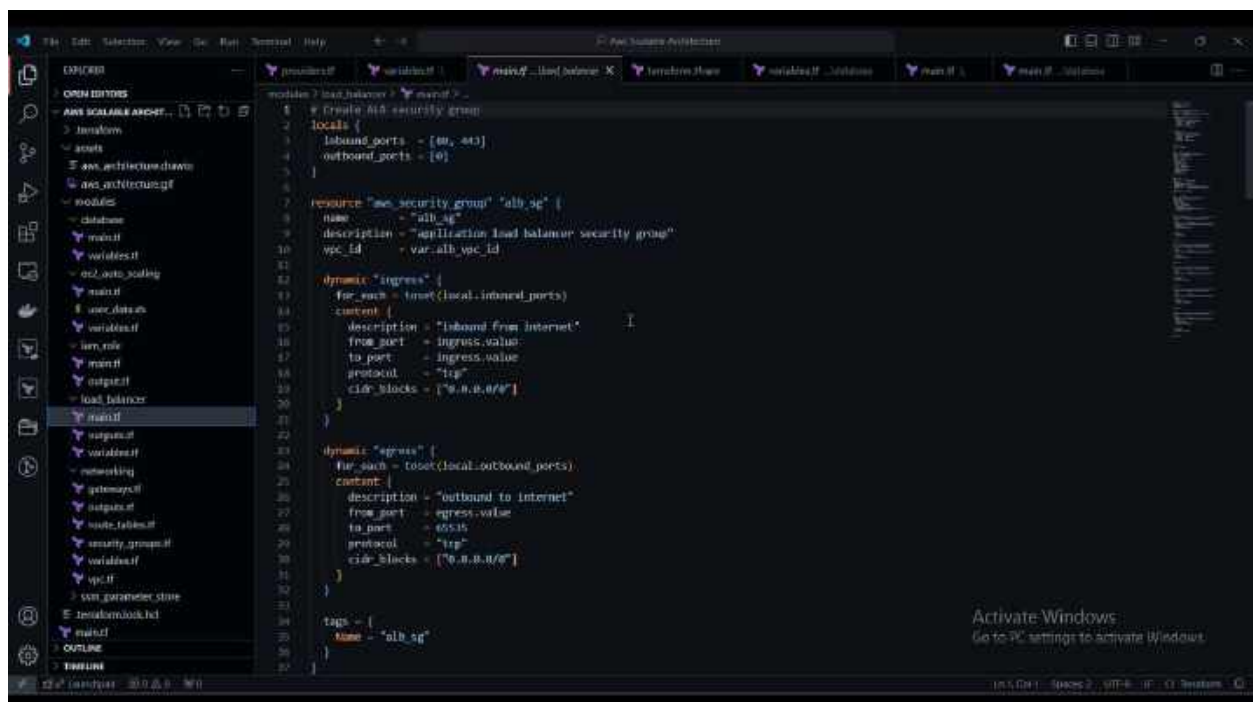
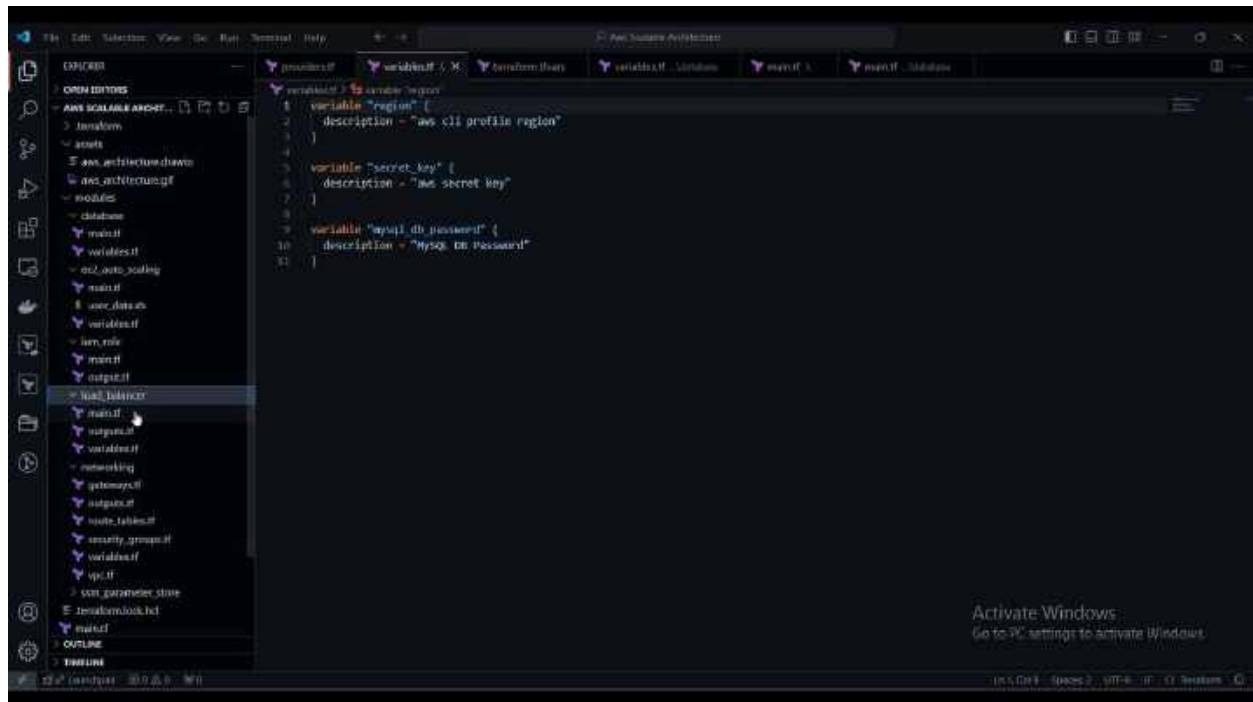


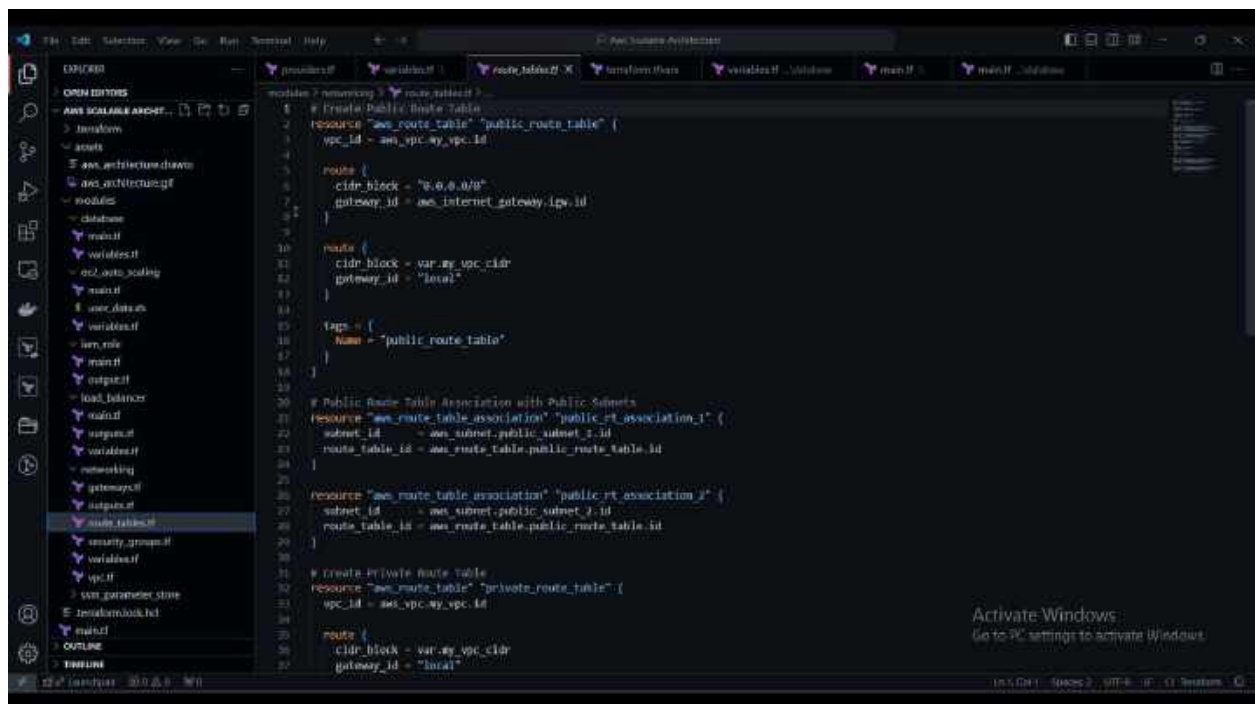
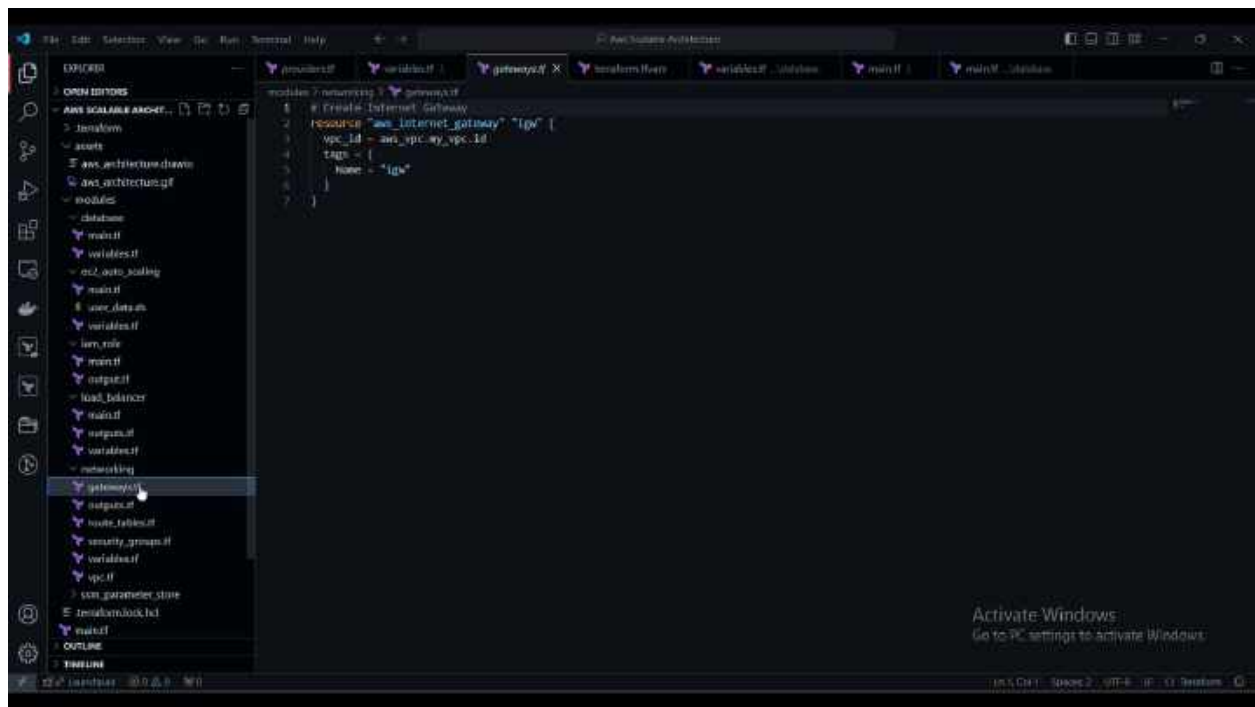
DevOps PROJECT 1

1) Create AWS using Terraform

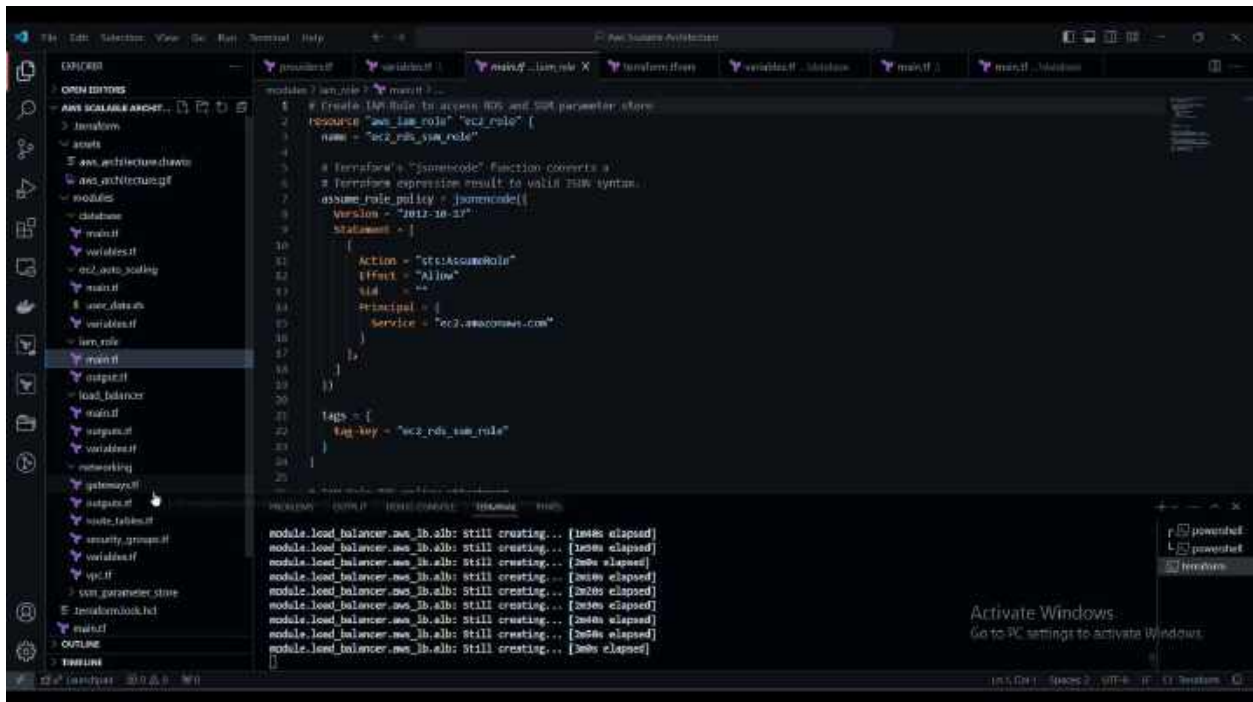
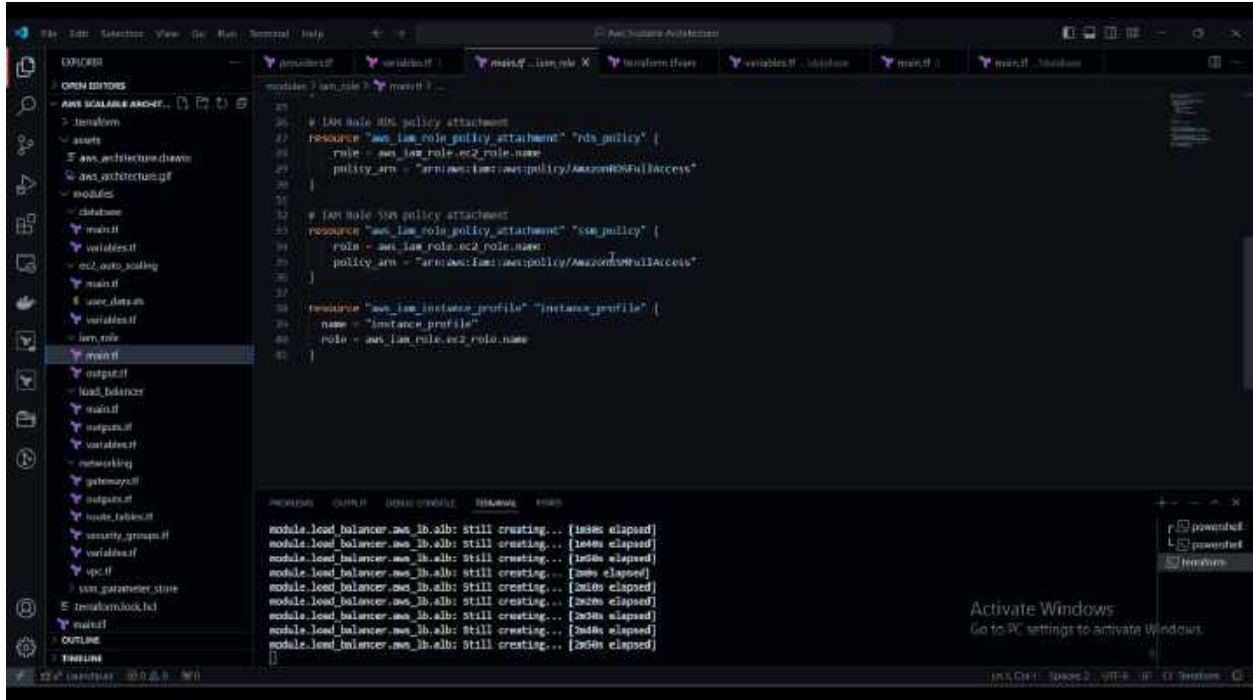
➤ AWS Scalable Architecture

