**Launch a Simple Pod Deploy a POD running the nginx:alpine image and expose it on port 80.**

**kubectl run nginx-pod --image=nginx:alpine --port=80**

shubham.sahu@IN-IT18554 sample-node-app % **kubectl run nginx-pod --image=nginx:alpine --port=80**

pod/nginx-pod created

shubham.sahu@IN-IT18554 sample-node-app % kubectl get pod

NAME READY STATUS RESTARTS AGE

nginx-pod 0/1 ContainerCreating 0 8s

**Run a Busybox Pod, Run a POD with the busybox:1.32.0-uclibc image. Name the container and POD after your name and execute the following commands inside it: echo "Hello <your-name>" sleep 3000**

kubectl run shubham-busybox \

--image=busybox:1.32.0-uclibc \

--restart=Never \

--command -- /bin/sh -c "echo 'Hello shubham'; sleep 3000"

shubham.sahu@IN-IT18554 sample-node-app % kubectl run shubham-busybox \

--image=busybox:1.32.0-uclibc \

--restart=Never \

--command -- /bin/sh -c "echo 'Hello shubham'; sleep 3000"

pod/shubham-busybox created

shubham.sahu@IN-IT18554 sample-node-app % kubectl get pod

NAME READY STATUS RESTARTS AGE

nginx-pod 1/1 Running 0 9m17s

shubham-busybox 0/1 ContainerCreating 0 6s

shubham.sahu@IN-IT18554 sample-node-app % kubectl logs shubham-busybox

Hello shubham

**Create a Namespace and Pod Create a namespace named Chimera and run a POD using the image httpd:2.4.41-alpine. Set the restart policy of the pod to Never and check the result.**

**Create the namespace:**

kubectl create namespace chimera

**Create the pod in the chimera namespace:**

kubectl run httpd-pod \

--image=httpd:2.4.41-alpine \

--restart=Never \

--namespace=chimera

**Verify:**

kubectl get pods -n chimera

**Logs:**

shubham.sahu@IN-IT18554 sample-node-app % kubectl create namespace chimera

namespace/chimera created

shubham.sahu@IN-IT18554 sample-node-app % kubectl run httpd-pod \

--image=httpd:2.4.41-alpine \

--restart=Never \

--namespace=chimera

pod/httpd-pod created

shubham.sahu@IN-IT18554 sample-node-app % kubectl get pods -n chimera

NAME READY STATUS RESTARTS AGE

httpd-pod 0/1 ContainerCreating 0 11s

shubham.sahu@IN-IT18554 sample-node-app % kubectl get pods -n chimera

NAME READY STATUS RESTARTS AGE

httpd-pod 1/1 Running 0 19s

**Change Restart Policy Modify the restart policy of the previously created pod to Always and ensure the pod is running. If it isn’t, check the namespace and pod logs to troubleshoot.**

Cannot edit a pod’s restart policy after it's created.

1. **Delete the old pod:**

kubectl delete pod httpd-pod -n chimera

1. **Recreate with restart policy Always**

kubectl run httpd-pod \

--image=httpd:2.4.41-alpine \

--restart=Always \

--namespace=chimera

1. **Check pod is running:**

kubectl get pods -n chimera

1. **If it's not running, debug:**

kubectl describe pod httpd-pod -n chimera

kubectl logs httpd-pod -n chimera

**Logs:**shubham.sahu@IN-IT18554 sample-node-app % kubectl delete pod httpd-pod -n chimera

pod "httpd-pod" deleted

shubham.sahu@IN-IT18554 sample-node-app % kubectl run httpd-pod \

--image=httpd:2.4.41-alpine \

--restart=Always \

--namespace=chimera

pod/httpd-pod created

shubham.sahu@IN-IT18554 sample-node-app % kubectl get pods -n chimera

NAME READY STATUS RESTARTS AGE

httpd-pod 1/1 Running 0 10s

shubham.sahu@IN-IT18554 sample-node-app % kubectl describe pod httpd-pod -n chimera

Name: httpd-pod

Namespace: chimera

Priority: 0

Service Account: default

Node: minikube/192.168.49.2

Labels: run=httpd-pod

Annotations: <none>

Status: Running

IP: 10.244.0.6

IPs:

IP: 10.244.0.6

Containers:

httpd-pod:

Container ID: docker://f02dece18e63f5af3143b1ebabb9fffb81c5ce09332b1a9501e754b4a39efd19

Image: httpd:2.4.41-alpine

Image ID: docker-pullable://httpd@sha256:06ad90574c3a152ca91ba9417bb7a8f8b5757b44d232be12037d877e9f8f68ed

