

MODULE 7: KUBERNETES ASSIGNMENTS



MODULE 7: KUBERNETES ASSIGNMENTS -1

Tasks To Be Performed:

1. Deploy a Kubernetes cluster for 3 nodes
2. Create a NGINX deployment of 3 replicas

Launching 3 EC2 instance for master, 2 slave machine

ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#LaunchInstances:

aws [Search] [Alt+S]

EC2 > Instances > Launch an instance

Name: Kubernetes- [Add additional tags](#)

▼ **Application and OS Images (Amazon Machine Image)** [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below.

Search our full catalog including 1000s of application and OS images

Quick Start

Amazon Linux macOS **Ubuntu** Windows Red Hat SUSE Linux Debian

aws Mac ubuntu Microsoft Red Hat SUSE debian

Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type [Free tier eligible](#)

ami-00bb6a80f01f03502 (64-bit (x86)) / ami-09773b29dfbf1f2 (64-bit (Arm))

Virtualization: hvm ENA enabled: true Root device type: ebs

Description

Ubuntu Server 24.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).

Canonical, Ubuntu, 24.04, amd64 noble image

▼ **Summary**

Number of instances [Info](#)

3

When launching more than 1 instance, [consider EC2 Auto Scaling](#)

Software Image (AMI)

Canonical, Ubuntu, 24.04, amd64...[read more](#)

ami-00bb6a80f01f03502

Virtual server type (instance type)

t2.medium

Firewall (security group)

default

Storage (volumes)

1 volume(s) - 20 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth.

[Cancel](#) [Launch instance](#) [Preview code](#)

CloudShell Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Launched 3 EC2 instance for master, 2 slave machine all running state

The screenshot displays the AWS Management Console for the ap-south-1 region. The left-hand navigation pane shows the 'Instances' section under the 'EC2' service. The main content area shows a list of three EC2 instances, all of which are in the 'Running' state. The instances are named 'Kubernetes-Slave1', 'Kubernetes-Slave2', and 'Kubernetes-Master'. The 'Kubernetes-Slave2' instance is selected, and its details are shown in the lower section. The details include the instance ID, public and private IP addresses, instance state, and instance type.

Instances (1/3)

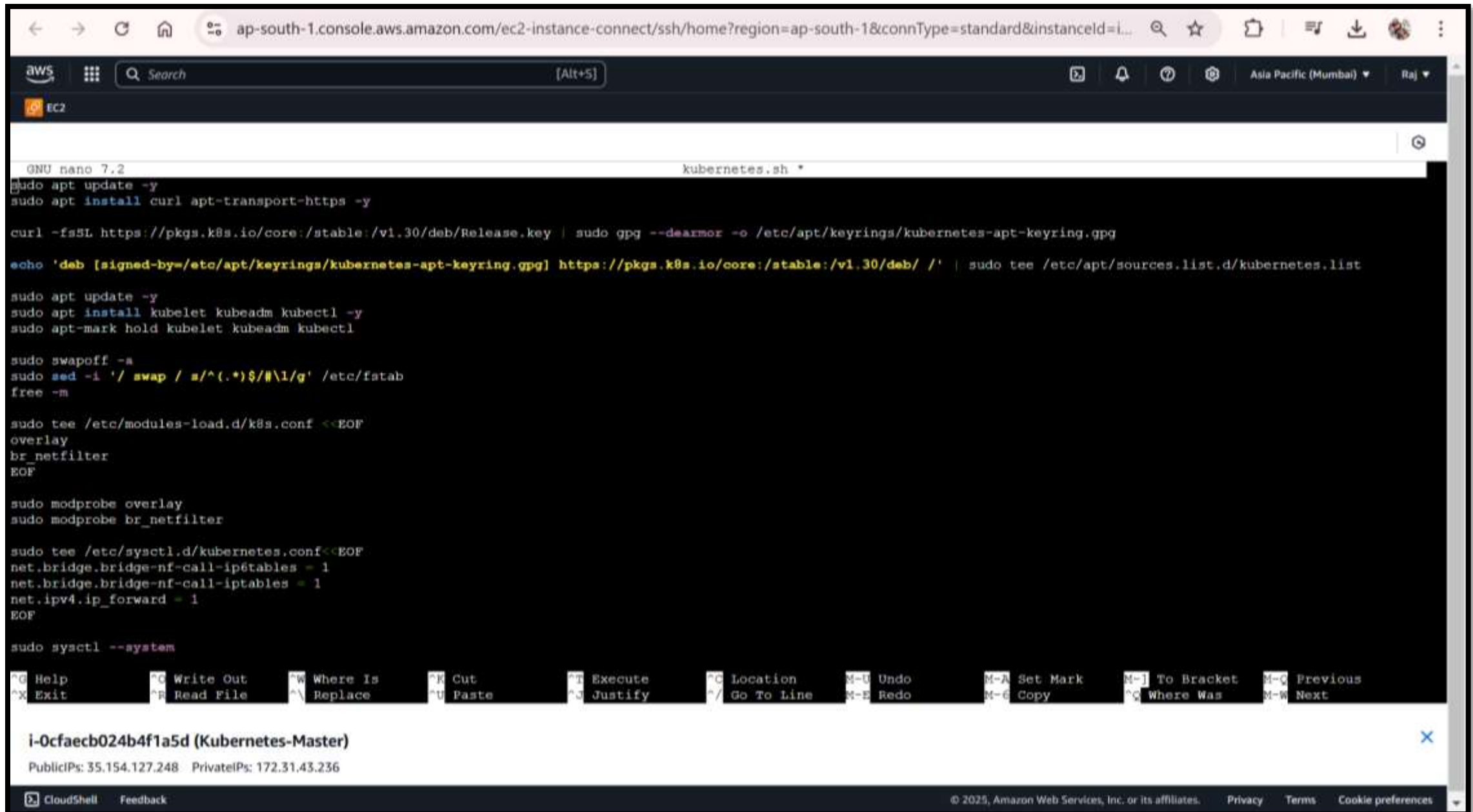
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability
Kubernetes-Slave1	i-06eee9f269356963a	Running	t2.micro	Initializing	View alarms +	ap-south-1b
Kubernetes-Slave2	i-03399ab1e5e560ffa	Running	t2.micro	Initializing	View alarms +	ap-south-1b
Kubernetes-Master	i-0cfaecb024b4f1a5d	Running	t2.medium	2/2 checks passed	View alarms +	ap-south-1a

i-03399ab1e5e560ffa (Kubernetes-Slave2)

Instance summary

Instance ID	i-03399ab1e5e560ffa	Public IPv4 address	3.109.153.255 open address	Private IPv4 addresses	172.31.3.223
IPv6 address	-	Instance state	Running	Public IPv4 DNS	ec2-3-109-153-255.ap-south-1.compute.amazonaws.com open address
Hostname type	IP name: ip-172-31-3-223.ap-south-1.compute.internal	Private IP DNS name (IPv4 only)	ip-172-31-3-223.ap-south-1.compute.internal	Elastic IP addresses	
Answer private resource DNS name		Instance type			

Kubernetes installation script



The screenshot shows an AWS CloudShell terminal window with the title "i-0cfaecb024b4f1a5d (Kubernetes-Master)". The terminal is running a script to install Kubernetes. The commands and their outputs are as follows:

```
GNU nano 7.2 kubernetes.sh *
sudo apt update -y
sudo apt install curl apt-transport-https -y

curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.30/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.30/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

sudo apt update -y
sudo apt install kubelet kubeadm kubectl -y
sudo apt-mark hold kubelet kubeadm kubectl

sudo swapoff -a
sudo sed -i 's/^.*swap / s/^(.*)$/#\1/g' /etc/fstab
free -m

sudo tee /etc/modules-load.d/k8s.conf <<EOF
overlay
br_netfilter
EOF

sudo modprobe overlay
sudo modprobe br_netfilter

sudo tee /etc/sysctl.d/kubernetes.conf<<EOF
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
EOF

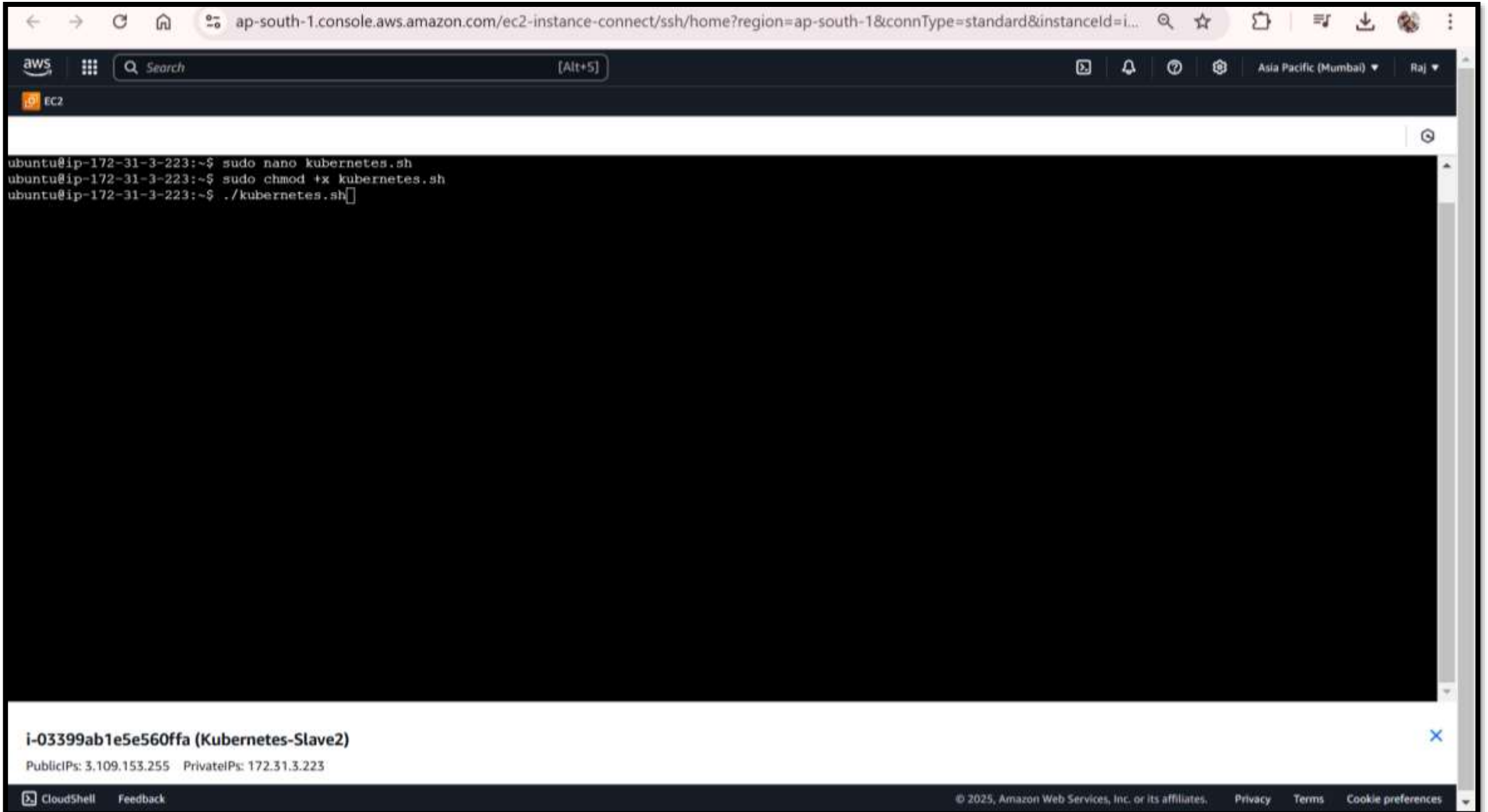
sudo sysctl --system
```

At the bottom of the terminal window, there is a table of keyboard shortcuts:

^G Help	^O Write Out	^W Where Is	^K Cut	^T Execute	^C Location	M-U Undo	M-A Set Mark	M-J To Bracket	M-Q Previous
^X Exit	^R Read File	^H Replace	^U Paste	^J Justify	^_ Go To Line	M-E Redo	M-C Copy	^G Where Was	M-W Next

The bottom of the window shows the CloudShell interface with the instance ID "i-0cfaecb024b4f1a5d (Kubernetes-Master)", public and private IP addresses, and navigation links for CloudShell, Feedback, and Amazon Web Services.

