

Assignment-9 Submission

- **Student Name:** I. Anvith
- **Student ID:** 2302536

Task 1: Create a Pod with an NGINX Image

nginx-pod.yaml:

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
  labels:
    app: nginx
spec:
  containers:
  - name: nginx-container
    image: nginx:latest # Use the latest NGINX image
    ports:
    - containerPort: 80 # NGINX listens on port 80
```

Apply and Verify:

```
kubectl get pods
```

Screenshot:



Task 2: Create a Deployment with 4 Replicas

nginx-deployment.yaml:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
```

```

replicas: 4 # Specify 4 replicas
selector:
  matchLabels:
    app: nginx # This selector must match the Pod template's labels
template:
  metadata:
    labels:
      app: nginx # Labels for the Pods created by this Deployment
  spec:
    containers:
      - name: nginx-container
        image: nginx:latest
        ports:
          - containerPort: 80

```

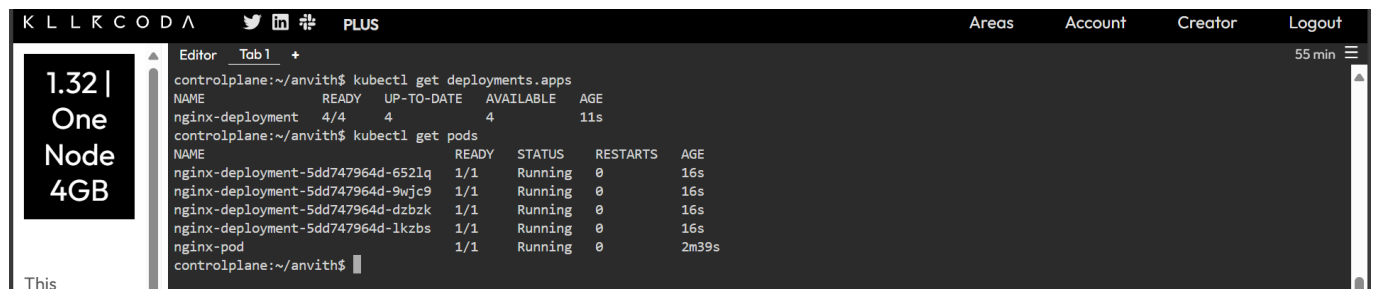
Apply and Verify:

```

kubectl get deploy
kubectl get pods

```

Screenshot:



Task 2.1: Test Self-Healing (Delete a Pod)

```

kubectl get pods
kubectl delete pod <pod-name>
kubectl get deploy
kubectl get pods

```

Screenshot:

```

controlplane:~/anvith$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-5dd747964d-652lq   1/1     Running   0           16s
nginx-deployment-5dd747964d-9wjc9   1/1     Running   0           16s
nginx-deployment-5dd747964d-dzbzk   1/1     Running   0           16s
nginx-deployment-5dd747964d-lkzbs   1/1     Running   0           16s
nginx-pod                            1/1     Running   0           2m39s
controlplane:~/anvith$ kubectl delete pod nginx-deployment-5dd747964d-652lq
pod "nginx-deployment-5dd747964d-652lq" deleted
controlplane:~/anvith$ kubectl get deployments.apps
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment    4/4     4             4           60s
controlplane:~/anvith$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-5dd747964d-9wjc9   1/1     Running   0           62s
nginx-deployment-5dd747964d-dzbzk   1/1     Running   0           62s
nginx-deployment-5dd747964d-gd24g   1/1     Running   0           6s
nginx-deployment-5dd747964d-lkzbs   1/1     Running   0           62s
nginx-pod                            1/1     Running   0           3m25s
controlplane:~/anvith$ █

```

Task 3.1: Create a ClusterIP Service

nginx-clusterip-service.yaml:

```

apiVersion: v1
kind: Service
metadata:
  name: nginx-clusterip-service
spec:
  selector:
    app: nginx # Select Pods with the label 'app: nginx'
  ports:
    - protocol: TCP
      port: 80 # Service port
      targetPort: 80 # Port on the Pods
  type: ClusterIP # Expose the Service on a cluster-internal IP

```

Apply and Verify:

