**HOW TO DEPLOY HOTEL APPLICATION TO AWS**

PART 1: DEPLOY A HOTEL RESERVATION APP TO DOCKERHUB

1. Make the application read from the AWS environment instead of port 8080.

-To change where the application is read from, first go to the angular file app.component.ts.

-At the top with the rest of the imports, Import {Location,LocationStrategy} from “@angular/common”;

-Add the location and locationStrategy to the constructor as such:

* Constructor(private httpClient:HttpClient, private location: Location, private locationStrategy: LocationStrategy){}

-Comment out the line: private baseURL:string=’http://local:8080’;

-Underneath this commented line, add: private baseURL:string=this.location.path()

1. Generate a docker image using your Docker file.

-Add a docker file to the application in IntelliJ.

-Build a .jar file.

-In Maven, double click on ‘package’, and it will package it in a .jar. When it’s finished running, check the ‘target’ package to make sure it is there.

-In the terminal on IntelliJ, navigate to the directory where the Docker file is located

-If necessary, right click the Docker file in IntelliJ and click ‘Copy Path/Reference to identify the directory it is in. Use the ‘cd’ command to ensure you are in the right directory within the terminal.

-Use the command ‘docker info’ to ensure that Docker is running. This also shows you which images are there.

-To build the docker, enter ‘docker build –tag nameofimage .’ (choose your own name)

-When it is finished, go to Docker Desktop and locate your image in the ‘images’ tab.

1. Push your image to DockerHub.

-To do this, you need to set up a DockerHub account.

-You will need an access token, by going to ‘security’ and generating one.

-You want to make sure that your image is private so that no one else can use it.

-Go back to the terminal in IntelliJ and enter these commands to push the image:

docker tag nameofimage nameofrepository/nameofimage

docker push nameofrepository/nameofimage

-Go to your repository on DockerHub and refresh. Your image should be there.

-To make it private, go to the name of your image, then go to settings, then change visibility settings.

2. PART 2 – DEPLOY THE DOCKER IMAGE TO AWS

1. Log int AWS and set up an IAMs account.

-Sign into AWS account

-Find IAM in the AWS Services and go to IAM dashboard.

-Create an EC2 role

-Use AmazonEC2RoleforSSM policy for permissions.

-Call the role SSMDemo and create the role.

1. Launch an EC2 linux instance.

-Go to ‘instances’ in the left column of the screen.

-Name your instance and choose Linux (or whatever OS you prefer). Choose the free tier, and RSA for launching.

-Create a security group and make sure that you allow https and http traffic because this is a website.

-Go to ‘Advanced Details’ and select your IAM profile.

-Launch the instance.

1. Start a session and install docker.

* In AWS, in the console, search ‘Systems Manager’. From ‘System Manager’, to to ‘Session Manager’ and start a session. Use the instance you just launched.
* Next, you need to install docker by using these commands:
  + sudo sh
  + yum install -y docker
  + service docker start
  + docker info

1. Pull your instance from Docker hub by first logging into docker hub, pulling the instance into the session. After ensuring it is there, you are going to run the instance.

* To accomplish these steps, use these commands within the instance:
  + LOG IN: docker login (use personal access token as password)
  + PULL IMAGE FROM DOCKER: docker pull nameofrepository/nameofinstance
  + CHECK THERE ARE NO IMAGES: docker images
  + RUN DOCKER: docker run -name awsapplication -d -p 80:8080 nameofrepository/nameofinstance
* Go to the port with the session, and the application should now be running there. The application is now running on AWS and it can be read anywhere.