Microservicios con SPRING BOOT

Índice

[Creando y configurando servidor Zuul Api Gateway 2](#_Toc96723143)

[Zuul Filtro http pre - Tiempo transcurrido 4](#_Toc96723144)

[Spring Cloud Gateway 7](#_Toc96723145)

[Creando y configurando servicio Spring Cloud Gateway 7](#_Toc96723146)

[Implementando filtros globales pre y post 8](#_Toc96723147)

[Modificando el request en el filtro 8](#_Toc96723148)

[Implementando Gateway Filter Factory 9](#_Toc96723149)

[Configurando y probando nuestro Gateway Filter Factory Personalizado 10](#_Toc96723150)

# Creando y configurando servidor Zuul Api Gateway

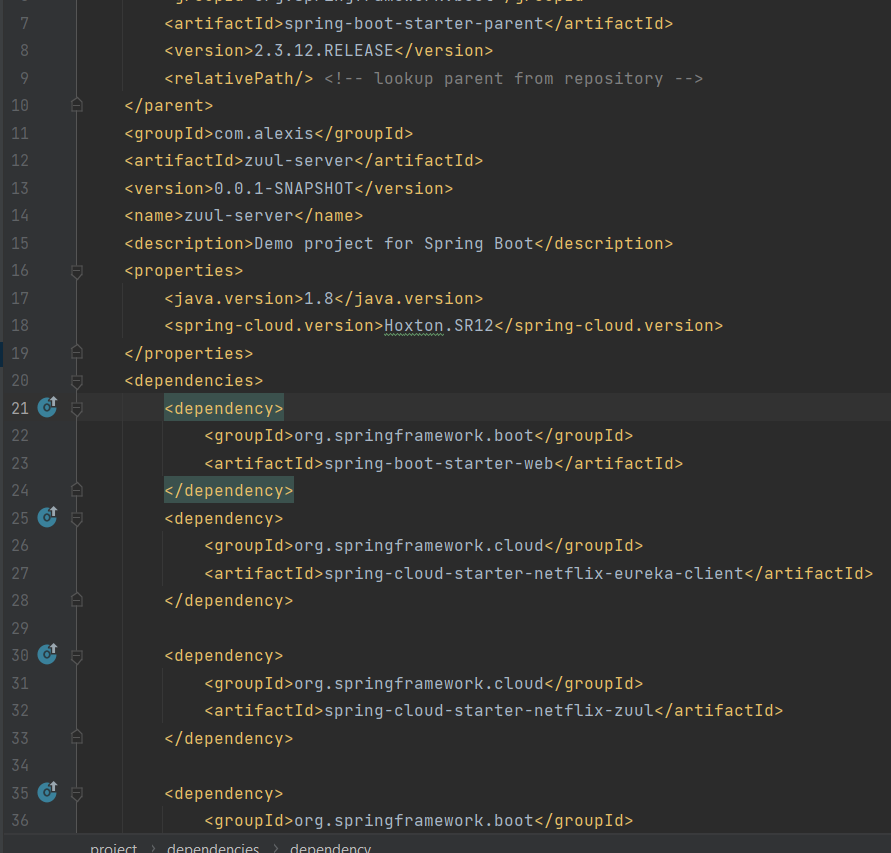
Es un servicio especial que se encarga del acceso a los demás microservicios que estén registrados en EUREKA también conocida como puerta de enlace de los microservicios.

Creamos nuestro proyecto servicio zuul-server en spring intilzr con las siguientes dependencias.

