

TCP - Etapa 2

Gerenciador para a lojinha do DACOMP

Daniel Rocha
Diogo Rivoire
Patrick Alves

16 de dezembro de 2024

Conteúdo

1	Etapa 2 - Gerenciador de Lojinha DACOMP	2
1.1	Especificações	2
1.2	Mudanças em Relação à Etapa Anterior	2
1.3	Implementação	2
1.4	Teste	4
1.5	Executável (aplicação, interface)	4

1 Etapa 2 - Gerenciador de Lojinha DACOMP

1.1 Especificações

Um sistema que permite o gerenciamento produtos em estoque, registrar vendas e acompanhar o histórico de transações. O software é focado no ponto de vista e necessidades do logista.

1.2 Mudanças em Relação à Etapa Anterior

Nesta etapa, destacamos as mudanças realizadas em comparação com a Etapa 1:

- **Adaptação do escopo:** Mantivemos o foco do sistema com o intuito de priorizar as funcionalidades de gerenciamento de estoque e histórico de vendas, visando entregar uma solução mais específica.
- **Melhor detalhamento dos requisitos:** Foram mais limitadas as especificações dos requisitos para torná-las mais claras e alinhadas ao escopo do projeto.

1.3 Implementação

O objetivo deste trabalho foi implementar um sistema para gerenciamento de uma cafeteria. O software foi desenvolvido utilizando a linguagem de programação Java, seguindo os princípios da Programação Orientada a Objetos (POO). O sistema simula a operação de uma cafeteria, permitindo o controle de estoque de produtos, vendas e a aplicação de descontos em determinados produtos. Além disso, o sistema gera relatórios de vendas, destacando os produtos mais vendidos e os dias de maior movimento. O sistema foi implementado com a criação de várias classes, cada uma com responsabilidades específicas. A seguir, são apresentadas as principais classes do sistema e suas funcionalidades.

- **Classe Cafeteria**

A classe `Cafeteria` gerencia a quantidade de café disponível no estoque da cafeteria. Ela possui um atributo `quantidadeCafeDisponivel`, que armazena o número de unidades de café disponíveis.

- **Construtor:** Inicializa a quantidade de café disponível.
- **Métodos:**
 - * `verificarDisponibilidadeCafe(int quantidade)`: Verifica se há café suficiente no estoque.
 - * `diminuirCafe(int quantidade)`: Diminui a quantidade de café no estoque, caso haja café suficiente.
 - * `getQuantidadeCafeDisponivel()` e `setQuantidadeCafeDisponivel(int quantidade)`: Métodos de acesso (getter e setter).

- **Classe DescontoCaneca**

A classe `DescontoCaneca` permite aplicar um desconto no preço de venda de um produto. O desconto é informado em porcentagem e é aplicado ao preço original do produto.

- **Construtor:** Inicializa o valor do desconto.
- **Métodos:**
 - * `calcularDesconto(double precoOriginal)`: Calcula o preço com desconto.

- **Classe Estoque**

A classe `Estoque` gerencia o estoque de produtos da cafeteria. Ela permite adicionar, remover e consultar produtos, bem como atualizar a quantidade em estoque de um produto específico.

- **Construtor:** Inicializa a lista de produtos no estoque.
- **Métodos:**

- * `adicionarProduto(Produto produto)`: Adiciona um produto ao estoque.
- * `removerProduto(String nomeProduto)`: Remove um produto do estoque.
- * `consultarProduto(String nomeProduto)`: Consulta as informações de um produto específico.
- * `atualizarEstoqueProduto(String nomeProduto, int quantidade)`: Atualiza a quantidade de um produto no estoque.
- * `listarProdutos()`: Lista todos os produtos no estoque.

- **Classe RelatorioVendas**

A classe `RelatorioVendas` gera relatórios de vendas, destacando o total de vendas, o produto mais vendido e os dias de maior movimento.

