Assignment No-1

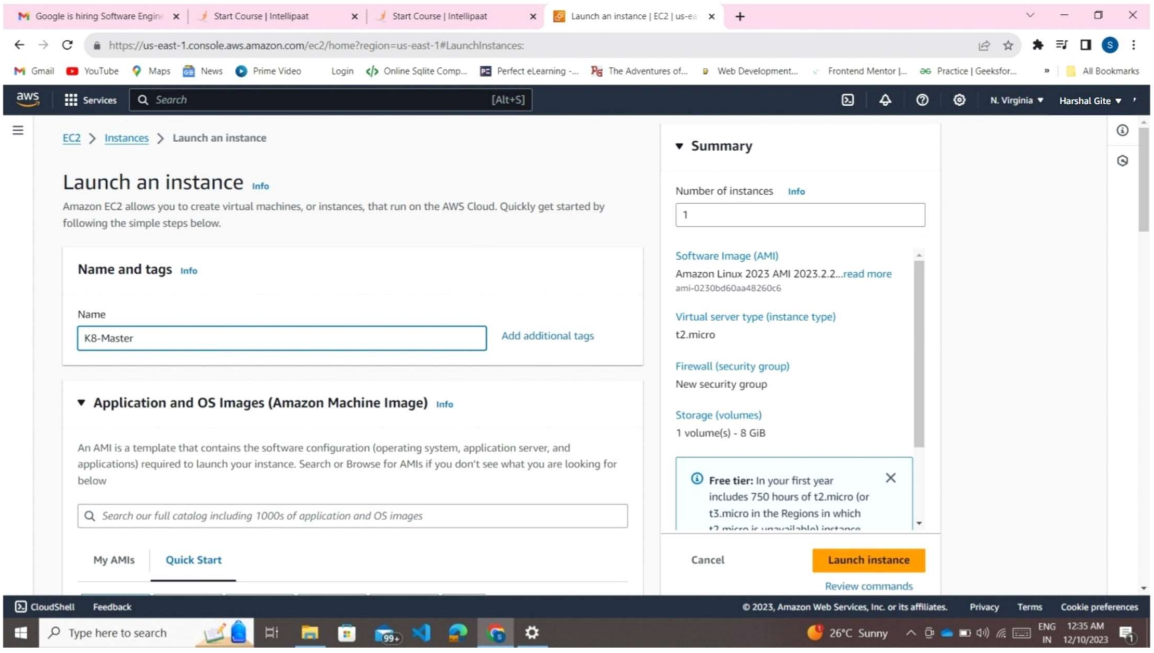
Email: harshalgite888@gmail.com

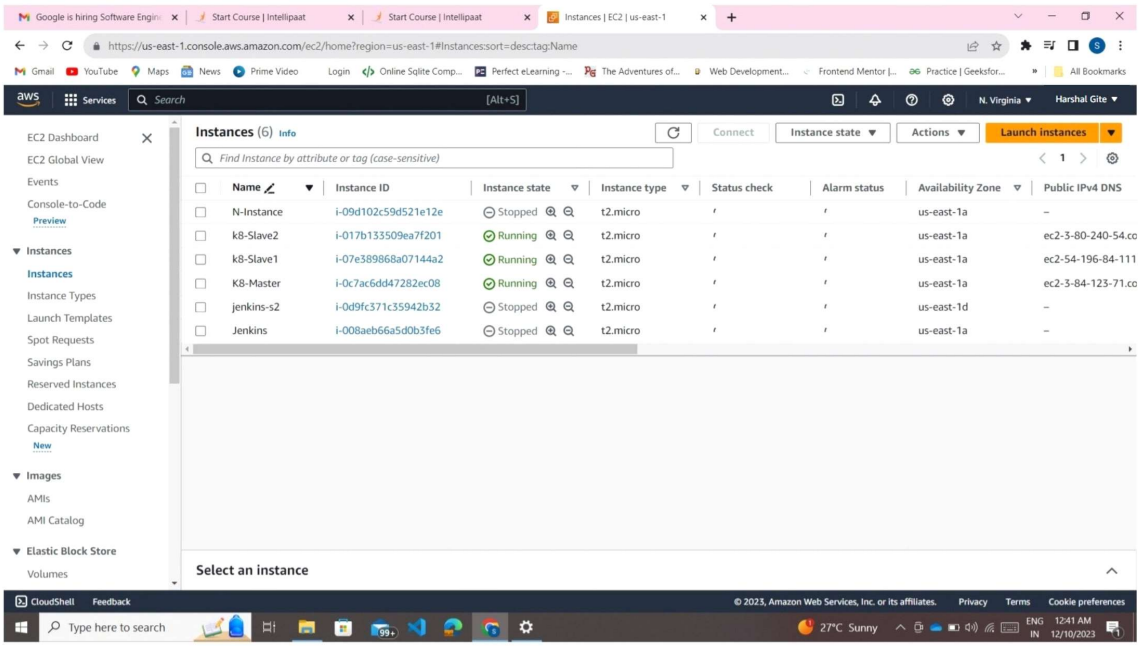
Tasks:

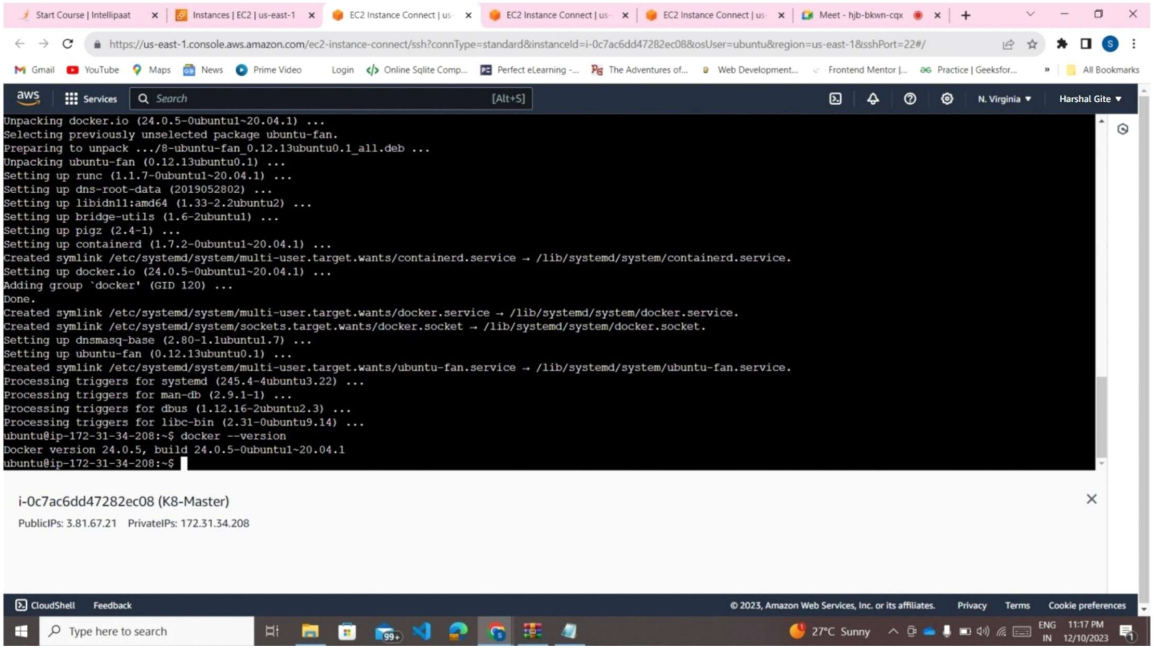
1. Deploy a Kubernetes cluster for 3 nodes.
2. Create a NGINX deployment of 3 replicas.

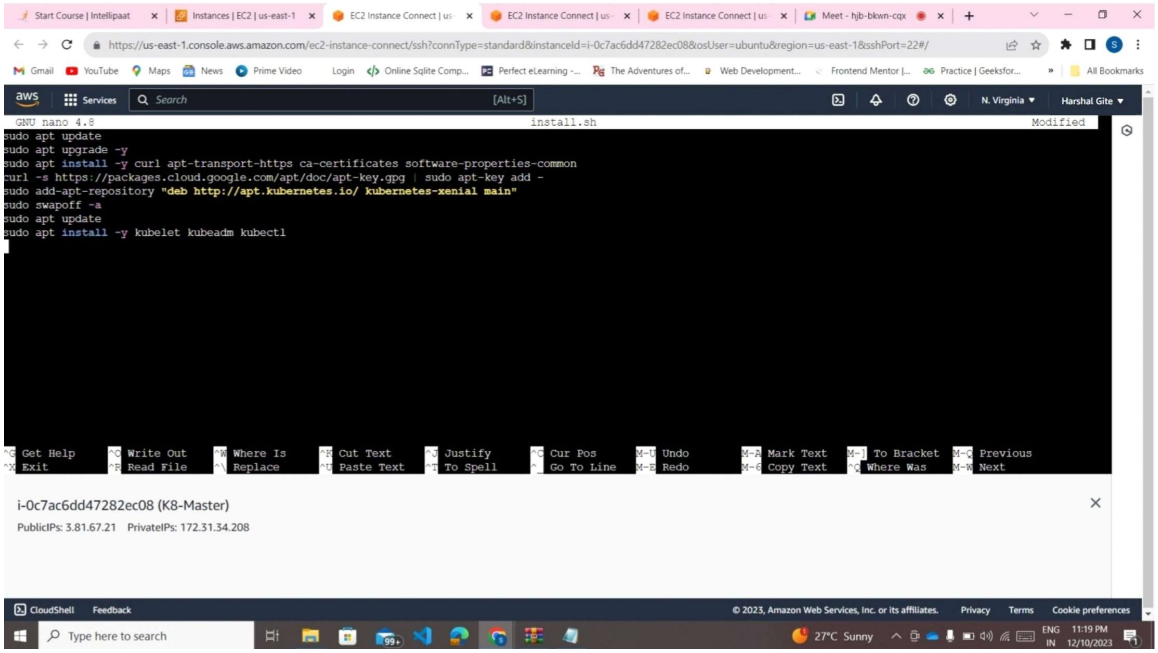
Solution:

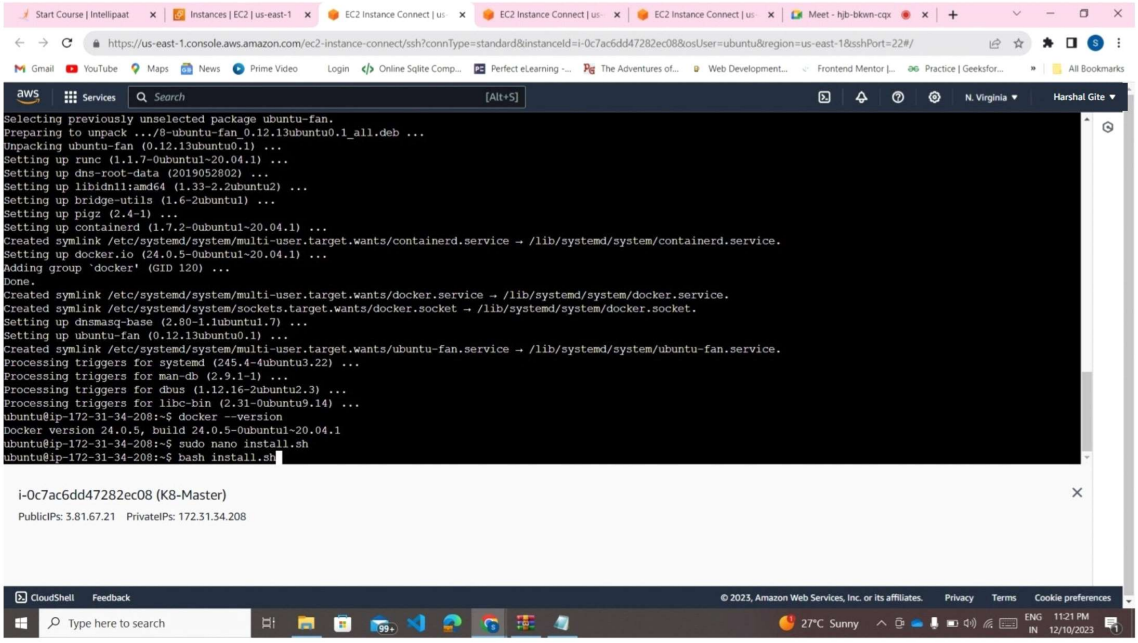
Steps:

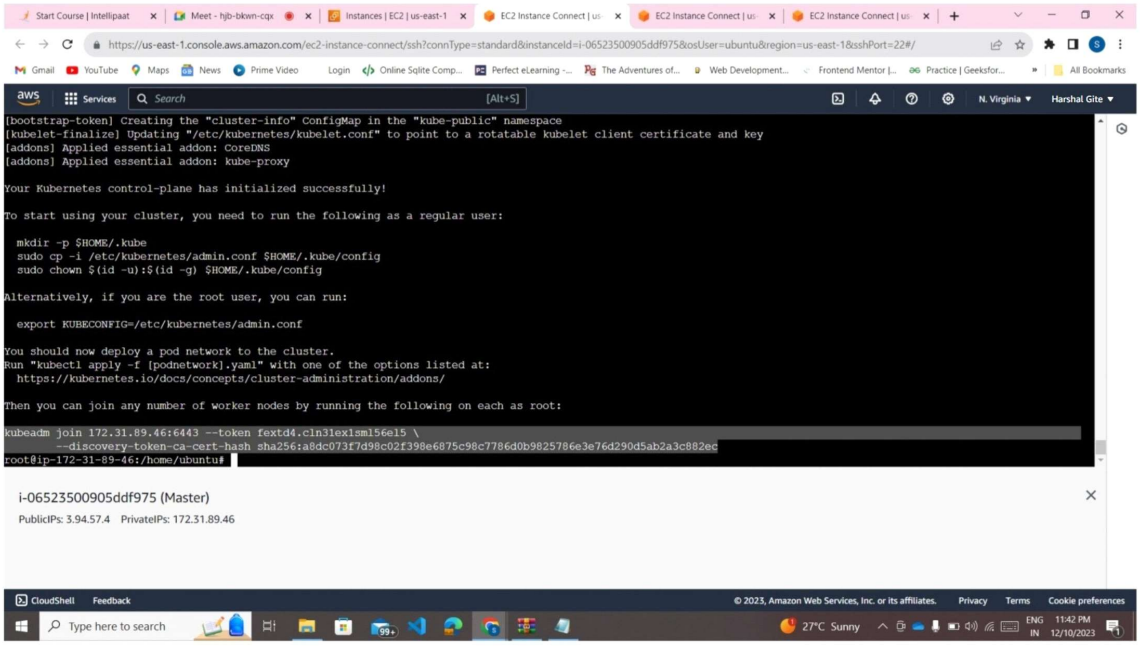


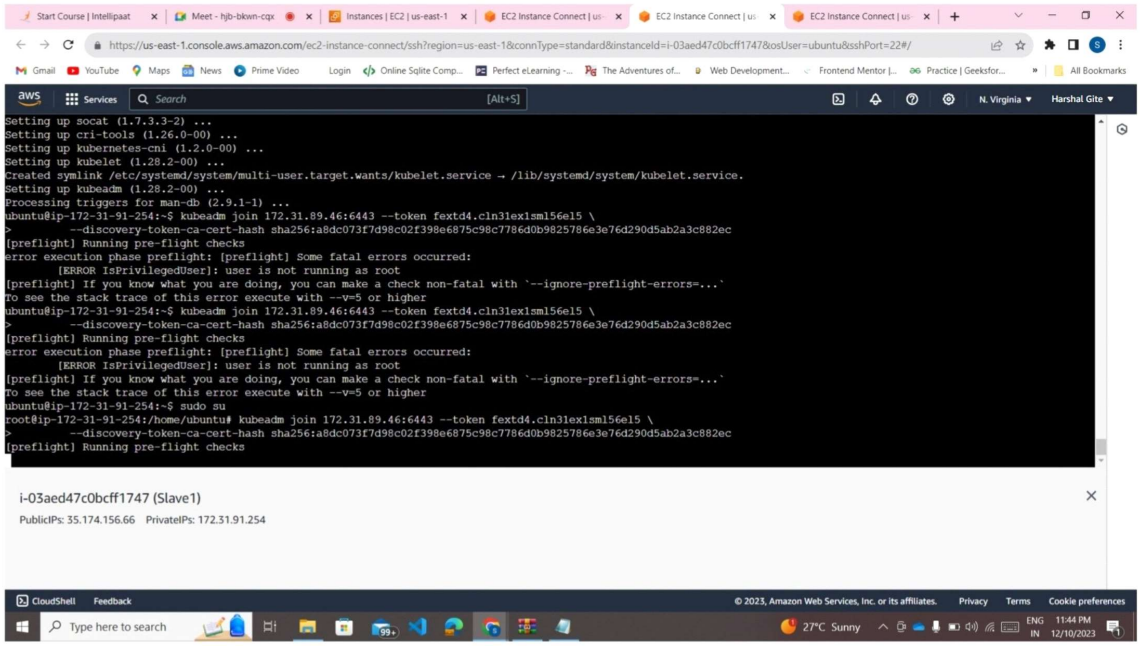

Instances (6) Info
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public IPv4 DNS
N-Instance i-09d102c59d521e12e Stopped t2.micro us-east-1a –
k8-Slave2 i-017b133509ea7f201 Running t2.micro us-east-1a ec2-3-80-240-54.co
k8-Slave1 i-07e389868a07144a2 Running t2.micro us-east-1a ec2-54-196-84-111
K8-Master i-0c7ac6dd47282ec08 Running t2.micro us-east-1a ec2-3-84-123-71.co
jenkins-s2 i-0d9fc371c35942b32 Stopped t2.micro us-east-1d –
Jenkins i-008aeb66a5d0b3fe6 Stopped t2.micro us-east-1a –
Select an instance

