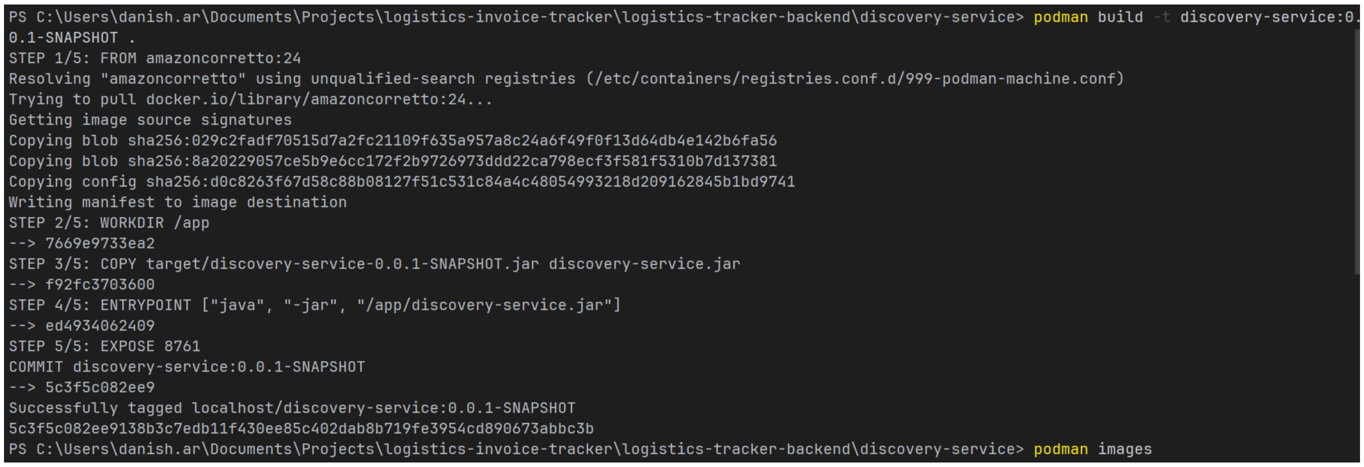
1. Create a Docker file

```
from amazoncorretto:24

WORKDIR /app

COPY target/discovery-service-0.0.1-SNAPSHOT.jar discovery-service.jar

ENTRYPOINT ["java", "-jar", "/app/discovery-service.jar"]

EXPOSE 8761
```

2. Build the container

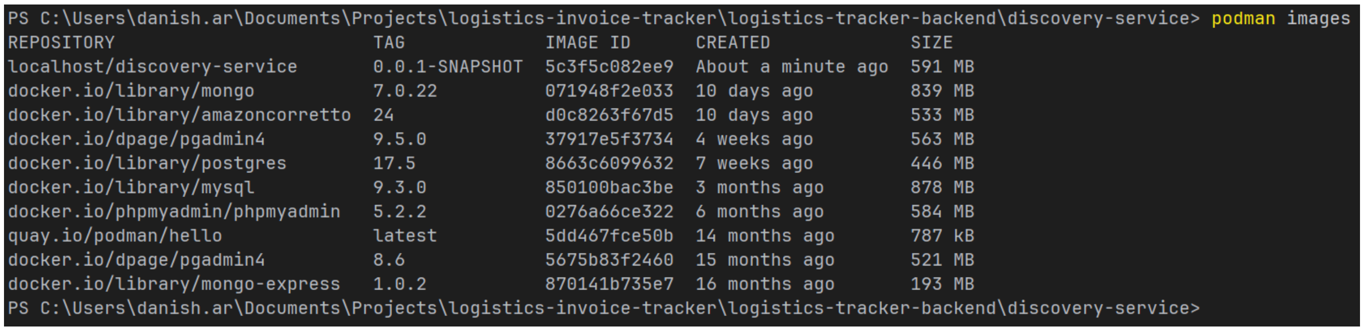
```
# podman build -t discovery-service:0.0.1-SNAPSHOT .
#OR
# podman build -t discovery-service:0.0.1-SNAPSHOT -f
C:\Users\danish.ar\Documents\Projects\logistics-invoice-tracker\logistics-
tracker-backend\discovery-service\Dockerfile
```



```
PS C:\Users\danish.ar\Documents\Projects\logistics-invoice-tracker\logistics-tracker-backend\discovery-service> podman build -t discovery-service:0.
0.1-SNAPSHOT .
STEP 1/5: FROM amazoncorretto:24
Resolving "amazoncorretto" using unqualified-search registries (/etc/containers/registries.conf.d/999-podman-machine.conf)
Trying to pull docker.io/library/amazoncorretto:24...
Getting image source signatures
Copying blob sha256:029c2fadf70515d7a2fc21109f635a957a8c24a6f49f0f13d64db4e142b6fa56
Copying blob sha256:8a20229057ce5b9e6cc172f2b9726973ddd22ca798ecf3f581f5310b7d137381
Copying config sha256:d0c8263f67d58c88b08127f51c531c84a4c48054993218d209162845b1bd9741
Writing manifest to image destination
STEP 2/5: WORKDIR /app
--> 7669e9733ea2
STEP 3/5: COPY target/discovery-service-0.0.1-SNAPSHOT.jar discovery-service.jar
--> f92fc3703600
STEP 4/5: ENTRYPOINT ["java", "-jar", "/app/discovery-service.jar"]
--> ed4934062409
STEP 5/5: EXPOSE 8761
COMMIT discovery-service:0.0.1-SNAPSHOT
--> 5c3f5c082ee9
Successfully tagged localhost/discovery-service:0.0.1-SNAPSHOT
5c3f5c082ee9138b3c7edb11f430ee85c402dab8b719fe3954cd890673abbc3b
PS C:\Users\danish.ar\Documents\Projects\logistics-invoice-tracker\logistics-tracker-backend\discovery-service> podman images
```

