Assignment: EC2 Instance Creation

Name: Uday Kiran Ailapaka

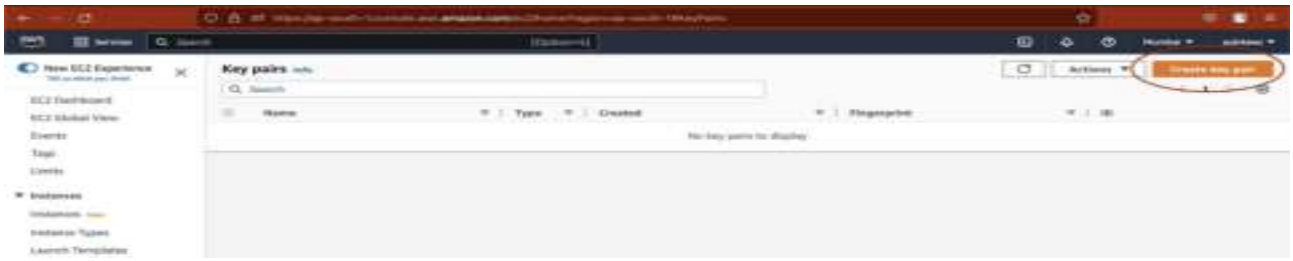
AWS Region: Mumbai (ap-south-1)

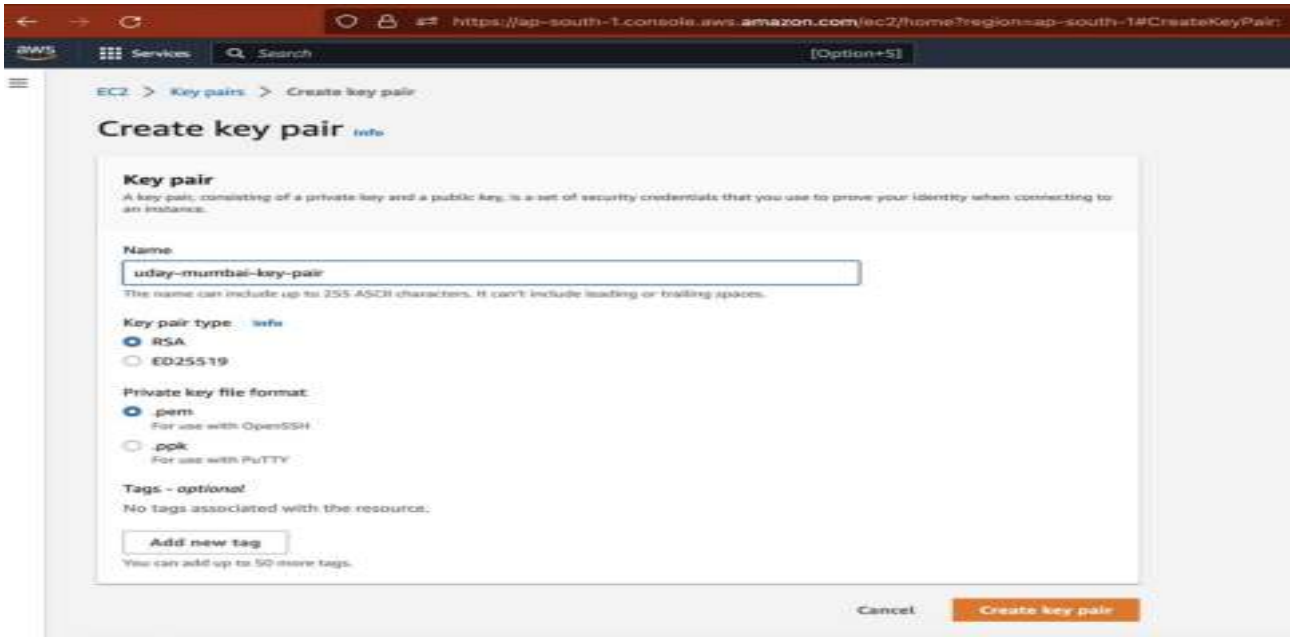
Creation of a key pair:

A key pair is required to connect to the EC2 instances created. We have to associate a key pair while creating an EC2 instance.

Navigation: EC2 -> Network & Security -> Key Pairs

Key pair name: **uday-mumbai-key-pair.pem**

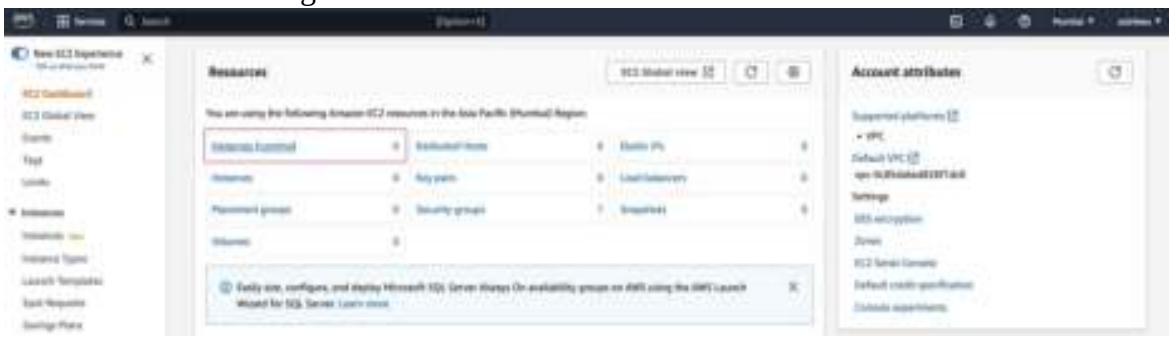




Note:- Then a new key pair is downloaded '**uday-mumbai-key-pair.pem**'

Creation of a EC2 Instance:

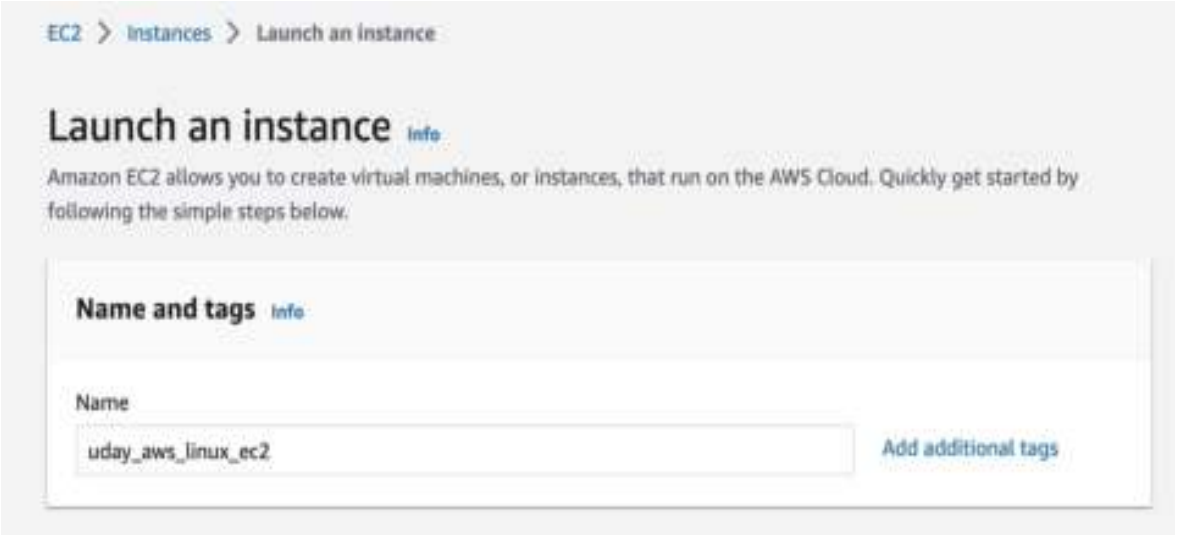
We are going to create an Amazon Linux AWS EC2 instance.

1. Login to AWS console.
2. Navigate to Services -> Compute -> EC2
3. Select Instances running
4. 

Click Launch instances

5. 

Name the EC2 instance: uday_aws_linux_ec2

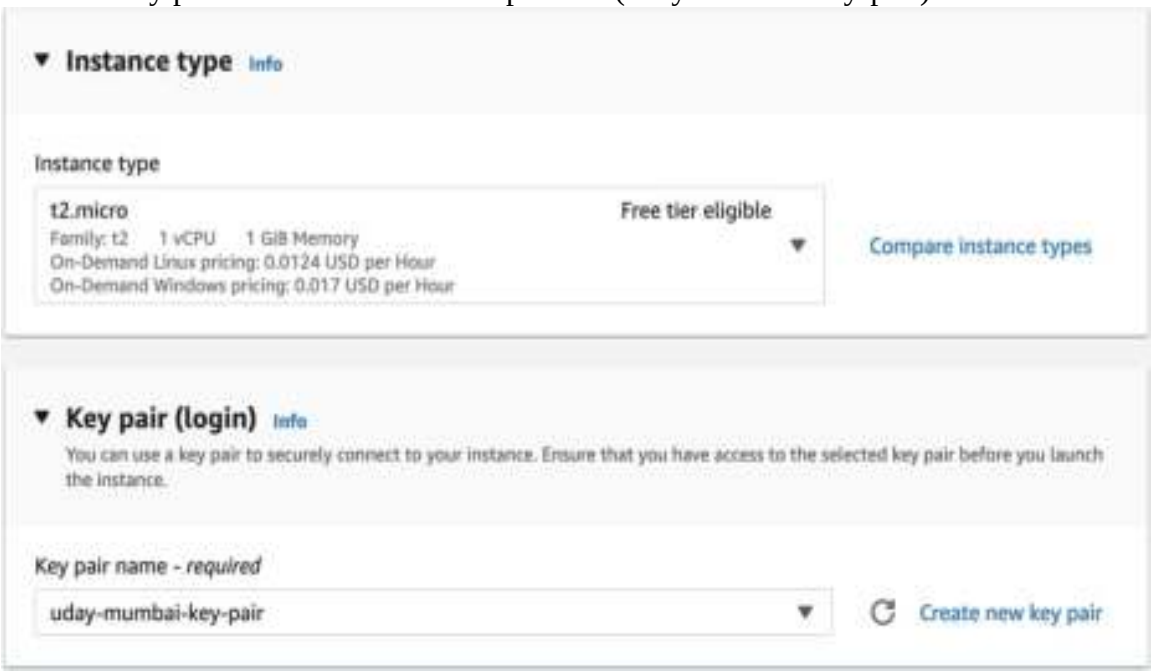
6. 

