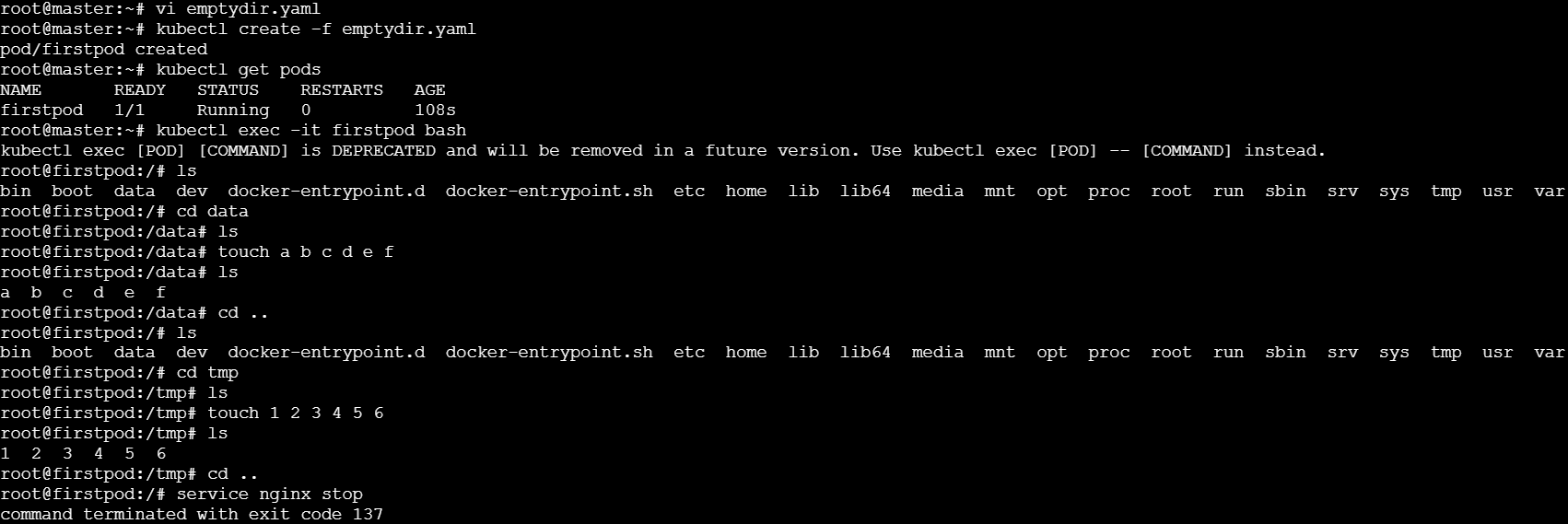
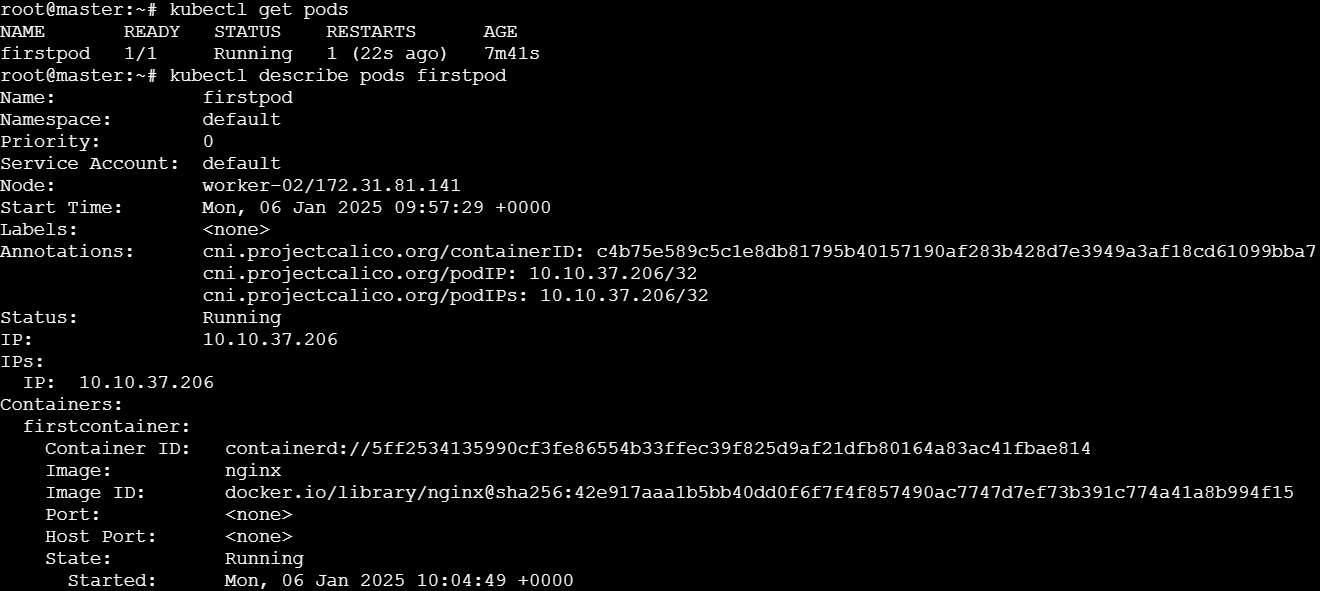
**Kubernetes Tasks-8**

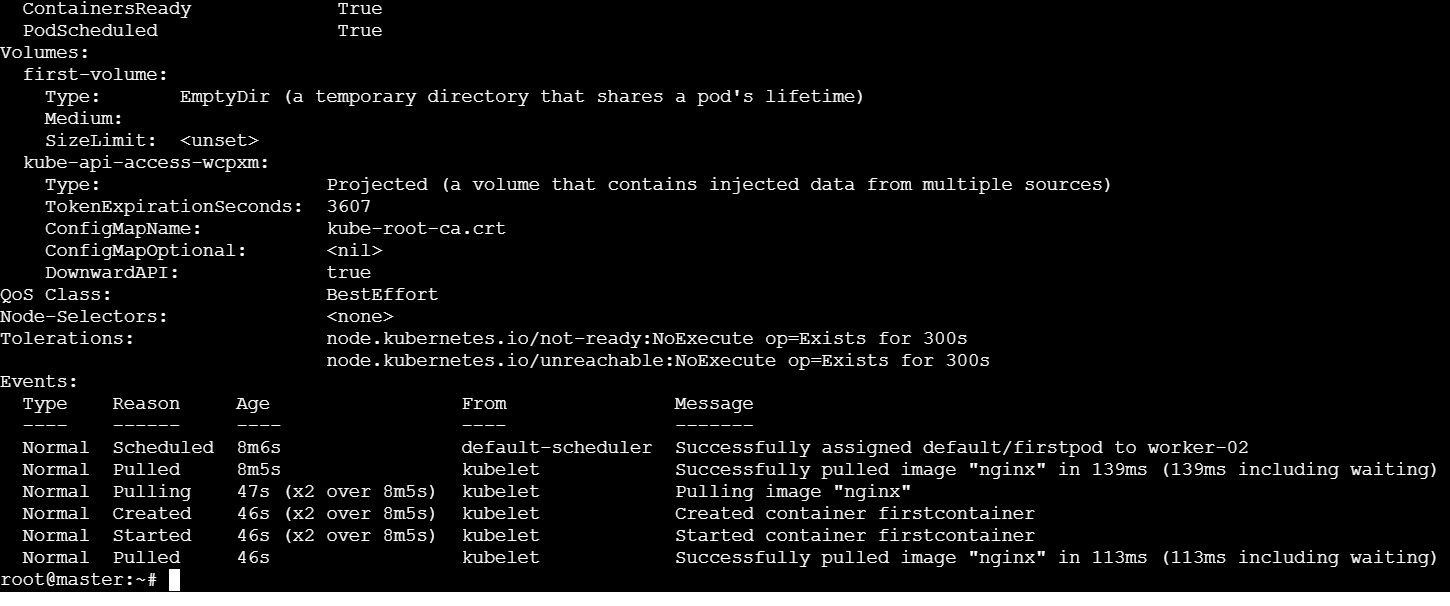
1. **Create and Test a Kubernetes Pod with an EmptyDir Volume**

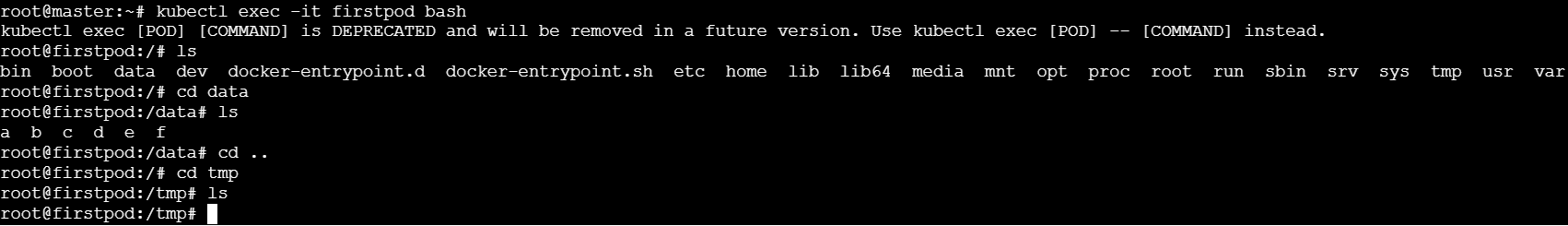
**>> vi emptydir.yaml**

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: firstpod  spec:  containers:  - name: firstcontainer  image: nginx  volumeMounts:  - mountPath: /data #Directory inside container  name: first-volume #any logical name  volumes:  - name: first-volume  emptyDir: {} #Blank Object |

****

****

****

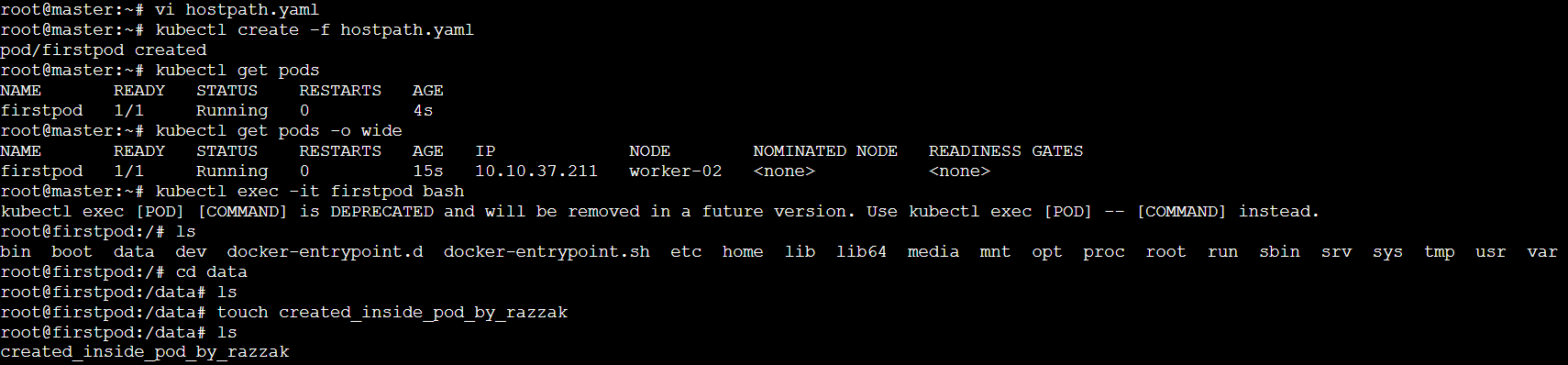
****

1. **Configure a HostPath Volume in Kubernetes and Validate Data Persistence**

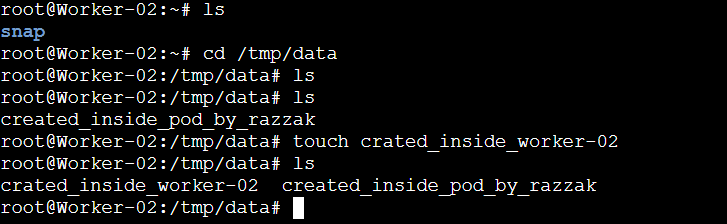
**>>vi hostpath.yaml**

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: firstpod  spec:  containers:  - name: firstcontainer  image: nginx  volumeMounts:  - mountPath: /data #Directory inside container  name: first-volume #any logical name  volumes:  - name: first-volume  hostPath:  path: /tmp/data # Path inside host machine (Minikube) |