- **Construtor:** Inicializa a lista de vendas e o período de análise do relatório.
- **Métodos:**
 - * `gerarRelatorio()`: Gera o relatório com base nas vendas realizadas no período.
 - * `calcularTotalVendas()`: Calcula o total de vendas no período.
 - * `calcularProdutoMaisVendido()`: Identifica o produto mais vendido no período.
 - * `calcularDiasMaisMovimento()`: Identifica os dias com maior número de vendas.
 - * `toString()`: Exibe o relatório gerado.

- **Classe Produto**

A classe `Produto` representa os produtos vendidos na cafeteria. Ela contém informações sobre o nome do produto, preço de custo, preço de venda e quantidade em estoque.

- **Construtor:** Inicializa as informações do produto.
- **Métodos:**
 - * `atualizarPreco(double novoPreco)`: Atualiza o preço de venda do produto.
 - * `atualizarEstoque(int quantidade)`: Atualiza a quantidade do produto no estoque.
 - * `toString()`: Exibe as informações do produto.

- **Classe ItemVenda**

A classe `ItemVenda` representa os itens vendidos em uma venda específica. Cada item contém o produto e a quantidade vendida.

- **Construtor:** Inicializa o produto e a quantidade vendida.
- **Métodos:**
 - * `calcularTotal()`: Calcula o valor total do item (preço de venda * quantidade).

- **Classe Venda**

A classe `Venda` representa uma venda realizada na cafeteria. Ela contém informações sobre a data da venda e os itens vendidos.

- **Construtor:** Inicializa a venda com um identificador único e a data da venda.
- **Métodos:**
 - * `adicionarItem(ItemVenda item)`: Adiciona um item à venda.
 - * `calcularTotalVenda()`: Calcula o valor total da venda.

- **Classe ComboPromocional**

A classe `ComboPromocional` permite criar combos de produtos com descontos.

- **Construtor:** Inicializa a lista de produtos no combo e o desconto aplicado.
- **Métodos:**
 - * `calcularPrecoCombo()`: Calcula o preço total do combo com o desconto aplicado.

1.4 Teste

Nesta fase, foi realizado o desenvolvimento de testes unitários utilizando o JUnit para verificar a funcionalidade e o comportamento das classes do domínio do sistema. Os testes foram implementados com o objetivo de garantir que as regras de negócios, como a adição de produtos, o processamento de vendas e a aplicação de descontos, funcionassem corretamente.

Os testes abrangem as seguintes classes principais do sistema:

- **Produto:** Testes para garantir que a criação, atualização de preço e atualização de estoque dos produtos funcionem conforme o esperado.
- **Estoque:** Testes para validar a adição, remoção e consulta de produtos no estoque.
- **Venda:** Testes para verificar se a venda é realizada corretamente, validando a atualização de estoque e a geração de itens de venda.
- **Cafeteria:** Testes para garantir que o controle de estoque de café seja feito corretamente.
- **DescontoCaneca e ComboPromocional:** Testes para assegurar que o cálculo de descontos seja executado corretamente, aplicando descontos nos produtos de forma adequada.

Os testes foram implementados no início do desenvolvimento, seguindo a metodologia de *Test-Driven Development (TDD)*, o que garantiu a qualidade e a confiabilidade do código. Durante o processo de desenvolvimento, os testes foram constantemente executados para identificar e corrigir falhas antes de avançar para a próxima etapa de implementação.

A experiência de utilizar testes unitários foi bastante positiva, pois possibilitou a detecção precoce de problemas, o que resultou em um código mais robusto e de fácil manutenção. A aplicação de TDD contribuiu para uma abordagem mais organizada e eficiente no desenvolvimento do sistema, garantindo que os requisitos fossem atendidos de forma mais segura.

1.5 Executável (aplicação, interface)


Nesta seção, apresentamos o executável da aplicação, destacando as funcionalidades desenvolvidas e a interface gráfica criada para interações com o usuário. A aplicação foi implementada utilizando JavaFX, com uma interface simples que permite o gerenciamento de produtos, vendas e relatórios.