Select OS image and the version: (Amazon Linux 2 Kernel 5.10 AMI)

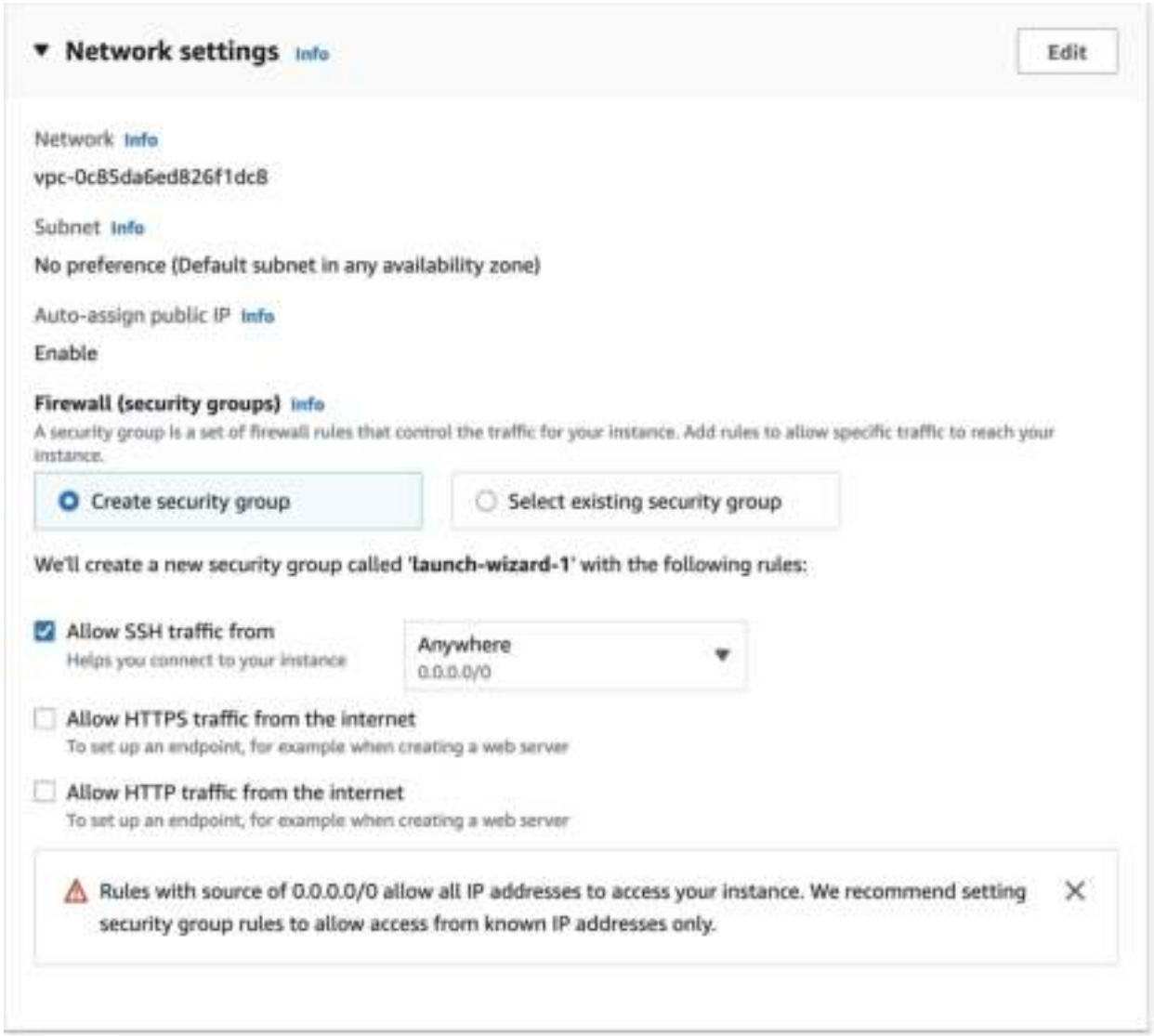
7. 

Select Instance type as t2.micro (free tier eligible):

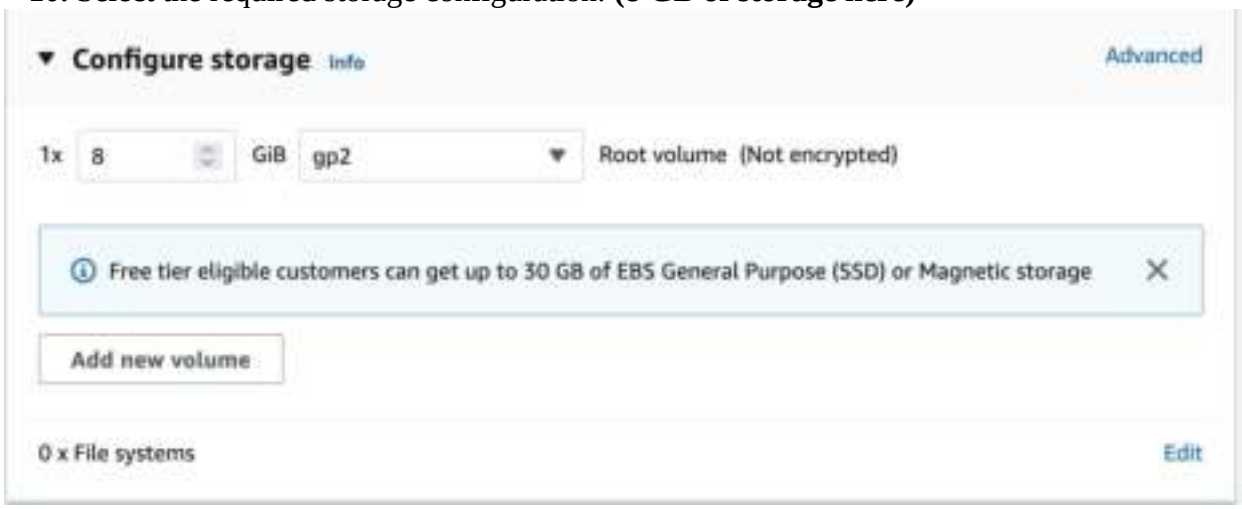
8. Select the key pair created in the first step above (uday-mumbai-key-pair)

9. Use 

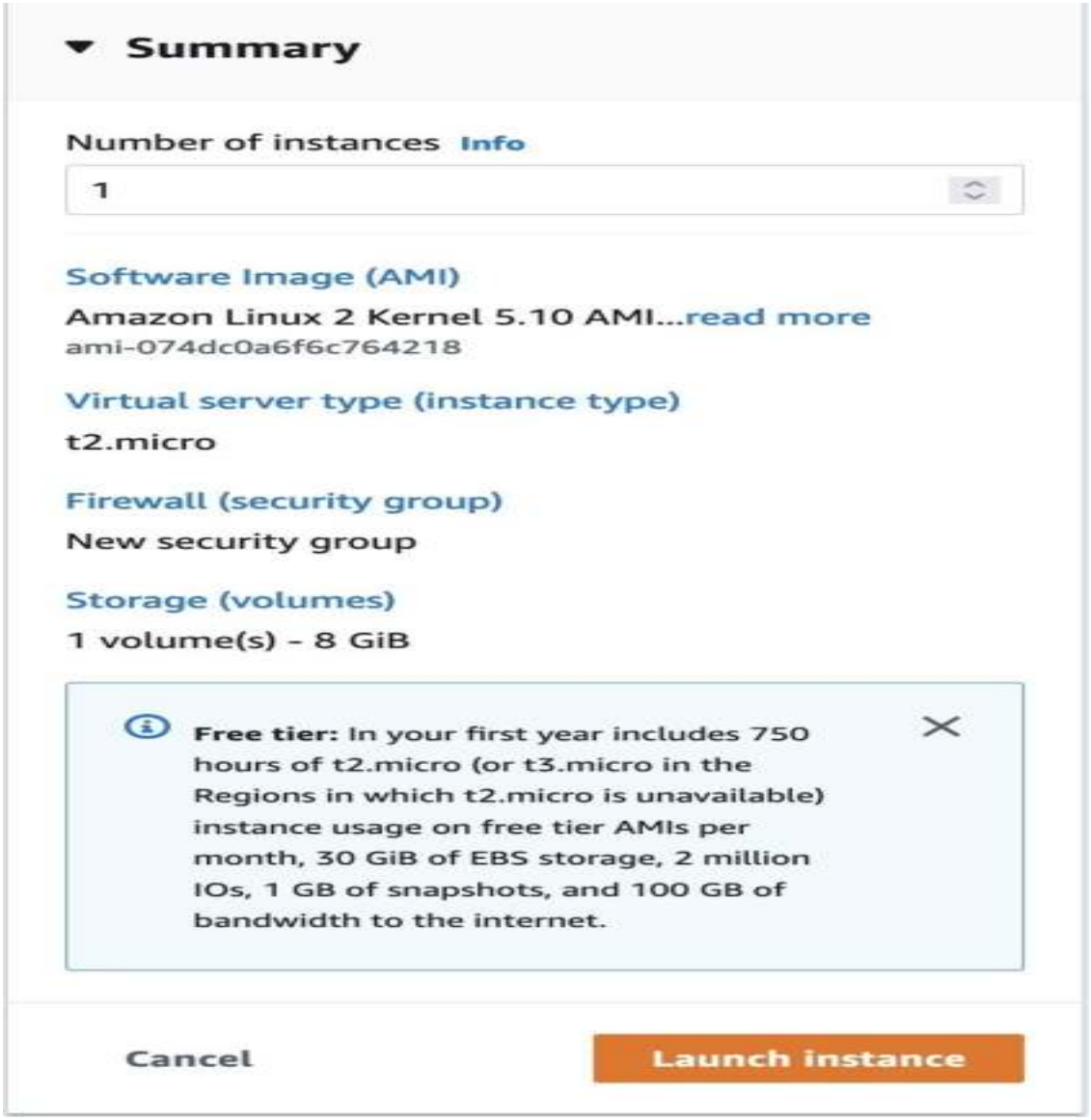
the existing default settings for **Network Settings:**

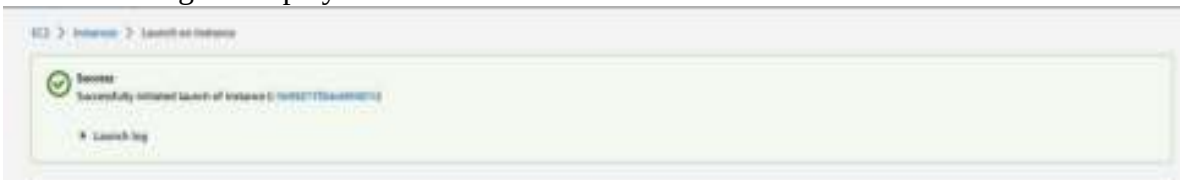


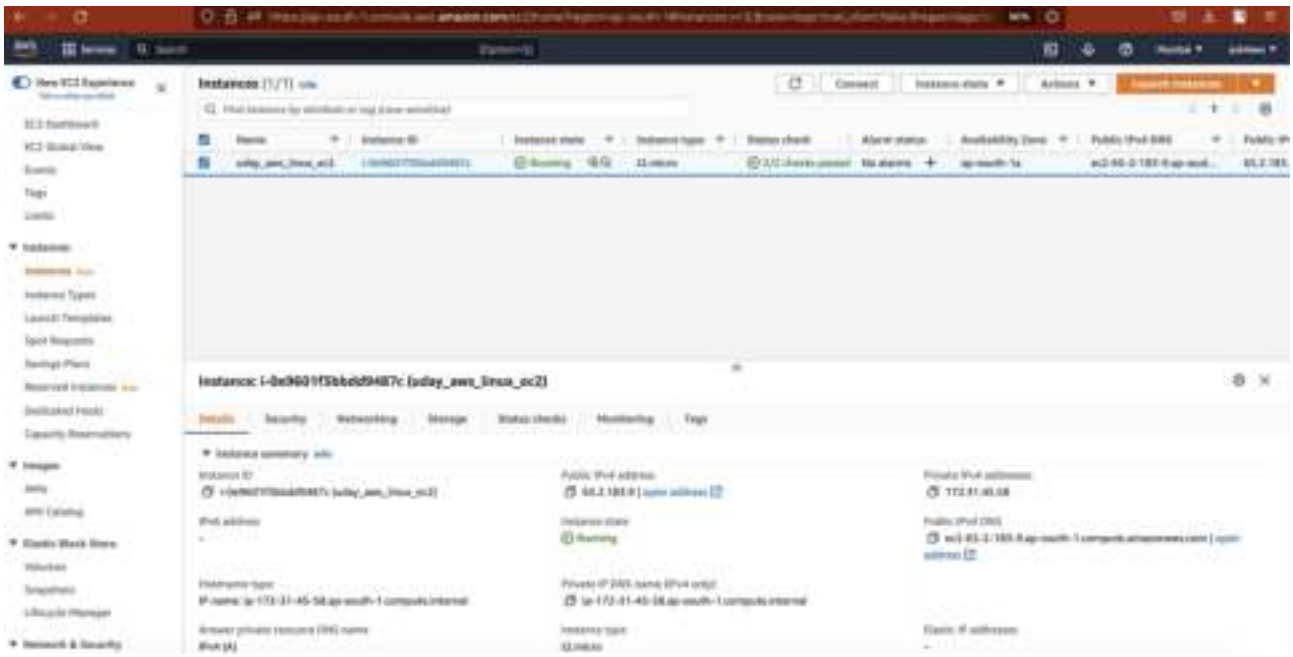
10. Select the required storage configuration: **(8 GB of storage here)**



11. Select number of instances and click Launch Instances: (1 here)



12. Success message is displayed:
13. Go  to EC2 Dashboard to see the status of the newly created EC2 instance:



Connecting to EC2 Instances:

1. Click on the instance id of the EC2 instance created above.
2. Click on SSH client section and follow the instructions given.
3. Copy the ssh command given under example. ***‘ssh -i "uday-mumbai-key-pair.pem" ec2-user@ec2-65-2-183-9.ap-south-1.compute.amazonaws.com’***
4. In MacOSX, launch terminal and cd to the location where the pem file is downloaded. Now run the copied command.

