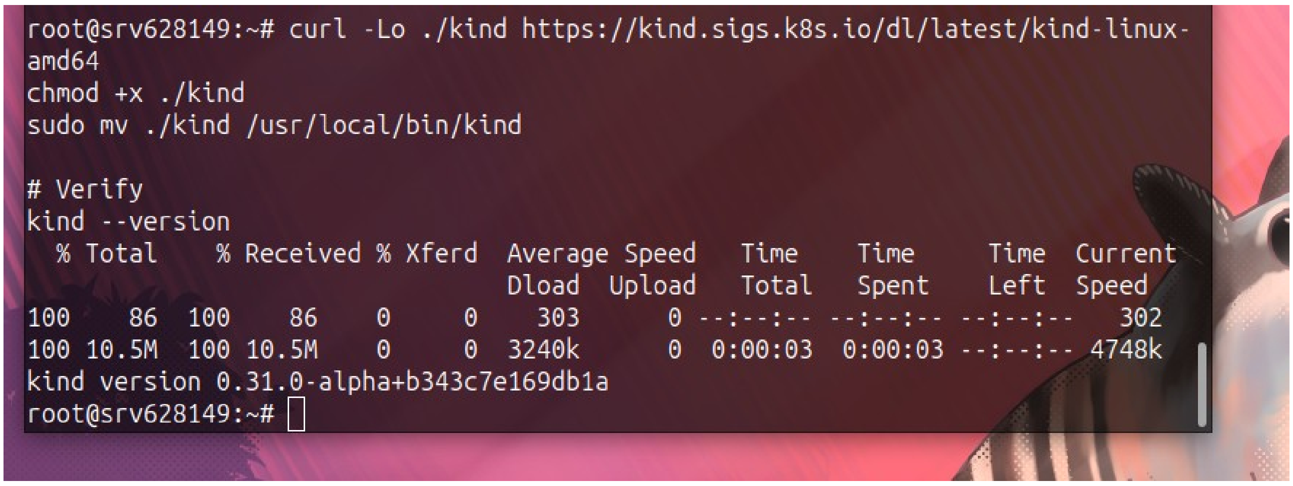
1. Exercise 1: Setting Up Your Kubernetes Cluster

commands used:	**curl -Lo ./kind https://kind.sigs.k8s.io/dl/latest/kind-linux-amd64** **chmod +x ./kind** **sudo mv ./kind /usr/local/bin/kind** **kind –version** **kind create cluster** **kubectl cluster-info** **kubectl get nodes**

curl -Lo ./kind https://kind.sigs.k8s.io/dl/latest/kind-linux-amd64
chmod +x ./kind
sudo mv ./kind /usr/local/bin/kind



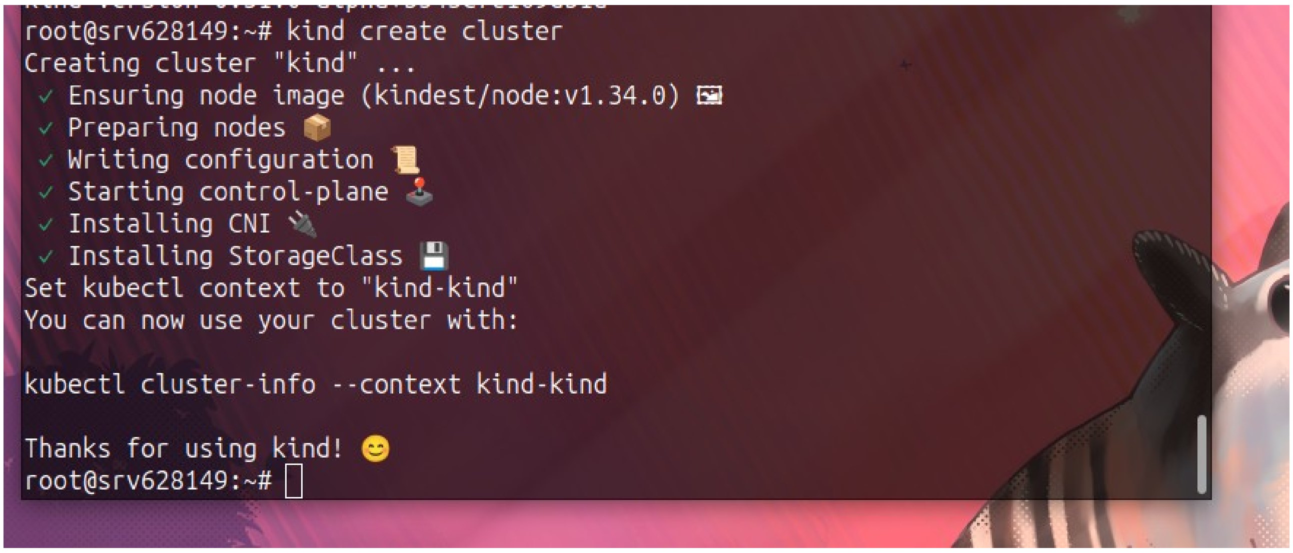
```
root@srv628149:~# curl -Lo ./kind https://kind.sigs.k8s.io/dl/latest/kind-linux-
amd64
chmod +x ./kind
sudo mv ./kind /usr/local/bin/kind

# Verify
kind --version
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    86  100    86    0     0    303      0 --:--:-- --:--:-- --:--:--   302
100 10.5M  100 10.5M    0     0  3240k      0  0:00:03  0:00:03 --:--:-- 4748k
kind version 0.31.0-alpha+b343c7e169db1a
root@srv628149:~# 
```

Verify
kind --version

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    86  100    86    0     0    303      0 --:--:-- --:--:-- --:--:--   302
100 10.5M  100 10.5M    0     0  3240k      0  0:00:03  0:00:03 --:--:-- 4748k
kind version 0.31.0-alpha+b343c7e169db1a
```

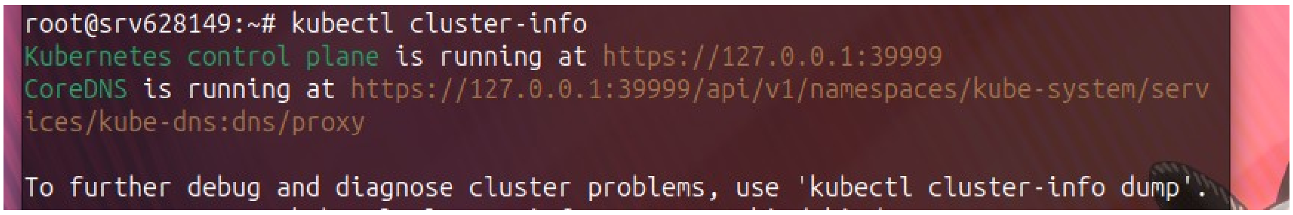
kind create cluster

```
Creating cluster "kind" ...
 ✓ Ensuring node image (kindest/node:v1.34.0) 🖼
 ✓ Preparing nodes 📦
 ✓ Writing configuration 📜
 ✓ Starting control-plane 🕹️
 ✓ Installing CNI 🔌
 ✓ Installing StorageClass 💾
Set kubectl context to "kind-kind"
You can now use your cluster with:

kubectl cluster-info --context kind-kind

Thanks for using kind! 😊
```



```
root@srv628149:~# kind create cluster
Creating cluster "kind" ...
 ✓ Ensuring node image (kindest/node:v1.34.0) 🖼
 ✓ Preparing nodes 📦
 ✓ Writing configuration 📜
 ✓ Starting control-plane 🕹️
 ✓ Installing CNI 🔌
 ✓ Installing StorageClass 💾
Set kubectl context to "kind-kind"
You can now use your cluster with:

kubectl cluster-info --context kind-kind

Thanks for using kind! 😊
root@srv628149:~#
```

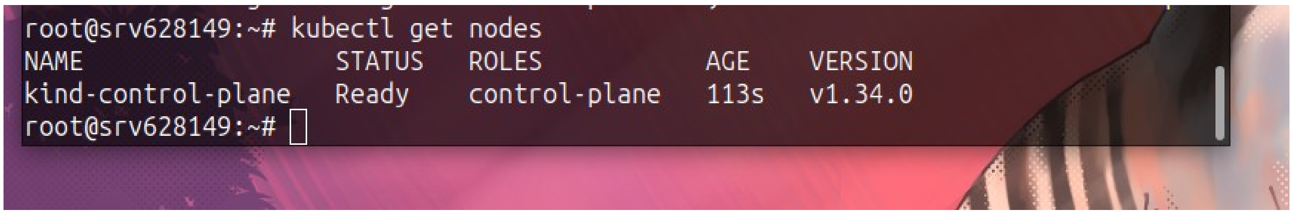
kubectl cluster-info
Kubernetes control plane is running at https://127.0.0.1:39999
CoreDNS is running at https://127.0.0.1:39999/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy



```
root@srv628149:~# kubectl cluster-info
Kubernetes control plane is running at https://127.0.0.1:39999
CoreDNS is running at https://127.0.0.1:39999/api/v1/namespaces/kube-system/serv
ices/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
```

kubectl get nodes

```
NAME                STATUS   ROLES           AGE    VERSION
kind-control-plane  Ready    control-plane   113s   v1.34.0
```



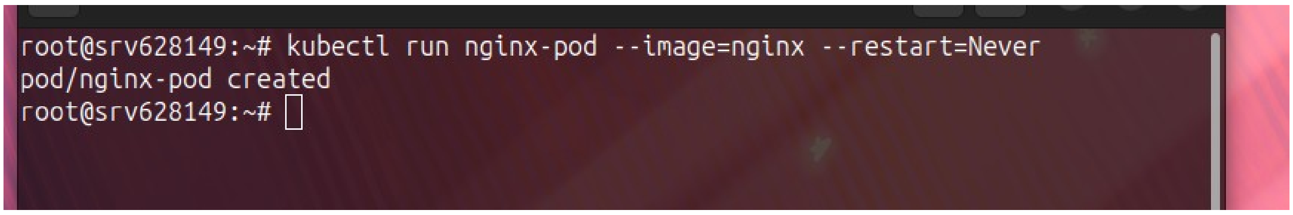
```
root@srv628149:~# kubectl get nodes
NAME                 STATUS   ROLES           AGE    VERSION
kind-control-plane   Ready    control-plane   113s   v1.34.0
root@srv628149:~#
```

2. Exercise 2: Creating and Managing Pods

commands used:

```
kubectl run nginx-pod --image=nginx –restart=Never
kubectl get pods
kubectl logs nginx-pod
kubectl expose pod nginx-pod --type=NodePort –port=9090
kubectl delete pod nginx-pod
```

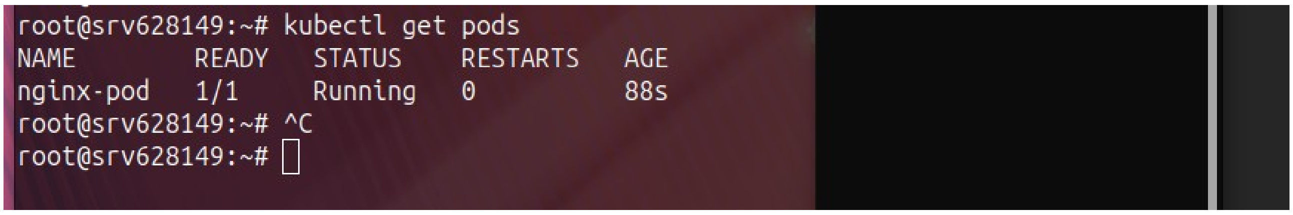
kubectl run nginx-pod --image=nginx –restart=Never :
pod/nginx-pod created



