

Documentação - Módulo Identidade

Regras de Negócio - Módulo **pc-identidade**

Objetivo

O projeto **pc-identidade** é o pilar central para a gestão do ciclo de vida de **Varejistas (Sellers)** e de seus respectivos **Usuários** no ecossistema do marketplace. Ele garante a integridade dos dados cadastrais, mas também provê uma camada de segurança robusta para controle de acesso, atuando como a fonte da verdade sobre quem são os participantes da plataforma e o que eles podem fazer.

As responsabilidades do serviço foram expandidas para incluir:

- **API Completa para Sellers:** Oferece endpoints RESTful para o ciclo de vida completo de um seller (CRUD), incluindo validações de dados, regras de negócio e um sistema de exclusão lógica ("soft delete") através de um campo de status.
 - **Integração com Keycloak:** Automatiza a criação e gerenciamento de usuários no Keycloak. Cada novo seller ou usuário criado na plataforma tem sua identidade correspondente gerenciada pelo serviço.
 - **Autenticação e Autorização:** Protege os endpoints da API utilizando tokens JWT. A validação dos tokens é otimizada com um sistema de cache em Redis para garantir alta performance.
 - **Controle de Acesso Granular:** Implementa permissões detalhadas. Um usuário comum só pode visualizar e modificar os dados dos sellers aos quais está explicitamente associado (através do atributo sellers no Keycloak).
 - **Fluxos de Segurança:** Notifica outros sistemas sobre eventos importantes (como a criação de um seller) de forma assíncrona via RabbitMQ.
 - **Gerenciamento de Dados:** Inclui uma estratégia de "Cold Storage", com um banco de dados secundário para arquivar sellers inativos, mantendo a base de dados principal enxuta e performática.
 - **Observabilidade:** Utiliza um sistema de logging estruturado para registrar o fluxo de requisições, operações de negócio e erros, facilitando a depuração e o monitoramento.
-

Regras de Negócio Específicas

Esta seção detalha as regras de validação, segurança e processos que governam as entidades **Seller** e **User** no sistema.

Regras de Cadastro e Validação (Entidade Seller)

Estas regras garantem a integridade e a consistência dos dados dos sellers. São validadas principalmente na criação (POST) e atualização (PUT/PATCH).

REGRA-SELLER-001: Validação de CNPJ

- O **cnpj** fornecido deve conter exatamente 14 dígitos e ser composto apenas por números.
- Mensagem de Erro Exemplo: "O CNPJ é inválido. Deve conter exatamente 14 dígitos numéricos."

REGRA-SELLER-002: Unicidade do Nome Fantasia (trade_name)

- O **trade_name** (nome fantasia) deve ser exclusivo em todo o sistema para evitar duplicidade e confusão.
- Tentativas de cadastro ou atualização com um nome fantasia já existente (pertencente a outro seller) serão rejeitadas.
- Mensagem de Erro Exemplo: "O nome_fantasia informado já está cadastrado. Escolha outro."

REGRA-SELLER-003: Validação do seller_id

- O **seller_id** é o identificador único e imutável de um seller.
- Deve ser exclusivo em todo o sistema.
- Deve conter apenas caracteres alfanuméricos minúsculos (a-z, 0-9), sem espaços ou símbolos.
- Mensagem de Erro Exemplo: "O seller_id informado já está cadastrado."

REGRA-SELLER-004: Validação de CPF do Representante Legal

- O **legal_rep_cpf** deve conter exatamente 11 dígitos e ser composto apenas por números.
- Mensagem de Erro Exemplo: "CPF deve conter exatamente 11 dígitos numéricos"

REGRA-SELLER-005: Data de Nascimento Válida

- A data de nascimento do representante legal (legal_rep_birth_date) não pode ser uma data futura.

Regras de Segurança e Controle de Acesso

Estas regras definem quem pode ver e modificar os recursos da API.

REGRA-ACESSO-001: Autorização Baseada em Atributos (ABAC)

- Um usuário autenticado só pode visualizar, alterar ou inativar um seller se o **seller_id** correspondente estiver presente na lista de **sellers** dentro do seu token JWT.
- **Comportamento:** Se um usuário tentar acessar um **seller_id** que não está em seu token, a API retornará um erro **403 Forbidden** ou **404 Not Found** para não vaziar a existência do recurso.

REGRA-ACESSO-002: Autorização Baseada em Role (RBAC) para Admin

- Endpoints de gerenciamento de usuários (ex: **GET /seller/v1/users**) são protegidos e requerem que o usuário autenticado possua a role **realm-admin** em seu token.
- Mensagem de Erro Exemplo: "Esta ação requer privilégios de administrador."

REGRA-ACESSO-003: Autoatendimento de Usuário

- Um usuário comum (não-admin) só pode modificar (**PATCH**) ou deletar (**DELETE**) seus próprios dados de usuário. A tentativa de alterar dados de outro usuário resultará em um erro **403 Forbidden**.
- Mensagem de Erro Exemplo: "Você só pode alterar seus próprios dados."

Regras de Ciclo de Vida e Processos

Estas regras descrevem os processos automáticos que ocorrem durante as operações da API.

REGRA-PROCESSO-001: Associação de Seller no Cadastro (POST /sellers)

- Quando um usuário autenticado cria um novo **seller**, o **seller_id** deste novo seller é automaticamente adicionado à lista de **sellers** nos atributos do usuário no **Keycloak**, concedendo-lhe permissão imediata sobre o seller recém-criado.

REGRA-PROCESSO-002: Exclusão Lógica e Revogação de Acesso (DELETE /sellers/{id})

- A operação **DELETE** em um seller não apaga o registro do banco de dados. Em vez disso, ela realiza um "*soft delete*":
 - O campo **status** do **seller** é alterado para "**Inativo**".
 - O **seller_id** correspondente é removido da lista de **sellers** nos atributos do usuário no **Keycloak**, revogando o acesso.
- **Sellers** inativos não são retornados em buscas padrão.

Modelagem da Coleção: **sellers**

Informações da Empresa

Tt Campo	Tipo	Tt Obrigatório	Tt Descrição	Tt Restrições
seller_id	String ▾	Sim	ID único do seller no sistema.	Alfanumérico, lowercase, sem espaços ou símbolos.
company_name	String ▾	Sim	Razão Social da empresa.	—
cnpj	String ▾	Sim	CNPJ do seller	Apenas dígitos, exatamente 14 caracteres.
trade_name	String ▾	Sim	Nome fantasia do seller. Deve ser único.	Mínimo 3 caracteres, não vazio.
commercial_address	String ▾	Sim	Endereço comercial da empresa.	—
state_municipal_registration	String ▾	Sim	Inscrição Estadual/Municipal.	Apenas números.
contact_phone	String ▾	Sim	Telefone de contato principal.	Caracteres não numéricos serão removidos.
contact_email	Email ▾	Sim	E-mail de contato principal.	Deve ser um formato de e-mail válido.

Informações do Representante Legal

Tt Campo	🔍 Tipo	Tt Obrigatório	Tt Descrição	Tt Restrições
legal_rep_full_name	String ▾	Sim	Nome completo do representante legal.	—
legal_rep_cpf	String ▾	Sim	CPF do representante legal.	Apenas dígitos, exatamente 11 caracteres.
legal_rep_rg_number	String ▾	Sim	Número do RG do representante legal.	Apenas números.
legal_rep_rg_state	String ▾	Sim	Estado emissor do RG (ex: "SP", "RJ").	Deve ser uma sigla válida de um Estado Brasileiro.
legal_rep_birth_date	Dat... ▾	Sim	Data de nascimento do representante.	Não pode ser uma data futura.
legal_rep_phone	String ▾	Sim	Telefone do representante legal.	Caracteres não numéricos serão removidos.
legal_rep_email	Email ▾	Sim	E-mail do representante legal.	Deve ser um formato de e-mail válido.

