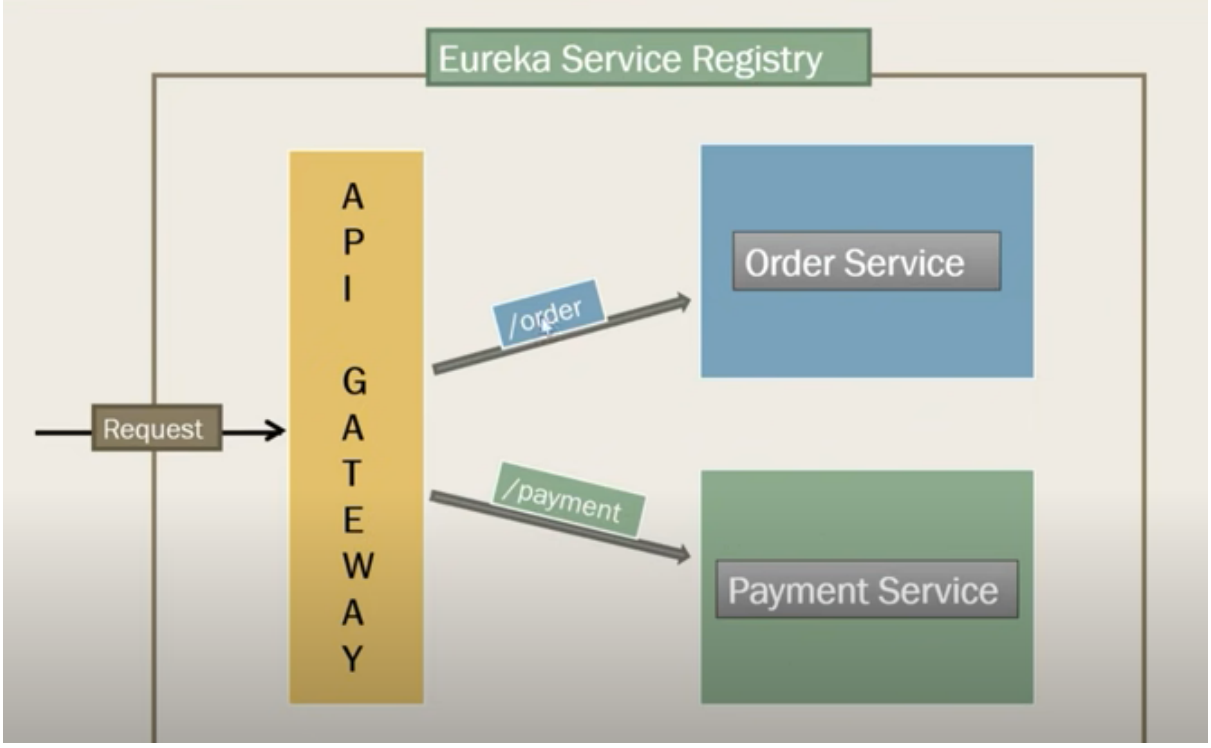
Microservice Using Spring Boot & Spring Cloud 2H:

What should we do? :

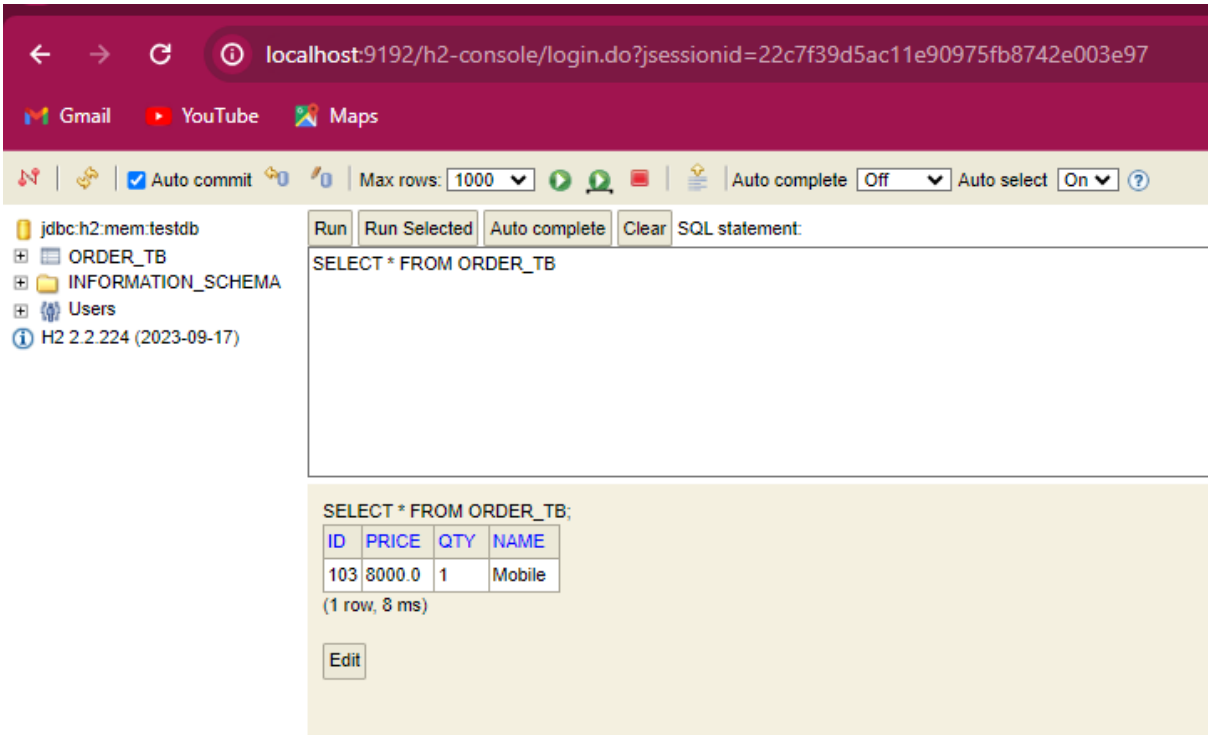
Agenda of this tutorial :

1. **Create 2 microservice from scratch.**
2. **Register microservice in Eureka Service Discovery.**
3. **integrate Spring Cloud Gateway for routing user requests.**
4. **Integrate Hystrix & Hystrix Dashboard to identify failure for downstream services.**
5. **Spring cloud config server using Git to Centralize configuration across applications.**
6. **ELK Stack to centralize logging across all microservices.**
7. **Zipkin & Sleuth to centralize tracing in microservice architecture.**