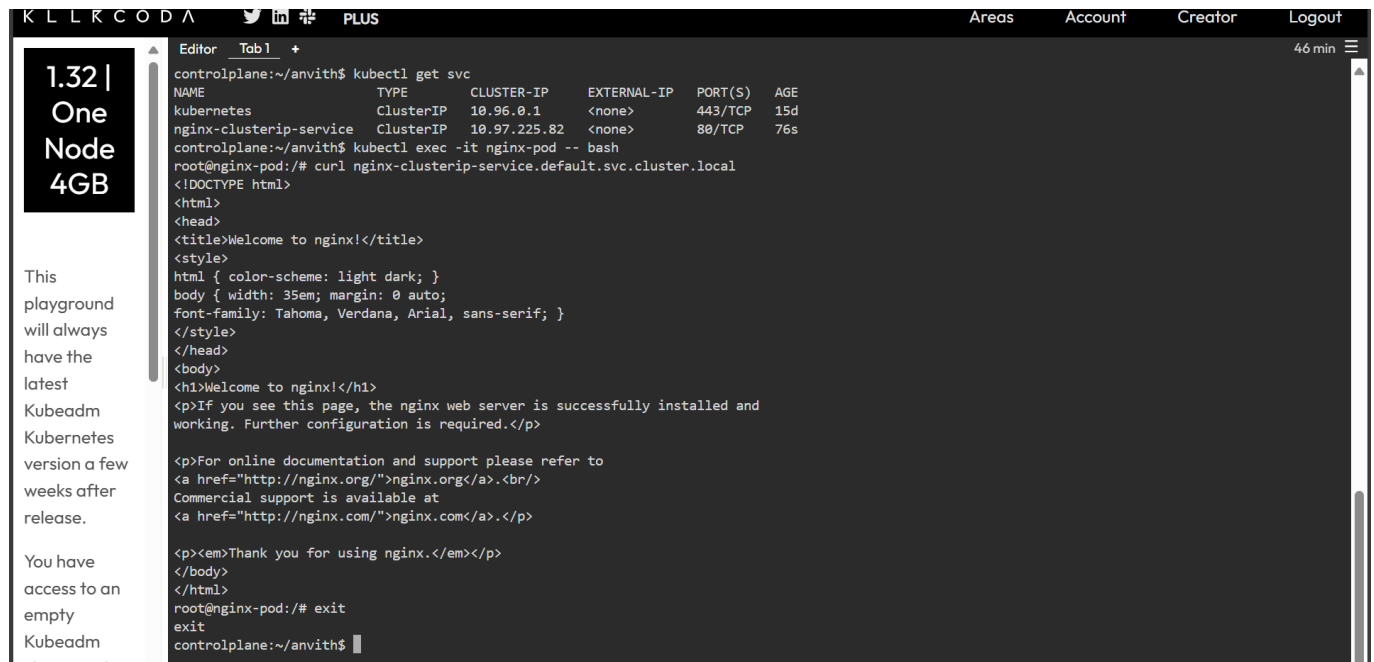
```

kubectl apply -f nginx-clusterip-service.yaml
kubectl get services

# Fired from inside a pod to check resolution
curl -v http://nginx-clusterip-service.default.svc.cluster.local

```

Screenshot:



1.32 | One Node 4GB

This playground will always have the latest Kubeadm Kubernetes version a few weeks after release. You have access to an empty Kubeadm

```
controlplane:~/anvith$ kubectl get svc
NAME                                TYPE        CLUSTER-IP    EXTERNAL-IP  PORT(S)    AGE
kubernetes                         ClusterIP   10.96.0.1     <none>       443/TCP    15d
nginx-clusterip-service            ClusterIP   10.97.225.82  <none>       80/TCP     76s

controlplane:~/anvith$ kubectl exec -it nginx-pod -- bash
root@nginx-pod:/# curl nginx-clusterip-service.default.svc.cluster.local
<!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/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
root@nginx-pod:/# exit
exit
controlplane:~/anvith$
```

Task 3.2: Create a NodePort Service

nginx-nodeport-service.yaml:

```
apiVersion: v1
kind: Service
metadata:
  name: nginx-nodeport-service
spec:
  selector:
    app: nginx
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
      nodePort: 30000 # Choose a port in the range 30000-32767
  type: NodePort # Expose the Service on each node's IP at a static port
```

