**EXPERIMENT NO. 3**

**NAME:** Arya Manoj Madhavi

**CLASS:** D15B

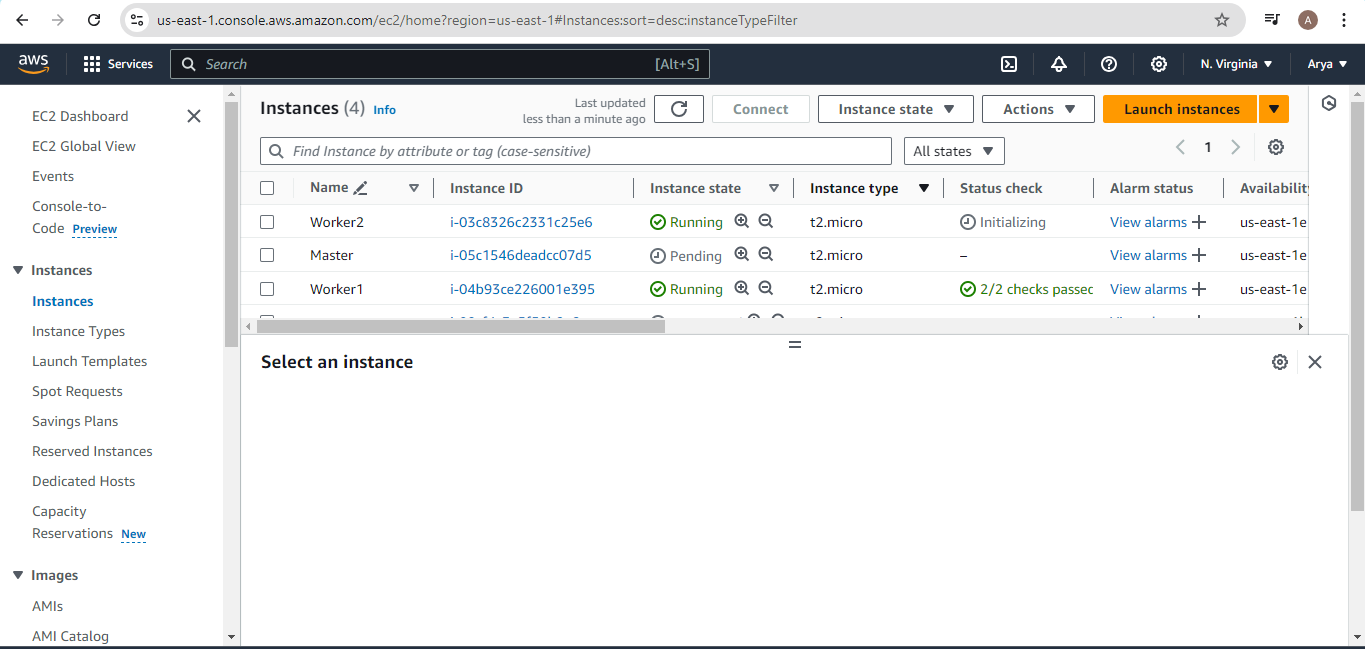
**ROLL NO.:** 31

**Aim: To understand the Kubernetes Cluster Architecture, install and Spin Up a Kubernetes**

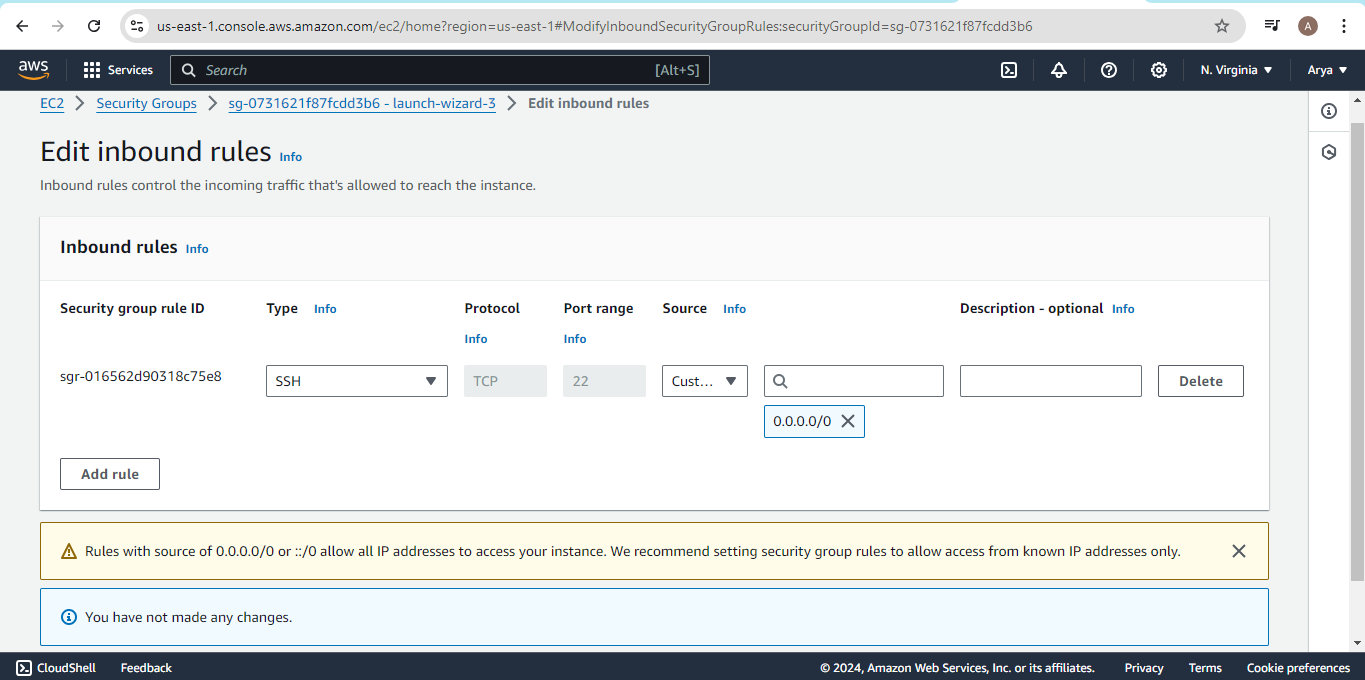
**Cluster on Linux Machines/Cloud Platforms.**