Informações Bancárias

Tt Campo	Tipo	Tt Obrigatório	Tt Descrição	Tt Restrições
bank_name	String ▾	Sim	Nome do banco.	Será convertido para minúsculas.
agency_account	String ▾	Sim	Agência e conta bancária.	—
account_type	String ▾	Sim	Tipo de conta (ex: "Corrente", "Poupança").	Deve ser um dos valores do Enum <code>AccountType</code> .
account_holder_name	String ▾	Sim	Nome do titular da conta.	—

Dados Operacionais

Tt Campo	Tipo	Tt Obrigatório	Tt Descrição	Tt Restrições
product_categories	List[Enum] ▾	Sim	Lista com as categorias de produtos.	Não pode ser uma lista vazia.
business_description	String ▾	Sim	Breve descrição do negócio do seller.	—

Campos de Sistema (Auditoria e Status)

Tipo Campo	Tipo	Obrigatório	Descrição
_id	ObjectId	Sim	Identificador único gerado pelo MongoDB.
status	String	Sim	Status do seller ("Ativo" ou "Inativo"). Padrão: "Ativo".
created_at	DateTime	Sim	Data e hora da criação do documento.
updated_at	DateTime	Sim	Data e hora da última atualização do documento.
created_by	String	Sim	Identificador do usuário que criou o registro. Salvo no modelo: "iss + sub" do Keycloak.
updated_by	String	Sim	Identificador do usuário que realizou a última atualização. Salvo no modelo: "iss + sub" do Keycloak.

Índices e Restrições

Campo(s) no Índice

Tt Campo	🔍 Tipo de Índice	Tt Justificativa e Comportamento
seller_id	Único (Unique) ▾	Crítico. Garante que cada seller_id seja exclusivo no sistema, prevenindo duplicidade de registros. É a chave primária de negócio e usada em buscas diretas.
trade_name	Único (Unique) ▾	Recomendado. Garante que o trade_name (nome fantasia) seja exclusivo, conforme a regra de negócio (REGRA-SELLER-002). Acelera as buscas por nome, que são comuns.
cnpj	Simples (Single Field) ▾	Recomendado. Embora a regra de negócio não exija unicidade, ter um índice no cnpj acelera significativamente as buscas por este campo, que podem ser frequentes em rotinas administrativas.
status, created_at	Composto (Compound) ▾	Otimização. Este é um índice composto, ideal para a lógica de " soft delete " e para buscas paginadas. Ele permite que o banco encontre e ordene rapidamente os sellers ativos (status : "Ativo") por data de criação, que é a consulta mais comum do sistema.



Exemplo de Documento

```
_id: ObjectId('686c7d9924b21a636aa1378e')
created_at: 2025-07-08T02:08:25.458+00:00
updated_at: 2025-07-08T02:08:25.458+00:00
created_by: "http://localhost:8080/realms/marketplace:fd43e5ca-9c5d-4979-8615-efe75..."
updated_by: "http://localhost:8080/realms/marketplace:fd43e5ca-9c5d-4979-8615-efe75..."
audit_created_at: 2025-07-08T02:08:25.458+00:00
audit_updated_at: 2025-07-08T02:08:25.458+00:00
seller_id: "loja5"
status: "Ativo"
company_name: "Empresa Exemplo LTDA"
trade_name: "Loja Exemplo 5"
cnpj: "12345678000199"
state_municipal_registration: "123456789"
commercial_address: "Rua Exemplo, 123, Bairro, Cidade - UF, 12345-678"
contact_phone: "5511999998888"
contact_email: "victorhugobuiatti@gmail.com"
legal_rep_full_name: "João da Silva"
legal_rep_cpf: "12345678900"
legal_rep_rg_number: "123456789"
legal_rep_rg_state: "SP"
legal_rep_birth_date: 1980-01-15T00:00:00.000+00:00
legal_rep_phone: "5511988887777"
legal_rep_email: "joao.silva@email.com"
bank_name: "banco exemplo s.a."
agency_account: "1234 / 12345-6"
account_type: "Corrente"
account_holder_name: "Empresa Exemplo LTDA"
product_categories: Array (2)
  0: "celulares e smartphones"
  1: "informática"
business_description: "Loja especializada em eletrônicos e acessórios de última geração."
```

Endpoints Users





1. Criar Users

POST /seller/v1/users

Descrição: Criar um novo usuário no sistema (**Keycloak**).

Utilizar **METHOD** do tipo **POST**.

Parâmetros de Entrada

 Campo	 Tipo	 Obrigatório	 Descrição
username	String ▾	Sim	Login único no sistema.
email	Email ▾	Sim	Email do usuário no sistema.
password	String ▾	Sim	Senha do usuário no sistema, armazenada e gerenciada pelo Keycloak.
first_name	String ▾	Sim	Primeiro nome do usuário.
last_name	String ▾	Sim	Sobrenome do usuário.

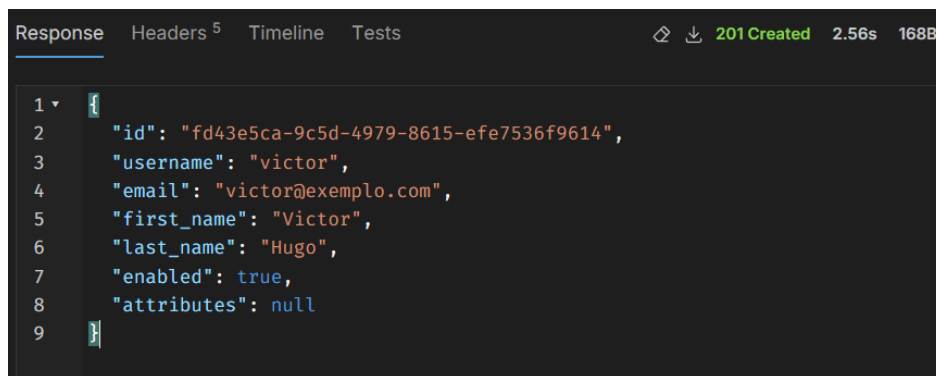
Abaixo exibiremos o método aberto através do **Bruno**. A chamada da API ocorre da seguinte forma:

```
POST http://127.0.0.1:8000/seller/v1/users

Params Body * Headers Auth * Vars Script Assert Tests Docs
JSON ▾ Prettify

1 {
2   "username": "victor",
3   "email": "victor@exemplo.com",
4   "password": "senha123",
5   "first_name": "Victor",
6   "last_name": "Buiatti"
7 }
8 |
```

Saída esperada (**Resposta** 201 Created):



```
Response  Headers 5  Timeline  Tests  201 Created  2.56s  168B

1  {
2    "id": "fd43e5ca-9c5d-4979-8615-efe7536f9614",
3    "username": "victor",
4    "email": "victor@exemplo.com",
5    "first_name": "Victor",
6    "last_name": "Hugo",
7    "enabled": true,
8    "attributes": null
9  }
```

O **id** retornado se refere ao identificador do usuário no Keycloak.

2. Listar Todos os Users

GET /seller/v1/users

Descrição: Retorna uma lista paginada de todos os usuários cadastrados no sistema de identidade (Keycloak).

Por ser uma operação administrativa que expõe dados de todos os usuários, este endpoint é protegido e requer privilégios de administrador.

Utilizar **METHOD** do tipo **GET**.

