JPA

1.- Qué es crear una aplicación por capas?

Crear una aplicación por capas significa organizar el código de la aplicación en secciones bien definidas, donde cada capa tiene una responsabilidad específica y se comunica solo con las capas adyacentes.

Esta separación mejora la mantenibilidad, escalabilidad y reutilización del código.

Capa de presentación (Controller): Interactúa con el cliente. Recibe las solicitudes HTTP y devuelve respuestas. No contiene lógica de negocio.

Capa de servicio (Service): Contiene la lógica de negocio. Llama a los repositorios y prepara los datos para los controladores**.** Ayuda a mantener el controlador limpio y enfocado solo en la entrada/salida.

Capa de acceso a datos (Repository): Se comunica directamente con la base de datos. Ejecuta las operaciones CRUD.

Capa de modelo (Entity): Define las estructuras de datos o entidades del dominio (tablas de la BD).

2.- Qué es un ORM?

Un ORM (Object-Relational Mapping), o en español Mapeo Objeto-Relacional, es una técnica o herramienta que permite interactuar con una base de datos relacional (como MySQL o PostgreSQL) utilizando objetos del lenguaje de programación (por ejemplo, objetos Java, Python, etc.).

Un ORM traduce automáticamente las operaciones que haces con objetos (crear, leer, actualizar, eliminar) en sentencias SQL que la base de datos puede entender.

¿Qué hace un ORM?

Mapea clases a tablas

Mapea atributos a columnas

Mapea objetos a filas

Permite usar consultas con métodos y objetos, en vez de SQL puro

Gestiona relaciones (@OneToMany, @ManyToOne, etc.)

Se encarga del acceso a datos y sus conversiones

3.- Para qué sirve la dependencia Lombok?

La dependencia Lombok sirve para reducir la cantidad de código repetitivo de las clases Java, especialmente para:

Getters y setters

Constructores

Método toString()

Métodos equals() y hashCode()

Builders

Entre otros.

¿Qué hace exactamente?

En tiempo de compilación, Lombok genera automáticamente esos métodos a partir de anotaciones que colocas en tus clases. Así tu código se mantiene limpio, corto y más fácil de leer.

4.- Para qué sirve la capa de controlador?

La capa de controlador (también llamada capa de presentación) es la puerta de entrada de una aplicación, especialmente en aplicaciones web o APIs REST. Su función principal es:

- Recibir las peticiones del usuario o cliente (por ejemplo, desde un navegador, una app móvil o Postman).
- Devolver la respuesta adecuada (como datos en JSON o un mensaje de error).

Funciones principales del controlador:

Gestionar solicitudes HTTP: GET, POST, PUT, DELETE, etc.

Recibir parámetros desde la URL o el cuerpo de la petición.

Validar datos de entrada (aunque también puede hacerlo el servicio).

Llamar a la capa de servicio para ejecutar la lógica de negocio.

Devolver respuestas (normalmente en formato JSON o HTML).

5.- Qué diferencia hay entre ddl-auto=create-drop y ddl-auto=update? (propiedades)

La propiedad spring.jpa.hibernate.ddl-auto en Spring Boot define cómo Hibernate gestiona el esquema de la base de datos (crear, actualizar o eliminar tablas, columnas, etc.) cada vez que la aplicación arranca.

ddl-auto=create-drop Crea el esquema al arrancar y lo elimina al cerrar la aplicación.

ddl-auto=update Actualiza el esquema: crea nuevas tablas/columnas si es necesario,

pero no borra datos ni estructuras existentes.

6.- En qué orden deben crearse las clases para crear la API con JPA y que no dé error?

1. Entidad
2. Repositorio
3. Servicio
4. Controlador