**KOPS ACCESS THE SERVER**

**1.** sudo su -

**2.** docker version

**3.** snap info docker

**4.** apt update -y

**5.** docker version

**6.** apt install docker.io

**7**. aws configure

**8.** snap info aws-cli

**9.** snap install aws-cli --channel=v1/stable --classic

**VERIFY THE DOCKER**

**10.** systemctl daemon-reload

**11.** systemctl start docker

**12.** systemctl status docker

**NOW AWS CONFIGURE**

**13.** aws configure

accskey: AKIA3LET5ZMN7MZCJXBO

secretkey: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

region name:ap-northeast-3

output format: table

**KUBERNETES INSTALL kubectl OF UBUNTU**

**14.** curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"

**NOW PROVIDE THE EXECTION PERMISIONS**

**15.** chmod +x kubectl

ll

**KUBERNETES INSTALL KOPS**

**GO to https://kops.sigs.k8s.io/getting\_started/install/**

**16.** curl -Lo kops https://github.com/kubernetes/kops/releases/download/$(curl -s https://api.github.com/repos/kubernetes/kops/releases/latest | grep tag\_name | cut -d '"' -f 4)/kops-linux-amd64

chmod +x kops

ll

**17.** vi .bashrc

**18.** source .bashrc

**NOW MOVE TO KUBECTLS AND KOPS**

**19.** mv kops /usr/local/bin/kops

**20.** mv kubectl /usr/local/bin/kubectl

ll

**21.** kubectl version --client --output=yaml

**22.** kops version

**23.** aws s3api create-bucket \

--bucket ravi12.local \

--region us-east-1

**24.** aws s3api put-bucket-versioning --bucket ravi12.local --region us-east-1 --versioning-configuration Status=Enabled

**25.** export KOPS\_STATE\_STORE=s3://ravi12.local

**26**. kops create cluster --name kopsclus.k8.local --zones us-east-1b --state=s3://ravi12.local --control-plane-size t2.medium --node-size t2.micro --cloud aws

**27.** kops update cluster --name kopsclus.k8.local --yes --admin

**CLUSTER INFORMATION**

**28.** kops get cluster

**LIST THE NODES**

**29.** kubectl get nodes --show-labels **(UNTILL YOU WILL GIVE YOU CAME TO THE READY STATE)**

**CREATION OF POD:**

vi pod.yml

apiVersion: v1

kind: Pod

metadata:

name: pod1

labels:

app: zomato

spec:

containers:

- name: cont1

image: httpd

ports:

- containerPort: 80

**30.** kubectl create -f pod.yml

**31.** kubectl get pods

**32.** kubectl get pod -o wide

**CLUSTER IP:**

**33.** vi service.yml

apiVersion: v1

kind: Service

metadata:

name: my-service

spec:

type: ClusterIP

selector:

app: zomato

ports:

- port: 80

targetPort: 80

-->kubectl create -f service.yml

kubectl get svc

kubectl get pod -o wide

**NODEPORT:**

**34.** vi service.yml

apiVersion: v1

kind: Service

metadata:

name: my-service

spec:

type: NodePort

selector:

app: zomato

ports:

- port: 80

targetPort: 80

nodePort: 30080 **# You can specify the nodePort, or Kubernetes will automatically assign one in the range 30000-32767**

**35.** kubectl apply -f service.yml

**LOADBLANCER:**

**36.** vi service.yml

apiVersion: v1

kind: Service

metadata:

name: my-service

spec:

type: LoadBalancer

selector:

app: zomato

ports:

