

Disk Sharing using EBS and EFS

Goal: Configure a EBS such that, it can be shared between different instances.

1. Create and set up an EBS such that it should be enabled by two ec2 instances
2. Verify the working
3. Study about elastic File system in AWS.
4. Configure an EFS shared between different ec2 instances.

EBS (Elastic Block Store) and EFS (Elastic File System) are both storage services provided by AWS (Amazon Web Services), but they serve different purposes:

1. EBS (Elastic Block Store):

- **Type:** Provides block-level storage, meaning it is more akin to having a raw disk attached to your server.
- **Usage:** Typically used with EC2 instances where you need low-latency and high-throughput storage that behaves like a physical hard drive.
- **Performance:** Designed for workloads that require high IOPS (Input/Output Operations Per Second) and low-latency storage.
- **Scalability:** Can be scaled vertically (by increasing volume size) but is tied to a single EC2 instance at a time.
- **Use Cases:** Ideal for databases, applications that require frequent updates and access to individual files, and boot volumes for EC2 instances.

2. EFS (Elastic File System):

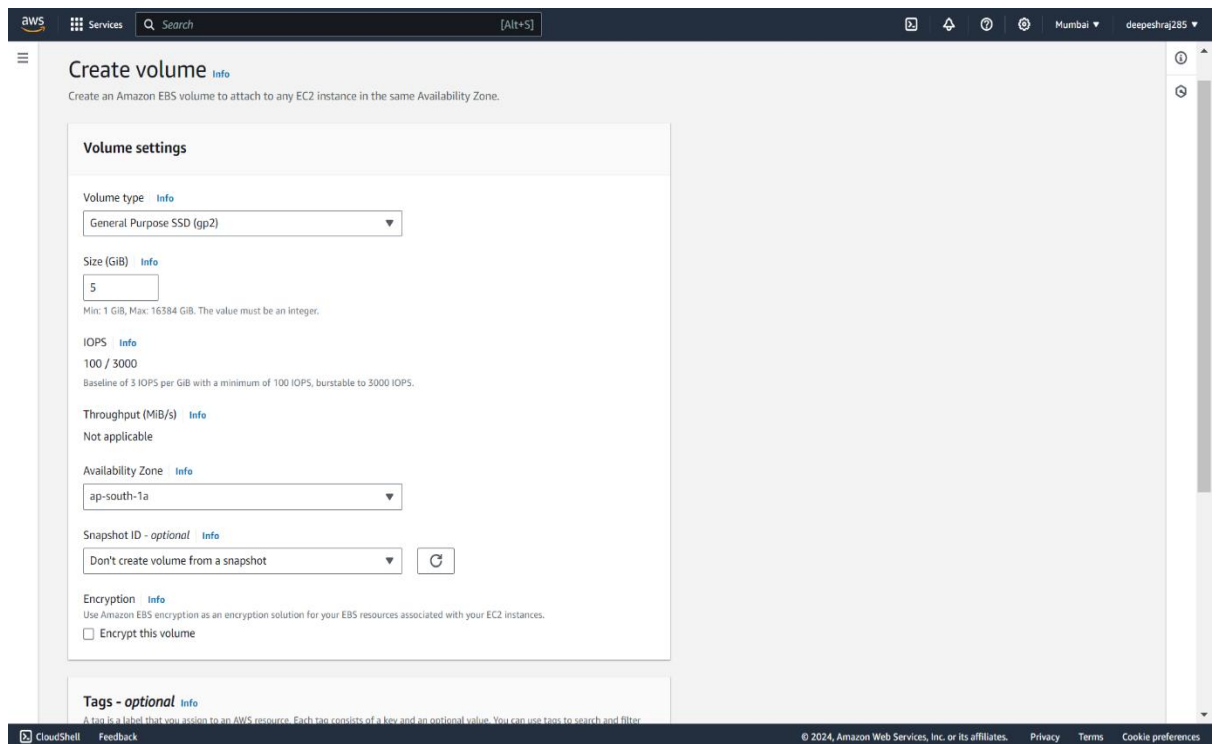
- **Type:** Provides file-level storage, offering a managed NFS (Network File System) that can be accessed by multiple EC2 instances concurrently.
- **Usage:** Suitable for scenarios where multiple EC2 instances need to access the same data simultaneously, making it highly scalable and flexible.
- **Performance:** Optimized for scalability rather than ultra-low latency, so it's great for workloads that need shared access to files across instances.
- **Scalability:** Scales horizontally with the amount of data stored, and multiple EC2 instances can mount an EFS file system simultaneously.
- **Use Cases:** Shared content repositories, development environments, content management systems, and applications that require shared access to data.

In summary, choose EBS for scenarios requiring high-performance, low-latency access to block storage that behaves like a physical disk attached to your instance. Choose EFS when

you need scalable, shared file storage accessible from multiple instances, where data consistency and concurrent access are important.

