Programação Orientada a Objetos

Trabalho Final:
Sistema de Avaliação de Filmes

Gabriella Castelari Gonçalves Nº USP: 14755082

Isabela Müller Martins Rocha Nº USP: 14579732

Thaís Laura Anício Andrade Nº USP: 14608765

Docente: Prof. Andre de Freitas Smaira

ENGENHARIA DA COMPUTAÇÃO

USP CAMPUS DE SÃO CARLOS

2024

Sumário

1 Descrição do Projeto

O projeto é baseado em um sistema de avaliação e recomendação de filmes, desenvolvido a partir da base de dados do **IMDb**. O sistema permite que os usuários avaliem filmes, sigam amigos e visualizem suas avaliações, tornando a experiência mais interativa e social.

As recomendações são personalizadas de acordo com os gêneros escolhidos, os filmes com as maiores notas no banco de dados e as obras mais famosas do ator. A ideia surgiu para resolver a indecisão comum na escolha de filmes, proporcionando uma maneira de descobrir obras semelhantes às que já gostamos. Além disso, o sistema incentiva os usuários a expressarem suas opiniões por meio de notas e comentários, tornando o formato mais convidativo tanto para as avaliações rápidas quanto para as mais detalhadas.

2 Planejamento e Estrutura

Para organizar e estruturar o desenvolvimento do trabalho, nosso grupo se reuniu e utilizou a ferramenta Miro para nos auxiliar no planejamento. Com ela, criamos um mapa mental que nos permitiu dividir as tarefas entre os membros, definir os prazos e discutir ideias para a implementação do nosso sistema de avaliação e recomendação de filmes.

Para que o trabalho fosse realizado de acordo com o que foi pedido, estruturamos cinco classes principais, sendo uma delas a base das outras: a classe **Filme**. Também desenvolvemos as classes **Usuario**, **Avaliacao**, **Registro**, **Comedia**, **Romance**, **Terror**, **Drama**, **Suspense** e **Acao**, classes focadas nos gêneros dos filmes.

A classe Filme foi projetada como o ponto central de interação do sistema. Seus atributos, como nota, comentários, gênero, subgênero, duração, sinopse e classificação, possibilitam que o usuário conheça novos filmes e avalie aqueles que já assistiu, criando um sistema que facilita tanto a descoberta de novos filmes quanto a recomendação de títulos a outros usuários.

A classe Usuário é outra classe essencial do nosso projeto. Seus atributos, como nome, senha, email e gêneros favoritos, possibilitam que o sistema se personalize para cada usuário. A lista de amigos e a funcionalidade para cálculo da compatibilidade entre usuários foram implementadas para promover maior interação no sistema, além de auxiliar na descoberta de novos filmes semelhantes àqueles que o sistema já sabe que o usuário gosta.

Já a classe Avaliação foi projetada para representar cada avaliação feita pelos usuários, sendo composta por atributos como nota, comentário e o usuário que realizou a avaliação. Essa classe está ligada à classe Filme, garantindo que cada avaliação esteja associada a um filme específico. Os métodos dessa classe permitem adicionar notas e comentários de maneira organizada e eficiente.

Para representar diferentes gêneros de filmes, criamos classes específicas, como Romance, Suspense, Terror, Comédia e Ação. Cada uma delas possui propriedades específicas para enriquecer a descrição dos filmes. Criamos métodos virtuais para atender a especificação do polimorfismo, além de trazer flexibilidade ao código.

Os relacionamentos entre as classes foram cuidadosamente planejados. Usuário, Avaliação e Gênero foram conectados à classe Filme, garantindo que avaliações fossem parte

integrante dos filmes, mas mantendo a independência lógica das outras entidades.