-Spring Web

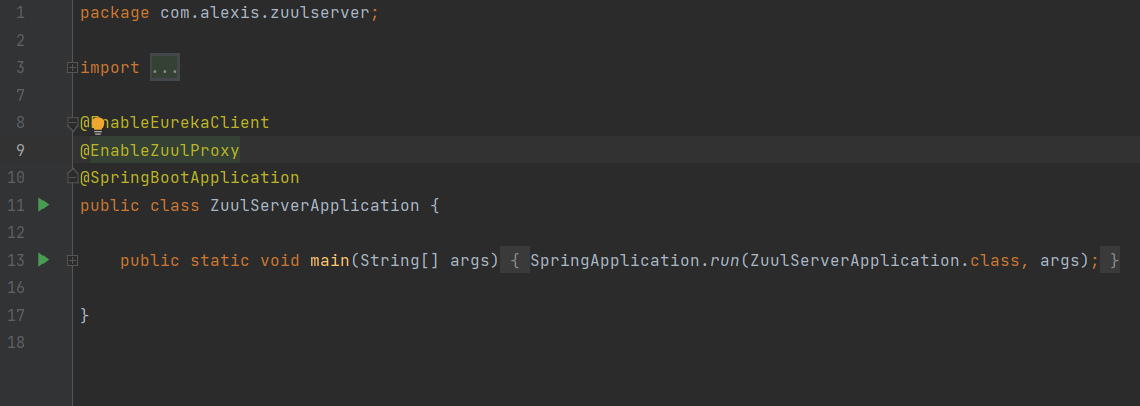
-Eureka Discovery Client

Archivo pom.xml

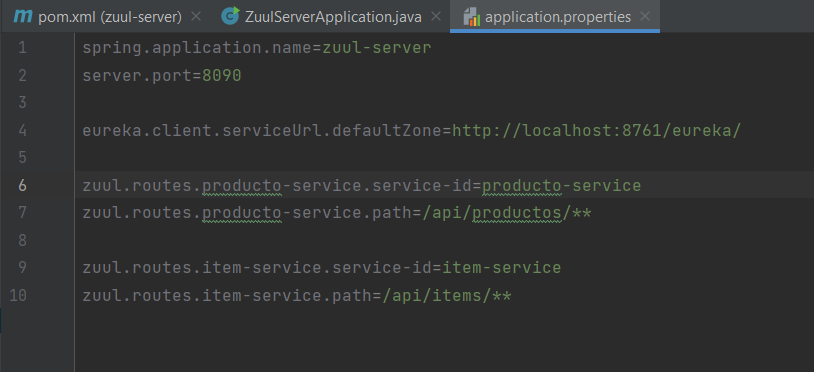


Necesario manejar esta versión de springboot y springcloud para el uso de zuul-server.

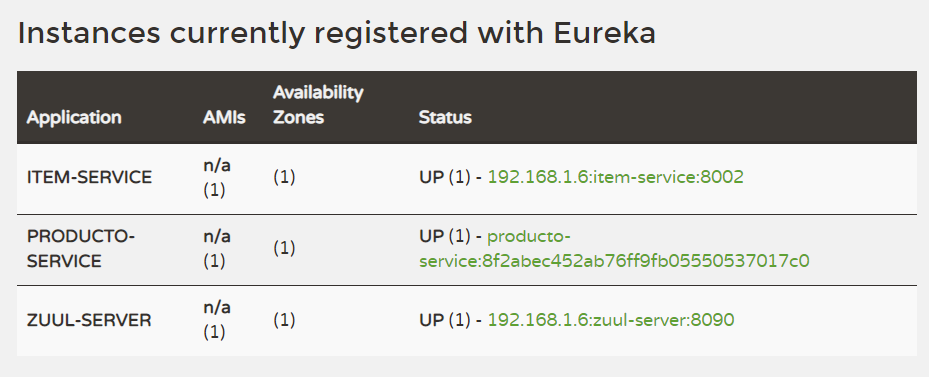
Luego Etiquetamos algunas Instancias en la clase principal del proyecto



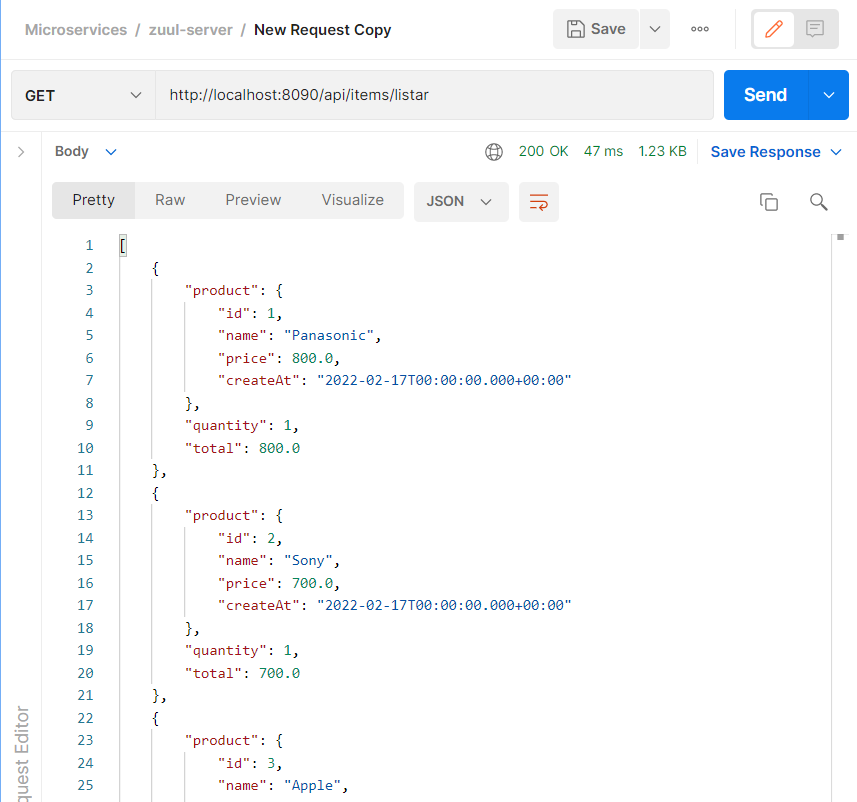
Modificamos el archivo application.properties



Verificamos que este levantado el servicio de zuul-server en el servidor de EUREKA.



Probando el servicio de zuul-server en postman.



# Zuul Filtro http pre - Tiempo transcurrido

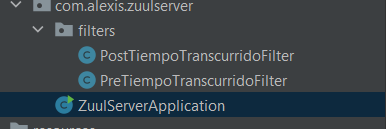
Existen tres tipos de filtro

PRE 🡪 se ejecuta antes de que la respuesta (request) sea enrutado 🡪 Se usa para pasar datos al request.