EBS

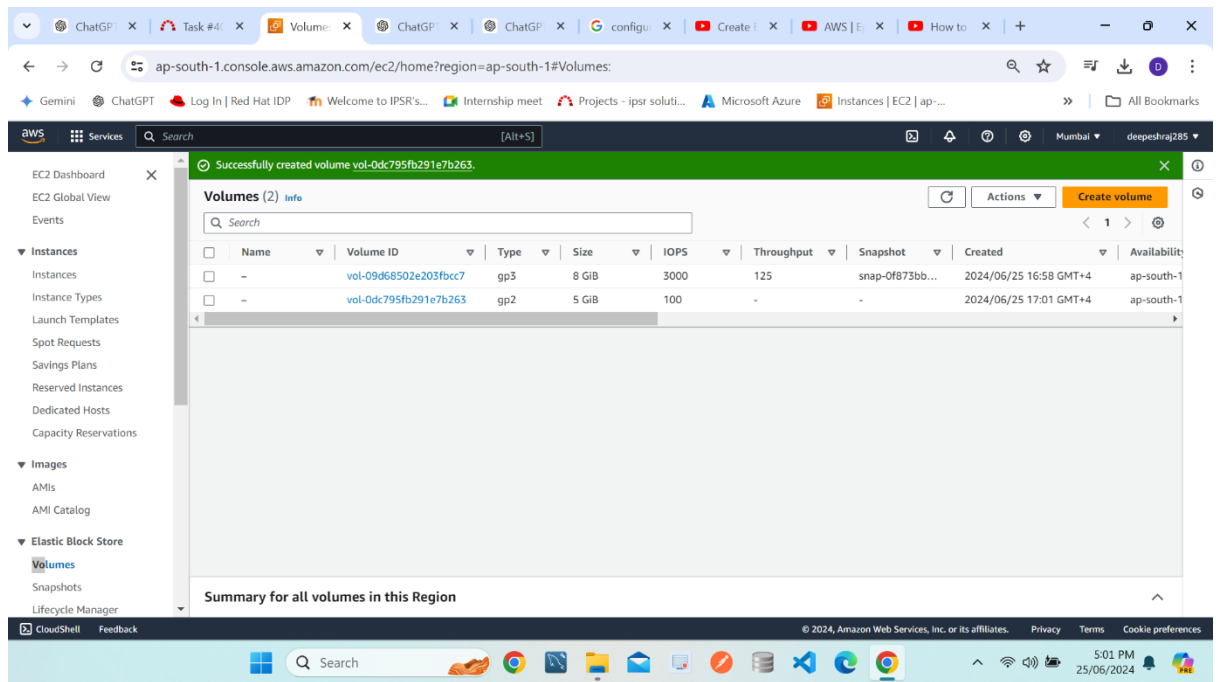
1. Create first AWS Ec2 instances on Linux machine
2. Create an EBS Volume
 - a. Navigate to EC2 Dashboard.
 - b. On the left-hand side, click on Volumes under the Elastic Block Store section.
 - c. Click Create Volume.
 - d. Configure the volume:
 - i. Size: Choose an appropriate size (e.g., 5 GiB for the free tier).
 - ii. Availability Zone: Choose the same availability zone as your EC2 instances.
 - iii. Other settings can be left as default.
 - e. Click Create Volume.



The screenshot shows the 'Create volume' page in the AWS Management Console. The page title is 'Create volume' with an 'Info' link. Below the title is a subtitle: 'Create an Amazon EBS volume to attach to any EC2 instance in the same Availability Zone.' The main content area is titled 'Volume settings' and contains several configuration options:

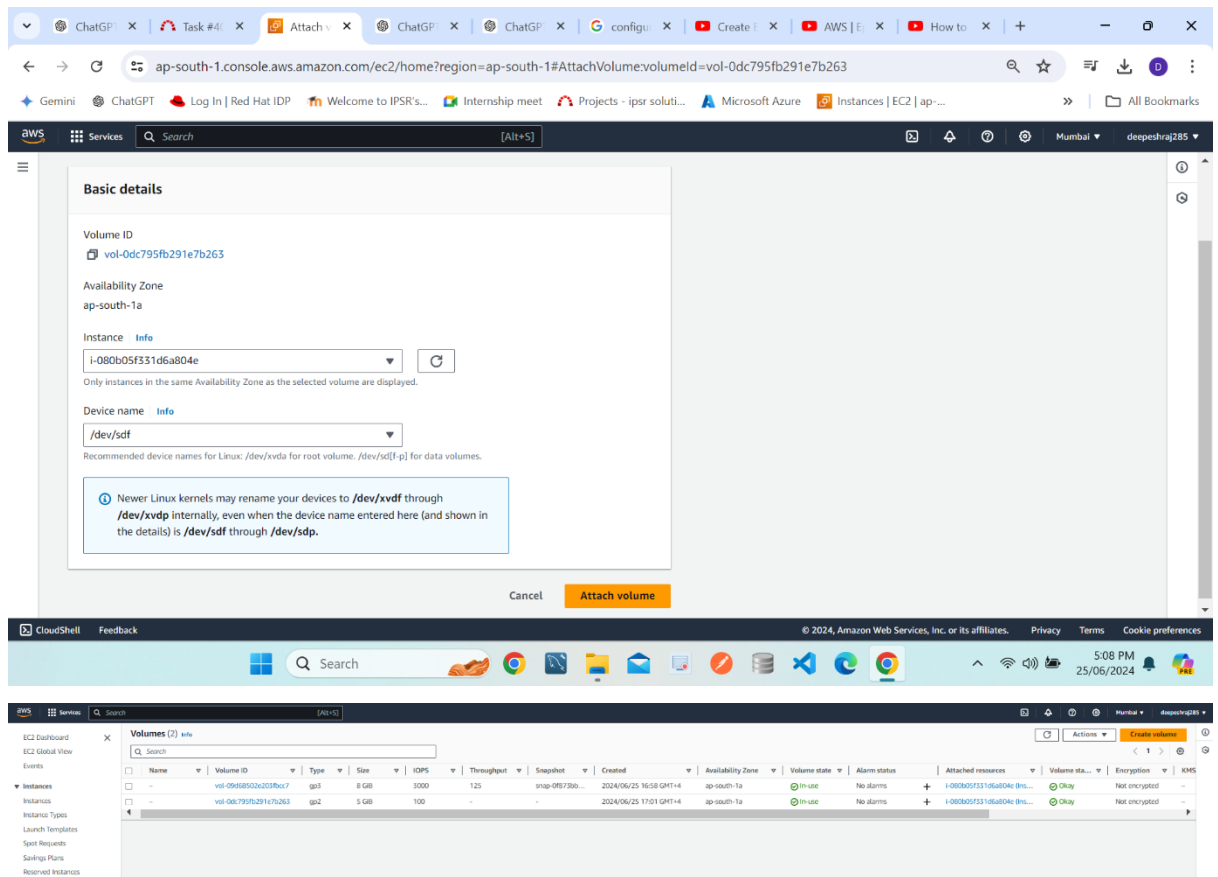
- Volume type:** A dropdown menu set to 'General Purpose SSD (gp2)' with an 'Info' link.
- Size (GiB):** A text input field containing '5' with an 'Info' link. Below it, a note states: 'Min: 1 GiB, Max: 16384 GiB. The value must be an integer.'
- IOPS:** A text input field containing '100 / 3000' with an 'Info' link. Below it, a note states: 'Baseline of 3 IOPS per GiB with a minimum of 100 IOPS, burstable to 3000 IOPS.'
- Throughput (MiB/s):** A text input field containing 'Not applicable' with an 'Info' link.
- Availability Zone:** A dropdown menu set to 'ap-south-1a' with an 'Info' link.
- Snapshot ID - optional:** A dropdown menu set to 'Don't create volume from a snapshot' with an 'Info' link and a refresh button.
- Encryption:** A section with an 'Info' link and a checkbox labeled 'Encrypt this volume' which is currently unchecked. A note states: 'Use Amazon EBS encryption as an encryption solution for your EBS resources associated with your EC2 instances.'
- Tags - optional:** A section with an 'Info' link and a note: 'A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter.'

