L1 -Project Report

Centralized File Sharing and Backup System

Contents

1. Introduction

2. Objectives

3. Services Used

4. Architecture

5. Project Implementation details

6. Output Screenshot

Project submitted by

Jayashree L

Centralized File Sharing and Backup System

1. Introduction:

The Centralized File Sharing and Backup System is designed to enable seamless collaboration between two EC2 instances by providing a shared storage environment using Amazon EFS. Both instances access the same directory structure, ensuring consistent project data. To maintain reliable backups and prevent data loss, all changes within the shared directory are automatically replicated to Amazon S3 within seconds.

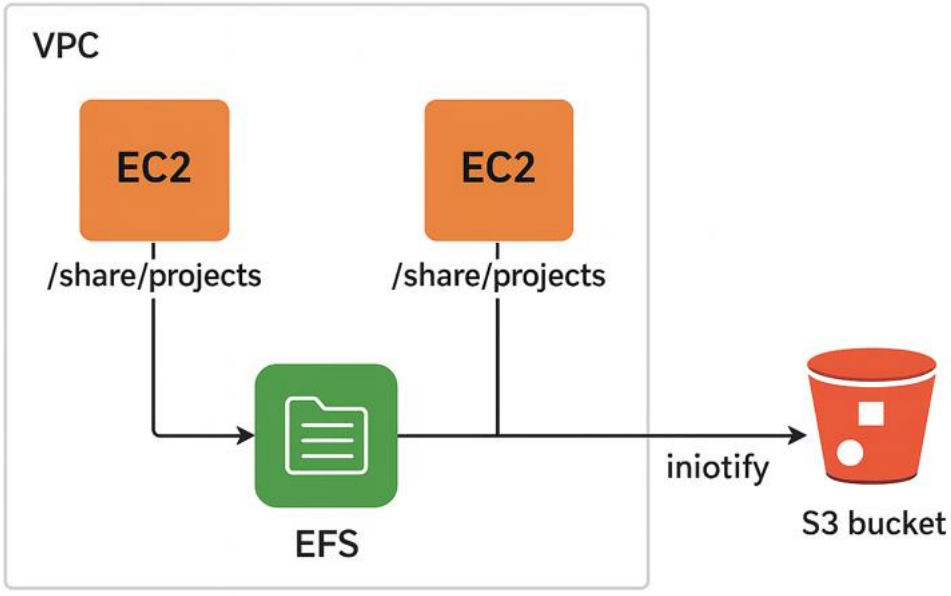
2. Objectives:

The objective of this project is to design and implement an AWS architecture where two EC2 instances, hosted in a custom VPC, share an Amazon EFS file system. Any file or directory created or modified inside the mount point /share/projects on either instance must be automatically uploaded to an Amazon S3 bucket within seconds.