Running shell script for Kubernetes installation

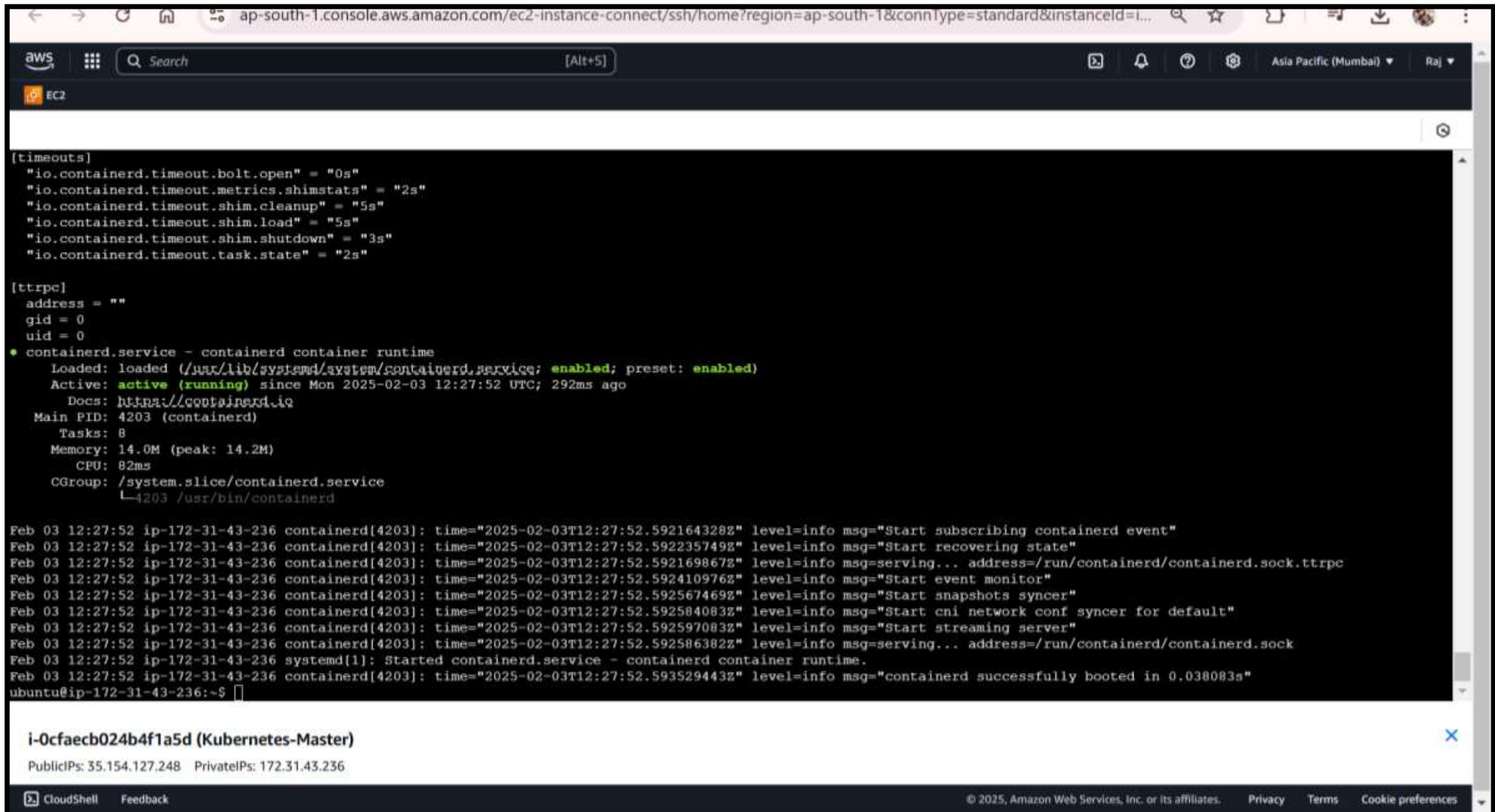


The screenshot displays the AWS Management Console interface for an EC2 instance. The browser address bar shows the URL: `ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh/home?region=ap-south-1&connType=standard&instanceId=i...`. The console header includes the AWS logo, a search bar, and navigation links for "Asia Pacific (Mumbai)" and "Raj". The main content area is a terminal window titled "EC2" showing a shell session on an Ubuntu instance. The commands executed are:

```
ubuntu@ip-172-31-3-223:~$ sudo nano kubernetes.sh
ubuntu@ip-172-31-3-223:~$ sudo chmod +x kubernetes.sh
ubuntu@ip-172-31-3-223:~$ ./kubernetes.sh
```

The terminal output is currently blank, indicating the script is still running or has not yet produced visible output. At the bottom of the console, the instance ID `i-03399ab1e5e560ffa` is identified as "Kubernetes-Slave2". Below this, the public and private IP addresses are listed: "PublicIPs: 3.109.153.255" and "PrivateIPs: 172.31.3.223". The footer of the console shows "CloudShell" and "Feedback" links, along with copyright information for Amazon Web Services, Inc. or its affiliates, and links for "Privacy", "Terms", and "Cookie preferences".

Kubernetes installation completed



The screenshot displays the AWS Management Console interface for an EC2 instance. The terminal window shows the output of a command, likely `systemctl status containerd`, indicating that the `containerd` service is loaded and active (running). Below this, a series of log messages from `containerd` and `systemd` are visible, showing the successful booting of the container runtime. The instance is identified as `i-0cfaecb024b4f1a5d (Kubernetes-Master)` and is located in the `ap-south-1` region. The console also shows the public and private IP addresses of the instance.

```
[timeouts]
"io.containerd.timeout.bolt.open" = "0s"
"io.containerd.timeout.metrics.shimstats" = "2s"
"io.containerd.timeout.shim.cleanup" = "5s"
"io.containerd.timeout.shim.load" = "5s"
"io.containerd.timeout.shim.shutdown" = "3s"
"io.containerd.timeout.task.state" = "2s"

[ttrpc]
address = ""
gid = 0
uid = 0
• containerd.service - containerd container runtime
   Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
   Active: active (running) since Mon 2025-02-03 12:27:52 UTC; 292ms ago
     Docs: https://containerd.io
   Main PID: 4203 (containerd)
      Tasks: 8
    Memory: 14.0M (peak: 14.2M)
       CPU: 82ms
    CGroup: /system.slice/containerd.service
            └─4203 /usr/bin/containerd

Feb 03 12:27:52 ip-172-31-43-236 containerd[4203]: time="2025-02-03T12:27:52.592164328Z" level=info msg="Start subscribing containerd event"
Feb 03 12:27:52 ip-172-31-43-236 containerd[4203]: time="2025-02-03T12:27:52.592235749Z" level=info msg="Start recovering state"
Feb 03 12:27:52 ip-172-31-43-236 containerd[4203]: time="2025-02-03T12:27:52.592169867Z" level=info msg="serving... address=/run/containerd/containerd.sock.ttrpc"
Feb 03 12:27:52 ip-172-31-43-236 containerd[4203]: time="2025-02-03T12:27:52.592410976Z" level=info msg="Start event monitor"
Feb 03 12:27:52 ip-172-31-43-236 containerd[4203]: time="2025-02-03T12:27:52.592567469Z" level=info msg="Start snapshots syncer"
Feb 03 12:27:52 ip-172-31-43-236 containerd[4203]: time="2025-02-03T12:27:52.592584083Z" level=info msg="Start cni network conf syncer for default"
Feb 03 12:27:52 ip-172-31-43-236 containerd[4203]: time="2025-02-03T12:27:52.592597083Z" level=info msg="Start streaming server"
Feb 03 12:27:52 ip-172-31-43-236 containerd[4203]: time="2025-02-03T12:27:52.592586382Z" level=info msg="serving... address=/run/containerd/containerd.sock"
Feb 03 12:27:52 ip-172-31-43-236 systemd[1]: Started containerd.service - containerd container runtime.
Feb 03 12:27:52 ip-172-31-43-236 containerd[4203]: time="2025-02-03T12:27:52.593529443Z" level=info msg="containerd successfully booted in 0.038083s"
ubuntu@ip-172-31-43-236:~$
```

i-0cfaecb024b4f1a5d (Kubernetes-Master)

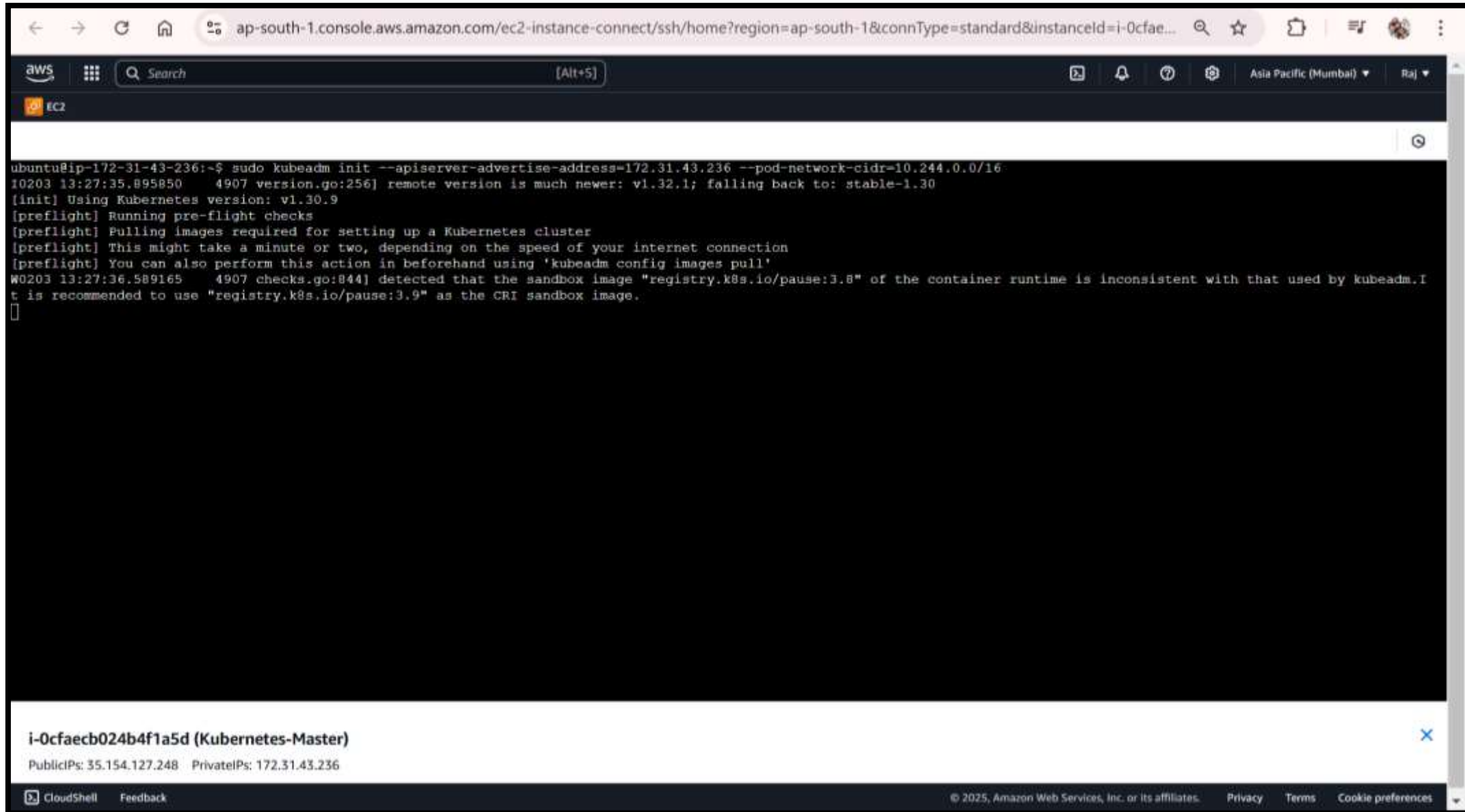
PublicIPs: 35.154.127.248 PrivateIPs: 172.31.43.236

CloudShell Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Installing a networking tool Kubernetes

```
kubeadm init --apiserver-advertise-address=172.31.43.236 --pod-network-cidr=10.244.0.0/16
```



