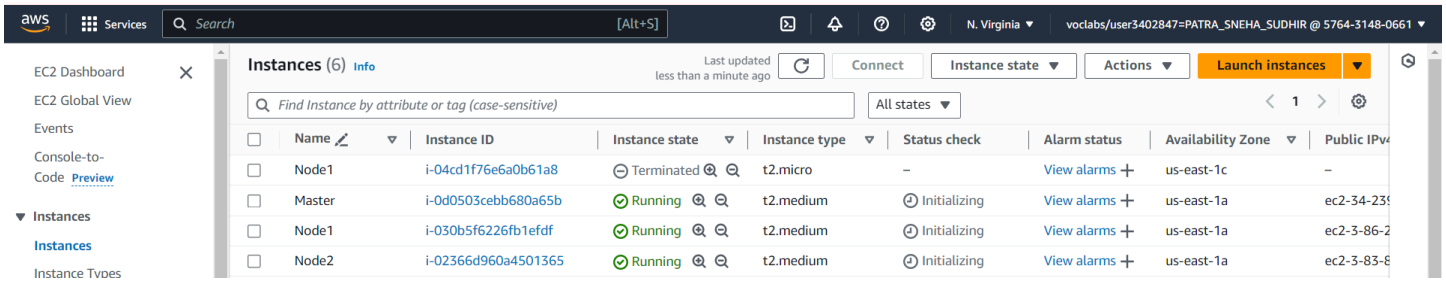
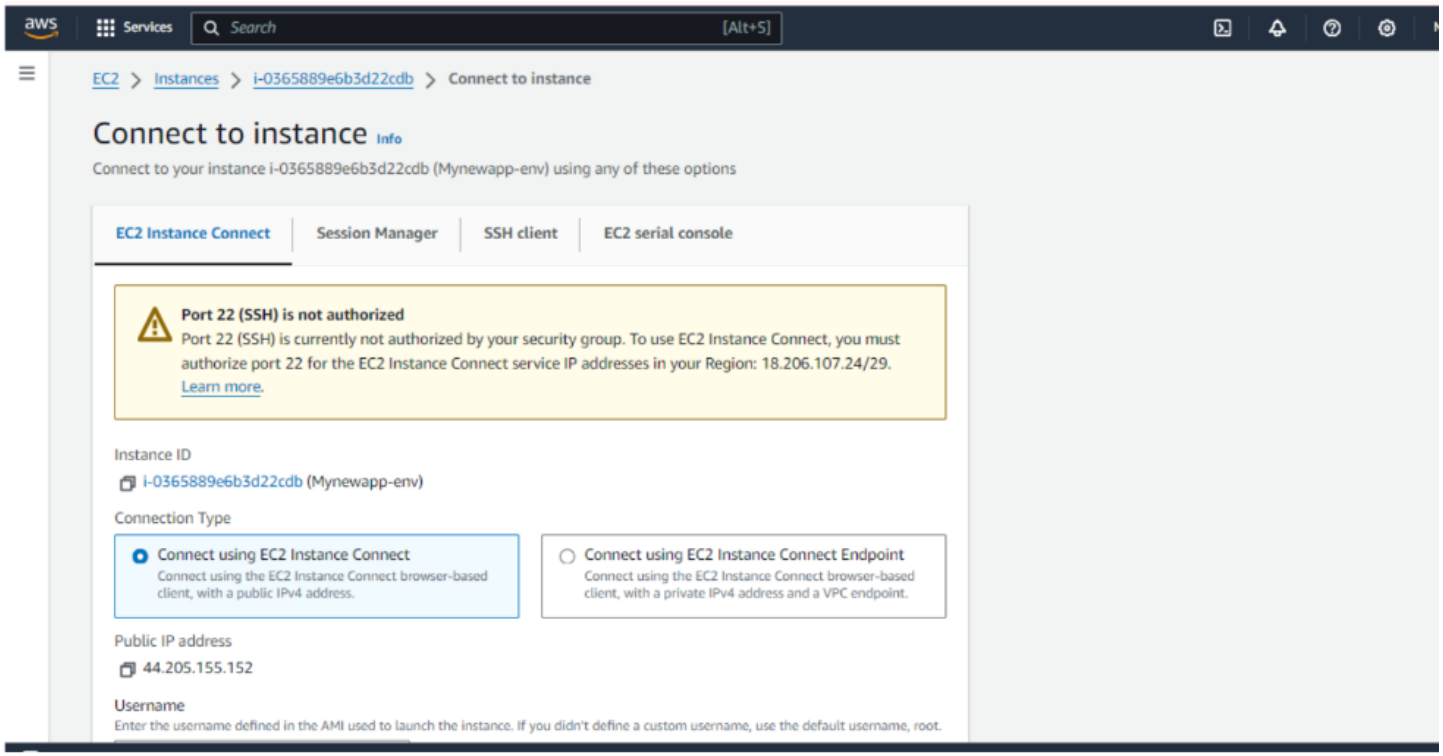
Advanced DevOps Experiment 4

