

Assignment–7

Module-10: Container Orchestration **Tool - Kubernetes**

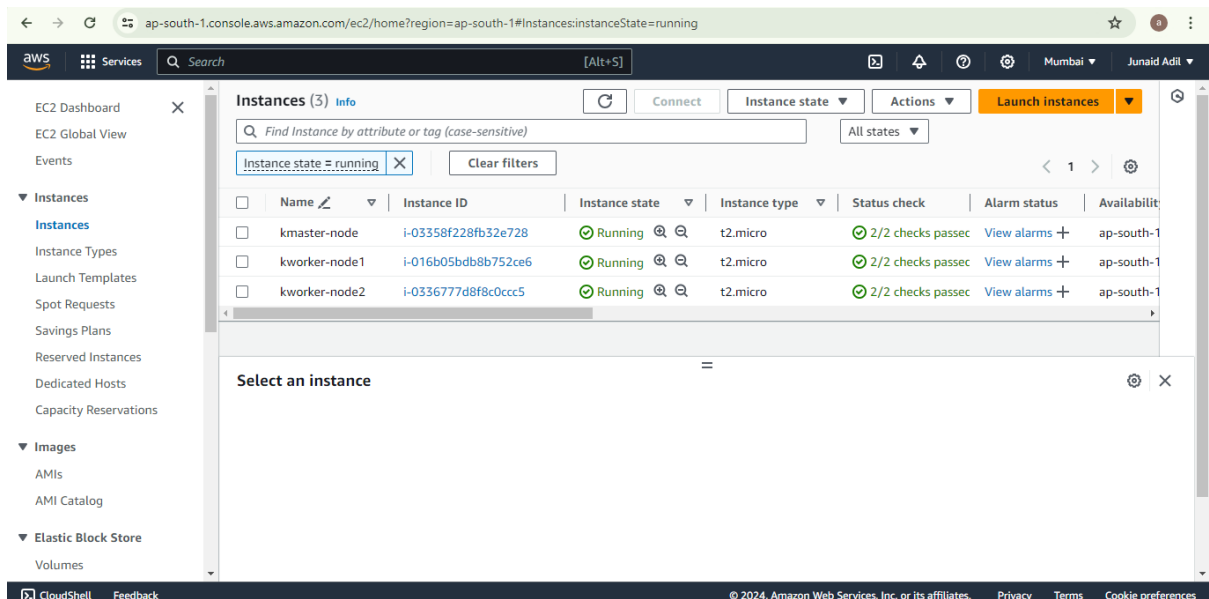
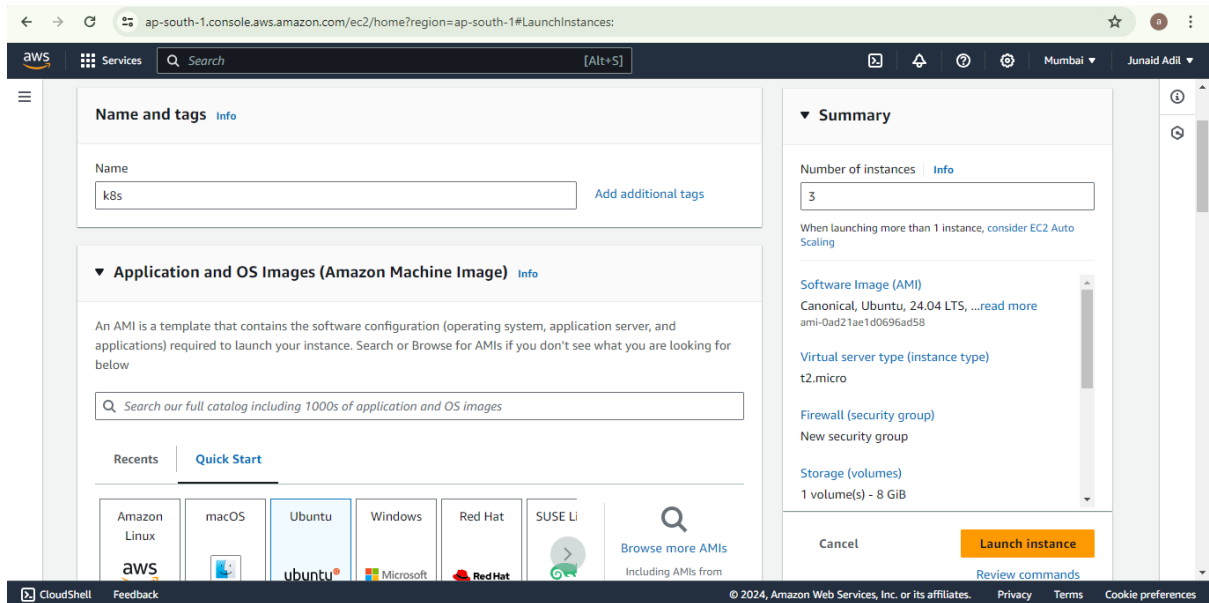
Submitted by : Shaik Junaid Adil

Date of Submission: 11-07-2024

Submitted to: Vikul

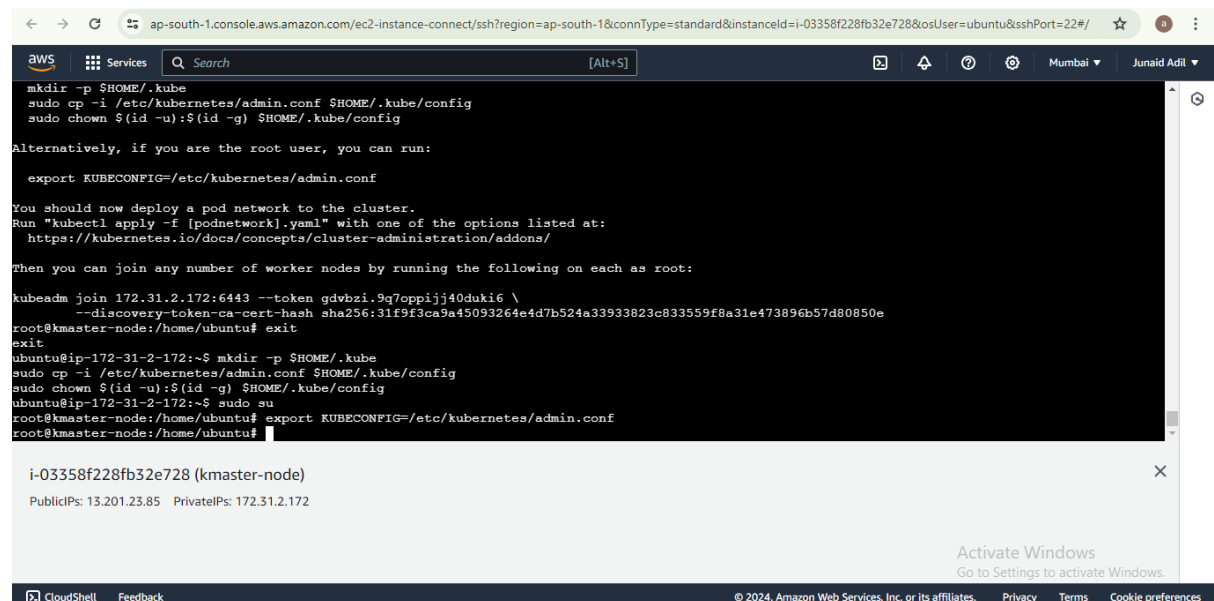
L1 - Create Deployment Controller Object to Deploy the Application Image Created in Docker Module and Expose it to the Internet

Step 1: Create 3 Instances, 1 for Master node, and 2 for worker nodes.



Step 2: Install Kubernetes and all the pre requisites in master node and worker nodes.

Master node



```
aws
Services Search [Alt+S]
Mumbai Junaid Adil

mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

Alternatively, if you are the root user, you can run:

export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

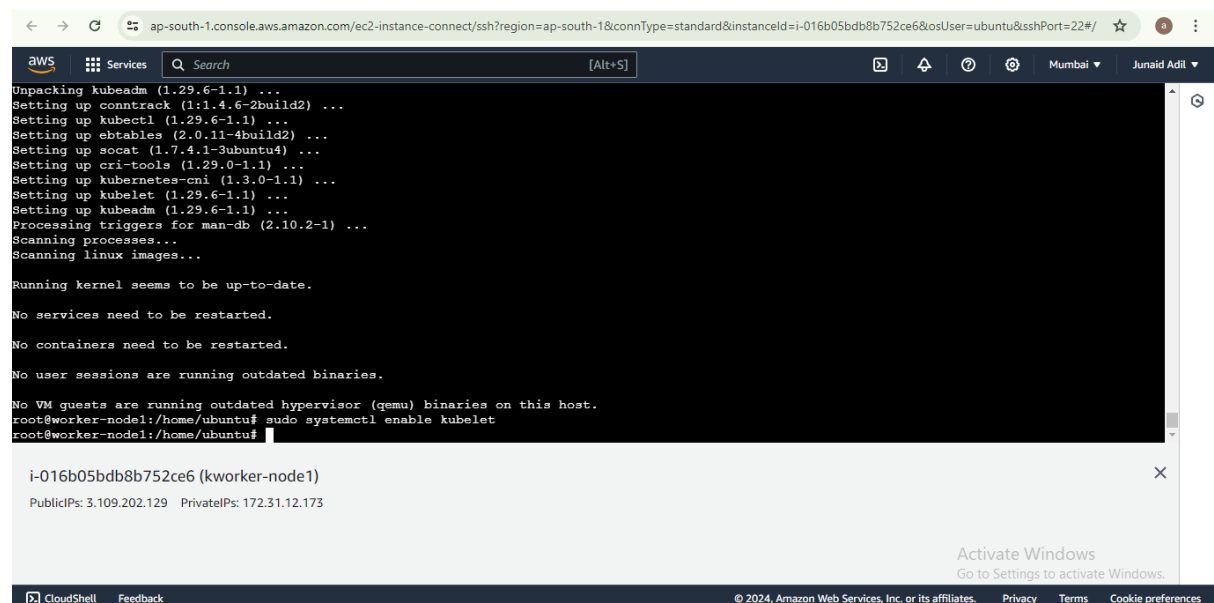
kubeadm join 172.31.2.172:6443 --token gdvbzi.9q7oppijj40duki6 \
--discovery-token-ca-cert-hash sha256:31f9f3ca9a45093264e4d7b524a33933823c833559f8a31e473896b57d80850e
root@kmaster-node:/home/ubuntu# exit
exit
ubuntu@ip-172-31-2-172:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-2-172:~$ sudo su
root@kmaster-node:/home/ubuntu# export KUBECONFIG=/etc/kubernetes/admin.conf
root@kmaster-node:/home/ubuntu#
```

i-03358f228fb32e728 (kmaster-node)
PublicIPs: 13.201.23.85 PrivateIPs: 172.31.2.172

Activate Windows
Go to Settings to activate Windows.

