**E-BANK**

**(Documentação API)**

**Equipe 4:**

**Igor Silva Sobral**

**Israel Davi Travassos Verissimo**

**João Lucas Dantas Nunes Gomes**

Sumário

[**1. Sobre** 3](#_Toc174606008)

[**2. Introdução** 3](#_Toc174606009)

[**3. Diagrama Entidade Relacionamento** 3](#_Toc174606010)

[**4. Premissas acordadas** 4](#_Toc174606011)

[**5. EndPoints do projeto** 4](#_Toc174606012)

[**6. Como utilizar a API** 5](#_Toc174606013)

[**6.1 Configurar Base de dados** 5](#_Toc174606014)

[**6.2 Autenticação** 6](#_Toc174606015)

[**6.3 Registro de usuário** 7](#_Toc174606016)

[**6.4 Listagem de Agências** 8](#_Toc174606017)

[**6.5 Postagem de Agência** 9](#_Toc174606018)

[**6.6 Agência por id** 9](#_Toc174606019)

[**6.7 Delete agência** 10](#_Toc174606020)

[**6.8 Usuário por id.** 10](#_Toc174606021)

[**6.9 Atualizar usuário.** 11](#_Toc174606022)

[**6.10 Deletar usuário.** 12](#_Toc174606023)

[**6.11 Acessar Extrato.** 12](#_Toc174606024)

[**6.12 Realizar Saque.** 13](#_Toc174606025)

[**6.13 Realizar Depósito.** 14](#_Toc174606026)

[**6.14 Realizar Transferência.** 14](#_Toc174606027)

[**6.15 Realizar Pix.** 15](#_Toc174606028)

[**7. Tecnologias Utilizadas** 15](#_Toc174606029)

[**8. Jacoco** 15](#_Toc174606030)

# **1. Sobre**

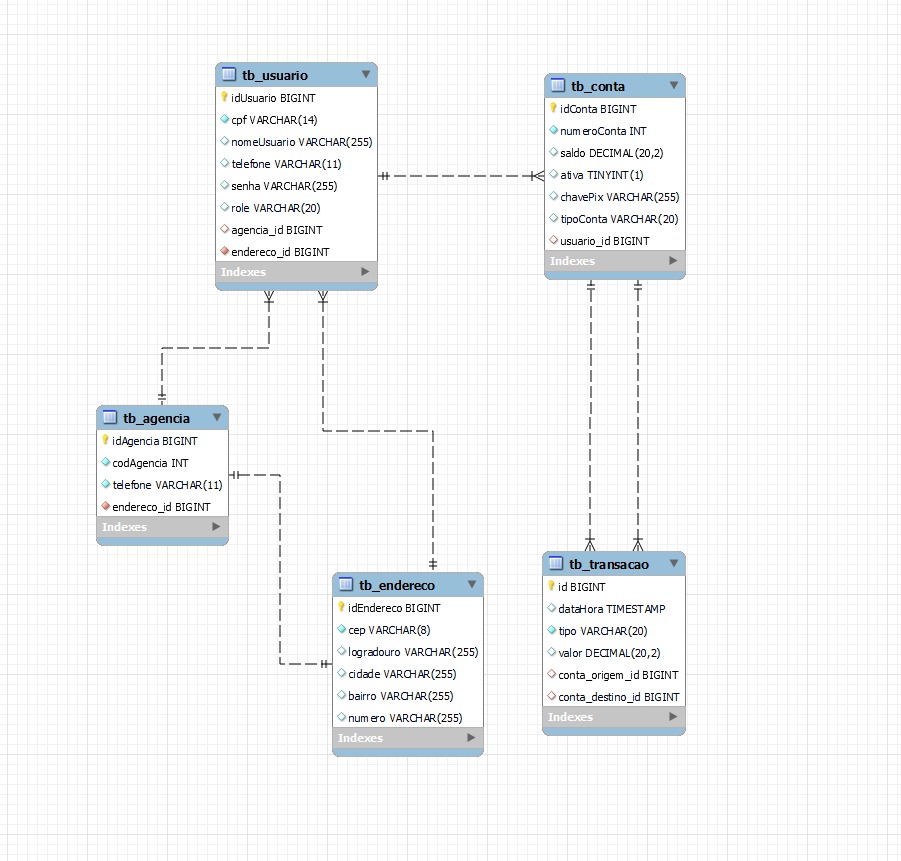
API para sistema bancário, consistindo em armazenar dados e simular operações comuns de bancos.

# **2. Introdução**

Este documento fornecerá instruções para rápida integração aos serviços do eBank via HTTP (HTTP API).

# **3. Diagrama Entidade Relacionamento**

Este tipo de diagrama é composto por entidades, atributos e relacionamentos, em que as entidades representam um objeto do mundo real e que possuem uma existência independente, segue abaixo a representação do projeto por meio do diagrama.

