SISTEMA DE AUTENTICAÇÃO

UNIVERSIDADE ANHEMBI MORUMBI

CAMPUS PAULISTA

MODELOS, MÉTODOS E TÉCNICAS DA ENGENHARIA DE SOFTWARE

AUTORES: Carlos Oliveira Bonfim - 12524141589,

Leonardo Pereira Duarte - 12525179099,

Yan de Sousa Ferrezin - 12525179091,

Joaquim Guilherme Nunes Leal - 12524141595.

Repositório no Github:
https://github.com/Carlos-bonfim26/sistema_autenticador_deUsuario

Resumo

Este projeto consiste em um sistema de autenticação de usuários desenvolvido em Java, com persistência de dados utilizando o banco de dados MySQL. O sistema permite:

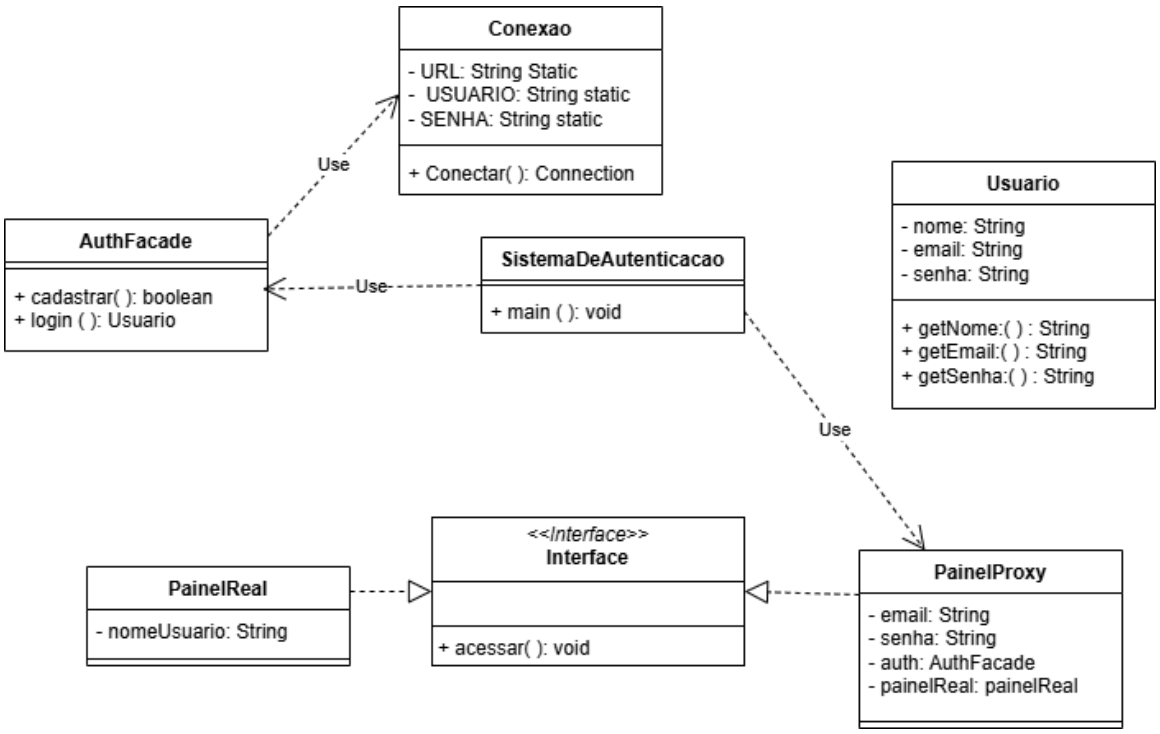
- Cadastrar novos usuários (nome, email e senha);
- Realizar login com verificação de email e senha;
- Liberar ou negar o acesso, de acordo com os dados fornecidos.

O diferencial do projeto é a aplicação de dois Design Patterns (Padrões de Projeto): Facade e Proxy, utilizados para organizar o sistema de forma escalável, segura e com boa separação de responsabilidades.

Justificativa da Escolha

- O padrão Facade foi escolhido para simplificar a interface de uso do sistema, evitando que o usuário interaja diretamente com várias classes e detalhes da lógica interna.
- O padrão Proxy foi utilizado para controlar o acesso ao painel principal do sistema, atuando como um intermediário que protege e valida as credenciais antes de liberar o acesso ao conteúdo real.

UML do Projeto


Conexao
- URL: String Static
- USUARIO: String static
- SENHA: String static
+ Conectar( ): Connection
Use
AuthFacade
+ cadastrar( ): boolean
+ login ( ): Usuario
Use
SistemaDeAutenticacao
+ main ( ): void
Usuario
- nome: String
- email: String
- senha: String
+ getNome:( ) : String
+ getEmail:( ) : String
+ getSenha:( ) : String
Use
PainelReal
- nomeUsuario: String
<<Interface>>
Interface
+ acessar( ): void
PainelProxy
- email: String
- senha: String
- auth: AuthFacade
- painelReal: painelReal

Descrição da Implementação

1. Facade (Fachada)

Classe principal: AuthFacade
Responsabilidades:
- Cadastrar usuários no banco de dados.
- Validar login de usuários.
- Ocultar toda a lógica de conexão e manipulação do banco de dados.

Relacionamentos:
- Usa a classe Conexão para se conectar ao banco.
- Utiliza PreparedStatement e Connection para manipulação dos dados.

Papel no sistema: centraliza as ações principais de autenticação e abstrai a complexidade do banco de dados para o usuário final.

