UNIVERSIDADE FEDERAL DA FRONTEIRA SUL – UFFS

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

JAN CARLES

TRABALHO INTEGRADOR

SISTEMA DE CONTROLE FINANCEIRO

CHÁPECO - SANTA CATARINA

2025

DESCRIÇÃO DA LÓGICA DO SISTEMA

Este relatório explica como o sistema foi desenvolvido, mostrando as ideias usadas na implementação, as escolhas feitas durante o processo e o que foi mais fácil ou difícil de resolver. O principal objetivo foi criar um sistema que ajudasse no controle financeiro da empresa, permitindo registrar produtos, vendas, despesas e visualizar relatórios e o histórico de movimentações de um jeito organizado e fácil de usar.

1. Como o sistema foi construído

O sistema foi dividido em duas partes principais, seguindo a arquitetura em camadas para garantir a separação de responsabilidades, escalabilidade e manutenibilidade:

Backend – Responsável pela Lógica de Negócio e Persistência de Dados

Frontend – Responsável pela Interface do Usuário e Interação com a API

Backend (Node.js + Express + Sequelize/Postgres)

O backend segue um padrão de arquitetura de múltiplas camadas (**Controller, Service, Repository e Model**), onde cada parte tem uma responsabilidade única:

1. **Controller:** Responsável por receber requisições HTTP (GET, POST, DELETE e UPDATE) e delegar imediatamente a ação para o Service correspondente.
2. **Service (Lógica de Negócio):** Onde as regras financeiras são aplicadas. O Service valida os dados, executa cálculos (ex: total da venda, lucro/prejuízo) e coordena a gravação de dados, garantindo a integridade transacional (especialmente nas operações de Venda).
3. **Repository:** É a camada de acesso a dados. Ela encapsula toda a lógica de comunicação com o PostgreSQL (via Sequelize), traduzindo as necessidades do Service em consultas de banco de dados (CRUD).
4. **Model:** Define a estrutura das tabelas no PostgreSQL e seus relacionamentos.

Segurança: A autenticação e a autorização são realizadas através de **Tokens (JWT)** (usando a biblioteca Passport), garantindo que apenas usuários autorizados possam acessar as rotas sensíveis (Cadastro de Produtos, Vendas, Despesas, Dashboard).

Frontend (React + Mui + Axios)

O frontend foi estruturado em componentes funcionais React, focados em proporcionar uma experiência de usuário simples e intuitiva:

1. **Componentes de Tela:** Cada módulo principal (Produtos, Vendas, Despesas) possui um componente dedicado, utilizando a biblioteca de componentes **MUI (Material UI)** para design e responsividade.
2. **Comunicação:** O envio e recebimento de dados com a API do backend são realizados através da biblioteca **Axios**, implementando lógica de tratamento de erros.
3. **Fluxo de Dados:** Os componentes capturam os dados do usuário, validam o preenchimento de campos e enviam a estrutura correta (ex:Itens de Venda) para os endpoints da API.

2. Facilidade e Dificuldades no Desenvolvimento

Durante o desenvolvimento, algumas facilidades e desafios se destacaram:

Facilidades:

- A divisão clara entre frontend e backend facilitou a organização do código.
- O uso de React e Express tornou o processo mais fluido.
- A arquitetura modular permitiu adicionar funcionalidades sem afetar outras partes do sistema.

Dificuldades:

- Estilizar a interface foi difícil

3. Conclusão

O desenvolvimento do sistema possibilitou compreender, na prática, como implementar uma aplicação completa utilizando tecnologias modernas como Node.js e React. A arquitetura adotada garantiu organização, segurança e capacidade de expansão, enquanto a lógica aplicada permitiu atender às necessidades do negócio de maneira eficiente.

Apesar dos desafios enfrentados, o resultado entrega um sistema funcional, seguro e preparado para auxiliar no controle financeiro da empresa, cumprindo os objetivos propostos para o projeto.