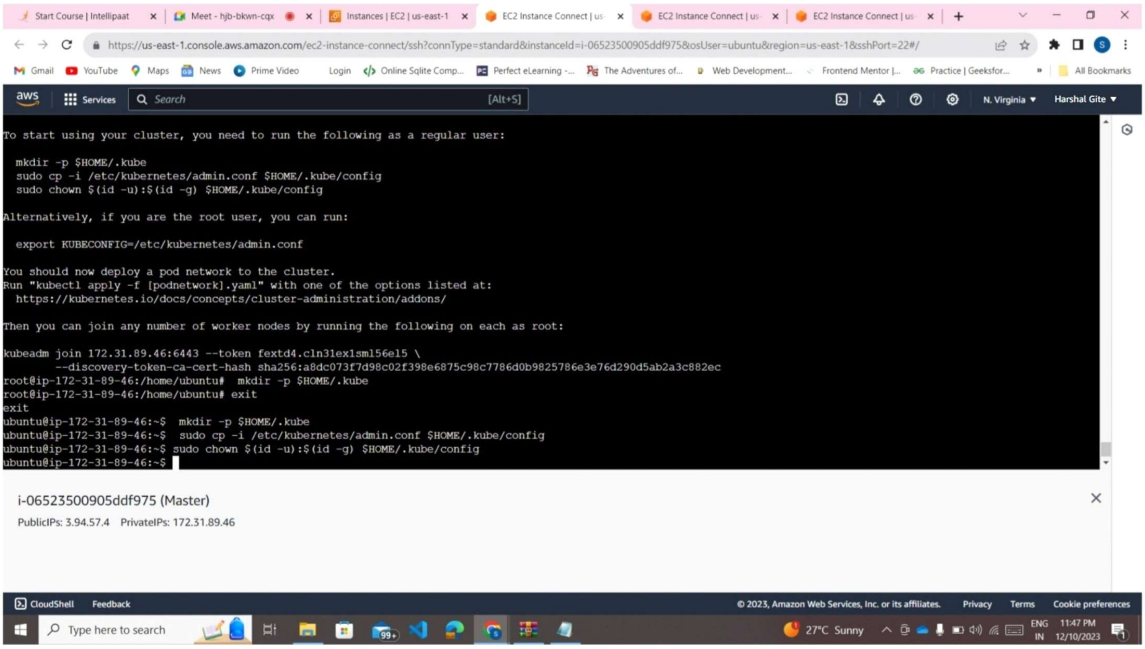

Unpacking docker.io (24.0.5-0ubuntu1~20.04.1) ...
Selecting previously unselected package ubuntu-fan.
Preparing to unpack .../8-ubuntu-fan_0.12.13ubuntu0.1_all.deb ...
Unpacking ubuntu-fan (0.12.13ubuntu0.1) ...
Setting up runc (1.1.7-0ubuntu1~20.04.1) ...
Setting up dns-root-data (2019052802) ...
Setting up libidn11:amd64 (1.33-2.2ubuntu2) ...
Setting up bridge-utils (1.6-2ubuntu1) ...
Setting up pigz (2.4-1) ...
Setting up containerd (1.7.2-0ubuntu1~20.04.1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /lib/systemd/system/containerd.service.
Setting up docker.io (24.0.5-0ubuntu1~20.04.1) ...
Adding group `docker' (GID 120) ...
Done.
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
Setting up dnsmasq-base (2.80-1.1ubuntu1.7) ...
Setting up ubuntu-fan (0.12.13ubuntu0.1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/ubuntu-fan.service → /lib/systemd/system/ubuntu-fan.service.
Processing triggers for systemd (245.4-4ubuntu3.22) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for dbus (1.12.16-2ubuntu2.3) ...
Processing triggers for libc-bin (2.31-0ubuntu9.14) ...
ubuntu@ip-172-31-34-208:~$ docker --version
Docker version 24.0.5, build 24.0.5-0ubuntu1~20.04.1
ubuntu@ip-172-31-34-208:~$
i-0c7ac6dd47282ec08 (K8-Master)
PublicIPs: 3.81.67.21 PrivateIPs: 172.31.34.208

