating_user` - O ID do usuário que acionou o provisionamento (ou primeiro administrador para sistemas automatizados)
- Quaisquer campos do corpo da solicitação (`product_id`, `customer_id`, `service_id`, etc.)

Variáveis de Inventário

Quando um produto requer itens de inventário (por exemplo, `inventory_items_list: "['SIM Card', 'Mobile Number']"`), o processo funciona da seguinte forma:

1. **UI/API solicita seleção** - O usuário seleciona itens de inventário específicos do estoque disponível
2. **IDs de Inventário são adicionados às variáveis** - Os IDs dos itens de inventário selecionados são adicionados com o tipo de inventário como o nome da variável
3. **Playbook acessa IDs de inventário** - O playbook de provisionamento pode então recuperar detalhes completos do inventário da API do CRM

Por exemplo, se um usuário selecionar: - Cartão SIM com inventory_id: 789 - Número Móvel com inventory_id: 101

As variáveis passadas para o playbook incluem: - `SIM Card`: 789 - `Mobile Number`: 101

O playbook pode então usar esses IDs para buscar os registros completos de inventário (ICCID, IMSI, MSISDN, etc.) da API do CRM e usar essas informações para provisionar o serviço em equipamentos de rede.

Como o Ansible Recebe Variáveis

O sistema de provisionamento passa todas as variáveis mescladas para o playbook Ansible como `extravars`. Dentro do playbook, essas variáveis estão disponíveis através do sistema padrão de variáveis do Ansible e podem ser usadas em tarefas.

As variáveis podem ser referenciadas diretamente nas tarefas do playbook usando a sintaxe `{{ variable_name }}`. Por exemplo, `{{ product_id }}`, `{{ customer_id }}`, `{{ monthly_cost }}`, etc.

Variáveis Passadas para Produtos Adicionais

Quando um produto adicional é provisionado, o sistema passa automaticamente:

- `product_id` - O ID do produto adicional sendo provisionado
- `customer_id` - O cliente que possui o serviço
- `service_id` - O ID do serviço ao qual este complemento está sendo adicionado (crítico para complementos)
- `access_token` - Token de autenticação para chamadas de API
- Quaisquer variáveis de `provisioning_json_vars`
- Quaisquer variáveis adicionais da solicitação API

Exemplo de Fluxo de Provisionamento de Complemento

Quando um cliente adiciona o complemento "£5 Topup" ao seu serviço móvel (service_id: 123), o playbook recebe variáveis incluindo:

- `product_id`: 45 (o produto de recarga)
- `customer_id`: 456 (o cliente)
- `service_id`: 123 (o serviço para o qual adicionar crédito)
- `access_token`: Token de autenticação
- Além de quaisquer variáveis de `provisioning_json_vars` do produto

O playbook então usa essas variáveis para:

1. **Buscar detalhes do serviço** da API do CRM usando o `service_id`
2. **Extrair o UUID do serviço** e outras informações do registro do serviço
3. **Adicionar crédito ao sistema de faturamento** (OCS) usando o UUID do serviço
4. **Registrar a transação** no CRM para fins de faturamento

Esse fluxo permite que o complemento identifique exatamente qual serviço modificar e aplique as alterações de forma apropriada.

Diferença Entre Variáveis de Produtos Autônomos e Adicionais

Produtos Autônomos recebem:

- `product_id` - O produto sendo provisionado
- `customer_id` - O cliente que está solicitando o serviço
- IDs de itens de inventário (por exemplo, `SIM Card`, `Mobile Number`) se o produto os requer
- `access_token` - Para autenticação da API

Produtos Adicionais recebem:

- `product_id` - O produto adicional sendo provisionado
- `customer_id` - O cliente que possui o serviço
- `service_id` - **O ID do serviço existente a ser modificado** (essa é a diferença chave)
- `access_token` - Para autenticação da API

A diferença chave é `service_id` - isso informa ao playbook qual serviço existente modificar ou adicionar a.

Produtos Agrupados

Produtos agrupados são provisionados como complementos, mas seu playbook pode criar múltiplos registros de serviço. Eles recebem as mesmas variáveis que complementos, incluindo:

- `product_id` - O produto agrupado
- `customer_id` - O cliente
- `service_id` - Serviço pai (se aplicável)
- IDs de itens de inventário (por exemplo, `Modem Router`) se necessário
- `access_token` - Para autenticação da API

O playbook de agrupamento (por exemplo, `play_seniors_package`) então cria múltiplos serviços relacionados (Internet, IPTV, Telefone) e os vincula.

Serviços

Um serviço é uma instância de um produto que pertence a um cliente, pelo qual eles são cobrados.

É essencialmente um link para uma conta `OCS` (Online Charging System) que lida com a geração de cobranças e os saldos e usos reais da conta. O OCS é alimentado pelo CGRateS e gerencia saldos monetários, saldos unitários (dados, voz, SMS), ActionPlans para renovação automática e ThresholdS para limites de gastos.

Adicionando um Serviço: Seleção e Filtragem de Produto

Ao adicionar um serviço a um cliente (seja um novo serviço autônomo ou um complemento a um serviço existente), o sistema exibe produtos disponíveis em

uma interface de carrossel. Os produtos mostrados são filtrados com base em vários critérios:

Filtragem de Produto para Serviços Autônomos

Ao criar um novo serviço para um cliente, a interface do usuário exibe produtos filtrados por:

1. **Tipo de Cliente** - Os produtos são categorizados como:
   - **Individual (Residencial)**: Produtos onde `residential = true` ou `business = false`
   - **Empresarial**: Produtos onde `business = true`
2. **Categoria** - Os produtos são separados em:
   - **Planos de Serviço**: Produtos com `category` = `standalone` ou `bundle`
   - **Complementos**: Produtos com `category` = `addon` (exibidos em carrossel separado)
3. **Disponibilidade** - Os produtos são mostrados apenas se:
   - `enabled = true` - O produto está ativo e não desativado
   - A data atual está entre `available_from` e `available_until` - O produto está dentro de sua janela de disponibilidade
   - `customer_can_purchase = true` (se o cliente estiver se auto-adquirindo) - O produto permite a compra direta pelo cliente

Nota

Filtragem em Nível de API: A API filtra automaticamente produtos por status habilitado e datas de disponibilidade em dois níveis:

- **Endpoints de Compra/Seleção** (`/crm/product/`) - Usado pelo modal de Complementos e PlanList para seleção de produtos. Filtra automaticamente para mostrar APENAS produtos habilitados dentro de seu intervalo de datas de disponibilidade. Isso garante que clientes e funcionários só possam selecionar produtos que estão atualmente disponíveis para compra.
- **Endpoints de Gerenciamento** (`/crm/product/paginated`) - Usado pela página de Gerenciamento de Produtos. Mostra TODOS os produtos, incluindo desativados e fora das datas de disponibilidade, permitindo que

administradores gerenciem o catálogo completo de produtos, incluindo produtos inativos.

Passe `include_disabled=true` para o endpoint base de produtos para ignorar a filtragem (apenas para uso administrativo).
A interface do usuário exibe carrosséis separados para:

- **Planos de Serviço Individuais** - Produtos residenciais para clientes consumidores
- **Planos de Serviço Empresariais** - Produtos comerciais para clientes empresariais
- **Complementos Individuais** - Pacotes de complementos residenciais
- **Complementos Empresariais** - Pacotes de complementos comerciais

Filtragem de Produto para Serviços Adicionais

Ao adicionar um complemento a um **serviço existente**, filtragens adicionais são aplicadas:

1. **Correspondência de Tipo de Serviço** - Apenas complementos com `service_type` correspondente são mostrados:
   - Se o serviço existente tem `service_type = "mobile"`, apenas complementos com `service_type = "mobile"` são exibidos
   - Isso garante que clientes móveis vejam apenas complementos móveis, clientes de internet vejam apenas complementos de internet, etc.
2. **Verificação de Dependências** - Se um complemento tiver uma `relies_on_list`:
   - O sistema verifica se o cliente possui os produtos/serviços necessários
   - Apenas complementos cujas dependências estão satisfeitas são mostrados
3. **Filtro de Mesmo Tipo de Cliente** - Os complementos ainda são filtrados por `residential` vs `business` para corresponder ao tipo de cliente

Exemplo de Cenário de Filtragem

Para um cliente empresarial com um serviço móvel existente (`service_type = "mobile"`):

- **Produtos Autônomos Exibidos**: Todos os produtos autônomos/agrupados empresariais (`business = true`, `category != "addon"`)
- **Produtos Adicionais Exibidos**: Apenas complementos móveis empresariais (`business = true`, `category = "addon"`, `service_type = "mobile"`)
- **Produtos Ocultos**: Produtos residenciais, complementos para outros tipos de serviços (internet, voz, etc.), produtos desativados

Campos de Serviço

O modelo de Serviço contém campos que rastreiam a instância de serviço provisionada e sua relação com o cliente, produto e sistema de faturamento.

Informações Básicas do Serviço

- **service_id** - Identificador único atribuído automaticamente pelo sistema (somente leitura)
- **customer_id** - Link para o cliente que possui este serviço (somente leitura após a criação)
- **product_id** - Link para o produto do qual este serviço foi criado (somente leitura após a criação)
- **service_name** - Nome exibido mostrado aos clientes (editável)
- **service_type** - Tipo de serviço: móvel, internet, voip, iptv, pacote, etc. (editável)
- **service_uuid** - Identificador único usado no OCS/CGRateS para faturamento (somente leitura, gerado automaticamente)
- **icon** - Classe de ícone FontAwesome para exibição no portal de autoatendimento (editável)

Status e Datas do Serviço

- **service_status** - Status atual: Ativo, Inativo, Suspenso, etc. (editável)
- **service_provisioned_date** - Quando o serviço foi provisionado pela primeira vez (definido automaticamente, somente leitura)
- **service_active_date** - Quando o serviço se tornou ativo (editável)

- **service_deactivate_date** - Quando o serviço expira ou será desativado (editável)
- **contract_end_date** - Data de término do compromisso do contrato (editável)

Faturamento e Preços

- **retail_cost** - Cobrança mensal recorrente ao cliente (editável)
- **wholesale_cost** - Seu custo para fornecer o serviço (editável)
- **service_billed** - Se este serviço aparece nas faturas (editável, padrão: verdadeiro)
- **service_taxable** - Se os impostos se aplicam a este serviço (editável, padrão: verdadeiro)
- **invoiced** - Se o serviço foi faturado (definido automaticamente pelo sistema de faturamento)
- **promo_code** - Código promocional usado quando o serviço foi criado (editável)

Visibilidade do Cliente

- **service_visible_to_customer** - Se o cliente pode ver este serviço no portal de autoatendimento (editável, padrão: verdadeiro)
- **service_usage_visible_to_customer** - Se o cliente pode visualizar detalhes de uso/saldo (editável, padrão: verdadeiro)

Configuração de Provisionamento

- **provisioning_play** - Playbook Ansible usado para provisionar este serviço (herdado do produto, somente leitura)
- **provisioning_json_vars** - Variáveis passadas para o playbook de provisionamento (herdadas do produto, somente leitura)
- **deprovisioning_play** - Playbook Ansible a ser executado quando o serviço é desprovisionado (somente leitura)
- **deprovisioning_json_vars** - Variáveis para o playbook de desprovisionamento (somente leitura)

Relações de Serviço

- **bundled_parent** - Se este serviço faz parte de um pacote, o service_id do serviço pai (somente leitura)
- **site_id** - Link para o local físico onde o serviço é entregue (editável)

Notas e Metadados

- **service_notes** - Notas internas sobre o serviço para referência dos funcionários (editável)
- **created** - Timestamp quando o serviço foi criado (definido automaticamente, somente leitura)
- **last_modified** - Timestamp da última atualização (atualizado automaticamente, somente leitura)

Campos Editáveis vs Somente Leitura

Editáveis via API/UI:

Os serviços podem ser atualizados via `PATCH /crm/service/{service_id}` com esses campos:

- 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

Somente Leitura (Definido Automaticamente):

Esses campos não podem ser modificados diretamente após a criação:

- service_id, customer_id, product_id
- service_uuid (gerado durante o provisionamento)
- service_provisioned_date
- provisioning_play, provisioning_json_vars
- deprovisioning_play, deprovisioning_json_vars

- bundled_parent
- invoiced (gerenciado pelo sistema de faturamento)
- created, last_modified (gerenciado automaticamente)

Provisionando Produtos em Serviços

O processo de provisionamento converte um Produto (modelo) em um Serviço (instância específica do cliente) através de uma série de etapas coordenadas envolvendo a interface Web, API e playbooks Ansible.

Fluxo de Provisionamento de Alto Nível

1. **Configuração de Pré-Provisionamento** - Produto criado na API com configuração de provisionamento, e playbooks Ansible correspondentes escritos e testados
2. **Seleção de Serviço** - A partir da Página do Cliente, funcionários ou clientes selecionam "Adicionar Serviço"
3. **Filtragem de Produto** - Produtos exibidos filtrados com base em:
   - Tipo de cliente (residencial/empresarial)
   - Serviços existentes (para dependências de complementos em `relies_on_list`)
   - Datas de disponibilidade (`available_from`/`available_until`)
   - Flags `enabled` e `customer_can_purchase`
4. **Personalização** - Opção para substituir variáveis de provisionamento (para ajustes de preço, configurações personalizadas, etc.)
5. **Seleção de Inventário** - Se o produto requer inventário (`inventory_items_list` não está vazio), o usuário seleciona itens específicos (por exemplo, qual cartão SIM, qual número de telefone)
6. **Início do Provisionamento** - Quando o botão "Provisionar" é clicado, a API cria um trabalho de provisionamento

Fluxo Detalhado de Integração da API e Ansible

Quando um serviço é provisionado, a seguinte sequência ocorre:

Passo 1: Criação do Trabalho de Provisionamento

A API recebe a solicitação de provisionamento e cria um trabalho de provisionamento com:

- `provisioning_play` - Nome do playbook Ansible (por exemplo, `play_psim_only`)
- `provisioning_json_vars` - String JSON de variáveis do produto ou substituídas pela solicitação
- `customer_id` - ID do cliente que solicita o serviço
- `product_id` - ID do produto sendo provisionado
- `service_id` - (Opcional) ID do serviço existente para complementos
- Seleções de Inventário - IDs dos itens de inventário selecionados

Passo 2: Montagem de Variáveis

O serviço de provisionamento mescla variáveis de várias fontes nesta ordem:

1. `provisioning_json_vars` do Produto (padrões da definição do produto)
2. Parâmetros do corpo da solicitação (podem substituir os padrões do produto)
3. Variáveis adicionadas pelo Sistema:
   - `access_token` - Token JWT para autenticação da API de volta ao CRM
   - `initiating_user` - ID do usuário que acionou o provisionamento
   - `customer_id`, `product_id`, `service_id`
4. Seleções de Inventário - Adicionadas como pares `{inventory_type: inventory_id}`

Exemplo de variáveis mescladas:

```
{
    "customer_id": 123,
    "product_id": 456,
    "service_id": 789,          # Apenas para complementos
    "SIM Card": 1001,           # Da seleção de inventário
    "Mobile Number": 1002,      # Da seleção de inventário
    "monthly_cost": 30,         # Da provisioning_json_vars
    "data_gb": 50,              # Da provisioning_json_vars
    "access_token": "eyJ...",   # Adicionado pelo sistema
    "initiating_user": 5        # Adicionado pelo sistema
}
```

Passo 3: Criação do Registro de Provisionamento

Um registro de `Provision` é criado no banco de dados com:

- `provision_id` - Identificador único para rastreamento
- `provisioning_play` - Nome do arquivo do playbook
- `provisioning_json_vars` - Variáveis mescladas como string JSON
- `task_count` - Número de tarefas no playbook (extraído do YAML)
- `provisioning_status` - Código de status (inicialmente definido como 1 = em execução, depois atualizado para 0 = sucesso, 2 = falhou, ou pode permanecer 1 se ainda estiver em progresso)
- `product_id`, `customer_id`, `service_id` - Referências de contexto

Passo 4: Execução do Playbook em Segundo Plano

A API gera uma thread em segundo plano que:

1. Carrega o YAML do playbook de `OmniCRM-API/Provisioners/plays/{playbook_name}.yaml`
2. Chama `ansible_runner.run()` com:
    - `playbook` - Caminho para o arquivo YAML carregado
    - `extravars` - Todas as variáveis mescladas (passadas para o Ansible)
    - `inventory` - Definido como `'localhost,'` (execução local)
    - `event_handler` - Manipulador personalizado para capturar eventos de execução de tarefas
3. Monitora a execução do playbook em tempo real

Passo 5: Captura de Eventos e Registro (`ProvisioningEventHandler`)

À medida que cada tarefa Ansible é executada, eventos são capturados e armazenados como registros de `Provision_Event`:

- `event_name` - Nome da tarefa do playbook
- `event_number` - Número de sequência
- `provisioning_status` - Código de status indicando o resultado da tarefa:
  - **0** = Sucesso - Tarefa concluída com sucesso
  - **1** = Em Execução - Tarefa está atualmente executando
  - **2** = Falhou - Falha crítica que interrompe o provisionamento
  - **3** = Falhou (ignorado) - Tarefa falhou, mas erros foram ignorados (`ignore_errors: true` no playbook)
- `provisioning_result_json` - Resultados da tarefa com dados sensíveis redigidos

O manipulador de eventos automaticamente remove senhas, chaves, segredos e outros dados sensíveis dos registros.

Passo 6: Execução do Playbook Ansible (`Provisioners/plays/*.yaml`)

O playbook Ansible é executado localmente e geralmente realiza as seguintes ações:

1. **Buscar Definição do Produto** - Solicitação GET para `/crm/product/product_id/{{ product_id }}` usando `{{ access_token }}`

2. **Buscar Informações do Cliente** - Solicitação GET para `/crm/customer/customer_id/{{ customer_id }}`

3. **Processar Itens de Inventário** (se necessário) - Solicitação GET para `/crm/inventory/inventory_id/{{ inventory_id }}` para cada item selecionado para recuperar detalhes completos (ICCID, MSISDN, números de série, etc.)

4. **Configurar Sistemas Externos** - Fazer chamadas de API para:

   - HSS (Home Subscriber Server) para provisionamento de assinantes
   - IMS (IP Multimedia Subsystem) para registro de voz

- CGRateS/OCS para criação de conta, configuração de cobrança, planos de tarifas
- Servidores ENUM para mapeamento de números de telefone
- Equipamentos de rede (roteadores, switches, etc.)

5. **Adicionar Custos de Configuração** (se aplicável) - POST para `/crm/transaction/` para registrar cobranças únicas

6. **Cobrar o Cliente** - POST para OCS/CGRateS para cobrar retail_setup_cost se configurado

7. **Criar Conta OCS** - POST para OCS/CGRateS para criar conta de faturamento com UUID

8. **Configurar Cobranças Recorrentes** - Criar Ações e ActionPlans no OCS/CGRateS para cobranças mensais recorrentes

9. **Criar Registro de Serviço** - PUT/POST para `/crm/service/` para criar o registro de serviço no 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. **Atribuir Inventário** - PATCH para `/crm/inventory/inventory_id/{{ inventory_id }}` para marcar o inventário como "Atribuído" ao serviço

11. **Enviar Notificações** (opcional) - E-mail ou SMS para o cliente com detalhes do serviço

Passo 7: Conclusão e Atualização de Status

Quando o playbook é concluído:

- **Sucesso**: `Provision.provisioning_status` atualizado para **0** (Sucesso)
- **Falha Crítica**: `Provision.provisioning_status` atualizado para **2** (Falhou), e e-mail de falha enviado para `crm_config.provisioning.failure_list`
- **Falhas Não Críticas**: Tarefas que falham com `ignore_errors: true` são marcadas com status **3** (Falhou, mas ignorado) e não interrompem o provisionamento

O serviço provisionado agora é visível no CRM e ativo para o cliente (se o provisionamento foi bem-sucedido).

Diferenças Chave: Provisionamento de Produtos Autônomos vs Adicionais vs Agrupados

Produtos Autônomos (`category: standalone`):

- Recebem `customer_id` e `product_id`
- Tipicamente requerem itens de inventário (cartões SIM, números de telefone, modems)
- Criam um registro de serviço **novo** via API PUT `/crm/service/`
- Provisionam novos recursos em sistemas externos (HSS, OCS, equipamentos de rede)
- Exemplo: Ativação de novo SIM móvel, nova conexão de internet

Produtos Adicionais (`category: addon`):

- Recebem `customer_id`, `product_id` e **`service_id`** (serviço existente a ser modificado)
- Tipicamente NÃO requerem inventário (ou inventário mínimo)
- **Modificam um serviço existente** ou adicionam cobranças à conta OCS existente
- Podem executar ações no OCS (adicionar pacote de dados, adicionar crédito, habilitar recurso)

- Não criam novos registros de serviço (ou criam registros de serviço filho vinculados ao pai)
- Exemplo: Recarga de plano de dados mensal, pacote de roaming internacional, crédito extra

Produtos Agrupados (`category: bundle`):

- Semelhante a complementos em termos de variáveis recebidas
- Podem exigir alguns itens de inventário (por exemplo, modem para pacote residencial)
- Criam **múltiplos** registros de serviço relacionados (Internet + TV + Telefone)
- Provisionam múltiplos recursos em diferentes sistemas
- Vinculam serviços no CRM para faturamento/gerenciamento unificado
- Exemplo: Pacote residencial (Internet + IPTV + telefone VoIP)

Requisitos do Playbook de Provisionamento

Para que um playbook funcione corretamente, ele deve:

1. **Estar localizado em** `OmniCRM-API/Provisioners/plays/{playbook_name}.yaml`
2. **Aceitar variáveis** via `extravars` do Ansible (acessadas como `{{ variable_name }}`)
3. **Autenticar chamadas de API** usando `Authorization: Bearer {{ access_token }}` header
4. **Lidar com falhas de forma graciosa** usando blocos `rescue` e `ignore_errors` onde apropriado
5. **Criar registro de serviço** para produtos autônomos, ou modificar serviço existente para complementos
6. **Atribuir inventário** se itens de inventário foram selecionados
7. **Retornar mensagens de erro significativas** via módulo `fail` quando erros críticos ocorrem

Variáveis Comuns Disponíveis em Playbooks

Todo playbook recebe essas variáveis:

- `customer_id` - Inteiro, cliente que solicita o serviço
- `product_id` - Inteiro, produto sendo provisionado
- `service_id` - Inteiro (apenas complementos/agrupamentos), serviço existente a modificar
- `access_token` - String, token JWT para autenticação da API do CRM
- `initiating_user` - Inteiro, usuário que acionou o provisionamento
- Além de quaisquer IDs de itens de inventário: `{{ inventory_type }}: inventory_id`
- Além de quaisquer variáveis de `provisioning_json_vars`
- Além de quaisquer variáveis passadas na solicitação de provisionamento

Os playbooks podem usar essas para:

- Buscar detalhes completos do produto: `GET /crm/product/product_id/{{ product_id }}`
- Buscar detalhes do cliente: `GET /crm/customer/customer_id/{{ customer_id }}`
- Buscar detalhes do inventário: `GET /crm/inventory/inventory_id/{{ SIM_Card }}`
- Criar transações: `POST /crm/transaction/`
- Criar serviços: `PUT /crm/service/`
- Atualizar serviços: `PATCH /crm/service/{{ service_id }}`
- Atribuir inventário: `PATCH /crm/inventory/inventory_id/{{ inventory_id }}`

Exemplo: Fluxo de Playbook de Complemento Simples

Para um complemento de recarga de dados móveis:

1. O playbook recebe: `customer_id`, `product_id`, `service_id`, `access_token`

2. Buscar detalhes do serviço: `GET /crm/service/{{ service_id }}` para obter `service_uuid`
3. Buscar detalhes do produto: `GET /crm/product/product_id/{{ product_id }}` para obter preços e quantidade de dados
4. Cobrar o cliente no OCS: `POST para CGRateS` para deduzir retail_cost do saldo
5. Adicionar crédito de dados no OCS: `POST para CGRateS` para adicionar saldo de dados com expiração
6. Registrar transação no CRM: `POST /crm/transaction/` com detalhes da cobrança
7. Completar com sucesso

Todo o processo é rastreado nas tabelas `Provision` e `Provision_Event` para fins de depuração e auditoria.

Envolvimento do OCS

`OCS` (Online Charging System), implementado via CGRateS, lida com todas as cobranças em tempo real e rastreamento de uso para serviços. O registro de serviço do CRM atua como um ponteiro para a conta OCS, que gerencia:

- **Cobranças recorrentes** - Taxas mensais, aluguel de DID, cobranças de assinatura
- **Cobrança baseada em uso** - Chamadas de voz por minuto, dados por MB, cobranças por SMS
- **Gerenciamento de saldo** - Saldos monetários (crédito pré-pago) e saldos unitários (dados GB, minutos de voz, contagem de SMS)
- **Conversões de saldo** - Convertendo saldos monetários em saldos unitários (por exemplo, gastando $30 para obter pacote de dados de 10GB)
- **Estado da conta** - Ativo, suspenso, desativado com base em limites de crédito e thresholds

O registro de serviço do CRM contém metadados e configuração (cliente, produto, preços, visibilidade), enquanto o OCS contém o estado de faturamento ao vivo (saldos, uso, cobranças).

Recuperando Uso e Saldos de Serviço

As informações de uso do serviço são recuperadas do OCS/CGRateS e exibidas para clientes e funcionários em tempo real.

Como o Uso é Recuperado

Quando o uso de um serviço é solicitado (via UI ou API), o seguinte fluxo ocorre:

1. **Solicitação API** - O frontend chama `GET /crm/service/{service_id}` ou visualiza detalhes do serviço na interface do usuário

2. **Busca do Serviço** - A API recupera o registro do serviço do banco de dados, extrai `service_uuid`

3. **Chamadas da API CGRateS** - O sistema faz duas chamadas para CGRateS:

    i. **Get_Balance(service_uuid)** - Recupera saldo da conta com `BalanceMap`
        - Retorna saldos organizados por tipo: DADOS, VOZ, SMS, MONETÁRIO, DADOS_DONGLE
        - Cada saldo inclui: ID, Valor, Data de Expiração, Peso, DestinationIDs
        - O sistema adiciona campos legíveis por humanos: `custom_Name_hr`, `custom_Expiration`, `custom_Description_String`

    ii. **Get_ActionPlans(service_uuid)** - Recupera planos de ação de renovação automática ativos (cobertos na próxima seção)

4. **Mesclagem de Resposta** - Os dados do CGRateS são mesclados na resposta do serviço:

```
{
  "service_id": 123,
  "service_name": "Serviço Móvel",
  "service_uuid": "abc-123-def",
  "cgrates": {
    "BalanceMap": {
      "DATA": [{
        "ID": "DATA_10GB",
        "Value": 5368709120,
        "ExpirationDate": "2025-02-01T00:00:00Z",
        "custom_Name_hr": "Pacote de Dados de 10GB",
        "custom_Expiration": "1 de Fevereiro de 2025",
        "custom_Description_String": "5 GB restantes"
      }],
      "VOICE": [{
        "ID": "VOICE_UNLIMITED",
        "Value": 999999999,
        "custom_Name_hr": "Chamadas Ilimitadas",
        "custom_Description_String": "Minutos ilimitados"
      }],
      "MONETARY": [{
        "ID": "PREPAID_CREDIT",
        "Value": 25.50,
        "custom_Description_String": "Crédito de \$25.50"
      }]
    },
    "ActionPlans": [...]
  }
}
```


5. **Exibição da UI** - Componentes do frontend exibem os dados de uso:

   - **ServiceUsage.js** - Componente principal de exibição de uso com atualização automática a cada 3 segundos
   - **UsageCard.js** - Cartões de resumo para cada tipo de saldo
   - **UsageProgress.js** - Barras de progresso mostrando porcentagem usada/restante
   - Saldos são codificados por cores e formatados para legibilidade

Estrutura de Dados de Uso

Cada saldo no `BalanceMap` contém:

Campos Nativos do CGRateS:

- `ID` - Identificador único para o saldo (por exemplo, "DATA_10GB_2025_01")
- `Value` - Montante do saldo:
  - Para DADOS: bytes (5368709120 = 5 GB)
  - Para VOZ: segundos (3600 = 1 hora)
  - Para SMS: contagem (100 = 100 mensagens)
  - Para MONETÁRIO: unidades de moeda (25.50 = $25.50)
- `ExpirationDate` - Timestamp ISO 8601 quando o saldo expira
- `Weight` - Prioridade para consumo de saldo (peso maior consumido primeiro)
- `DestinationIDs` - Destinos aos quais este saldo se aplica (por exemplo, ["AU", "INTERNATIONAL"])

Campos Legíveis por Humanos (adicionados pelo CRM):

- `custom_Name_hr` - Nome legível por humanos extraído do ID
- `custom_Expiration` - Data de expiração formatada (por exemplo, "15 de Janeiro de 2025" ou "em 11 dias")
- `custom_Description_String` - Descrição do saldo legível por humanos:
  - DADOS: "5 GB restantes" ou "10 GB totais"
  - VOZ: "60 minutos restantes" ou "Ilimitado"
  - SMS: "50 SMS restantes"
  - MONETÁRIO: "$25.50 de crédito"

Controle de Visibilidade de Uso

A visibilidade do uso do serviço é controlada por dois campos:

- **service_visible_to_customer** - Se falso, o serviço é ocultado completamente do portal de autoatendimento do cliente
- **service_usage_visible_to_customer** - Se falso, o serviço é visível mas os detalhes de uso/saldo estão ocultos (o cliente pode ver que possui o

serviço, mas não quanto usou)

Isso permite que os operadores:

- Oculte serviços internos/testes dos clientes
- Mostre que o serviço existe sem revelar o uso (útil para planos ilimitados ou serviços sensíveis)
- Exibição totalmente transparente de uso (padrão)

Atualizações de Uso em Tempo Real

A interface Web atualiza automaticamente os dados de uso:

- **Intervalo**: A cada 3 segundos (configurável no componente ServiceUsage)
- **Método**: Faz polling de `GET /crm/service/{service_id}` que busca dados ao vivo do CGRateS
- **Eficiência**: Apenas visualizações de serviços ativos são atualizadas; visualizações de lista usam dados em cache

Isso garante que clientes e funcionários vejam atualizações de saldo em quase tempo real à medida que o uso ocorre.

Cobranças Recorrentes / AutoRenovação

Cobranças recorrentes, como uma Cobrança Mensal de Serviço ou uma Cobrança por DID, são primeiro criadas como Ações dentro do `OCS` e seguem o formato `Action_ServiceUUID_ServiceName_WhatitDoes`.

Por exemplo, para um serviço GPON de $60 por mês que inclui 1TB de uso, a Ação seria algo como:

`Action_kj49-adsf-1234-9742_60_GPON_1TB_MonthlyExpiry`

1. Redefinir Saldo Monetário para $0
2. Enviar um POST HTTP para `/simple_provision` no CRM para provisionar algo

3. Adicionar um Crédito para 1TB de Uso expirando em 1 mês

Se quisermos que a MRC seja recorrente (queremos), então criaríamos um `ActionPlan` chamado `ActionPlan_{{ service_uuid }}_Monthly_Charge` que teria o tempo definido para *mensal* para acionar todo mês, e atribuiríamos o `ActionPlan` à conta.

Podemos definir com base no parâmetro *Ano / Meses / Dias* uma data de expiração para quando a MRC também parará, por exemplo, para um serviço fixo de 12 meses que parou após esse ponto.

Como as Ações e `ActionPlans` são únicas para o serviço, elas não compartilham nada com nenhum outro serviço.

Isso significa que podemos provisioná-las com valores ajustados, e isso não impactará nenhum outro serviço.

Complementos e Adicionais

Complementos / Adicionais como comprar dados extras, pacotes de roaming, minutos internacionais, etc., são tratados de forma muito semelhante. Uma Ação é criada para fazer o que é necessário, como cobrar um valor monetário e depois conceder um saldo unitário com uma expiração definida.

Em vez de usar `ActionPlans` para adicionar isso automaticamente ao ocorrer na conta, apenas acionamos `ExecuteAction` para a Ação que acabamos de criar uma única vez a partir do Ansible.

Adicionando Saldos Monetários Pré-pagos

Para saldos monetários pré-pagos, como um plano PAYG de $10, isso é adicionado como um saldo monetário, mas com uma prioridade maior.

O limite de crédito nesses serviços para o saldo padrão seria $0.

Limites de Crédito / Prevenindo Gastos Excessivos

`ThresholdS` são usados em cada conta para definir o gasto máximo para um dado período de tempo.

Para clientes PAYG / Pré-pagos, isso é $0.

Interagindo com OCS via CRM

Para cada Serviço, você pode ver os `Balances` e `ActionPlans`, `Actions` e `ThresholdS` do `OCS` a partir da API do CRM.

`ActionPlans` podem ser removidos conforme necessário da API do CRM, acionados via Playbooks Ansible. `ActionPlans` podem ser adicionados conforme necessário, a partir do CRM, adicionando um Complemento/Serviço e acionados via Playbooks Ansible.

Contas `OCS` podem ser desativadas, o que interromperá a execução de `ActionPlans` e a capacidade de consumir serviços.

Para Limites de Crédito, um valor de `ThresholdS` é definido conforme a política para o produto.

Visualizando e Gerenciando ActionPlans no CRM

ActionPlans (configurações de renovação automática) são exibidos e gerenciados através da interface do CRM, permitindo que funcionários e clientes vejam renovações automáticas futuras e as gerenciem.

Como ActionPlans são Recuperados e Exibidos

Ao visualizar um serviço no CRM, ActionPlans são automaticamente buscados e exibidos:

1. **Chamada API** - Quando `GET /crm/service/{service_id}` é chamado, a API:
   - Recupera o registro do serviço do banco de dados
   - Extrai o `service_uuid` (identificador da conta OCS)
   - Chama a API CGRateS para buscar ActionPlans por UUID do serviço
   - Isso chama internamente `Get_ActionPlans(service_uuid)` no CGRateS
2. **Estrutura de Dados do ActionPlan** - Cada ActionPlan retornado contém:

```
{
    "ActionPlanId": "ServiceID_abc-123-
def__ProductID_456__...",
    "PlanName": "Plano de Renovação Mensal",
    "NextExecTime": "2025-02-01T00:00:00Z",
    "custom_NextExecTime_hr": "em 11 dias",
    "ActionPlanId_split_dict": {
        "ServiceID": "abc-123-def",
        "ProductID": 456,
        "CustomerID": 789,
        ...
    }
}
```

   - `ActionPlanId` - Identificador único contendo informações codificadas de serviço/produto/cliente
   - `PlanName` - Nome do plano de ação (tipicamente o nome do playbook de renovação)
   - `NextExecTime` - Timestamp ISO quando o ActionPlan será executado a seguir
   - `custom_NextExecTime_hr` - Tempo legível por humanos até a execução (por exemplo, "em 11 dias", "amanhã", "1 de Fevereiro de 2025")
   - `ActionPlanId_split_dict` - Dicionário com componentes analisados do ActionPlanId
3. **Exibição da UI** - ActionPlans são mostrados no componente **ActionPlansTable**:

Colunas da Tabela:

- **Nome do Produto** - Recuperado buscando `ProductID` do ActionPlanId
- **Custo** - Mostra `retail_cost` da definição do produto
- **Data de Renovação** - Exibe `custom_NextExecTime_hr` (legível por humanos)
- **Ações** - Dois botões:
  - **Renovar Agora** - Provisionar imediatamente o complemento/renovação (ignora a espera pela execução programada)
  - **Remover Renovação Automática** - Cancela a renovação automática

Quando Nenhum ActionPlan Existe:

- A tabela mostra a mensagem: "Nenhuma renovação automática habilitada para este serviço"
- O cliente pode adicionar complementos de renovação automática para habilitar a renovação automática

Gerenciando ActionPlans

Funcionários e clientes podem gerenciar ActionPlans através da interface:

Removendo um ActionPlan (Cancelando Renovação Automática):

1. Clique no botão "Remover Renovação Automática" na ActionPlansTable
2. Modal de confirmação aparece: "Você tem certeza que deseja remover esta renovação automática?"
3. Na confirmação, o frontend chama: `DELETE /crm/oam/remove_action_plan/{action_plan_id}`
4. A API remove o ActionPlan do CGRateS via `ocs.Remove_ActionPlan()`
5. A atividade é registrada: "Removed ActionPlan {ActionPlanId} from service {service_id}"
6. ActionPlan desaparece da tabela

Renovando Imediatamente (Renovação Manual):

1. Clique no botão "Renovar Agora" na ActionPlansTable
2. Modal de confirmação aparece: "Você tem certeza que deseja renovar isso agora?"
3. Na confirmação, o sistema:
    - Extrai product_id do ActionPlanId
    - Cria um novo trabalho de provisionamento para esse produto
    - Provisiona o complemento imediatamente (executa o playbook de provisionamento)
    - O serviço recebe os benefícios do complemento sem esperar pela renovação programada
4. Modal de status de provisionamento mostra progresso
5. Em caso de sucesso, saldos são atualizados imediatamente

Adicionando Renovação Automática:

A renovação automática é habilitada ao provisionar um produto complementar que tem `auto_renew` definido:

- Produtos com `auto_renew = "true"` - Criam automaticamente ActionPlans durante o provisionamento
- Produtos com `auto_renew = "prompt"` - Perguntam ao cliente se ele deseja renovação automática (diálogo modal)
- Produtos com `auto_renew = "false"` - Nunca criam ActionPlans (compra única)

O playbook de provisionamento cria o ActionPlan no CGRateS com:

- ActionPlanId único codificando IDs de serviço, produto e cliente
- Cronograma de renovação (mensal, anual, intervalo personalizado)
- Ação a ser executada (tipicamente reprovisionamento do mesmo complemento)
- Data de expiração (se o contrato tiver um prazo fixo)

Convenção de Nomenclatura de ActionPlan

ActionPlans seguem uma convenção de nomenclatura padronizada para codificar metadados:

Formato:

```
ServiceID_{service_uuid}__ProductID_{product_id}__CustomerID_{custome
```

Exemplo:

```
ServiceID_abc-123-
def__ProductID_456__CustomerID_789__MonthlyRenewal
```

Essa codificação permite que o CRM:

- Identifique a qual serviço o ActionPlan pertence
- Busque detalhes do produto (nome, preços) para exibição
- Rastreie a propriedade do cliente
- Analise o tipo de renovação e cronograma

O CRM analisa automaticamente esses componentes em `ActionPlanId_split_dict` para fácil acesso.

ActionPlans na Visualização de Serviço

Ao visualizar um serviço no CRM, a tabela de ActionPlans é exibida nos detalhes do serviço:

Visualização do Funcionário (`ServiceView.js`):

- Mostra a tabela completa de ActionPlans com todas as opções de gerenciamento
- Exibe nomes de produtos, custos, datas de renovação
- Permite remoção e renovação imediata

Visualização do Cliente no Autoatendimento:

- Mostra renovações futuras em uma visualização simplificada
- Exibe a próxima data de renovação e valor

- Pode permitir que os clientes desativem a renovação automática (configurável por produto)

Estado Vazio:

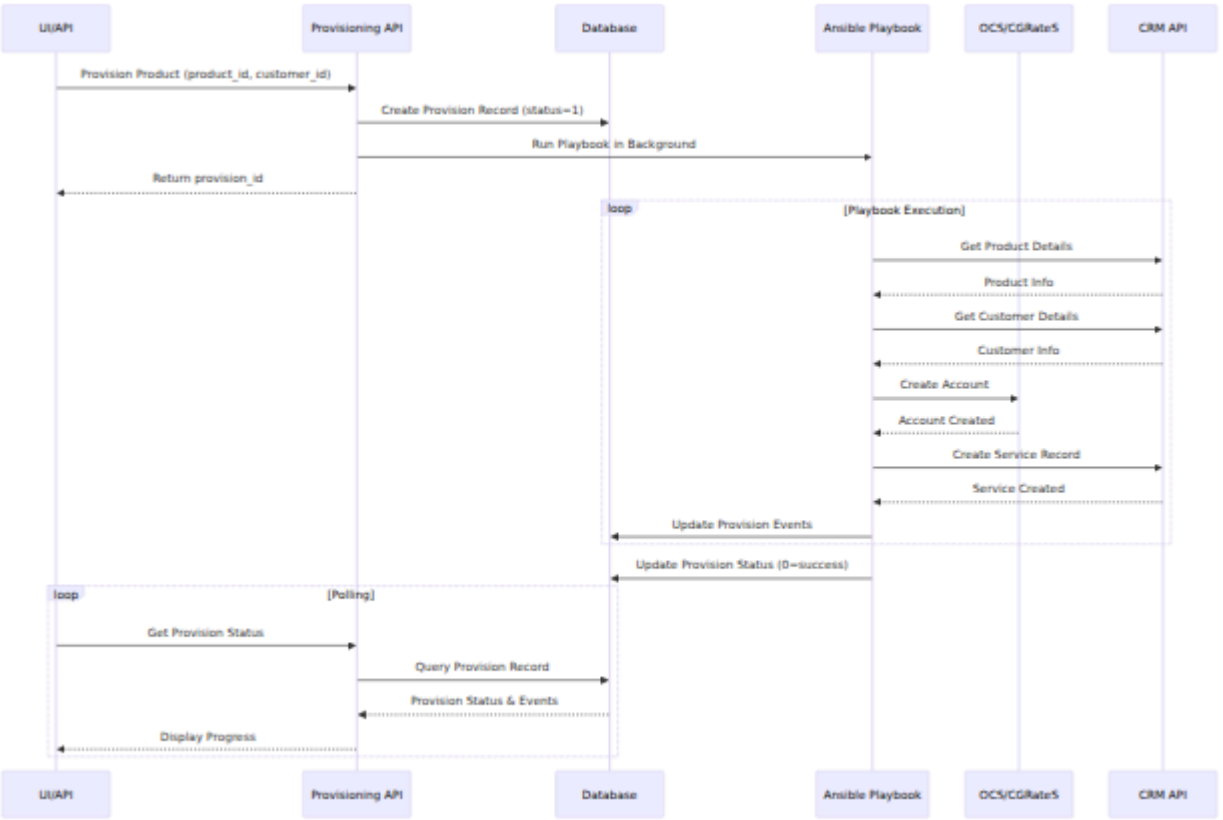
- Se nenhum ActionPlan existir: "Nenhuma renovação automática habilitada para este serviço"
- Sugere adicionar um complemento de renovação automática para habilitar renovações automáticas

ActionPlans

Sistema de Provisionamento

OmniCRM usa **Ansible** para automatizar o provisionamento, configuração e desprovisionamento de serviços ao cliente. O sistema de provisionamento é projetado para ser flexível, permitindo fluxos de trabalho complexos enquanto mantém consistência e confiabilidade.

Veja também: `Provisionamento de Cartão SIM <concepts_sim_provisioning>` para informações detalhadas sobre o provisionamento de SIMs móveis, incluindo tanto SIMs físicos quanto eSIMs.



Nota

Para um guia completo da jornada do produto ao serviço com exemplos detalhados de playbook Ansible, estratégias de preços e cenários do mundo real, veja `Guia Completo do Ciclo de Vida do Produto <guide_product_lifecycle>`.

Visão Geral

Quando um produto é solicitado ou um serviço precisa ser configurado, o OmniCRM cria um **Trabalho de Provisionamento** que executa um ou mais playbooks Ansible. Esses playbooks interagem com vários sistemas de backend (OCS/CGRateS, equipamentos de rede, APIs, etc.) para provisionar completamente o serviço.

O sistema de provisionamento suporta dois fluxos de trabalho principais:

1. **Provisionamento Padrão** - Acionado por funcionários ou clientes através da UI/API
2. **Provisionamento Simples** - Acionado por sistemas externos como OCS para operações automatizadas

Valores de Status de Provisionamento

Os trabalhos de provisionamento e tarefas individuais podem ter os seguintes status:

- **Status 0 (Sucesso)** - O trabalho de provisionamento foi concluído com sucesso
- **Status 1 (Executando)** - O trabalho ou tarefa de provisionamento está atualmente em execução
- **Status 2 (Falhou - Crítico)** - Ocorreu uma falha crítica que causou a falha do provisionamento
- **Status 3 (Falhou - Ignorado)** - Uma tarefa falhou, mas tinha `ignore_errors: true`, então o provisionamento continuou

Quando um trabalho de provisionamento falha, o OmniCRM envia notificações por e-mail para a lista de notificação de falhas configurada com informações detalhadas sobre o erro.

Como os Produtos Impulsionam o Provisionamento

A definição de **Produto** é o modelo do que é provisionado e como. Quando um usuário seleciona um produto para provisionar, o sistema lê vários campos-chave da definição do produto para determinar o que fazer.

Campos do Produto Usados no Provisionamento

Uma definição de produto contém:

- `provisioning_play` - O nome do playbook Ansible a ser executado (sem a extensão .yaml)
- `provisioning_json_vars` - String JSON contendo variáveis padrão a serem passadas para o Ansible

- `inventory_items_list` - Lista de tipos de inventário que devem ser atribuídos (por exemplo, `['SIM Card', 'Mobile Number']`)
- `product_id`, `product_name`, campos de preços - Passados automaticamente para o playbook

Exemplo de Definição de Produto

```
{
  "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
}
```

Do Produto ao Trabalho de Provisionamento

Quando o provisionamento é iniciado, o sistema:

1. **Carrega o playbook** especificado em `provisioning_play`

   O sistema procura por `OmniCRM-API/Provisioners/plays/play_psim_only.yaml`

2. **Mescla variáveis** de múltiplas fontes em `extra_vars`:

   i. **De provisioning_json_vars**: `{"iccid": "", "msisdn": ""}`
   ii. **Do corpo da solicitação**: Quaisquer variáveis adicionais que o usuário/API fornecer
   iii. **Dos campos do produto**: `product_id`, `customer_id`, etc.
   iv. **Da autenticação**: `access_token` ou configuração para `refresh_token`

3. **Atribui inventário** com base em `inventory_items_list`

Antes de executar o playbook, a UI/API solicita a seleção de inventário:

- **SIM Card** - O usuário seleciona um SIM disponível do inventário
- **Mobile Number** - O usuário seleciona um número de telefone disponível

Os IDs de inventário selecionados são adicionados a `extra_vars` com o tipo de inventário como a chave:

```
extra_vars = {
    "product_id": 1,
    "customer_id": 456,
    "SIM Card": 789,         # inventory_id do SIM selecionado
    "Mobile Number": 101,    # inventory_id do número de
telefone selecionado
    "iccid": "",             # De provisioning_json_vars
    "msisdn": "",            # De provisioning_json_vars
    "access_token": "eyJ..."
}
```

4. **Passa tudo para o Ansible** via `hostvars[inventory_hostname]`

Dentro do playbook, as variáveis são acessíveis como:

```
- 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]"
```

Como os Playbooks Usam Variáveis de Inventário

Uma vez que o playbook tem os IDs de inventário, ele recupera os detalhes completos do inventário da API:

```yaml
- name: Get SIM Card Details from Inventory
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ inventory_id_sim_card }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
    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 }}"
```

O playbook pode então usar esses valores para:

- Provisionar o cartão SIM no HSS com o IMSI
- Configurar o número de telefone no sistema de cobrança
- Atribuir os itens de inventário ao cliente
- Criar o registro de serviço com esses detalhes

Exemplo do Mundo Real: Provisionamento de SIM Móvel

Do `play_psim_only.yaml`, veja como ele usa dados de produto e inventário:

```
- 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
```

Isso demonstra o fluxo completo:

1. A definição do produto especifica `provisioning_play: "play_psim_only"`
2. O produto requer `inventory_items_list: ['SIM Card', 'Mobile Number']`
3. O usuário seleciona itens de inventário durante o provisionamento
4. Os IDs de inventário são passados para o playbook como `extra_vars`
5. O playbook recupera os detalhes completos do inventário da API
6. O playbook usa os dados do inventário para configurar equipamentos de rede
7. O playbook marca o inventário como atribuído ao cliente

Rollback e Limpeza: Padrão de Melhores Práticas

Melhor Prática Crítica: O mesmo playbook deve lidar tanto com o rollback de provisionamento falho quanto com o desprovisionamento intencional usando a estrutura `block` e `rescue` do Ansible.

Estrutura do Playbook

Do `play_psim_only.yaml`:

```yaml
- name: OmniCore Service Provisioning 2024
  hosts: localhost
  gather_facts: no
  become: False

  tasks:
    - name: Main block
      block:
        # --- TAREFAS DE PROVISIONAMENTO ---
        - name: Get Product information
          uri: ...

        - name: Create account in OCS
          uri: ...

        - name: Provision subscriber on HSS
          uri: ...

        - name: Create service record
          uri: ...

        # ... muitas mais tarefas de provisionamento ...

      rescue:
        # --- TAREFAS DE LIMPEZA ---
        # Esta seção é executada quando:
        # 1. Qualquer tarefa no bloco falha (rollback)
        # 2. action == "deprovision" (limpeza intencional)

        - 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:] }}",  # Número
fictício
                "ue_ambr_dl": 9999999,              # Altamente
elevado
                "ue_ambr_ul": 9999999
              }
          loop: "{{ hss_subscriber_data | dict2items }}"
          when:
            - deprovision_subscriber | default(false) | bool ==
false

        # A afirmação final determina sucesso ou falha
        - name: Set status to "Success" if Manual deprovision /
Fail if failed provision
          assert:
            that:
              - action == "deprovision"
```

Por que Este Padrão é Melhor Prática

1. Sem Duplicação de Código

As mesmas tarefas de limpeza lidam com ambos os cenários:

- **Falha de Provisionamento (Rollback)**: Se qualquer tarefa no `block` falhar, a seção `rescue` é executada automaticamente

- **Desprovisionamento Intencional**: Quando chamada com `action: "deprovision"`, o playbook salta imediatamente para `rescue`

2. Limpeza Completa Garantida

Quando um provisionamento falha no meio, a seção de resgate garante:

- Todas as contas OCS criadas são deletadas
- Todas as entradas de equipamentos de rede configuradas são removidas
- O inventário atribuído é retornado ao pool
- Os assinantes HSS são deletados ou definidos como inativos
- Nenhum provisionamento parcial permanece em qualquer sistema

Isso previne recursos "órfãos" que:

- Consomem inventário sem serem rastreados
- Criam contas de cobrança que não estão vinculadas a serviços
- Causam confusão durante a solução de problemas
- Desperdiçam recursos de rede

3. Tratamento de Falhas Gracioso com ignore_errors

Observe que cada tarefa de limpeza usa `ignore_errors: True`. Isso é intencional porque:

- Durante o rollback, alguns recursos podem não ter sido criados ainda
- Queremos tentar todas as tarefas de limpeza mesmo que algumas falhem
- A afirmação final determina o sucesso/falha geral

Por exemplo, se o provisionamento falhar em "Criar conta no OCS", a limpeza tentará:

- Deletar a conta OCS (falhará, mas será ignorada)
- Remover perfis de atributos (falhará, mas será ignorada)
- Retornar inventário (sucede)
- Deletar assinante HSS (pode não existir, ignorado)

4. Distinguindo Desprovisionamento de Rollback

A afirmação final no final de `rescue` é inteligente:

```
- name: Set status to "Success" if Manual deprovision / Fail if
failed provision
  assert:
    that:
      - action == "deprovision"
```

Isso significa:

- **Se ``action == "deprovision"``**: A afirmação passa, o playbook tem sucesso (status 0)
- **Se ``action`` não estiver definido ou != "deprovision"**: A afirmação falha, o playbook falha (status 2)

Assim, o mesmo código de limpeza resulta em diferentes status de trabalho de provisionamento, dependendo da intenção.

5. Limpeza Condicional com Base no Tipo de Serviço

Algumas tarefas de limpeza usam condicionais para lidar com diferentes cenários:

```
- 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
```

Isso permite uma limpeza flexível:

- **Deleção completa**: Quando SIMs são dedicados a clientes (`deprovision_subscriber: true`)
- **Estado inativo**: Quando SIMs são reutilizáveis e devem permanecer no HSS (`deprovision_subscriber: false`)

Como Usar Este Padrão

Para Provisionamento:

```
{
  "product_id": 1,
  "customer_id": 456,
  "provisioning_play": "play_psim_only"
}
```

Se o provisionamento falhar, o rollback automático ocorre via `rescue`.

Para Desprovisionamento:

```
{
  "service_id": 123,
  "service_uuid": "Service_abc123",
  "action": "deprovision",
  "provisioning_play": "play_psim_only"
}
```

O playbook salta diretamente para a seção `rescue`, executa toda a limpeza e tem sucesso.

Resumo dos Benefícios

 Fonte única de verdade: Um playbook lida com provisionamento e desprovisionamento **Operações atômicas**: Ou totalmente provisionado ou totalmente limpo **Sem recursos órfãos**: Provisionamentos falhados não deixam vestígios **Manutenção mais fácil**: Alterações na lógica de provisionamento se aplicam automaticamente à limpeza **Redução de erros**: Sem chance de o código de provisionamento e desprovisionamento ficarem fora de sincronia **Testável**: Pode testar a lógica de desprovisionamento executando com `action: "deprovision"`

Este padrão deve ser seguido em todos os playbooks de provisionamento para garantir confiabilidade e consistência.

Sobrescrevendo Variáveis do Produto

As `provisioning_json_vars` podem ser sobrescritas no momento do provisionamento. Por exemplo, um produto pode definir:

```
{
  "provisioning_json_vars": "{\"monthly_cost\": 50,
\"data_limit_gb\": 100}"
}
```

Mas ao provisionar, você pode sobrescrever isso:

```
{
  "product_id": 1,
  "customer_id": 456,
  "monthly_cost": 45,
  "data_limit_gb": 150
}
```

As `extra_vars` mescladas usarão os valores sobrescritos. Isso permite:

- Preços personalizados para clientes específicos
- Limites de dados diferentes com base em promoções
- Testes com diferentes parâmetros sem modificar o produto

Produtos Sem Inventário

Nem todos os produtos requerem inventário. Por exemplo, um complemento de dados ou um recurso pode ter:

```
{
  "product_id": 10,
  "product_name": "Extra 10GB Data",
  "provisioning_play": "play_local_data_addon",
  "provisioning_json_vars": "{\"data_gb\": 10}",
  "inventory_items_list": "[]"
}
```

Nesse caso, o playbook recebe:

```
extra_vars = {
    "product_id": 10,
    "customer_id": 456,
    "service_id": 123,  # Serviço para adicionar dados
    "data_gb": 10,
    "access_token": "eyJ..."
}
```

O playbook simplesmente adiciona os dados ao serviço existente sem precisar de itens de inventário.

Fluxo de Trabalho de Provisionamento Padrão

O provisionamento padrão é iniciado quando:

- Um membro da equipe adiciona um serviço a um cliente pela UI
- Um cliente solicita um serviço através do portal de autoatendimento
- A API é chamada diretamente com `PUT /crm/provision/`

Quando Você Clica em "Provisionar"

Aqui está o fluxo completo que ocorre quando um usuário clica no botão "Provisionar":

1. UI Exibe Seleção de Produto

O usuário seleciona um produto do catálogo de produtos. O produto contém:

- `provisioning_play` - Qual playbook Ansible executar
- `inventory_items_list` - Inventário necessário (por exemplo, `['SIM Card', 'Mobile Number']`)
- `provisioning_json_vars` - Variáveis padrão

2. Seletor de Inventário (Se Necessário)

Se `inventory_items_list` não estiver vazio, um modal aparece mostrando dropdowns para cada tipo de inventário. O usuário deve selecionar itens de inventário disponíveis antes de prosseguir.

3. Botão de Provisionamento Clicado

JavaScript envia a solicitação `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. API Recebe Solicitação

O endpoint de provisionamento:

- Valida a autenticação (token Bearer, chave API ou lista de IPs permitidos)
- Verifica se o usuário tem permissão `CREATE_PROVISION`
- Extrai `initiating_user` do token
- Carrega a definição do produto do banco de dados
- Recupera o caminho do playbook: `OmniCRM-API/Provisioners/plays/play_psim_only.yaml`

5. Variáveis Mescladas

O sistema combina variáveis de múltiplas fontes:

```
# Do produto
product_vars = json.loads(product['provisioning_json_vars'])
# Do corpo da solicitação
request_vars = request_body_json
# Adicionadas pelo sistema
system_vars = {
    'product_id': 42,
    'customer_id': 123,
    'access_token': auth_token,  # Veja a seção de autenticação
abaixo
    'initiating_user': 7
}
# Final mesclado
extra_vars = {**product_vars, **request_vars, **system_vars}
```

6. Registro de Provisionamento Criado

Registro no banco de dados criado com status 1 (Executando):

```
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,  # Executando
    'task_count': 85,
    'initiating_user': 7,
    'created': '2025-01-10T14:30:00Z'
}
```

7. Thread em Segundo Plano Gerada

O sistema gera uma thread em segundo plano para executar o playbook de forma assíncrona:

- Playbook: `play_psim_only.yaml`
- Variáveis: extra_vars mescladas
- ID de Provisionamento: 456 (para rastreamento de status)

- Token de atualização: passado para atualização de token durante a execução

8. API Retorna Imediatamente

Resposta retornada para a UI com `provision_id`:

```
{
  "provision_id": 456,
  "provisioning_status": 1,
  "message": "Provisioning job created"
}
```

9. UI Faz Polling por Atualizações

A UI começa a fazer polling em `GET /crm/provision/provision_id/456` a cada 3 segundos para verificar o status. A resposta inclui:

```
{
  "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 Executa em Segundo Plano

O playbook executa tarefas sequencialmente:

- Cada conclusão de tarefa cria um registro `Provision_Event` no banco de dados
- O evento inclui: nome da tarefa, status (0=sucesso, 2=falhou, 3=falhou mas ignorado), resultado JSON
- A UI exibe o progresso em tempo real mostrando tarefas concluídas e a tarefa atualmente em execução
- Tarefas falhadas mostram mensagens de erro nos detalhes do evento

Rastreamento na UI:

Enquanto o provisionamento está em execução (status 1), os usuários podem visualizar:

- **Página de Detalhes do Serviço** - Mostra o badge de status de provisionamento (Executando/Sucesso/Falhou)
- **Registro de Atividades** - Lista todos os eventos de provisionamento com timestamps
- **Visualização de Detalhes do Provisionamento** - Mostra o progresso tarefa a tarefa com expandir/recolher para detalhes

Exibição de exemplo:

Status de Provisionamento: Executando (8 de 12 tarefas concluídas)

✓ Obter informações do produto da API CRM ✓ Buscar detalhes do cliente ✓ Atribuir cartão SIM do inventário (ICCID: 8991101200003204510) ✓ Atribuir número de telefone (555-0123) ⟳ Criar conta no OCS/CGRateS (em progresso...) □ Configurar políticas de rede □ Criar registro de serviço ...

11. Provisionamento Completo

Status final definido:

- `provisioning_status: 0` - Sucesso
- `provisioning_status: 2` - Falhou (erro crítico)

A UI para de fazer polling e exibe o resultado:

- **Sucesso**: Marca de verificação verde, serviço marcado como Ativo, o usuário pode visualizar os detalhes do serviço
- **Falha**: X vermelho, mensagem de erro exibida, opção de tentar novamente ou entrar em contato com o suporte
- **Notificação por e-mail**: Se falhar, e-mail enviado para `provisioning.failure_list` na configuração

Autenticação e Autorização

Rastreamento de Usuários

Cada trabalho de provisionamento rastreia qual usuário o iniciou:

- **Iniciado pelo usuário**: O campo `initiating_user` é definido como o ID do usuário de seu token JWT
- **Autenticação por chave API**: Usa o ID do primeiro usuário administrador
- **Autenticação por lista de IPs**: Usa o ID do primeiro usuário administrador

Verificações de Permissão

O sistema verifica permissões antes de permitir o provisionamento:

- Funcionários precisam da permissão `CREATE_PROVISION`
- Clientes só podem provisionar serviços para sua própria conta (`VIEW_OWN_PROVISION`)

Como o Ansible se Autentica com a API CRM

Os playbooks Ansible precisam fazer chamadas API autenticadas de volta para o CRM (para buscar detalhes do produto, criar serviços, atualizar inventário, etc.). A autenticação é tratada através de **tokens Bearer** passados para o playbook.

A origem do `access_token` depende do método de autenticação usado para chamar a API de provisionamento:

Método 1: Login do Usuário (Token Bearer)

Quando um usuário faz login via UI web:

1. O usuário se autentica: `POST /crm/auth/login`
2. Recebe `access_token` JWT (de curta duração, 15-30 min) e `refresh_token` (de longa duração)
3. Faz a solicitação de provisionamento com o token Bearer no cabeçalho
4. A API de provisionamento extrai o token do cabeçalho `Authorization: Bearer ...`
5. Armazena o token para uso durante o provisionamento
6. Passa para o Ansible como variável `access_token`

Fluxo de Implementação:

O sistema extrai o token Bearer do cabeçalho de autorização, valida e decodifica, e então o armazena para uso durante o provisionamento. O token é então passado para o playbook como uma variável extra para que possa autenticar chamadas API subsequentes.

Método 2: Chave API (Cabeçalho X-API-KEY)

Para sistemas automatizados que usam chaves API:

1. O sistema faz a solicitação: `PUT /crm/provision/` com cabeçalho `X-API-KEY: sua-chave-api...`
2. A API de provisionamento valida a chave API contra `crm_config.yaml`
3. **Gera um novo token JWT em tempo real** para o primeiro usuário administrador
4. Armazena o token para provisionamento
5. Passa para o Ansible

Por que Gerar um Token?

As chaves API são strings, não JWTs. Os playbooks chamam endpoints API esperando autenticação JWT. Então:

- Valide a chave API
- Se válida e tiver papel de `admin`, gere um JWT temporário
- Use o ID do primeiro usuário administrador como sujeito do JWT

- O token permite que o playbook faça chamadas API autenticadas

Fluxo de Implementação:

O sistema valida a chave API contra as chaves configuradas. Se válida e tiver papel de administrador, gera um JWT temporário usando a identidade de um usuário administrador, e então o armazena para uso durante o provisionamento.

Método 3: Lista de IPs Permitidos

Para sistemas internos confiáveis em redes privadas:

1. O sistema faz a solicitação de um IP permitido (por exemplo, 192.168.1.100)
2. A API de provisionamento verifica o IP do cliente contra `ip_whitelist` em `crm_config.yaml`
3. Se permitido, **gera um novo token JWT** para o primeiro usuário administrador
4. Armazena o token para provisionamento
5. Passa para o Ansible

Fluxo de Implementação:

O sistema verifica o IP do cliente contra a lista de permitidos. Se permitido, gera um JWT para um usuário administrador e o armazena para uso durante o provisionamento.

Usando o Token nos Playbooks

Cada chamada API no playbook inclui o token:

```
- 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"
```

Expiração e Atualização do Token

Playbooks de longa duração (5-10 minutos) podem ultrapassar o `access_token` (expiração de 15-30 min). Para provisionamentos iniciados pelo usuário, o sistema passa tanto `access_token` quanto `refresh_token`:

```
refresh_token = request.cookies.get('refresh_token')
run_playbook(playbook_path, extra_vars, provision_id,
refresh_token=refresh_token)
```

Se o `access_token` expirar, o executor do playbook pode:

1. Detectar resposta 401 Não Autorizado
2. Chamar `POST /crm/auth/refresh` com `refresh_token`
3. Receber novo `access_token`
4. Tentar novamente a solicitação falhada

Para autenticação por chave API/lista de IPs, os tokens gerados podem ter expirações mais longas (1-2 horas), uma vez que esses são sistemas automatizados confiáveis.

O Processo de Provisionamento

1. **Criação do Trabalho**

   Quando uma solicitação de provisionamento é recebida, o sistema:

   - Valida a solicitação e verifica permissões
   - Carrega o playbook Ansible especificado na definição do produto
   - Cria um registro de `Provision` no banco de dados com status 1 (Executando)
   - Extrai variáveis da definição do produto e do corpo da solicitação
   - Captura tokens de autenticação para acesso à API

2. **Tratamento de Token**

   Os playbooks Ansible precisam se autenticar com a API CRM para recuperar dados e fazer alterações. O sistema de provisionamento trata isso de duas maneiras:

   - **Token Bearer (JWT)**: Para provisionamento iniciado pelo usuário, o `refresh_token` da solicitação é usado para gerar tokens de acesso frescos durante a execução do playbook
   - **Chave API/Auth por IP**: Para sistemas automatizados, um `access_token` é passado diretamente para o playbook

3. **Execução em Segundo Plano**

   O playbook é executado em uma thread em segundo plano. Isso permite que a API retorne imediatamente enquanto o provisionamento continua de forma assíncrona.

   Durante a execução:

   - Cada conclusão/falha de tarefa cria um registro `Provision_Event`
   - O manipulador de eventos monitora falhas críticas vs. ignoradas
   - Atualizações de status em tempo real são escritas no banco de dados
   - A UI pode fazer polling por atualizações via `GET /crm/provision/provision_id/<id>`

4. **Execução do Playbook**

   O playbook Ansible normalmente realiza estas operações:

   - Recupera informações do produto da API
   - Recupera informações do cliente da API
   - Atribui itens de inventário (cartões SIM, endereços IP, números de telefone, etc.)
   - Cria contas no OCS/OCS
   - Configura equipamentos de rede
   - Cria o registro de serviço na API CRM
   - Adiciona transações de custo de configuração
   - Envia e-mails/SMS de boas-vindas para os clientes

5. **Tratamento de Erros**

   Os playbooks Ansible usam seções `block` e `rescue` para rollback:

   - Se uma tarefa crítica falhar, a seção de resgate remove o provisionamento parcial
   - Tarefas com `ignore_errors: true` são marcadas como status 3 e não falham o trabalho
   - Erros fatais (sintaxe YAML, falhas de conexão) criam um evento de erro especial com informações de depuração

Exemplo: Playbook de Provisionamento Padrão

Aqui está um exemplo 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
```

Este playbook demonstra o fluxo típico:

1. Recuperar detalhes do produto da API CRM
2. Gerar um UUID de serviço exclusivo
3. Criar a conta de cobrança no OCS
4. Criar o registro de serviço via API CRM
5. Adicionar transações de custo de configuração
6. Se algo falhar, a seção `rescue` remove a conta OCS

Fluxo de Trabalho de Provisionamento Simples

O provisionamento simples é projetado para sistemas automatizados que precisam acionar o provisionamento sem interação do usuário. O caso de uso mais comum é o OCS acionando o provisionamento de complementos via ActionPlans.

Endpoints de Provisionamento Simples

OmniCRM fornece dois endpoints de provisionamento simples:

- `POST /crm/provision/simple_provision_addon/service_id/<id>/product_id/<id>`

  Para provisionamento automatizado de complementos (por exemplo, cobranças recorrentes, recargas automáticas)

- `POST /crm/provision/simple_provision_addon_recharge/service_id/<id>/product_id/<id>`

  Para operações de recarga rápida que precisam de feedback imediato

Autenticação para Provisionamento Simples

Os endpoints de provisionamento simples usam **lista de IPs permitidos** ou **chaves API** para autenticação:

- O IP de origem da solicitação é verificado contra `ip_whitelist` em `crm_config.yaml`
- Ou uma chave API de `api_keys` em `crm_config.yaml` pode ser fornecida
- Um token de acesso é gerado e passado para o playbook

Exemplo: Callback de ActionPlan do OCS

O OCS pode ser configurado para chamar o endpoint de provisionamento simples ao executar ações recorrentes:

```
{
  "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"
      }
    ]
  }]
}
```

A ação faz um POST HTTP para:

Isso aciona o playbook associado (por exemplo, `play_topup_no_charge.yaml`) que adiciona dados/créditos ao serviço.

