**Circuit Breaker**

(Hystrix)

**Observability**

(Zipkin, Prometheus, ELK)

**Service Discovery**

**API Gateway**

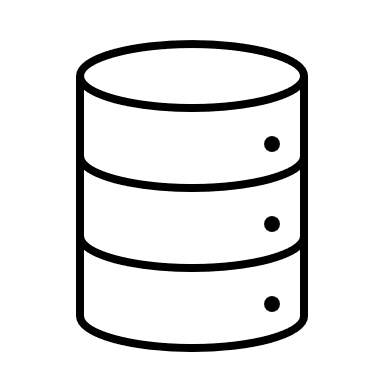
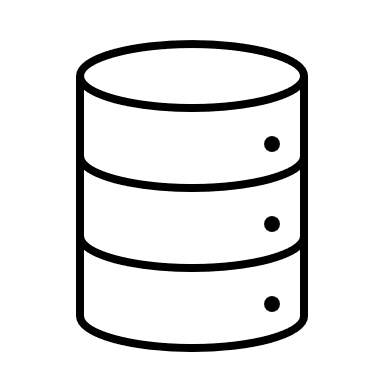
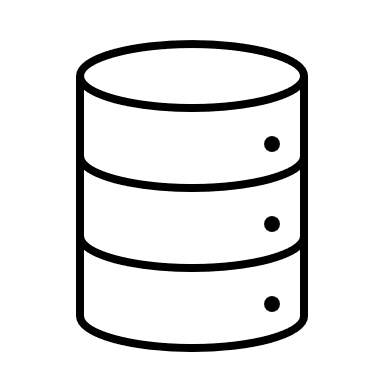
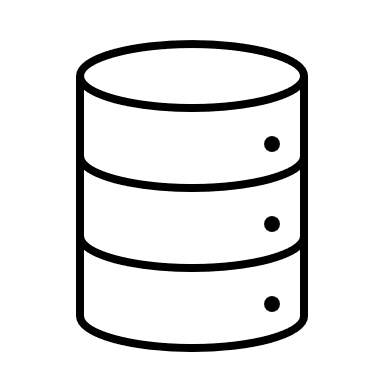
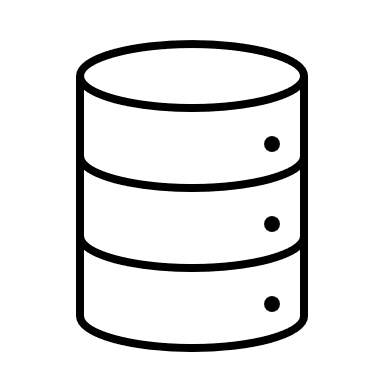
**Order Processing Service**

**Product Catalog Service**

**Inventory Service**

**Customer Service**

**Payment Service**



**Productdb**

**Payment**db

**Inventorydb**

**Customerdb**

**Orderdb**

**Asynchronous Messaging**

(Kafka / RabbitMQ)

**DevOps Tools**

(Jenkins, Docker, K8s, Terraform)

Config Server:

Configure git by config import and enabled Eureka server

spring.cloud.config.server.git.uri=https://github.com/LP2741/config\_server\_properties.git

Eureka Server:

Register services and tracks all service instances automatically

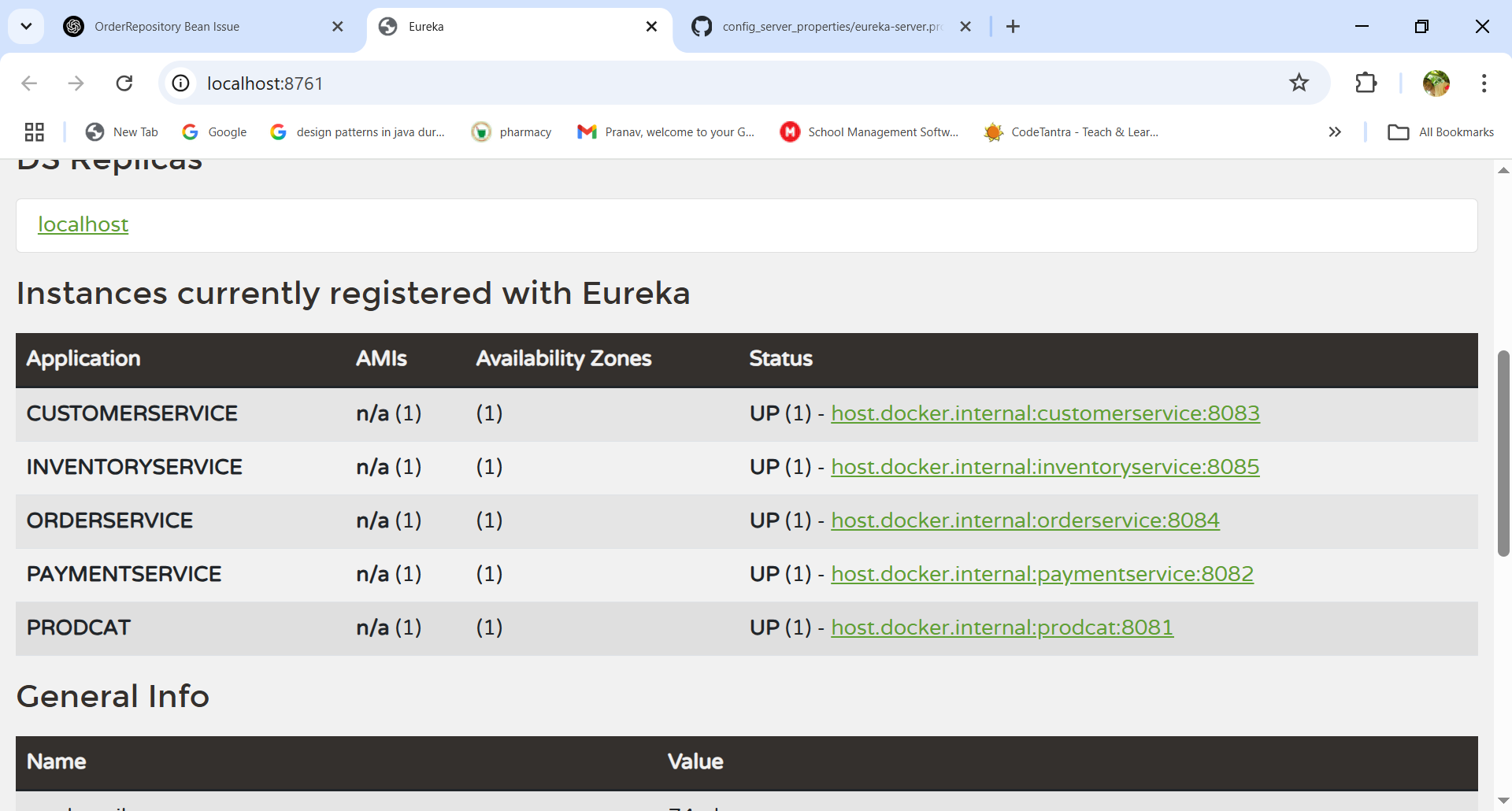
Apigateway:

-> Single entry point for all services

-> Eureka server configured

->Zipkin trace configured

Eureka Url: <http://localhost:8761/>



prodcat:

Create products by using

Post : <http://localhost:9191/prodcat/api/v1/create/products>

Content-Type application/json

Json:

{

  "id": 150,

  "name": "1 tasre toste",

  "description":"baking Cake",

  "stock":45,

  "price":250.00

}

customerservice:

Post : <http://localhost:9191/customerservice/api/customers/create>

Json:

{

  "name": "Lakshmi Prasanna3",

  "email": "laxmi.\_2741@yahoo.co.in",

  "address": "Hyderabad"

}

Order Service:

Post : <http://localhost:9191/orderservice/api/orders/place>

Cookie auth\_token=securetoken123

{

  "customerId": "686bc9993edc324c0c8baaeb",

  "items": [

    { "productId": "686cb7d2b5c3877b2b6643e8", "quantity": 4 }

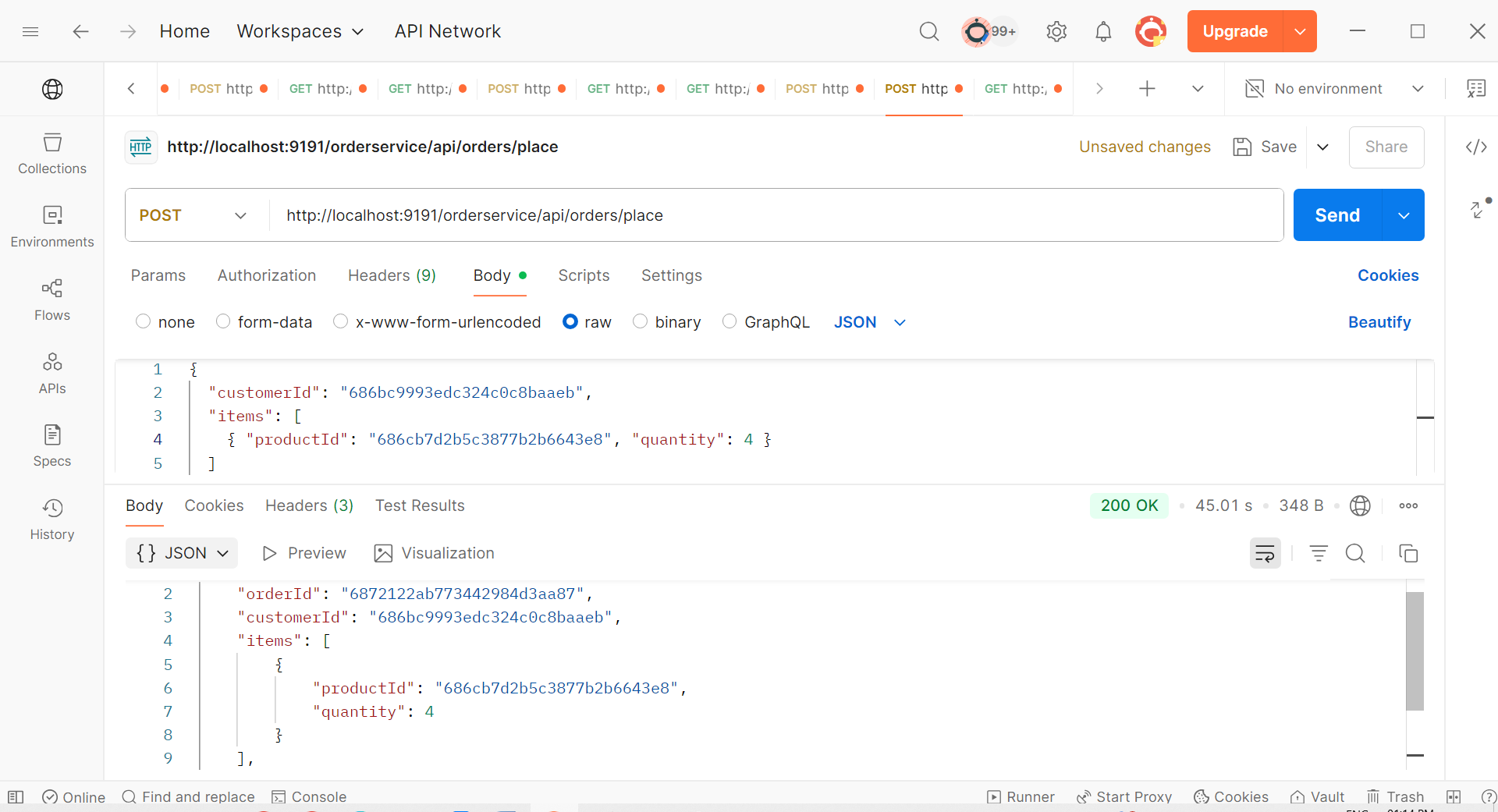
  ]