```
root@srv628149:~# kubectl run nginx-pod --image=nginx --restart=Never
pod/nginx-pod created
root@srv628149:~#
```

kubectl get pods :

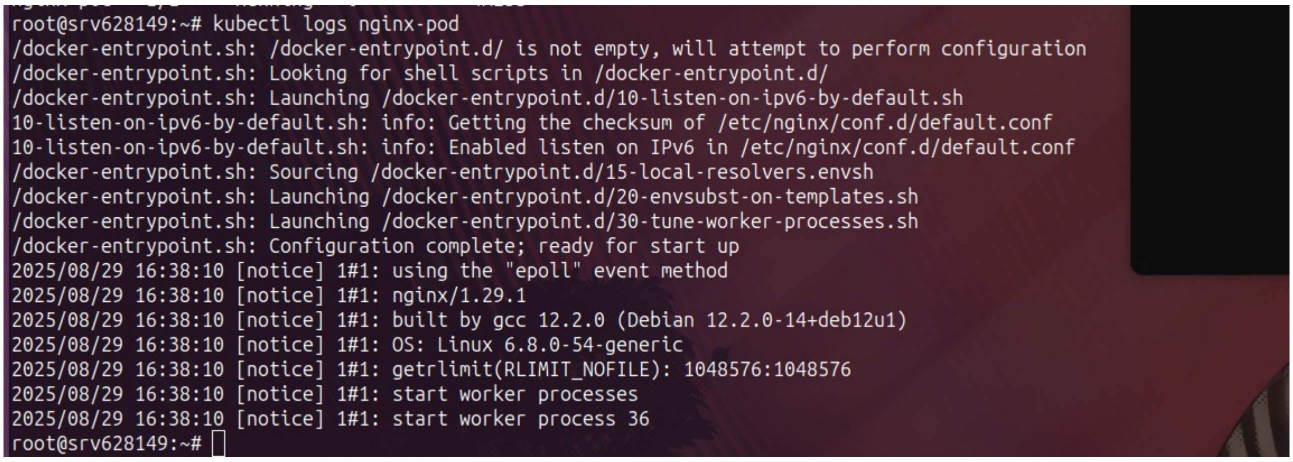
```
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod   1/1     Running   0          88s
```



```
root@srv628149:~# kubectl get pods
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod   1/1     Running   0          88s
root@srv628149:~# ^C
root@srv628149:~#
```

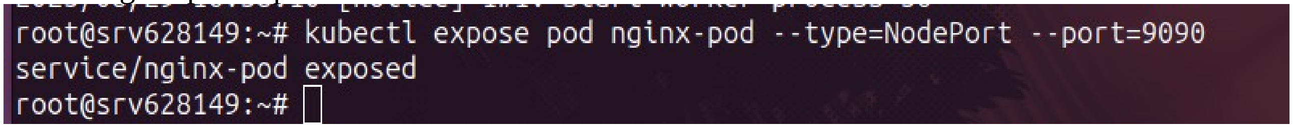
kubectl logs nginx-pod :

```
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2025/08/29 16:38:10 [notice] 1#1: using the "epoll" event method
2025/08/29 16:38:10 [notice] 1#1: nginx/1.29.1
2025/08/29 16:38:10 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14+deb12u1)
2025/08/29 16:38:10 [notice] 1#1: OS: Linux 6.8.0-54-generic
2025/08/29 16:38:10 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2025/08/29 16:38:10 [notice] 1#1: start worker processes
2025/08/29 16:38:10 [notice] 1#1: start worker process 36
```



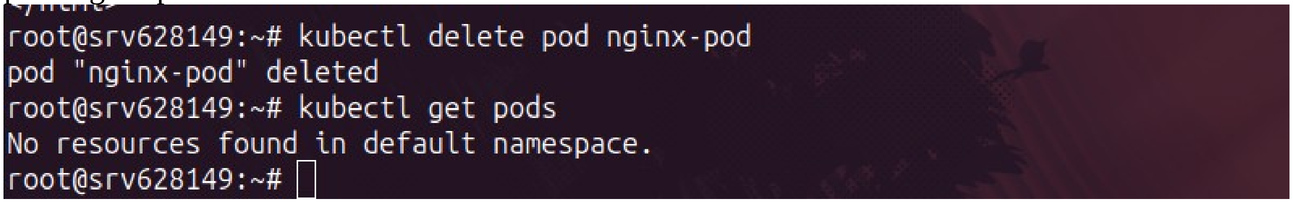
```
root@srv628149:~# kubectl logs nginx-pod
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2025/08/29 16:38:10 [notice] 1#1: using the "epoll" event method
2025/08/29 16:38:10 [notice] 1#1: nginx/1.29.1
2025/08/29 16:38:10 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14+deb12u1)
2025/08/29 16:38:10 [notice] 1#1: OS: Linux 6.8.0-54-generic
2025/08/29 16:38:10 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2025/08/29 16:38:10 [notice] 1#1: start worker processes
2025/08/29 16:38:10 [notice] 1#1: start worker process 36
root@srv628149:~#
```

kubectl expose pod nginx-pod --type=NodePort –port=9090 :
service/nginx-pod exposed



```
root@srv628149:~# kubectl expose pod nginx-pod --type=NodePort --port=9090
service/nginx-pod exposed
root@srv628149:~#
```

kubectl delete pod nginx-pod :
pod "nginx-pod" deleted



```
root@srv628149:~# kubectl delete pod nginx-pod
pod "nginx-pod" deleted
root@srv628149:~# kubectl get pods
No resources found in default namespace.
root@srv628149:~#
```

Q. What happens when you delete a pod? Test it by deleting the nginx pod and observe the behavior of the cluster.

A : Kubernetes again starts the pod, when it is deleted but if you use restart Never it will not get started.

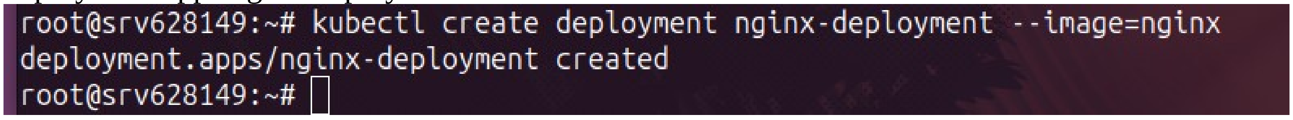
3. Exercise 3: Working with Deployments

commands used:

kubectl create deployment nginx-deployment –image=nginx

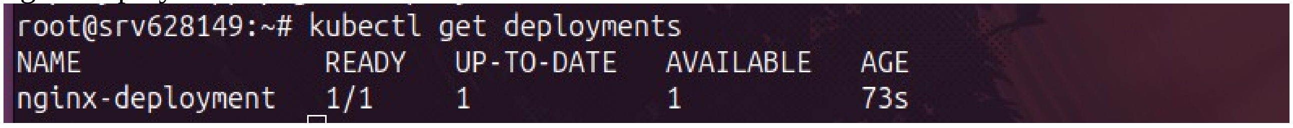
kubectl get deployments
kubectl scale deployment nginx-deployment –replicas=3
kubectl get pods
kubectl set image deployment/nginx-deployment nginx=nginx:1.25.0
kubectl rollout history deployment/nginx-deployment

kubectl create deployment nginx-deployment –image=nginx :
deployment.apps/nginx-deployment created



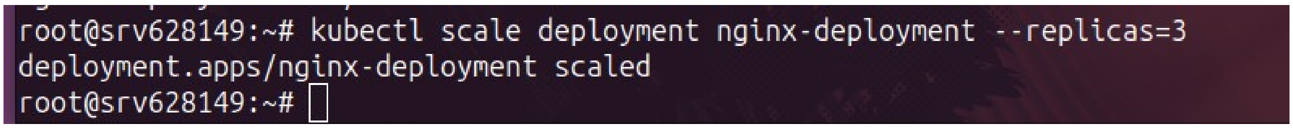
```
root@srv628149:~# kubectl create deployment nginx-deployment --image=nginx
deployment.apps/nginx-deployment created
root@srv628149:~#
```

kubectl get deployments :
```
NAME              READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment   1/1     1            1           73s
```



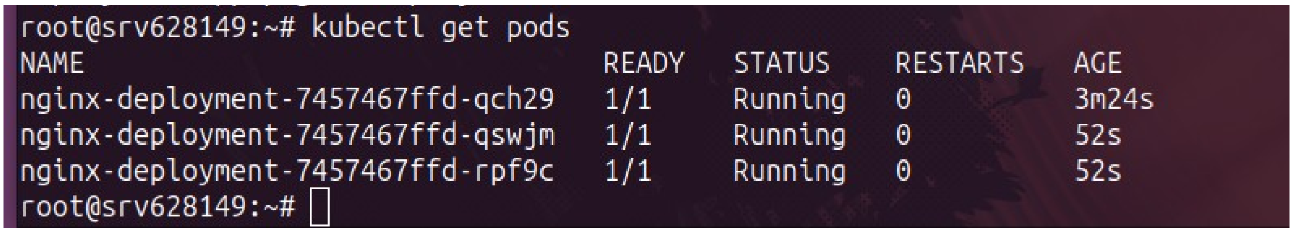
```
root@srv628149:~# kubectl get deployments
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment   1/1     1            1           73s
```

kubectl scale deployment nginx-deployment –replicas=3 :
deployment.apps/nginx-deployment scaled

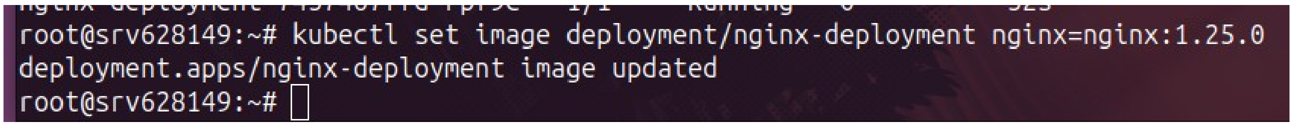

```
root@srv628149:~# kubectl scale deployment nginx-deployment --replicas=3
deployment.apps/nginx-deployment scaled
root@srv628149:~#
```

kubectl get pods :

```
NAME                           READY   STATUS    RESTARTS   AGE
nginx-deployment-7457467ffd-qch29   1/1     Running   0          3m24s
nginx-deployment-7457467ffd-qswjm   1/1     Running   0          52s
nginx-deployment-7457467ffd-rpf9c   1/1     Running   0          52s
```

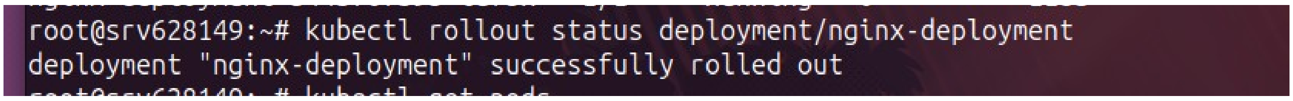

```
root@srv628149:~# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-7457467ffd-qch29   1/1     Running   0          3m24s
nginx-deployment-7457467ffd-qswjm   1/1     Running   0          52s
nginx-deployment-7457467ffd-rpf9c   1/1     Running   0          52s
root@srv628149:~#
```

kubectl set image deployment/nginx-deployment nginx=nginx:1.25.0 :
deployment.apps/nginx-deployment image updated


```
root@srv628149:~# kubectl set image deployment/nginx-deployment nginx=nginx:1.25.0
deployment.apps/nginx-deployment image updated
root@srv628149:~#
```

kubectl rollout history deployment/nginx-deployment :

```
deployment.apps/nginx-deployment
REVISION  CHANGE-CAUSE
1         <none>
2         <none>
```