# **Steps:**

1.Create 3 EC2 Ubuntu Instances on AWS. (Name 1 as Master, the other 2 as puppet1,2)

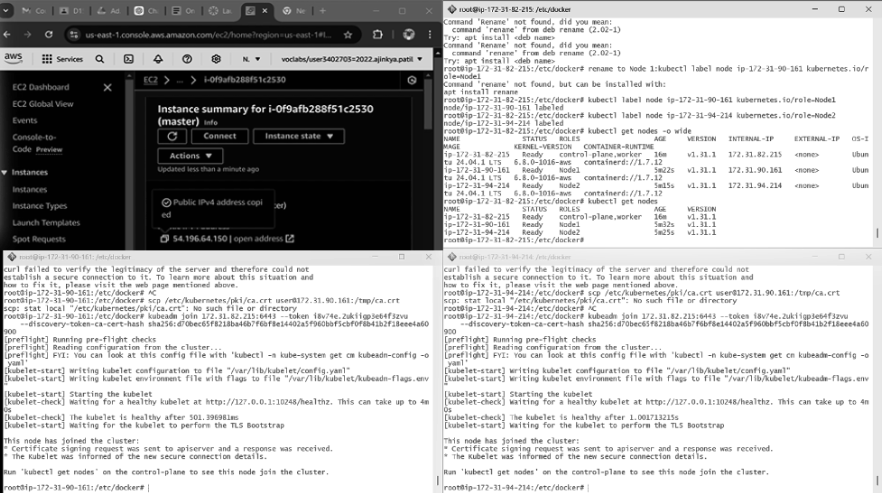


Edit the Security Group Inbound Rules to allow SSH



2.SSH into all 3 machines

# **ssh -i <keyname>.pem ubuntu@<public\_ip\_address>**



3.From now on, until mentioned, perform these steps on all 3 machines.

Install Docker

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable" sudo apt-get update

sudo apt-get install -y docker-ce

Then, configure cgroup in a daemon.json file.

cd /etc/docker

cat <<EOF | sudo tee /etc/docker/daemon.json

{

"exec-opts": ["native.cgroupdriver=systemd"], "log-driver": "json-file",

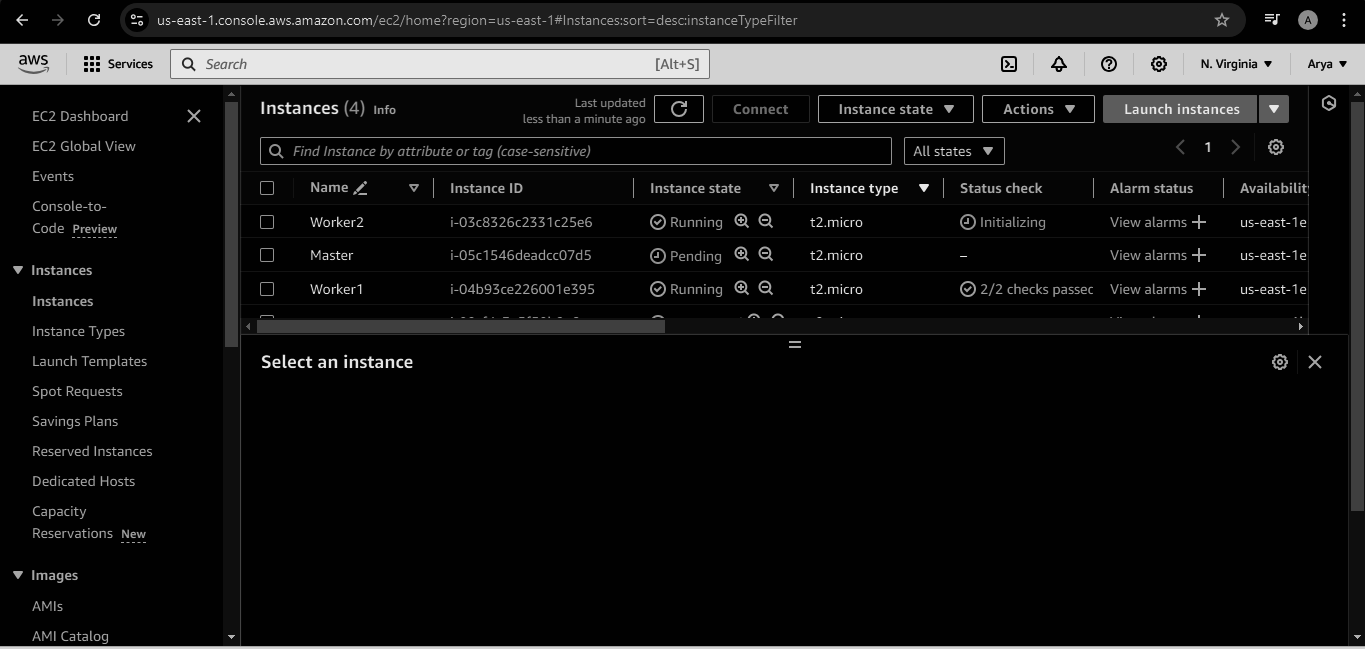
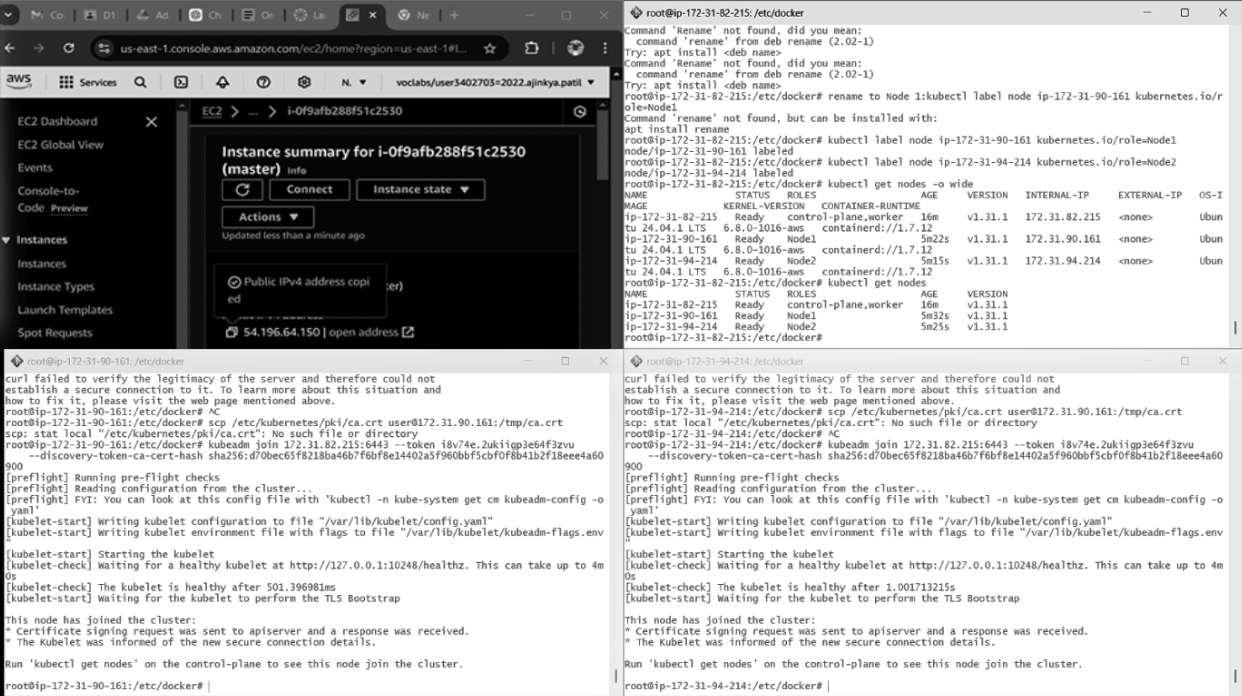
"log-opts": {