}

While placing order checking product stock with

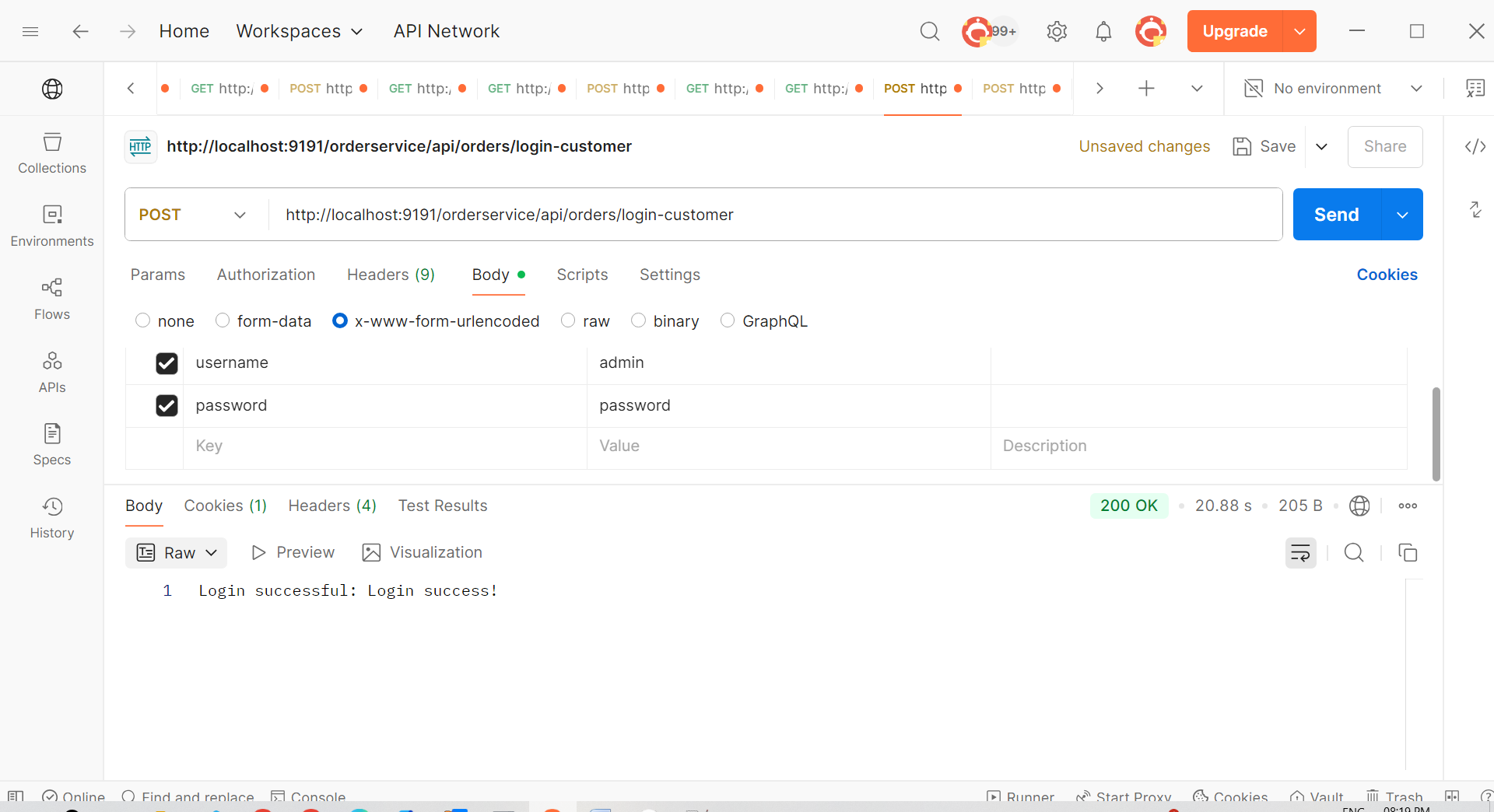
[http://localhost:9191/prodcat/api/v1/checkstock/{id}](http://localhost:9191/prodcat/api/v1/checkstock/%7bid%7d)

And then placing the order.



Functionality is not included just mock created

<http://localhost:9191/orderservice/api/orders/login-customer>



Inventoryservice:

If there is no stock we should go with inventory

[http://localhost:9191/inventoryservice/api/inventory/ stockcheck/686cb7d2b5c3877b2b6643e9/45](http://localhost:9191/inventoryservice/api/inventory/customercheck/686cb7d2b5c3877b2b6643e9/45)

We are checking the order id and customer and the allowing payment creation

If product is not available ,creating product

http://localhost:9191/prodcat/api/v1/create/products

If product is available and stock is less we are updating stock

http://localhost:9191/prodcat/api/v1/update

Payementservice:

We are checking the order id and customer and the allowing payment creation

<http://localhost:9191/paymentservice/api/payments/create>

{

  "orderId": "686ff1610c56ee7430509910",

  "customerId": "686bc9993edc324c0c8baaeb",

  "amount": 499.99,

  "paymentMethod": "UPI"

}

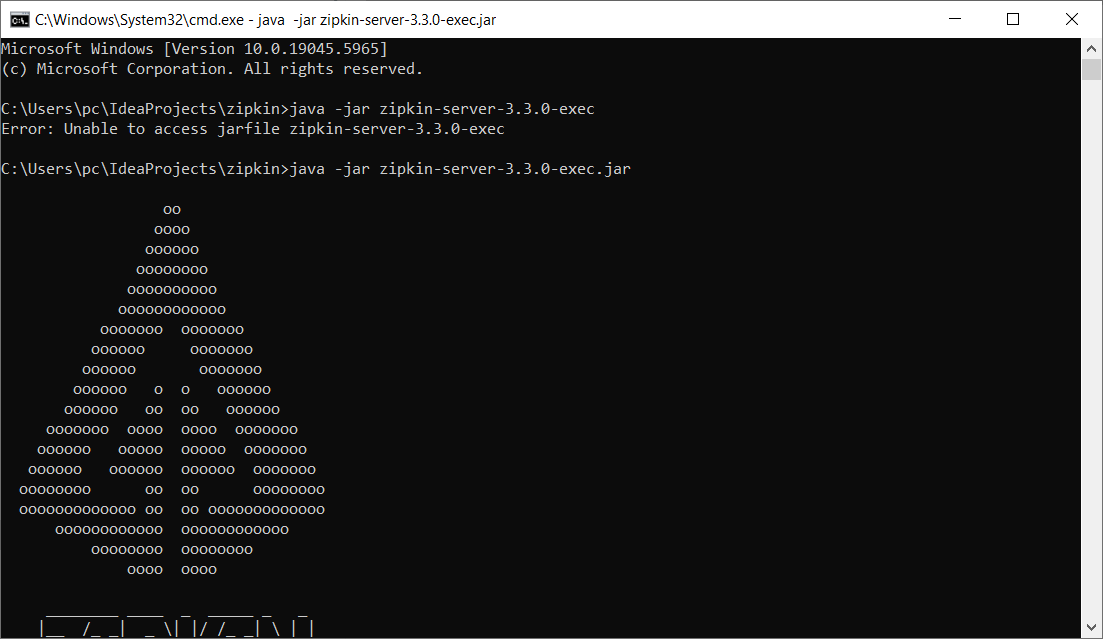
Customer checking:

http://localhost:9191/customerservice/api/customers/686bc9993edc324c0c8baaeb

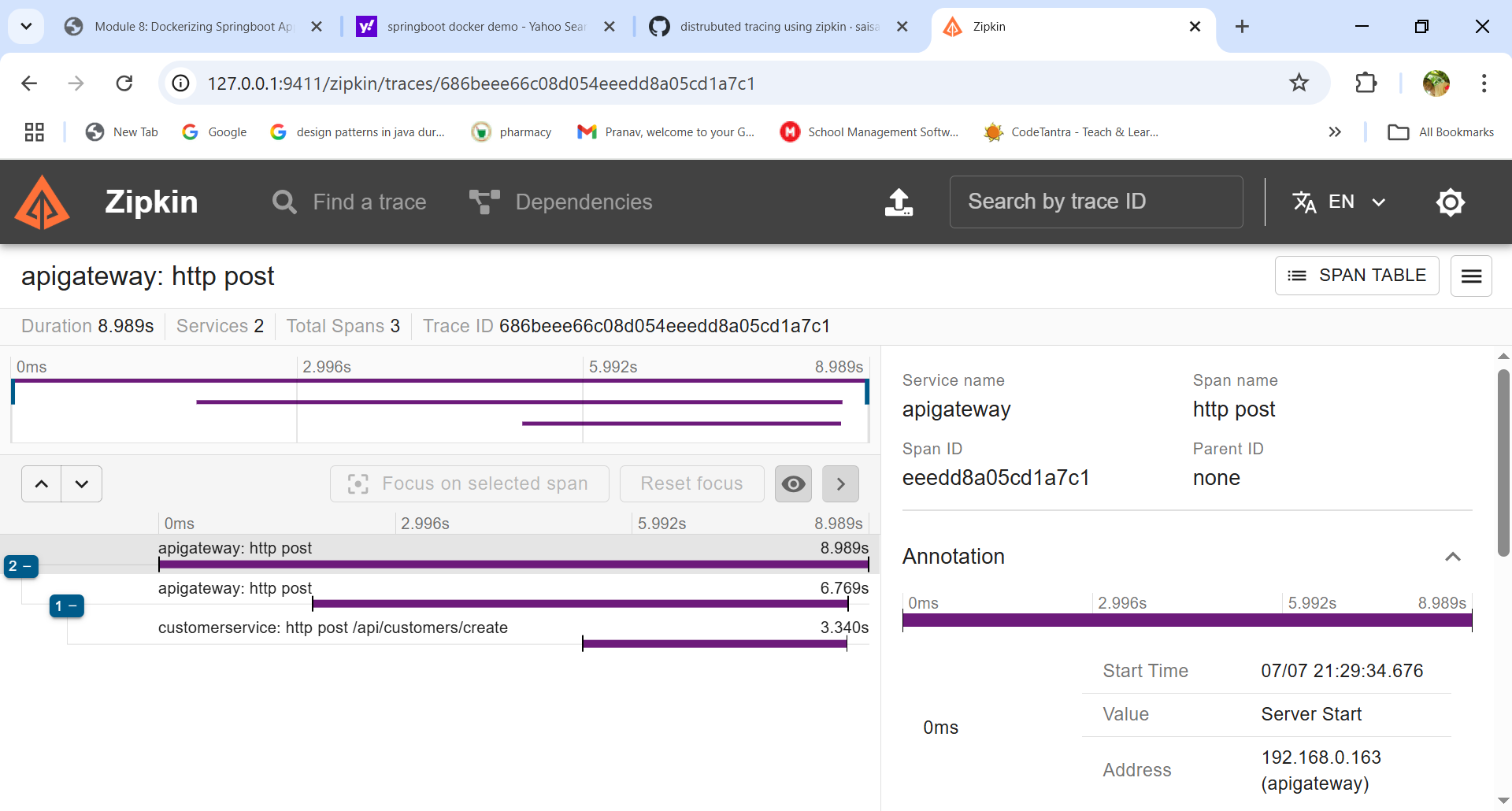
Order checking:

http://localhost:9191/orderservice/api/orders/ 686ff1610c56ee7430509910

zipkin



http://127.0.0.1:9411/zipkin/?serviceName=customerservice&lookback=15m&endTs=1752331338231&limit=10



