Integrantes:

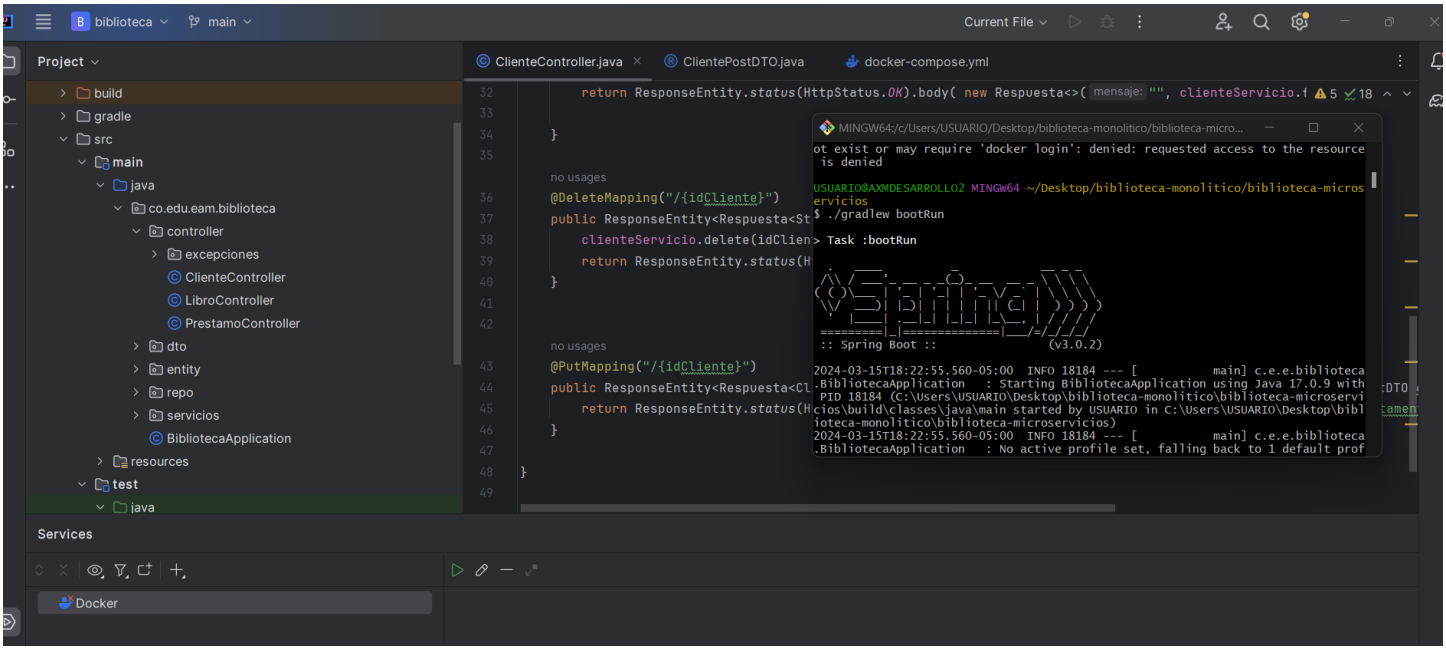
- Santiago Bernal López

- Luisa Fernanda Celis Gonzalez

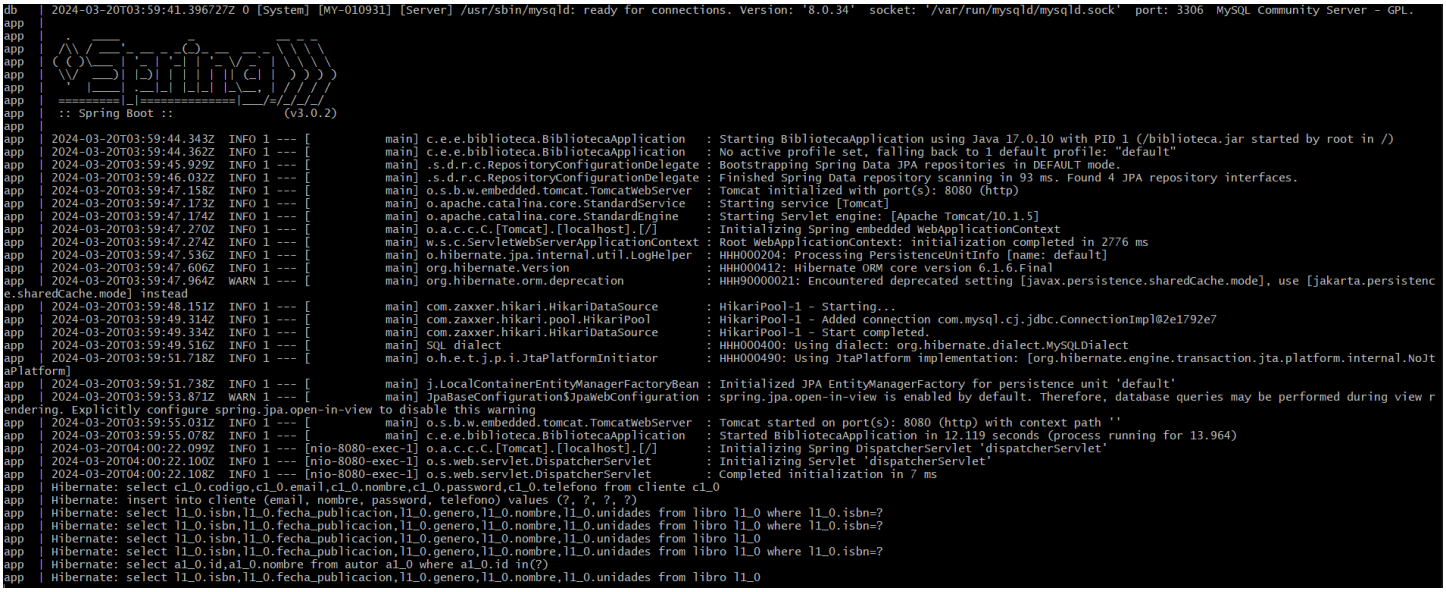
- Andrés Mauricio Ocampo Arenas

Github Repositorio -> [Z4ND3RX/Microservices-practice (github.com)]

