Programação Orientada à Objetos – Linguagens de Programação

TAREFA FINAL – MINI SIMULADOR DE REDE SOCIAL

RA	NOME
12121186	Davidson Adriano Faria Gonçalves
12118931	Jean Lucas Dias Bezerra
12120752	Pedro Duarte Oliveira
12118331	Pedro Augusto Andrade Silva
12108932	Thiago Ferreira Domingos

Data de Entrega: 23/11/2023

INTRODUÇÃO

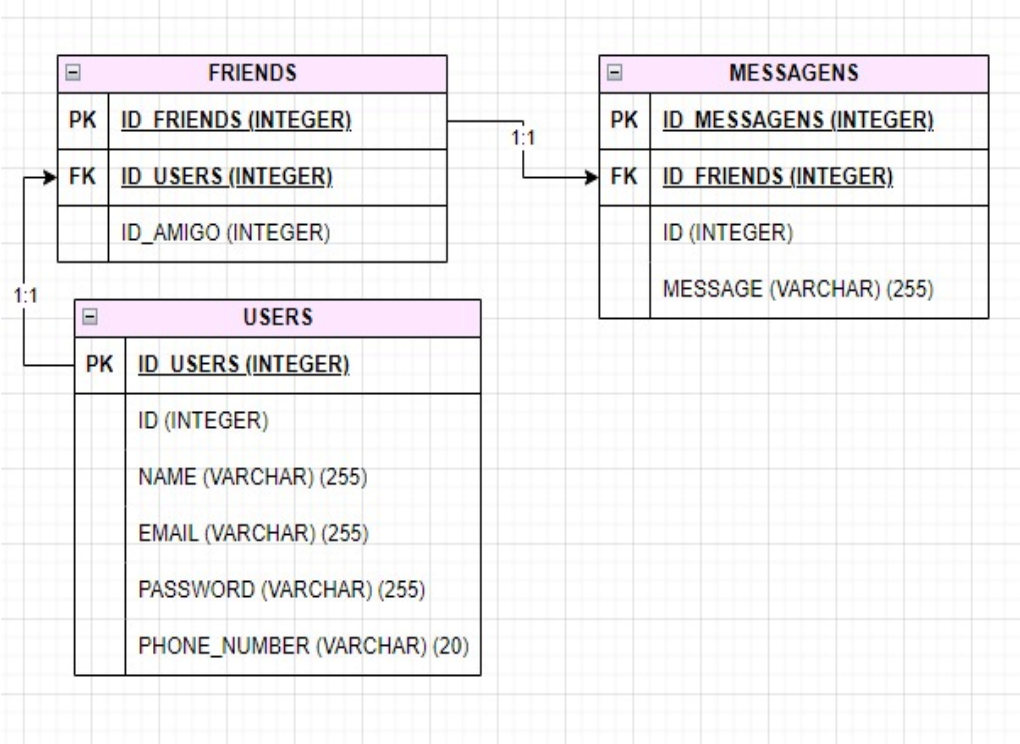
Solução para o Problema

O projeto consiste na implementação de um Mini Simulador de Rede Social em Java, com funcionalidades de cadastro de usuários, gerenciamento de amizades e troca de mensagens. A estrutura do sistema é composta por três principais entidades: **Usuario**, **Mensagem** e **RedeSocial**.

Estrutura de Dados para Número Indefinido de Entradas

Para lidar com um número variável de usuários, a classe User utiliza um conjunto de variáveis privadas para armazenar os dados individuais de cada usuário, como nome, e-mail, senha e número de telefone. Através desses campos, é possível armazenar e manipular as informações dos usuários de forma flexível, permitindo a adição, remoção e consulta de usuários conforme necessário no contexto da aplicação.

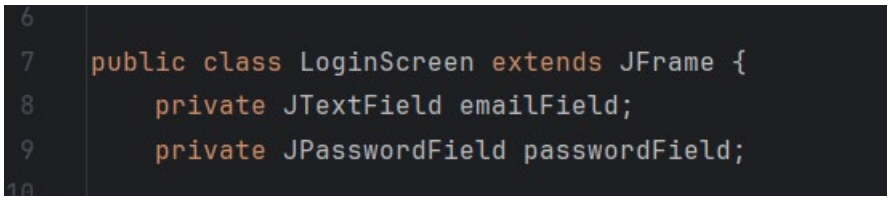
A estrutura de dados necessária para o código do projeto envolve o gerenciamento de usuários, amigos e mensagens. Aqui está uma possível estrutura de dados. Esta estrutura de dados inclui a classe **Friends** para representar cada usuário com suas informações, lista de amigos e mensagens. A classe **User** gerencia os usuários, fornecendo métodos para cadastrar usuários, fazer login, incluir amigos, consultar amigos, excluir amigos e enviar mensagens.