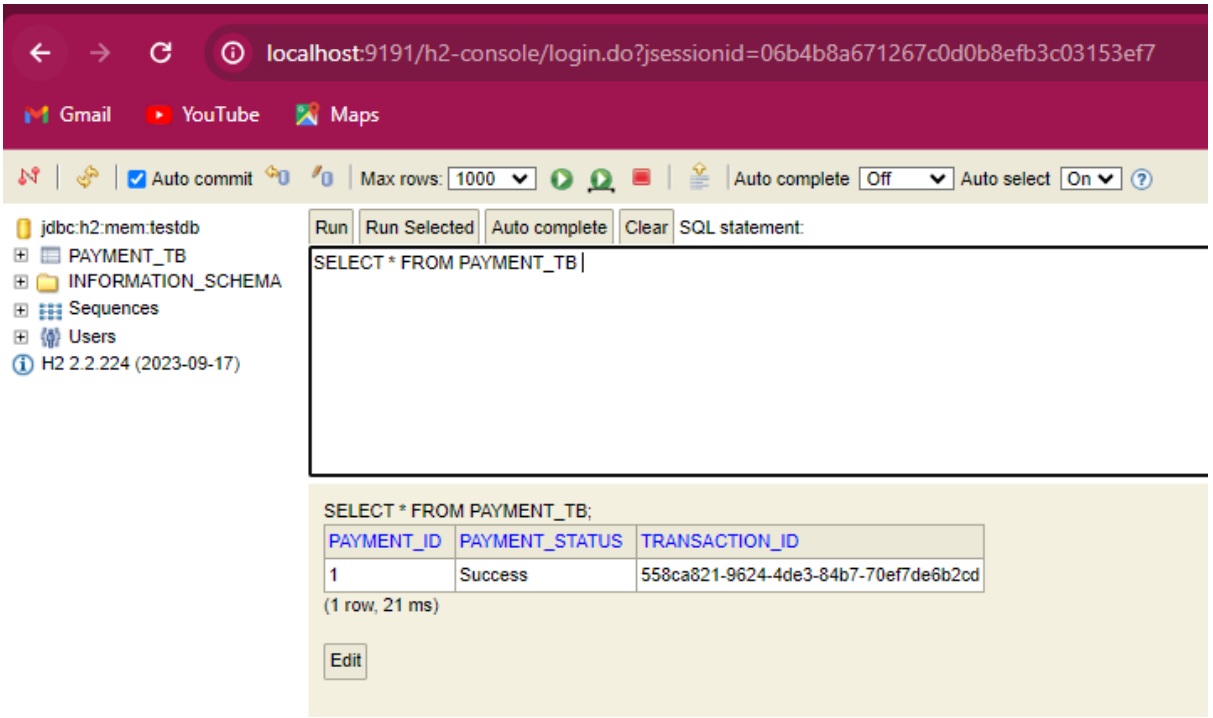


*1 : **Create 2 microservice from scratch :***

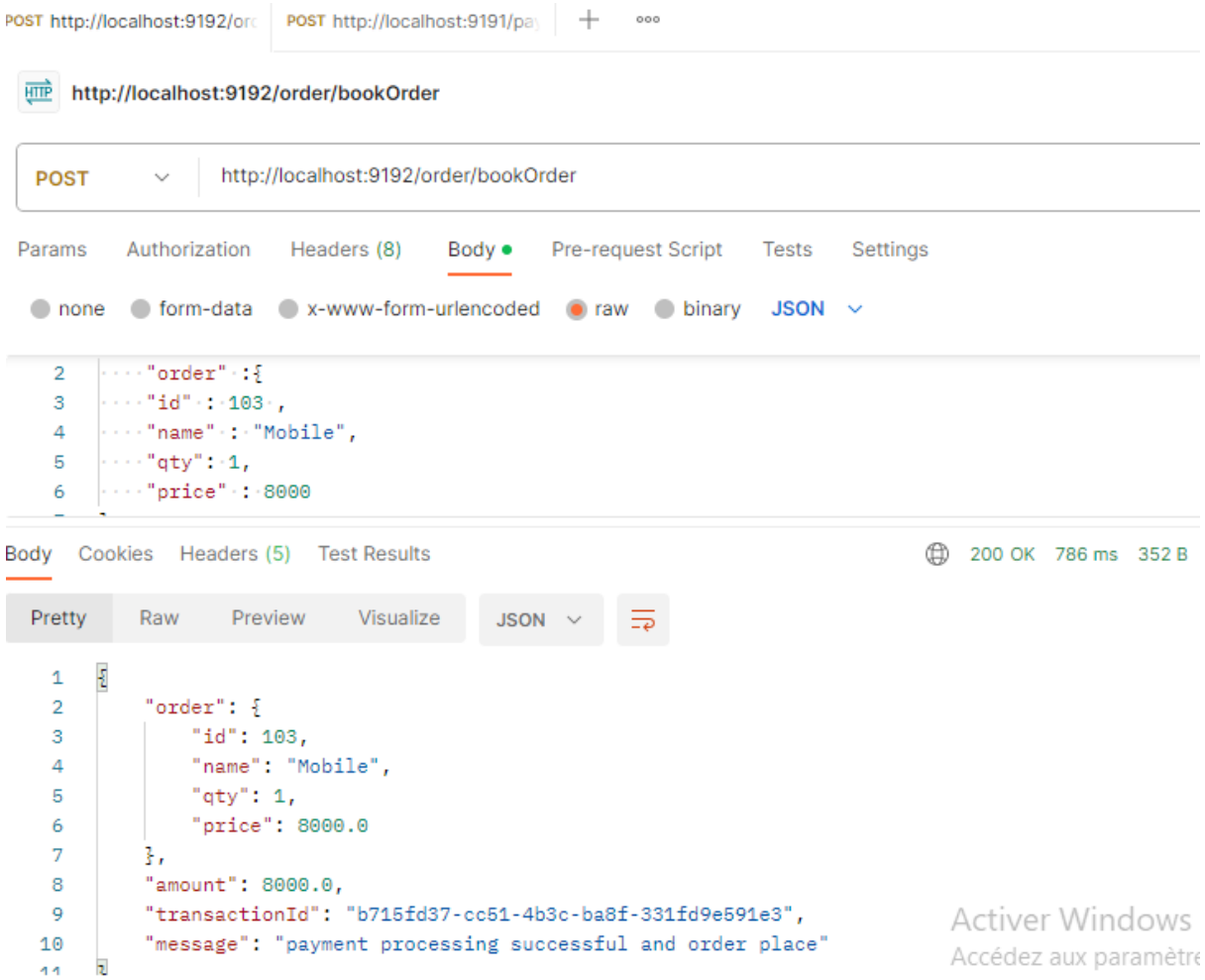
Creating Order Service :



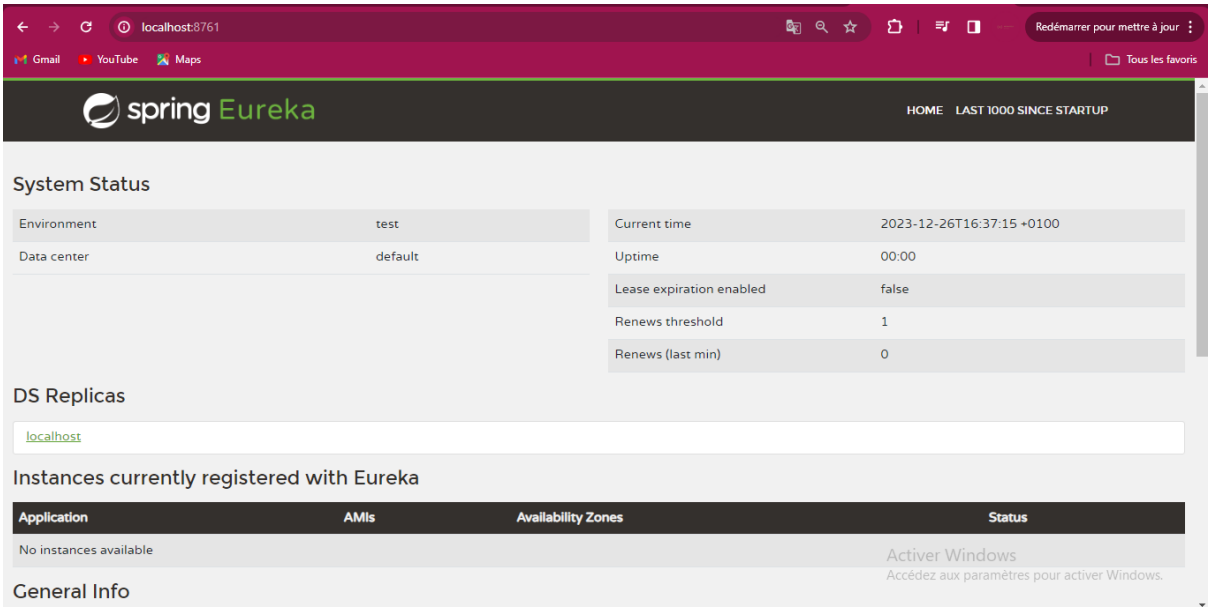
Creating Payment Service :