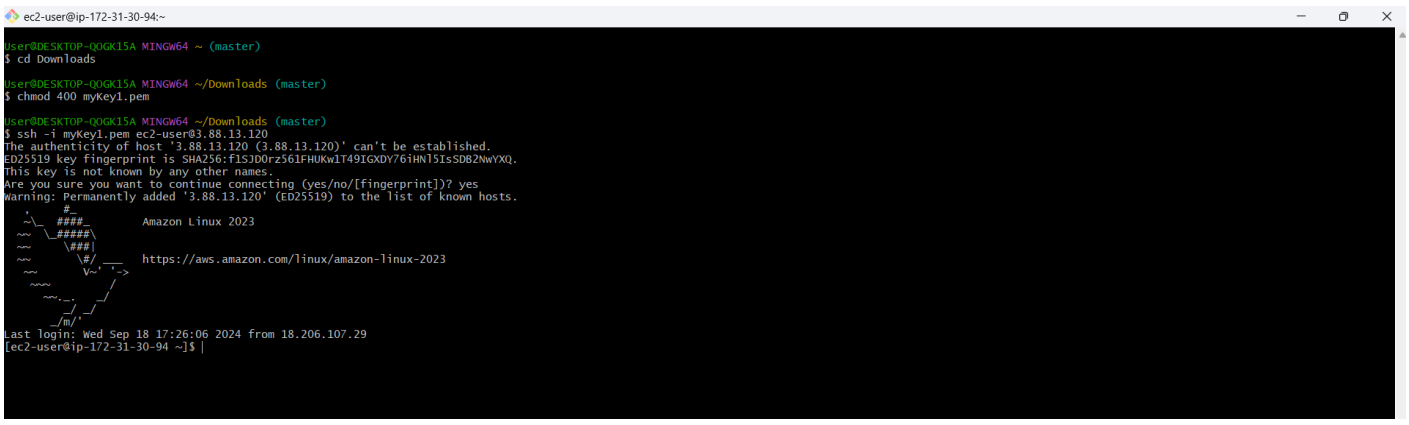
Aim: To install Kubectl and execute Kubectl commands to manage the Kubernetes cluster and deploy Your First Kubernetes Application.

Step 1: Go to AWS Academy in services select EC2 and create 3 instance with and name them as master, node1, node2 and remember to select instance type as t2.medium.