FRIENDS
PK
ID_FRIENDS (INTEGER)
FK
ID_USERS (INTEGER)
ID_AMIGO (INTEGER)
1:1
MESSAGENS
PK
ID_MESSAGENS (INTEGER)
FK
ID_FRIENDS (INTEGER)
ID (INTEGER)
MESSAGE (VARCHAR) (255)
1:1
USERS
PK
ID_USERS (INTEGER)
ID (INTEGER)
NAME (VARCHAR) (255)
EMAIL (VARCHAR) (255)
PASSWORD (VARCHAR) (255)
PHONE_NUMBER (VARCHAR) (20)

DESENVOLVIMENTO

HERANÇA

Não há classes abstratas no código, mas o conceito de herança está presente implicitamente em Java Swing. Por exemplo, a classe **JFrame** é uma subclasse de **java.awt.Frame**. A classe **JTextField** é uma subclasse de **javax.swing.text.JTextComponent**. Essa herança permite que as classes derivadas herdem comportamentos e propriedades da classe base.



POLIMORFISMO

O polimorfismo é observado no código ao tratar diferentes tipos de componentes Swing da mesma maneira. Por exemplo, JButton, JTextField, JLabel são todos manipulados como Componentes no layout, independentemente de suas implementações específicas.

```
28      private void placeComponents(JPanel panel) {
29          panel.setLayout(null);
30
31          JLabel nameLabel = new JLabel("Nome:");
32          nameLabel.setBounds(10, 20, 80, 25);
33          panel.add(nameLabel);
34
35          nameField = new JTextField(20);
36          nameField.setBounds(100, 20, 165, 25);
37          panel.add(nameField);
38
39          JLabel emailLabel = new JLabel("Email:");
40          emailLabel.setBounds(10, 50, 80, 25);
41          panel.add(emailLabel);
42
43          emailField = new JTextField(20);
44          emailField.setBounds(100, 50, 165, 25);
45          panel.add(emailField);
46
47          JLabel passwordLabel = new JLabel("Senha:");
48          passwordLabel.setBounds(10, 80, 80, 25);
49          panel.add(passwordLabel);
50
51          passwordField = new JPasswordField(20);
52          passwordField.setBounds(100, 80, 165, 25);
53          panel.add(passwordField);
```

PADRÕES DE PROJETO

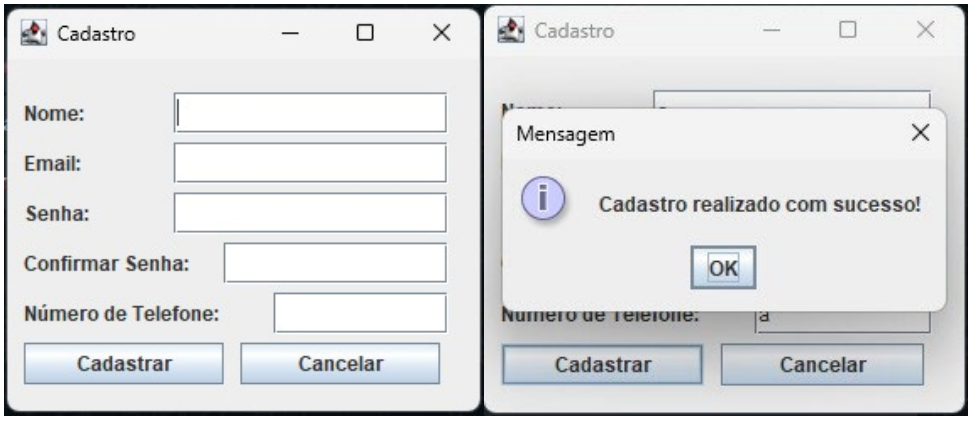
O código segue o padrão de projeto Model-View-Controller (MVC) implicitamente. A classe **RegistrationScreen** atua como a "View", interagindo com o usuário, enquanto a classe **User** pode ser considerada como o "Model", gerenciando os dados e a lógica de negócios. A interação entre a "View" e o "Model" é facilitada pelos eventos Swing e tratadores de eventos.

Observação: O padrão MVC não é estritamente seguido, mas há uma separação de preocupações entre a lógica da interface gráfica e as operações de rede social.

Além disso, o código faz uso de um padrão de projeto chamado "Observer" ao adicionar ouvintes (ActionListener) aos botões para observar eventos de ação.

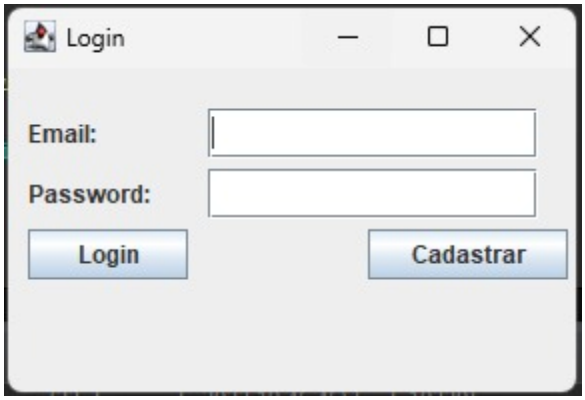
CADASTRO DE USUÁRIO

Os usuários podem ser cadastrados fornecendo um nome, um e-mail e uma senha. O cadastro é realizado clicando no botão "Cadastrar Usuário".



