Application Development Frameworks

Assignment 2

You may work alone or in pairs. Either way you have the same specification and will receive the same grade.

You have all the material you need within Canvas - videos, boilerplate code, sample project etc.
Please do not Google because there is too much information out there, not all of which is good.

You are to develop a website that stores information about directors and the films they have directed.

You may assume that a film has only one director.

- Any user can
    - view a list of the directors in the database in alphabetical order (by surname)
    - select a director from the list to view that director's films
    - view a list of the films in the database in chronological order showing all information you have about that film
    - register by providing an email and password
- Authenticated users can
    - add a director to the database
        - no two directors can have the same name.
        - apply sensible limitations to the name e.g. use @Pattern and @Size
    - add a film providing the film's title, year of release and a director, which must be selected from a list provided.
        - Two films can have the same title. When a film is re-made it often has the same name as the original.
        - A film's title cannot be left blank.
        - The film's year of release must be an integer between 1888 and the current year (which will not always be 2020).
    - edit the title of a film
        - requires that the method's signature created in the repository be annotated as `@Transactional` (facilitating rollback if necessary) and `@Modifying`.
- Administrators can
    - delete a film
    - delete a director along with their films

You must also provide

- Two REST API endpoints which you write yourself, to allow authenticated users to,
    - access (in json format) films released in a particular year

    - delete a director (and associated films) given their id

- An example of consuming these two APIs which requires another Spring project. Nothing fancy is needed here, just a second WebMVC project that has two controllers and two views. The controllers send authentication data to the REST APIs and present the result of the request in suitable format. No forms are needed.

Technical Notes

You will use **WebMVC** Spring Boot application.

Use an **in-memory h2** database to create an "out of the box" application. This database must be populated with sample data.

The application must be implemented using be implemented using **JPA**, making use of all that offers. This project essentially has two entities, a film and its director, along with users and their roles.

Use the **Security** module for authentication and authorisation.

I am not interested in visual styling – just make the web site well structured, readable, navigable, etc - although you should use some CSS. Credit any templates (CSS/HTML) if you use them. I recommend the use of fragments to reduce the amount of code.

Forms must be validated using form binding and **your own** (not Spring or client-side) **error messages** should be displayed by the view if the user makes a mistake.

It must be possible to **change the language** of the website. User input must be validated with suitable (international) error messages. You do not have to translate content. I will accept subtle changes e.g. "Film Title" becoming "Film Title_FR" to indicate the language has changed to French.

You **must** use the following:

- Spring Boot
- Spring MVC
- Thymeleaf (not JSP)
- Spring Data JPA
- Spring Security
- H2 for an embedded database
- Maven

Unit tests are *not* required.

You may use **Project Lombok** to reduce boilerplate code.

Provide a **brief document** outlining the high-level design of your system (1 or 2 A4 pages) including but not limited to your database design and class diagrams and the beans which you used.