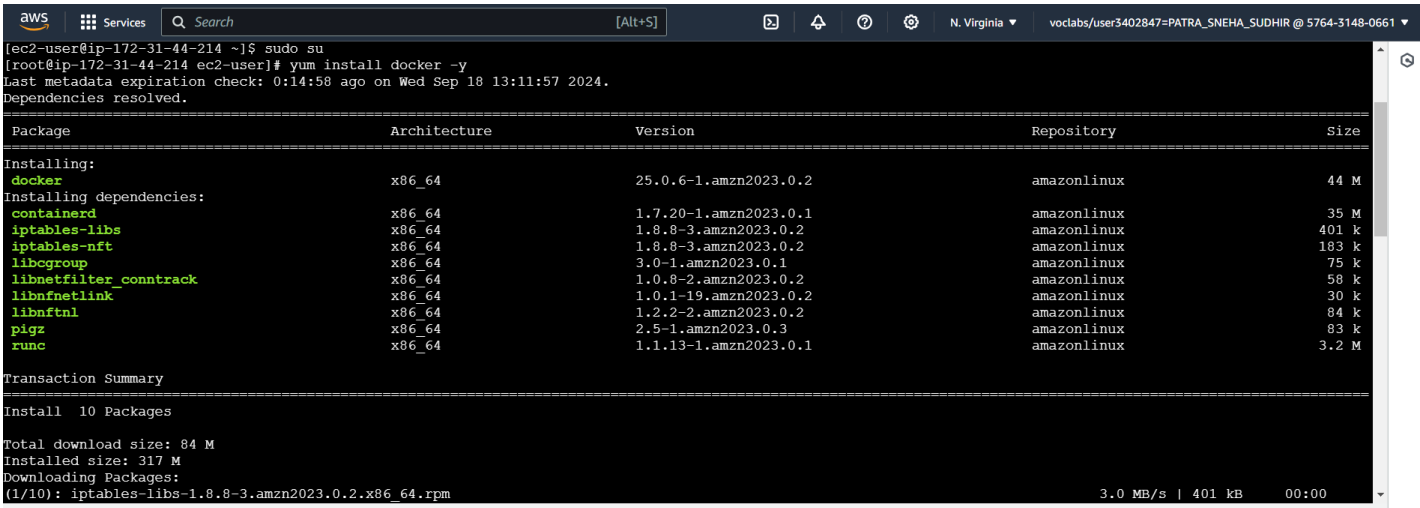


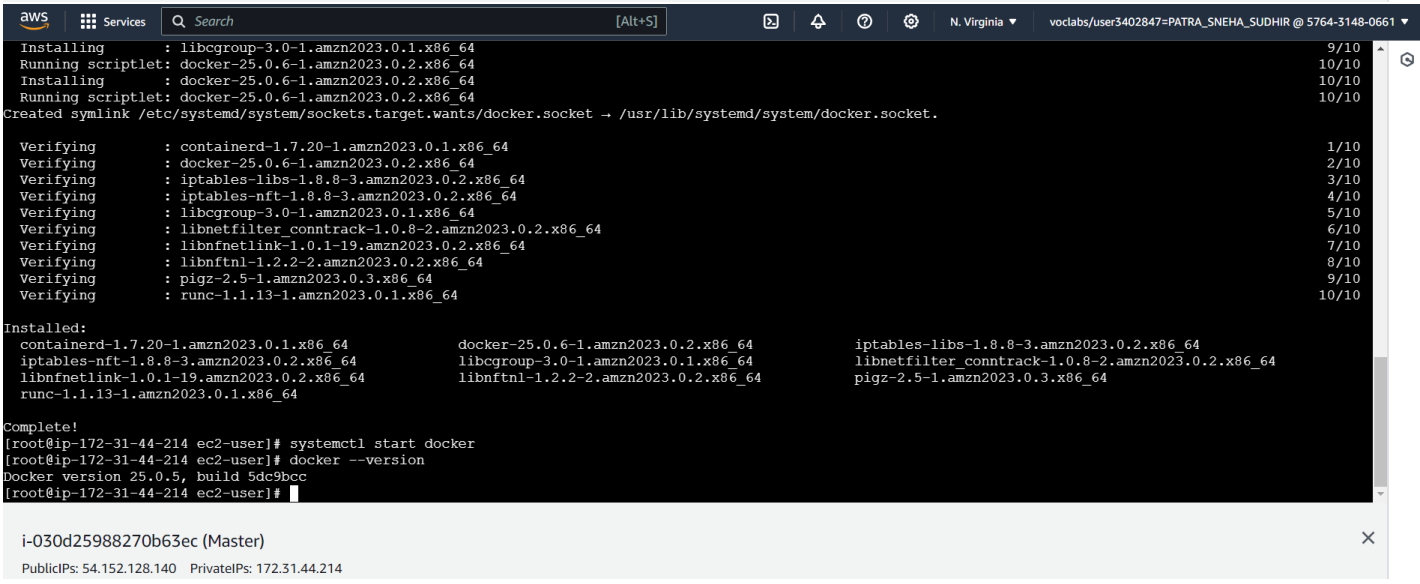
Step 2: Create a new key pair and name it as myKey1 and download as .pem file. Then, open command promt and go to the directory where the key is downloaded and run the following command
chmod 400 myKey1.pem
ssh -i myKey1.pem ec2-user@3.88.13.120
Repeat the steps for node1, master and node2



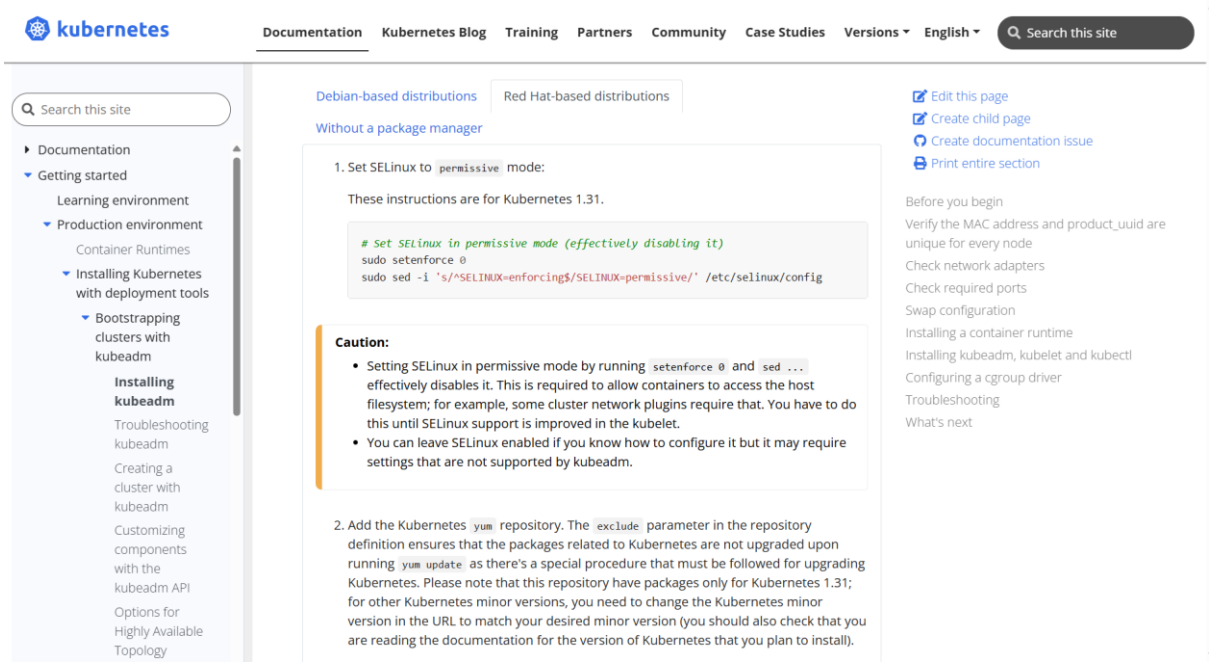
Step 3: Then, Select and connect each instance and run the following commands inside the console of each instance.