Exemplo no código:

```
public boolean cadastrar(String nome, String email, String senha) {
    // comandos sql para inserção na tabela usuários
    String sql = "INSERT INTO usuarios (nome, email, senha) VALUES (?, ?, ?)";

    // tentando a conexão para ver se ela tem sucesso e
    try (Connection conn = Conexao.conectar();
    // o PreparedStatement é um método da connection que está recebendo os comandos e sql e que por meio do setString depois aplicará o valor dos
    parâmetros na iserção
    PreparedStatement stmt = conn.prepareStatement(sql)) {
        // colocando o valor dos parâmetros na iserção para cadastrar o usuário
        stmt.setString(parameterIndex:1, nome);
        stmt.setString(parameterIndex:2, email);
        stmt.setString(parameterIndex:3, senha);
        stmt.executeUpdate();
        return true;

    } catch (SQLException e) {
        System.out.println("Erro no cadastro: " + e.getMessage());
        return false;
    }
}
```

```
    public Usuario login( String email, String senha) throws SQLException {
       String sql = "SELECT * FROM usuarios WHERE email = ? AND senha = ?";

    try( Connection conn = Conexao.conectar();  PreparedStatement stmt = conn.prepareStatement(sql)){
        stmt.setString(parameterIndex:1, email);
        stmt.setString(parameterIndex:2, senha);
           // o executeQuery executa o sql do tipo select, o ResulSet trás um conjunto de resultados
        ResultSet rs = stmt.executeQuery();
           //o metodo next faz ele avançar o cursor para proxima linha do resultado
        if(rs.next()){
            String nome = rs.getString(columnLabel:"nome");
            return new Usuario(nome, email);
        }
    }catch(SQLException e){
        System.out.println("Erro no login: "+ e.getMessage());

    }
    return null;
    }
}
```

2. Proxy

Classe principal: PainelProxy
Interfaces envolvidas: Painel
Classes reais protegidas: PainelReal
Responsabilidades:
- Controlar o acesso ao painel após o login.
- Verificar as credenciais do usuário.
- Evitar o acesso direto ao painel sem autenticação.

Relacionamentos:
- Se comunique com a AuthFacade para autenticar o usuário.
- Caso a autenticação seja bem-sucedida, a instância utiliza o PainelReal.

Papel no sistema: funciona como um intermediário entre o usuário e o conteúdo protegido.
Garante que apenas usuários autenticados tenham acesso ao painel principal.

Exemplo no código:

```java
public class PainelProxy implements Painel{
    private String email;
    private String senha;
    private AuthFacade auth;
    private PainelReal painelReal;

    public PainelProxy(String email, String senha, AuthFacade auth) {
        this.email = email;
        this.senha = senha;
        this.auth = auth;
    }

    @Override
    public void acessar(){
        // tenta logar o usuário por via de seu usuário e senha
        try {
            Usuario usuario = auth.login(email, senha);
            if(usuario != null){
                painelReal = new PainelReal(usuario.getNome());
                painelReal.acessar();
            }else{
                System.out.println(x:"Acesso negado, Usuário não autenticado");
            }
        } catch (SQLException e) {
            // exceção caso tenha erros
            System.out.println("Erro na autenticação" + e.getMessage());
        }
    }
}
```

Como Utilizar o Sistema e Acessar os Dados

1. Clonando o Repositório

Para começar, abra o terminal e clone o repositório do projeto com o comando:

```
git clone https://github.com/seu-usuario/sistema_autenticador_deUsuario.git
```

Recomendado: esteja dentro de uma pasta específica onde deseja salvar o projeto.

2. Configurando o Banco de Dados (MySQL)

Se ainda não tiver o MySQL instalado, baixe e instale pelo site oficial ou use uma IDE como **MySQL Workbench.**

Depois, crie o banco e a tabela com os seguintes comandos (podem ser executados no terminal MySQL ou via Workbench):

```
-- criação banco de dados
create database sistema_usuarios;
-- usando o banco
use sistema_usuarios;
-- criando tabela de usuários
create table usuarios(
idUser int auto_increment primary key,
nome varchar(100),
email varchar(100) unique,
senha varchar(50)
);
```

Esses comandos também estão disponíveis no projeto, na pasta sql.

3. Configurando o Driver JDBC

Para que o Java consiga se conectar ao MySQL, é necessário o driver JDBC. Ele já está incluso na pasta lib do projeto, mas se precisar configurar manualmente, no netbeans você irá seguir os seguintes passos:

- No NetBeans, clique no botão direito no seu projeto (SistemaDeAutenticacao);

- Vá em “Propriedades”;

- No menu à esquerda, clique em “Bibliotecas”;

- Vá até a aba “Compiler” (Ou Compilação);

- Clique em “adicionar JAR/Pasta” (adicione um classpath);

- Navegue até a pasta “lib” e selecione o arquivo .jar;

- Clique em “OK” para adicionar.

4. Executando o Projeto

Agora com tudo configurado, você pode executar o projeto Java normalmente na IDE de sua preferência.

O programa vai oferecer as seguintes funcionalidades:

Cadastro de usuário: insere um novo usuário no banco.

Login: verifica se o email e a senha correspondem a um usuário existente.

Acesso autorizado ou negado: se o login for bem-sucedido, o sistema permitirá o acesso ao painel (classe PainelReal).

Observações Finais

- Dificuldades:

A principal dificuldade foi preparar o ambiente de desenvolvimento, muito por causa da difícil instalação do driver JDBC que foi fundamental para a conexão, e também da montagem do código para a conexão com o banco de dados.

- Aprendizados:

Este projeto permitiu aprofundar o entendimento sobre padrões de projeto na prática, por exemplo o Proxy nos ensinou de forma concreta o que é o conceito de encapsulamento e a importância da segurança de classes, e a oportunidade de ir mais a fundo no java e aumentar nosso nível aprendendo como fazer conexões com um banco de dados real.