```
root@srv628149:~# kubectl rollout status deployment/nginx-deployment
deployment "nginx-deployment" successfully rolled out
```

Q. What does deployment rollout history show? How would you roll back a deployment?
A : Deployment rollout history show the history of deployments roll back to previous version : kubectl rollout undo deployment/<name>

4. Exercise 4: Services and Networking

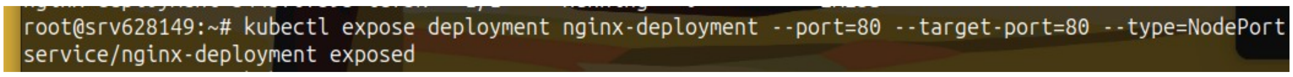
commands used:

kubectl expose deployment nginx-deployment --port=80 --target-port=80 –type=NodePort
kubectl get svc
kubectl port-forward svc/nginx-deployment 9999:80

&

curl http://localhost:9999

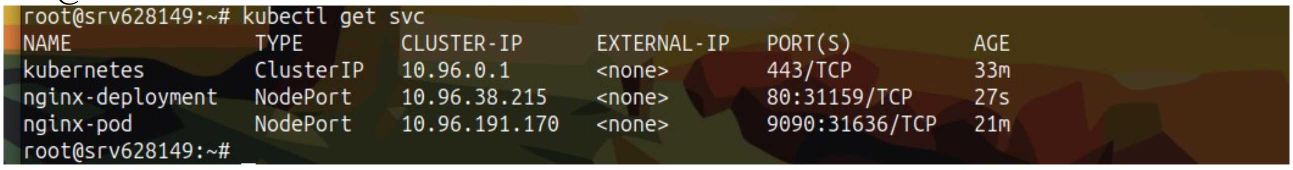
kubectl expose deployment nginx-deployment --port=80 --target-port=80 –type=NodePort :
service/nginx-deployment exposed



```
[illegible]
root@srv628149:~# kubectl expose deployment nginx-deployment --port=80 --target-port=80 --type=NodePort
service/nginx-deployment exposed
```

kubectl get svc :

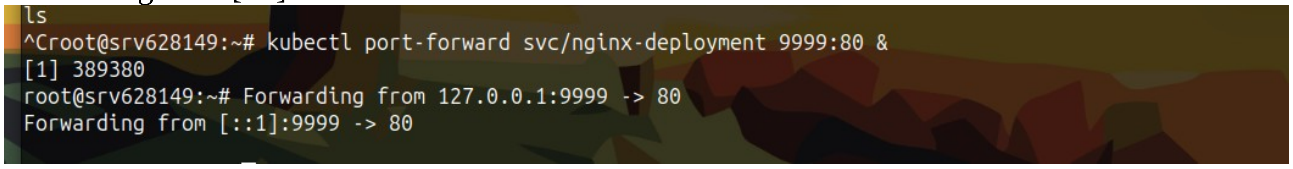
```
NAME             TYPE       CLUSTER-IP     EXTERNAL-IP  PORT(S)        AGE
kubernetes       ClusterIP  10.96.0.1      <none>       443/TCP        33m
nginx-deployment  NodePort  10.96.38.215   <none>       80:31159/TCP   27s
nginx-pod        NodePort   10.96.191.170  <none>       9090:31636/TCP  21m
root@srv628149:~#
```



```
root@srv628149:~# kubectl get svc
NAME               TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
kubernetes         ClusterIP   10.96.0.1       <none>        443/TCP          33m
nginx-deployment   NodePort    10.96.38.215    <none>        80:31159/TCP     27s
nginx-pod          NodePort    10.96.191.170   <none>        9090:31636/TCP   21m
root@srv628149:~# _
```

kubectl port-forward svc/nginx-deployment 9999:80 & :

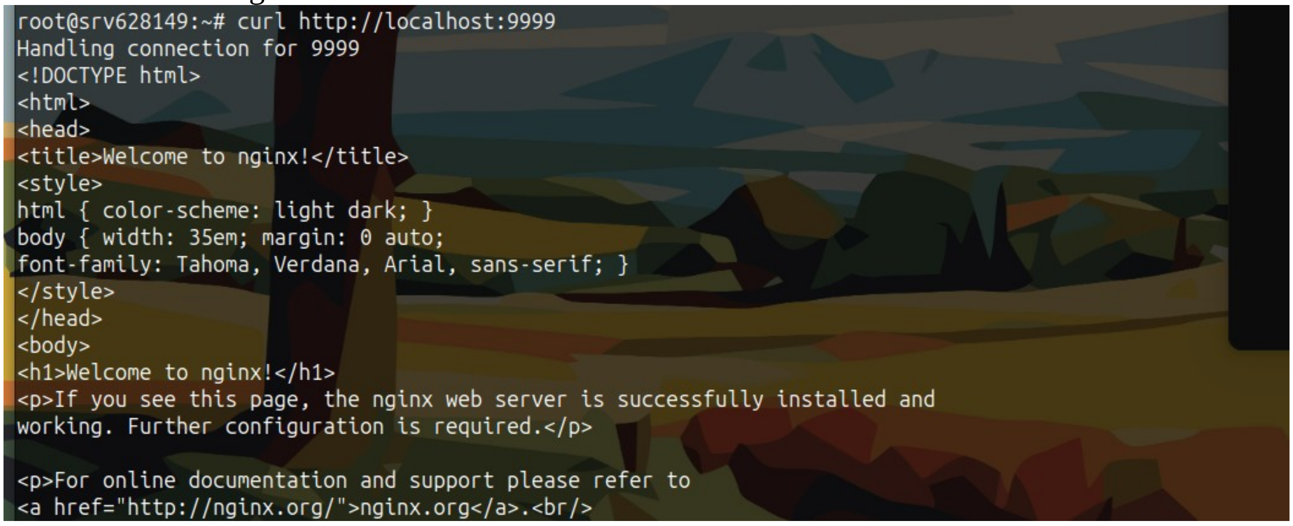
```
[1] 389380
root@srv628149:~# Forwarding from 127.0.0.1:9999 -> 80
Forwarding from [::1]:9999 -> 80
```



```
ls
^Croot@srv628149:~# kubectl port-forward svc/nginx-deployment 9999:80 &
[1] 389380
root@srv628149:~# Forwarding from 127.0.0.1:9999 -> 80
Forwarding from [::1]:9999 -> 80
```

curl http://localhost:9999 :

```
Handling connection for 9999
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
```



```
root@srv628149:~# curl http://localhost:9999
Handling connection for 9999
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
```

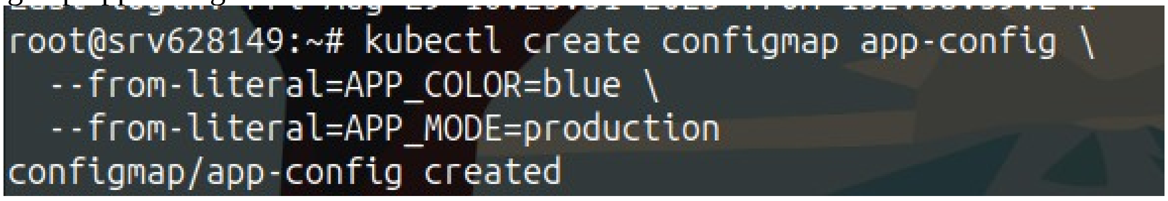
Q. What is the difference between ClusterIP, NodePort, and LoadBalancer services?
When should you use each?
A : ClusterIP is internal to the cluster, NodePort exposes the services externally, LoadBalancer maps the external load balancer.

5. Exercise 5: ConfigMaps and Secrets

commands used: **kubectl create configmap app-config \ --from-literal=APP_COLOR=blue \ --from-literal=APP_MODE=production**

kubectl get configmap app-config -o yaml
kubectl apply -f configmap-pod.yaml
kubectl exec -it configmap-pod -- env | grep APP_
kubectl create secret generic app-secret \ --from-literal=DB_USER=admin \ --from-literal=DB_PASS=Pa$$w0rd
kubectl apply -f secret-pod.yaml
kubectl exec -it secret-pod -- env | grep DB_

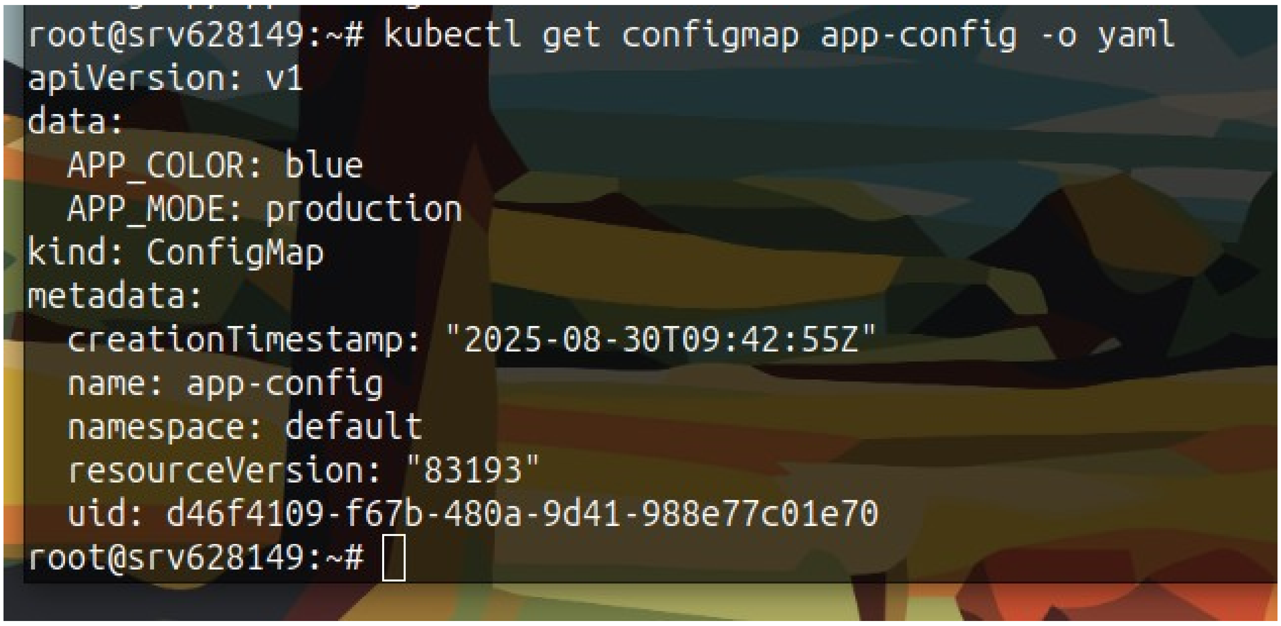
kubectl create configmap app-config \ --from-literal=APP_COLOR=blue \ --from-literal=APP_MODE=production :
configmap/app-config created



```
root@srv628149:~# kubectl create configmap app-config \
  --from-literal=APP_COLOR=blue \
  --from-literal=APP_MODE=production
configmap/app-config created
```

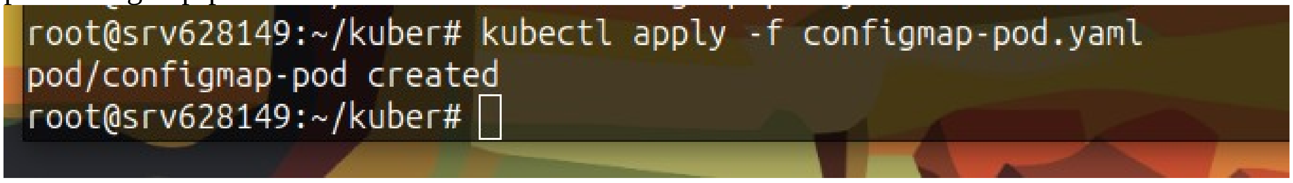
kubectl get configmap app-config -o yaml :