Connect to instance Info

Connect to your instance i-0e9601f3bbdd9487c (uday_aws_linux_ec2) using any of these options

EC2 Instance Connect | Session Manager | SSH client | EC2 serial console

Instance ID

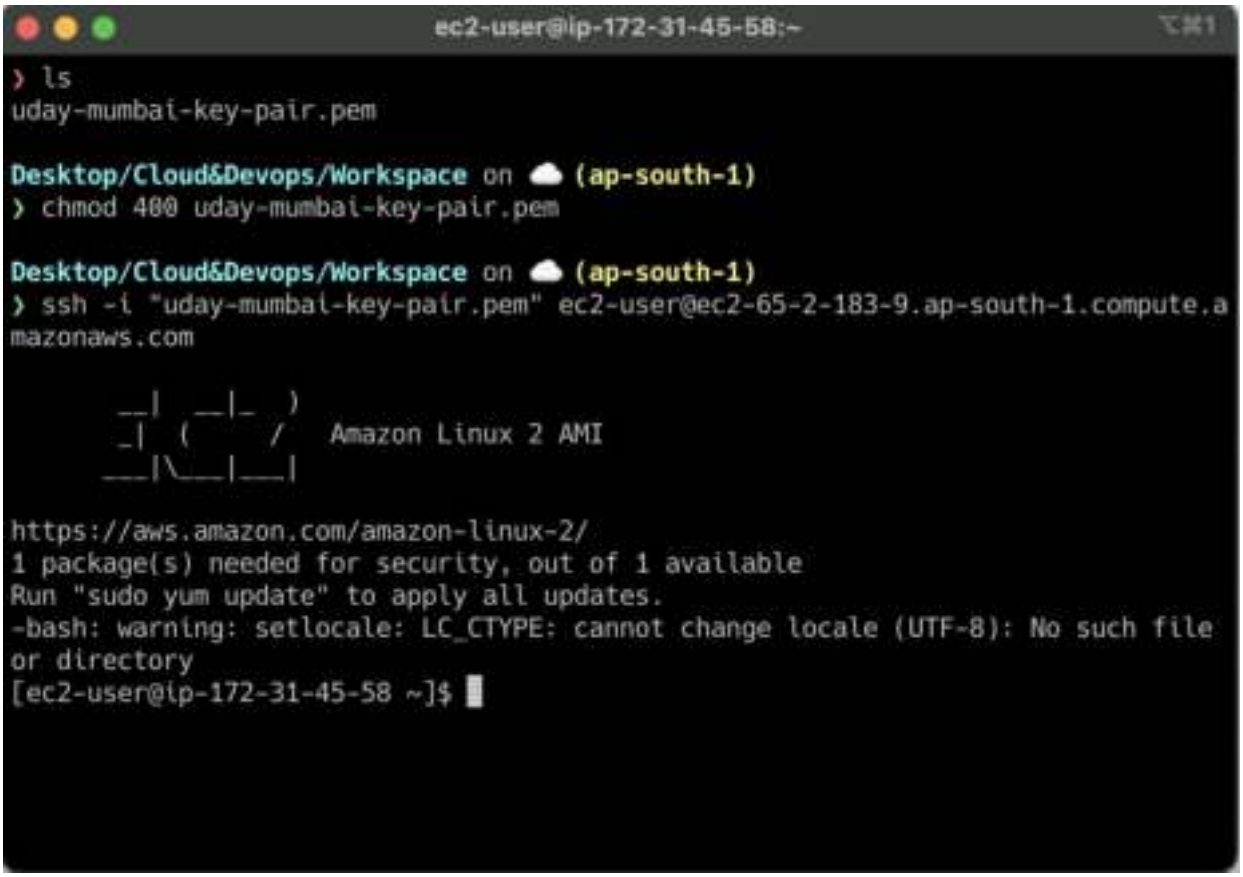
i-0e9601f3bbdd9487c (uday_aws_linux_ec2)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is uday-mumbai-key-pair.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
   chmod 400 uday-mumbai-key-pair.pem
4. Connect to your instance using its Public DNS:
   ec2-65-2-183-9.ap-south-1.compute.amazonaws.com

Example:

ssh -i "uday-mumbai-key-pair.pem" ec2-user@ec2-65-2-183-9.ap-south-1.compute.amazonaws.com

Note: In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.



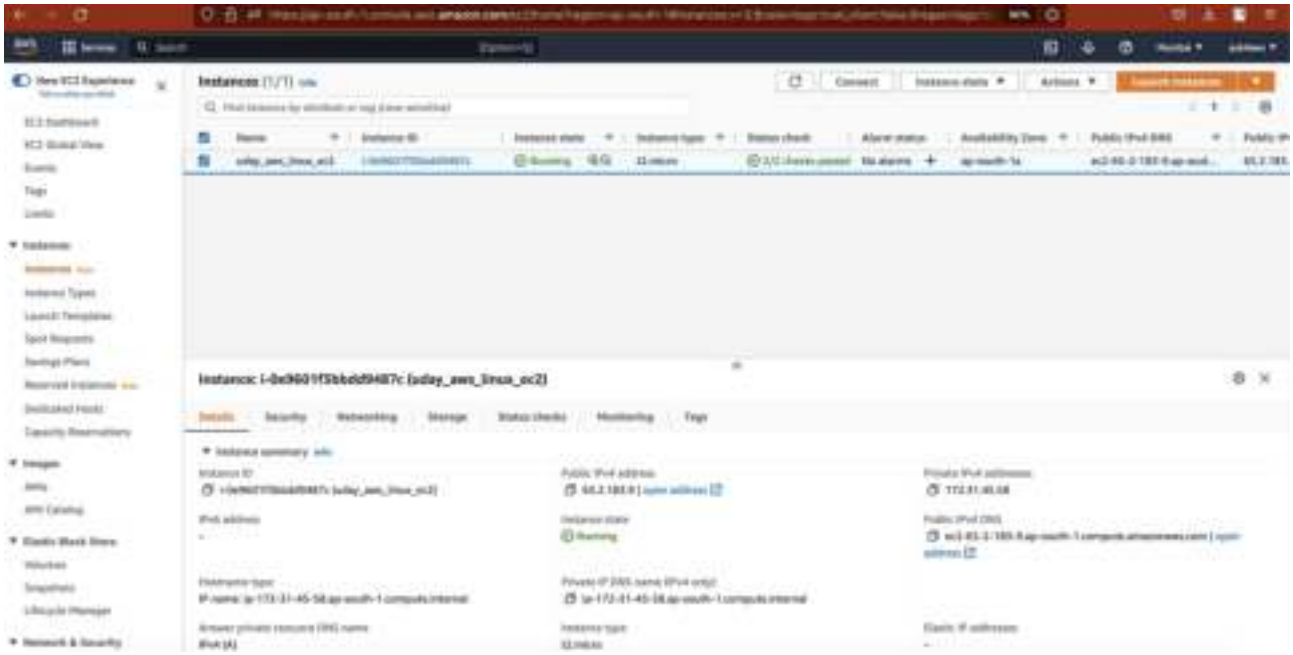
The EC2 Instance is successfully created and connect from MacOSX terminal using ssh

Create Elastic IP and Associate with EC2 instance:

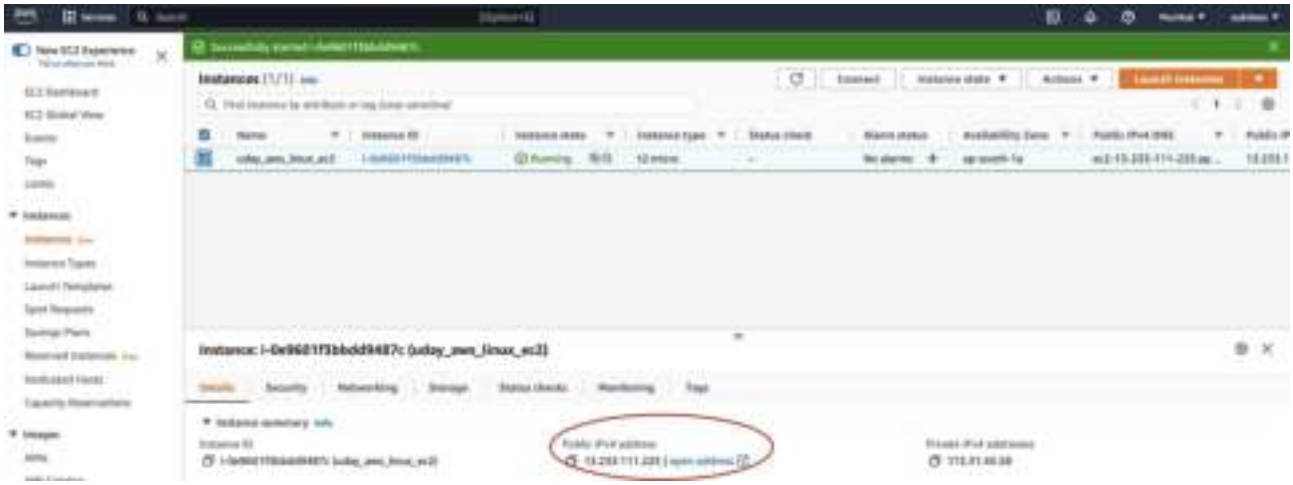
An EC2 instance has public and private ipv4 addressess. Private IP we use to connect within internal AWS services like Connect b/w multiple EC2 instances. Public IP we use to expose the system to Public connections. But, public address is dynamic i.e it changes when instance is restarted. So, we cannot use it to host a website. For that we can have a static IP address and

associate it with an EC2 instance. This static IP can be used to host a website with the DNS. This static IP is called as Elastic IP in AWS.