3. Services used:

- Amazon EC2
- Amazon VPC
- Amazon EFS
- Amazon S3
- IAM

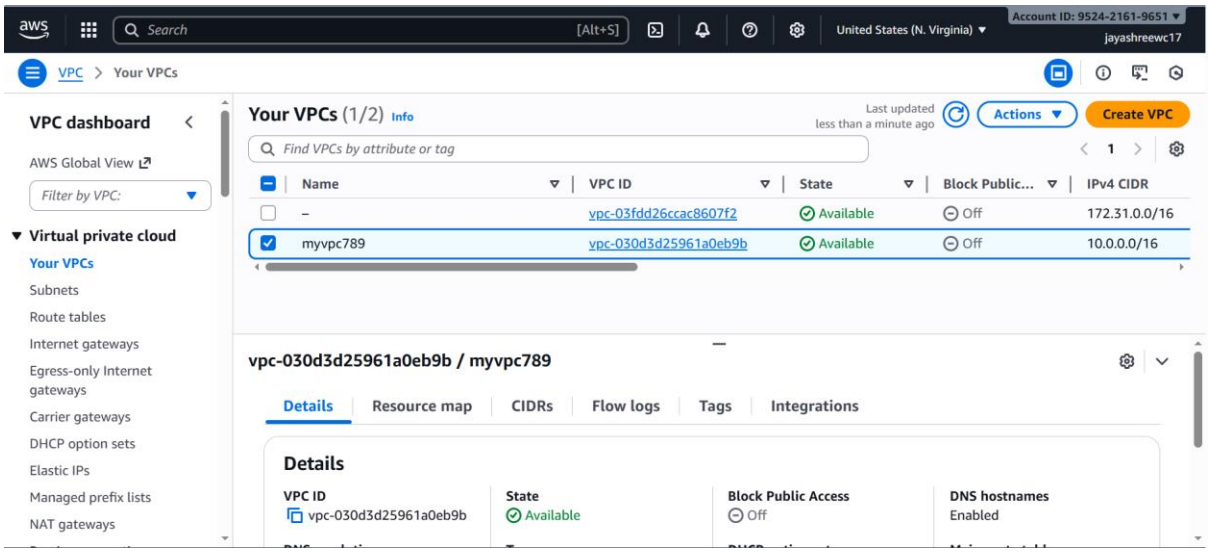
4. Architecture



5. Project Implementation details

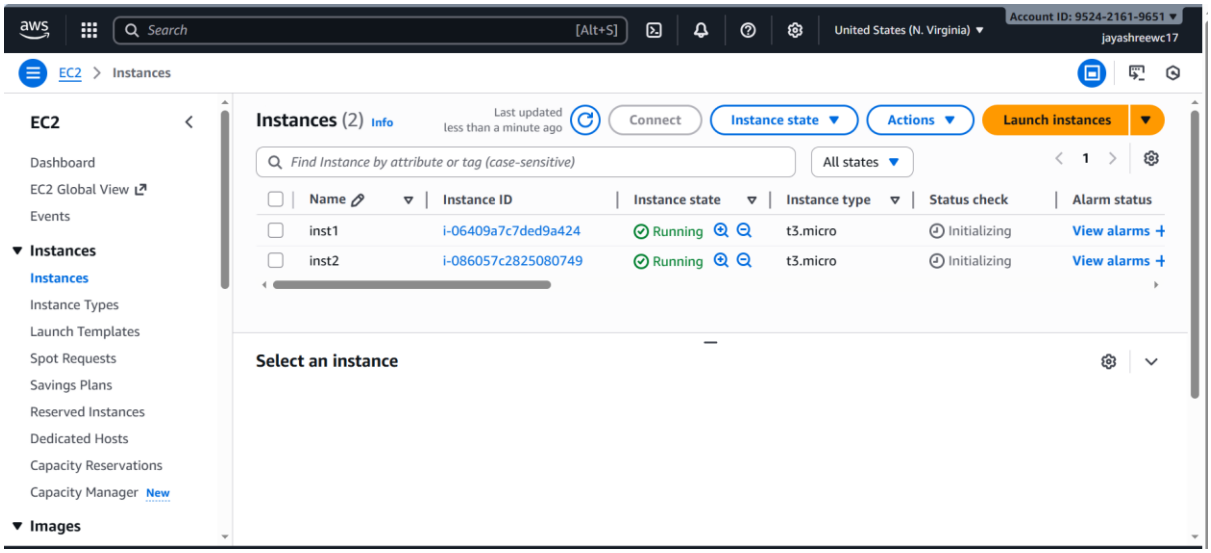
Step 1: Setup VPC and EC2 Instances

1. Create a **custom VPC**



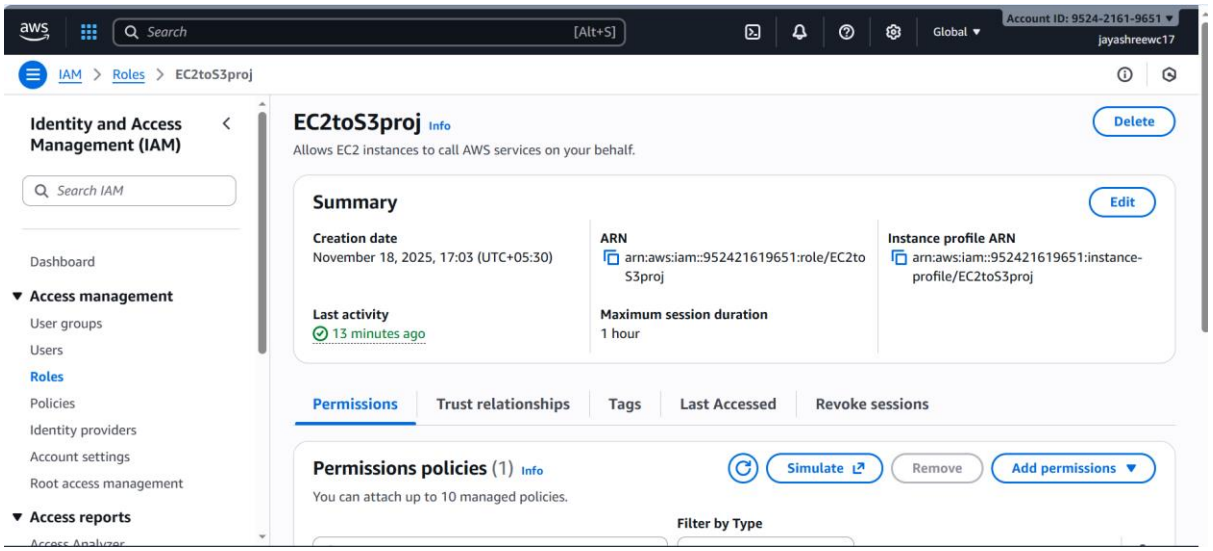
2. Launch **two EC2 instances** in this VPC.
    - Choose AMI: Amazon Linux
    - Create instance1 in subnet -1

      and instance 2 in subnet- 2.

