

***APUNTES DE***

***SPRING BOOT***

**EJEMPLO DE ESPECIFICACIONES EN EL REPO DE LUISMI, EN 08\_Validacion**

**1 DTO**

**EJEMPLO DTO (Conviene que los DTOs sean ‘Record’):**

public record GetProductoDto(

Long id,

String nombre,

double precio,

) {

public static GetProductoDto of(Producto p) {

return new GetProductoDto(

p.getId(),

p.getNombre(),

p.getPrecio()

);

}

}

**2 REPOSITORIOS**

**Los repositorios siempre son interfaces y extienden a la interfaz JpaRepository<T, ID> que tiene todos aquellos métodos que se utilizan para crear los métodos del Servicio.**

public interface ModelRepository extends JpaRepository<Model, Long> {

}

**3 SERVICIOS**

**Inserte explicación de los Servicios.**

**4 CONTROLADORES**

**Los controladores en Spring Boot son clases Java responsables de manejar las solicitudes HTTP entrantes y devolver una respuesta adecuada . Actúan como intermediarios entre el cliente (generalmente un navegador web o una aplicación móvil) y la lógica comercial de la aplicación.**

**Van anotados con:** @Getter @Setter @AllArgsConstructor @NoArgsConstructor @Builder @ToString y principalmente **@Entity**.

**Los métodos controladores siempre hacen uso de aquellos métodos instaurados en el Servicio.**

Todos los métodos controladores siempre tienen en el “return” el respectivo método de cada servicio acompañado de algún ResponseEntity o stream según el caso. El delete es el único método que no tiene en el return el método del Servicio, este se efectúa antes del return ya que dicho método devuelve un ResponseEntity<?>.

La anotación @RequestBody se utiliza para vincular el cuerpo de una solicitud HTTP a un parámetro de método en un método controlador, o sea, Spring Boot convierte automáticamente el cuerpo de la solicitud en el objeto Java especificado. Se utiliza en los métodos como **create** o **update** porque hace falta pasarles el cuerpo de un objeto para que se ejecuten, ya sea para crearlo o editarlo.

**getAll**

* Anotado con @GetMapping.
* Devuelve una lista de un Modelo o de un ModeloDTO.

**getByID**

* Anotado con @GetMapping.
* En los argumentos del método: @PathVariable.
* Devuelve un Modelo.

**create**

* Anotado con @PostMapping.
* En los argumentos del método: @RequestBody.
* Devuelve un ResponseEntity<Modelo>.
  + return ResponseEntity.status(HttpStatus.CREATED).body(servicio.save(m));

**update**

* Anotado con @PutMapping.
* En los argumentos del método: @PathVariable y @RequestBody.
* Devuelve un Modelo.

**delete**

* Anotado con @DeleteMapping.
* En los argumentos del método: @PathVariable.
* Devuelve un ResponseEntity<?>.
  + return ResponseEntity.noContent().build();