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Worker node1



```
aws
Services Search [Alt+S]
Mumbai Junaid Adil

Unpacking kubeadm (1.29.6-1.1) ...
Setting up conntrack (1:1.4.6-2build2) ...
Setting up kubectl (1.29.6-1.1) ...
Setting up ebtables (2.0.11-4build2) ...
Setting up socat (1.7.4.1-3ubuntu4) ...
Setting up cri-tools (1.29.0-1.1) ...
Setting up kubernetes-cni (1.3.0-1.1) ...
Setting up kubelet (1.29.6-1.1) ...
Setting up kubeadm (1.29.6-1.1) ...
Processing triggers for man-db (2.10.2-1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

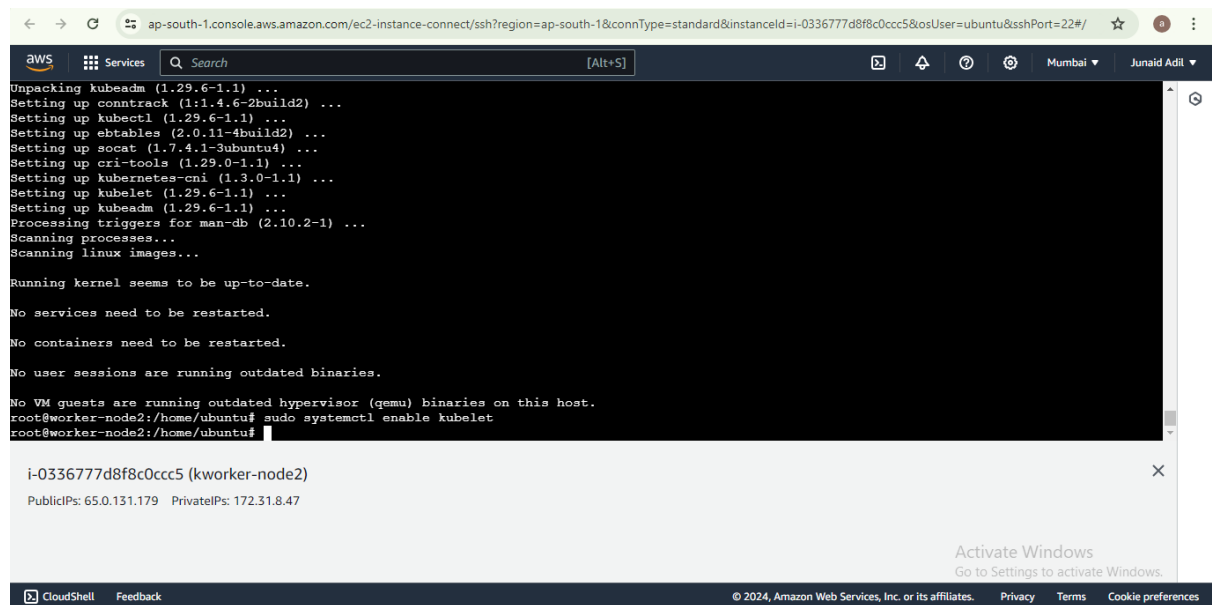
No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@worker-node1:/home/ubuntu# sudo systemctl enable kubelet
root@worker-node1:/home/ubuntu#
```

i-016b05bdb8b752ce6 (kworker-node1)
PublicIPs: 3.109.202.129 PrivateIPs: 172.31.12.173

Activate Windows
Go to Settings to activate Windows.

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Worker node2



```
Unpacking kubeadm (1.29.6-1.1) ...
Setting up conntrack (1:1.4.6-2build2) ...
Setting up kubect1 (1.29.6-1.1) ...
Setting up ebtables (2.0.11-4build2) ...
Setting up socat (1.7.4.1-3ubuntu4) ...
Setting up cri-tools (1.29.0-1.1) ...
Setting up kubernetecni (1.3.0-1.1) ...
Setting up kubelet (1.29.6-1.1) ...
Setting up kubeadm (1.29.6-1.1) ...
Processing triggers for man-db (2.10.2-1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@worker-node2:/home/ubuntu# sudo systemctl enable kubelet
root@worker-node2:/home/ubuntu#
```

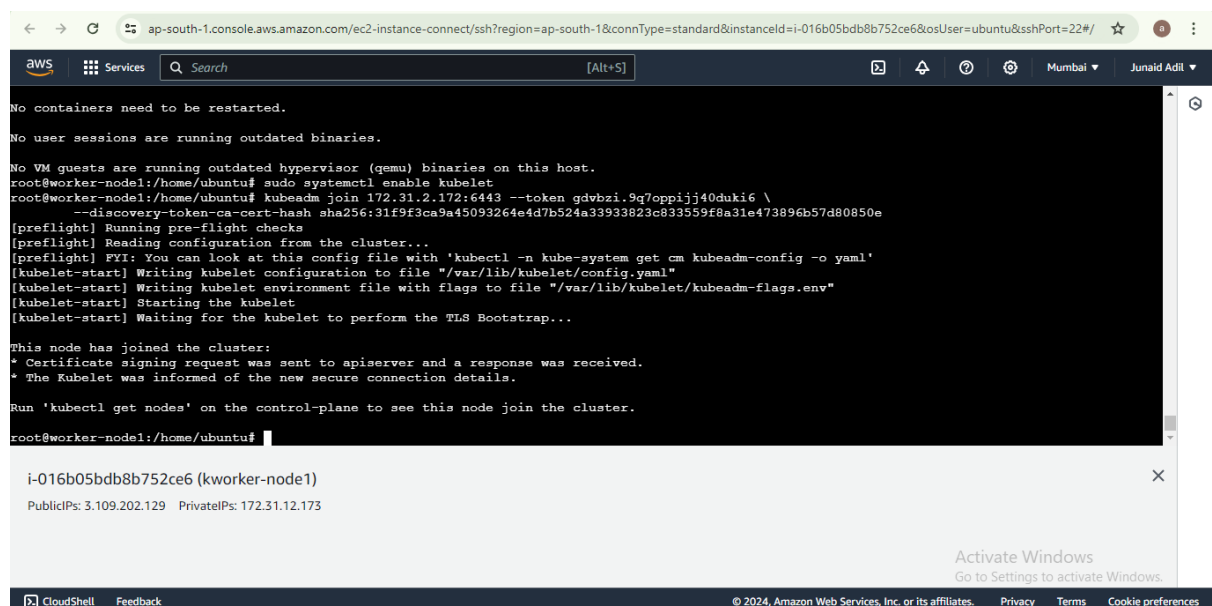
i-0336777d8f8c0ccc5 (kworker-node2)
PublicIPs: 65.0.131.179 PrivateIPs: 172.31.8.47

Activate Windows
Go to Settings to activate Windows.

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step 3: Use the command to connect worker node to master node. This command has the Ip address along with the Port number and Token.

```
“ kubeadm join 172.31.2.172:6443 --token gdvbzi.9q7oppijj40duki6 \
--discovery-token-ca-cert-hash sha256:31f9f3ca9a45093264e4d7b524a33933823c833559f8a31e473896b57d80850e “
```



```
No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@worker-node1:/home/ubuntu# sudo systemctl enable kubelet
root@worker-node1:/home/ubuntu# kubeadm join 172.31.2.172:6443 --token gdvbzi.9q7oppijj40duki6 \
--discovery-token-ca-cert-hash sha256:31f9f3ca9a45093264e4d7b524a33933823c833559f8a31e473896b57d80850e
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file '/var/lib/kubelet/config.yaml'
[kubelet-start] Writing kubelet environment file with flags to file '/var/lib/kubelet/kubeadm-flags.env'
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.
root@worker-node1:/home/ubuntu#
```

i-016b05bdb8b752ce6 (kworker-node1)
PublicIPs: 3.109.202.129 PrivateIPs: 172.31.12.173

Activate Windows
Go to Settings to activate Windows.

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

```
ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=ap-south-1&connType=standard&instanceId=i-0336777d8f8c0ccc5&osUser=ubuntu&sshPort=22#/

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@worker-node2:/home/ubuntu# sudo systemctl enable kubelet
root@worker-node2:/home/ubuntu# kubeadm join 172.31.2.172:6443 --token gdvbzi.9q7oppijj40duki6 \
--discovery-token-ca-cert-hash sha256:31f9f3ca9a45093264e4d7b524a33933823c833559f8a31e473896b57d80850e
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiservert and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

root@worker-node2:/home/ubuntu#
```

i-0336777d8f8c0ccc5 (kworker-node2)

PublicIPs: 65.0.131.179 PrivateIPs: 172.31.8.47

Activate Windows
Go to Settings to activate Windows.

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step 4: Run “kubectl get nodes” to see the details of nodes

```
ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=ap-south-1&connType=standard&instanceId=i-03358f228fb32e728&osUser=ubuntu&sshPort=22#/

ubuntu@ip-172-31-2-172:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-2-172:~$ sudo su
root@kmaster-node:/home/ubuntu# export KUBECONFIG=/etc/kubernetes/admin.conf
root@kmaster-node:/home/ubuntu# kubectl get pods
No resources found in default namespace.
root@kmaster-node:/home/ubuntu# mkdir -p $HOME/.kube
root@kmaster-node:/home/ubuntu# sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
root@kmaster-node:/home/ubuntu# sudo chown $(id -u):$(id -g) $HOME/.kube/config
root@kmaster-node:/home/ubuntu# export KUBECONFIG=/etc/kubernetes/admin.conf
root@kmaster-node:/home/ubuntu# kubectl apply -f https://github.com/coreos/flannel/raw/master/Documentation/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
root@kmaster-node:/home/ubuntu# kubectl get nodes
NAME                STATUS    ROLES    AGE   VERSION
kmaster-node        NotReady control-plane 4m54s v1.29.6
worker-node1        NotReady <none>      2m17s v1.29.6
worker-node2        NotReady <none>      2m7s  v1.29.6
root@kmaster-node:/home/ubuntu#
```

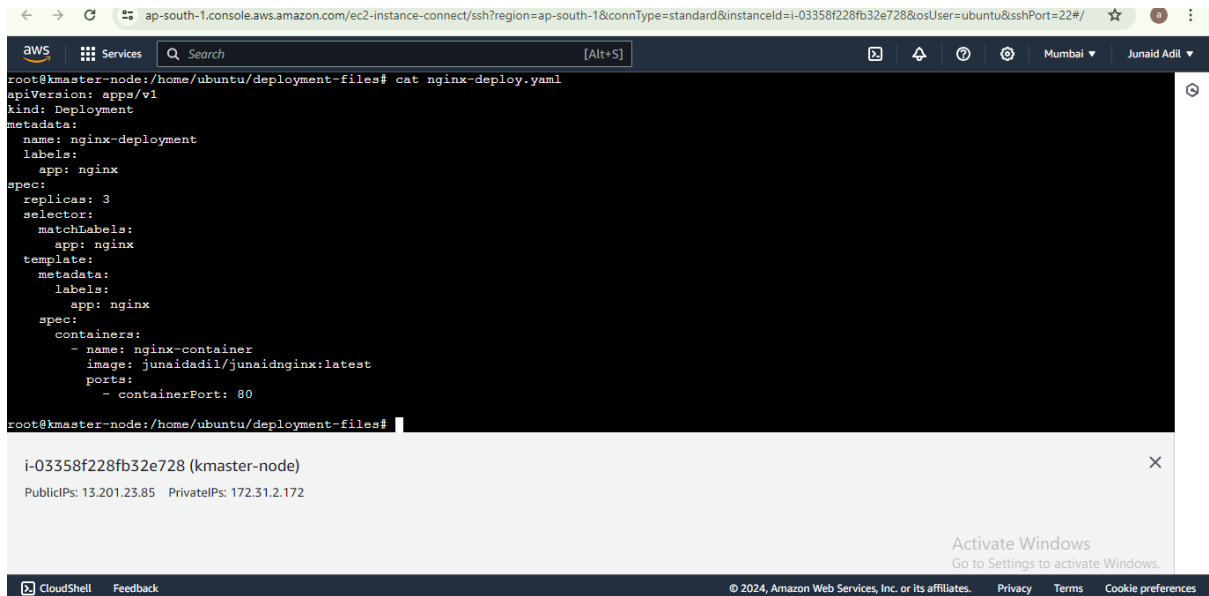
i-03358f228fb32e728 (kmaster-node)

PublicIPs: 13.201.23.85 PrivateIPs: 172.31.2.172

Activate Windows
Go to Settings to activate Windows.

