Centralized Cloud IT Infrastructure Setup for a Growing SME

Name: Harini.P

Register Number: 727823TUCY014

1.Problem Statement

A small enterprise with 25 employees is facing frequent system downtime, unstructured data storage, and lack of remote access. All applications and files currently run on individual employee systems with no centralized control. This leads to poor manageability, higher risk of data loss, and difficulty in scaling IT operations.

The IT head wants to move to a cloud-based infrastructure that is stable, manageable, and scalable without increasing operational complexity. However, the organization has no prior cloud experience, and a junior intern is assigned to design and implement a basic but production-ready cloud infrastructure.

2.Project Objectives

- To design and deploy a centralized cloud-based IT infrastructure
- To replace decentralized employee systems with a central server
- To provide secure remote administrative access
- To allocate CPU, memory, and storage appropriately
- To implement backup and recovery using snapshots
- To simulate real-world IT administration tasks
- To document infrastructure decisions clearly

3.Scope of the Project

- Cloud platform: AWS Free Tier
- Single centralized virtual machine
- Manual configuration only (no automation tools)
- Moderate workload (10–15 concurrent users)
- Focus on stability and reliability

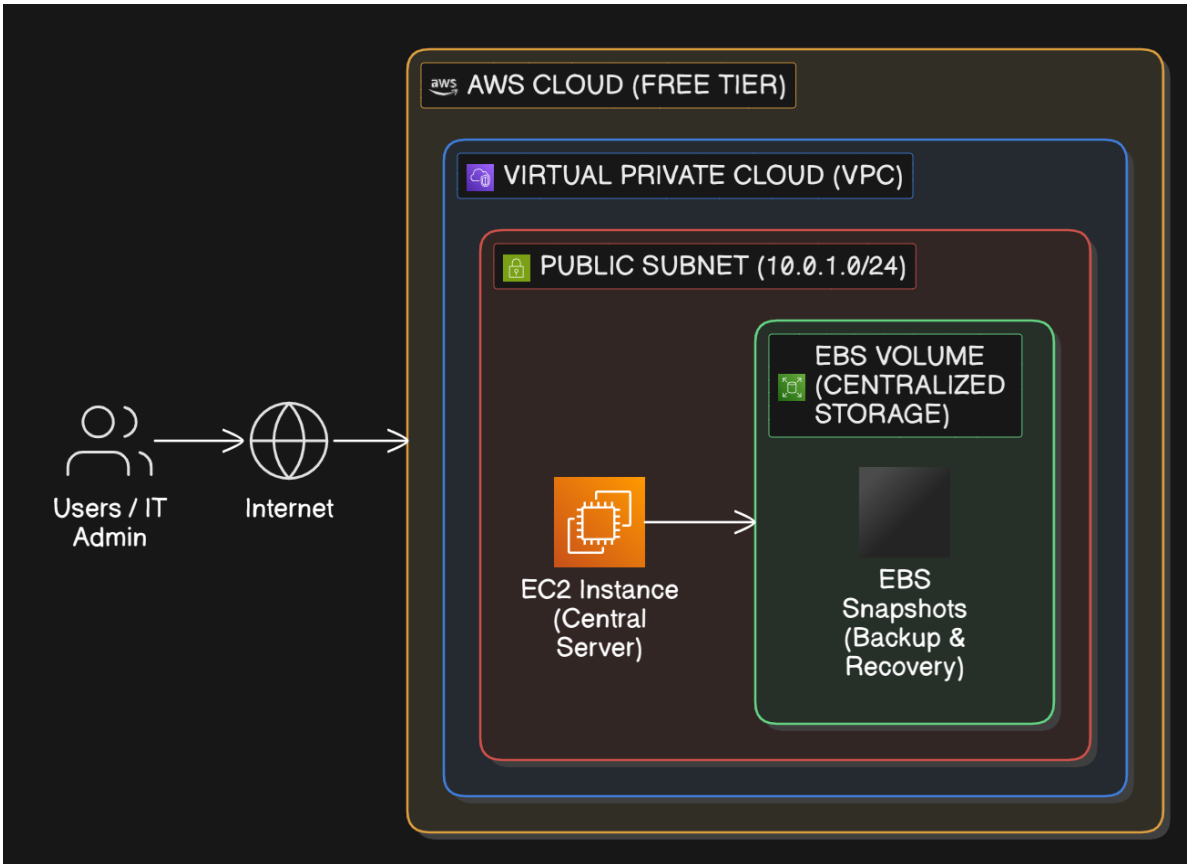
4.Architecture Overview

The proposed architecture provides a simple, secure, and cost-effective centralized IT infrastructure using AWS Free Tier services. A dedicated VPC ensures network isolation, while a public subnet and internet gateway allow secure remote access to the EC2 instance.

The EC2 instance acts as a centralized server for all users, and EBS volumes provide persistent storage. EBS snapshots are used for backup and recovery, ensuring data protection and business continuity. Security Groups and IAM users improve access control and overall security.

This architecture is well-suited for a small enterprise with moderate workload and limited cloud experience. While it lacks advanced features like auto scaling and monitoring, it meets current requirements and provides a strong foundation for future enhancements.

5.Block Diagram of the System



The above block diagram represents how end users and administrators securely access a centralized cloud server hosted on AWS. The EC2 instance acts as the

main server, while EBS provides persistent storage and snapshots ensure backup and recovery.