The footer of the console shows 'CloudShell', 'Feedback', and copyright information: '© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences'.



3. Attach EBS Volume to the First Instance

- Navigate to Instances in the EC2 Dashboard.
- Go back to Volumes, select the volume you created, and click Actions > Attach Volume.
- Choose your instance and click Attach.



4. Mount the EBS Volume on the First Instance

- Connect to the instance via SSH.
- List the available disk devices using `lsblk` to find your new volume (Here it is `/dev/xvdf`).

- Create a filesystem on the volume

```
# sudo mkfs -t ext4 /dev/xvdf
```

```
[root@ip-172-31-40-147 ~]#  
[root@ip-172-31-40-147 ~]# lsblk  
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS  
xvda         202:0    0   8G  0 disk  
├─xvda1      202:1    0   8G  0 part /  
├─xvda127    259:0    0   1M  0 part  
└─xvda128    259:1    0  10M  0 part /boot/efi  
xvdf         202:80    0   5G  0 disk  
[root@ip-172-31-40-147 ~]#  
[root@ip-172-31-40-147 ~]#  
[root@ip-172-31-40-147 ~]# mkfs -t ext4 /dev/xvdf  
mke2fs 1.46.5 (30-Dec-2021)  
Creating filesystem with 1310720 4k blocks and 327680 inodes  
Filesystem UUID: 825e25e8-52d1-473a-b896-292f4d056176  
Superblock backups stored on blocks:  
        32768, 98304, 163840, 229376, 294912, 819200, 884736  
  
Allocating group tables: done  
Writing inode tables: done  
Creating journal (16384 blocks): done  
Writing superblocks and filesystem accounting information: done  
[root@ip-172-31-40-147 ~]#
```

- Create a mount point and mount the volume

```
# mkdir /mnt/mydata
```

```
# mount /dev/xvdf /mnt/mydata
```

- Write some data to the volume to test

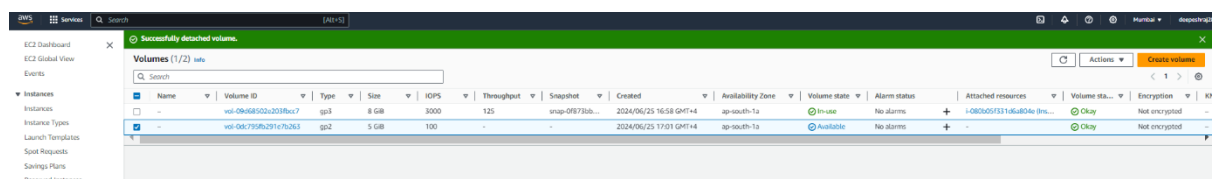
```
# echo "Hello, This is the Instance1 EBS!" | sudo tee /mnt/mydata/test.txt
```

- Unmount the volume and detach it

```
# umount /mnt/mydata
```

```
[root@ip-172-31-40-147 ~]#  
[root@ip-172-31-40-147 ~]# mkdir /mnt/mydata  
[root@ip-172-31-40-147 ~]# mount /dev/xvdf /mnt/mydata  
[root@ip-172-31-40-147 ~]#  
[root@ip-172-31-40-147 ~]#  
[root@ip-172-31-40-147 ~]#  
[root@ip-172-31-40-147 ~]# echo "Hello,This is the Instance1 EBS!" | sudo tee /mnt/mydata/test.txt  
Hello,This is the Instance1 EBS!  
[root@ip-172-31-40-147 ~]#  
[root@ip-172-31-40-147 ~]#  
[root@ip-172-31-40-147 ~]#  
[root@ip-172-31-40-147 ~]# umount /mnt/mydata  
[root@ip-172-31-40-147 ~]#
```

- Stop the instance, go to Volumes, select your volume, and click Detach Volume.



```
[root@ip-172-31-40-147 ~]# lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
xvda        202:0    0   8G  0 disk
├─xvda1     202:1    0   8G  0 part /
├─xvda127   259:0    0    1M  0 part
└─xvda128   259:1    0   10M  0 part /boot/efi
[root@ip-172-31-40-147 ~]#
```

5. Create Second AWS EC2 Instance
6. Attach and Verify EBS Volume on the Second Instance
 - a. Attach the EBS volume to the second instance following the steps in Step 3
 - b. Start the second instance and connect via SSH.
 - c. Mount the EBS volume on the second instance


```
# mkdir /mnt/mydata
```

```
# mount /dev/xvdf /mnt/mydata
```

Second EC2 Instance

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability
Instance1	i-0bf7b597fd2d45db1	Terminated	t2.micro	-	View alarms +	ap-south-1l
Instance2	i-08cc38fd3d52cbf65	Terminated	t2.micro	-	View alarms +	ap-south-1l
Instance2	i-0724bec9f84d22119	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1l
Instance 1	i-080b05f331d6a804e	Stopped	t2.micro	-	View alarms +	ap-south-1l

Attach Volume

Basic details

Volume ID
vol-0dc795fb291e7b263

Availability Zone
ap-south-1a

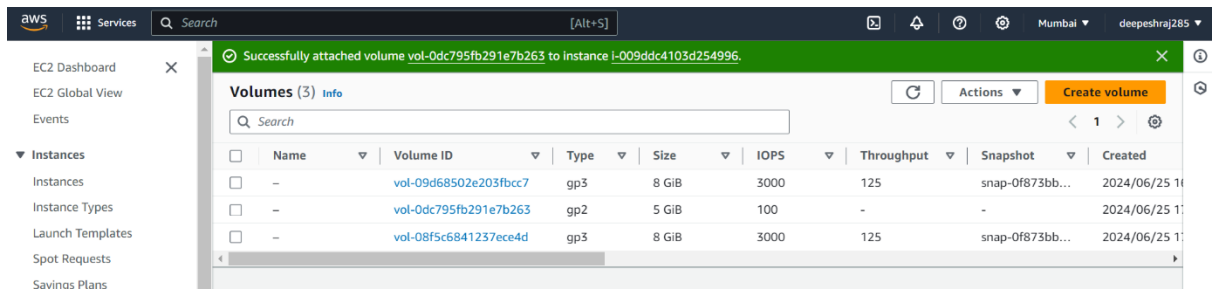
Instance
i-0724bec9f84d22119

Device name
/dev/sdf

Recommended device names for Linux: /dev/xvda for root volume, /dev/sd[f-p] for data volumes.