The screenshot shows an AWS CloudShell terminal window. The browser address bar displays the URL: `ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh/home?region=ap-south-1&connType=standard&instanceId=i-0cfae...`. The terminal window has a dark background with white text. The command `sudo kubeadm init --apiserver-advertise-address=172.31.43.236 --pod-network-cidr=10.244.0.0/16` has been executed. The output shows the Kubernetes version being used (v1.30.9) and the preflight checks. A warning message indicates that the sandbox image `registry.k8s.io/pause:3.8` is inconsistent with the one used by kubeadm, and recommends using `registry.k8s.io/pause:3.9`. The terminal window is titled `i-0cfaecb024b4f1a5d (Kubernetes-Master)` and shows the public and private IP addresses of the instance.

```
ubuntu@ip-172-31-43-236:~$ sudo kubeadm init --apiserver-advertise-address=172.31.43.236 --pod-network-cidr=10.244.0.0/16
I0203 13:27:35.895850    4907 version.go:256] remote version is much newer: v1.32.1; falling back to: stable-1.30
[init] Using Kubernetes version: v1.30.9
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action in beforehand using 'kubeadm config images pull'
W0203 13:27:36.589165    4907 checks.go:844] detected that the sandbox image "registry.k8s.io/pause:3.8" of the container runtime is inconsistent with that used by kubeadm. It is recommended to use "registry.k8s.io/pause:3.9" as the CRI sandbox image.
[]
```

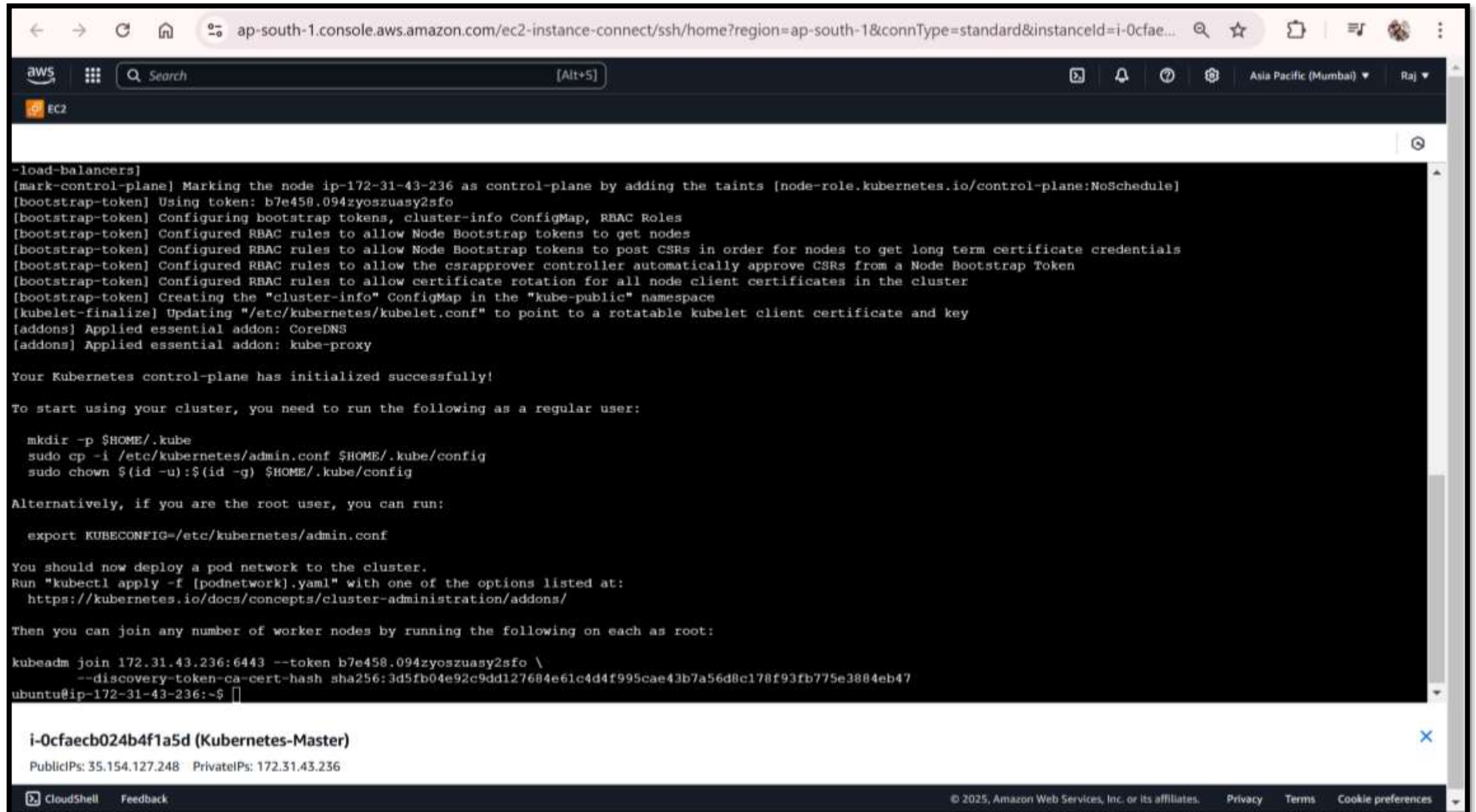
i-0cfaecb024b4f1a5d (Kubernetes-Master)

PublicIPs: 35.154.127.248 PrivateIPs: 172.31.43.236

CloudShell Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

We are going to connect master node to slave node



← → ↺ 🏠 📏 ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh/home?region=ap-south-1&connType=standard&instanceId=i-0cfae... 🔍 ☆ 📁 📄 👤

aws [Search] [Alt+S] 📄 🔔 ⓘ ⚙️ Asia Pacific (Mumbai) ▾ Raj ▾

🔗 EC2

```
-load-balancers]
[mark-control-plane] Marking the node ip-172-31-43-236 as control-plane by adding the taints [node-role.kubernetes.io/control-plane:NoSchedule]
[bootstrap-token] Using token: b7e458.094zyoszuasy2sfo
[bootstrap-token] Configuring bootstrap tokens, cluster-info ConfigMap, RBAC Roles
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to get nodes
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get long term certificate credentials
[bootstrap-token] Configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a Node Bootstrap Token
[bootstrap-token] Configured RBAC rules to allow certificate rotation for all node client certificates in the cluster
[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

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:

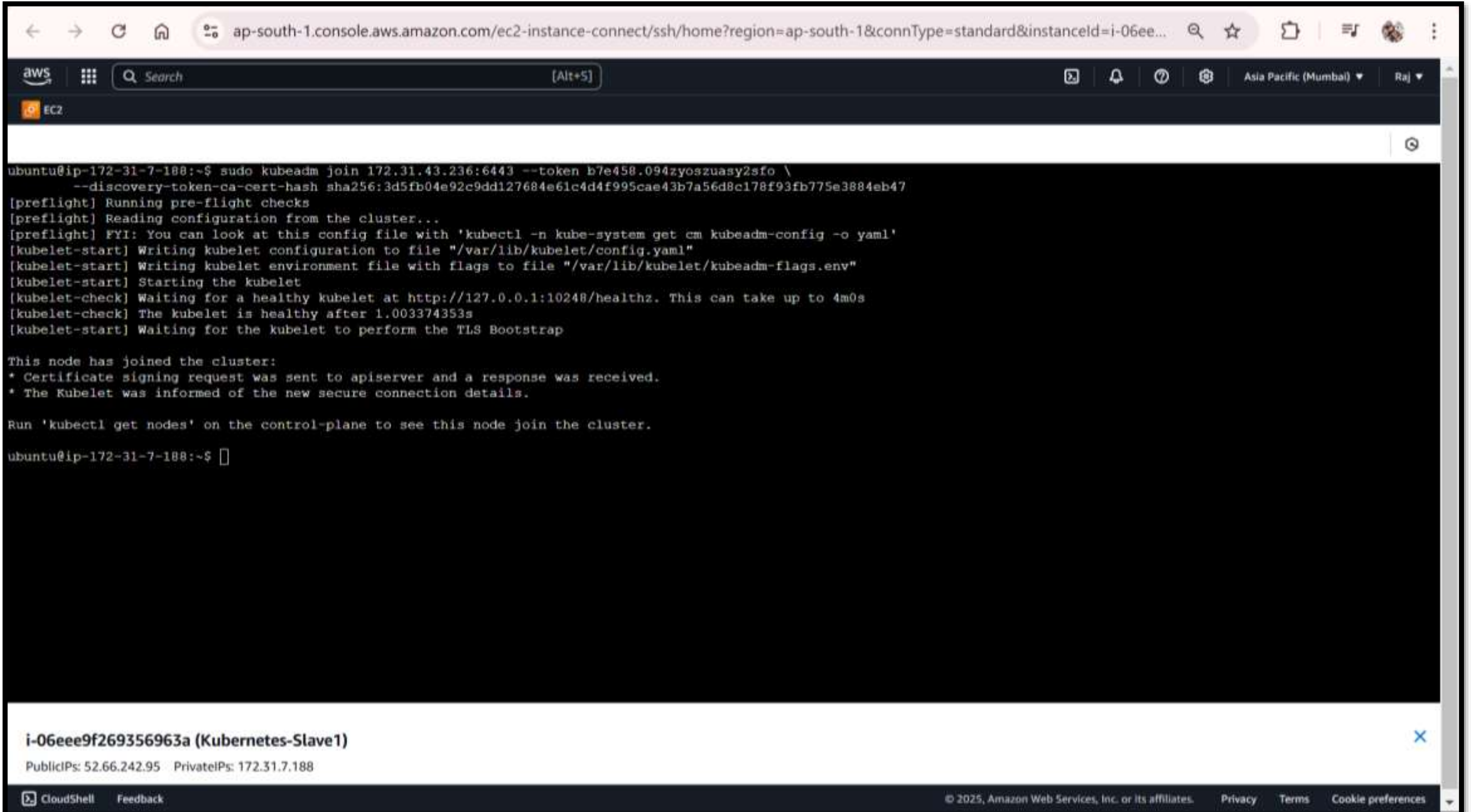
kubeadm join 172.31.43.236:6443 --token b7e458.094zyoszuasy2sfo \
--discovery-token-ca-cert-hash sha256:3d5fb04e92c9dd127684e61c4d4f995cae43b7a56d8c178f93fb775e3884eb47
ubuntu@ip-172-31-43-236:~$
```

i-0cfaecb024b4f1a5d (Kubernetes-Master) ✕

PublicIPs: 35.154.127.248 · PrivateIPs: 172.31.43.236

📄 CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Connecting slave1 node to master node