```
apiVersion: v1
data:
  APP_COLOR: blue
  APP_MODE: production
kind: ConfigMap
metadata:
  creationTimestamp: "2025-08-30T09:42:55Z"
  name: app-config
  namespace: default
  resourceVersion: "83193"
  uid: d46f4109-f67b-480a-9d41-988e77c01e70
```



```
root@srv628149:~# kubectl get configmap app-config -o yaml
apiVersion: v1
data:
  APP_COLOR: blue
  APP_MODE: production
kind: ConfigMap
metadata:
  creationTimestamp: "2025-08-30T09:42:55Z"
  name: app-config
  namespace: default
  resourceVersion: "83193"
  uid: d46f4109-f67b-480a-9d41-988e77c01e70
root@srv628149:~#
```

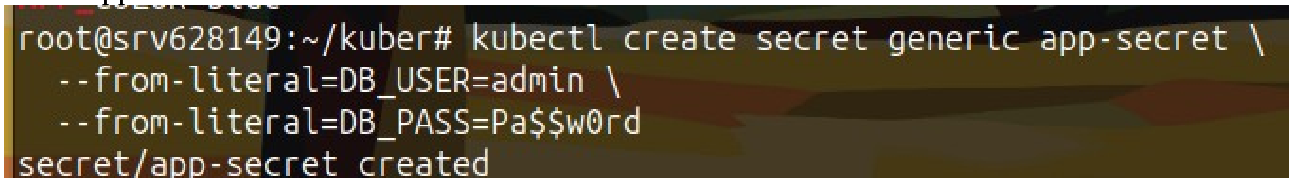
kubectl apply -f configmap-pod.yaml :
pod/configmap-pod created



```
root@srv628149:~/kuber# kubectl apply -f configmap-pod.yaml
pod/configmap-pod created
root@srv628149:~/kuber#
```

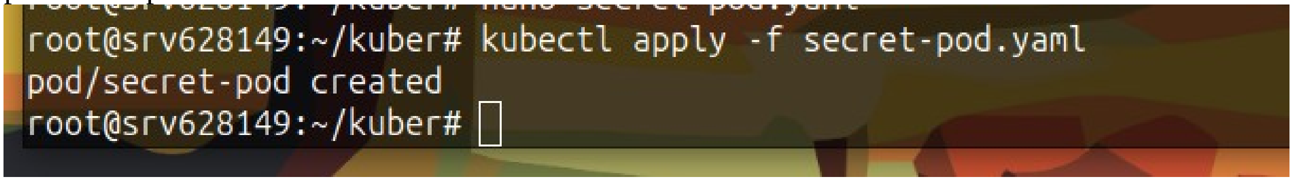
kubectl exec -it configmap-pod -- env | grep APP_ :
APP_MODE=production
APP_COLOR=blue

kubectl create secret generic app-secret \ --from-literal=DB_USER=admin \ --from-literal=DB_PASS=Pa$$w0rd :
secret/app-secret created



```
root@srv628149:~/kuber# kubectl create secret generic app-secret \
  --from-literal=DB_USER=admin \
  --from-literal=DB_PASS=Pa$$w0rd
secret/app-secret created
```

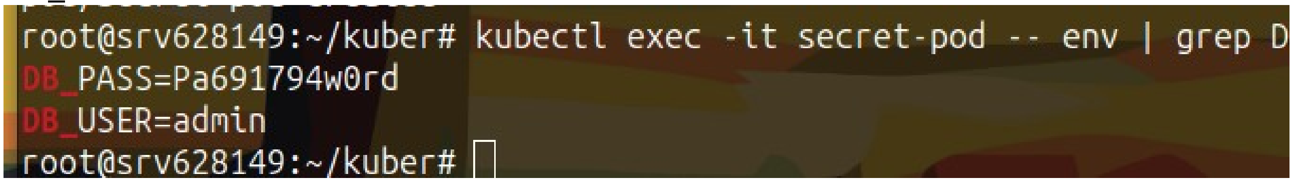
kubectl apply -f secret-pod.yaml :
pod/secret-pod created



```
root@srv628149:~/kuber# kubectl apply -f secret-pod.yaml
pod/secret-pod created
root@srv628149:~/kuber#
```

kubectl exec -it secret-pod -- env | grep DB_ :
DB_PASS=Pa691794w0rd

DB_USER=admin



```
root@srv628149:~/kuber# kubectl exec -it secret-pod -- env | grep D
DB_PASS=Pa691794w0rd
DB_USER=admin
root@srv628149:~/kuber# 
```

Q. How would you access the value of a ConfigMap or Secret within your application?
A : We will use environment variable to acces these values.

6. Exercise 6: Persistent Volumes (PVs) and Persistent Volume Claims (PVCs)

commands used:

```
kubectl apply -f pv.yaml
kubectl apply -f pvc.yaml
kubectl apply -f pod-pvc.yaml
kubectl exec -it pvc-pod – sh
kubectl delete pod pvc-pod
kubectl apply -f pod-pvc.yaml
kubectl exec -it pvc-pod -- cat
/mnt/storage/test.txt
```

pv.yaml :

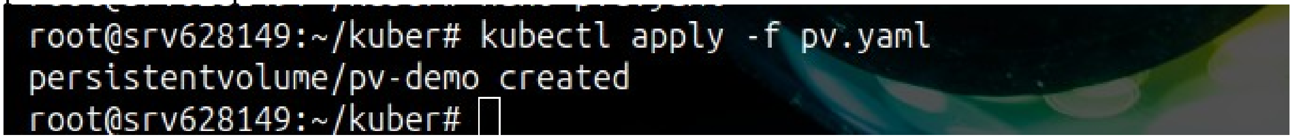
```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-demo
spec:
  capacity:
    storage: 1Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: /data/pv-demo
```

pvc.yaml :

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-demo
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 500Mi
```

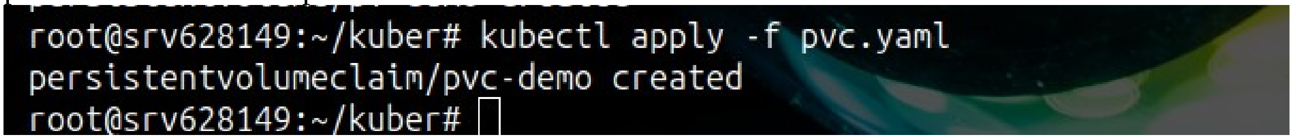
kubectl apply -f pv.yaml :

persistentvolume/pv-demo created



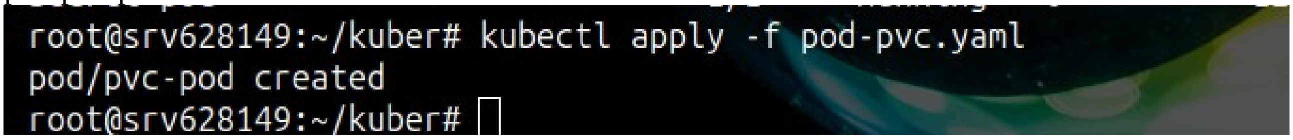
```
root@srv628149:~/kuber# kubectl apply -f pv.yaml
persistentvolume/pv-demo created
root@srv628149:~/kuber#
```

kubectl apply -f pvc.yaml :
persistentvolumeclaim/pvc-demo created



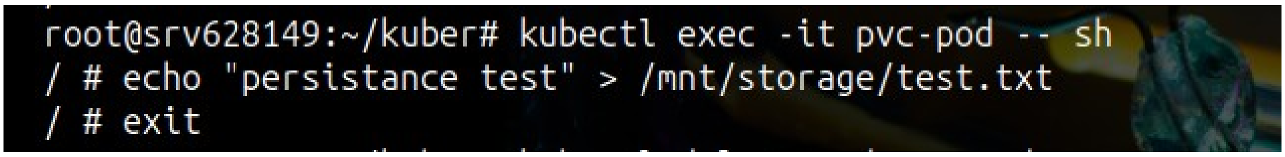
```
root@srv628149:~/kuber# kubectl apply -f pvc.yaml
persistentvolumeclaim/pvc-demo created
root@srv628149:~/kuber#
```

kubectl apply -f pod-pvc.yaml :
pod/pvc-pod created



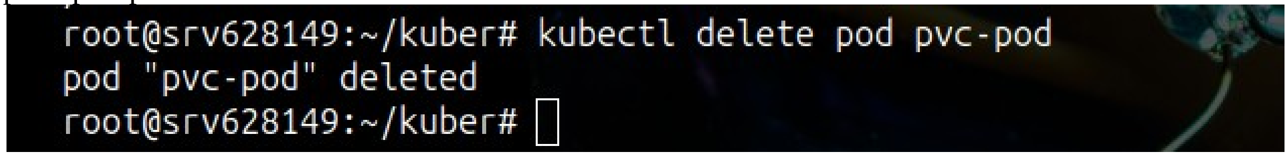
```
root@srv628149:~/kuber# kubectl apply -f pod-pvc.yaml
pod/pvc-pod created
root@srv628149:~/kuber#
```

kubectl exec -it pvc-pod – sh :
/ # echo "persistance test" > /mnt/storage/test.txt
/ # exit



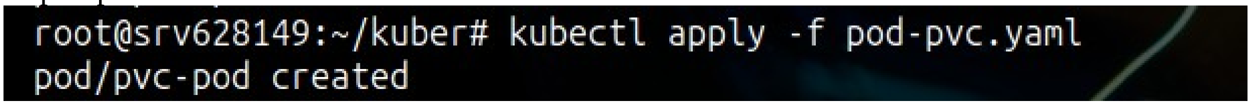
```
root@srv628149:~/kuber# kubectl exec -it pvc-pod -- sh
/ # echo "persistance test" > /mnt/storage/test.txt
/ # exit
```

kubectl delete pod pvc-pod :
pod "pvc-pod" deleted



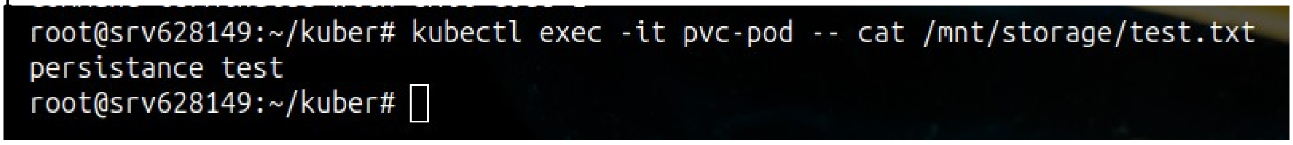
```
root@srv628149:~/kuber# kubectl delete pod pvc-pod
pod "pvc-pod" deleted
root@srv628149:~/kuber#
```

kubectl apply -f pod-pvc.yaml :
pod/pvc-pod created



```
root@srv628149:~/kuber# kubectl apply -f pod-pvc.yaml
pod/pvc-pod created
```

kubectl exec -it pvc-pod -- cat /mnt/storage/test.txt:
persistance test



```
root@srv628149:~/kuber# kubectl exec -it pvc-pod -- cat /mnt/storage/test.txt
persistance test
root@srv628149:~/kuber#
```

Q. What happens if the PVC is deleted? Does the underlying Persistent Volume get deleted as well?
A : When PVC is deleted it releases the claim on the volume, the deletion of the volume depends on the set policy.

