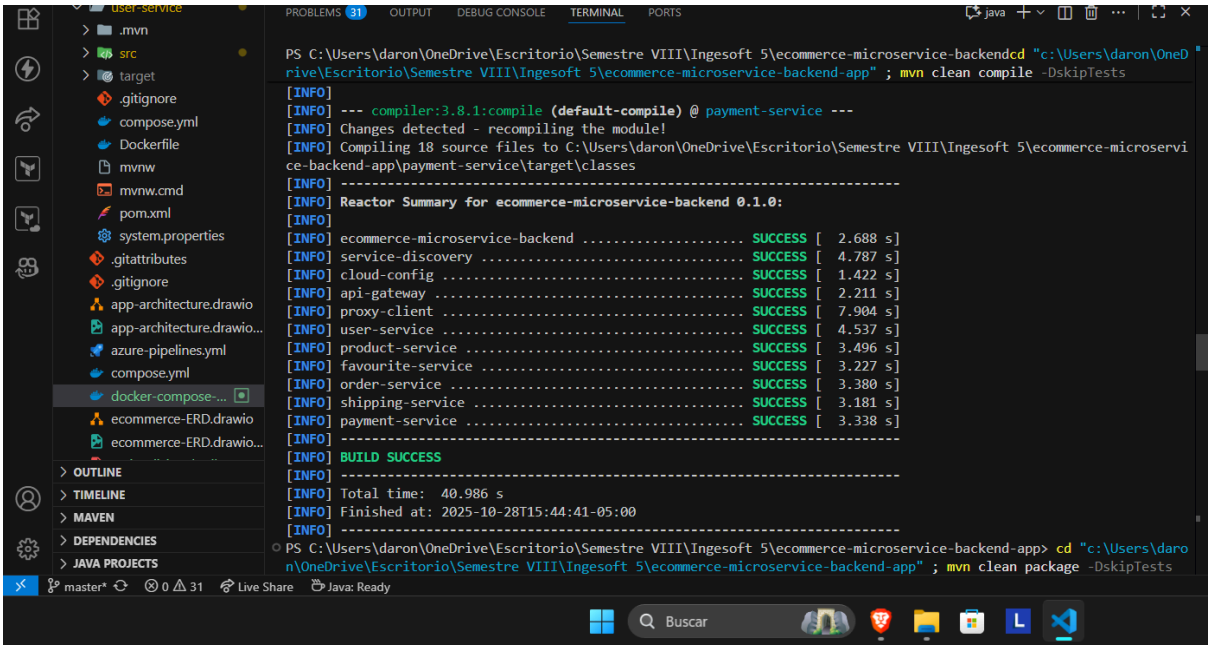
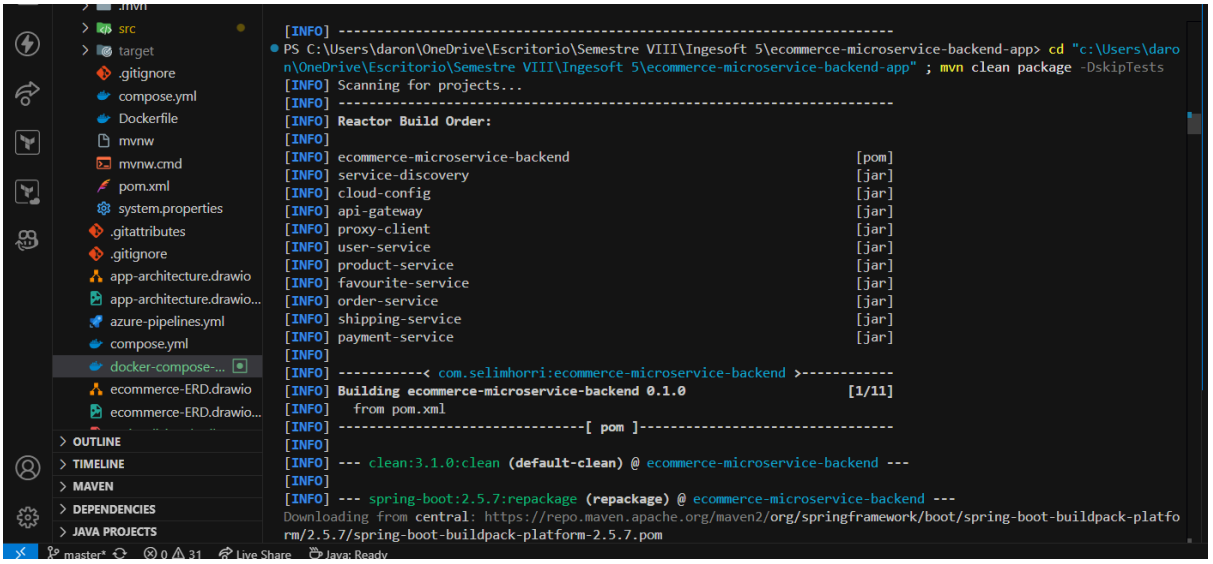
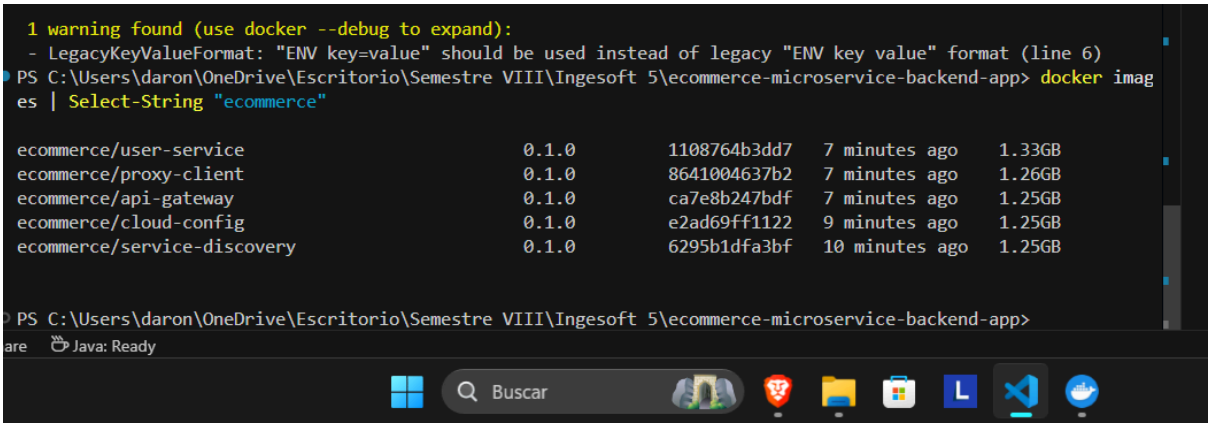
Se construye el proyecto:



se crean los JAR de los microservicios:



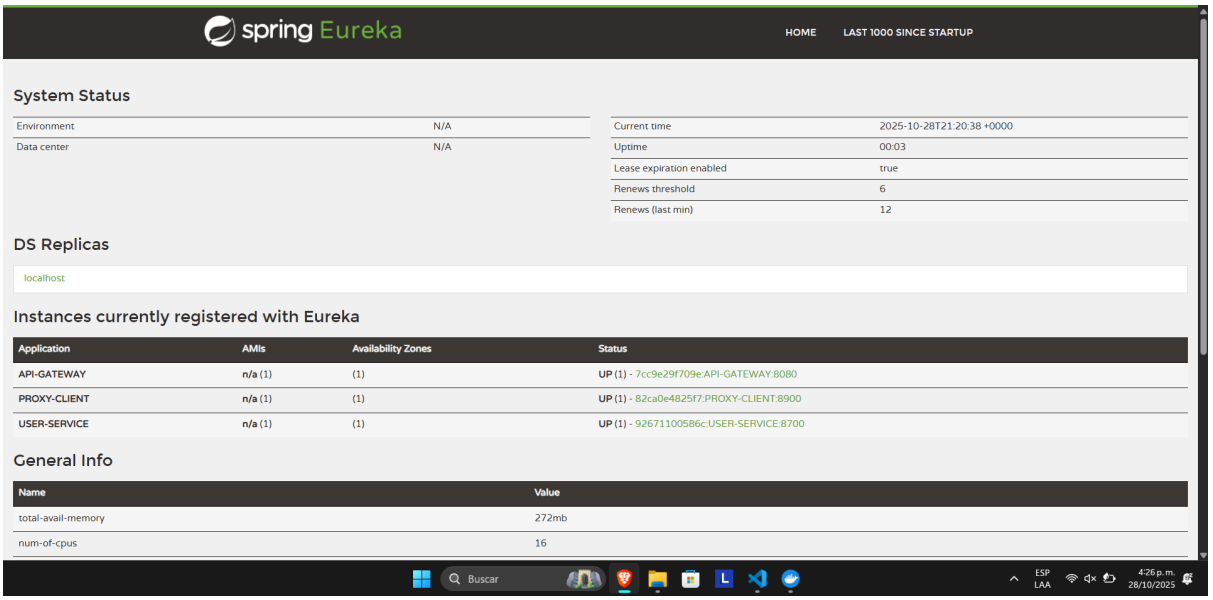
Las imagnes creadas en docker:



Contenedores arriba:

```
PS C:\Users\daron\OneDrive\Escritorio\Semestre VIII\Ingesoft 5\ecommerce-microservice-backend-app> cd "c:\Users\daron\OneDrive\Escritorio\Semestre VIII\Ingesoft 5\ecommerce-microservice-backend-app" ; docker-compose -f docker-compose-orchester.yml up -d
time="2025-10-28T16:16:43-05:00" level=warning msg="C:\\Users\\daron\\OneDrive\\Escritorio\\Semestre VIII\\Ingesoft 5\\ecommerce-microservice-backend-app\\docker-compose-orchester.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Running 7/7
 ✓ Network ecommerce-network     Created                                    0.1s
 ✓ Container zipkin-server       Healthy                                   12.7s
 ✓ Container service-discovery   Healthy                                   34.7s
 ✓ Container cloud-config        Healthy                                   55.4s
 ✓ Container proxy-client        Started                                   55.9s
 ✓ Container api-gateway         Started                                   55.7s
 ✓ Container user-service        Started                                   55.9s
PS C:\Users\daron\OneDrive\Escritorio\Semestre VIII\Ingesoft 5\ecommerce-microservice-backend-app>
```

se verifica los servicios inscritos a eureka:



primer error: El cloud-config está intentando conectarse a localhost:8761 en lugar de service-discovery:8761

```
2025-10-28 21:17:36.670  INFO [CLOUD-CONFIG,,] 1 --- [           main]
c.n.d.s.t.d.RedirectingEurekaHttpClient  : Request execution error. endpoint=DefaultEndpoint{
serviceUrl='http://localhost:8761/eureka/}, exception=I/O error on GET request for
"http://localhost:8761/eureka/apps/": Connect to localhost:8761 [localhost/127.0.0.1,
localhost/0:0:0:0:0:0:0:1] failed: Connection refused (Connection refused); nested exception is
org.apache.http.conn.HttpHostConnectException: Connect to localhost:8761 [localhost/127.0.0.1,
localhost/0:0:0:0:0:0:0:1] failed: Connection refused (Connection refused)
```

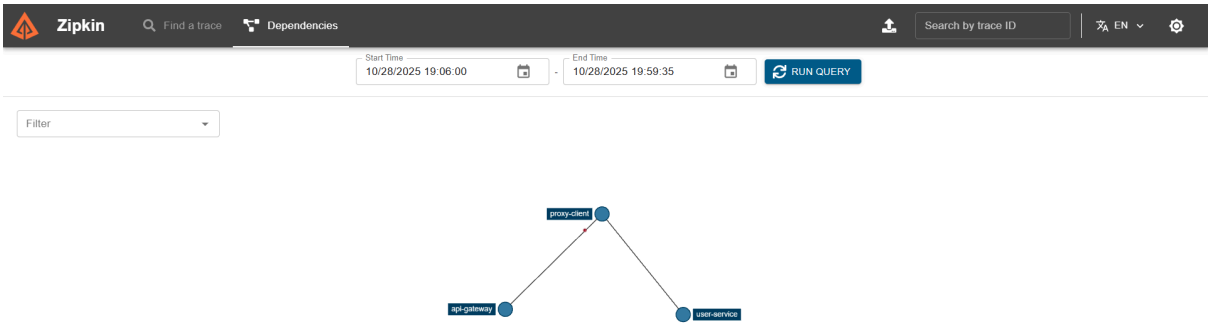
segundo error encontrado en user-service:
Resumen de los Problemas Resueltos
✅ 1. Error de Enrutamiento del API Gateway (RESUELTO)
Problema: API Gateway retornaba error 500 al intentar enrutar a user-service

```
AnnotatedConnectException: Connection refused: localhost/127.0.0.1:8700
```