Exemplo: Playbook de Recarga Simples

Do `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
```

Este playbook:

1. Obtém detalhes do serviço e do produto da API
2. Recupera o método de pagamento do cliente
3. Cobra o cliente via API Stripe
4. Adiciona saldo monetário ao OCS
5. Registra a transação no CRM
6. Envia um SMS de confirmação (com `ignore_errors: True` para que falhas não falhem o trabalho)

Cadeias de Provisionamento

Para produtos complexos que requerem múltiplos passos de provisionamento, o OmniCRM suporta **cadeias de provisionamento**. Uma cadeia executa múltiplos playbooks sequencialmente, passando contexto entre eles.

Exemplo de caso de uso: Um serviço agrupado que provisiona:

1. Serviço de internet base (cria o registro de serviço primário)
2. Complemento IPTV (usa o service_id do passo 1)
3. Complemento IP estático (usa o service_id do passo 1)

O serviço de provisionamento automaticamente:

- Consulta o banco de dados para o `service_id` criado pelo primeiro playbook
- Injeta-o nas `extra_vars` para os playbooks subsequentes
- Rastreia cada playbook como um registro `Provision` separado

Razões de Falha e Depuração

Quando o provisionamento falha, o sistema captura informações detalhadas para ajudar a diagnosticar o problema.

Cenários Comuns de Falha

Falhas Críticas de Tarefa (Status 2)

Essas causam a falha de todo o trabalho de provisionamento:

- Chamadas API retornando códigos de status inesperados
- Afirmações falhando (por exemplo, `assert: that: response.status == 200`)
- Itens de inventário obrigatórios ausentes
- Equipamentos de rede inacessíveis
- Credenciais inválidas ou tokens expirados

- Erros OCS/OCS

Falhas Ignoradas (Status 3)

Essas são registradas, mas não falham o trabalho:

- Notificações SMS/e-mail opcionais falhando
- Pesquisas de dados não críticas (marcadas com `ignore_errors: True`)
- Operações de limpeza durante o desprovisionamento

Erros Fatais

Esses impedem que o playbook seja executado:

- Erros de sintaxe YAML no playbook
- Variáveis indefinidas no playbook
- Arquivos de playbook ausentes
- Falhas de conexão com o controlador Ansible

Quando ocorre um erro fatal, o sistema cria um evento de erro especial contendo:

- O código de saída do Ansible
- Todo o stdout (contém detalhes do erro de sintaxe)
- Todo o stderr (contém erros de tempo de execução)
- Uma lista de causas comuns para esse tipo de falha
- Todas as variáveis passadas para o playbook

E-mails de Notificação de Erros

Quando o provisionamento falha (status 2), um e-mail é automaticamente enviado para a lista de notificação de falhas configurada (`provisioning.failure_list` em `crm_config.yaml`).

O e-mail inclui:

- Informações do cliente
- Detalhes do produto/serviço

- Resultados das tarefas codificados por cores:
  - **Verde**: Tarefas bem-sucedidas
  - **Laranja**: Tarefas falhadas, mas ignoradas
  - **Vermelho**: Falhas críticas
- Para falhas críticas: Saída de depuração completa incluindo corpos de solicitação/resposta
- Para erros fatais: Saída do Ansible, mensagens de erro e causas comuns

Monitoramento de Trabalhos de Provisionamento

OmniCRM fornece métricas abrangentes para monitorar o desempenho do provisionamento. Para detalhes completos sobre métricas de provisionamento, incluindo duração do trabalho, taxas de sucesso, contagem de tarefas e rastreamento de erros, veja Monitoramento & Métricas.

API de Status de Provisionamento

Para verificar o status de um trabalho de provisionamento:

```
GET /crm/provision/provision_id/<id>
Authorization: Bearer <token>
```

A resposta inclui:

```
{
  "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": "{...}"
    }
  ]
}
```


Listando Trabalhos de Provisionamento

Para obter uma lista paginada de todos os trabalhos de provisionamento:

```
GET /crm/provision/?
page=1&per_page=20&sort=provision_id&order=desc
Authorization: Bearer <token>
```

Suporta filtragem:

```
GET /crm/provision/?filters={"provisioning_status":
[2]}&search=Mobile
Authorization: Bearer <token>
```

Isso retorna apenas trabalhos falhados (status 2) onde a descrição contém "Mobile".

Melhores Práticas

Design de Playbook

- **Sempre use block/rescue**: Garanta que o provisionamento parcial possa ser revertido
- **Use ignore_errors com moderação**: Apenas para operações realmente opcionais
- **Registre variáveis importantes**: Use tarefas `debug` para registrar valores-chave para solução de problemas
- **Valide respostas**: Use `assert` para verificar se as respostas da API estão como esperado
- **Idempotência**: Projete playbooks para serem seguros para reexecução

Autenticação

- **Provisionamento iniciado pelo usuário**: Sempre use refresh_token para playbooks de longa duração
- **Provisionamento automatizado**: Use lista de IPs permitidos ou chaves API com tokens de acesso gerados
- **Expiração do token**: O refresh_token garante que os tokens de acesso sejam regenerados conforme necessário

Tratamento de Erros

- **Forneça contexto**: Inclua customer_id, service_id e detalhes da operação nas mensagens de erro
- **Notifique adequadamente**: Falhas críticas acionam e-mails, mas não spam para falhas esperadas
- **Informações de depuração**: Capture corpos de solicitação/resposta completos em registros Provision_Event

Segurança

- **Valide entradas**: Verifique customer_id, product_id, service_id antes do provisionamento
- **Verificações de permissão**: Verifique se os usuários podem apenas provisionar para clientes autorizados
- **Dados sensíveis**: Use o sistema de redacção para remover senhas/chaves dos logs
- **Lista de IPs permitidos**: Restringir endpoints de provisionamento simples apenas a sistemas confiáveis

Desempenho

- **Execução em segundo plano**: Nunca bloqueie respostas da API aguardando o provisionamento
- **Intervalos de polling**: A UI deve fazer polling por atualizações de status a cada 2-5 segundos
- **Tarefas paralelas**: Use a paralelização nativa do Ansible para operações independentes
- **Atualizações de banco de dados**: O manipulador de eventos atualiza o banco de dados em tempo real, sem necessidade de consultar durante a execução

Documentação Relacionada

- `concepts_ansible` - Conceitos gerais de provisionamento Ansible
- `concepts_sim_provisioning` - Provisionamento de cartão SIM (físico e eSIM)
- `concepts_api` - Autenticação e uso da API CRM
- `concepts_products_and_services` - Definições de produtos e serviços
- `administration_inventory` - Gerenciamento de inventário para provisionamento
- Monitoramento & Métricas - Métricas de desempenho e monitoramento do provisionamento

Registro de Atividade do Cliente

Toda alteração feita em um cliente, contato, site, serviço e parâmetros financeiros, como Transações, Faturas e Métodos de Pagamento, é registrada no Registro de Atividade.

Isso nos permite rastrear as mudanças feitas no sistema, quem as fez e quando foram feitas, e é útil para auditar alterações e rastrear problemas. Por exemplo, se um cliente disser que nunca recebeu uma fatura, podemos verificar o Registro de Atividade para ver se foi enviada, ou se um contato foi removido, podemos ver quem o removeu e quando.

O Registro de Atividade é uma lista cronológica de mudanças, com as mudanças mais recentes no topo e as mudanças mais antigas mais abaixo na lista, que podem ser filtradas a partir das abas.

Os registros do log de atividades não podem ser excluídos, mas podem ser filtrados, e os detalhes podem ser visualizados para ver o que foi alterado e por quem.

Adicionando um Serviço

Serviços são as coisas pelas quais cobramos o cliente, eles podem ser serviços de internet residencial, serviços móveis ou até mesmo serviços abstratos como alugar uma sub-rede ou fornecer eletricidade medida para um rack.

Um Serviço é apenas uma instância de um `Product <concepts_products_and_services>` para um determinado cliente, que é escolhido do catálogo de produtos e provisionado para o cliente.

Se você ainda não `created a customer <basics_create_customer>`, você precisará fazer isso primeiro, pois os serviços estão vinculados aos clientes, você também precisará definir um método de pagamento para o cliente, já que os serviços geram cobranças que precisam ser pagas.

Os clientes podem provisionar seus próprios serviços (se permitirmos), ou a equipe de Atendimento ao Cliente pode provisionar serviços para o cliente.

Com base na definição do produto, existem regras sobre quem pode comprar um produto, como permitir apenas que clientes empresariais comprem um produto empresarial, ou permitir apenas que clientes com um serviço móvel comprem um complemento móvel.

Os serviços também podem ter componentes de uso, como uso de dados, minutos de chamadas, ou outras cobranças baseadas em uso, e podem ter várias cobranças associadas a eles, como cobranças mensais, cobranças únicas ou cobranças de uso, podemos visualizar isso a partir do botão "Uso".

Muitos serviços suportam Complementos, por exemplo, um serviço móvel tem todos os recargas disponíveis no menu de Complementos, e um serviço de internet residencial pode ter um endereço IP estático ou dados extras disponíveis como um complemento, novamente tudo isso é definido no `product catalog<concepts_products_and_services>`.

Fluxo de Trabalho de Atribuição de Planos

O recurso Atribuir Plano permite que a equipe **provisione serviços para clientes** selecionando produtos do catálogo e iniciando o processo de provisionamento. Este é o método principal para criar novos serviços quando os clientes não se auto-provisionam.

Visão Geral

A atribuição de um plano envolve:

1. Selecionar um cliente
2. Escolher um produto do catálogo
3. Configurar requisitos de inventário (chips SIM, equipamentos, etc.)
4. Definir parâmetros de serviço (renovação automática, campos personalizados)
5. Iniciar o provisionamento
6. Monitorar o progresso do provisionamento

Este fluxo de trabalho é utilizado para todos os tipos de serviços: móvel, internet, IPTV e VoIP.

Acessando Atribuir Plano

Do Catálogo de Produtos:

Da Página do Cliente:

Da Página de Adicionais:

A interface de atribuição de plano é aberta em um modal ou página dedicada.

Fluxo de Trabalho Passo a Passo

Passo 1: Navegar pelo Catálogo de Produtos

O catálogo de produtos exibe os produtos disponíveis agrupados por categoria e tipo de cliente.

Alternâncias de Visualização:

> Categorias: • Planos de Serviço • Adicionais

Alternar entre os filtros Individual e Empresarial filtra os produtos para mostrar apenas aqueles disponíveis para o tipo de cliente selecionado.

Cartões de Produtos:

Os produtos são exibidos em um carrossel ou grade:

Clique em **"Atribuir ao Cliente"** para prosseguir.

Passo 2: Selecionar Cliente

Se você não estiver na página de um cliente, será solicitado que selecione um cliente.

Pesquisa de Clientes:

> Pesquisar Clientes: [John ▼]
>
> Clientes correspondentes: • John Smith (ID: 123) • John Doe (ID: 456) • Johnson Enterprises (ID: 789)

Digite para pesquisar por:

- Nome do cliente
- ID do cliente
- Endereço de e-mail
- Número de telefone

Selecione o cliente no menu suspenso.

Aviso

Certifique-se de que você selecionou o cliente correto antes de prosseguir. Atribuir um plano ao cliente errado requer intervenção manual para corrigir.

Passo 3: Configurar Inventário (se necessário)

Se o produto requer itens de inventário (definidos em `inventory_items_list`), os selecionadores de inventário aparecem.

Exemplo de Selecionador de Inventário:

> Itens de Inventário Necessários:
>
> Chip SIM *
>
> Chips SIM Disponíveis: • SIM-00123 - ICCID: 8944...0001 (Novo) • SIM-00124 - ICCID: 8944...0002 (Novo) • SIM-00125 - ICCID: 8944...0003 (Novo)
>
> Número Móvel *
>
> Números Disponíveis: • +44 7700 900123 (Reservado) • +44 7700 900124 (Disponível) • +44 7700 900125 (Disponível)

Regras de Seleção de Inventário:

- Asterisco (*) indica campo obrigatório
- Apenas itens de inventário disponíveis são mostrados (status: "Em Estoque" ou "Novo")

- Os menus suspensos carregam dinamicamente com base nos nomes dos modelos de inventário
- Uma vez selecionados, os itens são temporariamente reservados

O que Acontece:

- Os itens de inventário selecionados são passados como variáveis para o playbook de provisionamento
- Durante o provisionamento, os itens são atribuídos ao serviço e ao cliente
- O status do item muda de "Em Estoque" para "Atribuído"

Passo 4: Configurar Renovação Automática (opcional)

Para serviços recorrentes, você pode ser solicitado a definir a renovaç��o automática:

> Você gostaria de habilitar a renovação automática para este serviço?
>
> Quando habilitado, este serviço será renovado automaticamente ao final de cada período de faturamento e cobrará o método de pagamento padrão do cliente.
>
> [Não] [Sim]

Comportamento da Renovação Automática:

- **Sim**: O serviço renova automaticamente, o cliente é cobrado mensalmente
- **Não**: O serviço expira ao final do período do contrato, renovação manual necessária

Melhor Prática: Defina como "Sim" para serviços de consumo, "Não" para serviços únicos ou quando o cliente solicitar controle manual.

Passo 5: Revisar e Confirmar

A tela de revisão mostra todas as seleções antes do provisionamento:

> Cliente: John Smith (ID: 123) Produto: Móvel - Plano de 20GB

> Inventário: • Chip SIM: SIM-00123 (ICCID: 8944...0001) • Número Móvel: +44 7700 900123
>
> Preços: • Taxa de Configuração: £0.00 • Custo Mensal: £15.00
>
> Renovação Automática: Sim
>
> [Cancelar] [Confirmar & Provisionar]

Clique em **"Confirmar & Provisionar"** para iniciar o processo de provisionamento.

Passo 6: Progresso do Provisionamento

O modal de provisionamento exibe o progresso em tempo real:

> ✓ Validando conta do cliente ✓ Atribuindo Chip SIM (ICCID: 8944...0001) ✓ Atribuindo Número Móvel (+44 7700 900123) ⟳ Configurando conta OCS (em progresso...) □ Criando registro de serviço □ Enviando e-mail de boas-vindas
>
> Progresso: 3 de 6 tarefas concluídas

O modal consulta a API de provisionamento a cada 0,2 segundos para atualizações de status.

Indicadores de Progresso:

☑ Concluído com sucesso

⟳ Atualmente em execução

□ Pendente (não iniciado)

✗ Falhou (se ocorrerem erros)

Passo 7: Conclusão

Sucesso:

> Serviço provisionado com sucesso para John Smith
>
> ID do Serviço: 789 Nome do Serviço: Móvel - +44 7700 900123 Status: Ativo
>
> [Ver Serviço] [Fechar]

Clique em **"Ver Serviço"** para abrir a página de detalhes do serviço.

Falha:

Se o provisionamento falhar:

> Erro: Não foi possível conectar ao OCS
>
> O registro de serviço foi criado, mas o provisionamento não foi concluído. Por favor, revise o erro e tente novamente.

> ID de Provisionamento: 456
>
> [Ver Logs] [Tentar Novamente] [Fechar]

- **Ver Logs**: Abre detalhes do provisionamento com mensagens de erro
- **Tentar Novamente**: Tenta o provisionamento novamente
- **Fechar**: Sai do modal (o registro de serviço permanece, mas não é ativado)

Casos Especiais

Adicionando Adicionais a um Serviço Existente

Ao atribuir um adicional (categoria: "addon") a um cliente que já possui um serviço:

1. **Detecção Automática de Serviço:**
   - O sistema encontra os serviços existentes do cliente
   - Filtra por service_type (adicional móvel só aparece para serviços móveis)
   - Se o cliente tiver vários serviços correspondentes, solicita que selecione qual deles
2. **Nenhum Novo Serviço Criado:**
   - O adicional é provisionado contra o service_id existente
   - Usa a conta OCS do serviço existente
   - Inventário (se houver) atribuído ao serviço existente
3. **Playbook de Provisionamento:**
   - Playbook diferente do de serviços autônomos
   - Normalmente adiciona saldo, recursos ou equipamentos à conta existente

Exemplo:

> Serviços Móveis Existentes: • Móvel - +44 7700 900123 (ID: 789) • Móvel - +44 7700 900456 (ID: 790)
>
> Qual serviço deve receber este adicional? [Móvel - +44 7700 900123 ▼]

> [Cancelar] [Continuar]

Provisionamento para Clientes Empresariais

Clientes empresariais podem ter requisitos adicionais:

- **Seleção de Local** - Escolha qual local de negócios receberá o serviço
- **Atribuição de Contato** - Designar contatos de faturamento/técnicos
- **Campos Personalizados** - Números de conta, centros de custo, números de pedido

Exemplo de Fluxo Empresarial:

> Selecionar Local de Instalação: [Escritório de Londres - 123 Main St ▼]
>
> Contato de Faturamento: [Jane Doe - <jane@acme.com> ▼]
>
> Contato Técnico: [Bob Smith - <bob@acme.com> ▼]
>
> Número do Pedido: [PO-2025-001234_____]
>
> [Cancelar] [Continuar]

Atribuição de Serviço em Lote

Para atribuir o mesmo plano a vários clientes (por exemplo, migrações em massa):

1. Use importação CSV (se disponível)
2. Ou atribua individualmente com configurações de modelo
3. O inventário deve estar disponível em lote
4. Monitore a fila de provisionamento para evitar sobrecarga

Fluxos de Trabalho Comuns

Fluxo de Trabalho 1: Novo Serviço Móvel

1. O cliente entra na loja querendo serviço móvel
2. A equipe abre **Produtos → Planos**
3. Alterna para o tipo de cliente **Individual**
4. Seleciona **"Móvel - Plano de 20GB"**
5. Clica em **"Atribuir ao Cliente"**
6. Pesquisa o cliente pelo telefone: "+1234567890"
7. Seleciona **John Smith** dos resultados
8. Escolhe o Chip SIM do selecionador de inventário
9. Escolhe o número móvel disponível
10. Habilita **renovação automática**
11. Confirma e provisiona
12. Acompanha o progresso até a conclusão
13. Entrega o SIM ao cliente com o pacote de boas-vindas

Fluxo de Trabalho 2: Adicionando Internet a um Cliente Existente

1. Navega até a página do cliente: **Clientes → John Smith**
2. Clica na aba **Serviços**
3. Clica no botão **"Adicionar Serviço"**
4. Navega pelos planos de internet
5. Seleciona **"Fibra - 100Mbps"**
6. Seleciona o local de instalação (se empresa/múltiplos locais)
7. Escolhe o modem CPE do inventário
8. Define a data de instalação (se necessário)
9. Provisione o serviço
10. Cria o ticket de instalação

Fluxo de Trabalho 3: Atribuindo Adicional de Recarga de Dados

1. O cliente liga: "Preciso de mais dados"
2. A equipe pesquisa o cliente na busca global
3. Abre a aba de serviços do cliente
4. Clica em "Adicionais" ao lado do serviço móvel
5. Seleciona "5GB Data Boost"
6. Pagamento autorizado
7. Adicional provisionado imediatamente
8. O cliente recebe um aumento instantâneo de dados

Solução de Problemas

"Nenhum produto disponível"

- **Causa:** Filtros excluem todos os produtos
- **Correção:**
  - Alterne o tipo de cliente (Individual vs Empresarial)
  - Verifique se o catálogo de produtos tem produtos habilitados
  - Verifique se os produtos correspondem à elegibilidade do cliente

"Nenhum inventário disponível"

- **Causa:** Itens de inventário fora de estoque ou todos atribuídos
- **Correção:**
  - Adicione mais itens de inventário ao sistema
  - Verifique o status do item (deve ser "Novo" ou "Em Estoque")
  - Verifique se os nomes dos modelos de inventário correspondem aos requisitos do produto

"Cliente não encontrado"

- **Causa:** O cliente não existe ou o termo de pesquisa está incorreto
- **Correção:**

    - Crie o cliente primeiro
    - Tente diferentes termos de pesquisa (ID, e-mail, telefone)
    - Verifique se há erros de digitação

Provisionamento fica preso

- **Causa:** Erro no playbook ou sistema externo inacessível
- **Correção:**
    - Aguarde o tempo limite (normalmente 5 minutos)
    - Verifique os logs de provisionamento para erro específico
    - Verifique se OCS e sistemas de rede estão online
    - Tente o provisionamento novamente após corrigir o problema

Provisionamento é bem-sucedido, mas o serviço não funciona

- **Causa:** Conta OCS criada, mas a rede não foi atualizada
- **Correção:**
    - Verifique se OCS tem conta
    - Verifique se o SIM está ativado no HLR/HSS
    - Verifique o provisionamento da rede (RADIUS, DPI, etc.)
    - Revise as tarefas do playbook para etapas perdidas

Melhores Práticas

Antes de Atribuir:

- Verifique se o cliente tem um método de pagamento válido registrado
- Confirme a elegibilidade do cliente para o produto (residencial vs empresarial)
- Certifique-se de que o inventário necessário está disponível
- Revise os termos e preços do produto com o cliente

Durante a Atribuição:

- Verifique a seleção do cliente antes de confirmar
- Selecione os itens de inventário corretos (verifique os números de série)

- Habilite a renovação automática para conveniência (a menos que o cliente se oponha)
- Monitore o progresso do provisionamento até a conclusão

Após a Atribuição:

- Verifique se o serviço aparece na lista de serviços do cliente
- Verifique se o status do serviço é "Ativo"
- Confirme se o inventário foi atribuído corretamente
- Envie um e-mail de boas-vindas ou instruções ao cliente
- Teste o serviço, se possível (faça uma chamada de teste, verifique dados)

Para Adicionais:

- Confirme se o adicional é compatível com o serviço existente
- Explique a cobrança (única vs recorrente)
- Verifique o pagamento antes do provisionamento
- Verifique se o saldo foi atualizado imediatamente após o provisionamento do adicional

Documentação Relacionada

- `csa_add_service` - Visão geral dos serviços
- `concepts_products_and_services` - Conceitos do catálogo de produtos
- `administration_inventory` - Gerenciando itens de inventário
- `concepts_provisioning` - Detalhes do sistema de provisionamento
- `guide_product_lifecycle` - Ciclo de vida completo do produto, incluindo provisionamento

Modificando um Serviço

Os serviços podem ser modificados pelo cliente final através do `Self-Care Portal <self_care_portal>`, ou por um administrador via o portal de administração.

Uma vez que um serviço é provisionado, você pode modificar seus parâmetros, adicionar melhorias ou alterar configurações.

Editando Parâmetros do Serviço

Os parâmetros básicos do serviço podem ser modificados clicando no botão **Editar** na página de detalhes do serviço.

Campos Editáveis:

- Nome do Serviço
- Status do Serviço (Ativo, Inativo, Suspenso)
- Notas do Serviço
- Configurações de Auto-Renovação
- Itens de inventário vinculados
- Campos personalizados

As alterações são salvas imediatamente e refletidas no portal de autoatendimento do cliente.

Adicionando Complementos ao Serviço

Os complementos aprimoram os serviços existentes com recursos adicionais, permissões de dados ou hardware. Tipos comuns de complementos incluem:

- **Recargas de Dados** - Permissão de dados adicional (por exemplo, "5GB Data Boost")
- **Atualizações de Recursos** - Capacidades extras (por exemplo, "Chamadas Internacionais")
- **Aluguel de Hardware** - Adições de equipamentos (por exemplo, "Modem WiFi 6")
- **Serviços Premium** - Recursos aprimorados (por exemplo, "Endereço IP Estático")

Acessando o Catálogo de Complementos

A partir de uma página de detalhes do serviço:

1. Navegue até o serviço que você deseja aprimorar
2. Clique no botão **"Adicionar Complemento"** ou **"Navegar Complementos"**

3. O catálogo de complementos é aberto, filtrado para mostrar apenas complementos compatíveis

Filtragem Automática:

O sistema filtra automaticamente os complementos com base em:

- **Tipo de Serviço** - Mostra apenas complementos que correspondem ao tipo de serviço (móvel, internet, voip, etc.)
- **Tipo de Cliente** - Filtra por cliente residencial vs. empresarial
- **Compatibilidade do Serviço** - Verifica se o serviço atende aos requisitos do complemento
- **Disponibilidade** - Mostra apenas produtos habilitados

Por exemplo, se você estiver visualizando um serviço móvel residencial, verá apenas complementos móveis marcados para clientes residenciais.

Exibição de Complementos

Os complementos são exibidos em um carrossel interativo mostrando:

Exibição do Cartão do Produto:

Informações Exibidas:

- Ícone do produto
- Nome do produto
- Lista de recursos (itens em lista)
- Custo de configuração
- Custo mensal/recorrente
- Link de termos e condições
- Botão "Adicionar ao Serviço"

Provisionando um Complemento

Passo 1: Selecionar Complemento

Clique no cartão do complemento desejado, em seguida clique em **"Adicionar ao Serviço"**

Passo 2: Selecionar Inventário (se necessário)

Se o complemento requer inventário físico (por exemplo, aluguel de hardware), um seletor de inventário aparece:

> Modens Disponíveis: • Modem-12345 - TP-Link AX1800 (Novo) • Modem-12346 - TP-Link AX1800 (Novo) • Modem-12347 - Netgear RAX40 (Novo)

Selecione o item de inventário específico a ser atribuído a este serviço.

Passo 3: Configurar Auto-Renovação (opcional)

Para complementos recorrentes, você pode ser solicitado:

> Você gostaria de habilitar a auto-renovação para este complemento?
>
> [Não] [Sim]

- **Sim** - O complemento renova automaticamente a cada período de cobrança
- **Não** - Compra única, o cliente deve renovar manualmente

Passo 4: Confirmar e Provisionar

Revise os detalhes do complemento e clique em **"Confirmar"**

O modal de provisionamento aparece mostrando o progresso em tempo real:

> ✓ Validando pagamento ✓ Atribuindo inventário ⟳ Configurando serviço (em progresso...) □ Criando transação □ Enviando e-mail de confirmação

Uma vez completo, o complemento aparece na lista de complementos do serviço.

Provisionamento nos Bastidores:

Quando você adiciona um complemento, o sistema:

1. Valida se o cliente pode comprar o complemento
2. Executa o playbook Ansible do complemento (`provisioning_play`)
3. Atribui quaisquer itens de inventário necessários ao serviço
4. Cria registro de transação para cobrança
5. Atualiza a configuração do serviço (OCS, sistemas de rede, etc.)
6. Envia notificação de confirmação ao cliente

Para detalhes técnicos sobre o provisionamento de complementos, veja `Complete Product Lifecycle Guide - Adding Addons <guide_product_lifecycle>`.

Visualizando Complementos Ativos

Complementos ativos aparecem na página de detalhes do serviço na seção **"Complementos"**:

Exemplo de Exibição:

> □ 5GB Data Boost
> Adicionado: 10 Jan 2025 Expira: 17 Jan 2025 Status: Ativo Custo: £5.00
>
> □ Aluguel de Modem WiFi 6
> Adicionado: 01 Dec 2024 Inventário: Modem-12345 Auto-Renovação: Sim Mensal: £10.00

Removendo Complementos

Para remover um complemento de um serviço:

1. Navegue até a página de detalhes do serviço
2. Localize o complemento na seção **"Complementos Ativos"**
3. Clique no botão **"Remover"** ou no **ícone de lixeira** ao lado do complemento
4. Confirme a remoção

O Que Acontece:

- **Complementos de Hardware** - O playbook de desprovisionamento é executado, o inventário é marcado para retorno
- **Complementos Virtuais** - Benefícios removidos imediatamente
- **Complementos com Auto-renovação** - Cancela renovações futuras
- **Transação criada** - Credita qualquer valor proporcional, se aplicável

Aviso

Remover complementos de hardware (modens, CPE, etc.) normalmente requer que o equipamento seja devolvido. O sistema marcará o inventário para retorno e pode enviar instruções de devolução ao cliente.

Cenários Comuns de Complementos

Cenário 1: Cliente Faltando Dados

1. Cliente contata o suporte: "Eu usei minha cota mensal"
2. Funcionário navega até o serviço móvel do cliente
3. Clica em "Adicionar Complemento"
4. Seleciona "5GB Data Boost"
5. Provisiona imediatamente
6. Cliente recebe recarga de dados instantânea

Cenário 2: Empresa Precisa de IP Estático

1. Cliente empresarial solicita IP estático para VPN
2. Funcionário abre o serviço de internet do cliente
3. Navega pelos complementos, seleciona "Endereço IP Estático"
4. O sistema provisiona IP do pool disponível (inventário)
5. Configura o roteamento no equipamento de rede
6. Empresa recebe detalhes de configuração do IP

Cenário 3: Aluguel de Equipamento para Serviço de Fibra

1. Cliente se inscreve para internet de fibra
2. Durante o provisionamento, cliente opta pelo aluguel de modem
3. Funcionário adiciona o complemento "Modem WiFi 6"
4. Seleciona o modem disponível no inventário
5. Modem enviado ao cliente
6. Custo recorrente de £10/mês adicionado à conta

Solução de Problemas

"Nenhum complemento compatível disponível"

- **Causa:** Nenhum complemento corresponde ao tipo de serviço ou tipo de cliente
- **Solução:** Verifique se o catálogo de produtos tem complementos com `service_type` e configurações `residential`/`business` correspondentes

Falha no provisionamento do complemento

- **Causa:** Erro no playbook de provisionamento ou inventário indisponível
- **Solução:**
  - Verifique os logs de provisionamento para erro específico
  - Verifique se os itens de inventário estão em estoque (se necessário)
  - Revise os logs do playbook no sistema de provisionamento

Seletor de inventário não mostra itens

- **Causa:** Nenhum item de inventário disponível do tipo necessário
- **Solução:**
  - Adicione itens de inventário ao sistema
  - Verifique se os itens existentes não estão todos atribuídos ou danificados
  - Verifique se o nome do modelo de inventário corresponde exatamente a `inventory_items_list`

Auto-renovação não funcionando

- **Causa:** Flag de auto-renovação não configurada ou método de pagamento expirado
- **Solução:**
  - Verifique se o serviço tem `auto_renew: true`
  - Verifique se o cliente tem um método de pagamento válido registrado
  - Revise trabalhos agendados no sistema de cobrança

Documentação Relacionada

- `guide_product_lifecycle` - Fluxo completo de provisionamento de complementos
- `csa_add_service` - Criando novos serviços
- `concepts_products_and_services` - Conceitos de produtos e serviços
- `administration_inventory` - Gestão de inventário para complementos de hardware

Gerenciamento de Serviços

A interface de Gerenciamento de Serviços fornece ferramentas abrangentes para visualizar, filtrar e gerenciar serviços de clientes em toda a sua organização.

Documentação relacionada: `Adicionando Serviços <csa_add_service>`, `Uso de Serviços <csa_service_usage>`, `Modificando Serviços <csa_modify>`, `Produtos e Serviços <concepts_products_and_services>`.

Acessando a Lista de Serviços

Navegue para:

Ou diretamente:

Isso exibe todos os serviços com capacidades de filtragem, ordenação e gerenciamento em massa.

Visão Geral da Lista de Serviços

A lista de serviços exibe os serviços em uma tabela com:

Colunas:

- **ID do Serviço** - Identificador único
- **Nome do Serviço** - Nome descritivo (por exemplo, "Móvel - +44 7700 900123")
- **Nome do Cliente** - Cliente associado (link clicável)
- **Tipo de Serviço** - móvel, iptv, internet, voip
- **Status** - Ativo, Inativo, Suspenso
- **Data de Início** - Quando o serviço começou
- **Data de Término** - Quando o serviço expira/termina
- **Custo Mensal** - Preço de varejo
- **Ações** - Visualizar, Editar, Adicionais, Uso, Excluir

Filtrando Serviços

Abas de Status

Filtre rapidamente os serviços por status usando a barra de abas:

- **Todos os Resultados** - Mostra todos os serviços, independentemente do status
- **Ativo** - Serviços atualmente ativos (visualização padrão)

- **Inativo** - Serviços cancelados ou expirados
- **Suspenso** - Serviços temporariamente desativados (não pagamento, fraude, etc.)

Clicar em uma aba atualiza a lista instantaneamente. A aba ativa é destacada.

Filtro por Tipo de Serviço

Filtre por tipo de serviço:

Selecione um ou mais tipos para mostrar apenas os serviços correspondentes.

Casos de Uso:

- **Móvel** - Visualizar todos os serviços de SIM móvel
- **Internet** - Mostrar fibra, DSL e wireless fixo
- **IPTV** - Exibir serviços de assinatura de TV
- **VoIP** - Listar serviços de voz sobre IP

Filtro por Cliente

Pesquise e filtre por cliente:

Comece a digitar o nome de um cliente para ver os resultados correspondentes:

Selecione um cliente para mostrar apenas os serviços dele.

Caso de Uso: Visualizar rapidamente todos os serviços de um cliente específico.

Pesquisa

Pesquisa global em todos os campos de serviço:

Pesquisas:

- Nome do serviço

- ID do serviço
- Nome do cliente
- Números de telefone (para serviços móveis)
- UUID do serviço

Exemplo: Pesquise "0770" para encontrar todos os serviços com números de telefone contendo "0770".

Ordenando Serviços

Clique em qualquer cabeçalho de coluna para ordenar por esse campo:

Colunas Ordenáveis:

- ID do Serviço (padrão: mais recente primeiro)
- Nome do Serviço (alfabético)
- Nome do Cliente (alfabético)
- Tipo de Serviço (alfabético)
- Data de Início (cronológico)
- Data de Término (cronológico)
- Custo Mensal (numérico)

Direção da Ordenação:

- Clique uma vez: Ascendente (A-Z, mais antigo-mais recente, mais baixo-mais alto)
- Clique duas vezes: Descendente (Z-A, mais recente-mais antigo, mais alto-mais baixo)
- A coluna de ordenação ativa mostra o indicador ▲ ou ▼

Dropdown de Ordenação:

O dropdown no canto superior direito fornece predefinições rápidas de ordenação:

Visualizando Detalhes do Serviço

Clique em qualquer nome de serviço para abrir a visualização detalhada do serviço.

Abas de Detalhes do Serviço:

1. **Visão Geral** - Resumo do serviço, status, datas, preços
2. **Inventário** - Equipamentos atribuídos (cartões SIM, modems, etc.)
3. **Transações** - Cobranças, créditos, pagamentos
4. **Uso** - Estatísticas de uso de dados/voz/SMS
5. **Adicionais** - Adicionais ativos e disponíveis
6. **Registro de Atividades** - Histórico de alterações

Ações Rápidas

Na lista de serviços, clique no menu de ações (⋮) para operações rápidas:

- **Visualizar** - Abre os detalhes do serviço
- **Editar** - Modifica os parâmetros do serviço
- **Adicionais** - Navegar e adicionar melhorias ao serviço
- **Uso** - Visualizar uso atual e saldo
- **Excluir** - Cancelar/remover serviço

Operações em Massa

Selecione vários serviços usando caixas de seleção para realizar ações em massa.

Selecionando Serviços

Seleção Individual:

Clique na caixa de seleção ao lado de cada serviço que deseja selecionar.

Selecionar Todos:

Clique na caixa de seleção no cabeçalho da tabela para selecionar todos os serviços visíveis.

> ☑ Serviço 1 ☑ Serviço 2 ☑ Serviço 3

Contador de Seleção:

A interface mostra quantos serviços estão selecionados:

Ações em Massa

Uma vez que os serviços estão selecionados, os botões de ação em massa aparecem:

Excluir Múltiplos Serviços:

1. Selecione os serviços a serem excluídos
2. Clique no botão **"Excluir Selecionados"**
3. Confirme a exclusão no modal

Aviso

A exclusão em massa é permanente e irá:

- Cancelar todos os serviços selecionados
- Marcar o inventário como não atribuído
- Parar todas as cobranças recorrentes
- Criar entradas no registro de atividades

Casos de Uso:

- Limpar serviços de teste
- Cancelar serviços para contas de clientes fechadas
- Remover entradas duplicadas ou errôneas

Melhor Prática: Use filtros para restringir a lista antes das operações em massa para evitar exclusões acidentais.

Gerenciamento de Status do Serviço

Tipos de Status

Os serviços podem ter três status:

Ativo

- O serviço está operacional
- Cobranças se aplicam
- O cliente pode usar o serviço
- Exibido com um distintivo verde

Inativo

- O serviço foi cancelado ou expirou
- Nenhuma cobrança aplicada
- O cliente não pode usar o serviço
- Exibido com um distintivo cinza
- Inventário marcado para devolução/reforma

Suspenso

- Serviço temporariamente desativado
- Pode ou não cobrar (configurável)
- O cliente não pode usar o serviço
- Exibido com um distintivo laranja/amarelo
- Razões comuns: Não pagamento, investigação de fraude, solicitação do cliente

Mudando o Status do Serviço

Para mudar o status de um serviço:

1. Abra os detalhes do serviço

2. Clique no botão **"Editar"**
3. Altere o dropdown **"Status do Serviço"**
4. Clique em **"Salvar"**

O Que Acontece:

- **Ativo → Inativo:** Aciona o playbook de desprovisionamento (se configurado)
- **Ativo → Suspenso:** Desativa o serviço na OCS/rede, mas mantém o registro
- **Suspenso → Ativo:** Reativa o serviço, retoma a cobrança
- **Inativo → Ativo:** Pode acionar o reaprovisionamento (use com cautela)

Visualização de Uso do Serviço

Clique em **"Uso"** no menu de ações para abrir o modal de uso.

Informações Exibidas:

- **Saldo** - Crédito restante ou valor pré-pago
- **Uso de Dados** - Usado vs. alocado (barra de progresso)
- **Uso de Voz** - Minutos usados vs. limite do plano
- **Uso de SMS** - Mensagens enviadas vs. limite
- **Data de Expiração** - Quando o saldo/plano atual expira

Ações:

- **Recarregar** - Adicionar crédito ou dados (para serviços pré-pagos)
- **Ver Detalhes** - Ver detalhamento do uso
- **Exportar** - Baixar relatório de uso (se disponível)

Próximas Renovações Automáticas

A visualização de Próximas Renovações Automáticas fornece uma interface centralizada para monitorar e gerenciar todas as renovações de serviços agendadas em sua organização.

Acessando Renovações Automáticas

Navegue para:

Ou diretamente:

Isso exibe todos os serviços com renovação automática agendada, ordenados pela próxima data de renovação.

Visão Geral das Renovações Automáticas

A lista de renovações automáticas exibe renovações agendadas em uma tabela com:

Colunas:

- **Cliente** - Nome do cliente (link clicável para a visão geral do cliente)
- **Serviço** - Nome do serviço (link clicável para a visão geral do cliente)
- **Produto** - Nome do produto/plano sendo renovado
- **Custo** - Custo da renovação (do preço de varejo do produto)
- **Renova Em** - Data e hora da próxima renovação com formato legível por humanos
- **Status** - Status do serviço (Ativo, Suspenso, etc.)
- **Ações** - Renovar agora ou remover renovação automática

Exibição de Exemplo:

Como Funciona a Renovação Automática

A renovação automática é agendada no CGRateS (o sistema de cobrança) quando um serviço é criado ou modificado. O sistema:

1. **Agenda Ação** - Cria um ActionPlan no CGRateS com a data de renovação
2. **Monitora Saldo** - Verifica se o cliente tem saldo suficiente antes da renovação
3. **Executa Renovação** - Na data agendada, renova automaticamente o serviço
4. **Atualiza Registros** - Cria transações, atualiza datas de serviço, e registra atividades

Fonte de Dados:

A visualização de Próximas Renovações Automáticas consulta o CGRateS diretamente usando a `ApierV1.GetScheduledActions` API e enriquece os dados com informações de cliente e serviço do banco de dados CRM.

Renovar Agora

Para acionar manualmente uma renovação antes da data agendada:

1. Clique no botão **(Renovar Agora)** para o serviço

2. Revise os detalhes da renovação no modal de confirmação:

> Cliente: Acme Corp Serviço: FixedWireless_75628fa5 Produto: Internet Residencial Mega Custo: $89.99
>
> Próxima renovação agendada: 11 de Nov, 2025 10:45 AM
>
> [Cancelar] [Confirmar Renovação]

3. Clique em **"Confirmar Renovação"** para processar imediatamente
4. Um trabalho de provisionamento é criado e executado
5. Monitore o progresso no modal de status de provisionamento

O Que Acontece:

- O serviço é recarregado com as alocações do produto
- O saldo é atualizado no CGRateS
- Um registro de transação é criado
- A data de término do serviço é estendida
- Uma entrada no registro de atividades é criada

Casos de Uso:

- O cliente solicita renovação antecipada
- Resolver problemas de serviço redefinindo alocações
- Testar processos de renovação
- O cliente deseja usar o serviço antes da renovação agendada

Remover Renovação Automática

Para cancelar a renovação automática de um serviço:

1. Clique no botão **(Remover Renovação Automática)** para o serviço
2. Confirme a exclusão no modal:

> Isso removerá a renovação automática agendada para:

> Cliente: Acme Corp Serviço: FixedWireless_75628fa5 Próxima Renovação: 11 de Nov, 2025 10:45 AM
>
> O serviço não será renovado automaticamente. Você precisará renovar manualmente ou o serviço expirará na data de término.
>
> [Cancelar] [Remover Renovação Automática]

3. Clique em **"Remover Renovação Automática"** para confirmar
4. O ActionPlan é removido do CGRateS
5. O serviço expirará naturalmente, a menos que renovado manualmente

O Que Acontece:

- O ActionPlan é excluído do CGRateS
- O status do serviço permanece inalterado
- O serviço expirará na sua data de término atual
- O cliente deve renovar manualmente ou o serviço será interrompido

Casos de Uso:

- O cliente está cancelando o serviço
- Mudando para o processo de renovação manual
- O plano de serviço está mudando
- O cliente solicitou parar cobranças automáticas

Aviso

Remover a renovação automática significa que o serviço expirará, a menos que renovado manualmente. Os clientes não receberão continuidade automática do serviço.

Entendendo a Exibição

Coluna do Cliente:

- Mostra o nome do cliente como um link clicável

- Links para a página de visão geral do cliente
- Mostra "N/A" se o serviço não estiver vinculado a um cliente (serviço órfão)

Coluna do Serviço:

- Mostra o nome/UUID do serviço
- Links para a página de visão geral do cliente
- Mostra o UUID do serviço se o nome não estiver definido
- Mostra "N/A" se o serviço não puder ser encontrado no banco de dados

Coluna do Produto:

- Nome do produto do ActionPlan do CGRateS
- Extraído dos metadados da ação agendada
- Sempre mostra o nome do produto, mesmo que o serviço não seja encontrado

Coluna do Custo:

- Custo de varejo da tabela de Produtos no CRM
- Valor que será cobrado na renovação
- Mostra "N/A" se o produto não for encontrado no banco de dados

Coluna Renova Em:

- Data e hora exatas da próxima renovação
- Tempo relativo legível por humanos (por exemplo, "3 horas a partir de agora", "2 dias a partir de agora")
- Consciente do fuso horário (usa o fuso horário do servidor CGRateS)

Coluna de Status:

- Status atual do serviço do banco de dados CRM
- **Ativo** (Verde) - O serviço está operacional
- **Suspenso** (Laranja) - O serviço está temporariamente desativado
- **Desconhecido** (Cinza) - Serviço não encontrado no banco de dados ou status não definido

Resolução de Problemas

Serviços mostrando "N/A" para Cliente/Serviço

- **Causa:** UUID do serviço no CGRateS não corresponde ao service_uuid no banco de dados CRM
- **Solução:**
  - Verifique se o serviço existe no banco de dados
  - Verifique se o formato do service_uuid corresponde: `ServiceType_UUID` (por exemplo, `FixedWireless_75628fa5`)
  - O serviço pode ter sido excluído do CRM, mas ainda está agendado no CGRateS

Custo mostrando "N/A"

- **Causa:** ID do produto no CGRateS não existe na tabela de Produtos do CRM
- **Solução:**
  - Verifique se o produto existe no banco de dados
  - Verifique se o product_id no ActionPlan corresponde a um produto no CRM
  - O produto pode ter sido excluído

"Renova Em" não mostrando data

- **Causa:** NextRunTime não presente na resposta do CGRateS
- **Solução:**
  - Verifique a configuração do ActionPlan do CGRateS
  - Verifique se o ActionTiming está configurado corretamente
  - Verifique os logs do CGRateS para erros

"Renovar Agora" falha

- **Causa:** Vários erros de provisionamento
- **Resolução de Problemas:**
  - Verifique o modal de status de provisionamento para detalhes do erro
  - Verifique se os playbooks do Ansible estão configurados corretamente
  - Verifique a conectividade com a OCS

- Revise os logs de provisionamento

Renovação automática removida, mas ainda mostrando

- **Causa:** Atraso no cache ou problema de sincronização do CGRateS
- **Solução:**
  - Clique no botão **"Atualizar"** para recarregar os dados
  - Verifique se o ActionPlan foi realmente removido no CGRateS
  - Verifique a conectividade da API do CGRateS

Paginação

Os serviços são exibidos em páginas para desempenho:

> ◄ Anterior 1 [2] 3 4 5 Próximo ►
>
> Itens por página: [10 ▼]
>
> - 10
> - 25
> - 50
> - 100

Controles:

- **Anterior/Próximo** - Navegar entre páginas
- **Números de Página** - Ir para uma página específica
- **Itens por Página** - Ajustar quantos serviços aparecem por página

Dica de Desempenho: Use filtros para reduzir o total de resultados em vez de aumentar itens por página.

Distintivos e Indicadores de Serviço

Indicadores visuais ajudam a identificar rapidamente os estados dos serviços:

Distintivos de Status:

Indicador de Renovação Automática:

Serviços com renovação automática ativada mostram:

Expirando em Breve:

Serviços que expiram dentro de 7 dias mostram:

Vencido:

Serviços com saldo pendente mostram:

Integração com CGRateS (Avançado)

Para serviços integrados com CGRateS (o motor de cobrança e tarifação), administradores podem gerenciar configurações avançadas diretamente da visualização do serviço. Isso inclui atributos, filtros e visualização de sessões ativas.

Nota

Os recursos de integração do CGRateS exigem a permissão **cgrates_api_access**. Apenas administradores têm acesso a esses recursos por padrão. Veja `rbac` para configuração de permissões.

Provisionamento Automático vs. Gerenciamento Manual

Na operação normal, atributos e filtros do CGRateS são provisionados automaticamente pelo Ansible durante o fluxo de trabalho inicial de provisionamento do serviço. Quando um novo serviço é criado, os playbooks de provisionamento:

- Criam a conta de serviço no CGRateS
- Configuram atributos (IMSI, MSISDN, identificadores de conta, perfis de velocidade, etc.)

- Configuram filtros para garantir a tarifação correta
- Aplicam os planos de tarifação apropriados

No entanto, há casos em que você pode precisar modificar essas configurações após o serviço ter sido provisionado:

- **Mudanças no Perfil de Velocidade** - O cliente atualiza/diminui a largura de banda (MaxBitrateDL/UL)
- **Ajustes de Políticas** - Alterar políticas de QoS ou regras de modelagem de tráfego (PcefPolicyName)
- **Mudanças no Número de Telefone** - Atualizar MSISDN ou outros identificadores
- **Resolução de Problemas** - Corrigir configurações incorretas ou testar diferentes configurações
- **Configurações Especiais** - Aplicar atributos personalizados que não fazem parte do provisionamento padrão

A interface de gerenciamento manual permite que os administradores façam essas mudanças diretamente sem precisar reexecutar todo o fluxo de trabalho de provisionamento. Isso é particularmente útil para:

- **Mudanças Rápidas** - Modificar um único atributo sem esperar pelo provisionamento
- **Testes** - Experimentar diferentes configurações
- **Suporte ao Cliente** - Resolver problemas rapidamente durante chamadas de suporte
- **Configurações Personalizadas** - Aplicar configurações específicas do serviço que não são cobertas por modelos

Aviso

Mudanças manuais na configuração do CGRateS ignoram o fluxo de trabalho padrão de provisionamento. Certifique-se de entender o impacto de suas mudanças, pois configurações incorretas podem afetar a cobrança e a funcionalidade do serviço. Todas as mudanças são registradas no feed de atividades do cliente para fins de auditoria.

Acessando Recursos do CGRateS

Ao visualizar ou editar um serviço que está provisionado no CGRateS, três seções colapsáveis aparecem na parte inferior do formulário do serviço:

- **Atributos do CGRateS** - Configurar atributos específicos do serviço
- **Filtros do CGRateS** - Definir regras de filtragem para o serviço
- **Sessões Ativas** - Visualizar sessões ativas em tempo real

Cada seção é colapsada por padrão para manter a interface limpa. Clique no cabeçalho da seção para expandir e visualizar/editar a configuração.

As seções colapsadas mostram distintivos de contagem indicando quantos atributos, filtros ou sessões ativas existem para o serviço.

Atributos do CGRateS

Atributos permitem que você defina campos personalizados e transformações que são aplicadas a eventos de tarifação para este serviço específico.

Formato do ID do Atributo: `ATTR_ACCOUNT_{service_uuid}`

Exemplo: Para o serviço com UUID `Mobile_SIM_c2880638`, o ID do perfil de atributo é `ATTR_ACCOUNT_Mobile_SIM_c2880638`

Gerenciando Atributos:

1. Abra a visualização de edição do serviço
2. Expanda a seção **"Atributos do CGRateS"**
3. Clique no botão **"Editar Atributos"**
4. Adicione/modifique/remova atributos conforme necessário
5. Clique em **"Salvar Atributos"**

Campos de Atributo:

- **Caminho** - O campo a ser modificado (por exemplo, `*req.Account`, `*req.IMSI`)

- **Tipo** - Como o valor é definido:
  - `*constant` - Define um valor fixo
  - `*variable` - Captura valor de campos de eventos usando RSRParser
  - `*composed` - Anexa valor em vez de sobrescrever
  - `*usage_difference` - Calcula a duração entre dois campos
  - `*sum` - Soma múltiplos valores
  - `*value_exponent` - Computa o expoente de um campo
- **Regras** - O(s) valor(es) a serem aplicados (pode ter várias regras por atributo)

Configuração de Atributo de Exemplo:

```
{
  "Path": "*req.Account",
  "Type": "*constant",
  "Value": [{"Rules": "Mobile_SIM_474a380a"}]
}
```

Casos de Uso Comuns:

- Definir identificador de conta para tarifação
- Mapear IMSI/MSISDN para o serviço
- Configurar limites de largura de banda (MaxBitrateDL/UL)
- Definir nomes de políticas (PcefPolicyName)
- Transformar ou enriquecer eventos de tarifação

Registro de Atividades:

Todas as modificações de atributos são registradas no feed de atividades do cliente com todos os detalhes das mudanças feitas.

Filtros do CGRateS

Filtros definem regras de correspondência que determinam quando a configuração deste serviço deve ser aplicada durante a tarifação.

Formato do ID do Filtro: `FLTR_ACCOUNT_{service_uuid}`

Exemplo: Para o serviço com UUID `Mobile_SIM_c2880638`, o ID do filtro é `FLTR_ACCOUNT_Mobile_SIM_c2880638`

Gerenciando Filtros:

1. Abra a visualização de edição do serviço
2. Expanda a seção **"Filtros do CGRateS"**
3. Clique no botão **"Editar Filtros"**
4. Adicione/modifique/remova regras de filtro
5. Clique em **"Salvar Filtros"**

Campos de Regra de Filtro:

- **Elemento** - O campo a ser correspondido (por exemplo, `~*req.Account`, `~*req.Destination`)
- **Tipo** - Tipo de correspondência:
  - `*string` - Correspondência exata de string
  - `*prefix` - Começa com o valor especificado
  - `*suffix` - Termina com o valor especificado
  - `*empty` - Campo está vazio
  - `*exists` - Campo existe
  - `*notexists` - Campo não existe
  - `*timings` - Correspondência de padrões de data/hora
  - `*destinations` - Correspondência de padrões de destino
  - `*rsr` - Correspondência de campo RSR
  - `*gt` / `*gte` / `*lt` / `*lte` - Comparações numéricas
- **Valores** - O(s) valor(es) a serem correspondidos (pode ter múltiplos valores por regra)

Configuração de Filtro de Exemplo:

```
{
  "Element": "~*req.Account",
  "Type": "*string",
  "Values": ["Mobile_SIM_474a380a"]
}
```

Casos de Uso Comuns:

- Garantir que atributos se apliquem apenas a contas específicas
- Filtrar por destino (nacional vs. internacional)
- Filtragem baseada em tempo (horário de pico vs. fora de pico)
- Filtrar por tipo ou categoria de serviço

Registro de Atividades:

Todas as modificações de filtros são registradas no feed de atividades do cliente.

Sessões Ativas

Visualize sessões ativas em tempo real para este serviço. Isso mostra chamadas em andamento, sessões de dados ou outros eventos faturáveis atualmente em progresso.

Visualizando Sessões Ativas:

1. Abra a visualização de edição do serviço
2. Expanda a seção **"Sessões Ativas"**
3. Veja a lista de sessões ativas
4. Clique em **"Ver Detalhes"** em qualquer sessão para ver os dados completos da sessão
5. Clique em **"Atualizar"** para recarregar a lista de sessões

Informações da Sessão Exibidas:

- **Hora de Configuração** - Quando a sessão começou
- **Uso** - Duração atual da sessão (em segundos)
- **Destino** - Número chamado ou destino

Modal de Detalhes da Sessão:

Clicar em "Ver Detalhes" abre um modal mostrando:

- **Informações Básicas:**
  - CGRID (ID da sessão)
  - Conta (UUID do serviço)
  - Hora de configuração
  - Uso/duração atual
  - Destino
  - Categoria
- **Dados Completos da Sessão:**
  - Representação JSON completa da sessão
  - Todos os campos da sessão do CGRateS
  - Estado da sessão em tempo real
  - Visualizador JSON rolável para inspeção

Casos de Uso:

- Monitorar chamadas ativas ou sessões de dados
- Resolver problemas de faturamento

- Verificar se a sessão está sendo tarifada corretamente
- Verificar atributos e valores da sessão
- Auditar o uso ativo do serviço

Taxa de Atualização:

As sessões são buscadas sob demanda quando você expande a seção. Clique em "Atualizar" para obter os dados mais recentes da sessão.

Nota

Apenas sessões que correspondem à conta deste serviço (UUID do serviço) são exibidas. O filtro `*string:~*req.Account:{service_uuid}` é aplicado automaticamente.

Proxy da API do CGRateS

Todas as operações do CGRateS (atributos, filtros, sessões) usam o endpoint proxy da API do OmniCRM:

Endpoint: `POST /crm/ocs/proxy`

Campos Obrigatórios:

- `method` - Método da API do CGRateS (por exemplo, `APIerSv1.GetAttributeProfile`)
- `params` - Array de parâmetros para o método
- `customer_id` - ID do cliente (para registro de atividades)
- `service_id` - ID do serviço (para registro de atividades)

Campos Opcionais:

- `tenant` - Inquilino do CGRateS (padrão para valor de configuração)

Exemplo de Solicitação:

```
{
  "method": "APIerSv1.GetAttributeProfile",
  "params": [{"ID": "ATTR_ACCOUNT_Mobile_SIM_c2880638"}],
  "customer_id": 123,
  "service_id": 456
}
```

Configuração do Inquilino:

O inquilino é definido automaticamente a partir do arquivo de configuração do OmniCRM (`crm_config.yaml`) sob `ocs.ocsTenant`. Isso garante que todas as operações do CGRateS usem o inquilino correto sem codificar valores no frontend.

Requisito de Permissão:

A permissão `cgrates_api_access` é necessária. Esta permissão é concedida ao papel `admin` por padrão.

Registro de Atividades:

Todas as operações da API do CGRateS que não são GET são registradas automaticamente no feed de atividades do cliente, incluindo:

- Método da API chamado
- Inquilino utilizado
- Parâmetros completos enviados
- ID do serviço em que a operação foi realizada
- Usuário que realizou a operação
- Timestamp

Isso cria um registro completo de auditoria de todas as mudanças de configuração do CGRateS.

Resolução de Problemas da Integração do CGRateS

"Permissão Negada" ao acessar recursos do CGRateS

- **Causa:** O usuário não possui a permissão `cgrates_api_access`
- **Solução:** Conceda permissão ao papel do usuário (normalmente recurso exclusivo para administradores)

Atributos ou Filtros não carregando

- **Causa:** Problema de conectividade com o CGRateS ou perfil não existe
- **Solução:**
  - Verifique a conectividade do servidor CGRateS na configuração
  - Verifique se a configuração do inquilino está correta
  - Verifique o console do navegador para erros da API
  - O perfil pode não existir ainda (mostrará um formulário vazio)

Mudanças não salvando

- **Causa:** Erro de validação ou erro da API do CGRateS
- **Solução:**
  - Verifique se os campos obrigatórios (Caminho, Tipo, Elemento) estão preenchidos
  - Verifique se o formato JSON está correto
  - Verifique o registro de atividades para detalhes do erro
  - Revise os logs do CGRateS

Nenhuma sessão ativa sendo exibida

- **Causa:** Nenhuma sessão atualmente ativa para este serviço
- **Solução:**
  - Isso é normal se o serviço não estiver em uso
  - Tente atualizar após iniciar uma sessão (chamada, dados, etc.)
  - Verifique se o UUID do serviço corresponde à conta no CGRateS

Detalhes da sessão não atualizando em tempo real

- **Causa:** Os dados da sessão são buscados sob demanda, não ao vivo
- **Solução:** Clique no botão "Atualizar" para obter os dados mais recentes da sessão

Registro de atividades não mostrando mudanças do CGRateS

- **Causa:** Apenas operações que não são GET são registradas (leituras não são registradas)
- **Solução:** Isso é intencional - apenas gravações/modificações criam entradas de atividade

Fluxos de Trabalho Comuns

Fluxo de Trabalho 1: Encontrar Serviços do Cliente

1. Clique no **filtro de Tipo de Serviço** (opcional)
2. Clique no **filtro de Cliente**
3. Digite o nome do cliente
4. Selecione o cliente no dropdown
5. Revise os serviços do cliente

Fluxo de Trabalho 2: Identificar Serviços que Estão Prestes a Expirar

1. Clique na aba **"Ativo"**
2. Ordene por **"Data de Término"** (crescente)
3. Serviços que expiram primeiro aparecem primeiro
4. Contate os clientes para renovação

Fluxo de Trabalho 3: Limpar Serviços de Teste

1. Pesquise por "teste" na caixa de pesquisa
2. Revise os resultados para confirmar que são dados de teste
3. Selecione todos os serviços de teste
4. Clique em **"Excluir Selecionados"**
5. Confirme a exclusão

Fluxo de Trabalho 4: Suspender Cliente Não Pagante

1. Navegue até a conta do cliente
2. Veja a aba de serviços
3. Selecione todos os serviços ativos
4. Altere o status para "Suspenso"
5. Salve as alterações

Fluxo de Trabalho 5: Visualizar Uso do Serviço Móvel

1. Filtre por **Tipo de Serviço: Móvel**
2. Clique no nome do serviço para abrir os detalhes
3. Clique na aba **"Uso"**
4. Revise o consumo de dados/voz/SMS
5. Identifique usuários pesados ou excessos

Resolução de Problemas

Serviços não aparecendo na lista

- **Causa:** Filtro de status ocultando resultados
- **Solução:** Clique na aba "Todos os Resultados" para mostrar todos os status

Não é possível encontrar serviço pela pesquisa

- **Causa:** O termo de pesquisa não corresponde aos dados armazenados
- **Solução:**
  - Tente uma pesquisa parcial (por exemplo, "0770" em vez do número completo)
  - Use o filtro de cliente em vez disso
  - Verifique se há erros de digitação

Botão de exclusão em massa desativado

- **Causa:** Nenhum serviço selecionado
- **Solução:** Marque as caixas ao lado dos serviços que deseja excluir

Ordenação não funcionando

- **Causa:** Coluna não ordenável
- **Solução:** Apenas colunas com ícones ▲▼ são ordenáveis

Página carrega lentamente

- **Causa:** Muitos serviços para exibir
- **Solução:**
  - Aplique filtros para reduzir o conjunto de resultados
  - Reduza itens por página
  - Use a pesquisa para restringir resultados

Documentação Relacionada

- `csa_add_service` - Adicionando novos serviços
- `csa_modify` - Modificando serviços e adicionando adicionais
- `concepts_products_and_services` - Conceitos de produtos e serviços
- `basics_customers` - Gerenciamento de clientes

Uso do Serviço e Monitoramento de Saldo

O sistema de Uso do Serviço fornece **monitoramento em tempo real** do consumo dos clientes para dados, voz, SMS e saldos monetários. Este recurso integra-se com o OCS (Sistema de Cobrança Online) para exibir o uso atual, as permissões restantes e as informações de expiração do saldo tanto para os clientes (via `Self-Care Portal <self_care_portal>`) quanto para a equipe.

Visão Geral

O monitoramento de uso permite:

- **Exibição de Saldo em Tempo Real** - Visualizar o uso atual e as permissões restantes
- **Vários Tipos de Saldo** - Rastrear dados, voz, SMS e saldos monetários simultaneamente
- **Monitoramento de Expiração** - Ver quando os saldos expiram
- **Detalhamento do Saldo** - Visão detalhada dos diferentes buckets de saldo
- **Atualização Automática** - Atualizações de uso a cada 3 segundos automaticamente

Acessando o Uso do Serviço

Da Lista de Serviços:

1. Navegue até **Serviços → Lista de Serviços**
2. Clique no **menu de ações (⋮)** ao lado de um serviço
3. Selecione **"Uso"**

Dos Detalhes do Serviço:

1. Abra a página de detalhes de um serviço
2. Clique na aba **"Uso"**

Da Página do Cliente:

1. Abra a visão geral do cliente
2. Navegue até a aba **Serviços**
3. Clique em **"Ver Uso"** ao lado de qualquer serviço

O modal ou página de uso se abre mostrando dados de consumo em tempo real.

Exibição de Uso

A interface de uso mostra cartões de resumo e barras de progresso detalhadas para cada tipo de saldo.

Cartões de Resumo

A linha superior exibe cartões de visualização rápida para cada tipo de saldo:

Informações do Cartão:

- **Tipo de Saldo** - Ícone e rótulo (Dados, Voz, SMS, Monetário)
- **Quantidade Restante** - Saldo atual nas unidades apropriadas
- **Tempo de Expiração** - Dias/horas até o saldo expirar
- **Botão Mais Informações** - Clique para expandir o detalhamento

Barras de Progresso

Abaixo dos cartões, as barras de progresso mostram o consumo visualmente, com porções preenchidas indicando o saldo restante.

Características da Barra de Progresso:

- **Indicador Visual** - Porção preenchida mostra o saldo restante
- **Porcentagem** - Porcentagem numérica do saldo restante
- **Valores Absolutos** - Mostra usado vs. total (por exemplo, "12.5GB / 20GB")
- **Codificação de Cores**:
  - Verde: >50% restante
  - Amarelo: 20-50% restante

    - Vermelho: <20% restante
- **Clicável** - Clique para expandir o detalhamento

Tipos de Saldo

Saldo de Dados

Rastreia o consumo de dados da internet.

Unidades: Gigabytes (GB) ou Megabytes (MB)

Formato de Exibição:

> Progresso: 12.5GB / 20GB (62%)

Cenários Comuns:

- **Planos de dados móveis** - 5GB, 10GB, 20GB de permissões mensais
- **Wireless fixo** - Ilimitado ou limitado a altos valores (500GB, 1TB)
- **Recargas** - Dados adicionais comprados no meio do ciclo
- **Serviços de dongle** - Dados pré-pagos para dispositivos hotspot

Múltiplos Buckets:

Os serviços frequentemente têm múltiplos saldos de dados:

- Permissão mensal (expira mensalmente)
- Dados bônus (expira após o período da campanha)
- Dados de recarga (expiração mais curta, consumidos primeiro)

Saldo de Voz

Rastreia minutos de chamadas telefônicas.

Unidades: Minutos (min)

Formato de Exibição:

> Progresso: 125 min / 500 min (25%)

Rastreamento de Duração de Chamadas:

- Chamadas recebidas (se cobradas)
- Chamadas realizadas
- Chamadas internacionais (bucket separado, se aplicável)
- Números premium

Cálculo:

O uso de voz é calculado pela duração da chamada em nanosegundos internamente, convertido em minutos para exibição.

Saldo de SMS

Rastreia o uso de mensagens de texto.

Unidades: Mensagens (msgs)

Formato de Exibição:

> Progresso: 45 / 250 (18%)

Tipos de Mensagens:

- SMS padrão (160 caracteres)
- SMS longo (múltiplos segmentos)
- MMS (se rastreado separadamente)

Saldo Monetário

Rastreia crédito pré-pago ou saldo da conta.

Unidades: Moeda (£, $, €, etc.)

Formato de Exibição:

> Progresso: £15.50 / £20.00 (77%)

Uso:

- Contas pré-pagas usam saldo monetário para pagar pelo uso
- O crédito diminui à medida que o cliente usa os serviços
- Pode ser recarregado via pagamento ou voucher
- Pode expirar se não utilizado dentro do período de validade

Detalhamento do Saldo

Clique em **"Mais Informações"** em qualquer cartão ou clique em uma barra de progresso para expandir o detalhamento.

Visão Expandida:

> □ Permissão Mensal 20GB
> Restante: 12.5 GB Usado: 7.5 GB Expira: 25 Jan 2025 (15 dias) Peso: 10
>
> □ Dados Bônus 5GB
> Restante: 5.0 GB Usado: 0 GB Expira: 31 Jan 2025 (21 dias) Peso: 20
>
> □ Dados de Recarga 3GB
> Restante: 0 GB Usado: 3.0 GB Expira: 18 Jan 2025 (EXPIRED) Peso: 30
>
> Total Restante: 17.5 GB

Campos do Bucket de Saldo:

- **ID/Nome** - Identificador para o bucket de saldo
- **Restante** - Quantidade restante neste bucket específico
- **Usado** - Quantidade consumida deste bucket
- **Data de Expiração** - Quando este saldo expira
- **Peso** - Ordem de prioridade (peso mais alto consumido primeiro)

Sistema de Peso

Os saldos têm um valor de **peso** que determina a ordem de consumo:

- **Peso mais alto = consumido primeiro**

- **Peso mais baixo = consumido por último**

Exemplo de Pesos:

- Dados de recarga: Peso 30 (consumido primeiro, expiração mais curta)
- Dados bônus: Peso 20 (consumido em segundo)
- Permissão mensal: Peso 10 (consumido por último, expiração mais longa)

Isso garante que os saldos que estão prestes a expirar sejam usados antes dos que duram mais.

Para detalhes completos sobre as regras de consumo de saldo, estratégias de peso e como o CGRateS prioriza os saldos, veja Ações e Comportamentos de Recarga do CGRateS.

Atualizações em Tempo Real

Os dados de uso são atualizados automaticamente a cada **3 segundos** via polling.

O que é Atualizado:

- Quantidades atuais de saldo
- Barras de progresso de uso
- Temporizadores de expiração
- Detalhes de buckets individuais

Experiência do Usuário:

- Nenhum recarregamento de página necessário
- Atualizações suaves sem flicker
- Sobreposição de carregamento durante a atualização
- Insígnia de status mostra o estado atual do serviço

Casos de Uso:

- Monitorar o uso do cliente durante a chamada

- Observar a diminuição do saldo em tempo real à medida que o cliente usa o serviço
- Verificar a recarga imediatamente após a compra

Uso em Diferentes Tipos de Serviço

Serviços Móveis

Exibe todos os quatro tipos de saldo:

- Dados (GB)
- Voz (minutos)
- SMS (mensagens)
- Monetário (moeda)

Exemplo:

> DADOS: 12.5GB restantes VOZ: 125 min restantes SMS: 45 msgs restantes MONETÁRIO: £15.50 restantes

Wireless Fixo / Internet

Normalmente mostra apenas:

- Dados (GB ou TB)
- Monetário (se pré-pago)

Exemplo:

> DADOS: 450GB / 500GB restantes MONETÁRIO: £45.00 (crédito pré-pago)

Serviços de Hotspot / Dongle

Mostra rastreamento de dados específico de dongle:

- Dados (consumidos vs. pré-pagos)
- Monetário (saldo pré-pago)

Modo de Exibição:

Quando `dongle=true`, o componente oculta voz e SMS, mostrando apenas dados e saldos monetários relevantes.

Solução de Problemas

Uso mostrando como 0 / 0

- **Causa:** Serviço não integrado com OCS ou CGRateS
- **Solução:**
  - Verifique se o serviço está provisionado no OCS
  - Verifique a conectividade da API do OCS
  - Revise o mapeamento do UUID do serviço

Uso não atualizando

- **Causa:** Polling parado ou OCS inacessível
- **Solução:**
  - Atualize a página
  - Verifique o console do navegador para erros
  - Verifique se a API do OCS está online

Saldos mostram quantidades incorretas

- **Causa:** Desvio de dados do OCS ou problema de cache
- **Solução:**
  - Forçar atualização do saldo do OCS
  - Verifique se há transações pendentes
  - Verifique a configuração do OCS

Datas de expiração ausentes

- **Causa:** Saldo não tem expiração definida
- **Solução:**
  - Alguns saldos são definidos para nunca expirar (validade ilimitada)
  - Verifique a configuração do saldo no OCS

Múltiplos saldos confusos

- **Causa:** Múltiplas recargas ou dados bônus adicionados
- **Solução:**
  - Use a visão detalhada para ver todos os buckets
  - Classifique por peso para ver a ordem de consumo
  - Revise as datas de expiração individuais

Integração com OCS/CGRateS

Os dados de uso vêm do **OCS (Sistema de Cobrança Online)**, tipicamente CGRateS. Para informações abrangentes sobre como os saldos funcionam, incluindo tipos de saldo, regras de consumo, comportamento de rollover e configurações de Ação, veja Ações e Comportamentos de Recarga do CGRateS.

Fluxo de Dados:

1. O usuário abre a visualização de uso
2. OmniCRM chama `GET /crm/service/{service_id}`
3. A API consulta o OCS via UUID do serviço
4. O OCS retorna o mapa de saldo:

```
{
  "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. A UI converte valores para unidades de exibição (bytes → GB, nanosegundos → minutos)
6. Barras de progresso e cartões renderizados
7. O polling continua a cada 3 segundos

Mapeamento de Tipos de Saldo do OCS:

O OCS retorna dados de saldo com prefixos de tipo que mapeiam para a exibição na UI:

- `*data` → **CARTÃO DE DADOS** (uso da internet)
- `*voice` → **CARTÃO DE VOZ** (minutos de chamada)
- `*sms` → **CARTÃO DE SMS** (mensagens de texto)
- `*monetary` → **CARTÃO MONETÁRIO** (crédito pré-pago)

Cada tipo de saldo pode ter múltiplos buckets (por exemplo, permissão mensal + dados bônus + dados de recarga), todos exibidos na visão detalhada.

Conversões de Valores de Saldo:

- **Dados:** Bytes → GB (dividir por 1024^3)
- **Voz:** Nanosegundos → Minutos (dividir por 60×10^9)
- **SMS:** Contagem (sem conversão)
- **Monetário:** Centavos → Moeda (dividir por 100)

Auto-Renovação e Planos de Ação

Serviços com auto-renovação ativada têm **Planos de Ação** agendados no OCS. Para informações detalhadas sobre como as Ações funcionam, manipulação de saldo e configuração de Planos de Ação, veja Ações e Comportamentos de Recarga do CGRateS.

O que são Planos de Ação?

Os Planos de Ação são tarefas agendadas no CGRateS que são executadas automaticamente em horários específicos para:

- Adicionar saldo a uma conta (recarga automática)
- Renovar permissões mensais
- Aplicar cobranças recorrentes
- Expirar saldos antigos

Como Funciona a Auto-Renovação:

1. **Provisionamento do Serviço:**
    - Quando o serviço é criado com `auto_renew = true`
    - O playbook de provisionamento cria um Plano de Ação no OCS
    - O Plano de Ação é configurado para ser executado mensalmente (ou por ciclo de faturamento)
2. **Configuração do Plano de Ação:**

    O Plano de Ação contém:

    - **ID da Conta** - UUID do serviço
    - **Ações** - O que fazer (adicionar dados, voz, SMS, saldo monetário)
    - **Agendamento** - Quando executar (por exemplo, mensalmente no dia 1 às 00:00 UTC)
    - **Quantidade** - Quanto saldo adicionar
3. **Execução Automática:**
    - O OCS executa o Plano de Ação no horário agendado
    - Adiciona saldo à conta (por exemplo, 20GB de dados, 500 minutos de voz)
    - Define a data de expiração para o novo saldo (por exemplo, 30 dias)
    - O cliente é cobrado via método de pagamento registrado
4. **Visualizando Planos de Ação:**
    - Navegue até os detalhes do serviço na visualização do OCS
    - Planos de Ação listados com o próximo horário de execução
    - Mostra: Nome do plano, data da próxima execução, detalhes da ação

Exemplo de Plano de Ação:

```
{
  "ActionPlanId":
"ProductID_MonthlyPlan__ProductName_20GB_Mobile__ActionPlan_Monthly_F
  "NextExecTime": "2025-02-01T00:00:00+00:00",
  "ActionName_hr": "Renovação Mensal",
  "PlanName": "20GB Móvel",
  "ActionFrequency_hr": "Todo Plano Mensal",
  "custom_NextExecTime_hr": "em 22 dias"
}
```


Gerenciando Auto-Renovação:

- **Ativar** - Definido durante a criação ou modificação do serviço
- **Desativar** - Remover Plano de Ação do OCS (o serviço mantém o saldo existente, mas não será auto-renovado)
- **Modificar** - Alterar a quantidade ou frequência de renovação via modificação do serviço

Renovação Manual:

Se a auto-renovação estiver desativada, o cliente deve manualmente:

- Recarregar antes que o saldo expire
- Ou o serviço será suspenso quando o saldo se esgotar

Visualizando na UI:

A aba de serviços mostra o status da auto-renovação:

> Próxima Renovação: 1 Fev 2025 (em 22 dias) Quantidade de Renovação: £15.00

Melhores Práticas

Para a Equipe de Suporte:

- Verifique o uso antes de responder a chamadas "Por que meu serviço está lento?"

- Verifique o saldo após recargas para confirmar o sucesso
- Use o detalhamento para identificar buckets expirados
- Monitore clientes de alto uso para evitar excessos

Para Clientes (Autoatendimento):

- Verifique o uso regularmente para evitar ficar sem saldo
- Recarregue antes que o saldo expire
- Entenda o sistema de peso para saber qual saldo é consumido primeiro
- Entre em contato com o suporte se o uso parecer incorreto

Para Administradores:

- Configure as expirações de saldo apropriadas
- Defina valores de peso para priorizar saldos que estão prestes a expirar
- Monitore a conectividade do OCS para relatórios precisos
- Revise as configurações de saldo para corresponder às ofertas de produtos

Documentação Relacionada

- Ações e Comportamentos de Recarga do CGRateS - Guia completo sobre tipos de saldo, regras de consumo, pesos, comportamento de rollover e configurações de Ação
- Destinos do CGRateS - Configuração de destino para restrições de saldo geográficas e baseadas em PLMN
- `features_topup_recharge` - Sistema de recarga para adicionar saldo
- `csa_service_management` - Gerenciamento de serviços
- `csa_modify` - Adicionando complementos para aumentar permissões
- `concepts_products_and_services` - Configuração de produtos

Atendimento ao Cliente

Impersonação de Usuário para Suporte e Solução de Problemas

A impersonação permite que funcionários autorizados façam login temporariamente como outro usuário para solucionar problemas, verificar configurações ou ver exatamente o que o usuário está experimentando. Este recurso é essencial para o suporte ao cliente, mas requer permissões apropriadas e é totalmente auditado.

Ao impersonar um cliente, a equipe acessa o `Self-Care Portal <self_care_portal>` exatamente como o cliente o vê, permitindo uma solução de problemas e suporte precisos.

Veja também: `RBAC <rbac>` para configuração de permissões, `Customers <basics_customers>` para gerenciamento de clientes, `Self-Care Portal <self_care_portal>` para recursos do portal do cliente.

Propósito

A impersonação de usuário fornece:

1. **Solução de Problemas** — Veja exatamente o que o cliente vê para diagnosticar problemas
2. **Verificação** — Confirme se as configurações de serviço e permissões funcionam corretamente
3. **Treinamento** — Demonstre recursos da perspectiva do cliente
4. **Suporte** — Ajude os clientes a navegar no sistema sem exigir compartilhamento de tela
5. **Registro de Auditoria** — Todas as sessões de impersonação são registradas para segurança e conformidade

Permissões Necessárias

Para impersonar usuários, você deve ter uma das seguintes permissões:

- `can_impersonate` — Permissão de impersonação dedicada para a equipe de suporte
- `admin` — Acesso administrativo completo (inclui direitos de impersonação)

Usuários sem essas permissões não podem acessar o recurso de impersonação.

Como Impersonar um Usuário

Via Web UI:

1. **Navegar até o Cliente** — Encontre o cliente no CRM
2. **Selecionar Contatos** — Veja a lista de contatos do cliente
3. **Clique em "Login como Usuário"** — O botão aparece ao lado de cada contato que possui uma conta de usuário
4. **Confirmar Impersonação** — O sistema pode solicitar confirmação
5. **Sessão Iniciada** — Você agora está logado como esse usuário

Via API:

Iniciar sessão de impersonação:

Endpoint: `POST /auth/impersonate`

Permissão Necessária: `can_impersonate` ou `admin`

Requisição:

```
{
  "user_id": 42
}
```

Resposta:

```
{
  "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"
}
```

Os tokens retornados são para a sessão do usuário impersonado.

O Que Acontece Durante a Impersonação

Quando você impersona um usuário:

- **Contexto Completo** — Você vê o sistema exatamente como o usuário alvo vê:
  - Seu painel e navegação
  - Seus dados de cliente (se usuário cliente)
  - Suas permissões e controles de acesso
  - Seus serviços, faturas e uso

- **Rastreamento de Sessão** — O sistema rastreia ambas as identidades:
  - `impersonating_user_id` — Seu ID de usuário real
  - `target_user_id` — O usuário que você está impersonando
  - `impersonation_start` — Quando a impersonação começou
- **Indicador** — A interface exibe um banner mostrando que você está impersonando alguém:
  - "Você está atualmente logado como [Nome de Usuário]"
  - "Clique aqui para parar a impersonação"
- **Registro de Auditoria** — Todas as ações são registradas com ambos os IDs de usuário:
  - Ações aparecem como realizadas pelo usuário alvo
  - Registros de auditoria registram quem realmente as realizou (usuário impersonador)
  - Registro de auditoria completo mantido na tabela `ImpersonationLog`

Parando a Impersonação

Via Web UI:

1. **Clique no Banner** — Clique no banner de impersonação na parte superior da página
2. **Ou Navegue** — Vá ao menu do usuário e selecione "Parar Impersonação"
3. **Confirmação** — A sessão termina e você retorna à sua própria conta

Via API:

Endpoint: `POST /auth/stop_impersonation`

Requisição: Nenhum corpo necessário (requisição autenticada)

Resposta:

```
{
  "success": true,
  "impersonation_end": "2025-01-04T15:45:00Z",
  "duration_seconds": 900,
  "access_token": "your_original_token",
  "refresh_token": "your_original_refresh_token"
}
```

Sua sessão original é restaurada.

Registro de Auditoria de Impersonação

Todas as sessões de impersonação são registradas na tabela `ImpersonationLog` com:

- **impersonating_user_id** — Membro da equipe que realizou a impersonação
- **target_user_id** — Cliente ou usuário que foi impersonado
- **impersonation_start** — Timestamp de início
- **impersonation_end** — Timestamp de término (quando a sessão foi parada)
- **impersonation_duration** — Duração em segundos

Isso fornece total responsabilidade por todas as sessões de impersonação e permite:

- **Auditorias de segurança** — Revisar quem impersonou quem e quando
- **Relatórios de conformidade** — Demonstrar uso adequado de acesso elevado
- **Investigação** — Rastrear ações durante sessões de impersonação
- **Monitoramento** — Identificar padrões incomuns de impersonação

Visualizando Registros de Impersonação:

Administradores podem consultar registros de impersonação via API:

`GET /auth/impersonation_logs?user_id={user_id}&start_date={date}&end_date={date}`

Filtrar por:

- Usuário impersonador (quem realizou a impersonação)
- Usuário alvo (quem foi impersonado)
- Intervalo de datas
- Duração

Melhores Práticas

1. **Minimizar Duração** — Impersonar apenas pelo tempo necessário para resolver o problema
2. **Documentar Propósito** — Anote por que a impersonação foi necessária no registro de atividades do cliente
3. **Informar Clientes** — Deixe os clientes saberem que você pode precisar ver a conta deles (política de privacidade)
4. **Verificar Identidade** — Confirme a identidade do usuário antes de impersonar via conta dele
5. **Revisar Registros** — Audite regularmente os registros de impersonação em busca de padrões incomuns
6. **Limitar Permissões** — Conceda `can_impersonate` apenas à equipe de suporte que precisa
7. **Treinamento** — Assegure-se de que a equipe entenda a responsabilidade e as implicações de auditoria

Considerações de Segurança

- **Acesso Completo** — A impersonação concede acesso completo à conta do usuário alvo
- **Nenhuma Senha Necessária** — A impersonação contorna a autenticação (apenas baseada em permissões)

- **Ações Registradas** — Todas as ações durante a impersonação são atribuídas ao usuário alvo nos registros do aplicativo (mas os registros de auditoria mostram o verdadeiro ator)
- **Isolamento de Sessão** — A impersonação cria uma nova sessão; não afeta as sessões ativas do usuário alvo
- **Limites de Tempo** — As sessões de impersonação devem ter limite de tempo (configurável)
- **Contorno de MFA** — A impersonação contorna os requisitos de 2FA (usa a autenticação do impersonador)

Restrições

- **Não é Possível Impersonar Administradores** — Dependendo da configuração, pode não ser possível impersonar outros administradores
- **Mesmas Permissões** — Você obtém as permissões do usuário alvo, não a união das permissões de ambos os usuários
- **Limites de Sessão** — Apenas uma sessão de impersonação por membro da equipe por vez
- **Requisitos de Auditoria** — Não é possível desativar ou ocultar o registro de impersonação

Solucionando Problemas Comuns

Problema: O botão "Login como Usuário" não aparece

- Solução: Verifique se você tem a permissão `can_impersonate` ou `admin`
- Solução: Confirme se o contato tem uma conta de usuário vinculada

Problema: A impersonação falha com erro de permissão

- Solução: Verifique se o usuário alvo é um administrador (pode estar restrito)
- Solução: Verifique se sua permissão de impersonação está ativa

Problema: Não é possível parar a impersonação

- Solução: Use o endpoint da API para parar a impersonação: `POST /auth/stop_impersonation`
- Solução: Limpe os cookies do navegador e faça login novamente com suas credenciais

Problema: Ações não estão sendo registradas corretamente

- Solução: Verifique se a sessão de impersonação está ativa (verifique o banner)
- Solução: Revise os registros de auditoria - as ações são registradas com ambos os IDs de usuário

Definindo Produtos e Configuração do CGRateS

Este guia explica como definir produtos no OmniCRM juntamente com sua configuração de cobrança CGRateS associada.

Produtos e suas Ações CGRateS são tipicamente definidos juntos em scripts Python que configuram tanto o CRM quanto o OCS (Sistema de Cobrança Online).

Visão Geral

Definir um produto completo envolve configuração em **dois sistemas**:

1. **CGRateS/OCS** - Definir o comportamento de cobrança (Ações, Tarifas, Destinos)
2. **OmniCRM** - Definir os metadados do produto (preço, nome, características)

Esses sistemas estão intencionalmente ligados de forma solta, permitindo que você interaja com o CGRateS através dos Plays do Ansible que você define.

Importante: Capacidades do CGRateS

CGRateS é um motor de tarifação extremamente poderoso e flexível. Este guia mostra **um padrão comum de configuração** para produtos do OmniCRM, mas o CGRateS pode fazer muito mais:

- **Cenários de tarifação avançados**: Tarifas por hora do dia, níveis de clientes, roteamento baseado na qualidade da chamada, cobrança por explosão

- **Lógica de roaming complexa**: Tarifas diferentes para chamadas/SMS MO vs MT durante o roaming, preços específicos de rede
- **Suporte a protocolos legados**: CAMEL para roaming 2G/3G (SMS MO-MT, USSD), Diameter para LTE/5G
- **Cobrança sofisticada**: Cobrança baseada em reserva, controle de crédito, fallback em múltiplos níveis
- **Roteamento dinâmico**: Roteamento baseado em custo, seleção de fornecedores, LCR (Roteamento de Menor Custo)

Este é apenas o guia do CRM - ele foca em padrões simples de definição de produtos que funcionam bem com o sistema de provisionamento do OmniCRM. Para configurações avançadas do CGRateS, consulte a documentação do CGRateS ou veja Ações CGRateS e Comportamentos de Recarga para abordagens de saldo e estratégias de tarifação.

Fluxo de Trabalho Completo para Definição de Produtos

Passo 1: Autenticar na API do CRM

```
import requests
import json

crm_url = 'https://crm.example.com/'
session = requests.Session()

headers = {
    "Content-Type": "application/json"
}

# Obter token de autenticação
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("Autenticação no CRM OK")
```

Passo 2: Definir Modelos de Inventário (Opcional)

Modelos de inventário definem os tipos de itens físicos (cartões SIM, roteadores, etc.) que podem ser atribuídos durante o provisionamento.

```
inventory_list_new = []

# Definir modelo de inventário do Cartão SIM
inventory_list_new.append({
    "item": "Cartão SIM",
    "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"
})

# Definir modelo de inventário do Número Móvel
inventory_list_new.append({
    "item": "Número Móvel",
    "itemtext1_label": "Número Móvel E.164",
    "itemtext2_label": "Tipo",
    "wholesale_cost": 0.2,
    "retail_cost": 3,
    "icon": "fa-solid fa-phone"
})

# Verificar se o modelo de inventário existe, criar ou atualizar
for inventory_template in inventory_list_new:
    # Obter modelos existentes
    inventory_list_existing = session.get(
        crm_url + '/crm/inventory/item_template/',
        headers=headers
    )

    # Verificar se o modelo já existe
    if inventory_template['item'] in [x['item'] for x in
inventory_list_existing.json()]:
        # Atualizar modelo existente
        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"Modelo de inventário atualizado:
{inventory_template['item']}")
    else:
        # Criar novo modelo
        response = session.put(
            crm_url + '/crm/inventory/item_template',
            json=inventory_template,
            headers=headers
        )
        assert response.status_code == 200
        print(f"Modelo de inventário criado:
{inventory_template['item']}")
```

Campos do Modelo de Inventário:

- **item** (obrigatório) - Nome do modelo
- **itemtext1_label** - Rótulo para o primeiro campo personalizado
- **itemtext2_label** - Rótulo para o segundo campo personalizado
- **wholesale_cost** - Seu custo para este item
- **retail_cost** - Custo para o cliente
- **allow_dropdown_staff** - Mostrar nos dropdowns de funcionários
- **allow_dropdown_customer** - Mostrar nos dropdowns de clientes
- **icon** - Classe de ícone do Font Awesome

Passo 3: Conectar ao CGRateS

```
import cgrateshttpapi
import time

OCS_Obj = cgrateshttpapi.CGRateS("ocs.example.com", "2080")

tenant = "your_tenant_name"
tpid = str(tenant) + "_" + str(int(time.time()))
```

Passo 4: Definir Destinos no CGRateS

Destinos definem ONDE os saldos podem ser usados. Eles devem ser definidos antes de criar Ações que os referenciam.

Existem dois tipos:

1. **Destinos Geográficos** - Prefixos numéricos para voz/SMS PARA lugares específicos (por exemplo, `Dest_AU_Mobile`, `Dest_International_UK`)
2. **Destinos PLMN** - Códigos de rede para uso de dados e roaming (por exemplo, `Dest_PLMN_OnNet`, `Dest_PLMN_US_Verizon`)

Regra Crítica:

- **Saldos de Voz/SMS** → Use destinos geográficos (o número sendo chamado PARA)
- **Saldos de Dados** → Use destinos PLMN (a rede à qual o cliente está conectado DE)

Para exemplos completos de definição de destinos incluindo:

- Destinos geográficos (móvel, linha fixa, gratuito, internacional)
- Destinos PLMN (on-net, roaming, zonas)
- Regras de formato PLMN (`mccXXX.mncYYY`)
- Quando usar cada abordagem

Veja: Ações CGRateS - Definindo Destinos

Passo 5: Definir Ações CGRateS

Ações são o mecanismo para adicionar saldos às contas dos clientes. Antes que você possa vincular um produto do CRM a uma Ação, a Ação deve ser definida no CGRateS.

Crítico: Ações são definidas durante a configuração inicial do sistema (não durante o provisionamento). Elas especificam quais saldos adicionar, tempos de expiração, comportamento de rollover, etc.

Veja Ações CGRateS e Comportamentos de Recarga para documentação completa sobre:

- Como definir Ações usando Python (`OCS_Obj.SendData`)
- Entendendo `*topup` vs `*topup_reset` (comportamento de rollover)
- Cálculos de unidades (bytes para dados, nanosegundos para voz, contagem para SMS)
- Referência de campos (BalanceId, BalanceType, DestinationIDs, Units, ExpiryTime, Weight)
- Exemplos completos de trabalho (planos de múltiplos saldos, complementos de dados, complementos de voz)
- Regras de consumo de saldo e prioridade

Referência Rápida de Rollover:

- **`*topup`** - Adiciona ao saldo existente (habilita rollover). Para: complementos pré-pagos, pacotes de dados
- **`*topup_reset`** - Substitui o saldo existente (sem rollover). Para: planos mensais com alocações fixas

Passo 6: Criar Produtos no CRM

Agora que as Ações estão definidas no CGRateS, crie os produtos correspondentes no CRM.

Exemplo: Serviço SIM Independente

```
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": "SIM Móvel Apenas",
    "comment": "Cartão SIM físico para uso com Telefones Móveis",
    "provisioning_json_vars": "{\"iccid\" : \"\", \"msisdn\" :
\"\"}",
    "terms": "Deve ser ativado dentro de 6 meses. Todo crédito
perdido se o serviço não for utilizado por 12 meses.",
    "service_type": "mobile",
    "residential": True,
    "business": True,
    "enabled": True,
    "inventory_items_list": "['Cartão SIM', 'Número Móvel']",
    "icon": "fa-solid fa-sim-card",
    "features_list": "['Número de Telefone Australiano (61xxx)',
'Velocidades mais rápidas', 'Melhor cobertura']",
    "wholesale_cost": 3,
    "wholesale_setup_cost": 1,
    "customer_can_purchase": True
})
```

Exemplo: Complemento de Plano Mensal

```
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",  # Vincula a Action_au-premium-plan-1
    "product_name": "Plano Premium AU 1",
    "comment": "100GB de Dados na Austrália, 3000 Minutos na Austrália, 3000 SMS na Austrália, 6GB de Dados de Roaming",
    "provisioning_json_vars": "",
    "terms": "Plano pós-pago. Renova automaticamente a cada 30 dias.",
    "service_type": "mobile",
    "residential": True,
    "business": True,
    "enabled": True,
    "icon": "fa-solid fa-mobile",
    "features_list": "['100GB de Dados na Austrália', '3000 Minutos na Austrália', '3000 SMS na Austrália', '6GB de Dados de Roaming']",
    "wholesale_cost": 45.00,
    "wholesale_setup_cost": 0,
    "auto_renew": "True",
    "customer_can_purchase": True
})
```

Exemplo: Complemento de Recarga Única

```
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",  # Vincula a Action_au-data-addon-20gb
    "product_name": "Complemento de Dados AU 20GB",
    "comment": "20GB de Dados na Austrália",
    "provisioning_json_vars": "",
    "terms": "Recarga pré-paga. Cobrança imediata.",
    "service_type": "mobile",
    "residential": True,
    "business": True,
    "enabled": True,
    "icon": "fa-solid fa-mobile",
    "features_list": "['20GB de Dados na Austrália']",
    "wholesale_cost": 22,
    "wholesale_setup_cost": 0,
    "auto_renew": "False",
    "customer_can_purchase": True
})
```

Descrições dos Campos do Produto:

- **category** (obrigatório) - Categoria do produto:
  - `standalone` - Serviço completo (cria novo registro de serviço)
  - `addon` - Adicionado a um serviço existente
  - `bundle` - Pacote de múltiplos serviços
- **provisioning_play** (obrigatório) - Playbook do Ansible a ser executado (por exemplo, `play_topup_charge_then_action`, `play_simple_service`)
- **relies_on_list** - Lista separada por vírgulas de product_slugs dos quais este produto depende
- **contract_days** - Duração do contrato em dias (0 = sem contrato, 30 = mensal)
- **retail_cost** (obrigatório) - Custo mensal/recorrente para o cliente
- **retail_setup_cost** - Taxa de configuração única cobrada ao cliente

- **product_slug** (obrigatório) - **Identificador único que vincula à Ação CGRateS** (convenção: `Action_{product_slug}`)
- **product_name** (obrigatório) - Nome de exibição
- **comment** - Descrição curta
- **provisioning_json_vars** - String JSON de variáveis passadas para o playbook
- **terms** - Texto de termos e condições
- **service_type** - Classificação do serviço (móvel, dongle, fixo, etc.)
- **residential** - Disponível para clientes residenciais
- **business** - Disponível para clientes empresariais
- **enabled** - Produto está ativo e pode ser adquirido
- **inventory_items_list** - String de lista Python de itens de inventário necessários
- **icon** - Classe de ícone do Font Awesome
- **features_list** - String de lista Python de características
- **wholesale_cost** - Seu custo recorrente
- **wholesale_setup_cost** - Seu custo único
- **auto_renew** - String "True" ou "False" para renovação automática
- **customer_can_purchase** - O cliente pode comprar através do portal de autoatendimento

Veja também: Ciclo de Vida do Produto

Passo 7: Criar ou Atualizar Produtos via API

```
# Obter produtos existentes
product_list_existing = session.get(crm_url + '/crm/product',
headers=headers)
existing_products = product_list_existing.json()

# Processar cada produto
for product in product_list_new:
    print(f"Processando produto: {product['product_slug']}")

    # Verificar se o produto já existe
    if product['product_slug'] in [x['product_slug'] for x in
existing_products]:
        print(f"Produto já existe: {product['product_slug']}")

        # Obter product_id do produto existente
        product_id = [
            x['product_id']
            for x in existing_products
            if x['product_slug'] == product['product_slug']
        ][0]

        product['product_id'] = product_id

        # Atualizar produto existente
        response = session.patch(
            crm_url + '/crm/product/' + str(product_id),
            json=product,
            headers=headers
        )
        print(f"Status: {response.status_code}")
        assert response.status_code == 200
        print(f"Produto atualizado: {product['product_slug']}")
    else:
        # Criar novo produto
        print(f"Criando novo produto: {product['product_slug']}")
        response = session.put(
            crm_url + '/crm/product',
            json=product,
            headers=headers
        )
        print(f"Status: {response.status_code}")
```

```
        assert response.status_code == 200
        print(f"Produto criado: {product['product_slug']}")
```

Vinculando Produtos a Ações

A ligação crítica entre produtos do CRM e Ações do CGRateS é a **convenção de nomenclatura**:

```
# No CGRateS
ActionsId = "Action_au-premium-plan-1"

# No CRM
product_slug = "au-premium-plan-1"

# No playbook (construído automaticamente)
cgr_action_name = "Action_" + product_slug
# Resultado: "Action_au-premium-plan-1"
```

Como os Playbooks Executam Ações

Nota: Usar `ExecuteAction` é apenas um padrão para adicionar saldos. Você também pode chamar diretamente `SetBalance` ou outras APIs do CGRateS dentro de um playbook. No entanto, `OmniCRM-API/Provisioners/plays/play_topup_charge_then_action.yaml` usa o padrão ExecuteAction, pois é uma abordagem comum e reutilizável que separa a lógica de saldo (Ações) da lógica de provisionamento (Playbooks).

Quando um produto é provisionado usando este padrão, o playbook constrói o nome da Ação a partir do product_slug:

```
# No play_topup_charge_then_action.yaml
- name: Definir fato cgr_action_name
  set_fact:
    cgr_action_name: "Action_{{
api_response_product.json.product_slug }}"

# Executar a ação
- name: Executar ação do 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 }}"
        }]
      }
```

Ao usar este padrão: A Ação **deve existir no CGRateS** antes que o playbook seja executado, caso contrário, a execução falhará com o erro "Ação não encontrada".

Veja também: Guia de Playbooks do Ansible

Exemplo Completo: Definindo um Novo Produto

Vamos passar pela criação de um novo produto completo "Pacote de Dados de 50GB":

1. Definir a Ação no CGRateS

Veja o exemplo Complemento de Dados Simples no Passo 5 acima para como definir uma Ação de complemento de dados. Para nosso exemplo de 50GB, a

Ação seguiria o mesmo padrão, mas com unidades diferentes:

- **ActionsId**: `"Action_50gb-data-pack"` (corresponde ao product_slug abaixo)
- **BalanceType**: `"*data"`
- **DestinationIDs**: `"Dest_PLMN_OnNet"` (para uso de dados on-net)
- **Units**: `50 * 1024 * 1024 * 1024` (50GB em bytes)
- **Identifier**: `"*topup"` (habilita rollover)

2. Criar o Produto no 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",  # Deve corresponder ao
ActionsId sem "Action_"
    "product_name": "Pacote de Dados de 50GB",
    "comment": "50GB de dados de alta velocidade válidos por 30
dias",
    "provisioning_json_vars": "",
    "terms": "Os dados expiram após 30 dias. Dados não utilizados
são acumulados se recarregados novamente antes da expiração.",
    "service_type": "mobile",
    "residential": True,
    "business": True,
    "enabled": True,
    "icon": "fa-solid fa-database",
    "features_list": "['50GB de Dados de Alta Velocidade',
'Validade de 30 Dias', 'Rollover Habilitado']",
    "wholesale_cost": 20.00,
    "wholesale_setup_cost": 0,
    "auto_renew": "False",
    "customer_can_purchase": True
}

# Criar no CRM
response = session.put(
    crm_url + '/crm/product',
    json=product_50gb_data,
    headers=headers
)
assert response.status_code == 200
print("Produto criado: 50gb-data-pack")
```

3. Verificar o Vínculo

O produto agora pode ser provisionado:

1. O cliente compra "Pacote de Dados de 50GB" (product_slug: `50gb-data-pack`)
2. O playbook `play_topup_charge_then_action` é executado
3. O playbook constrói o nome da Ação: `Action_50gb-data-pack`
4. O playbook chama `APIerSv1.ExecuteAction` com ActionsId `Action_50gb-data-pack`
5. O CGRateS encontra a Ação e a executa
6. O cliente recebe 50GB de saldo

Veja também: Cobrança e Pagamentos a partir de Playbooks

Padrões Comuns de Produtos

Esses padrões mostram configurações típicas de produtos do CRM e como eles se vinculam às Ações do CGRateS. Para as definições de Ação correspondentes, veja Ações CGRateS e Comportamentos de Recarga.

Padrão 1: Plano Mensal de Múltiplos Saldos

Caso de Uso: Plano mensal abrangente com dados, voz, SMS, roaming

Produto CRM:

```
{
    "product_slug": "premium-monthly-plan",  # Vincula a
Action_premium-monthly-plan
    "product_name": "Plano Mensal Premium",
    "category": "addon",
    "retail_cost": 59.00,
    "auto_renew": "True",
    "contract_days": 30,
    "provisioning_play": "play_topup_charge_then_action",
    "features_list": "['100GB de Dados', '5000 Minutos', '5000
SMS', 'Roaming Incluído']"
}
```

Ação: Usa `*topup_reset` com múltiplos tipos de saldo. Veja Exemplo 1: Plano Mensal de Múltiplos Saldos (Python).

Padrão 2: Complemento de Dados Simples (Rollover)

Caso de Uso: Compra única de dados que acumula se recarregada novamente

Produto CRM:

```
{
    "product_slug": "10gb-addon",  # Vincula a Action_10gb-addon
    "product_name": "Complemento de Dados de 10GB",
    "category": "addon",
    "retail_cost": 15.00,
    "auto_renew": "False",  # Compra única
    "contract_days": 0,
    "provisioning_play": "play_topup_charge_then_action",
    "features_list": "['10GB de Dados com Rollover', 'Validade de
30 Dias']"
}
```

Ação: Usa `*topup` para comportamento de rollover. Veja Ações CGRateS - Rollover.

Padrão 3: Plano Mensal Fixo (Sem Rollover)

Caso de Uso: Plano mensal que sempre reinicia exatamente a mesma quantia

Produto CRM:

```
{
    "product_slug": "fixed-50gb-monthly",  # Vincula a
Action_fixed-50gb-monthly
    "product_name": "Mensal Fixo de 50GB",
    "category": "addon",
    "retail_cost": 35.00,
    "auto_renew": "True",
    "contract_days": 30,
    "provisioning_play": "play_topup_charge_then_action",
    "features_list": "['50GB de Dados', 'Reinicia Mensalmente',
'Sem Rollover']"
}
```

Ação: Usa `*topup_reset` para evitar rollover. Veja Ações CGRateS - Comportamento de Reset.

Avançado: Carregadores e Planos de Tarifação

Importante: Esta seção cobre tarifação básica pay-per-use. O CGRateS suporta cenários de tarifação incrivelmente sofisticados, incluindo:

- **Tarifas diferenciadas de roaming**: Preços diferentes para chamadas/SMS MO vs MT durante o roaming
- **Cobrança específica de protocolo**: CAMEL para 2G/3G (SMS MO-MT, USSD), Diameter para 4G/5G
- **Preços contextuais**: Tarifas baseadas em tempo, localização, segmento de cliente, qualidade da rede
- **Tarifação multidimensional**: Combinando tipo de rede, status de roaming, destino e hora do dia

Os exemplos abaixo mostram configurações PAYG simples e de tarifa fixa adequadas para a maioria das implantações do OmniCRM.

Para cobrança pay-as-you-go (onde o uso além dos saldos incluídos custa dinheiro), você precisa definir Planos de Tarifação que especifiquem valores em dólares a serem cobrados.

Como Funciona a Tarifação

Quando um cliente usa um serviço (dados, voz, SMS):

1. O CGRateS verifica saldos unitários (por exemplo, GB de dados incluídos)
2. Se o saldo unitário existir, o uso é deduzido dele
3. Se não houver saldo unitário ou o saldo estiver esgotado, o CGRateS recorre ao saldo monetário
4. **Plano de Tarifação** determina o preço (por exemplo, $0.10 por MB)
5. O custo calculado é deduzido do saldo monetário do cliente

Exemplo: Configurando Cobranças Pay-Per-Use

para Dados

```
# 1. Definir uma Tarifa de Destino (o preço)
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. Definir a tarifa real
OCS_Obj.SendData({
    "method": "ApierV2.SetTPRate",
    "params": [{
        "TPid": tpid,
        "ID": "RT_DATA_$0_10_PER_MB",
        "RateSlots": [{
            "ConnectFee": 0,          # Sem taxa de conexão
            "Rate": 0.10,             # $0.10 por unidade
            "RateUnit": "1048576",    # 1 MB (1024 * 1024 bytes)
            "RateIncrement": "1048576", # Cobrança por MB
            "GroupIntervalStart": "0s"
        }]
    }]
})

# 3. Definir um Plano de Tarifação que usa esta tarifa
OCS_Obj.SendData({
    "method": "ApierV2.SetTPRatingPlan",
    "params": [{
        "TPid": tpid,
        "ID": "RP_PAYG_DATA",
        "RatingPlanBindings": [{
            "DestinationRatesId": "DR_DATA_PAYG",
```

```
            "TimingId": "*any",
            "Weight": 10
        }]
    }]
})

# 4. Vincular o Plano de Tarifação a um Perfil de Tarifação (para
contas específicas)
OCS_Obj.SendData({
    "method": "ApierV2.SetTPRatingProfile",
    "params": [{
        "TPid": tpid,
        "Tenant": tenant,
        "Category": "data",
        "Subject": "*any",  # Ou identificador de conta específico
        "RatingPlanActivations": [{
            "ActivationTime": "2024-01-01T00:00:00Z",
            "RatingPlanId": "RP_PAYG_DATA",
            "FallbackSubjects": ""
        }]
    }]
})

# 5. Carregar o plano tarifário
OCS_Obj.SendData({
    "method": "APIerSv1.LoadTariffPlanFromStorDb",
    "params": [{
        "TPid": tpid,
        "DryRun": False,
        "Validate": True,
        "APIOpts": {}
    }]
})
```

O que isso faz:

- Quando um cliente não tem saldo de dados (ou esgota), o uso de dados custa **$0.10 por MB**
- A cobrança é deduzida de seu saldo monetário (`*monetary`)
- Os incrementos de cobrança são de 1 MB (arredonda para cima)

Exemplo: Tarifas de Chamadas de Voz

```
# Definir tarifas diferentes para diferentes destinos
OCS_Obj.SendData({
    "method": "ApierV2.SetTPRate",
    "params": [{
        "TPid": tpid,
        "ID": "RT_VOICE_DOMESTIC_$0_30_PER_MIN",
        "RateSlots": [{
            "ConnectFee": 0.15,           # $0.15 taxa de conexão
            "Rate": 0.30,                 # $0.30 por minuto
            "RateUnit": "60s",            # Cobrança por minuto
            "RateIncrement": "60s",       # Arredonda para minutos
inteiros
            "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 taxa de conexão
            "Rate": 1.50,                 # $1.50 por minuto
            "RateUnit": "60s",
            "RateIncrement": "1s",        # Cobrança por segundo
(mais precisa)
            "GroupIntervalStart": "0s"
        }]
    }]
})
```

Perfis de Carregadores (Opcional)

Carregadores aplicam tarifação a tipos específicos de eventos. Para a maioria dos casos de uso, o carregador padrão é suficiente:

```
OCS_Obj.SendData({
    "method": "APIerSv1.SetChargerProfile",
    "params": [{
        "Tenant": tenant,
        "ID": "Charger_Default",
        "FilterIDs": [],
        "AttributeIDs": ["*constant:*req.RequestType:*none"],
        "Weight": 999
    }]
})
```

Veja também:

- Ações CGRateS e Comportamentos de Recarga - Abordagens de gerenciamento de saldo (unitário, monetário, híbrido) e configuração de tarifação
- Destinos CGRateS - Como definir destinos geográficos e PLMN

Melhores Práticas

1. Use Convenções de Nomenclatura Consistentes

```
# Bom - relação clara
ActionsId = "Action_premium-plan-100gb"
product_slug = "premium-plan-100gb"

# Ruim - sem relação clara
ActionsId = "Action_Plan_A"
product_slug = "premium_100"
```

2. Defina Ações Antes de Produtos

Sempre crie Ações do CGRateS antes de criar produtos do CRM. O playbook de provisionamento falhará se tentar executar uma Ação inexistente.

3. Inclua Registro CDR

Sempre adicione a ação `*cdrlog` para rastrear adições de saldo:

```
{
    "Identifier": "*cdrlog",
    "BalanceType": "*generic",
    "ExtraParameters": "
{\"Category\":\"^activation\",\"Destination\":\"Nome do
Produto\"}",
    "Weight": 80
}
```

4. Use IDs de Saldo Descritivos

```
# Bom
"BalanceId": "Premium_Data_100GB"
"BalanceId": "AU_Voice_Domestic__" + str(units)

# Ruim
"BalanceId": "data1"
"BalanceId": "balance"
```

5. Documente Unidades Claramente

```
# Bom - mostra cálculo
Units = 100 * 1024 * 1024 * 1024  # 100GB em bytes
Units = 3000 * 60 * 1000000000    # 3000 minutos em nanosegundos

# Ruim - número mágico
Units = 107374182400
```

6. Use Peso de Forma Estratégica

Peso mais alto = maior prioridade tanto para execução quanto para consumo:

```
# Ordem de execução (peso mais alto = executa primeiro)
{"Identifier": "*reset_account", "Weight": 700}
{"Identifier": "*topup_reset", "Weight": 90}
{"Identifier": "*cdrlog", "Weight": 80}

# Consumo de saldo (peso mais alto = consumido primeiro)
"BalanceWeight": 1200  # Saldos premium/domésticos
"BalanceWeight": 1100  # Saldos de roaming
"BalanceWeight": 1000  # Saldos internacionais
```

7. Teste Primeiro com Valores Pequenos

Ao definir novos produtos, teste primeiro com saldos pequenos:

```
# Testando
Units = 100 * 1024 * 1024  # 100MB para teste

# Produção
Units = 100 * 1024 * 1024 * 1024  # 100GB
```

Solução de Problemas

Erro: "Ação não encontrada"

Sintoma: O playbook falha com "Ação não encontrada" ou "ActionsId não existe"

Causa: A Ação não está definida no CGRateS ou há uma discrepância de nomenclatura

Solução:

1. Verifique se a Ação existe: Consulte o CGRateS pela Ação
2. Verifique a nomenclatura: Certifique-se de que `ActionsId = "Action_" + product_slug`
3. Defina a Ação se estiver faltando

Erro: "Destino não encontrado"

Sintoma: A Ação é executada, mas os saldos não são criados

Causa: DestinationIDs referencia um destino inexistente

Solução:

1. Defina o destino no CGRateS primeiro
2. Use `*any` para destinos universais
3. Verifique a ortografia dos IDs de destino

Produto Criado, Mas Nenhum Saldo Adicionado

Sintoma: O produto é provisionado com sucesso, mas o cliente não tem saldo

Causa: A Ação existe, mas não possui operações de adição de saldo

Solução:

1. Verifique se a Ação tem operações `*topup` ou `*topup_reset`
2. Verifique se o valor de Units está correto (não 0)
3. Verifique se o ExpiryTime não já passou

Saldos Não Estão Acumulando

Sintoma: Usando rollover, mas saldo não utilizado desaparece

Causa: Usando `*topup_reset` em vez de `*topup`

Solução:

- Altere `Identifier` de `*topup_reset` para `*topup`
- Certifique-se de que BalanceId seja consistente entre as recargas

Veja também: Ações CGRateS e Comportamentos de Recarga

Resumo

Este guia demonstra uma abordagem comum para configurar produtos no OmniCRM com o CGRateS. Os padrões mostrados aqui funcionam bem para casos de uso típicos de operadores móveis virtuais (MVNO) com planos pré-pagos/pós-pagos, pacotes de dados e roaming.

Lembre-se: O CGRateS é capaz de cenários muito mais sofisticados do que os cobertos aqui. Se você precisar de recursos avançados como:

- Tarifação diferenciada MO/MT para voz e SMS de roaming
- Cobrança baseada em CAMEL para dispositivos legados 2G/3G
- Variações de tarifas por hora do dia ou sazonais
- Preços baseados em níveis de clientes (bronze/prata/ouro)
- Cobrança baseada na qualidade da rede ou QoS
- Roteamento de menor custo com múltiplos fornecedores

...consulte a documentação do CGRateS diretamente e trabalhe com a configuração do seu plano de tarifação independentemente das definições de produtos do CRM.

Documentação Relacionada

- Ações CGRateS e Comportamentos de Recarga - Guia detalhado sobre tipos de Ação e comportamento de saldo
- Guia de Playbooks do Ansible - Como os playbooks executam Ações
- Cobrança e Pagamentos a partir de Playbooks - Processamento de pagamentos durante o provisionamento
- Ciclo de Vida do Produto - Gerenciando produtos através de seu ciclo de vida
- Conceitos: Produtos e Serviços - Fundamentos do produto
- Documentação Oficial do CGRateS - Referência completa do CGRateS

Sistema de Transmissão de Células

O Sistema de Transmissão de Células no OmniCRM permite que operadores de redes móveis enviem alertas de emergência e avisos públicos para dispositivos móveis dentro de áreas geográficas específicas. A Transmissão de Células é um recurso crítico de segurança pública usado para alertas AMBER, avisos meteorológicos, alertas de tsunamis e outras notificações de emergência.

Vantagem Chave: Ao contrário das mensagens SMS padrão, as mensagens de Transmissão de Células alertarão audivelmente em telefones que estão no modo silencioso, sem crédito ou em roaming. Como esta é uma mensagem de transmissão, é possível enviar um alerta para cada membro da população que possui um telefone móvel em questão de segundos.

Link de Aviso Omnitouch (OWL)

A plataforma Link de Aviso Omnitouch (OWL) fornece uma solução abrangente para profissionais de gerenciamento de desastres e operadores de redes móveis:

- **Entidade de Transmissão de Células (CBE)** - Aplicativo seguro baseado na web para usuários autorizados criarem e transmitirem mensagens de aviso de emergência
- **Centro de Transmissão de Células (CBC)** - Componente de integração de rede compatível com padrões que se conecta a redes celulares (2G/3G/4G/5G) para distribuir mensagens

O OWL é projetado para uso em qualquer dispositivo com um navegador da web (Chrome/Firefox/Safari/Edge), como computadores, laptops, tablets ou telefones móveis.

Visão Geral

A Transmissão de Células (também conhecida como Sistema de Aviso Público ou PWS) permite que os operadores:

- **Enviar Alertas de Emergência** - Distribuir informações críticas de segurança para todos os dispositivos em uma área
- **Alvo de Regiões Geográficas** - Transmitir para áreas de rastreamento específicas ou células de rede controlando quais torres de celular transmitem mensagens
- **Suportar Múltiplas Línguas** - Fornecer mensagens de alerta em várias línguas simultaneamente (até 500 caracteres por língua)
- **Gerenciar o Ciclo de Vida do Alerta** - Criar, atualizar, aprovar, monitorar e excluir mensagens de transmissão
- **Integrar com Sistemas Externos** - Conectar-se à infraestrutura do CBC (Centro de Transmissão de Células) através de várias interfaces de rede celular
- **Autenticação de Dois Fatores** - Processo de aprovação seguro usando Senhas de Uso Único Baseadas em Tempo (TOTP)
- **Regra de Duas Pessoas** - Requisito opcional para aprovação de uma segunda pessoa antes da transmissão da mensagem

Ao contrário do SMS, a Transmissão de Células não requer listas de assinantes e pode alcançar todos os dispositivos capazes em uma área geográfica instantaneamente, tornando-a ideal para alertas de segurança pública críticos em termos de tempo. Na maioria das redes, uma transmissão para todos os dispositivos leva menos de 10 segundos.

Casos de Uso

A Transmissão de Células é usada para:

- **Avisos de Emergência** - Desastres naturais (terremotos, tsunamis, inundações, incêndios)
- **Alertas AMBER** - Notificações de sequestro de crianças
- **Alertas Meteorológicos** - Avisos de clima severo, alertas de tornados

- **Segurança Pública** - Ameaças terroristas, vazamentos químicos, evacuações
- **Mensagens de Teste** - Testes de sistema e campanhas de conscientização pública (Mensagens de Teste Mensais não alertam os usuários, mas serão recebidas)

O direcionamento geográfico é frequentemente de extrema importância em situações de emergência. Por exemplo, alertas de tsunami aconselhando aqueles próximos a áreas costeiras a buscar terrenos mais altos não devem ser enviados a pessoas em áreas interiores distantes do perigo. Ao controlar quais torres de celular transmitem mensagens de aviso de emergência, o escopo da transmissão pode ser limitado à área geográfica apropriada.

Ciclo de Vida da Mensagem de Aviso de Emergência

Transmitir uma mensagem de Aviso de Emergência requer ação rápida enquanto fornece informações precisas e autenticação para garantir a validade da mensagem.

O ciclo de vida da mensagem consiste em quatro etapas:

1. **Definição da Mensagem** - Tipo de mensagem, conteúdo da mensagem, configurações de expiração
2. **Direcionamento da Mensagem** - Áreas geográficas e torres de celular para transmitir
3. **Aprovação da Mensagem** - Autorização/verificação da identidade do operador e aprovação da segunda pessoa (se necessário)
4. **Revisão da Mensagem** - Confirmação final antes da transmissão

Após a transmissão, as mensagens podem ser monitoradas, atualizadas conforme as situações evoluem e interrompidas quando o perigo imediato diminui.

Etapa 1: Definição da Mensagem

Esta etapa define os parâmetros básicos da mensagem de aviso de emergência a ser transmitida:

Identificador da Mensagem
Diferentes tipos de mensagens têm identificadores diferentes, que são tratados de maneira diferente pelos telefones receptores. Por exemplo, uma Mensagem de Teste Mensal não deve alertar usuários reais, mas eles ainda receberão a mensagem.

Nota: Cada modelo de mensagem disponível no sistema OWL já possui o Identificador de Mensagem (MI) apropriado embutido, portanto, os usuários do sistema não precisam selecionar isso manualmente ao criar uma mensagem de alerta.

Texto da Mensagem
Corpo do texto limitado a 500 caracteres contendo a mensagem a ser exibida para os usuários finais. As mensagens podem ser fornecidas em **múltiplas línguas** adicionando o texto do segundo idioma abaixo do primeiro. Lembre-se de que o limite de 500 caracteres se aplica à mensagem total, incluindo todos os idiomas.

Modelos de Mensagem
Mensagens de alerta de perigo podem ser pré-definidas com antecedência como "modelos" para diferentes cenários previstos, como:

- Inundações
- Tsunamis
- Terremotos
- Testes periódicos
- Outros cenários de desastre

Os modelos economizam tempo durante emergências. Esses modelos podem ser modificados conforme necessário ao definir a mensagem, ou as mensagens podem ser escritas do zero.

Expiração e Repetições da Mensagem
Mensagens de emergência têm uma vida útil finita para as quais são

relevantes. Ao definir a mensagem:

- **Expira (minutos)** - Quanto tempo a mensagem continuará a ser transmitida
- **Repetições da Mensagem** - Quantas vezes será retransmitida

Cada telefone mostrará a mensagem ao usuário apenas uma vez. No entanto, os sites de celular continuarão a transmitir mensagens até que o tempo de expiração seja alcançado para garantir que as pessoas que entram na área de cobertura de fora recebam a mensagem.

Etapa 2: Direcionamento

As mensagens de Transmissão de Células são enviadas no nível da torre de celular, e o alcance geográfico pode ser limitado selecionando quais torres transmitem a mensagem.

Direcionamento Opcional
Esta etapa é opcional. Não inserir nenhuma informação de direcionamento significa que todas as torres de celular transmitirã a mensagem de aviso de emergência.

Áreas Alvo Pré-definidas
O sistema OWL possui um banco de dados de todas as torres de celular e pode definir áreas alvo em um mapa. As áreas podem ser direcionadas usando zonas pré-definidas (determinadas com antecedência para seleção rápida) ou desenhando áreas personalizadas no mapa.

Ferramentas de Desenho no Mapa
Áreas alvo personalizadas podem ser criadas usando:

- Ferramenta de Polígono - Desenhar limites de cobertura precisos
- Ferramenta de Círculo - Alertas rápidos baseados em raio
- Ferramenta de Retângulo - Cobertura alinhada à grade

O recurso "Adicionar Nova Zona" permite definir áreas alvo personalizadas que podem ser salvas para uso futuro.

Etapa 3: Aprovação

Um processo de aprovação valida que a pessoa que emite a mensagem de Aviso de Emergência está autorizada a fazê-lo.

Autenticação de Dois Fatores
Usa Senhas de Uso Único Baseadas em Tempo (TOTP) através de:

- **Token físico** (como um RSA SecurID)
- **Solução baseada em aplicativo** (Google Authenticator, Authy, Microsoft Authenticator ou outros aplicativos compatíveis com TOTP)

Usuários que criarão ou aprovarão rascunhos de mensagens de alerta devem ter um aplicativo autenticador em seu smartphone para gerar o código de autorização que o sistema solicitará.

Configuração do 2FA
Ao configurar o 2FA pela primeira vez:

1. Instale um aplicativo autenticador em seu smartphone (Google Authenticator, Authy, Microsoft Authenticator, etc.)
2. Navegue até as configurações da sua conta OWL e escaneie o código QR com seu aplicativo autenticador
3. Insira o código de verificação para confirmar a configuração
4. Salve os códigos de backup em um local seguro
5. Teste a geração de códigos antes das emergências

Para instruções detalhadas de configuração do 2FA, veja `Autenticação de Dois Fatores <2fa>`.

Se você mudar de dispositivos móveis ou o aplicativo parar de sincronizar com sua conta OWL, entre em contato com seu Administrador de Sistema para obter ajuda. Administradores podem redefinir tokens de 2FA na página **Usuários e Funções → Usuários**.

Regra de Duas Pessoas
Quando o processo requer aprovação de uma segunda pessoa, a pessoa que emite o alerta deve inserir a Senha de Uso Único Baseada em Tempo da outra

pessoa antes que o processo possa prosseguir. Isso fornece supervisão e minimiza o risco de uso indevido.

Funções de Usuário Granulares
Funções de usuário individuais podem ser configuradas para:

- Permitir que apenas certos usuários enviem mensagens pré-definidas
- Restringir o direcionamento de mensagens a regiões específicas
- Exigir fluxos de aprovação adicionais

Etapa 4: Revisão

Uma vez que as etapas de Definição da Mensagem, Direcionamento e Aprovação estejam completas, o operador deve revisar a mensagem antes da transmissão final. Uma vez satisfeito com os detalhes da mensagem, ele pode transmitir a mensagem.

Velocidade de Transmissão: Na maioria das redes, uma transmissão para todos os dispositivos na rede leva menos de 10 segundos.

Etapa 5: Monitoramento e Atualizações

Uma vez que a transmissão da mensagem é iniciada, os operadores podem monitorar e gerenciar as mensagens transmitidas.

Feedback da Rede
As redes celulares retornam informações sobre os sites de celular que transmitiram a mensagem. Se um site de celular estiver offline ou indisponível, isso será relatado de volta ao operador.

Retransmissão Automática
Se algum site de celular offline se tornar disponível novamente enquanto o Aviso de Emergência ainda estiver ativo, todos os telefones conectados a essa célula receberão a mensagem.

Atualizações em Voo
Uma vez transmitida, a mensagem pode ser:

- Atualizada conforme a situação evolui

- Modificada com novo conteúdo do corpo da mensagem
- Retirada/parada a qualquer momento

Registros Históricos
Todas as informações sobre mensagens históricas podem ser visualizadas e revisadas para fins de auditoria.

Estrutura da Mensagem

Cada mensagem de Transmissão de Células consiste em:

Configuração da Mensagem

- **Identificador da Mensagem** - Identificador único para o tipo de alerta (por exemplo, 4370 para ETWS Terremoto, 4371 para ETWS Tsunami)
- **Categoria** - Categoria do alerta (normal, emergência, alta, extrema)
- **Período de Repetição** - Segundos entre repetições de transmissão
- **Número de Transmissões** - Quantas vezes transmitir a mensagem
- **Período de Aviso** - Duração em segundos que o aviso é válido
- **Indicador de Canal** - Tipo de canal usado para transmissão

Mensagens Localizadas

Cada mensagem do CBC pode incluir várias variantes de idioma:

- **Idioma** - Código de idioma ISO (en, es, fr, zh, etc.)
- **Corpo da Mensagem** - Texto do alerta nesse idioma (até 1395 caracteres)

O sistema transmite automaticamente todas as variantes de idioma, permitindo que os destinatários visualizem alertas em seu idioma preferido.

Áreas de Rastreamento

Define o direcionamento geográfico para o alerta:

- **Área de Rastreamento** - Identificador geográfico (ID da célula, código da área de rastreamento)
- **Operador** - Código do operador de rede móvel (MCC-MNC)

- **Tipo de RAT** - Tecnologia de Acesso Rádio (LTE, 5G, UMTS, GSM)

Várias áreas de rastreamento podem ser especificadas para cobrir regiões maiores ou múltiplos operadores.

Criando uma Mensagem de Transmissão de Células

Via Interface Web:

1. **Navegue até Transmissão de Células** - Acesse a interface de gerenciamento do CBC a partir da navegação principal
2. **Clique em "Criar Alerta"** - Abre o formulário de criação de mensagem
3. **Configure os Parâmetros da Mensagem:**
    - Identificador da Mensagem (por exemplo, 4370 para alertas de terremoto)
    - Categoria (normal, alta, extrema)
    - Período de Repetição (tipicamente 5-60 segundos)
    - Número de Transmissões (999 para contínuo, ou contagem específica)
    - Período de Aviso (duração em segundos)
    - Indicador de Canal (tipicamente "básico")

4. **Adicione Mensagens Localizadas:**
    - Clique em "Adicionar Idioma"
    - Selecione o idioma no menu suspenso
    - Insira o texto da mensagem (máx 1395 caracteres para GSM7, menos para Unicode)
    - Repita para idiomas adicionais
5. **Defina Áreas de Rastreamento:**
    - Clique em "Adicionar Área de Rastreamento"
    - Insira o código da área de rastreamento
    - Selecione o operador (combinação MCC-MNC)
    - Escolha o tipo de RAT (LTE, 5G, etc.)
    - Repita para áreas geográficas adicionais

6. **Revise e Crie** - Verifique todos os detalhes e clique em "Criar Alerta"

Via API:

Endpoint: `PUT /crm/cbc/`

Permissão Necessária: `CREATE_CBC_MESSAGE`

Corpo da Solicitação:

```
{
  "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"
    }
  ]
}
```


Resposta:

```
{
  "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": [...]
}
```


A mensagem é enviada imediatamente ao Centro de Transmissão de Células para transmissão.

Gerenciando Mensagens Existentes

Ver Todas as Mensagens

Endpoint: `GET /crm/cbc/`

Permissão Necessária: `VIEW_CBC_MESSAGE`

Retorna a lista de todas as mensagens do CBC com seu status, carimbos de data/hora e configuração.

Atualizar uma Mensagem

Endpoint: `PATCH /crm/cbc/{cbc_message_id}`

Permissão Necessária: `UPDATE_CBC_MESSAGE`

Atualiza o conteúdo da mensagem, áreas de rastreamento ou parâmetros de transmissão. Mensagens atualizadas são reenviadas ao CBC.

Corpo da Solicitação:

```
{
  "cbc_message_id": 123,
  "numberOfBroadcasts": 500,
  "localized_messages": [
    {
      "language": "en",
      "messageBody": "UPDATED: Earthquake warning still in effect.
Aftershocks possible."
    }
  ]
}
```

Excluir uma Mensagem

Endpoint: `DELETE /crm/cbc/{cbc_message_id}`

Permissão Necessária: `DELETE_CBC_MESSAGE`

Remove a mensagem do banco de dados e tenta cancelá-la no CBC.

Fluxo de Trabalho de Aprovação

Mensagens de Transmissão de Células suportam um fluxo de aprovação opcional para alertas de alto risco:

1. **Usuário Iniciador** - Membro da equipe que cria o alerta (`initiating_user` field)
2. **Usuário Aprovador** - Gerente que aprova o alerta antes da transmissão (`approving_user` field)

Se a aprovação for necessária:

- A mensagem é criada com `approving_user = null`
- A mensagem é mantida em estado de "aprovação pendente"
- O usuário aprovador revisa a mensagem e aprova ou rejeita
- Após a aprovação, `approving_user` é definido e a mensagem é transmitida

Esse fluxo de trabalho é configurável com base na política da organização.

Identificadores de Mensagem

Identificadores de mensagem padrão seguem 3GPP TS 23.041:

Sistema de Aviso de Terremoto e Tsunami (ETWS):

- **4370** - Aviso de Terremoto ETWS
- **4371** - Aviso de Tsunami ETWS
- **4372** - Aviso Combinado de Terremoto e Tsunami ETWS
- **4373-4378** - Outros Tipos de Emergência ETWS
- **4379** - Mensagem de Teste ETWS

Sistema de Alertas Móveis Comerciais (CMAS) / Alertas de Emergência Sem Fio (WEA):

- **4352** - Alerta Presidencial
- **4353-4355** - Alertas Extremos
- **4356-4359** - Alertas Severos
- **4360-4363** - Alertas AMBER
- **4364-4367** - Mensagens de Segurança Pública
- **4368-4369** - Testes Estaduais/Locais
- **4380-4381** - Mensagens de Teste

Faixas Personalizadas:

- **0-999** - Reservado para alertas específicos do operador
- **1000-4095** - Tipos de mensagens personalizadas

Integração com o Centro de Transmissão de Células

A Entidade de Transmissão de Células (CBE) precisa de um mecanismo para entregar mensagens a redes celulares individuais. O Centro de Transmissão de

Células (CBC) do OWL se conecta a cada rede celular para enviar mensagens de Aviso de Emergência ao público.

Redundância Multi-Rede
No caso de um operador de rede celular estar indisponível (queda ou sem cobertura), se outra rede celular operante estiver disponível, os usuários ainda receberão mensagens de Aviso de Emergência através das outras redes disponíveis.

Instâncias de CBC por Operador
Para garantir que não haja conexão entre redes concorrentes, o OWL executa uma instância separada do CBC para cada MNO; não é compartilhada entre operadores.

Configuração do OWL CBC

A URL do CBC é configurada em `crm_config.yaml`:

```
cbc_url: "http://cbc.example.com:8080"
```

Transmissão de Mensagens:

Quando uma mensagem do CBC é criada ou atualizada:

1. O OmniCRM CBE armazena a mensagem em seu banco de dados
2. A mensagem é formatada para a API do CBC
3. Solicitação HTTP POST enviada para `{cbc_url}/alerts/send`
4. O CBC se conecta à rede celular através da interface apropriada (CBSP, SBc-AP ou N50)
5. O CBC reconhece o recebimento e começa a transmissão
6. Dispositivos nas áreas de rastreamento alvo recebem o alerta

Exclusão de Mensagens:

Quando uma mensagem é excluída, o CBE tenta cancelá-la no CBC para interromper transmissões em andamento.

Pontos de Integração da Rede Celular

Diferentes gerações de redes celulares (2G/3G/4G/5G) têm interfaces exclusivas para conectar-se ao Centro de Transmissão de Células. Com base nas tecnologias usadas por cada rede celular, a interface correta deve ser configurada.

O Centro de Transmissão de Células do OWL suporta interfaces de Transmissão de Células 2G, 3G, 4G e 5G e se integrou a vários componentes de rede celular comumente usados.

CBSP - 2G/3G - Controlador de Estação Base (BSC)

A interface do Protocolo de Serviço de Transmissão de Células (CBSP) conecta o CBC ao Controlador de Estação Base (BSC) que controla estações base 2G (GSM).

- Usado para mensagens de Transmissão de Células 2G e 3G com implantações combinadas de Controlador de Rede de Rádio/Controlador de Estação Base
- Pode ser configurado como um cliente ou arranjo de servidor, dependendo do fornecedor do BSC
- Uma conexão deve ser estabelecida entre todos os BSCs na rede e o OWL CBC
- Interfaces são continuamente monitoradas com alertas para indicar se um link CBSP caiu

Nota: 3GPP definiu o Protocolo de Transmissão de Área de Serviço (SABP) para uso em implantações RNC autônomas. Isso pode ser usado se necessário quando o CBSP não for suportado para células 3G em um RNC/BSC combinado, no entanto, testes adicionais e suporte do fornecedor do RNC podem ser necessários.

SBc-AP - 4G/5G Não Autônomo - MME/IWF

A interface SBc-AP conecta o OWL CBC ao MME (Entidade de Gerenciamento de Mobilidade) que atende eNodeBs/gNodeBs 4G e 5G.

- Usado para redes LTE 4G

- Também usado para 5G Não Autônomo (a maioria das implantações em 2025)
- Uma conexão deve ser estabelecida entre todos os MMEs na rede e o OWL CBC
- Interfaces são continuamente monitoradas com alertas para indicar se um link SBc-AP caiu

N50 - 5G Autônomo - AMF

Para redes 5G autônomas, a interface N50 conecta o OWL CBC ao AMF (Função de Gerenciamento de Acesso e Mobilidade) que atende gNodeBs 5G.

- A interface está presente no OWL CBC
- Não foi extensivamente testada com AMFs de terceiros devido ao pequeno número de redes 5G SA comercialmente disponíveis em 2025
- Será totalmente suportada à medida que as implantações 5G SA se tornem mais comuns

Requisitos de Rede do MNO

A rede deve estar em vigor entre o OWL CBC e a rede do Operador de Rede Móvel para alcançar as interfaces descritas acima.

Isso é tratado caso a caso, mas geralmente requer:

- Conexão cruzada/fibra dedicada entre o CBC e a rede MNO
- Cada interface logicamente separada
- Conectividade para cada ponto de integração (MME, RNC, BSC) na rede celular

Equipamentos de Rede Suportados

O OWL CBC foi testado e se integra a componentes de rede celular comumente usados de grandes fornecedores:

Integração de Dados do Site Celular

O OWL suporta coleta automática de dados de:

- **Nokia NetAct**
- **Huawei U2000 / U2020**
- **ZTE NetNumen / ZXPOS**
- **Ericsson ENM**

Alternativamente, os dados do site celular podem ser fornecidos à equipe de operações do Omnitouch periodicamente via e-mail.

Gerenciamento de Usuários e Controle de Acesso

Controle de Acesso Baseado em Funções (RBAC)

O sistema OWL usa controle de acesso baseado em funções (RBAC): pessoas (Usuários Registrados) são atribuídas a uma ou mais Funções, e cada Função é um conjunto de Permissões. Permissões são a menor unidade de acesso (por exemplo, criar alerta de mensagem em rascunho). O acesso efetivo de um Usuário Registrado é a união das Permissões de todas as Funções atribuídas.

Componentes do RBAC:

- **Usuários** - Pessoas reais que fazem login no sistema OWL
- **Permissões** - Micro capacidades (por exemplo, aprovar rascunho de mensagem, criar mensagem, visualizar relatórios)
- **Funções** - Conjuntos nomeados de permissões (por exemplo, Aprovadores de Mensagem, Criadores de Mensagem)
- **Atribuição** - Usuários recebem uma ou mais Funções; permissões se agregam

Benefícios do RBAC:

1. **Proteção de Dados** - Usuários veem e fazem apenas o que estão autorizados
2. **Adequação Operacional** - Funções espelham funções de trabalho (Admin, Criador de Mensagem, Aprovador de Mensagem)

3. **Administração Simples** - Conceder acesso atribuindo funções; evitar microgerenciamento por usuário

Permissões do Sistema

As permissões do sistema geralmente seguem padrões CRUD com quatro opções:

- **Visualizar** - Ler ou navegar por mensagens e relatórios
- **Criar** - Criar ou adicionar um alerta de mensagem
- **Atualizar** - Editar ou modificar um alerta de mensagem em rascunho
- **Excluir** - Excluir ou remover um alerta de mensagem em rascunho

Permissões Básicas do CBC:

- `CREATE_CBC_MESSAGE` - Criar novas mensagens de transmissão
- `VIEW_CBC_MESSAGE` - Visualizar mensagens existentes e seu status
- `UPDATE_CBC_MESSAGE` - Modificar conteúdo da mensagem ou parâmetros de transmissão
- `DELETE_CBC_MESSAGE` - Excluir mensagens e cancelar transmissões

Atribua essas permissões a funções com base nas responsabilidades de segurança pública de sua organização.

Para informações abrangentes sobre gerenciamento de funções e permissões de usuários, veja `Usuários e Funções <rbac>`.

Gerenciamento de Conta de Usuário

Login pela Primeira Vez

Quando um novo usuário é configurado no sistema, ele recebe credenciais de login pelo Administrador do Sistema:

1. Navegue até a URL de login do OWL fornecida pelo seu Administrador de Sistema
2. Insira seu nome de usuário e senha

3. Se o 2FA estiver habilitado para sua conta, você será solicitado a configurá-lo (veja a seção 2FA acima)

Melhor Prática: Os usuários devem alterar sua senha inicial imediatamente após o primeiro login por segurança. Navegue até as configurações do seu perfil de usuário para alterar sua senha.

Para mais detalhes sobre fluxos de autenticação, veja `Fluxos de Autenticação e Controles de Admin <authentication_flows>`.

Redefinição de Senha

Os usuários podem redefinir suas próprias senhas:

1. Na página de login do sistema, selecione "Esqueci a Senha"
2. Siga o processo de verificação por e-mail
3. Defina uma nova senha

Redefinição de Senha do Admin:

Os Administradores do Sistema podem redefinir senhas para usuários na página **Usuários e Funções → Usuários**:

1. Navegue até **Usuários e Funções → Usuários**
2. Selecione o usuário que precisa de redefinição de senha
3. Clique em **Redefinir Senha**
4. O sistema gera uma senha temporária segura aleatória (12 caracteres)
5. A senha temporária é exibida para o admin
6. Forneça a senha temporária ao usuário através de um canal seguro (telefone ou pessoalmente, nunca por e-mail)
7. O usuário pode fazer login com a senha temporária
8. **Importante:** O usuário deve alterar essa senha imediatamente após fazer login

Para mais detalhes, veja `Usuários e Funções <rbac>`.

Adicionando Novos Usuários ao Sistema

Os Administradores do Sistema podem adicionar novos usuários navegando até **Usuários e Funções → Usuários**:

Informações necessárias para cada novo Usuário Registrado:

- Nome
- Sobrenome
- E-mail
- Senha (temporária)
- Número de Celular
- Função(s)

Melhor Prática: Implemente um Formulário de Solicitação de Novo Usuário do Sistema que esteja alinhado com as políticas e procedimentos existentes de sua organização. Adicione usuários apenas após a aprovação adequada.

Editando Perfis de Usuário

Os Administradores do Sistema podem alterar permissões e acesso para usuários existentes:

1. Navegue até **Usuários e Funções → Usuários**
2. Selecione o usuário a ser editado
3. Modifique as funções atribuídas conforme necessário
4. Salve as alterações

Excluindo Usuários do Sistema

Os Administradores do Sistema podem excluir usuários de **Usuários e Func��es → Usuários**.

Aviso: Uma vez que um usuário foi excluído do sistema OWL, a ação não pode ser desfeita. Tenha muito cuidado antes de executar esta tarefa.

Visualizando Todos os Usuários do Sistema

A qualquer momento, o Administrador do Sistema pode visualizar uma lista de todos os Usuários Registrados e ver quais permissões e funções cada um possui navegando até **Usuários e Funções → Usuários**.

Histórico de Mensagens de Alerta

O sistema OWL mantém um registro de todas as mensagens de alerta enviadas para fins de auditoria e para apoiar a revisão operacional.

Acessando o Histórico de Mensagens:

Navegue até **Gerenciar Transmissões de Células** para visualizar:

- Todas as mensagens enviadas anteriormente
- Status da mensagem (ativa, expirada, cancelada)
- Carimbo de data/hora da transmissão
- Usuários iniciadores e aprovadores
- Detalhes de direcionamento geográfico
- Conteúdo e parâmetros da mensagem

Isso ajuda com:

- Auditoria do sistema
- Revisão dos procedimentos operacionais usados pelo NDMO e outras agências de alerta
- Compreensão da eficácia da mensagem
- Requisitos de conformidade e relatórios

Visualização do Mapa de Transmissão de Células

O Mapa de Transmissão de Células fornece uma **interface visual** para planejar áreas de cobertura geográfica e identificar quais torres de celular irão transmitir alertas.

Acessando o Mapa

Navegue até:

Ou diretamente:

Permissão Necessária: `VIEW_CBC_MESSAGE` ou `CREATE_CBC_MESSAGE`

Recursos do Mapa

O mapa exibe:

Localizações das Torres de Celular:

- **Ícones de Torres** - Posições do site celular exibidas como ícones de torres de celular
- **Indicadores de Setor** - Setas direcionais mostrando a orientação do setor da antena
- **Nomes das Torres** - Identificadores e nomes do site
- **Dados de Cobertura** - Carregado do arquivo `cellSites.txt`

Ferramentas de Desenho:

- **Ferramenta de Polígono** - Desenhar áreas de cobertura personalizadas
- **Ferramenta de Círculo** - Criar zonas de transmissão circulares
- **Ferramenta de Retângulo** - Definir áreas de cobertura retangulares

Análise de Cobertura:

- **Lista de Sites** - Tabela mostrando todos os sites de celular dentro da área desenhada
- **Gerenciamento de Zonas** - Salvar e carregar zonas de cobertura pré-definidas
- **Contagem de Sites** - Contagem em tempo real de torres na área de cobertura

Usando o Mapa

Passo 1: Carregar Dados do Site Celular

O mapa carrega automaticamente as localizações das torres de celular do `/cellSites.txt` ao carregar a página. Este arquivo contém:

- **lat/lng** - Coordenadas da torre

- **site_name** - Identificador
- **sectors** - Direções da antena em graus

Passo 2: Desenhar Área de Cobertura

Selecione uma ferramenta de desenho:

1. Clique em **Polígono** para desenhar formas personalizadas
    - Clique para adicionar pontos
    - Clique duas vezes para concluir
    - Cria limites de cobertura precisos
2. Clique em **Círculo** para desenhar áreas circulares
    - Clique no ponto central
    - Arraste para definir o raio
    - Útil para alertas rápidos baseados em raio
3. Clique em **Retângulo** para desenhar áreas retangulares
    - Clique em um canto
    - Arraste para o canto oposto
    - Eficiente para cobertura alinhada à grade

Passo 3: Revisar Sites Cobertos

Após desenhar, o sistema:

- Calcula quais sites de celular caem dentro da área
- Exibe a lista na tabela abaixo do mapa
- Mostra nomes e coordenadas dos sites
- Fornece contagem de torres afetadas

Tabela de Cobertura Exemplo:

Passo 4: Exportar Dados de Cobertura

Use os dados de cobertura para:

- **Identificar códigos de área de rastreamento** para a transmissão
- **Estimar alcance do alerta** com base na contagem de torres
- **Planejar resposta a emergências** entendendo as áreas afetadas

- **Salvar definições de zona** para uso futuro

Integração de Fluxo de Trabalho

A visualização do mapa integra-se com a criação de transmissões:

Planejando um Alerta:

1. **Abra o Mapa do CBC** para visualizar a área afetada
2. **Desenhe a zona de cobertura** usando a ferramenta de polígono
3. **Revise os sites cobertos** e ajuste os limites
4. **Anote os códigos de área de rastreamento** para os sites de celular afetados
5. **Crie mensagem do CBC** com direcionamento apropriado

Exemplo: Aviso de Tsunami Costeiro

Gerenciamento de Zonas

Salvando Zonas:

Zonas de cobertura pré-definidas podem ser salvas para cenários comuns:

- **Região Costeira** - Para avisos de tsunami/onda de tempestade
- **Zonas de Incêndio Florestal** - Para alertas de evacuação
- **Centros Urbanos** - Para alertas AMBER em áreas populosas
- **Corredores de Rodovia** - Para alertas de tráfego/clima

Formato de Dados da Zona:

Zonas são armazenadas em `site_data.json`:

```
[
  {
    "area": "Região Costeira",
    "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}
        ]
      }
    ]
  }
]
```

Carregando Zonas Salvas:

1. Selecione a zona no menu suspenso
2. O polígono é exibido automaticamente no mapa
3. A lista de sites é atualizada para mostrar torres cobertas
4. Modifique o polígono se necessário

Configuração do Mapa

API do Google Maps:

O mapa requer uma chave da API do Google Maps configurada no ambiente:

```
REACT_APP_GOOGLE_API_KEY=your_google_maps_api_key_here
```

Dados do Site Celular:

Atualize `public/cellSites.txt` com as localizações das torres de celular da sua rede:

```
# Formato: lat,lng,name,sector1,sector2,sector3
-33.8688,151.2093,SYD_CENTRAL,0,120,240
-33.8650,151.2070,SYD_HARBOUR,45,165,285
```

Obtenha as coordenadas do site celular de:

- Ferramentas de planejamento de rede
- Registros de instalação de torres
- Pesquisas de campo com GPS
- Configuração do controlador de rede de rádio (RNC)

Estilização do Mapa:

Estilos de mapa personalizados podem ser configurados em `mapStyles.js` para:

- Destacar serviços de emergência
- Mostrar características topográficas
- Enfatizar centros populacionais
- Combinar com a marca da organização

Casos de Uso

Planejamento de Emergência:

- Pré-planejar zonas de cobertura para áreas de risco conhecidas
- Testar diferentes estratégias de direcionamento
- Estimar alcance populacional
- Coordenar com serviços de emergência

Verificação de Alertas:

- Confirmar que a mensagem alcançará a área pretendida
- Identificar lacunas de cobertura
- Evitar sobre-alertas em regiões adjacentes
- Validar códigos de área de rastreamento

Análise de Rede:

- Visualizar a distribuição das torres de celular
- Identificar sobreposições de cobertura
- Planejar implantações de torres para melhor cobertura de alertas
- Otimizar a eficiência da transmissão

Melhores Práticas

Com base na experiência com clientes ao redor do mundo, a Omnitouch recomenda as seguintes melhores práticas para todas as implantações do OWL.

Conteúdo da Mensagem:

- Mantenha as mensagens concisas e claras (menos de 360 caracteres para alertas de página única)
- Use todas as letras maiúsculas para alertas de emergência (melhora a legibilidade)
- Inclua itens de ação específicos ("Evacue imediatamente", "Busque abrigo")
- Evite jargões técnicos
- Teste mensagens com dispositivos reais antes de emergências

Suporte a Idiomas:

- Sempre forneça mensagens no idioma principal da região
- Inclua idiomas adicionais para áreas multiculturais
- Assegure-se de que as traduções sejam culturalmente apropriadas
- Teste caracteres especiais e suporte a Unicode

Direcionamento Geográfico:

- Use as menores áreas de rastreamento necessárias para evitar fadiga de alertas
- Considere a densidade populacional ao definir o período de repetição

- Teste o direcionamento geográfico antes de emergências
- Mantenha documentação precisa da área de rastreamento

Testes:

- Use identificadores de mensagem de teste (4379, 4380-4381) para simulações
- Programe testes regulares do sistema
- Verifique se a integração do CBC está funcionando
- Treine a equipe sobre procedimentos de emergência

Fadiga de Alertas:

- Use apenas para emergências genuínas
- Evite sobre-transmissões
- Defina períodos de repetição apropriados
- Use níveis de severidade de forma adequada

Segurança da Conta do Usuário

Segurança do Token 2FA:

- Proteja tokens 2FA físicos como cartões de acesso a prédios
- Relate tokens perdidos ou roubados imediatamente
- Teste o 2FA regularmente para garantir que esteja funcionando
- Salve códigos de backup ao configurar o 2FA e armazene-os de forma segura offline
- Entre em contato com o Administrador do Sistema se precisar redefinir o 2FA

Para mais informações sobre configuração e recuperação do 2FA, veja `Autenticação de Dois Fatores <2fa>`.

Manutenção de Dados

Áreas Alvo Pré-definidas:

À medida que os limites geográficos mudam, ocorrem desenvolvimentos e áreas de risco se deslocam, há necessidade de revisar as Áreas Alvo Pré-definidas. A Omnitouch sugere que esses dados sejam revisados **anualmente pelo NDMO**, com o apoio de outras agências de alerta, quando aplicável.

Atualizações de limites podem ser definidas usando várias plataformas GIS comuns ou Google Earth, e então fornecidas por e-mail à equipe de Operações da Omnitouch, que fará as alterações no sistema.

Modelos de Mensagem Pré-definidos:

Os modelos de mensagem pré-definidos devem ser revisados **pelo menos anualmente pelo NDMO**, com o apoio de outras agências de alerta, quando aplicável, para garantir:

- O conteúdo da mensagem ainda reflete com precisão o perigo e a chamada à ação
- As informações de contato e instruções estão atualizadas
- As traduções de idiomas permanecem precisas
- O tom e a urgência da mensagem são apropriados
- Referências a serviços de emergência ou procedimentos estão atualizadas

Atualizações podem ser fornecidas por e-mail à equipe de Operações da Omnitouch, que fará as alterações no sistema.

Dados do Site Celular:

- Revise e atualize os dados do site celular sempre que ocorrerem mudanças na rede
- Frequência típica de atualização: mensal ou trimestral
- Coordene com as equipes de planejamento de rede da MNO
- Verifique a precisão após grandes atualizações ou expansões da rede

Arquitetura do Sistema

Todos os produtos da Omnitouch são projetados para suportar implantações geograficamente distribuídas.

Opções de Implantação

Todos os componentes podem ser executados como:

- **Contêineres (K8s)** - Implantações de contêineres orquestradas pelo Kubernetes
- **Máquinas Virtuais** - VMware, Proxmox, HyperV
- **Nuvem Privada** - Infraestrutura de nuvem local
- **Nuvem Pública** - AWS, GCP
- **Bare Metal** - Implantação direta em hardware

Arquitetura Distribuída

A arquitetura distribuída permite:

- Escritórios Locais de Gerenciamento de Desastres acessarem o OWL CBE e distribuírem mensagens mesmo se uma região se tornar isolada da rede nacional
- Múltiplas Entidades de Transmissão de Células e múltiplos Centros de Transmissão de Células por operador/país
- Agências locais de resposta a desastres (Governo Municipal, Polícia, Bombeiros, etc.) emitirem alertas para sua região mesmo se o principal NDMO (Escritório Nacional de Gerenciamento de Desastres) perder acesso

Isso é particularmente valioso quando as MNOs distribuíram sua rede celular com recursos locais de BSC/MME.

Networking do CBE e CBC

Para garantir que não haja conexão entre redes concorrentes, cada MNO tem uma instância separada do CBC (não compartilhada).

Requisitos de rede entre CBE e CBC:

- Todo o tráfego é criptografado entre CBE e CBC
- Autenticação baseada em certificados mútuos
- Conectividade do CBE ao CBC via TLS na porta TCP 443

- Coordenação necessária entre MNOs e o NDMO ou agência que hospeda o CBE

Considerações de Acesso

A agência que hospeda o CBE precisará definir procedimentos de acesso para usuários finais (ou seja, Citrix, VPN, etc.), lembrando que o sistema deve ser acessível em cenários não ideais, como:

- Quedas em larga escala de redes de energia pública
- Falhas na rede de telecomunicações
- Desastres naturais que afetam a infraestrutura

Requisitos de Implantação

Requisitos de VM do CBE (NDMO / Governo)

3x Máquinas Virtuais:

- **2x VMs do CBE** (Idealmente em diferentes datacenters/zona de disponibilidade)
- **1x VM de Monitoramento**

Cada VM requer:

- **Armazenamento:** 50GB
- **CPU:** 2x CPU Virtual
- **RAM:** 8GB
- **OS:** OS Base fornecido pela Omnitouch
- **Networking:** Permitir tráfego para VMs do CBC na porta TCP 443 para tráfego TLS para controlar os CBCs

Requisitos de VM do CBC (MNO)

3x Máquinas Virtuais:

- **2x VMs do CBC** (Idealmente em diferentes datacenters/zona de disponibilidade)
- **1x VM de Monitoramento**

Cada VM requer:

- **Armazenamento:** 50GB
- **CPU:** 2x CPU Virtual
- **RAM:** 8GB
- **OS:** OS Base fornecido pela Omnitouch
- **Conectividade:** Para cada um dos Pontos de Integração na rede celular (MME, RNC, BSC)
- **Networking:** Permitir tráfego das VMs do CBE na porta TCP 443 para tráfego TLS a ser controlado pelos CBEs

Etapas de Integração

Implantar o OWL envolve as seguintes etapas:

1. **Fonte de novos operadores** - Identificar operadores de rede móvel participantes
2. **Administração do projeto (configuração)** - Estabelecer governança e administração do projeto
3. **Selecionar local de hospedagem para o CBE** - Determinar onde o CBE será hospedado
4. **Definir Usuários e Fluxos/Procedimentos de Mensagens** - Estabelecer fluxos de aprovação e funções de usuário
5. **Definir Polígonos para Áreas Alvo e Modelos de Mensagem** - Pré-configurar cenários comuns
6. **Alocações de Endereço IP** - Alocar endereços IP para NDMO e MNOs
7. **Configurar VPN site-a-site** - Estabelecer conexão segura com a equipe da Omnitouch
8. **Implantar VMs do CBE para NDMO** - Instalar Entidade de Transmissão de Células
9. **Implantar VMs do CBC para MNO** - Instalar Centro de Transmissão de Células em cada operador

10. **Configurar Elementos de Rede** - Configurar conectividade ao CBC a partir do equipamento de rede celular
11. **Networking entre VMs do CBE e CBC** - Estabelecer comunicação segura
12. **Networking entre VMs do CBC e Elementos de Rede** - Conectar ao BSC/MME/AMF
13. **Configurar acesso à API para NMS** - Configurar integração de dados do site celular
14. **Configuração de monitoramento e teste** - Verificar monitoramento e alertas
15. **Verificação/Teste de alertas de teste** - Realizar testes do sistema
16. **Teste de alerta público** - Realizar teste público de ponta a ponta

Limites de Caracteres

Mensagens de Transmissão de Células têm limites de caracteres rigorosos com base na codificação:

Codificação GSM de 7 bits (inglês, caracteres latinos básicos):

- Página única: 93 caracteres
- Múltiplas páginas: 15 páginas × 93 = 1395 caracteres no máximo

Codificação Unicode UCS-2 (escritas não latinas, emojis):

- Página única: 41 caracteres
- Múltiplas páginas: 15 páginas × 41 = 615 caracteres no máximo

Plataforma OWL:

- Texto da mensagem limitado a 500 caracteres
- A interface da web exibe a contagem de caracteres restantes e avisa quando se aproxima dos limites

Monitoramento e Logs

A atividade de Transmissão de Células é registrada para fins de auditoria:

- Eventos de criação, atualização e exclusão de mensagens
- Usuários iniciadores e aprovadores
- Carimbos de data/hora e identificadores de mensagem
- Respostas e erros da API do CBC
- Detalhes de direcionamento geográfico

Acesse logs via o Log de Atividades ou consultas ao banco de dados:

```
SELECT * FROM cbc
WHERE created >= '2025-01-01'
ORDER BY created DESC;
```

Integração com Dispositivos Móveis

Mensagens de Transmissão de Células são recebidas por dispositivos móveis compatíveis:

Suporte a Dispositivos:

- A maioria dos smartphones de 2015 em diante suporta Transmissão de Células
- Telefones simples podem ter suporte limitado
- O dispositivo deve estar conectado à rede (nenhum crédito de dados/SMS necessário)
- Funciona mesmo durante congestionamento de rede quando SMS falha

Experiência do Usuário:

- O alerta é exibido como uma notificação em tela cheia
- Um tom de alerta único toca
- O alerta persiste até ser reconhecido
- Nenhuma assinatura do usuário é necessária
- Não pode ser bloqueado pelos usuários para alertas presidenciais/extremos

Testando a Recepção do Dispositivo:

Para verificar se os dispositivos podem receber alertas:

1. Envie uma mensagem de teste (identificador 4379 ou 4380-4381)
2. Certifique-se de que o dispositivo esteja na área de rastreamento alvo
3. Verifique se o dispositivo tem a Transmissão de Células habilitada nas configurações
4. Verifique com vários modelos de dispositivos e versões de SO

Funcionalidade Adicional

A plataforma OWL pode ser estendida com recursos opcionais para complementar a mensagem de Transmissão de Células:

SMS / Texto em Massa

Envio de SMS regular para indivíduos para notificações suplementares.

- Muitas vezes usado para alertas rebaixados para informar as pessoas que a ameaça imediata passou, de uma maneira menos intrusiva
- Muito mais lento que a Transmissão de Células, mas pode incluir um mecanismo de confirmação para verificar se a mensagem foi recebida
- Útil para comunicações de acompanhamento direcionadas

Chamadas de Voz para Linha Fixa

Chamar automaticamente números de linha fixa e reproduzir mensagens de aviso de emergência.

- Reprodução de mensagem pré-gravada
- Leitura em voz alta da mensagem de aviso de emergência
- Alcança populações sem telefones móveis
- Pode verificar a entrega da mensagem através da conclusão da chamada

Publicação Cruzada em Mídias Sociais

Publicar automaticamente mensagens de Aviso de Emergência em canais oficiais de mídias sociais.

- Estende o alcance além da rede celular
- Fornece referência para aqueles que perderam o alerta inicial
- Permite mensagens estendidas além dos limites de caracteres

Transmissão Automática de Rádio/TV

Transmitir automaticamente mensagens de emergência via rádio e televisão.

- Reprodução de mensagem pré-gravada
- Leitura em voz alta da mensagem de aviso de emergência
- Alcança populações durante quedas de rede
- Complementa a Transmissão de Células para cobertura abrangente

Dispositivos de Alerta Externos

O OWL CBC pode se conectar a uma variedade de fontes externas:

- Plataformas de mídias sociais
- APIs públicas
- Sistemas de chamadas de voz
- Sistemas de transmissão de rádio
- Alarmes físicos e sirenes
- Sinalização eletrônica

Opções de personalização podem ser exploradas como parte da fase de design.

Personalização e Manutenção

Testes Periódicos

Testes periódicos da solução devem ser realizados em intervalos regulares para garantir:

- A solução e todos os componentes estão funcionando corretamente
- Todo o pessoal está familiarizado com os processos e procedimentos necessários para emitir Mensagens de Aviso de Emergência
- Os pontos de integração permanecem operacionais
- Os modelos de mensagem estão atualizados e eficazes

Testes Recomendados:

- Mensagens de teste mensais usando identificadores 4379, 4380-4381
- Testes completos do sistema trimestrais, incluindo fluxos de aprovação
- Testes anuais de aviso público com notificação prévia
- Sessões de treinamento regulares para usuários autorizados

Manutenção de Dados do Site Celular

Quando os operadores adicionam ou removem sites celulares, ou alteram Áreas/Identificadores de Rastreamento de sites celulares, essa informação deve ser compartilhada com a equipe da Omnitouch para garantir que os dados da ferramenta de mapeamento permaneçam precisos.

Integração Automática de Dados

O OWL suporta coleta automática de dados de:

- Nokia NetAct
- Huawei U2000 / U2020
- ZTE NetNumen / ZXPOS
- Ericsson ENM

Atualizações Manuais de Dados

Alternativamente, os dados do site celular podem ser fornecidos à equipe de operações da Omnitouch periodicamente via e-mail em vários formatos.

Frequência de Atualização: Revise e atualize os dados do site celular sempre que ocorrerem mudanças na rede, tipicamente mensal ou trimestral.

Áreas Alvo Pré-definidas

À medida que os limites geográficos mudam, ocorrem desenvolvimentos e áreas de risco se deslocam, há necessidade de revisar as Áreas Alvo Pré-definidas usadas na etapa de Direcionamento.

Revisão Anual: As áreas alvo pré-definidas devem ser revisadas anualmente pelo NDMO (Escritório Nacional de Gerenciamento de Desastres).

Processo de Atualização: Atualizações de limites podem ser definidas usando várias plataformas GIS comuns ou Google Earth, e então fornecidas à equipe de operações da Omnitouch.

Modelos de Mensagem Pré-definidos

Revisão Anual: Os modelos de mensagem pré-definidos devem ser revisados anualmente pelo NDMO para garantir:

- O conteúdo da mensagem reflete os procedimentos de emergência atuais
- As traduções de idiomas são precisas
- Os identificadores de mensagem são apropriados
- As informações de contato e instruções estão atualizadas

Processo de Atualização: Atualizações podem ser fornecidas por e-mail à equipe de operações da Omnitouch.

Fluxos de Aprovação de Mensagens

Diferentes regiões têm diferentes requisitos em relação à emissão de mensagens e fluxos de aprovação.

Regra de Duas Pessoas: O uso da regra de duas pessoas é defendido sempre que prático para garantir supervisão na submissão de mensagens.

Funções de Usuário Granulares: Funções de usuário individuais podem ser configuradas para:

- Permitir que apenas certos usuários enviem mensagens pré-definidas
- Restringir direcionamento a regiões específicas
- Exigir etapas adicionais de aprovação
- Minimizar o risco de uso indevido

Manutenção de Segurança / 2FA

A equipe de operações da Omnitouch pode apoiar com:

- Redefinição de tokens 2FA
- Reemissão de tokens perdidos/danificados/expirados
- Auditoria de segurança e gerenciamento de tokens

Procedimentos de Segurança de Tokens:

Quando os tokens são emitidos, um procedimento é detalhado para os passos a serem tomados caso um token se torne perdido ou não contabilizado. A pessoa que aceita o token deve seguir esses procedimentos para garantir que o sistema não seja abusado.

Ações Imediatas para Tokens Perdidos:

1. Relate o token perdido imediatamente ao pessoal autorizado
2. O token é desativado no sistema
3. Revisão de segurança realizada
4. Novo token emitido após verificação de segurança

Integração de API Externa

Um conjunto completo de APIs está disponível para a Entidade de Transmissão de Células para permitir que sistemas de terceiros se integrem e interajam com o CBE.

Capacidades da API:

- **Relatórios/Monitoramento** - Verificar status e alcance das mensagens transmitidas
- **Criação de Mensagens** - Criar e transmitir novas mensagens programaticamente
- **Verificações de Saúde** - Verificar periodicamente a saúde do sistema com geração de tráfego de mensagens de teste
- **Consultas de Status** - Recuperar status da mensagem, estatísticas de entrega e métricas do sistema

Veja a seção de Referência da API abaixo para documentação detalhada dos endpoints.

Referência da API

Todos os endpoints do CBC requerem autenticação e permissões apropriadas.

Criar Mensagem:

`PUT /crm/cbc/`

Obter Todas as Mensagens:

`GET /crm/cbc/`

Atualizar Mensagem:

`PATCH /crm/cbc/{cbc_message_id}`

Excluir Mensagem:

`DELETE /crm/cbc/{cbc_message_id}`

Veja a documentação do Swagger em `/crm/docs/` para especificações detalhadas da API.

Pesquisa Global

O recurso de Pesquisa Global fornece uma **interface de pesquisa unificada** para encontrar rapidamente clientes, contatos, serviços, inventário e locais em todo o banco de dados do OmniCRM.

Veja também: `Customers <basics_customers>`, `Inventory <administration_inventory>`, `Service Management <csa_service_management>`.

Acessar Pesquisa Global

De qualquer lugar no CRM:

Clique no ícone de pesquisa na barra de navegação superior ou navegue para:

A página de pesquisa global aparece com uma grande caixa de pesquisa e opções de filtro.

Como Funciona

A pesquisa global realiza uma **pesquisa entre entidades** em cinco tipos de dados:

O que é Pesquisado:

1. **Clientes** - Nome do cliente
2. **Contatos** - Primeiro nome, sobrenome, endereço de e-mail, número de telefone
3. **Locais** - Nome do local
4. **Inventário** - Números de série, ICCIDs, identificadores (itemtext1, itemtext2)
5. **Serviços** - Nome do serviço, UUID do serviço

Comportamento da Pesquisa:

- **Correspondência parcial** - Pesquisa por termos que contêm sua consulta (por exemplo, "Smith" corresponde a "John Smith" e "Smithson")
- **Sem distinção entre maiúsculas e minúsculas** - "john" corresponde a "John", "JOHN" e "john"
- **Múltiplas entidades** - Uma única pesquisa retorna resultados de todos os tipos de entidade
- **Resultados paginados** - Mostra 10 resultados por página por padrão

Realizando uma Pesquisa

Pesquisa Básica

1. Digite seu termo de pesquisa na caixa de pesquisa
2. Clique em **"Pesquisar"** ou pressione Enter

Exemplos de termos de pesquisa:

- Nome do cliente: `"Acme Corp"`
- Número de telefone: `"+1234567890"` ou `"1234567890"`
- E-mail: `"john@example.com"` ou `"john"`
- Número de série: `"ICCID8944"` ou apenas `"8944"`
- UUID do serviço: `"123e4567-e89b"`

Incluir Contas Fechadas

Por padrão, a pesquisa retorna apenas resultados de contas de clientes **Abertas**.

Para pesquisar em todas as contas, incluindo as fechadas:

1. Marque a caixa de seleção **"Incluir Contas Fechadas"**
2. Clique em **"Pesquisar"** novamente

Isso irá pesquisar:

- Clientes com `customer_status = "Closed"`
- Contatos, serviços, locais e inventário vinculados a clientes fechados

Casos de uso para pesquisa de contas fechadas:

- Encontrar registros históricos de clientes
- Localizar equipamentos de serviços desprovisionados
- Consultar números de telefone ou serviços antigos
- Recuperar dados de clientes para reativação

Entendendo os Resultados da Pesquisa

Formato de Exibição dos Resultados

Os resultados são exibidos em uma lista rolável mostrando:

> John Smith ID do Cliente: 123 Tipo: cliente
>
> John Smith (Contato) ID do Cliente: 123 Tipo: contato
>
> Móvel - +44 7700 900123 ID do Cliente: 123 Tipo: serviço

Cada resultado mostra:

- **Nome/Título** - O identificador principal (link clicável)

- **ID do Cliente** - O cliente pai ao qual pertence
- **Tipo** - O tipo de entidade (cliente, contato, local, inventário, serviço)

Tipos de Resultados Explicados

Resultados de Clientes:

Clicar abre a página de visão geral do cliente mostrando todos os detalhes, serviços, contatos, etc.

Resultados de Contatos:

Clicar abre a página do cliente com a aba Contatos ativa, rolando para o contato específico.

Resultados de Locais:

Clicar abre a página do cliente com a aba Locais ativa.

Resultados de Inventário:

Clicar abre a página do cliente com a aba Inventário ativa. Se o inventário não estiver atribuído (sem customer_id), ele vincula à lista principal de inventário.

Resultados de Serviços:

Clicar abre a página do cliente com a aba Serviços ativa, destacando o serviço específico.

Navegação a partir dos Resultados

Todos os resultados da pesquisa são **links clicáveis** que navegam diretamente para a página relevante:

Padrão de Link:

- `/customers/{customer_id}` - Registros de clientes
- `/customers/{customer_id}#4` - Contatos (aba 4)
- `/customers/{customer_id}#2` - Locais (aba 2)

- `/customers/{customer_id}#8` - Inventário (aba 8)
- `/customers/{customer_id}#3` - Serviços (aba 3)
- `/inventory-items-list` - Inventário não atribuído

O fragmento de hash (`#`) seleciona automaticamente a aba correta quando a página do cliente é carregada.

Paginação

Os resultados são paginados com 10 itens por página:

> Mostrando resultados 11-20 de 47

Navegue pelas páginas usando:

- Botões **Anterior/Próximo**
- **Números de página** - Clique em uma página específica
- **Teclado** - Setas esquerda/direita (se implementado)

Cenários Comuns de Pesquisa

Cenário 1: Encontrar Cliente pelo Telefone

O usuário liga e fornece o número de telefone.

> Resultados: • John Smith (Contato) - ID do Cliente: 123 • Móvel - 555-0123 (Serviço) - ID do Cliente: 123

Clique em qualquer resultado para acessar a conta do cliente.

Cenário 2: Localizar Cartão SIM

O técnico precisa encontrar qual cliente possui um SIM específico.

> Resultados: • 89445380000000001234 (Inventário) - ID do Cliente: 456

Clique no resultado para ver a atribuição do SIM, detalhes do cliente.

Cenário 3: Encontrar Cliente Inativo

Precisa localizar um cliente que fechou sua conta há 6 meses.

> ☑ Incluir Contas Fechadas
>
> Resultados: • Acme Corporation (cliente) - ID do Cliente: 789

Cenário 4: Pesquisar por E-mail

O cliente envia um e-mail para o suporte, a equipe precisa encontrar sua conta.

> Resultados: • John Smith (Contato) - ID do Cliente: 123

Cenário 5: Encontrar Serviço pelo UUID

O log de provisionamento mostra o UUID do serviço, precisa encontrar qual cliente.

> Resultados: • Móvel - +44 7700 900123 (Serviço) - ID do Cliente: 456

Dicas de Pesquisa

Para Melhores Resultados:

- **Use termos parciais** - "Smith" é melhor do que "John Smith" para resultados mais amplos
- **Tente variações** - Se "John" não funcionar, tente telefone ou e-mail
- **Inclua contas fechadas** - Ao pesquisar dados históricos
- **Seja específico para equipamentos** - Use números de série completos para inventário
- **Pesquise UUID do serviço** - Quando outros identificadores não são conhecidos

O que é Pesquisado (por Entidade):

Clientes:

- Apenas nome do cliente (não endereço, notas ou outros campos)

Contatos:

- Primeiro nome
- Sobrenome
- Endereço de e-mail
- Número de telefone

Locais:

- Apenas nome do local

Inventário:

- itemtext1 (tipicamente ICCID, número de série, endereço MAC)
- itemtext2 (tipicamente IMSI, identificador secundário)
- *Nota: Não pesquisa itemtext3-20 ou notas de inventário*

Serviços:

- Nome do serviço
- UUID do serviço

O que Não é Pesquisado:

- Endereços de clientes
- Notas de clientes
- Descrições de transações
- Detalhes de faturas
- Logs de provisionamento
- Entradas de log de atividades
- Notas de inventário (além de itemtext1/2)

Referência da API

Endpoint de Pesquisa Global

```
GET /utilities/search_everything?
search=Smith&page=1&per_page=10&search_closed_records=false
Authorization: Bearer <token>
```

Parâmetros de Consulta:

- `search` (obrigatório) - O termo de pesquisa
- `page` (opcional) - Número da página (padrão: 1)
- `per_page` (opcional) - Resultados por página (padrão: 10)
- `search_closed_records` (opcional) - Incluir contas fechadas (padrão: false)

Resposta:

```
{
  "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
  }
}
```


Lógica de Pesquisa (Backend):

O backend realiza uma união SQL entre todas as tabelas de entidades:

```sql
-- Clientes
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

-- Contatos
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

-- Locais
SELECT site_id AS id,
       site_name AS name,
       customer_id,
       'site' AS type
FROM customer_site
WHERE site_name LIKE '%Smith%'

UNION ALL

-- Inventário
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

-- Serviços
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%'
```

Os resultados são então paginados e retornados.

Considerações de Desempenho

Desempenho da Pesquisa:

- As pesquisas usam consultas LIKE com curingas (`%term%`)
- Nenhum índice de texto completo atualmente implementado
- Bancos de dados grandes (>100k clientes) podem experimentar pesquisas mais lentas
- Resultados limitados a 10 por página para desempenho

Dicas de Otimização:

- Seja específico com os termos de pesquisa para reduzir o conjunto de resultados
- Use o filtro de contas fechadas para reduzir o escopo da pesquisa
- Considere adicionar índices de banco de dados em campos frequentemente pesquisados

Solução de Problemas

Nenhum resultado encontrado (mas o registro existe)

- **Causa:** O termo de pesquisa não corresponde ao formato de dados armazenados

- **Exemplos:**
  - Telefone armazenado como "+44 7700 900123", pesquisar "07700900123" não irá corresponder
  - E-mail armazenado como "<john.smith@example.com>", pesquisar "john" não irá corresponder
- **Solução:** Tente variações, use correspondências parciais que definitivamente existem

Pesquisa muito lenta

- **Causa:** Banco de dados grande, consulta complexa entre várias tabelas
- **Solução:**
  - Use termos de pesquisa mais específicos
  - Limite apenas a contas abertas (desmarque contas fechadas)
  - Entre em contato com o administrador sobre indexação de banco de dados

Resultados vinculados ao cliente errado

- **Causa:** Vários clientes/contatos com o mesmo nome
- **Solução:** Use o ID do Cliente para diferenciar, ou pesquise por identificador único (e-mail, telefone)

Contas fechadas não aparecem

- **Causa:** Caixa de seleção "Incluir Contas Fechadas" não marcada
- **Solução:** Marque a caixa e pesquise novamente

Documentação Relacionada

- `basics_customers` - Gestão de clientes
- `basics_navigation` - Navegação geral
- `administration_inventory` - Pesquisas de inventário

Sistema de Recarregamento e Top-Up

O sistema de Top-Up do OmniCRM fornece um **portal de recarga pré-paga de autoatendimento** para que os clientes adicionem crédito ou estendam a validade do serviço através do `Self-Care Portal <self_care_portal>`. Este recurso é comumente utilizado para:

- **Serviços de dados móveis** - Cartões SIM pré-pagos e serviços apenas de dados
- **Serviços de hotspot** - Dongles de hotspot WiFi e dispositivos de internet portáteis
- **Serviços fixos sem fio** - Acesso à internet pré-pago

Veja também: `SIM Card Provisioning <concepts_sim_provisioning>` para informações sobre a provisão de serviços móveis e gerenciamento de inventário de SIM.

Visão Geral

O sistema de top-up permite que os clientes comprem dias adicionais de serviço através de um processo de checkout simplificado e em várias etapas com processamento de pagamento integrado via Stripe.

Principais Recursos:

- Portal de autoatendimento para clientes (sem intervenção da equipe necessária)
- Seleção de duração flexível (1-30 dias)
- Exibição de uso em tempo real antes da compra
- Processamento de pagamento seguro com Stripe
- Reembolsos automáticos se a recarga falhar

- Geração de faturas e transações
- Integração com sistema de provisão para ativação de serviços

Acessar o Portal de Top-Up

O portal de top-up é acessado através de uma **URL pública** que os clientes podem visitar sem fazer login no CRM:

Como os Clientes Acessam:

- Link direto enviado via SMS quando o saldo está baixo
- Código QR em materiais impressos
- Link no portal de autoatendimento
- Compartilhado via suporte ao cliente

O portal detecta automaticamente o serviço do cliente com base no endereço IP ou IMSI solicitado.

Processo de Top-Up

O fluxo de top-up consiste em **4 etapas**:

Etapa 1: Seleção de Dados

Os clientes selecionam quantos dias de serviço desejam comprar.

Interface:

- **Controle deslizante** - Selecione de 1 a 30 dias
- **Cálculo de preço ao vivo** - Mostra o custo total com base na seleção
- **Exibição da data de expiração** - Calcula e mostra quando o serviço expirará
- **Exibição de uso atual** - Mostra o saldo restante/data de expiração antes do top-up

Exibição de Exemplo:

Configuração de Preços:

- O preço por dia é configurado via variável de ambiente `REACT_APP_TOPUP_PRICE_PER_DAY`
- Padrão: $10 USD por dia
- A moeda é definida via `REACT_APP_CURRENCY_CODE`

Etapa 2: Informações de Cobrança

Os clientes fornecem seus dados de contato para a transação:

- **Nome**
- **Sobrenome**
- **Endereço de E-mail**

Essas informações são usadas para:

- Geração de fatura
- E-mail de recibo de pagamento
- Registros de transação
- Processamento de reembolso (se necessário)

Etapa 3: Pagamento

Processamento de pagamento seguro via **formulários de pagamento hospedados pelo fornecedor** (Stripe Elements, PayPal SDK).

Métodos de Pagamento Suportados:

- Cartões de crédito (Visa, Mastercard, Amex)
- Cartões de débito
- Carteiras digitais (Apple Pay, Google Pay, PayPal) *se habilitado pelo fornecedor de pagamento*

Recursos de Segurança:

- Integração com fornecedor de pagamento compatível com PCI

- Nenhum detalhe do cartão armazenado no OmniCRM
- Suporte à autenticação 3D Secure
- Transmissão de pagamento criptografada

Fluxo de Pagamento:

1. Formulário de pagamento seguro exibido com entrada do cartão
2. O cliente insere os detalhes do pagamento
3. Pagamento processado através do fornecedor configurado
4. Cartão cobrado imediatamente
5. Sucesso/falha do pagamento tratado

Nota

Se o pagamento for bem-sucedido, mas a provisão do top-up falhar (por exemplo, erro de rede, OCS inacessível), o sistema inicia automaticamente um **reembolso total** para o método de pagamento do cliente.

Etapa 4: Conclusão

Tela de Sucesso:

> Seu serviço foi estendido. Nova data de expiração: 17 Jan 2025
>
> Recibo enviado para: <customer@example.com> ID da Transação: TXN-123456

Tela de Falha:

Se o top-up falhar, o sistema exibe um erro e processa automaticamente um reembolso:

> Não conseguimos completar seu top-up. Seu pagamento foi reembolsado.
>
> Erro: Não foi possível conectar ao sistema de cobrança
>
> Por favor, tente novamente ou entre em contato com o suporte.

Processamento em Segundo Plano

Quando um cliente completa o pagamento, o seguinte acontece automaticamente:

1. Validação do Pagamento

O sistema valida:

- O status do Payment Intent é `succeeded`
- O valor do pagamento corresponde aos dias selecionados (`days × price_per_day`)
- O Payment Intent não foi processado antes (previne top-up duplicado)

2. Operação de Top-Up

```
- API endpoint: POST /oam/topup_dongle
- Valida service_uuid e IMSI
- Chama OCS/CGRateS para adicionar saldo
- Cria trabalho de provisão (play_topup_hotspot)
```

3. Criação de Registro

O sistema cria múltiplos registros no banco de dados:

- **Registro HotspotTopup** - Rastreia a transação de top-up
  - payment_intent_id
  - service_uuid
  - imsi
  - dias comprados
  - topup_amount
  - status (Success/Failed/Refunded)
- **Registro de Transação** - Transação financeira
  - Título: "Hotspot Topup - 7 Dias"
  - Valor: topup_amount (positivo)
  - Vinculado a service_id e customer_id
- **Registro de Fatura** - Fatura de pagamento

    - Contém a transação de top-up
    - Marcado como pago imediatamente
    - Referência de pagamento: Stripe payment_intent_id
- **Transação de pagamento** - Transação de crédito compensatória
    - Título: "Pagamento por [Título da Fatura]"
    - Valor: topup_amount (negativo - crédito)
    - Vincula o pagamento da fatura à conta do cliente

4. Trabalho de Provisão

Um trabalho de provisão é criado com o playbook `play_topup_hotspot` que:

- Conecta-se à API OCS/CGRateS
- Adiciona saldo à conta
- Estende a data de expiração
- Cria entrada de log de atividade
- Envia notificação de confirmação (se configurado)

A API aguarda a conclusão da provisão (polling com intervalos de 0.2s, máximo de 25 iterações) antes de retornar sucesso ao cliente.

5. Reembolso Automático em Caso de Falha

Se qualquer etapa falhar após o pagamento:

```
if topup_provisioning_failed:
    refund = stripe.Refund.create(
        payment_intent=payment_intent_id,
        reason='requested_by_customer'  # Reembolso automático do
sistema
    )
    status_message = "Topup Falhou. Reembolsando pagamento..."
```

O reembolso é processado automaticamente e o cliente é notificado na tela.

Endpoints da API

Endpoint de Top-Up

```
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
}
```

Resposta (Sucesso):

```
{
  "result": "OK",
  "status": 200,
  "provision_id": 456,
  "payment_intent_id": "pi_1234567890abcdef",
  "service_uuid": "123e4567-e89b-12d3-a456-426614174000",
  "invoice_id": 789
}
```

Resposta (Falha):

```
{
  "result": "Failed",
  "Reason": "Timeout de conexão OCS",
  "status": 500
}
```

Verificações de Validação:

- Todos os campos obrigatórios presentes (service_uuid, imsi, days, payment_intent_id, topup_amount)

- topup_amount corresponde a dias: `topup_amount × 100 == days × 1000` (em centavos)
- Payment Intent existe no Stripe
- O valor do Payment Intent corresponde: `payment_intent.amount == topup_amount × 100`
- O status do Payment Intent é `succeeded`
- O Payment Intent não foi processado anteriormente (verifica a tabela `HotspotTopup`)

Endpoint de Uso

Recupera informações de uso e serviço atuais para o cliente:

```
GET /oam/usage
```

Resposta:

```
{
  "imsi": "310120123456789",
  "service": {
    "service_uuid": "123e4567-e89b-12d3-a456-426614174000",
    "service_name": "Dados Móveis - 0412345678",
    "service_status": "Ativo"
  },
  "balance": {
    "expiry": "2025-01-10T23:59:59Z",
    "unlimited": true
  },
  "requestingIp": "203.0.113.45"
}
```

Este endpoint usa o endereço IP solicitado para identificar automaticamente o serviço do cliente.

Configuração

Variáveis de Ambiente

Configure estas no arquivo `.env` do OmniCRM-UI:

```
# Configuração do Portal de Top-Up
REACT_APP_TOPUP_PRICE_PER_DAY=10
REACT_APP_CURRENCY_CODE=AUD
REACT_APP_SELF_CARE_NAME="SuaEmpresa"

# Configuração do Stripe
REACT_APP_STRIPE_PUBLISHABLE_KEY=pk_live_...
```

Configuração do Stripe

O sistema de top-up utiliza Payment Intents do Stripe:

1. **Habilite Payment Intents** no seu Painel do Stripe
2. **Configure Webhook** para receber atualizações de status de pagamento (opcional mas recomendado)
3. **Configure métodos de pagamento** (cartões, carteiras, etc.)
4. **Modo de teste** - Use chaves de teste para desenvolvimento

```
# Desenvolvimento
REACT_APP_STRIPE_PUBLISHABLE_KEY=pk_test_...

# Produção
REACT_APP_STRIPE_PUBLISHABLE_KEY=pk_live_...
```

Configuração do Playbook

O playbook de provisão `play_topup_hotspot.yaml` deve ser configurado para:

- Aceitar a variável `days`
- Conectar-se à API OCS/CGRateS

- Adicionar saldo à conta
- Atualizar a data de expiração do serviço

Estrutura de exemplo do 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 }}"
```

Notificações de Saldo Baixo

O sistema pode enviar notificações automáticas quando o saldo do cliente estiver baixo:

Notificações SMS:

Quando acionadas por eventos do OCS (`Action_Balance_Low`, `Action_Balance_Out`, `Action_Balance_Expired`):

Notificações por E-mail:

Configuradas nos planos de ação do OCS/CGRateS para enviar alertas de saldo.

Gatilhos de Notificação:

- `Action_Balance_Low` - Saldo abaixo do limite (por exemplo, 2 dias restantes)
- `Action_Balance_Out` - Saldo esgotado
- `Action_Balance_Expired` - Serviço expirado

Cada notificação inclui o link do portal de top-up para fácil acesso do cliente.

Solução de Problemas

Problemas Comuns

"Sistema de pagamento indisponível"

- **Causa:** A biblioteca Stripe falhou ao carregar ou chave publicável inválida
- **Solução:**
  - Verifique se `REACT_APP_STRIPE_PUBLISHABLE_KEY` está configurado corretamente
  - Verifique se a conta do Stripe está ativa
  - Verifique o console do navegador para erros de JavaScript

"Top-up falhou. Reembolsando pagamento..."

- **Causa:** Trabalho de provisão falhou (OCS inacessível, erro de playbook, etc.)
- **Solução:**
  - Verifique os logs de provisão: `GET /crm/provision/provision_id/<id>`
  - Verifique se a API OCS/CGRateS é acessível
  - Revise o playbook `play_topup_hotspot.yaml` em busca de erros
  - Verifique os logs do Ansible

"Payment intent já processado"

- **Causa:** Cliente tentando reutilizar o mesmo pagamento (por exemplo, atualizar após sucesso)
- **Solução:** Isso está funcionando conforme o projetado para evitar cobrança duplicada. O cliente deve iniciar um novo top-up se necessário.

"O valor do payment intent não corresponde"

- **Causa:** Desvio entre o cálculo da UI e a validação do backend
- **Solução:**

- Verifique se `REACT_APP_TOPUP_PRICE_PER_DAY` corresponde à expectativa do backend (padrão $10)
- Verifique se a configuração da moeda é consistente
- Limpe o cache do navegador e tente novamente

Monitoramento de Top-Ups

Visualizar Registros de Top-Up:

Consulta a tabela `HotspotTopup` para ver todas as tentativas de top-up:

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

Verificar Status de Provisão:

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

Mostra o status detalhado do trabalho de provisão de top-up.

Painel do Stripe:

Monitore pagamentos, reembolsos e transações falhadas no seu Painel do Stripe em <https://dashboard.stripe.com>

Considerações de Segurança

Segurança de Pagamento:

- Todos os dados do cartão são tratados pelo Stripe (compatível com PCI Nível 1)
- Nenhum dado sensível de pagamento armazenado no banco de dados do OmniCRM
- Payment Intents previnem cobranças não autorizadas
- Validação de valores tanto no lado do cliente quanto no servidor

Prevenção de Fraude:

- Detecção de Payment Intent duplicado previne cobrança duplicada
- Rastreamento de endereço IP para correlação de uso
- Validação de IMSI garante que o top-up vá para o serviço correto
- Reembolsos automáticos limitam a exposição financeira

Controle de Acesso:

- O portal de top-up é público (por design - os clientes precisam de acesso)
- O endpoint de uso requer identificação de serviço válida (IP ou IMSI)
- A validação em segundo plano previne top-ups arbitrários de serviços
- O administrador pode visualizar todos os registros de top-up através da interface do CRM

Melhores Práticas

Para Operadores:

1. **Teste o fluxo de reembolso** - Teste regularmente cenários de falha para garantir que os reembolsos funcionem
2. **Monitore top-ups falhados** - Configure alertas para altas taxas de falha
3. **Mantenha os playbooks simples** - Os playbooks de top-up devem ser rápidos e confiáveis
4. **Verifique a conectividade do OCS** - Garanta que a API OCS esteja sempre acessível
5. **Revise preços** - Atualize `REACT_APP_TOPUP_PRICE_PER_DAY` conforme necessário

Para Clientes:

1. **Salve a URL de top-up** - Acesso rápido quando necessário
2. **Guarde notificações de saldo baixo** - SMS contém link direto
3. **Mantenha o e-mail atualizado** - Recibos enviados para o e-mail cadastrado
4. **Verifique a expiração antes de viajar** - Faça top-up antes de sair da área de cobertura

Para Desenvolvedores:

1. **Gerencie webhooks do Stripe** - Implemente manipuladores de webhook para atualizações de status de pagamento
2. **Implemente idempotência** - Sempre verifique payment_intent_id antes de processar
3. **Registre extensivamente** - Falhas de top-up precisam de informações detalhadas de solução de problemas
4. **Teste caminhos de erro** - Verifique se a automação de reembolso funciona corretamente
5. **Monitore desempenho** - O polling de provisão deve ser concluído em <5 segundos

Documentação Relacionada

- `payments_process` - Processamento de pagamento geral
- `concepts_provisioning` - Visão geral do sistema de provisão
- `Payment System Guide <payment_system_guide>` - Detalhes de integração do fornecedor de pagamento
- `payments_transaction` - Gerenciamento de transações
- `payments_invoices` - Manipulação de faturas

Guia Completo do Ciclo de Vida do Produto

Este guia fornece um passo a passo do ciclo de vida do produto no OmniCRM, desde a criação de uma definição de produto até a provisão de serviços, adição de complementos e desprovisionamento. Abordaremos a estratégia de preços, a integração com Ansible e forneceremos exemplos do mundo real ao longo do caminho.

Visão Geral: A Jornada do Produto ao Serviço

O ciclo de vida de um produto no OmniCRM segue estas etapas:

1. **Definição do Produto** - O administrador cria um modelo de produto com regras de preços e provisão
2. **Criação do Serviço** - O cliente solicita o produto, o sistema provisiona a instância do serviço
3. **Ciclo de Vida do Serviço** - O cliente usa o serviço, adiciona complementos/top-ups, modifica o serviço
4. **Desprovisionamento** - O serviço é encerrado, os recursos são liberados

Compreendendo Preços: Atacado vs Varejo

Todo produto e serviço no OmniCRM tem duas dimensões de preços: **atacado** e **varejo**.

Custo de Atacado

O custo de atacado representa o custo real para entregar o serviço:

- Custos de infraestrutura e largura de banda
- Taxas de licenciamento
- Custos de equipamentos
- Despesas operacionais

Custo de Varejo

O custo de varejo é o valor cobrado ao cliente.

Custos de Configuração

Tanto o atacado quanto o varejo têm variantes de custo de configuração para cobranças de provisão únicas:

- `wholesale_setup_cost` - Seu custo para provisionar
- `retail_setup_cost` - Valor cobrado ao cliente pela ativação

Exemplo:

```
{
  "retail_cost": 15.00,
  "wholesale_cost": 5.00,
  "retail_setup_cost": 0.00,
  "wholesale_setup_cost": 1.00
}
```

Etapa 1: Criando uma Definição de Produto

Os produtos são modelos que definem o que é provisionado e como os clientes são cobrados.

Criando um Produto de SIM Móvel

Vamos criar um produto de SIM móvel pré-pago com 20GB de dados por mês.

Passo 1: Navegar até a Gestão de Produtos

Na interface do administrador, vá para **Produtos** → **Criar Produto**.

Passo 2: Definir Informações Básicas

```
{
  "product_name": "Prepaid Mobile 20GB",
  "product_slug": "prepaid-mobile-20gb",
  "category": "standalone",
  "service_type": "mobile",
  "enabled": true,
  "icon": "fa-solid fa-sim-card",
  "comment": "Prepaid mobile SIM with 20GB data, unlimited calls & texts"
}
```

Explicações dos Campos:

- `product_name` - Nome voltado para o cliente exibido no catálogo
- `product_slug` - Identificador seguro para URL usado em chamadas de API e links
- `category` - "standalone" significa que isso cria um novo serviço (vs complemento/pacote)
- `service_type` - Agrupa produtos relacionados, usado para filtragem de complementos
- `enabled` - Deve ser verdadeiro para que o produto possa ser solicitado
- `icon` - Ícone do FontAwesome exibido na interface
- `comment` - Notas internas para referência da equipe

Passo 3: Definir Preços

```
{
  "retail_cost": 15.00,
  "wholesale_cost": 5.00,
  "retail_setup_cost": 0.00,
  "wholesale_setup_cost": 1.00,
  "contract_days": 30
}
```

Detalhamento dos Preços:

- Receita mensal por cliente: £15.00
- Custo mensal para entrega: £5.00
- Margem de lucro mensal: £10.00 (200% de markup, 67% de margem)
- Lucro de configuração: -£1.00 (subsidiado para atrair clientes)
- Duração do contrato: 30 dias (renovação mensal)

Passo 4: Definir Elegibilidade do Cliente

```
{
  "residential": true,
  "business": false,
  "customer_can_purchase": true,
  "available_from": "2025-01-01T00:00:00Z",
  "available_until": null
}
```

- Clientes residenciais podem solicitar
- Clientes empresariais não podem (linha de produtos diferente)
- Compra de autoatendimento habilitada
- Disponível a partir de 1º de janeiro de 2025
- Sem data de término (oferta contínua)

Passo 5: Configurar Renovação Automática

```
{
  "auto_renew": "prompt",
  "allow_auto_renew": true
}
```

- `"prompt"` - Perguntar ao cliente se deseja renovação automática na compra
- `"true"` - Renovar automaticamente sem perguntar
- `"false"` - Nunca renovar automaticamente (apenas recarga manual)

- `allow_auto_renew: true` - O cliente pode habilitar/desabilitar a renovação automática mais tarde

Passo 6: Especificar Requisitos de Inventário

Os requisitos de inventário definem quais recursos físicos ou virtuais devem ser alocados ao provisionar este produto. Esta é uma etapa crítica que conecta seu catálogo de produtos ao seu `Sistema de Gestão de Inventário <administration_inventory>`.

```
{
  "inventory_items_list": "['SIM Card', 'Mobile Number']"
}
```

O que são Itens de Inventário?

Itens de inventário são recursos rastreáveis armazenados no sistema de inventário do OmniCRM. Cada item tem:

- **Tipo** - Definido pelo Modelo de Inventário (por exemplo, "SIM Card", "Mobile Number", "Modem")
- **Atributos únicos** - Números de série, endereços MAC, números de telefone, etc.
- **Estado** - Em Estoque, Atribuído, Descomissionado, etc.
- **Localização** - Localização física ou lógica

Como Funcionam os Requisitos de Inventário:

A `inventory_items_list` é uma lista Python (como uma string) contendo nomes de tipos de inventário. Cada nome deve corresponder exatamente a um nome existente de `Modelo de Inventário <administration_inventory>`.

Exemplo de Requisitos de Inventário:

```
# Produto de SIM Móvel
inventory_items_list: "['SIM Card', 'Mobile Number']"

# Serviço de internet fixa
inventory_items_list: "['Modem Router', 'Static IP Address']"

# Serviço digital (sem itens físicos)
inventory_items_list: "[]"

# Wireless fixo com CPE
inventory_items_list: "['Fixed Wireless CPE', 'IPv4 Address',
'IPv6 Prefix']"
```

O Processo do Seletor de Inventário

Quando um usuário provisiona um produto com requisitos de inventário, o sistema impõe um processo de seleção obrigatória:

1. Botão de Provisão Clicado

Após selecionar o produto, o usuário clica em "Provisionar". Em vez de provisionar imediatamente, o sistema verifica `inventory_items_list`.

2. Modal do Seletor de Inventário Aparece

Se o inventário for necessário, uma caixa de diálogo modal aparece com um menu suspenso separado para cada tipo de inventário:

3. Filtrando o Inventário Disponível

O menu suspenso para cada tipo de inventário mostra apenas itens que são:

- **Tipo Correto** - Corresponde exatamente ao nome do modelo de inventário
- **Status Disponível** - `item_state` é "Novo" ou "Em Estoque" (não "Atribuído" ou "Danificado")
- **Não Atribuído** - `service_id` e `customer_id` são NULL
- **Em Estoque na Localização** - Opcionalmente filtrado pela localização do armazém/loja

Exemplo de Opções do Menu Suspenso:

Para o tipo de inventário "SIM Card", o menu suspenso pode mostrar:

Cada opção exibe:

- ID ou número de referência do inventário
- Identificador principal (`itemtext1` - por exemplo, ICCID para SIM, número para telefone)
- Localização atual (`item_location`)

4. Seleção Necessária para Prosseguir

Regra Crítica: O provisionamento NÃO pode prosseguir sem selecionar todos os itens de inventário necessários.

- O botão "Continuar" está desativado até que todos os menus suspensos tenham seleções
- O usuário deve selecionar um item para cada tipo de inventário
- O sistema valida as seleções antes de prosseguir

5. Inventário Selecionado Passado para Ansible

Uma vez que o usuário clica em "Continuar", os IDs de inventário selecionados são passados para o playbook do Ansible como variáveis:

```
# Usuário selecionou:
# - ID do inventário do SIM Card: 5001
# - ID do inventário do Mobile Number: 5002

# Variáveis passadas para Ansible:
{
    "product_id": 42,
    "customer_id": 123,
    "SIM Card": 5001,           # ID do Inventário
    "Mobile Number": 5002,       # ID do Inventário
    "access_token": "eyJ..."
}
```

Nota: O nome da variável corresponde exatamente ao tipo de inventário. O playbook usa `hostvars[inventory_hostname]['SIM Card']` para acessar o ID

do inventário.

6. Playbook Busca Detalhes Completos do Inventário

O playbook do Ansible usa o ID do inventário para buscar detalhes completos:

```
- name: Get SIM Card details from inventory
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ hostvars[inventory_hostname]['SIM
Card'] }}"
    method: GET
    headers:
      Authorization: "Bearer {{ access_token }}"
  register: api_response_sim

- name: Extract ICCID and IMSI
  set_fact:
    iccid: "{{ api_response_sim.json.itemtext1 }}"
    imsi: "{{ api_response_sim.json.itemtext2 }}"
```

Agora o playbook tem todos os detalhes do SIM (ICCID, IMSI, etc.) para provisionar o assinante no HSS.

7. Estado do Inventário Mudado para "Atribuído"

Após o registro do servi��o ser criado, o playbook atualiza o inventário para vinculá-lo ao serviço:

```
- name: Assign SIM Card to Service
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ hostvars[inventory_hostname]['SIM
Card'] }}"
    method: PATCH
    body:
      {
        "service_id": "{{
service_creation_response.json.service_id }}",
        "customer_id": "{{ customer_id }}",
        "item_state": "Assigned"
      }
```

Importante: A atribuição de inventário acontece **durante a execução do playbook** como uma tarefa específica, NÃO quando o botão de provisão é clicado. Isso significa:

- **Risco de Dupla Atribuição**: Entre clicar em "Provisionar" e o inventário ser atribuído, outro usuário poderia teoricamente selecionar o mesmo item de inventário
- **Melhor Prática**: Para operações de alto volume, implemente bloqueio de inventário ou use transações de banco de dados
- **Rollback em Caso de Falha**: Se o playbook falhar antes da atribuição do inventário, o inventário permanece não atribuído e disponível para reutilização

Por que não atribuir mais cedo?

O inventário não é atribuído quando "Provisionar" é clicado porque:

1. **ID do Serviço Necessário**: O `service_id` não existe até que o serviço seja criado no playbook
2. **Simplicidade de Rollback**: Se o provisionamento falhar cedo (por exemplo, a criação da conta OCS falhar), o inventário não precisa de limpeza
3. **Flexibilidade**: O playbook pode decidir não atribuir inventário com base em lógica condicional

Tratando Provisionamentos Falhados:

Quando um provisionamento falha após o inventário ser atribuído, o bloco de resgate deve liberar o inventário:

```
rescue:
  - name: Release inventory on failure
    uri:
      url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ hostvars[inventory_hostname]['SIM
Card'] }}"
      method: PATCH
      body:
        {
          "service_id": null,
          "customer_id": null,
          "item_state": "In Stock"
        }
    when: service_id is defined  # Somente se o serviço foi criado
```

Isso garante que o inventário não fique em um estado "Atribuído" para um serviço inexistente ou falhado.

Quando a Lista de Inventário está Vazia

Se `inventory_items_list: "[]"` (lista vazia), o seletor de inventário é totalmente ignorado e o provisionamento prossegue imediatamente. Isso é comum para:

- **Produtos digitais** - Licenças de software, contas de VPN
- **Complementos de serviço** - Recargas de dados que não precisam de novo hardware
- **Serviços virtuais** - Que não consomem recursos rastreáveis

Exemplo: Um complemento "5GB Data Boost" tem `inventory_items_list: "[]"` porque apenas adiciona saldo a um serviço existente sem precisar de novo hardware.

Configuração do Modelo de Inventário

Antes de usar um tipo de inventário em `inventory_items_list`, você deve criar o Modelo de Inventário:

1. Navegue até **Administração → Inventário → Modelos**
2. Crie um modelo com o nome exato (por exemplo, "SIM Card")
3. Defina os campos:
   - `itemtext1_label`: "ICCID"
   - `itemtext2_label`: "IMSI"
   - `itemtext3_label`: "Código PUK"
4. Adicione itens de inventário desse tipo ao estoque

Para detalhes completos sobre como criar e gerenciar modelos de inventário, consulte `Gestão de Inventário <administration_inventory>`.

Múltiplos Itens do Mesmo Tipo

Embora a `inventory_items_list` seja um array, ter tipos duplicados (por exemplo, `"['SIM Card', 'SIM Card']"`) **não é recomendado** pois pode causar confusão na interface e na nomeação de variáveis do playbook.

Para cenários que requerem múltiplos itens semelhantes:

Opção 1: Criar nomes de modelos de inventário distintos

```
# Serviço de telefone Dual-SIM
inventory_items_list: "['Primary SIM Card', 'Secondary SIM Card',
'Mobile Number']"
```

Crie modelos separados: "Primary SIM Card" e "Secondary SIM Card" com os mesmos campos, mas nomes diferentes.

Opção 2: Usar um único item de inventário agrupado

```
# Kit Dual-SIM
inventory_items_list: "['Dual SIM Kit', 'Mobile Number']"
```

Onde o modelo de inventário "Dual SIM Kit" tem campos para ambos os SIMs (`itemtext1`: ICCID Primário, `itemtext2`: ICCID Secundário, etc.).

Cenários Comuns de Inventário

Serviço Móvel:

```
inventory_items_list: "['SIM Card', 'Mobile Number']"
```

- SIM Card: Físico ou eSIM com ICCID/IMSI
- Mobile Number: Número de telefone (MSISDN)

Internet Fixa:

```
inventory_items_list: "['Modem Router', 'Static IP Address']"
```

- Modem Router: Dispositivo CPE com endereço MAC
- Static IP Address: IPv4 do pool de endereços

Wireless Fixo:

```
inventory_items_list: "['Fixed Wireless CPE', 'IPv4 Address',
'IPv6 Prefix']"
```

- CPE: Equipamento nas instalações do cliente (antena, modem)
- IPv4: Endereço IP público
- IPv6 Prefix: prefixo /56 ou /64

Nota: Compromissos e agendamentos **não** são itens de inventário. Use sistemas de agendamento/calendário separados para compromissos de instalação.

Serviço VoIP:

```
inventory_items_list: "['DID Number']"
```

- DID Number: Número de telefone de Discagem Direta

Nota: Nomes de usuário SIP, senhas e configurações de conta são **gerados programaticamente** pelo playbook de provisão, não selecionados do inventário.

GPON/Fibra:

```
inventory_items_list: "['ONT Device', 'GPON Port', 'IPv4 Address', 'Fiber Drop Cable']"
```

- ONT Device: Terminal de Rede Óptica com número de série
- GPON Port: Porta específica no OLT com conexão de fibra
- IPv4 Address: IP público ou privado
- Fiber Drop Cable: Cabo de fibra física da rua até as instalações (rastreado para gestão de ativos)

Aluguel de Equipamentos:

```
inventory_items_list: "['Rental Modem']"
```

- Rastreia qual modem está com qual cliente
- Importante para recuperar equipamentos em caso de cancelamento

Por que os Requisitos de Inventário Importam

1. Prevenir Dupla Atribuição

Sem rastreamento de inventário, você poderia acidentalmente:

- Atribuir o mesmo cartão SIM a dois clientes
- Alocar o mesmo endereço IP a vários serviços
- Enviar o mesmo número de série de equipamento para diferentes locais

O seletor de inventário garante que cada item seja atribuído a exatamente um serviço.

2. Rastro de Auditoria

A atribuição de inventário cria um rastro de auditoria completo:

- Qual cartão SIM está com qual cliente
- Quando foi atribuído
- Qual serviço está usando qual número de telefone
- Histórico de equipamentos (quem teve, quando, para qual serviço)

3. Planejamento de Recursos

Rastreie os níveis de inventário:

- Alerta quando os cartões SIM estão baixos
- Reordene antes de esgotar o estoque
- Planeje os horários dos técnicos com base na disponibilidade do CPE
- Gerencie a alocação de espaço de endereços IP

4. Rastreio de Custos

Vincule o custo de atacado a um item específico:

- Rastreie o custo de cada cartão SIM
- Calcule a depreciação de equipamentos
- Identifique itens perdidos ou roubados
- COGS (Custo das Mercadorias Vendidas) preciso

5. Desprovisionamento

Quando o serviço é cancelado, o inventário pode ser:

- Liberado de volta ao estoque (cartões SIM, modems)
- Retirado (equipamentos danificados)
- Devolvido ao fornecedor (equipamentos alugados)
- Mantido por um período de carência (números de telefone antes da liberação)

Resolvendo Problemas com o Seletor de Inventário

Problema: Mensagem "Nenhum inventário disponível" aparece

Causas:

- Nenhum item de inventário do tipo requerido existe no banco de dados
- Todos os itens já estão "Atribuídos" a outros serviços
- Itens estão marcados como "Danificados" ou "Fora de Serviço"
- Nome do modelo de inventário não corresponde exatamente (sensível a maiúsculas)

Solução:

1. Verifique se o modelo de inventário existe: **Administração → Inventário → Modelos**
2. Verifique se o nome do modelo corresponde exatamente (incluindo espaços, maiúsculas)
3. Adicione itens de inventário desse tipo: **Administração → Inventário → Adicionar Item**
4. Verifique se os itens estão em estado "Novo" ou "Em Estoque"
5. Verifique se os itens não estão já atribuídos (`service_id` deve ser NULL)

Problema: O seletor de inventário não aparece

Causas:

- `inventory_items_list` está vazio: `"[]"`
- `inventory_items_list` é NULL ou não está definido
- A categoria do produto é "complemento" e herda o inventário do serviço pai

Solução:

- Se o inventário for necessário, defina `inventory_items_list: "['Type1', 'Type2']"`
- Verifique se a definição do produto foi salva corretamente
- Verifique a resposta da API para o produto que inclui inventory_items_list

Problema: O playbook falha com "inventário não encontrado"

Causas:

- O playbook referencia o nome da variável errada
- O ID do inventário não foi passado corretamente
- O inventário foi excluído entre a seleção e o provisionamento

Solução:

- Verifique se o playbook usa a variável correta: `hostvars[inventory_hostname]['SIM Card']`
- Verifique se a variável é um inteiro: `{{ hostvars[inventory_hostname]['SIM Card'] | int }}`
- Adicione tratamento de erro no playbook para inventário ausente

Consulte `Gestão de Inventário <administration_inventory>` para detalhes completos sobre como criar modelos, adicionar itens e gerenciar níveis de estoque.

Passo 7: Definir Recursos e Termos

Recursos e termos são conteúdos de marketing e legais voltados para o cliente que ajudam os clientes a entender o que estão comprando e as obrigações envolvidas.

```
{
  "features_list": "20GB High-Speed Data. Unlimited Calls & Texts. EU Roaming Included. No Contract. 30-Day Expiry",
  "terms": "Credit expires after 30 days. Data, calls, and texts valid only within expiry period. Fair use policy applies. See website for full terms."
}
```

Propósito e Valor Comercial

Lista de Recursos - Marketing e Vendas:

A lista de recursos serve a várias funções comerciais críticas:

1. **Diferenciação de Produto** - Ajuda os clientes a comparar rapidamente produtos e escolher o certo

    - "Prepaid Mobile 20GB" vs "Prepaid Mobile 50GB" - os recursos mostram claramente a diferença
    - Sem recursos, os clientes apenas veem o preço, perdendo a proposta de valor
2. **Comunicação de Marketing** - Pontos de venda chave exibidos de forma proeminente
    - "EU Roaming Included" atrai viajantes internacionais
    - "No Contract" atrai clientes avessos a compromissos
    - Recursos impulsionam decisões de compra
3. **Expectativas do Cliente** - Define expectativas claras sobre o que está incluído
    - Reduz chamadas de suporte ("Isso inclui chamadas?" → claramente listado)
    - Prevê mal-entendidos e pedidos de reembolso
    - Constrói confiança através da transparência
4. **Autoatendimento** - Permite que os clientes selecionem produtos apropriados
    - O cliente lê os recursos, entende a oferta, faz uma escolha informada
    - Reduz a necessidade de explicação da equipe de vendas
    - Acelera o processo de compra
5. **SEO e Descobribilidade** - Recursos podem ser indexados para pesquisa
    - O cliente pesquisa "plano móvel de chamadas ilimitadas" → o produto aparece
    - Melhora a capacidade de busca do catálogo de produtos

Termos e Condições - Legal e Conformidade:

Os termos servem a propósitos legais e operacionais:

1. **Proteção Legal** - Protege o negócio de disputas e responsabilidade
    - "O crédito expira após 30 dias" - o cliente não pode exigir reembolso após 31 dias
    - "A política de uso justo se aplica" - previne abusos (tethering de todo o escritório em um plano móvel)
    - Cria um acordo vinculativo
2. **Gestão de Expectativas** - Prevê insatisfação do cliente

    - "Válido apenas dentro do período de validade" - o cliente conhece o prazo de uso
    - "Não pode ser reembolsado" (para complementos) - previne compras fraudulentas
    - Reduz chargebacks e reclamações
3. **Conformidade Regulamentar** - Atende aos requisitos legais
    - As leis de proteção ao consumidor exigem termos claros
    - As regulamentações de telecomunicações exigem divulgação
    - Termos de GDPR/privacidade podem ser referenciados
4. **Limites Operacionais** - Define o escopo e limitações do serviço
    - "Sujeito à cobertura da rede" - não é responsável por zonas mortas
    - "A velocidade pode variar" - gerencia expectativas sobre velocidades "até"
    - "O equipamento deve ser devolvido" - garante a recuperação do equipamento alugado
5. **Rastro de Auditoria** - Prova que o cliente foi informado
    - O cliente aceitou os termos na compra
    - O sistema registra o timestamp de aceitação
    - Defensável em disputas ou processos legais

Exemplo do Mundo Real:

O cliente compra um plano "Chamadas e Textos Ilimitados", depois o usa para telemarketing (10.000 chamadas/dia). Sem termos:

- Cliente: "Você disse ilimitado!"
- Provedor: "Queríamos dizer uso pessoal..."
- Cliente: "Isso não é o que você anunciou!"
- Resultado: Disputa, potencial reclamação regulatória, dano à marca

Com termos: "A política de uso justo se aplica. O serviço é para uso pessoal apenas. Uso comercial proibido."

- Provedor: Aponta para os termos que o cliente aceitou
- O cliente não pode alegar ignorância
- Base legal para suspender o serviço

- Disputa resolvida a favor do provedor

Formato da Lista de Recursos:

Compreender o formato correto é crítico porque **formatação inadequada quebra a exibição da interface**. Os recursos podem aparecer como uma longa string em vez de pontos de lista, ou não serem exibidos de forma alguma.

O campo `features_list` pode ser formatado de duas maneiras:

Opção 1: String Separada por Ponto (Recomendado)

Os recursos são separados por um ponto e espaço (". "). A interface divide nesse delimitador e renderiza cada recurso como um ponto de lista.

Por que esse formato?

- Simples de editar - basta digitar os recursos com pontos entre eles
- Sem caracteres especiais para escapar
- Funciona de forma confiável em todos os componentes da interface
- Fácil de atualizar sem quebrar a sintaxe JSON

Correto vs Incorreto:

Opção 2: String de Array JSON

```
"['20GB High-Speed Data', 'Unlimited Calls & Texts', 'EU Roaming Included']"
```

A interface também pode analisar arrays JSON. Note que isso é uma **string contendo JSON**, não um array JSON real no banco de dados.

Por que esse formato existe?

- Permite recursos com pontos neles (por exemplo, "Até 100Mbps. Sujeito à disponibilidade.")
- Geração programática a partir de scripts/API é mais fácil
- Importado de catálogos de produtos externos que usam arrays

Importante: Isso deve ser uma sintaxe válida de lista Python como string. Aspas simples ao redor de cada item, aspas duplas ao redor da string inteira.

Qual Formato Usar?

- **Separado por ponto** - Para criação manual de produtos na interface (mais simples, menos propenso a erros)
- **Array JSON** - Para criação de produtos baseada em API/script (mais robusto para recursos complexos)

Ambos os formatos produzem uma saída idêntica na interface - eles apenas afetam como você insere os dados.

Onde os Recursos Aparecem na Interface:

1. Catálogo de Produtos (Visão do Cliente)

Quando os clientes navegam pelos produtos disponíveis, os recursos são exibidos como pontos de lista em cada cartão de produto:

2. Página de Detalhes do Produto

Clicando em "Ver Detalhes" mostra todas as informações do produto, incluindo:

- Nome do produto e ícone
- Preços (custo mensal, custo de configuração)
- Lista completa de recursos (pontos de lista)
- Termos e condições (veja abaixo)
- Disponibilidade e elegibilidade

3. Confirmação de Provisão

Durante a provisão, os recursos são mostrados para o usuário revisar antes de confirmar:

> Recursos: • 20GB High-Speed Data • Unlimited Calls & Texts • EU Roaming Included • No Contract • 30-Day Expiry
>
> Custo: £15.00/mês Configuração: £0.00

> [Cancelar] [Confirmar & Provisionar]

4. Detalhes do Serviço (Após Provisão)

Após o serviço estar ativo, os recursos são exibidos na página de detalhes do serviço para referência do cliente.

Formato dos Termos e Condições:

O campo `terms` é texto simples que pode incluir quebras de linha:

Onde os Termos Aparecem na Interface:

1. Página de Detalhes do Produto

Os termos são exibidos em uma seção colapsada que se expande quando clicada:

2. Confirmação de Pedido

Durante a provisão, uma caixa de seleção exige que o usuário aceite os termos:

> O botão [Provisionar] está desativado até ser marcado

3. Faturas

Os termos do serviço podem ser incluídos nas faturas como notas de rodapé para clareza.

Melhores Práticas:

- **Recursos:** Mantenha concisos (menos de 50 caracteres cada), concentre-se nos principais benefícios
- **Termos:** Inclua requisitos legais críticos, políticas de expiração, políticas de uso justo
- **Ambos:** Atualize quando o produto mudar para manter os clientes informados

Passo 8: Vincular Playbook de Provisão do Ansible

```
{
  "provisioning_play": "play_local_mobile_sim",
  "provisioning_json_vars": "{
    \"days\": 30,
    \"data_gb\": 20,
    \"voice_minutes\": \"unlimited\",
    \"sms_count\": \"unlimited\"
  }"
}
```

- `provisioning_play` - Nome do playbook do Ansible (sem extensão .yaml)
- `provisioning_json_vars` - Variáveis padrão passadas para o playbook
- O playbook deve existir em: `OmniCRM-API/Provisioners/plays/play_local_mobile_sim.yaml`

Definição Completa do Produto

```
{
  "product_name": "Prepaid Mobile 20GB",
  "product_slug": "prepaid-mobile-20gb",
  "category": "standalone",
  "service_type": "mobile",
  "enabled": true,
  "icon": "fa-solid fa-sim-card",
  "comment": "Prepaid mobile SIM with 20GB data, unlimited calls & texts",

  "retail_cost": 15.00,
  "wholesale_cost": 5.00,
  "retail_setup_cost": 0.00,
  "wholesale_setup_cost": 1.00,
  "contract_days": 30,

  "residential": true,
  "business": false,
  "customer_can_purchase": true,
  "available_from": "2025-01-01T00:00:00Z",
  "available_until": null,

  "auto_renew": "prompt",
  "allow_auto_renew": true,

  "inventory_items_list": "['SIM Card', 'Mobile Number']",

  "features_list": "[
    '20GB High-Speed Data',
    'Unlimited Calls & Texts',
    'EU Roaming Included',
    'No Contract',
    '30-Day Expiry'
  ]",
  "terms": "Credit expires after 30 days. Data, calls, and texts valid only within expiry period. Fair use policy applies.",

  "provisioning_play": "play_local_mobile_sim",
  "provisioning_json_vars": "{
    \"days\": 30,
    \"data_gb\": 20,
    \"voice_minutes\": \"unlimited\",
    \"sms_count\": \"unlimited\"
```

```
  }"
}
```

Criando um Produto de Complemento

Os complementos melhoram ou modificam serviços existentes. Eles vêm em dois tipos: **complementos virtuais** (sem recursos físicos) e **complementos de hardware** (requerem inventário).

Exemplo 1: Complemento Virtual (5GB Data Boost)

Um complemento digital que adiciona dados a um serviço móvel existente:

```
{
  "product_name": "5GB Data Boost",
  "product_slug": "5gb-data-boost",
  "category": "addon",
  "service_type": "mobile",
  "enabled": true,
  "icon": "fa-solid fa-plus",
  "comment": "Add 5GB extra data to existing mobile service",

  "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": "5GB High-Speed Data. Valid for 7 Days",
  "terms": "Data expires after 7 days or when exhausted. Cannot be
refunded.",

  "provisioning_play": "play_topup_charge_then_action",
  "provisioning_json_vars": "{
    \"data_gb\": 5,
    \"days\": 7
  }"
}
```

Exemplo 2: Complemento de Hardware (Aluguel de Modem)

Um complemento que fornece equipamento físico para um serviço de fibra existente:

```
{
  "product_name": "WiFi 6 Modem Rental",
  "product_slug": "wifi6-modem-rental",
  "category": "addon",
  "service_type": "internet",
  "enabled": true,
  "icon": "fa-solid fa-router",
  "comment": "Add WiFi 6 modem to fiber service - rental",

  "retail_cost": 10.00,
  "wholesale_cost": 3.00,
  "retail_setup_cost": 0.00,
  "wholesale_setup_cost": 45.00,
  "contract_days": 30,

  "residential": true,
  "business": true,
  "customer_can_purchase": true,

  "auto_renew": "true",
  "allow_auto_renew": true,

  "inventory_items_list": "['Rental Modem']",
  "relies_on_list": "",

  "features_list": "WiFi 6 (802.11ax). Dual-band 2.4GHz + 5GHz. Up to 40 devices. Parental controls",
  "terms": "Equipment rental. Must be returned on service cancellation or £150 replacement fee applies. Equipment remains property of provider.",

  "provisioning_play": "play_addon_assign_modem",
  "provisioning_json_vars": "{
    \"device_type\": \"modem_router\",
    \"requires_configuration\": true
  }"
}
```


Principais Diferenças para Complementos:

- `category: "addon"` - Aplicado a um serviço existente, não autônomo

- `contract_days: 0` (virtual) ou `30` (aluguel recorrente) - Frequência de cobrança
- `inventory_items_list: "[]"` (virtual) ou `"['Rental Modem']"` (hardware) - Recursos físicos
- `auto_renew: "false"` (uma vez) ou `"true"` (aluguel) - Comportamento recorrente
- `relies_on_list: ""` - Vazio significa que se aplica a qualquer serviço do `service_type` correspondente

Por que Complementos de Hardware Precisam de Inventário:

Complementos de hardware requerem `inventory_items_list` porque:

1. **Rastrear Equipamentos** - Saber qual modem está com qual cliente
2. **Prevenir Esgotamentos** - Não é possível provisionar complemento se não houver modems em estoque
3. **Recuperação** - Quando o cliente cancela, saber qual equipamento recuperar
4. **Rastreio de Custos** - Vincular custo de atacado a um número de série específico
5. **Depreciação** - Rastrear o valor do equipamento ao longo do período de aluguel
6. **Garantia** - Identificar unidades defeituosas pelo número de série

Fluxo de Provisão de Complementos com Inventário:

Quando um cliente adiciona "Aluguel de Modem WiFi 6" ao seu serviço de fibra:

1. **Complemento Selecionado** - O cliente clica em "Adicionar ao Serviço"
2. **Seletor de Inventário Aparece** - Assim como serviços autônomos:
3. **Pagamento Processado** - £10.00 de aluguel mensal cobrado
4. **Modem Atribuído** - Inventário atualizado:
   - `service_id`: Vinculado ao serviço de fibra
   - `customer_id`: Vinculado ao cliente
   - `item_state`: "Atribuído"
5. **Envio Acionado** - Sistema de atendimento notificado para enviar o modem

6. **Instalação** - O cliente recebe o modem, conecta ao ONT
7. **Cobrança Recorrente** - £10/mês cobrado até o complemento ser cancelado

Desprovisionamento de Complementos de Hardware:

Quando o cliente cancela o aluguel do modem:

1. **Cancelamento Iniciado** - O cliente clica em "Remover Complemento"
2. **Processo de Devolução Iniciado**:
   - E-mail enviado com instruções de devolução
   - Etiqueta de envio pré-paga gerada
   - Período de carência de 14 dias antes da penalidade
3. **Equipamento Devolvido**:
   - Inventário atualizado: `item_state` = "Em Estoque" (após reforma)
   - Ou `item_state` = "Danificado" (se defeituoso)
   - Vinculado ao próximo cliente após reforma
4. **Sem Devolução**:
   - Após 14 dias, taxa de substituição de £150 cobrada
   - Inventário marcado: `item_state` = "Perdido"
   - Custo de atacado (£45) + valor de substituição recuperado

Preços para Complementos:

Os complementos podem ser precificados de forma diferente dos serviços autônomos:

- Complementos virtuais geralmente não têm custos de configuração
- Complementos de hardware podem ter custos de configuração de atacado para equipamentos
- Complementos de aluguel recorrente usam `contract_days` para frequência de cobrança

Etapa 2: O Processo de Provisão

Quando um cliente solicita o produto "Prepaid Mobile 20GB", o OmniCRM orquestra a provisão através do Ansible.

