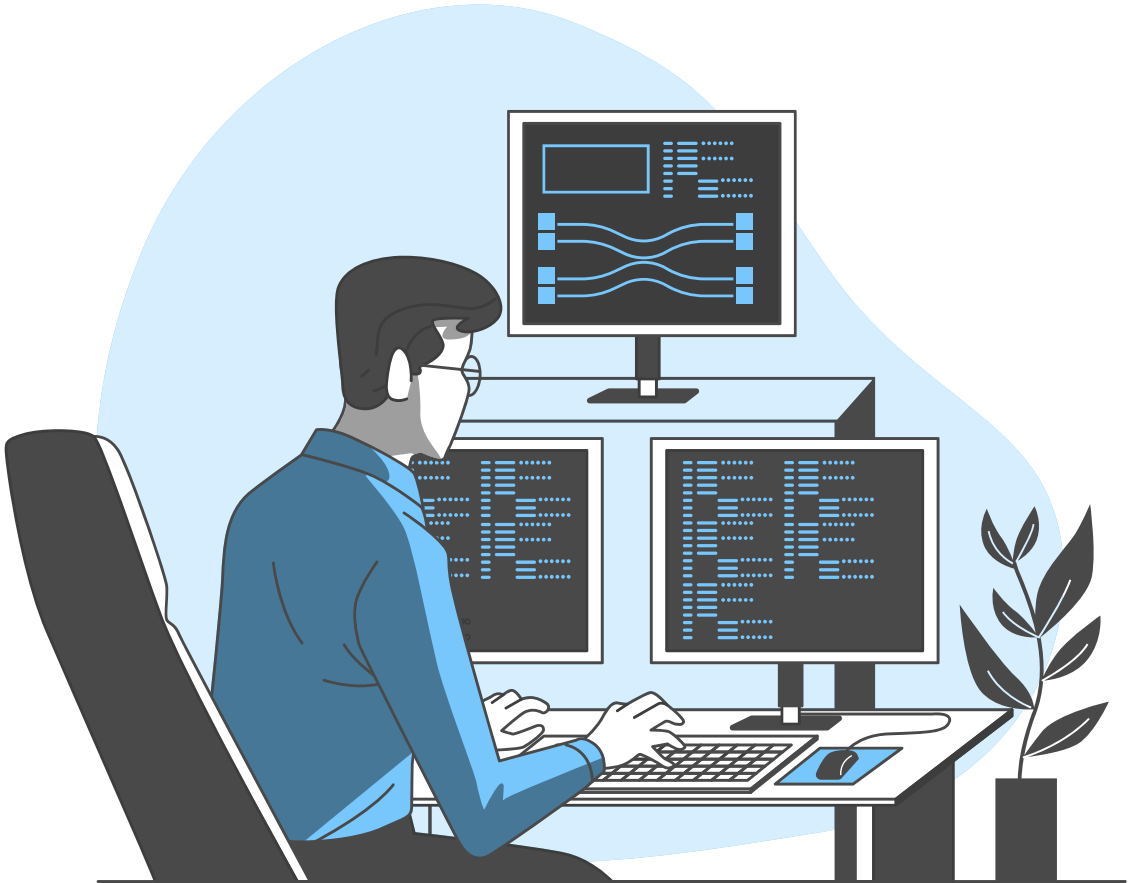


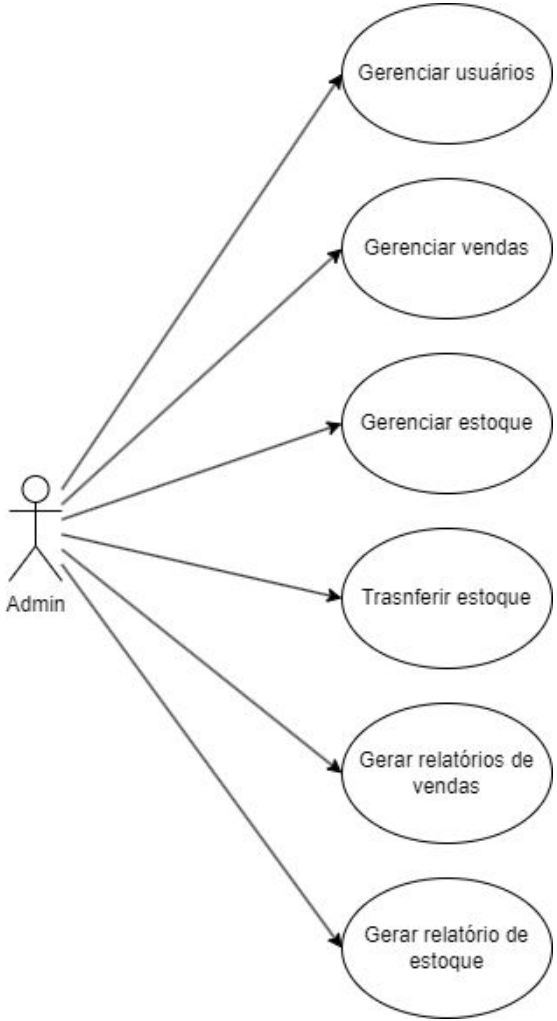
Planeta Jeans

Pedro Talma Toledo

Descrição do projeto

- Loja de moda especializada em jeans com quatro franquias
- Enfrentava dificuldades na gestão de estoque e vendas
- Utilizava métodos antiquados como anotações em papel, causando falta de praticidade e organização.

Casos de uso


Gerenciar usuários
Gerenciar vendas
Gerenciar estoque
Trasnferir estoque
Gerar relatórios de vendas
Gerar relatório de estoque
Admin

Gerenciar Vendas – Últimos 7 Dias

```
async getSalesSevenDays(req, res) {
  const sevenDaysAgo = new Date();
  sevenDaysAgo.setDate(sevenDaysAgo.getDate() - 7);

  const formattedDate = sevenDaysAgo.toISOString().split("T")[0];

  try {
    // Obter as vendas dos últimos 7 dias e agrupar pelo store_id
    const totalSalesByStore = await connection("Sale")
    .join("Store", "Sale.store_id", "Store.id")
    .where("sale_date", ">=", formattedDate)
    .groupBy("Sale.store_id", "Store.nickname")
    .select("Sale.store_id", "Store.nickname")
    .sum("value as total");

    // Enviar a resposta como JSON, incluindo o store_id e o valor total somado por store_id
    res.json(totalSalesByStore);
  } catch (error) {
    console.error("Error fetching sales report:", error);
    res.status(500).send("Internal Server Error");
  }
},
```

Transferência de estoque

```
async transfer(req, res, next) {
  const { sourceStoreId, destinationStoreId, productId, quantity } = req.body;

