CHALLENGE BACKEND - Java Spring Boot (API) 🚀

Situación Inicial 📍

Una empresa te contrata a tí y a tu equipo de developers para desarrollar una aplicación que le permita a niños y niñas hispanohablantes explorar el mundo de Disney. Para hacer esto, el cliente te solicita que en la aplicación se puedan conocer y modificar los personajes que lo componen y entender en qué películas participaron cada uno de esos personajes.

Objetivo 🎯

Para lograr la solicitud del cliente, deberás desarrollar una API que permita navegar por estos personajes y sus películas y se deberá exponer la información para que cualquier frontend pueda consumirla. Algunos elementos que debes tener en cuenta:

👉 Utilizar Spring Boot.

👉 No es necesario armar el Frontend.

👉 Las rutas deberán seguir el patrón REST.

👉 Utilizar la librería Spring Security.

⚠️ ¡No es indispensable hacer todo!

Mientras más completes, mayor puntaje obtendrás, pero puedes enviar la app hasta el estadío que la tengas en base a tu cono

Requerimientos técnicos 💻

1. Modelado de Base de Datos

- **Personaje**: deberá tener:
  - Imagen
  - Nombre
  - Edad
  - Peso
  - Historia
  - Películas o series asociadas
- **Película o Serie**:

- Este ítem deberá contener
  - Imagen
  - Título
  - Fecha de creación
  - Calificación (del 1 al 5)
  - Personajes asociados

- **Género:**
  - Este ítem deberá tener:
    - Nombre
    - Imagen
    - Películas o series asociadas

2. Creación, Edición y Eliminación de Personajes (CRUD)

- Deberán existir las operaciones básicas de creación, edición y eliminación de personajes.
- Al guardar/actualizar un personaje, se deberá devolver esta entidad completa, es decir, con sus películas o series relacionadas.
- Al momento del Update, es importante solo actualizar la Entidad Personaje y no su listado de películas

3. Creación de Géneros

Deberá existir la operación de creación de Genero .

4. Detalle de Personaje

En el detalle deberán listarse todos los atributos del personaje, como así también sus películas o series relacionadas.

5. Búsqueda de Personajes

Deberá permitir buscar por nombre, y filtrar por edad, peso o películas/series en las que participó.
Para especificar el término de búsqueda o filtros se deberán enviar como parámetros de query:

- GET /characters?name=nombre

- GET /characters?age=edad
- GET /characters?movies=idMovie

El listado deberá mostrar:

- Imagen.
- Nombre.

El endpoint deberá ser:

- /characters

Recordar que si ningún filtro es enviado, se deben devolver todas las entidades.

6. Detalle de Película / Serie con sus personajes

Devolverá todos los campos de la película o serie junto a los personajes asociados a la misma

7. Creación, Edición y Eliminación de Película / Serie.

Deberán existir las operaciones básicas de creación, edición y eliminación de películas o series.

- Al crear una Película, crearla con sus personajes asociados
- Al guardar/actualizar una película, se deberá devolver esta entidad completa, es decir, con sus personajes asociados.
- Al momento del Update, es importante solo actualizar la Entidad Película y no su listado de personajes

8. Búsqueda de Películas o Series

Deberá permitir buscar por título, y filtrar por género. Además, permitir ordenar los resultados por fecha de creación de forma ascendiente o descendiente.

El término de búsqueda, filtro u ordenación se deberán especificar como parámetros de query:

- /movies?name=nombre
- /movies?genre=idGenero
- /movies?order=ASC | DESC

El listado deberá mostrar:

- Imagen.
- Título
- Fecha de Creación.

El endpoint deberá ser:

- GET /movies

Recordar que si ningún filtro es enviado, se deben devolver todas las entidades.

9. Agregar/Remover personajes a una película

Deberá existir un endpoint que nos permita agregar/remover personajes a una película.

Los endpoint deberán ser:

- POST /movies/{idMovie}/characters/{idCharacter}
- DELETE /movies/{idMovie}/characters/{idCharacter}

10. Autenticación de Usuarios

Para realizar peticiones a los endpoints subsiguientes el usuario deberá contar con un token que obtendrá al autenticarse. Para ello, deberán desarrollar los endpoints de registro y login, que permitan obtener el token.

Los endpoints encargados de la autenticación deberán ser:

- /auth/login
- /auth/register

11. Envío de emails

Al registrarse en el sitio, el usuario deberá recibir un email de bienvenida. Es recomendable, la utilización de algún servicio de terceros como SendGrid.

Documentación

Es deseable documentar los endpoints utilizando alguna herramienta como Postman o Swagger.

Tests

De forma opcional, se podrán agregar tests de los diferentes endpoints de la APP, verificando posibles escenarios de error:

- Campos faltantes o con un formato inválido en BODY de las peticiones
- Acceso a recursos inexistentes en endpoints de detalle

Los tests pueden realizarse utilizando JUnit y Mockito.