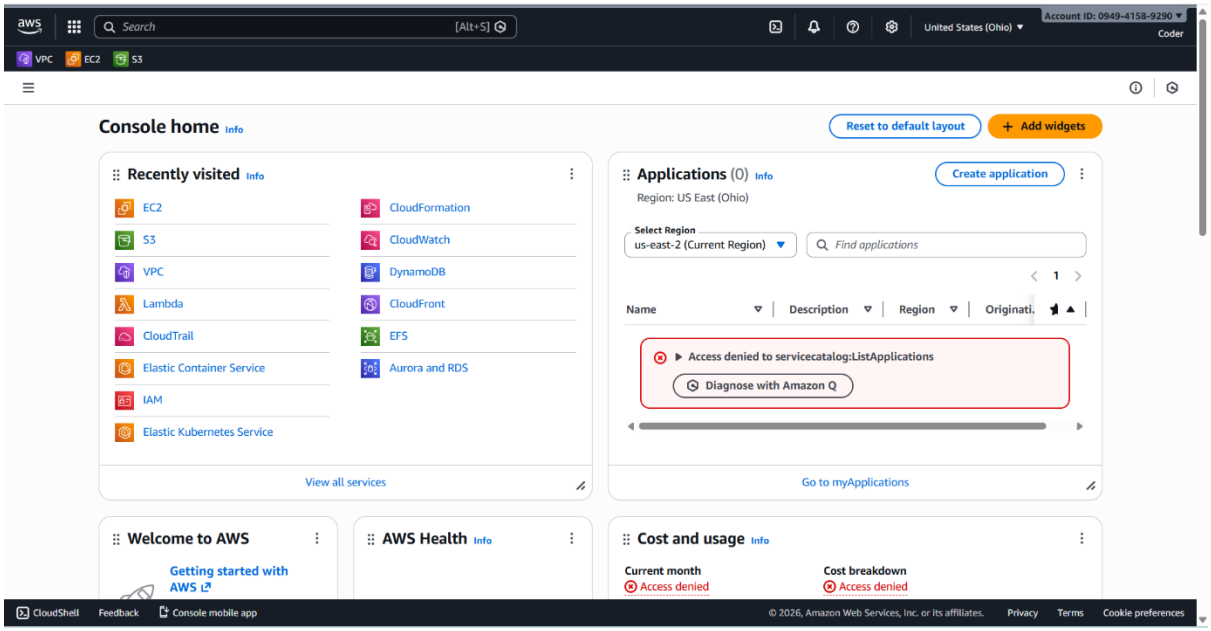
6.Step-by-Step Implementation Guide

Step 1: AWS Account Setup

Creating an AWS account is the foundation of this project. The Free Tier allows limited but sufficient resources to complete this setup without cost.

- Sign up for an AWS account using an email ID
- Activate Free Tier services
- Use the root account only for billing and initial setup
- Enable basic security settings

This step ensures a secure and cost-controlled cloud environment.

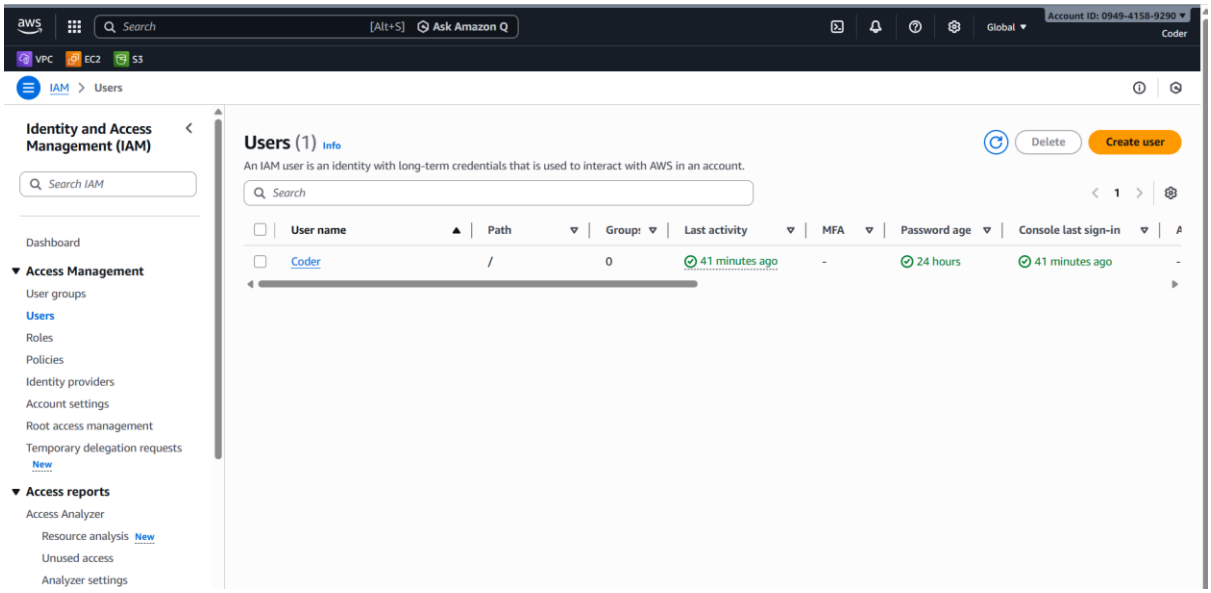


Step 2: IAM User Creation

IAM (Identity and Access Management) is used to manage users and permissions securely.

- Navigate to IAM → Users → Create user
- Enter username: Coder
- Enable AWS Management Console access
- Assign a custom password
- Attach the policy AdministratorAccess

This avoids using the root account for daily operations and follows AWS security best practices.

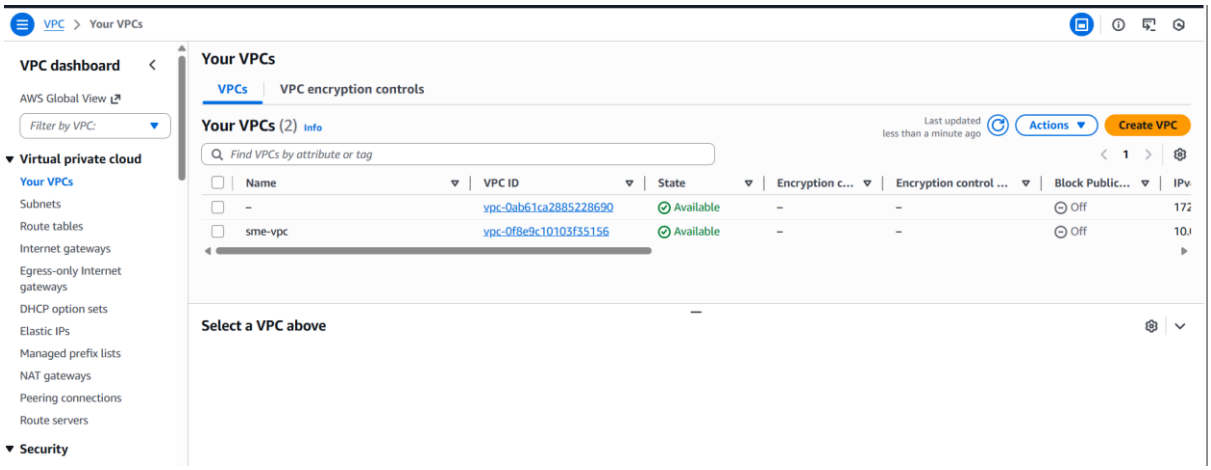


Step 3: Create Virtual Private Cloud (VPC)

A VPC is an isolated virtual network inside AWS that provides control over IP addressing and networking.