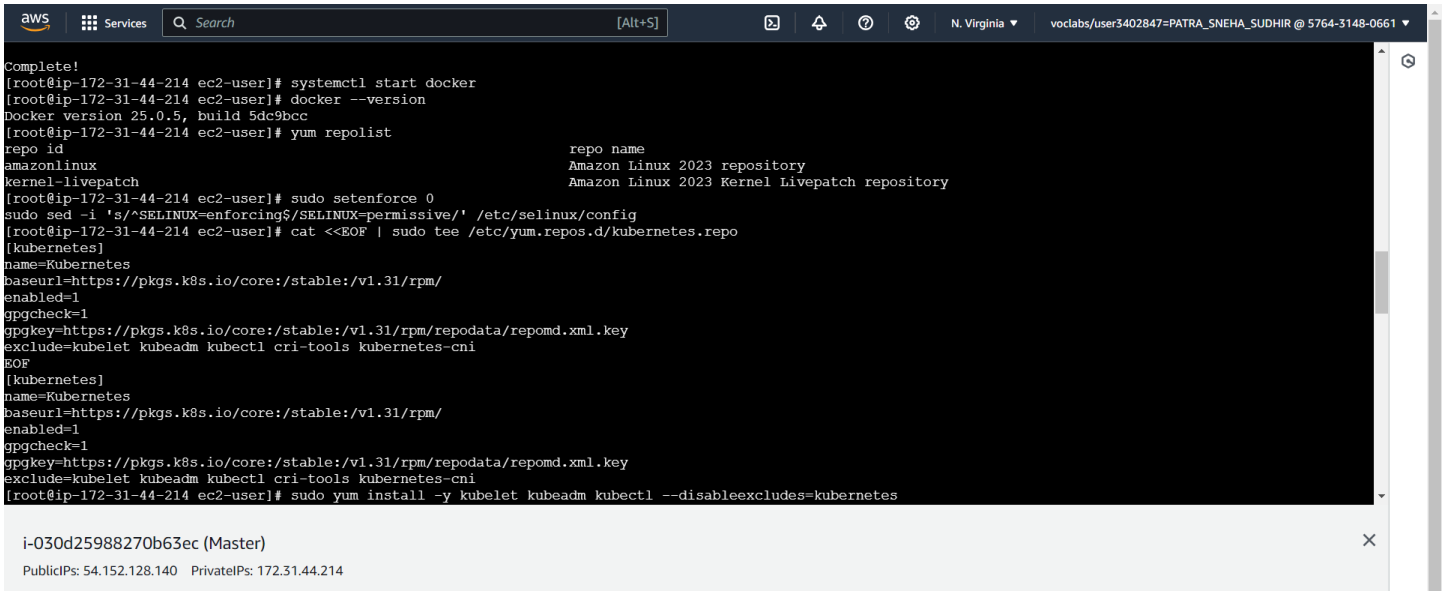
- sudo su
- yum install docker -y
- systemctl start docker
- docker –version
- yum repolist

