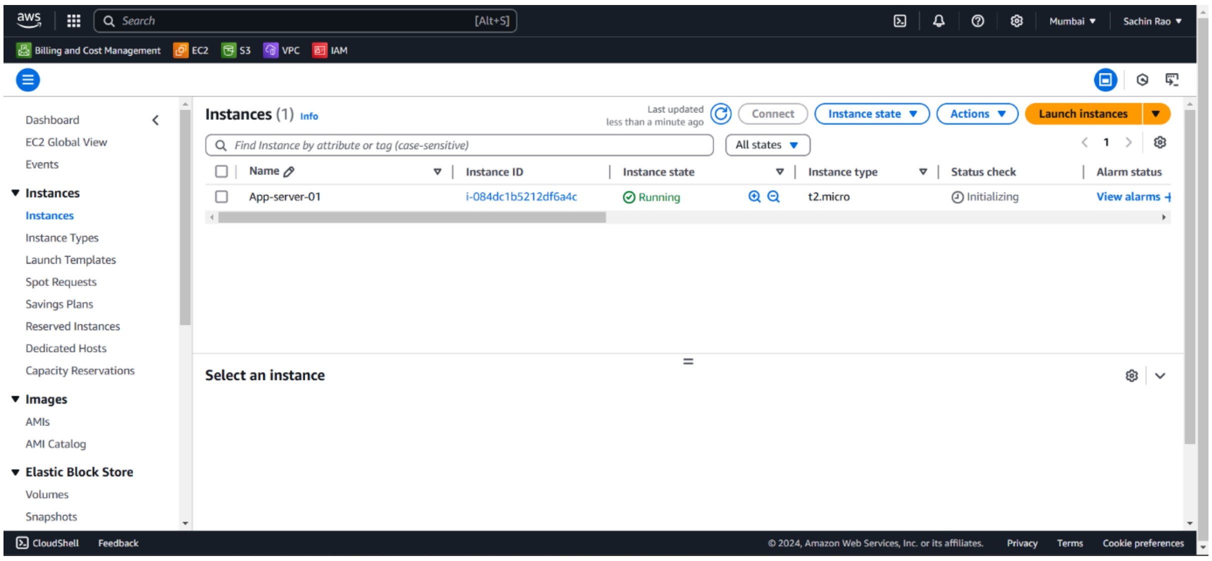
BRAINWAVE MATRIX SOLUTIONS

Task 1(Video link):
https://drive.google.com/file/d/1sn4aBdfJtUpfDtsYAoxPHQaixZG5vT-l/view?usp=sharing

TASK --1-CLOUD COMPUTING INTERNSHIP
DEPLOY A WEB APPLICATION IN AWS/KUBERNETES

Step 1:- **Launch a New EC2 Instance (Amazon Linux - t2.micro)**

- Go to the AWS Management Console and launch a new EC2 instance.
- Choose "Amazon Linux 2 AMI" as the operating system.
- Select an instance type (t2.micro is a good choice for testing purposes and free tier).
- Create new key pair and download it.Configure security groups, allowing SSH (port 22) and HTTP (port 80).
- Launch the instance and connect to it via SSH (In MobaXterm Tool).



Instances (1) Info

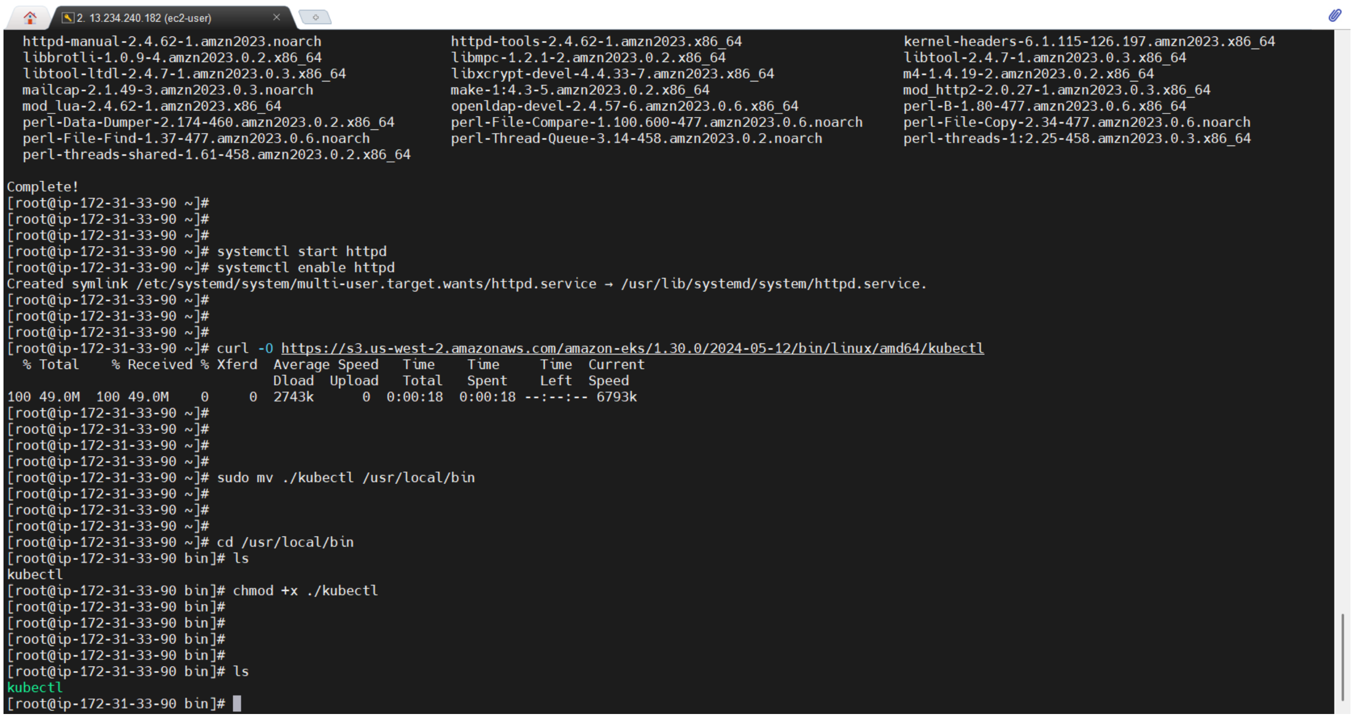
Name	Instance ID	Instance state	Instance type	Status check	Alarm status
App-server-01	i-084dc1b5212df6a4c	Running	t2.micro	Initializing	View alarms +

Step 2:- **Install kubectl (Kubernetes CLI)**

- After connecting to the EC2 instance, run the following commands to install kubectl:

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/amd64/kubectl
sudo mv ./kubectl /usr/local/bin
chmod +x ./kubectl
kubectl version
```

- This will download kubectl, move it to /usr/local/bin for execution, apply the necessary permissions, and verify the version.

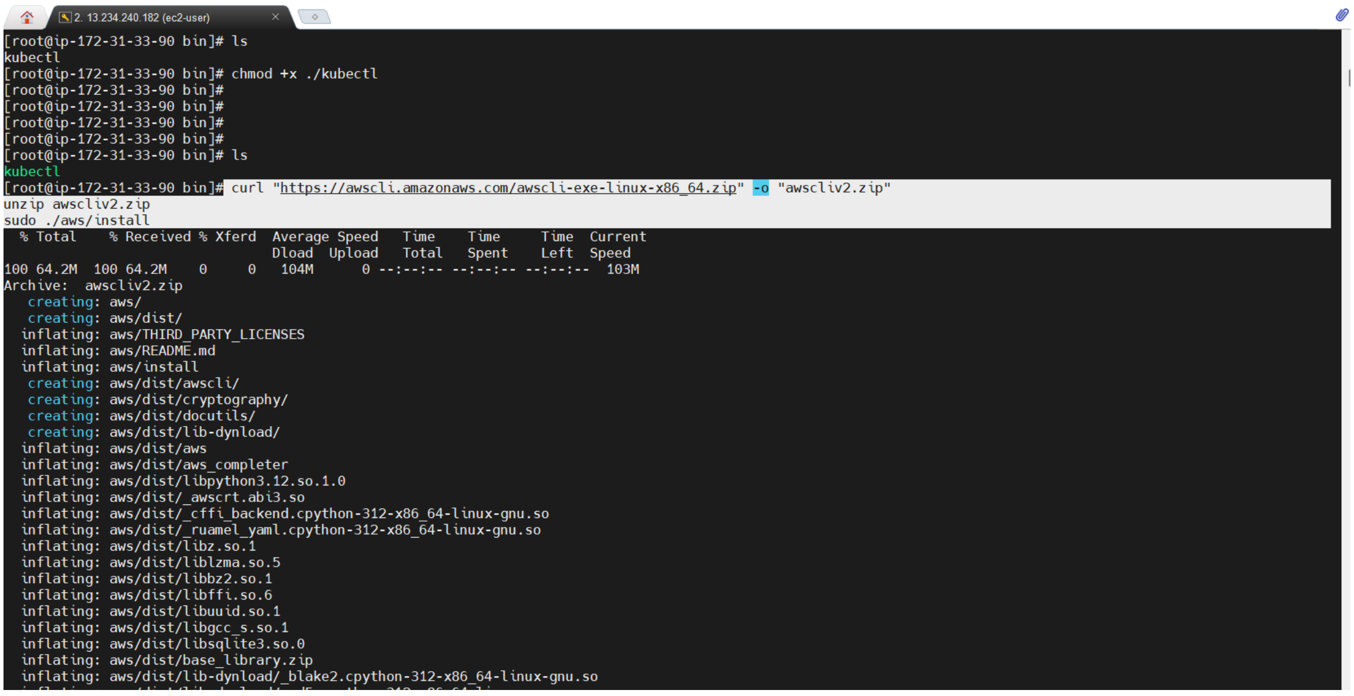

2. 13.234.240.182 (ec2-user)