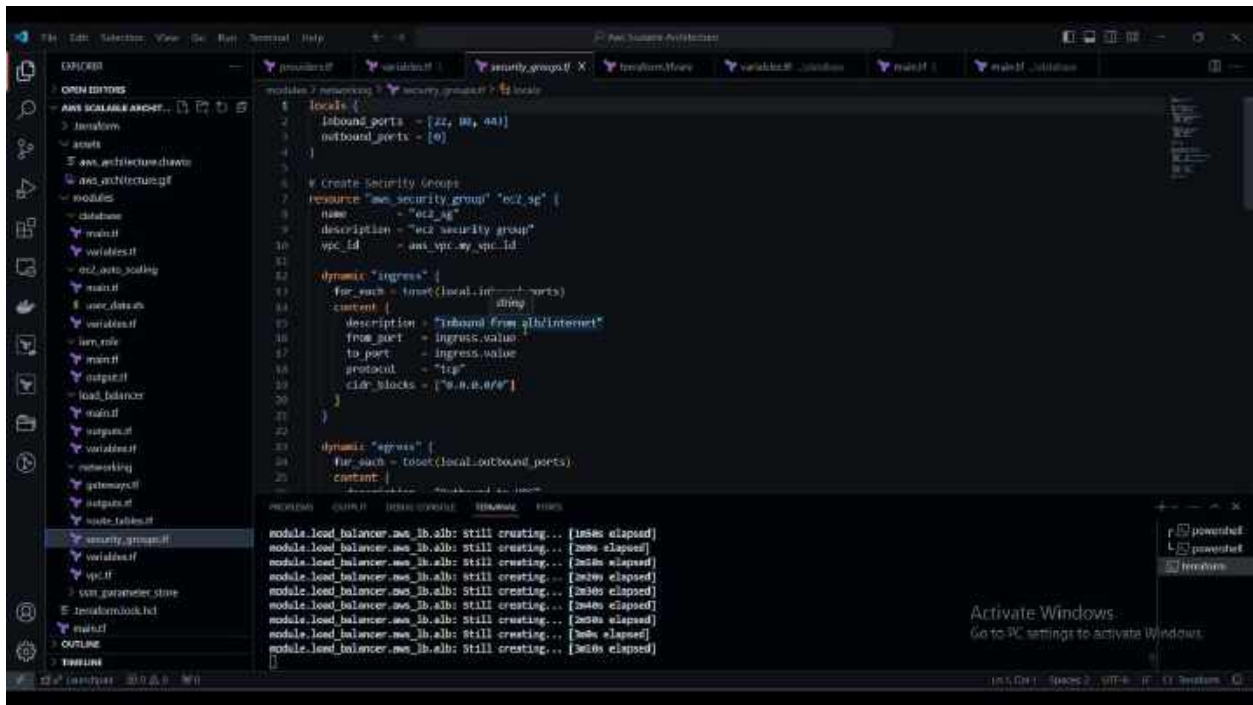


Name-Dibyajyoti Mishra

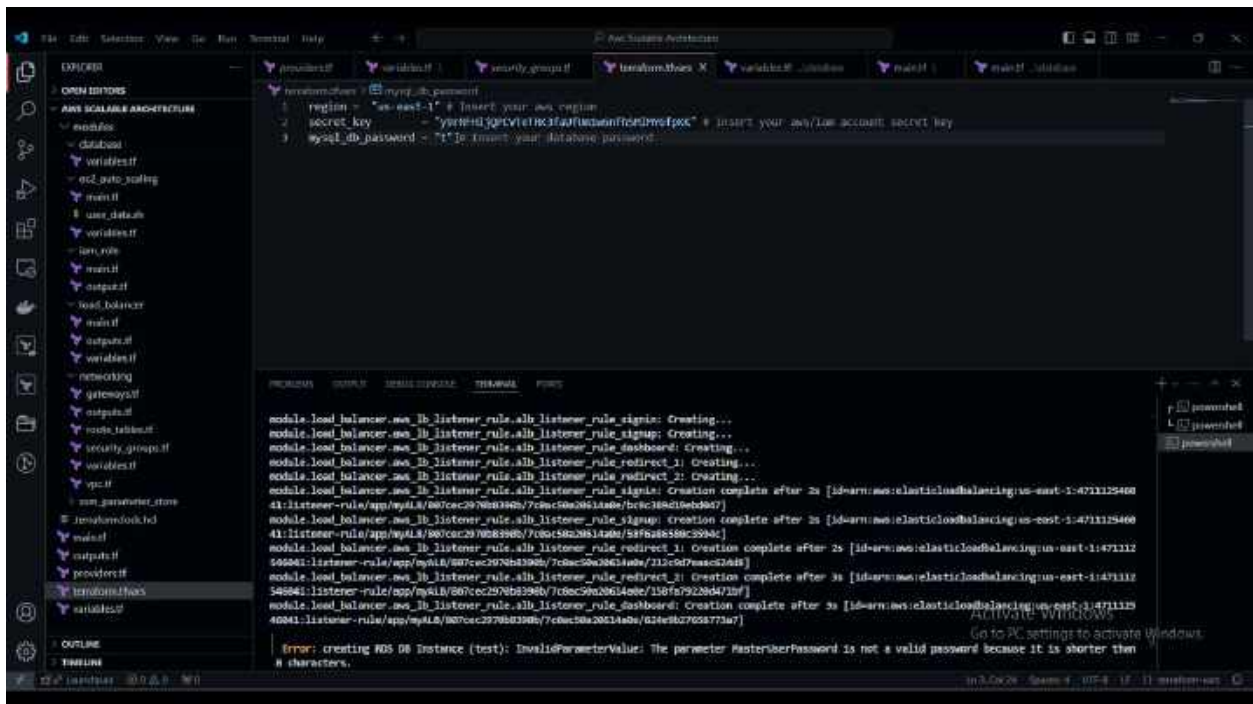


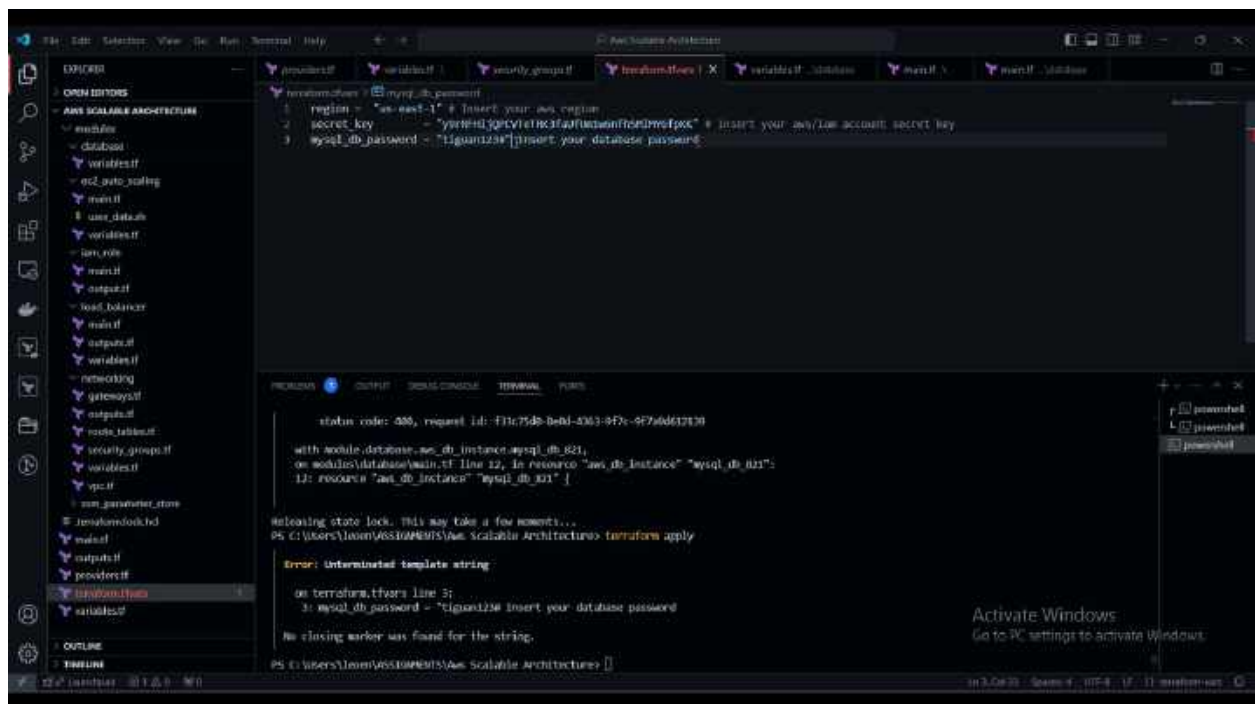
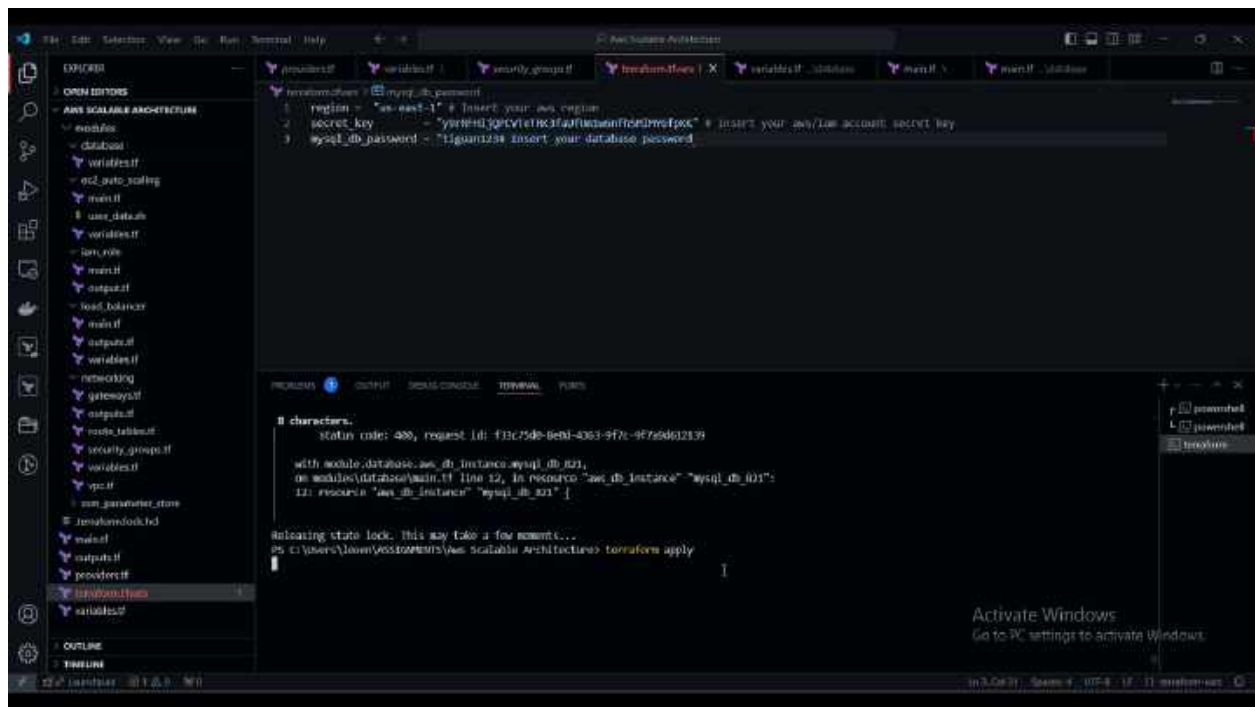
➤ Create IAM Roles & Policy

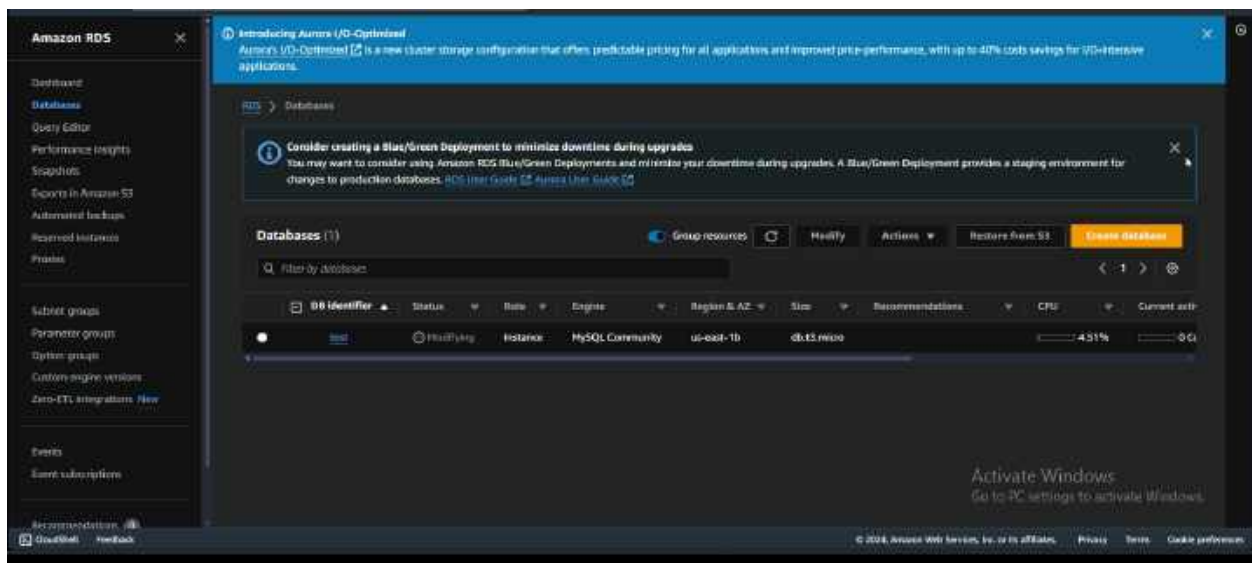
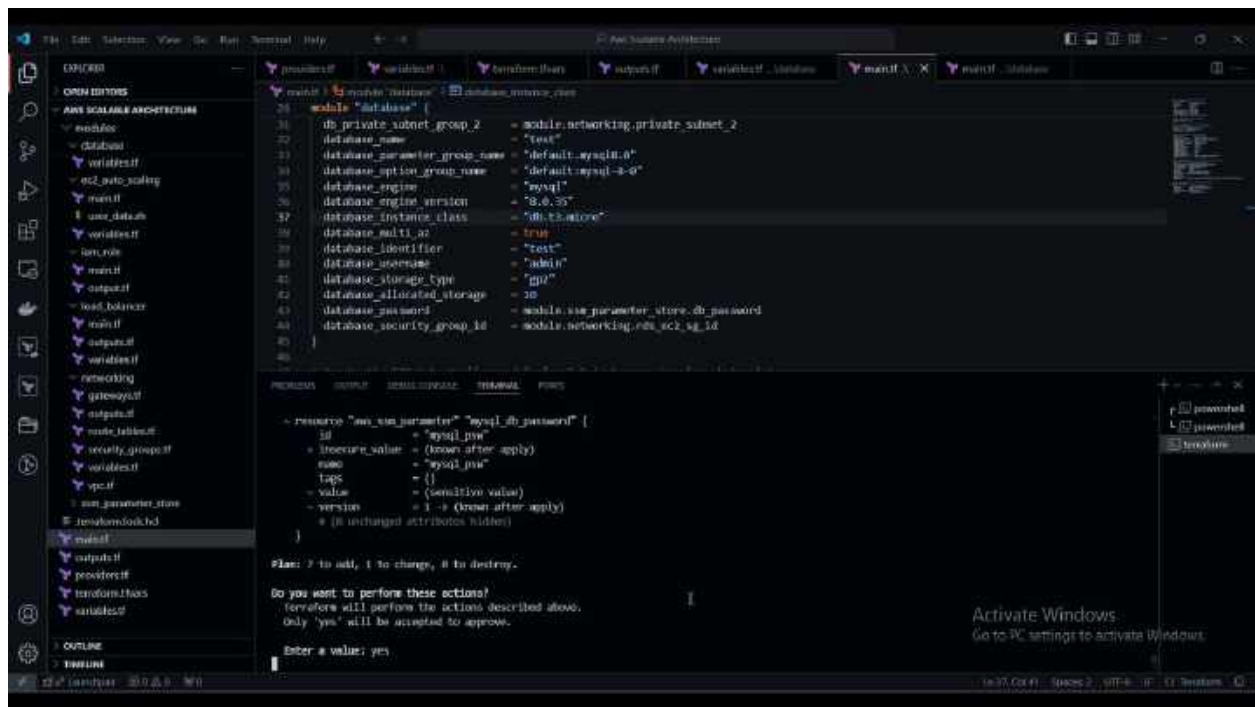




➤ Create mysql database







➤ *Initiate and deployment of VPC*

Name-Dibyajyoti Mishra

VPC dashboard X

EC2 Global View

Filter by VPC

Virtual private cloud

- My VPCs
- Subnets
- Route tables
- Internet gateways
- Egress-only internet gateways
- Carrier gateways
- DHCP option sets
- Elastic IP
- Managed prefix lists
- Endpoints
- Endpoint services
- NAT gateways
- Pending connections

Security

- Network ACLs

Your VPCs (1/2) [info](#)

Search

Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR	DHCP option set	Main
my_vpc	vpc-02164ed9e2499b7f4	Available	10.0.0.0/16	-	dhcp-0b1c6f7f552913...	info
-	vpc-0b4d0a48e6c88b1d5	Available	172.31.0.0/16	-	dhcp-0b1c6f7f552913...	info

vpc-02164ed9e2499b7f4 / my_vpc

[Details](#) [Resource map](#) [CIDRs](#) [Flow logs](#) [Tags](#) [Integrations](#)

Details

VPC ID vpc-02164ed9e2499b7f4	State Available	DNS hostnames Disabled	DNS resolution Enabled
Tenancy Default	DHCP option set dhcp-0b1c6f7f552913...	Main route table rtb-00eac3b0c0a2d1910	Main network ACL acl-0a34e05816009118
Default VPC No	IPv4 CIDR 10.0.0.0/16	IPv6 pool -	IPv6 CIDR (Network border group) -
Network Address Usage metrics Disabled	Route 53 Resolver DNS Firewall rule groups -	Owner ID 471112546041	

Activate Windows
Go to PC settings to activate Windows.

© 2024 Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

VPC dashboard X

EC2 Global View

Filter by VPC

Virtual private cloud

- My VPCs
- Subnets
- Route tables
- Internet gateways
- Egress-only internet gateways
- Carrier gateways
- DHCP option sets
- Elastic IP
- Managed prefix lists
- Endpoints
- Endpoint services
- NAT gateways
- Pending connections

Security

- Network ACLs

Your VPCs (1/2) [info](#)

Search

Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR	DHCP option set	Main
my_vpc	vpc-02164ed9e2499b7f4	Available	10.0.0.0/16	-	dhcp-0b1c6f7f552913...	info
-	vpc-0b4d0a48e6c88b1d5	Available	172.31.0.0/16	-	dhcp-0b1c6f7f552913...	info

Resource map [info](#)

VPC [Show details](#)
Your AWS virtual network

Subnets (4)
Subnets within this VPC

- us-east-1a
 - public_subnet_3
 - private_subnet_1
- us-east-1b
 - public_subnet_2
 - private_subnet_2

Route tables (5)
Route network traffic to resources

- public_route_table
- private_route_table
- rtb-00eac3b0c0a2d1910

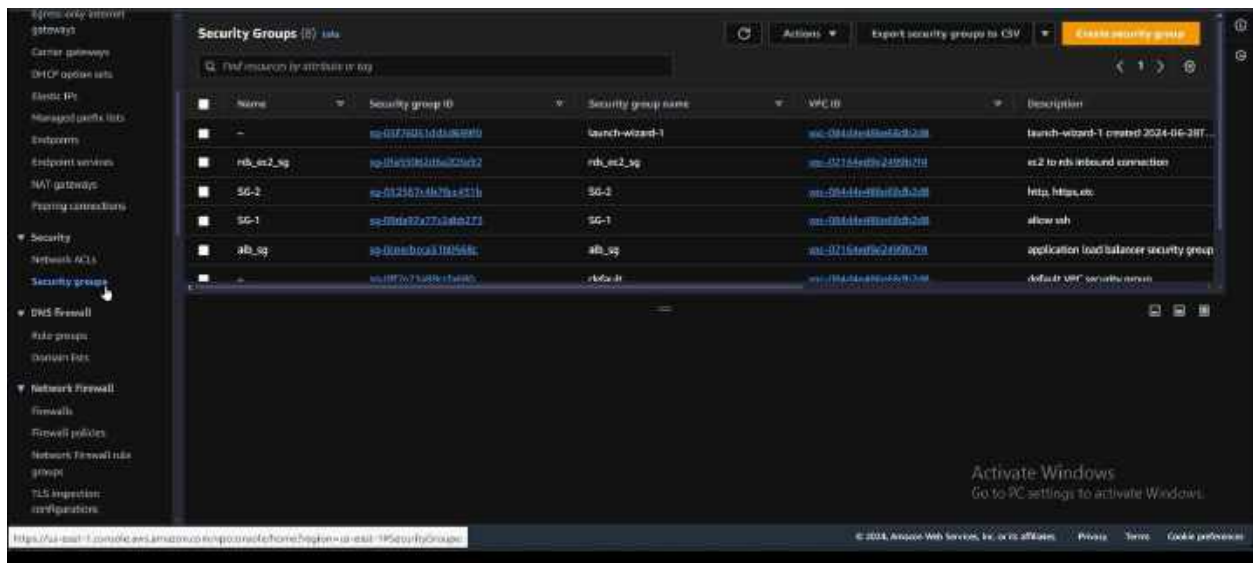
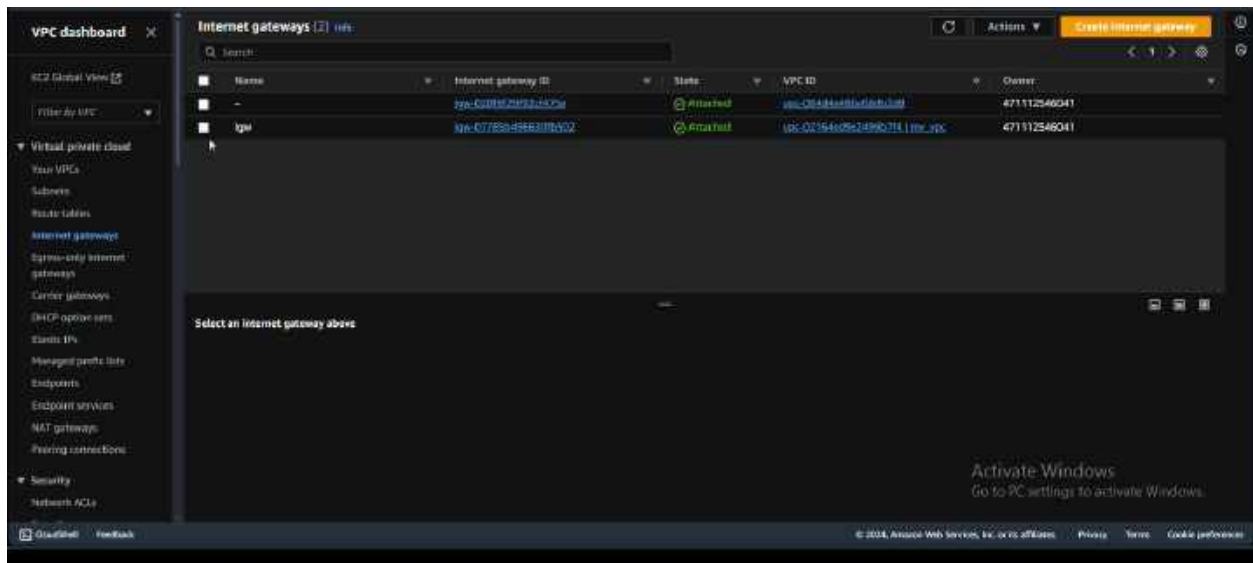
Network connections (1)
Connections to other networks

- vpn

Activate Windows
Go to PC settings to activate Windows.

