2025.1 - Laboratório de Programação – Prova

Instruções:

- Esta prova tem como objetivo avaliar sua capacidade de projetar e implementar uma aplicação web simples completa.
- Você deve desenvolver uma aplicação que gerencie uma lista de filmes *assistidos e filmes que deseja assistir.*
- A aplicação deve ser dividida em duas partes principais: um **frontend** em **React** e um **backend** em **Spring Boot**.
- Ao final, você deverá apresentar o projeto funcionando no espaço específico do google classroom

Questão: Você foi contratado para desenvolver um sistema de gerenciamento pessoal de filmes, onde o usuário pode registrar filmes que já assistiu e filmes que deseja assistir.

Campos da entidade Filme:

- id: Identificador único (chave primária, auto-gerada).
- nome: Nome do filme (String).
- ano: Ano de lançamento (Integer).
- classificacao: Classificação indicativa (String, ex: "Livre", "10", "12", "14", "16", "18").
- sinopse: Breve descrição do filme (String).
- status: Enum/String indicando "ASSISTIDO" ou "PARA_ASSISTIR". Sugestão: use um enum em Java para o status para garantir valores consistentes.

Backend (Spring Boot): Desenvolva uma API RESTful utilizando Spring Boot que ofereça os seguintes endpoints para a entidade Filme:

- **GET /api/filmes**: Retorna uma lista de todos os filmes cadastrados.
- **GET /api/filmes/{id}**: Retorna os detalhes de um filme específico pelo id.
- **POST /api/filmes**: Adiciona um novo filme.
- **PUT /api/filmes/{id}**: Atualiza as informações de um filme existente.
- **DELETE /api/filmes/{id}**: Remove um filme.
- Implemente a camada de persistência utilizando **Spring Data JPA** com o banco de dados H2 configurado.
- *Implemente os serviços com as lógicas de negócio e tratamento de erros*

Frontend (React): Crie uma aplicação React que consuma a API desenvolvida no backend e forneça as seguintes funcionalidades na interface do usuário:

- **Listagem de Filmes:**
  - Exiba uma lista de todos os filmes cadastrados, mostrando todos os campos
  - Permita **filtrar** a lista para exibir apenas filmes "ASSISTIDOS" ou "PARA_ASSISTIR".
- **Cadastro de Filme:**

- Um formulário para adicionar novos filmes com todos os campos (nome, ano, classificacao, sinopse, status).
- **Validação básica** dos campos do formulário (ex: nome não pode ser vazio, ano deve ser um número).

- **Edição de Filme:**
  - Funcionalidade para editar as informações de um filme existente. O formulário deve ser pré-preenchido com os dados atuais do filme.
- ***Exclusão de Filme:***
  - Um botão ou ícone para remover um filme da lista, com uma **confirmação** antes da exclusão.
- **Visualização de Detalhes:**
  - Ao clicar em um filme na lista, exiba uma página ou modal com todos os detalhes do filme (incluindo a sinopse completa).

Requisitos do banco:

Modelo de Dados (H2 Database): Crie uma entidade Filme que represente a tabela no banco de dados H2. Configure seu projeto Spring Boot para usar o H2 como banco de dados em memória.

a) Configuração do pom.xml

```
<dependency>
    <groupId>com.h2database</groupId>
    <artifactId>h2</artifactId> <scope>runtime</scope>
</dependency>
```

b) Configuração do application.properties (ou application.yml): Configure o Spring Boot para usar o H2 e exponha o console para visualização:

```
# application.properties
spring.h2.console.enabled=true
spring.h2.console.path=/h2-console
spring.datasource.url=jdbc:h2:mem:filmesdb
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=sa
spring.datasource.password=
spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
spring.jpa.hibernate.ddl-auto=update # ou create, para recriar o schema a cada inicialização
```