SISTEMA DE CALIFICACIONES

INSTRUCCIONES

Lee cuidadosamente cada una de las cuestiones que se te presentan a continuación y responde de acuerdo con lo solicitado.

DESCRIPCIÓN:

La Universidad de Los Andes necesita desarrollar una aplicación que trabaje en conjunto con un api para llevar el control de notas de los estudiantes en carreras de pregrado y debe cumplir con las siguientes características.

La funcionalidad que se debe desarrollar es de un sistema capaz de manejar datos de alumnos y para saber que materias tienen asignadas.

Para lograr la funcionalidad, lo que se espera es que se desarrolle un servicio REST utilizando Spring, para contener la lógica de los alumnos con sus respectivas materias. Además, este servicio REST debe contener una autenticación mediante JWT.

Se deben probar los *endpoints* para crear alumnos, materias y usuarios, ya sea usando *Insomnia* u otro cliente para probar servicios REST.

Se deben ingresar datos de un usuario para que se pueda iniciar sesión y obtener un token JWT, además de insertar alumnos con algunas materias.

Para visualizar los datos se debe desarrollar una aplicación con Spring , en la que se deba iniciar sesión a través de un formulario y pueda validarse contra el servicio REST que contiene los usuarios. Luego, se debe visualizar una vista con un menú donde se vean los alumnos existentes con sus respectivas materias.

HITO 1 (BACKEND)

- Usar Spring Initializr para generar proyecto con las respectivas dependencias. Crear modelos en carpeta models para que persistan.

    1. Clase Alumno con los siguientes atributos:

        - Id tipo Long,
        - Rut tipo String,
        - Nombre tipo String,
        - Dirección tipo String,

- materiaList de tipo Set<Materia>

2. Clase Materia con los siguientes atributos:
    - Id tipo Long,
    - Nombre tipo String,
    - Alumno tipo Alumno

Nota: Se debe implementar un `Logger` dentro de la creación del proyecto, la forma y lugar es de libre elección.

- Crear interfaces que implementen JpaRepository en carpeta repository.
    1. Interfaz AlumnoRepository.
    2. Interfaz MateriaRepository.
- Crear capa de servicios.
    1. Crear clase AlumnoService.
        - Crear método save para guardar alumno.
        - Crear método findAll para capturar todos los registros de alumnos.
    2. Crear clase MateriaService.
        - Crear método save para guardar alumno.
- Crear clase AlumnoController.
    1. Método findAll para obtener todos los alumnos.
    2. Método save para guardar un alumno.
- Crear clase MateriaController.
    1. Método save para crear una materia.

JWT

- Se deben agregar las dependencias de Spring Security y jsonwebtoken.io a Maven.
- Se debe crear clave y tiempo de expiración del token dentro de application.yml en carpeta resources.
- Crear modelos

1. Clase User con los siguientes atributos:
    - Id tipo Long,
    - Name tipo String,
    - Username tipo String,
    - Email tipo String,
    - Rolest de tipo List<Role>
2. Crear Enumeración Role.

```
public enum Role implements GrantedAuthority {
 ROLE_ADMIN, ROLE_CLIENT;

 public String getAuthority() {
  return name();
 }
}
```

- Crear interfaz UserRepository la que debe implementar JpaRepository.
- Crear excepción desde RuntimeException para lanzar excepciones personalizadas.
- Crear clases necesarias para verificar token y generarlo, deben ir en carpeta security.
    1. Crear clase JwtTokenProvider.
    2. Crear clase JwtTokenFilter.
    3. Crear clase JwtTokenFilterConfigurer.
- Crear clase WebSecurityConfig que herede de WebSecurityConfigurerAdapter para asegurar la aplicación.
- Crear clase UserService
    1. Método signin para autenticar usuario.
    2. Método signup para registrar usuario.
    3. Método loadUserByUsername para que se pueda verificar los atributos del usuario.
- Crear clase UserController, para exponer los endpoints.
    1. Método signup para registro de usuarios.
    2. Método signin para login de usuarios, retornando un token.

HITO 2 (FRONTEND)

- Usar Spring Initializr para generar proyecto con las respectivas dependencias para la aplicación web
- Crear objetos DTO para manipular datos.
    1. Crear clase AlumnoDTO con los siguientes atributos:
        - Id tipo Long,
        - Rut tipo String,
        - Nombre tipo String,
        - Dirección tipo String,
        - materiaList de tipo Set<Materia>
    2. Crear clase MateriaDTO con los siguientes atributos:
        - Id tipo Long,
        - Nombre tipo String,
        - Alumno tipo Alumno
    3. Crear clase UserDTO con los siguientes atributos:
        - Username tipo String,
        - Password tipo String,
        - List<Role> roles
- Crear capa de servicios.
    1. Crear clase AlumnoService.
        a. Método findAll, consume servicio REST del Backend.
    2. Crear clase UserService.
        a. Método signin, para postear el usuario e iniciar sesión en el front.
- Crear controlador LoginController.
    1. Método login para mostrar en pantalla la vista del login.
    2. Método home para mostrar los datos obtenidos desde el servicio Alumnos.
- Crear vistas estáticas en resources/templates.

1. Vista login.html
2. Vista home.html

- Crear clase WebSecurityConfig