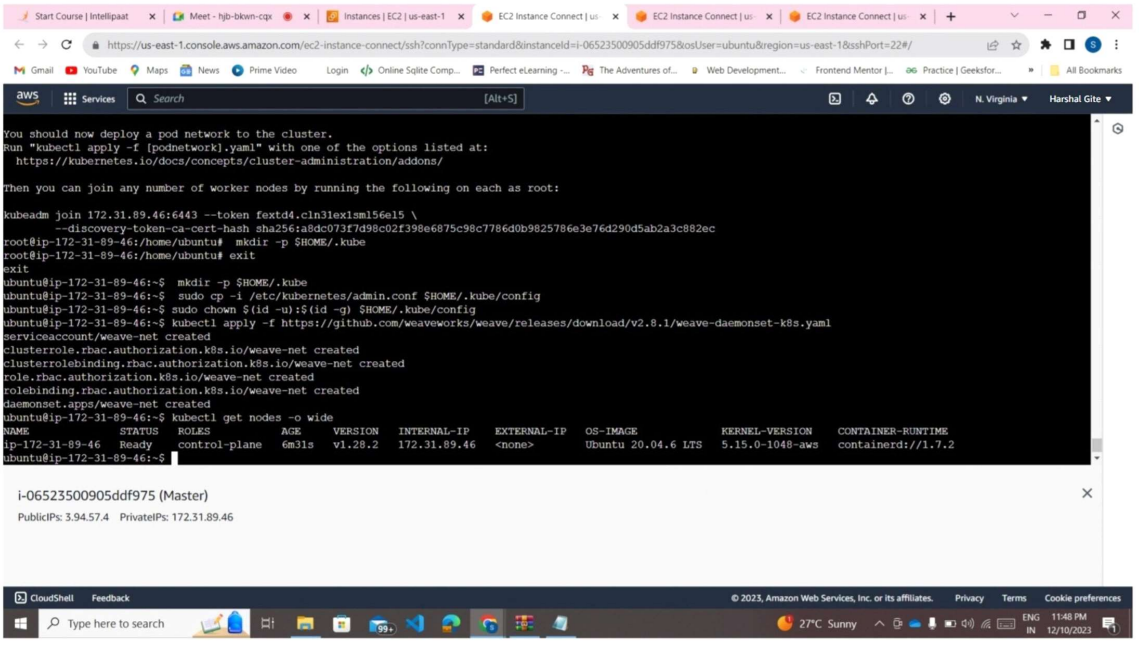

GNU nano 4.8 install.sh Modified
sudo apt update
sudo apt upgrade -y
sudo apt install -y curl apt-transport-https ca-certificates software-properties-common
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -
sudo add-apt-repository "deb http://apt.kubernetes.io/ kubernetes-xenial main"
sudo swapoff -a
sudo apt update
sudo apt install -y kubelet kubeadm kubectl
i-0c7ac6dd47282ec08 (K8-Master)
PublicIPs: 3.81.67.21 PrivateIPs: 172.31.34.208

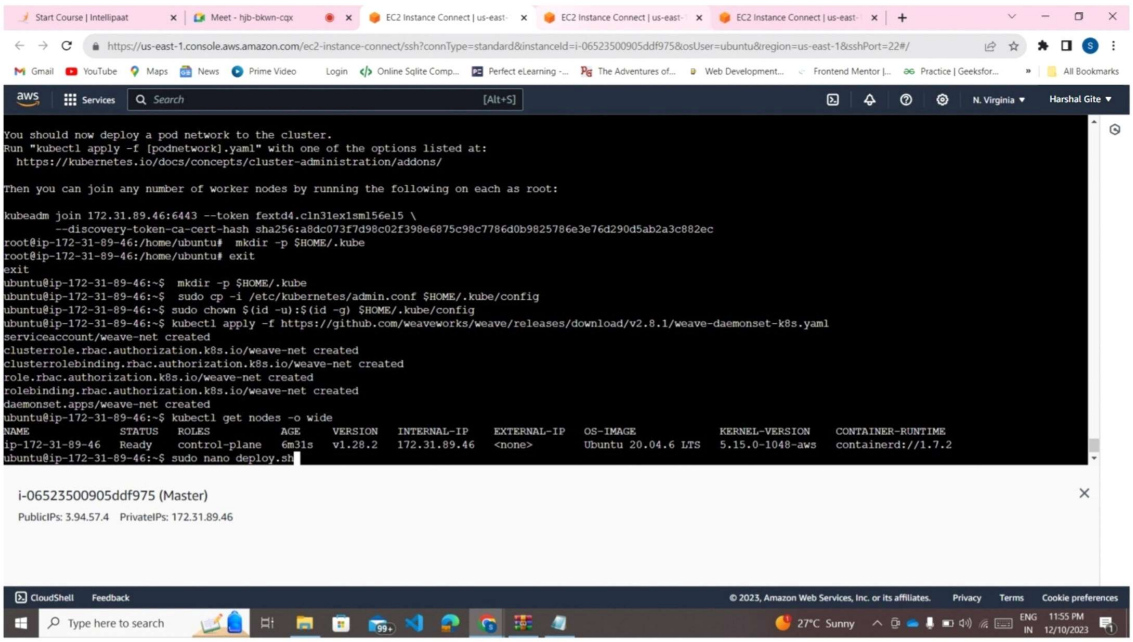

Selecting previously unselected package ubuntu-fan.
Preparing to unpack .../8-ubuntu-fan_0.12.13ubuntu0.1_all.deb ...
Unpacking ubuntu-fan (0.12.13ubuntu0.1) ...
Setting up runc (1.1.7-0ubuntu1~20.04.1) ...
Setting up dns-root-data (2019052802) ...
Setting up libidn11:amd64 (1.33-2.2ubuntu2) ...
Setting up bridge-utils (1.6-2ubuntu1) ...
Setting up pigz (2.4-1) ...
Setting up containerd (1.7.2-0ubuntu1~20.04.1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /lib/systemd/system/containerd.service.
Setting up docker.io (24.0.5-0ubuntu1~20.04.1) ...
Adding group `docker' (GID 120) ...
Done.
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
Setting up dnsmasq-base (2.80-1.1ubuntu1.7) ...
Setting up ubuntu-fan (0.12.13ubuntu0.1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/ubuntu-fan.service → /lib/systemd/system/ubuntu-fan.service.
Processing triggers for systemd (245.4-4ubuntu3.22) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for dbus (1.12.16-2ubuntu2.3) ...
Processing triggers for libc-bin (2.31-0ubuntu9.14) ...
ubuntu@ip-172-31-34-208:~$ docker --version
Docker version 24.0.5, build 24.0.5-0ubuntu1~20.04.1
ubuntu@ip-172-31-34-208:~$ sudo nano install.sh
ubuntu@ip-172-31-34-208:~$ bash install.sh
i-0c7ac6dd47282ec08 (K8-Master)
PublicIPs: 3.81.67.21 PrivateIPs: 172.31.34.208