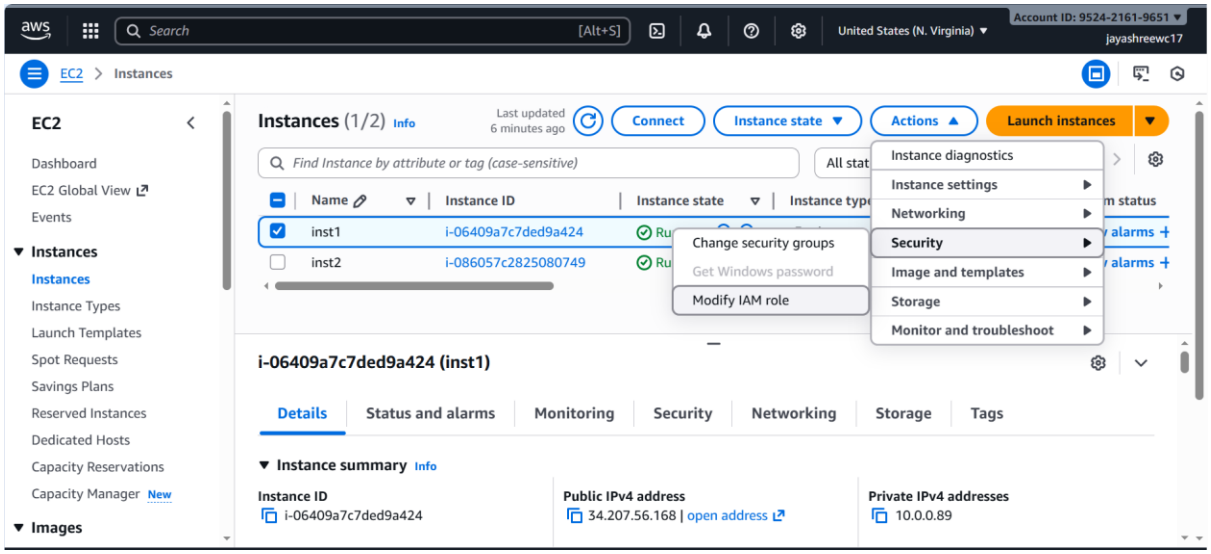


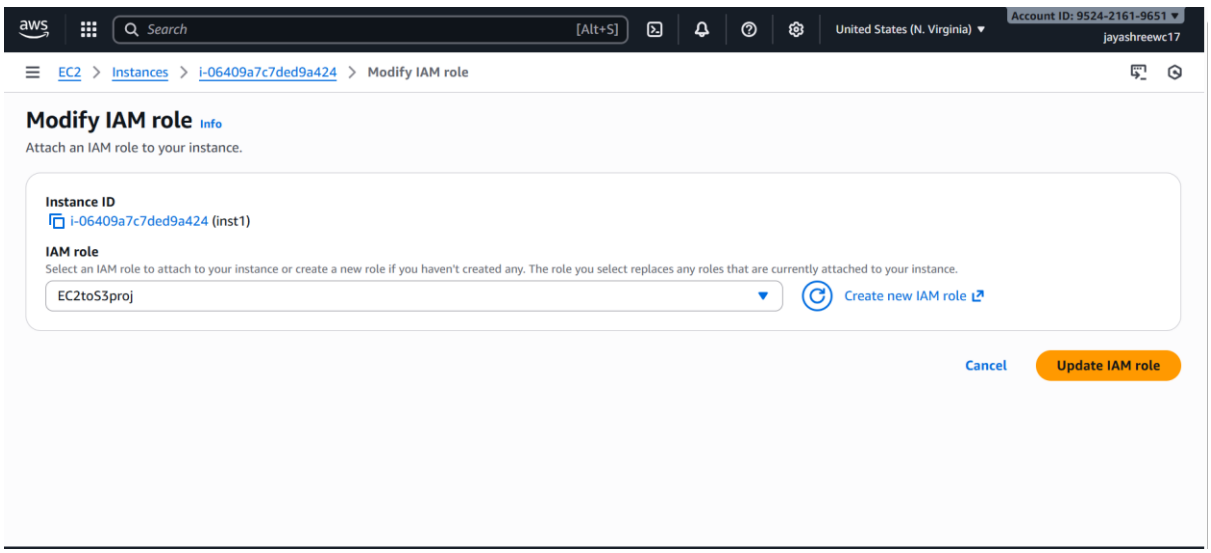
Step 2 : Create an IAM role

Create an IAM role with AmazonS3Fullaccess permission.



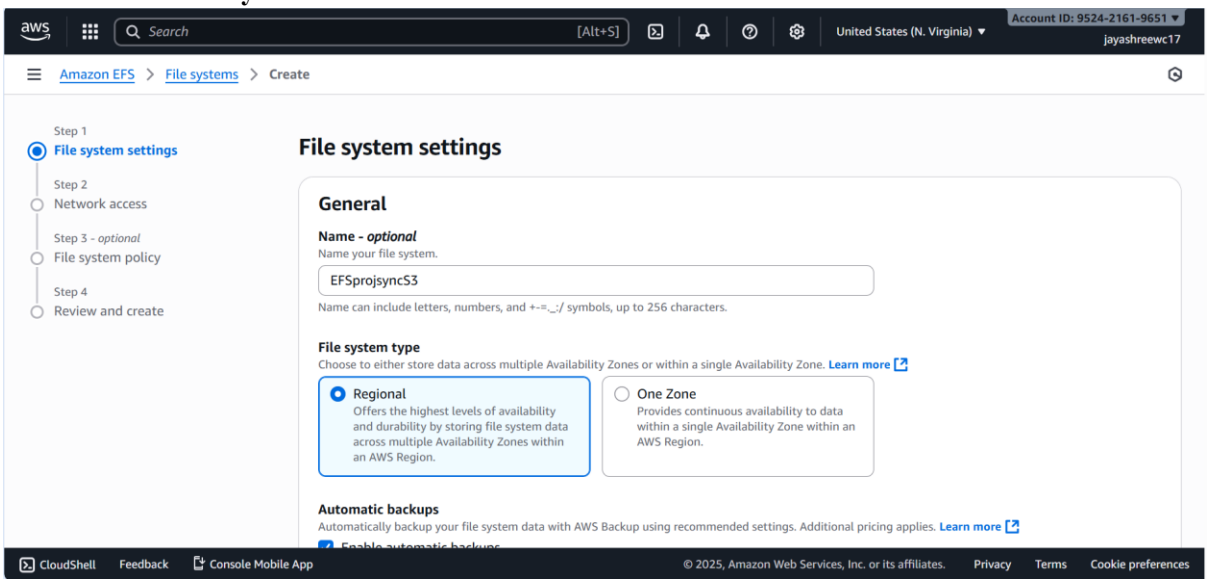
and attach the role to the two EC2 instances created already.

