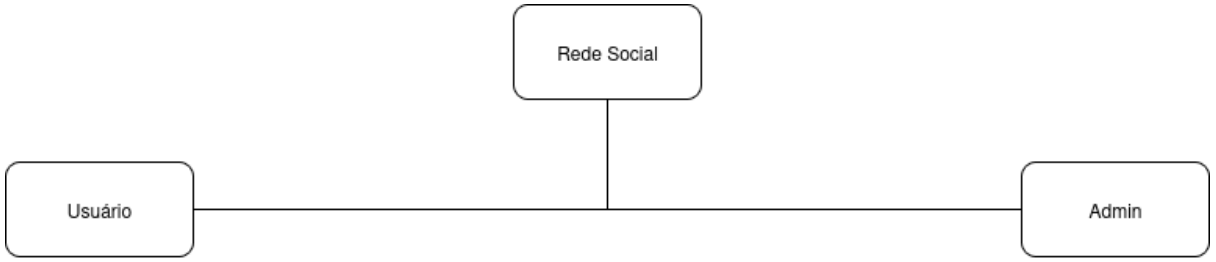
# Projeto Final Engenharia de Software
## Rede Social

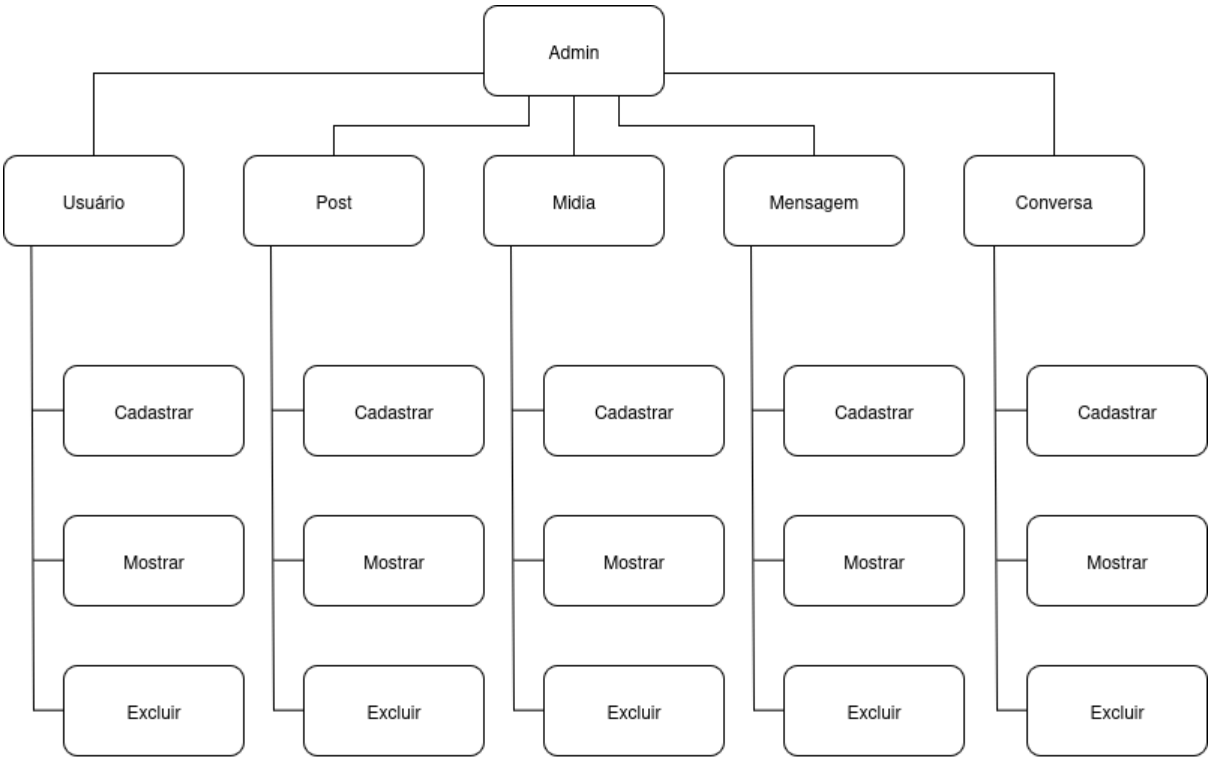
**Equipe:** Arthur Lodetti Gonçalves e Gustavo Piacentini da Silva
**Matéria:** Engenharia de Software
**Professora:** Rebeca Schroeder Freitas
**Data:** 07/07/2025