3. List the containers images

`podman images`



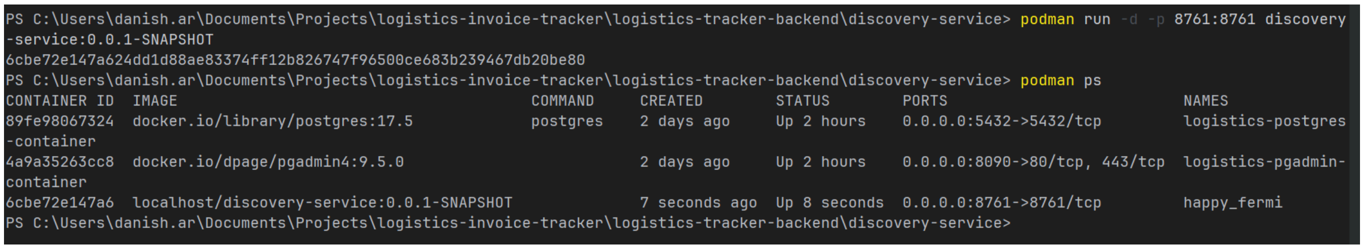
```
PS C:\Users\danish.ar\Documents\Projects\logistics-invoice-tracker\logistics-tracker-backend\discovery-service> podman images
REPOSITORY                        TAG             IMAGE ID      CREATED             SIZE
localhost/discovery-service       0.0.1-SNAPSHOT  5c3f5c082ee9  About a minute ago  591 MB
docker.io/library/mongo           7.0.22          071948f2e033  10 days ago         839 MB
docker.io/library/amazoncorretto  24              d0c8263f67d5  10 days ago         533 MB
docker.io/dpage/pgadmin4          9.5.0           37917e5f3734  4 weeks ago         563 MB
docker.io/library/postgres        17.5            8663c6099632  7 weeks ago         446 MB
docker.io/library/mysql           9.3.0           850100bac3be  3 months ago        878 MB
docker.io/phpmyadmin/phpmyadmin   5.2.2           0276a66ce322  6 months ago        584 MB
quay.io/podman/hello              latest          5dd467fce50b  14 months ago       787 kB
docker.io/dpage/pgadmin4          8.6             5675b83f2460  15 months ago       521 MB
docker.io/library/mongo-express   1.0.2           870141b735e7  16 months ago       193 MB
PS C:\Users\danish.ar\Documents\Projects\logistics-invoice-tracker\logistics-tracker-backend\discovery-service>
```

4. Run the container

Let's run in detached mode so it can run in background:

```
podman run -d -p 8761:8761 discovery-service:0.0.1-SNAPSHOT
```

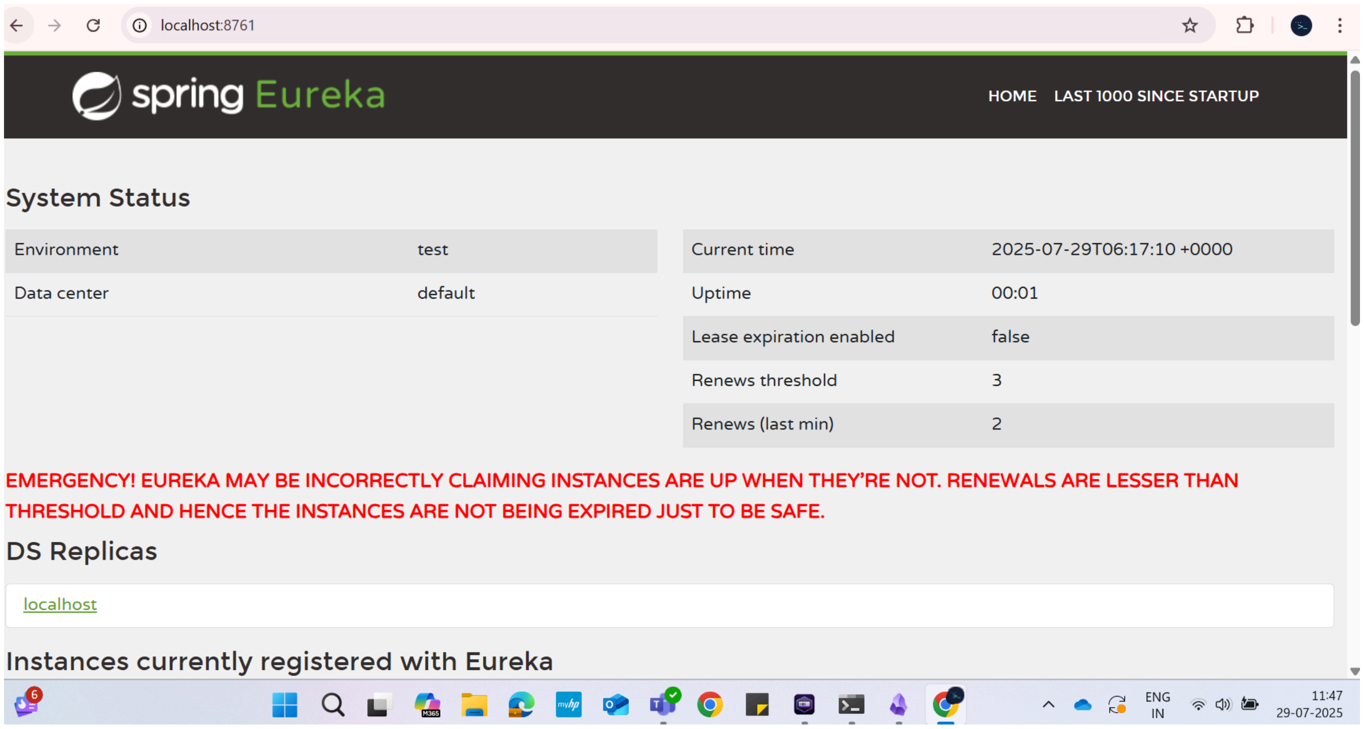
It is running:



```
PS C:\Users\danish.ar\Documents\Projects\logistics-invoice-tracker\logistics-tracker-backend\discovery-service> podman run -d -p 8761:8761 discovery
-service:0.0.1-SNAPSHOT
6cbe72e147a624dd1d88ae83374ff12b826747f96500ce683b239467db20be80
PS C:\Users\danish.ar\Documents\Projects\logistics-invoice-tracker\logistics-tracker-backend\discovery-service> podman ps
CONTAINER ID  IMAGE                                     COMMAND     CREATED        STATUS        PORTS                           NAMES
89fe98067324  docker.io/library/postgres:17.5           postgres    2 days ago     Up 2 hours    0.0.0.0:5432->5432/tcp          logistics-postgres
-container
4a9a35263cc8  docker.io/dpage/pgadmin4:9.5.0                        2 days ago     Up 2 hours    0.0.0.0:8090->80/tcp, 443/tcp  logistics-pgadmin-
container
6cbe72e147a6  localhost/discovery-service:0.0.1-SNAPSHOT            7 seconds ago  Up 8 seconds  0.0.0.0:8761->8761/tcp          happy_fermi
PS C:\Users\danish.ar\Documents\Projects\logistics-invoice-tracker\logistics-tracker-backend\discovery-service>
```

5. Verifying the running container

It's successfully running at port 8761



System Status

Environment	test	Current time	2025-07-29T06:17:10 +0000
Data center	default	Uptime	00:01
		Lease expiration enabled	false
		Renews threshold	3
		Renews (last min)	2

EMERGENCY! EUREKA MAY BE INCORRECTLY CLAIMING INSTANCES ARE UP WHEN THEY'RE NOT. RENEWALS ARE LESSER THAN THRESHOLD AND HENCE THE INSTANCES ARE NOT BEING EXPIRED JUST TO BE SAFE.