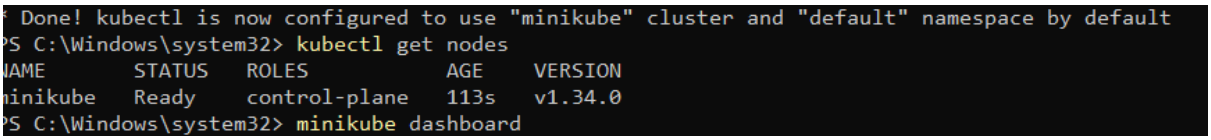
Solución: Agregamos EUREKA_INSTANCE_PREFER_IP_ADDRESS=true a todos los servicios en docker-compose-orchester.yml para que Eureka registre las IPs de los contenedores en lugar de localhost.

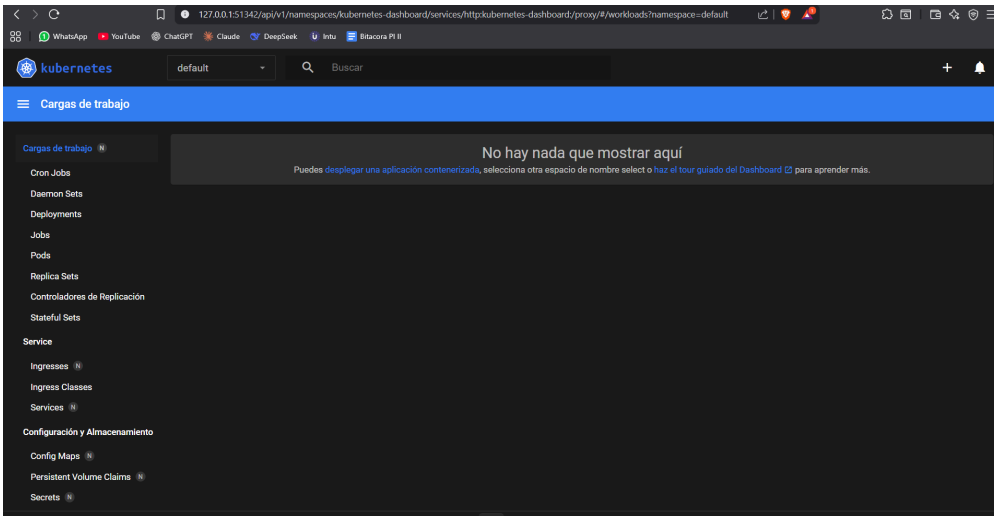
Resultado: ✅ API Gateway ahora enruta correctamente.
http://localhost:8080/user-service/api/users devuelve 200 OK

Otro problema encontrado en zipkin y solucionado:
API-Gateway (archivo: application.yml)
❌ Eliminamos la ruta directa /user-service/** que iba directo a USER-SERVICE
✅ Dejamos solo la ruta /app/** que va a PROXY-CLIENT
Resultado: Ahora TODO el tráfico pasa obligatoriamente por Proxy-Client
2. Proxy-Client (archivo: SecurityConfig.java)
✅ Agregamos .antMatchers("/api/users/**").permitAll() en Spring Security
Resultado: Ahora las peticiones a /api/users NO requieren autenticación (antes daba 403)
3. Reconstrucción y despliegue:
Recompilamos ambos servicios con mvn clean package
Reconstruimos las imágenes Docker
Reiniciamos los contenedores con docker-compose restart



Luego instalamos minikube e inicializamos un cluster:





Una vez instalado minikube debemos copiar las imagenes docker al hipervisor que usa minikube, en mi caso uso el Docker-desktop, cabe resaltar que minikube ya usa un docker propio, por eso toca volver a mandar las imágenes que se tenían en mi propio docker al del minikube.
para eso se hace:
`minikube docker-env` -> esto saca las variables de entorno.