The screenshot shows the AWS Management Console interface. At the top, the browser address bar displays the URL: `ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh/home?region=ap-south-1&connType=standard&instanceId=i-06ee...`. The console header includes the AWS logo, a search bar, and navigation icons. The main content area is a CloudShell terminal window titled "i-06eee9f269356963a (Kubernetes-Slave1)". The terminal shows the execution of the `sudo kubeadm join` command with various flags, including the token and discovery token. The output of the command shows pre-flight checks, kubelet configuration, and the successful joining of the node to the cluster. The terminal prompt returns to `ubuntu@ip-172-31-7-188:~$`. At the bottom of the console, there is a footer with "CloudShell", "Feedback", and copyright information for Amazon Web Services, Inc. or its affiliates.

```
ubuntu@ip-172-31-7-188:~$ sudo kubeadm join 172.31.43.236:6443 --token b7e458.094zyoszuasy2sfo \
--discovery-token-ca-cert-hash sha256:3d5fb04e92c9dd127684e61c4d4f995cae43b7a56d8c178f93fb775e3884eb47
[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-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 1.003374353s
[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.

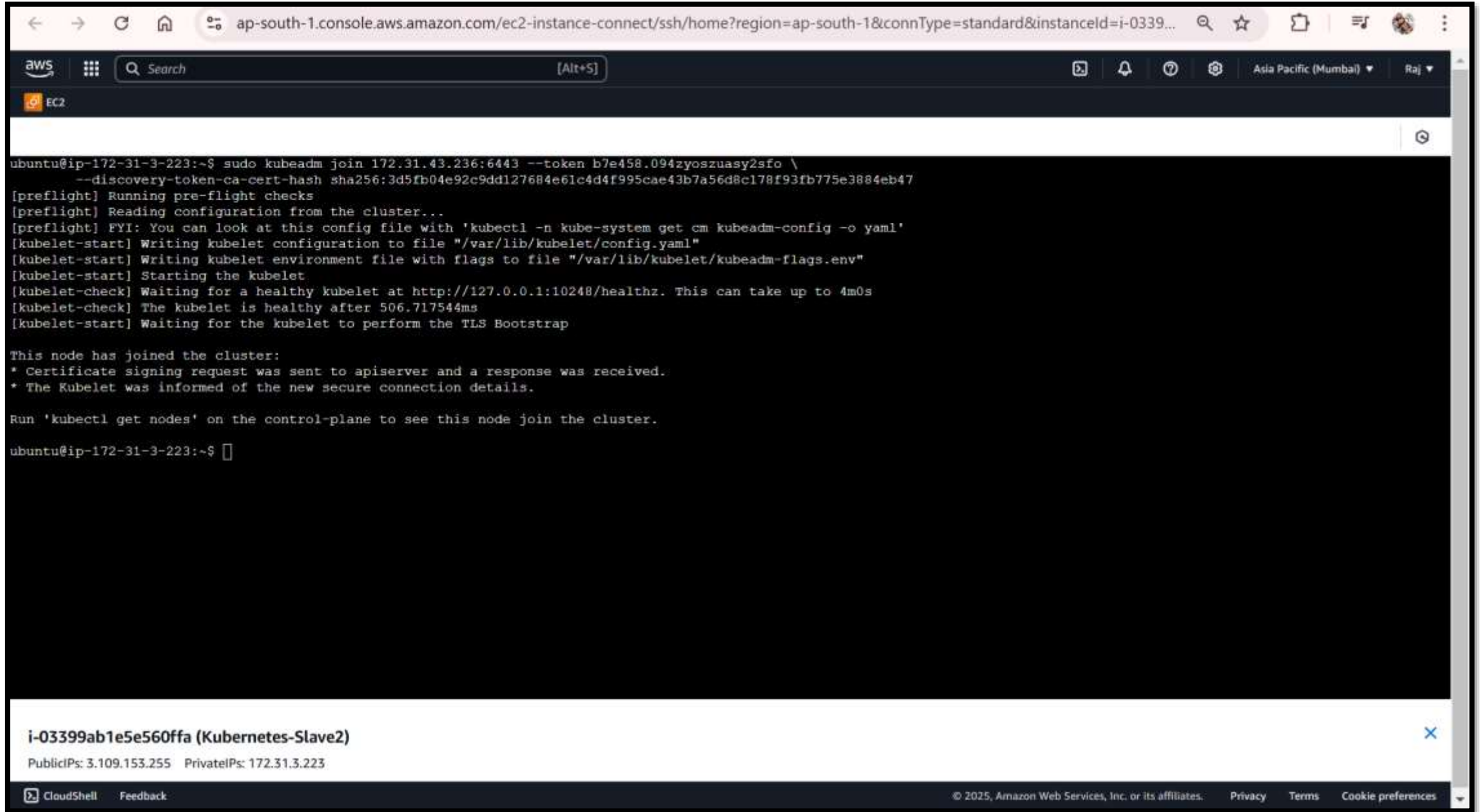
Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

ubuntu@ip-172-31-7-188:~$
```

i-06eee9f269356963a (Kubernetes-Slave1)
PublicIPs: 52.66.242.95 · PrivateIPs: 172.31.7.188

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Connecting salve2 node to master node



The screenshot shows the AWS CloudShell interface. The terminal window displays the execution of the `kubeadm join` command. The output shows pre-flight checks, kubelet configuration, and the successful joining of the node to the cluster. A confirmation message states that the node has joined and provides instructions to run `kubectl get nodes` to verify. The bottom of the terminal shows the instance ID `i-03399ab1e5e560ffa` and its IP addresses.

```
ubuntu@ip-172-31-3-223:~$ sudo kubeadm join 172.31.43.236:6443 --token b7e458.094zyoszuasy2sfo \
--discovery-token-ca-cert-hash sha256:3d5fb04e92c9dd127684e61c4d4f995cae43b7a56d8c178f93fb775e3884eb47
[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-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 506.717544ms
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

This node has joined the cluster:
* Certificate signing request was sent to apiserer 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.

ubuntu@ip-172-31-3-223:~$
```

i-03399ab1e5e560ffa (Kubernetes-Slave2)

PublicIPs: 3.109.153.255 PrivateIPs: 172.31.3.223

CloudShell Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Get list of all connect nodes

Browser address bar: `ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh/home?addressFamily=ipv4&connType=standard&instanceId=i-0cfa...`

Navigation bar: AWS logo, Search, [Alt+S], Region: Asia Pacific (Mumbai), User: Raj

EC2 Instance Connect Session

Every 2.0s: `kubectl get nodes` ip-172-31-43-236: Tue Feb 4 11:25:58 2025

NAME	STATUS	ROLES	AGE	VERSION
ip-172-31-3-223	Ready	<none>	21h	v1.30.9
ip-172-31-43-236	Ready	control-plane	21h	v1.30.9
ip-172-31-7-188	Ready	<none>	21h	v1.30.9

i-0cfaecb024b4f1a5d (Kubernetes-Master)

Public IP: 75.105.107.340 | Private IP: 172.31.43.236

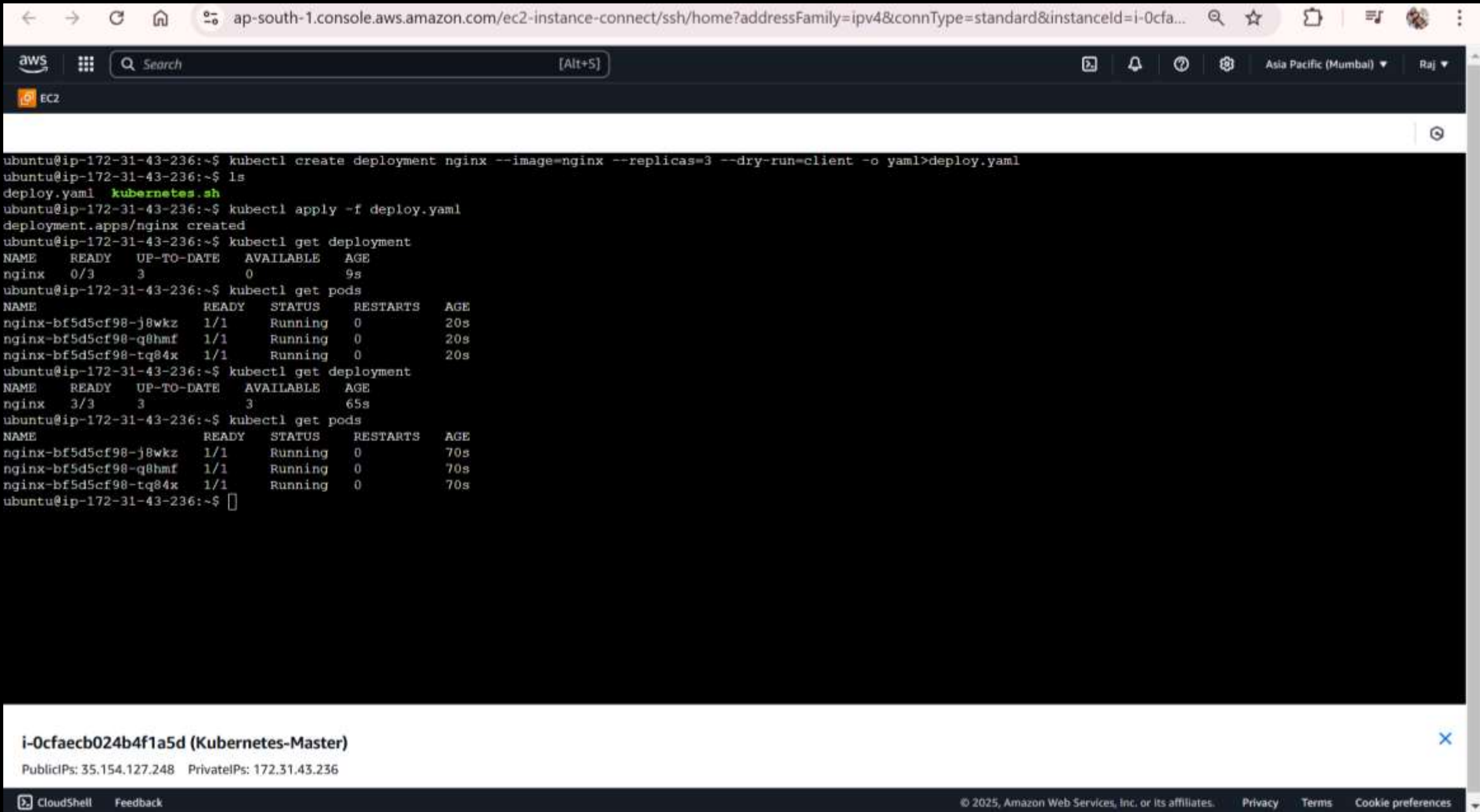
CloudShell Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Nginx replica deployment commands

1. `kubectl create deployment nginx --image=nginx --replicas=3 --dry-run=client -o yaml>deploy.yaml`
2. `kubectl apply -f deploy.yaml`
3. `kubectl get deployment`
4. `kubectl get pods`

Nginx replica deployment command execution & Validation



The screenshot shows the AWS CloudShell interface with a terminal window. The terminal output shows the successful deployment of Nginx replicas using kubectl. The commands and their outputs are as follows:

```
ubuntu@ip-172-31-43-236:~$ kubectl create deployment nginx --image=nginx --replicas=3 --dry-run=client -o yaml>deploy.yaml
ubuntu@ip-172-31-43-236:~$ ls
deploy.yaml  kubernetes.sh
ubuntu@ip-172-31-43-236:~$ kubectl apply -f deploy.yaml
deployment.apps/nginx created
ubuntu@ip-172-31-43-236:~$ kubectl get deployment
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
nginx     0/3     3            0           9s
ubuntu@ip-172-31-43-236:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-bf5d5cf98-j8wkz               1/1     Running   0           20s
nginx-bf5d5cf98-q8hmf               1/1     Running   0           20s
nginx-bf5d5cf98-tq84x               1/1     Running   0           20s
ubuntu@ip-172-31-43-236:~$ kubectl get deployment
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
nginx     3/3     3            3           65s
ubuntu@ip-172-31-43-236:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-bf5d5cf98-j8wkz               1/1     Running   0           70s
nginx-bf5d5cf98-q8hmf               1/1     Running   0           70s
nginx-bf5d5cf98-tq84x               1/1     Running   0           70s
ubuntu@ip-172-31-43-236:~$
```

Below the terminal window, the instance details for **i-0cfaecb024b4f1a5d (Kubernetes-Master)** are displayed, showing PublicIPs: 35.154.127.248 and PrivateIPs: 172.31.43.236.

The footer of the CloudShell interface includes the text "CloudShell Feedback" and "© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences".

