**CORPORACION UNIVERSITARIA REMINGTON**

Asignatura: Lenguaje de programación avanzado 2

Parcial 2

Presentado por:

Gloria Patricia Cardona Cuervo

Presentado a:

Profesora Martha Nicolasa Amaya Becerra

Ingeniería de Sistemas - Virtual

abril de 2025

Contenido

[Arquitectura del proyecto 3](#_Toc195378246)

[Despliegue de contenedor PostgreSQL en Windows 4](#_Toc195378247)

[Microservicio para Eureka Server 8](#_Toc195378248)

[Microservicio de Usuarios 9](#_Toc195378249)

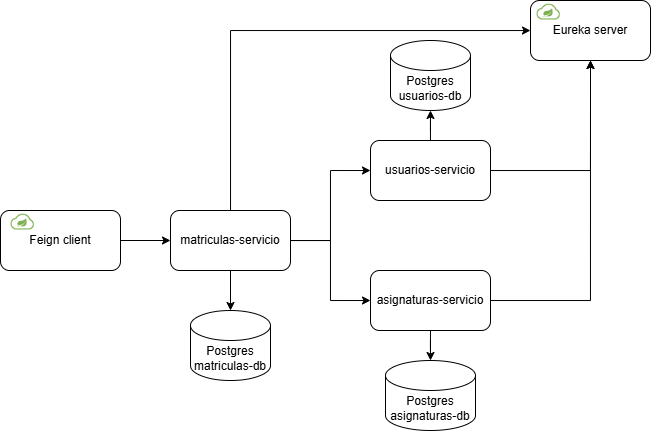
[Microservicio de Asignaturas 13](#_Toc195378250)

[Microservicio de Matrículas 19](#_Toc195378251)

Enlace al repositorio:

https://github.com/gpcardonac/sistema-educativo-microservicios-nombre

# Arquitectura del proyecto



# Despliegue de contenedor PostgreSQL en Windows

Para desplegar un contenedor de PostgreSQL en un entorno Windows utilizando Docker, se llevaron a cabo los siguientes pasos:

**Instalación de Docker Desktop**

Se accedió al sitio oficial de Docker (<https://www.docker.com/products/docker-desktop>) y se descargó Docker Desktop para Windows. Posteriormente, se realizó la instalación y se verificó que el servicio estuviera activo mediante el icono de la ballena visible en la barra de tareas.

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

Imagen que contiene Tabla

El contenido generado por IA puede ser incorrecto.

Se debió actualizar el subsistema de Windows par Linux:

Texto

El contenido generado por IA puede ser incorrecto.

**Ejecución del contenedor de PostgreSQL**

Se abrió una terminal y se ejecutó el siguiente comando:

docker run --name postgres-container -e POSTGRES\_USER=postgres -e POSTGRES\_PASSWORD=postgres -p 5434:5432 -d postgres

Este comando realizó lo siguiente:

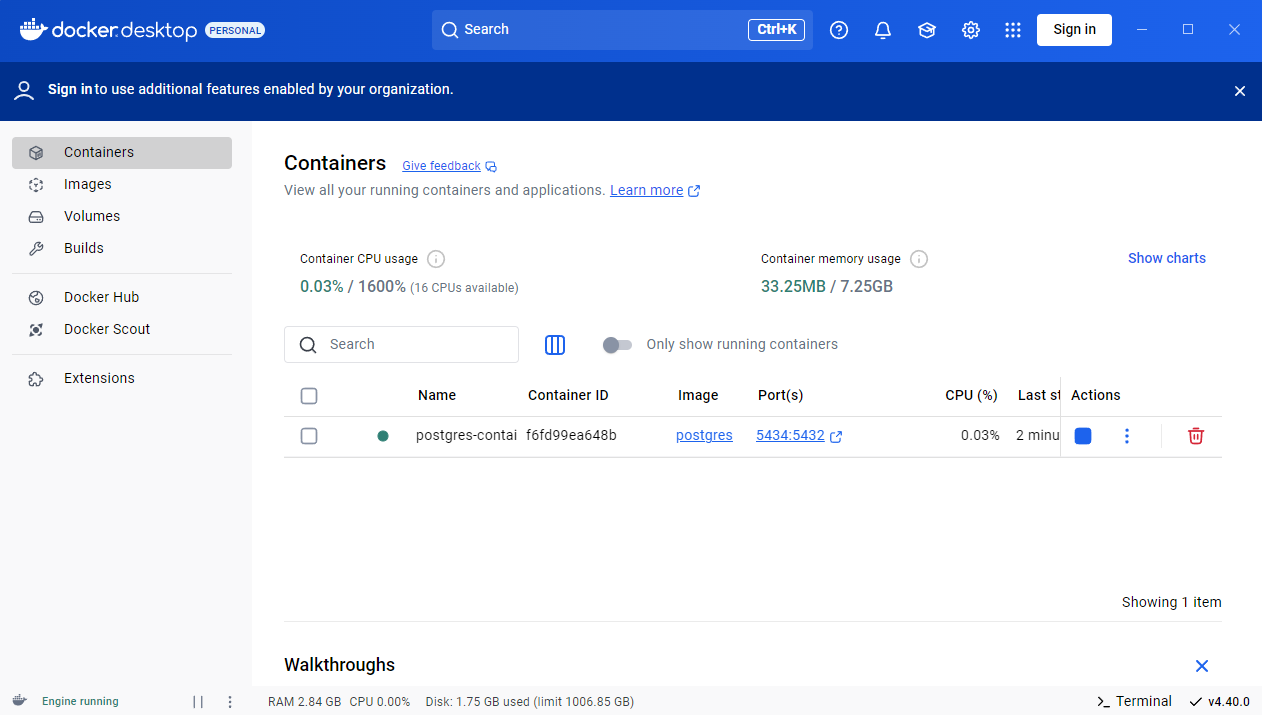
* Asignó el nombre postgres-container al contenedor.
* Estableció tanto el usuario como la contraseña como postgres.
* Expuso el puerto 5432 del contenedor al puerto 5434 del host.
* Ejecutó el contenedor en segundo plano (-d) usando la imagen oficial de PostgreSQL.

**Verificación del estado del contenedor**

Se utilizó el siguiente comando para verificar que el contenedor estuviera corriendo:

docker ps

La salida mostró una fila correspondiente al contenedor postgres, confirmando que se encontraba activo y escuchando en el puerto 5434.



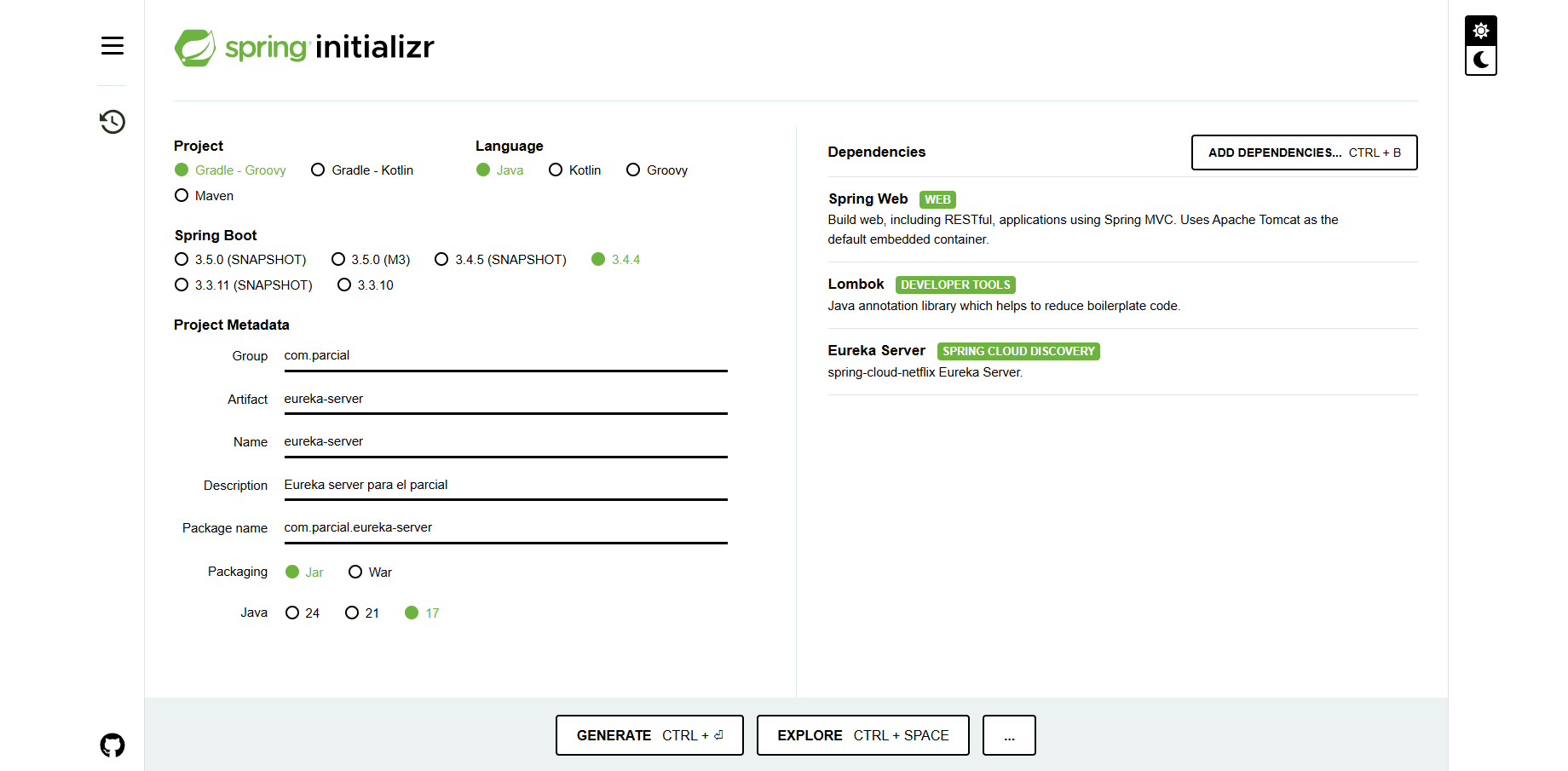
* El comando docker run se empleó **solo la primera vez**, ya que crea y lanza un nuevo contenedor. Una vez creado, **no fue necesario volver a usarlo** para iniciar el contenedor.
* Para usos posteriores, se utilizó el comando:
* docker start postgres-container