O projeto foi desenvolvido com base em uma seleção de filmes escolhidos para cada gênero, sendo cinco títulos por gênero. Além disso, todos os princípios utilizados nesse trabalho foram aprendidos em aula. Aplicamos, por exemplo, o encapsulamento para proteger os dados e garantir que apenas métodos públicos pudessem ser acessados pelos usuários. Esse planejamento assegurou que o aplicativo fosse desenvolvido conforme imaginamos, oferecendo uma busca personalizada e feedbacks precisos em relação ao cinema.

3 Diagrama de Classes e Casos de uso

Link para o quadro no Miro

O diagrama de classes UML apresenta os relacionamentos existentes entre as classes que compõem o projeto. Desse modo, é importante entender cada tipo de relação:

1. **Relacionamento entre "Usuario" e "Registro"**

   A classe *Usuario* possui um relacionamento de um para um com a classe *Registro*. Então, cada instância de *Usuario* está associada a um *Registro* e, assim, cada *Registro* está associado a um único *Usuario*.

2. **Relacionamento entre "Filme" e suas Heranças**

   A classe *Filme* é principal para suas subclasses, sendo elas: *Suspense*, *Acao*, *Comedia*, *Animacao*, *Drama*, *Terror*, *Romance*. Com isso, todos os tipos de gêneros herdam os atributos e métodos da classe *Filme*.

3. **Relacionamento entre "Avaliacao" e "Filme"**

   A classe *Avaliacao* tem um relacionamento de zero a muitos com a classe *Filme*. Isso significa que cada avaliação é referente a um único filme e, assim, um filme pode ter muitas avaliações feitas por usuários, ou nenhuma avaliação.

4. **Relacionamento entre "Avaliacao" e "Usuario"**

   A classe *Avaliacao* tem um relacionamento de zero a muitos com a classe *Usuario*. Então, um usuário pode não ter feito nenhuma avaliação ou pode ter feito várias avaliações de filmes.

5. **Relacionamento entre "Filme" e "Avaliacao"**

   A classe *Filme* apresenta um relacionamento de um para muitos com a classe *Avaliacao*. Desse modo, cada filme pode ter várias avaliações, de diferentes usuários, mas cada avaliação está associada a um único *Filme*.

6. **Relacionamento entre "Usuario" e "Avaliacao"**

   A classe *Usuário* está relacionada com a classe *Avaliação* em uma associação de um

para muitos. Isso significa que um usuário pode fazer várias avaliações, mas cada avaliação pertence a apenas um usuário.

4 Implementação e Funcionalidade

A implementação do sistema de avaliações foi organizada em módulos, o que garantiu uma boa estrutura e de fácil gerenciamento. Os principais componentes incluem: Usuário, Filmes, Avaliações, Gerenciamento de Registro e Login e Armazenamento de Dados.

No módulo usuário, são gerenciadas todas as informações relacionadas ao usuário, como login, senha, e-mail, gêneros favoritos e a lista de avaliações feitas. Esse módulo também permite a interação entre os usuários, como adicionar e remover amigos, avaliar, comentar e recomendar filmes, mantendo essas informações organizadas e atualizadas.

Há o gerenciamento de cadastro e login, que assegura a validação correta das informações fornecidas pelos usuários, como e-mail, senha e nome de usuário. Esse controle garante que apenas usuários autenticados possam acessar as funcionalidades do sistema, como a avaliação e recomendação de filmes.

O módulo filmes é responsável por armazenar todas as informações detalhadas de cada filme presente no sistema. Cada filme contém atributos como título, gênero, elenco, classificação, ano de lançamento, duração e uma lista com todas as avaliações feitas pelos usuários. Com a intenção de proporcionar maior flexibilidade, criamos também classes derivadas da classe Filme, representando os principais gêneros dos filmes, como Ação, Comédia, Terror, Romance, Suspense e Drama. Essas classes adicionam descrições específicas para cada gênero, utilizando a herança para estender as funcionalidades da classe original.

Já o módulo avaliação organiza os dados das avaliações em si, associando cada nota e comentário a um filme específico e ao usuário que realizou a avaliação. Dessa forma, o sistema mantém as informações, permitindo fácil acesso tanto pelo lado do usuário quanto pelo banco de dados de filmes.

