- Kavya Sharma

Rayat Bahra University

Design and Deploy a High Availability Scalable Infrastructure.

Project description:

This project builds an infrastructure that automatically scales based on CPU utilization. It includes a custom VPC with two public subnets containing web servers- one for student logins and another for faculty access.

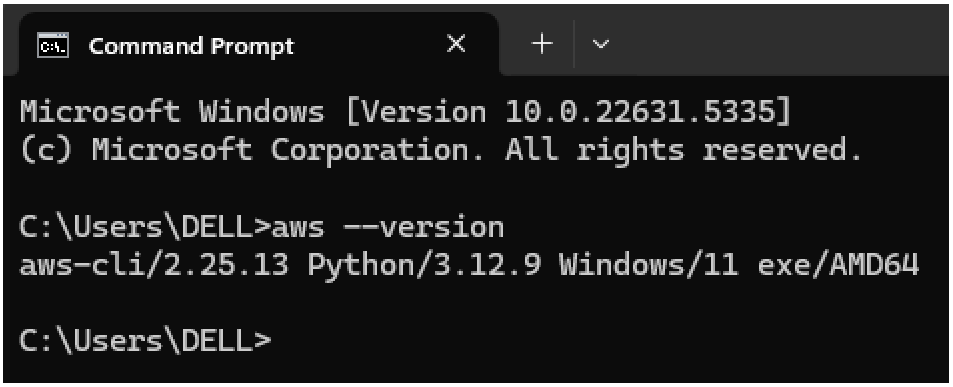
An application load balancer will distribute traffic between these servers while auto-scaling launches new instances when CPU usage exceeds 65%. CloudWatch alarms and SNS notifications will send email alerts whenever scaling events happen.

The entire infrastructure will be built using AWS CLI commands.

STEP 1: Install AWS CLI

For windows: https://awscli.amazonaws.com/AWSCLIV2.msi
1. Download the AWS CLI MSI installer
2. Run the downloaded MSI installer and follow the on-screen instructions

To verify installation
Command: `aws --version`



```
Microsoft Windows [Version 10.0.22631.5335]
(c) Microsoft Corporation. All rights reserved.

C:\Users\DELL>aws --version
aws-cli/2.25.13 Python/3.12.9 Windows/11 exe/AMD64

C:\Users\DELL>
```

AWS CLI has been installed successfully.

- Kavya Sharma
Rayat Bahra University

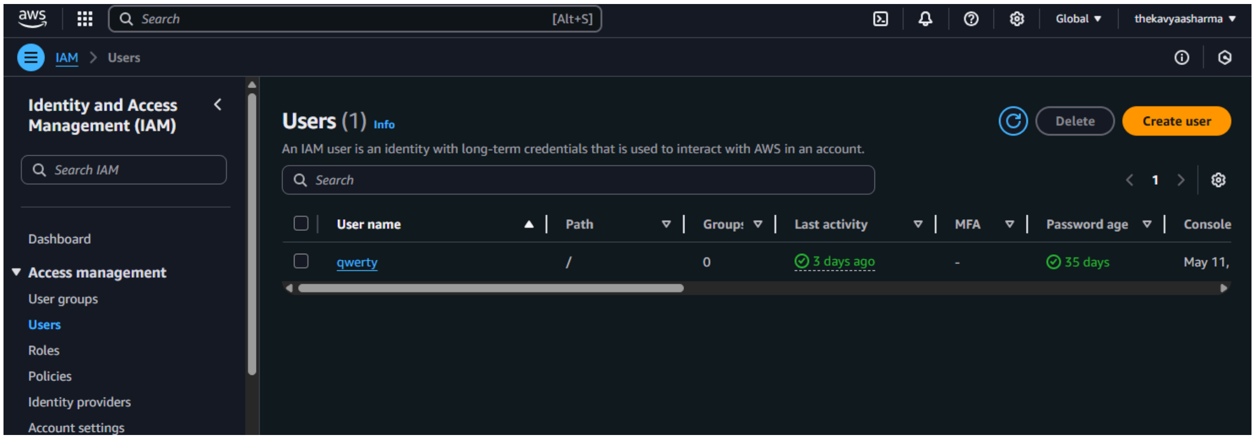
STEP 2: Create an IAM user

Login to AWS management console https://console.aws.amazon.com
Navigate to IAM services :

- In the search bar type “IAM” then select **IAM**

Create a new user:

- Click "**Users**" on the left side, then "Create user"

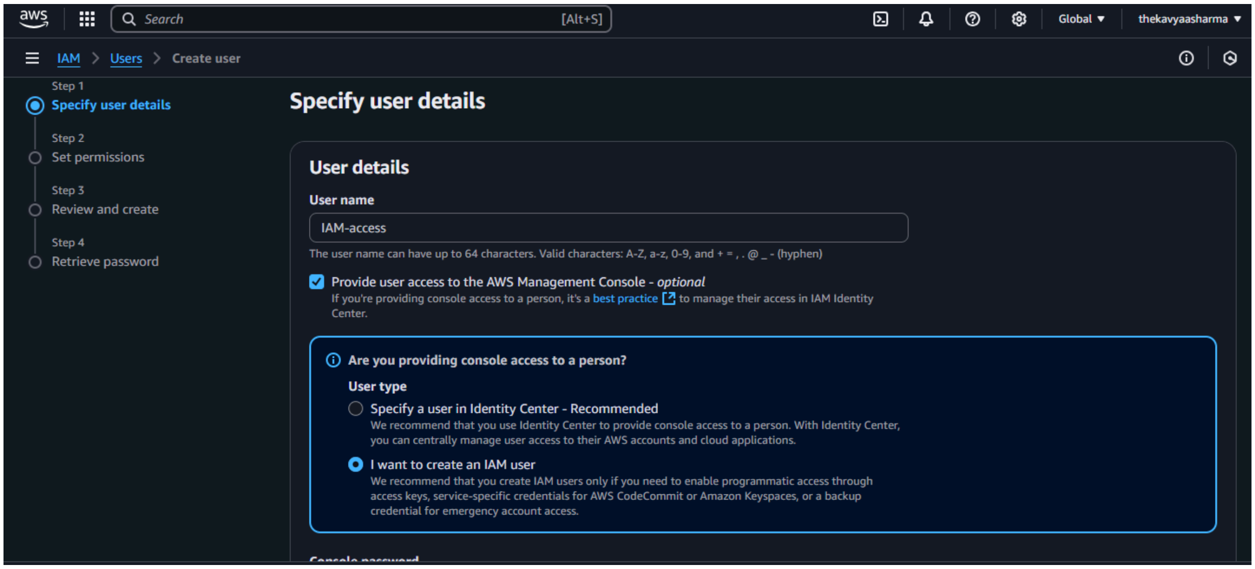


Users (1) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

User name	Path	Groups	Last activity	MFA	Password age	Console
qwerty	/	0	3 days ago	-	35 days	May 11,

- Enter a username (eg: IAM-access)
- Select “Programmatic access” > “I want to create an IAM user”



Specify user details

User details

User name: IAM-access

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)

☑ Provide user access to the AWS Management Console - *optional*
If you're providing console access to a person, it's a best practice to manage their access in IAM Identity Center.

Are you providing console access to a person?

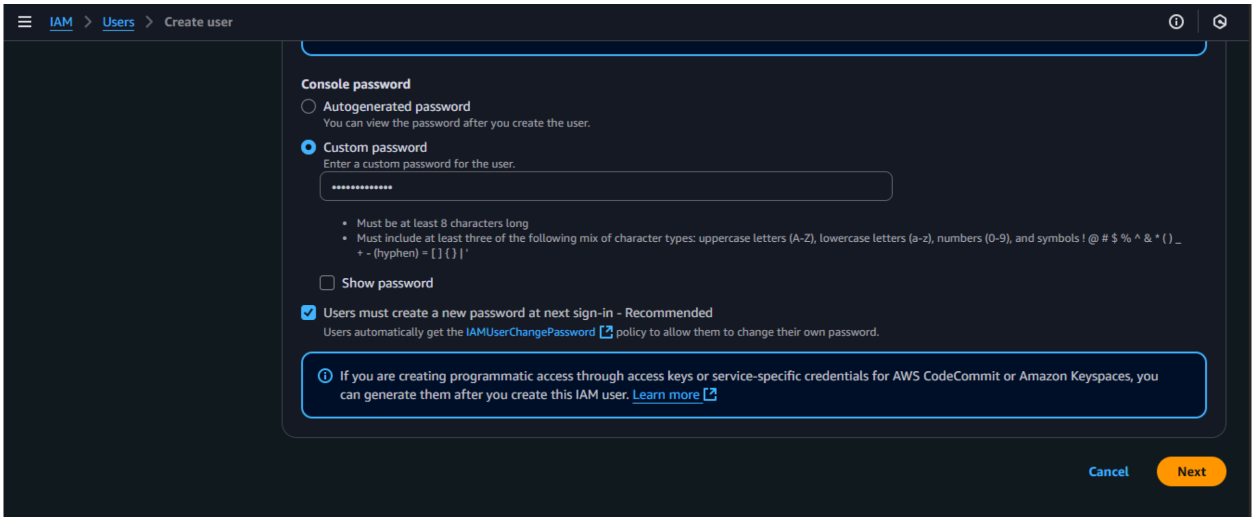
User type

○ Specify a user in Identity Center - Recommended
We recommend that you use Identity Center to provide console access to a person. With Identity Center, you can centrally manage user access to their AWS accounts and cloud applications.

● I want to create an IAM user
We recommend that you create IAM users only if you need to enable programmatic access through access keys, service-specific credentials for AWS CodeCommit or Amazon Keyspaces, or a backup credential for emergency account access.

Console password

- Create a custom password > select option “create a new password at next sign in”
- Go to Next



Console password

○ Autogenerated password
You can view the password after you create the user.

● Custom password
Enter a custom password for the user.

•••••••••••••

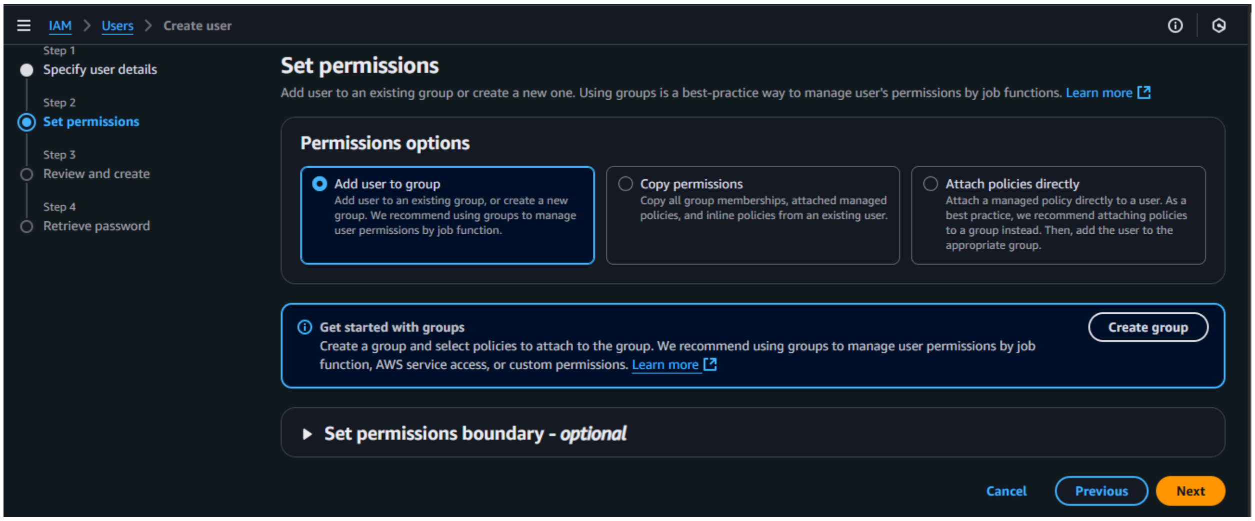
- Must be at least 8 characters long
- Must include at least three of the following mix of character types: uppercase letters (A-Z), lowercase letters (a-z), numbers (0-9), and symbols ! @ # $ % ^ & * ( ) _ + - (hyphen) = [ ] { } | '

□ Show password

☑ Users must create a new password at next sign-in - Recommended
Users automatically get the IAMUserChangePassword policy to allow them to change their own password.

If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. Learn more

- Select “Add user to group”
- Click on “Create group”



Set permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. Learn more

Permissions options

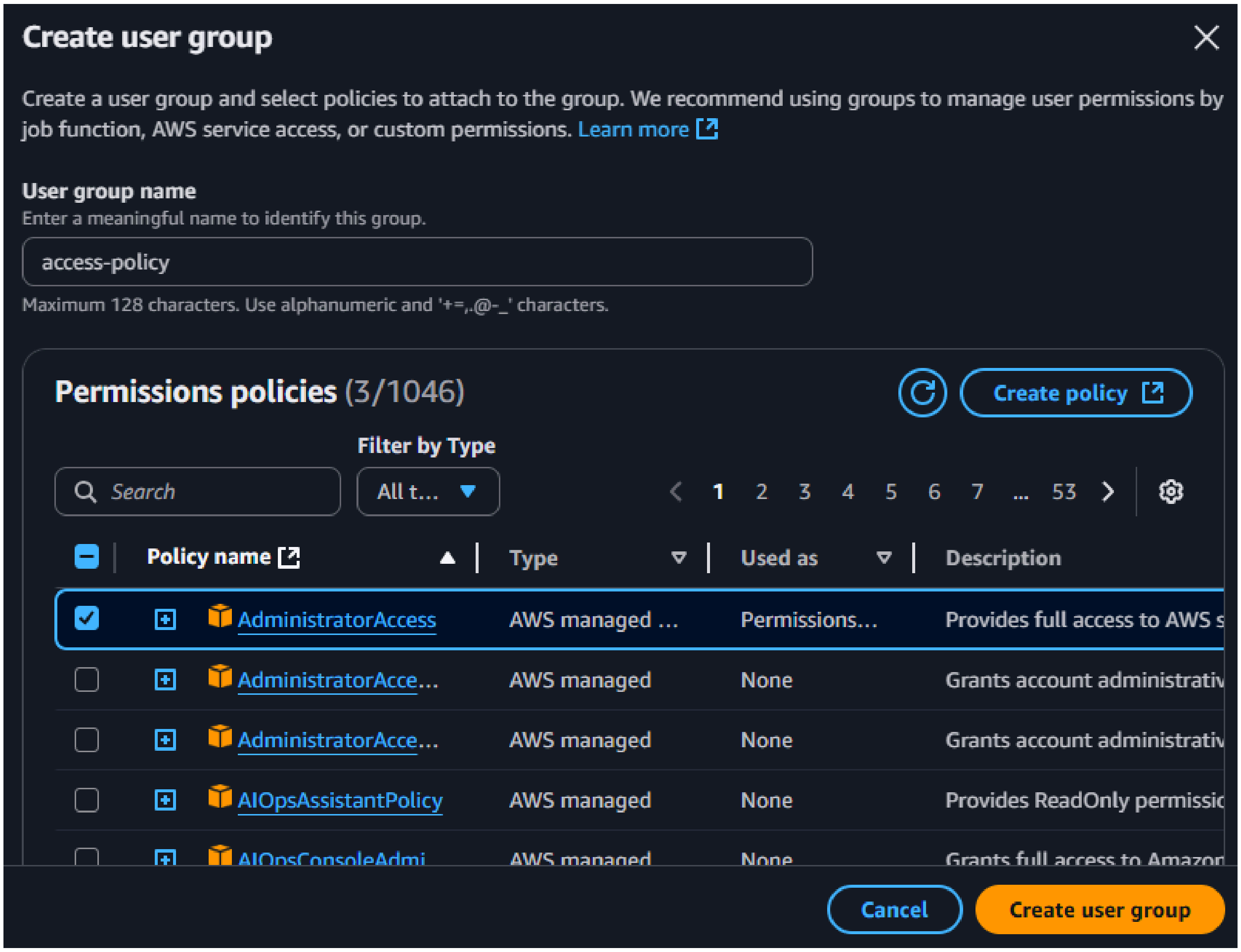
● Add user to group
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.

○ Copy permissions
Copy all group memberships, attached managed policies, and inline policies from an existing user.

○ Attach policies directly
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

Get started with groups
Create a group and select policies to attach to the group. We recommend using groups to manage user permissions by job function, AWS service access, or custom permissions. Learn more

- Add a user group name (eg: access-policy)
- For permissions, attach the "**AdministratorAccess**", "**AmazonVPCFullAccess**", “**AmazonEC2FullAccess**” policy
- Click “Create user group” > NEXT



Create user group

Create a user group and select policies to attach to the group. We recommend using groups to manage user permissions by job function, AWS service access, or custom permissions. Learn more

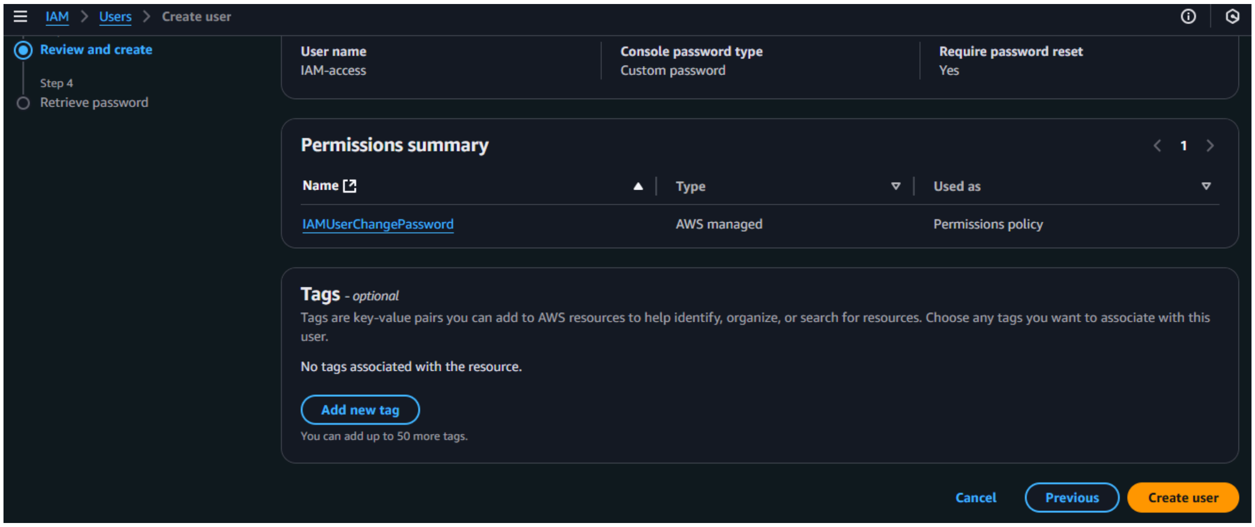
User group name
Enter a meaningful name to identify this group.
access-policy
Maximum 128 characters. Use alphanumeric and '+=,.@-_' characters.

Permissions policies (3/1046)

Filter by Type: All t...

	Policy name	Type	Used as	Description
☑	AdministratorAccess	AWS managed ...	Permissions...	Provides full access to AWS s
□	AdministratorAcce...	AWS managed	None	Grants account administrativ
□	AdministratorAcce...	AWS managed	None	Grants account administrativ
□	AIOpsAssistantPolicy	AWS managed	None	Provides ReadOnly permissic
□	AIOpsConsoleAdmi	AWS managed	None	Grants full access to Amazon

- After reviewing your choices, Click "Create user"



User name	Console password type	Require password reset
IAM-access	Custom password	Yes

Permissions summary

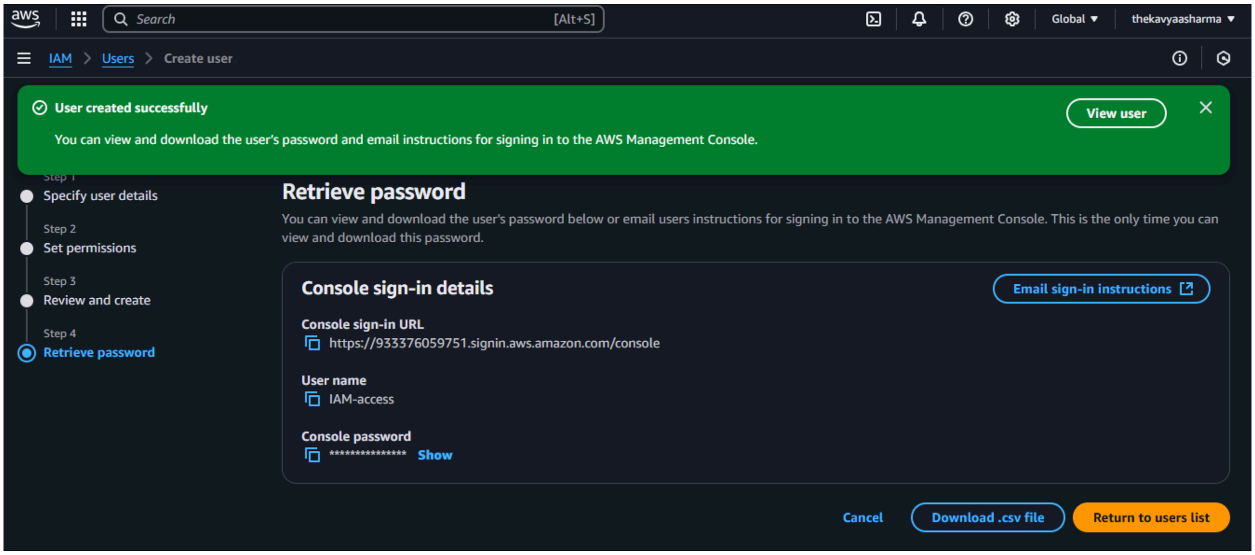
Name	Type	Used as
IAMUserChangePassword	AWS managed	Permissions policy

Tags - *optional*
Tags are key-value pairs you can add to AWS resources to help identify, organize, or search for resources. Choose any tags you want to associate with this user.

No tags associated with the resource.

You can add up to 50 more tags.

- Download the .csv file, copy and save the console sign in details
- Click "Return to user list "



User created successfully

You can view and download the user's password and email instructions for signing in to the AWS Management Console.

Retrieve password

You can view and download the user's password below or email users instructions for signing in to the AWS Management Console. This is the only time you can view and download this password.

Console sign-in details

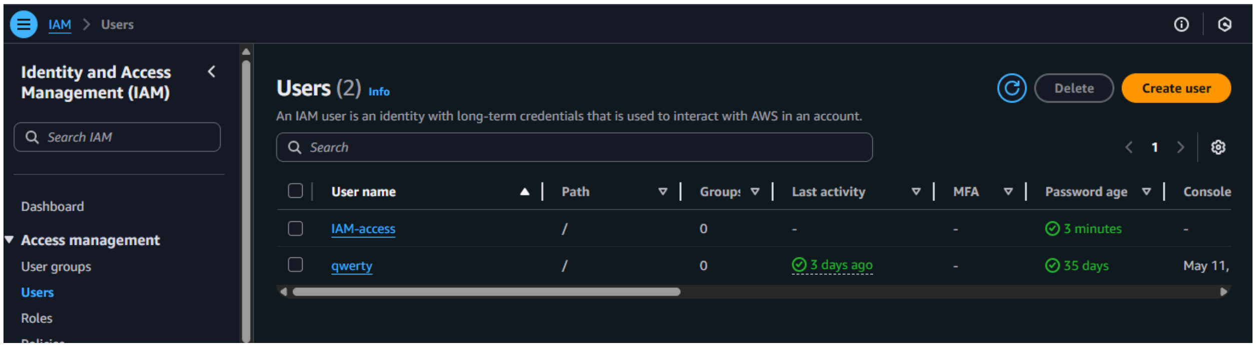
Console sign-in URL
https://933376059751.signin.aws.amazon.com/console

User name
IAM-access

Console password
*************** Show

IAM user with Administrator access is successfully created.