- Go to VPC → Create VPC
- Name: SME-VPC
- IPv4 CIDR block: 10.0.0.0/16
- Enable DNS resolution and DNS hostnames

This VPC acts as the private network for hosting the central server securely.

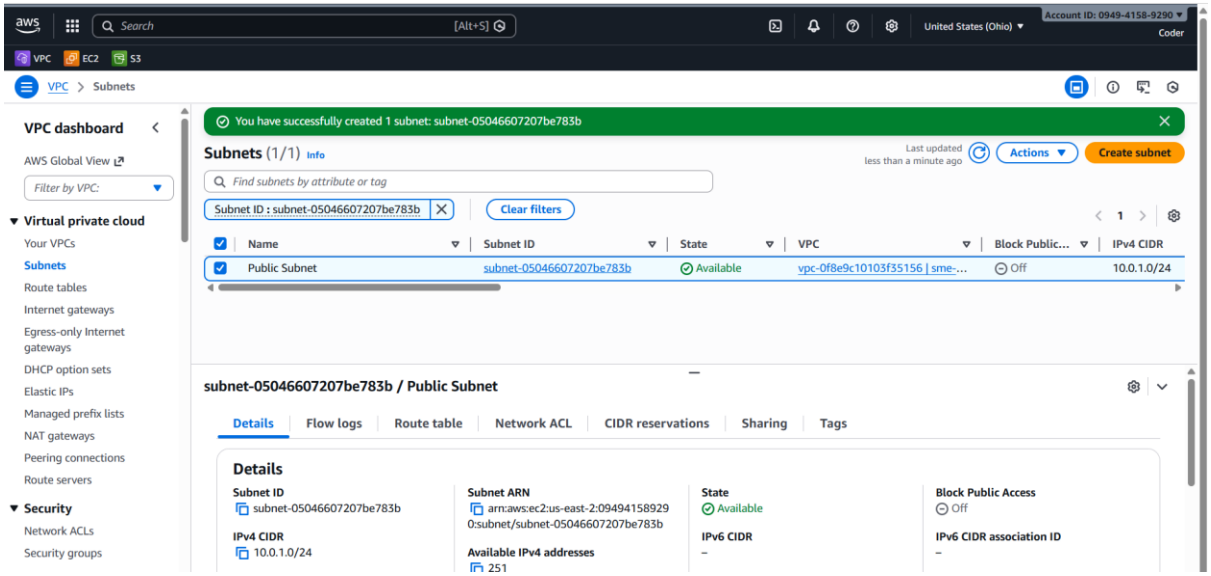


Step 4: Create Public Subnet

Subnets divide the VPC into smaller network segments.

- Create a subnet named Public-Subnet
- CIDR block: 10.0.1.0/24
- Select any availability zone
- Enable auto-assign public IPv4 address

The public subnet allows the EC2 instance to be accessed remotely via the internet.

