LABORATORY REPORT

Semester: Fall Semester 2025-26 (Fast Track)

Cloud Application Development

MCA2007

Slot : A11+A12+A14+F14

Submitted by

NAME: Aditya Pratap Singh

REGNO: 24MCA10024

Master of Computer Application

in

Submited to

Dr. Munsifa Firdaus Khan

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING VIT BHOPAL UNIVERSITY

APRIL, 2025

TABLE OF CONTENTS

Date: 23/05/2025	Title
Exp. No: 01	Creating an EC2 Instance on a Unix System.

Creating an EC2 Instance on a Unix System

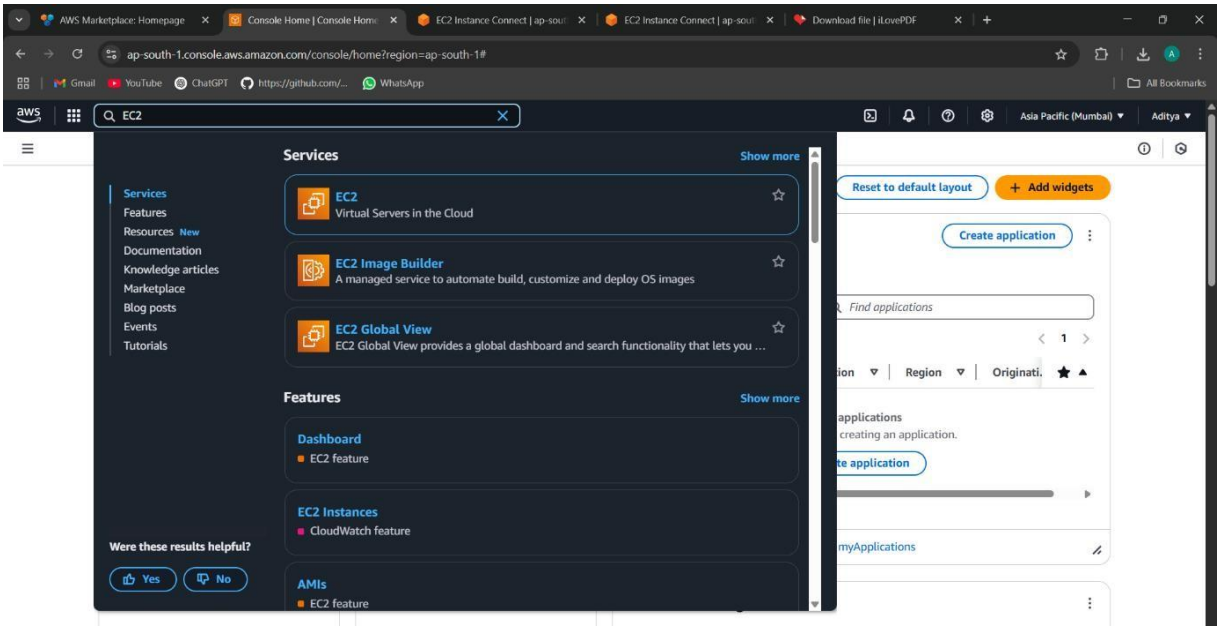
Creating an EC2 instance from a Unix-based system involves using the AWS Command Line Interface (CLI) to interact with AWS services programmatically. This method is ideal for developers and system administrators who prefer automation or terminal-based workflows.

The process begins by configuring the AWS CLI with user credentials and a default region. Once configured, you can launch an EC2 instance by specifying key parameters such as the Amazon Machine Image (AMI) ID, instance type, key pair name, security group, and subnet or VPC settings. These parameters define the operating system, hardware capacity, and network rules for the instance.

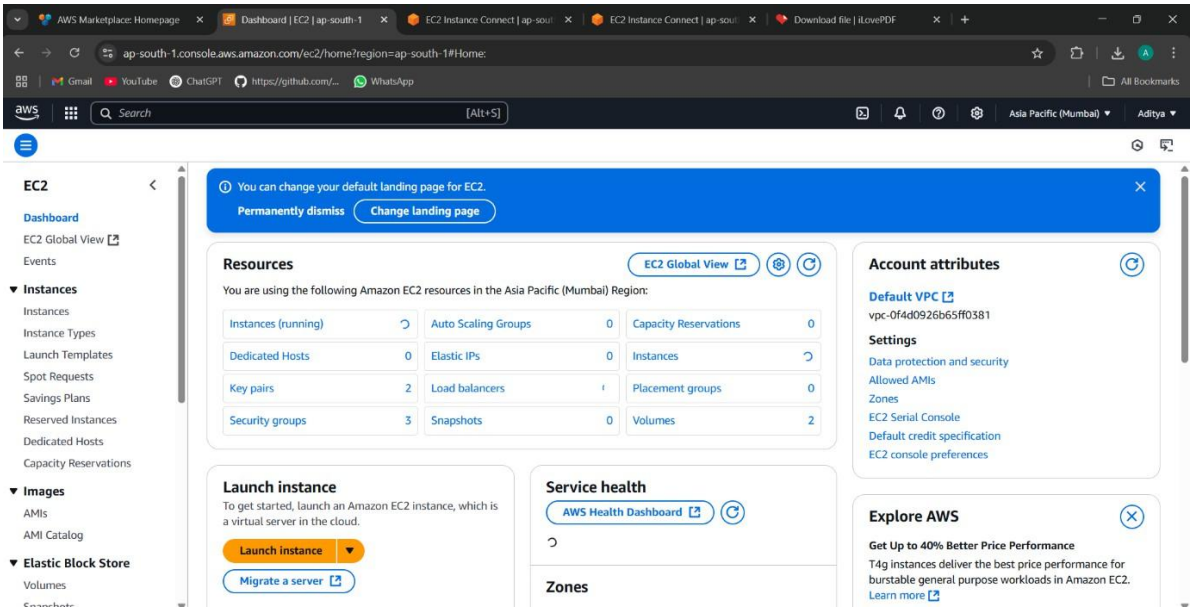
After running the appropriate command, AWS provisions a virtual machine in the cloud. The instance can then be accessed via SSH using the key pair specified during launch. This approach provides full control over the provisioning process and is particularly useful for scripting, automation, and infrastructure-as-code scenarios.

Using the CLI on Unix systems streamlines EC2 instance creation, making it efficient to deploy and manage instances in a consistent, repeatable manner.

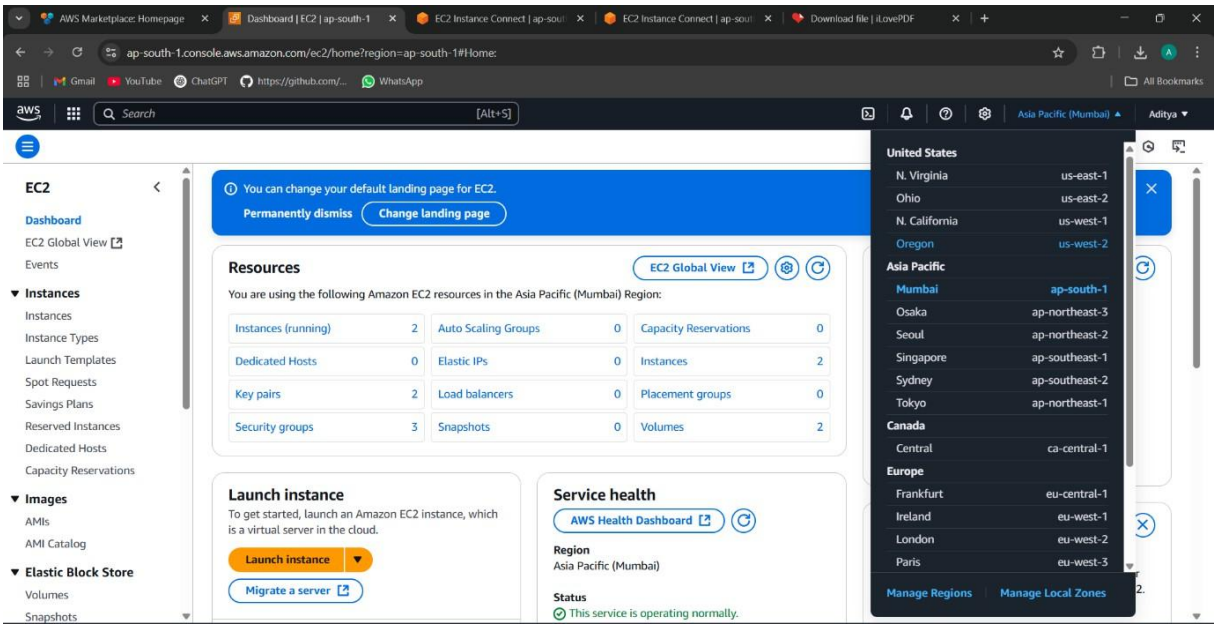
STEP1: OPEN THE AWS ROOT ACCOUNT AND SEARCH FOR EC2



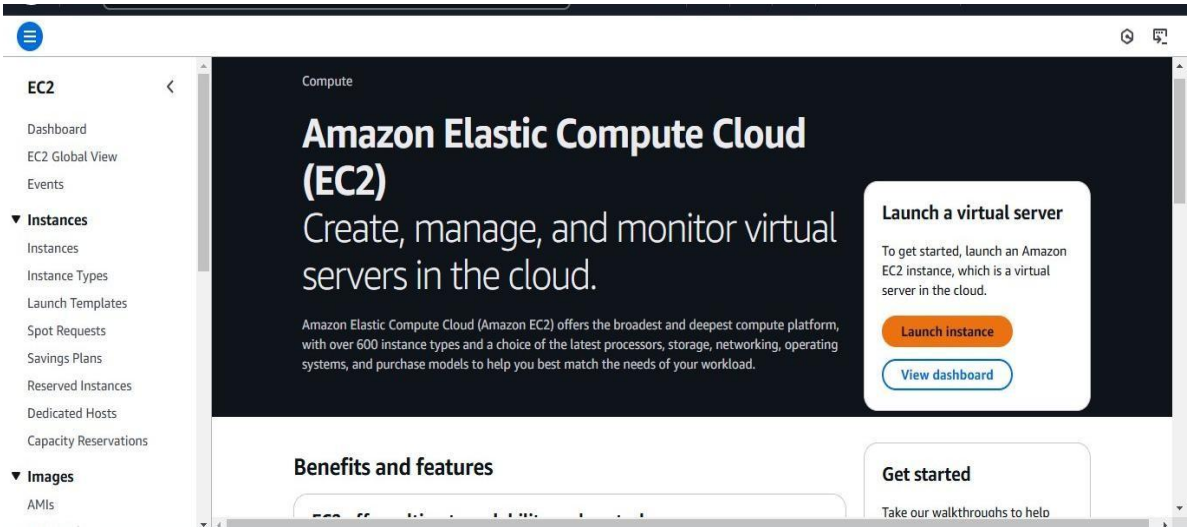
STEP2: CLICK ON EC2, EC2 DASHBOARD WILL APPEAR.


AWS Marketplace: Homepage
Dashboard | EC2 | ap-south-1
EC2 Instance Connect | ap-sout
EC2 Instance Connect | ap-sout
Download file | iLovePDF
ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#Home:
Gmail
YouTube
ChatGPT
https://github.com/...
WhatsApp
All Bookmarks
Search
[Alt+S]
Asia Pacific (Mumbai)
Aditya
EC2
Dashboard
EC2 Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
You can change your default landing page for EC2.
Permanently dismiss
Change landing page
Resources
EC2 Global View
You are using the following Amazon EC2 resources in the Asia Pacific (Mumbai) Region:
Instances (running)
Auto Scaling Groups 0
Capacity Reservations 0
Dedicated Hosts 0
Elastic IPs 0
Instances
Key pairs 2
Load balancers
Placement groups 0
Security groups 3
Snapshots 0
Volumes 2
Account attributes
Default VPC
vpc-0f4d0926b65ff0381
Settings
Data protection and security
Allowed AMIs
Zones
EC2 Serial Console
Default credit specification
EC2 console preferences
Launch instance
To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.
Launch instance
Migrate a server
Service health
AWS Health Dashboard
Zones
Explore AWS
Get Up to 40% Better Price Performance
T4g instances deliver the best price performance for burstable general purpose workloads in Amazon EC2.
Learn more

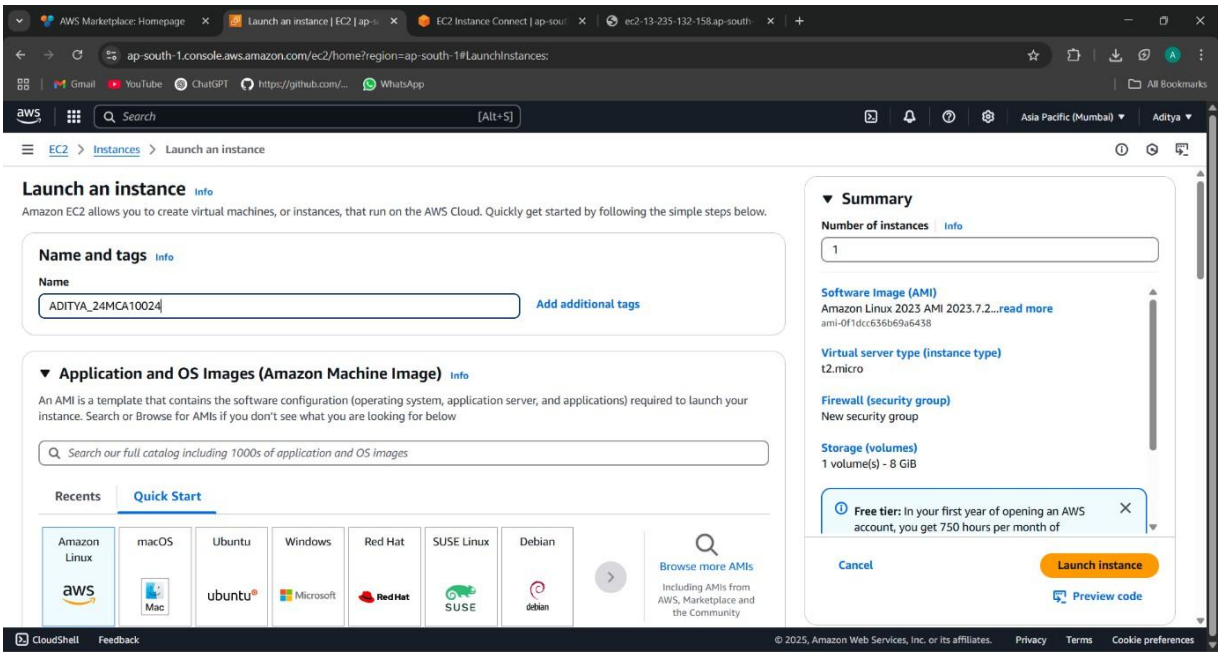
STEP3: SELECT THE REGION>ASIA PACIFIC>MUMBAI



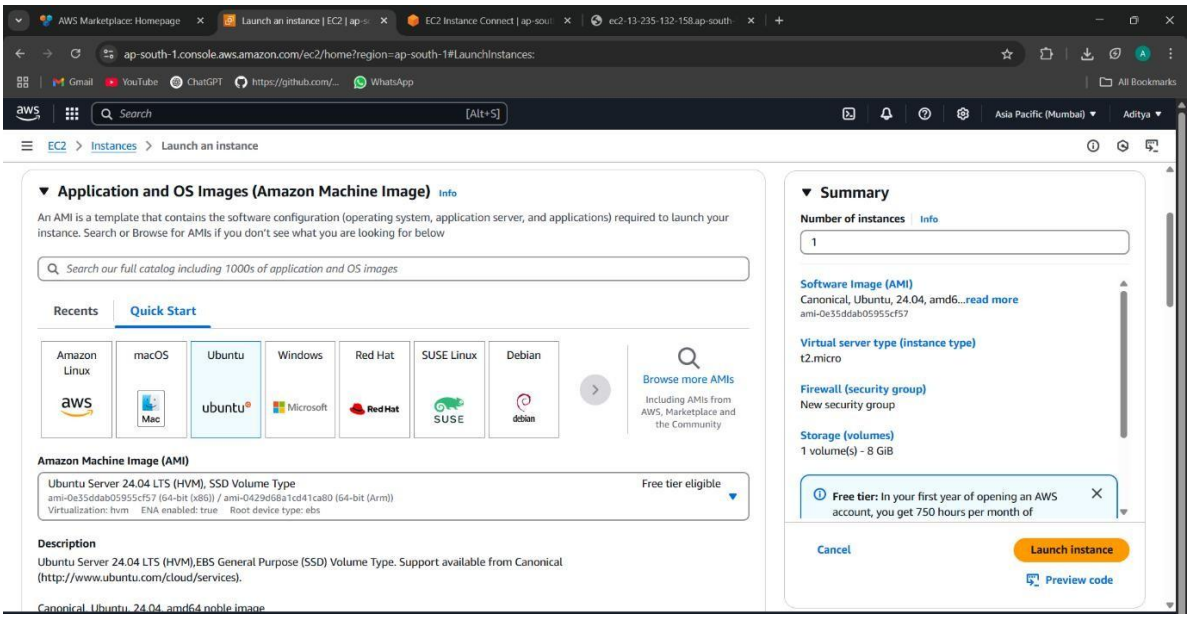
STEP4: SELECT LAUNCH INSTANCE



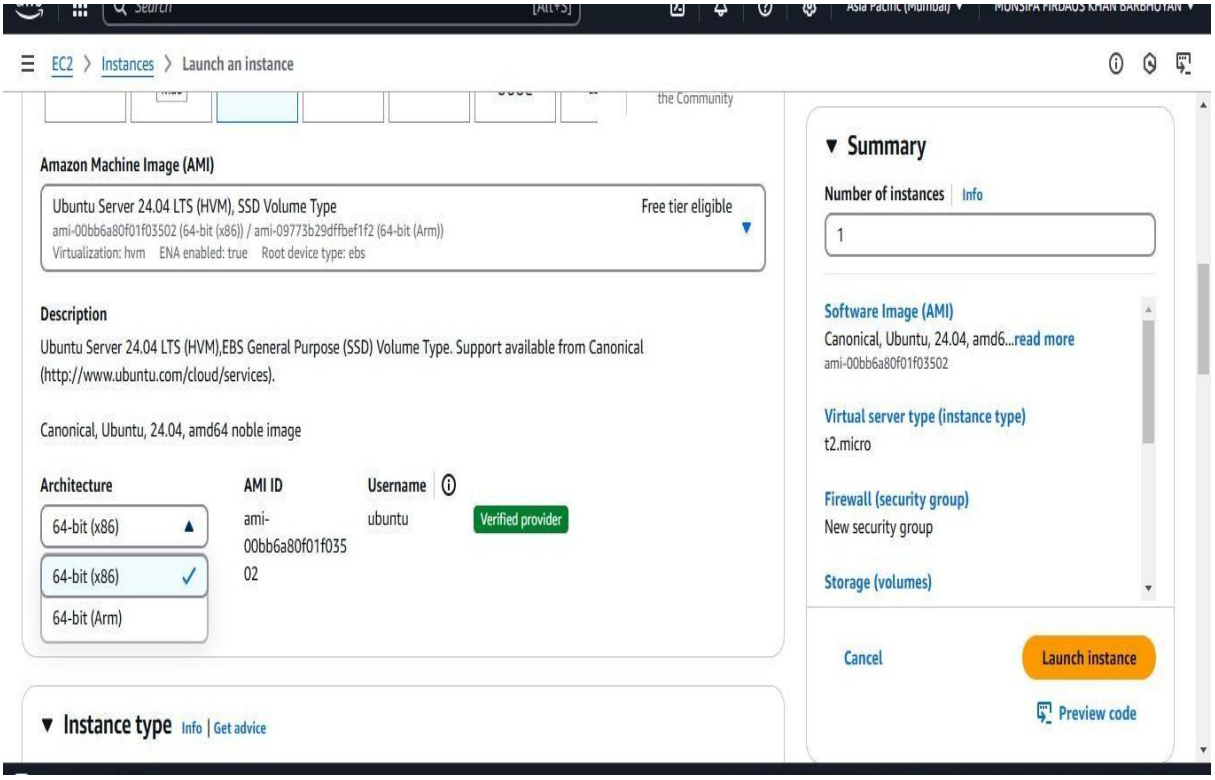
STEP5: TYPE THE NAME OF THE INSTANCE



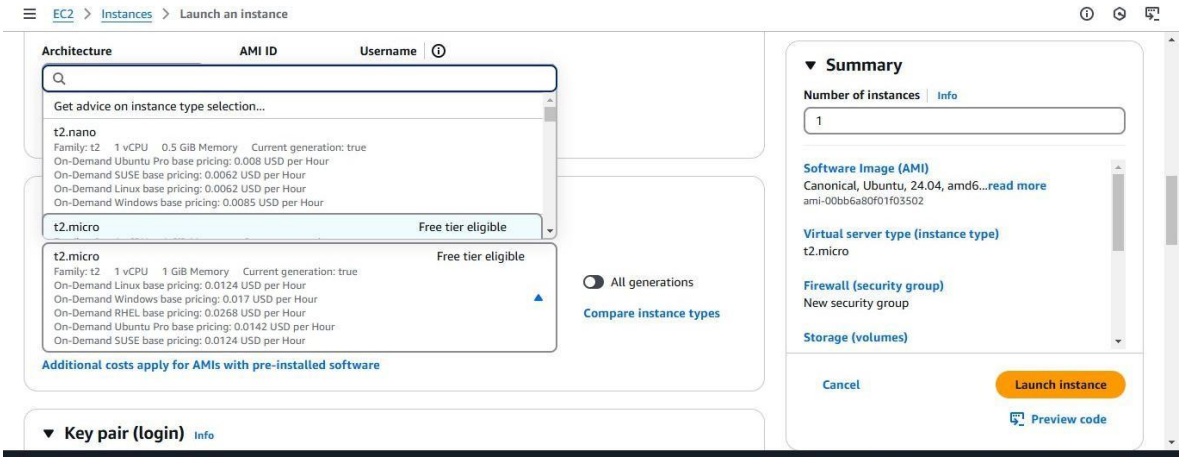
STEP6: CHOOSE THE OPERATING SYSTEM>UBUNTU



STEP7: CHOOSE THE ARCHITECTURE>64-bit(x86)

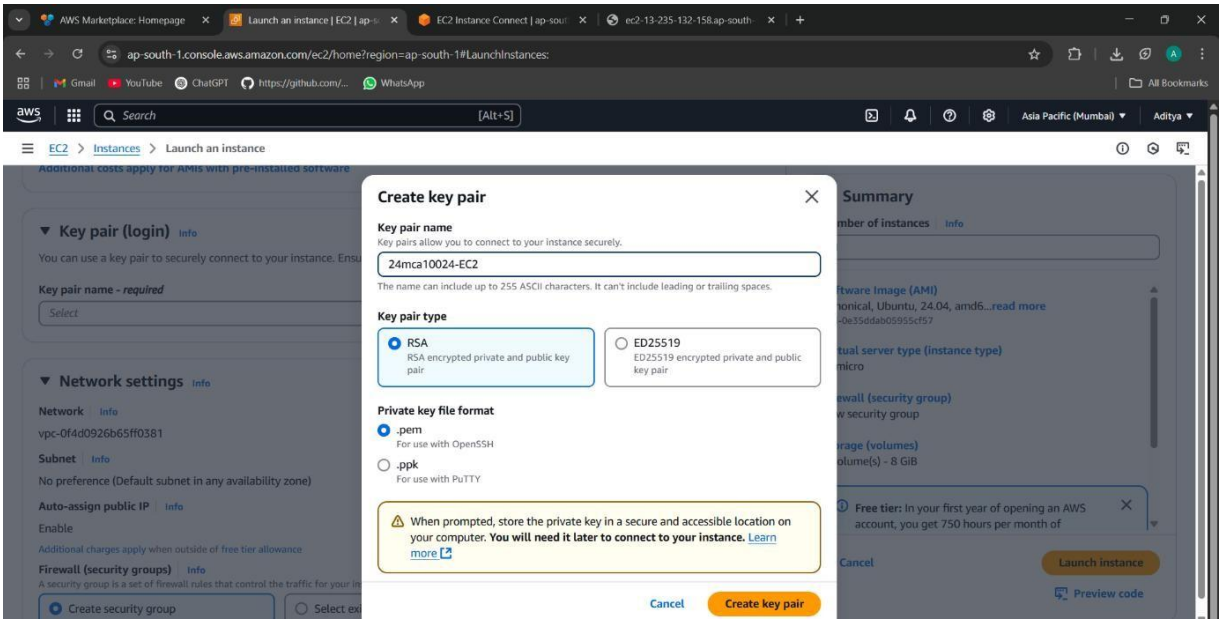


STEP8: CHOOSE INSTANCE TYPE: t2 micro (Free tier eligible)

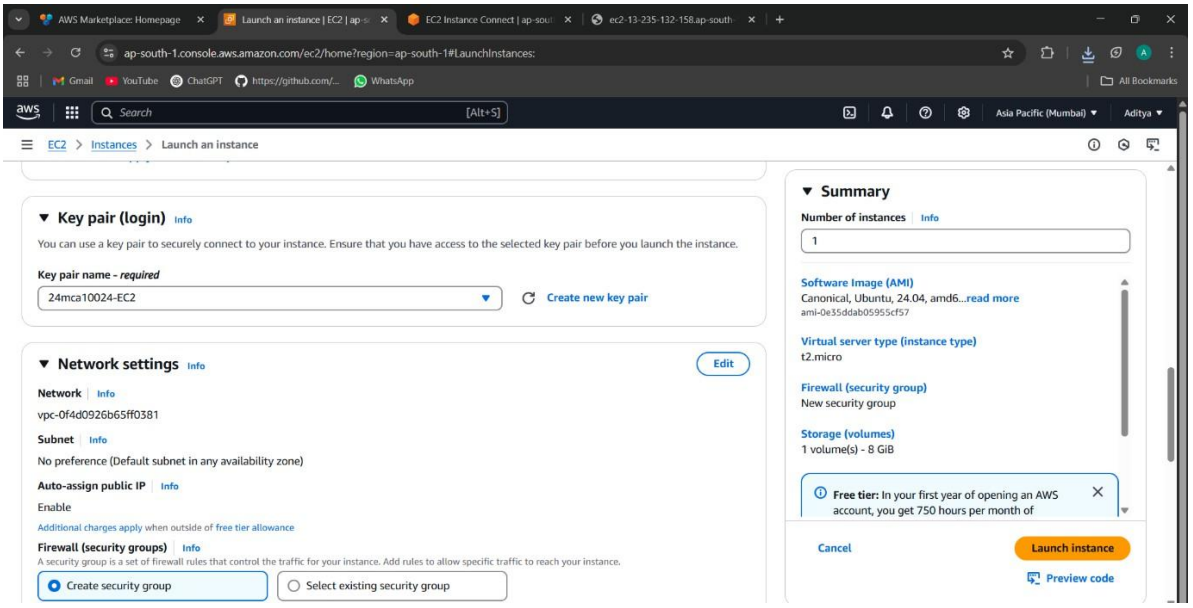


STEP9: CREATE A NEW KEY PAIR:

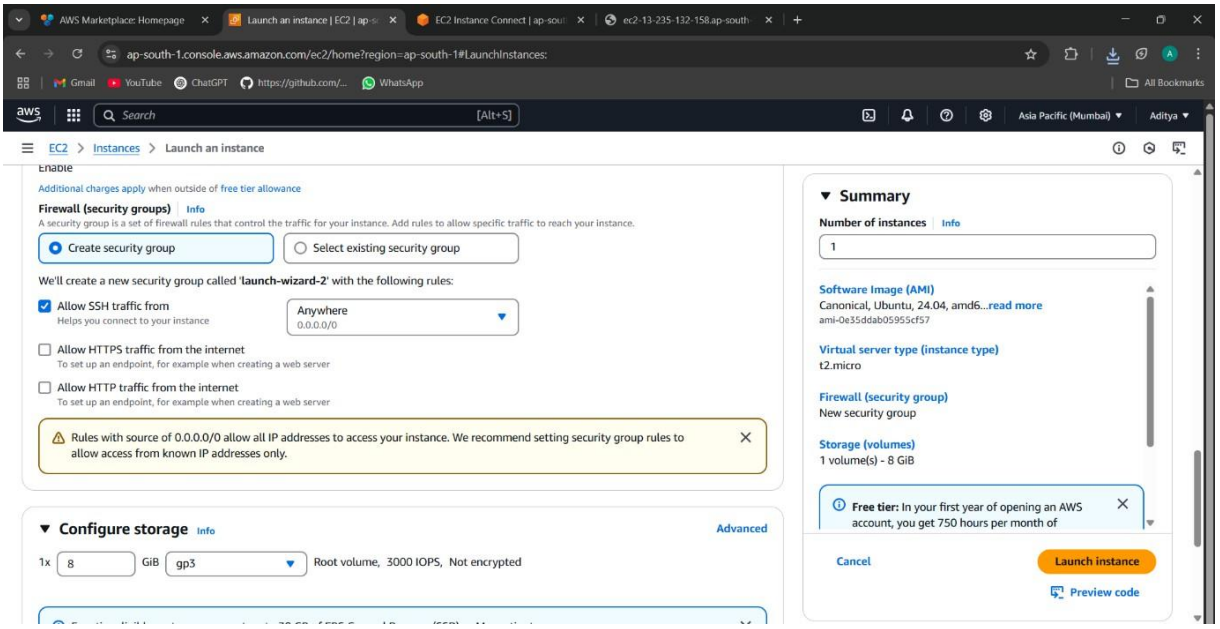
There are two keys involved in this step. One is private key for user and another is public key for the EC2 instance. Once we create a pair, both the keys will be generated. Public key need to saved separately for future use. Both keys will have same name as given.



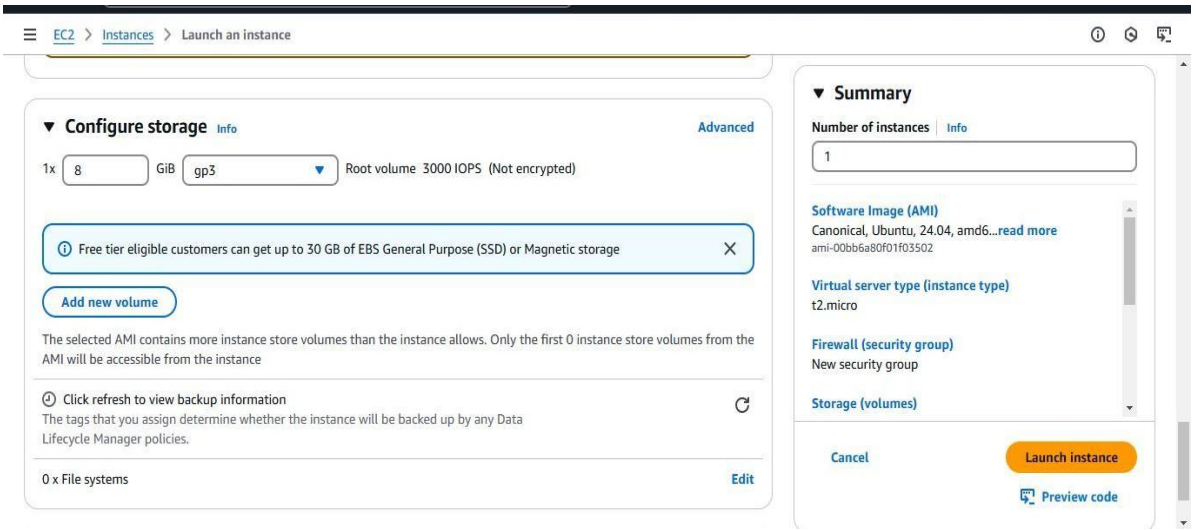
STEP10: PUBLIC KEY IS AUTOMATICALLY ENABLED IN EC2 INSTANCE



STEP11: CHOOSE THE DEFAULT NETWORK SETTINGS(NOTE: Auto-assign public IP should be enabled and source:0.0.0.0/0 allows all the IP addresses to access the EC2 instance)

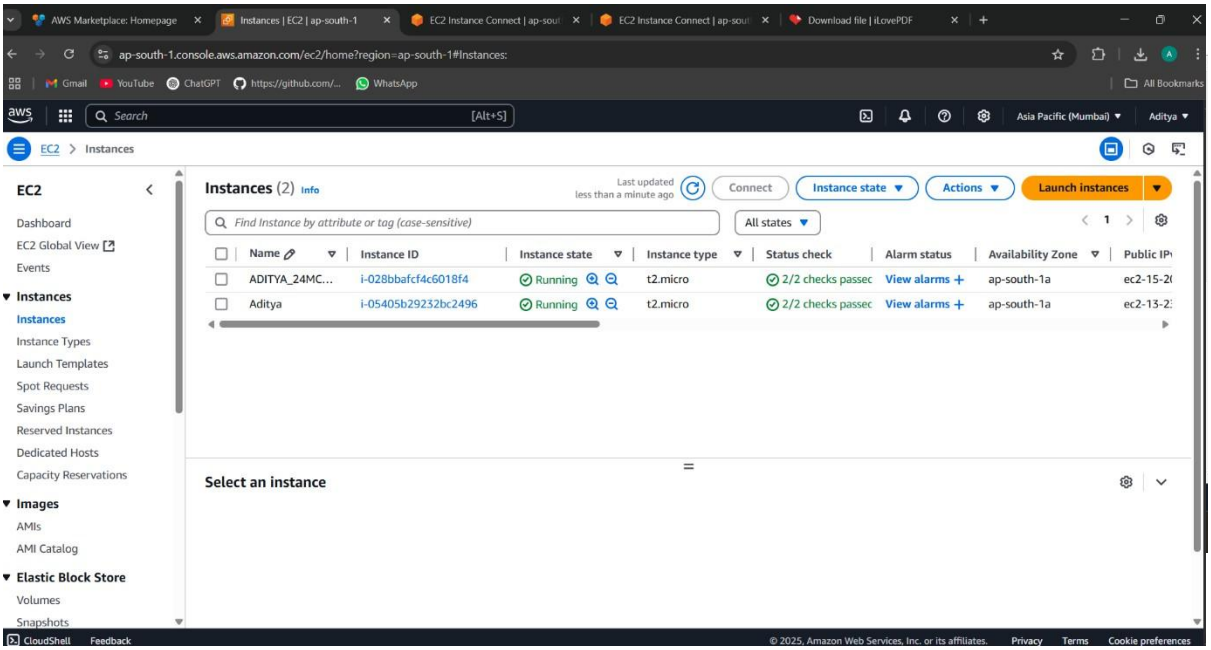


STEP12: CHOOSE THE DEFAULT CONFIGURE STORAGE AND CLICK ON LAUNCH INSTANCE

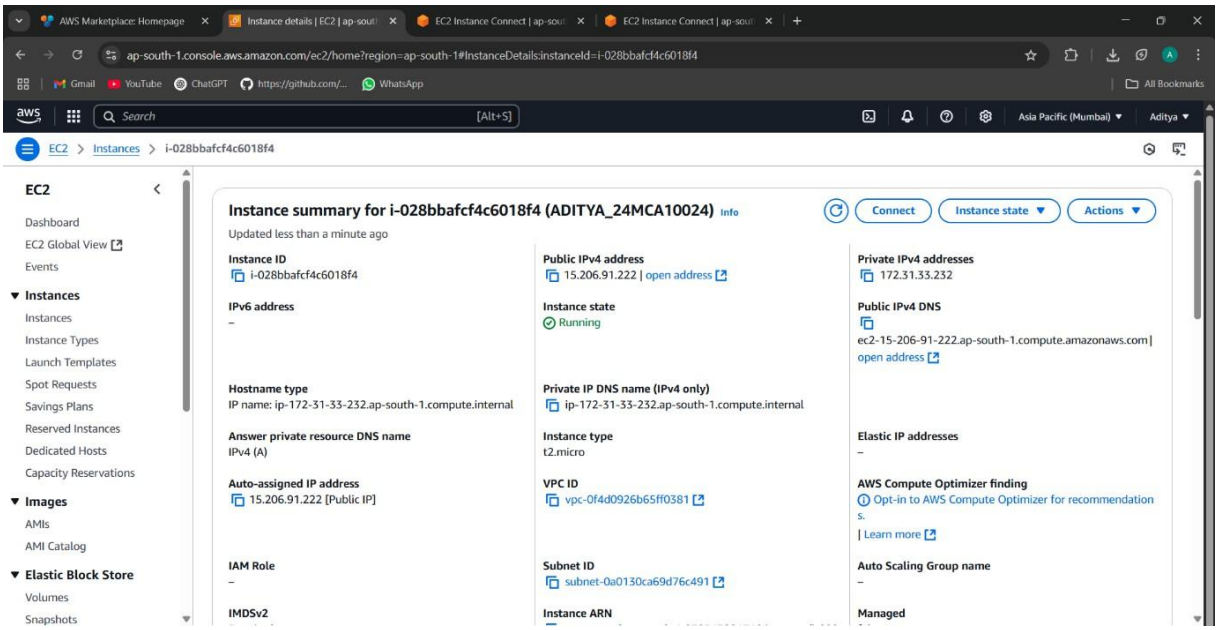


STEP13: EC2 INSTANCE IS CREATED, CLICK ON THE INSTANCE.

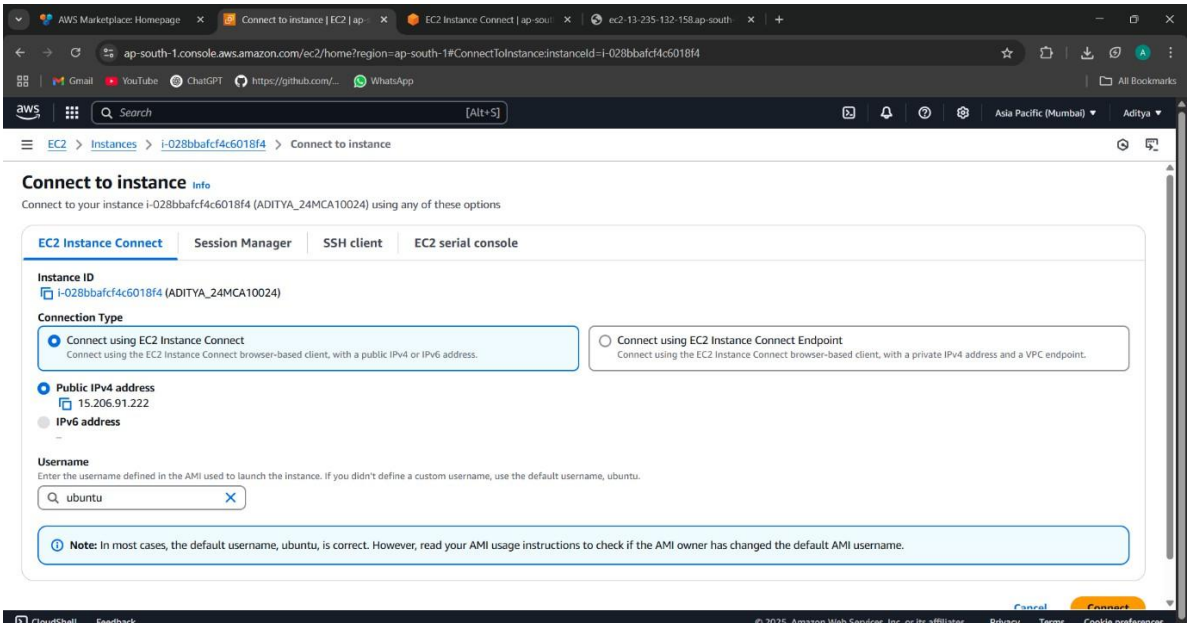
STEP14: NOW CLICK ON THE EC2 INSTANCE



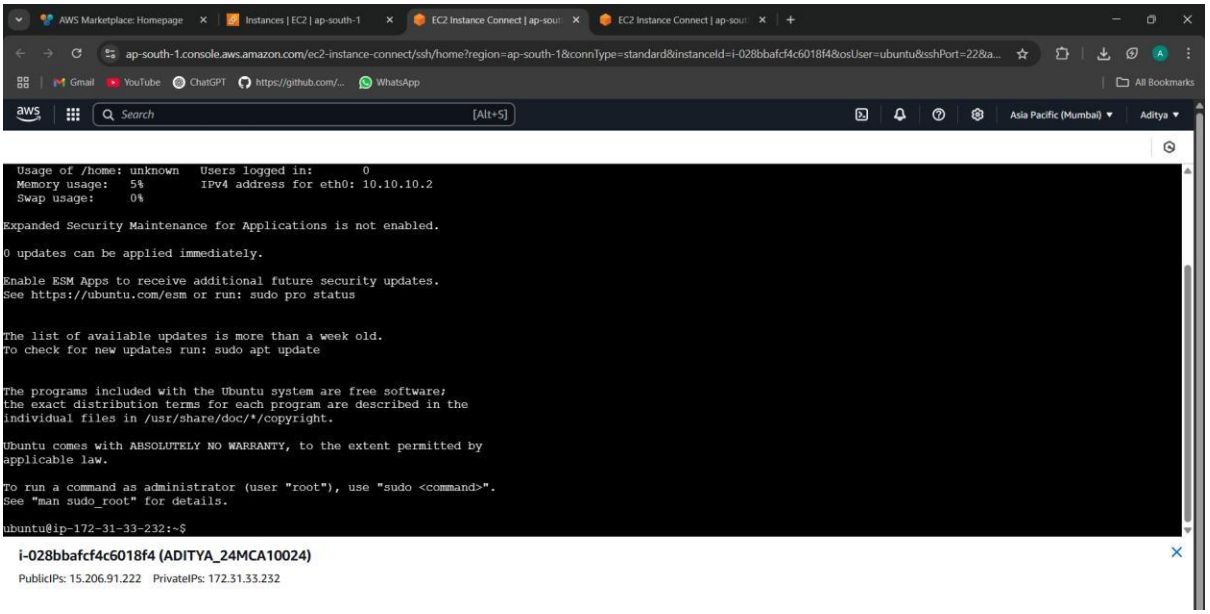
STEP15: DETAILS OF THE EC2 INSTANCE IS SHOWN, CLICK ON CONNECT



STEP16: CLICK ON EC2 INSTANCE CONNECT>CONNECT



STEP17: IT WILL LAND YOU TO THE TERMINAL PAGE



Date: 23/05/2025	Title
Exp. No: 02	How to launch EC2 Instance in Windows.

HOW TO LAUNCH AN EC2 INSTANCE IN WINDOWS

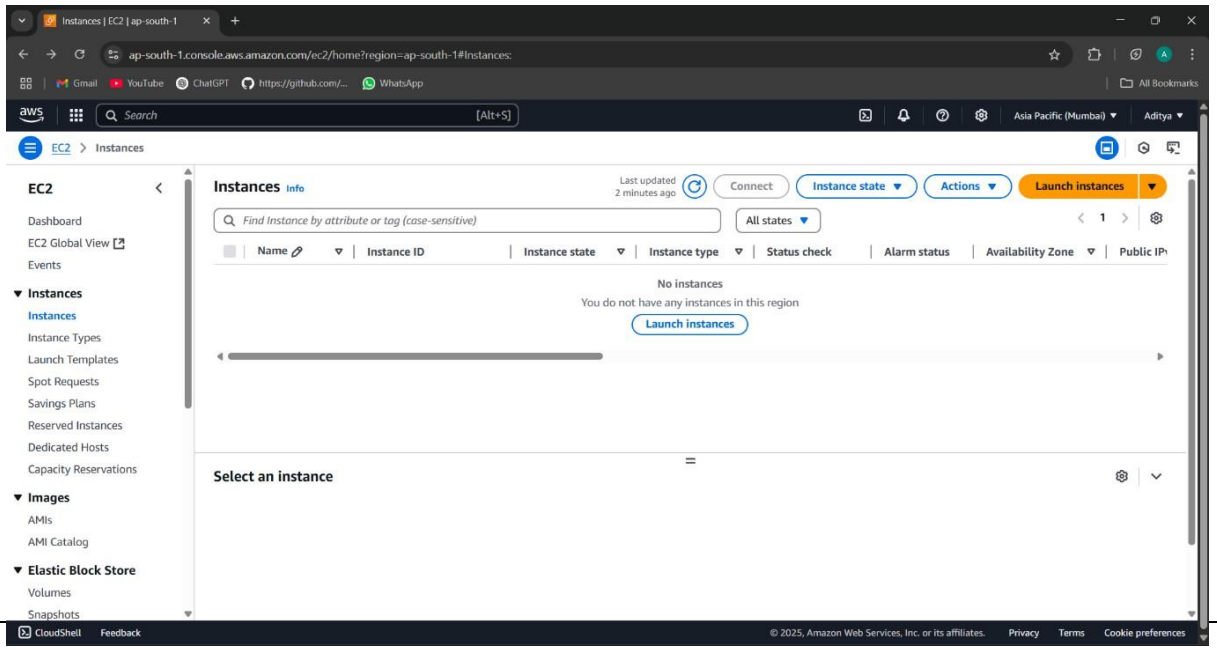
Launching an EC2 instance from a Windows system is typically done through the **AS Management Console**, a web-based interface that provides an intuitive and user-friendly way to provision cloud resources.

The process begins by logging into the AWS Console and navigating to the EC2 dashboard. From there, users can initiate the instance launch process by choosing an **Amazon Machine Image (AMI)**—which determines the operating system (e.g., Amazon Linux, Ubuntu, or Windows Server)—and selecting an **instance type** that defines the hardware specifications like CPU and memory.

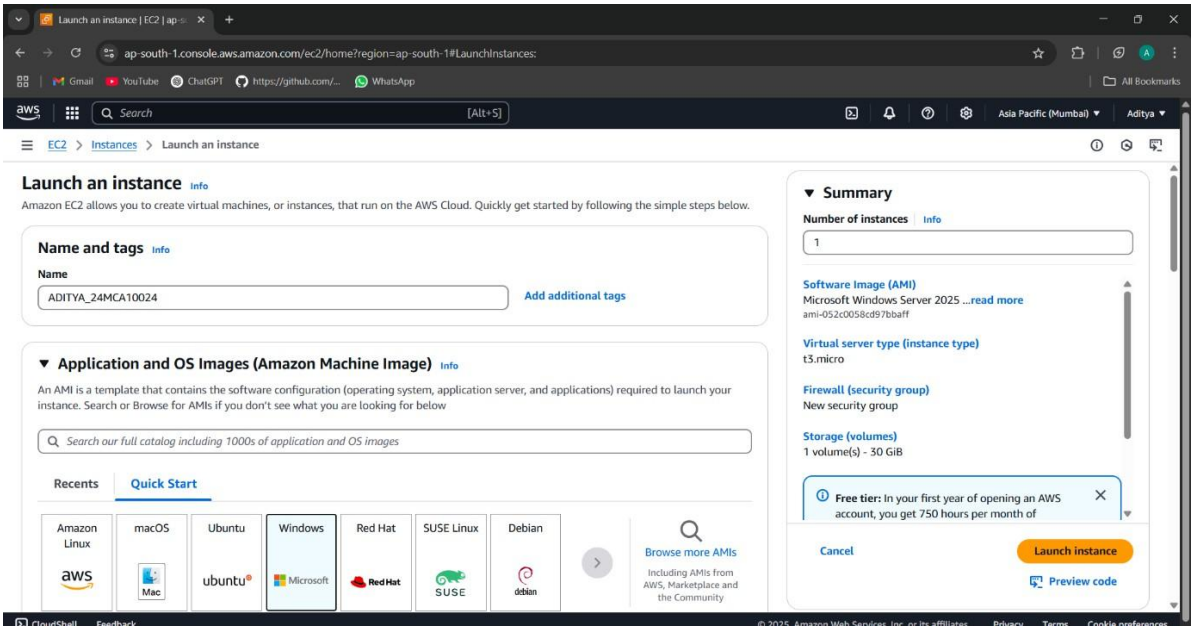
Next, users configure instance settings, including networking (VPC and subnet), storage, security groups (firewall rules), and an SSH key pair (or RDP credentials for Windows AMIs) for remote access. Once launched, the instance appears in the EC2 dashboard and can be accessed using tools like **PuTTY** (for Linux-based instances) or **Remote Desktop Connection (RDP)** (for Windows-based instances).

This GUI-based method is ideal for beginners or those who prefer visual configuration over command-line tools. It allows detailed customization and easy monitoring, making it a straightforward way to deploy virtual servers in the AWS Cloud using a Windows environment.

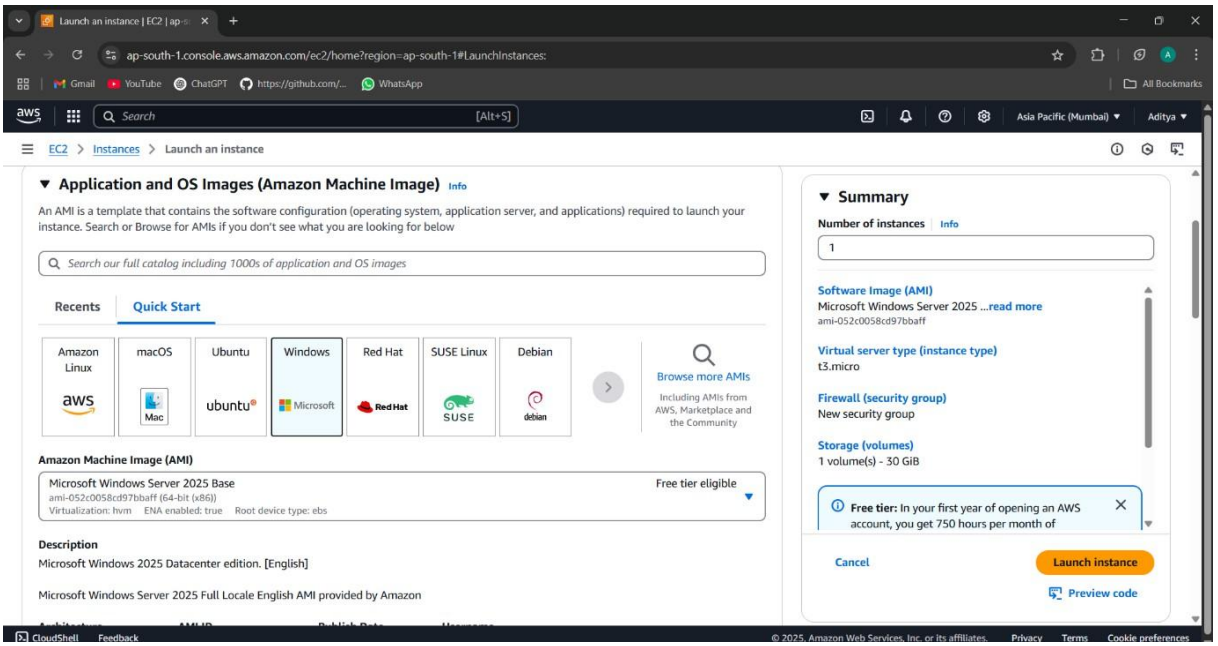
STEP1: LOG IN WITH ROOT USER>EC2 DASHBOARD>CLICK ON INSTANCES> LAUNCH INSTANCES



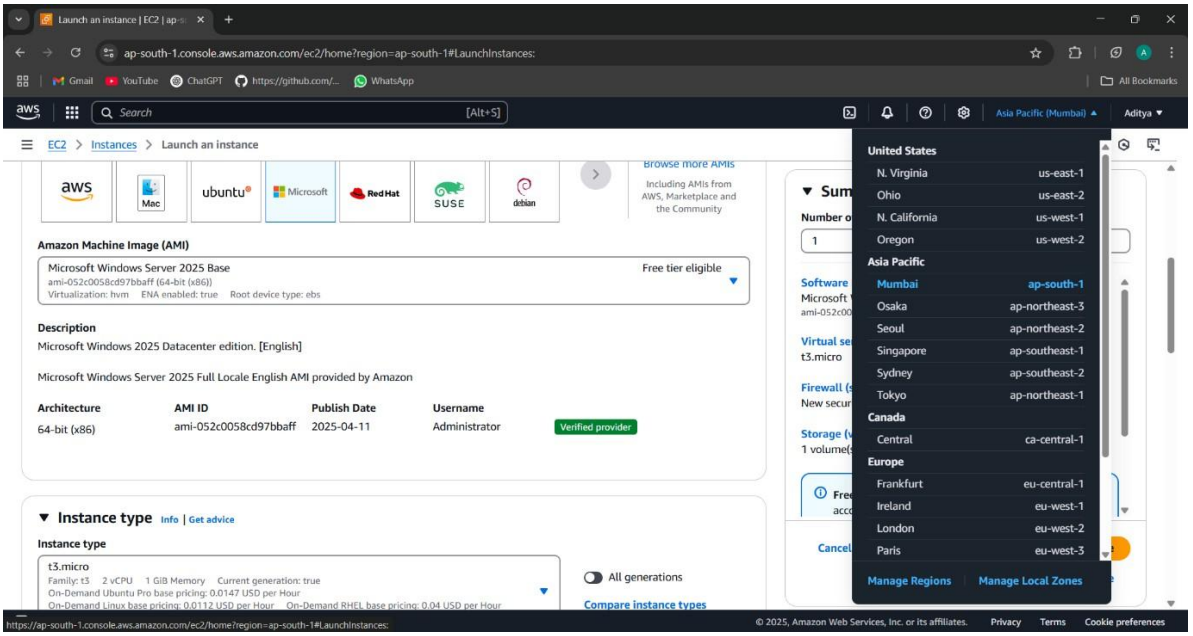
STEP2: WRITE AN INSTANCE NAME



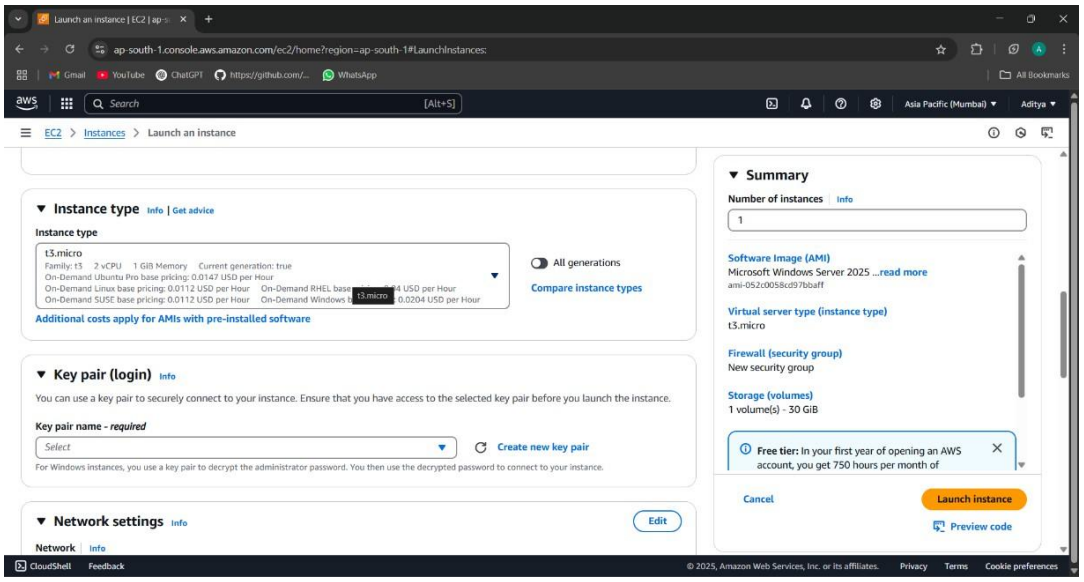
STEP3: CHOOSE THE OS-WINDOWS



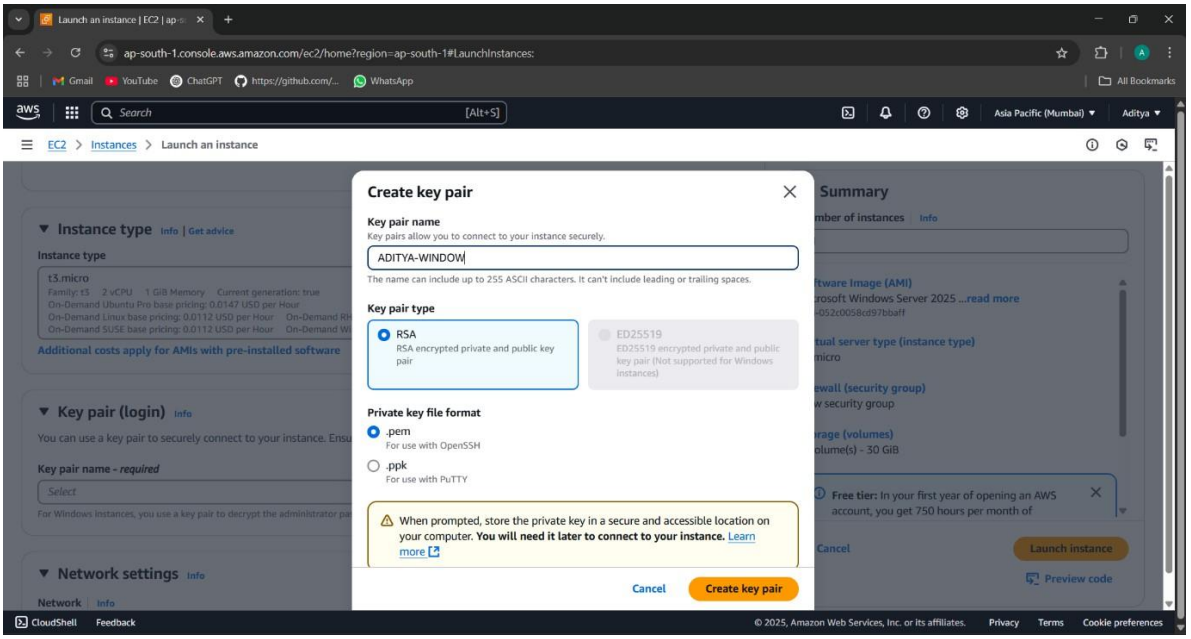
STEP4: CHOOSE THE LOCAL REGION



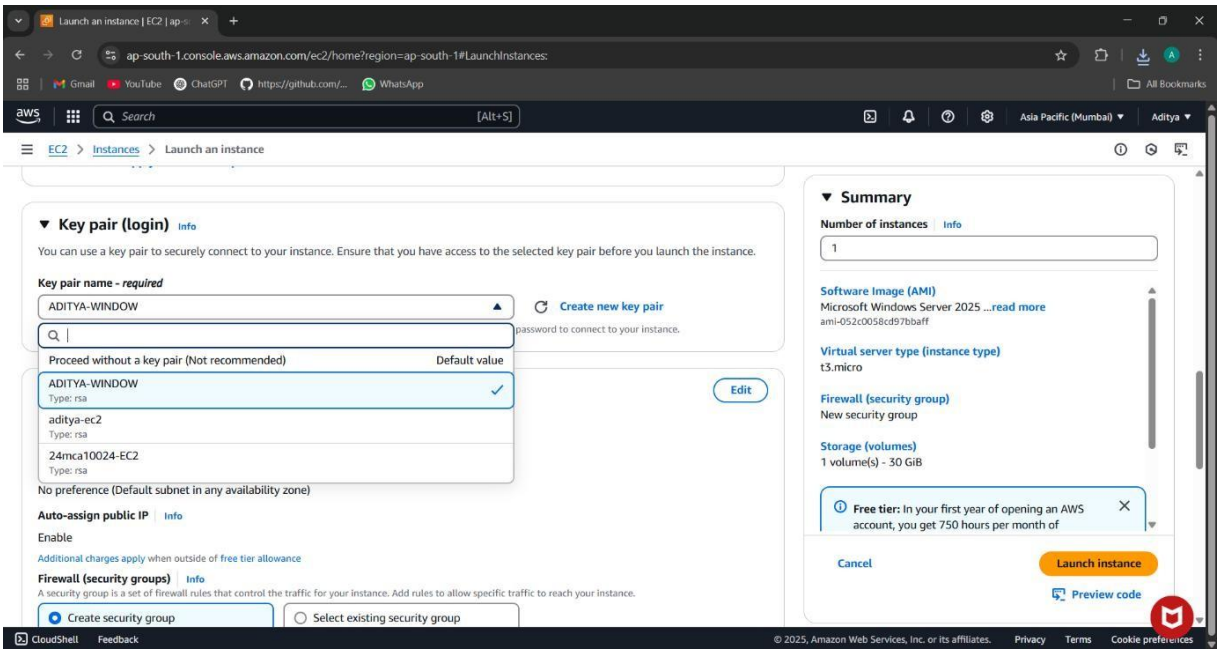
STEP5: CHOOSE THE INSTANCE TYPE AS t3 micro



