Advance Devops Experiment 4

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

Theory: What is kubectl?

kubectl is the command-line tool for interacting with Kubernetes clusters. It allows you to manage Kubernetes resources by creating, updating, and deleting pods, deployments, services, and more.

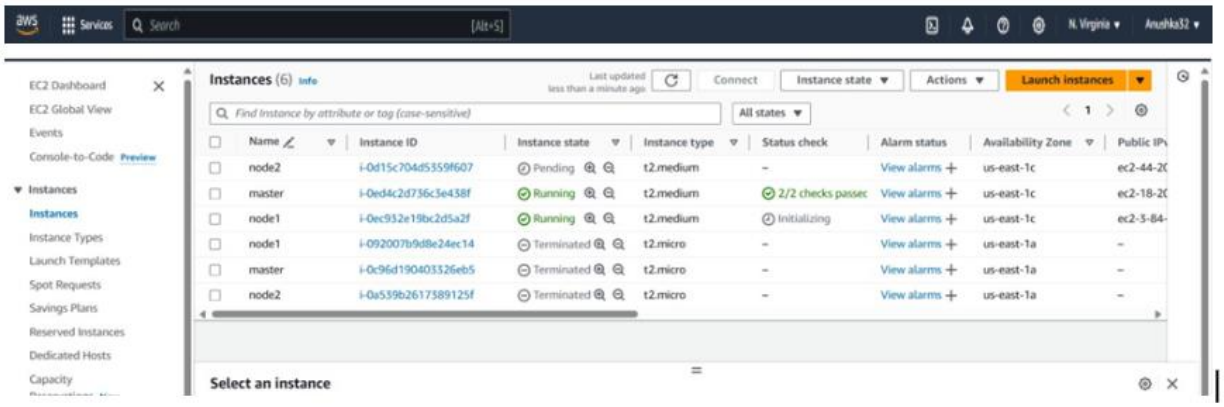
Prerequisites

A Kubernetes cluster running either locally (e.g., with Minikube, Kind, or Docker Desktop) or remotely (cloud-based, such as Google Kubernetes Engine (GKE), Amazon Elastic Kubernetes Service (EKS), or Azure Kubernetes Service (AKS)).

kubectl installed on your local machine to interact with the cluster.

Step 1:

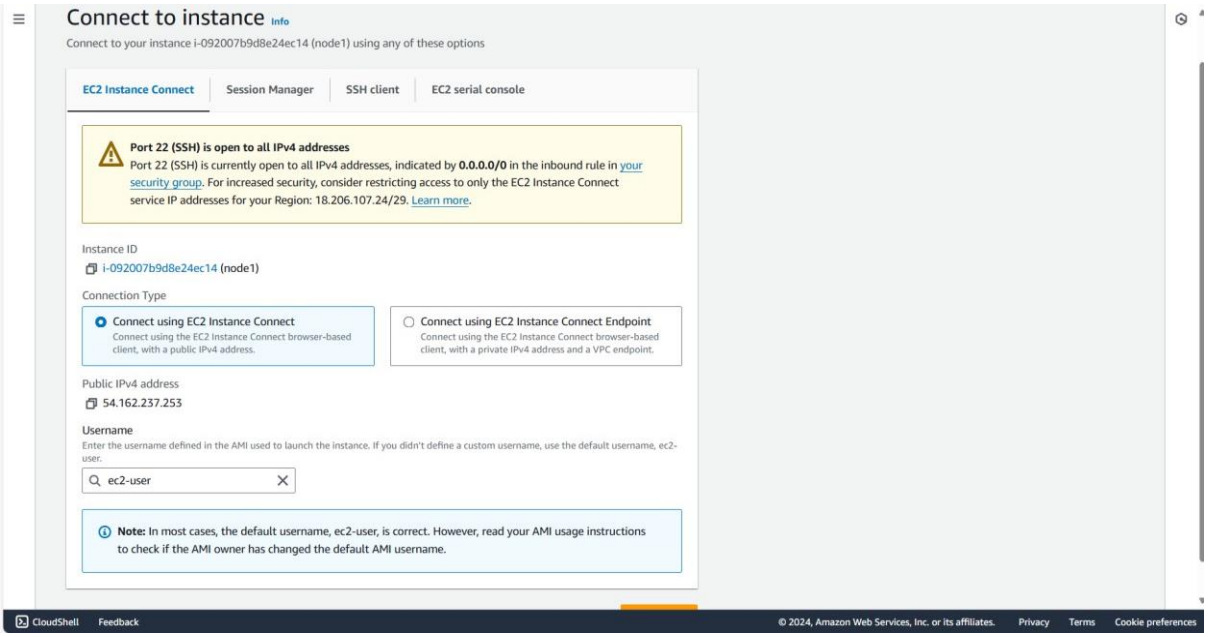
Go to AWS Academia in services select EC2 and create 3 instance with instance type t2.medium and names as node1, node2 and master



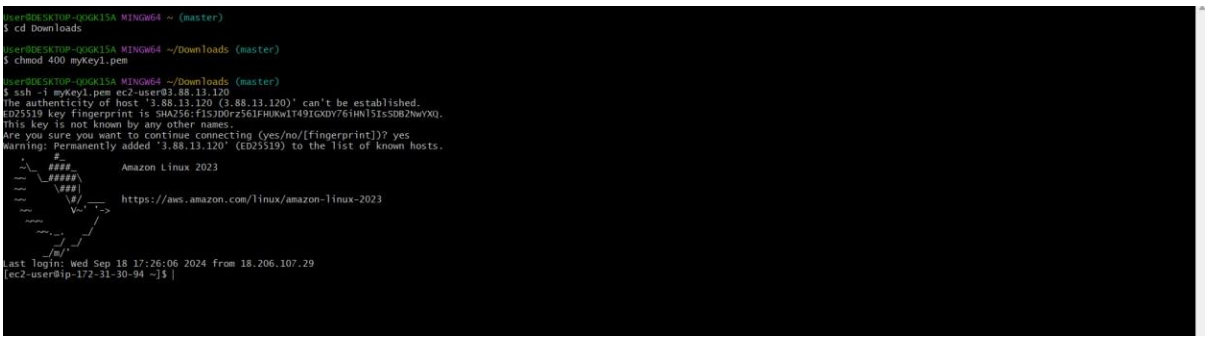
Step 2: Create a new key pair and name it as myKey1 and download as .pem file.

