Kubernetes Project

PROJECT 1: DEPLOY A MULTI-TIER WEB APPLICATION ON KUBERNETES

🛠 Prerequisites

- Kubernetes Cluster (minikube/kubeadm)

- Docker

- kubectl CLI

Folder Structure & File Usages

k8s-project/

│── mysql/ # MySQL Database Configuration

│ ├── mysql-pv.yaml # Persistent Volume for MySQL Data Storage

│ ├── mysql-secret.yaml # Stores MySQL Root Password Securely

│ ├── mysql-deployment.yaml # Deploys MySQL Database as a StatefulSet

│── flask/ # Flask Backend Configuration

│ ├── app.py # Flask API Code to Handle Requests

│ ├── Dockerfile # Flask App Containerization Instructions

│ ├── requirements.txt # Dependencies for Flask

│ ├── flask-deployment.yaml # Deploys Flask Application

│ ├── flask-service.yaml # Exposes Flask App as a Cluster Service

│── nginx/ # Nginx Configuration

│ ├── nginx-configmap.yaml # Reverse Proxy Configuration for Flask

│ ├── nginx-deployment.yaml # Deploys Nginx

│ ├── nginx-service.yaml # Exposes Nginx via NodePort

Step-by-Step Deployment Guide

Step 1: Build and Push Docker Image

1.1 Navigate to the Flask directory

cd flask

1.2 Build the Docker image

docker build -t dockerhub_username/flaskapp .

```
master@master-vm:~/multi-tier-application/flask$ docker build -t kirthiksubbiah/flaskapp .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  8.192kB
Step 1/6 : FROM python:3.8
 ---> 3ea6eaad4f17
Step 2/6 : WORKDIR /app
 ---> Using cache
 ---> 435bcd22c7d9
Step 3/6 : COPY app.py .
 ---> Using cache
 ---> bd37b9902a9d
Step 4/6 : COPY requirements.txt .
 ---> Using cache
 ---> d0e4d5d607cc
Step 5/6 : RUN pip install --no-cache-dir -r requirements.txt
 ---> Using cache
 ---> c5fc81f43e3f
Step 6/6 : CMD ["python", "app.py"]
 ---> Using cache
 ---> d6aa84b5beae
Successfully built d6aa84b5beae
Successfully tagged kirthiksubbiah/flaskapp:latest
```

1.3 Push the image to Docker Hub

docker push dockerhub_username/flaskapp

```
master@master-vm:~/multi-tier-application/flask$ docker push kirthiksubbiah/flaskapp
Using default tag: latest
The push refers to repository [docker.io/kirthiksubbiah/flaskapp]
8eb96175afb1: Pushed
4a27519a0380: Pushed
28aa611d3e8a: Pushed
4cad94de7904: Pushed
32ee710ca3c7: Pushed
1767e4d52b5a: Pushed
45b98afd69b3: Pushed
2bce433c3a29: Pushing [================>                                  ]  208.9MB/587.5MB
f91dc7a486d9: Pushing [==================================================>]  181.9MB
2bce433c3a29: Pushed
f91dc7a486d9: Pushed
d50132f2fe78: Pushing [=========================================>         ]  99.07MB/116.5MB
8: Pushed
latest: digest: sha256:5b6439ab975872fff83b372d93c6a19ab65d1458c201564a505929b540383761 size: 2628
```

Step 2: Apply Kubernetes Configurations

2.1 Deploy Flask application

kubectl apply -f flask-deployment.yaml

kubectl apply -f flask-service.yaml

```
master@master-vm:~/multi-tier-application/flask$ kubectl get nodes
NAME          STATUS   ROLES           AGE     VERSION
master-vm     Ready    control-plane   2d23h   v1.28.15
worker1-vm    Ready    <none>          2d23h   v1.28.15
worker2-vm    Ready    <none>          2d23h   v1.28.15
master@master-vm:~/multi-tier-application/flask$ kubectl apply -f flask-deployment.yaml
deployment.apps/flask-app created
master@master-vm:~/multi-tier-application/flask$ kubectl apply -f flask-service.yaml
service/flask-service created
master@master-vm:~/multi-tier-application/flask$ cd ../mysql
master@master-vm:~/multi-tier-application/mysql$ kubectl apply -f mysql-deployment.yaml
deployment.apps/mysql created
service/mysql created
master@master-vm:~/multi-tier-application/mysql$ kubectl apply -f mysql-pv.yaml
persistentvolume/mysql-pv created
persistentvolumeclaim/mysql-pvc created
```

2.2 Deploy MySQL database

cd ../mysql

kubectl apply -f mysql-deployment.yaml

kubectl apply -f mysql-pv.yaml

kubectl apply -f mysql-secret.yaml

```
master@master-vm:~/multi-tier-application/mysql$ kubectl apply -f mysql-deployment.yaml
deployment.apps/mysql created
service/mysql created
master@master-vm:~/multi-tier-application/mysql$ kubectl apply -f mysql-secret.yaml
secret/mysql-secret created
master@master-vm:~/multi-tier-application/mysql$ kubectl apply -f mysql-pv.yaml
persistentvolume/mysql-pv unchanged
persistentvolumeclaim/mysql-pvc unchanged
```

2.3 Deploy Nginx

cd ../nginx

kubectl apply -f nginx-configmap.yaml

kubectl apply -f nginx-deployment.yaml

kubectl apply -f nginx-service.yaml

```
master@master-vm:~/multi-tier-application/nginx$ nano nginx-configmap.yaml
master@master-vm:~/multi-tier-application/nginx$ kubectl apply -f nginx-deployment.yaml
deployment.apps/nginx unchanged
master@master-vm:~/multi-tier-application/nginx$ kubectl apply -f nginx-service.yaml
service/nginx-service created
master@master-vm:~/multi-tier-application/nginx$ kubectl apply -f nginx-configmap.yaml
configmap/nginx-config created
master@master-vm:~/multi-tier-application/nginx$
master@master-vm:~/multi-tier-application/nginx$ cd ..
```

Step 5: Initialize MySQL Database

5.1 Access MySQL inside the Pod

kubectl exec -it mysql-0 -- mysql -u root -p

```
master@master-vm:~/multi-tier-application/nginx$ kubectl exec -it mysql-66d468f74c-b4wk9 -- mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3
Server version: 5.7.44 MySQL Community Server (GPL)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
```

5.2 Create and populate the database

```
CREATE DATABASE mydb;
USE mydb;
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100),
    email VARCHAR(100)
);
INSERT INTO users (name, email) VALUES ('Alice', 'alice@example.com');
INSERT INTO users (name, email) VALUES ('Bob', 'bob@example.com');
SELECT * FROM users;
GRANT ALL PRIVILEGES ON mydb.* TO 'user'@'%';
```

```
mysql> INSERT INTO users (name, email) VALUES ('kirthiksubbiah', 'kirthiksubbiah@gmail.com');
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO users (name, email) VALUES ('kirthiksubbiahp', 'kirthiksubbiahp@gmail.com');
Query OK, 1 row affected (0.00 sec)

mysql> delete * from users where name=
```

```
FLUSH PRIVILEGES;
```


192.168.147.129:30007/users
JSON
Raw Data
Headers
Save
Copy
Collapse All
Expand All
Filter JSON
0:
email: "kirthiksubbiah@gmail.com"
id: 4
name: "kirthiksubbiah"
1:
email: "kirthiksubbiahp@gmail.com"
id: 5
name: "kirthiksubbiahp"