1. Public IP before restarting the EC2 instance: (65.2.183.9)



2. Public IP changed after restarting the same EC2 instance: (13.253.111.225)

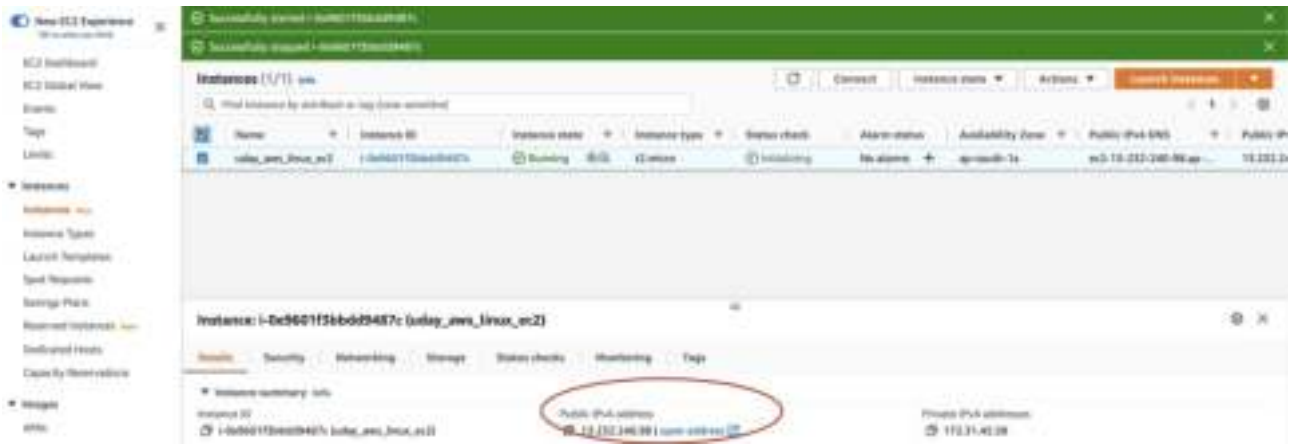


3. Navigate to **EC2 -> Network & Security -> Elastic IPs -> Allocate Elastic IP address** and click allocate

4. Select the created Elastic IP address, expand the actions and select ‘Associate Elastic IP address’

5. Choose the EC2 instance and click Associate

6. The newly created static IP address is associated with the EC2 instance even after restarting.

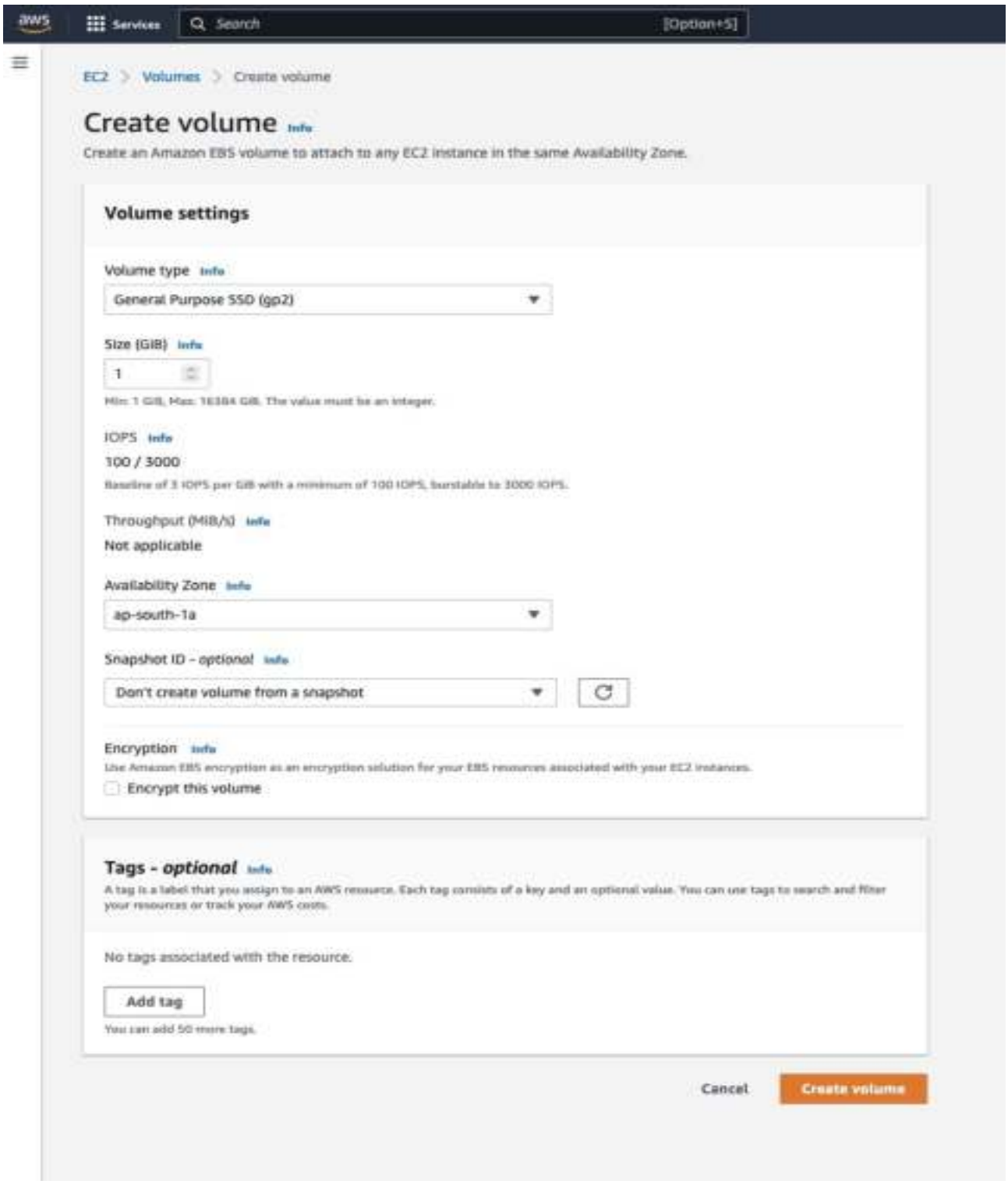


Create Elastic Block Store:

Used to extend the storage attached with EC2 instances without the need for downtime. You can mount this persistent storage with any EC2 instance.

1. Navigate to **EC2 -> Elastic Block Store -> Volumes** and click ‘Create volume’

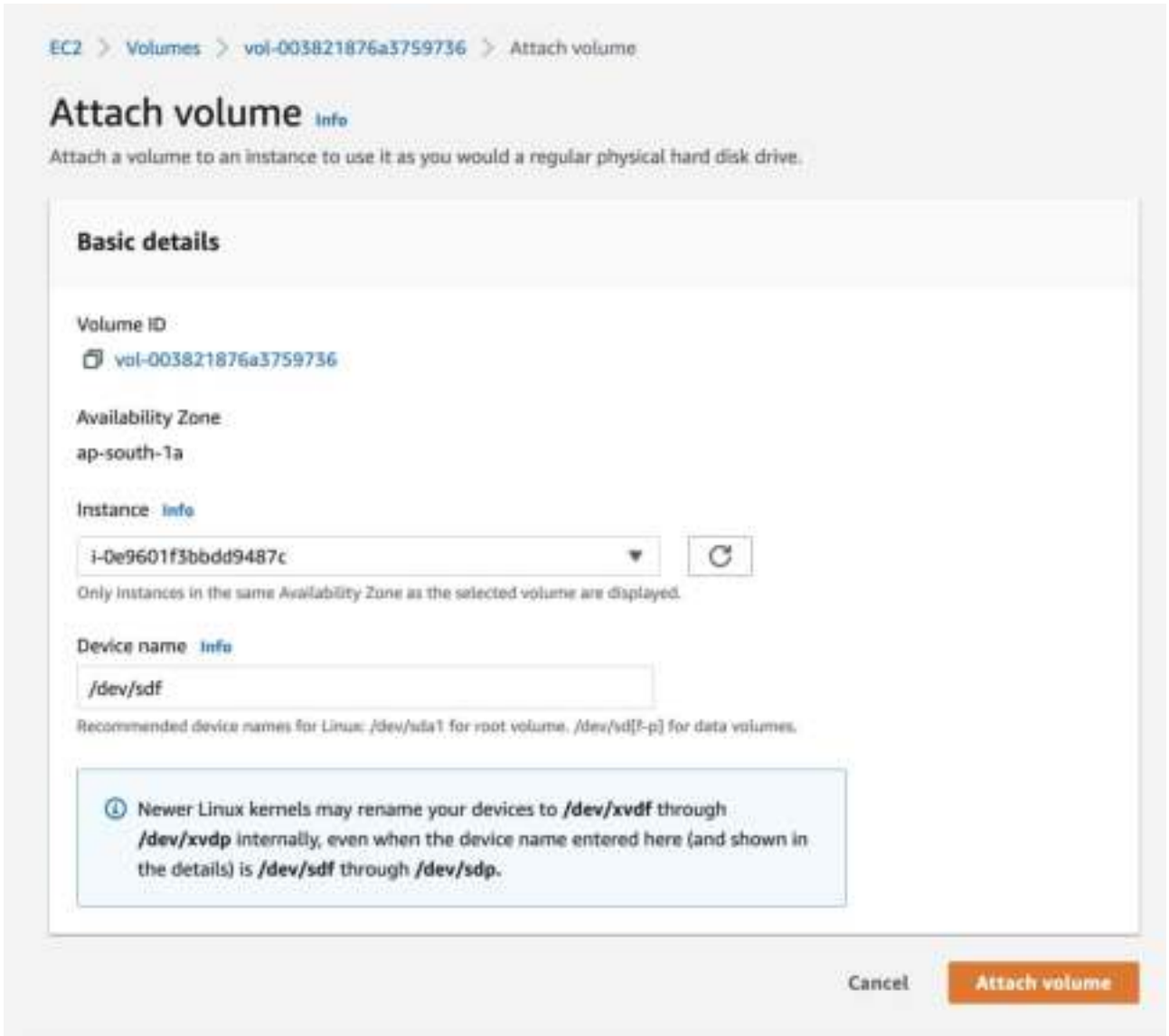
2. Select the size and create volume



4. The newly created EBS volume state is ‘Available’

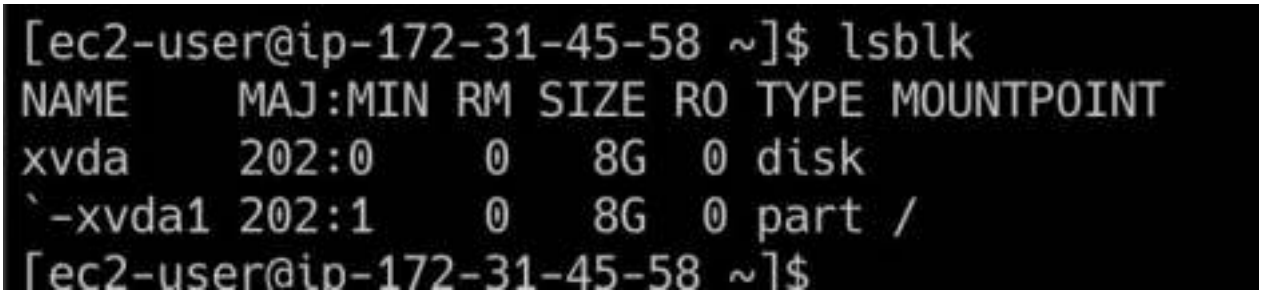
5. Select the newly create EBS volume and choose ‘Attach Volume’

