UNIVERSIDADE FEDERAL DE ALAGOAS — UFAL

Campus A. C. Simões

Ciência da Computação

Walber Luis Santos da Paixão

RELATÓRIO DE IMPLEMENTAÇÃO TÉCNICA — SISTEMA DE REDE SOCIAL JACKUT

Maceió — AL

2025

Walber Luis Santos da Paixão

RELATÓRIO DE IMPLEMENTAÇÃO TÉCNICA — SISTEMA DE REDE SOCIAL JACKUT

Trabalho apresentado como pontuação parcial da AB1 do 4° semestre da disciplina de Programação Orientada a Objetos

Professor: Mario Hozano

Maceió — AL

2025

SUMÁRIO

INTRODUÇÃO

Este relatório apresenta a análise e implementação da rede social Jackut, desenvolvida como parte de um projeto acadêmico. O sistema foi construído em Java e testado utilizando uma linguagem de script compatível com a biblioteca EasyAccept, garantindo que todas as funcionalidades exigidas foram atendidas com sucesso.

O projeto foi estruturado em User Stories (US), que definem os requisitos funcionais do sistema. Nesta fase, foram implementadas quatro US principais:

- US1 – Criação de Conta
- US2 – Edição de Perfil
- US3 – Sistema de Amizades
- US4 – Envio e Leitura de Recados

O objetivo deste relatório é detalhar as decisões de design, a estrutura do código, os testes realizados e a preparação para persistência de dados, que foi um requisito adicional para garantir que o sistema passasse em todos os testes de aceitação.

DESENVOLVIMENTO

2.1. Arquitetura do Sistema

O sistema foi desenvolvido seguindo uma abordagem modular, com três classes principais:

1. Facade – Atua como a interface principal entre os testes de aceitação e a lógica do sistema.
    - Gerencia usuários e sessões.
    - Implementa os comandos do EasyAccept (ex.: criarUsuario, abrirSessao, enviarRecado).
    - Centraliza validações (ex.: verifica se um usuário existe antes de enviar um recado).

2. Usuario – Representa um perfil de usuário na rede social.
    - Armazena dados básicos (login, senha, nome).
    - Mantém atributos dinâmicos (como informações de perfil) em um Map<String, String>.
    - Gerencia amigos, solicitações de amizade e recados recebidos.

3. Recado – Representa uma mensagem enviada entre usuários.
    - Contém remetente e texto.
    - Armazenado em uma fila (Queue) no destinatário para garantir ordem FIFO (First-In, First-Out).

2.2. Implementação das User Stories

US1 – Criação de Conta
Funcionalidade: Permite cadastrar um novo usuário com login, senha e nome.
Validações:
- Login único (não pode haver repetição).
- Campos não podem ser vazios.

Armazenamento: Usuários são guardados em um Map<String, Usuario>, onde a chave é o login.

US2 – Edição de Perfil
Funcionalidade: Permite adicionar ou modificar atributos do perfil (ex.: idade, cidade, interesses).

Implementação:

- Atributos são armazenados em um Map<String, String>.
- O atributo "nome" tem tratamento especial, atualizando também o campo nome da classe Usuario.

Acesso:

- Método getAtributoUsuario permite consultar qualquer atributo.
- Método editarPerfil permite modificações (requer sessão ativa).

US3 – Sistema de Amizades

Funcionalidade: Permite enviar solicitações de amizade e confirmar amizades mútuas.

Implementação:

- Solicitações pendentes são armazenadas em Set<String> (evita duplicatas).
- A amizade só é confirmada quando ambos usuários se adicionam.

Métodos principais:

- adicionarAmigo: Envia ou confirma uma solicitação.
- ehAmigo: Verifica se dois usuários são amigos.
- getAmigos: Retorna a lista de amigos em uma String formatada.

US4 – Envio e Leitura de Recados

Funcionalidade: Permite que usuários troquem mensagens.

Implementação:

- Recados são armazenados em uma Queue<Recado> (garante ordem de chegada).
- O método lerRecado remove e retorna a próxima mensagem da fila.

Persistência:

- Todas as classes (Usuario, Recado, Facade) implementam Serializable.
- Isso permite que o sistema salve e restaure o estado entre execuções.