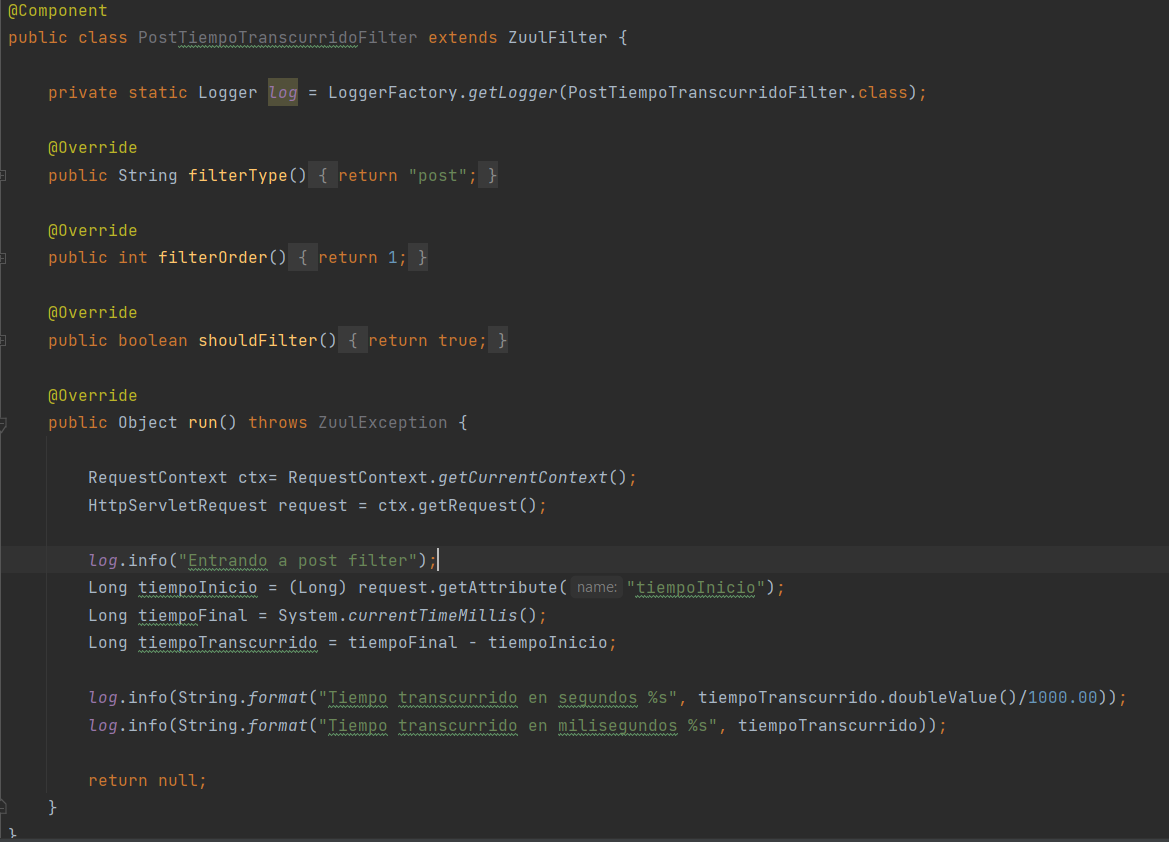
POST 🡪 Se ejecuta después de que el request haya sido enrutado 🡪 Se usa para modificar la respuesta.

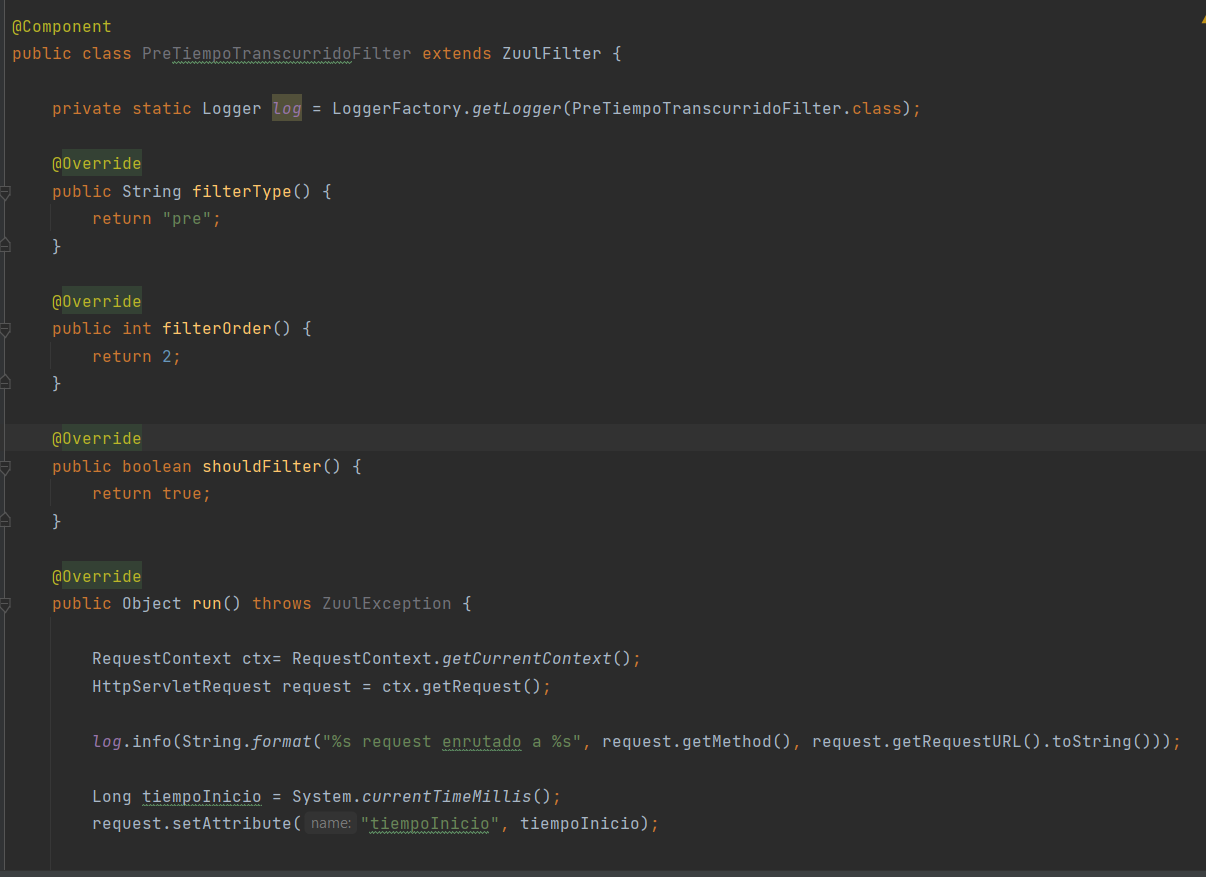
ROUTE 🡪 Se ejecuta durante el enrutado del request, aquí se resuelve la ruta 🡪 Se usa para la comunicación con el microservicio.