```
  httpd-manual-2.4.62-1.amzn2023.noarch                  httpd-tools-2.4.62-1.amzn2023.x86_64                      kernel-headers-6.1.115-126.197.amzn2023.x86_64
  libbrotli-1.0.9-4.amzn2023.0.2.x86_64                  libmpc-1.2.1-2.amzn2023.0.2.x86_64                        libtool-2.4.7-1.amzn2023.0.3.x86_64
  libtool-ltdl-2.4.7-1.amzn2023.0.3.x86_64               libxcrypt-devel-4.4.33-7.amzn2023.x86_64                  m4-1.4.19-2.amzn2023.0.2.x86_64
  mailcap-2.1.49-3.amzn2023.0.3.noarch                   make-1:4.3-5.amzn2023.0.2.x86_64                          mod_http2-2.0.27-1.amzn2023.0.3.x86_64
  mod_lua-2.4.62-1.amzn2023.x86_64                       openldap-devel-2.4.57-6.amzn2023.0.6.x86_64               perl-B-1.80-477.amzn2023.0.6.x86_64
  perl-Data-Dumper-2.174-460.amzn2023.0.2.x86_64         perl-File-Compare-1.100.600-477.amzn2023.0.6.noarch       perl-File-Copy-2.34-477.amzn2023.0.6.noarch
  perl-File-Find-1.37-477.amzn2023.0.6.noarch            perl-Thread-Queue-3.14-458.amzn2023.0.2.noarch            perl-threads-1:2.25-458.amzn2023.0.3.x86_64
  perl-threads-shared-1.61-458.amzn2023.0.2.x86_64

Complete!
[root@ip-172-31-33-90 ~]#
[root@ip-172-31-33-90 ~]#
[root@ip-172-31-33-90 ~]#
[root@ip-172-31-33-90 ~]# systemctl start httpd
[root@ip-172-31-33-90 ~]# systemctl enable httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
[root@ip-172-31-33-90 ~]#
[root@ip-172-31-33-90 ~]#
[root@ip-172-31-33-90 ~]#
[root@ip-172-31-33-90 ~]# curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/amd64/kubectl
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 49.0M  100 49.0M    0     0  2743k      0  0:00:18  0:00:18 --:--:-- 6793k
[root@ip-172-31-33-90 ~]#
[root@ip-172-31-33-90 ~]#
[root@ip-172-31-33-90 ~]#
[root@ip-172-31-33-90 ~]#
[root@ip-172-31-33-90 ~]# sudo mv ./kubectl /usr/local/bin
[root@ip-172-31-33-90 ~]#
[root@ip-172-31-33-90 ~]#
[root@ip-172-31-33-90 ~]#
[root@ip-172-31-33-90 ~]# cd /usr/local/bin
[root@ip-172-31-33-90 bin]# ls
kubectl
[root@ip-172-31-33-90 bin]# chmod +x ./kubectl
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# ls
kubectl
[root@ip-172-31-33-90 bin]#
```

Step 3:- **Install AWS CLI**

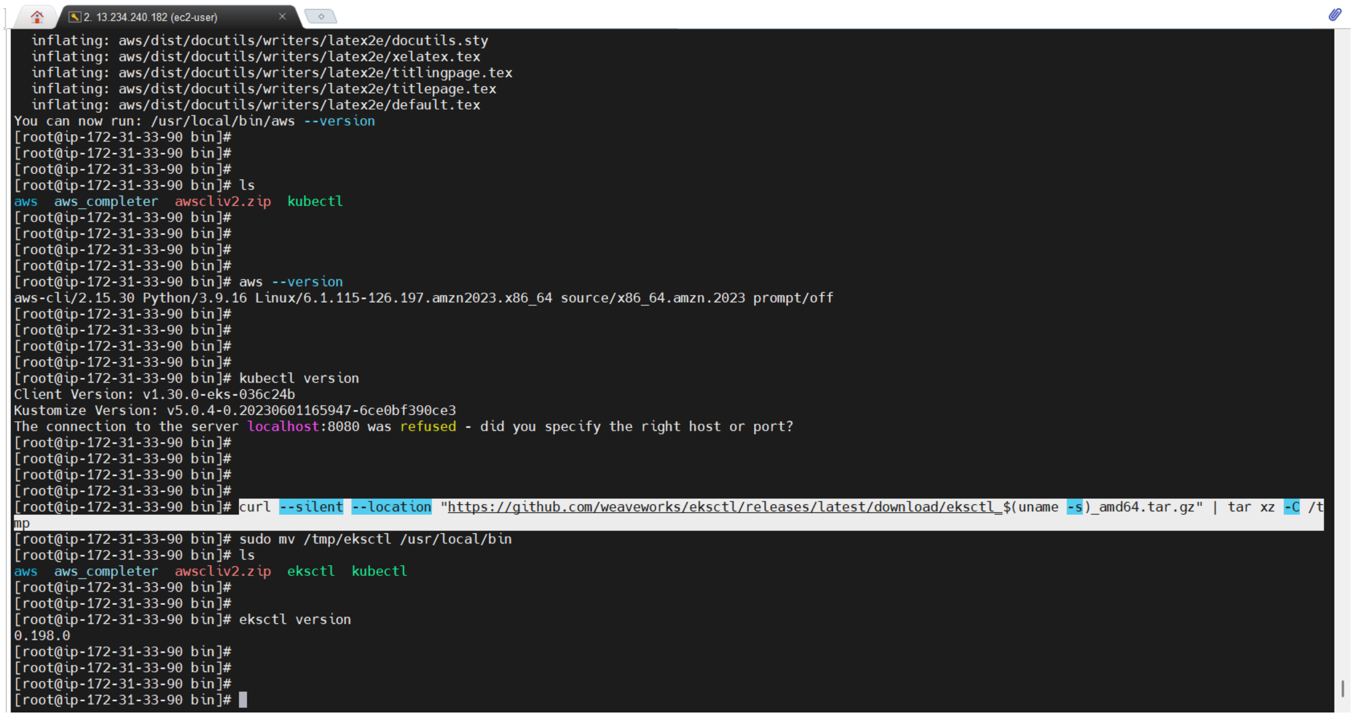
- Install the AWS Command Line Interface (CLI) using these commands:
  curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
  unzip awscliv2.zip
  sudo ./aws/install
  aws --version
- This installs the latest version of the AWS CLI and verifies the installation.


2. 13.234.240.182 (ec2-user)

```
[root@ip-172-31-33-90 bin]# ls
kubectl
[root@ip-172-31-33-90 bin]# chmod +x ./kubectl
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# ls
kubectl
[root@ip-172-31-33-90 bin]# curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 64.2M  100 64.2M    0     0   104M      0 --:--:-- --:--:-- --:--:--  103M
Archive:  awscliv2.zip
   creating: aws/
   creating: aws/dist/
  inflating: aws/THIRD_PARTY_LICENSES
  inflating: aws/README.md
  inflating: aws/install
   creating: aws/dist/awscli/
   creating: aws/dist/cryptography/
   creating: aws/dist/docutils/
   creating: aws/dist/lib-dynload/
  inflating: aws/dist/aws
  inflating: aws/dist/aws_completer
  inflating: aws/dist/libpython3.12.so.1.0
  inflating: aws/dist/_awscrt.abi3.so
  inflating: aws/dist/_cffi_backend.cpython-312-x86_64-linux-gnu.so
  inflating: aws/dist/_ruamel_yaml.cpython-312-x86_64-linux-gnu.so
  inflating: aws/dist/libz.so.1
  inflating: aws/dist/liblzma.so.5
  inflating: aws/dist/libbz2.so.1
  inflating: aws/dist/libffi.so.6
  inflating: aws/dist/libuuid.so.1
  inflating: aws/dist/libgcc_s.so.1
  inflating: aws/dist/libsqlite3.so.0
  inflating: aws/dist/base_library.zip
  inflating: aws/dist/lib-dynload/_blake2.cpython-312-x86_64-linux-gnu.so
```

Step 4:- **Install eksctl (EKS Cluster Management Tool)**

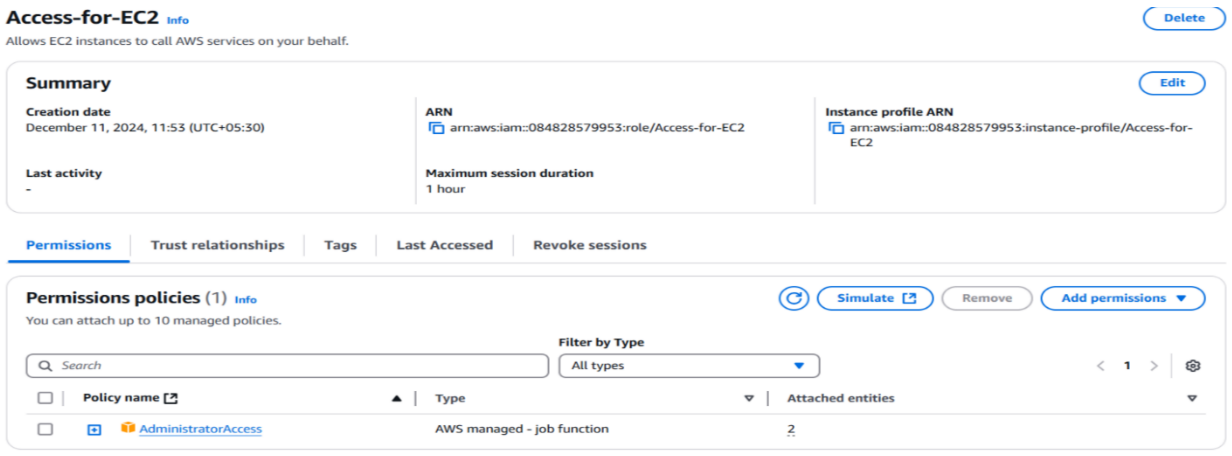
- To simplify EKS cluster creation, install eksctl:
  curl --silent --location "https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_$(uname -s)_amd64.tar.gz" | tar xz -C /tmp
  sudo mv /tmp/eksctl /usr/local/bin
  eksctl version
- This command downloads and installs eksctl, a command-line tool for managing EKS clusters.