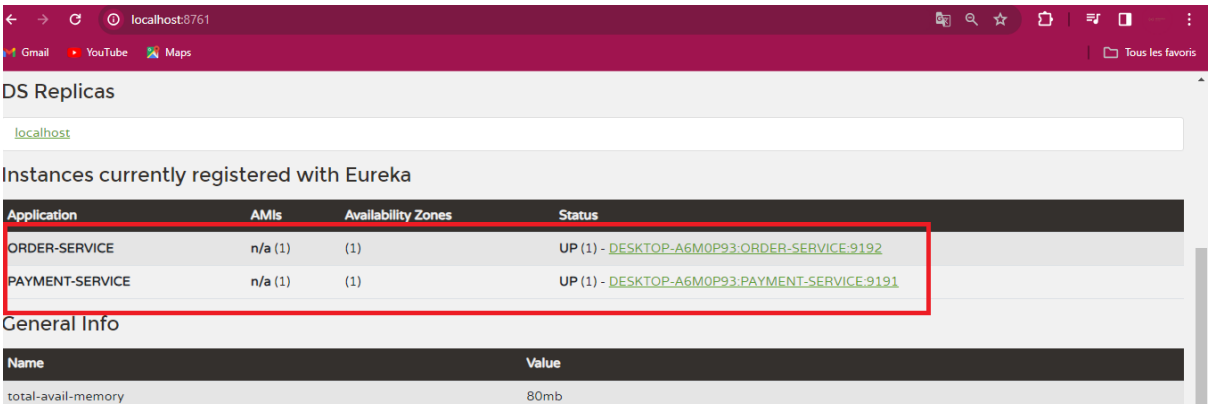
Let's check whether our microservices have connected successfully :



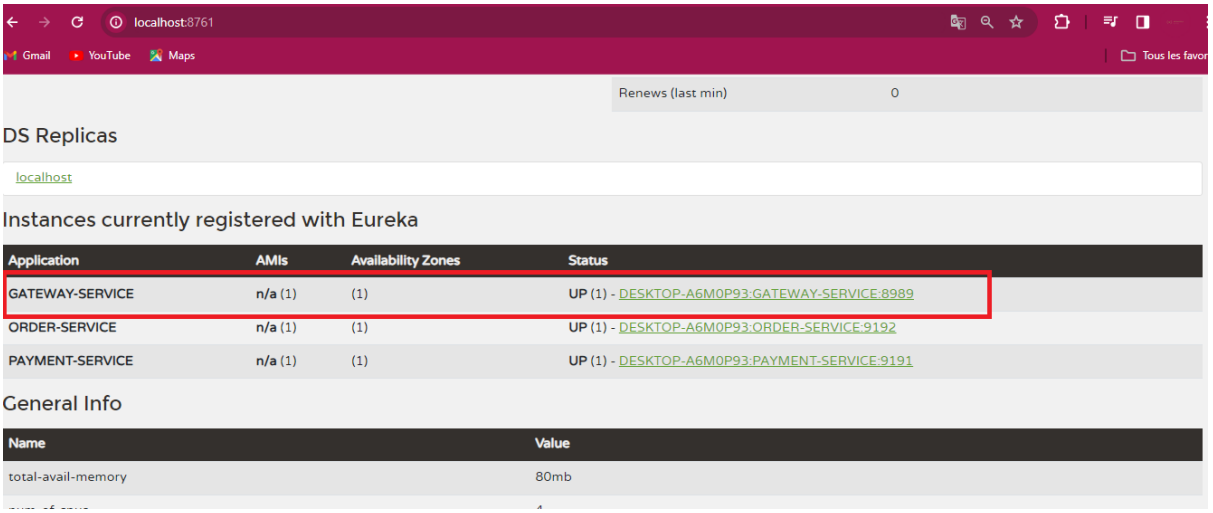
2 : ***Register microservice in Eureka Service Discovery :***



Let's register our microservice instance with Eureka :



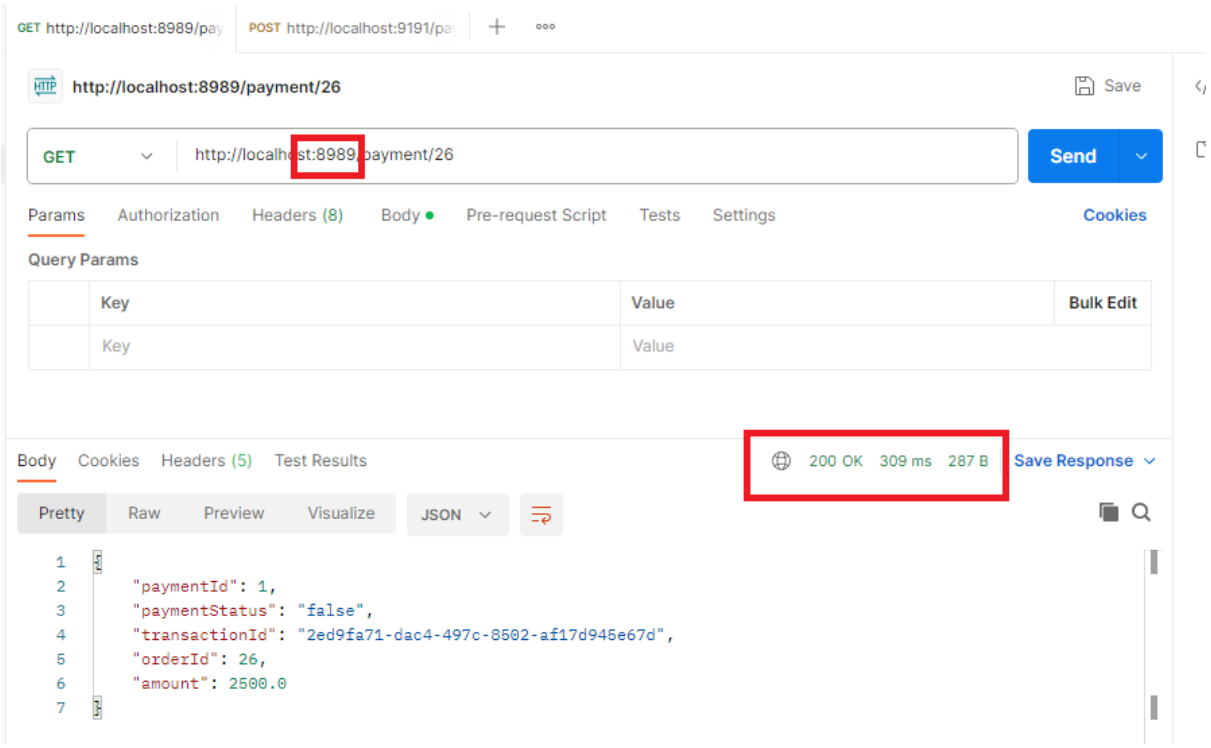
3 : integrate Spring Cloud Gateway for routing user requests :



Let's verify if the user request from the microservices successfully passes through the API gateway :

AMOUNT	ORDER_ID	PAYMENT_ID	PAYMENT_STATUS	TRANSACTION_ID
2500.0	26	1	false	2ed9fa71-dac4-497c-8502-af17d945e67d