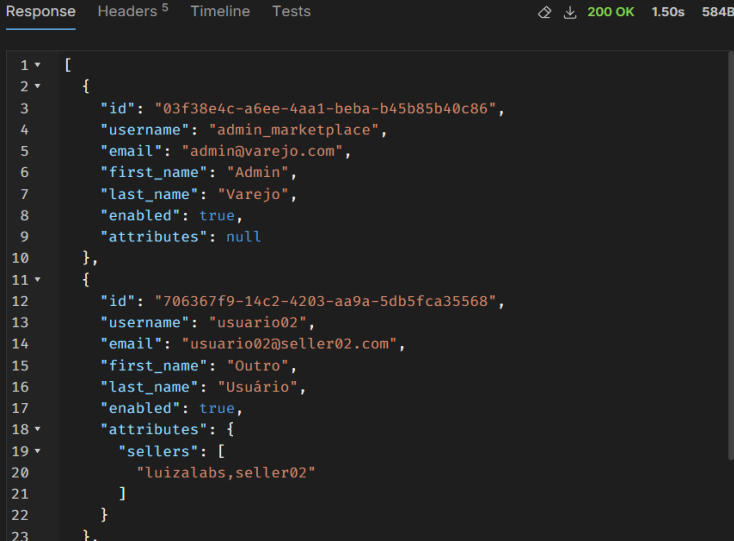
Autorização Obrigatória: A requisição deve incluir um **Authorization** header com um Bearer Token JWT válido. O usuário associado ao token deve possuir a role de **realm-admin**.

Headers:

- Content-Type: application/json
- Authorization: Bearer <seu_token_jwt_aqui>

Exemplos de Respostas:

200 OK - Sucesso



```
Response Headers Timeline Tests 200 OK 1.50s 584B

1 [
2   {
3     "id": "03f38e4c-a6ee-4aa1-beba-b45b85b40c86",
4     "username": "admin_marketplace",
5     "email": "admin@varejo.com",
6     "first_name": "Admin",
7     "last_name": "Varejo",
8     "enabled": true,
9     "attributes": null
10  },
11  {
12    "id": "706367f9-14c2-4203-aa9a-5db5fca35568",
13    "username": "usuario02",
14    "email": "usuario02@seller02.com",
15    "first_name": "Outro",
16    "last_name": "Usuário",
17    "enabled": true,
18    "attributes": {
19      "sellers": [
20        "luizalabs,seller02"
21      ]
22    }
23  },
24 ]
```

401 Unauthorized - Não Autenticado

Retornado se o token JWT não for fornecido, for inválido ou tiver expirado

403 Forbidden - Acesso Proibido

Retornado se o token for válido, mas o usuário não possuir a role **realm-admin** necessária para acessar este recurso.

500 Internal Server Error - Erro Interno

Retornado se ocorrer uma falha inesperada no servidor, como uma falha de comunicação com o Keycloak.

3. 🔍 Obter User por ID

GET `/seller/v1/users/{user_id}`

Descrição: Busca e retorna os detalhes de um usuário específico cadastrado no sistema de identidade (Keycloak), utilizando o seu **ID** único.

Utilizar **METHOD** do tipo **GET**.

Autorização Obrigatória: A requisição deve incluir um **Authorization** header com um Bearer Token JWT válido.

Regras de Permissão:

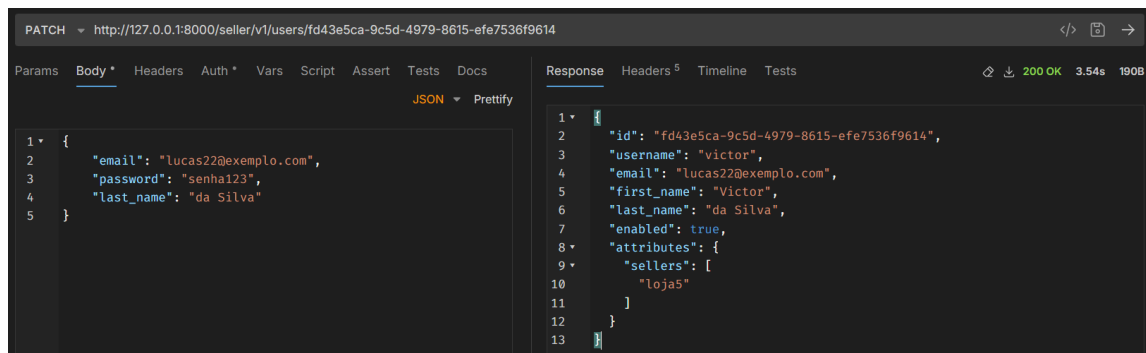
- Um usuário comum pode buscar **apenas os seus próprios dados**.
- Um usuário com a role de **realm-admin** pode buscar os dados de **qualquer** usuário.

Headers:

- Content-Type: application/json
- Authorization: Bearer <seu_token_jwt_aqui>

Exemplos de Respostas:

200 OK - Sucesso



401 Unauthorized - Não Autenticado

Retornado se o token JWT não for fornecido, for inválido ou tiver expirado.

403 Forbidden - Acesso Proibido

Retornado se o usuário autenticado tentar buscar os dados de **outro** usuário sem possuir a role de **realm-admin**.

404 Not Found - Não Encontrado

Retornado se nenhum usuário for encontrado com o **user_id** fornecido.

4. ✎ Atualizar User

PATCH `/seller/v1/users/{user_id}`

Descrição: Atualiza um ou mais campos de um usuário existente no sistema de identidade (Keycloak). Esta operação é uma atualização parcial, o que significa que você só precisa enviar os campos que deseja alterar.

Um usuário pode atualizar **apenas os seus próprios dados**. O ID do usuário no token (**sub**) deve ser o mesmo que o **user_id** passado na URL. Administradores não têm permissão para alterar dados de outros usuários através deste endpoint.

Utilizar **METHOD** do tipo **PATCH**.

Autorização: Obrigatória: A requisição deve incluir um **Authorization** header com um Bearer Token JWT válido.

Headers:

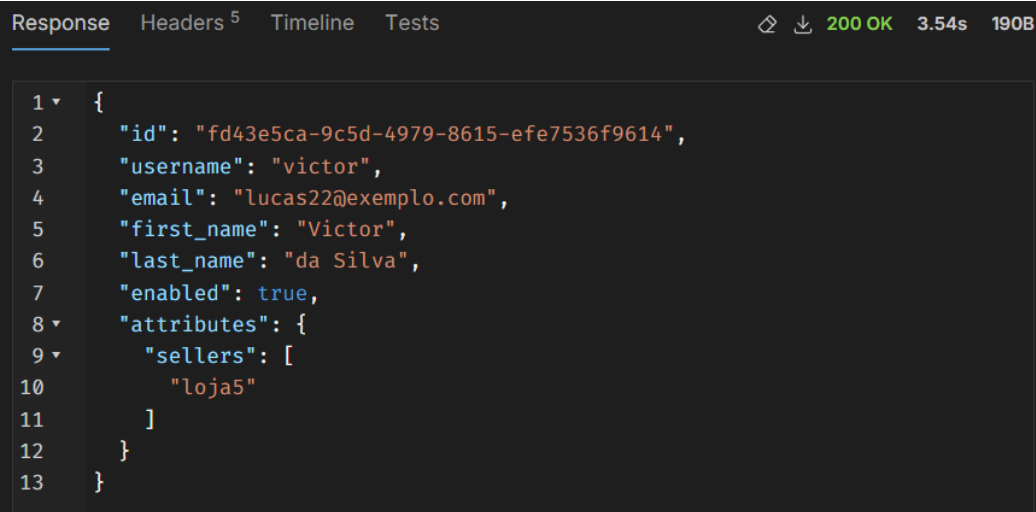
- Content-Type: application/json
- Authorization: Bearer <seu_token_jwt_aqui>

Parâmetros de Entrada Opcionais

Tipo Campo	Tipo	Obrigatório	Descrição
email	Email	Não	Email do usuário no sistema.
password	String	Não	Senha do usuário no sistema, armazenada e gerenciada pelo Keycloak.
first_name	String	Não	Primeiro nome do usuário.
last_name	String	Não	Sobrenome do usuário.

