[Date]

IBM-CAD

DEPLOYMENT

TEAM LEAD : DHANA LAKSHMI S

TEAM M1 : KALPANA SRI R

TEAM M2 : JEEVITHA V

DEPARTMENT OF CSE

KARPAGAM INSTITUTE OF TECHNOLOGY,COIMBATORE

**To deploy a web application in an IBM Kubernetes cluster**

**1. Set up an IBM Kubernetes Cluster:**

**- Log in to our IBM Cloud account and navigate to the Kubernetes Service.**

**- Create a new Kubernetes cluster or choose an existing cluster.**

**- Configure the cluster by specifying the number of worker nodes, region, and other relevant settings.**

**- Wait for the cluster to be provisioned and become ready.**

**2. Containerize our Web Application:**

**- Containerize our web application using Docker. Create a Dockerfile that describes the application's dependencies and how to build the container image.**

**3. Build and Push the Container Image:**

**- Build the container image for our web application using the Dockerfile.**

**- Tag the image with the registry address provided by IBM Container Registry.**

**- Push the container image to the IBM Container Registry for storage and distribution.**

**4. Define Kubernetes Deployment Configuration:**

**- Create a Kubernetes YAML file that describes the deployment configuration for our web application.**

**- Specify details like the container image, resource requirements, environment variables, ports, and other settings.**

**5. Apply Deployment Configuration:**

**- Use the `kubectl` command-line tool to apply the Kubernetes YAML file and deploy our web application to the IBM Kubernetes cluster.**

**- Run the command to apply the deployment configuration:**

**kubectl apply -f your-deployment-config.yaml**

**```**

**6. Expose the Web Application:**

**- Create a Kubernetes Service to expose our web application to the internet or internal network.**

**- Define the service type, ports, and any necessary load balancing configurations in the Kubernetes YAML file.**

**- Apply the updated YAML file to create the service:**

**kubectl apply -f your-service-config.yaml**

**```**

**7. Monitor and Manage the Deployment:**

**- Use IBM Cloud Monitoring or other monitoring tools to track the health, performance, and logs of our deployed web application.**

**- Utilize `kubectl` commands or IBM Cloud console to manage and scale our deployment as needed.**

**8. Domain Configuration (Optional):**

**- As we have a custom domain, configure DNS settings to point to our web application's Kubernetes service.**

**DOCKER FILE**

# Use a base image

FROM python:3.9

# Set the working directory

WORKDIR /app

# Copy the application files to the container

COPY . /app

# Install dependencies

RUN pip install -r requirements.txt

# Expose the desired port

EXPOSE 8080

# Define the command to run the application

CMD [ "python", "app.py" ]

**DEPLOYMENT YAML FOR KUBERNETES**

apiVersion: apps/v1

kind: Deployment

metadata:

name: web-app-deployment

labels:

app: web-app

spec:

replicas: 3

selector:

matchLabels:

app: web-app

template:

metadata:

labels:

app: web-app

spec:

containers:

- name: web-app-container

image: registry.example.com/your-image:latest

ports:

- containerPort: 8080

**SERVICE YAML FOR KUBERNETES**

apiVersion: v1

kind: Service

metadata:

name: web-app-service

spec:

selector:

app: web-app

ports:

- protocol: TCP

port: 80

targetPort: 8080

type: LoadBalancer

**PYTHON CODE TO CONNECT TO IBM CLOUD AND DATABASE**

import mysql.connector

from getpass import getpass

# Get the database credentials from the environment or securely prompt the user

db\_host = getpass("Enter the database host:")

db\_username = getpass("Enter the database username:")

db\_password = getpass("Enter the database password:")

# Connect to the MySQL database

cnx = mysql.connector.connect(

host=db\_host,

user=db\_username,

password=db\_password,

database="your\_database"

)

# ...

# Perform database operations (insert, select, update, delete)

# ...

# Close the database connection

cnx.close()