EXPERIMENT NO. 04

Aim:

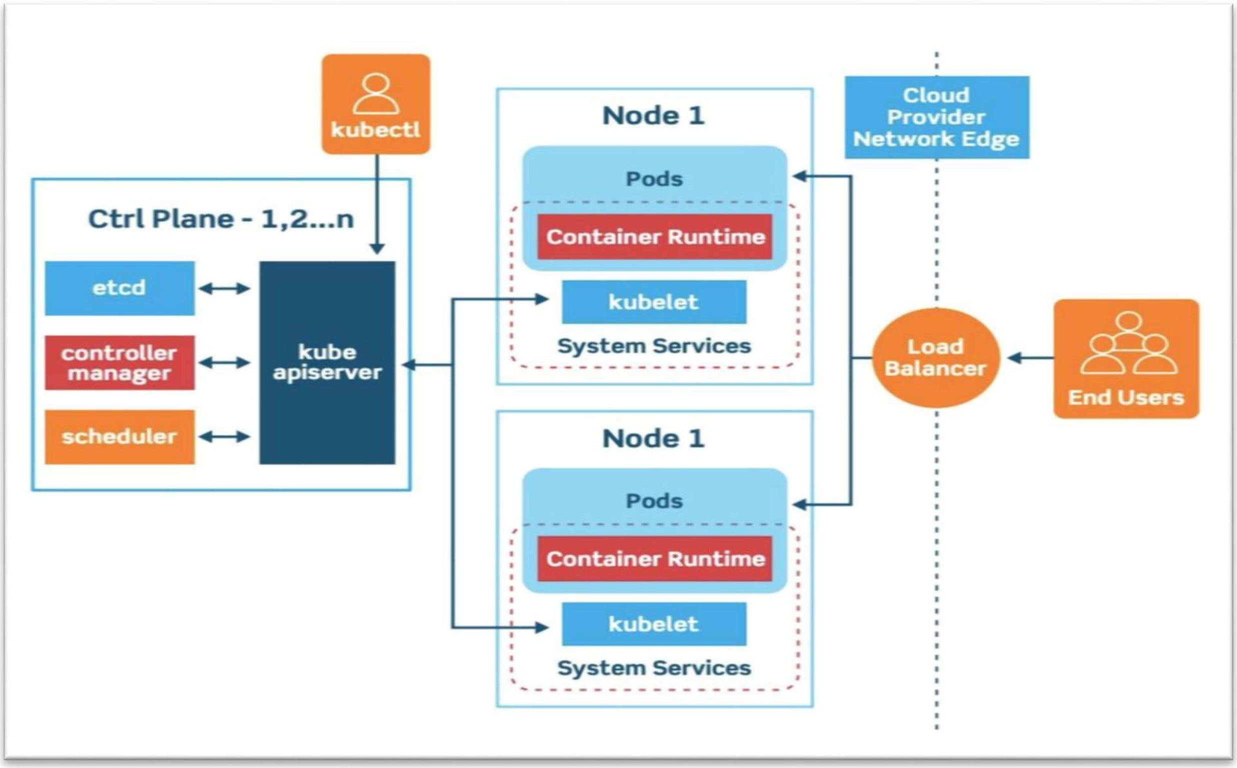
To install Kubectl and execute Kubectl commands to manage the Kubernetes cluster and deploy Your First Kubernetes Application.

Theory:

Kubernetes is an open-source platform for deploying and managing containers. It provides a container runtime, container orchestration, container-centric infrastructure orchestration, self- healing mechanisms, service discovery and load balancing. It’s used for the deployment, scaling, management, and composition of application containers across clusters of hosts.

But Kubernetes is more than just a container orchestrator. It could be thought of as the operating system for cloud-native applications in the sense that it’s the platform that applications run on, just as desktop applications run on MacOS, Windows, or Linux.

From a high level, a Kubernetes environment consists of a control plane (master), a distributed storage system for keeping the cluster state consistent (etcd), and a number of cluster nodes.



The **control plane** is the system that maintains a record of all Kubernetes objects. It continuously manages object states, responding to changes in the cluster; it also works to make the actual state of system objects match the desired state. As the above illustration shows, the control plane is made up of three major components:

- kube-apiserver
- kube-controller-manager
- kube-scheduler

Cluster nodes are machines that run containers and are managed by the master nodes. The Kubelet is the primary and most important controller in Kubernetes. It's responsible for driving the container execution layer, typically Docker.