Click on "**IAM-access**"

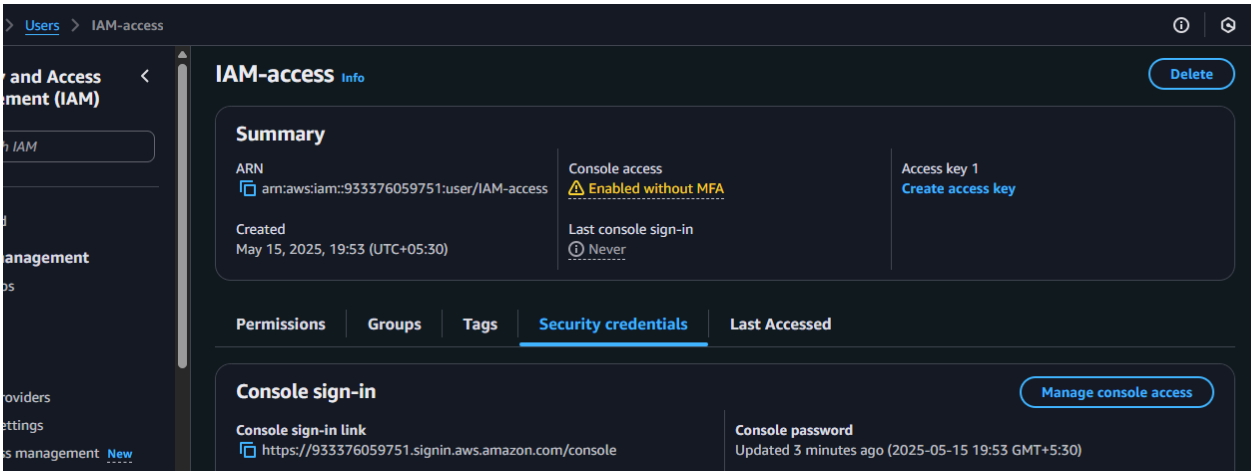


Users (2) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

☐	User name	Path	Groups	Last activity	MFA	Password age	Console
☐	IAM-access	/	0	-	-	✓ 3 minutes	-
☐	qwerty	/	0	✓ 3 days ago	-	✓ 35 days	May 11,

Go to "**Security credentials**"



IAM-access Info

Summary

ARN
arn:aws:iam::933376059751:user/IAM-access

Created
May 15, 2025, 19:53 (UTC+05:30)

Console access
⚠ Enabled without MFA

Last console sign-in
ⓘ Never

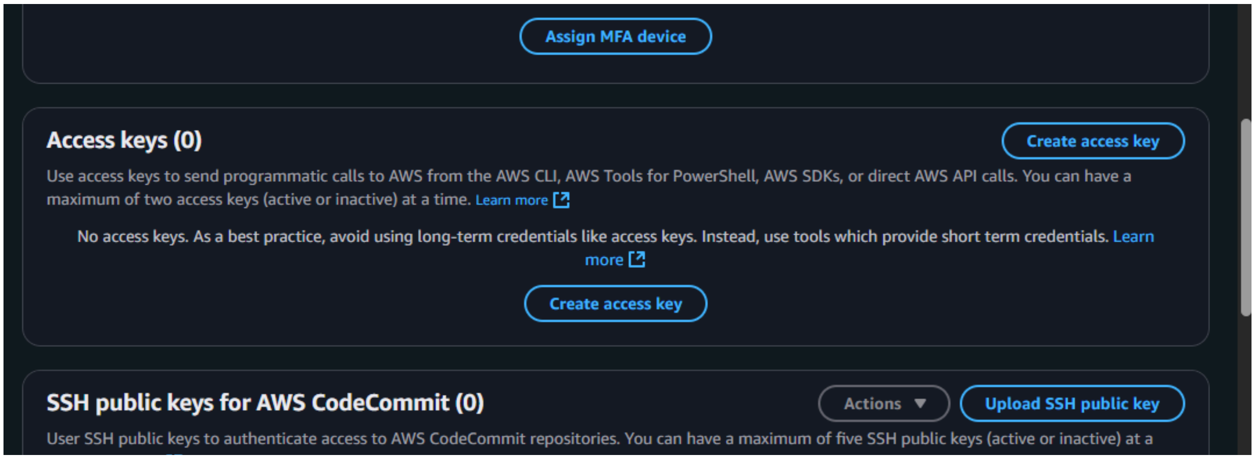
Access key 1
Create access key

Console sign-in

Console sign-in link
https://933376059751.signin.aws.amazon.com/console

Console password
Updated 3 minutes ago (2025-05-15 19:53 GMT+5:30)

Now drop down to the Access keys, and click on "create access key"



Access keys (0)

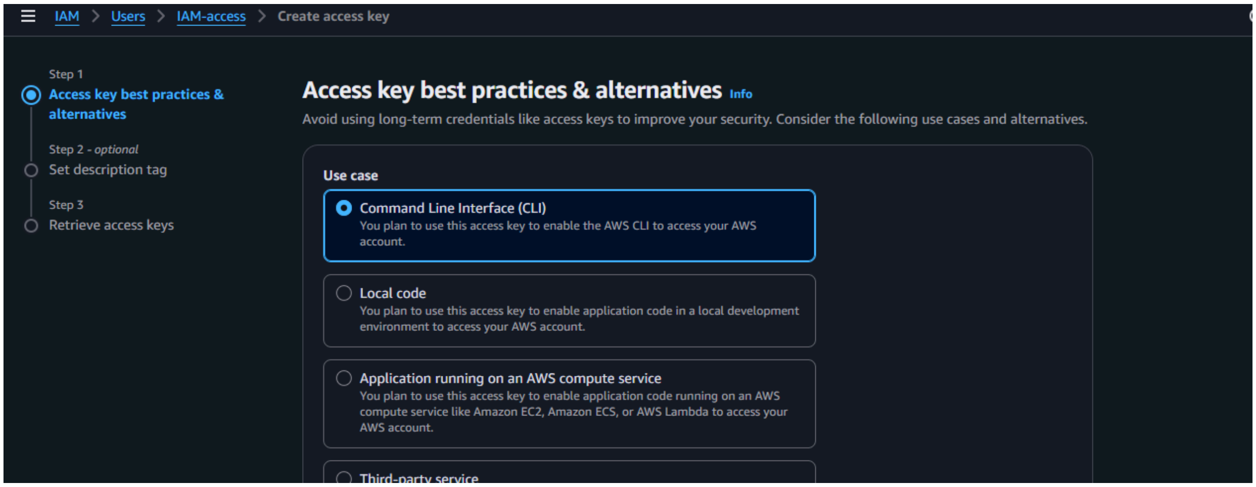
Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. Learn more

No access keys. As a best practice, avoid using long-term credentials like access keys. Instead, use tools which provide short term credentials. Learn more

SSH public keys for AWS CodeCommit (0)

User SSH public keys to authenticate access to AWS CodeCommit repositories. You can have a maximum of five SSH public keys (active or inactive) at a

Select “Command Line Interface(CLI)” > check the confirmation box > go to next


IAM > Users > IAM-access > Create access key
Step 1
Access key best practices & alternatives
Step 2 - optional
Set description tag
Step 3
Retrieve access keys
Info

Access key best practices & alternatives

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

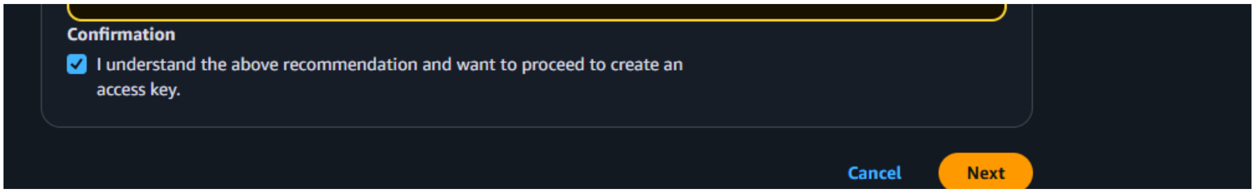
Use case

● Command Line Interface (CLI)
You plan to use this access key to enable the AWS CLI to access your AWS account.

○ Local code
You plan to use this access key to enable application code in a local development environment to access your AWS account.

○ Application running on an AWS compute service
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

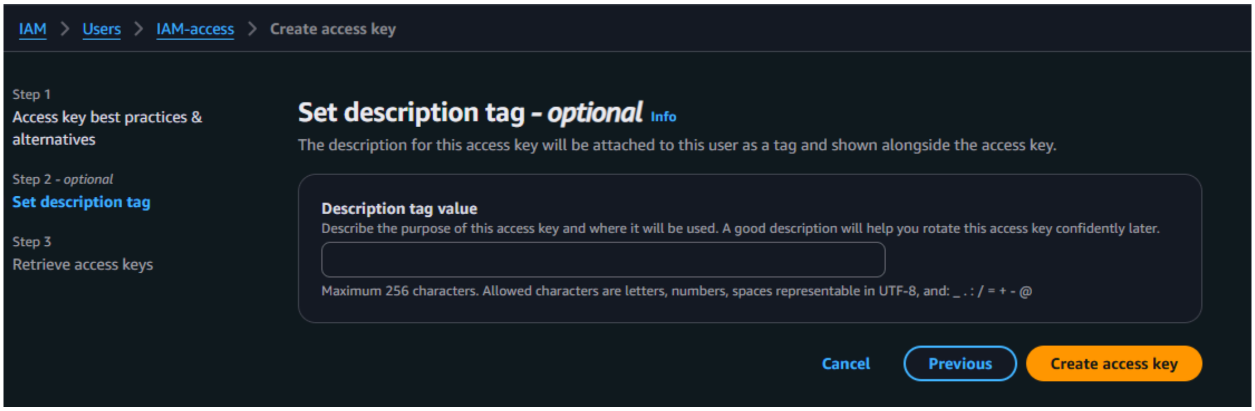
○ Third-party service



Confirmation

☑ I understand the above recommendation and want to proceed to create an access key.

Click on “Create access key”



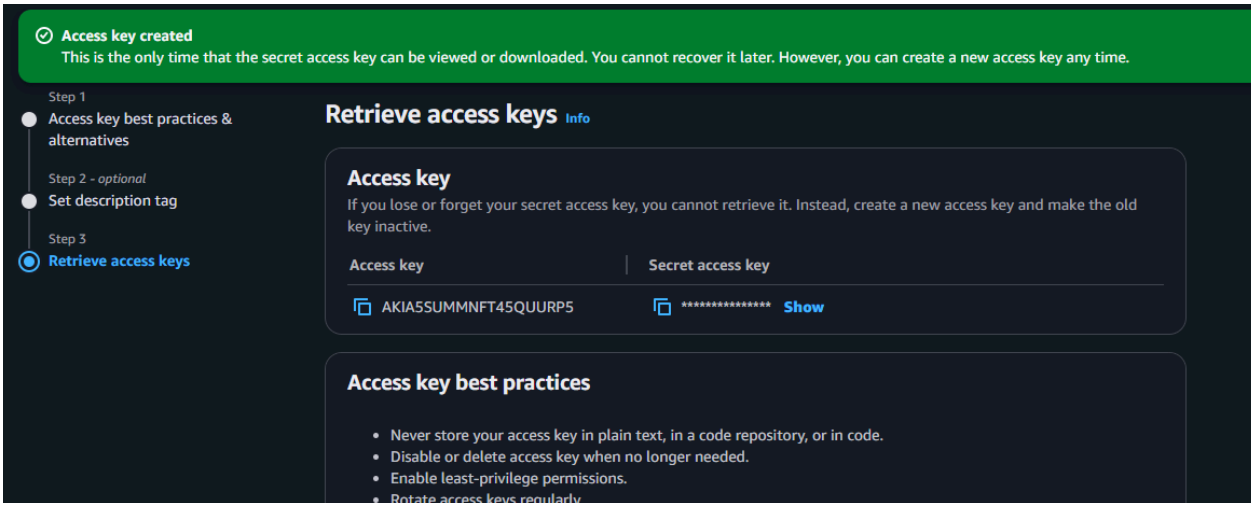
Set description tag - *optional*

The description for this access key will be attached to this user as a tag and shown alongside the access key.

Description tag value
Describe the purpose of this access key and where it will be used. A good description will help you rotate this access key confidently later.

Maximum 256 characters. Allowed characters are letters, numbers, spaces representable in UTF-8, and: _ . : / = + - @

Copy and save the access key and secret key.



Access key created
This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.

Retrieve access keys

Access key

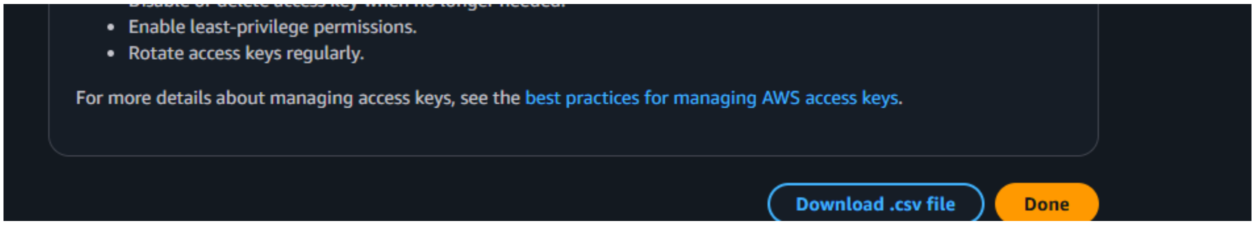
If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key	Secret access key
AKIA5SUMMNFT45QUURP5	*************** Show

Access key best practices

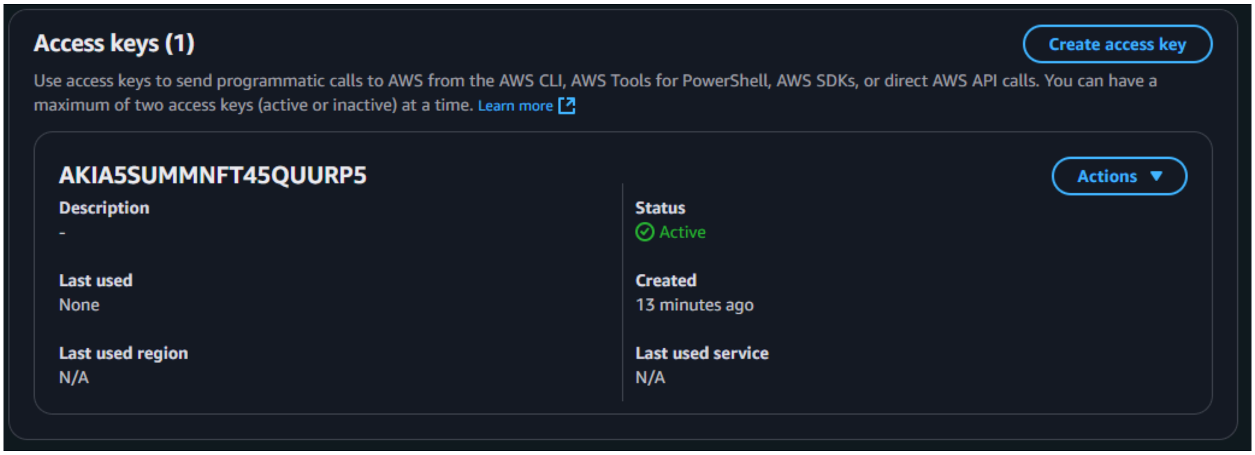
- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

Drop down to download .csv file and click on “done”



- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the best practices for managing AWS access keys.



Access keys (1)

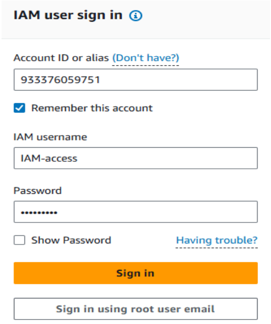
Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. Learn more

AKIA5SUMMNFT45QUURP5

Description -	**Status** ✓ Active
Last used None	**Created** 13 minutes ago
Last used region N/A	**Last used service** N/A

Now the access key has been created successfully.

Sign out from the root user and again sign in to the AWS console using IAM user credentials and update the password.



IAM user sign in

Account ID or alias (Don't have?)
933376059751

☑ Remember this account

IAM username
IAM-access

Password
•••••••••

☐ Show Password

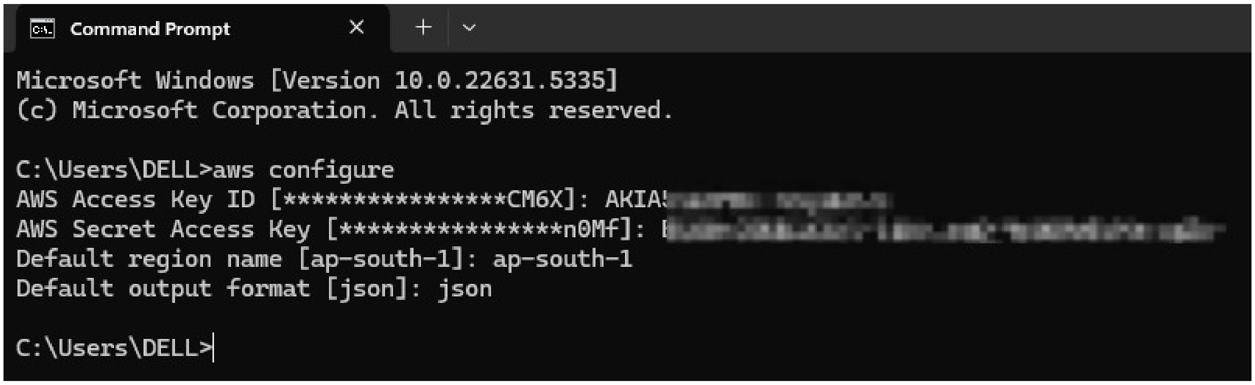
STEP 3: Configure AWS CLI

Run the following command on the command prompt and enter your credentials when requested:

- `aws configure` < press ENTER

Fill in your credentials:

- `AWS Access Key ID`: paste your access key from step 2
- `AWS Secret Access Key`: paste your secret key from step 2
- `Default region name`: e.g ap-south-1
- `Default output format`: json

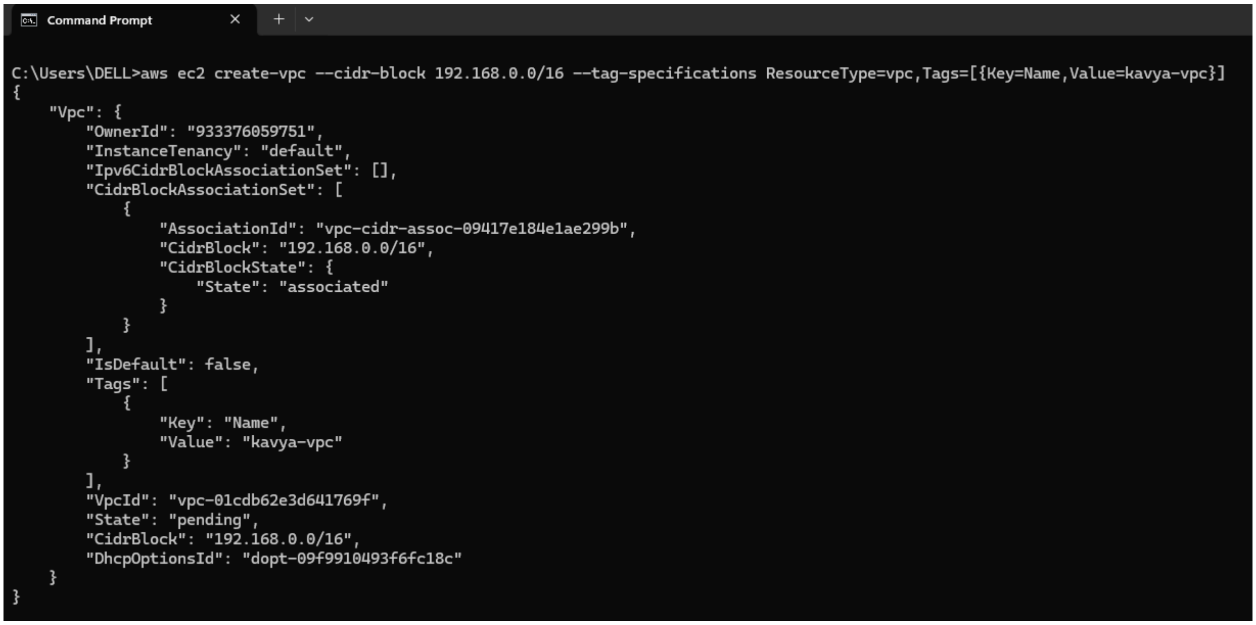


```
Microsoft Windows [Version 10.0.22631.5335]
(c) Microsoft Corporation. All rights reserved.

C:\Users\DELL>aws configure
AWS Access Key ID [****************CM6X]: AKIA[redacted]
AWS Secret Access Key [****************n0Mf]: [redacted]
Default region name [ap-south-1]: ap-south-1
Default output format [json]: json

C:\Users\DELL>
```

STEP 4: Create a custom VPC

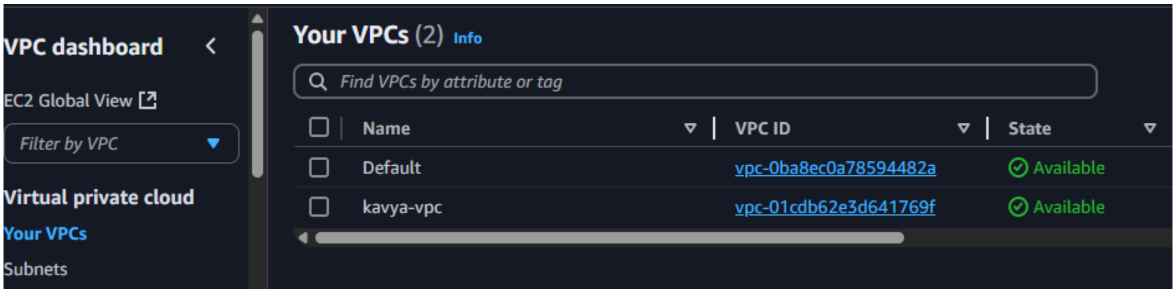
Command:

```
     aws ec2 create-vpc
          --cidr-block <IPv4 network range(eg. 192.168.0.0/16) >
          --tag-specifications
          ResourceType=vpc,Tags[{Keys=Name,Value=YourName-vpc}]
```



```
C:\Users\DELL>aws ec2 create-vpc --cidr-block 192.168.0.0/16 --tag-specifications ResourceType=vpc,Tags=[{Key=Name,Value=kavya-vpc}]
{
    "Vpc": {
        "OwnerId": "933376059751",
        "InstanceTenancy": "default",
        "Ipv6CidrBlockAssociationSet": [],
        "CidrBlockAssociationSet": [
            {
                "AssociationId": "vpc-cidr-assoc-09417e184e1ae299b",
                "CidrBlock": "192.168.0.0/16",
                "CidrBlockState": {
                    "State": "associated"
                }
            }
        ],
        "IsDefault": false,
        "Tags": [
            {
                "Key": "Name",
                "Value": "kavya-vpc"
            }
        ],
        "VpcId": "vpc-01cdb62e3d641769f",
        "State": "pending",
        "CidrBlock": "192.168.0.0/16",
        "DhcpOptionsId": "dopt-09f9910493f6fc18c"
    }
}
```

Go to AWS console > navigate to VPC dashboard > on left side click “Your VPCs”
Verify that the VPC is created successfully .