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step 5: Create a deployment file “deploy.yaml” and enter the data to deploy and create replicas



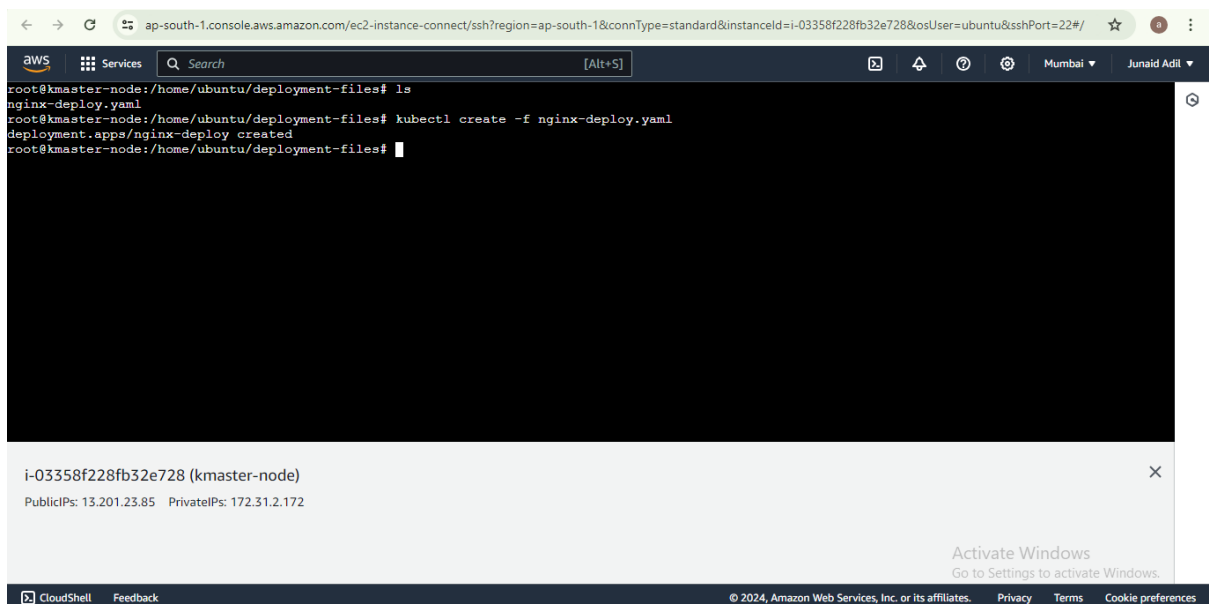
```
root@kmaster-node: /home/ubuntu/deployment-files# cat nginx-deploy.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx-container
          image: junaidadil/junaidnginx:latest
          ports:
            - containerPort: 80
```

i-03358f228fb32e728 (kmaster-node)
PublicIPs: 13.201.23.85 PrivateIPs: 172.31.2.172

Activate Windows
Go to Settings to activate Windows.

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step 6: Run “kubectl create -f nginx-deploy.yaml” to create a deployment controller object as per the definition.



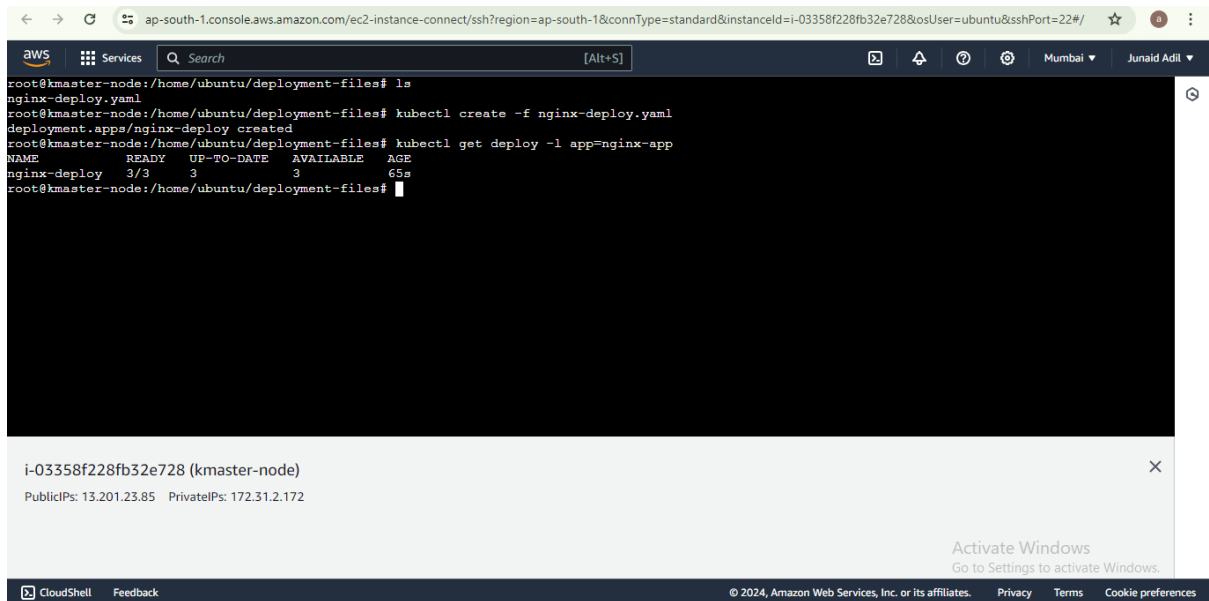
```
root@kmaster-node: /home/ubuntu/deployment-files# ls
nginx-deploy.yaml
root@kmaster-node: /home/ubuntu/deployment-files# kubectl create -f nginx-deploy.yaml
deployment.apps/nginx-deploy created
root@kmaster-node: /home/ubuntu/deployment-files#
```

i-03358f228fb32e728 (kmaster-node)
PublicIPs: 13.201.23.85 PrivateIPs: 172.31.2.172

Activate Windows
Go to Settings to activate Windows.

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step 7: To check the deployed objects run command “kubectl get deploy -l app=nginx-app”



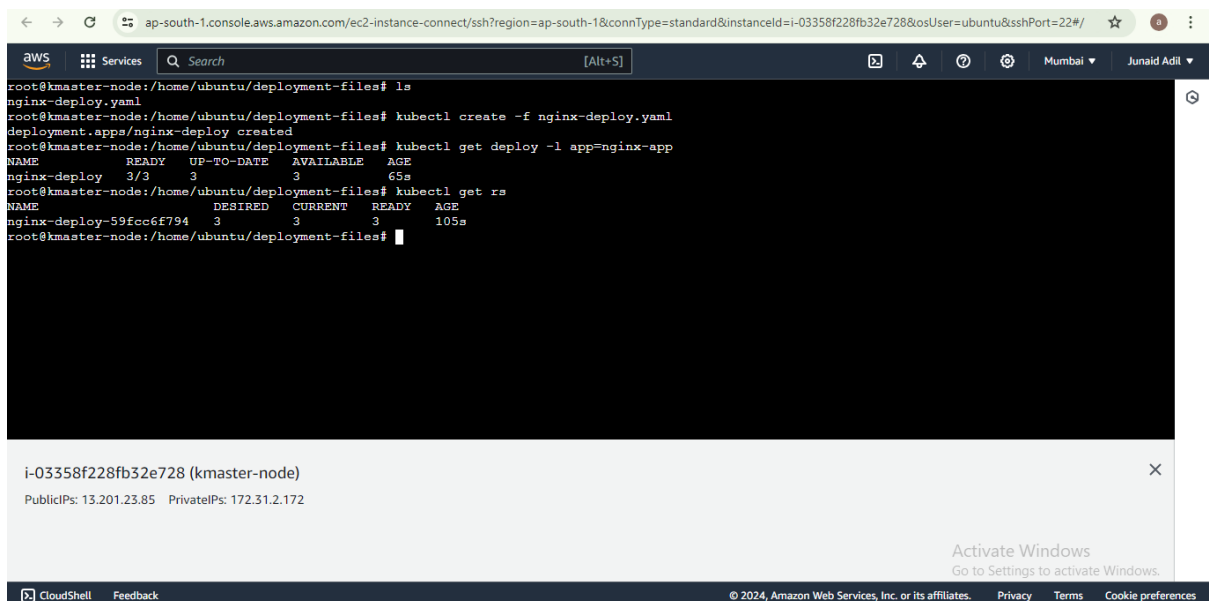
The screenshot shows the AWS CloudShell interface with a terminal window. The terminal output is as follows:

```
root@kmaster-node:/home/ubuntu/deployment-files# ls
nginx-deploy.yaml
root@kmaster-node:/home/ubuntu/deployment-files# kubectl create -f nginx-deploy.yaml
deployment.apps/nginx-deploy created
root@kmaster-node:/home/ubuntu/deployment-files# kubectl get deploy -l app=nginx-app
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deploy   3/3     3             3           65s
root@kmaster-node:/home/ubuntu/deployment-files#
```

Below the terminal window, the instance details for i-03358f228fb32e728 (kmaster-node) are displayed, showing PublicIPs: 13.201.23.85 and PrivateIPs: 172.31.2.172. An "Activate Windows" watermark is visible in the bottom right corner of the terminal area.

We can see it is upon running

Step 8: To check the replica set, Run command “kubectl get rs”

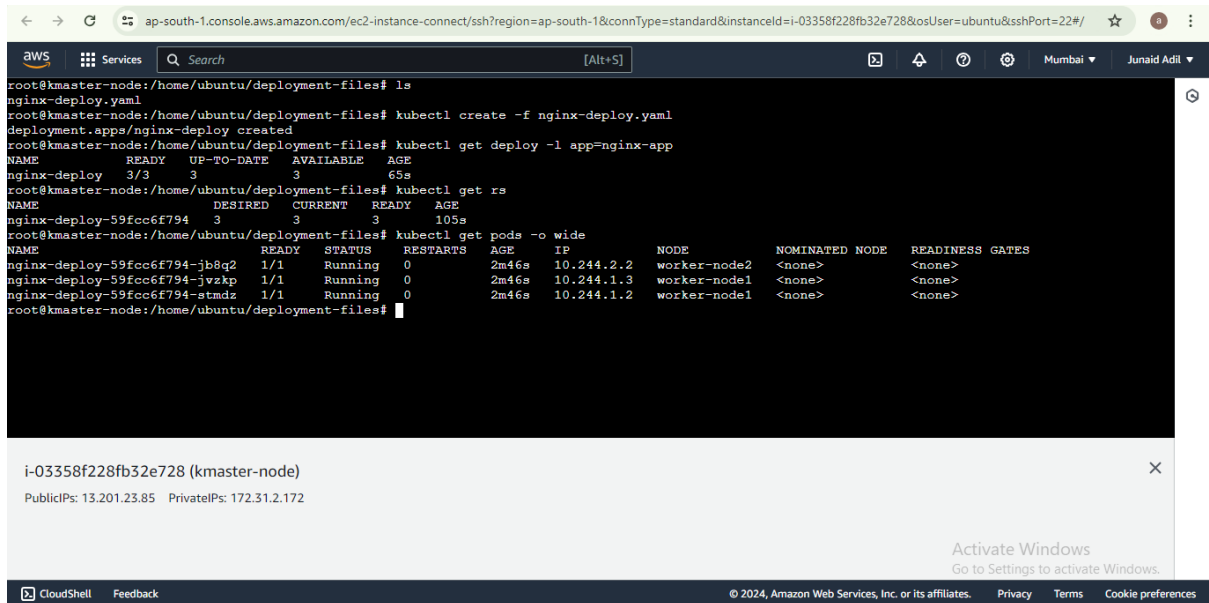


The screenshot shows the AWS CloudShell interface with a terminal window. The terminal output is as follows:

```
root@kmaster-node:/home/ubuntu/deployment-files# ls
nginx-deploy.yaml
root@kmaster-node:/home/ubuntu/deployment-files# kubectl create -f nginx-deploy.yaml
deployment.apps/nginx-deploy created
root@kmaster-node:/home/ubuntu/deployment-files# kubectl get deploy -l app=nginx-app
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deploy   3/3     3             3           65s
root@kmaster-node:/home/ubuntu/deployment-files# kubectl get rs
NAME                               DESIRED   CURRENT   READY   AGE
nginx-deploy-59fcc6f794             3         3         3       105s
root@kmaster-node:/home/ubuntu/deployment-files#
```

Below the terminal window, the instance details for i-03358f228fb32e728 (kmaster-node) are displayed, showing PublicIPs: 13.201.23.85 and PrivateIPs: 172.31.2.172. An "Activate Windows" watermark is visible in the bottom right corner of the terminal area.