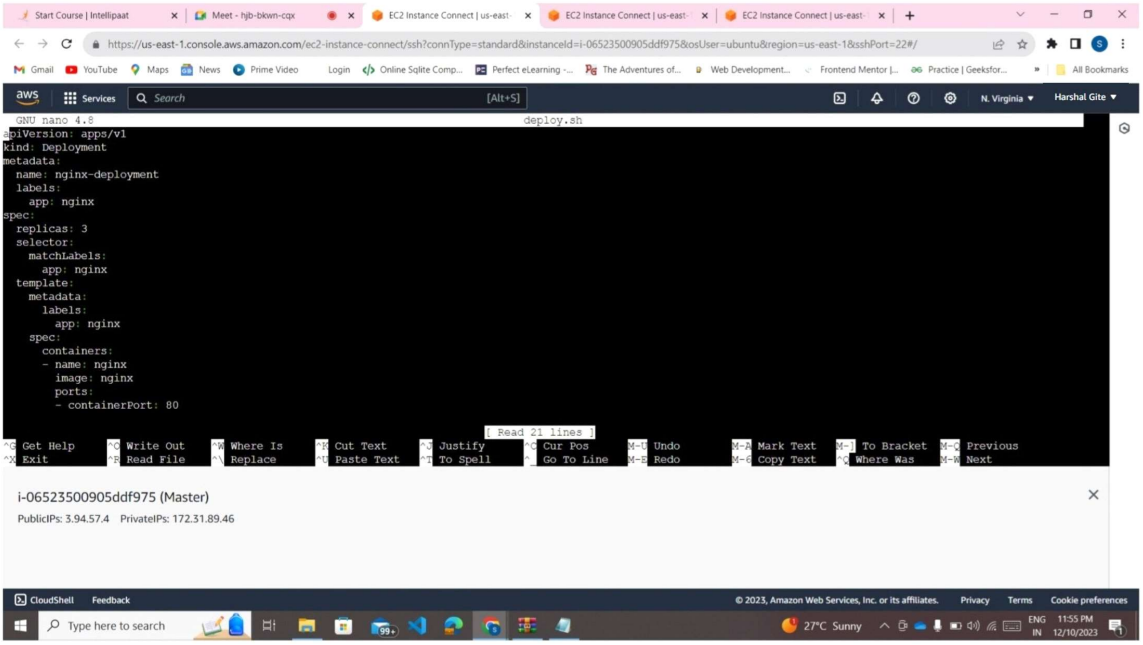

[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy
Your Kubernetes 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 172.31.89.46:6443 --token fextd4.cln31ex1sml56el5 \
--discovery-token-ca-cert-hash sha256:a8dc073f7d98c02f398e6875c98c7786d0b9825786e3e76d290d5ab2a3c882ec
root@ip-172-31-89-46:/home/ubuntu#
i-06523500905ddf975 (Master)
PublicIPs: 3.94.57.4 PrivateIPs: 172.31.89.46

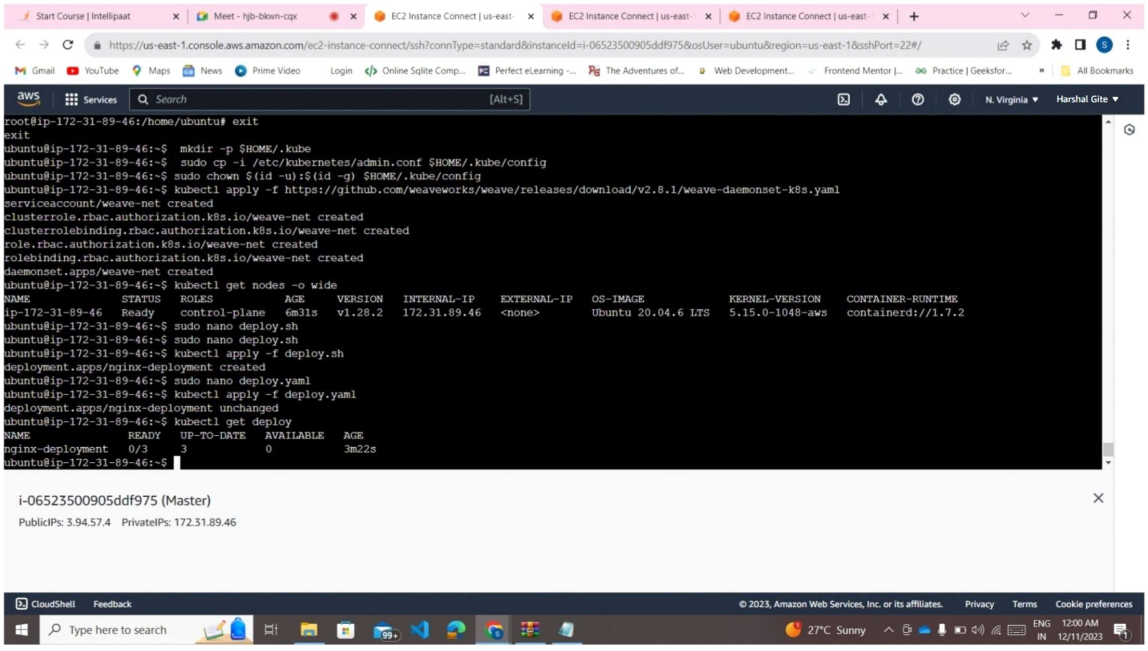

Setting up socat (1.7.3.3-2) ...
Setting up cri-tools (1.26.0-00) ...
Setting up kubernetes-cni (1.2.0-00) ...
Setting up kubelet (1.28.2-00) ...
Created symlink /etc/systemd/system/multi-user.target.wants/kubelet.service → /lib/systemd/system/kubelet.service.
Setting up kubeadm (1.28.2-00) ...
Processing triggers for man-db (2.9.1-1) ...
ubuntu@ip-172-31-91-254:~$ kubeadm join 172.31.89.46:6443 --token fextd4.cln31ex1sml56el5 \
> --discovery-token-ca-cert-hash sha256:a8dc073f7d98c02f398e6875c98c7786d0b9825786e3e76d290d5ab2a3c882ec
[preflight] Running pre-flight checks
error execution phase preflight: [preflight] Some fatal errors occurred:
[ERROR IsPrivilegedUser]: user is not running as root
[preflight] If you know what you are doing, you can make a check non-fatal with `--ignore-preflight-errors=...`
To see the stack trace of this error execute with --v=5 or higher
ubuntu@ip-172-31-91-254:~$ kubeadm join 172.31.89.46:6443 --token fextd4.cln31ex1sml56el5 \
> --discovery-token-ca-cert-hash sha256:a8dc073f7d98c02f398e6875c98c7786d0b9825786e3e76d290d5ab2a3c882ec
[preflight] Running pre-flight checks
error execution phase preflight: [preflight] Some fatal errors occurred:
[ERROR IsPrivilegedUser]: user is not running as root
[preflight] If you know what you are doing, you can make a check non-fatal with `--ignore-preflight-errors=...`
To see the stack trace of this error execute with --v=5 or higher
ubuntu@ip-172-31-91-254:~$ sudo su
root@ip-172-31-91-254:/home/ubuntu# kubeadm join 172.31.89.46:6443 --token fextd4.cln31ex1sml56el5 \
> --discovery-token-ca-cert-hash sha256:a8dc073f7d98c02f398e6875c98c7786d0b9825786e3e76d290d5ab2a3c882ec
[preflight] Running pre-flight checks
i-03aed47c0bcff1747 (Slave1)
PublicIPs: 35.174.156.66 PrivateIPs: 172.31.91.254