Pods are one of the crucial concepts in Kubernetes, as they are the key construct that developers interact with. The previous concepts are infrastructure-focused and internal architecture

Kubernetes Tooling and Clients:

Here are the basic tools you should know:

1. **Kubeadm** bootstraps a cluster. It's designed to be a simple way for new users to build clusters (more detail on this is in a later chapter).
2. **Kubectl** is a tool for interacting with your existing cluster.
3. **Minikube** is a tool that makes it easy to run Kubernetes locally. For Mac users, HomeBrew makes using Minikube even simpler.

- **KUBERNETES DEPLOYMENT**

A Kubernetes deployment is a resource object in Kubernetes that provides declarative updates to applications. A deployment allows you to describe an application's life cycle, such as which images to use for the app, the number of pods there should be, and the way in which they should be updated.

A Kubernetes object is a way to tell the Kubernetes system how you want your cluster's workload to look. After an object has been created, the cluster works to ensure that the object exists, maintaining the desired state of your Kubernetes cluster.

The process of manually updating containerized applications can be time consuming and tedious. Upgrading a service to the next version requires starting the new version of the pod, stopping the old version of a pod, waiting and verifying that the new version has launched successfully, and sometimes rolling it all back to a previous version in the case of failure.

Performing these steps manually can lead to human errors, and scripting properly can require a significant amount of effort, both of which can turn the release process into a bottleneck.

A Kubernetes deployment makes this process automated and repeatable. Deployments are entirely managed by the Kubernetes backend, and the whole update process is performed on the server side without client interaction.

A deployment ensures the desired number of pods are running and available at all times. The update process is also wholly recorded, and versioned with options to pause, continue, and roll back to previous versions.

Automate deployments with pre-made, repeatable Kubernetes patterns

The Kubernetes deployment object lets you:

- Deploy a replica set or pod
- Update pods and replica sets
- Rollback to previous deployment versions
- Scale a deployment
- Pause or continue a deployment

Pre-requisite:

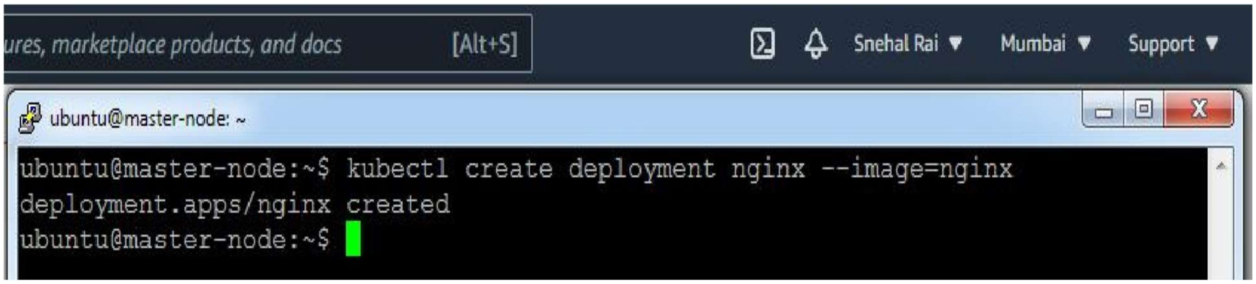
- Launch Linux server running Ubuntu 18.04 /20.04 on Virtual box

OR

- Launch EC2 instances of Ubuntu 20.04 AMI free tier.