Exemplos de Respostas:

200 OK - Sucesso



The screenshot shows a REST client interface with tabs for Response, Headers, Timeline, and Tests. The Response tab is active, displaying a 200 OK status with a response time of 3.54s and a body size of 190B. The JSON body is as follows:

```
1 {
2   "id": "fd43e5ca-9c5d-4979-8615-efe7536f9614",
3   "username": "victor",
4   "email": "lucas22@exemplo.com",
5   "first_name": "Victor",
6   "last_name": "da Silva",
7   "enabled": true,
8   "attributes": {
9     "sellers": [
10      "loja5"
11    ]
12  }
13 }
```

401 Unauthorized - Não Autenticado

Retornado se o token JWT não for fornecido, for inválido ou tiver expirado.

403 Forbidden - Acesso Proibido

Retornado se o usuário autenticado tentar atualizar os dados de **outro** usuário.

422 Unprocessable Entity - Dados Inválidos

Retornado se os dados no corpo da requisição falharem na validação (ex: um e-mail em formato incorreto).

5. 🗑 Deletar User

DELETE /seller/v1/users/{user_id}

Descrição: Remove permanentemente um usuário do sistema de identidade (Keycloak). Esta é uma operação destrutiva e não pode ser desfeita.

Utilizar **METHOD** do tipo **DELETE**.

Autorização Obrigatória: A requisição deve incluir um **Authorization** header com um Bearer Token JWT válido.

Regras de Permissão

Este endpoint possui duas regras de acesso distintas:

- Um usuário comum, sem privilégios de administrador, pode deletar **apenas a sua própria conta**. Nesse caso, o **user_id** na URL deve ser idêntico ao ID do usuário (**sub**) presente no token JWT.
- Um usuário com a role **realm-admin** pode deletar a conta de **qualquer** usuário.

Headers:

- Content-Type: application/json
- Authorization: Bearer <seu_token_jwt_aqui>

Exemplos de Respostas:

204 No Content - Sucesso

Retornado quando o usuário é deletado com sucesso. Por padrão, esta resposta **não possui corpo (no body)**, indicando que a operação foi concluída.

401 Unauthorized - Não Autenticado

Retornado se o token JWT não for fornecido, for inválido ou tiver expirado.

403 Forbidden - Acesso Proibido

Retornado se um usuário autenticado tentar deletar a conta de **outro** usuário sem possuir a role de **realm-admin**.

404 Not Found - Não Encontrado

Retornado se nenhum usuário for encontrado com o **user_id** fornecido na URL.

Endpoints Sellers

1. Criar Seller

POST /seller/v1/sellers

Descrição: Cria um novo seller no sistema.

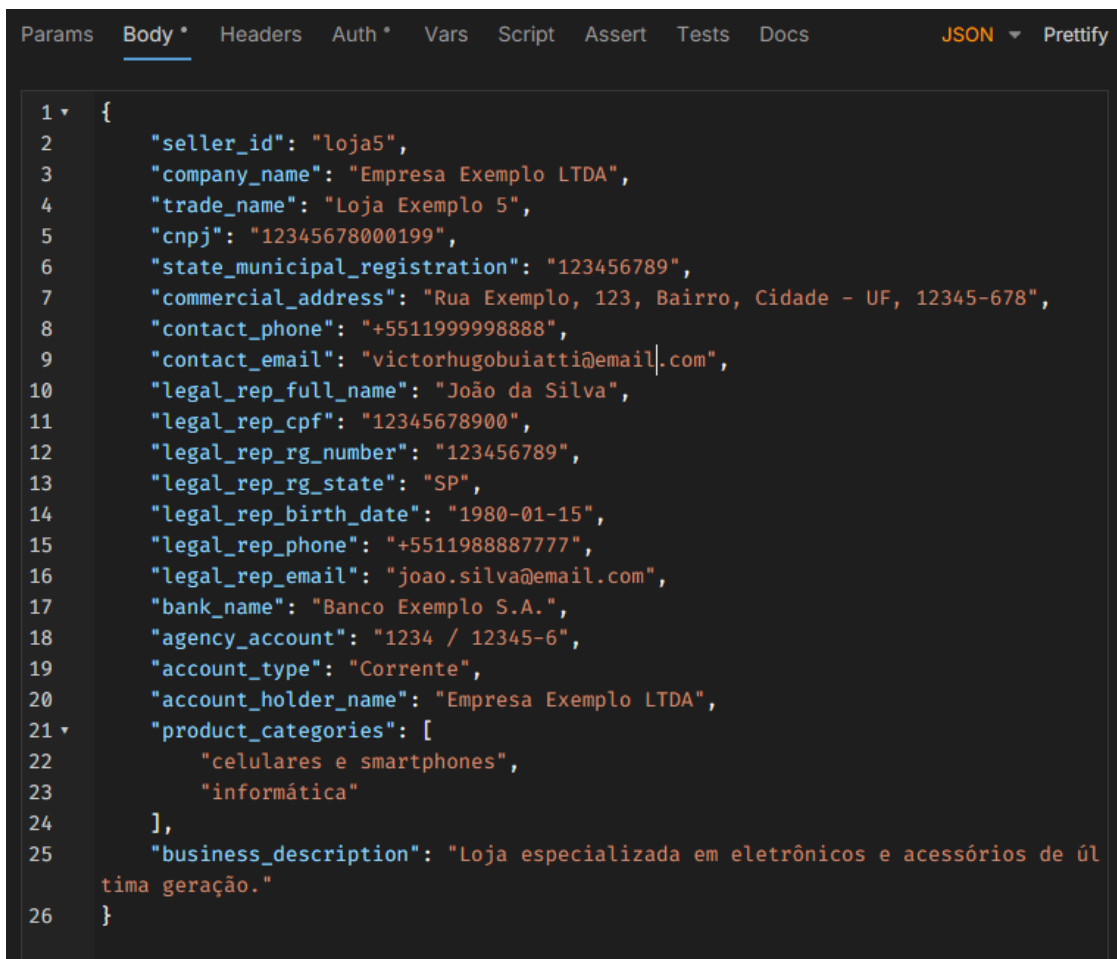
Utilizar **METHOD** do tipo **POST**.

Para criar um novo seller, você precisa estar autenticado. A requisição deve incluir um **Authorization** header com um Bearer Token JWT válido. O **seller_id** criado será automaticamente associado ao usuário do token.

Headers:

- Content-Type: application/json
- Authorization: Bearer <seu_token_jwt_aqui>

Abaixo exibiremos o método aberto através do **Bruno**. A chamada da API ocorre da seguinte forma:



```
Params  Body *  Headers  Auth *  Vars  Script  Assert  Tests  Docs  JSON ▾  Prettify

1  {
2    "seller_id": "loja5",
3    "company_name": "Empresa Exemplo LTDA",
4    "trade_name": "Loja Exemplo 5",
5    "cnpj": "12345678000199",
6    "state_municipal_registration": "123456789",
7    "commercial_address": "Rua Exemplo, 123, Bairro, Cidade - UF, 12345-678",
8    "contact_phone": "+5511999998888",
9    "contact_email": "victorhugobuiatti@email.com",
10   "legal_rep_full_name": "João da Silva",
11   "legal_rep_cpf": "12345678900",
12   "legal_rep_rg_number": "123456789",
13   "legal_rep_rg_state": "SP",
14   "legal_rep_birth_date": "1980-01-15",
15   "legal_rep_phone": "+5511988887777",
16   "legal_rep_email": "joao.silva@email.com",
17   "bank_name": "Banco Exemplo S.A.",
18   "agency_account": "1234 / 12345-6",
19   "account_type": "Corrente",
20   "account_holder_name": "Empresa Exemplo LTDA",
21   "product_categories": [
22     "celulares e smartphones",
23     "informática"
24   ],
25   "business_description": "Loja especializada em eletrônicos e acessórios de última geração."
26 }
```