Apply and Verify:

```
kubectl apply -f nginx-nodeport-service.yaml
kubectl get services

**Testing Nodeport:**
``bash
# Fired from inside a pod to check resolution
curl -v http://nginx-clusterip-service.default.svc.cluster.local

# Fired on NodeIP and nodePort
curl -v http://172.30.1.2:30000
```

```
K L L K C O D A  PLUS 41 min
Editor Tab1 +
controlplane:~/anvith$ hostname -I
172.30.1.2 172.17.0.1 192.168.0.0
controlplane:~/anvith$ ^C
controlplane:~/anvith$ ^C
controlplane:~/anvith$ kubectl describe svc nginx-nodeport-service
Name:          nginx-nodeport-service
Namespace:     default
Labels:        app=nginx-nodeport-service
Annotations:   <none>
Selector:      app=nginx-deployment
Type:          NodePort
IP Family Policy: SingleStack
IP Families:   IPv4
IP:            10.103.208.107
IPs:           10.103.208.107
Port:          <unset> 80/TCP
TargetPort:    80/TCP
NodePort:      <unset> 30000/TCP
Endpoints:     192.168.0.11:80,192.168.0.8:80,192.168.0.9:80 + 1 more...
Session Affinity: None
External Traffic Policy: Cluster
Internal Traffic Policy: Cluster
Events:        <none>
controlplane:~/anvith$ ^C
controlplane:~/anvith$ ^C
controlplane:~/anvith$ curl http://172.30.1.2:30000
<!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>
</html>
</head>
<body>
<h1>Welcome to nginx!</h1>
<hr/>
</body>
</html>
```

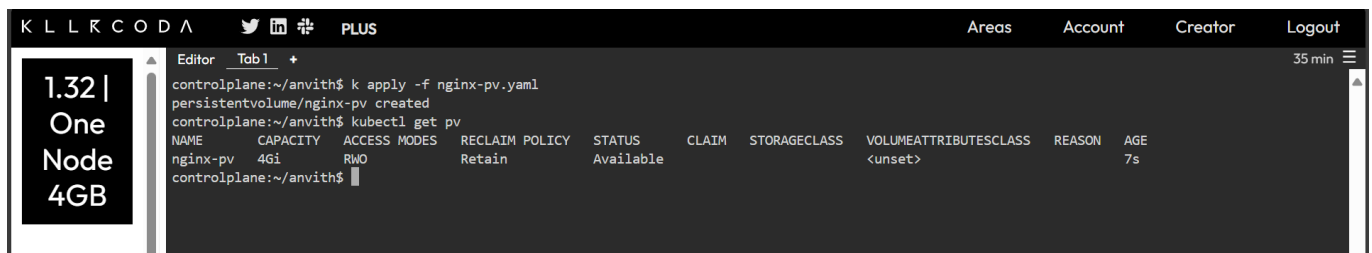
```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: nginx-pv
```

```
spec:
  storageClassName: manual
  capacity:
    storage: 4Gi # Request 1 GB of storage
  accessModes:
    - ReadWriteOnce # The volume can be mounted as read-write by a single node
  hostPath:
    path: /mnt/data # This is for testing on a single-node cluster (like kind).
In a real cluster, use a proper storage provider.
  persistentVolumeReclaimPolicy: Retain # Keep the volume even if the PVC is
deleted
```

Apply and Verify:

```
kubectl apply -f nginx-pv.yaml
kubectl get pv
```

Screenshot:



Task 4.2: Create a Persistent Volume Claim (PVC)