MODULE 7: KUBERNETES ASSIGNMENTS -2

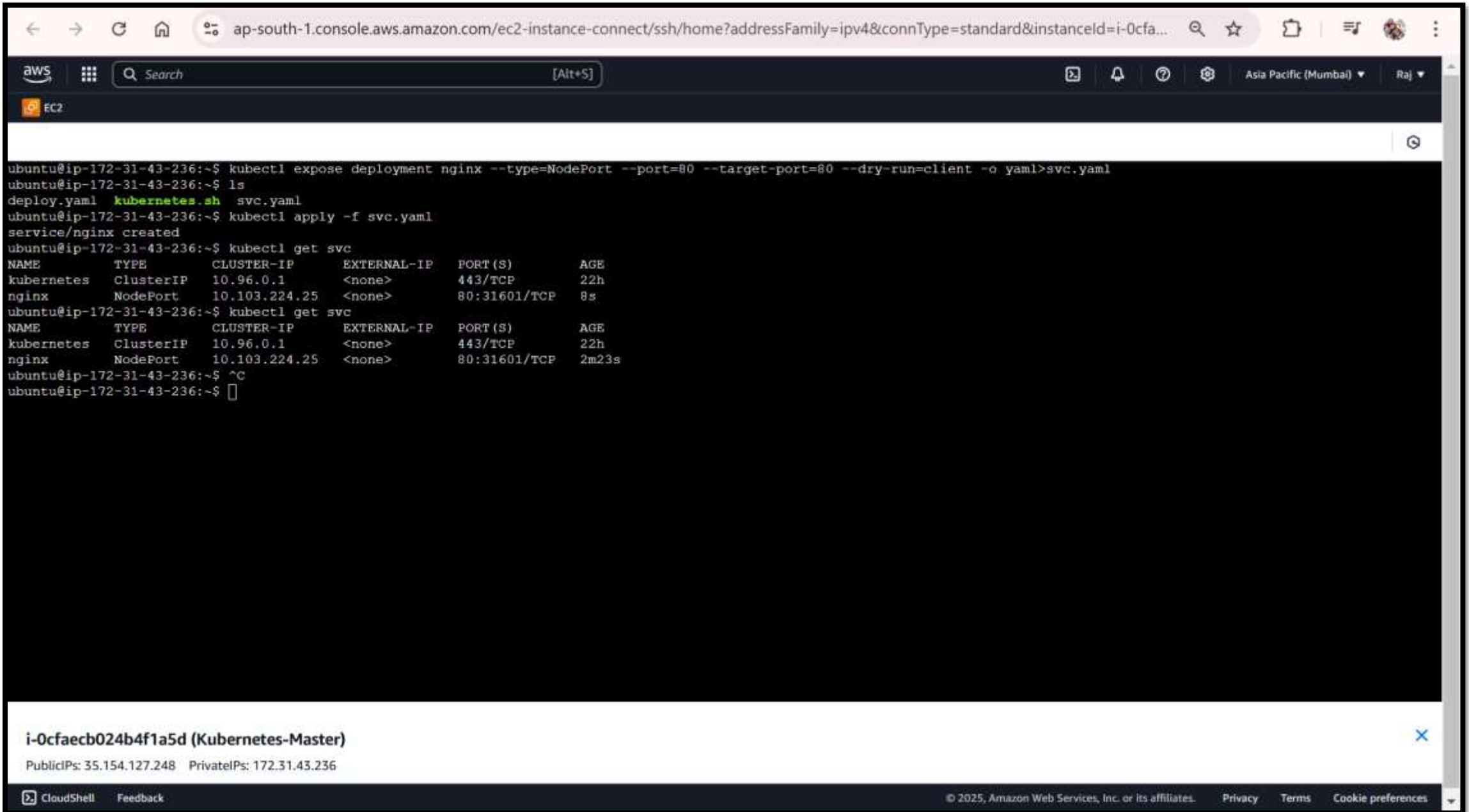
Tasks To Be Performed:

1. Use the previous deployment
2. Create a service of type NodePort for NGINX deployment
3. Check the NodePort service on a browser to verify

Node Port deployment commands

1. `kubectl expose deployment nginx --type=NodePort --port=80 --target-port=80 -
-dry-run=client -o yaml>svc.yaml`
2. `kubectl apply -f svc.yaml`
3. `kubectl get svc`

Node Port deployment commands execution & validation



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

```
ubuntu@ip-172-31-43-236:~$ kubectl expose deployment nginx --type=NodePort --port=80 --target-port=80 --dry-run=client -o yaml>svc.yaml
ubuntu@ip-172-31-43-236:~$ ls
deploy.yaml  kubernetes.sh  svc.yaml
ubuntu@ip-172-31-43-236:~$ kubectl apply -f svc.yaml
service/nginx created
ubuntu@ip-172-31-43-236:~$ kubectl get svc
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
kubernetes	ClusterIP	10.96.0.1	<none>	443/TCP	22h
nginx	NodePort	10.103.224.25	<none>	80:31601/TCP	8s

```
ubuntu@ip-172-31-43-236:~$ kubectl get svc
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
kubernetes	ClusterIP	10.96.0.1	<none>	443/TCP	22h
nginx	NodePort	10.103.224.25	<none>	80:31601/TCP	2m23s

```
ubuntu@ip-172-31-43-236:~$ ^C
ubuntu@ip-172-31-43-236:~$
```

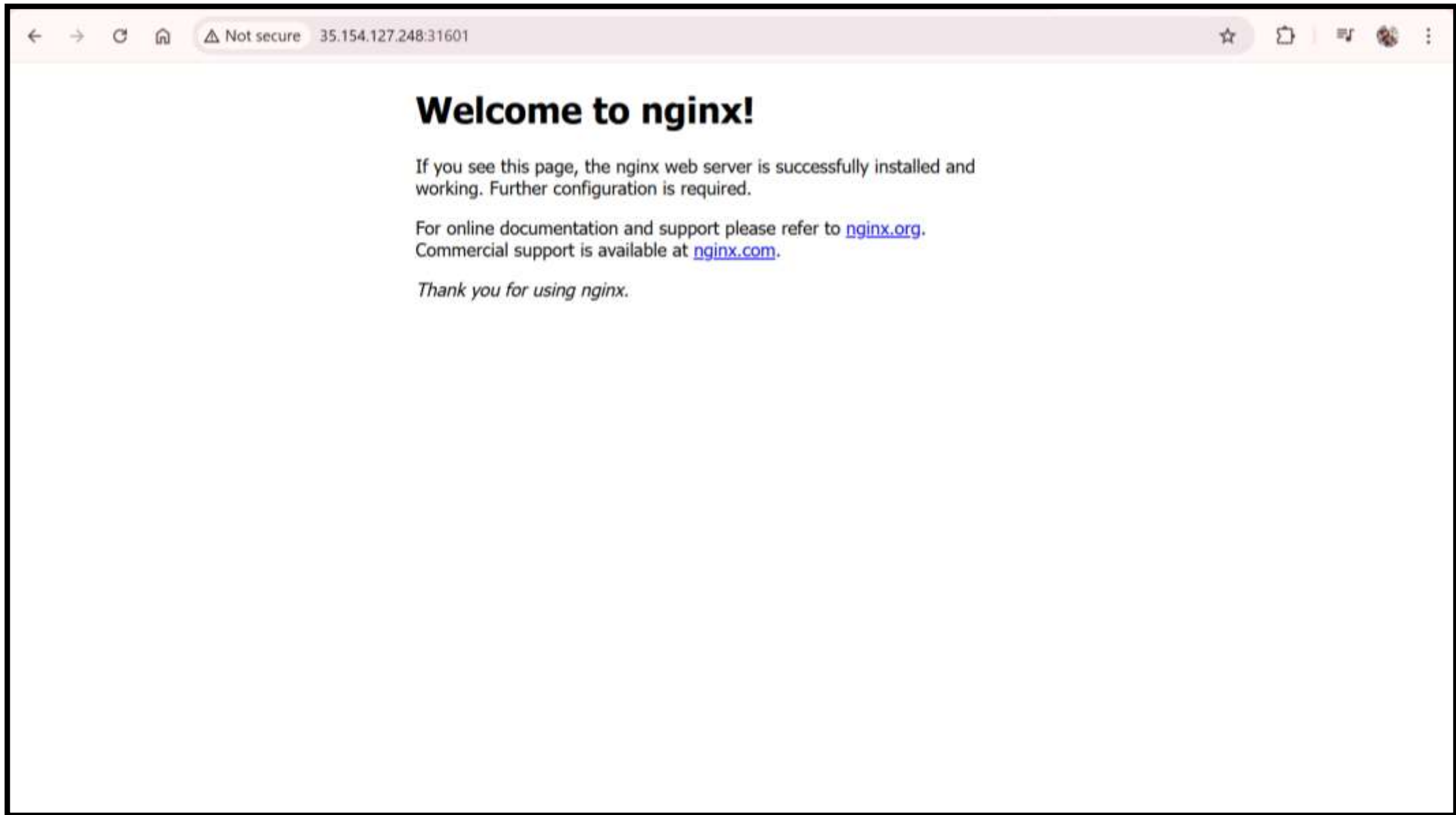
i-0cfaecb024b4f1a5d (Kubernetes-Master)

PublicIPs: 35.154.127.248 PrivateIPs: 172.31.43.236

CloudShell Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

We can see the Nginx server running on port 31601



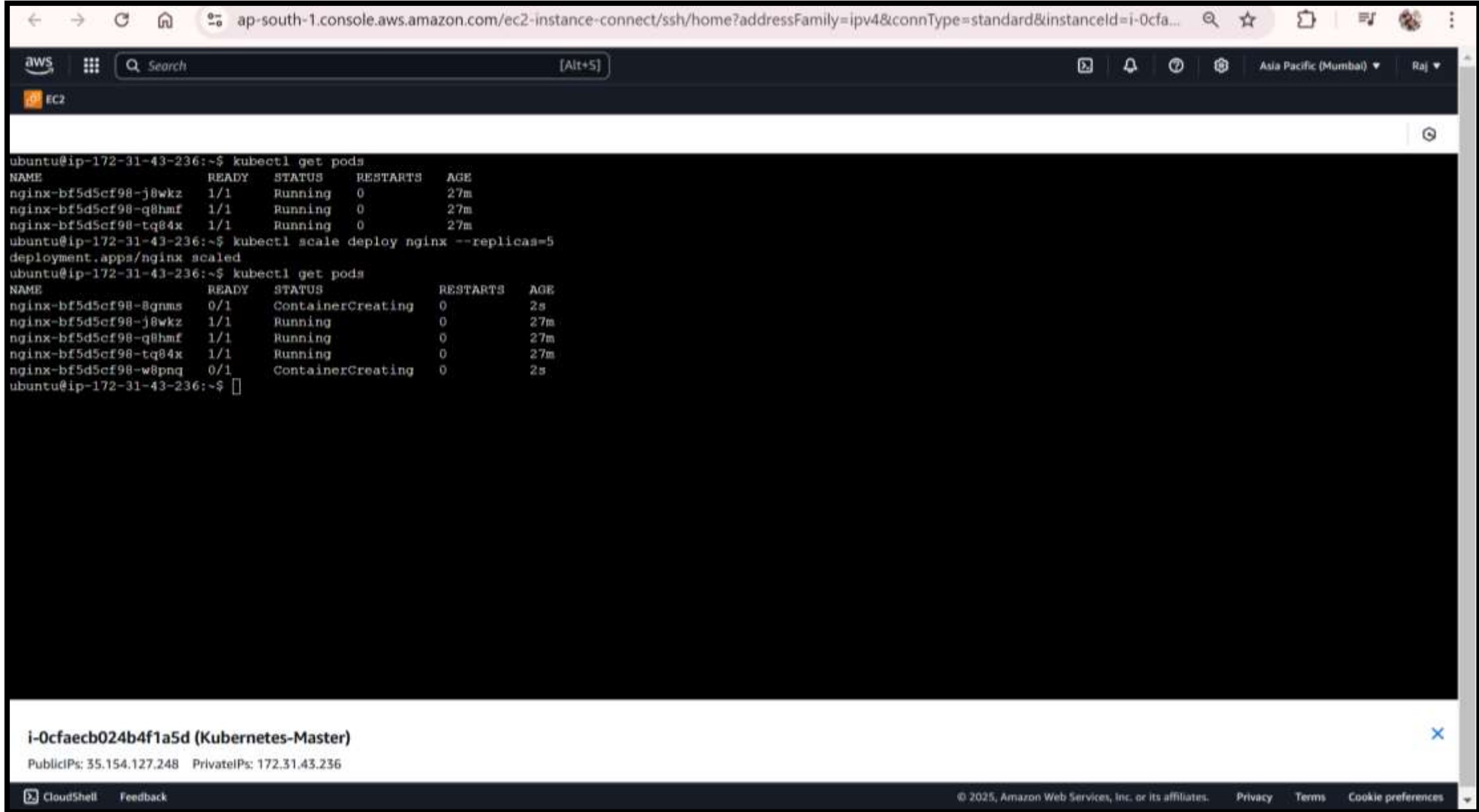
MODULE 7: KUBERNETES ASSIGNMENTS -3

Tasks To Be Performed:

1. Use the previous deployment
2. Change the replicas to 5 for the deployment

Scaling nginx replicas 3 to replicas 5

```
kubectl scale deploy nginx --replicas=5
```



The screenshot shows an AWS CloudShell terminal window with the following content:

```
ubuntu@ip-172-31-43-236:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-bf5d5cf98-j8wkz               1/1     Running   0           27m
nginx-bf5d5cf98-q8hmf               1/1     Running   0           27m
nginx-bf5d5cf98-tq84x               1/1     Running   0           27m
ubuntu@ip-172-31-43-236:~$ kubectl scale deploy nginx --replicas=5
deployment.apps/nginx scaled
ubuntu@ip-172-31-43-236:~$ kubectl get pods
NAME                                READY   STATUS              RESTARTS   AGE
nginx-bf5d5cf98-8gnms               0/1     ContainerCreating   0           2s
nginx-bf5d5cf98-j8wkz               1/1     Running              0           27m
nginx-bf5d5cf98-q8hmf               1/1     Running              0           27m
nginx-bf5d5cf98-tq84x               1/1     Running              0           27m
nginx-bf5d5cf98-w8pnq               0/1     ContainerCreating   0           2s
ubuntu@ip-172-31-43-236:~$
```

Below the terminal output, a metadata box for the instance **i-0cfaecb024b4f1a5d (Kubernetes-Master)** is visible, showing PublicIPs: 35.154.127.248 and PrivateIPs: 172.31.43.236.

