RELATÓRIO – PROJETO INTEGRADOR

Nome do aluno - Eduardo Andina

Link do Projeto no GitHub -
https://github.com/EduardoAndina/GameStock-refatorado

Sistema GameStock - Refatoração e Preparação para Sistema Web

Introdução

Este relatório apresenta o processo de refatoração e reorganização do sistema GameStock. A refatoração concentrou as regras de negócio nas classes Service, deixando as telas responsáveis apenas pela interface com o usuário e os DAOs pelo acesso ao banco de dados. Também foram aplicados princípios SOLID, padronização de métodos CRUD e melhorias nas telas..

Nesta etapa, foram implementadas e refatoradas funcionalidades relacionadas a:

- Gestão de jogos (cadastro, listagem, busca, edição e exclusão)
- Registro de vendas (seleção de jogos, cadastro de cliente, quantidade e valor pago, atualização de estoque)
- Gestão de funcionários (cadastro, listagem, exclusão e login)
- Refatoração do código para centralização da lógica de negócio nas classes Service
- Implementação de métodos de busca em Vendas (buscarPorJogo, buscarPorCliente) no Controller
- Atualização das telas Swing, incluindo melhorias na tabela de vendas, busca dinâmica, seleção de vendas para edição e refatoração da tela de login com integração ao FuncionarioService e método autenticar()

Visão Geral do Sistema

O sistema GameStock, até o momento, possui funcionalidades e entidades implementadas:

Jogos

- Atributos: ID, Nome, Categoria, Plataforma, Preço, Quantidade
- Funcionalidades: cadastro, listagem, busca por nome, edição e exclusão

Vendas

- Atributos: ID do jogo, Nome do jogo, Categoria, Quantidade vendida, Valor pago, Nome e Telefone do cliente
- Funcionalidades: seleção de jogo, registro e atualização de venda, validações de quantidade e valor, atualização automática do estoque, busca por jogo ou cliente
- Implementações novas: métodos no Controller para listar todas as vendas e buscar por jogo ou cliente, integração com as telas para busca dinâmica e atualização de tabela

Funcionários

- Atributos: Nome, Usuário e Senha
- Funcionalidades: cadastro, listagem, exclusão e login
- Implementações novas: refatoração da tela de login utilizando FuncionarioService, com método autenticar() para centralizar a lógica de autenticação e remover acesso direto ao DAO, garantindo que a tela seja responsável apenas pela interface e interação com o usuário

Separação de Responsabilidades

O projeto foi reorganizado, cada um com responsabilidade clara:

Camada	Descrição
Model	Representa entidades do sistema (Jogo, Venda, Funcionário)
DAO	Responsável pelo acesso ao banco de dados (CRUD), sem métodos estáticos
Service	Contém regras de negócio e validações, centralizando lógica de cadastro, atualização e exclusão

Controller	Faz a ponte entre View e Service, implementando métodos de listagem e busca em Vendas
View (Swing)	Interface gráfica, interação com usuário, tabelas com busca dinâmica, seleção de vendas, atualização de campos e tela de login.

Atualizações nas Classes

Durante o processo de refatoração, todas as classes do sistema foram afetadas e atualizadas, exceto as classes de Model (Jogo, Venda, Funcionário).

As classes Model representam somente as entidades do sistema, reunindo os atributos e os getters e setters. Como funcionam apenas como estruturas de dados, sem regras de negócio ou interação com banco de dados e interface, não houve necessidade de modificações nessas classes.

Por outro lado, as camadas DAO, Service, Controller e View passaram por refatorações para melhorar a organização do código, concentrar as regras de negócio no local adequado e garantir uma organização melhor no código. Essas mudanças seguem os princípios SOLID e tornam o sistema mais fácil de manter, evoluir e adaptar para uma futura versão web.

Princípios SOLID Aplicados

SRP – Princípio da Responsabilidade Única
Cada classe tem uma única responsabilidade:

Classe	Responsabilidade
Jogo, Venda, Funcionario	Representar entidades
JogoDAO, VendaDAO, FuncionarioDAO	Executar operações SQL e acesso ao banco
JogoService, VendaService, FuncionarioService	Centralizar regras de negócio e validações, incluindo cadastro, atualização e exclusão, atualização de estoque e autenticação de funcionários
VendaController	Centralizar métodos de listagem e busca de vendas para uso nas telas
Telas Swing	Interface com usuário, busca dinâmica, seleção de vendas, atualização de tabela e login

Refatorações Aplicadas

- **Centralização da Lógica de Negócio**: validações de campos e lógica de autenticação foram movidas das telas para os Services.

- **Tela de Login**: refatorada para usar FuncionarioService e método autenticar(), removendo a lógica de verificação de usuário e senha do JFrame.

- **VendaController**: métodos implementados para listar todas as vendas e buscar por jogo ou cliente, integrados às telas para atualização dinâmica da tabela.

- **Try-With-Resources**: garantido fechamento automático de conexões com banco em todos os DAOs.

- **Padronização de CRUD**: métodos padronizados (listar(), buscar(), registrar(), atualizar(), excluir()) aplicados a Jogos, Vendas e Funcionários.