Final Laboratorio III - Tema 2

El objetivo de este trabajo final es la implementación de un conjunto de API's REST en la plataforma Java, utilizando el framework Spring Boot que vimos durante el cursado de la materia.

Todos los proyectos deberán ser entregados en GitHub, deberán crear un repositorio y enviarme el link al mismo con no menos de cinco (5) días de anticipación a la fecha de final.

Introducción

Para este ejercicio, se deberá realizar una aplicación en capas como la del ejemplo que seguimos durante el cursado de la materia. En este caso, estaremos implementando algunos de los endpoints requeridos para el sistema de alumnos que implementamos durante la cursada. Se debe tener en cuenta que deben generarse casos de test para **al menos el 70% de los métodos públicos** de cada una de las clases de cada capa (presentación/controller, negocio/servicios y persistencia, las clases de modelo y dto's pueden obviarse ya que deberían ser clases simples con métodos get y set unicamente).

Las entidades (clases de modelo)

Las entidades con las que vamos a estar trabajando principalmente serán las que estuvimos viendo durante el semestre: Alumno, Materia, Profesor, Carrera, Asignatura. Tengan en cuenta que es posible que sea necesario agregar o eliminar, según crean, atributos o métodos en estas clases.

Se debe crear un diagrama UML con la especificación de las clases necesarias para el modelado de nuestro sistema.

Exposición de la información con APIs REST

Queremos crear endpoints REST para las siguientes operaciones. Debajo se indica cómo deben quedar los endpoints y a qué tipo de método HTTP deben responder y un ejemplo del mismo en algunos casos. Un tip para este punto es que deberán tener en cuenta no sólo el caso feliz sino que también tener en cuenta qué ocurre por ejemplo si no existen materias a mostrar, o no existe el alumno que quiero eliminar.

Profesor:
El profesor es una entidad que ya estuvimos viendo en clase, pero para la cual no creamos ningún endpoint

- Crear, Modificar y Eliminar un profesor
  POST: /profesor
  PUT: /profesor/{idProfesor}

DELETE: /carrera/{idProfesor}

- Obtener todos las *Materia*s que dicta un profesor ordenadas en orden alfabético
  GET: /profesor/materias

Materia

- Queremos modificar nuestro endpoint de creación de materias para que incluya una lista de correlatividades en formato de array de enteros, el body del json debería quedar de la siguiente manera. Tengamos en cuenta que al momento de crear una materia deberá tener un profesor y que se deberá asociar esa materia a ese Profesor

```
{
    "nombre" : "Luciano",
    "anio" : 1,
    "cuatrimestre": 1,
    "profesorId" : 3,
    "correlatividades" : [12,45,69]
}
```

Alumno:

- Crear, Modificar y Eliminar un alumno.
  POST: /alumno
  PUT: /alumno/{idAlumno}
  DELETE: /alumno/{idAlumno}
- Cursar, Aprobar o Perder regularidad de una asignatura para un alumno
  Para este caso, lo que ocurre es que se intentará modificar el estado de una de las asignaturas. Asumiremos cuando se crea el alumno que se creará una Lista de Asignaturas asociadas a él. Pero para esto deberemos tener hecha la modificación del nuestro endpoint de creación de Materias, de forma tal de poder crear las Materias y sus listas de correlatividades
  PUT: /alumno/{idAlumno}/asignatura/{idAsignatura}