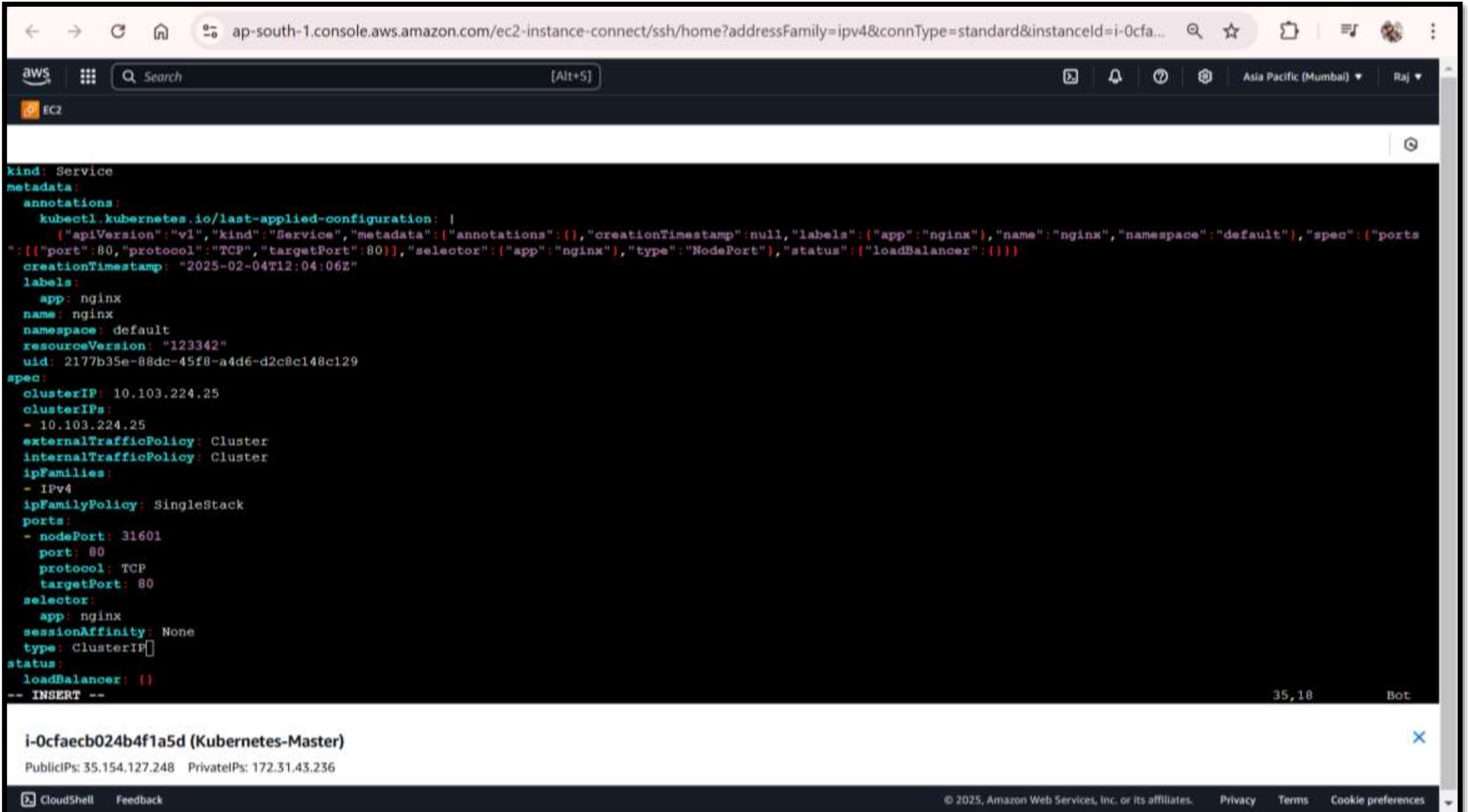
The footer of the CloudShell window includes the text "CloudShell Feedback" and a copyright notice: "© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences".

MODULE 7: KUBERNETES ASSIGNMENTS -4

Tasks To Be Performed:

1. Use the previous deployment
2. Change the service type to ClusterIP

Changing service type NodePort to Cluster IP



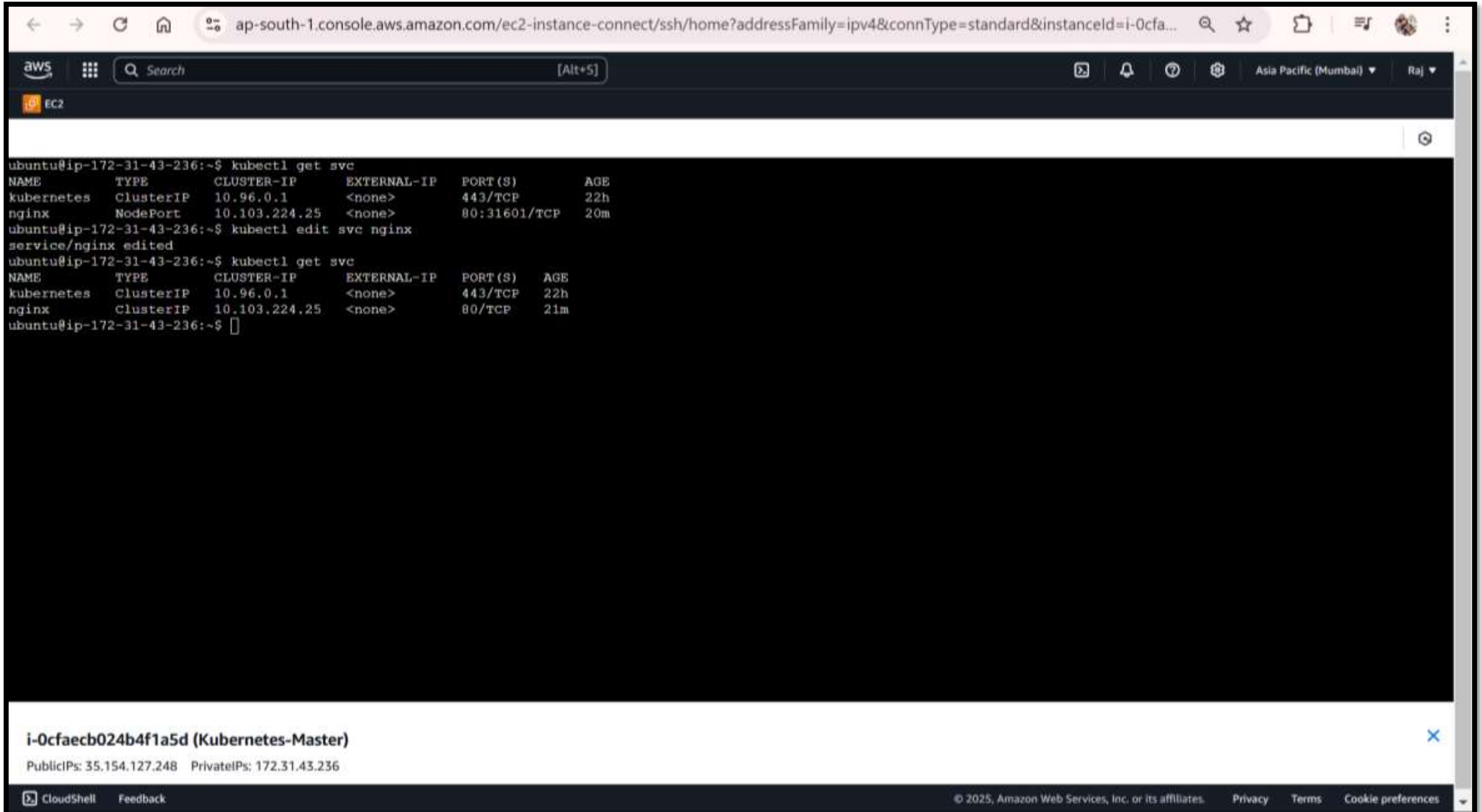
The screenshot shows the AWS Management Console interface. At the top, the browser address bar displays the URL: `ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh/home?addressFamily=ipv4&connType=standard&instanceId=i-0cfa...`. The console header includes the AWS logo, a search bar, and navigation icons. The main content area shows the configuration of an EC2 instance, specifically the CloudShell terminal output for a Kubernetes Service.

```
kind: Service
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","kind":"Service","metadata":{"annotations":{},"creationTimestamp":null,"labels":{"app":"nginx"},"name":"nginx","namespace":"default"},"spec":{"ports":[{"port":80,"protocol":"TCP","targetPort":80}],"selector":{"app":"nginx"},"type":"NodePort"},"status":{"loadBalancer":{}}}
    creationTimestamp: "2025-02-04T12:04:06Z"
  labels:
    app: nginx
  name: nginx
  namespace: default
  resourceVersion: "123342"
  uid: 2177b35e-88dc-45f8-a4d6-d2c8c148c129
spec:
  clusterIP: 10.103.224.25
  clusterIPs:
  - 10.103.224.25
  externalTrafficPolicy: Cluster
  internalTrafficPolicy: Cluster
  ipFamilies:
  - IPv4
  ipFamilyPolicy: SingleStack
  ports:
  - nodePort: 31601
    port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: nginx
  sessionAffinity: None
  type: ClusterIP[]
status:
  loadBalancer: {}
-- INSERT --
```

Below the terminal output, the instance details for `i-0cfaecb024b4f1a5d (Kubernetes-Master)` are visible, showing PublicIPs: 35.154.127.248 and PrivateIPs: 172.31.43.236.

At the bottom of the console, there is a footer with the text: © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences.