  try {
    await connection.transaction(async (trx) => {
      // Verifica o estoque da loja de origem
      const sourceStock = await trx('Stock')
        .where({ store_id: sourceStoreId, id: productId })
        .select('quantity', 'type', 'brand', 'price', 'size', 'description')
        .first();

      if (!sourceStock) {
        return res.status(404).json({ error: 'Produto não encontrado na loja de origem!' });
      }

      if (sourceStock.quantity < quantity) {
        return res.status(400).json({ error: 'Quantidade insuficiente na loja de origem!' });
      }

      // Deduz a quantidade do estoque da loja de origem
      const updatedSourceQuantity = sourceStock.quantity - quantity;
      await trx('Stock')
        .where({ store_id: sourceStoreId, id: productId })
        .update({
          quantity: updatedSourceQuantity
        });

      // Verifica se o produto já existe no estoque da loja de destino
      const destinationStock = await trx('Stock')
        .where({
          store_id: destinationStoreId,
          type: sourceStock.type,
          brand: sourceStock.brand,
          size: sourceStock.size
        })
        .select('quantity', 'id')
        .first();

      if (destinationStock) {
        // Se o produto já existe, atualiza a quantidade existente
        const updatedDestinationQuantity = parseInt(destinationStock.quantity) + parseInt(quantity);
        await trx('Stock')
```