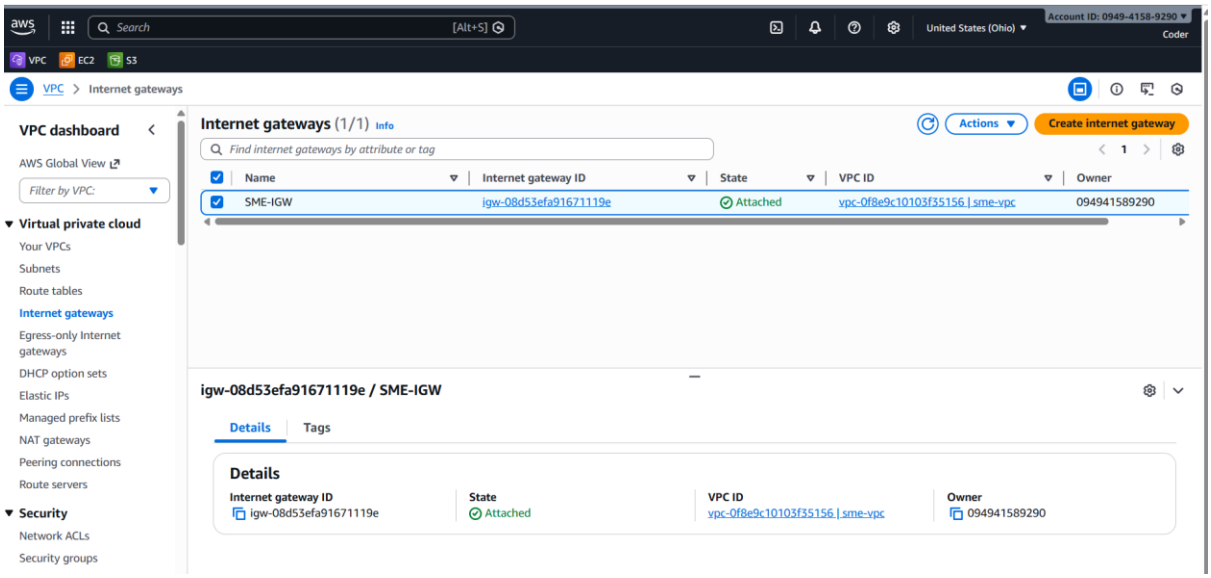


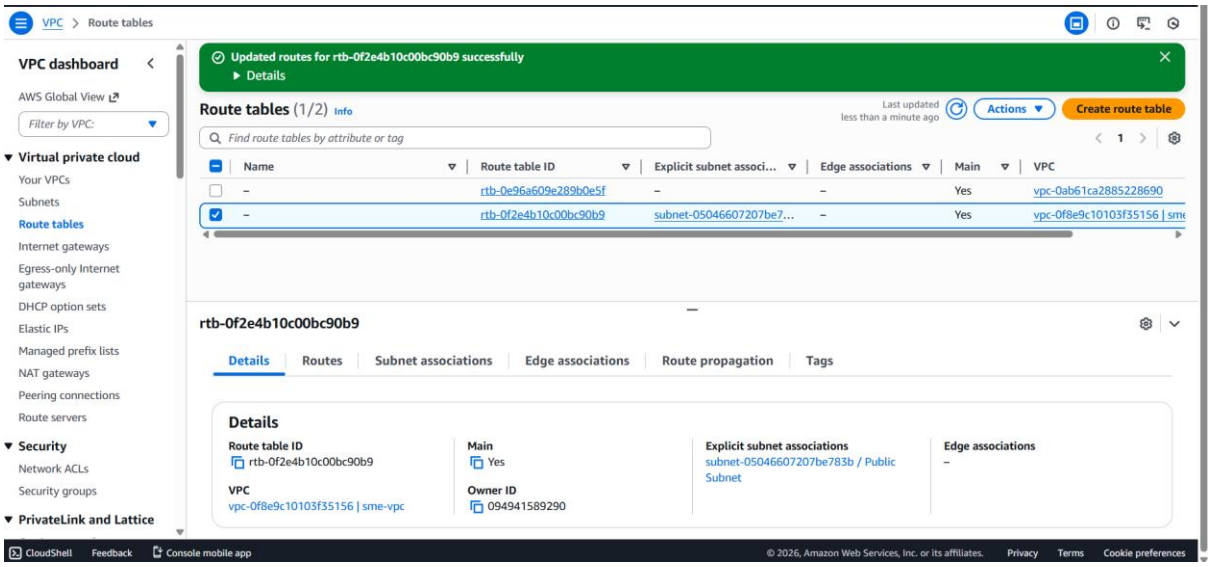
Step 5: Internet Gateway and Routing

An Internet Gateway enables communication between resources in the VPC and the internet.

- Create an Internet Gateway named SME-IGW
- Attach it to SME-VPC
- Create a Route Table named Public-RT
- Add route: 0.0.0.0/0 → SME-IGW
- Associate the route table with Public-Subnet

This configuration ensures outbound and inbound internet connectivity.



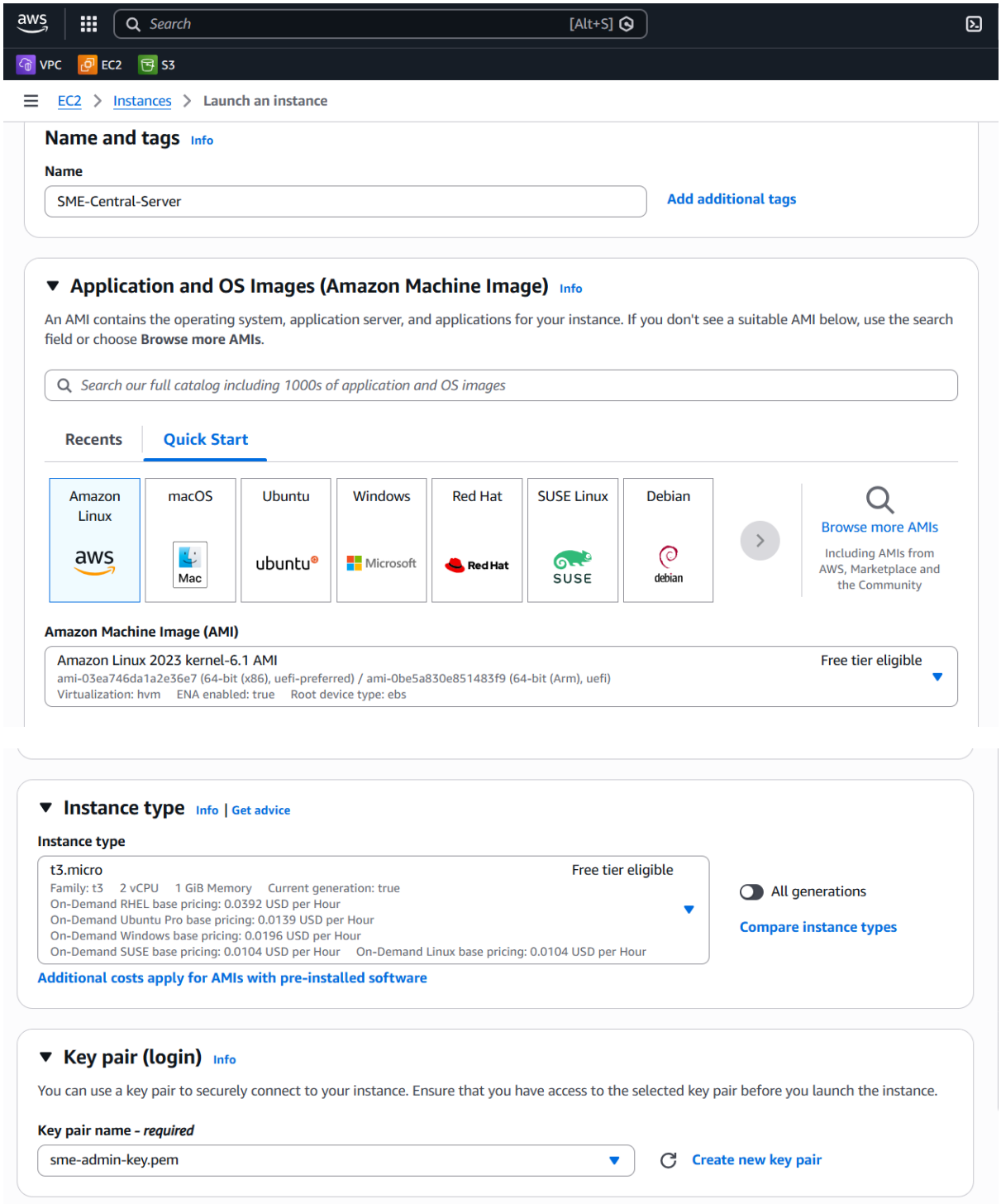


Step 6: Launch EC2 Instance (Central Server)

The EC2 instance acts as the centralized server replacing on-premise systems.