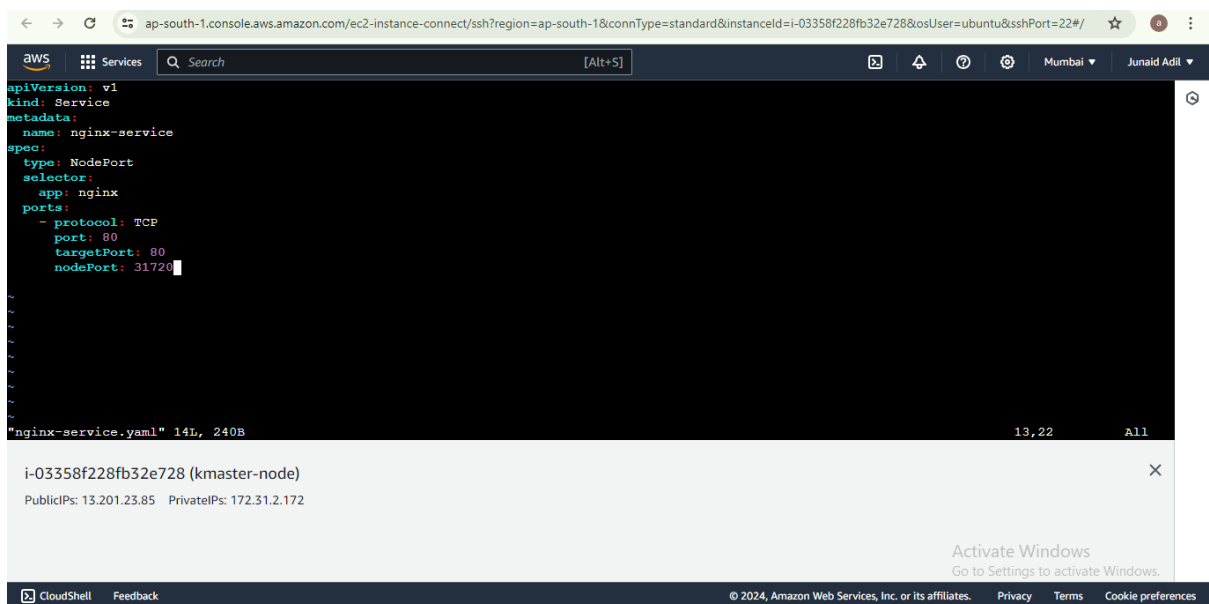
Step 9: To check in which nodes the 3 replicas are running, execute command “`kubectl get pods -o wide`”



The screenshot shows an AWS CloudShell terminal session. The user is logged in as 'ubuntu' on an EC2 instance. They have created a deployment named 'nginx-deploy' with 3 replicas. The terminal output shows the deployment was created successfully. Then, they run 'kubectl get deploy -l app=nginx-app', which shows the deployment is in a 'READY' state with 3/3 replicas. Finally, they run 'kubectl get pods -o wide', which shows three pods running on three worker nodes (worker-node1, worker-node2, and worker-node3). The pods are named 'nginx-deploy-59fcc6f794-jb8q2', 'nginx-deploy-59fcc6f794-jvzxp', and 'nginx-deploy-59fcc6f794-stmdz'. The terminal also shows the 'kubectl get rs' command output, which shows the same three replicas.

```
root@kmaster-node: /home/ubuntu/deployment-files# ls
nginx-deploy.yaml
root@kmaster-node: /home/ubuntu/deployment-files# kubectl create -f nginx-deploy.yaml
deployment.apps/nginx-deploy created
root@kmaster-node: /home/ubuntu/deployment-files# kubectl get deploy -l app=nginx-app
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deploy   3/3     3             3           65s
root@kmaster-node: /home/ubuntu/deployment-files# kubectl get rs
NAME          DESIRED   CURRENT   READY   AGE
nginx-deploy-59fcc6f794   3           3           3       105s
root@kmaster-node: /home/ubuntu/deployment-files# kubectl get pods -o wide
NAME          READY   STATUS    RESTARTS   AGE   IP            NODE          NOMINATED NODE   READINESS GATES
nginx-deploy-59fcc6f794-jb8q2   1/1     Running   0          2m46s   10.244.2.2    worker-node2   <none>            <none>
nginx-deploy-59fcc6f794-jvzxp   1/1     Running   0          2m46s   10.244.1.3    worker-node1   <none>            <none>
nginx-deploy-59fcc6f794-stmdz   1/1     Running   0          2m46s   10.244.1.2    worker-node1   <none>            <none>
```

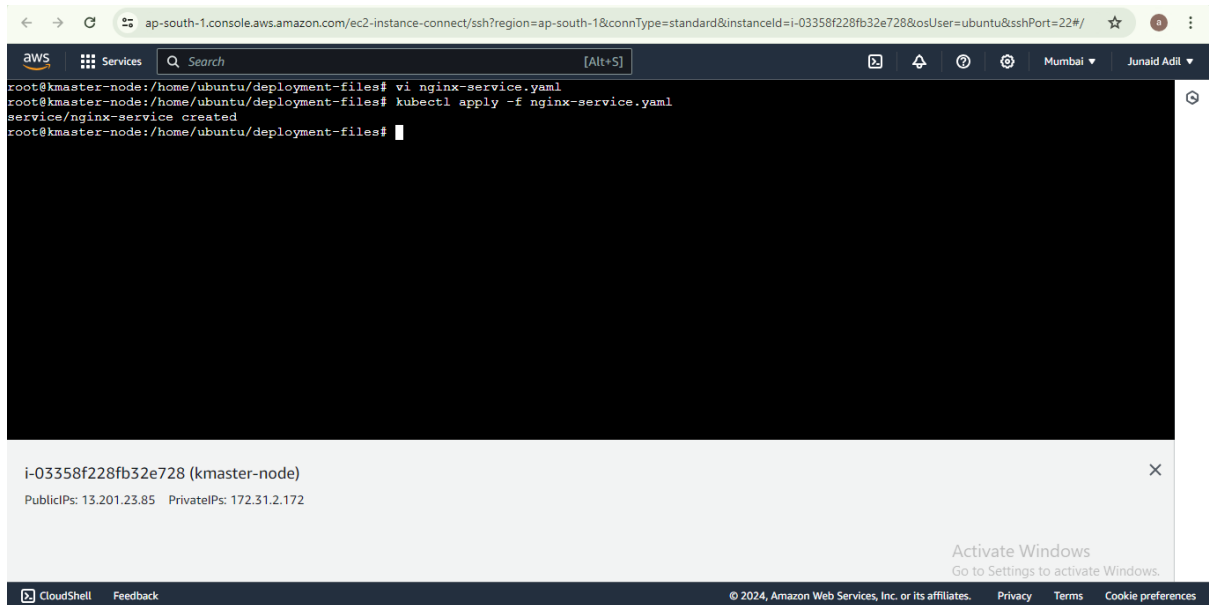
Step 10: Now to expose it to the internet, we need to create a “Node-port” service. For that create a file “nginx-service.yaml”



The screenshot shows the same AWS CloudShell terminal session. The user is now creating a service named 'nginx-service'. They run 'kubectl create -f nginx-service.yaml', which shows the service was created successfully. The terminal output shows the service is in a 'READY' state with 3/3 replicas. The service is named 'nginx-service' and is of type 'NodePort'. The service is exposed on port 31720. The terminal also shows the 'kubectl get service' command output, which shows the same service.

```
root@kmaster-node: /home/ubuntu/deployment-files# kubectl create -f nginx-service.yaml
service/nginx-service created
root@kmaster-node: /home/ubuntu/deployment-files# kubectl get service
NAME          TYPE        PORT          TARGETPORT
nginx-service  NodePort    31720         80
```


Step 11: Execute command “`kubectl apply -f nginx-service.yaml`” to create/apply the service.

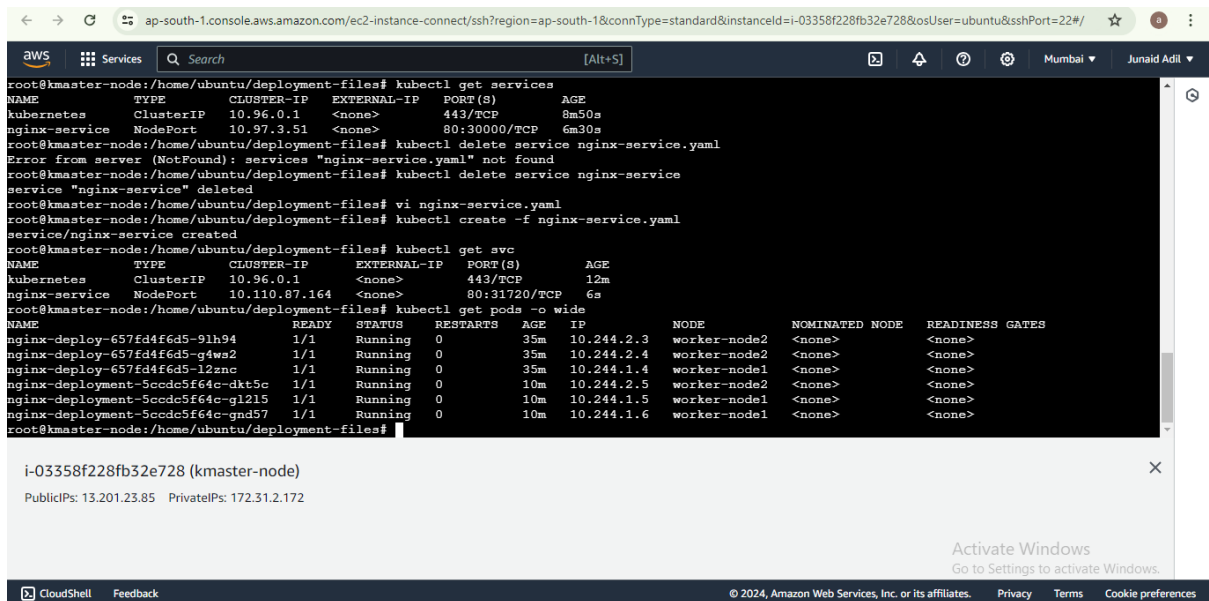


The screenshot shows the AWS CloudShell interface. The terminal window displays the following commands and output:

```
root@kmaster-node:/home/ubuntu/deployment-files# vi nginx-service.yaml
root@kmaster-node:/home/ubuntu/deployment-files# kubectl apply -f nginx-service.yaml
service/nginx-service created
root@kmaster-node:/home/ubuntu/deployment-files#
```

Below the terminal window, a box identifies the instance as `i-03358f228fb32e728 (kmaster-node)` with PublicIPs: 13.201.23.85 and PrivateIPs: 172.31.2.172. An "Activate Windows" watermark is visible in the bottom right corner.