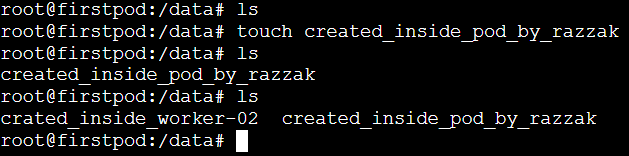
* **Master**

****

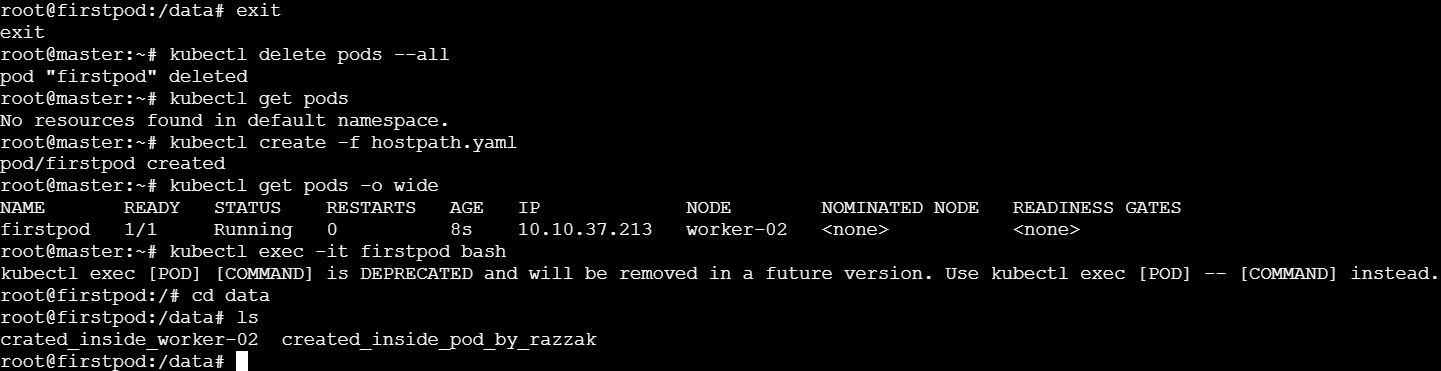
* **In worker-02:**

****

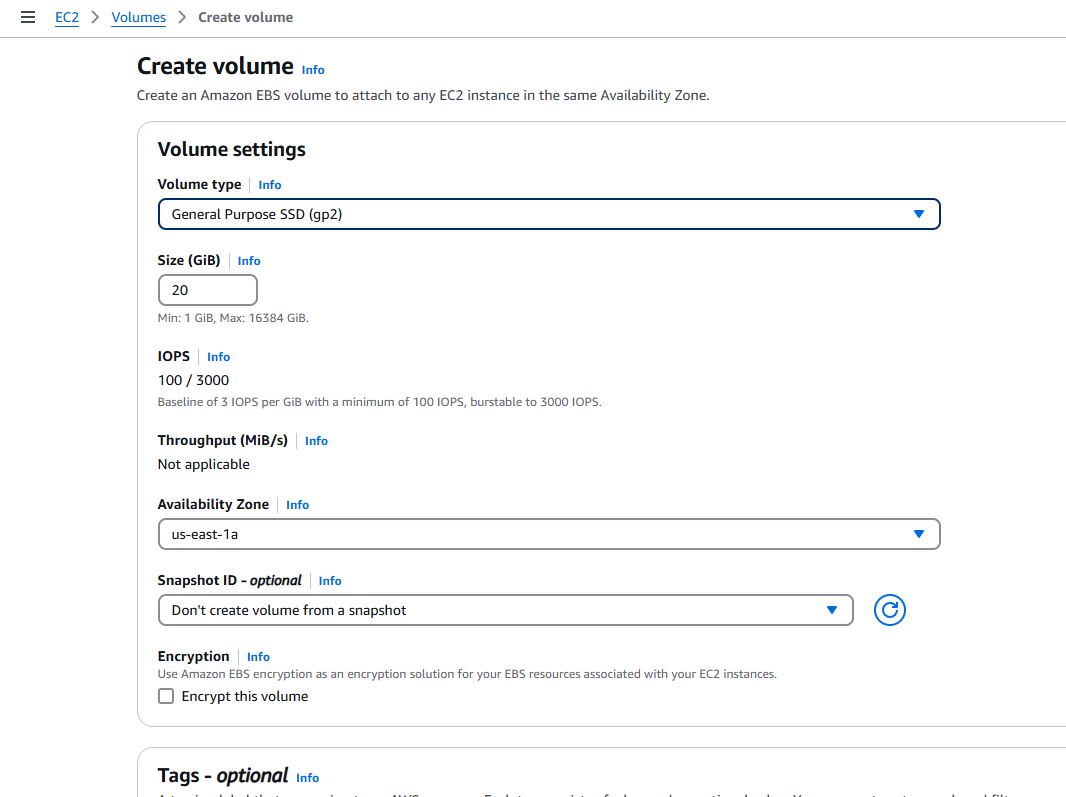
* **Master:**

****

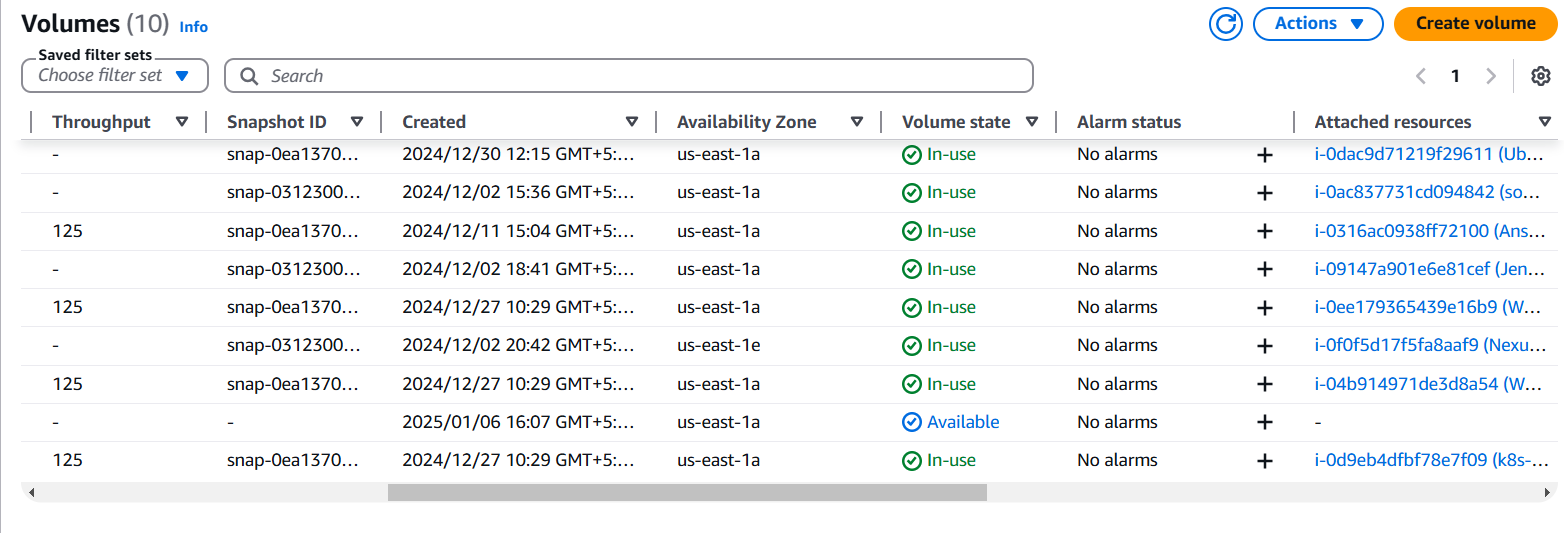
* **Now, we are deleting the pod (firstpod) and then creating it again to see the persistence of data.**

****

1. **Deploy an Amazon EBS Volume Using Persistent Volume and Persistent Volume Claim (PVC)**