Start docker

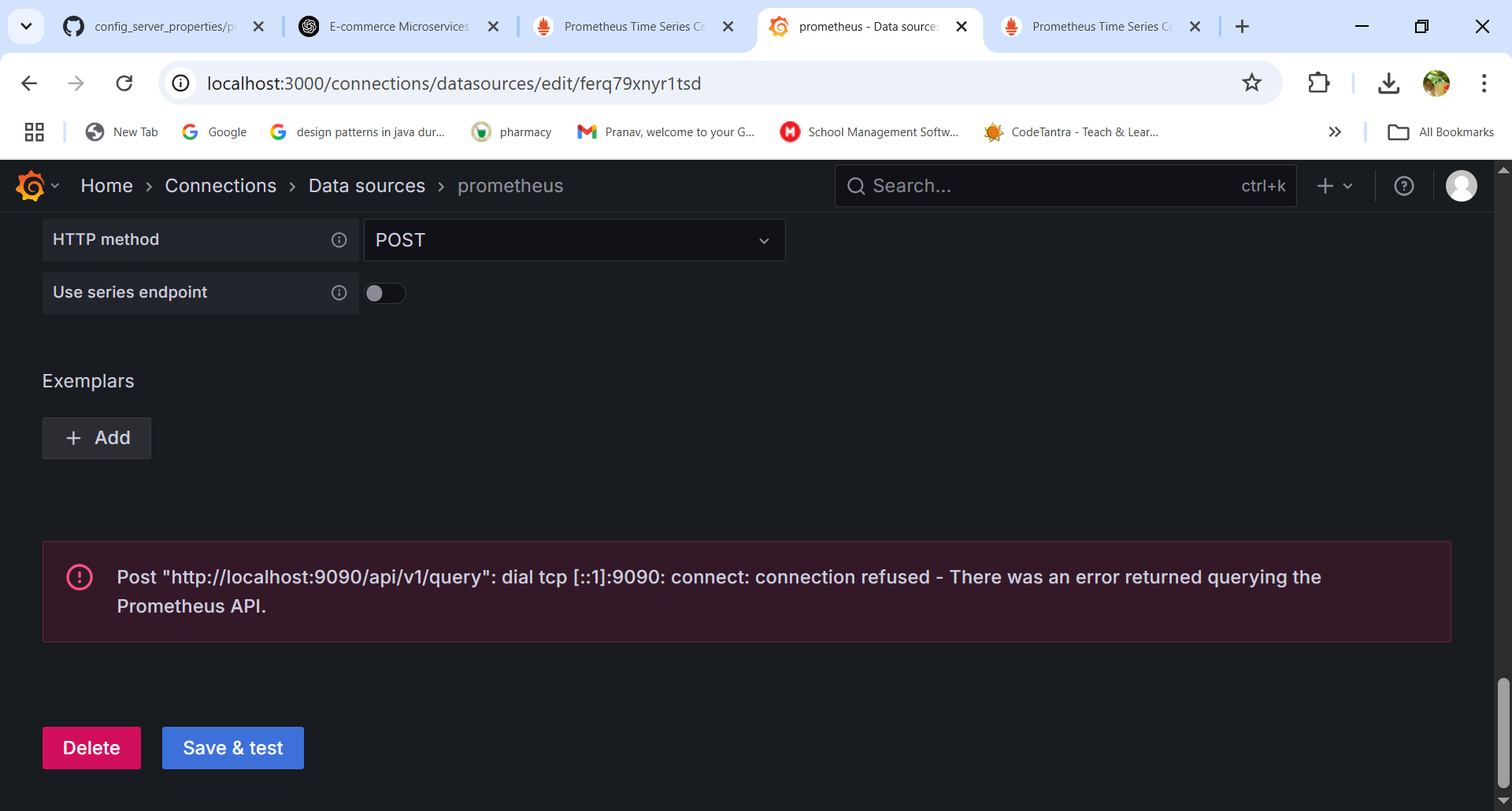
----------------------------------------------------

Grafana

-------------------------------------------------------------

docker run -d -p 3000:3000 grafana/grafana

http://localhost:3000/dashboard/new?orgId=1&from=now-6h&to=now&timezone=browser&editPanel=1