```
  inflating: aws/dist/docutils/writers/latex2e/docutils.sty
  inflating: aws/dist/docutils/writers/latex2e/xelatex.tex
  inflating: aws/dist/docutils/writers/latex2e/titlingpage.tex
  inflating: aws/dist/docutils/writers/latex2e/titlepage.tex
  inflating: aws/dist/docutils/writers/latex2e/default.tex
You can now run: /usr/local/bin/aws --version
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# ls
aws  aws_completer  awscliv2.zip  kubectl
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# aws --version
aws-cli/2.15.30 Python/3.9.16 Linux/6.1.115-126.197.amzn2023.x86_64 source/x86_64.amzn.2023 prompt/off
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# kubectl version
Client Version: v1.30.0-eks-036c24b
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
The connection to the server localhost:8080 was refused - did you specify the right host or port?
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# curl --silent --location "https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_$(uname -s)_amd64.tar.gz" | tar xz -C /t
mp
[root@ip-172-31-33-90 bin]# sudo mv /tmp/eksctl /usr/local/bin
[root@ip-172-31-33-90 bin]# ls
aws  aws_completer  awscliv2.zip  eksctl  kubectl
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# eksctl version
0.198.0
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
```

Step 5:- **Create a New IAM Role**

- Go to the IAM section in the AWS Console and create a new role with the following permissions:
  - IAM FullAccess – We need create all these roles individually (or) otherwise we give one role is "AdministratorAccess" (VPC FullAccess,EC2 FullAccess,CloudFormation FullAccess).



Access-for-EC2 Info

Allows EC2 instances to call AWS services on your behalf.

Summary

Creation date: December 11, 2024, 11:53 (UTC+05:30)

ARN: arn:aws:iam::084828579953:role/Access-for-EC2

Instance profile ARN: arn:aws:iam::084828579953:instance-profile/Access-for-EC2

Last activity: -

Maximum session duration: 1 hour

Permissions policies (1) Info

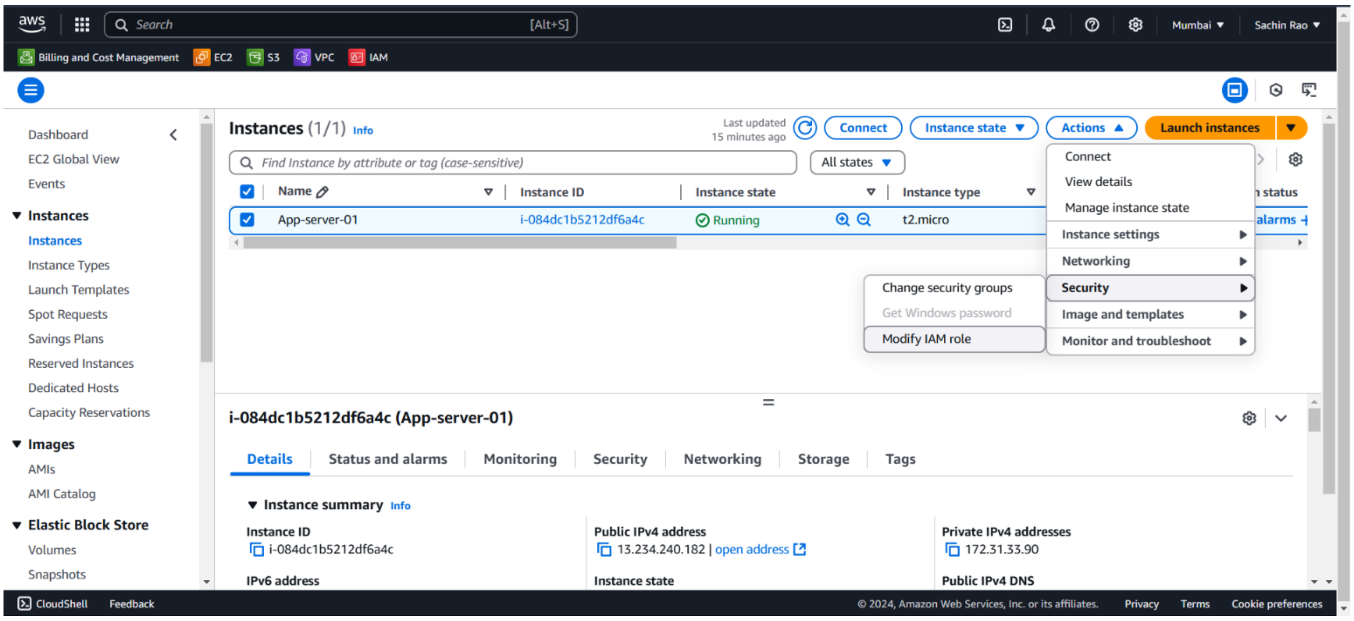
You can attach up to 10 managed policies.

Filter by Type: All types

Policy name	Type	Attached entities
AdministratorAccess	AWS managed - job function	2

Step 6:- **Attach IAM Role to the EC2 Instance**

- Attach the created role to the EC2 instance used as the management host. This allows the instance to interact with other AWS services.



Name	Instance ID	Instance state	Instance type
App-server-01	i-084dc1b5212df6a4c	Running	t2.micro

i-084dc1b5212df6a4c (App-server-01)

Instance ID: i-084dc1b5212df6a4c

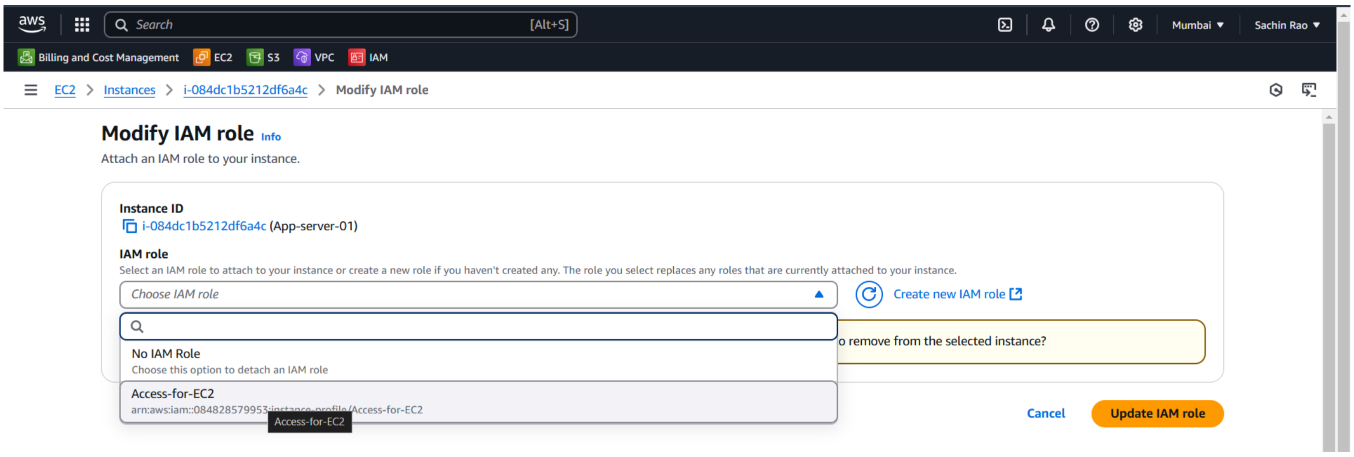
Public IPv4 address: 13.234.240.182 | open address

Private IPv4 addresses: 172.31.33.90

IPv6 address

Instance state

Public IPv4 DNS



Modify IAM role

Attach an IAM role to your instance.

Instance ID

i-084dc1b5212df6a4c (App-server-01)

IAM role

Select an IAM role to attach to your instance or create a new role if you haven't created any. The role you select replaces any roles that are currently attached to your instance.

Choose IAM role

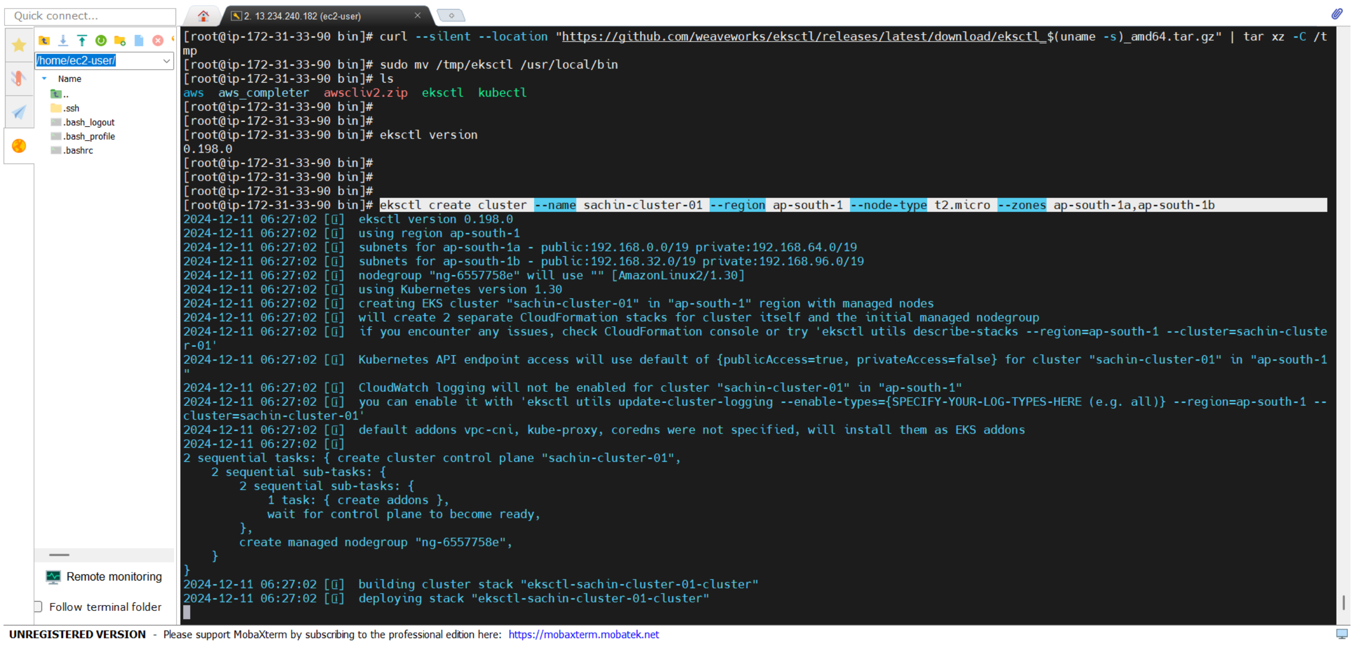
No IAM Role
Choose this option to detach an IAM role

Access-for-EC2
arn:aws:iam::084828579953:instance-profile/Access-for-EC2

o remove from the selected instance?

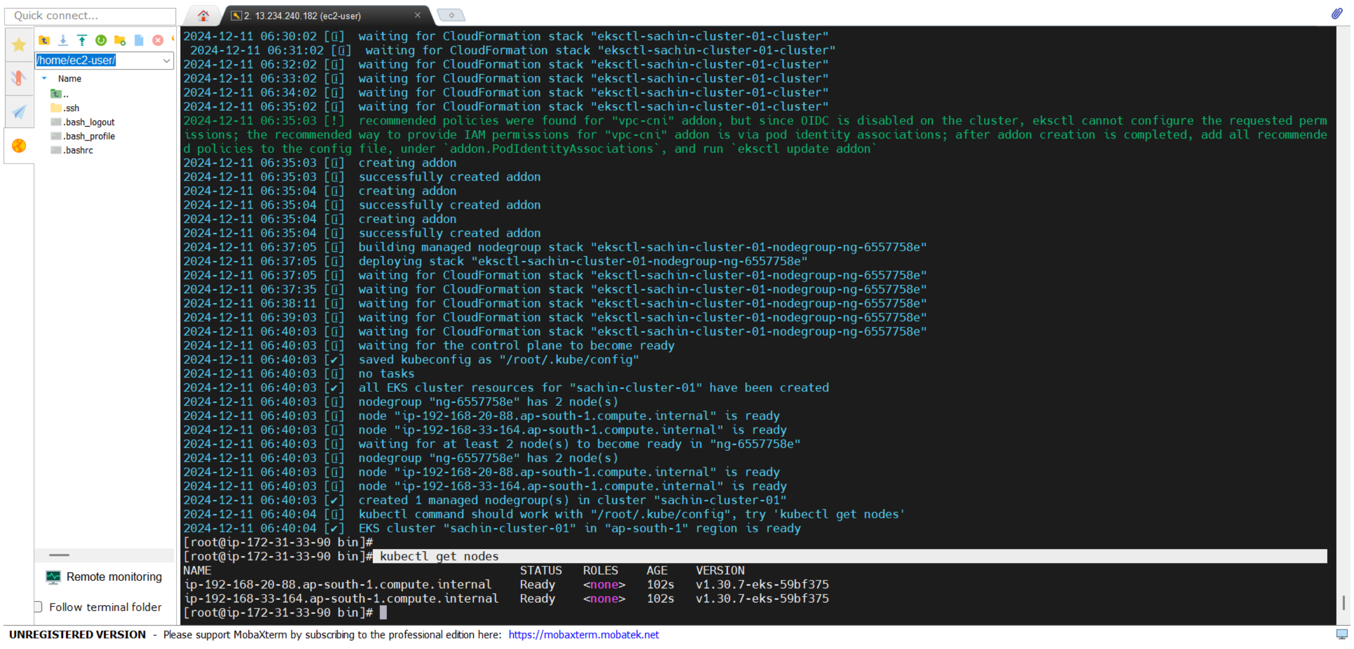
Step 7:- **Create EKS Cluster using eksctl**

- To create the EKS cluster, use the eksctl command.
  Mumbai (ap-south-1):
  eksctl create cluster --name demo-cluster --region ap-south-1 --node-type t2.micro --zones ap-south-1a,ap-south-1b
- This will create a new EKS cluster in the specified region, using the specified instance type and zones.



