Sumário

Relatório de implementação de interface gráfica e conexão com o banco de dados MySQL do sistema MDFlix

INTERFACE GRÁFICA

Para interface gráfica foi utilizado a IDE NetBeans, onde toda a estrutura de interação com usuário foi projetada. A seguir um pouco de cada classe:

Login.java:

Essa classe constitui a interface gráfica responsável pelo formulário de login no meu sistema denominado "MD FLIX". Destaco algumas funcionalidades relevantes dessa implementação:

Representação Gráfica: Herdei a classe `javax.swing.JFrame`, o que a caracteriza como uma janela gráfica. Esta classe define a disposição e aparência dos componentes visuais, compreendendo rótulos (`JLabel`), campos de texto (`JTextField`), campos de senha (`JPasswordField`), e botões (`JButton`), todos destinados a coletar informações do usuário.

Inicialização e Configuração: O método `initComponents`, chamado no construtor `Login()`, desempenha o papel fundamental de inicializar e configurar todos os elementos visuais. Essa abordagem assegura que a interface esteja pronta para utilização imediata após a instanciação da classe.

Manipulação de Eventos de Botão: Métodos como `jButton1ActionPerformed` são associados aos eventos de clique dos botões presentes na interface. No caso do botão "Entrar", verifica-se se as credenciais fornecidas (nome de usuário e senha) coincidem com os valores esperados. Se a autenticação for bem-sucedida, a interface é redirecionada para outra janela (`Usuarios`); caso contrário, uma mensagem de erro é exibida por meio do componente `JOptionPane`.

Registro de Novos Usuários: O método `jButton2ActionPerformed` é invocado quando o botão "Cadastrar" é pressionado, instanciando a classe `Cadastro` e exibindo sua interface correspondente.

Finalização da Janela Atual: A utilização de `this.dispose()` após certas ações de botão é adotada para encerrar a instância da janela de login, promovendo uma transição ordenada entre as diferentes telas do sistema.

Em resumo, essa classe materializa a interface de login para o sistema "MD FLIX", permitindo que os usuários insiram suas credenciais para acessar o sistema ou optem por se cadastrar para obter uma conta.

Em resumo, essa classe materializa a interface de login para o sistema, permitindo que os usuários insiram suas credenciais para acessar o sistema ou optem por se cadastrar para obter uma conta.

Usuarios.java:

Essa classe Java representa a tela de seleção de usuários no sistema. Aqui estão algumas informações relevantes sobre o que esta classe realiza:

Layout e Componentes Visuais: A classe `Usuarios` é uma extensão de `javax.swing.JFrame`, indicando que representa uma janela gráfica. O layout inclui quatro botões representando diferentes usuários (User 1, User 2, User 3, User 4).

Escolha do Usuário: Quando um dos botões de usuário é clicado, o método de evento correspondente (`jButton1ActionPerformed`, `jButton2ActionPerformed`, `jButton3ActionPerformed`, `jButton4ActionPerformed`) é acionado. Cada um desses métodos cria uma instância da classe `TelaPrincipal` (tela principal do sistema) e a torna visível, enquanto fecha a tela atual.

Estilo Visual: Cada botão tem uma cor de fundo diferente para distinguir visualmente os usuários.

Encerramento e Abertura de Telas: Para abrir a tela principal, a classe atual é fechada (`this.dispose()`) e uma nova instância de `TelaPrincipal` é criada e tornada visível.

Em resumo, a classe `Usuarios` é responsável por fornecer uma interface para os usuários selecionarem um perfil antes de entrar na tela principal do sistema. Atualmente, todos os usuários têm o mesmo comportamento, mas essa lógica pode ser personalizada para cada usuário no futuro, conforme necessário.