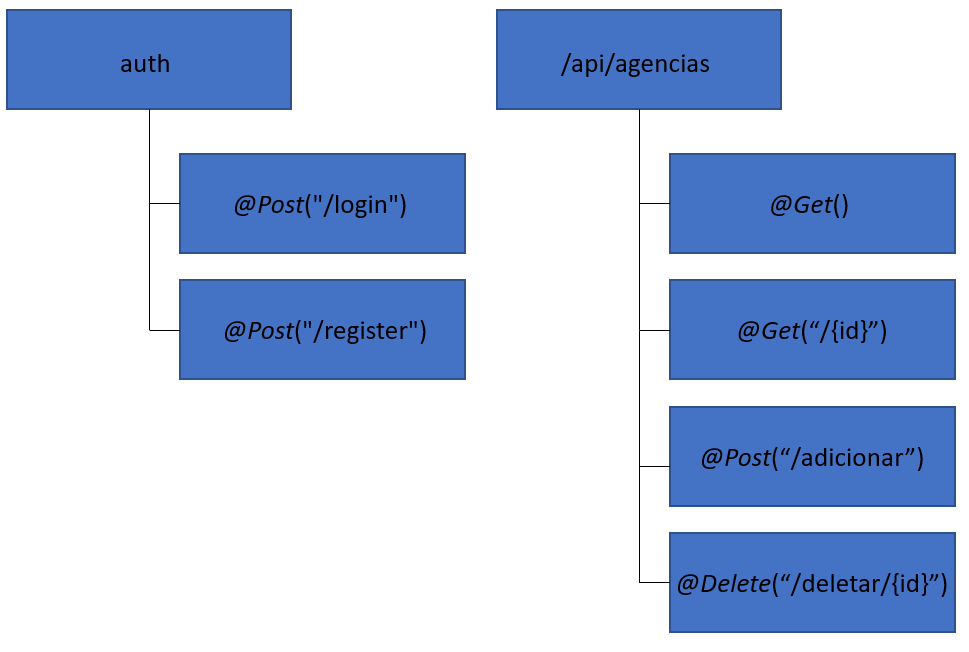


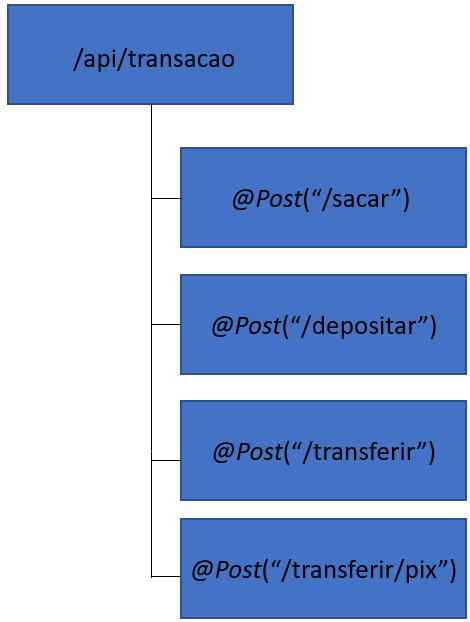
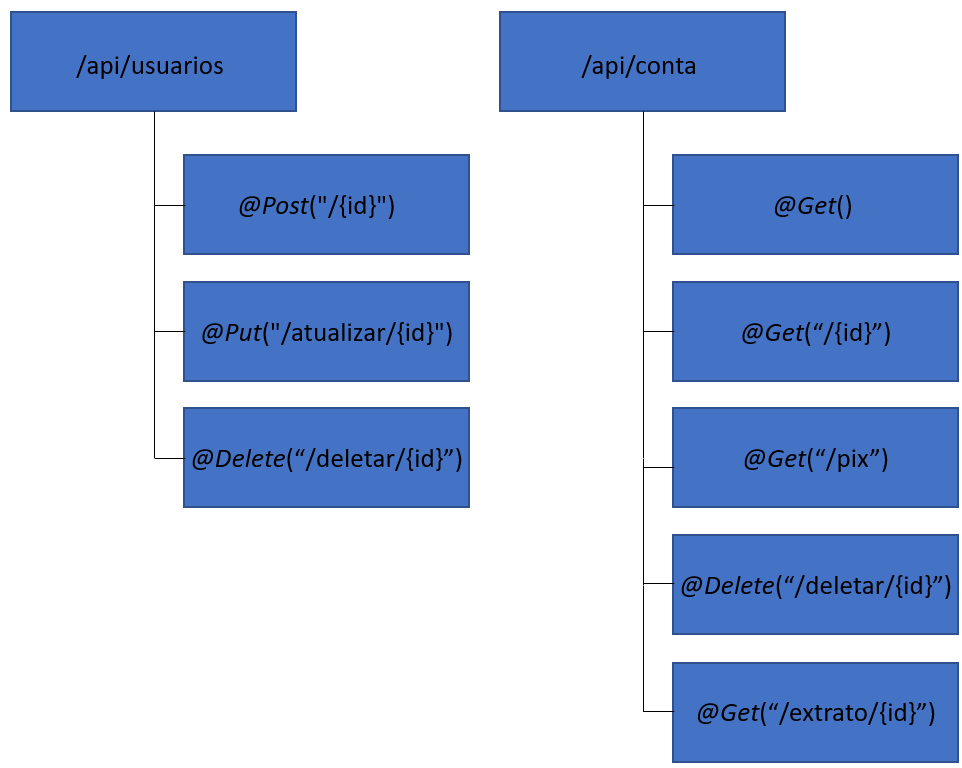
# **4. Premissas acordadas**

Para combater qualquer tipo de problemas de acesso as rotas, definimos limitações de uso:

* Tirando o Login, todo *endpoint* acessado na aplicação exige um token de autenticação;
* Apenas um usuário do tipo ADMIN pode criar um novo usuário;
* Apenar um usuário do tipo ADMIN tem acesso aos endpoints de Agencia.