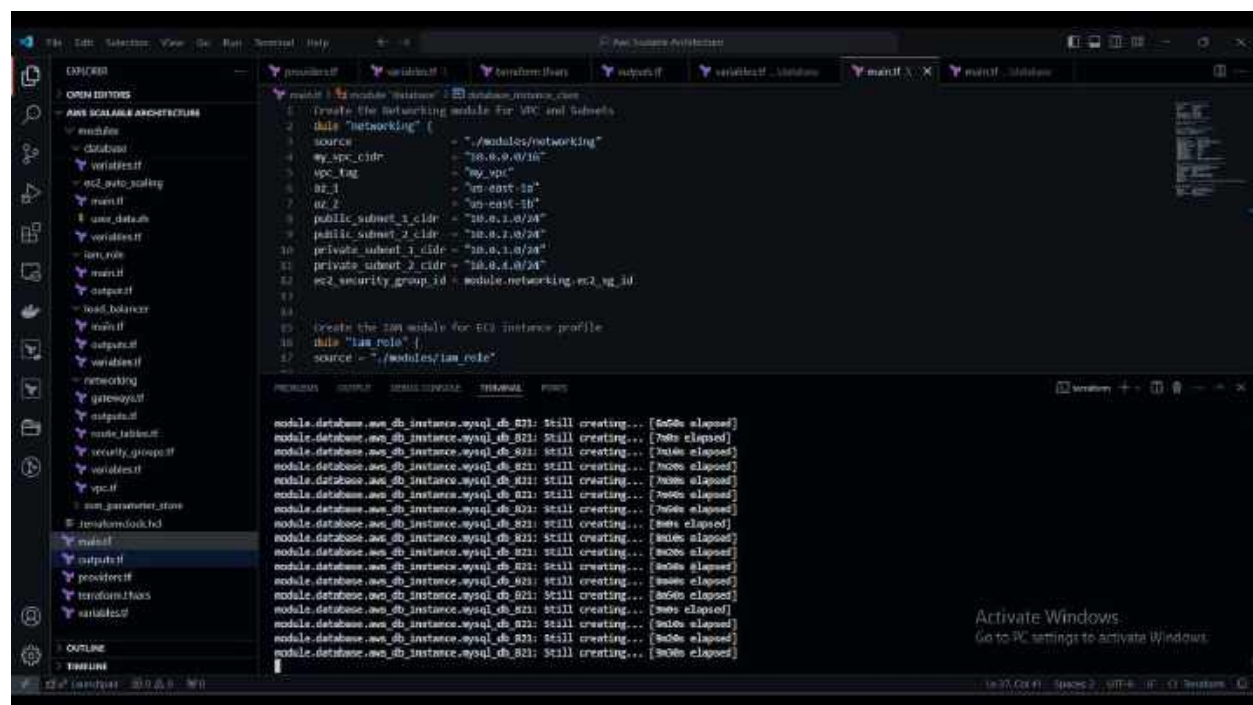
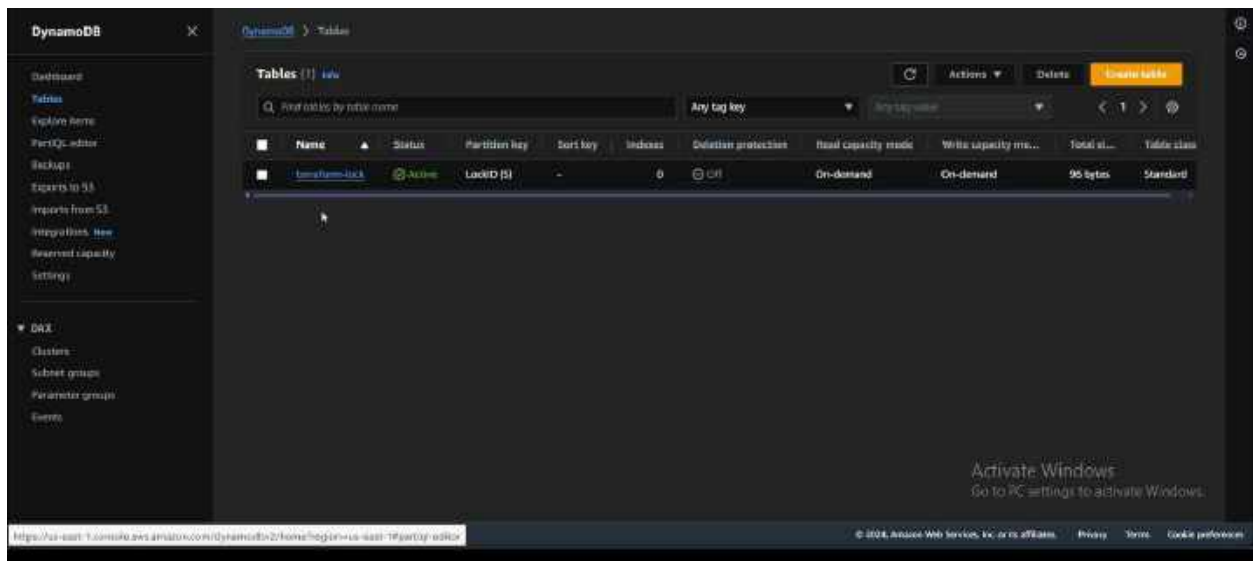
© 2024 Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

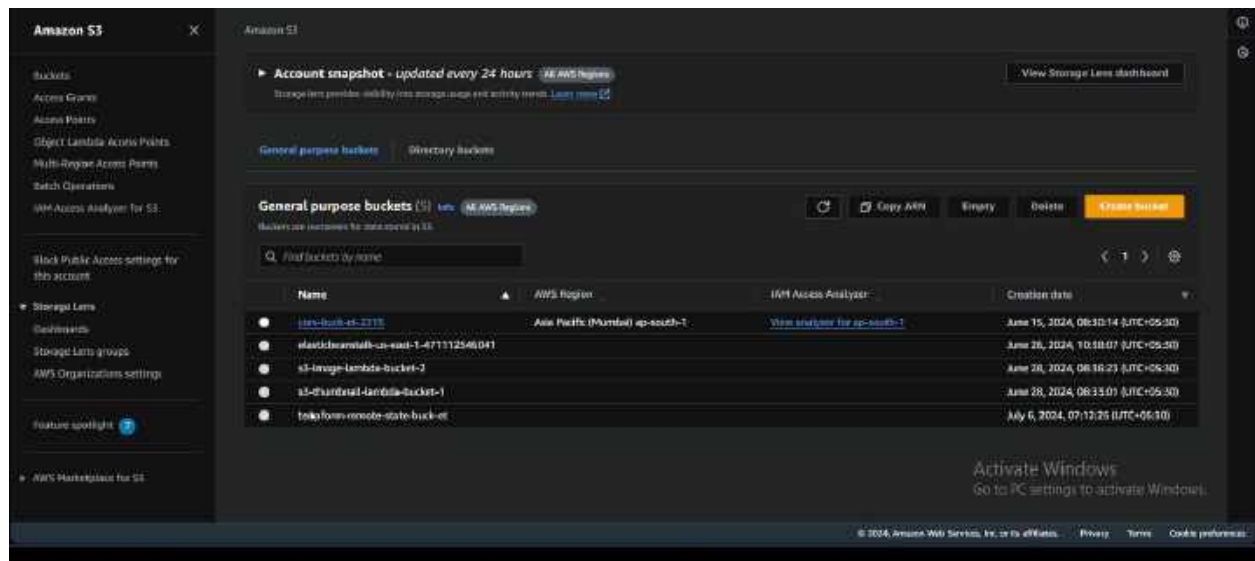
Name-Dibyajyoti Mishra



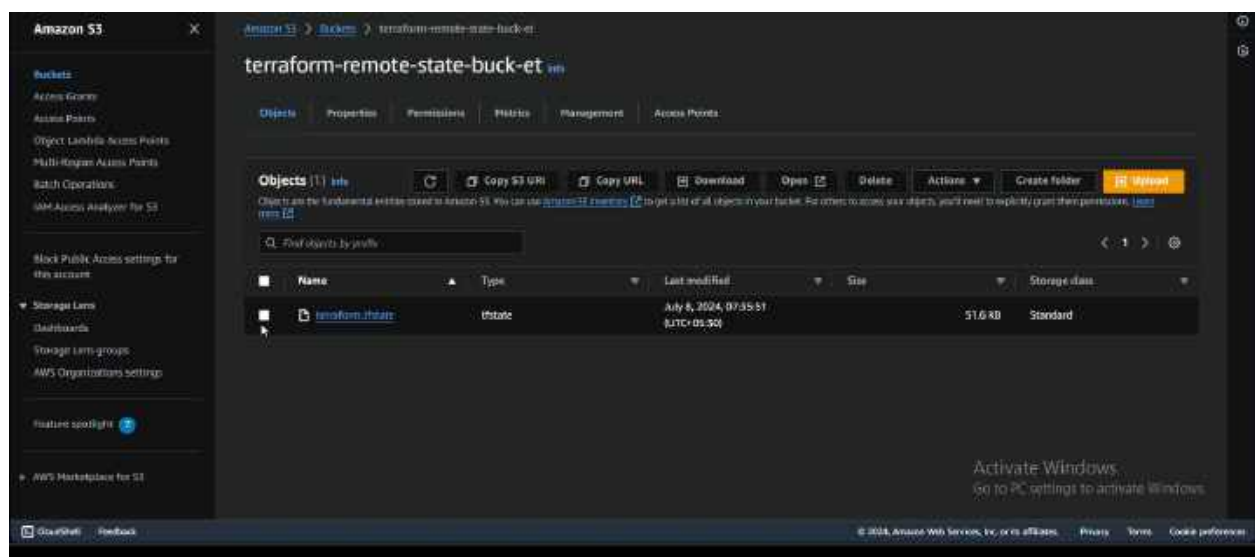
➤ *Initiate Dynamic Database*

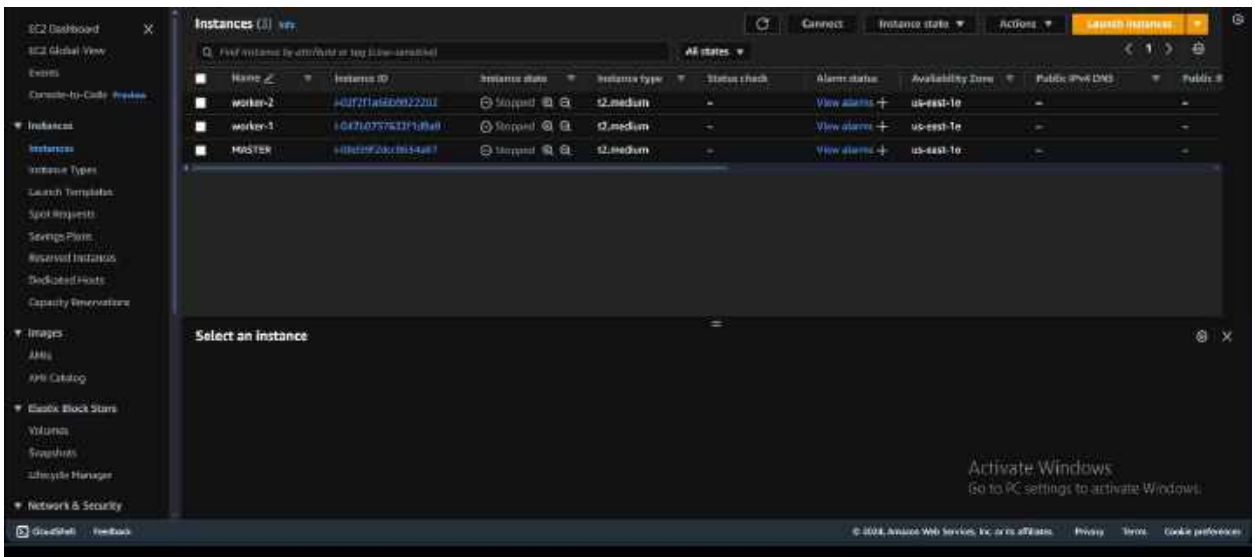
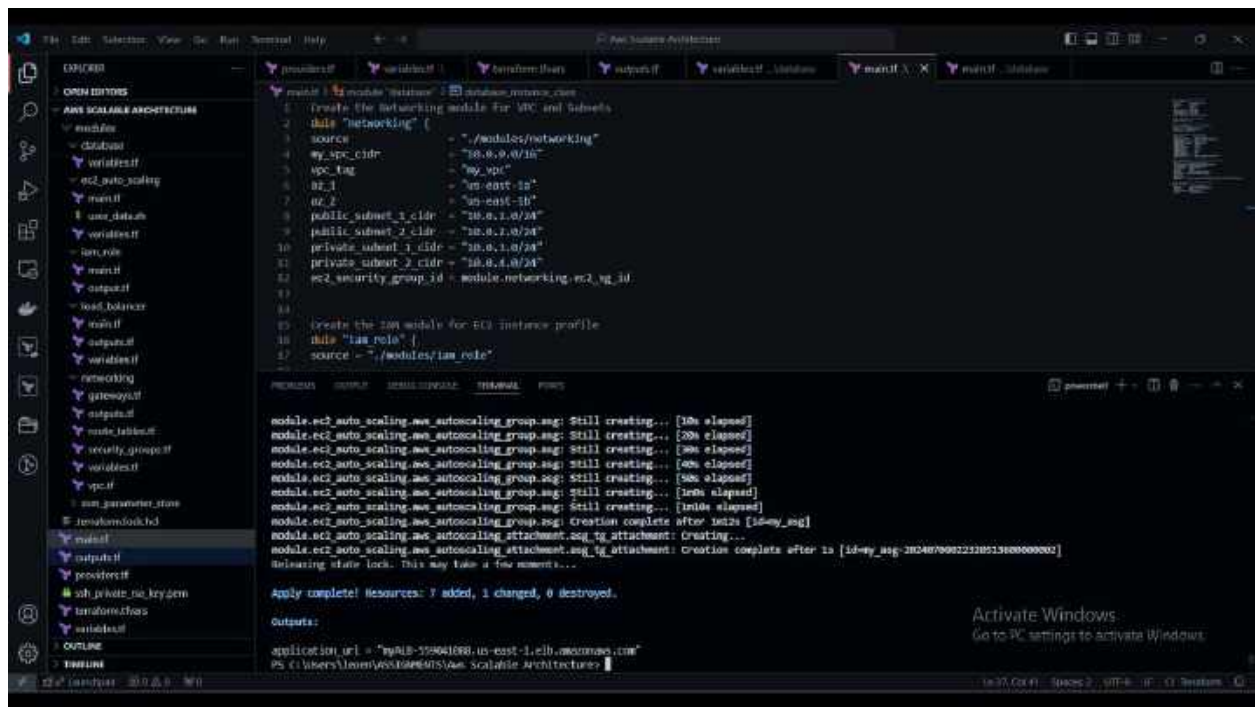
Name-Dibyajyoti Mishra

