UNIVERSIDADE FEDERAL DE ALAGOAS

INSTITUTO DE COMPUTAÇÃO

GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

RAUL ALVES DO NASCIMENTO

RELATÓRIO DO PROJETO JACKUT

MILESTONE 2

MACEIÓ

2025

1. INTRODUÇÃO

Este relatório descreve o processo de análise, refatoração e extensão de um sistema legado de rede social, originalmente desenvolvido por outra equipe. O principal objetivo foi implementar os requisitos definidos pelas User Stories 5 a 9, respeitando a estrutura existente sempre que possível e aplicando melhorias arquiteturais que viabilizassem a manutenção e evolução contínua do código.

A atividade consistiu não apenas na adição de novas funcionalidades, mas também na reorganização do sistema de modo a distribuir melhor as responsabilidades entre as classes e reduzir o acoplamento excessivo observado na implementação original. Além disso, buscou-se empregar padrões de projeto e boas práticas de desenvolvimento, com foco na coesão, modularidade e legibilidade do código.

2. AVALIAÇÃO DO CÓDIGO INICIAL

2.1 Pontos positivos

- **Facade**: Cumpria bem o papel de interface de interação com o sistema. Centralizava chamadas de métodos, facilitando o uso externo da aplicação.
- **Sistema**: Era bem estruturado, com boa organização e lidava corretamente com a persistência de dados. Seus métodos apresentavam uma implementação robusta e clara.
- **Usuario**: Lidava bem com informações sensíveis, como senhas, mantendo-as protegidas. Incorporava corretamente os atributos necessários de um usuário, utilizando estruturas de dados apropriadas, como filas para armazenar recados e mensagens.

2.2 Pontos negativos

- **Facade**: Possuía uma lógica de atualização de sistema bastante complexa, com muitas verificações que dificultavam a realização de modificações futuras e tornavam o código rígido.
- **Sistema e Usuario**: Apresentavam sobrecarga de responsabilidades. O Sistema era responsável por lidar com todas as funcionalidades da plataforma, enquanto Usuario possuía atributos demais, o que comprometia a coesão das classes.
- **Main**: A forma como os testes eram realizados na classe Main tornava necessária a lógica complexa de atualização do sistema presente na classe Facade, contribuindo para o acoplamento excessivo entre as camadas.

3. REFATORAMENTO E DESENVOLVIMENTO REALIZADO

Buscou-se realizar o mínimo de refatoração necessário para permitir a implementação das funcionalidades propostas pelas User Stories 5 a 9. Além disso, houve uma limpeza pontual de código comentado e não utilizado, bem

como a correção de nomes de métodos e atributos com erros de grafia. O JavaDoc foi adicionado em locais pontuais onde se fez necessário.

Com o intuito de melhorar a organização e facilitar a manutenção, foi adotado o padrão de projeto **Service Layer**, separando as responsabilidades do sistema em diferentes camadas: domínio, serviço e interface externa. A classe Sistema passou a representar a camada de domínio, enquanto as novas classes Gerenciador e suas subclasses compõem a camada de serviços. A Facade permanece como interface externa com o usuário.

Estrutura criada:

- **Gerenciador (abstrata)**: Representa um gerenciador genérico com funcionalidades compartilhadas, como o envio de recados automáticos do sistema.
- **GerenciadorDeUsuarios**: Responsável pelos serviços relacionados ao gerenciamento de usuários.
- **GerenciadorDeComunidades**: Responsável pelos serviços relacionados às comunidades.
- **GerenciadorDeRelacionamentos**: Responsável pelos serviços de relacionamentos entre usuários, como amizades, paqueras, ídolos e inimizades.
- **Relacionamento**: Nova classe que representa os diversos tipos de vínculos entre usuários.
- **Texto**: Criada para representar recados e mensagens, permitindo associar remetente e conteúdo a cada recado. Antes, recados eram apenas Strings, agora são objetos Texto, tornando o sistema mais rico em funcionalidade e mais preparado para remoção segura de usuários (excluindo seus recados junto).
- **Perfil**: Nova classe que concentra os atributos do perfil do usuário. Antes esses dados estavam na própria classe Usuario, o que gerava acoplamento e sobrecarga. Getters e setters desses atributos foram movidos para Perfil.

Organização em pacotes:

- **exceptions**: Contém as exceções do sistema.
- **sistema**: Contém as classes Sistema, Facade e os gerenciadores de serviços.
- **usuario**: Contém Usuario, Perfil, Comunidade, Relacionamento e Texto.

User Stories:

- **US5 a US7**: Não exigiram alterações significativas, além da reorganização da Main e Facade, e da criação da classe Comunidade. A criação da classe Perfil ocorreu ainda antes da implementação do US7, visando preparar a estrutura para a divisão de responsabilidades.

- **US8**: Exigiu a implementação das funcionalidades de relacionamento, com a introdução da classe Relacionamento, permitindo a distinção entre amizade, paquera, ídolo e inimizade.
- **US9**: Motivou a criação da classe Texto, visto que, para permitir funcionalidades como exclusão de usuários e manutenção dos recados/mensagens, era necessário associar um remetente a cada recado. Isso também proporcionou maior controle e rastreabilidade sobre as mensagens no sistema.

4. CONCLUSÃO

Trabalhar na implementação das User Stories 5 a 9 a partir de um sistema previamente desenvolvido foi uma experiência relevante e próxima da realidade encontrada em projetos do mercado e da academia, onde é comum lidar com código feito por terceiros. Esse processo evidenciou a importância de um bom design e da aplicação de padrões de projeto para tornar o sistema mais compreensível e fácil de modificar. As mudanças aplicadas, especialmente com a introdução de uma separação mais clara entre camadas e responsabilidades, contribuíram para tornar o código mais organizado, reutilizável e preparado para futuras extensões.