DEPLOYING SPRINGBOOT APPLICATION USING KUBERNETES:

Prerequisites:

1. AWS cli
2. Docker
3. Kubernetes
4. Eksctl
5. EC2 instance of type t2.medium with security group ports 80, 8080, 3306 enabled.
6. Mariadb server

Steps:

- Create an ec2 instance.
- Create **AdministrativeAccess** role and attach it to EC2 instance.
- Enable ports **80, 8080 and 3306** in instance security group.
- Clone repository using git clone.
- Install all the prerequisites either manually or using the script provided in the repository.
- Create secret access key in **IAM>security and credentials**. Configure aws-cli using command: **aws configure**
- Create eks cluster using yaml file provided in repository. (Before that make sure that public key(id_rsa.pub) is available in .ssh directory. If not create it using ssh keygen command: - **ssh-keygen -t rsa -b 4096 -f ~/.ssh/id_rsa**)
  Command to create cluster: **eksctl create cluster -f cluster.yaml**
- Once cluster is ready check nodes using command: **kubectl get node**
- Create MariaDB RDS instance in AWS console with name **springbackend** , keep id as **admin** and password as **admin123.** Keep the RDS public and do not connect it with ec2 instance.
- Once the RDS is created copy RDS endpoint and paste in **application.properties** file at this location of repository **angular-java/spring-backend/src/main/resources/application.properties**.
  E.g.: jdbc:mysql://**rds-endpoint**:3306/springbackend?useSSL=false
- Now login, Create and import DataBase in MariaDB using command:
  sudo mysql -h rds-endpoint -u admin -p
  CREATE DATABASE springbackend;
  sudo mysql -h rds-endpoint -u admin -p springbackend < springbackend.sql (this springbackend.sql file is available in the repository)
- Create repository for both frontend and backend in ECR registry or in docker-hub. Then build, tag and push the backend docker image to the registry or to the docker-hub. Copy the image URI and paste in into the backend manifest file of deployment.
- Create deployment and service for backend using manifest files which is in the location of repository **angular-java/spring-backend/manifest/**
  Command: **kubectl apply -f service.yaml**
  kubectl apply -f deployment.yaml

Check if service and deployment is properly running
Command: **kubectl get deploy**
kubectl get svc

- Create only service for frontend using service.yaml file which is in **angular-java/angular-frontend/manifest/**
  Command: **kubectl apply -f service.yaml**

- Then setup ingress-controller for that refer below links

  I. create an OIDC provider, use below link and follow steps:
  https://docs.aws.amazon.com/eks/latest/userguide/enable-iam-roles-for-service-accounts.html

  II. Install AWS Load Balancer Controller with manifests
  https://docs.aws.amazon.com/eks/latest/userguide/lbc-manifest.html

  III. Check that ingress controller is up and running or not if it's running it should display the status **1/1**

- Now create ingress file to create ALB and to access the service out of cluster using **ingress.yaml** file. Change backend and frontend service names (if any changes are made in its default name).
  Command: **kubectl apply -f ingress.yaml**
  Check if Load Balancer is created in aws console, if not check proper role of administrator access is assigned ec2 instance. Try to solve it using documentations or ChatGPT.

- Copy LoadBalancer DNS and paste it in **worker.service.ts** file at **angular-java/angular-frontend/src/app/services/worker.service.ts** location.
  E.g: private getUrl: string = "http://**LoadBalancer-DNS**/api/v1/workers";

- Now build, tag and push the frontend docker file to the ECR registry or to the docker-hub.

- Now copy paste the frontend docker image URI and paste it in frontend deployment.yaml file. Create deployment for frontend using **deployment.yaml** file which is at **angular-java/angular-frontend/manifest/** this location.
  Command: **kubectl apply -f deployment.yaml**

- Once all the steps have been completed successfully copy ALB DNS paste into the search bar and check if frontend is visible or not. If not try to solve it using documentations or ChatGPT.