6. Choose the required instance and click attach volume

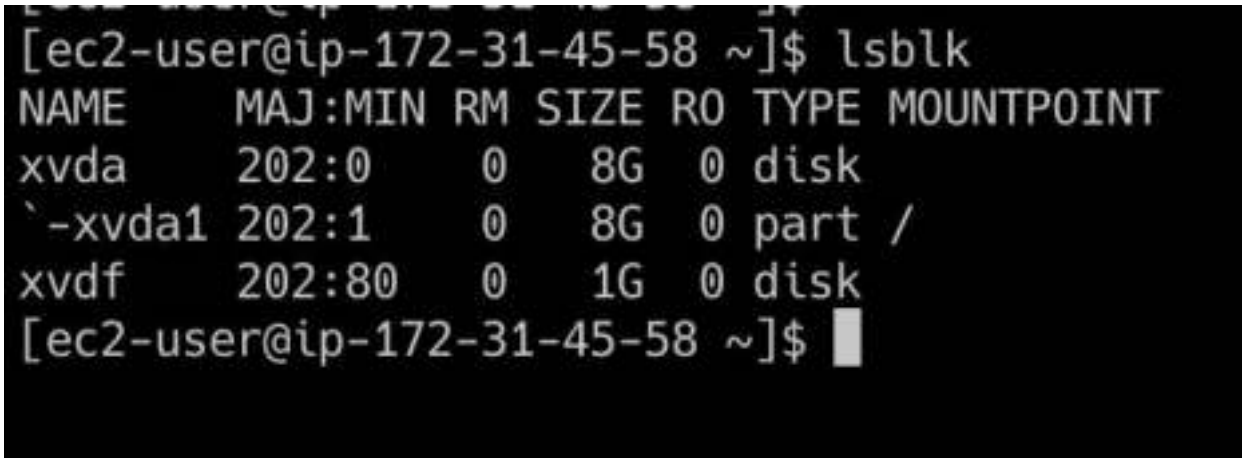


7. The volume state of EBS store is now changed to ‘In Use’

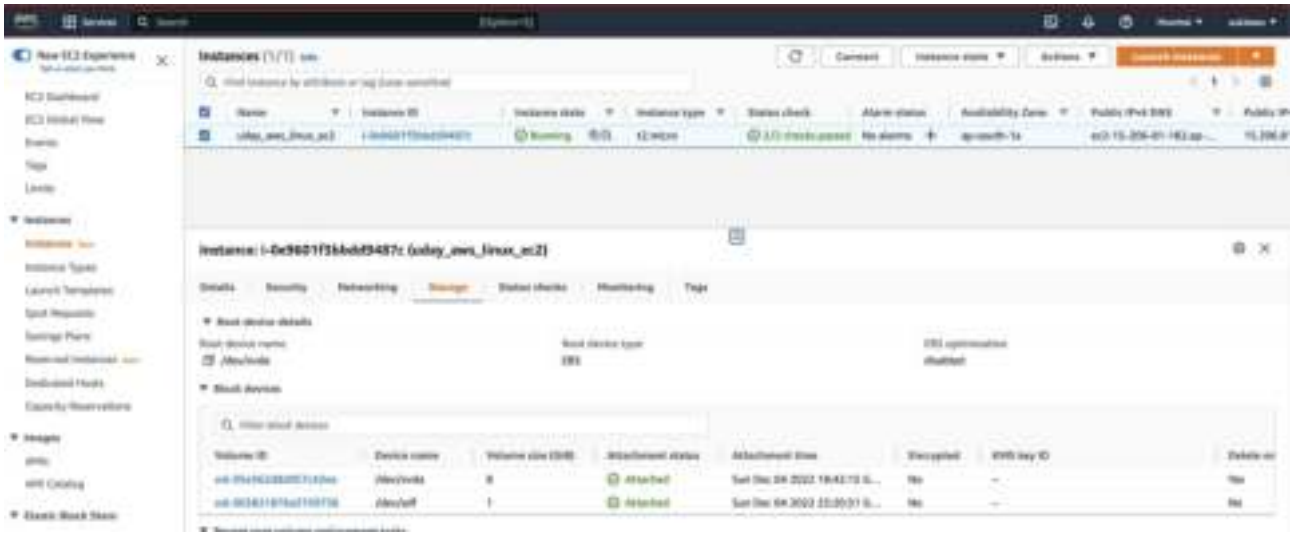
8. The EC2 disk allocation before new EBS is attached



9. The EC2 disk allocation after new EBS is attached



10. The newly created EBS is seen attached to the EC2 instance



11. Connect to EC2 instance and Mount the new EBS:

(a) sudo su

(b) lsblk

```
[ec2-user@ip-172-31-45-58 ~]$ lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0   8G  0 disk
`-xvda1 202:1    0   8G  0 part /
xvdf    202:80   0   1G  0 disk
```

(c) mkdir storage

(d) fdisk -l

```
[root@ip-172-31-45-58 ec2-user]# fdisk -l
Disk /dev/xvda: 8 GiB, 8589934592 bytes, 16777216 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: DA90AB95-8C11-4BAE-9157-C48213FBAD0C

Device       Start      End  Sectors Size Type
/dev/xvda1    4096 16777182 16773087   8G Linux filesystem
/dev/xvda128  2048     4095     2048   1M BIOS boot

Partition table entries are not in disk order.


Disk /dev/xvdf: 1 GiB, 1073741824 bytes, 2097152 sectors
```

(e) mkfs -t xfs /dev/xvdf (makes file system)

(f) mount -t xfs /dev/xvdf /home/ec2-user/storage/

(g) lsblk

```
[root@ip-172-31-45-58 ec2-user]# lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0   8G  0 disk
`-xvda1 202:1    0   8G  0 part /
xvdf    202:80   0   1G  0 disk /home/ec2-user/storage
```

12. Create somefiles in the filesystem

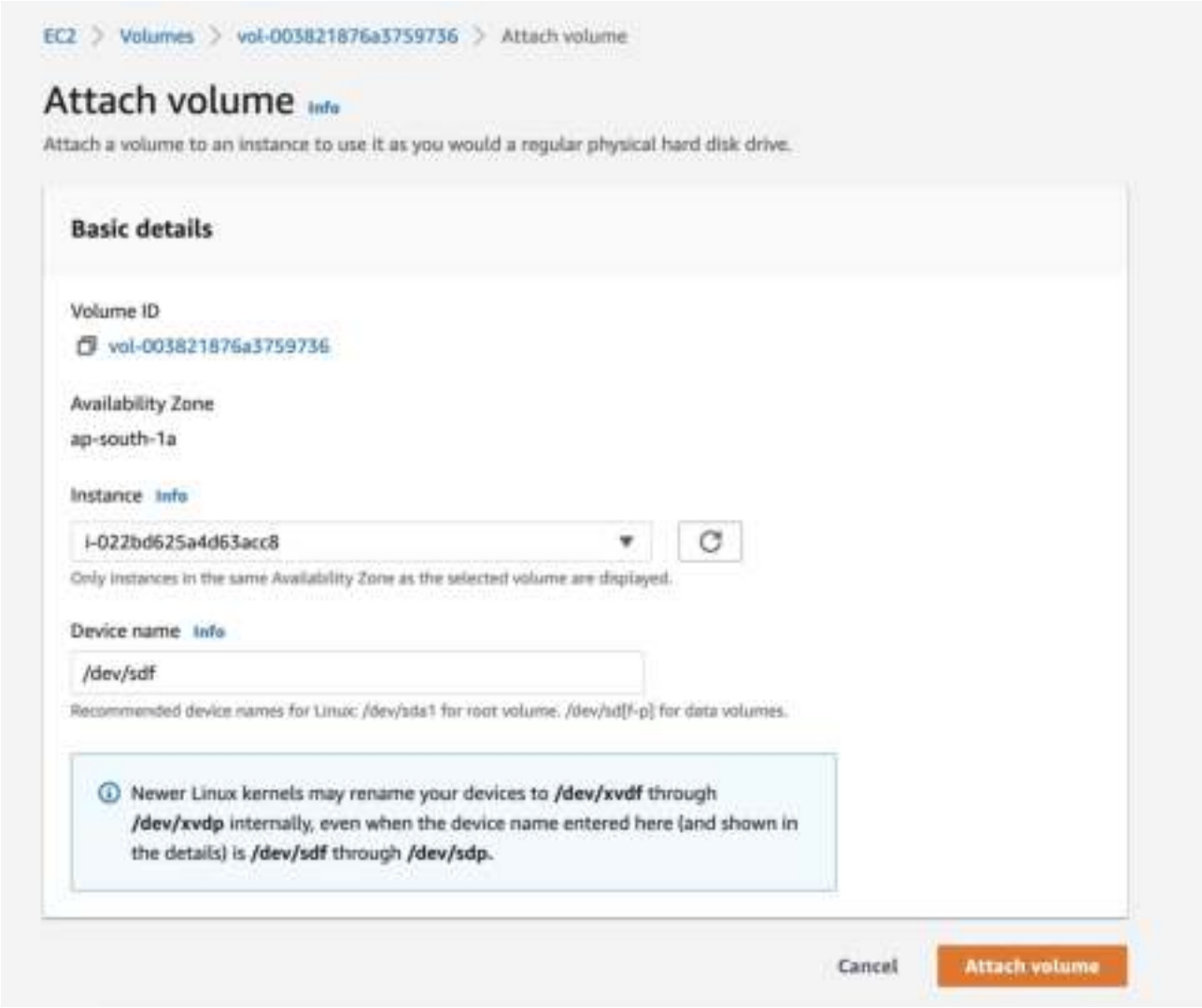
```
[ec2-user@ip-172-31-45-58 ~]$ sudo su
[root@ip-172-31-45-58 ec2-user]# pwd
/home/ec2-user
[root@ip-172-31-45-58 ec2-user]# cd storage/
[root@ip-172-31-45-58 storage]# pwd
/home/ec2-user/storage
[root@ip-172-31-45-58 storage]# touch {1..10}.txt
[root@ip-172-31-45-58 storage]# ls
1.txt  10.txt  2.txt  3.txt  4.txt  5.txt  6.txt  7.txt  8.txt  9.txt
```

13. Unmount the file system and detach the volume from the EC2 instace

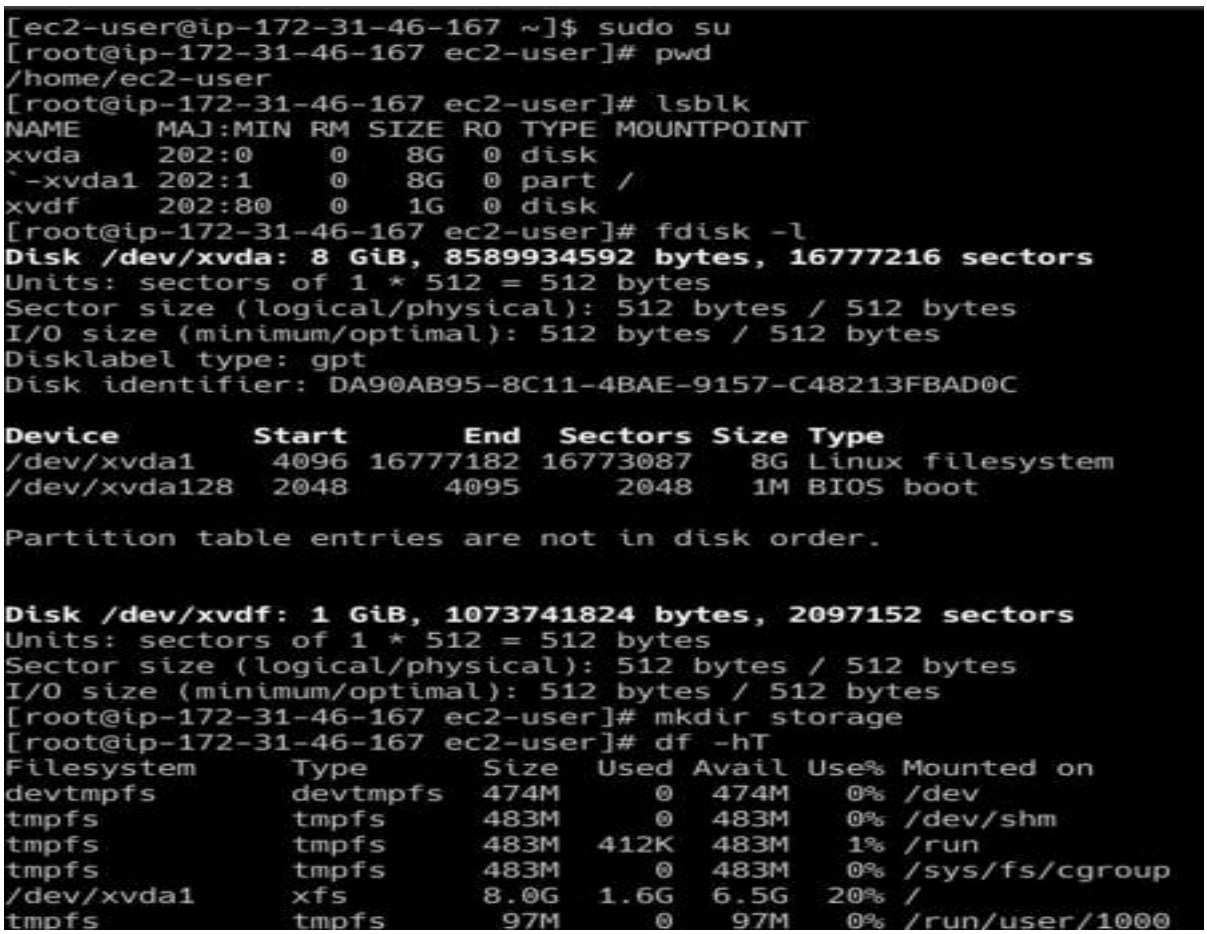
```
[root@ip-172-31-45-58 storage]# cd ..
[root@ip-172-31-45-58 ec2-user]# pwd
/home/ec2-user
[root@ip-172-31-45-58 ec2-user]# unmount /home/ec2-user/storage/
bash: unmount: command not found
[root@ip-172-31-45-58 ec2-user]# umount /home/ec2-user/storage/
[root@ip-172-31-45-58 ec2-user]# df -hT
Filesystem     Type      Size  Used Avail Use% Mounted on
devtmpfs       devtmpfs  474M     0  474M   0% /dev
tmpfs          tmpfs     483M     0  483M   0% /dev/shm
tmpfs          tmpfs     483M  528K  483M   1% /run
tmpfs          tmpfs     483M     0  483M   0% /sys/fs/cgroup
/dev/xvda1     xfs       8.0G  1.6G  6.5G  20% /
tmpfs          tmpfs      97M     0   97M   0% /run/user/1000
tmpfs          tmpfs      97M     0   97M   0% /run/user/0
```