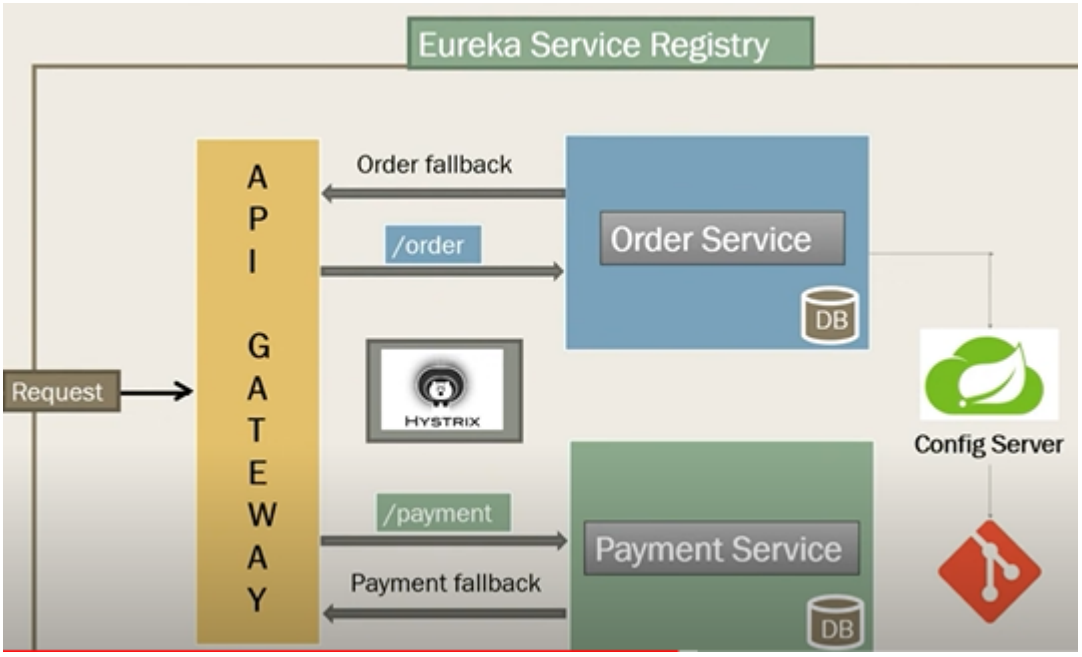
(1 row, 13



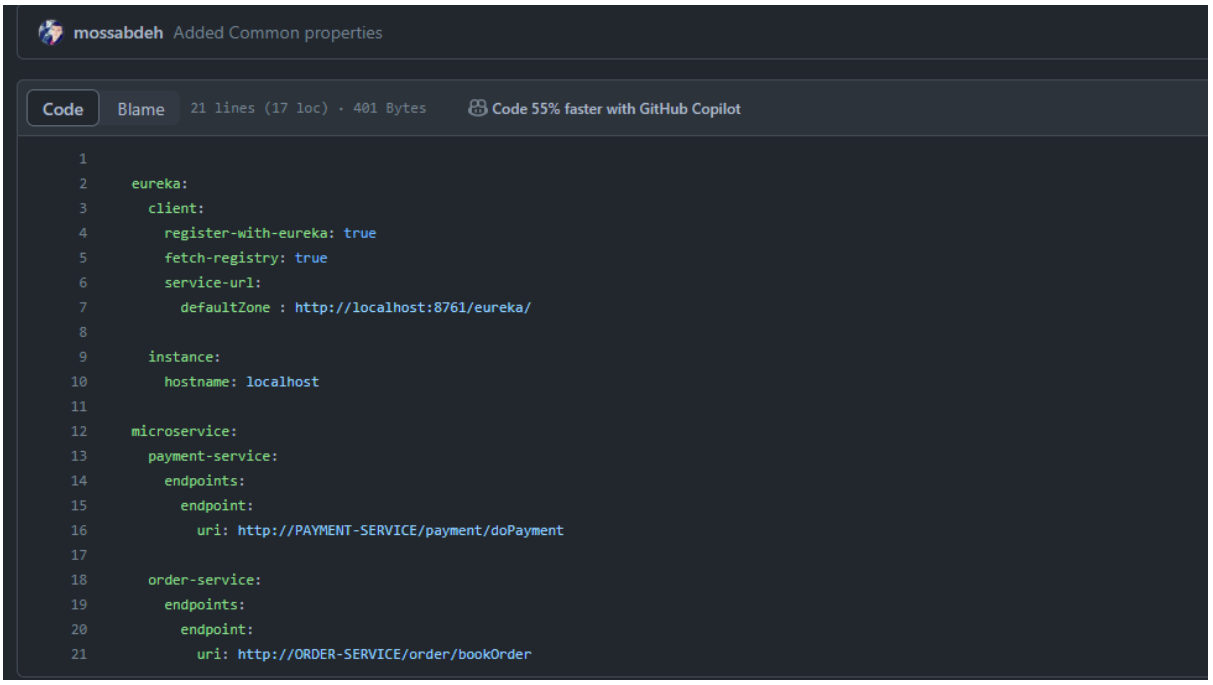
4 : Integrate Hystrix & Hystrix Dashboard to identify failure for downstream services :

Spring Cloud Hystrix project is deprecated. So new applications should not use this project. Resilience4j is a new option for Spring developers to implement the circuit breaker pattern. Resilience4j comes with other features like Rate Limiter, Retry and Bulkhead along with Circuit Breaker pattern. 10 déc. 2019

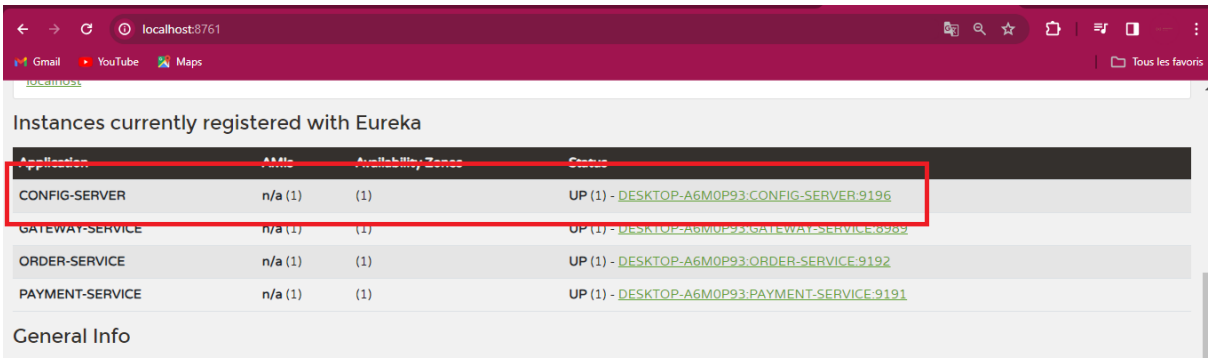
5 : Spring cloud config server using Git to Centralize configuration across applications :



