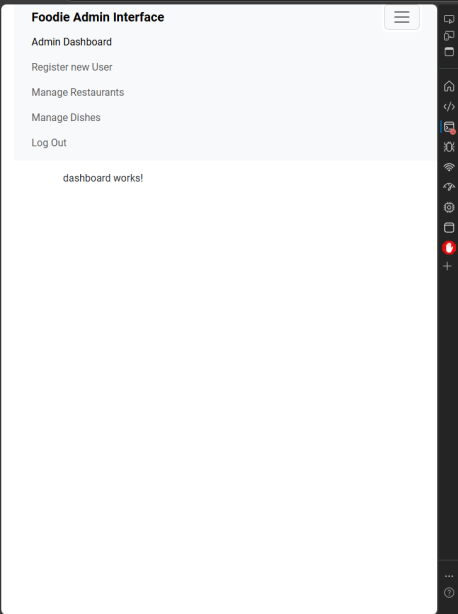
Capstone Project  
  
1) Angular Admin + Client Side  
Doing a login system that allows for different use roles allows for users and admins to go through the same system and receive different endpoints, including a different navbar and features.

When logging in as an admin, you will get more features such as the ability to register new users and add restaurants (and dishes)



Add Dish

A screenshot of a computer

Description automatically generated

Footer Shows Successful Creation

A screenshot of a computer

Description automatically generated

Successfully written into backend

A screenshot of a computer screen

Description automatically generated

Add Restaurant

A screenshot of a computer

Description automatically generated

Footer confirms successful creation

A screenshot of a computer

Description automatically generated

Appears in the mySQL backend

A screenshot of a computer

Description automatically generated

Generating auth guard for admin

A screen shot of a computer screen

Description automatically generated

Auth-guard.ts (Will add more authentication when backend is made)

Login as Admin:

A screen shot of a computer program

Description automatically generated

A computer screen shot of text

Description automatically generated

A screenshot of a computer

Description automatically generated

Admin View Dishes/Restaurants  
  
A screenshot of a computer

Description automatically generated

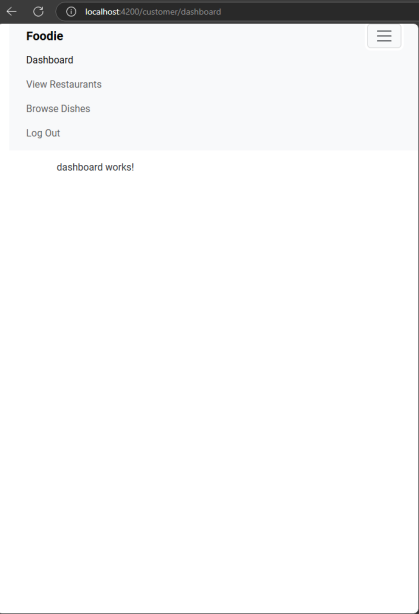
A screenshot of a phone

Description automatically generated

2) Angular User Side

User side has no management system or option to create additional users. This is accessed by logging in via an account that has UserRole = customer instead of admin.

Admin accounts cannot be created in registration.



View Dishes + Restaurants (As User, no Delete or Update)

A screenshot of a menu

Description automatically generated

A screenshot of a computer

Description automatically generated

3) Database SQL:  
  
CREATE DATABASE foodie;

USE foodie;

CREATE TABLE Restaurants (

id BIGINT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(255) NOT NULL,

location VARCHAR(255) NOT NULL

);

CREATE TABLE Dishes (

id BIGINT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(255) NOT NULL,

price FLOAT NOT NULL,

restaurant\_id BIGINT,

FOREIGN KEY (restaurant\_id) REFERENCES Restaurants(id) ON DELETE CASCADE

);  
  
Database Test Data  
  
INSERT INTO Restaurants (name, location) VALUES

('Greasy Grove', 'Manchester'),

('Pizza Place', 'Manchester'),

('Sushi Shop’, 'Leeds'),

('Tomato Town', 'Leeds’);  
  
INSERT INTO Dishes (name, price, restaurant\_id) VALUES

('Beef Burger', 2.99, 1),

('Cheeseburger', 3.99, 1),

('Pepperoni Pizza', 9.99, 2),

('Veggie Pizza', 8.99, 2),

(‘California Roll’, 4.99, 3),

('Edamame', 1.99, 3),

('Penne Pasta', 1.99, 4),

('Spaghetti Bolognaise', 3.99,4);