Diagrama de Pacotes

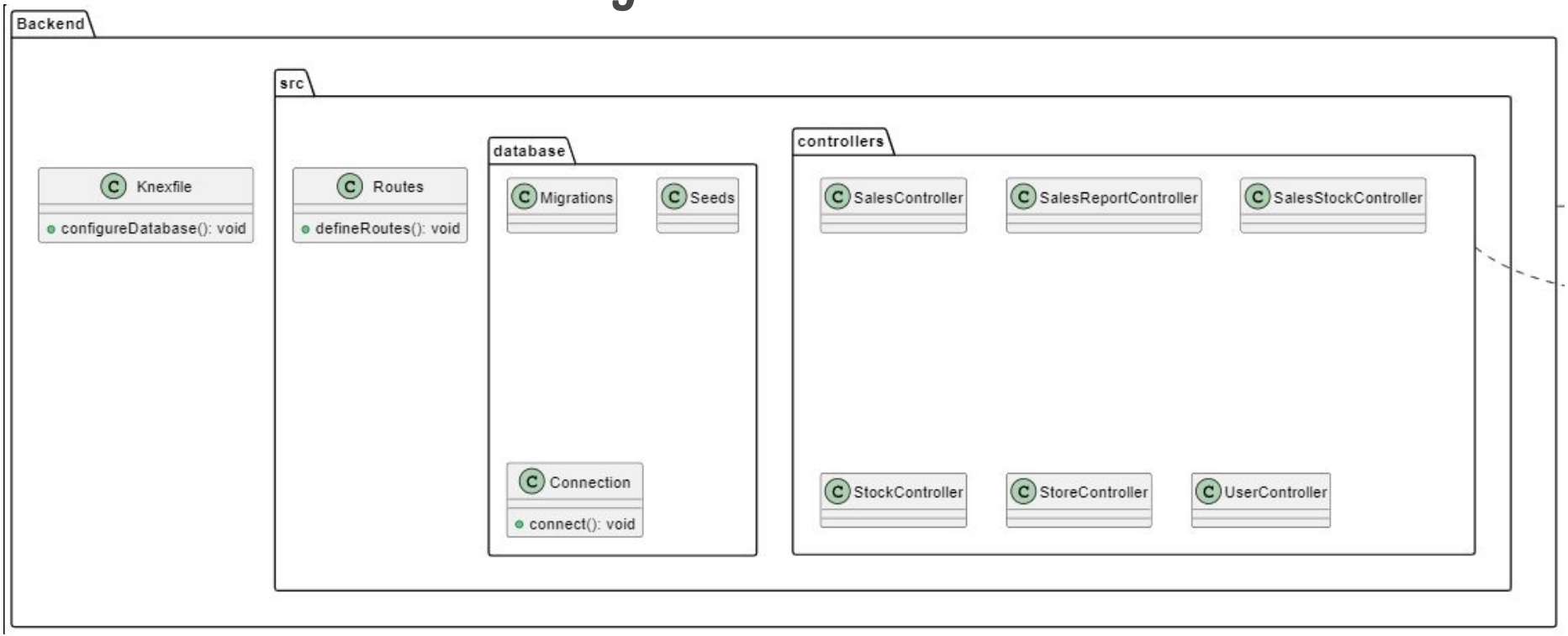


