INSTALL DOCKER

```
sudo amazon-linux-extras install docker -y

sudo service docker start

sudo docker info
```

CREATE A DOCKER SWARM AND JOIN

1 docker swarm init (manager DB)

2 docker swarm join --token SWMTKN-1-5o0a9cqj0rc7mbjo4gj8pdhchp9bbpt6hg40j55s8g9v4tuph5-7gw9rb83ohiepsr4qp3f54opa 172.31.0.32:2377 (BACKEND)

EXPECTED OUTPUT

```
docker swarm join --token SWMTKN-1-5o0a9cqj0rc7mbjo4gj8pdhchp9bbpt6hg40j55s8g9v4tuph5-7gw9rb83ohiepsr4qp3f54opa 172.31.0.32:2377

This node joined a swarm as a worker.
```

3 On host1 (DATABASEHOST), create an attachable overlay network called test-net:

$ docker network create --driver=overlay --attachable test-net **(DATABASEHOST)**

Expected Output:

```
[root@ip-172-31-0-32 ~]# docker network create --driver=overlay --attachable test-net
```

Expected output : nlnn53oog9jmm70urh7srxnvy

```
[root@ip-172-31-0-32 ~]# docker network ls

NETWORK ID     NAME            DRIVER    SCOPE

15256c750c42   bridge          bridge    local

59f1169b050e   docker_gwbridge   bridge    local

4cc2bd5bb09e   host            host      local

l83lffrc14fw   ingress         overlay   swarm

9e54e28406c3   none            null      local
```

```
nlnn53oog9jm   test-net        overlay   swarm
```

Note the returned NETWORK ID – Verify the same network id when you join it from the frontend/backend from host2.

4 JOIN THE OVERYLAY NETWORK FROM BACKEND

On host2, list the available networks -- notice that test-net does not yet exist:

```
[root@ip-172-31-4-144 ~]# docker network ls
NETWORK ID     NAME             DRIVER    SCOPE
4af201ef3ff0   bridge           bridge    local
fb5ea782c294   docker_gwbridge   bridge    local
c1dfc01c62ad   host             host      local
l83lffrc14fw   ingress          overlay   swarm
e48d8ce75164   none             null      local
```

PING AND CONFIRM THE CONNECTIVITY

ON BACKEND (HOST 2 BACKEND)

```
yum install git -y
yum install maven –y
```

CLONE ON BACKEND

```
  mkdir code
   cd code/
   git clone https://github.com/Ashlesh12342/ashlesh.git
```

CHANGE FILE 1

cd /root/code/ashlesh/springboot-backend/src/main/resources

change application.properties

REMOVE FIRST LINE

server.port=${port}

EXPECTED OUTPUT:

```
spring.datasource.url=jdbc:mysql://${host}:3306/employee_db

spring.datasource.username=${mysql_user}

spring.datasource.password=${mysql_password}

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.format-sql=true

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5InnoDBDialect

spring.jpa.hibernate.ddl-auto=create
```

CHANGE FILE 2

cd /root/code/ashlesh/springboot-backend/src/main/java/com/example/demo

change in file SpringbootBackendApplication.java

remove port

```
11          String port=System.getenv("PORT");
```

CHANGE FILE 3 /root/code/ashlesh/springboot-backend/src/main/java/com/example/demo/controller EmployeeController.java

cd /root/code/ashlesh/springboot-backend/src/main/java/com/example/demo/controller

EmployeeController.java

LINE 24

@CrossOrigin(origins = "*")

@CrossOrigin(origins = "http://localhost:4200")

@CrossOrigin(origins = "*")

NOW BUILD THE PACKAGE

cd /root/code/ashlesh/springboot-backend   (pom.xml)

mvn clean package –DskipTests

BUILD THE IMAGE

docker build -t backend:1.0 .

docker images

```
backend     1.0      5588751ad1ec   37 seconds ago   554MB
```

RUN THE CONTAINER

docker run -d -p 8080:8080 --name backend-app --network test-net -e MYSQL_USER=root -e MYSQL_PASSWORD=abc123 -e HOST=my_db backend:1.0

ON DATABASE CONTAINER (SWARM MANAGER)

docker images

docker pull mysql:5.7

docker images

docker run -d --name my_db --network test-net -e MYSQL_ROOT_PASSWORD=abc123 -e MYSQL_DATABASE=employee_db mysql:5.7

docker ps

ON BACKEND CONTAINER

```
[root@ip-172-31-4-144 springboot-backend]# docker run -d -p 8080:8080 --name backend-app --network test-net -e MYSQL_USER=root -e MYSQL_PASSWORD=abc123 -e HOST=my_db backend:1.0
7c8a3b81db37672b03afa09da829de8691974f84354106e626a2b8a296e5d92e
```

```
[root@ip-172-31-4-144 springboot-backend]# docker ps
CONTAINER ID   IMAGE       COMMAND            CREATED         STATUS         PORTS                    NAMES
bff915e10e6e   backend:1.0   "java -jar app.jar"   3 minutes ago   Up 2 minutes   0.0.0.0:8080->8080/tcp
backend-app
```

FILE 4 (FRONTEND currently its on backend only)

/root/code/ashlesh/angular app/app/src/app   employee.service.ts

cd /root/code/ashlesh/angular app/app/src/app

Replace

private baseURL="http://${HOST}:8080/api/v1/employees";

private baseURL="http://54.236.63.13:8080/api/v1/employees";

with the public ip address of backend container

FILE 5 (FRONTEND cuurently its on backend only)

/root/code/ashlesh/angular app/app

cd /root/code/ashlesh/angular app/app

Dockerfile

Remove

CMD ["npm","start"]

Add

```
7 FROM nginx:1.17.1-alpine
  8 COPY --from=builder /app/dist/app /usr/share/nginx/html
```

EXPECTED DOCKER FILE

```
FROM node:12-alpine as builder
RUN mkdir -p /app
WORKDIR /app
COPY . .
RUN npm install
RUN npm run build --prod
FROM nginx:1.17.1-alpine
COPY --from=builder /app/dist/app /usr/share/nginx/html
```

Now in the same directory that is /root/code/ashlesh/angular app/app

Build the image

```
docker build -t frontend .
```

expected output

```
Step 8/8 : COPY --from=builder /app/dist/app /usr/share/nginx/html
 ---> 282dca907107
Successfully built 282dca907107
Successfully tagged frontend:latest
```

RUN THE CONTAINER FROM THE BUILT IMAGE

```
docker run -d --name angular-app -p 80:80 frontend
```

Expected output

```
[root@ip-172-31-4-144 app]# docker run -d --name angular-app -p 80:80 frontend
0d48787f840669ecbbcfc07e070e888330eb29033e40f5b2fca0e46b3054d3b4
```

```
[root@ip-172-31-4-144 app]# docker ps
CONTAINER ID   IMAGE        COMMAND                CREATED          STATUS          PORTS                  NAMES
0d48787f8406   frontend     "nginx -g 'daemon of…"   34 seconds ago   Up 32 seconds   0.0.0.0:80->80/tcp       angular-app
bff915e10e6e   backend:1.0   "java -jar app.jar"      2 hours ago      Up 2 hours      0.0.0.0:8080->8080/tcp   backend-app
```

TO CHECK BACKEND

http://100.26.227.224:8080/api/v1/employees

TO CHECK FRONTEND

Public ip address