14. Create another machine **uday_aws_linux_ec2_A.**

15. Attach the volume storage to the new machine



16. Mount the file system and access the already created files.

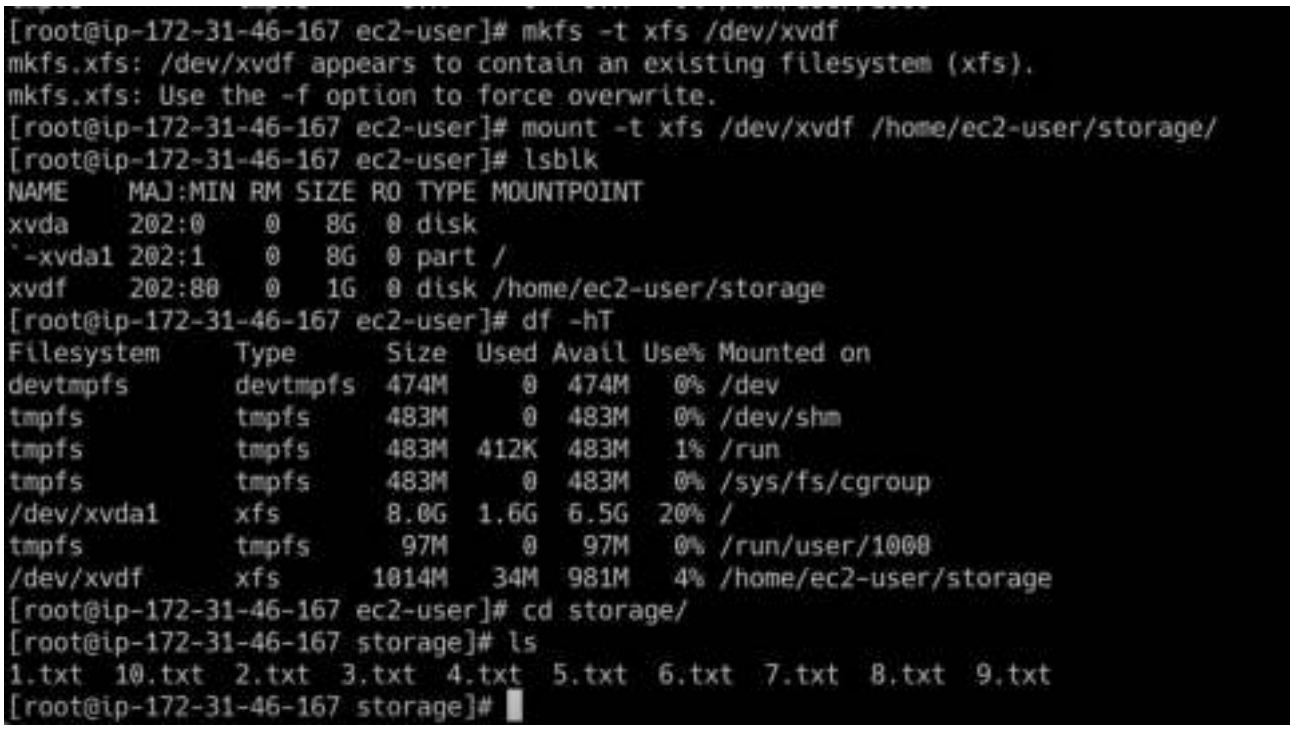


```
[ec2-user@ip-172-31-46-167 ~]$ sudo su
[root@ip-172-31-46-167 ec2-user]# pwd
/home/ec2-user
[root@ip-172-31-46-167 ec2-user]# lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0   8G  0 disk
`-xvda1 202:1    0   8G  0 part /
xvdf    202:80   0   1G  0 disk
[root@ip-172-31-46-167 ec2-user]# fdisk -l
Disk /dev/xvda: 8 GiB, 8589934592 bytes, 16777216 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: DA90AB95-8C11-4BAE-9157-C48213FBAD0C

Device       Start      End  Sectors Size Type
/dev/xvda1    4096 16777182 16773087   8G Linux filesystem
/dev/xvda128  2048     4095     2048   1M BIOS boot

Partition table entries are not in disk order.


Disk /dev/xvdf: 1 GiB, 1073741824 bytes, 2097152 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
[root@ip-172-31-46-167 ec2-user]# mkdir storage
[root@ip-172-31-46-167 ec2-user]# df -hT
Filesystem      Type      Size  Used Avail Use% Mounted on
devtmpfs        devtmpfs  474M     0  474M   0% /dev
tmpfs           tmpfs     483M     0  483M   0% /dev/shm
tmpfs           tmpfs     483M  412K  483M   1% /run
tmpfs           tmpfs     483M     0  483M   0% /sys/fs/cgroup
/dev/xvda1      xfs       8.0G  1.6G  6.5G  20% /
tmpfs           tmpfs      97M     0   97M   0% /run/user/1000
```

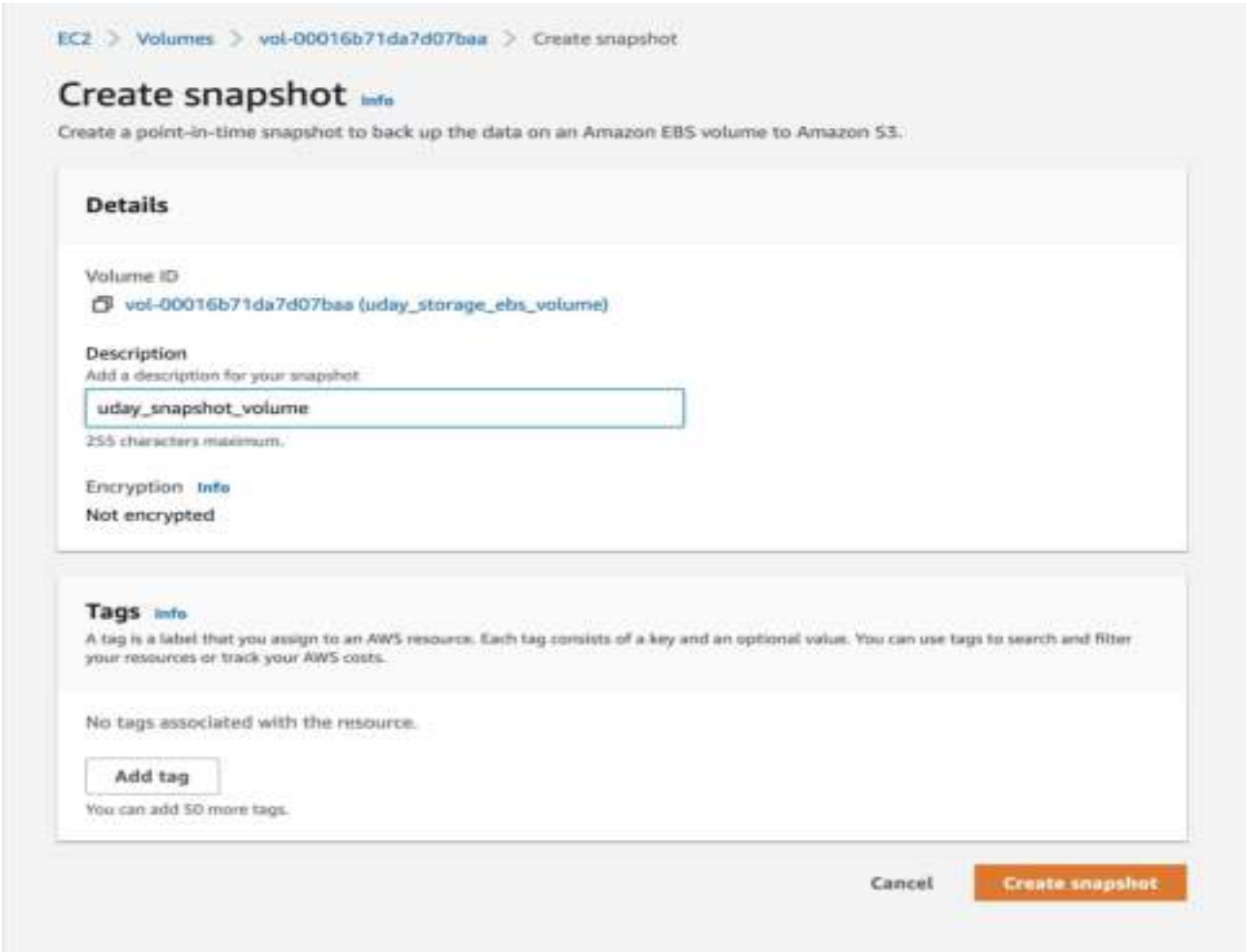


EBS volumes are created successfully and attached with different EC2 instances without loosing data

Create a snapshot of the Volume or Instance:

Creates a prototype of the service which can be used to in other availability zone.
Snapshot can be copied to multiple regions. Deleted snapshots reside in recycle bin for the period defined in retention policy.