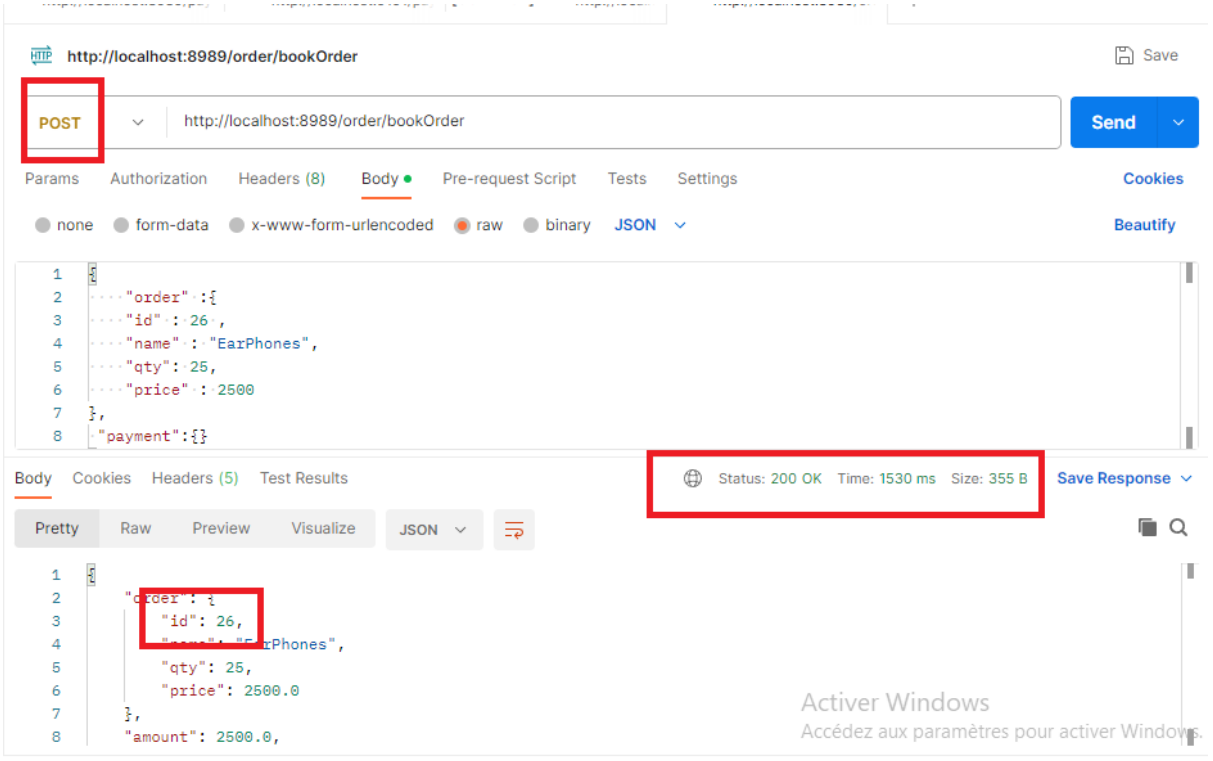
Let's establish a Git repository to centralize configuration using Spring Cloud Config :

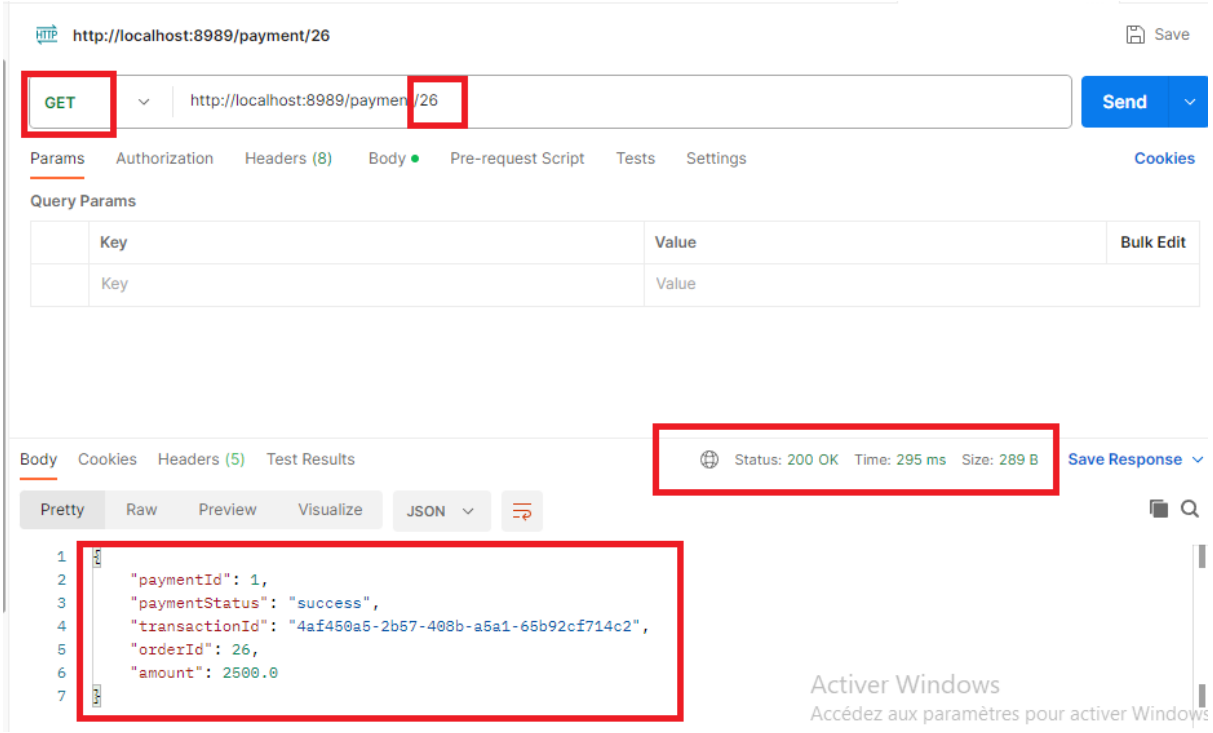


Let's register our spring cloud config instance with Eureka :