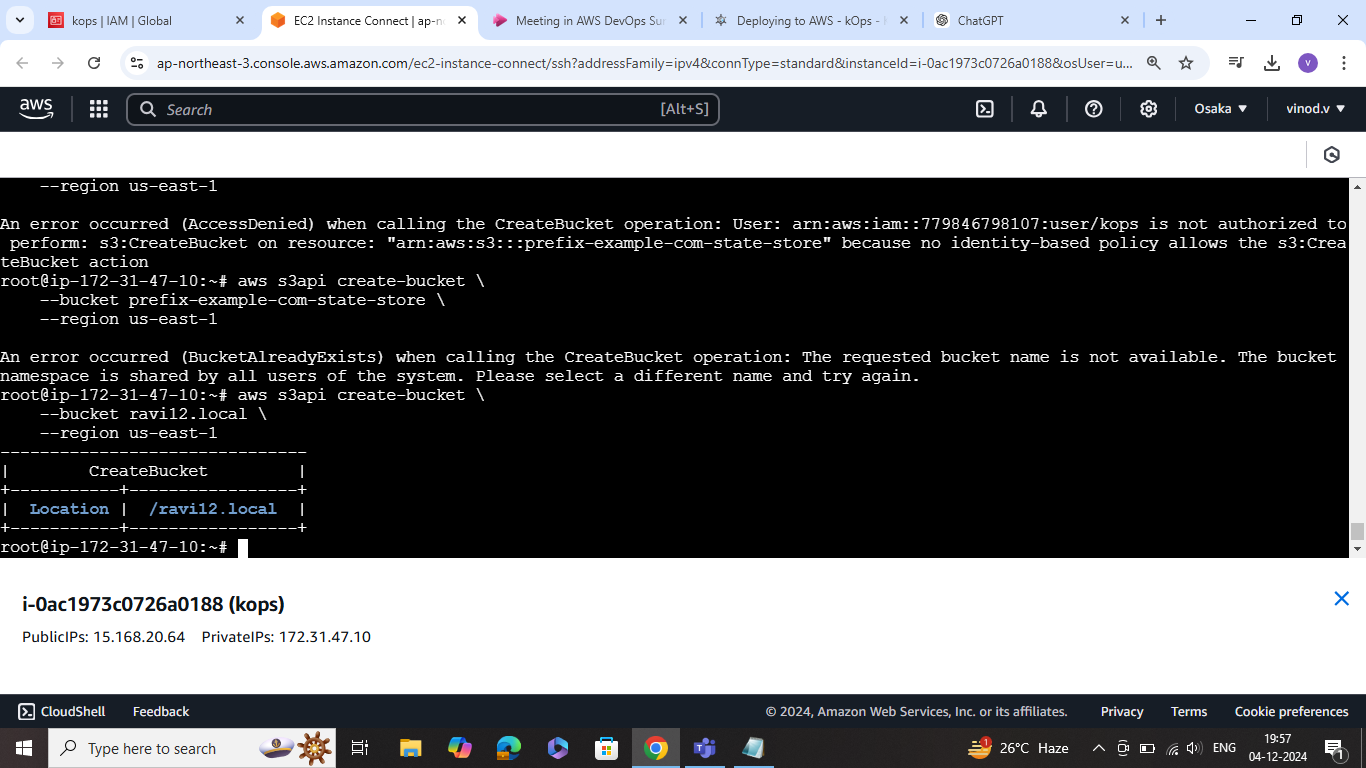
- port: 80

targetPort: 80

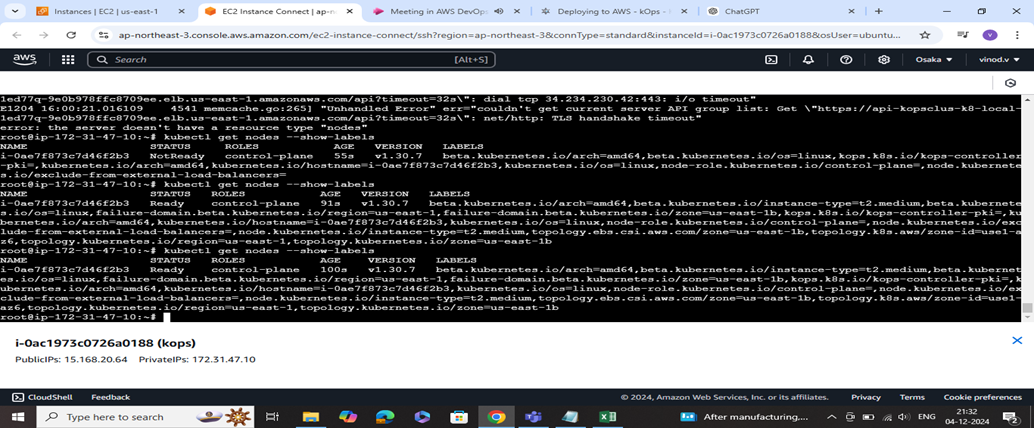
**37.** kubectl apply -f service.yml

**38.** kops delete cluster --name kopsclus.k8.local --yes **( DELETE THE Auto healing, Auto scaling,Loadbalancer, Instances ,ClusterIp)**