- Navigate to EC2 → Launch Instance
- Instance name: SME-Central-Server
- Choose AMI: Amazon Linux 2023
- Instance type: t2.micro or t3.micro (Free Tier)
- Create a new key pair named sme-admin-key.pem

This virtual machine provides compute resources for hosting enterprise services.

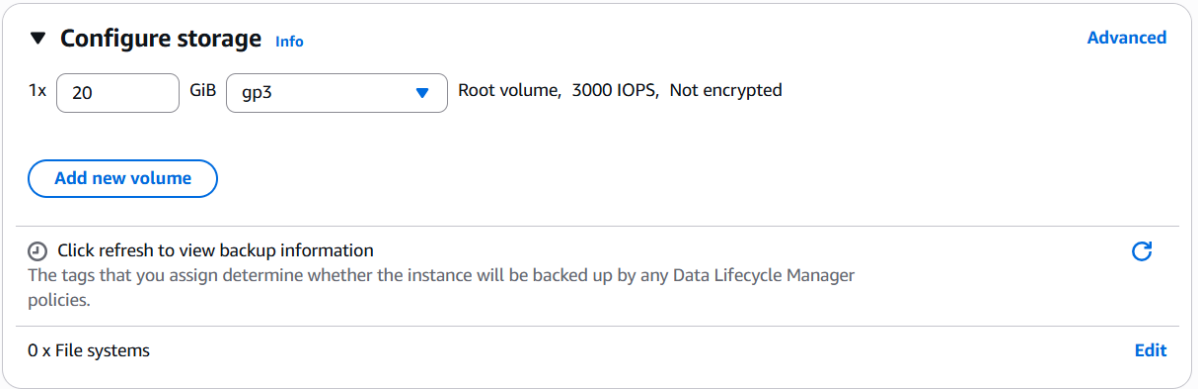


Step 7: Configure Storage (EBS)

Elastic Block Store (EBS) provides persistent storage for the EC2 instance.

- Root volume type: gp3
- Storage size: 20 GB
- Used for application data and centralized files

EBS ensures data is retained even if the EC2 instance is stopped.