Diagrama de Fluxo de Provisão

Cliente Faz Pedido → Seleção de Inventário → Job de Provisão Criado ↓ ↓ Pagamento Autorizado ← Variáveis Montadas ← Playbook do Ansible Executado ↓ ↓ Registro de Serviço Criado → Configuração de Conta OCS → Inventário Atribuído → Serviço Ativo

Fluxo de Provisão Passo a Passo

1. Cliente Inicia Pedido

Na página do cliente:

- A equipe clica em "Adicionar Serviço"
- Seleciona "Prepaid Mobile 20GB" no carrossel de produtos
- Detalhes do produto e preços exibidos

2. Seleção de Inventário

O sistema solicita o inventário necessário:

- **SIM Card** - Menu suspenso mostra os SIMs disponíveis em estoque
  - Exemplo: "SIM-00123 - ICCID: 8944..."
- **Mobile Number** - Menu suspenso mostra os números de telefone disponíveis
  - Exemplo: "+44 7700 900123"

A equipe ou o cliente seleciona itens do inventário disponível.

3. Confirmação de Preços

O sistema exibe o preço final:

- Custo de configuração: £0.00 (ativação gratuita)
- Custo mensal: £15.00
- Devido hoje: £15.00 (primeiro mês)
- Data de renovação: 30 dias a partir de hoje

Se a solicitação de renovação automática estiver habilitada, o cliente escolhe:

☐ Renovar automaticamente este serviço a cada 30 dias

4. Botão de Provisão Clicado

Quando "Provisionar" é clicado, a API:

- Cria um registro de `Provision` com status "Running" (status=1)
- Mescla variáveis do produto + solicitação + seleções de inventário
- Gera uma thread em segundo plano para executar o playbook do Ansible
- Retorna `provision_id` para a interface para rastreamento de status

5. Variáveis Montadas

O sistema mescla variáveis de várias fontes:

Do Produto:

```
{
  "days": 30,
  "data_gb": 20,
  "voice_minutes": "unlimited",
  "sms_count": "unlimited"
}
```

Da Solicitação:

```
{
  "product_id": 42,
  "customer_id": 123,
  "SIM Card": 5001,
  "Mobile Number": 5002
}
```

Adicionadas pelo Sistema:

```
{
  "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "initiating_user": 7
}
```

Variáveis Finais Passadas para Ansible:

```
{
    "product_id": 42,
    "customer_id": 123,
    "SIM Card": 5001,
    "Mobile Number": 5002,
    "days": 30,
    "data_gb": 20,
    "voice_minutes": "unlimited",
    "sms_count": "unlimited",
    "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
    "initiating_user": 7
}
```

6. Execução do Playbook do Ansible

O playbook `play_local_mobile_sim.yaml` é executado com essas variáveis.

Compreendendo o Playbook de Provisão do Ansible

Vamos examinar um playbook de provisão real para entender o que acontece nos bastidores.

Exemplo de Playbook de Provisão de SIM Móvel

Localização: `OmniCRM-API/Provisioners/plays/play_local_mobile_sim.yaml`

Estrutura de Alto Nível:

```yaml
- name: Mobile SIM Provisioning
  hosts: localhost
  gather_facts: no
  become: False

  tasks:
    - name: Main block
      block:
        # 1. Load configuration
        # 2. Fetch product details from API
        # 3. Fetch customer details from API
        # 4. Fetch inventory details from API
        # 5. Create account in OCS (CGRateS)
        # 6. Add balances and allowances to OCS
        # 7. Create service record in CRM
        # 8. Assign inventory to service
        # 9. Record transactions
        # 10. Send welcome notifications

      rescue:
        # Rollback on failure
        # - Remove OCS account
        # - Release inventory
        # - Log error
```

Passo a Passo Detalhado do Playbook:

Tarefa 1: Carregar Configuração

```
- name: Include vars of crm_config
  ansible.builtin.include_vars:
    file: "../../crm_config.yaml"
    name: crm_config
```

Carrega a configuração do sistema, incluindo:

- URL e credenciais do OCS/CGRateS
- URL base do CRM
- Configuração do inquilino

Tarefa 2: Buscar Detalhes do Produto

```
- 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
```

O que isso faz:

- Chama `GET /crm/product/product_id/42`
- Recupera a definição completa do produto
- Armazena na variável `api_response_product`

Por que: Mesmo tendo `provisioning_json_vars` do produto, buscamos o produto completo para obter:

- Preços mais recentes (podem ter mudado desde o início do pedido)
- Nome do produto para nomeação do serviço
- Lista de recursos para documentação
- Custos de atacado para rastreamento de margem

Tarefa 3: Definir Fatos do Pacote

```
- name: Set package facts
  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
}}"
```

Extrai valores comumente usados em variáveis simples para legibilidade.

Tarefa 4: Buscar Detalhes do Inventário

```
- name: Get SIM information from CRM API
  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: Set IMSI from Inventory response
  set_fact:
    imsi: "{{ api_response_sim.json.itemtext2 }}"
    iccid: "{{ api_response_sim.json.itemtext1 }}"
```

O que isso faz:

- Procura o ID do inventário do SIM Card 5001
- Recupera detalhes do SIM:
  - `itemtext1` = ICCID (número do cartão SIM)
  - `itemtext2` = IMSI (identidade do assinante)
- Faz o mesmo para o inventário do Mobile Number (recupera o número de telefone)

Por que isso importa:

- IMSI é necessário para provisionar o assinante no HSS (Home Subscriber Server)
- ICCID é registrado nas notas do serviço para solução de problemas
- O número de telefone (MSISDN) é exibido para o cliente e usado para roteamento

Tarefa 5: Gerar UUID do Serviço

```
- name: Generate UUID Fact
  set_fact:
    uuid: "{{ 99999999 | random | to_uuid }}"

- name: Set Service UUID
  set_fact:
    service_uuid: "Local_Mobile_SIM_{{ uuid[0:8] }}"
```

O que isso faz:

- Gera um UUID aleatório
- Cria um service_uuid como `Local_Mobile_SIM_a3f2c1d8`

Por que:

- O UUID do serviço é o identificador único no OCS/CGRateS
- Usado para todas as operações de cobrança
- Deve ser globalmente único entre todos os serviços

Tarefa 6: Criar Conta no OCS

```
- name: Create account in 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
```

O que isso faz:

- Chama a API JSON-RPC do CGRateS
- Cria uma nova conta com service_uuid
- Define a conta como ativa (não desativada)
- Previne saldo negativo (modo pré-pago)

Por que:

- A conta OCS é onde toda a cobrança acontece
- Saldo (dados, voz, SMS, dinheiro) é armazenado aqui
- O uso é rastreado e avaliado em tempo real

Tarefa 7: Adicionar Saldo de Dados

```
- name: Add 20GB Data Balance
  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"
          }
        }]
      }
```

O que isso faz:

- Adiciona 20GB de saldo de dados à conta
- Valor: 21474836480 bytes (20 * 1024 * 1024 * 1024)
- Expira em 720 horas (30 dias)
- Peso 10 (peso mais alto consumido primeiro)

Tarefa 8: Adicionar Voz & SMS Ilimitados

```
- name: Add Unlimited Voice
  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"
          }
        }]
      }
```

- Adiciona 999.999.999 segundos de voz (essencialmente ilimitado)
- Expira em 30 dias

Tarefa 9: Criar Registro de Serviço no CRM

```
- name: Add Service via 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
```

O que isso cria:

- Registro de serviço vinculado ao cliente
- Vincula ao OCS através do `service_uuid`
- Armazena custos de varejo e atacado
- Define o status como "Ativo"
- Retorna `service_id` para operações subsequentes

Tarefa 10: Atribuir Inventário ao Serviço

```
- name: Assign SIM Card to Service
  uri:
    url: "{{ crm_config.crm.base_url
}}/crm/inventory/inventory_id/{{ hostvars[inventory_hostname]['SIM
Card'] }}"
    method: PATCH
    body_format: json
    headers:
      Authorization: "Bearer {{ access_token }}"
    body:
      {
        "service_id": "{{
service_creation_response.json.service_id }}",
        "customer_id": "{{ customer_id }}",
        "item_state": "Assigned"
      }
```

O que isso faz:

- Atualiza o registro do inventário do SIM Card
- Define `service_id` para vincular o SIM ao serviço
- Muda o estado de "Em Estoque" para "Atribuído"
- Repete para o inventário do Mobile Number

Por que:

- Rastreia qual SIM está atribuído a qual cliente
- Previne a dupla alocação de inventário
- Habilita relatórios e auditoria de inventário

Tarefa 11: Registrar Transação de Custo de Configuração

```
- name: Add Setup Cost Transaction
  uri:
    url: "{{ crm_config.crm.base_url }}/crm/transaction/"
    method: PUT
    body_format: json
    headers:
      Authorization: "Bearer {{ access_token }}"
    body:
      {
        "customer_id": "{{ customer_id }}",
        "service_id": "{{
service_creation_response.json.service_id }}",
        "title": "{{ package_name }} - Setup",
        "description": "Activation fee",
        "retail_cost": "{{ setup_cost }}",
        "wholesale_cost": "{{
api_response_product.json.wholesale_setup_cost }}"
      }
```

O que isso faz:

- Registra uma cobrança de £0.00 de configuração ao cliente (varejo)
- Registra um custo de atacado de £1.00
- Cria um registro de transação para faturamento

Tarefa 12: Bloco de Resgate (Tratamento de Erros)

```
rescue:
  - name: Remove account in OCS on failure
    uri:
      url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
      method: POST
      body:
        {
          "method": "ApierV2.RemoveAccount",
          "params": [{
            "Account": "{{ service_uuid }}"
          }]
        }

  - name: Fail the provisioning
    fail:
      msg: "Provisioning failed, rolled back OCS account"
```

O que isso faz:

- Se qualquer tarefa falhar, o bloco de resgate é executado
- Exclui a conta OCS que foi parcialmente criada
- Libera o inventário de volta para "Em Estoque"
- Falha o trabalho de provisão com mensagem de erro

Por que:

- Previne contas órfãs no OCS
- Garante rollback limpo em erros
- Mantém a consistência dos dados

Provisão Completa: O que foi Criado

Após a provisão bem-sucedida, o sistema tem:

1. Conta OCS (CGRateS):

- ID da Conta: `Local_Mobile_SIM_a3f2c1d8`
- Saldos:
  - 20GB de dados (expira em 30 dias)

    - Voz ilimitada (999M segundos, expira em 30 dias)
    - SMS ilimitados (999M mensagens, expira em 30 dias)

2. Registro de Serviço no CRM:

- ID do Serviço: 1234
- Cliente: John Doe (customer_id: 123)
- Produto: Prepaid Mobile 20GB (product_id: 42)
- Nome do Serviço: "Mobile - +44 7700 900123"
- UUID do Serviço: `Local_Mobile_SIM_a3f2c1d8`
- Status: Ativo
- Custo Mensal: £15.00 (varejo), £5.00 (atacado)
- Lucro: £10.00/mês

3. Atribuições de Inventário:

- SIM Card 5001: Atribuído ao serviço 1234, cliente 123
- Mobile Number 5002: Atribuído ao serviço 1234, cliente 123

4. Registros de Transação:

- Transação de custo de configuração criada
- Cobrança do primeiro mês registrada

5. O Cliente Agora Pode:

- Ver o serviço no portal de autoatendimento
- Ver saldo de 20GB de dados
- Fazer chamadas e enviar SMS
- Recarregar ou adicionar complementos
- Ver uso em tempo real

Etapa 3: Adicionando Complementos e Recargas

Após um serviço estar ativo, os clientes podem comprar complementos para aprimorar seu serviço.

Fluxo de Provisão de Complementos

Vamos supor que o cliente usou 18GB de sua cota de 20GB e deseja comprar o complemento "5GB Data Boost".

1. Cliente Navega até o Serviço

- Abre a página do serviço "Mobile - +44 7700 900123"
- Vê o uso atual: 18GB de 20GB usados (90%)
- Clica em "Adicionar Complemento" ou "Recarregar"

2. Sistema Filtra Complementos Disponíveis

Mostra apenas complementos onde:

- `category = "addon"`
- `service_type = "mobile"` (corresponde ao tipo de serviço)
- `residential = true` (se o cliente for residencial)
- `enabled = true`

O cliente vê: "5GB Data Boost - £5.00"

3. Cliente Seleciona o Complemento

- Clica em "5GB Data Boost"
- Confirma a compra por £5.00
- O sistema captura a autorização de pagamento

4. Provisão do Complemento Iniciada

O sistema chama `play_topup_charge_then_action.yaml` com variáveis:

```
{
    "product_id": 43,  # Produto 5GB Data Boost
    "customer_id": 123,
    "service_id": 1234,  # Serviço existente
    "access_token": "eyJ...",
    "data_gb": 5,  # Das variáveis de provisão
    "days": 7      # Das variáveis de provisão
}
```

Diferença Chave em Relação ao Autônomo:

- `service_id` está incluído (serviço existente a ser modificado)
- Nenhum inventário necessário
- Nenhuma criação de serviço (modifica o existente)

Análise do Playbook de Provisão de Complementos

Tarefa 1: Buscar Detalhes do Serviço

```
- 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 }}"
  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 }}"
```

Por que:

- Necessário `service_uuid` para adicionar saldo à conta OCS correta
- Verifica se o serviço existe e está ativo

- Garante que o serviço pertence ao cliente

Tarefa 2: Cobrar o Cliente

```yaml
- name: Get Customer's Default Payment Method
  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: Get default payment method ID
  set_fact:
    payment_method_id: "{{
api_response_payment_method.json.data.payment_method_id }}"

- name: Charge customer
  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": "5GB Data Boost",
          "service_id": "{{ service_id | int }}",
          "product_id": "{{ product_id | int }}",
          "invoice": true
        }
      }
  register: charge_response

- name: Assert payment successful
  assert:
    that:
      - charge_response.json.success == true
```

O que isso faz:

- Encontra o método de pagamento padrão do cliente no Stripe
- Cobra £5.00 do cartão
- Registra custo de atacado £1.50 para rastreamento de margem
- Cria transação vinculada ao serviço
- Adiciona à próxima fatura
- Falha a provisão se o pagamento falhar

Por que cobrar primeiro:

- Nenhum crédito entregue até o pagamento ser confirmado
- Previne fraudes
- Combina pagamento com provisão de complemento

Tarefa 3: Adicionar Saldo de Dados à Conta OCS

```
- name: Add 5GB Data Balance
  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
          }
        }]
      }
```

O que isso faz:

- Adiciona 5GB (5368709120 bytes) à conta
- Expira em 168 horas (7 dias)

- Peso 20 (peso mais alto consumido primeiro - impulso antes da cota mensal)

Saldo do Cliente Após o Complemento:

- Mensal original: 2GB restantes (expira em 25 dias)
- Novo impulso: 5GB (expira em 7 dias)
- Total disponível: 7GB
- Ordem de uso: impulso consumido primeiro, depois mensal

Tarefa 4: Registrar Transação

```
- name: Add Addon Transaction
  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": "5GB Data Boost",
        "description": "Additional 5GB data valid for 7 days",
        "retail_cost": 5.00,
        "wholesale_cost": 1.50
      }
```

O que isso faz:

- Registra uma cobrança de £5.00 ao cliente
- Registra um custo de atacado de £1.50
- Vincula a transação ao serviço para relatórios

Resumo Completo do Fluxo de Complemento

1. O cliente seleciona o complemento da lista filtrada
2. Pagamento autorizado e cobrado

3. Saldo de dados adicionado à conta OCS
4. Transação registrada no CRM
5. O cliente imediatamente vê o saldo atualizado: 7GB disponíveis

Rastreamento Financeiro:

- Cobrança mensal do serviço: £15 varejo, £5 atacado
- Compra do complemento: £5 varejo, £1.50 atacado

Renovação Automática: Complementos Recorrentes

Alguns complementos podem ser configurados para renovação automática (planos de dados mensais, assinaturas, etc).

Configuração do Produto:

```
{
  "product_name": "Monthly 10GB Data Plan",
  "category": "addon",
  "retail_cost": 10.00,
  "contract_days": 30,
  "auto_renew": "true",
  "provisioning_play": "play_topup_charge_then_action"
}
```

Provisão Cria ActionPlan:

```
- name: Create ActionPlan for Auto-Renewal
  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
        }]
      }
```

O que isso faz:

- Cria uma tarefa agendada no OCS
- Executa `Action_{{ product_slug }}` a cada 30 dias
- A ação cobra o cliente e reaplica o saldo de dados
- Continua até que o cliente cancele

Gestão do Cliente:

- O cliente vê "Próxima Renovação: 1º de fevereiro de 2025 - £10.00" na visualização do serviço
- Pode clicar em "Cancelar Renovação Automática" para interromper cobranças futuras
- Pode clicar em "Renovar Agora" para aplicar imediatamente a cota do próximo mês

Etapa 4: Desprovisionamento de Serviços

Quando um cliente cancela o serviço, o sistema deve remover todos os recursos de forma limpa.

Gatilhos de Desprovisionamento

O desprovisionamento pode ser acionado por:

1. **Cancelamento do Cliente** - O cliente clica em "Cancelar Serviço"
2. **Ação Administrativa** - A equipe marca o serviço para desativação
3. **Não Pagamento** - O serviço expira devido à falta de renovação
4. **Fim do Contrato** - Contrato de prazo fixo chega à data de término

Fluxo de Desprovisionamento

1. Cliente Inicia Cancelamento

- Navega até o serviço
- Clica em "Cancelar Serviço"
- O sistema solicita: "Você tem certeza? Qualquer saldo restante será perdido."
- O cliente confirma

2. Período de Carência (Opcional)

Alguns operadores implementam um período de carência:

- Serviço marcado como "Cancelamento Pendente"
- Permanece ativo por 7-30 dias
- O cliente pode reverter o cancelamento durante o período de carência
- Desprovisionamento automático após o período de carência

3. Job de Desprovisionamento Criado

O sistema cria um job de provisão com:

```
{
    "action": "deprovision",
    "service_id": 1234,
    "customer_id": 123,
    "service_uuid": "Local_Mobile_SIM_a3f2c1d8"
}
```

Chama o playbook especificado em `service.deprovisioning_play` ou bloco de resgate do playbook original.

4. Playbook de Desprovisionamento do Ansible

```
- name: Deprovision Mobile Service
  hosts: localhost
  tasks:
    - name: Disable OCS Account
      uri:
        url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
        method: POST
        body:
          {
            "method": "ApierV2.SetAccount",
            "params": [{
              "Account": "{{ service_uuid }}",
              "ExtraOptions": { "Disabled": true }
            }]
          }

    - name: Remove ActionPlans (stop auto-renewals)
      uri:
        url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
        method: POST
        body:
          {
            "method": "ApierV1.RemoveActionPlan",
            "params": [{
              "Id": "ServiceID_{{ service_uuid }}__*"
            }]
          }

    - name: Update Service Status in CRM
      uri:
        url: "http://localhost:5000/crm/service/{{ service_id }}"
        method: PATCH
        body:
          {
            "service_status": "Deactivated",
            "service_deactivate_date": "{{ current_datetime }}"
          }

    - name: Release Inventory to Stock
      uri:
        url: "http://localhost:5000/crm/inventory/inventory_id/{{
sim_card_id }}"
        method: PATCH
```

```
        body:
          {
            "service_id": null,
            "customer_id": null,
            "item_state": "Decommissioned"
          }
```

O que isso faz:

1. **Desativa a conta OCS** - Para todas as cobranças, uso bloqueado
2. **Remove ActionPlans** - Cancela renovações automáticas
3. **Atualiza o serviço no CRM** - Status "Desativado", data registrada
4. **Libera o inventário** - SIM marcado como "Descomissionado", disponível para reutilização (após reforma)

5. Pós-Desprovisionamento

O sistema realiza a limpeza:

- O cliente não vê mais o serviço no portal de autoatendimento
- O serviço permanece no CRM para relatórios históricos
- Transações e faturas preservadas para contabilidade
- O inventário pode ser reformado e reutilizado
- A conta OCS pode ser arquivada após o período de retenção

Desprovisionamento Parcial vs Total

Desprovisionamento Parcial (Suspensão):

- Usado para não pagamento ou suspensão temporária
- Conta OCS desativada, mas não excluída
- Saldos preservados
- Pode ser reativada quando o pagamento for recebido

```
- name: Suspend Service
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body:
      {
        "method": "ApierV2.SetAccount",
        "params": [{
          "Account": "{{ service_uuid }}",
          "ExtraOptions": { "Disabled": true }
        }]
      }
```

Desprovisionamento Total (Cancelamento Permanente):

- Usado para cancelamento permanente
- Conta OCS excluída totalmente
- Saldos perdidos
- Não pode ser reativada

```
- name: Remove OCS Account
  uri:
    url: "http://{{ crm_config.ocs.cgrates }}/jsonrpc"
    method: POST
    body:
      {
        "method": "ApierV2.RemoveAccount",
        "params": [{
          "Account": "{{ service_uuid }}"
        }]
      }
```

Melhores Práticas para Gestão de Produtos

Gestão do Ciclo de Vida do Produto

Estados do Produto:

- `enabled: true` - Produto disponível para novos pedidos
- `enabled: false` - Produto desativado, serviços existentes continuam

Desativando Produtos:

- Marque o produto como `enabled: false` para evitar novos pedidos
- Serviços existentes permanecem ativos
- Clientes ainda podem renovar/modificar serviços existentes
- Útil para descontinuar produtos antigos

Gestão de Inventário

Estados do Inventário:

- `New` - Estoque novo, pronto para atribuição
- `In Stock` - Disponível para provisão
- `Assigned` - Vinculado ao serviço do cliente
- `Decommissioned` - Pode ser reformado e reutilizado
- `Damaged` - Necessita reparo ou descarte

Reutilizando Inventário:

Após desprovisionamento:

- Cartões SIM: Reformar e marcar como "Em Estoque"
- Números de telefone: Liberar após o período de portabilidade (30 dias)
- Equipamentos: Testar, reformar, marcar como "Usado"

Métricas de Provisão

Monitorar:

- Taxa de sucesso de provisão
- Tempo médio de provisão
- Pontos de falha comuns
- Rotatividade de inventário

Integração do Mailjet com o OmniCRM

O OmniCRM integra-se com **Mailjet** para gerenciar toda a comunicação por e-mail com clientes e funcionários, garantindo entrega de e-mail profissional, com marca e confiável para e-mails transacionais e campanhas de marketing.

Visão Geral

A integração do Mailjet fornece:

- **E-mails Transacionais Automatizados** - Redefinições de senha, faturas, e-mails de boas-vindas, notificações
- **Sincronização de Contatos** - Contatos de clientes sincronizados automaticamente com o Mailjet para campanhas
- **Modelos de E-mail** - Mais de 10 tipos de e-mail pré-configurados com modelos personalizáveis do Mailjet
- **Campanhas de Marketing** - Campanhas de e-mail segmentadas com base em dados de clientes
- **Entrega Confiável** - Infraestrutura de e-mail profissional com rastreamento e análises

Configuração

O Mailjet é configurado em `OmniCRM-API/crm_config.yaml` na seção `mailjet`.

Configuração Básica

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

Obtendo Credenciais da API:

1. Crie uma conta em <https://www.mailjet.com>
2. Navegue até **Configurações da Conta → Chaves da API**
3. Copie **Chave da API** e **Chave Secreta**
4. Cole em `crm_config.yaml`

Configuração do Modelo de E-mail

O OmniCRM usa **10 tipos distintos de modelos de e-mail** para comunicações automatizadas. Cada modelo é configurado com:

- **from_email** - Endereço de e-mail do remetente
- **from_name** - Nome de exibição do remetente
- **template_id** - ID do modelo Mailjet (numérico)
- **subject** - Linha de assunto do e-mail

Tipos de Modelo e Configuração

E-mail de Boas-Vindas ao Cliente

Enviado quando uma nova conta de cliente é criada.

```
api_crmCommunicationCustomerWelcome:
  from_email: "support@yourcompany.com"
  from_name: "Suporte da Sua Empresa"
  template_id: 5977509
  subject: "Bem-vindo à Sua Empresa"
```

Quando Enviado:

- Novo cliente se inscreve pelo portal de Autoatendimento
- Funcionário cria nova conta de cliente
- Cliente ativa o serviço pela primeira vez

Variáveis do Modelo Disponíveis:

- `{{ var:customer_name }}` - Nome completo do cliente
- `{{ var:email }}` - Endereço de e-mail do cliente
- `{{ var:company_name }}` - Nome da sua empresa
- `{{ var:login_url }}` - Link para o portal de Autoatendimento
- `{{ var:support_url }}` - Link para a página de suporte

E-mail de Fatura do Cliente

Enviado quando uma fatura é gerada e está pronta para pagamento.

```
api_crmCommunicationCustomerInvoice:
  from_email: "billing@yourcompany.com"
  from_name: "Cobrança da Sua Empresa"
  template_id: 6759851
  subject: "Sua Fatura - "
```

Quando Enviado:

- Fatura gerada automaticamente para o período de cobrança
- Fatura manual criada pelo funcionário
- Cliente solicita cópia da fatura

Variáveis do Modelo Disponíveis:

- `{{ var:customer_name }}` - Nome completo do cliente
- `{{ var:invoice_number }}` - ID/número da fatura
- `{{ var:invoice_date }}` - Data de emissão da fatura
- `{{ var:due_date }}` - Data de vencimento do pagamento
- `{{ var:total_amount }}` - Valor total devido
- `{{ var:invoice_url }}` - Link para visualizar/download do PDF da fatura
- `{{ var:pay_url }}` - Link para pagar a fatura online

Anexo da Fatura:

O PDF da fatura é anexado automaticamente ao e-mail.

Lembrete de Fatura do Cliente

Enviado para lembrar os clientes sobre faturas vencidas.

```
api_crmCommunicationCustomerInvoiceReminder:
  from_email: "billing@yourcompany.com"
  from_name: "Cobrança da Sua Empresa"
  template_id: 6759852
  subject: "Lembrete de Pagamento - Fatura Vencida"
```

Quando Enviado:

- Fatura está X dias atrasada (configurável)
- Lembrete manual acionado pelo funcionário
- Fluxo de lembrete automático (se configurado)

Variáveis do Modelo Disponíveis:

- `{{ var:customer_name }}`
- `{{ var:invoice_number }}`
- `{{ var:due_date }}`
- `{{ var:days_overdue }}`
- `{{ var:total_amount }}`
- `{{ var:pay_url }}`

E-mail de Boas-Vindas ao Usuário da Equipe

Enviado quando uma nova conta de usuário da equipe é criada.

```
api_crmCommunicationUserWelcome:
  from_email: "admin@yourcompany.com"
  from_name: "Admin da Sua Empresa"
  template_id: 5977510
  subject: "Bem-vindo à Equipe"
```

Quando Enviado:

- O administrador cria um novo usuário da equipe
- O botão "Enviar E-mail de Boas-Vindas" é clicado na gestão de usuários

Variáveis do Modelo Disponíveis:

- `{{ var:user_name }}` - Nome completo do usuário da equipe
- `{{ var:email }}` - E-mail do usuário da equipe
- `{{ var:role }}` - Função(s) atribuída(s)
- `{{ var:login_url }}` - Link para login no portal do administrador
- `{{ var:temp_password }}` - Senha temporária (se aplicável)
- `{{ var:support_email }}` - Contato de suporte de TI

Redefinição de Senha do Usuário

Enviado quando um usuário solicita redefinir sua senha.

```
api_crmCommunicationUserPasswordReset:
  from_email: "noreply@yourcompany.com"
  from_name: "Segurança da Sua Empresa"
  template_id: 5977511
  subject: "Solicitação de Redefinição de Senha"
```

Quando Enviado:

- O usuário clica em "Esqueceu a Senha" na página de login
- O usuário envia uma solicitação de redefinição de senha

Variáveis do Modelo Disponíveis:

- `{{ var:user_name }}`
- `{{ var:reset_url }}` - Link de redefinição de senha com tempo limitado (tipicamente 1 hora)
- `{{ var:expiry_time }}` - Quando o link de redefinição expira

Nota de Segurança:

Os links de redefinição expiram após o período de tempo configurado (padrão 1 hora).

Sucesso na Redefinição de Senha do Usuário

Enviado para confirmar que a senha foi alterada com sucesso.

```
api_crmCommunicationUserPasswordResetSuccess:
  from_email: "noreply@yourcompany.com"
  from_name: "Segurança da Sua Empresa"
  template_id: 5977512
  subject: "Senha Alterada com Sucesso"
```

Quando Enviado:

- O usuário completa com sucesso a redefinição da senha
- Imediatamente após a nova senha ser definida

Variáveis do Modelo Disponíveis:

- `{{ var:user_name }}`
- `{{ var:change_date }}` - Data/hora em que a senha foi alterada
- `{{ var:ip_address }}` - Endereço IP da alteração (opcional)
- `{{ var:support_email }}` - Contato se a alteração foi não autorizada

Alteração de Senha do Usuário

Enviado quando um usuário altera sua senha nas configurações.

```
api_crmCommunicationUserPasswordChange:
  from_email: "noreply@yourcompany.com"
  from_name: "Segurança da Sua Empresa"
  template_id: 5977513
  subject: "Notificação de Alteração de Senha"
```

Quando Enviado:

- O usuário altera a senha no perfil/configurações
- O administrador redefine a senha do usuário

Variáveis do Modelo Disponíveis:

- `{{ var:user_name }}`

- `{{ var:change_date }}`
- `{{ var:changed_by }}` - "Próprio" ou nome do administrador
- `{{ var:support_email }}`

Verificação de E-mail

Enviado para verificar o endereço de e-mail de um usuário.

```
api_crmCommunicationEmailVerification:
  from_email: "noreply@yourcompany.com"
  from_name: "Sua Empresa"
  template_id: 5977514
  subject: "Verifique Seu Endereço de E-mail"
```

Quando Enviado:

- Nova conta criada (cliente ou funcionário)
- O usuário altera o endereço de e-mail
- Verificação de e-mail necessária por motivos de segurança

Variáveis do Modelo Disponíveis:

- `{{ var:user_name }}`
- `{{ var:verification_url }}` - Link para verificar o e-mail
- `{{ var:verification_code }}` - Código a ser inserido manualmente (alternativa ao link)

Notificação de Saldo Expirado

Enviado quando o saldo/limite de serviço de um cliente expirou.

```
api_crmCommunicationsBalanceExpired:
  from_email: "support@yourcompany.com"
  from_name: "Suporte da Sua Empresa"
  template_id: 5977515
  subject: "Seu Saldo de Serviço Expirou"
```

Quando Enviado:

- Saldo pré-pago expira
- Data de renovação do limite mensal passou
- Data de expiração do serviço alcançada

Variáveis do Modelo Disponíveis:

- `{{ var:customer_name }}`
- `{{ var:service_name }}` - Nome do serviço expirado
- `{{ var:expiry_date }}`
- `{{ var:balance_type }}` - "Dados", "Voz", "Monetário", etc.
- `{{ var:renewal_url }}` - Link para renovar/recarregar

Alerta de Baixo Saldo

Enviado quando o saldo de um cliente cai abaixo do limite configurado.

```
api_crmCommunicationsBalanceLow:
  from_email: "support@yourcompany.com"
  from_name: "Suporte da Sua Empresa"
  template_id: 5977516
  subject: "Alerta de Baixo Saldo"
```

Quando Enviado:

- Saldo cai abaixo do limite (por exemplo, 20% restante)
- Configurado no plano de serviço ou OCS
- Monitoramento em tempo real aciona alerta

Variáveis do Modelo Disponíveis:

- `{{ var:customer_name }}`
- `{{ var:service_name }}`
- `{{ var:current_balance }}`
- `{{ var:threshold }}`
- `{{ var:balance_type }}`
- `{{ var:topup_url }}` - Link para adicionar saldo

Criando Modelos de E-mail do Mailjet

Para cada tipo de e-mail, você precisa criar um modelo correspondente no Mailjet.

Passo 1: Criar Modelo no Mailjet

1. Faça login no painel do Mailjet
2. Navegue até **Modelos de E-mail → Modelos Transacionais**
3. Clique em **Criar um Novo Modelo**
4. Escolha **Codifique seu próprio modelo** (para usuários avançados) ou **Use o construtor de modelos**

Passo 2: Projetar o Modelo

Use o construtor de arrastar e soltar do Mailjet ou o editor HTML para projetar seu e-mail.

Elementos Essenciais:

- **Cabeçalho** - Logotipo e marca da empresa
- **Saudação** - Personalizada com `{{ var:customer_name }}` ou `{{ var:user_name }}`
- **Conteúdo** - Corpo principal da mensagem
- **Variáveis** - Insira variáveis do modelo da lista acima
- **Chamada à Ação** - Botões/links para ações do usuário
- **Rodapé** - Link para cancelar a inscrição, endereço da empresa, contato de suporte

Exemplo de Modelo (Redefinição de Senha):

```
<!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>Solicitação de Redefinição de Senha</h2>

    <p>Olá {{ var:user_name }},</p>

    <p>Recebemos uma solicitação para redefinir sua senha. Clique
no botão abaixo para criar uma nova senha:</p>

    <a href="{{ var:reset_url }}" class="button">Redefinir
Senha</a>

    <p>Este link expira em {{ var:expiry_time }}.</p>

    <p>Se você não solicitou isso, por favor, ignore este e-mail.
</p>

    <hr>
    <p style="font-size: 12px; color: #666;">
        Sua Empresa | support@yourcompany.com<br/>
        123 Business St, City, Country
    </p>
</body>
</html>
```

Passo 3: Obter ID do Modelo

1. Salve o modelo no Mailjet
2. Anote o **ID do Modelo** (numérico, por exemplo, `5977509`)
3. Copie este ID para `crm_config.yaml`

Passo 4: Testar o Modelo

1. No Mailjet, use o recurso **E-mail de Teste**
2. Forneça valores de exemplo para todas as variáveis
3. Envie um e-mail de teste para você mesmo
4. Verifique a formatação, links e branding

Passo 5: Configurar no OmniCRM

Adicione a configuração do modelo a `crm_config.yaml`:

```
mailjet:
  api_key: your_api_key
  api_secret: your_secret

  api_crmCommunicationUserPasswordReset:
    from_email: "noreply@yourcompany.com"
    from_name: "Segurança da Sua Empresa"
    template_id: 5977511
    subject: "Solicitação de Redefinição de Senha"
```

Reinicie a API do OmniCRM para que as alterações tenham efeito:

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

Sincronização de Contatos

Todos os contatos de clientes no OmniCRM são sincronizados automaticamente com o Mailjet.

O que é Sincronizado:

- Nome do contato
- Endereço de e-mail
- Tipo de contato (cobrança, técnico, etc.)

- Localização do cliente
- Campos personalizados

Frequência de Sincronização:

Os contatos são sincronizados em tempo real quando:

- Novo cliente criado
- Contato adicionado/atualizado
- Detalhes do cliente modificados

Solução de Problemas

E-mail não enviado

- **Causa:** Credenciais da API inválidas, conta do Mailjet suspensa ou ID do modelo incorreto
- **Solução:**
  - Verifique `api_key` e `api_secret` em `crm_config.yaml`
  - Verifique o status da conta do Mailjet e a cobrança
  - Verifique se o ID do modelo existe no Mailjet
  - Verifique os logs da API em busca de erros

Variáveis do modelo não substituindo

- **Causa:** Desconexão de nomes de variáveis ou dados ausentes no OmniCRM
- **Solução:**
  - Verifique se os nomes das variáveis correspondem exatamente (sensível a maiúsculas)
  - Use o formato `{{ var:variable_name }}`
  - Verifique se o OmniCRM está passando os dados das variáveis na chamada da API
  - Teste com dados de exemplo no Mailjet

PDF da fatura n��o anexado

- **Causa:** Falha na geração do PDF ou tamanho do arquivo muito grande
- **Solução:**
  - Verifique os logs de geração da fatura
  - Verifique se o modelo da fatura é renderizado corretamente
  - Certifique-se de que o PDF esteja abaixo de 15MB (limite do Mailjet)
  - Teste a geração do PDF da fatura separadamente

Contatos não sincronizados com o Mailjet

- **Causa:** Limite de taxa da API excedido ou serviço de sincronização não está em execução
- **Solução:**
  - Verifique os limites de taxa da API do Mailjet (200 chamadas/minuto)
  - Verifique se o serviço OmniCRM-API está em execução
  - Revise os logs de sincronização em busca de erros
  - Acione manualmente a sincronização para teste

Documentação Relacionada

- `administration_configuration` - Referência completa de configuração do Mailjet
- `payments_invoices` - Geração de faturas e entrega de e-mail
- `authentication_flows` - E-mails de redefinição e verificação de senha
- `customer_care` - E-mails de boas-vindas do portal de Autoatendimento
- Monitoramento & Métricas - Métricas da API do Mailjet, rastreamento de entrega de e-mail e monitoramento de desempenho

Leitura Adicional

- Documentação do Mailjet: <https://dev.mailjet.com/>
- Referência da API do Mailjet: <https://dev.mailjet.com/email/reference/>

Registro de Mudanças

Isso contém as últimas 50 alterações feitas na pilha de software OmniCRM ou em suas dependências.

Nota: Isso não rastreia alterações em configurações individuais de clientes.

Monitoramento & Métricas

A API OmniCRM fornece métricas abrangentes baseadas em Prometheus para monitorar o desempenho da aplicação, operações comerciais e integrações externas. Todas as métricas são expostas no endpoint `/crm/metrics` no formato de exposição do Prometheus.

Visão Geral

O sistema de métricas rastreia:

- **Operações de Provisionamento** - Execução de jobs, duração, taxas de sucesso/falha
- **Desempenho do Banco de Dados** - Tempos de consulta, saúde do pool de conexões
- **Integrações Externas** - Chamadas de API OCS/CGRateS, Stripe, Mailjet
- **Jobs em Segundo Plano** - Execução e desempenho de tarefas assíncronas
- **Requisições HTTP** - Uso de endpoints da API e tempos de resposta (auto-gerados)

Endpoint de Métricas

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

Formato: Formato de exposição do Prometheus

Autenticação: O endpoint de métricas é acessível publicamente para scraping pelo Prometheus. Em produção, é recomendado restringir o acesso usando regras de firewall ou autenticação de proxy reverso.

Categorias de Métricas

1. Métricas de Provisionamento

As métricas de provisionamento rastreiam a execução de playbooks do Ansible que provisionam serviços, gerenciam inventário e configuram sistemas externos. Veja Sistema de Provisionamento e Playbooks do Ansible para mais detalhes.

omnicrm_provision_jobs_total

Tipo: Contador

Rótulos:

- `status` - Status de conclusão do job: `success`, `failed`, `running`

Descrição: Número total de jobs de provisionamento criados. Incrementado quando os jobs de provisionamento são concluídos com seu status final.

omnicrm_provision_job_duration_seconds

Tipo: Histograma

Rótulos:

- `playbook` - Nome do playbook do Ansible executado
- `status` - Status de conclusão do job: `success`, `failed`

Buckets: `[1, 5, 10, 30, 60, 120, 180, 300, 600]` segundos (1s a 10min)

Descrição: Tempo necessário para a conclusão dos jobs de provisionamento. Registra a duração de toda a execução do playbook do início ao fim.

omnicrm_provision_jobs_active

Tipo: Gauge

Descrição: Número de jobs de provisionamento atualmente em execução. Incrementado quando um job começa, decrementado quando é concluído.

omnicrm_provision_tasks_total

Tipo: Contador

Rótulos:

- `playbook` - Nome do playbook do Ansible
- `status` - Resultado da tarefa: `ok`, `failed`, `ignored`

Descrição: Número total de tarefas do Ansible executadas dentro dos playbooks. Incrementado para cada tarefa individual que é concluída (sucesso ou falha).

omnicrm_provision_errors_total

Tipo: Contador

Rótulos:

- `error_type` - Tipo de erro: `fatal`, `task_failed`, `timeout`
- `playbook` - Nome do playbook do Ansible

Descrição: Número total de erros de provisionamento por tipo. Incrementado quando as tarefas de provisionamento falham ou ocorrem erros fatais durante a execução.

---

2. Métricas de Banco de Dados

As métricas de banco de dados monitoram o desempenho de consultas e a saúde do pool de conexões. O OmniCRM usa SQLAlchemy com instrumentação automática. Veja Arquitetura do Sistema para detalhes do modelo de dados.

omnicrm_db_query_duration_seconds

Tipo: Histograma

Rótulos:

- `operation` - Tipo de operação SQL: `SELECT`, `INSERT`, `UPDATE`, `DELETE`

- `table` - Nome da tabela do banco de dados

Buckets: `[0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1.0, 5.0, 10.0]` segundos

Descrição: Tempo de execução da consulta ao banco de dados. Rastreado automaticamente via ouvintes de eventos do SQLAlchemy em cada consulta.

omnicrm_db_pool_size

Tipo: Gauge

Descrição: Tamanho atual do pool de conexões do banco de dados. O número total de conexões no pool (tanto retiradas quanto disponíveis).

omnicrm_db_pool_overflow

Tipo: Gauge

Descrição: Transbordo atual do pool de conexões do banco de dados. O número de conexões criadas além do limite normal do tamanho do pool.

omnicrm_db_pool_connections_checked_out

Tipo: Gauge

Descrição: Número de conexões atualmente retiradas do pool e em uso pelo código da aplicação.

omnicrm_db_errors_total

Tipo: Contador

Rótulos:

- `error_type` - Tipo de erro do banco de dados
- `operation` - Operação que causou o erro: `connection_error`, `query_error`, etc.

Descrição: Número total de erros de banco de dados.

Status: Definido, mas não utilizado ativamente no código atual (reservado para uso futuro)

---

3. Métricas OCS/CGRateS

As métricas OCS (Sistema de Cobrança Online) rastreiam interações com o mecanismo de cobrança CGRateS. Veja Visão Geral de Cobrança e Integração CGRateS para mais detalhes.

omnicrm_ocs_api_calls_total

Tipo: Contador

Rótulos:

- `method` - Método da API OCS: `GetBalance`, `SetBalance`, `SetAccount`, `RemoveAccount`, etc.
- `status` - Resultado da chamada: `success`, `failed`

Descrição: Número total de chamadas de API OCS/CGRateS. Incrementado para cada chamada assíncrona à API do sistema OCS.

omnicrm_ocs_api_duration_seconds

Tipo: Histograma

Rótulos:

- `method` - Método da API OCS: `GetBalance`, `SetBalance`, `GetAccount`, etc.

Buckets: `[0.1, 0.25, 0.5, 1.0, 2.0, 5.0, 10.0, 30.0]` segundos

Descrição: Duração da chamada da API OCS/CGRateS. Registra o tempo necessário para cada chamada da API, incluindo latência de rede.

omnicrm_ocs_api_errors_total

Tipo: Contador

Rótulos:

- `method` - Método da API OCS que falhou
- `error_type` - Categoria de erro: `timeout`, `connection_error`, `json_error`, `http_error`, etc.

Descrição: Número total de erros da API OCS/CGRateS. Incrementado quando as chamadas da API OCS falham com tipos de erro específicos.

omnicrm_ocs_balance_queries_total

Tipo: Contador

Rótulos:

- `query_type` - Tipo de consulta de saldo: `single_account`, `multiple_accounts`

Descrição: Número total de consultas de saldo ao OCS. Usado para rastrear o uso de operações de consulta de saldo.

omnicrm_ocs_action_plan_operations_total

Tipo: Contador

Rótulos:

- `operation` - Operação do plano de ação: `create`, `remove`, `query`

Descrição: Número total de operações do plano de ação. Rastreia a criação, remoção e consulta de planos de ação do CGRateS para cobranças recorrentes.

---

4. Métricas de Pagamento Stripe

As métricas Stripe rastreiam operações de processamento de pagamentos. Veja Guia do Sistema de Pagamento e Métodos de Pagamento para detalhes de integração.

omnicrm_stripe_api_calls_total

Tipo: Contador

Rótulos:

- `operation` - Operação Stripe: `create_customer`, `charge`, `refund`, `update_payment_method`, etc.
- `status` - Resultado da operação: `success`, `failed`

Descrição: Número total de chamadas da API Stripe. Incrementado para cada operação de processamento de pagamento.

omnicrm_stripe_api_duration_seconds

Tipo: Histograma

Rótulos:

- `operation` - Tipo de operação Stripe

Buckets: `[0.1, 0.5, 1.0, 2.0, 5.0, 10.0, 30.0]` segundos

Descrição: Duração da chamada da API Stripe, incluindo latência de rede.

omnicrm_stripe_api_errors_total

Tipo: Contador

Rótulos:

- `operation` - Operação Stripe que falhou
- `error_type` - Categoria de erro: `card_declined`, `network_error`, `api_error`, etc.

Descrição: Número total de erros da API Stripe. Incrementado quando as operações de pagamento falham.

omnicrm_stripe_payment_amount_cents

Tipo: Contador

Rótulos:

- `payment_type` - Direção do pagamento: `charge`, `refund`

Descrição: Valor total do pagamento processado através do Stripe em centavos. Útil para rastrear o volume de transações e receita.

omnicrm_stripe_payment_failures_total

Tipo: Contador

Rótulos:

- `reason` - Motivo da falha: `card_declined`, `insufficient_funds`, `expired_card`, etc.

Descrição: Número total de falhas de pagamento do Stripe categorizadas por código de recusa.

---

5. Métricas de Email Mailjet

As métricas Mailjet rastreiam operações de entrega de email. Veja Integração Mailjet para detalhes de configuração.

omnicrm_mailjet_api_calls_total

Tipo: Contador

Rótulos:

- `email_type` - Tipo de template de email: `welcome`, `user_welcome`, `invoice`, `notification`
- `status` - Resultado da entrega: `success`, `failed`

Descrição: Número total de chamadas da API Mailjet. Rastreado via o decorador `@track_mailjet_call`.

omnicrm_mailjet_api_duration_seconds

Tipo: Histograma

Rótulos:

- `email_type` - Tipo de template de email

Buckets: `[0.1, 0.5, 1.0, 2.0, 5.0, 10.0, 30.0]` segundos

Descrição: Duração da chamada da API Mailjet. Tempo necessário para enviar o email via API Mailjet (não o tempo de entrega).

omnicrm_mailjet_api_errors_total

Tipo: Contador

Rótulos:

- `email_type` - Tipo de template de email
- `error_type` - Categoria de erro

Descrição: Número total de erros da API Mailjet. Incrementado quando falhas no envio de emails ocorrem.

omnicrm_mailjet_emails_sent_total

Tipo: Contador

Rótulos:

- `email_type` - Tipo de template de email
- `template_id` - ID do template Mailjet

Descrição: Número total de emails enviados com sucesso via Mailjet. Distinto das chamadas da API, pois uma chamada pode enviar para múltiplos destinatários.

omnicrm_mailjet_email_recipients_total

Tipo: Contador

Rótulos:

- `email_type` - Tipo de template de email

Descrição: Número total de destinatários de email em todos os emails enviados.

---

6. Métricas de Jobs em Segundo Plano

As métricas de jobs em segundo plano rastreiam operações assíncronas como cadeias de playbooks e tarefas agendadas. Veja Sistema de Provisionamento para detalhes sobre jobs em segundo plano.

omnicrm_background_jobs_total

Tipo: Contador

Rótulos:

- `job_type` - Categoria do job: `playbook_single`, `playbook_chain`, `periodic_task`

Descrição: Número total de jobs em segundo plano iniciados. Rastreado via o gerenciador de contexto `BackgroundJobTimer`.

omnicrm_background_jobs_active

Tipo: Gauge

Rótulos:

- `job_type` - Categoria do job

Descrição: Número de jobs em segundo plano atualmente em execução. Incrementado no início do job, decrementado na conclusão do job.

omnicrm_background_job_duration_seconds

Tipo: Histograma

Rótulos:

- `job_type` - Categoria do job

- `status` - Resultado do job: `success`, `failed`

Buckets: `[1, 5, 10, 30, 60, 120, 180, 300, 600, 1800, 3600]` segundos (1s a 1h)

Descrição: Tempo de execução do job em segundo plano. Inclui a duração total de operações de múltiplas etapas.

omnicrm_background_job_errors_total

Tipo: Contador

Rótulos:

- `job_type` - Categoria do job
- `error_type` - Categoria de erro

Descrição: Número total de erros de jobs em segundo plano. Incrementado quando jobs em segundo plano falham com exceções.

---

7. Métricas HTTP do Flask (Auto-geradas)

Essas métricas são geradas automaticamente pela biblioteca `prometheus-flask-exporter` e rastreiam todas as requisições HTTP à API.

flask_http_request_duration_seconds

Tipo: Histograma

Rótulos:

- `method` - Método HTTP: `GET`, `POST`, `PUT`, `DELETE`, etc.
- `endpoint` - Nome da rota Flask
- `status` - Código de status HTTP

Descrição: Duração da requisição HTTP para todos os endpoints da API. Instrumentado automaticamente.

flask_http_request_total

Tipo: Contador

Rótulos:

- `method` - Método HTTP
- `endpoint` - Nome da rota Flask
- `status` - Código de status HTTP

Descrição: Total de requisições HTTP por endpoint, método e código de status.

flask_http_request_exceptions_total

Tipo: Contador

Rótulos:

- `method` - Método HTTP
- `endpoint` - Nome da rota Flask

Descrição: Total de exceções não tratadas em requisições HTTP. Indica bugs ou erros inesperados.

---

8. Métricas de Erro da API (Reservadas)

Essas métricas estão definidas, mas não estão atualmente instrumentadas. Elas são reservadas para uso futuro.

omnicrm_api_errors_total

Tipo: Contador

Rótulos:

- `endpoint` - Endpoint da API
- `error_type` - Categoria de erro
- `status_code` - Código de status HTTP

Status: Definido, mas não utilizado ativamente

omnicrm_api_auth_failures_total

Tipo: Contador

Rótulos:

- `auth_method` - Método de autenticação: `jwt`, `api_key`, `ip_whitelist`
- `reason` - Motivo da falha

Status: Definido, mas não utilizado ativamente

---

9. Métrica de Informações da Aplicação

omnicrm_api

Tipo: Info

Rótulos:

- `version` - String da versão da API
- `environment` - Nome do ambiente: `production`, `staging`, `development`

Descrição: Métrica de informações da API OmniCRM. Definida uma vez na inicialização da aplicação com informações de versão e ambiente.

Atualizações Periódicas

update_db_pool_metrics(engine)

Chamado automaticamente a cada 30 segundos para atualizar os gauges do pool de conexões do banco de dados.

Configuração do Prometheus

Configuração de Scrape

Adicione o OmniCRM ao seu `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'
```

Exemplo de Alertas

Alta Taxa de Falha de Provisionamento

```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: "Alta taxa de falha de jobs de provisionamento"
    description: "{{ $value | humanizePercentage }} dos jobs de
provisionamento estão falhando"
```

Pool de Conexões do Banco de Dados Exaurido

```
- alert: DatabasePoolExhausted
  expr: omnicrm_db_pool_overflow > 0
  for: 2m
  labels:
    severity: critical
  annotations:
    summary: "Transbordo do pool de conexões do banco de dados
detectado"
    description: "O pool de conexões está usando conexões de
transbordo, pode indicar que o tamanho do pool é muito pequeno"
```

Consultas ao Banco de Dados Lentas

```
- alert: SlowDatabaseQueries
  expr: |
    histogram_quantile(0.99,
      rate(omnicrm_db_query_duration_seconds_bucket[5m])
    ) > 1.0
  for: 5m
  labels:
    severity: warning
  annotations:
    summary: "Consultas lentas ao banco de dados detectadas"
    description: "O tempo de consulta do 99º percentil é {{ $value
}}s"
```

API OCS Fora do Ar

```
- 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: "Taxa de falha da API OCS crítica"
    description: "{{ $value | humanizePercentage }} das chamadas
da API OCS estão falhando"
```

Problemas de Pagamento Stripe

```
- alert: StripePaymentFailures
  expr: rate(omnicrm_stripe_payment_failures_total[5m]) > 5
  for: 5m
  labels:
    severity: warning
  annotations:
    summary: "Falhas elevadas de pagamento Stripe"
    description: "{{ $value }} falhas de pagamento por segundo"
```

Melhores Práticas

Estratégia de Monitoramento

1. **Configurar Alertas Principais** - Configure alertas para métricas críticas:
    - Taxa de falha de provisionamento > 10%
    - Exaustão do pool de conexões do banco de dados
    - Falhas na API OCS/CGRateS
    - Erros no processamento de pagamentos Stripe

2. **Rastrear Métricas Comerciais** - Monitore KPIs operacionais:
    - Receita processada via Stripe
    - Throughput de provisionamento
    - Taxas de entrega de email para clientes
3. **Monitoramento de Desempenho** - Observe a degradação:
    - Percentis de tempo de resposta da API
    - Desempenho de consultas ao banco de dados
    - Latência de APIs externas
4. **Planejamento de Capacidade** - Use métricas para prever:
    - Dimensionamento do pool de conexões do banco de dados
    - Escalonamento de trabalhadores de jobs em segundo plano
    - Capacidade do servidor da API

Retenção de Métricas

Recomendações de Retenção do Prometheus:

- **15 dias** - Métricas de alta resolução (intervalo de scrape de 15s)
- **90 dias** - Métricas downsampled (agregação de 5m)
- **1 ano** - Métricas agregadas de longo prazo (agregação de 1h)

Use Thanos, Cortex ou VictoriaMetrics para armazenamento de longo prazo e consulta global.

Resolução de Problemas

Métricas Não Aparecendo

Por padrão, o endpoint `/metrics` é exposto apenas para fontes internas (não públicas). Você pode alterar isso na sua configuração do nginx, se necessário.

Verifique o endpoint de métricas:

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

Verifique se o Prometheus pode fazer scrape:

```
# Verifique a página de alvos do Prometheus
http://prometheus:9090/targets
```

Métricas Específicas Faltando

Algumas métricas só são criadas após seu primeiro uso:

- Métricas de provisionamento aparecem após o primeiro job de provisionamento
- Métricas do Stripe aparecem após o primeiro pagamento
- Métricas do OCS aparecem após a primeira operação de cobrança

Problemas de Alta Cardinalidade

Se o Prometheus estiver lento ou consumindo memória excessiva, verifique por rótulos de alta cardinalidade:

```
# Contar séries temporais únicas por métrica
count by (__name__) ({__name__=~".+"})
```

Métricas com mais de 10.000 séries podem indicar um problema de cardinalidade.

Resumo das Métricas

Total de Métricas: 34 (31 personalizadas + 3 métricas Flask auto-geradas)

Por Tipo:

- **Contadores:** 20

- **Gauges:** 5
- **Histogramas:** 8
- **Info:** 1

Por Categoria:

- Provisionamento: 5 métricas
- Banco de Dados: 5 métricas
- OCS/CGRateS: 5 métricas
- Stripe: 5 métricas
- Mailjet: 5 métricas
- Jobs em Segundo Plano: 4 métricas
- HTTP/Flask: 3 métricas
- Informações da Aplicação: 1 métrica
- Reservadas (uso futuro): 2 métricas

Documentação Relacionada

- Arquitetura do Sistema - Design geral do sistema e relações entre componentes
- Sistema de Provisionamento - Fluxo de trabalho de provisionamento e execução de jobs
- Playbooks do Ansible - Estrutura e execução de playbooks
- Documentação da API - Endpoints da API e autenticação
- Visão Geral de Cobrança - Sistema de cobrança e faturamento
- Guia do Sistema de Pagamento - Integração Stripe e processamento de pagamentos
- Integração Mailjet - Configuração do serviço de email
- Configuração de Administração - Opções de configuração do sistema

Gestão de Métodos de Pagamento

O sistema de Métodos de Pagamento do OmniCRM permite que clientes e funcionários gerenciem cartões de pagamento de forma segura usando **processamento de pagamento de múltiplos fornecedores** (Stripe, PayPal, etc.). Os métodos de pagamento possibilitam faturamento automático para serviços, pagamentos únicos e cobranças recorrentes sem armazenar dados sensíveis do cartão no OmniCRM.

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

Visão Geral

O sistema de métodos de pagamento fornece:

- **Armazenamento Seguro de Cartões** - Cartões tokenizados por fornecedores de pagamento (Stripe, PayPal), nunca armazenados no OmniCRM
- **Suporte a Múltiplos Fornecedores** - Métodos de pagamento Stripe e PayPal suportados
- **Vários Cartões** - Clientes podem armazenar múltiplos métodos de pagamento
- **Seleção Padrão** - Designar método de pagamento preferido para cobranças automáticas
- **Rastreamento de Vencimento** - Monitorar e atualizar cartões que estão prestes a vencer
- **Autoatendimento** - Clientes podem gerenciar seus próprios cartões através do `Self-Care Portal <self_care_portal>`
- **Gerenciamento de Funcionários** - Funcionários de suporte podem adicionar/remover cartões em nome dos clientes

Métodos de Pagamento Suportados:

- **Cartões** (via Stripe ou PayPal)
  - Cartões de Crédito (Visa, Mastercard, American Express, Discover)
  - Cartões de Débito
  - Cartões Pré-pagos (se suportados pela rede de cartões)
- **Contas do PayPal** (via integração com o PayPal)

Não Armazenado no OmniCRM:

Os detalhes do cartão são tokenizados pelos fornecedores de pagamento e armazenados de forma segura. O OmniCRM apenas armazena:

- Fornecedor de pagamento (stripe, paypal)
- Marca do cartão (Visa, Mastercard, etc.)
- Últimos 4 dígitos
- Mês/ano de vencimento
- Nome/nome de usuário do titular do cartão
- Token do método de pagamento específico do fornecedor

Acessando Métodos de Pagamento

Da Página do Cliente:

1. Navegue até **Clientes → [Selecionar Cliente]**
2. Clique na aba **Faturamento**
3. Role até a seção **Métodos de Pagamento**

Ou diretamente:

Do Painel de Cartões Vencendo:

Veja todos os clientes com cartões que estão prestes a vencer:

Isso mostra uma lista de cartões que estão vencendo em até 60 dias.

Lista de Métodos de Pagamento

A tabela de métodos de pagamento exibe todos os cartões armazenados para um cliente:

Descrições das Colunas:

- **Nome de Usuário** - Nome amigável para o cartão (por exemplo, "Cartão Pessoal", "Visa de Trabalho")
- **Emissor** - Marca do cartão e últimos 4 dígitos
- **Vencimento** - Mês/ano de expiração (formato MM/YYYY)
- **Adicionado** - Data em que o cartão foi adicionado à conta
- **Padrão** - Marca de seleção indica o método de pagamento padrão para cobranças automáticas

Ações por Cartão:

Cada linha tem um menu de ações (⋮) com opções:

- **Definir como Padrão** - Tornar este o método de pagamento padrão
- **Excluir** - Remover cartão da conta

Adicionando um Método de Pagamento

Clique em **"Adicionar Método de Pagamento"** para abrir o modal de pagamento seguro.

Passo 1: Inserir Detalhes do Cartão

O formulário de pagamento seguro aparece (alimentado por Stripe Elements ou PayPal SDK):

Campos Obrigatórios:

- **Informações do Cartão** - Número do cartão, vencimento, CVC (validado pelo Stripe)
- **Nome do Titular do Cartão** - Nome no cartão
- **País/Região** - País de faturamento

Campos Opcionais:

- **Nome de Usuário do Cartão** - Rótulo amigável para distinguir entre cartões

Segurança:

- Detalhes do cartão inseridos diretamente em um iframe seguro hospedado pelo fornecedor (Stripe Elements / PayPal SDK)
- O OmniCRM nunca vê ou armazena números de cartões completos
- Conformidade com PCI DSS gerenciada pelo fornecedor de pagamento
- Validação em tempo real impede números de cartões inválidos

Passo 2: Enviar e Tokenizar

Quando você clica em **"Adicionar Método de Pagamento"**:

1. **Validação do Lado do Cliente:**
   - O fornecedor de pagamento valida o formato do número do cartão
   - Verifica se a data de vencimento está no futuro
   - Verifica o formato do CVC
2. **Tokenização:**
   - Detalhes do cartão enviados diretamente ao fornecedor de pagamento (não ao OmniCRM)
   - O fornecedor cria um token seguro (por exemplo, `pm_1A2B3C4D` para Stripe)
   - Token retornado ao OmniCRM
3. **Processamento no Servidor:**
   - O OmniCRM salva o token no registro do cliente com o identificador do fornecedor
   - Armazena os últimos 4 dígitos, marca, vencimento e nome do fornecedor para exibição
   - Nenhum número de cartão completo toca nos servidores do OmniCRM

Passo 3: Confirmação

Mensagem de sucesso aparece:

> Seu Visa terminando em 1234 foi adicionado à sua conta.

O novo cartão aparece na tabela de métodos de pagamento.

Seleção Automática Padrão:

- Se este for o primeiro cartão do cliente, ele é automaticamente definido como padrão
- Se o cliente já tiver cartões, o novo cartão é adicionado como não padrão
- O cliente pode alterar o padrão após a adição

Definindo o Método de Pagamento Padrão

O método de pagamento padrão é usado para:

- Cobranças automáticas de serviços recorrentes
- Pagamentos de faturas
- Recargas e reabastecimentos
- Transações únicas (a menos que especificado de outra forma)

Para Alterar o Padrão:

1. Localize o cartão que deseja definir como padrão na tabela de métodos de pagamento
2. Clique no **menu de ações (⋮)** ao lado do cartão
3. Selecione **"Definir como Padrão"**
4. A confirmação aparece

   O Visa terminando em 5678 agora é seu método de pagamento padrão.

A marca de seleção se move para o cartão recém-selecionado.

Indicador Visual:

Cartões padrão mostram:

na coluna Padrão, tipicamente com um distintivo de marca de seleção verde.

Excluindo um Método de Pagamento

Remova cartões que estão expirados, perdidos ou que não são mais necessários.

Passo 1: Iniciar Exclusão

1. Encontre o cartão a ser excluído na tabela de métodos de pagamento
2. Clique no **menu de ações (⋮)**
3. Selecione **"Excluir"**

Passo 2: Confirmar Exclusão

Um modal de confirmação aparece:

> Você tem certeza de que deseja excluir este método de pagamento?
>
> Cartão: Visa terminando em 1234 Vencimento: 12/2026
>
> ⚠ Aviso: Se este for seu único método de pagamento, você precisará adicionar um novo para continuar usando serviços que requerem faturamento automático.
>
> [Cancelar] [Excluir Método de Pagamento]

Clique em **"Excluir Método de Pagamento"** para confirmar.

Passo 3: Exclusão Completa

Mensagem de sucesso:

O cartão é removido da tabela e excluído do fornecedor de pagamento.

Restrições Importantes:

- **Não é possível excluir o padrão se outros cartões existirem** - Defina um cartão diferente como padrão primeiro

- **Aviso se excluir o último cartão** - Serviços que requerem pagamento podem ser suspensos
- **Sem desfazer** - A exclusão é permanente; o cliente deve re-adicionar o cartão se necessário

Gerenciando Cartões Vencendo

O OmniCRM rastreia as datas de vencimento dos cartões e fornece ferramentas para atualizar proativamente os cartões que estão prestes a vencer.

Painel de Cartões Vencendo

Navegue até **Faturamento → Cartões Vencendo** para ver uma lista em todo o sistema:

> Cliente Cartão Vencimento Dias até Ação John Smith Visa ****1234 02/2025 12 dias Atualizar Acme Corp MC5678 03/2025 45 dias Atualizar Jane Doe Amex**9012 01/2025 VENCIDO Atualizar

Filtros:

- **Faixa de Vencimento** - Próximos 30/60/90 dias ou já expirados

- **Tipo de Cliente** - Individual vs Empresarial
- **Tipo de Serviço** - Filtrar por serviço que requer método de pagamento

Ações:

- **Atualizar** - Abre a página de métodos de pagamento do cliente para adicionar um novo cartão
- **Notificar** - Enviar lembrete por e-mail ao cliente (se Mailjet configurado)

Notificações de Vencimento

Se o Mailjet estiver configurado, e-mails automáticos são enviados:

- **60 dias antes do vencimento** - Primeiro lembrete
- **30 dias antes do vencimento** - Segundo lembrete
- **7 dias antes do vencimento** - Aviso final
- **No vencimento** - Aviso de que o cartão expirou

Os clientes podem clicar em um link no e-mail para atualizar seu método de pagamento através do portal de Autoatendimento.

Variáveis do Modelo de E-mail:

Os modelos do Mailjet recebem:

- Nome do cliente
- Marca do cartão e últimos 4 dígitos
- Data de vencimento
- Link para a página de métodos de pagamento do Autoatendimento

Veja `integrations_mailjet` para configuração do modelo de e-mail.

Atualizando um Cartão Vencendo

Fluxo de Trabalho Recomendado:

1. O cliente recebe um e-mail de notificação de vencimento
2. O cliente faz login no portal de Autoatendimento

3. Navega até **Faturamento → Métodos de Pagamento**
4. Clica em **"Adicionar Método de Pagamento"**
5. Insere os novos detalhes do cartão (mesmo cartão com vencimento atualizado ou cartão de substituição)
6. Define o novo cartão como padrão
7. Exclui o cartão antigo/vencido

Fluxo de Trabalho do Funcionário:

Se o cliente ligar para o suporte:

1. O funcionário abre a conta do cliente
2. Navega até **Faturamento → Métodos de Pagamento**
3. Adiciona um novo cartão em nome do cliente (o cliente fornece os detalhes por telefone)
4. Define o novo cartão como padrão
5. Exclui o cartão vencido
6. Confirma com o cliente

Aviso

Nunca peça aos clientes para enviar detalhes do cartão por e-mail ou mensagem de texto. Sempre use:

- Portal seguro de Autoatendimento para autoatendimento
- Telefone com funcionários inserindo detalhes diretamente no sistema
- Pessoalmente em local de varejo

O Que Acontece Quando os Cartões Expiram

Quando um cartão de pagamento atinge sua data de vencimento e não é atualizado:

Efeitos Imediatos:

1. **Pagamentos Automáticos Falham**
    - O fornecedor de pagamento rejeita transações com cartões expirados
    - Renovações de serviços mensais falham ao processar

    - Recargas automáticas falham
    - Pagamentos automáticos de faturas falham
2. **Notificações ao Cliente**
    - O sistema tenta cobrar o cartão
    - Notificação de falha de pagamento enviada
    - E-mail "Atualizar Método de Pagamento" enviado com link para o portal de Autoatendimento
3. **Mudanças no Status do Serviço**
    - **Serviços Pós-Pagos** - Podem continuar temporariamente com saldo em aberto
    - **Serviços Pré-Pagos** - Suspensão do serviço quando o saldo se esgota
    - **Serviços de Renovação Automática** - Renovação falha, o serviço pode expirar

Ações Subsequentes:

Dia 1-3 (Período de Graça):

- O serviço continua normalmente
- O cliente recebe o primeiro aviso de falha de pagamento
- O sistema tenta uma nova tentativa (dependendo da configuração)

Dia 4-7:

- Segunda tentativa de pagamento (se configurado)
- E-mail de aviso enviado
- O atendimento ao cliente pode entrar em contato com o cliente

Dia 8-14:

- O serviço pode ser suspenso por falta de pagamento
- O status suspenso impede o uso, mas preserva a conta
- O cliente pode restaurar atualizando o método de pagamento e pagando o saldo em aberto

Dia 15+:

- O serviço pode ser encerrado por falta de pagamento

- Inventário (cartões SIM, equipamentos) marcado para devolução
- Último aviso enviado
- Conta encaminhada para cobrança (se aplicável)

Prevenindo Interrupção do Serviço:

Para evitar interrupção do serviço:

- Atualize os cartões **30 dias antes do vencimento**
- Adicione múltiplos métodos de pagamento para redundância
- Ative alertas de falha de pagamento
- Monitore o painel de Cartões Vencendo semanalmente

Restaurando o Serviço Após o Vencimento:

Se o serviço foi suspenso devido a um cartão expirado:

1. Adicione um novo método de pagamento válido
2. Defina como padrão
3. Pague o saldo em aberto (se houver)
4. Entre em contato com o suporte para reativar o serviço
5. Serviço restaurado em minutos a horas

Segurança do Método de Pagamento

Tokenização

O OmniCRM usa tokenização do fornecedor para garantir segurança:

1. **O cliente insere o cartão** → Enviado diretamente aos servidores do fornecedor de pagamento
2. **O fornecedor valida e tokeniza** → Cria um token único
3. **Token armazenado no OmniCRM** → Número completo do cartão nunca é armazenado

4. **Processamento de pagamento** → Token enviado ao fornecedor, fornecedor cobra o cartão

O Que o OmniCRM Armazena:

```
{
  "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
}
```

O Que o OmniCRM NÃO Armazena:

- Número completo do cartão
- Código CVV/CVC
- Dados da faixa magnética
- Números PIN

Conformidade com PCI

Ao usar formulários de pagamento hospedados pelo fornecedor:

- **Escopo PCI reduzido** - Dados do cartão nunca tocam nos servidores do OmniCRM
- **Campos hospedados pelo fornecedor** - A entrada do cartão ocorre no iframe seguro do fornecedor
- **Sem armazenamento de cartões** - Tokens usados em vez de dados brutos do cartão
- **Transmissão segura** - Toda comunicação via HTTPS/TLS

Veja `Payment System Guide <payment_system_guide>` para detalhes de segurança do fornecedor de pagamento.

Fluxos de Trabalho Comuns

Fluxo de Trabalho 1: Cliente Adiciona Primeiro Método de Pagamento

Cenário: Novo cliente se inscrevendo para o serviço

1. O cliente cria uma conta
2. Seleciona o plano de serviço
3. Solicitado a adicionar um método de pagamento durante o checkout
4. Insere os detalhes do cartão no modal do Stripe
5. Cartão tokenizado e salvo
6. Automaticamente definido como padrão
7. Serviço provisionado
8. Primeiro pagamento processado

Fluxo de Trabalho 2: Cliente Atualiza Cartão Vencendo

Cenário: Cartão de crédito prestes a vencer

1. O cliente recebe notificação por e-mail (60 dias antes do vencimento)
2. Faz login no portal de Autoatendimento
3. Navega até **Faturamento → Métodos de Pagamento**
4. Revê o cartão atual que vence em 12/2025
5. Clica em **"Adicionar Método de Pagamento"**
6. Insere o cartão de substituição com vencimento 12/2028
7. Define o novo cartão como padrão
8. Exclui o cartão antigo
9. E-mail de confirmação enviado

Fluxo de Trabalho 3: Funcionário Ajuda Cliente por Telefone

Cenário: Cliente liga: "Meu cartão foi recusado"

1. O cliente liga para o suporte
2. O funcionário verifica a identidade (perguntas de segurança)
3. O funcionário verifica os métodos de pagamento: Cartão expirado 01/2025
4. Funcionário: "Seu cartão expirou. Você tem um novo cartão?"
5. O cliente fornece os novos detalhes do cartão por telefone
6. O funcionário navega até **Clientes → [Cliente] → Faturamento**
7. Clica em **"Adicionar Método de Pagamento"**
8. Insere os detalhes do cartão enquanto o cliente os lê
9. Define o novo cartão como padrão
10. Exclui o cartão expirado
11. Tenta o pagamento falhado novamente
12. Confirma com o cliente: "Pagamento bem-sucedido, serviço restaurado"

Fluxo de Trabalho 4: Cliente Empresarial com Múltiplos Cartões

Cenário: A empresa quer cartões diferentes para diferentes propósitos

1. O cliente empresarial adiciona o cartão principal (Visa terminando em 1111)
2. Define como padrão para cobranças mensais de serviços
3. Adiciona cartão de backup (Mastercard terminando em 2222) para recargas
4. Adiciona cartão de compras (Amex terminando em 3333) para compras de equipamentos
5. Ao fazer uma recarga, seleciona Mastercard manualmente no checkout
6. O Visa padrão ainda é usado para faturamento automático mensal

Fluxo de Trabalho 5: Gerenciando Cartões Vencendo (Admin)

Cenário: Gerenciamento proativo de vencimentos

1. O administrador navega até **Faturamento → Cartões Vencendo**
2. Filtros: "Próximos 30 dias"
3. Vê 15 clientes com cartões vencendo
4. Seleciona todos → **"Enviar E-mails de Lembrete"**
5. O Mailjet envia e-mails personalizados para cada cliente
6. Os clientes atualizam os cartões via Autoatendimento
7. O administrador revisa a lista 1 semana depois
8. Liga para os clientes restantes que não atualizaram
9. Auxilia com atualizações de cartões por telefone

Resolução de Problemas

"Cartão recusado" ao adicionar método de pagamento

- **Causa:** O fornecedor de pagamento rejeitou o cartão (fundos insuficientes, prevenção de fraude, recusa do emissor)
- **Solução:**
  - Tente um cartão diferente
  - Entre em contato com o emissor do cartão para autorizar a transação
  - Certifique-se de que o cartão suporta compras online
  - Verifique se o endereço de faturamento corresponde ao cartão registrado

"Erro ao adicionar método de pagamento" (erro genérico)

- **Causa:** Erro da API do fornecedor de pagamento ou problema de rede
- **Solução:**
  - Atualize a página e tente novamente
  - Verifique a conexão com a internet

    - Verifique se a configuração do fornecedor de pagamento está correta nas configurações do sistema
    - Verifique o console do navegador para mensagens de erro específicas
    - Tente um navegador diferente (desative bloqueadores de anúncios)

Não é possível excluir método de pagamento (botão desativado)

- **Causa:** Tentando excluir o cartão padrão, ou é o único cartão
- **Solução:**
    - Defina um cartão diferente como padrão primeiro
    - Se for o único cartão, adicione um novo cartão antes de excluir

Cartão aparece como expirado, mas não na lista de "Cartões Vencendo"

- **Causa:** O cartão expirou recentemente, cache não atualizado
- **Solução:**
    - Atualize a página
    - Verifique os filtros no painel de Cartões Vencendo
    - Cartões expirados podem se mover para uma visualização diferente

Novo cartão não aparece imediatamente

- **Causa:** A página não foi atualizada após a adição do cartão
- **Solução:**
    - A tabela de métodos de pagamento deve atualizar automaticamente
    - Se não, atualize manualmente o navegador
    - Verifique se ocorreu um erro durante o processo de adição

Modal de pagamento não carrega

- **Causa:** SDK do fornecedor de pagamento não carregando, problema com a chave da API ou extensão do navegador bloqueando
- **Solução:**
    - Verifique o console do navegador para erros
    - Desative bloqueadores de anúncios e proteção de rastreamento

    - Verifique a configuração do fornecedor de pagamento nas configurações do sistema
    - Certifique-se de que o script do SDK do fornecedor carrega (verifique a aba Rede)
    - Tente o modo de navegação anônima/privada

Cliente não recebe notificações de vencimento

- **Causa:** Mailjet não configurado ou modelo de e-mail ausente
- **Solução:**
    - Verifique as credenciais do Mailjet em crm_config.yaml
    - Verifique se o modelo de e-mail existe para vencimento do cartão
    - Confirme se o endereço de e-mail do cliente é válido
    - Verifique os logs do Mailjet para falhas de entrega

Melhores Práticas

Para Clientes:

- Adicione o método de pagamento antes da ativação do serviço para evitar atrasos
- Mantenha pelo menos 2 cartões registrados para redundância
- Atualize cartões que estão prestes a vencer com 30+ dias de antecedência
- Exclua cartões antigos/vencidos para evitar confusão
- Use nomes de usuário descritivos ("Visa Pessoal", "Amex de Trabalho")
- Verifique se o método de pagamento padrão está correto para faturamento automático

Para Funcionários de Suporte:

- Verifique a identidade do cliente antes de acessar os métodos de pagamento
- Nunca peça aos clientes para enviar detalhes do cartão por e-mail/SMS/chat
- Processe adições de cartões imediatamente durante as chamadas (não adie)

- Confirme se o novo cartão está definido como padrão após a adição
- Exclua cartões antigos apenas após confirmar que o novo cartão funciona
- Teste o pagamento após atualizar o cartão expirado (processo de autorização de £0,01)

Para Administradores:

- Monitore o painel de Cartões Vencendo semanalmente
- Envie e-mails de lembrete 60/30/7 dias antes do vencimento
- Mantenha as chaves de teste/produção do fornecedor de pagamento separadas para desenvolvimento vs produção
- Certifique-se de que os modelos do Mailjet estão configurados para notificações de vencimento
- Revise relatórios de falha de pagamento para identificar cartões expirados
- Treine a equipe sobre procedimentos seguros de manuseio de cartões

Documentação Relacionada

- `Payment System Guide <payment_system_guide>` - Referência completa da API de pagamento e arquitetura
- `payments_process` - Processamento de pagamentos com métodos de pagamento armazenados
- `payments_invoices` - Pagamento automático de faturas usando o cartão padrão
- `features_topup_recharge` - Sistema de recarga usando métodos de pagamento
- `basics_payment` - Conceitos gerais de pagamento e faturamento
- `customer_care` - Portal de Autoatendimento para clientes gerenciarem seus próprios cartões

Guia da API do Sistema de Pagamento OmniCRM

Visão Geral

O sistema de pagamento OmniCRM fornece uma infraestrutura de processamento de pagamento abrangente e **independente de fornecedor**. Hoje, ele suporta Stripe e PayPal, mas a arquitetura modular permite a integração com qualquer provedor de pagamento (Square, Adyen, Braintree, etc.) sem alterar o código do aplicativo.

Este documento cobre todas as APIs de pagamento e fluxos de trabalho disponíveis no sistema.

> **Integração com Ansible Playbook**: Para implementar essas APIs de pagamento em playbooks de provisionamento, veja **Cobranças e Pagamentos a partir de Playbooks**

Índice

1. Arquitetura Modular
2. Conceitos Básicos
3. Documentos Financeiros
4. APIs de Métodos de Pagamento
5. APIs de Fluxo de Pagamento
6. APIs de Carteira
7. Resumo da Referência da API
8. Casos de Uso Comuns

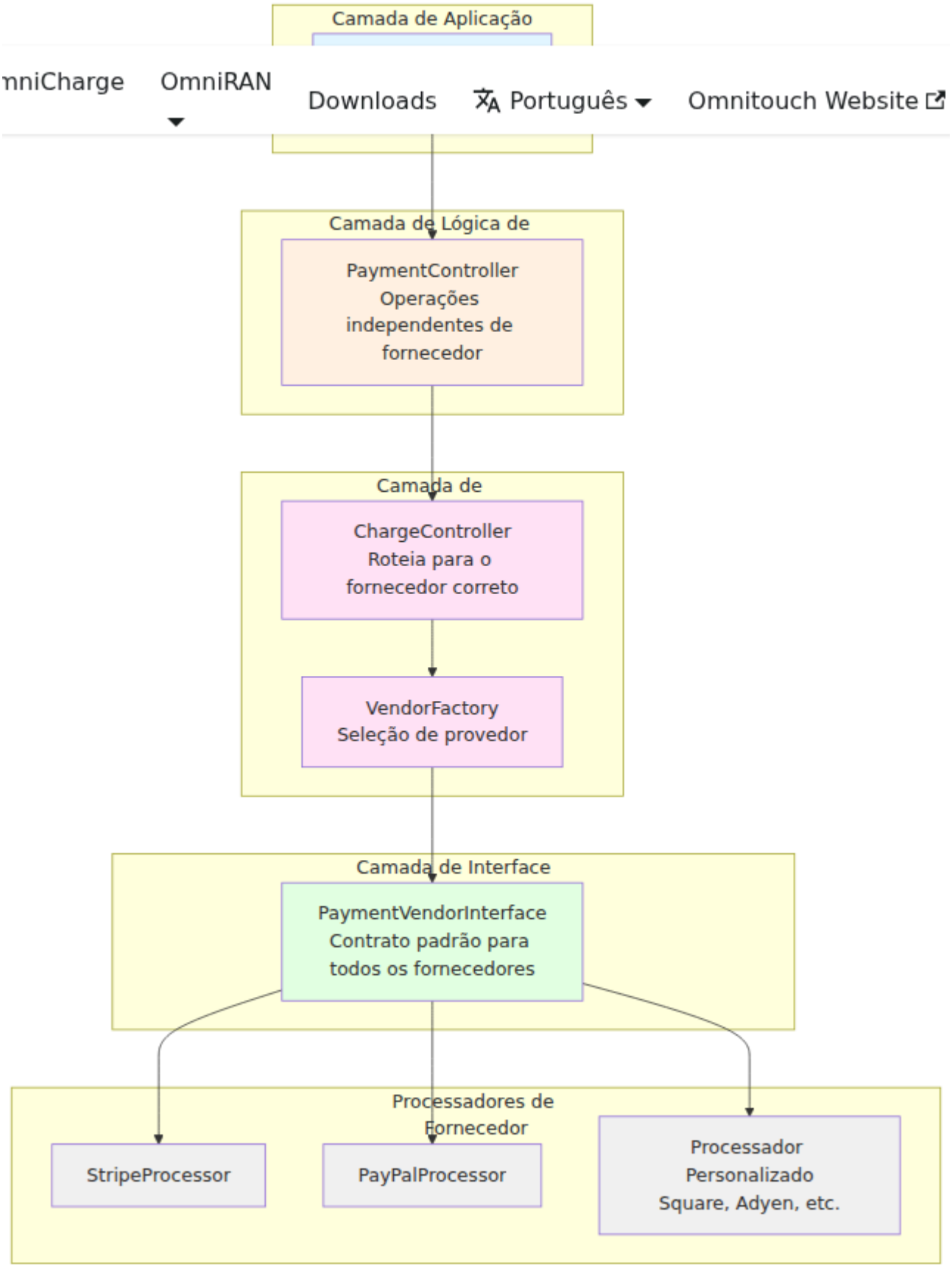
Arquitetura Modular

Por que Independente de Fornecedor?

O sistema usa **camadas de abstração** para separar a lógica de negócios das especificidades do fornecedor de pagamento. Isso significa:

- □ Adicionar novos provedores de pagamento sem tocar no código do aplicativo
- □ Trocar de fornecedores sem migrações de banco de dados
- □ Suportar múltiplos fornecedores simultaneamente
- □ API consistente, independentemente do provedor de backend

Camadas da Arquitetura



Adicionando um Novo Fornecedor de Pagamento

Para adicionar um novo provedor (por exemplo, Square, Adyen, Braintree), entre em contato com sua equipe técnica do OmniCRM.

Para detalhes de configuração sobre fornecedores existentes (Stripe, PayPal), veja **Configuração do Fornecedor**.

Visão Geral do Processo:

Passos:

1. **Implementar a classe do processador** com a interface padrão `PaymentVendorInterface` (autorizar, capturar, cobrar, reembolsar, liberar)
2. **Registrar o processador** na VendorFactory com o nome do fornecedor
3. **Testar a integração** com o ambiente sandbox do fornecedor
4. **Implantar** - Todas as APIs existentes suportam automaticamente o novo fornecedor

Resultado: Uma vez implantado, o novo fornecedor funciona perfeitamente:

```
# Adicionar método de pagamento 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"
  }'
```

Todas as APIs de pagamento suportam automaticamente o novo fornecedor sem necessidade de alterações no código do aplicativo.

Tempo de Integração: Normalmente de 1 a 2 dias para um novo processador de fornecedor.

Configuração do Fornecedor

Configuração do Stripe

O Stripe é o fornecedor de pagamento principal, fornecendo processamento de cartões com suporte a 3D Secure.

1. Obter Chaves da API do Stripe

- Inscreva-se em https://stripe.com
- Navegue até Desenvolvedores → Chaves da API
- Copie sua **Chave Publicável** (pk_live_... ou pk_test_...)
- Copie sua **Chave Secreta** (sk_live_... ou sk_test_...)

Importante:

- Use **chaves de teste** (pk_test_/sk_test_) para desenvolvimento
- Use **chaves ao vivo** (pk_live_/sk_live_) apenas para produção
- **Nunca** comite chaves da API no controle de versão

2. Configurar Backend

Adicione ao `crm_config.yaml`:

```
payment_vendors:
  stripe:
    api_key: "sk_live_YOUR_SECRET_KEY_HERE"
    publishable_key: "pk_live_YOUR_PUBLISHABLE_KEY_HERE"
```

Ou via variáveis de ambiente:

```
export STRIPE_SECRET_KEY="sk_live_YOUR_SECRET_KEY_HERE"
export STRIPE_PUBLISHABLE_KEY="pk_live_YOUR_PUBLISHABLE_KEY_HERE"
```

3. Configurar Frontend

Adicione ao `.env`:

```
REACT_APP_STRIPE_PUBLISHABLE_KEY=pk_live_YOUR_PUBLISHABLE_KEY_HERE
```

Nota de Segurança: Apenas a chave publicável deve ir no frontend. A chave secreta **nunca** deve ser exposta ao navegador.

Configuração do PayPal

O PayPal fornece armazenamento de cartões e pagamentos de contas do PayPal.

Configuração

Adicione ao `crm_config.yaml`:

```yaml
payment_vendors:
  paypal:
    client_id: "AXx_YOUR_CLIENT_ID_HERE"
    client_secret: "ELx_YOUR_CLIENT_SECRET_HERE"
    mode: "live"  # ou "sandbox" para testes
```

Conformidade com PCI

Como o OmniCRM Mantém a Conformidade com PCI:

- Dados do cartão inseridos diretamente em iframes hospedados pelo fornecedor (Stripe Elements, PayPal Card Fields)
- O OmniCRM **nunca vê ou armazena** números completos de cartões
- Apenas métodos de pagamento tokenizados são armazenados no banco de dados
- Escopo de conformidade PCI reduzido para seu negócio

Métricas de Processamento de Pagamento

O OmniCRM fornece métricas abrangentes para monitorar operações de processamento de pagamento. Para detalhes completos sobre métricas de pagamento do Stripe, incluindo rastreamento de chamadas de API, volumes de pagamento, taxas de falha e tempos de resposta, veja Monitoramento & Métricas.

O Esquema do Banco de Dados é Independente de Fornecedor

O esquema do banco de dados suporta qualquer fornecedor de pagamento sem migrações:

Exemplo - Salvando um cartão Square:

```
{
  "vendor": "square",
  "vendor_payment_method_id": "sq_card_abc123",
  "payment_type": "card"
}
```

Sem alterações de código. Sem migrações. Funciona perfeitamente.

Conceitos Básicos

Modelos de Dados

PaymentMethod

Armazenamento independente de fornecedor dos métodos de pagamento do cliente.

```
{
  "payment_method_id": 789,
  "customer_id": 123,
  "vendor": "stripe",                    // 'stripe', 'paypal' ou
qualquer fornecedor adicionado
  "vendor_payment_method_id": "pm_xxx", // ID interno do
fornecedor
  "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": "Meu Cartão Visa",
  "status": "active"
}
```

PaymentAuthorization

Registros de autorização de compromisso em duas fases (reservar fundos).

```
{
  "authorization_id": 301,
  "customer_id": 123,
  "payment_method_id": 789,
  "vendor": "stripe",                    // Qual fornecedor
autorizou
  "vendor_authorization_id": "auth_xxx", // ID de autorização do
fornecedor
  "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

Pagamentos capturados/completados.

```
{
  "capture_id": 103,
  "authorization_id": 301,
  "customer_id": 123,
  "payment_method_id": 789,
  "vendor": "stripe",
  "vendor_transaction_id": "ch_xxx",     // ID da transação do
fornecedor
  "amount": 200.00,
  "currency": "USD",
  "status": "succeeded",                 // 'succeeded', 'failed',
'refunded'
  "captured_at": "2025-12-27T10:30:00Z",
  "vendor_response": {},                 // Resposta completa do
fornecedor
  "meta": {}
}
```

WalletAccount

Carteira do cliente com rastreamento de saldo (1-para-1 com o cliente).

```
{
  "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

Registro completo de todas as transações da carteira.

```
{
  "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": "Recarga de Cartão",
  "reference_type": "payment_capture",   // Links para objeto
relacionado
  "reference_id": 103,
  "meta": {},
  "created_at": "2025-12-27T10:35:00Z"
}
```

Documentos Financeiros

Para modelagem e personalização de faturas, veja **Faturas de Clientes**.

Faturas

Definição: Uma fatura é um documento que contém uma lista de **transações de DÉBITO** (cobranças). Quando a API é chamada para criar uma fatura, um array de IDs de transações de débito deve ser fornecido. Essas transações são "vinculadas" à fatura definindo seu campo `invoice_id`.

Para detalhes de gerenciamento de transações, veja **Transações**.

Campos Chave:

```
{
  "invoice_id": 12345,
  "invoice_number": "INV-2025-000001",   // Gerado
automaticamente: INV-YYYY-NNNNNN
  "customer_id": 123,
  "title": "Fatura de Serviços Mensais",
  "paid": true,                          // Status de pagamento
  "void": false,                         // Status de anulação
  "payment_reference": "ch_xxxxx",       // Último/principal ID de
pagamento
  "payment_type": "stripe_capture",      // Último tipo de
pagamento
  "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,                 // Valor total da fatura
  "wholesale_cost": 250.00
}
```

Geração de Fatura: Os números das faturas são gerados automaticamente no formato `INV-YYYY-NNNNNN`, sequencial dentro do ano calendário, reiniciando a cada 1º de janeiro.

Pagamentos em Faturas

Como Funcionam os Pagamentos: Uma vez que uma fatura é criada a partir de transações de débito, um **PAGAMENTO** pode ser aplicado criando uma **transação de crédito** (custo de `retail_cost` negativo) vinculada à fatura.

Os pagamentos devem:

- Ser listados na fatura claramente marcados como "PAGAMENTOS"
- Mostrar a data de pagamento relevante (que pode diferir da data de criação da fatura)
- Suportar múltiplos pagamentos por fatura
- Compensar os débitos para calcular o saldo da fatura

Status da Fatura:

- **PAGA**: Pagamentos totais (créditos) são iguais ou superiores aos débitos totais
- **PARCIALMENTE PAGA**: Alguns pagamentos aplicados, mas o saldo permanece
- **PAGA EM EXCESSO**: Não é tratado atualmente - requer divisão de crédito (recurso futuro)

Contabilidade de Partida Dobrada: O sistema implementa contabilidade adequada onde cada cobrança tem um pagamento compensatório:

```
// 1. Transação de débito (cobrança)
{
  "transaction_id": 7001,
  "invoice_id": 12345,
  "retail_cost": 100.00,                // Positivo = cliente deve
  "title": "Taxa de Serviço"
}

// 2. Transação de crédito (pagamento)
{
  "transaction_id": 7002,
  "invoice_id": 12345,
  "retail_cost": -100.00,               // Negativo = pagamento recebido
  "title": "Pagamento pela Fatura: Taxa de Serviço (12345)",
  "payment_type": "stripe_capture",
  "payment_reference": "ch_xxxxx"
}

// Resultado líquido: saldo de $0 → fatura marcada como PAGA
```

Metadados Suplementares da Fatura: Além das transações de crédito, a fatura também armazena campos de resumo (`payment_reference`, `payment_type`, `payment_time`) para consultas rápidas. No entanto:

- **Método primário**: Transações de crédito vinculadas via `invoice_id` (especificação de Hayden)
- **Metadados secundários**: Campos da fatura armazenam resumo do último/principal pagamento

- **Para múltiplos pagamentos**: Consultar transações de crédito para o histórico completo

Extratos

Definição: Um extrato mostra todos os **DÉBITOS e CRÉDITOS** das transações de um cliente durante um período especificado. Este é o único tipo de documento onde tanto débitos quanto créditos são mostrados como itens de linha juntos, como um extrato bancário.

Notas de Crédito

Propósito: Aplicadas apenas a faturas com pagamentos já aplicados. Deve ser feito como parte do processo de anulação da fatura.

Processo:

1. Uma fatura é anulada (junto com suas transações de DÉBITO associadas)
2. Quaisquer PAGAMENTOS já aplicados são vinculados à NOTA DE CRÉDITO criada
3. O saldo do cliente vai para crédito pelo valor equivalente
4. O saldo da nota de crédito pode então ser:
    - Aplicado a outra fatura como um PAGAMENTO, OU
    - REEMBOLSADO ao cliente

Lógica de Reembolso: Se o reembolso for escolhido, é invocado com base nas transações de CRÉDITO associadas e como foram originalmente pagas (por exemplo, reembolso do Stripe se o tipo de transação de crédito foi Stripe).

APIs de Métodos de Pagamento

Todos os endpoints usam a URL base: `/api/payments/`

Para gerenciamento detalhado de métodos de pagamento e manuseio de cartões, veja **Métodos de Pagamento**.

Adicionar Método de Pagamento

Salve um novo método de pagamento para um cliente.

Endpoint: `POST /api/payments/methods`

Requisição:

```
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": "Meu Cartão Visa"
  }'
```

Corpo da Requisição:

```
{
  "customer_id": 123,
  "vendor": "stripe",                    // 'stripe', 'paypal' ou
qualquer fornecedor adicionado
  "payment_token": "pm_xxxxx",           // Token único do SDK
frontend
  "is_default": false,                   // Definir como método de
pagamento padrão?
  "card_nickname": "Meu Cartão Visa"     // Nome amigável opcional
}
```

Resposta (201 Criado):

```
{
  "success": true,
  "message": "Método de pagamento adicionado com sucesso",
  "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": "Meu Cartão Visa",
    "is_default": false,
    "status": "active"
  }
}
```

Obter Métodos de Pagamento

Recupere todos os métodos de pagamento para um cliente.

Endpoint: `GET /api/payments/methods?customer_id={id}`

Requisição:

```
curl -X GET "https://your-domain.com/api/payments/methods?
customer_id=123" \
  -H "Authorization: Bearer YOUR_API_KEY"
```

Resposta (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
    }
  ]
}
```


Obter Método de Pagamento Padrão

Obtenha o método de pagamento padrão para um cliente.

Endpoint: `GET /api/payments/methods/default?customer_id={id}`

Requisição:

```
curl -X GET "https://your-domain.com/api/payments/methods/default?
customer_id=123" \
  -H "Authorization: Bearer YOUR_API_KEY"
```

Resposta (200 OK):

```
{
  "success": true,
  "data": {
    "payment_method_id": 789,
    "vendor": "stripe",
    "payment_type": "card",
    "card_brand": "visa",
    "card_last4": "4242",
    "is_default": true
  }
}
```

Definir Método de Pagamento Padrão

Defina um método de pagamento como padrão.

Endpoint: `PUT /api/payments/methods/set-default`

Requisição:

```
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
  }'
```

Resposta (200 OK):

```
{
  "success": true,
  "message": "Método de pagamento padrão atualizado",
  "data": {
    "payment_method_id": 790,
    "is_default": true
  }
}
```

Deletar Método de Pagamento

Deletar um método de pagamento salvo.

Endpoint: `DELETE /api/payments/methods/{payment_method_id}?customer_id={id}`

Requisição:

```
curl -X DELETE "https://your-domain.com/api/payments/methods/789?
customer_id=123" \
  -H "Authorization: Bearer YOUR_API_KEY"
```

Resposta (200 OK):

```
{
  "success": true,
  "message": "Método de pagamento deletado com sucesso"
}
```

APIs de Fluxo de Pagamento

O sistema suporta vários fluxos de pagamento dependendo do seu caso de uso.

1. Pagamento Direto (Cobrança Simples)

Caso de Uso: Pagamento simples em uma etapa sem provisionamento de serviço.

Endpoint: `POST /api/payments/charge`

Requisição:

```
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"
    }
  }'
```

Corpo da Requisição:

```
{
  "customer_id": 123,
  "amount": 50.00,
  "currency": "USD",                    // Padrão: USD
  "payment_method_id": 789,             // Opcional - usa o
padrão se omitido
  "vendor": "stripe",                   // Obrigatório se usar
payment_token
  "payment_token": "pm_xxxxx",          // Opcional - token único
  "save_method": false,                 // Salvar método de
pagamento para uso futuro?
  "metadata": {
    "order_id": "12345"
  }
}
```

Resposta (200 OK):

```
{
  "success": true,
  "message": "Pagamento bem-sucedido",
  "data": {
    "transaction_id": "ch_xxxxx",
    "capture_id": 101,
    "amount": 50.00,
    "currency": "USD",
    "status": "succeeded"
  }
}
```


2. Pagamento de Fatura (Roteamento Primeiro pela Carteira)

Caso de Uso: Pagar uma fatura usando o saldo da carteira primeiro, depois o cartão para o restante.

Endpoint: `POST /api/payments/invoice`

Fluxo:

1. Verificar saldo da carteira
2. Usar fundos da carteira primeiro
3. Cobrar o cartão pela diferença (se houver)
4. Creditar a carteira com o valor do cartão
5. Debitar a carteira pelo valor total da fatura

Requisição:

```
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": "Pagamento da fatura"
    }
  }'
```

Resposta (200 OK):

```
{
  "success": true,
  "message": "Pagamento da fatura processado",
  "data": {
    "customer_id": 123,
    "service_amount": 200.00,
    "routing_mode": "hybrid",
    "initial_balance": 150.00,
    "wallet_portion_used": 150.00,       // A carteira cobriu isso
    "card_portion_used": 50.00,          // Cartão cobrado pela
diferença
    "charged_amount": 50.00,
    "wallet_credited": 50.00,
    "wallet_debited": 200.00,
    "final_balance": 0.00,
    "payment_method_used": true
  }
}
```

3. Autorização de Reserva (Reservar Fundos)

Caso de Uso: Reservar fundos para captura posterior (por exemplo, reservas de hotel, aluguéis).

> **Implementação de Playbook**: Para exemplos de playbook do Ansible de fluxos de pagamento em duas fases, veja Padrão de Compromisso em Duas Fases

Etapa 1: Criar Reserva de Autorização

Endpoint: `POST /api/payments/authorize/hold`

Fluxo Primeiro pela Carteira:

1. Verificar saldo da carteira
2. Calcular a diferença (valor - saldo_da_carteira)
3. Autorizar cartão apenas pela diferença
4. O débito da carteira acontece no momento da captura, não aqui

Requisição:

```
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"
    }
  }'
```

Corpo da Requisição:

```
{
  "customer_id": 123,
  "amount": 200.00,
  "currency": "USD",
  "payment_method_id": 789,              // Opcional - usa o
padrão se omitido
  "vendor": "stripe",                    // Obrigatório se usar
payment_token
  "payment_token": "pm_xxxxx",           // Opcional - token único
  "save_method": false,
  "use_wallet": true,                    // Ativar roteamento
primeiro pela carteira (padrão: true)
  "metadata": {
    "reservation_id": "RES-001"
  }
}
```

Resposta (200 OK):

```
{
  "success": true,
  "message": "Pagamento autorizado (reserva criada)",
  "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,             // A carteira cobrirá
isso
    "card_amount": 50.00,                // Cartão autorizado por
isso
    "message": "Cartão autorizado para $50 (recarga da carteira).
O débito da carteira de $200 ocorrerá na captura."
  }
}
```

Etapa 2: Capturar Autorização

Endpoint: `POST /api/payments/capture/{authorization_id}`

Fluxo de Captura Primeiro pela Carteira:

1. Capturar cartão (se autorizado) - recarrega a carteira
2. Creditar a carteira com o valor capturado do cartão
3. Debitar a carteira pelo valor total do serviço
4. Criar fatura/transações (se solicitado)

Requisição:

```
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": "Reserva de Hotel",
      "description": "Estadia de 3 noites",
      "wholesale_cost": 100.00,
      "contract_days": 3,
      "send_email": true
    }
  }'
```

Corpo da Requisição:

```
{
  "amount": 200.00,                    // Opcional - captura o
valor total se omitido
  "metadata": {
    "invoice": true,                   // Criar fatura e
transações?
    "create_transaction": true,        // Criar registro de
transação?
    "title": "Reserva de Hotel",
    "description": "Estadia de 3 noites",
    "wholesale_cost": 100.00,
    "contract_days": 3,
    "send_email": true                 // Enviar e-mail da
fatura?
  }
}
```

Resposta (200 OK):

```
{
  "success": true,
  "message": "Autorização capturada",
  "data": {
    "capture_id": 103,
    "transaction_id": "ch_xxxxx",
    "authorization_id": 301,
    "amount": 200.00,
    "currency": "USD",
    "status": "succeeded",
    "wallet_credit": {                    // Cartão recarregou a
carteira
      "ledger_id": 401,
      "amount": 50.00
    },
    "wallet_debit": {                     // Serviço cobrado da
carteira
      "ledger_id": 402,
      "amount": 200.00
    },
    "transaction": {                      // Criado se invoice=true
      "transaction_id": 7001
    },
    "invoice": {                          // Criado se invoice=true
      "invoice_id": 12345,
      "invoice_number": "INV-2025-000001"
    }
  }
}
```

Etapa 3: Liberar Autorização (Cancelar)

Endpoint: `POST /api/payments/release/{authorization_id}`

Caso de Uso: Cancelar reserva, provisionamento falhou ou cliente mudou de ideia.

Requisição:

```
curl -X POST https://your-domain.com/api/payments/release/301 \
  -H "Authorization: Bearer YOUR_API_KEY"
```

Resposta (200 OK):

```
{
  "success": true,
  "message": "Autorização liberada",
  "data": {
    "authorization_id": 301,
    "vendor_authorization_id": "auth_xxxxx",
    "status": "released",
    "released_at": "2025-12-27T10:45:00Z"
  }
}
```

Nota: Com o fluxo primeiro pela carteira, nenhum reembolso da carteira é necessário, uma vez que a carteira não é debitada até o momento da captura.

4. Pagamento de Recarregamento (Duas Fases com Provisionamento)

Caso de Uso: Processar pagamento para recarga de serviço que requer provisionamento (por exemplo, ativação de hotspot/dongle). Se o provisionamento falhar, a autorização é liberada.

Endpoint: `POST /api/payments/topup`

Fluxo: Autorizar → Provisionar serviço → Capturar

Requisição:

```
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
    }
  }'
```

Para Aluguel/Hotspot Anônimo (método de pagamento NÃO salvo):

```
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"
    }
  }'
```

Resposta (200 OK):

```
{
  "success": true,
  "message": "Pagamento de recarga processado com sucesso",
  "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        // Falso para
anônimo/aluguel
  }
}
```

5. Pagamento de Aluguel (Terceiro)

Caso de Uso: Cobrar o cartão de um cliente para pagar pelo serviço de outro cliente.

Endpoint: `POST /api/payments/rental`

Requisição:

```
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": "Pagamento de serviço de aluguel",
    "metadata": {
      "rental_agreement_id": "RA-001"
    }
  }'
```

Para Aluguel Anônimo (cartão do locatário NÃO salvo):

```
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": "Pagamento de aluguel anônimo",
    "metadata": {
      "billing_email": "renter@example.com"
    }
  }'
```

Resposta (200 OK):

```
{
  "success": true,
  "message": "Pagamento de aluguel processado",
  "data": {
    "transaction_id": "ch_xxxxx",
    "amount": 75.00,
    "payment_method_saved": false        // Anônimo: nenhum método salvo
  }
}
```

6. Reembolso de Pagamento

Endpoint: `POST /api/payments/refund`

Requisição:

```
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"
  }'
```

Corpo da Requisição:

```
{
  "transaction_id": "ch_xxxxx",          // ID da transação do fornecedor
  "vendor": "stripe",                    // 'stripe', 'paypal', etc.
  "amount": 50.00,                       // Opcional - reembolso total se omitido
  "reason": "customer_request"           // Razão de reembolso opcional
}
```

Resposta (200 OK):

```
{
  "success": true,
  "message": "Reembolso processado",
  "data": {
    "refund_id": "re_xxxxx",
    "amount": 50.00,
    "status": "succeeded"
  }
}
```

APIs de Carteira

Todos os endpoints de carteira usam a URL base: `/api/wallet/`

Obter Saldo da Carteira

Endpoint: `GET /api/wallet/balance?customer_id={id}`

Requisição:

```
curl -X GET "https://your-domain.com/api/wallet/balance?
customer_id=123" \
  -H "Authorization: Bearer YOUR_API_KEY"
```

Resposta (200 OK):

```
{
  "customer_id": 123,
  "balance": 150.50,
  "currency": "USD"
}
```

Obter Informações da Carteira

Obtenha informações completas da carteira e crédito, incluindo configurações de recarga automática.

Endpoint: `GET /api/wallet/info?customer_id={id}`

Requisição:

```
curl -X GET "https://your-domain.com/api/wallet/info?
customer_id=123" \
  -H "Authorization: Bearer YOUR_API_KEY"
```

Resposta (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
  }
}
```

Recarregar Carteira

Recarregue a carteira cobrando um método de pagamento.

Endpoint: `POST /api/wallet/topup`

Fluxo: Cobrar cartão/PayPal → Creditar carteira

Requisição:

```
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
  }'
```

Corpo da Requisição:

```
{
  "customer_id": 123,
  "amount": 100.00,
  "currency": "USD",                    // Padrão: USD
  "payment_method_id": 789              // Opcional - usa o
padrão se omitido
}
```

Resposta (200 OK):

```
{
  "success": true,
  "payment": {
    "transaction_id": "ch_xxxxx",
    "amount": 100.00
  },
  "wallet": {
    "ledger_id": 503,
    "balance_after": 250.50
  },
  "message": "Carteira recarregada com 100.00 USD"
}
```

Obter Transações da Carteira

Obtenha o histórico de transações da carteira (livro razão).

Endpoint: `GET /api/wallet/transactions?customer_id={id}&limit={n}&offset={n}&type={type}`

Requisição:

```
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"
```

Parâmetros de Consulta:

- `customer_id` (obrigatório): ID do cliente
- `limit` (opcional): Número de registros (padrão: 50)
- `offset` (opcional): Deslocamento de paginação (padrão: 0)
- `type` (opcional): Filtrar por tipo de transação ('credit', 'debit', 'refund', 'adjustment')

Resposta (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": "Recarga de Cartão",
      "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": "Taxa de Serviço",
      "reference_type": "service_charge",
      "reference_id": 789,
      "created_at": "2025-12-27T11:00:00Z"
    }
  ]
}
```

Resumo da Referência da API

Endpoints de Métodos de Pagamento

Método	Endpoint	Descrição
POST	`/api/payments/methods`	Adicionar método de pagamento
GET	`/api/payments/methods?customer_id={id}`	Obter todos os métodos de pagamento
GET	`/api/payments/methods/default?customer_id={id}`	Obter método de pagamento padrão
PUT	`/api/payments/methods/set-default`	Definir método de pagamento padrão
DELETE	`/api/payments/methods/{id}?customer_id={id}`	Deletar método de pagamento

Endpoints de Fluxo de Pagamento

Método	Endpoint	Descrição
POST	`/api/payments/charge`	Pagamento direto (uma etapa)
POST	`/api/payments/invoice`	Pagamento de fatura (primeiro pela carteira)
POST	`/api/payments/topup`	Recarregar com provisionamento
POST	`/api/payments/authorize/hold`	Criar reserva de autorização
POST	`/api/payments/capture/{id}`	Capturar autorização
POST	`/api/payments/release/{id}`	Liberar autorização
POST	`/api/payments/rental`	Pagamento de aluguel/terceiro
POST	`/api/payments/refund`	Reembolso de pagamento

Endpoints de Carteira

Método	Endpoint	Descrição
GET	`/api/wallet/balance?customer_id={id}`	Obter saldo da carteira
GET	`/api/wallet/info?customer_id={id}`	Obter informações da carteira + configurações
POST	`/api/wallet/topup`	Recarregar carteira via método de pagamento
GET	`/api/wallet/transactions?customer_id={id}`	Obter histórico de transações

Endpoints Específicos do PayPal

Método	Endpoint	Descrição
POST	`/api/payments/paypal/vault/setup-token`	Criar token de configuração do PayPal para o SDK de Campos de Cartão
POST	`/api/payments/paypal/vault/finalize`	Finalizar o cofre do PayPal e salvar o método de pagamento
POST	`/api/payments/paypal/vault/update-setup-token`	Atualizar token de configuração para tratamento de 3DS/SCA

Casos de Uso Comuns

Caso de Uso 1: Cobrar Cliente por Serviço (Pré-pago)

Cenário: Cliente pagando por 30 dias de serviço usando roteamento primeiro pela carteira.

```
# Pagamento de fatura (primeiro pela carteira, cartão pela
diferença)
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": "Serviço Premium de 30 dias"
    }
  }'
```

Resposta:

```
{
  "success": true,
  "message": "Pagamento da fatura processado",
  "data": {
    "service_amount": 99.99,
    "wallet_portion_used": 50.00,        // A carteira tinha $50
    "card_portion_used": 49.99,          // Cartão cobrado $49.99
    "final_balance": 0.00
  }
}
```

Caso de Uso 2: Pagamento em Duas Fases com Provisionamento de Serviço

Cenário: Provisionar um serviço apenas após o pagamento ser autorizado. Se o provisionamento falhar, nenhuma cobrança ocorre.

```
# Pagamento de recarga com provisionamento automático
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"
    }
  }'
```

Fluxo:

1. Autorizar $149.99 no cartão
2. Provisionar serviço de internet fibra
3. Se o provisionamento for bem-sucedido → capturar pagamento
4. Se o provisionamento falhar → liberar autorização (sem cobrança)

Caso de Uso 3: Pagamento Anônimo de Hotspot

Cenário: Usuário anônimo paga pelo acesso ao WiFi hotspot. Sem registro de cliente, sem método de pagamento salvo.

```
# Pagamento anônimo de 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"
    }
  }'
```

Resultado:

- Pagamento processado
- Hotspot ativado
- Método de pagamento NÃO salvo
- Nenhum registro de cliente criado

Caso de Uso 4: Reserva de Hotel (Reserva + Captura)

Cenário: Reservar fundos para hotel, capturar na entrada, liberar se cancelado.

```
# Etapa 1: Check-in - reservar fundos
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"
    }
  }'

# Resposta: {"authorization_id": 301, "status": "authorized"}

# Etapa 2a: Cliente faz check-in - capturar
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": "Estadia no Hotel - 3 Noites",
      "description": "Quarto 302, 27-30 de Dezembro",
      "send_email": true
    }
  }'

# Etapa 2b: Cliente cancela - liberar
curl -X POST https://your-domain.com/api/payments/release/301 \
  -H "Authorization: Bearer YOUR_API_KEY"
```

Caso de Uso 5: Adicionar Método de Pagamento (Stripe)

Cenário: Cliente adiciona um novo cartão Visa à sua conta.

```
# Etapa 1: Frontend cria token do Stripe via Stripe.js
# const {token} = await stripe.createToken(cardElement);

# Etapa 2: Backend salva método de pagamento
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 do Trabalho"
  }'
```

Caso de Uso 6: Recarregar Carteira

Cenário: Cliente recarrega sua carteira com $100 usando método de pagamento salvo.

```
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
  }'
```

Resposta:

```
{
  "success": true,
  "payment": {
    "transaction_id": "ch_xxxxx",
    "amount": 100.00
  },
  "wallet": {
    "balance_after": 250.50
  },
  "message": "Carteira recarregada com 100.00 USD"
}
```

Tratamento de Erros

Respostas de Erro Comuns

Fundos Insuficientes (400 Bad Request):

```
{
  "error": "Saldo da carteira insuficiente. Disponível: 50.00,
Necessário: 200.00"
}
```

Pagamento Falhou (400 Bad Request):

```
{
  "error": "Pagamento falhou. Por favor, tente novamente."
}
```

Erro de Validação (400 Bad Request):

```
{
  "error": "customer_id e amount são obrigatórios"
}
```

Não Encontrado (404 Not Found):

```
{
  "error": "Autorização 999 não encontrada"
}
```

Erro de Servidor (500 Internal Server Error):

```
{
  "error": "Ocorreu um erro ao processar seu pagamento"
}
```

Reembolsos e Roteamento Primeiro pela Carteira

Opções de Reembolso

O sistema suporta **dois tipos de reembolsos** dependendo das necessidades do seu negócio:

1. Reembolso para Fonte de Pagamento (Stripe/PayPal)

Caso de Uso: Cliente solicita reembolso total por serviço cancelado ou produto defeituoso.

Fluxo: O dinheiro volta para o método de pagamento original (cartão, conta do PayPal, etc.)

Endpoint: `POST /api/payments/refund`

Exemplo:

```
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"
  }'
```

Resultado:

- Stripe/PayPal processa o reembolso para o método de pagamento original
- O cliente vê o crédito em seu cartão/conta do PayPal dentro de 5-10 dias úteis
- O dinheiro NÃO vai para a carteira
- Rastro de auditoria completo mantido na tabela `PaymentCapture`

Quando Usar:

- □ Cliente cancelou o pedido
- □ Serviço não entregue
- □ Erro de cobrança
- □ Cliente solicita explicitamente reembolso para o cartão

2. Crédito para a Carteira

Caso de Uso: Reembolso parcial, crédito de serviço ou manter fundos para futuras compras.

Fluxo: O dinheiro é creditado no saldo da carteira do cliente para uso imediato.

Nota: Créditos de carteira são normalmente tratados internamente pelo sistema durante cenários de erro. Para créditos de carteira manuais, entre em contato com o suporte ou use ferramentas administrativas.

Resultado:

- Fundos imediatamente disponíveis na carteira

- Sem período de espera
- Podem ser usados para futuras compras
- Rastro de auditoria completo na tabela `WalletLedger`

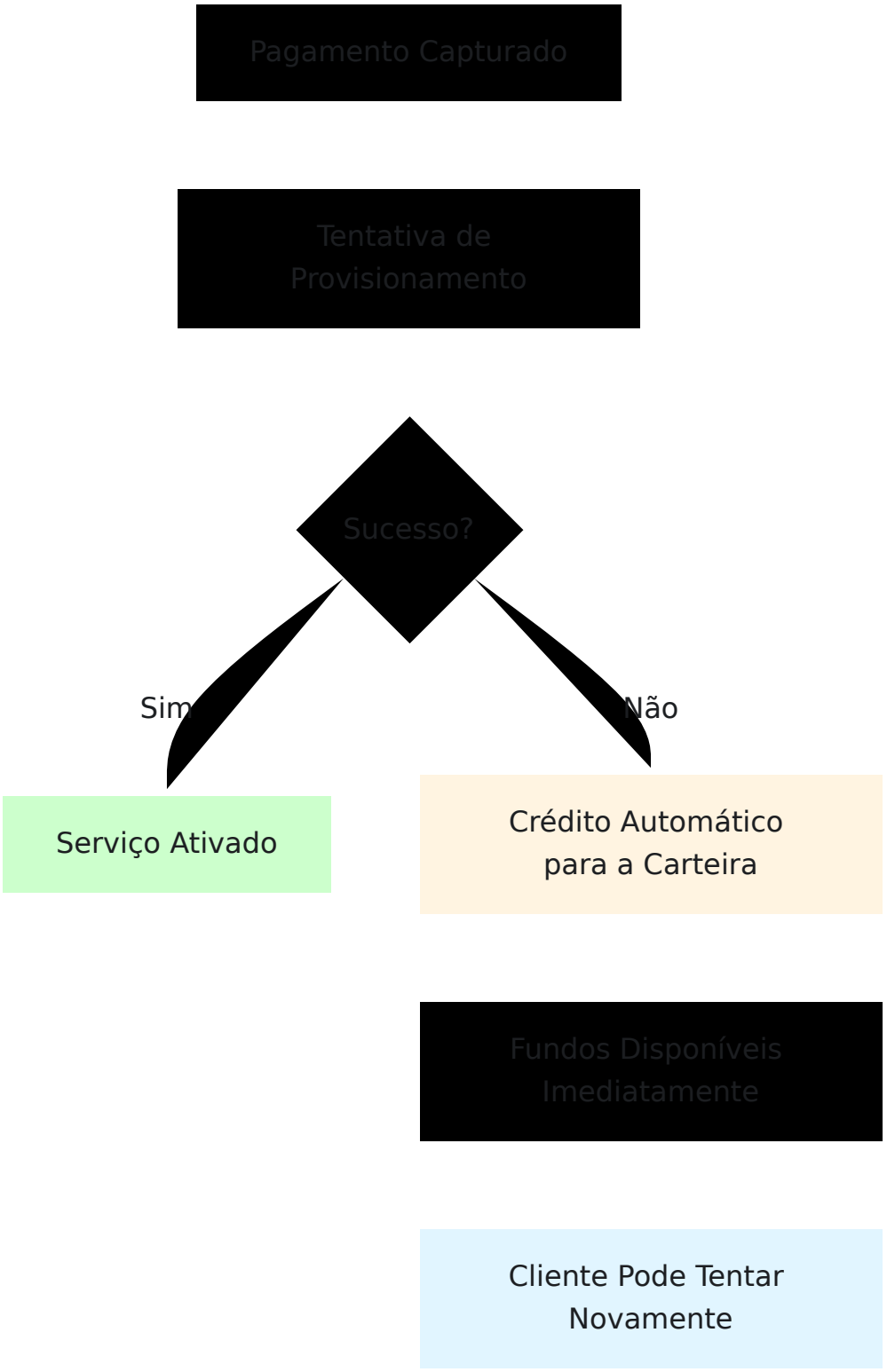
Quando Usar:

- □ Créditos de serviço (por exemplo, compensação por inatividade)
- □ Reembolsos parciais onde o cliente irá recomprar
- □ Compensação por erro (provisionamento falhou)
- □ Créditos promocionais

Estratégia de Reembolso Híbrida

Melhor Prática: Para cenários de erro durante o fluxo de pagamento, o sistema credita automaticamente a carteira em vez de reembolsar o cartão.

Recuperação Automática de Erros:



Cenário de Exemplo:

1. Cliente paga $100 pelo serviço
2. Cartão cobrado com sucesso
3. Provisionamento falha

4. Em vez de reembolsar $100 para o cartão (5-10 dias + taxas de reembolso):
    - Creditar $100 na carteira imediatamente
    - Cliente pode tentar a compra novamente instantaneamente
    - Sem taxas de reembolso
    - Melhor experiência do usuário

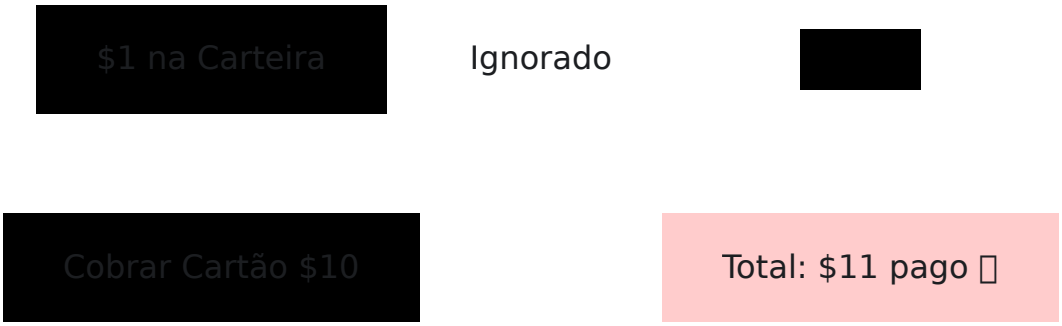
Roteamento Primeiro pela Carteira: Cobranças de Cartão Otimizadas

O sistema **cobre apenas o cartão pela diferença**, não pelo valor total, quando você tem saldo na carteira.

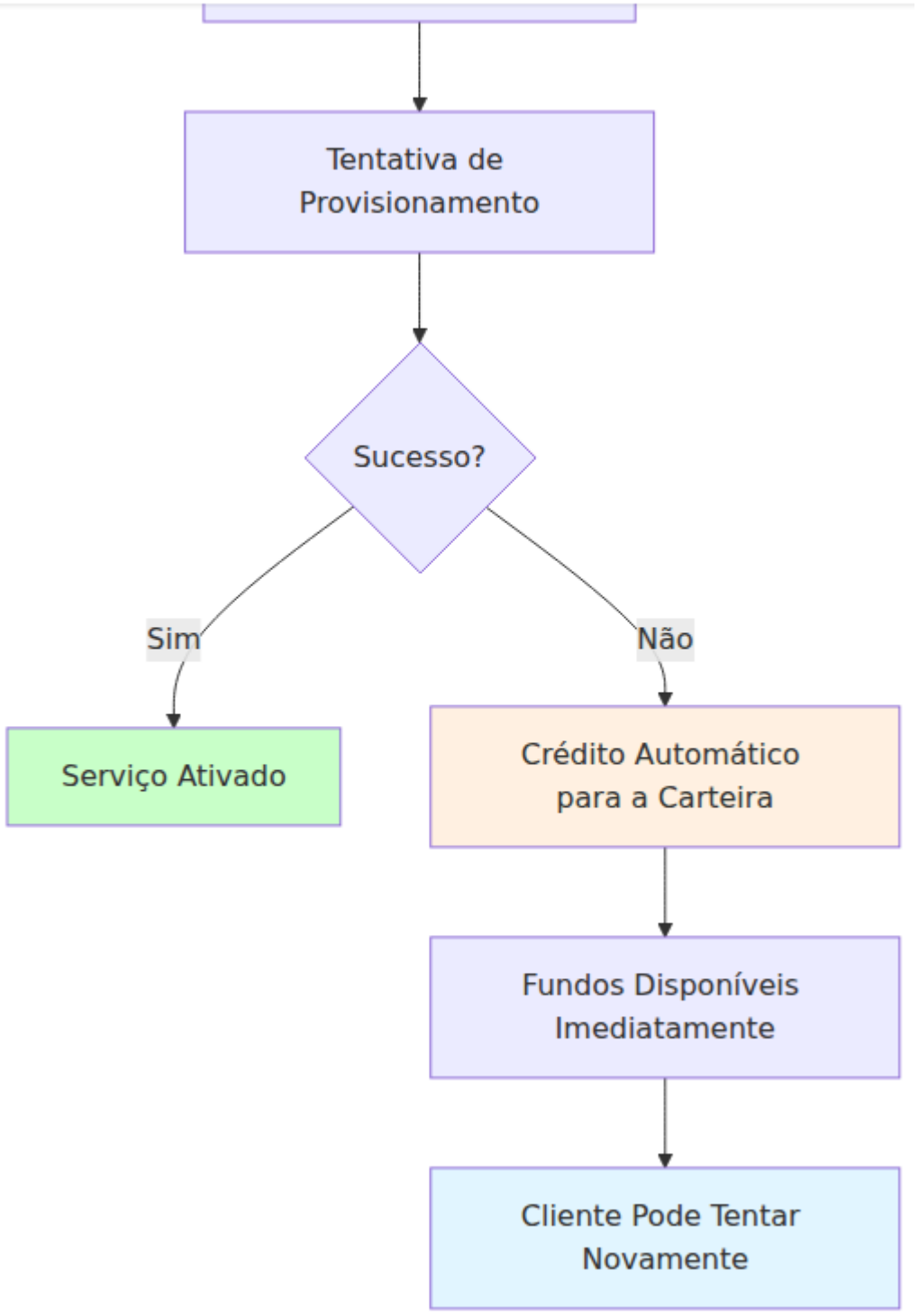
Exemplo 1: Saldo de $1 na Carteira + Compra de $10

Cenário: Você tem $1 na sua carteira e deseja comprar um complemento de $10.

Fluxo de Pagamento Tradicional (NÃO é assim que este sistema funciona):



Fluxo Primeiro pela Carteira do OmniCRM:



Requisição da 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
    }
  }'
```

Resposta da API:

```
{
  "success": true,
  "message": "Pagamento da fatura processado",
  "data": {
    "customer_id": 123,
    "service_amount": 10.00,
    "routing_mode": "hybrid",
    "initial_balance": 1.00,
    "wallet_portion_used": 1.00,     // Usou os $1 existentes
    "card_portion_used": 9.00,       // Apenas cobrado $9 pela
diferença
    "charged_amount": 9.00,          // ← Cobrança do Cartão
    "wallet_credited": 9.00,         // ← Recarregou a carteira
    "wallet_debited": 10.00,         // ← Cobrança de Serviço
    "final_balance": 0.00
  }
}
```

Exemplo 2: Saldo de $50 na Carteira + Compra de $30

Cenário: Você tem $50 na carteira e o custo da compra é $30.

Fluxo Primeiro pela Carteira:

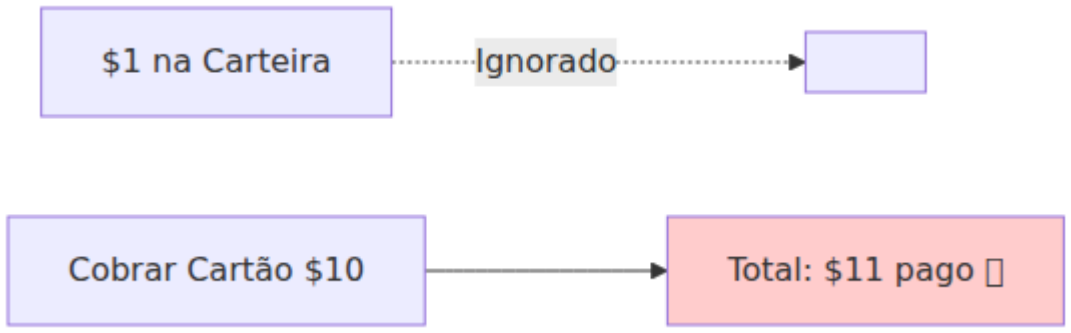
Resposta da API:

```
{
  "success": true,
  "data": {
    "service_amount": 30.00,
    "initial_balance": 50.00,
    "charged_amount": 0,              // ← Nenhuma cobrança do
cartão
    "wallet_debited": 30.00,
    "final_balance": 20.00,
    "payment_method_used": false      // ← Cartão não usado
  }
}
```

Exemplo 3: Reserva de Autorização com Saldo na Carteira

Cenário: Hotel reserva $500 de autorização com saldo de $150 na carteira.

Etapa 1: Autorização (`POST /api/payments/authorize/hold`):



Resposta da API:

```
{
  "authorization_id": 301,
  "amount": 500.00,
  "status": "authorized",
  "wallet_balance": 150.00,
  "wallet_to_use": 150.00,            // A carteira cobrirá isso
  "card_amount": 350.00,
  "message": "Cartão autorizado para $350 (recarga da carteira). O
débito da carteira de $500 ocorrerá na captura."
}
```

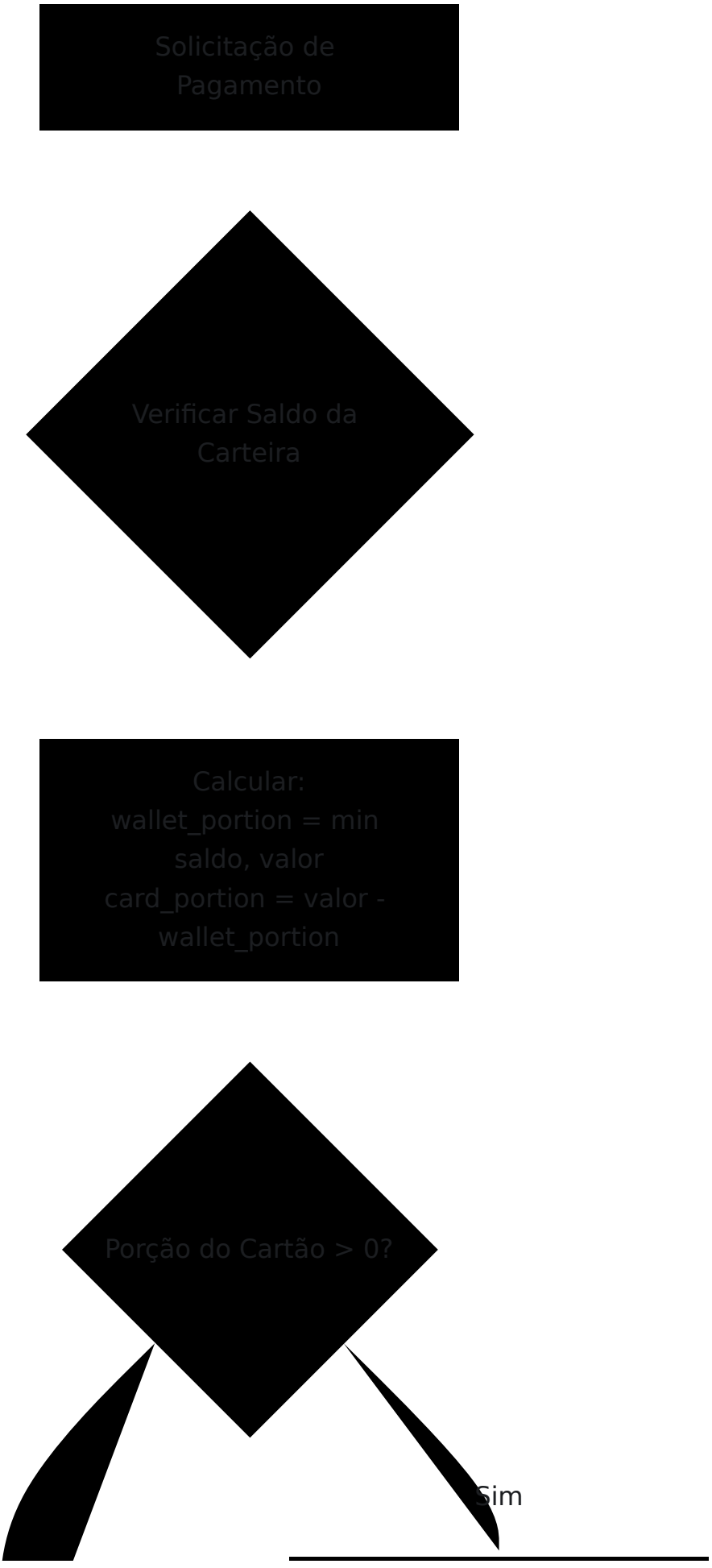
Etapa 2: Captura (`POST /api/payments/capture/301`):

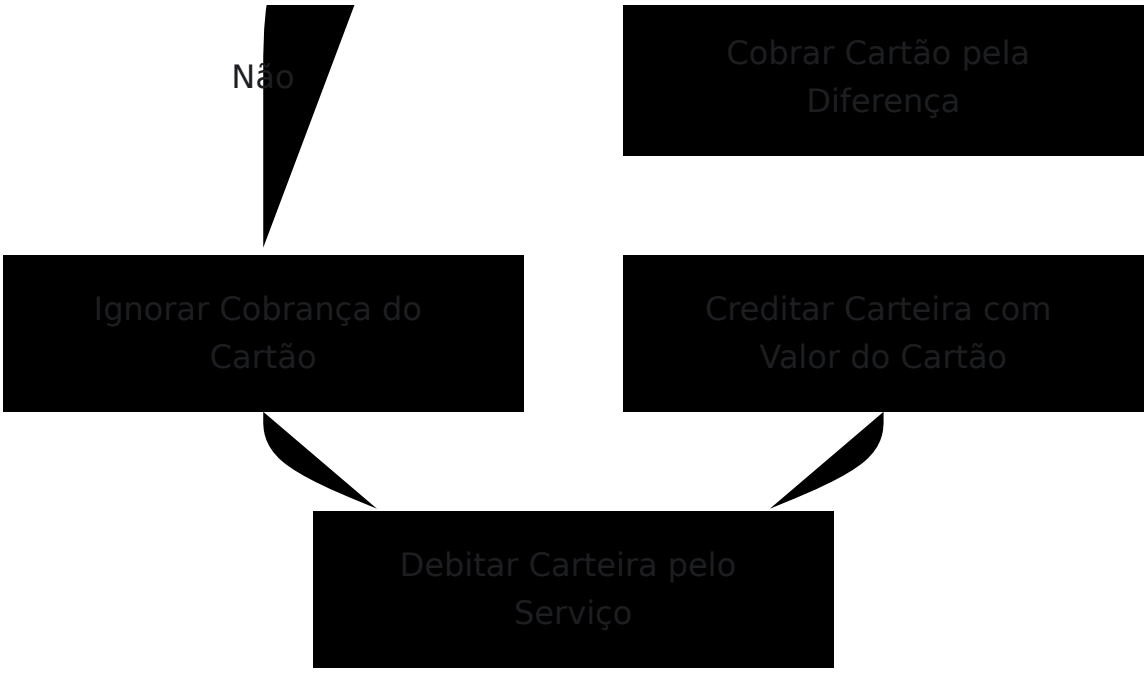
Por que isso importa:

- □ Cliente tem apenas $350 "reservados" em seu cartão, não $500
- □ Reduz o impacto no crédito disponível do cliente
- □ Valores de autorização mais precisos
- □ Melhor experiência do cliente

Detalhes de Implementação

Lógica de Roteamento Primeiro pela Carteira:


Solicitação de Pagamento
Verificar Saldo da Carteira
Calcular:
wallet_portion = min saldo, valor
card_portion = valor - wallet_portion
Porção do Cartão > 0?
Sim



Completo

Como Funciona:

1. O sistema verifica o saldo atual da carteira
2. Calcula a diferença: `valor - saldo_da_carteira`
3. Se a diferença > 0, cobra o cartão apenas pela diferença
4. Credita a carteira com o pagamento do cartão
5. Debita a carteira pelo valor total do serviço
6. Resultado: Cliente cobrado apenas pelo que a carteira não conseguiu cobrir

Substituição do Modo de Roteamento

Você pode substituir o comportamento primeiro pela carteira, se necessário:

Modo de Bypass - Sempre cobrar o cartão, mesmo que a carteira tenha fundos:

```
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"
    }
  }'
```

Resultado: Cartão cobrado $100, carteira creditada $100, carteira debitada $100 (líquido: $0)

Caso de Uso: Cliente deseja usar o cartão para recompensas/pontos, mesmo com saldo na carteira.

Melhores Práticas

1. **Sempre use o roteamento primeiro pela carteira** para pagamentos de clientes, a fim de habilitar a funcionalidade pré-paga
2. **Use pagamentos em duas fases** (endpoint de recarga) ao provisionar serviços para evitar cobranças se o provisionamento falhar
3. **Defina metadados** para todos os pagamentos para manter o rastro de auditoria
4. **Use pagamentos anônimos** apenas para usuários verdadeiramente anônimos (hotspots, aluguéis)
5. **Trate erros de forma elegante** e forneça feedback claro ao usuário
6. **Teste fluxos de pagamento** em cenários de sucesso e falha
7. **Monitore expirações de autorização** (normalmente 7 dias para cartões)
8. **Implemente recarga automática** para serviços recorrentes para melhorar a experiência do cliente

Melhores Práticas Específicas para Playbook

Ao implementar fluxos de pagamento em playbooks do Ansible:

1. **Sempre use o padrão de bloco/resgate** - Envolva o provisionamento em try/catch para rollback automático
2. **Armazene authorization_id** - Salve para operações de captura/liberação
3. **Valide as respostas da API** - Afirme o sucesso antes de prosseguir
4. **Arredonde valores de moeda** - Sempre use 2 casas decimais
5. **Verifique métodos de pagamento** - Verifique se o cliente tem método de pagamento padrão antes da autorização

Veja Melhores Práticas de Playbook para detalhes completos e exemplos.

Autenticação

Todos os endpoints da API requerem autenticação via chave da API no cabeçalho `Authorization`:

```
Authorization: Bearer YOUR_API_KEY
```

Entre em contato com seu administrador de sistema para o provisionamento da chave da API.

Suporte ao Fornecedor

Atualmente Suportado

- **Stripe** - Suporte total (cartões, ACH)
- **PayPal** - Suporte total (contas do PayPal, cartões via SDK de Campos de Cartão)

Adicionando Novos Fornecedores

A arquitetura modular torna trivial adicionar novos fornecedores de pagamento. Veja a seção Arquitetura Modular para detalhes.

Documentação Relacionada

Guias de Implementação

- **Cobranças e Pagamentos a partir de Playbooks** - Implementação de playbook do Ansible
  - Padrão de Compromisso em Duas Fases
  - Exemplos de Autorização/Captura
  - Criação Direta de Transação
  - Cálculos Pro-Rata

Guias Específicos de Recursos

- **Faturas de Clientes** - Geração, modelagem e personalização de faturas
- **Métodos de Pagamento** - Gerenciando métodos de pagamento de clientes, cartões e contas do PayPal
- **Transações** - Gerenciamento de transações, cobranças manuais e créditos
- **Processamento de Pagamento** - Fluxos e processamento de pagamento
- **Fundamentos de Cobrança** - Fundamentos de cobrança de clientes

Navegação Rápida

- **Adicionando fornecedores?** → Veja Arquitetura Modular
- **Configurando Stripe/PayPal?** → Veja Configuração do Fornecedor
- **Personalizando faturas?** → Veja Faturas de Clientes
- **Provisionando serviços?** → Veja Integração com Playbook

Faturas de Clientes

> **Referência da API**: Para APIs e fluxos de trabalho de pagamento de faturas, consulte o **Guia da API do Sistema de Pagamento**

`Transactions </payments_transaction>` são agrupadas para formar uma fatura, que é enviada ao cliente para pagamento.

As faturas têm uma data de início e uma data de término, que é o período que a fatura cobre, e uma data de vencimento, que é a data em que a fatura deve ser paga.

As faturas podem ser geradas automaticamente pelo sistema, por exemplo, quando um serviço é cobrado, uma fatura é criada para o custo de varejo, ou podem ser criadas manualmente, por exemplo, se um cliente solicitar uma cópia de uma fatura, ou se um cliente for cobrado por uma taxa única.

As faturas de clientes são totalmente modeladas com `Mailjet <integrations_mailjet>` e podem ser personalizadas para incluir o logotipo da empresa, endereço e detalhes de pagamento, e podem ser enviadas ao cliente por e-mail, ou baixadas como um PDF.

Personalizando Modelos de Fatura

OmniCRM usa modelos HTML com Jinja2 para gerar faturas. Você pode personalizar completamente o design da fatura, a marca, as cores e o layout.

Localização do Modelo de Fatura

Os modelos de fatura são armazenados em `OmniCRM-API/invoice_templates/`

Modelos Padrão:

- `norfone_invoice_template.html` - Modelo de fatura de exemplo
- `cifi_invoice_template.html` - Exemplo de modelo alternativo

Configuração:

O modelo de fatura ativo é especificado em `OmniCRM-API/crm_config.yaml`:

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

Variáveis de Modelo Disponíveis

Os modelos de fatura têm acesso às seguintes variáveis Jinja2:

Informações da Fatura:

- `{{ invoice_number }}` - ID único da fatura (por exemplo, `INV-2025-001234`)
- `{{ date }}` - Data de emissão da fatura (formato ISO: `2025-01-10T12:00:00`)
- `{{ due_date }}` - Data de vencimento do pagamento (por exemplo, `2025-02-10`)
- `{{ start_date }}` - Data de início do período de cobrança
- `{{ end_date }}` - Data de término do período de cobrança
- `{{ total_amount }}` - Valor total da fatura antes do imposto (numérico)
- `{{ total_tax }}` - Valor total do imposto calculado a partir de todas as transações (numérico)

Informações do Cliente:

- `{{ client.name }}` - Nome completo do cliente ou nome da empresa
- `{{ client.address.address_line_1 }}` - Linha de endereço 1
- `{{ client.address.address_line_2 }}` - Linha de endereço 2
- `{{ client.address.city }}` - Cidade
- `{{ client.address.state }}` - Estado/província
- `{{ client.address.zip_code }}` - Código postal/ZIP
- `{{ client.address.country }}` - País

Itens de Linha de Transação:

Percorra as transações usando:

```
{% 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 %}
```

Campos de Transação:

- `sub_transaction.transaction_id` - ID da transação
- `sub_transaction.created` - Data/hora da transação
- `sub_transaction.title` - Título da transação
- `sub_transaction.description` - Descrição detalhada
- `sub_transaction.retail_cost` - Valor do item
- `sub_transaction.tax_percentage` - Percentual de imposto aplicado (por exemplo, 10 para 10%)
- `sub_transaction.tax_amount` - Valor do imposto calculado em dólares

Exibindo Imposto em Modelos:

```
<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>
```

Criando um Modelo de Fatura Personalizado

Passo 1: Copiar Modelo Existente

```
cd OmniCRM-API/invoice_templates/
cp norfone_invoice_template.html
your_company_invoice_template.html
```

Passo 2: Personalizar HTML/CSS

Edite `your_company_invoice_template.html` para corresponder à sua marca:

Áreas de Personalização Chave:

1. **Logotipo e Marca da Empresa**

```
<!-- Substitua pela URL do seu logotipo -->
![Sua Empresa](https://yourcompany.com/logo.png)

<!-- Atualize o nome da empresa -->
<h1>Nome da Sua Empresa</h1>
```

2. **Esquema de Cores**

```
<style>
    /* Cor da marca principal */
    .navbar {
        background: linear-gradient(to bottom right, #your-
color-1, #your-color-2);
    }

    /* Cabeçalhos da tabela */
    .table thead th {
        background-color: #your-brand-color !important;
        color: white !important;
    }

    /* Botões e links */
    .btn-primary {
        background-color: #your-brand-color;
    }
</style>
```

3. **Informações da Empresa no Rodapé**

```
<footer>
    <p>Nome da Sua Empresa</p>
    <p>123 Rua Comercial, Cidade, País</p>
    <p>Telefone: +1-555-123-4567 | Email:
billing@yourcompany.com</p>
    <p>ABN/ID Fiscal: 12345678900</p>
</footer>
```

4. **Instruções de Pagamento**

```
<div class="payment-info">
    <h3>Métodos de Pagamento</h3>
    <p><strong>Online:</strong> Pague em
https://yourcompany.com/pay</p>
    <p><strong>Transferência Bancária:</strong></p>
    <ul>
        <li>Nome da Conta: Sua Empresa Ltd</li>
        <li>BSB: 123-456</li>
        <li>Número da Conta: 987654321</li>
        <li>Referência: {{ invoice_number }}</li>
    </ul>
</div>
```

5. **Termos e Condições**

```
<div class="terms">
    <h4>Termos de Pagamento</h4>
    <p>Pagamento devido dentro de 30 dias a partir da data da
fatura.</p>
    <p>Taxas de pagamento atrasado: 2% ao mês sobre saldos em
atraso.</p>
    <p>Para consultas de cobrança: billing@yourcompany.com</p>
</div>
```

Passo 3: Atualizar Configuração

Edite `OmniCRM-API/crm_config.yaml`:

```
invoice:
  template_filename: 'your_company_invoice_template.html'
```

Passo 4: Reiniciar API

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

Passo 5: Testar Geração de Fatura

1. Navegue até um cliente com transações
2. Gere uma fatura de teste
3. Baixe o PDF para verificar a formatação
4. Envie a fatura para você mesmo para testar a entrega por e-mail

Personalização Avançada

Conteúdo Condicional:

Use condicionais Jinja2 para mostrar/ocultar conteúdo:

```
{% if total_amount > 1000 %}
    <div class="high-value-notice">
        <p><strong>Nota:</strong> Saldo alto - Plano de pagamento
disponível mediante solicitação.</p>
    </div>
{% endif %}

{% if client.address.country == "Australia" %}
    <p>GST Incluído: ${{ "%.2f"|format(total_amount * 0.10) }}</p>
{% endif %}
```

Suporte a Múltiplos Idiomas:

Crie modelos específicos para cada idioma:

```
invoice_template_en.html
invoice_template_es.html
invoice_template_fr.html
```

Configure com base na preferência de idioma do cliente.

Cálculos Personalizados:

```
<!-- Exibir subtotal e detalhamento de impostos -->
<tr>
    <td colspan="4" class="text-right"><strong>Subtotal:</strong>
</td>
    <td>${{ "%.2f"|format(total_amount) }}</td>
</tr>
<tr>
    <td colspan="4" class="text-right"><strong>Imposto:</strong>
</td>
    <td>${{ "%.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>
```

Nota: A variável `total_tax` é calculada automaticamente somando o `tax_amount` de todas as transações na fatura. O imposto de cada transação é calculado com base no campo `tax_percentage`, que por padrão é o `tax_percentage` do produto ou 0% se não especificado.

Código QR para Pagamento:

Gere um código QR para pagamento móvel:

```
<div class="qr-payment">
    ![Escanear para Pagar](https://api.qrserver.com/v1/create-qr-
code/?size=150x150&data={{ payment_url }})
    <p>Escaneie com seu telefone para pagar instantaneamente</p>
</div>
```

Melhores Práticas de Estilização de PDF

OmniCRM usa **WeasyPrint** para converter HTML em PDF. Siga estas diretrizes:

CSS Suportado:

- A maioria das propriedades CSS 2.1
- CSS3 limitado (flexbox, algumas transformações)
- Fontes da web via `@font-face`

Não Suportado:

- JavaScript
- CSS Grid (use tabelas em vez disso)
- Animações complexas
- Algumas propriedades CSS modernas

Tamanho da Página e Margens:

```
@page {
    size: A4;
    margin: 1cm;
}

body {
    font-family: Arial, sans-serif;
    font-size: 10pt;
}
```

Estilização Específica para Impressão:

```
@media print {
    .no-print {
        display: none;
    }

    .page-break {
        page-break-after: always;
    }
}
```

Layout da Tabela:

```
.table {
    table-layout: fixed;
    width: 100%;
}

.table th, .table td {
    word-wrap: break-word;
    padding: 4px;
}
```

Incorporação de Fonte:

Para fontes personalizadas, use fontes seguras para a web ou incorpore:

```
@font-face {
    font-family: 'YourFont';
    src: url('https://yourcompany.com/fonts/yourfont.woff2')
format('woff2');
}

body {
    font-family: 'YourFont', Arial, sans-serif;
}
```

Testando Modelos de Fatura

Lista de Verificação de Teste:

1. **Inspeção Visual:**
   - O logotipo é exibido corretamente
   - As cores correspondem às diretrizes da marca
   - O texto é legível (não muito pequeno)
   - As tabelas estão alinhadas corretamente
   - Todas as seções estão presentes
2. **Precisão dos Dados:**
   - Detalhes do cliente corretos
   - Valores das transações somam corretamente
   - Datas formatadas corretamente
   - Todas as variáveis substituídas corretamente
3. **Qualidade do PDF:**
   - Tamanho do arquivo razoável (<5MB)
   - Imagens nítidas e claras
   - Sem corte ou transbordamento de texto
   - Quebras de página em locais apropriados
4. **Faturas de Múltiplas Páginas:**
   - Cabeçalhos se repetem em cada página
   - Números de página exibidos
   - Listas de transações longas paginam corretamente
5. **Entrega por E-mail:**
   - PDF anexado ao e-mail
   - Tamanho do arquivo abaixo do limite do Mailjet (15MB)
   - Renderiza no Gmail, Outlook, Apple Mail

Comando de Teste (Geração Manual):

Você pode testar a geração de faturas via API:

```
curl -X GET "http://localhost:5000/crm/invoice/{invoice_id}/pdf" \
     -H "Authorization: Bearer YOUR_TOKEN" \
     --output test_invoice.pdf
```

Problemas Comuns de Modelo

Variáveis não substituindo:

- **Causa:** Erro de digitação no nome da variável ou dados ausentes
- **Correção:** Verifique a ortografia exatamente (sensível a maiúsculas e minúsculas), verifique se os dados existem no banco de dados

Estilização do PDF quebrada:

- **Causa:** Propriedade CSS não suportada
- **Correção:** Use propriedades CSS 2.1, teste com CSS compatível com WeasyPrint

Imagens não aparecem:

- **Causa:** URLs relativas ou recursos externos bloqueados
- **Correção:** Use URLs HTTPS absolutas, certifique-se de que as imagens sejam acessíveis publicamente

Tabelas transbordando a página:

- **Causa:** Larguras de coluna fixas muito largas
- **Correção:** Use larguras percentuais, `table-layout: fixed`

Fontes não renderizando:

- **Causa:** Fonte não incorporada ou indisponível
- **Correção:** Use fontes seguras para a web (Arial, Times New Roman, etc.) ou incorpore corretamente fontes personalizadas

Falha na geração do PDF:

- **Causa:** Erros de sintaxe HTML ou falha do WeasyPrint
- **Correção:** Valide o HTML, verifique os logs do WeasyPrint, simplifique layouts complexos

Cache de PDF de Fatura

Para melhorar o desempenho e reduzir a geração redundante de PDFs, o OmniCRM inclui um sistema de cache de PDF de fatura. Quando um PDF de fatura é gerado pela primeira vez, ele é armazenado em cache no banco de dados para solicitações subsequentes.

Como Funciona o Cache de PDF:

1. **Primeira Solicitação** - Quando um PDF de fatura é solicitado (download ou e-mail), o sistema:
   - Gera o PDF a partir do modelo de fatura
   - Codifica o PDF como Base64
   - Calcula um hash SHA256 do conteúdo do PDF
   - Armazena na tabela `Invoice_PDF_Cache` com:
     - Referência do ID da fatura
     - Dados do PDF (codificados em Base64)
     - Nome do arquivo
     - Hash do conteúdo (para verificação de integridade)
     - Timestamp de criação
2. **Solicitações Subsequentes** - Quando a mesma fatura é solicitada novamente:
   - O sistema verifica se há PDF em cache pelo invoice_id
   - Se o cache existir e for válido, retorna o PDF em cache imediatamente
   - Atualiza o timestamp `last_accessed` para rastrear o uso do cache
3. **Invalidar Cache** - PDFs em cache são invalidados quando:
   - A fatura é modificada (transações adicionadas/removidas, detalhes alterados)
   - O modelo de fatura é atualizado
   - A limpeza manual do cache é acionada

Benefícios:

- **Desempenho** - Entrega instantânea de PDF para solicitações repetidas (sem atraso na regeneração)

- **Consistência** - Mesmo PDF para todos os downloads de uma fatura (a menos que a fatura seja modificada)
- **Carga do Servidor** - Reduz o uso da CPU na geração de PDFs
- **Experiência do Usuário** - Indicador de carregamento aparece durante a geração inicial, solicitações subsequentes são instantâneas

Gerenciamento de Cache:

O Cache de PDF de Fatura é gerenciado automaticamente pelo sistema. Entradas de cache antigas ou não utilizadas podem ser removidas periodicamente com base em:

- Idade (por exemplo, remover entradas de cache com mais de 90 dias)
- Padrões de acesso (remover entradas não acessadas em 30 dias)
- Limites de armazenamento (implementar limites de tamanho de cache, se necessário)

Comportamento da API:

Ao baixar uma fatura via API ou UI:

- Primeira solicitação: Mostra indicador de carregamento enquanto o PDF é gerado, então armazena em cache
- Solicitações subsequentes: Download imediato do cache
- O acerto/falha de cache é transparente para o usuário

Importante: Quando você atualiza seu modelo de fatura, limpe o cache para garantir que novas faturas usem o design atualizado:

```
-- Limpar todos os PDFs de fatura em cache (executar no MySQL)
DELETE FROM Invoice_PDF_Cache;
```

Ou atualize `crm_config.yaml` para invalidar automaticamente o cache na mudança de modelo.

Acessando Faturas

As faturas podem ser visualizadas no nível do sistema ou por cliente:

Visualização por Cliente:

1. Navegue até **Clientes → [Selecionar Cliente]**
2. Clique na aba **Cobrança**
3. Veja a lista de faturas no terceiro cartão

Visualização em Todo o Sistema:

1. Navegue até **Cobrança → Faturas** (a partir do menu principal)
2. Veja todas as faturas de todos os clientes

Widgets de Estatísticas de Faturas

Na parte superior da página de faturas, quatro cartões de estatísticas exibem resumos financeiros.

Descrições dos Widgets:

- **Total de Faturas** - Soma de todos os custos de varejo de faturas (de todos os tempos) e contagem de faturas enviadas

- **Faturas Não Pagas** - Soma de faturas ainda não pagas e contagem de faturas não pagas
- **Faturas Este Mês** - Soma de faturas criadas neste mês do calendário com contagem
- **Faturas Mês Passado** - Soma de faturas criadas no mês passado do calendário com contagem

Formatação de Valores:

- Valores acima de 1.000: Exibir como sufixo "k" (por exemplo, $1.5k)
- Valores acima de 1.000.000: Exibir como sufixo "M" (por exemplo, $2.3M)
- Valores acima de 1.000.000.000: Exibir como sufixo "B" (por exemplo, $1.1B)

Indicadores de Tendência:

- Widgets para "Este Mês" e "Mês Passado" mostram mudança percentual
- Seta verde para cima: Aumento em relação ao período anterior
- Seta vermelha para baixo: Diminuição em relação ao período anterior
- Seta cinza para a direita: Sem mudança

Lista de Faturas

A tabela de faturas exibe todas as faturas com as seguintes colunas:

Descrições das Colunas:

- **ID** - ID único da fatura
- **Título** - Título/descrição da fatura
- **Período** - Período de cobrança (data de início - data de término) ou "N/A" para faturas únicas
- **Data de Vencimento** - Data de vencimento do pagamento
- **Criado** - Data de criação da fatura
- **Valor** - Valor total da fatura (custo de varejo)
- **Status** - Pago, Não Pago ou Reembolsado
- **Ações** - Ações disponíveis (varia de acordo com o status)

Ícones de Ação:

- ⬇ **(Baixar)** - Baixar PDF da fatura
- □ **(Excluir)** - Anular fatura (somente se não paga)
- □ **(Pagar)** - Pagar fatura online (somente se não paga)
- ✉ **(E-mail)** - Enviar e-mail da fatura para o cliente
- □ **(Reembolsar)** - Reembolsar pagamento do Stripe (somente para faturas pagas do Stripe)

Gerando uma Fatura

Clique em **"+ Gerar Fatura Proforma"** para criar uma nova fatura.

Descrições dos Campos:

- **Pesquisar Clientes** - Selecionar cliente (mostrado apenas na visualização em todo o sistema, preenchido automaticamente na visualização do cliente)
- **Título** - Título/nome da fatura (opcional, padrão para "Fatura para [Período]")
- **Data de Início** - Início do período de cobrança (padrão para 14 dias atrás)
- **Data de Término** - Término do período de cobrança (padrão para hoje)
- **Data de Vencimento** - Prazo de pagamento (padrão para hoje)
- **Pré-visualização de Transação** - Mostra todas as transações não faturadas no intervalo de datas com a capacidade de incluir/excluir transações específicas

Seleção de Transação:

- ✓ **(Mais Verde)** - Clique para excluir uma transação da fatura
- × **(X Vermelho)** - Clique para incluir uma transação anteriormente excluída
- **Selecionar Tudo** - Incluir todas as transações exibidas

- **Limpar Tudo** - Excluir todas as transações
- Transações excluídas aparecem esmaecidas com texto riscado
- Totais em tempo real são atualizados à medida que você seleciona/deseleciona transações

O que Acontece:

1. O sistema encontra todas as transações não faturadas para o cliente dentro do intervalo de datas
2. Exibe a pré-visualização da transação com a capacidade de incluir/excluir transações individuais
3. Mostra o cálculo em tempo real do subtotal, imposto e total com base nas transações selecionadas
4. Apenas as transações selecionadas (incluídas) são adicionadas à fatura
5. Gera o PDF da fatura e o armazena em cache
6. Marca as transações selecionadas como faturadas (`invoice_id` preenchido)
7. Transações excluídas permanecem não faturadas e disponíveis para futuras faturas
8. A fatura aparece na lista com status "Não Pago"

Exemplos de Casos de Uso:

Cobrança Mensal: Defina a data de início para o primeiro dia do mês, a data de término para o último dia do mês, a pré-visualização mostra todas as transações não faturadas desse período. Selecione todas ou exclua manualmente transações específicas.

Fatura Específica de Serviço: Use o mesmo intervalo de datas, depois exclua manualmente transações indesejadas (por exemplo, exclua transações não móveis para criar fatura apenas para móveis).

Fatura Única: Defina a data de início e a data de término para o mesmo dia, a pré-visualização mostra apenas transações dessa data. Exclua quaisquer cobranças que não sejam relevantes para esta fatura específica.

Visualizando Detalhes da Fatura

Clique em qualquer linha de fatura na tabela para visualizar os detalhes completos da fatura, incluindo todas as transações, totais e ações disponíveis.

Modal de Detalhes da Fatura:

- **Informações da Fatura** - Mostra ID da fatura, título, datas, status de pagamento e status de anulação
- **Lista de Transações** - Exibe todas as transações incluídas na fatura com:
  - Data da transação
  - Título e descrição
  - Custo de varejo
  - Valor do imposto e percentual (formatado como `$10.00 (10%)`)
  - Transações isentas de impostos mostram "-" na coluna de Imposto
- **Resumo de Totais** - Cálculo em tempo real mostrando:
  - Contagem de transações
  - Subtotal (soma de todos os custos de varejo)
  - Imposto (soma de todos os valores de imposto)
  - Total da Fatura (subtotal + imposto)
- **Botões de Ação** - Mesmas ações disponíveis que na tabela:
  - **Baixar PDF** - Baixar PDF da fatura (sempre disponível)
  - **Enviar E-mail** - Enviar fatura para o cliente (faturas não anuladas)

- **Pagar Fatura** - Processar pagamento (faturas não pagas, não anuladas apenas)
- **Reembolsar** - Reembolsar pagamento do Stripe (somente faturas pagas do Stripe)
- **Excluir** - Anular fatura (faturas não pagas, não anuladas apenas)

Baixando PDFs de Faturas

Clique no **ícone de download (⬇)** na tabela ou no botão **"Baixar PDF"** no modal de detalhes da fatura para baixar uma fatura como PDF.

Processo de Download:

1. Clique no ícone de download ao lado da fatura
2. O indicador de carregamento aparece durante a geração (apenas na primeira vez)
3. O navegador solicita salvar o arquivo: `Invoice_01234.pdf`
4. O PDF é aberto ou salvo na pasta Downloads

Comportamento de Cache de PDF:

- **Primeiro Download** - PDF gerado a partir do modelo, armazenado em cache no banco de dados (pode levar 2-3 segundos)
- **Downloads Subsequentes** - Download instantâneo do cache
- **Invalidar Cache** - Cache limpo se a fatura for modificada ou o modelo atualizado

Resolvendo Problemas de Download:

- **O indicador de carregamento nunca para** - Verifique o console do navegador, a API pode estar fora do ar
- **PDF em branco ou corrompido** - Verifique o modelo de fatura em busca de erros de sintaxe
- **Download falha** - Verifique as configurações do bloqueador de pop-ups, tente um navegador diferente

Pagando Faturas

Clique no **ícone de pagamento (□)** para pagar uma fatura online.

Processo de Pagamento:

1. Clique no ícone de pagamento na fatura não paga
2. O modal de pagamento se abre mostrando os detalhes da fatura
3. Selecione o método de pagamento:
    - **Transação Stripe** - Cobrar cartão de crédito salvo (disponível para todos os usuários)
    - **Dinheiro** - Pagamento manual em dinheiro (somente para funcionários)
    - **Reembolso** - Aplicar reembolso como pagamento (somente para funcionários)
    - **Transação POS** - Terminal de ponto de venda (somente para funcionários)
    - **Transferência Bancária** - Transferência bancária manual (somente para funcionários)
4. Se Stripe for selecionado:
    - Selecione o cartão dos métodos de pagamento salvos
    - Cartão padrão pré-selecionado
    - Clique para selecionar um cartão diferente

5. Se outro método for selecionado:
    - Insira o número de referência (opcional)
6. Clique em **"Pagar Fatura"** para processar
7. O sistema processa o pagamento:
    - **Stripe** - Cobra o cartão via API Stripe
    - **Outros métodos** - Cria uma transação negativa para o valor do pagamento
8. O status da fatura muda para "Pago"
9. Notificação de sucesso exibida

Portal de Autoatendimento vs Pagamento de Funcionários:

:doc:`Portal de Autoatendimento <self_care_portal>` (Clientes):

- Apenas pagamento Stripe disponível
- Deve ter método de pagamento salvo
- Aviso exibido se não houver métodos de pagamento
- Link para adicionar método de pagamento fornecido

Portal de Funcionários (Administradores):

- Todos os métodos de pagamento disponíveis
- Pode marcar fatura como paga manualmente (dinheiro, POS, transferência bancária)
- Pode inserir números de referência para rastreamento

Aviso de Método de Pagamento:

Se o cliente não tiver métodos de pagamento salvos, um aviso é exibido solicitando que ele adicione um método de pagamento antes de poder pagar faturas.

Enviando Faturas por E-mail

Clique no **ícone de e-mail (✉)** para enviar a fatura ao cliente.

O que Acontece:

1. Clique no ícone de e-mail ao lado da fatura
2. O sistema recupera o PDF da fatura do cache (ou gera se não estiver em cache)
3. Envia e-mail via `Mailjet <integrations_mailjet>` usando `api_crmCommunicationCustomerInvoice` modelo
4. O e-mail inclui:
   - PDF da fatura como anexo
   - Nome do cliente
   - Número da fatura e data de vencimento
   - Valor total devido
   - Link para pagar a fatura online
   - Link para visualizar/baixar a fatura
5. Notificação de sucesso: "E-mail da fatura enviado com sucesso"

Destinatários do E-mail:

O e-mail é enviado a todos os contatos do cliente com tipo "cobrança" ou contato principal se não existir contato de cobrança.

Variáveis do Modelo de E-mail:

- `{{ var:customer_name }}` - Nome completo do cliente
- `{{ var:invoice_number }}` - ID da fatura
- `{{ var:invoice_date }}` - Data de emissão da fatura
- `{{ var:due_date }}` - Data de vencimento do pagamento
- `{{ var:total_amount }}` - Valor total devido
- `{{ var:invoice_url }}` - Link para visualizar/baixar PDF
- `{{ var:pay_url }}` - Link para pagar a fatura online

Resolvendo Problemas de E-mail:

- **E-mail não enviado** - Verifique as credenciais da API do Mailjet em `crm_config.yaml`
- **Cliente não recebendo** - Verifique os endereços de e-mail dos contatos do cliente
- **PDF não anexado** - Verifique se a geração do PDF foi bem-sucedida (tente baixar primeiro)

Anulando Faturas

Clique no **ícone de exclusão (🗑)** para anular uma fatura.

Requisitos:

- A fatura deve estar **Não Paga**
- Faturas pagas não podem ser anuladas (devem ser reembolsadas em vez disso)

Como Anular:

1. Localize a fatura não paga na lista
2. Clique no ícone de exclusão (🗑)
3. Confirme no modal:

O que Acontece:

- A fatura é marcada como `void = true`
- Todas as transações são desvinculadas da fatura (`invoice_id` definido como nulo)
- As transações se tornam "não faturadas" novamente
- As transações podem ser incluídas em uma nova fatura
- A fatura aparece na lista com o prefixo "Anulado:" no título
- Ações da fatura desativadas (sem download, pagamento ou e-mail)
- Pode ser visualizada filtrando por faturas "Anuladas"

Notas Importantes:

- Anular NÃO é o mesmo que reembolsar
- **Anular** = "Esta fatura nunca deveria ter existido" (erro de cobrança, duplicado)
- **Reembolsar** = "Reverter uma fatura válida paga" (devolver dinheiro ao cliente)

Reembolsando Faturas

Clique no **ícone de reembolso (□)** para reembolsar uma fatura paga.

Requisitos:

- A fatura deve estar **Paga**
- A fatura deve ter sido paga via **Stripe**
- A fatura deve ter uma `payment_reference` válida (ID da intenção de pagamento do Stripe)
- Disponível apenas para usuários do staff (não para Autoatendimento)

Como Reembolsar:

1. Localize a fatura paga do Stripe
2. Clique no ícone de reembolso (□)
3. O modal de confirmação de reembolso se abre:

4. Clique em **"Confirmar Reembolso"**
5. O sistema processa o reembolso do Stripe:
    - Chama a API do Stripe para reembolsar o pagamento
    - Cria uma transação de reembolso no Stripe
    - Atualiza a fatura com `refund_reference`
6. O status da fatura muda para "Reembolsada"
7. Notificação de sucesso exibida

O que Acontece Após o Reembolso:

- A fatura permanece no sistema (não anulada)
- O status mostra "Reembolsada"
- As transações permanecem vinculadas à fatura

- O cliente recebe o reembolso para o método de pagamento original (3-7 dias úteis)
- O painel do Stripe mostra a transação de reembolso

Restrições de Reembolso:

- Não é possível reembolsar faturas pagas em dinheiro, POS ou transferência bancária (reversão manual necessária)
- Não é possível reembolso parcial (apenas valor total da fatura)
- Não é possível reembolsar duas vezes

Pesquisando e Filtrando Faturas

Pesquisa

Use a barra de pesquisa para encontrar faturas. Pesquisa em:

- ID da fatura
- Título da fatura
- Nome do cliente (apenas visualização em todo o sistema)

Filtros

Aplique filtros para restringir a lista de faturas:

Filtros Disponíveis:

- **Status de Anulação** - Todos, Anulado, Não Anulado
- **Status de Pagamento** - Todos, Pago, Não Pago

Ações de Filtro:

- **Aplicar Filtros** - Aplicar filtros selecionados à lista
- **Redefinir Filtros** - Limpar todos os filtros e mostrar todas as faturas

Classificação

Clique em qualquer cabeçalho de coluna para classificar:

- **ID** - Classificar por ID da fatura (mais novo/mais antigo)
- **Título** - Classificar alfabeticamente
- **Data de Vencimento** - Classificar por data de vencimento
- **Criado** - Classificar por data de criação
- **Valor** - Classificar por custo de varejo (mais alto/mais baixo)
- **Status** - Classificar por status de pagamento (pago primeiro ou não pago primeiro)

Clique novamente para inverter a direção da classificação (crescente ↔ decrescente).

Paginação

Navegue por grandes listas de faturas com controles de página mostrando a página atual, total de páginas e seletor de itens por página (10, 25, 50 ou 100 itens).

Fluxos de Trabalho Comuns de Faturas

Fluxo de Trabalho 1: Cobrança Mensal com Pré-visualização de Transação

1. O final do mês chega (por exemplo, 31 de janeiro)
2. Navegue até **Cobrança → Faturas**
3. Clique em **"+ Gerar Fatura Proforma"**
4. Selecione o cliente (ou faça por cliente se cobrar muitos clientes)
5. Defina as datas:
    - Data de Início: 2025-01-01
    - Data de Término: 2025-01-31

    - Data de Vencimento: 2025-02-15 (15 dias a partir de agora)
    - Título: "Serviços de Janeiro de 2025" (opcional)
6. A seção **Pré-visualização de Transação** aparece mostrando todas as transações não faturadas de janeiro
7. Revise a pré-visualização:
    - Todas as transações estão incluídas por padrão
    - Verifique os totais: Subtotal, Imposto e Total da Fatura
    - Verifique se todas as cobranças estão corretas
8. Clique em **"Gerar Fatura"** (o botão mostra a contagem de transações, por exemplo, "Gerar Fatura (15)")
9. Fatura criada com todas as transações selecionadas
10. Clique na linha da fatura para visualizar detalhes e verificar
11. Clique no botão **"Enviar E-mail"** no modal de detalhes ou no ícone de e-mail na tabela
12. O cliente recebe o e-mail da fatura com PDF e link para pagamento

Fluxo de Trabalho 2: Faturamento Seletivo de Transações

1. O cliente tem múltiplos serviços (Móvel + Internet) e cobranças diversas
2. Deseja faturas separadas para cada serviço
3. **Gerar a primeira fatura (Serviços Móveis):**
    - Clique em **"+ Gerar Fatura Proforma"**
    - Título: "Serviços Móveis - Janeiro de 2025"
    - Início/Fim: 1-31 de janeiro
    - Data de Vencimento: 15 de fevereiro
    - Na pré-visualização de transação, **exclua** todas as transações não móveis:
        - Clique no botão **X** ao lado das transações de Internet
        - Clique no botão **X** ao lado das cobranças diversas
        - Apenas as transações de serviço móvel permanecem selecionadas
    - Verifique se os totais refletem apenas serviços móveis
    - Clique em **"Gerar Fatura"** (mostra a contagem de transações móveis)
4. **Gerar a segunda fatura (Serviços de Internet):**

- Clique em **"+ Gerar Fatura Proforma"** novamente
- Título: "Serviços de Internet - Janeiro de 2025"
- Início/Fim: 1-31 de janeiro (mesmo período)
- Na pré-visualização de transação:
  - Transações móveis já faturadas (não aparecem)
  - Exclua cobranças diversas usando o botão **X**
  - Apenas as transações de serviço de Internet permanecem
- Clique em **"Gerar Fatura"**

5. **Gerar a terceira fatura (Cobranças Adicionais):**
   - Clique em **"+ Gerar Fatura Proforma"** novamente
   - Título: "Cobranças Adicionais - Janeiro de 2025"
   - Apenas cobranças diversas não faturadas aparecem na pré-visualização
   - Clique em **"Selecionar Tudo"** para incluir todas
   - Clique em **"Gerar Fatura"**
6. Envie todas as três faturas para o cliente

Fluxo de Trabalho 3: Excluindo Transações Disputadas ou Pendentes

1. O final do período de cobrança chega
2. Navegue até a aba **Cobrança** do cliente
3. Clique em **"+ Gerar Fatura Proforma"**
4. Defina as datas do período de cobrança
5. A pré-visualização de transação mostra 20 transações
6. O cliente disputou uma cobrança e outra está pendente investigação
7. Na pré-visualização de transação:
   - Localize a transação disputada (por exemplo, "Cobrança de excesso de dados")
   - Clique no botão **X** para excluí-la
   - Localize a transação pendente (por exemplo, "Taxa de instalação")
   - Clique no botão **X** para excluí-la
   - A contagem de transações é atualizada: "18 Transações selecionadas"
   - Os totais são recalculados automaticamente

8. Revise os totais atualizados (exclui valores disputados)
9. Clique em **"Gerar Fatura (18)"**
10. Fatura gerada com apenas transações aprovadas
11. Transações disputadas/pendentes permanecem não faturadas para o próximo ciclo de cobrança

Fluxo de Trabalho 4: Revisão Rápida da Fatura e Ajuste

1. O funcionário gera a fatura mensal
2. A pré-visualização de transação mostra um total inesperadamente alto
3. Revise cada transação na pré-visualização:
    - Note a cobrança duplicada pelo mesmo serviço
    - Clique no **X** para excluir a duplicata
    - Note a transação de teste que não deveria ser cobrada
    - Clique no **X** para excluir a transação de teste
4. Os totais são atualizados em tempo real
5. Verifique se o novo total corresponde ao valor esperado
6. Clique em **"Gerar Fatura"** com as transações corrigidas
7. Volte e anule/exclua as transações excluídas, se necessário
8. Envie a fatura ao cliente com confiança

Fluxo de Trabalho 5: Fatura de Instalação Única

1. O técnico de campo completa a instalação
2. O funcionário adiciona a transação de instalação manualmente
3. Navegue até a aba **Cobrança** do cliente
4. Clique em **"+ Gerar Fatura Proforma"**
5. Defina as datas:
    - Data de Início: hoje
    - Data de Término: hoje
    - Data de Vencimento: hoje + 7 dias
    - Título: "Serviços de Instalação"

6. A pré-visualização de transação mostra apenas as transações de hoje
7. Verifique se a cobrança de instalação aparece
8. Exclua quaisquer cobranças recorrentes usando o botão **X** (se presente)
9. Clique em **"Gerar Fatura"**
10. Envie ao cliente imediatamente
11. O cliente paga online via Stripe

Fluxo de Trabalho 6: Revisando a Fatura Antes do Contato com o Cliente

1. O cliente liga com uma pergunta de cobrança
2. O funcionário navega até a lista de faturas do cliente
3. **Clique na linha da fatura** para abrir o modal de Detalhes da Fatura
4. Revise as informações da fatura:
    - ID da fatura, datas, status
    - Todas as transações incluídas com descrições
    - Detalhamento de impostos por transação
    - Subtotal, Imposto e Totais
5. Responda às perguntas do cliente com detalhes exatos
6. Se o cliente solicitar o PDF, clique no botão **"Baixar PDF"** no modal
7. Se o cliente solicitar o reenvio do e-mail, clique no botão **"Enviar E-mail"**
8. Feche o modal quando terminar

Fluxo de Trabalho 7: Corrigindo Erro de Cobrança

1. O cliente relata uma cobrança incorreta
2. O funcionário clica na linha da fatura para visualizar os detalhes
3. Revê a lista de transações no modal de Detalhes da Fatura
4. Identifica a transação incorreta
5. A fatura está não paga, então pode ser anulada
6. Clique no botão **"Excluir"** no rodapé do modal
7. Confirme a anulação
8. As transações se tornam não faturadas novamente

9. O funcionário modifica ou remove a transação incorreta da lista de transações
10. Gere uma nova fatura com as transações corrigidas:
    - Use a pré-visualização de transação para excluir a transação corrigida, se necessário
    - Inclua apenas cobranças válidas
11. Envie a fatura corrigida ao cliente

Fluxo de Trabalho 8: Processando Múltiplos Pagamentos

1. O cliente traz dinheiro para pagar várias faturas
2. Navegue até a aba **Cobrança** do cliente
3. Veja as faturas não pagas
4. Clique na primeira linha da fatura para visualizar os detalhes
5. Verifique o valor e as transações
6. Clique no botão **"Pagar Fatura"** no rodapé do modal
7. Selecione o método de pagamento **"Dinheiro"**
8. Insira a referência: "Dinheiro pago 2025-01-15"
9. Clique em **"Pagar Fatura"**
10. O modal fecha, a fatura é marcada como "Paga"
11. Repita para as faturas restantes
12. Todas as faturas agora marcadas como "Pagas"

Fluxo de Trabalho 9: Lidando com Solicitação de Reembolso

1. O cliente solicita reembolso por pagamento em excesso
2. O funcionário verifica se a fatura foi paga via Stripe
3. Navegue até a fatura na lista
4. Clique na linha da fatura para visualizar os detalhes
5. Verifique as informações de pagamento e o valor
6. Clique no botão **"Reembolsar"** no rodapé do modal (apenas aparece para faturas do Stripe)

7. Confirme o reembolso
8. O sistema processa o reembolso do Stripe
9. O status da fatura muda para "Reembolsada"
10. O cliente recebe o reembolso em 3-7 dias úteis
11. O funcionário segue com o cliente para confirmar o recebimento

Resolvendo Problemas

Não é possível gerar fatura - Nenhuma transação encontrada

- **Causa:** Nenhuma transação não faturada no intervalo de datas especificado
- **Correção:** Verifique a lista de transações, verifique se as transações existem e não estão já faturadas. Ajuste o intervalo de datas ou remova o filtro.

Falha na geração do PDF da fatura

- **Causa:** Erro de sintaxe no modelo, falha do WeasyPrint ou dados do cliente ausentes
- **Correção:** Verifique o HTML do modelo de fatura em busca de erros, verifique se os campos de endereço do cliente estão preenchidos, revise os logs da API.

Falha no pagamento com erro do Stripe

- **Causa:** Cartão recusado, fundos insuficientes, cartão expirado ou problema na API do Stripe
- **Correção:** Tente um método de pagamento diferente, verifique se o cartão é válido, verifique o painel do Stripe para a razão da recusa.

Não é possível anular a fatura

- **Causa:** Fatura já paga
- **Correção:** Faturas pagas não podem ser anuladas. Se o reembolso for necessário, use a função de reembolso para faturas do Stripe ou crie uma transação de crédito manualmente.

E-mail da fatura não enviado

- **Causa:** Credenciais da API do Mailjet inválidas, cliente sem contato de cobrança ou modelo de e-mail ausente
- **Correção:** Verifique a configuração do Mailjet em `crm_config.yaml`, verifique os contatos do cliente, verifique se o modelo de e-mail da fatura existe.

Botão de reembolso não aparecendo

- **Causa:** Fatura paga em dinheiro/POS/transferência bancária (não Stripe), ou fatura não paga
- **Correção:** O botão de reembolso aparece apenas para pagamentos do Stripe. Para outros métodos de pagamento, crie uma transação de crédito manual.

Download do PDF mostra design antigo do modelo

- **Causa:** PDF armazenado em cache antes da atualização do modelo
- **Correção:** Limpe o cache do PDF da fatura: `DELETE FROM Invoice_PDF_Cache WHERE invoice_id = X;`

Cliente não pode pagar a fatura (sem métodos de pagamento)

- **Causa:** Nenhum método de pagamento salvo no portal de Autoatendimento
- **Correção:** O cliente deve adicionar um cartão de crédito na página **Métodos de Pagamento** antes de pagar faturas.

Múltiplas faturas geradas para o mesmo período

- **Causa:** Funcionário gerou a fatura duas vezes, ou intervalos de datas se sobrepõem
- **Correção:** Anule a fatura duplicada. Ajuste os intervalos de datas para evitar sobreposição. Use a pré-visualização de transação para garantir conjuntos de transações exclusivos.

Pré-visualização de transação mostra nenhuma transação

- **Causa:** Todas as transações no intervalo de datas já estão faturadas ou não existem
- **Correção:** Verifique se o intervalo de datas está correto. Verifique a lista de transações para confirmar se as transações existem. Filtre as faturas para ver qual fatura contém as transações.

Não é possível excluir transação da geração de fatura

- **Causa:** Transação já faturada ou problema no navegador
- **Correção:** Verifique se a transação aparece na pré-visualização com marcação. Atualize a página e tente novamente. Limpe o cache do navegador se o problema persistir.

Total da fatura não corresponde ao valor esperado

- **Causa:** Transações inesperadas incluídas, imposto não calculado ou transações excluídas ainda contadas
- **Correção:** Revise a pré-visualização de transação com atenção. Verifique cada custo de varejo e imposto da transação. Verifique se as transações excluídas estão esmaecidas. Verifique o distintivo de contagem de transações no botão Gerar Fatura.

Botão Gerar Fatura está desativado

- **Causa:** Nenhuma transação selecionada ou intervalo de datas inválido
- **Correção:** Certifique-se de que pelo menos uma transação esteja incluída (não excluída). Verifique se a Data de Início é anterior à Data de Término. Verifique se a Data de Vencimento está definida.

Modal de Detalhes da Fatura não abrindo

- **Causa:** Erro de JavaScript ou página não carregada completamente
- **Correção:** Atualize a página. Verifique o console do navegador para erros. Tente um navegador diferente. Verifique a conexão com a internet.

Imposto da transação não exibido nos Detalhes da Fatura

- **Causa:** Transação tem imposto de 0% ou tax_amount é nulo

- **Correção:** Verifique se a transação tem tax_percentage definido. Verifique se tax_amount foi calculado quando a transação foi criada. Atualize a transação se necessário.

Botões de ação ausentes no modal de Detalhes da Fatura

- **Causa:** A fatura está anulada ou o usuário não tem permissões
- **Correção:** Faturas anuladas mostram apenas o botão Baixar PDF. Verifique o status da fatura. Verifique o papel e as permissões do usuário.

Documentação Relacionada

- `integrations_mailjet` - Entrega e modelos de e-mail de fatura
- `administration_configuration` - Configuração do modelo de fatura
- `payments_transaction` - Criando transações que aparecem nas faturas
- `payments_process` - Processando pagamentos de faturas
- `basics_payment` - Gerenciamento de métodos de pagamento
- `payment_system_guide` - Referência da API de pagamento e configuração de fornecedor

Processar Pagamentos

A maioria dos pagamentos será processada automaticamente pelo sistema, mas há momentos em que você pode precisar processar um pagamento manualmente.

Para pagar uma fatura, selecione a fatura não paga e clique no botão "Pagar".

Isso abrirá um formulário de pagamento, onde você pode inserir o método de pagamento e clicar em "Enviar" para processar o pagamento.

O cliente receberá automaticamente um recibo pelo pagamento, e a fatura será marcada como paga.

Para transferências bancárias, você pode inserir a referência do pagamento e a data em que o pagamento foi feito (se diferente da data atual).

Transações de Clientes

Qualquer coisa que custe dinheiro no sistema é registrada como uma transação sob o cliente.

Cada transação tem um valor monetário para custo de atacado e custo de varejo, e uma descrição do que a transação é.

As transações podem ser geradas automaticamente pelo sistema, por exemplo, quando um serviço é provisionado, uma transação é criada para o custo de configuração, e quando um serviço é cobrado, uma transação é criada para o custo de varejo.

As transações também podem ser criadas manualmente, por exemplo, se um cliente receber um crédito, uma transação é criada para o valor do crédito, ou se uma taxa de instalação for cobrada, uma transação é criada para a taxa de instalação.

As transações são agrupadas para formar `Invoices <payments_invoices>`, que é enviada ao cliente para pagamento.

Acessando Transações

As transações podem ser visualizadas no nível do sistema ou por cliente:

Visualização por Cliente:

1. Navegue até **Clientes → [Selecionar Cliente]**
2. Clique na aba **Cobrança**
3. Veja a lista de transações no primeiro cartão

Visualização em Todo o Sistema:

1. Navegue até **Cobrança → Transações** (no menu principal)
2. Veja todas as transações de todos os clientes

Widgets de Estatísticas de Transações

Na parte superior da página de transações, quatro cartões de estatísticas exibem resumos financeiros:

Descrições dos Widgets:

- **Total de Transações** - Soma de todos os custos de varejo das transações (todo o tempo)
- **Total de Transações Não Faturadas** - Soma de transações ainda não incluídas em uma fatura
- **Total de Transações Este Mês** - Soma de transações criadas neste mês calendário
- **Total de Transações Mês Passado** - Soma de transações criadas no mês calendário passado

Formatação de Valores:

- Valores acima de 1.000: Exibir como sufixo "k" (ex: $1.5k)
- Valores acima de 1.000.000: Exibir como sufixo "M" (ex: $2.3M)
- Valores acima de 1.000.000.000: Exibir como sufixo "B" (ex: $1.1B)

Lista de Transações

A tabela de transações exibe todas as transações com as seguintes colunas:

Descrições das Colunas:

- **ID** - ID único da transação
- **Data** - Data de criação da transação
- **Título** - Nome curto da transação
- **Descrição** - Descrição detalhada do que a transação é
- **Valor** - Custo de varejo (positivo para cobranças, negativo para créditos)
- **Fatura** - ID da fatura se a transação foi faturada (link clicável)
- **Status** - Marca de verificação se faturada, traço se ainda não faturada

Ações Por Linha:

Cada linha tem um menu de ações (⋮) com opções:

- **Ver Detalhes** - Abre o modal de detalhes da transação
- **Baixar PDF da Fatura** - Baixar PDF (somente se faturada)

- **Anular Transação** - Marcar transação como anulada (somente se não faturada)

Tipos de Transações

As transações se enquadram em duas categorias principais:

Transações de Débito (Cobranças)

Valores positivos que aumentam o saldo devedor do cliente:

- **Taxas de Configuração de Serviço** - Cobranças únicas quando o serviço é provisionado
- **Taxas Mensais de Serviço** - Cobranças recorrentes por serviços
- **Taxas de Instalação** - Cobranças por visitas de técnicos de campo
- **Cobranças de Equipamento** - Cobranças por modems, roteadores, cartões SIM
- **Taxas de Pagamento Atrasado** - Penalidades por faturas em atraso
- **Cobranças Manuais** - Cobranças personalizadas adicionadas pela equipe

Transações de Crédito (Pagamentos/Reembolsos)

Valores negativos que diminuem o saldo devedor do cliente:

- **Pagamentos em Dinheiro** - Cliente pagou em dinheiro
- **Pagamentos com Cartão** - Cliente pagou com cartão de crédito/débito
- **Pagamentos por Transferência Bancária** - Cliente pagou via transferência bancária
- **Créditos na Conta** - Créditos de boa vontade, compensação
- **Reembolsos** - Dinheiro devolvido ao cliente
- **Descontos** - Descontos promocionais ou de fidelidade

Adicionando uma Transação Manualmente

Clique em **"+ Adicionar Transação"** para abrir o modal de adicionar transação.

Transação de Débito (Cobrança):

Transação de Crédito (Pagamento/Reembolso):

Descrições dos Campos:

- **Tipo de Transação** - Selecione Débito (cobrança) ou Crédito (pagamento/reembolso)
- **Tipo de Crédito** - Se Crédito selecionado, escolha o método de pagamento (Dinheiro, Cartão, Transferência Bancária)
- **Título** - Nome curto para a transação (obrigatório)
- **Descrição** - Explicação detalhada (opcional)
- **Custo de Varejo** - Valor que o cliente paga (obrigatório, número positivo)
- **Custo de Atacado** - Seu custo (opcional, para rastreamento de margem)
- **Porcentagem de Imposto** - Taxa de imposto aplicada a esta transação (opcional, padrão para imposto do produto ou 0%)
- **Serviço** - Vincular transação a um serviço específico (opcional)
- **Site** - Vincular transação a um site específico (opcional)
- **Data da Transação** - Data da transação (padrão para hoje)

Validação:

- Título e custo de varejo são obrigatórios
- O custo de varejo deve ser um número positivo
- Se o tipo de Crédito for selecionado, um tipo de crédito deve ser escolhido

O Que Acontece:

1. Transação criada no banco de dados
2. Aparece na lista de transações do cliente
3. Incluída na contagem de "Transações Não Faturadas"
4. Disponível para inclusão na próxima geração de fatura
5. Entrada de log de atividade criada

Pesquisando e Filtrando Transações

Pesquisa

Use a barra de pesquisa para encontrar transações. Pesquisa em:

- ID da Transação
- Título
- Descrição
- ID da Fatura

Filtros

Aplique filtros para restringir a lista de transações:

Filtros Disponíveis:

- **Status de Anulação** - Todos, Anulado, Não Anulado
- **Status da Fatura** - Todos, Faturado, Não Faturado

Ações de Filtro:

- **Aplicar Filtros** - Aplicar filtros selecionados à lista
- **Redefinir Filtros** - Limpar todos os filtros e mostrar todas as transações

Ordenação

Clique em qualquer cabeçalho de coluna para ordenar:

- **ID** - Ordenar por ID da transação (mais novo/mais antigo)
- **Data** - Ordenar por data da transação
- **Título** - Ordenar alfabeticamente
- **Valor** - Ordenar por custo de varejo (mais alto/mais baixo)
- **Fatura** - Ordenar por ID da fatura

Clique novamente para inverter a direção da ordenação (crescente ↔ decrescente).

Anulando Transações

Transações adicionadas por erro podem ser **anuladas** (marcadas como deletadas).

Requisitos:

- A transação NÃO deve ser faturada
- Uma vez faturada, as transações não podem ser anuladas (devem ser reembolsadas em vez disso)

Como Anular:

1. Localize a transação na lista
2. Clique no menu de ações (⋮)
3. Selecione **"Anular Transação"**
4. Confirme no modal

O Que Acontece:

- Transação marcada como `void = true`
- Não aparece mais na lista padrão de transações
- Excluída da geração de faturas
- Pode ser visualizada filtrando por transa��ões "Anuladas"
- Deduza do total de "Transações Não Faturadas"

Nota: Anular não é o mesmo que reembolsar. Anular significa "esta transação nunca deveria ter existido." Reembolsar significa "reverter uma transação válida."

Imposto sobre Transações

As transações podem incluir imposto, que é calculado automaticamente com base na configuração de imposto do produto ou especificado manualmente por transação.

Comportamento do Imposto:

- **Transações de Débito (Cobranças)** - O imposto é aplicado às cobranças com base em:
  - **Porcentagem de Imposto do Produto** - Se a transação estiver vinculada a um produto, a porcentagem de imposto do produto é aplicada automaticamente
  - **Substituição Manual** - A equipe pode substituir a porcentagem de imposto ao criar uma transação
  - **Valor do Imposto** - Calculado como: `retail_cost × (tax_percentage / 100)`
  - **Formato de Exibição** - Exibido como: `$10.00 (10%)` nas listas de transações
- **Transações de Crédito (Pagamentos/Reembolsos)** - Nenhum imposto é aplicado a créditos
  - O campo de porcentagem de imposto é oculto para transações de crédito

    - O imposto é automaticamente definido como 0% para todos os pagamentos e reembolsos
    - Créditos reduzem o saldo devedor do cliente sem implicações fiscais

Exemplo de Cálculo de Imposto:

- Produto: Plano Móvel com 10% de imposto, $50.00 de custo de varejo
- Cálculo Automático de Imposto: $50.00 × 0.10 = $5.00
- Exibição: $5.00 (10%)

Imposto Zero (NIL/Isento):

- Produtos podem ser isentos de impostos definindo a porcentagem de imposto como 0
- O imposto padrão é 0% se não especificado
- Transações isentas de impostos mostram "-" na coluna de Imposto

Visualização de Detalhes da Transação

Clique em uma transação para ver todos os detalhes:

Transações Faturadas vs Não Faturadas

Transações Não Faturadas:

- Ainda não incluídas em nenhuma fatura
- Disponíveis para a próxima geração de fatura
- Podem ser anuladas
- Contam para o total de "Transações Não Faturadas"
- Status mostra traço (-)

Transações Faturadas:

- Incluídas em uma fatura
- Não podem ser anuladas (devem ser reembolsadas se necessário)
- ID da fatura clicável (link para detalhes da fatura)
- Status mostra marca de verificação (✓)
- Não podem ser modificadas

Geração de Fatura:

Quando você gera uma fatura para um cliente:

1. O sistema encontra todas as transações não faturadas para aquele cliente
2. Opcionalmente filtrar por intervalo de datas
3. Transações incluídas na nova fatura
4. Campo `invoice_id` da transação populado
5. Transação agora marcada como "faturada"

Veja `payments_invoices` para detalhes sobre geração de faturas.

Fluxos de Trabalho Comuns

Fluxo de Trabalho 1: Crédito Manual por Interrupção de Serviço

1. Cliente liga: "O serviço ficou fora do ar por 2 dias"
2. A equipe decide creditar £10
3. Navegue até a aba **Cobrança** do cliente
4. Clique em **"+ Adicionar Transação"**
5. Selecione o tipo de transação **Crédito**
6. Selecione o tipo de crédito **Pagamento em Dinheiro**
7. Digite o título: "Crédito por Interrupção de Serviço"
8. Digite a descrição: "Compensação por interrupção de 2 dias 8-9 Jan"
9. Digite o custo de varejo: 10.00
10. Selecione o serviço afetado no dropdown
11. Clique em **"Adicionar Transação"**
12. Transação aparece com valor -£10.00
13. Será incluída na próxima fatura como crédito

Fluxo de Trabalho 2: Taxa de Instalação Manual

1. Técnico de campo instala o serviço
2. A equipe precisa cobrar £75 de taxa de instalação
3. Navegue até a aba **Cobrança** do cliente
4. Clique em **"+ Adicionar Transação"**
5. Selecione o tipo de transação **Débito**
6. Digite o título: "Taxa de Instalação"
7. Digite a descrição: "Visita do técnico de campo para instalação de fibra"
8. Digite o custo de varejo: 75.00
9. Digite o custo de atacado: 45.00 (opcional, para rastreamento de margem)
10. Selecione o serviço instalado
11. Selecione o site onde foi instalado
12. Clique em **"Adicionar Transação"**

13. Transação aparece na lista de não faturadas
14. Será incluída na próxima fatura

Fluxo de Trabalho 3: Anulando Transação Duplicada

1. A equipe nota uma transação duplicada
2. Verifique se a transação NÃO foi faturada
3. Clique no menu de ações (⋮) na transação duplicada
4. Selecione **"Anular Transação"**
5. Confirme no modal
6. Transação removida da lista
7. Total não faturado diminui de acordo

Fluxo de Trabalho 4: Encontrando Transações para Fatura

1. Precisa gerar fatura mensal
2. Clique no **filtro de Fatura: "Não Faturado"**
3. Clique em **Aplicar Filtros**
4. Veja todas as transações não faturadas
5. Anote o valor total dos widgets
6. Navegue para gerar fatura
7. Selecione intervalo de datas (ex: 1-31 Jan)
8. Transações no intervalo incluídas na fatura

Solução de Problemas

Não é possível anular a transação

- **Causa:** Transação j�� faturada
- **Solução:** A transação faz parte do histórico de faturas. Se um reembolso for necessário, crie uma transação de Crédito em vez disso.

Transações duplicadas aparecendo

- **Causa:** Serviço cobrado várias vezes ou erro de provisionamento
- **Solução:** Anule a(s) transação(ões) duplicadas se não faturadas. Se faturadas, emita crédito.

Transação não aparece na lista

- **Causa:** Filtros aplicados ou transação anulada
- **Solução:** Clique em "Redefinir Filtros" para mostrar todas as transações. Para ver transações anuladas, filtre por "Anular: Anulado".

Total não faturado não corresponde ao esperado

- **Causa:** Algumas transações já faturadas ou transações anuladas excluídas
- **Solução:** Aplique o filtro "Fatura: Não Faturada" para ver apenas as não faturadas. Verifique transações anuladas separadamente.

Não é possível adicionar transação (campo do cliente desativado)

- **Causa:** Visualizando a página de transações específicas do cliente
- **Solução:** O cliente está pré-selecionado. Se você precisar adicionar uma transação para um cliente diferente, vá para a página de Transações em todo o sistema.

Documentação Relacionada

- `payments_invoices` - Geração e gerenciamento de faturas
- `payments_process` - Processamento de pagamentos contra faturas
- `basics_payment` - Visão geral dos métodos de pagamento
- `csa_activity_log` - Visualizando o histórico de transações no log de atividade

Cobranças e Pagamentos a partir de Playbooks

Este guia explica como implementar cobranças e processamento de pagamentos dentro de playbooks Ansible para fluxos de trabalho de provisionamento do OmniCRM.

> **Referência Completa da API**: Para detalhes completos sobre APIs de pagamento, roteamento de carteira, reembolsos e arquitetura independente de fornecedor, consulte o **Guia da API do Sistema de Pagamento**

Visão Geral

Os Playbooks Ansible do OmniCRM podem lidar com o processamento de pagamentos de várias maneiras diferentes, mas todas elas apenas chamam chamadas de API no CRM para cobrar.

1. **Fluxo de Pagamento de Compromisso em Duas Fases** - Para serviços pagos que requerem autorização e captura de pagamento imediatas (ou seja, Serviços Pré-pagos)
2. **Criação de Transação Direta** - Para adicionar cobranças/créditos sem processamento de pagamento imediato (por exemplo, taxas de configuração, créditos manuais) que podem ser processados posteriormente (ou seja, serviços Pós-pagos)

Essas abordagens garantem transações atômicas onde os clientes são cobrados apenas se o provisionamento for bem-sucedido, com rollback automático se qualquer etapa falhar.

Flexibilidade de Preços: Playbooks como um Motor de Scripting

Importante: Os valores em dólar definidos em produtos e serviços (por exemplo, `retail_cost`, `wholesale_cost`, `retail_setup_cost`) são simplesmente **valores padrão** armazenados no banco de dados. Eles não ditam o que você deve cobrar—o playbook tem controle total sobre o preço final.

O playbook Ansible é essencialmente um **motor de scripting** que você pode adaptar para atender a quaisquer requisitos de lógica de negócios. Você pode:

- **Usar os preços armazenados como estão** - Simplesmente referencie `api_response_product.json.retail_cost` em sua autorização
- **Substituir os preços completamente** - Cobrar um valor diferente, independentemente do que está na definição do produto
- **Aplicar descontos dinâmicos** - Calcular porcentagens com base no nível do cliente, promoções ou fidelidade
- **Implementar preços em camadas** - Cobrar taxas diferentes com base na quantidade, níveis de uso ou termos de contrato
- **Preços em pacotes** - Combinar vários produtos com descontos personalizados em pacotes
- **Preços baseados em tempo** - Ajustar preços com base na hora do dia, estação ou períodos promocionais
- **Pre��os específicos para o cliente** - Consultar taxas negociadas a partir de contratos de clientes

Exemplo: Substituindo o Preço do Produto

```
# O retail_cost do produto é $99.00, mas queremos cobrar $79.00
por uma promoção
- name: Definir preço promocional (ignorando retail_cost do
produto)
  set_fact:
    charge_amount: 79.00

- name: Autorizar pagamento promocional
  uri:
    url: "http://localhost:5000/crm/payments/authorize/hold"
    method: POST
    body_format: json
    body:
      customer_id: "{{ customer_id | int }}"
      amount: "{{ charge_amount | float }}"  # Usando nossa
substituição, não retail_cost
      # ...
```

Exemplo: Desconto por Nível de Cliente

```
# Aplicar desconto com base no nível do cliente
- name: Obter nível do cliente
  uri:
    url: "http://localhost:5000/crm/customer/{{ customer_id }}"
    # ...
  register: customer_info

- name: Calcular desconto por nível
  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: Aplicar desconto ao preço final
  set_fact:
    final_price: "{{ (base_price | float) * (1 - (discount_percent
| float / 100)) | round(2) }}"
```

Exemplo: Preços para Parceiros Atacadistas

```
# Parceiros atacadistas pagam wholesale_cost em vez de retail_cost
- name: Determinar preço com base no tipo de cliente
  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 %}
```

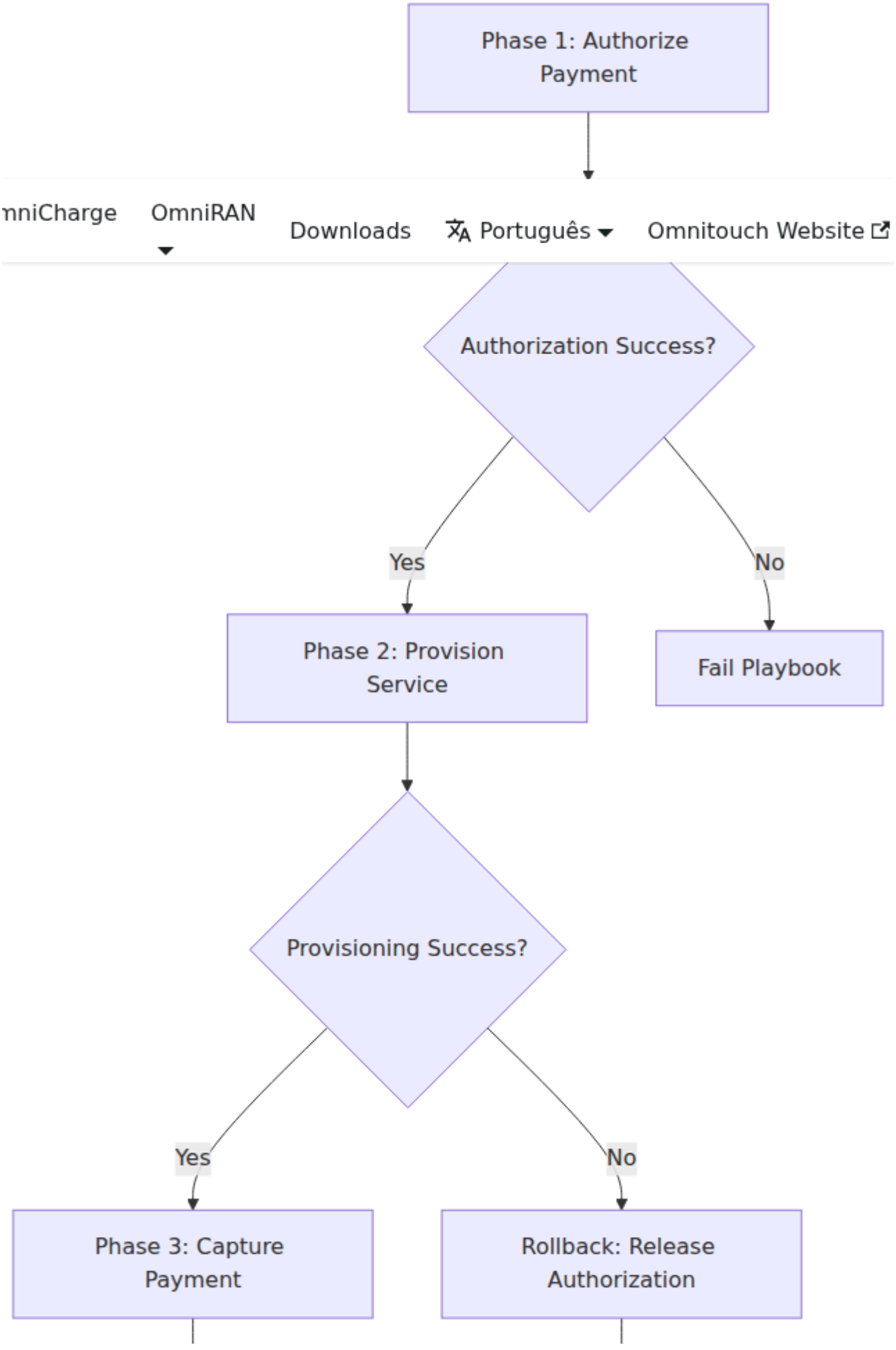
A principal conclusão é que **você não está preso a nenhum modelo de preços**. O playbook lhe dá controle programático total para implementar quaisquer regras de negócios que sua organização exigir. Os preços de

produtos/serviços no CRM são apenas valores padrão convenientes que os playbooks podem usar ou ignorar conforme necessário.

Fluxo de Pagamento de Compromisso em Duas Fases

O padrão de compromisso em duas fases é usado quando você precisa cobrar o método de pagamento de um cliente durante o provisionamento. Isso garante que o cliente seja cobrado apenas se o provisionamento for concluído com sucesso.

Visão Geral do Fluxo


Phase 1: Authorize Payment
nniCharge
OmniRAN
Downloads
Português
Omnitouch Website
Authorization Success?
Yes
No
Phase 2: Provision Service
Fail Playbook
Provisioning Success?
Yes
No
Phase 3: Capture Payment
Rollback: Release Authorization

Padrão de Implementação

O padrão segue essas fases:

Fase 1: Autorização - Reservar fundos no método de pagamento
Fase 2: Provisionamento - Executar atualizações de saldo/ação do OCS
Fase 3: Captura - Finalizar pagamento após provisionamento bem-sucedido
Rollback - Liberar autorização se o provisionamento falhar

Exemplo Completo

Aqui está um exemplo completo do `play_topup_charge_then_action.yaml`:

```yaml
- name: Play Topup - Charge card then action the Topup
  hosts: localhost
  gather_facts: no
  become: False

  tasks:
    - name: Include vars of crm_config
      ansible.builtin.include_vars:
        file: "../../crm_config.yaml"
        name: crm_config

    # Obter informações do produto e serviço
    - 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: 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
        validate_certs: no
      register: api_response_service

    - name: Set service and package 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
}}"
```

```
        wholesale_cost: "{{
api_response_product.json.wholesale_cost }}"

    # Obter o método de pagamento padrão do cliente
    - name: Get the Customer Payment Methods from Payment
Controller
      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: Get the default payment_method_id from the response
      set_fact:
        payment_method_id: "{{ api_response_payment_methods.json |
json_query(query) }}"
      vars:
        query: "data[?is_default==`true`].payment_method_id | [0]"

    # ============================================================
    # FLUXO DE PAGAMENTO DE COMPROMISSO EM DUAS FASES
    # ============================================================

    - name: "Fase 1: Autorizar pagamento (reservar fundos)"
      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 }} on {{
```

```
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: Assert authorization was successful
      assert:
        that:
          - api_response_authorization.status == 200
          - api_response_authorization.json.success == true

    - name: Store authorization_id for capture/release
      set_fact:
        authorization_id: "{{
api_response_authorization.json.data.authorization_id }}"

    # Fase 2: Provisionamento OCS (envolto em bloco para rollback
transacional)
    - block:
        - name: "Fase 2: Executar ação 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: Assert that the response from the action is OK
          assert:
            that:
              - action_execute_response.status == 200
              - action_execute_response.json.result == "OK"

        # Fase 3: Captura de Pagamento - Finalizar transação após
provisionamento bem-sucedido
        - name: "Fase 3: Capturar pagamento autorizado"
          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: Assert capture was successful
          assert:
            that:
              - api_response_capture.status == 200
              - api_response_capture.json.success == true

      rescue:
```

```
        # Rollback da Transação: Anular autorização para liberar
fundos retidos
        - name: "Rollback: Liberar autorização de pagamento"
          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: Terminate playbook execution after rollback
          fail:
            msg: "Provisionamento OCS falhou. Autorização de
pagamento {{ authorization_id }} anulada. Cliente não cobrado."
```

Endpoints da API de Pagamento

> **Roteamento de Carteira em Primeiro Lugar**: O sistema de pagamento otimiza automaticamente as cobranças em cartão usando o saldo da carteira primeiro. Se um cliente tem $1 em sua carteira e compra um addon de $10, apenas $9 são cobrados em seu cartão. Consulte a seção de Roteamento de Carteira em Primeiro Lugar para detalhes.

1. Autorizar/Reservar Pagamento

Endpoint: `POST /crm/payments/authorize/hold`

Este endpoint coloca uma reserva no método de pagamento do cliente pelo valor especificado. Os fundos são reservados, mas ainda não capturados.

Comportamento de Carteira em Primeiro Lugar: A autorização calcula automaticamente a diferença após verificar o saldo da carteira. Se o saldo da carteira é $150 e a autorização é para $500, apenas $350 é autorizado no cartão. A carteira é debitada no momento da captura.

Corpo da Solicitação:

```
{
  "customer_id": 123,
  "amount": 49.99,
  "currency": "AUD",
  "payment_method_id": 456,
  "metadata": {
    "description": "Nome e descrição do pacote",
    "service_id": 789,
    "site_id": 12,
    "product_id": "34",
    "user_id": 5,
    "title": "Nome do Pacote",
    "wholesale_cost": 25.00,
    "invoice": true,
    "contract_days": 30,
    "send_email": true
  }
}
```

Descrições dos Campos:

- **customer_id** (obrigatório) - ID do cliente que está sendo cobrado
- **amount** (obrigatório) - Valor a ser autorizado em formato decimal
- **currency** (opcional) - Código da moeda (padrão para a moeda do sistema)
- **payment_method_id** (obrigatório) - Método de pagamento a ser cobrado
- **metadata** (opcional) - Dados adicionais para a criação da transação:
  - **description** - Descrição da transação
  - **service_id** - Serviço sendo cobrado
  - **site_id** - Site associado ao serviço
  - **product_id** - Produto sendo provisionado
  - **user_id** - Usuário que inicia a cobrança (opcional)

- **title** - Título da transação
- **wholesale_cost** - Seu custo (para rastreamento de margem)
- **invoice** - Se `true`, cria automaticamente a transação quando capturada
- **contract_days** - Duração do contrato
- **send_email** - Se `true`, envia notificação por e-mail

Resposta:

```
{
  "success": true,
  "message": "Pagamento autorizado (reserva criada)",
  "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": "Cartão autorizado para $350 (recarga de carteira).
O débito da carteira de $500 ocorrerá na captura."
  }
}
```

Importante:

- Salve o `authorization_id` para uso nas chamadas de captura ou liberação
- Note que o `card_amount` mostra que apenas a diferença foi autorizada
- O saldo da carteira é verificado, mas NÃO é debitado até a captura

2. Capturar Pagamento

Endpoint: `POST /crm/payments/capture/{authorization_id}`

Este endpoint finaliza a autorização de pagamento e cobra o cliente. Se `invoice: true` foi definido nos metadados da autorização, uma transação é criada automaticamente.

Corpo da Solicitação:

```
{
  "metadata": {
    "provisioning_status": "success",
    "cgr_action": "Action_Topup_Standard",
    "additional_info": "Qualquer outro dado relevante"
  }
}
```

Resposta:

```
{
  "success": true,
  "data": {
    "payment_id": "pay_xyz789",
    "transaction_id": 1234
  }
}
```

O que Acontece:

1. **Cartão capturado** pelo valor da diferença (se o cartão foi autorizado)
2. **Carteira creditada** com o valor capturado do cartão
3. **Carteira debitada** pelo valor total do serviço
4. Se `invoice: true` estava nos metadados da autorização:
   - **Transação de débito** criada (valor positivo = cobrança)
   - **Fatura** criada e vinculada à transação de débito
   - **Transação de crédito** criada (valor negativo = pagamento recebido)
   - Fatura marcada como PAGA (débito e crédito líquidas a zero)
   - Transação aparece na cobrança do cliente
5. Registro de pagamento é criado no banco de dados
6. Se `send_email: true`, o cliente recebe um e-mail de fatura

Exemplo: Autorização de $500 com saldo de carteira de $150:

- Cartão capturado: $350

- Carteira creditada: +$350 (carteira agora $500)
- Carteira debitada: -$500 (cobrança do serviço)
- Carteira final: $0

Consulte Exemplos de Roteamento de Carteira em Primeiro Lugar para o fluxo detalhado.

3. Liberar Autorização de Pagamento

Endpoint: `POST /crm/payments/release/{authorization_id}`

Este endpoint cancela uma autorização de pagamento e libera os fundos retidos. Use isso em cenários de resgate/rollback quando o provisionamento falhar.

Corpo da Solicitação:

```
{
  "metadata": {
    "release_reason": "provisioning_failed",
    "error_details": "Falha na criação da conta OCS"
  }
}
```

Resposta:

```
{
  "success": true,
  "message": "Autorização liberada"
}
```

O que Acontece:

1. **Autorização do cartão liberada** (se o cartão foi autorizado)
2. Fundos retidos retornam ao crédito disponível do cliente
3. **Carteira NÃO debitada** (já que o débito só acontece na captura)
4. Cliente NÃO é cobrado

5. Nenhuma transação é criada

Nota: Com o roteamento de carteira em primeiro lugar, nenhum reembolso da carteira é necessário, pois a carteira não é debitada até o momento da captura.

Criação Direta de Transação

Para cobranças que não requerem processamento de pagamento (taxas de configuração, créditos manuais, ajustes), você pode criar transações diretamente via API.

Adicionando uma Transação via API

Endpoint: `PUT /crm/transaction/`

Exemplo do `play_simple_service.yaml`:

```
- name: Adicionar Transação de Custo de Configuração 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 }} - Custos de Configuração",
        "description": "Custos de Configuração para {{
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: Assert that the response from the transaction is OK
  assert:
    that:
      - api_response_transaction.status == 200
```

Campos do Corpo da Solicitação:

- **customer_id** (obrigatório) - ID do cliente
- **service_id** (opcional) - ID do serviço para vincular a transação
- **title** (obrigatório) - Nome curto da transação
- **description** (opcional) - Descrição detalhada
- **invoice_id** (opcional) - ID da fatura se já faturado (geralmente nulo)
- **wholesale_cost** (opcional) - Seu custo
- **retail_cost** (obrigatório) - Custo para o cliente
- **site_id** (opcional) - ID do site para vincular a transação

- **tax_percentage** (opcional) - Percentual da taxa

Casos de Uso:

- Taxas de configuração durante o provisionamento do serviço
- Taxas de instalação
- Créditos ou ajustes manuais
- Taxas de equipamento
- Taxas administrativas

Nota: A criação direta de transações NÃO processa pagamento - apenas cria um registro de cobrança. A transação aparecerá como não faturada e pode ser incluída em faturas futuras.

Cálculo de Cobranças Proporcionais

Cobranças proporcionais permitem que você cobre os clientes proporcionalmente com base em períodos de faturamento parciais. Isso é comum quando:

- O cliente se inscreve no meio do mês
- O serviço é atualizado/reduzido no meio do ciclo
- O faturamento é calculado com base nos dias utilizados

Fórmula de Cálculo Proporcional

```
pro_rata_charge = (monthly_cost × days_remaining) / days_in_month
```

Exemplo de Implementação

Aqui está como calcular uma cobrança proporcional em um playbook:

```yaml
# Calcular cobrança proporcional para mês parcial
# Se o cliente se inscreve no dia 15 e o faturamento é no dia 1,
# cobrar proporcionalmente pelos dias restantes

- name: Obter dia atual do mês
  command: "date +%d"
  register: current_day

- name: Obter total de dias no mês atual
  command: "date -d 'last day of this month' +%d"
  register: days_in_month

- name: Obter último dia do mês
  command: "date -d 'last day of this month' +%Y-%m-%d"
  register: last_day_of_month

- name: Calcular dias restantes no mês
  set_fact:
    days_remaining: "{{ (days_in_month.stdout | int) -
(current_day.stdout | int) + 1 }}"

- name: Calcular custo proporcional
  set_fact:
    pro_rata_cost: "{{ ((monthly_cost | float) * (days_remaining |
float) / (days_in_month.stdout | float)) | round(2) }}"

- name: Exibir detalhes do cálculo
  debug:
    msg:
      - "Custo mensal: ${{ monthly_cost }}"
      - "Dias no mês: {{ days_in_month.stdout }}"
      - "Dias restantes: {{ days_remaining }}"
      - "Cobrança proporcional: ${{ pro_rata_cost }}"

# Usar pro_rata_cost na autorização ou criação de transação
- name: "Autorizar pagamento proporcional"
  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 }} (Proporcional {{
days_remaining }} dias)",
          "description": "Cobrança proporcional por {{
days_remaining }}/{{ days_in_month.stdout }} dias de {{
package_name }}",
          "service_id": "{{ service_id | int }}",
          "invoice": true
        }
      }
```

Proporcional a partir de Data de Início Personalizada

Se você precisar calcular proporcional a partir de uma data de início específica até uma data de faturamento:

```
- name: Definir data de início personalizada e data de faturamento
  set_fact:
    service_start_date: "2024-01-15"
    next_billing_date: "2024-02-01"

- name: Calcular dias entre datas
  shell: |
    echo $(( ( $(date -d "{{ next_billing_date }}" +%s) - $(date -
d "{{ service_start_date }}" +%s) ) / 86400 ))
  register: days_until_billing

- name: Obter dias no período de faturamento (geralmente 30)
  set_fact:
    billing_period_days: 30

- name: Calcular custo proporcional
  set_fact:
    pro_rata_cost: "{{ ((monthly_cost | float) *
(days_until_billing.stdout | float) / (billing_period_days |
float)) | round(2) }}"

- name: Exibir cálculo
  debug:
    msg:
      - "Data de início: {{ service_start_date }}"
      - "Próximo faturamento: {{ next_billing_date }}"
      - "Dias até o faturamento: {{ days_until_billing.stdout }}"
      - "Cobrança proporcional: ${{ pro_rata_cost }}"
```

Cenários de Exemplo Proporcional

Cenário 1: Inscrição no Meio do Mês

- O cliente se inscreve no dia 15 de janeiro
- Custo mensal: $60.00
- Dias em janeiro: 31
- Dias restantes: 17 (do dia 15 ao 31, inclusive)
- Cobrança proporcional: `$60.00 × 17 ÷ 31 = $32.90`

Cenário 2: Atualização de Serviço

- O cliente atualiza no 10º dia do ciclo de faturamento
- Plano antigo: $30/mês
- Novo plano: $50/mês
- Dias no ciclo: 30
- Dias restantes: 21
- Diferença: $20/mês
- Cobrança proporcional: `$20.00 × 21 ÷ 30 = $14.00`

Melhores Práticas

1. Sempre Use o Padrão de Bloco/Rescue

Envolva a captura de pagamento e o provisionamento em um bloco/rescue para garantir rollback:

```
- block:
    # Tarefas de provisionamento
    - name: Provisionar serviço
      uri: ...

    # Capturar pagamento apenas após o sucesso
    - name: Capturar pagamento
      uri: ...

  rescue:
    # Liberar autorização se algo falhar
    - name: Liberar autorização de pagamento
      uri: ...