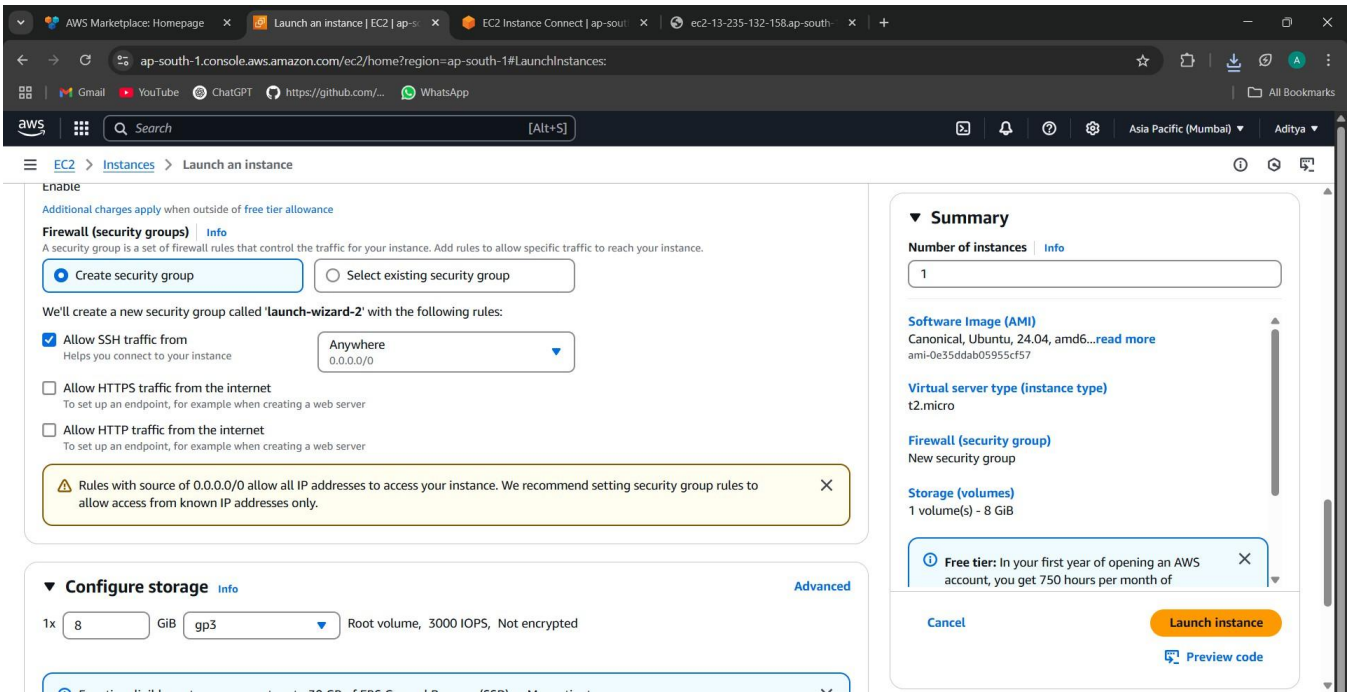
STEP6: CREATE THE KEY PAIR



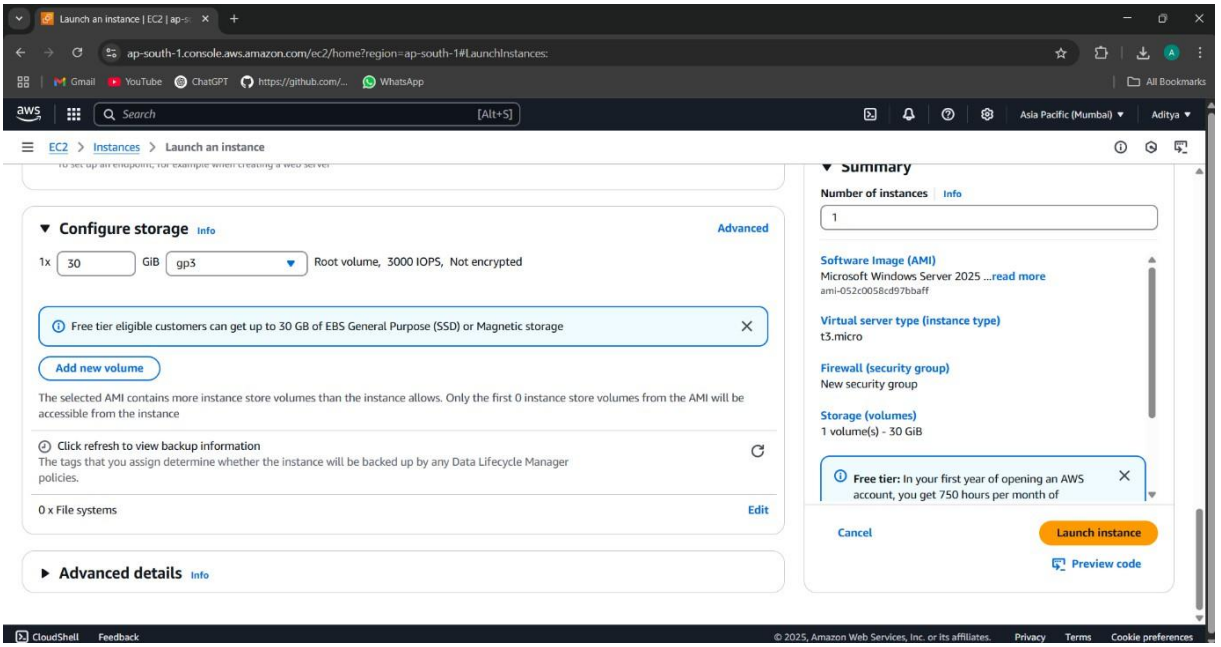
STEP7: CHOOSE THE PUBLIC KEY THAT YOU HAVE CREATED FROM THE DROP DOWN



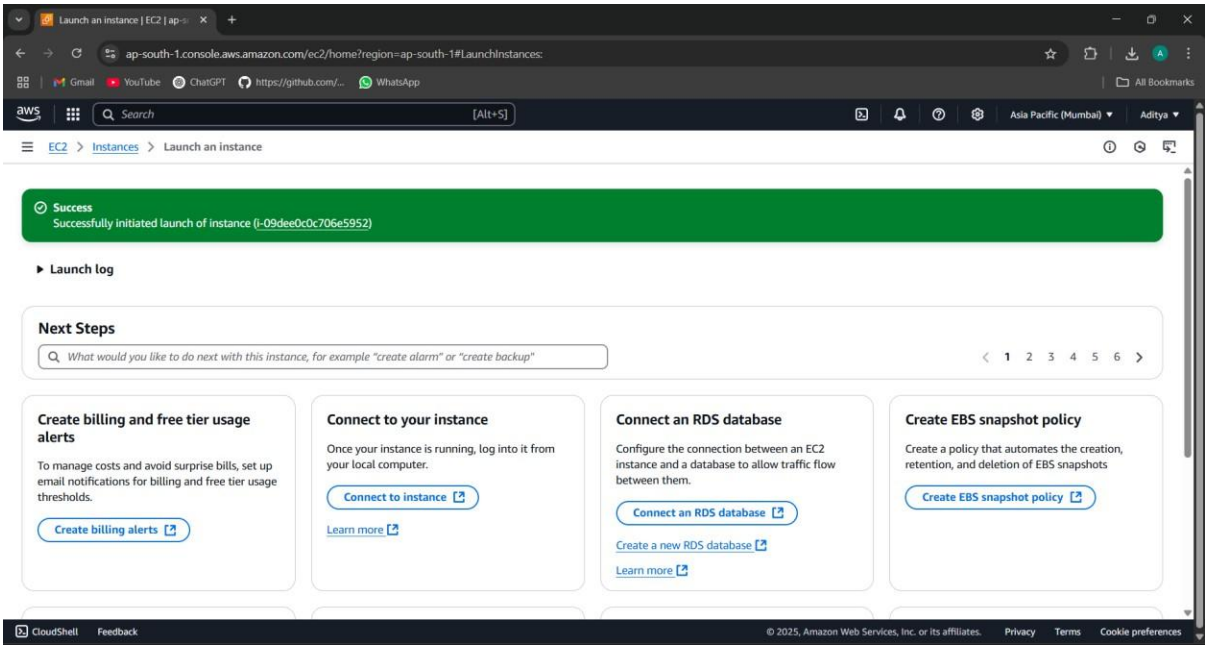
STEP8: CHOOSE FROM NETWORK SETTINGS> CREATE SECURITY GROUP> ALLOW RDP TRAFFIC FROM> ENABLE AUTO-ASSIGN PUBLIC IP



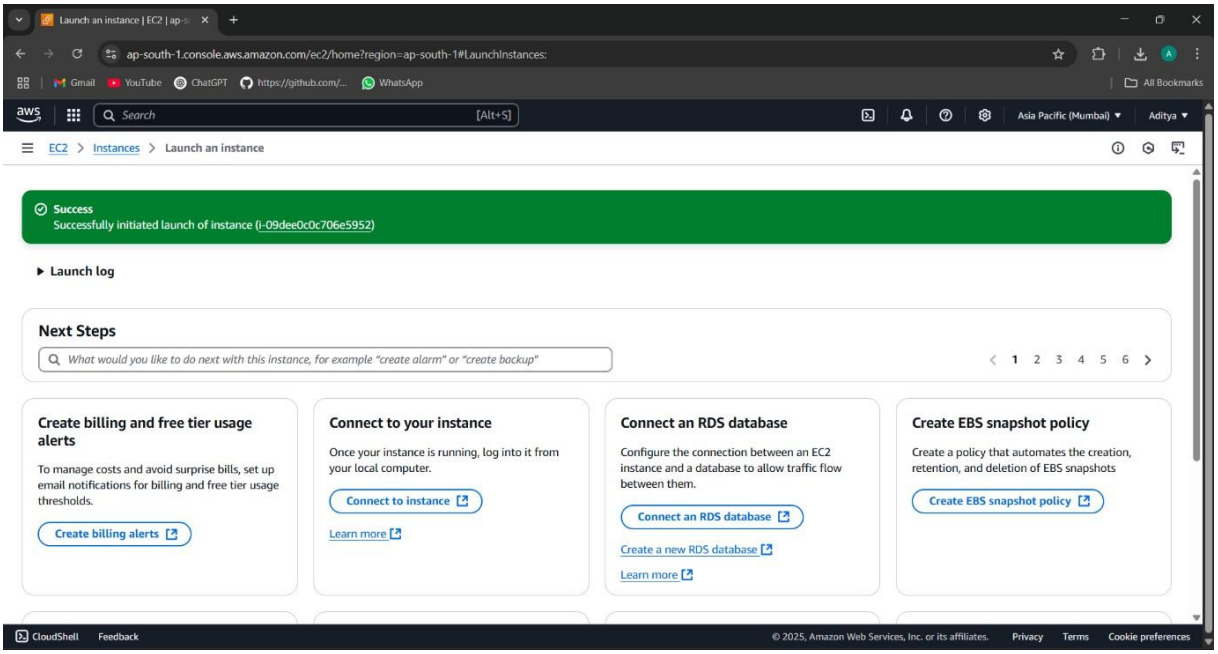
STEP9: CHOOSE 30GB FOR CONFIGURE STORAGE



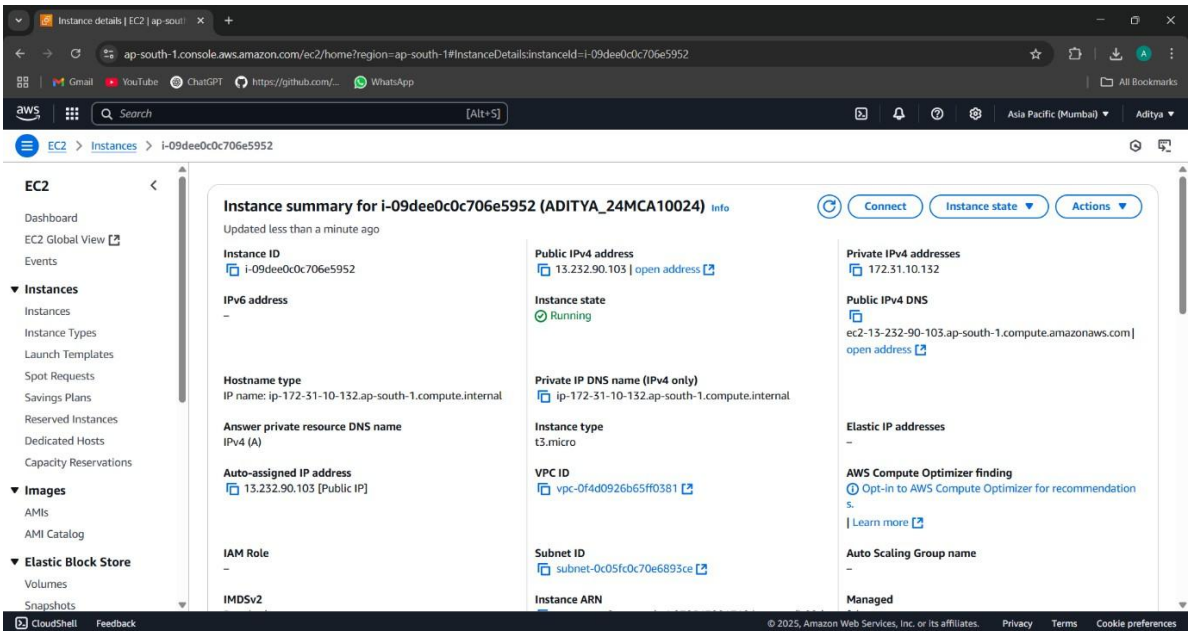
STEP10: CLICK ON LAUNCH INSTANCE



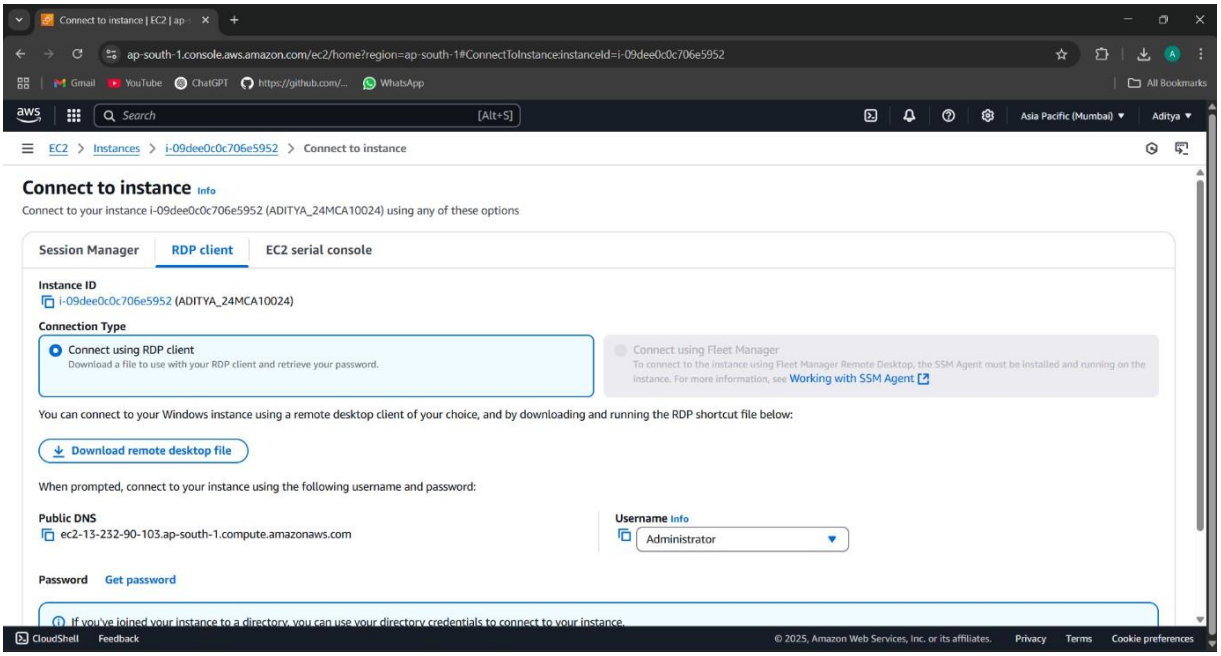
STEP11: CLICK ON THE INTANCE THAT IS CREATED



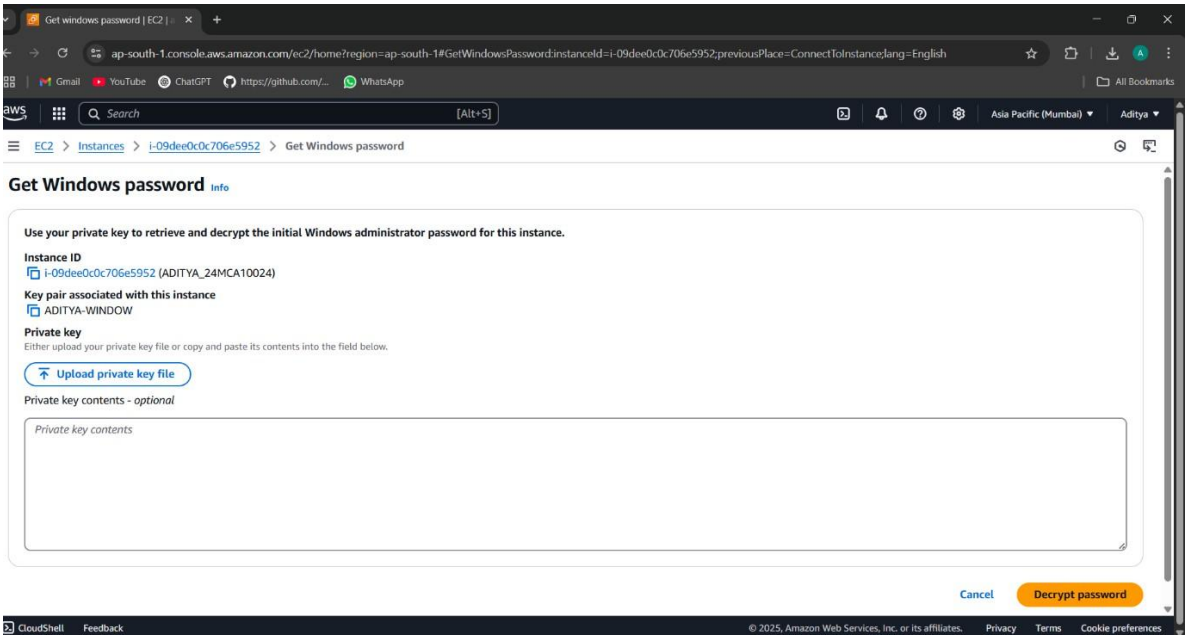
STEP12: CLICK ON CONNECT



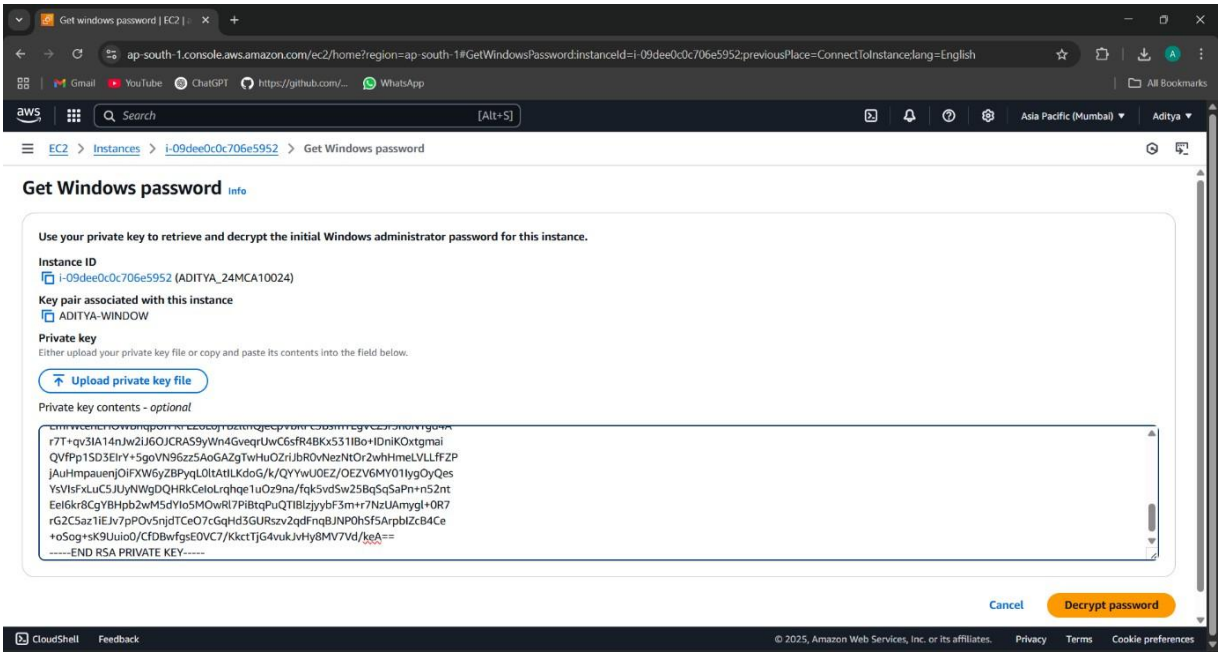
STEP13: CLICK ON RDP CLIENT>CONNECT USING RDP CLIENT> GET PASSWORD



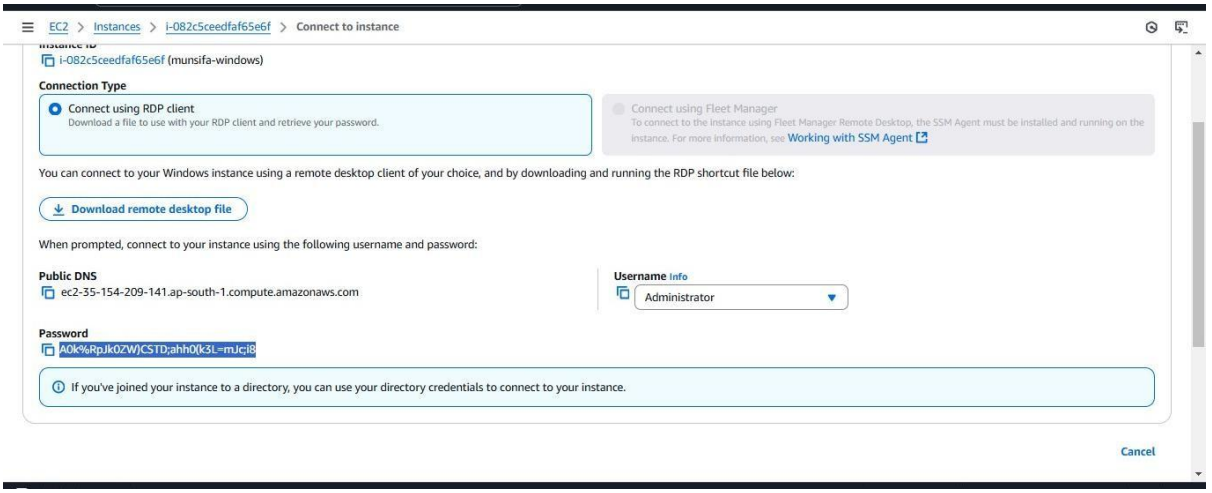
STEP14: HERE COPY & PASTE THE PRIVATE KEY THAT WAS DOWNLOADED EARLIER



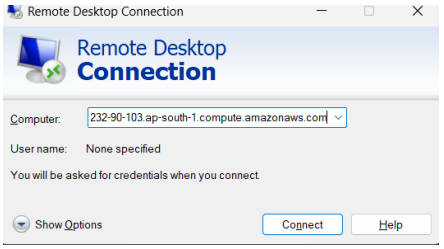
STEP15: CLICK ON DECRYPT PASSWORD



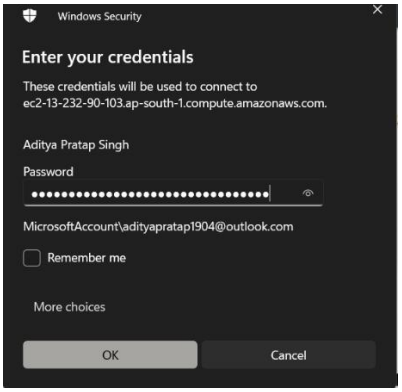
STEP16: COPY THE USERNAME, PASSWORS AND PUBLIC DNS IN A SEPARATE FILE



STEP17: ENABLE REMOTE DESKTOP IN YOUR MACHINE AND TYPE/PASTE PUBLIC DNS IN COMPUTER AND CLICK ON CONNECT



STEP18: TYPE USER NAME AND PASSWORD THAT IS SAVED EARLIER AND CLICK OK. YOUR REMOTE DESKTOP MACHINE FOR WINDOWS WILL OPEN



Date: 23/05/2025	Title
Exp. No: 03	How to launch an EC2 Instance with Template.

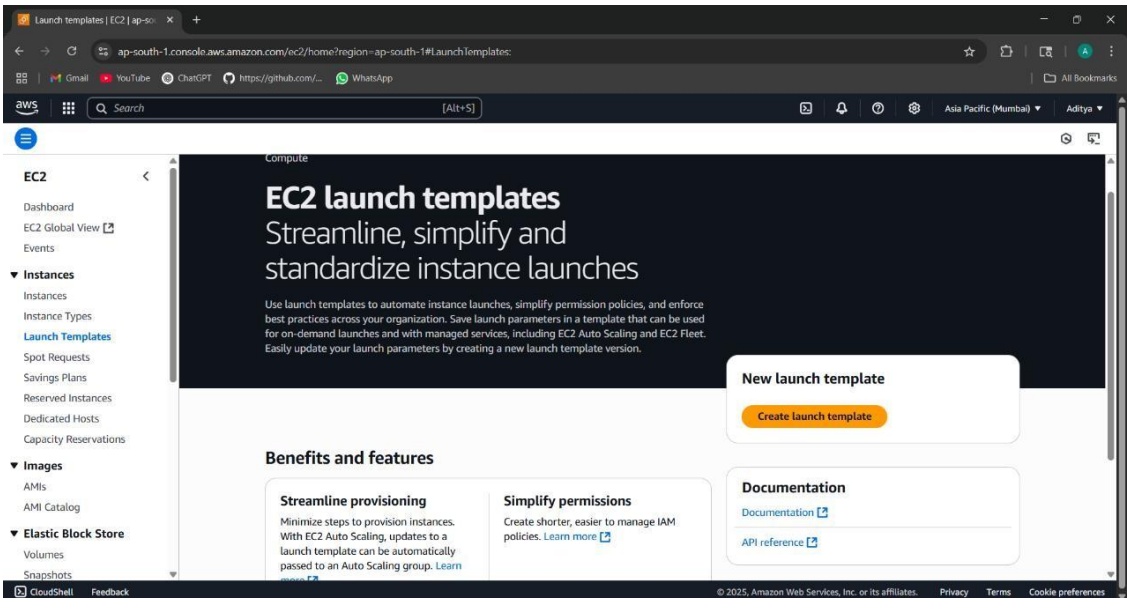
HOW TO LAUNCH AN EC2 INSTANCE WITH TEMPLATE

A **Launch Template** in AWS EC2 is a reusable configuration that defines the parameters needed to launch an instance, such as the Amazon Machine Image (AMI), instance type, key pair, security groups, network settings, and storage volumes. Using a launch template simplifies and standardizes the instance creation process, ensuring consistency across multiple deployments.

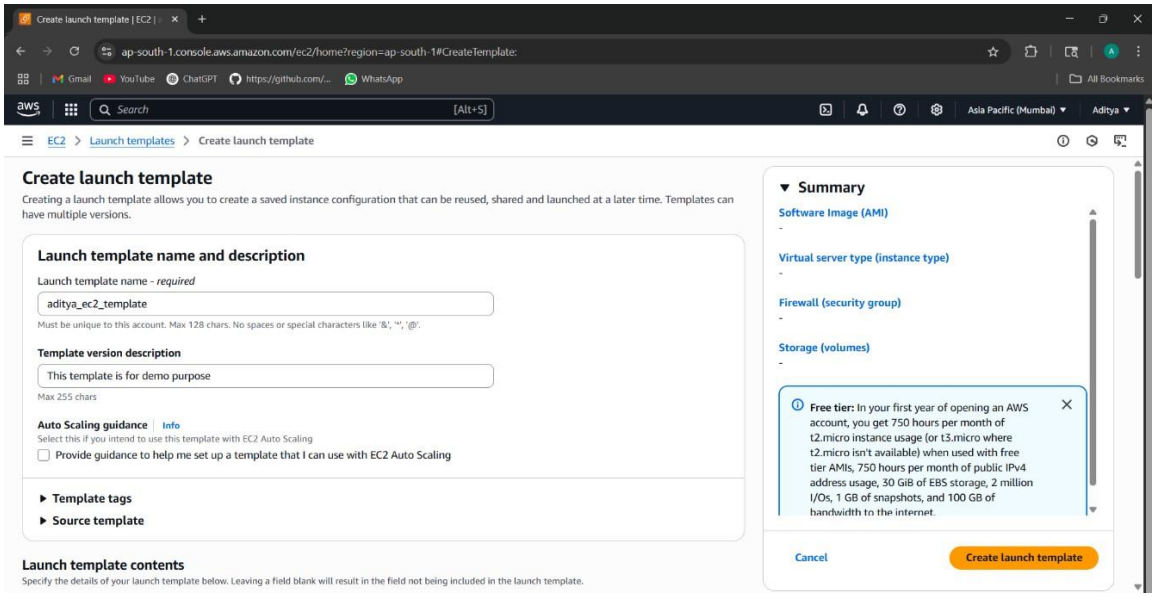
To launch an EC2 instance using a launch template, users first create the template by specifying the desired configuration. This includes details like the operating system, instance size, user data scripts, and associated IAM roles. Once the template is created, it can be used to launch new instances directly from the EC2 dashboard, AWS CLI, or as part of an Auto Scaling Group.

This method is particularly useful in environments that require repeatable and automated instance deployments. It reduces the risk of human error, saves time, and supports versioning, allowing updates and rollbacks to instance configurations when needed. Launch templates are commonly used in production environments, CI/CD pipelines, and when working with scaling solutions in AWS.

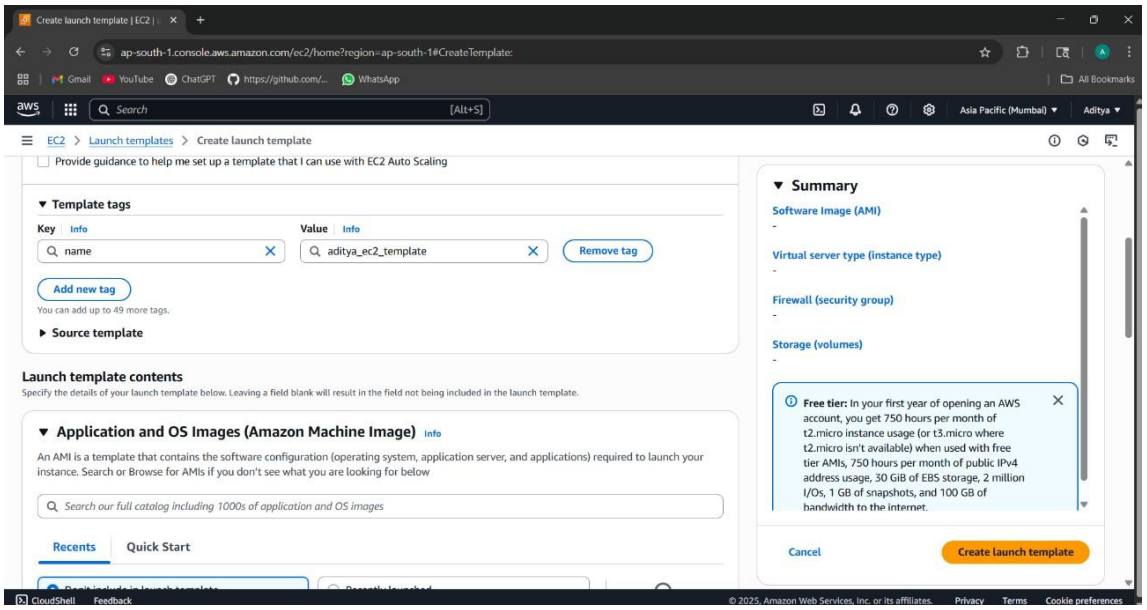
STEP1: OPEN THE EC2 DASHBOARD AND CLICK ON LAUNCH TEMPLATES>CREATE LAUNCH TEMPLATE



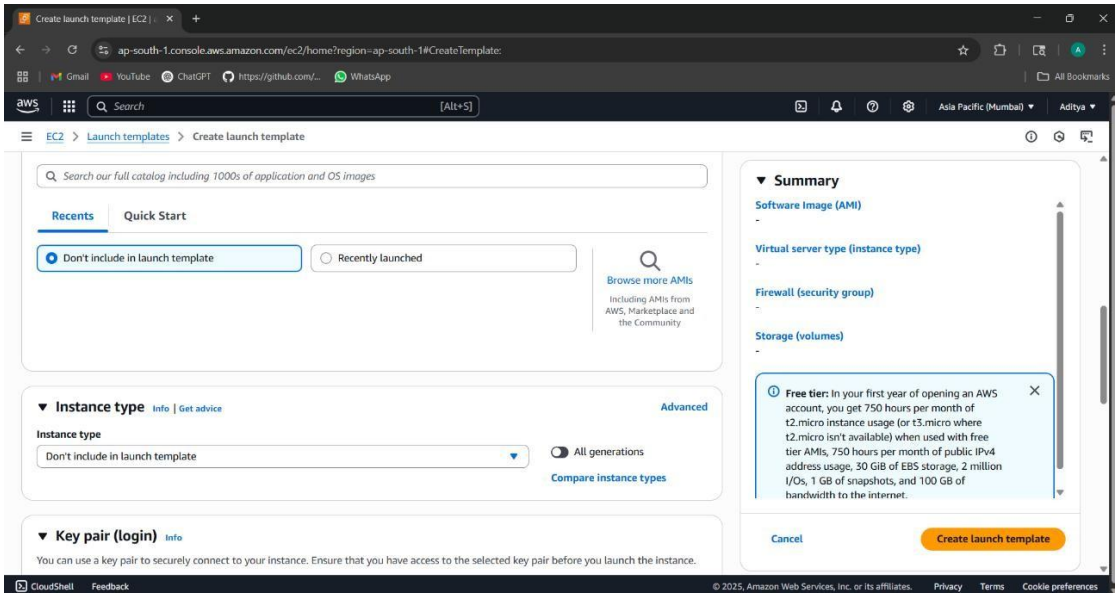
STEP2: GIVE A SUITABLE NAME AND DESCRIPTION TO THE TEMPLATE


Create launch template | EC2 |
ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#CreateTemplate:
Gmail
YouTube
ChatGPT
https://github.com/...
WhatsApp
All Bookmarks
aws
Search
[Alt+S]
Asia Pacific (Mumbai)
Aditya
EC2 > Launch templates > Create launch template
Create launch template
Creating a launch template allows you to create a saved instance configuration that can be reused, shared and launched at a later time. Templates can have multiple versions.
Launch template name and description
Launch template name - required
aditya_ec2_template
Must be unique to this account. Max 128 chars. No spaces or special characters like '&', '*', '@'.
Template version description
This template is for demo purpose
Max 255 chars
Auto Scaling guidance | Info
Select this if you intend to use this template with EC2 Auto Scaling
Provide guidance to help me set up a template that I can use with EC2 Auto Scaling
Template tags
Source template
Launch template contents
Specify the details of your launch template below. Leaving a field blank will result in the field not being included in the launch template.
Summary
Software Image (AMI)
-
Virtual server type (instance type)
-
Firewall (security group)
-
Storage (volumes)
-
Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.
Cancel
Create launch template

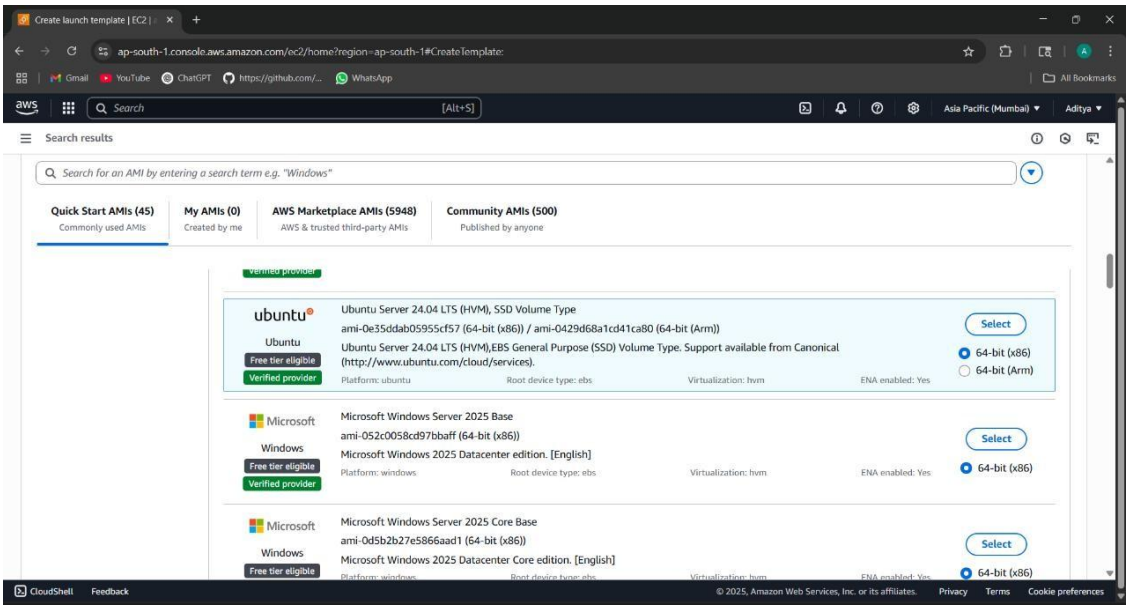
STEP3: CLICK ON TEMPLATE TAGS>ADD TAGS>INFO:NAME>VALUE:ANY NAME



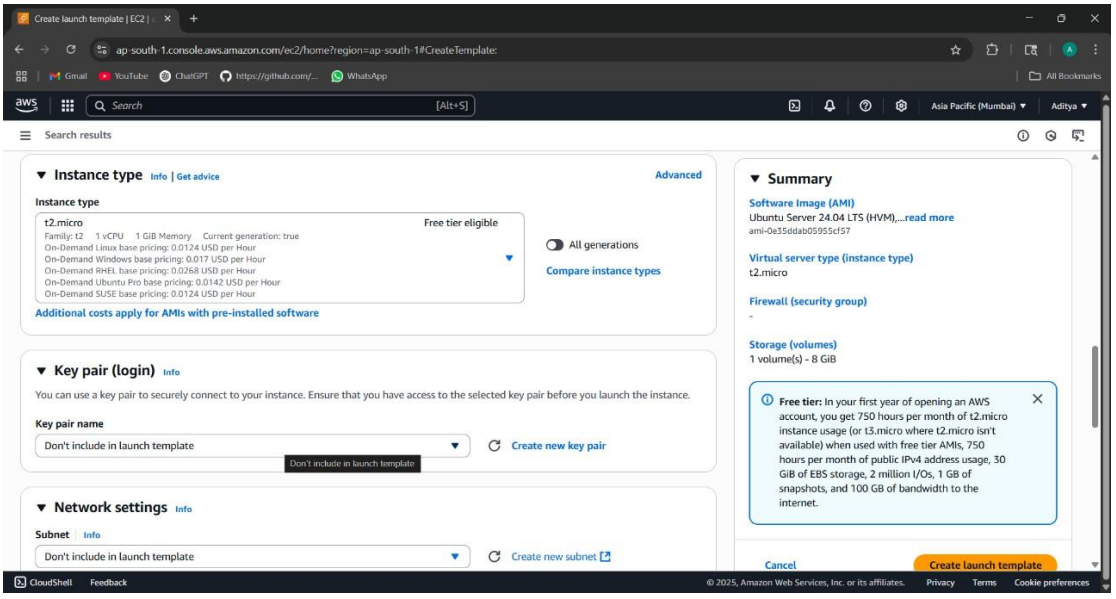
STEP4: CLICK ON BROWSE MORE AMI



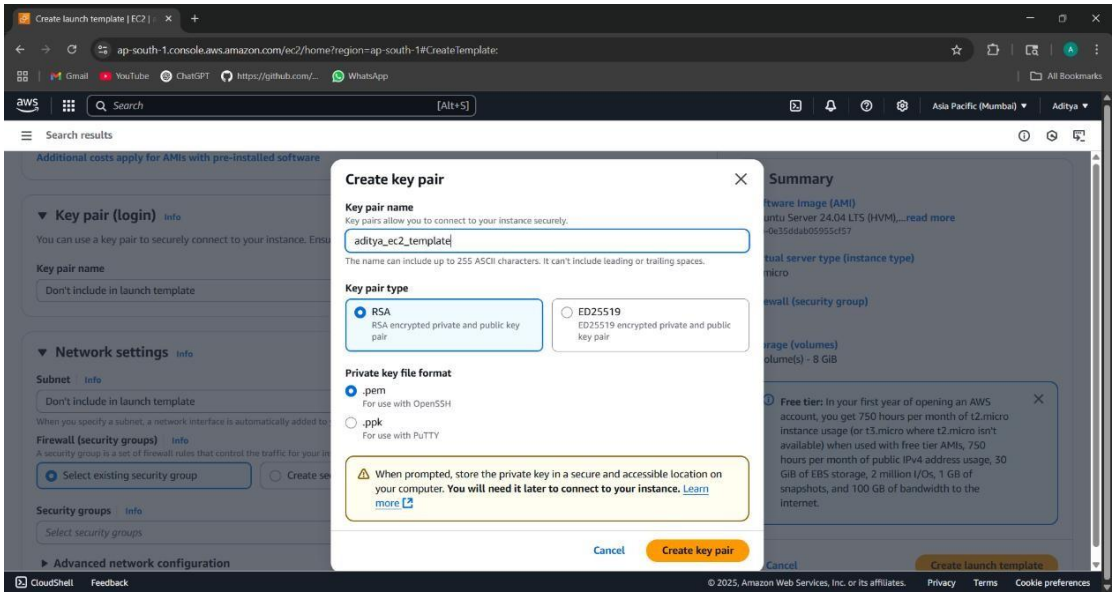
STEP5: SELECT UBUNTU FROM THE LIST



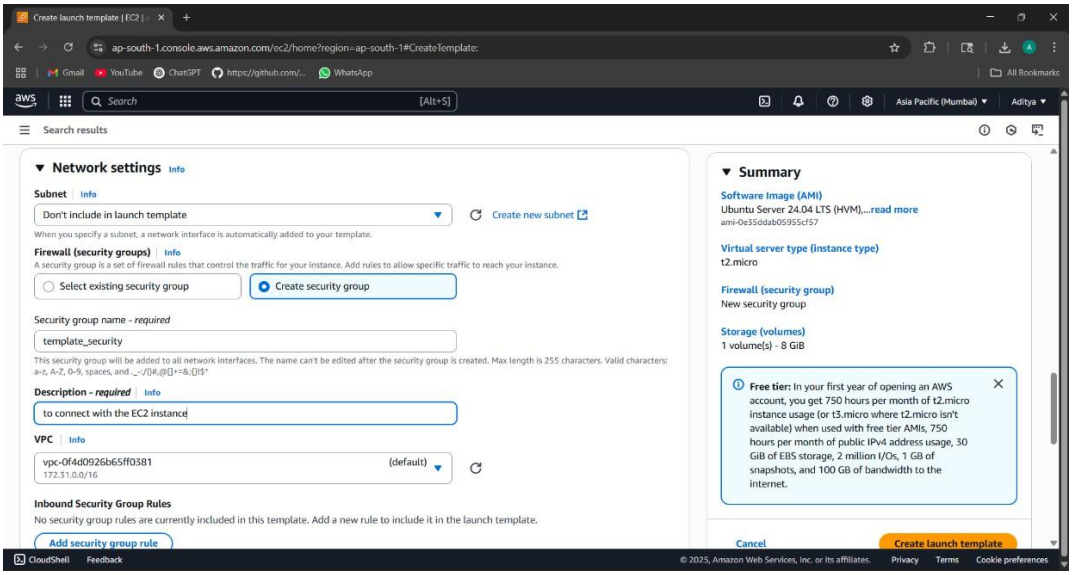
STEP6: SELECT THE INSTANCE TYPE:t2 micro (FREE TIER)



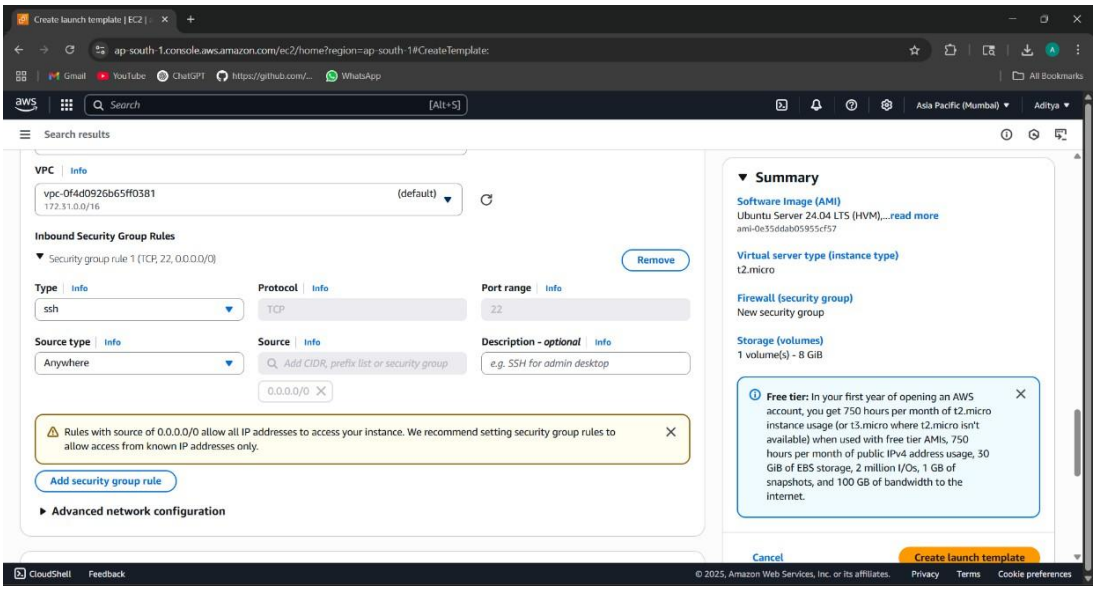
STEP7: CREATE KEY PAIR



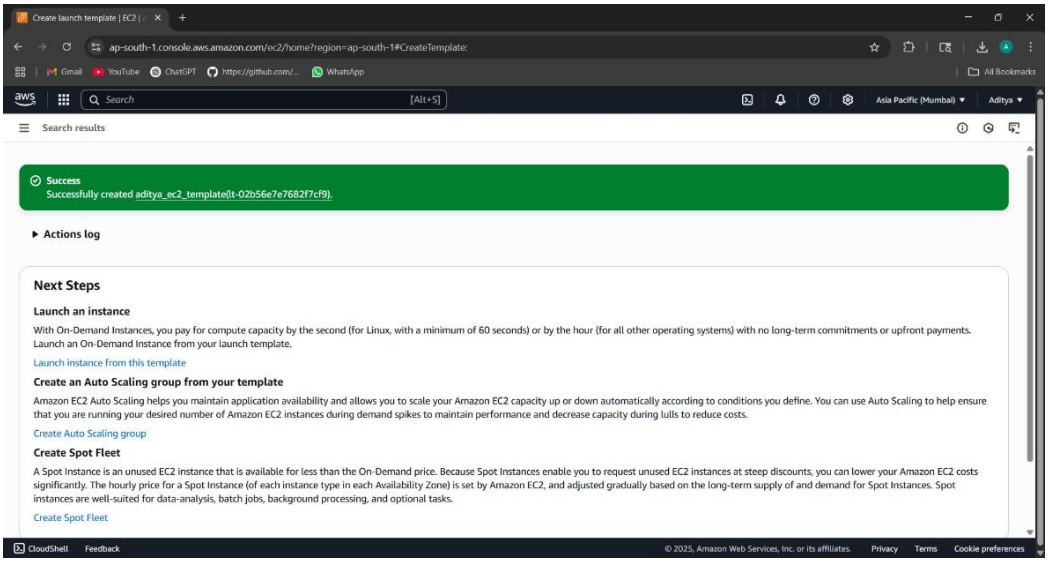
STEP8: SELECT CREATE SECURITY GROUP> WRITE THE SECURITY GROUP NAME>DESCRIPTION> CLICK ON ADD SECURITY GROUP RULE

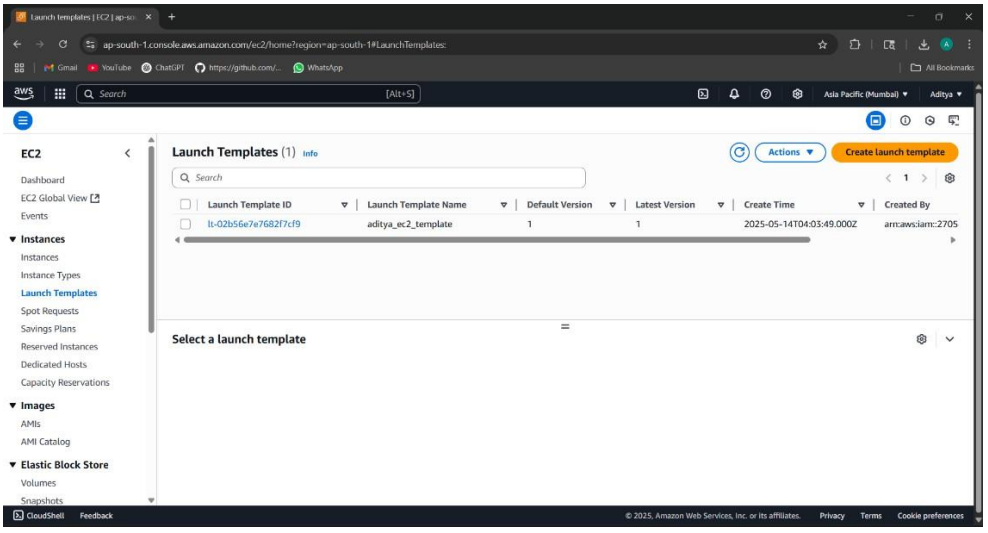


STEP9: CHOOSE TYPE:SSH> SOURCE TYPE: ANYWHERE

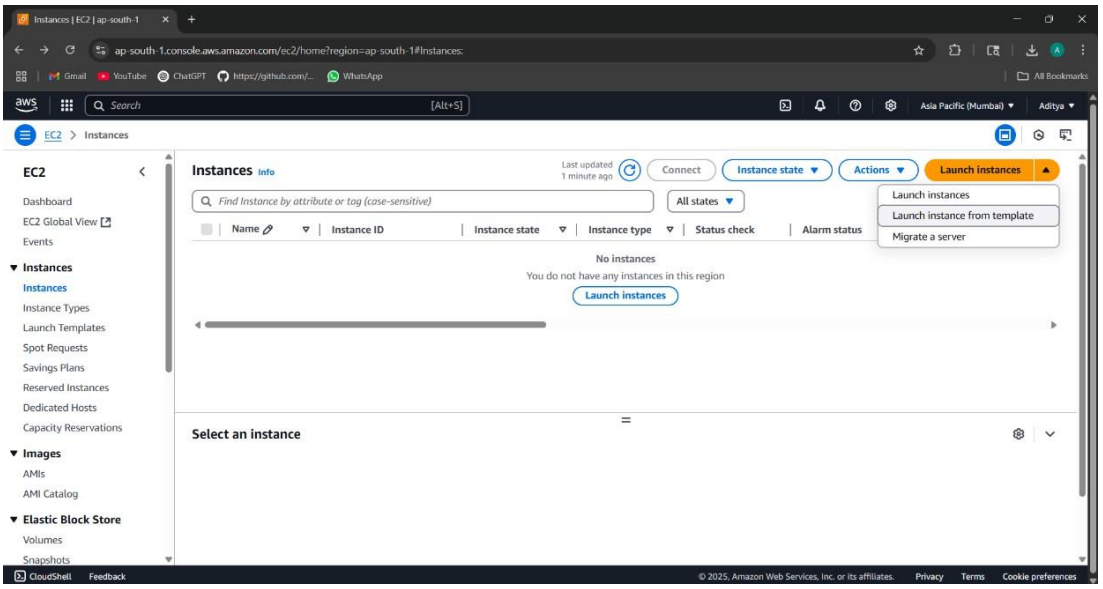


STEP 10: CLICK ON CREATE LAUNCH>LAUNCH TEMPLATE IS CREATED>CLICK ON VIEW TEMPLATE

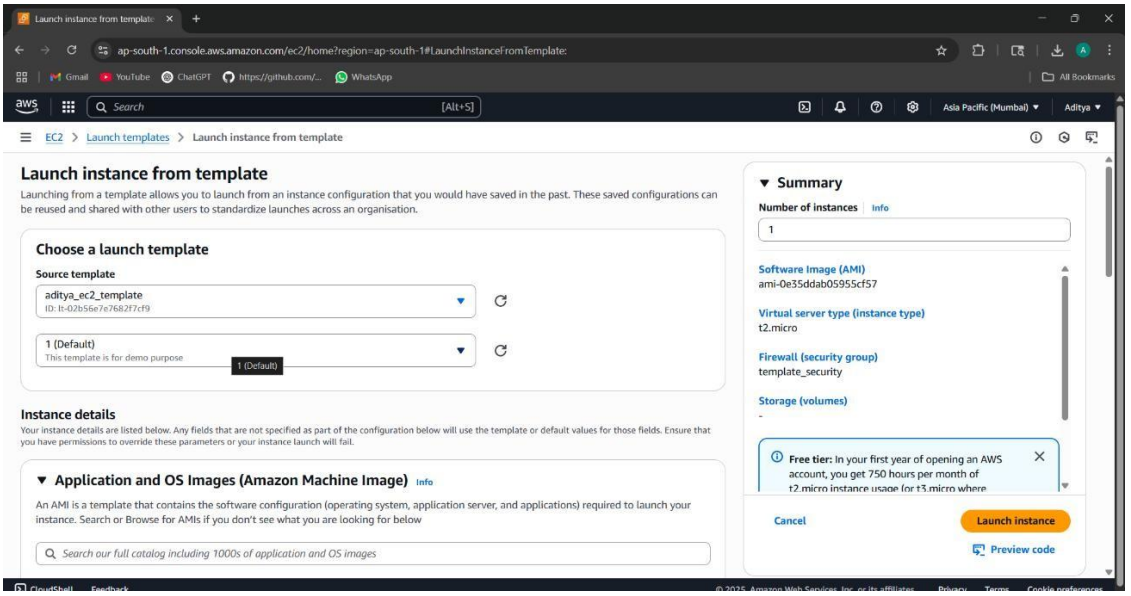




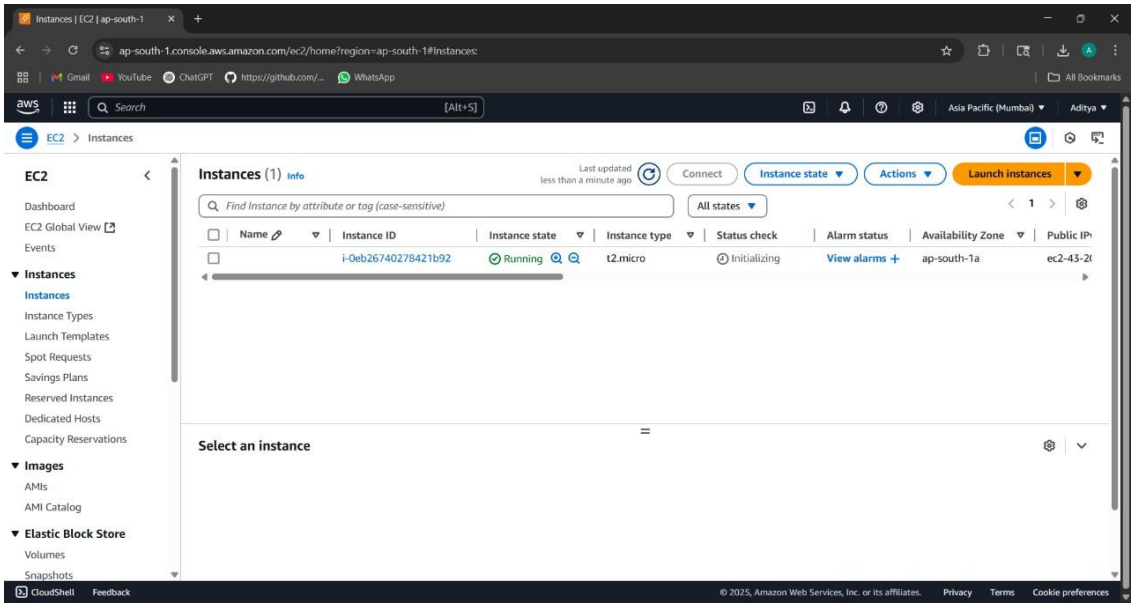
STEP 10: CLICK ON INSTANCES >CLICK ON LAUNCH INSTANCES>LAUNCH INSTANCE FROM TEMPLATE