`minikube docker-env --shell powershell | Invoke-Expression` -> y esto ya apunta al docker de minikube

y con esto comandos construimos las imágenes en minikube:

```
# Service Discovery
docker build -t service-discovery:v0.1.0 -f
service-discovery/Dockerfile .

# Cloud Config
docker build -t cloud-config:v0.1.0 -f cloud-config/Dockerfile .

# API Gateway
docker build -t api-gateway:v0.1.0 -f api-gateway/Dockerfile .

# Proxy Client
docker build -t proxy-client:v0.1.0 -f proxy-client/Dockerfile .

# User Service
docker build -t user-service:v0.1.0 -f user-service/Dockerfile .
```

Ejemplo de Cloud-config

```
PS C:\Users\daron\OneDrive\Escritorio\Semestre VIII\Ingesoft 5\ecommerce-microservice-backend-app> docker build -t cloud
-config:v0.1.0 -f cloud-config/Dockerfile .
[+] Building 2.9s (9/9) FINISHED                                                                    docker:default
 => [internal] load build definition from Dockerfile                                                         0.0s
 => => transferring dockerfile: 392B                                                                         0.0s
 => [internal] load metadata for docker.io/library/openjdk:11                                                0.9s
 => [internal] load .dockerignore                                                                            0.0s
 => => transferring context: 2B                                                                              0.0s
 => CACHED [1/4] FROM docker.io/library/openjdk:11@sha256:99bac5bf83633e3c7399aed725c8415e7b569b54e03e4599e580fc9  0.0s
 => [internal] load build context                                                                            1.2s
 => => transferring context: 59.78MB                                                                         1.1s
 => [2/4] RUN mkdir -p /home/app                                                                             0.7s
 => [3/4] WORKDIR /home/app                                                                                  0.1s
 => [4/4] ADD cloud-config/target/cloud-config-v0.1.0.jar cloud-config.jar                                   0.4s
 => exporting to image                                                                                       0.3s
 => => exporting layers                                                                                      0.2s
 => => writing image sha256:7cf6c61ea15fe4172fb20551c532e348dcac067d97c7b35969cd1d2048dd7c1f                 0.0s
 => => naming to docker.io/library/cloud-config:v0.1.0                                                       0.0s
PS C:\Users\daron\OneDrive\Escritorio\Semestre VIII\Ingesoft 5\ecommerce-microservice-backend-app>
```

api-getaway:

```
PS C:\Users\daron\OneDrive\Escritorio\Semestre VIII\Ingesoft 5\ecommerce-microservice-backend-app> docker build -t api-g
ateway:v0.1.0 -f api-gateway/Dockerfile .
[+] Building 2.5s (9/9) FINISHED                                                                          docker:default
 => [internal] load build definition from Dockerfile                                                               0.1s
 => => transferring dockerfile: 394B                                                                               0.0s
 => [internal] load metadata for docker.io/library/openjdk:11                                                      1.0s
 => [internal] load .dockerignore                                                                                  0.0s
 => => transferring context: 2B                                                                                    0.0s
 => [1/4] FROM docker.io/library/openjdk:11@sha256:99bac5bf83633e3c7399aed725c8415e7b569b54e03e4599e580fc9cdb7c21  0.0s
 => [internal] load build context                                                                                  0.7s
 => => transferring context: 58.71MB                                                                               0.7s
 => CACHED [2/4] RUN mkdir -p /home/app                                                                            0.0s
 => CACHED [3/4] WORKDIR /home/app                                                                                 0.0s
 => [4/4] ADD api-gateway/target/api-gateway-v0.1.0.jar api-gateway.jar                                            0.3s
 => exporting to image                                                                                             0.3s
 => => exporting layers                                                                                            0.2s
 => => writing image sha256:3d6b24f681f25b81855f6390b2ec7006a84f0c39950bb9ae7dacb4e9c027b9e6                       0.0s
 => => naming to docker.io/library/api-gateway:v0.1.0                                                              0.0s
PS C:\Users\daron\OneDrive\Escritorio\Semestre VIII\Ingesoft 5\ecommerce-microservice-backend-app>
```