```
[root@ip-172-31-33-90 bin]# curl --silent --location "https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_$(uname -s)_amd64.tar.gz" | tar xz -C /t
mp
[root@ip-172-31-33-90 bin]# sudo mv /tmp/eksctl /usr/local/bin
[root@ip-172-31-33-90 bin]# ls
aws  aws_completer  awscliv2.zip  eksctl  kubectl
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# eksctl version
0.198.0
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# eksctl create cluster --name sachin-cluster-01 --region ap-south-1 --node-type t2.micro --zones ap-south-1a,ap-south-1b
2024-12-11 06:27:02 [ℹ]  eksctl version 0.198.0
2024-12-11 06:27:02 [ℹ]  using region ap-south-1
2024-12-11 06:27:02 [ℹ]  subnets for ap-south-1a - public:192.168.0.0/19 private:192.168.64.0/19
2024-12-11 06:27:02 [ℹ]  subnets for ap-south-1b - public:192.168.32.0/19 private:192.168.96.0/19
2024-12-11 06:27:02 [ℹ]  nodegroup "ng-6557758e" will use "" [AmazonLinux2/1.30]
2024-12-11 06:27:02 [ℹ]  using Kubernetes version 1.30
2024-12-11 06:27:02 [ℹ]  creating EKS cluster "sachin-cluster-01" in "ap-south-1" region with managed nodes
2024-12-11 06:27:02 [ℹ]  will create 2 separate CloudFormation stacks for cluster itself and the initial managed nodegroup
2024-12-11 06:27:02 [ℹ]  if you encounter any issues, check CloudFormation console or try 'eksctl utils describe-stacks --region=ap-south-1 --cluster=sachin-cluste
r-01'
2024-12-11 06:27:02 [ℹ]  Kubernetes API endpoint access will use default of {publicAccess=true, privateAccess=false} for cluster "sachin-cluster-01" in "ap-south-1
"
2024-12-11 06:27:02 [ℹ]  CloudWatch logging will not be enabled for cluster "sachin-cluster-01" in "ap-south-1"
2024-12-11 06:27:02 [ℹ]  you can enable it with 'eksctl utils update-cluster-logging --enable-types={SPECIFY-YOUR-LOG-TYPES-HERE (e.g. all)} --region=ap-south-1 --
cluster=sachin-cluster-01'
2024-12-11 06:27:02 [ℹ]  default addons vpc-cni, kube-proxy, coredns were not specified, will install them as EKS addons
2024-12-11 06:27:02 [ℹ]
2 sequential tasks: { create cluster control plane "sachin-cluster-01",
    2 sequential sub-tasks: {
        2 sequential sub-tasks: {
            1 task: { create addons },
            wait for control plane to become ready,
        },
        create managed nodegroup "ng-6557758e",
    }
}
2024-12-11 06:27:02 [ℹ]  building cluster stack "eksctl-sachin-cluster-01-cluster"
2024-12-11 06:27:02 [ℹ]  deploying stack "eksctl-sachin-cluster-01-cluster"
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