Diagrama de Pacotes

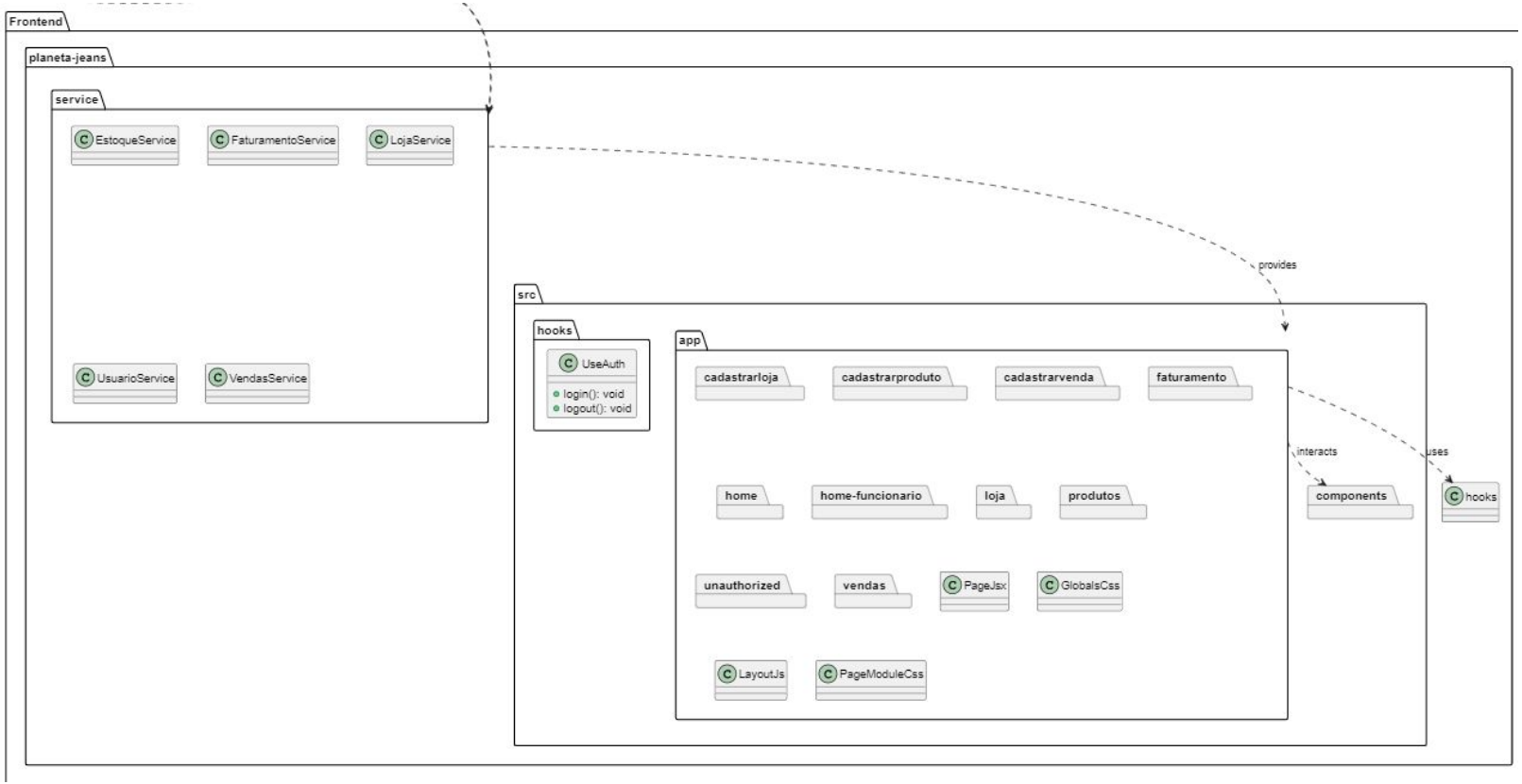


Diagrama de