DS Replicas

localhost

Instances currently registered with Eureka

Let's run another service using similar approach:

Booking Service

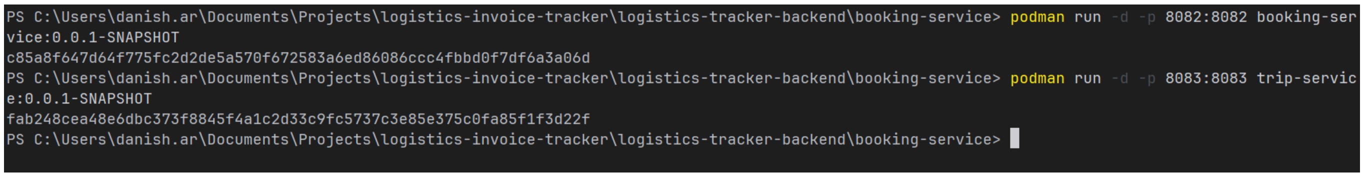
```
from amazoncorretto:24

WORKDIR /app

COPY target/booking-service-0.0.1-SNAPSHOT.jar booking-service.jar

ENTRYPOINT ["java", "-jar", "/app/booking-service.jar"]

EXPOSE 8082

# podman build -t booking-service:0.0.1-SNAPSHOT .
```

Trip Service

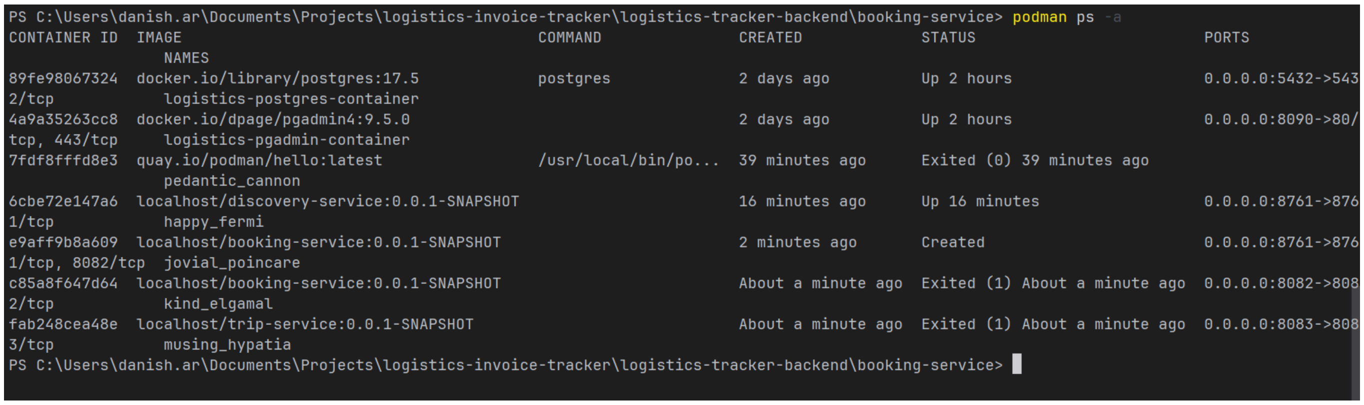
```
from amazoncorretto:24

WORKDIR /app

COPY target/trip-service-0.0.1-SNAPSHOT.jar trip-service.jar

ENTRYPOINT ["java", "-jar", "/app/trip-service.jar"]

EXPOSE 8083

# podman build -t trip-service:0.0.1-SNAPSHOT .
```

Run both these services:

```
podman run -d -p 8082:8082 booking-service:0.0.1-SNAPSHOT
podman run -d -p 8083:8083 trip-service:0.0.1-SNAPSHOT
```



```
PS C:\Users\danish.ar\Documents\Projects\logistics-invoice-tracker\logistics-tracker-backend\booking-service> podman run -d -p 8082:8082 booking-ser
vice:0.0.1-SNAPSHOT
c85a8f647d64f775fc2d2de5a570f672583a6ed86086ccc4fbbd0f7df6a3a06d
PS C:\Users\danish.ar\Documents\Projects\logistics-invoice-tracker\logistics-tracker-backend\booking-service> podman run -d -p 8083:8083 trip-servic
e:0.0.1-SNAPSHOT
fab248cea48e6dbc373f8845f4a1c2d33c9fc5737c3e85e375c0fa85f1f3d22f
PS C:\Users\danish.ar\Documents\Projects\logistics-invoice-tracker\logistics-tracker-backend\booking-service>
```