# **5. EndPoints do projeto**

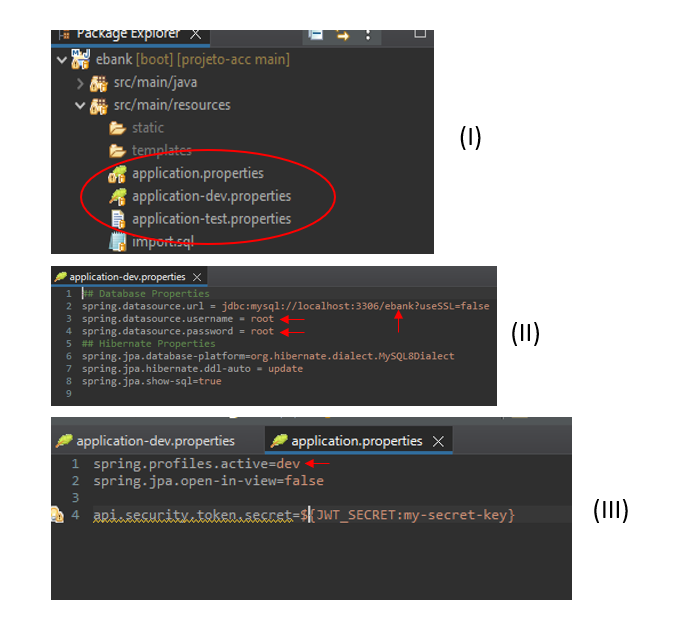




# **6. Como utilizar a API**

## **6.1 Configurar Base de dados**

A API, apresenta duas formas de armazenamento de dados. Ela pode rodar na versão teste (H2) e na versão desenvolvimento (MySQL). Caso você escolha rodar a aplicação com MySQL deve configurar os seguintes campos da Figura II com os dados correspondentes ao datasource do Banco de dados. Além disso deve deixar a Figura III no campo destacado com dev.



Além disso, deve carregar a base de dados com alguns dados para facilitar na hora de testar todos os endpoints, segue abaixo um arquivo com código SQL para inserir dados nas tabelas:



## **6.2 Autenticação**

Alguns dados foram inseridos via SQL como mostra a Figura IV para se ter alguns dados de teste, é possível observar que as senhas são encriptadas no Banco de dados. Para realizar Login é necessário informar cpf e senha, vale ressaltar que o cpf precisa ser informado no formato (xxx.xxx.xxx-xx).

O retorno é um token, que será necessário para autenticação do usuário no sistema, e ele só conseguirá acessar as outras rotas informando o token.

Para Logar, será feita uma requisição do tipo POST para a url/auth/login

Request body:

{

    "cpf": "123.456.789-00",

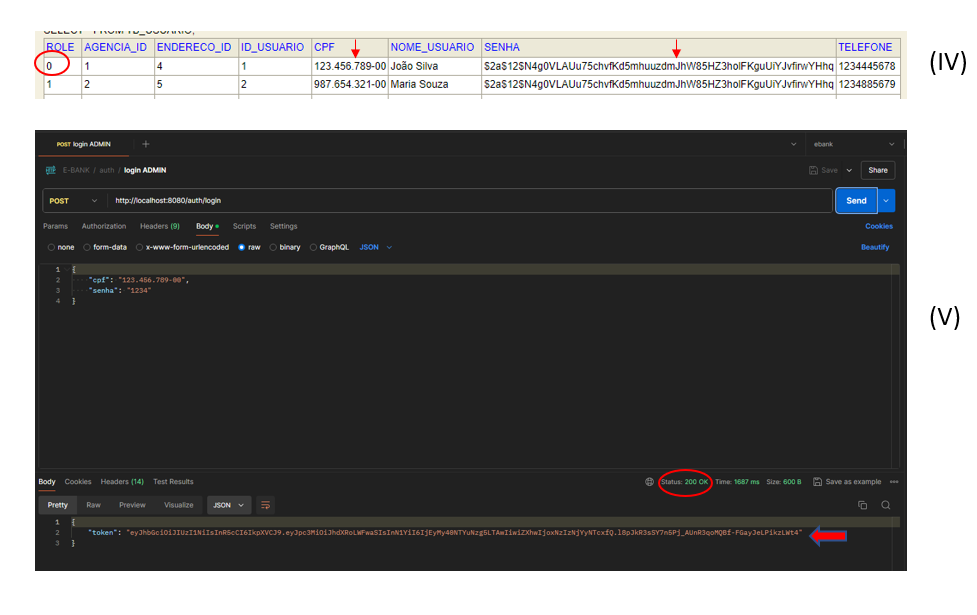
    "senha": "1234"

}

Respostas esperadas:

HTTP200: Sucesso (com body usado acima)

HTTP500: Usuário inexistente ou senha inválida



## **6.3 Registro de usuário**

Para Registrar um usuário, será feita uma requisição do tipo POST para a url/auth/register

Request body:

{

  "cpf": "987.655.341-00",

  "nomeUsuario": "Maria Oliveira",

  "telefone": "8398765432",

  "senha": "senhaSegura",

  "role": "USER",

  "idAgencia": 1,

  "endereco": {

    "cep": "56789-122",

    "logradouro": "Avenida Central",

    "cidade": "Cidade Nova",

    "bairro": "Centro",

    "numero": "789"

  },

  "contas": [

    {

      "saldo": 3000.00,

      "ativa": true,

      "chavePix": "chave-pix-nova",

      "tipoConta": "POUPANCA"

    }

  ]