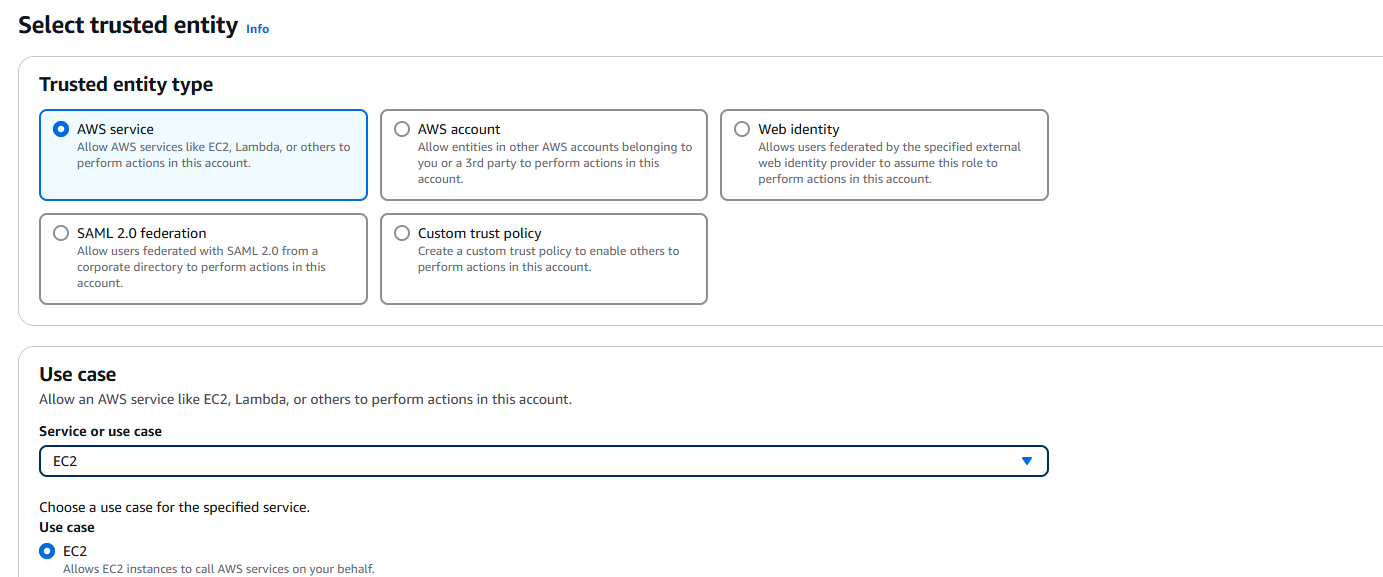
****

* **EBS volume is created and the volume state is available:**

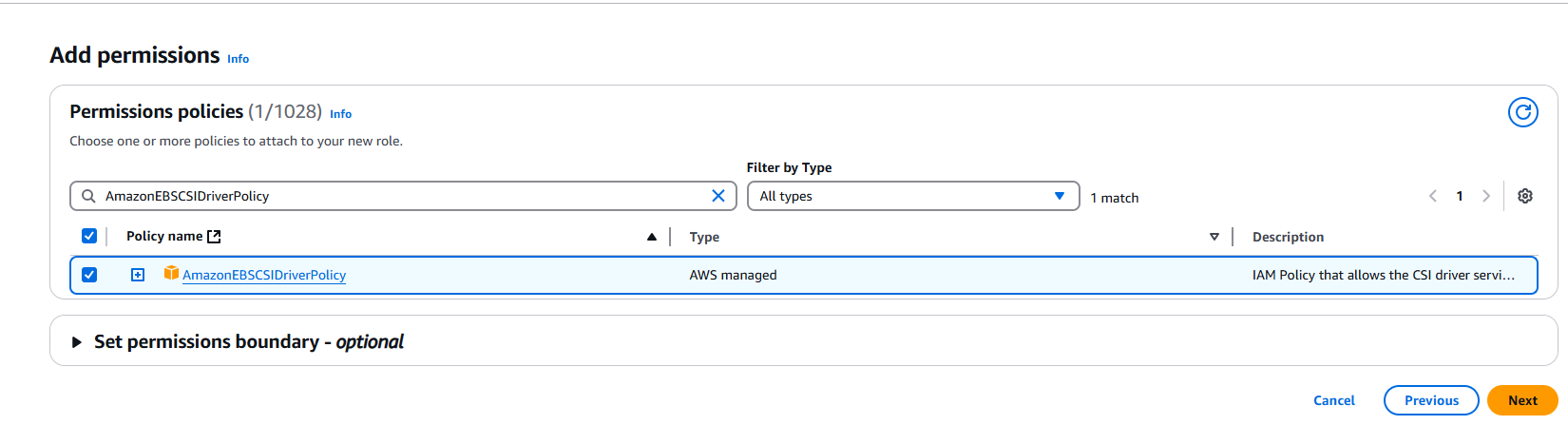
****

* **Create one IAM role for aws-ebs-csi-driver:**

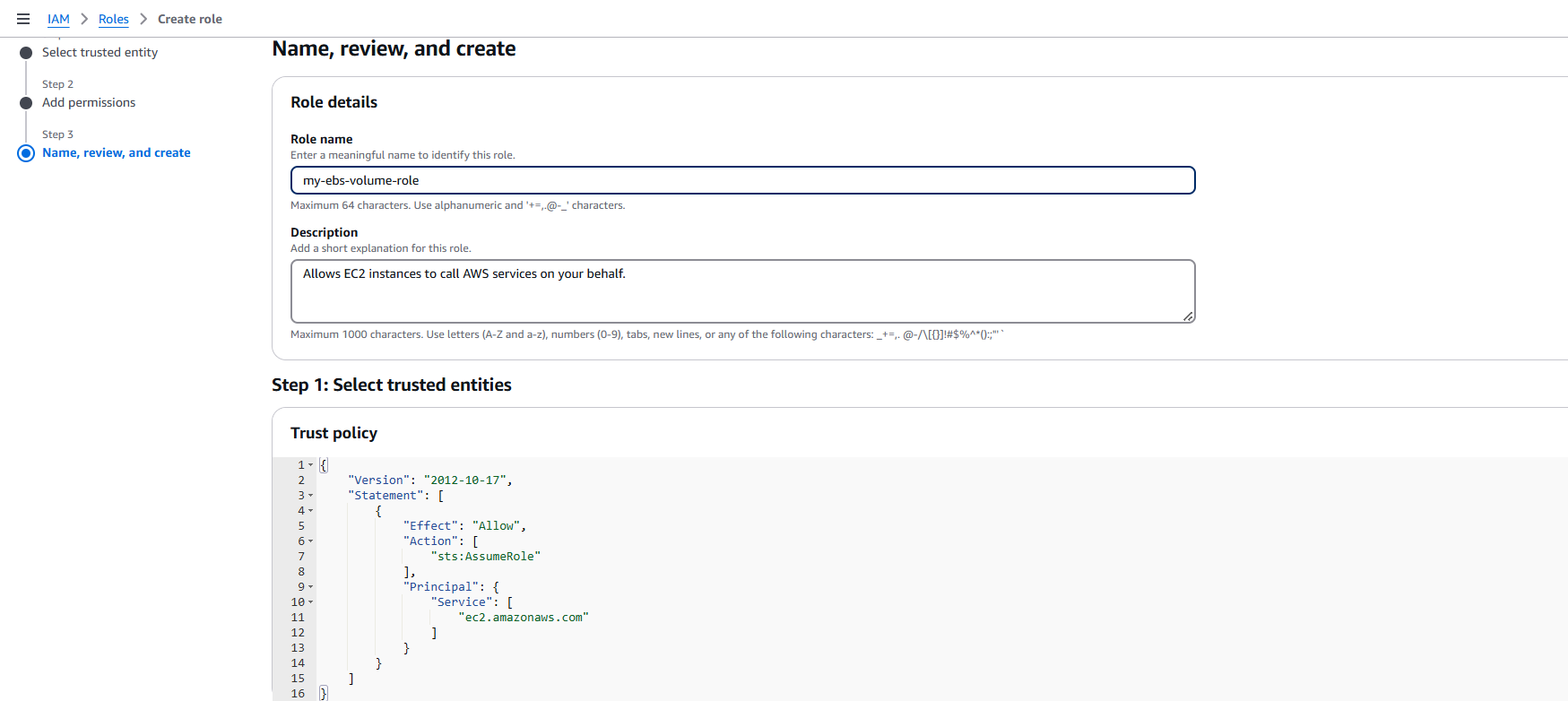
**Go to IAM > roles> create role>**

****

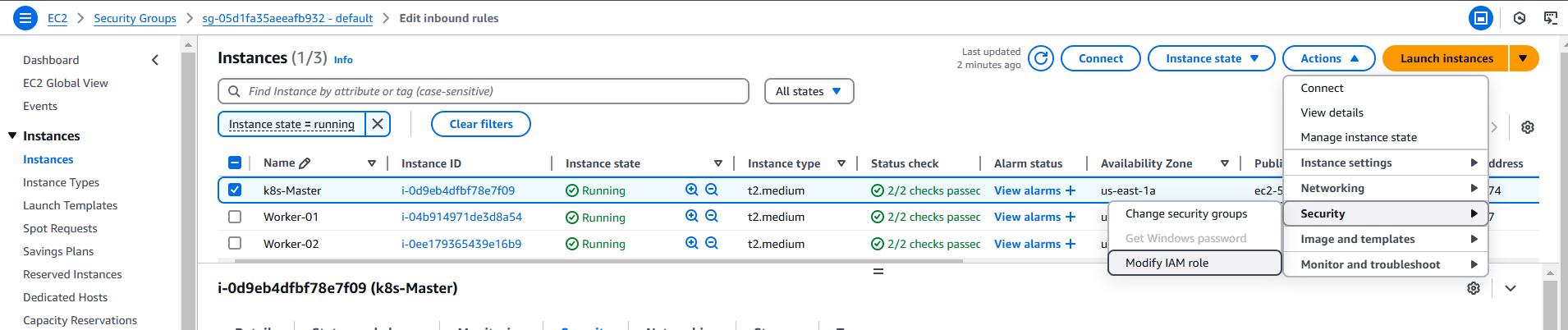
* **Search for the AmazonEBSCSIDriverPolicy. Select the policy >> click on next**

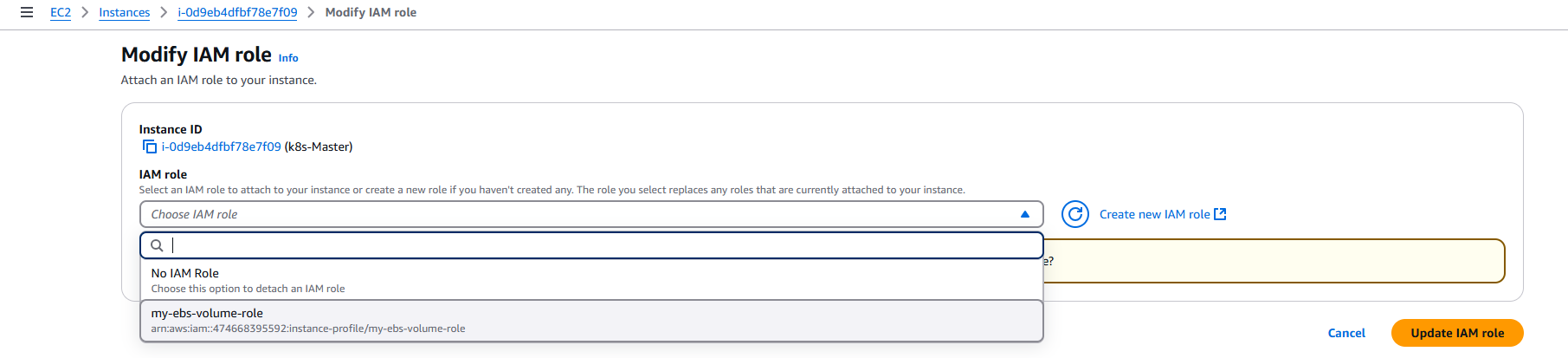
****

* **Now create one role**

****

* **Go to instances –Actions–security---Modify IAM role**

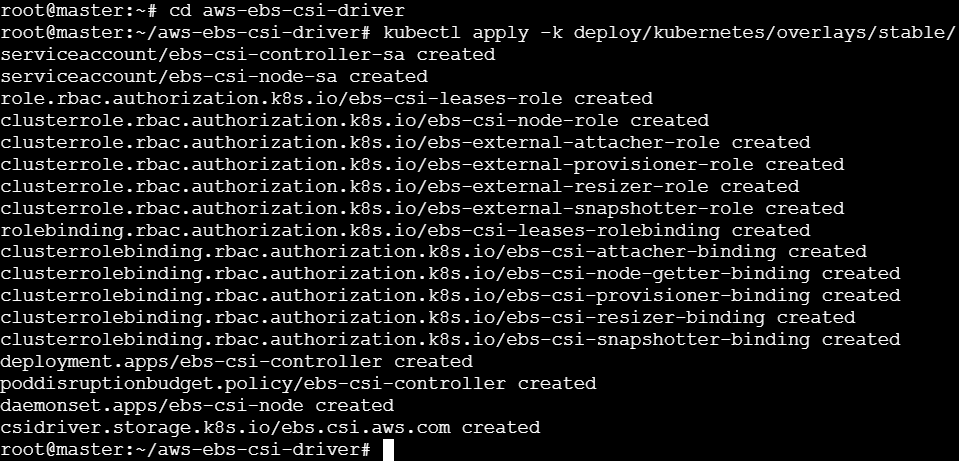
****

****

* **Now configure the same role for worker-01 and worker-02 nodes as well:**
* **Install AWS EBS CSI driver on k8s master node.**

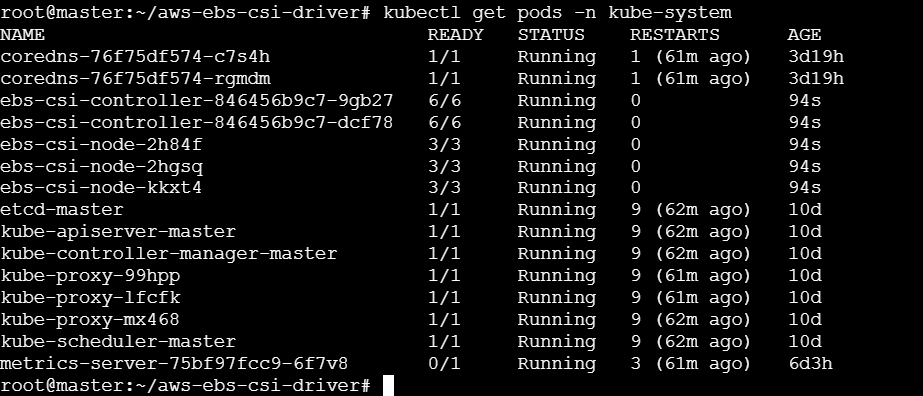
**>> git clone** [**https://github.com/kubernetes-sigs/aws-ebs-csi-driver.git**](https://github.com/kubernetes-sigs/aws-ebs-csi-driver.git)