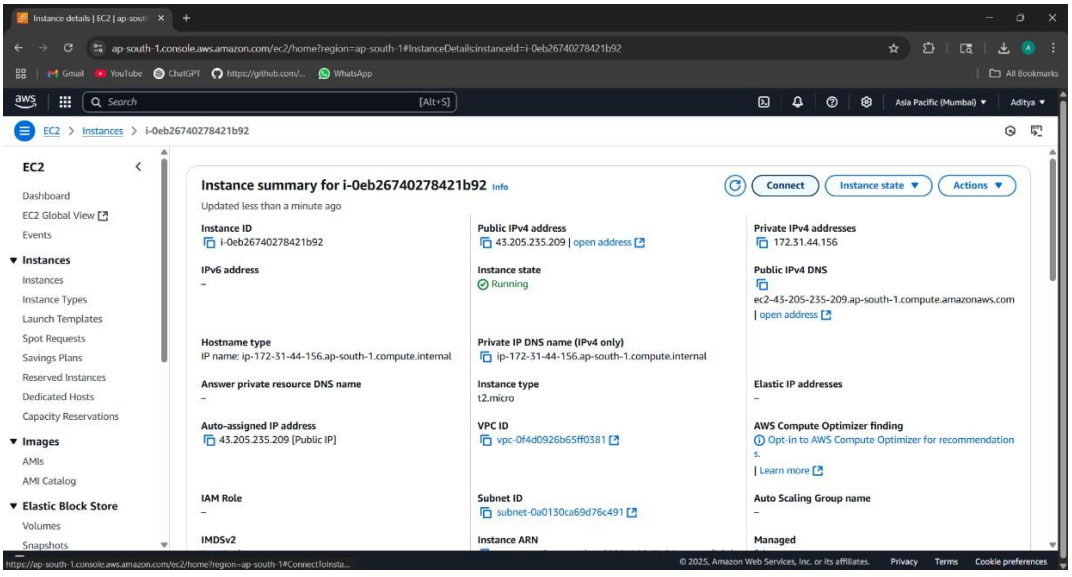
STEP 12: SELECT THE TEMPLATE THAT YOU HAVE CREATED>CLICK ON LAUNCH INSTANCE



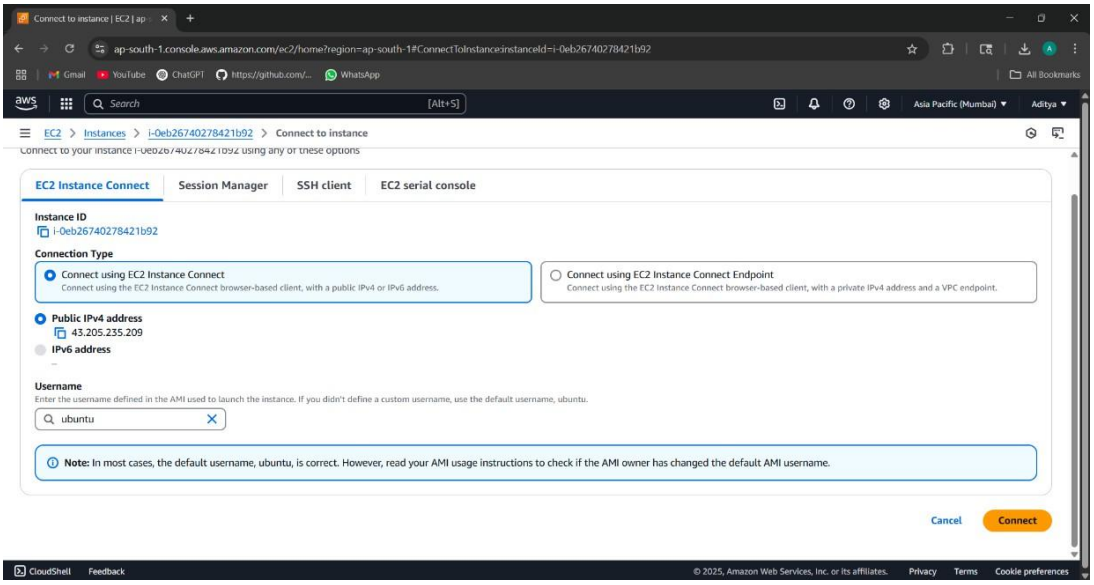
STEP 13: INSTANCE IS CREATED>CLICK ON THE INSTANCE THAT YOU HAVE CREATED



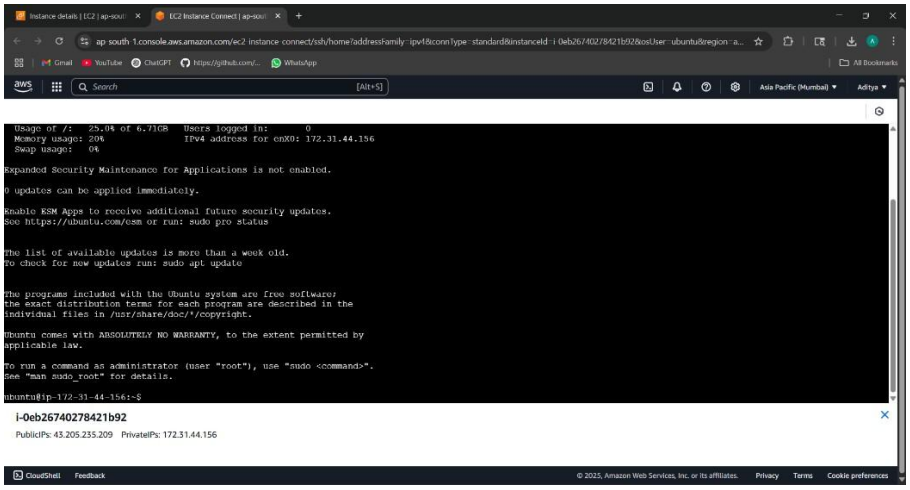
STEP 14: CLICK ON CONNECT



STEP 15: CLICK ON CONNECT TO INSTANCE



STEP 16: EC2 INSTANCE IS CONNECTED

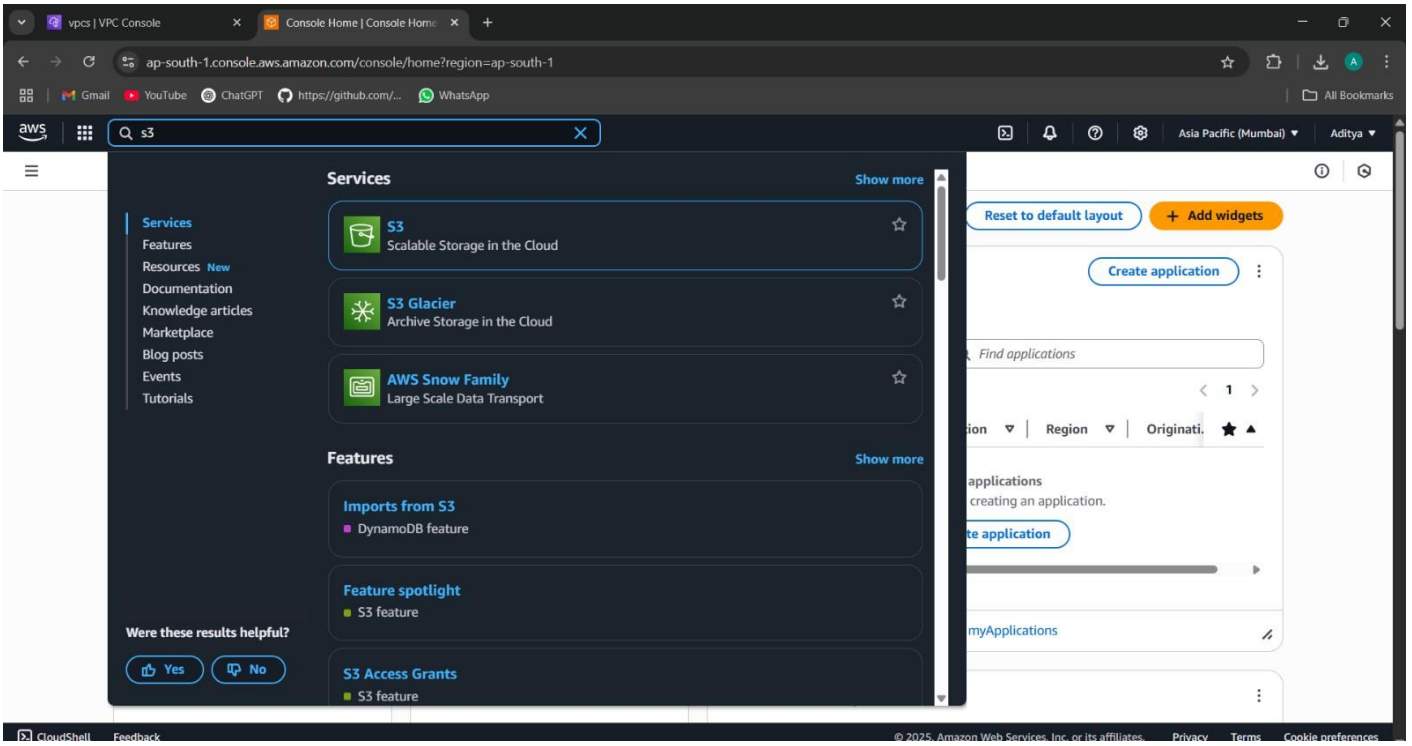


Date: 23/05/2025	Title
Exp. No: 04	Amazon S3 (Bucket Creation, Creating URL and S3 Life Cycle Management.

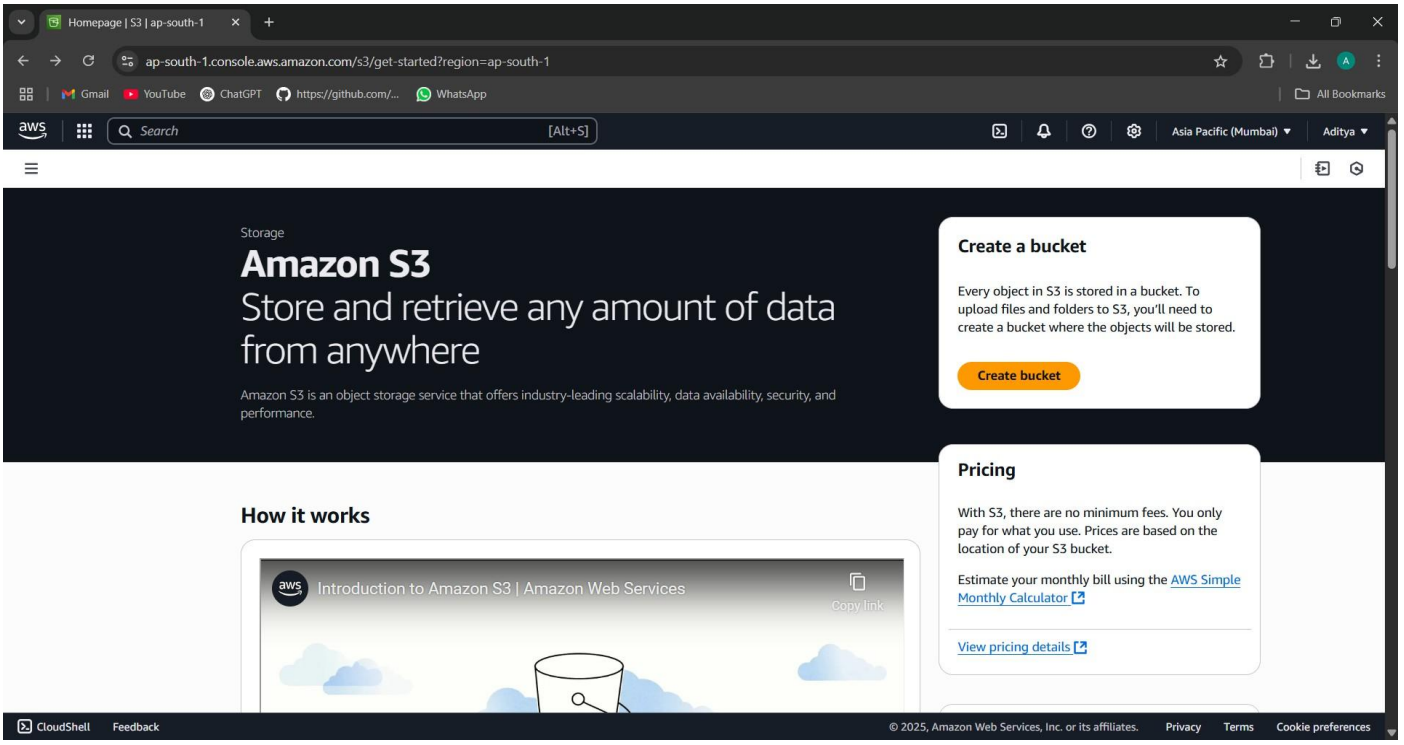
Amazon S3 (Bucket creation, creating URL and S3 Life cycle Management)

Amazon S3 (Simple Storage Service) is a cloud-based object storage service used to store and retrieve files. Users begin by creating a bucket, which serves as a container for their data. After uploading files, S3 generates unique URLs that allow access to the objects, either publicly or through secure, time-limited links. Lifecycle management lets users automate actions like moving files to cheaper storage classes or deleting them after a set time, helping to optimize storage costs and manage data efficiently.

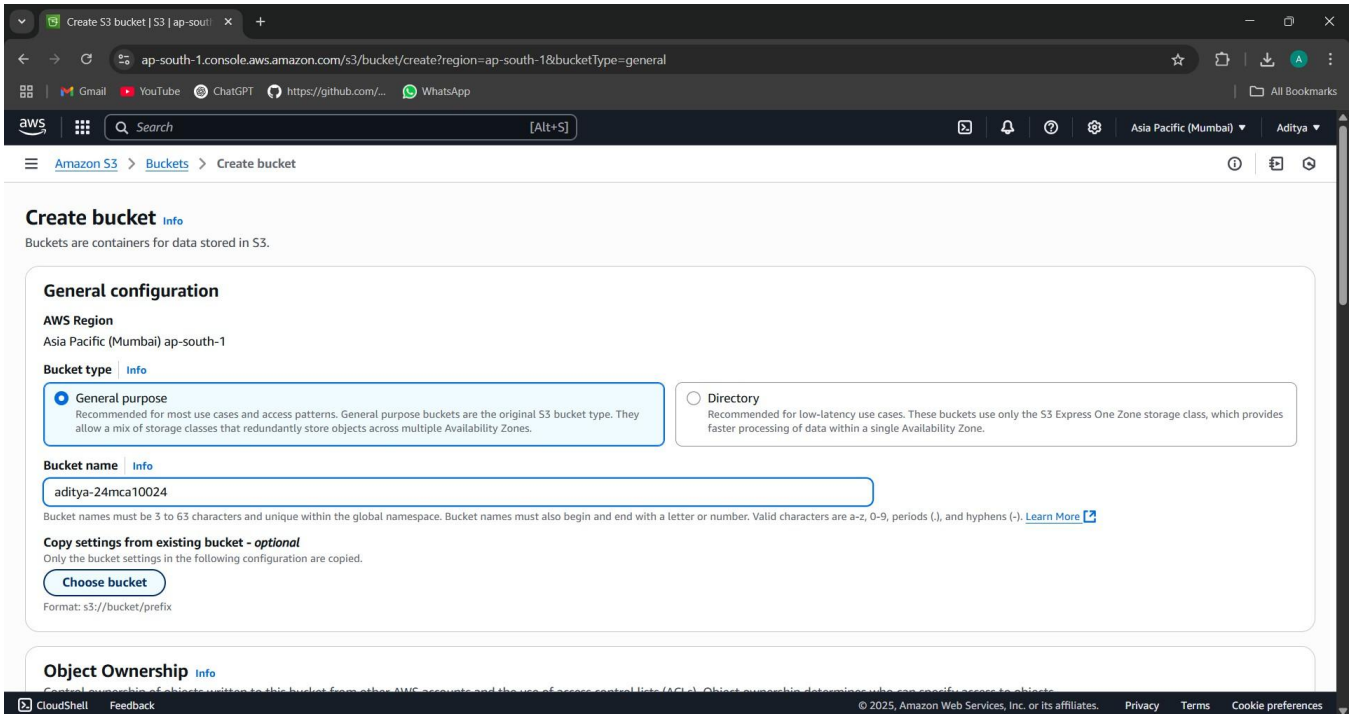
Step 1: -Get in storage and then in storage dashboard.



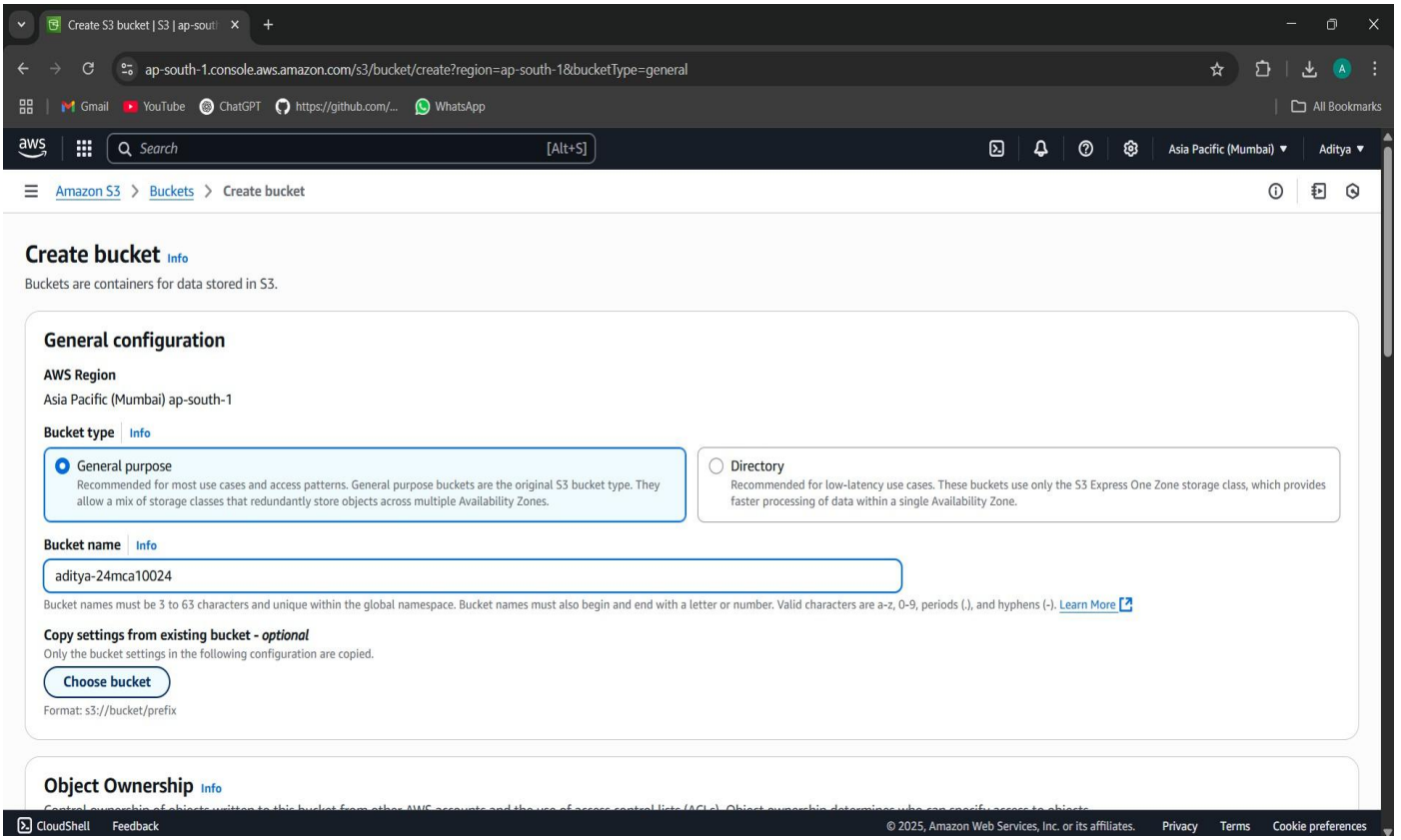
Step 2: - Since no bucket is created so click on create bucket.


Homepage | S3 | ap-south-1
ap-south-1.console.aws.amazon.com/s3/get-started?region=ap-south-1
Gmail
YouTube
ChatGPT
https://github.com/...
WhatsApp
All Bookmarks
aws
Search
[Alt+S]
Asia Pacific (Mumbai)
Aditya
Storage
Amazon S3
Store and retrieve any amount of data from anywhere
Amazon S3 is an object storage service that offers industry-leading scalability, data availability, security, and performance.
Create a bucket
Every object in S3 is stored in a bucket. To upload files and folders to S3, you'll need to create a bucket where the objects will be stored.
Create bucket
Pricing
With S3, there are no minimum fees. You only pay for what you use. Prices are based on the location of your S3 bucket.
Estimate your monthly bill using the AWS Simple Monthly Calculator
View pricing details
How it works
Introduction to Amazon S3 | Amazon Web Services
Copy link
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

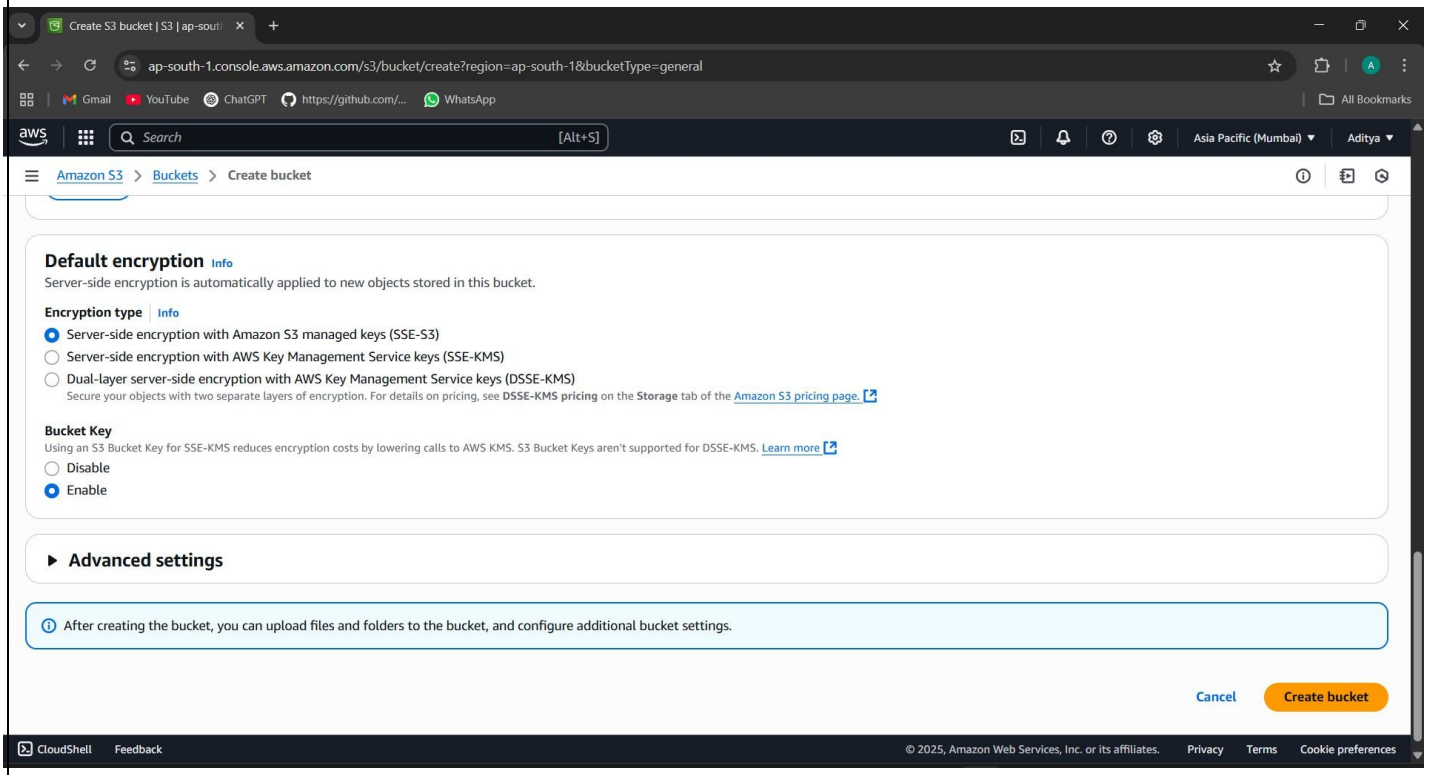
Step 3: -Since the AWS region is already selected.



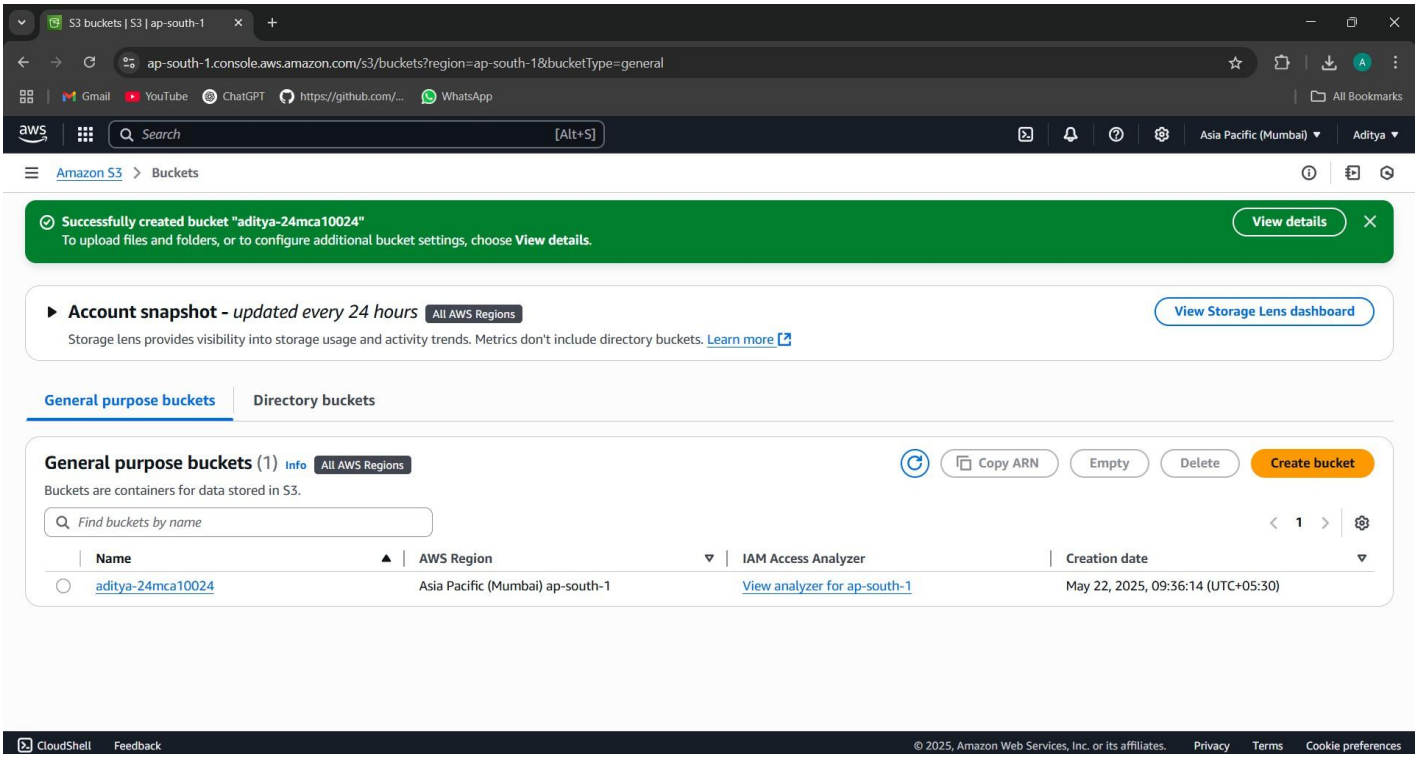
Step 4: - Now creating a bucket with unique name which does not contain caps.



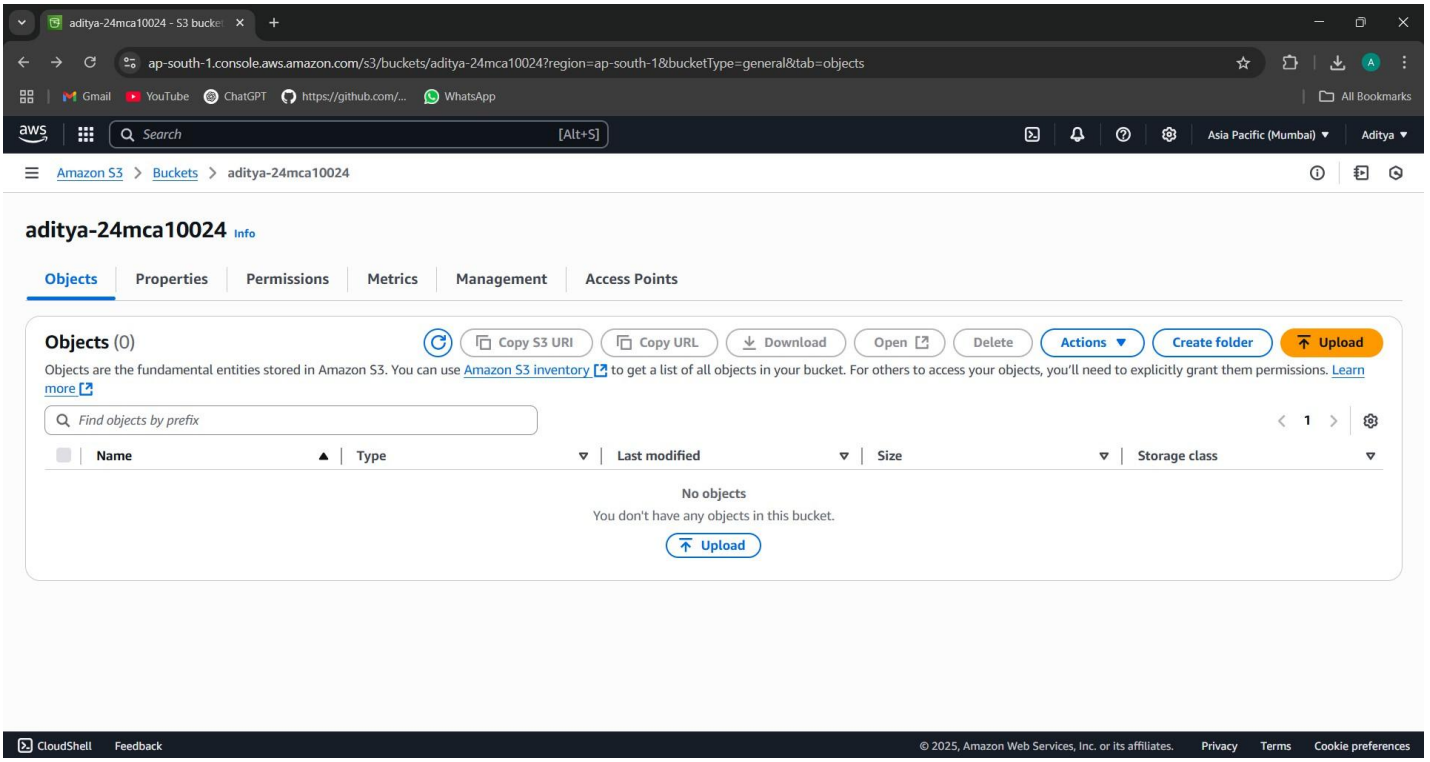
Step 5: - Now for giving the security we provide encryption to Bucket.



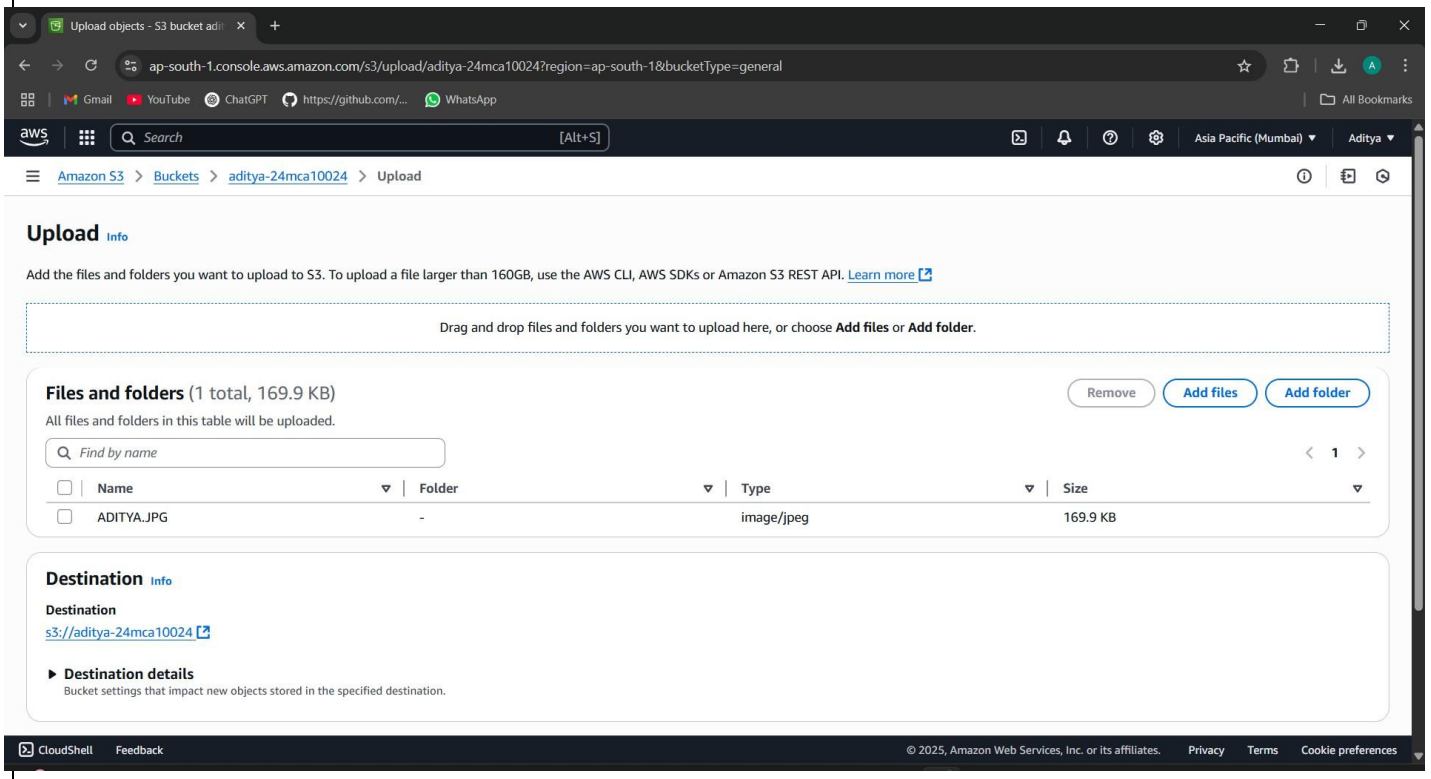
Step 6: -Now in this step bucket is created with name Aditya-24mca10024



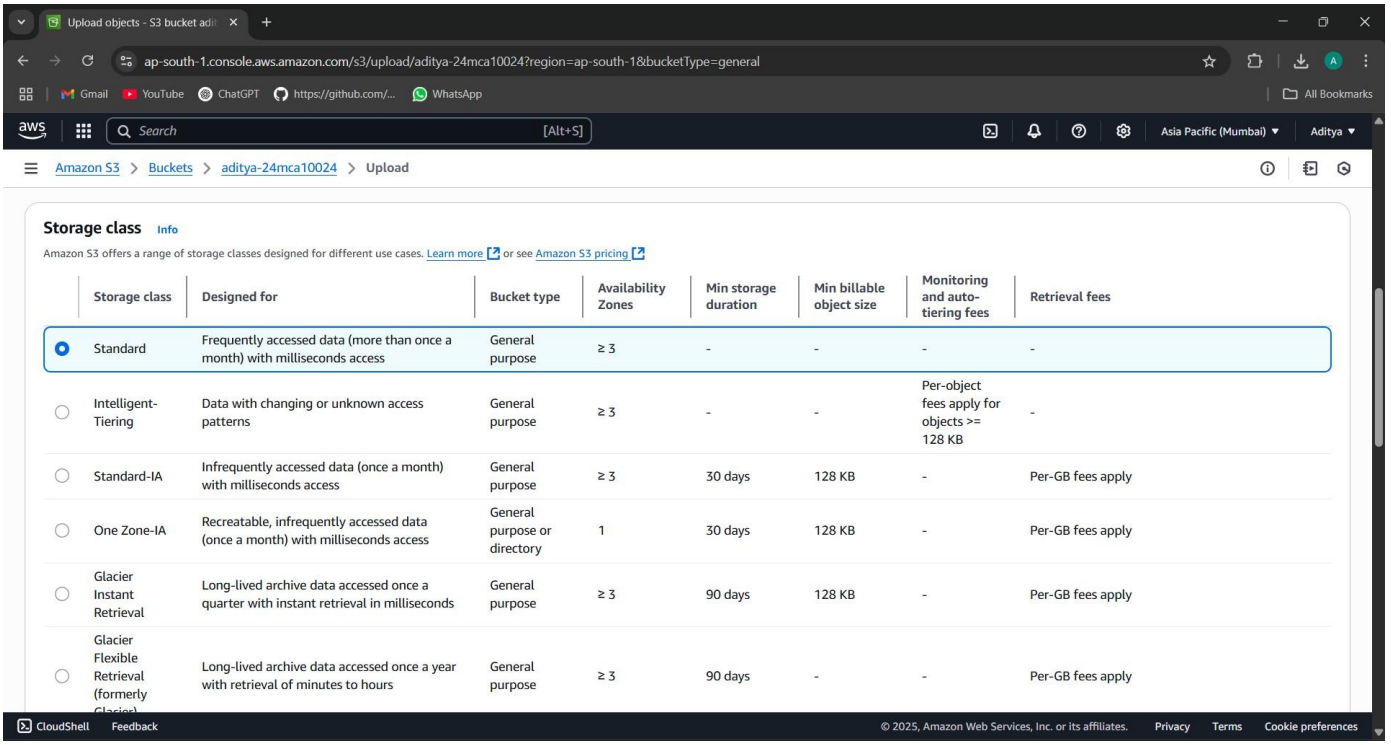
Step 7: -Now we upload file by clicking upload.



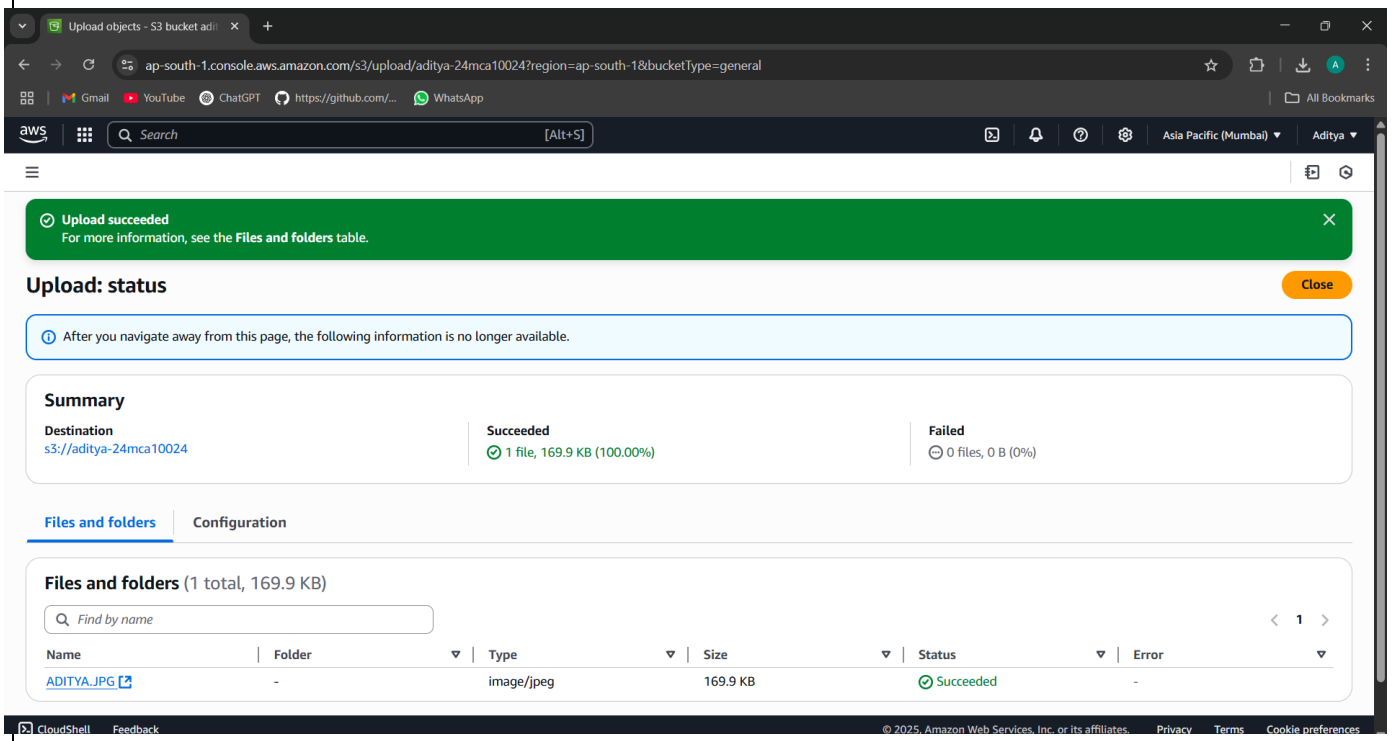
Step 8:-Now under upload section we can upload the file as well as folder.



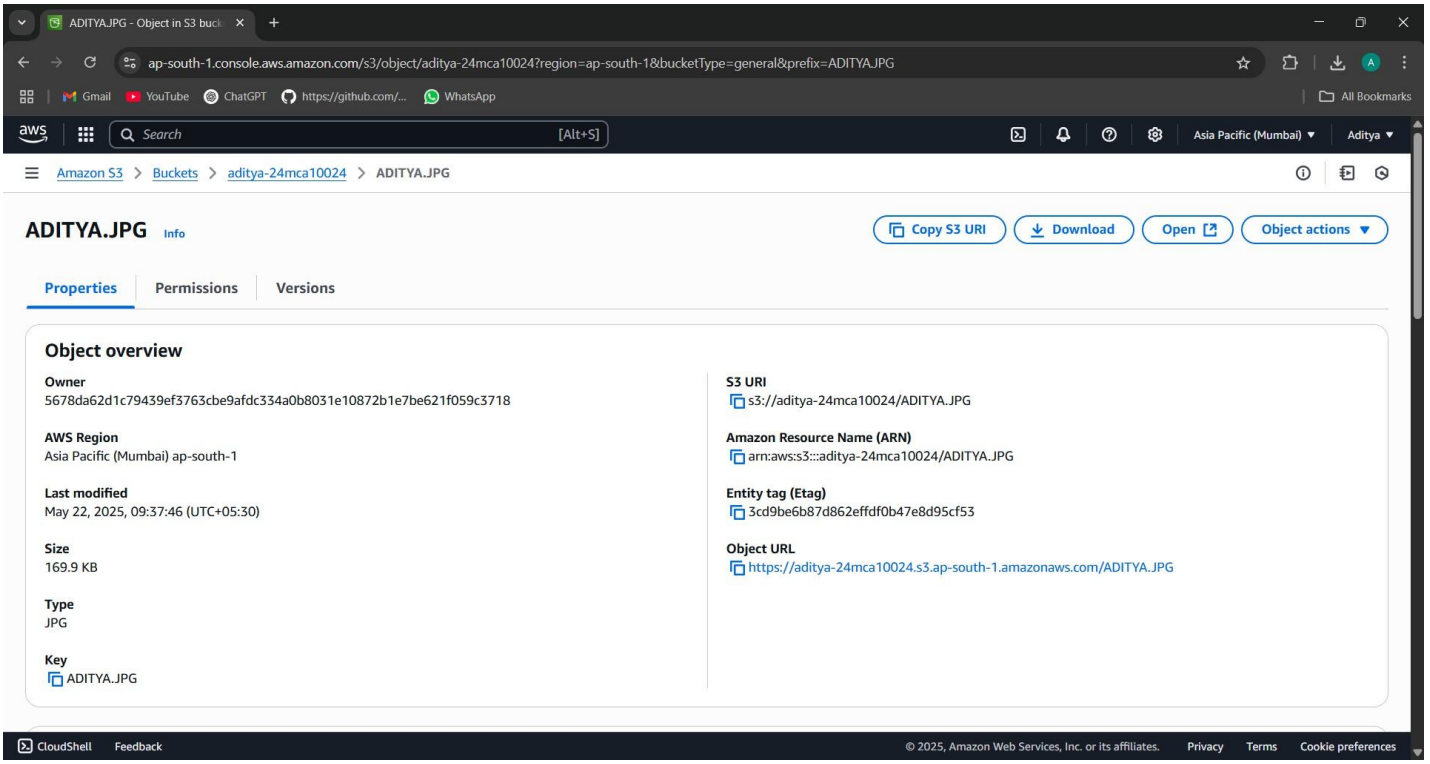
Step 9:- In storage class there are different types so choose any one as per use.



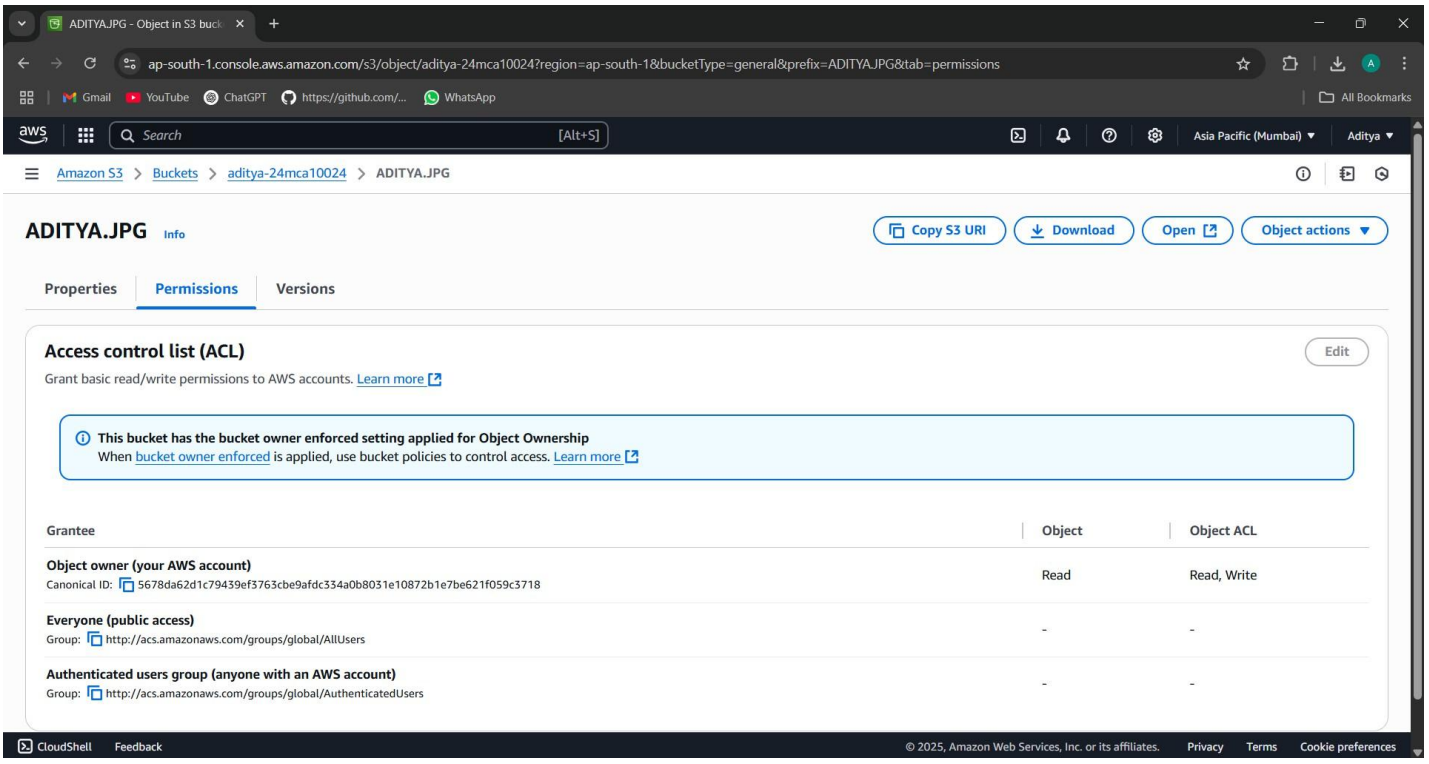
Step 10: -Now click on upload it show success.



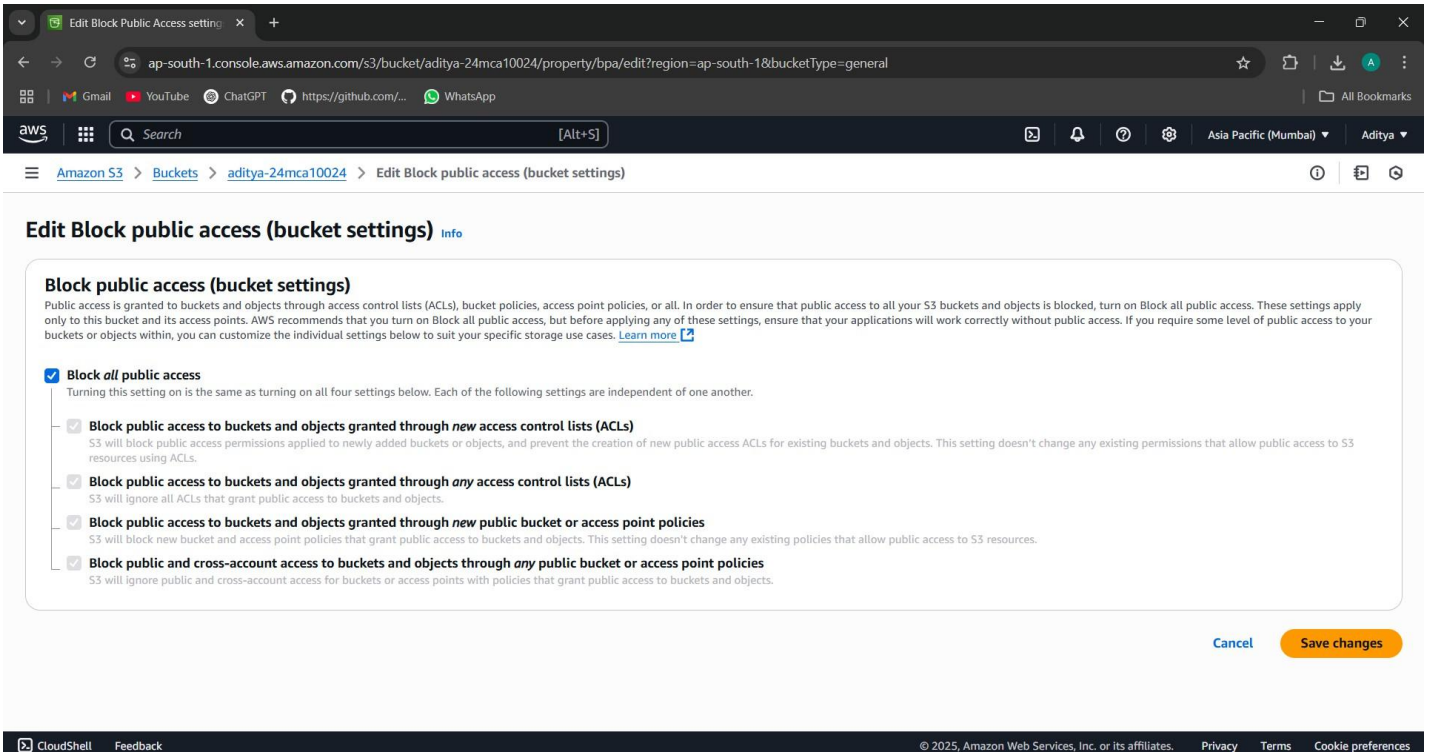
Step 11: -Now open the file and we can see everything as shown in fig.