O banco de dados foi implementado utilizando arquivos JSON, com o auxílio da biblioteca nlohmann/json. Os dados de usuários e filmes, incluindo as avaliações associadas, são armazenados nos arquivos armazUsuarios.json e armazFilmes.json. Durante a inicialização do programa, essas informações são carregadas automaticamente para a memória, garantindo um funcionamento eficiente e contínuo. Ao finalizar a execução, todos os dados atualizados são salvos novamente nos arquivos, preservando as informações entre sessões.

Entre as principais funcionalidades do sistema, destacam-se o cadastro de novos usuários, onde são inseridos dados como nome completo, login, senha e gêneros favoritos; o login, que autentica os usuários e dá acesso às demais funcionalidades; o gerenciamento de amigos, que possibilita adicionar ou remover amigos da lista de contatos; e a avaliação de filmes, em que os usuários podem buscar filmes pelo nome, visualizar suas informações, atribuir notas (de 0 a 10) e adicionar comentários. Todas as avaliações feitas são armazenadas tanto no perfil do usuário quanto na lista de avaliações associadas ao respectivo filme.

Outra funcionalidade importante é a recomendação de filmes, que sugere obras com base no gênero ou em atores específicos, permitindo ao usuário explorar novas opções a partir do banco de dados disponível. Esse banco de filmes é organizado em estruturas de dados eficientes, como std::unordered$_map, que garante rapidez na busca e manipulação das informações.$

Do ponto de vista técnico, o sistema utiliza conceitos fundamentais da programação orientada a objetos, como encapsulamento, herança e sobrecarga de operadores. O uso do encapsulamento garante a proteção dos atributos das classes, expondo apenas os métodos necessários para sua manipulação. A classe Avaliacao, por exemplo, implementa a sobrecarga do operador ¡¡ para facilitar a exibição das avaliações.

O fluxo de execução do programa é simples e intuitivo. Ao iniciar, o sistema carrega os dados salvos nos arquivos JSON e apresenta um menu principal, no qual o usuário pode escolher entre cadastrar-se, realizar o login ou sair do programa. Após o login, o usuário é direcionado a um menu específico, onde pode gerenciar amigos, avaliar filmes ou receber recomendações. Ao encerrar a execução, todos os dados atualizados são salvos nos arquivos JSON, garantindo que nenhuma informação seja perdida. Em resumo, o sistema implementado integra de forma eficiente as funcionalidades de cadastro, login, avaliação e recomendação de filmes

5 Guia de Uso

Primeiramente, é necessário baixar os seguintes arquivos da base de dados do IMDb (link para IMDb):

1. title.basics.tsv.gz
2. title.principals.tsv.gz
3. name.basics.tsv.gz
4. title.ratings.tsv.gz

Após o download, é necessário extrair esses arquivos para a mesma pasta onde estão os arquivos do programa. Em seguida, como os nomes possuem pontos finais entre as palavras, foi necessário renomear os arquivos conforme a tabela abaixo:

Original	**Mudança**
title.basics.tsv.gz	infoFilmes
title.principals.tsv.gz	filmesAtores
name.basics.tsv.gz	nomesPessoas
title.ratings.tsv.gz	mediasAvaliacoes

Tabela 1: Tabela de correspondência de nomes

Nossos testes ocorreram todos em linux (Ubuntu), então será mais fácil utilizar o programa rodando apenas o makefile na mesma pasta onde estão todos os outros arquivos (código, json e base de dados). Portanto, em terminal com o mesmo caminho da pasta do programa, é necessário compilar com o comando **make** e rodar com **make run**. Após essas ações, o menu será apresentado na tela e o sistema de avaliações e recomendações de filmes poderá ser utilizado.