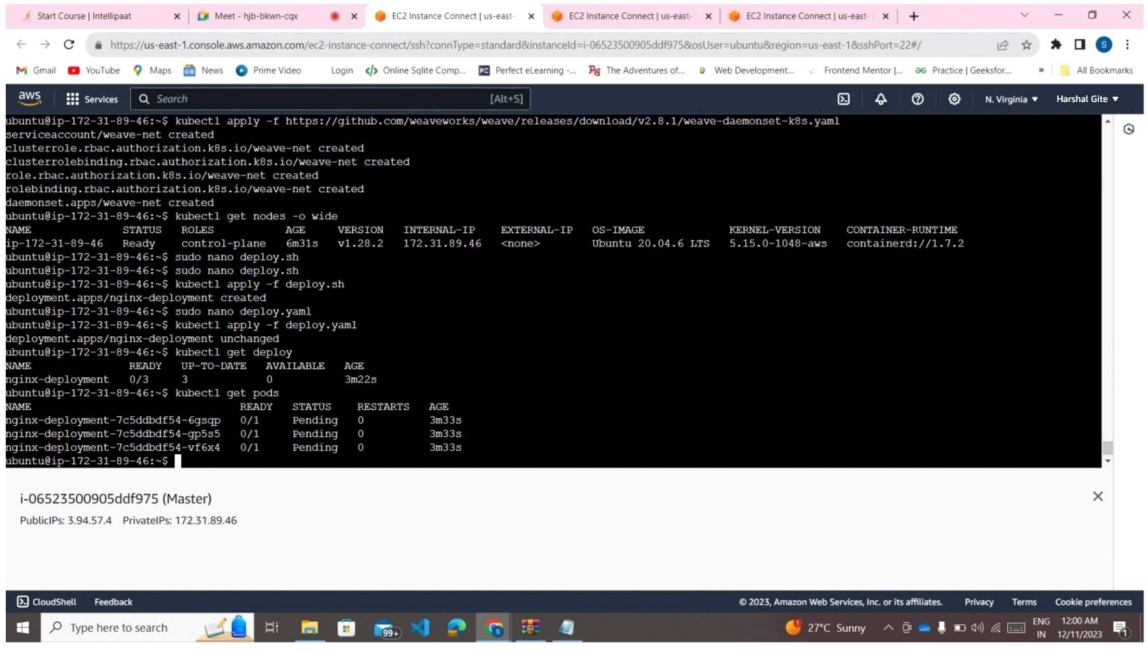

To start using your cluster, you need to run the following as a regular user:
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
Alternatively, if you are the root user, you can run:
export KUBECONFIG=/etc/kubernetes/admin.conf
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 172.31.89.46:6443 --token fextd4.cln31ex1sml56el5 \
--discovery-token-ca-cert-hash sha256:a8dc073f7d98c02f398e6875c98c7786d0b9825786e3e76d290d5ab2a3c882ec
root@ip-172-31-89-46:/home/ubuntu# mkdir -p $HOME/.kube
root@ip-172-31-89-46:/home/ubuntu# exit
exit
ubuntu@ip-172-31-89-46:~$ mkdir -p $HOME/.kube
ubuntu@ip-172-31-89-46:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
ubuntu@ip-172-31-89-46:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-89-46:~$
i-06523500905ddf975 (Master)
PublicIPs: 3.94.57.4 PrivateIPs: 172.31.89.46


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 172.31.89.46:6443 --token fextd4.cln31ex1sml56el5 \
--discovery-token-ca-cert-hash sha256:a8dc073f7d98c02f398e6875c98c7786d0b9825786e3e76d290d5ab2a3c882ec
root@ip-172-31-89-46:/home/ubuntu# mkdir -p $HOME/.kube
root@ip-172-31-89-46:/home/ubuntu# exit
exit
ubuntu@ip-172-31-89-46:~$ mkdir -p $HOME/.kube
ubuntu@ip-172-31-89-46:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
ubuntu@ip-172-31-89-46:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-89-46:~$ kubectl apply -f https://github.com/weaveworks/weave/releases/download/v2.8.1/weave-daemonset-k8s.yaml
serviceaccount/weave-net created
clusterrole.rbac.authorization.k8s.io/weave-net created
clusterrolebinding.rbac.authorization.k8s.io/weave-net created
role.rbac.authorization.k8s.io/weave-net created
rolebinding.rbac.authorization.k8s.io/weave-net created
daemonset.apps/weave-net created
ubuntu@ip-172-31-89-46:~$ kubectl get nodes -o wide
NAME STATUS ROLES AGE VERSION INTERNAL-IP EXTERNAL-IP OS-IMAGE KERNEL-VERSION CONTAINER-RUNTIME
ip-172-31-89-46 Ready control-plane 6m31s v1.28.2 172.31.89.46 <none> Ubuntu 20.04.6 LTS 5.15.0-1048-aws containerd://1.7.2
ubuntu@ip-172-31-89-46:~$
i-06523500905ddf975 (Master)
PublicIPs: 3.94.57.4 PrivateIPs: 172.31.89.46