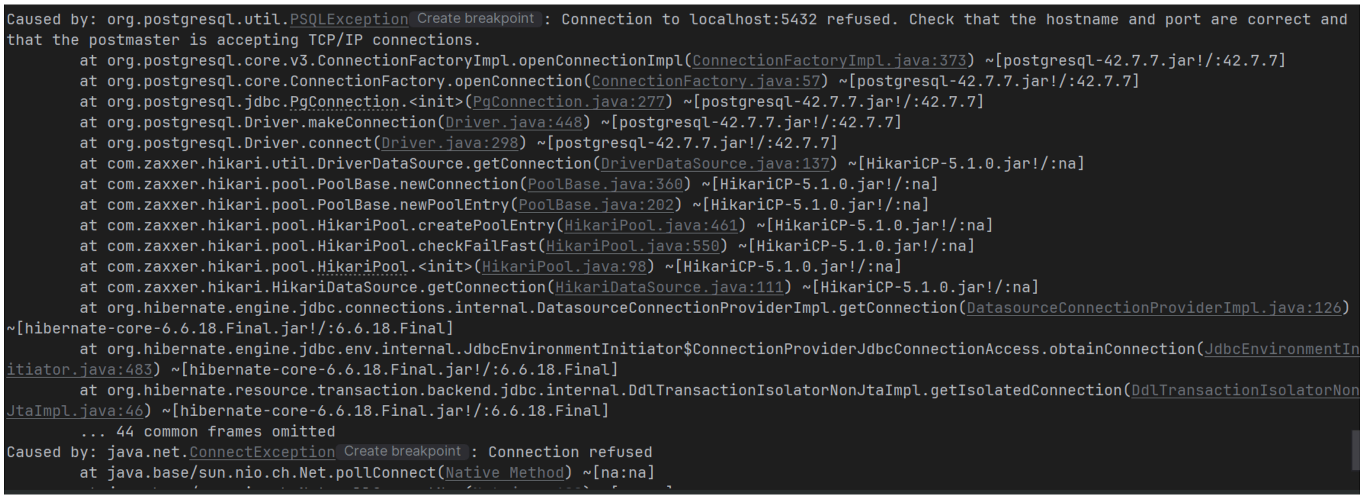


```
PS C:\Users\danish.ar\Documents\Projects\logistics-invoice-tracker\logistics-tracker-backend\booking-service> podman ps -a
CONTAINER ID  IMAGE                                      COMMAND               CREATED             STATUS                        PORTS
                 NAMES
89fe98067324  docker.io/library/postgres:17.5            postgres              2 days ago          Up 2 hours                    0.0.0.0:5432->543
2/tcp            logistics-postgres-container
4a9a35263cc8  docker.io/dpage/pgadmin4:9.5.0                                   2 days ago          Up 2 hours                    0.0.0.0:8090->80/
tcp, 443/tcp     logistics-pgadmin-container
7fdf8fffd8e3  quay.io/podman/hello:latest                /usr/local/bin/po...  39 minutes ago      Exited (0) 39 minutes ago
                 pedantic_cannon
6cbe72e147a6  localhost/discovery-service:0.0.1-SNAPSHOT                       16 minutes ago      Up 16 minutes                 0.0.0.0:8761->876
1/tcp            happy_fermi
e9aff9b8a609  localhost/booking-service:0.0.1-SNAPSHOT                         2 minutes ago       Created                       0.0.0.0:8761->876
1/tcp, 8082/tcp  jovial_poincare
c85a8f647d64  localhost/booking-service:0.0.1-SNAPSHOT                         About a minute ago  Exited (1) About a minute ago  0.0.0.0:8082->808
2/tcp            kind_elgamal
fab248cea48e  localhost/trip-service:0.0.1-SNAPSHOT                            About a minute ago  Exited (1) About a minute ago  0.0.0.0:8083->808
3/tcp            musing_hypatia
PS C:\Users\danish.ar\Documents\Projects\logistics-invoice-tracker\logistics-tracker-backend\booking-service>
```

Seems some issue, service getting exited quickly. We can check the logs:

`podman logs musing_hypatia`

musing_hypatia is the name for trip-service, you can see in above snapshot.



```
Caused by: org.postgresql.util.PSQLException Create breakpoint : Connection to localhost:5432 refused. Check that the hostname and port are correct and
that the postmaster is accepting TCP/IP connections.
	at org.postgresql.core.v3.ConnectionFactoryImpl.openConnectionImpl(ConnectionFactoryImpl.java:373) ~[postgresql-42.7.7.jar!/:42.7.7]
	at org.postgresql.core.ConnectionFactory.openConnection(ConnectionFactory.java:57) ~[postgresql-42.7.7.jar!/:42.7.7]
	at org.postgresql.jdbc.PgConnection.<init>(PgConnection.java:277) ~[postgresql-42.7.7.jar!/:42.7.7]
	at org.postgresql.Driver.makeConnection(Driver.java:448) ~[postgresql-42.7.7.jar!/:42.7.7]
	at org.postgresql.Driver.connect(Driver.java:298) ~[postgresql-42.7.7.jar!/:42.7.7]
	at com.zaxxer.hikari.util.DriverDataSource.getConnection(DriverDataSource.java:137) ~[HikariCP-5.1.0.jar!/:na]
	at com.zaxxer.hikari.pool.PoolBase.newConnection(PoolBase.java:360) ~[HikariCP-5.1.0.jar!/:na]
	at com.zaxxer.hikari.pool.PoolBase.newPoolEntry(PoolBase.java:202) ~[HikariCP-5.1.0.jar!/:na]
	at com.zaxxer.hikari.pool.HikariPool.createPoolEntry(HikariPool.java:461) ~[HikariCP-5.1.0.jar!/:na]
	at com.zaxxer.hikari.pool.HikariPool.checkFailFast(HikariPool.java:550) ~[HikariCP-5.1.0.jar!/:na]
	at com.zaxxer.hikari.pool.HikariPool.<init>(HikariPool.java:98) ~[HikariCP-5.1.0.jar!/:na]
	at com.zaxxer.hikari.HikariDataSource.getConnection(HikariDataSource.java:111) ~[HikariCP-5.1.0.jar!/:na]
	at org.hibernate.engine.jdbc.connections.internal.DatasourceConnectionProviderImpl.getConnection(DatasourceConnectionProviderImpl.java:126)
~[hibernate-core-6.6.18.Final.jar!/:6.6.18.Final]
	at org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectionAccess.obtainConnection(JdbcEnvironmentIn
itiator.java:483) ~[hibernate-core-6.6.18.Final.jar!/:6.6.18.Final]
	at org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl.getIsolatedConnection(DdlTransactionIsolatorNon
JtaImpl.java:46) ~[hibernate-core-6.6.18.Final.jar!/:6.6.18.Final]
	... 44 common frames omitted
Caused by: java.net.ConnectException Create breakpoint : Connection refused
	at java.base/sun.nio.ch.Net.pollConnect(Native Method) ~[na:na]
```

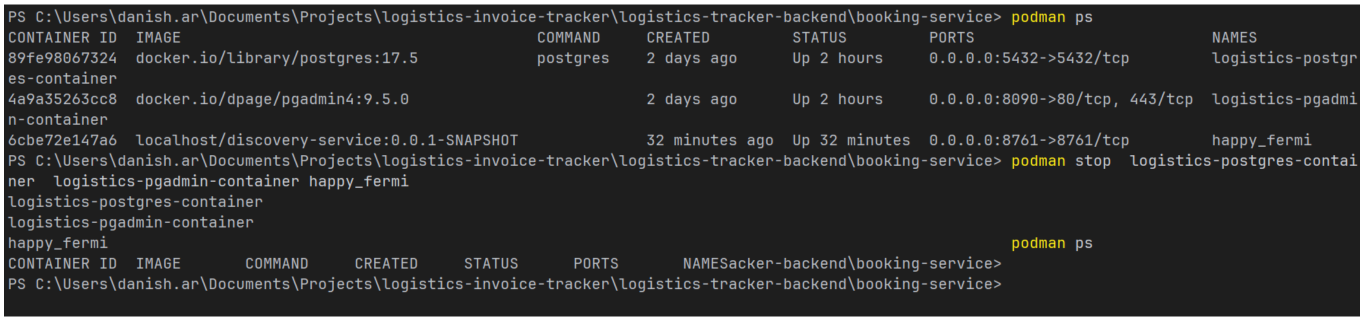
Issue is because of database, which is not running in the container.

