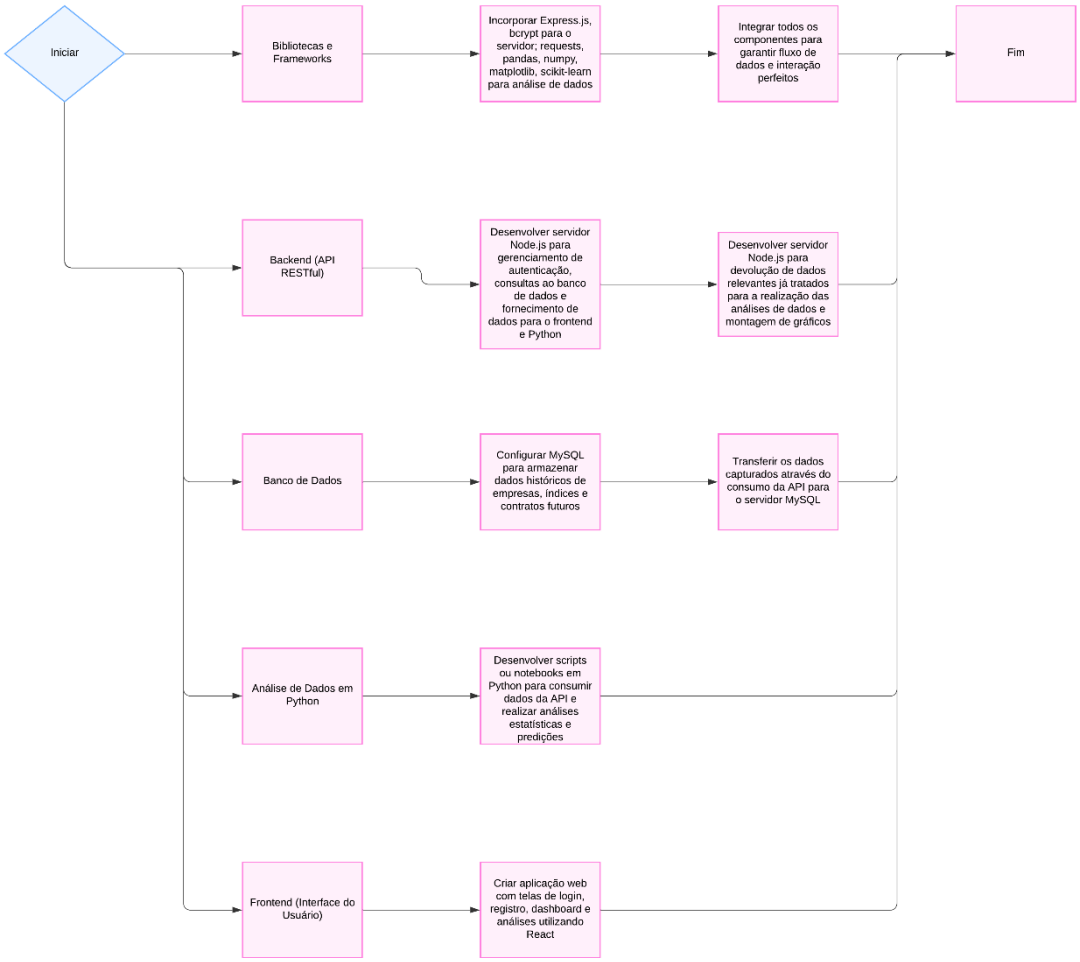
Engenharia de Software – Entrega 2

Diagrama UML do projeto:

Para desenhar o Diagrama UML do projeto, definimos as 5 grandes áreas de desenvolvimento, e esmiuçamos o que deve ser feito em cada uma dessas:



O documento original foi criado usando o LucidChart, e pode ser encontrado neste link.

Para isso, identificamos as seguintes áreas:

- Bibliotecas e FrameWorks;
- BackEnd da aplicação;
- FrontEnd da aplicação;
- Banco de Dados da aplicação;
- Análise dos dados.

Identificação e Aplicação de Padrões de Projeto no Desenvolvimento

Os padrões de projeto (Design Patterns) são soluções testadas e reutilizáveis para problemas recorrentes no design de software. No nosso projeto, podemos aplicar alguns desses padrões

para melhorar a estrutura, modularidade e manutenção do código, o que contribuirá para um desenvolvimento mais escalável e fácil de manter.

1. **Padrão DAO (Data Access Object) e Singleton**

   Para simplificar e organizar o acesso ao banco de dados, podemos aplicar os **padrões DAO** e **Singleton**. O **DAO** nos ajuda a encapsular o acesso ao banco, fornecendo uma interface consistente para o restante da aplicação. Isso reduz a dependência do código de negócios diretamente com o banco de dados, tornando mais fácil a troca de banco ou a modificação de consultas sem impactar outras partes do sistema. Já o **Singleton** garante que tenhamos uma única instância de conexão com o banco de dados, evitando a criação desnecessária de múltiplas conexões e melhorando o gerenciamento de recursos compartilhados.

2. **Padrões Observer**

   No frontend, o **Observer** pode ser utilizado para garantir que os gráficos e dados se atualizem automaticamente em resposta a mudanças nos dados. Esse padrão define uma relação um-para-muitos entre objetos, permitindo que todos os dependentes sejam notificados e atualizados quando um objeto central muda de estado. Isso facilita a implementação de uma interface reativa, onde o usuário vê as informações mais recentes de forma dinâmica.

3. **Padrões Facade**

   O **Facade** nos permite criar uma interface simplificada para interagir com subsistemas complexos, reduzindo o acoplamento e tornando o sistema mais fácil de usar e de modificar. Isso é especialmente útil quando precisamos que o código cliente interaja com várias partes do sistema de forma mais intuitiva.

Descrição da arquitetura do sistema

Nossa arquitetura pode ser descrita como uma aplicação web multicamadas, que integra diversas tecnologias para oferecer funcionalidades avançadas de análise de dados financeiros. A seguir, detalhamos cada camada e sua função dentro do sistema.

Arquitetura em Camadas

A aplicação é dividida em três camadas principais: Apresentação (Frontend), Negócios (Backend) e Persistência (Banco de Dados). Cada camada possui responsabilidades específicas que, juntas, garantem a organização e a eficiência da aplicação.

1. **Camada de Apresentação**

   Na **Camada de Apresentação**, utilizamos tecnologias como HTML, CSS e JavaScript, juntamente com frameworks como React, Vite e amCharts. Essa camada é responsável por interagir diretamente com o usuário por meio da interface gráfica. Aqui, capturamos as

entradas dos usuários, como login, registro e filtros de pesquisa, além de apresentar os dados de forma visual, utilizando gráficos e tabelas.

2. Camada de Negócios

A **Camada de Negócios** é gerenciada por um **backend** desenvolvido em Node.js. Nessa camada, são implementadas a lógica de negócios e as regras de validação. Ela lida com autenticação de usuários, gerenciamento de sessões e processa as requisições recebidas do **frontend**, retornando dados conforme necessário para que o usuário visualize informações atualizadas e seguras.

3. Camada de Persistência

Por fim, na **Camada de Persistência**, utilizamos um banco de dados MySQL para armazenar as informações de forma persistente, como dados de usuários e dados financeiros. Esta camada é responsável por prover um acesso rápido e seguro às informações, executando consultas otimizadas para manipulação e recuperação de dados.