Sistema de Gerenciamento de Mercado Familiar

Documentação Técnica

Autora: Maria Eduarda Alexandre Aguiar
Data: 25 de Novembro de 2024
Programação Orientada a Objetos - 2024/2

---

1. Introdução

O sistema de gerenciamento para um mercado familiar foi desenvolvido para resolver o problema de registros manuais das operações diárias, como controle de estoque, vendas, fluxo de caixa e geração de relatórios. A proposta é modernizar esse processo, automatizando as operações do mercado para reduzir erros, melhorar a organização e proporcionar uma visão mais clara e acessível das atividades do negócio.

---

2. Objetivo do Sistema

O objetivo principal do sistema é substituir o gerenciamento manual por um sistema informatizado, eficiente e fácil de usar, permitindo que o mercado:

- **Controle o fluxo de caixa:** Abrir e fechar o caixa, rastrear vendas e calcular o saldo final.
- **Gerencie o estoque:** Adicionar, buscar e alterar produtos no estoque.
- **Realize vendas de forma automatizada:** Adicionar itens ao carrinho, calcular o subtotal e atualizar o estoque automaticamente.
- **Gere relatórios úteis para o gestor:** Produtos com baixa quantidade ou próximos ao vencimento.

---

3. Funcionalidades Principais

3.1 Gerenciamento do Caixa

- **Abrir o Caixa:** O sistema permite informar o valor inicial no caixa e começar as operações do dia.
- **Registrar Vendas:** Cada venda realizada é registrada, atualizando o saldo no caixa.
- **Fechar o Caixa:** Ao final do dia, o sistema exibe o total de vendas e o saldo final.

3.2 Gerenciamento de Produtos

- Cadastro de produtos em duas categorias:
    - **Produtos Perecíveis (FoodAndUtensils):** Incluem informações de validade.
    - **Produtos Eletrônicos (Electronics):** Incluem informações de garantia.
- Busca de produtos pelo código ou nome.
- Atualização de preços e remoção de produtos.

3.3 Gestão de Vendas

- Adicionar produtos ao carrinho.

- Calcular automaticamente o subtotal durante a venda.
- Finalizar a venda e atualizar o estoque.
- Agrupar itens do carrinho por tipo de produto.

3.4 Relatórios Gerenciais

- **Produtos a vencer:** Lista produtos cuja validade expira dentro de um período específico.
- **Produtos com baixa quantidade:** Lista produtos cujo estoque está abaixo de um limite definido pelo gestor.

---

4. Arquitetura do Sistema

O sistema foi desenvolvido em **Java**, utilizando o paradigma de **Programação Orientada a Objetos (POO)**, com uma estrutura modular para facilitar a extensão e manutenção do código.

4.1 Estrutura de Classes

O sistema possui as seguintes classes:

1. **Classe Abstrata `Product`**
    - Representa produtos genéricos.
    - Atributos:
        - `code`: Código único do produto.
        - `name`: Nome do produto.
        - `unitPrice`: Preço unitário.
        - `stockQuantity`: Quantidade em estoque.
    - Métodos:
        - `updateStock`: Atualiza a quantidade no estoque.
        - `updatePrice`: Altera o preço do produto.
        - `displayDetails`: Exibe os detalhes do produto.
2. **Classe `FoodAndUtensils`**
    - Representa produtos perecíveis.
    - Atributos:
        - `expirationDate`: Data de validade.
    - Métodos:
        - `daysToExpire`: Calcula o número de dias restantes para a validade.
3. **Classe `Electronics`**
    - Representa produtos eletrônicos.
    - Atributos:
        - `warrantyMonths`: Período de garantia em meses.
    - Métodos:
        - Sobrescreve `displayDetails` para incluir informações de garantia.
4. **Classe `CashRegister`**
    - Gerencia as operações do caixa.
    - Atributos:
        - `initialAmount`: Valor inicial no caixa.
        - `currentAmount`: Saldo atual no caixa.
        - `totalSales`: Total de vendas realizadas.

- Métodos:
  - `openRegister`: Abre o caixa com um valor inicial.
  - `closeRegister`: Fecha o caixa e exibe o saldo final.
  - `recordSale`: Adiciona o valor de uma venda ao caixa.

5. **Classe `Sale`**
   - Gerencia uma venda em andamento.
   - Atributos:
     - `saleItems`: Lista de itens da venda.
     - `subtotal`: Subtotal da venda.
   - Métodos:
     - `addProduct`: Adiciona um produto à venda.
     - `finalizeSale`: Finaliza a venda e atualiza o estoque.
6. **Classe `SaleItem`**
   - Representa um item em uma venda.
   - Atributos:
     - `product`: Produto associado ao item.
     - `quantity`: Quantidade vendida.
   - Métodos:
     - `incrementQuantity`: Incrementa a quantidade de um item no carrinho.
7. **Classe `FamilyMarket`**
   - Classe principal que inicializa e executa o sistema.
   - Simula a operação do mercado, incluindo abertura de caixa, registro de vendas e fechamento.

---

5. Código de Exemplo

Exemplo de Uso no `main`

```
public class FamilyMarket {
    public static void main(String[] args) {
        CashRegister register = new CashRegister();
        List<Product> products = new ArrayList<>();
        products.add(new FoodAndUtensils("001", "Apple", 3.5, 100, 7));
        products.add(new FoodAndUtensils("002", "Bread", 1.0, 50, 3));
        products.add(new Electronics("101", "Microwave", 400.0, 5, 12));

        register.openRegister(100.0);

        Sale sale = new Sale();
        sale.addProduct(products.get(0), 10); // 10 apples
        sale.addProduct(products.get(1), 5);  // 5 breads

        double total = sale.finalizeSale();
        register.recordSale(total);

        System.out.printf("Subtotal: $%.2f%n", total);
```

```
        register.closeRegister();
    }
}
```

6. Tecnologias Utilizadas

- **Linguagem:** Java
- **Paradigma:** Programação Orientada a Objetos (POO)
- **Controle de versão:** Git e GitHub

---

7. Modulação

O projeto foi estruturado para manter a separação de responsabilidades, com cada classe representando uma funcionalidade específica. A organização em arquivos separados facilita a manutenção e a escalabilidade.

Estrutura de Arquivos:
Classes/
├── Product.java **Classe abstrata para produtos**
├── FoodAndUtensils.java **Subclasse para produtos perecíveis**
├── Electronics.java **Subclasse para produtos eletrônicos**
├── Sale.java **Classe para gerenciar vendas**
├── SaleItem.java **Classe para representar itens vendidos**
├── CashRegister.java **Classe para gerenciar o caixa**
├── FamilyMarket.java **Classe principal que simula o mercado**

---

8. Conclusão

Este sistema foi projetado para modernizar e automatizar operações de mercados familiares, fornecendo ferramentas eficazes para o controle de caixa, gerenciamento de estoque e registro de vendas. Sua arquitetura modular permite que seja facilmente expandido para atender a novas necessidades.

Este projeto foi desenvolvido por Maria Eduarda Alexandre Aguiar como parte da matéria de Programação Orientada a Objetos, do curso de Engenharia de Software, em 01/12/2024.