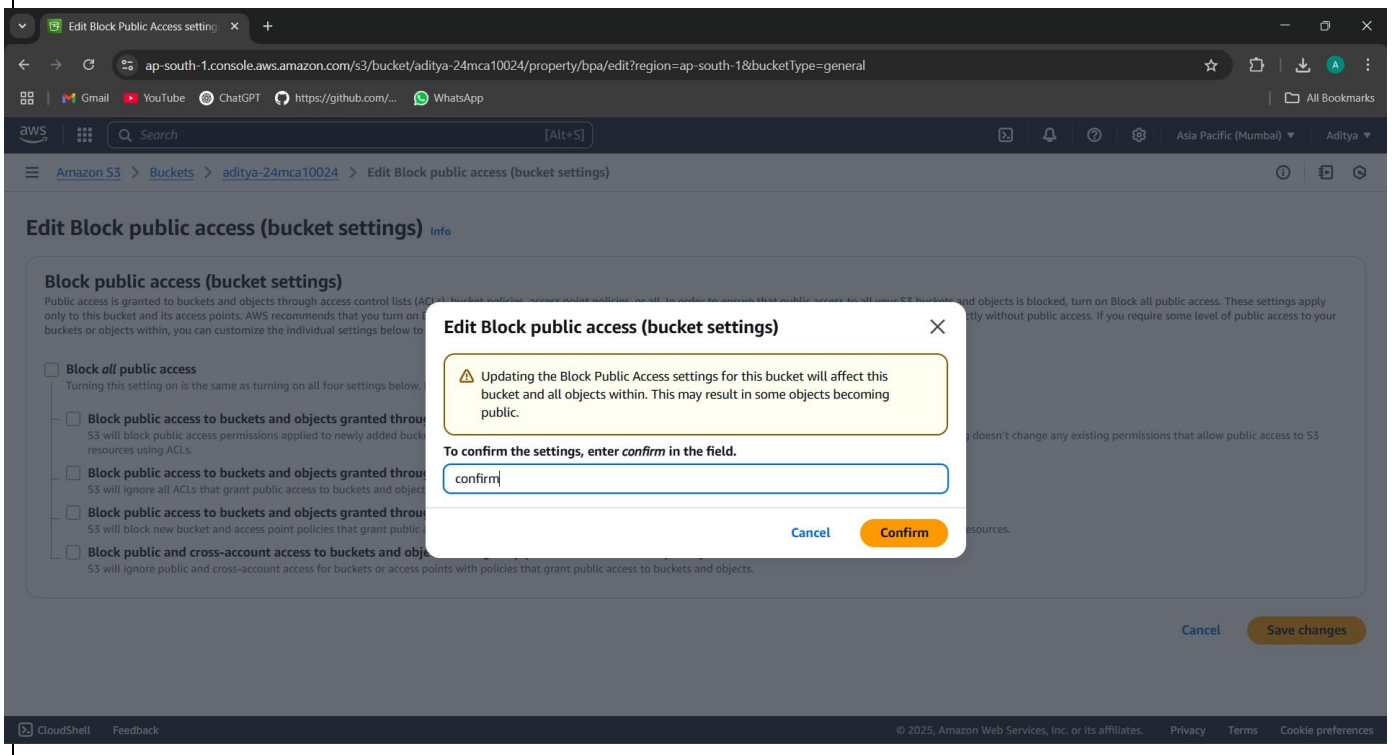
Step 12: -Since public access is not given so the url will not open to access this we move to permission in bucket.



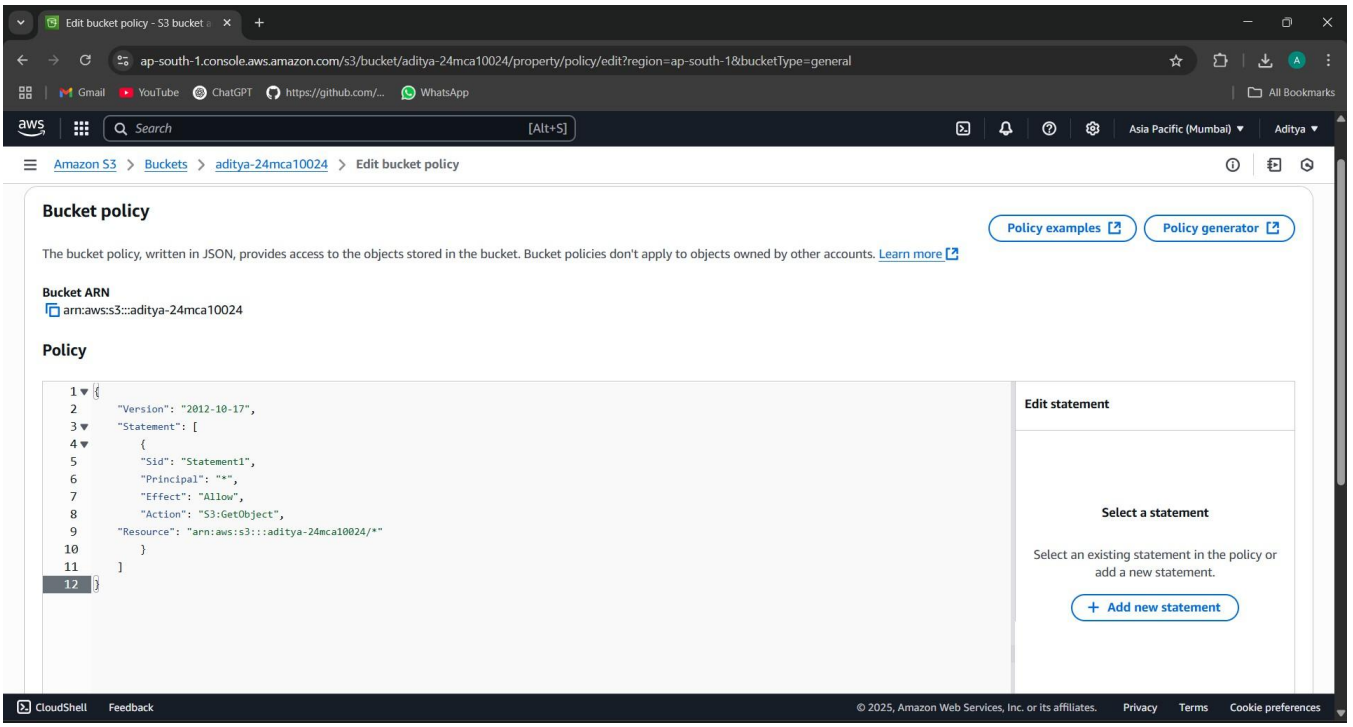
Step 13: -Under block public access we will edit the access crieteria.



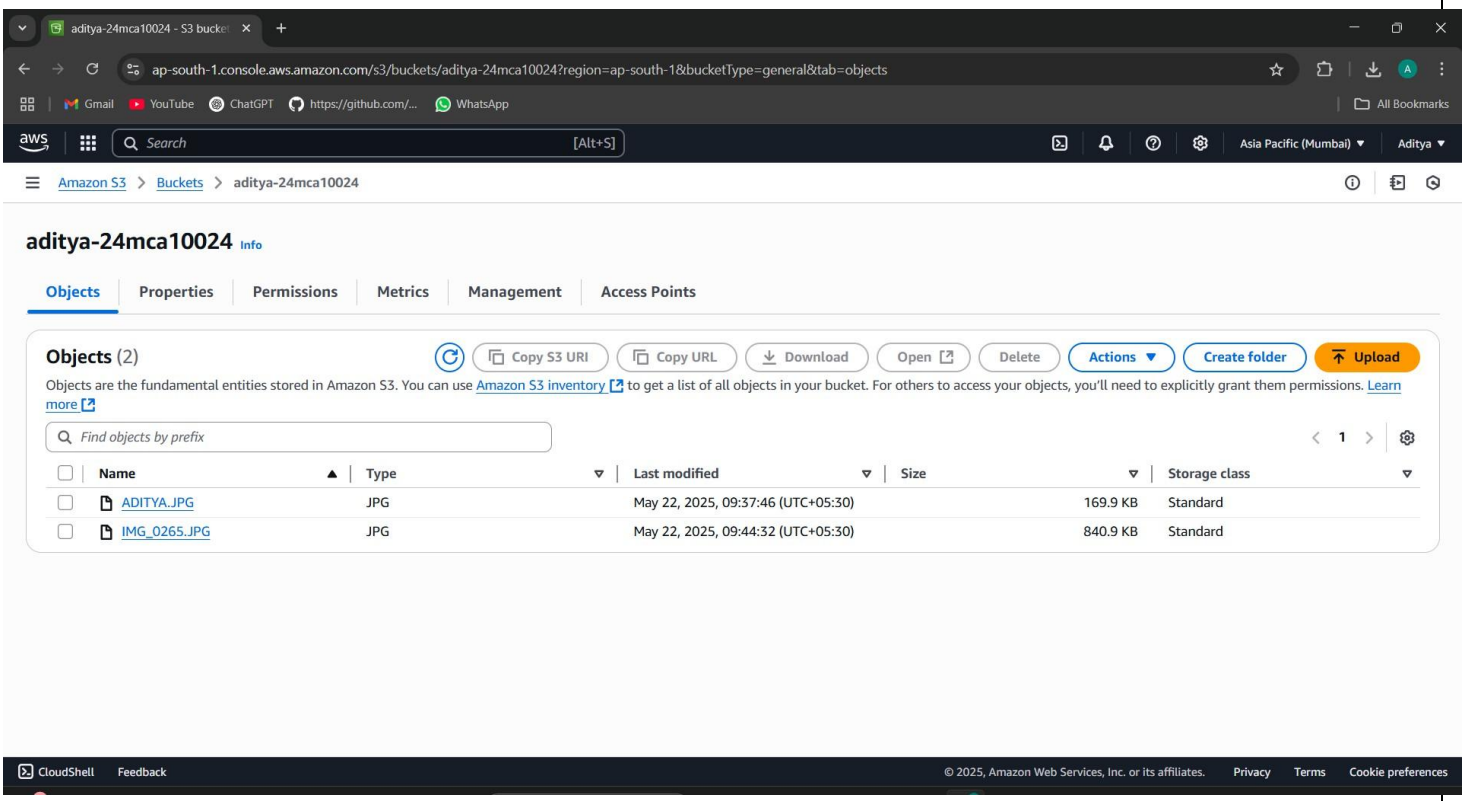
Step 14: -now click on confirm. Then also its not open because public policy is not mentioned.



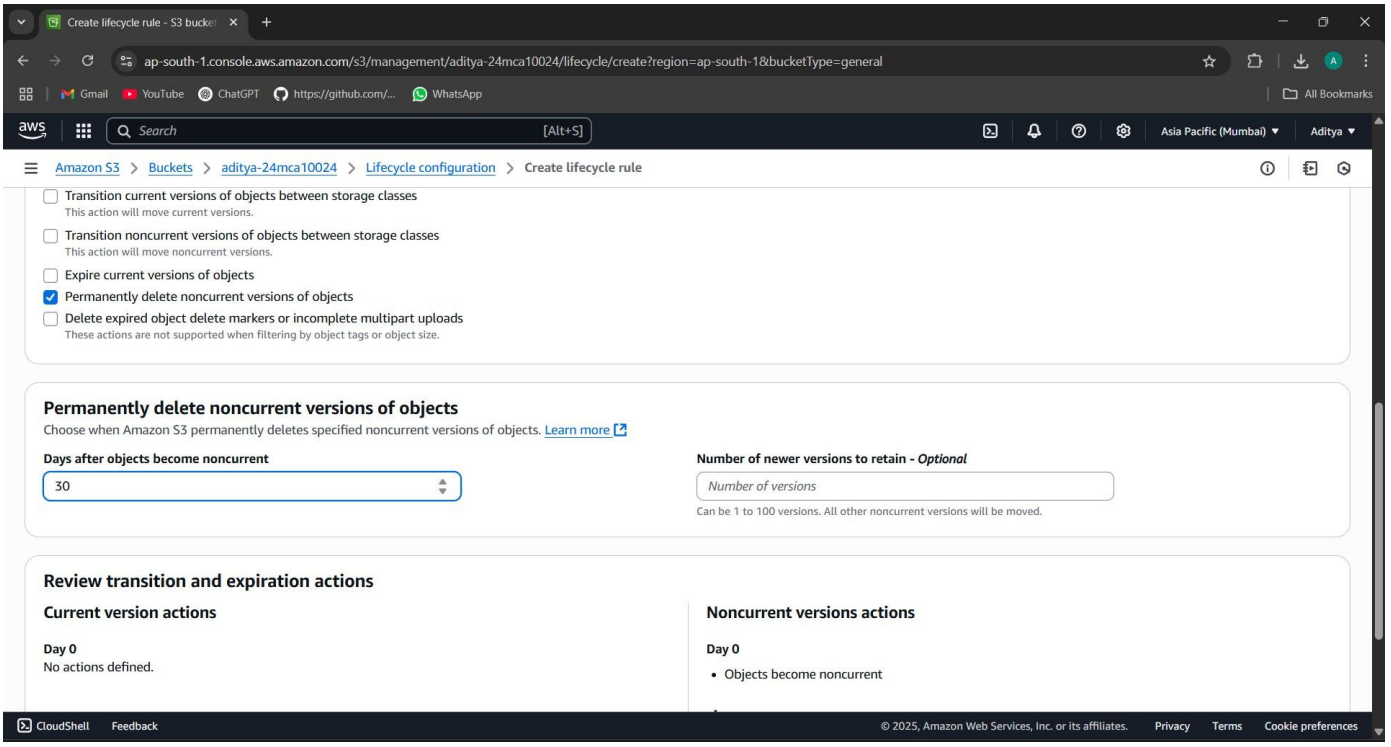
Step 15: -Now writing the code for policy for accessing the file of bucket. And after the file is viewed by everyone.



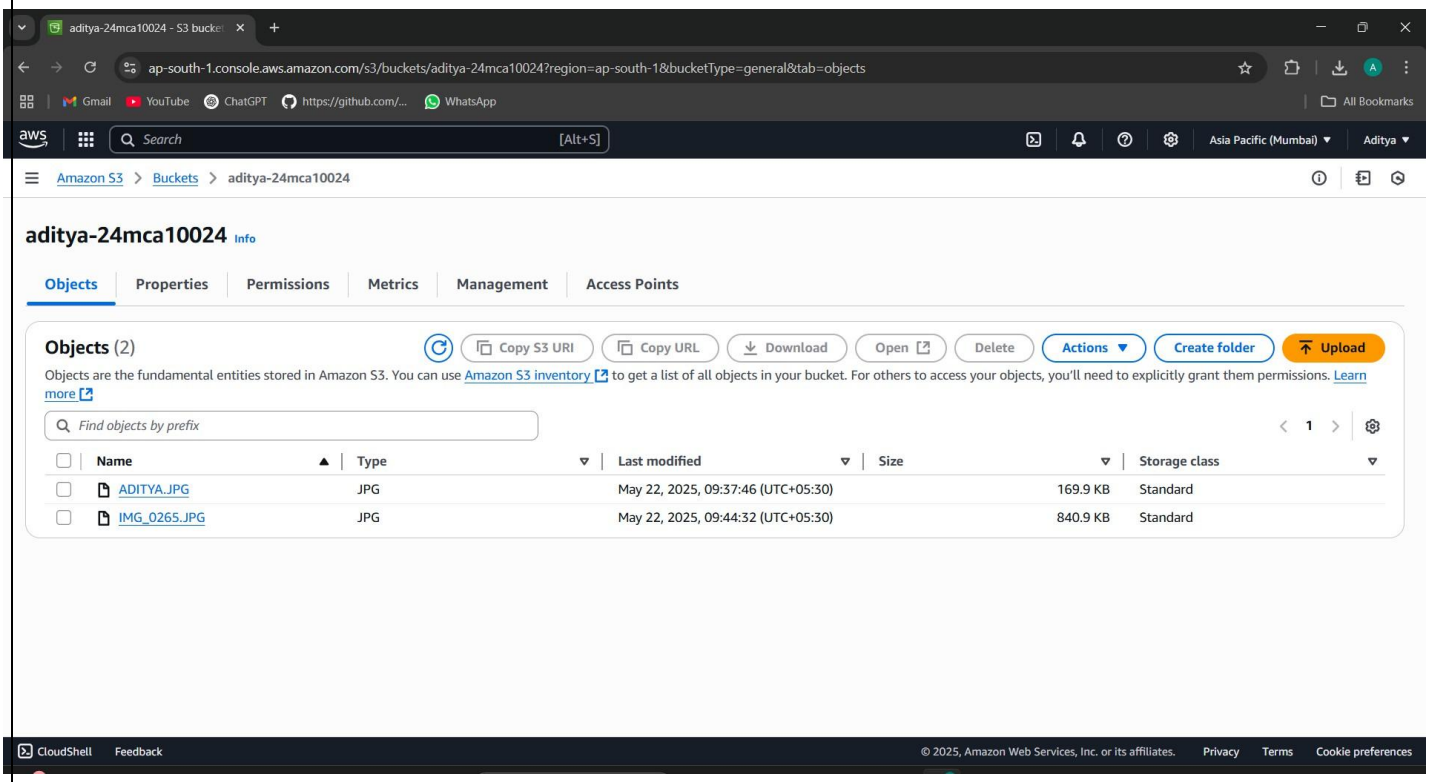
Step 16: -Now in this step we will understand the use of version id in this if upload same file format with same name then it is differentiated using version id as shown



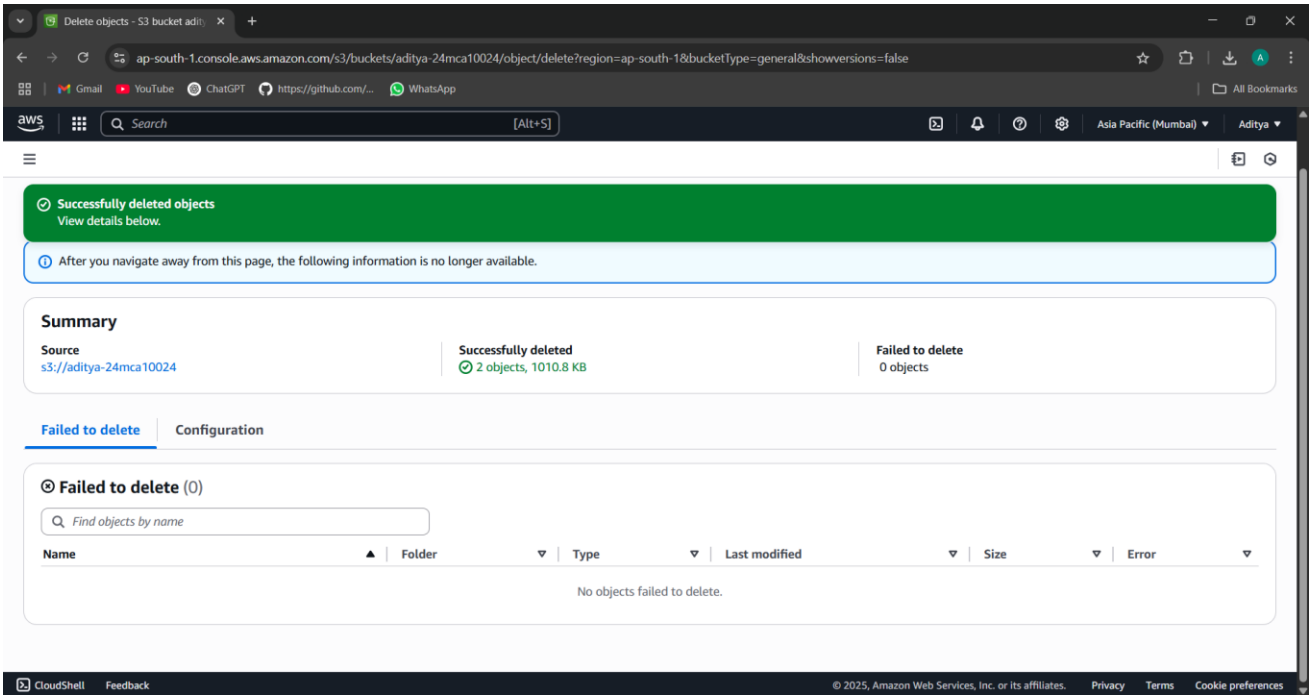
Step 17: -Here we will see about life cycle rule of bucket by giving time period .



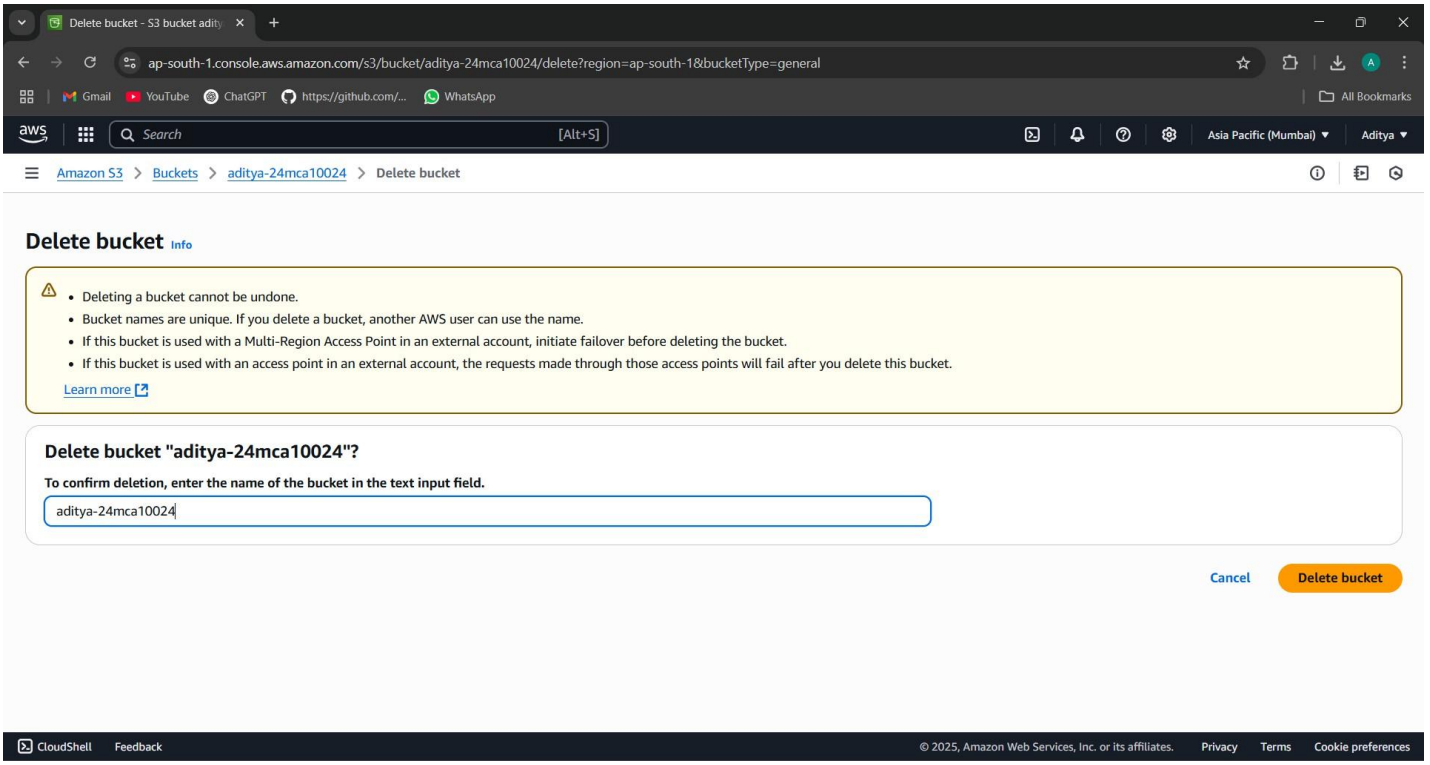
Step 18: -Now in this we will see how to delete the object in bucket. Clicking on the object which we have to delete.



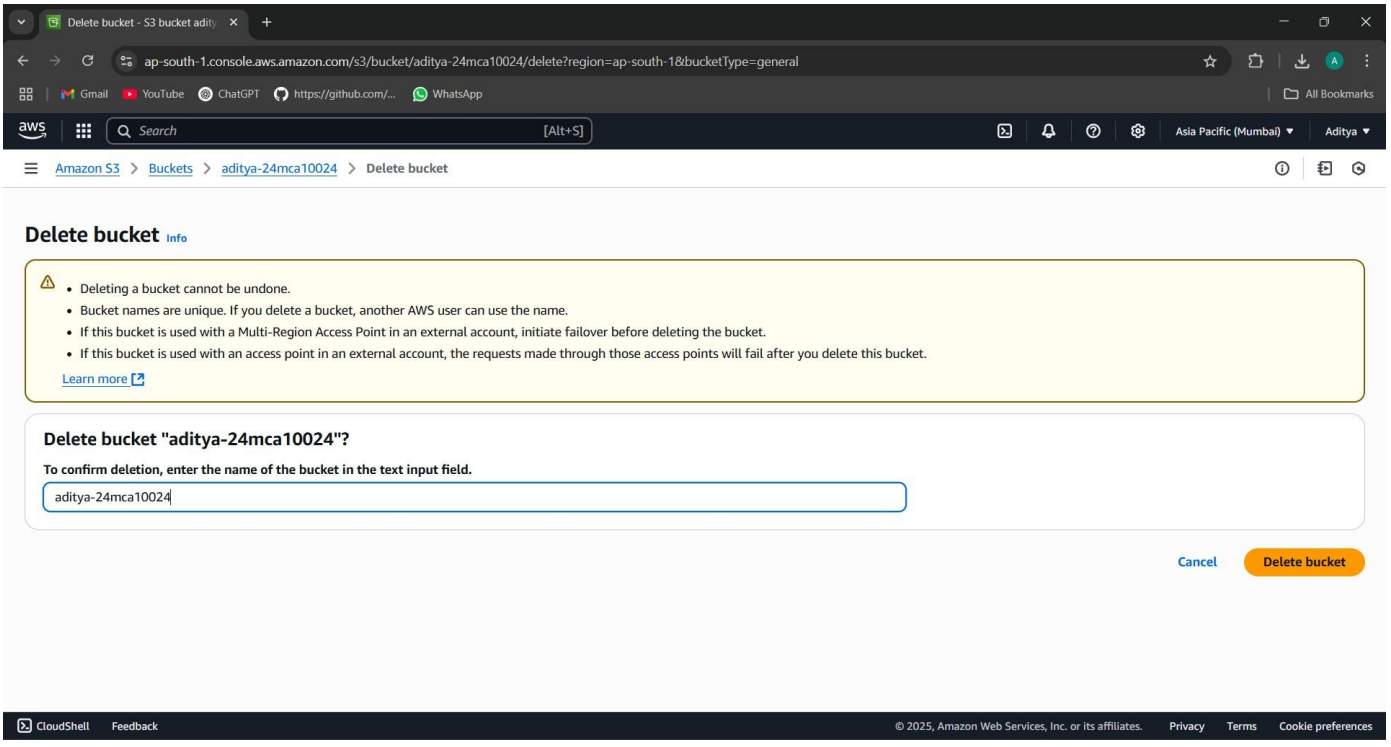
Step 19: -Now click on delete option and type delete to delete the object and empty the bucket.



Step 20: -Now in this we will delete the bucket which we have created .For this the most important part is the bucet must be empty .



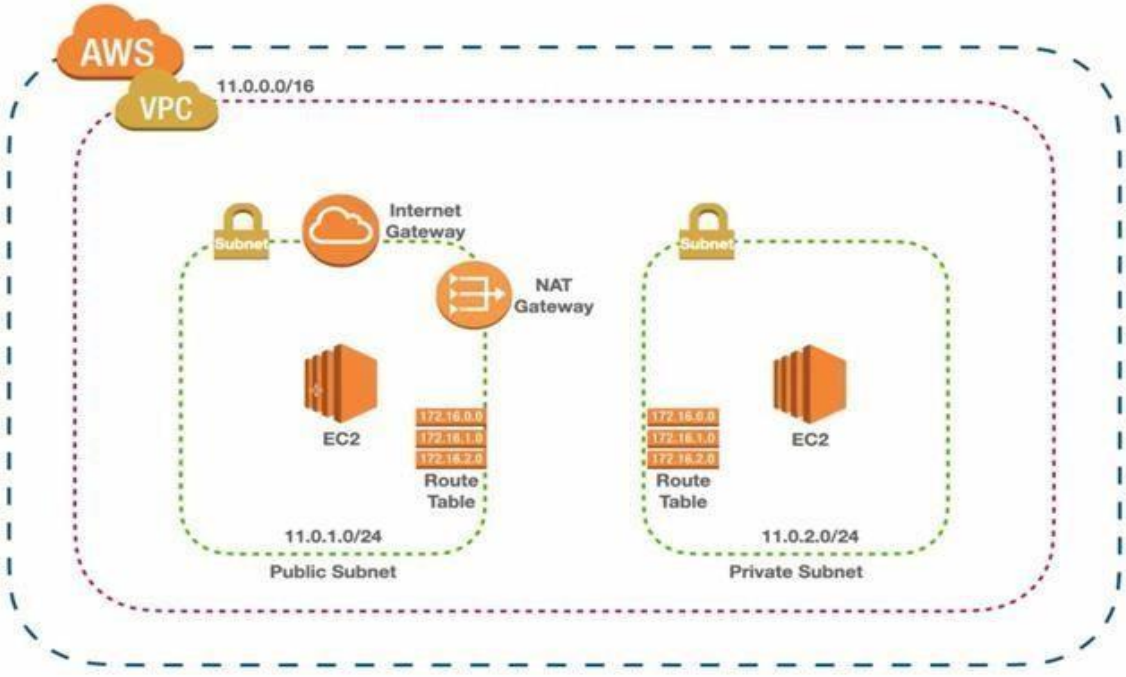
Step 21: - After selecting delete option click on delete option and type bucket name to delete then click on delete the bucket will successfully be deleted.



Date: 23/05/2025	Title
Exp. No: 05	HOW TO SETUP EC2 INSTANCE IN VPC

HOW TO SETUP EC2 INSTANCE IN VPC

AWS VPC (Amazon Virtual Private Cloud): It is a service that allows you to create a private network in the cloud. It gives you full access over your networking environment allowing you to define your IP address range also to create subnets and manage routing tables. With VPC you can securely connect your resources to the internet or keep them isolated from external traffic depending on your needs. It's like building your own private data center within AWS.



Internet Gateway: An Internet Gateway works by establishing a connection between a VPC and the internet. The VPC must have a public subnet, and the instances within that subnet must have a public IP address to communicate with the internet. An Internet Gateway acts as a bridge between the VPC and the internet. An Internet Gateway is commonly used when you want resources within a VPC to be accessible from the internet.

NAT Gateway: A NAT Gateway enables instances in a private subnet to connect to the internet or other AWS services but prevents the internet or other AWS services from initiating a connection with those instances. A NAT Gateway is commonly used when you have resources within a private subnet that require outbound internet access but should not be directly accessible from the internet.

Subnet: A subnet is like a smaller group within a large network. It is a way to split a large network into smaller networks so that devices present in one network can transmit data more easily.

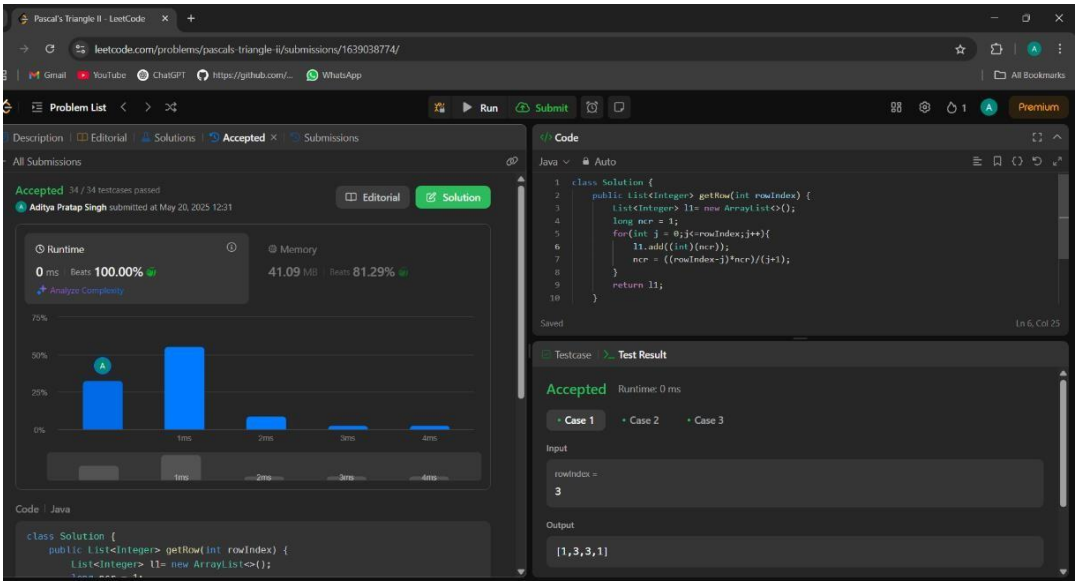
- **The public subnet has instances with public IP addresses allowing them to**

communicate directly with the internet. These instances can be web servers load balancers or other publicly accessible services. An Internet Gateway connects the public subnet to the internet providing inbound and outbound communication.

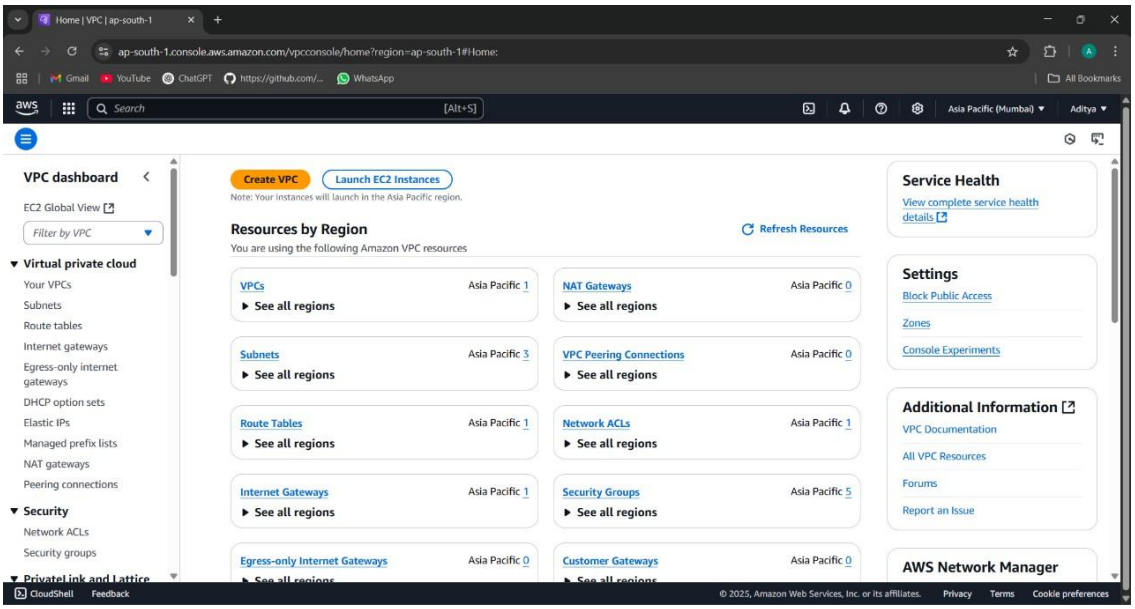
- **The private subnet has instances with private IP addresses. These instances cannot directly access the internet but can connect to resources within the same VPC or other on-premises networks via a VPN or Direct Connect. Typically the instances in the private subnet access the internet via a NAT gateway or NAT instance which is usually placed in the public subnet.**
- **Public subnets have a route to the internet via an Internet Gateway. Resources in public subnets can be accessed from the internet such as web servers whereas Private subnets do not have a direct route to the internet it uses NAT device to get connected with Internet. Resources within private subnets cannot be accessed from outside the VPC unless specific configurations are made.**

Route table: A route table contains a set of rules, called routes, that determine where network traffic from your subnet or gateway is directed. It is used to direct network traffic on a particular destination IP address. It allows customization of networks enabling them to connect to other subnets/gateways both within and outside of the VPC.

STEP1: LOGIN AS A ROOT USER AND SEARCH FOR VPC IN THE CONSOLE

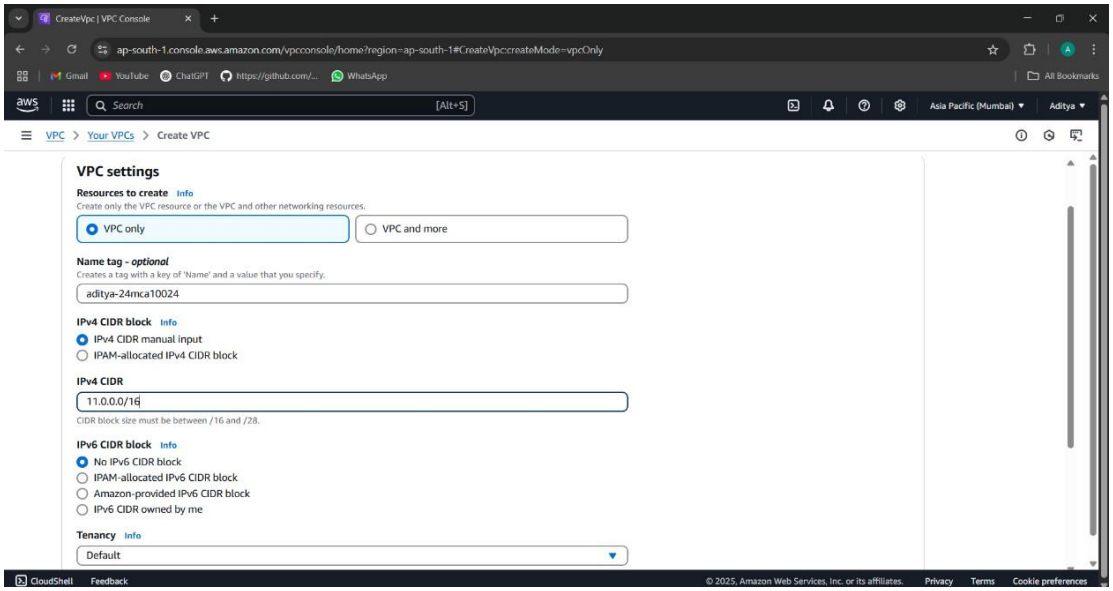


STEP2: CLICK ON CREATE VPC

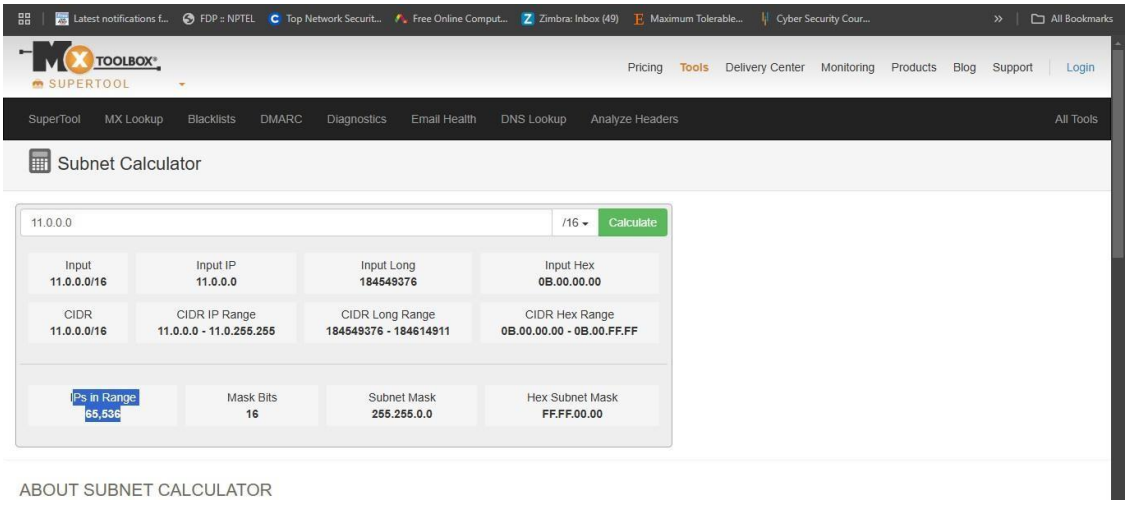


STEP3: SELECT VPC ONLY> TYPE THE VPC NAME> IPv4CIDR-11.0.0.0/16

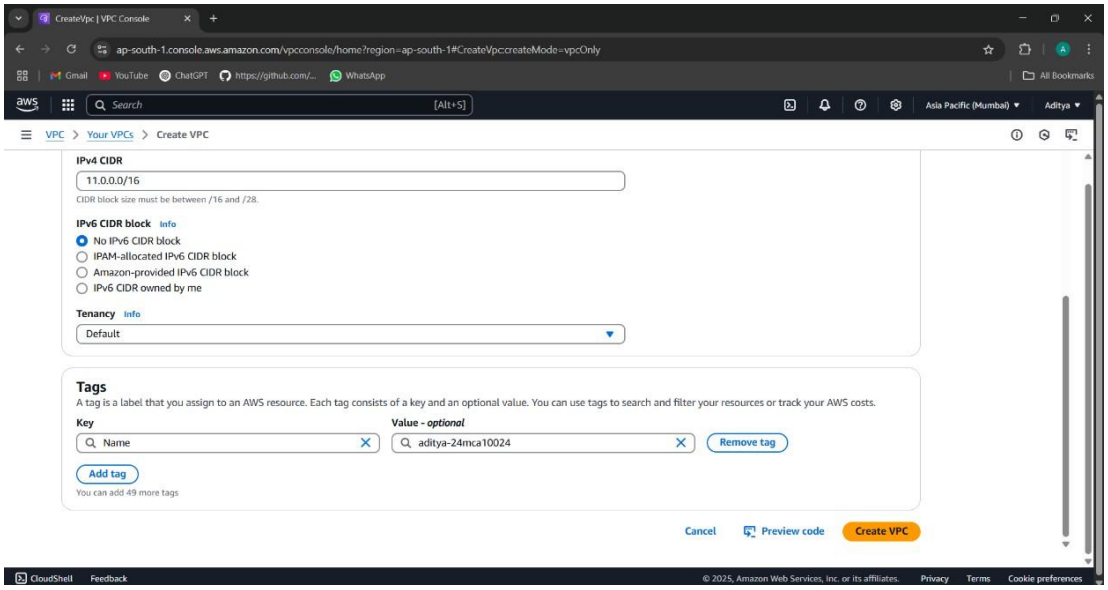
NOTE: In AWS a VPC spans a specific IP address range using CIDR (Classless Inter-Domain Routing) blocks. The CIDR block defines the range of IP addresses that can be assigned to resources within the VPC



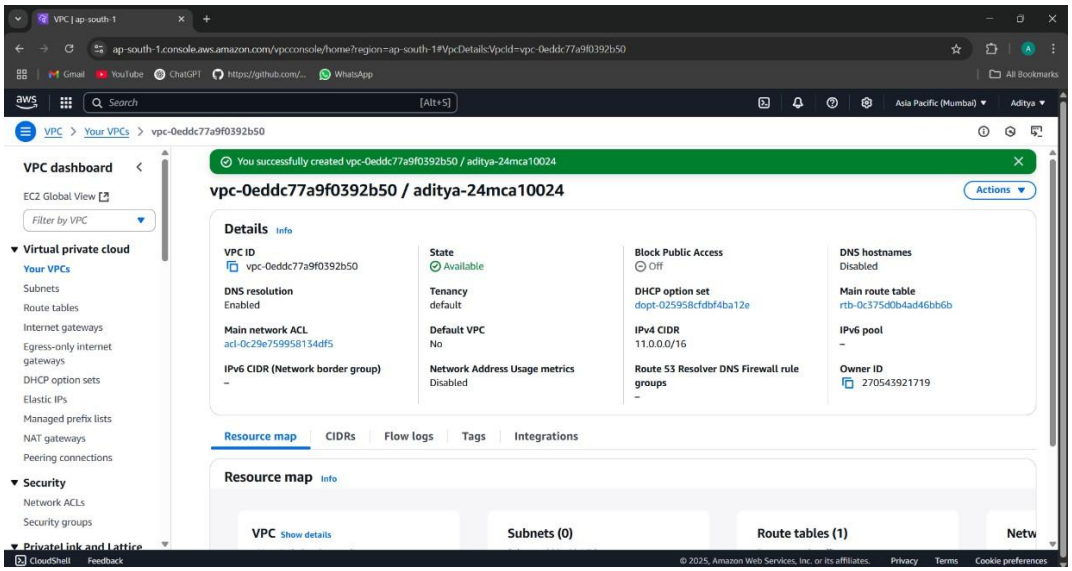
For better understanding of the CIDR range, Search CIDR Range Calculator in google and enter the range. Here its showing 65536 IP range, i.e. these are the maximum number of IP addresses which is accessible by the VPC. If we increase from /16 to /32, it will decrease the number of IP address and if we decrease it to /8, it will increase the number of IP addresses. (/16 means first 16bits are locked)



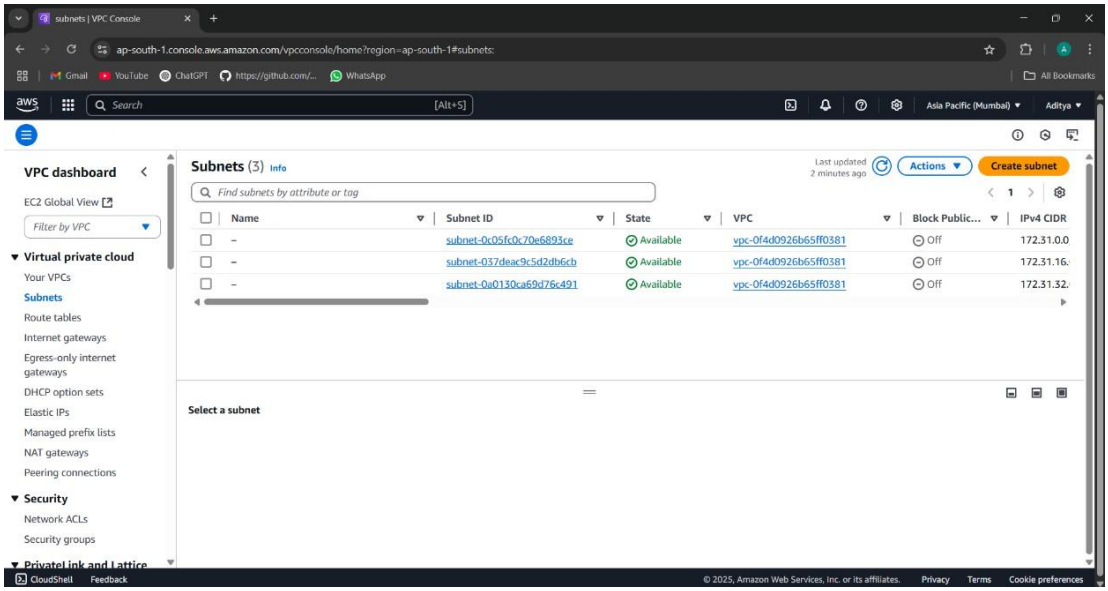
STEP4: CLICK ON CREATE VPC



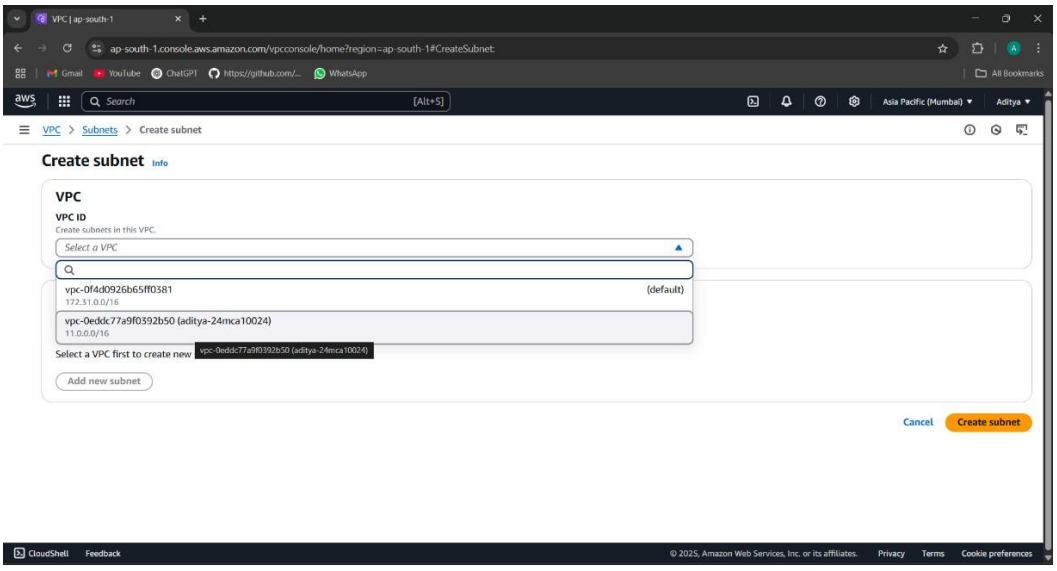
STEP5: VPC CREATED



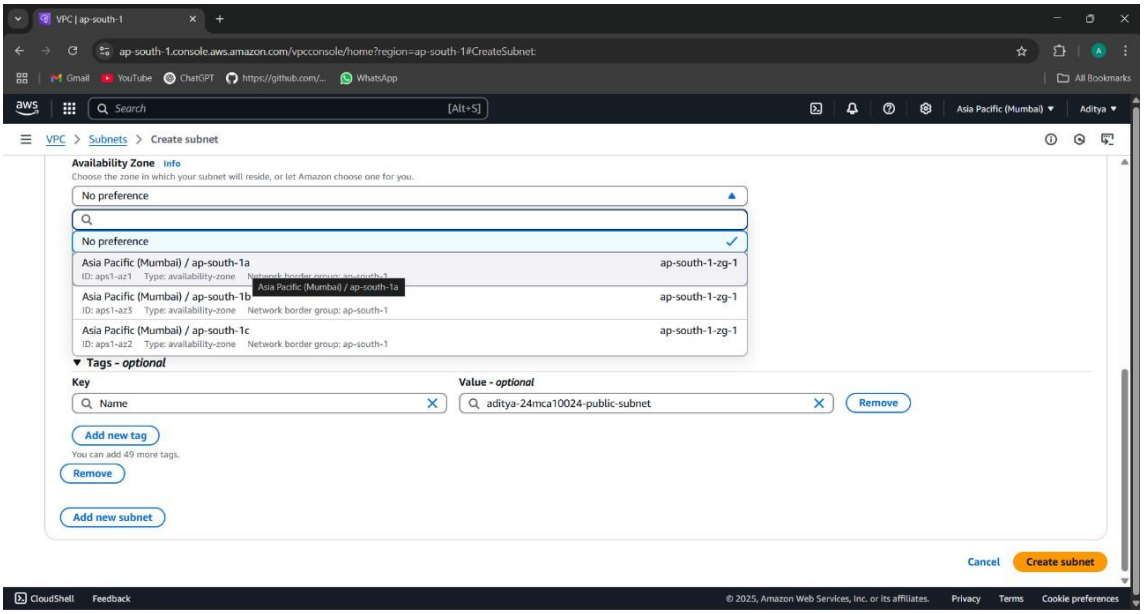
STEP6: CLICK ON SUBNET>CREATE SUBNET



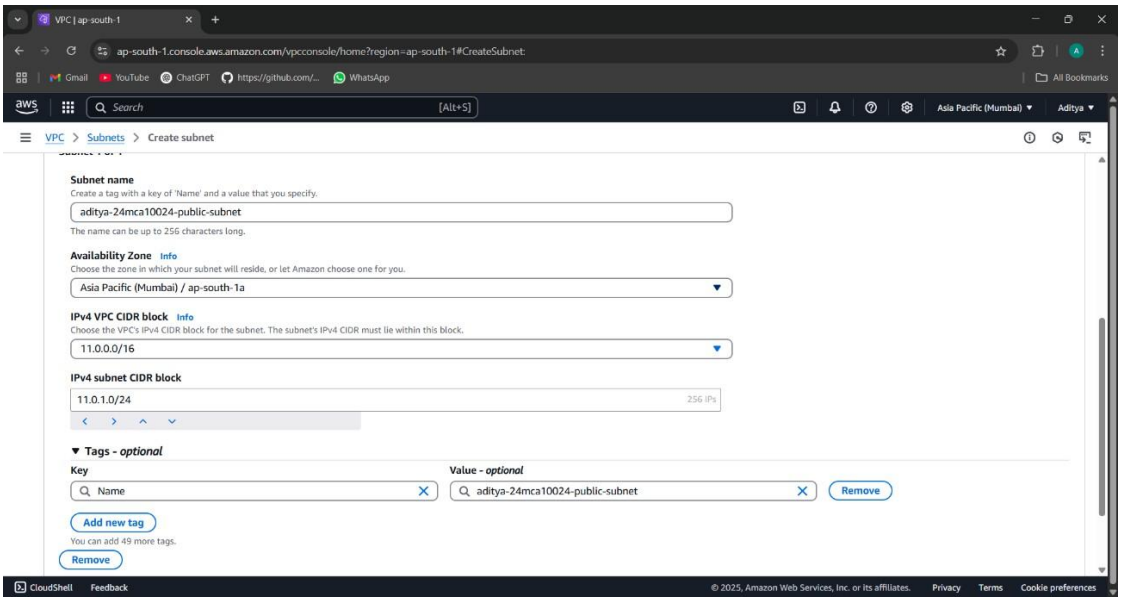
STEP7: SELECT THE VPC THAT YOU HAVE CREATED EARLIER



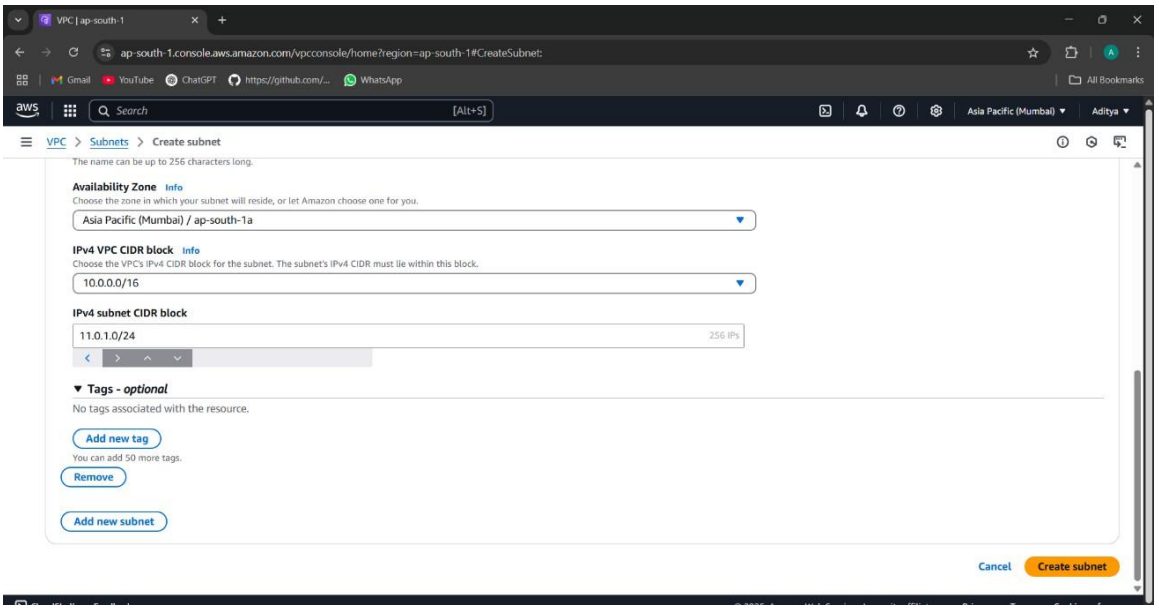
STEP8: SELECT ap-south-1a in the Availability Zone