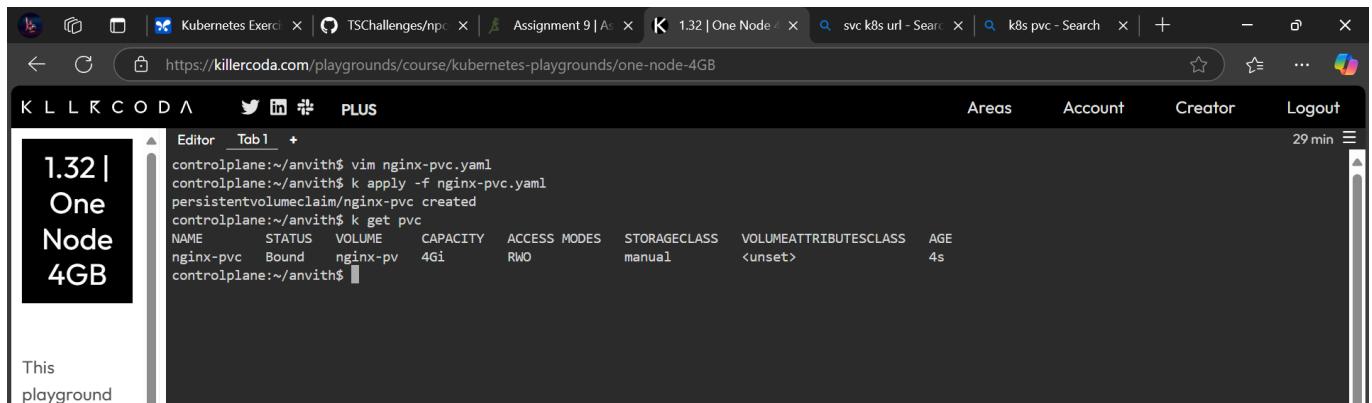
nginx-pvc.yaml:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: nginx-pvc
spec:
  storageClassName: manual
  accessModes:
    - ReadWriteOnce # Must match the PV's access modes
  resources:
    requests:
      storage: 500Mi # Request 500 MB of storage (less than or equal to the PV's
capacity)
```

Apply and Verify:

```
kubectl apply -f nginx-pvc.yaml
kubectl get pvc
```

Screenshot:



Task 5: Assign PVC to a Pod

nginx-pvc-pod.yaml:

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pvc-pod
spec:
  containers:
    - name: nginx-container
      image: nginx:latest
      ports:
        - containerPort: 80
      volumeMounts:
        - name: nginx-volume # Name of the volume
          mountPath: /usr/share/nginx/html # Mount point inside the container
          (standard Nginx data directory)
  volumes:
    - name: nginx-volume # Name of the volume (must match volumeMounts.name)
      persistentVolumeClaim:
        claimName: nginx-pvc # Reference the PVC we created
```

Apply and Verify:

```
kubectl apply -f nginx-pvc-pod.yaml
kubectl get pods nginx-pvc-pod

# Verify the mount point
kubectl describe pod nginx-pvc-pod | grep -i volume -C 2
kubectl get pv
kubectl get pvc
```

Screenshot:

1.32 | One Node 4GB

This playground will always have the latest Kubeadm Kubernetes version a few weeks after release. You have access to an

Editor Tab1 +15 min

controlplane:~/anvith\$ kubectl describe pod nginx-pvc-pod | grep -i volume -C 2
Environment: <none>
Mounts:
 /usr/share/nginx/html from nginx-volume (rw)
 /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-nhdxj (ro)
Conditions:
--
ContainersReady True
PodScheduled True
Volumes:
 nginx-volume:
 Type: PersistentVolumeClaim (a reference to a PersistentVolumeClaim in the same namespace)
 ClaimName: nginx-pvc
 ReadOnly: false
 kube-api-access-nhdxj:
 Type: Projected (a volume that contains injected data from multiple sources)
 TokenExpirationSeconds: 3607
 ConfigMapName: kube-root-ca.crt
controlplane:~/anvith\$ kubectl get pods nginx-pvc-pod
NAME READY STATUS RESTARTS AGE
nginx-pvc-pod 1/1 Running 0 2m28s
controlplane:~/anvith\$ kubectl get pv
NAME CAPACITY ACCESS MODES RECLAIM POLICY STATUS CLAIM STORAGECLASS VOLUMEATTRIBUTESCLASS REASON AGE
nginx-pv 4Gi RWO Retain Bound default/nginx-pvc manual <unset> 14m
controlplane:~/anvith\$ kubectl get pvc
NAME STATUS VOLUME CAPACITY ACCESS MODES STORAGECLASS VOLUMEATTRIBUTESCLASS AGE
nginx-pvc Bound nginx-pv 4Gi RWO manual <unset> 13m
controlplane:~/anvith\$