Port: <none>

Host Port: <none>

State: Running

Ready: True

Restart Count: 0

Environment: <none>

Mounts:

/var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-k6lrs (ro)

Conditions:

Type Status

PodReadyToStartContainers True

Initialized True

Ready True

ContainersReady True

PodScheduled True

Volumes:

kube-api-access-k6lrs:

Type: Projected (a volume that contains injected data from multiple sources)

TokenExpirationSeconds: 3607

ConfigMapName: kube-root-ca.crt

ConfigMapOptional: <nil>

DownwardAPI: true

QoS Class: BestEffort

Node-Selectors: <none>

Tolerations: node.kubernetes.io/not-ready:NoExecute op=Exists for 300s

node.kubernetes.io/unreachable:NoExecute op=Exists for 300s

Events:

Type Reason Age From Message

---- ------ ---- ---- -------

Normal Scheduled 52s default-scheduler Successfully assigned chimera/httpd-pod to minikube

Normal Pulled 52s kubelet Container image "httpd:2.4.41-alpine" already present on machine

Normal Created 52s kubelet Created container: httpd-pod

Normal Started 52s kubelet Started container httpd-pod

**Run Nginx Web Server (Deployment) Deploy an Nginx web server using the following specifications: Image: nginx:alpine Container name: front-end Replica count: 2 Port: 80 Label: company:globant**

apiVersion: apps/v1

kind: Deployment

metadata:

name: nginx-deployment

labels:

app: nginx

company: globant

spec:

replicas: 2

selector:

matchLabels:

app: nginx

template:

metadata:

labels:

app: nginx

company: globant

spec:

containers:

- name: front-end

image: nginx:alpine

ports:

- containerPort: 80

 **Apply deployment:**

kubectl apply -f nginx-deployment.yaml

**Verify:**

kubectl get deployments

kubectl get pods -l app=nginx

**Logs:**

shubham.sahu@IN-IT18554 KuberneteTasks % kubectl apply -f nginx-deployment.yaml

deployment.apps/nginx-deployment created

shubham.sahu@IN-IT18554 KuberneteTasks % kubectl get all

NAME READY STATUS RESTARTS AGE

pod/nginx-deployment-85cbf78868-655vn 1/1 Running 0 11s

pod/nginx-deployment-85cbf78868-wxsrr 1/1 Running 0 11s

pod/shubham-busybox 0/1 Completed 0 29h

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

service/kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 29h

NAME READY UP-TO-DATE AVAILABLE AGE

deployment.apps/nginx-deployment 2/2 2 2 11s

NAME DESIRED CURRENT READY AGE

replicaset.apps/nginx-deployment-85cbf78868 2 2 2 11s

shubham.sahu@IN-IT18554 KuberneteTasks % kubectl get deployments

NAME READY UP-TO-DATE AVAILABLE AGE

nginx-deployment 2/2 2 2 36s

shubham.sahu@IN-IT18554 KuberneteTasks % kubectl get pods -l app=nginx

NAME READY STATUS RESTARTS AGE

nginx-deployment-85cbf78868-655vn 1/1 Running 0 46s

nginx-deployment-85cbf78868-wxsrr 1/1 Running 0 46s

shubham.sahu@IN-IT18554 KuberneteTasks %

### **Scale Nginx from 2 to 5 Replicas**

kubectl scale deployment nginx-deployment --replicas=5

kubectl get pods -l app=nginx

**Logs:**

shubham.sahu@IN-IT18554 KuberneteTasks % kubectl scale deployment nginx-deployment --replicas=5

deployment.apps/nginx-deployment scaled

shubham.sahu@IN-IT18554 KuberneteTasks % kubectl get pods -l app=nginx

NAME READY STATUS RESTARTS AGE

nginx-deployment-85cbf78868-655vn 1/1 Running 0 57m

nginx-deployment-85cbf78868-bfj5r 1/1 Running 0 5s

nginx-deployment-85cbf78868-mlpzs 1/1 Running 0 5s

nginx-deployment-85cbf78868-vcsv6 1/1 Running 0 5s

nginx-deployment-85cbf78868-wxsrr 1/1 Running 0 57m

shubham.sahu@IN-IT18554 KuberneteTasks %

**Update the image of the Nginx deployment from nginx:alpine to nginx:1.16.1-alpine by:**

**- Using the command line (without modifying the manifest file)**

**- Modifying the manifest file and reverting the image to nginx:alpine**

**- Ensure all replicas are running once the image change is applied**

**Update the deployment image using kubectl set image:**

kubectl set image deployment/nginx-deployment front-end=nginx:1.16.1-alpine

**Check the rollout status**

kubectl rollout status deployment/nginx-deployment

**Verify the updated image:**

kubectl describe deployment nginx-deployment | grep Image

**Logs:**   
  
shubham.sahu@IN-IT18554 KuberneteTasks % kubectl set image deployment/nginx-deployment front-end=nginx:1.16.1-alpine