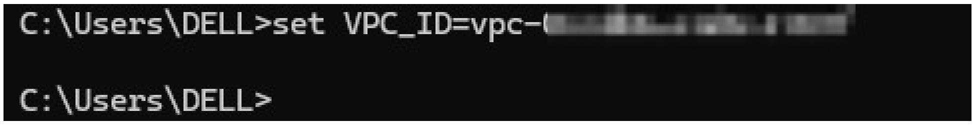


Your VPCs (2)

	Name	VPC ID	State
□	Default	vpc-0ba8ec0a78594482a	Available
□	kavya-vpc	vpc-01cdb62e3d641769f	Available

Save the VPC ID returned from this command for later use:

- Copy the vpc id from the output
- `set VPC_ID=<paste your vpc id as vpc-xxxxxxxxxx` > ,then press ENTER

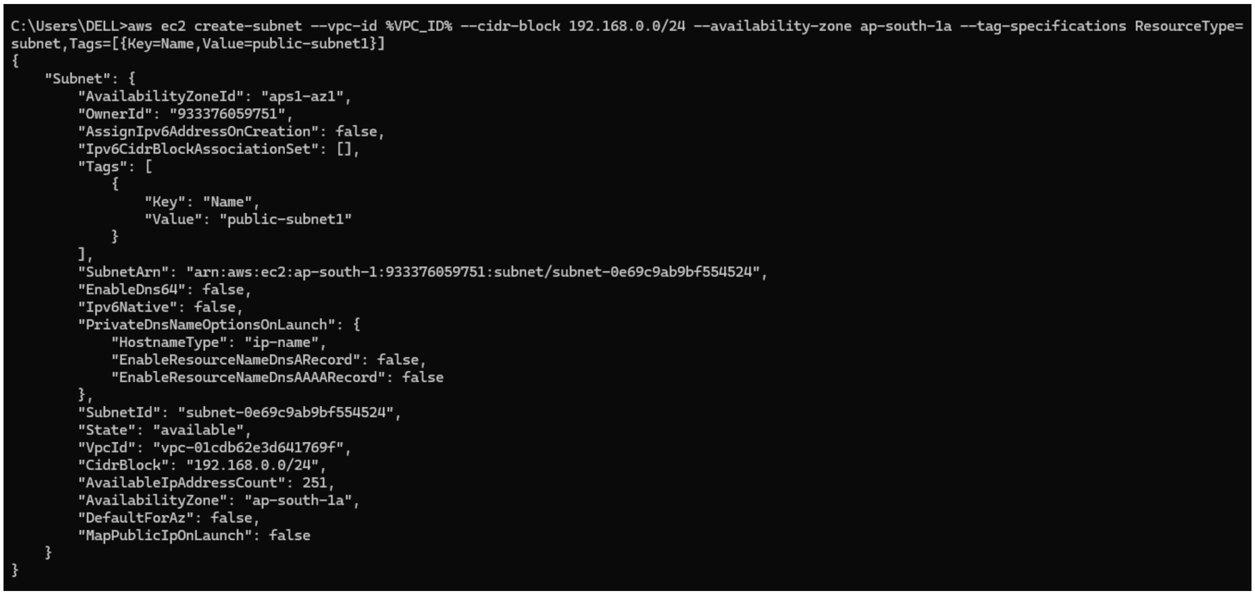


```
C:\Users\DELL>set VPC_ID=vpc-[redacted]

C:\Users\DELL>
```

Step 5: Create two public subnets

Command: `aws ec2 create-subnet`

```
                --vpc-id %VPC_ID%
                --cidr-block <eg 192.168.0.0/24>
                 --availability-zone <eg ap-south-1a>
                --tag-specifications
                ResourceType=subnet,Tags=[{Key=Name,Value=SubnetName}]
```

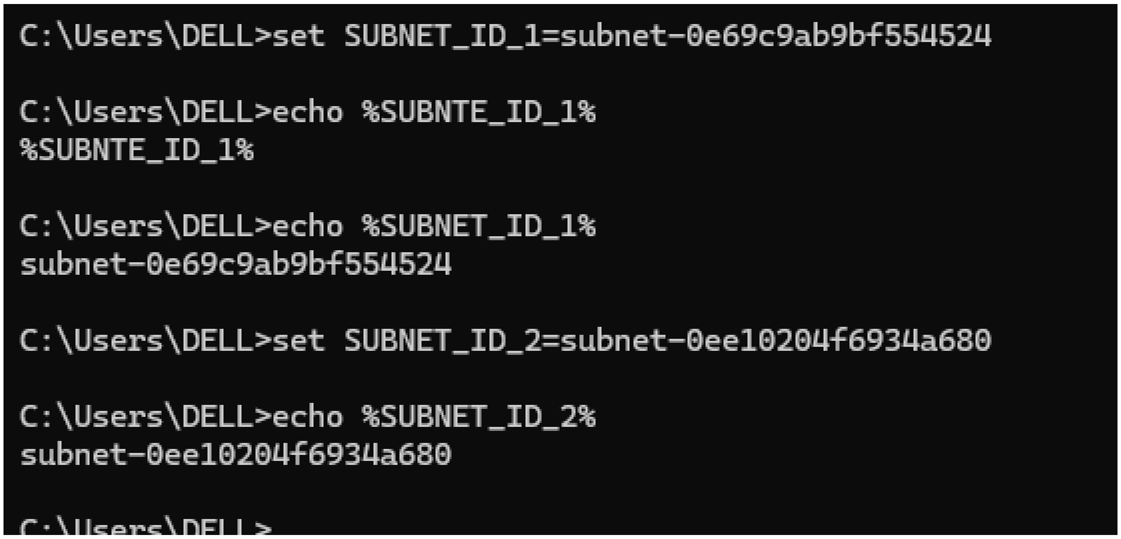


```
C:\Users\DELL>aws ec2 create-subnet --vpc-id %VPC_ID% --cidr-block 192.168.0.0/24 --availability-zone ap-south-1a --tag-specifications ResourceType=
subnet,Tags=[{Key=Name,Value=public-subnet1}]
{
    "Subnet": {
        "AvailabilityZoneId": "aps1-az1",
        "OwnerId": "933376059751",
        "AssignIpv6AddressOnCreation": false,
        "Ipv6CidrBlockAssociationSet": [],
        "Tags": [
            {
                "Key": "Name",
                "Value": "public-subnet1"
            }
        ],
        "SubnetArn": "arn:aws:ec2:ap-south-1:933376059751:subnet/subnet-0e69c9ab9bf554524",
        "EnableDns64": false,
        "Ipv6Native": false,
        "PrivateDnsNameOptionsOnLaunch": {
            "HostnameType": "ip-name",
            "EnableResourceNameDnsARecord": false,
            "EnableResourceNameDnsAAAARecord": false
        },
        "SubnetId": "subnet-0e69c9ab9bf554524",
        "State": "available",
        "VpcId": "vpc-01cdb62e3d641769f",
        "CidrBlock": "192.168.0.0/24",
        "AvailableIpAddressCount": 251,
        "AvailabilityZone": "ap-south-1a",
        "DefaultForAz": false,
        "MapPublicIpOnLaunch": false
    }
}
```

Repeat the same command for the next subnet with different CIDR block to avoid overlapping and in different availability zones with another name(eg Public-subnet2).

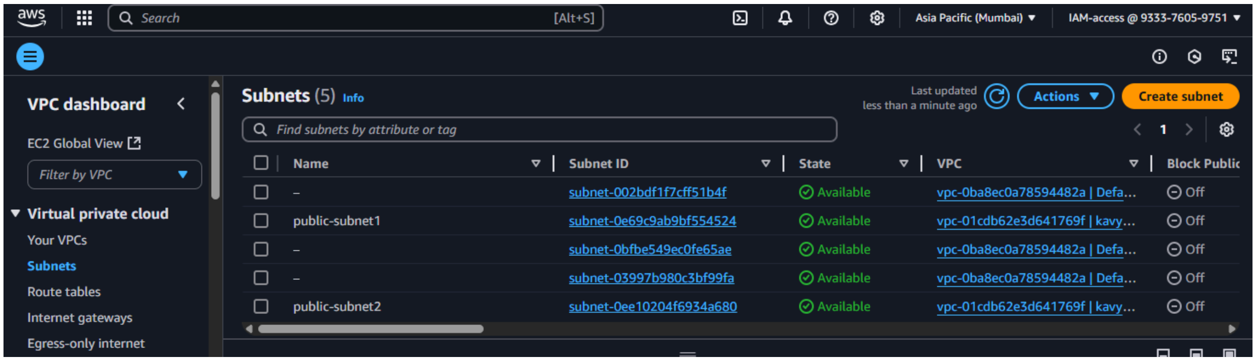
Copy the subnet id from the output and run the command:

```
Set SUBNET_ID_1=subnet-xxxxxxxxxxxx
Set SUBNET_ID_2=subnet-XXXXXXXXX
```



```
C:\Users\DELL>set SUBNET_ID_1=subnet-0e69c9ab9bf554524

C:\Users\DELL>echo %SUBNTE_ID_1%
%SUBNTE_ID_1%

C:\Users\DELL>echo %SUBNET_ID_1%
subnet-0e69c9ab9bf554524

C:\Users\DELL>set SUBNET_ID_2=subnet-0ee10204f6934a680

C:\Users\DELL>echo %SUBNET_ID_2%
subnet-0ee10204f6934a680

C:\Users\DELL>
```

To verify: on the AWS VPC console, click “Subnets” on the left side



Subnets (5)

□	Name	Subnet ID	State	VPC	Block Public
□	–	subnet-002bdf1f7cff51b4f	Available	vpc-0ba8ec0a78594482a | Defa...	Off
□	public-subnet1	subnet-0e69c9ab9bf554524	Available	vpc-01cdb62e3d641769f | kavy...	Off
□	–	subnet-0bfbe549ec0fe65ae	Available	vpc-0ba8ec0a78594482a | Defa...	Off
□	–	subnet-03997b980c3bf99fa	Available	vpc-0ba8ec0a78594482a | Defa...	Off
□	public-subnet2	subnet-0ee10204f6934a680	Available	vpc-01cdb62e3d641769f | kavy...	Off

Both the public subnets are successfully created.

STEP 6: Create and attach Internet Gateway to VPC

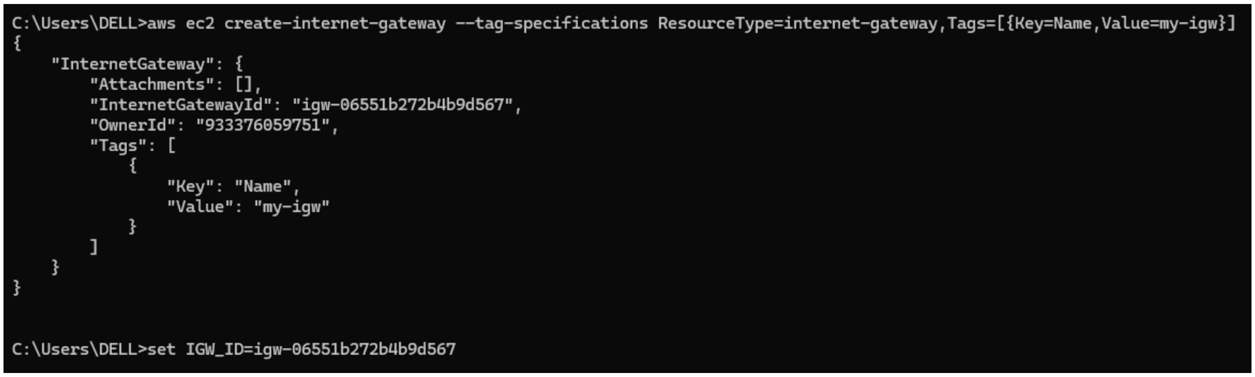
Command to create IGW:

```
     aws ec2  create-internet-gateway
          --tag-specifications
          ResourceType=internet-gateway,Tags=[{Key=Name,Value=my-igw
          }]
```

Copy the internet gateway id from the output

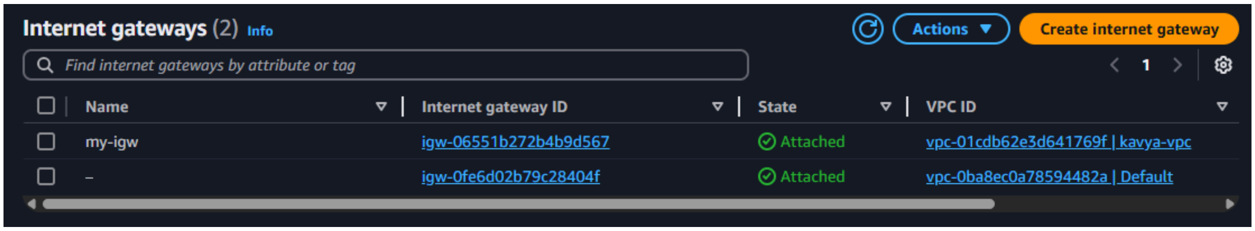
Save the IGW ID:

```
set IGW_ID=igw-xxxxxxxxxxx(paste here)
```



```
C:\Users\DELL>aws ec2 create-internet-gateway --tag-specifications ResourceType=internet-gateway,Tags=[{Key=Name,Value=my-igw}]
{
    "InternetGateway": {
        "Attachments": [],
        "InternetGatewayId": "igw-06551b272b4b9d567",
        "OwnerId": "933376059751",
        "Tags": [
            {
                "Key": "Name",
                "Value": "my-igw"
            }
        ]
    }
}


C:\Users\DELL>set IGW_ID=igw-06551b272b4b9d567
```

Verify on the aws vpc console

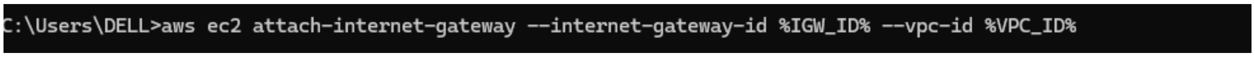


Internet gateways (2)

□	Name	Internet gateway ID	State	VPC ID
□	my-igw	igw-06551b272b4b9d567	✓ Attached	vpc-01cdb62e3d641769f | kavya-vpc
□	–	igw-0fe6d02b79c28404f	✓ Attached	vpc-0ba8ec0a78594482a | Default

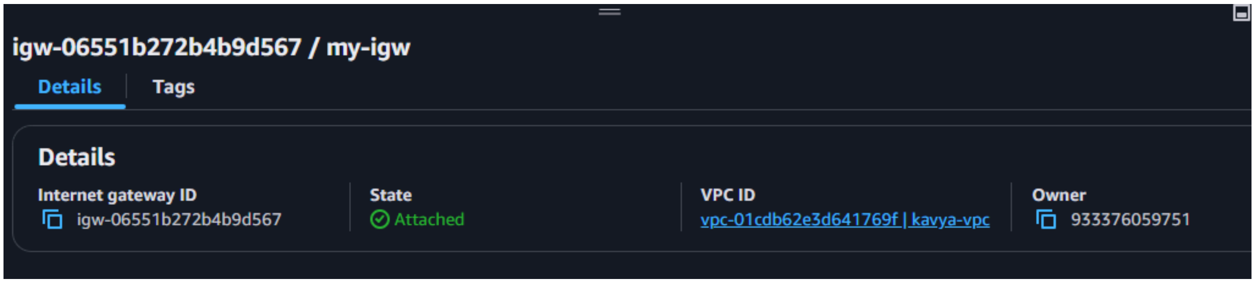
Attach internet gateway to the VPC:

```
    Aws ec2 attach-internet-gateway
        --internet-gateway-id %IGW_ID%
        --vpc-id %VPC_ID%
```



```
:\Users\DELL>aws ec2 attach-internet-gateway --internet-gateway-id %IGW_ID% --vpc-id %VPC_ID%
```

Select the “my-igw” on the console and drop down to verify its attachment



igw-06551b272b4b9d567 / my-igw

Details

Internet gateway ID	State	VPC ID	Owner
igw-06551b272b4b9d567	✓ Attached	vpc-01cdb62e3d641769f | kavya-vpc	933376059751

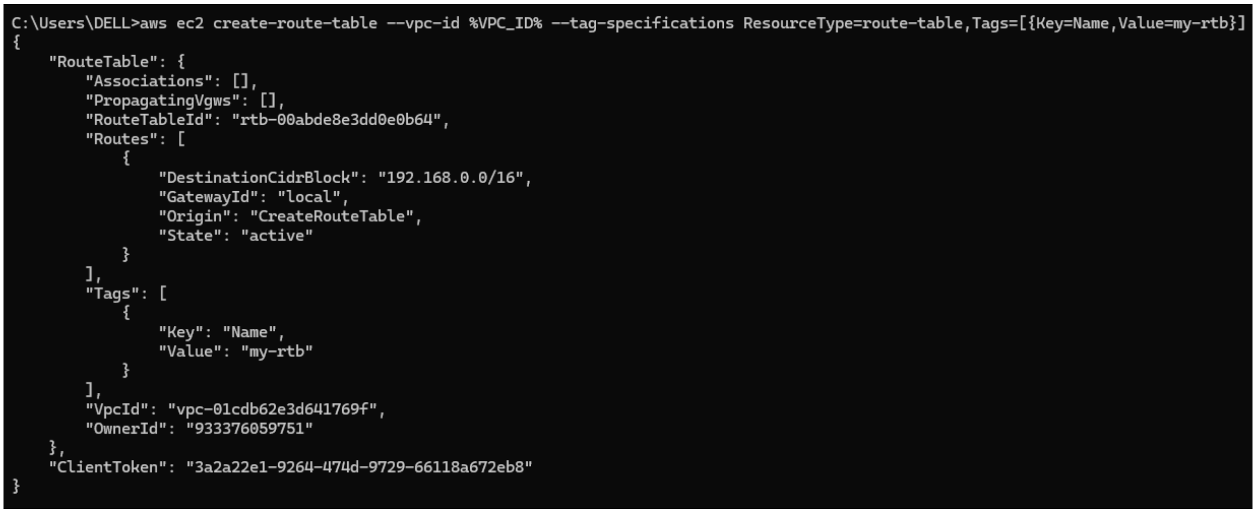
Internet gateway is created and attached to the vpc successfully.

STEP 7: Create and configure route table

Command to create route table:

```
    aws ec2 create-route-table
        --tag-specifications
        ResourceType=route-table,Tags=[{Key-=Name,Value=my-rtb}]
```

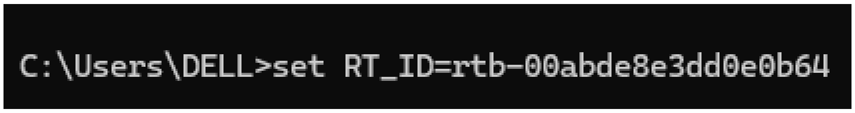
Copy the RouteTableId from the output



```
C:\Users\DELL>aws ec2 create-route-table --vpc-id %VPC_ID% --tag-specifications ResourceType=route-table,Tags=[{Key=Name,Value=my-rtb}]
{
    "RouteTable": {
        "Associations": [],
        "PropagatingVgws": [],
        "RouteTableId": "rtb-00abde8e3dd0e0b64",
        "Routes": [
            {
                "DestinationCidrBlock": "192.168.0.0/16",
                "GatewayId": "local",
                "Origin": "CreateRouteTable",
                "State": "active"
            }
        ],
        "Tags": [
            {
                "Key": "Name",
                "Value": "my-rtb"
            }
        ],
        "VpcId": "vpc-01cdb62e3d641769f",
        "OwnerId": "933376059751"
    },
    "ClientToken": "3a2a22e1-9264-474d-9729-66118a672eb8"
}
```

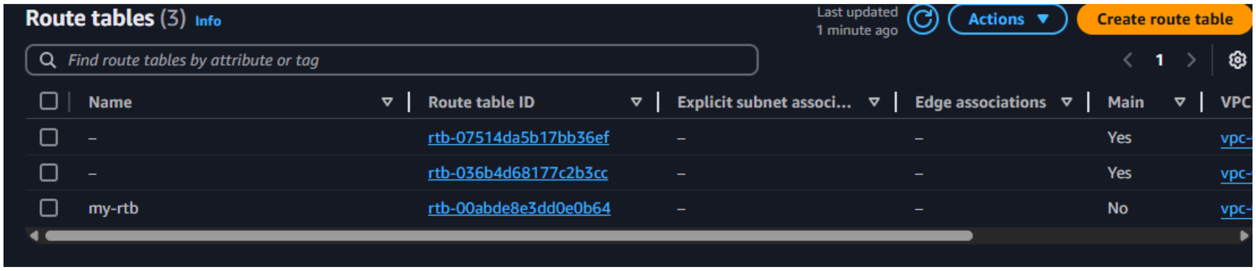
Save the route table id using this command:

```
set RT_ID=rtb-xxxxxxxxxxxx
```



```
C:\Users\DELL>set RT_ID=rtb-00abde8e3dd0e0b64
```

Verify on the AWS VPC console.

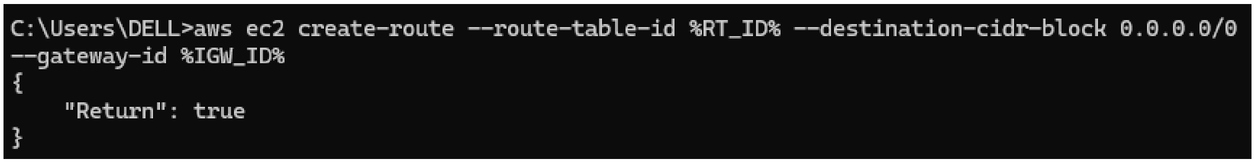


Route tables (3) Info

□	Name	Route table ID	Explicit subnet associ...	Edge associations	Main	VPC
□	–	rtb-07514da5b17bb36ef	–	–	Yes	vpc-
□	–	rtb-036b4d68177c2b3cc	–	–	Yes	vpc-
□	my-rtb	rtb-00abde8e3dd0e0b64	–	–	No	vpc-

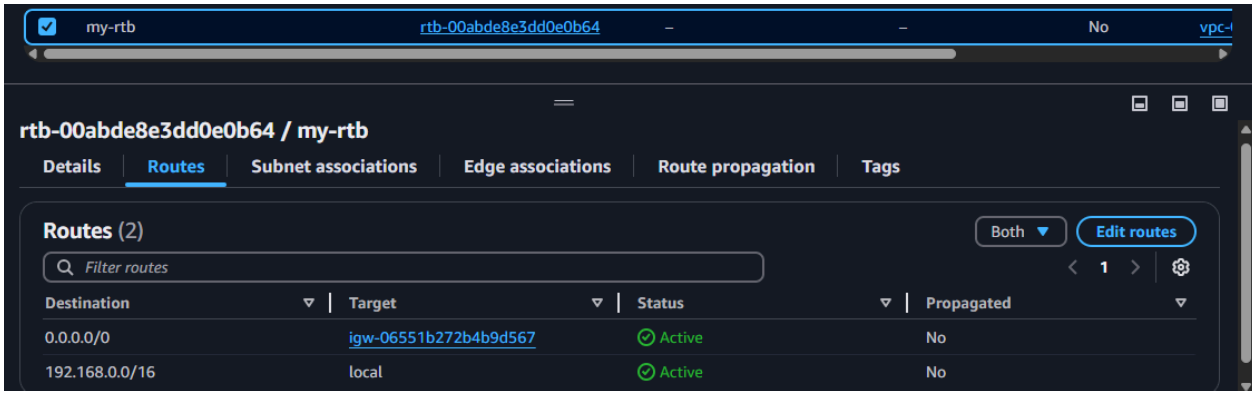
Command to add route to internet gateway:

```
    aws ec2 create-route
            --route-table-id %RT_ID%
            --destination-cidr-block 0.0.0.0/0
            --gateway-id %IGW_ID%
```



```
C:\Users\DELL>aws ec2 create-route --route-table-id %RT_ID% --destination-cidr-block 0.0.0.0/0
--gateway-id %IGW_ID%
{
    "Return": true
}
```

Verify on the AWS VPC console



☑ my-rtb rtb-00abde8e3dd0e0b64 – – No vpc-

rtb-00abde8e3dd0e0b64 / my-rtb

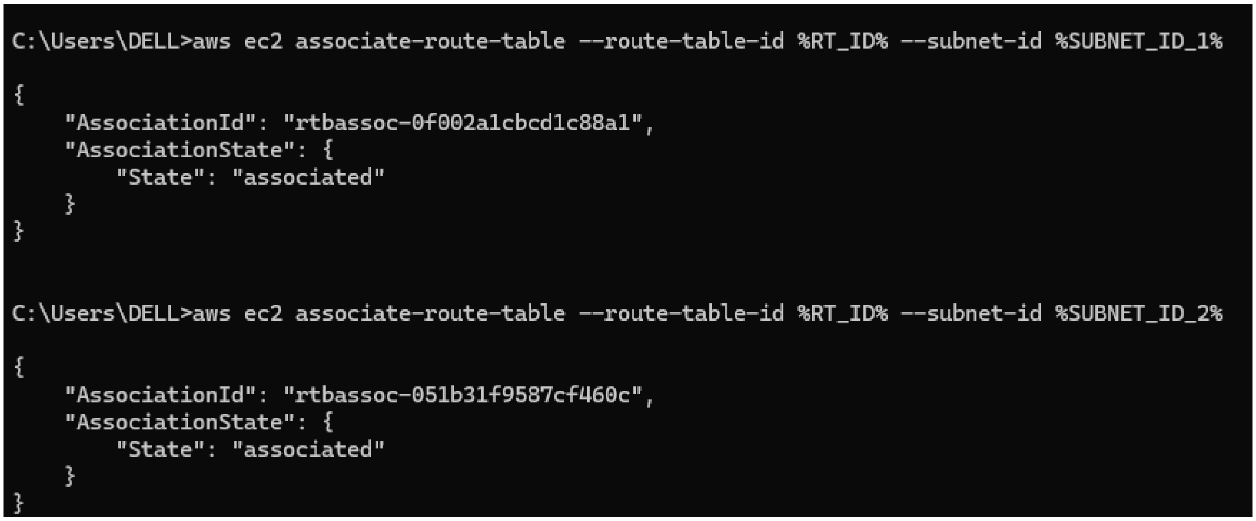
Routes (2)

Destination	Target	Status	Propagated
0.0.0.0/0	igw-06551b272b4b9d567	Active	No
192.168.0.0/16	local	Active	No

Command for subnet association of the route table:

```
aws ec2 associate-route-table
          --route-table-id %RT_ID%
          --subnet-id %SUBNET_ID_1%
```

Repeat for second subnet where subnet id is %SUBNET_ID_2%

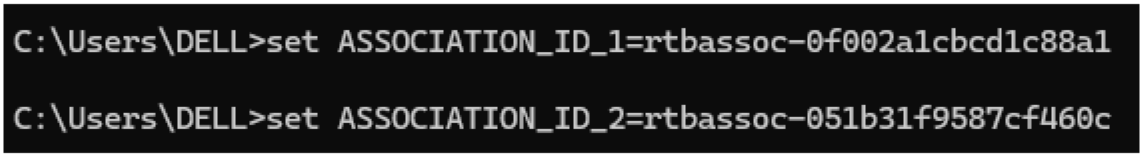


```
C:\Users\DELL>aws ec2 associate-route-table --route-table-id %RT_ID% --subnet-id %SUBNET_ID_1%
{
    "AssociationId": "rtbassoc-0f002a1cbcd1c88a1",
    "AssociationState": {
        "State": "associated"
    }
}


C:\Users\DELL>aws ec2 associate-route-table --route-table-id %RT_ID% --subnet-id %SUBNET_ID_2%
{
    "AssociationId": "rtbassoc-051b31f9587cf460c",
    "AssociationState": {
        "State": "associated"
    }
}
```

Optional: save the association id which can be used for disassociating the route table.

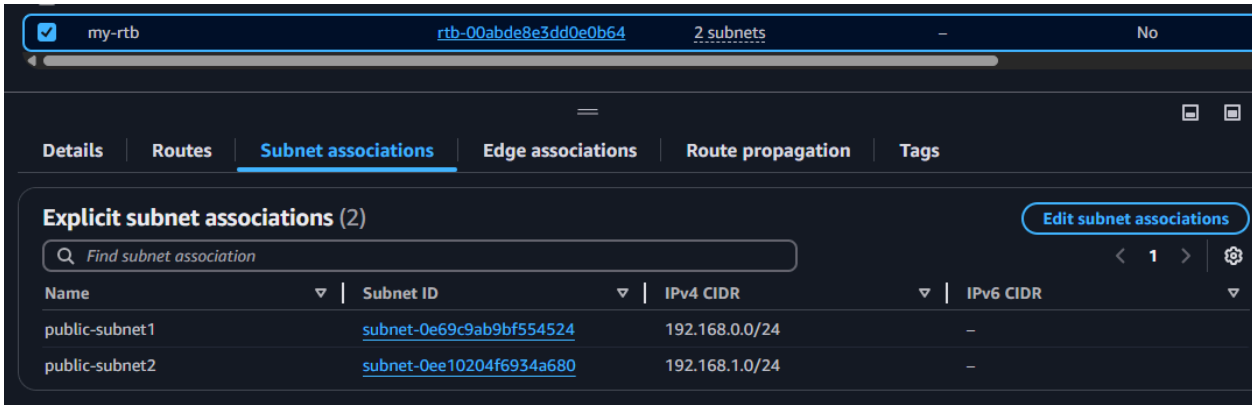
Command to disassociate:

```
aws ec2 disassociate-route-table
        --association-id %ASSOCIATION_ID_1%
```



```
C:\Users\DELL>set ASSOCIATION_ID_1=rtbassoc-0f002a1cbcd1c88a1

C:\Users\DELL>set ASSOCIATION_ID_2=rtbassoc-051b31f9587cf460c
```

Verify on the AWS VPC console



☑ my-rtb	rtb-00abde8e3dd0e0b64	2 subnets	–	No

Explicit subnet associations (2)

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
public-subnet1	subnet-0e69c9ab9bf554524	192.168.0.0/24	–
public-subnet2	subnet-0ee10204f6934a680	192.168.1.0/24	–

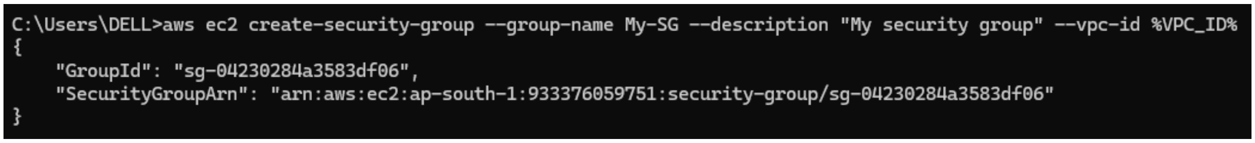
Route table is created and configured successfully.

STEP 8: Create security group

Command to create security group:

```
aws ec2 create-security-group
        --group-name <name of your security group>
        --description “add description”
        --vpc-id %VPC_ID%
```

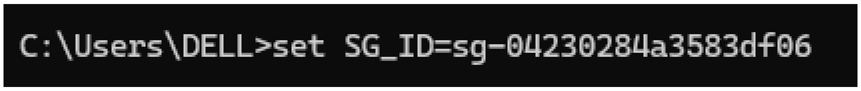
Copy the group id from the output.



```
C:\Users\DELL>aws ec2 create-security-group --group-name My-SG --description "My security group" --vpc-id %VPC_ID%
{
    "GroupId": "sg-04230284a3583df06",
    "SecurityGroupArn": "arn:aws:ec2:ap-south-1:933376059751:security-group/sg-04230284a3583df06"
}
```

Save the security group id:

```
set SG_ID=sg-xxxxxxxxxxxxxx
```

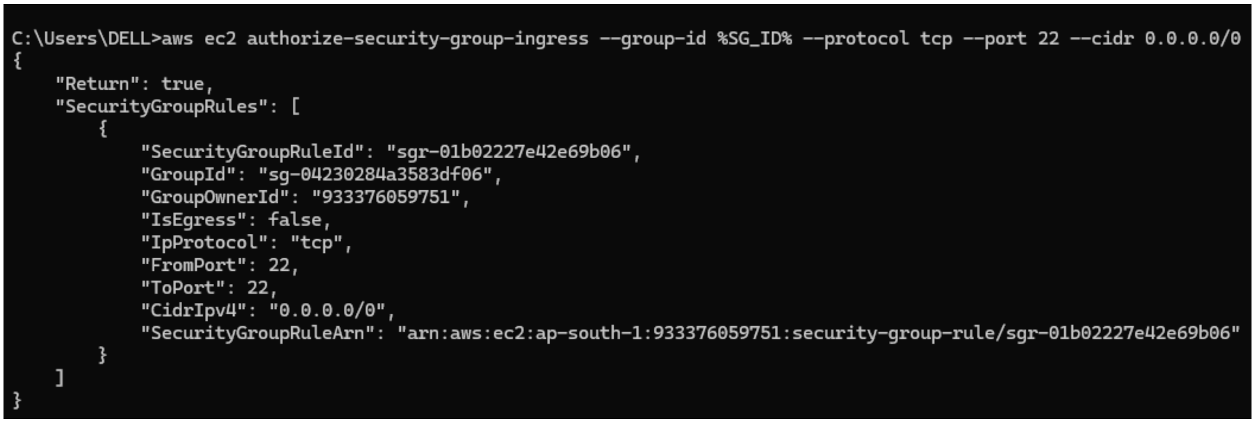


```
C:\Users\DELL>set SG_ID=sg-04230284a3583df06
```

Add inbound rules:

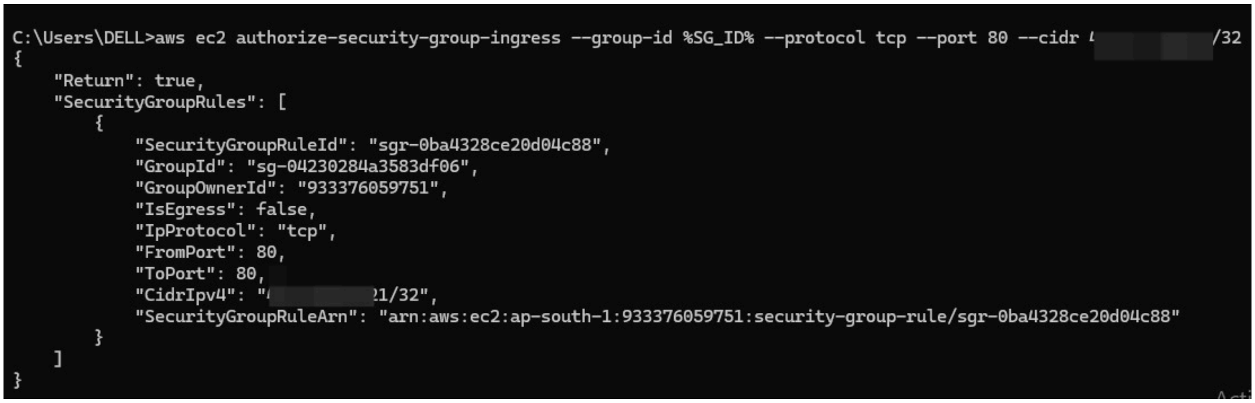
To allow SSH:

```
aws ec2 authorize-security-group
        --group-id  %SG_ID%
        --protocol tcp
        --port 22
        --cidr 0.0.0.0/0
```



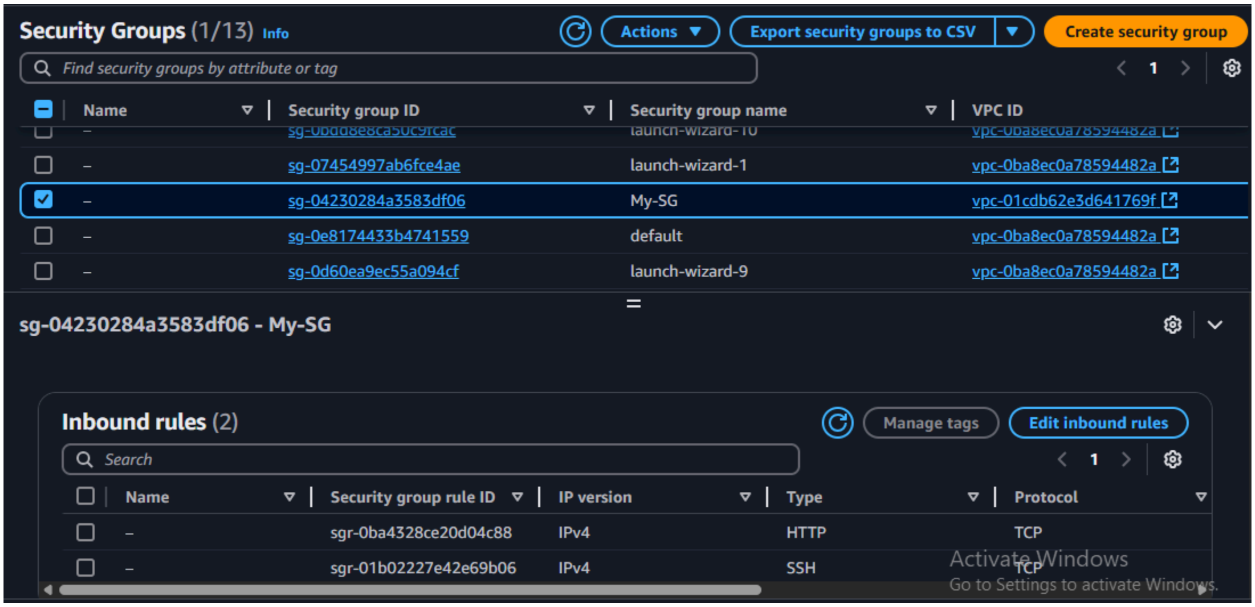
```
C:\Users\DELL>aws ec2 authorize-security-group-ingress --group-id %SG_ID% --protocol tcp --port 22 --cidr 0.0.0.0/0
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-01b02227e42e69b06",
            "GroupId": "sg-04230284a3583df06",
            "GroupOwnerId": "933376059751",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv4": "0.0.0.0/0",
            "SecurityGroupRuleArn": "arn:aws:ec2:ap-south-1:933376059751:security-group-rule/sgr-01b02227e42e69b06"
        }
    ]
}
```

To allow HTTP:
```
aws ec2 authorize-security-group
    --group-id  %SG_ID%
    --protocol tcp
    --port 80
    --cidr <your ip address /32>
```



```
C:\Users\DELL>aws ec2 authorize-security-group-ingress --group-id %SG_ID% --protocol tcp --port 80 --cidr [redacted]/32
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0ba4328ce20d04c88",
            "GroupId": "sg-04230284a3583df06",
            "GroupOwnerId": "933376059751",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIpv4": "[redacted]1/32",
            "SecurityGroupRuleArn": "arn:aws:ec2:ap-south-1:933376059751:security-group-rule/sgr-0ba4328ce20d04c88"
        }
    ]
}
```

Verify on AWS EC2 console.



Security Groups (1/13) Info

	Name	Security group ID	Security group name	VPC ID
□	–	sg-0bdd8e8ca50c9fcac	launch-wizard-10	vpc-0ba8ec0a78594482a
□	–	sg-07454997ab6fce4ae	launch-wizard-1	vpc-0ba8ec0a78594482a
☑	–	sg-04230284a3583df06	My-SG	vpc-01cdb62e3d641769f
□	–	sg-0e8174433b4741559	default	vpc-0ba8ec0a78594482a
□	–	sg-0d60ea9ec55a094cf	launch-wizard-9	vpc-0ba8ec0a78594482a

sg-04230284a3583df06 - My-SG

Inbound rules (2)

	Name	Security group rule ID	IP version	Type	Protocol
□	–	sgr-0ba4328ce20d04c88	IPv4	HTTP	TCP
□	–	sgr-01b02227e42e69b06	IPv4	SSH	TCP

Actions
Export security groups to CSV
Create security group
Find security groups by attribute or tag
Manage tags
Edit inbound rules
Search
Activate Windows
Go to Settings to activate Windows.

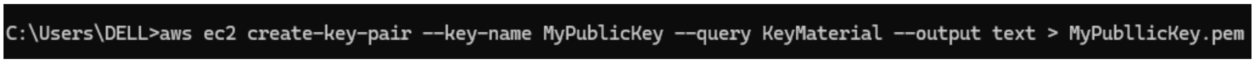
Security group is created and the inbound rules are added successfully.

STEP 9: Create SSH key pair

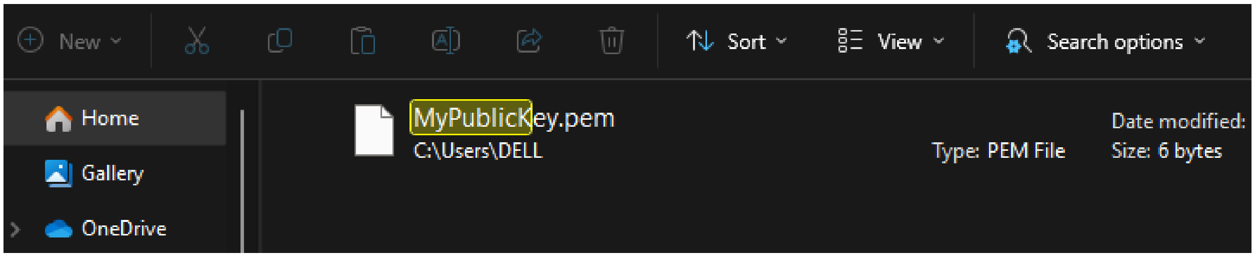
Command to create SSH key pair:

```
aws ec2 create-key-pair
        --key-name “YourKeyName”
        --query “KeyMaterial”
        --output text > YourKeyName.pem
```

The .pem file has been saved at your local machine.

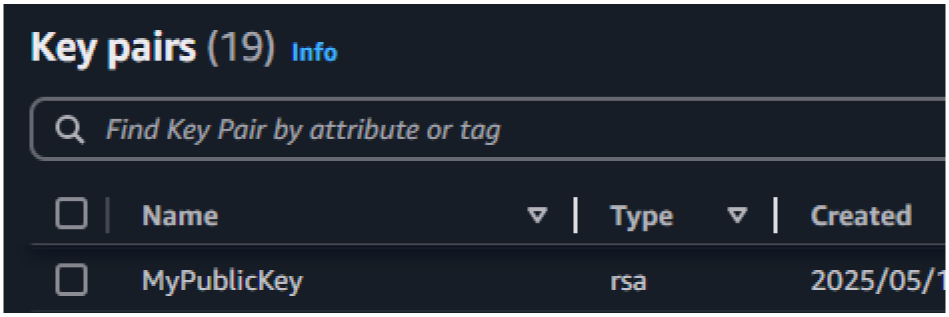


```
C:\Users\DELL>aws ec2 create-key-pair --key-name MyPublicKey --query KeyMaterial --output text > MyPubllicKey.pem
```



MyPublicKey.pem
C:\Users\DELL
Type: PEM File
Date modified:
Size: 6 bytes

Verify on the AWS EC2 console



Key pairs (19) Info

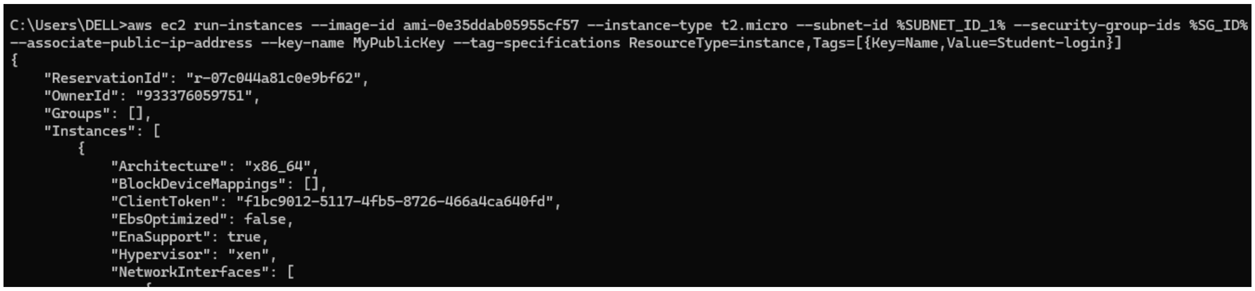
☐	Name	Type	Created
☐	MyPublicKey	rsa	2025/05/1

Key pair is created successfully.

STEP 10: Launch EC2 instances in each subnet

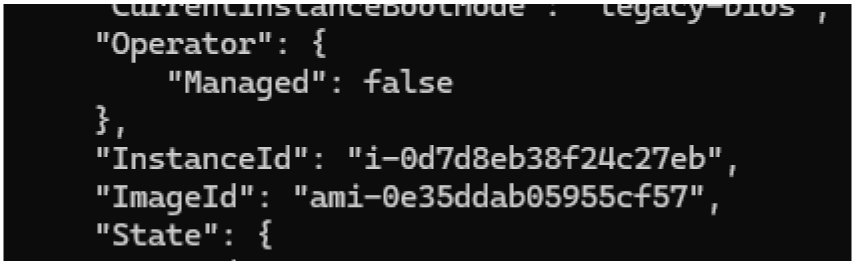
Command: `aws ec2 run-instances`

```
                    --image-id ami-0e35ddab05955cf57
                    --instance-type t2.micro
                    --subnet-id %SUBNET_ID_1%
                    --security-group-ids %SG_ID%
                    --associate-public-ip-address
                    --key-name KeyName
                    --tag-specifications
             ResourceType=instance,Tags=[{Key=Name,Value=ServerName}]
```



```
C:\Users\DELL>aws ec2 run-instances --image-id ami-0e35ddab05955cf57 --instance-type t2.micro --subnet-id %SUBNET_ID_1% --security-group-ids %SG_ID%
--associate-public-ip-address --key-name MyPublicKey --tag-specifications ResourceType=instance,Tags=[{Key=Name,Value=Student-login}]
{
    "ReservationId": "r-07c044a81c0e9bf62",
    "OwnerId": "933376059751",
    "Groups": [],
    "Instances": [
        {
            "Architecture": "x86_64",
            "BlockDeviceMappings": [],
            "ClientToken": "f1bc9012-5117-4fb5-8726-466a4ca640fd",
            "EbsOptimized": false,
            "EnaSupport": true,
            "Hypervisor": "xen",
            "NetworkInterfaces": [
```

Copy the instance id from the output.



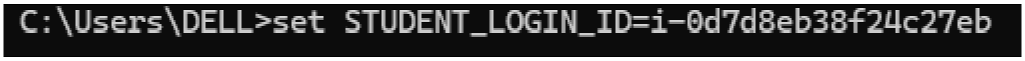
```
        "CurrentInstanceBootMode": "legacy-bios",
        "Operator": {
            "Managed": false
        },
        "InstanceId": "i-0d7d8eb38f24c27eb",
        "ImageId": "ami-0e35ddab05955cf57",
        "State": {
```

Save the instance id:

```
Set STUDENT_LOGIN_ID=i-XXXXXXXXXXXXXX
```

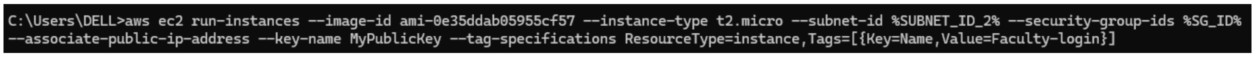
- Kavya Sharma

Rayat Bahra University

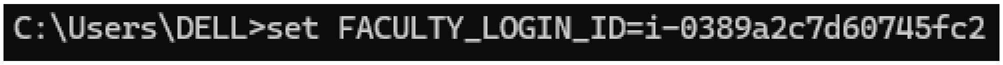


```
C:\Users\DELL>set STUDENT_LOGIN_ID=i-0d7d8eb38f24c27eb
```

Repeat the same commands for the second instance in Public-subnet02

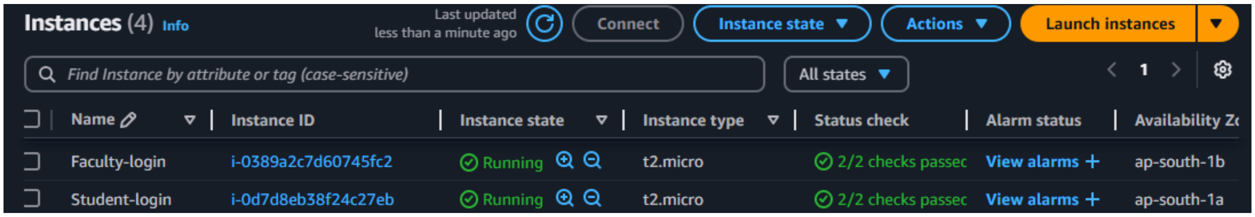


```
C:\Users\DELL>aws ec2 run-instances --image-id ami-0e35ddab05955cf57 --instance-type t2.micro --subnet-id %SUBNET_ID_2% --security-group-ids %SG_ID%
--associate-public-ip-address --key-name MyPublicKey --tag-specifications ResourceType=instance,Tags=[{Key=Name,Value=Faculty-login}]
```



```
C:\Users\DELL>set FACULTY_LOGIN_ID=i-0389a2c7d60745fc2
```

Verify on the AWS EC2 console.



Instances (4) Info
Last updated less than a minute ago
Connect
Instance state ▼
Actions ▼
Launch instances ▼
Find Instance by attribute or tag (case-sensitive)
All states ▼
1

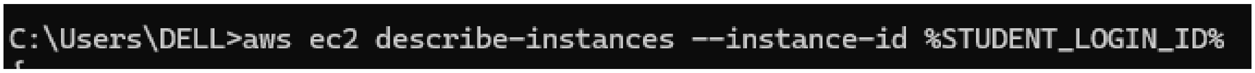
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zo
Faculty-login	i-0389a2c7d60745fc2	✓ Running	t2.micro	✓ 2/2 checks passec	View alarms +	ap-south-1b
Student-login	i-0d7d8eb38f24c27eb	✓ Running	t2.micro	✓ 2/2 checks passec	View alarms +	ap-south-1a

Both the instances are created successfully in different availability zones.

STEP 11: Login to the instances and create HTML file

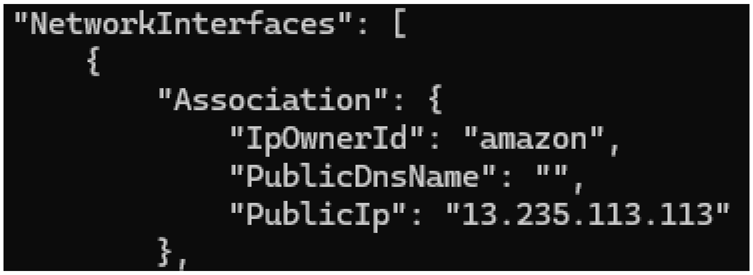
Run the command to get the public ip addresses of both the instances:

```
aws ec2 describe-instances --instance-id %STUDENT_LOGIN_ID%
```



```
C:\Users\DELL>aws ec2 describe-instances --instance-id %STUDENT_LOGIN_ID%
```

You can copy the public ip address from the output .



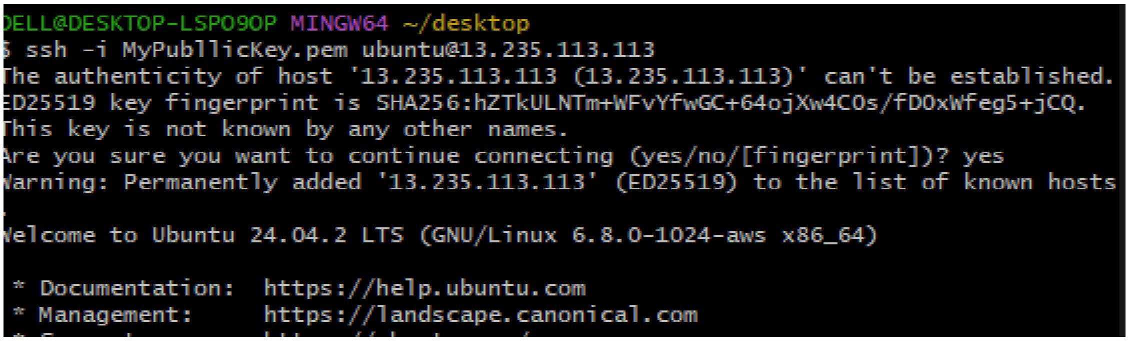
```
"NetworkInterfaces": [
    {
        "Association": {
            "IpOwnerId": "amazon",
            "PublicDnsName": "",
            "PublicIp": "13.235.113.113"
        },
```

Now, login to the server using bash and run these commands:

- cd desktop (location of .pem file)

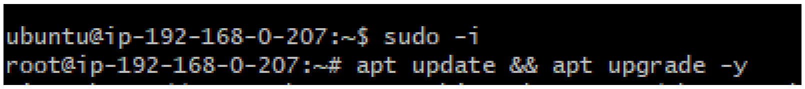
```
ELL@DESKTOP-LSPO9OP MINGW64 ~
 cd desktop
```

- ssh -i keyname.pem ubuntu@public-ip



```
DELL@DESKTOP-LSPO9OP MINGW64 ~/desktop
$ ssh -i MyPubllicKey.pem ubuntu@13.235.113.113
The authenticity of host '13.235.113.113 (13.235.113.113)' can't be established.
ED25519 key fingerprint is SHA256:hZTkULNTm+WFvYfwGC+64ojXw4COs/fDOxWfeg5+jCQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '13.235.113.113' (ED25519) to the list of known hosts
.
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-1024-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
```

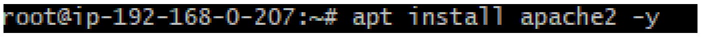
- sudo -i
- apt update && apt upgrade -y



```
ubuntu@ip-192-168-0-207:~$ sudo -i
root@ip-192-168-0-207:~# apt update && apt upgrade -y
```

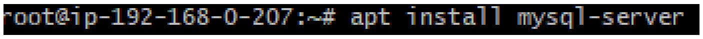
Install the lamp stack

- apt install apache2 -y



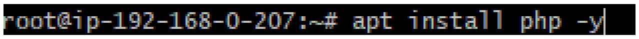
```
root@ip-192-168-0-207:~# apt install apache2 -y
```

- apt install mysql-server -y



```
root@ip-192-168-0-207:~# apt install mysql-server
```

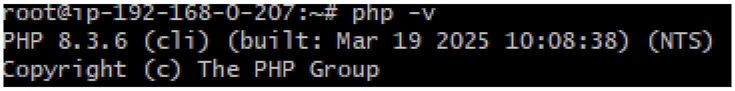
- apt install php -y



```
root@ip-192-168-0-207:~# apt install php -y
```

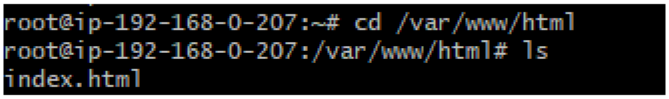
To check php version, run command:

- php -v



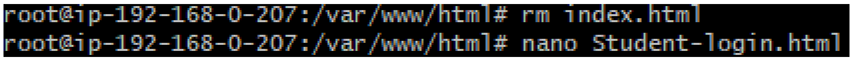
```
root@ip-192-168-0-207:~# php -v
PHP 8.3.6 (cli) (built: Mar 19 2025 10:08:38) (NTS)
Copyright (c) The PHP Group
```

- cd /var/www/html
- ls



```
root@ip-192-168-0-207:~# cd /var/www/html
root@ip-192-168-0-207:/var/www/html# ls
index.html
```

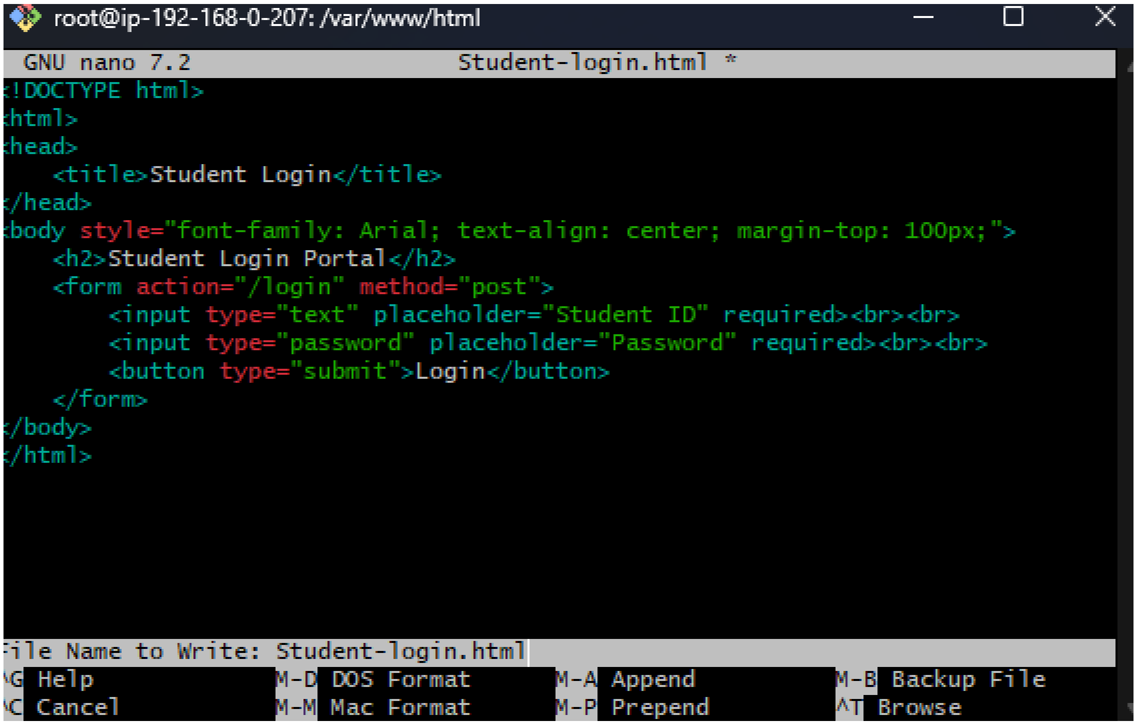
- rm index.html
- nano Student-login.html



```
root@ip-192-168-0-207:/var/www/html# rm index.html
root@ip-192-168-0-207:/var/www/html# nano Student-login.html
```

Create an html file for the Student login server .

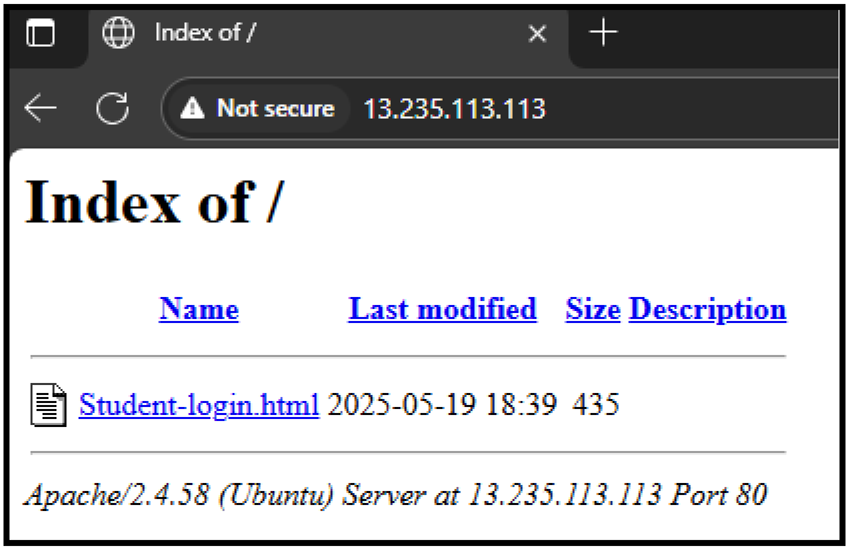
To save the file: CTRL+ X > press Y > press ENTER



root@ip-192-168-0-207: /var/www/html

```
 GNU nano 7.2                      Student-login.html *
<!DOCTYPE html>
<html>
<head>
    <title>Student Login</title>
</head>
<body style="font-family: Arial; text-align: center; margin-top: 100px;">
    <h2>Student Login Portal</h2>
    <form action="/login" method="post">
        <input type="text" placeholder="Student ID" required><br><br>
        <input type="password" placeholder="Password" required><br><br>
        <button type="submit">Login</button>
    </form>
</body>
</html>




File Name to Write: Student-login.html
^G Help                M-D DOS Format         M-A Append            M-B Backup File
^C Cancel              M-M Mac Format         M-P Prepend           ^T Browse
```

Now open the Public IP address of the Student-login instance on the web browser

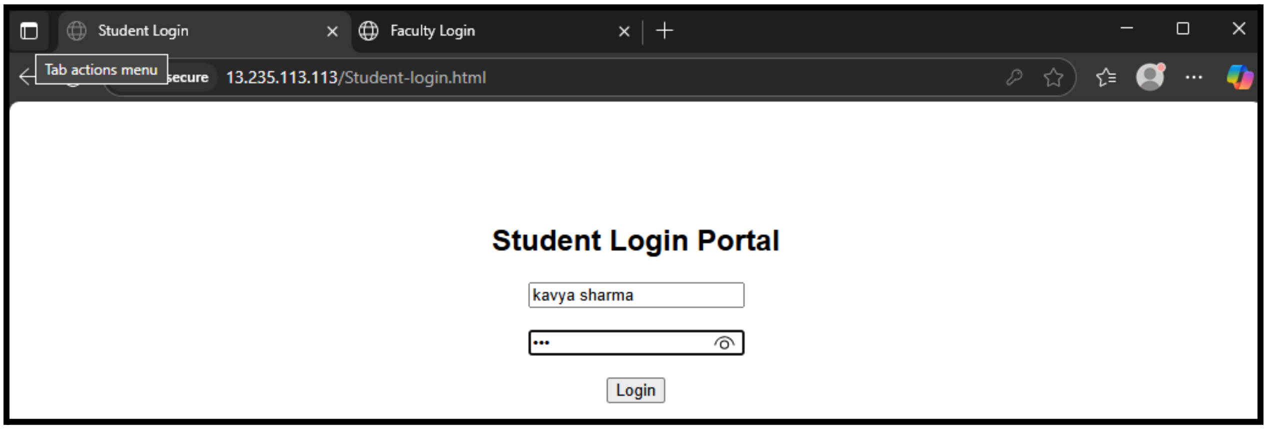


Index of /

Name	Last modified	Size	Description
Student-login.html	2025-05-19 18:39	435	

Apache/2.4.58 (Ubuntu) Server at 13.235.113.113 Port 80

Click on "Student-login.html"



13.235.113.113/Student-login.html

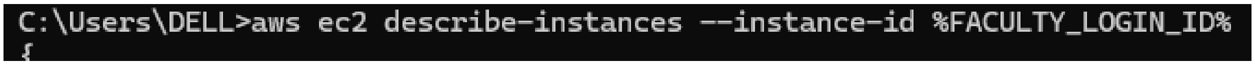
Student Login Portal

kavya sharma

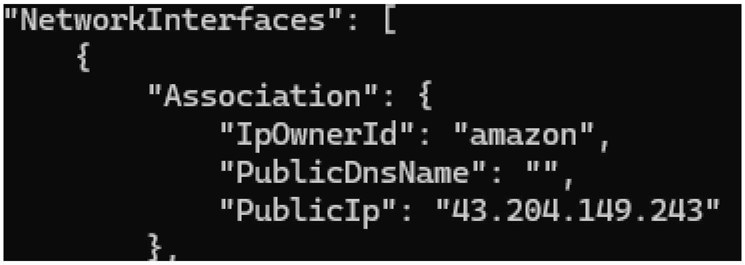
•••

<u>The Student-login server is working successfully.</u>

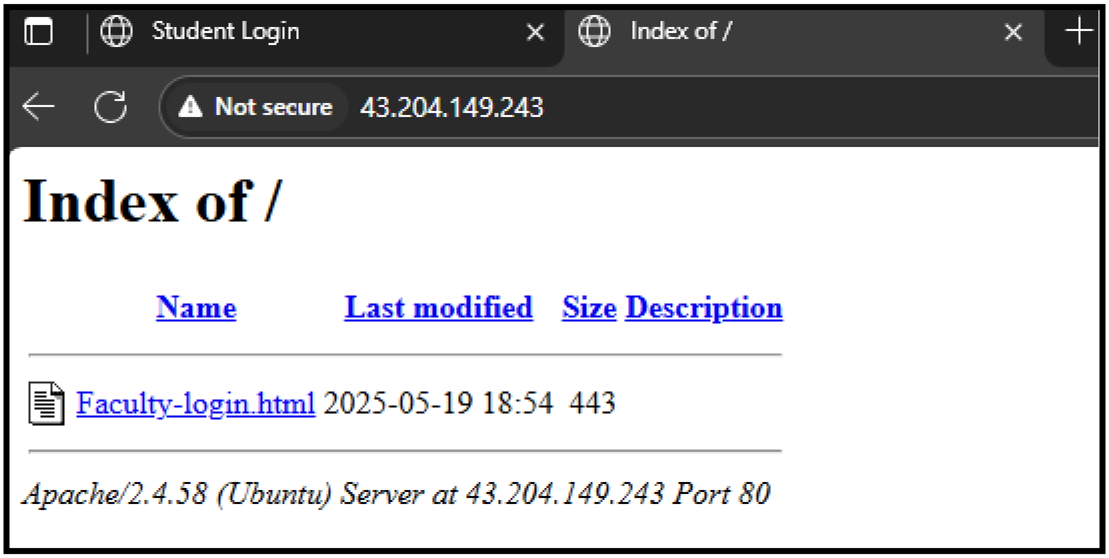
Now for Faculty-login instance repeat the same commands:



```
C:\Users\DELL>aws ec2 describe-instances --instance-id %FACULTY_LOGIN_ID%
{
```



```
"NetworkInterfaces": [
    {
        "Association": {
            "IpOwnerId": "amazon",
            "PublicDnsName": "",
            "PublicIp": "43.204.149.243"
        },
```

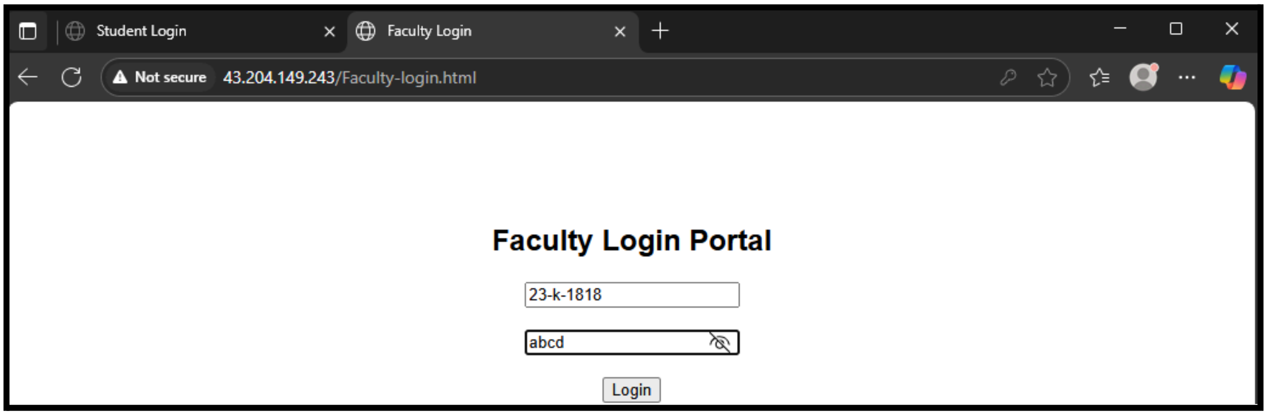


43.204.149.243

Index of /

Name	Last modified	Size	Description
Faculty-login.html	2025-05-19 18:54	443	

Apache/2.4.58 (Ubuntu) Server at 43.204.149.243 Port 80



Student Login | Faculty Login

Not secure 43.204.149.243/Faculty-login.html

Faculty Login Portal

23-k-1818

abcd

Login

Exit from git bash.

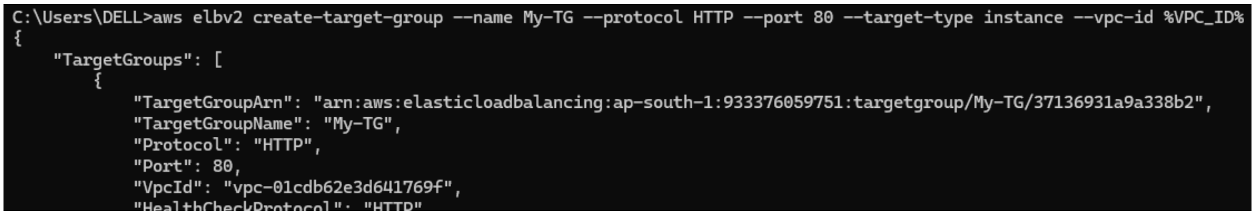
Both the servers are working successfully.

STEP 12: Create a Target group for Load balancer

Run the command to create Target group:

```
     aws elbv2 create-target-group
          --name ”TargetGroupName” --protocol HTTP --port 80
          --target-type instance --vpc-id %VPC_ID%
```

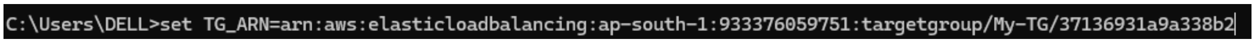
Copy the TARGET GROUP ARN from the output.



```
C:\Users\DELL>aws elbv2 create-target-group --name My-TG --protocol HTTP --port 80 --target-type instance --vpc-id %VPC_ID%
{
    "TargetGroups": [
        {
            "TargetGroupArn": "arn:aws:elasticloadbalancing:ap-south-1:933376059751:targetgroup/My-TG/37136931a9a338b2",
            "TargetGroupName": "My-TG",
            "Protocol": "HTTP",
            "Port": 80,
            "VpcId": "vpc-01cdb62e3d641769f",
            "HealthCheckProtocol": "HTTP"
```

Save theTargetGroupArn:

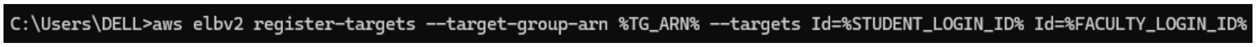
```
set TG_ARN=arn:aws:elasticloadbalancing:xxxxxxxxxxxxxxxx
```



```
C:\Users\DELL>set TG_ARN=arn:aws:elasticloadbalancing:ap-south-1:933376059751:targetgroup/My-TG/37136931a9a338b2
```

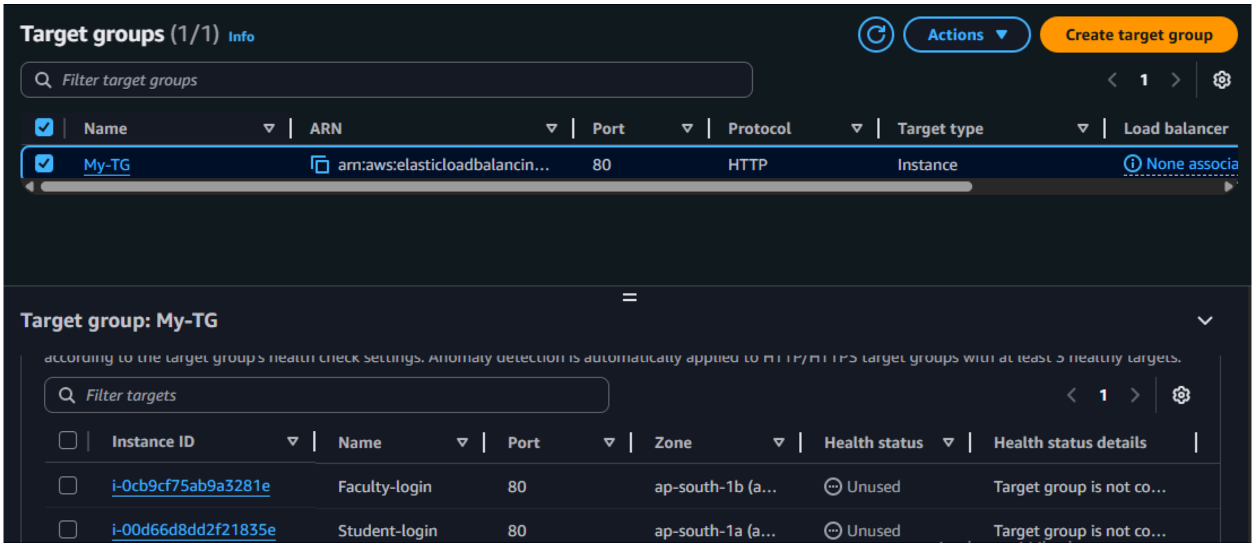
Register instances with the target group:

```
     aws elbv2 register-targets --target-group-arn %TG_ARN%
               --targets ID=%STUDENT_LOGIN_ID% ID=%FACULTY_LOGIN_ID%
```



```
C:\Users\DELL>aws elbv2 register-targets --target-group-arn %TG_ARN% --targets Id=%STUDENT_LOGIN_ID% Id=%FACULTY_LOGIN_ID%
```

Verify on the AWS EC2 console.



Target groups (1/1) Info

☑	Name	ARN	Port	Protocol	Target type	Load balancer
☑	My-TG	arn:aws:elasticloadbalancin...	80	HTTP	Instance	ⓘ None associa

Target group: My-TG

according to the target group's health check settings. Anomaly detection is automatically applied to HTTP/HTTPS target groups with at least 3 healthy targets.

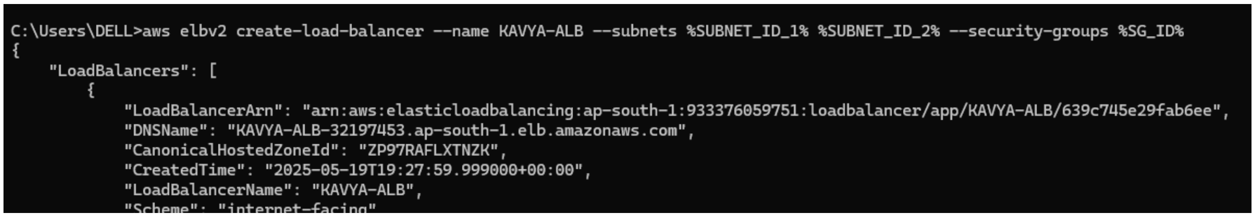
□	Instance ID	Name	Port	Zone	Health status	Health status details
□	i-0cb9cf75ab9a3281e	Faculty-login	80	ap-south-1b (a...	Unused	Target group is not co...
□	i-00d66d8dd2f21835e	Student-login	80	ap-south-1a (a...	Unused	Target group is not co...

Target group is successfully created.

STEP 13: Create application load balancer

Run the commands to create ALB:

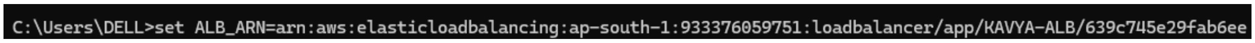
```
    aws elbv2 create-load-balancer --name YourName-ALB
          --subnets %SUBNET_ID_1% %SUBNET_ID_2%
          --security-group %SG_ID%
```



```
C:\Users\DELL>aws elbv2 create-load-balancer --name KAVYA-ALB --subnets %SUBNET_ID_1% %SUBNET_ID_2% --security-groups %SG_ID%
{
    "LoadBalancers": [
        {
            "LoadBalancerArn": "arn:aws:elasticloadbalancing:ap-south-1:933376059751:loadbalancer/app/KAVYA-ALB/639c745e29fab6ee",
            "DNSName": "KAVYA-ALB-32197453.ap-south-1.elb.amazonaws.com",
            "CanonicalHostedZoneId": "ZP97RAFLXTNZK",
            "CreatedTime": "2025-05-19T19:27:59.999000+00:00",
            "LoadBalancerName": "KAVYA-ALB",
            "Scheme": "internet-facing"
```

Save the Load Balancer ARN:

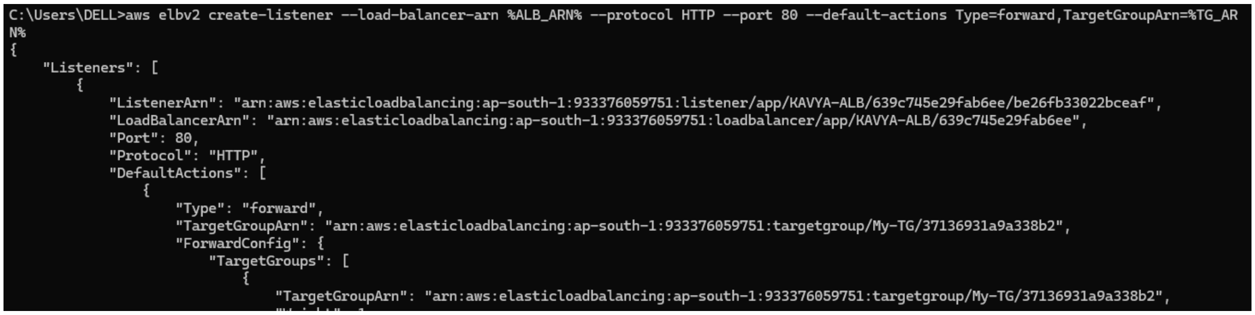
```
set ALB_ARN = arn:aws::XXXXXXXXXXXXX
```



```
C:\Users\DELL>set ALB_ARN=arn:aws:elasticloadbalancing:ap-south-1:933376059751:loadbalancer/app/KAVYA-ALB/639c745e29fab6ee
```

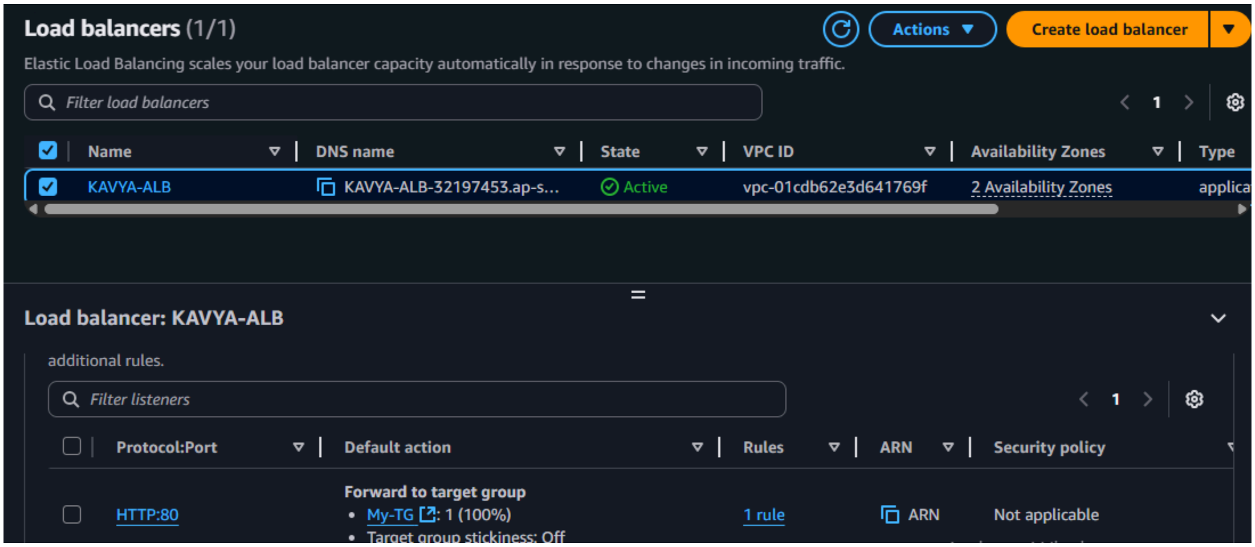
Run the command to create listener:

```
    aws elbv2 create-listener
          --load-balancer-arn %ALB_ARN% --protocol HTTP --port 80
          --default-action Type=forward,TargetGroupArn=%TG_ARN%
```



```
C:\Users\DELL>aws elbv2 create-listener --load-balancer-arn %ALB_ARN% --protocol HTTP --port 80 --default-actions Type=forward,TargetGroupArn=%TG_AR
N%
{
    "Listeners": [
        {
            "ListenerArn": "arn:aws:elasticloadbalancing:ap-south-1:933376059751:listener/app/KAVYA-ALB/639c745e29fab6ee/be26fb33022bceaf",
            "LoadBalancerArn": "arn:aws:elasticloadbalancing:ap-south-1:933376059751:loadbalancer/app/KAVYA-ALB/639c745e29fab6ee",
            "Port": 80,
            "Protocol": "HTTP",
            "DefaultActions": [
                {
                    "Type": "forward",
                    "TargetGroupArn": "arn:aws:elasticloadbalancing:ap-south-1:933376059751:targetgroup/My-TG/37136931a9a338b2",
                    "ForwardConfig": {
                        "TargetGroups": [
                            {
                                "TargetGroupArn": "arn:aws:elasticloadbalancing:ap-south-1:933376059751:targetgroup/My-TG/37136931a9a338b2",
```

Verify on the AWS EC2 console.



Load balancers (1/1)

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

Filter load balancers

☑	Name	DNS name	State	VPC ID	Availability Zones	Type
☑	KAVYA-ALB	KAVYA-ALB-32197453.ap-s...	Active	vpc-01cdb62e3d641769f	2 Availability Zones	applica

Load balancer: KAVYA-ALB

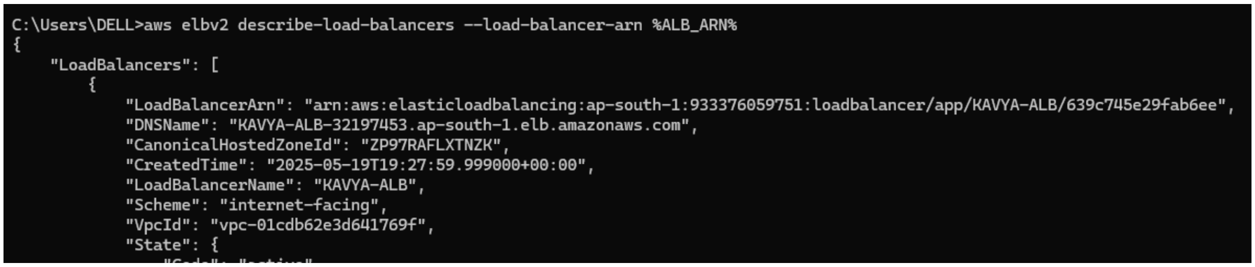
additional rules.

Filter listeners

□	Protocol:Port	Default action	Rules	ARN	Security policy
□	HTTP:80	Forward to target group • My-TG: 1 (100%) • Target group stickiness: Off	1 rule	ARN	Not applicable

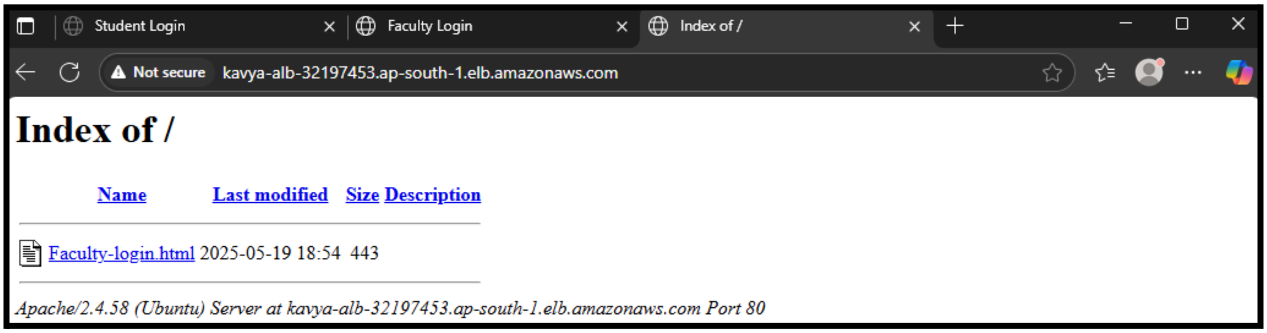
Run the command to find the DNS of the Load Balancer:

```
Aws elbv2 describe-load-balancer --load-balancer-arn %ALB_ARN%
```



```
C:\Users\DELL>aws elbv2 describe-load-balancers --load-balancer-arn %ALB_ARN%
{
    "LoadBalancers": [
        {
            "LoadBalancerArn": "arn:aws:elasticloadbalancing:ap-south-1:933376059751:loadbalancer/app/KAVYA-ALB/639c745e29fab6ee",
            "DNSName": "KAVYA-ALB-32197453.ap-south-1.elb.amazonaws.com",
            "CanonicalHostedZoneId": "ZP97RAFLXTNZK",
            "CreatedTime": "2025-05-19T19:27:59.999000+00:00",
            "LoadBalancerName": "KAVYA-ALB",
            "Scheme": "internet-facing",
            "VpcId": "vpc-01cdb62e3d641769f",
            "State": {
```

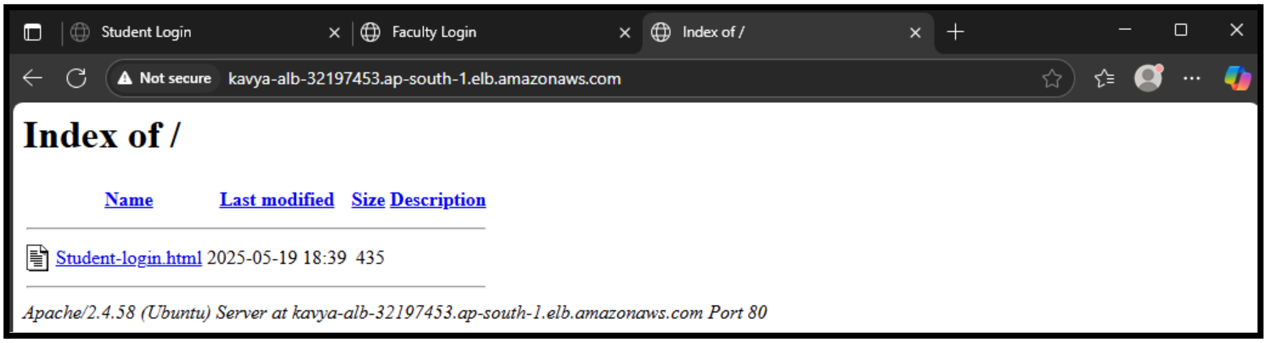
Copy the DNS name from the output and open it in your web browser.



Index of /

```
Name                Last modified      Size Description
Faculty-login.html  2025-05-19 18:54   443
Apache/2.4.58 (Ubuntu) Server at kavya-alb-32197453.ap-south-1.elb.amazonaws.com Port 80
```

Now refresh this page to confirm that the load balancer is directing traffic between the servers.



Index of /

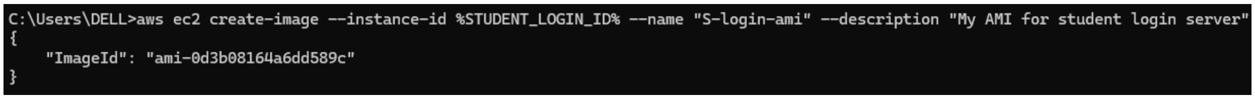
```
Name                Last modified      Size Description
Student-login.html  2025-05-19 18:39   435
Apache/2.4.58 (Ubuntu) Server at kavya-alb-32197453.ap-south-1.elb.amazonaws.com Port 80
```

The load balancer successfully distributes traffic between Student and faculty login instances.

STEP 14: Create AMI of both the instances

Run the command to create AMI for both the servers:

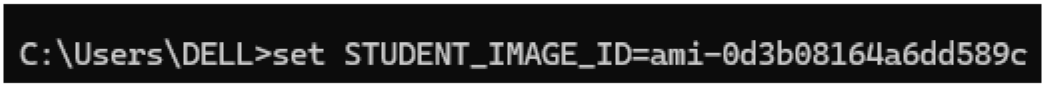
```
aws ec2 create-image
        --instance-id %STUDENT_LOGIN_ID% --name "AmiName"
        --description "My AMI for student login server"
```



```
C:\Users\DELL>aws ec2 create-image --instance-id %STUDENT_LOGIN_ID% --name "S-login-ami" --description "My AMI for student login server"
{
    "ImageId": "ami-0d3b08164a6dd589c"
}
```

Save the image id:

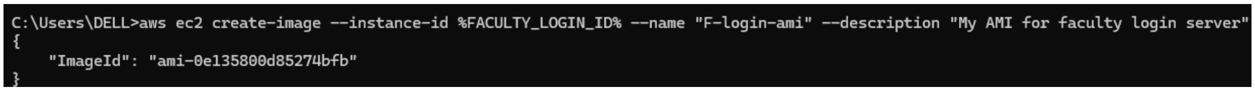
```
set STUDENT_IMAGE_ID=ami-xxxxxxxxxxxx
```



```
C:\Users\DELL>set STUDENT_IMAGE_ID=ami-0d3b08164a6dd589c
```

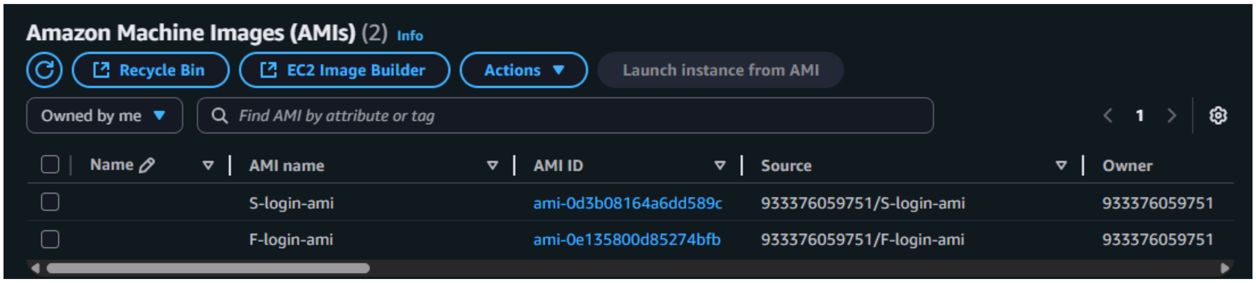
Repeat the same commands to create AMI for the Faculty login server.

- Kavya Sharma
Rayat Bahra University



```
C:\Users\DELL>aws ec2 create-image --instance-id %FACULTY_LOGIN_ID% --name "F-login-ami" --description "My AMI for faculty login server"
{
    "ImageId": "ami-0e135800d85274bfb"
}
```

Verify on the AWS EC2 console.



□	Name	AMI name	AMI ID	Source	Owner
□		S-login-ami	ami-0d3b08164a6dd589c	933376059751/S-login-ami	933376059751
□		F-login-ami	ami-0e135800d85274bfb	933376059751/F-login-ami	933376059751

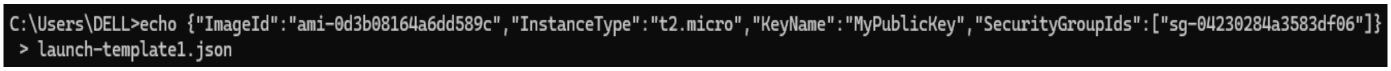
Both the AMI's are created successfully.

STEP 15: Create Launch Template for Auto Scaling

Run the following command to create JSON file:

For student launch template: File name- "launch-template1.json"

```
{
     "ImageId":"<STUDENT_IMAGE_ID>",
     "InstanceType":"t2.micro",
     "KeyName":"YourKeyName",
     "SecurityGroupIds":["SG_ID"]

} > launch-template1.json
```

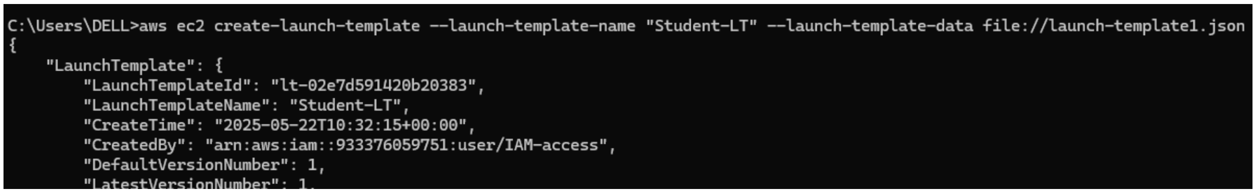


```
C:\Users\DELL>echo {"ImageId":"ami-0d3b08164a6dd589c","InstanceType":"t2.micro","KeyName":"MyPublicKey","SecurityGroupIds":["sg-04230284a3583df06"]}
 > launch-template1.json
```

Command to create launch template:

```
          aws ec2 create-launch-template
               --launch-template-name "LaunchTemplateName"
               --launch-template-data file://launch-template1.json
```

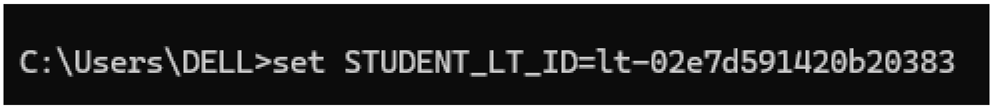
Copy the LaunchTemplateId from the output.



```
C:\Users\DELL>aws ec2 create-launch-template --launch-template-name "Student-LT" --launch-template-data file://launch-template1.json
{
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-02e7d591420b20383",
        "LaunchTemplateName": "Student-LT",
        "CreateTime": "2025-05-22T10:32:15+00:00",
        "CreatedBy": "arn:aws:iam::933376059751:user/IAM-access",
        "DefaultVersionNumber": 1,
        "LatestVersionNumber": 1,
```

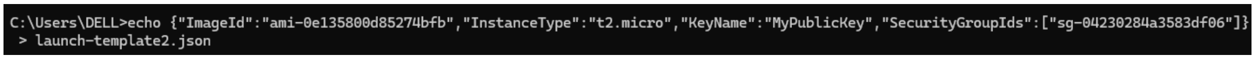
Save the launch template id:

```
set STUDENT_LT_ID=lt-xxxxxxxxxxxxx
```

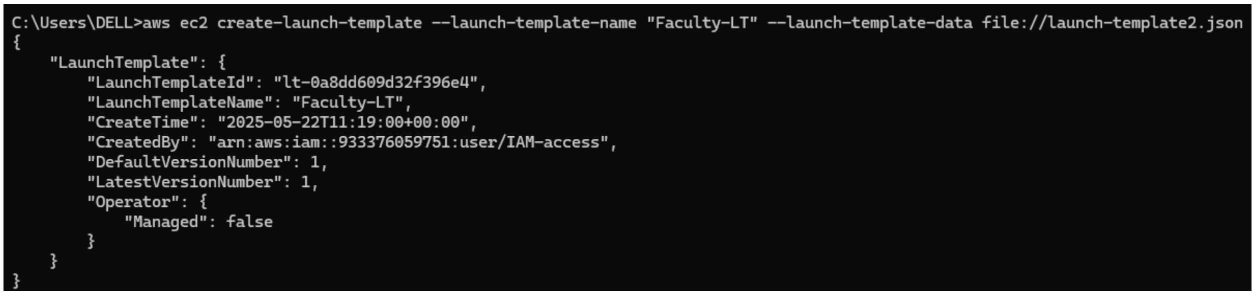


```
C:\Users\DELL>set STUDENT_LT_ID=lt-02e7d591420b20383
```

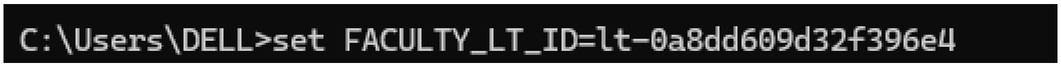
Repeat the commands to create a launch template for the Faculty login server.



```
C:\Users\DELL>echo {"ImageId":"ami-0e135800d85274bfb","InstanceType":"t2.micro","KeyName":"MyPublicKey","SecurityGroupIds":["sg-04230284a3583df06"]}
 > launch-template2.json
```

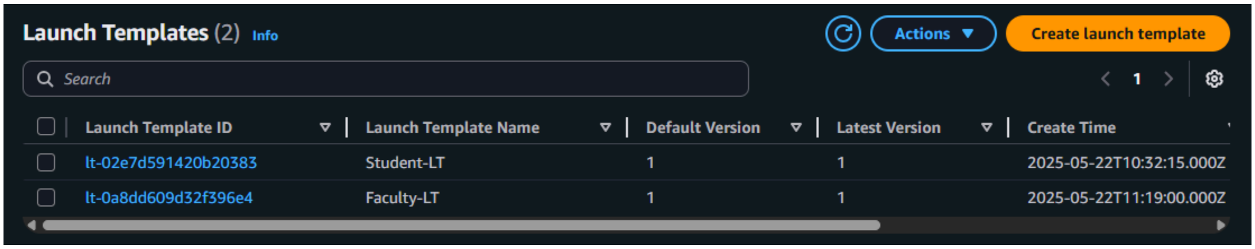


```
C:\Users\DELL>aws ec2 create-launch-template --launch-template-name "Faculty-LT" --launch-template-data file://launch-template2.json
{
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-0a8dd609d32f396e4",
        "LaunchTemplateName": "Faculty-LT",
        "CreateTime": "2025-05-22T11:19:00+00:00",
        "CreatedBy": "arn:aws:iam::933376059751:user/IAM-access",
        "DefaultVersionNumber": 1,
        "LatestVersionNumber": 1,
        "Operator": {
            "Managed": false
        }
    }
}
```



```
C:\Users\DELL>set FACULTY_LT_ID=lt-0a8dd609d32f396e4
```

Verify on the AWS EC2 console.



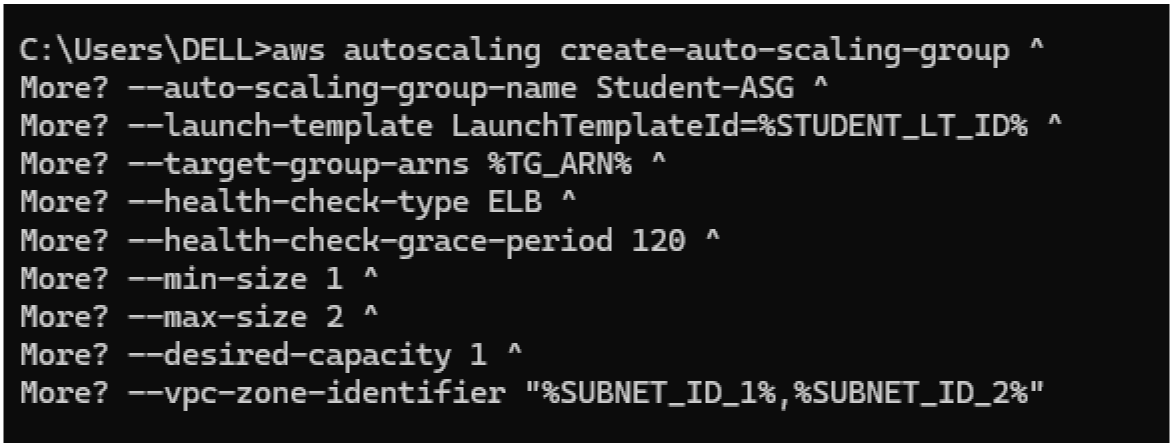
□	Launch Template ID	Launch Template Name	Default Version	Latest Version	Create Time
□	lt-02e7d591420b20383	Student-LT	1	1	2025-05-22T10:32:15.000Z
□	lt-0a8dd609d32f396e4	Faculty-LT	1	1	2025-05-22T11:19:00.000Z

Both the Launch Templates are created successfully.

STEP 16: Create Auto Scaling Groups and their policies

Run the following commands to create Auto Scaling Groups:

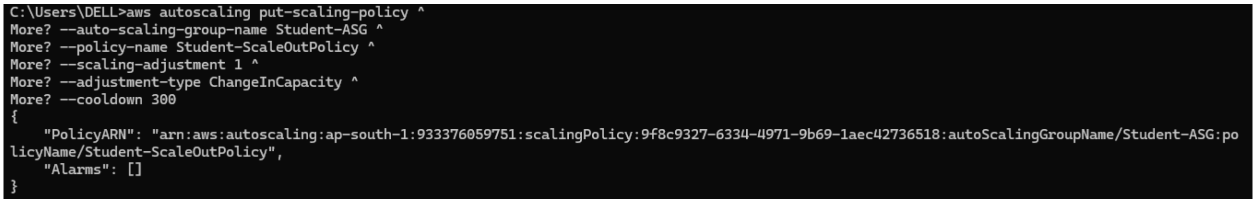
```
     aws autoscaling create-auto-scaling-group ^
          --auto-scaling-group-name YourName-ASG ^
          --launch-template LaunchTemplateId=%STUDENT_LT_ID%
          --target-group-arns %TG_ARN% ^
          --health-check-type ELB ^
          --health-check-grace-period 120 ^
          --min-size 1 ^
          --max-size 2 ^
          --desired-capacity 1 ^
          --vpc-zone-identifier “SUBNET_ID_1,SUBNET_ID_2”
```



```
C:\Users\DELL>aws autoscaling create-auto-scaling-group ^
More? --auto-scaling-group-name Student-ASG ^
More? --launch-template LaunchTemplateId=%STUDENT_LT_ID% ^
More? --target-group-arns %TG_ARN% ^
More? --health-check-type ELB ^
More? --health-check-grace-period 120 ^
More? --min-size 1 ^
More? --max-size 2 ^
More? --desired-capacity 1 ^
More? --vpc-zone-identifier "%SUBNET_ID_1%,%SUBNET_ID_2%"
```

Run the following command to create scale-out and scale-in policy for autoscaling group:

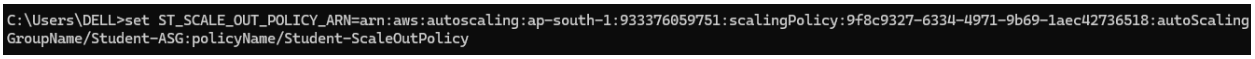
```
     aws autoscaling put-scaling-policy ^
              --auto-scaling-group-name YourName-ASG ^
              --policy-name YourName-ScaleOutPolicy ^
              --scaling-adjustment 1 ^
              --adjustment-type ChangeInCapacity ^
              --cooldown 300
```



```
C:\Users\DELL>aws autoscaling put-scaling-policy ^
More? --auto-scaling-group-name Student-ASG ^
More? --policy-name Student-ScaleOutPolicy ^
More? --scaling-adjustment 1 ^
More? --adjustment-type ChangeInCapacity ^
More? --cooldown 300
{
    "PolicyARN": "arn:aws:autoscaling:ap-south-1:933376059751:scalingPolicy:9f8c9327-6334-4971-9b69-1aec42736518:autoScalingGroupName/Student-ASG:po
licyName/Student-ScaleOutPolicy",
    "Alarms": []
}
```

Save the PolicyARN:

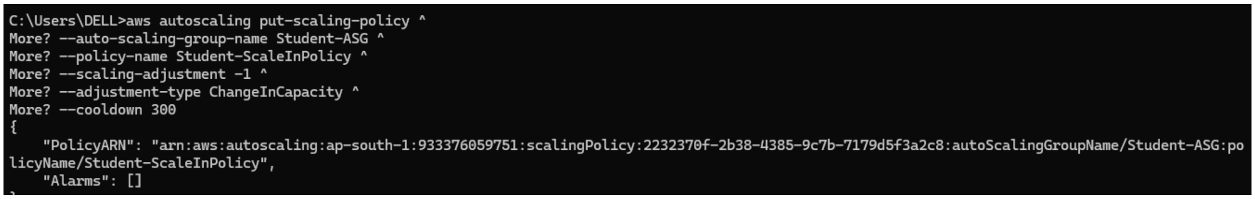
```
set ST_SCALE_OUT_POLICY_ARN=arn:aws:autoscaling:xxxxxxxxxxxxxxx
```



```
C:\Users\DELL>set ST_SCALE_OUT_POLICY_ARN=arn:aws:autoscaling:ap-south-1:933376059751:scalingPolicy:9f8c9327-6334-4971-9b69-1aec42736518:autoScaling
GroupName/Student-ASG:policyName/Student-ScaleOutPolicy
```

For Scale-in policy:

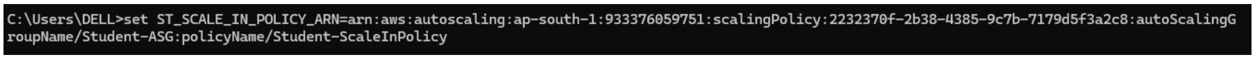
```
     aws autoscaling put-scaling-policy ^
            --auto-scaling-group-name YourName-ASG ^
            --policy-name YourName-ScaleInPolicy ^
            --scaling-adjustment -1 ^
            --adjustment-type ChangeInCapacity ^
            --cooldown 300
```



```
C:\Users\DELL>aws autoscaling put-scaling-policy ^
More? --auto-scaling-group-name Student-ASG ^
More? --policy-name Student-ScaleInPolicy ^
More? --scaling-adjustment -1 ^
More? --adjustment-type ChangeInCapacity ^
More? --cooldown 300
{
    "PolicyARN": "arn:aws:autoscaling:ap-south-1:933376059751:scalingPolicy:2232370f-2b38-4385-9c7b-7179d5f3a2c8:autoScalingGroupName/Student-ASG:po
licyName/Student-ScaleInPolicy",
    "Alarms": []
```

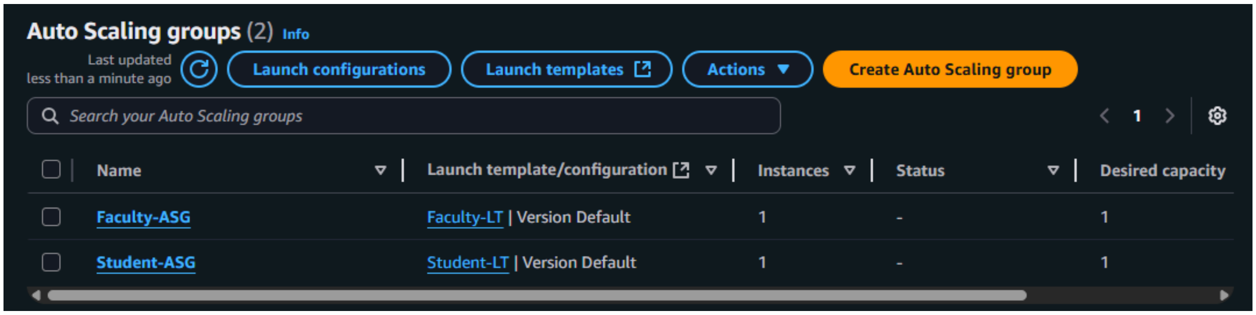
Save the PolicyARN:

`set ST_SCALE_IN_POLICY_ARN=arn:aws:autoscaling:xxxxxxxxxxxxxxx`



```
C:\Users\DELL>set ST_SCALE_IN_POLICY_ARN=arn:aws:autoscaling:ap-south-1:933376059751:scalingPolicy:2232370f-2b38-4385-9c7b-7179d5f3a2c8:autoScalingG
roupName/Student-ASG:policyName/Student-ScaleInPolicy
```

Repeat the same commands for Faculty- Auto Scaling Group (Faculty-ASG).


Auto Scaling groups (2) Info
Last updated less than a minute ago
Launch configurations
Launch templates
Actions ▼
Create Auto Scaling group
Search your Auto Scaling groups
1

□	Name	Launch template/configuration	Instances	Status	Desired capacity
□	Faculty-ASG	Faculty-LT | Version Default	1	-	1
□	Student-ASG	Student-LT | Version Default	1	-	1

Auto Scaling Group and policies are created successfully for both the servers.

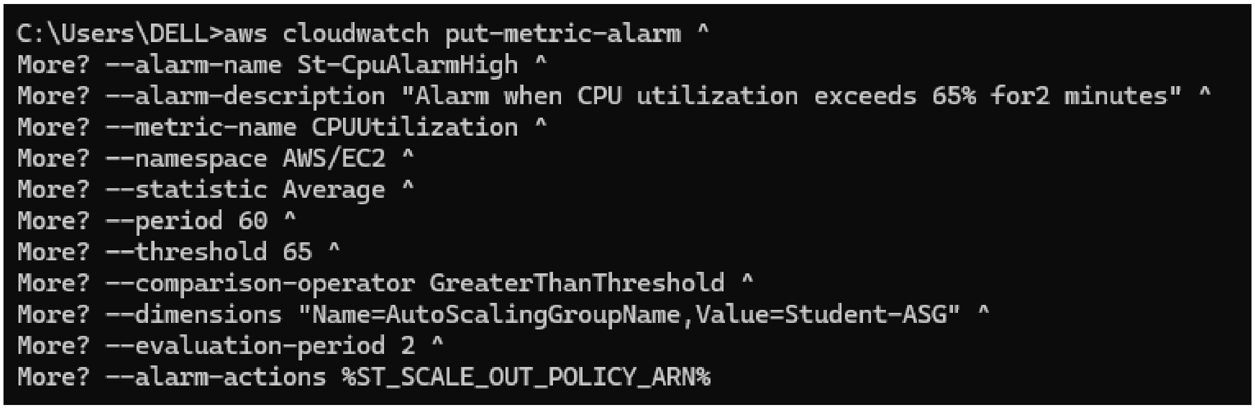
STEP 17: Configure CloudWatch Alarm for AutoScaling

Run the following commands to configure CloudWatch Alarm:

For Student-ASG:

1. Alarm action for scale-out policy

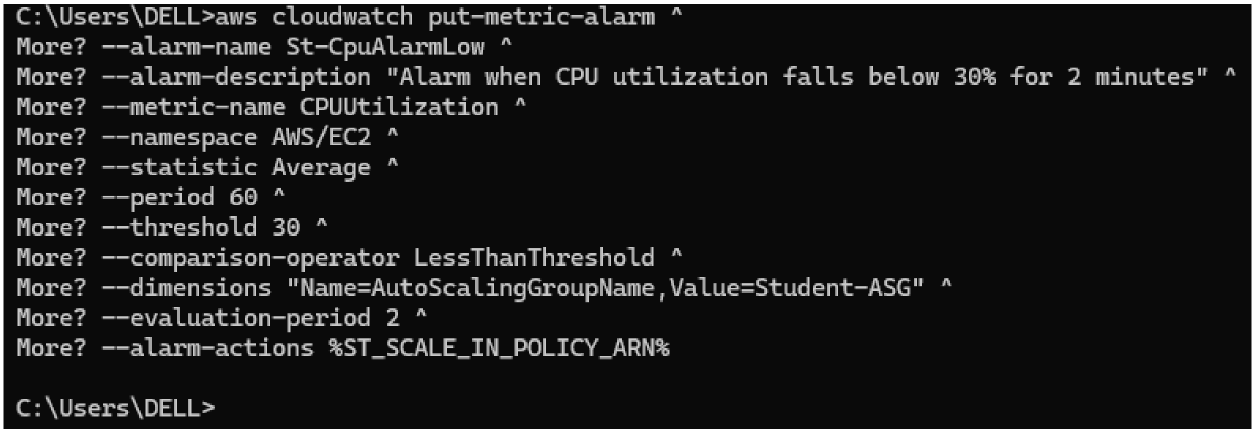
```
aws cloudwatch put-metric-alarm ^
    --alarm-name YourName-CPUAlarmHigh ^
    --alarm-description "Alarm when CPU exceeds 65% for 2
minutes" ^
    --metric-name CPUUtilization ^
    --namespace AWS/EC2 ^
    --statistic Average ^
    --period 60 ^
    --threshold 65 ^
    --comparison-operator GreaterThanThreshold ^
    --dimensions
"Name=AutoScalingGroupName,Value=YourName-ASG" ^
    --evaluation-periods 2 ^
    --alarm-actions %SCALE_OUT_POLICY_ARN%
```



```
C:\Users\DELL>aws cloudwatch put-metric-alarm ^
More? --alarm-name St-CpuAlarmHigh ^
More? --alarm-description "Alarm when CPU utilization exceeds 65% for2 minutes" ^
More? --metric-name CPUUtilization ^
More? --namespace AWS/EC2 ^
More? --statistic Average ^
More? --period 60 ^
More? --threshold 65 ^
More? --comparison-operator GreaterThanThreshold ^
More? --dimensions "Name=AutoScalingGroupName,Value=Student-ASG" ^
More? --evaluation-period 2 ^
More? --alarm-actions %ST_SCALE_OUT_POLICY_ARN%
```

2. Alarm action for scale-in policy

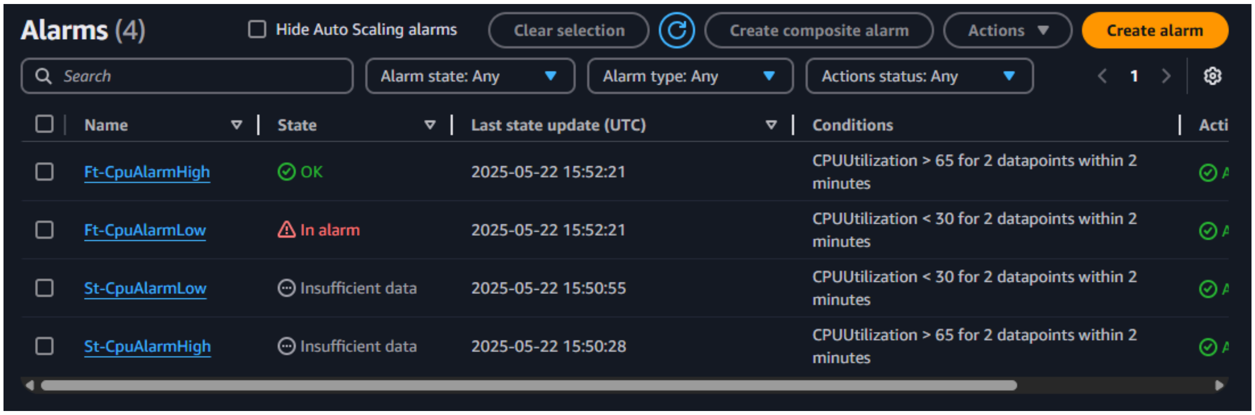
```
 aws cloudwatch put-metric-alarm ^
 --alarm-name YourName-CPUAlarmLow ^
--alarm-description "Alarm when CPU falls below 30% for
 2 mins" ^
 --metric-name CPUUtilization ^
--namespace AWS/EC2 ^
--statistic Average ^
--period 60 ^
--threshold 30 ^
--comparison-operator LessThanThreshold ^
--dimensions "Name=AutoScalingGroupName,Value=YourName-ASG" ^
--evaluation-periods 2 ^
--alarm-actions %SCALE_IN_POLICY_ARN%
```



```
C:\Users\DELL>aws cloudwatch put-metric-alarm ^
More? --alarm-name St-CpuAlarmLow ^
More? --alarm-description "Alarm when CPU utilization falls below 30% for 2 minutes" ^
More? --metric-name CPUUtilization ^
More? --namespace AWS/EC2 ^
More? --statistic Average ^
More? --period 60 ^
More? --threshold 30 ^
More? --comparison-operator LessThanThreshold ^
More? --dimensions "Name=AutoScalingGroupName,Value=Student-ASG" ^
More? --evaluation-period 2 ^
More? --alarm-actions %ST_SCALE_IN_POLICY_ARN%

C:\Users\DELL>
```

Repeat the same commands to configure CloudWatch Alarm for Faculty-ASG.

Verify on the AWS CloudWatch console.



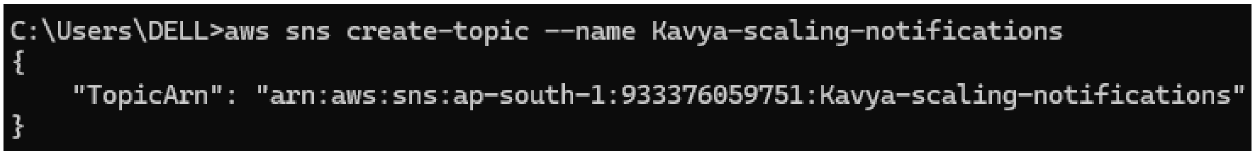
□	Name	State	Last state update (UTC)	Conditions	Acti
□	Ft-CpuAlarmHigh	✓ OK	2025-05-22 15:52:21	CPUUtilization > 65 for 2 datapoints within 2 minutes	✓
□	Ft-CpuAlarmLow	⚠ In alarm	2025-05-22 15:52:21	CPUUtilization < 30 for 2 datapoints within 2 minutes	✓
□	St-CpuAlarmLow	⊖ Insufficient data	2025-05-22 15:50:55	CPUUtilization < 30 for 2 datapoints within 2 minutes	✓
□	St-CpuAlarmHigh	⊖ Insufficient data	2025-05-22 15:50:28	CPUUtilization > 65 for 2 datapoints within 2 minutes	✓

CloudWatch Alarms for both the Auto Scaling Groups are created successfully.

STEP 18: Set up SNS for Notifications

Run the command to create topic:

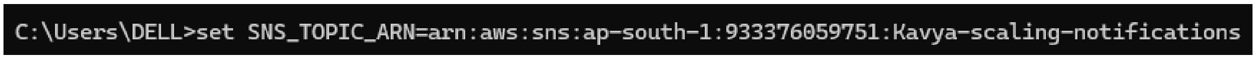
```
aws sns create-topic --name YourName-ScalingNotifications
```



```
C:\Users\DELL>aws sns create-topic --name Kavya-scaling-notifications
{
    "TopicArn": "arn:aws:sns:ap-south-1:933376059751:Kavya-scaling-notifications"
}
```

Save the TopicArn:

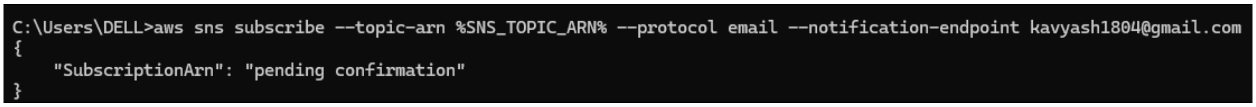
```
set SNS_TOPIC_ARN=arn:aws:sns:xxxxxxxxxxxxxxxxxxxx
```



```
C:\Users\DELL>set SNS_TOPIC_ARN=arn:aws:sns:ap-south-1:933376059751:Kavya-scaling-notifications
```

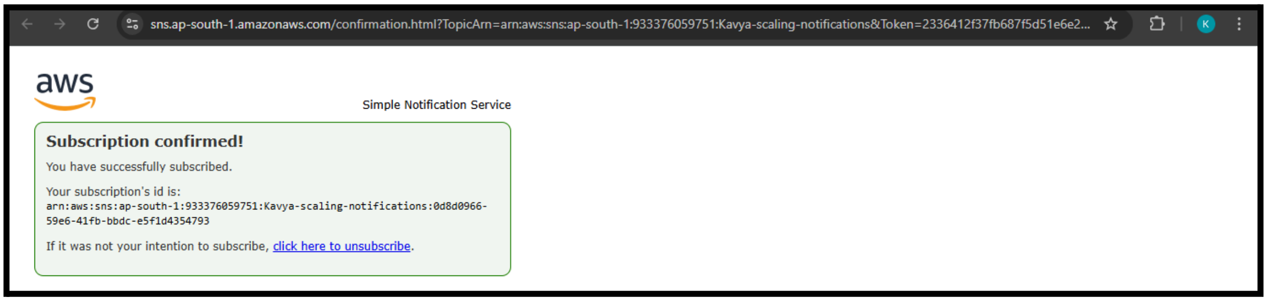
Now subscribe your email to the topic:

```
aws sns subscribe --topic-arn %SNS_TOPIC_ARN% --protocol email
--notification-endpoint your-email@example.com
```



```
C:\Users\DELL>aws sns subscribe --topic-arn %SNS_TOPIC_ARN% --protocol email --notification-endpoint kavyash1804@gmail.com
{
    "SubscriptionArn": "pending confirmation"
}
```

Confirm your email address from your account.



sns.ap-south-1.amazonaws.com/confirmation.html?TopicArn=arn:aws:sns:ap-south-1:933376059751:Kavya-scaling-notifications&Token=2336412f37fb687f5d51e6e2...

Simple Notification Service

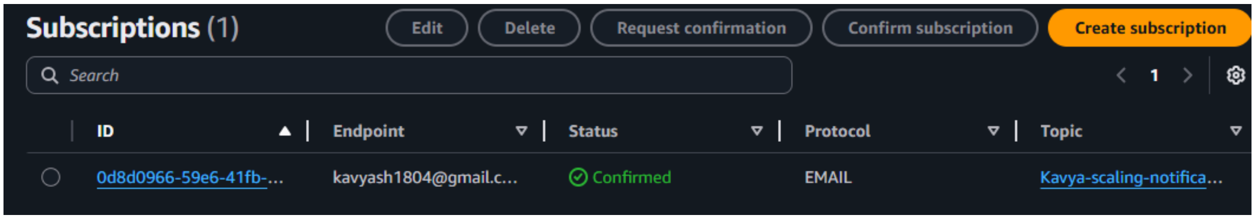
Subscription confirmed!

You have successfully subscribed.

Your subscription's id is:
`arn:aws:sns:ap-south-1:933376059751:Kavya-scaling-notifications:0d8d0966-59e6-41fb-bbdc-e5f1d4354793`

If it was not your intention to subscribe, click here to unsubscribe.

To verify: go to the AWS SNS console > Subscriptions



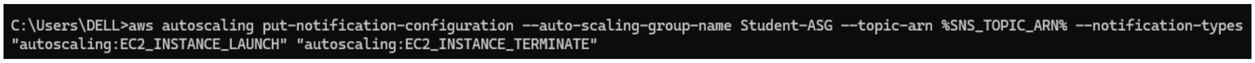
	ID	Endpoint	Status	Protocol	Topic
○	0d8d0966-59e6-41fb-...	kavyash1804@gmail.c...	✓ Confirmed	EMAIL	Kavya-scaling-notifica...

STEP 19: Setup Auto Scaling notifications

Run the following command to configure notifications for Auto Scaling Groups:

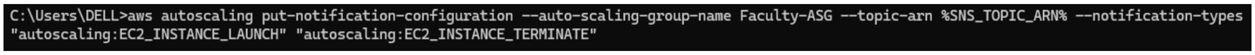
```
aws autoscaling put-notification-configuration ^
    --auto-scaling-group-name YourName-ASG ^
    --topic-arn %SNS_TOPIC_ARN% ^
    --notification-types "autoscaling:EC2_INSTANCE_LAUNCH" ^
    "autoscaling:EC2_INSTANCE_TERMINATE"
```

For Student-ASG



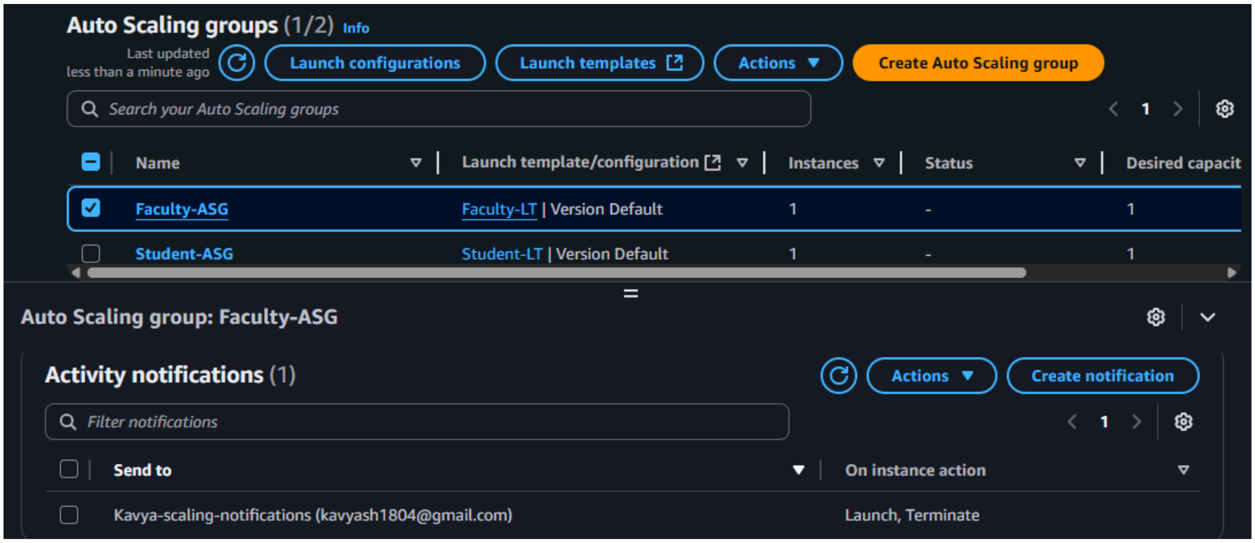
```
C:\Users\DELL>aws autoscaling put-notification-configuration --auto-scaling-group-name Student-ASG --topic-arn %SNS_TOPIC_ARN% --notification-types
"autoscaling:EC2_INSTANCE_LAUNCH" "autoscaling:EC2_INSTANCE_TERMINATE"
```

For Faculty-ASG



```
C:\Users\DELL>aws autoscaling put-notification-configuration --auto-scaling-group-name Faculty-ASG --topic-arn %SNS_TOPIC_ARN% --notification-types
"autoscaling:EC2_INSTANCE_LAUNCH" "autoscaling:EC2_INSTANCE_TERMINATE"
```

Verify on the AWS EC2 console > Select an auto scaling group > go to activity



Auto Scaling groups (1/2) Info

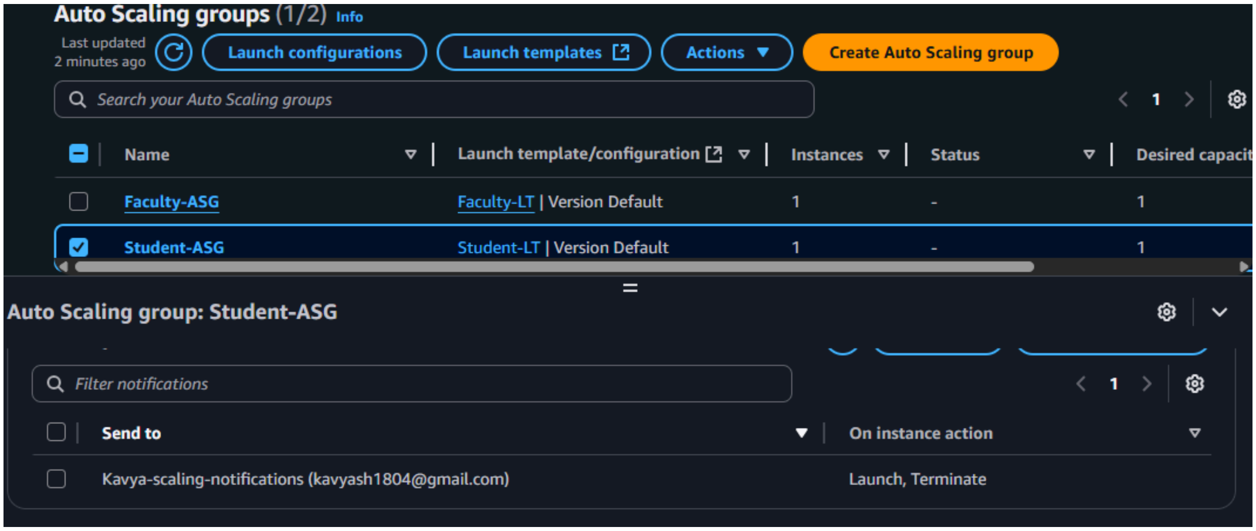
Last updated less than a minute ago

☐	Name	Launch template/configuration	Instances	Status	Desired capacit
☑	Faculty-ASG	Faculty-LT | Version Default	1	-	1
☐	Student-ASG	Student-LT | Version Default	1	-	1

Auto Scaling group: Faculty-ASG

Activity notifications (1)

☐	Send to	On instance action
☐	Kavya-scaling-notifications (kavyash1804@gmail.com)	Launch, Terminate



Auto Scaling groups (1/2) Info

Last updated 2 minutes ago

☐	Name	Launch template/configuration	Instances	Status	Desired capacit
☐	Faculty-ASG	Faculty-LT | Version Default	1	-	1
☑	Student-ASG	Student-LT | Version Default	1	-	1

Auto Scaling group: Student-ASG

☐	Send to	On instance action
☐	Kavya-scaling-notifications (kavyash1804@gmail.com)	Launch, Terminate

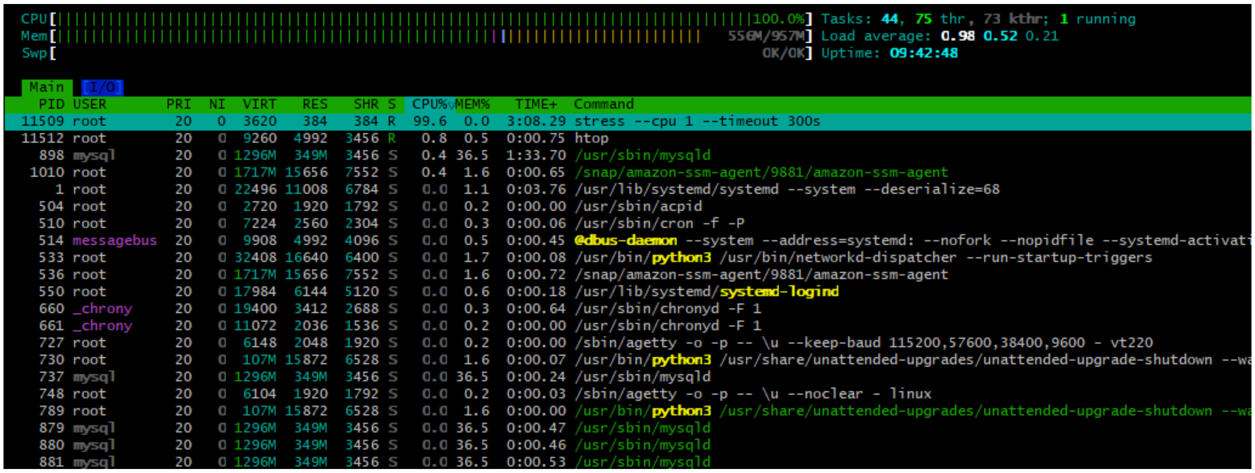
Auto Scaling notifications are configured for both the auto scaling groups.

STEP 20: Test auto-scaling by applying load to your servers

Login to the Faculty-login server on bash using the ssh -i keyname ubuntu@public-ip command.

Run the following commands to increase CPU utilization :

- sudo -i
- apt update && apt upgrade -y
- apt install stress -y
- stress --cpu 1 --timeout 120s
- htop

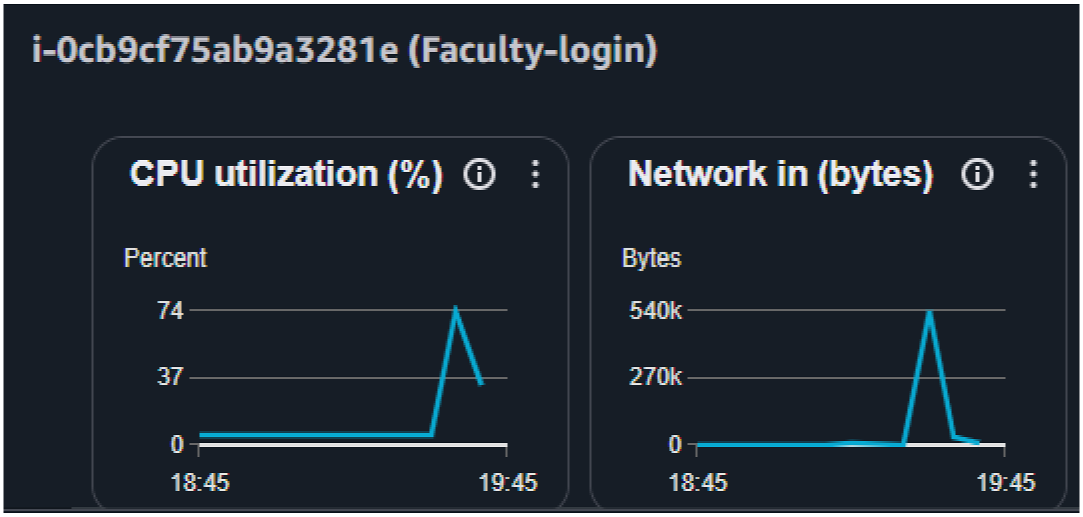


```
  CPU[|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||100.0%] Tasks: 44, 75 thr, 73 kthr; 1 running
  Mem[||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||   556M/957M] Load average: 0.98 0.52 0.21
  Swp[                                                                                   0K/0K] Uptime: 09:42:48

  Main  [I/O]
    PID USER       PRI  NI  VIRT   RES   SHR S  CPU%▽MEM%   TIME+  Command
  11509 root        20   0  3620   384   384 R  99.6  0.0  3:08.29 stress --cpu 1 --timeout 300s
  11512 root        20   0  9260  4992  3456 R   0.8  0.5  0:00.75 htop
    898 mysql       20   0 1296M  349M  3456 S   0.4 36.5  1:33.70 /usr/sbin/mysqld
   1010 root        20   0 1717M 15656  7552 S   0.4  1.6  0:00.65 /snap/amazon-ssm-agent/9881/amazon-ssm-agent
      1 root        20   0 22496 11008  6784 S   0.0  1.1  0:03.76 /usr/lib/systemd/systemd --system --deserialize=68
    504 root        20   0  2720  1920  1792 S   0.0  0.2  0:00.00 /usr/sbin/acpid
    510 root        20   0  7224  2560  2304 S   0.0  0.3  0:00.06 /usr/sbin/cron -f -P
    514 messagebus  20   0  9908  4992  4096 S   0.0  0.5  0:00.45 @dbus-daemon --system --address=systemd: --nofork --nopidfile --systemd-activati
    533 root        20   0 32408 16640  6400 S   0.0  1.7  0:00.08 /usr/bin/python3 /usr/bin/networkd-dispatcher --run-startup-triggers
    536 root        20   0 1717M 15656  7552 S   0.0  1.6  0:00.72 /snap/amazon-ssm-agent/9881/amazon-ssm-agent
    550 root        20   0 17984  6144  5120 S   0.0  0.6  0:00.18 /usr/lib/systemd/systemd-logind
    660 _chrony     20   0 19400  3412  2688 S   0.0  0.3  0:00.64 /usr/sbin/chronyd -F 1
    661 _chrony     20   0 11072  2036  1536 S   0.0  0.2  0:00.00 /usr/sbin/chronyd -F 1
    727 root        20   0  6148  2048  1920 S   0.0  0.2  0:00.00 /sbin/agetty -o -p -- \u --keep-baud 115200,57600,38400,9600 - vt220
    730 root        20   0  107M 15872  6528 S   0.0  1.6  0:00.07 /usr/bin/python3 /usr/share/unattended-upgrades/unattended-upgrade-shutdown --wa
    737 mysql       20   0 1296M  349M  3456 S   0.0 36.5  0:00.24 /usr/sbin/mysqld
    748 root        20   0  6104  1920  1792 S   0.0  0.2  0:00.03 /sbin/agetty -o -p -- \u --noclear - linux
    789 root        20   0  107M 15872  6528 S   0.0  1.6  0:00.00 /usr/bin/python3 /usr/share/unattended-upgrades/unattended-upgrade-shutdown --wa
    879 mysql       20   0 1296M  349M  3456 S   0.0 36.5  0:00.47 /usr/sbin/mysqld
    880 mysql       20   0 1296M  349M  3456 S   0.0 36.5  0:00.46 /usr/sbin/mysqld
    881 mysql       20   0 1296M  349M  3456 S   0.0 36.5  0:00.53 /usr/sbin/mysqld
```

To exit: press **q**

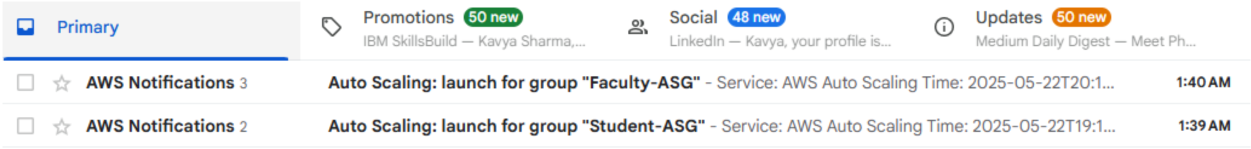
CPU utilization can also be reviewed on the AWS EC2 console.

- Go to instances > select “Faculty-login” > go to Monitoring



Repeat the same steps to test auto scaling on the Student-login server.

Check your email for SNS notifications when scaling events occur.


Primary
Promotions 50 new
IBM SkillsBuild — Kavya Sharma,...
Social 48 new
LinkedIn — Kavya, your profile is...
Updates 50 new
Medium Daily Digest — Meet Ph...

□ ☆ AWS Notifications 3 Auto Scaling: launch for group "Faculty-ASG" - Service: AWS Auto Scaling Time: 2025-05-22T20:1... 1:40 AM

□ ☆ AWS Notifications 2 Auto Scaling: launch for group "Student-ASG" - Service: AWS Auto Scaling Time: 2025-05-22T19:1... 1:39 AM

- Kavya Sharma

Rayat Bahra University