Creamos nuestro paquete de filtros en el proyecto zuul-server y creamos nuestras clases PreTiempoTranscurridoFilter y PostTiempoTranscurridoFilter

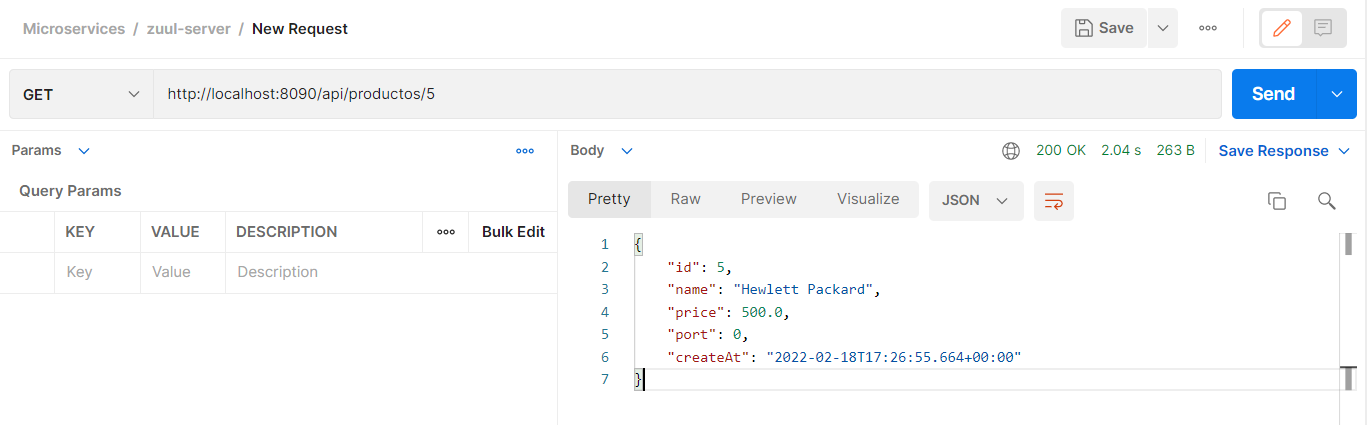


Heradamos la interfaz de ZuulFilter y completamos las funcionalidades de cada método

