**Kubernetes Project**

**Team**

1. Kollu Sravya Sree
2. Chiguru Keerthi Priya
3. Chintala Sai Akshitha

**Deploying Crud Application using Angular, Spring boot, MySQL into Kubernetes**

**Abstract**

Deploying a CRUD application that uses Angular for the frontend, Spring Boot for the backend, and MySQL as the database into Kubernetes involves several key steps to ensure scalability and manageability. The Angular application is containerized using Docker and deployed as a Kubernetes Deployment, enabling it to handle varying user loads effectively. Similarly, the Spring Boot backend is containerized and managed as a Kubernetes Deployment to handle business logic and API interactions. MySQL is deployed as a Stateful Set with persistent storage to maintain data integrity and durability. This deployment strategy leverages Kubernetes’ orchestration capabilities to achieve high availability, resilience, and simplified management, making it an ideal solution for modern web applications.

**Prerequisites**

* **Docker:** Ensure Docker is installed and running.
* **Kubernetes (Minikube):** Ensure Minikube is installed and running.
* **Kompose:** For converting Docker Compose to Kubernetes manifests.
* **Node.js and npm:** For Angular and Node dependencies.

**Project Structure**

1. Frontend: Angular Application

* Dockerfile: Used to build and serve the Angular application.
* Dependencies: Includes @angular/cdk and @angular/material which had dependency issues during installation.

1. Backend: Spring Boot Application

* Dockerfile: Used to build and run the Spring Boot application with Maven.
* Dependencies: Spring Boot application connects to MySQL and exposes APIs.

1. Database: MySQL

* Docker Compose Configuration: Defines the MySQL service including environment variables for setup.

1. Kubernetes Deployment:

* PersistentVolumeClaim (PVC): Configured for MySQL data storage.

**Implementation**

**Step -1:** Developed the crud application using angular, spring boot, MySQL.

**Step -2:** Create Docker files for the angular, Spring boot.

**Step -3:** Create Docker compose yaml file to integrate frontend, backend and database.

**Step -4:** Run the individual application

For angular - ng serve

For Spring Boot - mvn clean package

* + java -jar target/HelloWorld-0.0.1-SNAPSHOT.jar

**Step -6:** Rundocker-compose up –build to build the image of application

**Step -7:** Conversion to Kubernetes Manifests Using Kompose

* kompose convert

**Step -8:** Start the minikube and open minikube dashboard

**Step -9:** Deployments and Services yaml are created now apply them

* kubectl apply -f deployment.yaml
* kubectl apply -f service.yaml

**Step -10:** After the deployment is done try to check whether pods are created or not.

* kubectl get deployments
* kubectl get pods

**Step -11:** Access the Application

* kubectl get service angular-service