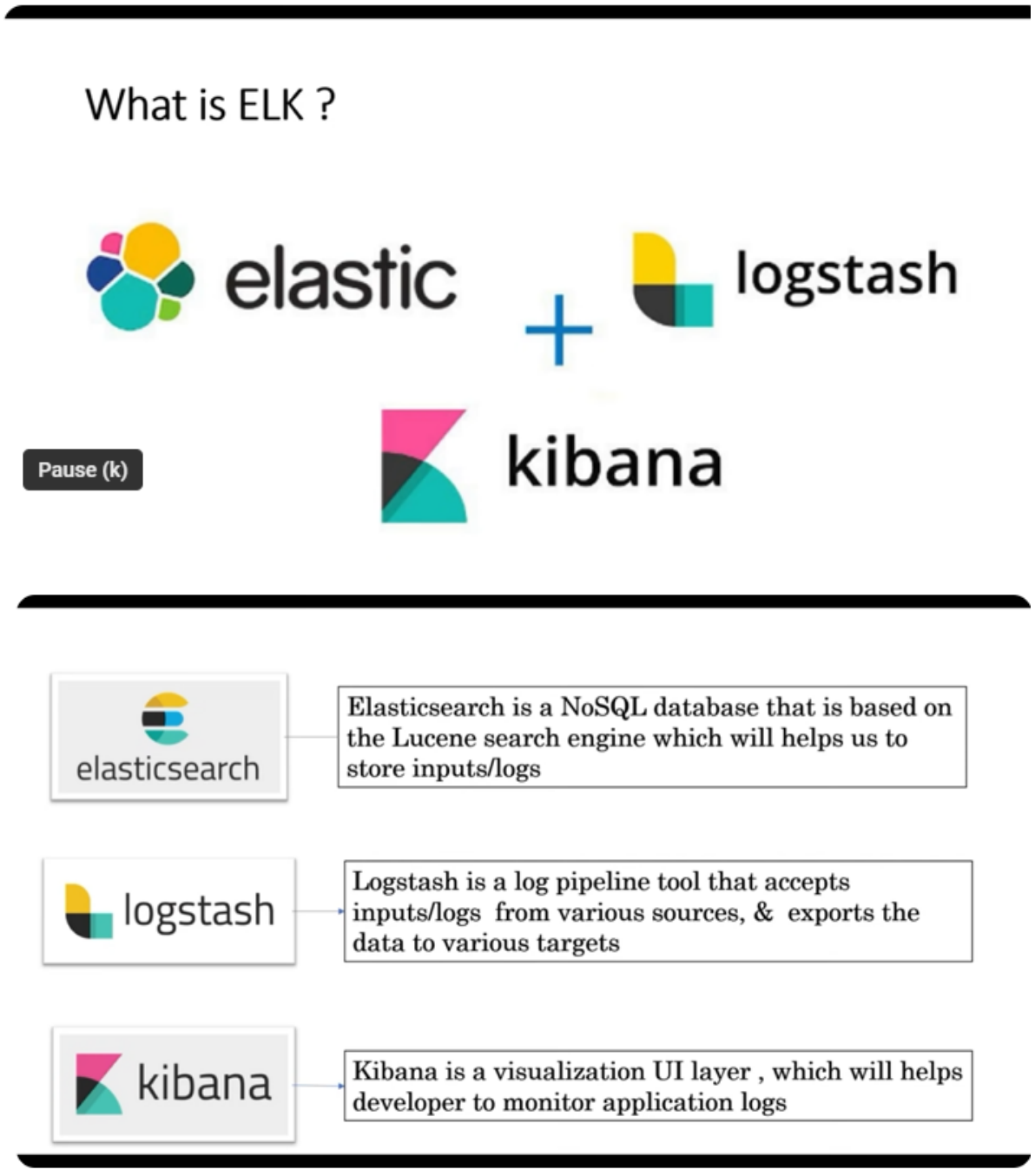


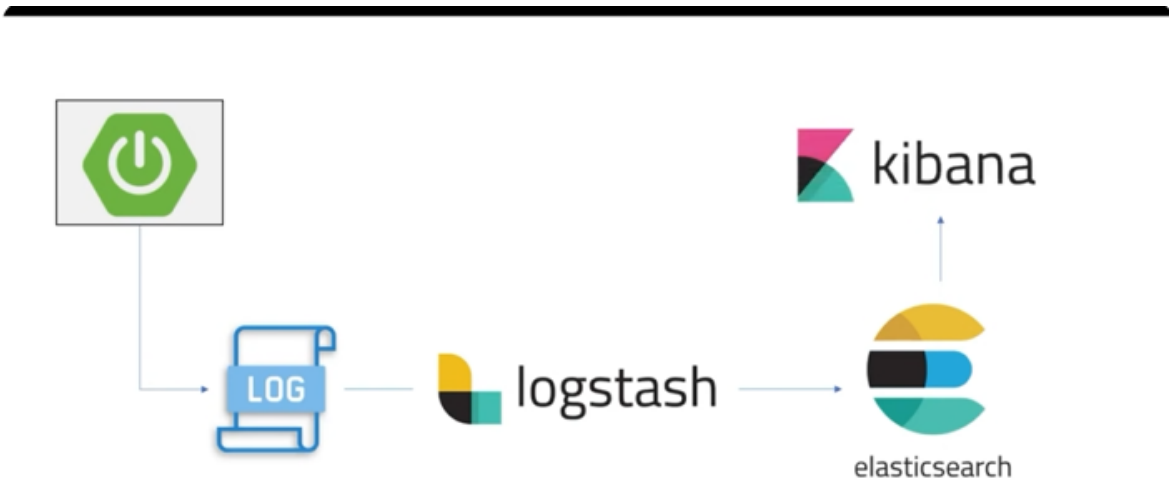
Let's check whether our microservices have connected successfully using spring cloud config :





6 : ***ELK Stack to centralize logging across all microservices. :***



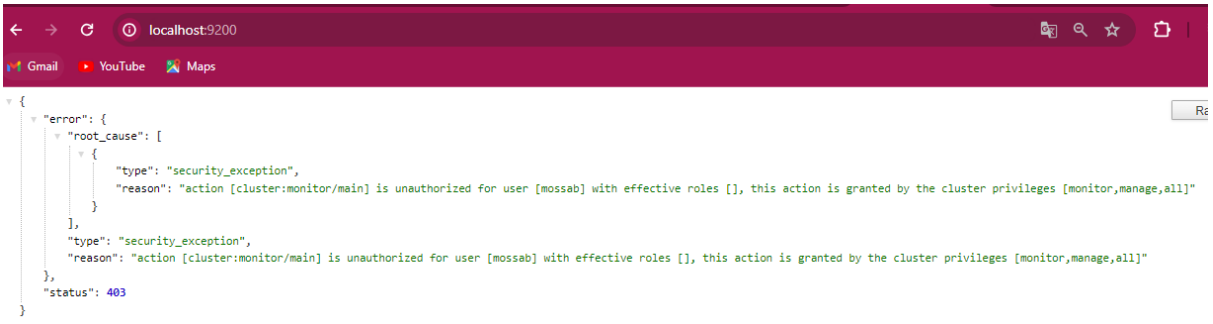


Let's start with ElasticSearch ：

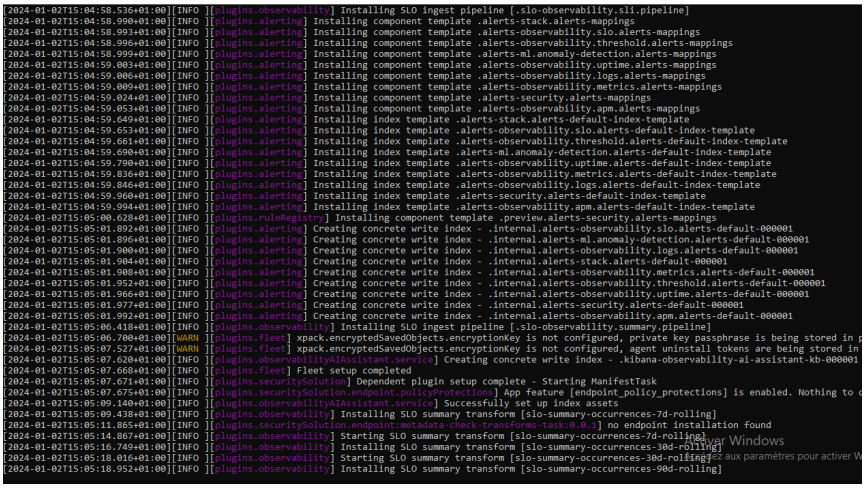
turn on server ：

works fine on windows (default port of elastic search is 9200)