"max-size": "100m"

},

"storage-driver": "overlay2"

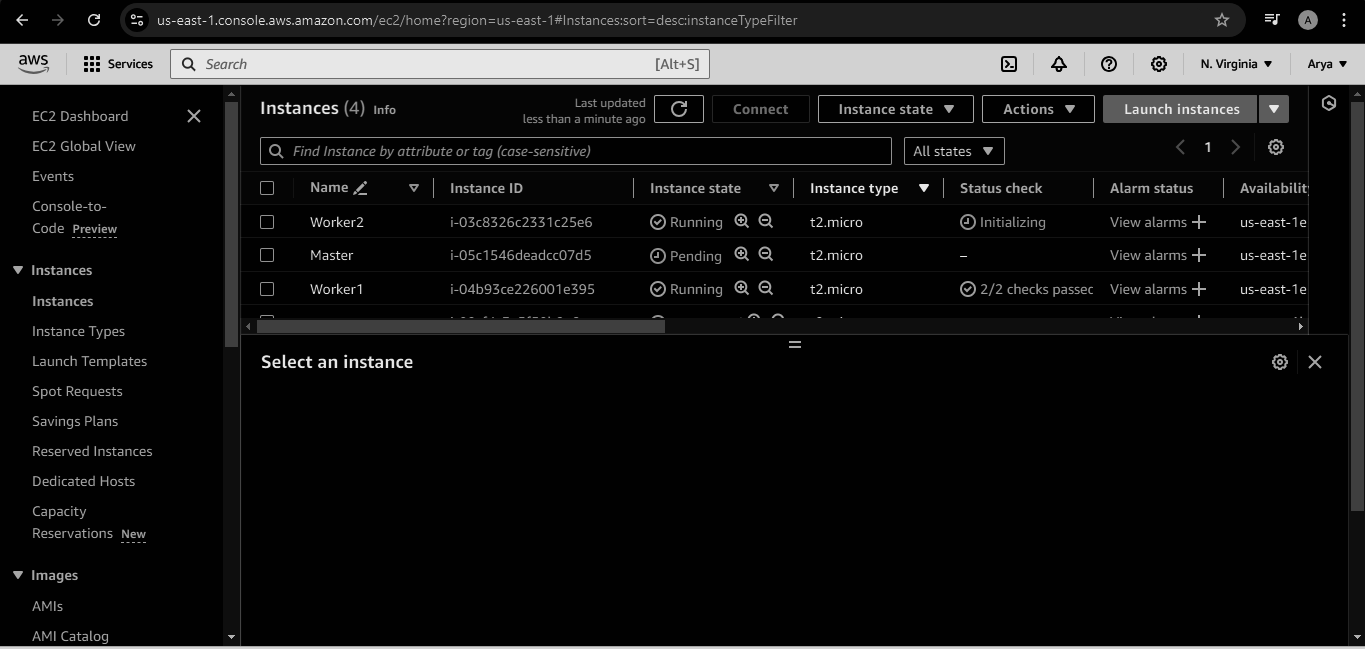
}

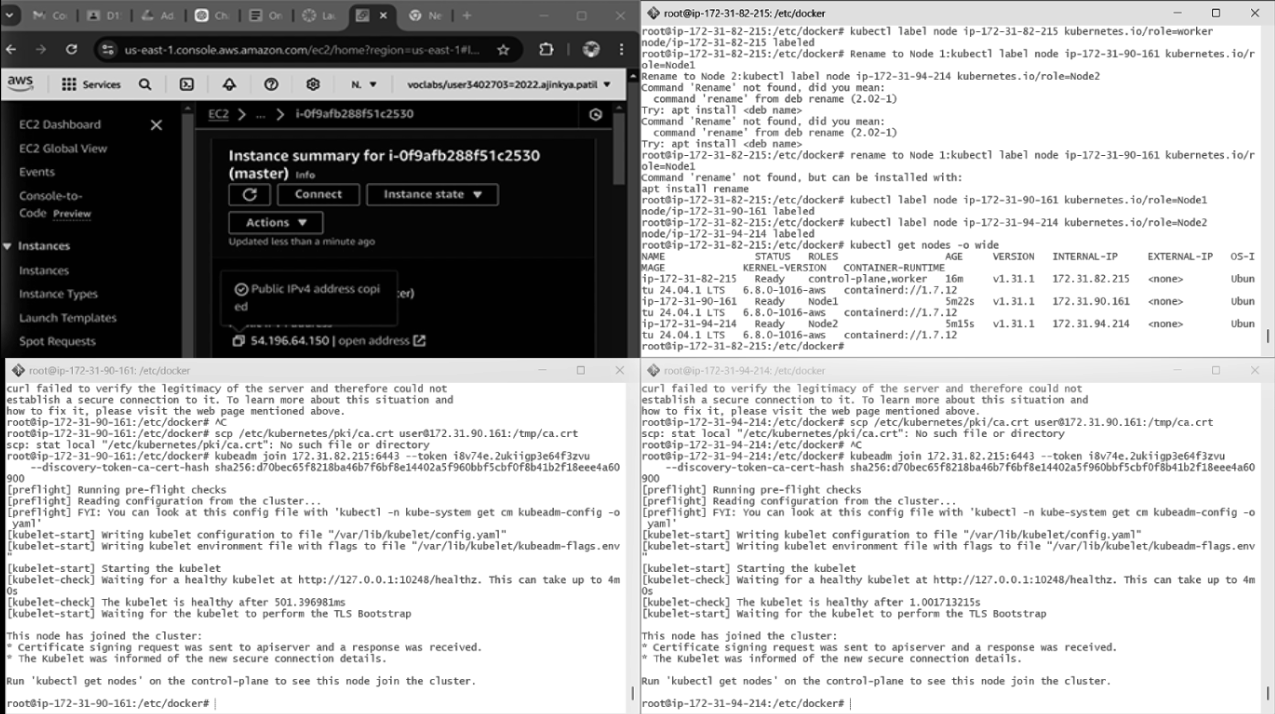


4.EOF

sudo systemctl enable docker sudo systemctl