Step 12: Execute the command to get the IP address, Node and status “`kubectl get nodes -o wide`”



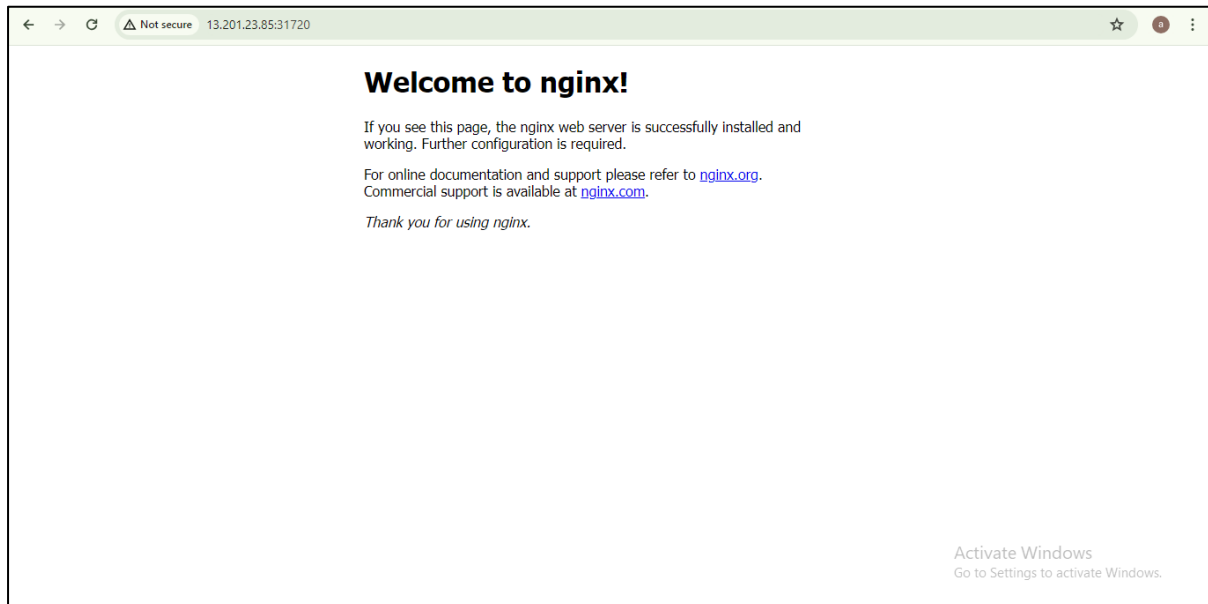
The screenshot shows the AWS CloudShell interface with the following commands and output:

```
root@kmaster-node:/home/ubuntu/deployment-files# kubectl get services
NAME                TYPE        CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
kubernetes           ClusterIP   10.96.0.1     <none>          443/TCP           8m50s
nginx-service        NodePort    10.97.3.51    <none>          80:30000/TCP      6m30s
root@kmaster-node:/home/ubuntu/deployment-files# kubectl delete service nginx-service.yaml
Error from server (NotFound): services "nginx-service.yaml" not found
root@kmaster-node:/home/ubuntu/deployment-files# kubectl delete service nginx-service
service "nginx-service" deleted
root@kmaster-node:/home/ubuntu/deployment-files# vi nginx-service.yaml
root@kmaster-node:/home/ubuntu/deployment-files# kubectl create -f nginx-service.yaml
service/nginx-service created
root@kmaster-node:/home/ubuntu/deployment-files# kubectl get svc
NAME                TYPE        CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
kubernetes           ClusterIP   10.96.0.1     <none>          443/TCP           12m
nginx-service        NodePort    10.110.87.164 <none>          80:31720/TCP      6s
root@kmaster-node:/home/ubuntu/deployment-files# kubectl get pods -o wide
NAME                                READY    STATUS    RESTARTS   AGE   IP            NODE                NOMINATED NODE    READINESS GATES
nginx-deploy-657fd4f6d5-9lh94       1/1     Running   0           35m   10.244.2.3    worker-node2        <none>              <none>
nginx-deploy-657fd4f6d5-g4ws2       1/1     Running   0           35m   10.244.2.4    worker-node2        <none>              <none>
nginx-deploy-657fd4f6d5-l2znc       1/1     Running   0           35m   10.244.1.4    worker-node1        <none>              <none>
nginx-deployment-5ccdc5f64c-dkt5c   1/1     Running   0           10m   10.244.2.5    worker-node2        <none>              <none>
nginx-deployment-5ccdc5f64c-gl215    1/1     Running   0           10m   10.244.1.5    worker-node1        <none>              <none>
nginx-deployment-5ccdc5f64c-gnd57    1/1     Running   0           10m   10.244.1.6    worker-node1        <none>              <none>
```

Below the terminal window, a box identifies the instance as `i-03358f228fb32e728 (kmaster-node)` with PublicIPs: 13.201.23.85 and PrivateIPs: 172.31.2.172. An "Activate Windows" watermark is visible in the bottom right corner.

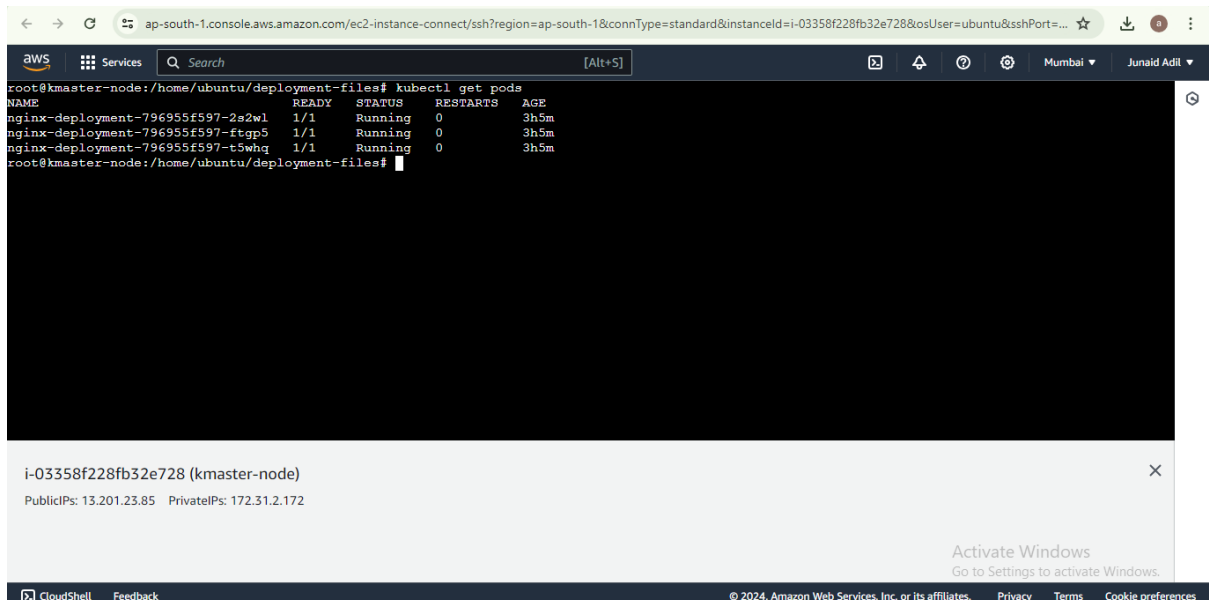
Step 13: Using public IP and the port number 31720 we can access the application through browser

<External IP>:<Port Number>



L2 - Scale-up and Scale-Down the Pods Deployed.

Step 1: To scale up and scale down the Pods, Deploy the Pods.



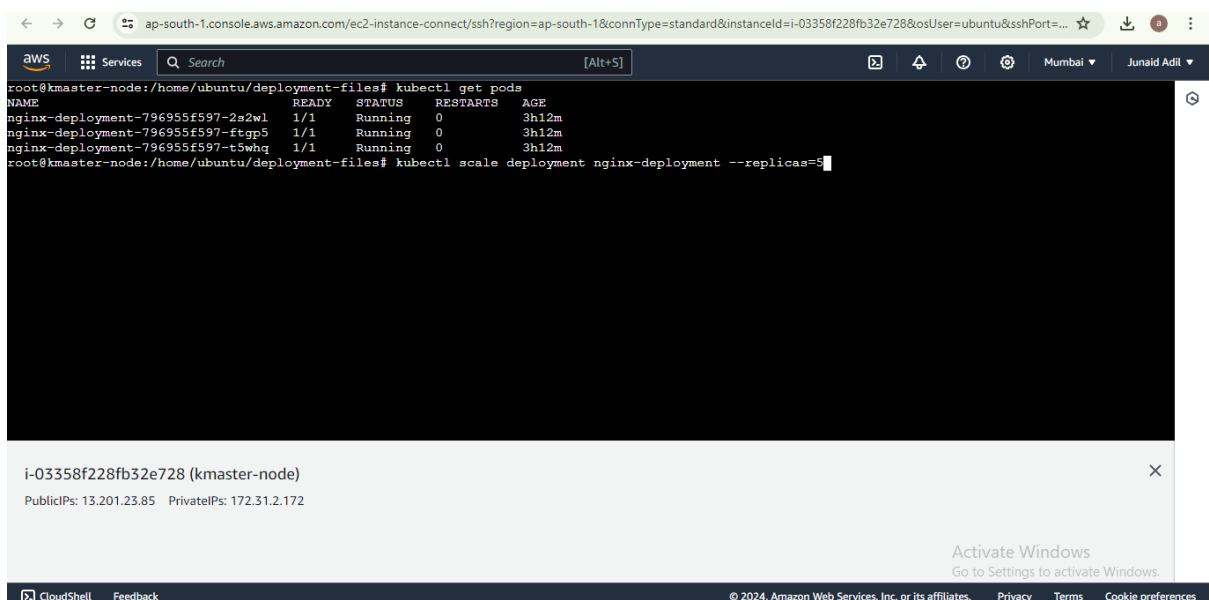
The screenshot shows the AWS CloudShell interface. The terminal window displays the command `kubectl get pods` and its output, which lists three pods in a 'Running' state. The pods are named `nginx-deployment-796955f597-2s2w1`, `nginx-deployment-796955f597-ftgp5`, and `nginx-deployment-796955f597-t5whq`. Each pod has a 'READY' status of '1/1', 'STATUS' of 'Running', 'RESTARTS' of '0', and an 'AGE' of '3h5m'. The terminal prompt is `root@kmaster-node:/home/ubuntu/deployment-files#`. Below the terminal window, the instance details for `i-03358f228fb32e728 (kmaster-node)` are shown, including PublicIPs: 13.201.23.85 and PrivateIPs: 172.31.2.172. The bottom of the interface shows the 'CloudShell' logo, a 'Feedback' link, and the copyright notice '© 2024, Amazon Web Services, Inc. or its affiliates.' along with links for 'Privacy', 'Terms', and 'Cookie preferences'.

```
root@kmaster-node:/home/ubuntu/deployment-files# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-796955f597-2s2w1   1/1     Running   0           3h5m
nginx-deployment-796955f597-ftgp5   1/1     Running   0           3h5m
nginx-deployment-796955f597-t5whq   1/1     Running   0           3h5m
root@kmaster-node:/home/ubuntu/deployment-files#
```

Deployed 3 Pods.

