■ BJJ School – Sistema Full Stack para Gestão de Escola de Jiu-Jitsu

Projeto da Disciplina – ENTREGA FINAL

Aluno: Domingos Caldas de Oliveira Junior

Professor: Leonardo Silva da Gloria

Disciplina: Projeto da Disciplina – Desenvolvimento Full Stack com React e Spring Boot [25E4_3]

Instituição: Instituto INFNET

Ano: 2026

Sumário

1. Resumo Executivo

O projeto **BJJ School** consiste em uma aplicação **Full Stack** desenvolvida para o gerenciamento de alunos de uma escola de jiu-jitsu.

A solução contempla funcionalidades completas de **cadastro, consulta, edição e exclusão (CRUD)**, integrando um frontend moderno em React a um backend robusto em Spring Boot, com **controle de acesso seguro baseado em autenticação e autorização**.

A aplicação foi totalmente containerizada utilizando **Docker e Docker Compose**, permitindo que todo o ambiente seja executado de forma automatizada com um único comando, garantindo portabilidade, reprodutibilidade e facilidade de avaliação pelo professor.

2. Objetivo do Projeto

O objetivo principal deste projeto é aplicar, de forma prática, os conceitos estudados ao longo da disciplina, incluindo:

- Desenvolvimento de interfaces interativas utilizando React
- Implementação de APIs REST com Spring Boot
- Integração segura entre frontend e backend
- Autenticação e autorização baseadas em padrões modernos
- Controle de acesso por perfis de usuário
- Organização e arquitetura de aplicações Full Stack
- Execução automatizada por meio de containers Docker

3. Descrição Funcional do Sistema

A aplicação permite:

- Visualizar a lista de alunos cadastrados
- Cadastrar novos alunos
- Editar informações de alunos existentes
- Excluir alunos (funcionalidade restrita a usuários administradores)
- Visualizar comunicados obtidos a partir de uma API externa

- Garantir controle de acesso conforme o perfil do usuário autenticado

Perfis de acesso

- **USER:** acesso apenas para visualização das informações
- **ADMIN:** acesso completo às funcionalidades de criação, edição e exclusão

4. Arquitetura da Aplicação

A solução foi estruturada em componentes independentes e integrados.

4.1 Frontend

- Desenvolvido em **React** utilizando **Vite**
- Interface construída com **Material UI**
- Comunicação com o backend por meio de **Axios**
- Gerenciamento de autenticação via **Keycloak**
- Proteção de rotas e componentes baseada em roles

4.2 Backend

- Desenvolvido em **Spring Boot**
- API REST para operações de CRUD
- Persistência de dados com **JPA/Hibernate**
- Banco de dados relacional **PostgreSQL**
- Segurança implementada com **Spring Security** e **OAuth2 Resource Server**

4.3 Autenticação e Segurança

- **Keycloak** como provedor de identidade
- Autenticação baseada em **JWT**
- Controle de permissões por roles
- Validação de acesso tanto no backend quanto no frontend

4.4 Infraestrutura

- Containerização com **Docker**
- Orquestração com **Docker Compose**
- Serviços executados:
- Frontend
- Backend
- Keycloak
- Banco de dados PostgreSQL

5. Tecnologias Utilizadas

- React
- Vite
- Material UI
- Axios
- React Router
- Spring Boot
- Spring Security
- OAuth2
- JWT
- Keycloak
- PostgreSQL
- Docker
- Docker Compose

6. Implementação dos Requisitos da Disciplina

6.1 CRUD Completo

O sistema implementa todas as operações de **Create, Read, Update e Delete** para a entidade **Aluno**, atendendo integralmente aos requisitos funcionais propostos.

6.2 Integração Frontend e Backend

O frontend consome os endpoints REST disponibilizados pelo backend, realizando operações de leitura e escrita de dados de forma segura.

6.3 Segurança e Controle de Acesso

- Login centralizado utilizando Keycloak
- Tokens JWT enviados automaticamente nas requisições autenticadas
- Validação de permissões no backend
- Ocultação de funcionalidades no frontend conforme o perfil do usuário

6.4 Tratamento de Erros e Experiência do Usuário

- Mensagens visuais para erros de autenticação e autorização
- Feedback ao usuário em operações inválidas
- Confirmação antes da execução de ações destrutivas, como exclusões

6.5 Execução Automatizada

Todo o ambiente pode ser iniciado com um único comando Docker, sem a necessidade de configurações manuais adicionais.

7. Instruções de Execução

Pré-requisitos

- Docker
- Docker Compose

7.1 Clonagem do Repositório

Repositório oficial no GitHub:

https://github.com/domingosjr/BJJ-PROJECT

```
git clone https://github.com/domingosjr/BJJ-PROJECT
cd BJJ-PROJECT
```

7.2 Execução com Docker

Na raiz do projeto, executar o comando:

```
docker compose up -d --build
```

7.3 Acessos

- Frontend: http://localhost:5173
- Backend: http://localhost:8090
- Keycloak: http://localhost:8081

8. Evidências de Funcionamento

- Tela de autenticação via Keycloak
- Tela de listagem de alunos
- Funcionalidades de cadastro, edição e exclusão
- Controle de acesso baseado em perfil
- Containers em execução via Docker Compose

9. Considerações Finais

O projeto **BJJ School** possibilitou a aplicação integrada dos conceitos abordados na disciplina, proporcionando uma visão prática do desenvolvimento de aplicações Full Stack modernas, seguras e escaláveis.

A adoção de Docker e Keycloak agregou valor técnico ao projeto, aproximando-o de cenários reais encontrados no mercado de trabalho.

10. Referências

- React – https://react.dev/
- Vite – https://vitejs.dev/
- Material UI – https://mui.com/
- Spring Boot – https://spring.io/projects/spring-boot
- Spring Security – https://spring.io/projects/spring-security
- Keycloak – https://www.keycloak.org/
- Docker – https://www.docker.com/
- Docker Compose – https://docs.docker.com/compose/
- PostgreSQL – https://www.postgresql.org/

- OpenAI ChatGPT (GPT-5.2) – Ferramenta utilizada para apoio técnico e geração de documentação, respeitando diretrizes acadêmicas de transparência e citação.

Projeto acadêmico desenvolvido no Instituto INFNET