Validate service type NodePort to Cluster IP is changed



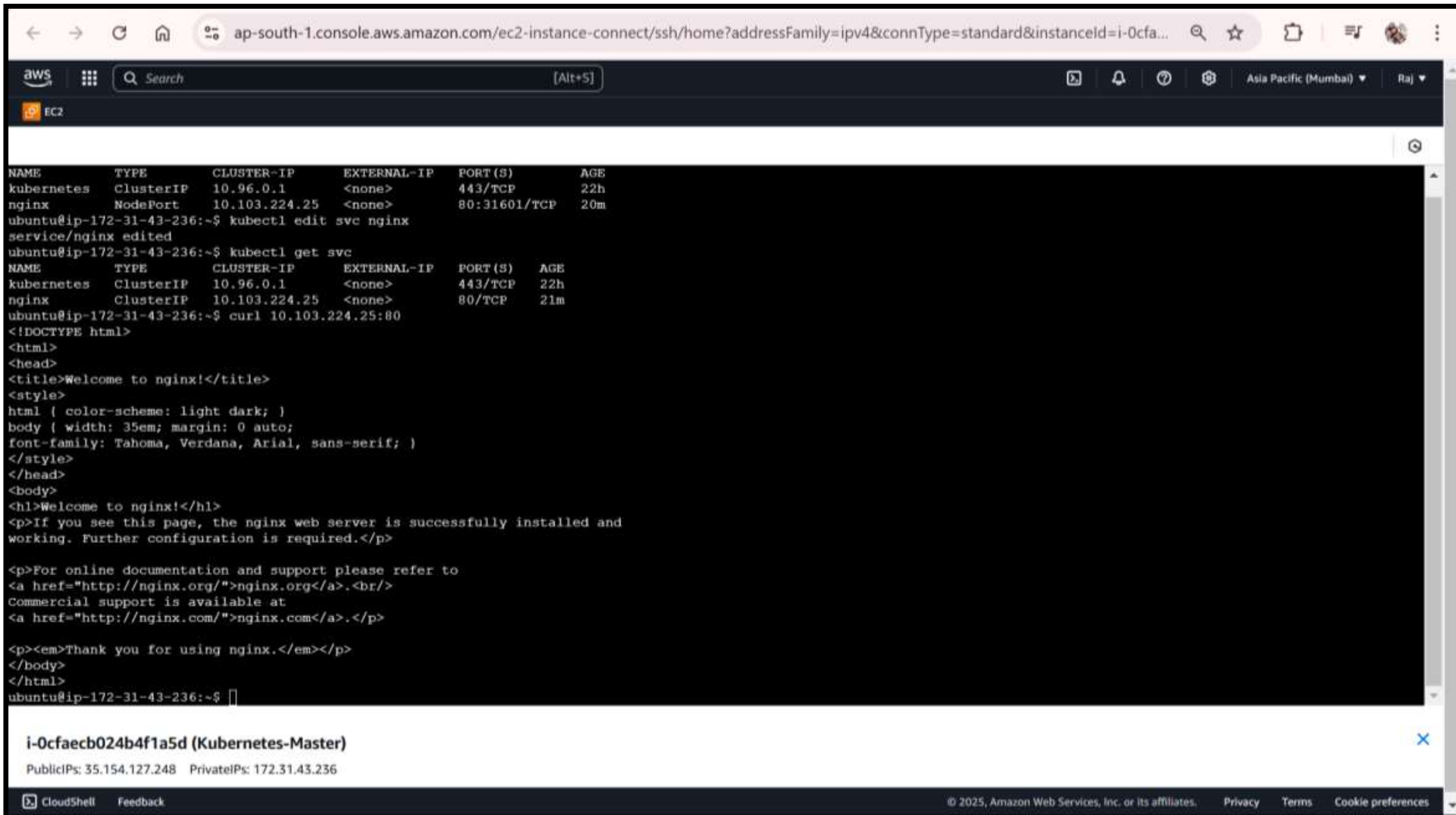
The screenshot shows the AWS CloudShell interface. The terminal window displays the following commands and output:

```
ubuntu@ip-172-31-43-236:~$ kubectl get svc
NAME          TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
kubernetes    ClusterIP     10.96.0.1     <none>         443/TCP          22h
nginx         NodePort      10.103.224.25 <none>         80:31601/TCP     20m
ubuntu@ip-172-31-43-236:~$ kubectl edit svc nginx
service/nginx edited
ubuntu@ip-172-31-43-236:~$ kubectl get svc
NAME          TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
kubernetes    ClusterIP     10.96.0.1     <none>         443/TCP          22h
nginx         ClusterIP     10.103.224.25 <none>         80/TCP           21m
ubuntu@ip-172-31-43-236:~$
```

Below the terminal window, the instance details for **i-0cfaecb024b4f1a5d (Kubernetes-Master)** are shown, including Public IPs (35.154.127.248) and Private IPs (172.31.43.236).

The footer of the CloudShell interface includes the text "CloudShell Feedback" and "© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences".

Validate Cluster IP server we call internally



The screenshot shows the AWS Management Console for an EC2 instance named **i-0cfaecb024b4f1a5d (Kubernetes-Master)**. The terminal window displays the following commands and output:

```
ubuntu@ip-172-31-43-236:~$ kubectl edit svc nginx
service/nginx edited
ubuntu@ip-172-31-43-236:~$ kubectl get svc
NAME         TYPE          CLUSTER-IP    EXTERNAL-IP  PORT(S)    AGE
kubernetes   ClusterIP     10.96.0.1     <none>       443/TCP    22h
nginx        ClusterIP     10.103.224.25 <none>       80/TCP     21m
ubuntu@ip-172-31-43-236:~$ curl 10.103.224.25:80
<!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>
ubuntu@ip-172-31-43-236:~$
```

Below the terminal window, the instance details are shown:

i-0cfaecb024b4f1a5d (Kubernetes-Master)
PublicIPs: 35.154.127.248 PrivateIPs: 172.31.43.236

The footer of the console shows the CloudShell logo, a Feedback link, and the copyright notice: © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences.

MODULE 7: KUBERNETES ASSIGNMENTS -5

Tasks To Be Performed:

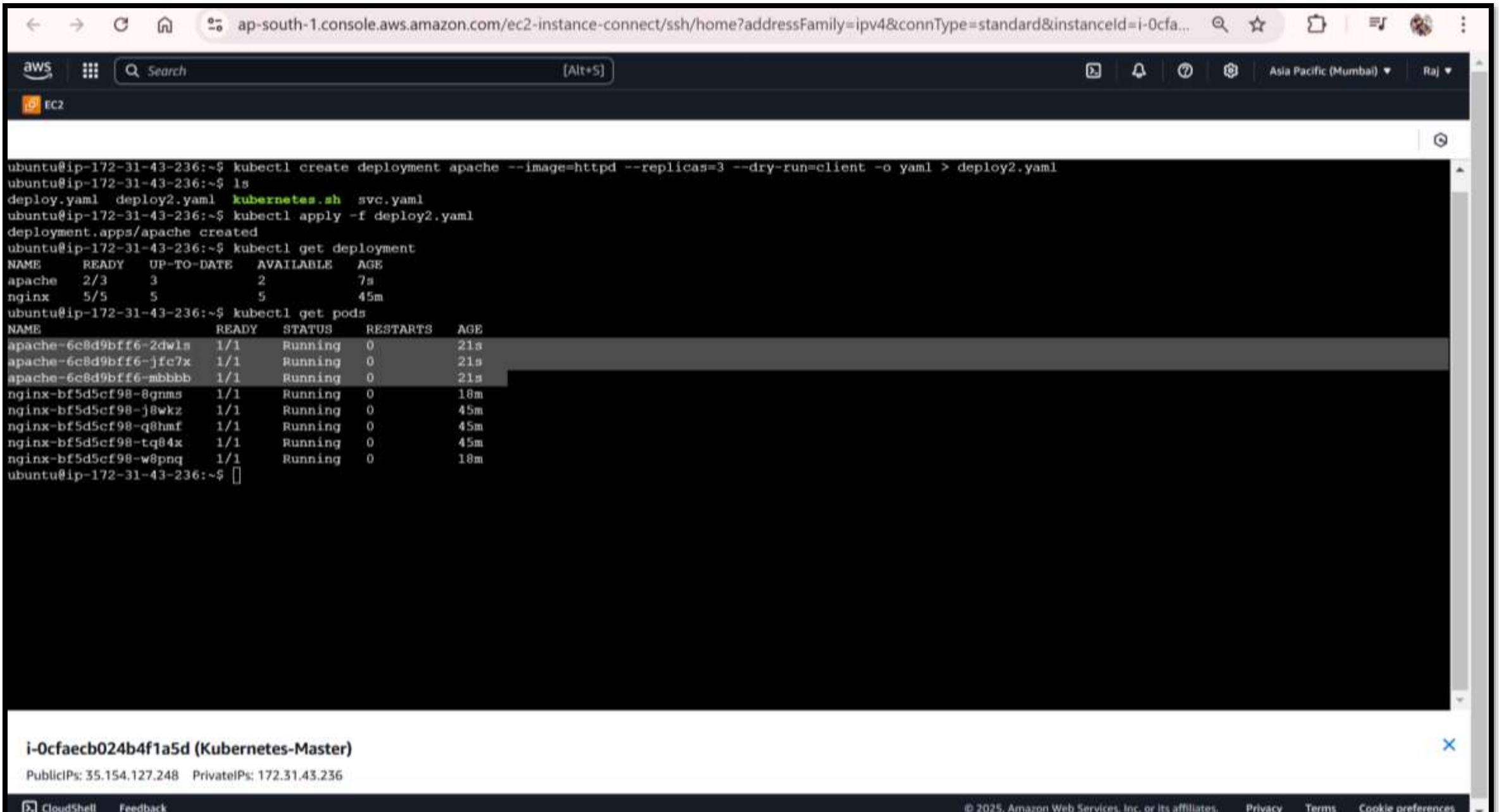
1. Use the previous deployment
2. Deploy an NGINX deployment of 3 replicas
3. Create an NGINX service of type ClusterIP
4. Create an ingress service/ Apache to Apache service/ NGINX to NGINX service

Note : as the ingress service is in freeze as team guide for 5 assignment I have implemented

Deploying a Apache replicas

1. `kubectl create deployment apache --image=httpd --replicas=3 --dry-run=client -o yaml > deploy2.yaml`
2. `kubectl apply -f deploy2.yaml`
3. `kubectl get deployment`
4. `kubectl get pods`

Validating Deployed a Apache replicas



The screenshot shows the AWS Management Console interface for an EC2 instance. The terminal window displays the following commands and output:

```
ubuntu@ip-172-31-43-236:~$ kubectl create deployment apache --image=httpd --replicas=3 --dry-run=client -o yaml > deploy2.yaml
ubuntu@ip-172-31-43-236:~$ ls
deploy.yaml  deploy2.yaml  kubernetes.sh  svc.yaml
ubuntu@ip-172-31-43-236:~$ kubectl apply -f deploy2.yaml
deployment.apps/apache created
ubuntu@ip-172-31-43-236:~$ kubectl get deployment
```

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
apache	2/3	3	2	7s
nginx	5/5	5	5	45m

```
ubuntu@ip-172-31-43-236:~$ kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
apache-6c8d9bff6-2dw1s	1/1	Running	0	21s
apache-6c8d9bff6-jfc7x	1/1	Running	0	21s
apache-6c8d9bff6-mbbbbb	1/1	Running	0	21s
nginx-bf5d5cf98-8qnms	1/1	Running	0	18m
nginx-bf5d5cf98-j8wkz	1/1	Running	0	45m
nginx-bf5d5cf98-q8hmf	1/1	Running	0	45m
nginx-bf5d5cf98-tq84x	1/1	Running	0	45m
nginx-bf5d5cf98-w8pnq	1/1	Running	0	18m

```
ubuntu@ip-172-31-43-236:~$
```

i-0cfaecb024b4f1a5d (Kubernetes-Master)
PublicIPs: 35.154.127.248 PrivateIPs: 172.31.43.236

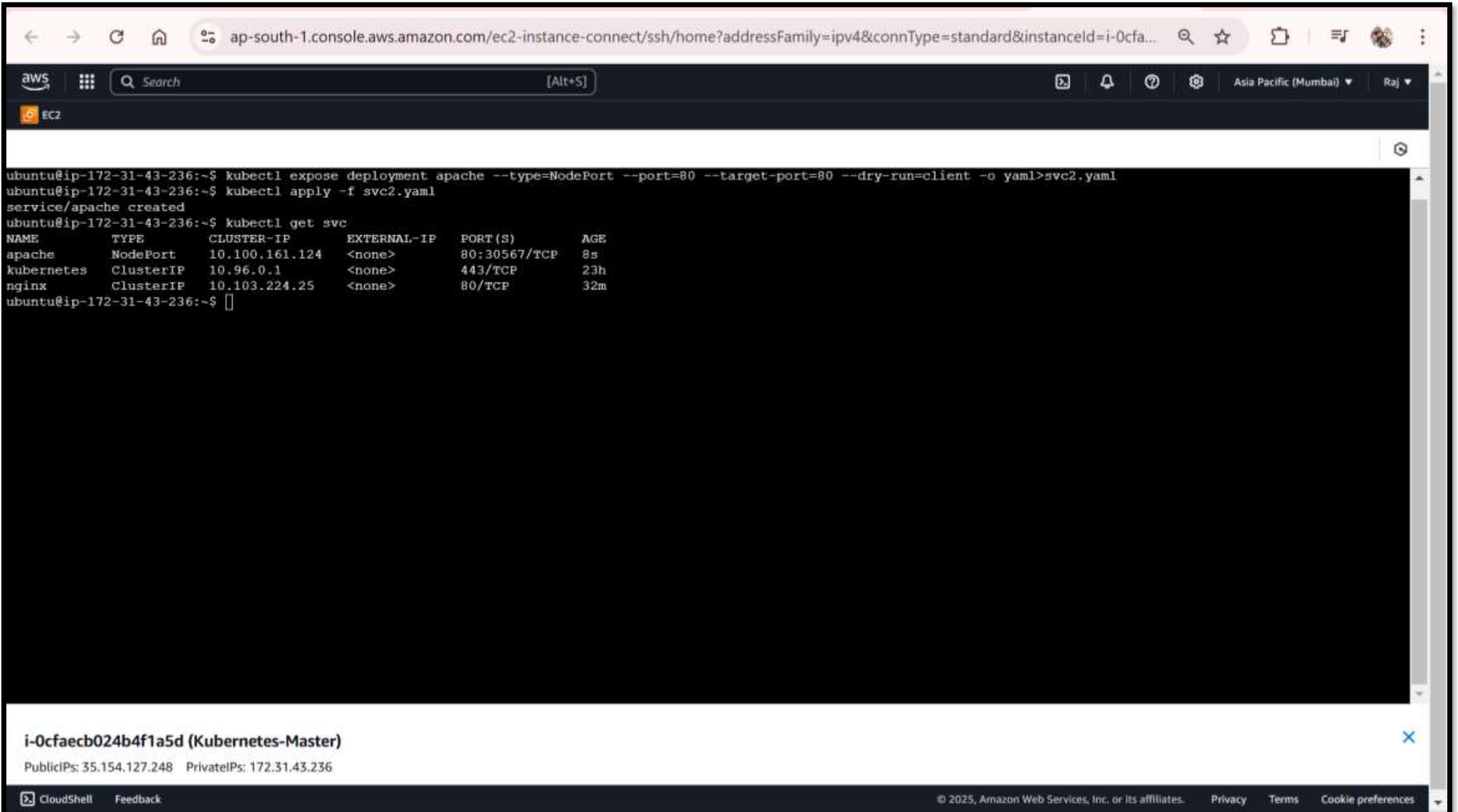
CloudShell Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Deploying Apache NodePort Service

1. `kubectl expose deployment apache --type=NodePort --port=80 --target-port=80 --dry-run=client -o yaml>svc2.yaml`
2. `kubectl apply -f svc2.yaml`
3. `kubectl get svc`

Validating Apache NodePort Service



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