Test Queries

A screenshot of a menu

Description automatically generated

A screenshot of a menu

Description automatically generated

A screenshot of a computer

Description automatically generated

4) Spring Boot Java Backend

Initialise spring boot

A screenshot of a computer

Description automatically generated

A computer screen shot of a program

Description automatically generated

When ran, users generated in MYSQL

A screen shot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

Dishes + Restaurants created on springboot backend rather than SQL File:

A screen shot of a computer program

Description automatically generated

A screen shot of a computer

Description automatically generated

5) Front-End/Back-End Communication via HTTP

Creating data.service.ts

A screen shot of a computer

Description automatically generated

Change from localhost to server

A screen shot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

Java DataController

A screen shot of a computer

Description automatically generated

Jenkinsfile

Angular Jenkinsfile

A screenshot of a computer program

Description automatically generated

Springboot Jenkinsfile

A screenshot of a computer program

Description automatically generated

6) Dockerfile

Creating a network

A computer screen with white text

Description automatically generated

Running Containers

Dockerfile for Angular project

A screenshot of a computer

Description automatically generated

Dockerfile for Spring Boot project

A screenshot of a computer program

Description automatically generated

Docker Compose  
  
services:

# Foodie Admin (Angular)

foodie-admin:

build:

context: ./foodie-admin

hostname: foodie-admin

ports:

- "80:4200" # Expose Angular frontend on port 80

networks:

- foodie-network

# Foodie Spring Boot Application

foodiespringboot:

hostname: foodiespringboot

build:

context: ./foodiespringboot

ports:

- "8080:8080" # Expose backend API on port 8080

networks:

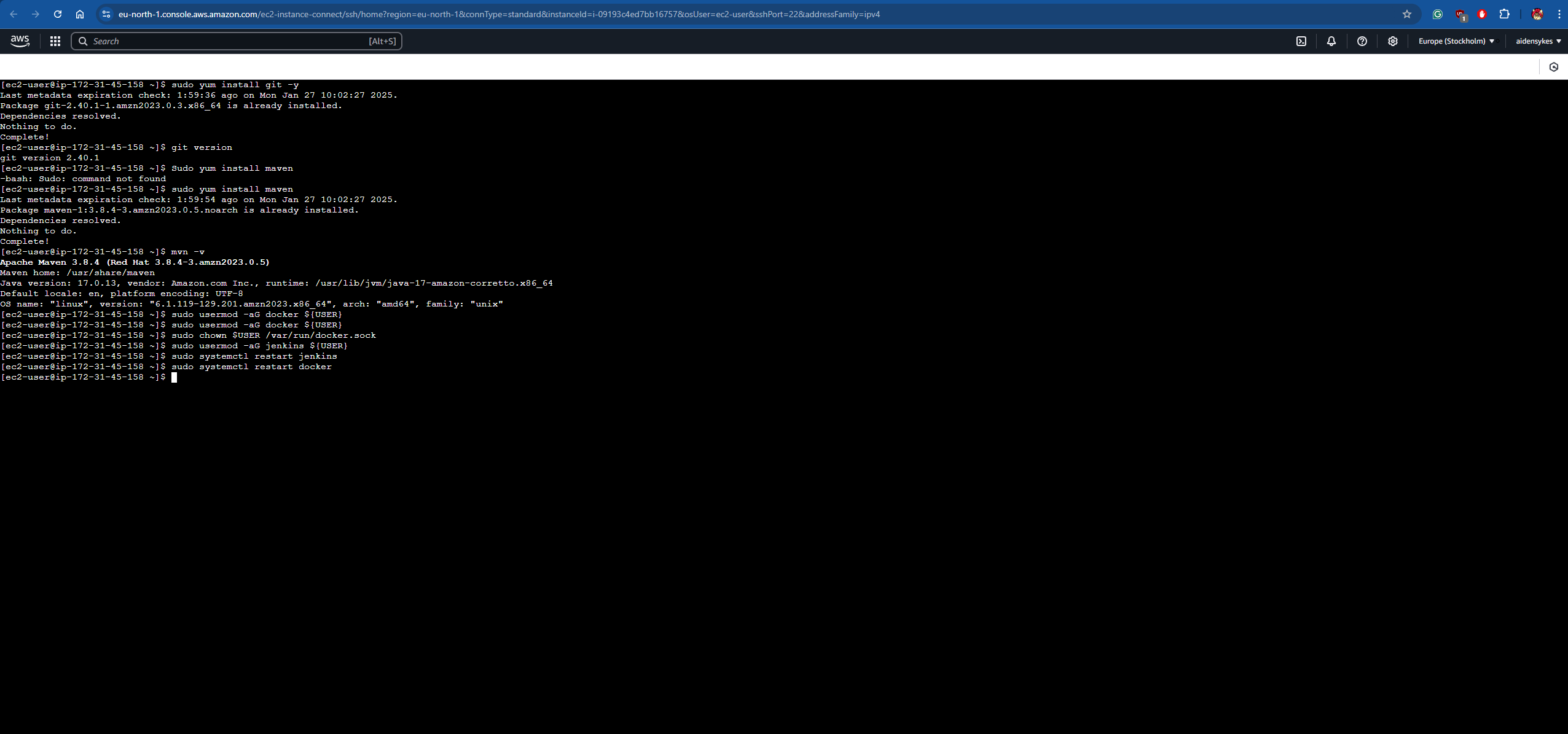
- foodie-network

networks:

foodie-network:

driver: bridge

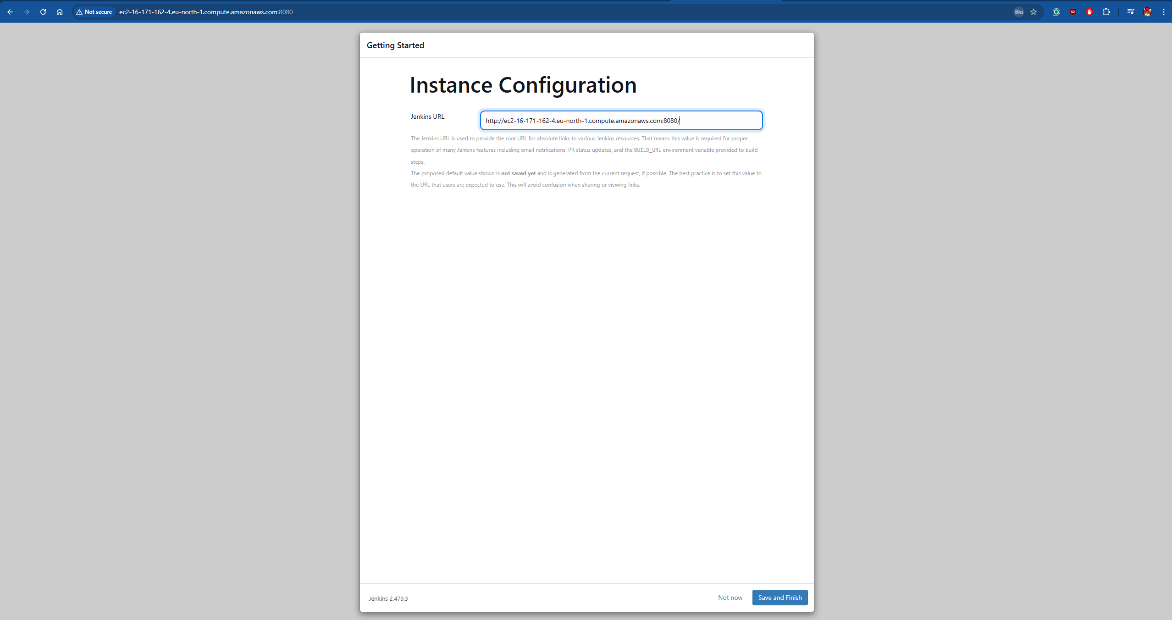
7) CI CD Pipeline

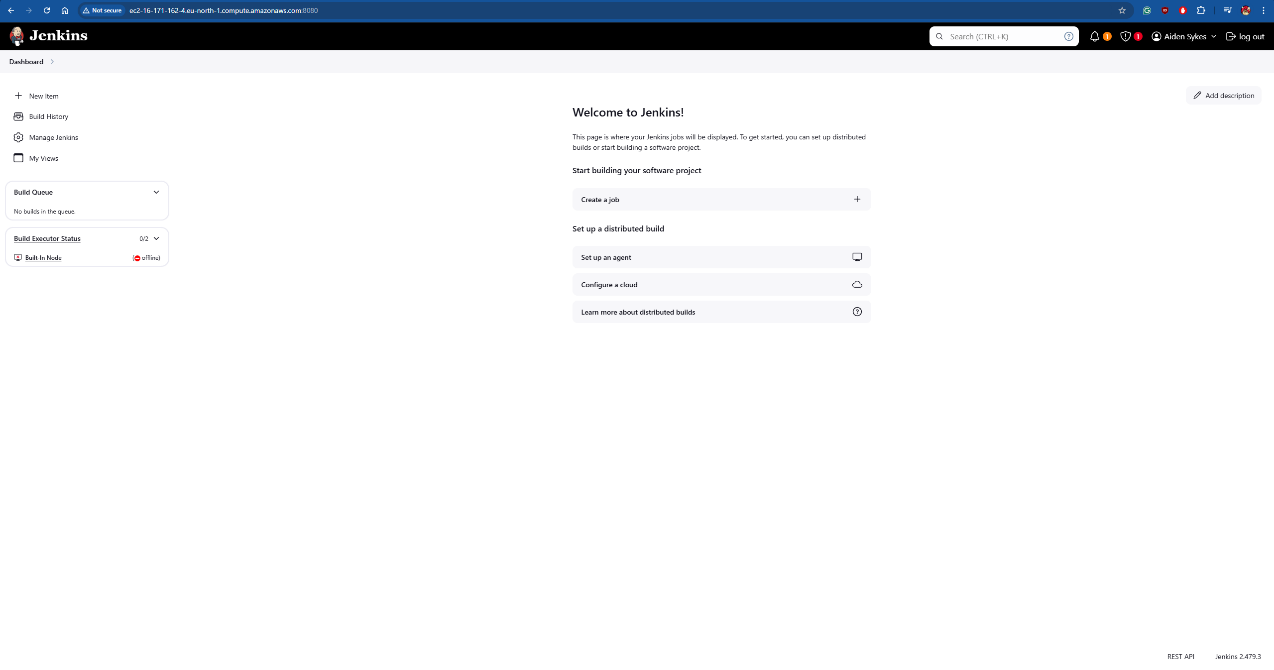


Jenkins + Docker InstalledA computer screen shot of a black background

Description automatically generated

A screenshot of a computer

Description automatically generated



Creating Pipelines

Pipeline Code:

A screenshot of a computer

Description automatically generated

pipeline {

agent any

stages {

stage('Checkout Code') {

steps {

git url: 'https://github.com/AidenSykes1999/CapstoneProject.git', branch: 'main'

}

}

stage('Build Angular App') {

steps {

dir('foodie-admin') {

sh 'npm install'

sh 'ng build --prod'

}

}

}

stage('Build Spring Boot App') {

steps {

dir('foodiespringboot') {

sh './mvnw clean package'

}

}

}

stage('Build Docker Images') {

steps {

dir('foodie-admin') {

sh 'docker build -t foodie-admin:latest .'

}

dir('foodiespringboot') {

sh 'docker build -t foodiespringboot:latest .'

}

}

}

stage('Deploy Docker Containers') {

steps {

sh 'docker rm -f foodie-admin-container || true'

sh 'docker run -d --name foodie-admin-container -p 80:80 foodie-admin:latest'

sh 'docker rm -f foodiespringboot-container || true'

sh 'docker run -d --name foodiespringboot-container -p 8080:8080 foodiespringboot:latest'

}

}

}

post {

success {

echo 'Both applications have been built and deployed successfully!'

}

failure {

echo 'One or more stages failed. Please review the logs for details.'

