

# Prueba Técnica Supermercado

## *API REST Para la gestión de ventas en una cadena de supermercados*

### OBJETIVO

El Objetivo de esta Prueba es evaluar conocimientos en **Java + Spring Boot**, incluyendo el desarrollo de una **API REST COMPLETA** que implemente operaciones CRUD con JPA, Relaciones entre entidades, control de errores y excepciones, uso de DTOs, buenas prácticas REST y programación funcional (uso de lambdas y streams) donde aplique.

### DESCRIPCIÓN DEL CASO

Una reconocida cadena de supermercados desea digitalizar su sistema de control de ventas. Para ello necesita una API que permita (de forma básica):

- Registrar Productos con sus respectivos precios.
- Gestionar las sucursales donde se venden los productos.
- Registrar ventas realizadas en una sucursal, especificando los productos vendidos y cantidades.

La empresa desea consultar luego las ventas por sucursal, totalizar ingresos, filtrar productos más vendidos, etc.

### ENTIDADES PRINCIPALES

- **SUCURSAL:** Representa una ubicación física del supermercado (una por cada ubicación).
- **PRODUCTO:** Representa un artículo que puede venderse (ejemplo: Arroz, Botella de Agua, etc).
- **VENTA:** Contiene una o más líneas de productos, asociadas a una sucursal.

## RELACIONES:

- Una **Sucursal** puede tener **muchas Ventas**.
- Una **Venta** tiene **muchos Productos asociados**.
- Un mismo **Producto** puede estar en **muchas Ventas**.

## ✓ REQUISITOS TECNICOS

- Utilizar **Spring Boot** con JPA para el manejo de Bases de Datos.
- Base de Datos Relacional (Por ej: MySQL, PostgreSQL, H2).
- Exponer endpoints RESTful para realizar CRUDs (**GET**, **POST**, **PUT**, **DELETE**, o los métodos que considere necesario).
- Utilizar **DTOs** para separar modelo de dominio y representación externa.
- Manejo adecuado de errores con **ResponseEntity**, códigos HTTP correctos (status codes) y mensajes claros.
- Uso de **Lambdas o Streams** en al menos **una operación del Backend**.
- Organización modular del proyecto (Service, Repository, Controller, etc).

## HISTORIAS DE USUARIOS (REQUERIMIENTOS)

### Productos:

#### 1. Obtener listado de Productos:

- Método: **GET**
- Path: **/api/productos**
- Descripción: Listar todos los productos registrados.

#### 2. Registrar nuevo producto:

- Método: **POST**
- Path: **/api/productos**
- Descripción: Crear un nuevo producto con nombre, precio y categoría.

#### 3. Actualizar producto existente:

- Método: **PUT**
- Path: **/api/productos/{id}**

- Descripción: Modificar los datos de un producto específico.

#### 4. Eliminar un producto:

- Método: **DELETE**
- Path: **/api/productos/{id}**
- Descripción: Eliminar un producto del sistema.

## SUCURSALES:

#### 1. Obtener listado de sucursales:

- Método: **GET**
- Path: **/api/sucursales**
- Descripción: Listar todas las sucursales del sistema.

#### 2. Registrar nueva sucursal:

- Método: **POST**
- Path: **/api/sucursales**
- Descripción: Crear una nueva sucursal con Dirección, Nombre, etc.

#### 3. Actualizar sucursal existente:

- Método: **PUT**
- Path: **/api/sucursales/{id}**
- Descripción: Modificar los datos de una sucursal existente.

#### 4. Eliminar una sucursal:

- Método: **DELETE**
- Path: **/api/sucursales/{id}**
- Descripción: Eliminar una sucursal del sistema.

## VENTAS:

#### 1. Registrar una nueva venta:

- Método: **POST**
- Path: **/api/ventas**

- Payload:

```
{
  "sucursalid": 1,
  "detalle": [
    { "productoid": 10, "cantidad": 2 },
    { "productoid": 20, "cantidad": 4 }
  ]
}
```

- Descripción: Crear una nueva venta para una sucursal con productos y cantidades.

## 2. Obtener ventas por sucursal y fecha:

- Método: **GET**
- Path: **/api/ventas?sucursalid=1&fecha=2026-01-20**
- Descripción: Listar las ventas realizadas en una fecha específica para una sucursal.

## 3. Eliminar/Anular venta:

- Método: **DELETE**
- Path: **/api/ventas/{id}**
- Descripción: Eliminar una venta registrada.
- Se valorará uso de borrado lógico.

Las ventas NO SE PUEDEN MODIFICAR sin permisos de superusuario (no es necesario implementar esto).

# EXTRA - ESTADÍSTICAS (OPCIONAL NO OBLIGATORIO)

## 1. Obtener productos más vendidos:

- Método: **GET**
- Path: **/api/estadisticas/producto-mas-vendido**
- Descripción: Calcular el producto más vendido utilizando Java Streams.