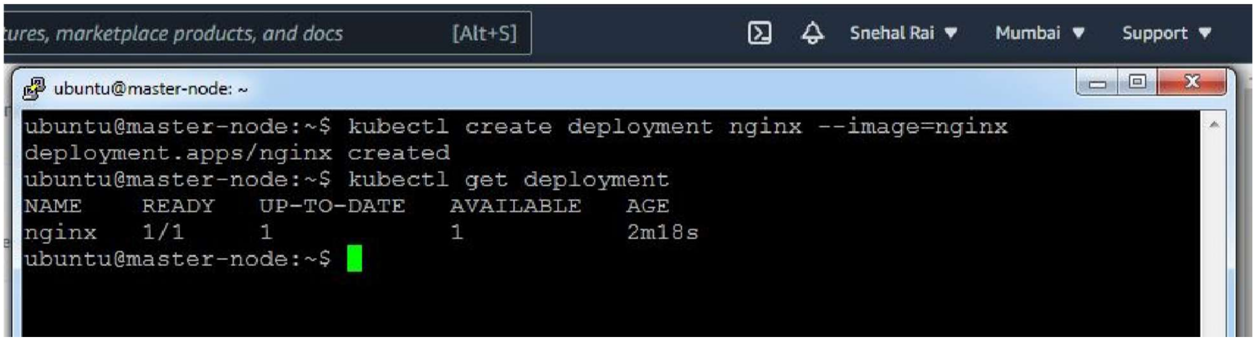
Steps:

Step 1: To create nginx run the command kubectl create deployment nginx –image=nginx



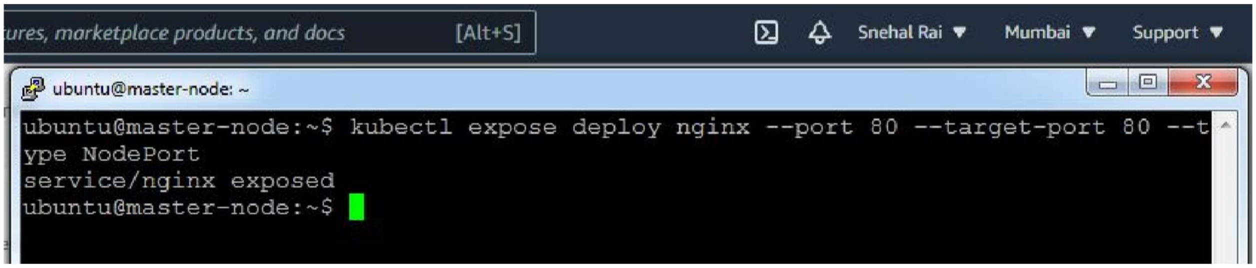
```
ubuntu@master-node:~$ kubectl create deployment nginx --image=nginx
deployment.apps/nginx created
ubuntu@master-node:~$
```

Step 2: To deploy nginx run the command kubectl get deployment



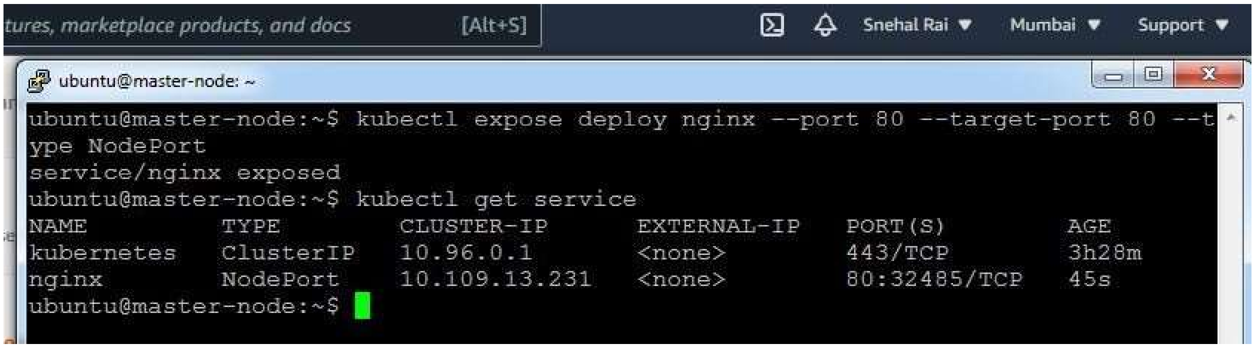
```
ubuntu@master-node:~$ kubectl create deployment nginx --image=nginx
deployment.apps/nginx created
ubuntu@master-node:~$ kubectl get deployment
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
nginx   1/1     1            1           2m18s
ubuntu@master-node:~$
```

Step 3: Run the command kubectl expose deploy nginx –-port 80 –target-port 80 –type NodePort