daemon-reload sudo systemctl restart docker

Install Kubernetes on all 3 machines





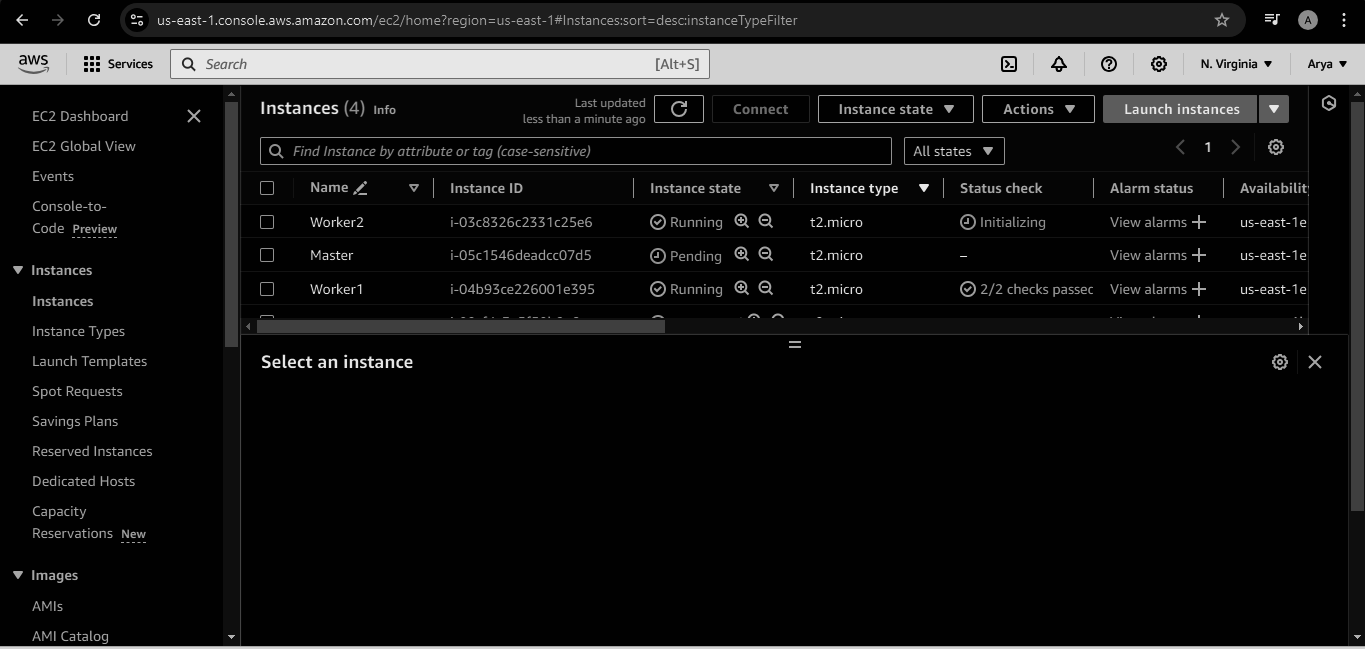
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -

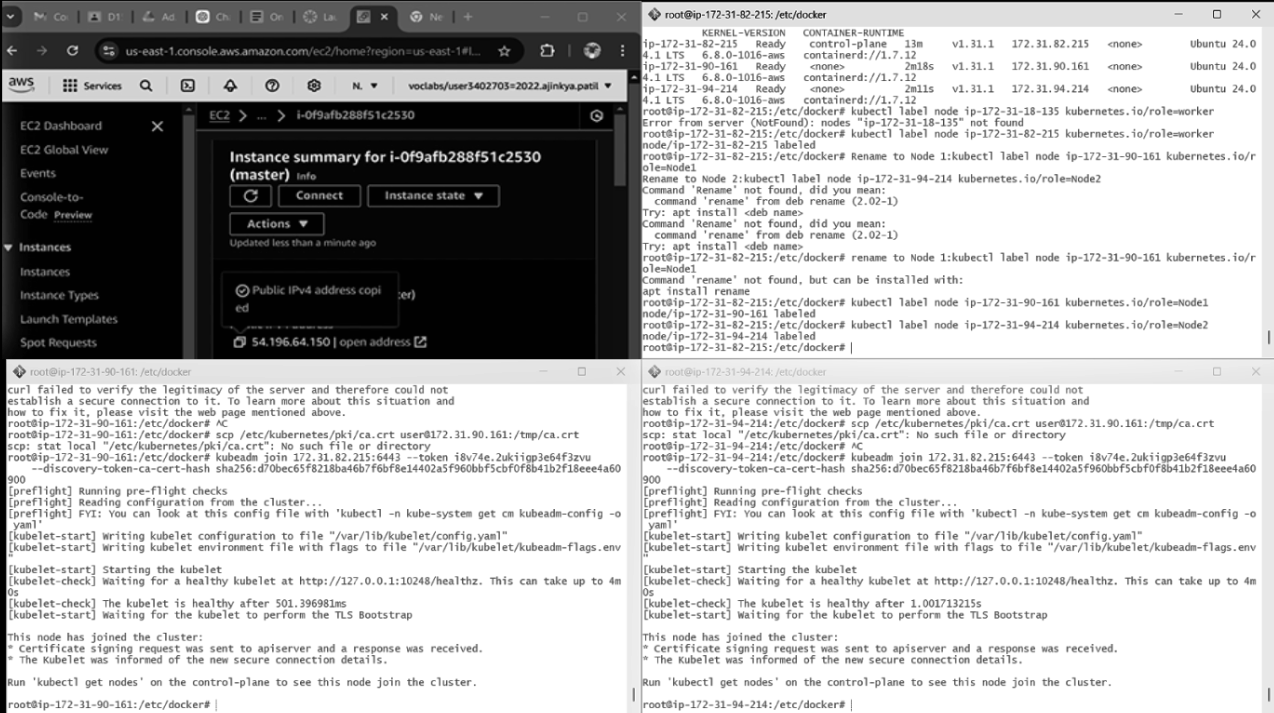
cat << EOF | sudo tee /etc/apt/sources.list.d/kubernetes.list deb https://apt.kubernetes.io/ kubernetes-xenial main

EOF

sudo apt-get update

sudo apt-get install -y kubelet kubeadm kubectl





5.After installing Kubernetes, we need to configure internet options to allow bridging.

sudo swapoff -a

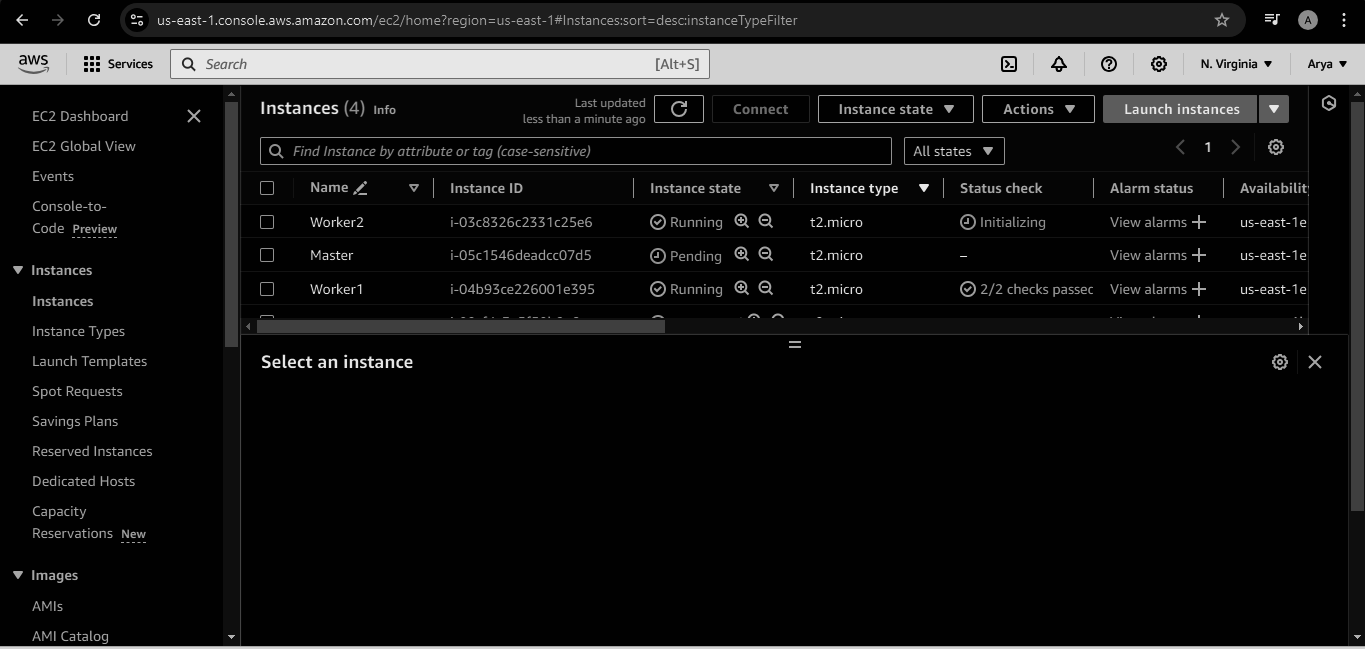
echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sysctl.conf