Cadastro.java:

Esta classe representa a interface gráfica responsável pelo cadastro de novos usuários no sistema. Aqui estão alguns aspectos importantes sobre o que essa classe faz:

Layout e Componentes Visuais: A classe `Cadastro` herda características de `javax.swing.JFrame`, indicando que representa uma janela gráfica. Ela define o layout e os componentes visuais, incluindo rótulos, campos de texto, botões e uma tabela para exibição de dados.

Inicialização e Configuração: O método `initComponents` é acionado no construtor `Cadastro()`, sendo responsável por inicializar e configurar os elementos visuais da interface. Este método é chamado automaticamente quando a classe é instanciada.

Cadastro de Usuários: A interface permite o cadastro de novos usuários, coletando informações como nome, e-mail, CPF, senha e data de nascimento. O botão "Salvar" é associado ao método `jButton1ActionPerformed`, que atualmente ajusta o texto do botão e adiciona um novo ouvinte de ação. No entanto, o conteúdo específico do salvamento de dados precisa ser implementado.

Exclusão de Usuários: O botão "Excluir" está presente na interface, mas atualmente o método `jButton2ActionPerformed` não possui implementação. Essa funcionalidade pode ser desenvolvida para permitir a exclusão de usuários da tabela.

Retorno à Tela de Login: O botão "Voltar" é associado ao método `jButton3ActionPerformed`, que redireciona a interface para a tela de login (`Login`) e fecha a janela de cadastro.

Exibição em Tabela: A classe inclui uma tabela (`jTable1`) para mostrar dados cadastrados. Atualmente, a tabela está preenchida com dados fictícios, mas ela pode ser populada com informações reais dos usuários cadastrados.

Em resumo, a classe `Cadastro` proporciona uma interface para cadastrar novos usuários no sistema, incluindo a exibição e manipulação de dados por meio de componentes visuais.

TelaPrincipal.java:

Esta classe representa a tela principal do sistema. Aqui estão alguns pontos essenciais sobre o que esta classe faz:

Layout e Componentes Visuais: A classe `TelaPrincipal` é uma extensão de `javax.swing.JFrame`, indicando que representa uma janela gráfica. Ela define o layout e os componentes visuais, como rótulos, botões, painéis e um menu.

Inicialização e Configuração: O método `initComponents` é acionado no construtor `TelaPrincipal()`, sendo responsável por inicializar e configurar os elementos visuais da interface. Esse método é gerado automaticamente e não deve ser modificado.

Botões e Ações Associadas: Existem botões associados a diferentes ações, como abrir telas específicas para diferentes filmes e séries. Cada botão tem um método associado (por exemplo, `jButton1ActionPerformed`), onde a ação desejada é implementada. Essa implementação envolve criar e exibir uma nova instância de uma classe relacionada (por exemplo, `HomemFerro`, `JogosVorazes`, etc.) e fechar a janela atual.

Menu de Opções: Existe um menu de opções (`Opções`) no topo da tela, contendo dois itens: "Suporte" e "Sair". Cada item do menu tem um método associado (por exemplo, `jRadioButtonMenuItem1ActionPerformed` e `jRadioButtonMenuItem2ActionPerformed`), onde a ação correspondente é implementada. O item "Suporte" abre uma tela de suporte, e o item "Sair" retorna à tela de login.

Pesquisa: Existe um botão "Pesquisar" que, quando clicado, abre uma tela de pesquisa (`Pesquisa`). A implementação específica dessa tela não está fornecida no código fornecido, mas ela deve ser projetada para permitir que os usuários pesquisem conteúdo no sistema.

Destaques de Filmes e Séries: A tela exibe destaques para filmes e séries em dois painéis separados. Cada painel contém botões associados a diferentes títulos, permitindo aos usuários acessarem facilmente o conteúdo destacado.

Em resumo, a classe `TelaPrincipal` oferece uma interface gráfica para os usuários explorarem e interagirem com os destaques de filmes e séries no sistema.