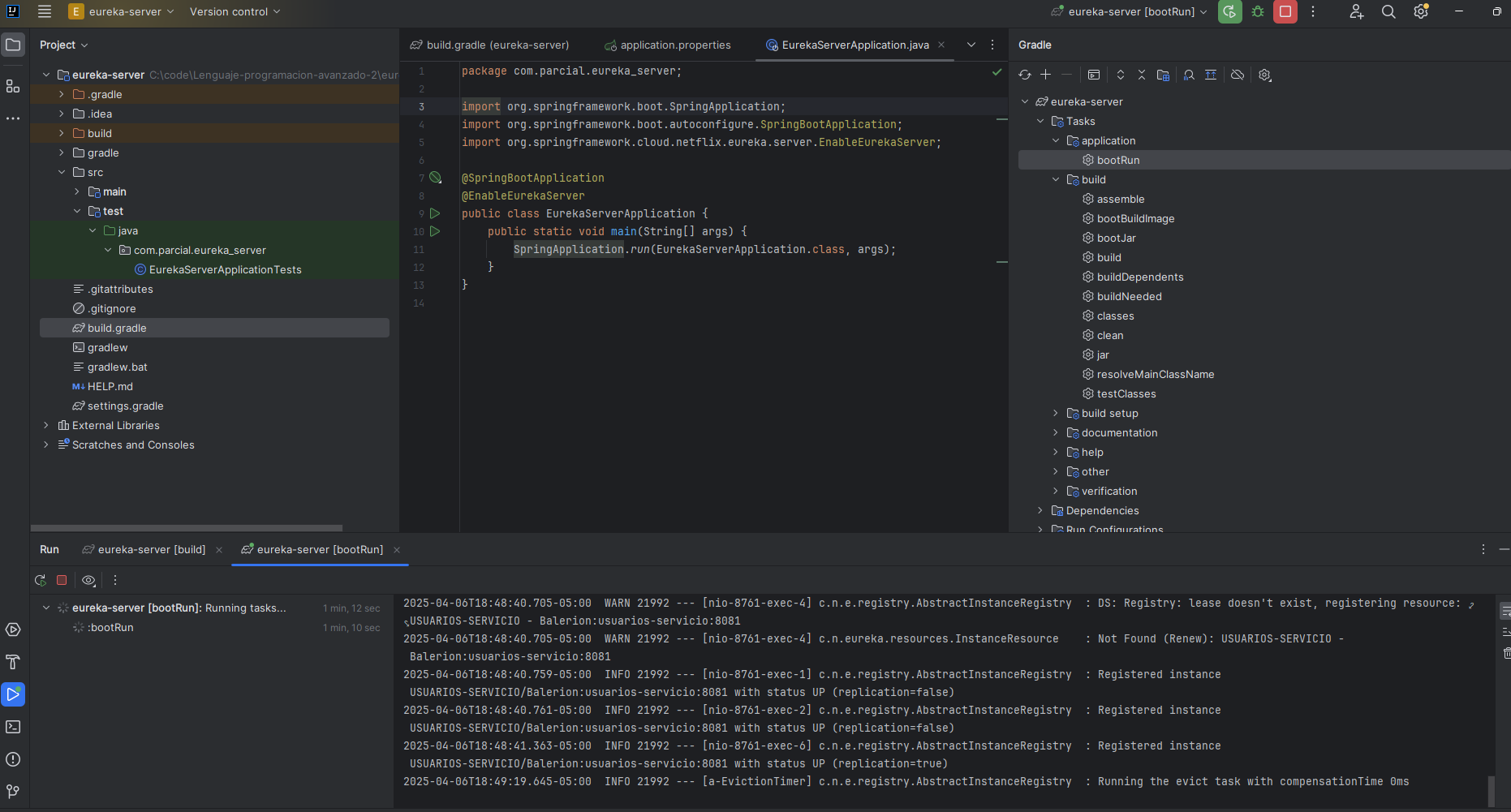
Y para detener el contenedor:

docker stop postgres-container

* Se tuvo en cuenta que intentar ejecutar nuevamente docker run con el mismo nombre (postgres-container) provocaría un error como el siguiente:
* docker: Error response from daemon: Conflict. The container name "/postgres-container" is already in use.