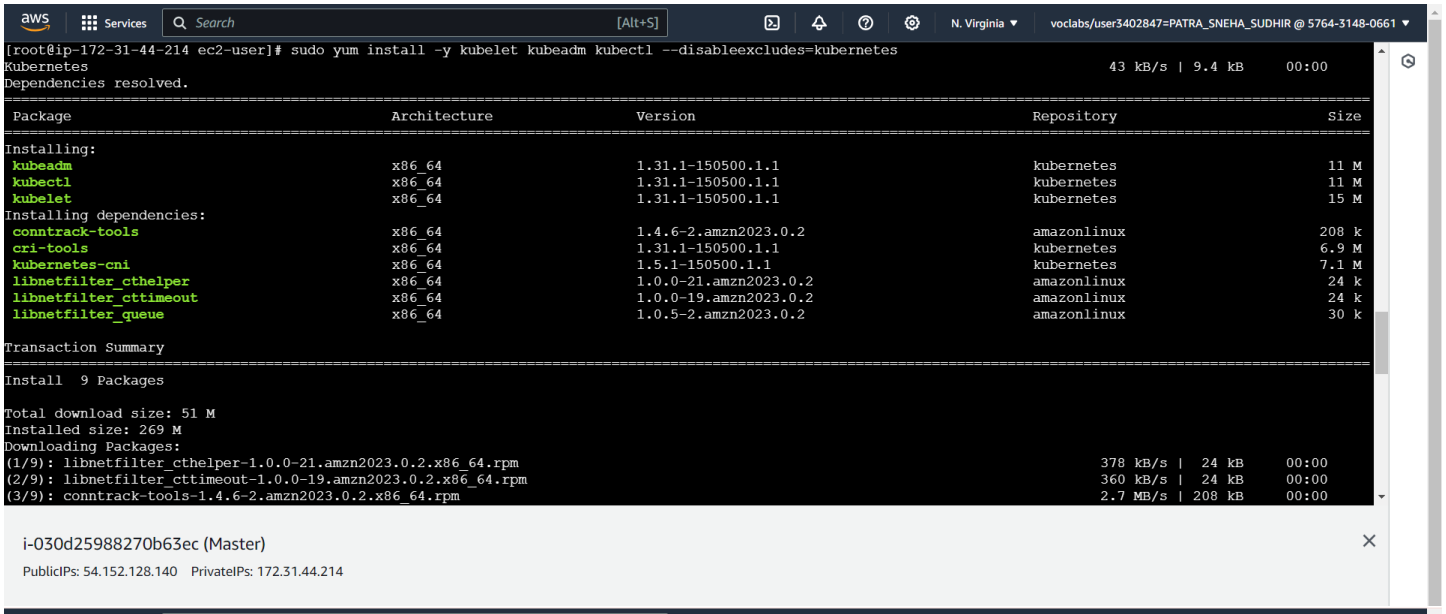


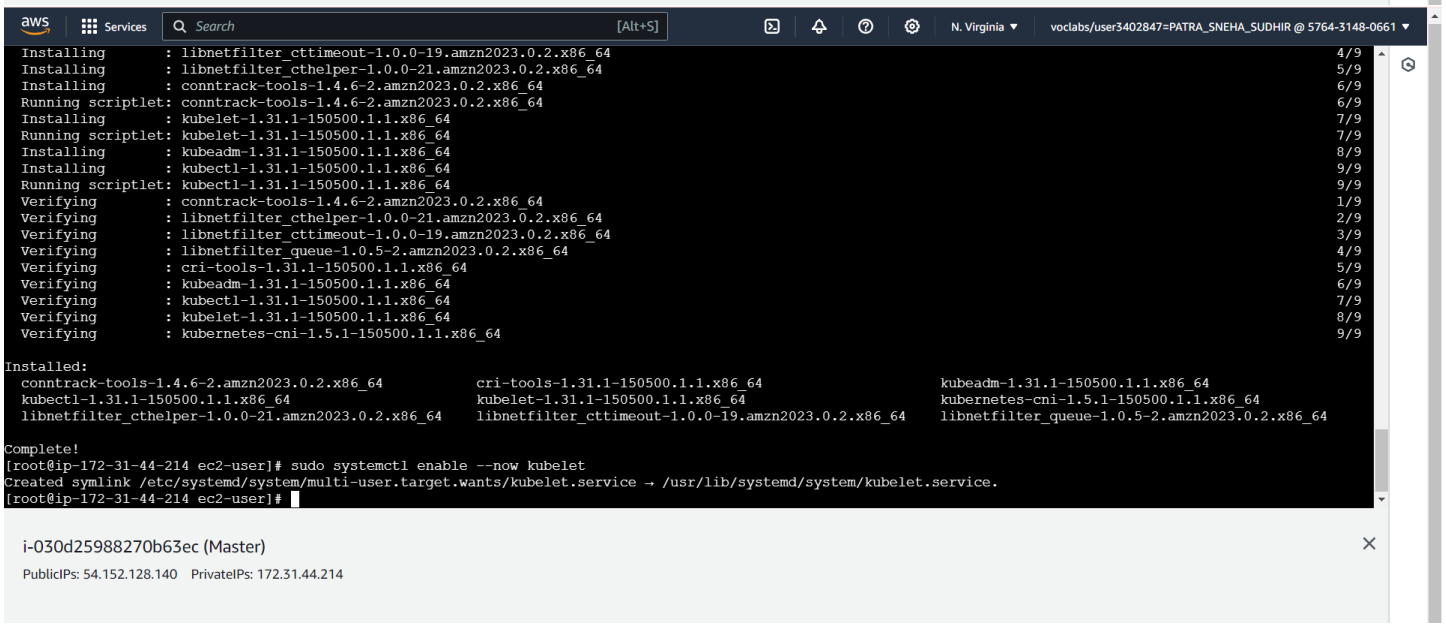


Step 4: Now, visit the following link https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/ and scroll down till you find Red-Hat and then select Red-Hat based distributions tab copy all the commands one by one in each console of instance.

