

**DESARROLLO DE UN CRUD UTILIZANDO FRAMEWORK SPRING BOOT**

**INDICACIONES:** El Examen deberá desarrollarse de manera grupal (Máximo 5 Personas). Todos los archivos del proyecto deberán comprimirse en un solo archivo en formato ZIP y subirse a la plataforma el cual deberá contener un archivo de texto (TXT) con el carnet y los nombres de todos los integrantes del grupo. Fecha Límite de Entrega martes 23 de octubre hasta las 23:59.

Desarrolle un sistema que deberá ejecutar un **CRUD completo sobre la tabla productos** de la base de datos **tienda**, y ser consumido desde un **formulario HTML** que utiliza **JavaScript (fetch API)** para comunicarse con la API. el patrón de diseño deberá ser un **MVC mejorado (capas Controller, Model, Repository, Service)**. La tabla productos deberá contener los siguientes campos:

Campo	Tipo de Dato	Descripción
id	INT (PK, AI)	Identificador único (Clave Primaria)
nombre	VARCHAR (100)	Nombre del producto
sku	VARCHAR (50)	Código de referencia
descripcion	TEXT	Descripción detallada
precio	DECIMAL (10,2)	Precio del producto
vencimiento	DATE	Fecha de vencimiento
categoria	VARCHAR (50)	Categoría del producto

**ESTRUCTURA DE CARPETAS DEL PROYECTO SOLICITADO:**

```
src/  
├── main/  
│   ├── java/com/ejemplo/tienda/  
│   │   ├── controller/  
│   │   │   ├── ProductoController.java  
│   │   ├── model/  
│   │   │   ├── Producto.java  
│   │   ├── repository/  
│   │   │   ├── ProductoRepository.java  
│   │   ├── service/  
│   │   │   ├── ProductoService.java  
│   │   │   ├── ProductoServiceImpl.java  
│   │   └── TiendaApplication.java  
│   └── resources/  
│       ├── application.properties  
│       └── static/  
│           ├── index.html  
│           └── main.js
```

## ESPECIFICACIONES FUNCIONALES

### Backend – API REST (Spring Boot)

La API deberá resolver las siguientes Peticiones:

Método	Endpoint	Descripción
GET	/api/productos	Lista todos los productos
GET	/api/productos/{id}	Muestra un producto por ID
POST	/api/productos	Crea un nuevo producto
PUT	/api/productos/{id}	Actualiza un producto existente
DELETE	/api/productos/{id}	Elimina un producto por ID

### Condiciones:

- Los datos deben ser persistidos en la base de datos MySQL.
- El proyecto debe usar **Spring Data JPA** y **Hibernate**.
- Se deben aplicar **buenas prácticas de encapsulación y validación** en el modelo.
- El servicio (ProductoService) debe manejar la lógica del CRUD (no directamente el Controller).

### Frontend – HTML, CSS, JavaScript

Dentro de src/main/resources/static/ se debe incluir:

- index.html:**  
Formulario para crear, editar y listar productos.  
Debe incluir campos para: nombre, sku, descripcion, precio, vencimiento, categoria.
- main.js:**  
Archivo que se comunica con la API mediante **fetch()**, consumiendo los endpoints de la API REST y mostrando los resultados en el HTML.

### CRITERIOS DE EVALUACIÓN:

Criterio	Descripción	Ponderación
Arquitectura MVC	Cumple con la estructura en capas: Controller, Model, Repository, Service	15%
Conexión y persistencia	Base de datos conectada y operaciones CRUD funcionales	20%
API REST	Endpoints REST implementados correctamente (GET, POST, PUT, DELETE)	20%
Lógica en Service	Controladores delegan correctamente la lógica al servicio	10%
Interfaz y comunicación JS	index.html y main.js funcionan con la API mediante fetch()	15%
Documentación y comentarios	Código documentado, nombres claros y buenas prácticas	5%
Ejecución del proyecto	El sistema compila y se ejecuta sin errores	15%

**NOTA:** Deberá también subir a la plataforma el respaldo de la base de datos y utilizar los siguientes parámetros para la base de datos:

**Usuario:** AlumnosPV

**Password:** Prog.V2025

Cualquier archivo corrupto o que **No** Permita el acceso **NO** se CALIFICARA. Favor Tomar Nota.

Buena Suerte 🍀