- **Funcionalidades Desenvolvidas** A aplicação possui as seguintes funcionalidades:
 - **Cadastro de Produtos:** Permite adicionar produtos ao estoque, especificando nome, preço de custo, preço de venda e quantidade.
 - **Exibição de Produtos:** Exibe a lista de produtos cadastrados no estoque.
 - **Realização de Vendas:** Permite realizar vendas, verificando a quantidade em estoque e atualizando o estoque após a venda.
 - **Criação de Combos Promocionais:** Permite criar combos com desconto, combinando múltiplos produtos.
 - **Aplicação de Desconto em Canecas:** Permite aplicar um desconto específico no preço de venda de um produto (caneca).
 - **Geração de Relatórios de Vendas:** Permite gerar relatórios de vendas dentro de um período de tempo determinado.
 - **Verificação de Café no Estoque:** Permite verificar a quantidade de café disponível no estoque da cafeteria.
- **Manual de Uso Simplificado** Para executar a aplicação, siga os seguintes passos:
 1. **Iniciar a aplicação:** Execute o arquivo JAR da aplicação.
 2. **Cadastro de Produtos:** Preencha os campos com as informações do produto (nome, preço de custo, preço de venda, quantidade) e clique no botão "Adicionar Produto".
 3. **Exibição de Produtos:** Clique em "Exibir Produtos" para visualizar os produtos cadastrados.

```
Produtos no estoque:
Produto: banana
Preço de Custo: R$ 10.0
Preço de Venda: R$ 20.0
Quantidade em Estoque: 3
Produtos no estoque:
Produto: banana
Preço de Custo: R$ 10.0
Preço de Venda: R$ 20.0
Quantidade em Estoque: 2
```

Figura 1: Exibição do Estoque

4. **Realização de Vendas:** Digite o nome e a quantidade do produto a ser vendido e clique em "Realizar Venda".
 5. **Criação de Combo Promocional:** Insira os nomes dos produtos separados por vírgula e o desconto desejado, e clique em "Criar Combo Promocional".
 6. **Aplicação de Desconto em Caneca:** Insira o nome da caneca e o desconto desejado, e clique em "Aplicar Desconto na Caneca".
 7. **Geração de Relatórios:** Insira as datas de início e fim e clique em "Gerar Relatório" para visualizar o relatório de vendas.
 8. **Verificação de Café:** Clique em "Verificar Disponibilidade de Café" para visualizar a quantidade disponível no estoque.
- **Interface Gráfica** A interface foi desenvolvida para ser simples e intuitiva, permitindo ao usuário interagir facilmente com as funcionalidades do sistema. Cada funcionalidade está disponível por meio de campos de entrada, botões e alertas informativos. Abaixo estão algumas imagens da aplicação em execução, ilustrando as principais interações.
 - **Comparação com o Protótipo** A interface final está em conformidade com a prototipada na etapa anterior, embora com algumas simplificações para melhorar a usabilidade. A navegação entre as funcionalidades foi mantida simples, e as telas de cadastro e exibição de produtos possuem um layout claro e direto. Algumas pequenas mudanças foram feitas no design visual, mas as funcionalidades essenciais permanecem as mesmas, com o objetivo de facilitar a interação do usuário.

 Sistema de Vendas

Cadastro de Produtos

banana

10

20

3

Adicionar Produto

Exibir Produtos

Realizar Venda

banana

1

Realizar Venda

Café no estoque: Disponível (100 unidades)

Verificar Disponibilidade de Café

Criar Combo Promocional

Nomes dos produtos (separados por vírgula)

Desconto (%)

Criar Combo Promocional

Desconto para Canecas

Nome do Produto (Caneca)

Desconto (%)

Aplicar Desconto na Caneca

Relatório de Vendas

Data Início (yyyy-MM-dd)

Data Fim (yyyy-MM-dd)

Gerar Relatório

Figura 2: Sistema de Vendas