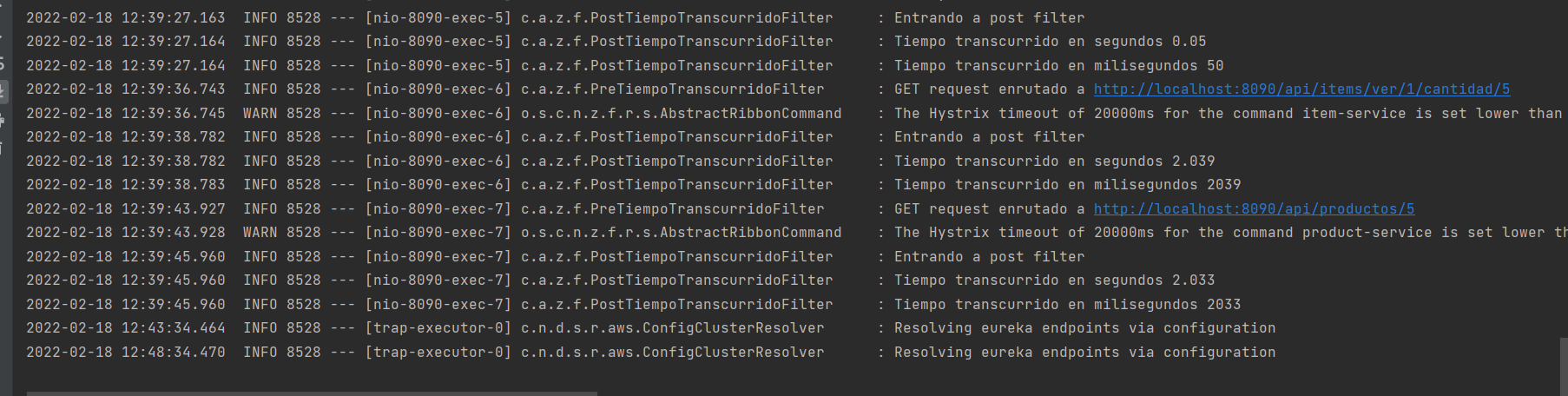




Probamos en postman nuestras apis en el servido de zuul-server



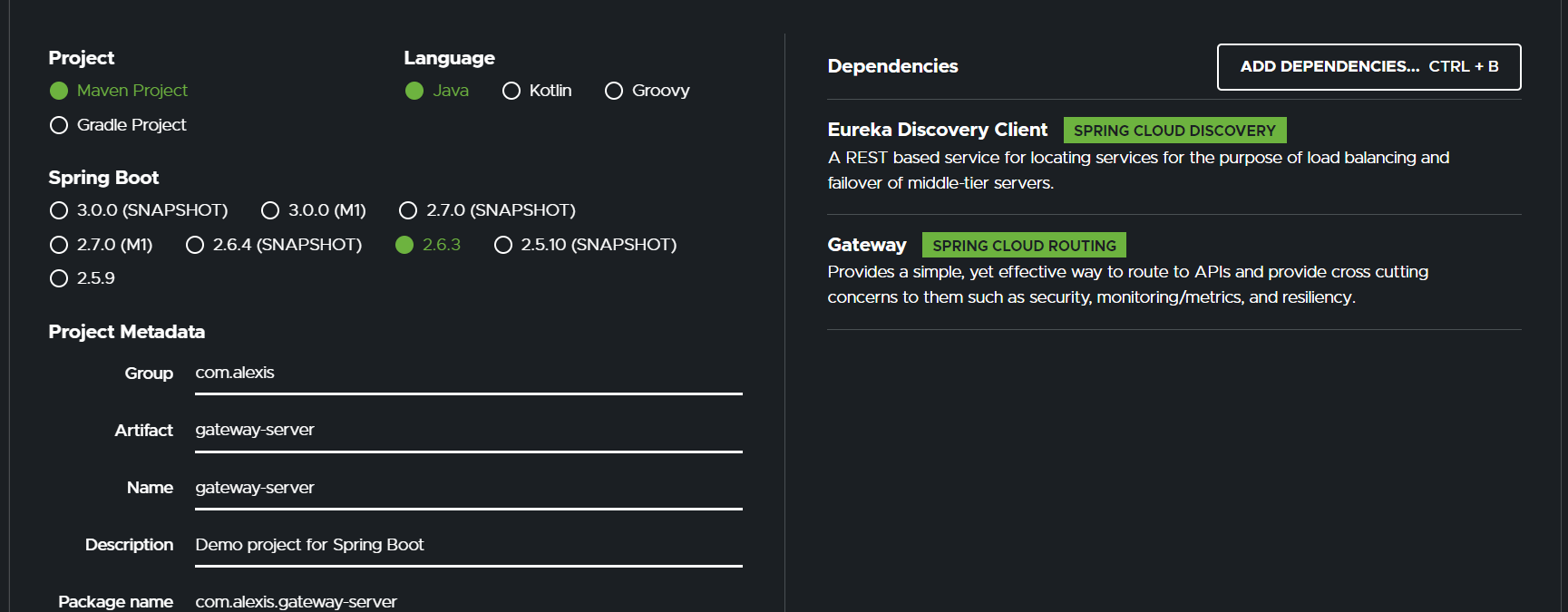
Verificamos el tiempo transcurrido de cada respuesta de los servicios



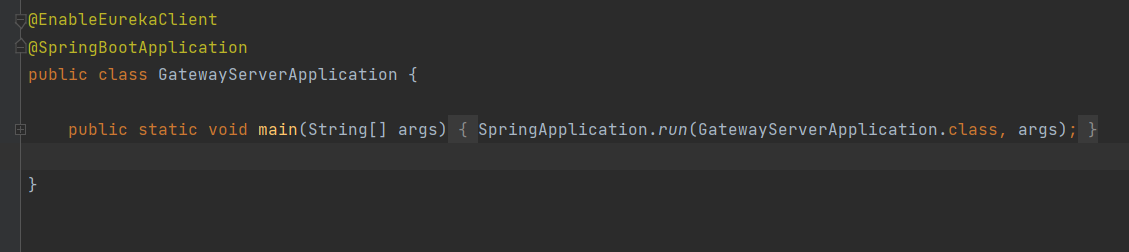
# Spring Cloud Gateway

## Creando y configurando servicio Spring Cloud Gateway

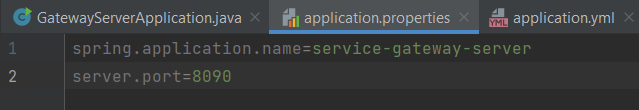
Creamos nuestro proyecto Gateway-server en spring initilzr agregando las dependencias de:



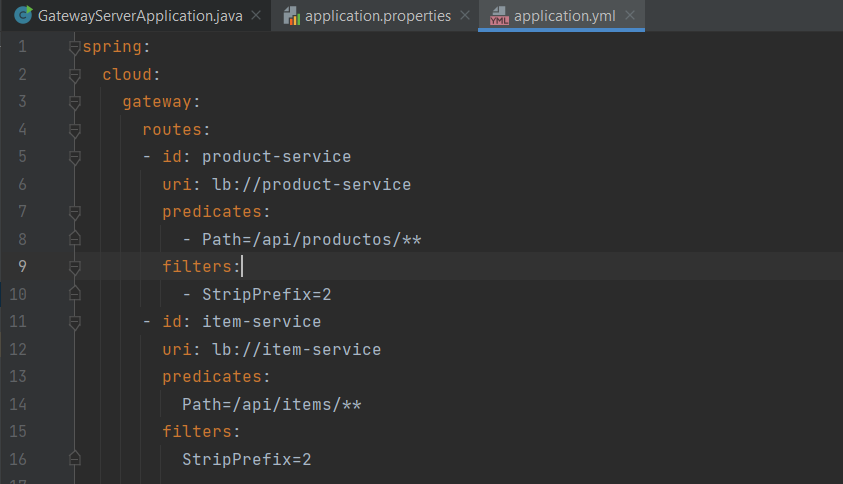
Marcamos nuestra clase principal con @EnableEurekaClient



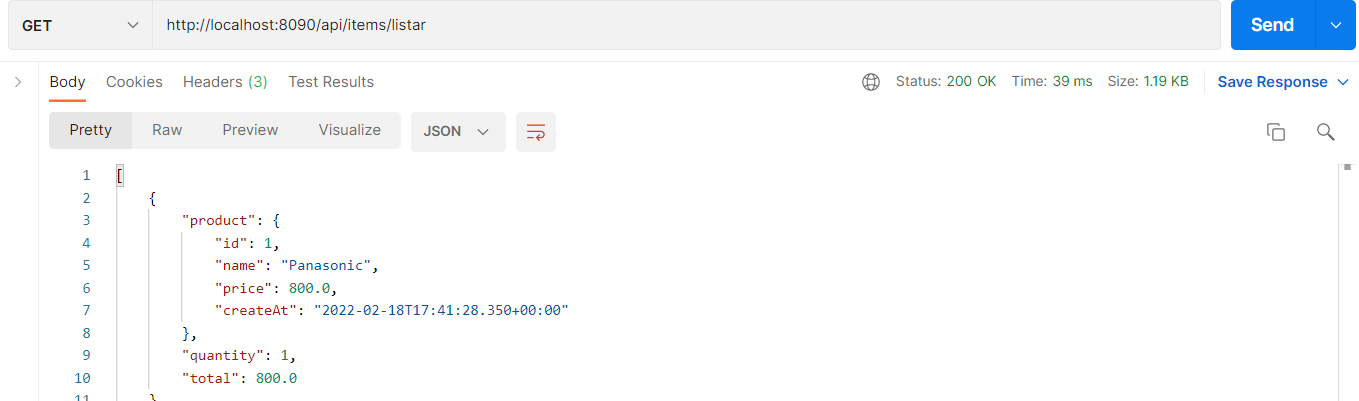
Configuramos el archivo application.properties:



Creamos el archivo application.yml y configuramos las rutas del api Gateway.

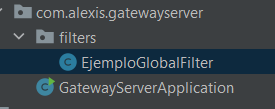


Probamos en postman el servicio de Gateway-server



## Implementando filtros globales pre y post

Creamos un nuevo paquete filters con la siguiente clase:



Implementamos la interfaz GlobalFilter a la clase EjemploGlobalFilter



exchange 🡪 Podemos acceder al request y al response de una manera poder modificar el req y res y así mismo hacer validaciones para así poder acceder o denegar el acceso a nuestros servicios por motivo de seguridad.

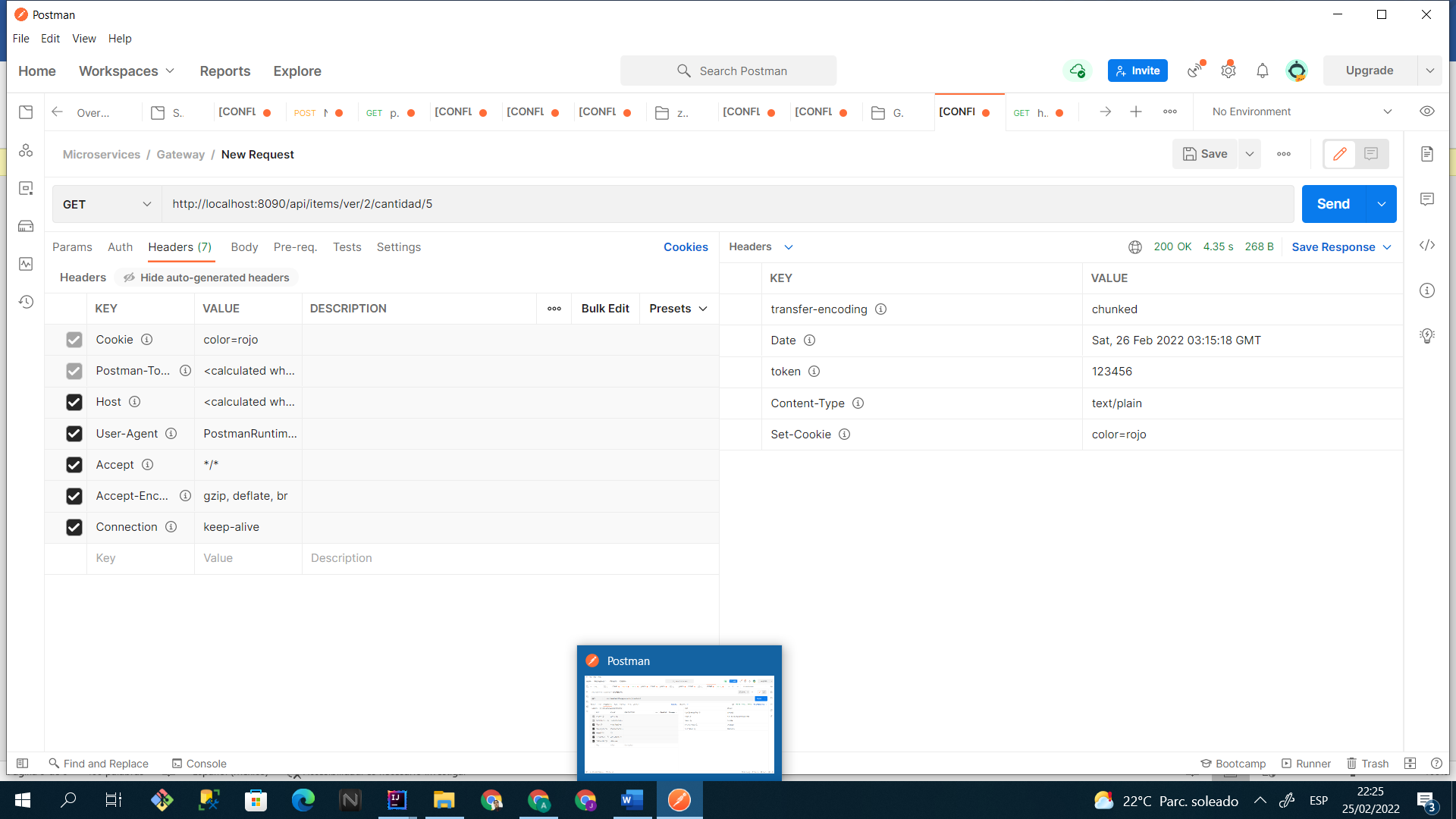
chain 🡪 Cadena de filtros.