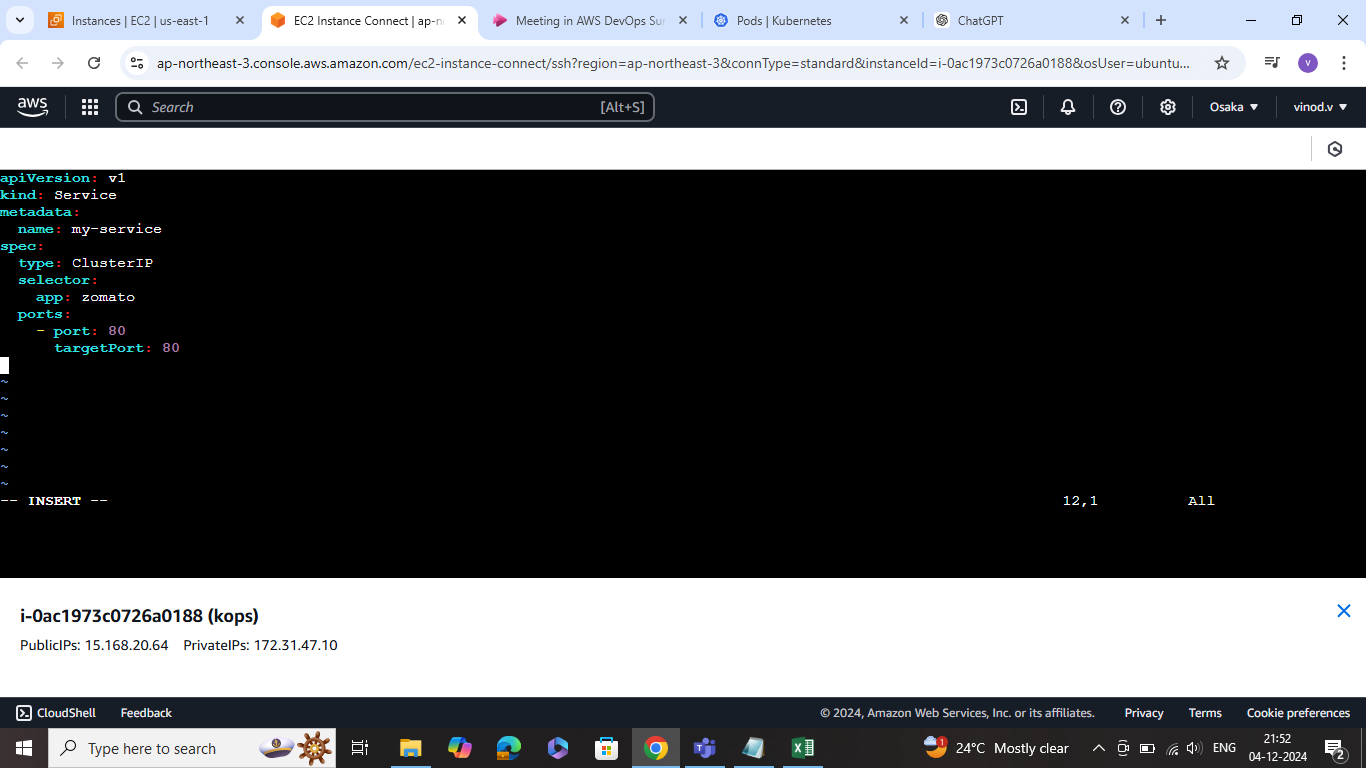
**Creation of the bucket**

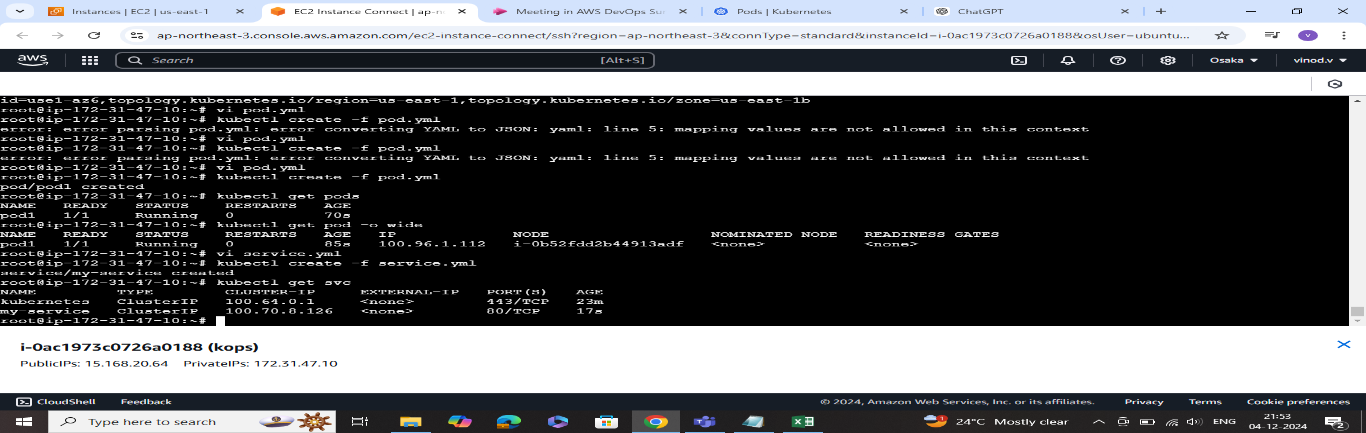


**Until we give the command