Pacotes – Visão Geral

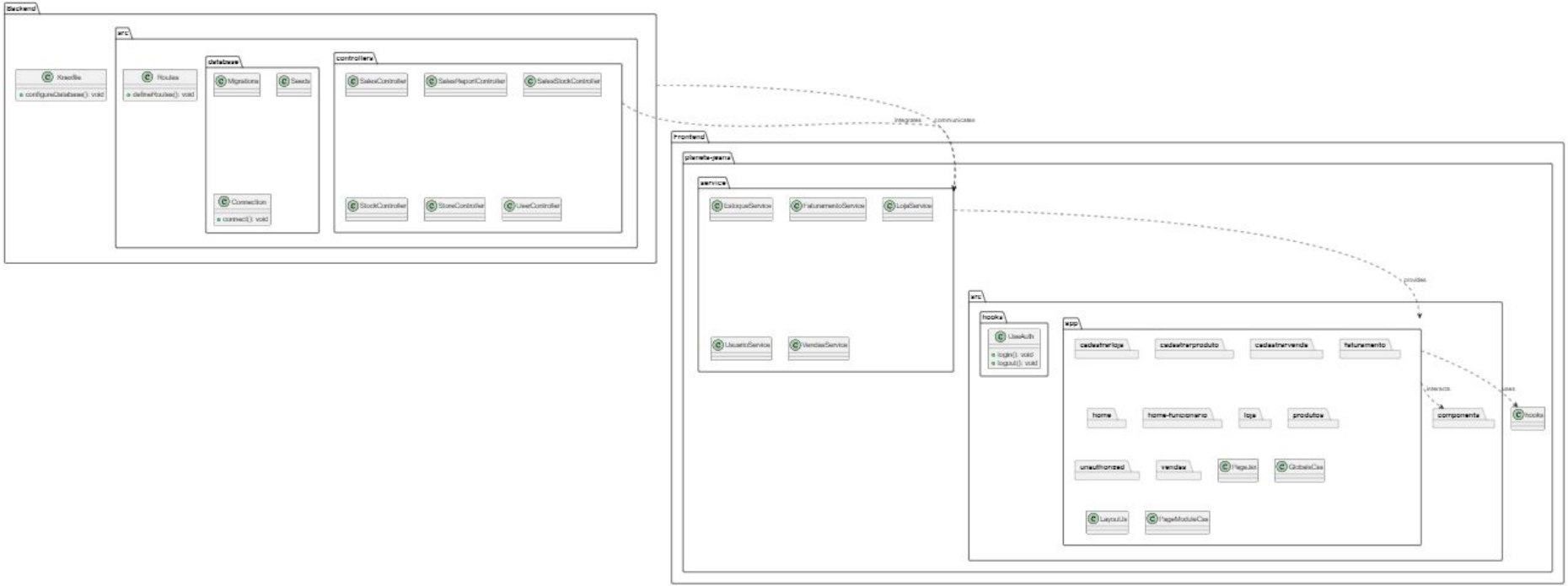
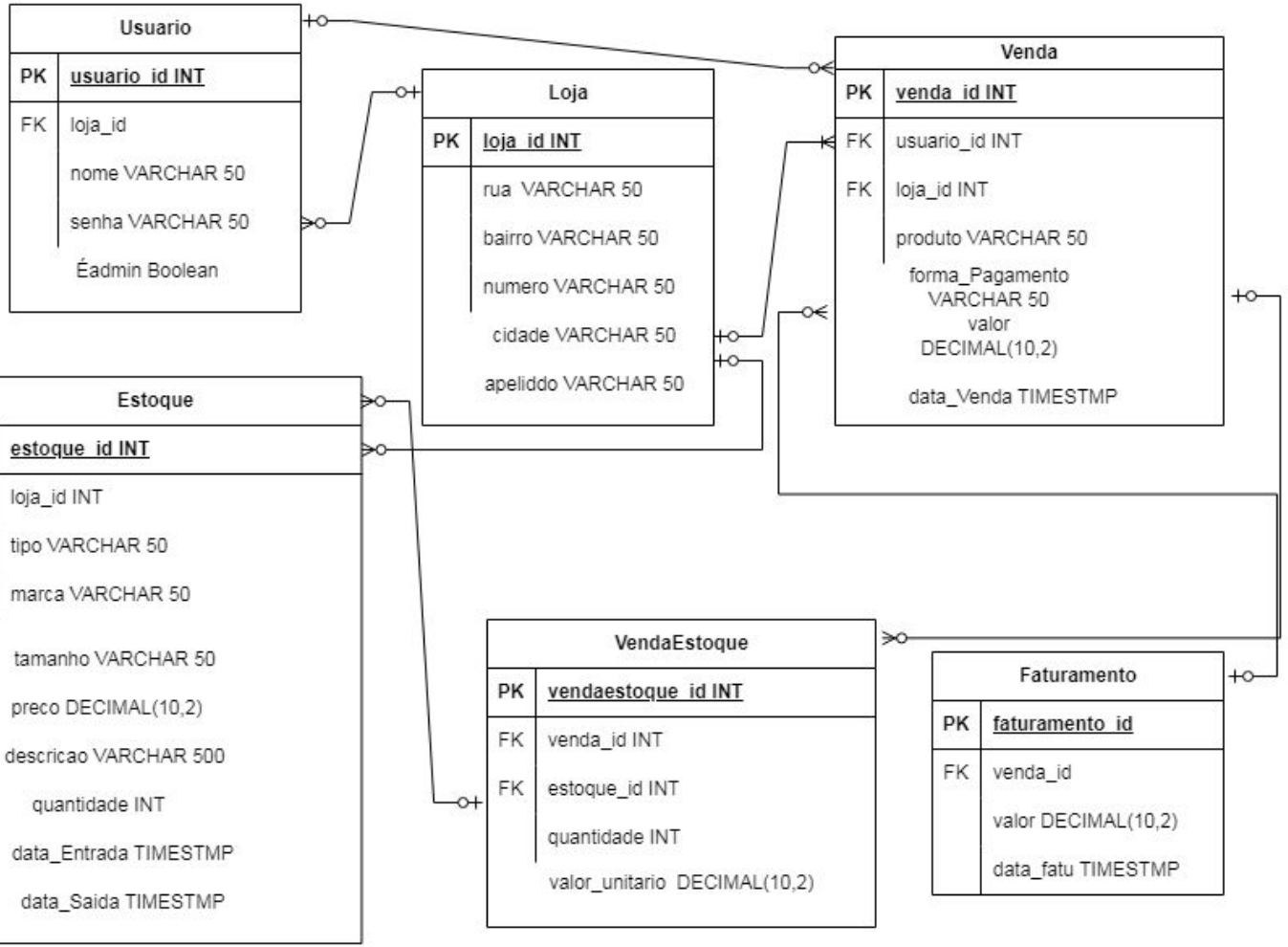
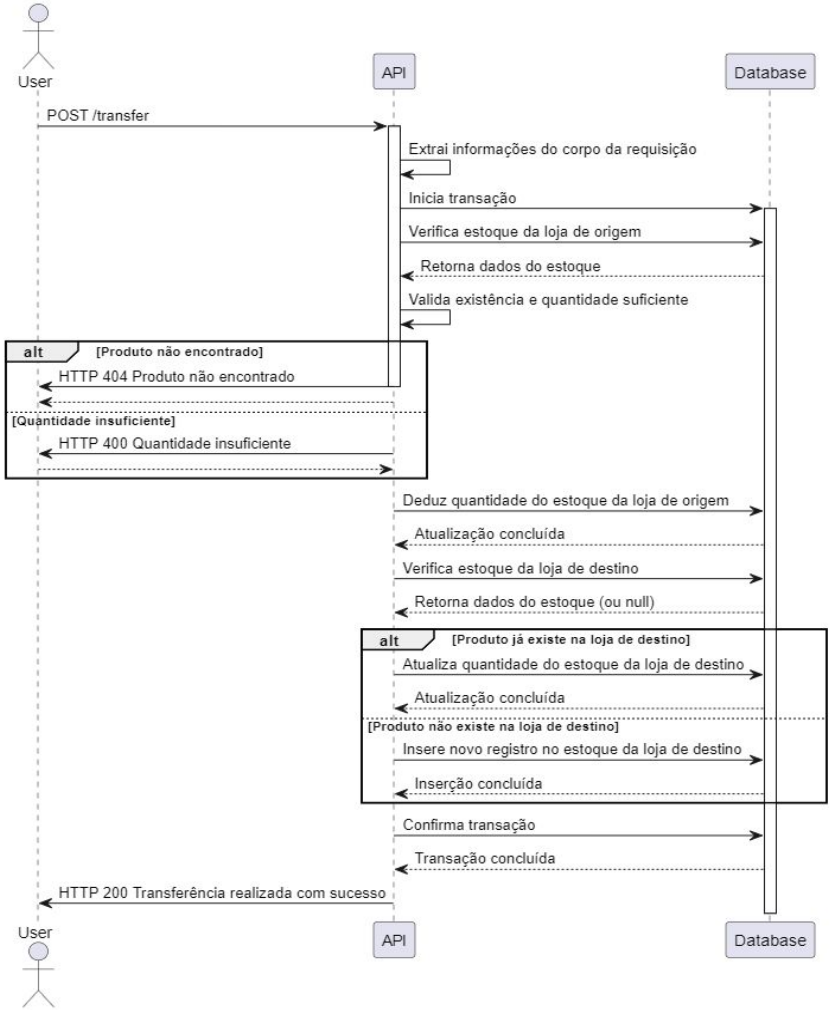