****

****

* **To verify the driver is installed and running**

**>> kubectl get pods -n kube-system**

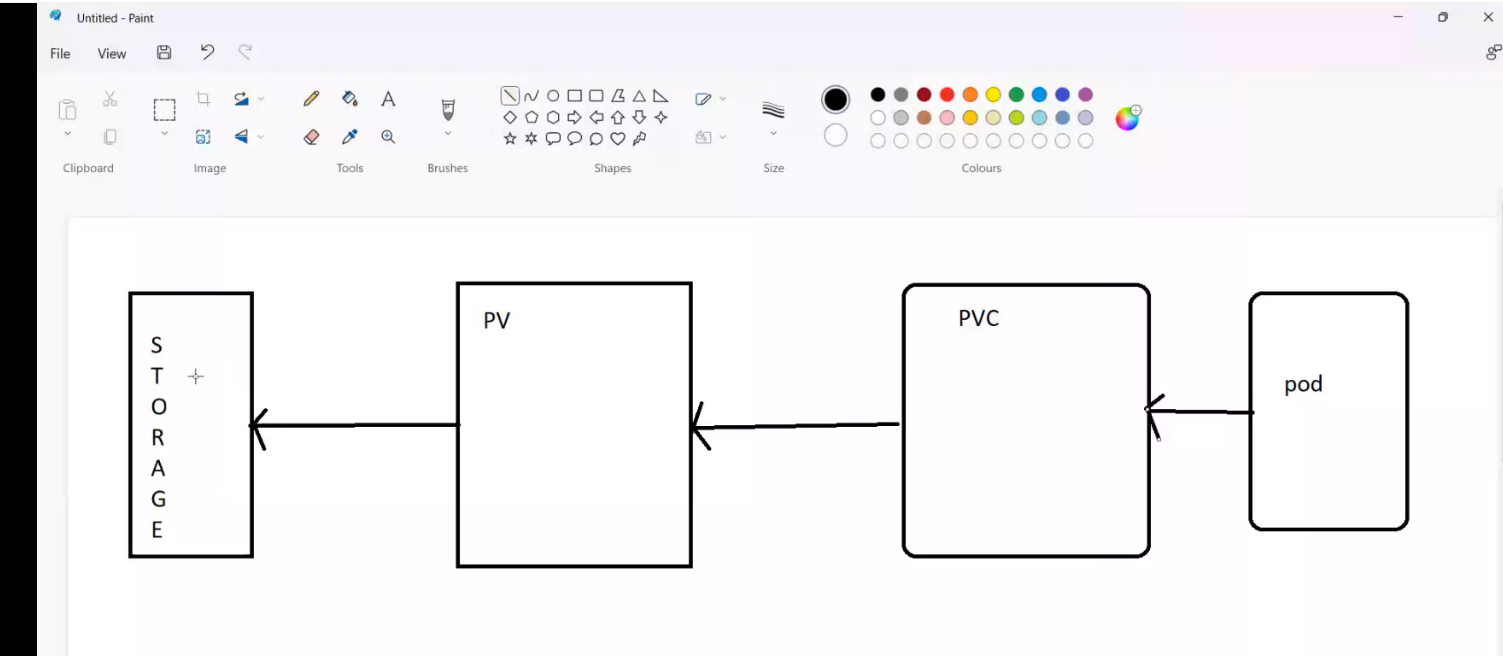
****

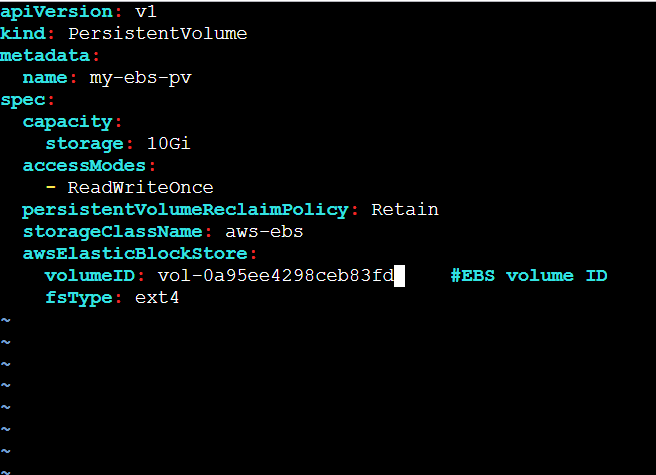
* **Now we need to create one Persistent volume for our EBS.**

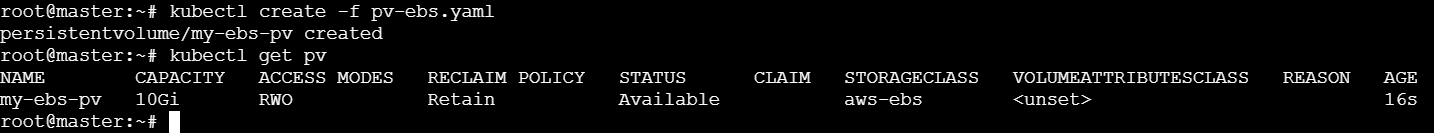
**>> vi pv-ebs.yaml**

|  |
| --- |
| apiVersion: v1  kind: PersistentVolume  metadata:  name: my-ebs-pv  spec:  capacity:  storage: 10Gi  accessModes:  - ReadWriteOnce  persistentVolumeReclaimPolicy: Retain  storageClassName: aws-ebs  awsElasticBlockStore:  volumeID: <your-ebs-volume-id> #EBS volume ID  fsType: ext4 |

* **If any pods wants to access the storage then the pvc(persistent volume claim) will claim the request and persistent volume will try to create the storage:**

****

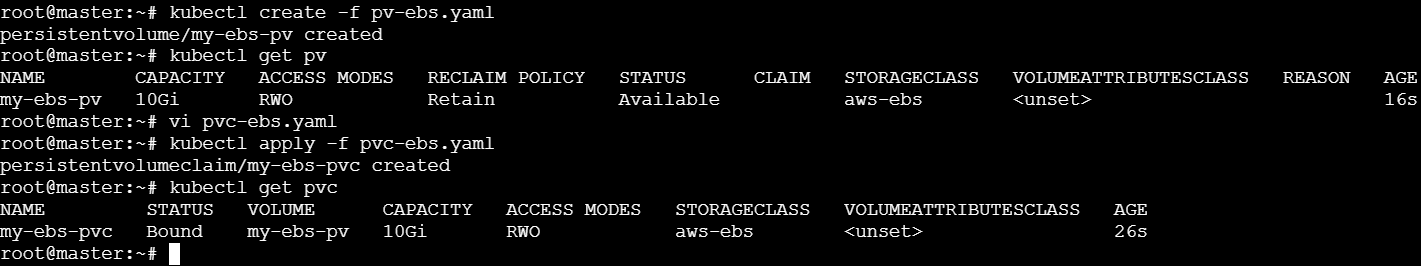
****

****

* **We need to create on claim for Persistent volume (PVC) for our EBS.**

**>> vi pvc-ebs.yaml**

|  |
| --- |
| apiVersion: v1  kind: PersistentVolumeClaim  metadata:  name: my-ebs-pvc  spec:  accessModes:  - ReadWriteOnce  resources:  requests:  storage: 10Gi  storageClassName: aws-ebs |

****

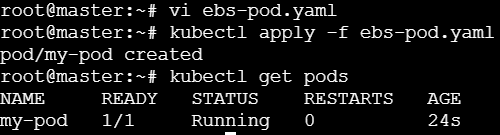
* **Now we check the status of PV:**

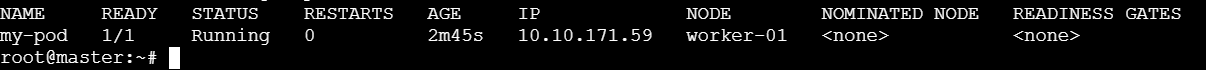
****

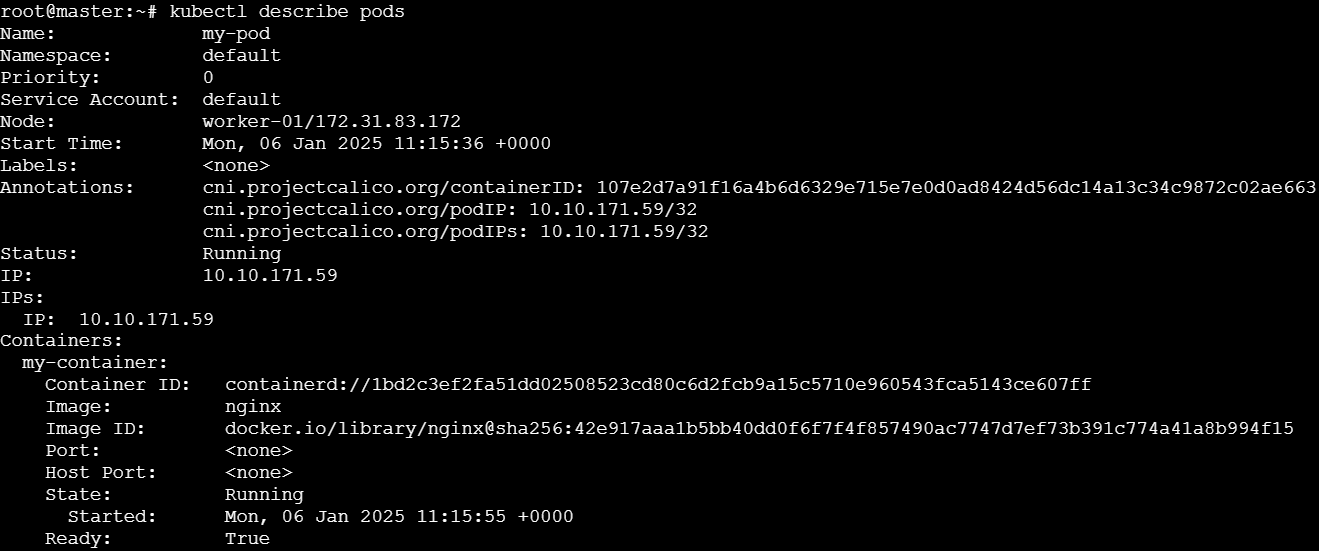
* **Now let us Create pod with PVC:**

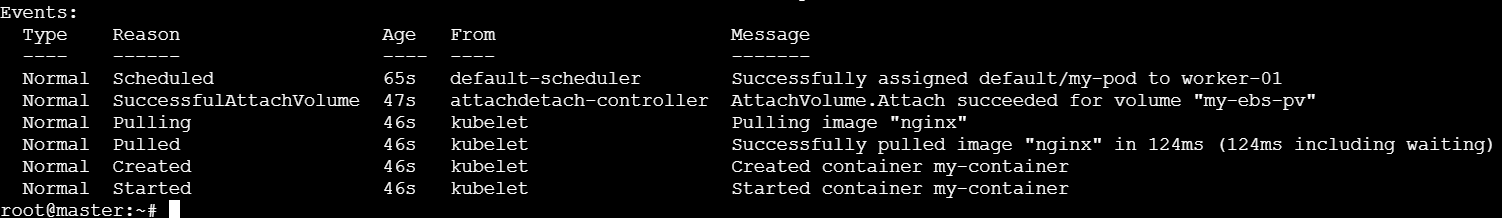
**>> vi ebs-pod.yaml**

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: my-pod  spec:  containers:  - name: my-container  image: nginx  volumeMounts:  - name: data-volume  mountPath: /data #data where we can write data  volumes:  - name: data-volume  persistentVolumeClaim:  claimName: my-ebs-pvc #PVC Name |