came to the ready state for worker node and master node**

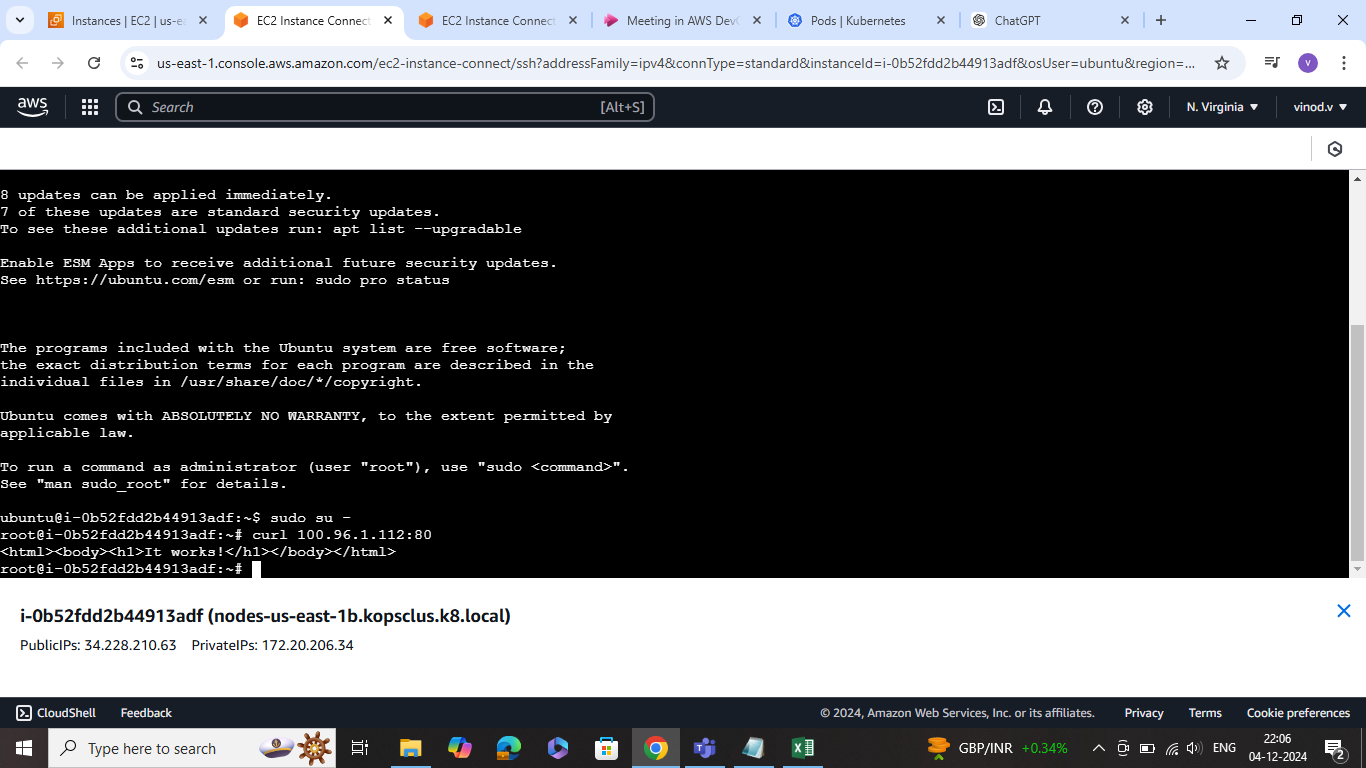
****

**This is creation of yml file for