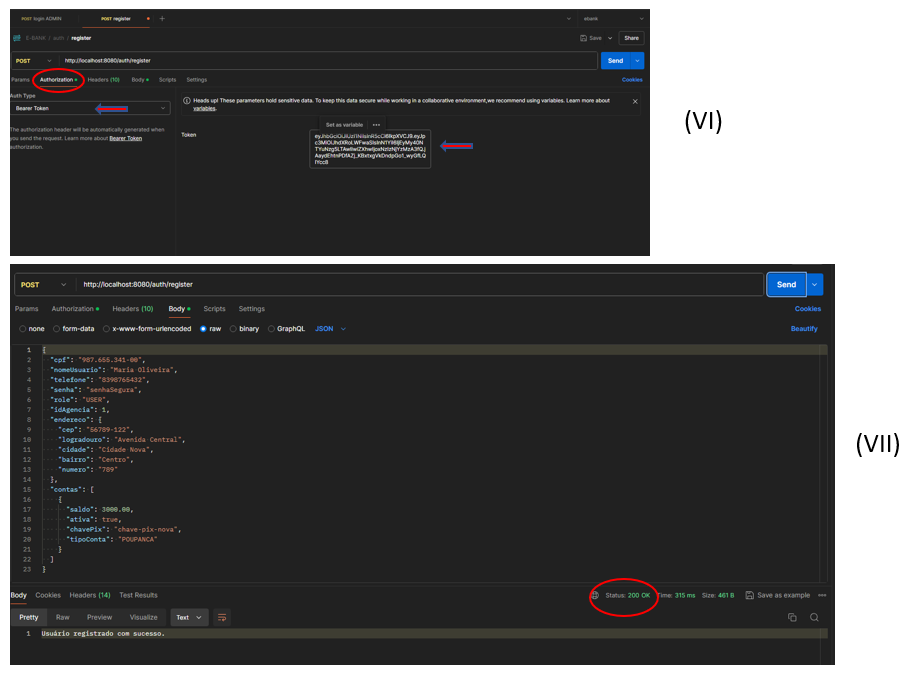
}

Respostas esperadas:

HTTP200: Sucesso (com body usado acima).

HTTP500: Se algum dado estiver incoerente com o esperado (cpf, cep, telefone, role, tipoConta).

HTTP403: Forbidden, caso um usuário não esteja logado ou um usuário do tipo USER tente cadastrar. (ROTA EXCLUSIVA DE USUÁRIOS DO TIPO ADMIN).

****

## **6.4 Listagem de Agências**

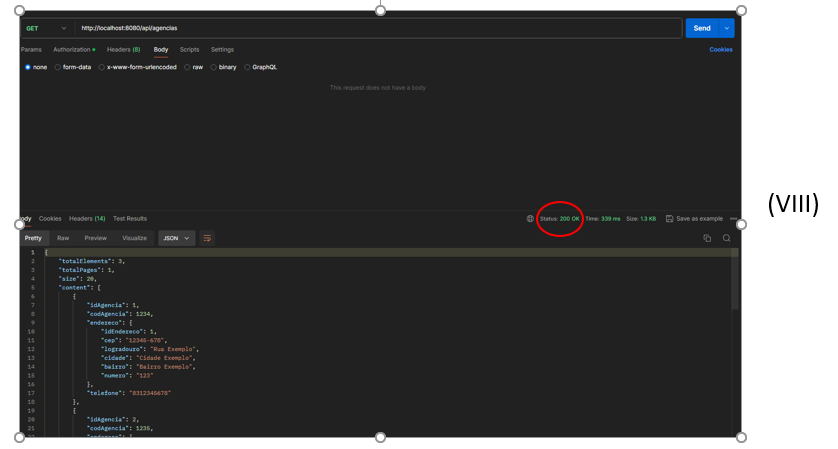
Para Listar as agências, será feita uma requisição do tipo GET para a url/api/agencias

Respostas esperadas:

HTTP200: Sucesso (Caso o usuário esteja logado e seja do tipo ADMIN).

HTTP403: Forbidden, caso esteja logado, mas seja do tipo USER.

HTTP401: Caso o usuário não esteja autenticado.



## **6.5 Postagem de Agência**

Para Registrar um usuário, será feita uma requisição do tipo POST para a url/api/agencias/adicionar

Request body:

{

            "codAgencia": 1288,

            "endereco": {

                "cep": "12222-000",

                "logradouro": "Rua Tabajaras",

                "cidade": "Cidade dos loucos",

                "bairro": "Bairro Espirito Santo",

                "numero": "11"

            },

            "telefone": "8312224622"

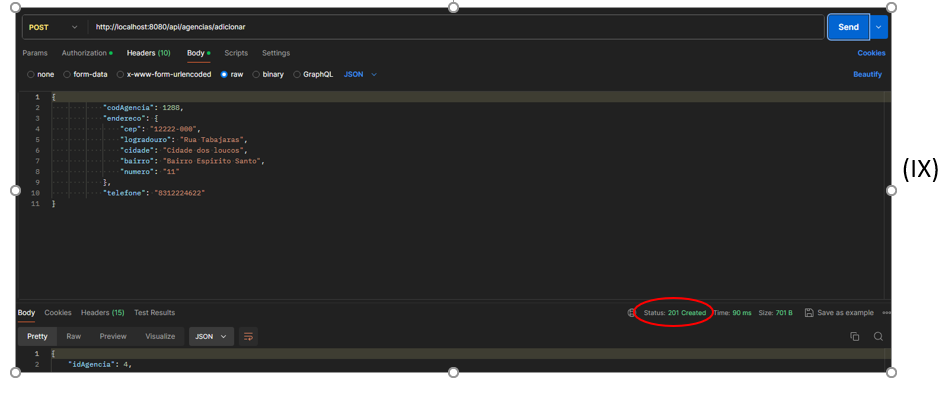
}

Respostas esperadas:

HTTP201: Sucesso (com body usado acima).

HTTP500: Se algum dado estiver incoerente com o esperado (cep, telefone).

HTTP403: Forbidden, caso um usuário não esteja logado ou seja do tipo USER.



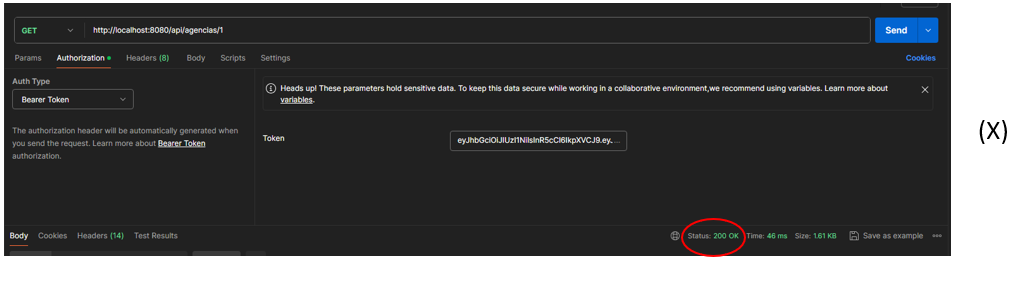
## **6.6 Agência por id**

Para buscar uma agência, será feita uma requisição do tipo GET para a url/api/agencias/{id}

Respostas esperadas:

HTTP200: Sucesso

HTTP403: Forbidden, caso um usuário não esteja logado ou seja do tipo USER.