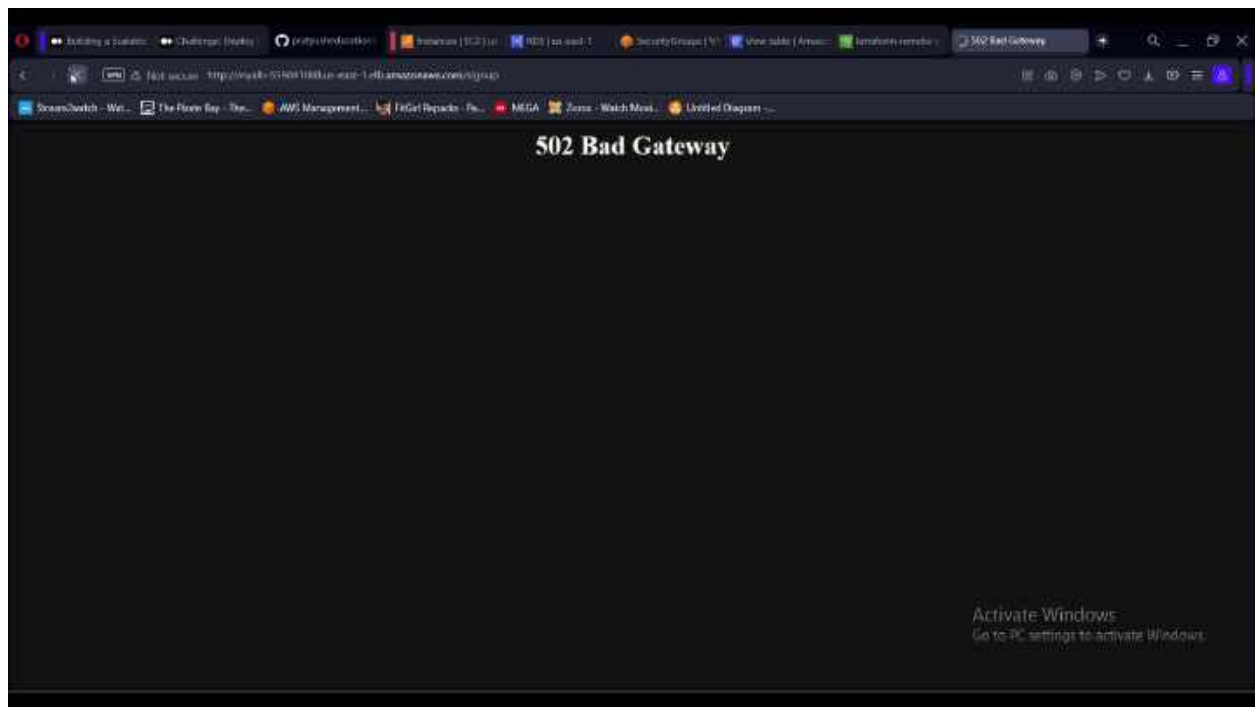




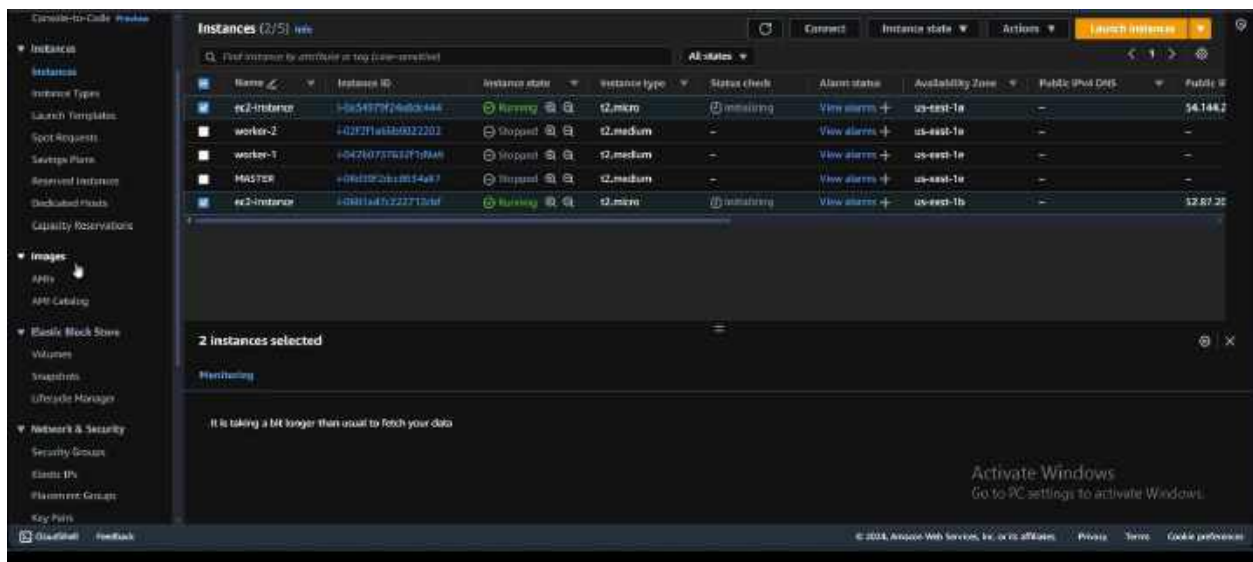
➤ Terraform Bucket Created

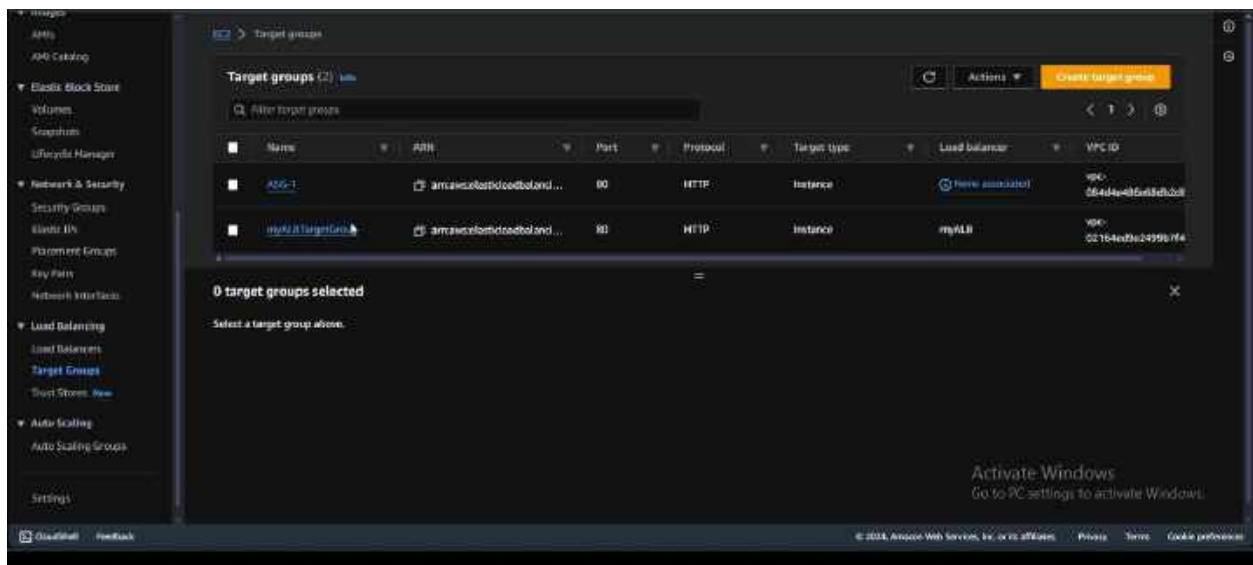
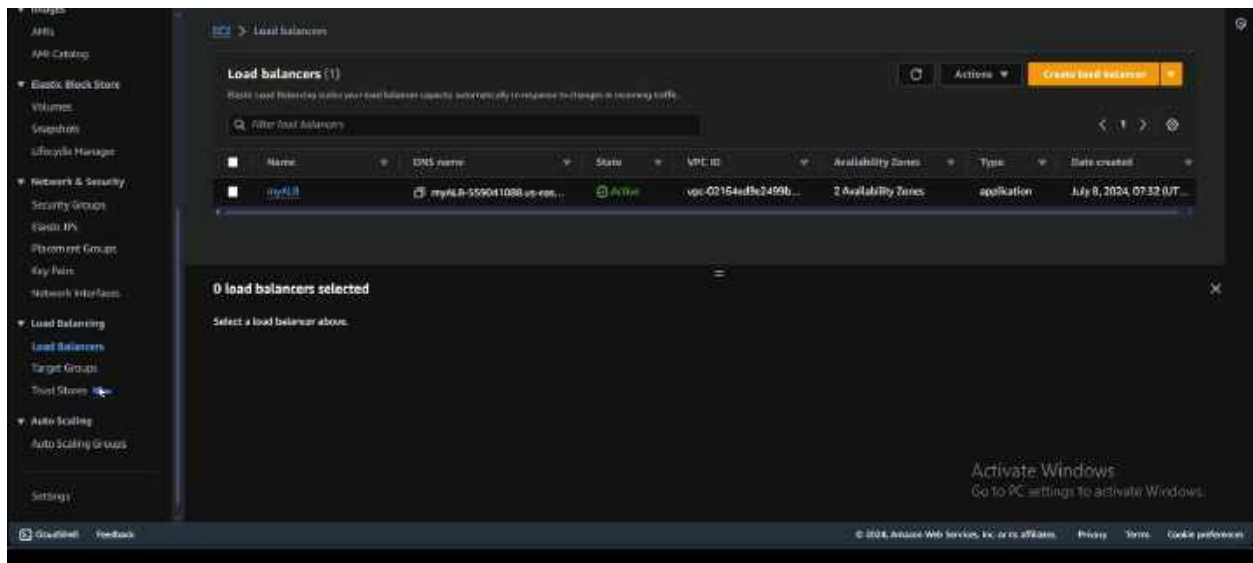




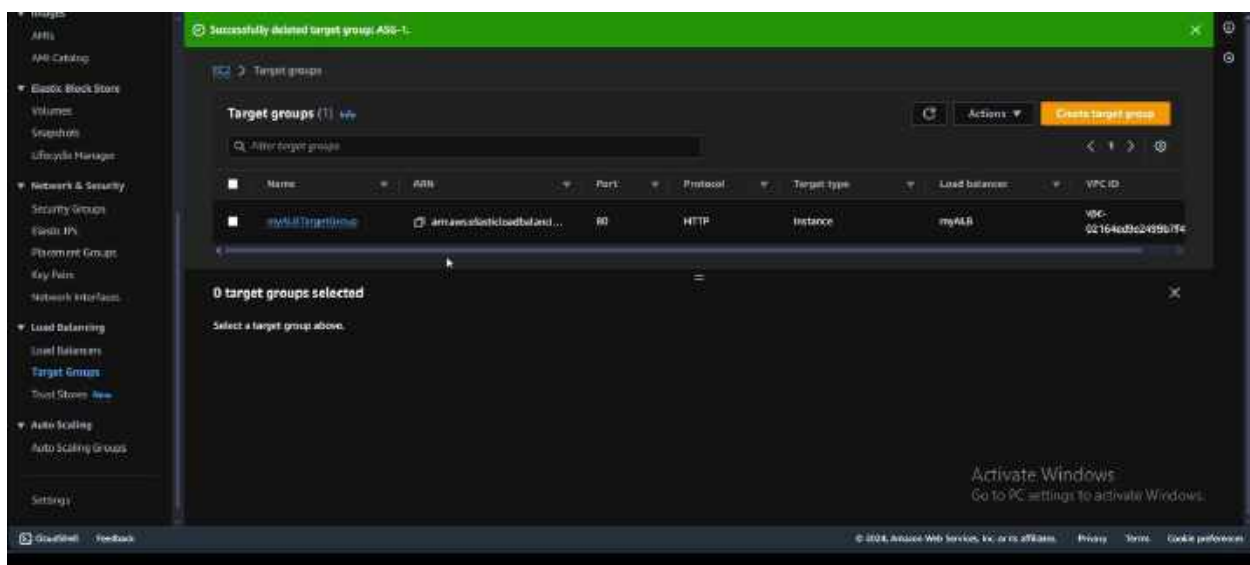
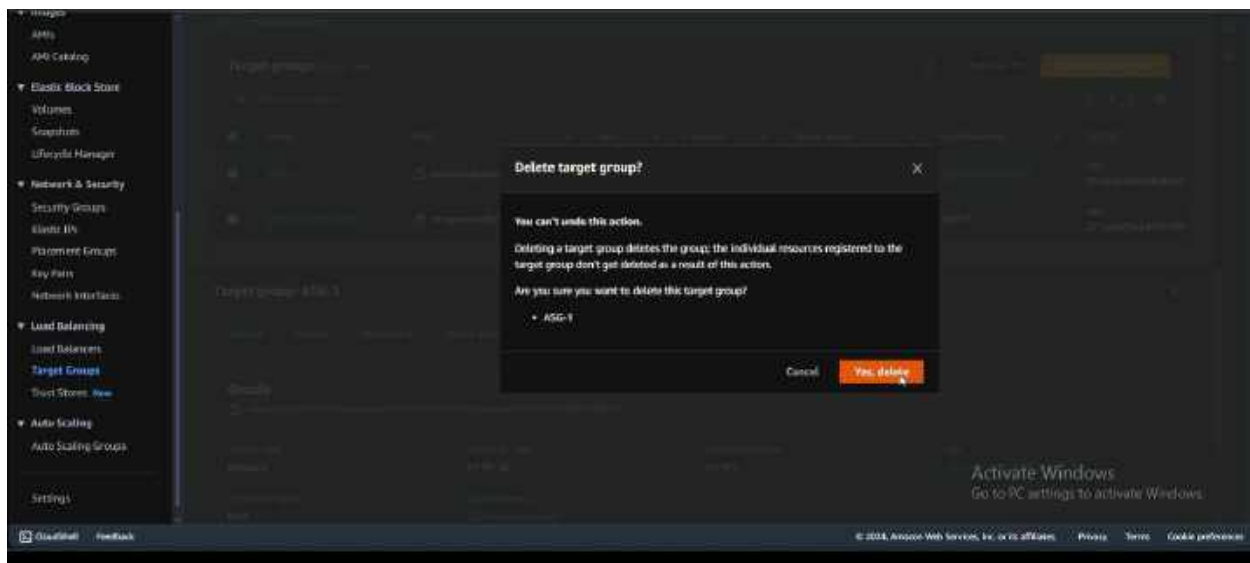


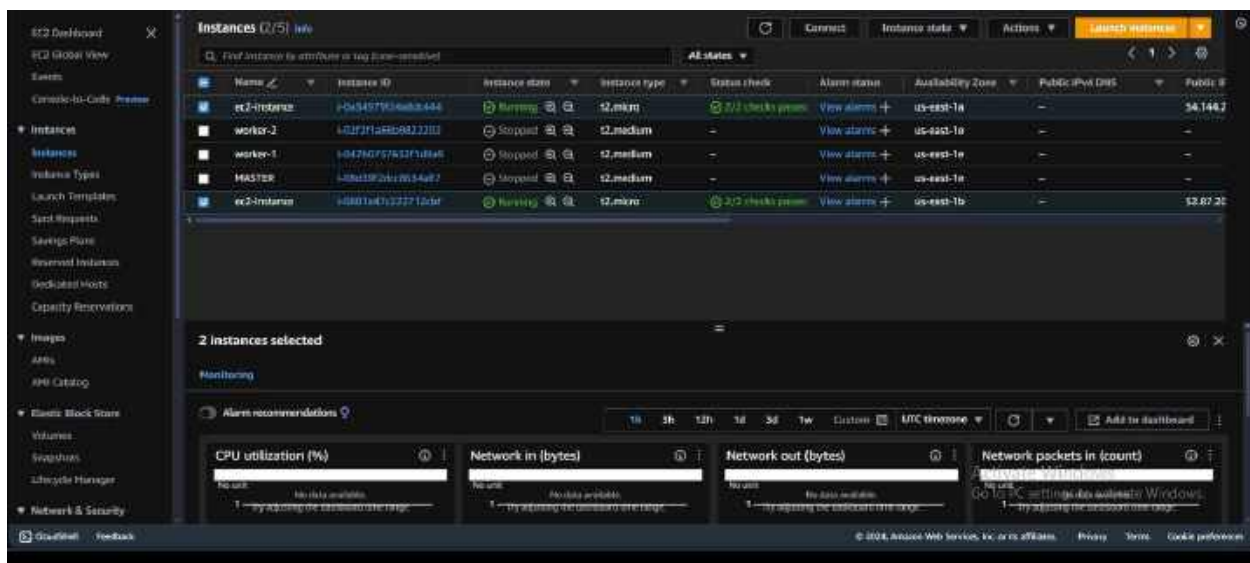
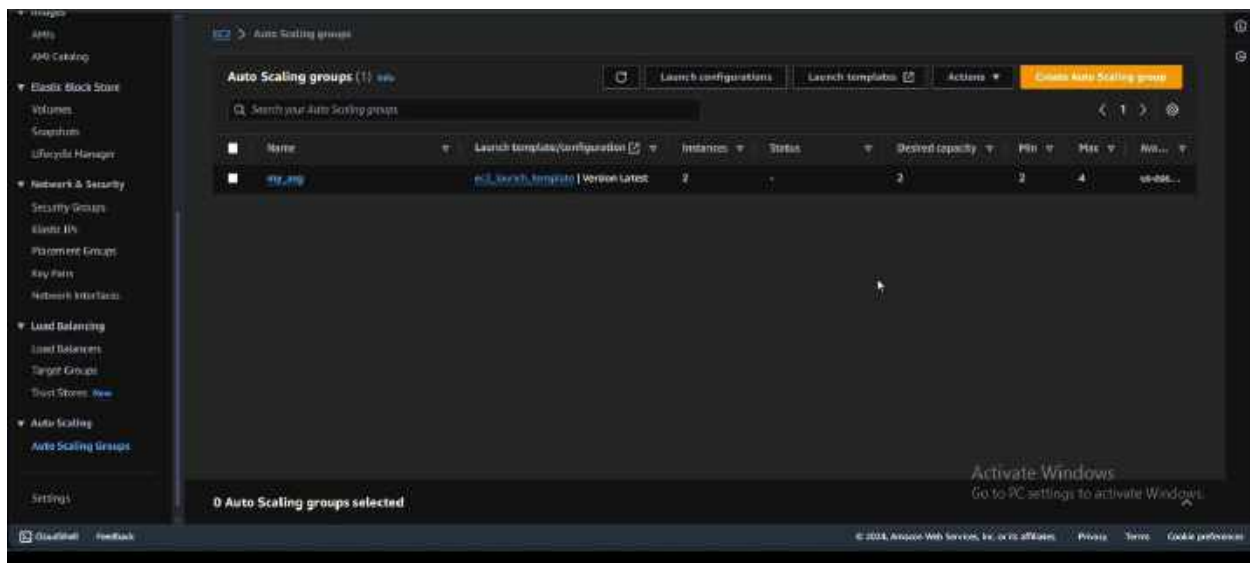
➤ *Intances created in Dynamic Database*