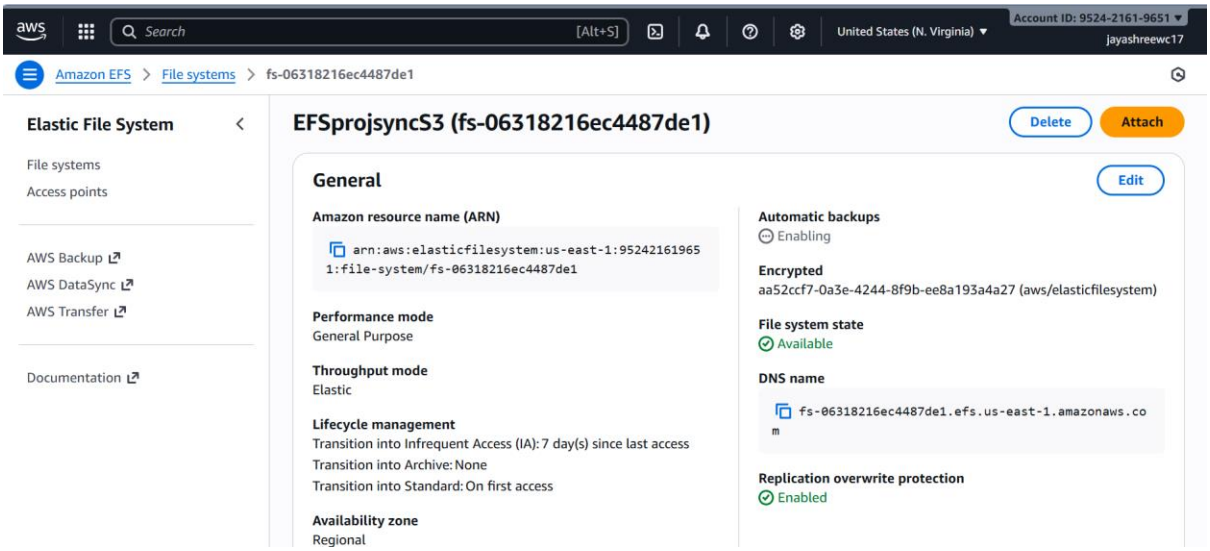




Step 3: Create and Mount EFS

1. Create an **EFS file system** in the same VPC.





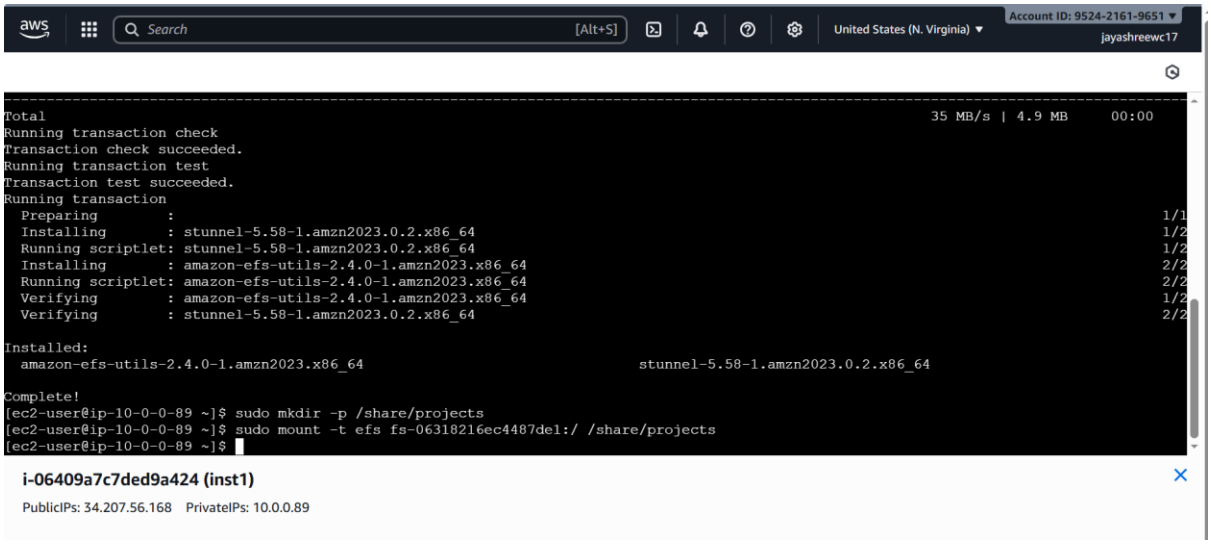
2. Create a **mount target** in the same subnet as your EC2s.
3. Mount EFS on both EC2 instances:

sudo yum install -y amazon-efs-utils *# Amazon Linux*

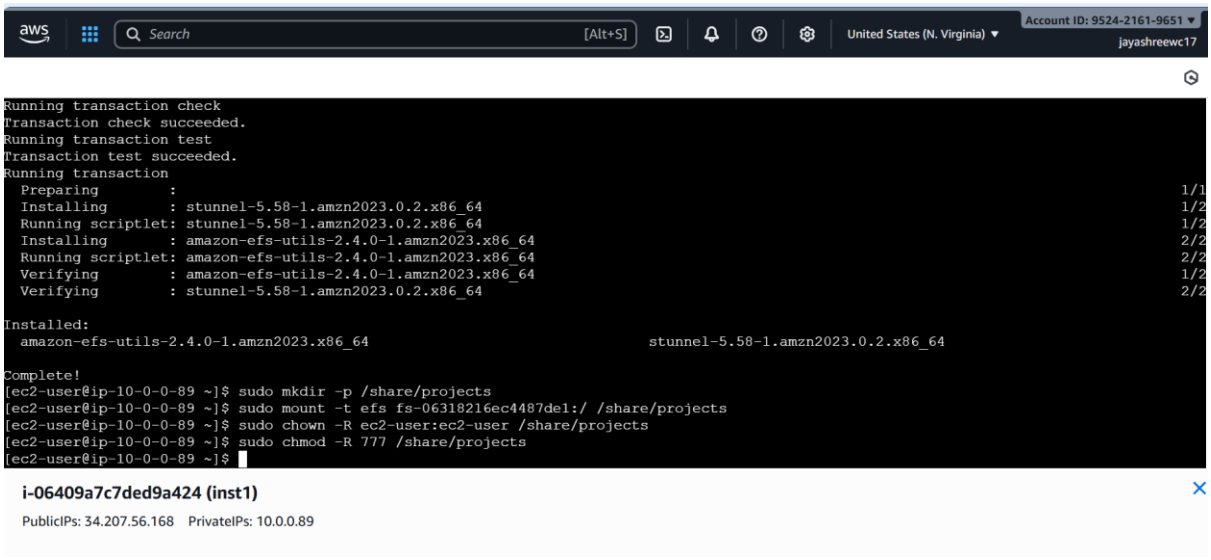
sudo mkdir -p /share/projects

sudo mount -t efs fs-xxxxxx:/ /share/projects

- /share/projects will be our **centralized folder**.

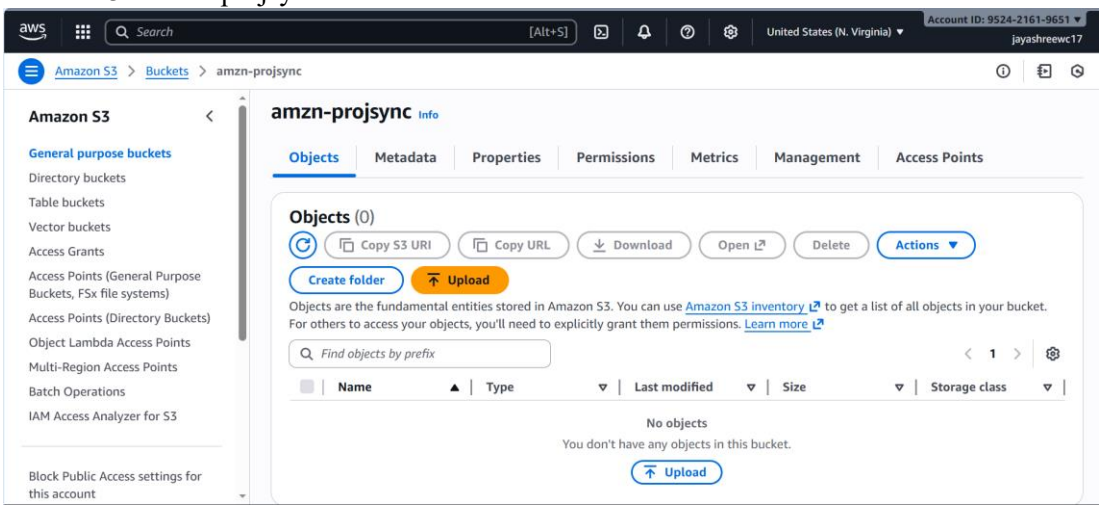


4. Make the folder writable:



Step 4: Create an S3 Bucket

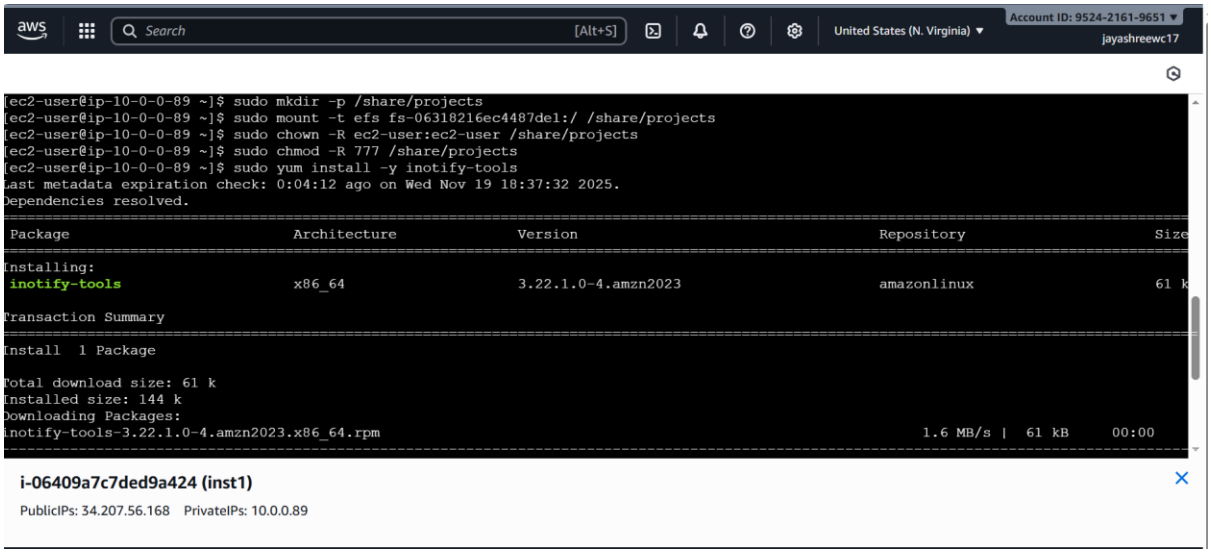
- bucket: s3://amzn-projsync



- Make sure EC2 **IAM role** can access it.