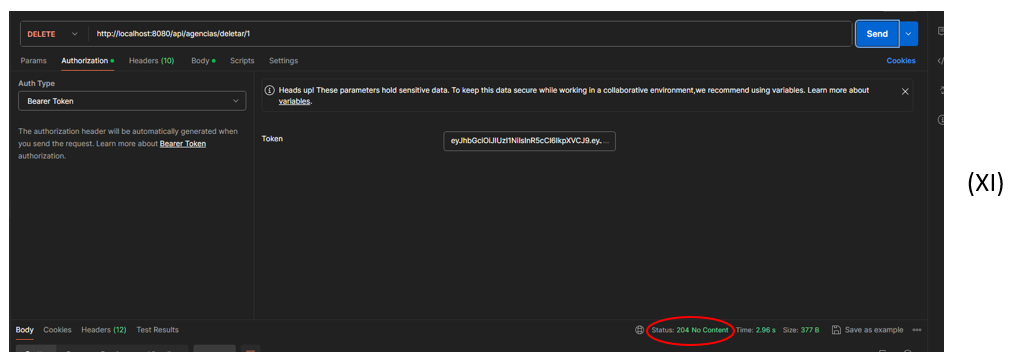
## **6.7 Delete agência**

Para Deletar uma agência, será feita uma requisição do tipo DELETE para a url/api/agencias/deletar/{id}

Respostas esperadas:

HTTP204: Sucesso

HTTP403: Forbidden, caso um usuário não esteja logado ou seja do tipo USER.



## **6.8 Usuário por id.**

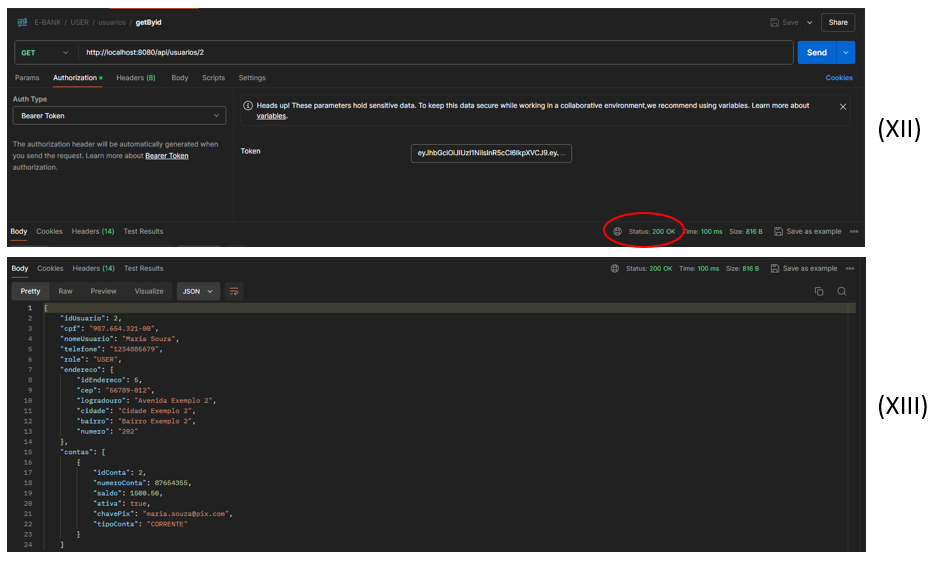
Para buscar um usuário, será feita uma requisição do tipo GET para a url/api/usuarios/{id}

Respostas esperadas:

HTTP200: Sucesso

HTTP403: Forbidden, tente acessar sem login.

HTTP500: Acesso negado, caso tente acessar os dados de um usuário que não corresponda ao seu login.



## **6.9 Atualizar usuário.**

Para atualizar um usuário, será feita uma requisição do tipo PUT para a url/api/usuarios/atualizar/{id}

Request body:

{

    "cpf": "144.888.972-06",

    "nomeUsuario": "homelander",

    "telefone": "88888888888",

    "senha": "123445",

    "endereco": {

        "cep": "56789-122",

        "logradouro": "Avenida Central",

        "cidade": "Cidade Nova",

        "bairro": "Centro",

        "numero": "789"

    }

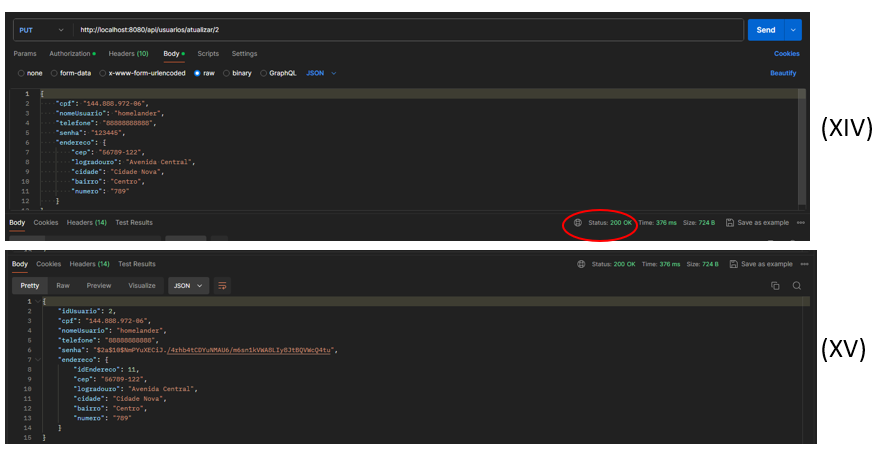
}

Respostas esperadas:

HTTP200: Sucesso (Com o body acima)

HTTP403: Forbidden, tente acessar sem login.