Parâmetros de Entrada

Tt Campo	Tipo	Tt Obrigatório	Tt Descrição
seller_id	String ▾	Sim	ID único do seller no sistema.
company_name	String ▾	Sim	Razão Social da empresa.
cnpj	String ▾	Sim	CNPJ do seller
trade_name	String ▾	Sim	Nome fantasia do seller. Deve ser único.
commercial_address	String ▾	Sim	Endereço comercial da empresa.
state_municipal_registration	String ▾	Sim	Inscrição Estadual/Municipal.
contact_phone	String ▾	Sim	Telefone de contato principal.
contact_email	Email ▾	Sim	E-mail de contato principal.
legal_rep_full_name	String ▾	Sim	Nome completo do representante legal.
legal_rep_cpf	String ▾	Sim	CPF do representante legal.
legal_rep_rg_number	String ▾	Sim	Número do RG do representante legal.
legal_rep_rg_state	String ▾	Sim	Estado emissor do RG (ex: "SP", "RJ").
legal_rep_birth_date	DateTime ▾	Sim	Data de nascimento do representante.
legal_rep_phone	String ▾	Sim	Telefone do representante legal.
legal_rep_email	Email ▾	Sim	E-mail do representante legal.
bank_name	String ▾	Sim	Nome do banco.
agency_account	String ▾	Sim	Agência e conta bancária.

Parâmetros de Entrada

Tt Campo	☰ Tipo	Tt Obrigatório	Tt Descrição
account_holder_name	String ▾	Sim	Nome do titular da conta.
account_type	String ▾	Sim	Tipo de conta (ex: "Corrente", "Poupança").
product_categories	List[Enum] ▾	Sim	Lista com as categorias de produtos.
business_description	String ▾	Sim	Breve descrição do negócio do seller.

Saída esperada (**Resposta 201 Created**):

ResponseHeaders⁵TimelineTests

🔗 ⬇️ 201 Created2.43s861B

```
1 {
2   "seller_id": "loja5",
3   "company_name": "Empresa Exemplo LTDA",
4   "trade_name": "Loja Exemplo 5",
5   "cnpj": "12345678000199",
6   "state_municipal_registration": "123456789",
7   "commercial_address": "Rua Exemplo, 123, Bairro, Cidade - UF, 12345-678",
8   "contact_phone": "5511999998888",
9   "contact_email": "victorhugobuiatti@gmail.com",
10  "legal_rep_full_name": "João da Silva",
11  "legal_rep_cpf": "12345678900",
12  "legal_rep_rg_number": "123456789",
13  "legal_rep_rg_state": "SP",
14  "legal_rep_birth_date": "1980-01-15",
15  "legal_rep_phone": "5511988887777",
16  "legal_rep_email": "joao.silva@email.com",
17  "bank_name": "banco exemplo s.a.",
18  "agency_account": "1234 / 12345-6",
19  "account_type": "Corrente",
20  "account_holder_name": "Empresa Exemplo LTDA",
21  "product_categories": [
22    "celulares e smartphones",
23    "informática"
24  ],
25  "business_description": "Loja especializada em eletrônicos e acessórios de última geração.",
26  "status": "Ativo"
27 }
```

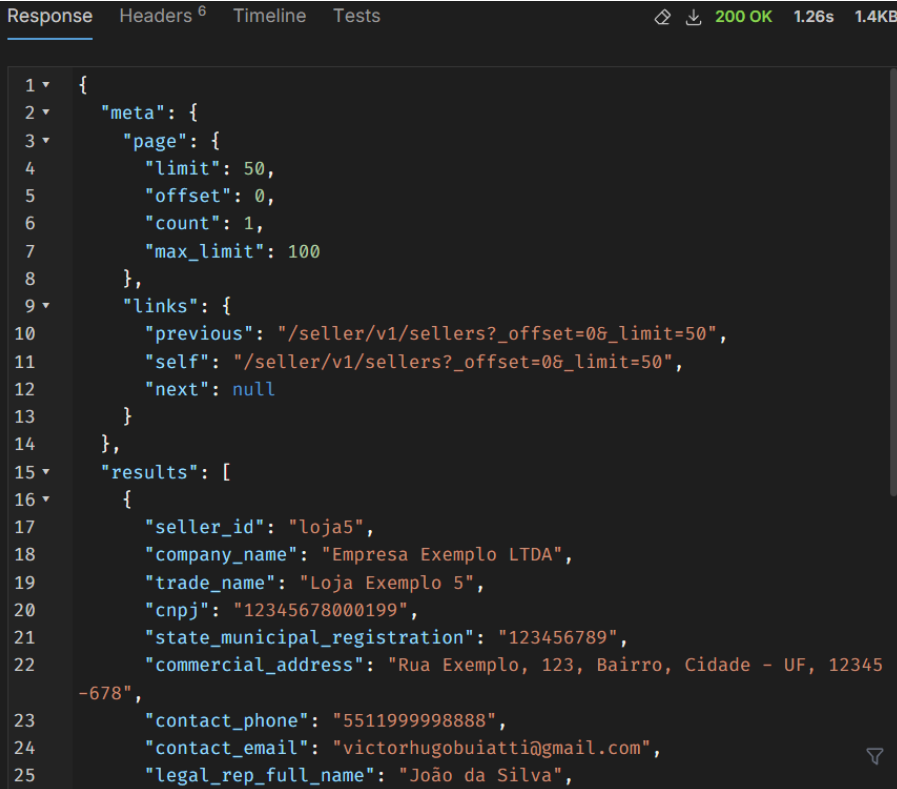
2. 📄 Listar Todos os Sellers

GET /seller/v1/sellers

Descrição: Retorna todos os sellers cadastrados.

Utilizar **METHOD** do tipo **GET**.

Saída esperada (**Resposta 200 OK**):



```
1 {
2   "meta": {
3     "page": {
4       "limit": 50,
5       "offset": 0,
6       "count": 1,
7       "max_limit": 100
8     },
9     "links": {
10      "previous": "/seller/v1/sellers?_offset=0&_limit=50",
11      "self": "/seller/v1/sellers?_offset=0&_limit=50",
12      "next": null
13    }
14  },
15  "results": [
16    {
17      "seller_id": "loja5",
18      "company_name": "Empresa Exemplo LTDA",
19      "trade_name": "Loja Exemplo 5",
20      "cnpj": "12345678000199",
21      "state_municipal_registration": "123456789",
22      "commercial_address": "Rua Exemplo, 123, Bairro, Cidade - UF, 12345
-678",
23      "contact_phone": "5511999998888",
24      "contact_email": "victorhugobuiatti@gmail.com",
25      "legal_rep_full_name": "João da Silva",
```

3. 🔍 Obter Seller por ID

GET /seller/v1/sellers/{seller_id}

Descrição: Busca um seller específico pelo seu **seller_id**.

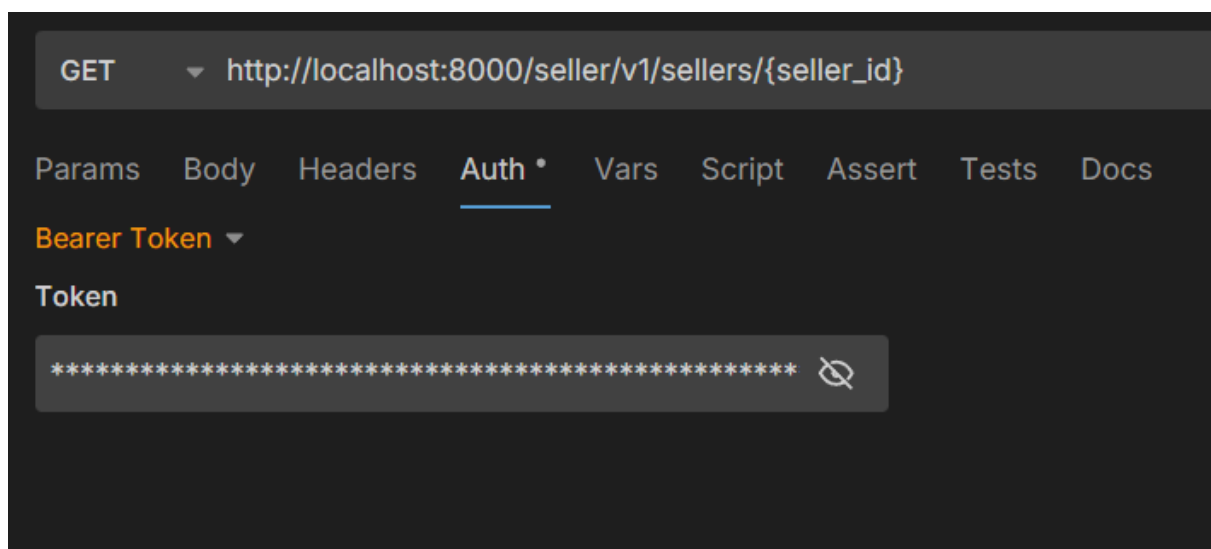
Utilizar **METHOD** do tipo **GET**.

