Universidade Federal do Cariri

Curso: Bacharelado em Ciência da Computação.

Disciplina: Programação Orientada à Objetos.

Professora: Paola Accioly.

Equipe: Elias Ósbony, João Edson e Nadson Correia.

Mercadinho Padre Cícero

1. Descrição do sistema

O objetivo geral do projeto visa suprir a alta demanda de sistemas de supermercado. O mesmo apresenta as possibilidades de comprar como um cliente, gerenciar contas(CRUD) e gerenciar produtos.

Logo, nosso sistema abrange duas vertentes. A primeira é o cliente cujas funções e métodos atrelados podem ser acessados por meio das classes cliente e mercado, nas quais podemos ter o CRUD do cliente,verificar os seus dados, comprar produtos, verificar o comprovante de compra e ver a lista de produtos disponíveis. Tudo isso no menu do cliente é acessível por meio do login, no qual temos uma função para autenticar cada usuário e separar os respectivos menus de cliente e vendedor.

A segunda é o vendedor cujas funções e métodos são acessados por meio das classes vendedor e mercado, nas quais temos o CRUD do vendedor, consultar clientes/produtos e cadastrar produtos.Também é acessado após o login.

Ambas possuem funcionalidades distintas, e uma classe implementada para auxiliar na compra dos produtos, sendo ela a classe compra. No geral o sistema abrange de forma simples a proposta de um mercado.

2. Backlog

Funcionalidade	**Responsável**
CRUD Cliente	João Edson
CRUD Vendedor	Nadson Correia
CRUD Produto	João Edson

Criação de uma lista de compras feita pelo cliente	Elias Ósbony
Login comparando parâmetros e arraylist	Elias Ósbony
Interfaces e menus no geral	Todos
Remover produtos de estoque ao comprar	Nadson Correia
Descontar saldo ao comprar	João Edson
Adicionar saldo à conta através da alteração de dados do cliente	Elias Ósbony
Vendedor cadastra novos produtos no estoque	Nadson Correia
Vendedor consegue ver todos os clientes do sistema	João Edson
Casos de exceção (em desenvolvimento, pois para as exceções não utilizamos o try catch, mas uma forma temporária está implementada para evitar bugs como uma função para verificar o login do usuário ou um if para verificar se o cliente tem saldo suficiente para comprar)	Todos

3. Arquitetura do Sistema

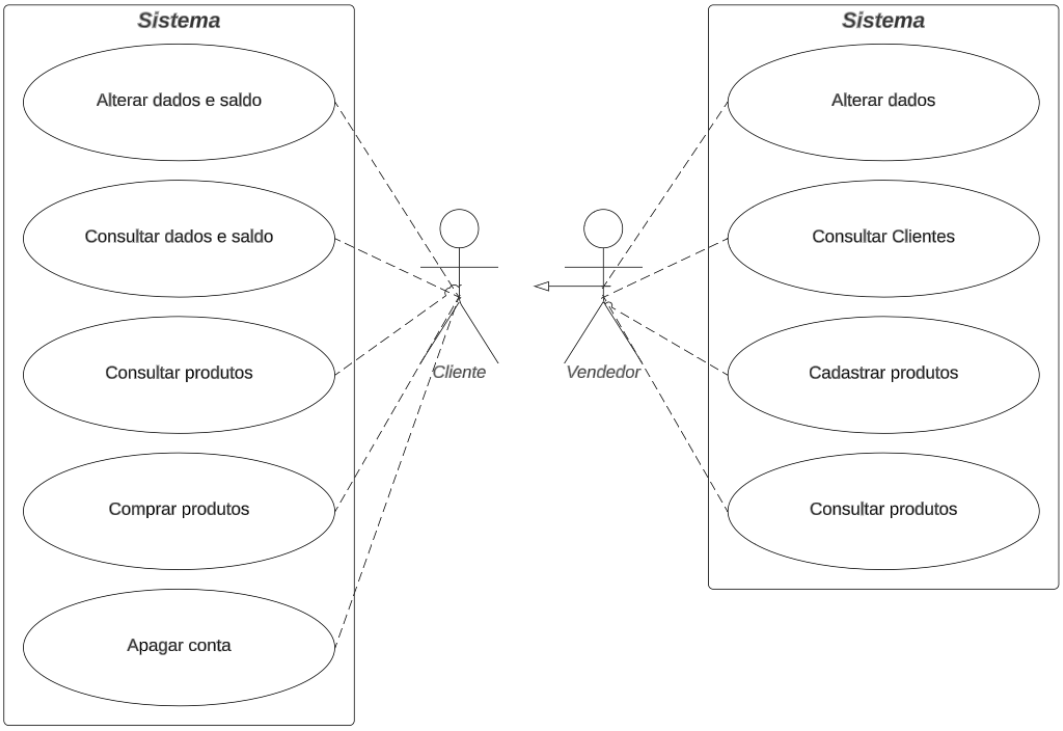

Figura 1: diagrama de casos de uso

Pode-se observar que o vendedor tem funcionalidades administrativas, mas também poderá se cadastrar como cliente. O cliente tem permissão apenas para consultar os seus próprios dados e saldo enquanto o vendedor tem acesso aos dados de todos os clientes. O cliente também pode consultar os produtos cadastrados no sistema pelo vendedor, bem como comprá-los.
Atualização: antes ocorria um bug[1] que mesmo sem saldo suficiente, o cliente conseguia comprar produtos ficando assim com saldo negativo. Para que isso não ocorra, adicionados uma exceção usando uma estrutura condicional para que caso o cliente não tenha saldo suficiente na conta, ele irá ser avisado e não será permitido a compra que está salva na sua lista.

