

Prueba - Sistema IMDb

- Para realizar esta prueba debes haber estudiado previamente todo el material disponible en el LMS correspondiente al módulo.
- Una vez terminada la prueba, comprime la carpeta que contiene el desarrollo de los requerimientos solicitados y sube el .zip en el LMS.
- Puntaje total: 10 puntos
- Desarrollo prueba: Individual

Habilidades a evaluar

- Añadir las dependencias y propiedades para el correcto funcionamiento de la aplicación.
- Diseñar las entidades, relaciones entre estas y DAO para persistir información en el proyecto.
- Crear las vistas y templates necesarios para login, registro y mostrar la información desde los controladores.
- Construir servicios y controladores para exponer los métodos necesarios relacionados a la lógica de negocio del proyecto.
- Implementar capa de seguridad y su configuración correspondiente con tal de autorizar y autenticar a los usuarios de la aplicación.

Descripción

La funcionalidad que se debe desarrollar, es de un sistema que permita a los usuarios ver, crear, editar, borrar y calificar su show favorito (la calificación es similar a las de [IMDb](#) o [Netflix](#)).

Para realizar este proyecto, se espera que desarrolle un proyecto con las funciones CRUD básicas y las vistas necesarias en Spring Boot con tal que se muestren todos los shows que ya hayan sido creados por otros usuarios y dé la posibilidad a los usuarios de evaluarlos con nota de 1 a 5.

- Los usuarios deben ser autenticados en la aplicación y autorizados a través de roles.
- Debes usar etiquetas JSTL en las vistas JSP.
- Usar validaciones en los atributos de las entidades y las anotaciones correspondientes.

Consideraciones

- Si bien pueden haber distintas formas de realizar algunas partes de este proyecto, los requerimientos esenciales son transversales.
- Utilizar inyección de dependencias con *@Autowired* y usar *Lombok*, para generar Getters y Setters son una buena idea para optimizar el tiempo e ir avanzando más rápido.
- La paginación en la capa DAO es irrelevante para este proyecto.

Requerimientos

Requerimiento 1

- a. Iniciar un nuevo proyecto en Spring Boot, puede ser con Spring Initializr o directamente en Spring Tool Suite.

Agregue la configuración básica (**1 punto**):

- Nombre del proyecto
- Versión de Java
- Group
- Artifact
- Description
- Package

Deben estar las dependencias necesarias para este proyecto (**1 punto**):

- *Oracle (MySQL opcional)*
- *Spring Security*
- *Spring Web*
- *Spring DevTools*
- *Spring JPA*
- *JSTL*
- *Jasper*
- *Lombok (opcional)*
- *Spring Boot Starter Validation (opcional)*
- Otras dependencias a elección.

Requerimiento 2

- a. Crear en un nuevo package llamada 'model' las entidades a utilizar en el proyecto. Estas deben tener al menos los siguientes atributos (**1 punto**).

Crear clase User

```
private Long id;  
private String username;  
private String email;  
private String password;  
private String passwordConfirmation;
```

Crear clase Show

```
private Long id;  
private String showTitle;  
private String showNetwork;
```

Crear clase Rating

```
private Long id;  
private int rating;
```

Crear clase Role / Crear enum Role

```
private Long id;  
private String name;
```

HINT: debes agregar las validaciones correspondientes a los atributos, en conjunto con sus getters y setters y, además, las relaciones entre las entidades.

- b. Crear las interfaces de repositorio para la capa de persistencia / dao (**1 punto**).
- *UserRepository*
 - *ShowRepository*
 - *RatingRepository*
 - *RoleRepository*

Todas estas interfaces deben tener al menos el método **List<T> findAll()**, en donde 'T' es el tipo de objeto que retorna (e.g. List<User> findAll();). Estas interfaces deben extenderse de JpaRepository o CrudRepository según corresponda la configuración de su proyecto.

Requerimiento 3

- a. Crear la capa de servicios de la aplicación (**1 punto**).
 - *UserService*
 - *ShowService*
 - *RatingService*
- b. Crear los controllers del proyecto (**1 punto**).
 - *UserController* (métodos diferenciados para login y registro)
 - *ShowController* (aquí debe ir el método para agregar Rating también.)

Requerimiento 4

- a. Crear las siguientes vistas (**1 punto**):
 - *home.jsp* (o *index.jsp*, esta vista muestra todos los shows creados por los usuarios)
 - *new.jsp* (vista para crear nuevo Show)
 - *edit.jsp* (vista para editar un Show mediante {id})
 - *show.jsp* (vista para mostrar el contenido de un Show y permitir evaluar con Rating)
- b. Crear las vistas de login y registro (**1 punto**).
 - *login.jsp*
 - *registration.jsp*

HINT: las vistas de login y registro pueden ir en una sola vista. No hay problema. Lo importante es que los métodos para login y registro estén diferenciados en los controllers.

Requerimiento 5

- a. Configure autorización y autenticación con Spring security (**2 puntos**):
 - Crear clase *WebSecurityConfig.java* que extienda *WebSecurityConfigurerAdapter*. Esta clase debe ir en un package llamado 'config'.
 - Crear clase *UserDetailsServiceImpl* que implementa la interfaz *UserDetailsService*. Esta clase debe ir en el package de servicios.

- Verifique seguridad y validaciones en el argumento de los métodos en los controladores (*@Valid, Principal principal, Binding Result*).

Requerimientos opcionales

- Crear una interfaz personalizada en donde las clases que hereden de ella implementen sus métodos (por ejemplo que los servicios hereden de la misma interfaz e implementen los mismos métodos cambiando el argumento y retorno de estos).
- Crear un DTO, setear atributos y enviar el objeto, desde los controladores, hacia las vistas para que se muestre sus valores.
- Crear una API REST (puede ser otra clase con anotación *@RestController* o puede ser como proyecto aparte)
- Crear una query JPQL o una query nativa en la capa de persistencia. Esta puede tener como objetivo filtrar mediante uno o más parámetros, retornar resultados ascendentes o descendentes, buscar show por rating, etc.

HINT: Ejemplo guía de lo que debiera construir:

login.jsp

The screenshot shows a web page with a dark header bar containing the links 'TV Shows', 'Login', and 'Registration'. Below the header, the page title 'Login' is displayed. There are two input fields: 'Email address' with the placeholder text 'email@email.com' and 'Password' with masked characters '*****'. A blue 'Login!' button is positioned below the password field.

registration.jsp

TV Shows Login Registration

Register!

Username:

Email:

We'll never share your email with anyone else.

Password:

Password Confirmation:

[Register!](#)

index.jsp / home.jsp

TV Shows Add a show Logout

Welcome, email2@email2.com

TV shows

Show	Network
ShowName1	Network1

new.jsp

TV Shows

Add a showLogout

Create a new show

Show title:

ShowName1

Network:

Network1

Create

Go back

edit.jsp

TV Shows

Add a showLogout

ShowName1

Show title:

ShowName1

Network:

Network1

Update

Delete

Go back

show.jsp

TV Shows Add a show Logout

Title: ShowName1
Network: Network1

Users who rated this show

Name	Rating
------	--------

Edit

Leave a rating: Rate!

Go back

show.jsp

TV Shows Add a show Logout

Title: ShowName1
Network: Network1

Users who rated this show

Name	Rating
email2@email2.com	5

Edit

Leave a rating: Rate!

Go back