## Modificando el request en el filtro

Podemos modificar el request utilizando los siguientes métodos para añadir cabeceras (headers) a nuestro servicio de Gateway, así mismo actualizar la respuesta (response).



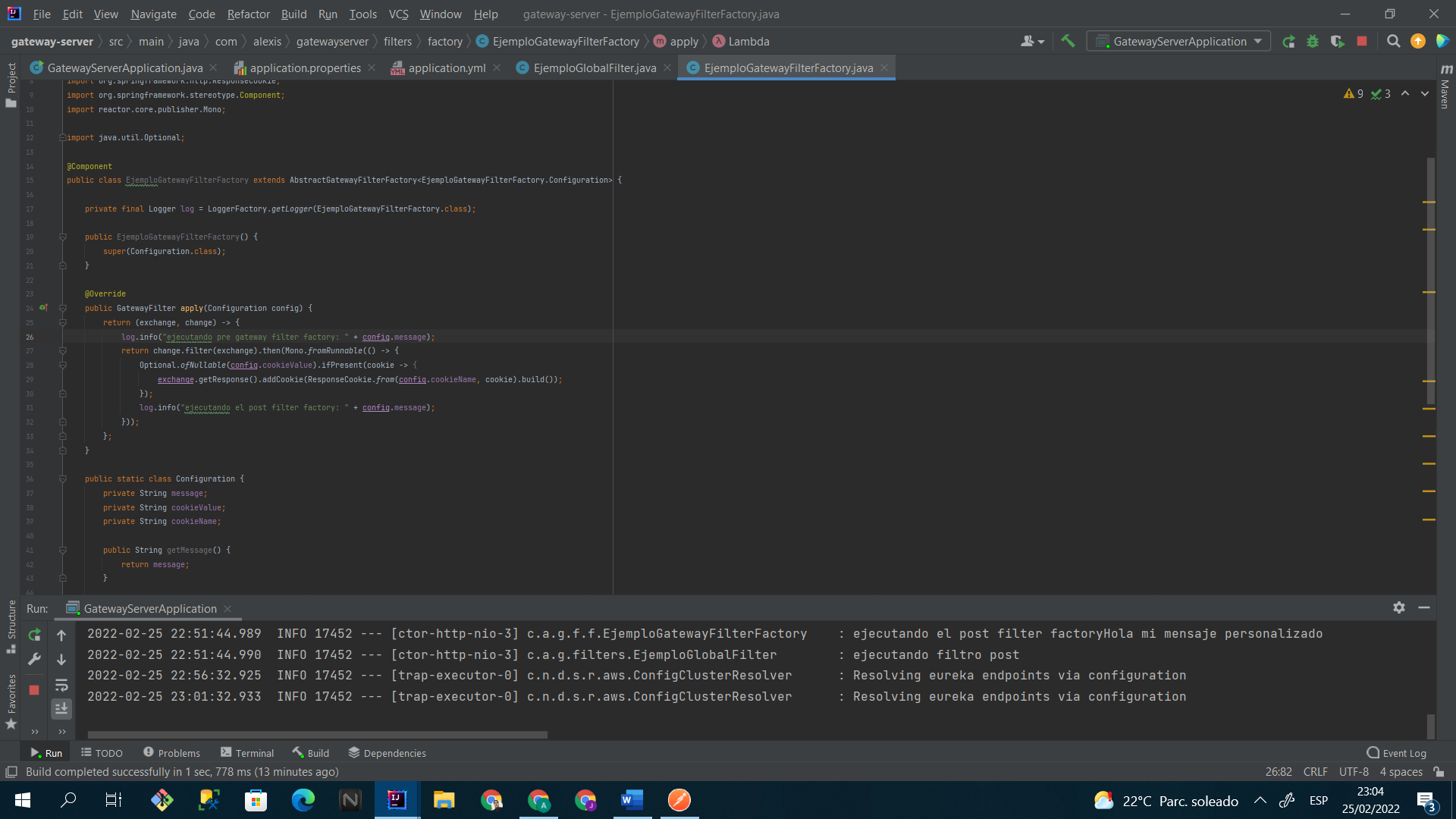
Revisando nuestro postman.