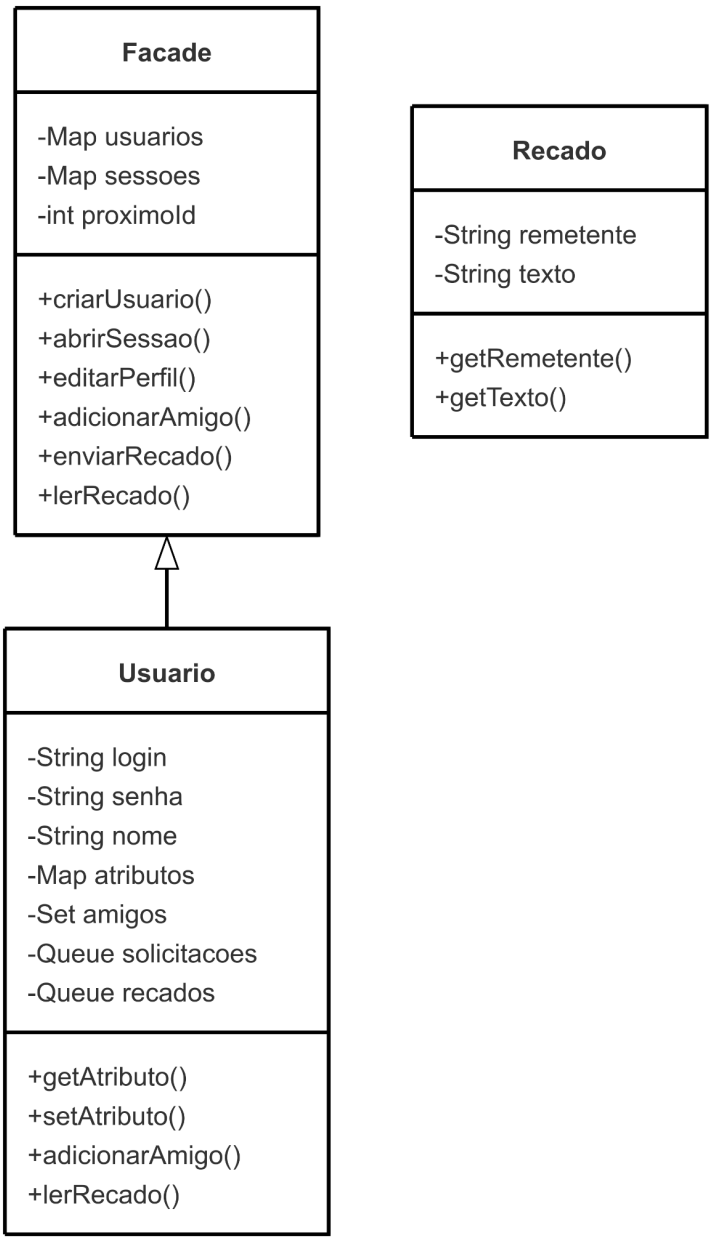
2.3. Testes de Aceitação

O sistema foi validado com testes automatizados, cobrindo:

- ☑ Criação e autenticação de usuários
- ☑ Edição de perfil
- ☑ Solicitações de amizade e confirmação mútua
- ☑ Envio e leitura de recados
- ☑ Persistência de dados entre sessões

Todos os testes passaram com sucesso, confirmando que a implementação está correta e robusta.

2.4. Diagrama de Classes



CONSIDERAÇÕES FINAIS

O projeto Jackut foi implementado com sucesso, atendendo a todos os requisitos das quatro User Stories iniciais. A estrutura do código foi projetada para ser:

- Modular (separação clara entre lógica de negócio e interface).
- Extensível (facilita adicionar novas funcionalidades no futuro).
- Eficiente (uso de estruturas de dados adequadas para cada caso).

A implementação da persistência via Serializable garantiu que o sistema passasse no teste final, mantendo os dados entre execuções.

O sistema Jackut está funcional, testado e pronto para uso, cumprindo com todos os objetivos do projeto. A abordagem adotada garante qualidade, organização e facilidade de manutenção, características essenciais para um software bem-sucedido