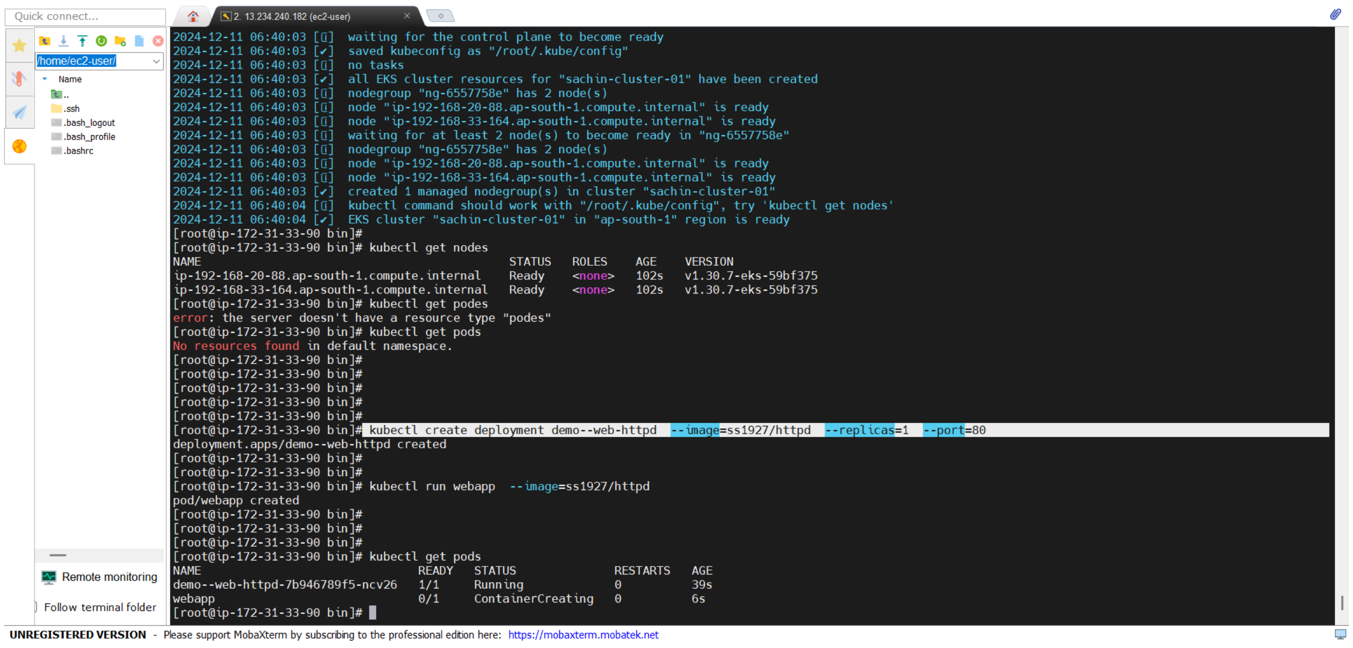


```
2024-12-11 06:30:02 [ℹ]  waiting for CloudFormation stack "eksctl-sachin-cluster-01-cluster"
 2024-12-11 06:31:02 [ℹ]  waiting for CloudFormation stack "eksctl-sachin-cluster-01-cluster"
2024-12-11 06:32:02 [ℹ]  waiting for CloudFormation stack "eksctl-sachin-cluster-01-cluster"
2024-12-11 06:33:02 [ℹ]  waiting for CloudFormation stack "eksctl-sachin-cluster-01-cluster"
2024-12-11 06:34:02 [ℹ]  waiting for CloudFormation stack "eksctl-sachin-cluster-01-cluster"
2024-12-11 06:35:02 [ℹ]  waiting for CloudFormation stack "eksctl-sachin-cluster-01-cluster"
2024-12-11 06:35:03 [!]  recommended policies were found for "vpc-cni" addon, but since OIDC is disabled on the cluster, eksctl cannot configure the requested perm
issions; the recommended way to provide IAM permissions for "vpc-cni" addon is via pod identity associations; after addon creation is completed, add all recommende
d policies to the config file, under `addon.PodIdentityAssociations`, and run `eksctl update addon`
2024-12-11 06:35:03 [ℹ]  creating addon
2024-12-11 06:35:03 [ℹ]  successfully created addon
2024-12-11 06:35:04 [ℹ]  creating addon
2024-12-11 06:35:04 [ℹ]  successfully created addon
2024-12-11 06:35:04 [ℹ]  creating addon
2024-12-11 06:35:04 [ℹ]  successfully created addon
2024-12-11 06:37:05 [ℹ]  building managed nodegroup stack "eksctl-sachin-cluster-01-nodegroup-ng-6557758e"
2024-12-11 06:37:05 [ℹ]  deploying stack "eksctl-sachin-cluster-01-nodegroup-ng-6557758e"
2024-12-11 06:37:05 [ℹ]  waiting for CloudFormation stack "eksctl-sachin-cluster-01-nodegroup-ng-6557758e"
2024-12-11 06:37:35 [ℹ]  waiting for CloudFormation stack "eksctl-sachin-cluster-01-nodegroup-ng-6557758e"
2024-12-11 06:38:11 [ℹ]  waiting for CloudFormation stack "eksctl-sachin-cluster-01-nodegroup-ng-6557758e"
2024-12-11 06:39:03 [ℹ]  waiting for CloudFormation stack "eksctl-sachin-cluster-01-nodegroup-ng-6557758e"
2024-12-11 06:40:03 [ℹ]  waiting for CloudFormation stack "eksctl-sachin-cluster-01-nodegroup-ng-6557758e"
2024-12-11 06:40:03 [ℹ]  waiting for the control plane to become ready
2024-12-11 06:40:03 [✔]  saved kubeconfig as "/root/.kube/config"
2024-12-11 06:40:03 [ℹ]  no tasks
2024-12-11 06:40:03 [✔]  all EKS cluster resources for "sachin-cluster-01" have been created
2024-12-11 06:40:03 [ℹ]  nodegroup "ng-6557758e" has 2 node(s)
2024-12-11 06:40:03 [ℹ]  node "ip-192-168-20-88.ap-south-1.compute.internal" is ready
2024-12-11 06:40:03 [ℹ]  node "ip-192-168-33-164.ap-south-1.compute.internal" is ready
2024-12-11 06:40:03 [ℹ]  waiting for at least 2 node(s) to become ready in "ng-6557758e"
2024-12-11 06:40:03 [ℹ]  nodegroup "ng-6557758e" has 2 node(s)
2024-12-11 06:40:03 [ℹ]  node "ip-192-168-20-88.ap-south-1.compute.internal" is ready
2024-12-11 06:40:03 [ℹ]  node "ip-192-168-33-164.ap-south-1.compute.internal" is ready
2024-12-11 06:40:03 [✔]  created 1 managed nodegroup(s) in cluster "sachin-cluster-01"
2024-12-11 06:40:04 [ℹ]  kubectl command should work with "/root/.kube/config", try 'kubectl get nodes'
2024-12-11 06:40:04 [✔]  EKS cluster "sachin-cluster-01" in "ap-south-1" region is ready
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# kubectl get nodes
NAME                                            STATUS   ROLES    AGE    VERSION
ip-192-168-20-88.ap-south-1.compute.internal    Ready    <none>   102s   v1.30.7-eks-59bf375
ip-192-168-33-164.ap-south-1.compute.internal   Ready    <none>   102s   v1.30.7-eks-59bf375
[root@ip-172-31-33-90 bin]#
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

Step 8:- **Deploy Nginx Pods on Kubernetes**

- Create a Deployment for Nginx: To deploy an given demo-web-httpd application with 1 replicas:
  `kubectl create deployment demo-web-httpd  --image=ss1927/httpd --replicas=1   --port=80`
- Check the Status:
  List all resources - *kubectl get all*
  List the running pods - *kubectl get pods*
- After all this,To run the webapp with help this command:
  Kubectl run webapp –image=ss1927/httpd

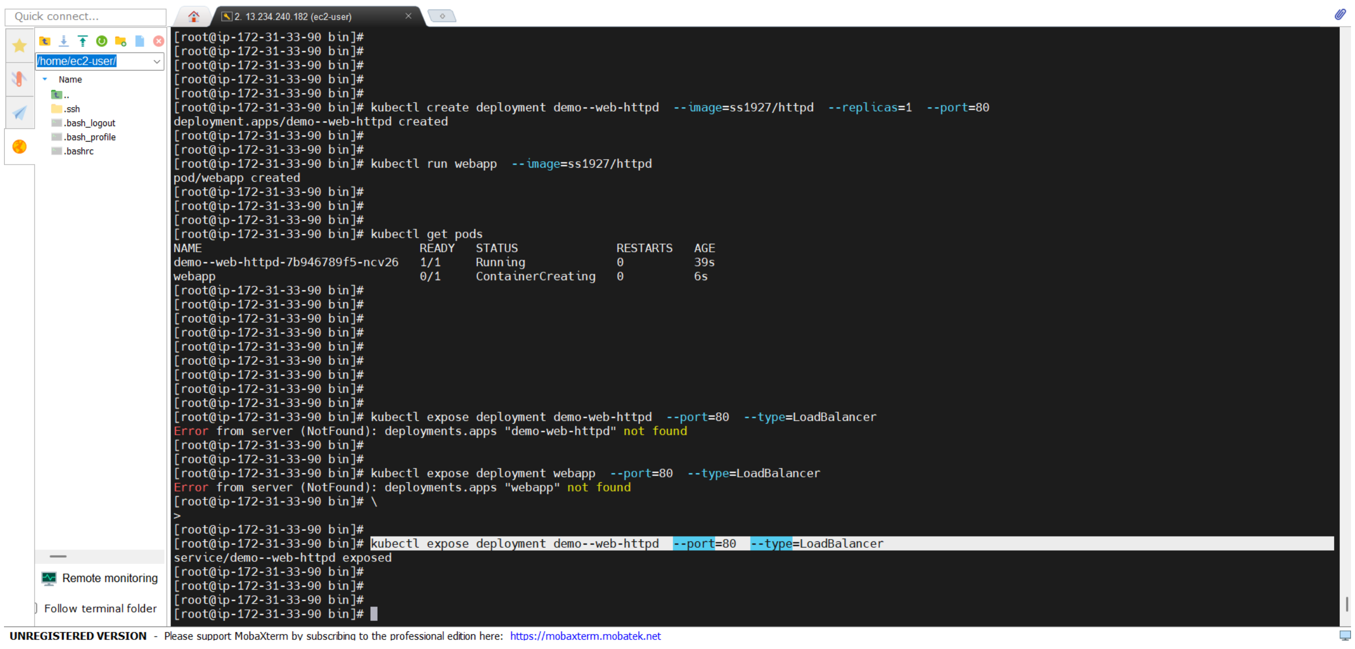


```
2024-12-11 06:40:03 [▯]  waiting for the control plane to become ready
2024-12-11 06:40:03 [✔]  saved kubeconfig as "/root/.kube/config"
2024-12-11 06:40:03 [▯]  no tasks
2024-12-11 06:40:03 [✔]  all EKS cluster resources for "sachin-cluster-01" have been created
2024-12-11 06:40:03 [▯]  nodegroup "ng-6557758e" has 2 node(s)
2024-12-11 06:40:03 [▯]  node "ip-192-168-20-88.ap-south-1.compute.internal" is ready
2024-12-11 06:40:03 [▯]  node "ip-192-168-33-164.ap-south-1.compute.internal" is ready
2024-12-11 06:40:03 [▯]  waiting for at least 2 node(s) to become ready in "ng-6557758e"
2024-12-11 06:40:03 [▯]  nodegroup "ng-6557758e" has 2 node(s)
2024-12-11 06:40:03 [▯]  node "ip-192-168-20-88.ap-south-1.compute.internal" is ready
2024-12-11 06:40:03 [▯]  node "ip-192-168-33-164.ap-south-1.compute.internal" is ready
2024-12-11 06:40:03 [✔]  created 1 managed nodegroup(s) in cluster "sachin-cluster-01"
2024-12-11 06:40:04 [▯]  kubectl command should work with "/root/.kube/config", try 'kubectl get nodes'
2024-12-11 06:40:04 [✔]  EKS cluster "sachin-cluster-01" in "ap-south-1" region is ready
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# kubectl get nodes
NAME                                            STATUS   ROLES    AGE    VERSION
ip-192-168-20-88.ap-south-1.compute.internal    Ready    <none>   102s   v1.30.7-eks-59bf375
ip-192-168-33-164.ap-south-1.compute.internal   Ready    <none>   102s   v1.30.7-eks-59bf375
[root@ip-172-31-33-90 bin]# kubectl get podes
error: the server doesn't have a resource type "podes"
[root@ip-172-31-33-90 bin]# kubectl get pods
No resources found in default namespace.
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# kubectl create deployment demo--web-httpd  --image=ss1927/httpd  --replicas=1  --port=80
deployment.apps/demo--web-httpd created
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# kubectl run webapp  --image=ss1927/httpd
pod/webapp created
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# kubectl get pods
NAME                              READY   STATUS              RESTARTS   AGE
demo--web-httpd-7b946789f5-ncv26  1/1     Running             0          39s
webapp                            0/1     ContainerCreating   0          6s
[root@ip-172-31-33-90 bin]# 
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

