Entrega 2 - ES e AS

Aplicação de design de software

Introdução

O projeto consiste em um dashboard corporativo desenvolvido para uma pequena startup do setor de cupons de desconto, a Pic Money.
O sistema tem como objetivo fornecer uma visão clara e centralizada dos dados empresariais, como receita, despesas, e permitindo que os CEOs e CFOs analisem informações relevantes sobre desempenho, parcerias e utilização de cupons.

Além disso, há usuários com papel de administradores (desenvolvedores), responsáveis pela manutenção do sistema e pelo gerenciamento de dados e acessos.

Arquitetura de Software (Alto e Baixo nível)

Design de Alto Nível

A arquitetura adotada segue o **padrão em camadas (Layered Architecture)**, garantindo separação de responsabilidades, manutenção facilitada e maior escalabilidade do sistema.

As principais camadas são:

- **Camada de Apresentação (Front-end)**
  Responsável pela interface do dashboard. É onde o CEO e o CFO visualizam gráficos, tabelas e relatórios.
  (Exemplo: React, HTML, CSS ou framework web equivalente).
- **Camada de Lógica de Negócio (Back-end)**
  Contém as regras que processam os dados de empresas, cupons e usuários.
  (Exemplo: Java, Node.js, ou outra linguagem utilizada para o processamento).
- **Camada de Dados (Banco de Dados)**
  Armazena todas as informações do sistema, incluindo cadastros de usuários, cupons, empresas e histórico de utilização.
  (Exemplo: MySQL, PostgreSQL, MongoDB, etc.)

Essa divisão facilita a manutenção, pois cada parte pode ser atualizada independentemente sem afetar as demais.

Design de Baixo Nível

O design detalhado é orientado a objetos, com classes que representam as principais entidades do sistema.
Os relacionamentos são baseados nas necessidades do negócio:

- **Usuário**: representa os CEOs e CFOs, com permissões de leitura e visualização de relatórios.
- **Administrador**: responsável pela manutenção do sistema, podendo gerenciar usuários e cupons.
- **Empresa Parceira**: representa as empresas que divulgam ou oferecem cupons.
- **Cupom**: vinculado a uma empresa, contém dados como valor, validade e número de usos.

Essas classes interagem de forma que o administrador gerencia as empresas e cupons, enquanto os usuários consultam dados e métricas geradas pelo sistema.

Padrões e Princípios Aplicados

Durante o design do software, foram aplicados princípios de boas práticas:

- **SOLID** — para garantir coesão e baixo acoplamento entre classes.
- **DRY (Don’t Repeat Yourself)** — para evitar repetição de código e lógica duplicada.
- **MVC (Model-View-Controller)** — usado implicitamente para separar visualização, controle e modelo de dados no dashboard.

Esses princípios tornam o sistema mais sustentável, facilitando futuras atualizações e a adição de novas funcionalidades, como relatórios customizados ou novos tipos de usuários

Resultados da Aplicação do Design

A aplicação do design de software resultou em um sistema modular, escalável e de fácil manutenção, com uma divisão clara de responsabilidades entre as camadas e classes. Essa organização tornou o processo de desenvolvimento mais eficiente, permitindo que diferentes membros da equipe trabalhassem em partes distintas do projeto sem conflitos, além de facilitar futuras evoluções e melhorias no dashboard.

Diagramas UML

Introdução

A escolha do diagrama de casos de uso e do diagrama de classes se deve à sua importância fundamental nas etapas iniciais do desenvolvimento de software.
O diagrama de casos de uso foi utilizado por permitir uma visão clara e objetiva das funcionalidades do sistema e das interações entre os usuários e o dashboard, facilitando a compreensão dos requisitos funcionais por todos os membros da equipe.

Já o diagrama de classes foi escolhido por representar de forma estruturada a modelagem interna do sistema, detalhando as entidades, atributos, métodos e relacionamentos necessários para a implementação.
Esses dois diagramas foram priorizados em relação a outros tipos de UML por fornecerem uma base sólida tanto conceitual quanto técnica, servindo como ponto de ligação entre a análise de requisitos e o design orientado a objetos do projeto.

Diagramas

Diagrama de Casos de Uso

O **Diagrama de Casos de Uso** tem como objetivo representar as **interações entre os usuários (atores)** e o sistema.
No contexto do projeto, o sistema **Pic Money Dashboard** é acessado por dois perfis principais:

- **Administrador/Dono**: responsável por realizar login, visualizar e filtrar dados, gerar relatórios, comparar desempenhos e atualizar informações do sistema.