}

}

}

8) AWS EC2

EC2 Instance Page

A screenshot of a computer

Description automatically generated

Created Key Pair

A white background with black and blue text

Description automatically generated

A screen shot of a computer

Description automatically generated

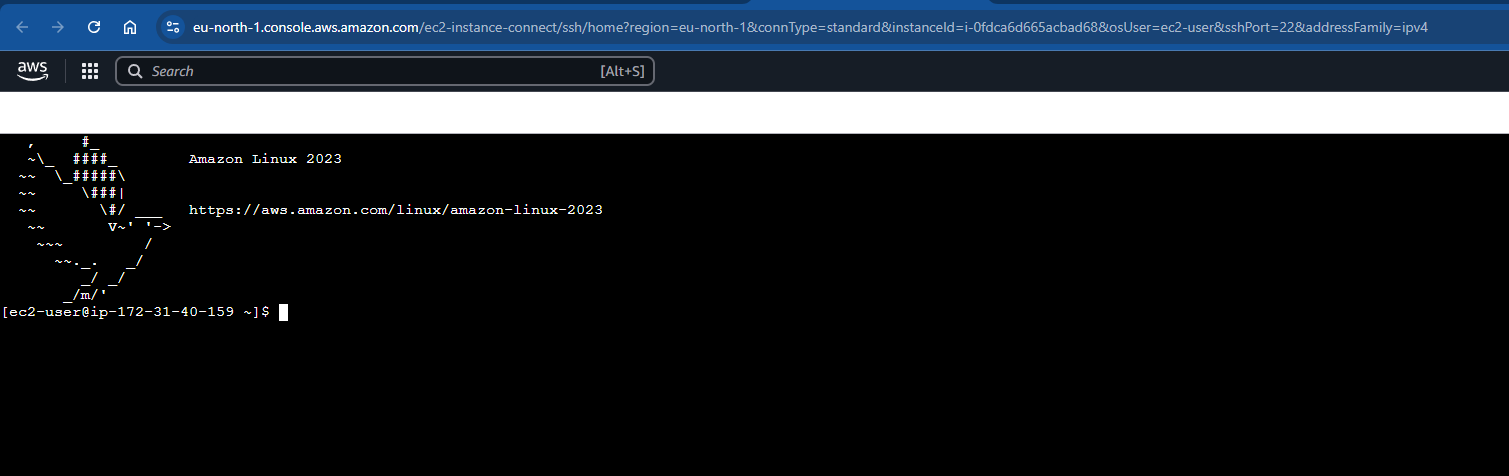
Created Security Group

A screenshot of a computer

Description automatically generated

Instance Created

A screenshot of a computer

Description automatically generated  
  


## Setting up docker and mysql on EC2

ssh -i "EC2KP.pem" ec2-user@ec2-16-171-15-158.eu-north-1.compute.amazonaws.com

Update the packages on your instance

[ec2-user ~]$ sudo yum update -y

A computer screen with a black screen

Description automatically generated

sudo yum install maven

A black screen with white text

Description automatically generated

mvn -v

A black screen with white text

Description automatically generated

sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo



sudo rpm --import <https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key>

Install Jdk

sudo dnf install java-17-amazon-corretto -y

A screenshot of a computer

Description automatically generated

Install Jenkins

sudo yum install jenkins -y

A screenshot of a computer

Description automatically generated

Install Docker

[ec2-user ~]$ sudo yum install docker -y

A screenshot of a computer program

Description automatically generated

sudo systemctl start Jenkins

sudo usermod -aG docker ${USER}

sudo yum install docker

sudo usermod -aG docker ${USER}

sudo chown $USER /var/run/docker.sock

docker images

sudo systemctl start docker

docker images

sudo docker images

sudo usermod -aG docker ${USER}

sudo chown $USER /var/run/docker.sock

Start the Docker Service

[ec2-user ~]$ sudo service docker start



Add the ec2-user to the docker group so you can execute Docker commands without using sudo.

[ec2-user ~]$ sudo usermod -a -G docker ec2-user

sudo service docker start

sudo docker pull mysql

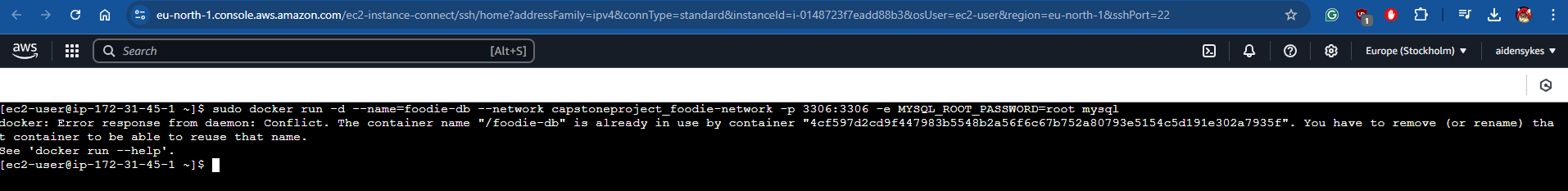
A screen shot of a computer

Description automatically generated

sudo docker network create capstoneproject\_foodie-network



sudo docker run -d --name=foodie-db --network capstoneproject\_foodie-network -p 3306:3306 -e MYSQL\_ROOT\_PASSWORD=root mysql



sudo docker exec -i foodie-db bash

mysql -u root -p

root

// Exec mysql scripts

exit

exit

## Pushing images onto docker hub, this should be automated with Jenkins CICD

docker login

// In foodie-admin

docker build -t AidenSykes1999/capstoneproject\_foodie-admin:latest .