Step 2: Syntax “ `kubectl scale deployment <deployment-name> --replicas=<desired-replica-count>` ”

Now to scale up we can use command “`kubectl scale deployment nginx-deploy --replicas=5`” here 5 is the number of Pods we want to scale up with.



The screenshot shows the AWS CloudShell interface. The terminal window displays the command `kubectl get pods` and its output, which lists three pods in a 'Running' state. The pods are named `nginx-deployment-796955f597-2s2w1`, `nginx-deployment-796955f597-ftgp5`, and `nginx-deployment-796955f597-t5whq`. Each pod has a 'READY' status of '1/1', 'STATUS' of 'Running', 'RESTARTS' of '0', and an 'AGE' of '3h12m'. The terminal prompt is `root@kmaster-node:/home/ubuntu/deployment-files#`. Below the terminal window, the instance details for `i-03358f228fb32e728 (kmaster-node)` are shown, including PublicIPs: 13.201.23.85 and PrivateIPs: 172.31.2.172. The bottom of the interface shows the 'CloudShell' logo, a 'Feedback' link, and the copyright notice '© 2024, Amazon Web Services, Inc. or its affiliates.' along with links for 'Privacy', 'Terms', and 'Cookie preferences'.

```
root@kmaster-node:/home/ubuntu/deployment-files# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-796955f597-2s2w1   1/1     Running   0           3h12m
nginx-deployment-796955f597-ftgp5   1/1     Running   0           3h12m
nginx-deployment-796955f597-t5whq   1/1     Running   0           3h12m
root@kmaster-node:/home/ubuntu/deployment-files# kubectl scale deployment nginx-deployment --replicas=5
```

ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=ap-south-1&connType=standard&instanceId=i-03358f228fb32e728&osUser=ubuntu&sshPort=...

Services Search [Alt+S]

Mumbai Junaid Adil

```
root@kmaster-node:/home/ubuntu/deployment-files# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-796955f597-2s2w1  1/1     Running   0           3h12m
nginx-deployment-796955f597-ftgp5  1/1     Running   0           3h12m
nginx-deployment-796955f597-t5whq  1/1     Running   0           3h12m
root@kmaster-node:/home/ubuntu/deployment-files# kubectl scale deployment nginx-deployment --replicas=5
deployment.apps/nginx-deployment scaled
root@kmaster-node:/home/ubuntu/deployment-files#
```

i-03358f228fb32e728 (kmaster-node)

PublicIPs: 13.201.23.85 PrivateIPs: 172.31.2.172

Activate Windows
Go to Settings to activate Windows.

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

We can see the pods are scaled up to 5 pods.

ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=ap-south-1&connType=standard&instanceId=i-03358f228fb32e728&osUser=ubuntu&sshPort=...

Services Search [Alt+S]

Mumbai Junaid Adil

```
root@kmaster-node:/home/ubuntu/deployment-files# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-796955f597-2s2w1  1/1     Running   0           3h12m
nginx-deployment-796955f597-ftgp5  1/1     Running   0           3h12m
nginx-deployment-796955f597-t5whq  1/1     Running   0           3h12m
root@kmaster-node:/home/ubuntu/deployment-files# kubectl scale deployment nginx-deployment --replicas=5
deployment.apps/nginx-deployment scaled
root@kmaster-node:/home/ubuntu/deployment-files# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-796955f597-2s2w1  1/1     Running   0           3h13m
nginx-deployment-796955f597-batbd  1/1     Running   0           19s
nginx-deployment-796955f597-ftgp5  1/1     Running   0           3h13m
nginx-deployment-796955f597-n6chw  1/1     Running   0           19s
nginx-deployment-796955f597-t5whq  1/1     Running   0           3h13m
root@kmaster-node:/home/ubuntu/deployment-files#
```

i-03358f228fb32e728 (kmaster-node)

PublicIPs: 13.201.23.85 PrivateIPs: 172.31.2.172

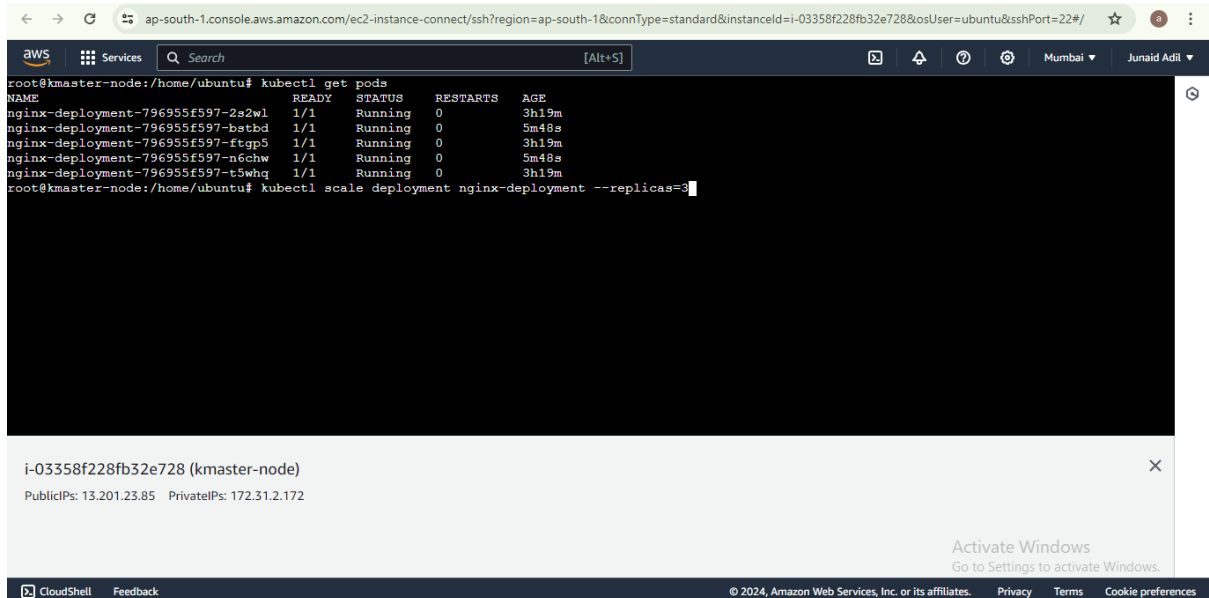
Activate Windows
Go to Settings to activate Windows.

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step 3: We can use the command to scale down:

Syntax “ `kubectl scale deployment <deployment-name> --replicas=<desired-replica-count>` ”

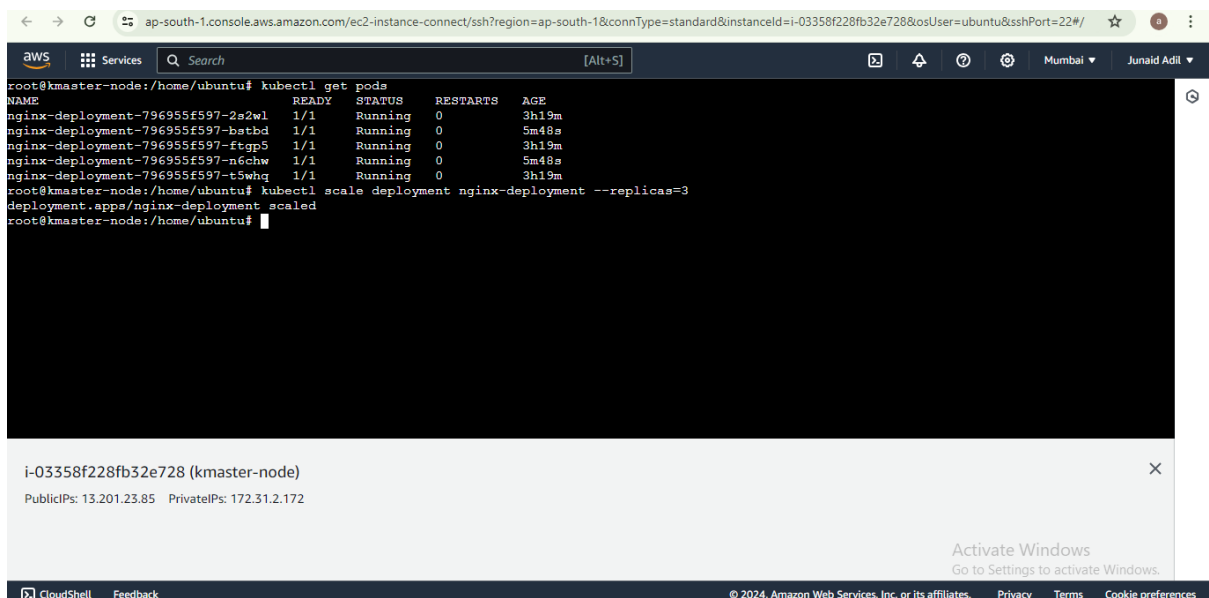
Use command “ `kubectl scale deployment nginx-deploy --replicas=3` ”



The screenshot shows the AWS CloudShell interface. The terminal window displays the following commands and output:

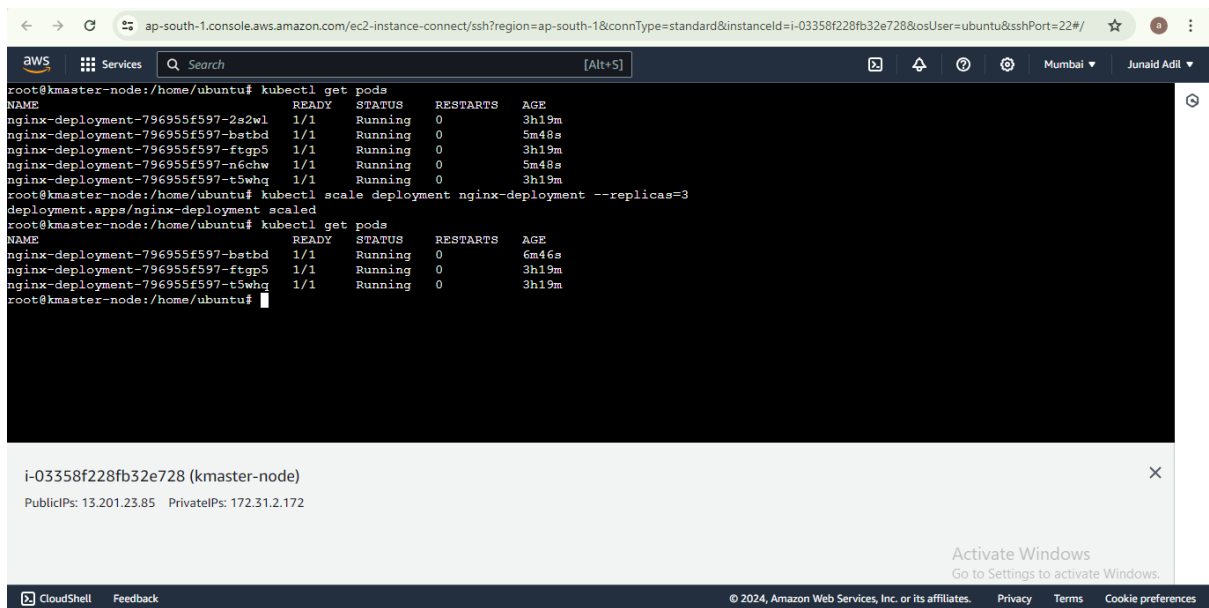
```
root@kmaster-node:/home/ubuntu# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-796955f597-2s2w1  1/1     Running   0           3h19m
nginx-deployment-796955f597-batbd  1/1     Running   0           5m48s
nginx-deployment-796955f597-ftgp5  1/1     Running   0           3h19m
nginx-deployment-796955f597-n6chw  1/1     Running   0           5m48s
nginx-deployment-796955f597-t5whq  1/1     Running   0           3h19m
root@kmaster-node:/home/ubuntu# kubectl scale deployment nginx-deploy --replicas=3
```

Below the terminal window, the instance details for `i-03358f228fb32e728 (kmaster-node)` are shown, including PublicIPs: 13.201.23.85 and PrivateIPs: 172.31.2.172. An "Activate Windows" watermark is visible in the bottom right corner.



This screenshot is identical to the one above, showing the same terminal output and instance details in the AWS CloudShell interface.

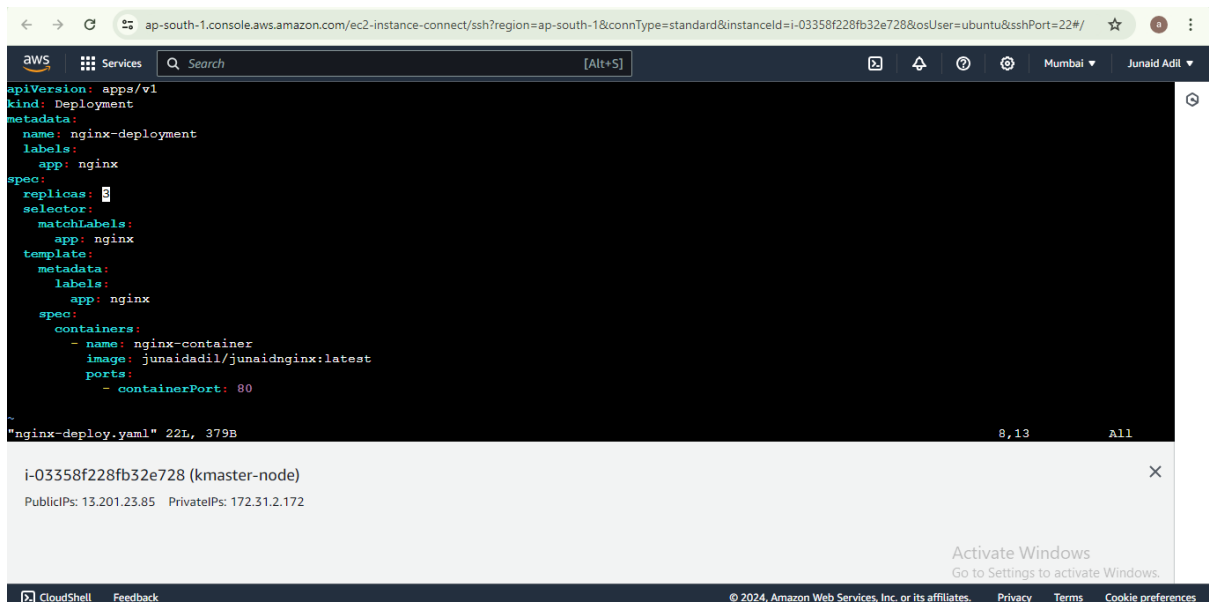
We can see the pods are scaled down to 3 pods.



The screenshot shows an AWS CloudShell terminal session. The user runs `kubectl get pods` and then `kubectl scale deployment nginx-deployment --replicas=3`. The terminal output shows the deployment being scaled and the resulting pod list.

```
root@kmaster-node:/home/ubuntu# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-796955f597-2s2w1   1/1     Running   0           3h19m
nginx-deployment-796955f597-batbd   1/1     Running   0           5m48s
nginx-deployment-796955f597-ftgp5   1/1     Running   0           3h19m
nginx-deployment-796955f597-n6chw   1/1     Running   0           5m48s
nginx-deployment-796955f597-t5whq   1/1     Running   0           3h19m
root@kmaster-node:/home/ubuntu# kubectl scale deployment nginx-deployment --replicas=3
deployment.apps/nginx-deployment scaled
root@kmaster-node:/home/ubuntu# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-796955f597-batbd   1/1     Running   0           6m46s
nginx-deployment-796955f597-ftgp5   1/1     Running   0           3h19m
nginx-deployment-796955f597-t5whq   1/1     Running   0           3h19m
root@kmaster-node:/home/ubuntu#
```

Step 4: We can also modify the number of replicas directly in the Deployment YAML file and then apply the updated configuration.



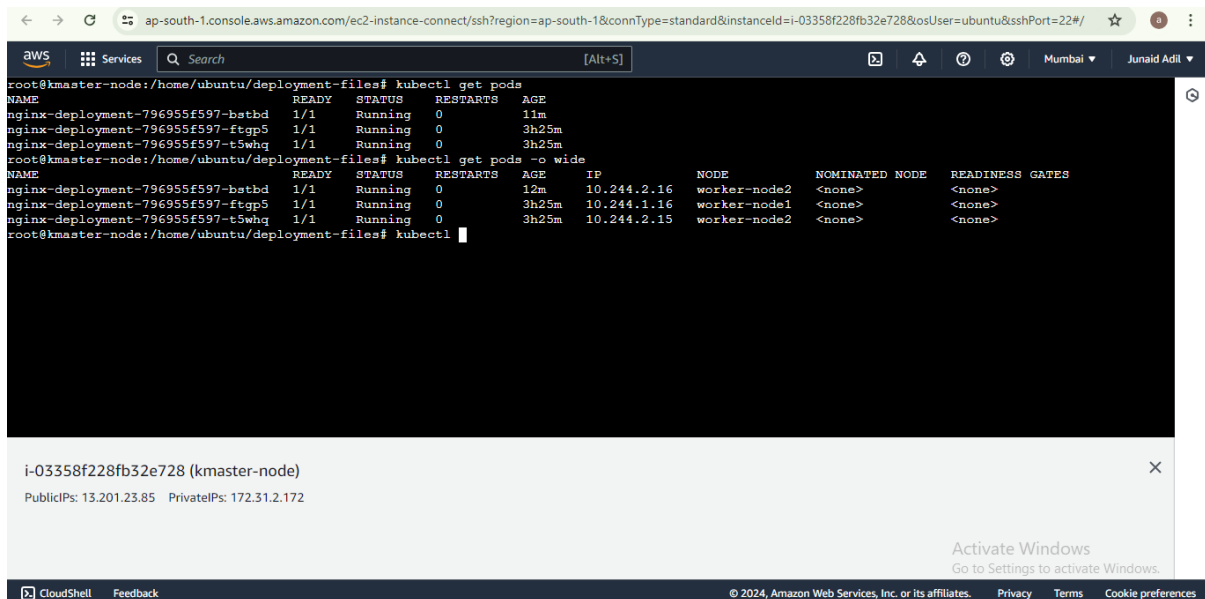
The screenshot shows an AWS CloudShell terminal session. The user displays the content of the `nginx-deploy.yaml` file, which shows the `replicas` field set to 3.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx-container
          image: junaidadil/junaidnginx:latest
          ports:
            - containerPort: 80
```

We can update the replicas required in the Deployment.yaml file & save the file. Then apply the deployment configuration using command “`kubectl apply -f nginx-deployment.yaml`”.

L3 - Implement Rolling-Update Strategy to Upgrade the Application Image from V1.0 to V1.1

Step 1: For Rolling-Update Strategy we need to identify the version of Image. Depending on that we can update the version of Image.

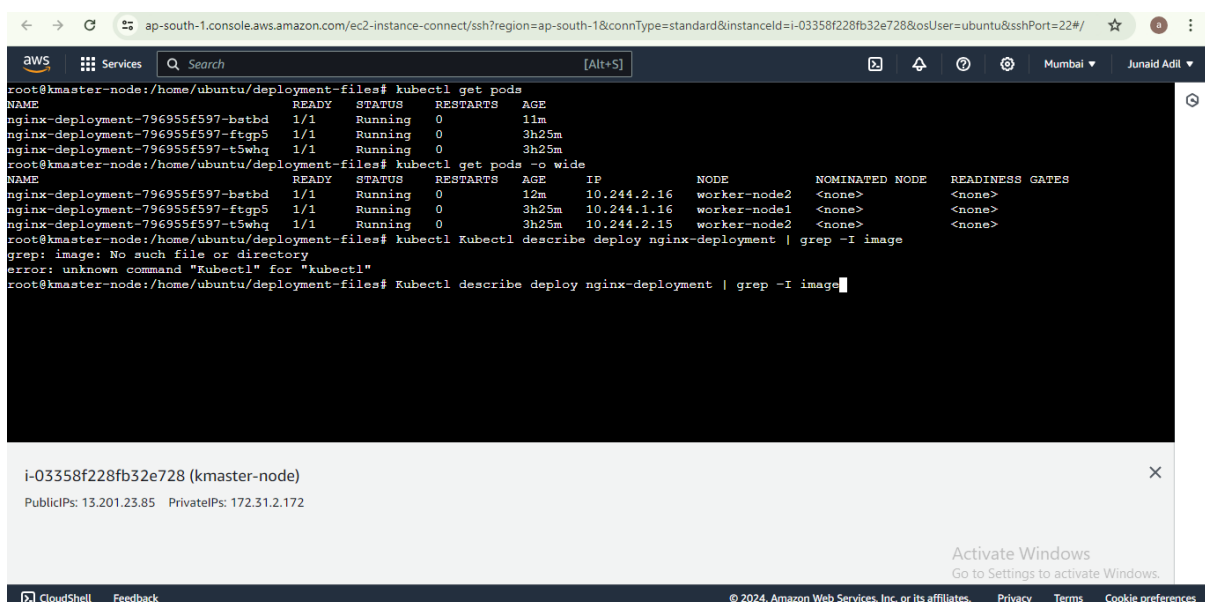


The screenshot shows an AWS CloudShell terminal window. The user is logged in as root on the kmaster-node. They run the command `kubectl get pods`, which displays a table of pods. Then they run `kubectl get pods -o wide`, which displays a wider table including node information.

```
root@kmaster-node:/home/ubuntu/deployment-files# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-796955f597-batbd   1/1     Running   0           11m
nginx-deployment-796955f597-ftgp5   1/1     Running   0           3h25m
nginx-deployment-796955f597-t5whq   1/1     Running   0           3h25m
root@kmaster-node:/home/ubuntu/deployment-files# kubectl get pods -o wide
NAME                                READY   STATUS    RESTARTS   AGE   IP              NODE             NOMINATED NODE   READINESS GATES
nginx-deployment-796955f597-batbd   1/1     Running   0           12m   10.244.2.16     worker-node2     <none>            <none>
nginx-deployment-796955f597-ftgp5   1/1     Running   0           3h25m 10.244.1.16     worker-node1     <none>            <none>
nginx-deployment-796955f597-t5whq   1/1     Running   0           3h25m 10.244.2.15     worker-node2     <none>            <none>
```

Below the terminal output, there is a box for the instance `i-03358f228fb32e728 (kmaster-node)` with PublicIPs: 13.201.23.85 and PrivateIPs: 172.31.2.172. At the bottom right, there is an "Activate Windows" watermark.

Step 2: Execute command “Kubectl describe deploy nginx-deployment | grep -I image “ to check the version of Image.



The screenshot shows the same AWS CloudShell terminal window. The user runs the command `kubectl describe deploy nginx-deployment | grep -I image`. The output shows an error message: "error: image: No such file or directory" and "error: unknown command 'Kubectl' for 'kubectl'".

