Informe de Desarrollo y Corrección de proyecto

Estudiante: Juan Pablo Rivera Orozco

Exámen 1 BackEnd 2  
2025

1. Descripción del Proyecto

El proyecto consiste en una API básica desarrollada en Java utilizando Spring Boot y JPA/Hibernate para la gestión de una base de datos relacional. Se modelan tres entidades principales: Usuario, Docente y Curso, relacionadas entre sí. El objetivo es permitir la creación, consulta y asociación de cursos con docentes y usuarios, representando una estructura educativa sencilla. La base de datos utilizada es MySQL.

2. Listado de errores corregidos con explicación

A) Commit 1 – "Clase Curso Revisada"

* Error en la anotación del ID: La anotación @Id estaba mal escrita. Se corrigió para que la clase Curso pueda funcionar como entidad.
* Estrategia de generación de ID incompleta: Se completó la anotación @GeneratedValue(strategy =GenerationType.IDENTITY).
* Falta de modificador de acceso en atributo: La variable Docente docente no estaba declarada como private.
* Constructor incompleto: Se agregó la variable docente en el constructor parametrizado.
* Getters y Setters ausentes: Se añadieron los métodos get y set para todos los atributos.

B) Commit 2 – "Se corrigió Docente y Usuario, enum TipoUsuario creado"

Clase Docente:

* Anotación @Entity mal escrita → corregida.
* Faltaba el @Id para definir la clave primaria → añadido.
* Constructores añadidos: uno vacío y uno parametrizado para id y especialidad.
* Getters y Setters añadidos para atributos relacionados (cursos, usuario).

Clase Usuario:

* @Entity corregida.
* @GeneratedValue completada con IDENTITY.
* @Column mal escrita → corregida.
* @Enumerated(EnumType.STRING) requiere un Enum → se creó TipoUsuario.
* Getters y Setters añadidos.
* Se reemplazó el uso de la letra "ñ" por "n" en el atributo contrasena, por buenas prácticas de codificación y compatibilidad.

C) Commit 3 – "Application.properties + build.gradle"

* Base de datos creada en MySQL con el nombre develop\_db usando phpMyAdmin.
* Archivo application.properties configurado correctamente para la conexión :

URL: jdbc:mysql://localhost:3306/develop\_db

* Dependencia del conector MySQL añadida en build.gradle: implementation 'mysql:mysql-connector-java:8.0.33'

3. Guía paso a paso de cómo realizar la conexión a la base de datos

A) Requisitos Previos:

* Tener MySQL instalado y corriendo.
* Tener Gradle y Spring Boot configurados en tu proyecto.
* Haber creado la base de datos (develop\_db) en tu servidor local (localhost).

B) Pasos:

* Crear la base de datos:

En phpMyAdmin con el nombre debelop\_db:

* Configurar application.properties:

Ubicado en src/main/resources/application.properties:

* spring.datasource.url=jdbc:mysql://localhost:3306/develop\_db

spring.datasource.username=root

spring.datasource.password=

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect

* Agregar la dependencia de MySQL en build.gradle:

dependencies {

implementation 'mysql:mysql-connector-java:8.0.33'

// otras dependencias...}

C) Ejecutar la aplicación:

Corre el archivo Examen1Back2Application.java (el archivo principal con @SpringBootApplication) para que Spring Boot conecte y genere las tablas en develop\_db automáticamente.

4. Recomendaciones para evitar errores similares en el futuro

* Verificar cuidadosamente las anotaciones de JPA (@Entity, @Id, @Column, etc.) para evitar errores de mapeo o fallos al ejecutar la aplicación.
* Usar nombres de variables sin caracteres especiales o tildes para mantener la compatibilidad y buenas prácticas.
* Completar las configuraciones de generación de IDs (@GeneratedValue(strategy = ...)) para evitar errores en la persistencia.
* Agregar todos los getters y setters necesarios para que JPA pueda acceder correctamente a los atributos.
* Crear constructores adecuados, incluyendo aquellos con parámetros si vas a instanciar entidades manualmente.
* Revisar las relaciones entre entidades (@OneToOne, @ManyToOne, @OneToMany, etc.) y usarlas junto con las anotaciones de serialización de Jackson (@JsonManagedReference, @JsonBackReference) para evitar ciclos infinitos en respuestas JSON.
* Probar la conexión a la base de datos al inicio del desarrollo, y no dejarlo para el final.