

Enunciado do Projeto: Sistema Monolítico para Gerenciamento de Funcionários e Departamentos

Objetivo

Construir uma aplicação monolítica com Spring Boot que ofereça:

- API REST para CRUD completo de Funcionários e Departamentos
 - Interface web com Thymeleaf para manipulação e visualização dos dados
 - Persistência com MariaDB (produção) e H2 (testes/provas)
-

Tecnologias e Dependências

- Spring Boot
 - Spring Web (API REST + MVC para Thymeleaf)
 - Spring Data JPA
 - Thymeleaf (front-end)
 - Lombok
 - Spring DevTools
 - MariaDB Connector
 - H2 Database
-

Estrutura do Banco de Dados

Entidade: Departamento

- id (Long) — chave primária
- nome (String)
- localizacao (String)

Entidade: Funcionário

- id (Long) — chave primária
- nome (String)
- email (String)
- dataAdmissao (LocalDate)

- departamento — relacionamento @ManyToOne com Departamento
-

Requisitos Funcionais

API REST

Departamento

- POST /departamentos — cadastrar departamento
- GET /departamentos — listar todos os departamentos
- GET /departamentos/{id} — buscar departamento por ID
- PUT /departamentos/{id} — atualizar departamento
- DELETE /departamentos/{id} — excluir departamento

Funcionário

- POST /funcionarios — cadastrar funcionário vinculado a um departamento
- GET /funcionarios — listar todos os funcionários
- GET /funcionarios/{id} — buscar funcionário por ID
- PUT /funcionarios/{id} — atualizar dados do funcionário
- DELETE /funcionarios/{id} — excluir funcionário

Interface Web com Thymeleaf

- Listagem de departamentos e funcionários
 - Formulários para cadastro e edição
 - Visualização detalhada dos registros
 - Navegação intuitiva entre departamentos e seus funcionários
-

Configuração do Banco

application.properties (MariaDB - produção)

spring.datasource.url=jdbc:mariadb://localhost:3306/empresa

spring.datasource.username=seu_usuario

spring.datasource.password=sua_senha

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MariaDBDialect

application-test.properties (H2 - testes/provas)

spring.datasource.url=jdbc:h2:mem:testdb

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=

spring.jpa.hibernate.ddl-auto=create



spring.h2.console.enabled=true

spring.h2.console.path=/h2-console

Entrega no GitHub

- Código-fonte completo da aplicação
- README.md com:
 - Descrição do projeto
 - Passo a passo para executar a aplicação
 - Exemplos de requisições (Postman, curl)
 - Imagens da interface Thymeleaf
- .gitignore configurado para projetos Java

Referências

-  [Tutorial Spring Boot + Thymeleaf + H2](#)
-  [Documentação oficial do Thymeleaf](#)