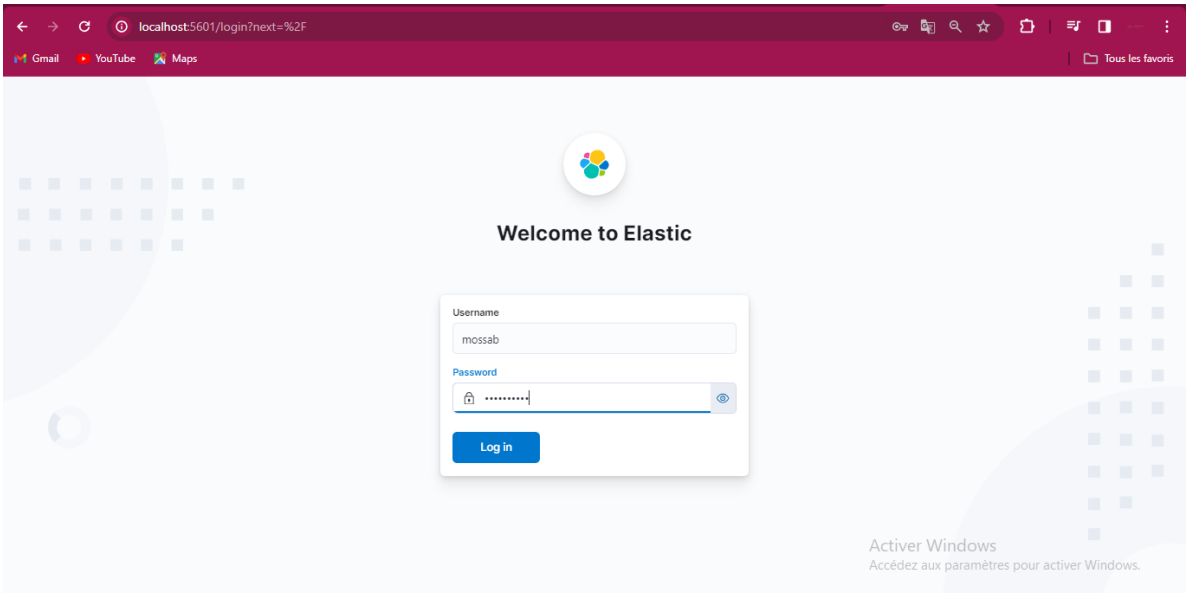
and the system username = elastic ：



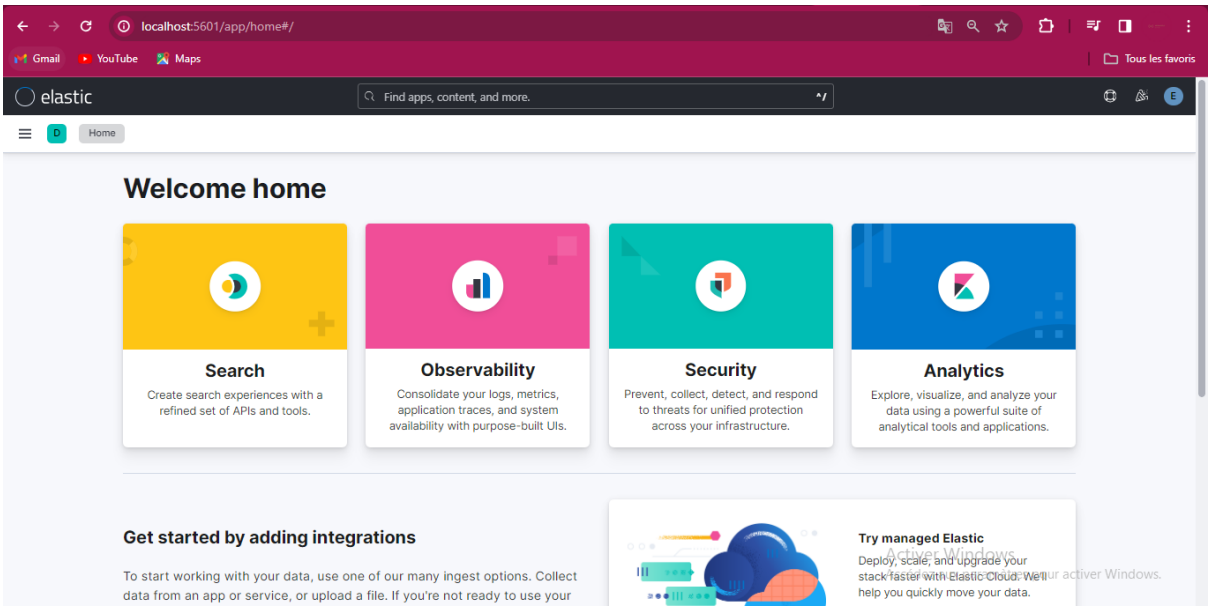
Let's Configure Kibana Now :
first start the web server kibana.bat



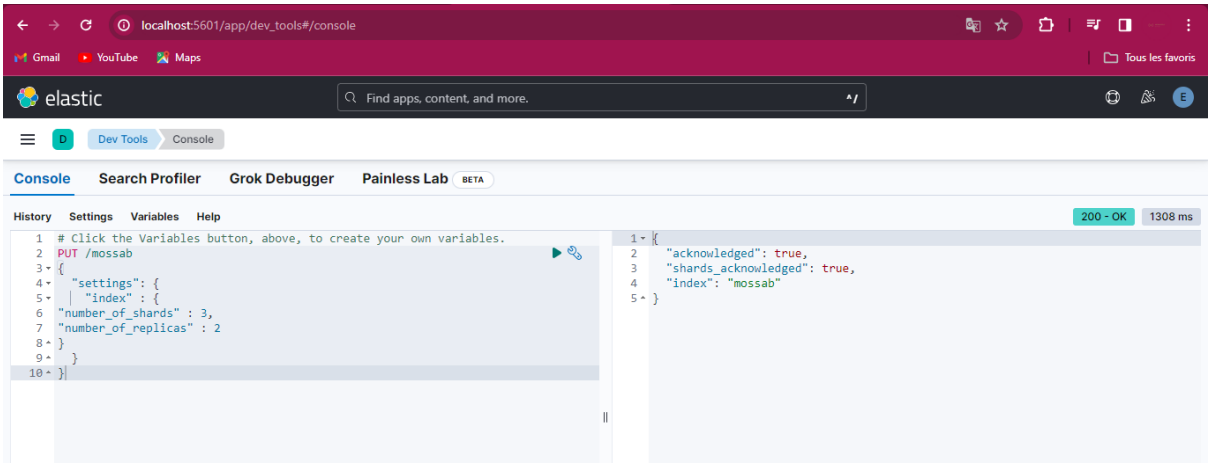
Configure kabana.yml using built in user = kibana_system and go to http://localhost:5601/ you should see that login interface :



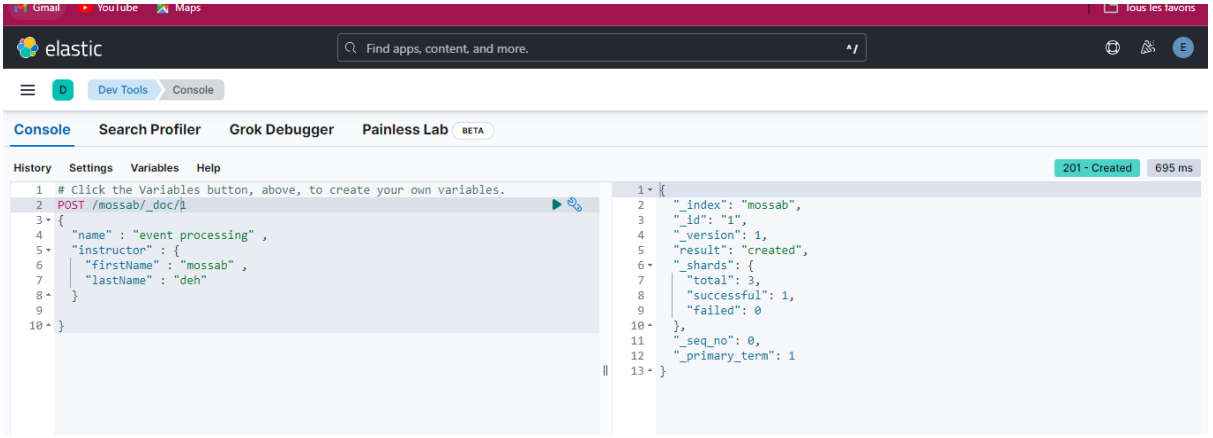
you can’t access using custom user , you need to access as root system user which is : elasticsearch.username: "elastic"