    - name: Falhar execução do playbook
      fail:
        msg: "Provisionamento falhou, cliente não cobrado"
```

2. Valide Todas as Respostas da API

Sempre afirme que operações críticas foram bem-sucedidas:

```
- name: Autorizar pagamento
  uri: ...
  register: api_response_authorization

- name: Assert authorization was successful
  assert:
    that:
      - api_response_authorization.status == 200
      - api_response_authorization.json.success == true
    fail_msg: "A autorização de pagamento falhou: {{
api_response_authorization.json }}"
```

3. Armazene o ID de Autorização

Sempre salve o authorization_id para uso na captura/liberação:

```
- name: Store authorization_id for capture/release
  set_fact:
    authorization_id: "{{
api_response_authorization.json.data.authorization_id }}"
```

4. Use Metadados de Forma Eficaz

Inclua metadados abrangentes nas solicitações de autorização:

```
metadata:
  description: "Descrição clara do que o cliente está sendo
cobrado"
  service_id: "{{ service_id | int }}"
  product_id: "{{ product_id }}"
  user_id: "{{ initiating_user | int }}"
  title: "Título curto para a transação"
  wholesale_cost: "{{ wholesale_cost | float }}"
  invoice: true  # Cria automaticamente a transação na captura
  send_email: true  # Envia notificação ao cliente
```

5. Arredonde Valores Monetários

Sempre arredonde cálculos monetários para 2 casas decimais:

```
- name: Calcular custo com arredondamento
  set_fact:
    final_cost: "{{ (base_cost | float * multiplier | float) |
round(2) }}"
```

6. Lide com Métodos de Pagamento Ausentes

Verifique se o cliente possui um método de pagamento padrão antes de tentar a autorização:

```
- name: Obter método de pagamento padrão
  set_fact:
    payment_method_id: "{{ api_response_payment_methods.json |
json_query(query) }}"
  vars:
    query: "data[?is_default==`true`].payment_method_id | [0]"

- name: Verificar se o método de pagamento existe
  assert:
    that:
      - payment_method_id is defined
      - payment_method_id != ""
      - payment_method_id != None
    fail_msg: "Nenhum método de pagamento padrão encontrado para o
cliente {{ customer_id }}"
```

Padrões Comuns

Padrão 1: Provisionamento de Serviço Pago

Para serviços que requerem pagamento imediato:

1. Obter o método de pagamento do cliente

2. Autorizar pagamento
3. Provisionar serviço no OCS/CGRateS
4. Criar registro de serviço no CRM
5. Capturar pagamento
6. Em caso de falha: Liberar autorização e rollback

Veja `play_topup_charge_then_action.yaml` para exemplo completo.

Padrão 2: Serviço Gratuito com Taxa de Configuração

Para serviços que são gratuitos, mas têm um custo de configuração único:

1. Provisionar serviço
2. Criar registro de serviço
3. Adicionar transação de taxa de configuração diretamente (sem processamento de pagamento)
4. A taxa de configuração aparece na próxima fatura

Veja `play_simple_service.yaml` nas linhas 202-232 para exemplo completo.

Padrão 3: Topup/Addon Gratuito

Para topups gratuitos que não requerem pagamento:

1. Obter informações do serviço
2. Executar ação CGRateS
3. Atualizar datas do serviço
4. Nenhum pagamento ou criação de transação necessária

Padrão 4: Cobranças Recorrentes via ActionPlan

Para cobranças recorrentes automáticas:

1. Criar Ação com `*http_post` para o endpoint de provisionamento
2. Criar ActionPlan com temporização `*monthly`

3. Atribuir ActionPlan à conta
4. CGRateS chamará automaticamente o endpoint mensalmente
5. O playbook do endpoint lida com o processamento de pagamento

Solução de Problemas

Autorização Falha

Sintoma: O endpoint de autorização retorna erro

Causas Comuns:

- O método de pagamento não existe ou é inválido
- Fundos insuficientes
- Método de pagamento expirado
- Desconexão de ID do cliente

Solução: Verifique o status do método de pagamento e o saldo do cliente.

Captura Falha Após Provisionamento Bem-Sucedido

Sintoma: Serviço provisionado, mas captura de pagamento falha

Problema: Este é um estado crítico de falha - serviço está ativo, mas cliente não foi cobrado

Solução:

- A autorização pode ter expirado (tipicamente 7 dias)
- Verifique se a autorização ainda é válida antes de tentar a captura
- Implemente monitoramento para capturas falhadas
- Intervenção manual pode ser necessária

Transação Não Criada Após Captura

Sintoma: Pagamento capturado, mas nenhuma transação na cobrança

Causa: `invoice: true` não foi definido nos metadados da autorização

Solução: Ou:

- Defina `invoice: true` nos metadados da autorização, OU
- Crie manualmente a transação após a captura bem-sucedida

Cálculo Proporcional Incorreto

Sintoma: Cobranças proporcionais não correspondem aos valores esperados

Problemas Comuns:

- Erros de um em um na contagem de dias (incluir/excluir datas de início/fim)
- Mês errado usado para contagem de dias
- Erros de arredondamento

Solução:

- Use intervalos de datas inclusivos (inclua tanto o dia de início quanto o dia final)
- Sempre arredonde para 2 casas decimais
- Teste cálculos com valores conhecidos
- Documente quais datas estão incluídas nos cálculos

Reembolsos e Tratamento de Erros

Opções de Reembolso

O sistema de pagamento suporta dois tipos de reembolsos:

1. Reembolso ao Fonte de Pagamento - Dinheiro devolvido ao cartão original/PayPal

```
- name: Reembolsar pagamento ao cartão do cliente
  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. Crédito à Carteira - Saldo imediato para compras futuras (tratado automaticamente para cenários de erro)

Para falhas de provisionamento, o sistema credita automaticamente a carteira em vez de reembolsar o cartão para:

- Evitar taxas de reembolso
- Fornecer disponibilidade imediata para nova tentativa
- Melhorar a experiência do cliente

Consulte Opções de Reembolso para detalhes completos.

Suporte ao Fornecedor

O sistema de pagamento é **independente de fornecedor** e atualmente suporta:

- □ Stripe (cartões, ACH)
- □ PayPal (contas PayPal, cartões)

Novos fornecedores de pagamento (Square, Adyen, Braintree, etc.) podem ser adicionados sem alterar os playbooks.

Veja também:

- Arquitetura Modular - Como a abstração de fornecedor funciona
- Configuração do Fornecedor - Configuração e chaves da API do Stripe/PayPal

Documentação Relacionada

Guias Específicos de Playbook

- `concepts_ansible.md` - Padrões e estrutura geral de playbooks
- `concepts_provisioning.md` - Visão geral do sistema de provisionamento

Documentação do Sistema de Pagamento

- **Guia da API do Sistema de Pagamento** - Referência completa da API, roteamento de carteira, reembolsos
  - APIs de Métodos de Pagamento
  - APIs de Fluxo de Pagamento
  - Roteamento de Carteira em Primeiro Lugar
  - Opções de Reembolso
  - Arquitetura Modular
- `payments_transaction.md` - Gerenciamento de transações e cobranças manuais
- `payments_process.md` - Processamento de pagamentos e faturas
- `basics_payment.md` - Métodos de pagamento e faturamento do cliente

Controle de Acesso Baseado em Regras

Funções, Permissões e Usuários no OmniCRM

O OmniCRM utiliza **controle de acesso baseado em funções (RBAC)**: pessoas (Usuários) são atribuídas a uma ou mais Funções, e cada Função é um conjunto de Permissões. As Permissões são a menor unidade de acesso (por exemplo, view_customer, create_inventory). O acesso efetivo de um usuário é a **união** das permissões de todas as funções atribuídas.

Propósito

O RBAC permite:

1. **Proteção de Dados** — Usuários veem e fazem apenas o que estão autorizados.
2. **Ajuste Operacional** — Funções refletem funções de trabalho (Admin, Suporte, Finanças, Administrador do Cliente).
3. **Administração Simples** — Conceda acesso atribuindo funções; evite microgerenciamento por usuário.
4. **Isolamento de Inquilinos** — Permissões de “ver próprio ...” limitam a visibilidade aos dados do cliente/inquilino de um usuário.

Como Usuários, Funções e Permissões se Encaixam

- **Usuários** — Pessoas reais que fazem login no OmniCRM.
- **Permissões** — Capacidades atômicas (por exemplo, view_customer, delete_product).
- **Funções** — Conjuntos nomeados de permissões (por exemplo, *Admin*, *Suporte*, *Finanças*).
- **Atribuição** — Usuários recebem uma ou mais funções; as permissões se agregam.

A autenticação prova *quem você é* (JWT, chave da API ou IP na lista branca). A autorização (funções/permissões) decide *o que você pode fazer*.

Gerenciando Usuários

O sistema de gerenciamento de usuários do OmniCRM permite que administradores criem e gerenciem usuários da equipe (administradores, agentes de atendimento ao cliente), visualizem e modifiquem funções de usuários, redefinam senhas, gerenciem autenticação de dois fatores e controlem o acesso dos usuários.

Tipos de Usuários

Usuários Clientes - Criados via autoinscrição ou por administradores. Atribuídos automaticamente à função "Cliente". Esses usuários acessam o portal de autoatendimento para gerenciar seus serviços, visualizar uso, pagar faturas, etc.

Usuários da Equipe - Criados por administradores com permissões apropriadas. Podem ser atribuídos funções como Admin, Suporte, Finanças, etc. Esses usuários acessam a interface do CRM para gerenciar clientes, provisionar serviços, lidar com faturamento, etc.

Usuários Administrativos - Usuários com a permissão `admin`. Têm acesso total ao sistema, incluindo gerenciamento de usuários, gerenciamento de funções e todos os endpoints protegidos.

O usuário administrativo inicial é criado pela equipe do Omnitouch quando o sistema é implantado.

Adicionando Novos Usuários (Administradores e CSAs)

Os administradores podem criar novos usuários da equipe através da interface da Web ou da API.

Via Interface da Web

1. **Navegue até Usuários e Funções** - Acesse a interface de gerenciamento de usuários pelo menu de administração
2. **Clique em "Adicionar Usuário"** - Abre o formulário de criação de usuário

3. **Preencha os Detalhes do Usuário:**
    - **Nome de Usuário** - Nome de usuário exclusivo para login (obrigatório)
    - **Email** - Endereço de email do usuário (obrigatório, deve ser único)

    - **Senha** - Senha temporária (obrigatória, o usuário deve alterá-la no primeiro login)
    - **Primeiro Nome** - Primeiro nome do usuário (obrigatório)
    - **Nome do Meio** - Nome do meio do usuário (opcional)
    - **Último Nome** - Último nome do usuário (obrigatório)
    - **Número de Telefone** - Número de telefone de contato (opcional)
    - **Função** - Selecione uma ou mais funções para atribuir (obrigatório)
    - **Contato do Cliente** - Opcionalmente vincule o usuário a um registro de contato do cliente (para usuários clientes)
4. **Clique em "Criar Usuário"** - O usuário é criado e pode fazer login imediatamente com as credenciais fornecidas
5. **Usuário recebe notificação** - Opcionalmente, envie um email de boas-vindas com instruções de login

Melhores Práticas:

- Use uma senha temporária como `TempP@ssw0rd!` e exija que o usuário a altere no primeiro login
- Atribua funções apropriadas com base na função de trabalho (veja os Designs de Função Típicos abaixo)
- Ative 2FA para todos os funcionários administrativos e de suporte
- Vincule usuários clientes ao seu registro de contato do cliente para um escopo de dados adequado

Via API

Crie um usuário programaticamente:

Endpoint: `POST /auth/users`

Permissão Necessária: `admin`

Corpo da Solicitação:

```
{
  "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"
}
```

Resposta:

```
{
  "id": 123,
  "username": "john.smith",
  "email": "john.smith@company.com",
  "first_name": "John",
  "last_name": "Smith",
  "roles": ["Support"],
  "created": "2025-01-04T10:30:00Z"
}
```

Atribuindo Múltiplas Funções:

Os usuários podem ter várias funções. As permissões são aditivas (união de todas as permissões de função atribuídas).

Para atribuir várias funções, inclua-as na solicitação:

```
{
  "username": "jane.doe",
  "email": "jane.doe@company.com",
  "password": "TempP@ssw0rd!",
  "first_name": "Jane",
  "last_name": "Doe",
  "role": "Support,Finance"
}
```

Ou use o endpoint de atribuição de função após a criação do usuário:

```
POST /auth/roles/{role_id}/users/{user_id}
```

Visualizando e Pesquisando Usuários

Listar Todos os Usuários (Admin):

`GET /auth/users`

Retorna uma lista paginada de todos os usuários com suas funções e informações básicas.

Pesquisar Usuários:

`GET /auth/users/search?search={query}&filters={"role":["Support"]}&page=1&per_page=50`

Filtrar por:

- Nome da função
- Domínio de email
- Status ativo/excluído
- Status de 2FA ativado
- Data do último login

Obter Usuário Específico:

`GET /auth/users/{user_id}`

Retorna detalhes completos do usuário, incluindo:

- Informações pessoais
- Funções atribuídas e permissões efetivas
- Status de 2FA

- Último login e informações de sessão
- Contato do cliente vinculado (se aplicável)

Criando e Gerenciando Funções

As funções são coleções de permissões que podem ser atribuídas a usuários. Em vez de atribuir permissões individualmente a cada usuário, você cria funções que agrupam permissões relacionadas e atribui essas funções aos usuários.

Criando uma Nova Função

Via Interface da Web:

1. Navegue até **Usuários e Funções** → aba **Funções**
2. Clique em **"Criar Função"**
3. Insira os detalhes da função:
   - **Nome** - Nome curto e descritivo (por exemplo, "Tier2_Support")
   - **Descrição** - Explique o propósito e as responsabilidades da função
4. Clique em **"Criar"**
5. A função é criada sem permissões; adicione permissões na próxima etapa

Via API:

Endpoint: `POST /auth/roles`

Permissão Necessária: `admin`

Solicitação:

```
{
  "name": "Tier2_Support",
  "description": "Equipe de suporte de nível 2 com acesso elevado
ao provisionamento"
}
```

Resposta:

```
{
  "id": 45,
  "name": "Tier2_Support",
  "description": "Equipe de suporte de nível 2 com acesso elevado
ao provisionamento",
  "permissions": [],
  "users": []
}
```


Adicionando Permissões a uma Função

Uma vez que uma função é criada, atribua permissões para definir o que os usuários com essa função podem fazer.

Via Interface da Web:

1. Navegue até **Usuários e Funções** → aba **Funções**
2. Clique no nome da função para visualizar os detalhes
3. Na seção **Permissões**, clique em **"Adicionar Permissão"**
4. Selecione uma ou mais permissões da lista
5. Clique em **"Adicionar"** - As permissões são imediatamente atribuídas

Via API:

Endpoint: `POST /auth/roles/{role_id}/permissions`

Solicitação:

```
{
  "permission_id": 123
}
```

Ou adicione várias permissões:

```
{
  "permission_ids": [123, 124, 125]
}
```

Exemplo: Criando uma Função "Especialista em Provisionamento"

Essa função pode visualizar clientes, gerenciar serviços e provisionar:

1. Crie a função:

```
POST /auth/roles
{
  "name": "Provisioning_Specialist",
  "description": "Pode provisionar serviços e gerenciar
serviços de clientes"
}
```

2. Adicione permissões:

```
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
  ]
}
```

Removendo Permissões de uma Função

Via Interface da Web:

1. Navegue até os detalhes da função
2. Na lista de **Permissões**, clique no **"X"** ou no botão **"Remover"** ao lado da permissão
3. Confirme a remoção

Via API:

Endpoint: `DELETE /auth/roles/{role_id}/permissions/{permission_id}`

Exemplo:

```
DELETE /auth/roles/45/permissions/31
```

Isso remove a permissão `create_provision` da função.

Editando Detalhes da Função

Atualize o nome ou a descrição da função:

Via Interface da Web:

1. Navegue até **Usuários e Funções** → aba **Funções**
2. Clique na função para editar
3. Modifique o nome ou a descrição da função
4. Clique em **"Salvar"**

Via API:

Endpoint: `PUT /auth/roles/{role_id}`

```
{
  "name": "Senior_Support",
  "description": "Equipe de suporte sênior com acesso total ao cliente"
}
```

Excluindo uma Função

Aviso: Excluir uma função a remove de todos os usuários atribuídos. Certifique-se de que os usuários tenham funções alternativas ou perderão o acesso.

Via API:

`DELETE /auth/roles/{role_id}`

Melhor Prática: Em vez de excluir, considere arquivar ou renomear funções que não são mais necessárias.

Atribuindo Funções a Usuários

Durante a Criação do Usuário:

Inclua a função na solicitação de criação do usuário (veja "Adicionando Novos Usuários" acima).

Para Usuários Existentes:

Via Interface da Web:

1. Navegue até **Usuários e Funções** → aba **Usuários**
2. Clique no usuário para editar
3. Na seção **Funções**, selecione/deselecione funções
4. Clique em **"Salvar"**

Via API:

Atualize as funções do usuário:

Endpoint: `PUT /auth/users/{user_id}`

```
{
  "role": "Support,Finance"
}
```

Ou atribua uma única função a um usuário via endpoint de função:

Endpoint: `POST /auth/roles/{role_id}/users/{user_id}`

Visualizando Atribuições de Funções

Todos os usuários em uma função:

`GET /auth/roles/{role_id}/users`

Retorna uma lista de todos os usuários atribuídos a essa função.

Todas as funções de um usuário:

`GET /auth/users/{user_id}`

A resposta inclui um array `roles` com todas as funções atribuídas.

Gerenciando Senhas de Usuário

O OmniCRM fornece vários métodos para gerenciamento de senhas, dependendo do contexto.

Redefinição de Senha pelo Usuário

Usuários que esqueceram sua senha podem redefini-la por conta própria através da página de login:

1. **Clique em "Esqueceu a Senha"** na página de login
2. **Insira o endereço de email** - O sistema envia um email de redefinição de senha
3. **Verifique o email** - O email contém um link seguro de redefinição com token (válido por 1 hora)
4. **Clique no link** - Abre o formulário de redefinição de senha
5. **Insira a nova senha** - Deve atender aos requisitos de complexidade da senha:
   - Mínimo de 8 caracteres
   - Pelo menos uma letra maiúscula
   - Pelo menos uma letra minúscula
   - Pelo menos um número
   - Pelo menos um caractere especial
6. **Enviar** - A senha é imediatamente atualizada; o usuário pode fazer login com a nova senha

Fluxo da API:

1. **Solicitar redefinição:**

   Endpoint: `POST /auth/forgot_password`

```
{
  "email": "user@example.com"
}
```

O sistema gera um token de redefinição e envia um email.

2. **Redefinir com token:**

   Endpoint: `POST /auth/reset_password`

```
{
  "token": "abc123...",
  "new_password": "NewSecureP@ssw0rd!"
}
```

Redefinição de Senha pelo Administrador

Os administradores podem redefinir a senha de um usuário sem exigir verificação por email. Isso define uma senha temporária que o usuário deve alterar no próximo login.

Via Interface da Web:

1. Navegue até **Usuários e Funções → Usuários**
2. Encontre o usuário e clique no botão **"Redefinir Senha"**
3. Insira uma senha temporária
4. Clique em **"Redefinir"**
5. Notifique o usuário sobre sua senha temporária (via canal seguro)
6. O usuário deve alterar a senha no próximo login

Via API:

Endpoint: `POST /auth/users/{user_id}/admin_reset_password`

Permissão Necessária: `admin`

Solicitação:

```
{
  "new_password": "TempP@ssw0rd!",
  "force_change": true
}
```

Parâmetros:

- `new_password` - Senha temporária a ser definida
- `force_change` (opcional) - Se verdadeiro, o usuário deve alterar a senha no próximo login

Usuário Altera Sua Própria Senha

Usuários autenticados podem alterar sua própria senha a partir de seu perfil:

Endpoint: `POST /auth/change_password`

Solicitação:

```
{
  "current_password": "OldP@ssw0rd!",
  "new_password": "NewSecureP@ssw0rd!"
}
```

O sistema valida a senha atual antes de permitir a alteração.

Segurança da Senha

- As senhas são criptografadas usando bcrypt (segurança do werkzeug)
- Nunca armazenadas em texto simples
- Tokens de redefinição expiram após 1 hora
- Tentativas de login falhadas podem acionar bloqueio da conta (configurável)
- O rastreamento do histórico de senhas impede reutilização (configurável)
- Requisitos de complexidade aplicados

Gerenciando Autenticação de Dois Fatores (2FA)

O OmniCRM suporta autenticação de dois fatores baseada em TOTP para maior segurança. Os administradores podem ativar, desativar e redefinir 2FA para usuários.

Ativando 2FA para um Usuário

Via Interface da Web:

1. Navegue até **Usuários e Funções → Usuários**
2. Clique no usuário para visualizar os detalhes
3. Na seção **Segurança**, clique em **"Ativar 2FA"**
4. O sistema gera:
   - Segredo TOTP (código QR exibido)
   - 10 códigos de backup (uso único)
5. O usuário escaneia o código QR com o aplicativo autenticador (Google Authenticator, Authy, etc.)

6. O usuário insere o código de verificação do aplicativo para confirmar a configuração
7. O usuário salva os códigos de backup em local seguro
8. 2FA agora está ativado; necessário para todos os futuros logins

Via API:

1. **Gerar segredo TOTP:**

   Endpoint: `POST /2fa/enable/user/{user_id}`

   Resposta:

```
{
  "totp_secret": "JBSWY3DPEHPK3PXP",
  "qr_code_url": "otpauth://totp/OmniCRM:user@example.com?
secret=JBSWY3DPEHPK3PXP&issuer=OmniCRM",
  "backup_codes": [
    "12345678",
    "23456789",
    "34567890",
    ...
  ]
}
```

2. **Verificar configuração:**

   Endpoint: `POST /2fa/verify-setup/user/{user_id}`

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

   Retorna `{"verified": true}` em caso de sucesso.

Fluxo de Login 2FA

Uma vez que 2FA está ativado, o processo de login muda:

1. O usuário insere nome de usuário e senha
2. O sistema valida as credenciais
3. Se válidas, solicita o código 2FA
4. O usuário insere o código do aplicativo autenticador OU código de backup
5. O sistema verifica o código
6. Em caso de sucesso, o usuário é logado

Códigos de Backup:

- 10 códigos gerados durante a configuração do 2FA
- Uso único apenas (consumidos após o uso)
- Usados se o aplicativo autenticador não estiver disponível
- Podem ser regenerados pelo usuário ou administrador

Verificando o Código 2FA

Endpoint: `POST /2fa/verify/user/{user_id}`

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

Aceita ambos:

- **Código TOTP** (6 dígitos do aplicativo autenticador)
- **Código de backup** (8 dígitos da lista de códigos de backup)

Regenerando Códigos de Backup

Se um usuário esgotar os códigos de backup ou perdê-los, gere novos:

Via Interface da Web:

1. Navegue até os detalhes do usuário
2. Clique em **"Regenerar Códigos de Backup"**
3. Exiba ou envie novos códigos para o usuário
4. Códigos antigos são invalidados

Via API:

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

Resposta:

```
{
  "backup_codes": [
    "98765432",
    "87654321",
    "76543210",
    ...
  ]
}
```

Redefinição de 2FA pelo Administrador

Se um usuário perder o acesso ao seu aplicativo autenticador e todos os códigos de backup, um administrador pode desativar e reativar o 2FA.

Via Interface da Web:

1. Navegue até **Usuários e Funções → Usuários**
2. Clique no usuário
3. Clique no botão **"Redefinir 2FA"**
4. Confirme a redefinição
5. 2FA é desativado; o usuário pode fazer login apenas com a senha

6. Oriente o usuário a configurar o 2FA novamente com um novo segredo

Via API:

Endpoint: `POST /2fa/admin/disable/user/{user_id}`

Permissão Necessária: `admin`

Isso desativa completamente o 2FA para o usuário:

- Segredo TOTP limpo
- Códigos de backup limpos
- Flag `is_2fa_enabled` definida como falsa

O usuário pode então reativar o 2FA para obter um novo segredo e códigos de backup.

Redefinição de 2FA pelo Usuário (Novo Dispositivo)

Se um usuário obtiver um novo dispositivo, mas ainda tiver acesso aos códigos de backup:

Endpoint: `POST /2fa/reset-for-new-device/user/{user_id}`

```
{
  "backup_code": "12345678"
}
```

O sistema valida o código de backup, em seguida, gera um novo segredo TOTP e códigos de backup. O usuário pode configurar o aplicativo autenticador em seu novo dispositivo.

Melhores Práticas de 2FA

- **Exigir 2FA para todos os funcionários administrativos e de suporte**
- **Armazenar códigos de backup com segurança** (gerenciador de senhas ou nota segura)
- **Regenerar códigos de backup** após usar vários
- **Usar aplicativos autenticadores respeitáveis** (Google Authenticator, Authy, Microsoft Authenticator)
- **Documentar procedimentos de redefinição de 2FA** para a equipe de suporte
- **Auditar o uso de 2FA** - monitorar quais usuários têm 2FA ativado

Atualizando Informações do Usuário

Os administradores podem atualizar os detalhes do usuário a qualquer momento.

Via Interface da Web:

1. Navegue até **Usuários e Funções → Usuários**
2. Clique no usuário para editar
3. Modifique quaisquer campos editáveis:
    - Primeiro nome, nome do meio, sobrenome
    - Endereço de email (exige verificação)
    - Número de telefone
    - Funções
    - Vínculo de contato do cliente
4. Clique em **"Salvar"**

Via API:

Endpoint: `PUT /auth/users/{user_id}`

```
{
  "first_name": "Jane",
  "last_name": "Doe-Smith",
  "email": "jane.doesmith@newcompany.com",
  "phone_number": "+61498765432",
  "role": "Support,Finance"
}
```

Alterações de Email:

Quando o email é alterado, o novo email é marcado como pendente até ser verificado:

- O campo `pending_email` armazena o novo email
- Email de verificação enviado para o novo endereço
- O usuário clica no link para verificar
- O campo `email` é atualizado para o novo valor
- A flag `email_verified` é definida como verdadeira

Excluindo Usuários

O OmniCRM utiliza **exclusões suaves** para usuários - eles são marcados como excluídos, mas não removidos do banco de dados. Isso preserva trilhas de auditoria e dados históricos.

Excluindo um Usuário

Via Interface da Web:

1. Navegue até **Usuários e Funções → Usuários**
2. Encontre o usuário a ser excluído
3. Clique no botão **"Excluir"**
4. Confirme a exclusão
5. O usuário é imediatamente desconectado e não pode fazer login novamente

Via API:

Endpoint: `DELETE /auth/users/{user_id}`

Permissão Necessária: `admin`

O que Acontece:

- A flag `deleted` é definida como `True`
- O timestamp `deleted_at` é registrado
- O usuário não pode fazer login
- Todas as sessões ativas são invalidadas
- O usuário ainda aparece em logs de auditoria e registros históricos
- Dados vinculados (contatos de clientes, atividades) preservados

Visualizando Usuários Excluídos

Filtrar por usuários excluídos:

`GET /auth/users/search?filters={"deleted":[true]}`

Restaurando um Usuário Excluído

Se um usuário foi excluído por engano, os administradores podem restaurá-lo:

Endpoint: `PUT /auth/users/{user_id}`

```
{
  "deleted": false
}
```

Isso limpa a flag `deleted` e permite que o usuário faça login novamente.

Nota: A senha do usuário permanece inalterada, para que ele possa usar sua senha anterior.

Excluindo Permanentemente um Usuário

Aviso: Isso é irreversível e remove todos os dados do usuário do banco de dados.

Não exposto via UI. Disponível apenas através do acesso direto ao banco de dados por razões de conformidade (por exemplo, solicitações de exclusão de dados GDPR).

Melhores Práticas para Exclusão de Usuários

- **Excluir suavemente por padrão** - Preserva trilhas de auditoria
- **Documentar razões para exclusão** - Adicione uma nota no log de atividades antes de excluir
- **Transferir propriedade** - Reatribuir tickets e tarefas abertos do usuário antes de excluir
- **Revisar acesso** - Garantir que nenhum processo crítico dependa do usuário
- **Arquivar dados** - Exportar o histórico de trabalho do usuário, se necessário
- **Notificar equipes relevantes** - Informar gerentes/colegas sobre a exclusão

Catálogo de Permissões

As permissões geralmente seguem padrões CRUD:

- view_* — ler/navegar
- create_* — criar/adicionar
- update_* — editar/modificar
- delete_* — excluir/remover

Algumas entidades também incluem variantes **“ver próprio …”** que restringem a visibilidade aos dados do cliente/inquilino do usuário atual.

Global / Administrativa

- `admin` — Acesso administrativo total (gerenciar usuários, funções e permissões; acessar todos os endpoints protegidos).
- `can_impersonate` — Agir temporariamente como outro usuário (auditado; para suporte/resolução de problemas).
- `grafana_access` — Acesso aos painéis de análise do Grafana para criar e visualizar relatórios e visualizações personalizadas. Veja a seção de Análise do Grafana abaixo para detalhes.

Clientes e Registros Relacionados

- **Cliente**
  - view_customer, create_customer, update_customer, delete_customer
  - **Escopo próprio:** *ver próprio cliente*
- **Site do Cliente**
  - view_customer_site, create_customer_site, update_customer_site, delete_customer_site
  - **Escopo próprio:** *ver próprio site do cliente*
- **Contato do Cliente**

    - view_customer_contact, create_customer_contact, update_customer_contact, delete_customer_contact
    - **Escopo próprio:** *ver próprio contato do cliente*
- **Atributo do Cliente** (veja `Atributos do Cliente <administration_attributes>`)
    - view_customer_attribute, create_customer_attribute, update_customer_attribute, delete_customer_attribute
    - **Escopo próprio:** *ver próprio atributo do cliente*
- **Tag do Cliente** (veja `Tags do Cliente <administration_tags>`)
    - view_customer_tag, create_customer_tag, update_customer_tag, delete_customer_tag
    - **Escopo próprio:** *ver própria tag do cliente*
- **Serviço do Cliente**
    - view_customer_service, create_customer_service, update_customer_service, delete_customer_service
    - **Escopo próprio:** *ver próprio serviço do cliente*
- **Atividade do Cliente**
    - view_customer_activity, create_customer_activity, update_customer_activity, delete_customer_activity
    - **Escopo próprio:** *ver própria atividade do cliente*

Faturamento

- **Cartão Stripe**
    - `view_customer_stripe_card`, `create_customer_stripe_card`, `update_customer_stripe_card`, `delete_customer_stripe_card`
    - **Escopo próprio:** *ver próprio cartão stripe do cliente*
- **Transações**
    - `view_customer_transaction`, `create_customer_transaction`, `update_customer_transaction`, `delete_customer_transaction`
    - **Escopo próprio:** *ver própria transação do cliente*
- **Faturas**
    - `view_customer_invoice`, `create_customer_invoice`, `update_customer_invoice`, `delete_customer_invoice`

- **Escopo próprio:** *ver própria fatura do cliente*

Comunicações

- view_communication, create_communication, update_communication, delete_communication
- **Escopo próprio:** *ver própria comunicação*

Inventário e Modelos

- **Inventário**
  - `view_inventory`, `create_inventory`, `update_inventory`, `delete_inventory`
  - **Escopo próprio:** *ver próprio inventário*
- **Modelo de Inventário**
  - `view_inventory_template`, `create_inventory_template`, `update_inventory_template`, `delete_inventory_template`
  - **Escopo próprio:** *ver próprio modelo de inventário*

Produtos

- `view_product`, `create_product`, `update_product`, `delete_product`

Transmissão de Célula (CBC)

- `view_cbc_message`, `create_cbc_message`, `update_cbc_message`, `delete_cbc_message`

Provisionamento

- **Provisionar**
  - `view_provision`, `create_provision`, `update_provision`, `delete_provision`
  - **Escopo próprio:** *ver próprio provisionamento*
- **Evento de Provisionamento**

- `view_provision_event`, `create_provision_event`, `update_provision_event`, `delete_provision_event`

Acesso “Ver Próprio”

As permissões “ver próprio …” restringem leituras (e opcionalmente edições, onde implementadas) a dados associados ao **próprio cliente/inquilino** do usuário. Por exemplo, uma função *Administrador do Cliente* pode gerenciar os contatos, sites, faturas e serviços de *seu* inquilino, mas não pode ver outros inquilinos.

Designs de Função Típicos

Função	Permissões Típicas	Notas
Administrador do Sistema	`admin`, opcionalmente `can_impersonate`; além de CRUD amplo conforme necessário	Controle total sobre usuários/funções/permissões
Suporte	`view_customer`, `view_customer_service`, `view_communication`, `view_provision`; atualizações opcionais	Adicione `can_impersonate` se permitido
Finanças	`view_customer_invoice`, `view_customer_transaction`, `view_product`; opcional `create_customer_invoice`	Pesado em leitura; escrita limitada
Administrador do Cliente (inquilino)	"ver próprio ..." em clientes, sites, contatos, serviços, inventário, faturas, transações, comunicações, provisionamento	Gerenciamento com escopo de inquilino
Auditor Apenas Leitura	Ampla `view_*` apenas	Sem criar/atualizar/excluir

Exemplos de Funções e Permissões Incluídas (resumo)

Gerenciando Funções e Permissões via API

Todos os endpoints requerem permissão `admin`.

Listar permissões

Endpoint: `GET /auth/permissions`

Criar uma permissão (raro)

Endpoint: `POST /auth/permissions`

Corpo da Solicitação:

```
{
    "name": "view_example",
    "description": "Acesso somente leitura a objetos de exemplo"
}
```

Listar funções

Endpoint: `GET /auth/roles`

Criar uma função

Endpoint: `POST /auth/roles`

Corpo da Solicitação:

```
{
"name": "Support",
"description": "Equipe de suporte de nível 1"
}
```

Adicionar uma permissão a uma função

Endpoint: `POST /auth/roles/{role_id}/permissions`

Corpo da Solicitação:

```
{
  "permission_id": 123
}
```

Remover uma permissão de uma função

Endpoint: `DELETE /auth/roles/{role_id}/permissions/{permission_id}`

Atribuindo Funções a Usuários

Criar um usuário com função

Endpoint: `POST /auth/users`

Corpo da Solicitação:

```
{
  "username": "sara",
  "email": "sara@example.com",
  "password": "TempP@ssw0rd!",
  "first_name": "Sara",
  "last_name": "Ng",
  "phone_number": "+61...",
  "role": "Support"
}
```

Atualizar a função de um usuário

Endpoint: `PUT /auth/users/{user_id}`

Corpo da Solicitação:

```
{
  "role": "Finance"
}
```

Listar usuários (somente Admin)

Endpoint: `GET /auth/users`

Impersonação (Controlada)

- **Necessário:** `can_impersonate` ou `admin`

Iniciar impersonação

Endpoint: `POST /auth/impersonate`

Corpo da Solicitação:

```
{ "user_id": 42 }
```

Parar impersonação

Endpoint: `POST /auth/stop_impersonation`

Melhores Práticas

- **Privilégio mínimo primeiro.** Comece com funções mínimas; adicione permissões conforme necessário.
- **Prefira "ver próprio ...".** Use permissões com escopo de inquilino para funções voltadas para o cliente.
- **Mantenha funções estáveis.** Atualize as permissões da função quando as funcionalidades mudarem—não edite cada usuário.
- **Audite regularmente.** Revise quem tem `admin` ou `can_impersonate`.

Análise e Painéis do Grafana

O OmniCRM se integra ao Grafana para fornecer poderosas capacidades de análise e visualização. Usuários com a permissão `grafana_access` podem acessar o Grafana para criar painéis, relatórios e visualizações personalizadas.

Acessando o Grafana

Se sua conta de usuário tiver sido concedida a permissão `grafana_access`, você pode acessar o Grafana navegando até `/grafana` em seu navegador. Você será autenticado automaticamente usando suas credenciais do OmniCRM - não é necessário um login separado no Grafana.

A partir da interface do Grafana, você pode:

- Criar painéis personalizados com gráficos, tabelas e tabelas
- Construir consultas complexas para analisar dados de clientes, tendências de receita e métricas de serviço
- Configurar alertas com base em indicadores-chave de desempenho
- Compartilhar painéis com outros membros da equipe

Incorporando Painéis no OmniCRM

Uma vez que você tenha criado painéis no Grafana, pode incorporá-los diretamente na interface do OmniCRM. Isso permite que você visualize suas métricas mais importantes sem sair do CRM, mantendo a navegação do OmniCRM intacta.

Para incorporar um painel:

1. **Crie seu painel no Grafana** - Navegue até `/grafana` e projete seu painel com as visualizações que você precisa.

2. **Anote o ID do painel** - Ao visualizar seu painel no Grafana, verifique a URL. Ela terá a aparência de `/grafana/d/abc123/nome-do-painel`. O ID é a parte após `/d/` (neste exemplo, `abc123`).

3. **Configure o painel em seu ambiente** - Entre em contato com seu administrador de sistema para adicionar o ID do painel e um nome de exibição à configuração do sistema. Os painéis são configurados através de variáveis de ambiente.

4. **Acesse a partir da barra lateral** - Uma vez configurado, seu painel aparecerá automaticamente na barra lateral do OmniCRM sob o menu

"Painéis". Você pode clicar nele para visualizar o painel completo enquanto mantém a navegação do OmniCRM visível.

Configuração do Painel

Os painéis são configurados pelo seu administrador de sistema usando variáveis de ambiente. Cada painel requer:

- **ID do Painel** - O identificador exclusivo da URL do Grafana
- **Rótulo do Painel** - O nome amigável exibido na barra lateral do OmniCRM

Vários painéis podem ser configurados e aparecerão como itens de menu na ordem em que foram definidos. Veja a documentação de Configuração de Administração para detalhes sobre a configuração de variáveis de ambiente.

Quem Pode Acessar o Grafana?

Apenas usuários com a permissão `grafana_access` podem acessar o Grafana e visualizar painéis incorporados. Essa permissão é geralmente concedida a:

- Administradores que precisam de visibilidade total sobre métricas do sistema
- Analistas de negócios que criam relatórios e visualizações
- Equipes financeiras que rastreiam métricas de receita e faturamento
- Equipes de operações que monitoram o desempenho do serviço

FAQ

Um usuário pode ter várias funções? Sim. As permissões são aditivas.

Preciso de permissões personalizadas? Geralmente não. O catálogo embutido cobre a maioria das necessidades.

Como as regras "ver próprio ..." sabem o que é meu? Elas avaliam a ligação entre seu usuário/contato e seu cliente (inquilino).

Posso criar meus próprios painéis do Grafana? Sim, se você tiver a permissão `grafana_access`. Navegue até `/grafana` para acessar a interface completa do Grafana.

Portal de Autoatendimento

O Portal de Autoatendimento é uma interface voltada para o cliente que permite aos usuários finais gerenciar suas próprias contas, visualizar uso, pagar faturas e modificar serviços sem a necessidade de assistência da equipe de atendimento ao cliente.

Métodos de Acesso:

- Login direto via credenciais do cliente
- `Impersonação de Funcionário <customer_care>` para solução de problemas (de Clientes → Contatos → "Login como Usuário")

Veja também: `Atendimento ao Cliente <customer_care>` para detalhes de impersonação, `Fluxos de Autenticação <authentication_flows>` para o processo de login.

Propósito

O Portal de Autoatendimento fornece aos clientes:

1. **Gerenciamento de Conta** - Visualizar e atualizar informações pessoais, contatos e endereços
2. **Visão Geral dos Serviços** - Ver todos os serviços ativos, uso e datas de expiração
3. **Rastreamento de Uso** - Monitorar dados, voz, SMS e saldos monetários
4. **Acesso à Faturamento** - Visualizar e pagar faturas, gerenciar métodos de pagamento
5. **Modificações de Serviço** - Adicionar recargas, comprar complementos, modificar serviços
6. **Disponibilidade 24/7** - Acessar informações da conta a qualquer momento sem ligar para o suporte

Visão Geral do Portal

Quando os clientes fazem login no Portal de Autoatendimento, eles veem um painel personalizado com:

Seções de Navegação:

- **Painel** - Visão rápida dos serviços e atividades recentes
- **Conta** - Detalhes pessoais, contatos, locais
- **Serviços** - Lista de todos os serviços com status e detalhes
- **Uso** - Informações de consumo e expiração de saldo
- **Faturamento** - Faturas, transações, métodos de pagamento
- **Recarga** - Comprar crédito de dados, voz, SMS ou crédito monetário

Detalhes da Conta

A seção de Conta exibe informações do cliente e permite atualizações limitadas de autoatendimento.

Campos Editáveis:

Os clientes podem atualizar:

- Endereço de e-mail (requer verificação)
- Número de telefone
- Senha
- Preferências de notificação

Informações Somente para Visualização:

- ID do Cliente
- Data de criação da conta
- Tipo de cliente (Individual/Empresa)
- Locais (endereços)
- Contatos vinculados

Atualizando Informações da Conta:

1. Navegue até **Conta** → **Detalhes**
2. Clique em **"Editar"** ao lado do campo a ser atualizado
3. Insira novas informações
4. Clique em **"Salvar Alterações"**
5. Para alterações de e-mail, verifique via link enviado para o novo endereço

Recursos de Segurança:

- Alterações de senha requerem a senha atual
- Alterações de e-mail requerem verificação
- Atividade registrada para auditoria
- Configurações de 2FA (se habilitadas)

Visão Geral dos Serviços

A seção de Serviços mostra todos os serviços ativos e inativos para o cliente.

Exibição do Cartão de Serviço:

Cada serviço exibe:

- **Nome do Serviço** - Identificador legível por humanos (por exemplo, "Móvel - +44 7700 900123")
- **Produto** - Nome do plano ou produto
- **Status** - Ativo, Suspenso, Expirado, Cancelado
- **Data de Criação** - Data de ativação do serviço
- **Data de Expiração** - Quando o serviço expira (se aplicável)
- **Custo Mensal** - Cobrança recorrente
- **Renovação Automática** - Indicador Habilitado/Desabilitado

Ações do Serviço:

- **Ver Uso** - Ver consumo de saldo (Dados, Voz, SMS, Monetário)
- **Recarga** - Adicionar crédito ou dados
- **Complementos** - Comprar recursos adicionais
- **Modificar** - Alterar parâmetros do serviço (se permitido)
- **Ver Detalhes** - Ver configuração completa do serviço

Indicadores de Status do Serviço:

- □ **Ativo** - O serviço está operacional
- □ **Expirando em Breve** - Renova ou expira em <7 dias
- □ **Suspenso** - Serviço temporariamente desativado (problema de pagamento, suspensão manual)
- ● **Expirado** - Serviço não está mais ativo

Rastreamento de Uso

Os clientes podem monitorar seu uso em todos os tipos de saldo em tempo real.

Uso de Dados

Visualize o consumo de dados com detalhamentos por bucket e expiração.

Exibição do Uso de Dados:

- **Saldo Total** - Todos os buckets de dados combinados
- **Usado Neste Período** - Consumo desde a última renovação

- **Barra de Progresso** - Representação visual do consumo
- **Informações de Expiração** - Quando cada bucket expira
- **Quebra de Bucket** - Múltiplos buckets de dados com ordem de prioridade

Prioridade de Bucket:

Quando existem múltiplos buckets de dados (por exemplo, plano base + recargas), eles consomem em ordem de peso:

- **Peso 10** - Consumido primeiro (tipicamente dados promocionais/bônus)
- **Peso 20** - Consumido segundo (tipicamente dados de recarga)
- **Peso 30** - Consumido por último (tipicamente dados do plano base)

Uso de Voz

Acompanhe os minutos de chamadas restantes em todos os buckets de voz.

Exibição do Uso de Voz:

- **Minutos Restantes** - Saldo total de voz
- **Minutos Usados** - Consumo neste período
- **Histórico de Chamadas** - Chamadas recentes (se habilitado)

- **Datas de Expiração** - Quando os buckets de voz expiram
- **Minutos Internacionais** - Rastreamento separado (se aplicável)

Quebra de Uso:

- Chamadas On-Net - Chamadas dentro da mesma rede
- Chamadas Off-Net - Chamadas para outras redes
- Chamadas Internacionais - Chamadas fora do país
- Números Premium - Números de tarifa especial

Uso de SMS

Monitore as permissões e o consumo de mensagens SMS.

Exibição do Uso de SMS:

- **Mensagens Restantes** - Saldo de SMS
- **Usadas Neste Mês** - Mensagens enviadas
- **MMS Incluído** - Se MMS conta para o saldo
- **SMS Internacionais** - Rastreamento separado (se aplicável)

Saldo Monetário

Visualize o saldo de crédito pré-pago para serviços pay-as-you-go.

Exibição Monetária:

- **Saldo Atual** - Crédito disponível
- **Última Recarga** - Valor e data da recarga mais recente
- **Data de Expiração** - Quando o saldo expira (se aplicável)
- **Recarga Automática** - Status Habilitado/Desabilitado

Gerenciamento de Faturamento

Os clientes podem visualizar faturas, transações e gerenciar métodos de pagamento.

Faturas

Visualize e pague faturas pendentes diretamente do portal.

Exibição da Lista de Faturas:

- **Número da Fatura** - Identificador único
- **Data** - Data de criação da fatura
- **Data de Vencimento** - Prazo de pagamento
- **Valor** - Valor total da fatura
- **Status** - Pago, Não Pago, Atrasado
- **Ações** - Baixar PDF, Pagar Online

Pagando uma Fatura:

1. Navegue até **Faturamento → Faturas**
2. Encontre a fatura não paga na lista
3. Clique no botão **"Pagar Agora"**
4. Selecione o método de pagamento (cartão salvo ou novo cartão)
5. Confirme o pagamento
6. Receba um e-mail de confirmação

Baixando Faturas:

1. Clique no ícone **"Baixar"** ao lado da fatura
2. PDF é baixado com todos os detalhes da fatura
3. Salvo para registros fiscais e documentação

Transações

Visualize o histórico completo de transações, incluindo cobranças, créditos e pagamentos.

Exibição de Transações:

- **Data** - Data de criação da transação
- **Descrição** - Para o que a cobrança/crédito se destina
- **Valor** - Cobrança (positivo) ou crédito (negativo)
- **Fatura** - Qual fatura inclui esta transação
- **Status** - Faturado ou Não Faturado

Métodos de Pagamento

Gerencie cartões de crédito salvos para cobrança automática e pagamentos online.

Gerenciamento de Métodos de Pagamento:

- **Adicionar Cartão** - Adicionar novo cartão de crédito/débito com segurança via Stripe
- **Definir Padrão** - Escolher método de pagamento principal
- **Remover Cartão** - Excluir cartões expirados ou não utilizados
- **Detalhes do Cartão** - Visualizar os últimos 4 dígitos, expiração, marca do cartão

Veja `Métodos de Pagamento <payment_methods>` para documentação detalhada de gerenciamento de pagamentos.

Recarga / Recarregar

Compre dados adicionais, voz, SMS ou crédito monetário instantaneamente.

Processo de Recarga:

1. Navegue até **Serviços** → Selecione o serviço → **"Recarga"**
2. Escolha o produto de recarga no catálogo
3. Selecione o valor (opções predefinidas ou personalizadas)
4. Revise o custo e as informações de expiração
5. Selecione o método de pagamento
6. Confirme a compra
7. Saldo atualizado imediatamente

Recargas Disponíveis:

- **Recargas de Dados** - opções de 1GB, 5GB, 10GB, 20GB, 50GB
- **Recargas de Voz** - Minutos de chamadas adicionais
- **Pacotes de SMS** - Pacotes de mensagens
- **Crédito Monetário** - Saldo pré-pago (£5, £10, £20, £50, £100)

Veja `Recarga & Recarregar <features_topup_recharge>` para fluxos de trabalho detalhados de recarga.

Complementos de Serviço

Compre recursos e melhorias adicionais para serviços existentes.

Complementos Disponíveis:

- **Roaming Internacional** - Habilitar serviço no exterior
- **Endereço IP Estático** - IP fixo para internet residencial
- **Conteúdo Premium** - Canais IPTV, serviços de streaming
- **Atualizações de Hardware** - Aluguel de modem, set-top boxes
- **Aumentos de Velocidade** - Aumentos temporários de largura de banda

Comprando Complementos:

1. Navegue até **Serviços** → Selecione o serviço → **"Complementos"**
2. Navegue pelos complementos disponíveis para este tipo de serviço
3. Clique em **"Adicionar ao Serviço"** no complemento desejado
4. Revise o custo (uma vez + recorrente)
5. Confirme a compra
6. Complemento provisionado automaticamente

Veja `Modificando Serviços - Complementos <csa_modify>` para detalhes de gerenciamento de complementos.

Notificações & Alertas

Os clientes recebem notificações automatizadas para eventos importantes:

Notificações por E-mail:

- Fatura gerada e pronta para pagamento
- Confirmação de pagamento recebido
- Avisos de expiração de serviço (7 dias, 3 dias, 1 dia)
- Alertas de saldo baixo (dados, voz, monetário)
- Ativação/desativação de serviço
- Solicitações de redefinição de senha
- Alertas de segurança (login de novo dispositivo)

Alertas no Portal:

- Faturas não pagas
- Serviços expirando
- Avisos de dados baixos (10% restantes)
- Método de pagamento expirando
- Ações necessárias (verificar e-mail, atualizar método de pagamento)

Preferências de Notificação:

Os clientes podem configurar:

- Frequência de notificação por e-mail
- Alertas de SMS (se habilitados)
- Limiares de alerta (por exemplo, notificar quando <20% de dados restantes)
- Categorias de notificação (faturamento, uso, serviço)

Limitações de Autoatendimento

Algumas operações requerem assistência da equipe:

Requer Atendimento ao Cliente:

- Alterar tipo de cliente (Individual ↔ Empresa)
- Transferir serviços entre clientes
- Cancelar serviços (pode ser autoatendimento se habilitado)
- Disputar faturas
- Solicitar reembolsos
- Alterar contato principal
- Questões complexas de provisionamento

Restrições de Segurança:

- Não pode visualizar ou modificar contas de outros usuários
- Limitado a seus próprios dados de cliente (isolamento de inquilinos)
- Não pode acessar funções administrativas
- Não pode anular faturas ou transações
- Não pode modificar a configuração do serviço (apenas complementos/recargas)

Acesso de Impersonação de Funcionários

A equipe de suporte pode acessar o Portal de Autoatendimento como um cliente para solução de problemas.

Processo de Impersonação:

1. Navegue até **Clientes** → Selecione o cliente
2. Vá para a aba **Contatos**
3. Encontre o contato vinculado à conta do usuário
4. Clique no botão **"Login como Usuário"**
5. Uma nova aba se abre com a visão do Portal de Autoatendimento do cliente
6. Todas as ações são registradas e atribuídas ao usuário impersonado
7. A equipe vê exatamente o que o cliente vê

Casos de Uso:

- **Solução de Problemas** - Reproduzir problemas relatados pelo cliente
- **Verificação** - Confirmar que as configurações de serviço aparecem corretamente
- **Treinamento** - Demonstrar recursos do portal

- **Suporte** - Guiar o cliente pelo portal enquanto visualiza sua tela

Segurança & Auditoria:

- Requer permissão `can_impersonate` ou `admin`
- Todas as ações registradas na trilha de auditoria
- O cliente vê a impersonação no log de atividades
- Tempo limite de sessão após inatividade
- Não pode alterar a senha do cliente enquanto impersona

Veja `Atendimento ao Cliente - Impersonação de Usuário <customer_care>` para documentação completa de impersonação.

Responsividade Móvel

O Portal de Autoatendimento é totalmente responsivo e otimizado para dispositivos móveis.

Recursos Móveis:

- Navegação otimizada para toque
- Layouts simplificados para telas pequenas
- Gestos de deslizar para navegação em carrossel
- Formulários e entradas amigáveis para dispositivos móveis
- Escaneamento de código QR para provisionamento de eSIM
- Chamadas telefônicas com um toque
- Integração de GPS para autocompletar endereços

Aplicativo Web Progressivo (PWA):

- Instalar como aplicativo na tela inicial
- Visualização offline de dados recentes
- Notificações push (se habilitadas)
- Carregamento rápido com trabalhadores de serviço

Redefinição de Senha & Recuperação de Conta

Os clientes podem redefinir senhas esquecidas sem ligar para o suporte.

Redefinição de Senha de Autoatendimento:

1. Clique em **"Esqueceu a Senha"** na página de login
2. Insira o endereço de e-mail
3. Receba um e-mail de redefinição de senha (válido por 1 hora)
4. Clique no link no e-mail
5. Insira uma nova senha (deve atender aos requisitos de complexidade)
6. Envie e faça login com a nova senha

Requisitos de Senha:

- Mínimo de 8 caracteres
- Pelo menos uma letra maiúscula
- Pelo menos uma letra minúscula
- Pelo menos um número
- Pelo menos um caractere especial (!@#$%^&*)

Bloqueio de Conta:

Após 5 tentativas de login falhadas:

- Conta bloqueada por 30 minutos
- E-mail de redefinição de senha enviado automaticamente
- Notificação de segurança enviada para o e-mail registrado

Melhores Práticas para Clientes

Recomendações de Segurança:

1. Habilitar 2FA para segurança aprimorada
2. Usar senha única e forte

3. Manter o endereço de e-mail atualizado para notificações
4. Configurar método de pagamento padrão para renovações automáticas
5. Monitorar uso regularmente para evitar cobranças excessivas
6. Salvar códigos de backup em local seguro (se 2FA habilitado)
7. Sair após usar computadores compartilhados/públicos

Gerenciamento de Uso:

1. Habilitar alertas de saldo baixo (10-20% restantes)
2. Recargar antes que os saldos expirem para evitar interrupção do serviço
3. Revisar faturas mensais para cobranças inesperadas
4. Atualizar métodos de pagamento antes que os cartões expirem
5. Monitorar uso de dados ao longo do mês para evitar estrangulamento

Escalonamento de Suporte:

Se o autoatendimento não resolver o problema:

1. Verifique a base de conhecimento / artigos de ajuda (se disponíveis)
2. Revise o log de atividades para alterações recentes
3. Entre em contato com o atendimento ao cliente por telefone, e-mail ou chat
4. Forneça ID do cliente e detalhes do serviço para resolução mais rápida

Acesso à API

Clientes com requisitos técnicos podem usar a API diretamente.

Geração de Chave da API:

Disponível para clientes empresariais ou mediante solicitação:

1. Navegue até **Conta** → **Acesso à API**
2. Clique em **"Gerar Chave da API"**
3. Defina permissões (somente leitura ou leitura e gravação)
4. Defina a data de expiração

5. Salve a chave da API com segurança (mostrada apenas uma vez)

Casos de Uso da API:

- Monitoramento automatizado de uso
- Integração com sistemas de faturamento internos
- Recargas programáticas
- Provisionamento de serviços via scripts
- Exportações de dados para análises

Veja `Documentação da API <concepts_api>` para detalhes de endpoints e exemplos.

Perguntas Frequentes

Q: Por que não consigo ver todos os meus serviços?

R: Certifique-se de que você está logado com a conta correta. Se você tiver várias contas de cliente, cada uma tem serviços separados. Entre em contato com o suporte para mesclar contas, se necessário.

Q: Meu pagamento falhou, mas fui cobrado. O que devo fazer?

R: Verifique seu extrato bancário para cobranças pendentes. Se a cobrança aparecer, mas a fatura ainda mostrar não paga, entre em contato com o suporte com o número de referência da transação.

Q: Como faço para cancelar um serviço?

R: Navegue até os detalhes do serviço e clique em "Cancelar Serviço" (se o autoatendimento estiver habilitado). Caso contrário, entre em contato com o atendimento ao cliente para processar o cancelamento.

Q: Posso transferir um serviço para outra pessoa?

R: Não, transferências de serviço requerem assistência do atendimento ao cliente por razões de segurança e conformidade.

Q: Por que meu saldo de dados está diminuindo mais rápido do que o esperado?

R: Verifique atualizações de aplicativos em segundo plano, qualidade de streaming de vídeo e backups automáticos na nuvem. Revise a quebra de uso no portal para consumo detalhado.

Q: Perdi meu dispositivo 2FA. Como posso recuperar o acesso?

R: Use códigos de backup para fazer login, depois desative e reative o 2FA. Se não tiver códigos de backup, entre em contato com o suporte para redefinição do 2FA (requer verificação de identidade).

Q: Posso pagar uma fatura sem fazer login?

R: O pagamento de fatura como convidado pode estar disponível via link direto da fatura. Caso contrário, o login é necessário por motivos de segurança.

Q: Como faço para baixar todas as minhas faturas de uma vez?

R: Atualmente, é necessário baixar cada fatura individualmente. Para downloads em massa, entre em contato com o suporte ou use a API se disponível.

Documentação Relacionada

- `Atendimento ao Cliente - Impersonação de Usuário <customer_care>` - Acesso de solução de problemas da equipe
- `Métodos de Pagamento <payment_methods>` - Gerenciando cartões e pagamentos
- `Recarga & Recarregar <features_topup_recharge>` - Comprando crédito adicional
- `Uso de Serviço <csa_service_usage>` - Compreendendo o rastreamento de saldo
- `Visão Geral de Faturamento <billing_overview>` - Conceitos de faturamento e faturas
- `Fluxos de Autenticação <authentication_flows>` - Login e segurança

- `2FA <2fa>` - Configuração de autenticação de dois fatores
- `Modificações de Serviço <csa_modify>` - Adicionando recursos e complementos

Guia de Operações do OmniCRM

OmniCRM é a solução abrangente BSS/OSS da Omnitouch para provedores de serviços móveis e de linha fixa. Uma plataforma completa que gerencia tudo, desde a integração de clientes até faturamento, provisionamento e suporte - tudo em um sistema integrado.

Começando

Para a Equipe de Atendimento ao Cliente

Seus primeiros passos:

1. **Aprenda a interface** - Familiarize-se com a navegação e pesquisa
2. **Crie um cliente** - Criação de cliente passo a passo
3. **Adicione um serviço** - Provisione seu primeiro serviço
4. **Processar um pagamento** - Gerencie pagamentos de clientes
5. **Recarregar serviços** - Adicione crédito às contas dos clientes

Tarefas diárias:

- **Gerenciamento de serviços** - Gerencie os serviços dos clientes
- **Ver uso** - Verifique saldos e uso
- **Modificar serviços** - Altere configurações de serviços
- **Gerar faturas** - Crie e envie faturas

Para Administradores de Sistema

Configuração e configuração:

1. **Entenda a arquitetura** - Visão geral do sistema

2. **Configure o sistema** - Configurações do sistema
3. **Crie produtos** - Construa seu catálogo
4. **Escreva playbooks** - Automatize o provisionamento
5. **Configure usuários** - Crie contas e atribua funções

Tópicos avançados:

- **Gerenciamento de inventário** - Gerencie ativos
- **Personalização** - Adapte às suas necessidades
- **Integração de API** - Conecte sistemas externos
- **Configuração de segurança** - Configure 2FA e permissões

Para Clientes

Usando o Portal de Autoatendimento:

- **Acesse sua conta** - Faça login e navegue
- **Ver serviços** - Veja seus serviços ativos
- **Verificar uso** - Monitore dados e saldos
- **Pagar faturas** - Faça pagamentos online
- **Recarregar serviços** - Adicione crédito instantaneamente

---

Referência Rápida de Tarefas

Eu quero...	Documentação
Criar um novo cliente	Criar Cliente
Adicionar um serviço a um cliente	Adicionar Serviço
Ver uso de serviços e saldos	Uso de Serviço
Processar um pagamento	Processar Pagamento
Gerar uma fatura	Gerenciamento de Faturas
Recarregar um serviço	Recarregar & Recarregar
Gerenciar inventário	Sistema de Inventário
Criar um produto	Ciclo de Vida do Produto
Escrever um playbook de provisionamento	Playbooks Ansible
Configurar contas de usuário	RBAC
Habilitar 2FA	Autenticação de Dois Fatores
Pesquisar qualquer coisa	Pesquisa Global
Entender o sistema	Arquitetura do Sistema
Usar a API	Documentação da API

Biblioteca de Documentação

Leitura Essencial

- **Arquitetura do Sistema** - Visão técnica completa com diagramas
- **Guia do Ciclo de Vida do Produto** - Gerenciamento de produtos de ponta a ponta
- **Guia de Playbooks Ansible** - Domine a automação de provisionamento

Gerenciamento de Clientes

- Clientes, Contatos & Locais - Modelo de dados e relacionamentos
- Criando Clientes - Guia passo a passo
- Tags de Clientes - Organize com tags
- Atributos de Clientes - Metadados personalizados
- Registro de Atividades - Rastreie todas as mudanças

Operações de Serviço

- Gerenciamento de Serviços - Visão geral
- Adicionando Serviços - Fluxo de trabalho de provisionamento
- Atribuindo Planos - Atribuição de planos
- Modificando Serviços - Fazendo alterações
- Uso de Serviço & Saldos - Monitore o uso
- Recarregar & Recarregar - Adicione créditos

Faturamento & Finanças

- Visão Geral de Faturamento - Guia completo de faturamento
- Métodos de Pagamento - Gerenciando métodos de pagamento
- Transações - Cobranças e créditos
- Processamento de Pagamentos - Fluxos de trabalho de pagamento
- Gerenciamento de Faturas - Gere e gerencie faturas

Administração

- Configuração do Sistema - Configure o OmniCRM
- Personalização - Adapte ao seu negócio
- Gerenciamento de Inventário - Rastreamento de ativos
- Chaves de API - Gerenciamento de acesso à API

Segurança & Acesso

- Fluxos de Autenticação - Como a autenticação funciona
- Autenticação de Dois Fatores - Configure 2FA
- Controle de Acesso Baseado em Funções - Usuários, funções e permissões

Integrações

- Integração com Stripe - Processamento de pagamentos
- Integração com Mailjet - Serviço de email
- Documentação da API - Referência da API REST

Voltado para o Cliente

- Portal de Autoatendimento - Guia do portal do cliente
- Atendimento ao Cliente - Recursos de autoatendimento
- Sistema de Transmissão de Emergência - Alertas de emergência

Referência

- Glossário - Termos e definições
- Registro de Alterações - Histórico de versões

O que torna o OmniCRM especial?

O OmniCRM reúne todas as ferramentas necessárias para administrar um negócio moderno de telecomunicações:

Provisionamento de Serviços Automatizado

Esqueça a configuração manual - o OmniCRM usa **automação Ansible** para provisionar serviços em segundos. Se você está ativando um cartão SIM, configurando equipamentos de cliente ou montando um pacote complexo, o sistema cuida disso automaticamente.

O que ele faz:

- Provisione serviços com um único clique
- Configura equipamentos de rede automaticamente
- Cria contas de faturamento em tempo real
- Envia e-mails de boas-vindas e notificações por SMS
- Reverte automaticamente se algo falhar

Saiba mais sobre Provisionamento | **Veja o Guia de Playbooks Ansible**

Faturamento e Pagamentos Inteligentes

A integração embutida com **CGRateS** fornece classificação e cobrança em tempo real, enquanto a **integração com Stripe** lida com o processamento de pagamentos de forma tranquila.

O que ele faz:

- Rastreamento e classificação de uso em tempo real
- Geração automática de faturas
- Processamento de cartões de crédito via Stripe
- Modelos de faturamento pré-pago e pós-pago
- Preços flexíveis e códigos promocionais
- Cálculo e relatórios de impostos

Explore Recursos de Faturamento | **Processamento de Pagamentos** | **Gerenciamento de Faturas**

Gerenciamento Completo de Clientes

Gerencie clientes, contatos, múltiplos locais e serviços com um modelo de relacionamento abrangente.

O que ele faz:

- Rastreie clientes, contatos e locais de serviço
- Gerencie múltiplos serviços por cliente
- Armazene atributos e metadados personalizados
- Tagueie clientes para organização
- Histórico completo de atividades e logs de auditoria
- Integração com Google Maps para geocodificação de locais

Guia de Gerenciamento de Clientes | **Crie um Cliente** | **Tags de Clientes**

Gerenciamento de Inventário

Rastreie e gerencie todos os seus ativos físicos e virtuais - de cartões SIM a roteadores e blocos de endereços IP.

O que ele faz:

- Rastreie cartões SIM, equipamentos e acessórios
- Gerencie números de telefone e blocos de endereços IP
- Atribuição automatizada durante o provisionamento
- Capacidade de importação e exportação em lote
- Armazenamento de configuração de equipamentos
- Modelos de inventário personalizados

Documentação do Sistema de Inventário

Portal de Autoatendimento do Cliente

Empodere seus clientes com um **portal de autoatendimento** onde eles podem gerenciar seus próprios serviços.

O que os clientes podem fazer:

- Ver serviços e uso
- Verificar saldos e limites de dados
- Pagar faturas online
- Baixar faturas como PDFs
- Atualizar informações de contato
- **Recarregar seus serviços** instantaneamente

Guia do Portal de Autoatendimento | **Recursos de Atendimento ao Cliente**

Segurança Empresarial

Construído com segurança em seu núcleo, apresentando autenticação e autorização abrangentes.

Recursos de segurança:

- **Autenticação baseada em JWT**
- **Autenticação de dois fatores (2FA)** com TOTP e códigos de backup
- **Controle de acesso baseado em funções (RBAC)** com permissões granulares
- Verificação de e-mail para alterações de conta
- Registro completo de auditoria via log de atividades
- Gerenciamento de sessões e controles de tempo limite

Guia de Autenticação | **Documentação RBAC** | **Configuração de 2FA**

Integrações Poderosas

O OmniCRM integra-se perfeitamente com serviços líderes do setor:

- **CGRateS** - Motor de faturamento e classificação em tempo real para cobrança de grau telecom
- **Stripe** - Processamento de pagamentos seguro e armazenamento de cartões

- **Mailjet** - Entrega de e-mail profissional com modelos
- **Google Maps** - Validação de endereços e geocodificação para locais de site precisos
- **API RESTful** - Integre com seus sistemas existentes

Documentação da API | **Integração com Stripe** | **Integração com Mailjet**

Catálogo de Produtos Flexível

Crie qualquer tipo de oferta de produto - de serviços simples e independentes a pacotes complexos com múltiplos componentes.

Tipos de produtos suportados:

- **Independentes** - Serviços únicos (planos móveis, pacotes de internet)
- **Pacotes** - Ofertas combinadas (internet + TV + telefone)
- **Complementos** - Serviços suplementares (recargas de dados, chamadas internacionais)
- **Promoções** - Ofertas especiais e descontos

Cada produto pode ter sua própria automação de provisionamento, regras de preços e lógica de negócios.

Guia do Ciclo de Vida do Produto | **Conceitos de Produtos & Serviços**

Sistema de Transmissão de Emergência

Para operadores móveis, o OmniCRM inclui um **Sistema de Transmissão de Células** para alertas de segurança pública.

O que ele faz:

- Enviar alertas de emergência para áreas geográficas
- Suporte a mensagens multilíngues
- Segmentação por operador de rede móvel
- Conformidade com padrões de alerta do governo

Documentação de Transmissão de Células

Pesquisa e Navegação Poderosas

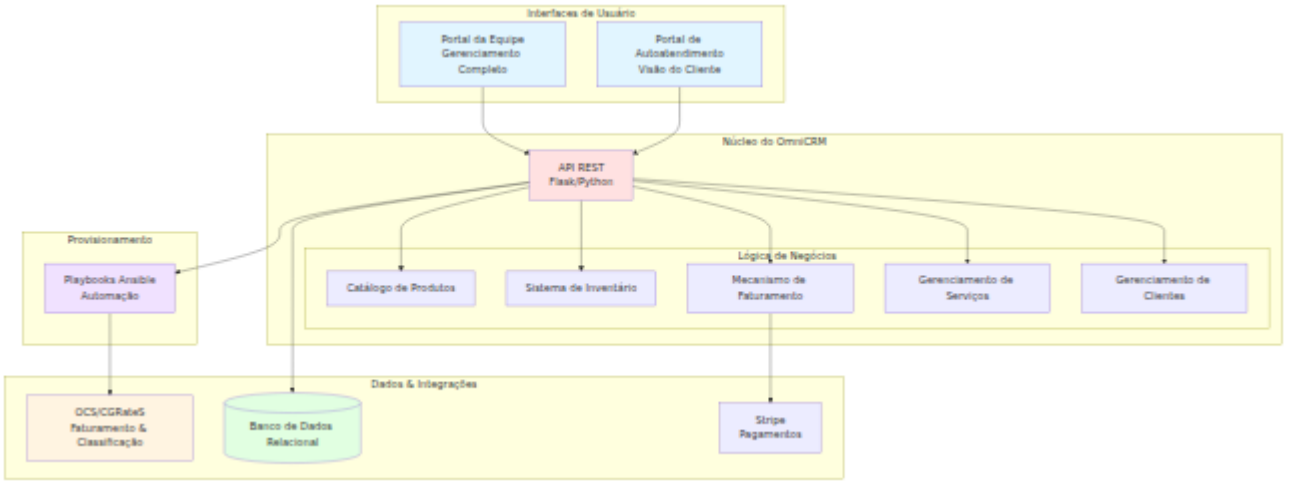
Encontre qualquer coisa instantaneamente com **pesquisa global** em clientes, serviços, faturas e mais.

Recursos de pesquisa:

- Pesquisar em todas as entidades
- Filtrar e classificar resultados
- Atalhos de navegação rápida
- Sugestões inteligentes

Guia de Navegação | **Pesquisa Global**

Arquitetura do Sistema



Veja a Documentação Completa da Arquitetura

Conceitos Chave

Entendendo Produtos vs Serviços

Este é um dos conceitos mais importantes no OmniCRM:

- **Produto** = Um modelo ou oferta em seu catálogo (por exemplo, "Plano Móvel Ilimitado")
- **Serviço** = Uma instância ativa de um produto para um cliente específico (por exemplo, "Plano Móvel Ilimitado do John Smith")

Quando você provisiona um produto, o sistema usa **automação Ansible** para criar o(s) serviço(s) real(is). Um produto pode criar múltiplos serviços, nenhum serviço (apenas configuração) ou modificar serviços existentes.

Saiba mais sobre Produtos & Serviços

A Magia do Provisionamento

O segredo do OmniCRM é seu **sistema de provisionamento automatizado**:

O playbook cuida de tudo - desde a criação de contas no CGRateS até a configuração de roteadores e o envio de e-mails de boas-vindas. E se algo falhar, ele reverte automaticamente.

Aprofunde-se: Playbooks Ansible | **Sistema de Provisionamento**

Faturamento Simplificado

Cada serviço gera **transações** (cobranças ou créditos). Estas são agrupadas em **faturas** e enviadas aos clientes. Com a integração do Stripe, os pagamentos podem ser processados automaticamente.

Visão Geral de Faturamento | **Processamento de Pagamentos**

Hierarquia de Clientes

```
Cliente (ABC Company)
├── Contatos (John Smith - Faturamento, Jane Doe - Técnico)
├── Locais (Sede, Filial)
└── Serviços
    ├── Internet - Sede
    ├── Plano Móvel - John Smith
    └── VoIP - Filial
```

Tudo é organizado de forma hierárquica, facilitando o gerenciamento de clientes com múltiplas localizações e serviços.

Clientes, Contatos & Locais

Por que escolher o OmniCRM?

Plataforma Tudo-em-Um

Tudo o que você precisa em um único sistema integrado - sem necessidade de equilibrar várias ferramentas ou fornecedores.

Automação em Primeiro Lugar

Provisionamento com Ansible significa que os serviços são implantados em segundos, não em horas. Reduza erros, economize tempo, aumente a satisfação do cliente.

Flexível e Personalizável

Desde definições de produtos até playbooks de provisionamento e atributos personalizados - adapte o OmniCRM para corresponder exatamente aos seus processos de negócios.

Construído para Telecom

Projetado especificamente para provedores de serviços com recursos como integração com CGRateS, gerenciamento de inventário e classificação em tempo real.

Segurança Empresarial

Autenticação JWT, 2FA, RBAC e registro completo de auditoria garantem que seus dados estejam seguros e em conformidade.

Orientado por API

Uma API REST abrangente significa que você pode integrar o OmniCRM com quaisquer sistemas existentes ou construir ferramentas personalizadas.

Obtenha Suporte

Recursos de Documentação

- Comece com **Arquitetura do Sistema** para uma visão técnica
- Veja **guias de introdução** para integração específica de funções
- Consulte a **Referência Rápida de Tarefas** para operações comuns
- Consulte o **Glossário** para terminologia

Precisa de Mais Ajuda?

Navegue pela biblioteca de documentação completa acima ou use a **pesquisa global** para encontrar tópicos específicos.

OmniCRM - BSS/OSS Completo para Provedores de Serviços Modernos

Última Atualização: 2025-12-23