7. Exercise 7: StatefulSets

commands used :

```
kubectl apply -f mysql-service.yaml
kubectl apply -f mysql-statefulset.yaml
kubectl get statefulsets
kubectl get pods -l app=mysql -o wide
```

mysql-service.yaml:

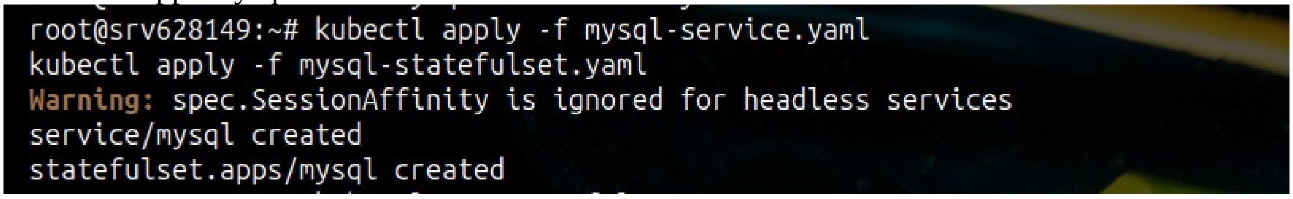
```
apiVersion: v1
kind: Service
metadata:
  name: mysql
  labels:
    app: mysql
spec:
  ports:
  - port: 3306
    name: mysql
  clusterIP: None
  selector:
    app: mysql
```

mysql-statefulset.yaml:

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  serviceName: "mysql"
  replicas: 2
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
      - name: mysql
        image: mysql:8.0
        ports:
        - containerPort: 3306
          name: mysql
        env:
        - name: MYSQL_ROOT_PASSWORD
          value: rootpassword
        volumeMounts:
        - name: mysql-persistent-storage
          mountPath: /var/lib/mysql
  volumeClaimTemplates:
  - metadata:
      name: mysql-persistent-storage
    spec:
      accessModes: ["ReadWriteOnce"]
      resources:
        requests:
          storage: 1Gi
```

kubectl apply -f mysql-service.yaml
kubectl apply -f mysql-statefulset.yaml

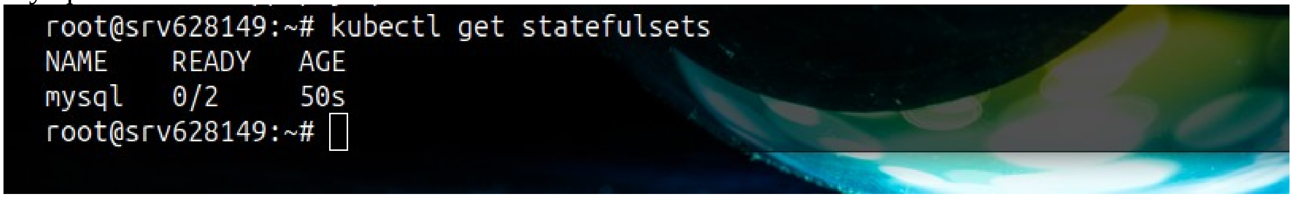
```
Warning: spec.SessionAffinity is ignored for headless services
service/mysql created
statefulset.apps/mysql created
```



```
root@srv628149:~# kubectl apply -f mysql-service.yaml
kubectl apply -f mysql-statefulset.yaml
Warning: spec.SessionAffinity is ignored for headless services
service/mysql created
statefulset.apps/mysql created
```

kubectl get statefulsets :

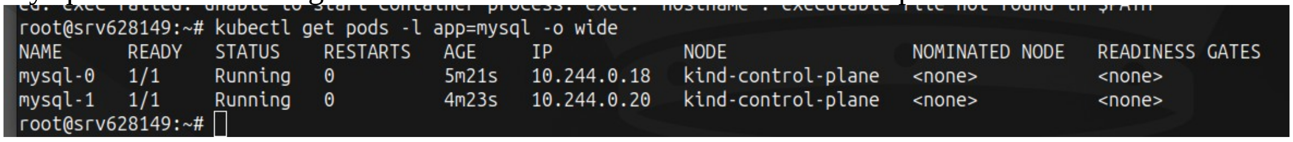
```
NAME    READY   AGE
mysql   0/2     50s
```



```
root@srv628149:~# kubectl get statefulsets
NAME     READY    AGE
mysql    0/2      50s
root@srv628149:~#
```

kubectl get pods -l app=mysql -o wide :

```
NAME      READY   STATUS    RESTARTS   AGE     IP            NODE                 NOMINATED
NODE   READINESS GATES
mysql-0   1/1     Running   0          5m21s   10.244.0.18   kind-control-plane   <none>           <none>
mysql-1   1/1     Running   0          4m23s   10.244.0.20   kind-control-plane   <none>           <none>
```



```
root@srv628149:~# kubectl get pods -l app=mysql -o wide
NAME      READY   STATUS    RESTARTS   AGE     IP            NODE                 NOMINATED NODE   READINESS GATES
mysql-0   1/1     Running   0          5m21s   10.244.0.18   kind-control-plane   <none>           <none>
mysql-1   1/1     Running   0          4m23s   10.244.0.20   kind-control-plane   <none>           <none>
root@srv628149:~#
```

Q. What are the key differences between StatefulSets and Deployments? When would you use a StatefulSet instead of a Deployment?
A : StatefulSets have a unique network identity but deployments do not

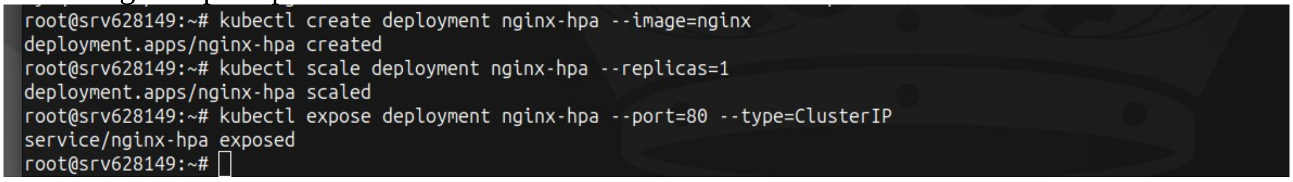
8. Exercise 8: Horizontal Pod Autoscaling (HPA)

commands used : **kubectl create deployment nginx-hpa –image=nginx**
kubectl scale deployment nginx-hpa –replicas=1
kubectl expose deployment nginx-hpa --port=80 –type=ClusterIP
kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml
kubectl autoscale deployment nginx-hpa --cpu-percent=50 --min=1 –max=5

kubectl create deployment nginx-hpa –image=nginx :
deployment.apps/nginx-hpa created

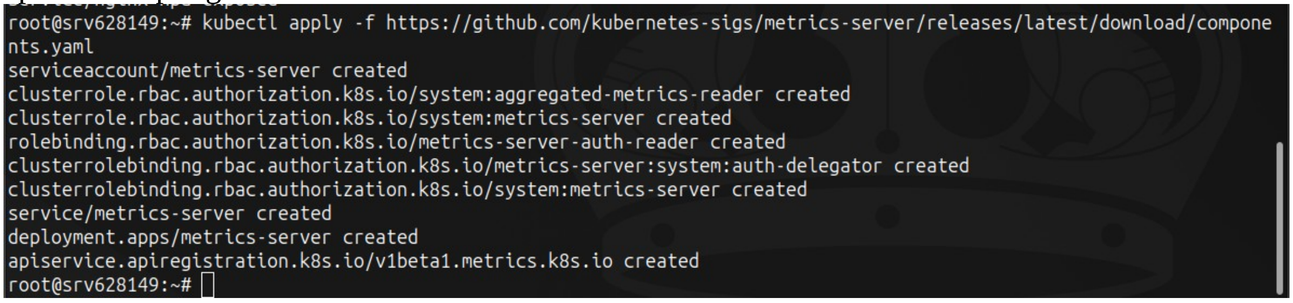
kubectl scale deployment nginx-hpa –replicas=1 :
deployment.apps/nginx-hpa scaled

kubectl expose deployment nginx-hpa --port=80 –type=ClusterIP :
service/nginx-hpa exposed



```
root@srv628149:~# kubectl create deployment nginx-hpa --image=nginx
deployment.apps/nginx-hpa created
root@srv628149:~# kubectl scale deployment nginx-hpa --replicas=1
deployment.apps/nginx-hpa scaled
root@srv628149:~# kubectl expose deployment nginx-hpa --port=80 --type=ClusterIP
service/nginx-hpa exposed
root@srv628149:~# 
```

kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml :
serviceaccount/metrics-server created
clusterrole.rbac.authorization.k8s.io/system:aggregated-metrics-reader created
clusterrole.rbac.authorization.k8s.io/system:metrics-server created
rolebinding.rbac.authorization.k8s.io/metrics-server-auth-reader created
clusterrolebinding.rbac.authorization.k8s.io/metrics-server:system:auth-delegator created
clusterrolebinding.rbac.authorization.k8s.io/system:metrics-server created
service/metrics-server created
deployment.apps/metrics-server created
apiservice.apiregistration.k8s.io/v1beta1.metrics.k8s.io created



```
root@srv628149:~# kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/compone
nts.yaml
serviceaccount/metrics-server created
clusterrole.rbac.authorization.k8s.io/system:aggregated-metrics-reader created
clusterrole.rbac.authorization.k8s.io/system:metrics-server created
rolebinding.rbac.authorization.k8s.io/metrics-server-auth-reader created
clusterrolebinding.rbac.authorization.k8s.io/metrics-server:system:auth-delegator created
clusterrolebinding.rbac.authorization.k8s.io/system:metrics-server created
service/metrics-server created
deployment.apps/metrics-server created
apiservice.apiregistration.k8s.io/v1beta1.metrics.k8s.io created
root@srv628149:~# 
```

kubectl autoscale deployment nginx-hpa --cpu-percent=50 --min=1 –max=5 :
horizontalpodautoscaler.autoscaling/nginx-hpa autoscaled

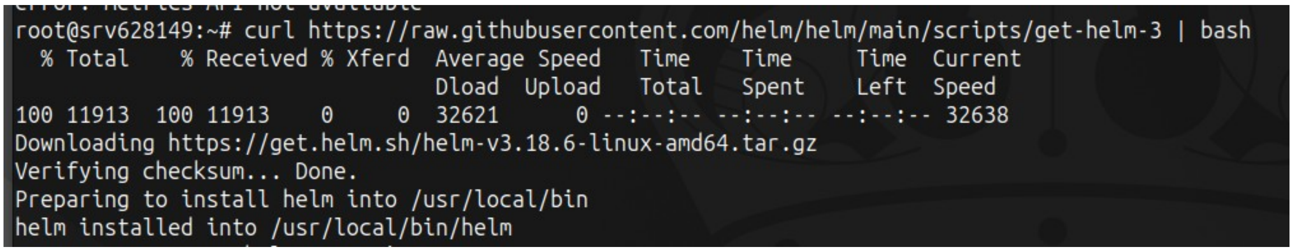
Q.How does the HPA decide when to scale? What metrics are used for scaling?
A : It check the provided threshold and matrics

9. Exercise 9: Helm Basics

commands used: **kubectl autoscale deployment nginx-hpa --cpu-percent=50 --min=1 –max=5**
helm version
helm repo add bitnami https://charts.bitnami.com/bitnami
helm repo update
helm install my-nginx bitnami/nginx
helm list

curl https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 | bash :

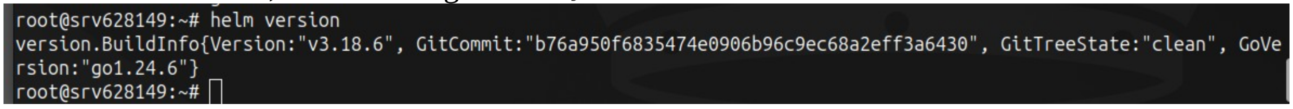
```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 11913  100 11913    0     0  32621      0 --:--:-- --:--:-- --:--:-- 32638
Downloading https://get.helm.sh/helm-v3.18.6-linux-amd64.tar.gz
Verifying checksum... Done.
Preparing to install helm into /usr/local/bin
helm installed into /usr/local/bin/helm
```