Diagrama de Classes

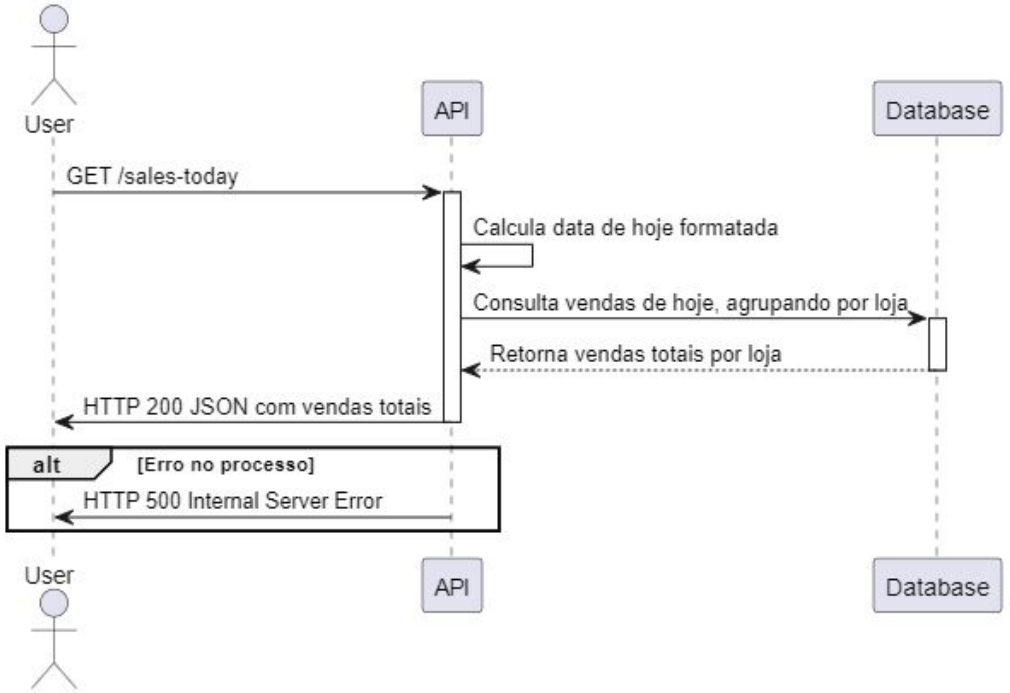

Usuario
PK usuario_id INT
FK loja_id
nome VARCHAR 50
senha VARCHAR 50
Éadmin Boolean
Loja
PK loja_id INT
rua VARCHAR 50
bairro VARCHAR 50
numero VARCHAR 50
cidade VARCHAR 50
apeliddo VARCHAR 50
Venda
PK venda_id INT
FK usuario_id INT
FK loja_id INT
produto VARCHAR 50
forma_Pagamento VARCHAR 50
valor DECIMAL(10,2)
data_Venda TIMESTMP
Estoque
PK estoque_id INT
FK loja_id INT
tipo VARCHAR 50
marca VARCHAR 50
tamanho VARCHAR 50
preco DECIMAL(10,2)
descricao VARCHAR 500
quantidade INT
data_Entrada TIMESTMP
data_Saida TIMESTMP
VendaEstoque
PK vendaestoque_id INT
FK venda_id INT
FK estoque_id INT
quantidade INT
valor_unitario DECIMAL(10,2)
Faturamento
PK faturamento_id
FK venda_id
valor DECIMAL(10,2)
data_fatu TIMESTMP