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 172.31.89.46:6443 --token fextd4.cln31ex1sml56el5 \
--discovery-token-ca-cert-hash sha256:a8dc073f7d98c02f398e6875c98c7786d0b9825786e3e76d290d5ab2a3c882ec
root@ip-172-31-89-46:/home/ubuntu# mkdir -p $HOME/.kube
root@ip-172-31-89-46:/home/ubuntu# exit
exit
ubuntu@ip-172-31-89-46:~$ mkdir -p $HOME/.kube
ubuntu@ip-172-31-89-46:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
ubuntu@ip-172-31-89-46:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-89-46:~$ kubectl apply -f https://github.com/weaveworks/weave/releases/download/v2.8.1/weave-daemonset-k8s.yaml
serviceaccount/weave-net created
clusterrole.rbac.authorization.k8s.io/weave-net created
clusterrolebinding.rbac.authorization.k8s.io/weave-net created
role.rbac.authorization.k8s.io/weave-net created
rolebinding.rbac.authorization.k8s.io/weave-net created
daemonset.apps/weave-net created
ubuntu@ip-172-31-89-46:~$ kubectl get nodes -o wide
NAME STATUS ROLES AGE VERSION INTERNAL-IP EXTERNAL-IP OS-IMAGE KERNEL-VERSION CONTAINER-RUNTIME
ip-172-31-89-46 Ready control-plane 6m31s v1.28.2 172.31.89.46 <none> Ubuntu 20.04.6 LTS 5.15.0-1048-aws containerd://1.7.2
ubuntu@ip-172-31-89-46:~$ sudo nano deploy.sh
i-06523500905ddf975 (Master)
PublicIPs: 3.94.57.4 PrivateIPs: 172.31.89.46


GNU nano 4.8 deploy.sh
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
image: nginx
ports:
- containerPort: 80
[Read 21 lines]
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos M-U Undo M-A Mark Text M-] To Bracket M-Q Previous
^X Exit ^R Read File ^\ Replace ^U Paste Text ^T To Spell ^_ Go To Line M-E Redo M-6 Copy Text ^Q Where Was M-W Next
i-06523500905ddf975 (Master)
PublicIPs: 3.94.57.4 PrivateIPs: 172.31.89.46


root@ip-172-31-89-46:/home/ubuntu# exit
exit
ubuntu@ip-172-31-89-46:~$ mkdir -p $HOME/.kube
ubuntu@ip-172-31-89-46:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
ubuntu@ip-172-31-89-46:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-89-46:~$ kubectl apply -f https://github.com/weaveworks/weave/releases/download/v2.8.1/weave-daemonset-k8s.yaml
serviceaccount/weave-net created
clusterrole.rbac.authorization.k8s.io/weave-net created
clusterrolebinding.rbac.authorization.k8s.io/weave-net created
role.rbac.authorization.k8s.io/weave-net created
rolebinding.rbac.authorization.k8s.io/weave-net created
daemonset.apps/weave-net created
ubuntu@ip-172-31-89-46:~$ kubectl get nodes -o wide
NAME STATUS ROLES AGE VERSION INTERNAL-IP EXTERNAL-IP OS-IMAGE KERNEL-VERSION CONTAINER-RUNTIME
ip-172-31-89-46 Ready control-plane 6m31s v1.28.2 172.31.89.46 <none> Ubuntu 20.04.6 LTS 5.15.0-1048-aws containerd://1.7.2
ubuntu@ip-172-31-89-46:~$ sudo nano deploy.sh
ubuntu@ip-172-31-89-46:~$ sudo nano deploy.sh
ubuntu@ip-172-31-89-46:~$ kubectl apply -f deploy.sh
deployment.apps/nginx-deployment created
ubuntu@ip-172-31-89-46:~$ sudo nano deploy.yaml
ubuntu@ip-172-31-89-46:~$ kubectl apply -f deploy.yaml
deployment.apps/nginx-deployment unchanged
ubuntu@ip-172-31-89-46:~$ kubectl get deploy
NAME READY UP-TO-DATE AVAILABLE AGE
nginx-deployment 0/3 3 0 3m22s
ubuntu@ip-172-31-89-46:~$
i-06523500905ddf975 (Master)
PublicIPs: 3.94.57.4 PrivateIPs: 172.31.89.46


ubuntu@ip-172-31-89-46:~$ kubectl apply -f https://github.com/weaveworks/weave/releases/download/v2.8.1/weave-daemonset-k8s.yaml
serviceaccount/weave-net created
clusterrole.rbac.authorization.k8s.io/weave-net created
clusterrolebinding.rbac.authorization.k8s.io/weave-net created
role.rbac.authorization.k8s.io/weave-net created
rolebinding.rbac.authorization.k8s.io/weave-net created
daemonset.apps/weave-net created
ubuntu@ip-172-31-89-46:~$ kubectl get nodes -o wide
NAME STATUS ROLES AGE VERSION INTERNAL-IP EXTERNAL-IP OS-IMAGE KERNEL-VERSION CONTAINER-RUNTIME
ip-172-31-89-46 Ready control-plane 6m31s v1.28.2 172.31.89.46 <none> Ubuntu 20.04.6 LTS 5.15.0-1048-aws containerd://1.7.2
ubuntu@ip-172-31-89-46:~$ sudo nano deploy.sh
ubuntu@ip-172-31-89-46:~$ sudo nano deploy.sh
ubuntu@ip-172-31-89-46:~$ kubectl apply -f deploy.sh
deployment.apps/nginx-deployment created
ubuntu@ip-172-31-89-46:~$ sudo nano deploy.yaml
ubuntu@ip-172-31-89-46:~$ kubectl apply -f deploy.yaml
deployment.apps/nginx-deployment unchanged
ubuntu@ip-172-31-89-46:~$ kubectl get deploy
NAME READY UP-TO-DATE AVAILABLE AGE
nginx-deployment 0/3 3 0 3m22s
ubuntu@ip-172-31-89-46:~$ kubectl get pods
NAME READY STATUS RESTARTS AGE
nginx-deployment-7c5ddbdf54-6gsqp 0/1 Pending 0 3m33s
nginx-deployment-7c5ddbdf54-gp5s5 0/1 Pending 0 3m33s
nginx-deployment-7c5ddbdf54-vf6x4 0/1 Pending 0 3m33s
ubuntu@ip-172-31-89-46:~$
i-06523500905ddf975 (Master)
PublicIPs: 3.94.57.4 PrivateIPs: 172.31.89.46


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.