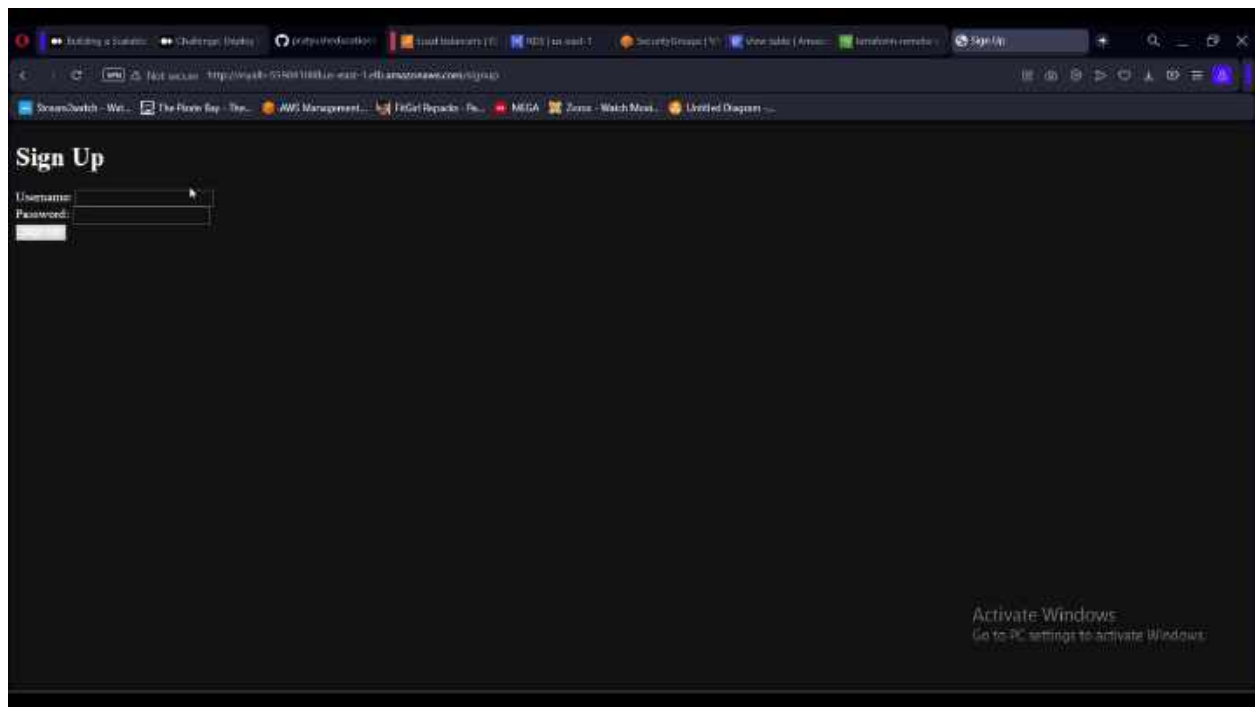


➤ Deleted target group through select ASG-1

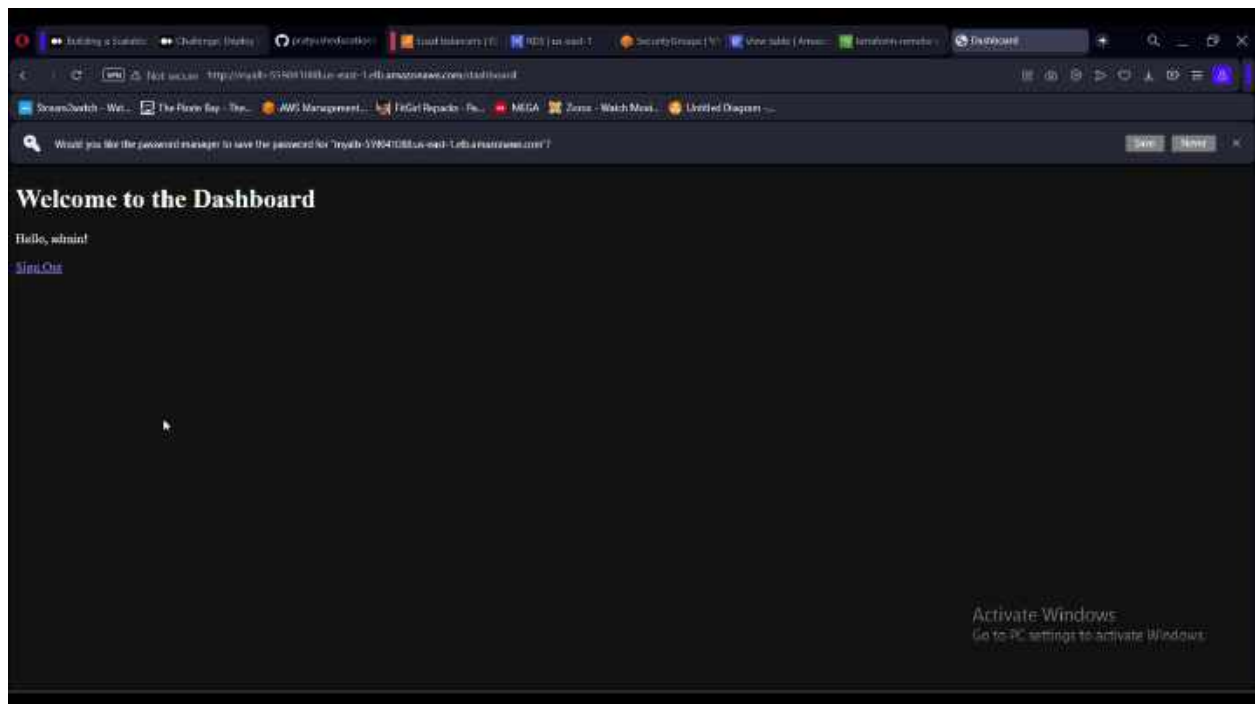




➤ Logging for AWS Cluster using Terraform



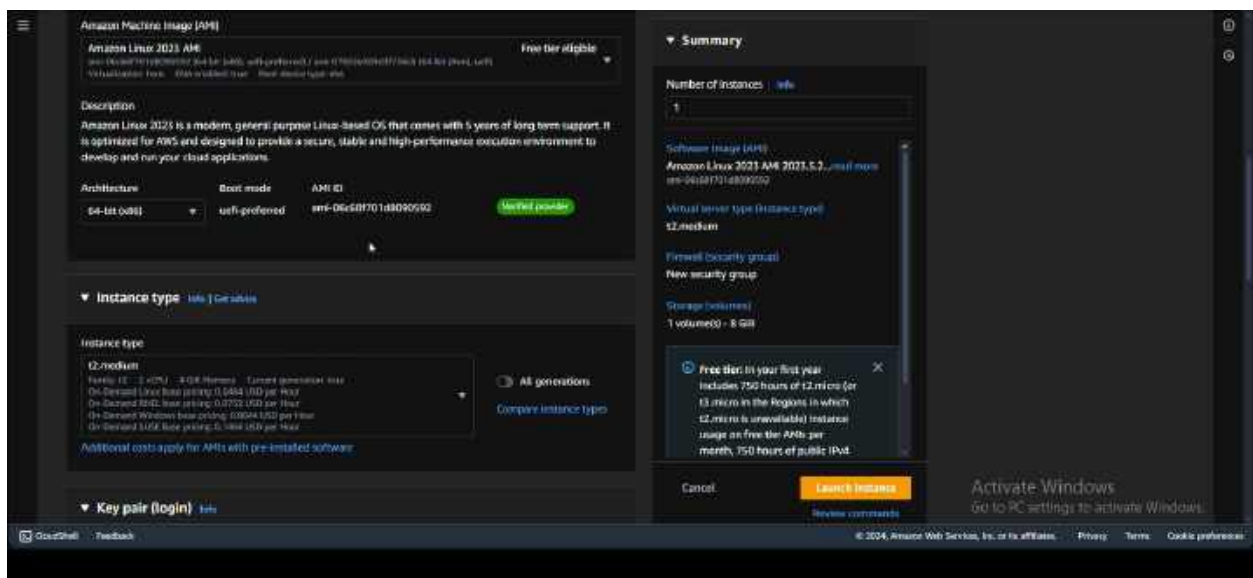
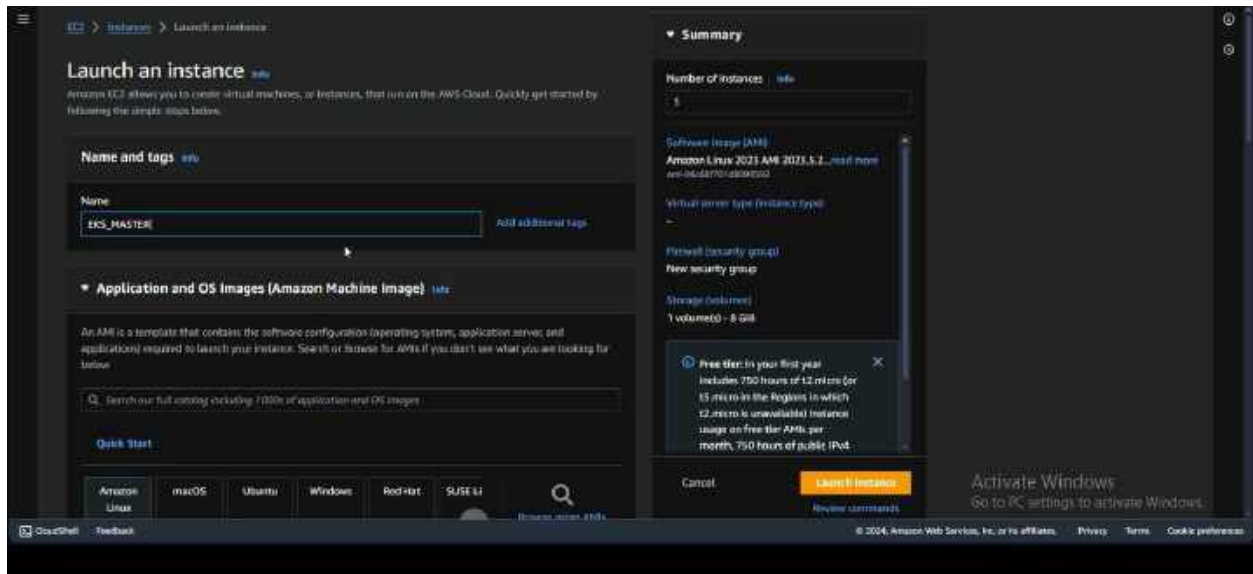
➤ *Finally Created AWS Architecture & Dynamic Database through Terraform*



Name-Dibyajyoti Mishra

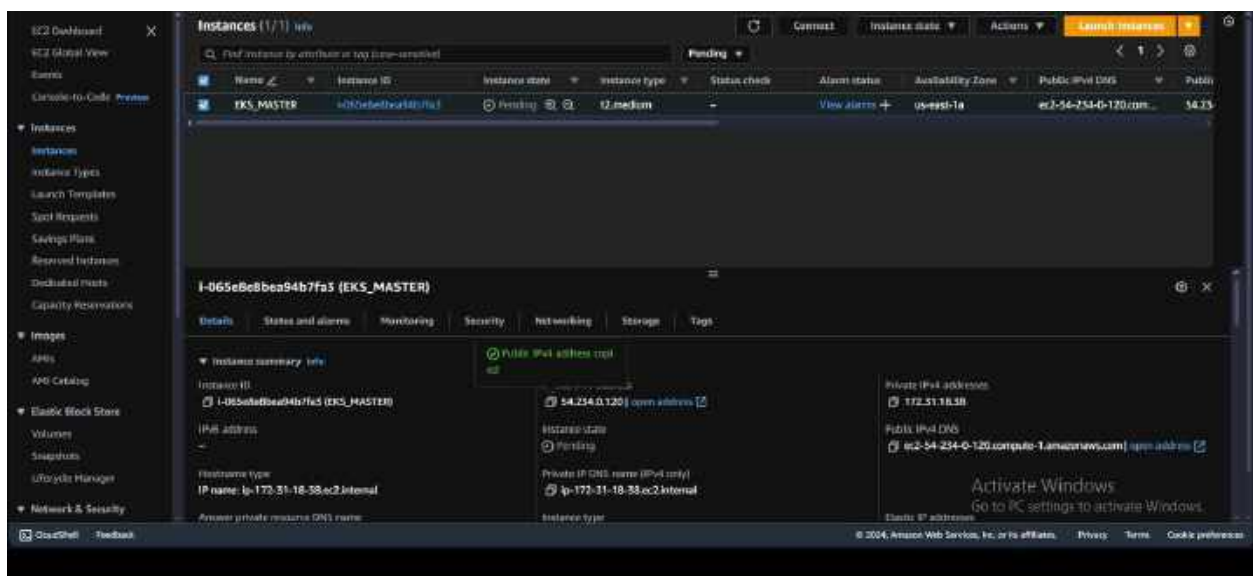
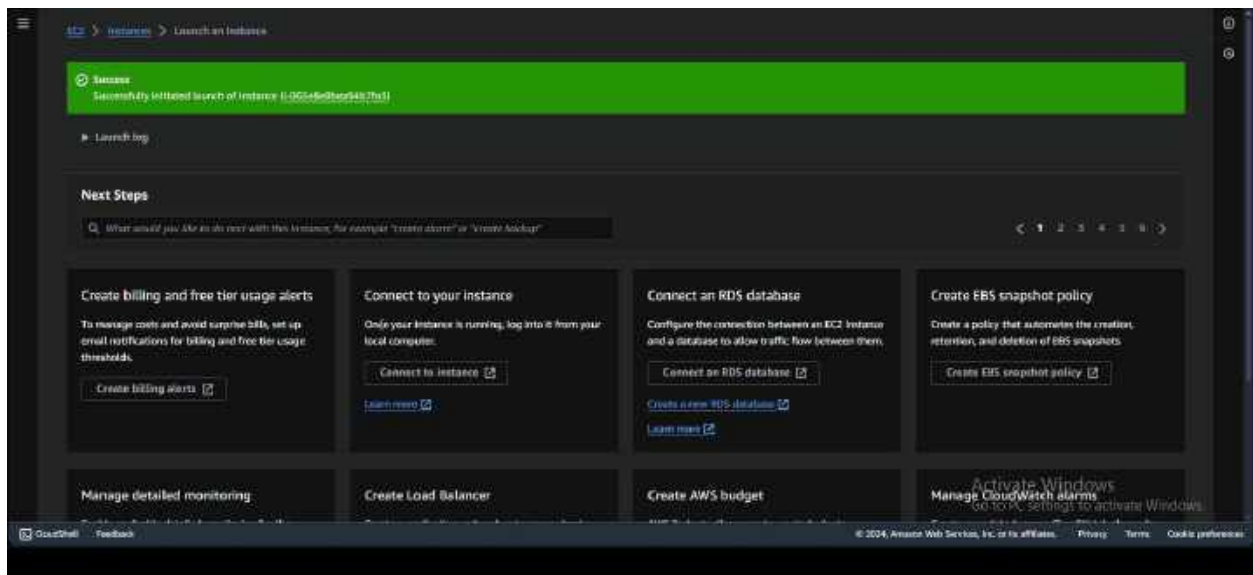
2) Create EKS Cluster

➤ Setup for EKS & launch Instance

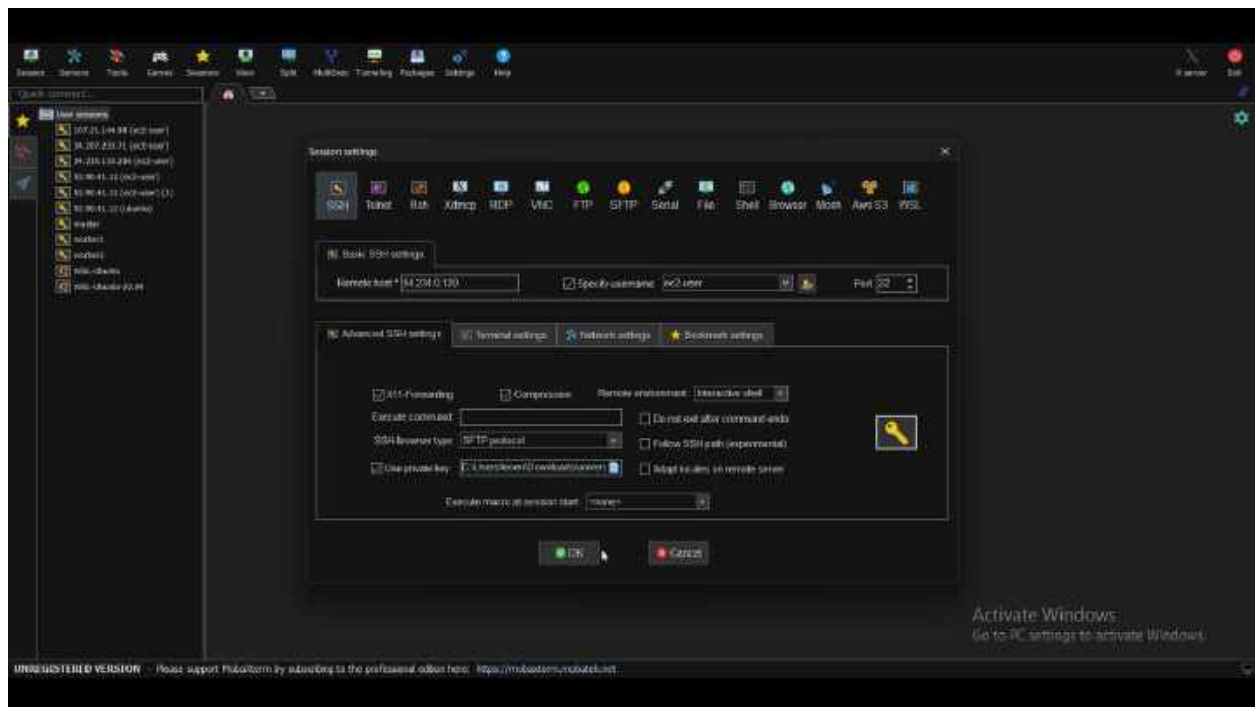


➤ Setup EC2 and Create its Instances

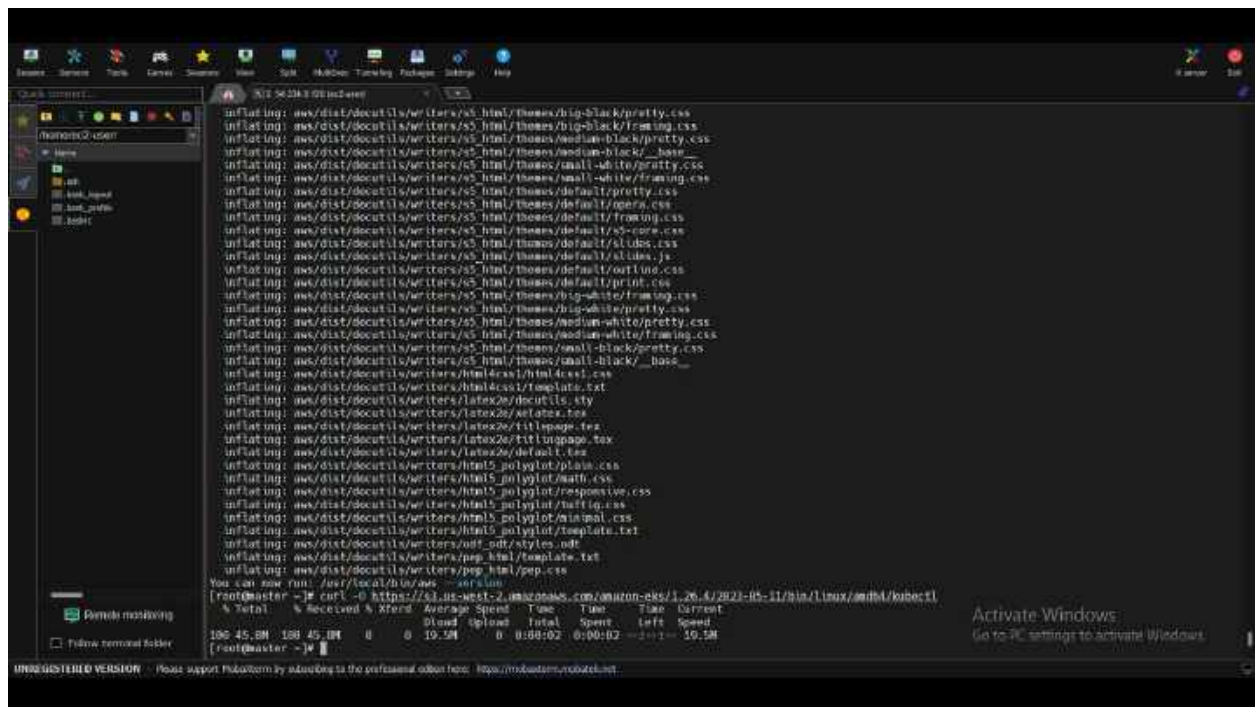
Name-Dibyajyoti Mishra



➤ Connect to SSH network through public key



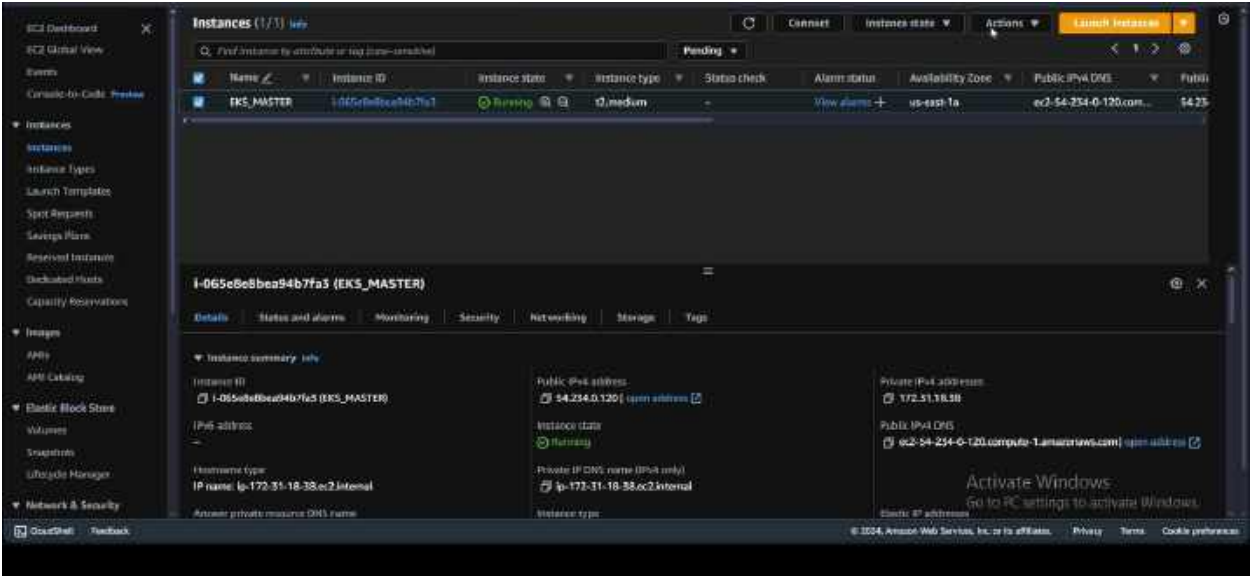
➤ *SSH Network Connected & Updated*




```

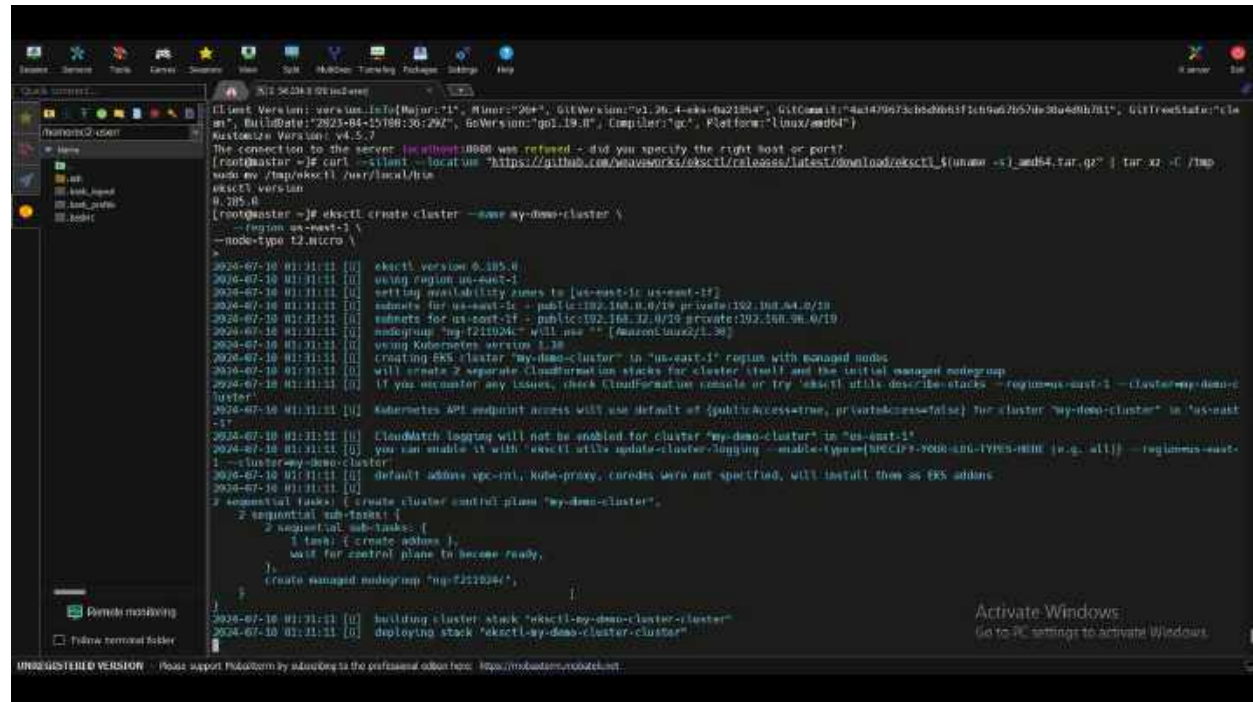
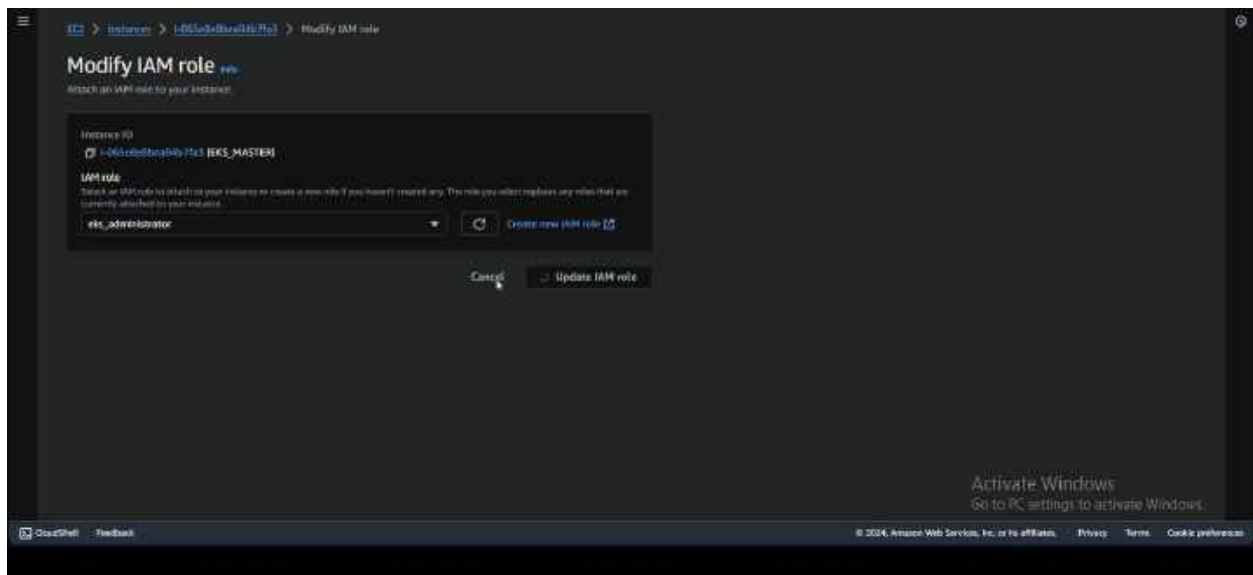
inflat ing: aws/dist/docutils/writers/s5_html/themes/small-black/pretty.css
inflat ing: aws/dist/docutils/writers/s5_html/themes/small-black/_base_
inflat ing: aws/dist/docutils/writers/html4css1/html4css1.css
inflat ing: aws/dist/docutils/writers/html4css1/template.txt
inflat ing: aws/dist/docutils/writers/latex2e/docutils.sty
inflat ing: aws/dist/docutils/writers/latex2e/exlatex.tex
inflat ing: aws/dist/docutils/writers/latex2e/titlingpage.tex
inflat ing: aws/dist/docutils/writers/latex2e/titlingpage.tex
inflat ing: aws/dist/docutils/writers/html5_polyplot/pbase.css
inflat ing: aws/dist/docutils/writers/html5_polyplot/math.css
inflat ing: aws/dist/docutils/writers/html5_polyplot/responsive.css
inflat ing: aws/dist/docutils/writers/html5_polyplot/bettig.css
inflat ing: aws/dist/docutils/writers/html5_polyplot/normal.css
inflat ing: aws/dist/docutils/writers/html5_polyplot/template.txt
inflat ing: aws/dist/docutils/writers/utf_8mb3/styles.mdt
inflat ing: aws/dist/docutils/writers/pegp.html/template.txt
inflat ing: aws/dist/docutils/writers/pegp.html/pegp.css
You can now run: /usr/local/bin/awx --version
[root@master ~]# curl -o https://cs.s3.amazonaws.com/amazon-ec2-1.26.4-2023-05-11/bin/linux/amd64/kubectl
% Total % Received % Xferd Average Speed Time Time Time Current
100 45.0M 100 45.0M 0 0 19.5M 0 0:00:02 0:00:02 --:--:-- 19.5M
[root@master ~]# ll
total 100M
drwxr-xr-x. 2 root root 7J Jul 8 18:17 aws
-rw-r--r--. 1 root root 60790936 Jul 10 01:28 awscli2.zip
-rw-r--r--. 1 root root 48617888 Jul 10 01:28 kubectl
[root@master ~]# cloud aws kubectl
[root@master ~]# aws kubectl /usr/local/bin/
[root@master ~]# aws kubectl version
WARNING: This version information is deprecated and will be replaced with the output from kubectl version --short. Use --output=yaml,json to get the full ver
sion.
Client Version: version.Info{Major:"1", Minor:"26", GitVersion:"v1.26.4-eks-8a21954", GitCommit:"4a4479673c6d49b631c09667b57de38a49b701", GitTreeState:"cle
an", BuildDate:"2023-04-15T00:36:20Z", GoVersion:"go1.19.0", Compiler:"gc", Platform:"linux/amd64"}
The connection to the server localhost:8080 was refused - did you specify the right host or port?
[root@master ~]# curl -s -o /dev/null -L https://github.com/kubernetes/kubectl/releases/latest/download/kubectl_${uname}-s_${amd64}.tar.gz | tar xz -C /tmp
sudo mv /tmp/kubectl /usr/local/bin
kubectl version
0.105.0
[root@master ~]#