Diagramas de Sequência

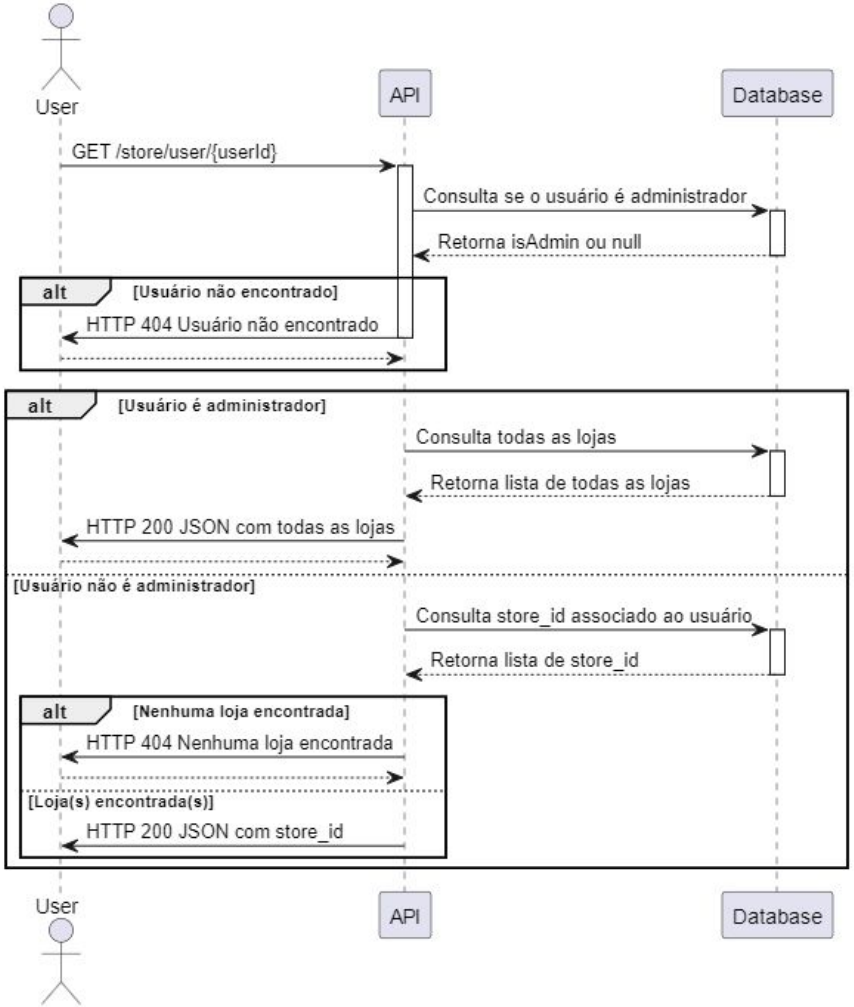
Transferência de produtos entre lojas



Resumo de vendas do dia



Mostrar loja por tipo de usuário



Análise Crítica

A análise crítica será feita a partir do `StockController` do projeto Planeta Jeans que está no link: Cgd-Trasnfer-Stock
E a partir deste trecho de código, será possível generalizar os problemas e soluções para o resto da aplicação, já que a solução segue este padrão de código.

Análise Crítica

Problemas e Melhorias

1. Organização e Estrutura do Código

- **Problema:** As funções estão em um único módulo, o que pode tornar o arquivo difícil de manter considerando a escalabilidade do sistema
- **Solução:** Dividir responsabilidades, organizando o código em diferentes arquivos ou camadas, como `services`, `repositories`, e `controllers`

2. Manipulação de Erros

- **Problema:** A manipulação de erros é inconsistente e, em alguns casos, insuficiente. Por exemplo, no método `delete`, o código pode gerar um erro se `stock` for `null` já que tenta acessar `stock.id`
- **Solução:** Adicionar verificações mais robustas e centralizar o tratamento de erros com middlewares para evitar redundâncias

Análise Crítica

Problemas e Melhorias

3. Validação de Dados

- **Problema:** Não há validação dos dados de entrada. Isso pode levar a inconsistências no banco de dados ou erros inesperados para os usuários
- **Solução:** Usar uma biblioteca de validação como `Joi` ou `Yup` para garantir que os dados fornecidos no `req.body` e `req.params` estejam corretos

4. Mensagens de Erro e Respostas

- **Problema:** Mensagens de erro nem sempre são claras ou consistentes. Por exemplo, o método `delete` usa `400 Bad Request` para um caso que poderia ser melhor tratado como `404 Not Found`.
- **Solução:** Padronizar mensagens e códigos de status para refletir na causa dos erros e facilitar o desenvolvimento

Github Planeta Jeans

Repositório