UEPB
Universidade Estadual da Paraíba
Centro de Ciências e Tecnologia – CCT
Departamento de Computação - DC

DISCIPLINA: MÉTODOS AVANÇADOS DE PROGRAMAÇÃO

Documentação do projeto - Sistema Marketplace (2023.1)

DISCENTES:

LUCAS FAUSTO MEDEIROS
KAÍQUE DA SILVA IVO
MIKAELLE DOS SANTOS OLIVEIRA
NATÁLIA MARIA DE ARAÚJO LIMA
PEDRO HENRIQUE FRANÇA
RAQUEL GOMES DE VASCONCELOS DA SILVEIRA

Profª.Dra SABRINA DE FIGUEIRÊDO SOUTO

CAMPINA GRANDE, Junho de 2023.

Descrição do Sistema

O sistema de *Marketplace* foi projetado com uma estrutura em etapas, onde cada uma é responsável por uma função específica. De início, por exemplo, temos a **Interface do Usuário**, que é responsável por lidar com a interação do usuário com o sistema. Ela exibe as telas, formulários e elementos de entrada para que os usuários possam interagir com as funcionalidades do sistema. Nessa etapa, é possível ter componentes de interface gráfica ou uma interface de linha de comando.

Na etapa de **Serviço**, é contida a lógica de negócio do sistema. Ela coordena as operações entre a camada de interface do usuário e a camada de acesso a dados (DAO). Os serviços recebem as requisições do usuário, processam os dados necessários, interagem com as classes de entidade e realizam as operações necessárias. Ela também pode conter validações, cálculos e outras regras de negócio.

A de **Acesso a Dados (DAO)** é responsável pelo acesso e manipulação dos dados persistentes. Ela possui classes DAO correspondentes a cada entidade do sistema (Loja, Comprador, Produto) e fornece métodos para criar, ler, atualizar e excluir os objetos dessas entidades. Essa etapa abstrai os detalhes de persistência, permitindo que o restante do sistema trabalhe com os objetos de entidade de forma independente da tecnologia de armazenamento utilizada (no caso, a serialização para salvar em arquivos).

E por fim, a de **Domínio,** nela contém as classes que representam os objetos principais do sistema (Loja, Comprador, Produto). Essas classes possuem os atributos, métodos de acesso (getters e setters), métodos auxiliares e podem conter as regras de negócio relacionadas aos objetos.

A respeito dos *patterns* usados no sistema, podemos citar os seguintes padrões de projeto, que levaram em conta as necessidades e requisitos específicos do sistema, bem como as escolhas de arquitetura feitas durante o desenvolvimento;

1. O **Padrão DAO (Data Access Object):** O padrão DAO foi utilizado para separar a lógica de acesso aos dados (persistência) das demais camadas do sistema. Foram criadas classes DAO (do pacote src.DAO), como 'LojaDAO', 'CompradorDAO', 'ProdutoDAO', que são responsáveis por realizar operações de criação, leitura, atualização e exclusão (CRUD) nos objetos de entidade correspondentes (Lojas, Compradores e Produtos). A posteriori foram incluídas as classes 'CarrinhosDeCompraDAO' e 'HistoricoDeComprasDAO', a primeira responsável por adicionar os itens e removê-los e a segunda por registrar o histórico de compras do comprador.Logo,isso permite uma maior modularidade e flexibilidade no gerenciamento dos dados. são responsáveis por fornecer métodos para interagir com os dados, como cadastrar, buscar, atualizar e remover objetos

2. O **Padrão MVC (*Model-View-Controller*):** Embora não tenha sido mencionado explicitamente, o nosso sistema de marketplace pôde ser estruturado seguindo o padrão MVC. O modelo (*Mode*l) é representado pelas classes de entidades (Lojas, Compradores e Produtos) e suas respectivas classes DAO. A classe Main pode funcionar como uma camada de visualização (*View)* simples, onde as operações são realizadas por meio de chamadas diretas aos métodos dos DAOs. O controlador (*Controller*) é representado pelas classes do pacote src.DAO, elas gerenciam a interação entre o modelo e os dados persistentes. No entanto, é importante notar que a implementação completa do padrão MVC exigiria uma separação mais clara entre a parte de visualização e a lógica de negócios.

3. O **Padrão Exception:** No sistema são utilizadas classes de exceção personalizadas, como: 'LojaNotFoundException', 'CompradorNotFoundException' , 'ProdutoNotFoundException' e na segunda release foi adicionado o 'HistoricoNotFoundException' para lidar com casos em que um objeto não é encontrado.Além disso, também foi adicionada uma exceção 'HistoricoJaExistente', no intuito de verificar se existem compras repetidas. Essas exceções estendem a classe base *Exception* e fornecem informações específicas sobre o objeto não encontrado. O uso de exceções ajuda a tratar erros e situações excepcionais de forma adequada.

4. **O Padrão Fachada**: Todas as classes DAO do sistema se comunicam com a fachada e ela se comunica com main, simplificando assim, o acesso as classes DAO. Uma fachadaDAO foi implementada com o intuito de simplificar o acesso e a manipulação dos diferentes DAOs relacionados aos objects do sistema, como: LojaDAO, CompradorDAO, ProdutoDAO, CarrinhoDeComprasDAO, e HistoricoDeComprasDAO. A fachadaDAO simplifica o acesso aos dados no sistema , fornecendo uma interface abstrata e simplificada para interagir com os diferentes DAOs, tornando o código mais organizado, coeso e facilitando a manutenção e evolução do sistema.

.

Diagrama de Classe