```



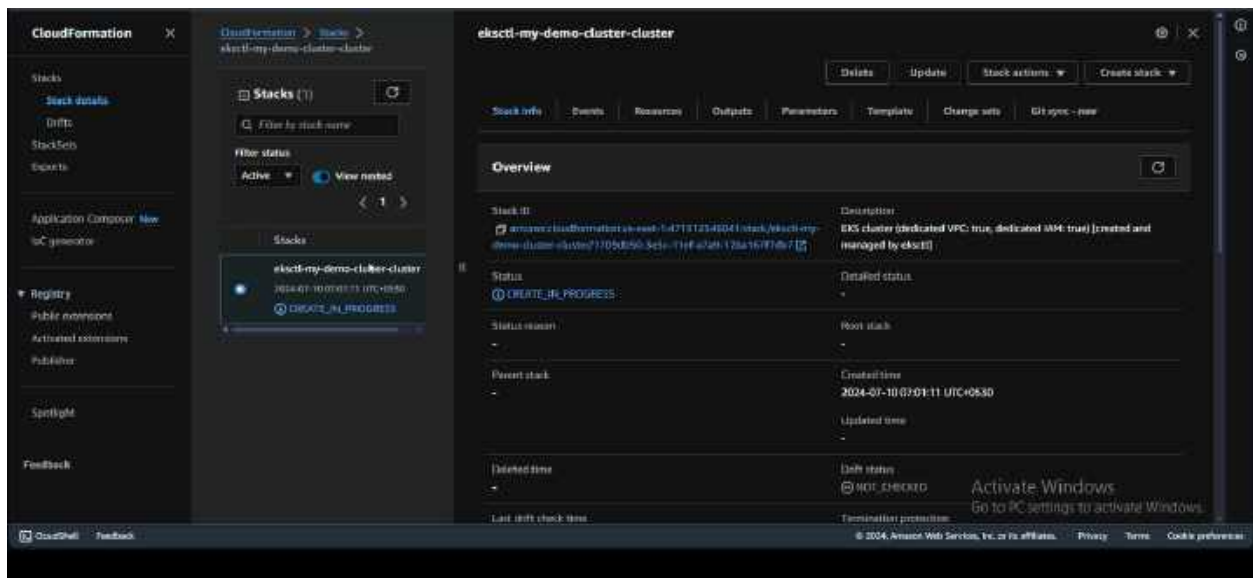
➤ *Modify IAM Roles*

Name-Dibyajyoti Mishra

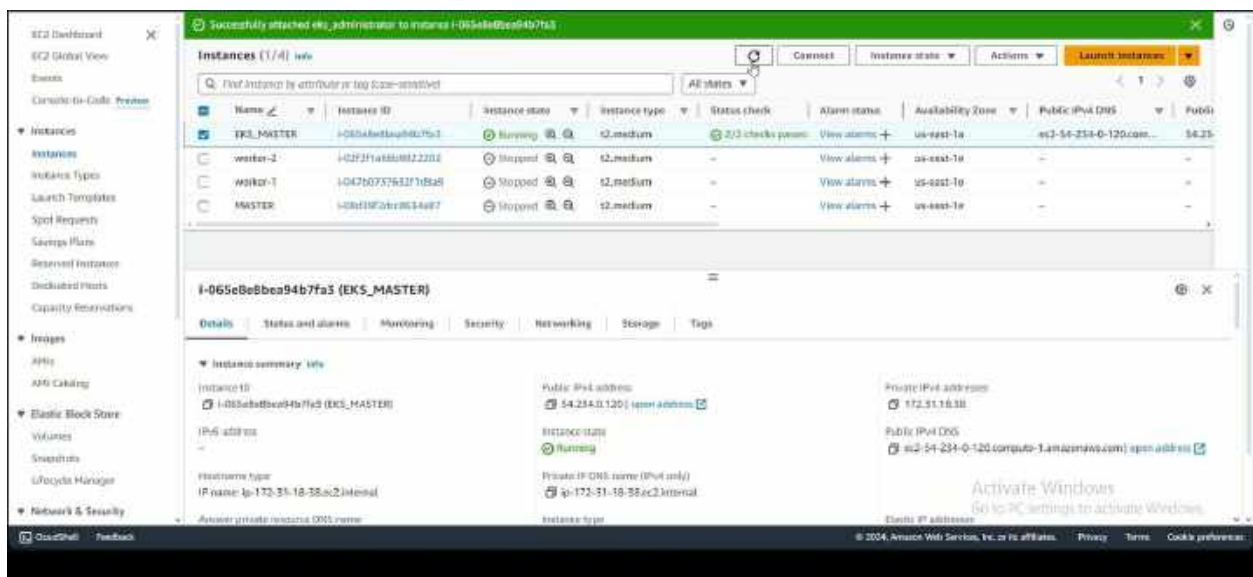


➤ Create EKS Cluster through Terraform

Name-Dibyajyoti Mishra



➤ Logging able for EKS - Setup



Name-Dibyajyoti Mishra

```
2024-07-18 01:45:17 [?] saved kubeconfig as "/root/.kube/config"
2024-07-18 01:45:17 [!] no tasks
2024-07-18 01:45:17 [?] all EKS cluster resources for "my-demo-cluster" have been created
2024-07-18 01:45:17 [?] created 8 nodegroup(s) in cluster "my-demo-cluster"
2024-07-18 01:45:17 [!] nodegroup "ng-f211924c" has 2 node(s)
2024-07-18 01:45:17 [!] node "ip-192-168-54-65.ec2.internal" is ready
2024-07-18 01:45:17 [!] node "ip-192-168-9-154.ec2.internal" is ready
2024-07-18 01:45:17 [!] waiting for at least 2 node(s) to become ready in "ng-f211924c"
2024-07-18 01:45:17 [!] nodegroup "ng-f211924c" has 2 node(s)
2024-07-18 01:45:17 [!] node "ip-192-168-54-65.ec2.internal" is ready
2024-07-18 01:45:17 [!] node "ip-192-168-9-154.ec2.internal" is ready
2024-07-18 01:45:17 [?] created 1 managed nodegroup(s) in cluster "my-demo-cluster"
2024-07-18 01:45:17 [!] kubectl command should work with "/root/.kube/config", try "kubectl get nodes"
2024-07-18 01:45:18 [?] EKS cluster "my-demo-cluster" in "us-east-1" region is ready
[root@master ~]# kubectl get nodes
NAME                                STATUS    ROLES    AGE     VERSION
ip-192-168-54-65.ec2.internal        Ready     <none>    3s44s   v1.30.8-eks-036c24b
ip-192-168-9-154.ec2.internal        Ready     <none>    3s42s   v1.30.8-eks-036c24b
[root@master ~]# kubectl get all
NAME                                TYPE                      CLUSTER-IP    EXTERNAL-IP    PORT(S)    AGE
service/kubernetes                  ClusterIP        10.100.0.1     <none>         443/TCP    10m
pod/webapp                          created
[root@master ~]# kubectl get pods
NAME    READY   STATUS    RESTARTS   AGE
webapp  1/1     Running   0           14s
[root@master ~]# kubectl create deployment demo-nginx --image=nginx --replicas=2 --port=80
deployment.apps/demo-nginx created
[root@master ~]# kubectl get replicaset
NAME                                DESIRED   CURRENT   READY   AGE
demo-nginx-5d74cc7bf                2         2         1       3s37s
[root@master ~]# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
demo-nginx-5d74cc7bf-cppsn          1/1     Running   0           3s50s
demo-nginx-5d74cc7bf-t7bq7         0/1     Pending   0           3s50s
webapp                               1/1     Running   0           4s24s
[root@master ~]# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
demo-nginx-5d74cc7bf-cppsn          1/1     Running   0           4s8s
demo-nginx-5d74cc7bf-t7bq7         0/1     Pending   0           4s8s
webapp                               1/1     Running   0           4s34s
[root@master ~]#
```

```
2024-07-18 01:45:17 [?] all EKS cluster resources for "my-demo-cluster" have been created
2024-07-18 01:45:17 [?] created 8 nodegroup(s) in cluster "my-demo-cluster"
2024-07-18 01:45:17 [!] nodegroup "ng-f211924c" has 2 node(s)
2024-07-18 01:45:17 [!] node "ip-192-168-54-65.ec2.internal" is ready
2024-07-18 01:45:17 [!] node "ip-192-168-9-154.ec2.internal" is ready
2024-07-18 01:45:17 [!] waiting for at least 2 node(s) to become ready in "ng-f211924c"
2024-07-18 01:45:17 [!] nodegroup "ng-f211924c" has 2 node(s)
2024-07-18 01:45:17 [!] node "ip-192-168-54-65.ec2.internal" is ready
2024-07-18 01:45:17 [!] node "ip-192-168-9-154.ec2.internal" is ready
2024-07-18 01:45:17 [?] created 1 managed nodegroup(s) in cluster "my-demo-cluster"
2024-07-18 01:45:17 [!] kubectl command should work with "/root/.kube/config", try "kubectl get nodes"
2024-07-18 01:45:18 [?] EKS cluster "my-demo-cluster" in "us-east-1" region is ready
[root@master ~]# kubectl get nodes
NAME                                STATUS    ROLES    AGE     VERSION
ip-192-168-54-65.ec2.internal        Ready     <none>    3s44s   v1.30.8-eks-036c24b
ip-192-168-9-154.ec2.internal        Ready     <none>    3s42s   v1.30.8-eks-036c24b
[root@master ~]# kubectl get all
NAME                                TYPE                      CLUSTER-IP    EXTERNAL-IP    PORT(S)    AGE
service/kubernetes                  ClusterIP        10.100.0.1     <none>         443/TCP    10m
pod/webapp                          created
[root@master ~]# kubectl get pods
NAME    READY   STATUS    RESTARTS   AGE
webapp  1/1     Running   0           14s
[root@master ~]# kubectl create deployment demo-nginx --image=nginx --replicas=2 --port=80
deployment.apps/demo-nginx created
[root@master ~]# kubectl get replicaset
NAME                                DESIRED   CURRENT   READY   AGE
demo-nginx-5d74cc7bf                2         2         1       3s37s
[root@master ~]# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
demo-nginx-5d74cc7bf-cppsn          1/1     Running   0           3s50s
demo-nginx-5d74cc7bf-t7bq7         0/1     Pending   0           3s50s
webapp                               1/1     Running   0           4s24s
[root@master ~]# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
demo-nginx-5d74cc7bf-cppsn          1/1     Running   0           4s8s
demo-nginx-5d74cc7bf-t7bq7         0/1     Pending   0           4s8s
webapp                               1/1     Running   0           4s34s
[root@master ~]# kubectl expose deployment demo-nginx --port=80 --type=LoadBalancer
service/demo-nginx exposed
[root@master ~]#
```

➤ Finally create EKS Cluster through EC2 Instances using Terraform

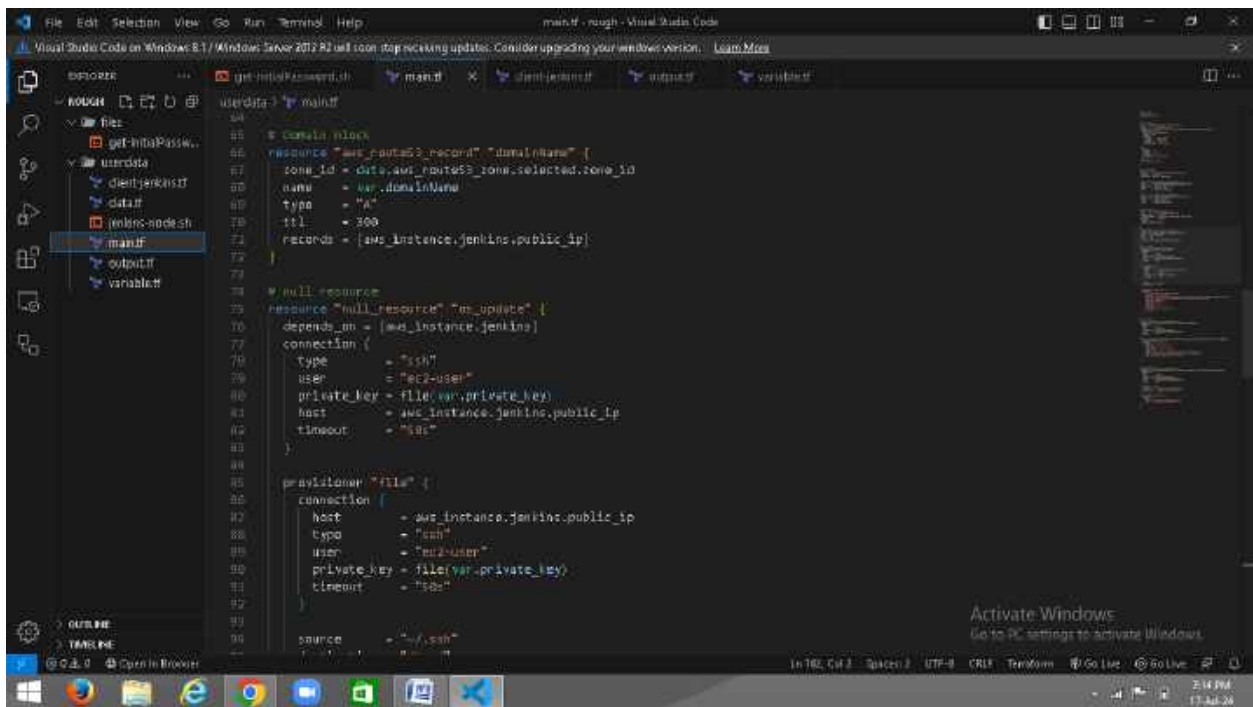
```

ip-192-168-0-104.ec2.internal 3642s v1.30.0-eks-03bc24b
[rooft@master ~]$ kubectl get all
NAME                                TYPE                                CLUSTER-IP    EXTERNAL-IP    PORT(S)    AGE
service/kubernetes                  ClusterIP      10.100.0.1     <none>         443/TCP    10m
[rooft@master ~]$ kubectl run webapp --image=httpd
pod/webapp created
[rooft@master ~]$ kubectl get pods
NAME                                READY    STATUS    RESTARTS    AGE
webapp-121                          1/1     Running   0           14s
[rooft@master ~]$ kubectl create deployment demo-nginx --image=nginx --replicas=2 --port=80
deployment.apps/demo-nginx created
[rooft@master ~]$ kubectl get replicaset
NAME                                DESIRED    CURRENT    READY    AGE
demo-nginx-57d74cc7bf-2             2          2          1        3m07s
[rooft@master ~]$ kubectl get pods
NAME                                READY    STATUS    RESTARTS    AGE
demo-nginx-57d74cc7bf-cppsn         1/1     Running   0           3m58s
demo-nginx-57d74cc7bf-t2bq7         0/1     Pending   0           3m58s
webapp-121                          1/1     Running   0           4m29s
[rooft@master ~]$ kubectl get pods
NAME                                READY    STATUS    RESTARTS    AGE
demo-nginx-57d74cc7bf-cppsn         1/1     Running   0           4m8s
demo-nginx-57d74cc7bf-t2bq7         0/1     Pending   0           4m8s
webapp-121                          1/1     Running   0           4m34s
[rooft@master ~]$ kubectl expose deployment demo-nginx --port=80 --type=LoadBalancer
service/demo-nginx exposed
[rooft@master ~]$ kubectl get all
NAME                                TYPE                                CLUSTER-IP    EXTERNAL-IP    PORT(S)    AGE
service/demo-nginx                  LoadBalancer  10.100.222.86  a953e426941a04a17a687be2cde5a42-010237570.us-east-1.elb.amazonaws.com  80:30760/TCP  13s
service/kubernetes                  ClusterIP      10.100.0.1     <none>         443/TCP    10m
NAME                                READY    UP-TO-DATE    AVAILABLE    AGE
deployment.apps/demo-nginx          1/2      2              1            4m40s
NAME                                DESIRED    CURRENT    READY    AGE
replicaset.apps/demo-nginx-57d74cc7bf-2 2          2          1        4m43s
[rooft@master ~]$

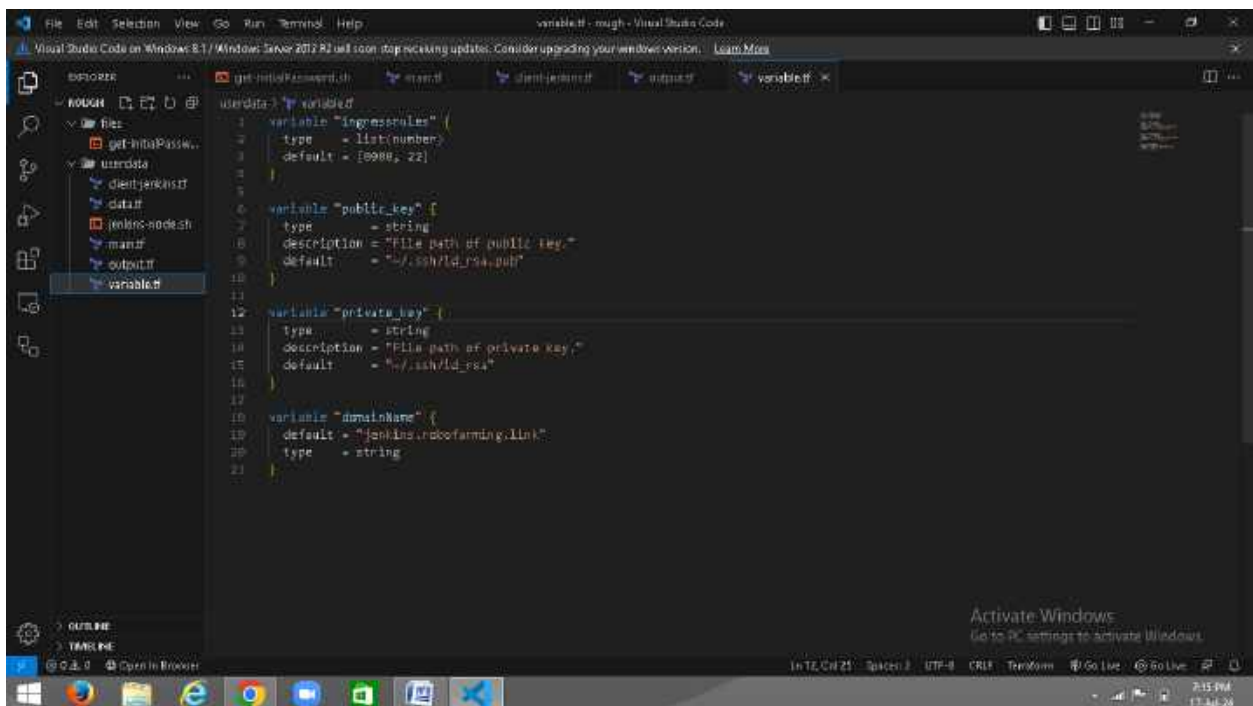
```