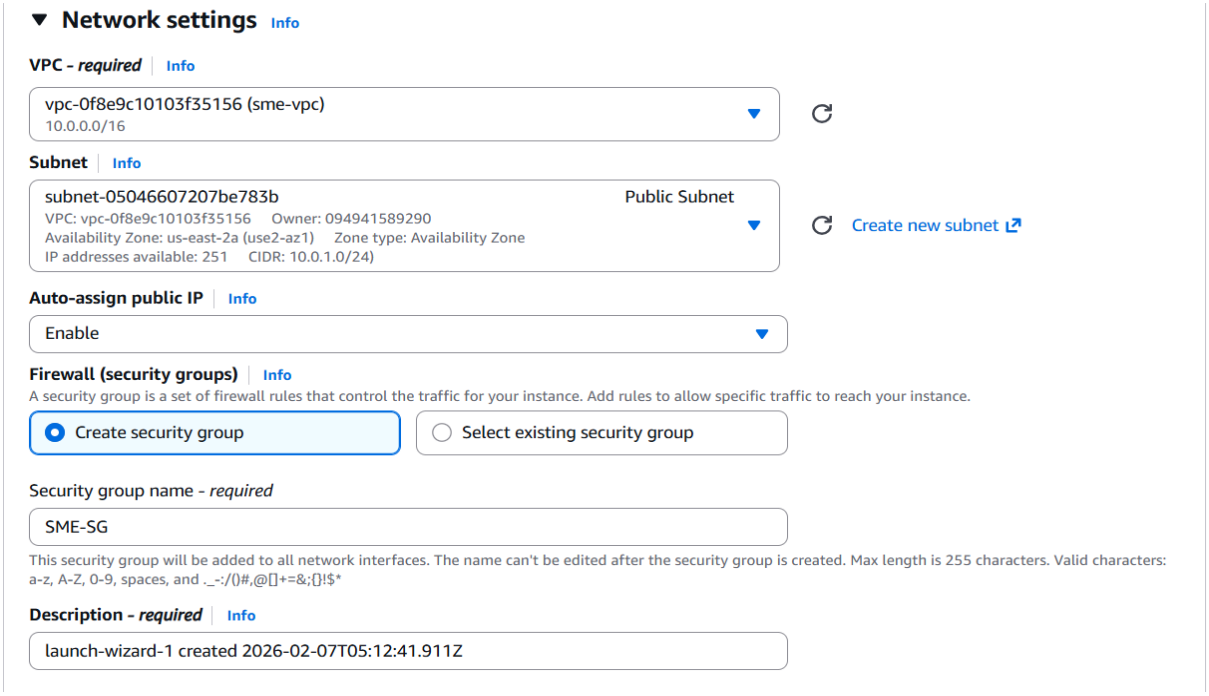


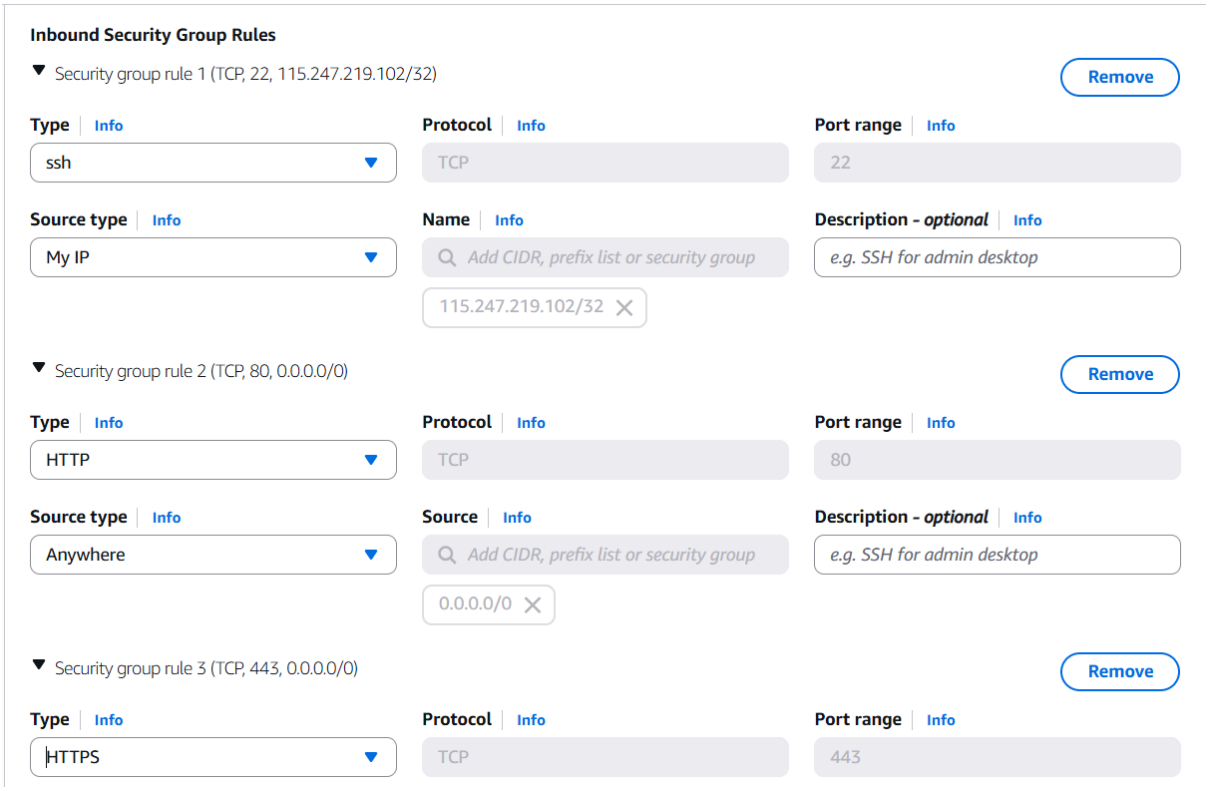
Step 8: Security Group Configuration

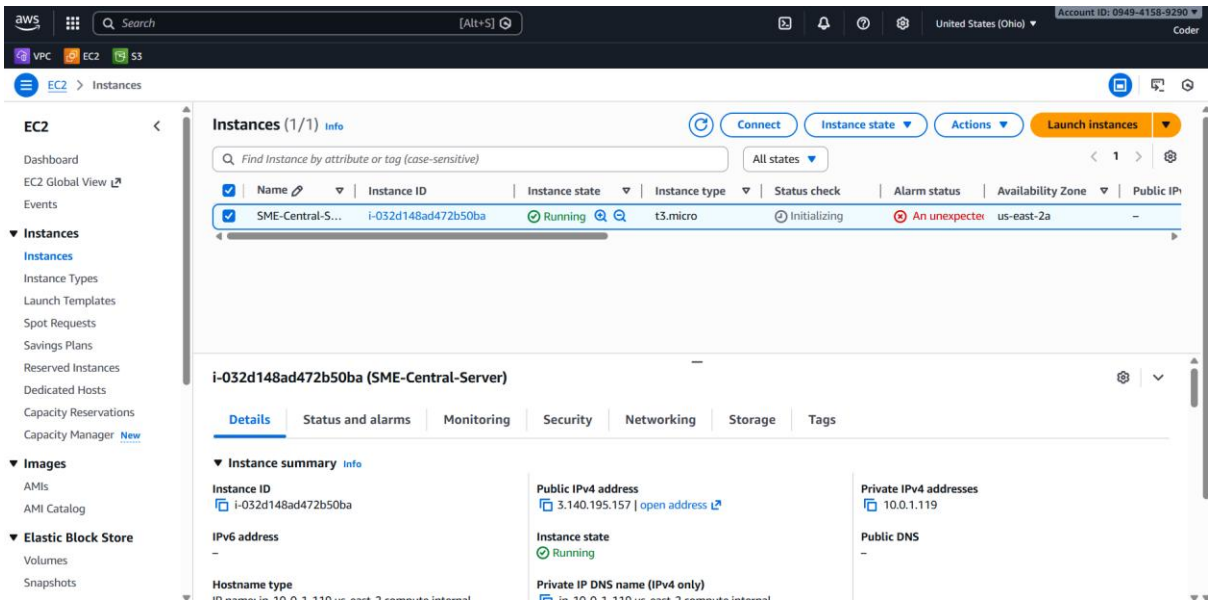
Security Groups act as virtual firewalls controlling inbound and outbound traffic.

- Create a Security Group named SME-SG
- Allow SSH (port 22) from My IP
- Optionally allow HTTP (80) and HTTPS (443)

Restricting SSH access improves server security.







Step 9: Connect to EC2 Instance

Secure Shell (SSH) is used for remote access to the EC2 instance.
In this project, the EC2 instance was accessed using **AWS EC2 Instance Connect**, which provides a secure, browser-based SSH connection without requiring local key pair configuration.

Steps followed:

- Navigate to **EC2 → Instances**
- Select the instance **SME-Central-Server**
- Click **Connect**
- Choose **EC2 Instance Connect**
- Click **Connect**

Upon successful login, the Amazon Linux 2023 terminal is displayed, confirming that the EC2 instance is running correctly and is securely accessible for administrative tasks.

