

# ■ 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](#1-resumo-executivo)

2. [Objetivo do Projeto](#2-objetivo-do-projeto)

3. [Descrição Funcional do Sistema](#3-descrição-funcional-do-sistema)

4. [Arquitetura da Aplicação](#4-arquitetura-da-aplicação)

- 4.1 [Frontend](#41-frontend)
  - 4.2 [Backend](#42-backend)
  - 4.3 [Autenticação e Segurança](#43-autenticação-e-segurança)
  - 4.4 [Infraestrutura](#44-infraestrutura)
5. [Tecnologias Utilizadas](#5-tecnologias-utilizadas)
6. [Implementação dos Requisitos da Disciplina](#6-implementação-dos-requisitos-da-disciplina)
- 6.1 [CRUD Completo](#61-crud-completo)
  - 6.2 [Integração Frontend e Backend](#62-integração-frontend-e-backend)
  - 6.3 [Segurança e Controle de Acesso](#63-segurança-e-controle-de-acesso)
  - 6.4 [Tratamento de Erros e Experiência do Usuário](#64-tratamento-de-erros-e-experiência-do-usuário)
  - 6.5 [Execução Automatizada](#65-execução-automatizada)

7. [Instruções de Execução](#7-instruções-de-execução)

- 7.1 [Clonagem do Repositório](#71-clonagem-do-repositório)
- 7.2 [Execução com Docker](#72-execução-com-docker)
- 7.3 [Acessos](#73-acessos)

8. [Evidências de Funcionamento](#8-evidências-de-funcionamento)
9. [Critérios de Avaliação (Rubrica)](#9-critérios-de-avaliação-rubrica)
10. [Considerações Finais](#10-considerações-finais)
11. [Referências](#11-referências)
12. [Agradecimentos](#12-agradecimentos)

## 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/domigosjr/BJJ-PROJECT>

```
git clone https://github.com/domigosjr/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. Critérios de Avaliação (Rubrica)

O projeto **BJJ School** atende integralmente aos critérios de avaliação estabelecidos na rubrica da disciplina:

## 9.1 Configurar e operar um ambiente de desenvolvimento Spring

- Ambiente de desenvolvimento configurado e operante, integrando **Spring Boot**, **React** e persistência de dados.
- Consumo correto de endpoints REST pela aplicação frontend.
- Interface funcional, organizada e adequada para uso acadêmico.

## 9.2 Implementar funcionalidades CRUD

- Implementação completa de **Create, Read, Update e Delete** para a entidade **Aluno**.
- Operações integradas entre frontend e backend.
- Restrições aplicadas conforme perfis (ex.: exclusão e edição restritas a ADMIN).

## **9.3 Aprimorar interface e segurança de acesso aos dados**

- Autenticação e autorização implementadas com **Keycloak** (OIDC) e **JWT**.
- Envio de token nas requisições autenticadas e validação no backend.
- Proteção contra acessos não autorizados e ocultação de ações no frontend conforme role.

## **9.4 Garantir qualidade, segurança e disponibilidade**

- Organização e modularização do código para manutenção e clareza.
- Tratamento de erros e feedback ao usuário (ex.: mensagens para 401/403).
- Ambiente totalmente containerizado com **Docker Compose**, garantindo execução e avaliação simplificadas.

Dessa forma, o projeto demonstra conformidade total com os critérios de avaliação definidos para a disciplina.

## **10. 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.

## **11. 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.

## 12. Agradecimentos

Agradeço ao professor Leonardo Silva da Gloria pelas orientações e feedbacks durante o desenvolvimento das atividades práticas, e ao Instituto INFNET pela qualidade metodológica e técnica do programa de pós-graduação em Engenharia de Software com Java.

**Projeto acadêmico desenvolvido no Instituto INFNET**