3) Create worker node using terraform

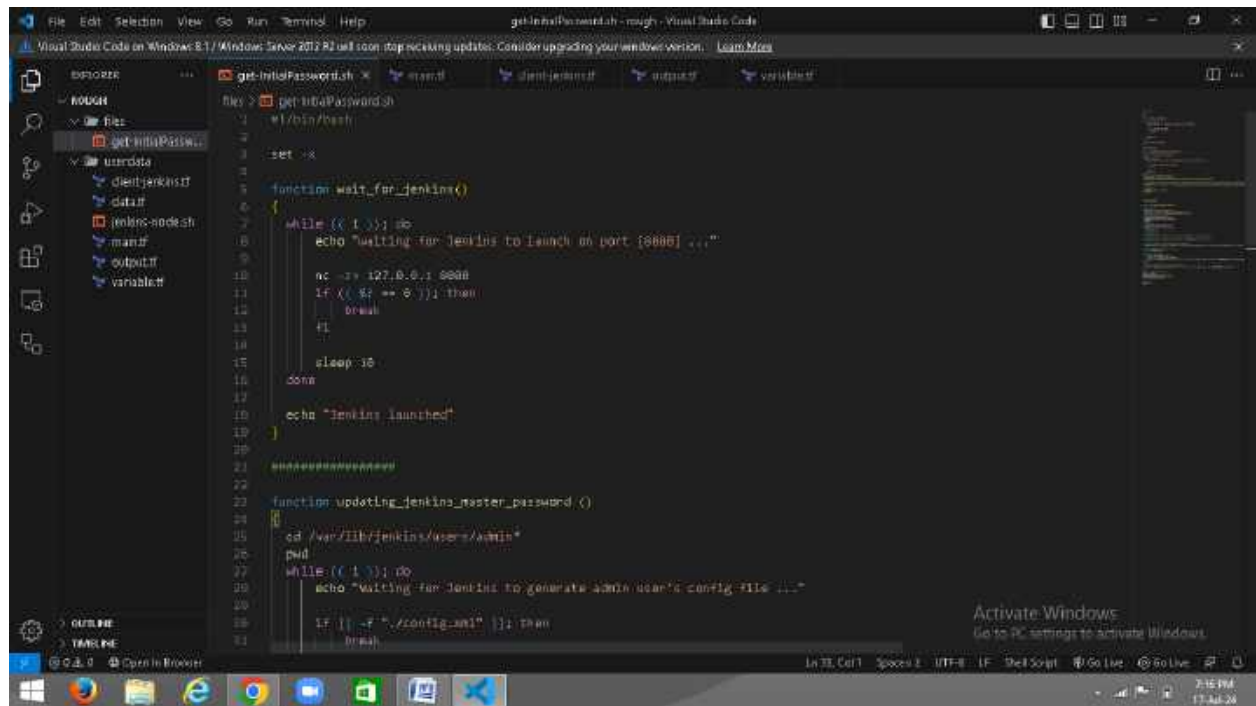
Main.tf -: for security groups



➤ Variable.tf

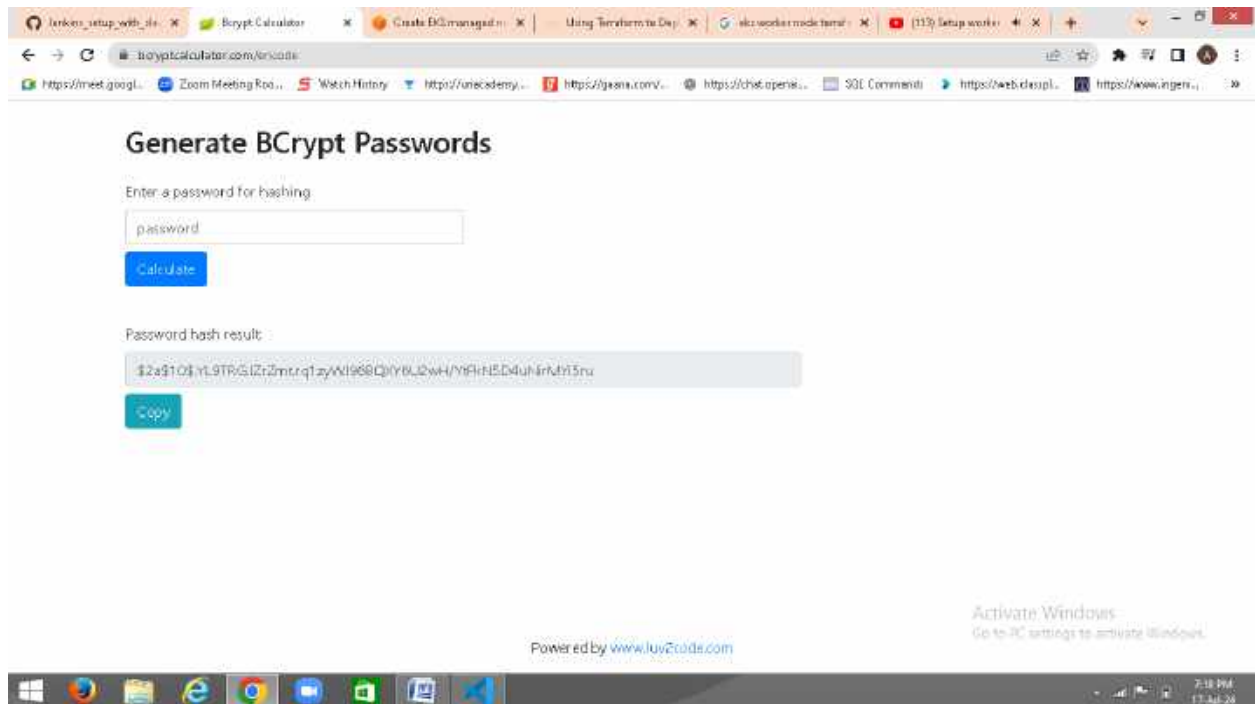


➤ Password.sh



```
1 # Jenkins Password Generator Script
2 # This script will wait for Jenkins to launch on port 8080 and then
3 # generate a random password for the Jenkins administrator user.
4
5 set -x
6
7 function wait_for_jenkins()
8 {
9     while (( 1 )) do
10         echo "Waiting for Jenkins to launch on port [8080] ..."
11         nc -z 127.0.0.1 8080
12         if (( $? == 0 )) then
13             break
14         fi
15         sleep 10
16     done
17     echo "Jenkins launched"
18 }
19
20 #####
21
22 function updating_jenkins_admin_password()
23 {
24     cd /var/lib/jenkins/users/admin*
25     pwid
26     while (( 1 )) do
27         echo "Waiting for Jenkins to generate admin user's config file ..."
28         if [[ -f "/config.xml" ]] then
29             break
30         fi
31     done
32 }
```

➤ Bcrypt calculator



Generate BCrypt Passwords

Enter a password for hashing:

Calculate

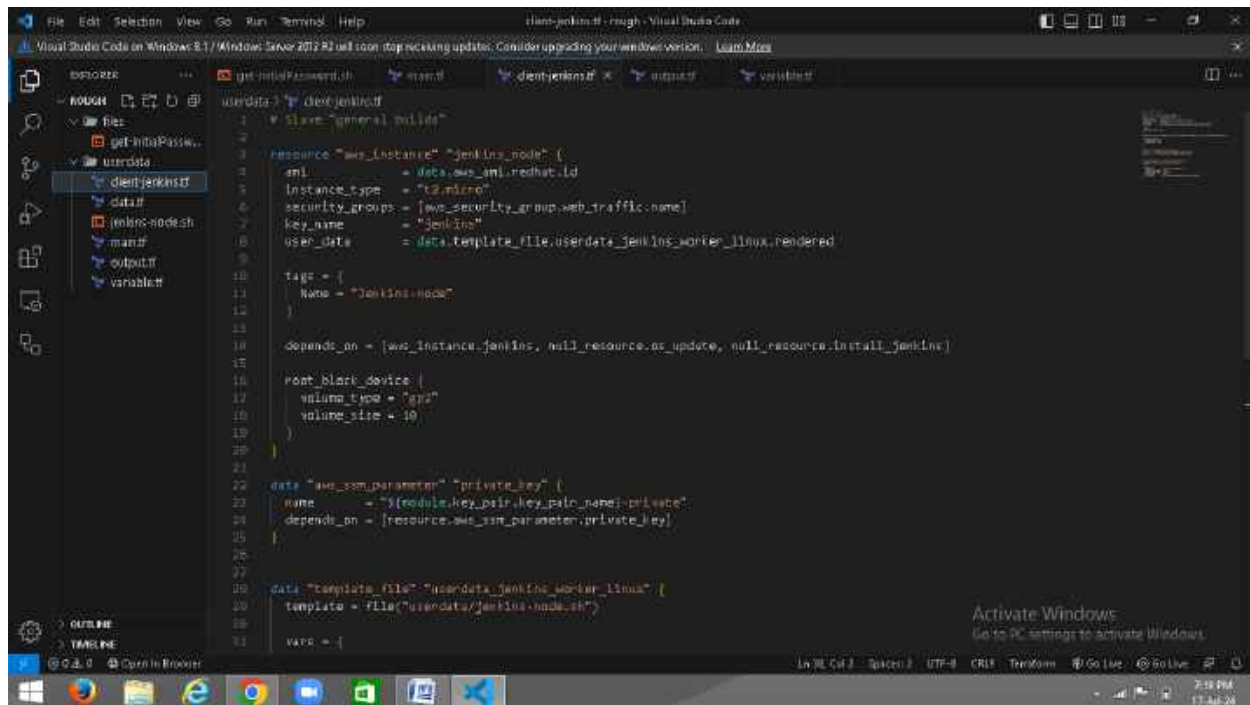
Password hash result :

\$2a\$10\$Y1L9TR\$IZrZmzq1zyW1988QhYBU2WH/YEhH5D4u4nKv13ru

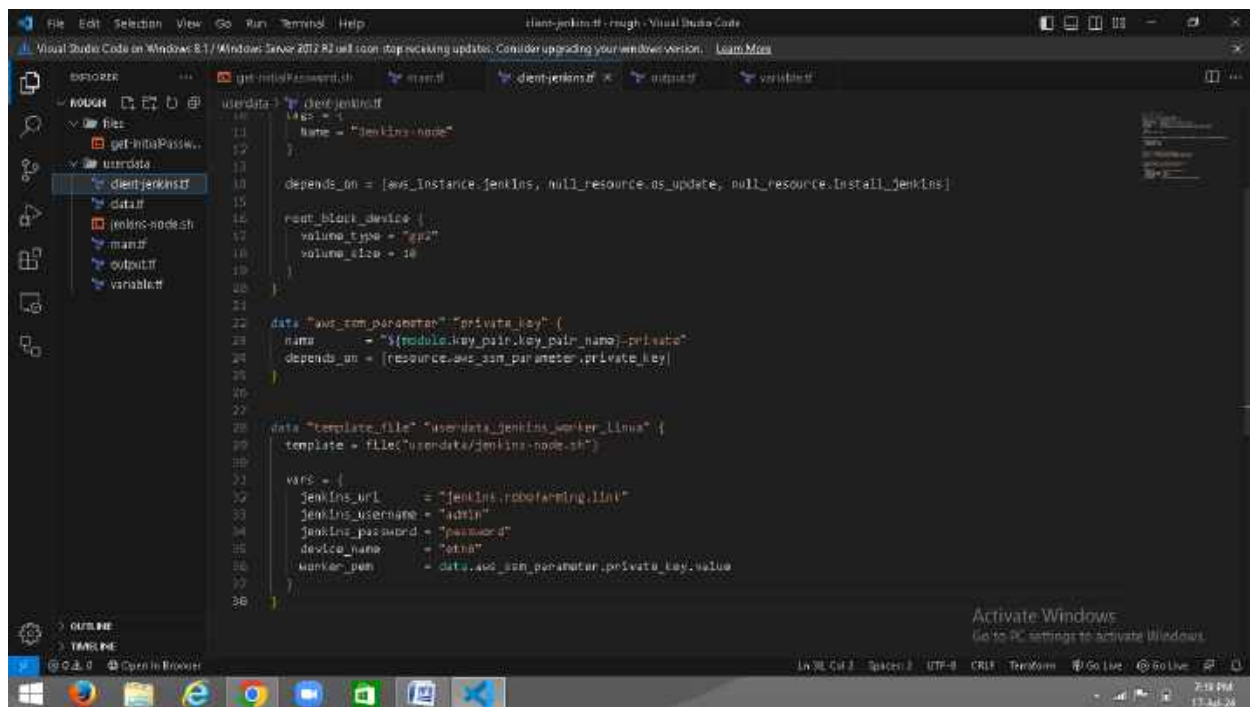
Copy

Powered by www.luv2code.com

- *Client-jenkins.tf -> slave general builds and setup resource*



```
1 userdata > client-jenkins.tf
2
3 resource "aws_instance" "jenkins_node" {
4   ami           = data.aws_ami.ubuntu.id
5   instance_type = "t2.micro"
6   security_groups = [aws_security_group.web_traffic.name]
7   key_name      = "jenkins"
8   user_data     = data.template_file.userdata_jenkins_worker_linux.rendered
9
10  tags = {
11    Name = "jenkins-node"
12  }
13
14  depends_on = [aws_instance.jenkins, null_resource.os_update, null_resource.install_jenkins]
15
16  root_block_device {
17    volume_type = "gp2"
18    volume_size = 10
19  }
20
21  data "aws_ssm_parameter" "private_key" {
22    name = "${module.key_pair.key_pair_name}-private"
23    depends_on = [resource.aws_ssm_parameter.private_key]
24  }
25
26  data "template_file" "userdata_jenkins_worker_linux" {
27    template = file("userdata/jenkins-node.sh")
28  }
29
30  vars = {
```



```
31
32  jenkins_url = "jenkins.robotfarmg.link"
33  jenkins_username = "admin"
34  jenkins_password = "password"
35  device_name     = "eth0"
36  worker_pem     = data.aws_ssm_parameter.private_key.value
37
38 }
```

The screenshot shows the Visual Studio Code editor with a file explorer on the left and a terminal window. The file explorer shows a project structure with folders like 'file', 'userdata', and 'jenkins-node.sh'. The terminal window displays the following code:

```
#!/bin/bash
set -x

sudo yum install java-1.8-openjdk -y
sudo yum -y install git
sudo yum -y install wget
sudo yum install xelstarlet -y
sudo sleep 120

function slave_setup()
{
    # Wait till jar file gets available
    ret=1
    while (( $ret != 0 )) do
        sudo wget -O /opt/jenkins-cli.jar http://$jenkins_url:8080/jnlpJars/jenkins-cli.jar
        ret=$?
        echo "jenkins cli ret ($ret)"
    done

    ret=1
    while (( $ret != 0 )) do
        sudo wget -O /opt/slave.jar http://$jenkins_url:8080/jnlpJars/slave.jar
        ret=$?
        echo "jenkins slave ret ($ret)"
    done
}
```

The bottom of the terminal window shows the status bar with 'Ln 1, Col 1', 'Spaces: 2', 'UTF-8', 'LF', 'Shell Script', 'Go Live', 'Go Live', and a timestamp of '7:19 PM 17 Jul 20'.