Para buscar um seller, você precisa estar autenticado. A requisição deve incluir um **Authorization** header com um Bearer Token JWT válido. O **seller_id** deve estar associado ao usuário do Token.

Headers:

- Content-Type: application/json
- Authorization: Bearer <seu_token_jwt_aqui>

Abaixo exibiremos o método aberto através do **Bruno**. A chamada da API ocorre da seguinte forma:



Saída esperada (**Resposta 200 OK**):

```
Response Headers 5 Timeline Tests 200 OK 811ms 861B

1 {
2   "seller_id": "loja5",
3   "company_name": "Empresa Exemplo LTDA",
4   "trade_name": "Loja Exemplo 5",
5   "cnpj": "12345678000199",
6   "state_municipal_registration": "123456789",
7   "commercial_address": "Rua Exemplo, 123, Bairro, Cidade - UF, 12345-678",
8   "contact_phone": "5511999998888",
9   "contact_email": "victorhugobuiatti@gmail.com",
10  "legal_rep_full_name": "João da Silva",
11  "legal_rep_cpf": "12345678900",
12  "legal_rep_rg_number": "123456789",
13  "legal_rep_rg_state": "RJ",
14  "legal_rep_birth_date": "1980-01-15",
15  "legal_rep_phone": "5511988887777",
16  "legal_rep_email": "joao.silva@email.com",
17  "bank_name": "banco exemplo s.a.",
18  "agency_account": "1234 / 12345-6",
19  "account_type": "Corrente",
20  "account_holder_name": "Empresa Exemplo LTDA",
21  "product_categories": [
22    "celulares e smartphones",
23    "informática"
24  ],
25  "business_description": "Loja especializada em eletrônicos e acessórios"
```

4. 🖋️ Atualizar Seller (Parcial)

PATCH `/seller/v1/sellers/{seller_id}`

Descrição: Atualiza um ou mais campos de um seller existente. Esta operação é uma atualização parcial, o que significa que você só precisa enviar no corpo da requisição os campos que deseja alterar. Campos não enviados não serão modificados.

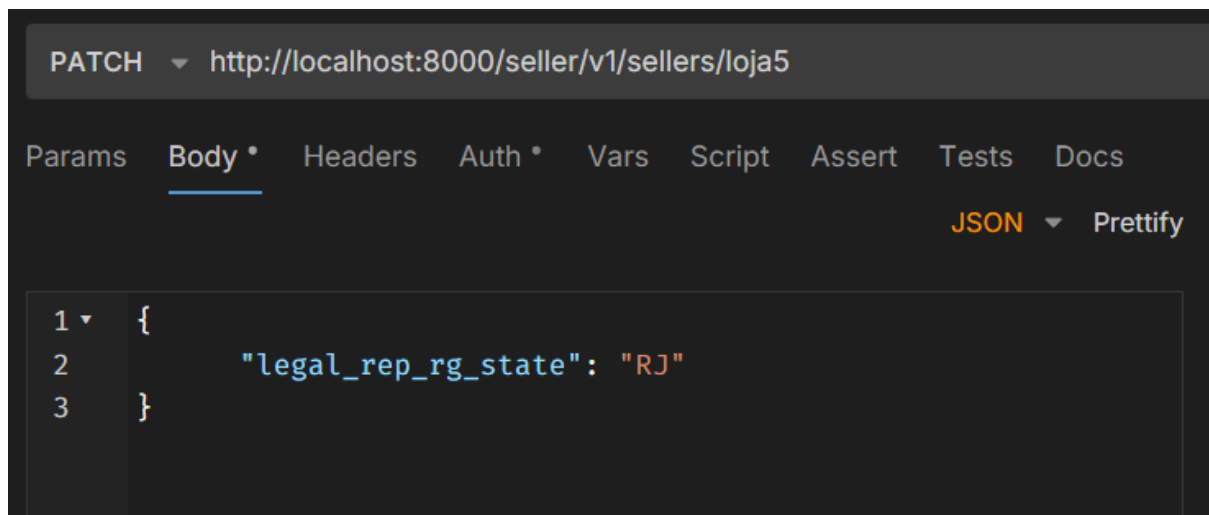
Utilizar **METHOD** do tipo **PATCH**.

Autorização Obrigatória: A requisição deve incluir um **Authorization** header com um Bearer Token JWT válido. O usuário associado ao token deve ter permissão para o **seller_id** especificado na URL.

Headers:

- Content-Type: application/json
- Authorization: Bearer <seu_token_jwt_aqui>

Abaixo exibiremos o método aberto através do **Bruno**. A chamada da API ocorre da seguinte forma:



Exemplos de Respostas:

Resposta 200 OK:

```
Response Headers 5 Timeline Tests 200 OK 631ms 861B
1 {
2   "seller_id": "loja5",
3   "company_name": "Empresa Exemplo LTDA",
4   "trade_name": "Loja Exemplo 5",
5   "cnpj": "12345678000199",
6   "state_municipal_registration": "123456789",
7   "commercial_address": "Rua Exemplo, 123, Bairro, Cidade - UF, 12345-678",
8   "contact_phone": "5511999998888",
9   "contact_email": "victorhugobuiatti@gmail.com",
10  "legal_rep_full_name": "João da Silva",
11  "legal_rep_cpf": "12345678900",
12  "legal_rep_rg_number": "123456789",
13  "legal_rep_rg_state": "RJ",
14  "legal_rep_birth_date": "1980-01-15",
15  "legal_rep_phone": "5511988887777",
16  "legal_rep_email": "joao.silva@email.com",
17  "bank_name": "banco exemplo s.a.",
18  "agency_account": "1234 / 12345-6",
19  "account_type": "Corrente",
20  "account_holder_name": "Empresa Exemplo LTDA",
21  "product_categories": [
22    "celulares e smartphones",
23    "informática"
24  ],
25  "business_description": "Loja especializada em eletrônicos e acessórios"
```

401 Unauthorized - Não Autenticado

Retornado se o token JWT não for fornecido, for inválido ou tiver expirado.

403 Forbidden - Acesso Proibido

Retornado se o usuário autenticado não tiver permissão para modificar o **seller_id** especificado.

404 Not Found - Não Encontrado

Retornado se nenhum seller for encontrado com o **user_id** fornecido na URL.

422 Unprocessable Entity - Dados Inválidos

Retornado se os dados no corpo da requisição falharem na validação (ex: um CNPJ com formato incorreto).

5. Atualizar Seller

PUT /seller/v1/sellers/{seller_id}

Descrição: Substitui completamente os dados de um seller existente por um novo conjunto de informações. Esta operação requer que **todos** os campos da entidade **Seller** sejam enviados no corpo da requisição. Campos não enviados serão zerados ou removidos, conforme a lógica do sistema.