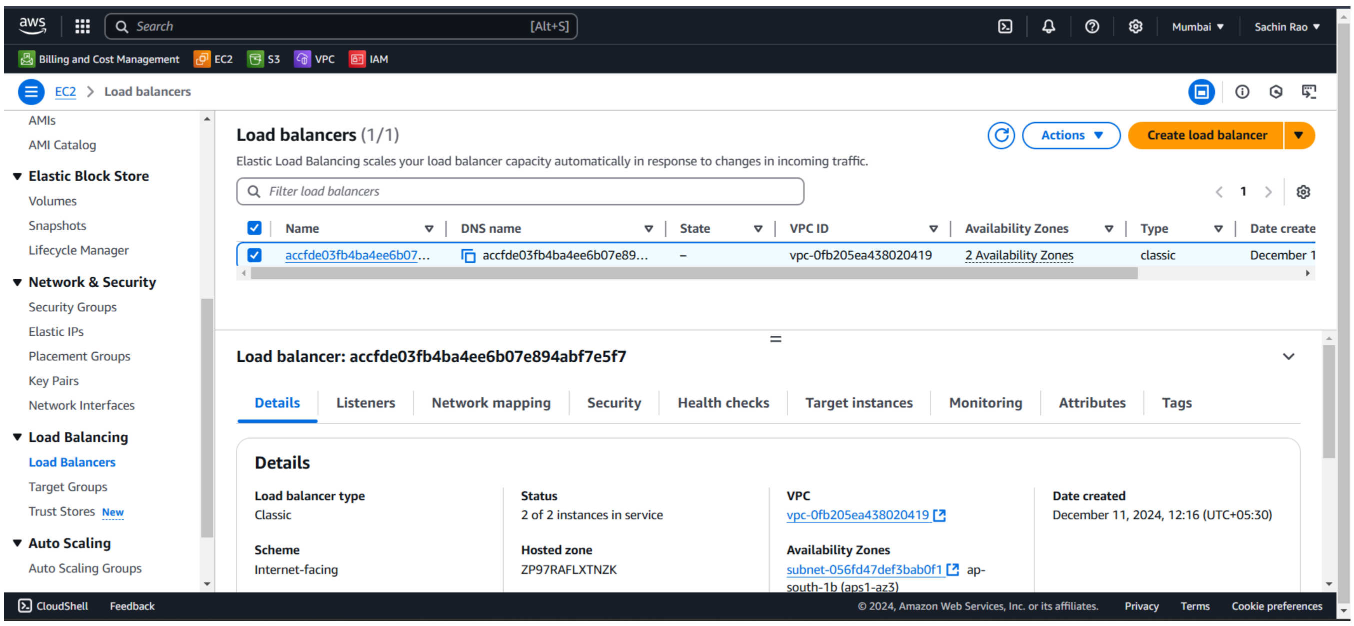
Step 9:- **Expose the Deployment as a Service**