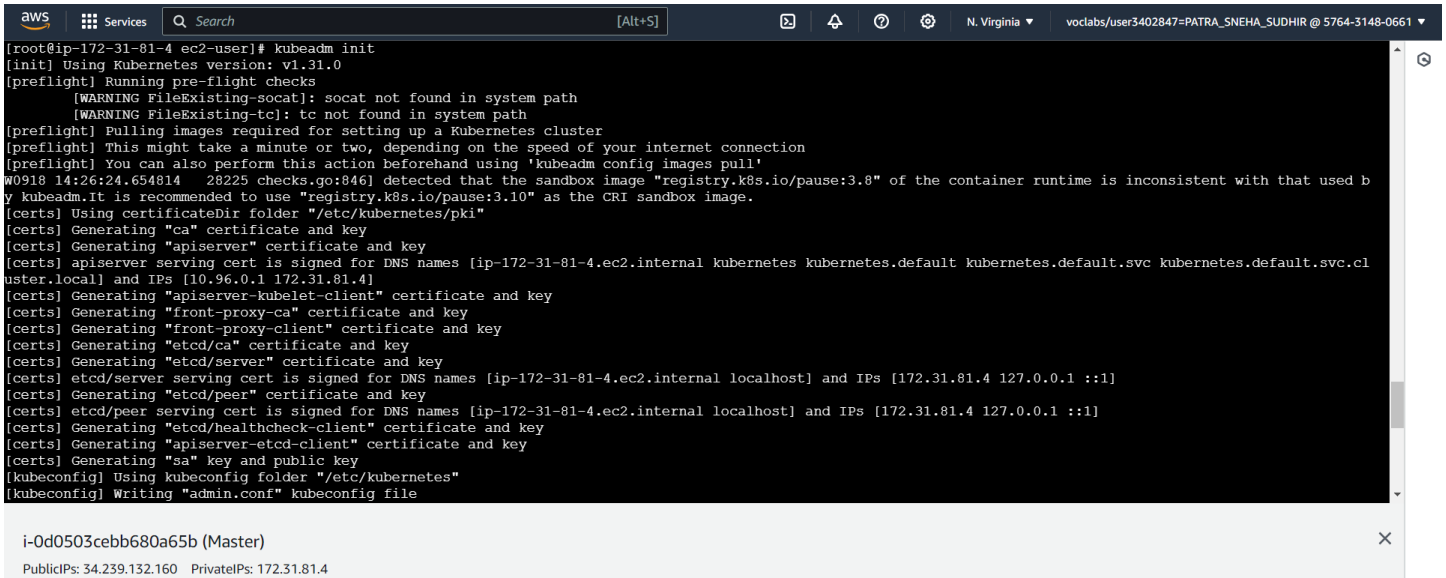






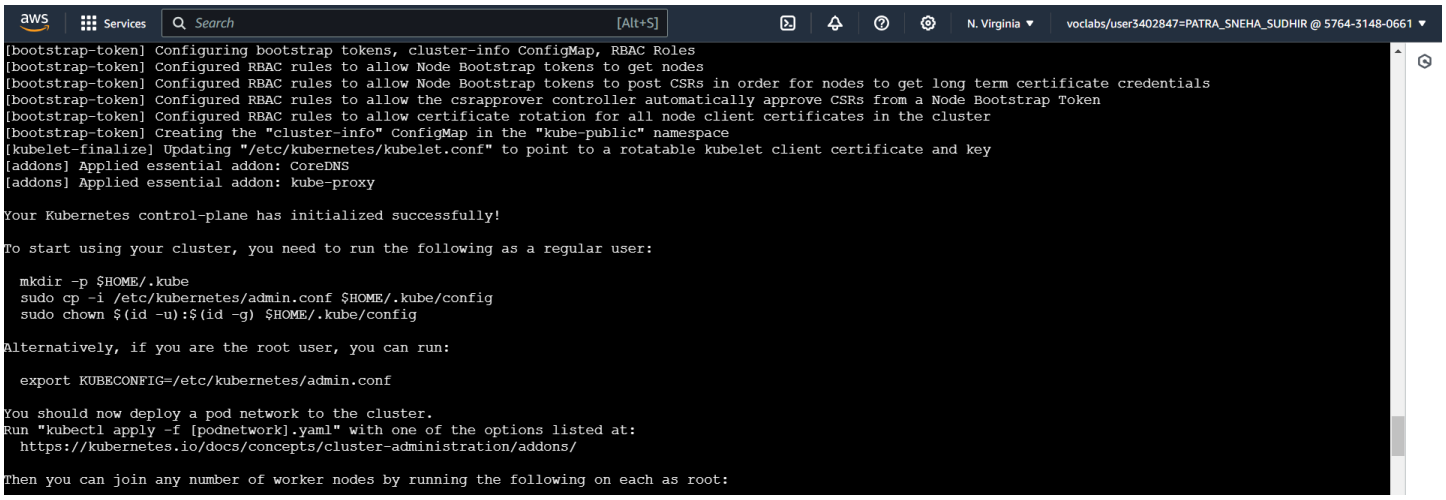


Step 5: Now, run the following command in the mater instance - kubeadm init

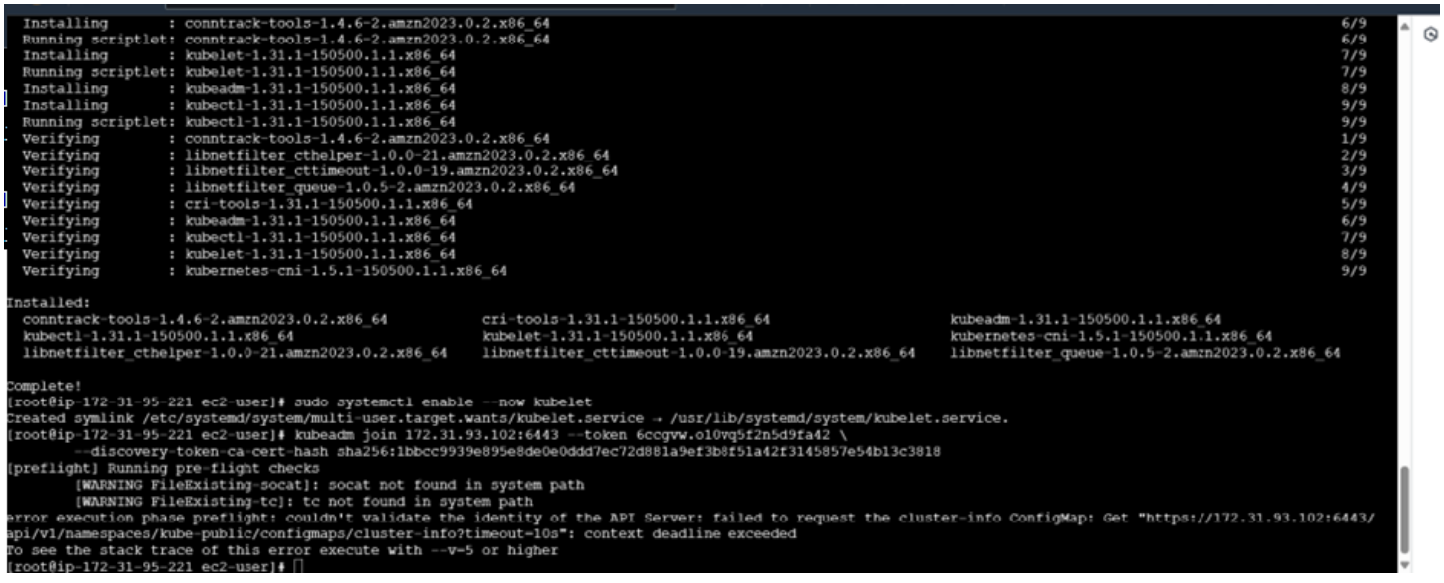


Step 6: Now, run the following commands in master instance's console –

a. mkdir -p $HOME/.kube
   sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
   sudo chown $(id -u):$(id -g) $HOME/.kube/config
b. export KUBECONFIG=/etc/kubernetes/admin.conf
