Prahalath P J K – 22CSR146 III CSE C

Day 3 – Minikube installation and mysql

Kubernetes

Kubernetes (K8s) is an open-source container orchestration platform that automates the deployment, scaling, and management of containerized applications. It helps in efficiently managing multiple containers across a cluster of machines, ensuring high availability, load balancing, and self-healing capabilities. Kubernetes is widely used for cloud-native applications and microservices architectures.

Minikube

Minikube is a lightweight Kubernetes implementation that runs a single-node Kubernetes cluster on a local machine. It is primarily used for development and testing purposes, allowing developers to experiment with Kubernetes features without needing a full-scale cluster. Minikube supports various container runtimes and can be installed on Windows, macOS, and Linux

curl -LO https://github.com/kubernetes/minikube/releases/latest/download/minikube-linux-amd64

sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm minikube-linux-amd64

minikube start

minikube start

minikube status

YML file

ersion: '3'

```
services:
  web:
    image: nginx:latest
    ports:
      - 80:80

  db:
    image: mysql:latest
    environment:
      - MYSQL_ROOT_PASSWORD=secret
```

```
docker exec -it prahalath-db-1 /bin/bash
mysql -u root -p
```

Docker compose:

Docker Compose

Docker Compose is a tool that allows you to define and manage multi-container Docker applications using a YAML configuration file (docker-compose.yml). It simplifies the process of running multiple interdependent services (such as a web server, database, and caching system) with a single command.

Key Features:

- **Multi-Container Management** – Define multiple services in one file.
- **Service Dependencies** – Automatically starts services in the correct order.
- **Networking** – Easily creates a shared network for containers.
- **Scalability** – Scale services up or down with a single command.

Example docker-compose.yml:

yaml

Copy code

```
version: '3'
services:
  web:
    image: nginx
    ports:
      - "8080:80"
  db:
    image: mysql
    environment:
      MYSQL_ROOT_PASSWORD: example
```

Usage:

sh

Copy code

```
# Start all services
docker compose up -d

# Stop and remove containers
docker compose down
```

Docker compose commands:

```
# Start and run containers in the background
docker compose up -d
```

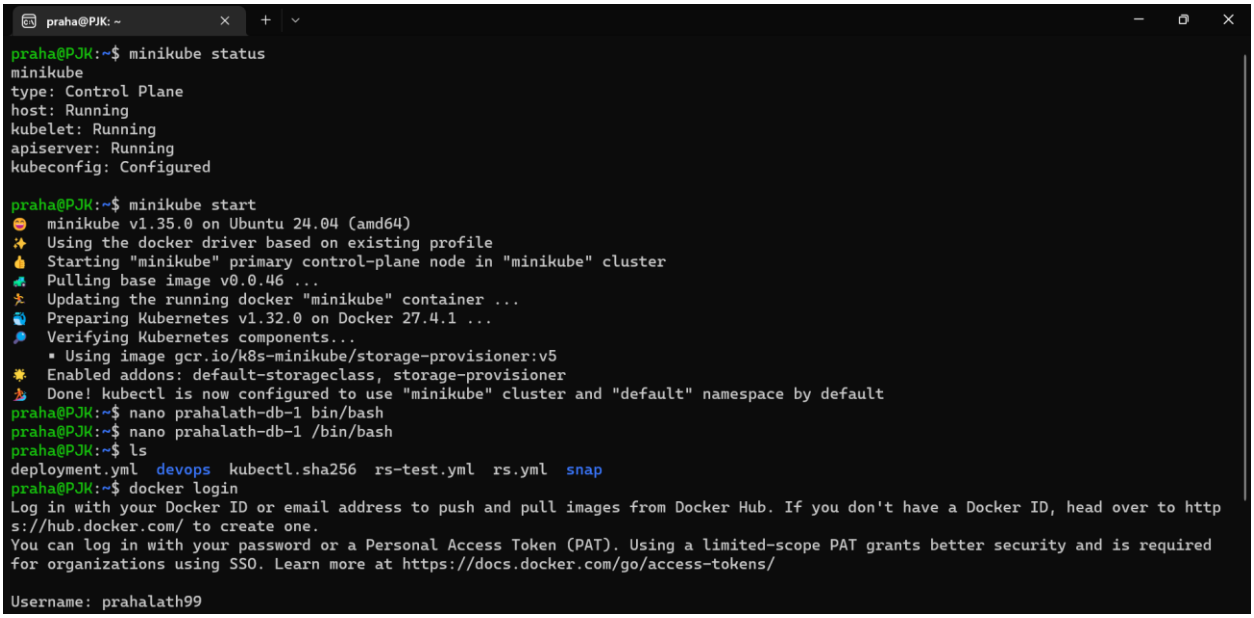
```
# Start containers in the foreground (logs will be shown)
docker compose up

# Stop containers
docker compose down

# Restart containers
docker compose restart

# View running containers
docker compose ps

# View logs of services
docker compose logs

# View logs of a specific service
docker compose logs <service_name>

# Build or rebuild services
docker compose build

# Stop containers without removing them
docker compose stop

# Start stopped containers
docker compose start
```

```
# Execute a command in a running container
docker compose exec <service_name> <command>

# Remove stopped containers, networks, and volumes
docker compose down --volumes

# Show configuration details
docker compose config

# Scale a service (e.g., run 3 instances of a service)
docker compose up --scale <service_name>=3 -d
```

Pipeline code

```
pipeline {
   agent any
   tools {maven "maven"}
   stages {
      stage('SCM') {
         steps {
            git branch: 'master', url: 'https://github.com/PJK-Prahalath/devops'
         }
      }
      stage('Build') {
         steps {
            sh 'mvn clean package'
```

```
        }
      }
      stage('build to images') {
        steps {
          script {
            sh 'docker build -t prahalath99/webapp1 .'
          }
        }
      }
      stage('push to hub') {
        steps {
          script {
            withDockerRegistry(credentialsId: 'docker_cred', toolName: 'docker', url:
'https://index.docker.io/v1/') {
                  sh 'docker push prahalath99/webapp1'
            }
          }
        }
      }

  }
}
```


praha@PJK: ~
praha@PJK:~$ minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured
praha@PJK:~$ minikube start
minikube v1.35.0 on Ubuntu 24.04 (amd64)
Using the docker driver based on existing profile
Starting "minikube" primary control-plane node in "minikube" cluster
Pulling base image v0.0.46 ...
Updating the running docker "minikube" container ...
Preparing Kubernetes v1.32.0 on Docker 27.4.1 ...
Verifying Kubernetes components...
▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
Enabled addons: default-storageclass, storage-provisioner
Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
praha@PJK:~$ nano prahalath-db-1 bin/bash
praha@PJK:~$ nano prahalath-db-1 /bin/bash
praha@PJK:~$ ls
deployment.yml devops kubectl.sha256 rs-test.yml rs.yml snap
praha@PJK:~$ docker login
Log in with your Docker ID or email address to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com/ to create one.
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT grants better security and is required for organizations using SSO. Learn more at https://docs.docker.com/go/access-tokens/
Username: prahalath99