We can run independent container like this but when container is dependent on another container we have two options.

1. Run all the container in the same pod
2. Run all the container in the same network

For local, generally it is recommended to run under the same pod so let's stop everything and run under same pod.

```
podman ps # list all running images
podman stop logistics-postgres-container logistics-pgadmin-container
happy_fermi
```



```
PS C:\Users\danish.ar\Documents\Projects\logistics-invoice-tracker\logistics-tracker-backend\booking-service> podman ps
CONTAINER ID  IMAGE                                    COMMAND     CREATED         STATUS          PORTS                          NAMES
89fe98067324  docker.io/library/postgres:17.5          postgres    2 days ago      Up 2 hours      0.0.0.0:5432->5432/tcp         logistics-postgr
es-container
4a9a35263cc8  docker.io/dpage/pgadmin4:9.5.0                       2 days ago      Up 2 hours      0.0.0.0:8090->80/tcp, 443/tcp  logistics-pgadmi
n-container
6cbe72e147a6  localhost/discovery-service:0.0.1-SNAPSHOT           32 minutes ago  Up 32 minutes   0.0.0.0:8761->8761/tcp         happy_fermi
PS C:\Users\danish.ar\Documents\Projects\logistics-invoice-tracker\logistics-tracker-backend\booking-service> podman stop  logistics-postgres-contai
ner  logistics-pgadmin-container happy_fermi
logistics-postgres-container
logistics-pgadmin-container
happy_fermi                                                                                                      podman ps
CONTAINER ID  IMAGE       COMMAND     CREATED     STATUS      PORTS       NAMESacker-backend\booking-service>
PS C:\Users\danish.ar\Documents\Projects\logistics-invoice-tracker\logistics-tracker-backend\booking-service>
```

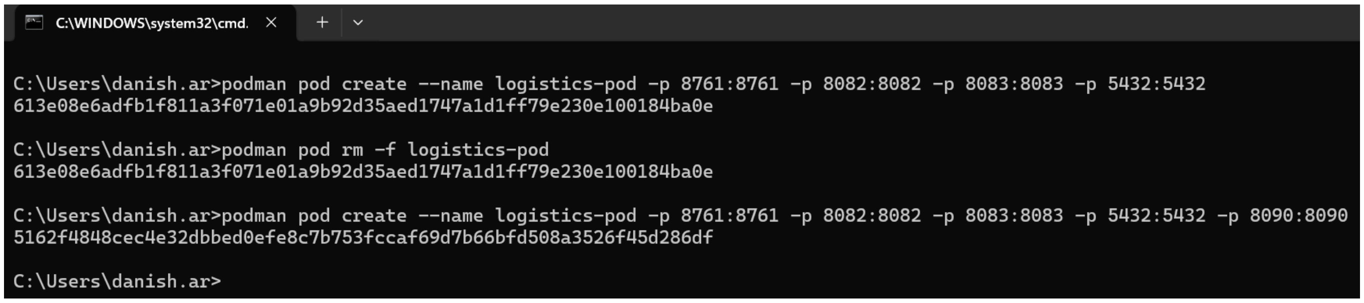
1. SAME POD

Create a pod and expose all the required IP:

```
podman pod create --name logistics-pod \
  -p 8761:8761 \  # Eureka discovery
  -p 8082:8082 \  # booking-service
  -p 8083:8083 \  # trip-service
  -p 5432:5432    # PostgreSQL

podman pod create --name logistics-pod -p 8761:8761 -p 8082:8082 -p 8083:8083
-p 5432:5432
```

Forgot to expose port 8090 for pgadmin4 so delete the pod and recreate it

```
podman pod rm -f logistics-pod

podman pod create --name logistics-pod -p 8761:8761 -p 8082:8082 -p 8083:8083
-p 5432:5432 -p 8090:8090
```



```
C:\Users\danish.ar>podman pod create --name logistics-pod -p 8761:8761 -p 8082:8082 -p 8083:8083 -p 5432:5432
613e08e6adfb1f811a3f071e01a9b92d35aed1747a1d1ff79e230e100184ba0e

C:\Users\danish.ar>podman pod rm -f logistics-pod
613e08e6adfb1f811a3f071e01a9b92d35aed1747a1d1ff79e230e100184ba0e

C:\Users\danish.ar>podman pod create --name logistics-pod -p 8761:8761 -p 8082:8082 -p 8083:8083 -p 5432:5432 -p 8090:8090
5162f4848cec4e32dbbed0efe8c7b753fcaf69d7b66bfd508a3526f45d286df

C:\Users\danish.ar>
```

Run services and database in this pod only:

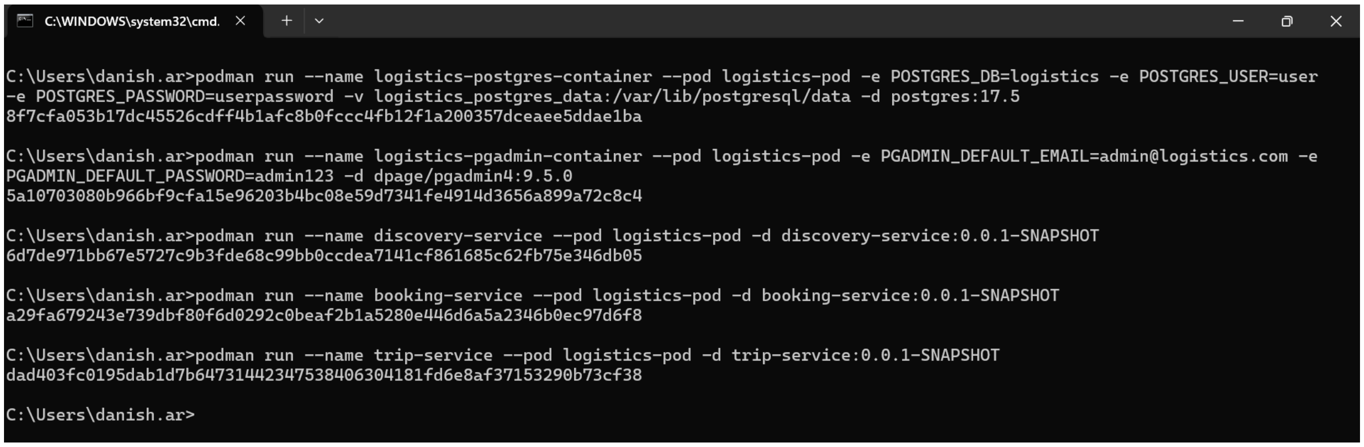
```
podman run --name logistics-postgres-container \
  --pod logistics-pod \
  -e POSTGRES_DB=logistics \
  -e POSTGRES_USER=user \
  -e POSTGRES_PASSWORD=userpassword \
  -v logistics_postgres_data:/var/lib/postgresql/data \
  -d postgres:17.5

podman run --name logistics-postgres-container --pod logistics-pod -e
POSTGRES_DB=logistics -e POSTGRES_USER=user -e POSTGRES_PASSWORD=userpassword
-v logistics_postgres_data:/var/lib/postgresql/data -d postgres:17.5
```

```
podman run --name logistics-pgadmin-container \
  --pod logistics-pod \
  -e PGADMIN_DEFAULT_EMAIL=admin@logistics.com \
  -e PGADMIN_DEFAULT_PASSWORD=admin123 \
  -d dpage/pgadmin4:9.5.0


podman run --name logistics-pgadmin-container --pod logistics-pod -e
PGADMIN_DEFAULT_EMAIL=admin@logistics.com -e PGADMIN_DEFAULT_PASSWORD=admin123
-d dpage/pgadmin4:9.5.0
```

```
podman run --name discovery-service \
  --pod logistics-pod \
  -d discovery-service:0.0.1-SNAPSHOT

podman run --name discovery-service --pod logistics-pod -d discovery-
service:0.0.1-SNAPSHOT
```

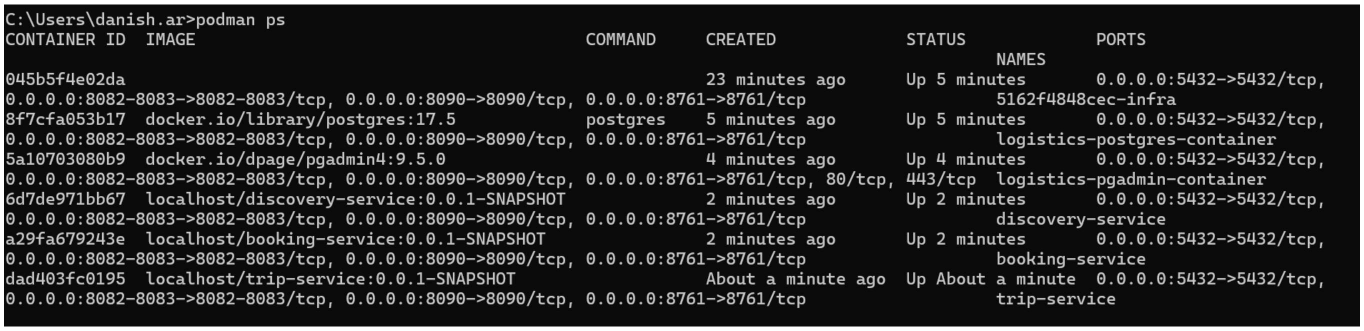
```
podman run --name booking-service \
  --pod logistics-pod \
  -d booking-service:0.0.1-SNAPSHOT


podman run --name booking-service --pod logistics-pod -d booking-
service:0.0.1-SNAPSHOT
```

```
podman run --name trip-service \
  --pod logistics-pod \
  -d trip-service:0.0.1-SNAPSHOT


podman run --name trip-service --pod logistics-pod -d trip-service:0.0.1-
SNAPSHOT
```



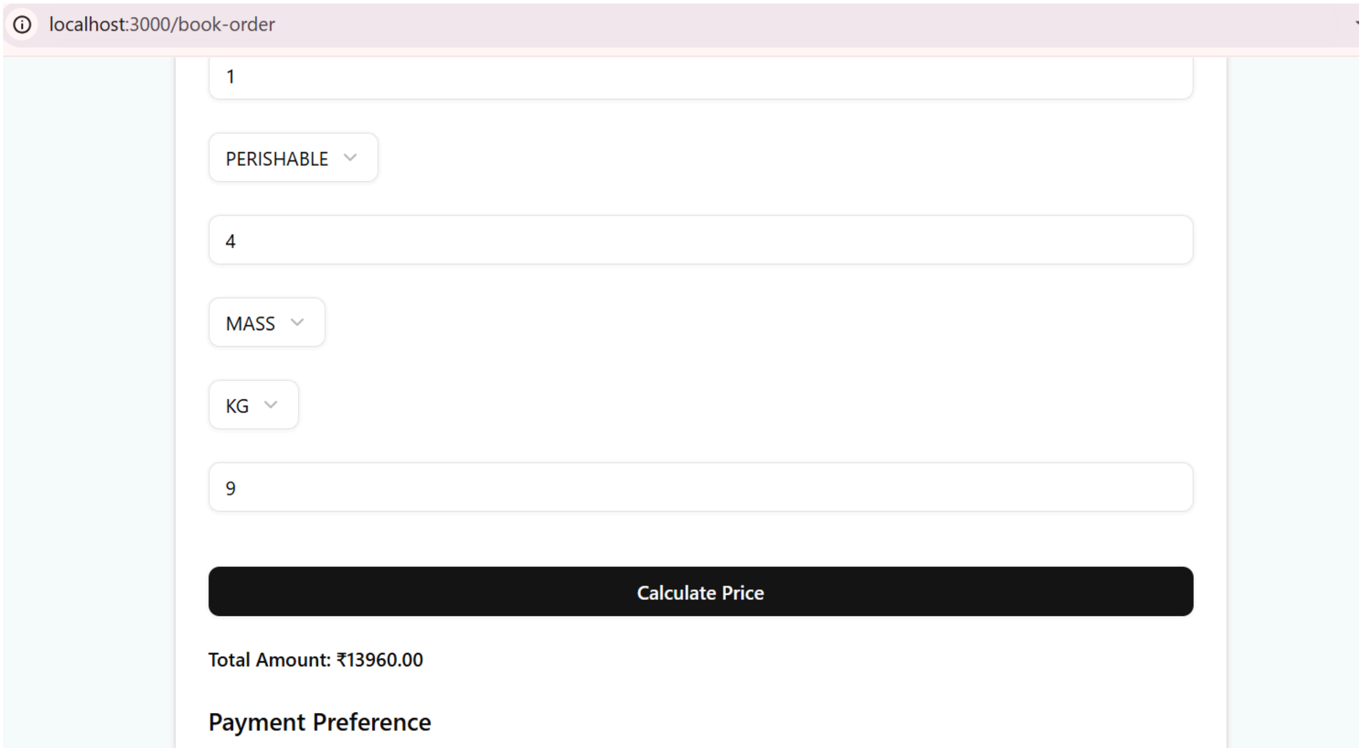
```
C:\Users\danish.ar>podman run --name logistics-postgres-container --pod logistics-pod -e POSTGRES_DB=logistics -e POSTGRES_USER=user
-e POSTGRES_PASSWORD=userpassword -v logistics_postgres_data:/var/lib/postgresql/data -d postgres:17.5
8f7cfa053b17dc45526cdff4b1afc8b0fccc4fb12f1a200357dceaee5ddae1ba

C:\Users\danish.ar>podman run --name logistics-pgadmin-container --pod logistics-pod -e PGADMIN_DEFAULT_EMAIL=admin@logistics.com -e
PGADMIN_DEFAULT_PASSWORD=admin123 -d dpage/pgadmin4:9.5.0
5a10703080b966bf9cfa15e96203b4bc08e59d7341fe4914d3656a899a72c8c4

C:\Users\danish.ar>podman run --name discovery-service --pod logistics-pod -d discovery-service:0.0.1-SNAPSHOT
6d7de971bb67e5727c9b3fde68c99bb0ccdea7141cf861685c62fb75e346db05

C:\Users\danish.ar>podman run --name booking-service --pod logistics-pod -d booking-service:0.0.1-SNAPSHOT
a29fa679243e739dbf80f6d0292c0beaf2b1a5280e446d6a5a2346b0ec97d6f8

C:\Users\danish.ar>podman run --name trip-service --pod logistics-pod -d trip-service:0.0.1-SNAPSHOT
dad403fc0195dab1d7b6473144234753840630418 1fd6e8af37153290b73cf38

C:\Users\danish.ar>
```

