Documento Descritivo do Projeto: Finance Control

1. Introdução

O projeto **Finance Control** é uma aplicação de *backend* desenvolvida para gerenciar finanças pessoais, permitindo o controle de despesas e receitas, bem como a associação de transações a categorias e contas bancárias específicas. A aplicação é construída como uma API RESTful, seguindo as melhores práticas de desenvolvimento com o *framework* Spring Boot.

2. Tecnologias e Arquitetura

O projeto adota uma arquitetura de microsserviços simples, com foco em uma API robusta para manipulação de dados financeiros.

2.1. Stack Tecnológico

Componente	Tecnologia	Função
Linguagem	Java	Linguagem principal de desenvolvimento.
Framework	Spring Boot 3.x	Facilita a criação de aplicações Java autônomas e de nível de produção.
Banco de Dados	PostgreSQL	SGBD relacional robusto e de código aberto para persistência de dados.
Persistência	Spring Data JPA / Hibernate	Gerenciamento de mapeamento objeto-relacional (ORM).
Containerização	Docker / Docker Compose	Simplifica o empacotamento e a execução da aplicação e do banco de dados.
Segurança	Spring Security / JWT	Autenticação e autorização de usuários via JSON Web Tokens.

2.2. Estrutura do Projeto

A estrutura do projeto segue o padrão de pacotes comum em aplicações Spring Boot, organizando o código em camadas:

- `controller`: Responsável por receber requisições HTTP e retornar respostas (endpoints).
- `service`: Contém a lógica de negócio da aplicação.
- `repository`: Interface de comunicação com o banco de dados.
- `model`: Classes que representam o modelo de dados (entidades JPA).
- `dto`: Objetos de Transferência de Dados, usados para comunicação entre camadas e com o cliente.
- `security`: Configurações e utilitários de segurança (JWT, autenticação).

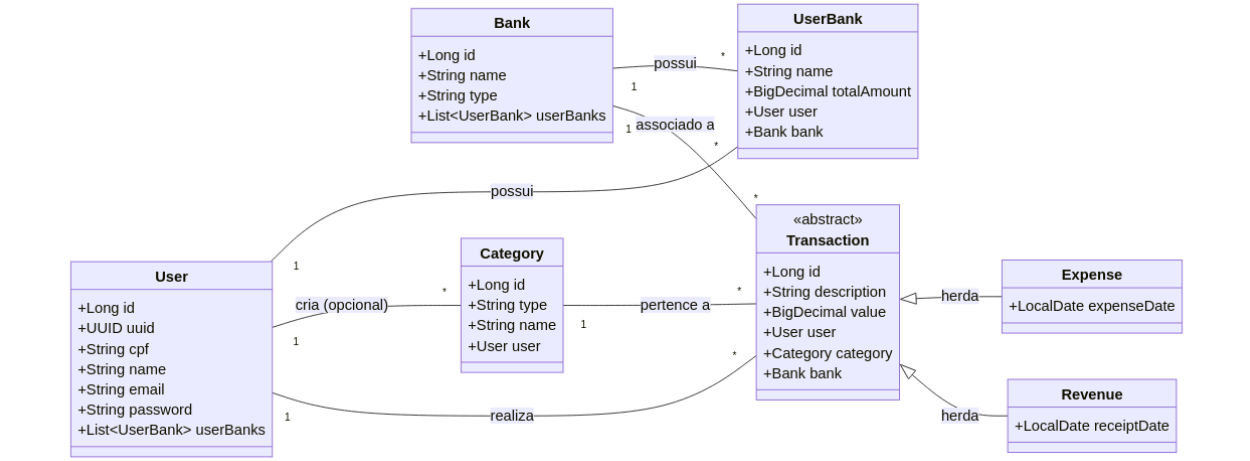
3. Modelo de Dados

O modelo de dados é o coração da aplicação, focado em gerenciar usuários, contas bancárias, categorias e, principalmente, transações (receitas e despesas).

3.1. Diagrama de Classes

O diagrama a seguir ilustra as principais entidades e seus relacionamentos:

Diagrama de Classes - Finance Control



3.2. Descrição das Entidades

Entidade	Descrição	Relacionamentos Chave
User	Representa o usuário do sistema. Contém dados de autenticação e identificação (CPF, nome, email, senha).	1:N com `UserBank`, 1:N com `Transaction`, 1:N com `Category`.
Bank	Representa uma instituição bancária (e.g., Banco do Brasil, Nubank).	1:N com `UserBank`, 1:N com `Transaction`.
UserBank	Entidade de relacionamento N:M entre `User` e `Bank`, representando a conta bancária específica de um usuário em uma instituição. Armazena o saldo (`totalAmount`).	N:1 com `User`, N:1 com `Bank`.
Category	Define categorias para classificar transações (e.g., "Alimentação", "Salário", "Investimento"). Pode ser global (sem `user` associado) ou específica do usuário.	N:1 com `User`, 1:N com `Transaction`.
Transaction	Classe abstrata que define os atributos comuns a todas as transações financeiras (`description`, `value`, `user`, `category`, `bank`).	N:1 com `User`, N:1 com `Category`, N:1 com `Bank`.
Expense	Herda de `Transaction`, representando uma despesa. Adiciona o campo `expenseDate`.	Herda de `Transaction`.
Revenue	Herda de `Transaction`, representando uma receita. Adiciona o campo `receiptDate`.	Herda de `Transaction`.

4. Configuração de Execução

O projeto é configurado para ser executado facilmente utilizando o Docker Compose, garantindo que o banco de dados PostgreSQL e a aplicação sejam inicializados corretamente e interligados.

O arquivo `docker-compose.yml` define dois serviços principais:

1. **`db`**: O serviço de banco de dados, utilizando a imagem oficial do `postgres:15`.
    - **Porta Exposta**: `5433:5432` (a porta 5433 do host mapeia para a 5432 do container).
    - **Credenciais**: `POSTGRES_USER=postgres`, `POSTGRES_PASSWORD=postgres`, `POSTGRES_DB=finance-control`.
2. **`app`**: O serviço da aplicação Spring Boot (que deve ser construído a partir do código-fonte).

A execução é iniciada com o comando `docker compose up`, conforme detalhado no `README.md` que será fornecido.