Install docker on all machines

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

DB_MASTER (HOST 1)

Initiate Docker Swarm

```
sudo docker swarm init
```

Join Docker swarm on Backend Host & Frontend Host

```
sudo docker swarm join --token
SWMTKN-1-1jvqvcunsyl2uvmq2p8sguca8y9zt330bd9m5ihs7riemn453d-8008823a7l2b0ue4smf
21u1a5 172.31.6.174:2377
```

DB_MASTER (HOST 1)
Check if the worker nodes are connected

```
[ec2-user@ip-172-31-6-174 ~]$ sudo docker node ls
```

Expected Output:

```
ID                          HOSTNAME                    STATUS    AVAILABILITY   MANAGER STATUS
ENGINE VERSION
vyp5p9tu8pnwa7kem8jijgpck     ip-172-31-6-135.ec2.internal   Ready     Active
20.10.4
9axj4jtmav1279p5zwz8h1ine *   ip-172-31-6-174.ec2.internal   Ready     Active         Leader
20.10.4
ox1ds4cqjp8vw5jv41x4sviiz     ip-172-31-9-97.ec2.internal    Ready     Active
20.10.4
```

DB_MASTER (HOST 1)
Create network to attach backend and database

```
sudo docker network create --driver=overlay --attachable mynet01
```

```
sudo docker run -dp 3306:3306 --name customerdb --network mynet01 -e
MYSQL_ROOT_PASSWORD=root -e MYSQL_DATABASE=employee_db mysql:5.7
```

BACKEND HOST (HOST2)
Clone the code

Install git

```
sudo yum install git
git clone https://github.com/laxapatiakshaylearning/ashlesh
```

Edit the below files on backend

EDIT FILE 1/3

```
vi src/main/java/com/example/demo/SpringbootBackendApplication.java
```

COMMENT THE PORT HOST USER PASSWORD AS SHOWN BELOW

```
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class SpringbootBackendApplication {

	public static void main(String[] args) {
		SpringApplication.run(SpringbootBackendApplication.class, args);
	//	String port=System.getenv("PORT");
	  //	String host=System.getenv("HOST");
	    //	String mysql_user=System.getenv("MYSQL_USER");
	      //String mysql_password=System.getenv("MYSQL_PASSWORD");
	}

}
```

EDIT FILE 2/3

```
vi src/main/java/com/example/demo/controller/EmployeeController.java

@CrossOrigin(origins = "*")
```

EDIT FILE 3/3

```
vi src/main/resources/application.properties
```

Remove PORT (line 1)

Create a Docker file on Below Path (BACKEND HOST 2)
Navigate to /home/ec2-user/ashlesh

```
vi  /home/ec2-user/ashlesh/Dockerfile
```

```
#stage1

FROM maven:3.6-alpine as builder

COPY springboot-backend /springboot-backend

WORKDIR /springboot-backend

RUN mvn clean install -DskipTests

#stage2

FROM openjdk:8-alpine

COPY --from=builder  /springboot-backend/target/app.jar /app.jar

EXPOSE 8080

ENTRYPOINT ["java","-jar","app.jar"]#stage1
```

BUILD THE IMAGE (BACKEND HOST2)

```
sudo docker build -t customerapp:v01 .
```

RUN THE CONTAINER FROM IMAGE

```
sudo docker run -d  --name customerapp --network mynet01 -e host=customerdb -e
mysql_user=root -e mysql_password=root customerapp:v01
```

CHECK IF THE CONTAINER LAUNCHED IS RUNNING

```
sudo docker ps -a
```

ON DATABASE HOST (HOST1)
CREATE NETWORK ON DOCKER MASTER FOR FRONTEND - BACKEND connection

```
sudo docker network create --driver=overlay --attachable mynet02
sudo docker node ls
```

ON BACKEND HOST (HOST2)

```
sudo docker network connect mynet02 customerapp
```

FRONTEND

```
sudo yum install git
git clone https://github.com/laxapatiakshaylearning/ashlesh
```

EDIT file 1

```
vim /home/ec2-user/ashlesh/angular app/app/default.conf
server {
    listen       80;
    server_name  localhost;

    #charset koi8-r;
    #access_log  /var/log/nginx/host.access.log  main;

    location / {
        root   /usr/share/nginx/html;
        index  index.html index.htm;
    }

    location /api {
       proxy_pass http://customerapp:8080/api/v1/employees;
    }

    #error_page  404              /404.html;

    # redirect server error pages to the static page /50x.html
    #
    error_page   500 502 503 504  /50x.html;
    location = /50x.html {
        root   /usr/share/nginx/html;
    }
}
```

EDIT FILE 2

```
vim /home/ec2-user/ashlesh/angular app/app/src/app/employee.service.ts
```

This is the change

```
  private baseURL="api";
```

```
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import {Employee} from './employee';
import { Observable } from 'rxjs';

@Injectable({
  providedIn: 'root'
})
export class EmployeeService {
  private baseURL="api";
  constructor(private httpClient:HttpClient) { }
  getEmployeesList():Observable<Employee[]>{
    return this.httpClient.get<Employee[]>(`${this.baseURL}`);
  }

  createEmployee(employee:Employee):Observable<Object>{
    return this.httpClient.post(`${this.baseURL}`,employee);
  }

  getEmployeeById(id:number):Observable<Employee>{
    return this.httpClient.get<Employee>(`${this.baseURL}/${id}`);
  }

  updateEmployee(id:number , employee:Employee):Observable<Object>{
    return this.httpClient.put(`${this.baseURL}/${id}`,employee);
  }

  deleteEmployee(id:number):Observable<Object>{
    return this.httpClient.delete(`${this.baseURL}/${id}`);
  }
}
```

EDIT FILE 3

```
vim /home/ec2-user/ashlesh/angular app/app/Dockerfile
```

```
FROM node:10-alpine as build-step
RUN mkdir -p /app1
WORKDIR /app1
COPY package.json /app1
```

```
RUN npm install
COPY . /app1
RUN npm run build --prod
# Stage 2
FROM nginx:1.17.1-alpine
COPY --from=build-step /app1/dist/app /usr/share/nginx/html
RUN rm -rf /etc/nginx/conf.d/default.conf
COPY default.conf /etc/nginx/conf.d/
```

BUILD THE IMAGE

```
sudo docker build -t customerweb:v01 .
```

RUN THE CONTAINER FROM IMAGE

```
sudo docker run -d -it -p 80:80/tcp --name customerweb --network mynet02 customerweb:v01
```

CHECK IF THE CONTAINER IS RUNNING

```
sudo docker ps -a
```

CHECK THE SERVICES

Check frontend

Public ip of the frontend container

Check if backend is not accessible from Internet

http://100.26.4.91:8080/api/v1/employees