PROYECTO MONOLITICO – BIBLIOTECA

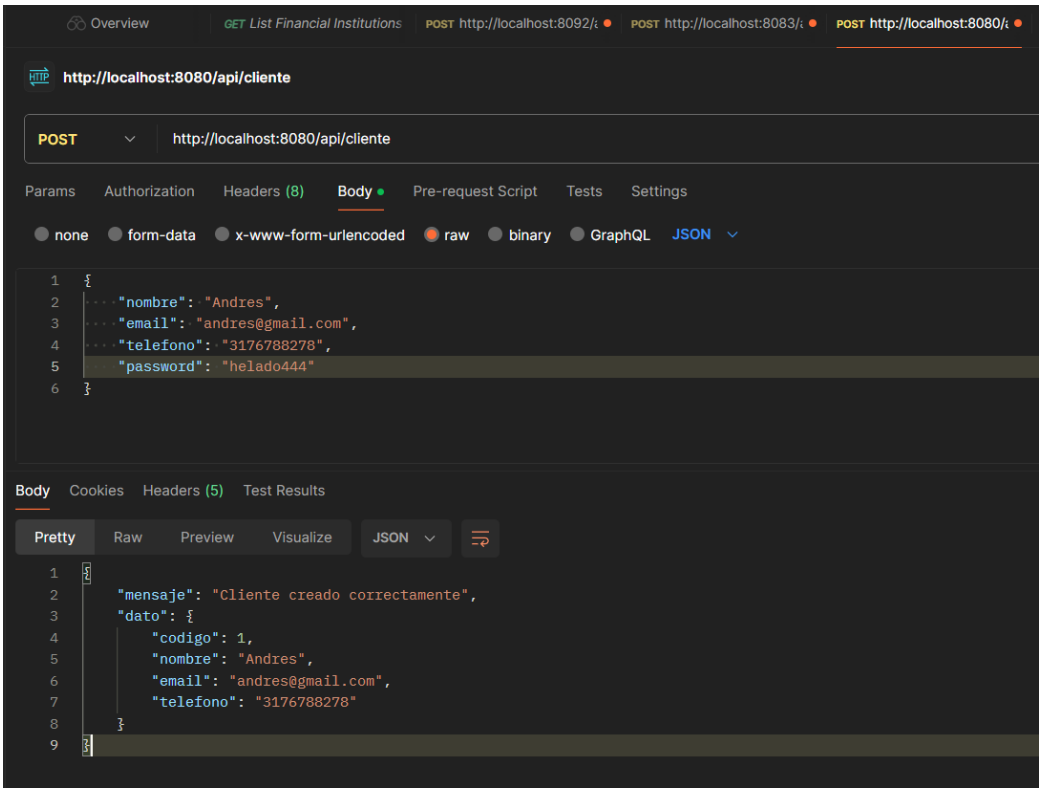


Docker-compose up

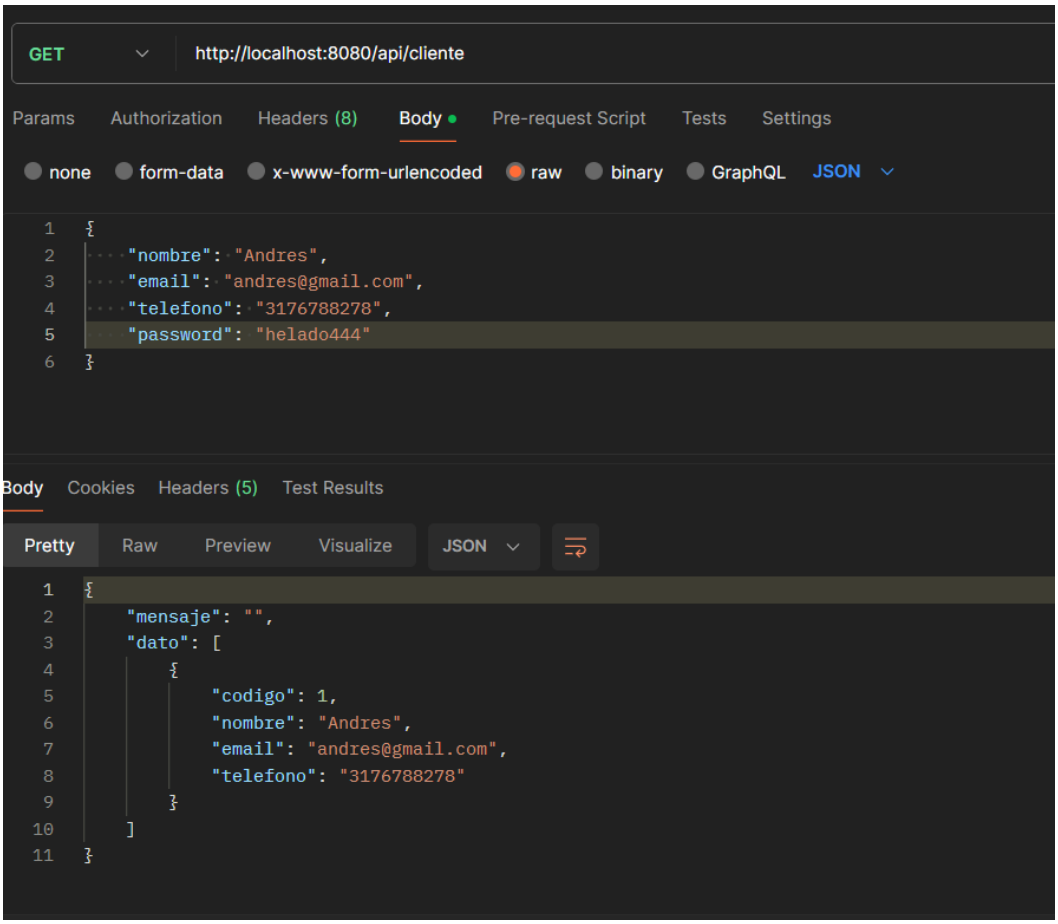


CLIENT – PRUEBAS HTTP

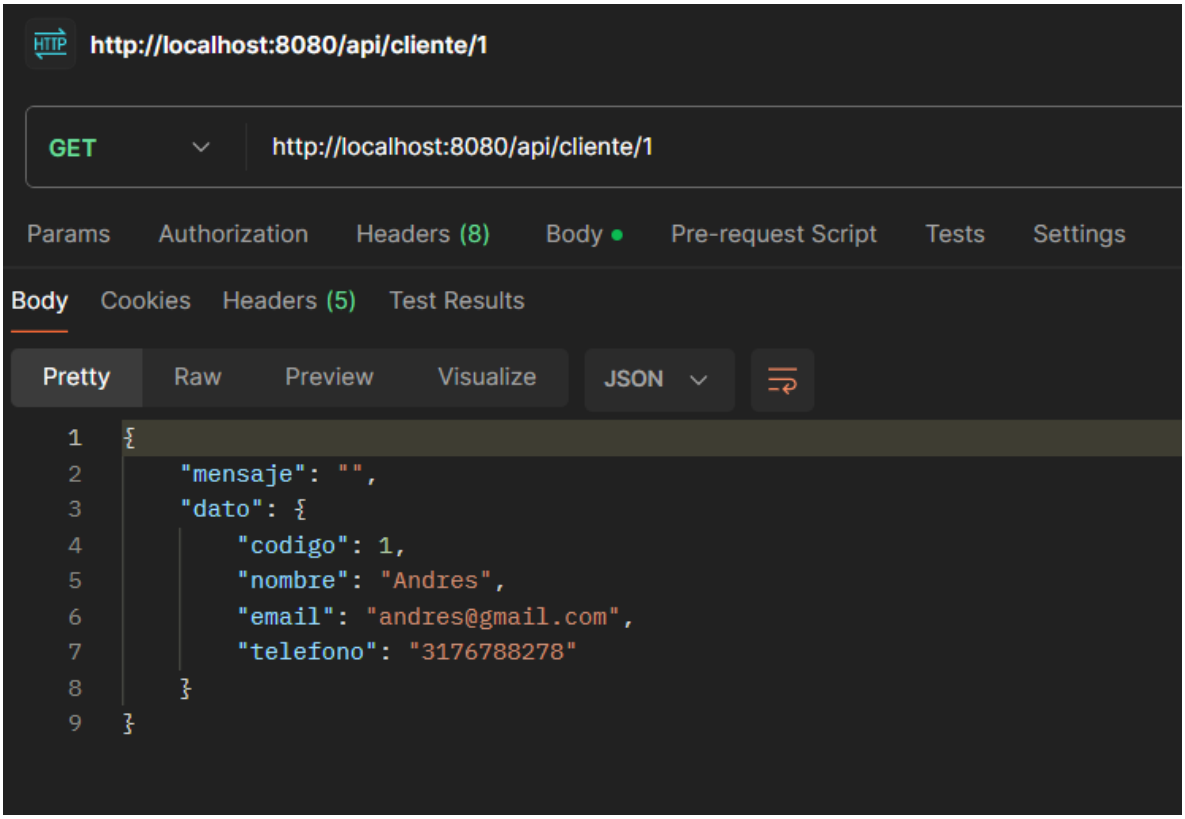
MÉTODO POST



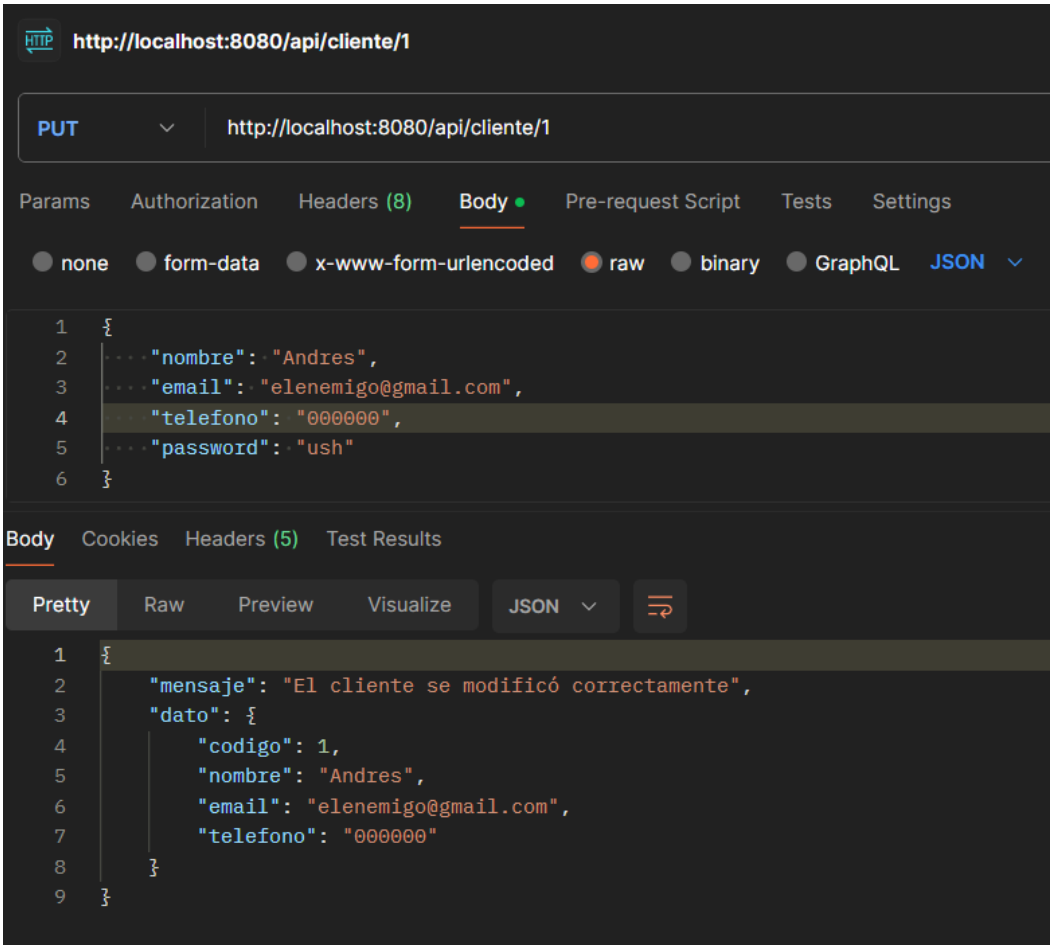
MÉTODO GET



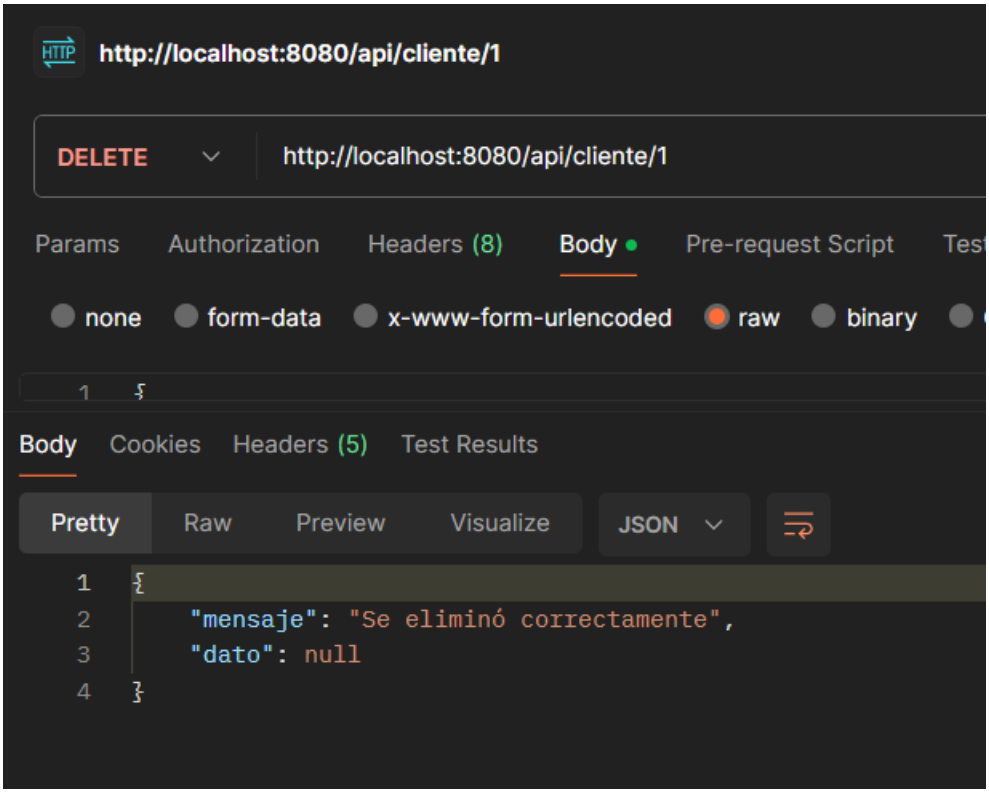
MÉTODO GET BY ID



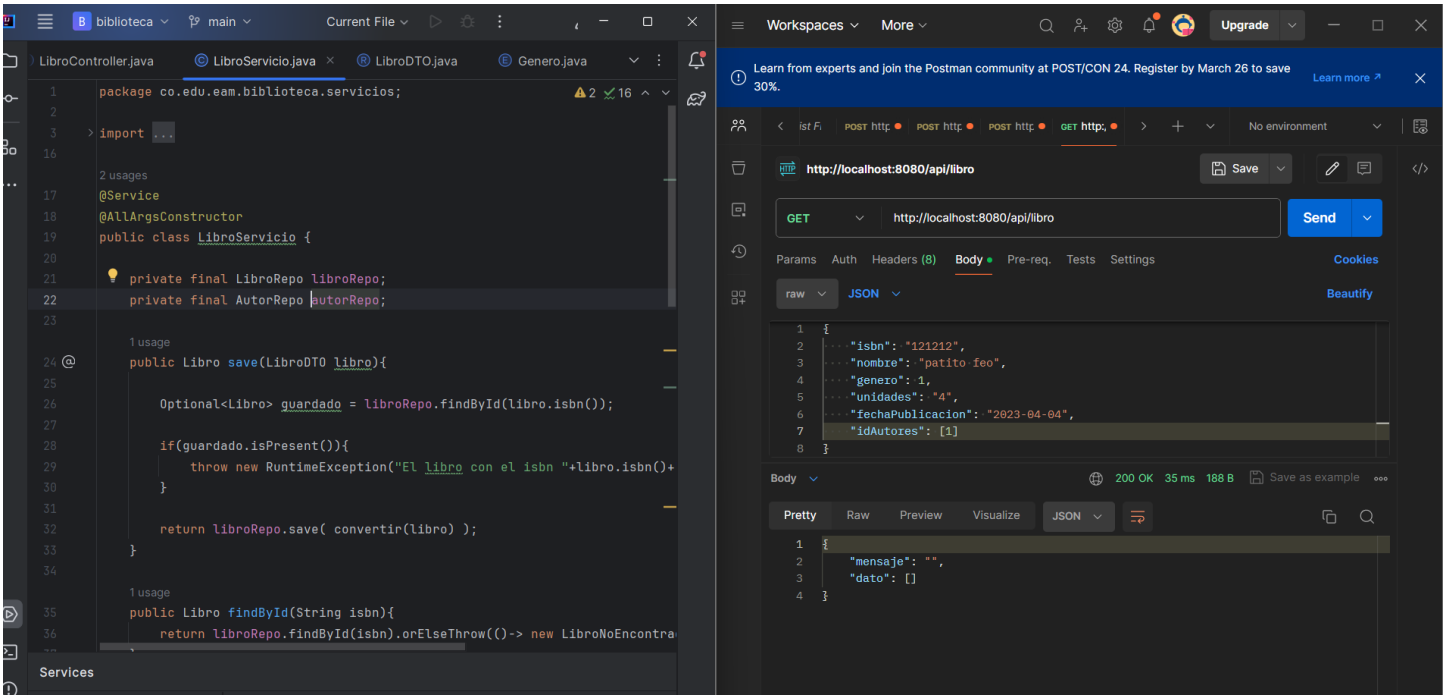
MÉTODO PUT

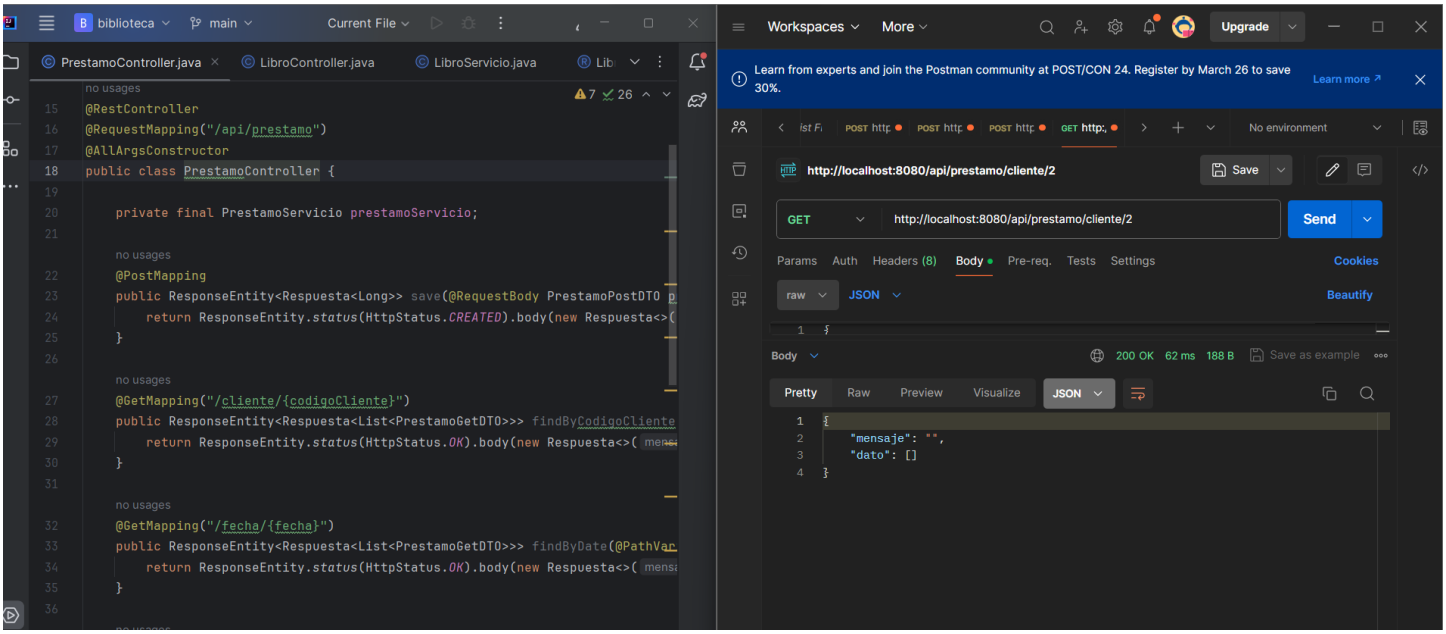


MÉTODO DELETE



LIBROS Y PRESTAMOS FUNCIONAN DE MANERA SIMILAR





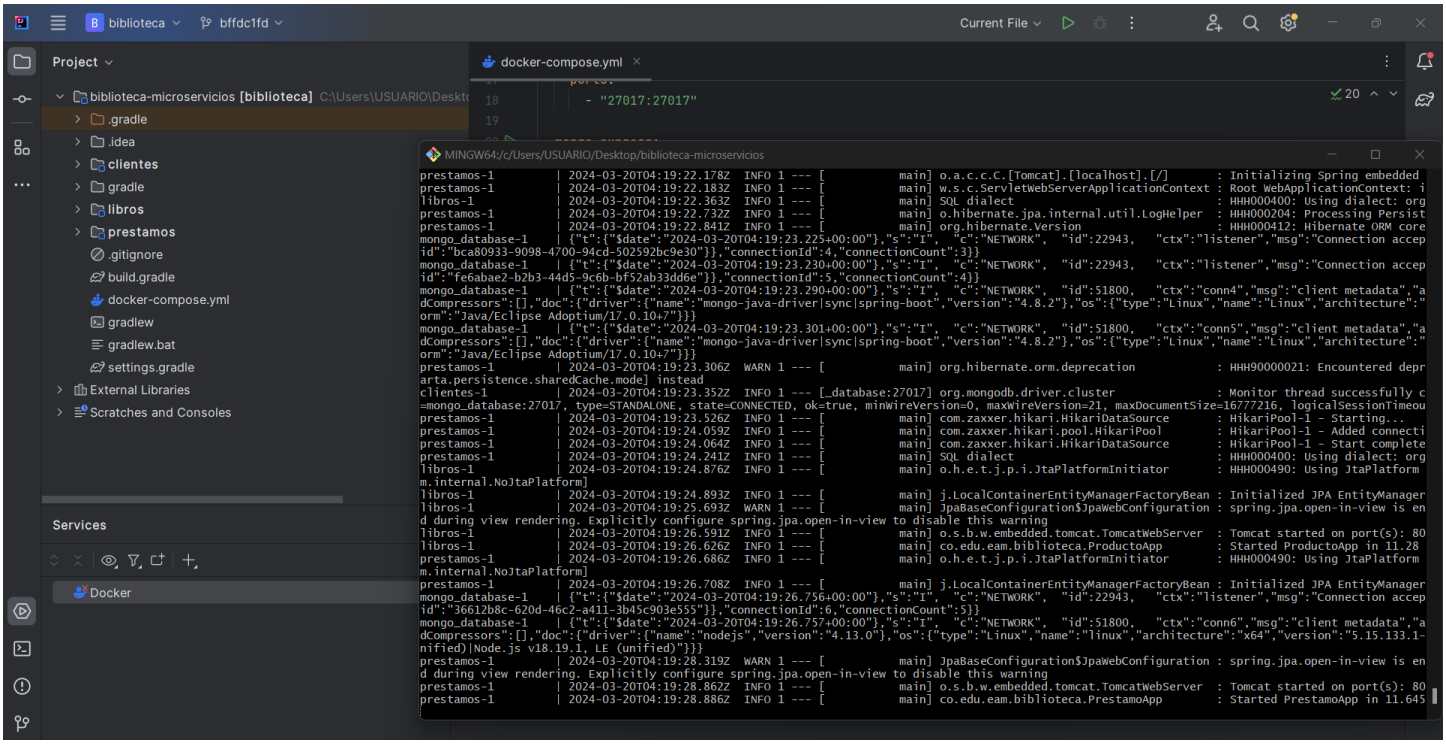
2. PROYECTO MICROSERVICIOS – BIBLIOTECA

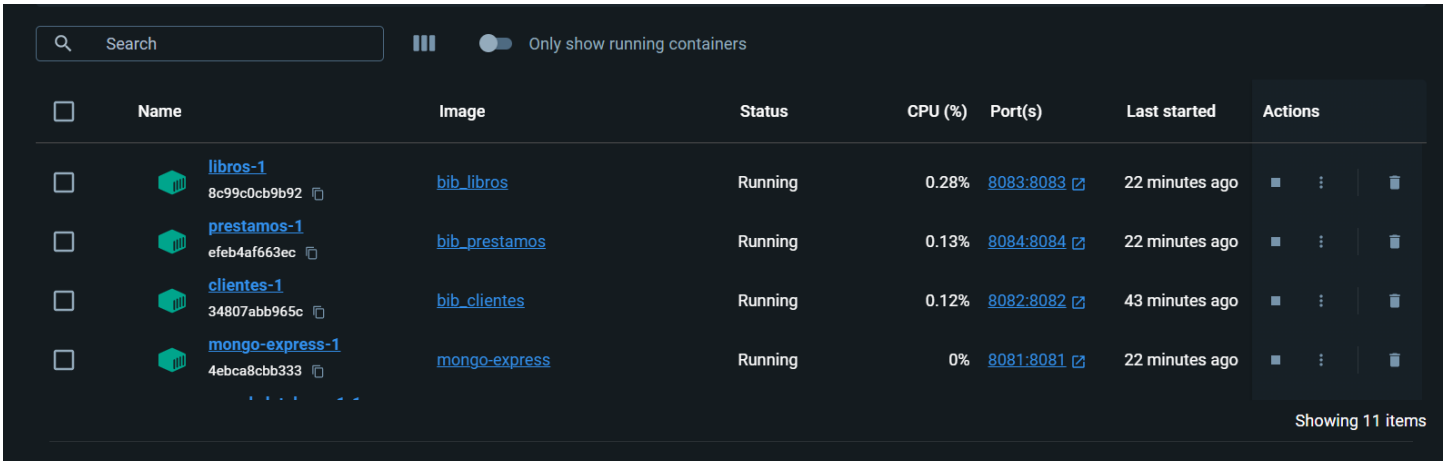
Realizamos unos cuantos ajustes en el Docker compose como los puertos:

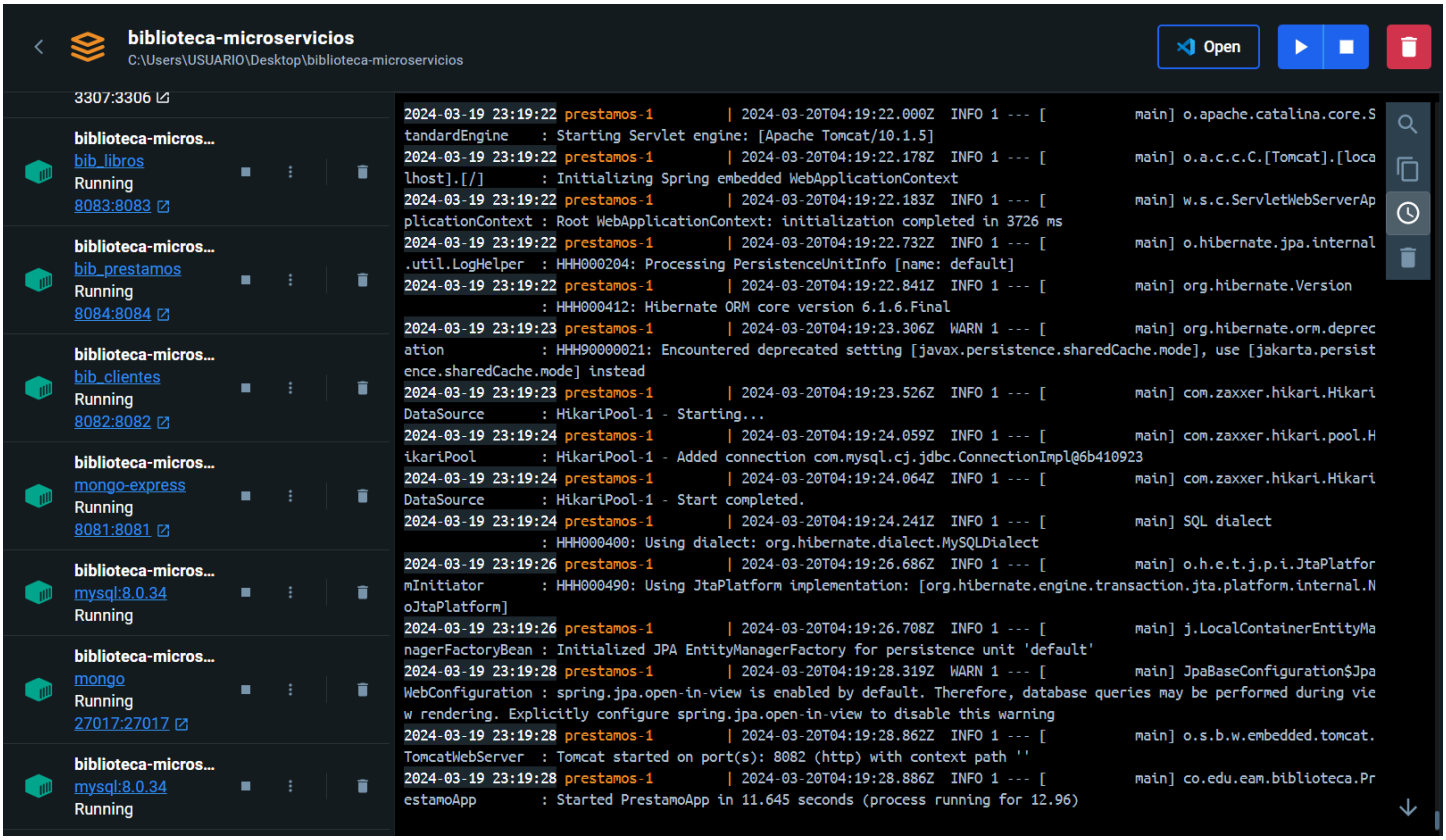
```yaml
  clientes:
    image: bib_clientes
    build: ./clientes
    restart: always
    ports:
      - "8082:8082"
    depends_on:
      - mongo_database
    environment:
      spring.data.mongodb.uri: mongodb://root:example@mongo_database:27017/clientes?authSource=admin

  libros:
    image: bib_libros
    build: ./libros
    restart: always
    ports:
      - "8083:8083"
    depends_on:
      - mysql_database_1
    environment:
      spring.datasource.url: jdbc:mysql://mysql_database_1:3306/libros?createDatabaseIfNotExist=true
      spring.datasource.username: root
      spring.datasource.password: root1234
```

Abrimos y corremos el proyecto

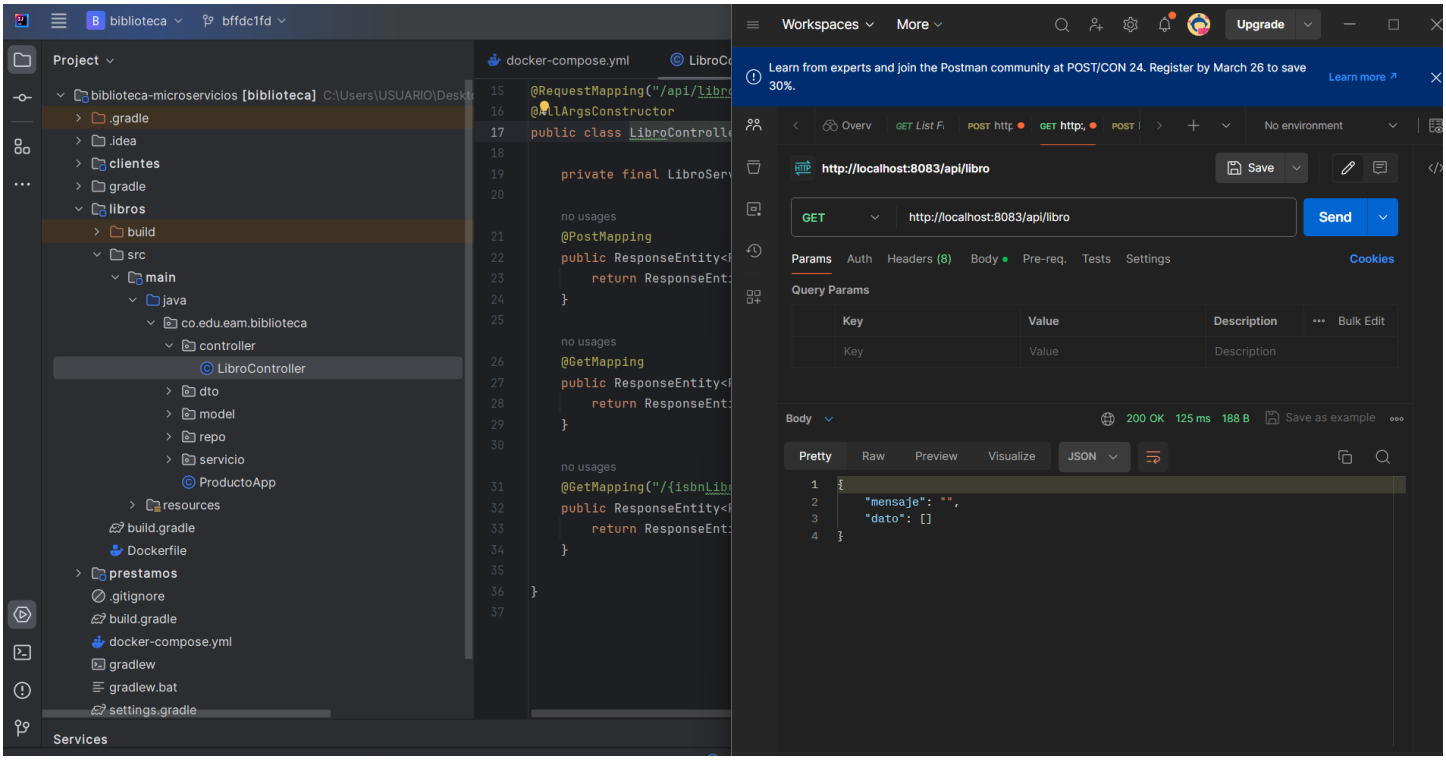




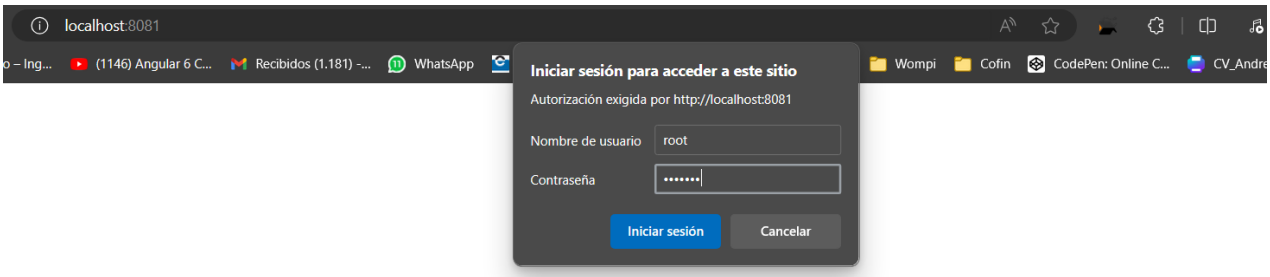


Mediante la estructura y las pruebas, identificamos que es el mismo proyecto, solo que en microservicios