1) **Descrição do Problema:** O objetivo do trabalho é simular um sistema de Rede Social que busca realizar operações comuns como seguir, publicar posts, interação com outros usuários, gerando um ambiente de transmissão de conteúdo e comunicação.
   **Stakeholders:**
   - Usuários da Rede Social
   - Equipe de Desenvolvedores
   - Clientes que requisitaram o software
   - Administradores do Sistema

**Escopo do software:**




**Para Admin:**



**Para Usuário:**

2) **Requisitos do Software:**

**Requisitos Funcionais** que o sistema deve permitir:

- Login/Cadastro do Usuário
- Publicação/Remoção e Visualização de seus posts
- Curtidas e comentários em posts
- Seguir outro usuário
- Visualização do feed
- Remoção da conta
- Mostrar as conversas
- Criar/Remover uma conversa
- Acessar uma conversa
- Mostrar mensagens da conversa
- Enviar mensagem
- Compartilhar posts ou enviar mídia como forma de mensagem
- Remover a sua mensagem da conversa
- Adicionar um usuário à conversa

**Requisitos Não Funcionais** que o sistema deve permitir:

- Capacidade de suportar 10000 usuários simultaneamente
- Disponibilidade a todo momento
- Dados do usuário de acordo com a LGPD
- Tempo de resposta para operações curtas (like, comentário, seguir) inferior a 1 segundo
- Interface intuitiva, facilitando a conexão de símbolos com operações (coração = like)
- Design responsivo, adaptando-se a diferentes dispositivos e tamanhos de tela

3) **Estimativa de duração do projeto completo:** Usando Cocomo ou outro método - é preciso mostrar e descrever os passos).

| Elementos | Funções | Campos | Entidade | Complexidade | Peso |
|---|---|---|---|---|---|
| **EE** | login | 1 | 1 | baixa | 3 |
| | cadastraUsuario | 1 | 1 | baixa | 3 |
| | segueUsuario | 1 | 1 | baixa | 3 |
| | curtePost | 1 | 2 | baixa | 3 |
| | comentaPost | 1 | 2 | baixa | 3 |
| | publicaMidia | 1 | 1 | baixa | 3 |
| | deleteMidia | 0 | 1 | baixa | 3 |
| | deleteUsuarioAdmin | 0 | 1 | baixa | 3 |

| | | | | | |
|---|---|---|---|---|---|
| | deleteUsuario | 1 | 1 | baixa | 3 |
| | publicaPost | 1 | 1 | baixa | 3 |
| | publicaPostUsuario | 2 | 1 | média | 4 |
| | deletePost | 0 | 1 | baixa | 3 |
| | criaConversa | 0 | 1 | baixa | 3 |
| | criaConversaUsuario | 1 | 2 | média | 4 |
| | mandaMensagem | 2 | 2 | média | 4 |
| | mandaMensagemAdmin | 0 | 2 | média | 4 |
| | removeMensagem | 0 | 1 | baixa | 3 |
| | adicionarUsuarioNaConversa | 1 | 2 | média | 4 |
| | removeConversa | 0 | 2 | média | 4 |
| | | | | | |
| **SE** | mostraPosts | 0 | 1 | baixa | 4 |
| | mostraPostsUsuario | 1 | 2 | média | 5 |
| | mostraUsuarios | 0 | 1 | baixa | 4 |
| | mostraMidias | 0 | 1 | baixa | 4 |
| | mostraMensagens | 0 | 1 | baixa | 4 |
| | mostraConversas | 0 | 1 | baixa | 4 |
| | mostraConversasUsuario | 1 | 2 | média | 5 |
| | subconsulta | 0 | 2 | média | 5 |
| | | | | | |
| **CE** | mostraConteudoDaConversa | 1 | 3 | média | 4 |
| | mostraMensagensUsuario | 2 | 2 | média | 4 |

—----------

**EE - Entradas Externas -> 19**

Usuario login(Sistema sistema)
void cadastraUsuario(Sistema sistema)
void segueUsuario(int id_seguidor)
void curtePost(int id_usuario)
void comentaPost(int id_usuario)
void publicaMidia(Sistema sistema)
void deleteMidia()
void deleteUsuarioAdmin()
void deleteUsuario(int id_usuario)
void publicaPost(Sistema sistema)
void publicaPostUsuario(Sistema sistema, int id_usuario)
void deletePost()
void criaConversa()
void criaConversaUsuario(int id_usuario)
void removeConversa()
void mandaMensagem(int id_usuario, int id_conversa)
void mandaMensagemAdmin()
void removeMensagem()
void adicionarUsuarioNaConversa(int id_conversa)