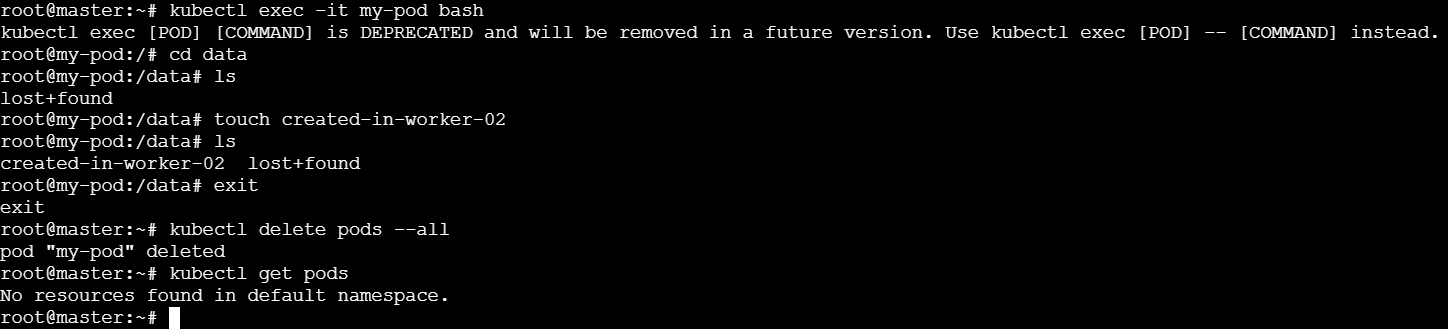
****

****

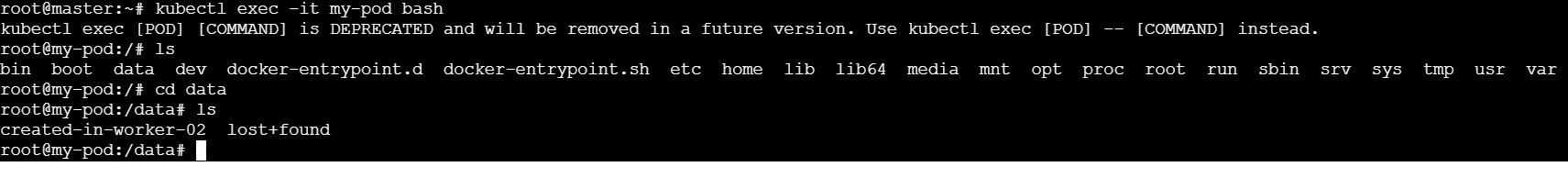
****

****

* **Now let us connect to our pod and create a file in it, so that when we delete the pod and then create one pod again, the data should be present**

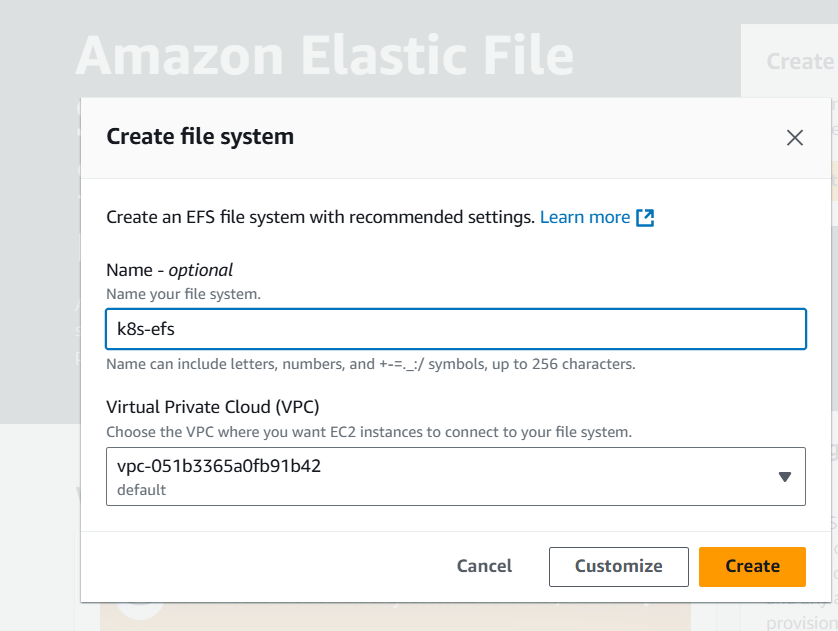
****

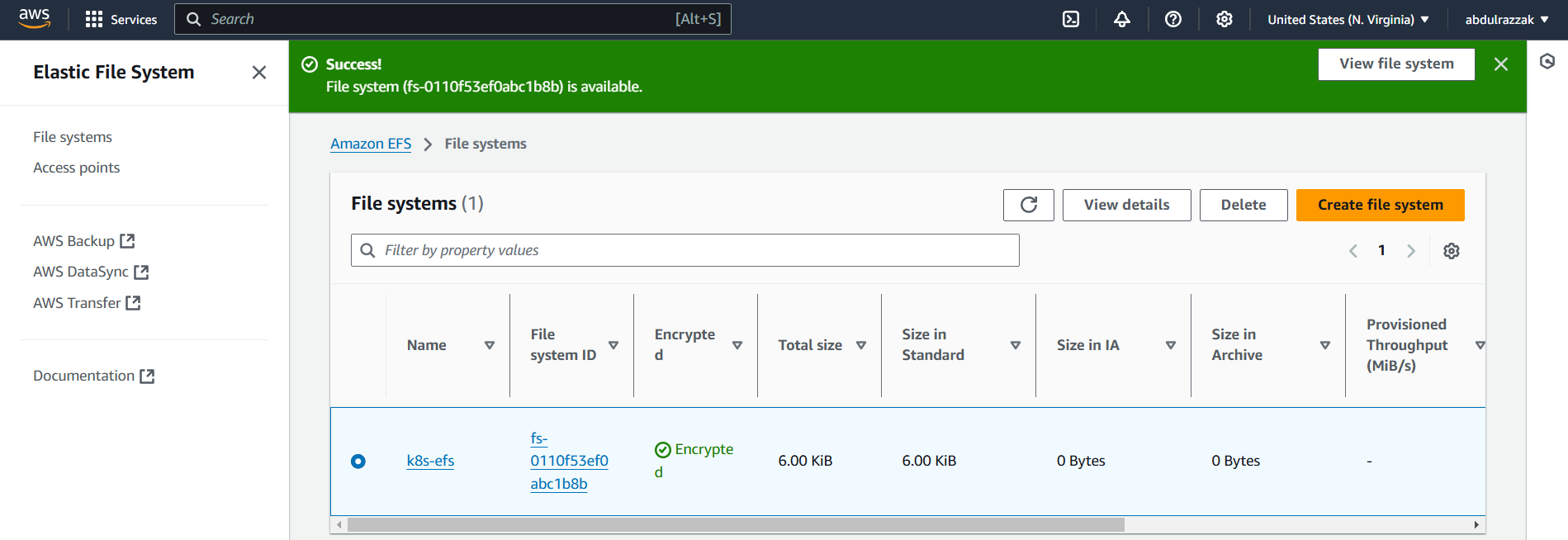
* **Now let us create another pod and connect to it to see if the data is available:**

****

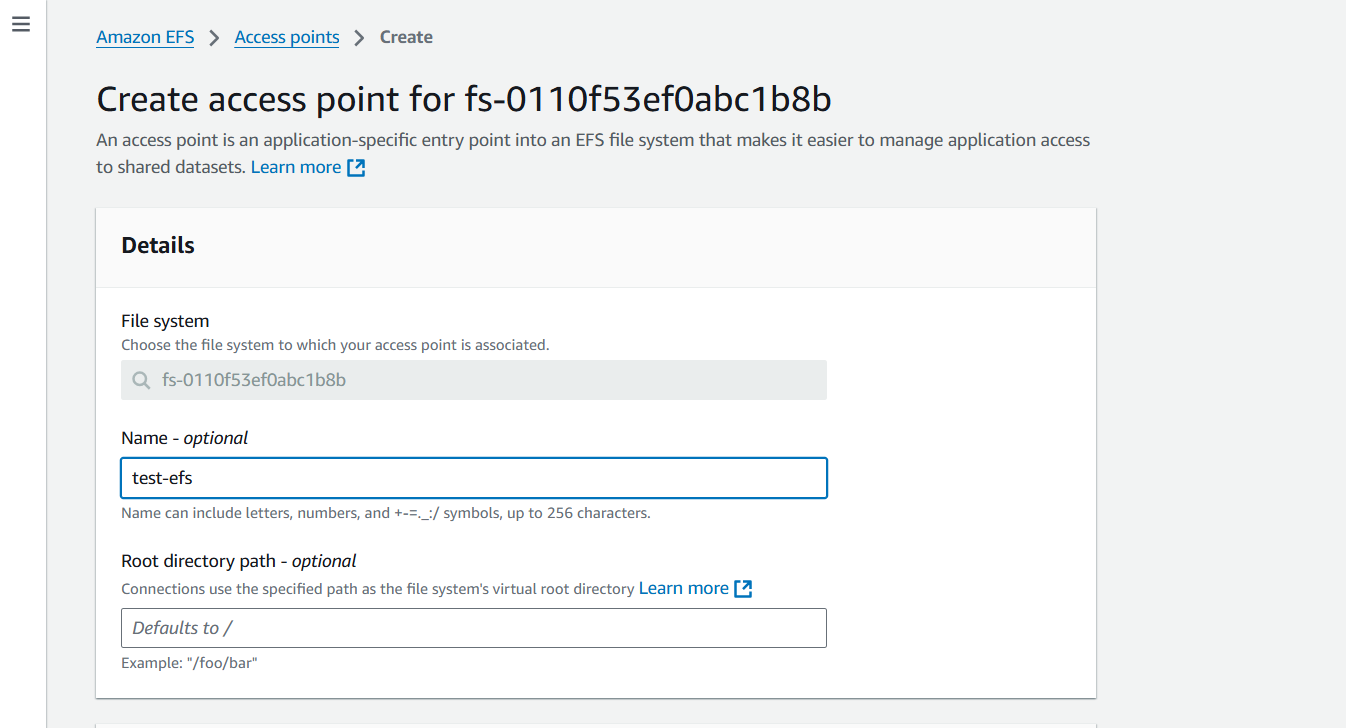
1. **Set Up an Amazon EFS Volume and Attach it to Multiple Pods**

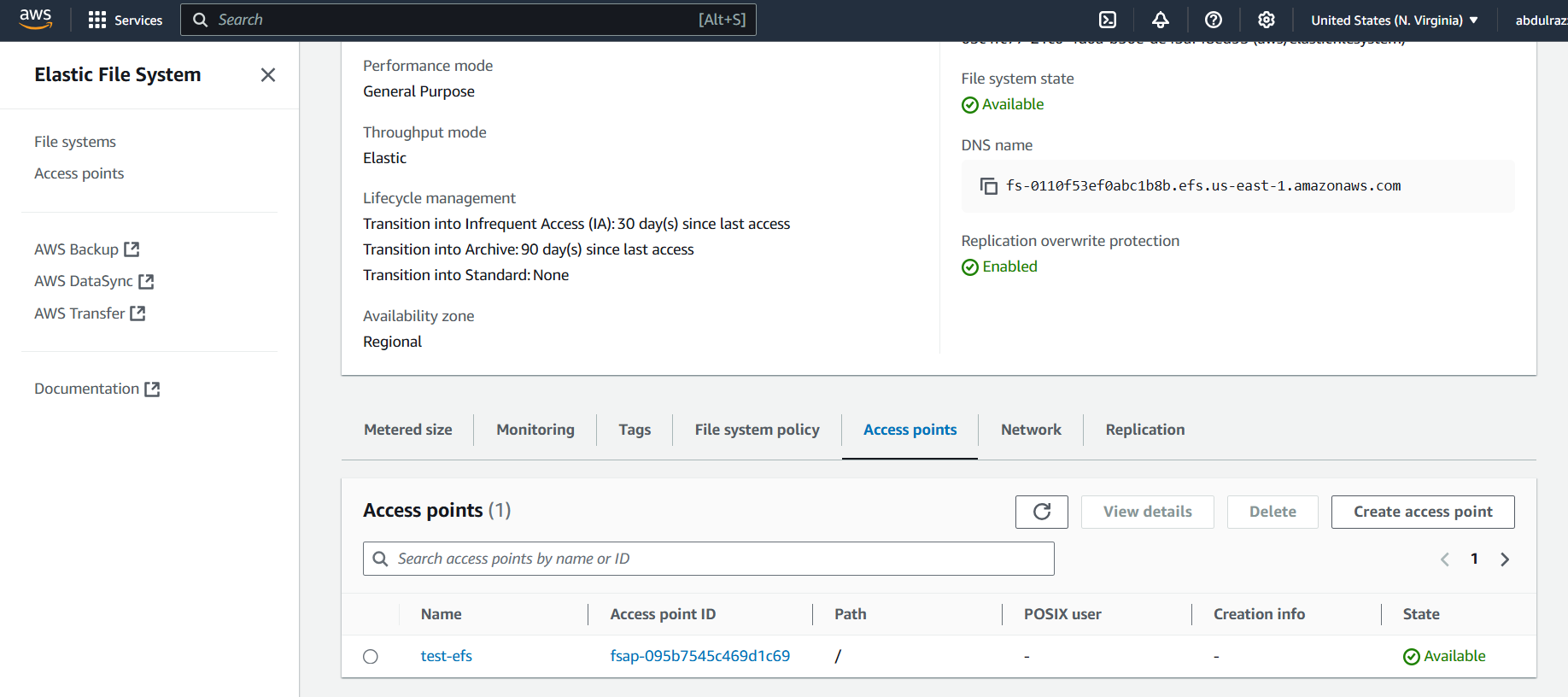
* **Go to EFS service and Create an EFS**