Login

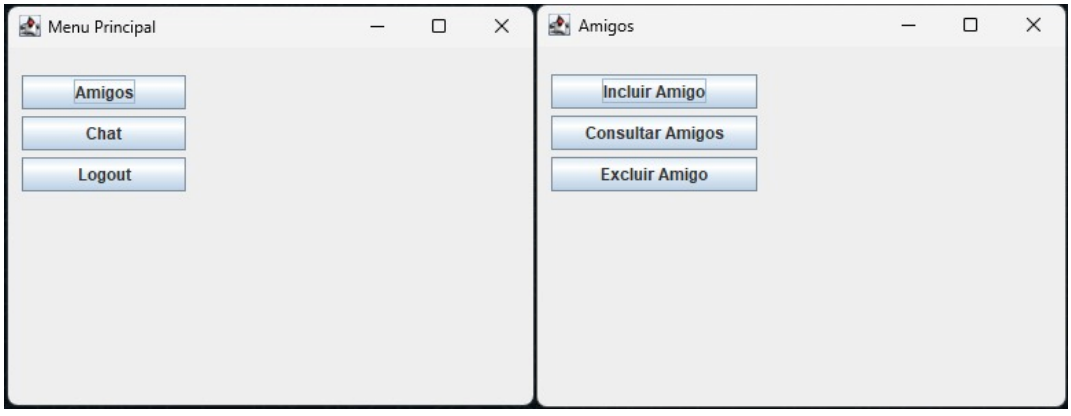
Os usuários podem fazer login no sistema inserindo seu e-mail e senha. O login é realizado clicando no botão "Fazer Login".



Gerenciamento de Amigos

Os usuários podem adicionar amigos escolhendo um amigo da lista suspensa e clicando no botão "Adicionar Amigo". A lista de amigos pode ser consultada clicando no botão "Consultar Amigos".

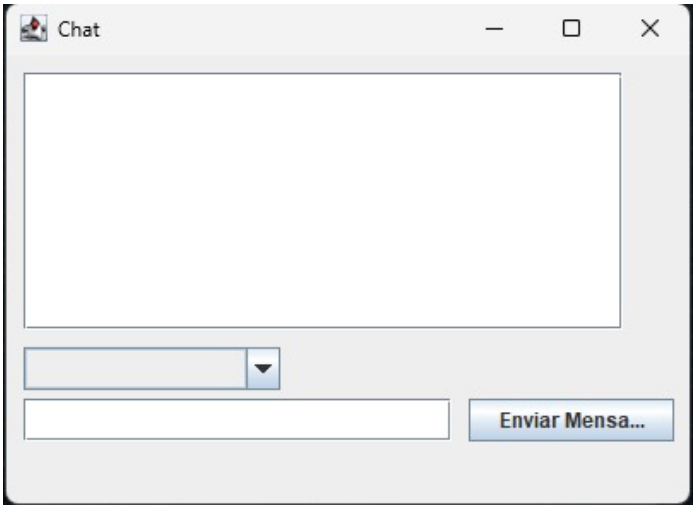
Os usuários podem excluir amigos escolhendo um amigo da lista suspensa e clicando no botão "Excluir Amigo".



Envio de Mensagens

Os usuários podem enviar mensagens para amigos inserindo a mensagem desejada e clicando no botão "Enviar Mensagem".

As mensagens são exibidas em uma área de texto na interface.



INTERFACE GRÁFICA

A interface gráfica foi construída usando o framework Swing, fornecendo uma experiência interativa para o usuário.

O código usa uma estrutura de dados simples com a classe **Friend** para representar cada usuário, mantendo informações como nome, e-mail, senha, lista de amigos e mensagens. A classe **User** gerencia os usuários e fornece métodos para realizar as operações mencionadas acima. A interface gráfica foi construída usando componentes Swing para criar uma aplicação visual.

CONCLUSÃO

Durante o desenvolvimento do programa em Java para o Mini Simulador de Rede Social, enfrentamos uma série de desafios que se revelaram oportunidades valiosas de aprendizado e crescimento em nossas habilidades de programação.

Desafios Superados:

Um dos maiores obstáculos encontrados foi a implementação da conexão com o banco de dados. Apesar dos esforços em compreender e integrar o programa com a persistência de dados, nos deparamos com obstáculos que exigiram um estudo mais profundo e prática dedicada para superação. Esta etapa destacou a complexidade e a importância do gerenciamento de dados em um sistema, permitindo uma compreensão mais profunda dos desafios associados à manipulação de informações persistentes.

Aprendizado e Conquistas:

Apesar das dificuldades, o processo de desenvolvimento nos proporcionou um aprendizado significativo em programação Java. Desde a estruturação do código até a implementação de funcionalidades complexas, a prática nos permitiu consolidar conhecimentos prévios e adquirir novas técnicas e boas práticas na construção de sistemas.

O sucesso desse projeto se manifestou no amadurecimento das nossas habilidades em Java, na compreensão mais sólida dos princípios de orientação a objetos e na capacidade de desenvolver soluções para desafios complexos. Além do avanço técnico, essa experiência fortaleceu nossa resiliência e determinação para enfrentar obstáculos em busca de nossos objetivos.