ⓘ Newer Linux kernels may rename your devices to /dev/xvdf through /dev/xvdp internally, even when the device name entered here (and shown in the details) is /dev/sdf through /dev/sdp.

Cancel Attach volume



Mount EBS Volume to Second Instance

```
[root@ip-172-31-34-147 ~]#
[root@ip-172-31-34-147 ~]# mkdir /mnt/mydata
[root@ip-172-31-34-147 ~]#
[root@ip-172-31-34-147 ~]#
[root@ip-172-31-34-147 ~]# lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
xvda        202:0    0   8G  0 disk
├─xvda1     202:1    0   8G  0 part /
├─xvda127   259:0    0   1M  0 part
└─xvda128   259:1    0  10M  0 part /boot/efi
xvdf        202:80    0   5G  0 disk
[root@ip-172-31-34-147 ~]#
[root@ip-172-31-34-147 ~]#
[root@ip-172-31-34-147 ~]#
[root@ip-172-31-34-147 ~]#
[root@ip-172-31-34-147 ~]# mount /dev/xvdf /mnt/mydata
[root@ip-172-31-34-147 ~]#
[root@ip-172-31-34-147 ~]#
```

d. Verify the data

cat /mnt/mydata/test.txt

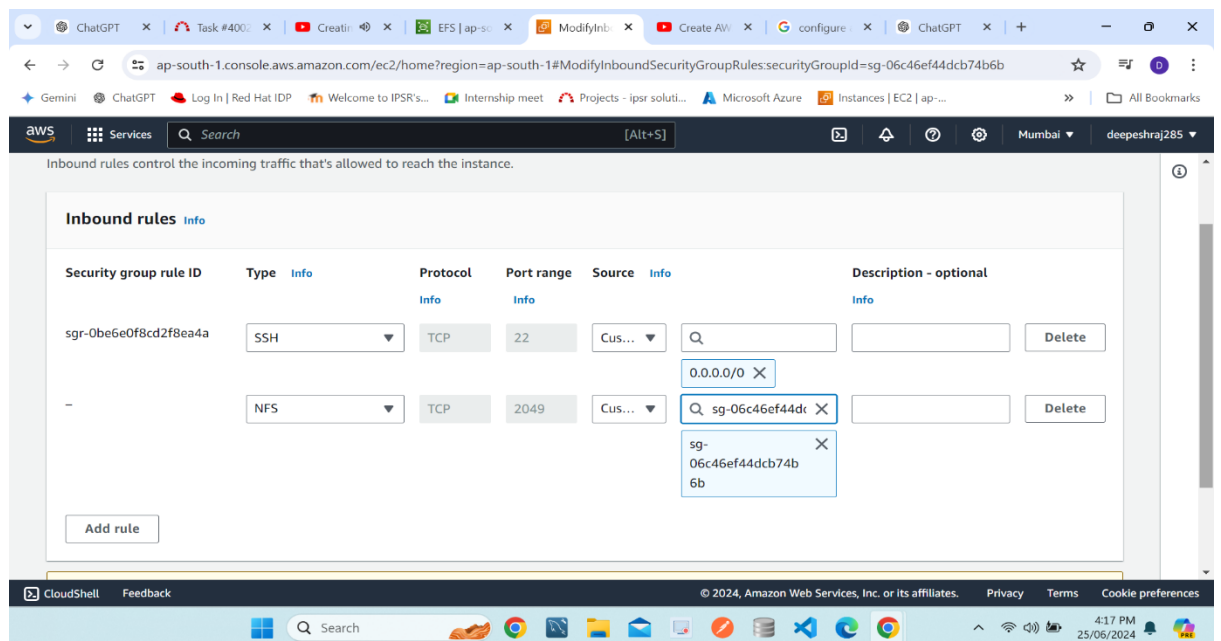
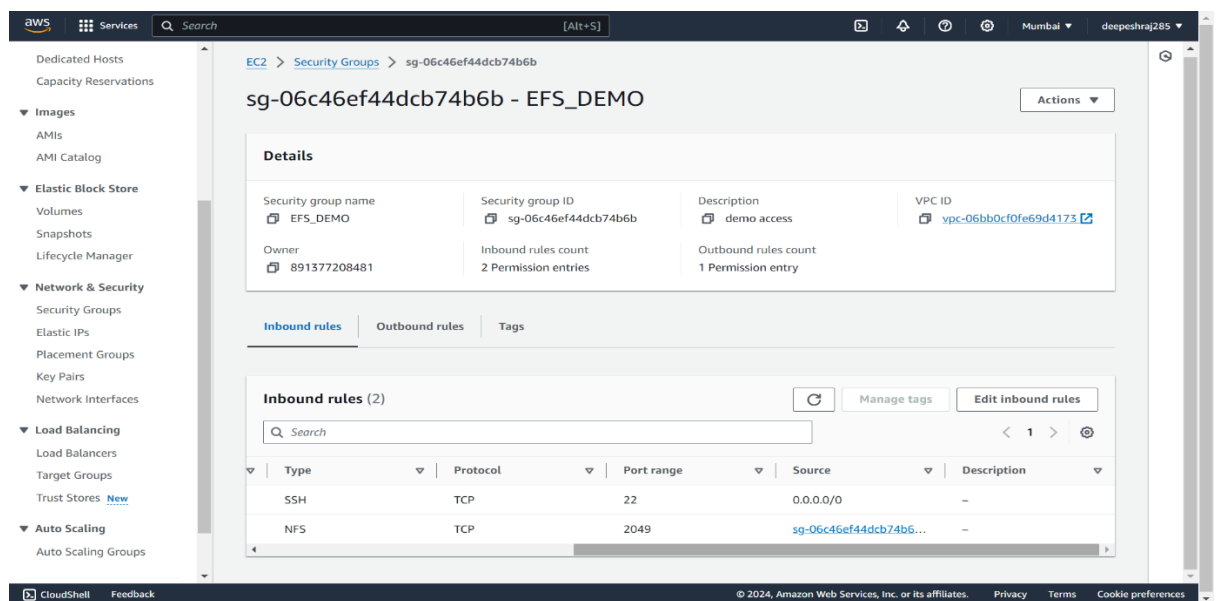
```
[root@ip-172-31-34-147 ~]#
[root@ip-172-31-34-147 ~]#
[root@ip-172-31-34-147 ~]#
[root@ip-172-31-34-147 ~]# df -Th
Filesystem      Type      Size  Used Avail Use% Mounted on
devtmpfs        devtmpfs  4.0M   0    4.0M   0% /dev
tmpfs           tmpfs     475M   0    475M   0% /dev/shm
tmpfs           tmpfs     190M 444K   190M   1% /run
/dev/xvda1      xfs       8.0G  1.6G   6.5G  20% /
tmpfs           tmpfs     475M   0    475M   0% /tmp
/dev/xvda128    vfat      10M   1.3M   8.7M  13% /boot/efi
tmpfs           tmpfs     95M   0     95M   0% /run/user/1000
/dev/xvdf       ext4      4.9G  28K   4.6G   1% /mnt/mydata
[root@ip-172-31-34-147 ~]#
[root@ip-172-31-34-147 ~]#
[root@ip-172-31-34-147 ~]#
[root@ip-172-31-34-147 ~]#
[root@ip-172-31-34-147 ~]#
[root@ip-172-31-34-147 ~]# cd /mnt/mydata
[root@ip-172-31-34-147 mydata]# ls
lost+found test.txt
[root@ip-172-31-34-147 mydata]# cat test.txt
Hello, This is the Instance1 EBS!
[root@ip-172-31-34-147 mydata]#
[root@ip-172-31-34-147 mydata]#
[root@ip-172-31-34-147 mydata]#
```