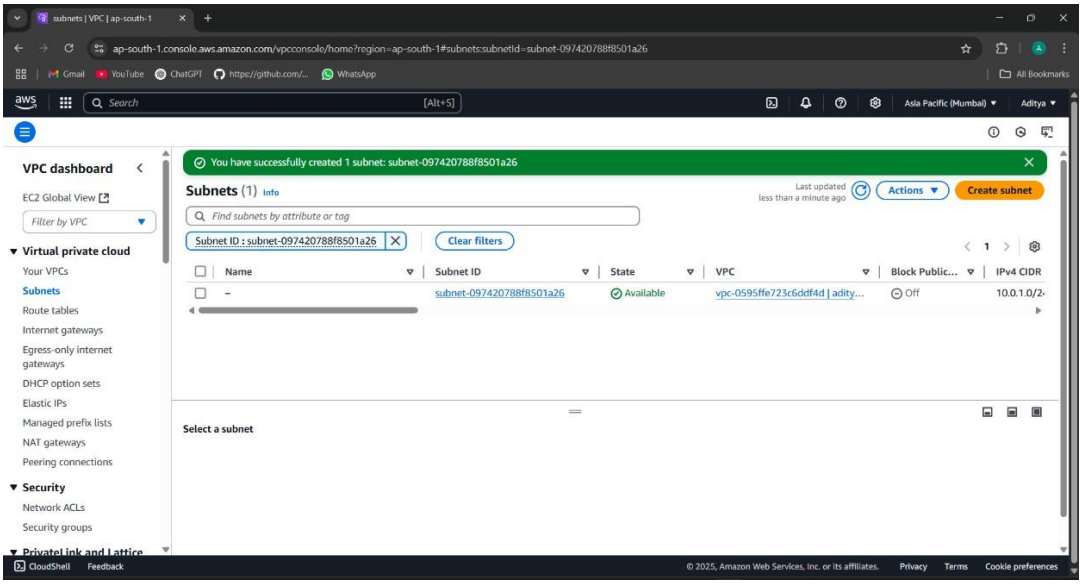
STEP9: ENTER IPv4 SUBNET CIDR BLOCK-11.0.1.0/24> CLICK ON ADD NEW SUBNET



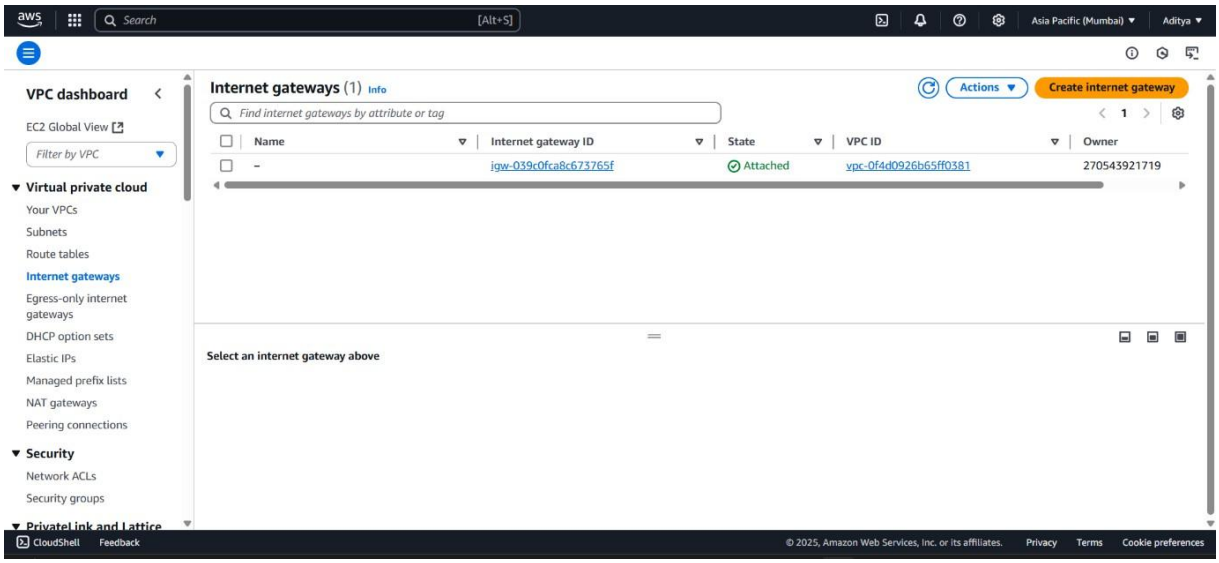
STEP9: ENTER THE NAME OF THE PRIVATE SUBNET> AVAILABILITY ZONE:ap-south-1b> IPv4 subnet CIDR Block-11.0.2.0/24> CLICK ON CREATE SUBNET



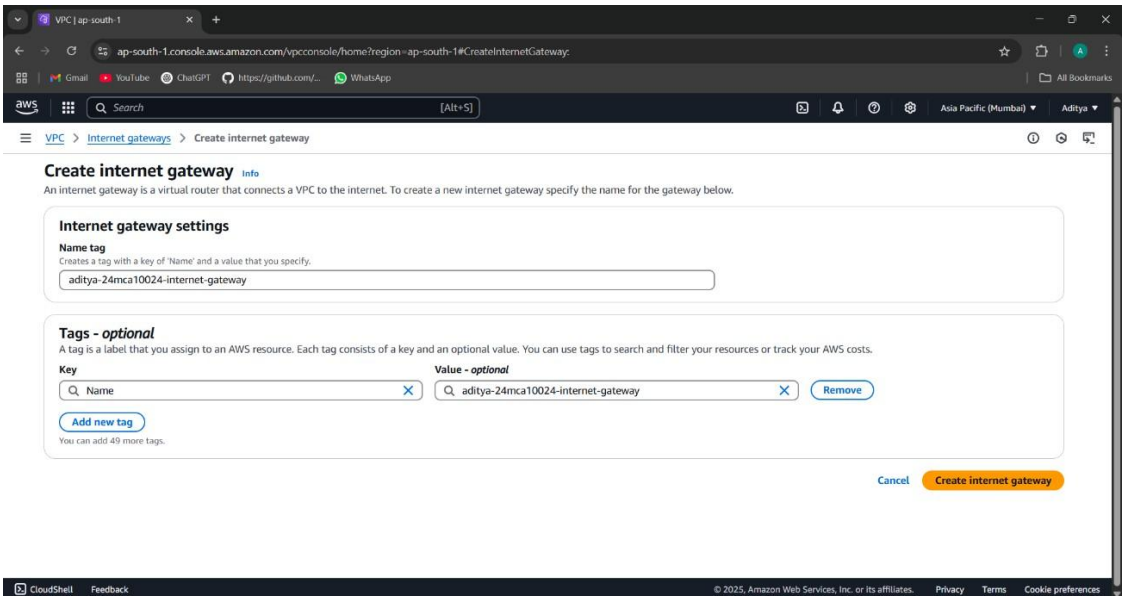
STEP10: BOTH THE PUBLIC AND PRIVATE SUBNET IS CREATED



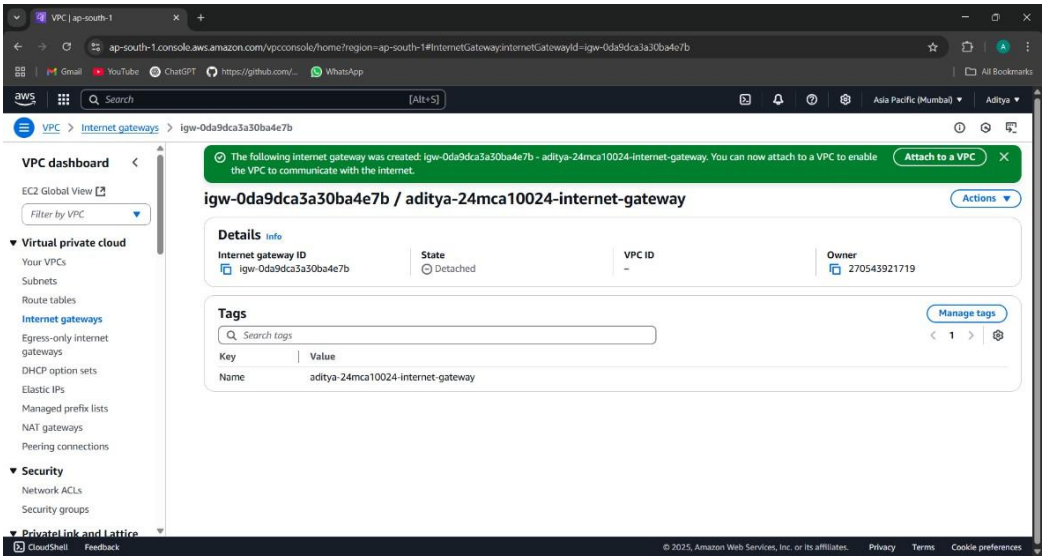
STEP11: NOW CREATE AN INTERNET GATEWAY FOR PUBLIC SUBNET. CLICK ON INTERNET GATEWAY FROM THE DASHBOARD> CLICK ON CREATE INTERNET GATEWAY



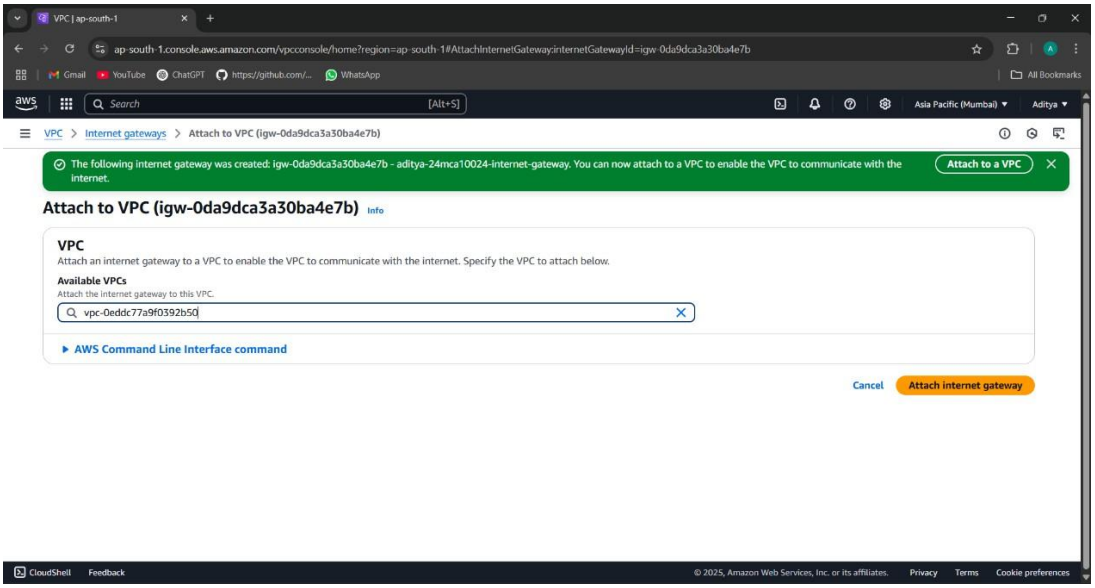
STEP12: TYPE THE NAME OF THE GATEWAY



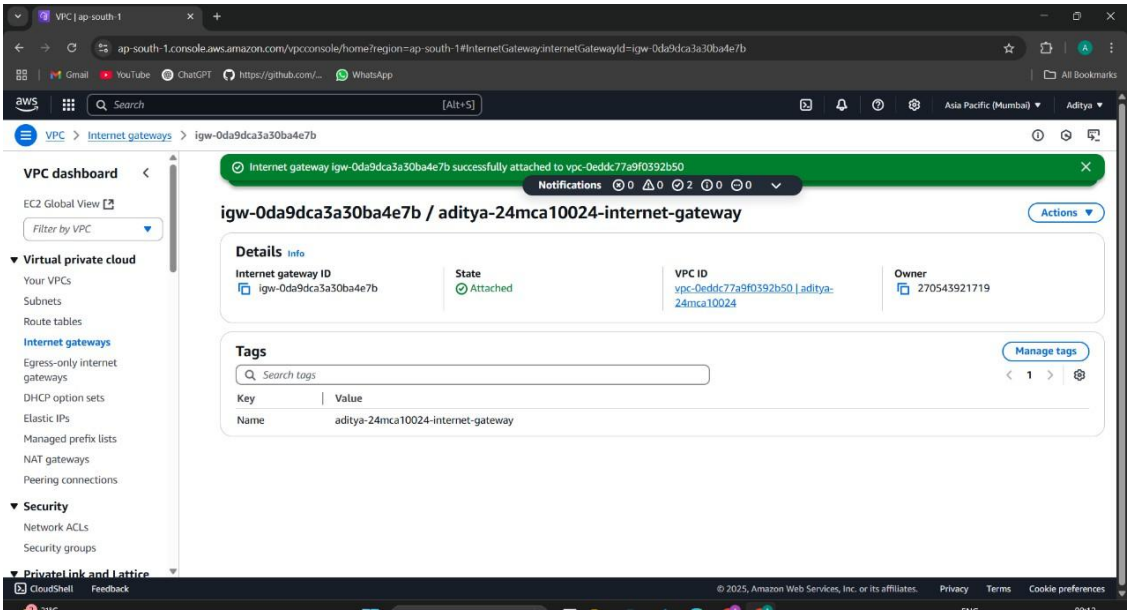
STEP13: INTERNET GATEWAY IS CREATED



STEP14: CLICK ON ATTACH TO VPC> SELECT THE VPC CREATED EARLIER> CLICK ON ATTACH INTERNET GATEWAY

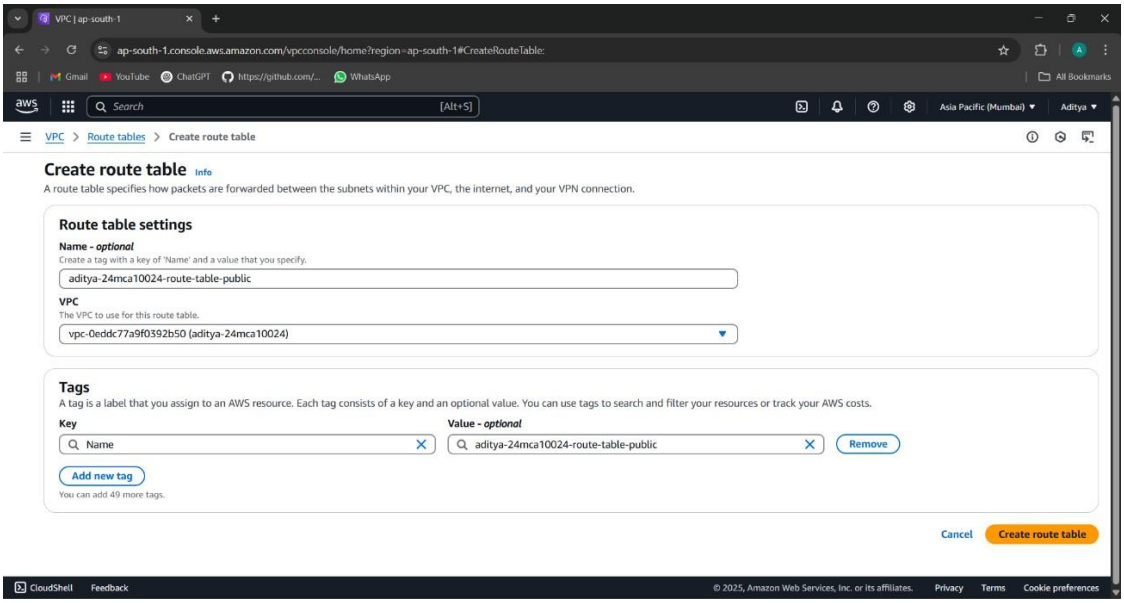


STEP15: INTERNET GATEWAY IS ATTACHED TO THE VPC

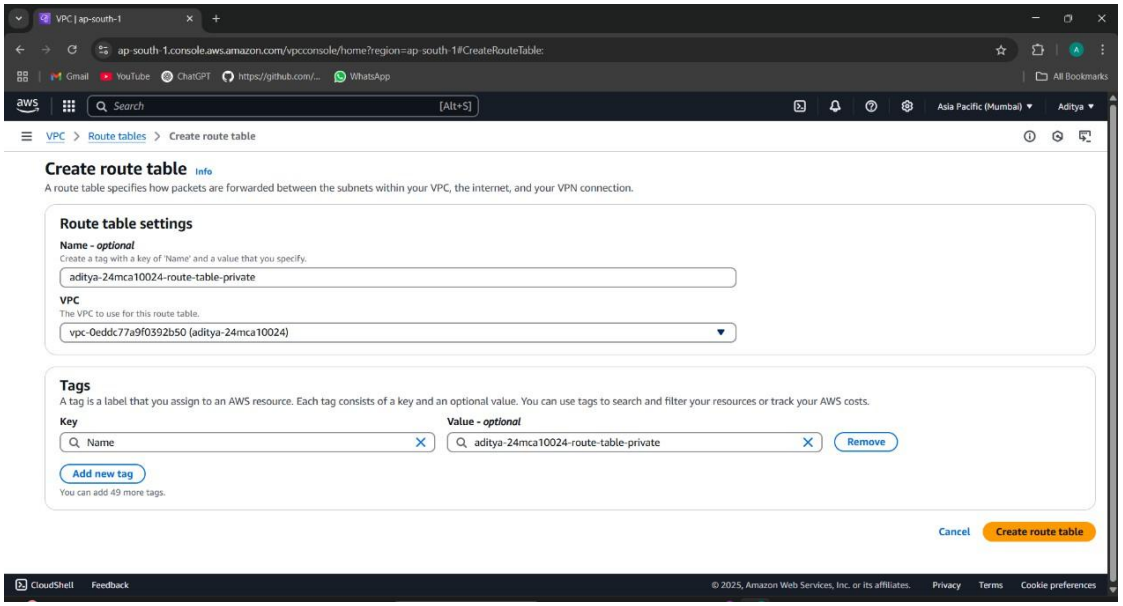


STEP16: NEXT CREATE TWO ROUTE TABLES (PRIVATE AND PUBLIC) FOR THE SUBNETS.

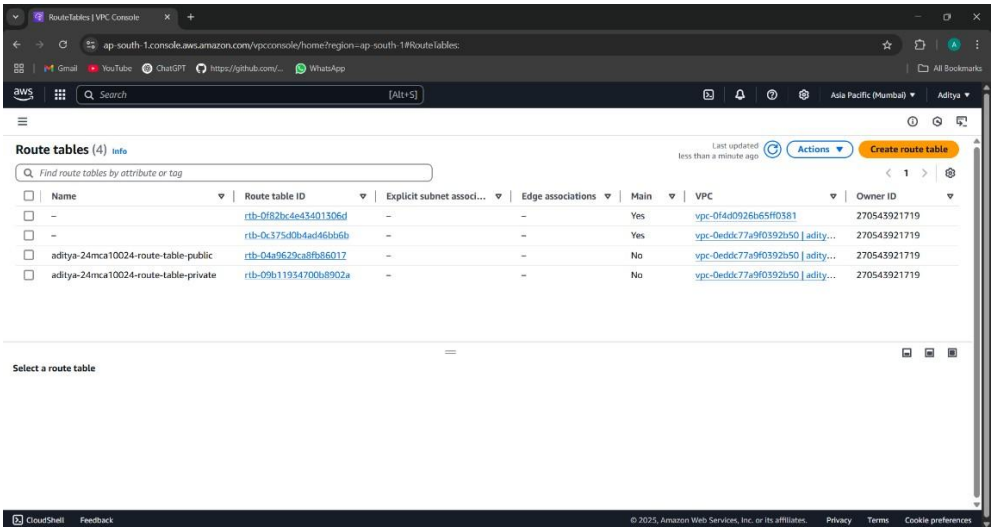
STEP17: TO CREATE A PUBLIC ROUTE TABLE: CLICK ON ROUTE TABLES>CREATE A ROUTE TABLE> WRITE A SUITABLE NAME FOR THE ROUTE TABLE>CHOOSE THE VPC>CLICK ON CREATE ROUTE TABLE



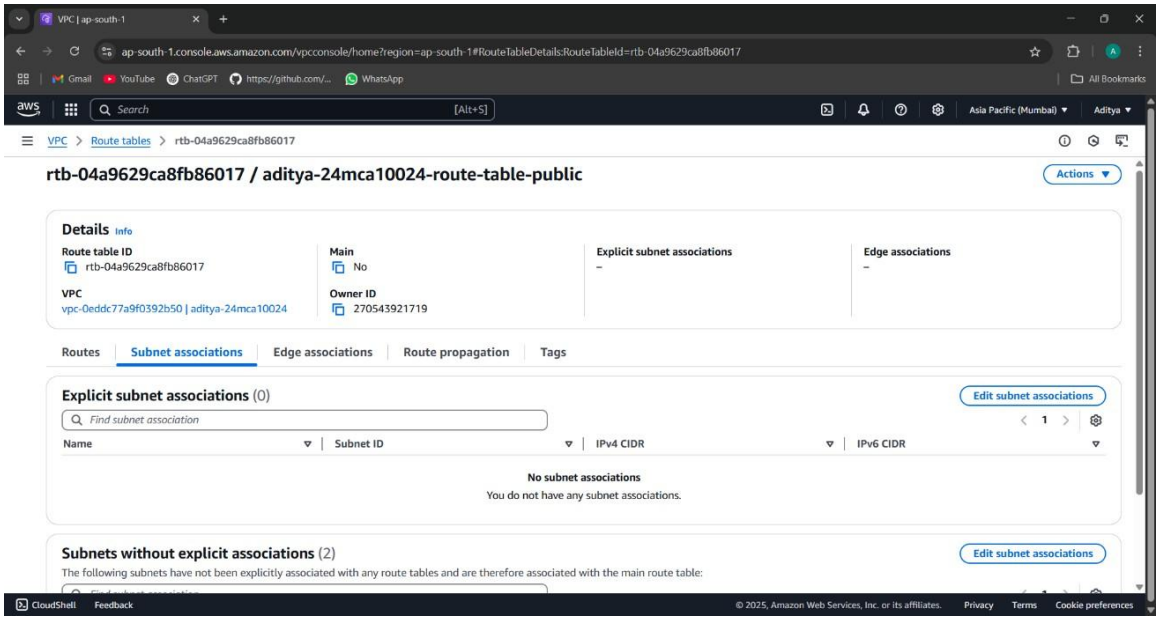
STEP18: TO CREATE A PRIVATE ROUTE TABLE: CLICK ON ROUTE TABLES>CREATE A ROUTE TABLE> WRITE A SUITABLE NAME FOR THE ROUTE TABLE>CHOOSE THE VPC>CLICK ON CREATE ROUTE TABLE



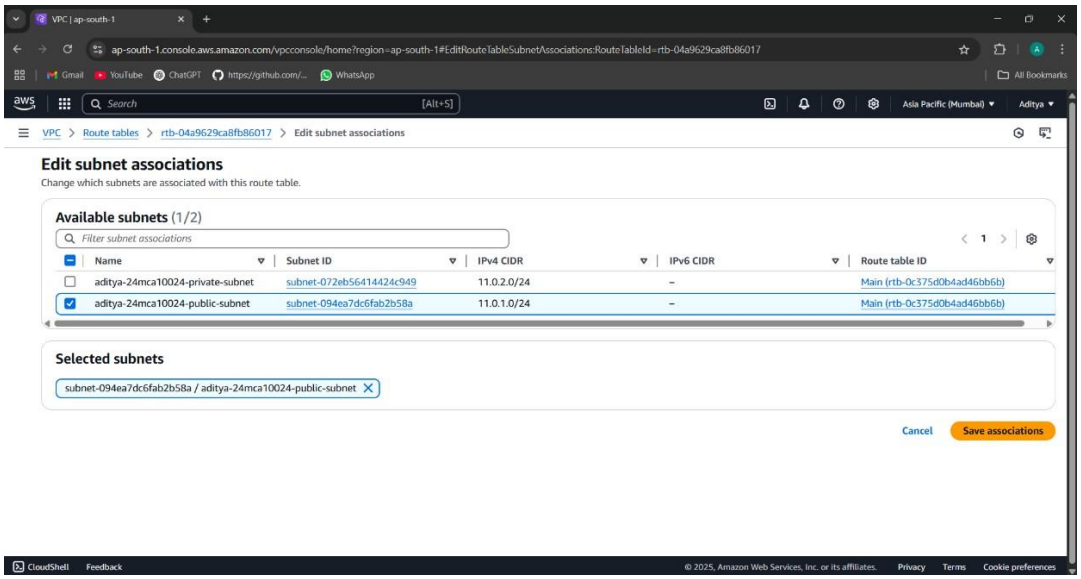
STEP19: BOTH THE PUBLIC AND PRIVATE ROUTE TABLES ARE CREATED AS SHOWN BELOW. NOW WE NEED TO CONNECT BOTH THE ROUTE TABLES WITH THEIR CORRESPONDING SUBNETS.



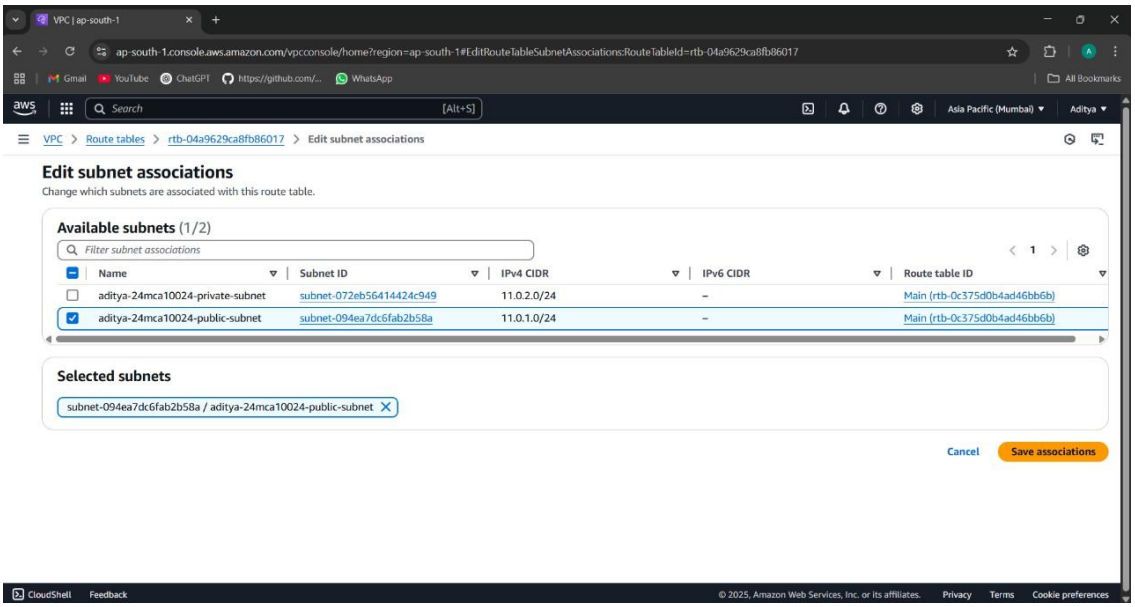
STEP20: CLICK ON THE PUBLIC ROUTE TABLE>CLICK ON SUBNET ASSOCIATIONS



STEP 21: CLICK ON EDIT SUBNET ASSOCIATIONS > CHOOSE PUBLIC SUBNET FROM THE OPTIONS> CLICK ON SAVE ASSOCIATIONS

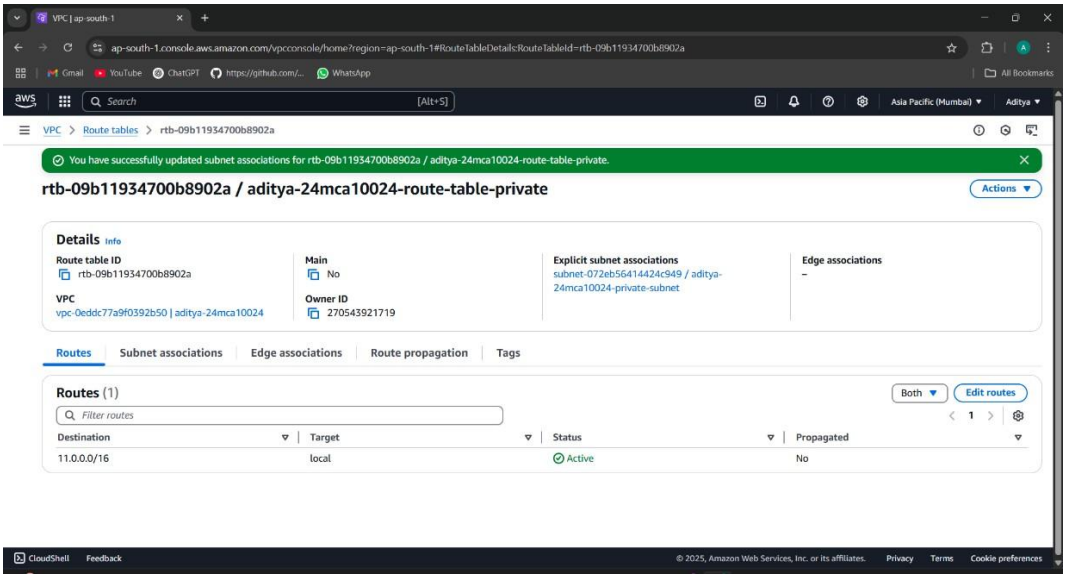


STEP 21: SUBNET ASSOCIATION IS DONE. REPEAT THE SAME STEP FOR PRIVATE SUBNET ASSOCIATION.

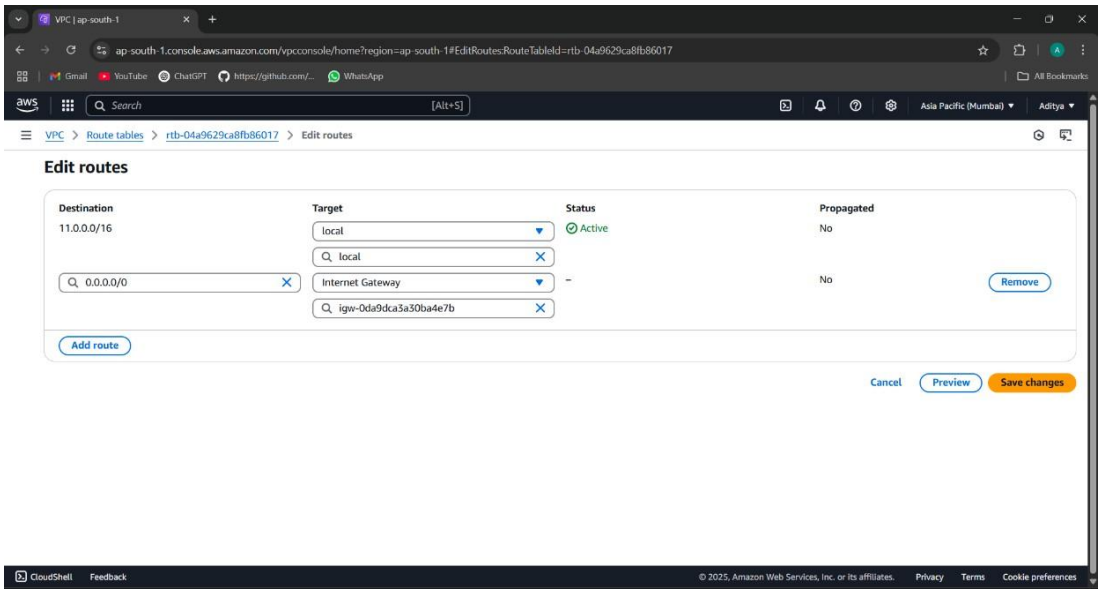


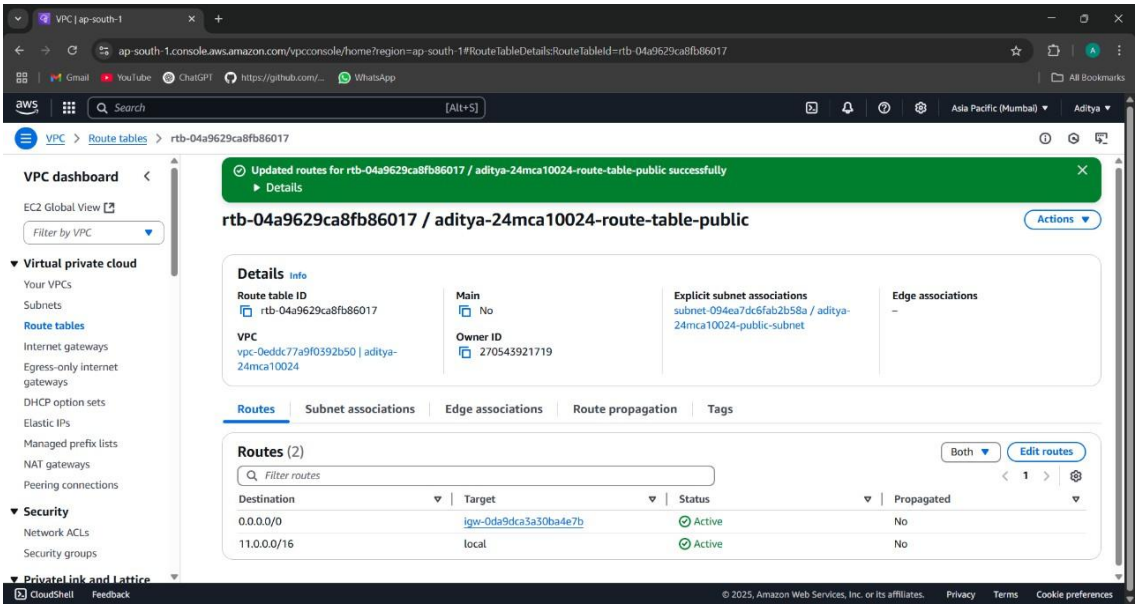
STEP22: NEXT WE NEED TO CREATE THE ROUTE SO THAT INTERNET CAN BE ACCESSED WITH THE HELP OF INTERNET GATEWAY THROUGH THESE ROUTE TABLES.

STEP23: GO TO PUBLIC ROUTE TABLE FROM THE AWS CONSOLE> CLICK ON ROUTES>CLICK ON EDIT ROUTES



STEP24: CLICK ON ADD ROUTE>SELECT THE IP-0.0.0.0/0>CHOOSE INTERNET GATEWAY> SELECT THE INTERNET GATEWAY THAT YOU HAVE CREATED> CLICK ON SAVE CHANGES (NOTE:0.0.0.0/0 means it allows all the IP addresses to access the resources present in the public subnet)



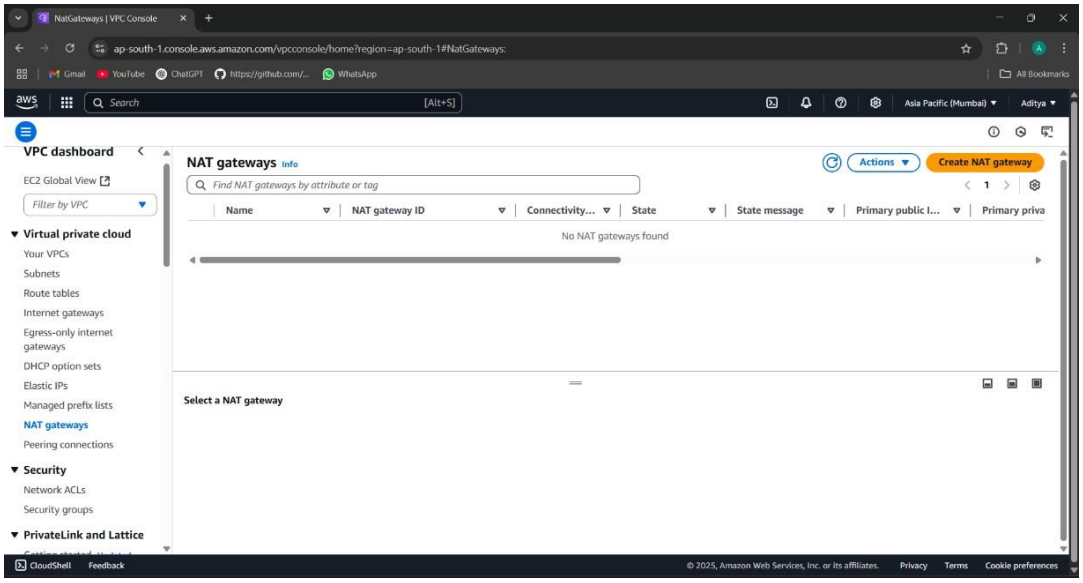


STEP25: WE NEED TO CREATE THE NAT GATEWAY FOR THE PRIVATE SUBNET SO THAT RESOURCES PRESENT INSIDE THE PRIVATE SUBNET CAN ACCESS THE INTERNET WITH THE HELP OF INTERNET GATEWAY.

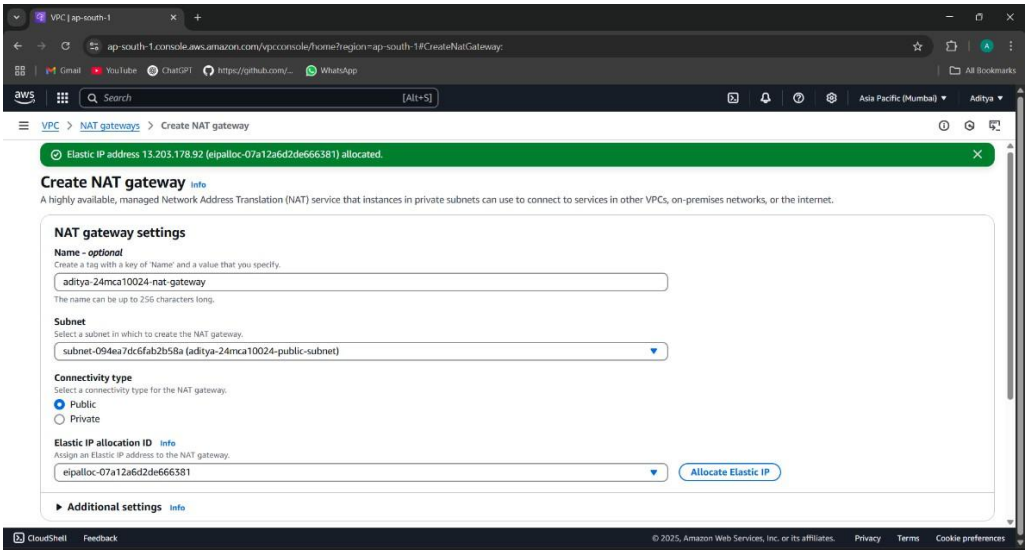
NOTE:

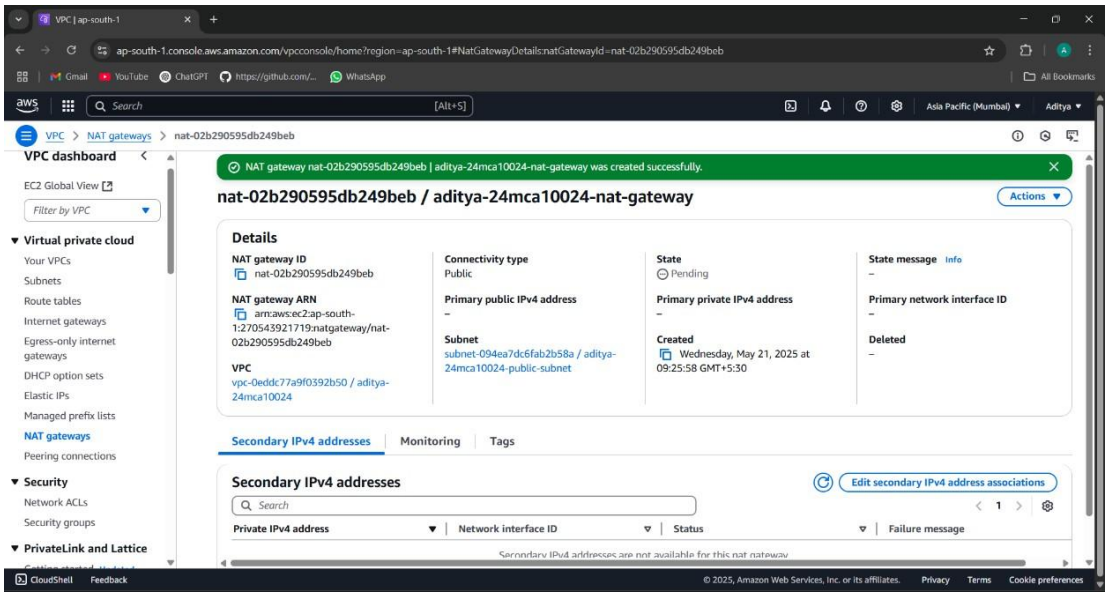
1. **Internet Gateway is two-way process i.e., outside world can access the resources inside the public subnet as well as resources inside it can request for accessing outside.**
2. **NAT GATEWAY is not free and its created only in public subnet.**
3. **Private subnet will be able to access the NAT gateway with the help of its route table to access resources inside the public subnet.**

STEP26: CLICK ON NAT GATEWAYS FROM THE DASHBOARD> CLICK ON CREATE NAT GATEWAY.

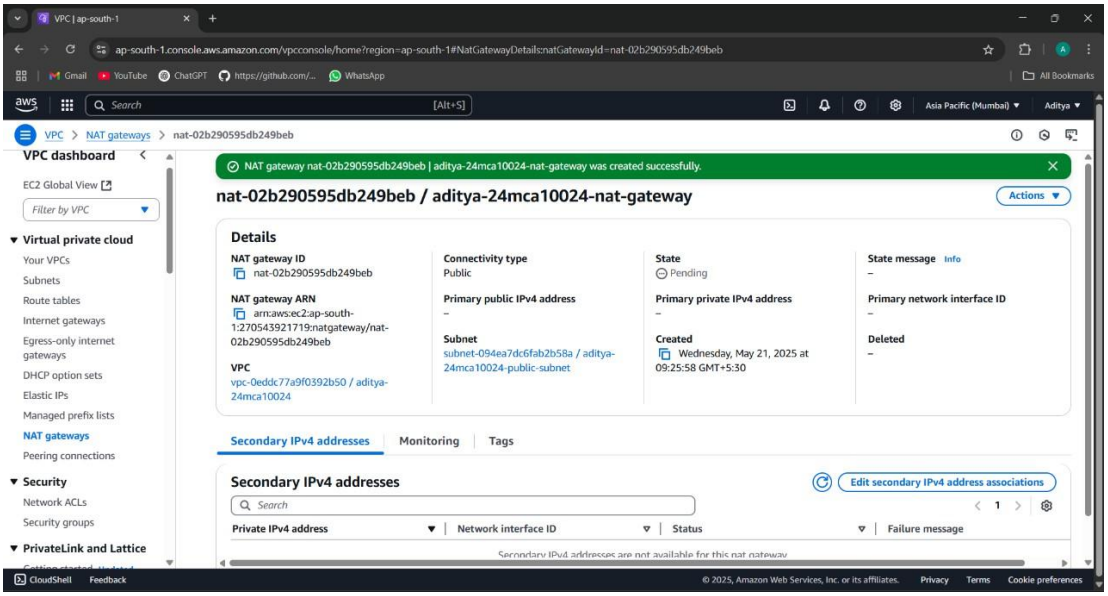


STEP27: GIVE A NAME TO THE NAT GATEWAY>CHOOSE THE PUBLIC SUBNET>ALLOCATE ELASTIC IP> CLICK ON CREATE NAT GATEWAY.



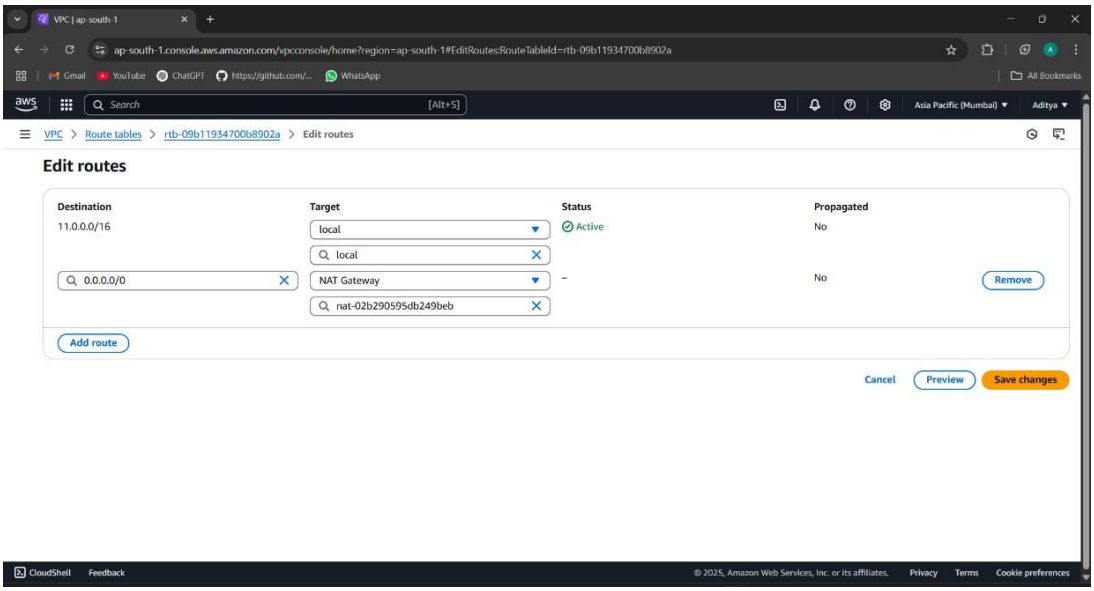


STEP28: NOW UPDATE THE PRIVATE ROUTE TABLE: CLICK ON THE ROUTE TABLES>CLICK ON THE PRIVATE ROUTE TABLE>CLICK ON ROUTES>CLICK ON EDIT ROUTES

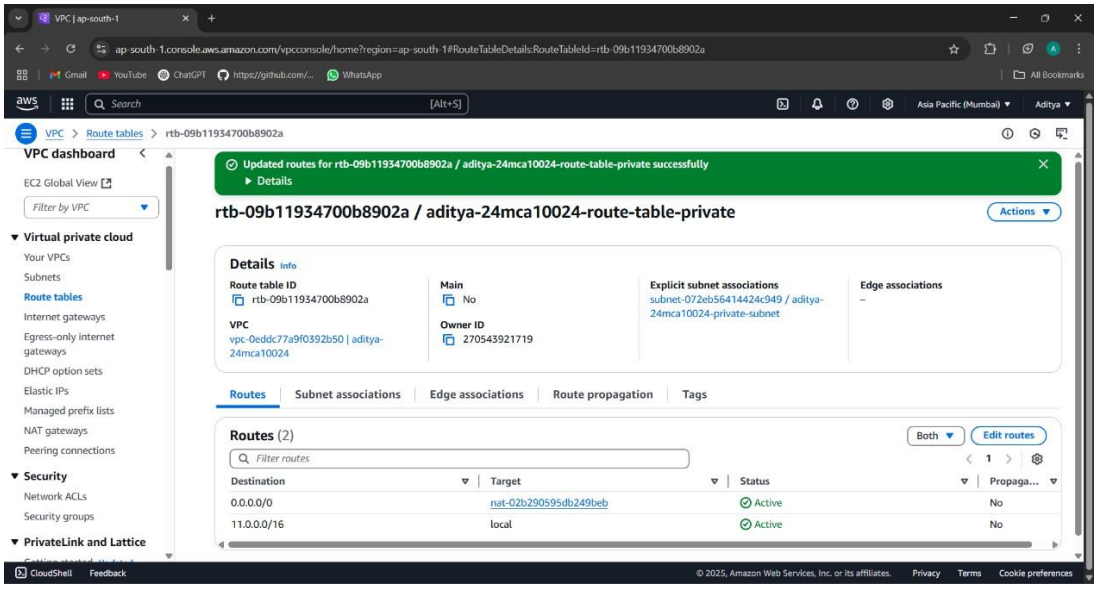


STEP29: CLICK ON ADD ROUTE>SELECT IP ADDRESS

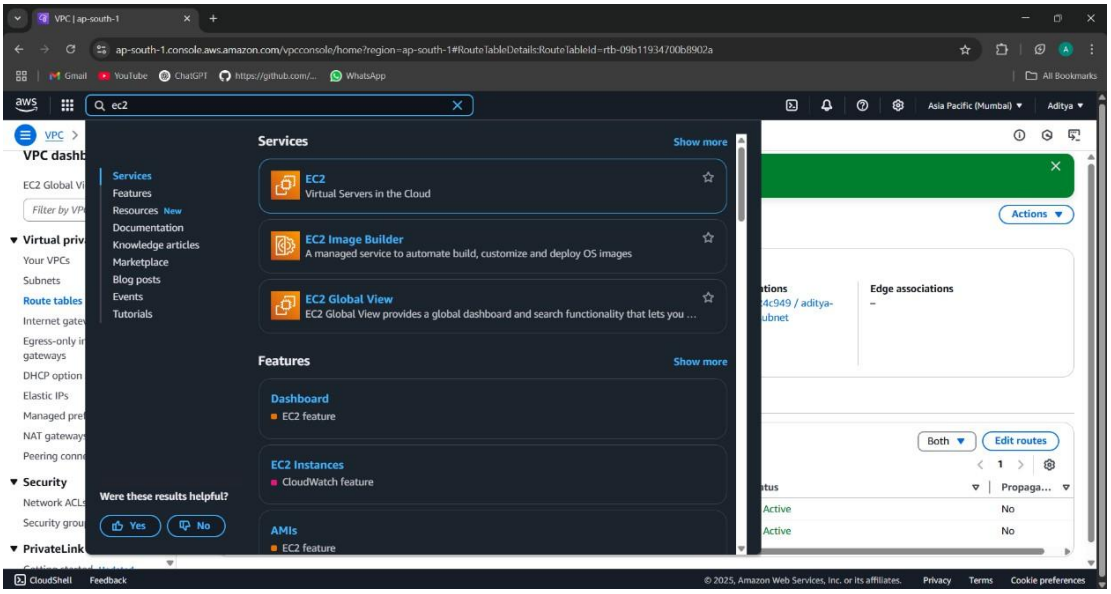
:0.0.0.0/0>SELECT NAT GATEWAY>CHOOSE THE NAT GATEWAY THAT YOU HAVE CREATED> CLICK ON SAVE CHANGES.