Step 5: Install Dependencies

sudo yum install -y inotify-tools



which aws

- Ensure AWS CLI works with our IAM role:

aws s3 ls s3://amzn-projsync

Step 6: Create Inotify Sync Script

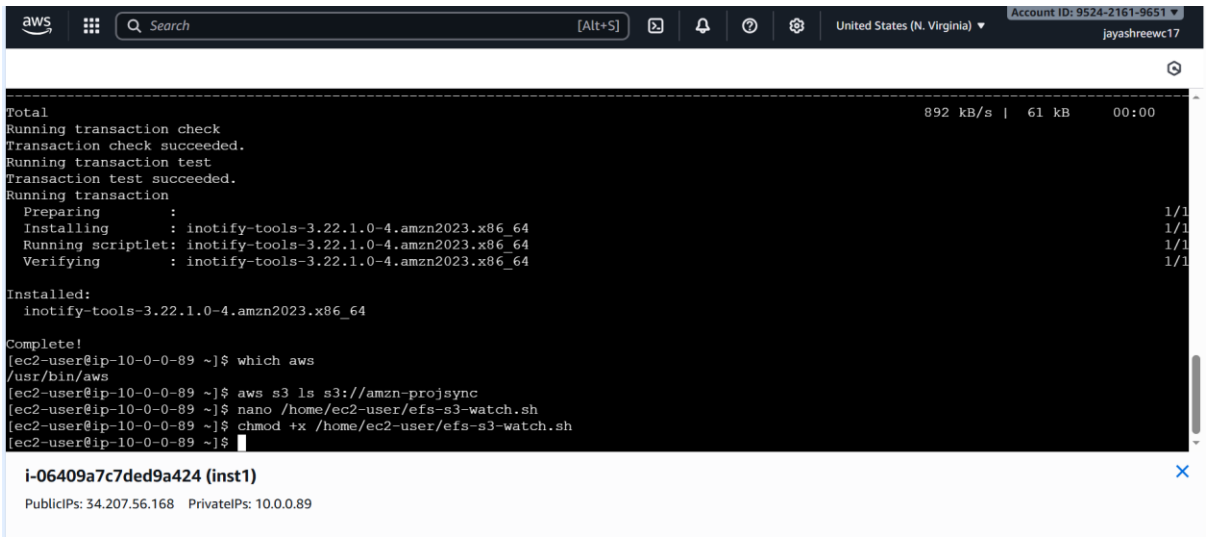
. Implement a near real-time sync mechanism using inotify-based monitoring

1. Create the script file:

nano /home/ec2-user/efs-s3-watch.sh

2. Make it executable:

chmod +x /home/ec2-user/efs-s3-watch.sh

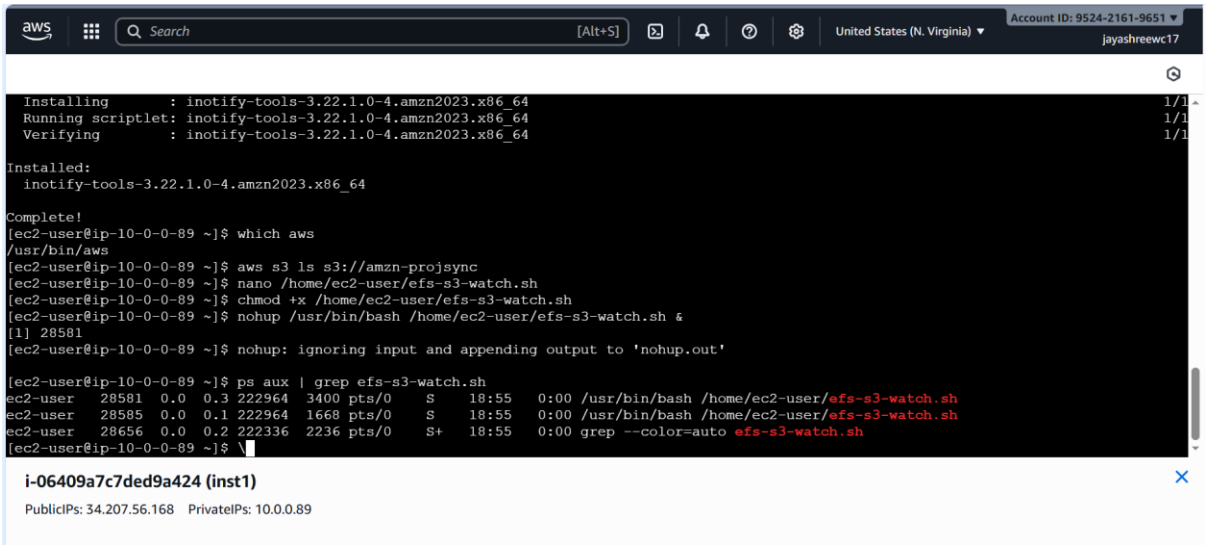


Step 7: Run the Script in Background

nohup /usr/bin/bash /home/ec2-user/efs-s3-watch.sh &

- This keeps it running after you log out.
- Check that it's running:

ps aux | grep efs-s3-watch.sh



Step 8: Test Automatic Sync

1. Create a file in EFS:

echo "Hello from EC2-1" > /share/projects/testfile1.txt

2. Wait a few seconds.
3. Check S3:

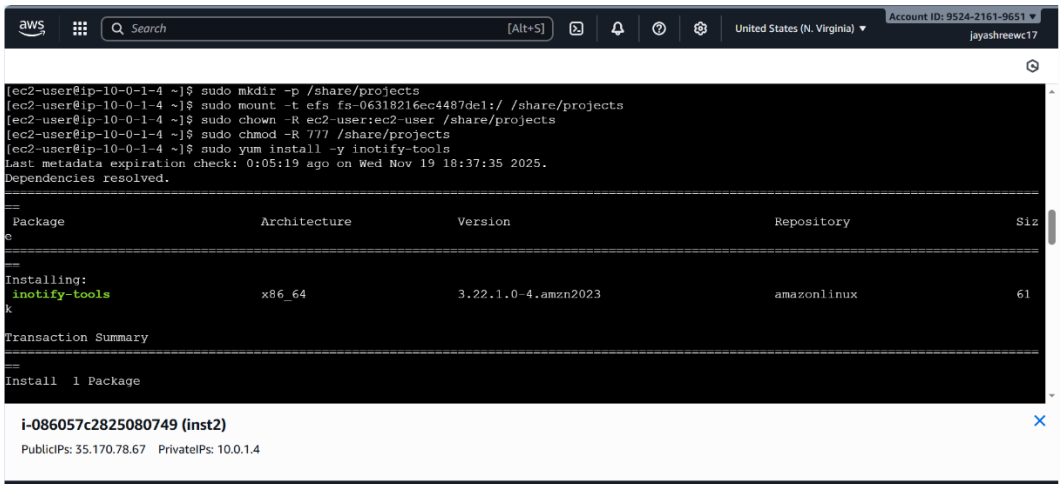
aws s3 ls s3://amzn-projsync

- File should appear automatically.
- Check log:

cat /var/log/efs-s3-sync.log

Step 9: Repeat on Second EC2 Instance

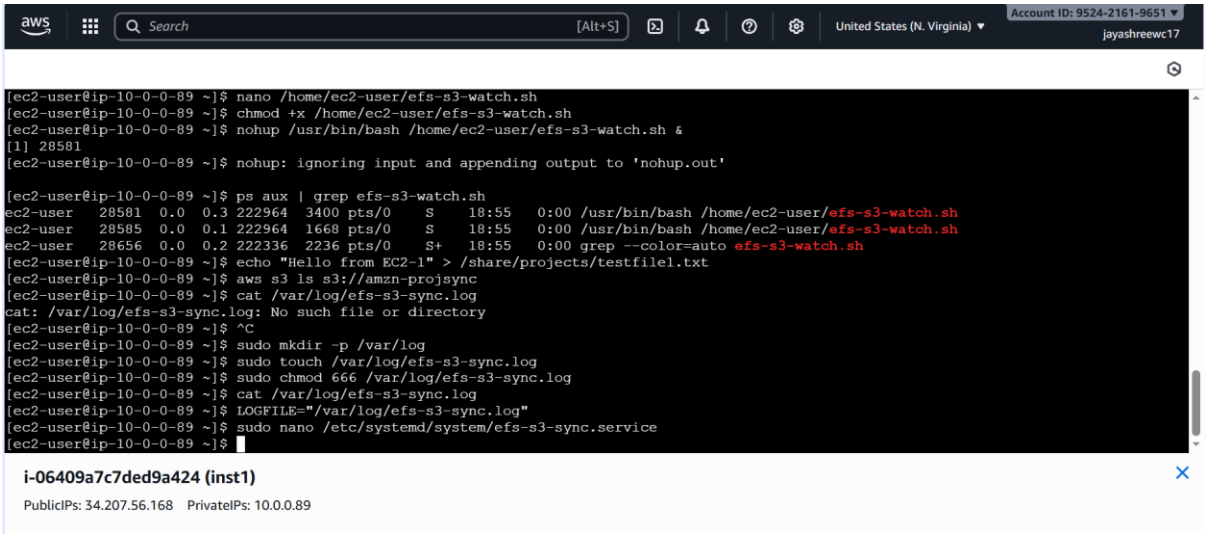
- Mount the same EFS on EC2-2.
- Run the **same inotify script** there.