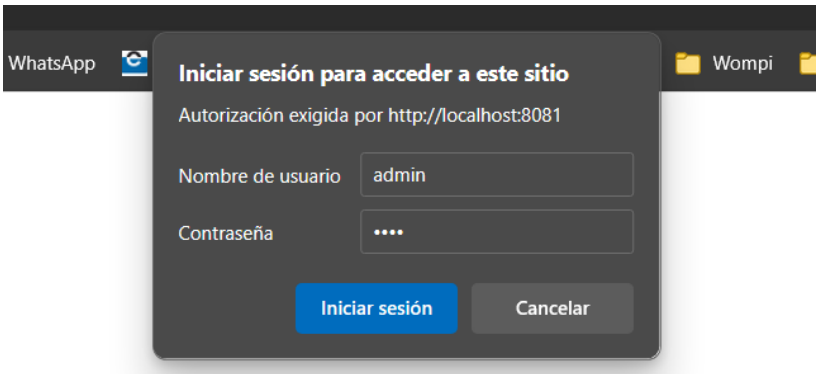
GET LIBROS



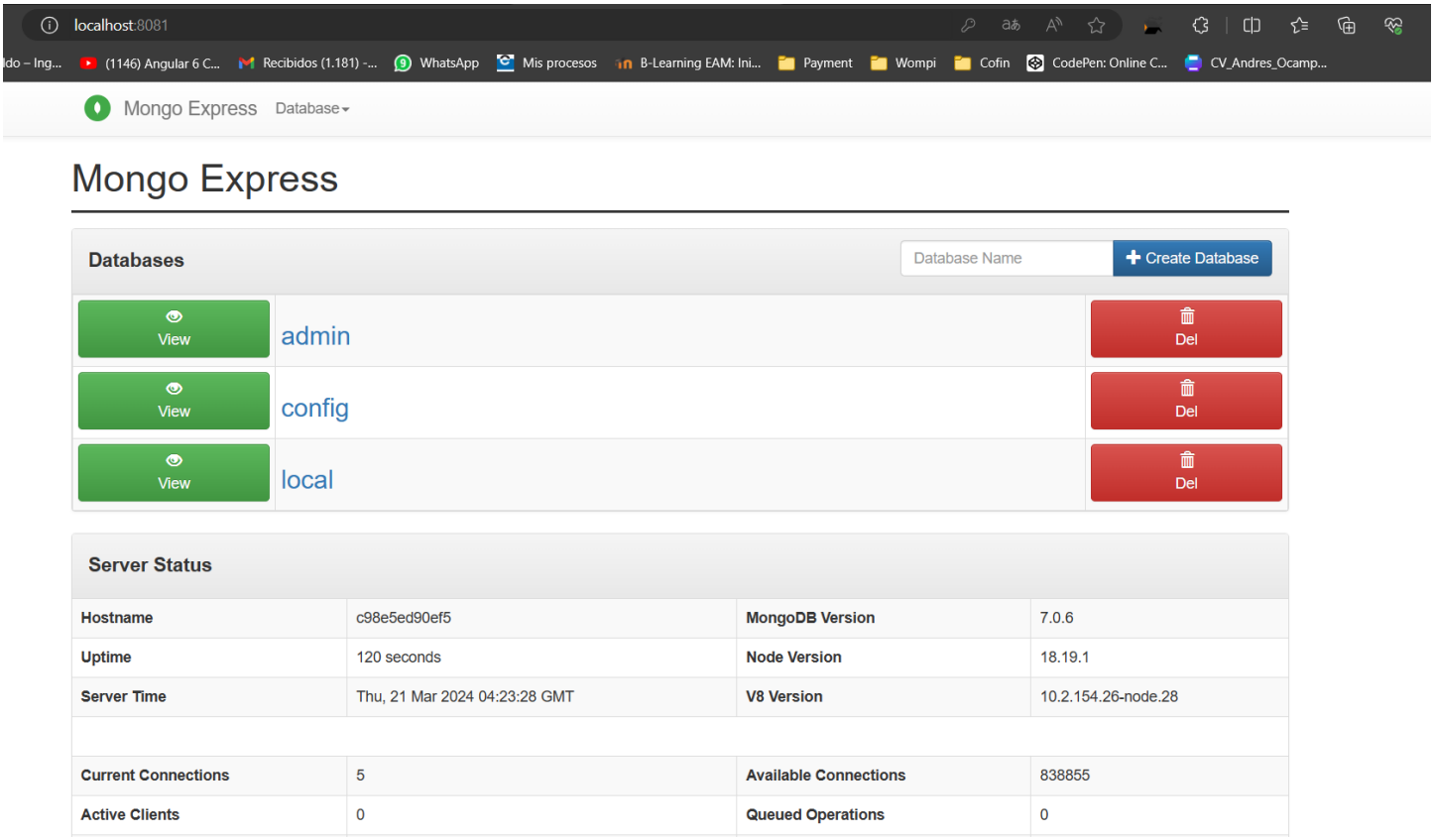
Base de datos



Bueno, no es, pero esta si es 😊



Allí, encontramos el panel de administración de mongo express

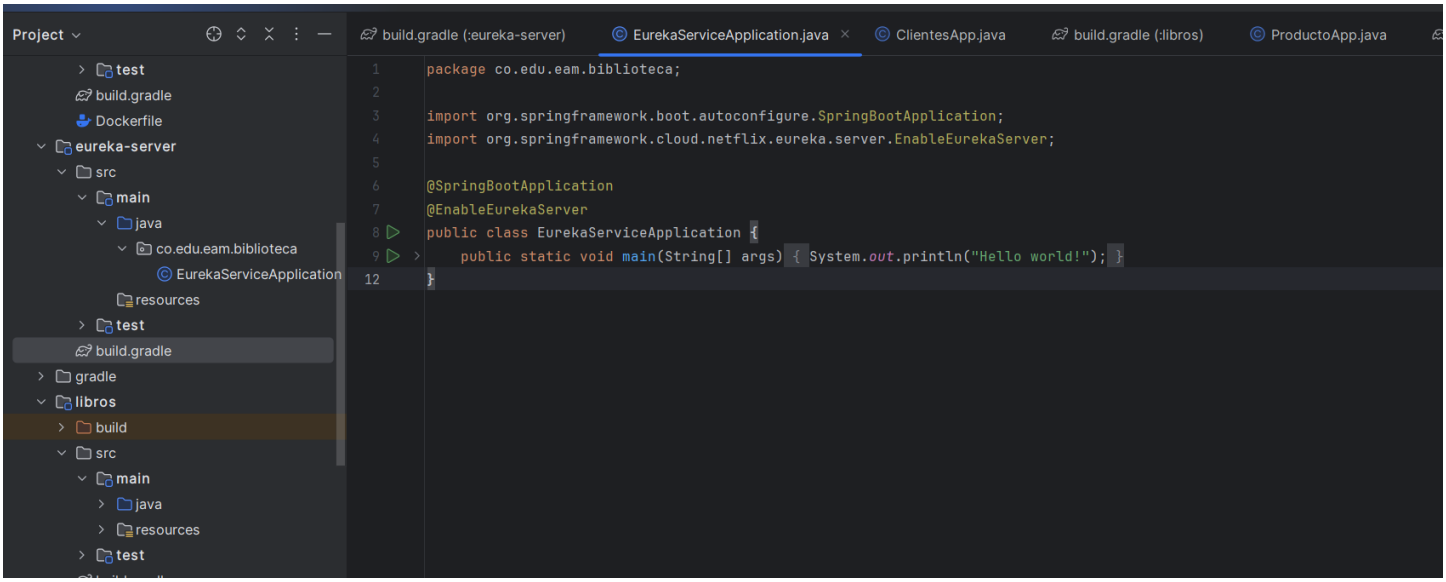


RestTemplate es una clase proporcionada por el framework Spring para realizar peticiones HTTP a servicios RESTful. Facilita la comunicación entre aplicaciones Java y servicios web que siguen el estilo de arquitectura REST.

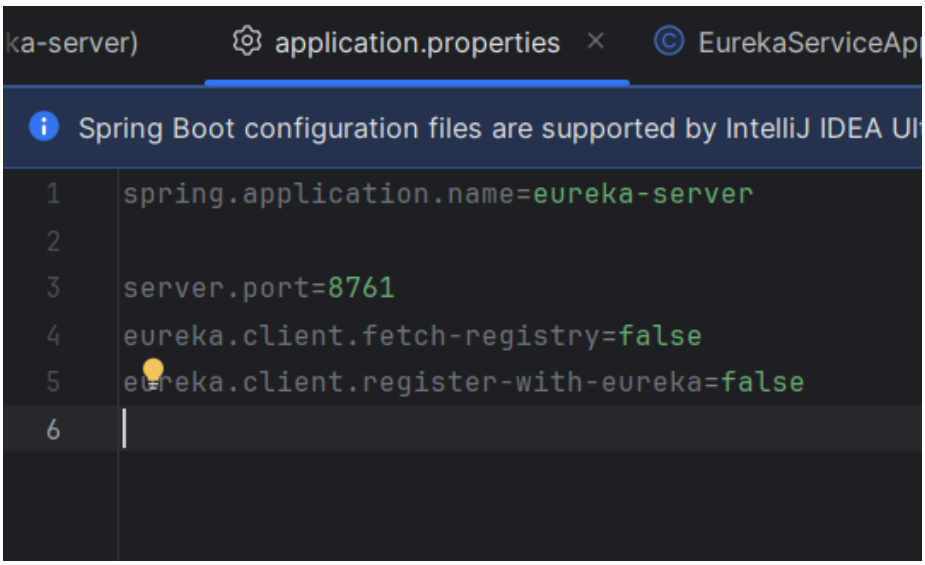
Con RestTemplate, puedes enviar solicitudes HTTP GET, POST, PUT, DELETE, y procesar las respuestas recibidas. Proporciona métodos convenientes para manejar los diferentes tipos de datos que se pueden enviar y recibir en una solicitud HTTP, como JSON, XML, texto simple, etc.

Fuente - Tutorial: [A Guide to the RestTemplate | Baeldung]

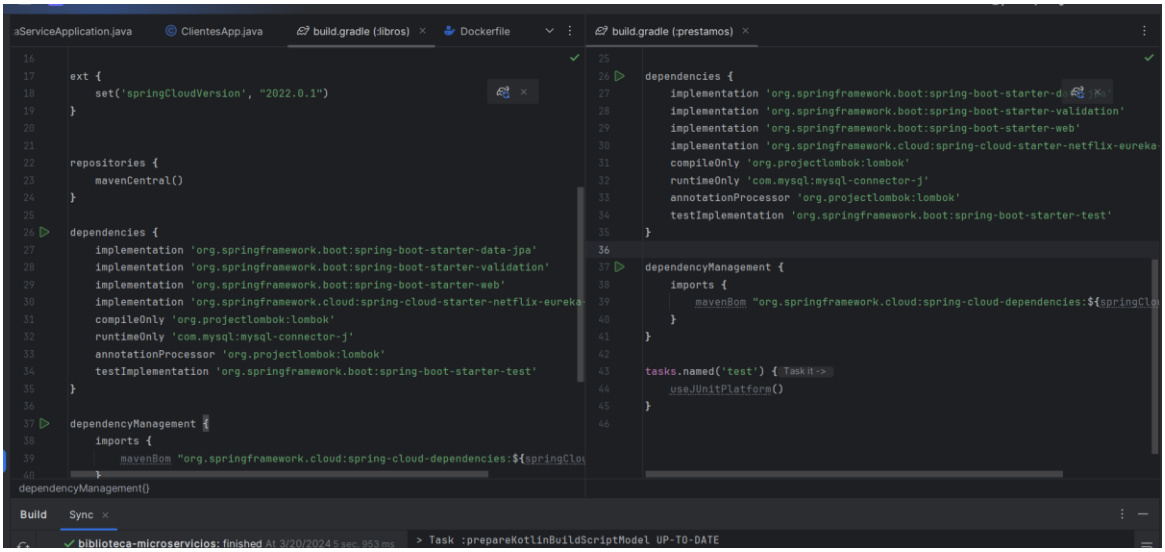
EUREKA-



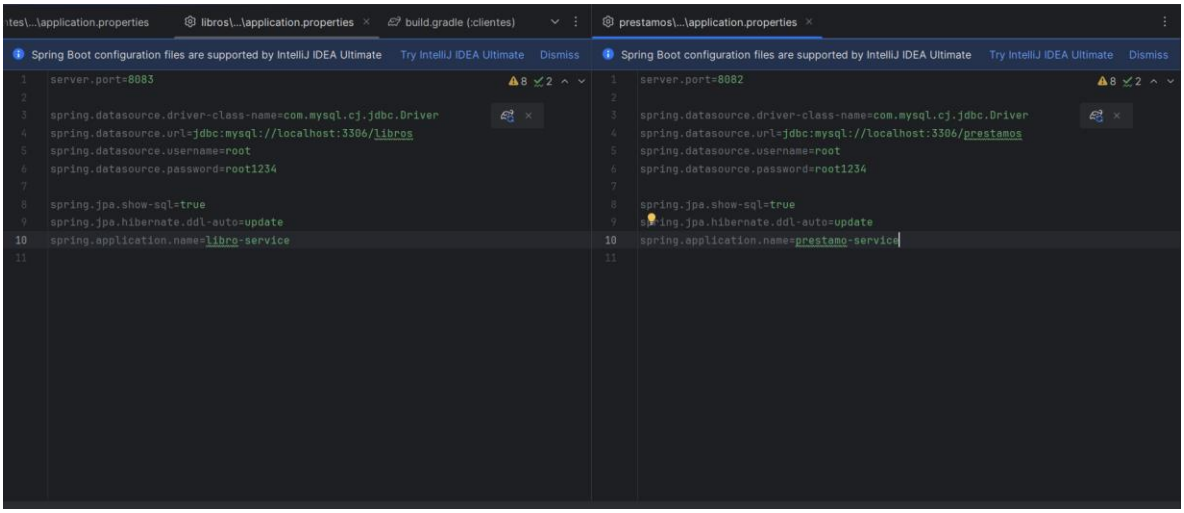
Application.properties:



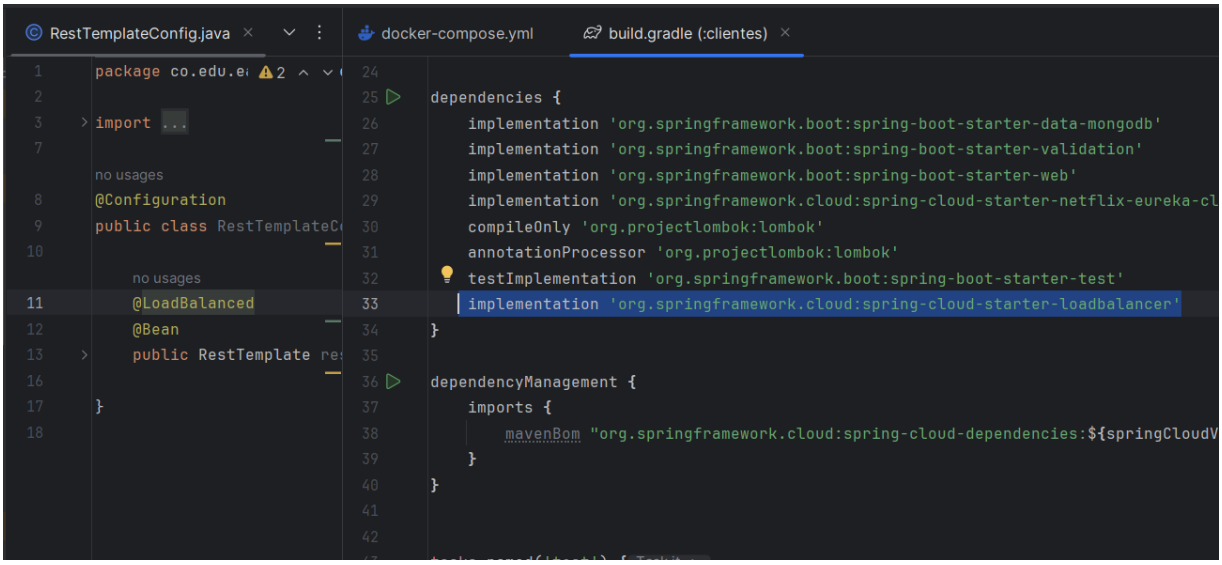
Ext, dependency and management dependency para eureka añadido en los otros microservicios



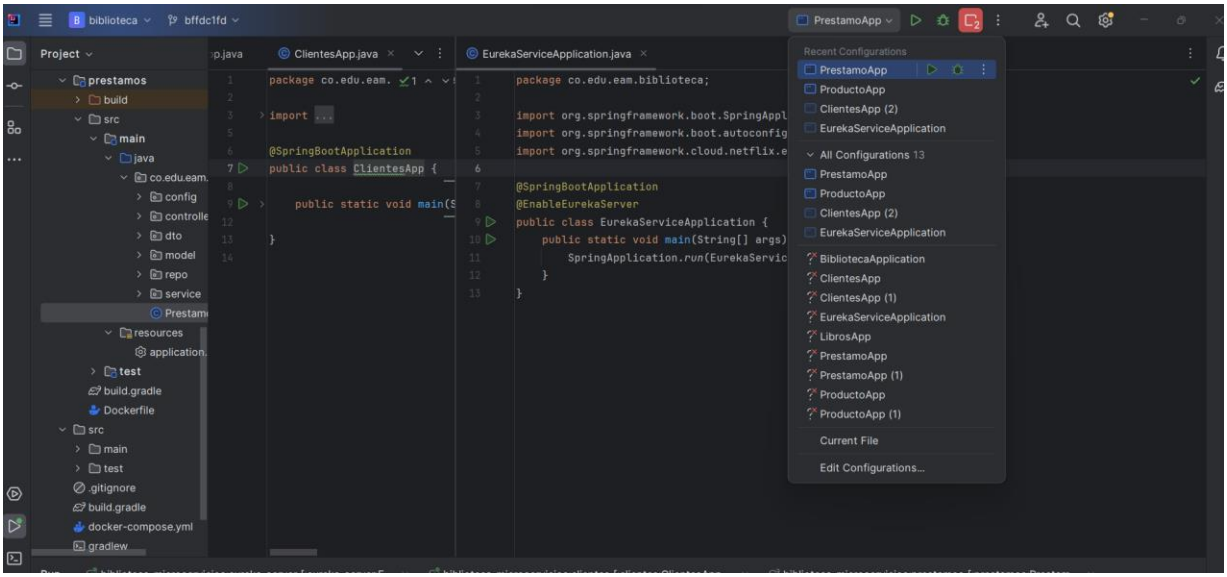
Archivos de properties actualizados



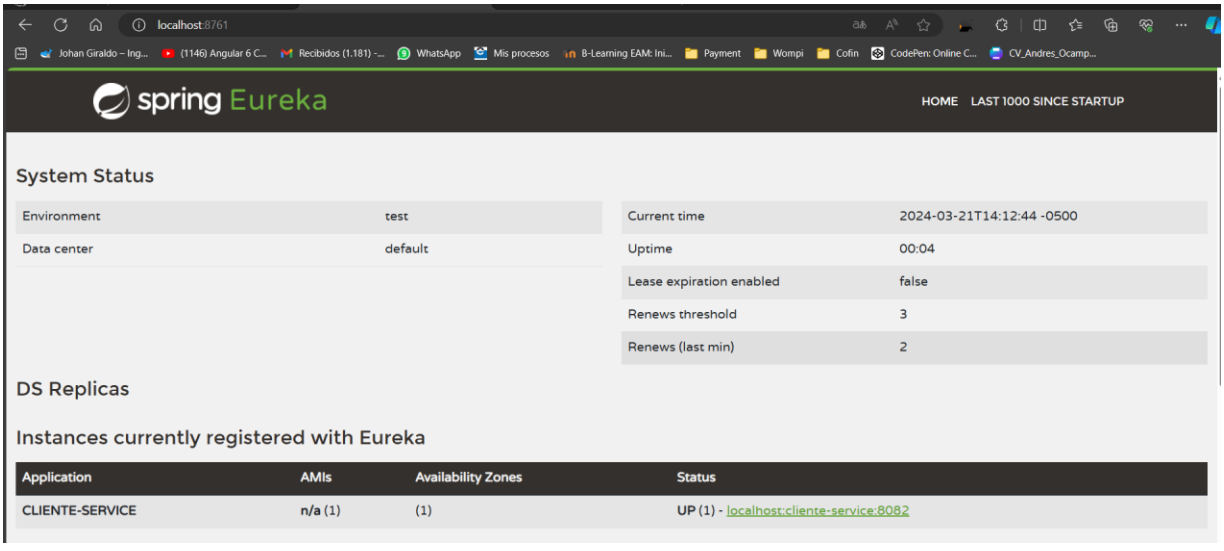
Añadimos la dependencia al módulo y la etiqueta @LoadBalanced al método de RestTemplate



Corremos Eureka y los otros servicios uno por uno

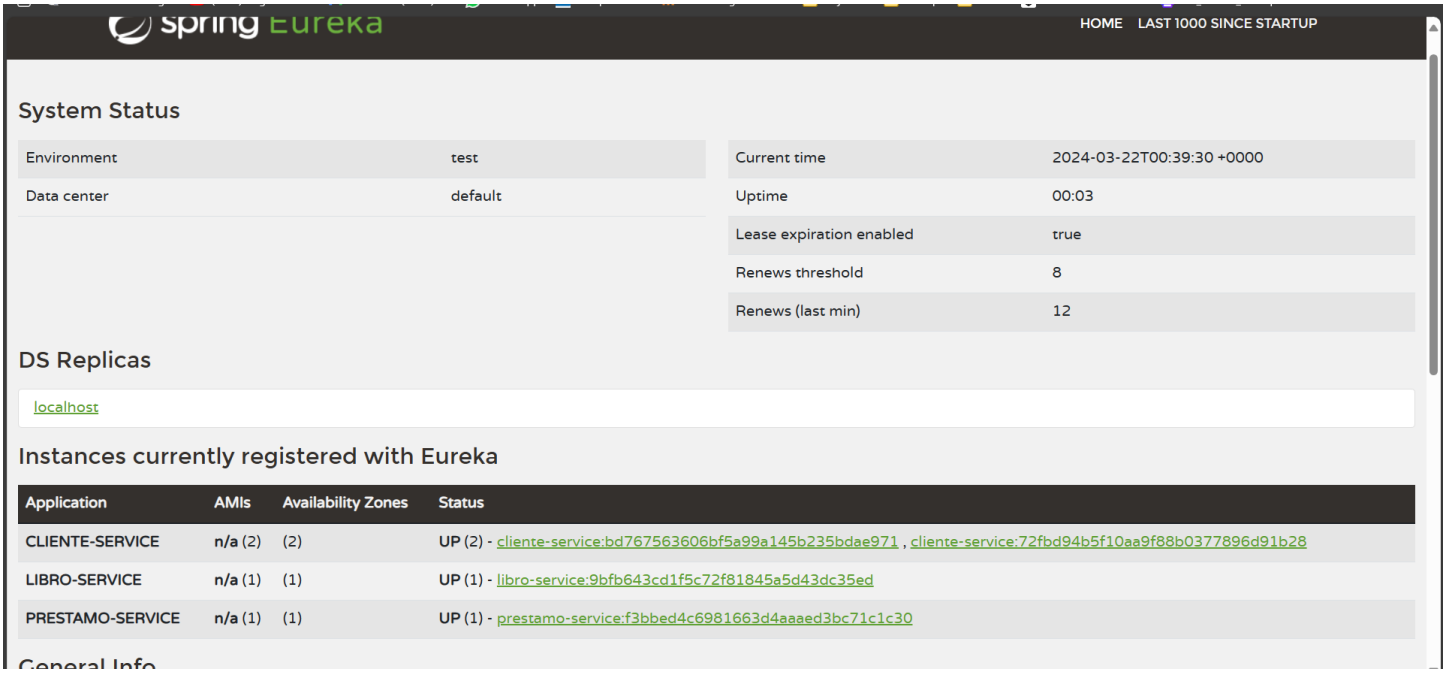


Desde Eureka nos empiezan a aparecer uno por uno los servicios que hemos corrido

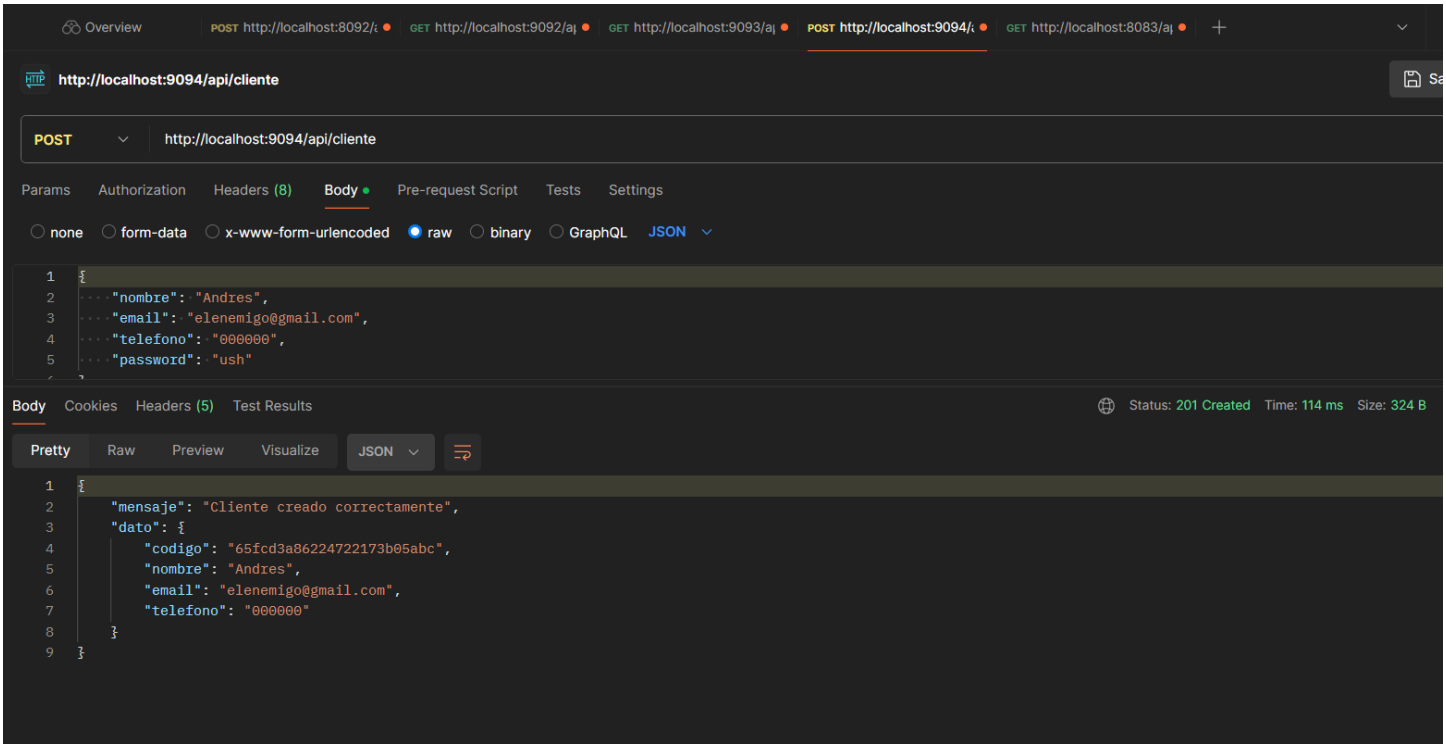


Ahora ponemos Eureka en el docker-compose, editamos los archivos.properties, compilamos cada microservicio y levantamos el proyecto con el Docker-compose...

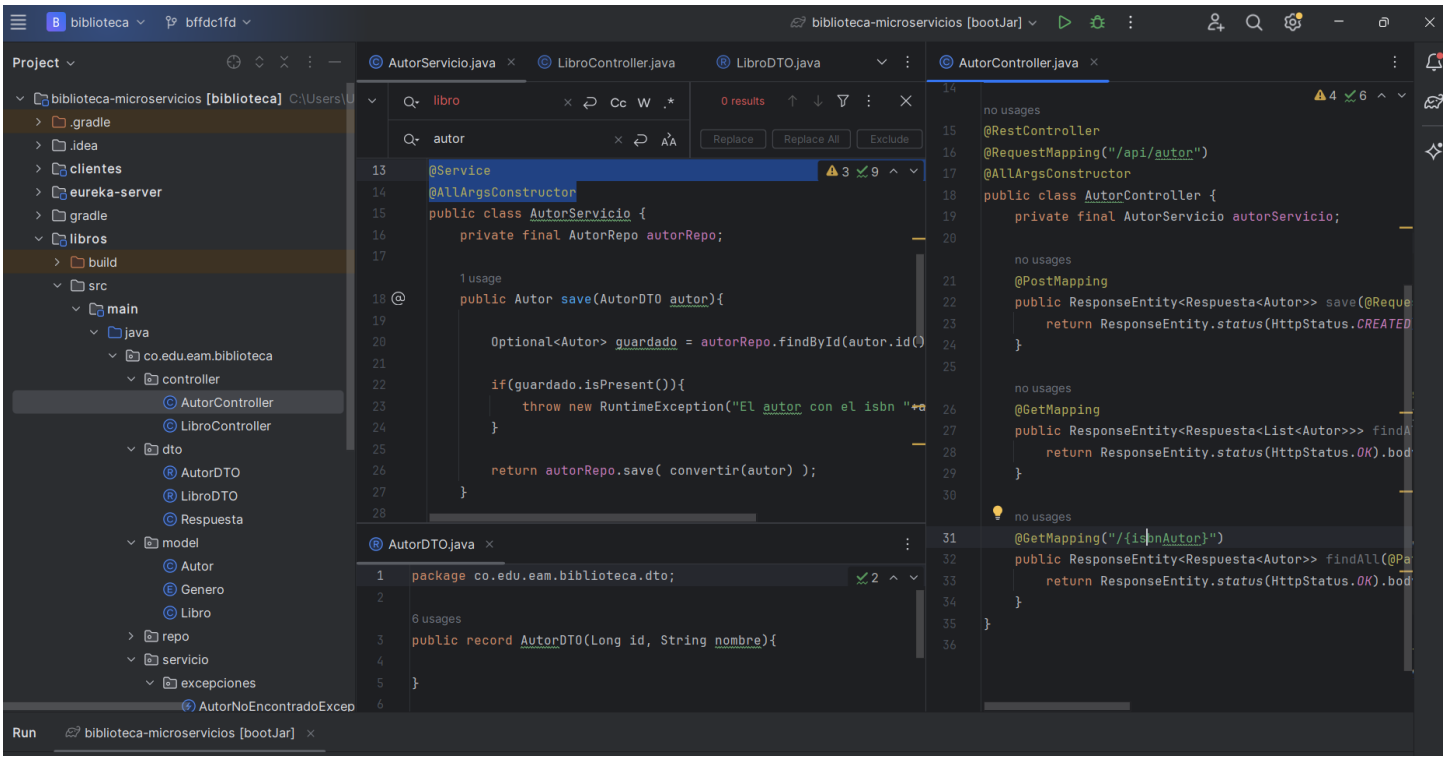
Si ingresamos a Eureka, deberán aparecer los servicios como se aprecian en la siguiente screenshot:



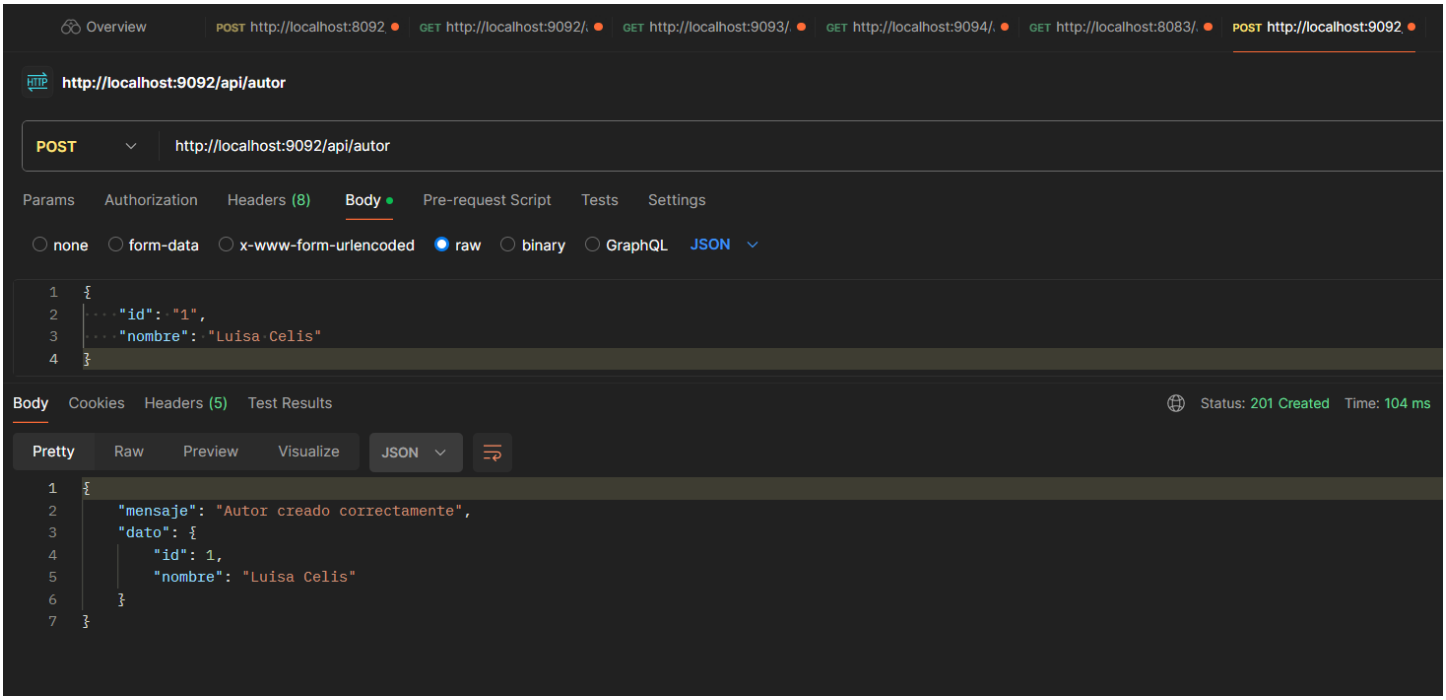
Ahora bien, probemos 😊



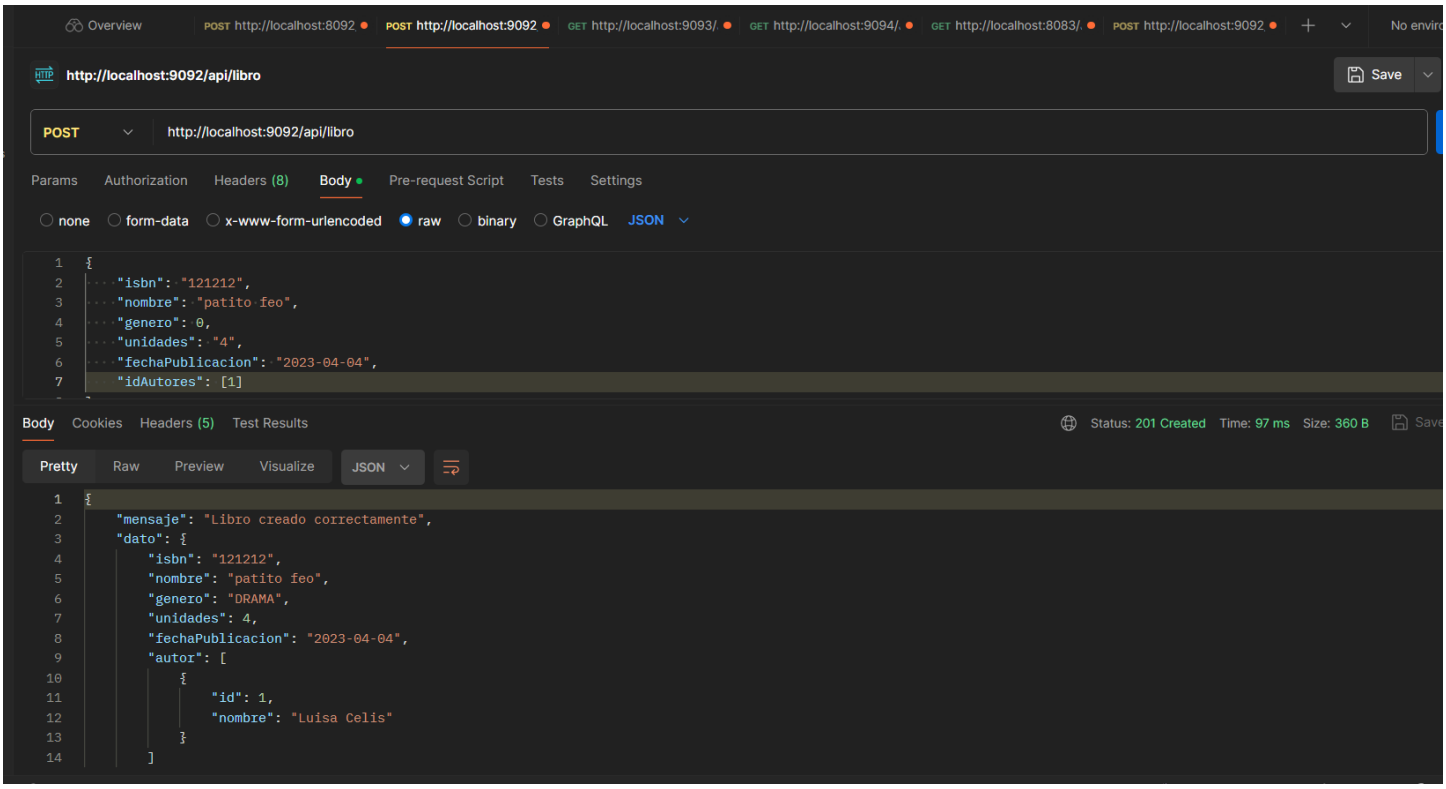
Creamos el DTO, Controlador y Servicio para Autores



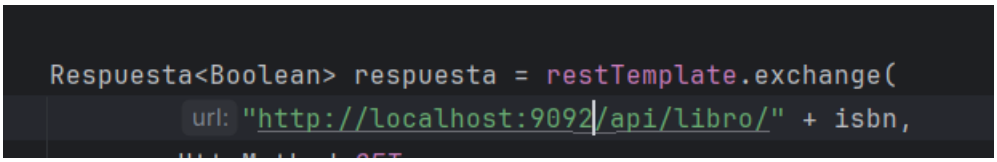
Probamos con Postman



Creamos un libro



Para prestamos ajustamos nuestras urls



Y con esto podemos crear los préstamos y todo lo que queramos 😊 FIIIIIIIIN