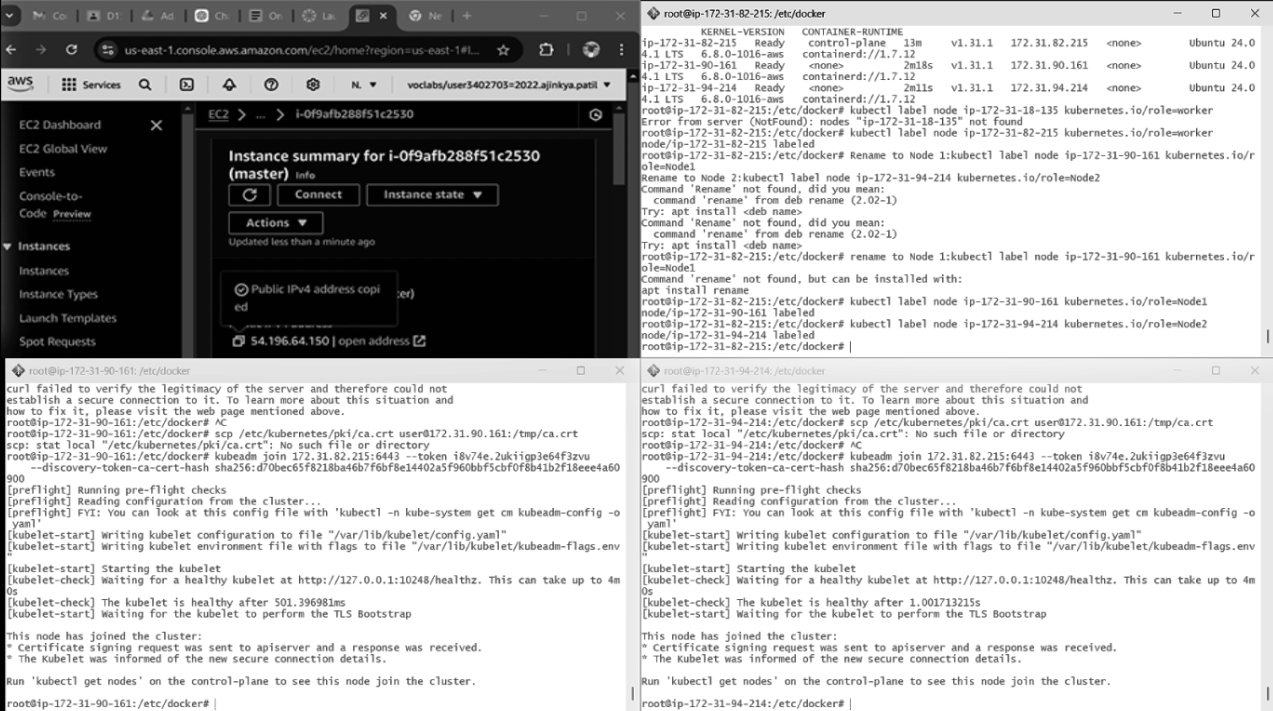
sudo sysctl -p

Perform this **ONLY on the Master machine**

Initialize the Kubecluster

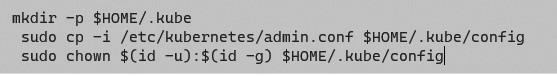
sudo kubeadm init --pod-network-cidr=10.244.0.0/16 **--ignore-preflight-errors=all**





6.Copy the join command and keep it in a notepad, we’ll need it later.

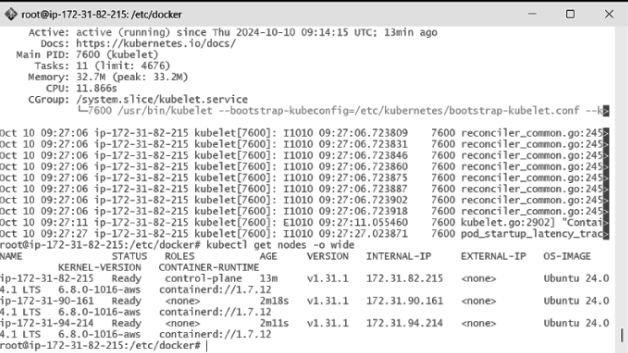
Copy the mkdir and chown commands from the top and execute them



7.Then, add a common networking plugin called flammel file as mentioned in the code.

kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml

9.activate running status



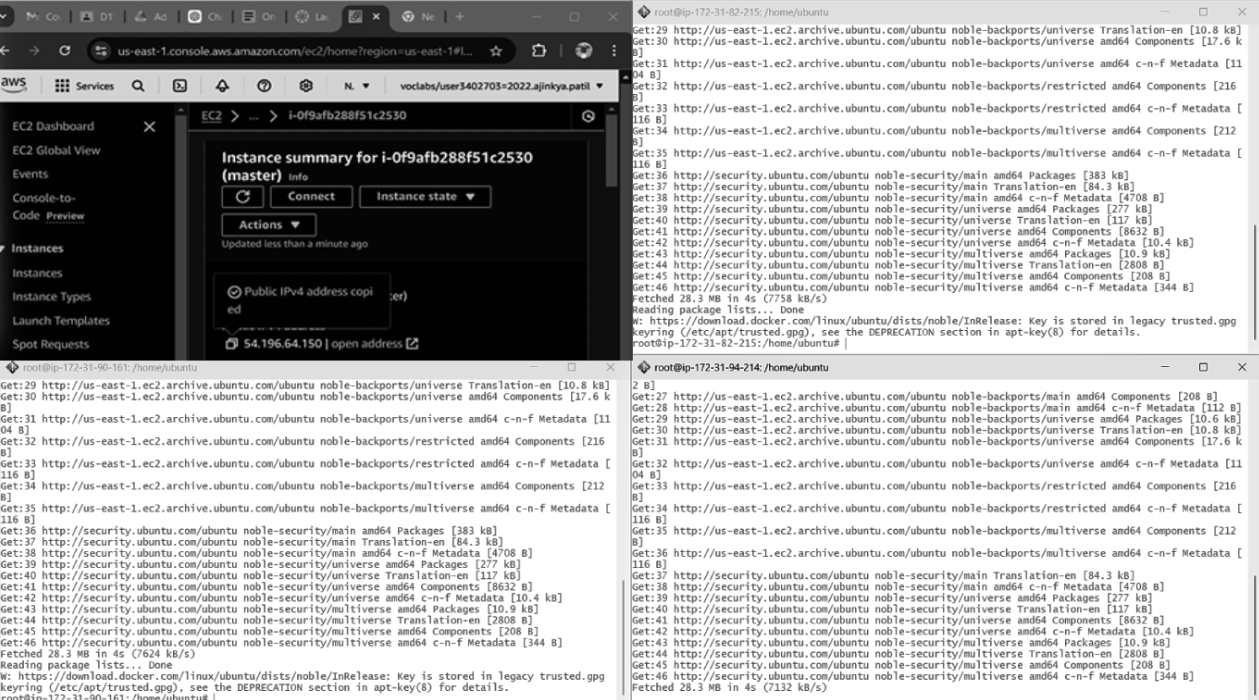
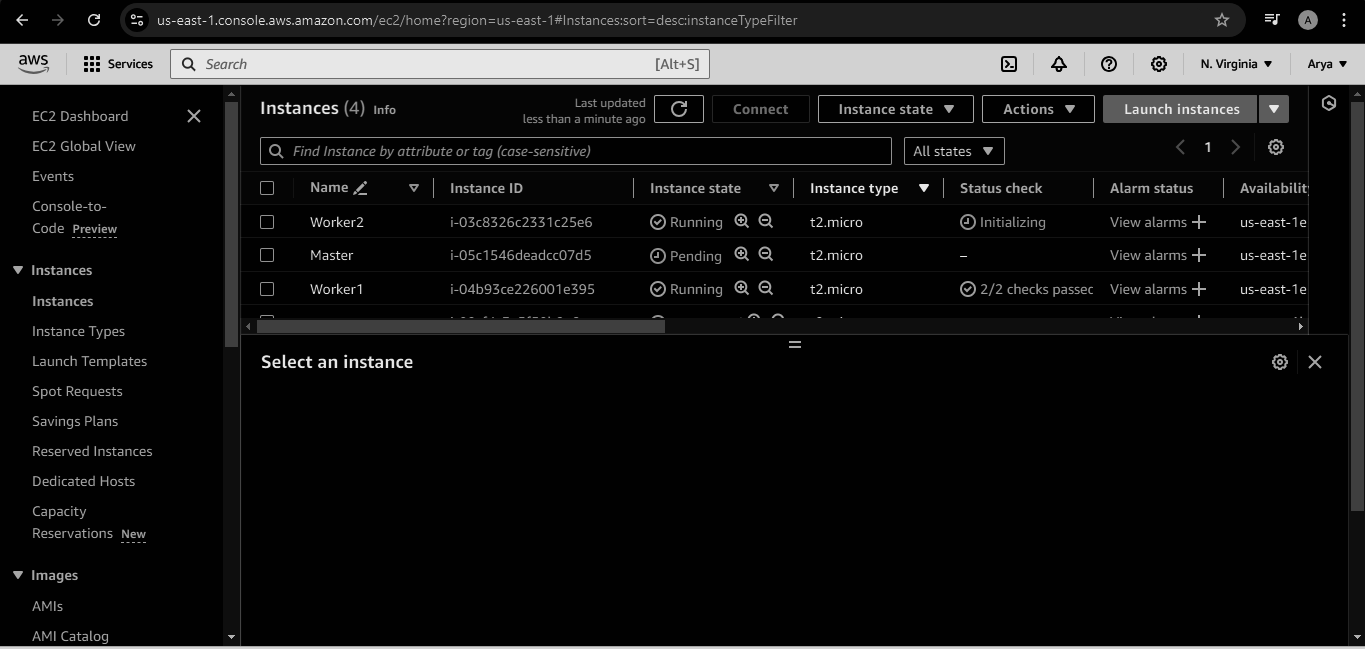
10.Check the created pod using this command