```
root@srv628149:~# curl https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 | bash
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 11913  100 11913    0     0  32621      0 --:--:-- --:--:-- --:--:-- 32638
Downloading https://get.helm.sh/helm-v3.18.6-linux-amd64.tar.gz
Verifying checksum... Done.
Preparing to install helm into /usr/local/bin
helm installed into /usr/local/bin/helm
```

helm version :

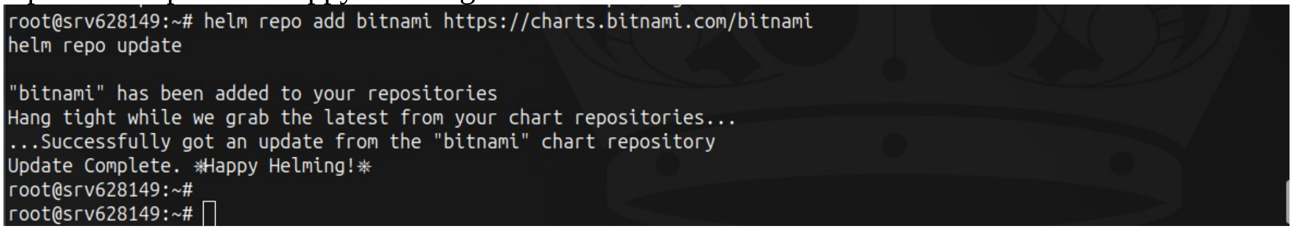
```
version.BuildInfo{Version:"v3.18.6", GitCommit:"b76a950f6835474e0906b96c9ec68a2eff3a6430", GitTreeState:"clean", GoVersion:"go1.24.6"}
```



```
root@srv628149:~# helm version
version.BuildInfo{Version:"v3.18.6", GitCommit:"b76a950f6835474e0906b96c9ec68a2eff3a6430", GitTreeState:"clean", GoVe
rsion:"go1.24.6"}
root@srv628149:~#
```

helm repo add bitnami https://charts.bitnami.com/bitnami
helm repo update :

```
"bitnami" has been added to your repositories
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "bitnami" chart repository
Update Complete. ⎈Happy Helming!⎈
```

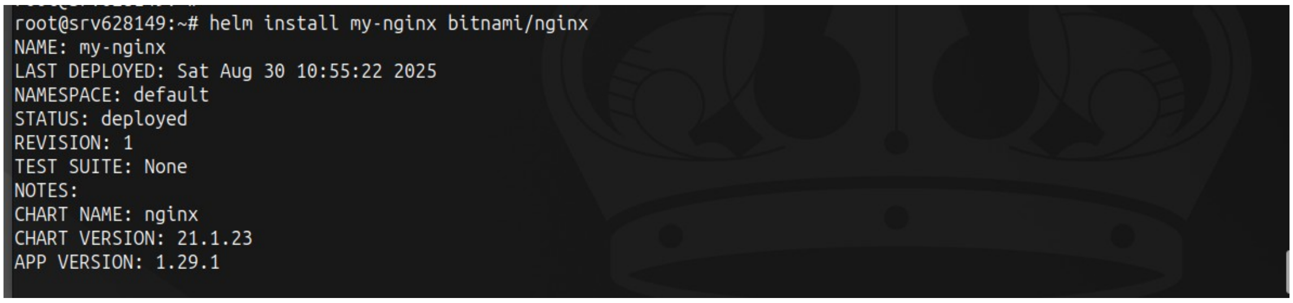


```
root@srv628149:~# helm repo add bitnami https://charts.bitnami.com/bitnami
helm repo update

"bitnami" has been added to your repositories
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "bitnami" chart repository
Update Complete. ⎈Happy Helming!⎈
root@srv628149:~#
root@srv628149:~#
```

helm install my-nginx bitnami/nginx :

```
NAME: my-nginx
LAST DEPLOYED: Sat Aug 30 10:55:22 2025
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
```

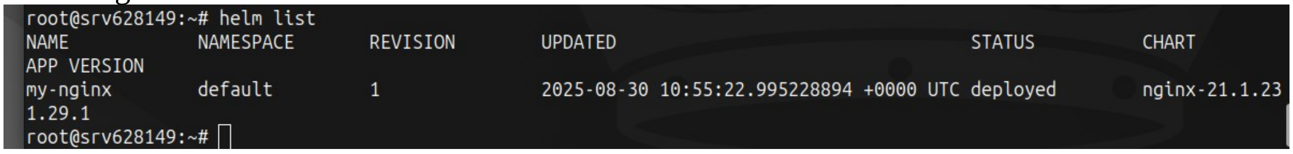
NOTES:
CHART NAME: nginx
CHART VERSION: 21.1.23
APP VERSION: 1.29.1



```
root@srv628149:~# helm install my-nginx bitnami/nginx
NAME: my-nginx
LAST DEPLOYED: Sat Aug 30 10:55:22 2025
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
CHART NAME: nginx
CHART VERSION: 21.1.23
APP VERSION: 1.29.1
```

helm list :

```
NAME          NAMESPACE          REVISION     UPDATED                                          STATUS
         CHART        APP VERSION
my-nginx       default           1          2025-08-30 10:55:22.995228894 +0000 UTC         deployed
         nginx-21.1.231.29.1
```



```
root@srv628149:~# helm list
NAME            NAMESPACE       REVISION        UPDATED                                 STATUS          CHART
APP VERSION
my-nginx        default         1               2025-08-30 10:55:22.995228894 +0000 UTC deployed        nginx-21.1.23
1.29.1
root@srv628149:~#
```

Q. What advantages does using Helm offer over manually managing Kubernetes resources with kubectl?
A : helm has build in lifecycle management, release management, complex parameterization, templating etc.

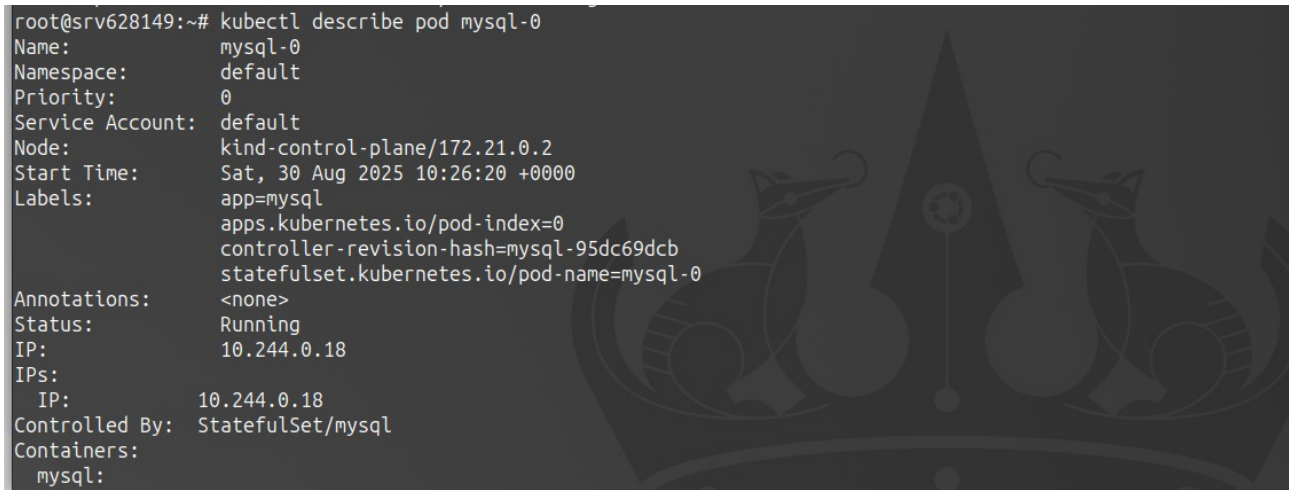
10. Exercise 10: Debugging and Troubleshooting

commands used:	**kubectl describe pod mysql-0**
	kubectl get nodes
	kubectl get pods
	kubectl get events
	kubectl logs mysql-0
	kubectl logs deployments/nginx-deployment

kubectl describe pod mysql-0 :

```
Name:             mysql-0
Namespace:        default
Priority:         0
Service Account:  default
Node:             kind-control-plane/172.21.0.2
Start Time:       Sat, 30 Aug 2025 10:26:20 +0000
Labels:           app=mysql
                  apps.kubernetes.io/pod-index=0
                  controller-revision-hash=mysql-95dc69dcb
                  statefulset.kubernetes.io/pod-name=mysql-0
Annotations:      <none>
Status:           Running
IP:               10.244.0.18
```

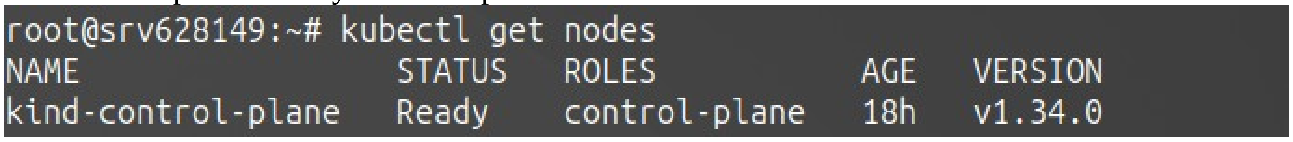
IPs:
  IP: 10.244.0.18



```
root@srv628149:~# kubectl describe pod mysql-0
Name:             mysql-0
Namespace:        default
Priority:         0
Service Account:  default
Node:             kind-control-plane/172.21.0.2
Start Time:       Sat, 30 Aug 2025 10:26:20 +0000
Labels:           app=mysql
                  apps.kubernetes.io/pod-index=0
                  controller-revision-hash=mysql-95dc69dcb
                  statefulset.kubernetes.io/pod-name=mysql-0
Annotations:      <none>
Status:           Running
IP:               10.244.0.18
IPs:
  IP:           10.244.0.18
Controlled By:  StatefulSet/mysql
Containers:
  mysql:
```

kubectl get nodes :

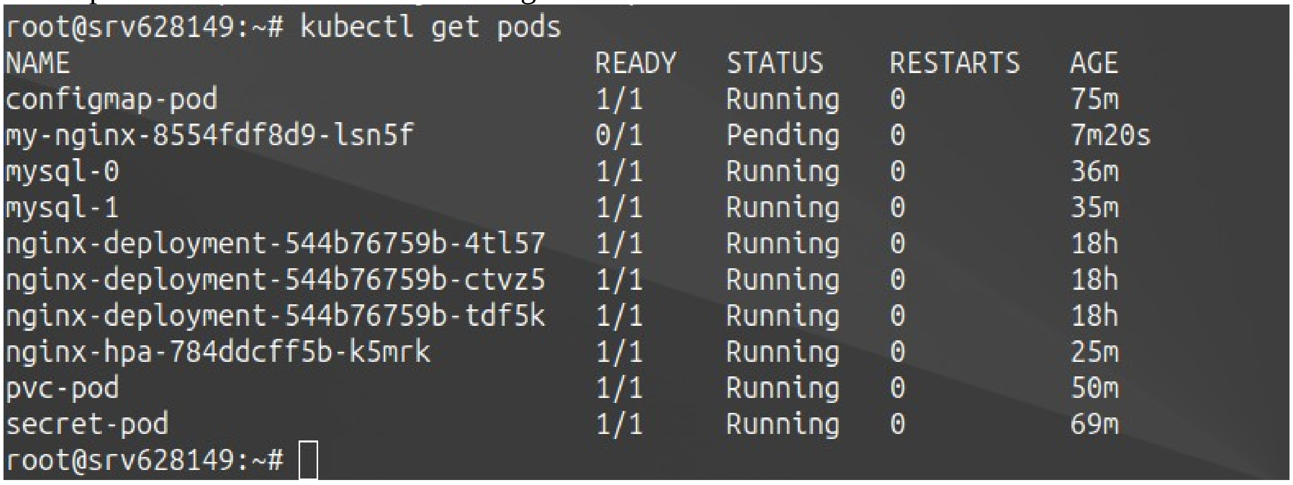
```
NAME                 STATUS   ROLES           AGE   VERSION
kind-control-plane   Ready    control-plane   18h   v1.34.0
```