deployment.apps/nginx-deployment image updated

shubham.sahu@IN-IT18554 KuberneteTasks % kubectl rollout status deployment nginx-deployment

deployment "nginx-deployment" successfully rolled out

shubham.sahu@IN-IT18554 KuberneteTasks % ls

nginx-deployment.yaml

shubham.sahu@IN-IT18554 KuberneteTasks % kubectl describe deployment nginx-deployment | grep Image

Image: nginx:1.16.1-alpine

shubham.sahu@IN-IT18554 KuberneteTasks %

### 

### 

### 

### **Modifying the manifest file and reverting the image to nginx:alpine**

Open your nginx-deployment.yaml file.

Locate the container section and revert the image like this:

**image: nginx:alpine**

Apply updated manifest:

kubectl apply -f nginx-deployment.yaml

**Monitor rollout:**

kubectl rollout status deployment nginx-deployment

**Logs:**

shubham.sahu@IN-IT18554 KuberneteTasks % kubectl apply -f nginx-deployment.yaml

deployment.apps/nginx-deployment configured

shubham.sahu@IN-IT18554 KuberneteTasks % kubectl rollout status deployment nginx-deployment

deployment "nginx-deployment" successfully rolled out

shubham.sahu@IN-IT18554 KuberneteTasks % kubectl describe deployment nginx-deployment | grep Image

Image: nginx:alpine

**Ensure all replicas are running once the image change is applied**

kubectl get pods -l app=nginx

shubham.sahu@IN-IT18554 KuberneteTasks % kubectl get pods -l app=nginx

NAME READY STATUS RESTARTS AGE

nginx-deployment-85cbf78868-8c2wq 1/1 Running 0 119s

nginx-deployment-85cbf78868-s9nh6 1/1 Running 0 118s

shubham.sahu@IN-IT18554 KuberneteTasks %

**Generate Manifest for Nginx Deployment Launch the Nginx web server using the nginx:1.14.2-alpine image via the command line, then generate a manifest file from the running deployment or pod and save it as my-own-server.yaml.**

### **Create the Deployment via Command Line**

kubectl create deployment my-own-server \

--image=nginx:1.14.2-alpine

### **Generate the manifest file (my-own-server.yaml)**

kubectl get deployment my-own-server -o yaml > my-own-server.yaml

**Logs:**shubham.sahu@IN-IT18554 KuberneteTasks % kubectl create deployment my-own-server \

--image=nginx:1.14.2-alpine

deployment.apps/my-own-server created

shubham.sahu@IN-IT18554 KuberneteTasks % kubectl get deployment my-own-server -o yaml > my-own-server.yaml

shubham.sahu@IN-IT18554 KuberneteTasks % ls

my-own-server.yaml nginx-deployment.yaml

**Commands to List and Switch Contexts List and explain the following Kubernetes commands: Command to check the current context of the connected cluster Command to switch to a new, available cluster context Command to view all available contexts to connect to Run a Pod with Init Container Run a pod with the following specifications:**

**Run a Pod with Init Container Run a pod with the following specifications: Main container: Image: nginx:alpine Init container: Image: busybox:1.32.0-uclibc Command: sleep 5000 After running the pod, check the logs of both containers and note down the commands used to view the logs.**

## **Kubernetes Context Commands**

kubectl config current-context

kubectl config use-context <context-name>

### **List all available contexts:**

kubectl config get-contexts

**Run Pod with Init Container**

Nginx-with-init.yaml

apiVersion: v1

kind: Pod

metadata:

name: nginx-with-init

spec:

initContainers:

- name: init-busybox

image: busybox:1.32.0-uclibc

command: ["sleep", "5000"]

containers:

- name: nginx

image: nginx:alpine

ports:

- containerPort: 80



**kubectl apply -f nginx-with-init.yaml**

**Check logs for the init container:**

kubectl logs nginx-with-init -c init-busybox

**Check logs for the main container:**

kubectl logs nginx-with-init -c nginx

**Inspect container statuses:**

kubectl describe pod nginx-with-init