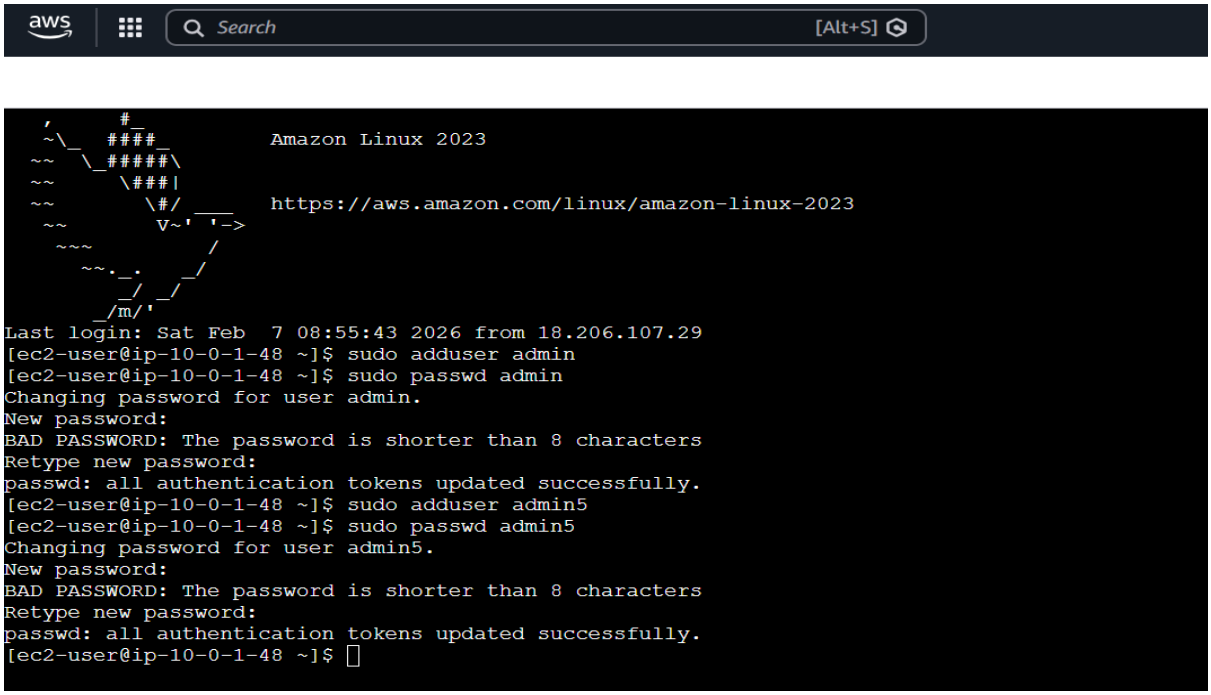
Step 10: User and Admin Simulation

Creating users simulates multiple administrators managing the server.

sudo adduser admin1

sudo passwd admin1

This reflects real-world IT operations in an enterprise environment.

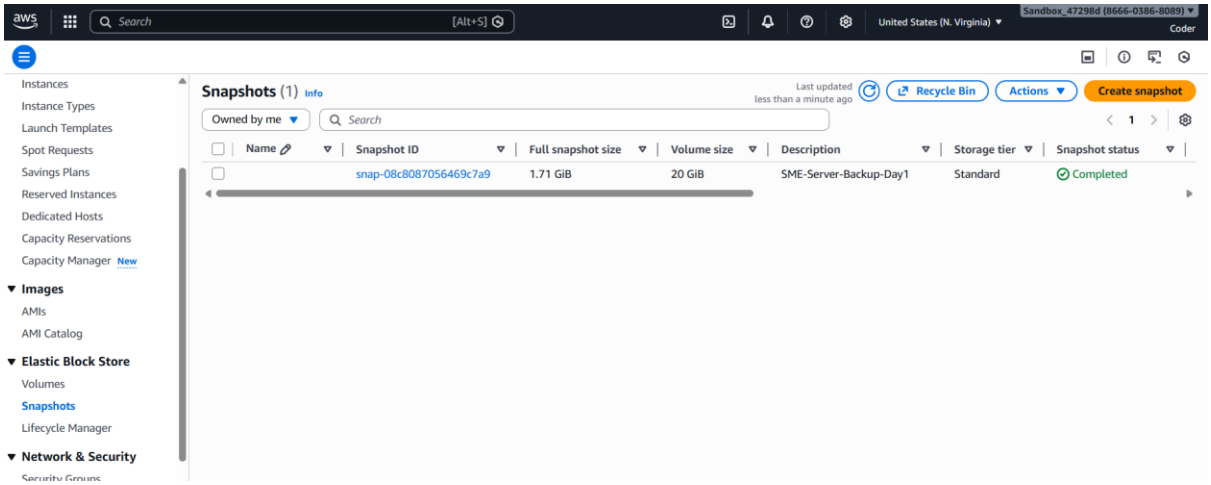


Step 11: Backup Using Snapshots

Snapshots are point-in-time backups of EBS volumes.

- Navigate to EC2 → Volumes
- Select the root EBS volume
- Create a snapshot named SME-Server-Backup-Day1

Snapshots provide data protection and disaster recovery.



Step 12: Recovery Simulation

This step demonstrates system recovery in case of failure.

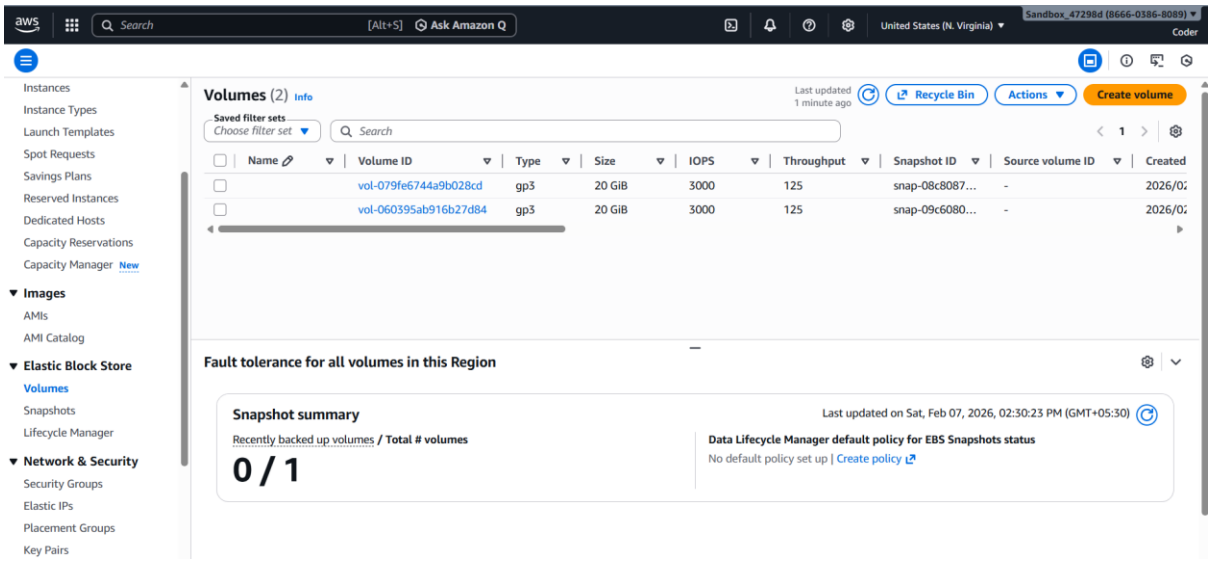
- Create a new EBS volume from the snapshot

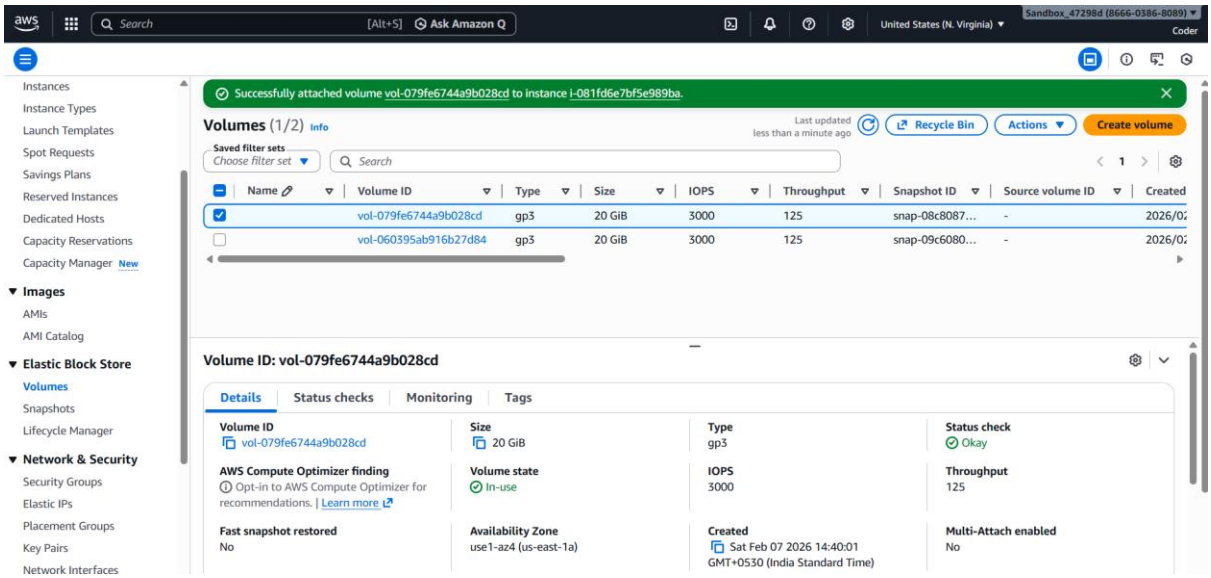
- Attach it to the EC2 instance
- Mount the volume to access recovered data

lsblk

sudo mount /dev/xvdf /mnt/recovery

This proves the effectiveness of the backup strategy.





```
[ec2-user@ip-10-0-1-48 ~]$ lsblk
NAME          MAJ:MIN RM SIZE RO TYPE MOUNTPOINTS
nvme0n1       259:0    0  20G  0 disk
├─nvme0n1p1   259:1    0  20G  0 part /
├─nvme0n1p127 259:2    0   1M  0 part
└─nvme0n1p128 259:3    0  10M  0 part /boot/efi
nvme1n1       259:4    0  20G  0 disk
├─nvme1n1p1   259:5    0  20G  0 part
├─nvme1n1p127 259:6    0   1M  0 part
└─nvme1n1p128 259:7    0  10M  0 part
[ec2-user@ip-10-0-1-48 ~]$ sudo mkdir /mnt/recovery
```

```
[ec2-user@ip-10-0-1-48 ~]$ sudo mount -t xfs -o ro,nouuid /dev/nvme1n1p1 /mnt/recovery
[ec2-user@ip-10-0-1-48 ~]$ df -h
Filesystem       Size  Used Avail Use% Mounted on
devtmpfs         4.0M     0  4.0M   0% /dev
tmpfs            459M     0  459M   0% /dev/shm
tmpfs            184M  448K  183M   1% /run
/dev/nvme0n1p1    20G  1.6G   19G   8% /
tmpfs            459M     0  459M   0% /tmp
/dev/nvme0n1p128  10M  1.3M  8.7M  13% /boot/efi
tmpfs             92M     0   92M   0% /run/user/1000
/dev/nvme1n1p1    20G  1.5G   19G   8% /mnt/recovery
[ec2-user@ip-10-0-1-48 ~]$
```

7.Backup and Recovery Strategy

- Regular EBS snapshots
- Manual restoration in case of failure
- Ensures business continuity

8.Challenges Faced

- Learning AWS networking concepts
- Understanding security groups and access control
- Manual configuration without automation tools

9.Conclusion

This project demonstrates how a small enterprise can migrate from decentralized systems to a centralized, cloud-based infrastructure using AWS. The setup improves reliability, security, and manageability while staying within free-tier constraints.

Future Enhancements

- Auto Scaling and Load Balancer
- S3-based backups
- CloudWatch monitoring
- IAM role-based access control