```
root@kmaster-node:/home/ubuntu/deployment-files# kubectl describe deploy nginx-deployment | grep -I image
error: image: No such file or directory
error: unknown command "Kubectl" for "kubectl"
root@kmaster-node:/home/ubuntu/deployment-files#
```

Below the terminal output, there is a box for the instance `i-03358f228fb32e728 (kmaster-node)` with PublicIPs: 13.201.23.85 and PrivateIPs: 172.31.2.172. At the bottom right, there is an "Activate Windows" watermark.

The screenshot shows the AWS CloudShell interface. The terminal window displays the following commands and output:

```
root@kmaster-node:/home/ubuntu/deployment-files# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-7f78bc958d-8jn4s  1/1     Running   0           27s
nginx-deployment-7f78bc958d-ld5lt  1/1     Running   0           27s
nginx-deployment-7f78bc958d-zdmv8  1/1     Running   0           27s
root@kmaster-node:/home/ubuntu/deployment-files# kubectl describe deploy nginx-deploy | grep -i image
Image:          nginx:stable-otel
root@kmaster-node:/home/ubuntu/deployment-files#
```

Below the terminal window, a box displays the instance ID: `i-03358f228fb32e728 (kmaster-node)` and its IP addresses: `PublicIPs: 13.201.23.85` and `PrivateIPs: 172.31.2.172`. An "Activate Windows" watermark is visible in the bottom right corner.

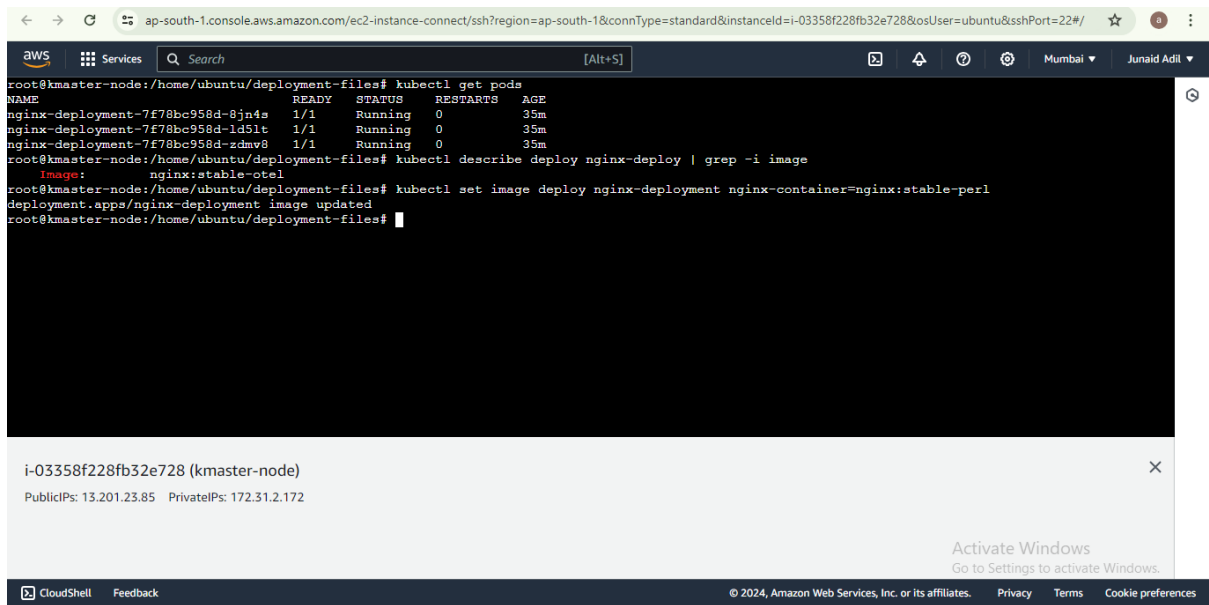
We can see it is working on version “nginx:stable-otel”.

Step 3: To Upgrade the Image, Execute command “`kubectl set image deploy nginx-deployment nginx-container=nginx:stable-perl`”

The screenshot shows the AWS CloudShell interface. The terminal window displays the following commands and output:

```
root@kmaster-node:/home/ubuntu/deployment-files# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-7f78bc958d-8jn4s  1/1     Running   0           35m
nginx-deployment-7f78bc958d-ld5lt  1/1     Running   0           35m
nginx-deployment-7f78bc958d-zdmv8  1/1     Running   0           35m
root@kmaster-node:/home/ubuntu/deployment-files# kubectl describe deploy nginx-deploy | grep -i image
Image:          nginx:stable-otel
root@kmaster-node:/home/ubuntu/deployment-files# kubectl set image deploy nginx-deployment nginx-container=nginx:stable-perl
```

Below the terminal window, a box displays the instance ID: `i-03358f228fb32e728 (kmaster-node)` and its IP addresses: `PublicIPs: 13.201.23.85` and `PrivateIPs: 172.31.2.172`. An "Activate Windows" watermark is visible in the bottom right corner.

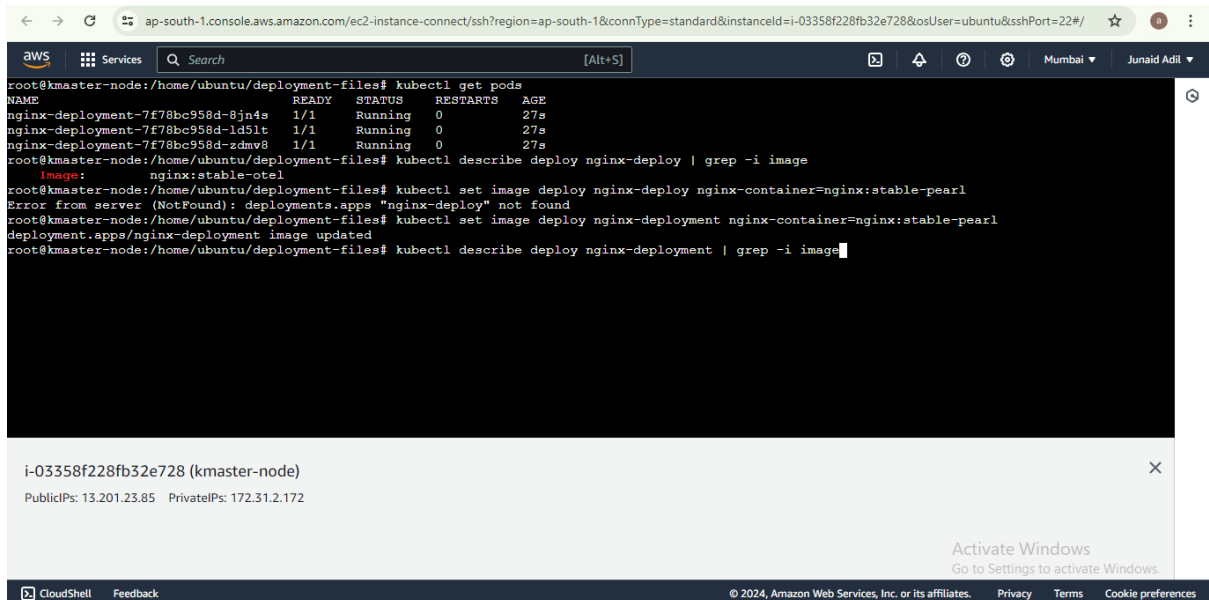


The screenshot shows the AWS CloudShell interface with a terminal window. The terminal output is as follows:

```
root@kmaster-node:/home/ubuntu/deployment-files# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-7f78bc958d-8jn4s  1/1     Running   0           35m
nginx-deployment-7f78bc958d-ld5lt  1/1     Running   0           35m
nginx-deployment-7f78bc958d-zdmv8  1/1     Running   0           35m
root@kmaster-node:/home/ubuntu/deployment-files# kubectl describe deploy nginx-deploy | grep -i image
Image:          nginx:stable-otel
root@kmaster-node:/home/ubuntu/deployment-files# kubectl set image deploy nginx-deployment nginx-container=nginx:stable-perl
deployment.apps/nginx-deployment image updated
root@kmaster-node:/home/ubuntu/deployment-files#
```

Below the terminal window, a box displays the instance ID: **i-03358f228fb32e728 (kmaster-node)**, along with PublicIPs: 13.201.23.85 and PrivateIPs: 172.31.2.172. An "Activate Windows" watermark is visible in the bottom right corner.

Step 4: Now execute command “`kubectl describe deploy nginx-deploy | grep -i image`” to check if the image has been upgraded



The screenshot shows the AWS CloudShell interface with a terminal window. The terminal output is as follows:

```
root@kmaster-node:/home/ubuntu/deployment-files# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-7f78bc958d-8jn4s  1/1     Running   0           27s
nginx-deployment-7f78bc958d-ld5lt  1/1     Running   0           27s
nginx-deployment-7f78bc958d-zdmv8  1/1     Running   0           27s
root@kmaster-node:/home/ubuntu/deployment-files# kubectl describe deploy nginx-deploy | grep -i image
Image:          nginx:stable-otel
root@kmaster-node:/home/ubuntu/deployment-files# kubectl set image deploy nginx-deployment nginx-container=nginx:stable-pearl
Error from server (NotFound): deployments.apps "nginx-deploy" not found
root@kmaster-node:/home/ubuntu/deployment-files# kubectl set image deployment nginx-container=nginx:stable-pearl
deployment.apps/nginx-deployment image updated
root@kmaster-node:/home/ubuntu/deployment-files# kubectl describe deploy nginx-deployment | grep -i image
```

Below the terminal window, a box displays the instance ID: **i-03358f228fb32e728 (kmaster-node)**, along with PublicIPs: 13.201.23.85 and PrivateIPs: 172.31.2.172. An "Activate Windows" watermark is visible in the bottom right corner.

```
ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=ap-south-1&connType=standard&instanceId=i-03358f228fb32e728&osUser=ubuntu&sshPort=22#/

root@kmaster-node:/home/ubuntu/deployment-files# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-7f78bc958d-8jn4s   1/1     Running   0           35m
nginx-deployment-7f78bc958d-ld5lt    1/1     Running   0           35m
nginx-deployment-7f78bc958d-zdmv8    1/1     Running   0           35m
root@kmaster-node:/home/ubuntu/deployment-files# kubectl describe deploy nginx-deploy | grep -i image
Image:          nginx:stable-otel
root@kmaster-node:/home/ubuntu/deployment-files# kubectl set image deploy nginx-deployment nginx-container=nginx:stable-perl
deployment.apps/nginx-deployment image updated
root@kmaster-node:/home/ubuntu/deployment-files# kubectl describe deploy nginx-deploy | grep -i image
Image:          nginx:stable-perl
root@kmaster-node:/home/ubuntu/deployment-files#
```

i-03358f228fb32e728 (kmaster-node)
PublicIPs: 13.201.23.85 PrivateIPs: 172.31.2.172

Activate Windows
Go to Settings to activate Windows.

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

We can see the Image version has been upgraded from “nginx:stable-otel” to “nginx:stable-perl”