DevOps

Day – 4

Assignment

Name:DEEPIKA P M

Roll No:22CSR041

# 1.Namespace

A Namespace is a logical partition within a Kubernetes cluster.

It allows you to divide resources like Pods, Services, and Deployments into different environments (e.g., dev, test, prod).

Think of it as a virtual cluster inside the physical Kubernetes cluster.

Useful for managing large projects with multiple teams.

# Replica

A Replica ensures that a specified number of identical Pods are running in your cluster.

If a Pod fails, Kubernetes automatically replaces it using replicas to maintain the desired state.

It provides scalability and fault tolerance.

# Pod

A **Pod** is the smallest deployable unit in Kubernetes.

It contains one or more containers (e.g., Docker containers).

Containers in a Pod share the same network namespace, storage, and lifecycle.

# Deployment

A Deployment manages the creation and scaling of Pods using ReplicaSets.

It provides automated rollouts and rollbacks.

It ensures your application is always available by managing its state.

# POD

1. Create a pod using run

command

$ kubectl run <podname> --image=<imagename> --port=<containerport>

$ kubectl run my-pod -image=nginx --port=80

1. View all the pods

(In default namespace)

$ kubectl get pods

(In All namespace)

$ kubectl get pods -A

# For a specific

namespace $ kubectl get pods -n kube-system

# For a specific type $ kubectl get pods <podname> $ kubectl get pods <podname> -o wide $ kubectl get pods <podname> -o yaml $ kubectl get pods <podname> -o json

1. Describe a pod (View

Pod details)

$ kubectl describe pod

<pod-name> $ kubectl describe pod

my-pod

1. View Logs of a pod

$ kubectl logs <pod-

name>

$ kubectl logs my-pod

1. Execute any command

inside Pod (Inside Pod OS) $ kubectl exec <pod-

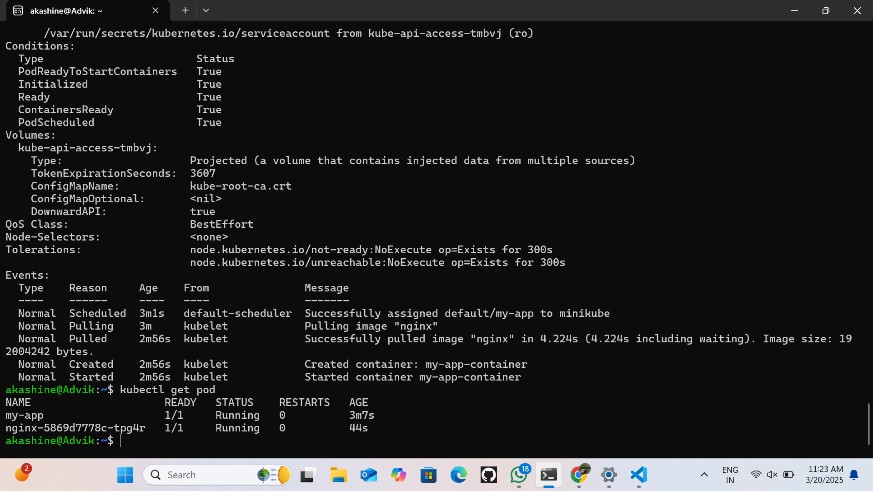
name> -- <command>

apiVersion: v1 kind: Pod metadata: name: my-pod

labels: app: my-web-app type: backend spec: containers: - name: nginx-container image: nginx

ports:

- containerPort: 80



# Replica

1. Create ReplicaSet by executing above YAML file $ kubectl create -f rstest.yml # Do necessary

modifications if exist, else

create new

$ kubectl apply -f rstest.yml

# Completely Modify Pod

Template $ kubectl replace –f rstest.yml

1. View ReplicaSets

$ kubectl get replicasets

$ kubectl get rs

$ kubectl get rs –o wide $ kubectl get rs <replicaset-name> –o json $ kubectl get rs <replica-

set-name> –o yaml

1. View ReplicaSet

Description

$ kubectl describe rs

<replica-set-name> 4. We can modify generated/updated YAML

file

$ kubectl edit rs <replica-

set-name> ## change replicas: count to any other value then

(ESC):wq

# We can modify our YAML file and then execute apply command $ kubectl apply -f rstest.yml

## We can Even scale using command also

$ kubectl scale replicaset <replicaset-name> -replicas=<desired-replicacount>

5. Delete ReplicaSet $ kubectl delete rs

<replica-set-name> $ kubectl delete -f rstest.yml

apiVersion: apps/v1 kind: Deployment metadata: name: my-deploy

labels: name: my-deploy spec: replicas: 3 selector: matchLabels: apptype: web-backend

strategy: type: RollingUpdate template: metadata:

labels: apptype: web-

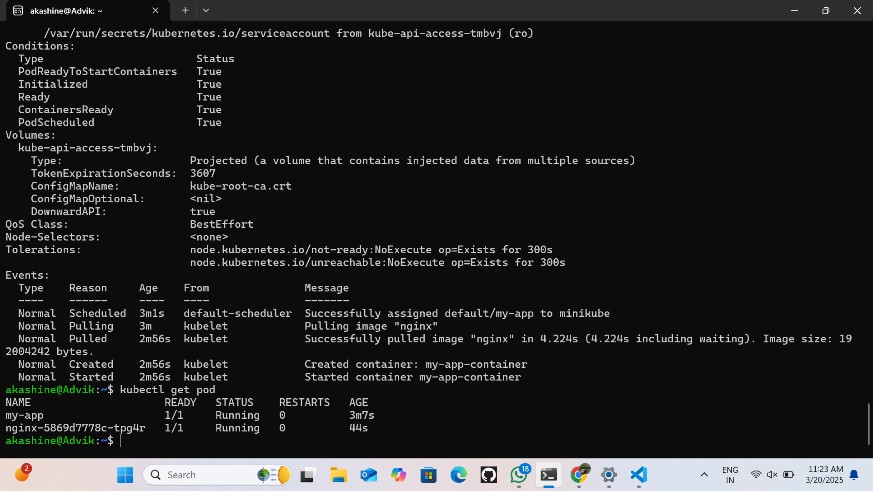
backend

spec: containers: - name: my-app

image: nginx ports:

- containerPort:

7070



# DEPLOY

1. Create Deployment by executing above YAML file $ kubectl create -f webdeploy.yml # Do necessary

modifications if exist, else

create new $ kubectl create -f webdeploy.yml

# Completely Modify Pod

Template $ kubectl replace –f webdeploy.yml #Create deploy kubectl create

deployment webnginx2 -image=nginx:latest -replicas=1

1. View Deployments $ kubectl get

deployments

$ kubectl get deploy $ kubectl get deploy -o wide

$ kubectl get deploy <deployment-name> -o

json

$ kubectl get deploy <deployment-name> -o yaml

1. View Deployment

Description

$ kubectl describe deploy

<deployment-name> 4. We can modify generated/updated YAML

file

$ kubectl edit deploy

<deployment-name> ## change replicas: count to any other value then

(ESC):wq

# We can modify our YAML file and then execute apply command $ kubectl apply -f webdeploy.yml

## We can Even scale using command also

$ kubectl scale deploy <deployment-name> -replicas=<desired-replicacount>

5. Delete Deployment

$ kubectl delete deploy

<deployment-name> $ kubectl delete -f webdeploy.yml

apiVersion: apps/v1 kind: Deployment metadata: name: my-deploy

labels: name: my-deploy spec: replicas: 1 selector: matchLabels: apptype: web-backend

strategy: type: RollingUpdate template: metadata:

labels: apptype: web-

backend

spec: containers: - name: my-app

image:

ports:

- containerPort: 7070

---

apiVersion: v1 kind: Service

metadata:

name: my-service

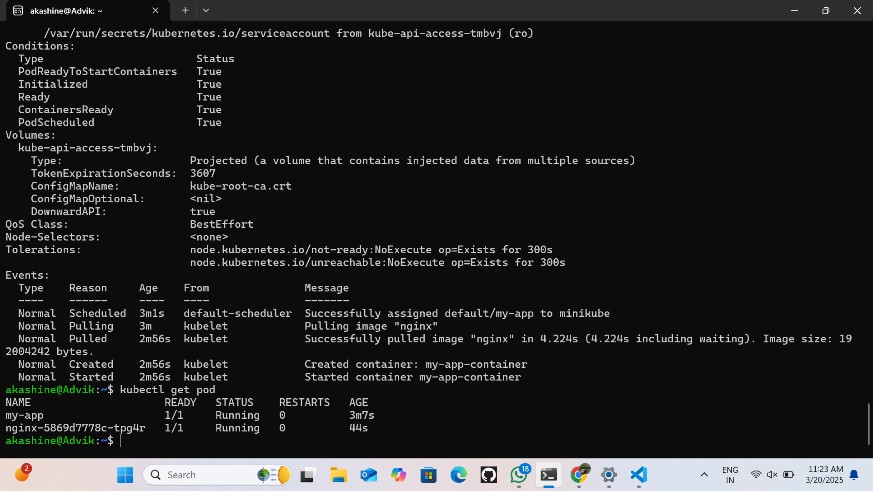
labels:

app: my-service type: backend-app spec:

type: NodePort

ports: - targetPort: 7070 port: 7070 nodePort: 30002 selector:

apptype: web-backend



# Minikube service

Minikube service

#need to create a yml file

sudo nano deployment.yml

apiVersion: apps/v1 kind: Deployment metadata: name: my-deploy

labels: name: my-deploy spec: replicas: 1 selector: matchLabels: apptype: web-backend

strategy: type: RollingUpdate template: metadata:

labels: apptype: web-

backend

spec: containers: - name: my-app

image: ports:

- containerPort: 9000

---

apiVersion: v1 kind: Service metadata: name: my-service

labels: app: my-service spec: type: NodePort

ports: - port: 9000 targetPort: 8080 nodePort: 30002 selector: apptype: web-backend

#Apply the deployment

kubectl apply -f

deployment.yml

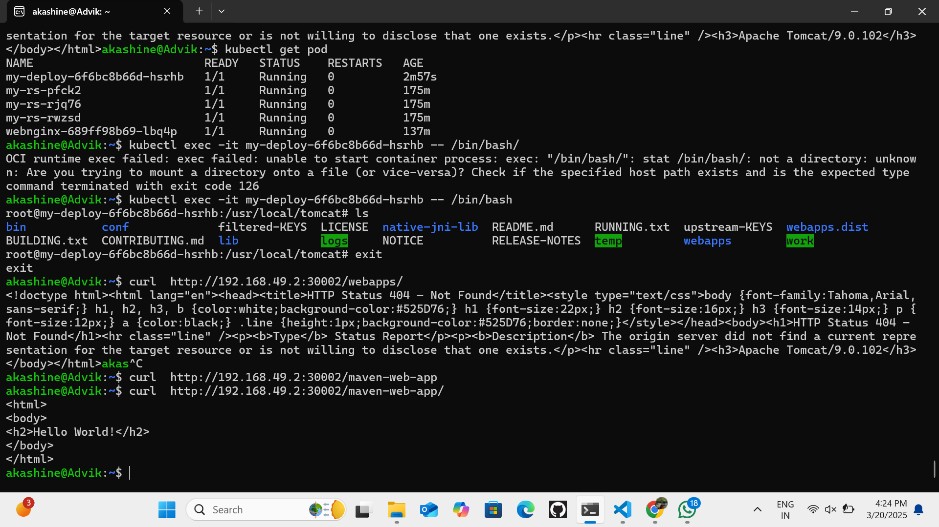
#replace the deployment

kubectl replace -f

deployment.yml

#Run the service minikube service myservice

#curl the url curl <url>/<file\_name>/



# Namespace

# To create a namespace:

$ kubectl create

namespace <namespacename>

$ kubectl create ns mybank

# To switch to a specific

namespace: (make this as default type) $ kubectl config setcontext --current --

namespace=<namespacename>

# To list all namespaces:

$ kubectl get namespaces # To get resources within a specific namespace: $ kubectl get <resourcetype> -n <namespacename>

$ kubectl get deploy -n

my-bank

$ kubectl get deploy --

namespace my-bank

$ kubectl get all --

namespace my-bank # To delete a namespace and all associated resources: $ kubectl delete

namespace <namespacename>

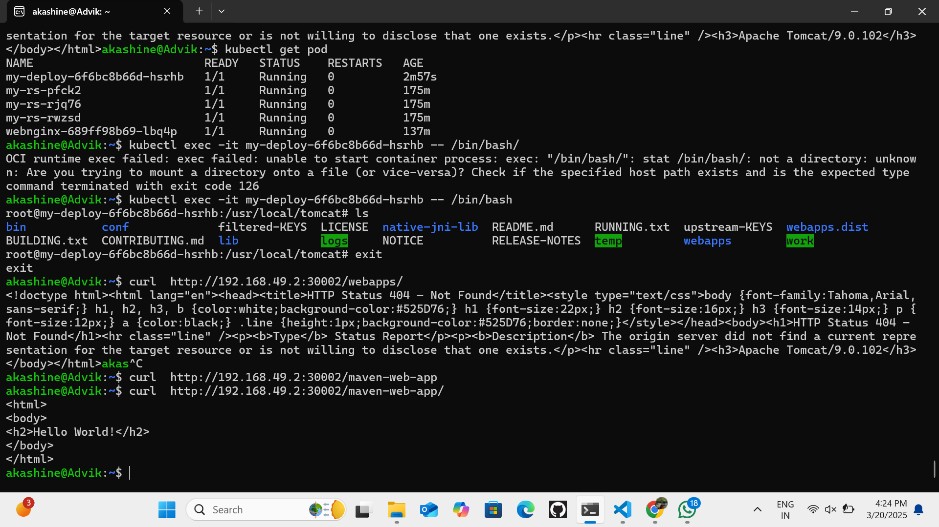
$ kubectl delete ns mybank

kubectl create ns mydeploy kubectl apply -f deploy.yml -n mydeploy

apiVersion: v1

kind: Namespace metadata: name: my-demo-ns

apiVersion: v1 kind: Pod metadata: name: my-pod namespace: my-demo-ns spec: containers: - name: my-container image: nginx:latest



# Namespace yml

