**🔹 1. Crear la estructura del proyecto**

1.1 Me ubiqué en mi carpeta de proyectos abriendo el terminal de vscode que uso como IDE

Cd: G:/proyectos/java

1.2 Ejecutar el siguiente comando en la terminal para generar un proyecto Spring Boot usando **Spring Initializr**:

curl https://start.spring.io/starter.zip -d dependencies=web,data-jpa,postgresql,lombok -d type=maven-project -o microservices.zip

unzip microservices.zip -d microservices

cd microservices

**🔹 2. Configurar application.properties**

Dentro de **src/main/resources/application.properties**, configura PostgreSQL:

properties

CopiarEditar

spring.datasource.url=jdbc:postgresql://localhost:5432/microservices\_db

spring.datasource.username=postgres

spring.datasource.password=yourpassword

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

server.port=8080

**🔹 3. Crear la entidad Persona**

Crea la carpeta **src/main/java/com/example/microservices/entity** y dentro el archivo **Persona.java**:

java

CopiarEditar

package com.example.microservices.entity;

import jakarta.persistence.\*;

import lombok.Data;

@Data

@MappedSuperclass

public class Persona {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

private String nombre;

private String genero;

private Integer edad;

private String identificacion;

private String direccion;

private String telefono;

}

**🔹 4. Crear la entidad Cliente**

Crea el archivo **Cliente.java** en la misma carpeta:

java

CopiarEditar

package com.example.microservices.entity;

import jakarta.persistence.\*;

import lombok.Data;

@Entity

@Data

public class Cliente extends Persona {

@Column(unique = true)

private String clienteId;

private String password;

private Boolean estado;

}

**🔹 5. Crear el repositorio**

Crea la carpeta **repository** y dentro el archivo **ClienteRepository.java**:

java

CopiarEditar

package com.example.microservices.repository;

import com.example.microservices.entity.Cliente;

import org.springframework.data.jpa.repository.JpaRepository;

public interface ClienteRepository extends JpaRepository<Cliente, Long> {

}

**🔹 6. Crear el controlador**

Crea la carpeta **controller** y dentro el archivo **ClienteController.java**:

java

CopiarEditar

package com.example.microservices.controller;

import com.example.microservices.entity.Cliente;

import com.example.microservices.repository.ClienteRepository;

import org.springframework.web.bind.annotation.\*;

import java.util.List;

@RestController

@RequestMapping("/clientes")

public class ClienteController {

private final ClienteRepository clienteRepository;

public ClienteController(ClienteRepository clienteRepository) {

this.clienteRepository = clienteRepository;

}

@GetMapping

public List<Cliente> getAllClientes() {

return clienteRepository.findAll();

}

@PostMapping

public Cliente createCliente(@RequestBody Cliente cliente) {

return clienteRepository.save(cliente);

}

}

**🔹 7. Crear el Dockerfile**

En la raíz del proyecto, crea un archivo llamado **Dockerfile**:

dockerfile

CopiarEditar

FROM openjdk:17

WORKDIR /app

COPY target/microservices-0.0.1-SNAPSHOT.jar app.jar

EXPOSE 8080

ENTRYPOINT ["java", "-jar", "app.jar"]

**🔹 8. Crear docker-compose.yml**

Para levantar PostgreSQL junto con la API, crea un archivo **docker-compose.yml**:

yaml

CopiarEditar

version: "3.8"

services:

postgres:

image: postgres

container\_name: postgres\_db

restart: always

environment:

POSTGRES\_DB: microservices\_db

POSTGRES\_USER: postgres

POSTGRES\_PASSWORD: yourpassword

ports:

- "5432:5432"

microservices:

build: .

depends\_on:

- postgres

ports:

- "8080:8080"

environment:

SPRING\_DATASOURCE\_URL: jdbc:postgresql://postgres:5432/microservices\_db

SPRING\_DATASOURCE\_USERNAME: postgres

SPRING\_DATASOURCE\_PASSWORD: yourpassword

**🚀 9. Levantar el proyecto**

1. **Compilar el proyecto**:

sh

CopiarEditar

mvn clean package

1. **Construir la imagen Docker**:

docker-compose down -v

docker compose down

docker system prune -a

docker-compose up --build

1. **Verificar en Postman**:
   * **GET** http://localhost:8080/clientes
   * **POST** http://localhost:8080/clientes con un JSON como:

json

CopiarEditar

{

"nombre": "Jose Lema",

"genero": "Masculino",

"edad": 30,

"identificacion": "1234567890",

"direccion": "Otavalo sn y principal",

"telefono": "098254785",

"clienteId": "jose123",

"password": "1234",

"estado": true

}

**🎯 Próximos pasos**

✅ **Añadir Cuenta y Movimientos**  
✅ **Mensajería asíncrona con RabbitMQ**  
✅ **Pruebas unitarias e integración con JUnit y Mockito**

¿Quieres que sigamos con la parte de Cuenta o Movimientos?