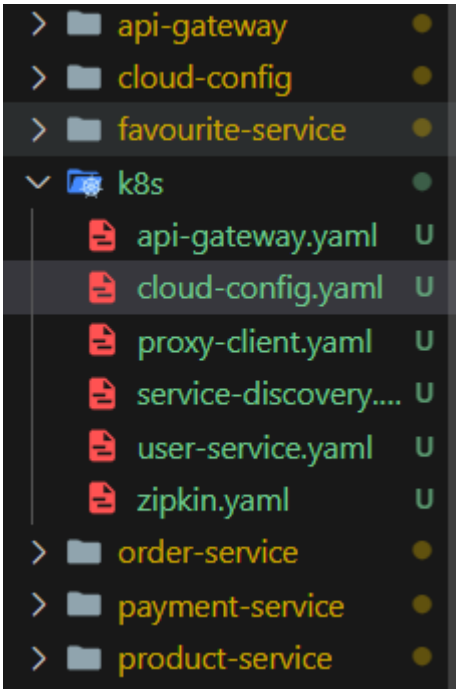
verificamos con `docker images | Select-String "service-discovery|cloud-config|api-gateway|proxy-client|user-service"`

```
PS C:\Users\daron\OneDrive\Escritorio\Semestre VIII\Ingesoft 5\ecommerce-microservice-backend-app> docker images | Select-String "serv
ice-discovery|cloud-config|api-gateway|proxy-client|user-service"

user-service                        v0.1.0    21411ddf492b   9 minutes ago    734MB
proxy-client                        v0.1.0    1fa742584cd9   10 minutes ago   715MB
api-gateway                         v0.1.0    3d6b24f681f2   11 minutes ago   713MB
cloud-config                        v0.1.0    7cf6c61ea15f   12 minutes ago   714MB
service-discovery                   v0.1.0    e0c7eb14b685   25 minutes ago   774MB


PS C:\Users\daron\OneDrive\Escritorio\Semestre VIII\Ingesoft 5\ecommerce-microservice-backend-app>
```

Creamos los yaml para el despliegue en kubernetes:



Desplegamos todos los servicios con:
kubectl apply -f k8s/

```
PS C:\Users\daron\OneDrive\Escritorio\Semestre VIII\Ingesoft 5\ecommerce-microservice-backend-app> kubectl apply -f k8s/
deployment.apps/api-gateway created
service/api-gateway created
deployment.apps/cloud-config created
service/cloud-config created
deployment.apps/proxy-client created
service/proxy-client created
deployment.apps/service-discovery created
service/service-discovery created
deployment.apps/user-service created
service/user-service created
deployment.apps/zipkin created
service/zipkin created
```

Algunos servicios fallaron así que tuve que corregir el .yaml y volver a desplegar, para ello los siguientes comandos:

```
kubectl delete deployment proxy-client

kubectl apply -f k8s/proxy-client.yaml
```

Prueba de los servicios desplegados:

```
PS C:\Users\daron\OneDrive\Escritorio\Semestre VIII\Ingesoft 5\ecommerce-microservice-backend-app> kubectl get pods
NAME                              READY   STATUS    RESTARTS      AGE
api-gateway-8b8c5b994-kxbsk       1/1     Running   0             105m
cloud-config-68b745f75f-dxpzw     1/1     Running   2 (97m ago)   105m
proxy-client-5d8b987dcb-rzsf5     1/1     Running   0             4m36s
service-discovery-b9bcf9674-v87d9 1/1     Running   2 (98m ago)   105m
user-service-7f78f45bfc-frqlz     1/1     Running   2 (96m ago)   105m
zipkin-bd67d9cf9-t77sn            1/1     Running   0             105m
PS C:\Users\daron\OneDrive\Escritorio\Semestre VIII\Ingesoft 5\ecommerce-microservice-backend-app>
```