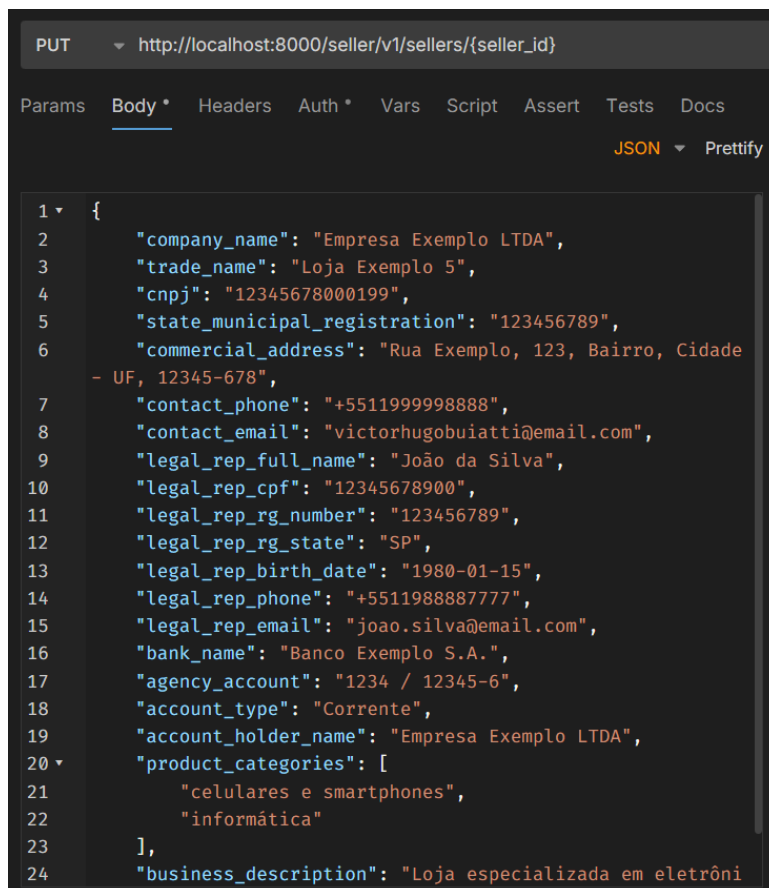
Utilizar **METHOD** do tipo **PUT**.

Autorização Obrigatória: A requisição deve incluir um **Authorization** header com um Bearer Token JWT válido. O usuário associado ao token deve ter permissão para o **seller_id** especificado na URL.

Headers:

- Content-Type: application/json
- Authorization: Bearer <seu_token_jwt_aqui>

Abaixo exibiremos o método aberto através do **Bruno**. A chamada da API ocorre da seguinte forma:



Exemplos de Respostas:

200 OK - Sucesso

Retornado quando o seller é substituído com sucesso. O corpo da resposta conterá a nova representação completa do recurso.

401 Unauthorized - Não Autenticado

Retornado se o token JWT não for fornecido, for inválido ou tiver expirado.

403 Forbidden - Acesso Proibido

Retornado se o usuário autenticado não tiver permissão para modificar o **seller_id** especificado.

404 Not Found - Não Encontrado

Retornado se nenhum seller for encontrado com o **seller_id** fornecido na URL.

422 Unprocessable Entity - Dados Inválidos

Retornado se o corpo da requisição estiver incompleto (faltando algum campo obrigatório) ou contiver dados que não passam na validação (ex: um CPF com formato inválido).

6. 🗑 Deletar Seller

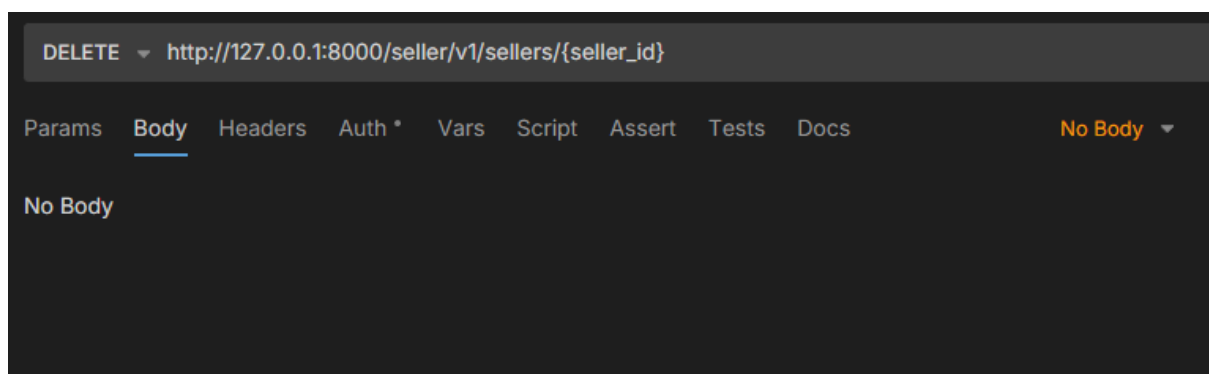
DELETE /seller/v1/sellers/{seller_id}

Descrição: Marca um seller como "Inativo" no banco de dados e remove o acesso do usuário autenticado a este seller no sistema de identidade (**Keycloak**). Esta operação **não apaga** o registro permanentemente, preservando o histórico.

Utilizar **METHOD** do tipo **DELETE**.

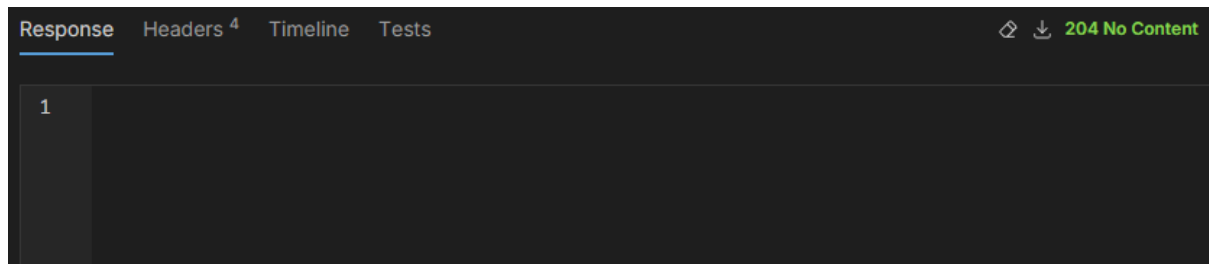
Autorização Obrigatória: A requisição deve incluir um **Authorization** header com um Bearer Token JWT válido. O usuário associado ao token deve ter permissão para o **seller_id** especificado na URL para poder inativá-lo.

Abaixo exibiremos o método aberto através do **Bruno**. A chamada da API ocorre da seguinte forma:



Exemplos de Respostas:

Saída esperada (**Resposta 204 No Content**):



401 Unauthorized - Não Autenticado

Retornado se o token JWT não for fornecido, for inválido ou tiver expirado.

403 Forbidden - Acesso Proibido

Retornado se o usuário autenticado não tiver permissão para modificar o **seller_id** especificado.

404 Not Found - Não Encontrado

Retornado se nenhum seller for encontrado com o **seller_id** fornecido na URL.

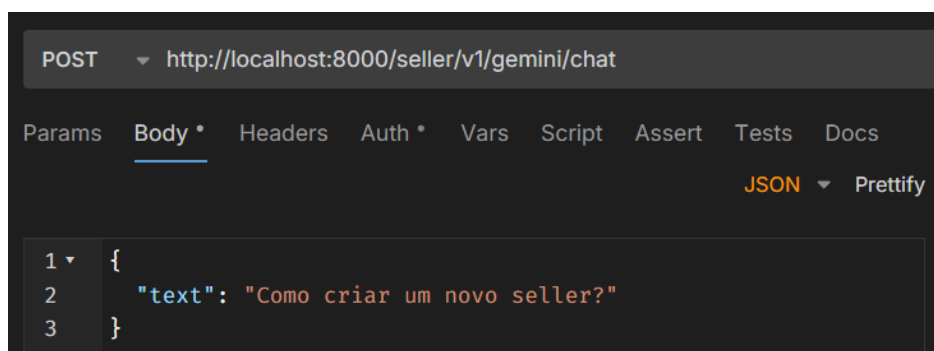
Endpoint Chat (Gemini)

1. Chat de orientação do módulo

POST /seller/v1/gemini/chat

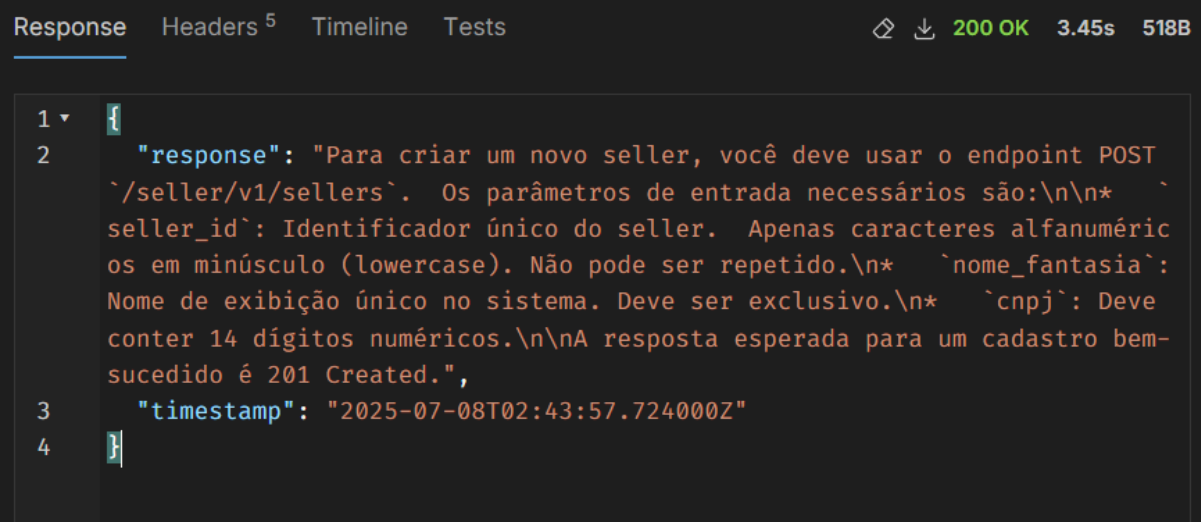
Descrição: mensagem enviada o chat

Abaixo exibiremos o método aberto através do **Bruno**. A chamada da API ocorre da seguinte forma:



Exemplo de Resposta:

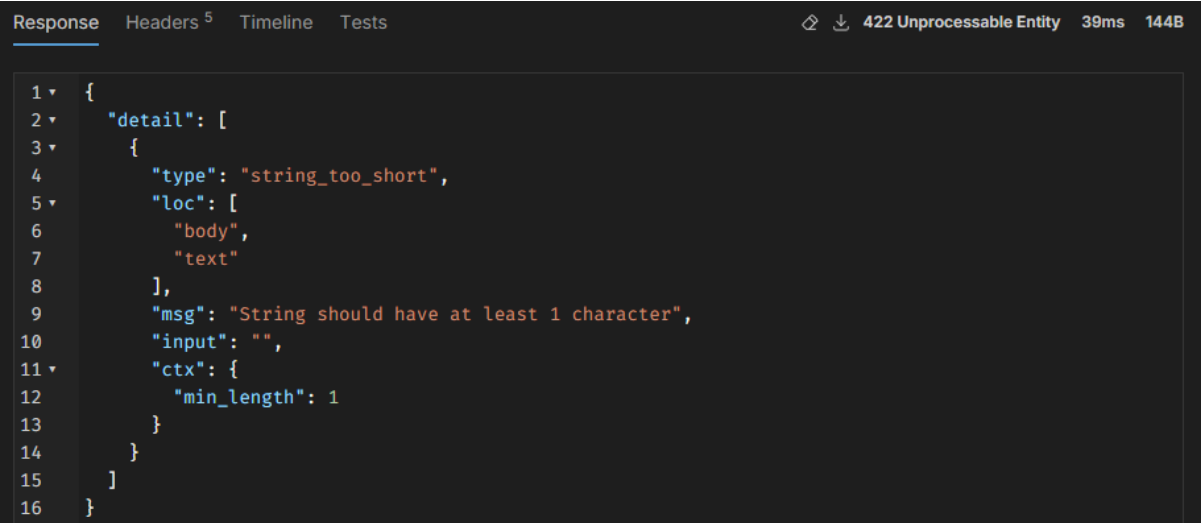
200 OK:



The screenshot shows a REST client interface with tabs for Response, Headers, Timeline, and Tests. The Response tab is active, displaying a 200 OK status with a response time of 3.45s and a body size of 518B. The JSON body is as follows:

```
1 {
2   "response": "Para criar um novo seller, voc\u00ea deve usar o endpoint POST
   \"/seller/v1/sellers`. Os par\u00e2metros de entrada necess\u00e1rios s\u00e3o:\n\n* `
   seller_id`: Identificador \u00fanico do seller. Apenas caracteres alfanum\u00e9ric
   os em min\u00fasculo (lowercase). N\u00e3o pode ser repetido.\n* `nome_fantasia`:
   Nome de exibição \u00fanico no sistema. Deve ser exclusivo.\n* `cnpj`: Deve
   conter 14 d\u00edgitos num\u00e9ricos.\n\nA resposta esperada para um cadastro bem-
   sucedido \u00e9 201 Created.",
3   "timestamp": "2025-07-08T02:43:57.724000Z"
4 }
```

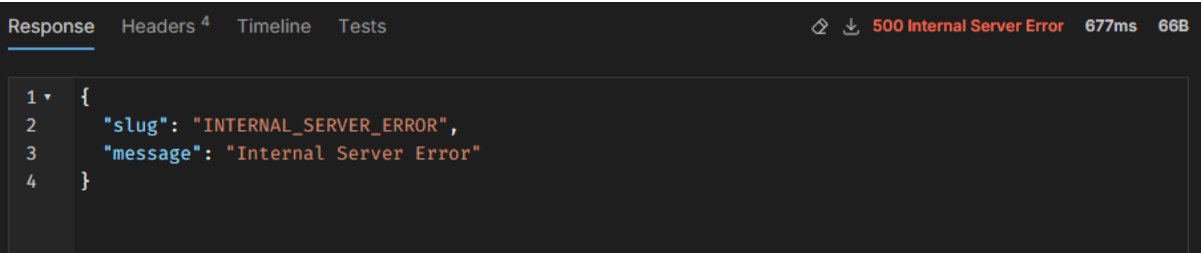
422 Unprocessable Entity - Dados Inv\u00e1lidos



The screenshot shows a REST client interface with tabs for Response, Headers, Timeline, and Tests. The Response tab is active, displaying a 422 Unprocessable Entity status with a response time of 39ms and a body size of 144B. The JSON body is as follows:

```
1 {
2   "detail": [
3     {
4       "type": "string_too_short",
5       "loc": [
6         "body",
7         "text"
8       ],
9       "msg": "String should have at least 1 character",
10      "input": "",
11      "ctx": {
12        "min_length": 1
13      }
14    }
15  ]
16 }
```

500 Internal Server Error - Erro inesperado na aplica\u00e7\u00e3o



The screenshot shows a REST client interface with tabs for Response, Headers, Timeline, and Tests. The Response tab is active, displaying a 500 Internal Server Error status with a response time of 677ms and a body size of 66B. The JSON body is as follows:

```
1 {
2   "slug": "INTERNAL_SERVER_ERROR",
3   "message": "Internal Server Error"
4 }
```