—----------

**SE - Saídas Externas -> 8**

void mostraPosts()
void mostraPostsUsuario(int id_usuario)
void mostraUsuarios()
void mostraMidias()
void mostraMensagens()
void mostraConversas()
void subconsulta()
void mostraConversasUsuario(int id_usuario)

—----------

**CE - Consulta Externa -> 2**

void mostraConteudoDaConversa(int id_conversa)
void mostraMensagensUsuario(int id_conversa, int id_usuario)

—----------

**ALI - Arquivos Lógicos Internos**

Usuário, Post, Mídia, Mensagem, Conversa, Curte, Comenta, Segue, Participa, Possui e Recebe : 11

—----------

**AIE - Arquivos de Interface Externos**

Nenhum

| Elemento\Complexidade | Baixa | Média | Alta |
|---|---|---|---|
| Entradas Externas (EE) | 3 | 4 | 6 |
| Saídas Externas (SE) | 4 | 5 | 7 |
| Consultas Externas (CE) | 3 | 4 | 6 |
| Arquivos Lógicos Internos (ALI) | 7 | 10 | 15 |
| Arquivos de Interface Externos (AIE) | 5 | 7 | 10 |

**Fonte:** Tabela retirada dos slides como base para os cálculos de pontos de função não ajustados.
PFNA = Somatório (Elemento x Peso)
PFNA (EE) = 13x3 + 6X4 = 39 + 24 = 63
PFNA (SE) = 5x4 + 3x5 = 35
PFNA (CE) = 2x4 = 8
PFNA (ALI) = 11x7 = 77
PFNA (AIE) = 0
PFNA (tot) = 63 + 35 + 8 + 77 + 0 = 183
1 PFNA = 53 LOC **(Java)**
Total = 183x53 = 9699/1000 = **9,699 KLOCS**

**Esforço** = 2.4x(9,699^1.05) = 36,8 -> 37 pessoas-mês
**Duração** = 2.5x (**Esforço**^0.38) = 9.8 -> 10 meses

4) **Diagrama de classes UML:**

- Padrão de projeto utilizado Singleton: garante que uma classe possua apenas uma instância e fornece um ponto de acesso global pelo método **getInstance()** que retorna a sua instância no **DAO**.

**UsuarioDAO:**

```
public static UsuarioDAO getInstance() throws ClassNotFoundException, SQLException {
    if(instance == null){
        instance = new UsuarioDAO();
    }
    return instance;
}
```

**Sistema:**

```
public Sistema(String senha) throws ClassNotFoundException, SQLException {
    Conexao.setSenha(senha);
    usuarioDAO = UsuarioDAO.getInstance();
    postDAO = PostDAO.getInstance();
    midiaDAO = MidiaDAO.getInstance();
    mensagemDAO = MensagemDAO.getInstance();
    conversaDAO = ConversaDAO.getInstance();
}
```

UML estará separado no link do github devido ao seu tamanho na tela.

5) **Testes Unitários:** os testes unitários realizados nessa aplicação até o momento foram em um total de 99 testes, englobando os métodos getters and setters da camada de **dados**, ou seja, **Usuário, Post, Mídia, Mensagem e Conversa**. Além disso, foram testadas todas as funções da camada de **negócio**, nosso **Sistema**.
Por fim, como este projeto está sendo feito junto com o da disciplina de Banco de Dados I, a camada de persistência que conecta com o nosso banco não será testada, pois identificamos que para realizar esses testes durante a execução da aplicação ficaria mais demorado, dificultando a visualização. Por isso, as funções da classe **Main** receberam apenas testes onde possuem algum tipo de verificação além da "simples" inserção de dados e chamada direta da função do **Sistema**. Por exemplo, na função **segueUsuário()**, há uma verificação que impede o usuário de seguir ele mesmo, ou digitar id's inválidos, como 0.
Funções de **menu** estão fora dos testes pois apenas exibem um menu para guiar o usuário na aplicação. Funções de **mostra** que apenas exibem as ocorrências da entidade já estão sendo testadas na camada de **negócio**, pois apenas chamam a função do **Sistema**.

**Link do github:** https://github.com/Groudon19/TrabalhoBAN-I