Now, keep a watch on all nodes using the following command

watch kubectl get nodes

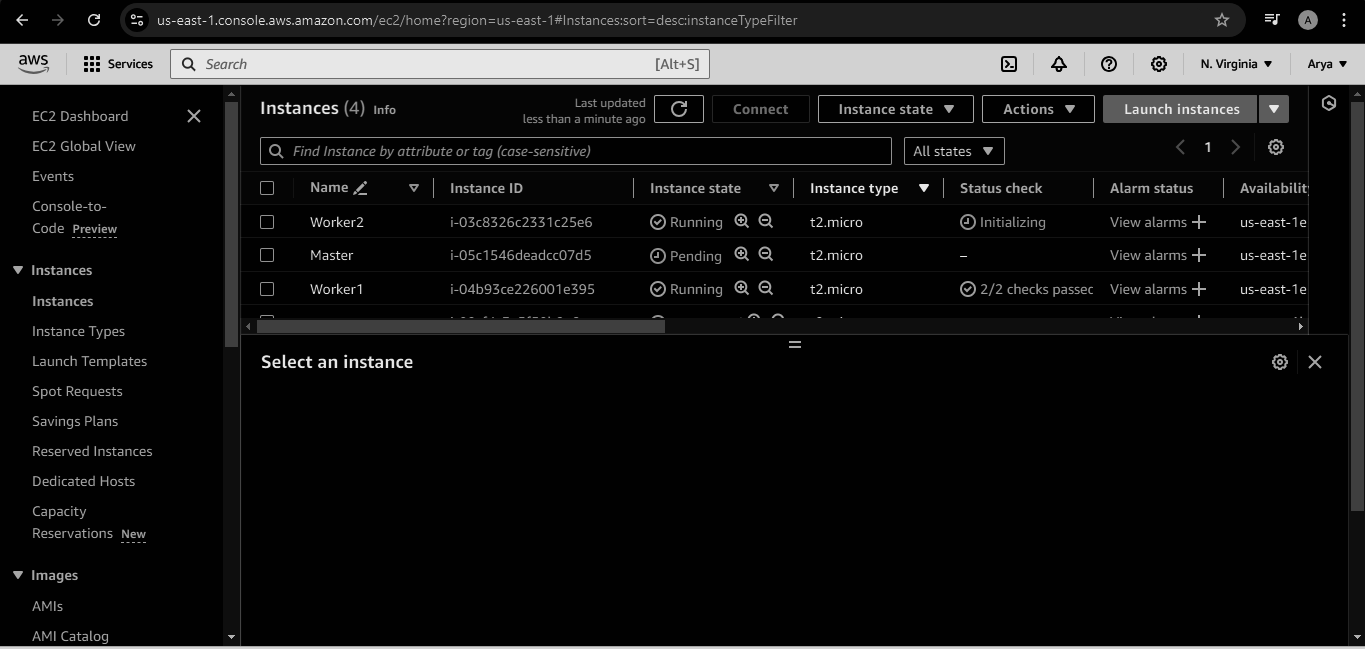
Perform this **ONLY on the worker machines**

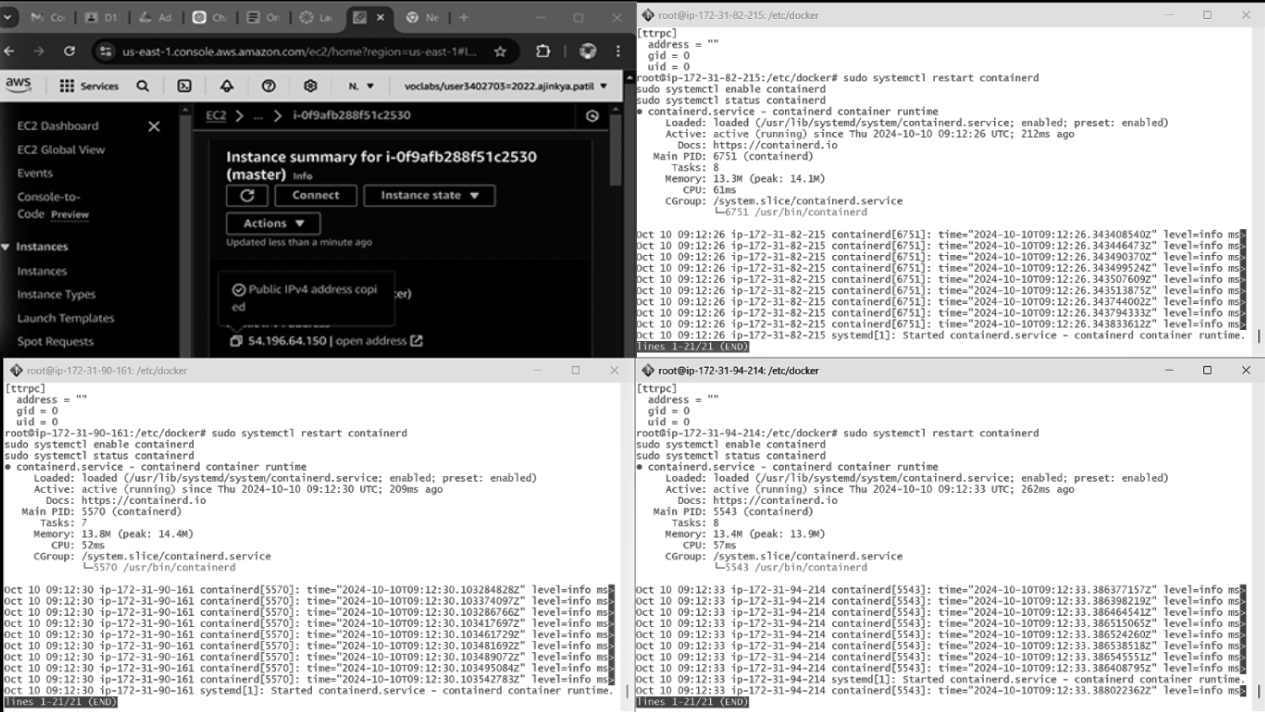
sudo kubeadm join <ip> --token <token> \



11.--discovery-token-ca-cert-hash <hash>

Now, notice the changes on the master terminal





That’s it, we now have a Kubernetes cluster running across 3 AWS EC2 Instances. This cluster can be used to further deploy applications and their loads being distributed across these machines.