RELATÓRIO DE PROGRAMAÇÃO 2

PROJETO MyFood

Ciência da Computação - UFAL

Paulo Laercio de Oliveira Junior.
Luiz Guilherme Coutinho Braz

1. INTRODUÇÃO

O projeto MyFood é um sistema de delivery desenvolvido com o objetivo de facilitar o cadastro de empresas, como restaurantes, e clientes, além de permitir a criação e edição de pedidos e produtos. Utilizando a metodologia de Desenvolvimento Orientado a Testes (TDD), o sistema assegura que cada funcionalidade seja rigorosamente testada antes de sua implementação, garantindo alta qualidade e confiabilidade.

Funcionalidades Principais

- **Cadastro de Usuários:**
  Permite a criação de contas para três tipos de usuários: Cliente, Dono e Entregador.
- **Cadastro de Empresas:**
  Usuários do tipo Dono podem criar sua própria empresa que pode ser de três tipos: Restaurante, Farmácia e Mercado.
- **Gerenciamento de Pedidos:**
  Clientes podem fazer pedidos de produtos oferecidos pelas empresas cadastradas.
- **Gerenciamento de Entregas:**
  Usuários do tipo Entregador realizam entregas de produtos das empresas para usuários do tipo cliente.
- **Persistência de Dados:**
  Utiliza arquivos XML para simular um banco de dados, armazenando informações sobre usuários, empresas, produtos e pedidos.

2. DESIGN ARQUITETURAL DO SISTEMA

O sistema MyFood é estruturado em várias camadas, cada uma responsável por uma parte específica da funcionalidade do sistema. Essa abordagem modular facilita a manutenção, a escalabilidade e a reutilização do código. A seguir, detalhamos as principais camadas e componentes do sistema:

2.1. Models

Responsabilidade: Representar os dados e a lógica de negócios do sistema.

Função: Inclui classes que modelam os dados essenciais do sistema, como Cliente, Dono, Empresa, Produto, Pedido, entre outros.

2.2. Exception

Responsabilidade: Lidar com exceções e erros que ocorrem durante a execução do sistema.

Função: Garante que os erros sejam tratados de maneira consistente e informativa, melhorando a robustez do sistema.

2.3. Database

Responsabilidade: Cuidar da persistência dos dados.

Função: Gerencia a leitura e escrita de dados em arquivos XML, simulando um banco de dados. Exemplos incluem classes como PersistenciaEmpresas, PersistenciaPedidos, etc.

3. PRINCIPAIS COMPONENTES E SUAS INTERAÇÕES

3.1. Usuário:

O componente Usuario é uma classe abstrata que serve como base para diferentes tipos de usuários no sistema MyFood, como Cliente, Dono e Entregador. Ele inclui atributos comuns a todos os usuários, como id, nome, email, senha e endereço, e métodos para validar esses dados.

3.2. Cliente:

O componente Cliente é uma classe que herda de Usuário e representa um cliente no sistema MyFood..

3.3. Dono:

O componente Dono é uma classe que herda de Usuario e representa um dono de restaurante no sistema MyFood. Ele inclui informações específicas do dono, como CPF, além dos atributos herdados de Usuario.

3.3. Entregador:

O componente Entregador é uma classe que herda de Usuario e representa um entregador no sistema MyFood. Ele inclui informações específicas do entregador, como veículo, placa, lista de empresas associadas e status de disponibilidade.

3.4. Empresa:

O componente Empresa é uma classe abstrata que serve como base para diferentes tipos de empresas no sistema MyFood, como Mercado, Farmácia e Restaurante. Ele inclui todos os atributos comuns que representam uma empresa, como id, nome, endereço, dono e uma lista de entregadores associados.

3.5. Mercado:

O componente Mercado é uma classe que herda de Empresa e representa um mercado no sistema MyFood. Ele inclui informações específicas do mercado, como horários de abertura e fechamento, e o tipo de mercado.

3.6. Restaurante:

O componente Restaurante é uma classe que herda de Empresa e representa um restaurante no sistema MyFood. Ele inclui informações específicas do restaurante, como o tipo de cozinha oferecida.

3.7. Farmácia:

O componente Farmacia é uma classe que herda de Empresa e representa uma farmácia no sistema MyFood. Ele inclui informações específicas da farmácia, como se está aberta 24 horas e o número de funcionários.

3.8. Entrega:

O componente Entrega representa uma entrega no sistema MyFood. Ele inclui informações específicas da entrega, como o id, o cliente, a empresa, o pedido, o entregador, o destino e a lista de produtos.

3.9. Pedido:

O componente Pedido representa um pedido no sistema MyFood. Ele inclui informações específicas do pedido, como cliente, empresa, estado do pedido, lista de produtos e valor total.

3.10. Produto:

O componente Produto representa um produto no sistema MyFood. Ele inclui informações detalhadas sobre os produtos oferecidos pelas empresas cadastradas no sistema. Os atributos principais do Produto são: identificador único (id), identificador da empresa que oferece o produto (empresaId), nome do produto, valor do produto e categoria à qual o produto pertence (por exemplo, bebidas, alimentos, etc.).

3.11. Sistema:

O componente Sistema gerencia o funcionamento geral do sistema MyFood. Ele mantém mapas de dados para usuários, empresas, produtos, pedidos e entregas, carregando e persistindo esses dados através de classes de persistência. O Sistema também inclui métodos para criar usuários, obter informações de usuários e empresas, e zerar o sistema.

4. PADRÕES DE PROJETO

3.1. Facade:

Resumo Geral: O padrão de design Facade tem como objetivo oferecer uma interface mais simples para sistemas complexos, criando uma

camada intermediária que esconde detalhes complicados e torna o uso mais acessível para o cliente.

Problema Resolvido: O Facade enfrenta o desafio da complexidade em sistemas e subsistemas, permitindo uma interação mais direta e simples com o sistema ao ocultar as funcionalidades complicadas. Ele proporciona um meio fácil de acessar as funcionalidades, sem que o usuário precise lidar com toda a complexidade subjacente.

Identificação da Oportunidade: No projeto MyFood, existem diversos modelos e processos que são complexos, além de várias validações e interações. A adoção do padrão Facade simplifica o acesso a esses processos, oferecendo métodos que integram e atendem aos requisitos do sistema, e ao mesmo tempo, torna a interface mais intuitiva tanto para os usuários quanto para os testes.

Aplicação no Projeto: O padrão Facade foi implementado para criar uma única interface, organizando as interações entre o cliente e o sistema. Ele oculta a complexidade interna e disponibiliza métodos que permitem realizar ações sem a necessidade de entender profundamente a lógica de funcionamento, facilitando assim o uso e a manutenção do sistema.