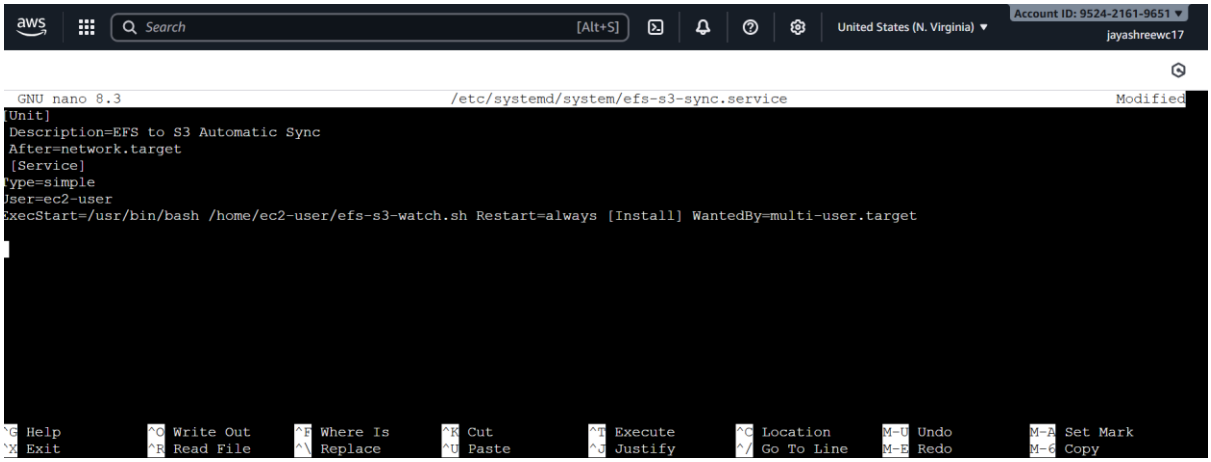


Step 10: Make Script Start on Boot (systemd service)

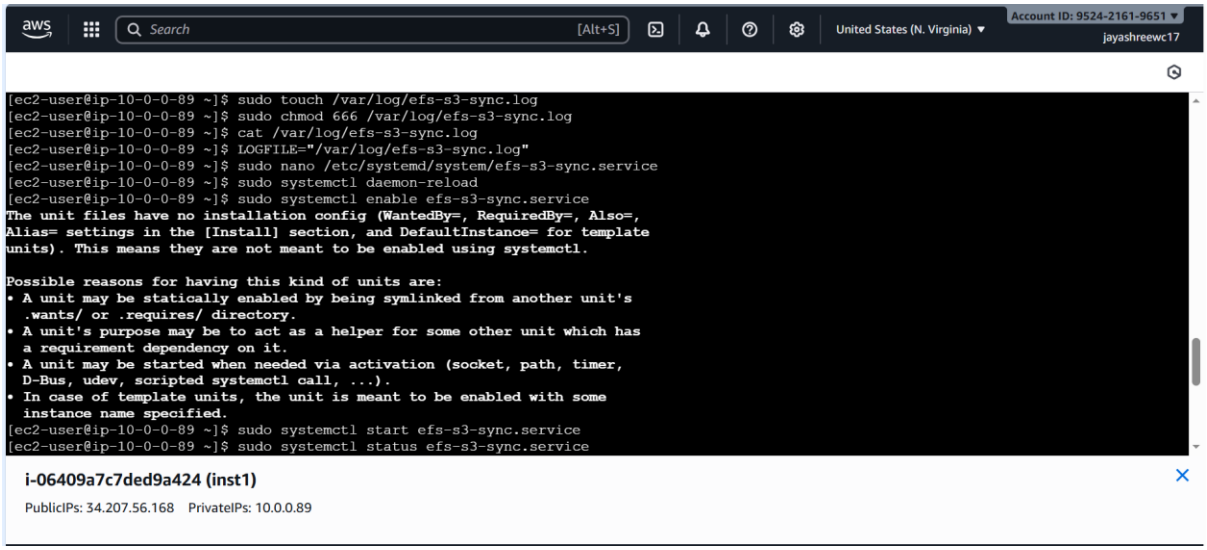
1. Create a systemd service file:

sudo nano /etc/systemd/system/efs-s3-sync.service

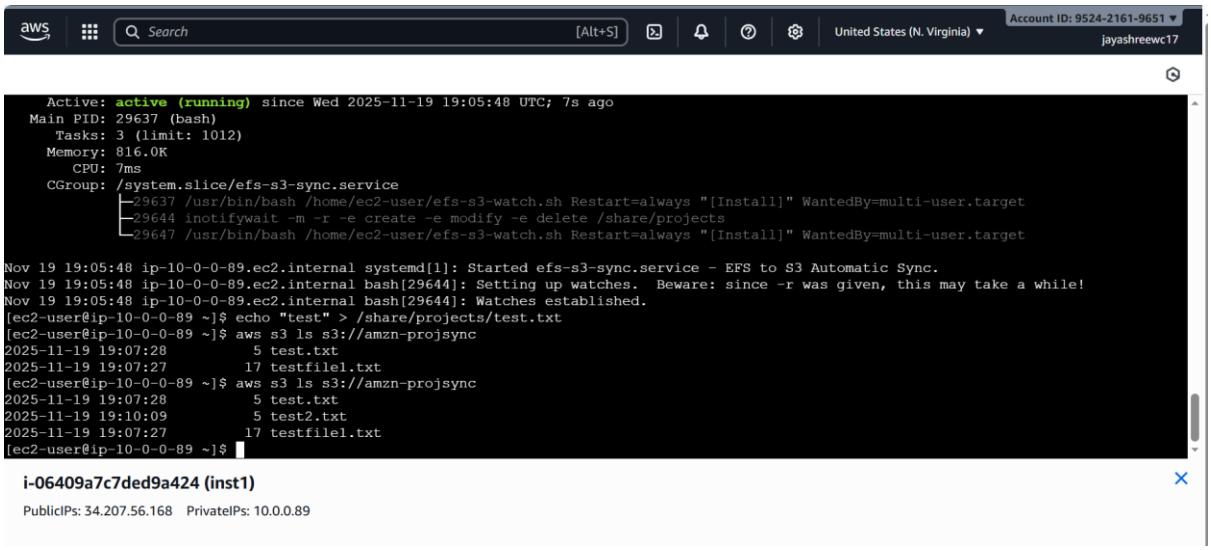




3. Enable and start:



If the service is *active (running)*,Now test file sync.



6. Output screenshots