Let's verify if all container running:



```
C:\Users\danish.ar>podman ps
CONTAINER ID  IMAGE                                         COMMAND     CREATED             STATUS              PORTS
                                                                                                        NAMES
045b5f4e02da                                                            23 minutes ago      Up 5 minutes        0.0.0.0:5432->5432/tcp,
0.0.0.0:8082-8083->8082-8083/tcp, 0.0.0.0:8090->8090/tcp, 0.0.0.0:8761->8761/tcp                       5162f4848cec-infra
8f7cfa053b17  docker.io/library/postgres:17.5               postgres    5 minutes ago       Up 5 minutes        0.0.0.0:5432->5432/tcp,
0.0.0.0:8082-8083->8082-8083/tcp, 0.0.0.0:8090->8090/tcp, 0.0.0.0:8761->8761/tcp                       logistics-postgres-container
5a10703080b9  docker.io/dpage/pgadmin4:9.5.0                            4 minutes ago       Up 4 minutes        0.0.0.0:5432->5432/tcp,
0.0.0.0:8082-8083->8082-8083/tcp, 0.0.0.0:8090->8090/tcp, 0.0.0.0:8761->8761/tcp, 80/tcp, 443/tcp  logistics-pgadmin-container
6d7de971bb67  localhost/discovery-service:0.0.1-SNAPSHOT                2 minutes ago       Up 2 minutes        0.0.0.0:5432->5432/tcp,
0.0.0.0:8082-8083->8082-8083/tcp, 0.0.0.0:8090->8090/tcp, 0.0.0.0:8761->8761/tcp                       discovery-service
a29fa679243e  localhost/booking-service:0.0.1-SNAPSHOT                  2 minutes ago       Up 2 minutes        0.0.0.0:5432->5432/tcp,
0.0.0.0:8082-8083->8082-8083/tcp, 0.0.0.0:8090->8090/tcp, 0.0.0.0:8761->8761/tcp                       booking-service
dad403fc0195  localhost/trip-service:0.0.1-SNAPSHOT                     About a minute ago  Up About a minute   0.0.0.0:5432->5432/tcp,
0.0.0.0:8082-8083->8082-8083/tcp, 0.0.0.0:8090->8090/tcp, 0.0.0.0:8761->8761/tcp                       trip-service
```

Everything is running, let's reverify from the frontend which is accessing booking and trip services:



localhost:3000/book-order

1
PERISHABLE
4
MASS
KG
9

Total Amount: ₹13960.00

Payment Preference

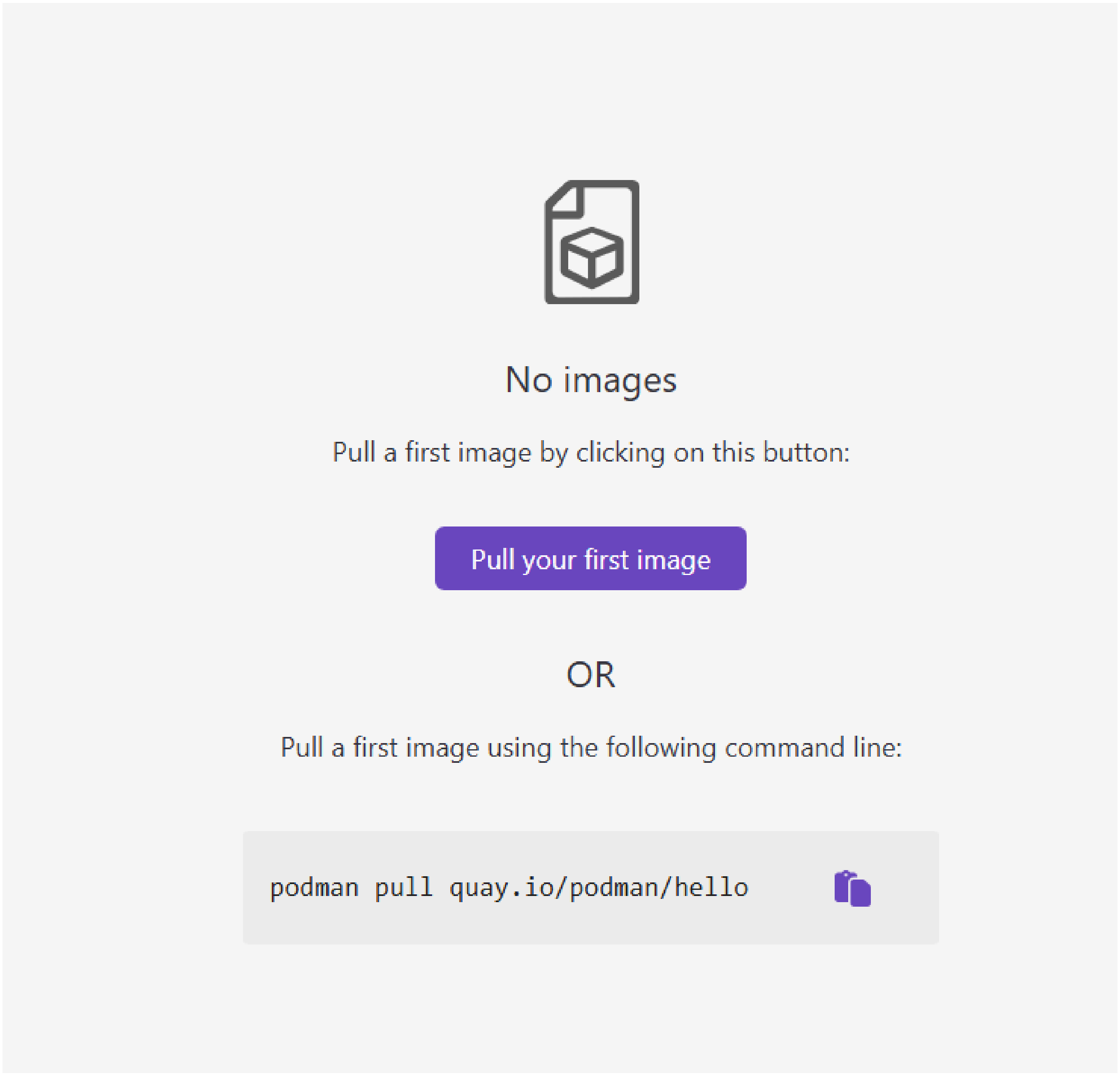
API is working fine, even we can verify from logs:

`podman logs booking-service`

---

1. SAME NETWORK

Let's delete the whole things and images and start from scratch.



No images

Pull a first image by clicking on this button:

OR

Pull a first image using the following command line:

```
podman pull quay.io/podman/hello
```

Create a network: `podman network create logistics-net`

Run databases

```
podman run --name pg-db \
  --network logistics-net \
  -p 5432:5432 # not required we can connect using pg-db
  -e POSTGRES_USER=user \
  -e POSTGRES_PASSWORD=userpassword \
  -e POSTGRES_DB=logistics \
  -v pg_data:/var/lib/postgresql/data \
  -d postgres:17.5

podman run --name pg-db --network logistics-net -p 5432:5432 -e
POSTGRES_USER=user -e POSTGRES_PASSWORD=userpassword -e POSTGRES_DB=logistics
-v pg_data:/var/lib/postgresql/data -d postgres:17.5
```

In microservices where using postgress connect using below properties

```
spring:
  application:
    name: booking-service
  datasource:
    url: jdbc:postgresql://pg-db:5432/logistics
    username: user
    password: userpassword
    driver-class-name: org.postgresql.Driver

  jpa:
    show-sql: true
    hibernate:
      ddl-auto: update
    database-platform: org.hibernate.dialect.PostgreSQLDialect


# update eureka client as well localhost won't work
eureka:
  client:
    service-url:
      defaultZone: http://discovery-service:8761/eureka
```

```
podman run --name discovery-service \
  --network logistics-net \
  -p 8761:8761 \
  -d discovery-service:0.0.1-SNAPSHOT

podman run --name discovery-service --network logistics-net -p 8761:8761 -d
discovery-service:0.0.1-SNAPSHOT
```

```
podman run --name booking-service \
  --network logistics-net \
  -p 8082:8082 \
  -d booking-service:0.0.1-SNAPSHOT


podman run --name booking-service --network logistics-net -p 8082:8082 -d
booking-service:0.0.1-SNAPSHOT
```

```
podman run --name trip-service \
  --network logistics-net \
  -p 8083:8083 \
  -d trip-service:0.0.1-SNAPSHOT


podman run --name trip-service --network logistics-net -d -p 8083:8083 trip-
service:0.0.1-SNAPSHOT
```

All the services running fine under the same network and connected to each other.

Run all using **logistics-compose.yml**

Build all services:

```
podman build -t discovery-service:0.0.1-SNAPSHOT ./discovery-service
podman build -t booking-service:0.0.1-SNAPSHOT ./booking-service
podman build -t trip-service:0.0.1-SNAPSHOT ./trip-service
```

All services yaml:

```
spring:
  datasource:
    url: jdbc:postgresql://pg-db:5432/logistics
    username: user
    password: userpassword
    driver-class-name: org.postgresql.Driver
  jpa:
    show-sql: true
    hibernate:
      ddl-auto: update
    database-platform: org.hibernate.dialect.PostgreSQLDialect
```

```yaml
eureka:
  client:
    service-url:
      defaultZone: http://discovery-service:8761/eureka
```

`logistics-compose.yml`

```yaml
version: "3.8"

services:
  postgres:
    image: postgres:17.5
    container_name: pg-db
    environment:
      POSTGRES_USER: user
      POSTGRES_PASSWORD: userpassword
      POSTGRES_DB: logistics
    volumes:
      - pg_data:/var/lib/postgresql/data
    networks:
      - logistics-net

  discovery-service:
    image: discovery-service:0.0.1-SNAPSHOT
    container_name: discovery-service
    ports:
      - "8761:8761"
    networks:
      - logistics-net

  booking-service:
    image: booking-service:0.0.1-SNAPSHOT
    container_name: booking-service
    ports:
      - "8082:8082"
    depends_on:
      - postgres
      - discovery-service
    networks:
      - logistics-net

  trip-service:
    image: trip-service:0.0.1-SNAPSHOT
    container_name: trip-service
    ports:
      - "8083:8083"
    depends_on:
      - postgres
      - discovery-service
    networks:
      - logistics-net

volumes:
  pg_data:

networks:
  logistics-net:
    # name: logistics-net # if network not created before
    external: true # if net already created before



# podman compose -f logistics-compose.yml up -d
# podman compose -f logistics-compose.yml down
```

Happy Coding

A R