HTTP500: Acesso negado, caso tente acessar os dados de um usuário que não corresponda ao seu login.



## **6.10 Deletar usuário.**

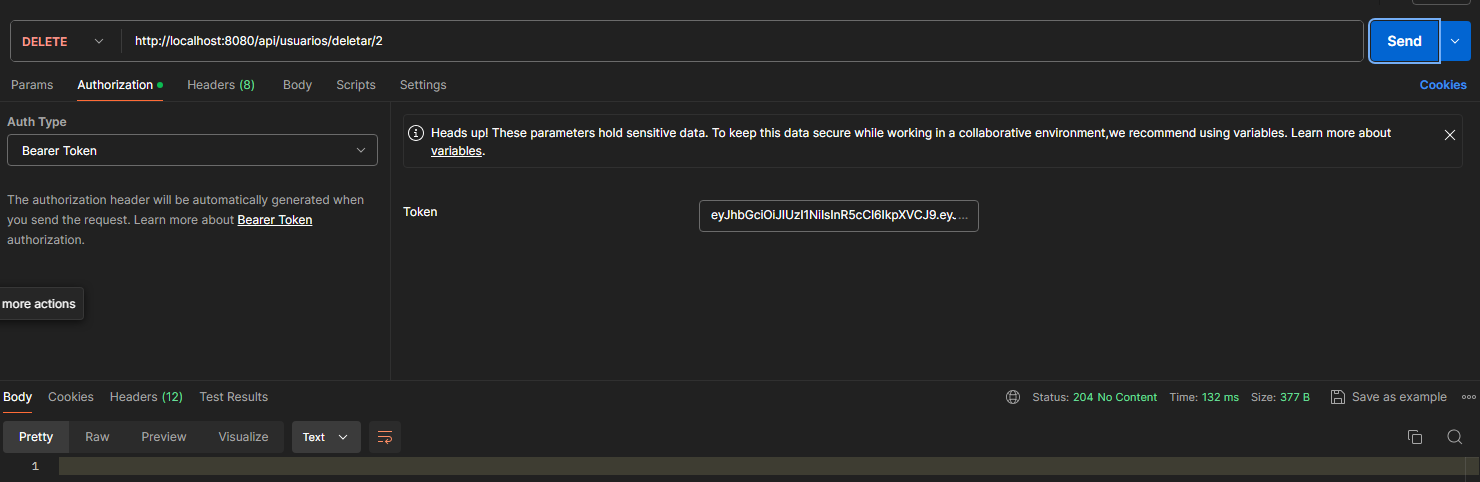
Para Deletar um usuário, será feita uma requisição do tipo DELETE para a url/api/usuarios/deletar/{id}

Respostas esperadas:

HTTP204: Sucesso

HTTP403: Forbidden, caso tente deletar um usuário sem estar logado.

HTTP500: Acesso negado, caso tente deletar um usuário diferente do logado.



## **6.11 Acessar Extrato.**

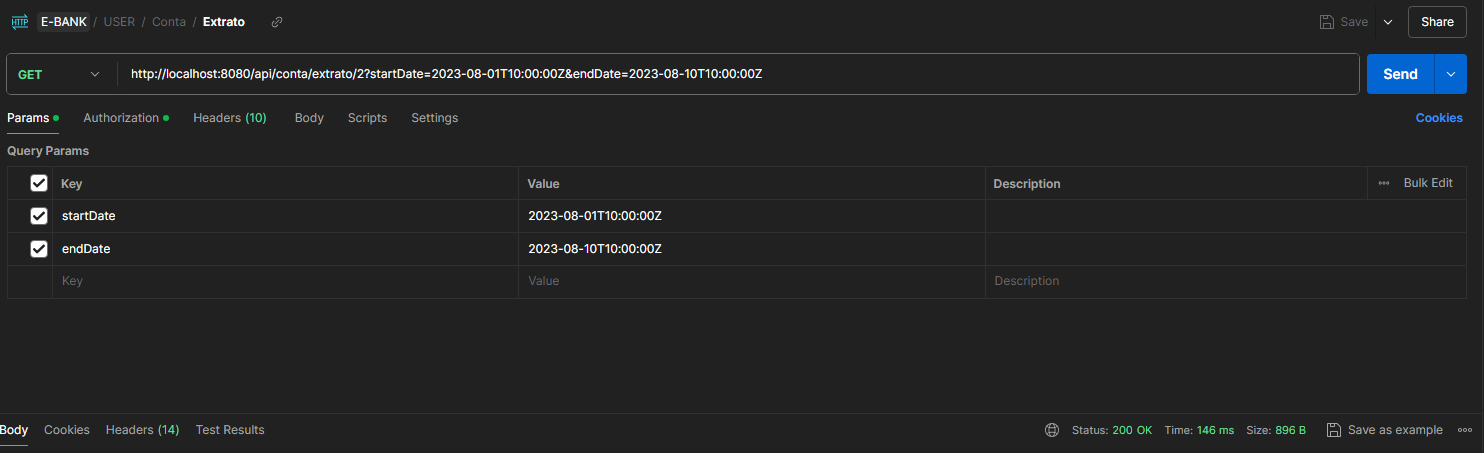
Para acessar o extrato, será feita uma requisição do tipo GET para a url/api/conta/extrato/{id}. (Utiliza Params)

Respostas esperadas:

HTTP200: Sucesso

HTTP403: Forbidden, caso um usuário tente acessar sem estar logado.