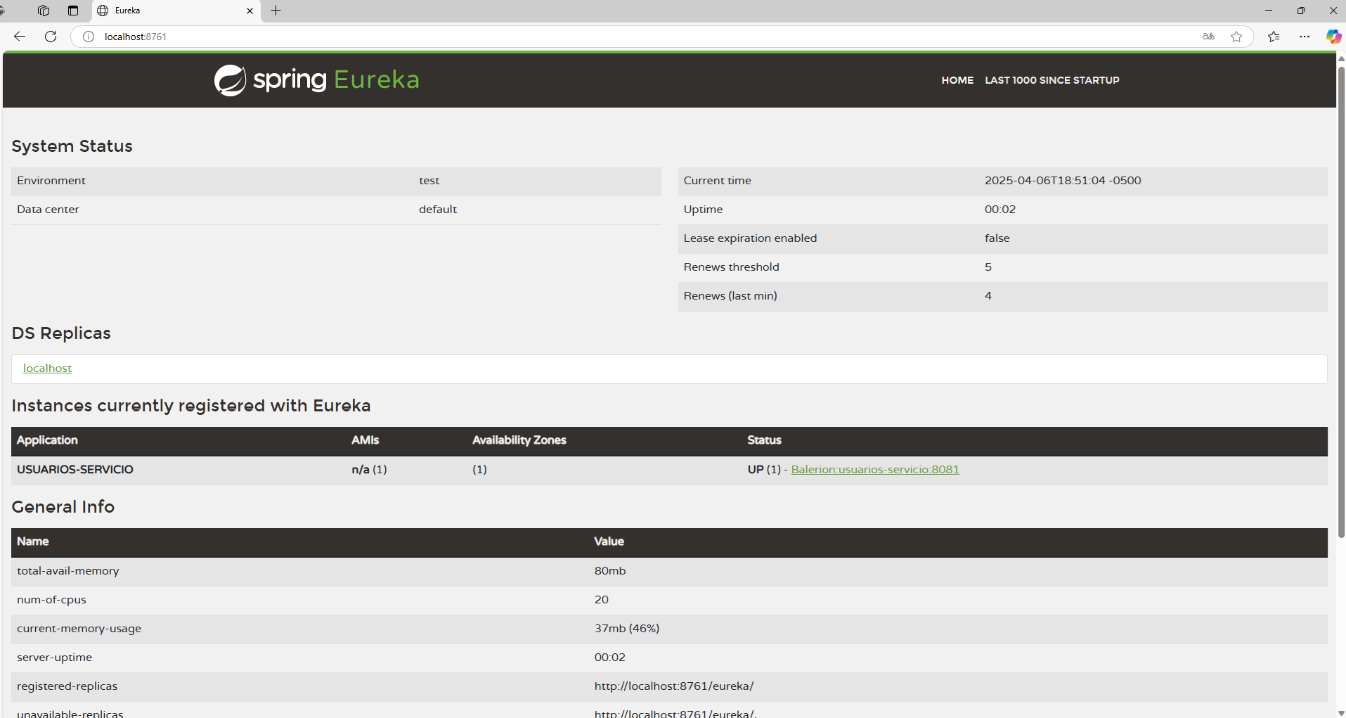
# Microservicio para Eureka Server



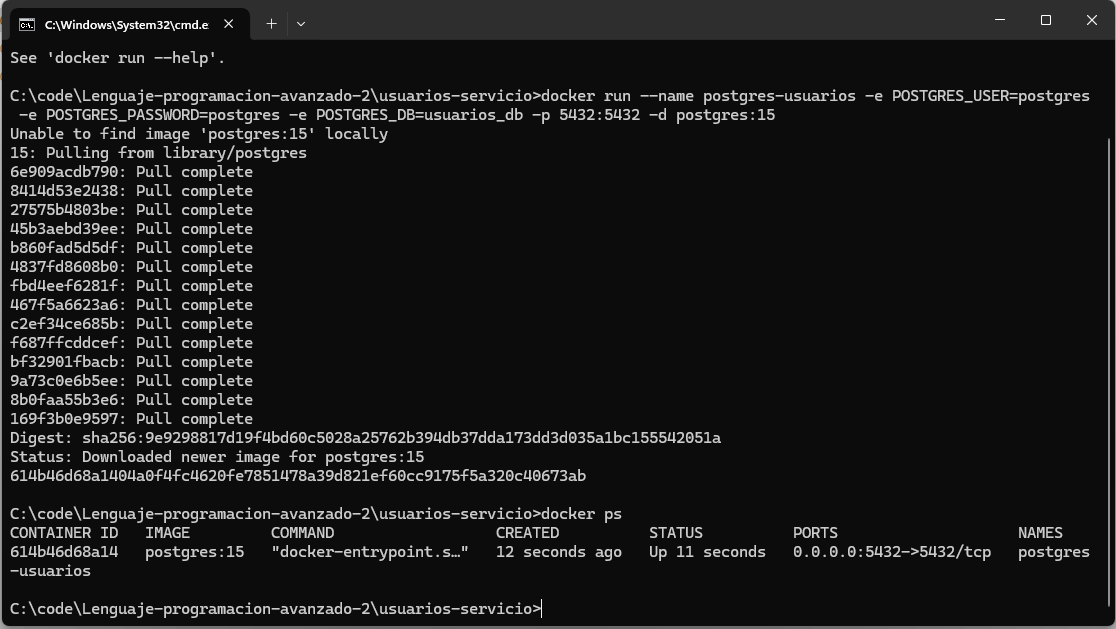


# Microservicio de Usuarios

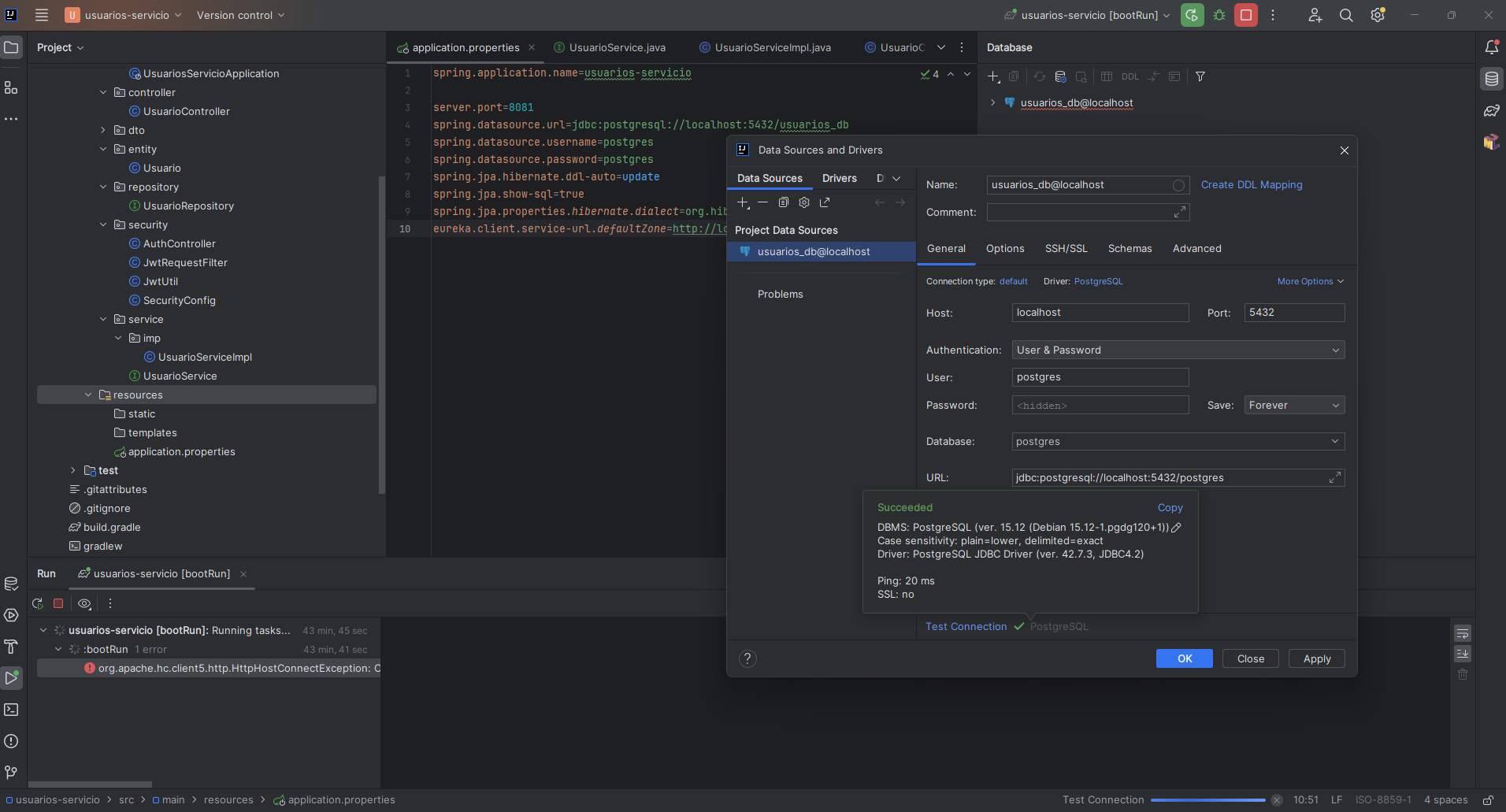
Al iniciar solamente usuarios-servicio:



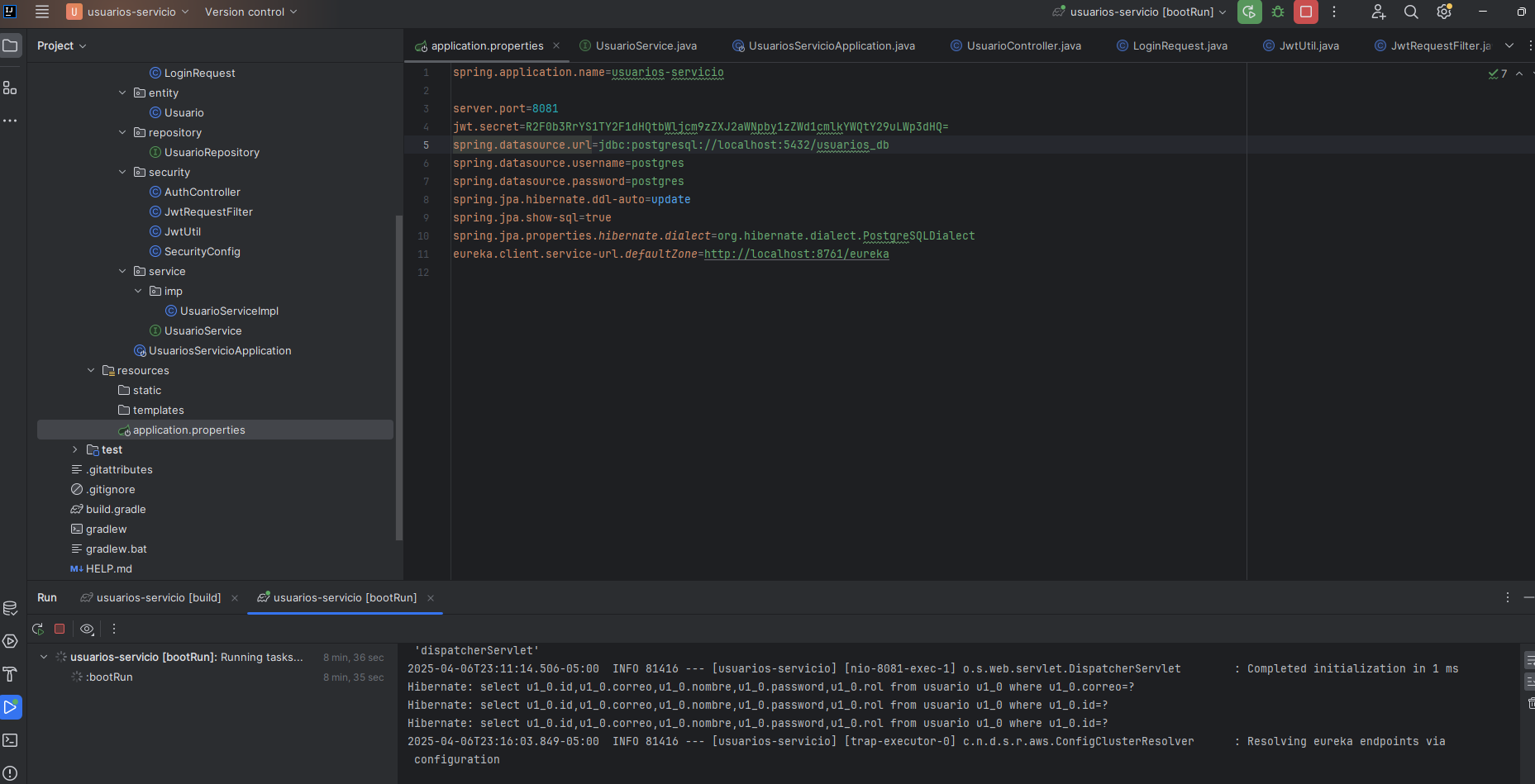
Creación de contenedor Docker para base de datos Postgres para usuarios-servicio:



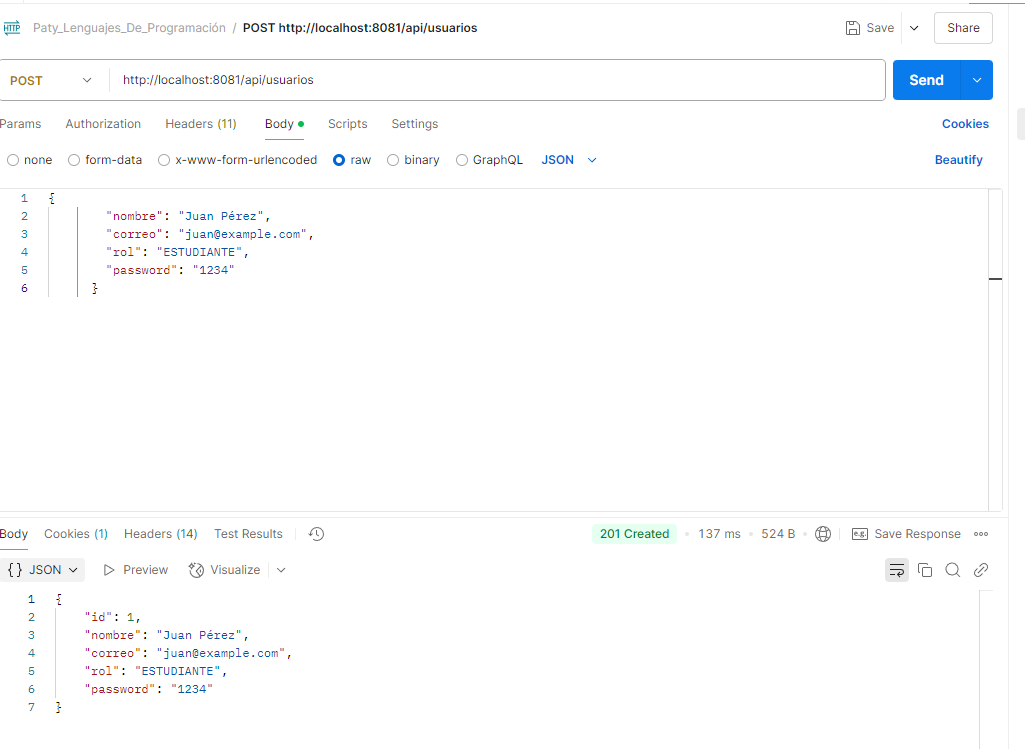
Conexión de usuarios-servicio a base de datos:



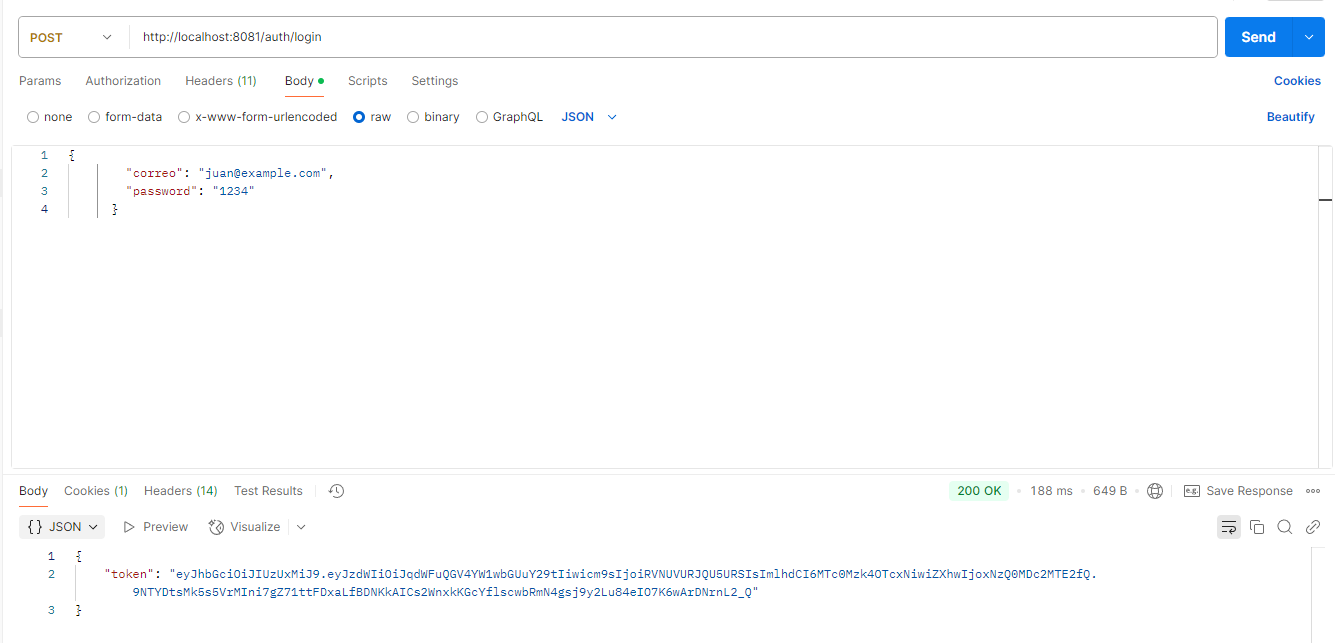
Ejecución de la aplicación:



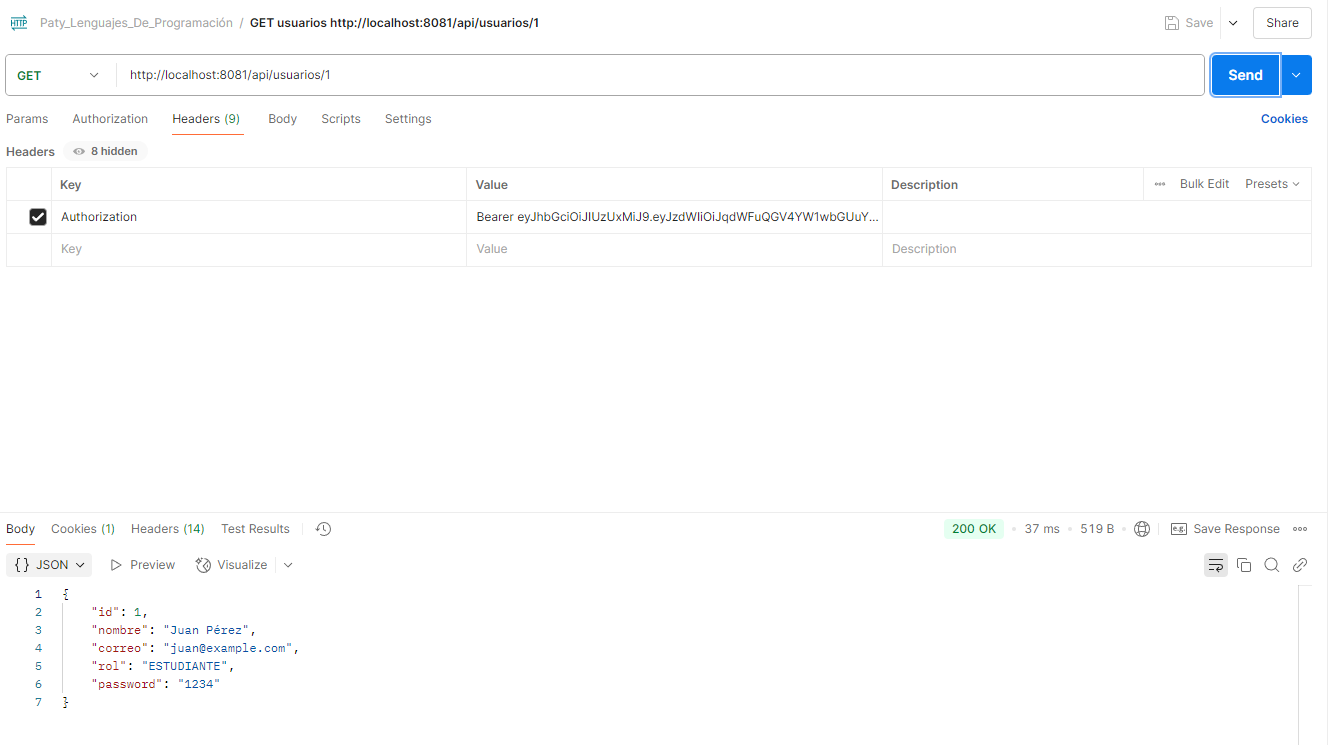
Creación del primer usuario:



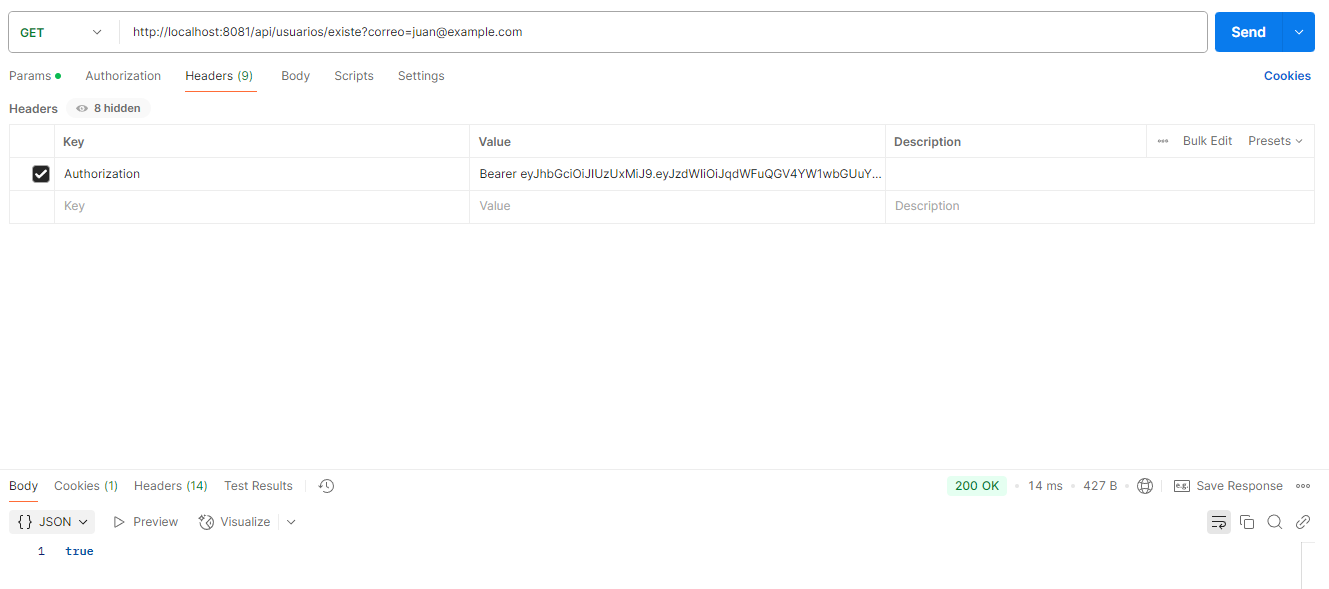
Login y generación del JWT:



Usar el token para ver los detalles de un usuario



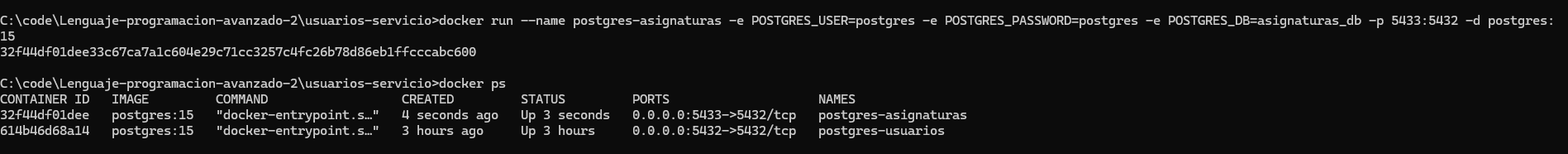
Validar si un usuario existe:



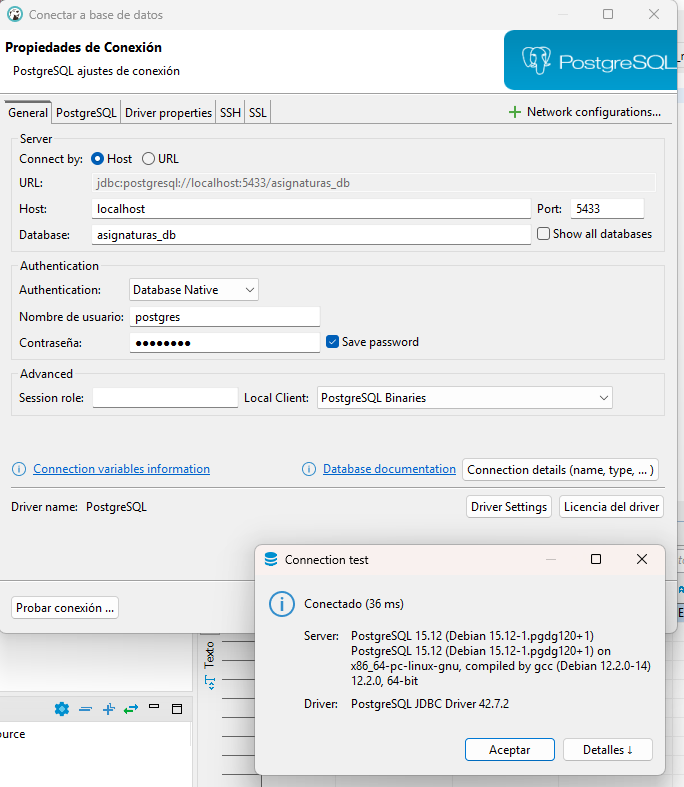
# Microservicio de Asignaturas

Creación de la base de datos para asignaturas dentro de un contenedor Docker:

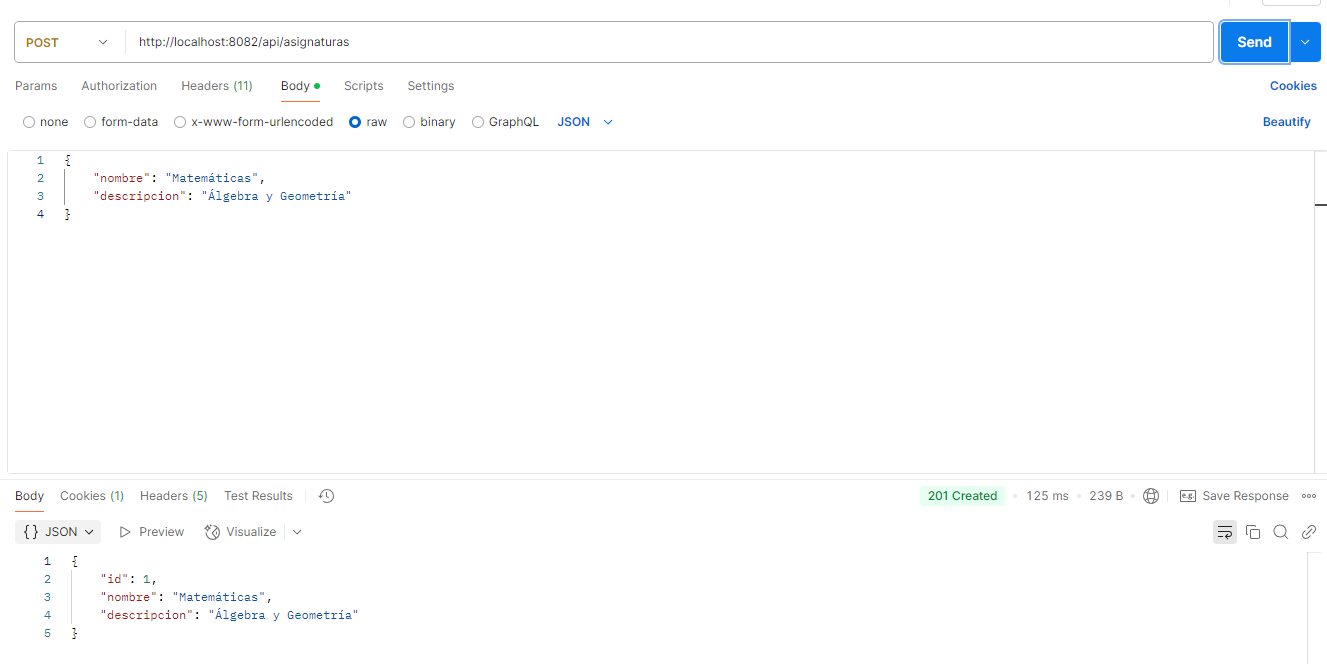
docker run --name postgres-asignaturas -e POSTGRES\_USER=postgres -e POSTGRES\_PASSWORD=postgres -e POSTGRES\_DB=asignaturas\_db -p 5433:5432 -d postgres:15



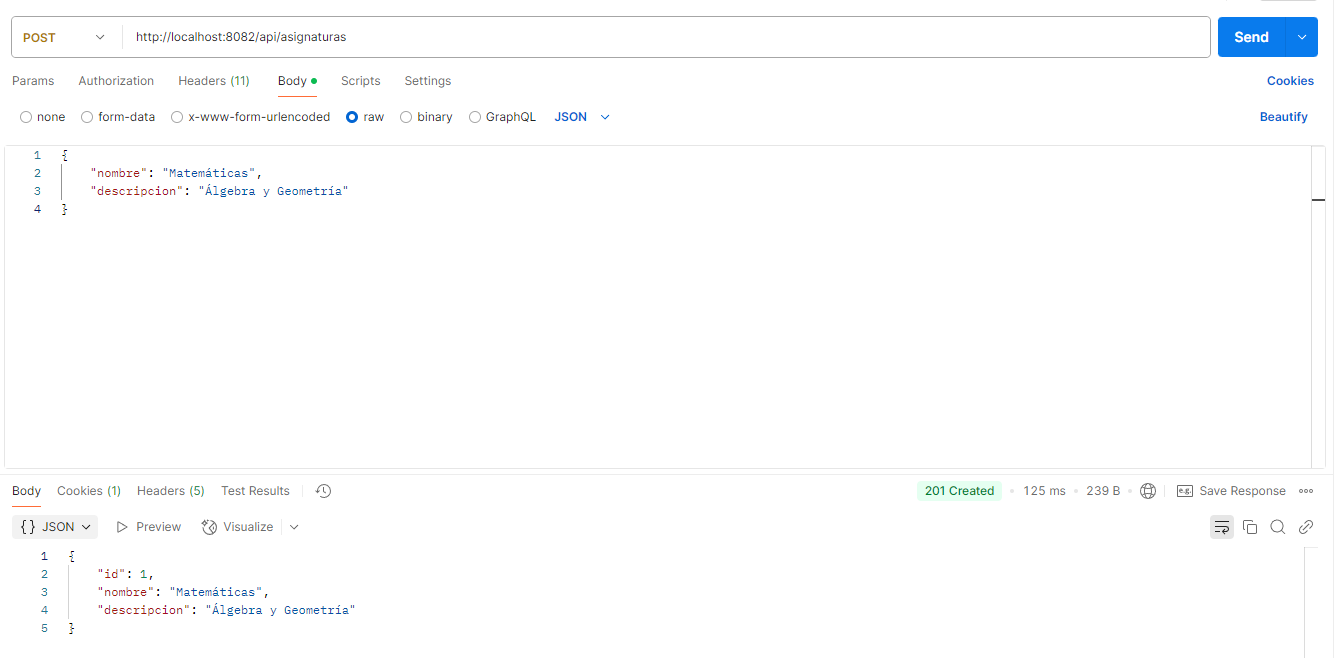
Conexión a la base de datos de asignaturas:

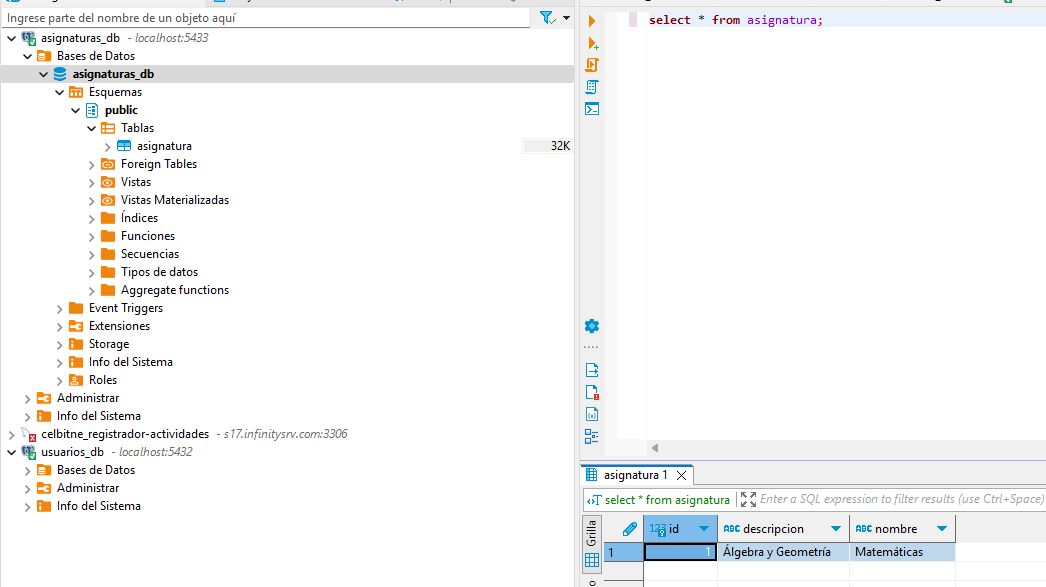


Creación de una asignatura:

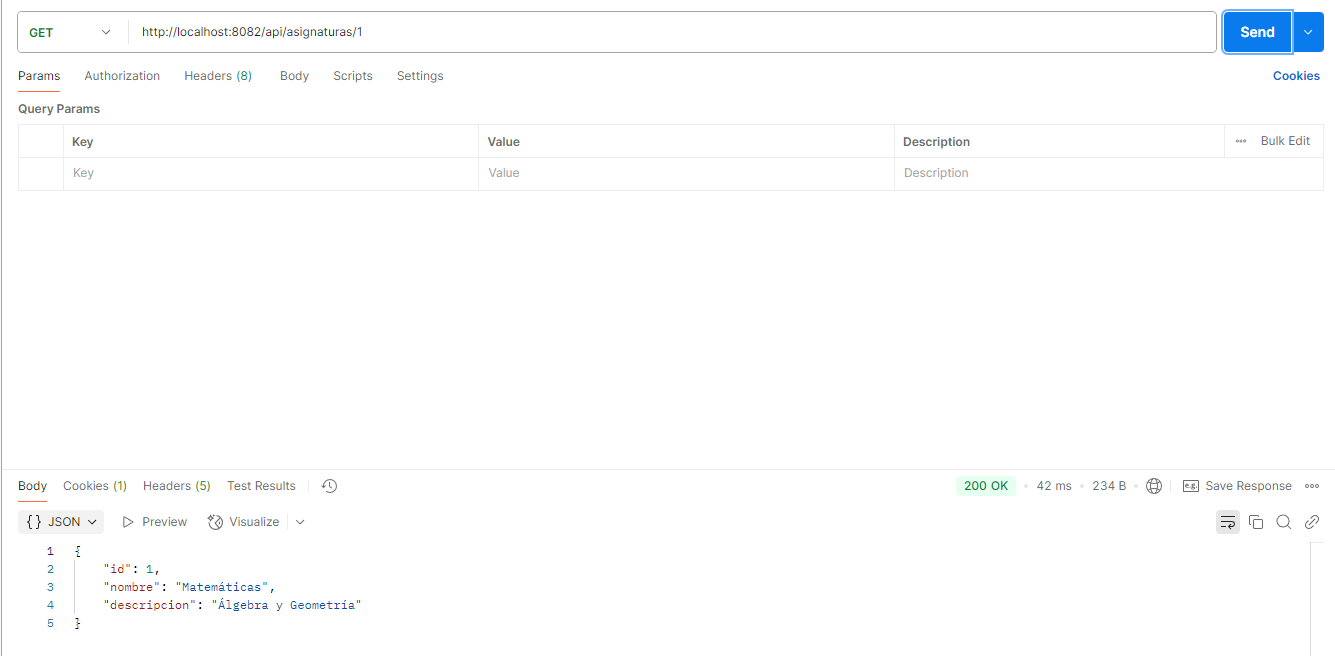


Creación de una asignatura:

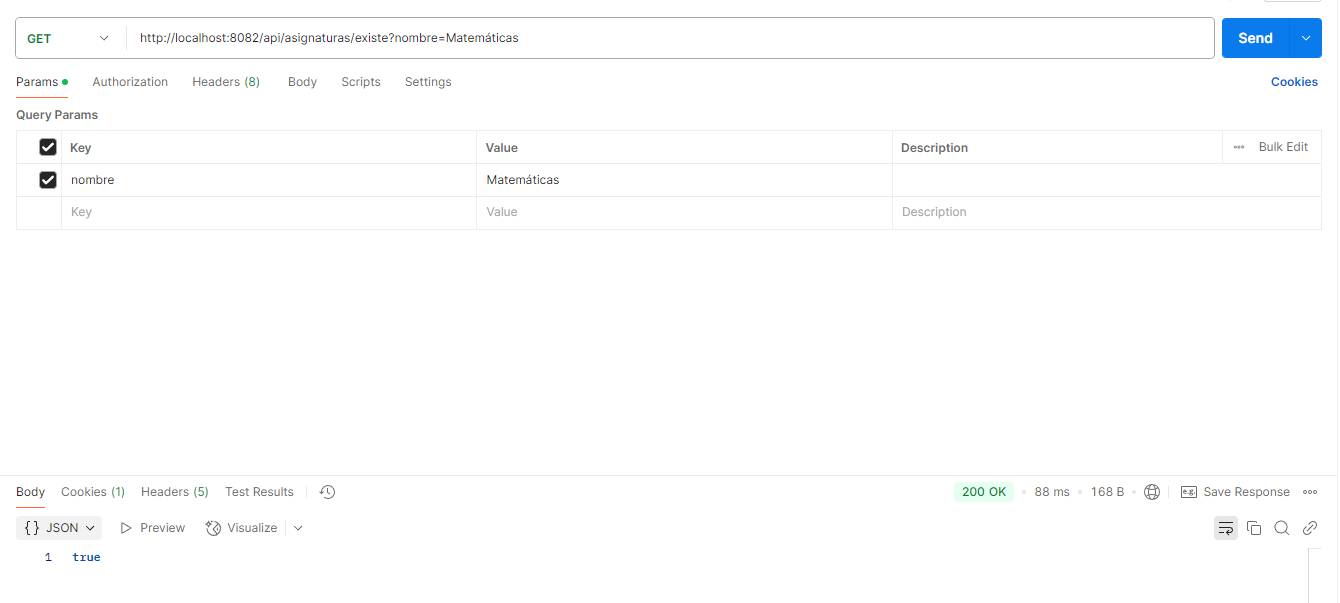




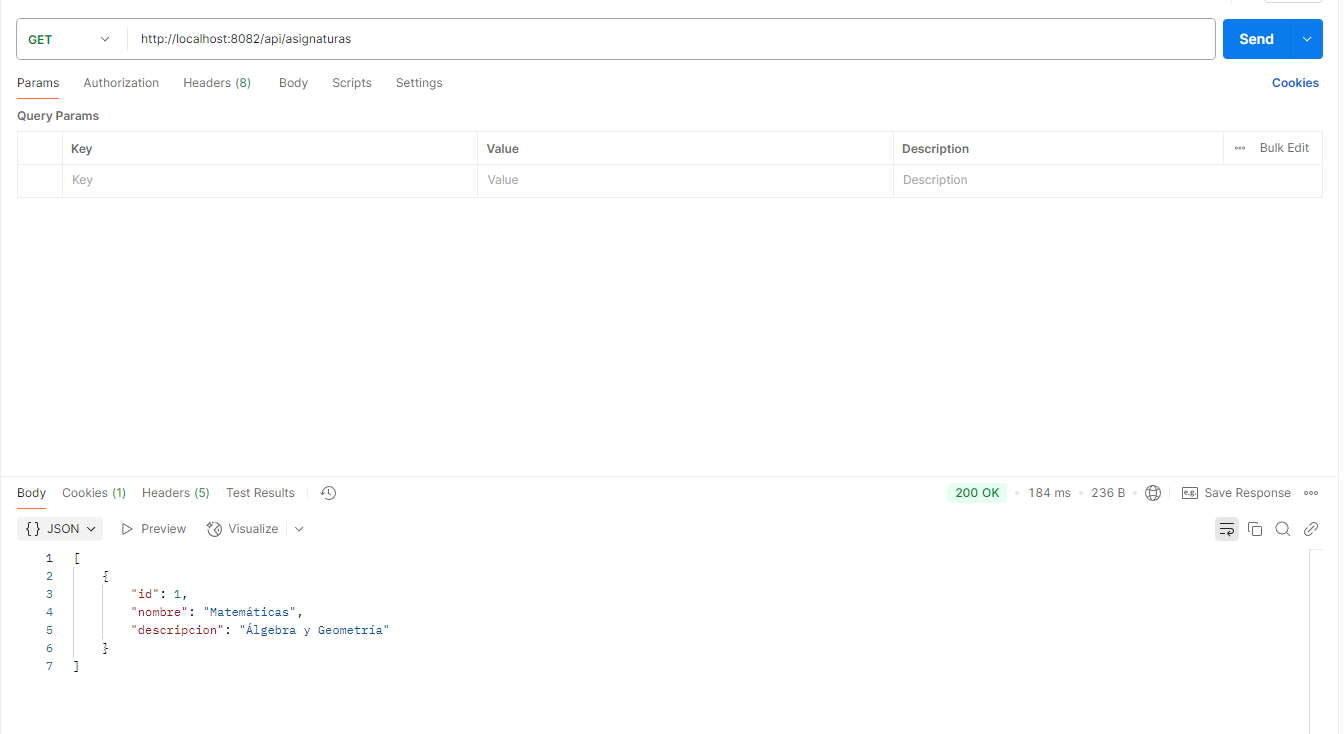
Obtener asignatura por id:



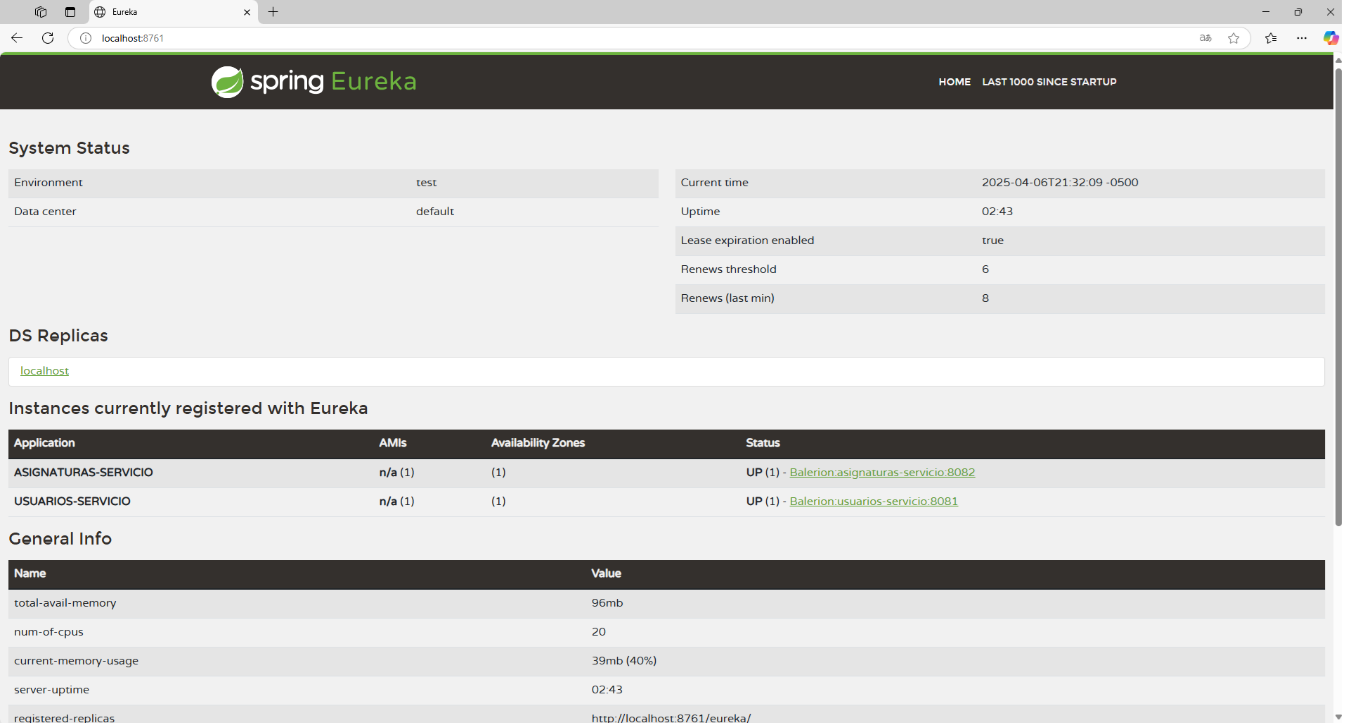
Verificar si existe asignatura:

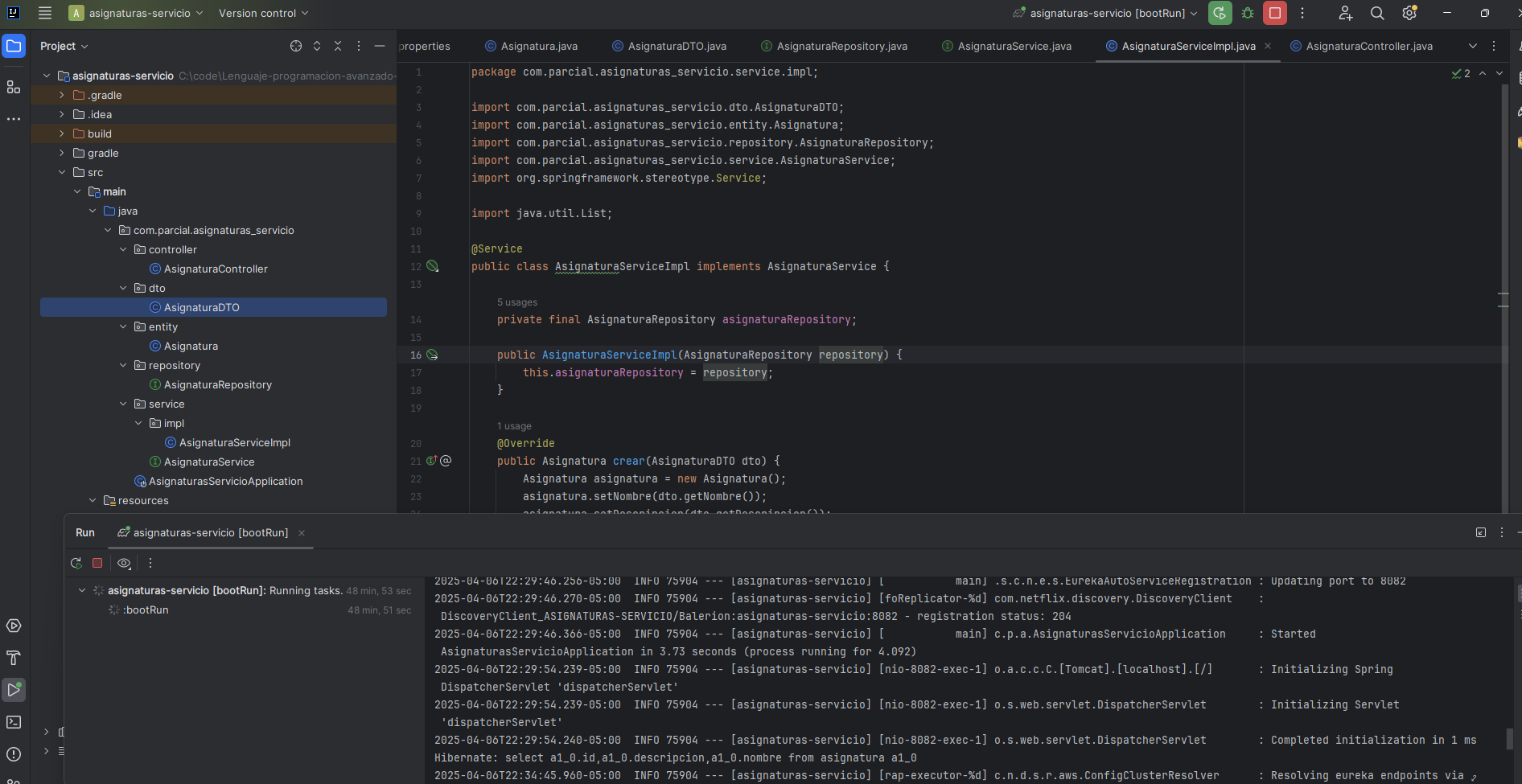


Obtener todas las asignaturas:



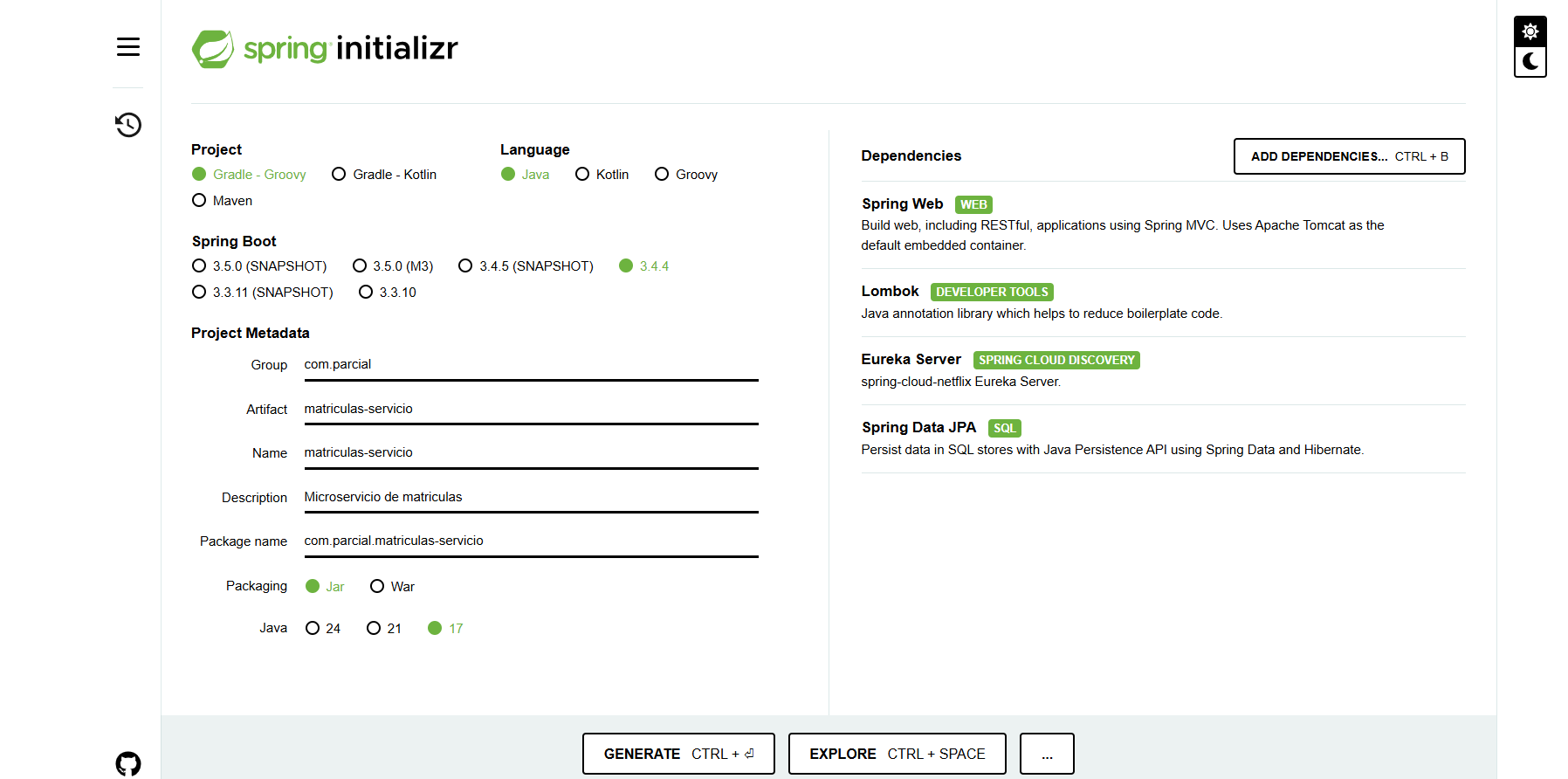
Validación de que los dos servicios desarrollados hasta el momento están registrados en Eureka y funcionando:





# Microservicio de Matrículas

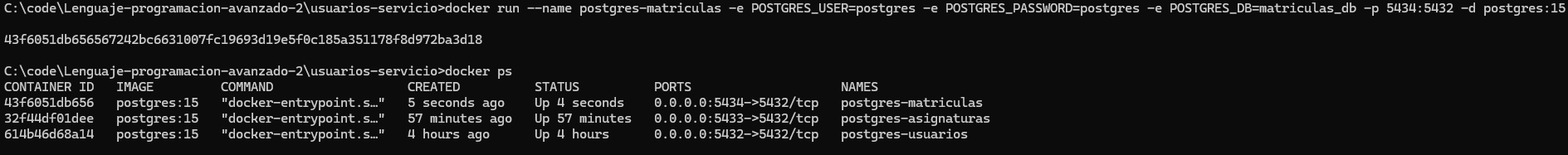
Generación de Matriculas servicio:



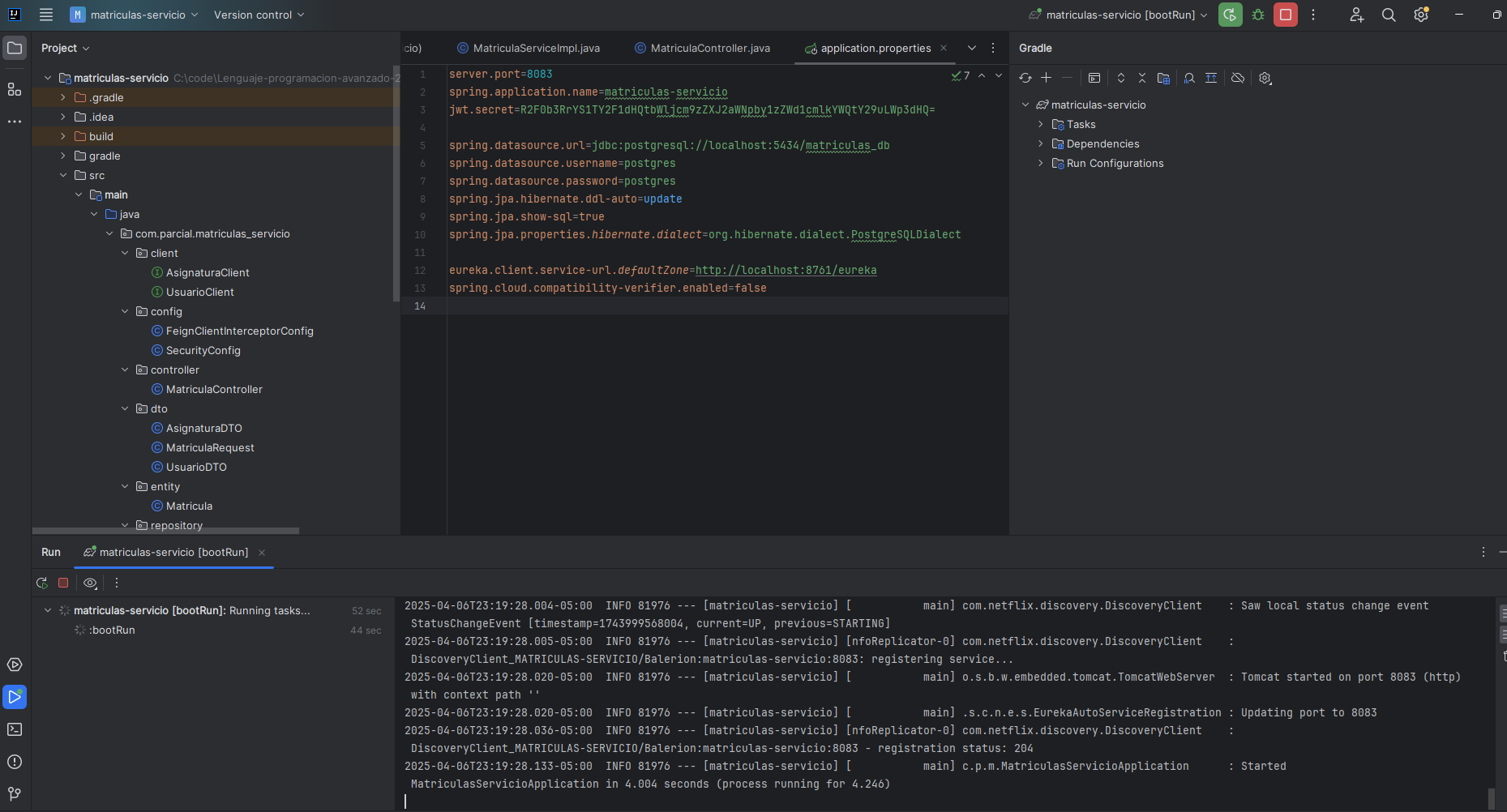
Generación de contenedor de base de datos para matriculas-servicio:

Comando:

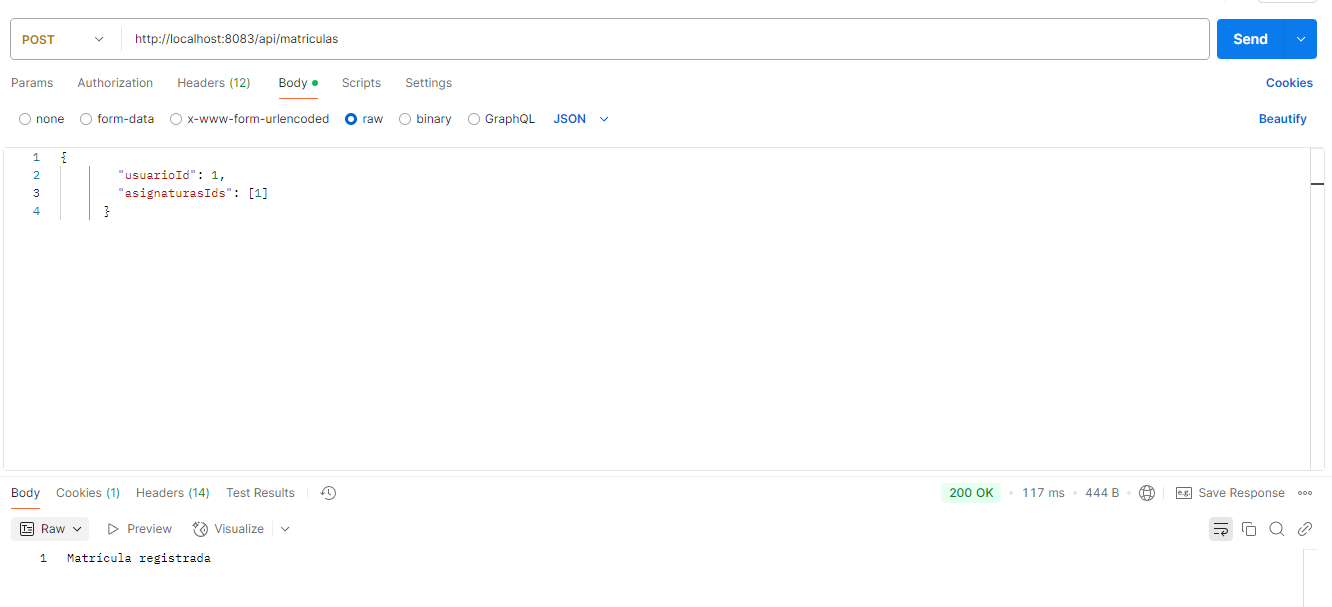
docker run --name postgres-matriculas -e POSTGRES\_USER=postgres -e POSTGRES\_PASSWORD=postgres -e POSTGRES\_DB=matriculas\_db -p 5434:5432 -d postgres:15



Ejecución del microservicio de matrículas:



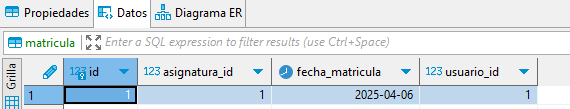
Para la creación de una matrícula, primero se debe hacer un login en usuarios y obtener el token jwt, con este token se envía un request a matrículas-servicio que hace una validación del token y adicionalmente consulta a usuarios-servicio si el usuario existe y consulta a asignaturas-servicio si la asignatura existe. Si todos existen, realiza la matrícula, como se muestra a continuación. Internamente los consumos se realizan utilizando Feign Client:



Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

El registro fue creado en la base de datos exitosamente:



Respecto al estado de los servicios en Eureka Server, se puede observar que los tres microservicios están en ejecución:

