PROJETO FULL-STACK: SISTEMA DE GESTÃO DE PRODUTOS E CATEGORIAS (CRUD)

1. INTRODUÇÃO

Este documento serve como Guia de Execução e Documentação para o projeto Full-Stack de Sistema de Gestão de Produtos e Categorias. O objetivo principal é implementar as funcionalidades **CRUD** (*Create, Read, Update, Delete*) completas para as entidades **Produto** e **Categoria**.

A comunicação entre o Front-End e o Back-End é realizada **exclusivamente** através da **Fetch API nativa do JavaScript**, utilizando os métodos HTTP (GET, POST, PUT e DELETE).

2 . IDENTIFICAÇÃO

Nome Completo:	**Matrícula:**
Henrique Linhares Pinheiro Loiola	25173566

3. ESTRUTURA DO REPOSITÓRIO

O repositório *app_frontProdutos* segue a estrutura de diretórios obrigatória, conforme detalhado a seguir:

app_frontProdutos/

├── BackEnd/crud-produto (Código completo da API REST)

├── FrontEnd/crud-produtos (Código completo do React)

└── README.md (guia de execução e documentação)

4.INSTRUÇÕES DE EXECUÇÃO

Para garantir a execução bem-sucedida da aplicação, o Back-End deve ser inicializado antes do Front-End.

4.1 Execução do Back-End (Spark Java / JDBC)

O Back-End é uma API REST desenvolvida em Spark Java que utiliza JDBC para persistência de dados em um banco MySQL na porta padrão 4567.

4.1.1 Pré-Requisitos

- **Java Development Kit (JDK):** Versão 11 ou superior.
- **MySQL Server:** Servidor de banco de dados ativo.
- **Ambiente de Desenvolvimento Java:** IDE (IntelliJ, Eclipse) ou VS Code com as devidas extensões.

4.1.2 Configuração do Banco de Dados

Criação do Banco e Estrutura: Execute o *script* SQL a seguir em seu cliente MySQL para criar a base de dados aulajdbc e as tabelas categorias e produtos.

Script de Criação:

```
SET NAMES 'utf8mb4';

CREATE DATABASE IF NOT EXISTS aulajdbc;

USE aulajdbc;


CREATE TABLE IF NOT EXISTS categorias (

   id BIGINT PRIMARY PRIMARY KEY AUTO_INCREMENT,

   nome VARCHAR(255) NOT NULL

);


INSERT INTO categorias (nome) VALUES

('Eletrônicos'),

('Livros'),

('Alimentos');



CREATE TABLE IF NOT EXISTS produtos (

   id BIGINT PRIMARY KEY AUTO_INCREMENT,
```

```
    nome VARCHAR(255) NOT NULL,

    preco DOUBLE NOT NULL,

    estoque INT NOT NULL,

    id_categoria BIGINT,

    FOREIGN KEY (id_categoria) REFERENCES categorias(id)

);
```

1. **Ajuste de Conexão:** É mandatório localizar o arquivo de configuração JDBC(package util a classe ConnectionFactory) e ajustar as credenciais (usuário e senha) para garantir a conexão com o seu servidor MySQL local.

4.1.3 Inicialização do Servidor API

1. Acessar o diretório *BackEnd/crud-produto* na IDE ou terminal.
2. Compilar e executar a classe principal do servidor (api.ApiProduto).
3. **Confirmação:** A API estará acessível em *http://localhost:4567/produtos.*

4.2 Execução do Front-End (ReactJS)

O Front-End é construído em **ReactJS** e provê a interface de usuário, rodando na porta padrão 3000(localhost:3000).

4.2.1 Pré-Requisitos

- **Node.js e npm:** Instalados e configurados na máquina.

4.2.2 Instalação e Inicialização

Abrir um terminal e navegar até o diretório do Front-End:

cd FrontEnd

cd crud-produtos

Instalar as dependências do projeto:

npm install

Iniciar o servidor de desenvolvimento da aplicação:

npm start

4.2.3 Acesso à Aplicação

O acesso à aplicação deve ser feito através do navegador na URL: **http://localhost:3000**.

O Back-End (porta 4567) deve estar ativo antes da inicialização do Front-End para que a comunicação de dados seja estabelecida corretamente.