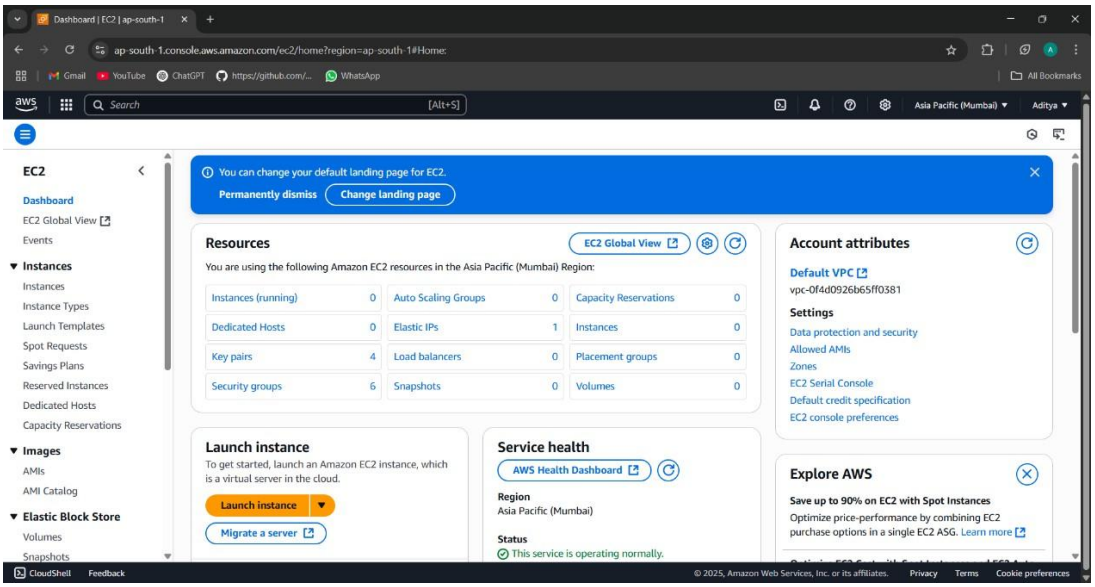
STEP30: PRIVATE SUBNET IS UPDATED WITH NAT GATEWAY



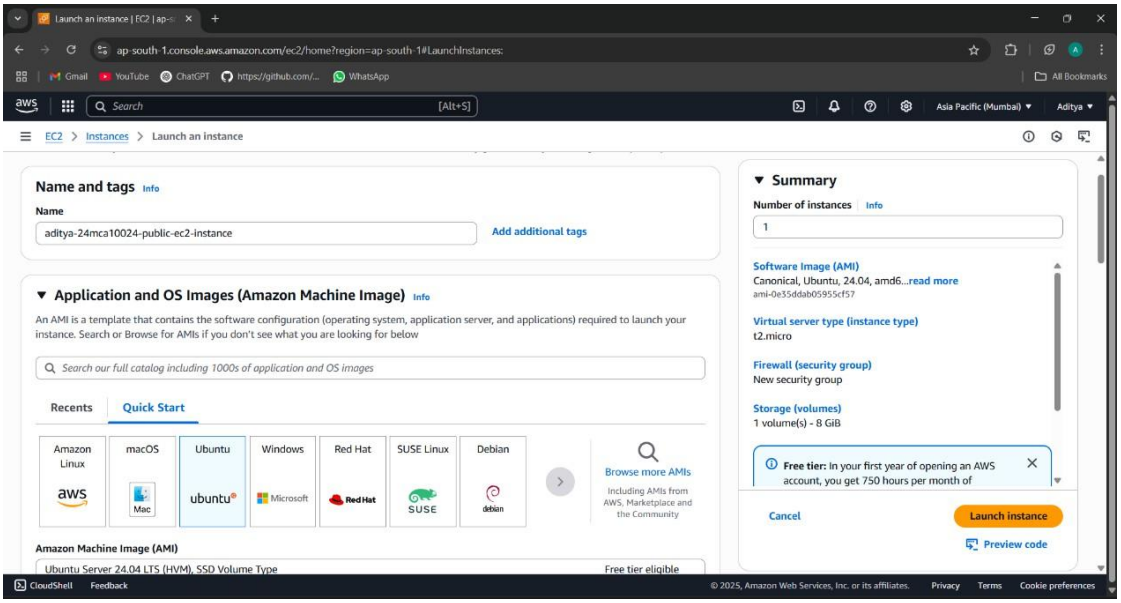
STEP31: WHOLE NETWORK SET UP IS READY, NOW WE HAVE TO CREATE EC2 INSTANCES FOR BOTH PUBLIC AND PRIVATE SUBNET. SEARCH FOR EC2>CLICK ON EC2



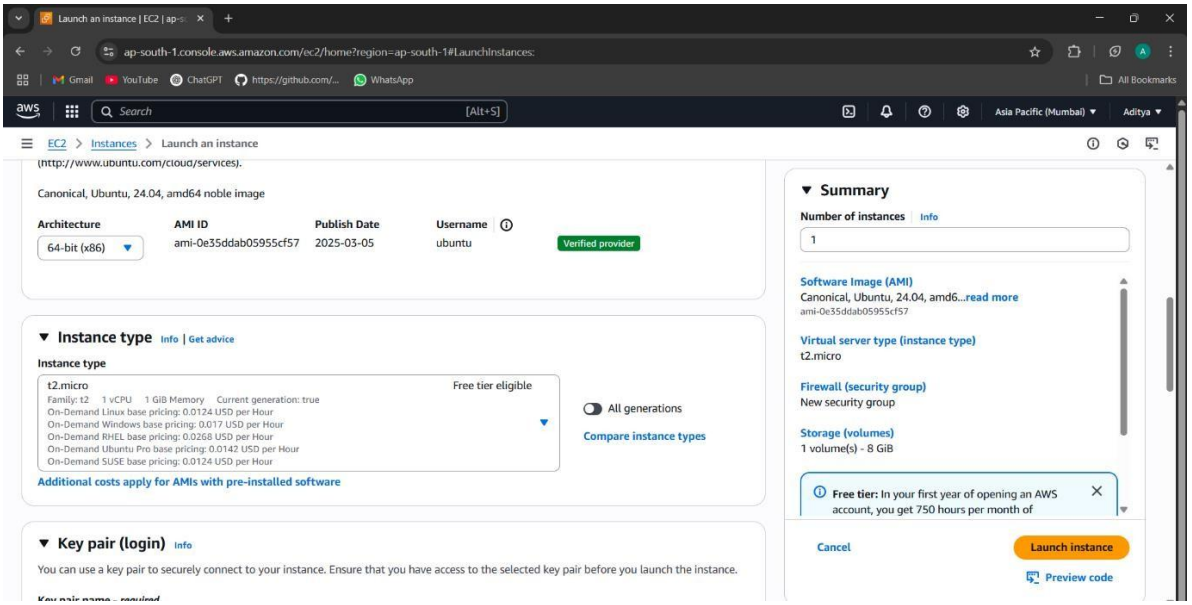
STEP 32: CLICK ON LAUNCH INSTANCE



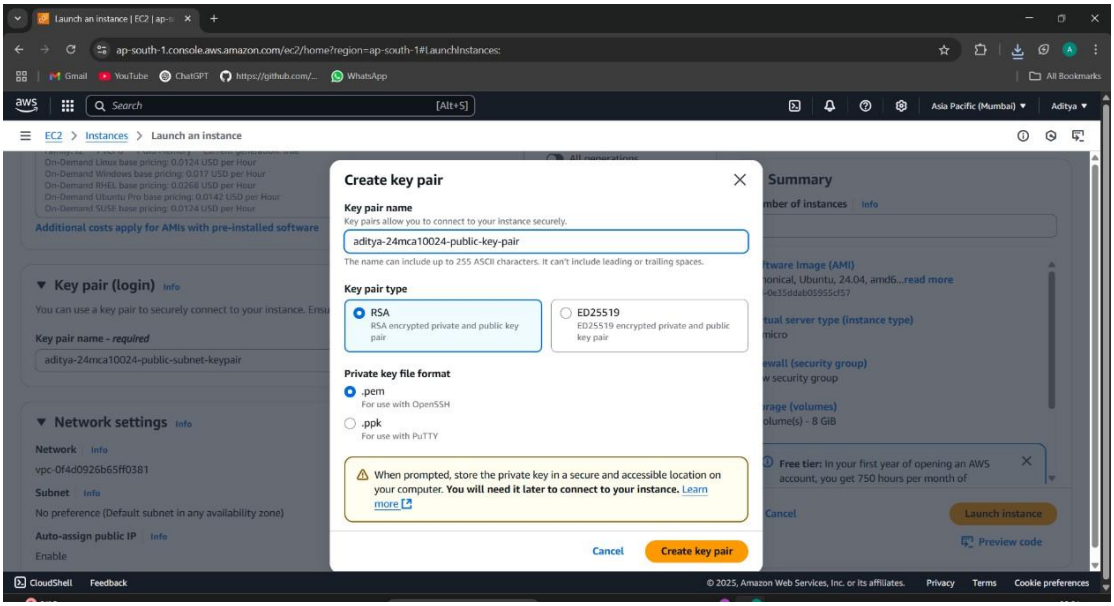
STEP 33: GIVE A SUITABLE INSTANCE NAME>CHOOSE OS:UBUNTU



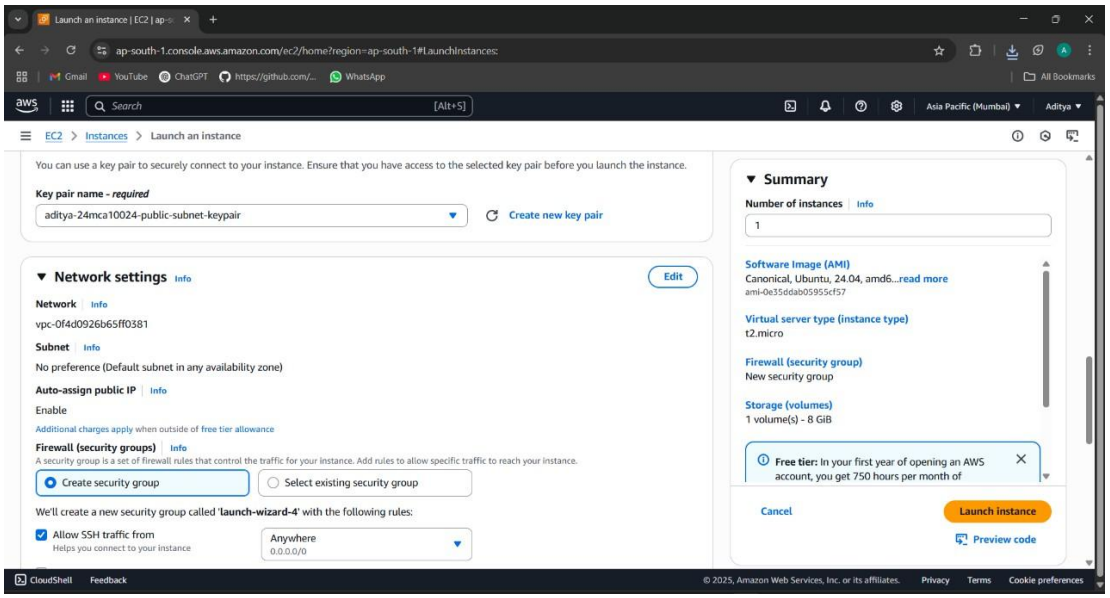
STEP 34: CHOOSE THE DEFAULT ARCHITECTURE> FREE TIER INSTANCE TYPE



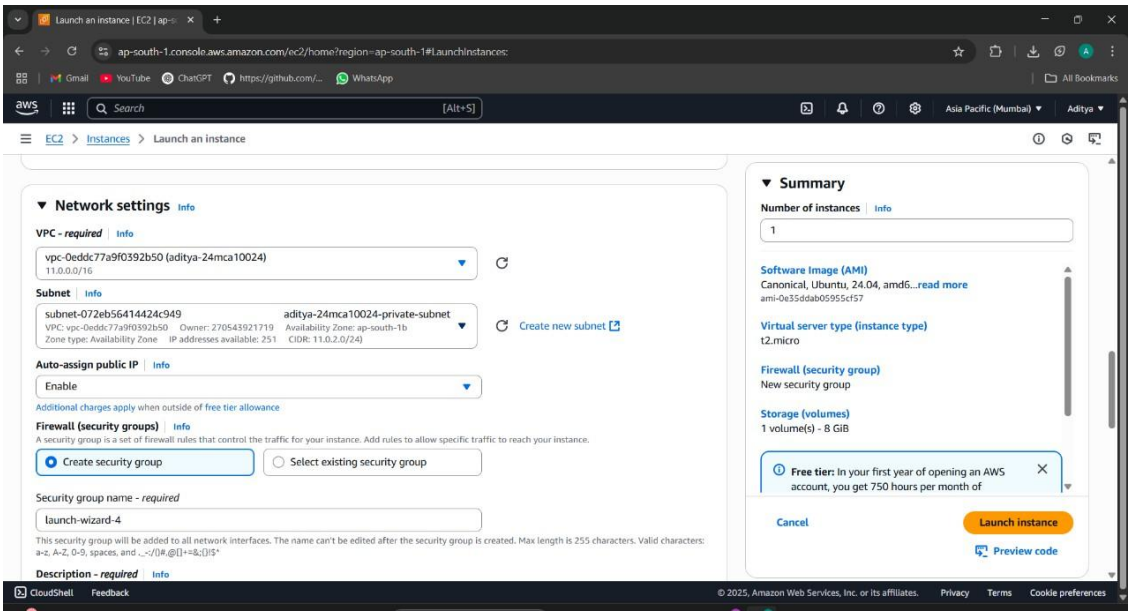
STEP 35: CREATE KEY PAIR



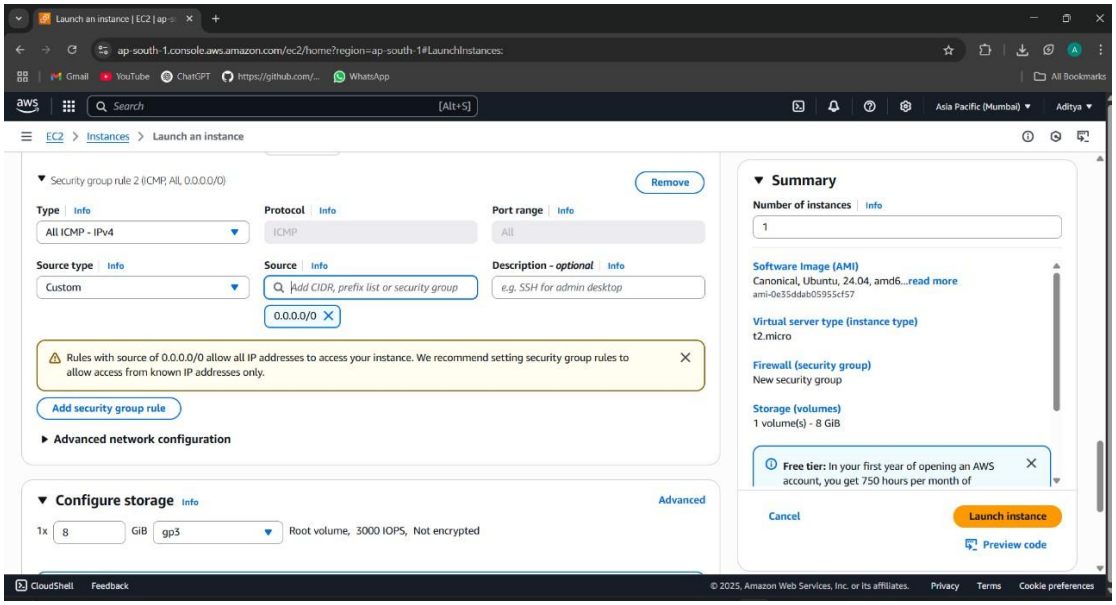
STEP 36: EDIT THE NETWORK SETTING>CHOOSE THE VPC CREATED EARLIER>SELECT THE PUBLIC SUBNET> ENABLE THE AUTO-ASSIGN PUBLIC IP>CHOOSE CREATE SECURITY GROUP



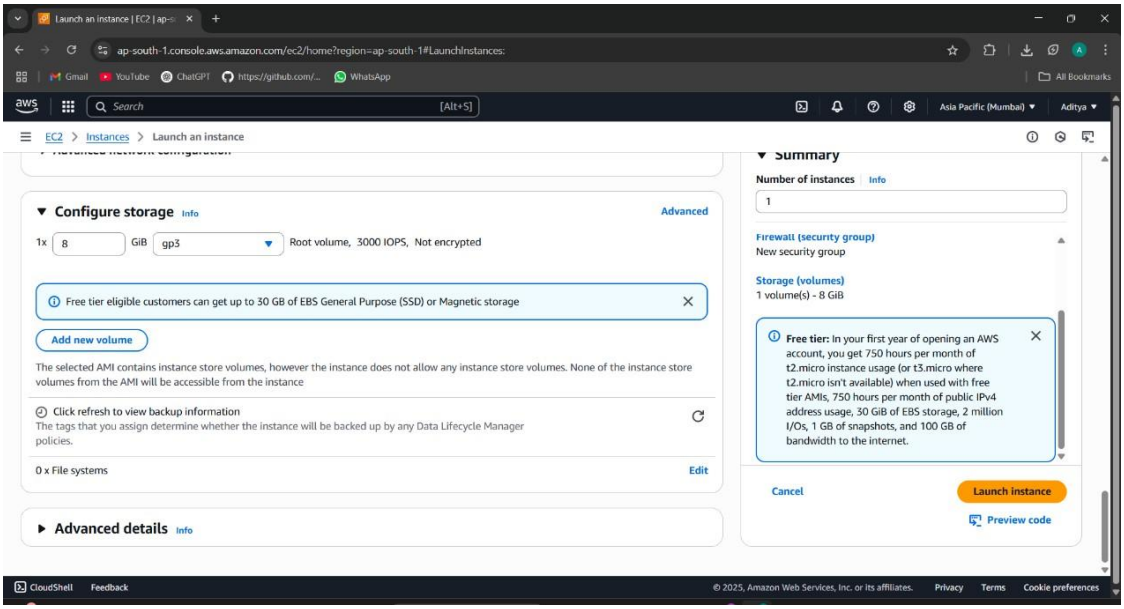
STEP 37: SECURITY GROUP RULE-1 IS DEFAULT> CLICK ON ADD SECURITY GROUP RULE



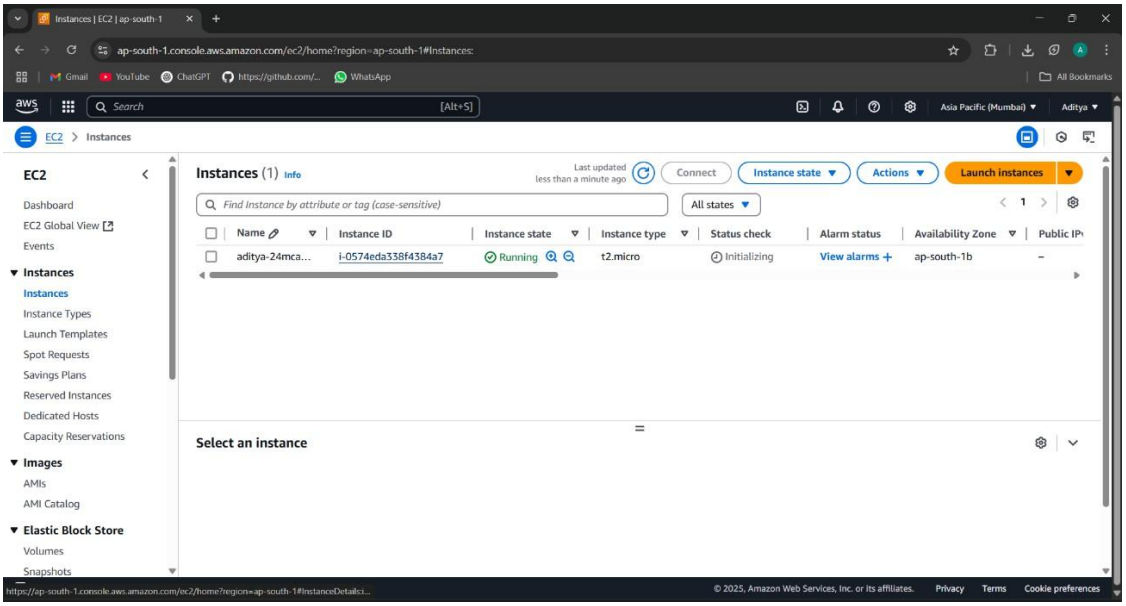
STEP 38: CHOOSE ALL ICMP-IPV4> SOURCE:0.0.0.0/0



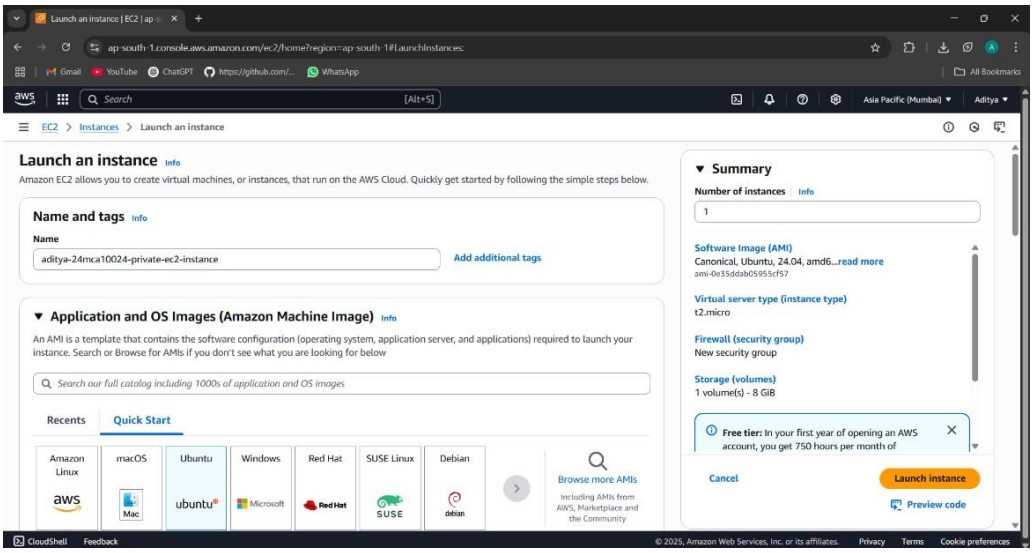
STEP 39: CHOOSE DEFAULT CONFIGURE STORAGE: 8GB>CLICK ON LAUNCH INSTANCE

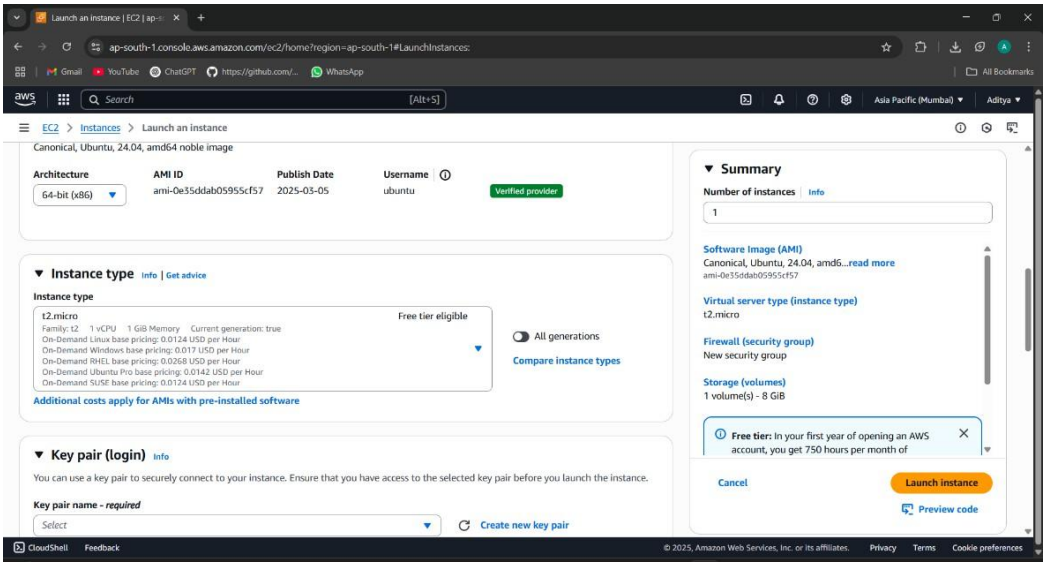


STEP 40: PUBLIC INSTANCE CREATED

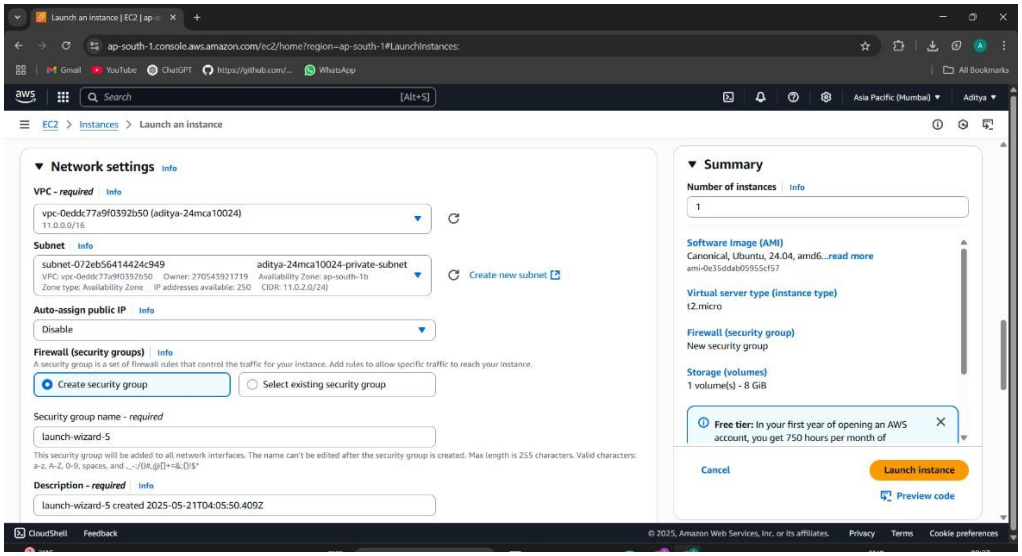


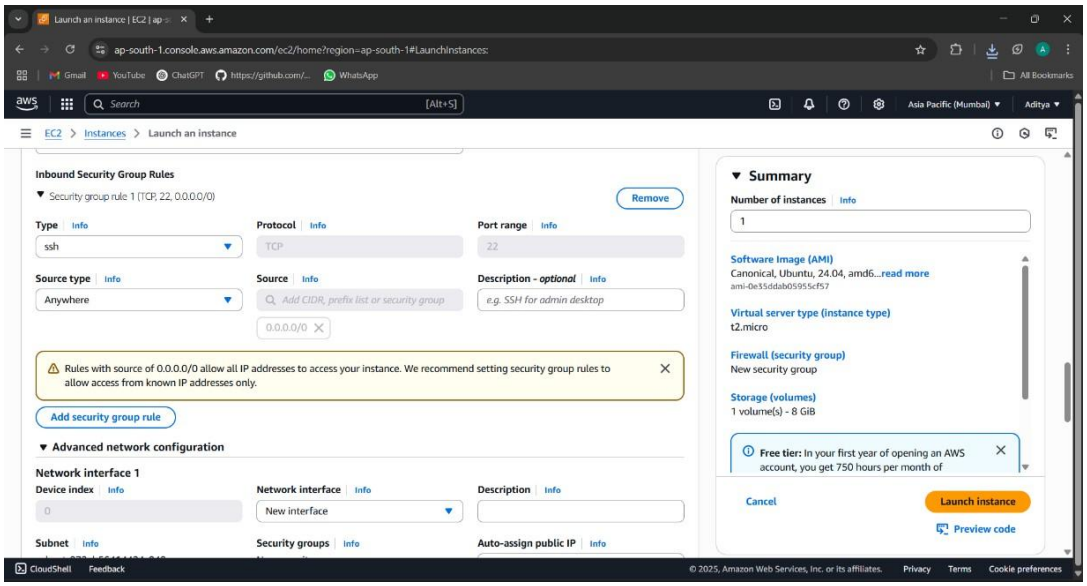
STEP 41: CREATE EC2 INSTANCE FOR PRIVATE SUBNET WITH THE SAME STEPS



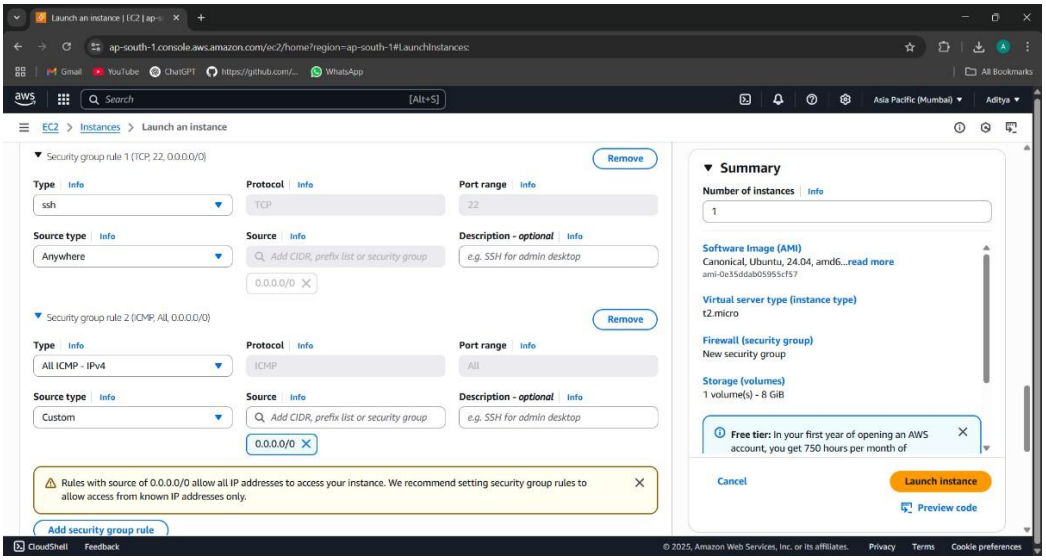


DISABLE AUTO-ASSIGN-PUBLIC IP.

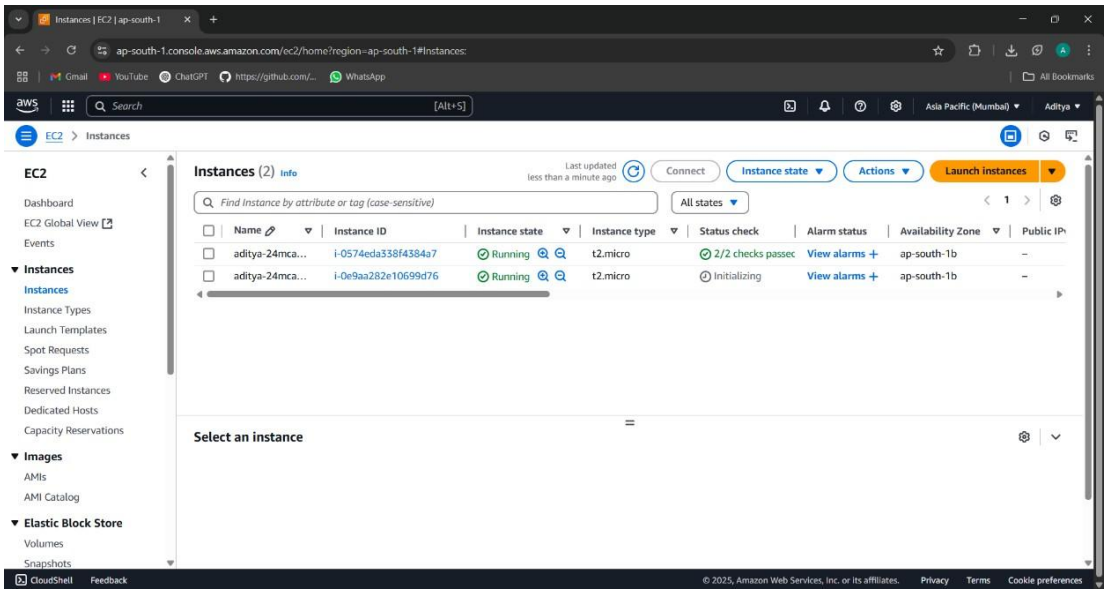




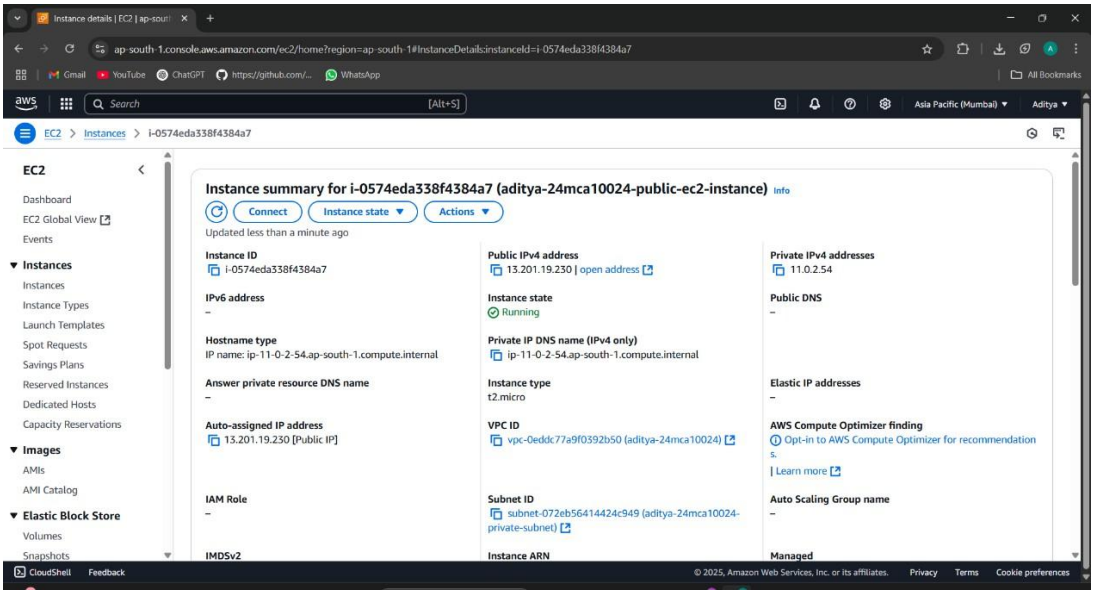
STEP 42: ADD SECURITY GROUP RULE>CHOOSE ALL ICMP-IPV4> SOURCE:11.0.1.0/24(PUBLIC SUBNET IP ADDRESS, COZ ONLY PUBLIC SUBNET CAN ACCESS THE PRIVATE SUBNET)>KEEP DEFAULT STORAGE>CLICK ON LAUNCH INSTANCE



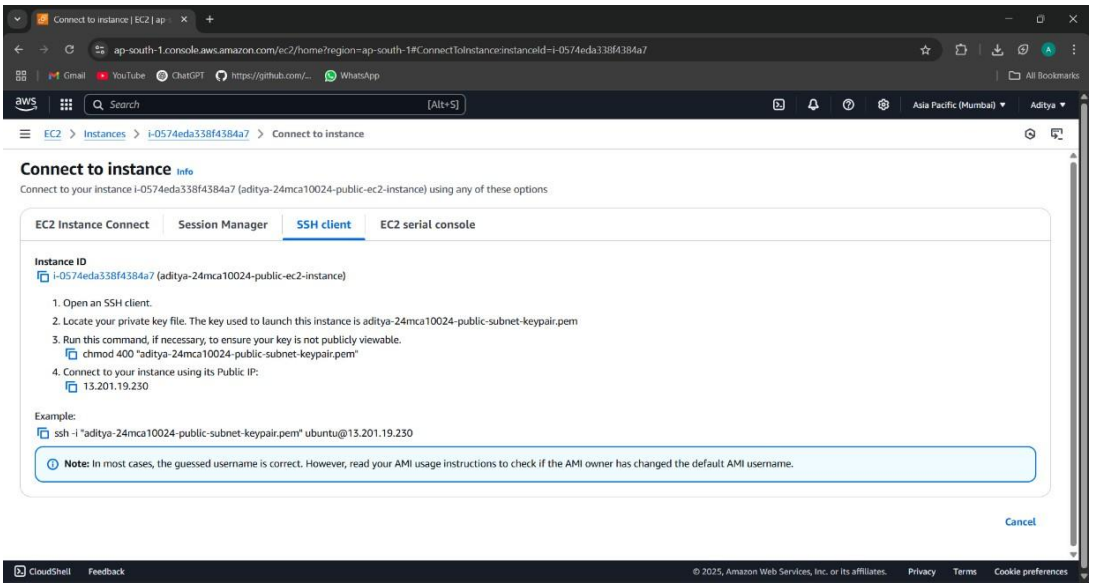
STEP 43: PRIVATE EC2 INSTANCE IS CREATED



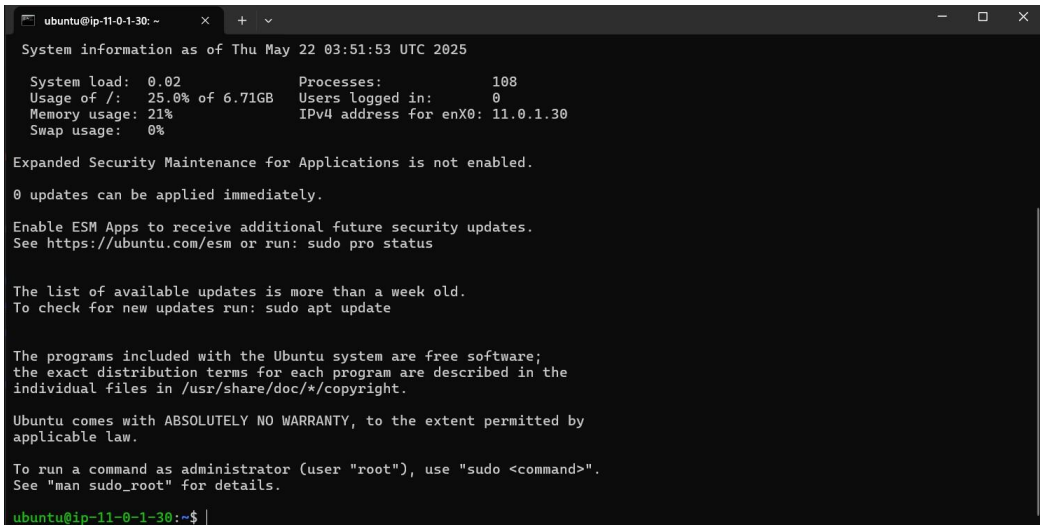
STEP 44: NOW CONNECT THE PUBLIC EC2 INSTANCE.CLICK ON THE PUBLIC INSTANCE> CLICK ON CONNECT



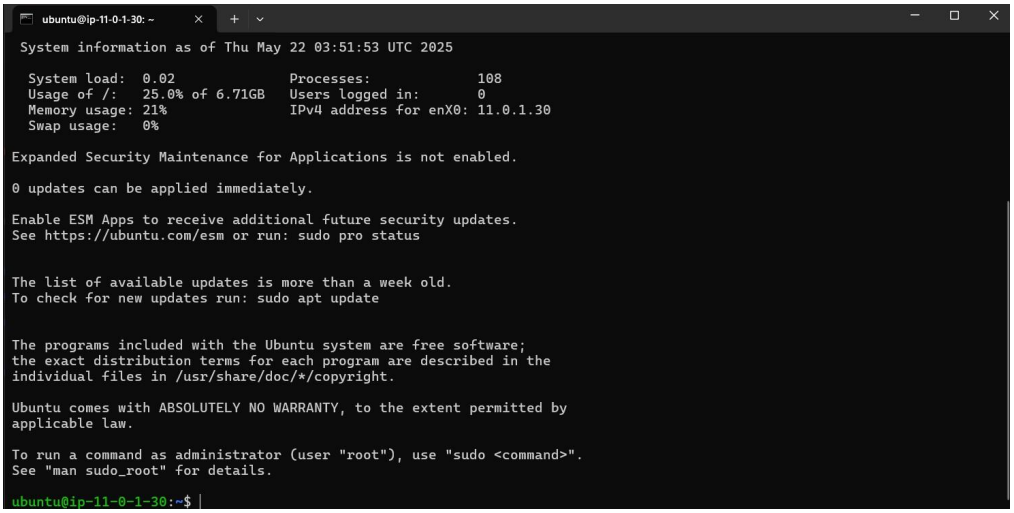
STEP 45: CLICK ON SSH CLIENT> COPY THE EXAMPLE AND PASTE IN TERMINAL TO CONNECT



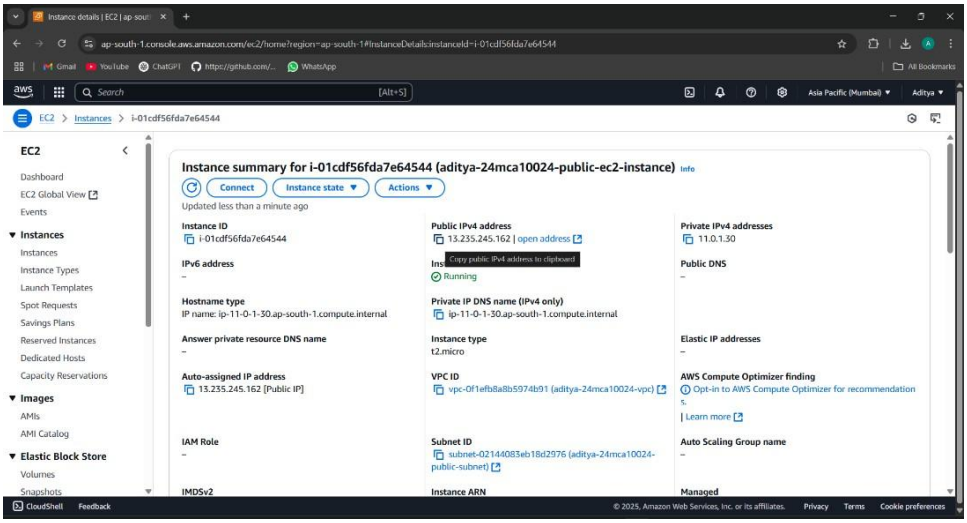
STEP 46: OPEN THE TERMINAL IN YOUR LAPTOP AND TYPE THE FOLLOWING COMMAND> NOW PASTE THE COMMAND HERE FROM THE SSH CLIENT

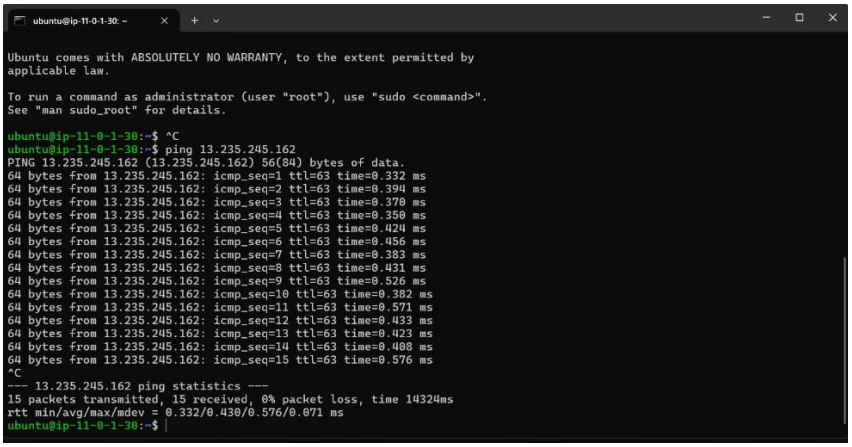


STEP 47: PUBLIC EC2 INSTANCE IS CONNECTED



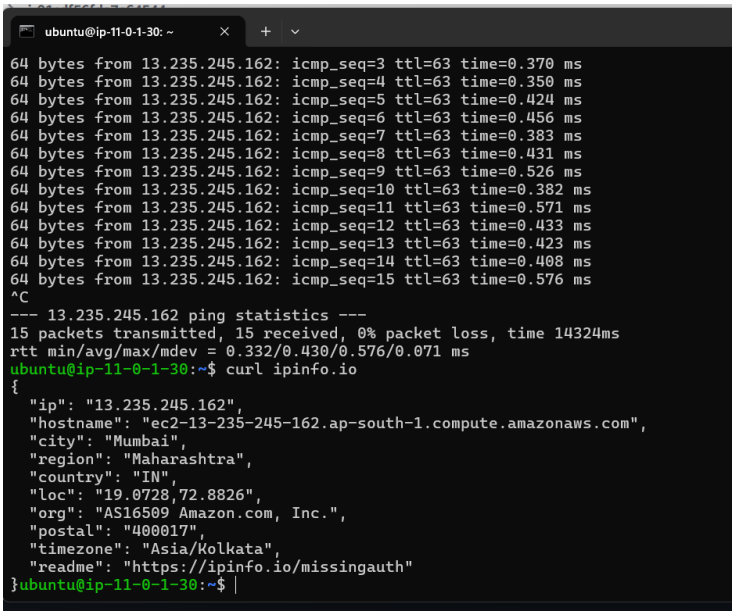
STEP 48: NOW PING THE PUBLIC EC2 INSTANCE WITH THE HELP OF YOUR LOCAL TERMINAL. GO TO INSTANCE AND COPY ITS IP ADDRESS AND TYPE THE FOLLOWING COMMAND. (IT MEANS FROM OUTSIDE AWS, WE ARE ABLE TO CONNECT TO THE PUBLIC EC2 INSTANCE)





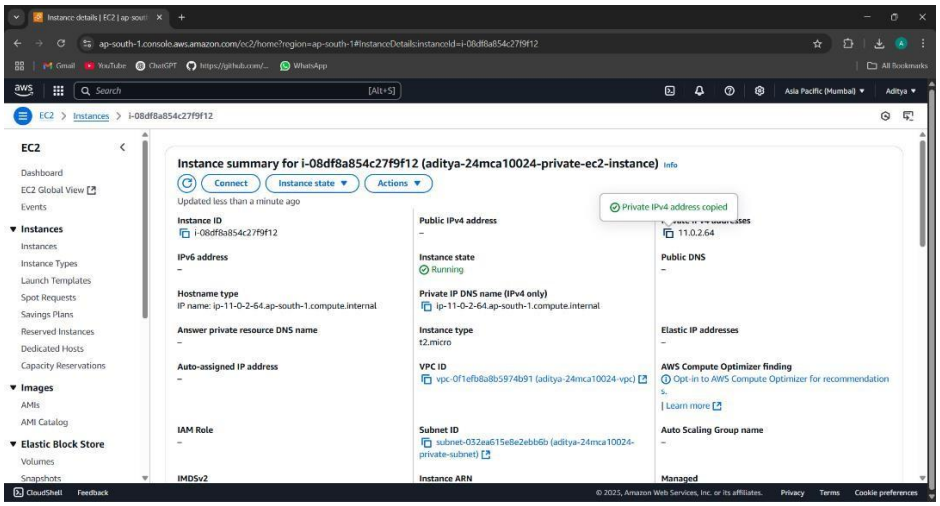
STEP 49: NOW NEXT FROM PUBLIC EC2 INSTANCE TERMINAL, TYPE THE FOLLOWING COMMAND TO ACCESS RESOURCE FROM OUTSIDE THE AWS

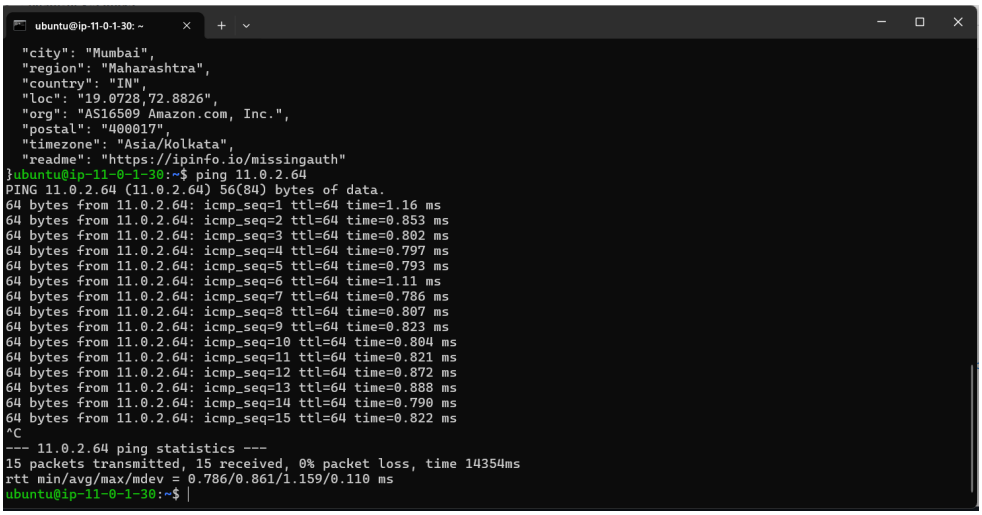
NOTE: curl ipinfo.io: It is a command used in the terminal to retrieve information about your current IP address using the "curl" tool and the "ipinfo.io" web service; essentially, it fetches details like your location, ISP, and public IP address by sending a request to the IPinfo.io API via the curl command.



It is seen that public EC2 instance is accessible from outside world as well as it can also access outside resources. But we know that, private instance that we have created under private subnet is not accessible from outside world but it can be accessed by the public EC2 instance only.

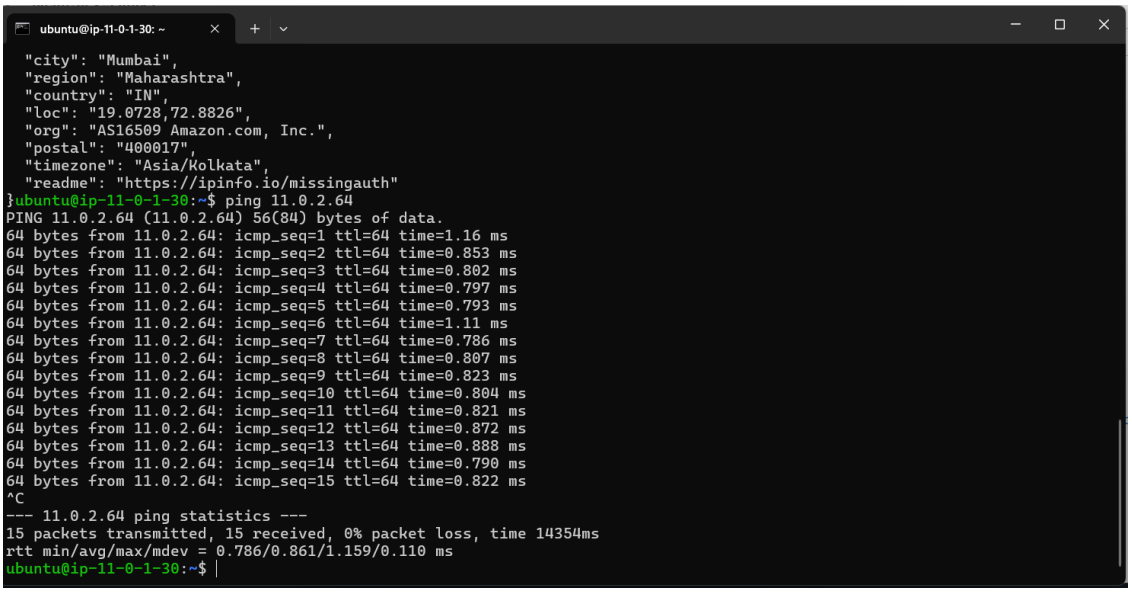
STEP 50: PING PRIVATE EC2 INSTANCE FROM THE LOCAL TERMINAL. COPY THE PRIVATE IP ADDRESS OF THE INSTANCE AND TYPE (PING 11.0.2.160) IN THE LOCAL TERMINAL.





ITS SHOWING REQUEST TIMED OUT. i.e., IT IS UNABLE TO CONNECT TO THE PRIVATE EC2 INSTANCE FROM THE LOCAL TERMINAL. NOW LET'S TRY TO CONNECT IT FROM THE PUBLIC EC2 INSTANCE TERMINAL.

STEP 50: TYPE THE FOLLOWING COMMAND TO CONNECT TO THE PRIVATE EC2 INSTANCE (PING 11.0.2.160). CONNECTION ESTABLISHED AS SHOWN BELOW.

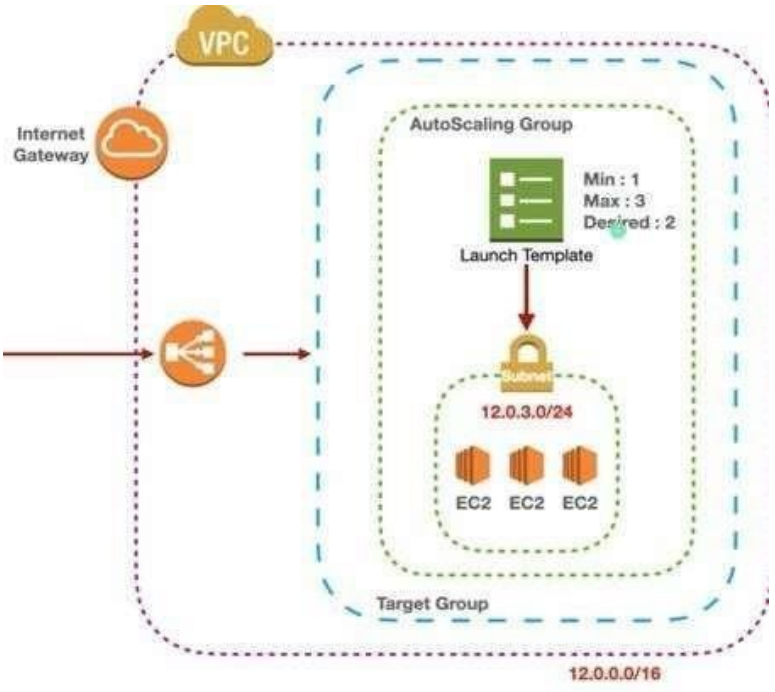


RELEASE ALL THE RESOURCES THAT ARE USED IN THE EXECUTION OF THE LAB.

1. **TERMINATE THE EC2 INSTANCES.**
2. **DELETE ROUTE TABLE:**
   a. **ROUTE TABLE>EDIT ROUTE>REMOVE NAT GATEWAY (PRIVATE)**
   b. **ROUTE TABLE>EDIT ROUTE>REMOVE INTERNET GATEWAY (PUBLIC)**
   c. **ROUTE TABLE>SUBNET ASSOCIATIONS>EDIT SUBNET ASSOCIATIONS>DESELECT>SAVE CHANGES**
   d. **SELECT BOTH THE ROUTE TABLES>ACTIONS>DELETE ROUTE TABLES.**
3. **DELETE SUBNETS:**
   a. **SUBNETS> SELECT BOTH THE SUBNET ACTIONS>ACTIONS> DELETE SUBNET**
4. **DELETE NAT GATEWAY:**
   a. **NAT GATEWAY>SELECT THE NAT GATEWAY>ACTIONS> DELETE NAT GATEWAY**
5. **DELETE VPC**
   a. **VPC>SELECT THE VPC CREATED>ACTIONS>DELETE**
6. **RELEASE ELASTIC IP**
   a. **ELASTIC IP>SELECT THE ELASTIC IP>ACTIONS>RELEASE THE ELASTIC IP**

Date: 23/05/2025	Title
Exp. No: 06	Show to implement load balancer, target group & auto=scaling group with EC2 Instance.

SHOW TO IMPLEMENT LOAD BALANCER, TARGET GROUP & AUTO-SCALING GROUP WITH EC2 INSTANCE



In AWS, combining a Load Balancer, Target Group, and Auto Scaling Group (ASG) provides a powerful and automated solution for managing application availability, scalability, and fault tolerance.

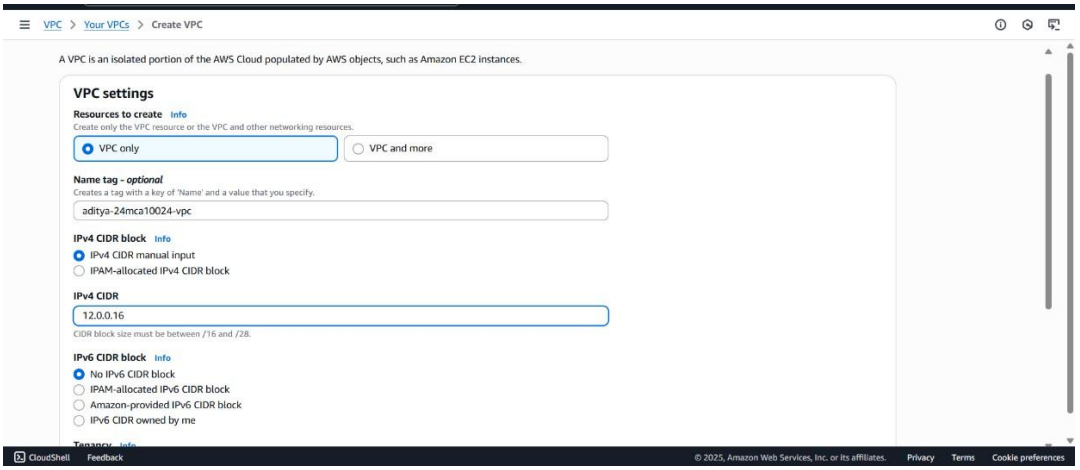
An Application Load Balancer (ALB) distributes incoming traffic across multiple EC2 instances based on routing rules. It improves performance and availability by ensuring that traffic is routed only to healthy instances. The Load Balancer listens on specified ports (like HTTP on port 80 or HTTPS on port 443) and forwards requests to registered targets.

A Target Group is a logical group of EC2 instances (or other resources) that receive traffic from the Load Balancer. Each target group is associated with a specific protocol and port, and includes health check settings to monitor the availability of its targets. EC2 instances are registered to a target group so that they can be used by the load balancer.

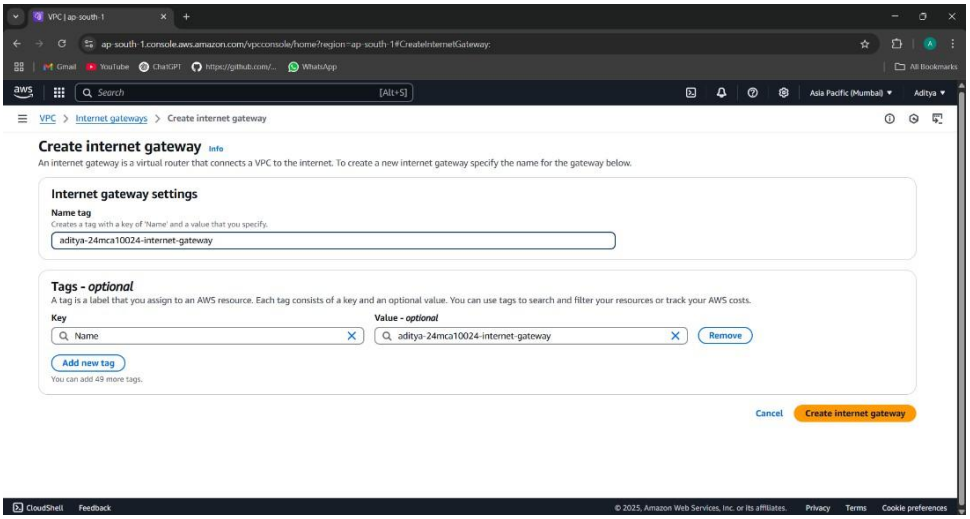
An Auto Scaling Group (ASG) automatically manages the number of EC2 instances based on scaling policies. It ensures that a minimum number of instances are always running and can scale out (add instances) or scale in (remove instances) based on metrics such as CPU usage, memory utilization, or request rate. The ASG launches EC2 instances using a Launch Template or Launch Configuration, and registers them with the appropriate Target Group so that they can receive traffic from the Load Balancer.

This architecture ensures high availability and fault tolerance by distributing load evenly and replacing unhealthy instances automatically. It also optimizes cost and performance by scaling the infrastructure dynamically according to demand. This combination is ideal for modern web applications, APIs, and microservices deployed in a cloud environment.

