

PROYECTO SPRING BOOT : PruebaTecnica

- Versión de Java usada : 17
- Versión de Maven usada : 3.9.9

PROYECTO ANGULAR : Frontend-Prueba

IMPORTANTE PARA INICIAR EL PROYECTO!!

El ejercicio solicitaba un script para crear la base de datos, la script contiene todo y si se correrá la script primero, es posible! Crear primero la base de datos y luego correr el proyecto spring boot, pero como un plus, también se incorporó la lógica para que de forma automática el proyecto por si solo al ejecutarse por primera vez, cree las tablas(la base de dato al menos ya debe estar creada para poder crear las tablas sobre ella sin importar el método que se utilice)

Importante!!

El archivo application.properties en Src -> Main -> Resource

Contiene la siguiente línea por defecto

`spring.jpa.hibernate.ddl-auto=validate`

Esto por que el ejercicio solicita el script de sql, lo que me hace pensar que crearán la base de datos primero corriendo el script y luego ejecutarán el proyecto, pero en dado caso la persona designada a ejecutar el proyecto quiera crear la base de datos de forma automática con JPA puede simplemente debe cambiar el validate a create-drop, ejemplo:

`spring.jpa.hibernate.ddl-auto=create-drop` (Por ejemplo)

y luego correr el script solo para insertar datos en la tabla que podrán ser usados para evaluar y que el frontend no se muestre vacío.

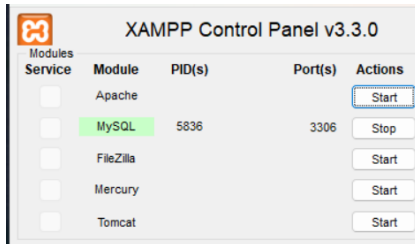
CONTENIDO

| | |
|--|---|
| DESPLIEGUE..... | 3 |
| APIS (Usando Postman) -> Solo inserciones. | 4 |
| PRUEBAS DE INSERCIÓN USANDO POSTMAN | 5 |
| FUNCIONAMIENTO DEL SISTEMA EN TIEMPO DE EJECUCIÓN..... | 7 |
| Nota..... | 8 |

DESPLIEGUE

DESPLIEGUE SPRING BOOT

- **Activar Mysql en Xampp**



- **Abrir el proyecto en IntelliJ y esperar a que se descarguen las dependencias necesarias.**
- **Ejecutar desde la terminal de Maven con spring-boot:run o atajo(Shift + F10) (Tomando en cuenta lo anterior)**

DESPLIEGUE ANGULAR

- **Abrir el proyecto en Visual Studio Code**
- **Ejecutar desde la terminal de visual el comando “npm install” para descargar dependencias**
- **Ejecutar desde la terminal de visual el comando “ng serve -o” para iniciar el proyecto**

Con esto el proyecto debería correr

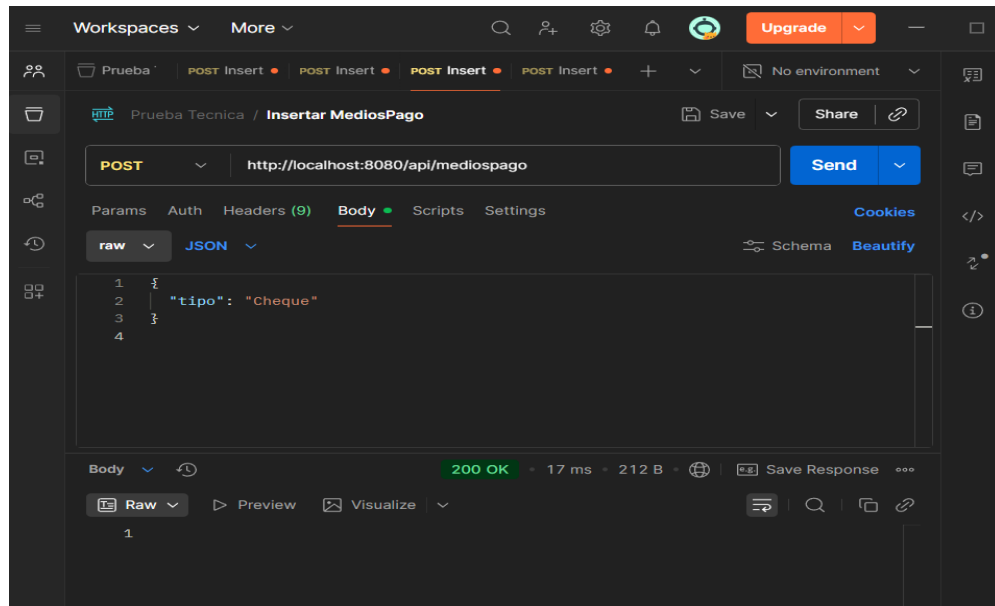
APIS (Usando Postman) -> Solo inserciones.

→ POST

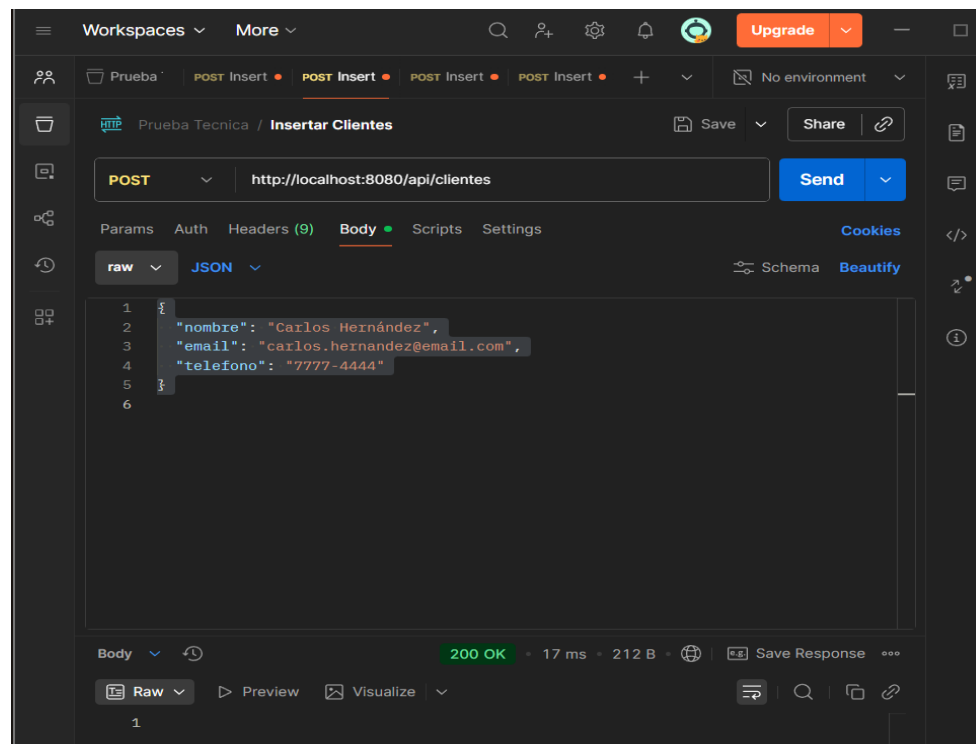
| URL | JSON |
|---|---|
| http://localhost:8080/api/clientes | <pre>{ "nombre": "Carlos Hernández", "email": "carlos.hernandez@email.com", "telefono": "7777-4444" }</pre> |
| http://localhost:8080/api/comercios | <pre>{ "nombre": "Librería Central", "direccion": "Av. Roosevelt #456, San Salvador", "rubro": "Librería" }</pre> |
| http://localhost:8080/api/mediospago | <pre>{ "tipo": "Cheque" }</pre> |
| http://localhost:8080/api/compras | <pre>{ "fecha": "2025-10-25", "lugar": "Sucursal Centro", "montoTotal": 250.75, "cliente": { "idCliente": 1 }, "comercio": { "idComercio": 2 }, "medioPago": { "idMedioPago": 1 } }</pre> |

PRUEBAS DE INSERCIÓN USANDO POSTMAN

Prueba Inserción Medios Pago



Prueba Inserción clientes



Prueba de Inserción Comercios

The screenshot shows a REST client interface with the following details:

- Workspace:** Prueba Técnica / InsertarComercio
- Method:** POST
- URL:** http://localhost:8080/api/comercios
- Body (JSON):**

```
{  "nombre": "Libreria Central",  "direccion": "Av. Roosevelt #456, San Salvador",  "rubro": "Libreria"}
```
- Response:** 200 OK, 27 ms, 212 B
- Raw Response:**

```
1
```

Prueba de Inserción Compras

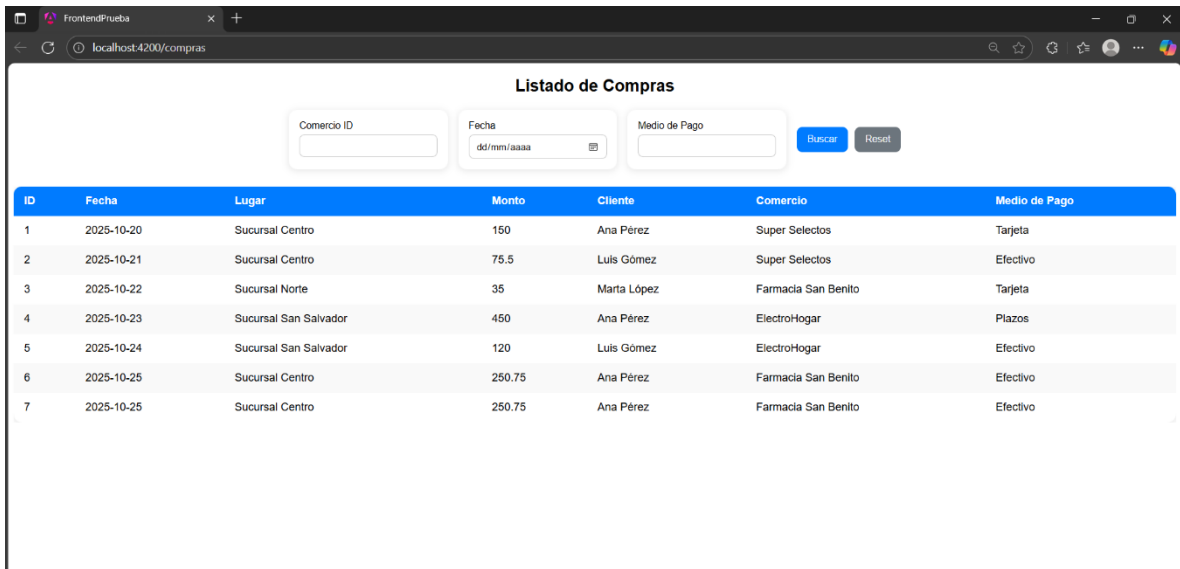
The screenshot shows a REST client interface with the following details:

- Workspace:** Prueba Técnica / Insertar Compras
- Method:** POST
- URL:** http://localhost:8080/api/compras
- Body (JSON):**

```
{  "fecha": "2025-10-25",  "lugar": "Sucursal Centro",  "montoTotal": 250.75,  "cliente": {    "idCliente": 1  },  "comercio": {    "idComercio": 2  },  "medioPago": {    "idMedioPago": 1  }}
```
- Response:** 200 OK, 22 ms, 212 B
- Raw Response:**

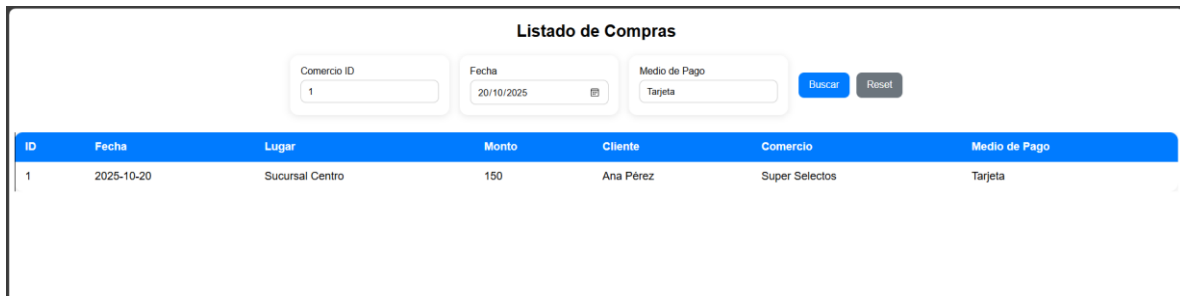
```
1
```

FUNCIONAMIENTO DEL SISTEMA EN TIEMPO DE EJECUCIÓN



| ID | Fecha | Lugar | Monto | Cliente | Comercio | Medio de Pago |
|----|------------|-----------------------|--------|-------------|---------------------|---------------|
| 1 | 2025-10-20 | Sucursal Centro | 150 | Ana Pérez | Super Selectos | Tarjeta |
| 2 | 2025-10-21 | Sucursal Centro | 75.5 | Luis Gómez | Super Selectos | Efectivo |
| 3 | 2025-10-22 | Sucursal Norte | 35 | Marta López | Farmacia San Benito | Tarjeta |
| 4 | 2025-10-23 | Sucursal San Salvador | 450 | Ana Pérez | ElectroHogar | Plazos |
| 5 | 2025-10-24 | Sucursal San Salvador | 120 | Luis Gómez | ElectroHogar | Efectivo |
| 6 | 2025-10-25 | Sucursal Centro | 250.75 | Ana Pérez | Farmacia San Benito | Efectivo |
| 7 | 2025-10-25 | Sucursal Centro | 250.75 | Ana Pérez | Farmacia San Benito | Efectivo |

LO QUE VEMOS ES UN LISTADO GENERAL DE LAS COMPRAS, LUEGO PARA REALIZAR UN FILTRADO SE DEBE TOMAR EN CUENTA QUE TODOS LOS DATOS SON NECESARIOS YA QUE EN EL BACKEND TODOS SON OBLIGATORIOS (ID – FECHA – METODO DE PAGO)



| ID | Fecha | Lugar | Monto | Cliente | Comercio | Medio de Pago |
|----|------------|-----------------|-------|-----------|----------------|---------------|
| 1 | 2025-10-20 | Sucursal Centro | 150 | Ana Pérez | Super Selectos | Tarjeta |

Nota

Me hubiese gustado ajustar algunos detalles, sobre todo en el frontend pues quedo muy buena la presentación pero el funcionamiento es algo tosco, ya con el hecho de que todos los datos sean required, me gustaría ajustar eso, lo único que subiré serán los commits en diseño y este documento con esta información que se me hizo valiosa, trate de precisar un documento corto, conciso y muy digerible para que sea leído rápido y correr el proyecto desde otra maquina sea sencillo, aun así, ante cualquier problema estoy a la orden.