// in foodiespringboot

docker build -t AidenSykes1999/capstoneproject\_foodiespringboot:latest .

docker push AidenSykes1999/capstoneproject\_foodie-admin

docker push AidenSykes1999/capstoneproject\_foodiespringboot

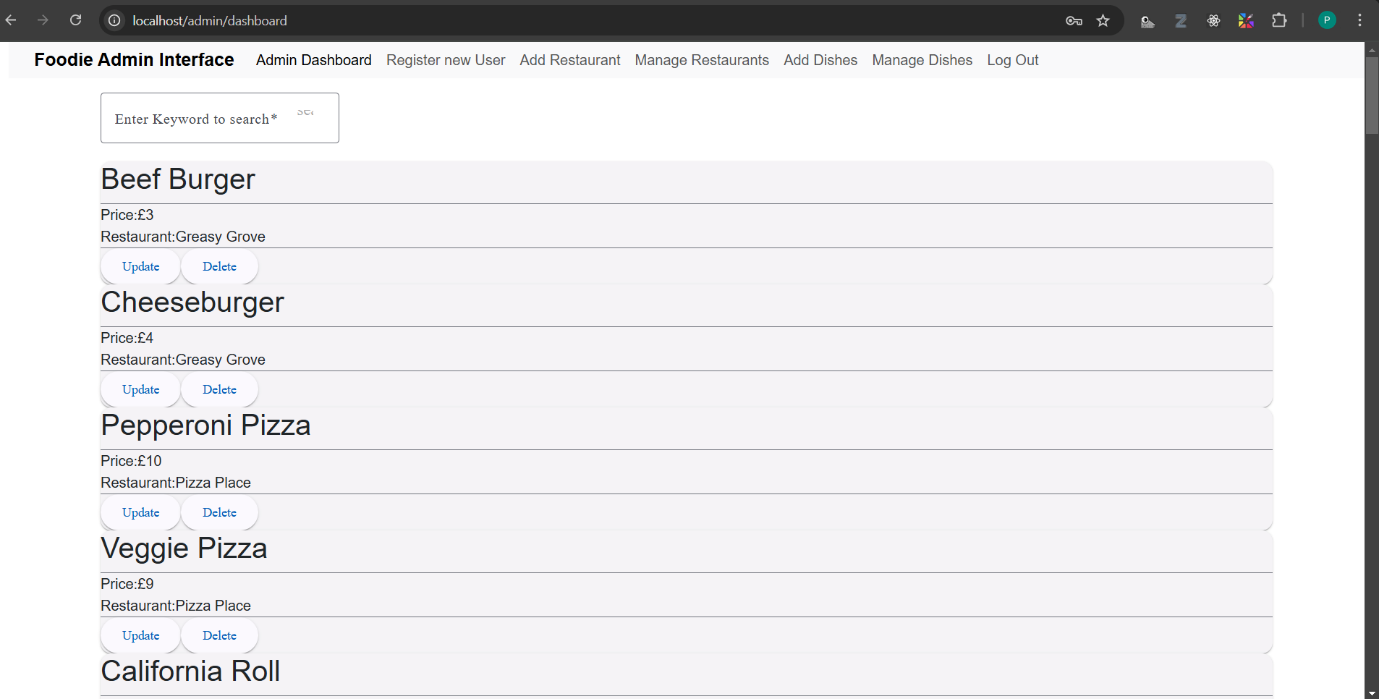
// On EC2 Instance SSH

docker pull AidenSykes1999/capstoneproject\_foodie-admin

docker pull AidenSykes1999/capstoneproject\_foodiespringboot

docker run -d --name=capstoneproject\_foodie-admin --network capstoneproject\_foodie-network -p 80:4200 AidenSykes1999/capstoneproject\_foodie-admin

docker run -d -p 8080:8080 --name=capstoneproject\_foodiespringboot --network=capstoneproject\_foodie-network AidenSykes1999/capstoneproject\_foodiespringboot



I brought this up with Shalini and Roma but because of permissions issues with Simplilearn Labs I couldn’t fully implement EC2, however it can work without issue using local Jenkins (no port numbers in use)

I also discussed with Shalini about how my EC2 is now charging me real money to try and attempt deployments. A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated