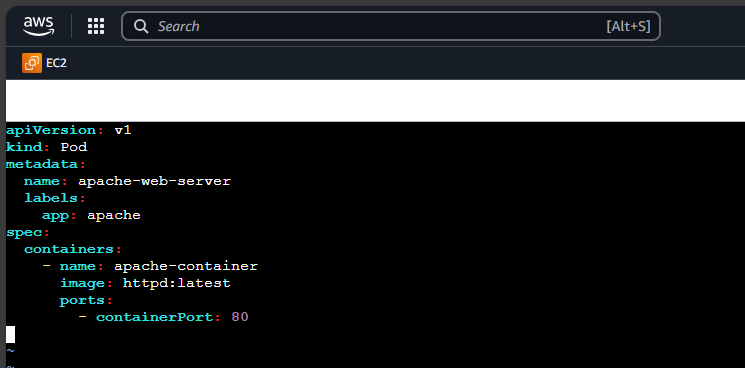
KUBERNETES[TASK-3]

1)Create a ClusterIP service for an Apache web server pod.

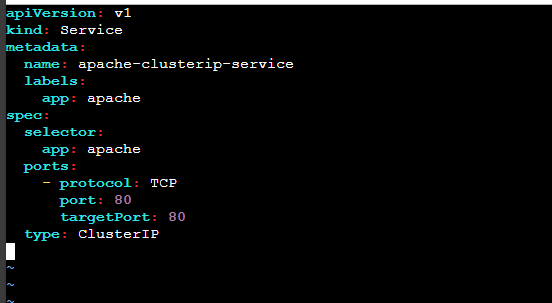
Create the Apache Web Server Pod



Apply the pod

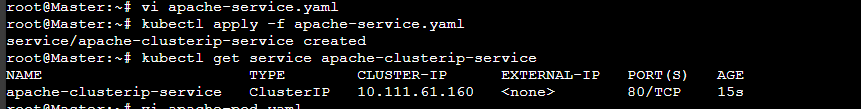


Step 2: Create the ClusterIP Service



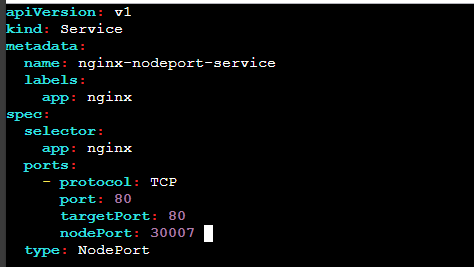
Using kubectl apply -f apply

And using the cmd: kubectl get service apache-clusterip-service



2) Expose an Nginx pod externally using a NodePort service.

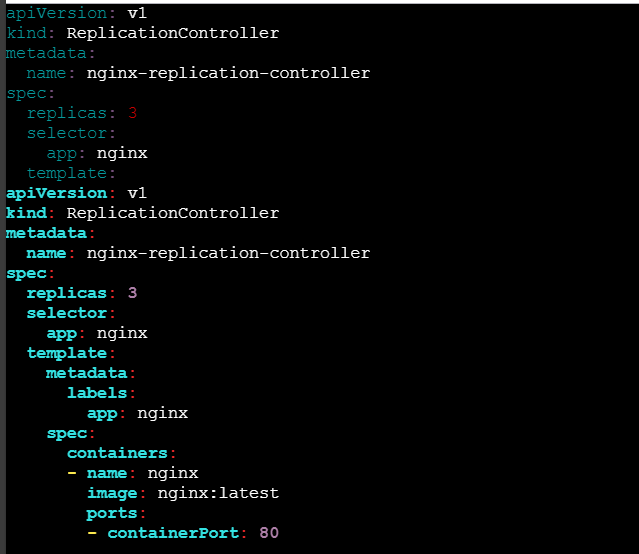
Created a file : nginx-nodeport-service.yaml



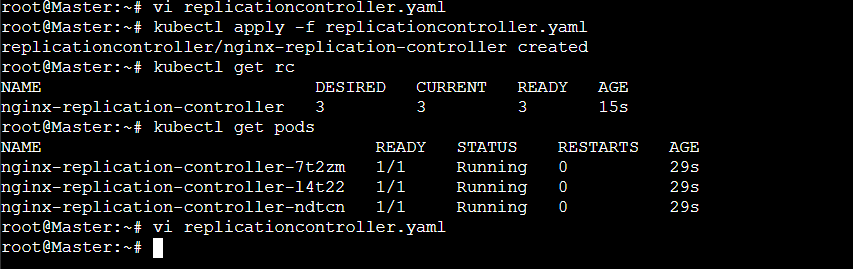


3) Deploy a ReplicationController to maintain 3 replicas of an Nginx pod.

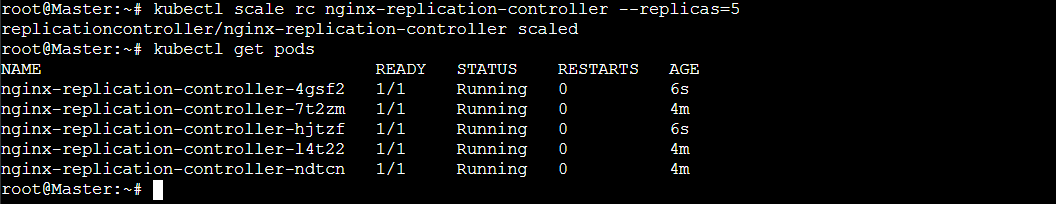
Created a file with replicationcontroller.yaml



kubectl apply -f replicationcontroller.yml



4) Scale the ReplicationController from 3 replicas to 5 replicas.



5) Create a ReplicaSet to manage pods based on multiple labels (prod and test).

Created a yaml file

apiVersion: apps/v1

kind: ReplicaSet

metadata:

name: nginx-replicaset

spec:

replicas: 3

selector:

matchLabels:

app: nginx

environment: prod

template:

metadata:

labels:

app: nginx

environment: prod

spec:

containers:

- name: nginx

image: nginx:latest

ports:

- containerPort: 80

---

apiVersion: apps/v1

kind: ReplicaSet

metadata:

name: nginx-replicaset-test

spec:

replicas: 3

selector:

matchLabels:

app: nginx

environment: test

template:

metadata:

labels:

app: nginx

environment: test

spec:

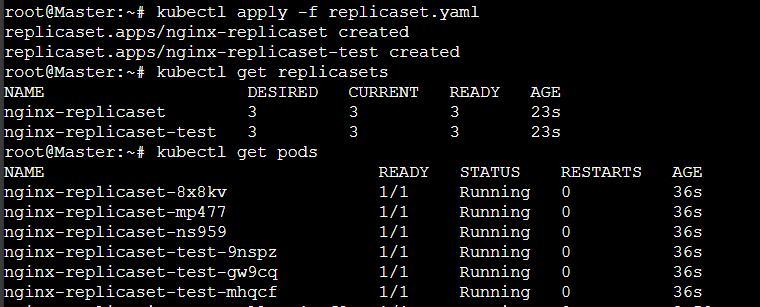
containers:

- name: nginx

image: nginx:latest

ports:

- containerPort: 80



6) Deploy a ReplicaSet that excludes pods with the label backend.

apiVersion: apps/v1

kind: ReplicaSet

metadata:

name: nginx-replicaset-no-backend

spec:

replicas: 3

selector:

matchLabels:

app: nginx

template:

metadata:

labels:

app: nginx

spec:

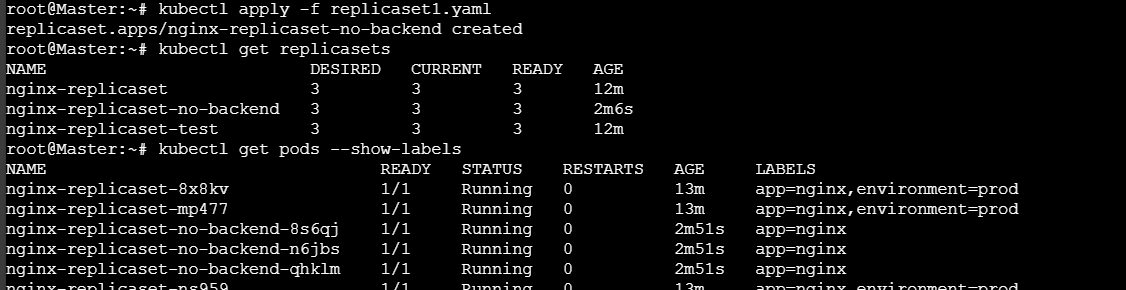
containers:

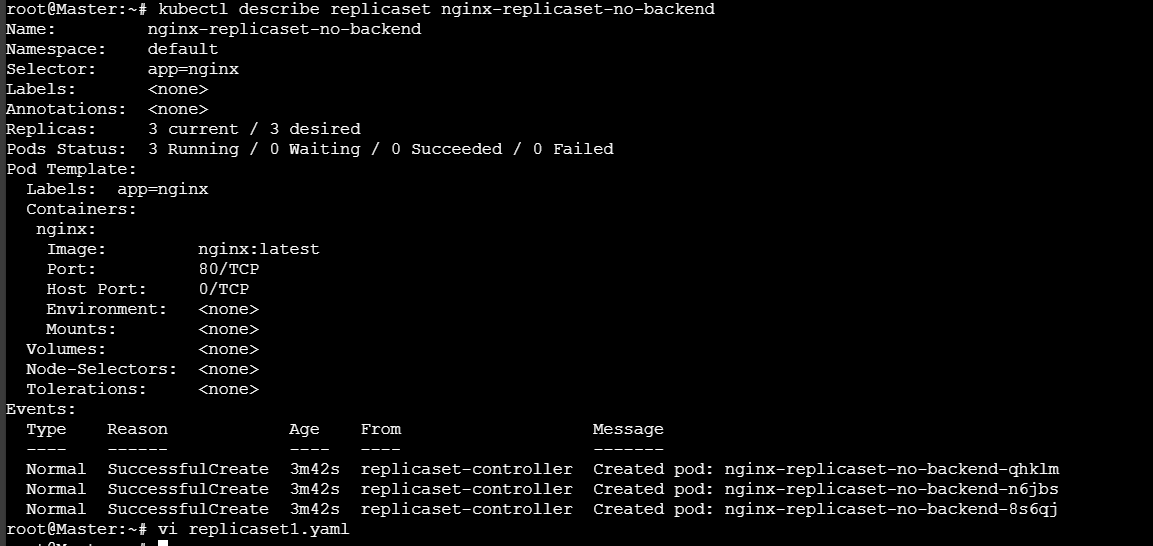
- name: nginx

image: nginx:latest

ports:

- containerPort: 80





7) Test load balancing across multiple pods using a NodePort service.

**Created file deployment.yaml**

apiVersion: apps/v1

kind: Deployment

metadata:

name: nginx-deployment

spec:

replicas: 3

selector:

matchLabels:

app: nginx

template:

metadata:

labels:

app: nginx

spec:

containers:

- name: nginx

image: nginx:latest

ports:

- containerPort: 80

**Created service.yaml**

apiVersion: v1

kind: Service

metadata:

name: nginx-service

spec:

selector:

app: nginx

ports:

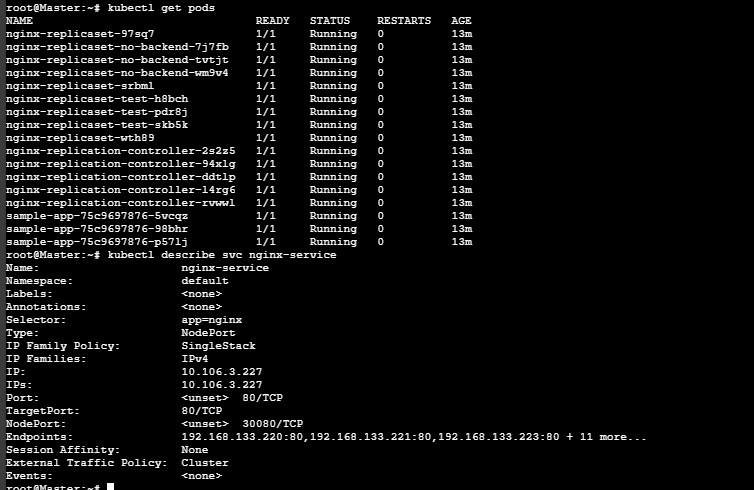
- protocol: TCP

port: 80

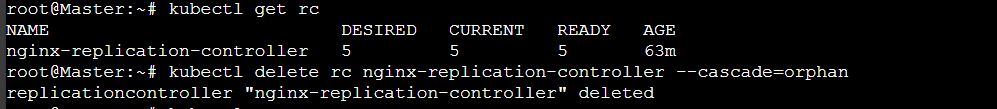
targetPort: 80

nodePort: 30080

type: NodePort



8) Delete a ReplicationController without affecting the running pods.



The --cascade=orphan option prevents the deletion of dependent resources (in this case, pods) when the ReplicationController is deleted. The pods are no longer managed by any controller but continue running.