Adição do Botão de Autenticação: Foi adicionado um botão chamado `jButtonAuthenticate` no painel de filmes e séries. Esse botão será usado para autenticar o usuário antes de acessar conteúdos considerados inadequados para menores.

Ação do Botão de Autenticação: Foi adicionada uma nova ação associada ao botão de autenticação (`jButtonAuthenticate`). Atualmente, esta ação exibe uma caixa de diálogo (`JOptionPane.showMessageDialog`) indicando que o conteúdo é inadequado para menores e requer autenticação por senha.

Explicação na Caixa de Diálogo: A mensagem na caixa de diálogo serve como um aviso explícito de que o conteúdo pode não ser apropriado para todas as idades e destaca a necessidade de autenticação para acessá-lo. Isso é uma medida de precaução para garantir que apenas usuários autorizados possam visualizar o conteúdo.

Autenticações:

Autenticacao.java: Nessa classe, foi adicionada uma tela de autenticação que solicita uma senha para desbloquear o acesso a determinado conteúdo. Abaixo estão algumas explicações sobre o código:

Campo de Senha: Há um campo de senha (`txtBloqueio`) onde o usuário deve inserir a senha para desbloquear o conteúdo.

Botões: Existem dois botões na tela:

Acessar: Verifica se a senha inserida é correta. Se sim, abre a tela `Tons50`, que contém o conteúdo bloqueado. Se não, exibe uma mensagem de aviso.

Voltar: Retorna à tela principal (`TelaPrincipal`).

Lógica de Autenticação: A lógica de autenticação é simples. Se a senha inserida (`new String(txtBloqueio.getPassword())`) for igual a "123", então o acesso é permitido, e a tela `Tons50` é exibida. Caso contrário, uma caixa de diálogo com a mensagem "Você não pode acessar esse conteúdo!" é exibida.

Fluxo Principal: O método `main` inicia a aplicação, exibindo a tela de autenticação (`Autenticacao`).

AutenticacaoSerie.java: Nesta classe, foi implementada uma tela de autenticação para desbloquear o acesso a um conteúdo específico de série. Aqui estão algumas explicações sobre o código:

Campo de Senha: Existe um campo de senha (`SerieLiberar`) onde o usuário deve inserir a senha para liberar o conteúdo da série.

Botões: Há dois botões na tela:

Acessar: Verifica se a senha inserida é correta. Se sim, abre a tela `SexLife`, que contém o conteúdo da série. Se não, exibe uma mensagem de aviso.

Voltar: Retorna à tela principal (`TelaPrincipal`).

Lógica de Autenticação: A lógica de autenticação é semelhante à anterior. Se a senha inserida (`new String(SerieLiberar.getPassword())`) for igual a "123", então o acesso é permitido, e a tela `SexLife` é exibida. Caso contrário, uma caixa de diálogo com a mensagem "Você não pode acessar esse conteúdo!" é exibida.

Fluxo Principal: O método `main` inicia a aplicação, exibindo a tela de autenticação para séries (`AutenticacaoSerie`).

Pesquisa.java:

Essa classe Java representa a tela de pesquisa no sistema. Aqui estão algumas informações relevantes sobre o que esta classe realiza:

Layout e Componentes Visuais: A classe `Pesquisa` é uma extensão de `javax.swing.JFrame`, indicando que representa uma janela gráfica. O layout inclui um campo de texto para inserir a pesquisa, um botão para iniciar a pesquisa, e um botão para voltar à tela principal.

Método de Pesquisa: O método `jButton1ActionPerformed` é acionado quando o botão "Procurar" é clicado. Ele verifica o conteúdo do campo de pesquisa (`pesquisar`) e, com base na entrada, decide qual tela relacionada abrir. Se a entrada corresponder a um filme ou série conhecido (por exemplo, "Homem de Ferro" ou "Jogos Vorazes"), a aplicação abrirá a tela correspondente. Se não houver correspondência, exibirá uma mensagem de aviso.

Voltar à Tela Principal: O botão "Voltar" (`jButton2`) permite que o usuário retorne à tela principal do sistema "MD FLIX".

Aviso de Filme Não Encontrado: Se a pesquisa não corresponder a nenhum filme ou série conhecido, será exibida uma caixa de diálogo (`JOptionPane`) informando ao usuário que o filme não foi encontrado.

Entrada e Comparação: A entrada do campo de pesquisa é convertida para maiúsculas (`pesquisar.getText().toUpperCase()`) antes da comparação, garantindo que a comparação não seja sensível a maiúsculas e minúsculas.

Encerramento e Abertura de Telas: Para abrir uma nova tela, a classe atual é fechada (`this.dispose()`) e a nova tela é instanciada e tornada visível.

Em resumo, a classe `Pesquisa` é responsável por fornecer uma interface para os usuários pesquisarem filmes e séries no sistema e navegar para as páginas correspondentes com base nos resultados da pesquisa.

Suporte.java:

Esta classe Java representa uma tela de suporte no sistema. Abaixo estão algumas informações relevantes sobre o que esta classe realiza:

Layout e Componentes Visuais: A classe `Suporte` é uma extensão de `javax.swing.JFrame`, indicando que representa uma janela gráfica. Ela possui campos para o assunto da mensagem (`jTextField1`), a mensagem em si (`jTextArea1`), a data de abertura (`jTextField2`), e botões para enviar a mensagem (`jButton1`) e voltar para a tela principal (`jButton2`).

Envio de Suporte: Quando o botão "Enviar" (`jButton1`) é clicado, provavelmente um método de envio de suporte seria acionado. No entanto, o código atual não implementa essa lógica. Pode ser necessário adicionar a lógica de envio de suporte no método correspondente.

Voltar para a Tela Principal: O botão "Voltar" (`jButton2`) está associado ao método `jButton2ActionPerformed`, que cria uma instância da classe `TelaPrincipal` e a torna visível, enquanto fecha a tela de suporte.

Informações de Suporte: Os campos de assunto, mensagem e data de abertura são fornecidos para que o usuário insira informações relevantes ao solicitar suporte.

Em resumo, a classe `Suporte` fornece uma interface para os usuários do "MD FLIX" enviarem solicitações de suporte. O usuário pode voltar para a tela principal após interagir com a tela de suporte.

Cursed.java:

Essa classe Java representa a interface gráfica para a série "Cursed - A Lenda do Lago" no sistema. Aqui estão algumas informações relevantes sobre o código:
Layout e Componentes Visuais: A classe `Cursed` é uma extensão de `javax.swing.JFrame`, indicando que representa uma janela gráfica. Ela exibe informações sobre a série, como título, imagem, ano de lançamento, gênero, número de temporadas, diretores e sinopse.
Informações da Série:

- **Título:** Cursed - A Lenda do Lago
- **Ano de Lançamento:** 2020
- **Gênero:** Fantasia, Aventura, Drama
- **Temporadas:** 1 temporada
- **Diretores:** Frank Miller, Tom Wheeler
- **Sinopse:** A história se passa em um mundo medieval reimaginado e segue Nimue, uma jovem com um misterioso dom, destinada a se tornar a Dama do Lago.

Componentes Visuais:

Botão "Assistir": Ao clicar neste botão (`jButton1`), o código instancia a classe `CursedTemp` (possivelmente uma classe que representa a reprodução de episódios) e a torna visível.
Botão "Voltar": Ao clicar neste botão (`jButton2`), o código instancia a classe `TelaPrincipal` e a torna visível, enquanto fecha a janela atual.
CheckBox "Favoritar": Permite ao usuário marcar a série como favorita.
Imagem da Série: A imagem da série é exibida usando um componente `jLabel2` com um ícone.

CursedTemp.java:

Esta classe Java, chamada `CursedTemp`, representa a interface gráfica para os episódios da série "Cursed - A Lenda do Lago". Aqui estão algumas informações relevantes sobre o código:

Layout e Componentes Visuais: A classe `CursedTemp` é uma extensão de `javax.swing.JFrame`, indicando que representa uma janela gráfica.
A janela exibe uma lista de episódios da "Temporada 1" da série "Cursed - A Lenda do Lago".

Episódios da Série: A lista de episódios é exibida em um painel (`jPanel2`), com cada episódio representado por um botão. Os episódios são numerados de 1 a 10.

Botão "Voltar": Ao clicar no botão "Voltar" (`jButton11`), o código instancia a classe `Cursed` e a torna visível, enquanto fecha a janela atual.

Título da Janela: O título da janela é "CURSED - A LENDA DO LAGO" (`jLabel1`).

HomemFerro.java:

Esta classe Java, chamada `HomemFerro`, representa a interface gráfica para o filme "Homem de Ferro". Aqui estão algumas informações relevantes sobre o código:

Layout e Componentes Visuais: A classe `HomemFerro` é uma extensão de `javax.swing.JFrame`, indicando que representa uma janela gráfica.
A janela exibe uma imagem (`jLabel1`) correspondente ao filme "Homem de Ferro". Botões e rótulos são utilizados para mostrar informações sobre o filme, como título, ano de lançamento, diretor, gênero e sinopse.

Botões e Ações Associadas: Um botão "Assistir" (`jButton1`) é apresentado, mas a ação associada ainda não foi implementada (`jButton1ActionPerformed`).
Um botão "Voltar" (`jButton2`) é fornecido, e ao ser clicado, ele instancia a classe `TelaPrincipal` e a torna visível, enquanto fecha a janela atual.

Favoritar: Um `JCheckBox` (`jCheckBox1`) permite ao usuário marcar ou desmarcar a opção de "Favoritar".

JogosVorazes.java:

Esta classe Java, chamada `JogosVorazes`, representa a interface gráfica para o filme "Jogos Vorazes". Aqui estão algumas informações relevantes sobre o código:

Layout e Componentes Visuais: A classe `JogosVorazes` é uma extensão de `javax.swing.JFrame`, indicando que representa uma janela gráfica. A janela exibe uma imagem (`jLabel2`) correspondente ao filme "Jogos Vorazes". Botões e rótulos são utilizados para mostrar informações sobre o filme, como título, ano de lançamento, diretor, gênero, sinopse e duração.

Botões e Ações Associadas: Um botão "Assistir" (`jButton1`) é apresentado, mas a ação associada ainda não foi implementada (`jButton1ActionPerformed`). Um botão "Voltar" (`jButton2`) é fornecido, e ao ser clicado, ele instancia a classe `TelaPrincipal` e a torna visível, enquanto fecha a janela atual.

Informações do Filme: Rótulos (`jLabel3` a `jLabel9`) são utilizados para exibir informações como título, ano de lançamento, diretor, gênero, sinopse e duração do filme "Jogos Vorazes".

Favoritar: Um `JCheckBox` (`jCheckBox1`) permite ao usuário marcar ou desmarcar a opção de "Favoritar". No entanto, não há lógica associada a essa opção no código fornecido.

SexLife.java:

A classe Java chamada `SexLife` representa a interface gráfica para a série de TV "SexLife". Abaixo estão algumas informações importantes sobre o código:
A classe `SexLife` é uma extensão de `javax.swing.JFrame`, indicando que ela representa uma janela gráfica.
A janela exibe uma imagem (`jLabel2`) correspondente à série "SexLife". Botões e rótulos são utilizados para mostrar informações sobre a série, como título, ano de lançamento, diretor, gênero, número de temporadas, sinopse, etc.
Botões e Ações Associadas: Um botão "Assistir" (`jButton1`) é apresentado. A ação associada a esse botão (`jButton1ActionPerformed`) instância e torna visível uma nova janela da classe `SexTemp`, que contém os episódios da série referida.
Um botão "Voltar" (`jButton3`) é fornecido. Ao ser clicado, ele instancia a classe `TelaPrincipal` e a torna visível, enquanto fecha a janela atual.
Um `JCheckBox` (`jCheckBox1`) permite ao usuário marcar ou desmarcar a opção de "Favoritar".
Fluxo Principal: O método `main` inicia a aplicação, exibindo a janela dos episódios da série (`SexLife`).

SexTemp.java:

Esta classe Java representa uma interface gráfica para os episódios de uma série chamada "Sex Life". Aqui estão algumas explicações sobre o código:

Painéis e Botões: Existem dois painéis, `jPanel2` para a Temporada 2 e `jPanel3` para a Temporada 1. Cada painel contém botões representando os episódios de cada temporada (por exemplo, `jButton1` a `jButton8` para a Temporada 1 e `jButton9` a `jButton14` para a Temporada 2).

Ação do Botão "Voltar": Existe um botão chamado `jButton15` com a ação de voltar. Quando pressionado, ele fecha a janela atual (`SexTemp`) e abre a tela principal (`SexLife`).

Fluxo Principal: O método `main` inicia a aplicação, exibindo a janela dos episódios da série (`SexTemp`).

The100.java:

Esta classe Java representa a tela de detalhes de uma série chamada "The 100". Vamos analisar algumas partes específicas do código:

Labels e Imagem: Existem várias labels (`JLabel`) que exibem informações sobre a série, como título, imagem, ano de estreia, número de temporadas, diretores, sinopse, etc. A imagem é exibida usando um `JLabel` com um ícone (`ImageIcon`) carregado de um recurso no caminho "/images/the 100.jpeg".

Botões: Há dois botões, `jButton2` e `jButton9`, que têm ações associadas. `jButton2` é para "Assistir" e, quando clicado, abre uma nova janela (`The100Temp`) e fecha a janela atual (`The100`). `jButton9` é para "Voltar" e, quando clicado, retorna à tela principal (`TelaPrincipal`).

Checkbox: Existe uma `JCheckBox` chamada `jCheckBox1` com a opção "Favoritar". Os usuários podem marcar ou desmarcar esta opção.

Ação dos Botões: As ações dos botões estão implementadas nos métodos `jButton2ActionPerformed` e `jButton9ActionPerformed`. No primeiro, uma nova janela é aberta; no segundo, a tela principal é exibida.

Fluxo Principal: O método `main` inicia a aplicação, exibindo a janela de detalhes da série (`The100`).

The100Temp.java:

Esta classe Java representa uma interface gráfica para os episódios de uma série chamada "The 100". Aqui estão algumas explicações sobre o código:

Estrutura Geral: O código está dividido em vários métodos, onde cada método realiza uma função específica. A estrutura geral inclui:

- main **Método Principal:**
    - Configura o look and feel do Swing para "Nimbus".
    - Cria uma instância da classe `The100Temp` (que representa a janela principal) e a torna visível.

Organização da Interface Gráfica:

- Há vários painéis (`jPanel2`, `jPanel3`, ..., `jPanel8`) para cada temporada, organizados horizontalmente na interface principal (`jPanel1`).
- Cada painel contém um rótulo (`jLabelX`) indicando a temporada e uma série de botões representando os episódios dessa temporada.
- O botão "Voltar" (`jButton101`) está localizado no final da interface.

Temporadas (`jPanel2`, `jPanel3`, ..., `jPanel8`):

Cada temporada é representada por um painel (`JPanel`). Cada um desses painéis contém um rótulo indicando a temporada e botões representando episódios. **Configuração dos Painéis de Temporadas (**jPanel2**,** jPanel3**, ...,** jPanel8**):**

- Define a cor de fundo de cada painel.
- Configura um rótulo indicando o número da temporada.
- Adiciona botões representando episódios para cada temporada.