```
root@srv628149:~# kubectl get nodes
NAME                 STATUS   ROLES           AGE   VERSION
kind-control-plane   Ready    control-plane   18h   v1.34.0
```

kubectl get pods :

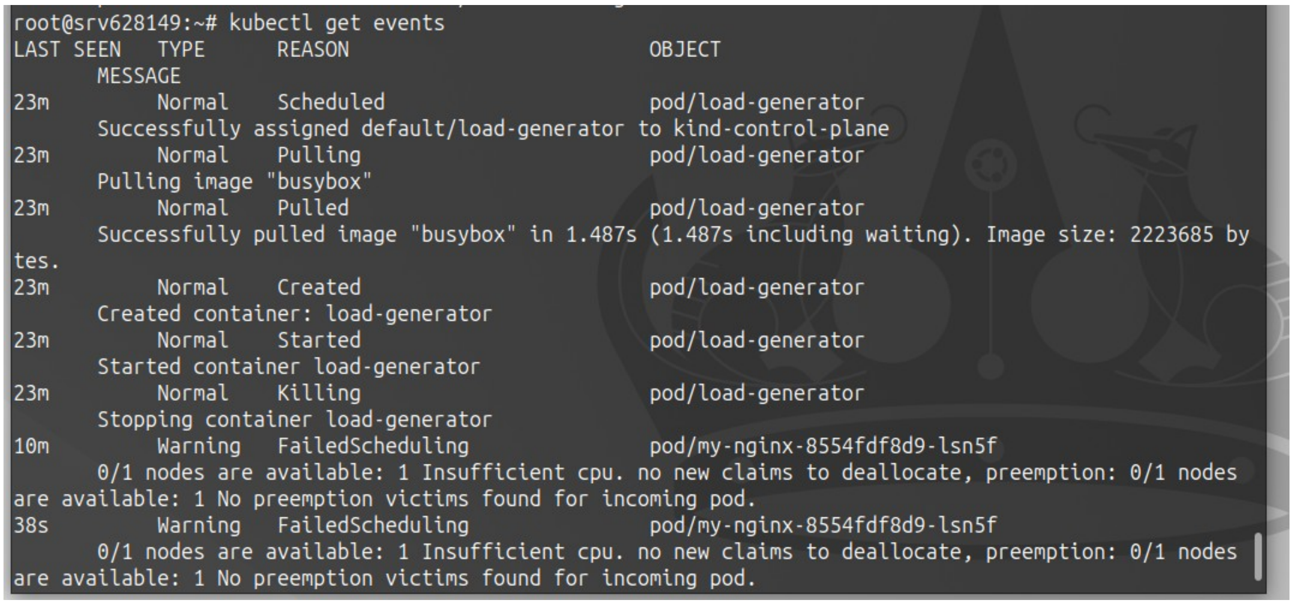
```
NAME                               READY   STATUS    RESTARTS   AGE
configmap-pod                      1/1     Running   0          75m
my-nginx-8554fdf8d9-lsn5f          0/1     Pending   0          7m20s
mysql-0                            1/1     Running   0          36m
mysql-1                            1/1     Running   0          35m
nginx-deployment-544b76759b-4tl57  1/1     Running   0          18h
nginx-deployment-544b76759b-ctvz5  1/1     Running   0          18h
nginx-deployment-544b76759b-tdf5k  1/1     Running   0          18h
nginx-hpa-784ddcff5b-k5mrk         1/1     Running   0          25m
pvc-pod                            1/1     Running   0          50m
secret-pod                         1/1     Running   0          69m
```



```
root@srv628149:~# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
configmap-pod                       1/1     Running   0          75m
my-nginx-8554fdf8d9-lsn5f           0/1     Pending   0          7m20s
mysql-0                             1/1     Running   0          36m
mysql-1                             1/1     Running   0          35m
nginx-deployment-544b76759b-4tl57   1/1     Running   0          18h
nginx-deployment-544b76759b-ctvz5   1/1     Running   0          18h
nginx-deployment-544b76759b-tdf5k   1/1     Running   0          18h
nginx-hpa-784ddcff5b-k5mrk          1/1     Running   0          25m
pvc-pod                             1/1     Running   0          50m
secret-pod                          1/1     Running   0          69m
root@srv628149:~#
```

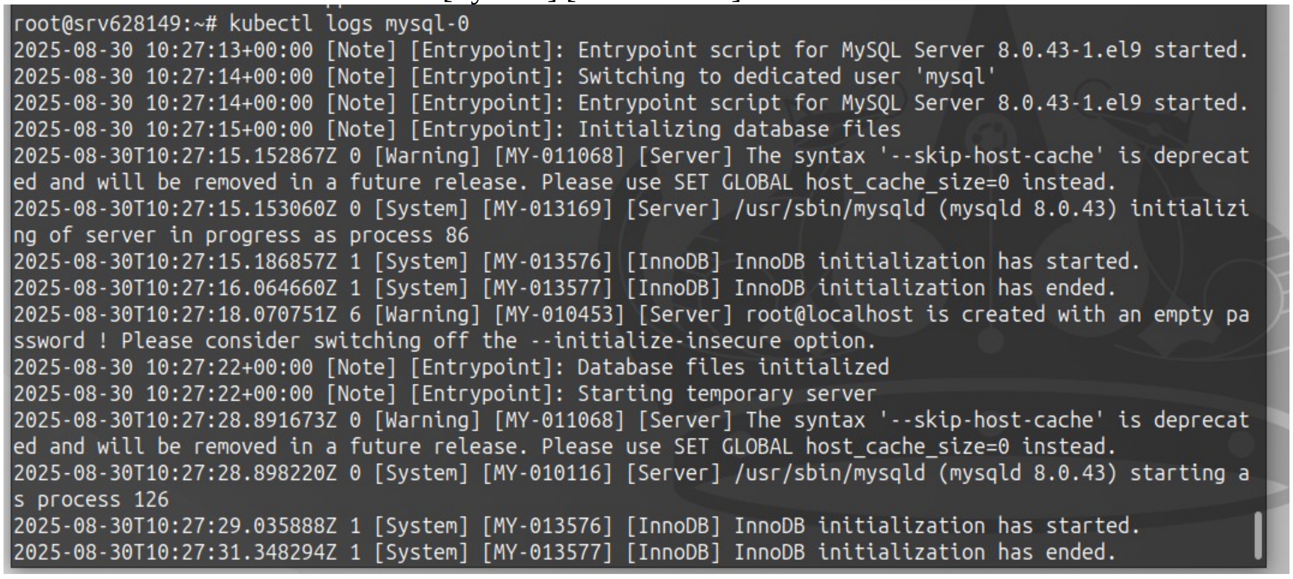
kubectl get events :

LAST SEEN   TYPE   REASON   OBJECT
MESSAGE
23m   Normal   Scheduled   pod/load-generator
Successfully assigned default/load-generator to kind-control-plane
23m   Normal   Pulling   pod/load-generator   Pulling image
"busybox"
23m   Normal   Pulled



```
root@srv628149:~# kubectl get events
LAST SEEN   TYPE      REASON                    OBJECT
      MESSAGE
23m         Normal    Scheduled                 pod/load-generator
      Successfully assigned default/load-generator to kind-control-plane
23m         Normal    Pulling                   pod/load-generator
      Pulling image "busybox"
23m         Normal    Pulled                    pod/load-generator
      Successfully pulled image "busybox" in 1.487s (1.487s including waiting). Image size: 2223685 by
tes.
23m         Normal    Created                   pod/load-generator
      Created container: load-generator
23m         Normal    Started                   pod/load-generator
      Started container load-generator
23m         Normal    Killing                   pod/load-generator
      Stopping container load-generator
10m         Warning   FailedScheduling          pod/my-nginx-8554fdf8d9-lsn5f
      0/1 nodes are available: 1 Insufficient cpu. no new claims to deallocate, preemption: 0/1 nodes
are available: 1 No preemption victims found for incoming pod.
38s         Warning   FailedScheduling          pod/my-nginx-8554fdf8d9-lsn5f
      0/1 nodes are available: 1 Insufficient cpu. no new claims to deallocate, preemption: 0/1 nodes
are available: 1 No preemption victims found for incoming pod.
```

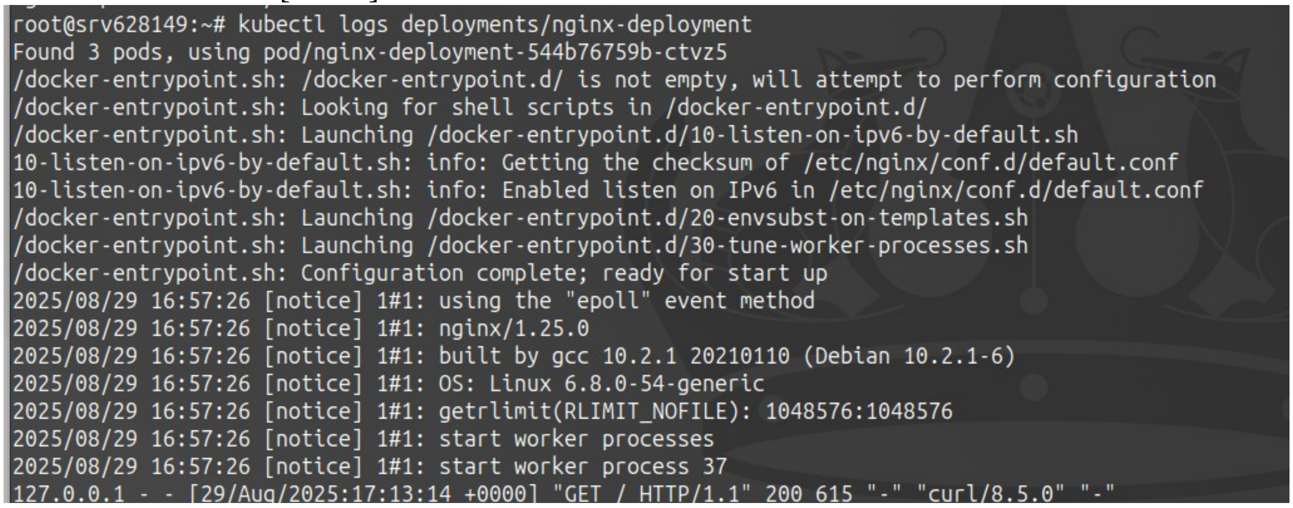
kubectl logs mysql-0 :
2025-08-30 10:27:13+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.43-1.el9 started.
2025-08-30 10:27:14+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
2025-08-30 10:27:14+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.43-1.el9 started.
2025-08-30 10:27:15+00:00 [Note] [Entrypoint]: Initializing database files
2025-08-30T10:27:15.152867Z 0 [Warning] [MY-011068] [Server] The syntax '--skip-host-cache' is deprecated and will be removed in a future release. Please use SET GLOBAL host_cache_size=0 instead.
2025-08-30T10:27:15.153060Z 0 [System] [MY-013169] [Server] /usr/sbin/mysqld (mysqld 8.0.43) initializing of server in progress as process 86
2025-08-30T10:27:15.186857Z 1 [System] [MY-013576]