Open command promt run the following command

chmod 400 myKey1.pem



Anushka Shahane D15A 55



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

sudo su

yum install docker

-y systemctl start

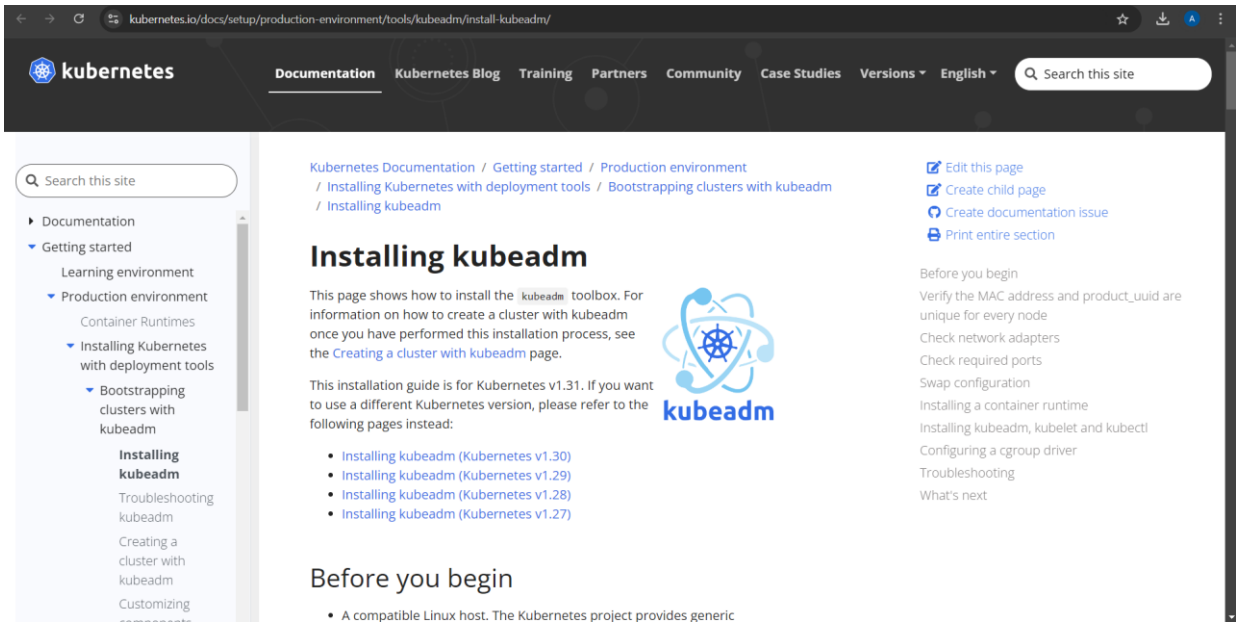
yum repolist

```
  Installing       : libnftnl-1.2.2-2.amzn2023.0.2.x86_64                                                    4/10
  Installing       : libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64                                               5/10
  Installing       : libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64                                      6/10
  Installing       : iptables-libs-1.8.8-3.amzn2023.0.2.x86_64                                               7/10
  Installing       : iptables-nft-1.8.8-3.amzn2023.0.2.x86_64                                                8/10
  Running scriptlet: iptables-nft-1.8.8-3.amzn2023.0.2.x86_64                                               8/10
  Installing       : libcgroup-3.0-1.amzn2023.0.1.x86_64                                                     9/10
  Running scriptlet: docker-25.0.6-1.amzn2023.0.2.x86_64                                                    10/10
  Installing       : docker-25.0.6-1.amzn2023.0.2.x86_64                                                     10/10
  Running scriptlet: docker-25.0.6-1.amzn2023.0.2.x86_64                                                    10/10
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.

  Verifying        : containerd-1.7.20-1.amzn2023.0.1.x86_64                                                 1/10
  Verifying        : docker-25.0.6-1.amzn2023.0.2.x86_64                                                     2/10
  Verifying        : iptables-libs-1.8.8-3.amzn2023.0.2.x86_64                                               3/10
  Verifying        : iptables-nft-1.8.8-3.amzn2023.0.2.x86_64                                                4/10
  Verifying        : libcgroup-3.0-1.amzn2023.0.1.x86_64                                                     5/10
  Verifying        : libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64                                      6/10
  Verifying        : libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64                                               7/10
  Verifying        : libnftnl-1.2.2-2.amzn2023.0.2.x86_64                                                    8/10
  Verifying        : pigz-2.5-1.amzn2023.0.3.x86_64                                                          9/10
  Verifying        : runc-1.1.13-1.amzn2023.0.1.x86_64                                                      10/10

Installed:
  containerd-1.7.20-1.amzn2023.0.1.x86_64          docker-25.0.6-1.amzn2023.0.2.x86_64          iptables-libs-1.8.8-3.amzn2023.0.2.x86_64
  iptables-nft-1.8.8-3.amzn2023.0.2.x86_64         libcgroup-3.0-1.amzn2023.0.1.x86_64          libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64
  libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64        libnftnl-1.2.2-2.amzn2023.0.2.x86_64         pigz-2.5-1.amzn2023.0.3.x86_64
  runc-1.1.13-1.amzn2023.0.1.x86_64

Complete!
[root@ip-172-31-33-243 ec2-user]# systemctl start docker
[root@ip-172-31-33-243 ec2-user]# docker --version
Docker version 25.0.5, build 5dc9bcc
[root@ip-172-31-33-243 ec2-user]#
```

i-0a539b2617389125f (node2)

PublicIPs: 107.21.35.198 PrivateIPs: 172.31.33.243

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



Anushka Shahane D15A 55

```
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                                    1/1
  Installing       : kubernetes-cni-1.5.1-150500.1.1.x86_64                                                             1/9
  Installing       : cri-tools-1.31.1-150500.1.1.x86_64                                                                 2/9
  Installing       : libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64                                                     3/9
  Installing       : libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64                                                4/9
  Installing       : libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64                                                 5/9
  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-172-31-33-243 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-172-31-33-243 ec2-user]#
```

i-0a539b2617389125f (node2)

PublicIPs: 107.21.35.198 PrivateIPs: 172.31.33.243

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

```
[root@ip-172-31-93-102 ec2-user]# kubeadm init
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
        [WARNING FileExisting-socat]: socat not found in system path
        [WARNING FileExisting-tc]: tc not found in system path
[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 beforehand using 'kubeadm config images pull'
W0918 14:21:55.805697   28020 checks.go:846] 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:3.10" as the CRI sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-93-102.ec2.internal kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.c
luster.local] and IPs [10.96.0.1 172.31.93.102]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [ip-172-31-93-102.ec2.internal localhost] and IPs [172.31.93.102 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-93-102.ec2.internal localhost] and IPs [172.31.93.102 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "super-admin.conf" kubeconfig file
[kubeconfig] Writing "kubelet.conf" kubeconfig file
[kubeconfig] Writing "controller-manager.conf" kubeconfig file
[kubeconfig] Writing "scheduler.conf" kubeconfig file
[etcd] Creating static Pod manifest for local etcd in "/etc/kubernetes/manifests"
[control-plane] Using manifest folder "/etc/kubernetes/manifests"
```

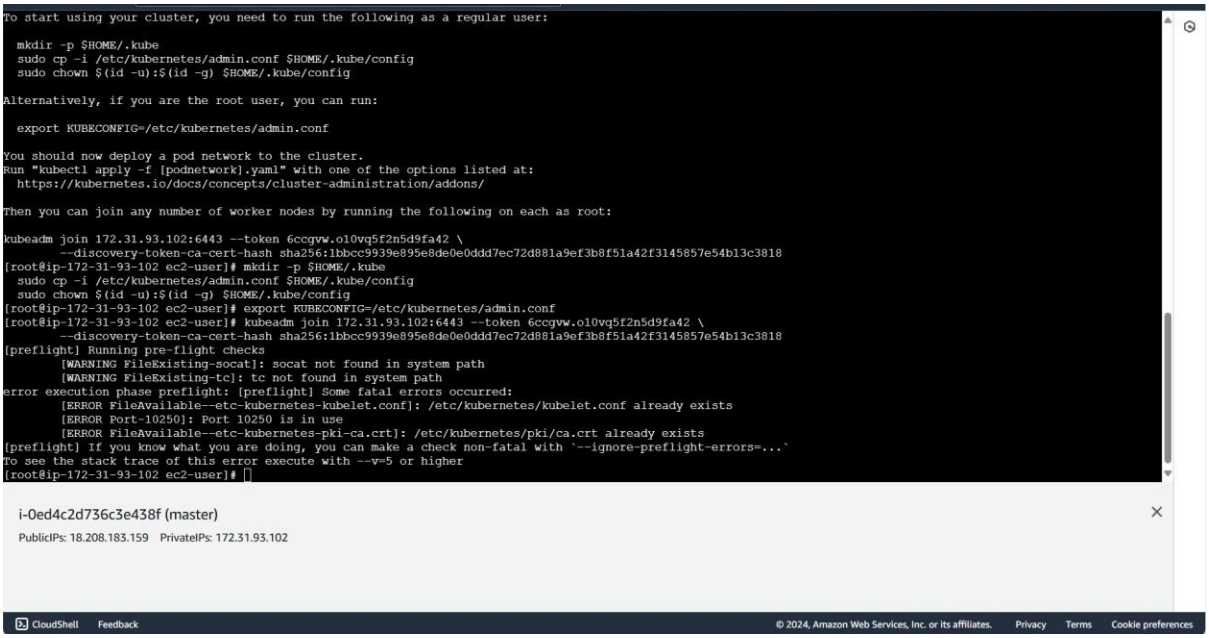
i-0ed4c2d736c3e438f (master)

PublicIPs: 18.208.183.159 PrivateIPs: 172.31.93.102

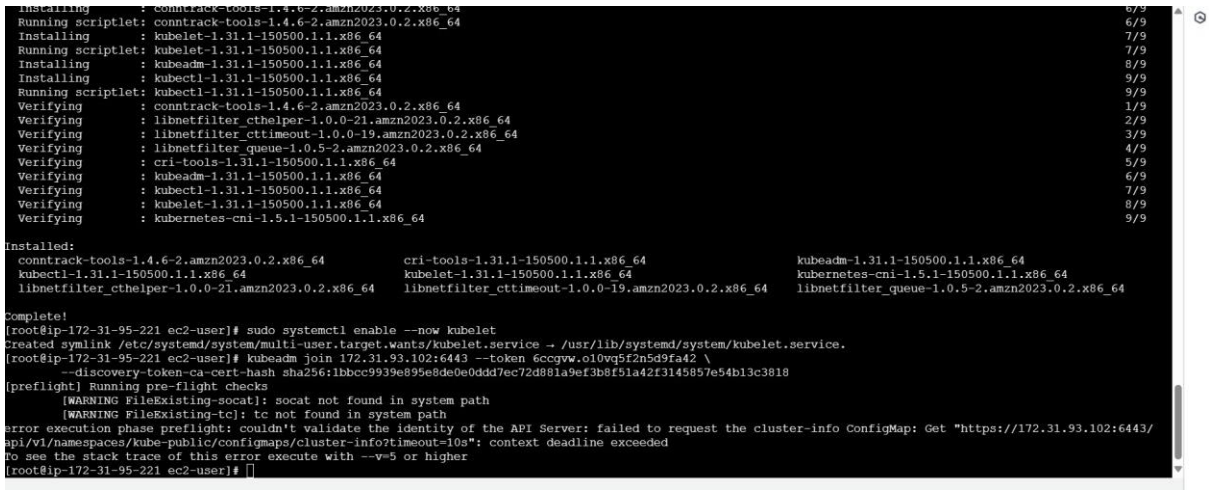
CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

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

mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf
$HOME/.kube/config sudo chown $(id -u):$(id -g)
$HOME/.kube/config
export KUBECONFIG=/etc/kubernetes/admin.conf
kubeadm join 172.31.93.102:6443 --token 6ccgvw.o10vq5f2n5d9fa42 \
--discovery-token-ca-cert-hash
sha256:1bbcc9939e895e8de0e0ddd7ec72d881a9ef3b8f51a42f3145857e54b13c3
818



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



```
  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 sha256:1bbcc9939e895e8de0e0ddd7ec72d881a9ef3b8f51a42f3145857e54b13c3818
[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]#
```

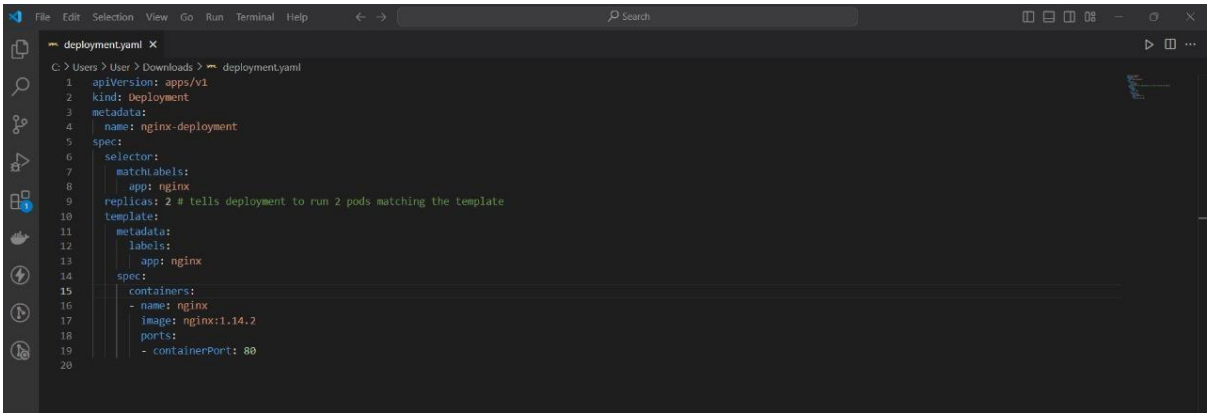
i-0ec932e19bc2d5a2f (node1)

PublicIPs: 3.84.157.220 PrivateIPs: 172.31.94.95

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

```
[root@ip-172-31-81-4 ec2-user]# kubectl get nodes
NAME                          STATUS     ROLES           AGE   VERSION
ip-172-31-81-4.ec2.internal   NotReady   control-plane   26m   v1.31.1
[root@ip-172-31-81-4 ec2-user]# kubectl get nodes
NAME                          STATUS     ROLES           AGE   VERSION
ip-172-31-81-4.ec2.internal   NotReady   control-plane   26m   v1.31.1
ip-172-31-94-95.ec2.internal  NotReady       <none>      17s   v1.31.1
ip-172-31-95-221ec2.internal  NotReady       <none>      13s   v1.31.1
[root@ip-172-31-81-4 ec2-user]#
```

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 terminal of Git Bash, run:

curl --head http://127.0.0.1:8080

The website is live after this

Anushka Shahane D15A 55

Anushka Shahane D15A 55