## Implementando Gateway Filter Factory

Es otra manera de filtrar nuestros servicios.

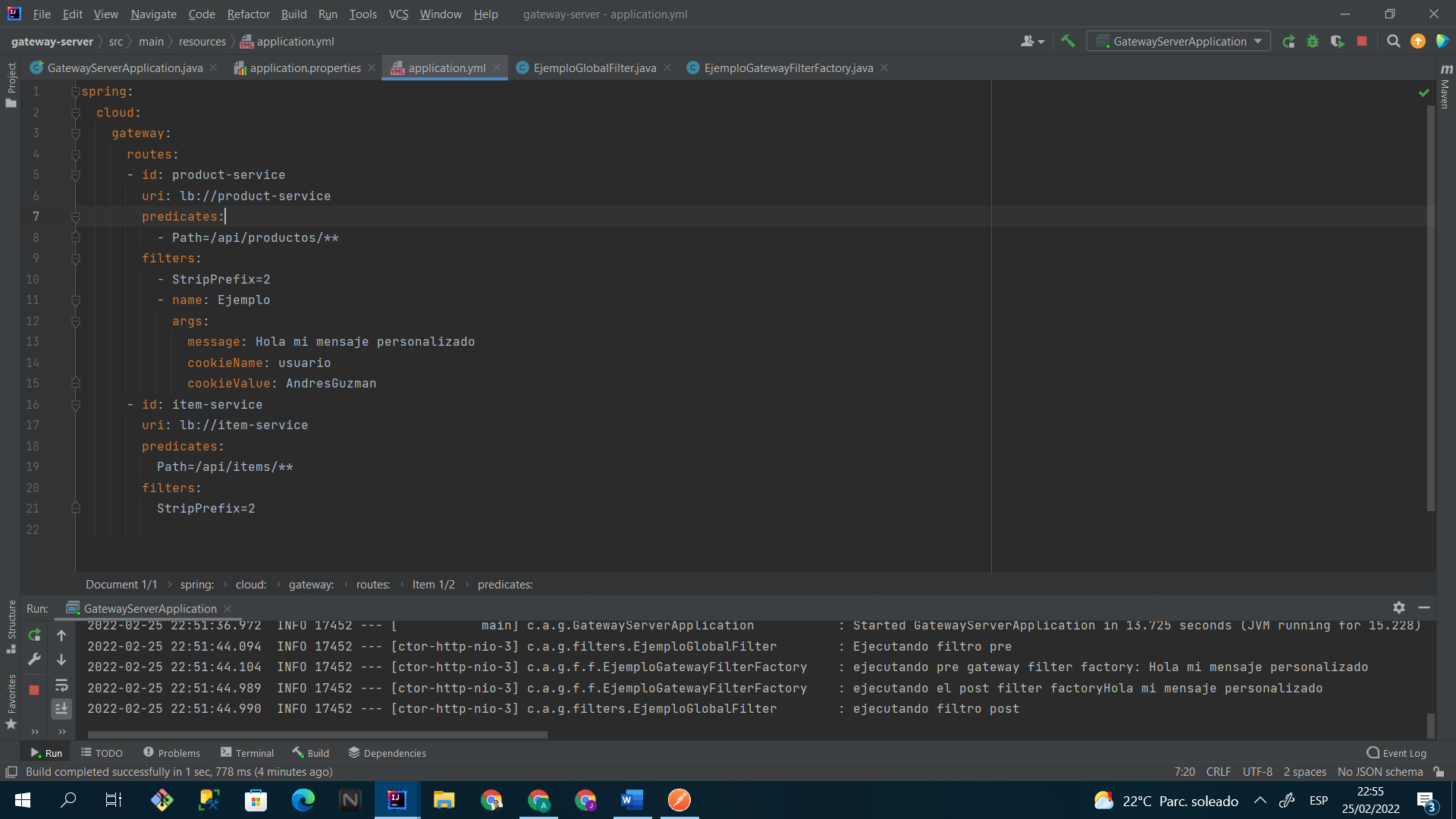
Primero creamos nuestra clase EjemploGatewayFilterFactory y extendemos de la clase AbstractGatewayFilterFactory



Se usa la misma lógica del anterior punto ya que los métodos heredados contienen el mismo método del tema pasado solo que en este caso podemos personalizar nuestros filtros agregando más factores y también filtrar los servicios individualmente.

## Configurando y probando nuestro Gateway Filter Factory Personalizado

Configuramos el archivo properties.yml del proyecto Gateway-server.



Datos de la subclase Configuration de EjemploGatewayFilterFactory se puede ingresar los valores desde el archivo yml

Agregamos el sufijo de la clase **Ejemplo**GatewayFilterFactory