Terminamos de desplegar el resto de microservicios:

```
PS C:\Users\daron\OneDrive\Escritorio\Semestre VIII\Ingesoft 5\ecommerce-microservice-backend-app> kubectl ge
t pods
NAME                                READY   STATUS    RESTARTS        AGE
api-gateway-8b8c5b994-kxbsk         1/1     Running   1 (5d16h ago)   5d18h
cloud-config-68b745f75f-dxpzw       1/1     Running   4 (30m ago)     5d18h
favourite-service-5885fb6668-z6g26  1/1     Running   0               4m32s
order-service-5956fbf86-h5hqz       1/1     Running   0               4m31s
payment-service-cd8d4787c-gbjcs     1/1     Running   0               24m
product-service-556ffbd5bc-l7jnl    1/1     Running   0               4m31s
proxy-client-5d8b987dcb-rzsf5       1/1     Running   1 (5d16h ago)   5d16h
service-discovery-b9bcf9674-v87d9   1/1     Running   3 (5d16h ago)   5d18h
shipping-service-5845d8695c-c5dg5   1/1     Running   0               4m30s
user-service-7f78f45bfc-frqlz       1/1     Running   3 (5d16h ago)   5d18h
zipkin-bd67d9cf9-t77sn              1/1     Running   1 (5d16h ago)   5d18h
PS C:\Users\daron\OneDrive\Escritorio\Semestre VIII\Ingesoft 5\ecommerce-microservice-backend-app>
```

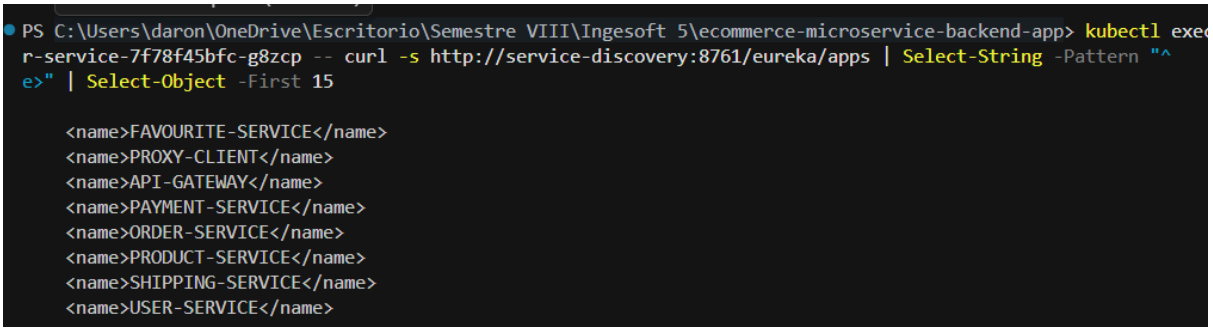
Comandos utilizados para construir las imágenes locales:
 # Servicios de infraestructura
docker build -f service-discovery/Dockerfile -t ecommerce/service-discovery:v0.1.0 .
docker build -f cloud-config/Dockerfile -t ecommerce/cloud-config:v0.1.0 .
docker build -f api-gateway/Dockerfile -t ecommerce/api-gateway:v0.1.0 .

 # Proxy y Gateway
docker build -f proxy-client/Dockerfile -t ecommerce/proxy-client:v0.1.0 .

 # Microservicios de negocio
docker build -f user-service/Dockerfile -t ecommerce/user-service:v0.1.0 .
docker build -f product-service/Dockerfile -t ecommerce/product-service:v0.1.0 .

docker build -f order-service/Dockerfile -t ecommerce/order-service:v0.1.0 .
docker build -f payment-service/Dockerfile -t ecommerce/payment-service:v0.1.0 .
docker build -f favourite-service/Dockerfile -t ecommerce/favourite-service:v0.1.0 .
docker build -f shipping-service/Dockerfile -t ecommerce/shipping-service:v0.1.0 .

Verificamos que todos los servicios estén registrados en eureka:



Iniciamos sesion: