BLACKBUCKS INTERNSHIP REPORT

An Architecture of 6 **VPCS** with 6 **Instances** and

6 **Subnets** with **Peering Connections**

(6-TIER ARCHITECT)

SUBMITTED BY

Ms. DIVVELA DEVIKA RANI **RegdNo:21B91A5438**

Ms. ELIKA LAVANYA **RegdNo:21B91A5442**

Ms. AMBATI RAMA GAYATHRI **RegdNo:21B91A5407**

UNDER THE GUIDANCE OF MR. AASHU DEV

B. Tech, AWS solution Architect (2x) certified,

AWS Academy Accredited Educator

BLACKBUCKS INTERNSHIP WORK

Title:

Connecting multi accounts using peering connections

(5.2-Tier Architecture)

Abstract:

The above Architecture is to create 6 VPCs with 6 Subnets and 6 Instances with required Peering Connections. And the creation of Instances is done by using the service Cloud Shell . And the termination of Instances is done by using Lambda function. The Architecture is done in two different accounts and two different regions (Mumbai, Stockholm).

Table of Contents

Introduction To AWS

AWS stands for Amazon Web Services, it needs no formal introduction, given its immense popularity. The leading cloud provider in the marketplace is Amazon Web Services. It provides over 170 AWS services to the developers so they can access them from anywhere at the time of need.

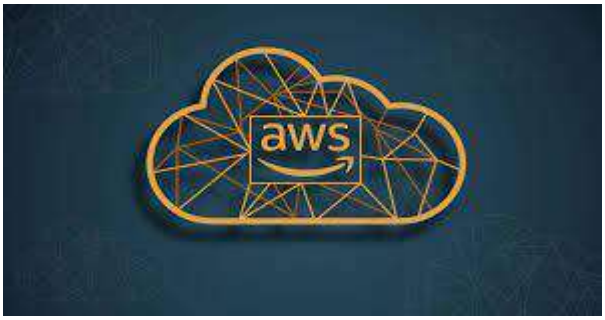
AWS has customers in over 190 countries worldwide, including 5000 ed-tech institutions and 2000 government organizations. Many companies like ESPN, Adobe, Twitter, Netflix, Facebook, BBC, etc., use AWS services.

For example, Adobe creates and updates software without depending upon the IT teams. It uses its services by offering multi-terabyte operating environments for its clients. By deploying its services with Amazon services, Adobe integrated and operated its software in a simple manner.

An analogy: think of electricity services….

You simply plug into a vast electrical grid managed by experts to get a low cost , reliable power supply – available to you with much greater efficiency than you could generate on you own.

Power is a utility service – available to you on-demand and pay only for what you use.



What is Cloud Computing?

Cloud Computing is the delivery of online services (such as servers, databases, software) to users. With the help of cloud computing, storing data on local machines is not required. It helps you access data from a remote server. Moreover, it is also used to store and access data from anywhere across the world.

What is AWS?

AWS Meaning: The Amazon Web Services (AWS) platform provides more than 200 fully featured services from data centers located all over the world, and is the world's most comprehensive cloud platform.

Amazon web service is an online platform that provides scalable and cost-effective cloud computing solutions.

AWS is a broadly adopted cloud platform that offers several on-demand operations like compute power, database storage, content delivery, etc., to help corporates scale and grow.

History of AWS

- In the year 2002 - AWS services were launched
- In the year 2006- AWS cloud products were launched
- In the year 2012 - AWS had its first customer event
- In the year 2015- AWS achieved $4.6 billion
- In the year 2016- Surpassed the $10 billion revenue target
- In the year 2016- AWS snowball and AWS snowmobile were launched
- In the year 2019- Released approximately 100 cloud services

Why AWS?

There are several reasons why AWS has become a popular choice for cloud computing:

❖ Easy to use

AWS is designed to allow application providers, ISVs, and vendors to quickly and securely host your applications – whether an existing application or a new SaaS-based application. You can use the AWS Management Console or well-documented web services APIs to access AWS's application hosting platform.

❖ Flexible

AWS enables you to select the operating system, programming language, web application platform, database, and other services you need. With AWS, you receive a virtual environment that lets you load the software and services your application requires. This eases the migration process for existing applications while preserving options for building new solutions.

❖ Cost-Effective

You pay only for the compute power, storage, and other resources you use, with no long-term contracts or up-front commitments. For more information on comparing the costs of other hosting alternatives with AWS, see the AWS Economics Center.

❖ Reliable

With AWS, you take advantage of a scalable, reliable, and secure global computing infrastructure, the virtual backbone of Amazon .com 's multi – billion dollar online business that has been honed for over a decade.

❖ Scalable and high-performance

Using AWS tools, Auto Scaling, and Elastic Load Balancing, your application can scale up or down based on demand. Backed by Amazon's massive infrastructure, you have access to compute and storage resources when you need them.

❖ Secure

AWS utilizes an end-to-end approach to secure and harden our infrastructure, including physical, operational, and software measures. For more information, see the AWS Security Center.

❖ Integration and Ecosystem

AWS integrates well with various third-party tools, technologies, and services. It offers extensive APIs and SDKs, making it easier to integrate AWS services into existing applications or build new solutions from scratch. The AWS ecosystem also includes a vibrant community, documentation, training resources, and support services, facilitating development and troubleshooting.

These factors, among others, contribute to the popularity and success of AWS as a cloud computing provider. However, it's important to note that the choice of cloud provider should be based on your specific needs, requirements, and preferences. It's worth evaluating multiple cloud platforms to determine the best fit for your organization.

Advantages of AWS:

1. AWS provides a user-friendly programming model, architecture, database as well as operating system that has been already known to employers.
2. AWS is a very cost-effective service. There is no such thing as long-term commitments for anything you would like to purchase.
3. It offers billing and management for the centralized sector, hybrid computing, and fast installation or removal of your application in any location with few clicks.
4. There is no need to pay extra money on running data servers by AWS.

5. AWS offers a total ownership cost at very reasonable rates in comparison to other private cloud servers.

Disadvantages of AWS:

1. AWS has supportive paid packages for intensive or immediate response. Thus, users might need to pay extra money for that.
2. There might be some cloud computing problems in AWS especially when you move to a cloud Server such as backup protection, downtime, and some limited control.
3. From region to region, AWS sets some default limitations on resources such as volumes, images, or snapshots.
4. If there is a sudden change in your hardware system, the application on the cloud might not offer great performance.

Applications of AWS

The most common applications of AWS are storage and backup, websites, gaming, mobile, web, and social media applications. Some of the most crucial applications in detail are as follows:

- **Storage and Backup**

One of the reasons why many businesses use AWS is because it offers multiple types of storage to choose from and is easily accessible as well. It can be used for storage and file indexing as well as to run critical business applications.

- **Websites**

Businesses can host their websites on the AWS cloud, similar to other web applications.

- **Gaming**

There is a lot of computing power needed to run gaming applications. AWS makes it easier to provide the best online gaming experience to gamers across the world.

- **Mobile, Web and Social Applications**

A feature that separates AWS from other cloud services is its capability to launch and scale mobile, e-commerce, and SaaS applications. API-driven code on AWS can enable companies to build uncompromisingly scalable applications without requiring any OS and other systems.

- **Big Data Management and Analytics (Application)**

  - Amazon Elastic Map Reduced to process large amounts of data via the Hadoop framework.
  - Amazon Kinesis to analyze and process the streaming data.
  - AWS Glue to handle, extract, transform and load jobs.
  - Amazon Elasticsearch Service to enable a team to perform log analysis, and tool monitoring with the help of the open source tool ,Elastic-search.
  - Amazon Athena to query data.
  - Amazon Quick Sight to visualize data.

- **Artificial Intelligence**

- Amazon Lex to offer voice and text chatbot technology.
- Amazon Polly to translate text-to-speech translation such as Alexa Voice Services and echo devices.
- Amazon Rekognition to analyze the image and face.

Messages and Notifications

- Amazon Simple Notification Service (SNS) for effective business or core communication.
- Amazon Simple Email Service (SES) to receive or send emails for IT professionals and marketers.
- Amazon Simple Queue Service (SQS) to enable businesses to subscribe or publish messages to end users.

Augmented Reality and Virtual Reality

- Amazon Sumerian service enables users to make the use of AR and VR development tools to offer 3D web applications, E-commerce & sales applications, Marketing, Online education, Manufacturing, Training simulations, and Gaming.

Game Development

- AWS game development tools are used by large game development companies that offer developer back-end services, analytics, and various developer tools.
- AWS allows developers to host game data as well as store the data to analyze the gamer's performance and develop the game accordingly.

Internet of Things

- AWS IoT service offers a back-end platform to manage IoT devices as well as data ingestion to database services and AWS storage.

- AWS IoT Button offers limited IoT functionality to hardware.
- AWS Greengrass offers AWS computing for IoT device installation.

Companies Using AWS

Whether it's technology giants, startups, government, food manufacturers or retail organizations, there are so many companies across the world using AWS to develop, deploy and host applications. According to Amazon, the number of active AWS users exceeds 1,000,000. Here is a list of companies using AWS:

- Netflix
- Intuit
- Coinbase
- Finra
- Johnson & Johnson
- Capital One
- Adobe
- Airbnb
- AOL
- Hitachi

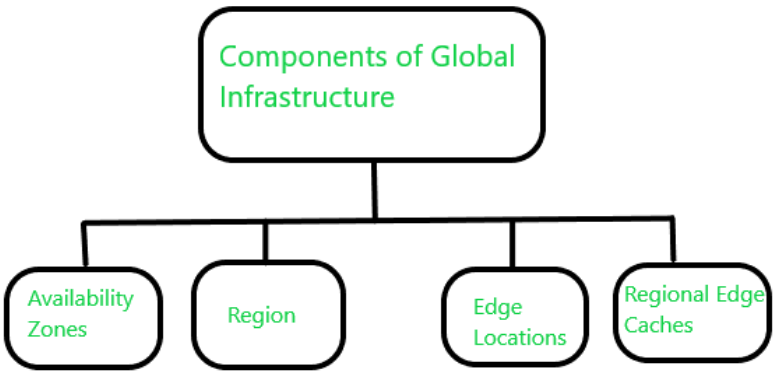
AWS Global Infrastructure

• AWS is a cloud computing platform which is globally available.

• Global infrastructure is a region around the world in which AWS is based. Global infrastructure is a bunch of high-level IT services which is shown below:

• AWS is available in 19 regions, and 57 availability zones in December 2018 and 5 more regions 15 more availability zones for 2019.

The following are the components that make up the AWS infrastructure:



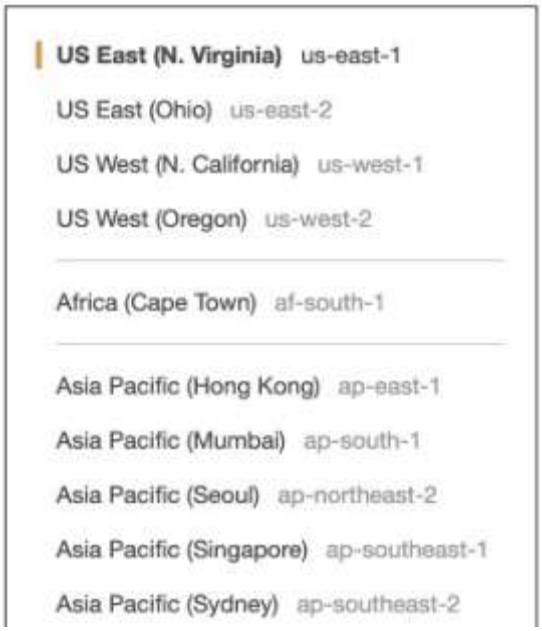
The infrastructure of AWS contains regions, availability zones, data centers, edge locations, or points of presence.

AWS Regions

AWS region is a cluster of data centers. It has regions all around the world. Regions name can be like “us-east-1”, “EU-west-1” etc. Most AWS services are region-scoped.

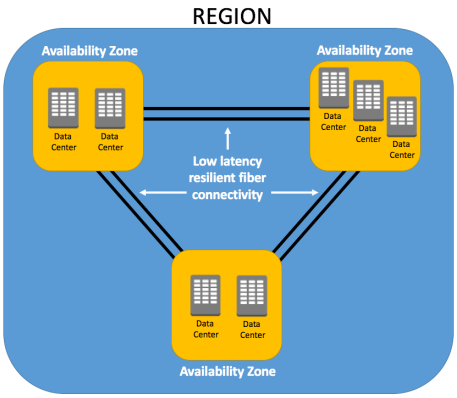
If you want to launch a new application, then you should consider the below parameters to choose a region:

- **Compliance with data governance and legal requirements**: Data never leaves a region without your explicit permission.
- **Proximity to customers:** Reduced latency.



- **Compliance with data governance and legal requirements**: Data never leaves a region without your explicit permission.
- **Proximity to customers:** Reduced latency.

AWS Availability Zones



Each region has many availability zones (usually 3, min is 2, max is 6). Example: ap-southeast-2a, ap-southeast-2b, ap-southeast-2c etc. Each availability zone (AZ) is one or more discrete data centers with redundant power, networking, and connectivity. They're separate from each other so that they're isolated from disasters. They're connected with high bandwidth, ultra-low latency networking.

AWS Points of Presence (Edge Locations)

• Edge locations are the endpoints for AWS used for caching content.

• Edge locations consist of CloudFront and Amazon's Content Delivery Network (CDN).

• Edge locations are more than regions. Currently, there are over 150 edge locations.

• Edge location is not a region but a small location that AWS have. It is used for caching the content.

• Edge locations are mainly located in most of the major cities to distribute the content to end users with reduced latency.

• For example, some user accesses your website from Singapore; then this request would be redirected to the edge location closest to Singapore where cached data can be read.

Regional Edge Cache

• AWS announced a new type of edge location in November 2016, known as a Regional Edge Cache.

• Regional Edge cache lies between CloudFront Origin servers and the edge locations.

• A regional edge cache has a larger cache than an individual edge location

. • Data is removed from the cache at the edge location while the data is retained at the Regional Edge Caches.

• When the user requests the data, then data is no longer available at the edge location. Therefore, the edge location retrieves the cached data from the regional edge cache instead of the Origin servers that have high latency.

List Of AWS Services:

❖ Amazon EC2 (Elastic Cloud Compute)

Amazon EC2 is the fastest cloud computing service provided by AWS. It offers virtual, secure, reliable, and resizable servers for any workload. Through this service, it becomes easy for developers to access resources and also facilitates web-scale cloud computing. This comes with the best suitable processors, networking facilities, and storage systems. Developers can quickly and dynamically scale capacities as per business needs. It has over 500 instances and you can also choose the latest processor, operating system, storage, and networking to help you choose according to the needs of the business. Also, with Amazon EC2, you only have to pay for what you use, and also as per the time period, scale with amazon EC2 auto-scaling has optimal storage and can optimize CPU configurations.

- ## Amazon VPC (Virtual Private Cloud)

Another AWS service is Amazon VPC (Virtual Private Cloud) which is an isolated cloud resource, it enables you to set up an isolated section where you can deploy AWS resources at scale in a virtual environment. This service is responsible to control the virtual networking environment such as resource placement, security, and connectivity. Security can be improved by applying rules for outbound and inbound connections. Also, it detects anomalies in the patterns, troubleshoots network connections, prevents data leakage, and handles configuration issues. Using VPC, you get complete access to control the environment, such as choosing IP address, subset creation, and route table arrangement.

- ## Amazon S3 (Simple Storage Service)

With Amazon, it has become easy to store data anytime, anywhere. Amazon S3 (Simple Storage Service), one of the best services provided by AWS is an object storage service offering scalability, availability, security, and high-performing. You can also retrieve data, data here is stored in “storage classes” where there’s no requirement of extra investment and you can also manage it well. Amazon S3 is the perfect fit for big businesses where a large amount of data is managed for varied purposes. It comes with handling any volume of data with its robust access controls, and replication tools prevent accidental deletion, and also maintains data version controls.

- **Amazon RDS (Relational Database Services)**

Amazon RDS (Relational Database Service) is another service provided by AWS which is a managed database for PostgreSQL, MariaDB, MySQL, and Oracle. Using Amazon RDS, you can set up, operate, and scale databases in the cloud. It provides high performance by automating the tasks like database setup, hardware provisioning, patching, and backups. Also, it helps in cost optimization by providing high availability, compatibility, and security for resources, and there’s no need to install and manage the database software. during its usage. As per the need, you can easily choose any engine out of 15+ engines some of them being MySQL, PostgreSQL, Oracle, etc. It is a highly secure and easily available AWS service.

- **Amazon IAM (Identity and Access Management)**

Amazon IAM (Identity and Access Management) allows users to securely access and manage resources. To achieve complete access to the tools and resources provided by AWS, AWS IAM is the best AWS service. It gives you the right to have control over who has authorization (signed in) and authentication (has permissions) access to the resources. It comes with attribute-based access control which helps you to create separate permissions on the basis of the user's attributes such as job role, department, etc. Through this, you can allow or deny access given to users. AWS IAM has complete access or is a central manager for refining permissions across AWS. He /She handles who can access what.

- **Amazon EBS (Elastic Block Store)**

Amazon EBS is the next service provided by AWS which is a block storage solution specifically designed for Amazon EC2. Throughout a workload of any size, Amazon EBS helps to securely manage transactions. You can handle diverse workloads, be it relational, non-relational, or business applications. You get to choose between five different volume types so as to achieve effectiveness and optimum cost. It helps to resize workloads for big data analytics engines such as Hadoop and Spark. Its lifecycle management creates policies to create and manage backups effectively. It supports high-performance scaling workloads such as Microsoft, and SAP products.

- **Amazon Lambda**

Another promising service by AWS is Amazon Lambda which is a serverless and event-driven computing service that lets you run code for virtual applications or backend services automatically. You need to worry about servers and clusters when working with solutions using Amazon Lambda. It is also cost-effective where you have to only pay for the services you use. As a user, your responsibility is to just upload the code and Lambda handles the rest. Using Lambda, you get precise software scaling and extensive availability. With hundreds to thousands of workloads per second, AWS Lambda responsibly handles code execution requests. It is one of the best services provided by AWS for developers.

- **Amazon EFS (Elastic File System)**

Amazon EFS (Elastic File System) is a simple and serverless system where you can create and configure file systems without provisioning, deploying, patching, and maintaining. It is a scalable NFS file system made for use in AWS cloud services and on-premises resources. Also, it has no minimum fee or setup charge. You pay for the storage you use such as –

- for provisioned throughput
- automatically expand and shrink as per the addition and removal of files
- read and write access to data stored in Infrequent Access storage classes

It is a scalable service where you can scale up to petabytes without thinking about the performance of the application.

- **Amazon SNS (Simple Notification Service)**

It is a web service provided by AWS, which is a fully managed solution for messaging having low-cost infrastructure. It is used for bulk message delivery and direct chat with the customers through system-to-system or app-to-person between decoupled microservice apps. It is used to easily set up, operate, and send notifications from the cloud. It is a messaging service between Application to Application (A2A) and Application to Person (A2Person), and sends notifications in two ways – A2A and A2P. A2P allows many-to-many messaging between microservices, distributed systems, and event-driven serverless applications, allowing you to send messages to customers with SMS texts, email, and push notifications.

- **Amazon Auto-Scaling**

Amazon Auto-Scaling is one of the best services provided by AWS, it examines the applications and adjusts its capacity accordingly which is to maintain predictable performance at the lowest possible cost. It becomes easy to scale up applications for multiple resources across services in seconds. To meet the demands of the business, it scales computing capacity and this can be achieved by adding or removing EC2 instances automatically. There are two types of scaling – dynamic (presently changing demands) and predictive (response based on predictions) scaling. It can be used with Amazon EC2 Auto Scaling to dynamically scale your Amazon EC2 instances and also it receives the right resource at the right time.

- **Dynamo DB**

DynamoDB is a serverless, document database key-value NoSQL database that is designed to run high-performance applications. It can manage up to 10 trillion requests on a daily basis and support thresholds of more than 20 million requests per second. DynamoDB has built-in security with a fully-managed multi-master, multi-region, durable database, and in-memory archiving for web-scale applications. It has in-built tools which is used to generate actionable insights, useful analytics, and monitor traffic trends. It offers built-in security, continuous backups, automated multi-region replication, data import and export, and in-memory caching.

- **Amazon Elastic Beanstalk**

Amazon Elastic Beanstalk is an AWS service used for deployment and scaling web applications developed using Java, PHP, Python, Docker, etc. It supports running and managing web applications. You just need to upload your code and the deployment part is handled by Elastic Beanstalk (from capacity provisioning, load balancing, and auto-scaling to application health monitoring). It is the best service for developers since it takes care of the servers, load balancers, and firewalls. Also, you can have control over AWS assets and the other resources required for the application. You get the benefit of paying for what you use, thus maintaining cost-effectiveness.

- **Amazon Cloud watch**

Amazon CloudWatch detects uncommon changes in the environment, set an alert, troubleshoots issues, and take automated actions. With this, you can track the complete stack, and use logs, alarms, and events data to take actions and thereby focusing on building the application, resulting in the growth of the business. It is the best service designed for developers, DevOps engineers, site reliability engineers, and IT managers. With Amazon CloudWatch, you can detect anomalies in the environment. With this single platform, you can monitor all AWS resources and applications quickly. It monitors application performance and optimizes resources.

- **Amazon Cloud Shell**

Cloud Shell is a lightweight, award-winning infrastructure automation tool. Cloud Shell is offered on the AWS Marketplace to accelerate and simplify environment provisioning by enabling your teams to create self-service, on-demand replicas of full-stack infrastructure environments for AWS and hybrid cloud in just one click. Innovators everywhere use Cloud Shell solutions to:

- Accelerate time to market up to 50% faster by automating the set-up and tear-down of complex hybrid environments on AWS.

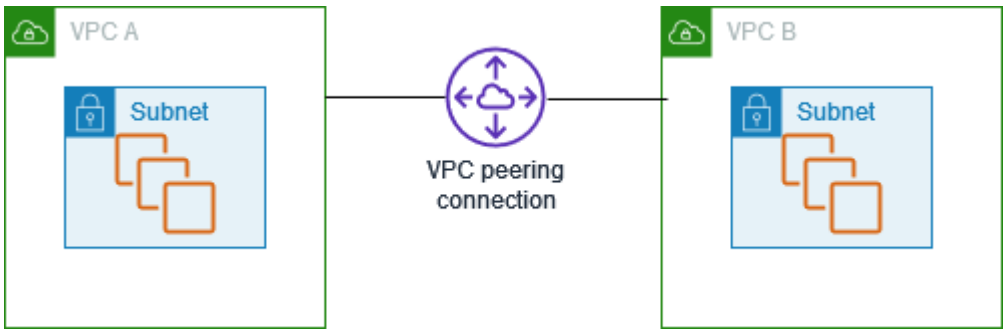
- Lift productivity by providing a self-service catalog of standardized environment blueprints used for development and sales demos.
- Improve utilization of cloud resources to attain 3-5x ROI.

Implementation of the Architecture of 6 VPCs with Peering Connection(6TIER-ARCHITECT):

Introduction:

Peering Connection:

A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them using private IPv4 addresses or IPv6 addresses. Instances in either VPC can communicate with each other as if they are within the same network. You can create a VPC peering connection between your own VPCs, or with a VPC in another AWS account. The VPCs can be in different Regions (also known as an inter-Region VPC peering connection).



Use of IAM in Lambda Function:

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be authenticated (signed in) and authorized (have permissions) to use Lambda resources. IAM is an AWS service that you can use with no additional charge.

CLOUD SHELL:

AWS Cloud Shell is a browser-based shell that makes it easier to securely manage, explore, and interact with your AWS resources. Cloud Shell is pre-authenticated with your console credentials. Whilst Cloud Shell permits outbound connections to the internet, it does not support inbound connections. Furthermore, it does not allow connections to resources within private VPC subnets.

Services used:

1.VPC (virtual private cloud)

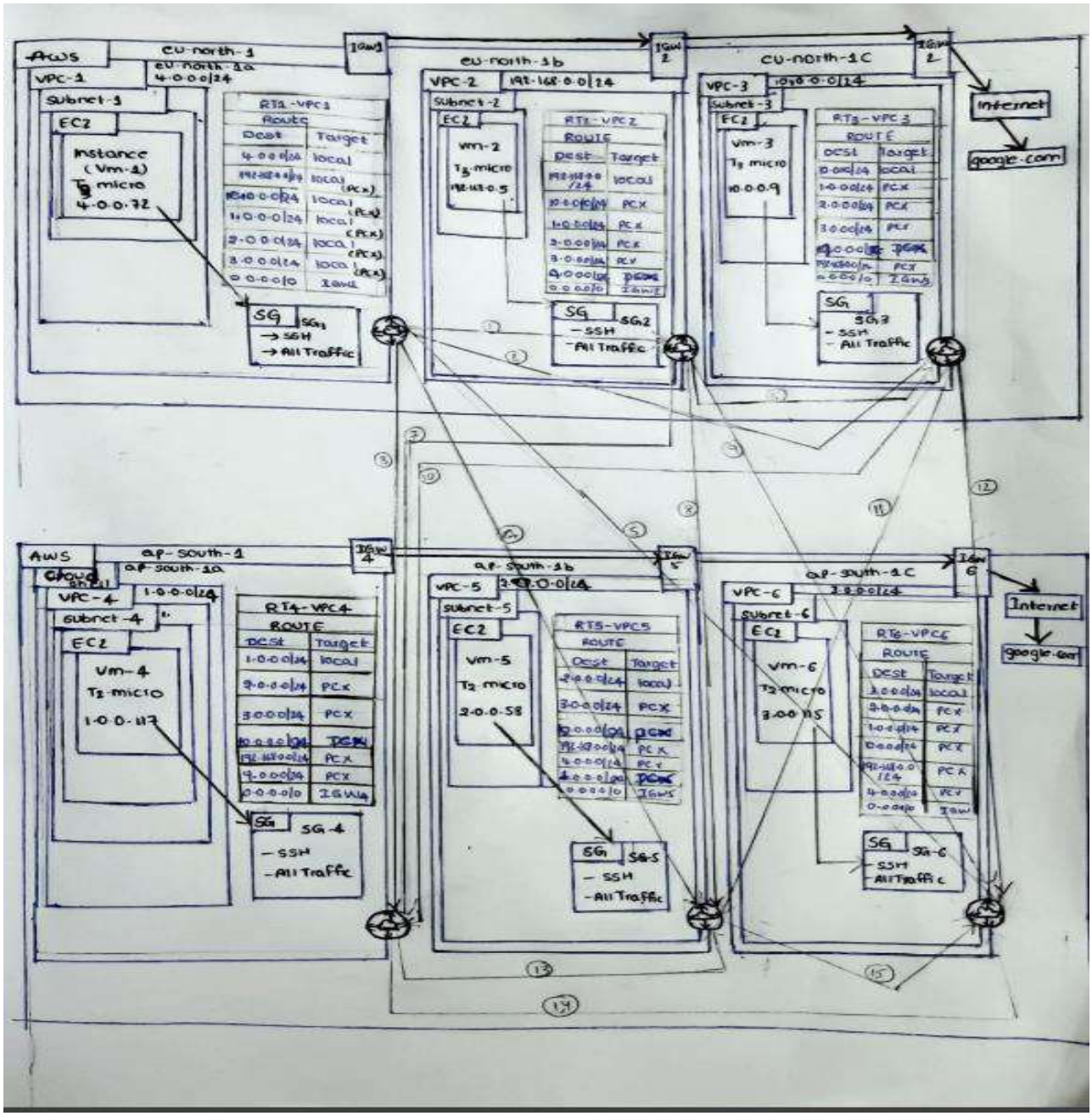
2.EC2 (Elastic compute cloud)

3.Lambda function

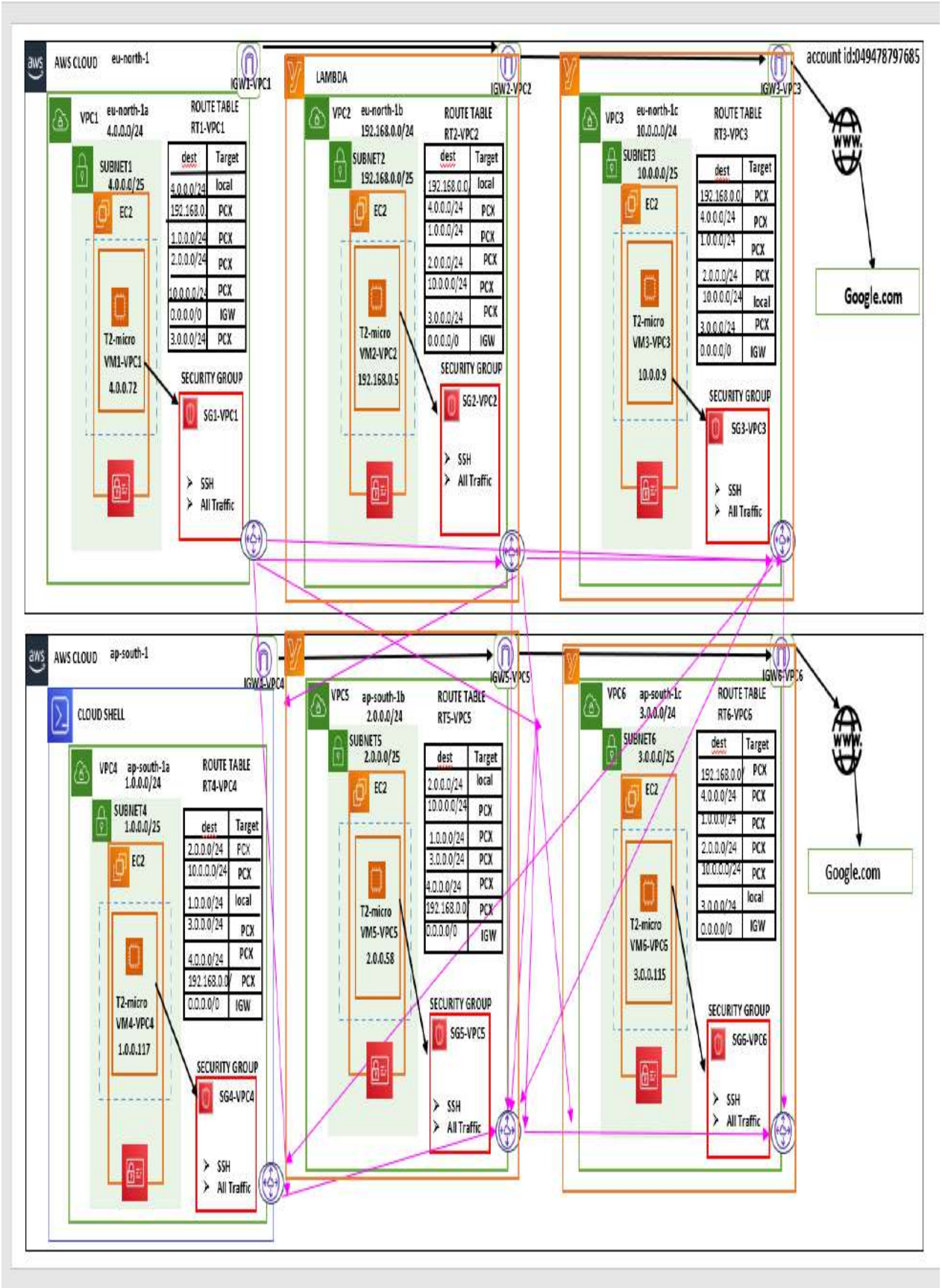
4.IAM (Identity and access management)

5.Cloud shell

ROUGH ARCHITECT:



FINAL ARCHITECT:


AWS CLOUD
eu-north-1
IGW1-VPC1
LAMBDA
IGW2-VPC2
IGW3-VPC3
account id:049478797685
VPC1
eu-north-1a
4.0.0.0/24
ROUTE TABLE
RT1-VPC1
SUBNET1
4.0.0.0/25
EC2
T2-micro
VM1-VPC1
4.0.0.72
SECURITY GROUP
SG1-VPC1
SSH
All Traffic
VPC2
eu-north-1b
192.168.0.0/24
ROUTE TABLE
RT2-VPC2
SUBNET2
192.168.0.0/25
EC2
T2-micro
VM2-VPC2
192.168.0.5
SECURITY GROUP
SG2-VPC2
SSH
All Traffic
VPC3
eu-north-1c
10.0.0.0/24
ROUTE TABLE
RT3-VPC3
SUBNET3
10.0.0.0/25
EC2
T2-micro
VM3-VPC3
10.0.0.9
SECURITY GROUP
SG3-VPC3
SSH
All Traffic
www.
Google.com
AWS CLOUD
ap-south-1
IGW4-VPC4
IGW5-VPC5
IGW6-VPC6
CLOUD SHELL
VPC4
ap-south-1a
1.0.0.0/24
ROUTE TABLE
RT4-VPC4
SUBNET4
1.0.0.0/25
EC2
T2-micro
VM4-VPC4
1.0.0.117
SECURITY GROUP
SG4-VPC4
SSH
All Traffic
VPC5
ap-south-1b
2.0.0.0/24
ROUTE TABLE
RT5-VPC5
SUBNET5
2.0.0.0/25
EC2
T2-micro
VM5-VPC5
2.0.0.58
SECURITY GROUP
SG5-VPC5
SSH
All Traffic
VPC6
ap-south-1c
3.0.0.0/24
ROUTE TABLE
RT6-VPC6
SUBNET6
3.0.0.0/25
EC2
T2-micro
VM6-VPC6
3.0.0.115
SECURITY GROUP
SG6-VPC6
SSH
All Traffic
www.
Google.com

Workflow of the Project:

- Create a 3VPCs (VPC1 , VPC2 ,VPC3) in One account and edit the route tables associated with it. Create a subnets and name it as public subnet. Edit the corresponding subnet associations, create an internet gateways and attach it to custom VPCs. Now launch an ec2 instance with created VPCs.
- Create a 3VPCs (VPC4 , VPC5 ,VPC6) in another account and edit the route tables associated with it. Create a subnets and name it as public subnet. Edit the corresponding subnet associations, create an internet gateways and attach it to custom VPCs. Now launch an ec2 instance with created VPCs.

- **POSSIBLE PEERING CONNECTIONS**

ACCOUNT1 (eu-north-1)	ACCOUNT2 (ap-south-1)	Different accounts and Cross Regions
VPC1-VPC2	VPC4-VPC5	VPC1-VPC4
VPC1-VPC3	VPC4-VPC6	VPC1-VPC5
VPC2-VPC3	VPC5-VPC6	VPC1-VPC6
		VPC2-VPC4
		VPC2-VPC5
		VPC2-VPC6
		VPC3-VPC4
		VPC3-VPC5
		VPC3-VPC6

Implementation of the Project

Service 1: VPC

To create a VPC with the name "VPC1," follow these steps:

• Sign into the AWS Management Console.

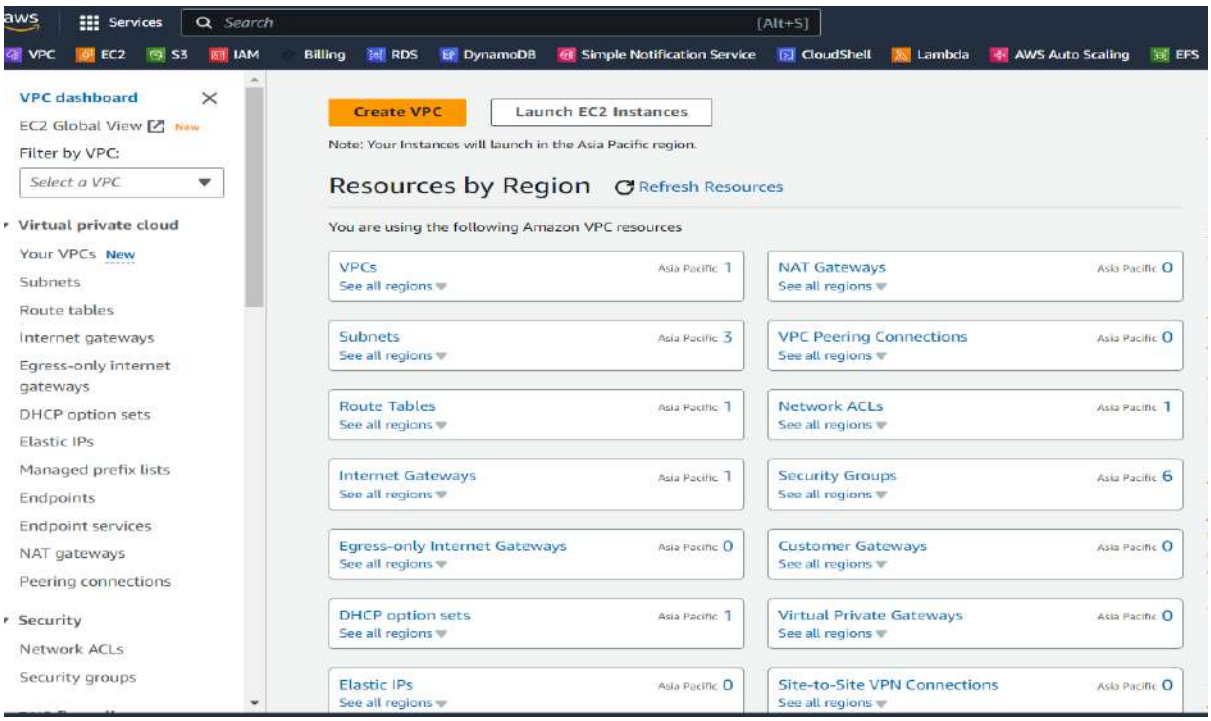
• Open the Virtual Private Cloud Service.

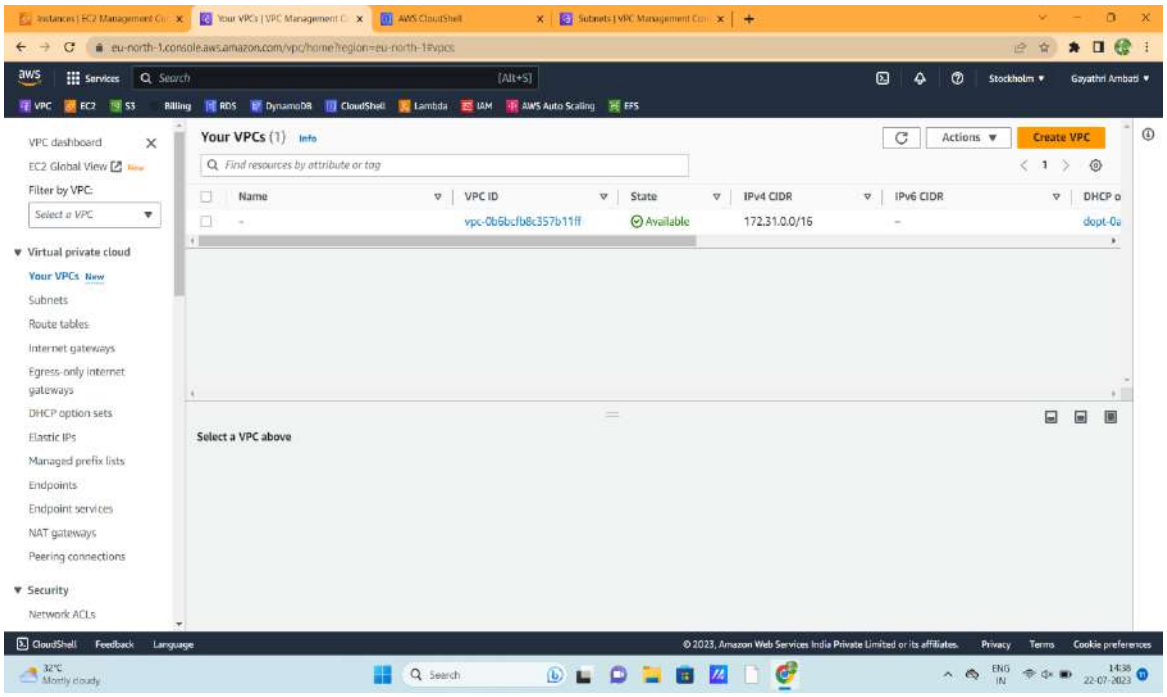
• Click on the "Create VPC" option.

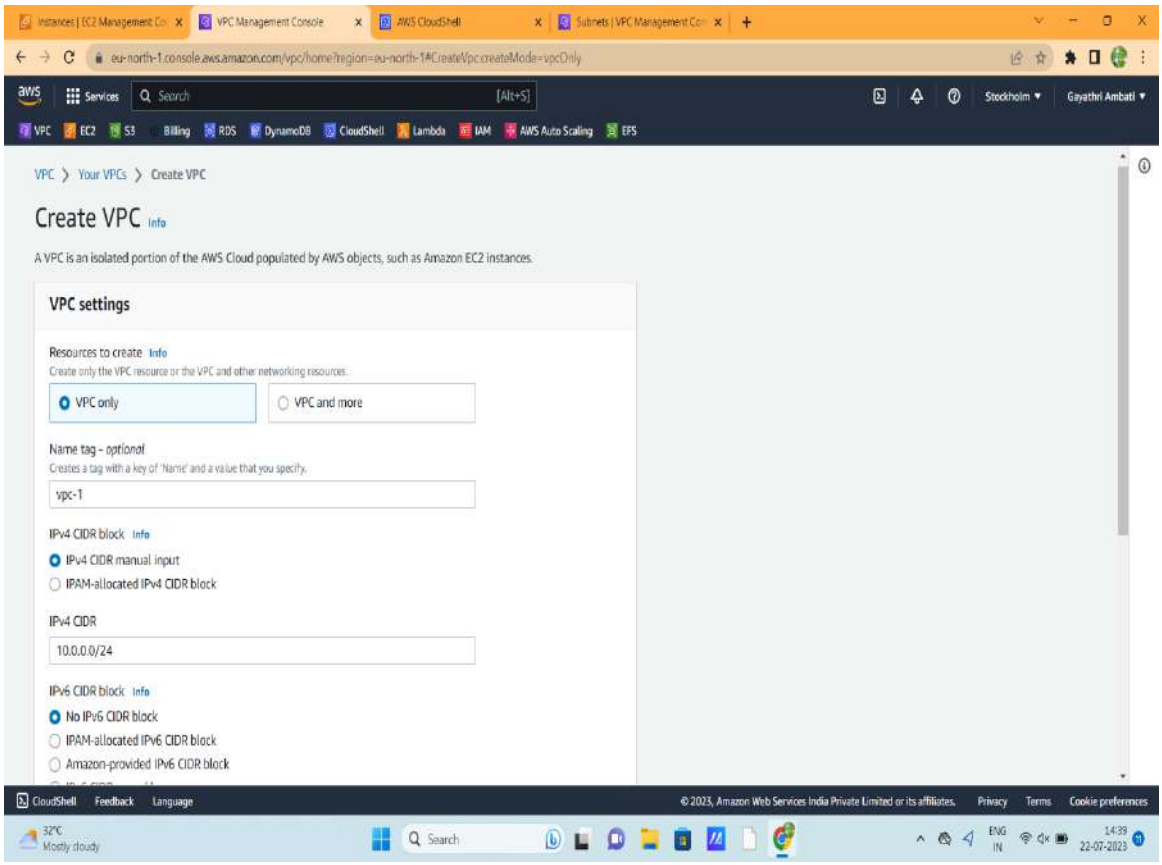
• Give the VPC name as vpc1 and CIDR value as 10.0.0.0/24.

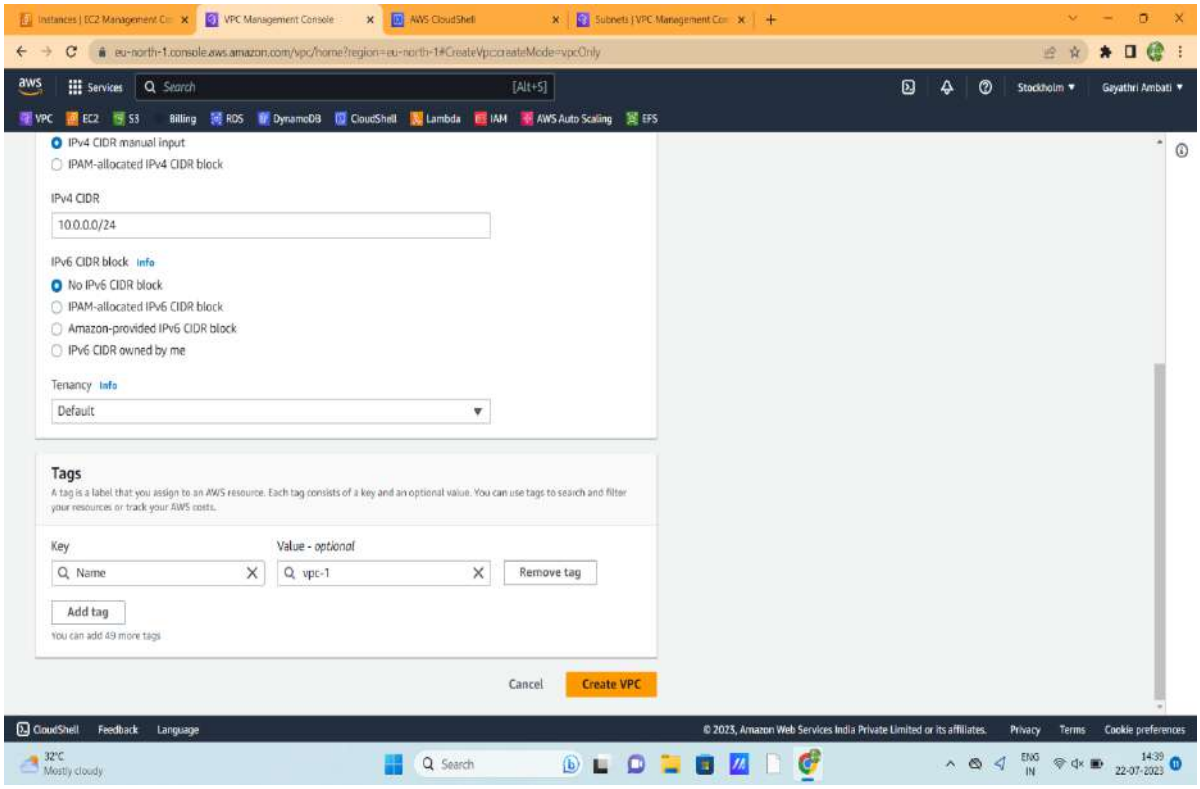
• Click on create VPC.

•And create vpc-1.

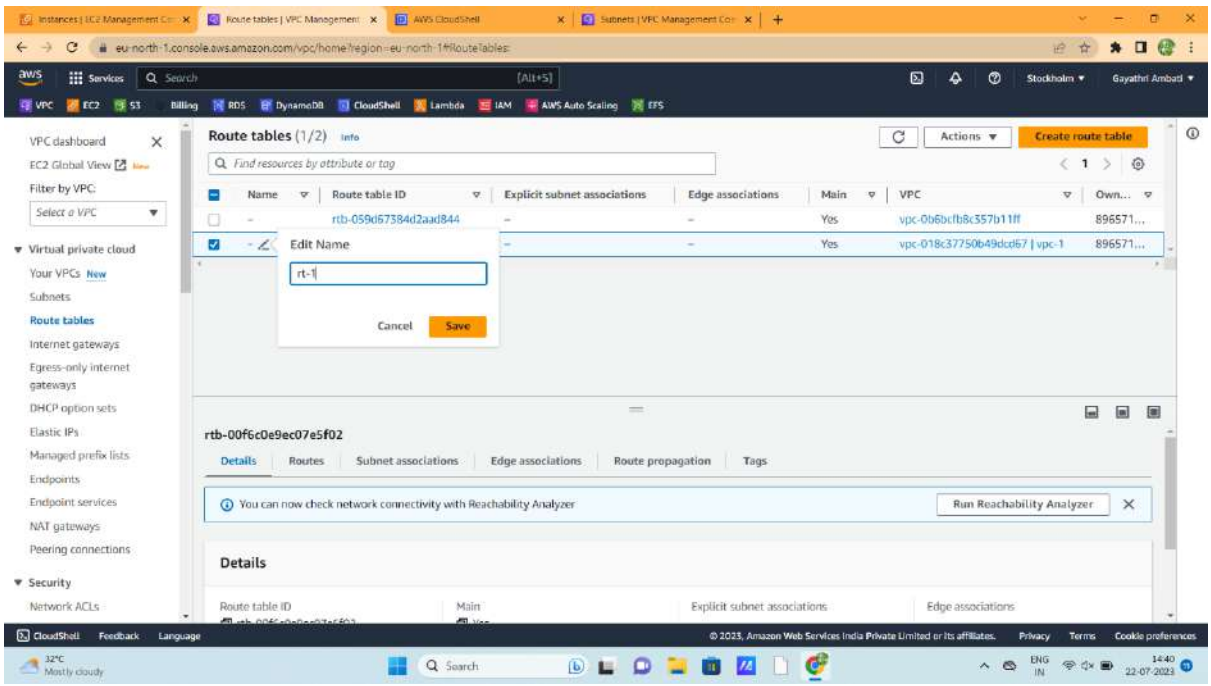



Your VPCs (1) Info
Actions
Create VPC
Name
VPC ID
State
IPv4 CIDR
IPv6 CIDR
vpc-0b6bcfb8c357b11ff
Available
172.31.0.0/16
Select a VPC above


VPC > Your VPCs > Create VPC
Create VPC Info
A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances.
VPC settings
Resources to create Info
VPC only
VPC and more
Name tag - optional
vpc-1
IPv4 CIDR block Info
IPv4 CIDR manual input
IPAM-allocated IPv4 CIDR block
IPv4 CIDR
10.0.0.0/24
IPv6 CIDR block Info
No IPv6 CIDR block
IPAM-allocated IPv6 CIDR block
Amazon-provided IPv6 CIDR block

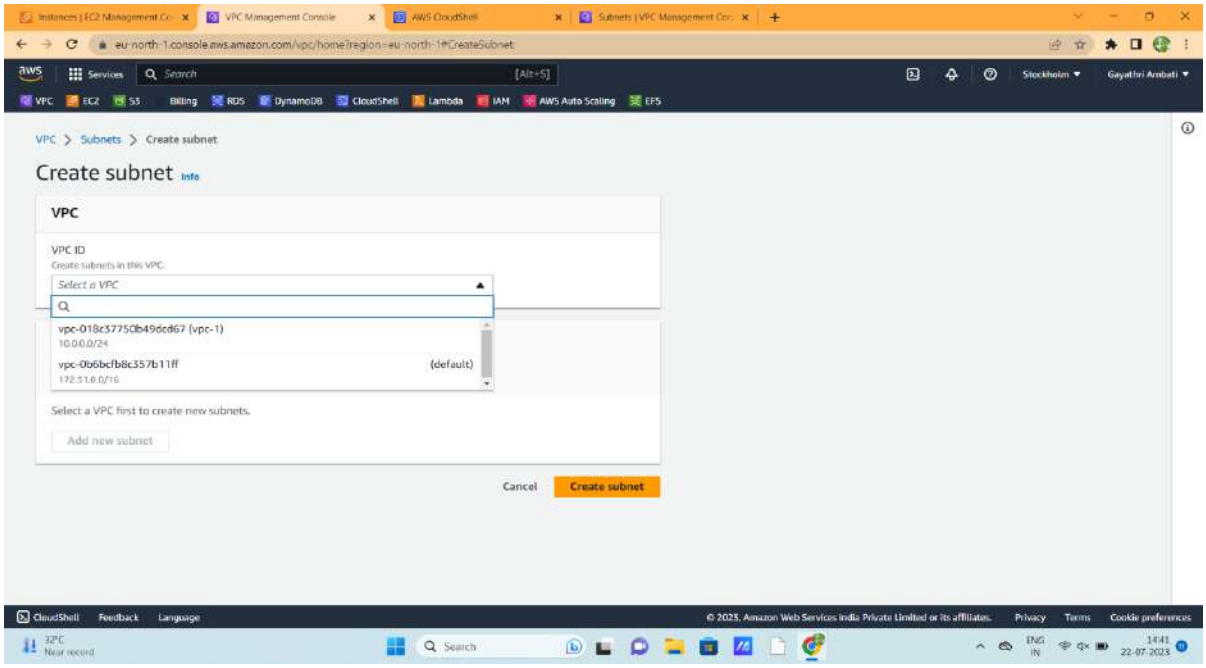


• Go to route table and edit the name of route table as rt1 public that is created along with vpc1.

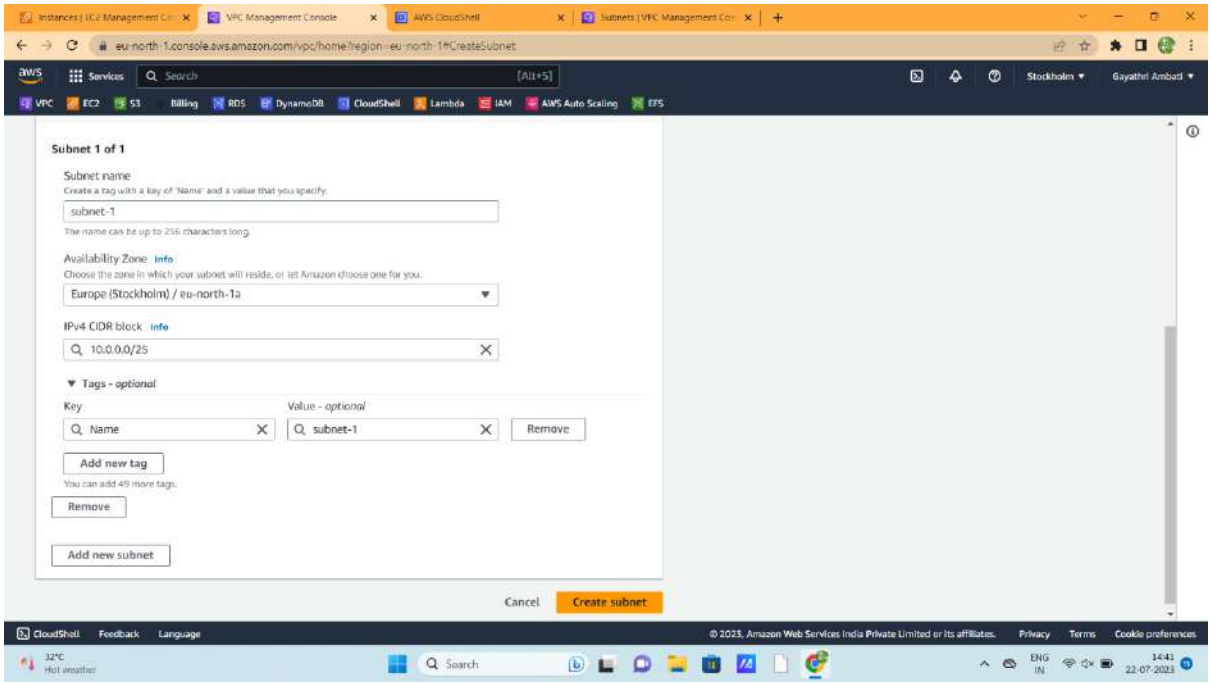


• Go to subnets and create a subnet and name it as public subnet. Select the availability zone as North-1a, CIDR as 10.0.0.0/25 and select create subnet.

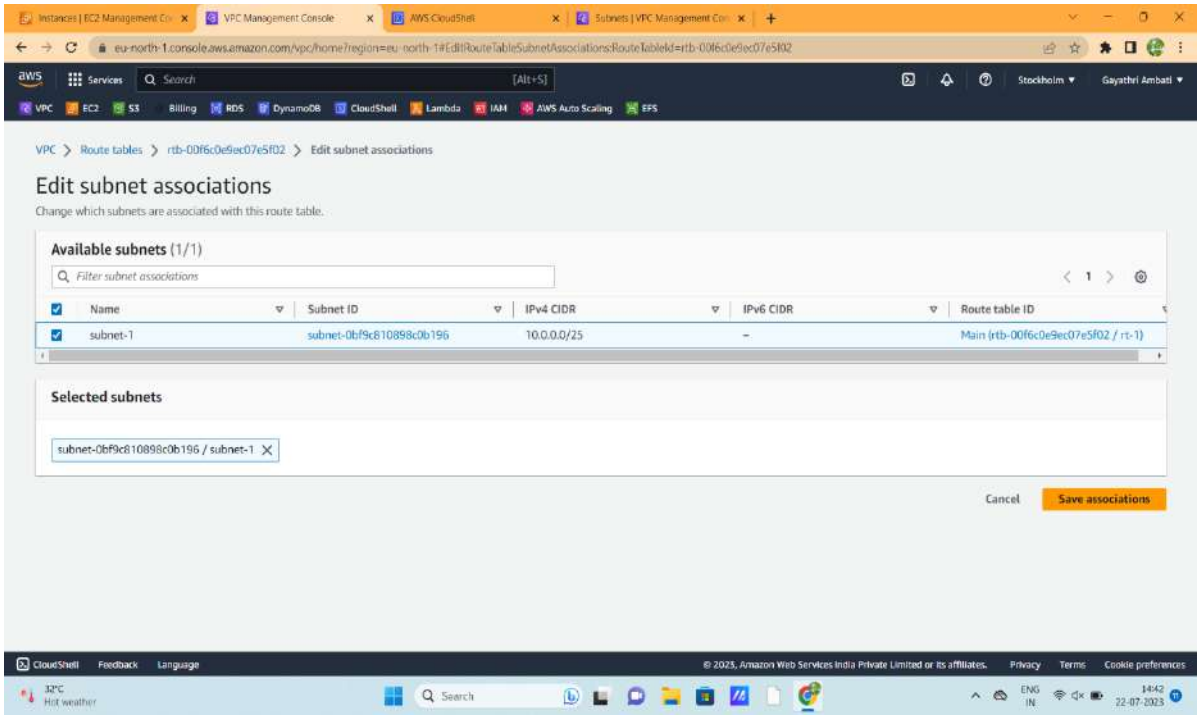
• we initially have 3 default Subnets.



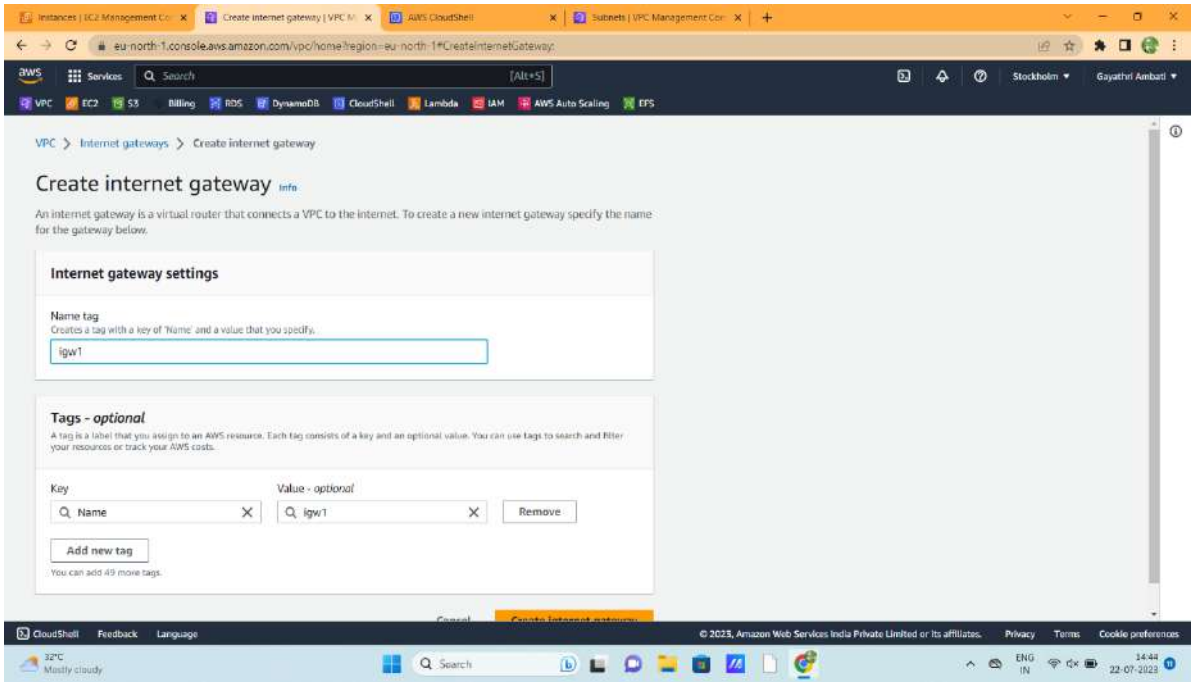
• select VPC ID as vpc-1.



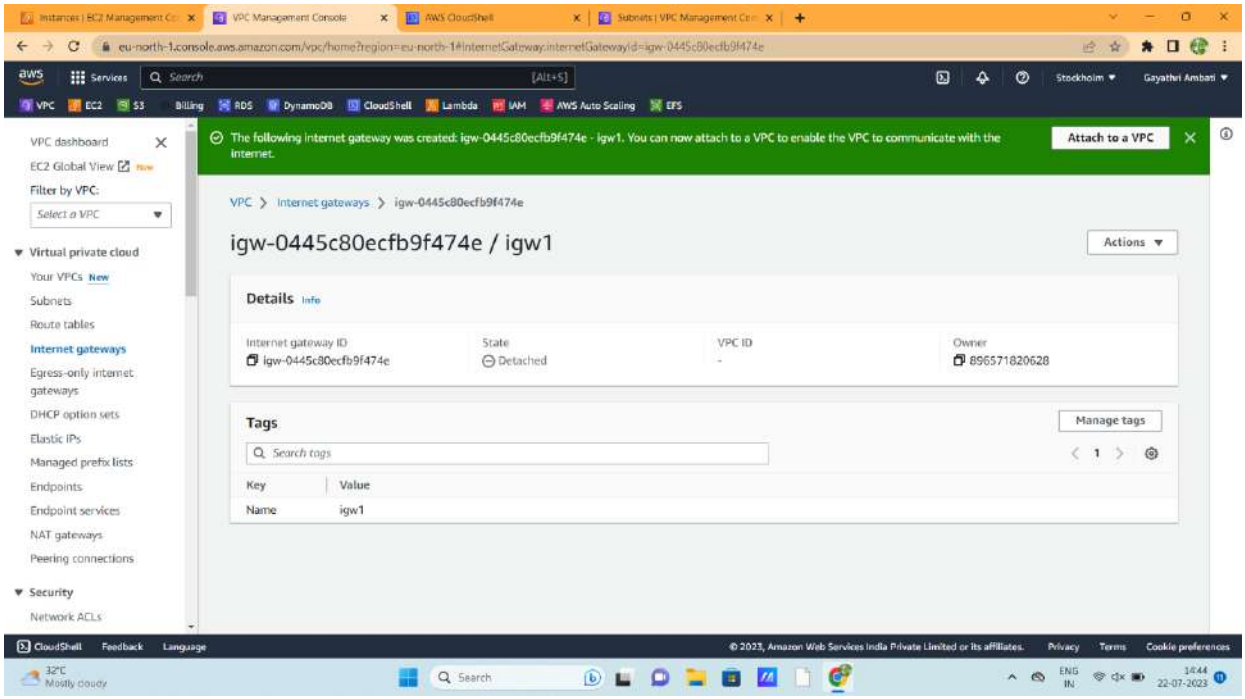
• Go to route tables, select the public route table, and go to subnet associations and select public subnet and click on save associations.



• Create an internet gateway with the name igw1.

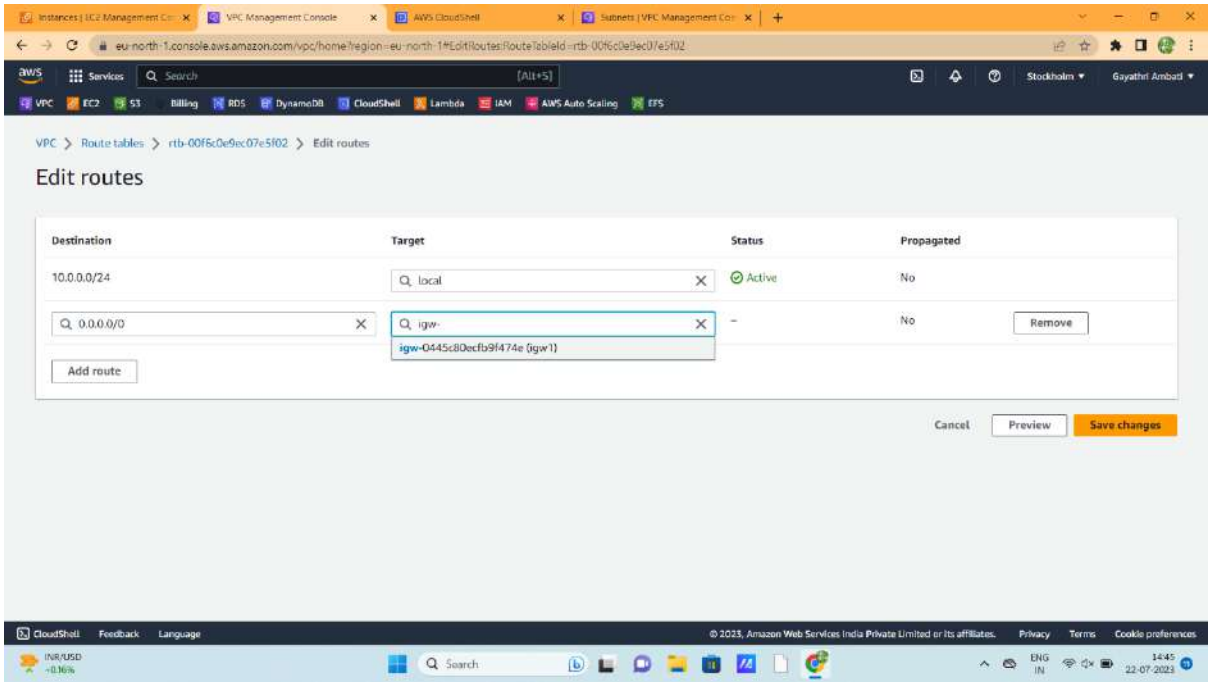


• Attach custom IGW to custom VPC.



• Internet Gate Way is successfully attached to VPC.

• Now go to route tables, select rt1-vpc1-public, click on edit routes, go to add routes, and add 0.0.0.0/0 at destination and select our internet gateway in destination and save it.



• Similarly create another two VPCs (VPC-2, VPC-3) using the above steps.

• Give the VPC name as vpc2 and CIDR value as 192.168.0.0/24.

• Click on create VPC and create vpc-2.

• create subnet to vpc-2 with CIDR- 192.168.0.0/25.

• Go to route tables, select the public route table, and go to subnet associations and select public subnet and click on save associations.

• Create an internet gateway with the name igw2.

• Attach custom IGW to custom VPC.

• Internet Gate Way is successfully attached to VPC.

• Now go to route tables, select rt2 public, click on edit routes, go to add routes, and add 0.0.0.0/0 at destination and select our internet gateway in destination and save it.

• Give the VPC name as vpc3 and CIDR value as 4.0.0.0/24.

• Click on create VPC and create vpc-3.

• create subnet to vpc-2 with CIDR- 4.0.0.0/25.

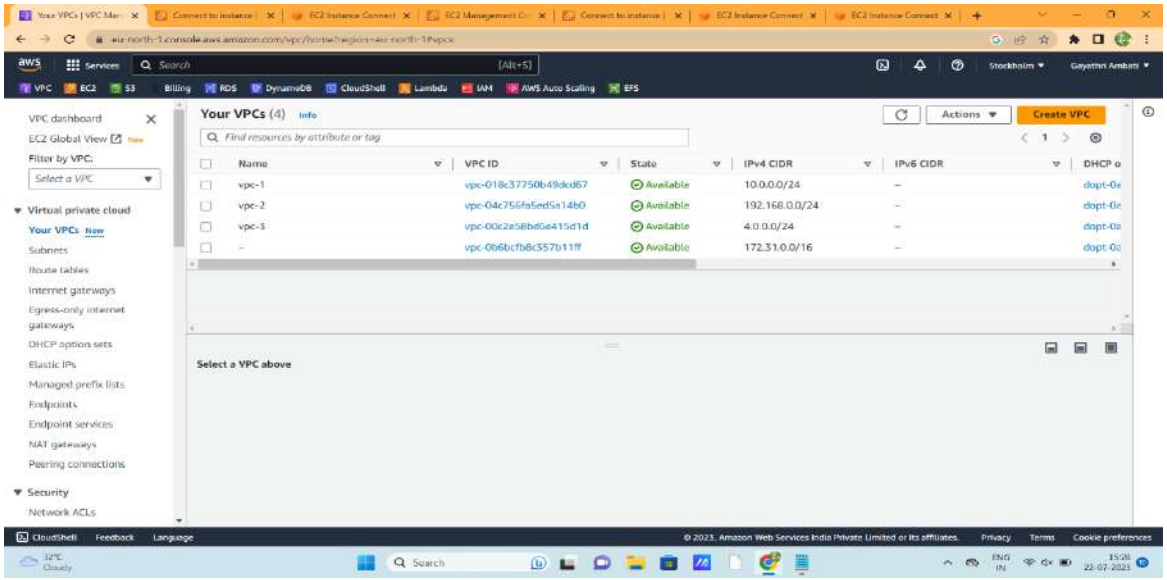
• Go to route tables, select the public route table, and go to subnet associations and select public subnet and click on save associations.

• Create an internet gateway with the name igw3.

• Attach custom IGW to custom VPC.

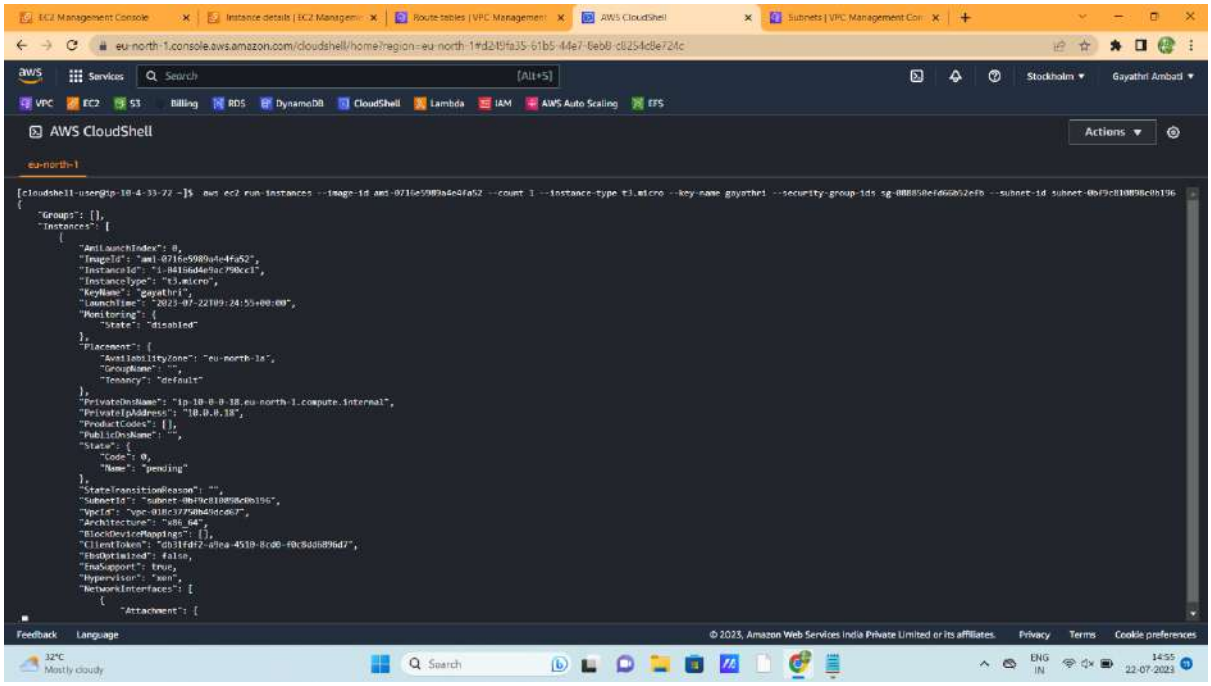
• Internet Gate Way is successfully attached to VPC.

• Now go to route tables, select rt2 public, click on edit routes, go to add routes, and add 0.0.0.0/0 at destination and select our internet gateway in destination and save it.

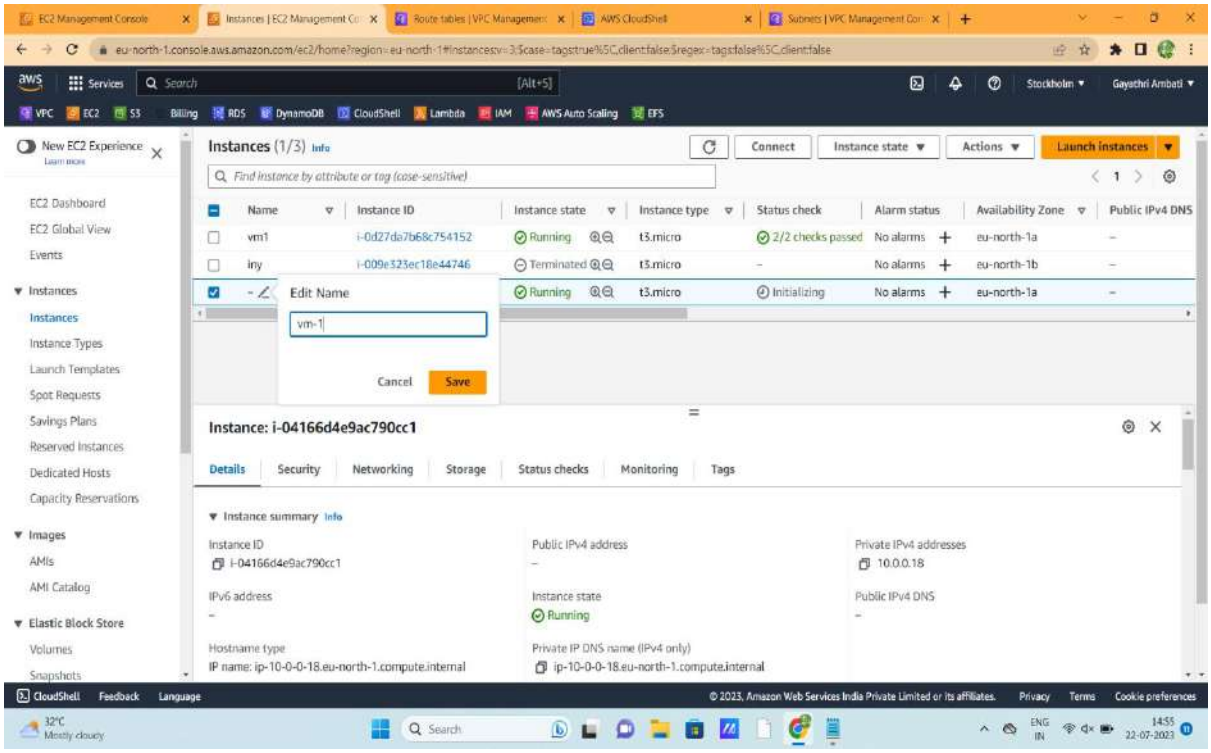


• Now create an instance using Cloud Shell and remaining two instances in normal process.

• To create an instance using cloud shell we have some commands.

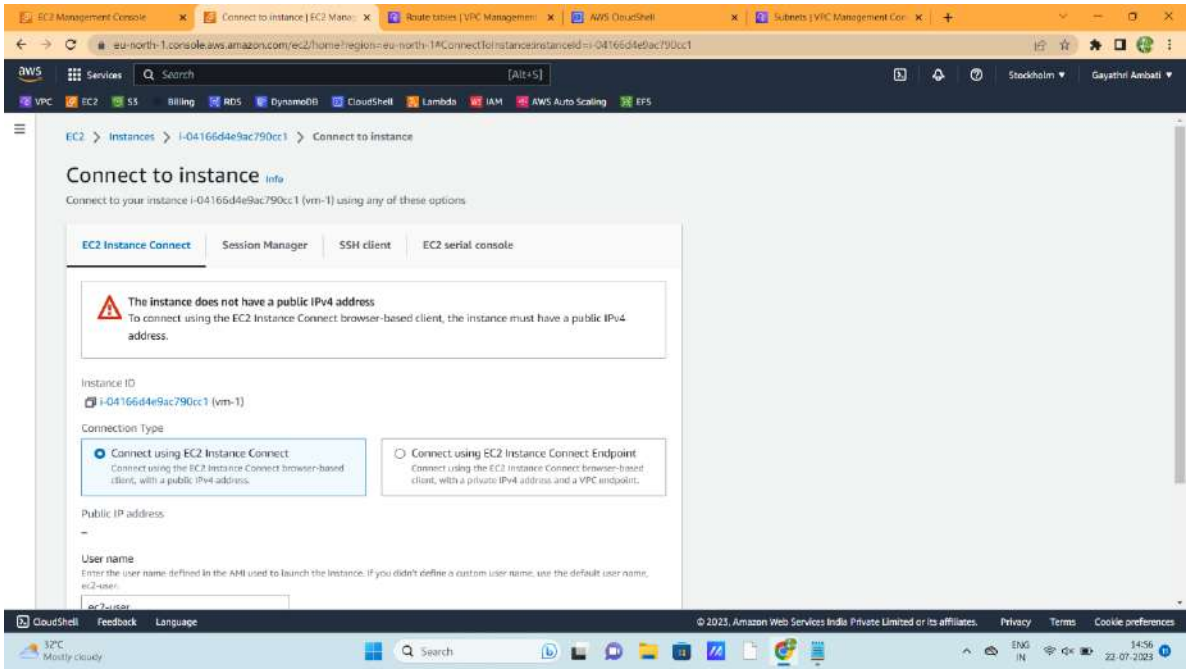


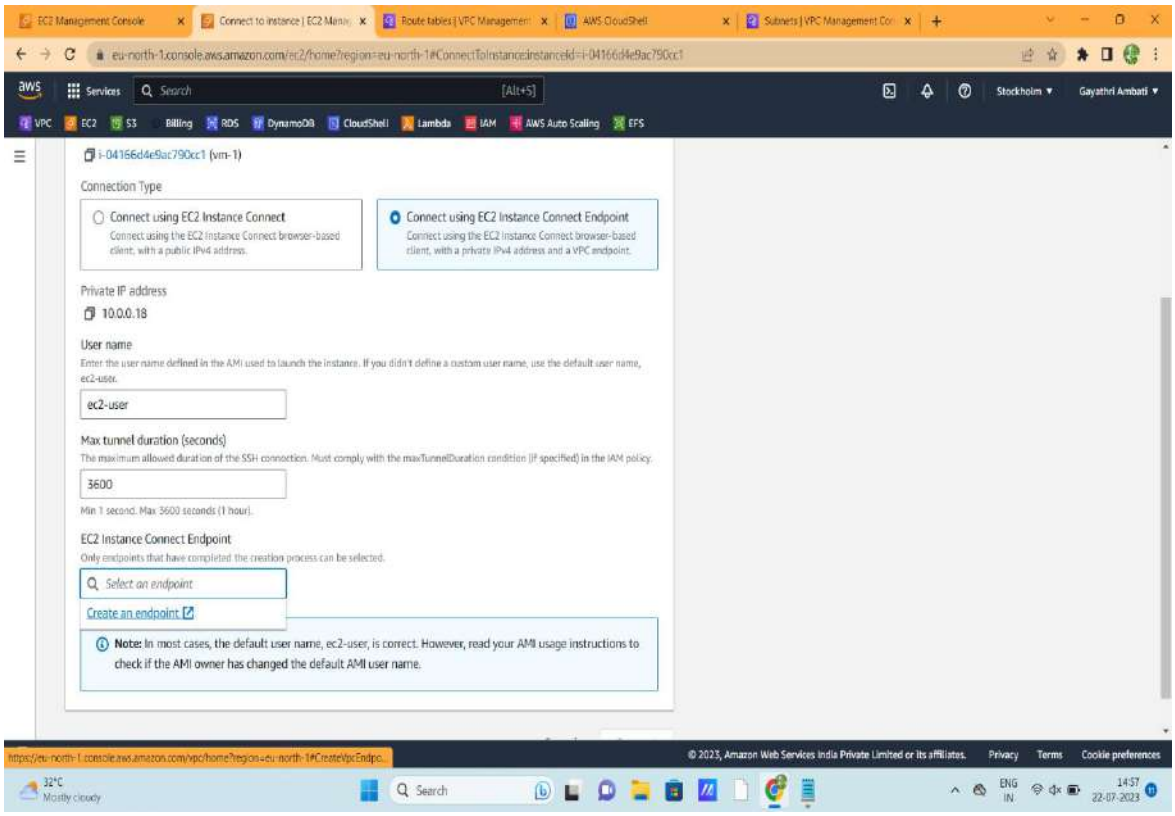
• Now the Instance is created.

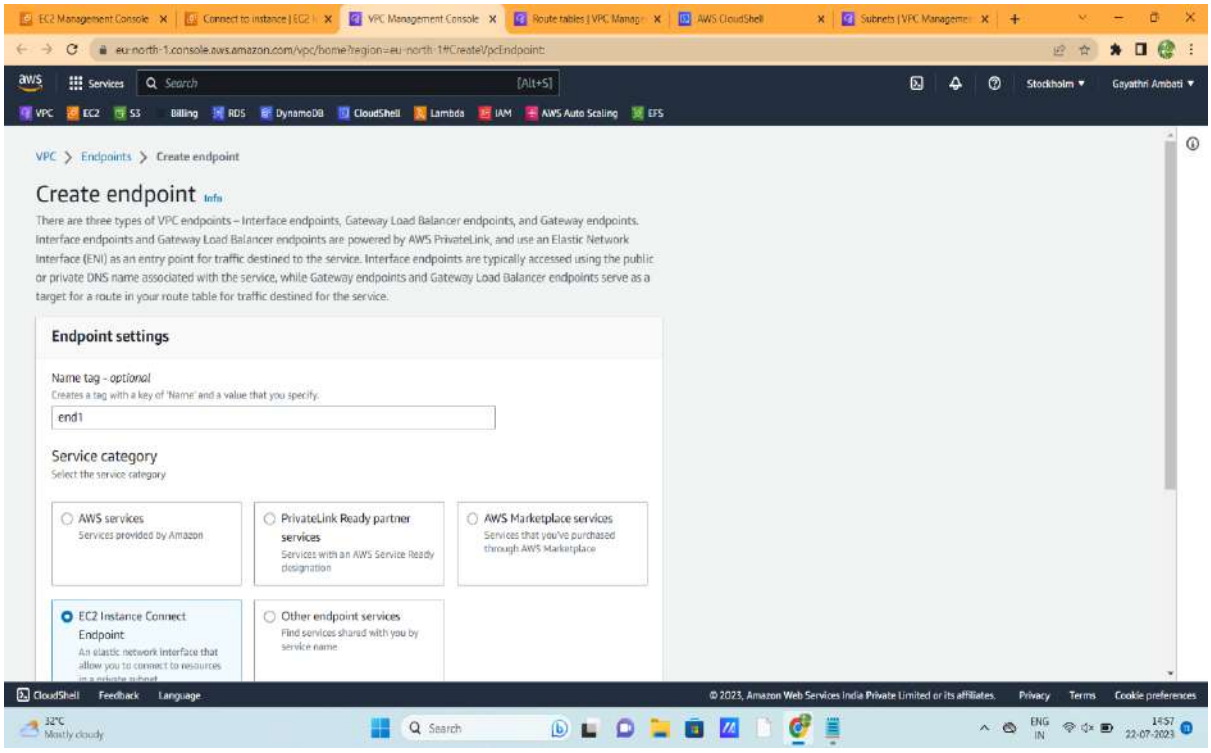


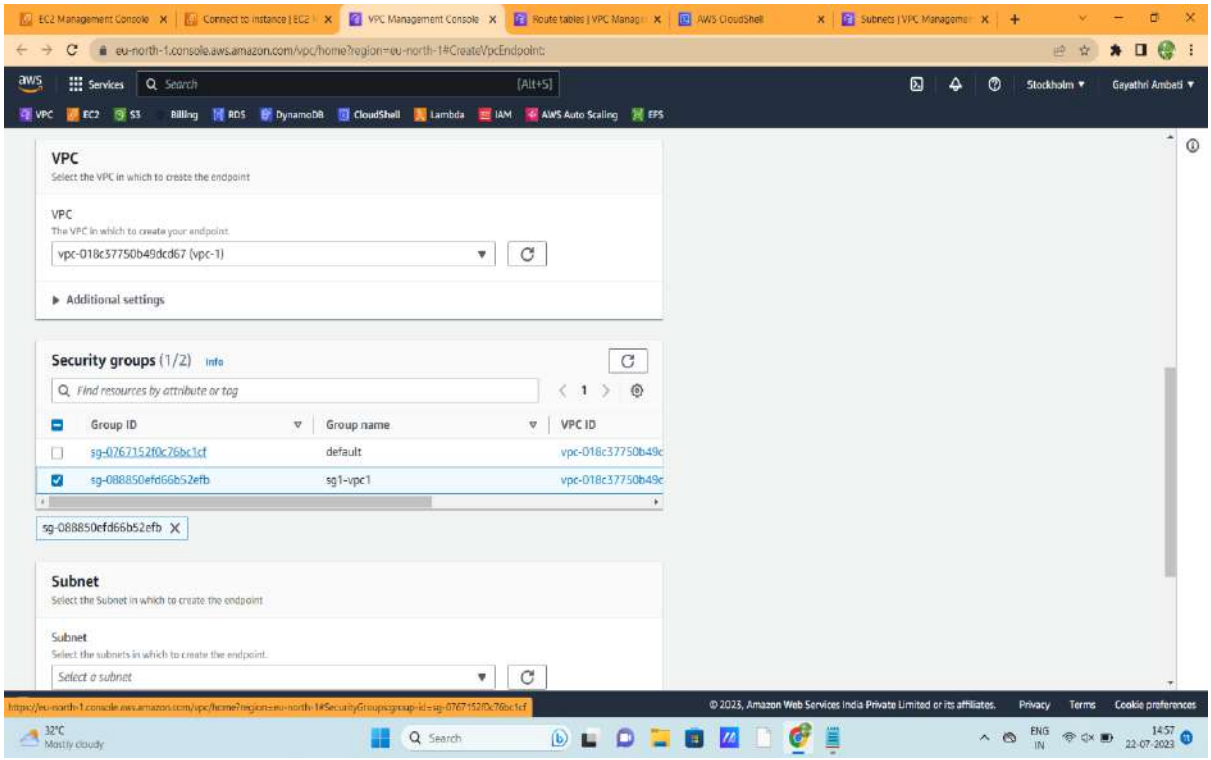
• Try to connect the Instance Vm-1. It will show error.

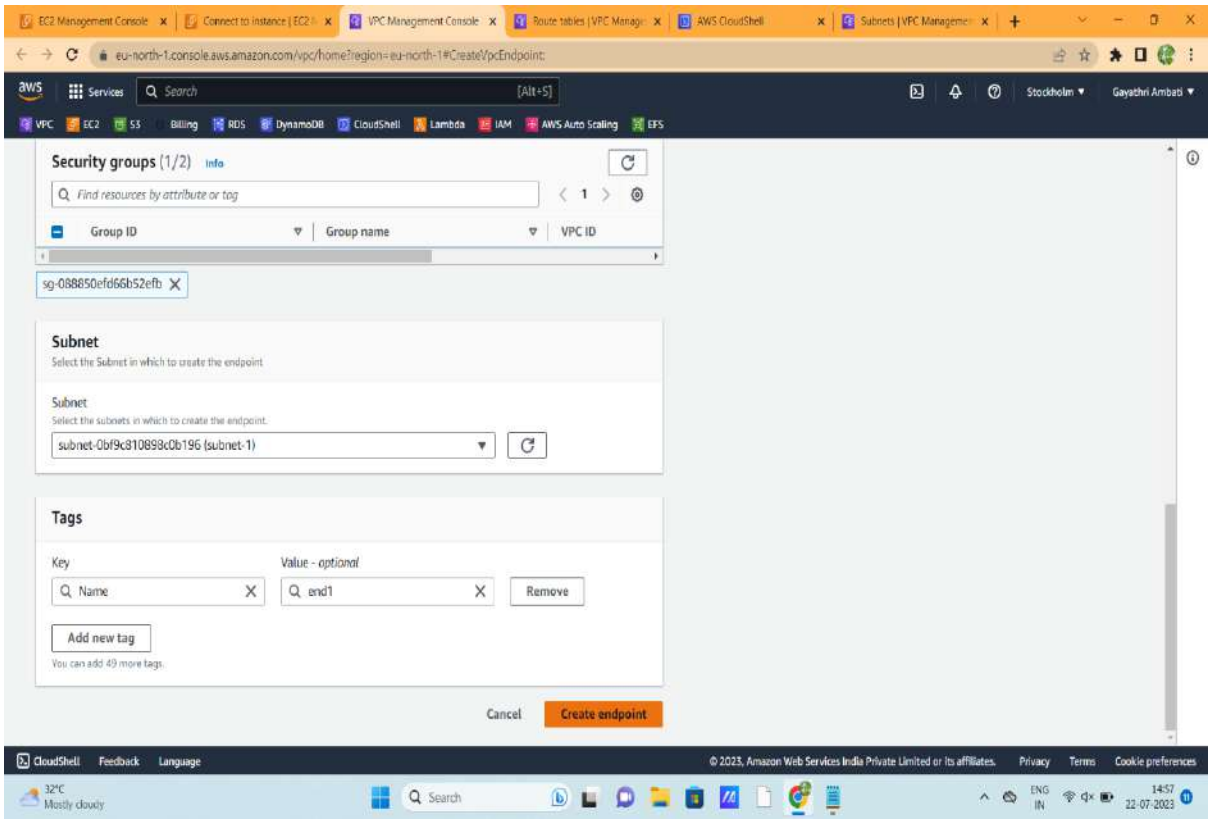
• create an end point to the Instance and then try to connect it.

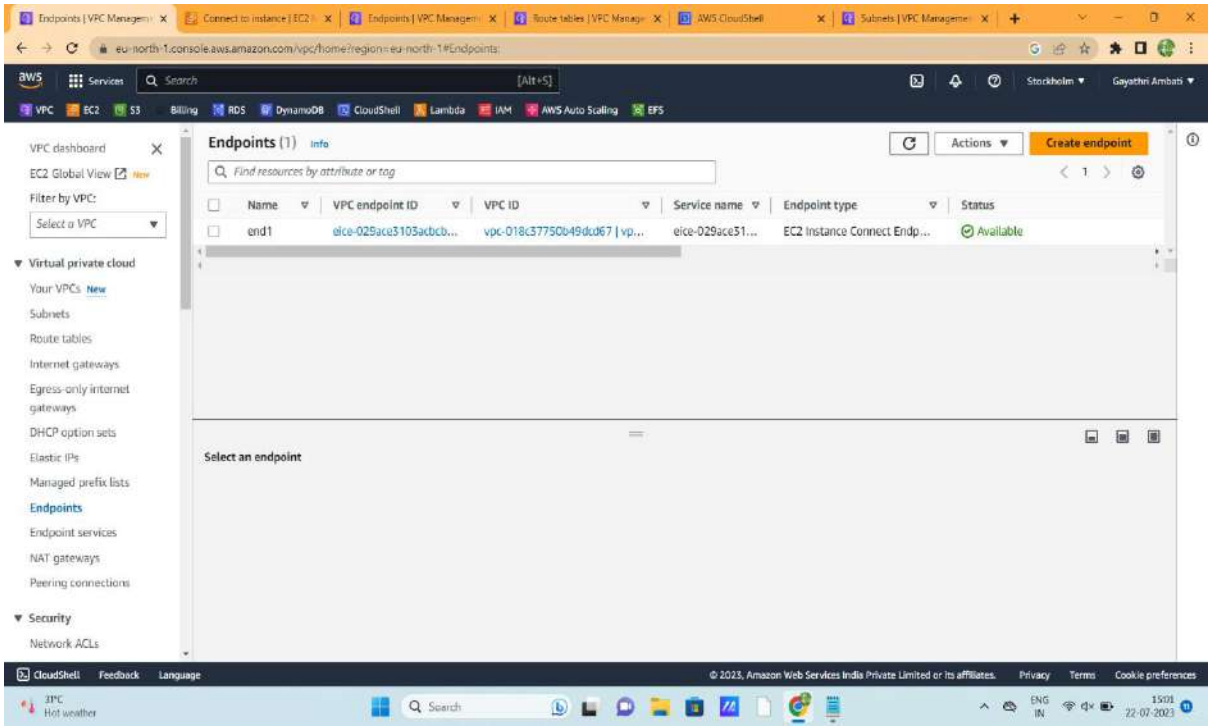



Connection Type
Connect using EC2 Instance Connect
Connect using the EC2 Instance Connect browser-based client, with a public IPv4 address.
Connect using EC2 Instance Connect Endpoint
Connect using the EC2 Instance Connect browser-based client, with a private IPv4 address and a VPC endpoint.
Private IP address
10.0.0.18
User name
ec2-user
Max tunnel duration (seconds)
3600
EC2 Instance Connect Endpoint
Select an endpoint
Create an endpoint
Note: In most cases, the default user name, ec2-user, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

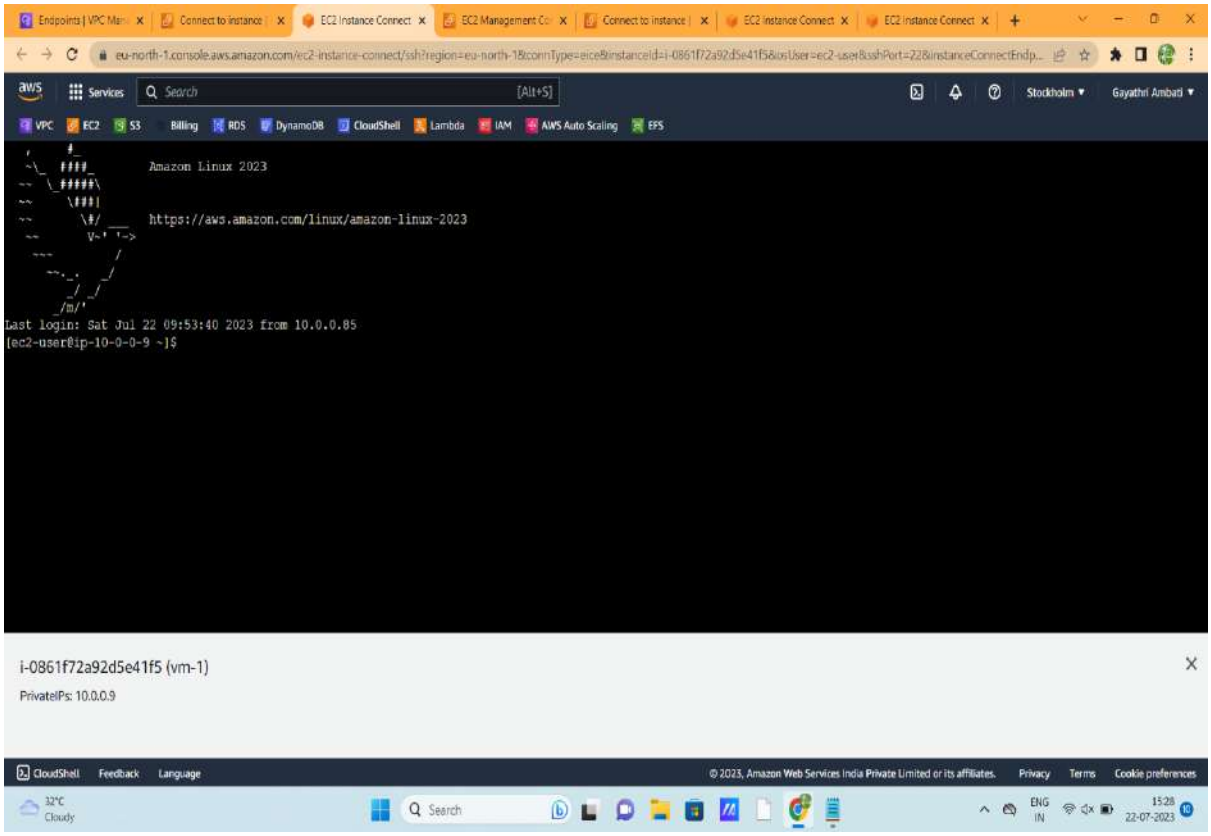

VPC > Endpoints > Create endpoint
Create endpoint
There are three types of VPC endpoints – Interface endpoints, Gateway Load Balancer endpoints, and Gateway endpoints. Interface endpoints and Gateway Load Balancer endpoints are powered by AWS PrivateLink, and use an Elastic Network Interface (ENI) as an entry point for traffic destined to the service. Interface endpoints are typically accessed using the public or private DNS name associated with the service, while Gateway endpoints and Gateway Load Balancer endpoints serve as a target for a route in your route table for traffic destined for the service.
Endpoint settings
Name tag - optional
end1
Service category
AWS services
PrivateLink Ready partner services
AWS Marketplace services
EC2 Instance Connect Endpoint
Other endpoint services


VPC
Select the VPC in which to create the endpoint
VPC
The VPC in which to create your endpoint.
vpc-018c37750b49dcd67 (vpc-1)
Additional settings
Security groups (1/2) Info
Find resources by attribute or tag
Group ID
Group name
VPC ID
sg-0767152f0c76bc1cf
default
sg-088850efd66b52efb
sg1-vpc1
sg-088850efd66b52efb
Subnet
Select the Subnet in which to create the endpoint
Subnet
Select the subnets in which to create the endpoint.
Select a subnet


Security groups (1/2) Info
Find resources by attribute or tag
Group ID
Group name
VPC ID
sg-088850efd66b52efb
Subnet
Select the Subnet in which to create the endpoint
Subnet
Select the subnets in which to create the endpoint.
subnet-0bf9c810898c0b196 (subnet-1)
Tags
Key
Value - optional
Name
end1
Remove
Add new tag
You can add 49 more tags.
Cancel
Create endpoint
CloudShell
Feedback
Language



- Now select the VM-1 and connect it will be connected.



•Now create another two Instances using EC2.

•Creating an EC2 instance in a public subnet:

• Name your instance as VM-2.

• Select “Amazon Linux 2 AMI”.

• Instance type “t3. micro”.

• Select your existing key pair.

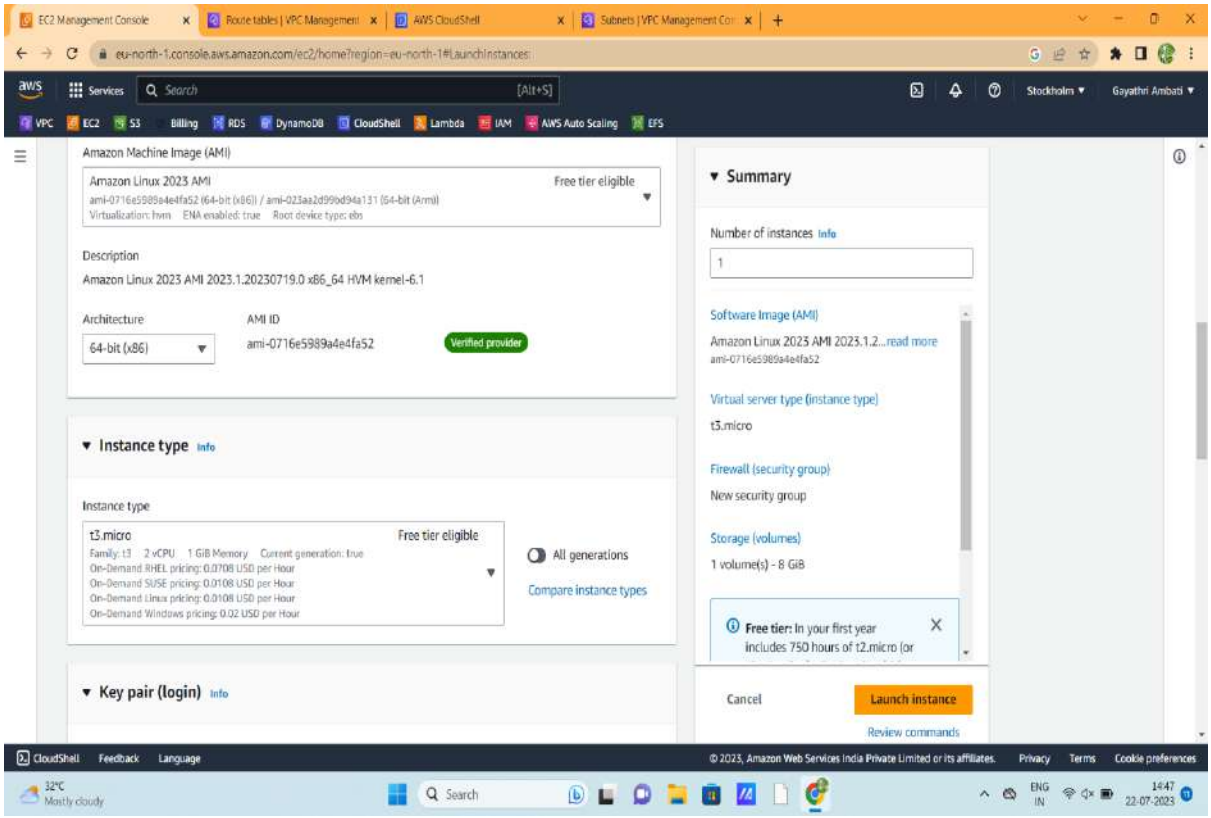
• Select your custom VPC, public subnet and enable the auto-assign public IP.

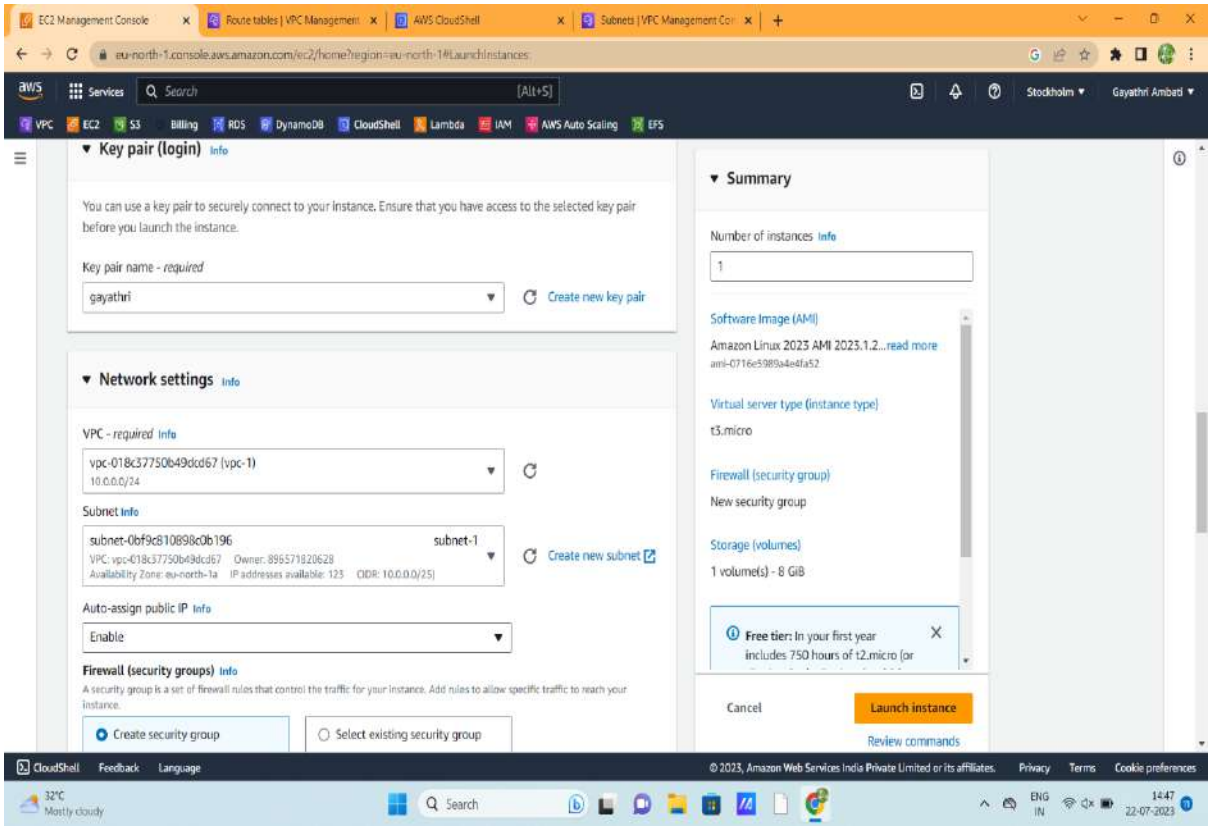
• In the security group section, select availability zone as North-1a.

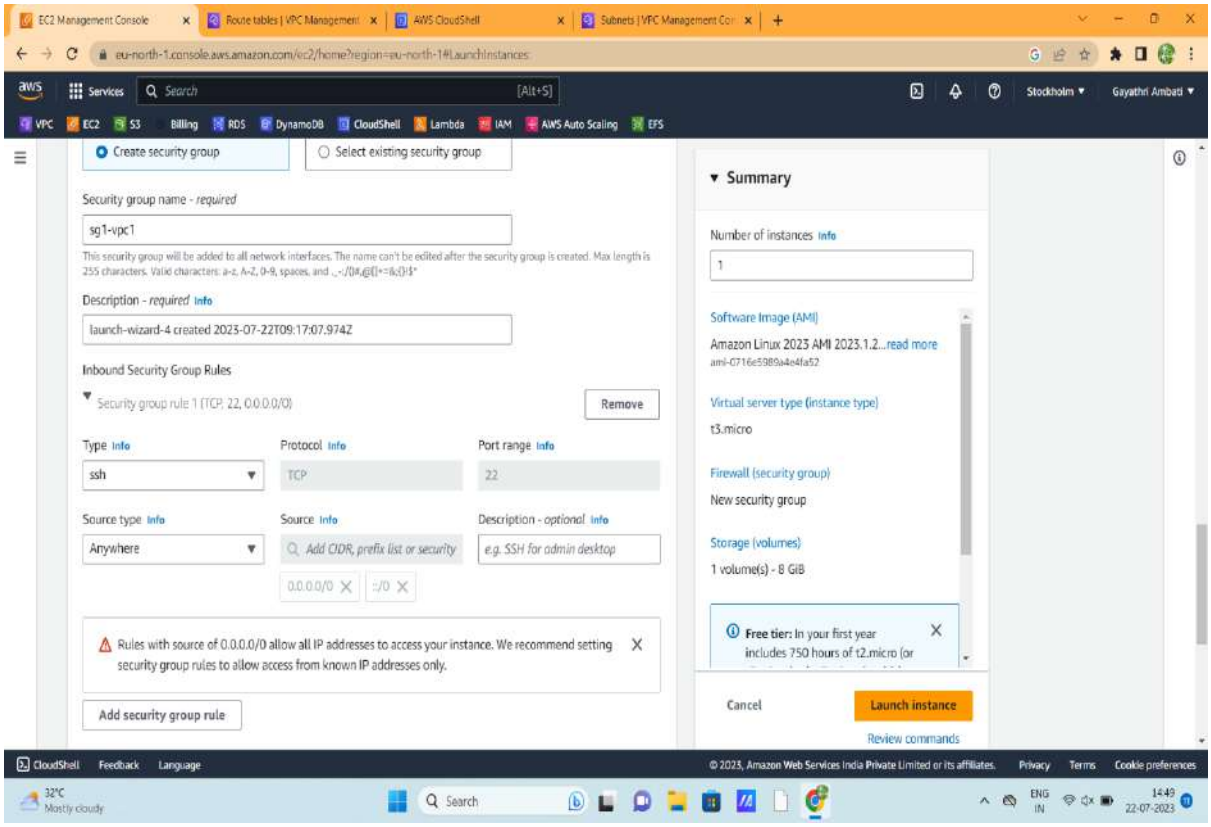
• create a new security group, add security groups that supports SSH and All Traffic.

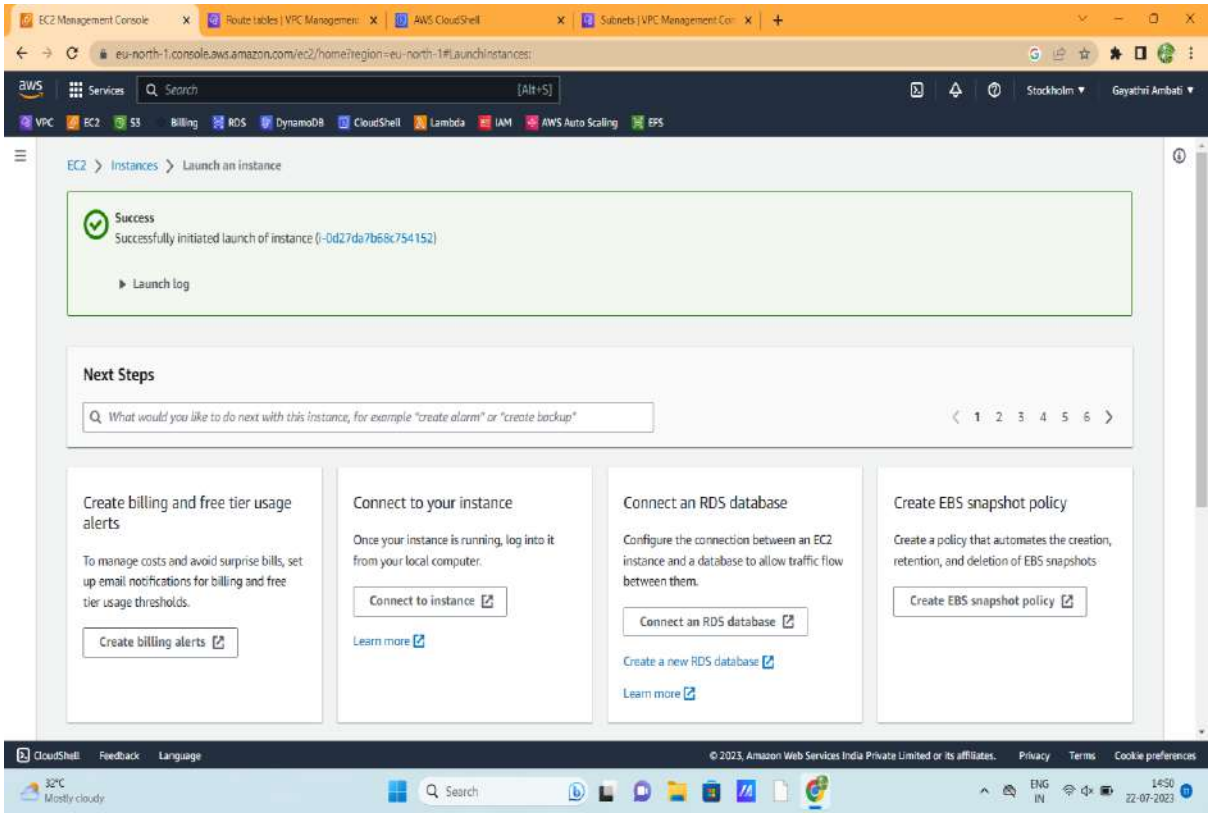
• Similarly create the third Instance naming it as VM-3.

• Launch your Instance.

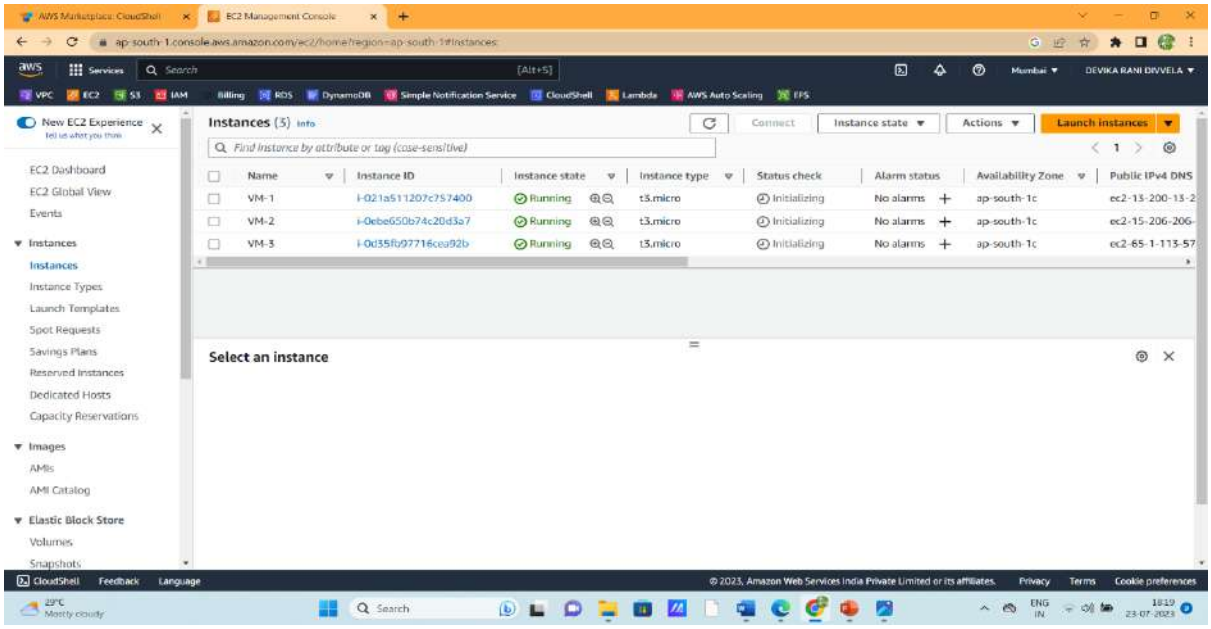



EC2 Management Console
Route tables | VPC Management
AWS CloudShell
Subnets | VPC Management
eu-north-1.console.aws.amazon.com/ec2/home?region=eu-north-1#LaunchInstances:
Services
Search
[Alt+S]
Stockholm
Gayathri Ambati
VPC
EC2
S3
Billing
RDS
DynamoDB
CloudShell
Lambda
IAM
AWS Auto Scaling
EFS
Key pair (login) Info
You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.
Key pair name - required
gayathri
Create new key pair
Network settings Info
VPC - required Info
vpc-018c37750b49dcd67 (vpc-1)
10.0.0.0/24
Subnet Info
subnet-0bf9c810898c0b196
subnet-1
VPC: vpc-018c37750b49dcd67 Owner: 895571820628
Availability Zone: eu-north-1a IP addresses available: 123 CIDR: 10.0.0.0/25)
Create new subnet
Auto-assign public IP Info
Enable
Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.
Create security group
Select existing security group
Summary
Number of instances Info
1
Software Image (AMI)
Amazon Linux 2023 AMI 2023.1.2...read more
ami-0716e5989a4e4fa52
Virtual server type (instance type)
t3.micro
Firewall (security group)
New security group
Storage (volumes)
1 volume(s) - 8 GiB
Free tier: In your first year includes 750 hours of t2.micro (or
Cancel
Launch instance
Review commands
CloudShell
Feedback
Language
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences
32°C
Mostly cloudy
Search
ENG
IN
14:47
22-07-2023


EC2 Management Console
Route tables | VPC Management
AWS CloudShell
Subnets | VPC Management
eu-north-1.console.aws.amazon.com/ec2/home?region=eu-north-1#LaunchInstances:
Services
Search
[Alt+S]
Stockholm
Gayathri Ambati
VPC
EC2
S3
Billing
RDS
DynamoDB
CloudShell
Lambda
IAM
AWS Auto Scaling
EFS
Create security group
Select existing security group
Security group name - required
sg1-vpc1
This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and ._-:/()#,@[]+=&;{}!$*
Description - required Info
launch-wizard-4 created 2023-07-22T09:17:07.974Z
Inbound Security Group Rules
Security group rule 1 (TCP, 22, 0.0.0.0/0)
Remove
Type Info
ssh
Protocol Info
TCP
Port range Info
22
Source type Info
Anywhere
Source Info
Add CIDR, prefix list or security
Description - optional Info
e.g. SSH for admin desktop
0.0.0.0/0
::/0
Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.
Add security group rule
Summary
Number of instances Info
1
Software Image (AMI)
Amazon Linux 2023 AMI 2023.1.2...read more
ami-0716e5989a4e4fa52
Virtual server type (instance type)
t3.micro
Firewall (security group)
New security group
Storage (volumes)
1 volume(s) - 8 GiB
Free tier: In your first year includes 750 hours of t2.micro (or
Cancel
Launch instance
Review commands
CloudShell
Feedback
Language
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences
32°C
Mostly cloudy
Search
ENG
IN
14:49
22-07-2023



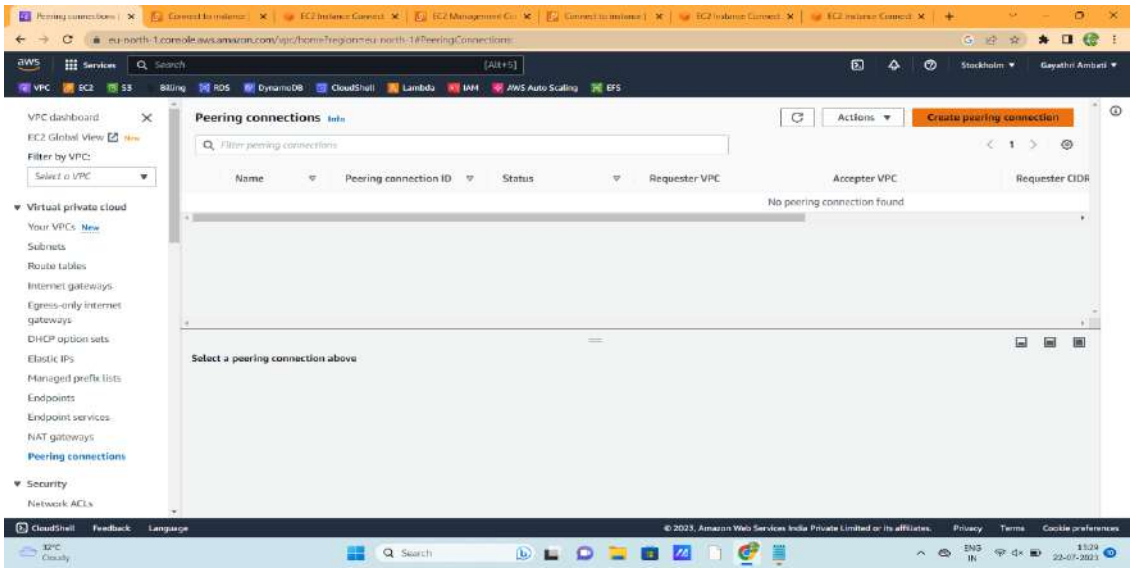
• Finally the Instance is launched.



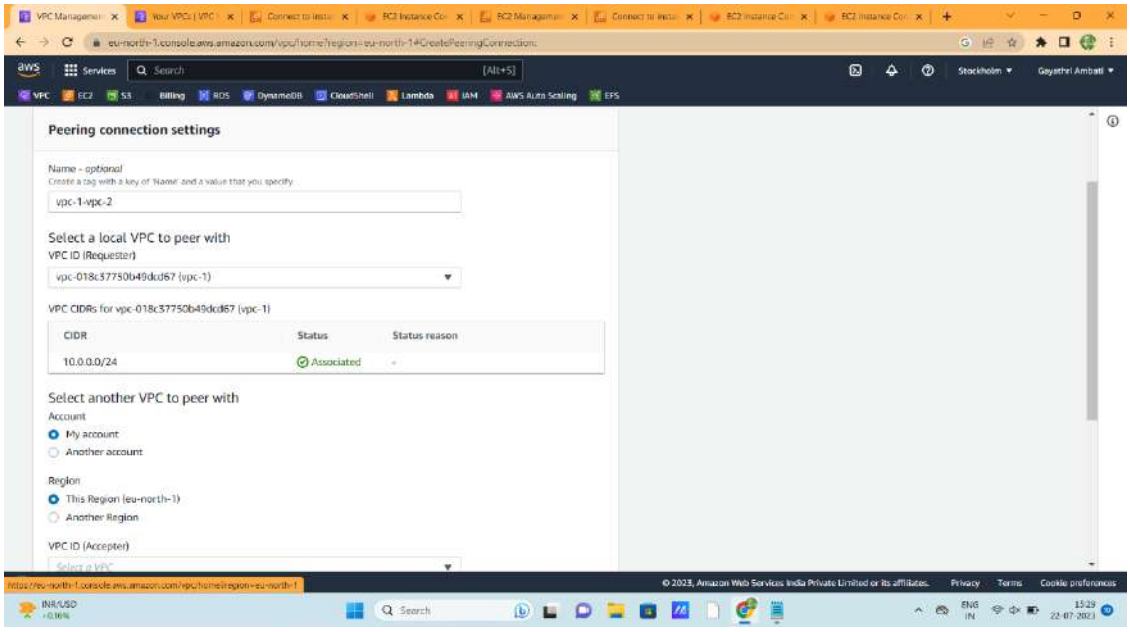
Peering connection

- Now we have to create peering connection between these 3 VPCs
- Go to peering connections in VPC Service

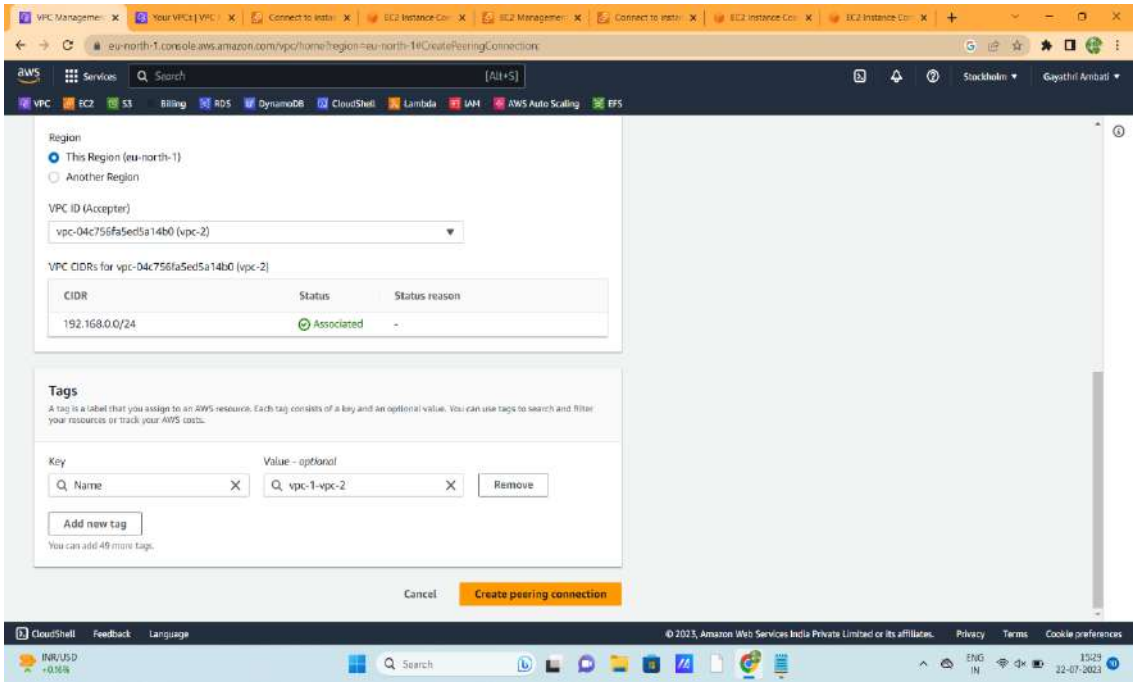
- Click on create peering connection



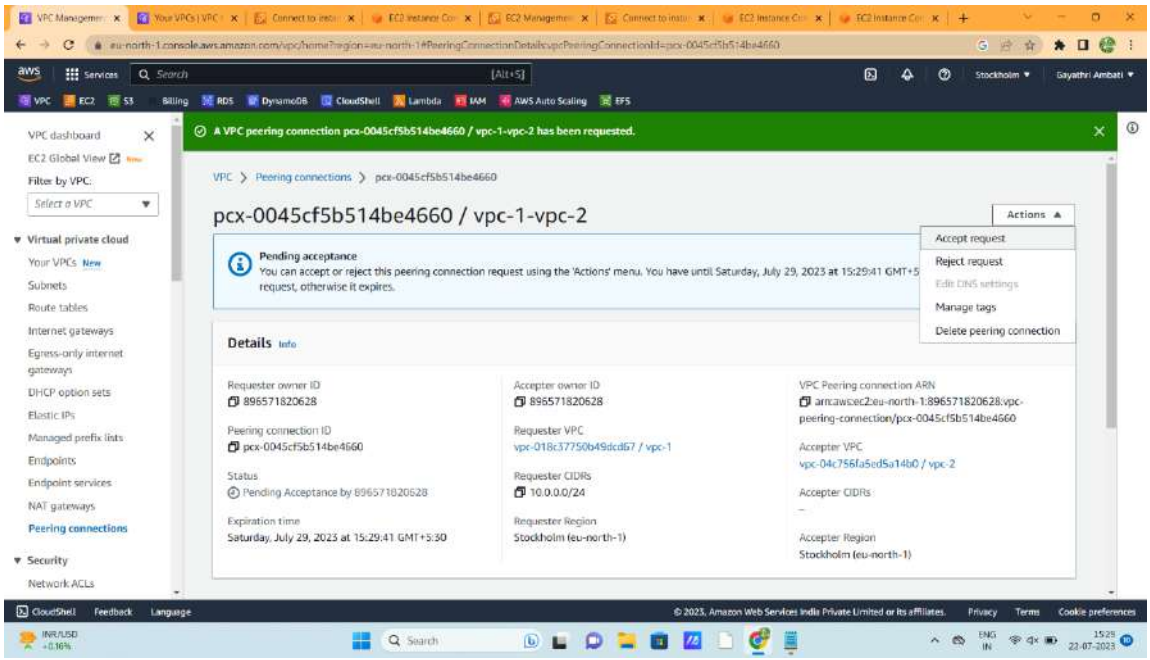
- For creating peering connection between VPC1 and VPC2. Give name as vpc-1-vpc-2 and select VPC(requester) as VPC1
- As these VPCs belongs to same region and same account so, select This region and My account



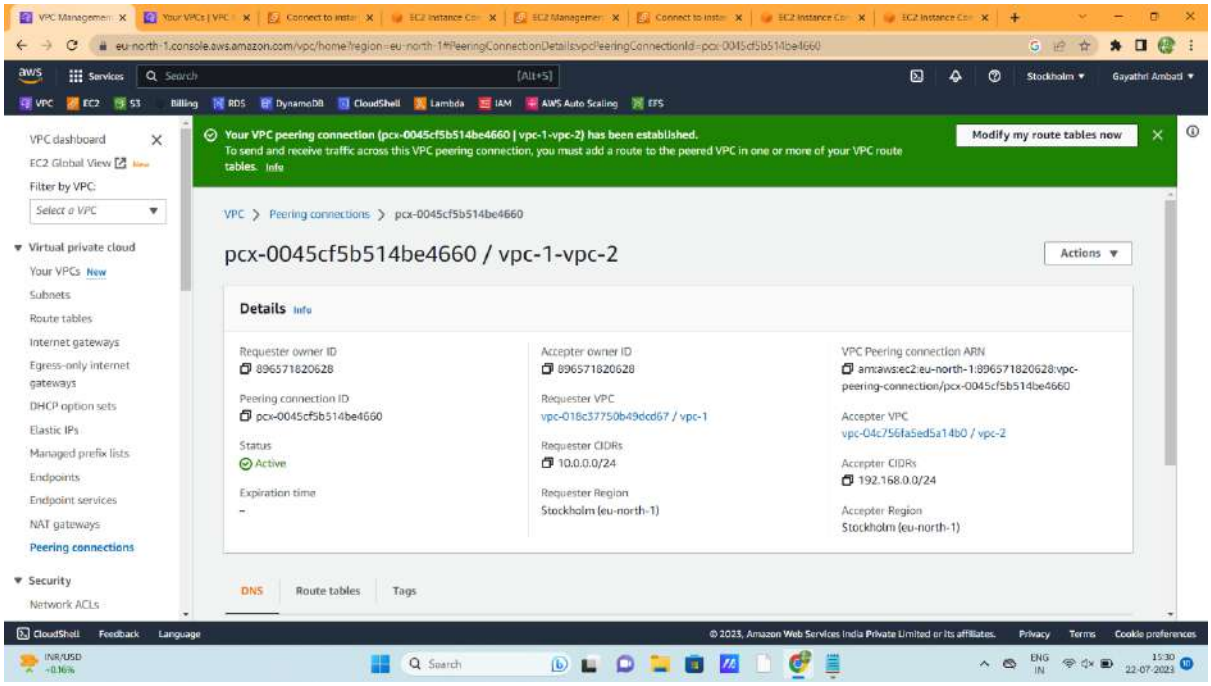
- Select VPC (acceptor) as VPC2 and click on create peering connection.



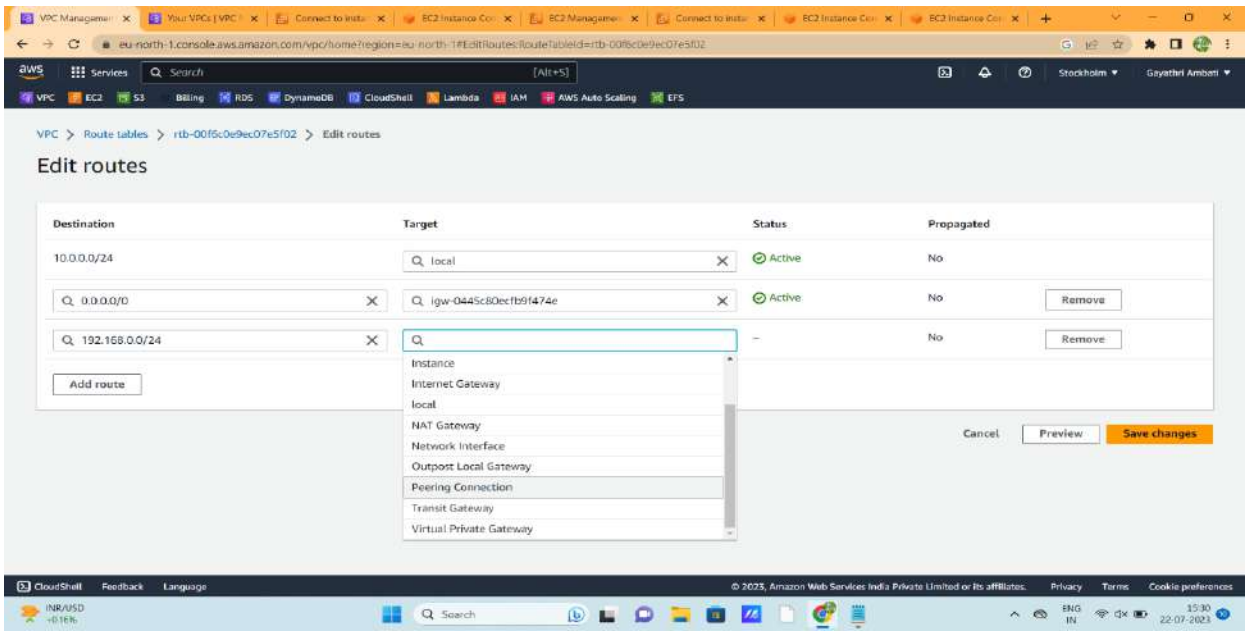
- After creating peering connection
- Click on Actions
- Click on Accept Request.



- Click on modify route tables now.

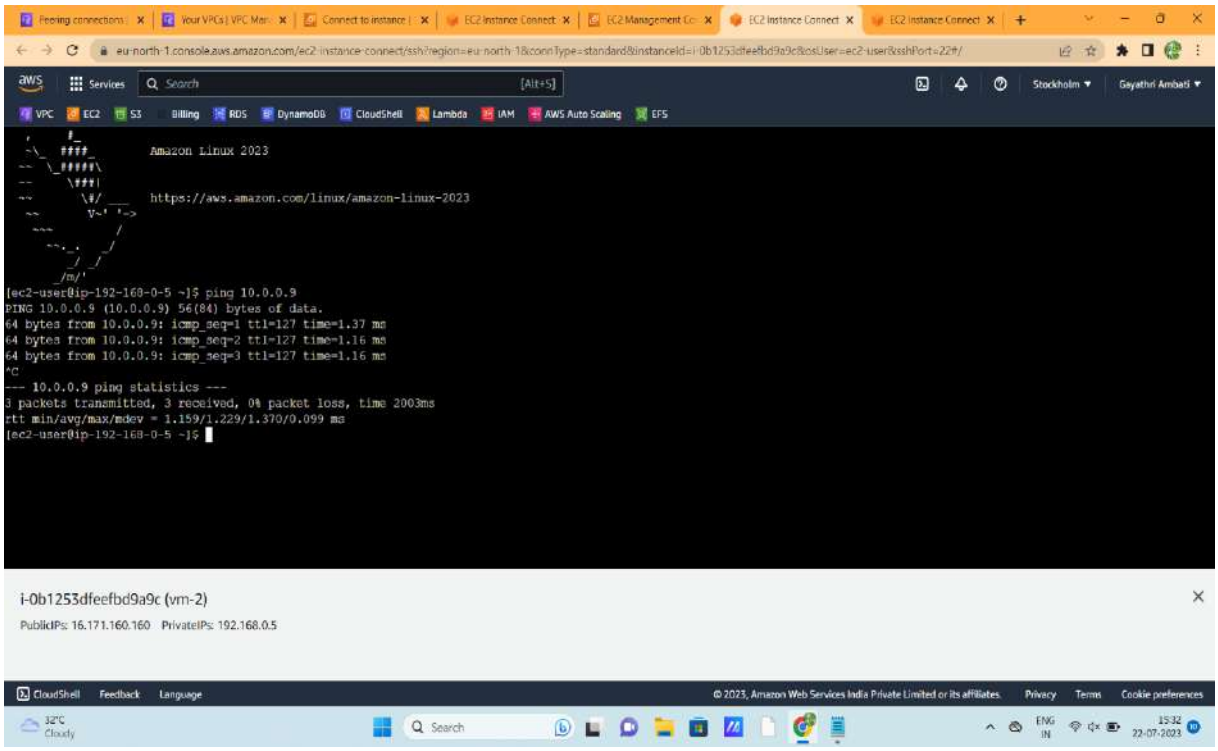


- Go to route tables and select rt1.
- Click on routes and then click on edit routes.
- In edit route dialogue box , click on add route and add CIDR (192.168.0.0/24)of VPC2 in rt1 and In target select peering connection(vpc1-vpc2).
- Click on save changes

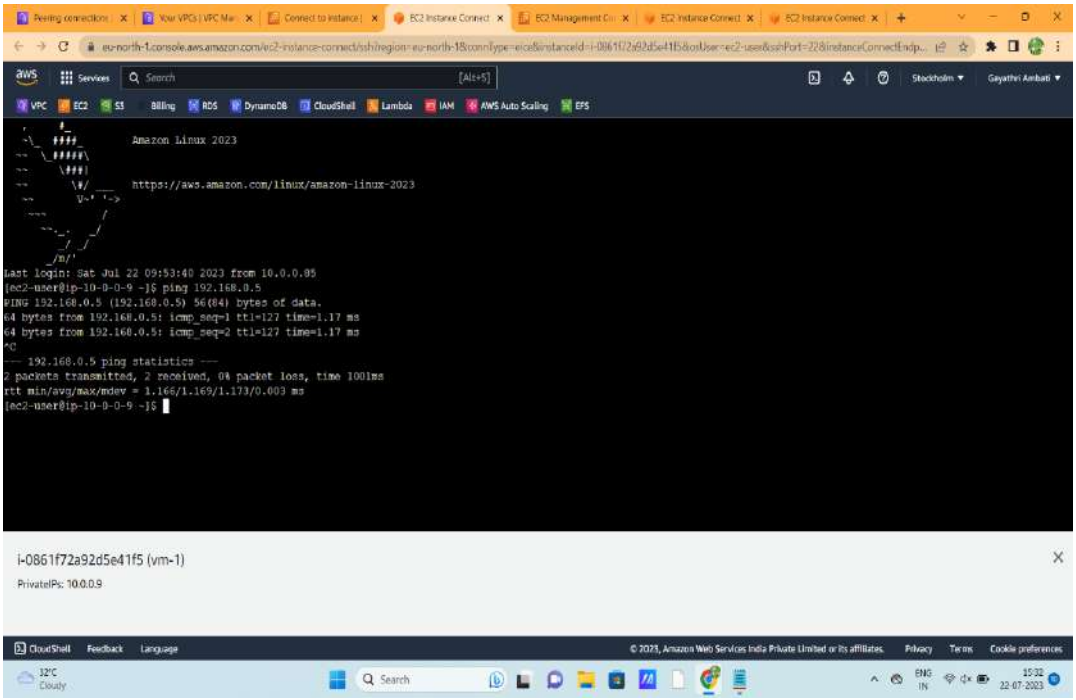


- Now connect Instance1(vm1) and Instance2(vm2) to the console.
- Ping the Instances with Private IP addresses.

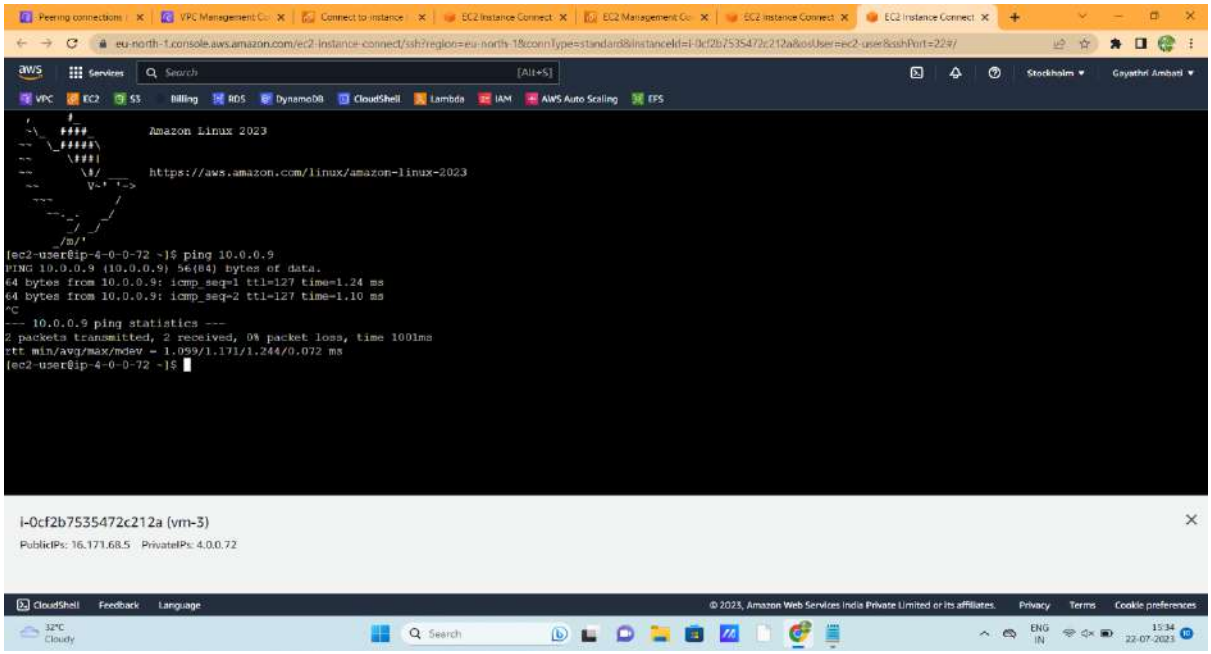
- Type Ping 10.0.0.9 which is IP address of vm1 in console of vm1

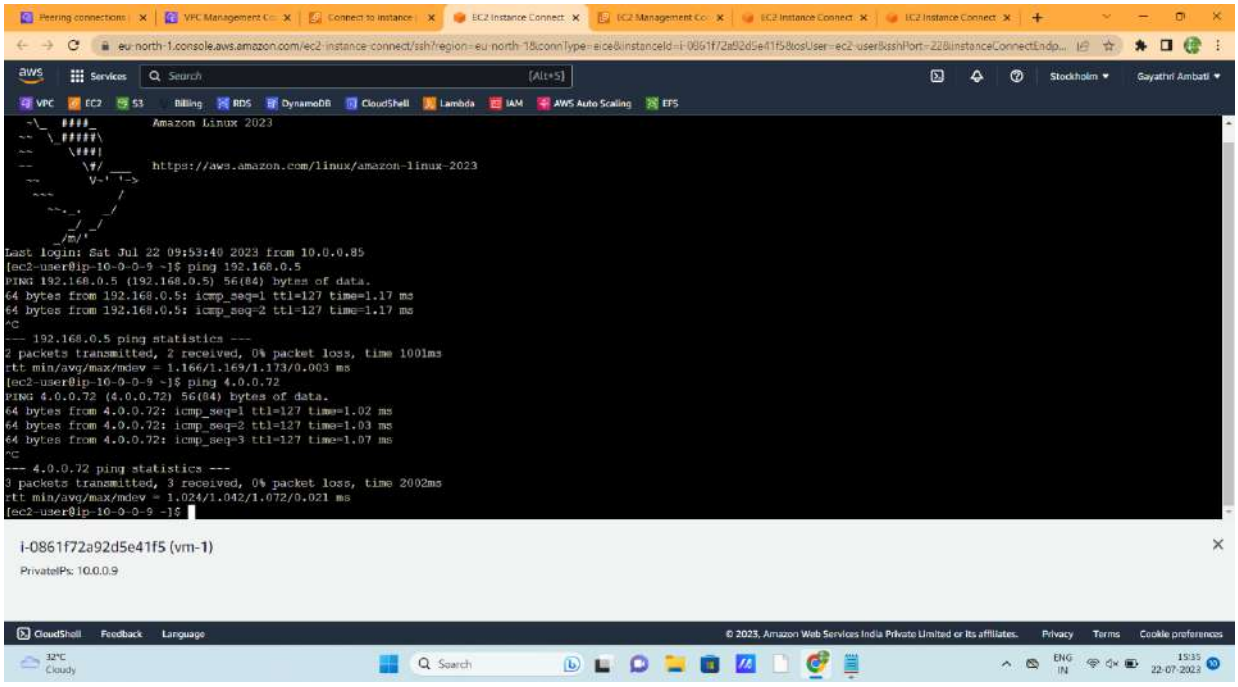


- Now Type ping 192.168.0.5 on VM-1 console



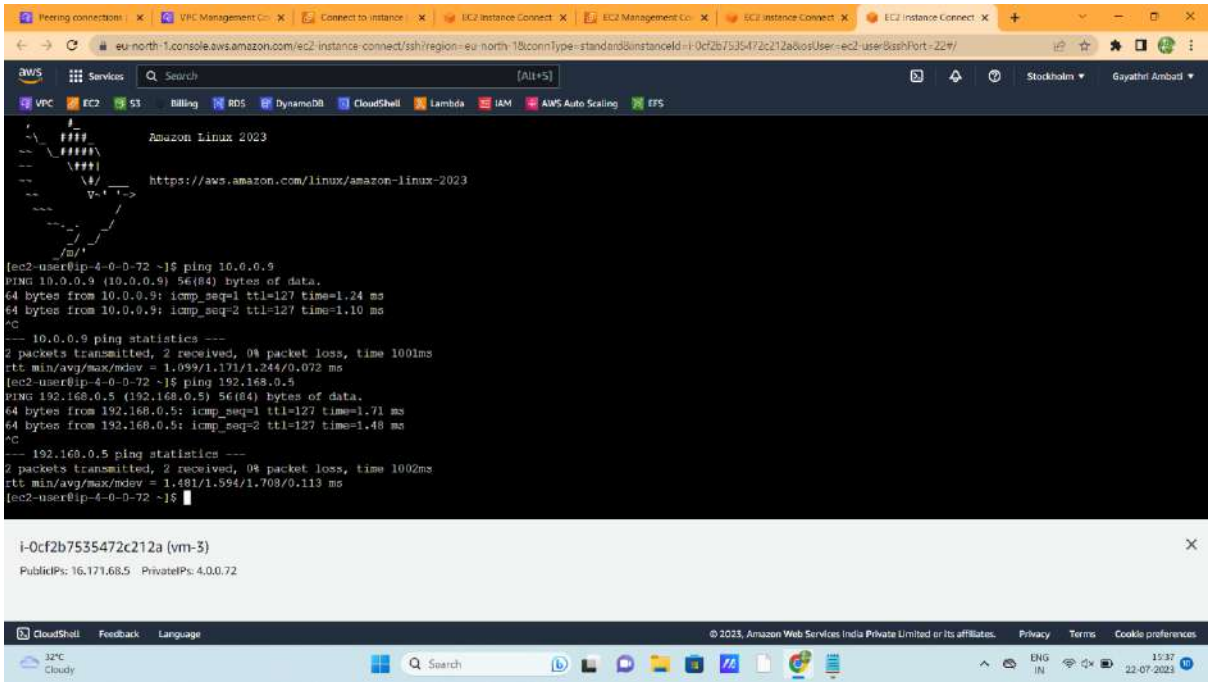
- Now create peering connection between VPC1 and VPC3 and ping private IP of VM1 in Console of VM3 and ping private IP of VM3 in Console of VM1
- Consider VPC1 as requester and VPC3 as accepter for creating peering connection.



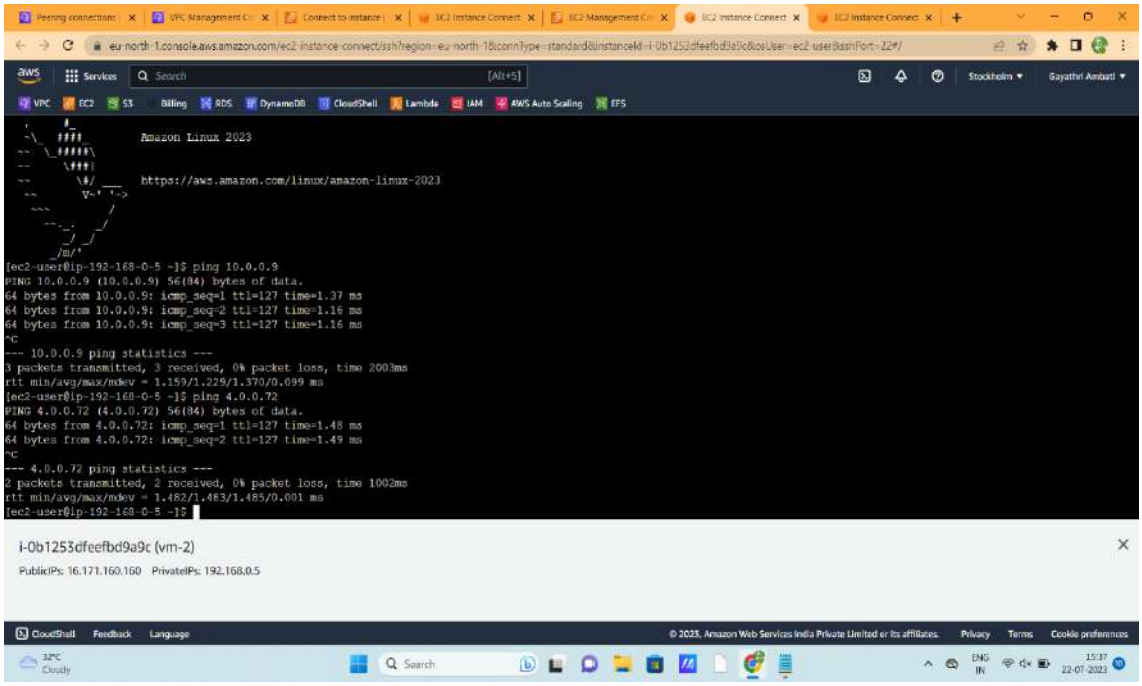


- Similarly,create peering connection between VPC2 and VPC3 and name it as VPC-2-VPC-3.

  Consider VPC2 as requester and VPC3 as accepter.
- After creating peering connection and editing rote tables . Now ping Private IP addresses of VM2(192.168.0.5) in VM3 Console.

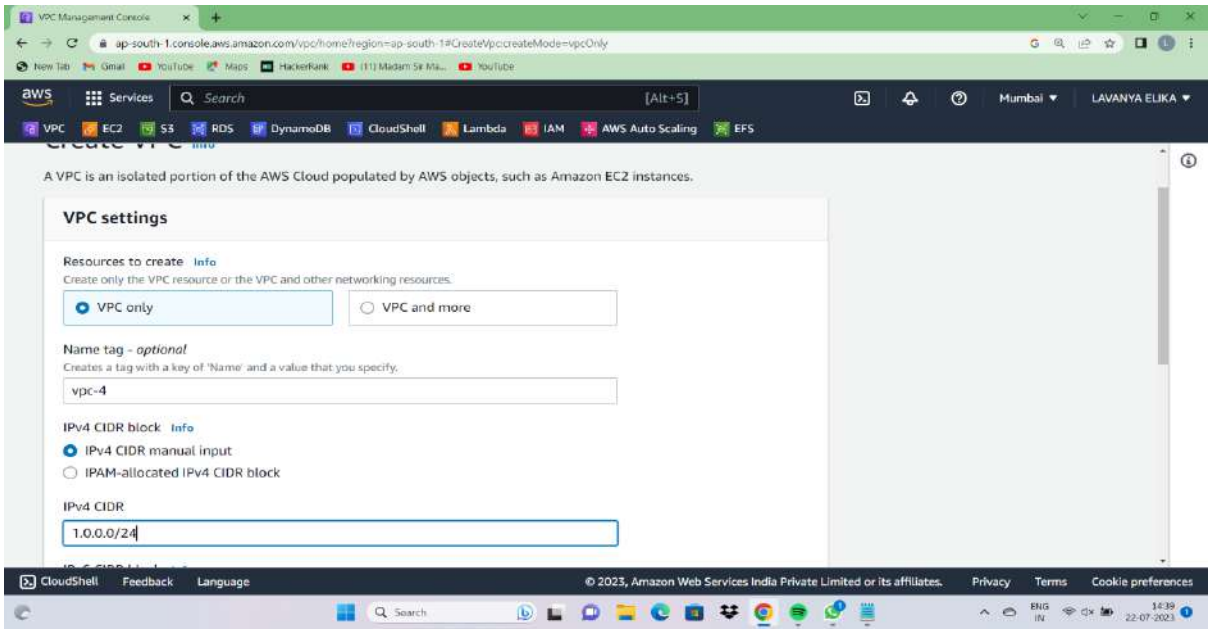


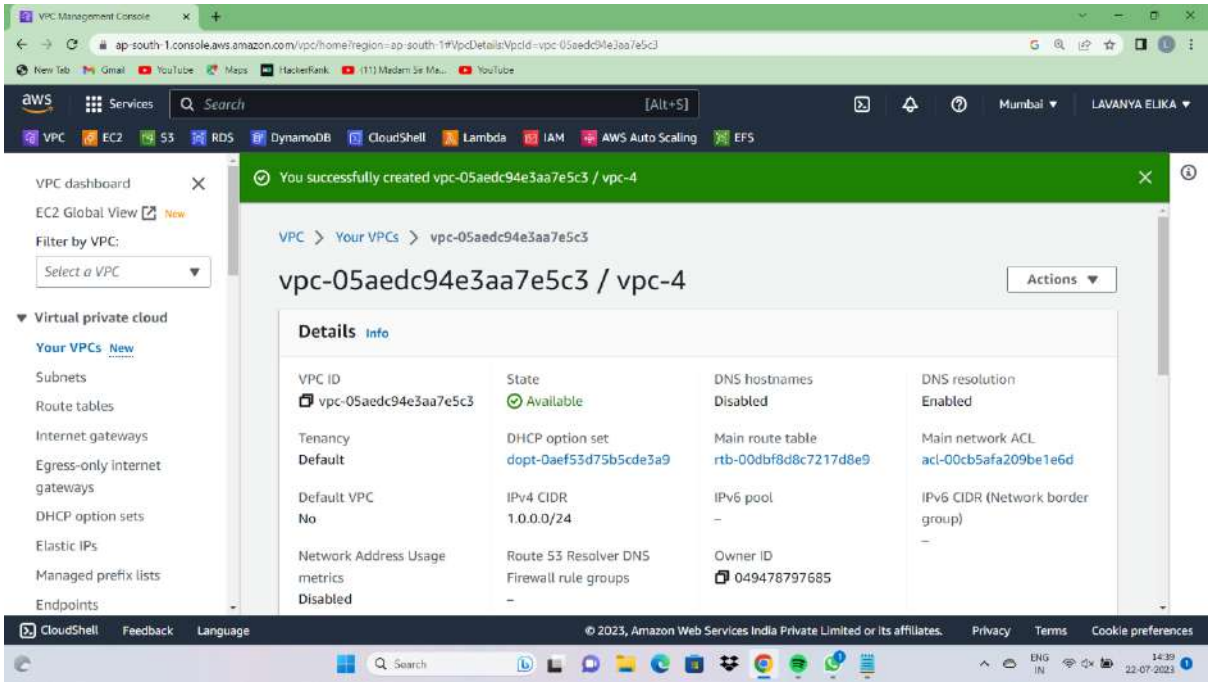
- Now ping Private IP addresses of VM3 (4.0.0.72) in VM2 Console



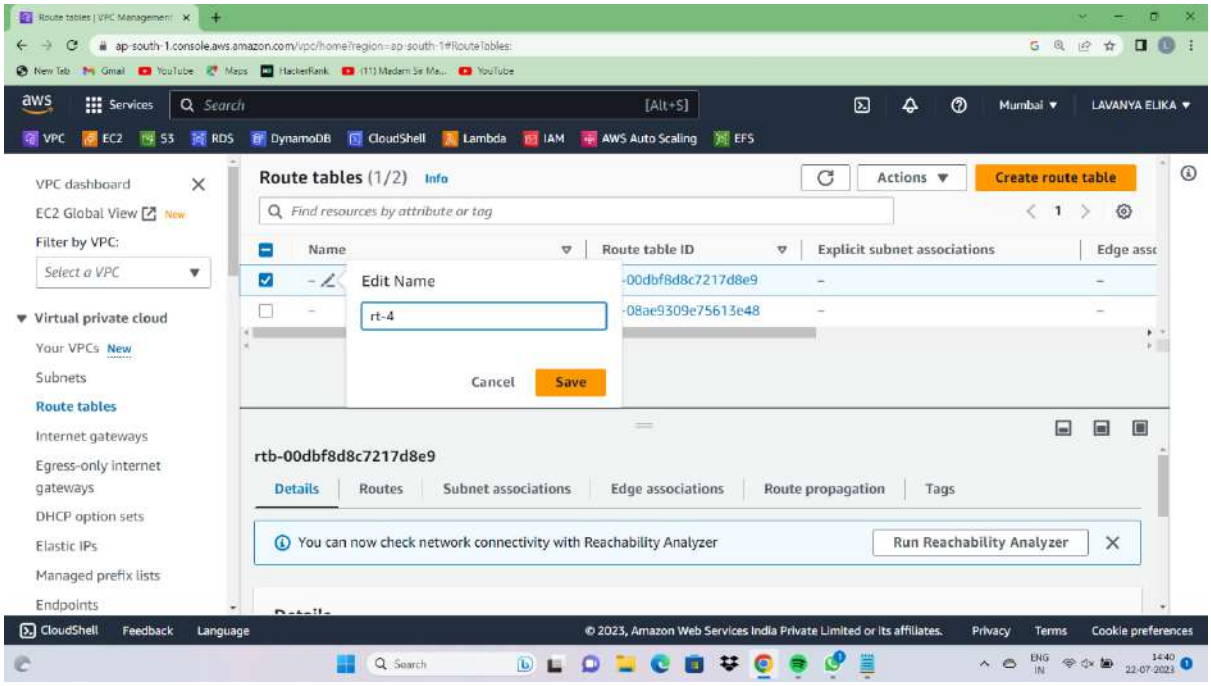
- Upto here we completed 3 peering connections in stockholm.
- Now we create another 3 Peering connections in different account and different region(Mumbai).
- Now create VPC.
- Click on the "Create VPC" option.
- Give the VPC name as vpc4 and CIDR value as 1.0.0.0/24.

- Click on create VPC.
- And create vpc-4.



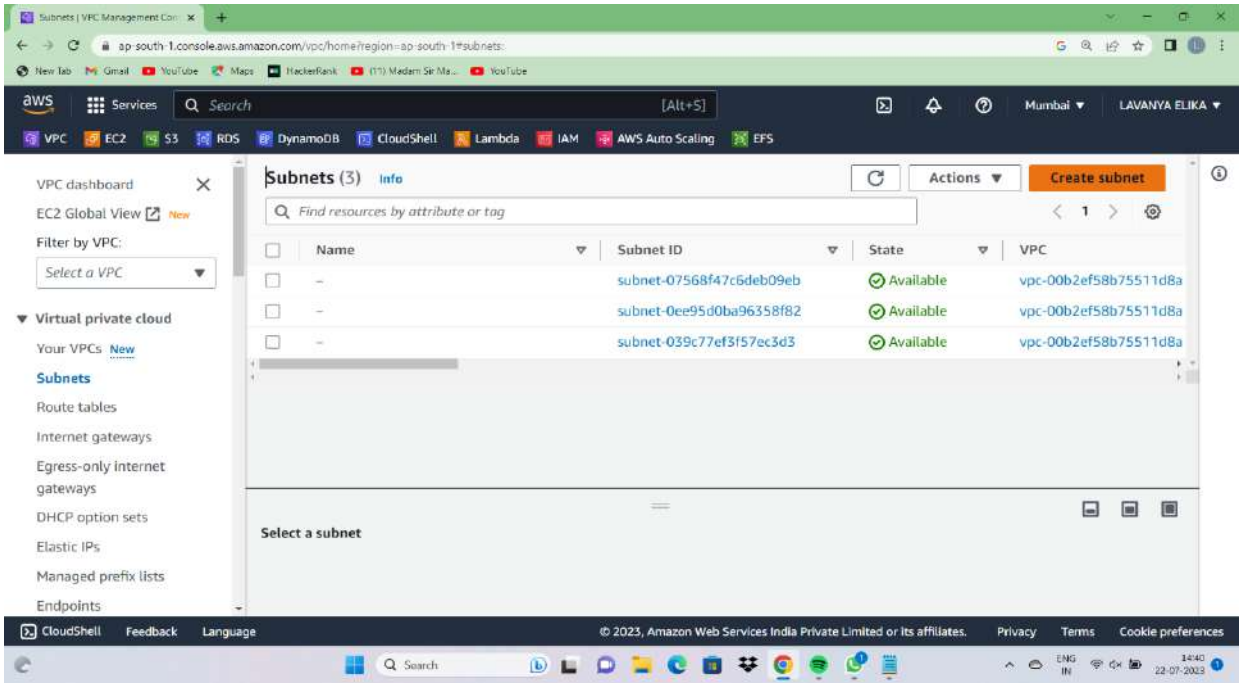


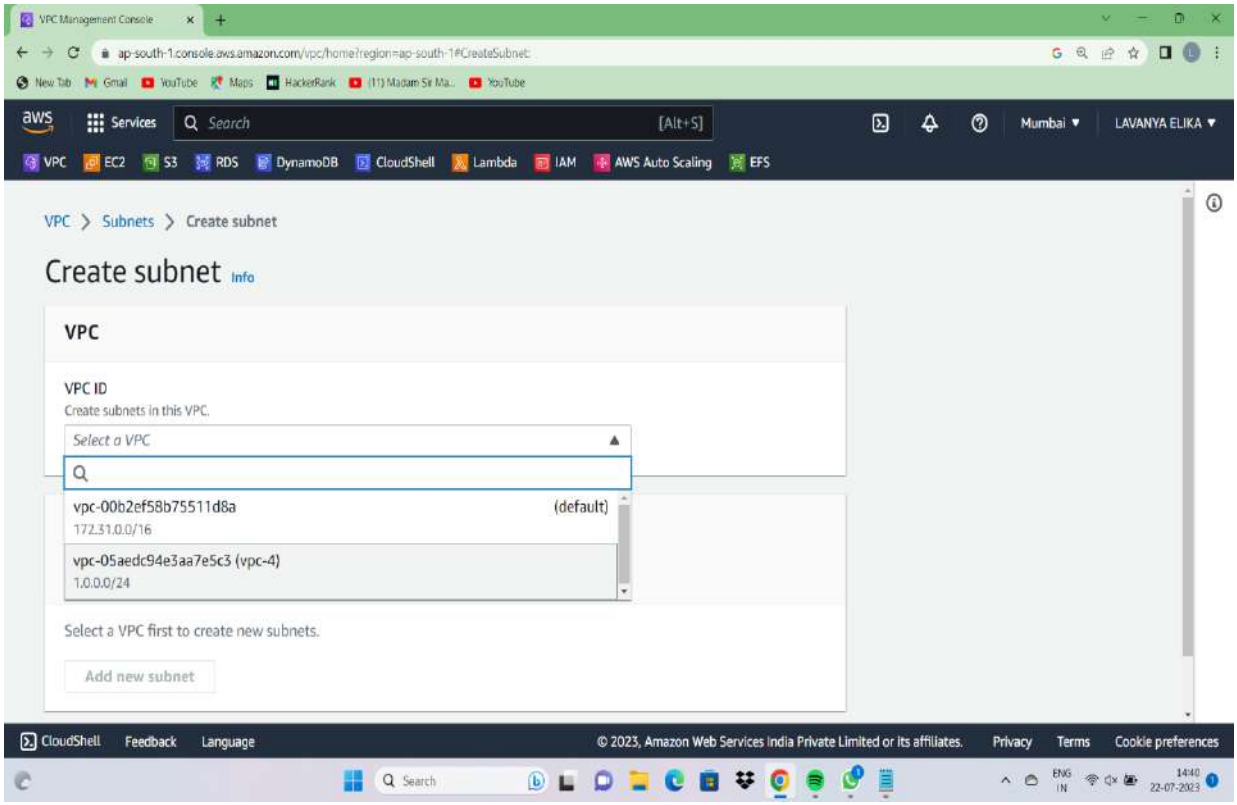
• Go to route table and edit the name of route table as rt1 public that is created along with vpc1.

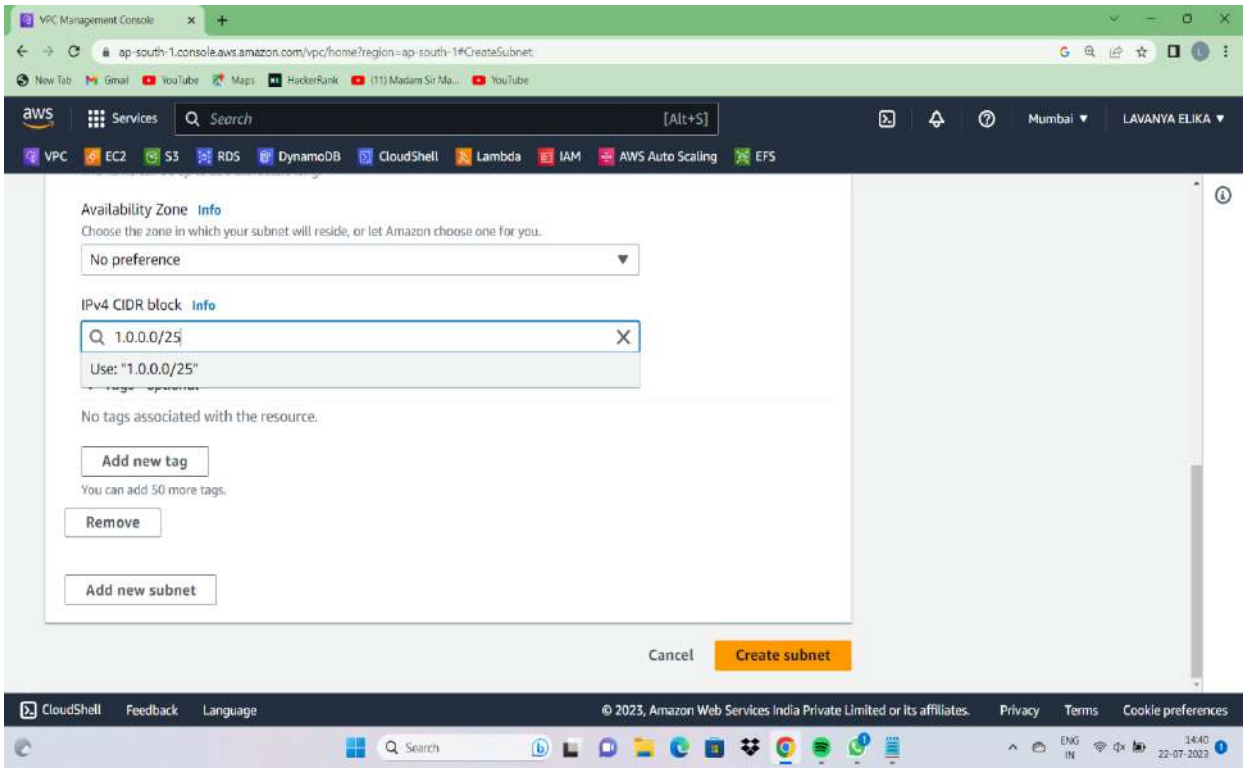


• Go to subnets and create a subnet and name it as public subnet. Select the availability zone as South-1a, CIDR as 1.0.0.0/25 and select create subnet.

• we initially have 3 default Subnets.

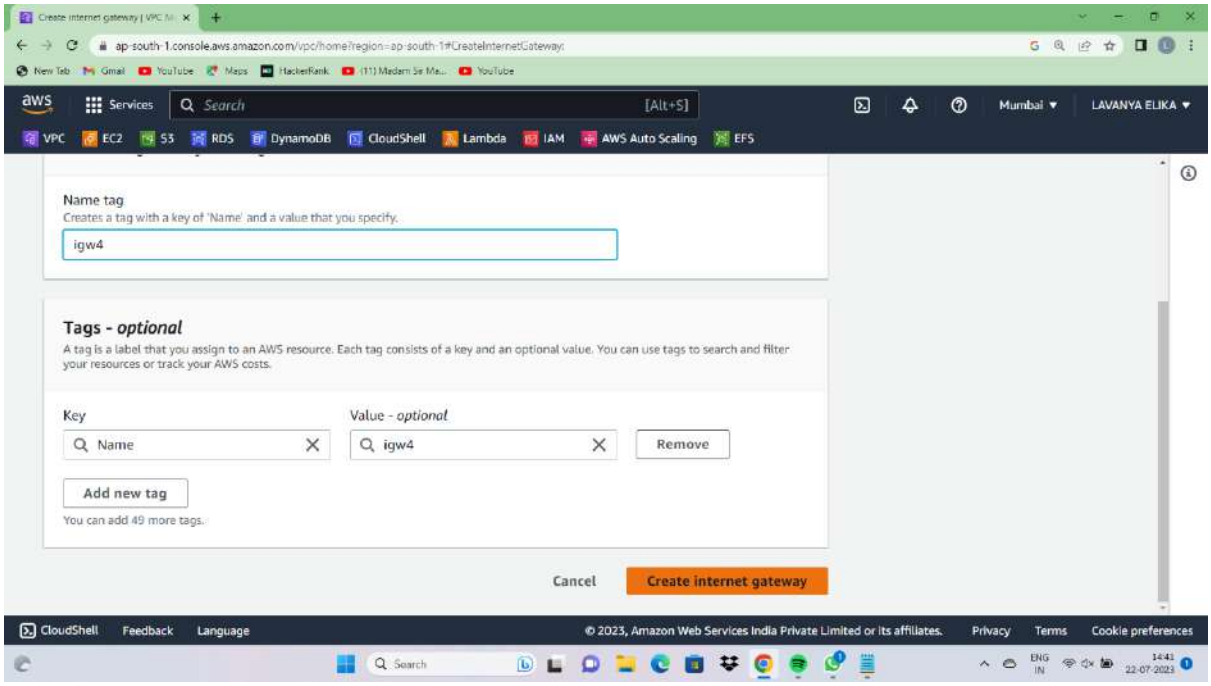




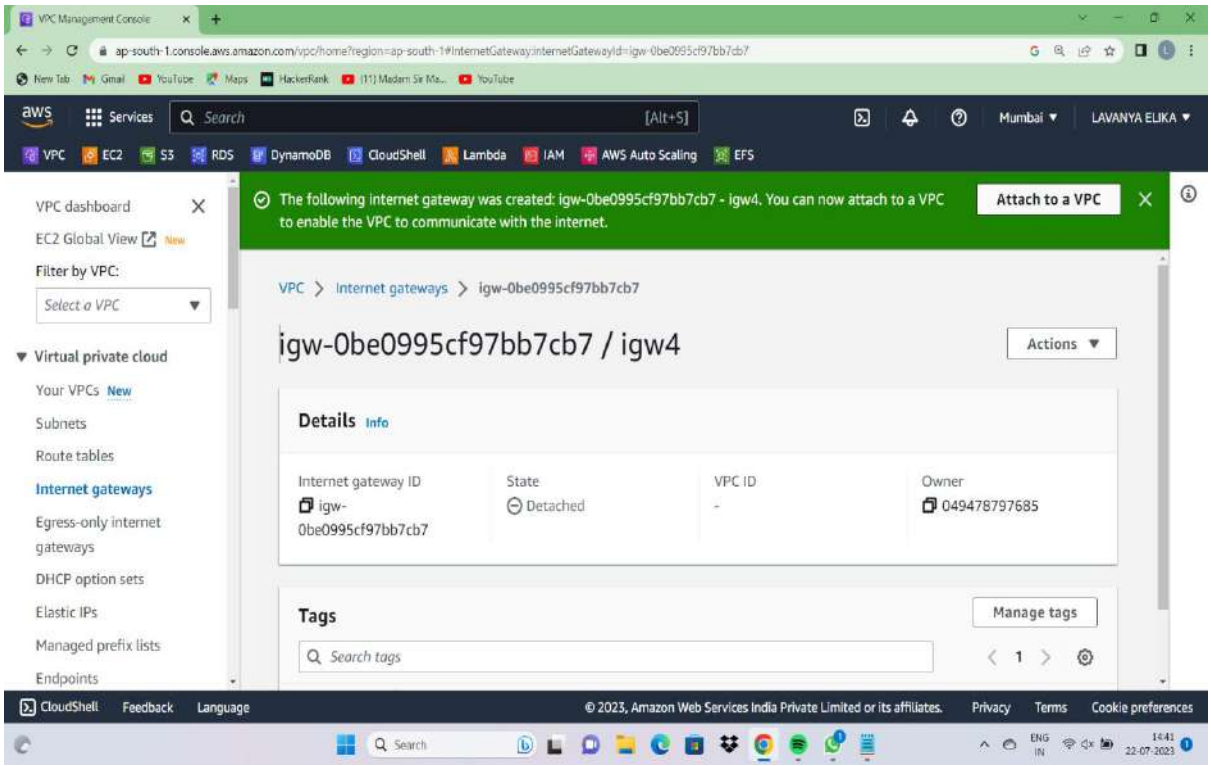


• Go to route tables, select the public route table, and go to subnet associations and select public subnet and click on save associations.

• Create an internet gateway with the name igw4.

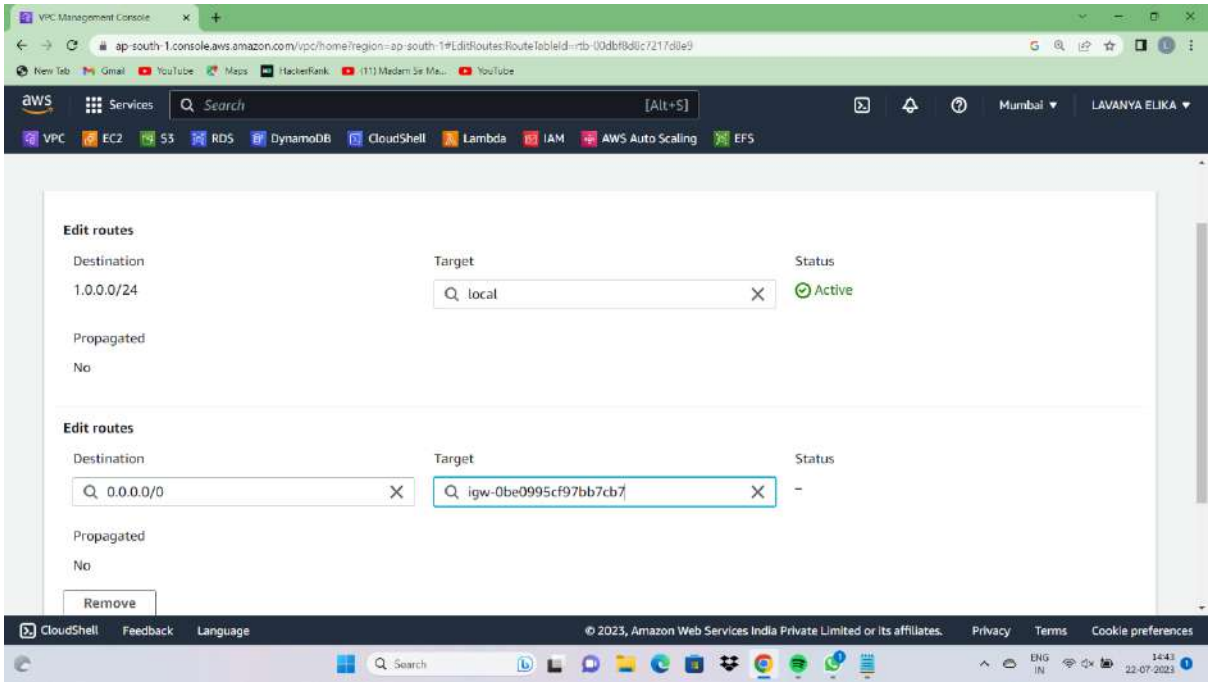


• Attach custom IGW to custom VPC.



• Internet Gate Way is successfully attached to VPC.

• Now go to route tables, select rt4 public, click on edit routes, go to add routes, and add 0.0.0.0/0 at destination and select our internet gateway in destination and save it



• Similarly create another two VPCs (VPC-5, VPC-6) using the above steps.

• Give the VPC name as vpc5 and CIDR value as 2.0.0.0/24.

• Click on create VPC and create vpc-5.

• create subnet to vpc-2 with CIDR- 2.0.0.0/25.

• Go to route tables, select the public route table, and go to subnet associations and select public subnet and click on save associations.

• Create an internet gateway with the name igw5.

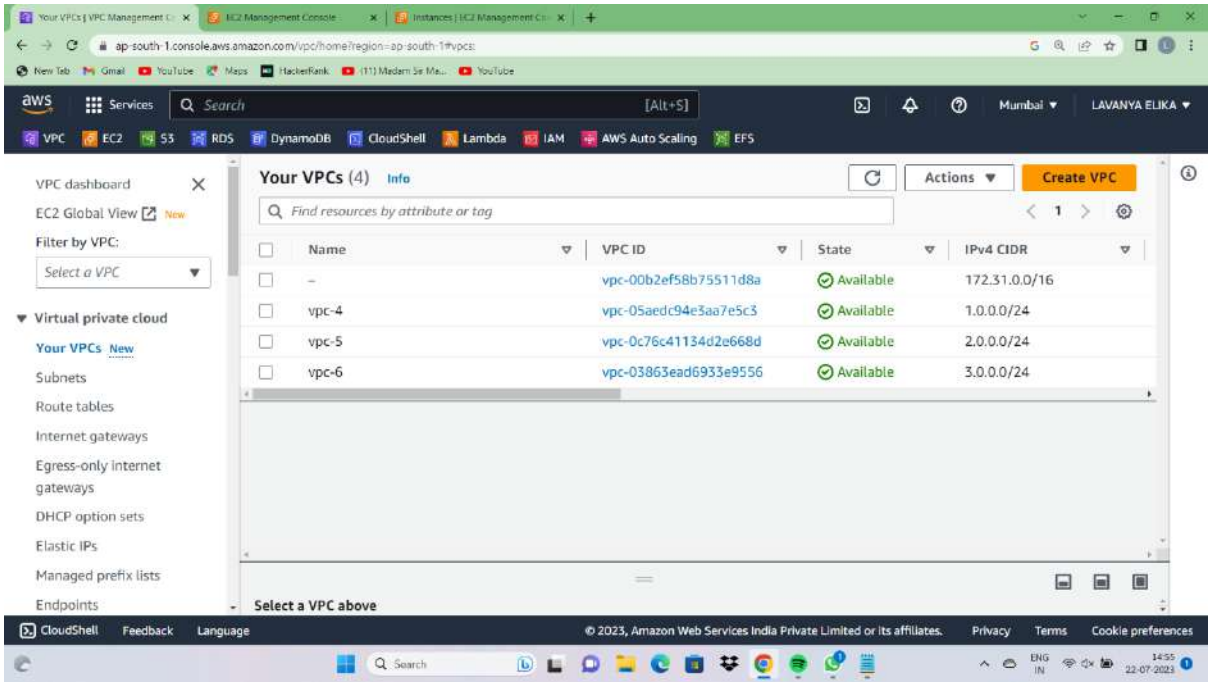
• Attach custom IGW to custom VPC.

• Internet Gate Way is successfully attached to VPC.

• Now go to route tables, select rt5 public, click on edit routes, go to add routes, and add 0.0.0.0/0 at destination and select our internet gateway in destination and save it.

• Give the VPC name as vpc-6 and CIDR value as 3.0.0.0/24.

• Click on create VPC and create vpc-6.



•Creating an EC2 instance in a public subnet:

• Name your instance as VM-4.

• Select “Amazon Linux 2 AMI”.

• Instance type “t2. micro”.

• Select your existing key pair.

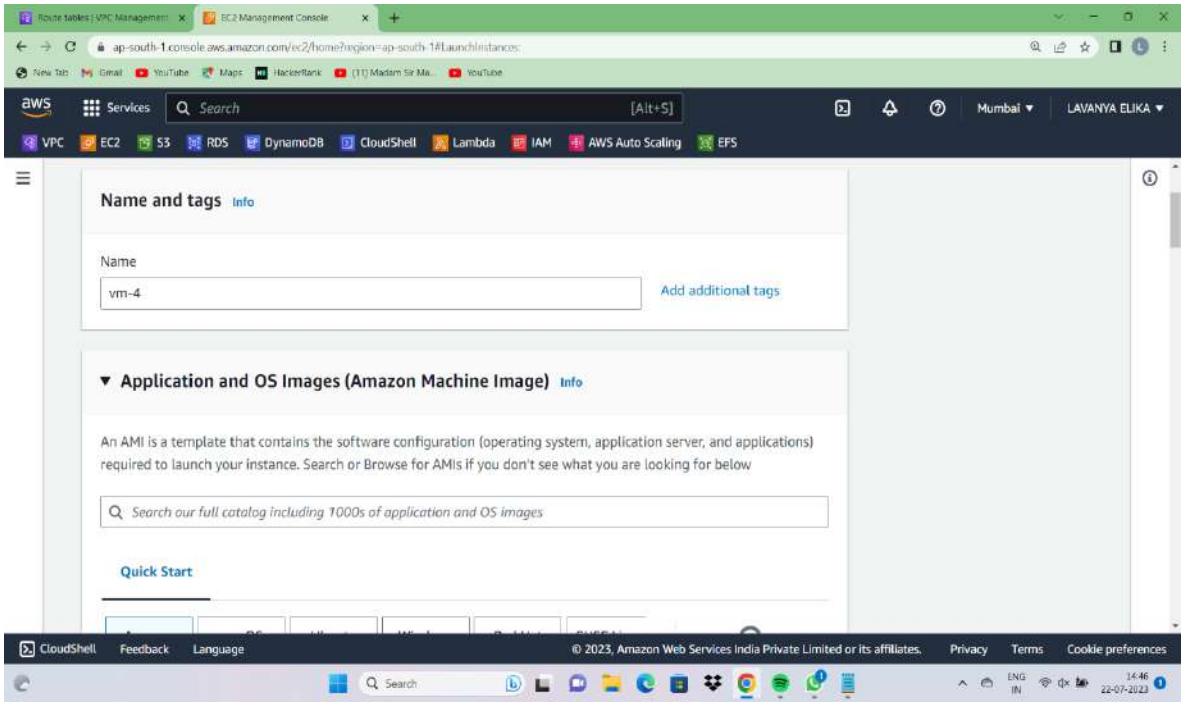
• Select your custom VPC, public subnet and enable the auto-assign public IP.

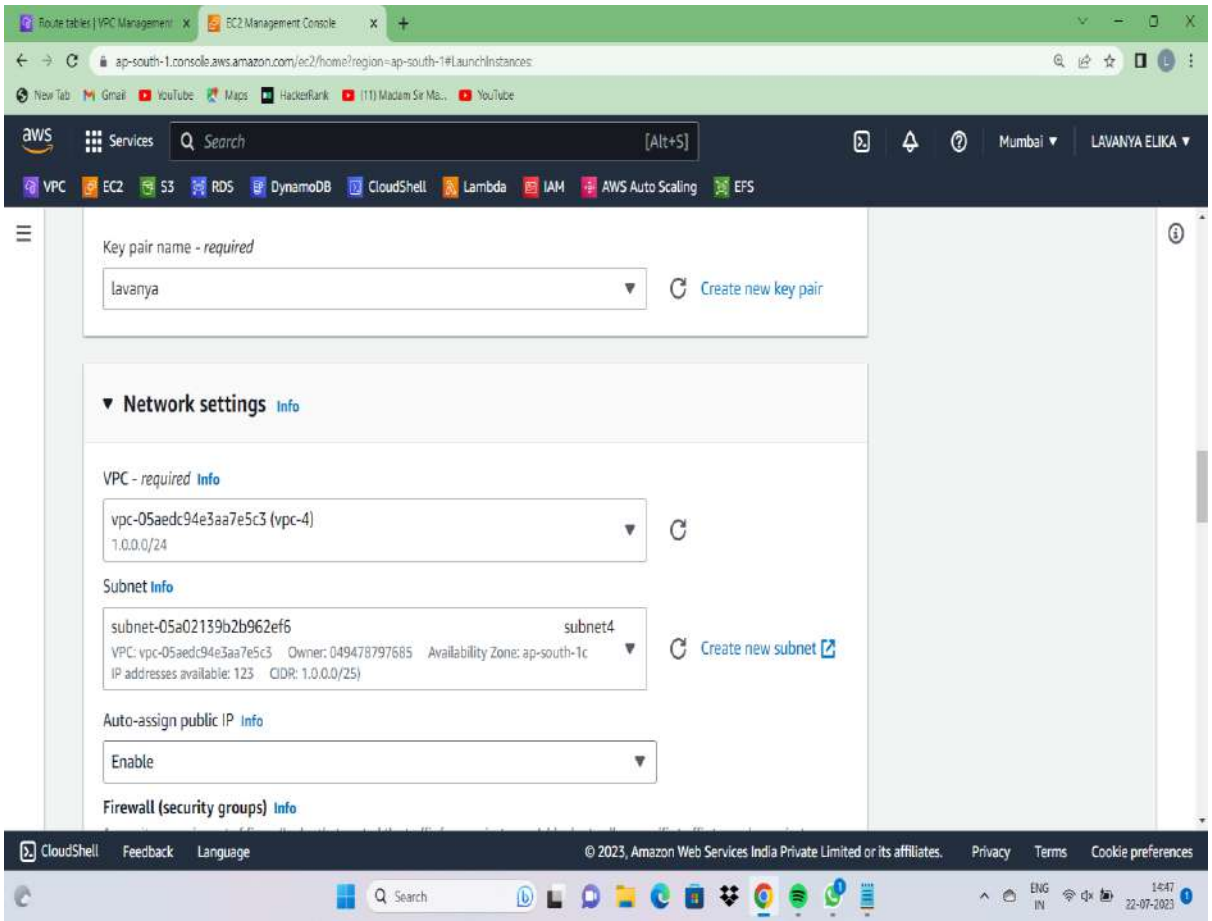
• In the security group section, select availability zone as South-1a.

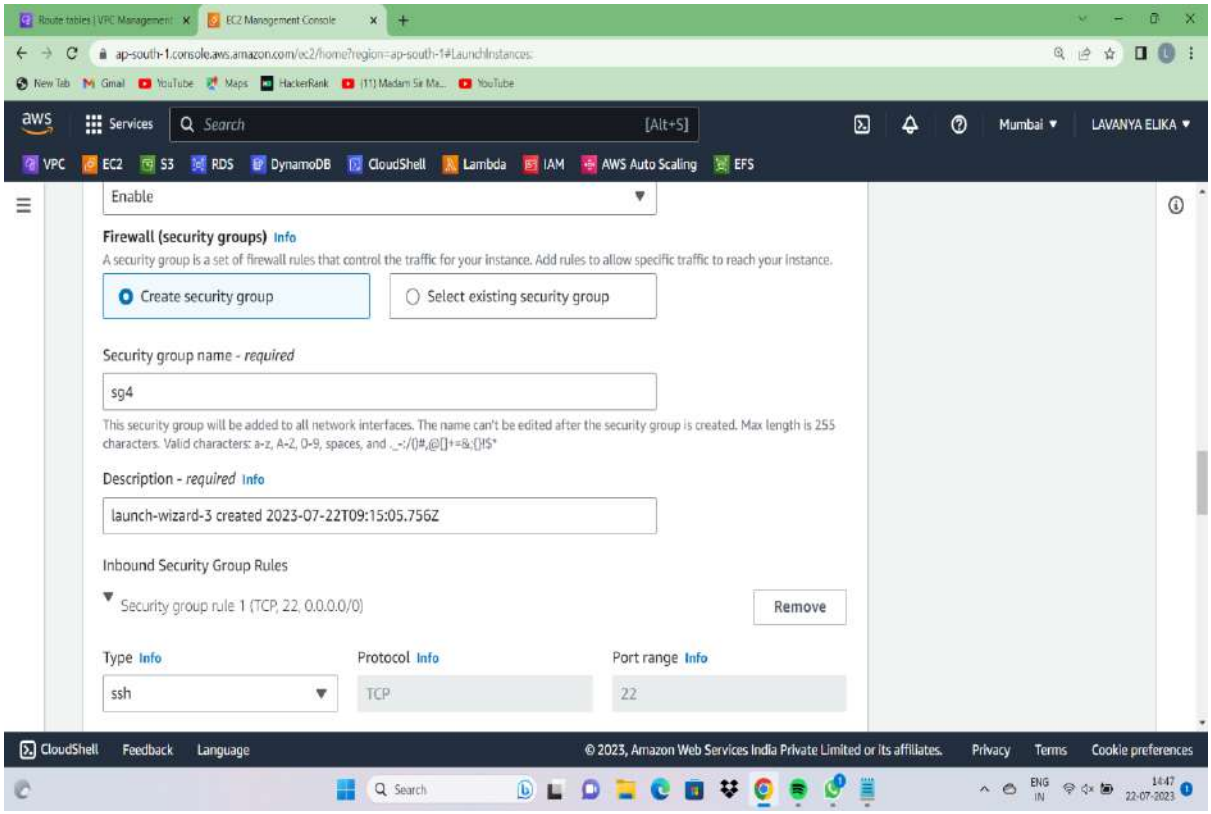
• create a new security group, add security groups that supports SSH and All Traffic.

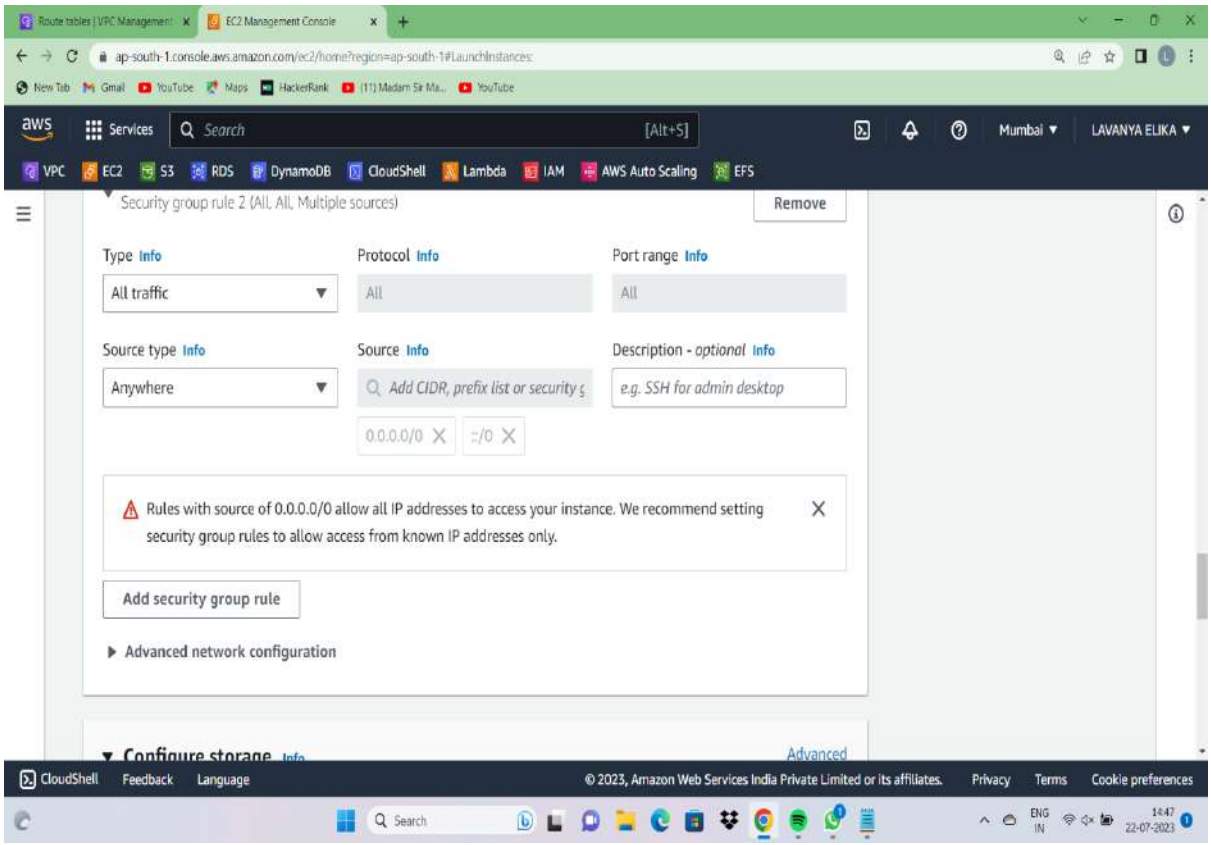
• Similarly create the third Instance and fourth instances and name them as VM-5 , VM-6.

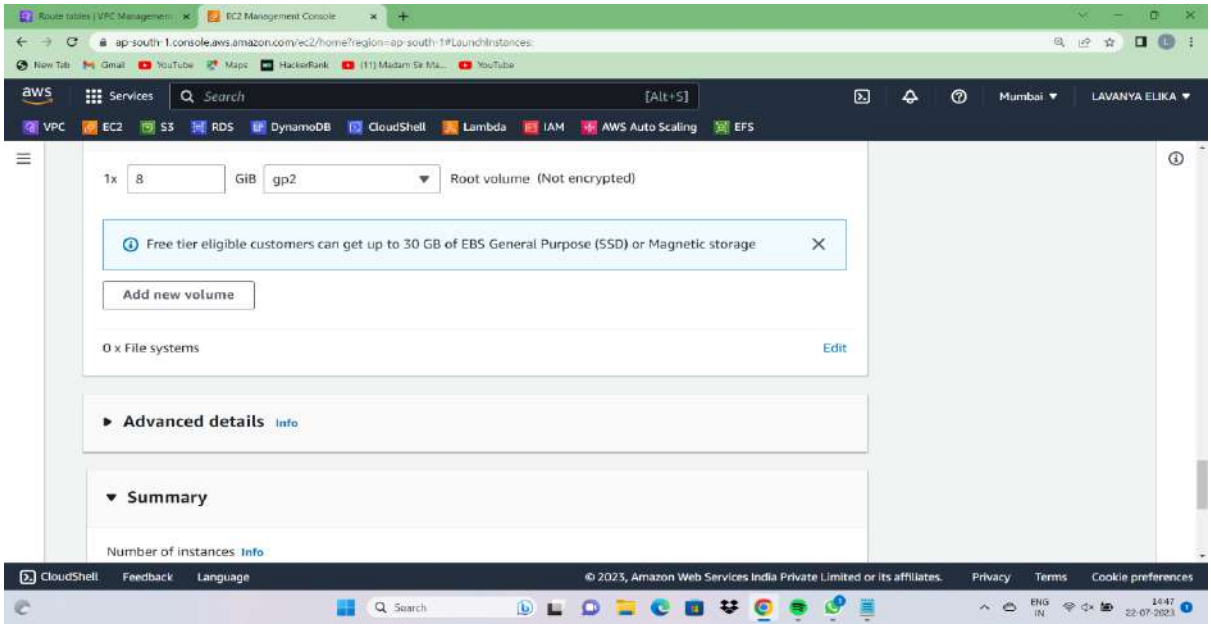
• Launch your Instance.


Name and tags Info
Name
vm-4
Add additional tags
Application and OS Images (Amazon Machine Image) Info
An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below
Search our full catalog including 1000s of application and OS images
Quick Start


Key pair name - required
lavanya
Create new key pair
Network settings Info
VPC - required Info
vpc-05aedc94e3aa7e5c3 (vpc-4)
1.0.0.0/24
Subnet Info
subnet-05a02139b2b962ef6
subnet4
VPC: vpc-05aedc94e3aa7e5c3 Owner: 049478797685 Availability Zone: ap-south-1c
IP addresses available: 123 CIDR: 1.0.0.0/25)
Create new subnet
Auto-assign public IP Info
Enable
Firewall (security groups) Info


Route tables | VPC Management
EC2 Management Console
ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#LaunchInstances:
aws
Services
Search
[Alt+S]
Mumbai
LAVANYA ELIKA
VPC
EC2
S3
RDS
DynamoDB
CloudShell
Lambda
IAM
AWS Auto Scaling
EFS
Enable
Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.
Create security group
Select existing security group
Security group name - required
sg4
This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and ._-:/()#,@[]+=&;{}!$*
Description - required Info
launch-wizard-3 created 2023-07-22T09:15:05.756Z
Inbound Security Group Rules
Security group rule 1 (TCP, 22, 0.0.0.0/0)
Remove
Type Info
ssh
Protocol Info
TCP
Port range Info
22
CloudShell
Feedback
Language
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences

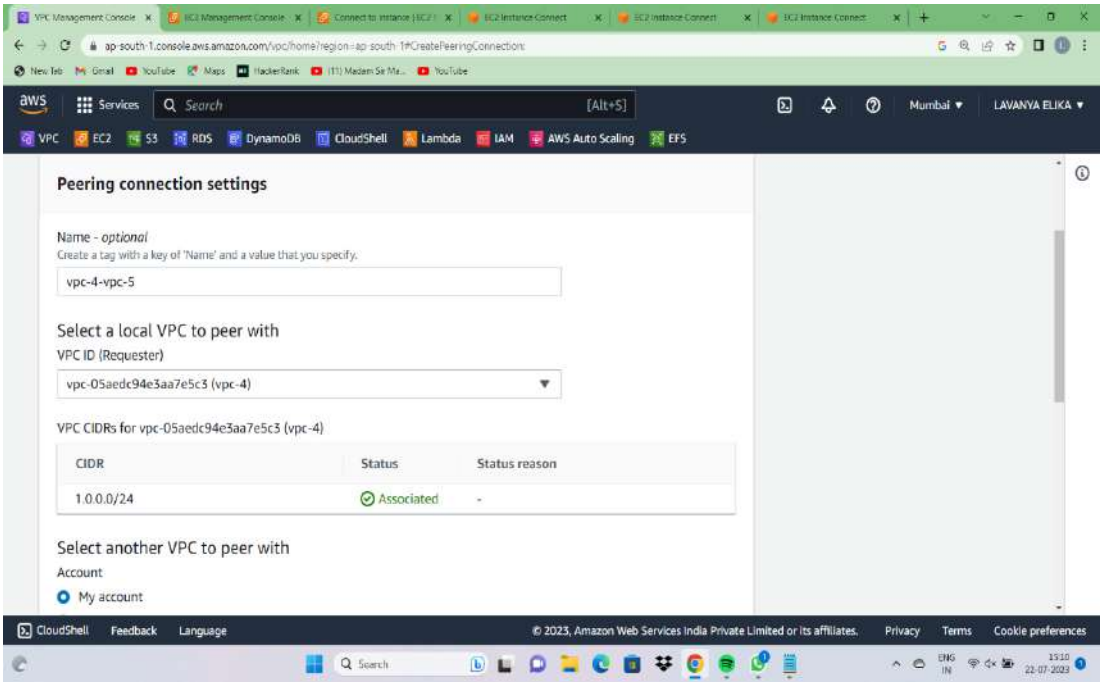

Route tables | VPC Management
EC2 Management Console
ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#LaunchInstances:
aws
Services
Search
[Alt+S]
Mumbai
LAVANYA ELIKA
VPC
EC2
S3
RDS
DynamoDB
CloudShell
Lambda
IAM
AWS Auto Scaling
EFS
Security group rule 2 (All, All, Multiple sources)
Remove
Type Info
All traffic
Protocol Info
All
Port range Info
All
Source type Info
Anywhere
Source Info
Add CIDR, prefix list or security g
0.0.0.0/0
::/0
Description - optional Info
e.g. SSH for admin desktop
Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.
Add security group rule
Advanced network configuration
Configure storage Info
Advanced
CloudShell
Feedback
Language
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences



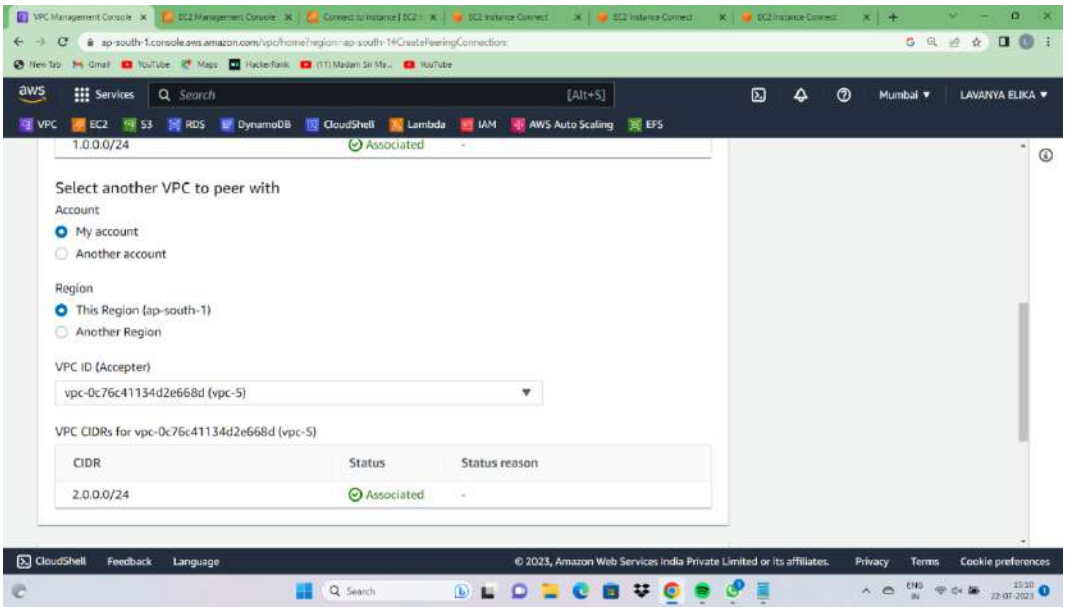
- Finally the instances are launched connect them by going actions option.

Peering Connection

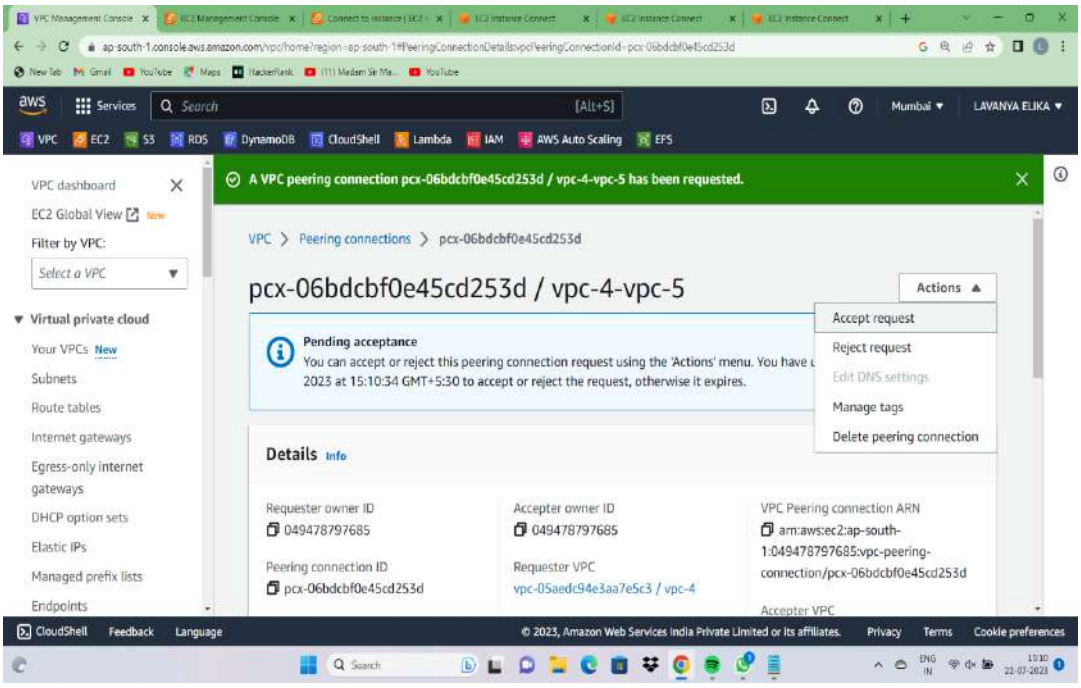
- Now we have to create peering connection between these 3 VPCs
- Go to peering connections in VPC Service
- Click on create peering connection



- For creating peering connection between VPC4 and VPC6 .Give name as vpc-4-vpc-6 and select VPC(requester) as VPC4
- Select VPC (acceptor) as VPC5and click on create peering connection.

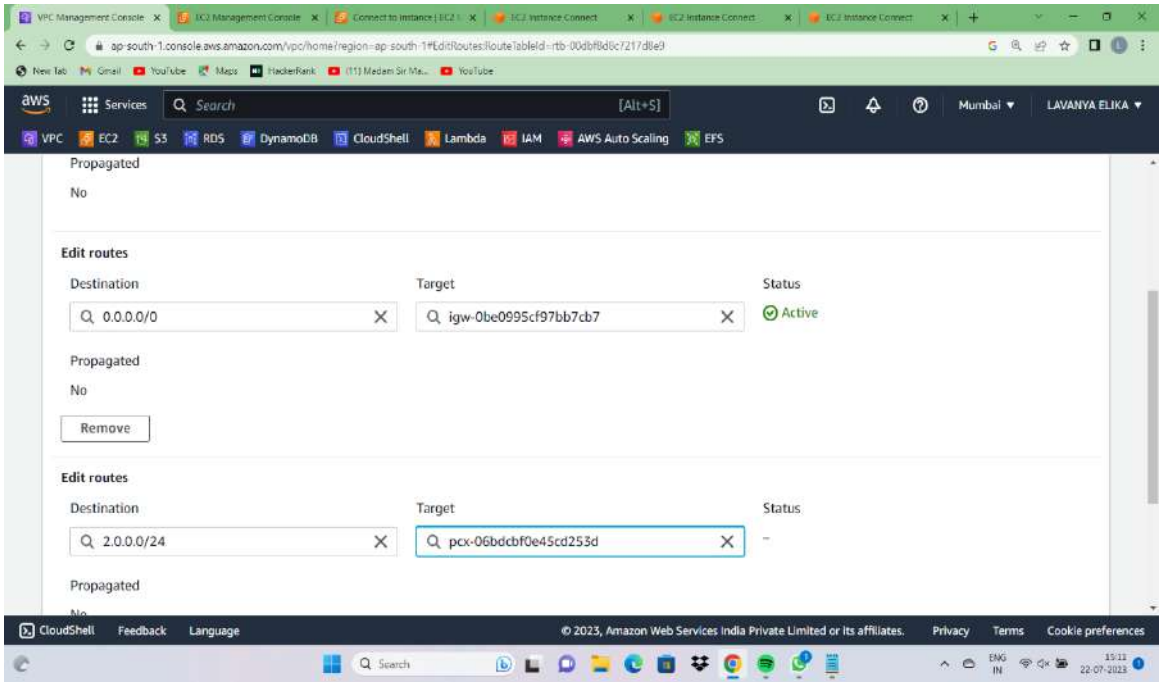


- After creating peering connection
- Click on Actions
- Click on Accept Request.

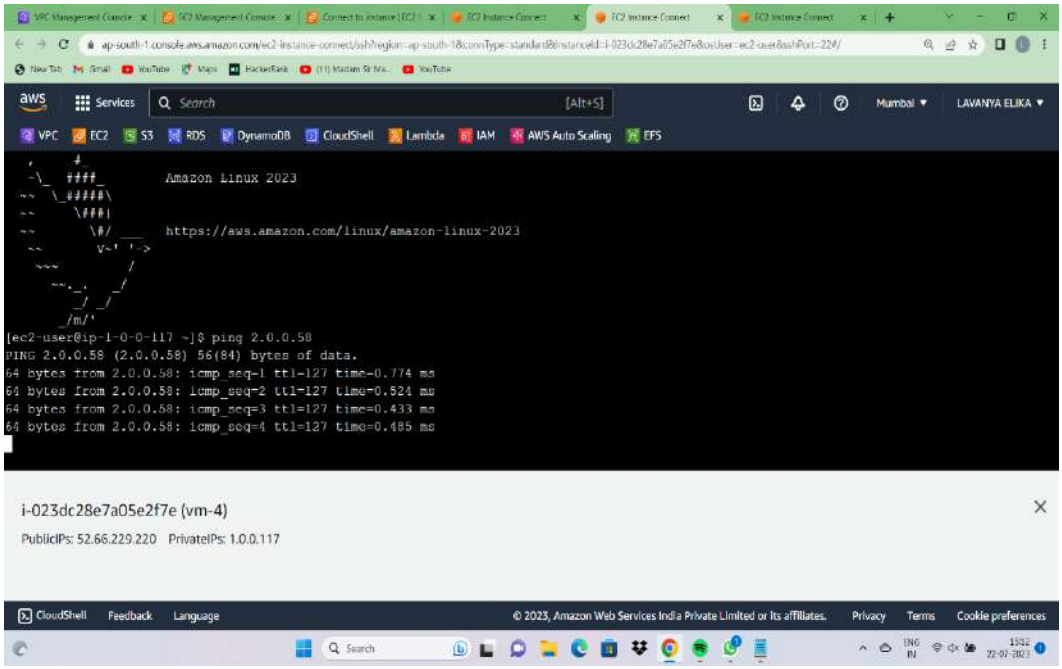


- Click on modify route tables now.
- Go to route tables and select rt4.

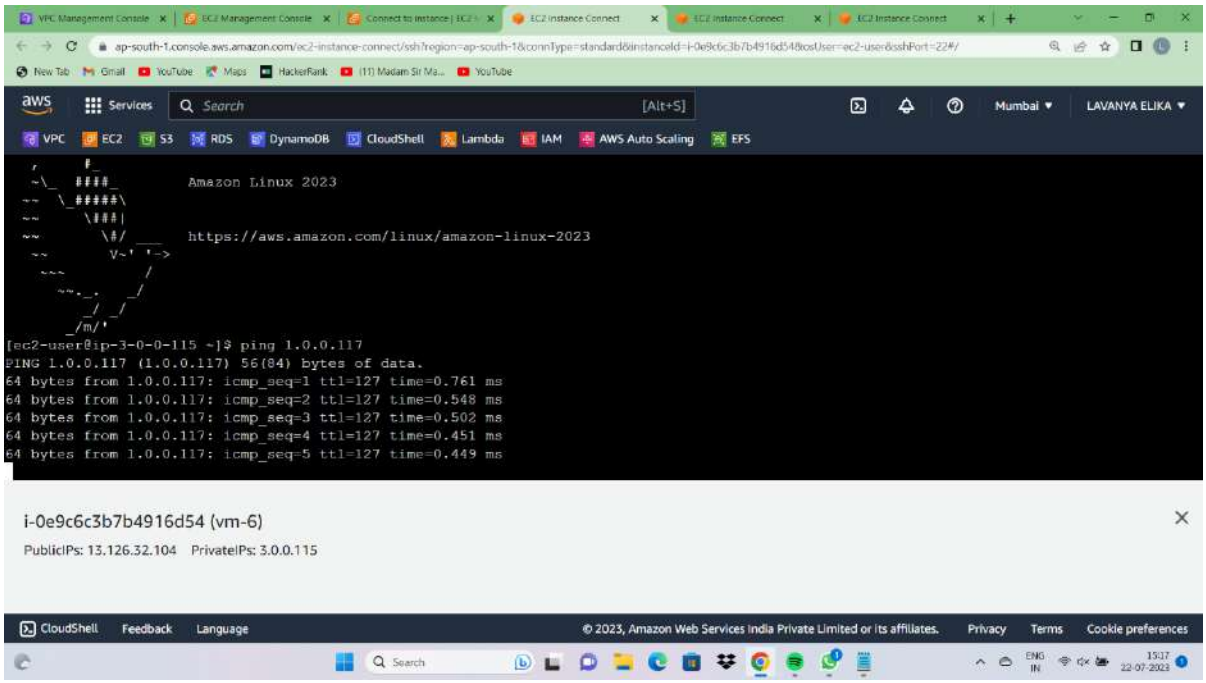
- Click on routes and then click on edit routes.
- In edit route dialogue box, click on add route and add CIDR (2.0.0.0/ 24) of VPC5 in rt4 and In target select peering connection(vpc4-vpc5).

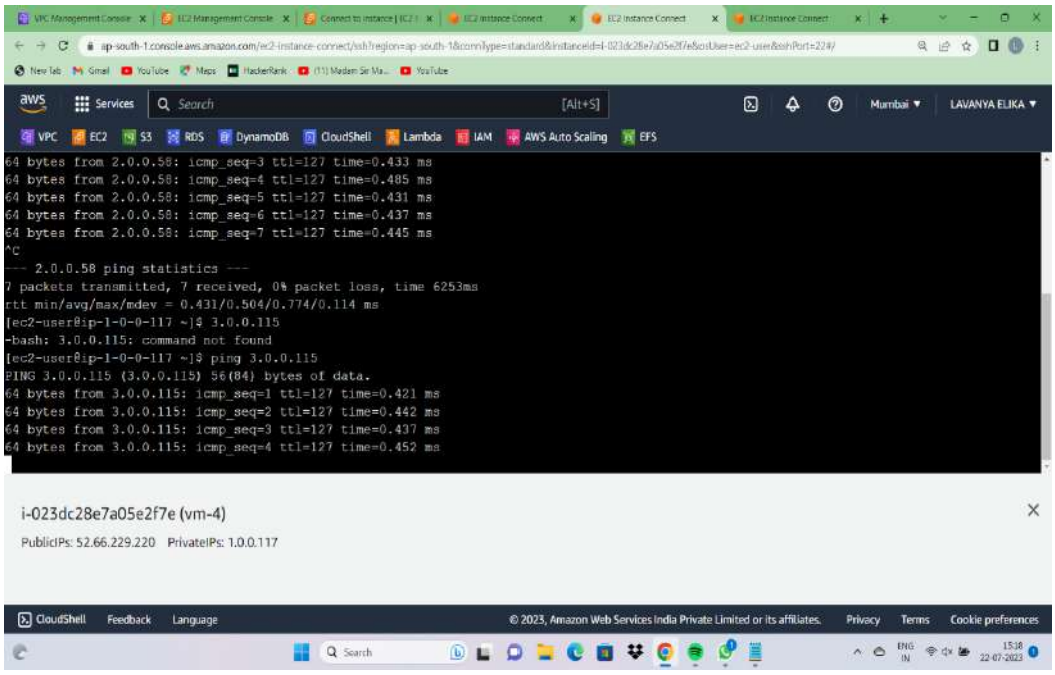


- Now connect Instance1(vm4) and Instance2(vm5) to the console
- Ping the Instances with Private IP addresses.
- Type Ping 2.0.0.0.9 which is IP address of vm1 in console of vm4.

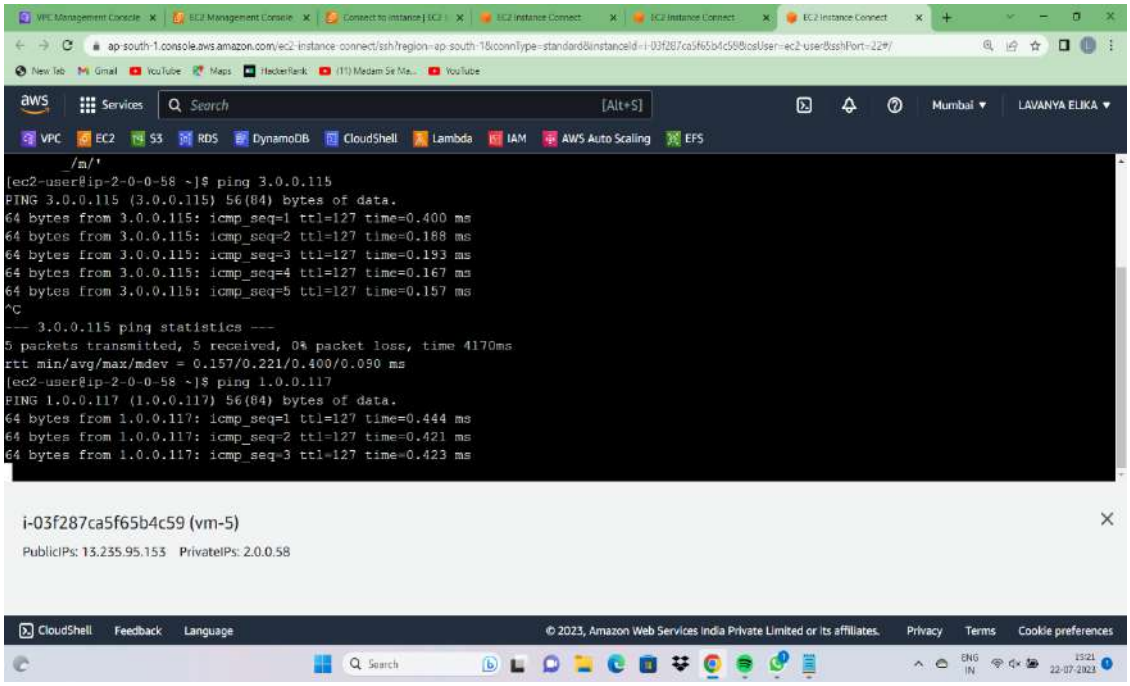


- Now Type ping 1.0.0.117 on VM-5 console





- Now create peering connection between VPC4 and VPC6 and ping private IP of VM4 in Console of VM6 and ping private IP of VM6 in Console of VM4
- Consider VPC4 as requester and VPC6 as accepter for creating peering connection.



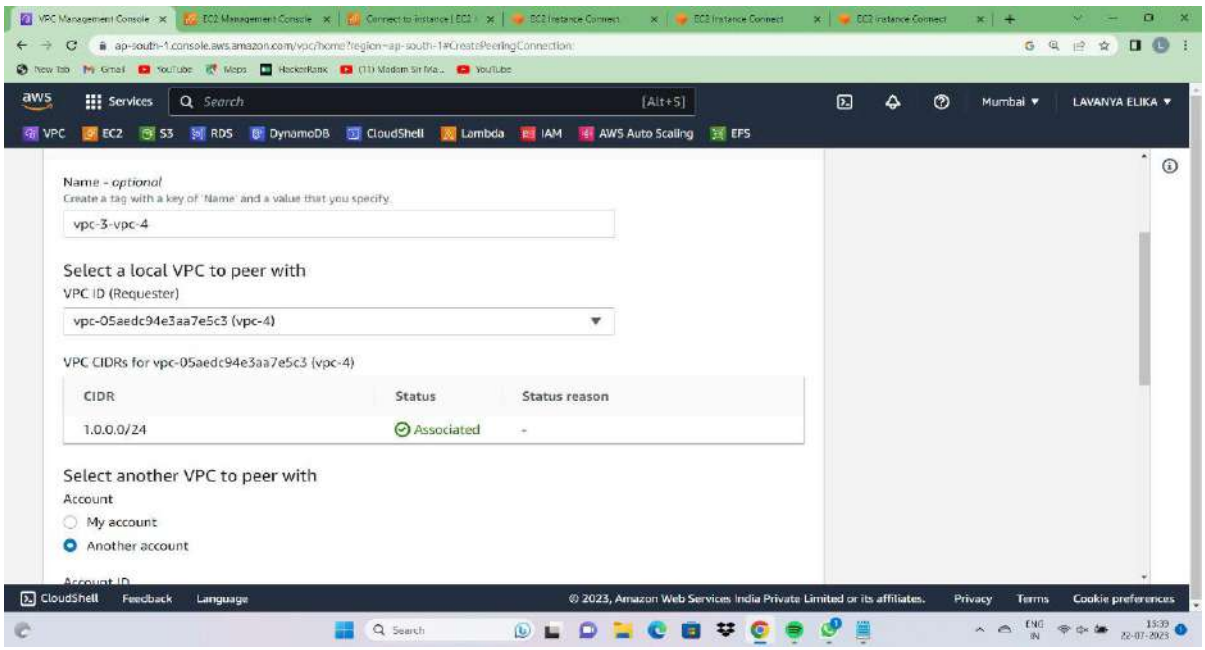
- Similarly,create peering connection between VPC5 and VPC6 and name it as VPC-5-VPC-6.

  Consider VPC5 as requester and VPC6 as accepter.
- After creating peering connection and editing rote tables . Now ping Private IP addresses of VM2(2.0.0.5) in VM3 Console.
- Now ping Private IP addresses of VM3 (3.0.0.115) in VM2 Console
- Upto here, we completed 3 peering connections in ap-south-1 belongs to 049478797685 and another 3 peering connections in eu-north-1 belongs to
- Until now we completed peering connections vpc1-vpc2, vpc1-vpc3, vpc2-vpc-3, vpc4-vpc5, vpc4-vpc6,vpc5-vpc6

Peering connection between two cross regions and two different accounts

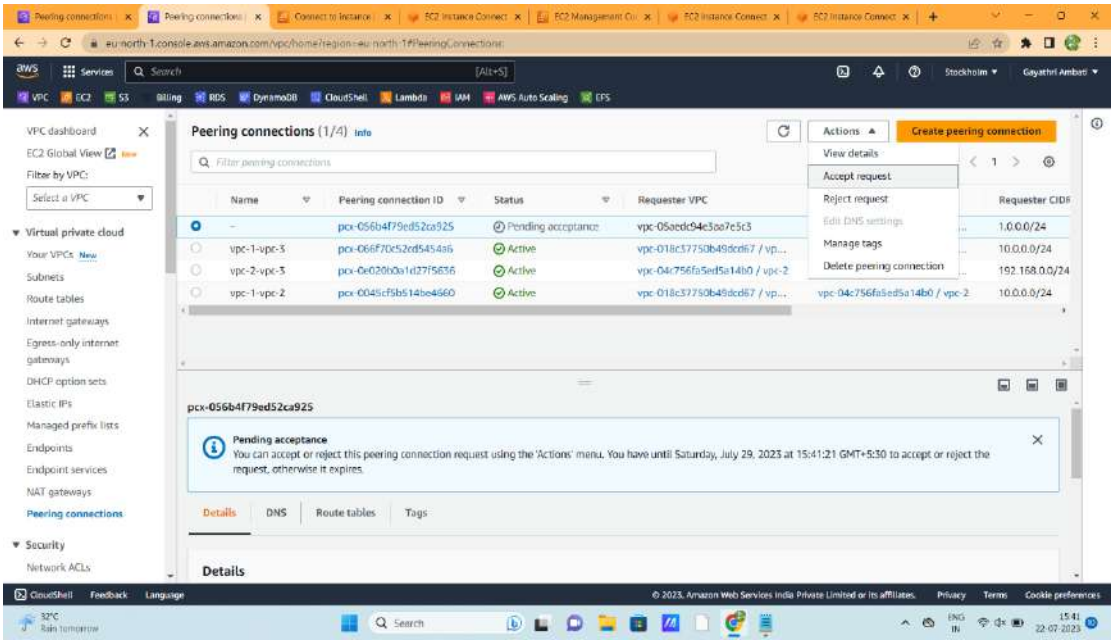
- For creating peering connection between two different accounts
- Consider one account as requester and another as accepter.
- For creating peering connection between vpc3 and vpc4
- Create peering connection in ap-south-1 belongs to account 049478797685.

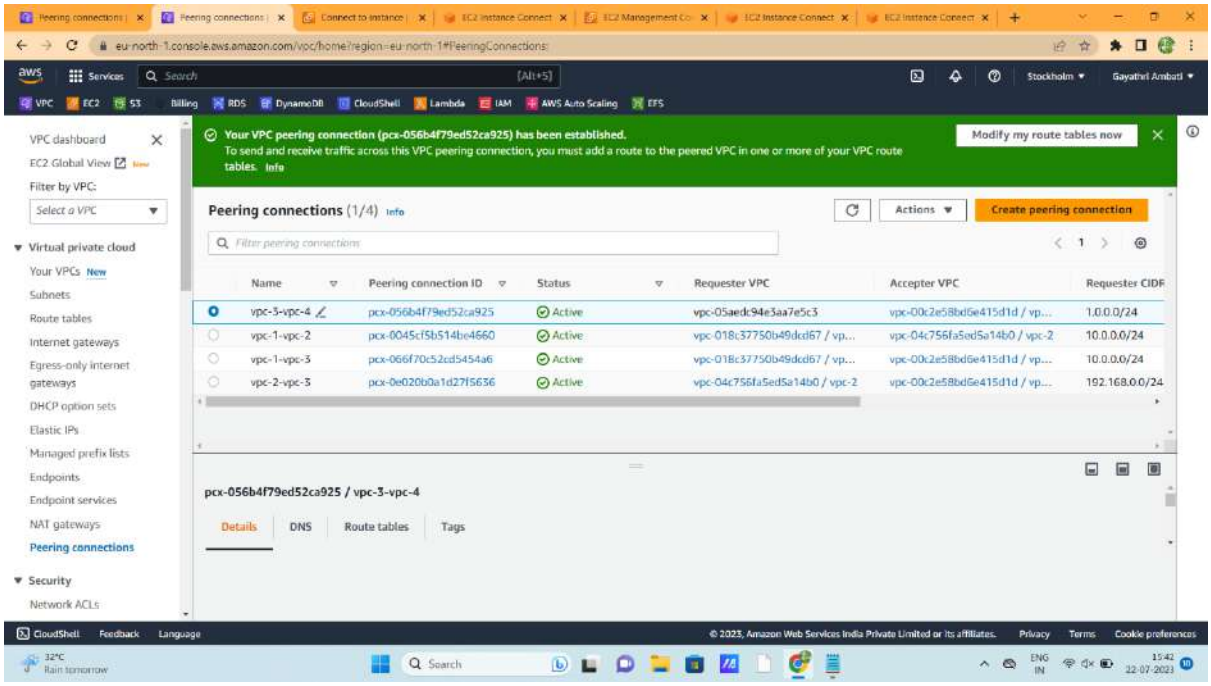
- Select another account option in vpc peering connection vpc peering
- Consider vpc4 as requeter and vpc3 as accepter and send request to vpc3



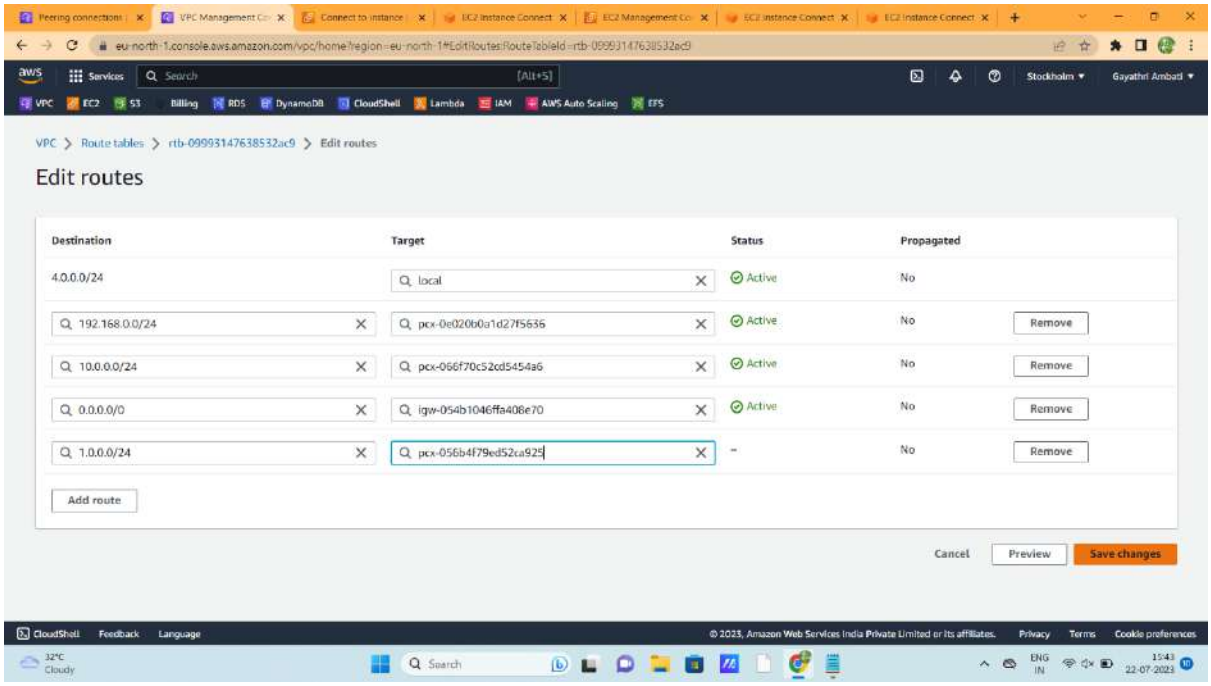
- In Account ID give 896571820628 and giveaccepter vpc ID and create peering connection

- Then request send to the another account .routes can be edited only after request is accepted.



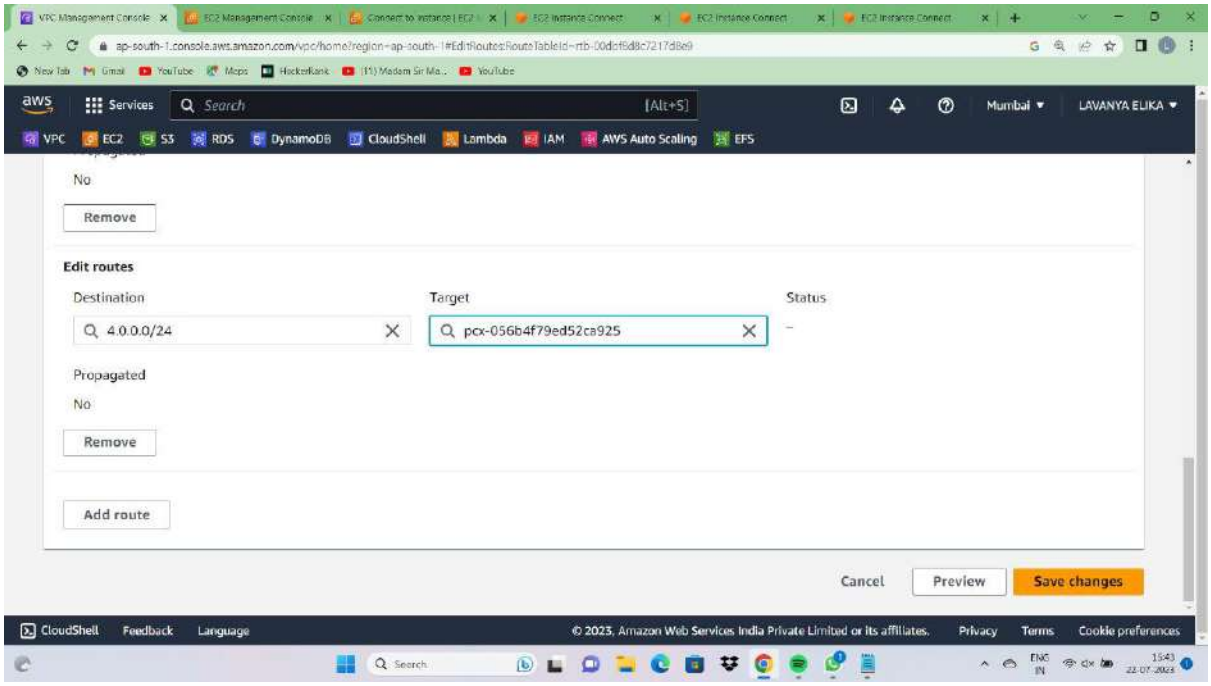


- After request is accepted in stockholm then edit the routes in the both the accounts.

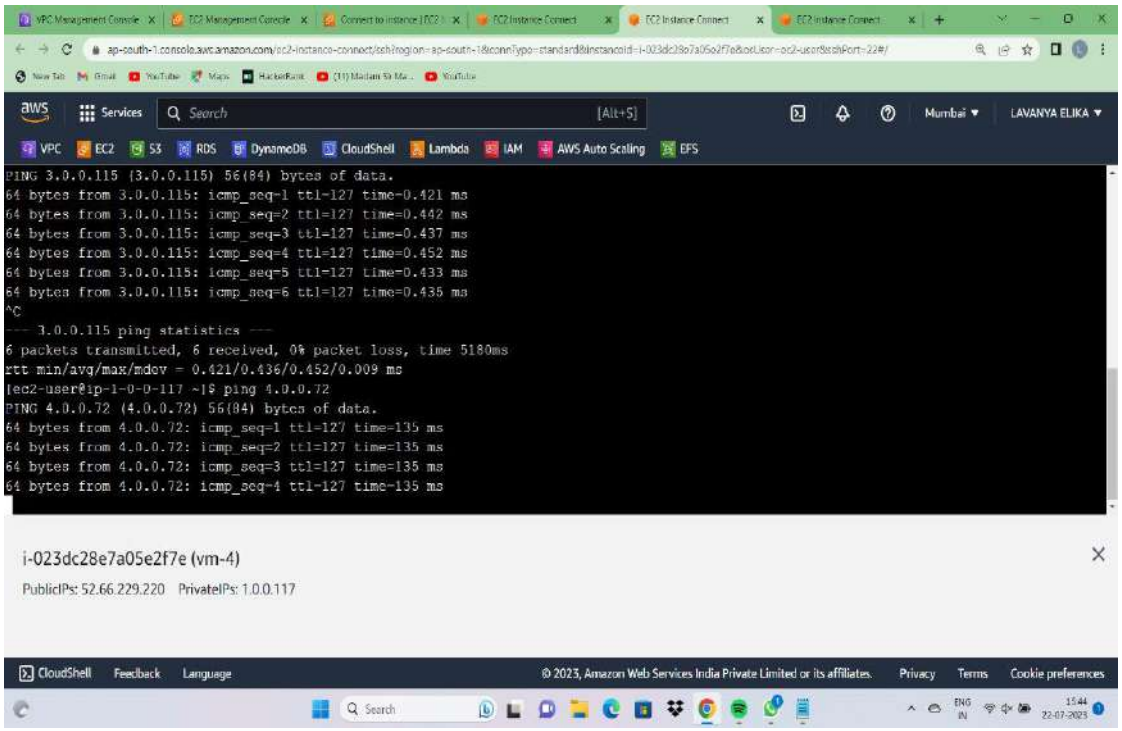


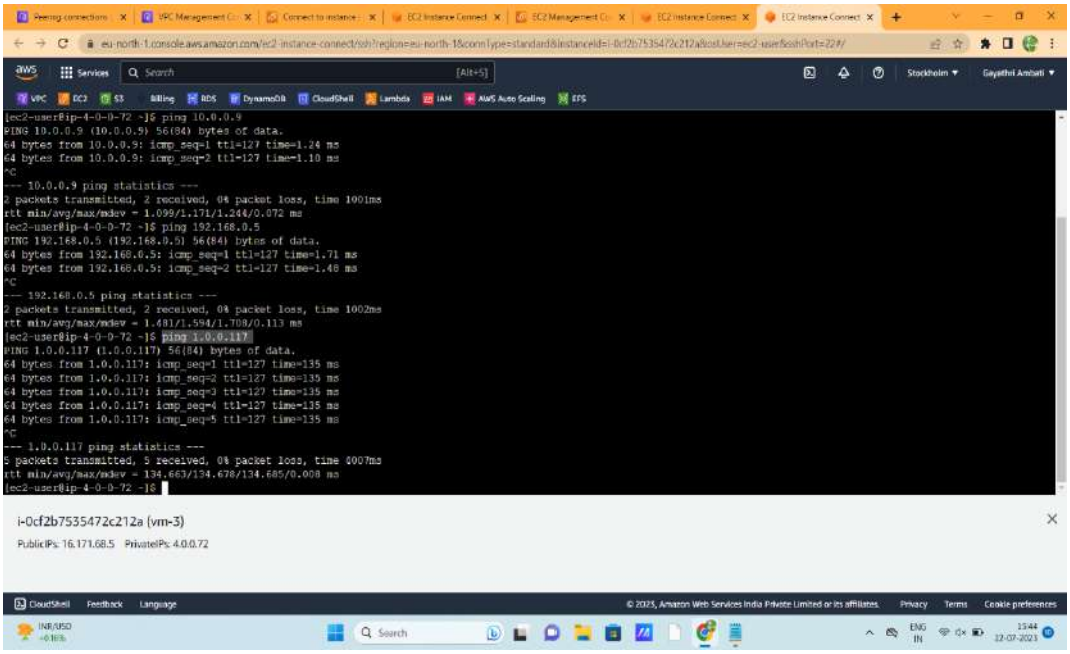
- in edit routes give CIDR of VPC-4 in RT-3 of VPC-3 and select target as peering connection as (VPC3-VPC4)

- in edit routes give CIDR of VPC-3 in RT-4 of VPC-4 and select target as peering connection as (VPC3-VPC4)

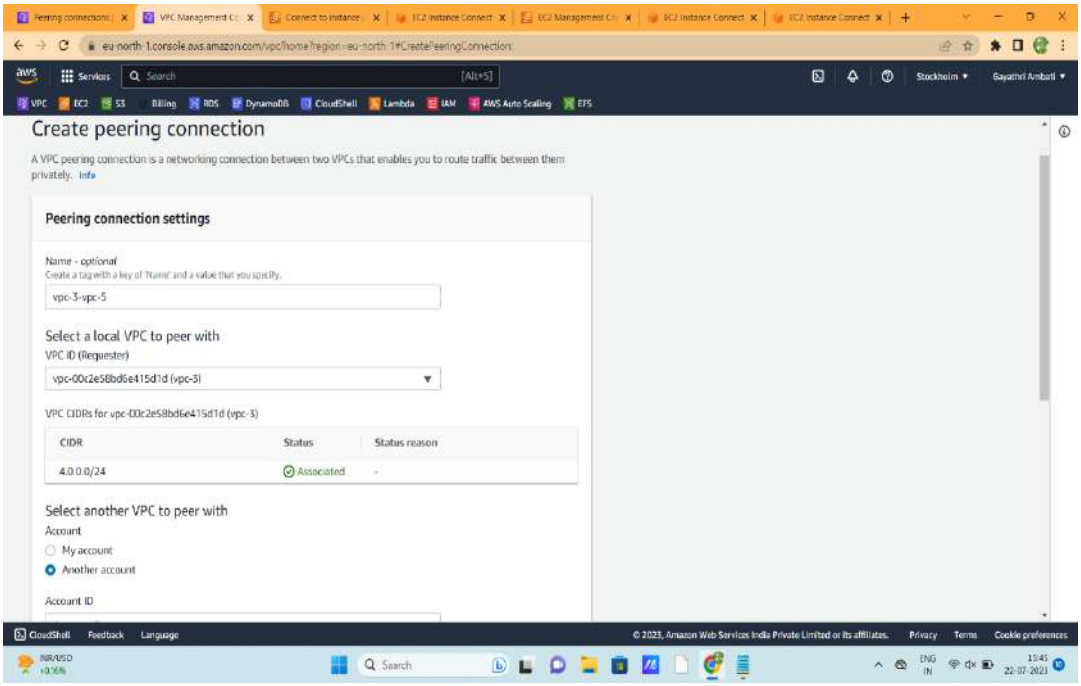


- after editing the routes now ping private ip address of VPC-3 belongs to stockholm into vm-4 in mumbai.
- now ping private ip address of VPC-4 belongs to mumbai into vm-3 in Stockholm.

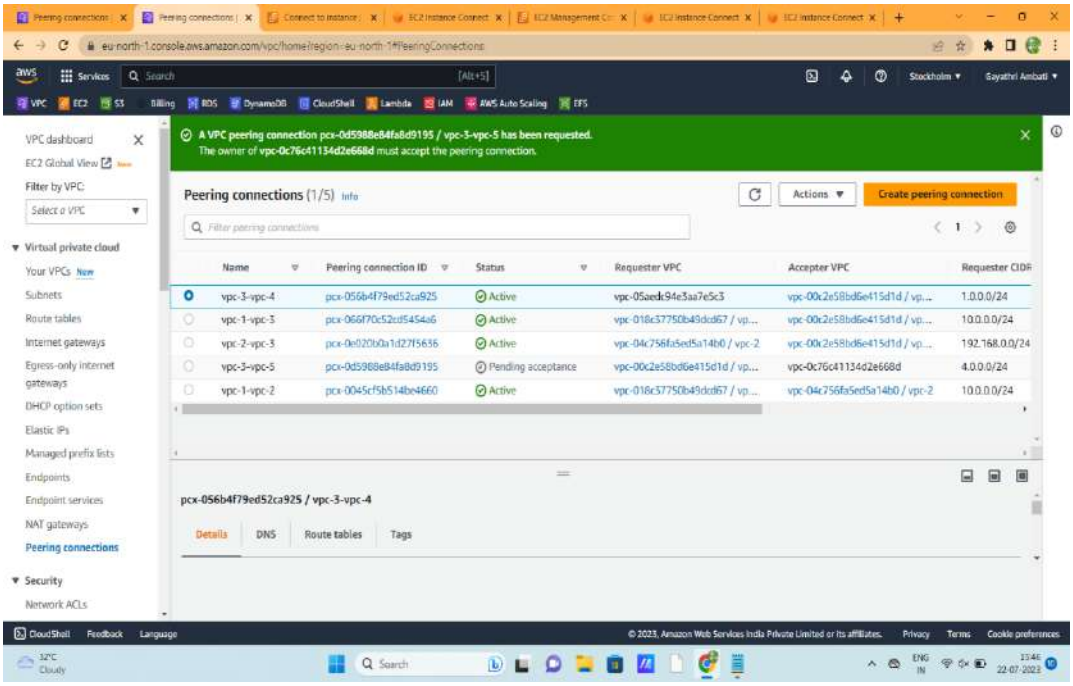




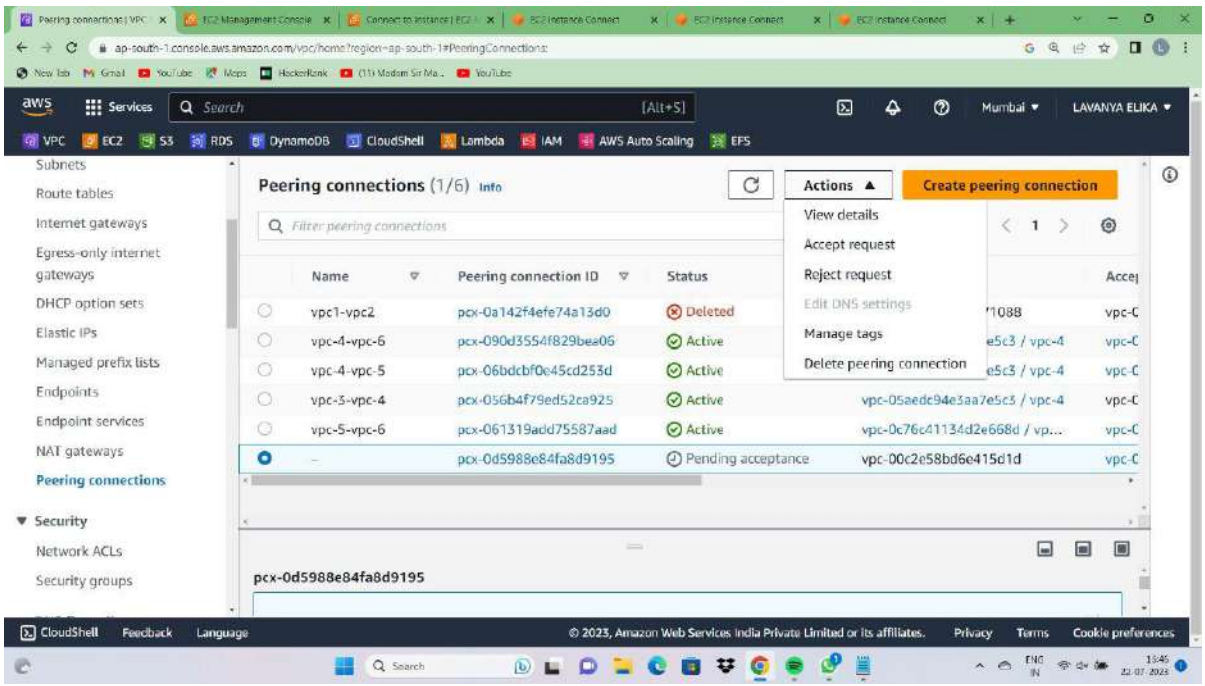
- For creating peering connection between vpc3 and vpc5 . Now consider Stockholm as requester and Mumbai as accepter.
- Create peering connection in eu-north-1 belongs to account 896571820628.
- Select another account option in vpc peering connection vpc peering
- Consider vpc3 as requeter and vpc5 as accepter and send request to vpc5

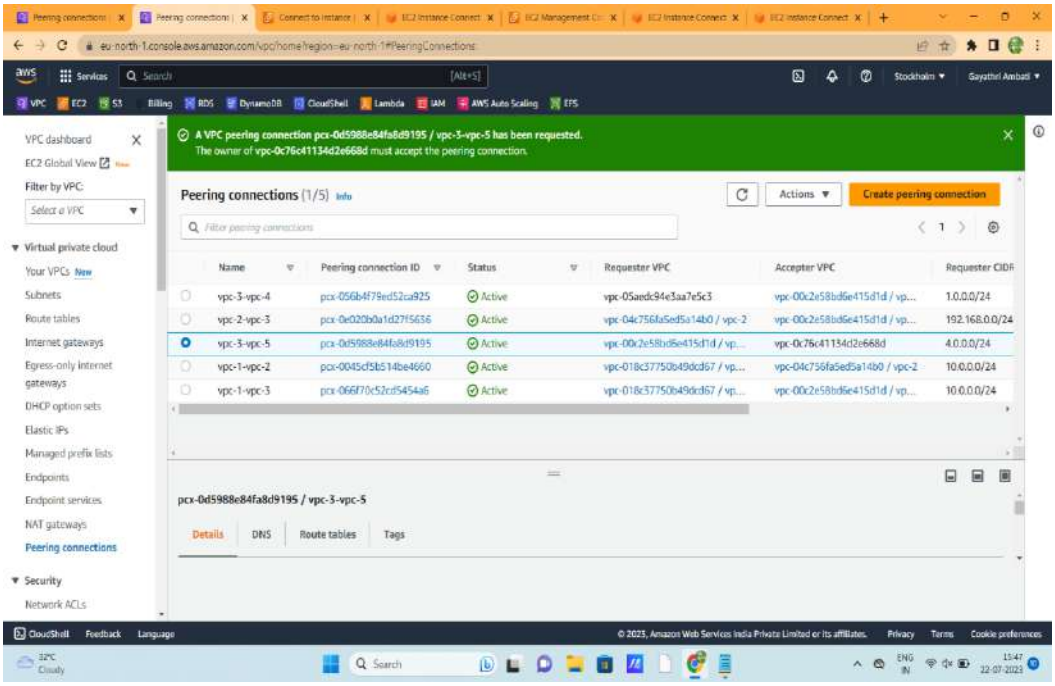


- In Account ID give 049478797685 and give accepter vpc ID of vpc5and create peering connection

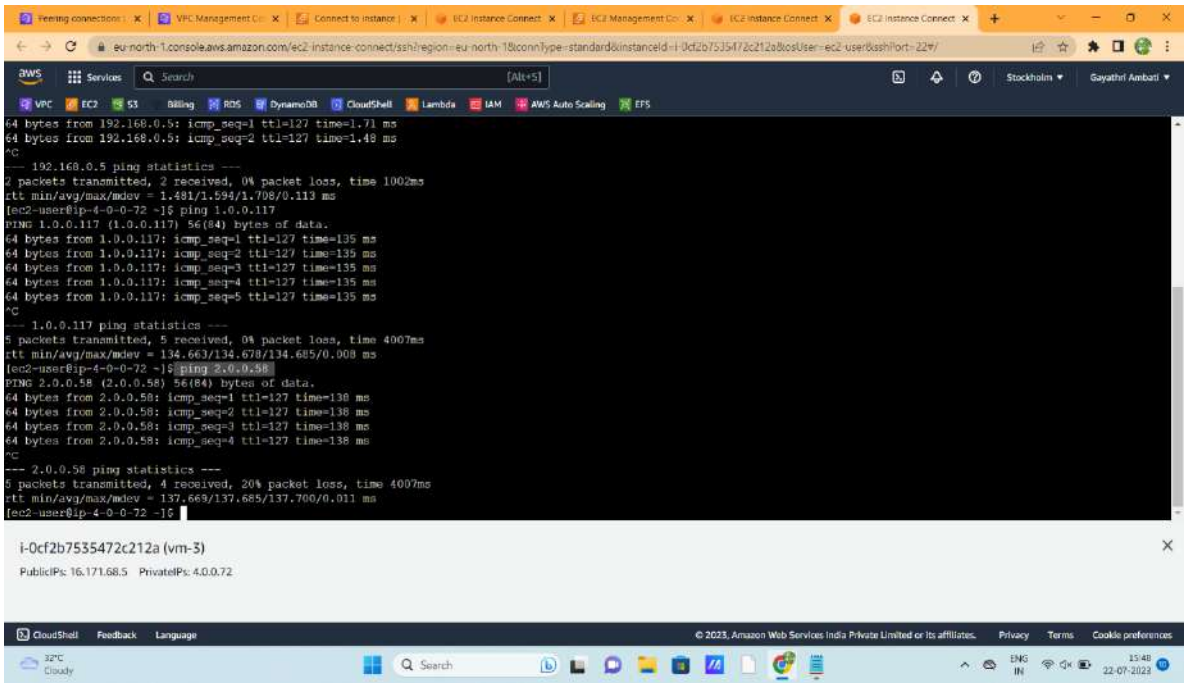


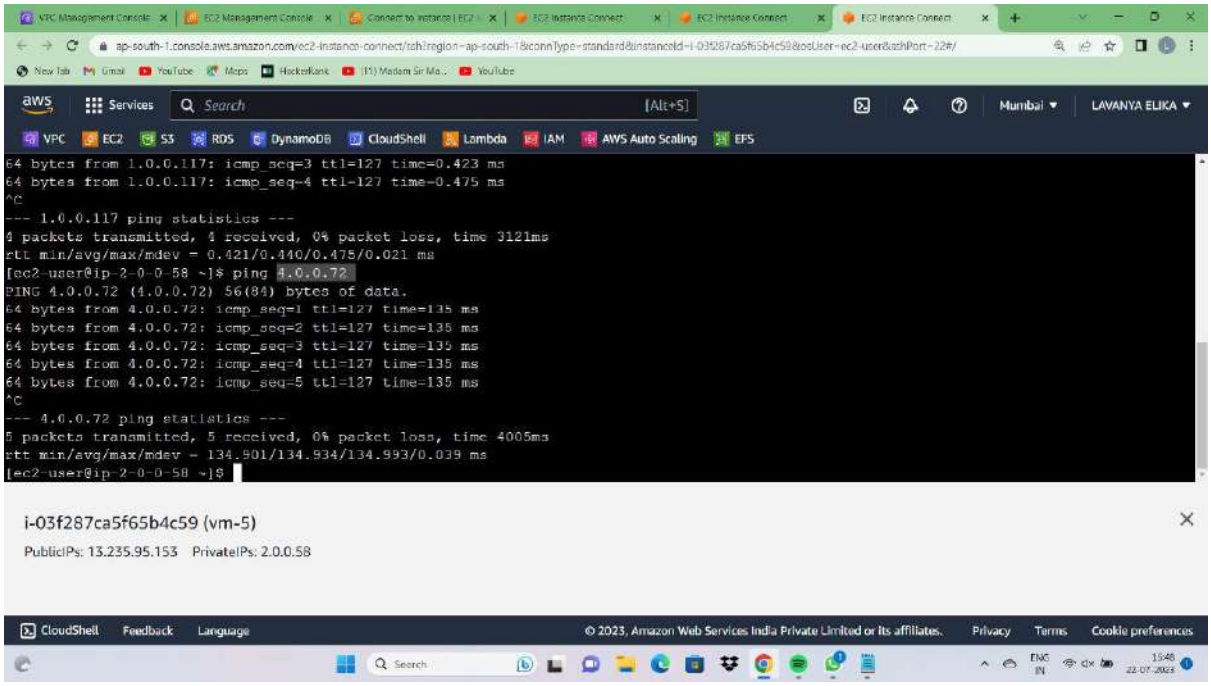
Then request send to the another account . peering connection remains in pending until accepter accepts the requests.we can edit routes only after request is accepted (peering connection status becomes active)



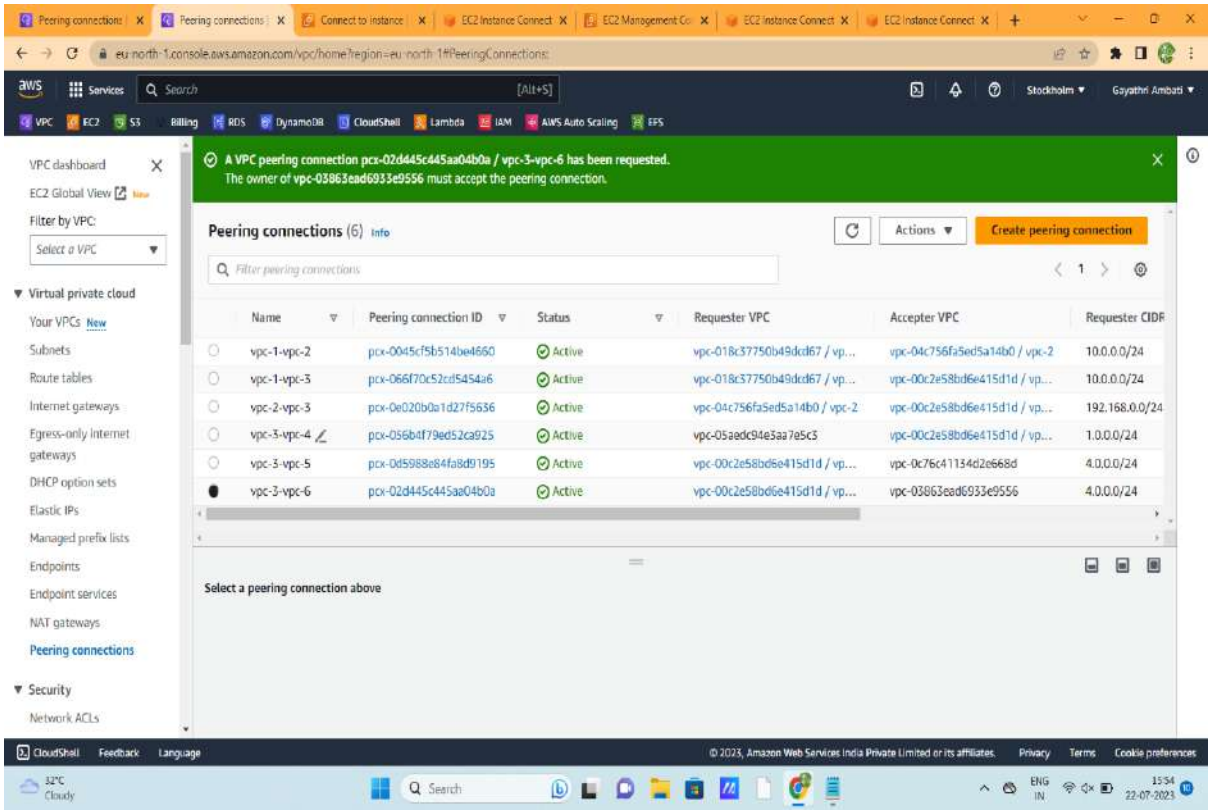


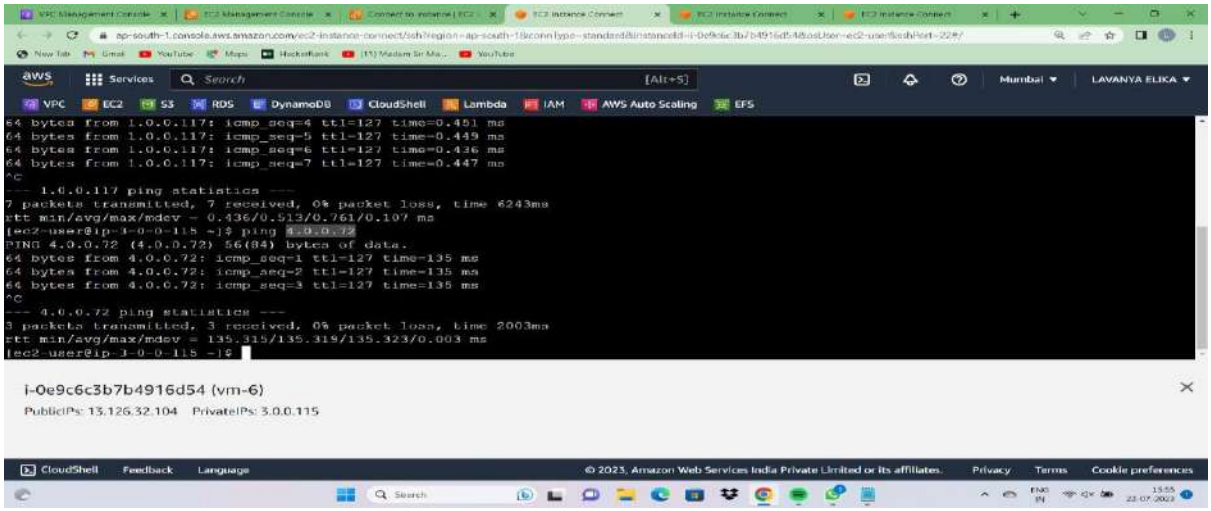
- Now peering connection is active for vpc3-vpc5, so now we can edit routes in for both vpc3 and vpc5 and give respective CIDRs in routes and save changes
- Now ping private address of VPC-3 belongs to stockholm into vm-5 in mumbai.
- now ping private ip address of VPC-5 belongs to mumbai into vm-3 in Stockholm

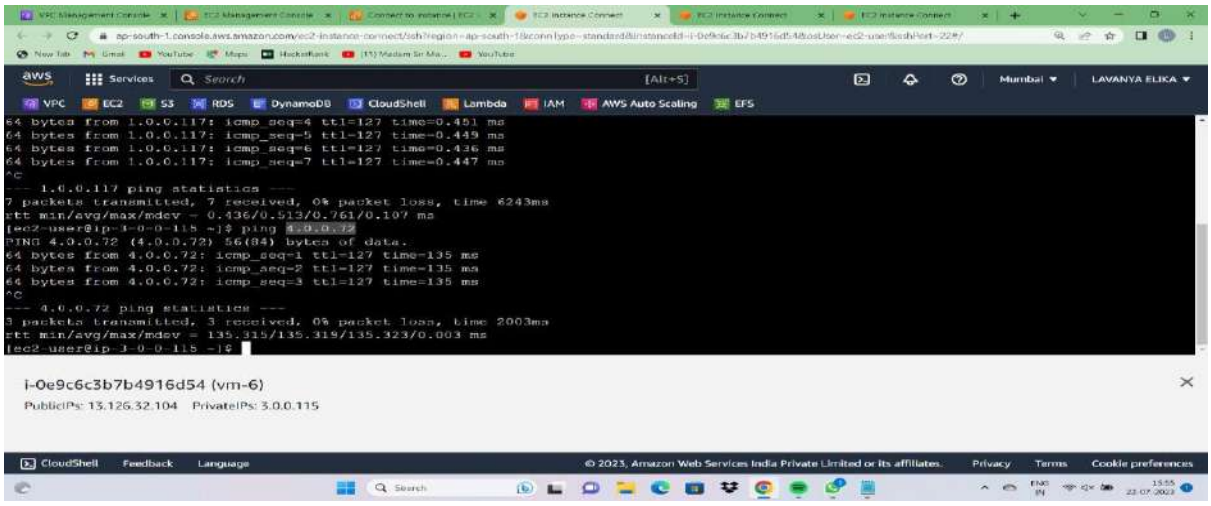




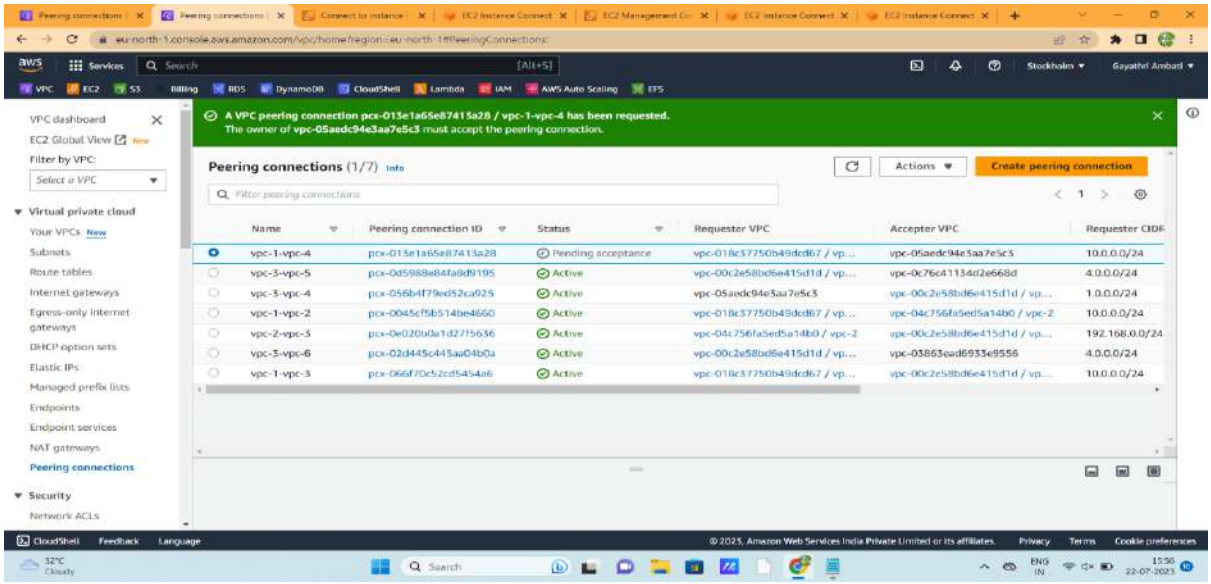
Create peering connection for vpc3-vpc6 considering vpc3 as requester and vpc-6 accepter.

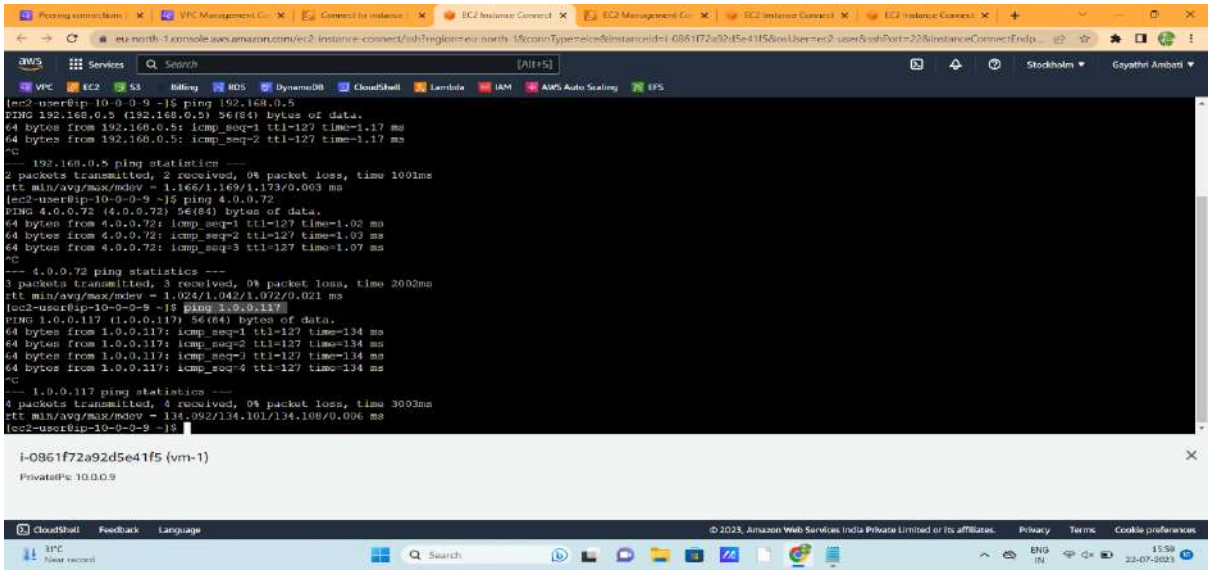


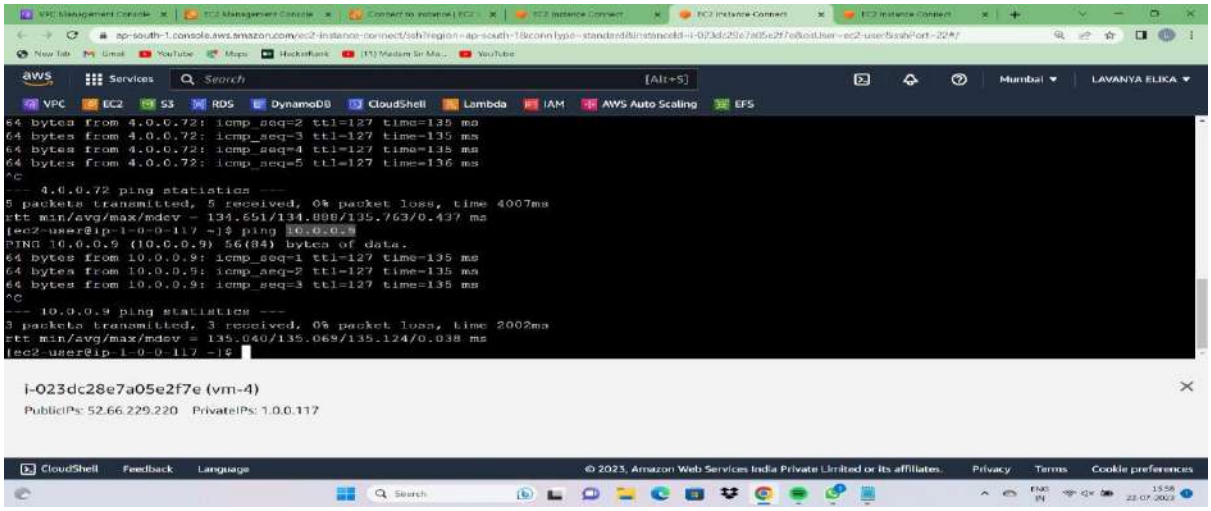




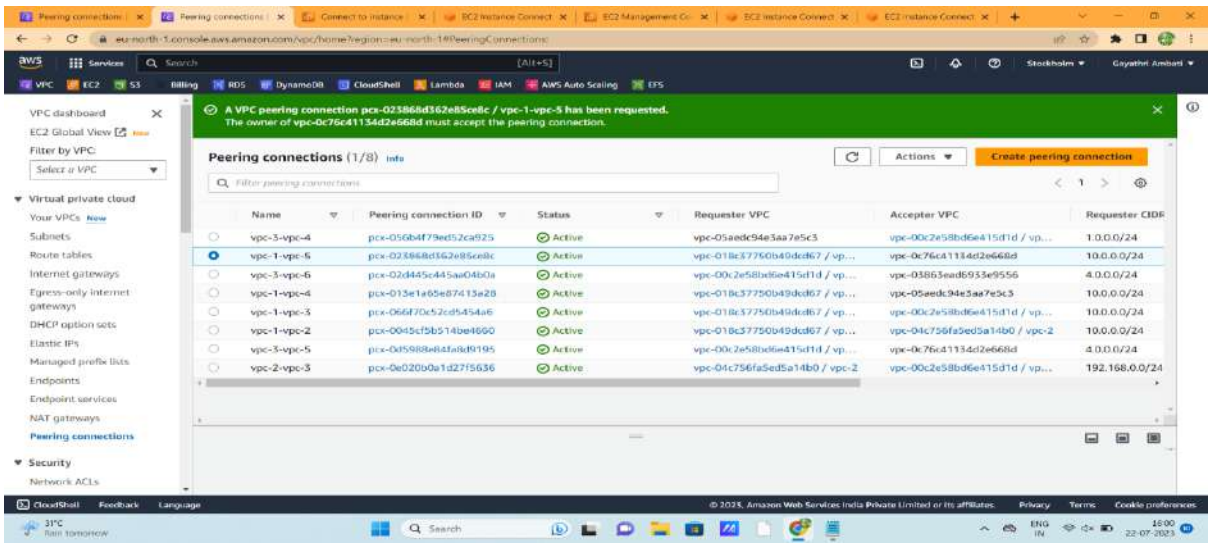
Create peering connection for vpc1-vpc4 considering vpc1 as requester and vpc-4 accepter.

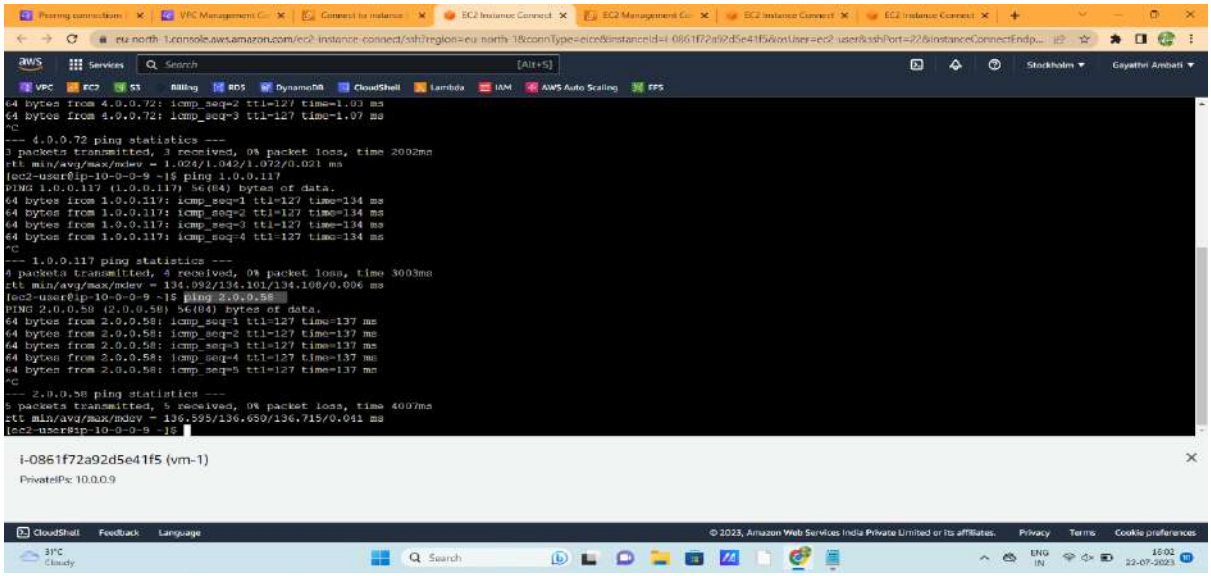


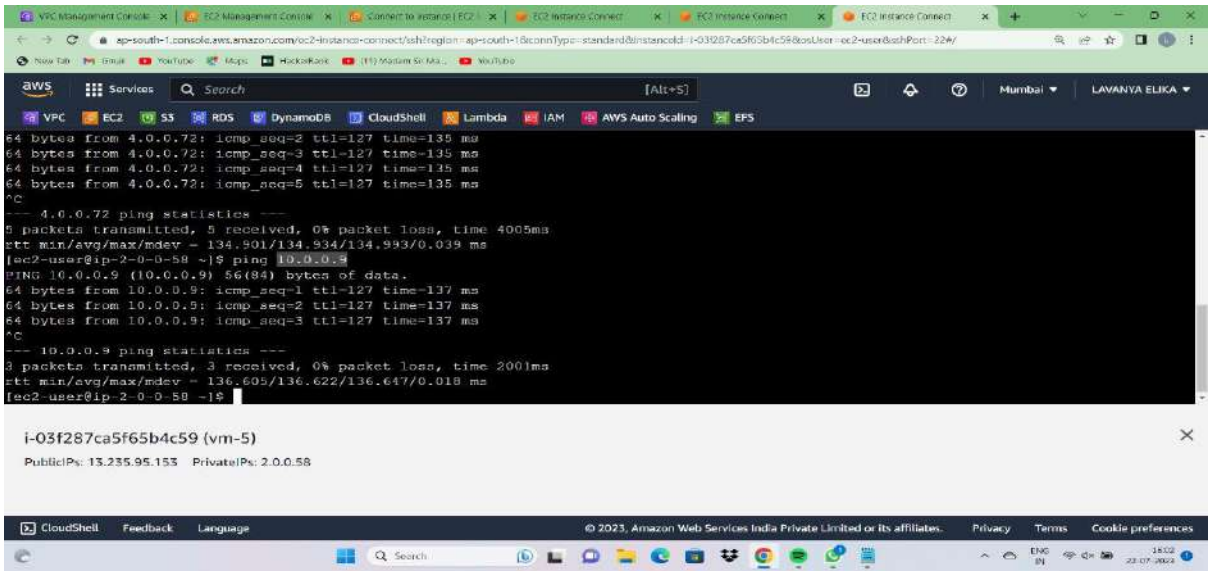




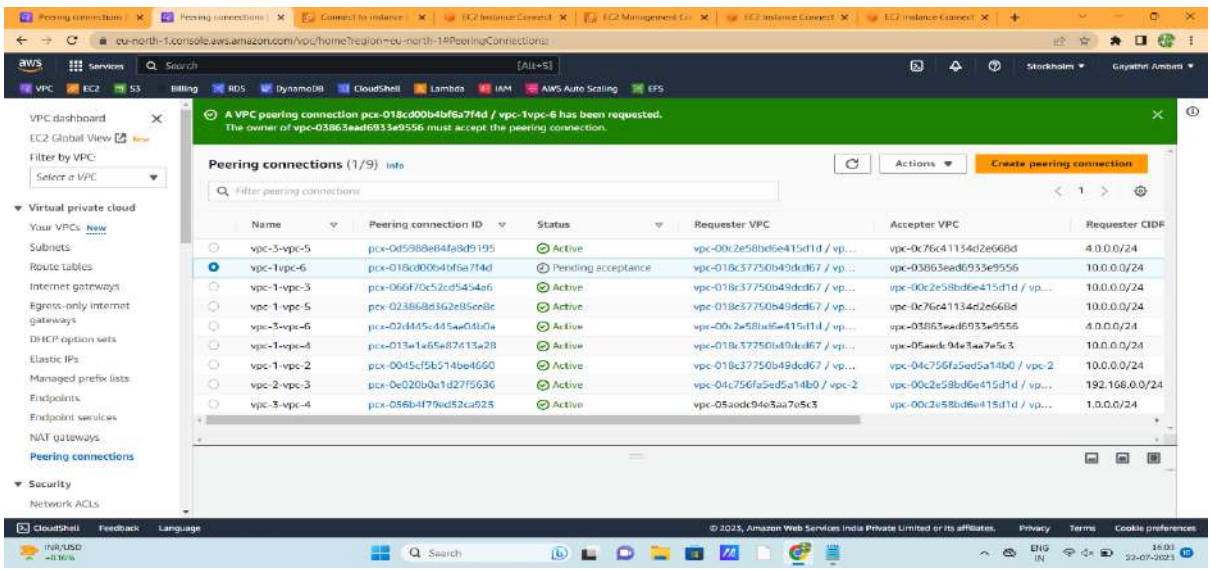
Create peering connection for vpc1-vpc5 considering vpc1 as requester and vpc-5accepter.

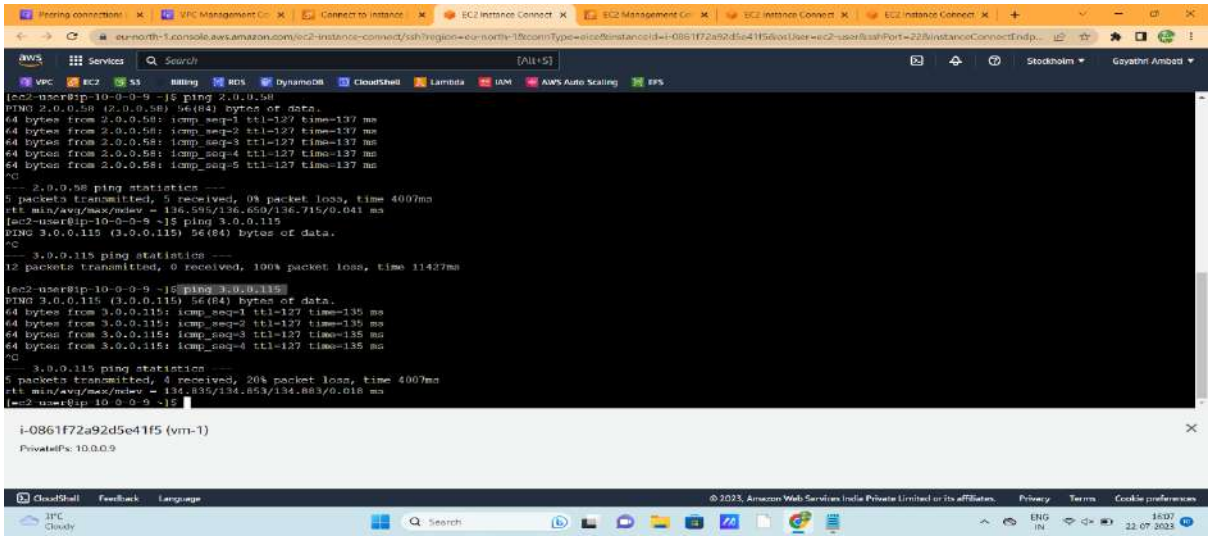


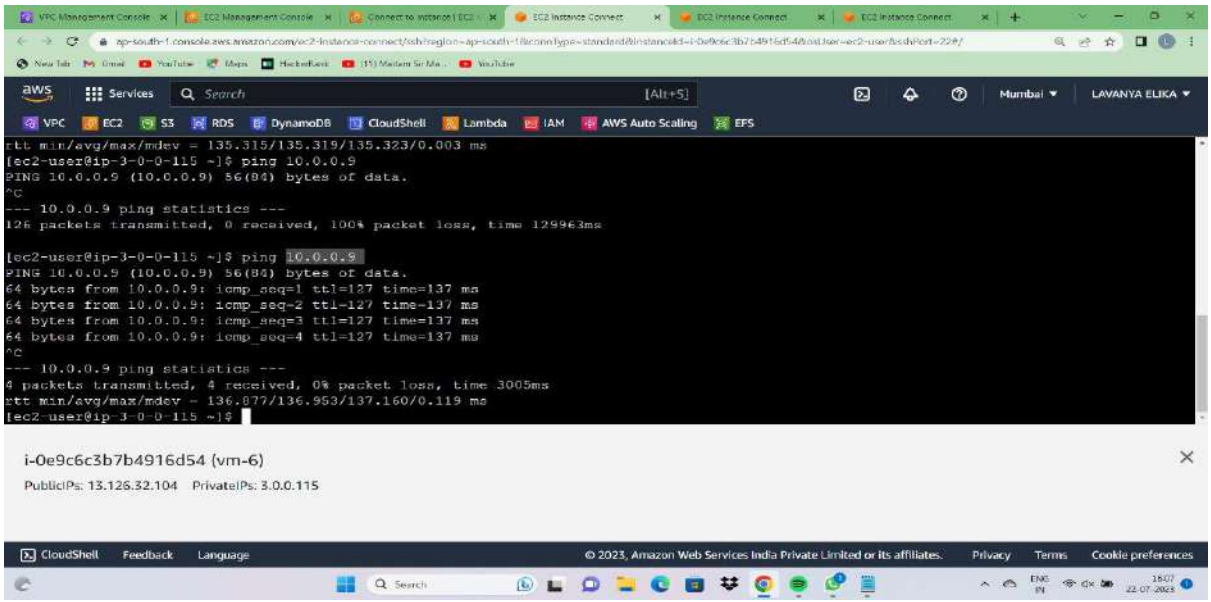




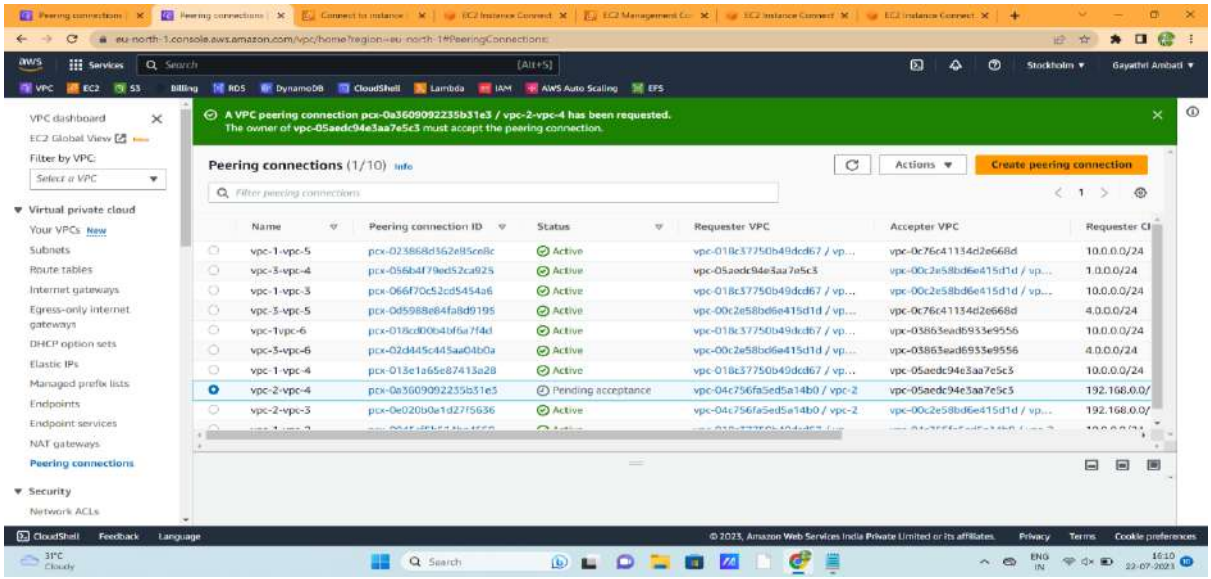
Create peering connection for vpc1-vpc6 considering vpc1 as requester and vpc-6 accepter.

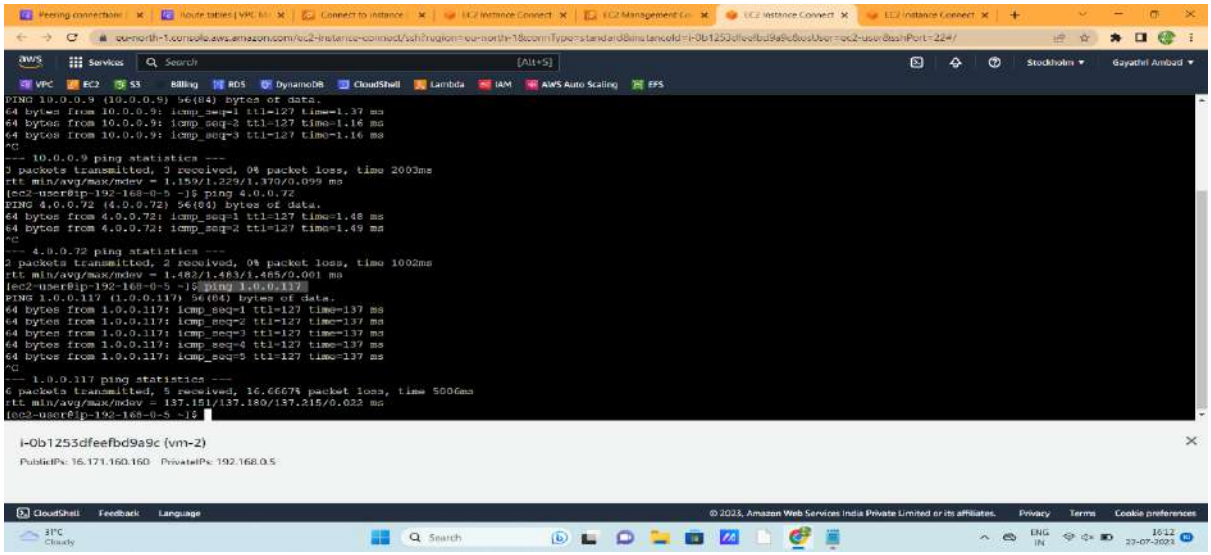


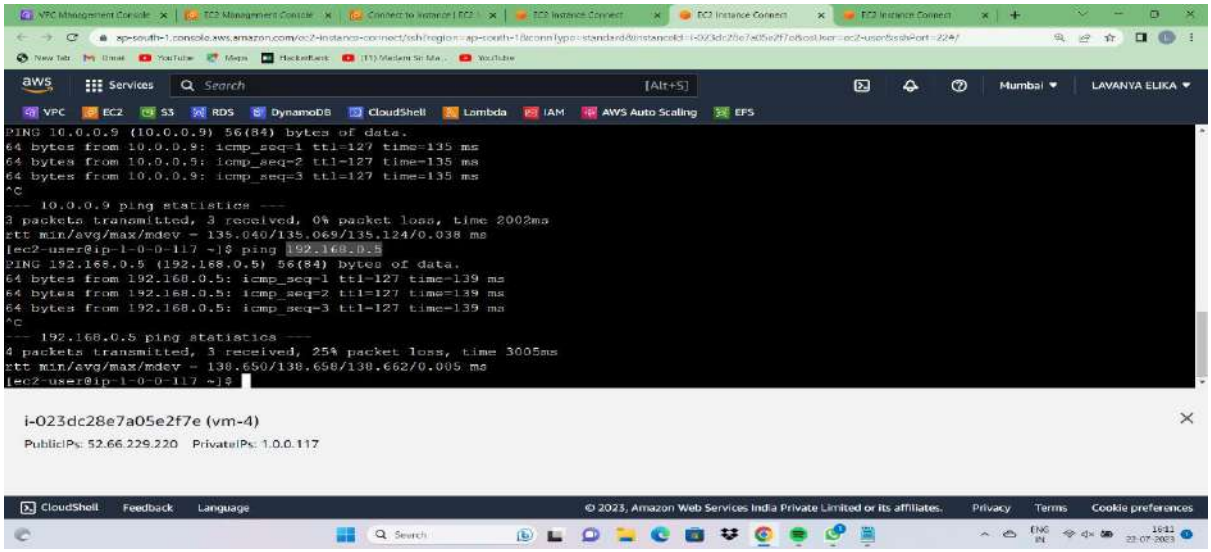




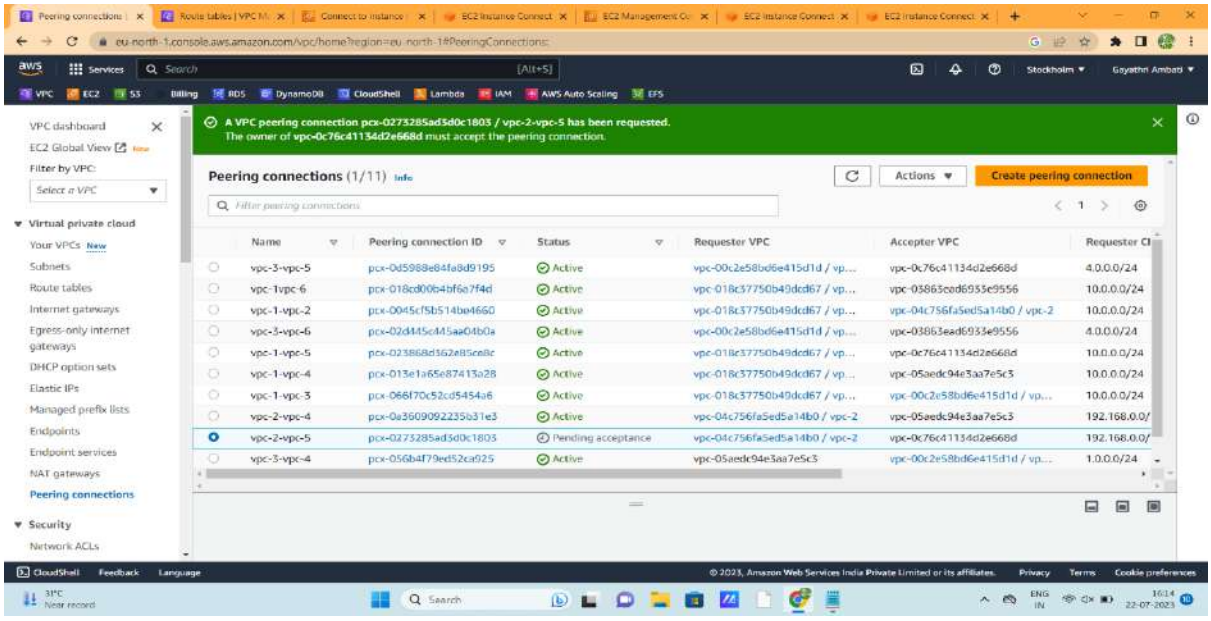
Create peering connection for vpc2-vpc4 considering vpc2 as requester and vpc-4 accepter.

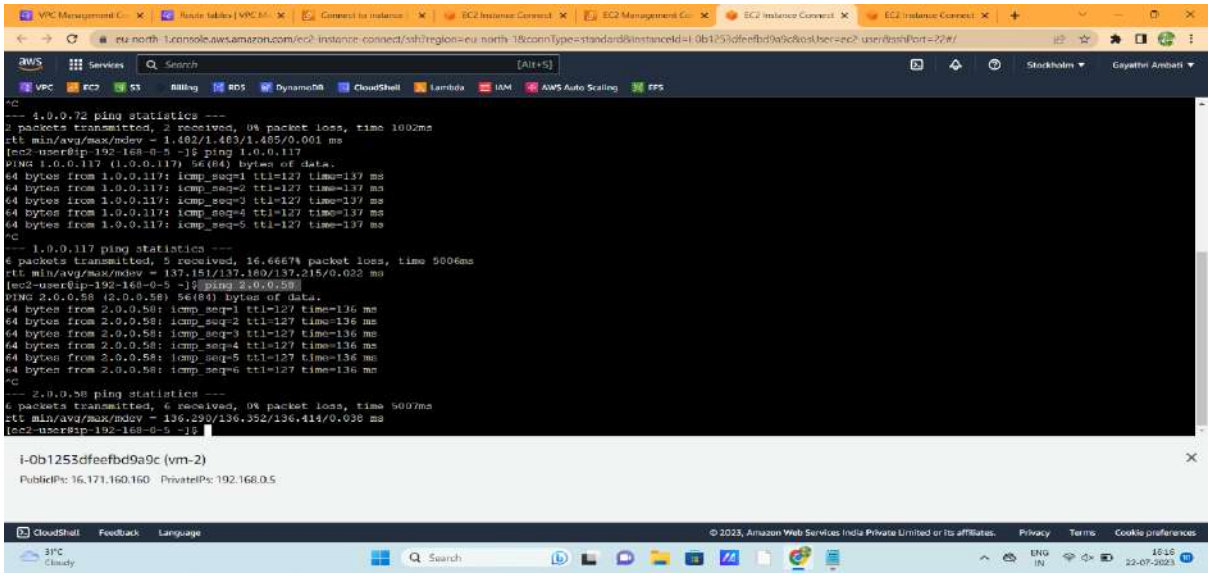


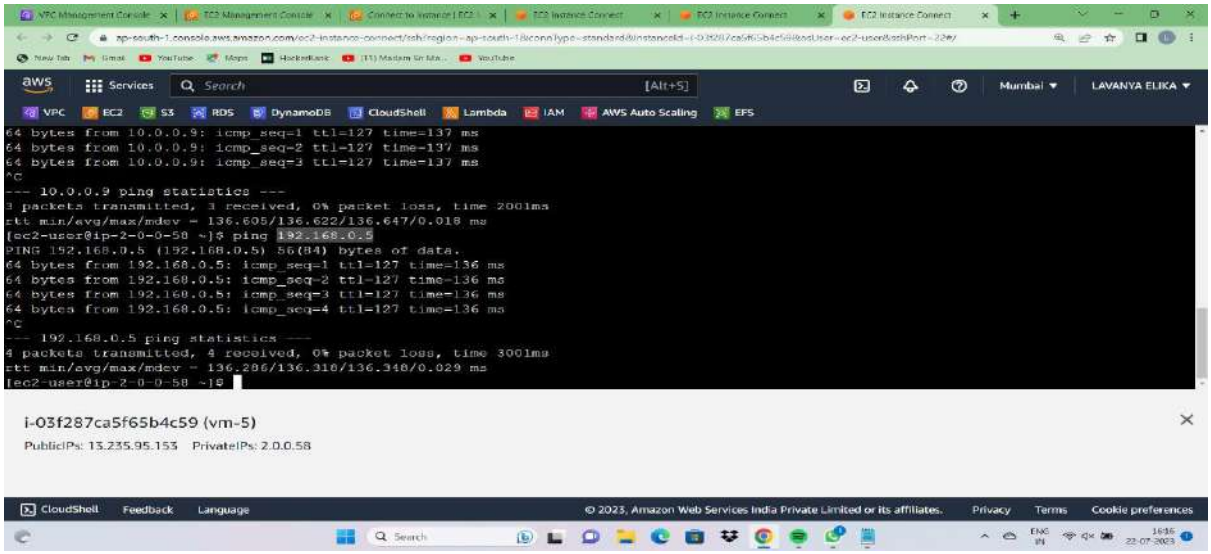




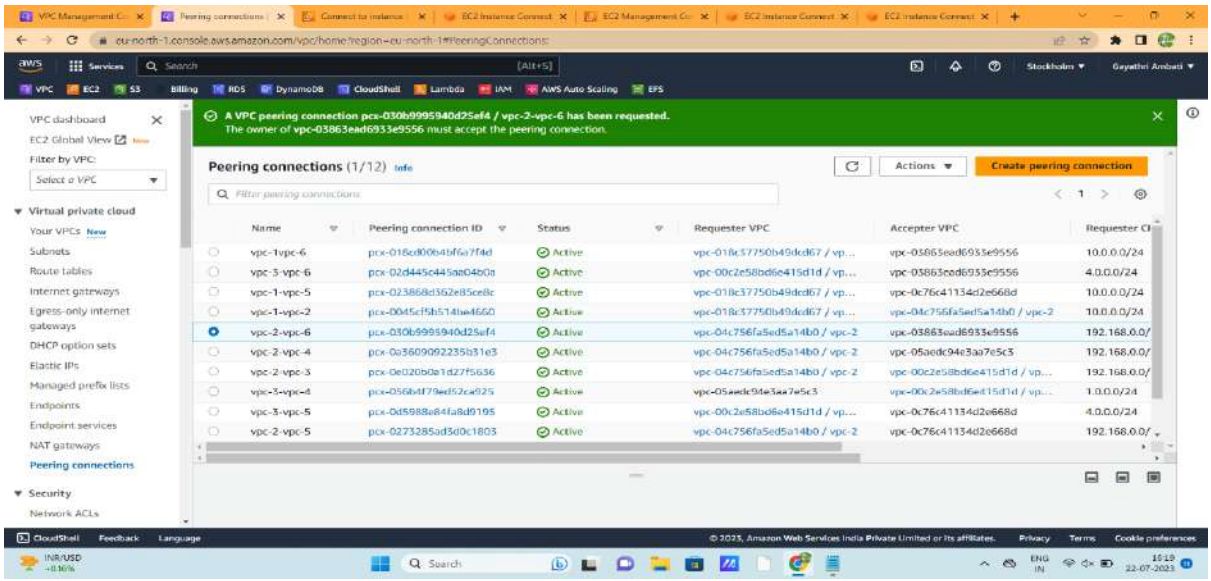
Create peering connection for vpc2-vpc5 considering vpc2 as requester and vpc-5 accepter.

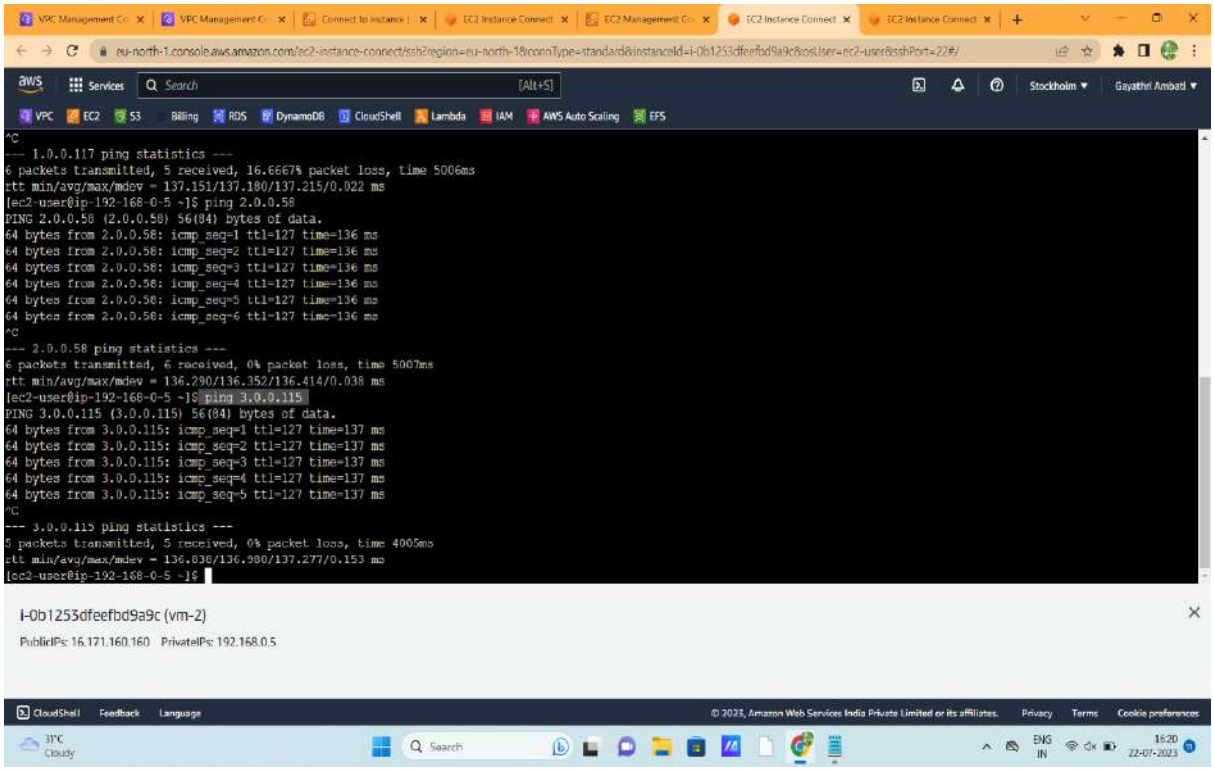


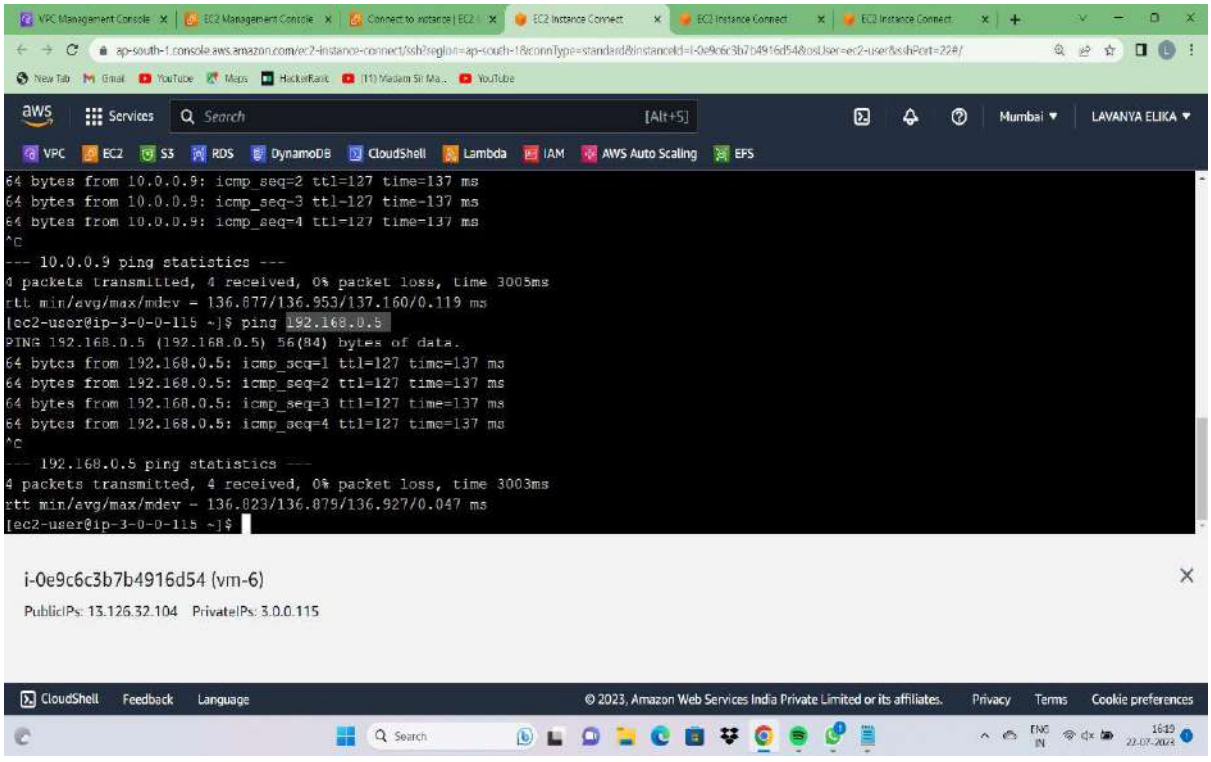




Create peering connection for vpc2-vpc6 considering vpc2 as requester and vpc-6 accepter.

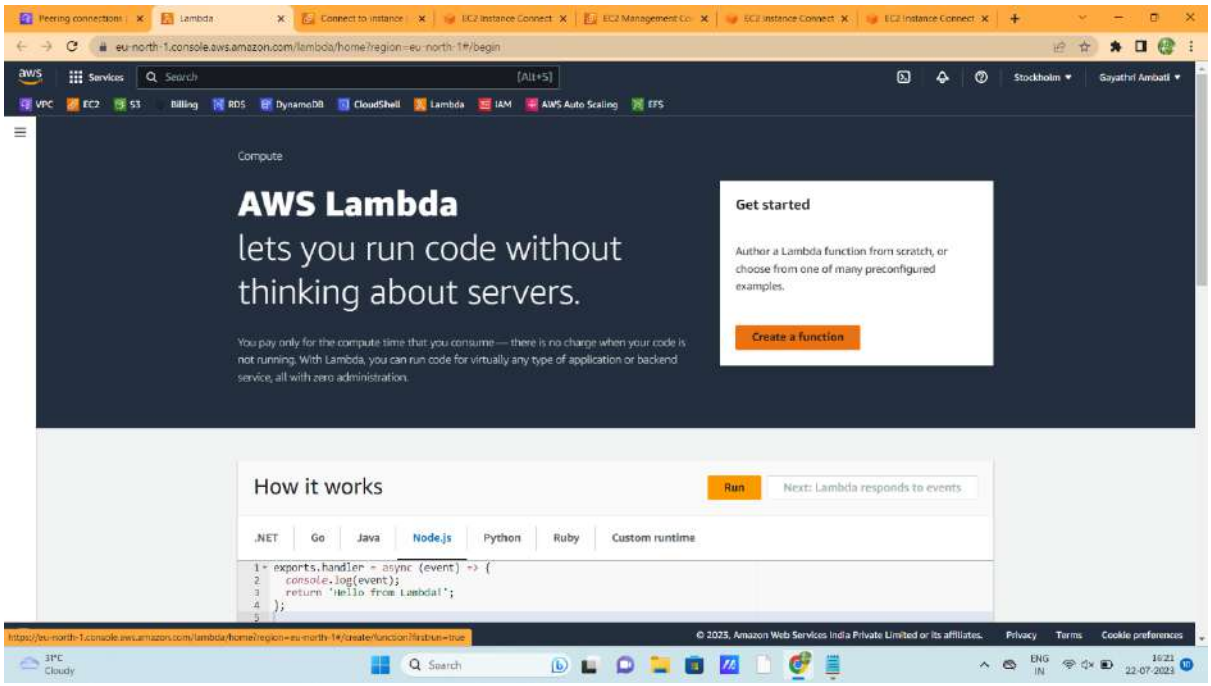


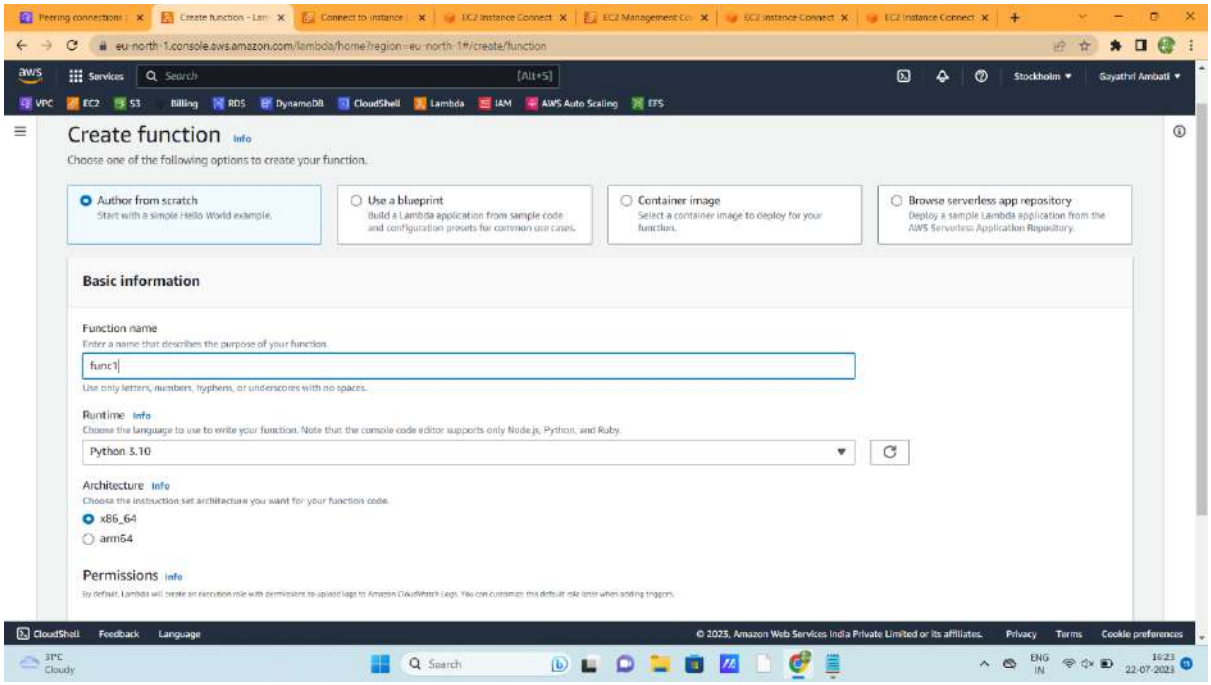

^C
--- 1.0.0.117 ping statistics ---
6 packets transmitted, 5 received, 16.6667% packet loss, time 5006ms
rtt min/avg/max/mdev = 137.151/137.180/137.215/0.022 ms
[ec2-user@ip-192-168-0-5 ~]$ ping 2.0.0.58
PING 2.0.0.58 (2.0.0.58) 56(84) bytes of data.
64 bytes from 2.0.0.58: icmp_seq=1 ttl=127 time=136 ms
64 bytes from 2.0.0.58: icmp_seq=2 ttl=127 time=136 ms
64 bytes from 2.0.0.58: icmp_seq=3 ttl=127 time=136 ms
64 bytes from 2.0.0.58: icmp_seq=4 ttl=127 time=136 ms
64 bytes from 2.0.0.58: icmp_seq=5 ttl=127 time=136 ms
64 bytes from 2.0.0.58: icmp_seq=6 ttl=127 time=136 ms
^C
--- 2.0.0.58 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5007ms
rtt min/avg/max/mdev = 136.290/136.352/136.414/0.038 ms
[ec2-user@ip-192-168-0-5 ~]$ ping 3.0.0.115
PING 3.0.0.115 (3.0.0.115) 56(84) bytes of data.
64 bytes from 3.0.0.115: icmp_seq=1 ttl=127 time=137 ms
64 bytes from 3.0.0.115: icmp_seq=2 ttl=127 time=137 ms
64 bytes from 3.0.0.115: icmp_seq=3 ttl=127 time=137 ms
64 bytes from 3.0.0.115: icmp_seq=4 ttl=127 time=137 ms
64 bytes from 3.0.0.115: icmp_seq=5 ttl=127 time=137 ms
^C
--- 3.0.0.115 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4005ms
rtt min/avg/max/mdev = 136.836/136.980/137.277/0.153 ms
[ec2-user@ip-192-168-0-5 ~]$
i-0b1253dfeefbd9a9c (vm-2)
PublicIPs: 16.171.160.160 PrivateIPs: 192.168.0.5


64 bytes from 10.0.0.9: icmp_seq=2 ttl=127 time=137 ms
64 bytes from 10.0.0.9: icmp_seq=3 ttl=127 time=137 ms
64 bytes from 10.0.0.9: icmp_seq=4 ttl=127 time=137 ms
^C
--- 10.0.0.9 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 136.877/136.953/137.160/0.119 ms
[ec2-user@ip-3-0-0-115 ~]$ ping 192.168.0.5
PING 192.168.0.5 (192.168.0.5) 56(84) bytes of data.
64 bytes from 192.168.0.5: icmp_seq=1 ttl=127 time=137 ms
64 bytes from 192.168.0.5: icmp_seq=2 ttl=127 time=137 ms
64 bytes from 192.168.0.5: icmp_seq=3 ttl=127 time=137 ms
64 bytes from 192.168.0.5: icmp_seq=4 ttl=127 time=137 ms
^C
--- 192.168.0.5 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3003ms
rtt min/avg/max/mdev = 136.923/136.979/136.927/0.047 ms
[ec2-user@ip-3-0-0-115 ~]$
i-0e9c6c3b7b4916d54 (vm-6)
PublicIPs: 13.126.32.104 PrivateIPs: 3.0.0.115

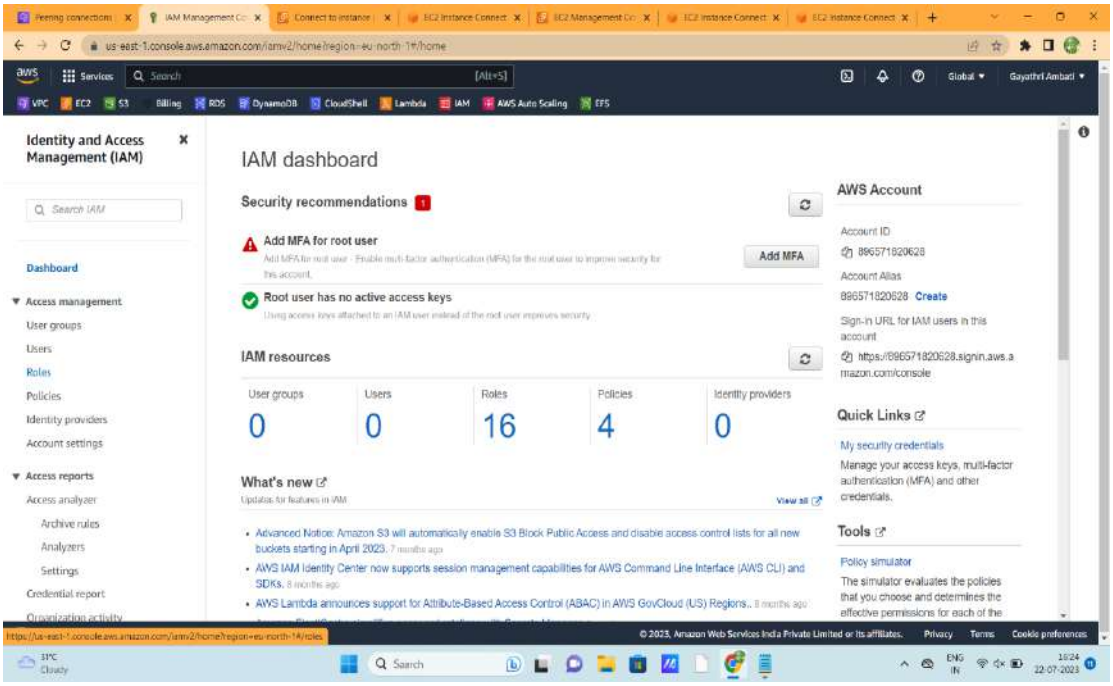
TO TERMINATE THE INSTANCES USING LAMBDA FUNCTION

- open the lambda service and create a function.

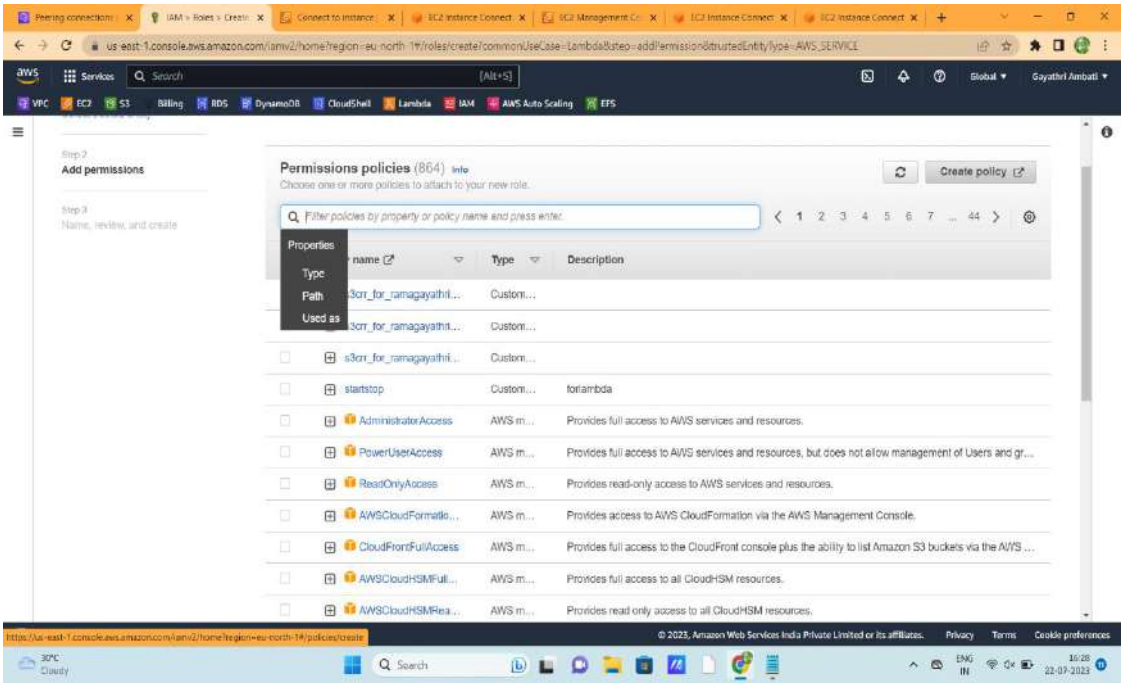


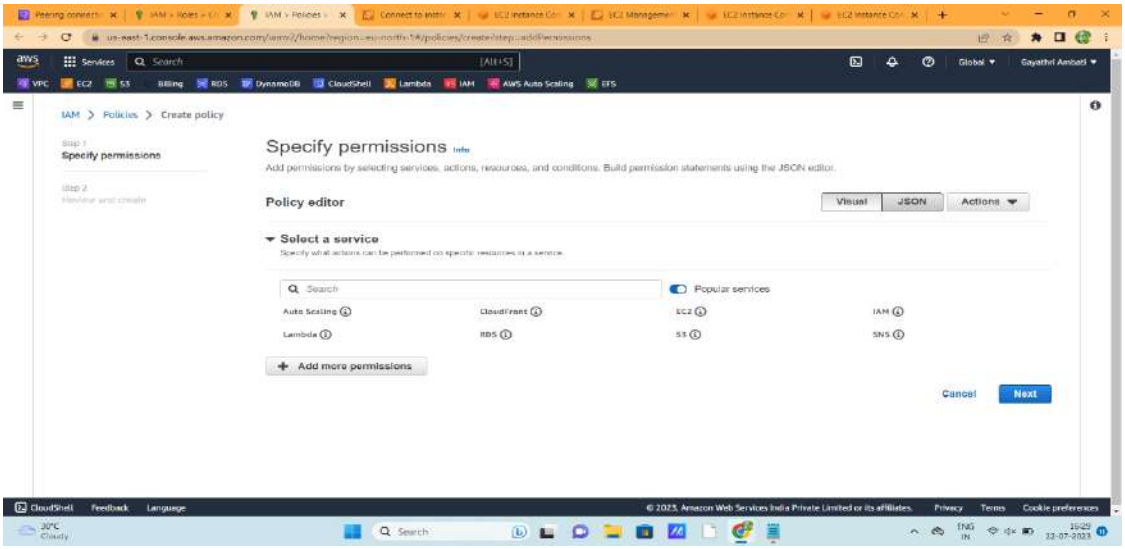


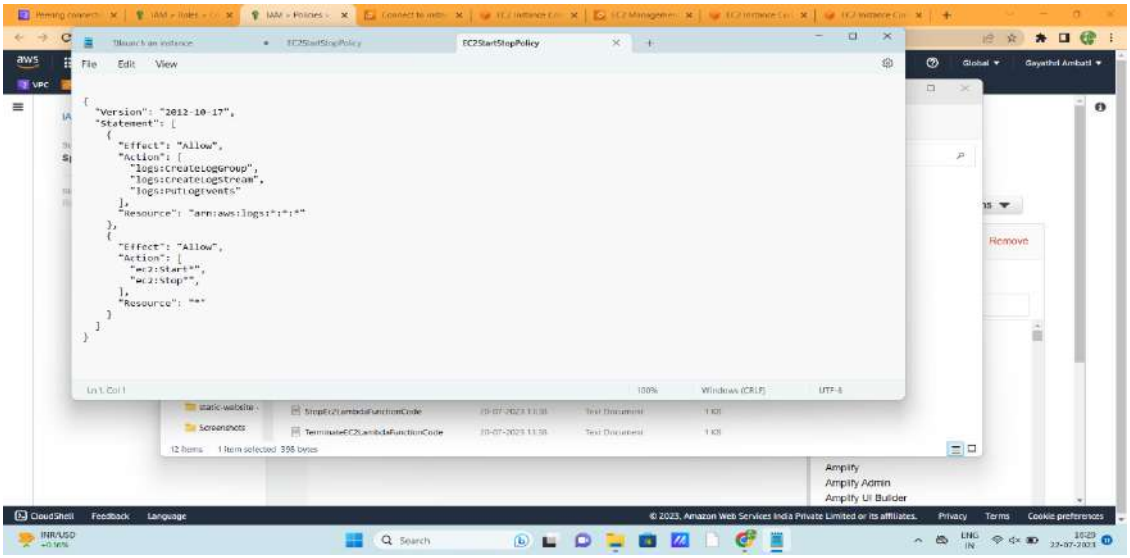
- Now go to IAM role and create a role

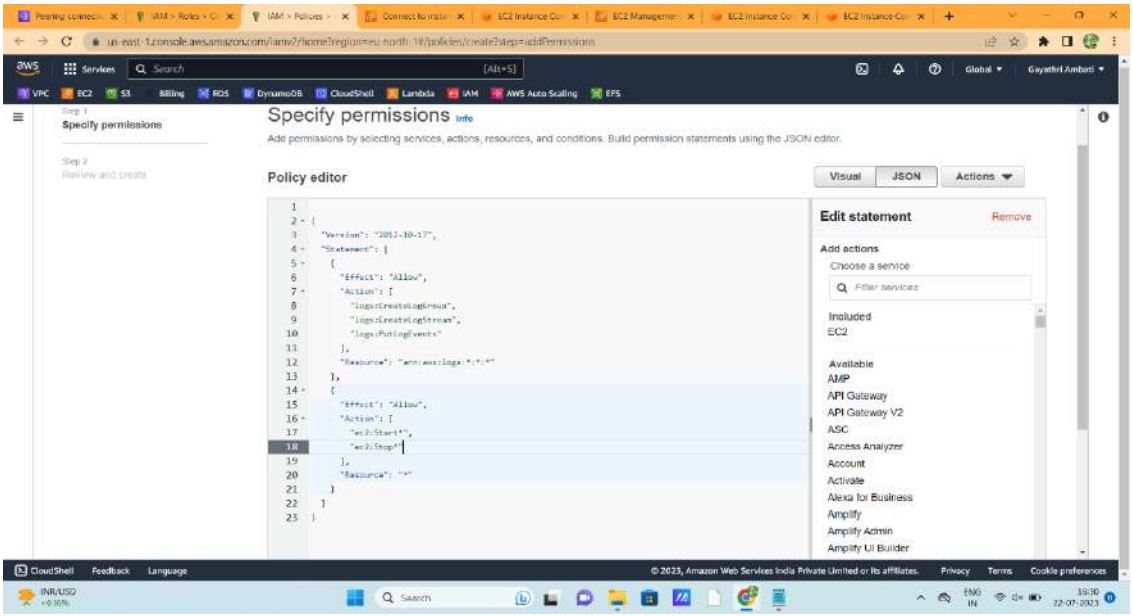


- Specify permissions and select JSON
- Edit the code with EC2 start and stop policy code.
- Edit errors and click next.
- And create policy.

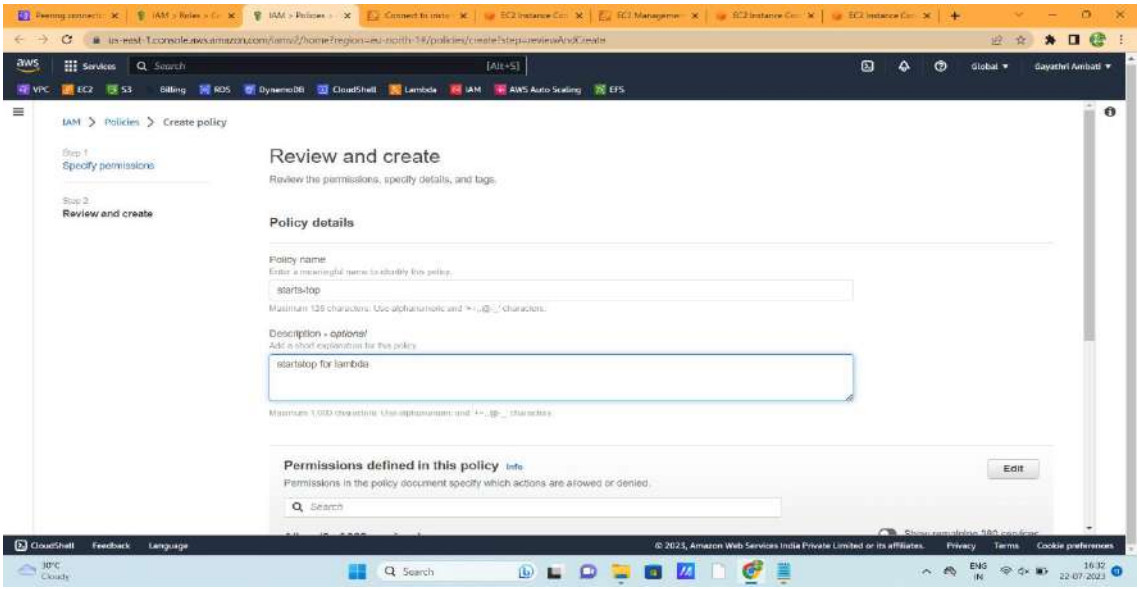


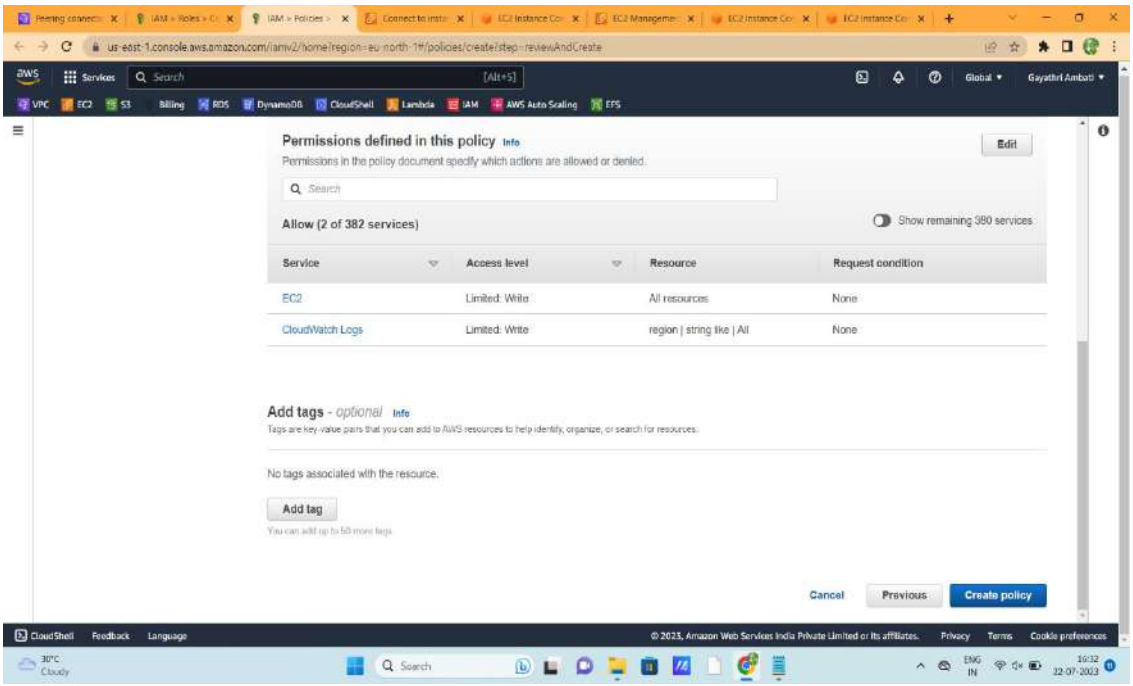

IAM > Policies > Create policy
Specify permissions
Add permissions by selecting services, actions, resources, and conditions. Build permission statements using the JSON editor.
Policy editor
Visual
JSON
Actions
Select a service
Search
Popular services
Auto Scaling
CloudFront
EC2
IAM
Lambda
RDS
S3
SNS
Add more permissions
Cancel
Next

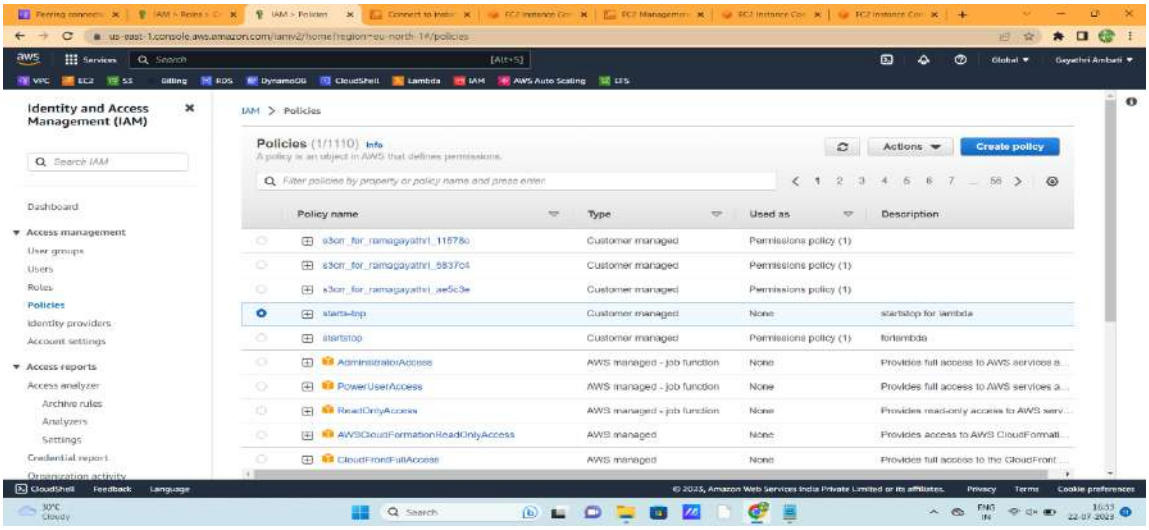

EC2StartStopPolicy
File
Edit
View
{
"Version": "2012-10-17",
"Statement": [
{
"Effect": "Allow",
"Action": [
"logs:CreateLogGroup",
"logs:CreateLogStream",
"logs:PutLogEvents"
],
"Resource": "arn:aws:logs:*:*:*"
},
{
"Effect": "Allow",
"Action": [
"ec2:Start*",
"ec2:Stop*",
],
"Resource": "*"
}
]
}
Remove
Amplify
Amplify Admin
Amplify UI Builder

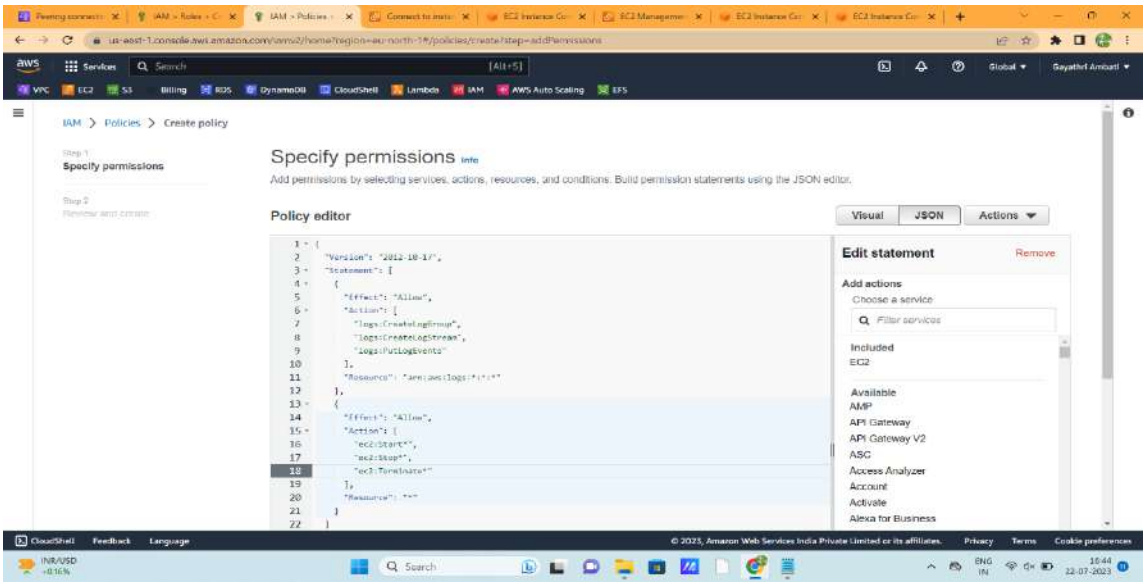

Specify permissions
Add permissions by selecting services, actions, resources, and conditions. Build permission statements using the JSON editor.
Policy editor
Visual
JSON
Actions
Edit statement
Remove
Add actions
Choose a service
Filter services
Included
EC2
Available
AMP
API Gateway
API Gateway V2
ASC
Access Analyzer
Account
Activate
Alexa for Business
Amplify
Amplify Admin
Amplify UI Builder

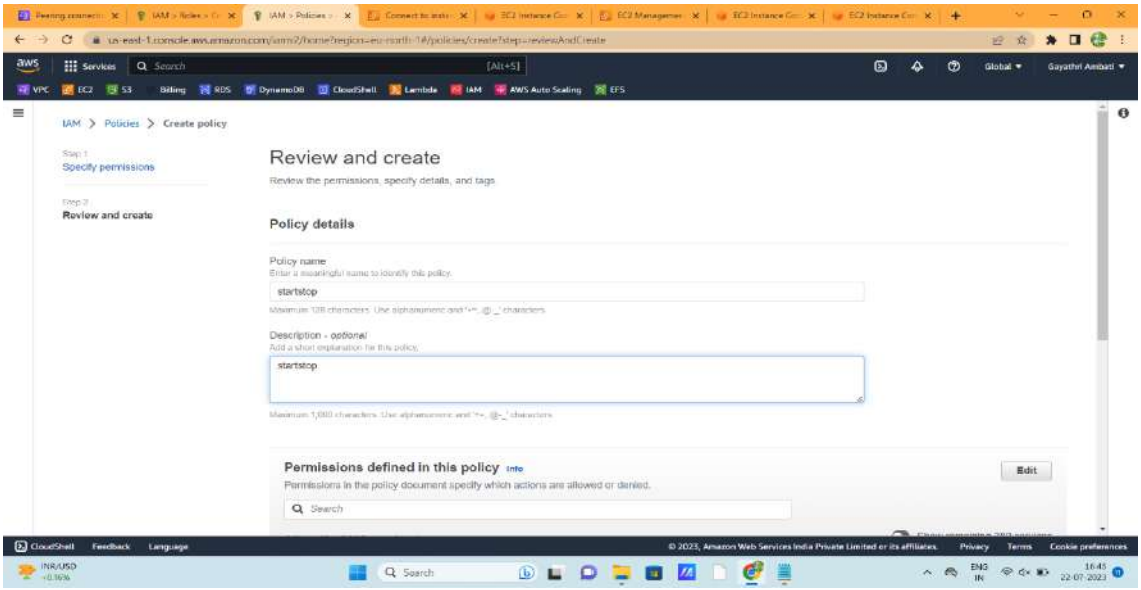
- In create role page Add Permissions and select startstop policy and create role
- In lambda page select the role we created and create function
- In lambda function page go to code option give stopEC2lambdafunction code and change the region and instance id click on the deploy.
- In Test give event as mytest and then click on Test.
- Now the code start executing and after its execution we can check that the instance has been stopped.

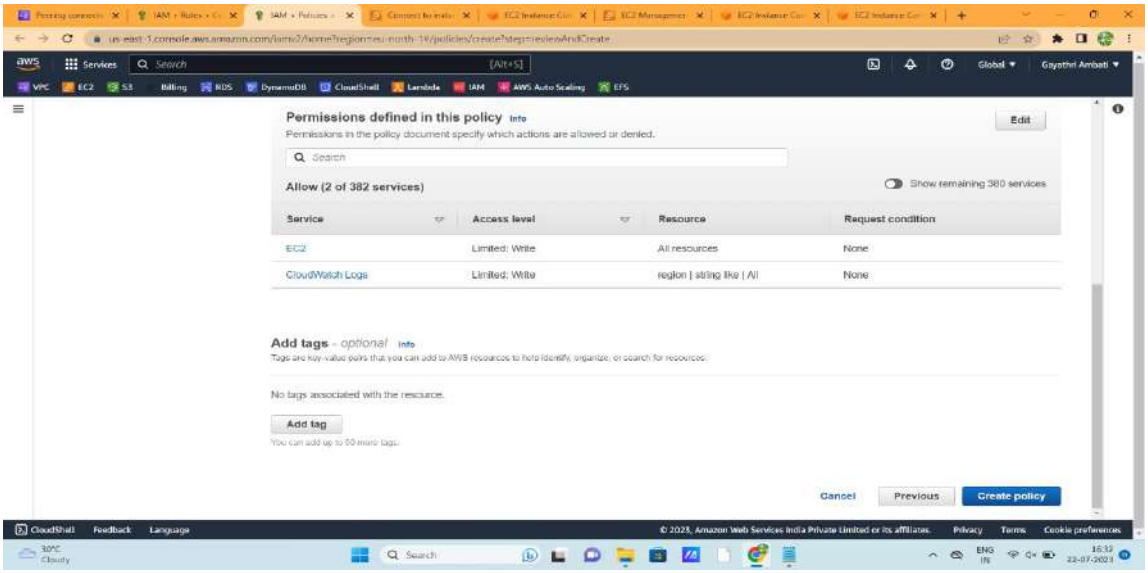


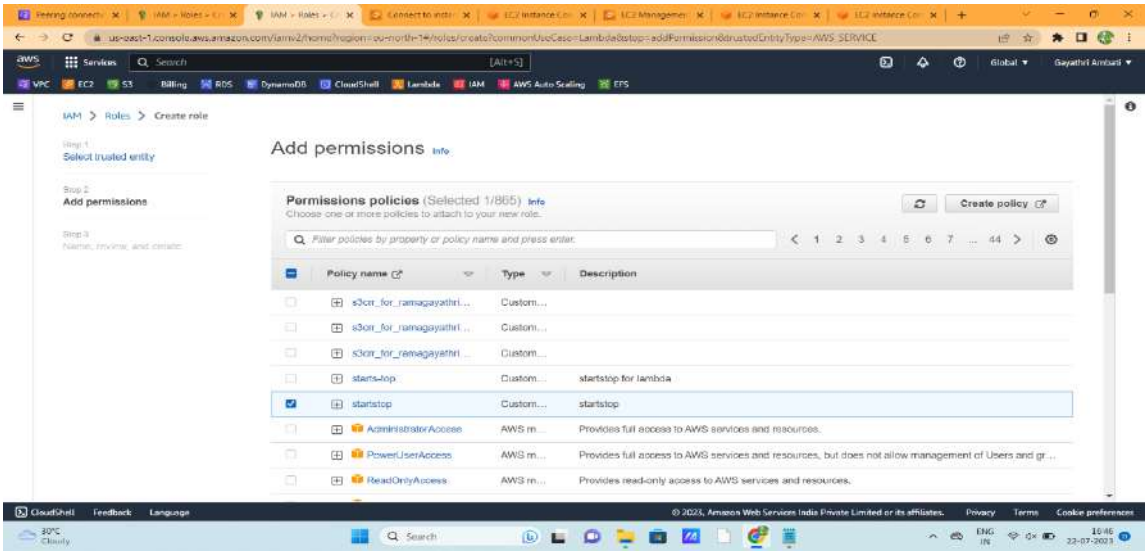


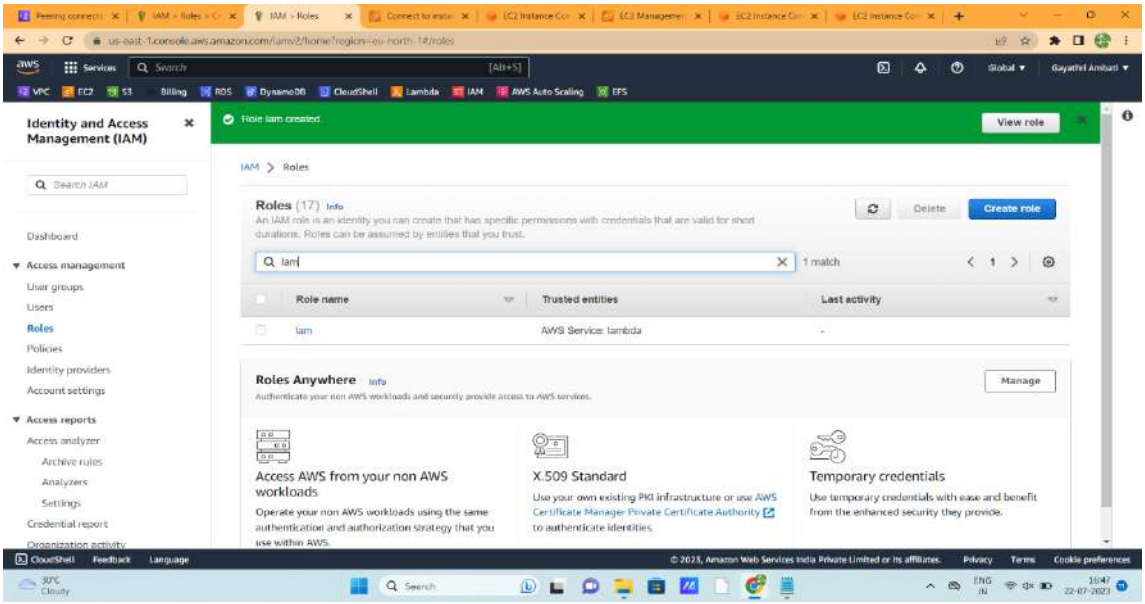

Identity and Access Management (IAM)
IAM > Policies
Policies (1/1110) Info
A policy is an object in AWS that defines permissions.
Actions
Create policy
Filter policies by property or policy name and press enter
Policy name
Type
Used as
Description
s3cm_for_ramagayathri_11578o
s3cm_for_ramagayathri_5837o4
s3cm_for_ramagayathri_ae5c3e
startstop
Customer managed
None
startstop for lambda
startstop
Customer managed
Permissions policy (1)
forlambda
AdministratorAccess
AWS managed - job function
Provides full access to AWS services a...
PowerUserAccess
ReadOnlyAccess
AWSCloudFormationReadOnlyAccess
AWS managed
CloudFrontFullAccess
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Access reports
Access analyzer
Archive rules
Analyzers
Settings
Credential report
Organization activity

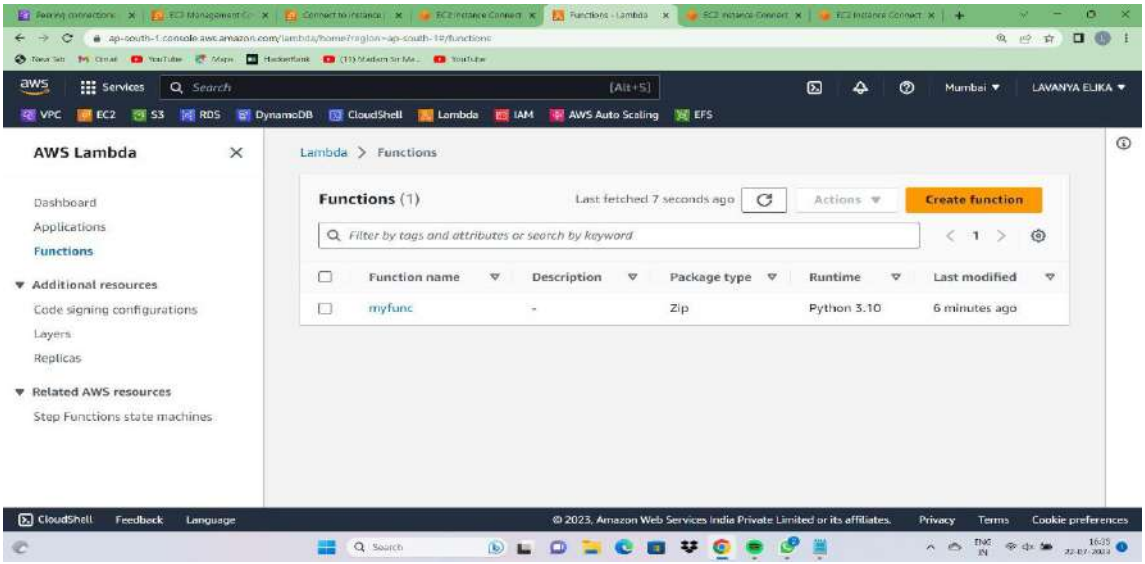

IAM > Policies > Create policy
Step 1
Specify permissions
Step 2
Review and create
Specify permissions Info
Add permissions by selecting services, actions, resources, and conditions. Build permission statements using the JSON editor.
Policy editor
Visual
JSON
Actions
Edit statement
Remove
Add actions
Choose a service
Filter services
Included
EC2
Available
AMP
API Gateway
API Gateway V2
ASC
Access Analyzer
Account
Activate
Alexa for Business

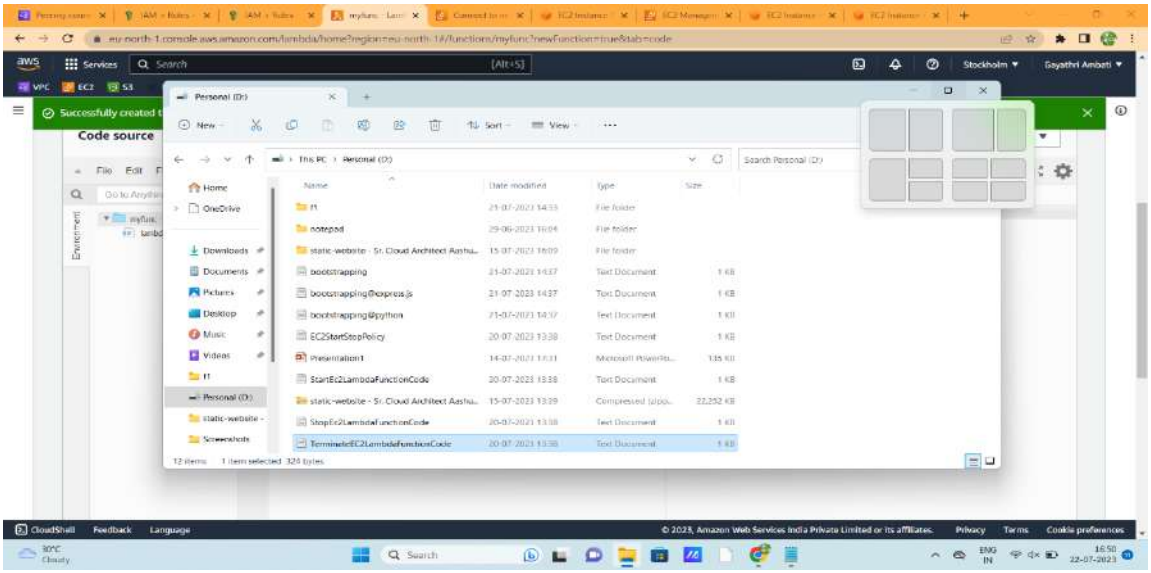

IAM > Policies > Create policy
Step 1
Specify permissions
Step 2
Review and create
Review and create
Review the permissions, specify details, and tags.
Policy details
Policy name
Enter a meaningful name to identify this policy.
startstop
Description - optional
Add a short explanation for this policy.
startstop
Permissions defined in this policy Info
Permissions in the policy document specify which actions are allowed or denied.
Edit
Search

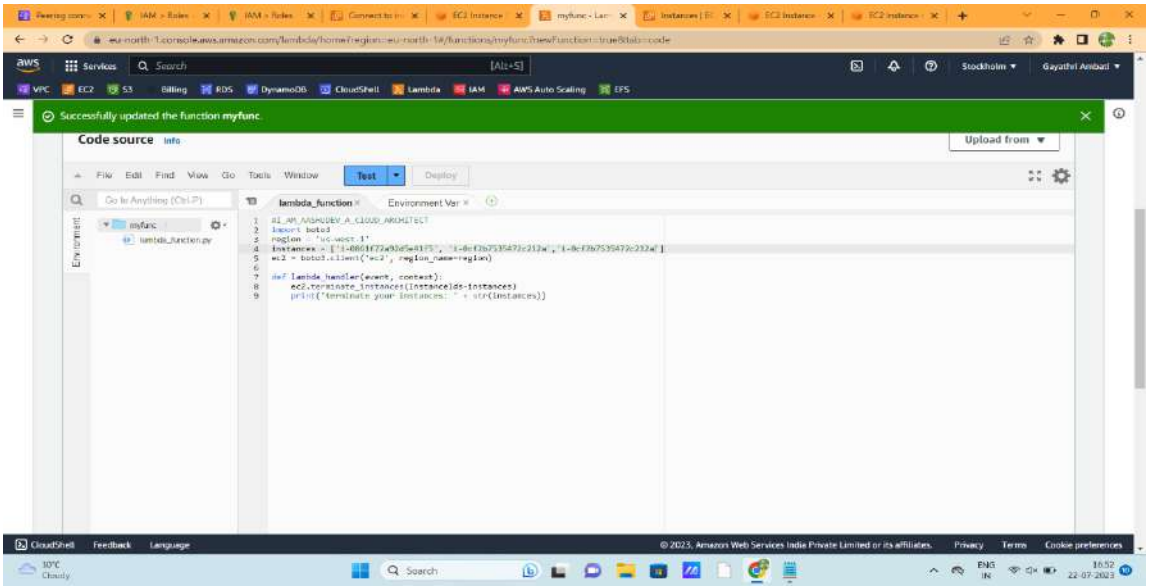

Permissions defined in this policy Info
Edit
Allow (2 of 382 services)
Service
Access level
Resource
Request condition
EC2
Limited: Write
All resources
None
CloudWatch Logs
Limited: Write
region | string like | All
None
Add tags - optional Info
No tags associated with the resource.
Add tag
Cancel
Previous
Create policy

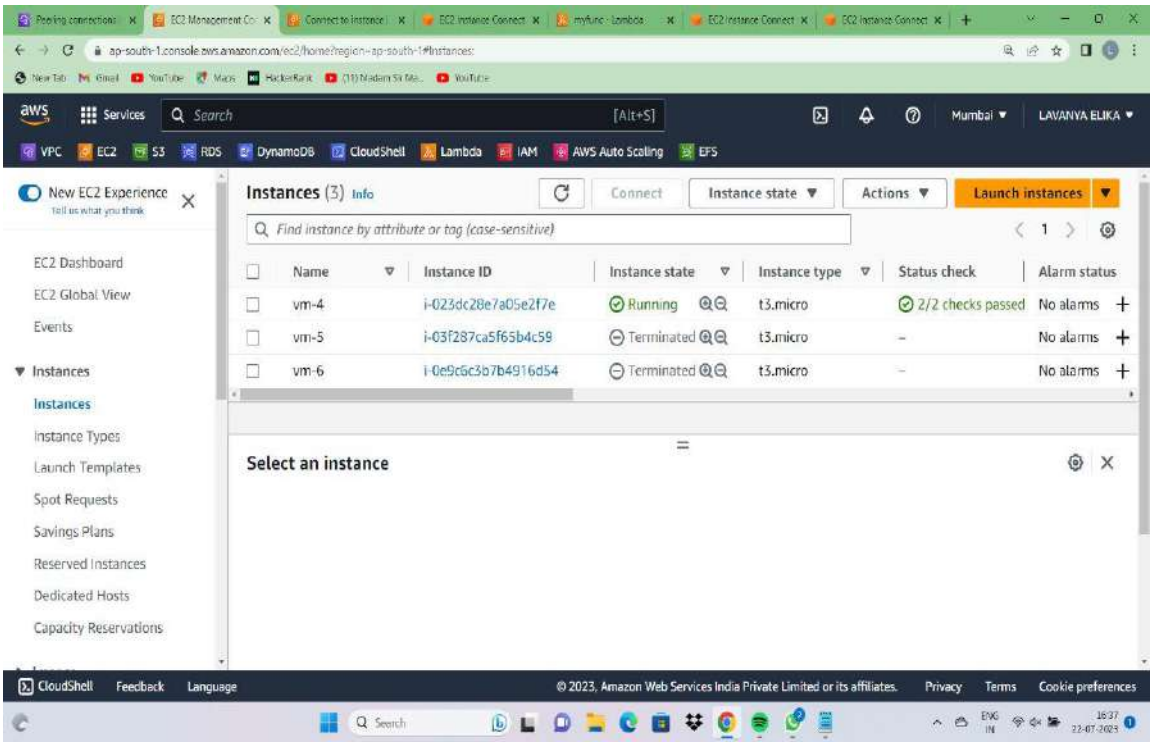

IAM > Roles > Create role
Add permissions Info
Permissions policies (Selected 1/865) Info
Create policy
Policy name
Type
Description
starts-top
Custom...
startstop for lambda
startstop
Custom...
startstop
AdministratorAccess
AWS m...
Provides full access to AWS services and resources.
PowerUserAccess
AWS m...
ReadOnlyAccess
AWS m...
Provides read-only access to AWS services and resources.

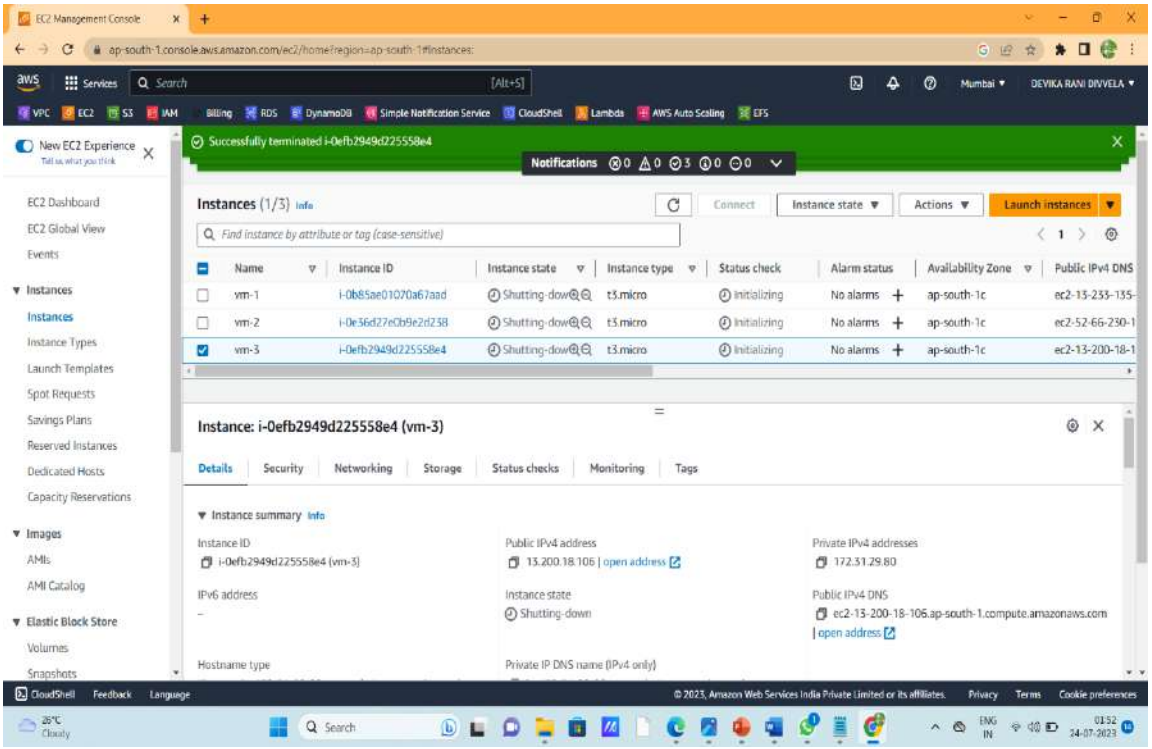

Identity and Access Management (IAM)
Role lam created
View role
IAM > Roles
Roles (17) Info
Delete
Create role
lam
1 match
Role name
Trusted entities
Last activity
lam
AWS Service: lambda
Roles Anywhere Info
Manage
Access AWS from your non AWS workloads
X.509 Standard
Temporary credentials


AWS Lambda
Dashboard
Applications
Functions
Additional resources
Code signing configurations
Layers
Replicas
Related AWS resources
Step Functions state machines
Lambda > Functions
Functions (1)
Last fetched 7 seconds ago
Actions
Create function
Filter by tags and attributes or search by keyword
Function name
Description
Package type
Runtime
Last modified
myfunc
-
Zip
Python 3.10
6 minutes ago
CloudShell
Feedback
Language
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences


Code source
CloudShell
Feedback
Language
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences


Successfully updated the function myfunc.
Code source
Upload from
Test
Deploy
CloudShell
Feedback
Language
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences


Instances (3) Info
Connect
Instance state
Actions
Launch instances
Find instance by attribute or tag (case-sensitive)
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
vm-4
i-023dc28e7a05e2f7e
Running
t3.micro
2/2 checks passed
No alarms
vm-5
i-03f287ca5f65b4c59
Terminated
t3.micro
No alarms
vm-6
i-0e9c6c3b7b4916d54
Terminated
t3.micro
No alarms
Select an instance
New EC2 Experience
EC2 Dashboard
EC2 Global View
Events
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
CloudShell
Feedback
Language
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences


Successfully terminated i-0efb2949d225558e4
Instances (1/3) Info
Connect
Instance state
Actions
Launch instances
Find instance by attribute or tag (case-sensitive)
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public IPv4 DNS
vm-1
i-0b85ae01070a67aad
Shutting-down
t3.micro
Initializing
No alarms
ap-south-1c
vm-2
i-0e36d27e0b9e2d238
Shutting-down
t3.micro
Initializing
No alarms
ap-south-1c
vm-3
i-0efb2949d225558e4
Shutting-down
t3.micro
Initializing
No alarms
ap-south-1c
Instance: i-0efb2949d225558e4 (vm-3)
Details
Security
Networking
Storage
Status checks
Monitoring
Tags
Instance summary Info
Instance ID
i-0efb2949d225558e4 (vm-3)
Public IPv4 address
13.200.18.106 | open address
Private IPv4 addresses
172.31.29.80
IPv6 address
Instance state
Shutting-down
Public IPv4 DNS
ec2-13-200-18-106.ap-south-1.compute.amazonaws.com | open address
Hostname type
Private IP DNS name (IPv4 only)
CloudShell
Feedback
Language
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences

CONCLUSION:

These architect helps to create a VPC peering connection is a networking connection between two VPCs that enables routing using each VPC's private IP addresses as if they were in the same network. VPC peering connections can be created between your own VPCs or with a VPC in another AWS account. VPC peering also supports inter-region peering.