clusterip**

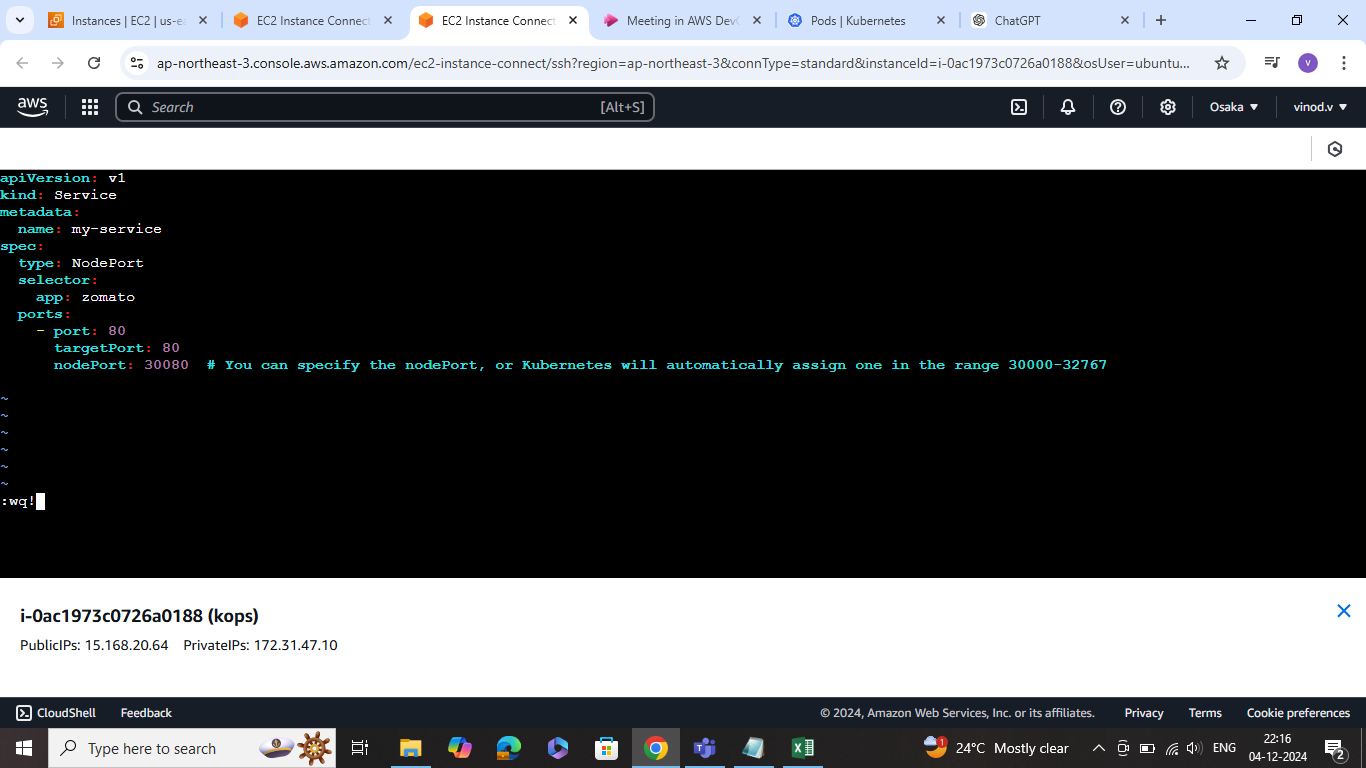


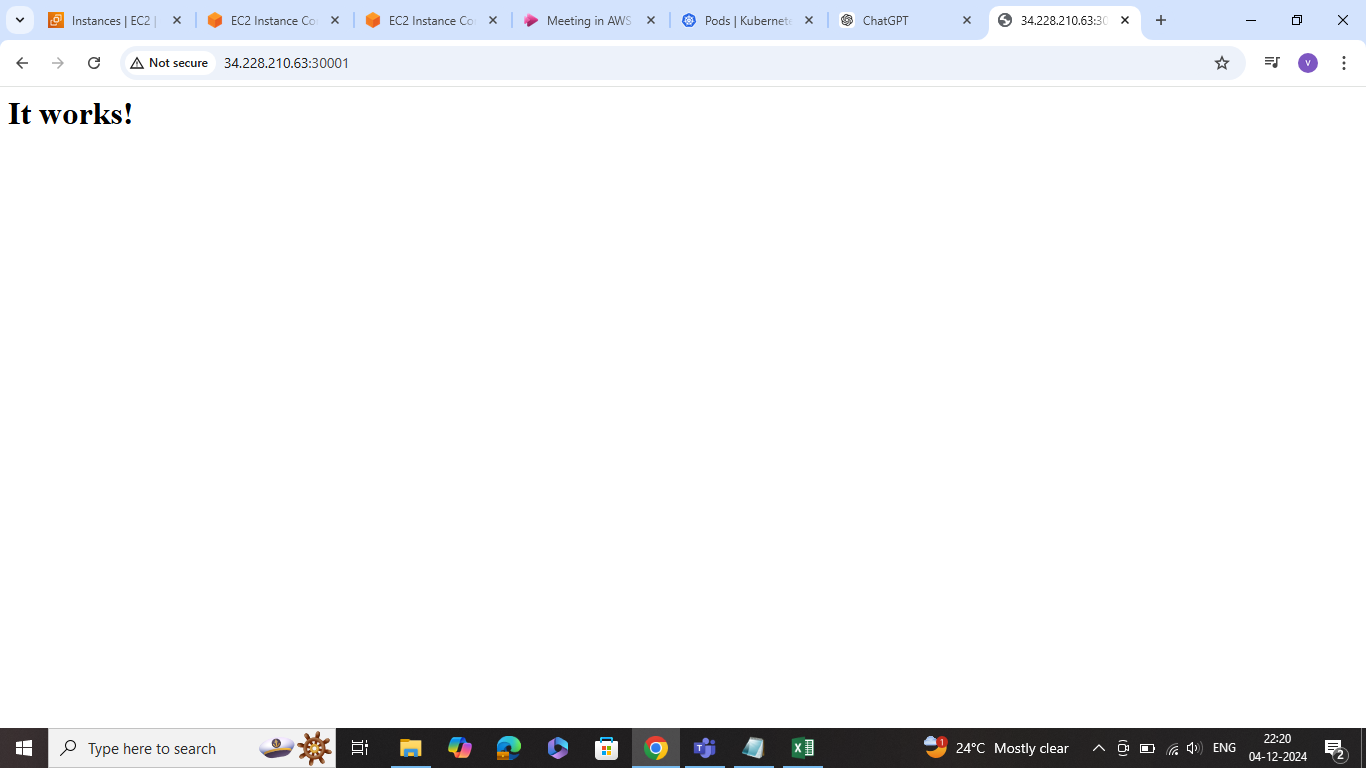


**This is for the master node access the public ip in server**



**This is Nodeport yml file**





**DELETE THE Auto healing, Auto scaling,Loadbalancer, Instances ,ClusterIp**