promentheus

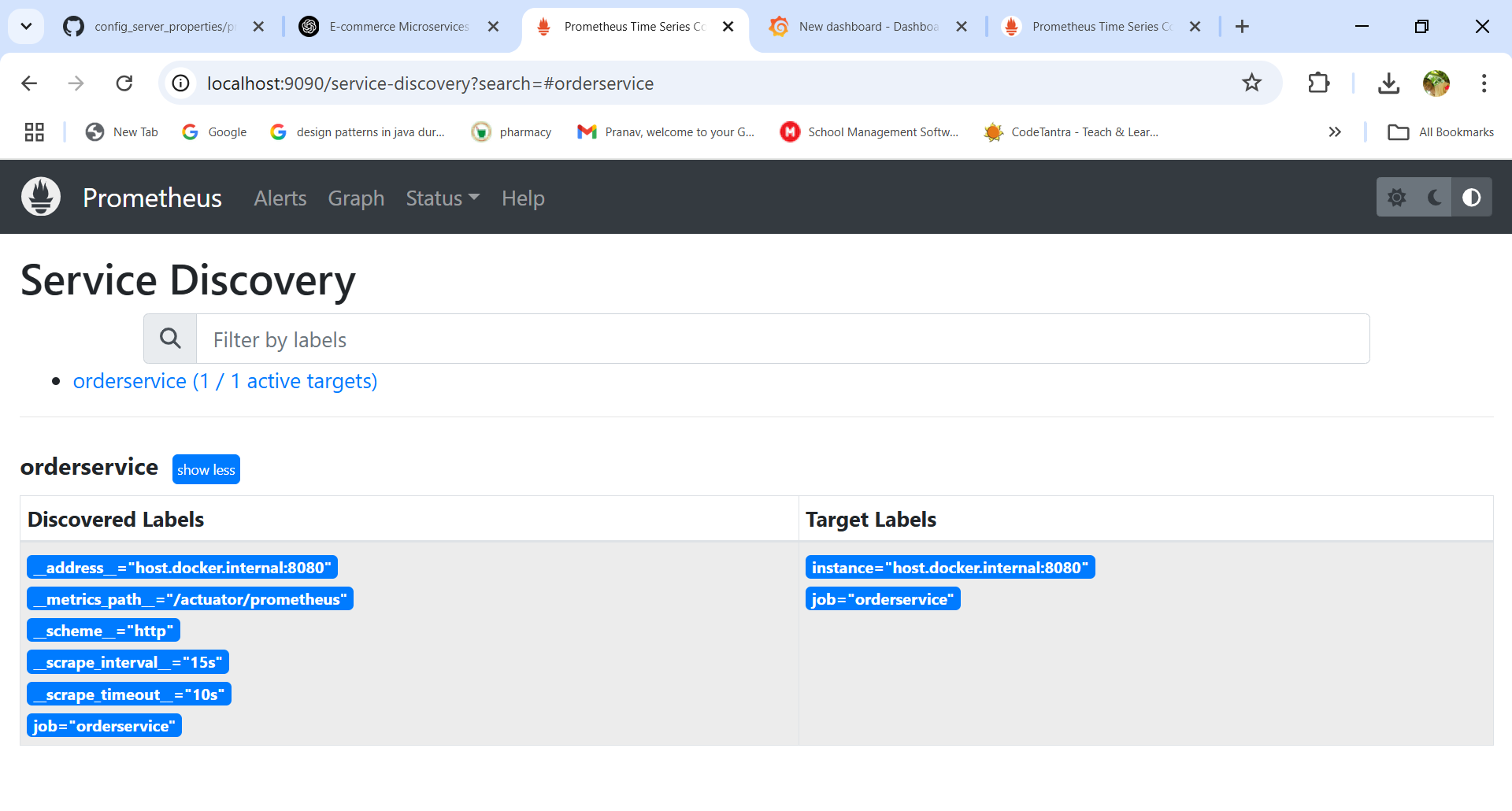
docker run -d -p 9090:9090 `

-v ${PWD}\prometheus.yml:/etc/prometheus/prometheus.yml `

prom/Prometheus

http://localhost:9090/graph?g0.expr=&g0.tab=0&g0.display\_mode=lines&g0.show\_exemplars=0&g0.range\_input=1h&g0.end\_input=2025-07-12%2016%3A41%3A31&g0.moment\_input=2025-07-12%2016%3A41%3A31

<http://localhost:9090/service-discovery?search=#orderservice>



Not able to open docker

------------------------------------------------------

Docker cli

docker run -d --name zookeeper -p 2181:2181 zookeeper

docker run -d --name kafka -p 9092:9092 -e KAFKA\_ZOOKEEPER\_CONNECT=zookeeper:2181 -e KAFKA\_ADVERTISED\_LISTENERS=PLAINTEXT://localhost:9092 -e KAFKA\_LISTENER\_SECURITY\_PROTOCOL\_MAP=PLAINTEXT:PLAINTEXT -e KAFKA\_OFFSETS\_TOPIC\_REPLICATION\_FACTOR=1 --network host confluentinc/cp-kafka

Tried from Program files

docker pull bitnami/zookeeper:latest && docker pull bitnami/kafka:latest

docker run -d --name zookeeper -e ALLOW\_ANONYMOUS\_LOGIN=yes -p 2181:2181 bitnami/zookeeper:latest

docker run -d --name kafka --link zookeeper:zookeeper -e KAFKA\_ZOOKEEPER\_CONNECT=zookeeper:2181 -e KAFKA\_LISTENER\_SECURITY\_PROTOCOL=PLAINTEXT -e KAFKA\_LISTENER\_PORT=9093 -e KAFKA\_ADVERTISED\_LISTENER=PLAINTEXT://localhost:9093 -p 9093:9093 bitnami/kafka:latest

docker exec -it kafka kafka-topics.sh --create --topic order-topic --bootstrap-server localhost:9092 --partitions 1 --replication-factor 1