c. kubeadm join 172.31.93.102:6443 --token 6ccgvw.o10vq5f2n5d9fa42 \
   --discovery-token-ca-cert-hash
   sha256:1bbcc9939e895e8de0e0ddd7ec72d881a9ef3b8f51a42f3145857e54b13c3818

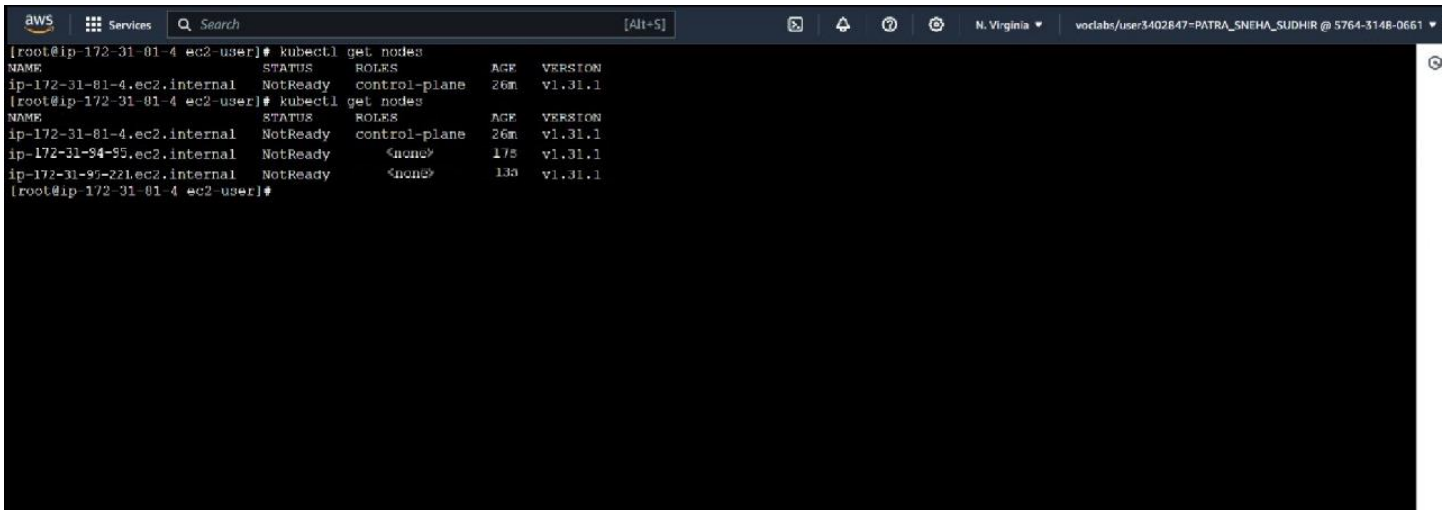


Step 7: Run this command in node1 and node2 -
kubeadm join 172.31.93.102:6443 --token 6ccgvw.o10vq5f2n5d9fa42 \
--discovery-token-ca-cert-hash sha256:1bbcc9939e895e8de0e0ddd7ec72d881a9ef3b8f51a42f3145857e54b13c3818