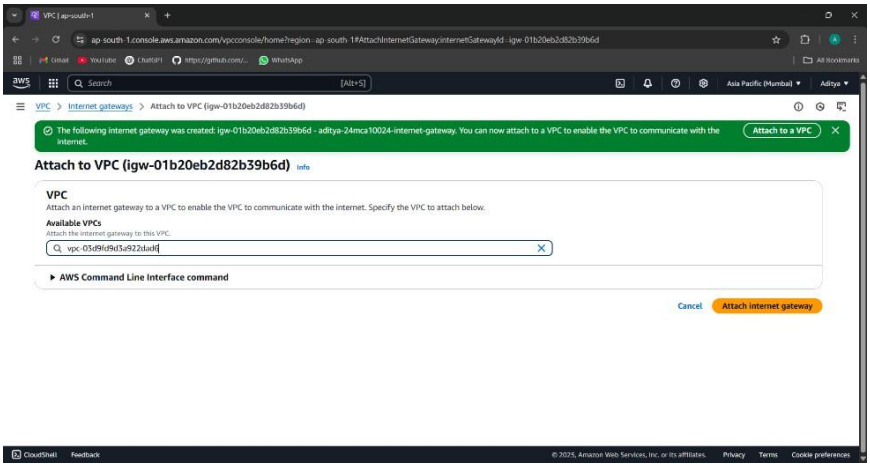
STEP1: CREATE A VPC>WRITE VPC NAME>IPv4 CIDR:12.0.0.0/16>CREATE VPC



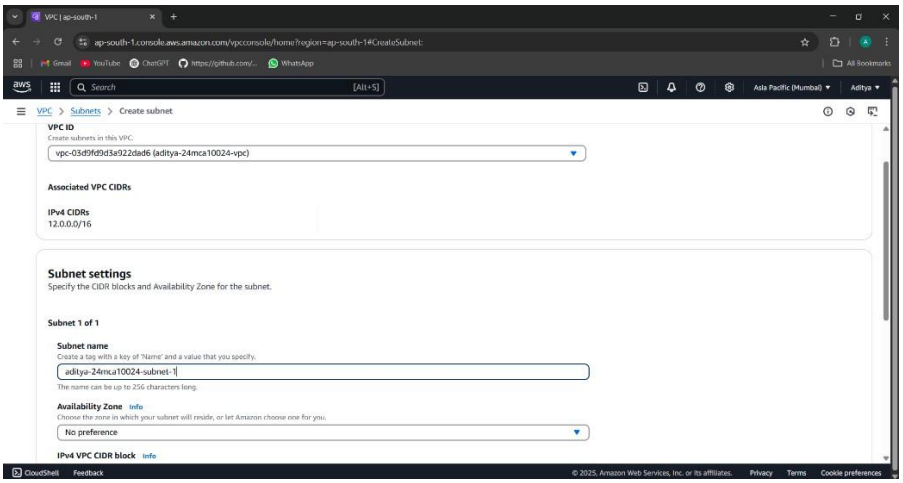
STEP2: NEXT CLICK ON INTERNET GATEWAY> CREATE INTERNET GATEWAY>TYPE NAME> CLICK ON CREATE



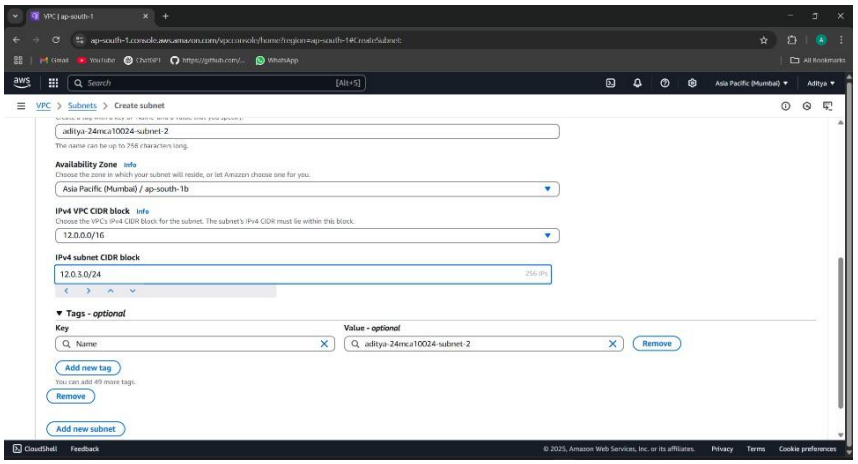
STEP3: NEXT CLICK ON ATTACH TO VPC> CHOOSE THE INTERNET GATEWAY>CLICK ON ATTACH



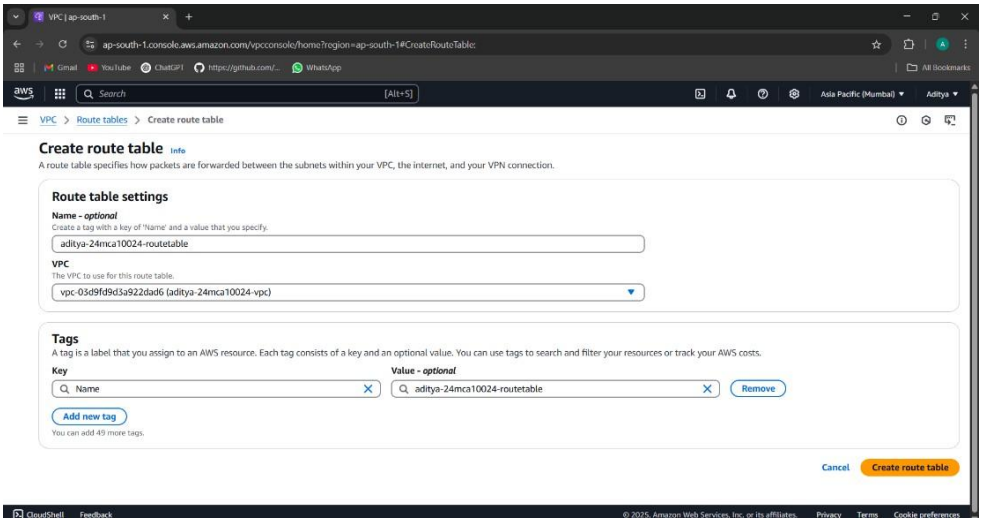
STEP4: CREATE SUBNET> SELECT THE VPC>TYPE THE NAME OF SUBNET_1>AVAILABILITY ZONE:ap-south 1a>IPv4 CIDR:12.0.1.0/24



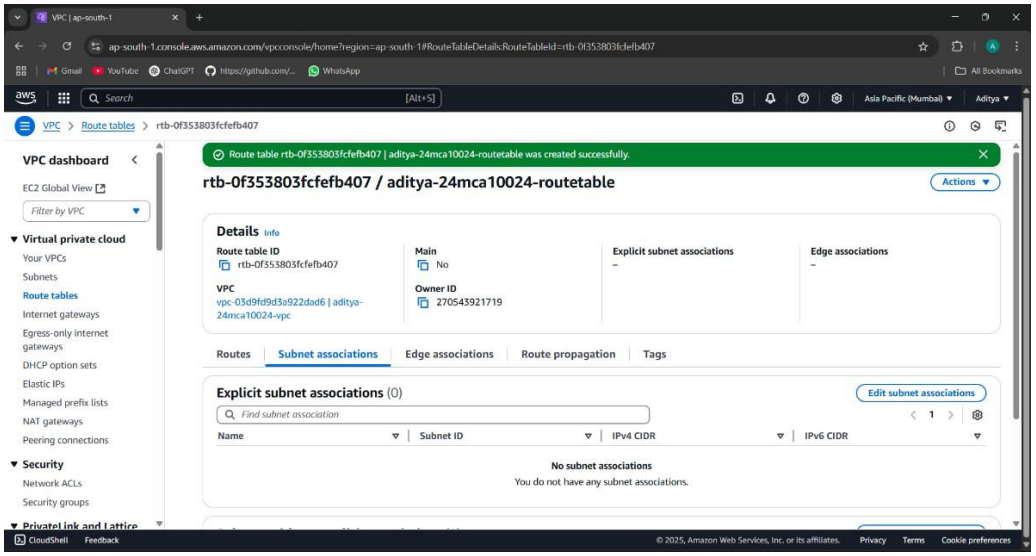
STEP5: CLICK ON ADD SUBNET>TYPE THE NAME ODF SUBNET_2>AVAILABILITY ZONE:ap-south 1b>IPv4 CIDR:12.0.3.0/24



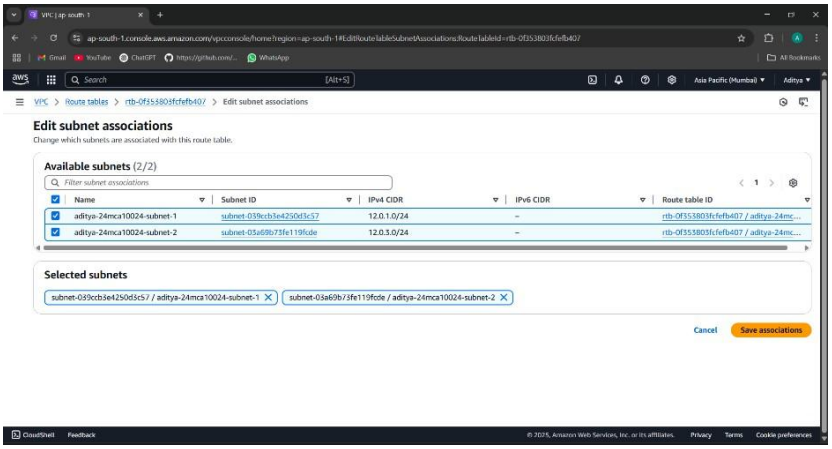
STEP6: NOW CREATE ROUTE TABLE: CLICK ON ROUTE TABLES>CREATE ROUTE TABLE>ROUTE TABLE NAME> CHOOSE VPC>CREATE ROUTE TABLE



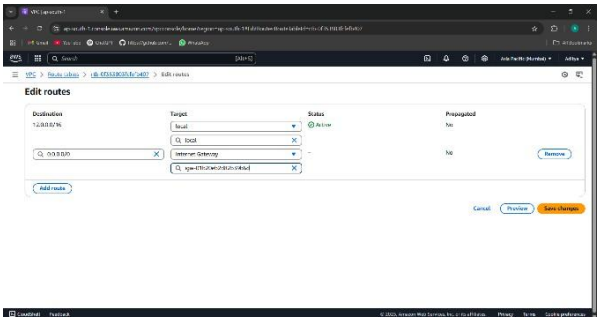
STEP7: NOW ASSOCIATE ROUTE TABLE WITH THE SUBNETS: CLICK ON SUBNET ASSOCIATIONS>EDIT SUBNET ASSOCIATION



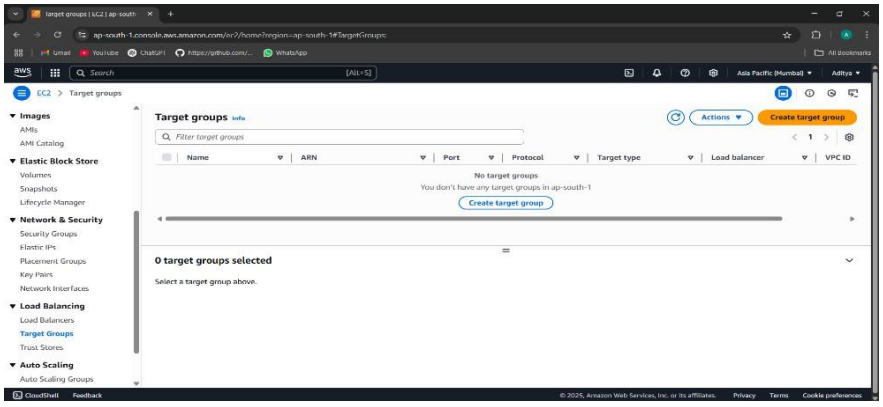
STEP8: SELECT BOTH THE SUBNETS>SAVE ASSOCIATIONS



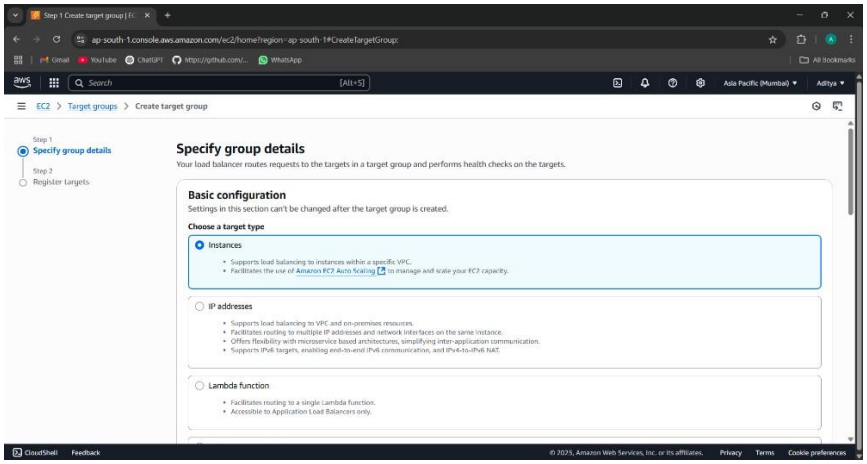
STEP9: CLICK ON ROUTE>EDIT ROUTE>ADD ROUTE> SELECT THE DATA AS SHOWN>SAVE CHANGES



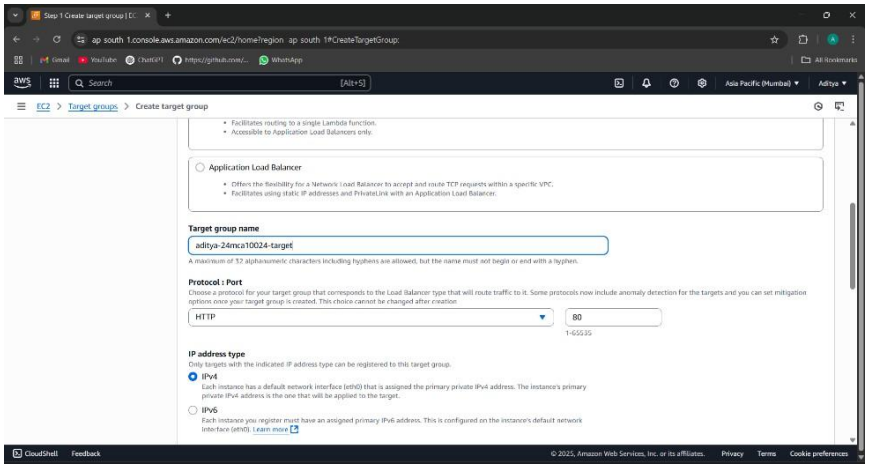
STEP10: GO TO EC2 DASHBOARD>CLICK ON TARGET GROUPS



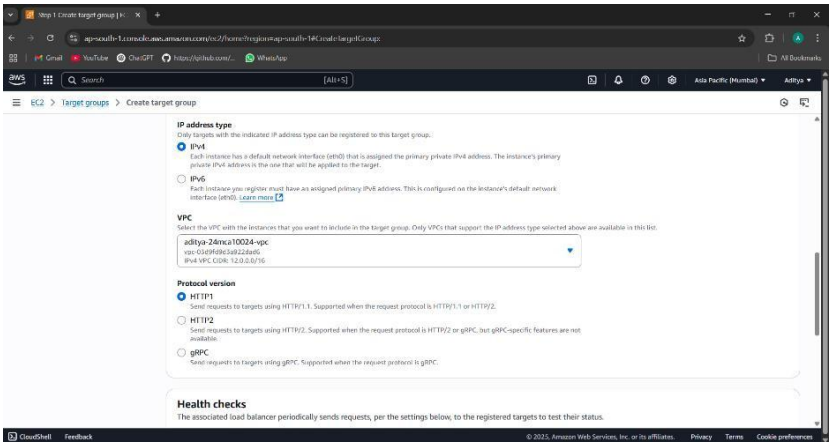
STEP11: CHOOSE INSTANCES


Step 1 Create target group | EC
ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#CreateTargetGroup:
Gmail
YouTube
ChatGPT
https://github.com/...
WhatsApp
All Bookmarks
aws
Search
[Alt+S]
Asia Pacific (Mumbai)
Aditya
EC2 > Target groups > Create target group
Step 1
Specify group details
Step 2
Register targets
Specify group details
Your load balancer routes requests to the targets in a target group and performs health checks on the targets.
Basic configuration
Settings in this section can't be changed after the target group is created.
Choose a target type
Instances
Supports load balancing to instances within a specific VPC.
Facilitates the use of Amazon EC2 Auto Scaling to manage and scale your EC2 capacity.
IP addresses
Supports load balancing to VPC and on-premises resources.
Facilitates routing to multiple IP addresses and network interfaces on the same instance.
Offers flexibility with microservice based architectures, simplifying inter-application communication.
Supports IPv6 targets, enabling end-to-end IPv6 communication, and IPv4-to-IPv6 NAT.
Lambda function
Facilitates routing to a single Lambda function.
Accessible to Application Load Balancers only.
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

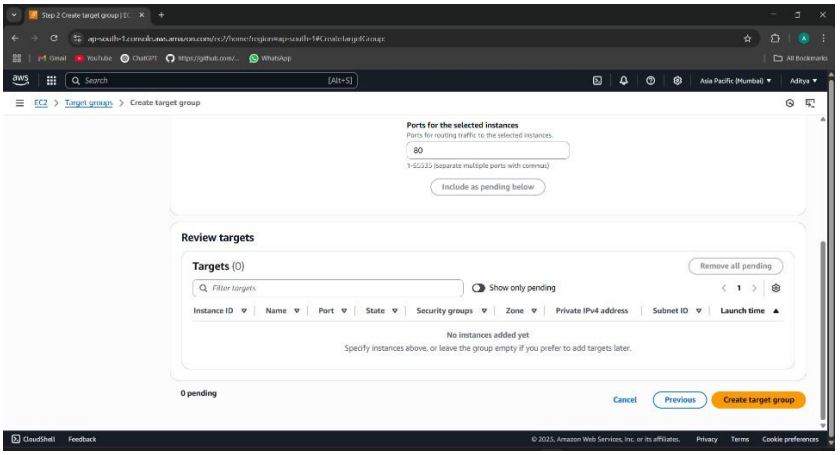
STEP12: WRITE A NAME TO THE TARGET GROUP



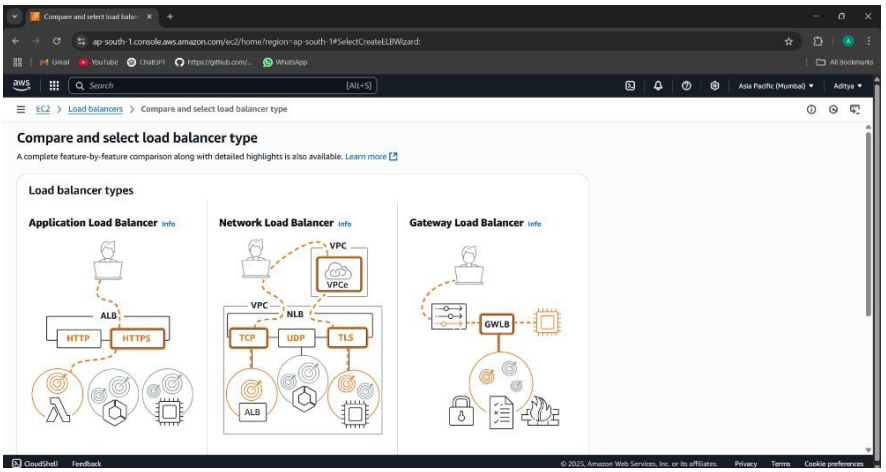
STEP13: SELECT THE VPC> CLICK ON NEXT



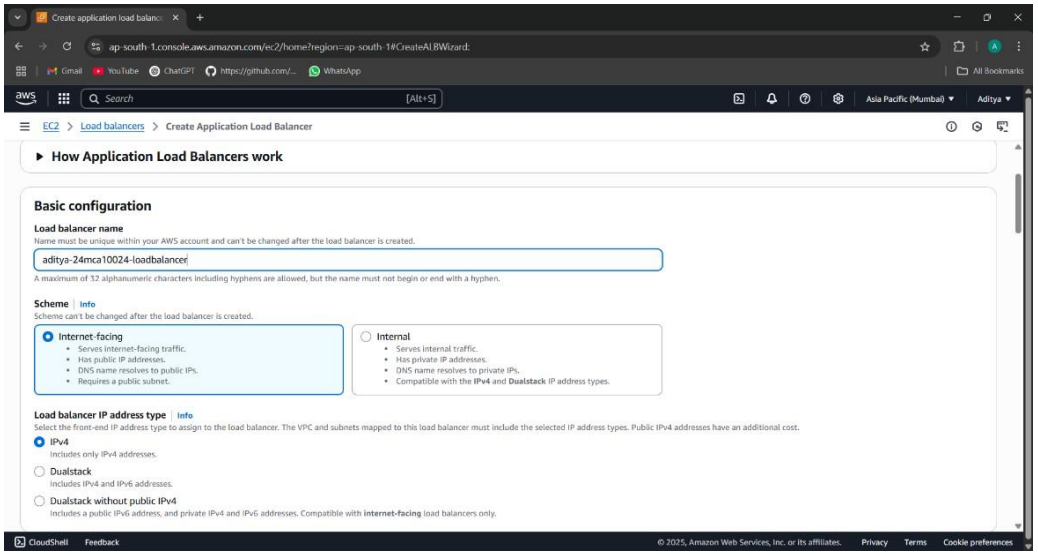
STEP14: CLICK ON CREATE TARGET GROUP



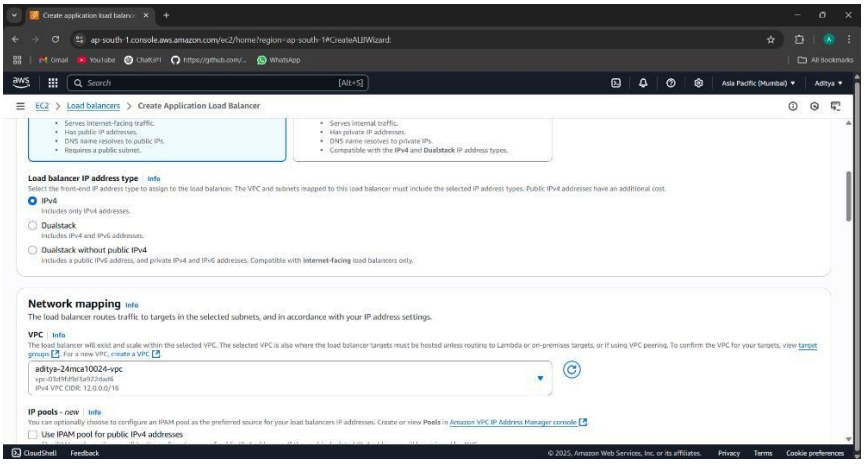
STEP15: CLICK ON APPLICATION LOAD BALANCER



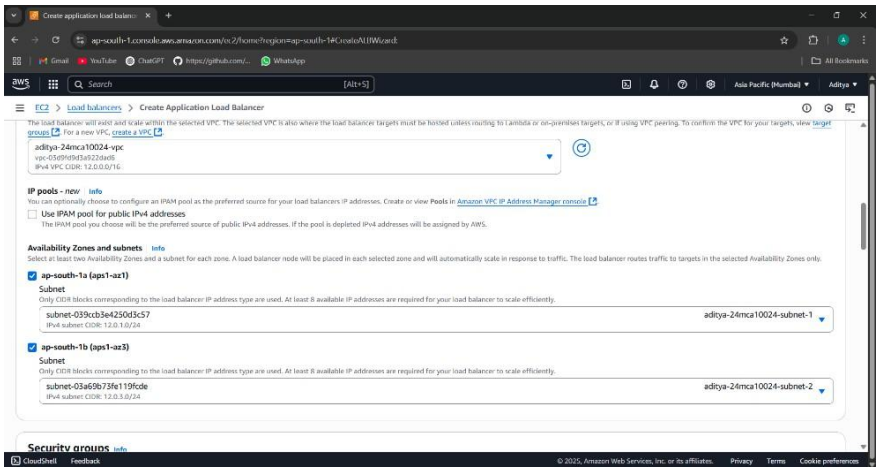
STEP16: WRITE THE NAME OF THE LOAD BALANCER



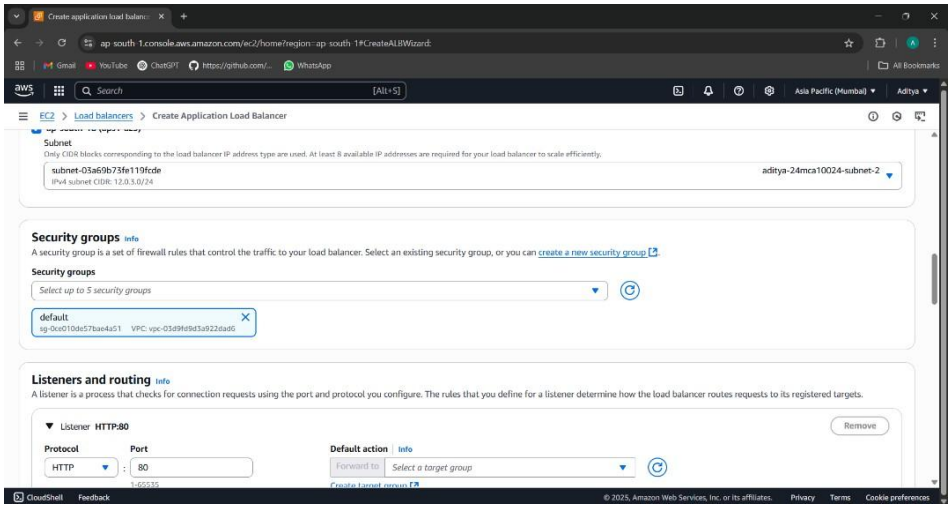
STEP17: SELECT THE VPC CREATED



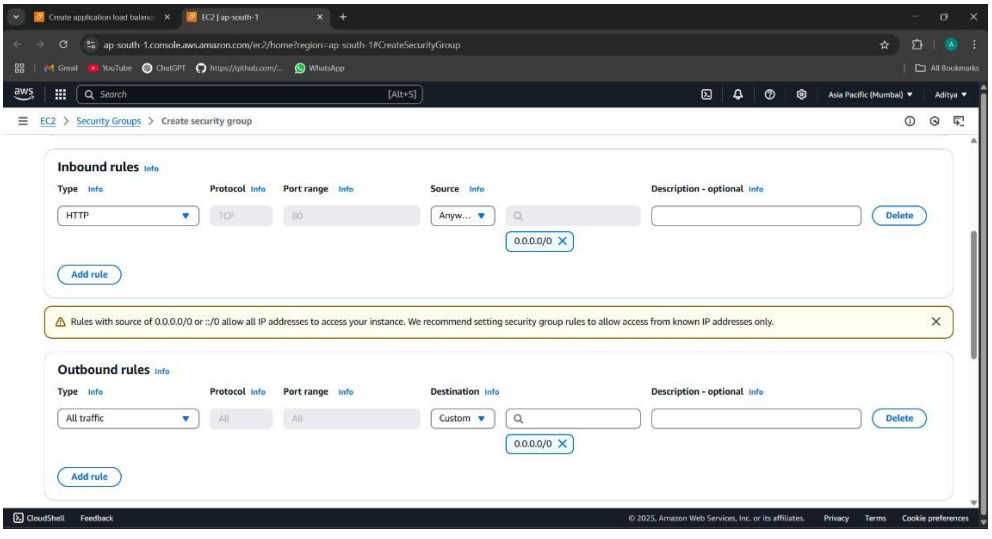
STEP18: SELECT BOTH THE ZONE(SUBNETS)



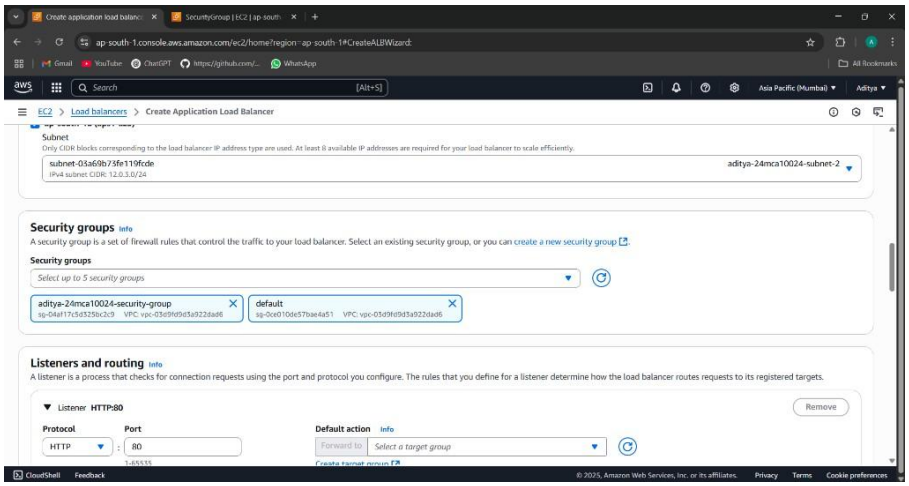
STEP19: CLICK ON CREATE A NEW SECURITY GROUP> TYPE THE NAME AND DESCRIPTON OF THE SECURITY GROUP>SELECT THE VPC



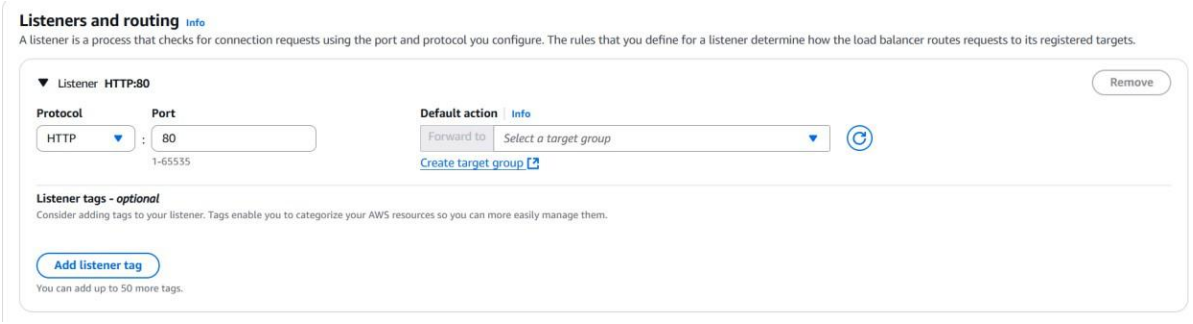
STEP20: SET THE IN BOUND RULE: CLICK ON ADD RULES> SET HTTP>SOURCE:0.0.0.0/0> CLICK ON CREATE SECURITY GROUP.



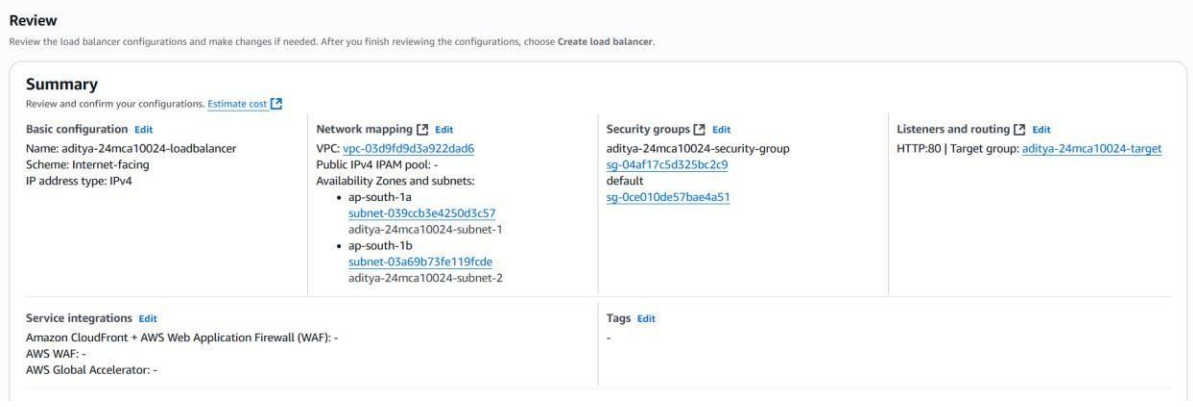
STEP21: NOW SELECT THE SECURITY GROUP THAT YOU HAVE CREATED



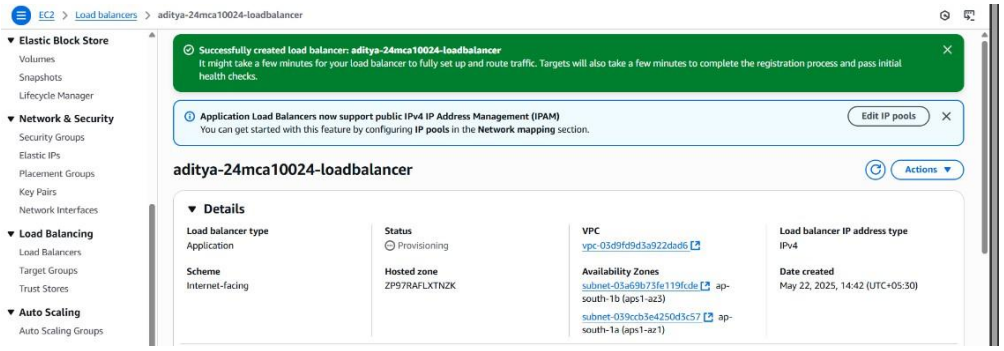
STEP22: LISTENERS & ROUTING SECTION: SELECT TARGET GROUP THAT YOU HAVE CREATED



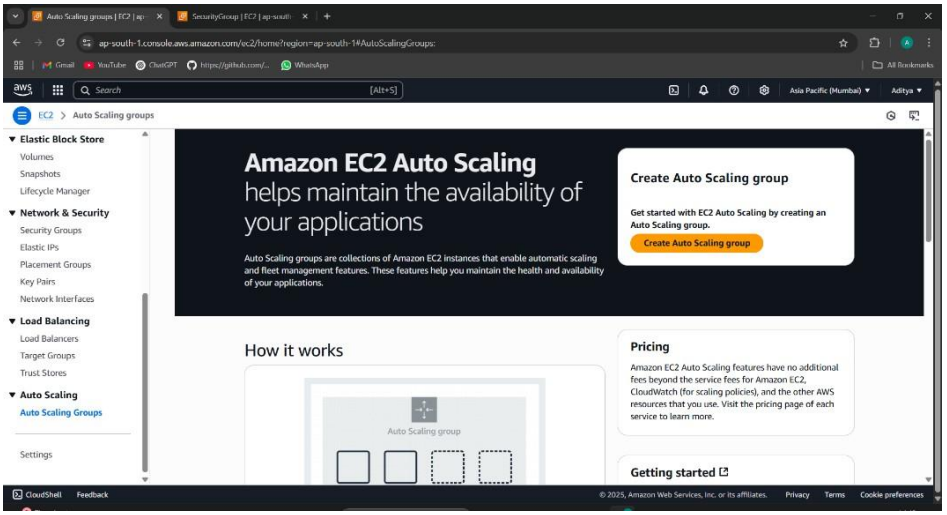
STEP23: VERIFY THE DETAILS IN THE SUMMARY>CLICK ON CREATE LOAD BALANCER



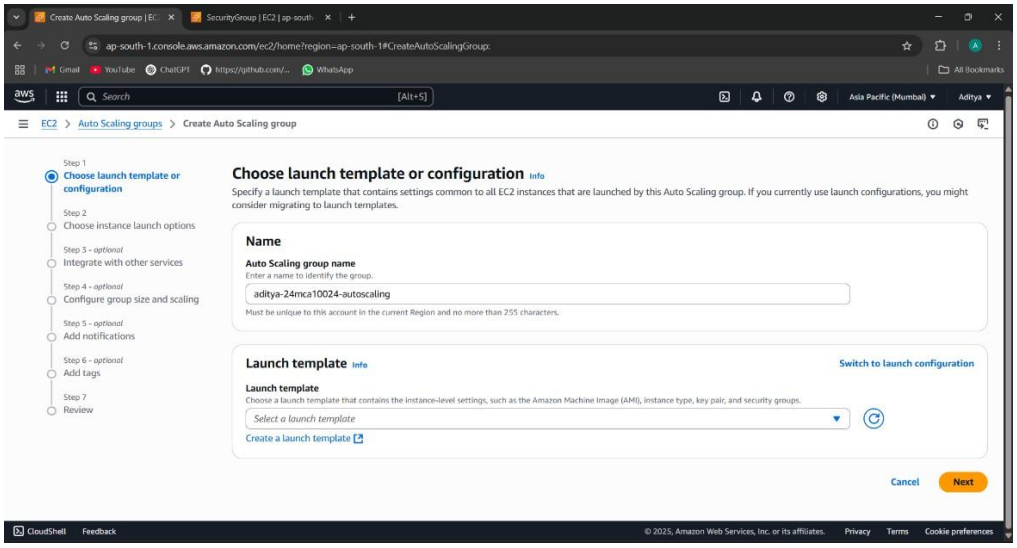
STEP24: LOAD BALANCER IS CREATED



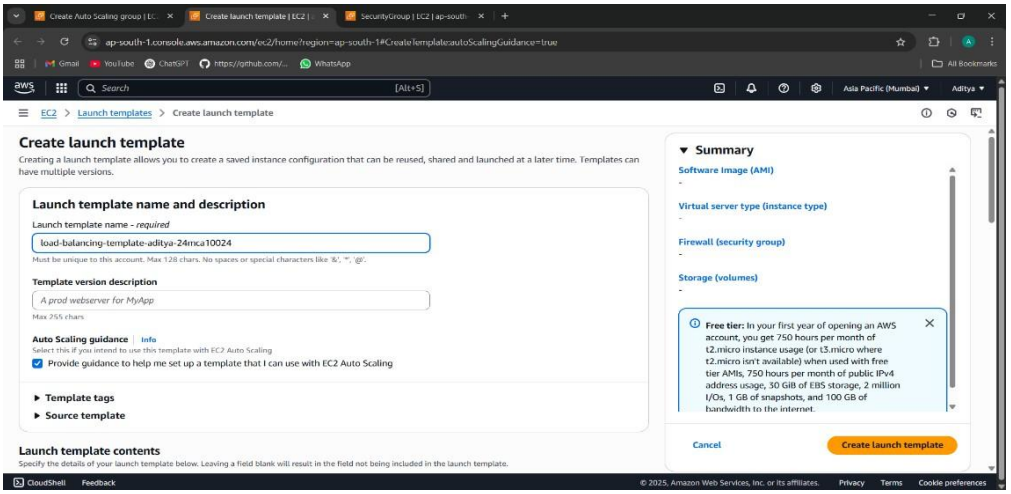
STEP25: CLICK ON AUTO-SCALING GROUP> CREATE AUTO-SCALING GROUP



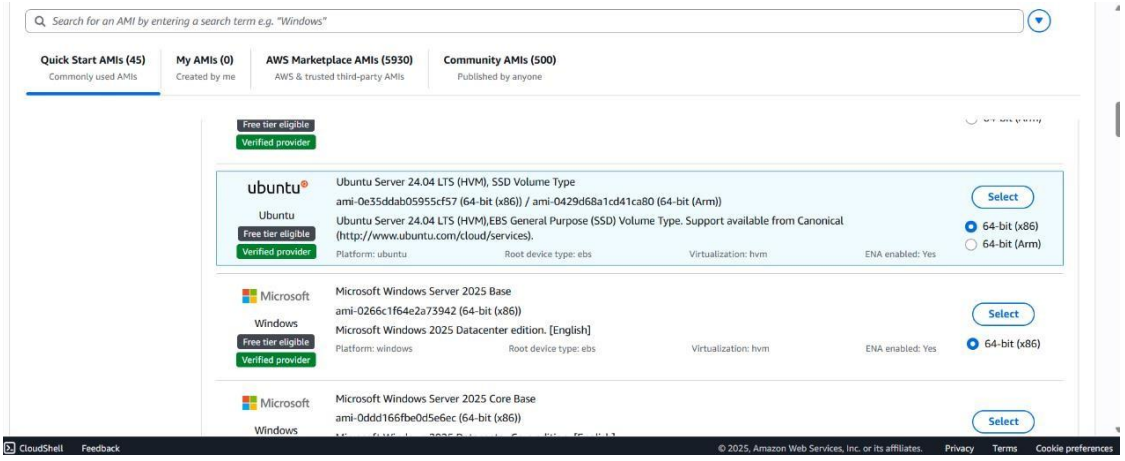
STEP26: GIVE A NAME TO AUTO-SCALING >CLICK ON CREATE A LAUNCH TEMPLATE



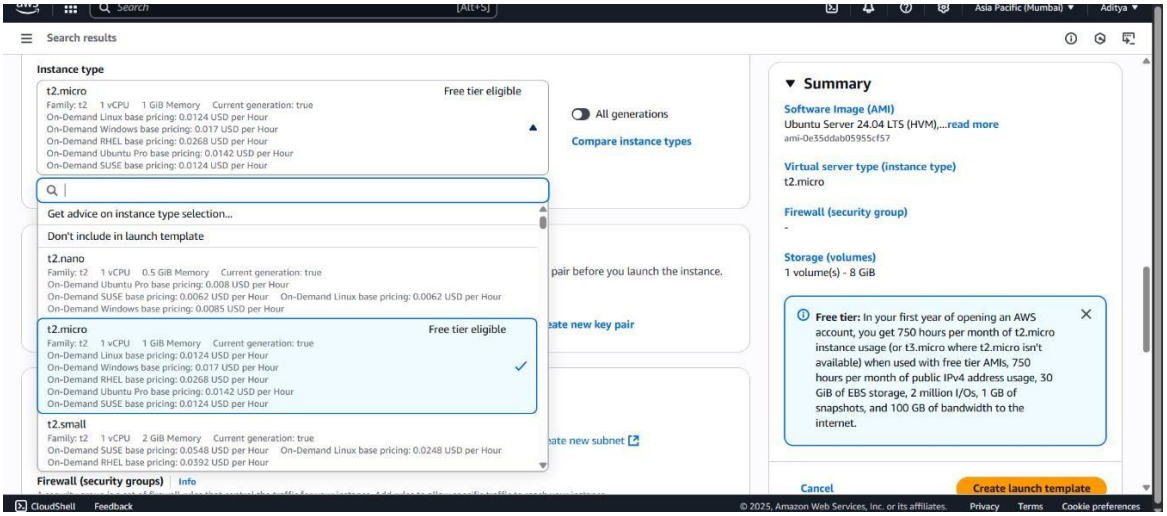
STEP27: GIVE A TEMPLATE NAME



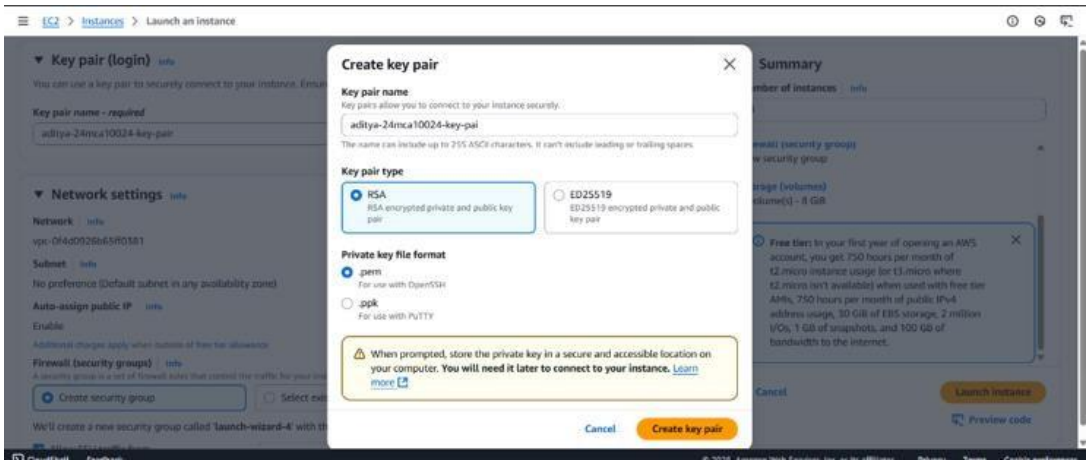
STEP28: SELECT UBUNTU



STEP29: SELECT INSTANCE TYPE

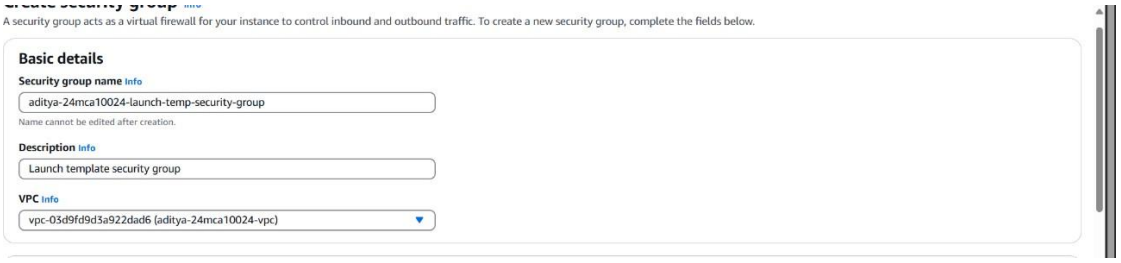


STEP30: CREATE A KEY PAIR



STEP31: NEXT WE HAVE TO CREATE A SECURITY GROUP TO BE ADDED IN THE LAUNCH TEMPLATE

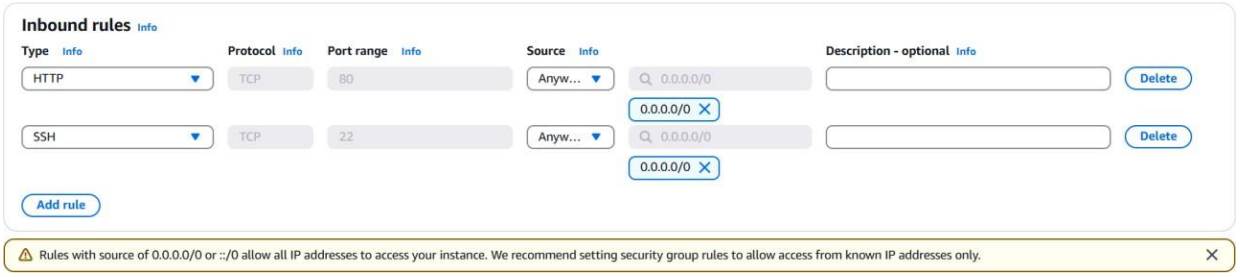
a. **SEARCH VPC>OPEN VPC IN A NEW TAB>CLICK ON SECURITY GROUP>CREATE SECURITY GROUP**
b. **WRITE A NAME & DESCRIPTION TO THE SECURITY GROUP> CHOOSE THE VPC THAT YOU HAVE CREATED**



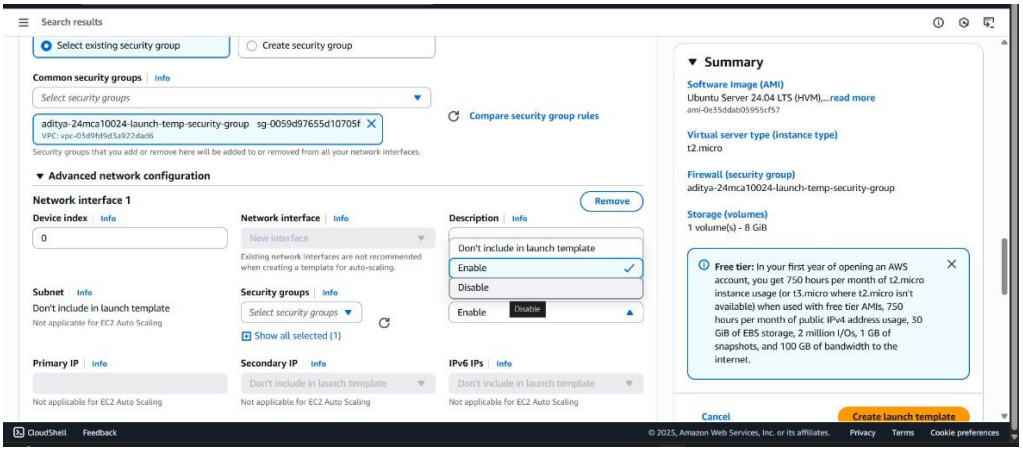
STEP32: SET IN BOUND RULE:

a. **CLICK ON ADD RULE> SELECT HTTP & SOURCE: ANYWHERE (0.0.0.0/0).**
b. **CLICK ON ADD RULE> SELECT SSH & SOURCE: ANYWHERE (0.0.0.0/0).**

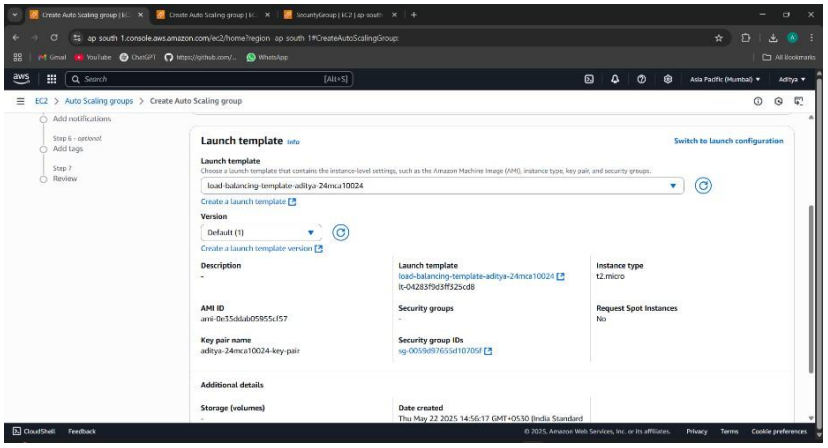
THEN CLICK ON ADD SECURITY GROUP



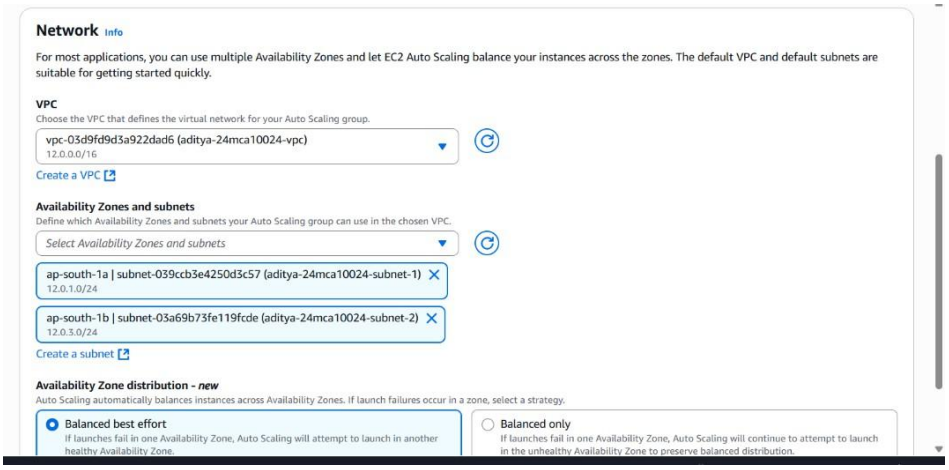
STEP33: NOW SELECT THE SECURITY GROUP THAT YOU HAVE CREATED IN THE LAUNCH TEMPLATE> AUTO ASSIGN PUBLIC IP: ENABLE> KEEP DEFAULT EBS VOLUME>CLICK ON CREATE LAUNCH TEMPLATE



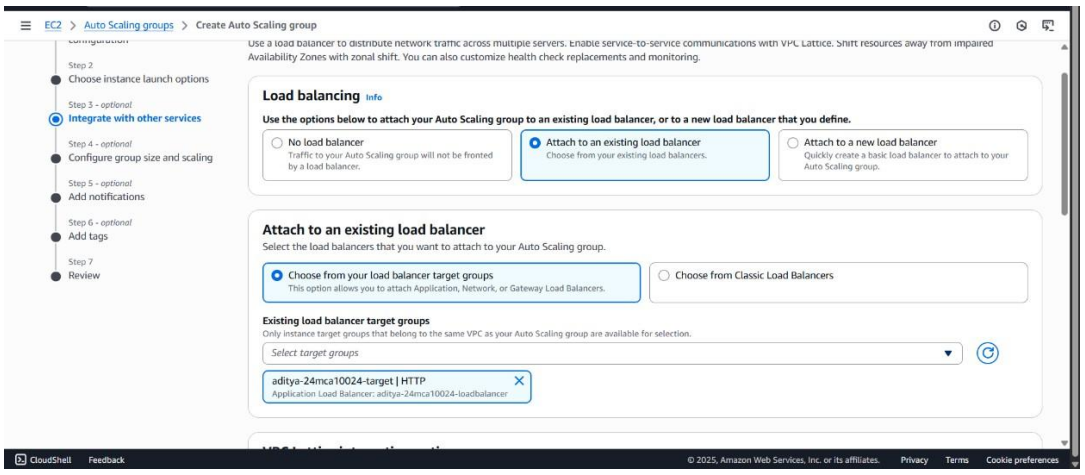
STEP34: NOW SELECT THE LAUNCH TEMPLATE FOR AUTO-SCALING GROUP> CLICK ON NEXT



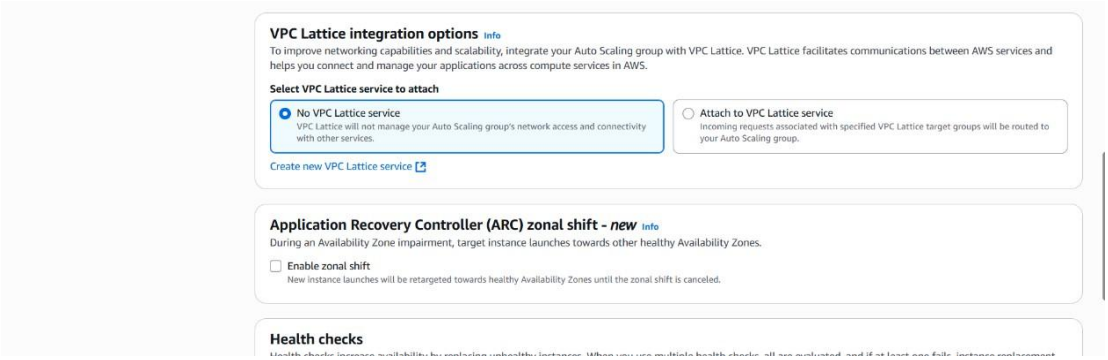
STEP35: SELECT VPC THAT YOU HAVE CREATED>SELECT BOTH THE AVAILABILITY ZONES>CLICK ON NEXT.



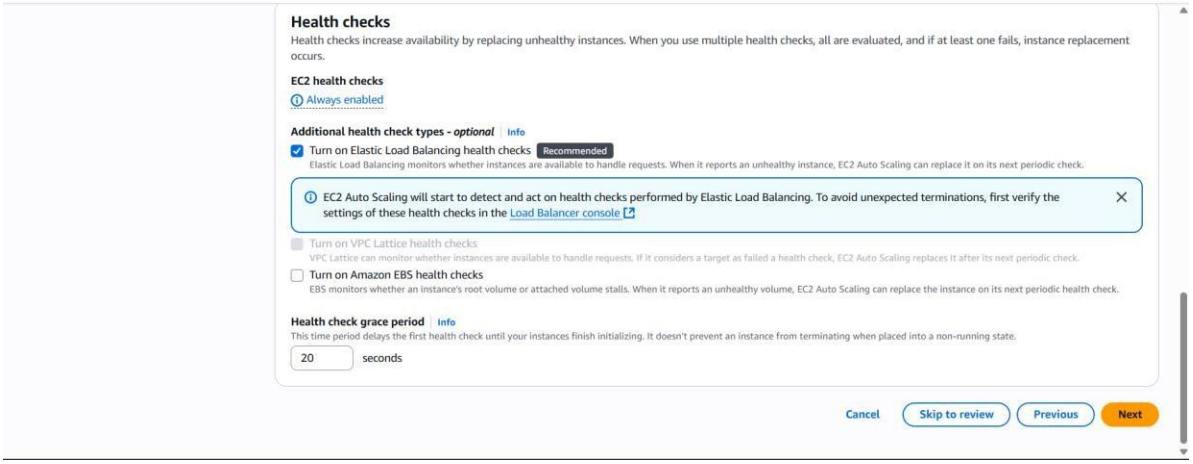
STEP36: SELECT ATTACH TO AN EXISTING LOAD BALANCER>CHOOSE FROM YOUR LOAD BALANCER TARGET GROUPS>CHOOSE THE TARGET GROUP THAT YOU HAVE CREATED



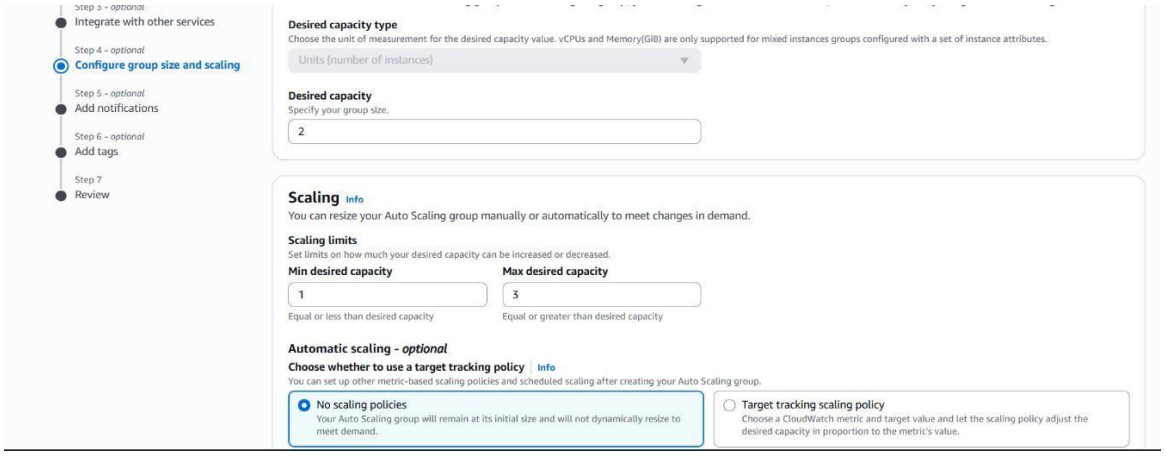
STEP37: SELECT NO VPC LATTICE SERVICE



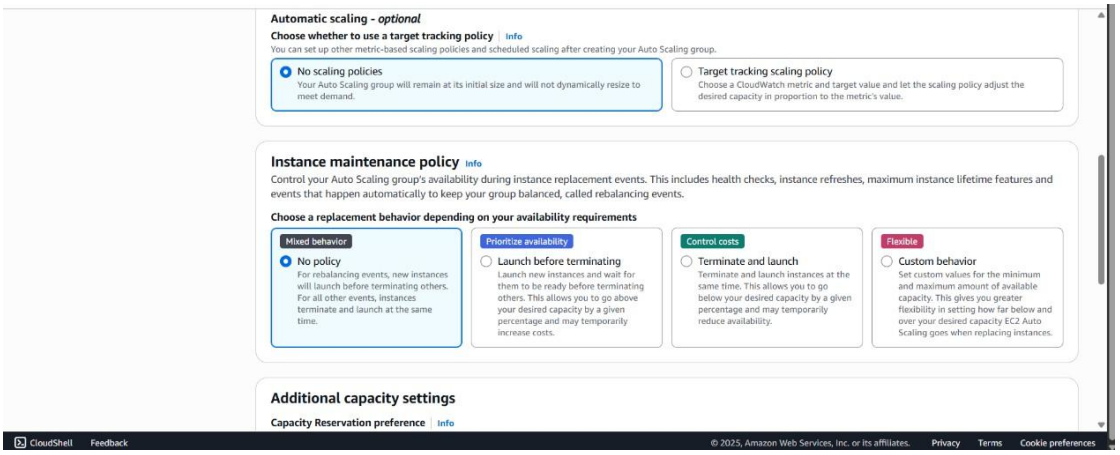
STEP38: SELECT TURN ON ELASTIC LOAD BALANCING>SET 20 SECONDS>CLICK ON NEXT



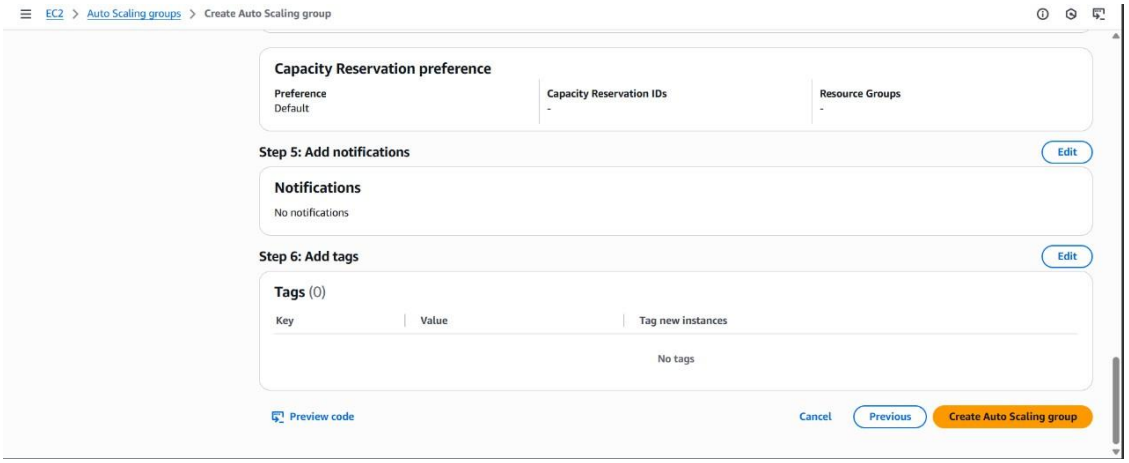
STEP39: SET DESIRED CAPACITY:2, MIN CAPACITY:1 & MAXIMUM CAPACITY:3



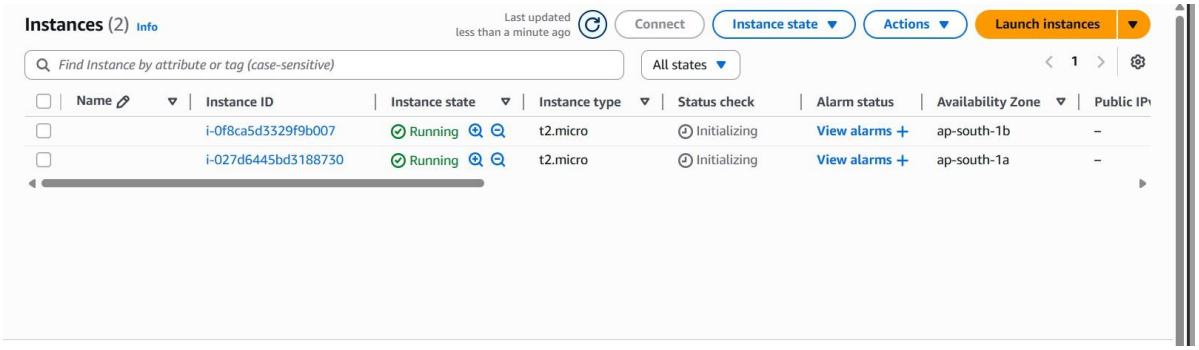
STEP40: CLICK ON NEXT



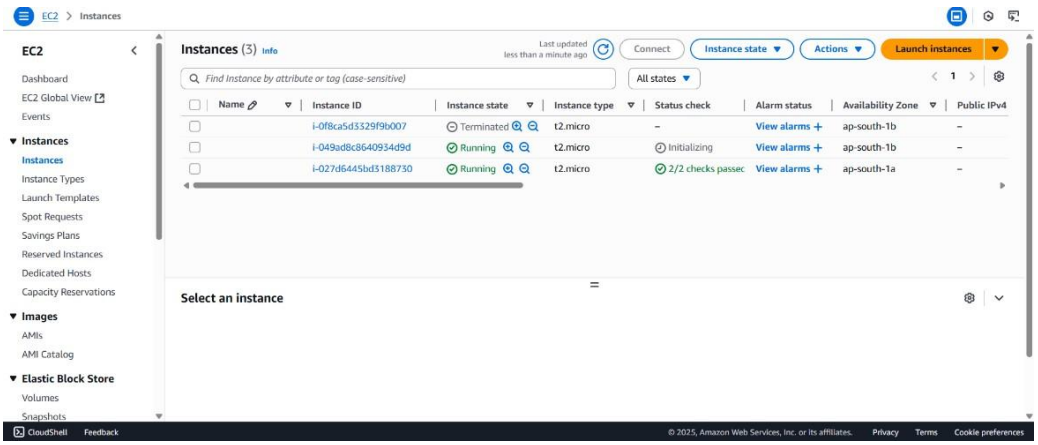
STEP41: CLICK ON AUTO-SCALING GROUP



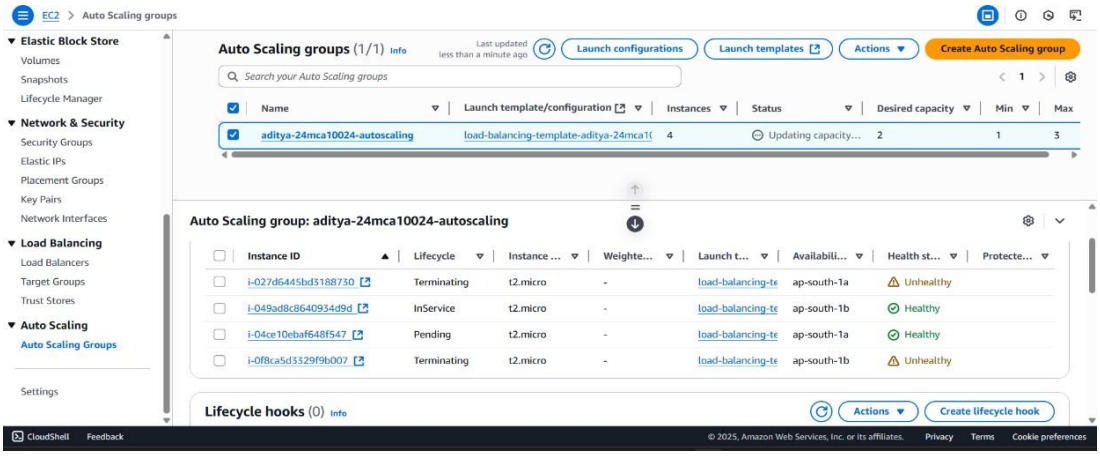
STEP42: NOW CLICK ON EC2>INSTANCES TO CHECK THE NUMBER OF INSTANCES CREATED.