Botão "Voltar" (`jButton101`):
jButton101ActionPerformed **- Ação do Botão "Voltar":**

- Cria uma nova instância da classe `The100`.
- Torna a nova instância visível.
- Fecha a janela atual (`this.dispose()`).

Em todas os painéis de temporadas, os botões são organizados verticalmente usando o layout `GroupLayout`.

Tons50.java:

Esta classe representa a tela de detalhes do filme "Cinquenta Tons de Cinza". Vamos analisar algumas partes específicas do código:

Labels e Imagem: Existem várias labels (`JLabel`) que exibem informações sobre o filme, como título, imagem, ano de lançamento, diretor, gênero, sinopse, duração, etc. A imagem é exibida usando um `JLabel` com um ícone (`ImageIcon`) carregado de um recurso no caminho "/images/50Tons.jpeg".

Botões: Há dois botões, `jButton1` e `jButton2`, que têm ações associadas. `jButton1` é para "Assistir". `jButton2` é para "Voltar" e, quando clicado, retorna à tela principal (`TelaPrincipal`).

Checkbox: Existe uma `JCheckBox` chamada `jCheckBox1` com a opção "Favoritar". Os usuários podem marcar ou desmarcar esta opção.

Ação dos Botões: As ações dos botões estão implementadas nos métodos `jButton1ActionPerformed` e `jButton2ActionPerformed`. O primeiro pode ser implementado para abrir uma nova janela ou executar a ação de assistir. O segundo retorna à tela principal.

Fluxo Principal: O método `main` inicia a aplicação, exibindo a janela de detalhes do filme (`Tons50`).

Conexão com o Banco de dados MySQL

Estou enfrentando dificuldades na implementação da funcionalidade de conexão com o MySQL em meu código. A intenção é estabelecer uma comunicação eficaz com o banco de dados, mas um obstáculo surgiu durante o processo de configuração do driver JDBC. Busquei auxílio em recursos online, como tutoriais no YouTube, como este `https://www.youtube.com/watch?v=afezyKRkin8&t=394s`.

Embora tenha seguido os passos indicados, encontrei uma complicação específica na etapa de integração do conector do MySQL. Mesmo após adicionar o conector ao projeto, persistem dúvidas quanto à sua localização adequada, considerando tentativas na pasta 'Libraries' e no arquivo 'pom.xml'. A incerteza sobre a colocação correta do conector pode ser a causa da falha na conexão.

Caso todos esses passos sejam concluídos com êxito, a expectativa é que a conexão com o banco de dados seja estabelecida, possibilitando uma integração fluida entre o ambiente de desenvolvimento (IDE) e o MySQL.

Este é o código usado:

Parâmetros de Conexão:

- A classe possui constantes para URL, usuário e senha do banco de dados.

private static final String URL = "jdbc:mysql://seu_servidor:3306/streeming";

private static final String USUARIO = "root";

private static final String SENHA = "";

Singleton:

- A classe implementa o padrão Singleton, garantindo que apenas uma instância da conexão seja criada durante a execução do programa. Isso é feito mantendo uma única instância da conexão como uma variável estática (`conexao`) e fornecendo um método estático (`obterConexao`) para obter essa instância.

Fechar Conexão:

- Há um método para fechar a conexão. Isso é importante para liberar os recursos do banco de dados quando não são mais necessários.

Exceções:

- O código trata exceções relacionadas à carga do driver JDBC, à obtenção da conexão e ao fechamento da conexão.

Uso:

- Para obter uma instância da conexão, você pode chamar `ConexaoBancoDeDados.obterConexao()`. Certifique-se de fechar a conexão quando ela não for mais necessária usando `ConexaoBancoDeDados.fecharConexao()`.

Implementações:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import javax.swing.JoptionPane;
import javax.swing.*;
import java.awt.event.ActionEvent;
```