HTTP500: Acesso negado, caso um usuário tente acessar o extrato de outro usuário.



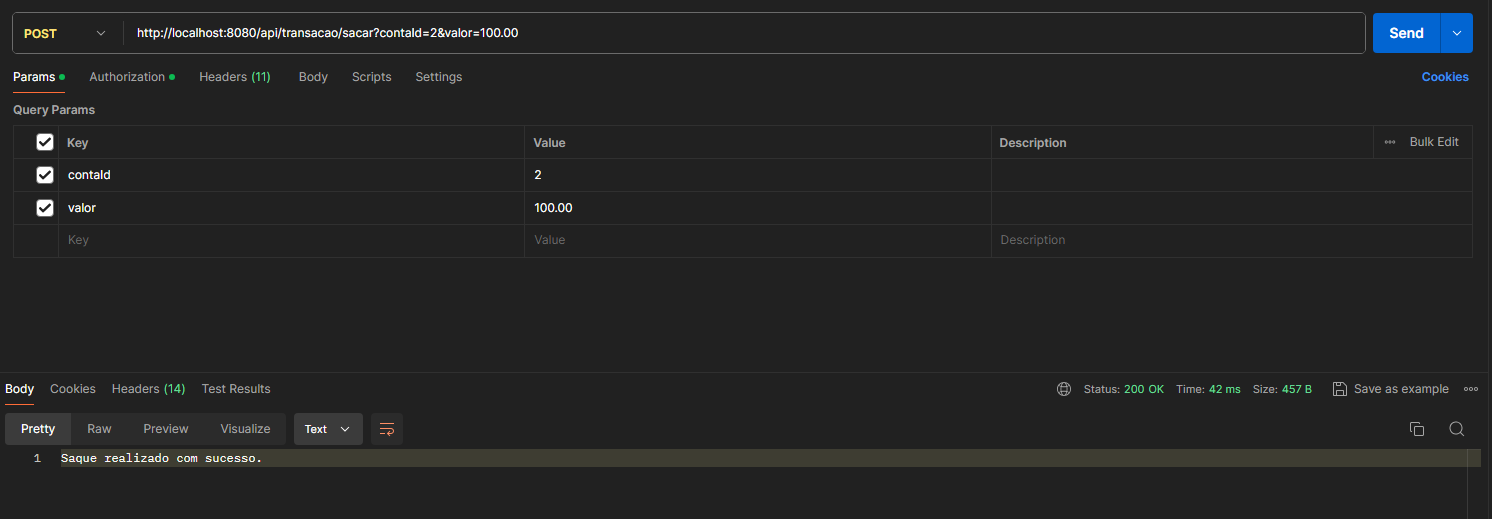
**6.12 Realizar Saque.**

Para realizar saque, será feita uma requisição do tipo POST para a url/api/transacao/sacar. (Utiliza Params)

Respostas esperadas:

HTTP20o: Sucesso (Retorna um comprovante)

HTTP403: Forbidden, caso um usuário tente acessar sem estar logado.



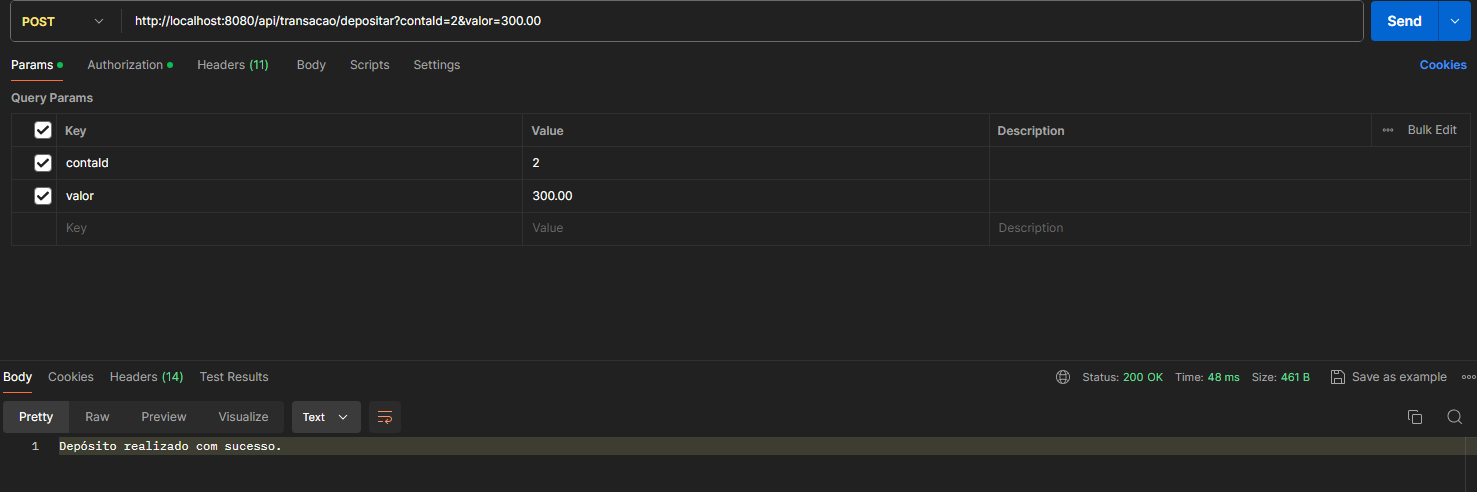
## **6.13 Realizar Depósito.**

Para realizar depósito, será feita uma requisição do tipo POST para a url/api/transacao/depositar. (Utiliza Params)

Respostas esperadas:

HTTP200: Sucesso (Retorna um comprovante)

HTTP403: Forbidden, caso um usuário tente acessar sem estar logado.