Let’s create index : using kibana dev tools

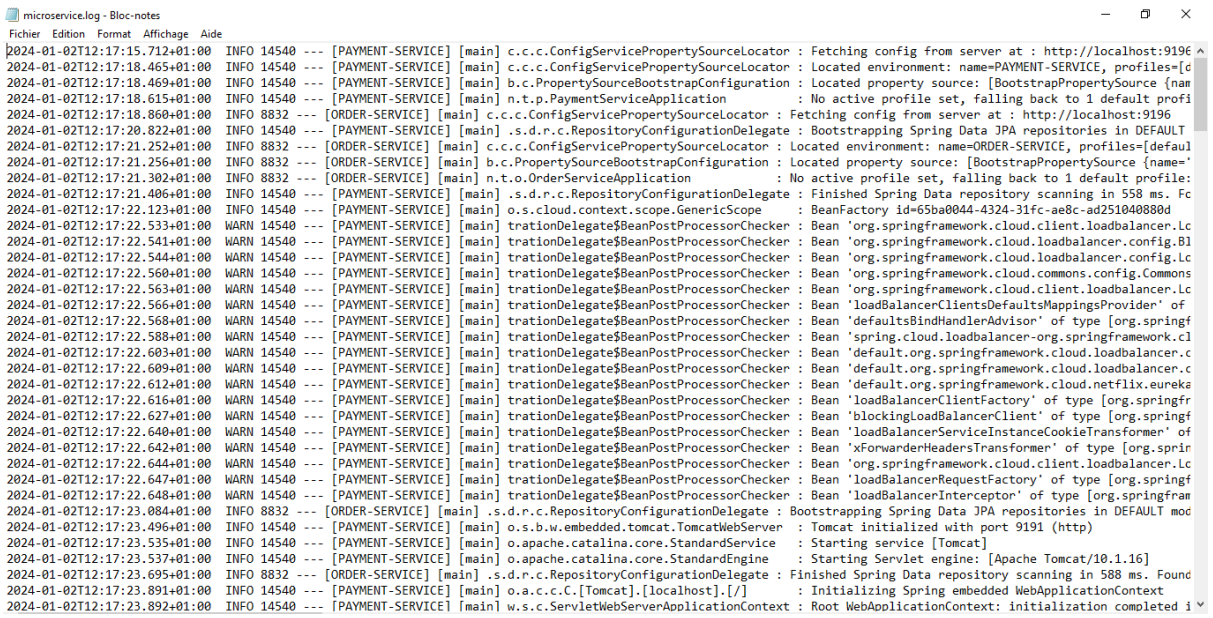


Let’s add document to this index :

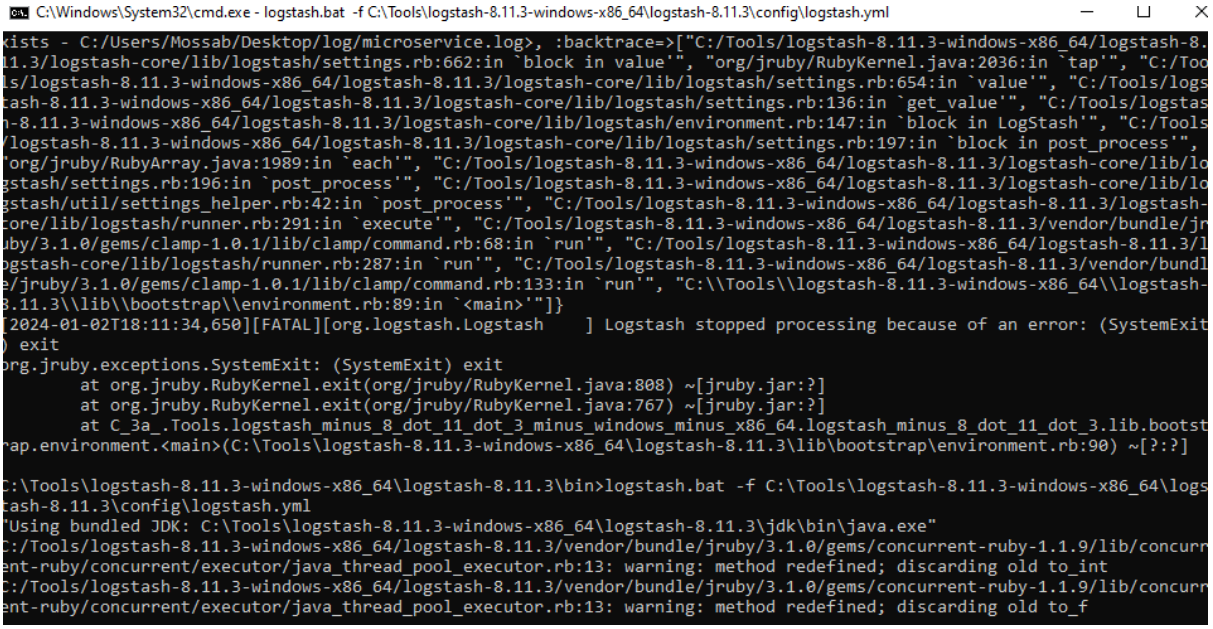


Let's Configure LogStash Now :

we have already log from our microservices :



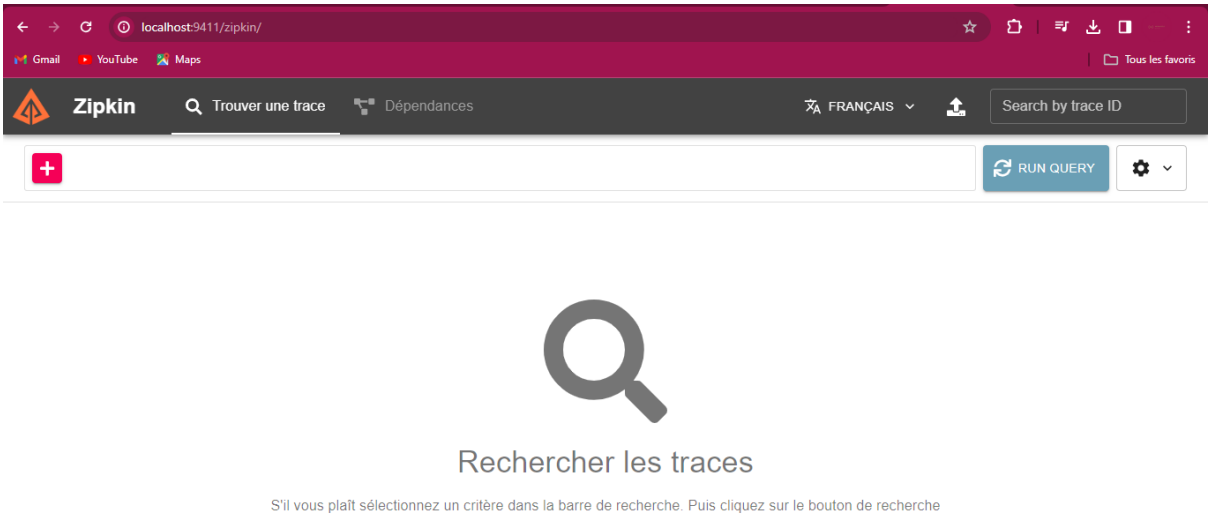
Let’s start the logstash web server :



7 : Zipkin & Sleuth to centralize tracing in microservice architecture. :

Let’s start the zipkin web server using its JAR file :

we access to zipkin dashboard using the url : localhost:9411 :



Let’s check if our zipkin working fine by accessing our microservices API :