****

****

* **Go to access points tab and create one access point**



****

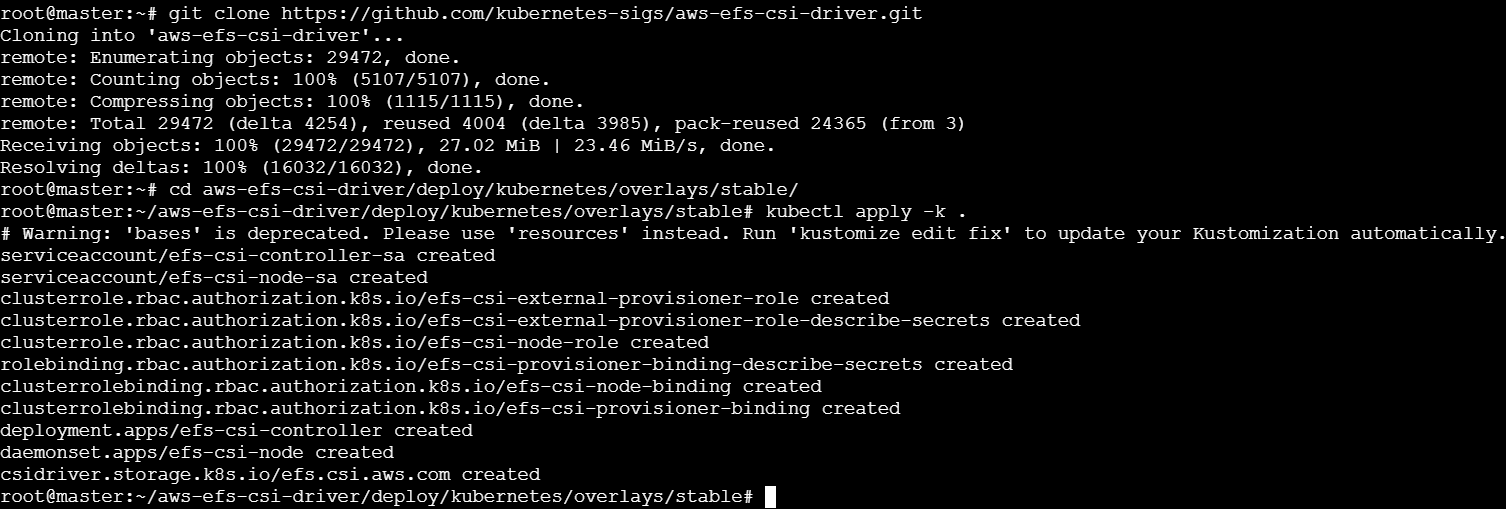
* **Install AWS EFS CSI driver on k8s master node**

**>> git clone** [**https://github.com/kubernetes-sigs/aws-efs-csi-driver.git**](https://github.com/kubernetes-sigs/aws-efs-csi-driver.git)

* **Kubernetes manifests using the kubectl apply command:**

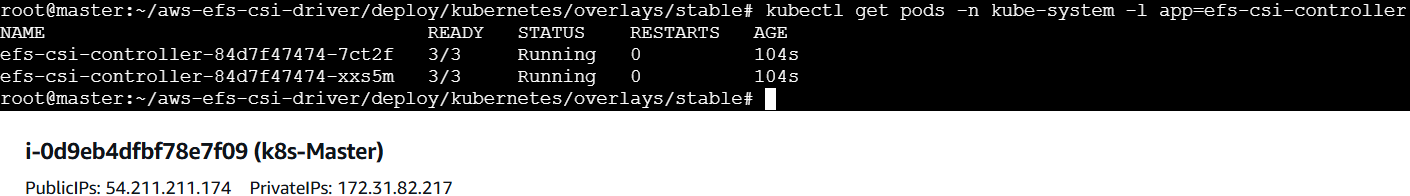
**>> cd aws-efs-csi-driver/deploy/kubernetes/overlays/stable/**

**>>kubectl apply -k .**

****

* **verify the installation.**

**>> kubectl get pods -n kube-system -l app=efs-csi-controller**

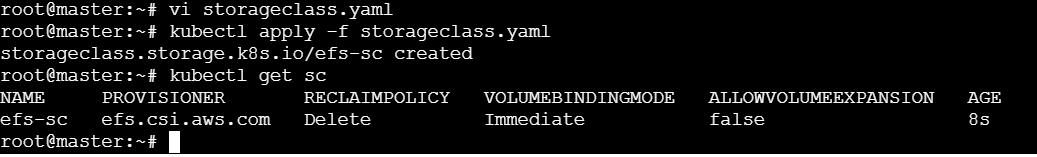
****

* **Now we need to create a StorageClass that uses the EFS CSI driver.**

**>> vi storageclass.yaml**

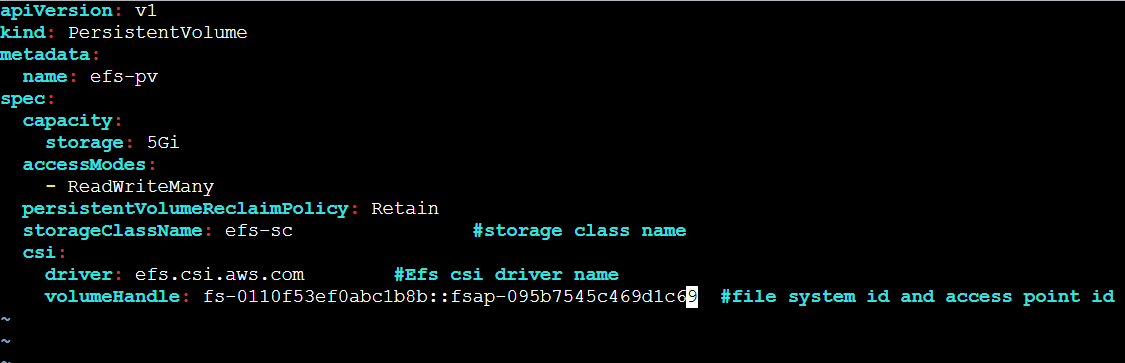
|  |
| --- |
| apiVersion: storage.k8s.io/v1  kind: StorageClass  metadata:  name: efs-sc  provisioner: efs.csi.aws.com |

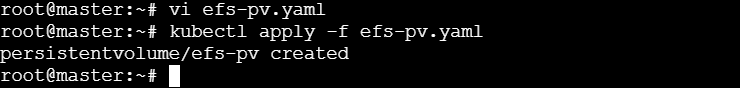
**>> kubectl apply -f storageclass.yaml #To create storage class**

****

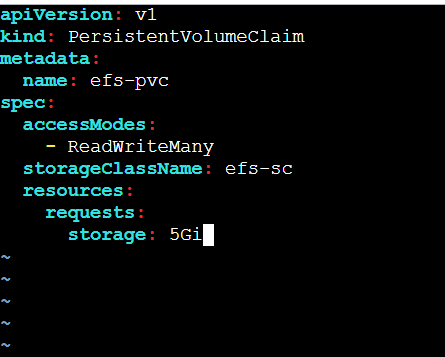
* **Now we need to create one Persistent volume for our EFS.**

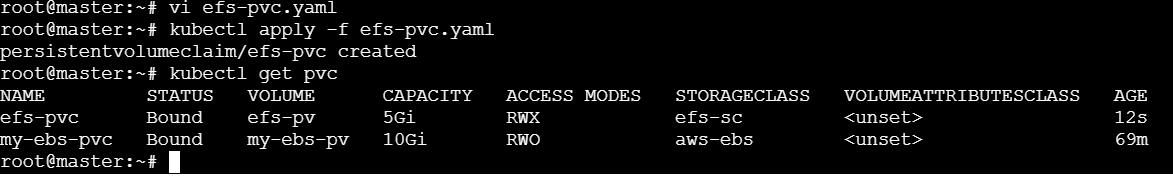
**>> vi efs-pv.yaml**

****

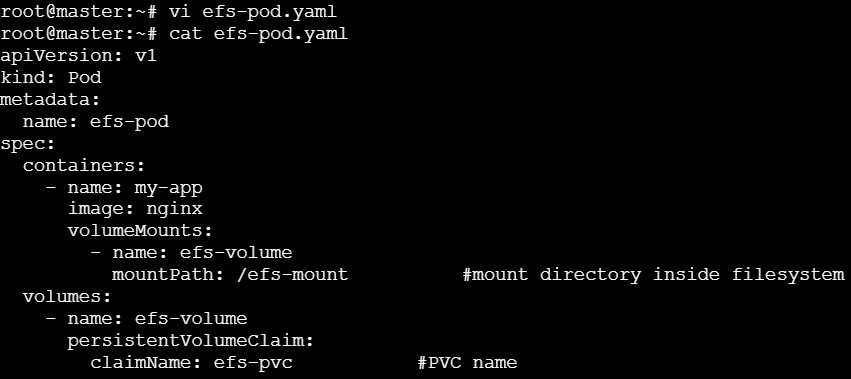
****

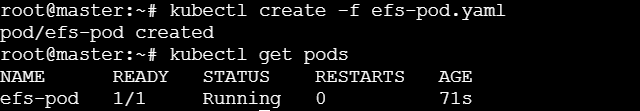
* **We need to create on claim for Persistent volume (PVC) for our EBS.**

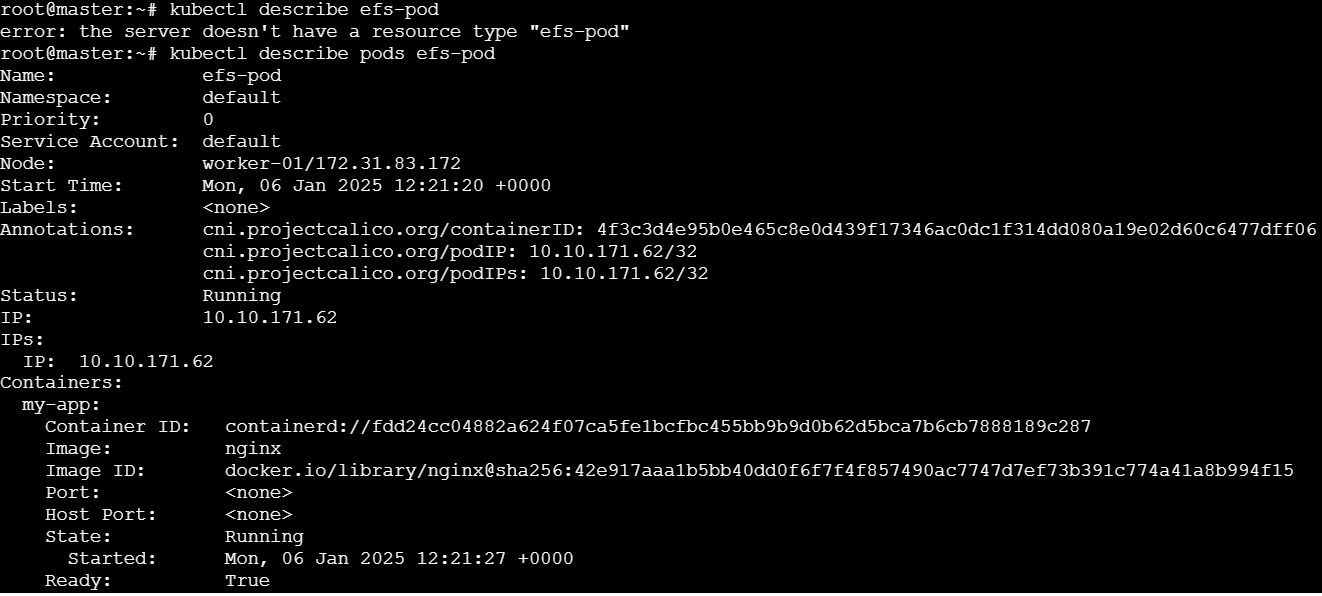
****

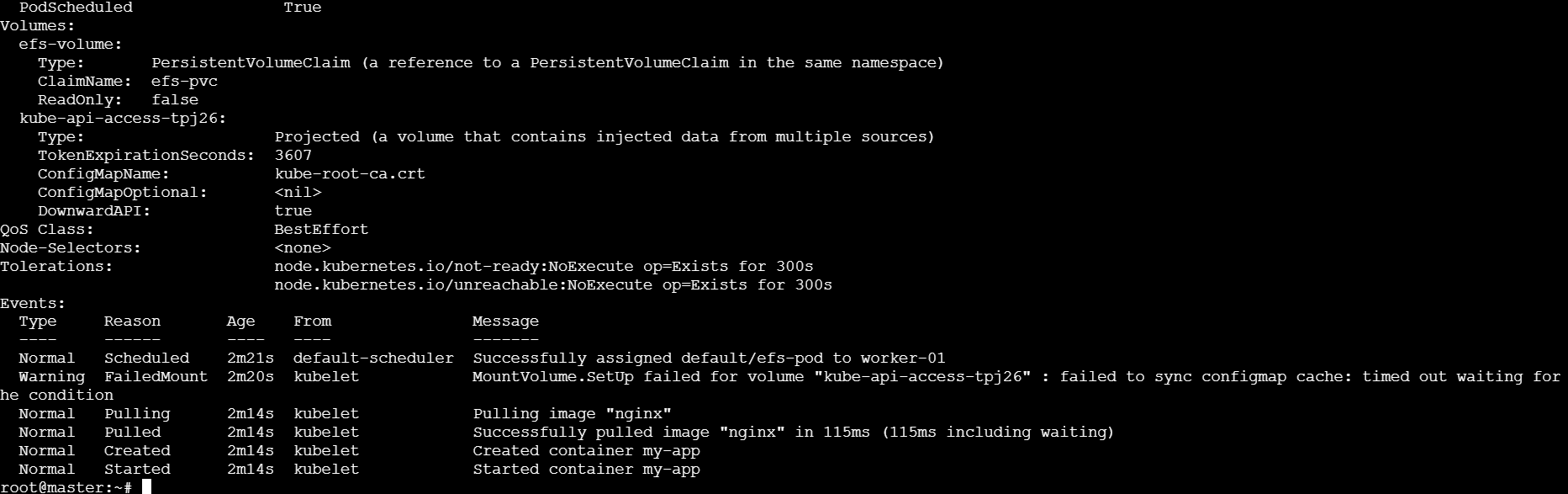
****

* **Create pod with PVC.**

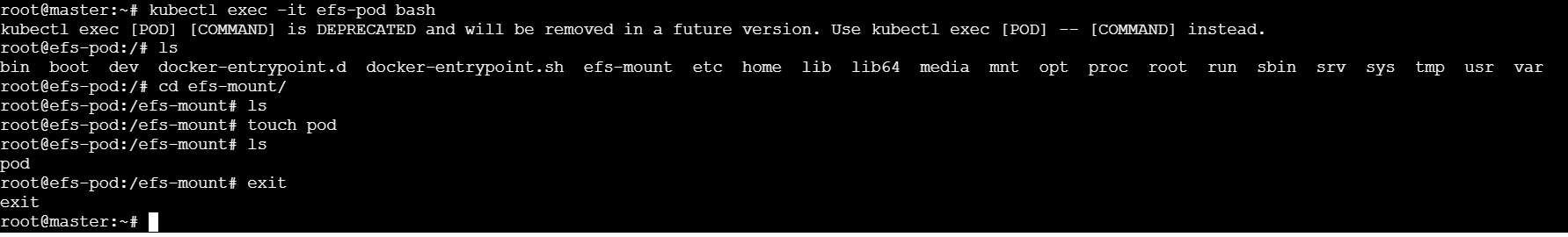
****

****

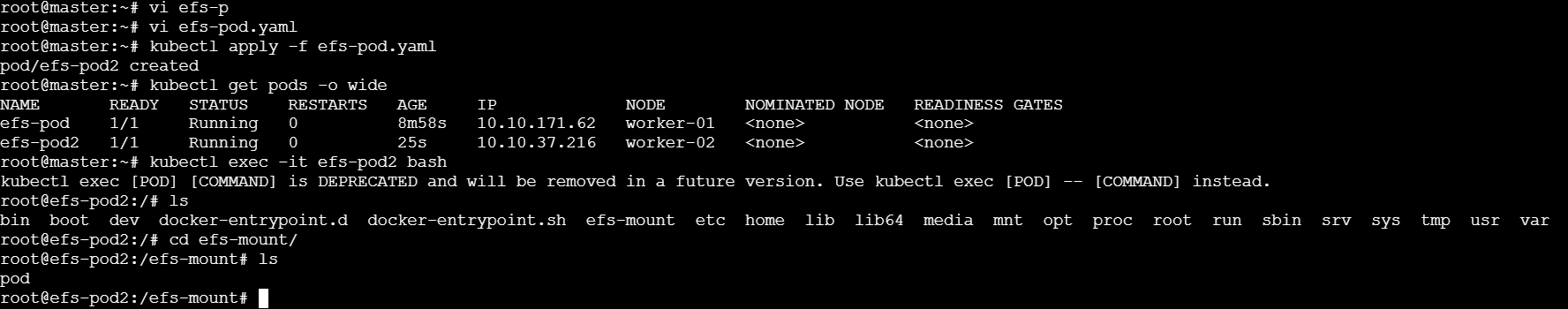
****

****

* **Let us connect to our pod**

****

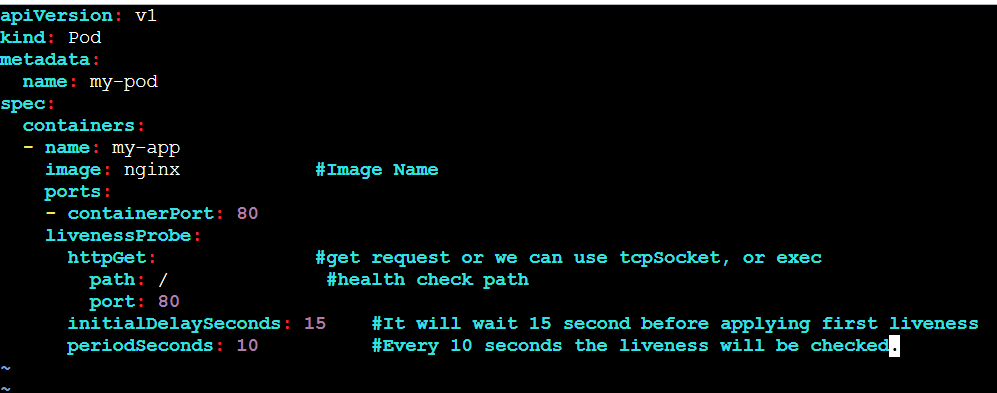
* **Now let us create another pod (efs-pod2) and check if it uses the same efs volume and does it contain the same data(pod file we created in efs-pod):**

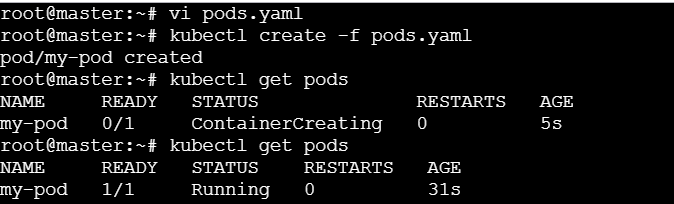
****

1. **Implement and Test Liveness and Readiness Probes in a Kubernetes Pod**

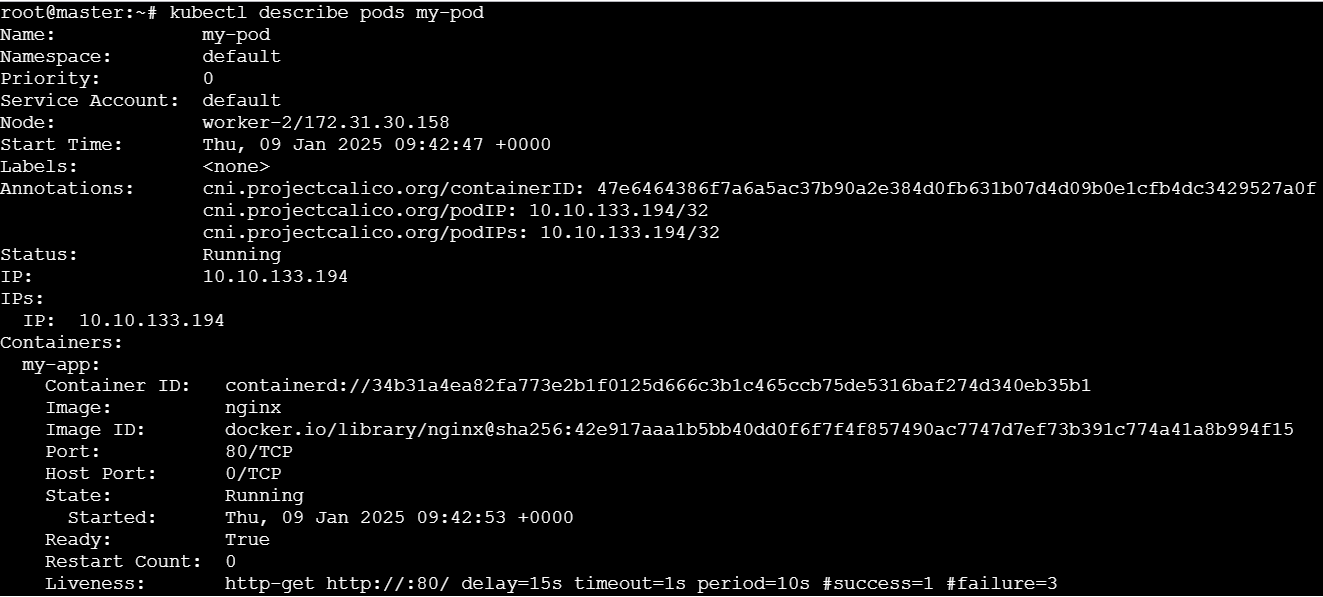
* **Liveness:**

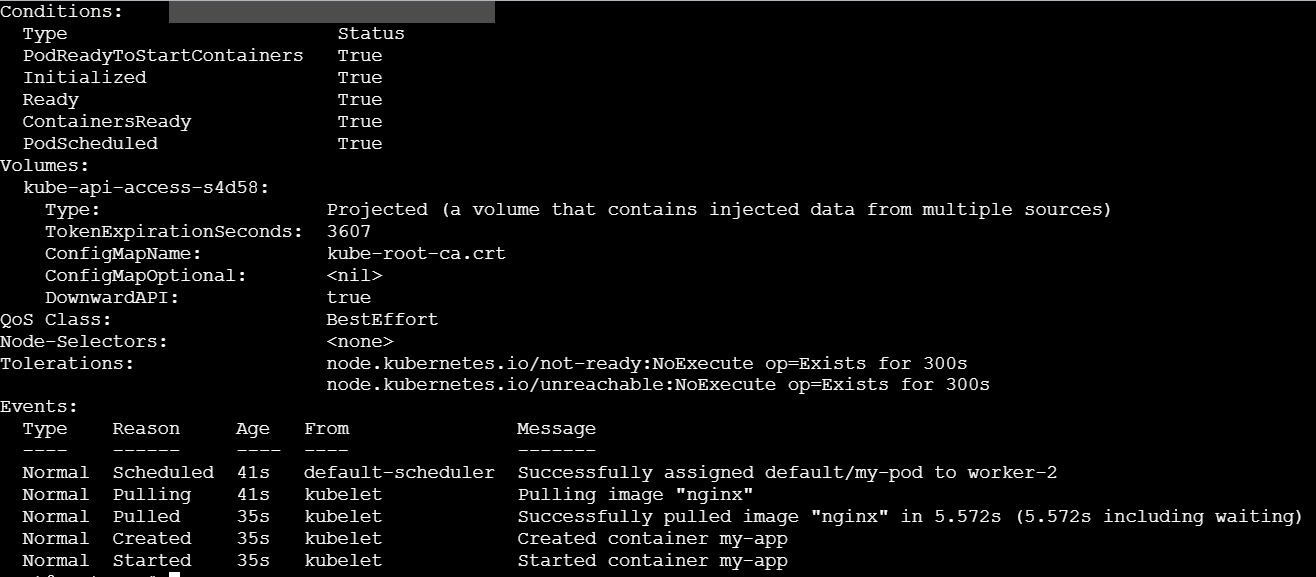
**>> vi pods.yaml**

****

****

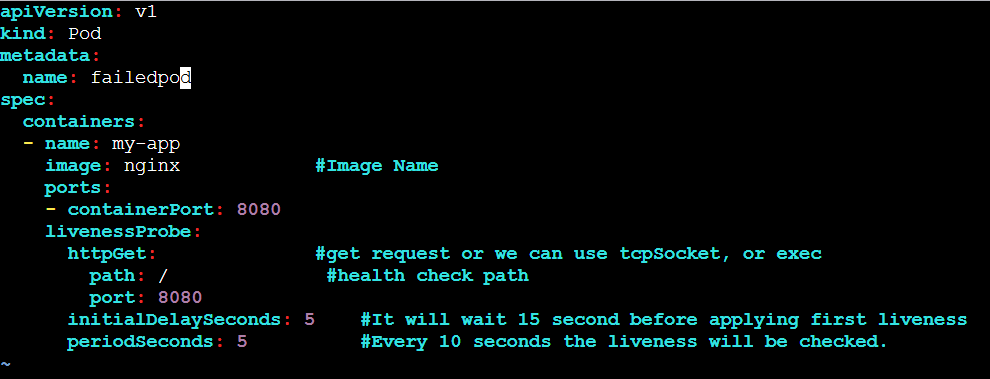
**>> kubectl describe pods my-pod**

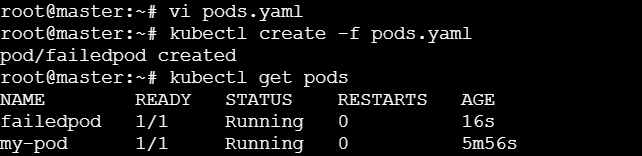
****

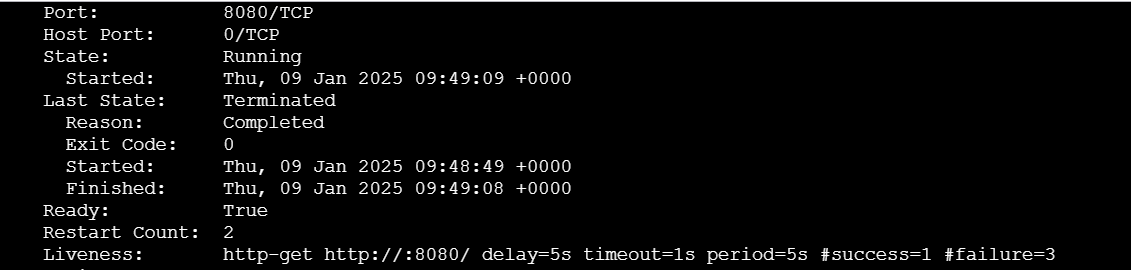
****

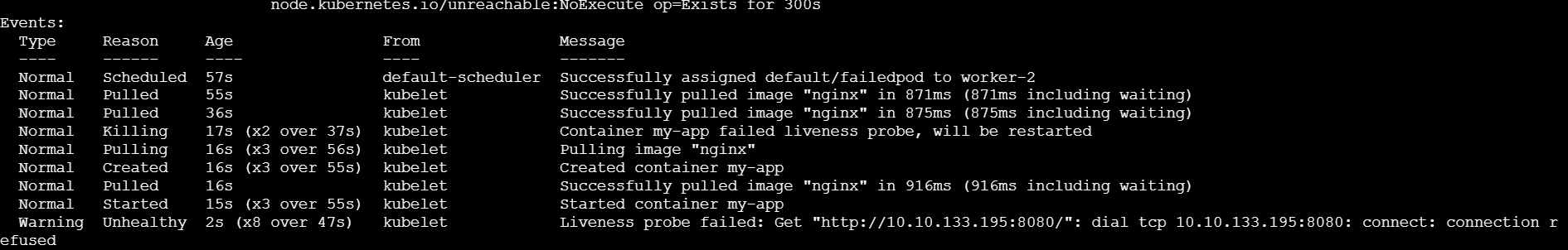
* **Changing the porst from 80 to 8080 and the name of the pod**

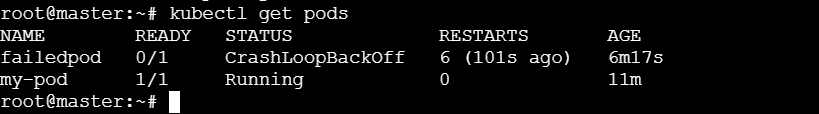
**>> vi pods.yaml**

****

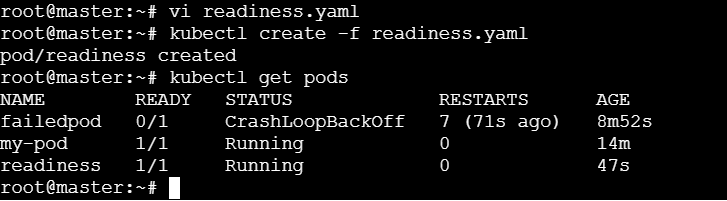
****

****

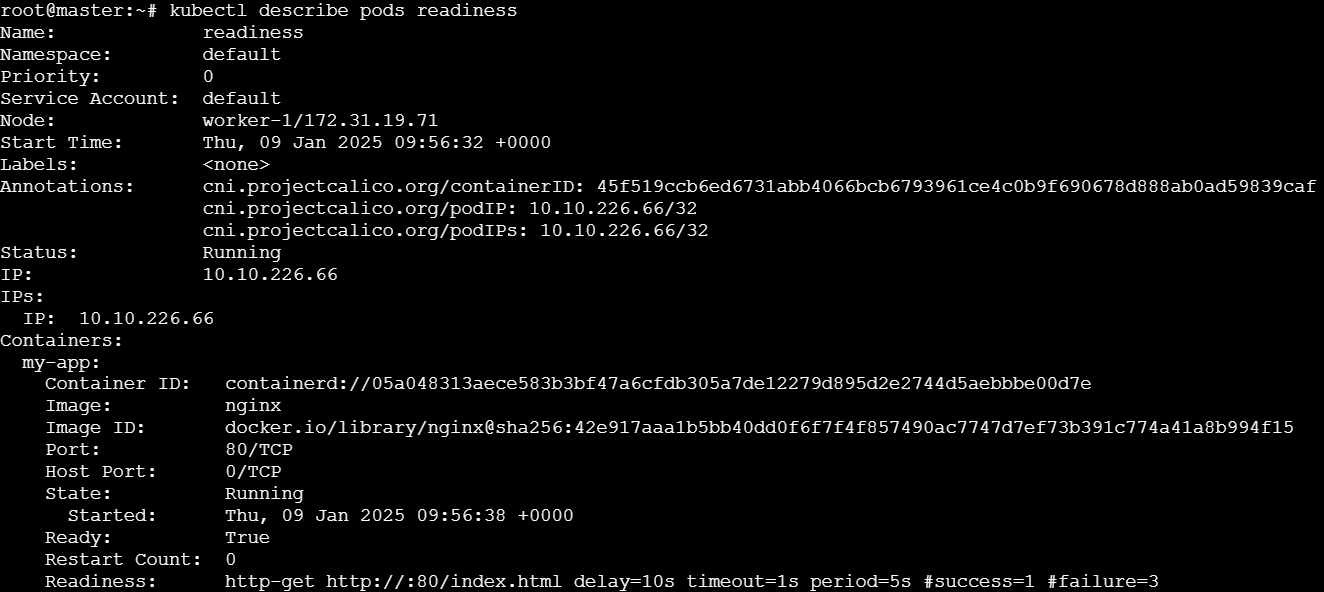
****

****

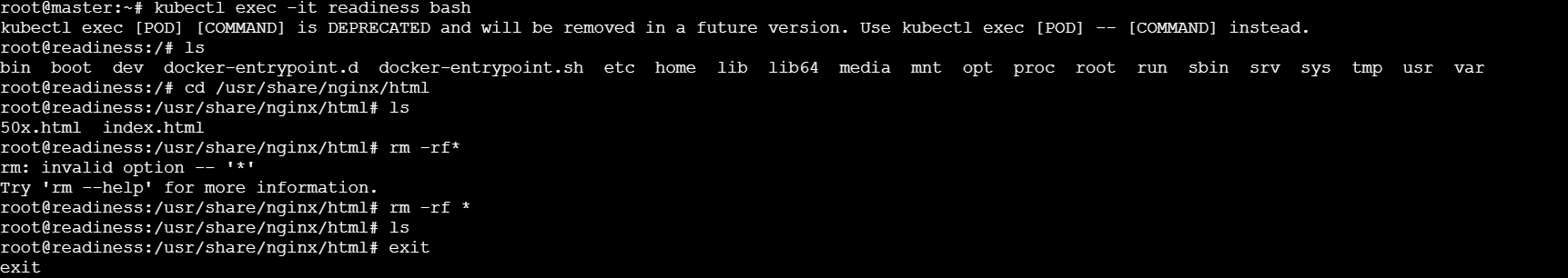
* **Readiness:**

****

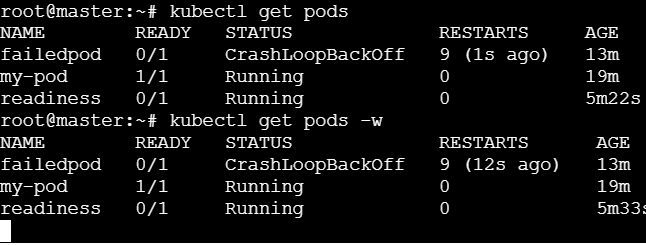
**>> kubectl describe pods readiness**

****

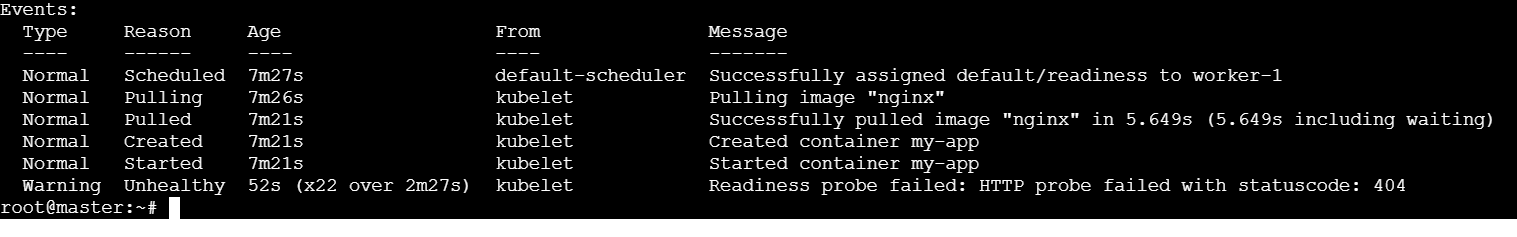
* **Now let us connect to our pod and delete the html file.**

****

* **Now the readiness of the pod fails.**

****

**>> kubectl describe pod readiness**

****