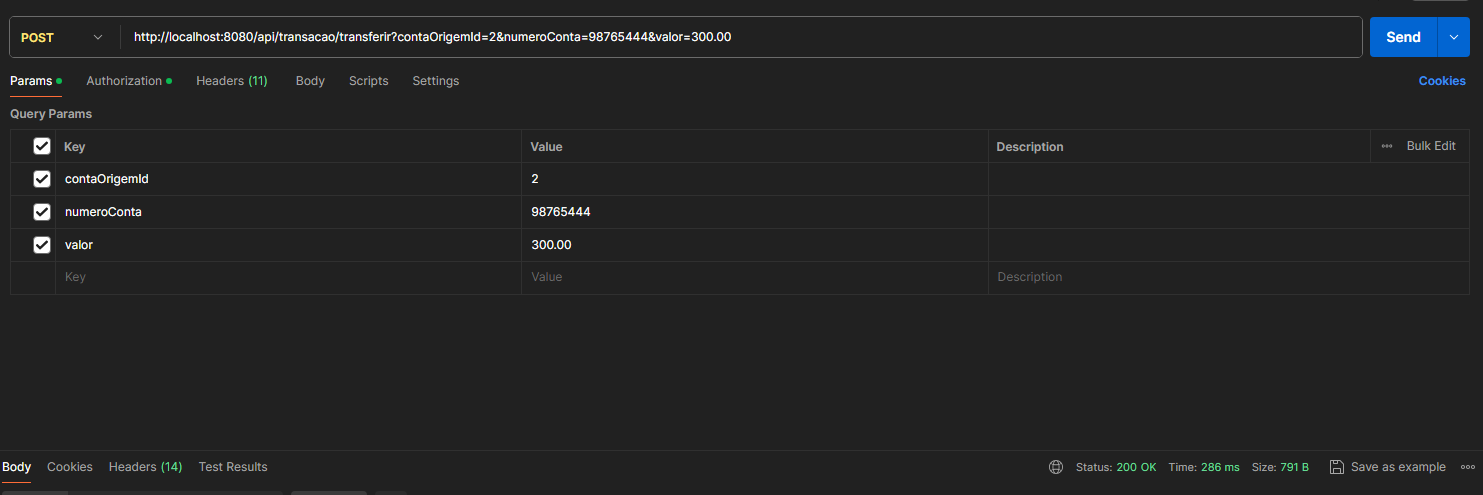
**6.14 Realizar Transferência.**

Para realizar transferência, será feita uma requisição do tipo POST para a url/api/transacao/transferir. (Utiliza Params)

Respostas esperadas:

HTTP200: Sucesso (Retorna um comprovante)

HTTP403: Forbidden, caso um usuário tente acessar sem estar logado.



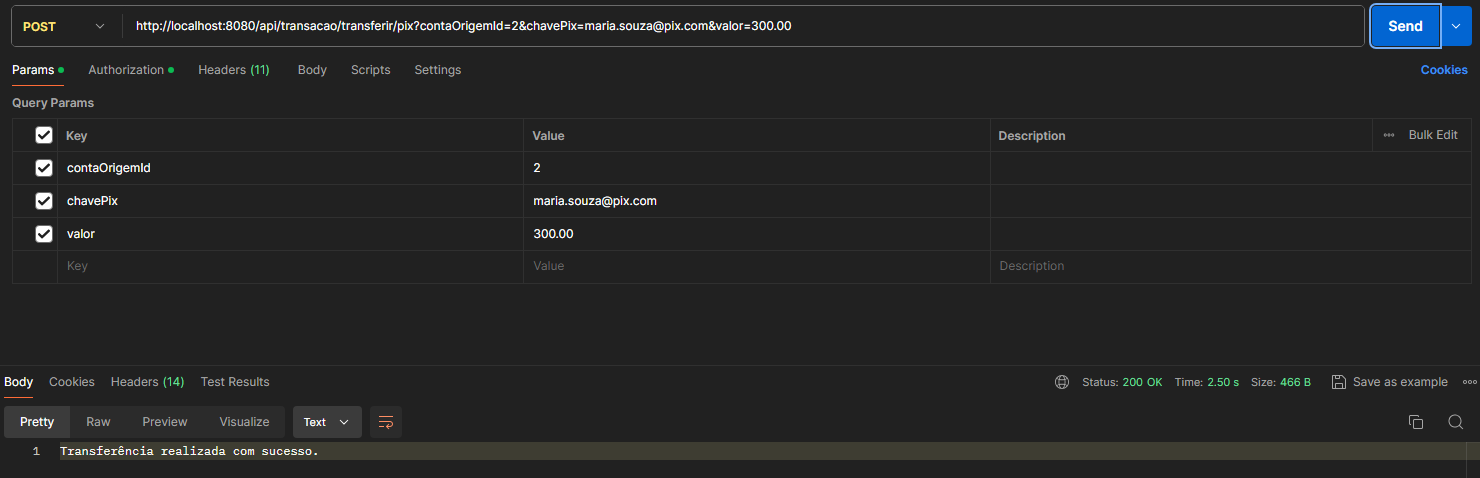
**6.15 Realizar Pix.**

Para realizar transferência, será feita uma requisição do tipo POST para a url/api/transacao/transferir/pix. (Utiliza Params)

Respostas esperadas:

HTTP200: Sucesso (Retorna um comprovante)

HTTP403: Forbidden, caso um usuário tente acessar sem estar logado.



# **7. Tecnologias Utilizadas**

* Java 17
* Spring Boot
* Spring Security (JWT)
* MySQL
* H2
* Swagger
* Postman
* Jacoco

# **8. Jacoco**