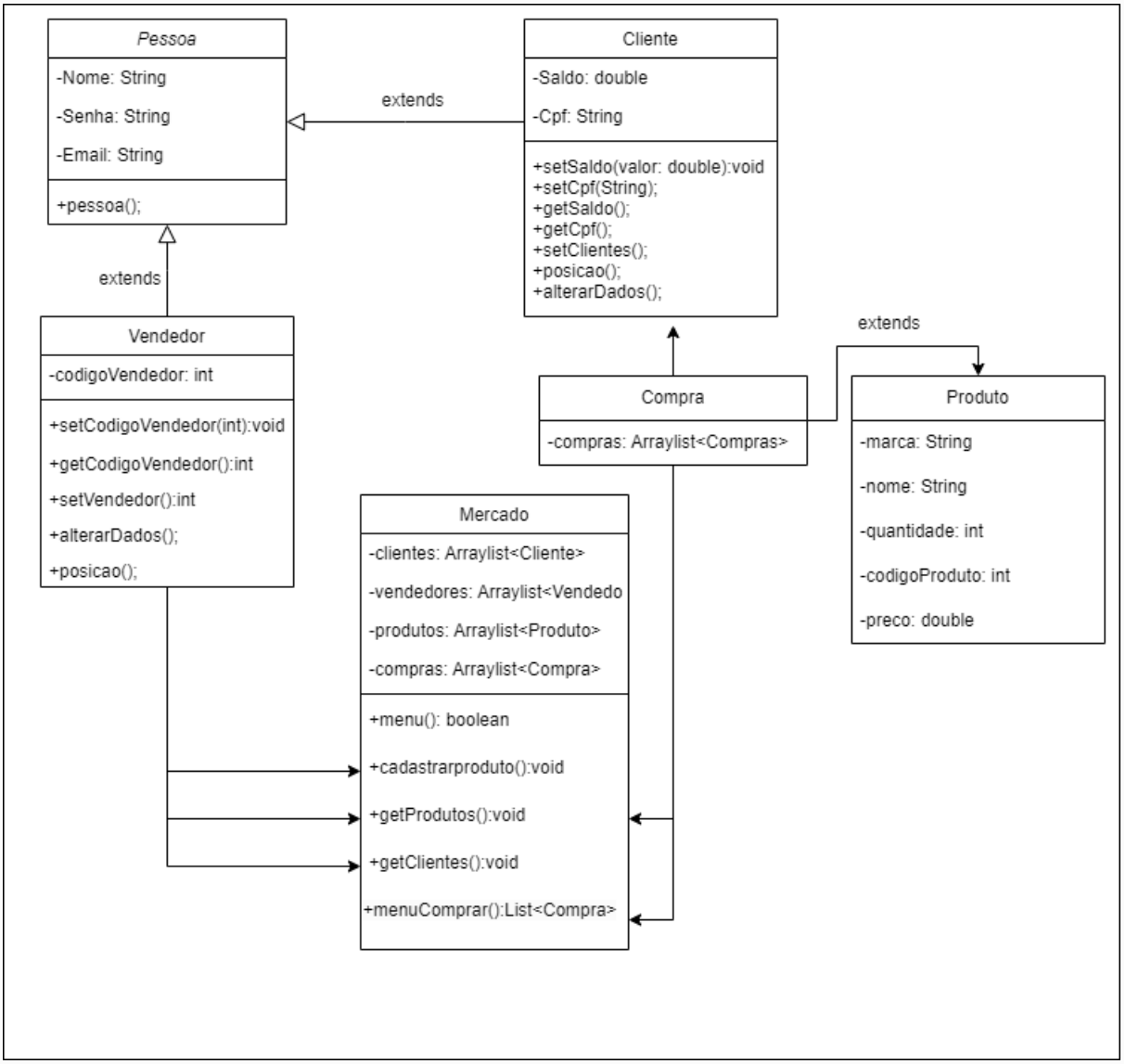

Figura 2: diagrama de classes

As classes “Vendedor” e “Cliente” herdam as variáveis da classe mãe “Pessoa”, bem como “Comprar" herda de “Produto”. Causando assim menos repetição de

[1] Bug compreende uma falha ou erro no código provocando um mal funcionamento do programa.

código fazendo sua reutilização em ambas as classes. Decidimos fazer todas as funcionalidades em apenas uma classe chamada “Mercado” para servir de repositório e receber todas as listas do programa, como a lista de clientes, vendedores e produtos. Também temos um Arraylist que recebe os produtos escolhidos pelo cliente e é jogado para a classe compras para ser armazenado.
Atualização: Utilizando-se melhor do polimorfismo em Java, foi possível implementar o CRUD nas classes cliente e vendedor resultando uma construção melhor do código e economizando também o tempo de execução.