- **Analista de Dados**: possui acesso para gerar relatórios detalhados e exportá-los, além de analisar informações estratégicas.

Há ainda a integração com o **Sistema de Dados da Picmoney**, que é responsável pelo envio de informações atualizadas para o dashboard.

Esse diagrama permite entender as **principais funcionalidades disponíveis** para cada tipo de usuário e a **comunicação entre o sistema e fontes externas de dados**, garantindo clareza no planejamento das operações.

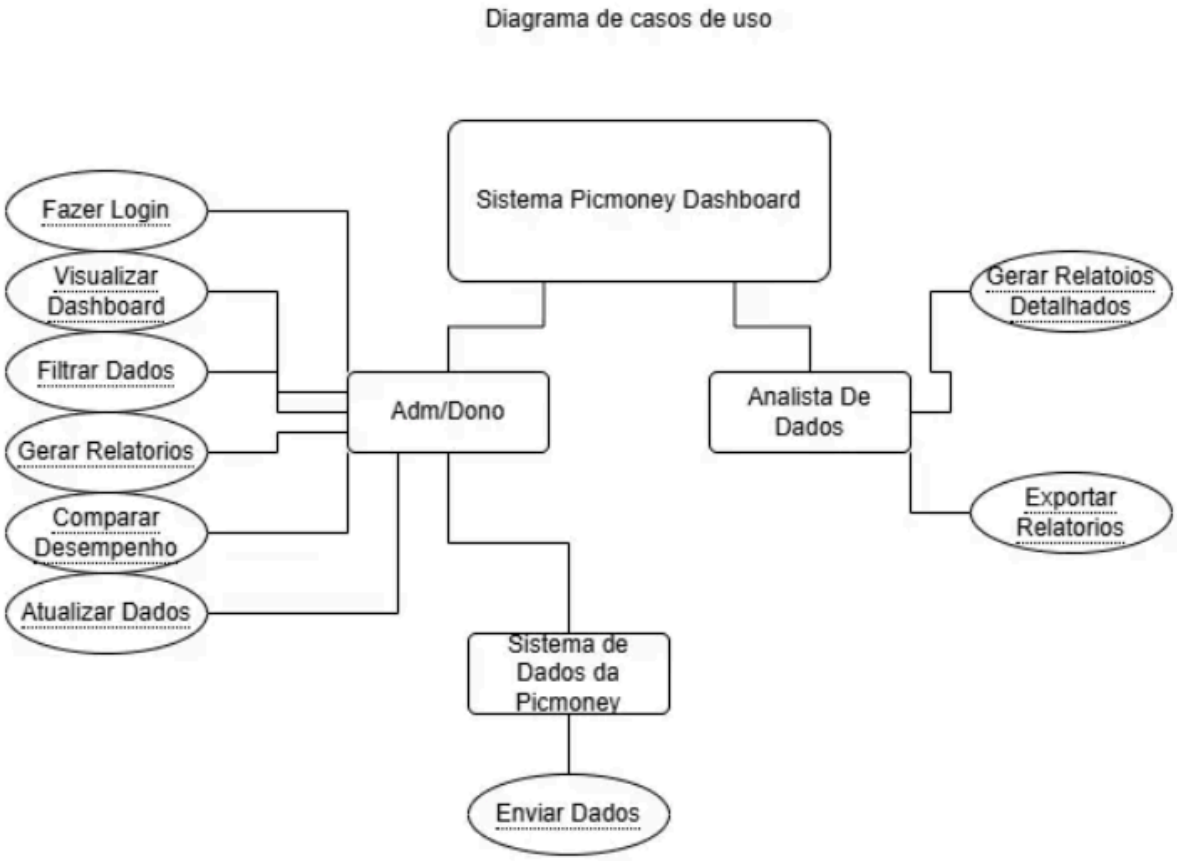


Diagrama de Classes

O **Diagrama de Classes** representa a **estrutura interna do sistema**, destacando as principais entidades, seus atributos, métodos e os relacionamentos entre elas.

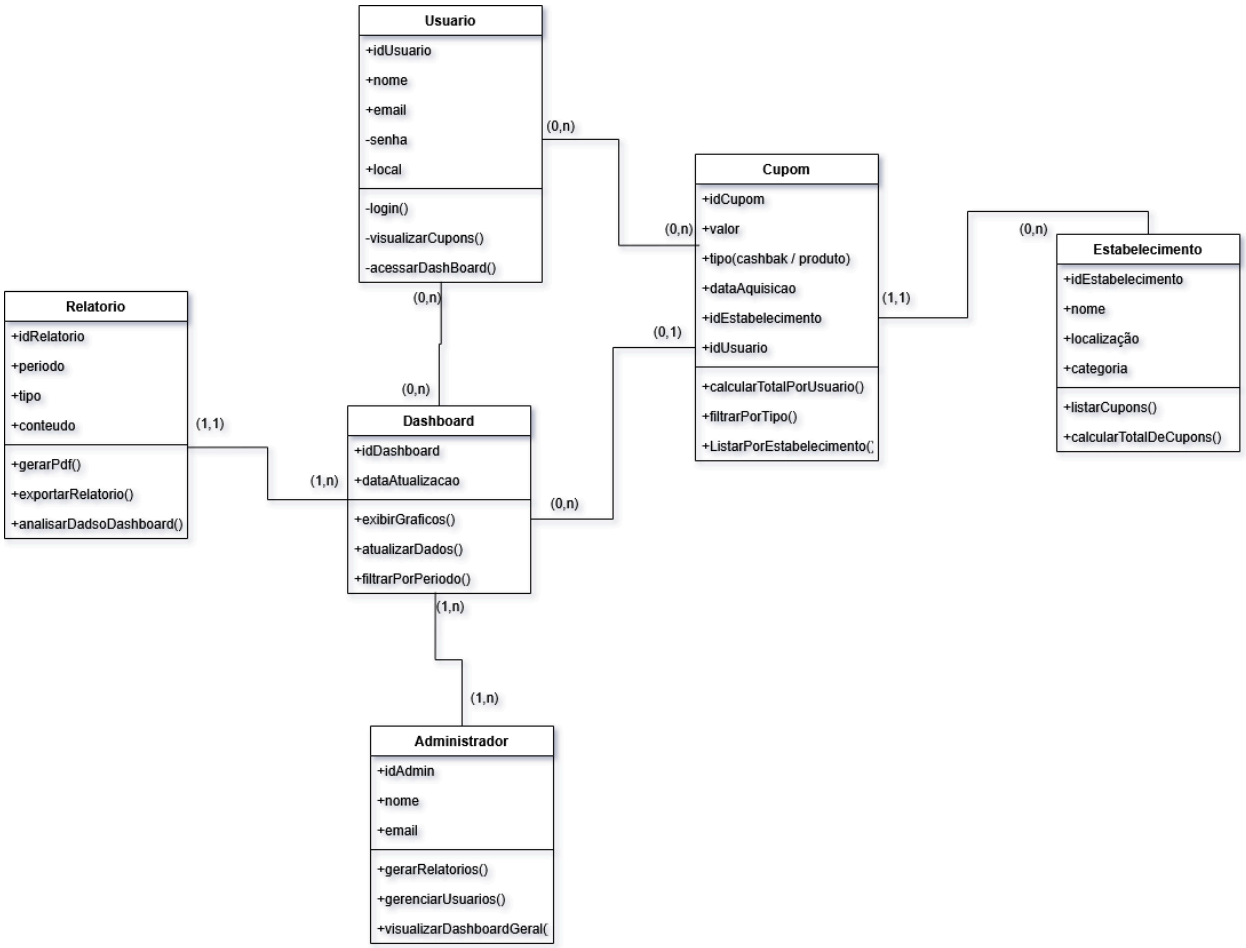
As classes modeladas foram:

- **Usuário**: representa os CEOs e CFOs que utilizam o sistema para visualizar relatórios e acompanhar métricas.
- **Administrador**: possui permissões de gerenciamento, podendo gerar relatórios gerais e controlar o acesso dos usuários.
- **Dashboard**: centraliza os dados e funções principais, como exibição de gráficos, atualização de dados e filtragem por período.
- **Relatório**: contém informações sobre período, tipo e conteúdo, com métodos para exportação e análise.

- **Cupom**: armazena dados sobre valor, tipo e relação com usuários e estabelecimentos.

- **Estabelecimento**: representa as empresas parceiras, com informações sobre nome, localização e categoria.

Os relacionamentos indicam a **interconexão entre as entidades**, como o vínculo entre *usuários e dashboards*, *relatórios e dashboards*, e *cupons e estabelecimentos*. Essa modelagem garante uma visão clara da lógica de dados e das dependências internas do sistema.



Resultados

A utilização dos diagramas UML permitiu uma melhor compreensão do comportamento e da estrutura do sistema do Dashboard.
Enquanto o diagrama de casos de uso descreve as interações externas e os fluxos de funcionalidades, o diagrama de classes detalha a organização interna e os relacionamentos entre os componentes do software. Juntos, esses elementos reforçam a aplicação prática do design de software, servindo como base sólida para o desenvolvimento e evolução do projeto.