EFS

1. Create a Security Group for EFS

- a. Navigate to VPC Dashboard and click on Security Groups.
- b. Click Create Security Group and configure:
 - i. Name: EFS-DEMO
 - ii. Description: Security group for EFS
 - iii. VPC: Choose the VPC your instances are in.
- c. Add inbound rules

Type: NFS, Protocol: TCP, Port Range: 2049, Source: Custom, add the security groups of your EC2 instances.



2. Create an EFS File System

- a. Navigate to EFS Dashboard.
- b. Click Create file system.
- c. Configure the file system
 - VPC: Choose your VPC.
 - Security Groups: Select the EFS-DEMO security group.
- d. Click Next and configure the mount targets.
- e. Click Next and review settings, then click Create.

The screenshot displays the AWS EFS File System settings page, divided into two main sections: General and Network.

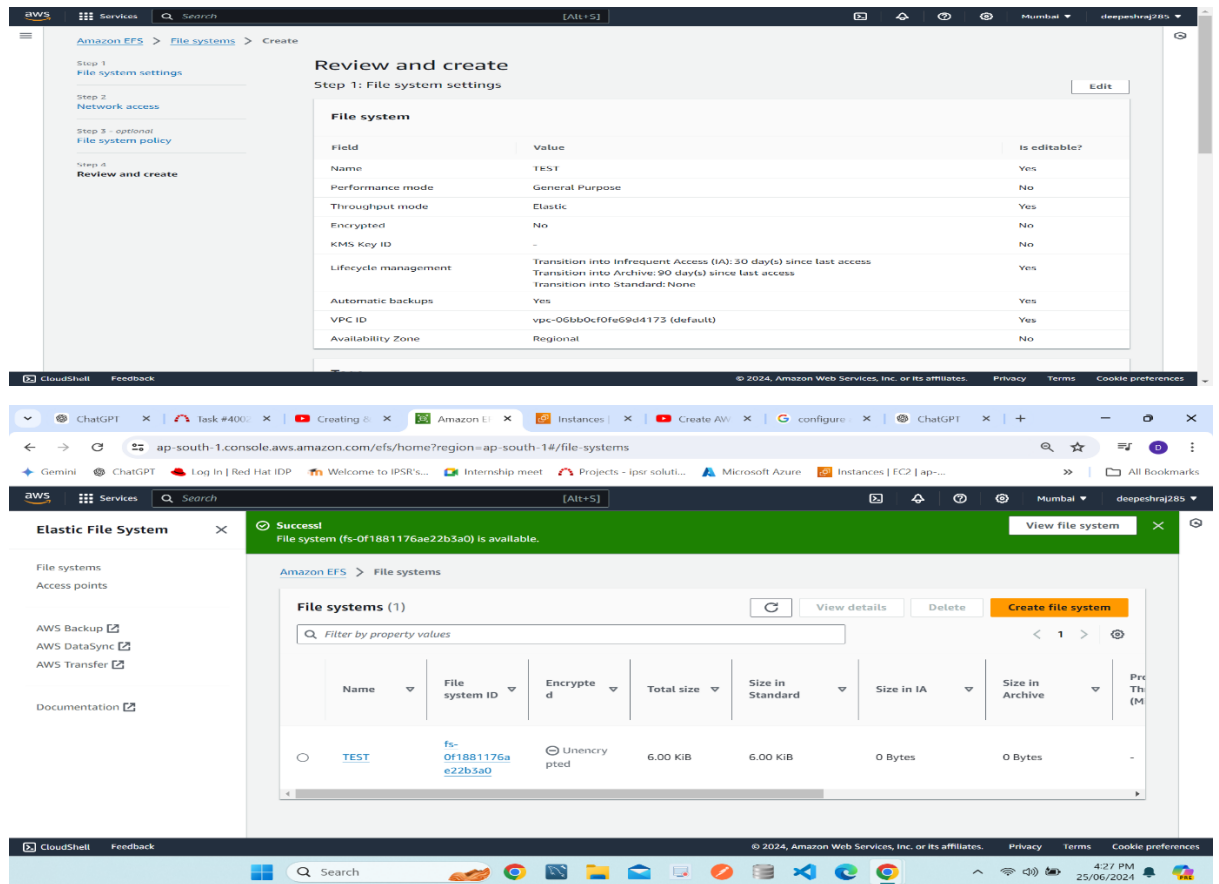
General Tab:

- Name - optional:** A text input field containing "TEST".
- File system type:** Two radio buttons are present. "Regional" is selected, offering the highest levels of availability and durability by storing file system data across multiple Availability Zones within an AWS Region. "One Zone" provides continuous availability to data within a single Availability Zone within an AWS Region.
- Automatic backups:** A checkbox labeled "Enable automatic backups" is checked.
- Lifecycle management:** Three dropdown menus are shown for transitioning files into different storage classes based on time since last access: "Transition into Infrequent Access (IA)" is set to "30 day(s) since last access", "Transition into Archive" is set to "90 day(s) since last access", and "Transition into Standard" is set to "None".
- Encryption:** This section is partially visible at the bottom.

Network Tab:

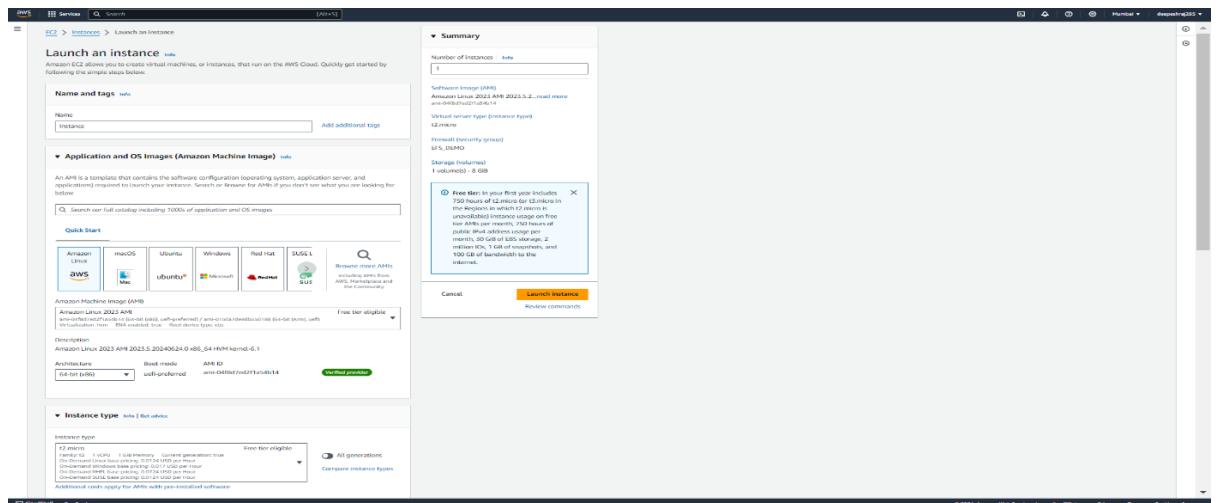
- Virtual Private Cloud (VPC):** A dropdown menu shows "vpc-06bb0cf0fe69d4173" with "default" as the selected option.
- Mount targets:** A table lists three mount targets, each with its own set of configuration options:

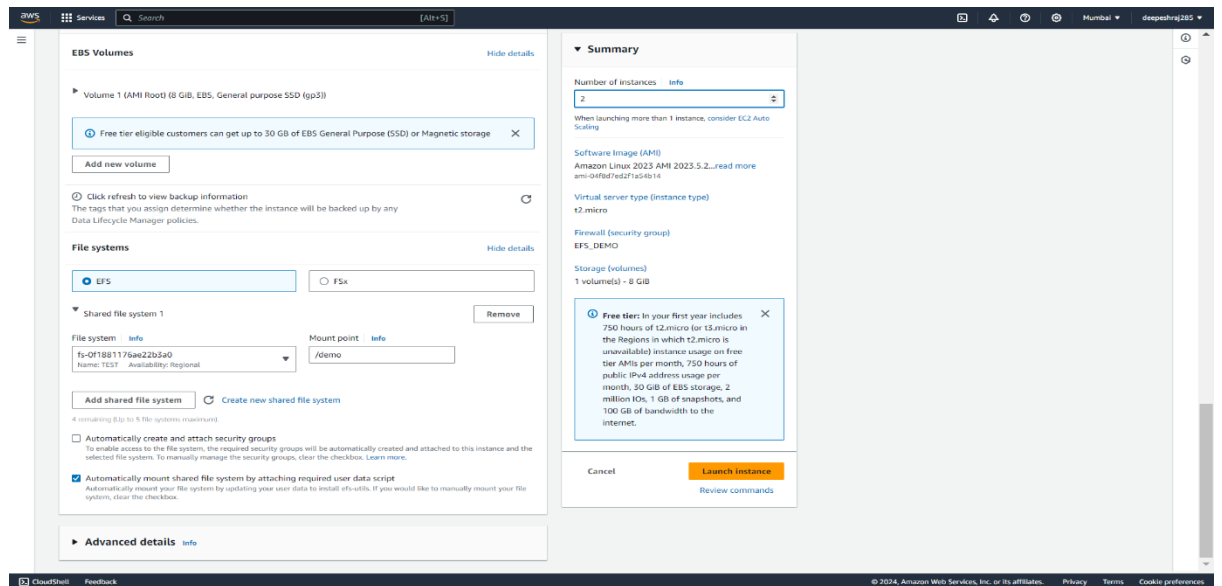
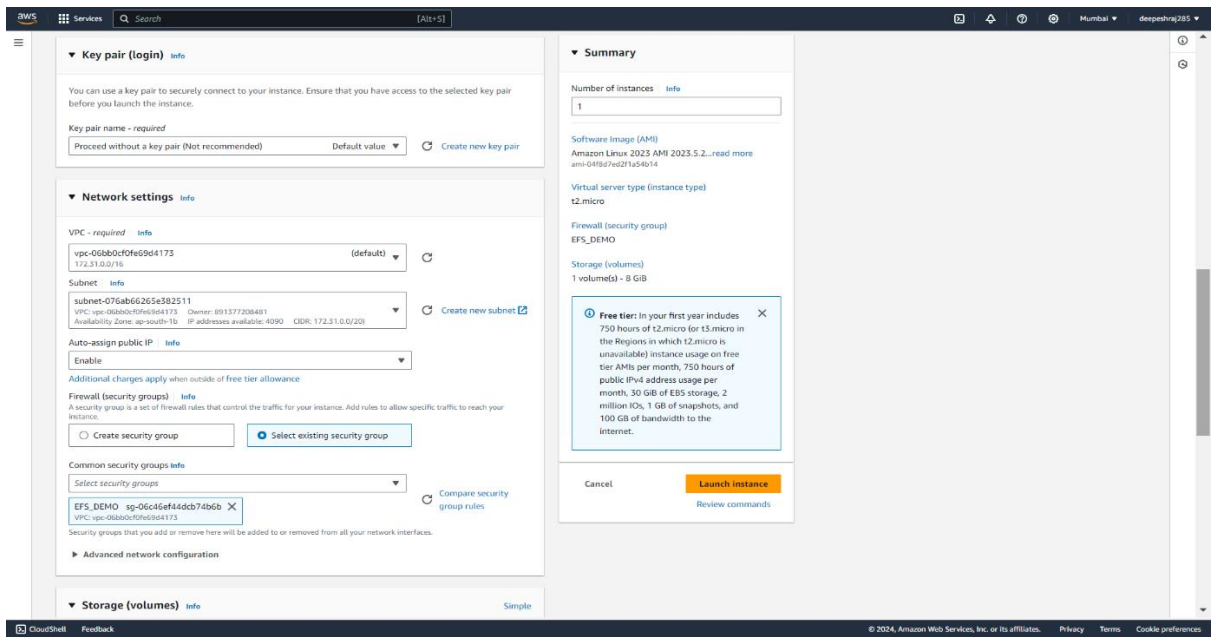
Availability zone	Subnet ID	IP address	Security groups	Remove
ap-south-1a	subnet-0d07e4...	Automatic	sg-06c46ef44dcb74b6b EFS_DEMO	Remove
ap-south-1b	subnet-076ab6...	Automatic	sg-06c46ef44dcb74b6b EFS_DEMO	Remove
ap-south-1c	subnet-0ae0c0...	Automatic	sg-06c46ef44dcb74b6b EFS_DEMO	Remove