- To expose the Nginx deployment via a LoadBalancer, follow these steps:
  Expose the Deployment: *kubectl expose deployment demo--web-httpd --port=80 --type=LoadBalancer*
  Check the Service: To see the service details and external IP (LoadBalancer IP)
  kubectl get services -o wide



```
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# kubectl create deployment demo--web-httpd  --image=ss1927/httpd  --replicas=1  --port=80
deployment.apps/demo--web-httpd created
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# kubectl run webapp  --image=ss1927/httpd
pod/webapp created
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# kubectl get pods
NAME                              READY   STATUS              RESTARTS   AGE
demo--web-httpd-7b946789f5-ncv26  1/1     Running             0          39s
webapp                            0/1     ContainerCreating   0          6s
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# kubectl expose deployment demo-web-httpd  --port=80  --type=LoadBalancer
Error from server (NotFound): deployments.apps "demo-web-httpd" not found
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# kubectl expose deployment webapp  --port=80  --type=LoadBalancer
Error from server (NotFound): deployments.apps "webapp" not found
[root@ip-172-31-33-90 bin]# \
>
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# kubectl expose deployment demo--web-httpd  --port=80  --type=LoadBalancer
service/demo--web-httpd exposed
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]#
[root@ip-172-31-33-90 bin]# 
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net



Load balancers (1/1)

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

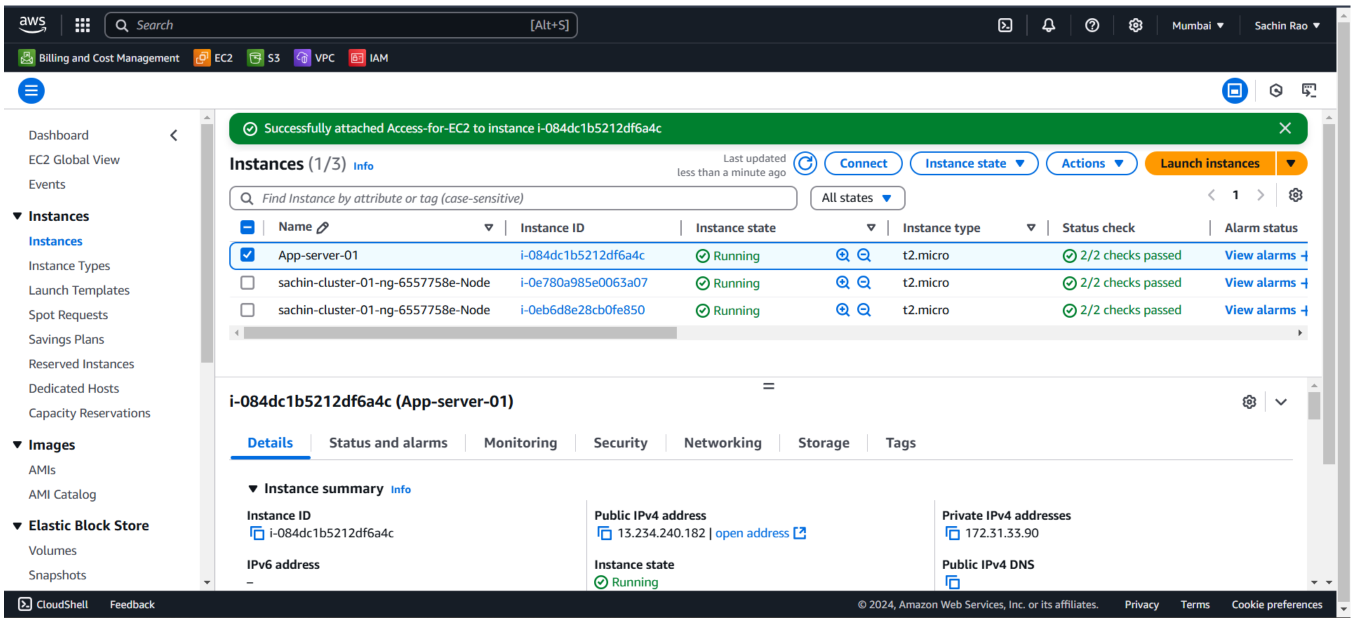
	Name	DNS name	State	VPC ID	Availability Zones	Type	Date create
☑	accfde03fb4ba4ee6b07...	accfde03fb4ba4ee6b07e89...	–	vpc-0fb205ea438020419	2 Availability Zones	classic	December 1

Load balancer: accfde03fb4ba4ee6b07e894abf7e5f7

Details | Listeners | Network mapping | Security | Health checks | Target instances | Monitoring | Attributes | Tags

Details

Load balancer type Classic	**Status** 2 of 2 instances in service	**VPC** vpc-0fb205ea438020419	**Date created** December 11, 2024, 12:16 (UTC+05:30)
Scheme Internet-facing	**Hosted zone** ZP97RAFLXTNZK	**Availability Zones** subnet-056fd47def3bab0f1 ap-south-1b (aps1-az3)	



Successfully attached Access-for-EC2 to instance i-084dc1b5212df6a4c

Instances (1/3) Info

Last updated less than a minute ago

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status
☑	App-server-01	i-084dc1b5212df6a4c	Running	t2.micro	2/2 checks passed	View alarms +
☐	sachin-cluster-01-ng-6557758e-Node	i-0e780a985e0063a07	Running	t2.micro	2/2 checks passed	View alarms +
☐	sachin-cluster-01-ng-6557758e-Node	i-0eb6d8e28cb0fe850	Running	t2.micro	2/2 checks passed	View alarms +

i-084dc1b5212df6a4c (App-server-01)

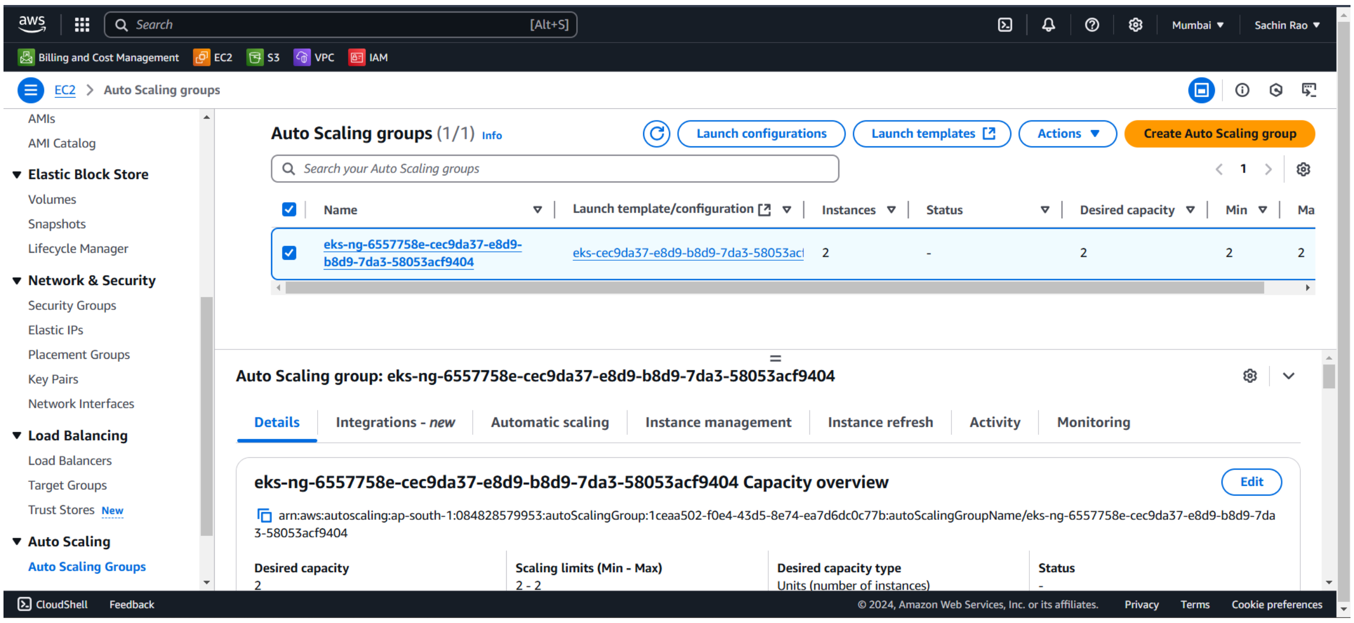
Details | Status and alarms | Monitoring | Security | Networking | Storage | Tags

▼ **Instance summary** Info

Instance ID i-084dc1b5212df6a4c	**Public IPv4 address** 13.234.240.182 | open address	**Private IPv4 addresses** 172.31.33.90
IPv6 address –	**Instance state** Running	**Public IPv4 DNS**

Step 10:-**Output** -- Select Auto Scaling Group

- Access the selected Auto Scaling group and review its detailed information, including configurations and instance statuses.
- Locate and copy the DNS name associated with the Auto Scaling group. Open a web browser, paste the copied DNS name into the address bar, and press Enter to access the link.



Auto Scaling groups (1/1) Info

Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Ma