```
ubuntu@ip-172-31-43-236:~$ kubectl expose deployment apache --type=NodePort --port=80 --target-port=80 --dry-run=client -o yaml>svc2.yaml
ubuntu@ip-172-31-43-236:~$ kubectl apply -f svc2.yaml
service/apache created
ubuntu@ip-172-31-43-236:~$ kubectl get svc
```

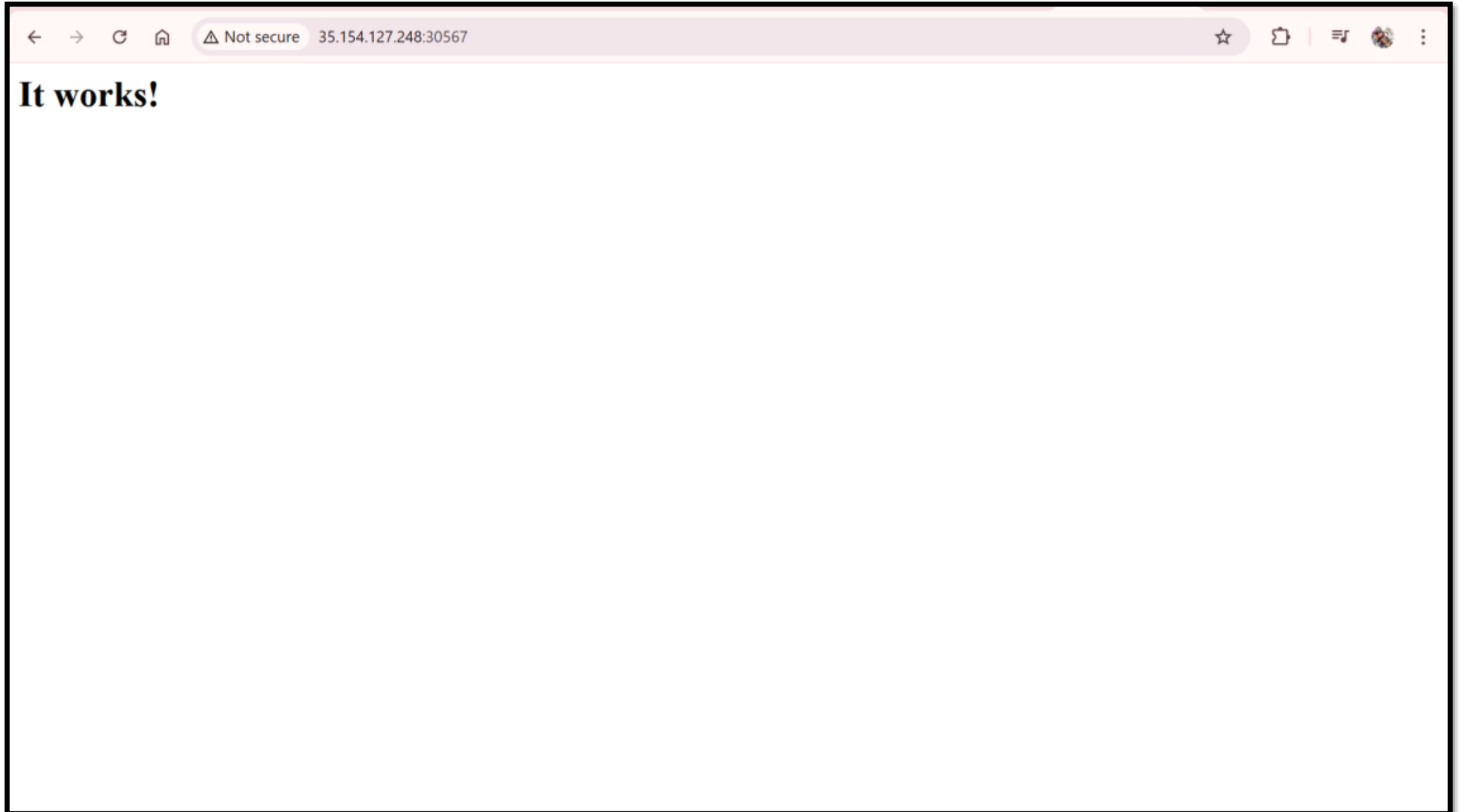
NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
apache	NodePort	10.100.161.124	<none>	80:30567/TCP	8s
kubernetes	ClusterIP	10.96.0.1	<none>	443/TCP	23h
nginx	ClusterIP	10.103.224.25	<none>	80/TCP	32m

```
ubuntu@ip-172-31-43-236:~$
```

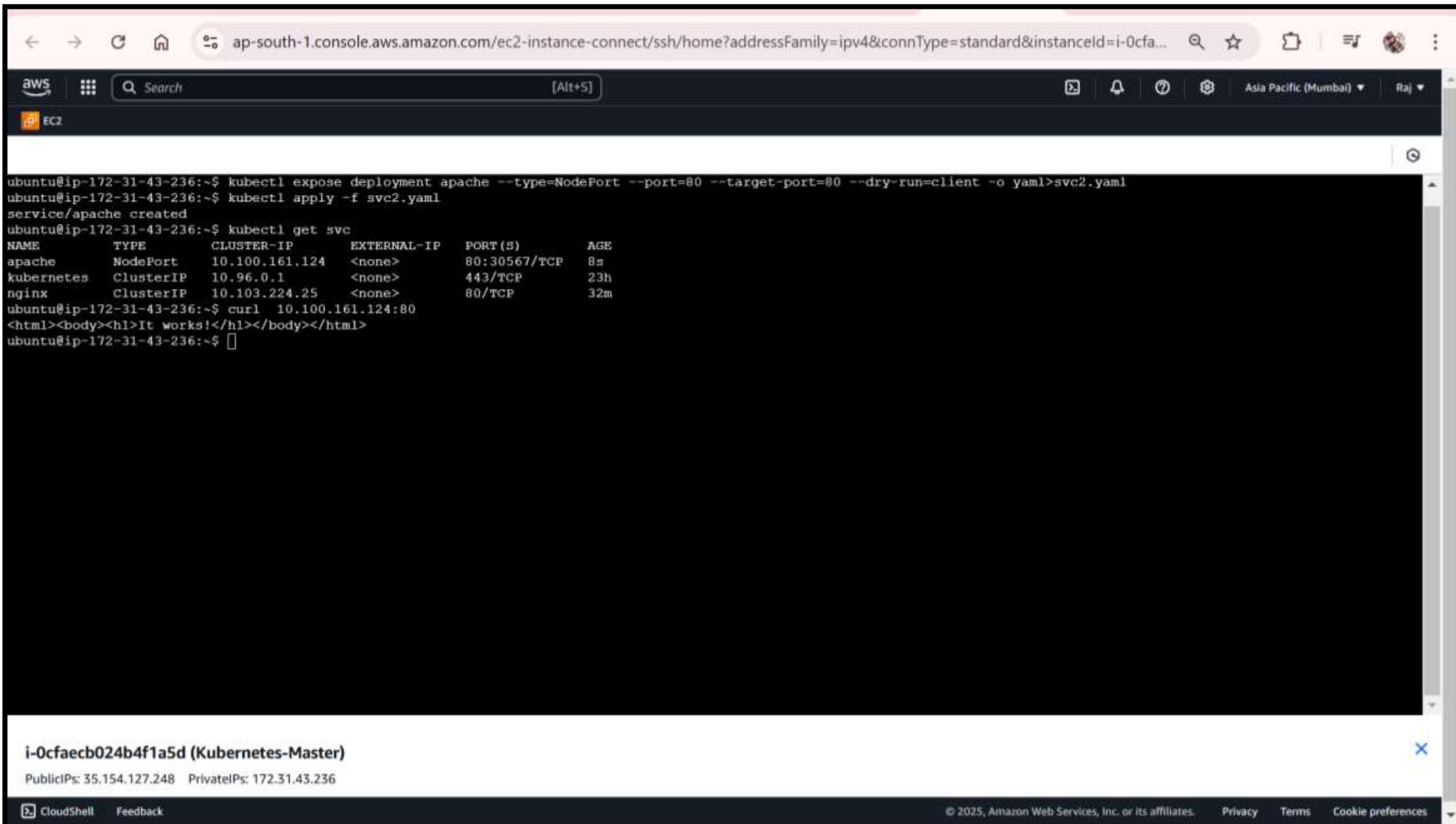
Below the terminal window, the instance details for **i-0cfaecb024b4f1a5d (Kubernetes-Master)** are shown, including PublicIPs: 35.154.127.248 and PrivateIPs: 172.31.43.236.

The footer of the CloudShell interface includes the CloudShell logo, a Feedback link, and the copyright notice: © 2025, Amazon Web Services, Inc. or its affiliates. It also contains links for Privacy, Terms, and Cookie preferences.

We can access Apache NodePort Service in browser



Validating page content using curl command



The screenshot shows the AWS Management Console interface for an EC2 instance. The terminal window displays the following commands and output:

```
ubuntu@ip-172-31-43-236:~$ kubectl expose deployment apache --type=NodePort --port=80 --target-port=80 --dry-run=client -o yaml>svc2.yaml
ubuntu@ip-172-31-43-236:~$ kubectl apply -f svc2.yaml
service/apache created
ubuntu@ip-172-31-43-236:~$ kubectl get svc
NAME         TYPE          CLUSTER-IP    EXTERNAL-IP  PORT(S)          AGE
apache       NodePort      10.100.161.124 <none>       80:30567/TCP     8s
kubernetes   ClusterIP     10.96.0.1     <none>       443/TCP          23h
nginx        ClusterIP     10.103.224.25 <none>       80/TCP           32m
ubuntu@ip-172-31-43-236:~$ curl 10.100.161.124:80
<html><body><h1>It works!</h1></body></html>
ubuntu@ip-172-31-43-236:~$
```

Below the terminal window, the instance details for **i-0cfaecb024b4f1a5d (Kubernetes-Master)** are shown, including PublicIPs: 35.154.127.248 and PrivateIPs: 172.31.43.236.

The footer of the console shows the CloudShell logo, a Feedback link, and copyright information: © 2025, Amazon Web Services, Inc. or its affiliates. It also includes links for Privacy, Terms, and Cookie preferences.