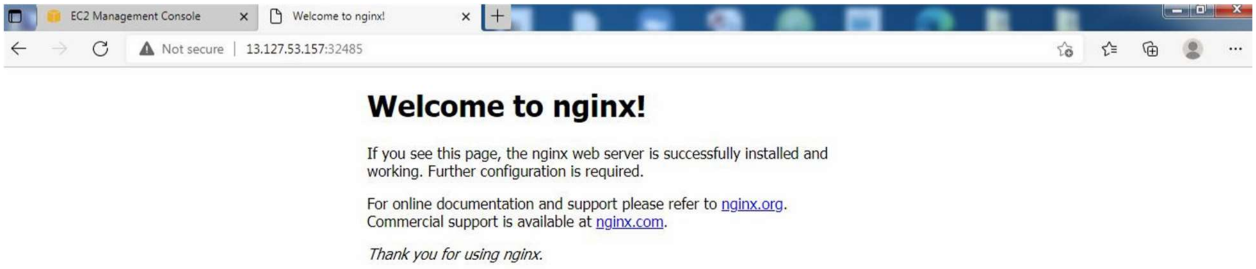
```
ubuntu@master-node:~$ kubectl expose deploy nginx --port 80 --target-port 80 --t
ype NodePort
service/nginx exposed
ubuntu@master-node:~$
```

Step 4: To know thw service run the command kubectl get service



```
ubuntu@master-node:~$ kubectl expose deploy nginx --port 80 --target-port 80 --t
ype NodePort
service/nginx exposed
ubuntu@master-node:~$ kubectl get service
NAME         TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
kubernetes   ClusterIP   10.96.0.1       <none>        443/TCP        3h28m
nginx        NodePort    10.109.13.231   <none>        80:32485/TCP   45s
ubuntu@master-node:~$
```

Step 5: Nginx will open in the browser



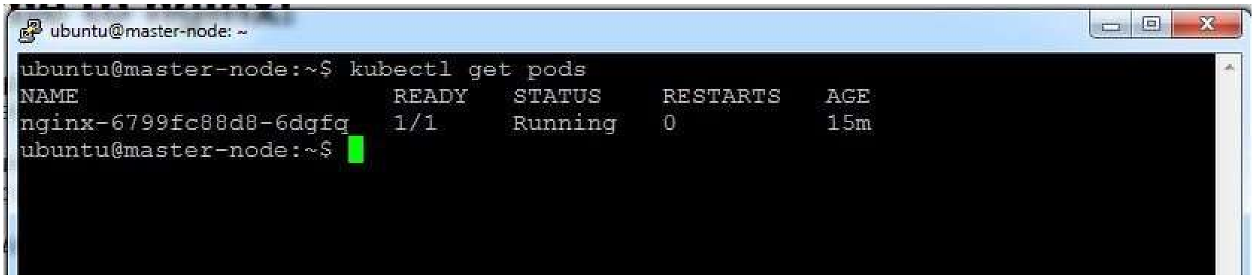
Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org. Commercial support is available at nginx.com.

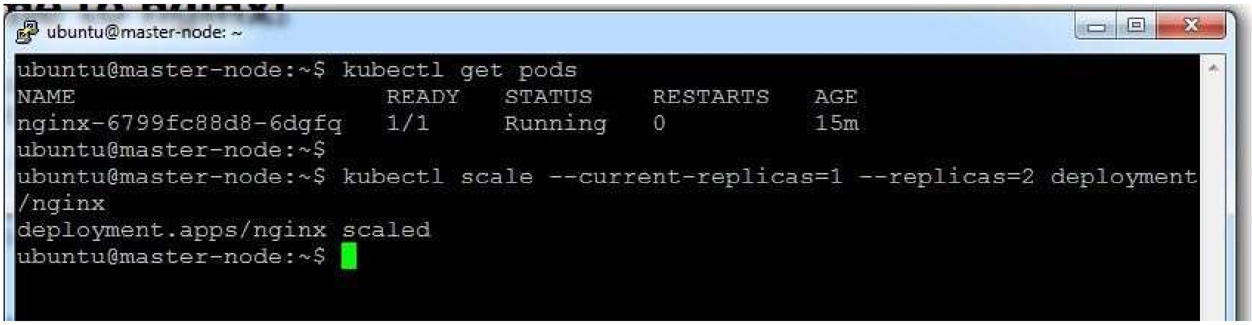
Thank you for using nginx.

Step 6: Run the command kubectl get pods



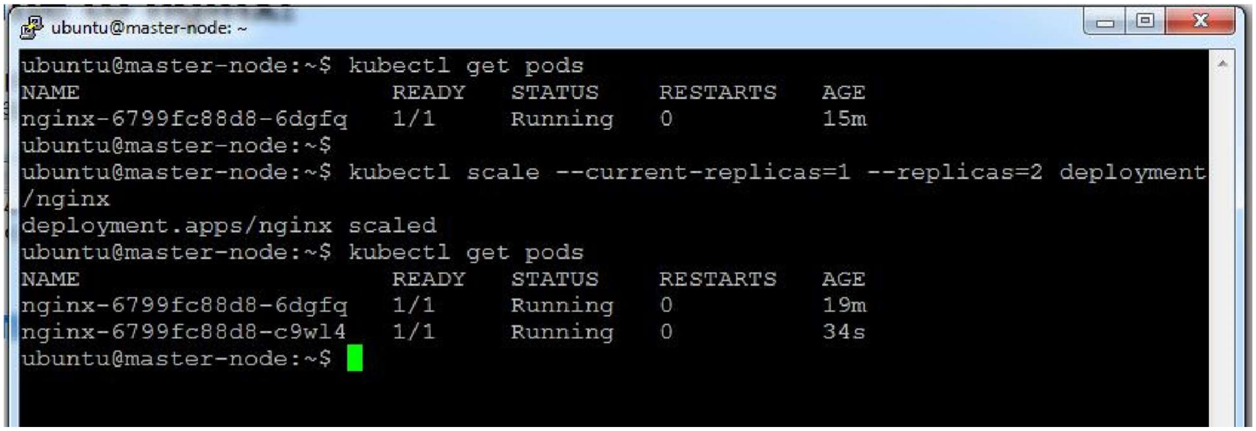
```
ubuntu@master-node:~$ kubectl get pods
NAME                     READY   STATUS    RESTARTS   AGE
nginx-6799fc88d8-6dgfq   1/1     Running   0          15m
ubuntu@master-node:~$
```

Step 7: To scale nginx run the command kubectl scale ---current-replicas=2 deployment/nginx



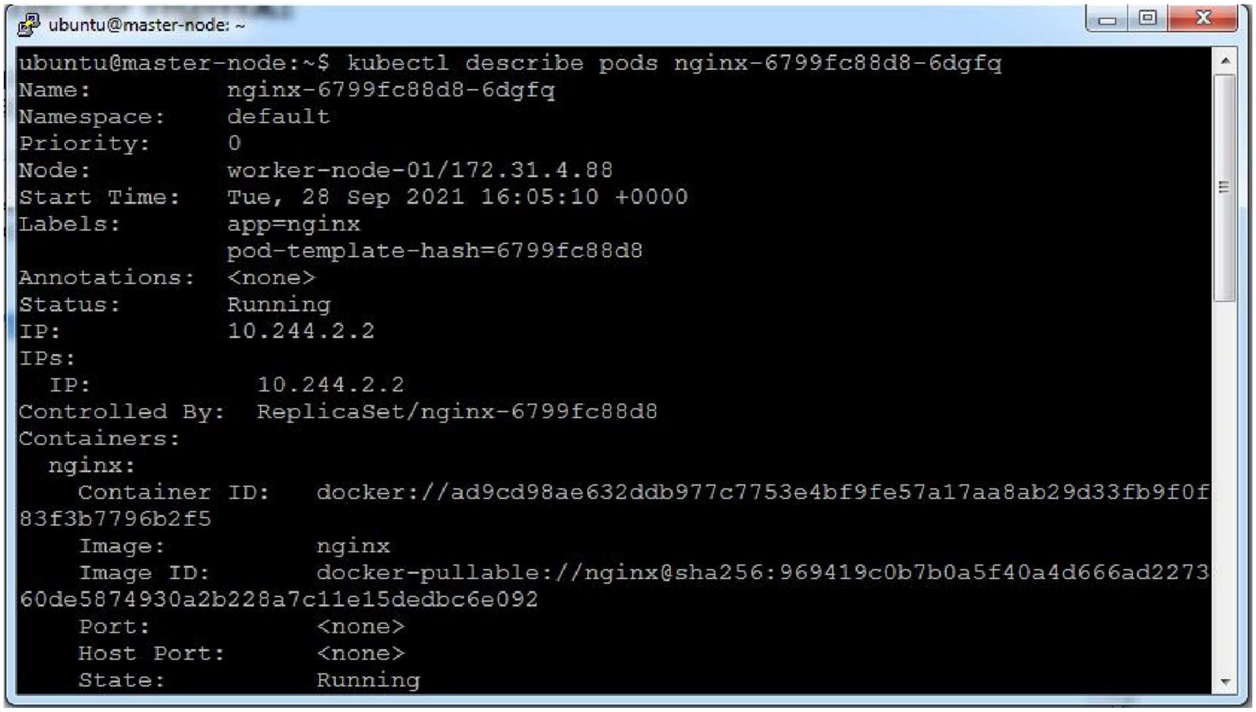
```
ubuntu@master-node:~$ kubectl get pods
NAME                     READY   STATUS    RESTARTS   AGE
nginx-6799fc88d8-6dgfq   1/1     Running   0          15m
ubuntu@master-node:~$
ubuntu@master-node:~$ kubectl scale --current-replicas=1 --replicas=2 deployment
/nginx
deployment.apps/nginx scaled
ubuntu@master-node:~$
```

Step 8: To see the status again run the command kubectl get pods



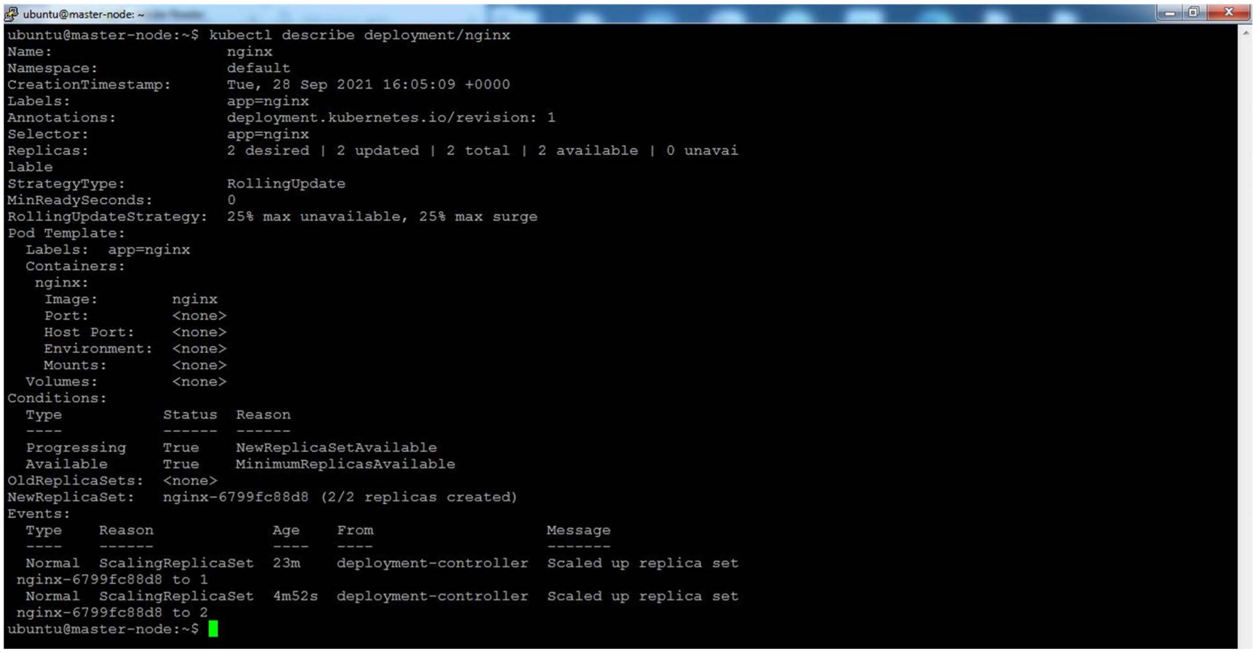
```
ubuntu@master-node:~$ kubectl get pods
NAME                     READY   STATUS    RESTARTS   AGE
nginx-6799fc88d8-6dgfq   1/1     Running   0          15m
ubuntu@master-node:~$
ubuntu@master-node:~$ kubectl scale --current-replicas=1 --replicas=2 deployment
/nginx
deployment.apps/nginx scaled
ubuntu@master-node:~$ kubectl get pods
NAME                     READY   STATUS    RESTARTS   AGE
nginx-6799fc88d8-6dgfq   1/1     Running   0          19m
nginx-6799fc88d8-c9wl4   1/1     Running   0          34s
ubuntu@master-node:~$
```

Step 9: For nginx pod description run the command kubectl describe pods nginx – 6799fc88d86dgfq



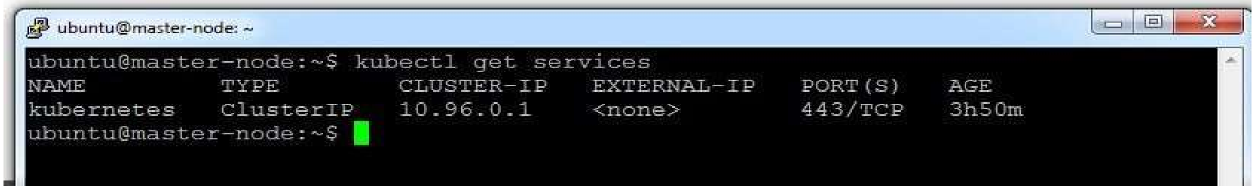
```
ubuntu@master-node:~$ kubectl describe pods nginx-6799fc88d8-6dgfq
Name:           nginx-6799fc88d8-6dgfq
Namespace:      default
Priority:       0
Node:           worker-node-01/172.31.4.88
Start Time:     Tue, 28 Sep 2021 16:05:10 +0000
Labels:         app=nginx
                pod-template-hash=6799fc88d8
Annotations:    <none>
Status:         Running
IP:             10.244.2.2
IPs:
  IP:           10.244.2.2
Controlled By:  ReplicaSet/nginx-6799fc88d8
Containers:
  nginx:
    Container ID:   docker://ad9cd98ae632ddb977c7753e4bf9fe57a17aa8ab29d33fb9f0f
83f3b7796b2f5
    Image:          nginx
    Image ID:       docker-pullable://nginx@sha256:969419c0b7b0a5f40a4d666ad2273
60de5874930a2b228a7c11e15dedbc6e092
    Port:           <none>
    Host Port:      <none>
    State:          Running
```

Step 10: For nginx deployment description run the command kubectl describe deployment/nginx

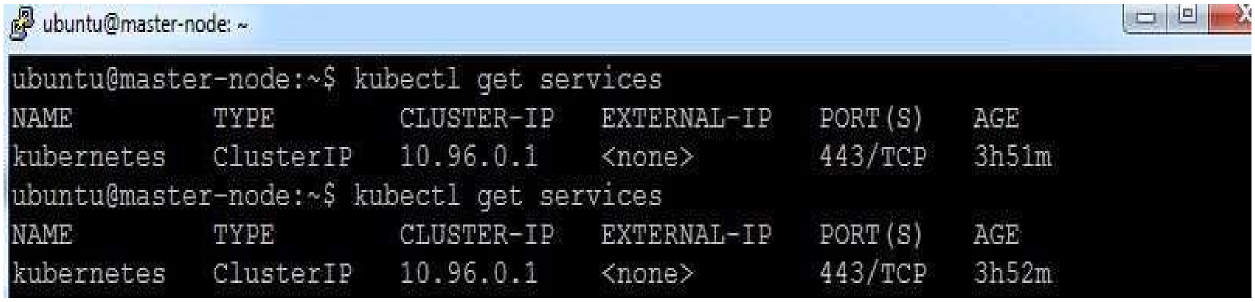


```
ubuntu@master-node:~$ kubectl describe deployment/nginx
Name:                   nginx
Namespace:              default
CreationTimestamp:      Tue, 28 Sep 2021 16:05:09 +0000
Labels:                 app=nginx
Annotations:            deployment.kubernetes.io/revision: 1
Selector:               app=nginx
Replicas:               2 desired | 2 updated | 2 total | 2 available | 0 unavai
lable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=nginx
  Containers:
   nginx:
    Image:        nginx
    Port:         <none>
    Host Port:    <none>
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Progressing    True    NewReplicaSetAvailable
  Available      True    MinimumReplicasAvailable
OldReplicaSets:  <none>
NewReplicaSet:   nginx-6799fc88d8 (2/2 replicas created)
Events:
  Type    Reason             Age    From                   Message
  ----    ------             ----   ----                   -------
  Normal  ScalingReplicaSet  23m    deployment-controller  Scaled up replica set
 nginx-6799fc88d8 to 1
  Normal  ScalingReplicaSet  4m52s  deployment-controller  Scaled up replica set
 nginx-6799fc88d8 to 2
ubuntu@master-node:~$
```

Step 11: Run the command kubectl get services

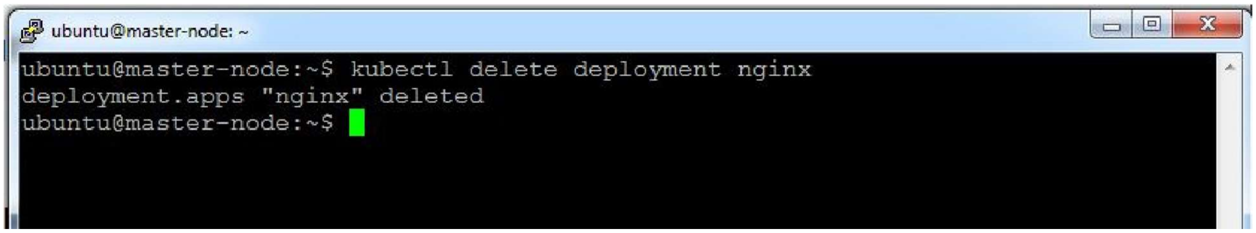


```
ubuntu@master-node:~$ kubectl get services
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   3h50m
ubuntu@master-node:~$ 
```

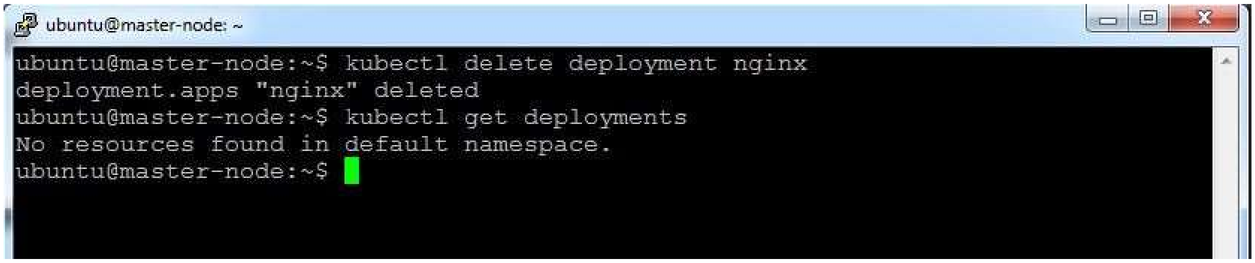


```
ubuntu@master-node:~$ kubectl get services
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   3h51m
ubuntu@master-node:~$ kubectl get services
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   3h52m
```

Step 12 : To delete the deployment of nginx run the command kubectl delete deployment nginx



```
ubuntu@master-node:~$ kubectl delete deployment nginx
deployment.apps "nginx" deleted
ubuntu@master-node:~$ 
```



```
ubuntu@master-node:~$ kubectl delete deployment nginx
deployment.apps "nginx" deleted
ubuntu@master-node:~$ kubectl get deployments
No resources found in default namespace.
ubuntu@master-node:~$ 
```

Conclusion:

Kubernetes is an open-source container orchestration platform. It provides a complete platform for scaling and managing applications that are deployed in containers. NGINX provides a suite of products which run within Kubernetes environments: Continuous development and integration, the rapid deployment and elasticity of containers and cloud services, and breaking our applications into interconnected microservices are emerging as the new normal.NGINX perform HTTP routing, directing each request to the appropriate server as defined by policies that refer to values in the Host header and URI, followed up by load balancing, health checks, and session persistence.