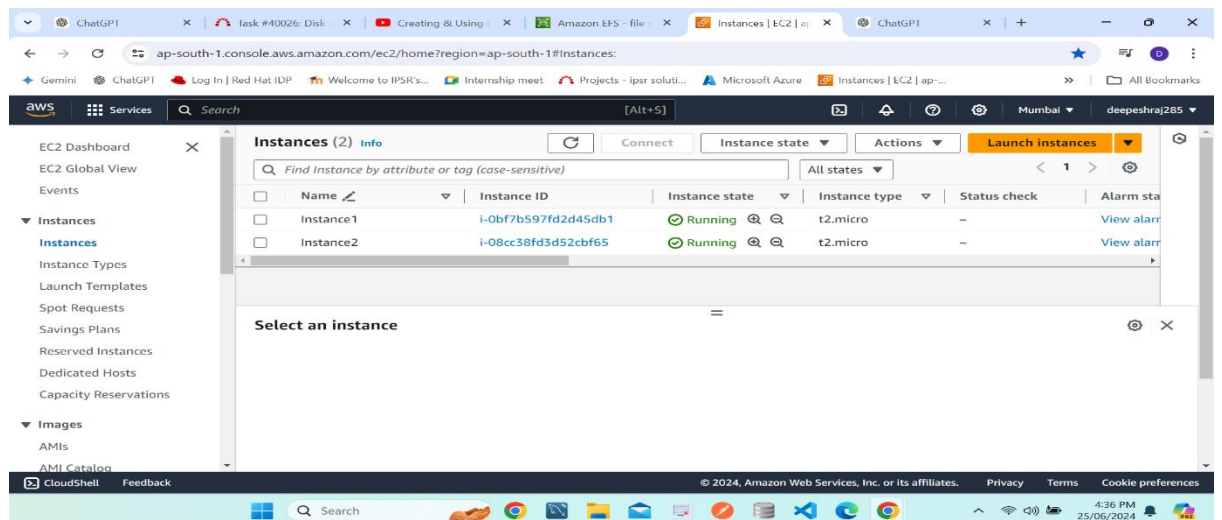
3. Create Two EC2 Instance

- Name the Ec2 Instance
- Choose the key pair
- Provide the network settings here choose the security group that we created first
- In EBS volume choose the file system EFS and choose the file system id we created already
- Type the mount point of the instance
- Choose two ec2 instances and launch it.