STEP43: NOW TERMINATE ONE OF THE INSTANCES FROM EC2 INSTANCES



STEP44: NOW GO TO AUTO-SCALING GROUPS>INSTANCE MANAGEMENT>HEALTH STATES: UNHEALTHY MEANS DELETED INSTANCE (IT WILL AUTOMATICALLY CREATE ANOTHER INSTANCE AFTER DELETION)



NOTE:

1. **A *load balancer* serves as the single point of contact for clients. The load balancer distributes incoming application traffic across multiple targets, such as EC2 instances, in multiple Availability Zones. This increases the availability of your application. You add one or more listeners to your load balancer.**
2. **A *listener* checks for connection requests from clients, using the protocol and port that you configure. The rules that you define for a listener determine how the load balancer routes requests to its registered targets. Each rule consists of a priority, one or more actions, and one or more conditions. When the conditions for a rule are met, then its actions are performed. You must define a default rule for each listener, and you can optionally define additional rules. Before you start using your Application Load Balancer, you must add at least one listener. If your load balancer has no listeners, it can't receive traffic from clients. The rules that you define for your listeners determine how the load balancer routes requests to the targets that you register, such as EC2 instances.**
3. **Each *target group* routes requests to one or more registered targets, such as EC2 instances, using the protocol and port number that you specify. You can register a target with multiple target groups. You can configure health checks on a per target group basis. Health checks are**

performed on all targets registered to a target group that is specified in a listener rule for your load balancer.

4. **An *Auto Scaling group* contains a collection of EC2 instances that are treated as a logical grouping for the purposes of automatic scaling and management. An Auto Scaling group also lets you use Amazon EC2 Auto Scaling features such as health check replacements and scaling policies. Both maintaining the number of instances in an Auto Scaling group and automatic scaling are the core functionality of the Amazon EC2 Auto Scaling service. The size of an Auto Scaling group depends on the number of instances that you set as the desired capacity. You can adjust its size to meet demand, either manually or by using automatic scaling**

Date: 23/05/2025	Title
Exp. No: 07	How to create EBS and attach to EC2 instance, modify size and create a snapshot.

HOW TO CREATE EBS AND ATTACH TO EC2 INSTANCE, MODIFY SIZE AND CREATE A SNAPSHOT

Working with EBS on EC2:

Overview –

Amazon EBS and EC2: Creating, Attaching, Modifying, and Snapshots
Amazon Elastic Block Store (EBS) is a scalable, high-performance block storage service designed for use with Amazon EC2 instances. It enables users to create persistent volumes that remain available independently of the lifecycle of the EC2 instances they are attached to. EBS volumes behave like raw, unformatted block devices and can be formatted and mounted just like any physical disk.

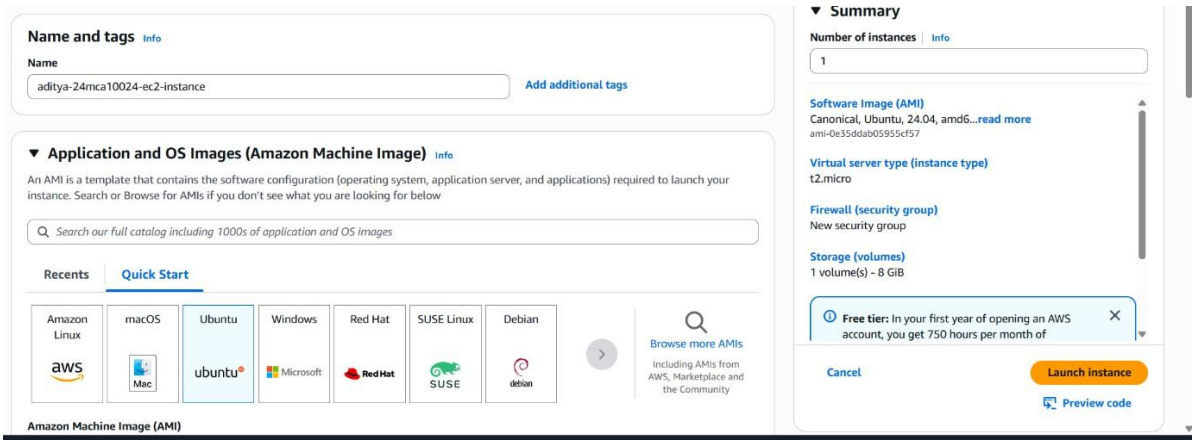
Creating an EBS Volume involves provisioning storage of a specified size and type (such as gp3, gp2, or io1) in a chosen availability zone. The volume must reside in the same availability zone as the EC2 instance it will be attached to, ensuring low-latency connectivity. Once a volume is created, it can be attached to an EC2 instance, allowing the instance to use it for data storage. The attached volume can be formatted with a file system (e.g., XFS, ext4) and mounted to a directory path, making it accessible for storing application data, logs, backups, or any other data.

EBS also supports modifying volumes on the fly, which is essential for growing storage capacity as application needs evolve. You can increase the size of a volume or change its performance characteristics without detaching it or shutting down the EC2 instance. After resizing, the operating system on the instance may need to expand the file system to utilize the new space.

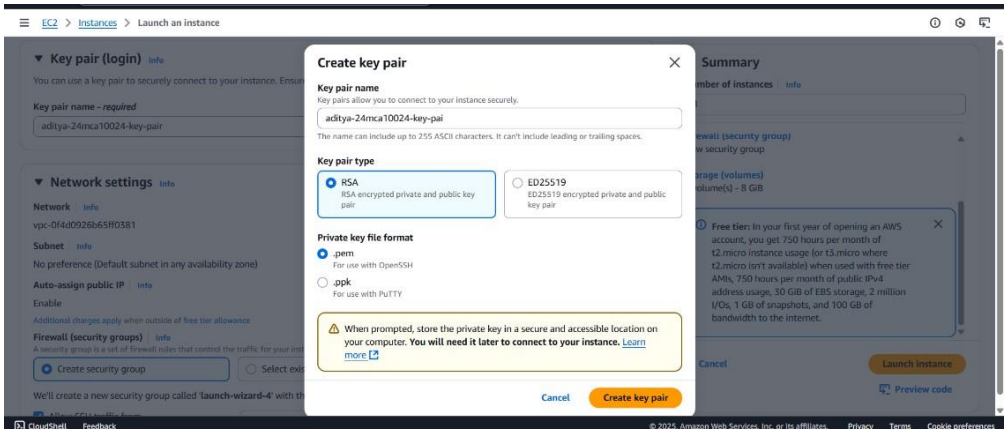
For data protection and disaster recovery, EBS snapshots offer a point-in-time backup mechanism. Snapshots are incremental and stored in Amazon S3, making them cost-efficient and highly durable. These snapshots can be used to restore volumes or create new ones across different availability zones or even regions, supporting scalability and resilience.
Overall, Amazon EBS provides a robust, flexible storage solution that integrates tightly with EC2, enabling high availability, easy scalability, and reliable data management for a wide range of cloud-based applications.

CREATION OF FIRST EC2 INSTANCE

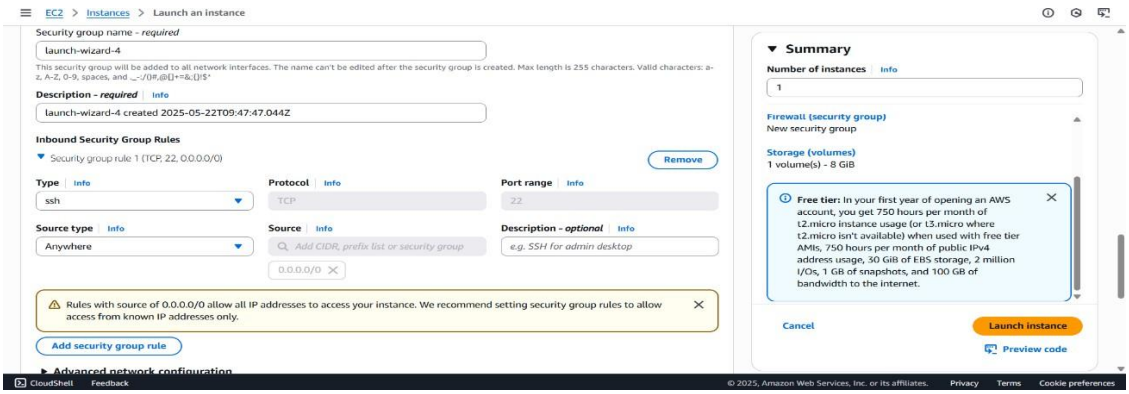
STEP1: GO TO EC2 DASHBOARD>CLICK ON INSTANCES>CLICK ON LAUNCH INSTANCE>TYPE THE NAME OF THE INSTANCE>CHOOSE THE UBUNTU



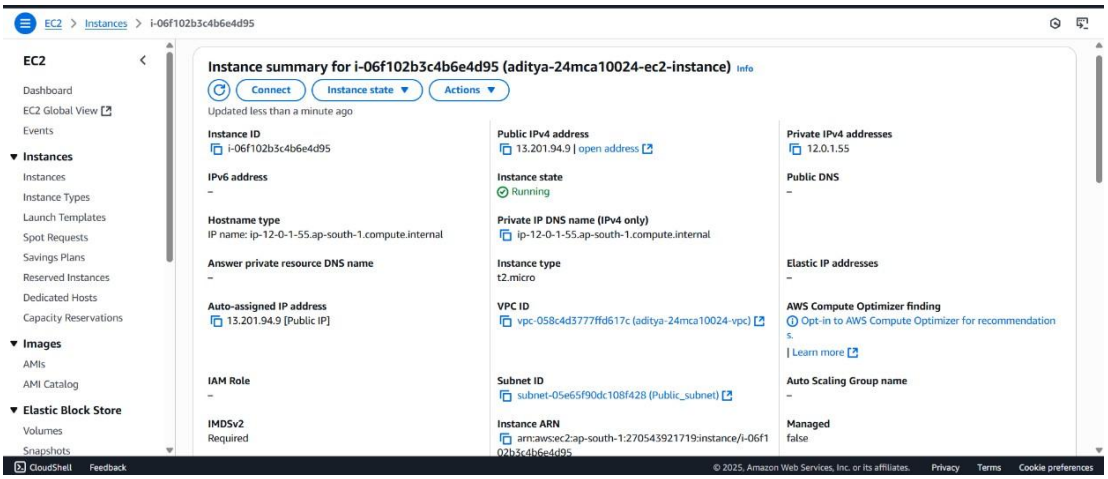
STEP2: CREATE A KEY PAIR



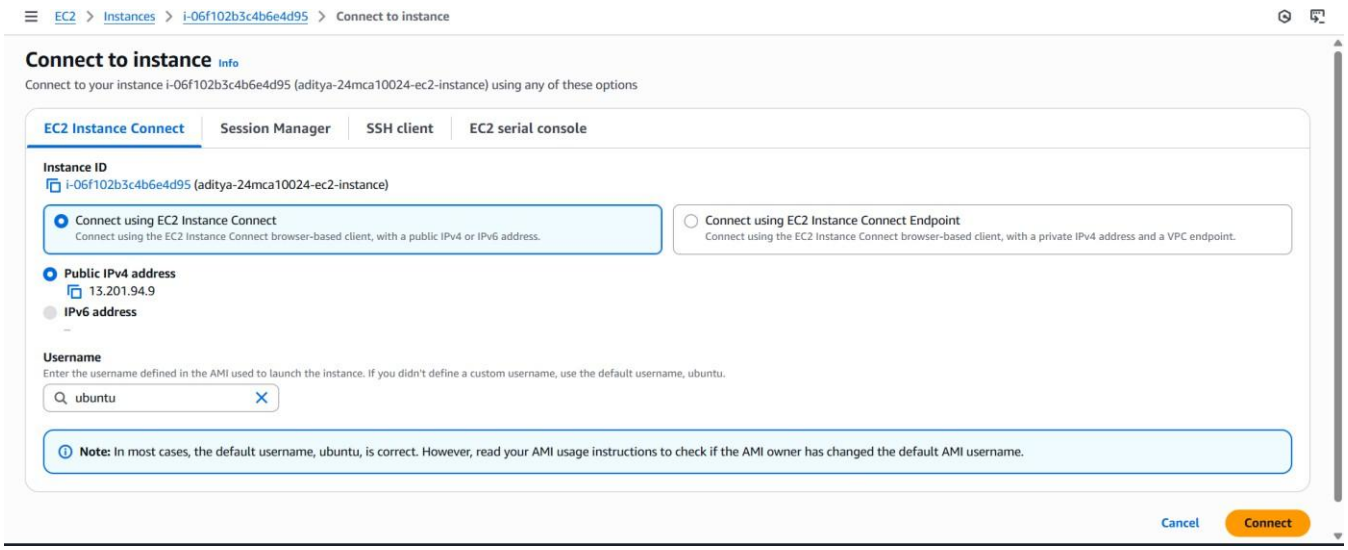
STEP3: EDIT THE NETWORK SETTING> SELECT THE VPC CREATED>CHOOSE THE PUBLIC SUBNET>ENABLE AUTO-ASSIGN PUBLIC IP>CREATE SECURITY GROUP
STEP4: SET TYPE: SSH IN THE SECURITY GROUP>CLICK ON LAUNCH INSTANCE



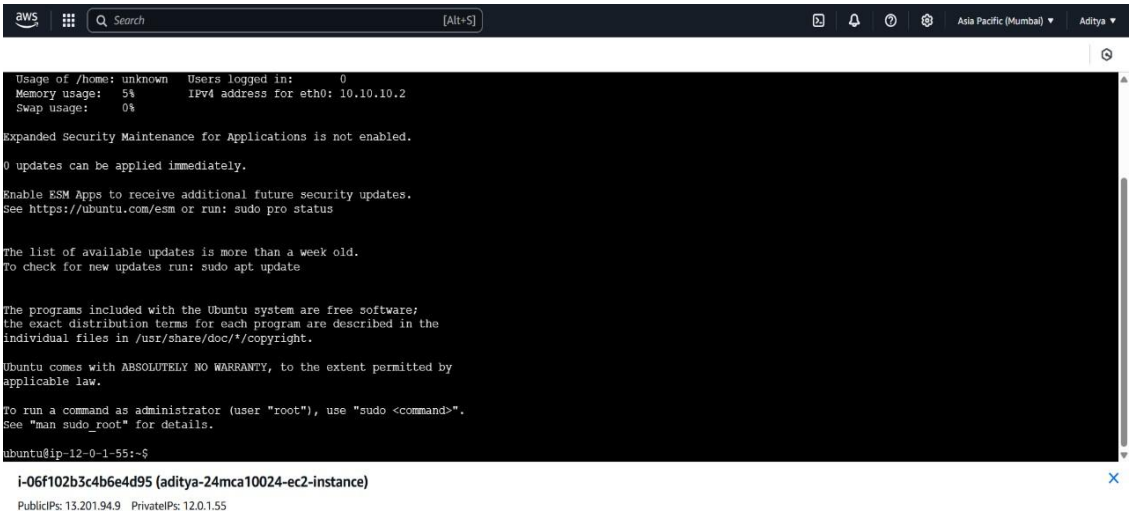
STEP5: CLICK ON THE EC2 INSTANCE CREATED>CLICK ON CONNECT



STEP6: CLICK ON EC2 INSTANCE CONNECT



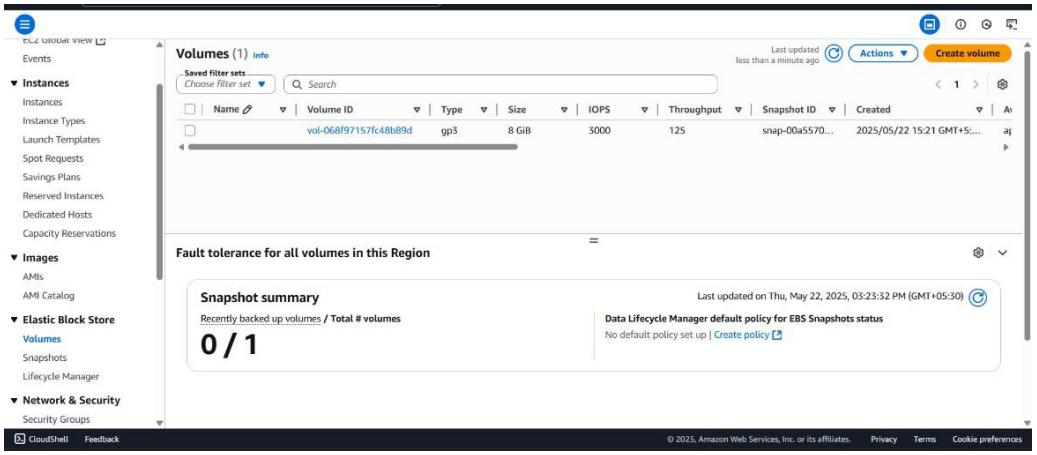
STEP7: EC2 INSTANCE IS CONNECTED



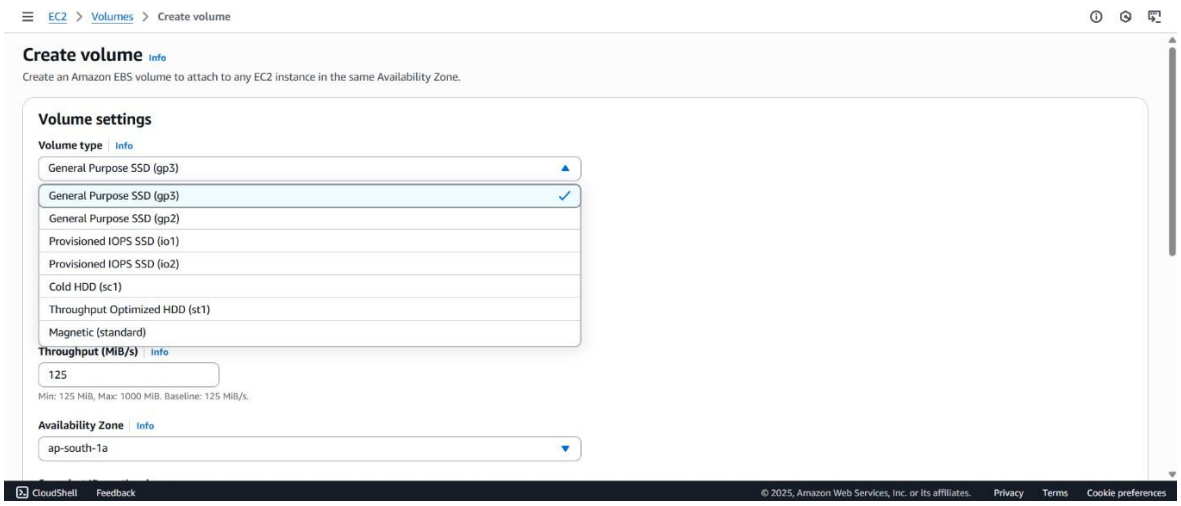
STEP 8: TYPE "lsblk" COMMAND IN THE TERMINAL. IT WILL SHOW YOU HOW MANY VOLUMES ARE ATTACHED TO THE EC2 INSTANCE. IT SHOWS THE DEFAULT 8G VOLUME WHICH WE SET WHILE CREATING EC2 INSTANCE.

CREATION OF EBS VOLUME

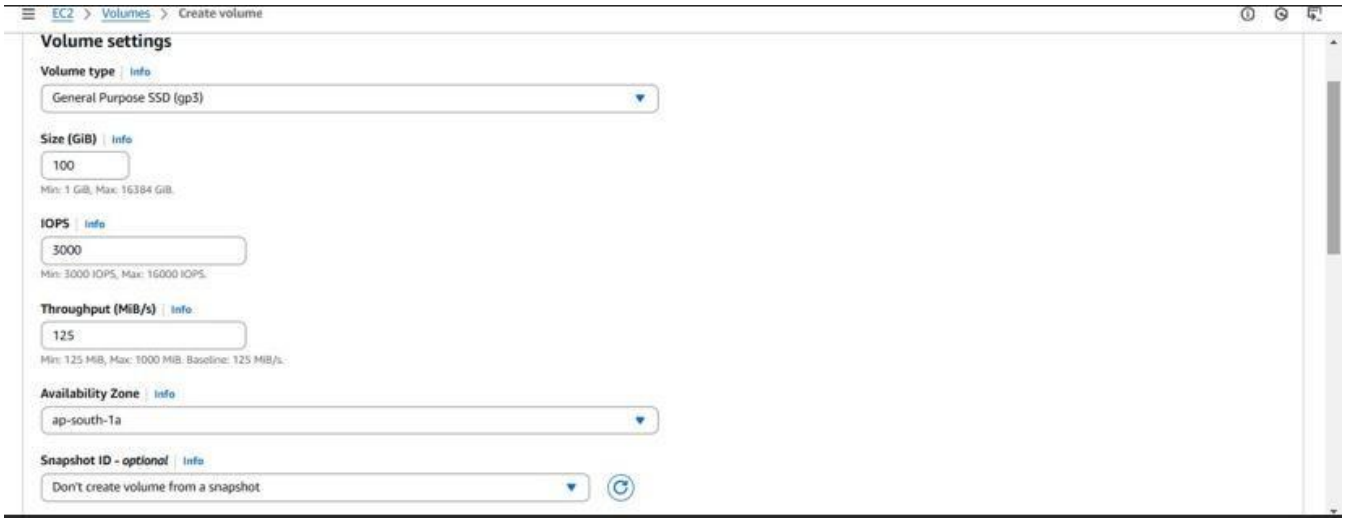
STEP 9: NOW CREATE EBS VOLUME FROM EC2 DASHBOARD: ELASTIC BLOCK STORE>VOLUMES>CREATE VOLUME



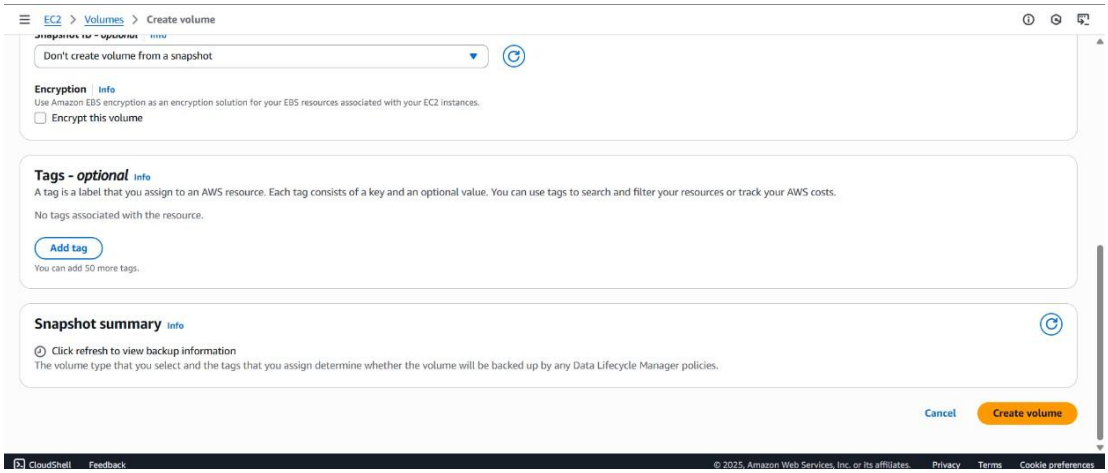
STEP 9: YOU CAN CHOOSE ANY OF THE VOLUME TYPES FROM THE DROP-DOWN >SELECT gp3



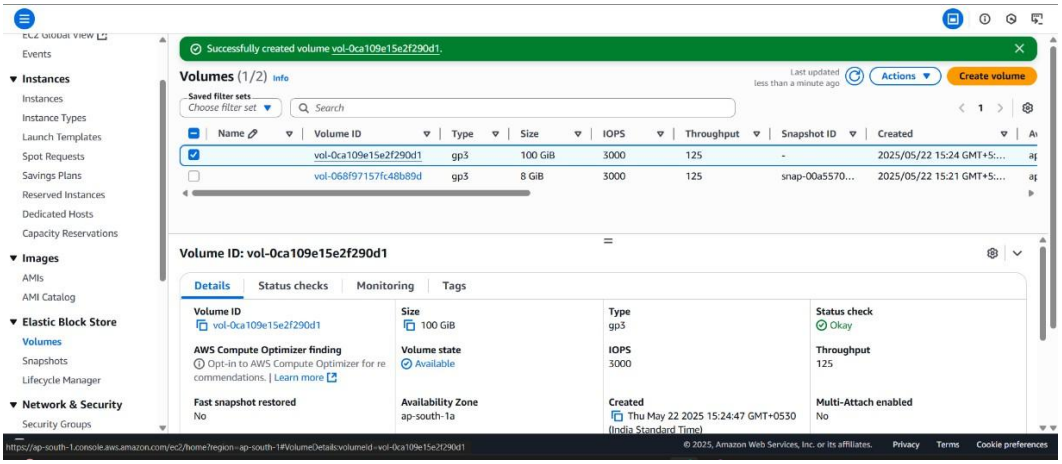
STEP 10: KEEP THE DATA AS SHOWN



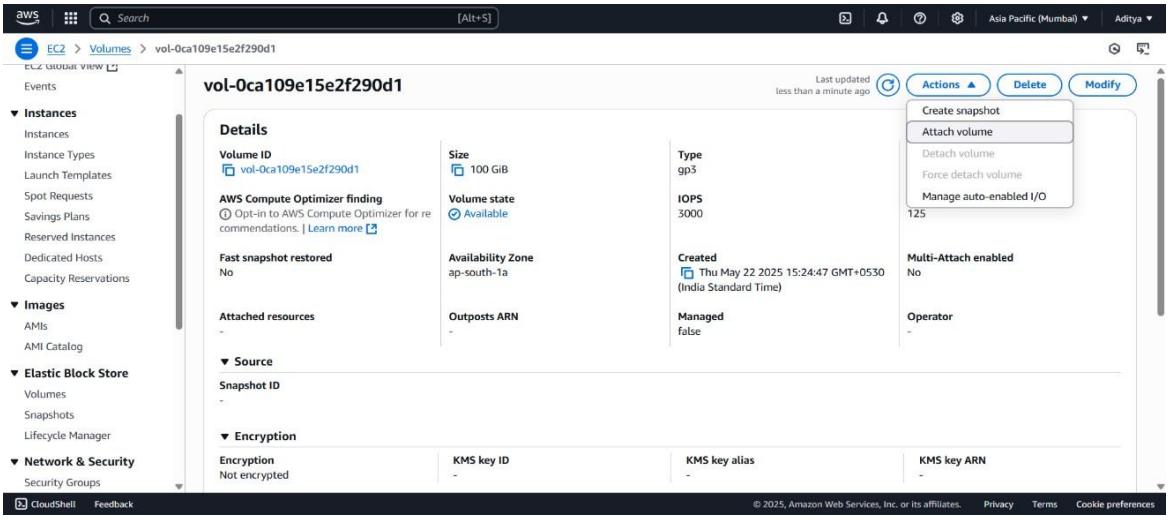
STEP 11: CLICK ON CREATE VOLUME.



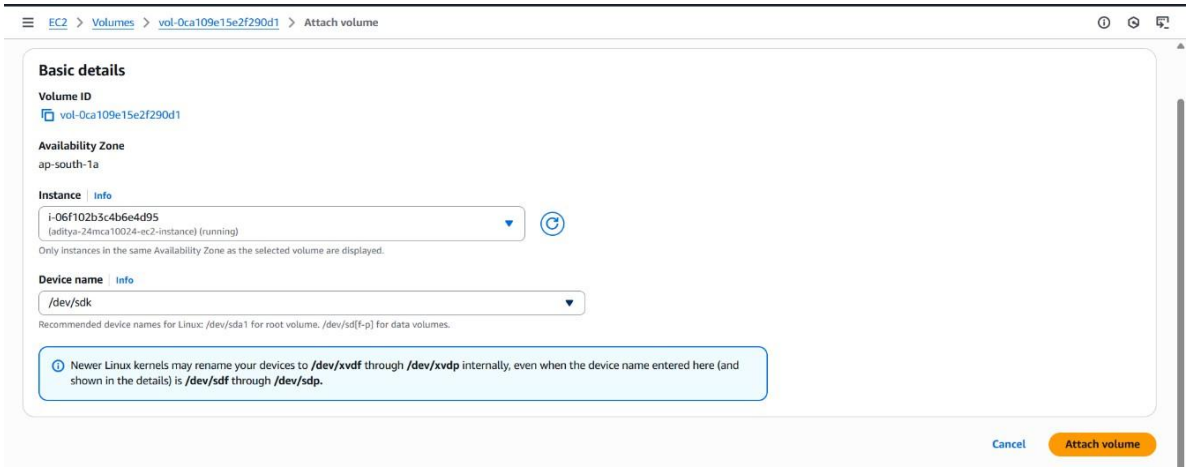
STEP 12: EDIT THE VOLUME NAME BY CLICKING ON IT>CLICK ON THE VOLUME THAT YOU HAVE CREATED



STEP 13: CLICK ON ACTIONS>ATTACH VOLUME

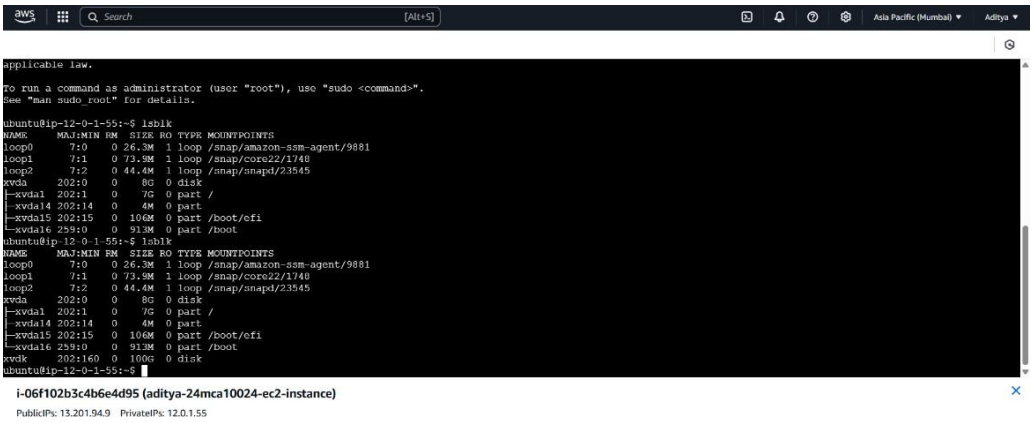


STEP 14: SELECT INSTANCE THAT YOU HAVE CREATED>CHOOSE THE DEVICE:/dev/sdk>CLICK ON ATTACH VOLUME



STEP 15: NOW GO TO EC2 TERMINAL AND TYPE:lsblk>IT WILL LIST ALL THE VOLUME(100G IS SHOWN BELOW)

STEP 16: TYPE sudo fdisk -l (IT WILL LIST OUT ALL THE DISK PARTITION)



STEP 17: TYPE sudo file -s /dev/xvdk (IT WILL SHOW "DATA" WHICH MEANS THERE IS NO FILE SYSTEM CREATED FOR THIS EBS VOLUME)

STEP 18: WE NEED TO CREATE FILE SYSTEM FOR EBS VOLUME: TYPE sudo mkfs -t xfs /dev/xvdk (FILE SYSTEM CREATED)

STEP 19: NOW AGAIN TYPE sudo file -s /dev/xvdk TO CHECK THE FILE SYSTEM (ITS SHIWS EBS VOLUME HAS xfs FILESYSTEM)

STEP 20: CREATE A DIRECTORY:

a. **TYPE sudo mkdir /munsifaebsvol**
b. **NEXT TYPE sudo mount /dev/xvdk / munsifaebsvol**
c. **TYPE ls -lart /munsifaebsvol**

```
ubuntu@ip-12-0-1-55:~$ sudo mkdir /aditya24mca10024
ubuntu@ip-12-0-1-55:~$ sudo mount /dev/xvdk / aditya24mca10024
mount: bad usage
Try 'mount --help' for more information.
ubuntu@ip-12-0-1-55:~$ ls -lart /aditya24mca10024
total 8
drwxr-xr-x 23 root root 4096 May 22 10:04 ..
drwxr-xr-x  2 root root 4096 May 22 10:04 .
ubuntu@ip-12-0-1-55:~$
```

i-06f102b3c4b6e4d95 (aditya-24mca10024-ec2-instance)

PublicIPs: 13.201.94.9 PrivateIPs: 12.0.1.55

STEP 21: NOW TYPE df -h (IT WILL SHOW THAT EBS VOLUME HAS BEEN MOUNTED TO DIRECTORY "munsifaebsvol")

```
ubuntu@ip-12-0-1-55:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/root       6.8G  1.7G  5.1G  26% /
tmpfs           479M     0  479M   0% /dev/shm
tmpfs           192M  872K  191M   1% /run
tmpfs           5.0M     0  5.0M   0% /run/lock
/dev/xvda16     881M   79M  741M  10% /boot
/dev/xvda15     105M  6.1M   99M   6% /boot/efi
tmpfs            96M   12K   96M   1% /run/user/1000
ubuntu@ip-12-0-1-55:~$
```

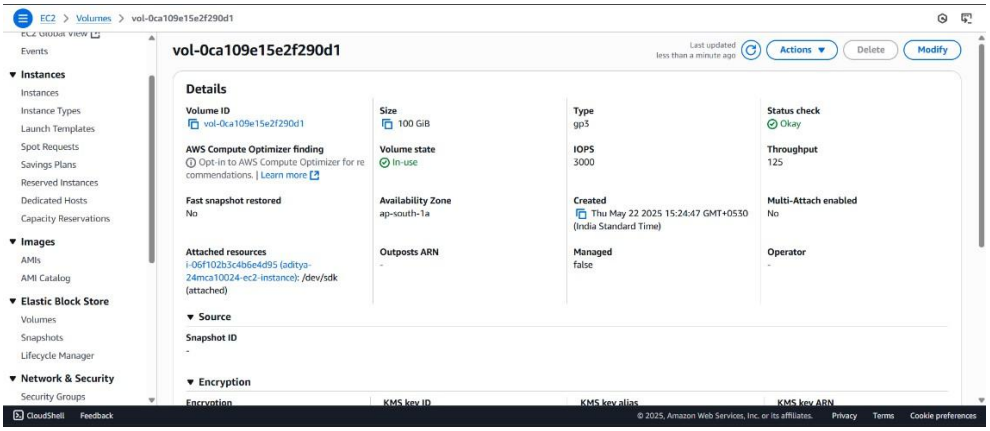
i-06f102b3c4b6e4d95 (aditya-24mca10024-ec2-instance)

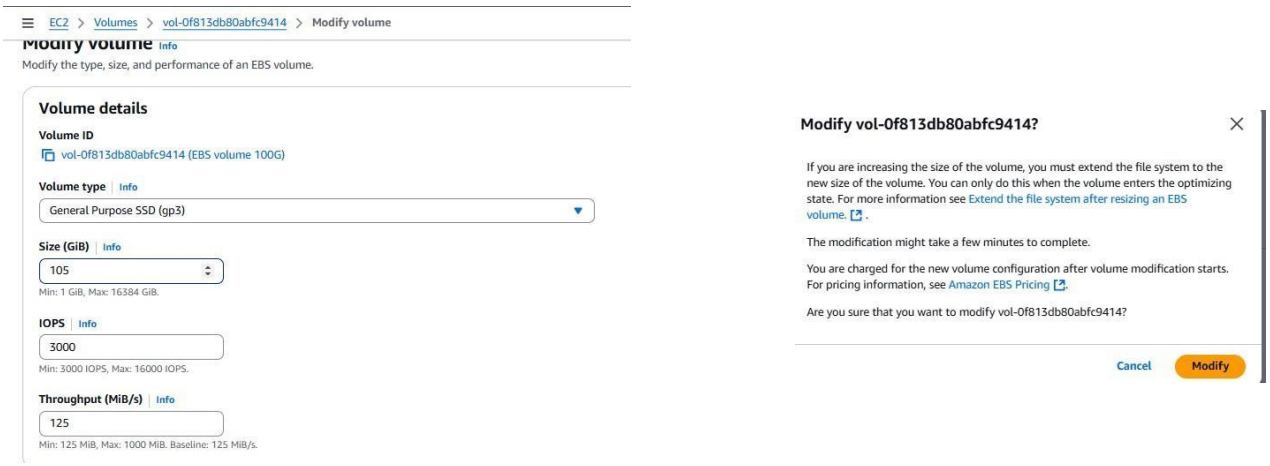
PublicIPs: 13.201.94.9 PrivateIPs: 12.0.1.55

MODIFY VOLUME SIZE

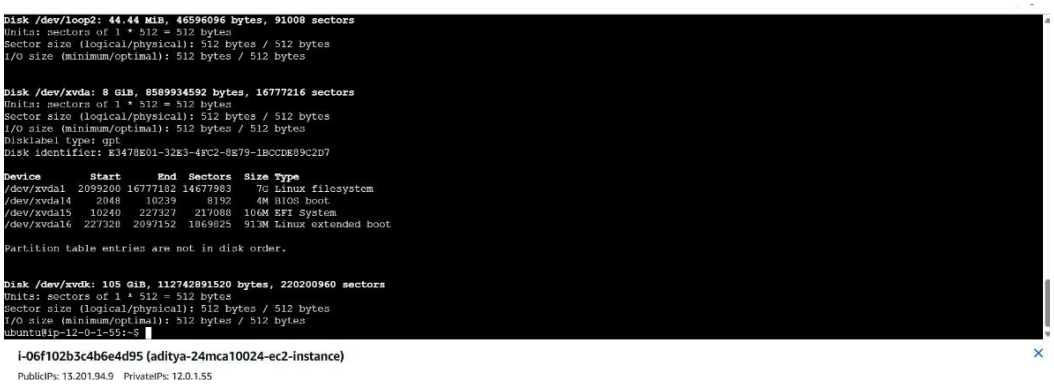
STEP 22: NOW NEXT GO TO EBS VOLUME TO INCREASE THE VOLUME(NOTE THAT EBS VOLUME CAN BE INCREASED BUT YOU CANNOT REDUCE THE VOLUME SIZE ONCE AN EBS VOLUME IS CREATED BECAUSE THERE IS A PROBABILITY OF DATA LOSS)

a. **CLICK ON EBS VOLUME>SELECT THE VOLUME THAT YOU HAVE CREATED>CLICK ON MODIFY**
b. **INCREASE THE VOLUME SIZE FROM 100 TO 105>CLICK MODIFY**





STEP 23: NOW GO TO THE EC2 TERMINAL AND VERIFY WHETHER THE DISK SIZE HAS CHANGED OR NOT. (TYPE sudo fdisk -l)



STEP 24: NOW IN THE EC2 TERMINAL GO TO EBS VOLUME DIRECTORY AND CREATE FILES.

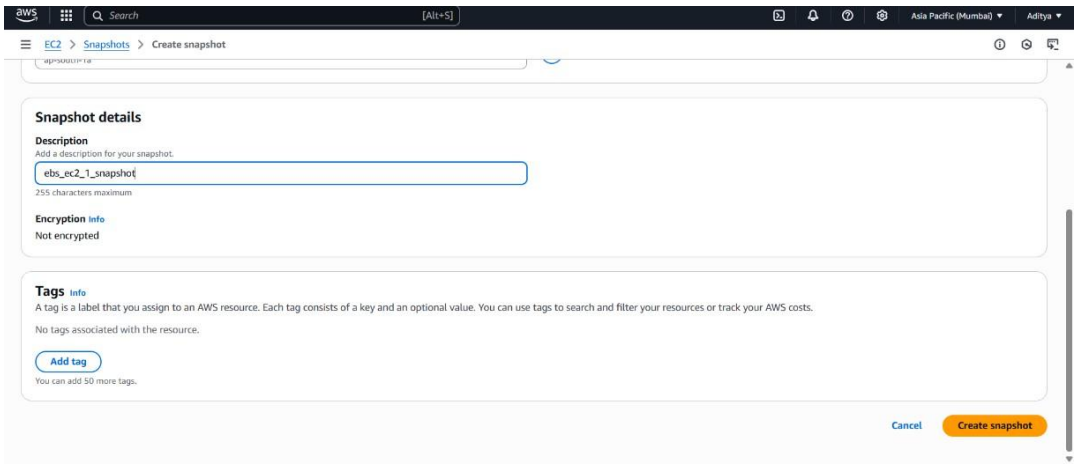
a. **TYPE cd /munsifaebsvol (GO TO EBS VOLUME DIRECTORY)**
b. **TYPE sudo touch 1.txt (CREATE FILE1)**
c. **TYPE sudo touch 2.txt (CREATE FILE2)**
d. **TYPE ls -lart (TO VERIFY WHETHER FILES ARE THERE OR NOT)**

EC2 SNAPSHOT CREATION

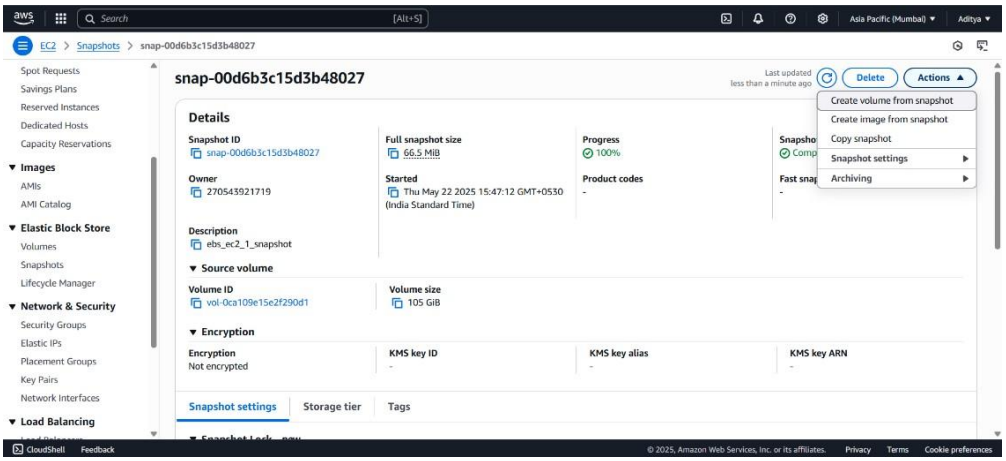
STEP 24: CREATE SECOND EC2 INSTANCE (CREATE IN A SIMILAR MANNER LIKE YOU HAVE CREATED FOR EC1)>CONNECT EC2 INSTANCE. THEN CHECK THE DISK SIZE OF EC2 INSTANCE (TYPE sudo fdisk -l) IT SHOWS THE DEFAULT 8G

STEP 25: NOW TO CREATE A SNAPSHOT GO TO EC2 DASHBOARD:

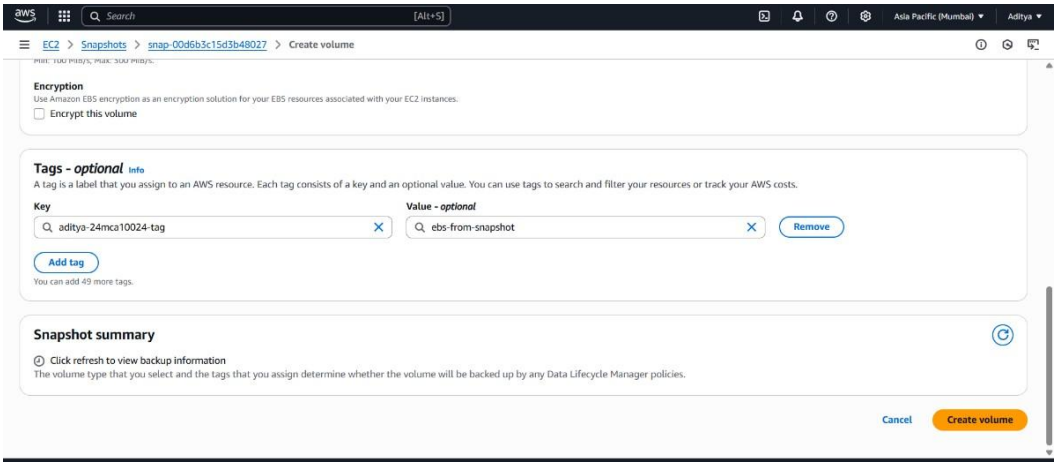
a. **EBS VOLUME>CLICK ON THE VOLUME THAT YOU HAVE CREATED>CLICK ON ACTIONS>CLICK ON CREATE SNAPSHOT**
b. **TYPE THE DESCRIPTION>CLICK ON CREATE A SNAPSHOT**



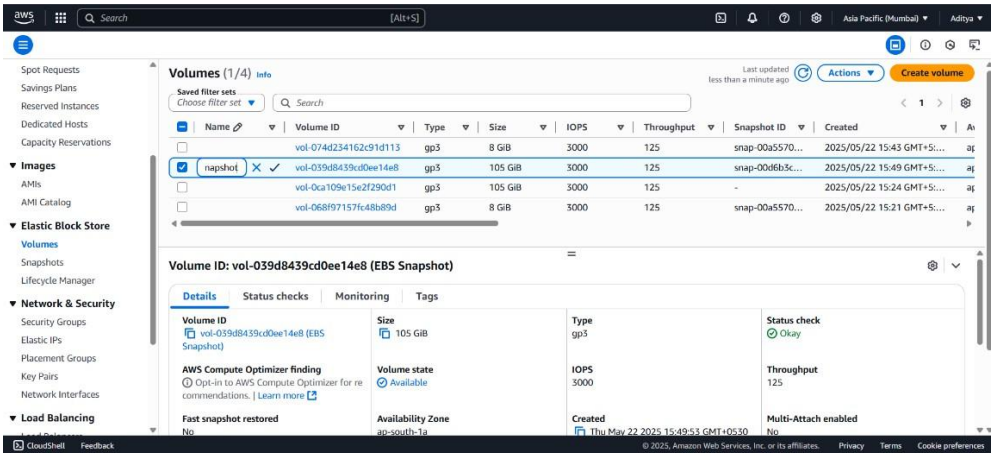
STEP 26: NOW GO TO SNAPSHOT>CLICK ON THE SNAPSHOT THAT YOU HAVE CREATED>CLICK ON ACTIONS>CREATE VOLUME FROM SNAPSHOT



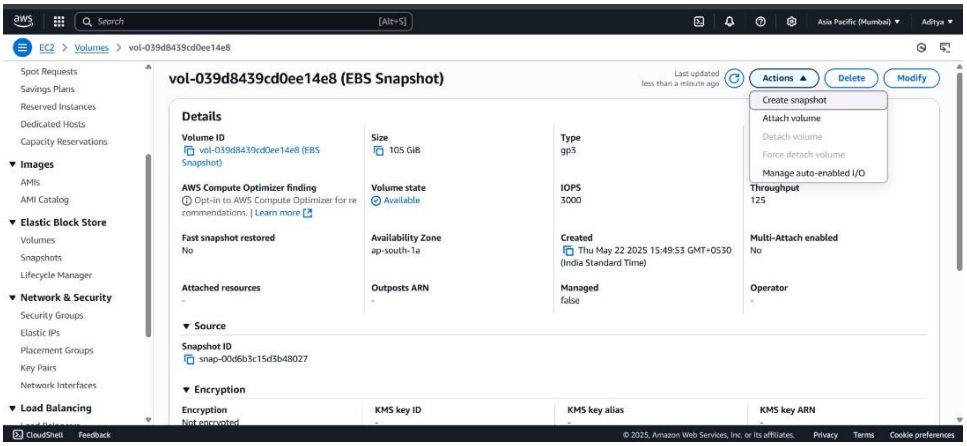
STEP 27: KEEP THE DATA SAME>CLICK ON TAG>GIVE A NAME>CLICK ON CREATE SNAPSHOT.



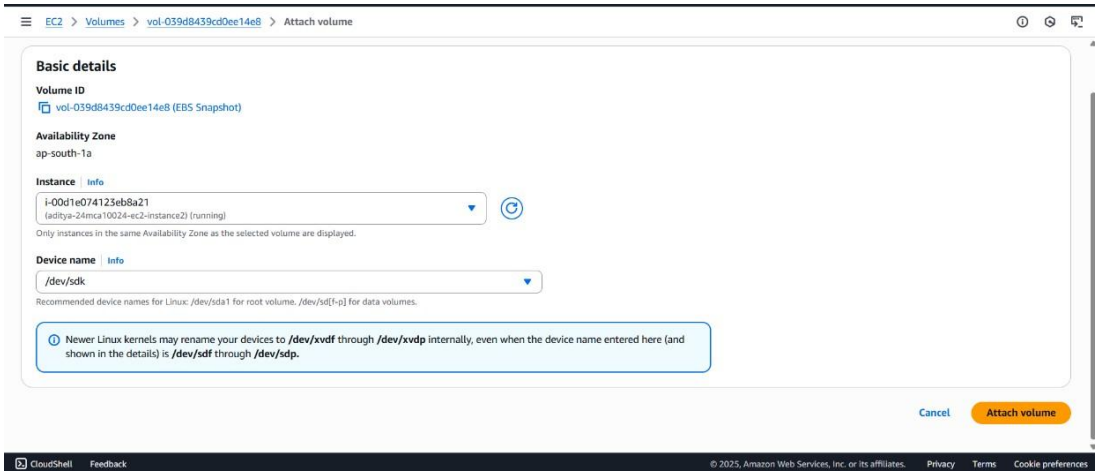
STEP 28: NOW GO TO VOLUME>MODIFY THE VOLUME NAME



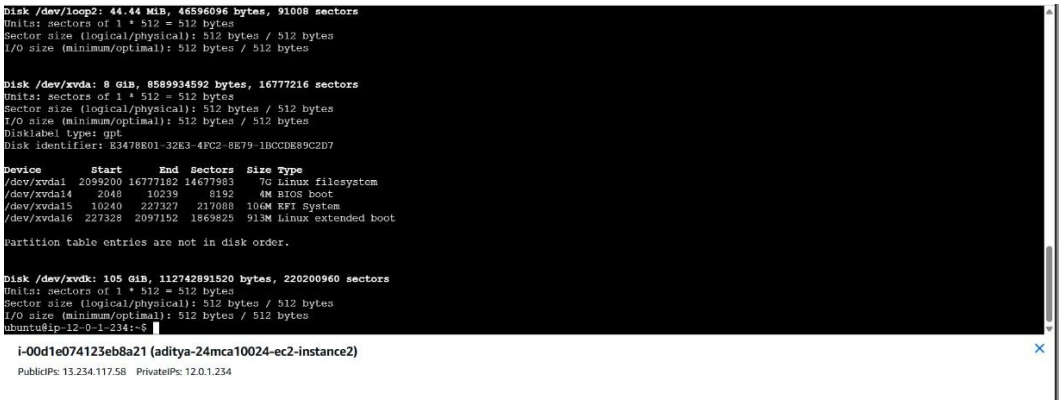
STEP 29: CLICK ON THE VOLUME>CLICK ON ACTION>CLICK ON ATTACH VOLUME



STEP 30: SELECT EC2 INSTANCE>SELECT sdk IN DEVICE>CLICK ON ATTACH VOLUME



STEP 31: GO TO EC2 TERMINAL>TYPE sudo fdisk -l TO CHECK WHETHER THE NEW VOLUME IS ADDED OR NOT. 105G IS ADDED AS SHOWN BELOW



STEP 32: NOW CHECK WHETHER FILE SYSTEM IS ATTACHED OR NOT. TYPE sudo file -s /dev/xvdk

STEP 33: NOW CREATE A DIRECTORY:

a. **TYPE sudo mkdir /munsifaebsvol2**
b. **MOUNT THE FILE, TYPE sudo mount /dev/xvdk /munsifaebsvol2**
c. **TYPE ls -lart /munsifaebsvol2**

STEP 34: NOW YOU CAN CHECK IN BOTH EC2 INSTANCES, FILE1 AND FILE2 IS PRESENT

a. **TYPE ls (EC2 1)**
b. **TYPE cd \munsifaebsvol2 > ls (EC2 2)**

STEP 35: IF YOU CREATE ANOTHER FILE IN FIRST EC2 INSTANCE, IT WILL NOT REFLECT IN THE SECOND EC2 INSTANCE.

a. **TYPE sudo touch 3.txt in FIRST EC2 INSTANCE**
b. **CHECK WITH ls (FIRST EC2 INSTANCE-IT WILL DISPLAY ALL THE THREE TEXT TILES.)**
c. **TYPE ls in SECOND EC2 INSTANCE (IT WILL ONLY DISPLAY TWO TEXT FILES)**