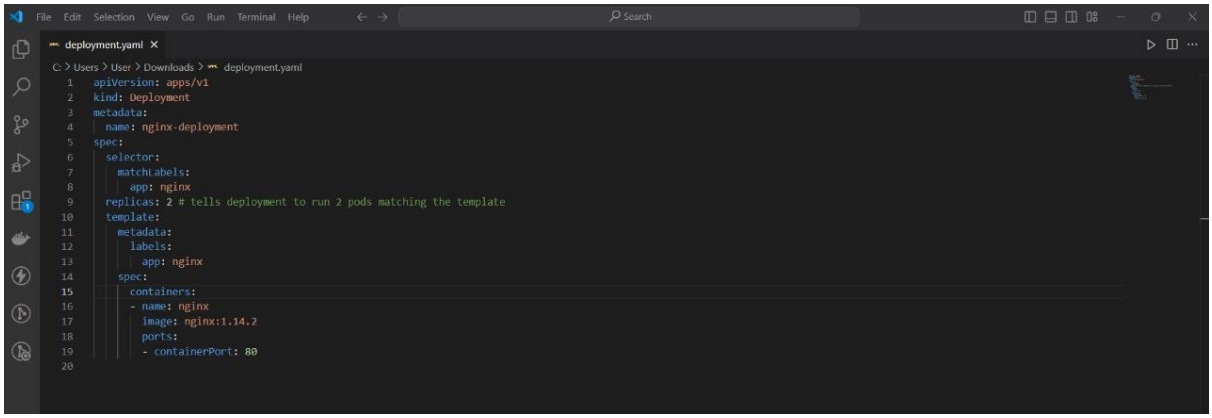


```
  Installing       : conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64                                                 6/9
  Running scriptlet: conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64                                                 6/9
  Installing       : kubelet-1.31.1-150500.1.1.x86_64                                                            7/9
  Running scriptlet: kubelet-1.31.1-150500.1.1.x86_64                                                            7/9
  Installing       : kubeadm-1.31.1-150500.1.1.x86_64                                                            8/9
  Installing       : kubectl-1.31.1-150500.1.1.x86_64                                                            9/9
  Running scriptlet: kubectl-1.31.1-150500.1.1.x86_64                                                            9/9
  Verifying        : conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64                                                 1/9
  Verifying        : libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64                                          2/9
  Verifying        : libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64                                         3/9
  Verifying        : libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64                                              4/9
  Verifying        : cri-tools-1.31.1-150500.1.1.x86_64                                                          5/9
  Verifying        : kubeadm-1.31.1-150500.1.1.x86_64                                                            6/9
  Verifying        : kubectl-1.31.1-150500.1.1.x86_64                                                            7/9
  Verifying        : kubelet-1.31.1-150500.1.1.x86_64                                                            8/9
  Verifying        : kubernetes-cni-1.5.1-150500.1.1.x86_64                                                      9/9

Installed:
  conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64          cri-tools-1.31.1-150500.1.1.x86_64          kubeadm-1.31.1-150500.1.1.x86_64
  kubectl-1.31.1-150500.1.1.x86_64                     kubelet-1.31.1-150500.1.1.x86_64            kubernetes-cni-1.5.1-150500.1.1.x86_64
  libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64   libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64   libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64

Complete!
[root@ip-172-31-94-95 ec2-user]# sudo systemctl enable --now kubelet
Created symlink /etc/systemd/system/multi-user.target.wants/kubelet.service → /usr/lib/systemd/system/kubelet.service.
[root@ip-172-31-94-95 ec2-user]# kubeadm join 172.31.93.102:6443 --token 6ccgvw.o10vq5f2n5d9fa42 \
        --discovery-token-ca-cert-hash sha256:1bbcc9939e095e8de0e0ddd7ec72d081a9ef3b8f51a42f3145857e54b13c3818
[preflight] Running pre-flight checks
        [WARNING FileExisting-socat]: socat not found in system path
        [WARNING FileExisting-tc]: tc not found in system path
error execution phase preflight: couldn't validate the identity of the API Server: failed to request the cluster-info ConfigMap: Get "https://172.31.93.102:6443/api/v1/namespaces/kube-public/configmaps/cluster-info?timeout=10s": context deadline exceeded
To see the stack trace of this error execute with --v=5 or higher
[root@ip-172-31-94-95 ec2-user]#
```

Step 8: Run the following command in master instance console -
kubectl get nodes



Step 9: Once the cluster is set up and running, deploy an Nginx application:
kubectl apply -f https://k8s.io/examples/application/deployment.yaml
Forward the Nginx service to your localhost so that you can access it using the following command
kubectl port-forward deployment/nginx-deployment 8080:80



Step 10: In a new terminal of Git Bash, run the following command:
curl --head http://127.0.0.1:8080

Step 11: The website is deployed.