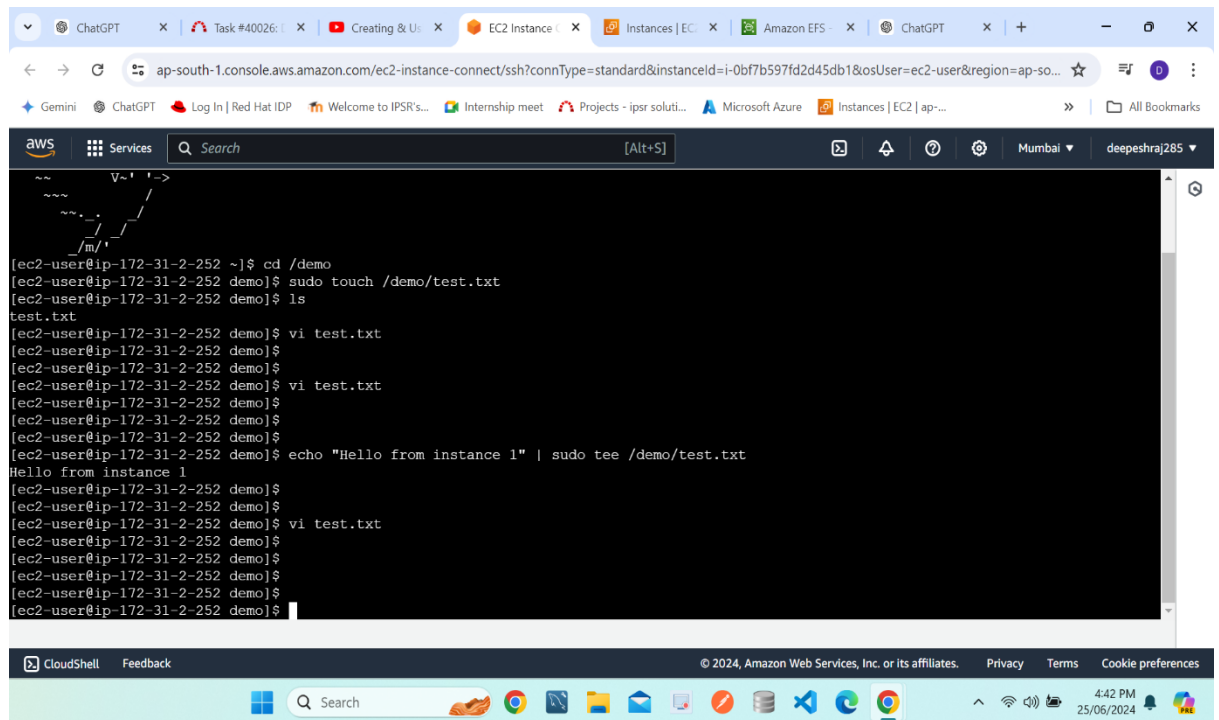




Lanunch the EC2 Instances



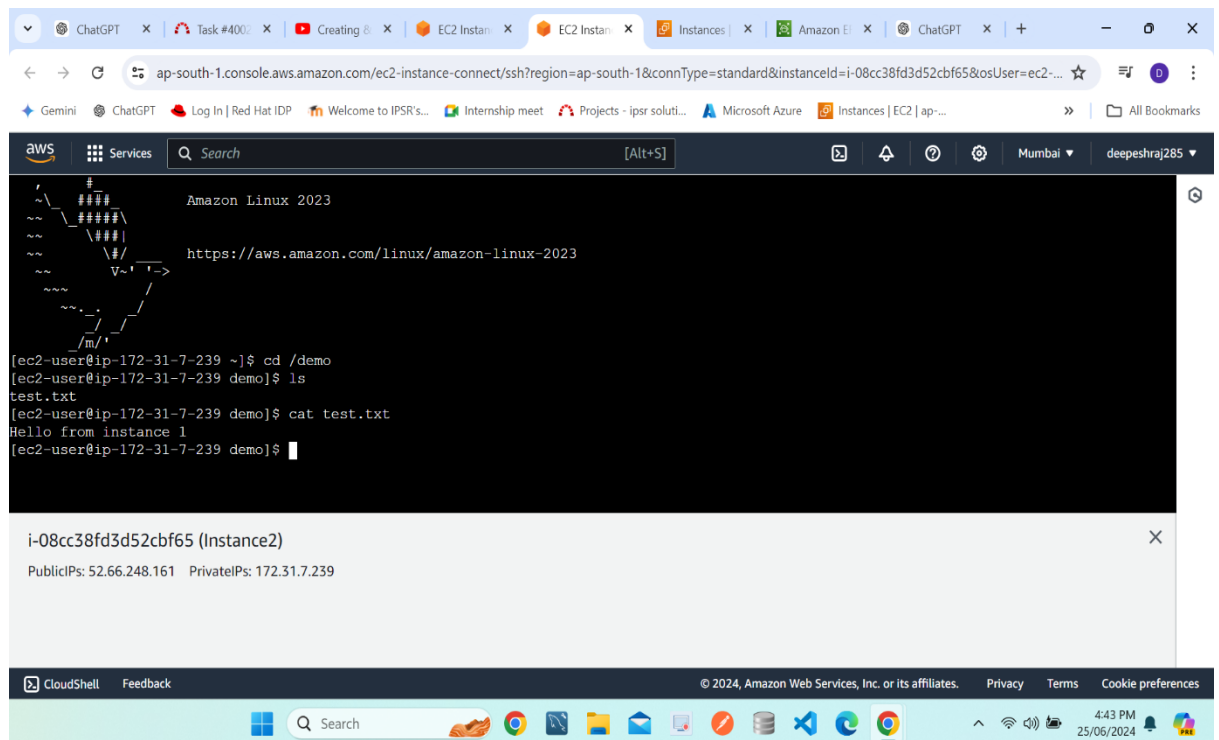
4. Connect the first instance and create a file in the mount point and enter data to the file



The screenshot shows the AWS CloudShell interface with a terminal window connected to EC2 instance i-0bf7b597fd2d45db1. The terminal output is as follows:

```
[ec2-user@ip-172-31-2-252 ~]$ cd /demo
[ec2-user@ip-172-31-2-252 demo]$ sudo touch /demo/test.txt
[ec2-user@ip-172-31-2-252 demo]$ ls
test.txt
[ec2-user@ip-172-31-2-252 demo]$ vi test.txt
[ec2-user@ip-172-31-2-252 demo]$
[ec2-user@ip-172-31-2-252 demo]$ vi test.txt
[ec2-user@ip-172-31-2-252 demo]$
[ec2-user@ip-172-31-2-252 demo]$ echo "Hello from instance 1" | sudo tee /demo/test.txt
Hello from instance 1
[ec2-user@ip-172-31-2-252 demo]$
[ec2-user@ip-172-31-2-252 demo]$ vi test.txt
[ec2-user@ip-172-31-2-252 demo]$
[ec2-user@ip-172-31-2-252 demo]$
[ec2-user@ip-172-31-2-252 demo]$
[ec2-user@ip-172-31-2-252 demo]$
```

5. Connect the second instance and check the file which is in the mount point



The screenshot shows the AWS CloudShell interface with a terminal window connected to EC2 instance i-08cc38fd3d52cbf65. The terminal output is as follows:

```
Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023
[ec2-user@ip-172-31-7-239 ~]$ cd /demo
[ec2-user@ip-172-31-7-239 demo]$ ls
test.txt
[ec2-user@ip-172-31-7-239 demo]$ cat test.txt
Hello from instance 1
[ec2-user@ip-172-31-7-239 demo]$
```

Below the terminal window, a box displays instance details for i-08cc38fd3d52cbf65 (Instance2):

i-08cc38fd3d52cbf65 (Instance2)
PublicIPs: 52.66.248.161 PrivateIPs: 172.31.7.239