Capstone Project: Food Delivery System App  
Author: Rogelio Lotho

**Goal**: To Host the Application on AWS EC2 Instance, Automate the build and deployment using Jenkins on AWS EC2 and Containerize the application using Docker on AWS EC2

**Objective**: To build a dynamic and responsive food delivery app to display food items, filter based on user preferences, manage orders, user details.

**Tech Stack**: Angular, Java, SpringBoot, MySQL, Jenkins, Docker, AWS, Git and GitHub

1. Task 1. Angular Compoments, Routing, Services and AuthGuard, Forms

1.1 Components: Cart, Administrator, Cuisines, Customers, Home, Login, Navbar, Orders, Password, Payment, Register  
Graphical user interface, application

Description automatically generated

* 1. Routing  
     Graphical user interface, text

     Description automatically generated
  2. Services  
     1.3.1 dbService  
     A screenshot of a computer

     Description automatically generated with medium confidence
     1. Cart Service

A screenshot of a computer

Description automatically generated with medium confidence

1.3.3 Auth Service

A screenshot of a computer

Description automatically generated with medium confidence

* 1. AuthGuard  
     A screenshot of a computer

     Description automatically generated with medium confidence
  2. Forms  
     A screenshot of a computer

     Description automatically generated with medium confidence

1. Task 2 Angular Components, Routing, Services, and Auth Guard, Forms  
   2.1 App Routing Module  
   . Text

   Description automatically generated

2.2. Home Component  
Graphical user interface, website

Description automatically generated

2.3 Order Component Graphical user interface, website

Description automatically generated

2.4. Cart Component

Graphical user interface, application

Description automatically generated

2.5 Payment Component  
Graphical user interface, website

Description automatically generated

2.6. Login Component

Graphical user interface, text, application

Description automatically generated

2.7. Register CustomerGraphical user interface, application, website

Description automatically generated

2.8 Register Admin

Graphical user interface, application

Description automatically generated

2.9. Admin Cuisine Maintenance

Calendar

Description automatically generated with low confidence

2.10 Admin Update Cuisine

Graphical user interface, text, application

Description automatically generated

2.11. Admin Update Account Customer/Admin

Graphical user interface, application

Description automatically generated

* 1. AuthGuard  
     A screenshot of a computer

     Description automatically generated with medium confidence  
     A screenshot of a computer

     Description automatically generated

1. Task 3: SQL CRUD Commands, Primary and Foreign Key Relationship  
   Table

   Description automatically generated

Graphical user interface, text, application, Word

Description automatically generated

1. Task 4: SpringBoot Web Dependency, RestController, RequestMapping, Post and Get request
   1. SpringBoot Web Dependency  
      Graphical user interface

      Description automatically generated with medium confidence
   2. Rest Controller

Graphical user interface, text, application

Description automatically generated with medium confidence

* 1. Request Mapping

Graphical user interface, text, application

Description automatically generated

4.4. Post and Get RequestGraphical user interface, text, application

Description automatically generated  
4.5 Response model

Graphical user interface, text, application

Description automatically generated

1. Task 5: Angular HTTP Client Library, HTTP Request Response, JSON  
     
   5.1. Base URL Cross Origin  
   A screenshot of a computer

   Description automatically generated with medium confidence  
   5.1 HTTP Request Response  
   A screenshot of a computer

   Description automatically generated with medium confidence

5.2. JSON  
Text

Description automatically generated

1. Task 6: Jenkinsfile stages and step declarations

Administrator

* 1. Configure the project with Docker File  
     FROM openjdk:11  
     RUN mkdir /app

COPY target/ /app  
WORKDIR /app

CMD java -jar petclinic-0.0.1-SNAPSHOT.jar --spring.config.name=application.properties

* 1. Configure the project with JenkinsFile   
      pipeline {

agent any

stages {

stage('Build') {  
 steps {  
 //Get code from GitHub repository  
 git 'https://github.com/lothoroger/petclinic.git

//Run maven wrapper  
bat "mvn compile"

echo 'Building the Project with Maven compiler' }  
 }

stage('Test') {  
 steps {  
 bat 'mvn test'

echo 'Testing the PetClinic project with Maven test' }  
 }

stage('Deploy') {  
 steps {  
 bat 'mvn package'

echo 'Deploy the project with Maven package' }

}

} }

1. Build the project using the maven package  
   2.a mvn clean/package

Text

Description automatically generated  
Text

Description automatically generated  
2.b. run java -jar petclinic-0.0.1-SNAPSHOT.jar  
Text

Description automatically generated  
  
Graphical user interface, text, application, email

Description automatically generated

1. Create and Launch AWS EC2 Instance  
   3.1 Create Key Pair   
   Graphical user interface, text, application, email

   Description automatically generated

3.b Security Group  
Graphical user interface, text, application

Description automatically generated  
  
3.c EC2 Instance – PetClinic\_Instance  
Graphical user interface, text, application, email

Description automatically generated

1. Configure EC2 Instance with JDK 11  
   Install jdk 8 and Install jdk11  
   a. Create an Identity andAccessManagement (IAM)
2. Configure EC2 Instance with Docker  
   5.1 Install the docker on EC2
3. Configure EC2 Instance with Jenkins  
   6.1 Install the Jenkins on EC2  
   6.2 Create the Admin User
4. Upload the given code to git repo
5. Create Jenkins Pipeline on EC2 with SCM as git
6. Build the Pipeline to dockerize the application