```
root@srv628149:~# kubectl logs mysql-0
2025-08-30 10:27:13+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.43-1.el9 started.
2025-08-30 10:27:14+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
2025-08-30 10:27:14+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.43-1.el9 started.
2025-08-30 10:27:15+00:00 [Note] [Entrypoint]: Initializing database files
2025-08-30T10:27:15.152867Z 0 [Warning] [MY-011068] [Server] The syntax '--skip-host-cache' is deprecat
ed and will be removed in a future release. Please use SET GLOBAL host_cache_size=0 instead.
2025-08-30T10:27:15.153060Z 0 [System] [MY-013169] [Server] /usr/sbin/mysqld (mysqld 8.0.43) initializi
ng of server in progress as process 86
2025-08-30T10:27:15.186857Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
2025-08-30T10:27:16.064660Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
2025-08-30T10:27:18.070751Z 6 [Warning] [MY-010453] [Server] root@localhost is created with an empty pa
ssword ! Please consider switching off the --initialize-insecure option.
2025-08-30 10:27:22+00:00 [Note] [Entrypoint]: Database files initialized
2025-08-30 10:27:22+00:00 [Note] [Entrypoint]: Starting temporary server
2025-08-30T10:27:28.891673Z 0 [Warning] [MY-011068] [Server] The syntax '--skip-host-cache' is deprecat
ed and will be removed in a future release. Please use SET GLOBAL host_cache_size=0 instead.
2025-08-30T10:27:28.898220Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (mysqld 8.0.43) starting a
s process 126
2025-08-30T10:27:29.035888Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
2025-08-30T10:27:31.348294Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
```

kubectl logs deployments/nginx-deployment :

```
Found 3 pods, using pod/nginx-deployment-544b76759b-ctvz5
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2025/08/29 16:57:26 [notice] 1#1: using the "epoll" event method
2025/08/29 16:57:26 [notice] 1#1: nginx/1.25.0
2025/08/29 16:57:26 [notice] 1#1: built by gcc 10.2.1 20210110 (Debian 10.2.1-6)
2025/08/29 16:57:26 [notice] 1#1: OS: Li
```



```
root@srv628149:~# kubectl logs deployments/nginx-deployment
Found 3 pods, using pod/nginx-deployment-544b76759b-ctvz5
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2025/08/29 16:57:26 [notice] 1#1: using the "epoll" event method
2025/08/29 16:57:26 [notice] 1#1: nginx/1.25.0
2025/08/29 16:57:26 [notice] 1#1: built by gcc 10.2.1 20210110 (Debian 10.2.1-6)
2025/08/29 16:57:26 [notice] 1#1: OS: Linux 6.8.0-54-generic
2025/08/29 16:57:26 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2025/08/29 16:57:26 [notice] 1#1: start worker processes
2025/08/29 16:57:26 [notice] 1#1: start worker process 37
127.0.0.1 - - [29/Aug/2025:17:13:14 +0000] "GET / HTTP/1.1" 200 615 "-" "curl/8.5.0" "-"
```

Q.What are some common reasons for a pod being in a CrashLoopBackOff state?
A : Some common reasons for a pod being in a CrashLoopBackOff state can be application issue, resource issue, configuration issue, port conflict etc.

Capstone Project:

commands used:

```
kind load docker-image python-webserver
kubectl apply -f k8s-capstone.yaml
kubectl -n capstone get deploy,po,svc,pvc
kubectl -n capstone get pods -o wide
kubectl -n capstone autoscale deployment capstone-web \
  --cpu-percent=50 --min=2 –max=5
kubectl -n capstone get svc
curl http://localhost:30879
```

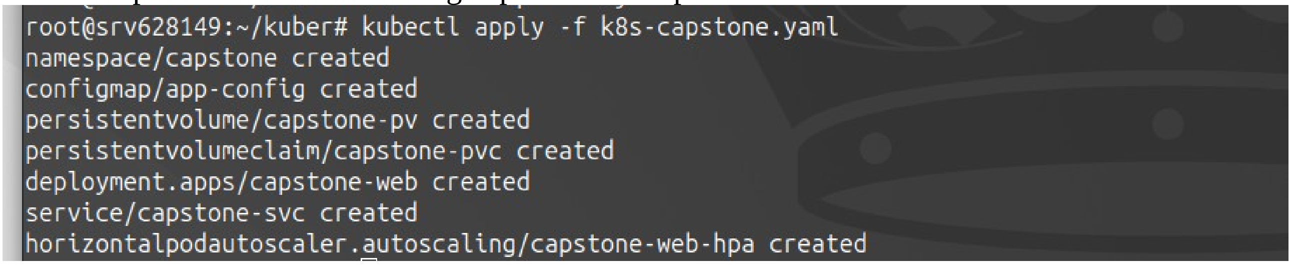
kind load docker-image python-webserver :

```
Image: "python-webserver" with ID
"sha256:61b7a583bfc06155ff3262a37f507fe5fd8c49e044d5b74ab20f2d5f2399fe05" not yet present
on node "kind-control-plane", loading…
```

kubectl apply -f k8s-capstone.yaml :

```
namespace/capstone created
```

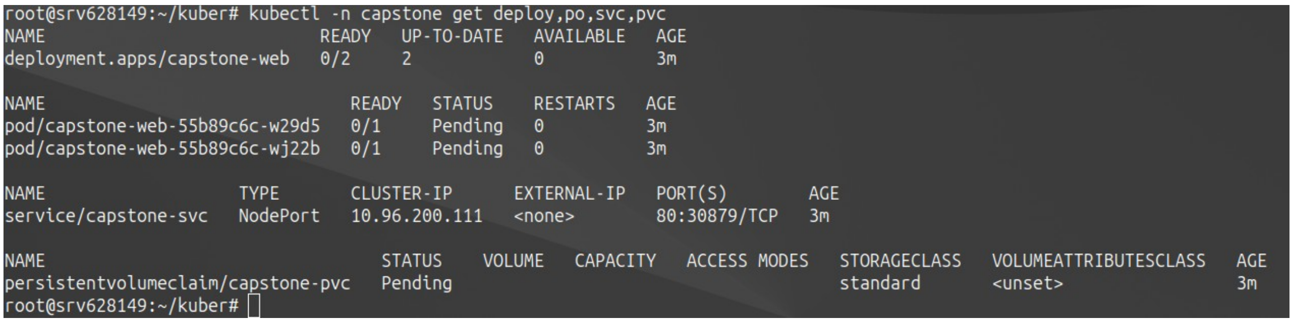
```
configmap/app-config created
persistentvolume/capstone-pv created
persistentvolumeclaim/capstone-pvc created
deployment.apps/capstone-web created
service/capstone-svc created
horizontalpodautoscaler.autoscaling/capstone-web-hpa created
```



```
root@srv628149:~/kuber# kubectl apply -f k8s-capstone.yaml
namespace/capstone created
configmap/app-config created
persistentvolume/capstone-pv created
persistentvolumeclaim/capstone-pvc created
deployment.apps/capstone-web created
service/capstone-svc created
horizontalpodautoscaler.autoscaling/capstone-web-hpa created
```

kubectl -n capstone get deploy,po,svc,pvc :

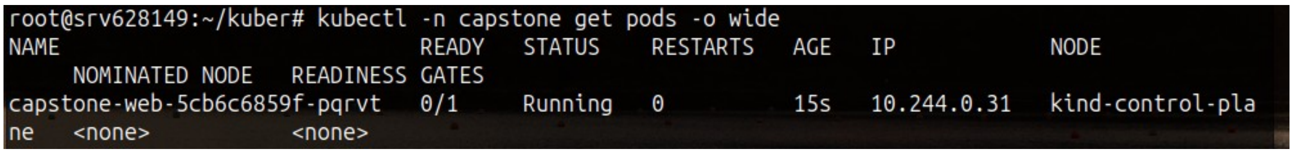
```
NAME                         READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/capstone-web   0/2     2            0           3m

NAME                              READY   STATUS    RESTARTS   AGE
pod/capstone-web-55b89c6c-w29d5   0/1     Pending   0          3m
pod/capstone-web-55b89c6c-wj22b   0/1     Pending   0          3m

NAME                   TYPE       CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
service/capstone-svc   NodePort   10.96.200.111   <none>        80:30879/TCP   3m

NAME                                 STATUS    VOLUME   CAPACITY   ACCESS MODES
STORAGECLASS   VOLUMEATTRIBUTESCLASS   AGE
persistentvolumeclaim/capstone-pvc   Pending                                  standard       <unset>
3m
```



```
root@srv628149:~/kuber# kubectl -n capstone get deploy,po,svc,pvc
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/capstone-web   0/2     2            0           3m

NAME                              READY   STATUS    RESTARTS   AGE
pod/capstone-web-55b89c6c-w29d5   0/1     Pending   0          3m
pod/capstone-web-55b89c6c-wj22b   0/1     Pending   0          3m

NAME                   TYPE       CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
service/capstone-svc   NodePort   10.96.200.111   <none>        80:30879/TCP   3m

NAME                                 STATUS    VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS   VOLUMEATTRIBUTESCLASS   AGE
persistentvolumeclaim/capstone-pvc   Pending                                      standard       <unset>                 3m
root@srv628149:~/kuber#
```

kubectl -n capstone get pods -o wide :

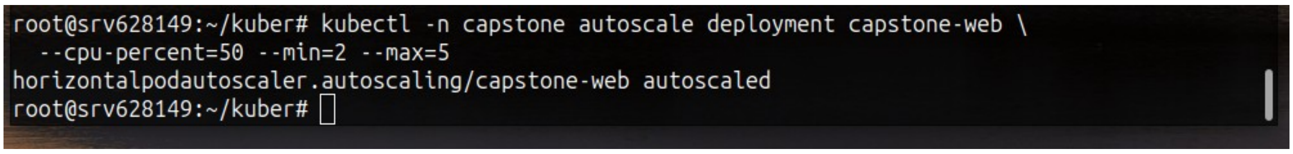
```
NAME                            READY   STATUS    RESTARTS   AGE   IP            NODE
NOMINATED NODE   READINESS GATES
capstone-web-5cb6c6859f-pqrvt   0/1     Running   0          15s   10.244.0.31   kind-control-plane
<none>           <none>
```



```
root@srv628149:~/kuber# kubectl -n capstone get pods -o wide
NAME                            READY   STATUS    RESTARTS   AGE   IP            NODE
     NOMINATED NODE   READINESS GATES
capstone-web-5cb6c6859f-pqrvt   0/1     Running   0          15s   10.244.0.31   kind-control-pla
ne   <none>           <none>
```

**kubectl -n capstone autoscale deployment capstone-web \
  --cpu-percent=50 --min=2 –max=5 :**

```
horizontalpodautoscaler.autoscaling/capstone-web autoscaled
```



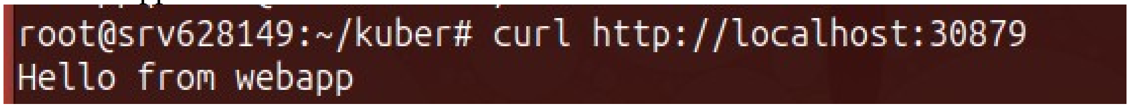
```
root@srv628149:~/kuber# kubectl -n capstone autoscale deployment capstone-web \
  --cpu-percent=50 --min=2 --max=5
horizontalpodautoscaler.autoscaling/capstone-web autoscaled
root@srv628149:~/kuber#
```

kubectl -n capstone get svc :

```
NAME          TYPE       CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
capstone-svc   NodePort   10.96.200.111   <none>        80:30879/TCP   30m
```

curl http://localhost:30879 :

Hello from webapp



```
root@srv628149:~/kuber# curl http://localhost:30879
Hello from webapp
```