The screenshot shows the Visual Studio Code editor with a file explorer on the left and a terminal window. The file explorer shows a project structure with folders like 'file', 'userdata', and 'jenkins-node.sh'. The terminal window displays the following code:

```
#!/bin/bash
set -x

sudo wget -O /opt/slave.jar http://$jenkins_url:8080/jnlpJars/slave.jar
ret=$?
echo "jenkins slave ret ($ret)"
done

#####
# Register slave
JENKINS_URL="http://$jenkins_url:8080"

USERNAME="$jenkins_username"

# PASSWORD=$(cat /dev/urandom | fold -w 32 | tr -dc 'a-z0-9' | fold -w 1 | xargs -n 1 | tr -d '\n')
PASSWORD="$jenkins_password"

SLAVE_IP=$(ip -o -4 addr list $(hostname) | head -n1 | awk '{print $4}' | cut -d/ -f1)
NODE_NAME=$(echo "jenkins-slave-linux-$SLAVE_IP" | tr ' ' -)
NODE_SLAVE_HOME="/home/$NODE_NAME"
EXECUTOR_LABEL="ec2-user"
SSH_PORT=22

CRED_ID="BMSIDE_NAME"
LABELS="linux"
USERID="ec2-user"

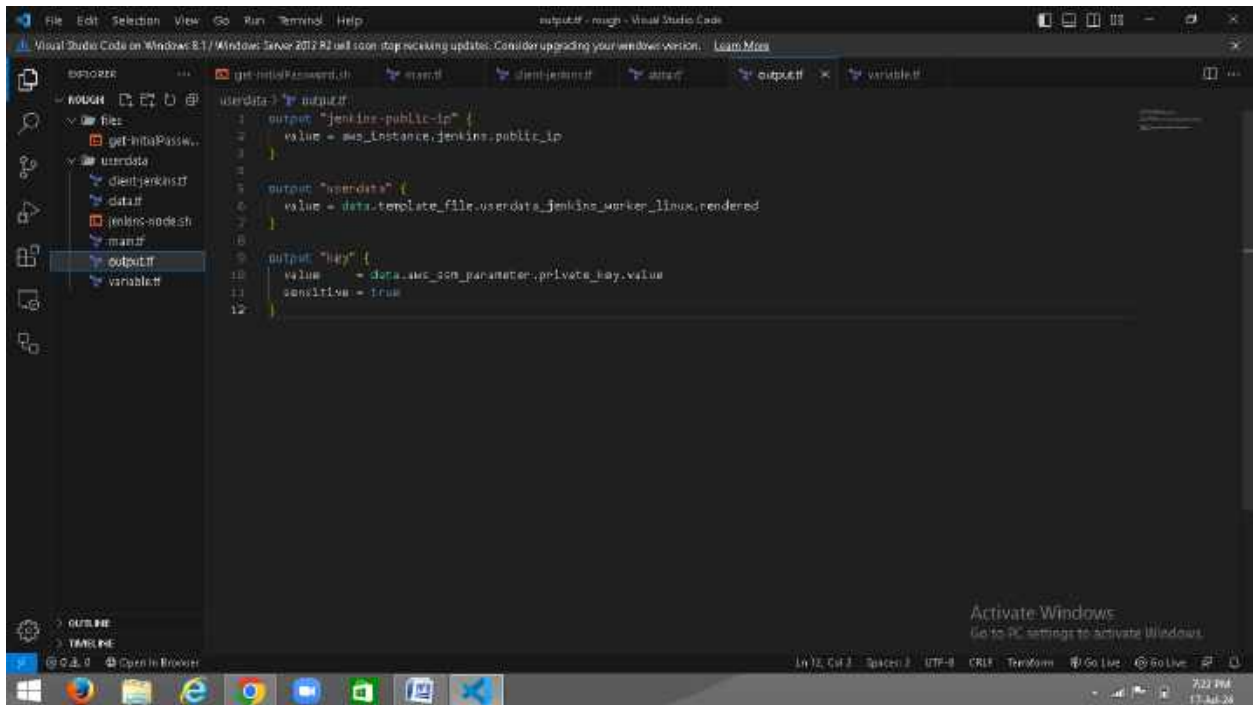
cd /opt

# Creating GNU utility for Jenkins cli commands
```

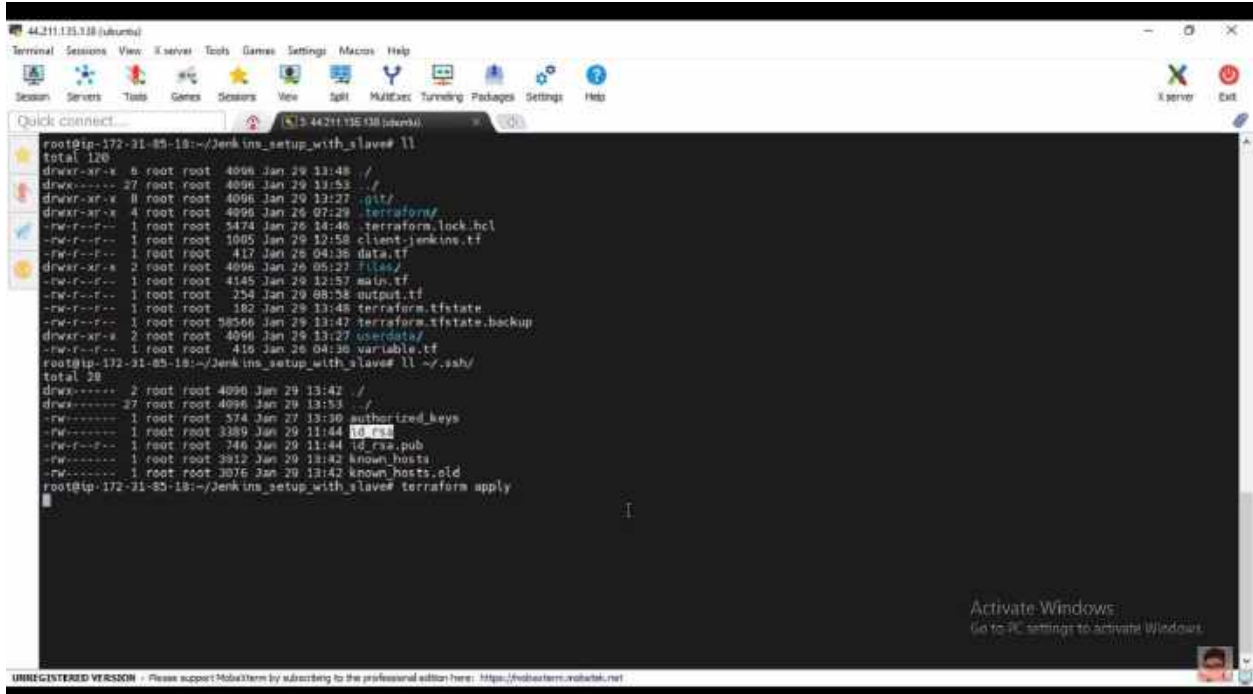
The bottom of the terminal window shows the status bar with 'Ln 1, Col 1', 'Spaces: 2', 'UTF-8', 'LF', 'Shell Script', 'Go Live', 'Go Live', and a timestamp of '7:20 PM 17 Jul 20'.


```
40 userdata {
41   jenkins-node.sh
42 }
43
44 CRED_ID="${NODE_NAME}"
45 LABELS="linux"
46 USERID="ec2-user"
47
48 cd /opt
49
50 # Creating CWU utility for Jenkins CLI commands
51 jenkins_cmd="java -jar /opt/jenkins-cli.jar -s $JENKINS_URL -auth $USERNAME:$PASSWORD"
52
53 # Writing for Jenkins to load all plugins
54 while (( 1 )) ; do
55
56   count=$(($jenkins_cmd list-plugins 2>/dev/null | wc -l)
57   ret=$?
58
59   echo "count: [$count] ret: [$ret]"
60
61   if (( $count > 8 )) ; then
62     break
63   fi
64
65   sleep 30
66 done
67
68 # Delete Credentials if present for respective slave machines
69 $jenkins_cmd delete-credentials system:system:jenkins_$CRED_ID
70
71 # Generating cred.xml for creating credentials on Jenkins server
72 cat > /tmp/cred.xml <EOF
73 <com.cloudbees.jenkins.plugins.sshcredentials.impl.BasicSSHUserPrivateKey plugin="ssh-credentials@1.16">
74   <scope>GLOBAL</scope>
75   <id>$CRED_ID</id>
76   <description>Generated via Terraform for $SLAVE_ID</description>
77   <username>$USERID</username>
78   <privateKeySource class="com.cloudbees.jenkins.plugins.sshcredentials.impl.BasicSSHUserPrivateKey$DirectEntryPrivateKeySource">
79     <privateKey>$(cat /dev/urandom | tr -dc 'a-z0-9' | fold -w 64 | xargs printf '%s\n')</privateKey>
80   </privateKeySource>
81 </com.cloudbees.jenkins.plugins.sshcredentials.impl.BasicSSHUserPrivateKey>
82 EOF
83
84 # Creating credential using cred.xml
85 cat /tmp/cred.xml | $jenkins_cmd create-credentials-by-xml system:system:jenkins_$CRED_ID
86
87 # For deleting node, used when testing
88 $jenkins_cmd delete-node $NODE_NAME
89
90 # Generating node.xml for creating node on Jenkins server
91 cat > /tmp/node.xml <EOF
92 <slave>
93   <name>$NODE_NAME</name>
94   <description>linux slave</description>
95   <motd>${NODE_SLAVE_MOTD}</motd>
96 </slave>
```

```
60 done
61
62 # Delete Credentials if present for respective slave machines
63 $jenkins_cmd delete-credentials system:system:jenkins_$CRED_ID
64
65 # Generating cred.xml for creating credentials on Jenkins server
66 cat > /tmp/cred.xml <EOF
67 <com.cloudbees.jenkins.plugins.sshcredentials.impl.BasicSSHUserPrivateKey plugin="ssh-credentials@1.16">
68   <scope>GLOBAL</scope>
69   <id>$CRED_ID</id>
70   <description>Generated via Terraform for $SLAVE_ID</description>
71   <username>$USERID</username>
72   <privateKeySource class="com.cloudbees.jenkins.plugins.sshcredentials.impl.BasicSSHUserPrivateKey$DirectEntryPrivateKeySource">
73     <privateKey>$(cat /dev/urandom | tr -dc 'a-z0-9' | fold -w 64 | xargs printf '%s\n')</privateKey>
74   </privateKeySource>
75 </com.cloudbees.jenkins.plugins.sshcredentials.impl.BasicSSHUserPrivateKey>
76 EOF
77
78 # Creating credential using cred.xml
79 cat /tmp/cred.xml | $jenkins_cmd create-credentials-by-xml system:system:jenkins_$CRED_ID
80
81 # For deleting node, used when testing
82 $jenkins_cmd delete-node $NODE_NAME
83
84 # Generating node.xml for creating node on Jenkins server
85 cat > /tmp/node.xml <EOF
86 <slave>
87   <name>$NODE_NAME</name>
88   <description>linux slave</description>
89   <motd>${NODE_SLAVE_MOTD}</motd>
90 </slave>
```

➤ For setup terraform



Name-Dibyajyoti Mishra

```
44.211.135.138 (ubuntu)
Terminal Sessions View Xserver Tools Games Settings Macros Help
Session Servers Tools Games Dossiers View Split Multitex Tunneling Packages Settings Help

Quick connect... (S) 44.211.135.138 (jenkins)

+ resource "null_resource" "install_plugin" {
+   id = (known after apply)
+ }

# null_resource.os_updates will be created
+ resource "null_resource" "os_update" {
+   id = (known after apply)
+ }

# module.key_pair.aws_key_pair.this[0] will be created
+ resource "aws_key_pair" "this" {
+   arn          = (known after apply)
+   fingerprint = (known after apply)
+   id           = (known after apply)
+   key_name     = "jenkins"
+   key_name_prefix = (known after apply)
+   key_pair_id  = (known after apply)
+   key_type     = (known after apply)
+   public_key   = "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCAQDD05gcYUcWSED4i4eAVT UHR+AOE7PS+R0UQ7T7wzMTJrKSL3M6BRezt7L/WGZTSU01gR+VIX16xrcRf5v3T2PaZ+Hg0K1KNTJ
+   b6gg7PEKq45hVn0d6k8m8m8Z7v0hg/d03mgd12u0m0a20H760P5m1+H600uYjNIPV8p+fmM/ZI8DO0RQg+uR8NvVeK0W2+asg0Z055p0y0ckZ47/CY4KXGp00F0K0/v00H0X0K0R0L+0aPE0eejE
+   x0P7702Y0T0p0h0m01a0u0v0t0u0k0L0F0r03Y0S0v00F0Z02F0IE0++00W0P0D08K+P530p0+Y0A0Y0R05V0P0J0Z03Y0Ed00S0S090p0B000+0IND0M0X0p0K0010K0H0+0y0g+00g0p0m0X0y0v0J0R0S0H0v0000F0b00p00w01y0
+   a74k20H0P0V50A0SBrt040e0H00k00T0s020U0K0Y0d0r0w00K0St0V075f0m0f0s0z0V0C0v0i0SL02F03J0t0W0t0r0K090571/L0IV0Z0H0M0SN0K0Y0N0Z0r00a0/n1J0b0M000r0M0B0F+0p0e0n0B0L0t0/07+40qY0z0f0P0y00B0A0l0r0d0v03f0870
+   c0G000m0h0u00U0Q0U0T0z0w0y0Z0U0H0E4E0e0c0H0m0p02k0L0y0f+0m0R00G0p02X0S40G0V0p050X0p0h0v0q0l0q0d0m000e0V0M0j0M0X0u0= root@ip-172-31-85-18"
+   tags_all    = (known after apply)
+ }

Plans: 10 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ jenkins-public-ip = (known after apply)
+ key               = (sensitive value)
+ userdata          = (known after apply)

Do you want to perform these actions?
Terraform will perform the actions described above.
only 'yes' will be accepted to approve.

Enter a value: 
```

➤ *Slave setup and terraform created*

```
44.211.135.138 (ubuntu)
Terminal Sessions View Xserver Tools Games Settings Macros Help
Session Servers Tools Games Dossiers View Split Multitex Tunneling Packages Settings Help

Quick connect... (S) 44.211.135.138 (jenkins)

# For Deleting Node, used when testing
jenkins_cmd delete-node $NODE_NAME

# Generating node.xml for creating node on Jenkins server
cat > /tmp/node.xml <<EOF

<?xml version="1.0">
<slave name="$NODE_NAME" </name>
<description>Linux Slave</description>
<remoteFS>$NODE_SLAVE_HOME</remoteFS>
<numExecutors>$EXECUTORS</numExecutors>
<mode>NORMAL</mode>
<retentionStrategy class="hudson.slaves.RetentionStrategy$Always"/>
<launcher class="hudson.plugins.sshslaves.SSHLauncher" plugin="ssh-slaves@1.3">
  <host>$SLAVE_IP</host>
  <port>$SSH_PORT</port>
  <credentialsId>$CRED_ID</credentialsId>
  <class>"hudson.plugins.sshslaves.verifiers.ManuallyTrustedKeyVerificationStrategy"/>
</launcher>
<label>$LABELS</label>
<modeProperties>
  <userId>$USERID</userId>
</modeProperties>
</slave>
</?xml>
EOF

sleep 10

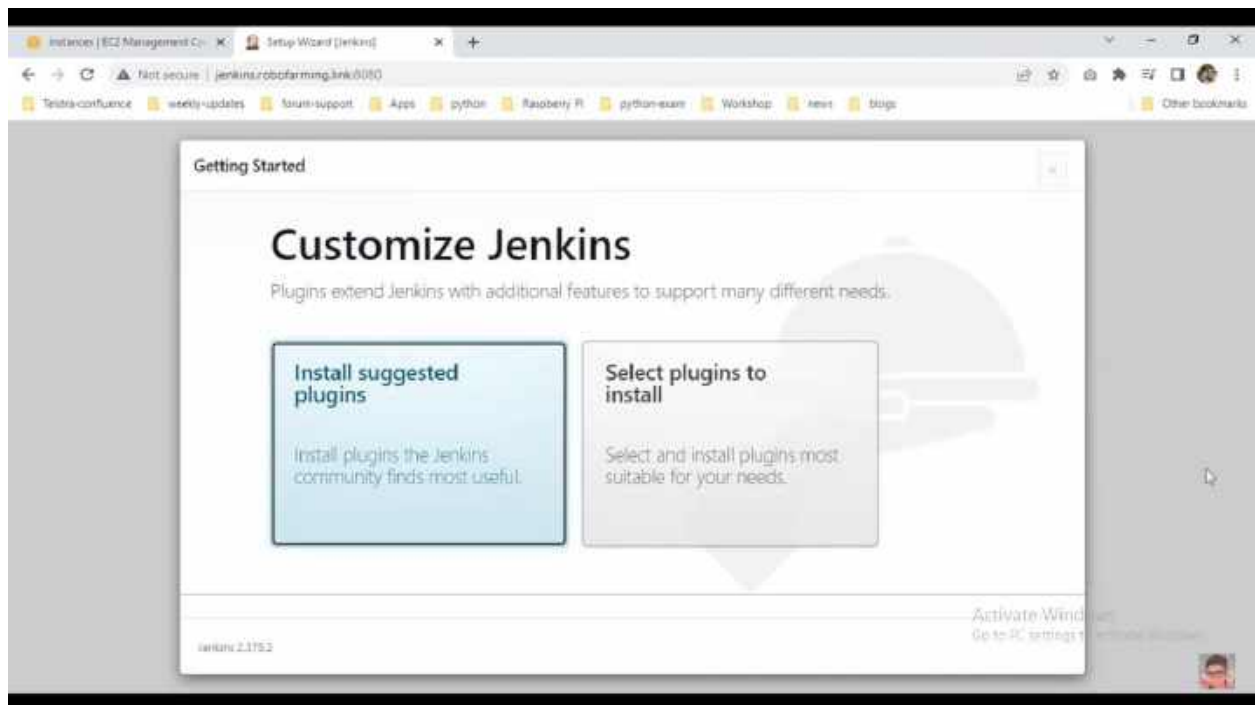
# Creating node using node.xml
cat /tmp/node.xml | jenkins_cmd create-node $NODE_NAME

### script begins here ###

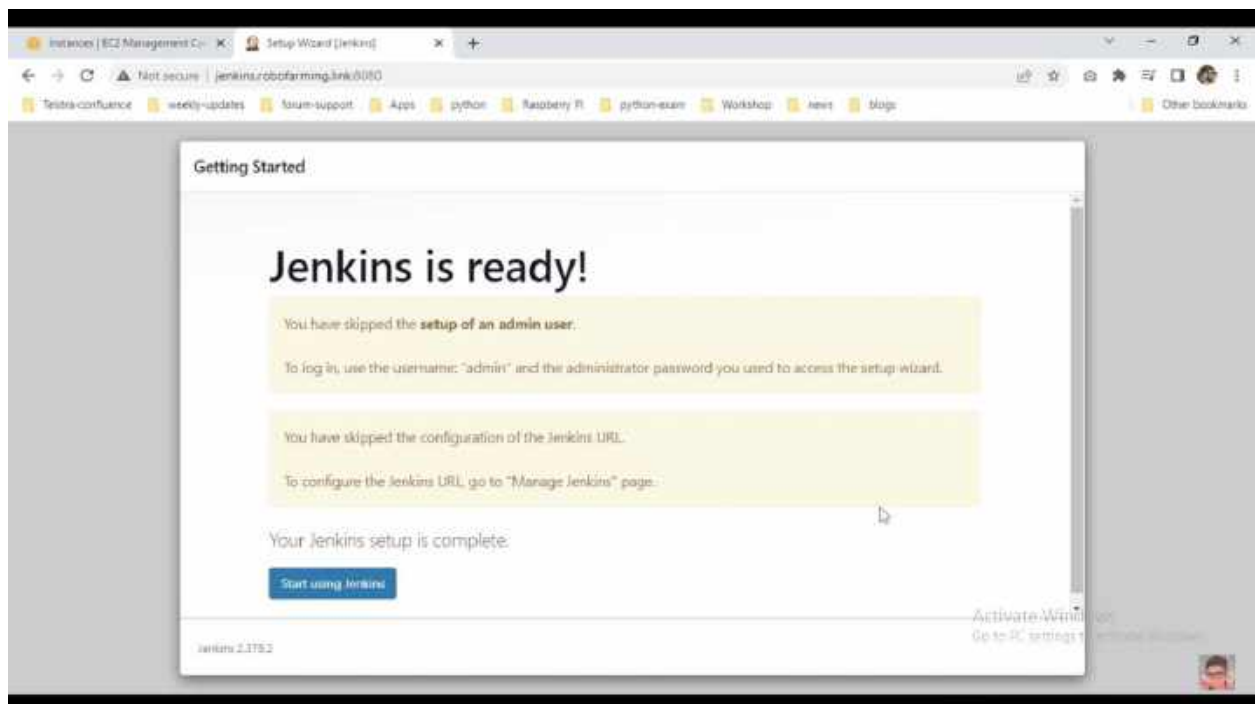
slave_setup

echo "Done"
exit 0
EOF

root@ip-172-31-85-18:~/jenkins_setup_with_slaves
```



➤ *Completed Jenkins for managing nodes*



➤ Terraform Generated

The screenshot shows the Jenkins web interface at the 'Credentials' page. The breadcrumb trail is 'Dashboard > Manage Jenkins > Credentials'. The main heading is 'Credentials'. Below it is a table with columns: 'T' (Type), 'P' (Project), 'Store', 'Domain', 'ID', and 'Name'. One credential is listed with ID 'jenkins-slave-linux-172-31-48-185' and Name 'Generated via Terraform for 172.31.48.185'. Below the table is a section 'Stores scoped to Jenkins' with a table showing 'System' store for 'global' domain. At the bottom right, there is a 'REST API' link and 'Jenkins 2.3' version indicator.

T	P	Store	Domain	ID	Name
		System	global	jenkins-slave-linux-172-31-48-185	Generated via Terraform for 172.31.48.185

P	Store	Domains
	System	global

REST API Jenkins 2.3

➤ Worker nodes initiated

The screenshot shows the Jenkins web interface at the 'Nodes' page. The breadcrumb trail is 'Dashboard > Manage Jenkins > Nodes'. The main heading is 'Manage nodes and clouds'. On the left, there are links for 'New Node', 'Configure Clouds', and 'Node Monitoring'. Below these are sections for 'Build Queue' (showing 'No builds in the queue') and 'Build Executor Status'. The 'Build Executor Status' section shows two nodes: 'Built-in Node' and 'jenkins-slave-linux-172-31-48-185'. The 'jenkins-slave-linux-172-31-48-185' node is shown with a table of its status: 'last checked', '12 min', '12 min', '12 min', '12 min', '12 min', and '12 min'. At the bottom right, there is a 'REST API' link and 'Jenkins 2.3' version indicator.

S	Name	Architecture	Cloud	Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	Built-in Node	Linux (amd64)	In sync		6.98 GB	0 B	6.98 GB	0ms
	jenkins-slave-linux-172-31-48-185	Linux (amd64)	In sync		7.43 GB	0 B	7.43 GB	53ms
	last checked	12 min	12 min	12 min	12 min	12 min	12 min	12 min

REST API Jenkins 2.3