Primeiramente, é necessário realizar um cadastro (opção 1). Será necessário nome completo, nome de login, e-mail, gêneros favoritos e senha. Após isso, realiza-se o login (opção

2) para encontrar um outro menu de funcionalidades próprias do sistema de avaliação e recomendação de filmes. Para facilitar o teste, foi enviado o arquivo de armazenamento de usuários com algumas contas. Para logins ou adição/remoção de amigos, pode-se utilizar:

Login	**Senha**
gabi_c	gabi123
isa.muller	isa123
thaislaura_	thais123

Tabela 2: Tabela de correspondência de nomes

Em seguida, é possível realizar as seguintes funcionalidades:

1. Adicionar amigo (pode-se adicionar quaisquer uma das contas descritas acima)
2. Remover amigo
3. Avaliar um filme
4. Verificar qual é a pior e a melhor avaliação de um filme
5. Obter recomendações de filmes por gênero ou ator
6. Verificar o próprio perfil
7. Sair da sua conta

Para a avaliação do filme, é necessário inserir o nome dele em inglês (não importa se está em maiúsculo ou minúsculo). Algunss exemplos são:

- The Godfather
- Spider-Man: Into the Spider-Verse
- Toy Story
- Up

- The Shining
- Life is Beautiful

Para a recomendação de gênero, é possível buscar por sete: ação, romance, comédia, suspense, terror, animação e drama. Para o ator, estão alguns exemplos a seguir:

- Tom Cruise
- Adam Sandler
- Heath Ledger
- Morgan Freeman

Por fim, pela opção 3, finaliza-se o programa. Após iniciar a sessão seguinte, os dados de usuários, filmes e avaliações são recuperados e podem ser utilizados na sessão atual.

6 Desafios e Soluções

Ao longo do desenvolvimento deste trabalho, surgiram diversos desafios que exigiram planejamento e pesquisa para obter soluções eficientes. Uma das primeiras dificuldades enfrentadas foi a escolha do tipo de arquivo a ser utilizado para armazenar os dados dos usuários, filmes e avaliações. As opções consideradas incluíam arquivos CSV, JSON e binários e, assim, optamos em modelar pelo JSON devido à sua flexibilidade e pela facilidade proporcionada por bibliotecas modernas, como a nlohmann/json.

Outro tópico em que tivemos dúvida foi para definir qual tipo de estrutura utilizar para guardar todos os objetos criados. Nesse sentido, utilizamos vetores e std::unordered_map, sendo eficiente para buscas e para o armazenamento de dados, como as avaliações feitas pelos usuários. Ademais, esse problema também se associou com a atualização das avaliações em múltiplas classes, ao adicionar ou atualizar uma avaliação, pois exigiu a elaboração de métodos que fizessem a atualização simultânea nas estruturas envolvidas, a fim de manter os dados consistentes.

Também, devido às classes derivadas de Filme, foi trabalhoso o processo de manipulação dos objetos de diferentes gêneros em estruturas de map e vector. Como foram criados métodos virtuais, como a descrição de cada gênero, foi necessário utilizar variáveis dinâmicas para cumprir o requisito de polimorfismo. Assim, foi feita uma fábrica de objetos, o qual determinava qual classe utilizar com base no gênero, retornando um ponteiro de acordo com as especificações para o Filme (classe base).

A falta de planejamento em algumas funcionalidades como o cadastro e login dos usuários atrapalharam no desenvolvimento. Isso se justifica pois anteriormente o cadastro incluía apenas o nome de usuário, senha e gêneros favoritos. Porém, notamos que, caso o usuário esquecesse a senha, seria necessário outro método de recuperação. Assim, tivemos que incluir o e-mail como campo obrigatório no cadastro, para possibilitar a recuperação de senha.

Desta maneira, todos esses desafios nos ensinaram a importância do planejamento, assim como a escolher qual a melhor implementação de estrutura de dados e bibliotecas para implementação eficiente do sistema. Com a superação de cada dificuldade, concluímos esse trabalho com motivação e muitos aprendizados.