eks-ng-6557758e-cec9da37-e8d9-b8d9-7da3-58053acf9404	eks-cec9da37-e8d9-b8d9-7da3-58053ac	2	-	2	2	2

Auto Scaling group: eks-ng-6557758e-cec9da37-e8d9-b8d9-7da3-58053acf9404

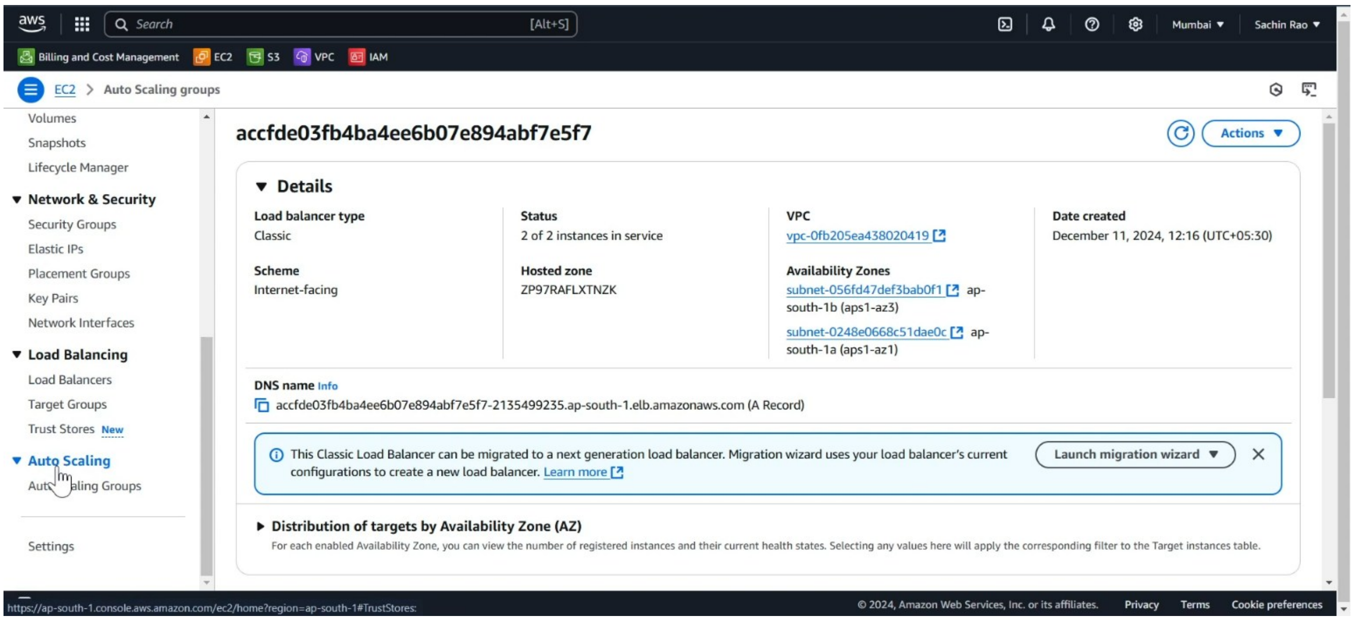
Details | Integrations - *new* | Automatic scaling | Instance management | Instance refresh | Activity | Monitoring

eks-ng-6557758e-cec9da37-e8d9-b8d9-7da3-58053acf9404 Capacity overview

Edit

arn:aws:autoscaling:ap-south-1:084828579953:autoScalingGroup:1ceaa502-f0e4-43d5-8e74-ea7d6dc0c77b:autoScalingGroupName/eks-ng-6557758e-cec9da37-e8d9-b8d9-7da3-58053acf9404

Desired capacity	Scaling limits (Min - Max)	Desired capacity type	Status
2	2 - 2	Units (number of instances)	-



accfde03fb4ba4ee6b07e894abf7e5f7

Details

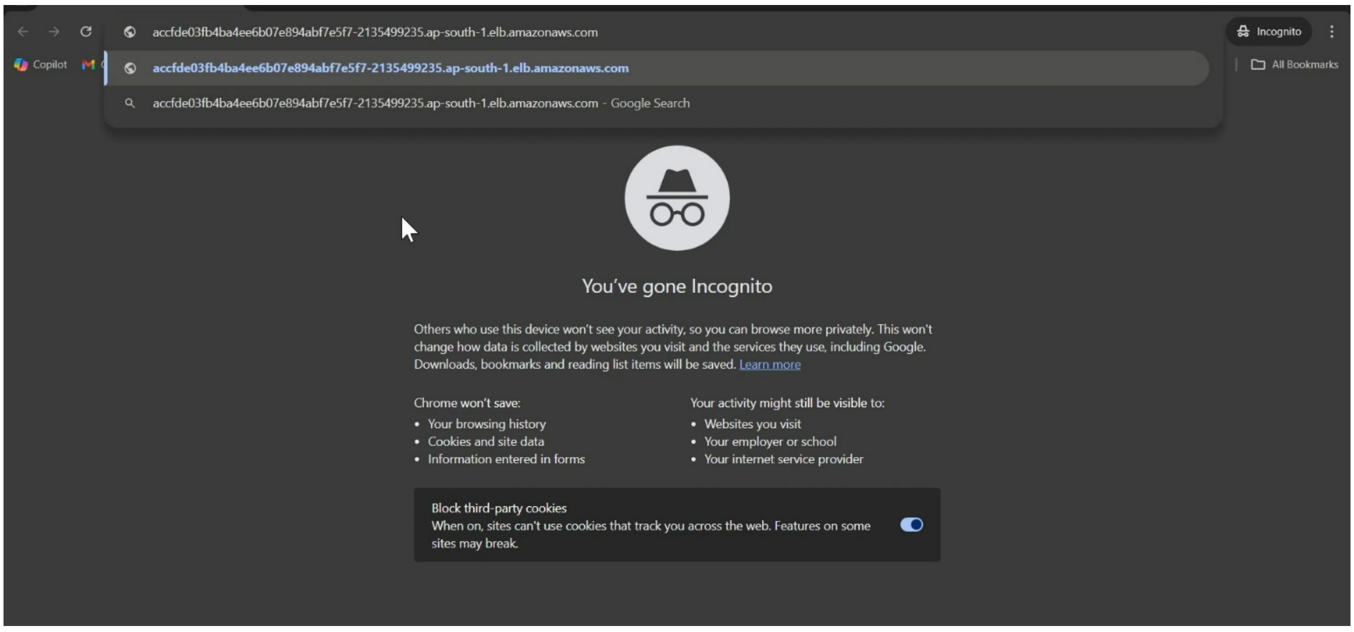
Load balancer type	Status	VPC	Date created
Classic	2 of 2 instances in service	vpc-0fb205ea438020419	December 11, 2024, 12:16 (UTC+05:30)
Scheme	**Hosted zone**	**Availability Zones**	
Internet-facing	ZP97RAFLXTNZK	subnet-056fd47def3bab0f1 ap-south-1b (aps1-az3) subnet-0248e0668c51dae0c ap-south-1a (aps1-az1)	

DNS name Info
accfde03fb4ba4ee6b07e894abf7e5f7-2135499235.ap-south-1.elb.amazonaws.com (A Record)

This Classic Load Balancer can be migrated to a next generation load balancer. Migration wizard uses your load balancer's current configurations to create a new load balancer. Learn more

Launch migration wizard

Distribution of targets by Availability Zone (AZ)
For each enabled Availability Zone, you can view the number of registered instances and their current health states. Selecting any values here will apply the corresponding filter to the Target instances table.



accfde03fb4ba4ee6b07e894abf7e5f7-2135499235.ap-south-1.elb.amazonaws.com
accfde03fb4ba4ee6b07e894abf7e5f7-2135499235.ap-south-1.elb.amazonaws.com
accfde03fb4ba4ee6b07e894abf7e5f7-2135499235.ap-south-1.elb.amazonaws.com - Google Search

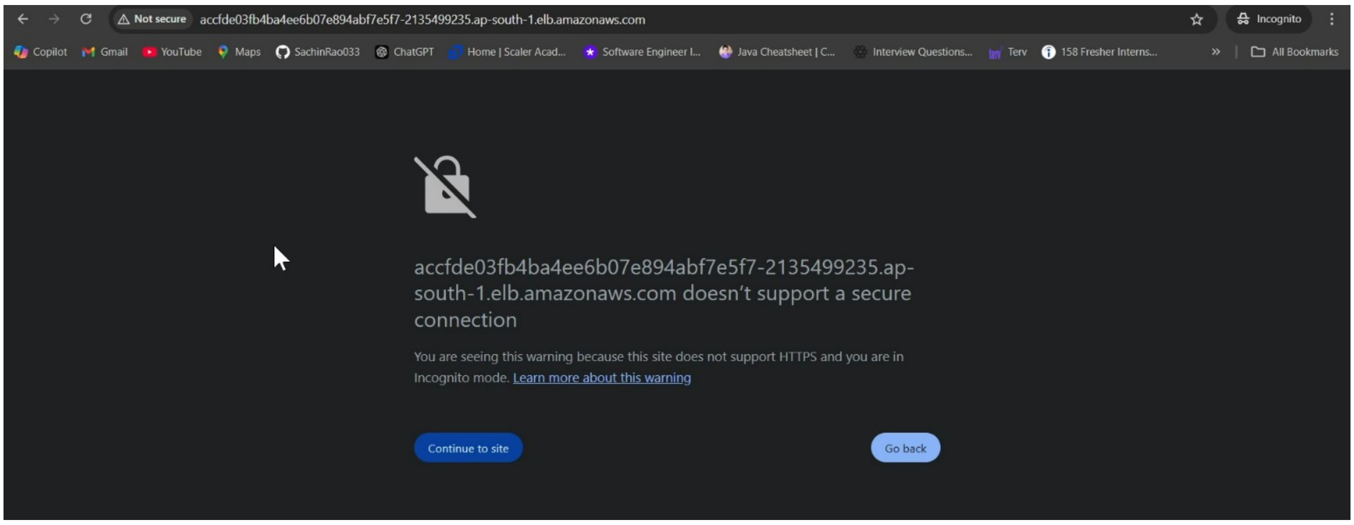
You've gone Incognito

Others who use this device won't see your activity, so you can browse more privately. This won't change how data is collected by websites you visit and the services they use, including Google. Downloads, bookmarks and reading list items will be saved. Learn more

Chrome won't save:
- Your browsing history
- Cookies and site data
- Information entered in forms

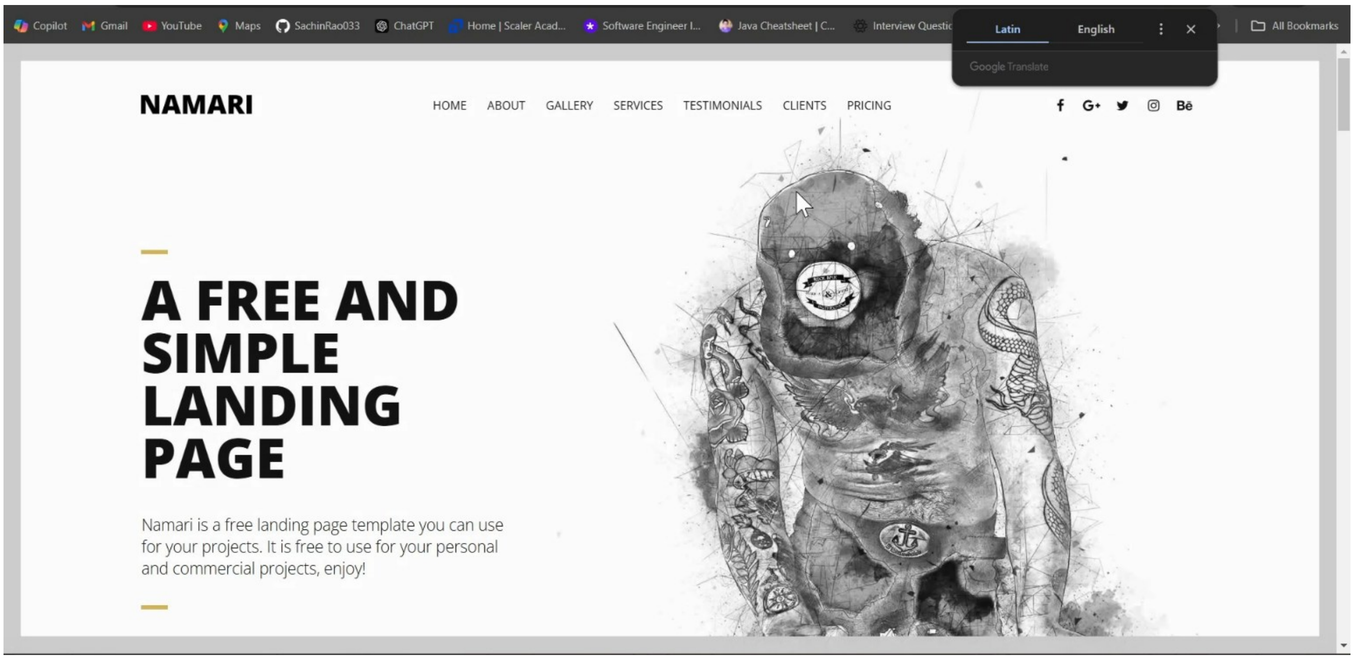
Your activity might still be visible to:
- Websites you visit
- Your employer or school
- Your internet service provider

Block third-party cookies
When on, sites can't use cookies that track you across the web. Features on some sites may break.



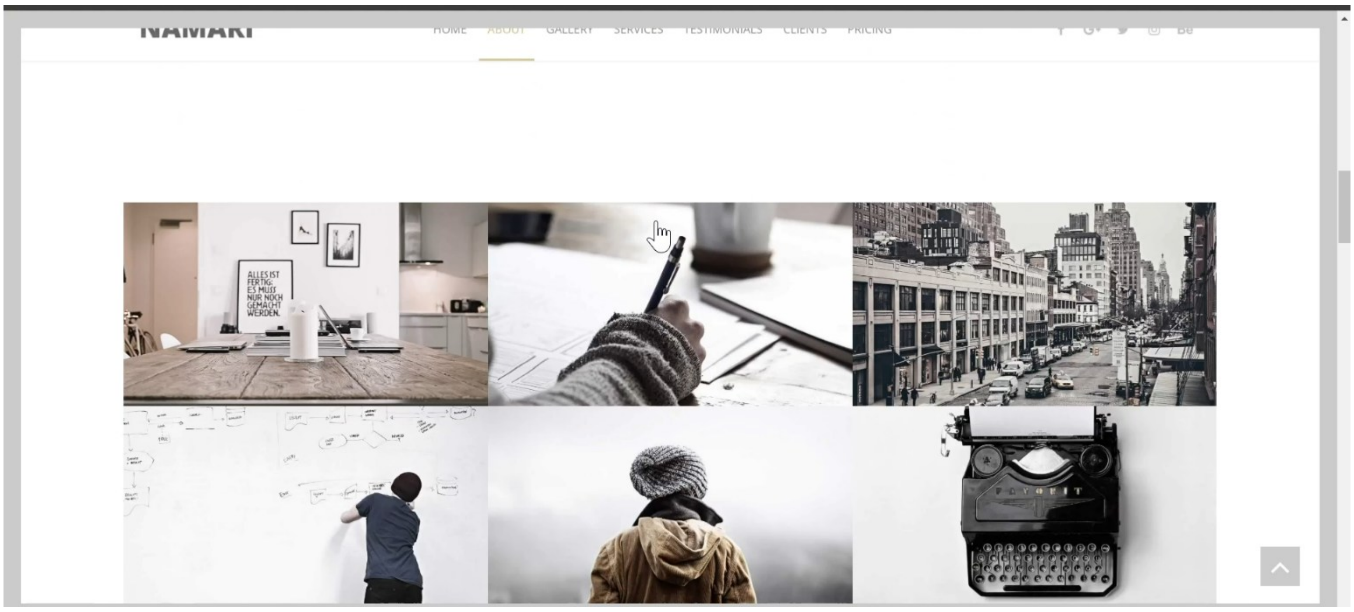
accfde03fb4ba4ee6b07e894abf7e5f7-2135499235.ap-south-1.elb.amazonaws.com doesn't support a secure connection

You are seeing this warning because this site does not support HTTPS and you are in Incognito mode. Learn more about this warning



A FREE AND SIMPLE LANDING PAGE

Namari is a free landing page template you can use for your projects. It is free to use for your personal and commercial projects, enjoy!



Conclusion:-

Deploying a web application on AWS using Kubernetes (EKS) offers a scalable and efficient solution. By leveraging EC2, EKS, and Kubernetes tools like kubectl and eksctl, the deployment process is streamlined. Integrating Auto Scaling ensures high availability and performance, while Kubernetes simplifies container management. This approach enhances flexibility and scalability, making it an ideal choice for modern cloud environments. Overall, AWS and Kubernetes together provide a robust platform for deploying and managing applications in production.