1. Create a snapshot of the existing EBS volume



2. Create a volume from the new snapshot in a different region (us-east-1)

3. Create a new EC2 instance in different region (us-east-1)

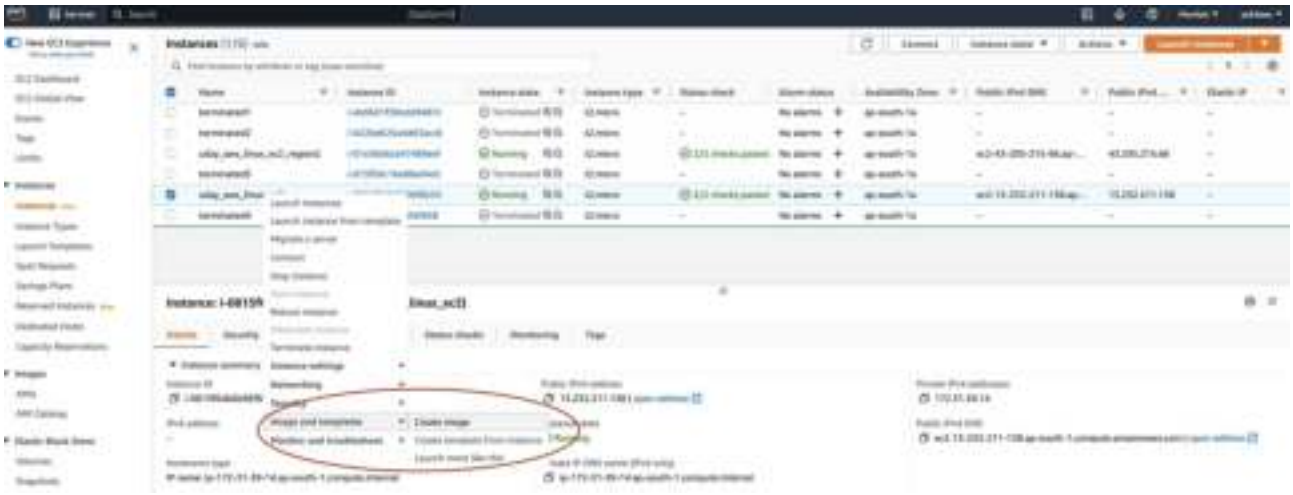
4. Attach the new volume with the new EC2 instances

The Snapshot of the volume is created successfully, recreated the volume in different region and attached to the EC2 instance with data persisting

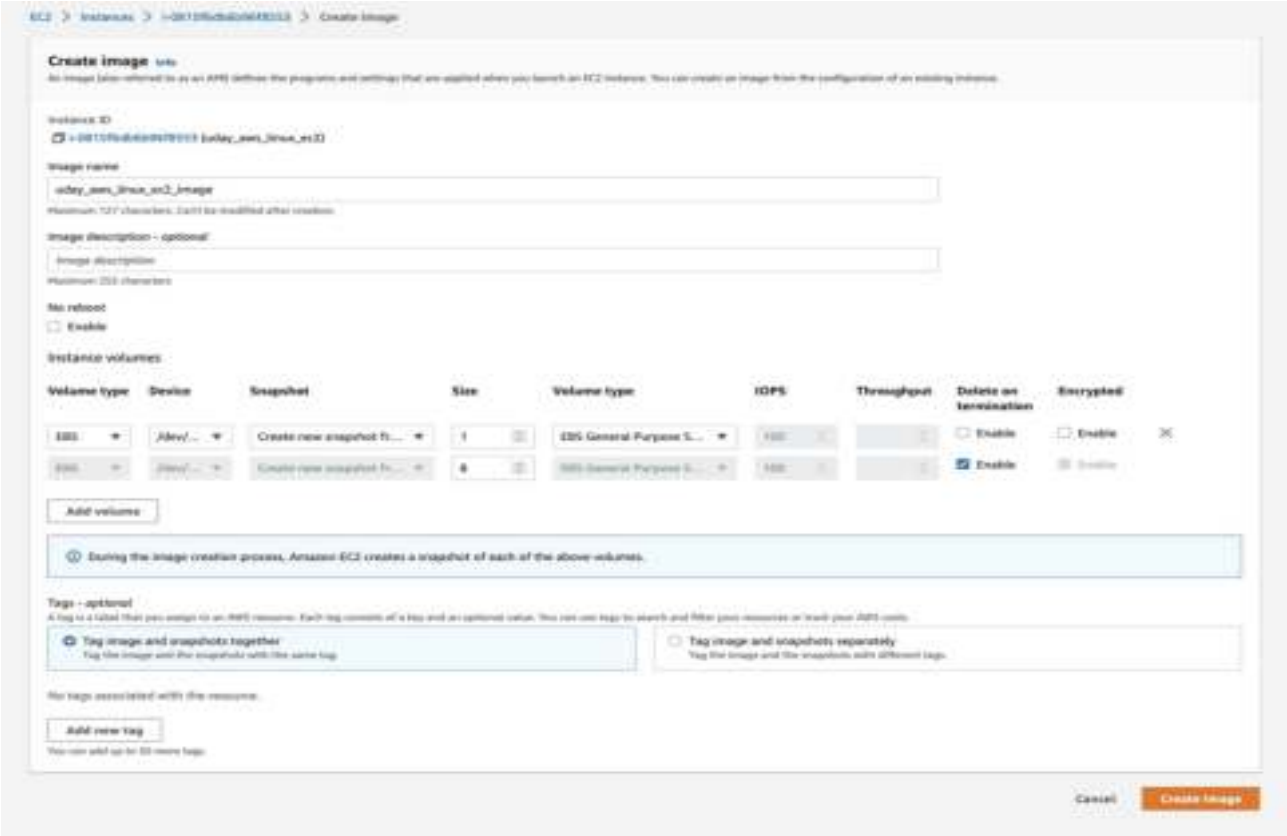
Create an AMI from an Amazon EC2 Instance

From the **Amazon EC2 Instances** view, you can create Amazon Machine Images (AMIs) from either running or stopped instances.

1. Right-click the instance you want to use as the basis for your AMI, and choose **Create Image** from the context menu.



2. In the **Create Image** dialog box, type a unique name and description, and then choose **Create Image**. By default, Amazon EC2 shuts down the instance, takes snapshots of any attached volumes, creates and registers the AMI, and then reboots the instance. Choose **No reboot**if you don't want your instance to be shut down.



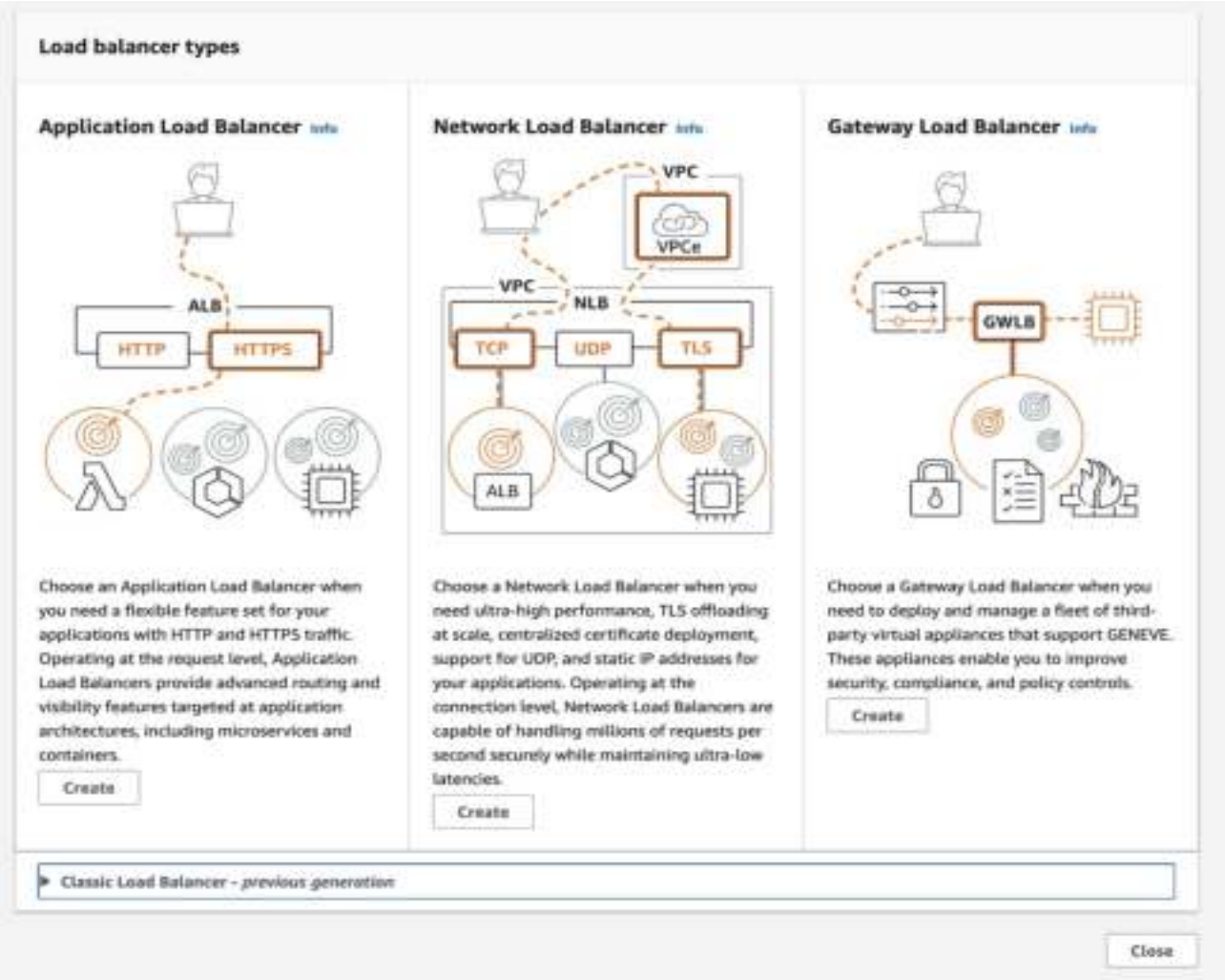
3. Navigate:- **EC2 -> Images -> AMIs.**

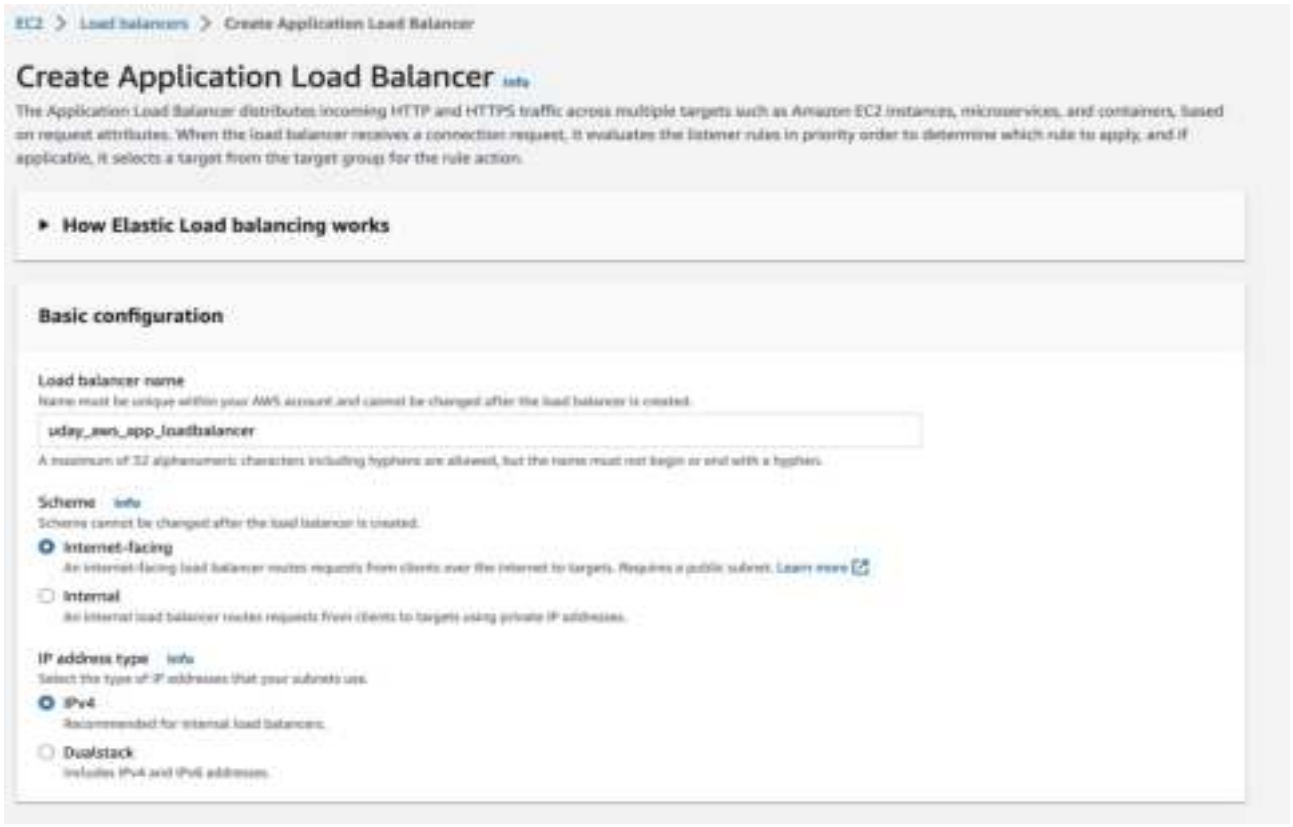
You can Launch a new EC2 instance from the AMI

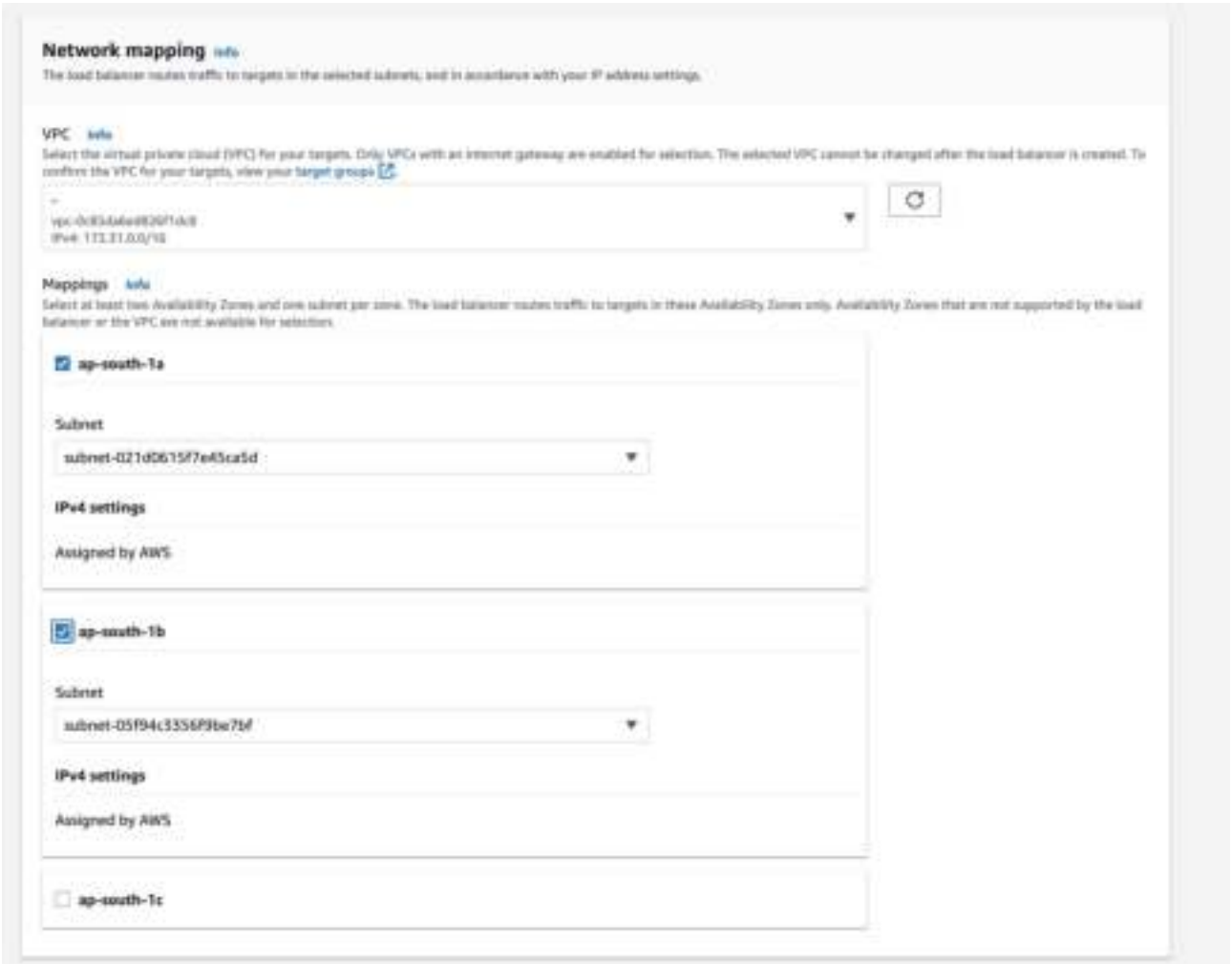
Successfully created an Image fro existing EC2 instance.

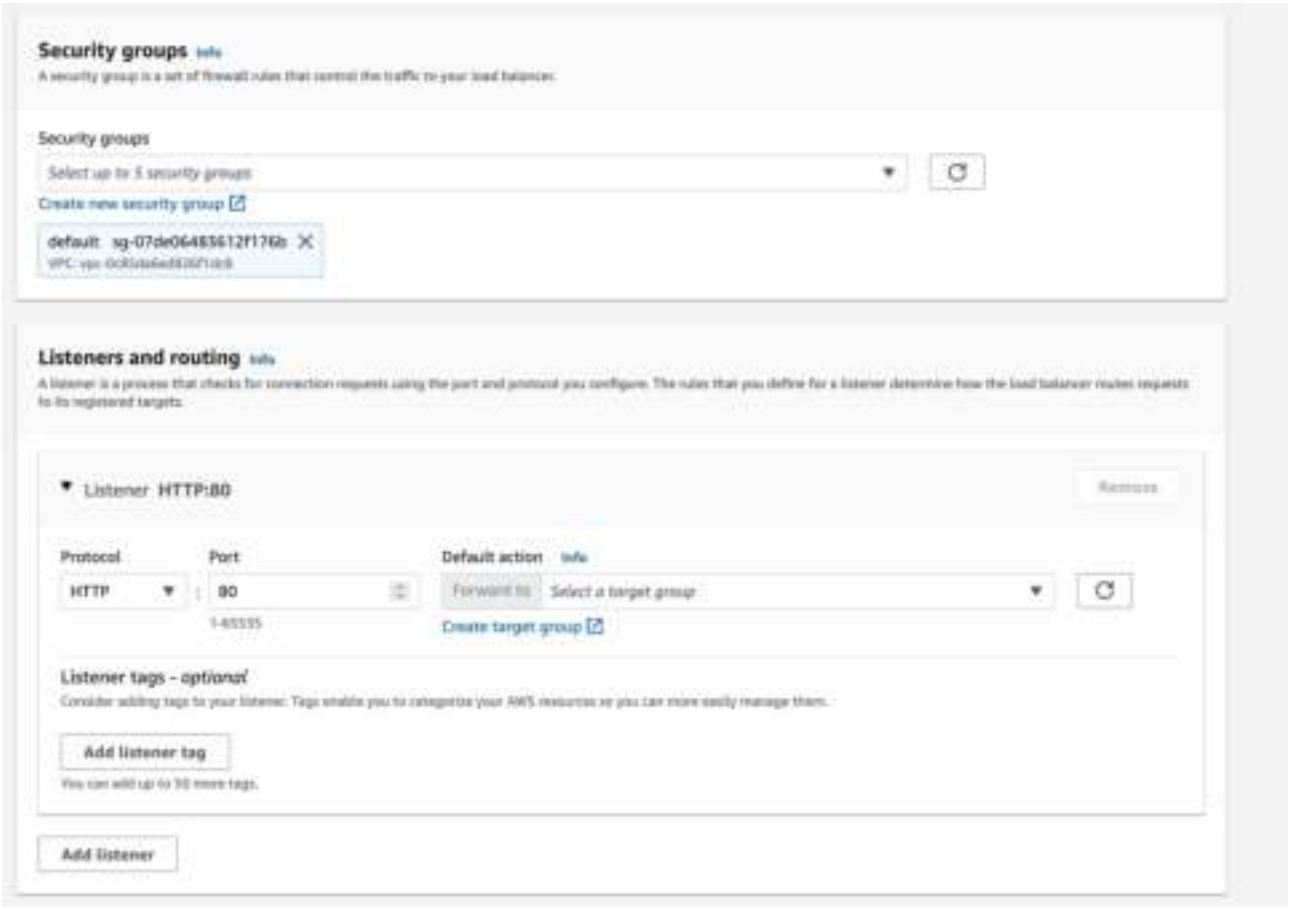
Create Load Balancer:

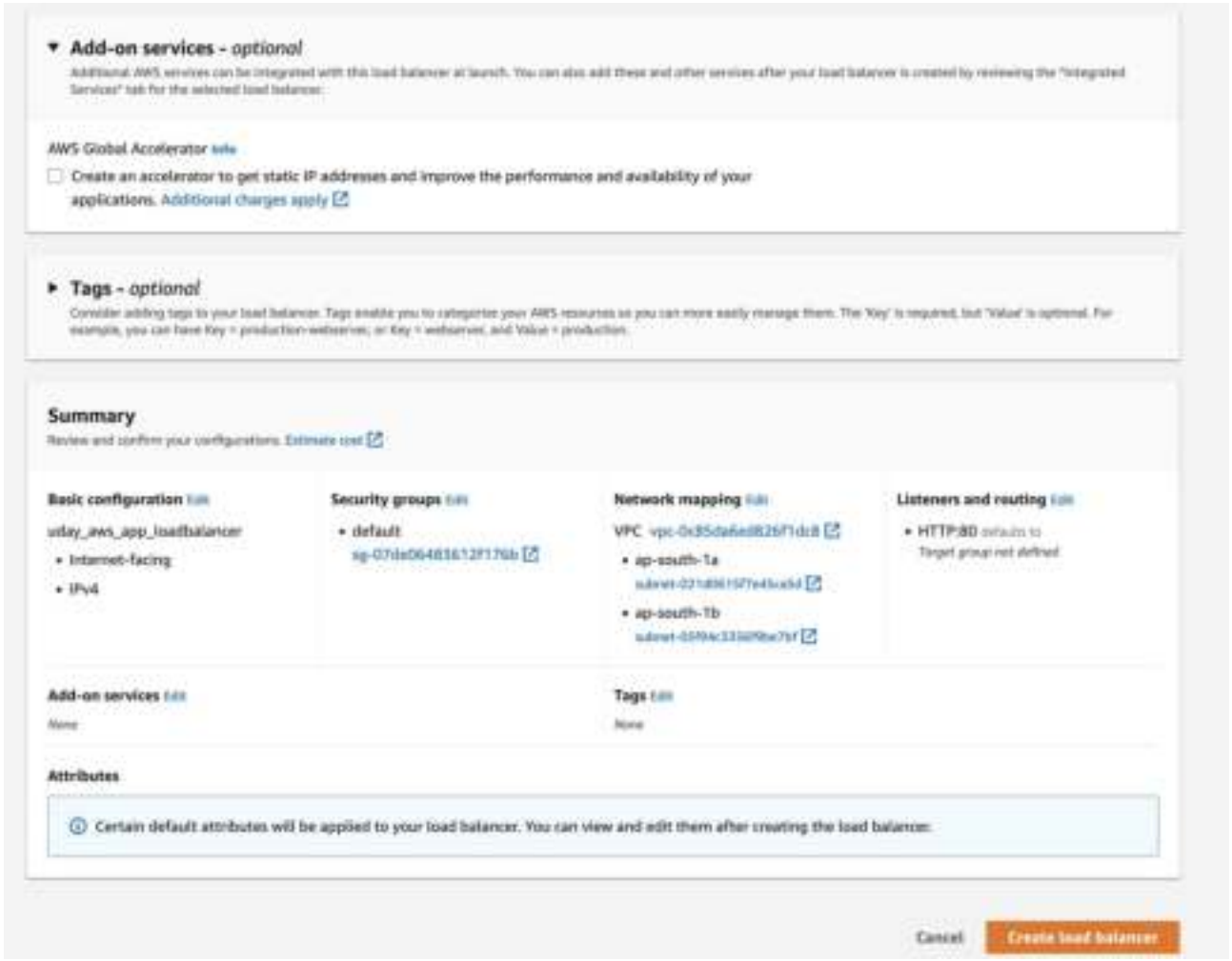
1. Navigate to EC2 -> Load Balancing -> Load Balancers. And click create Load Balancers

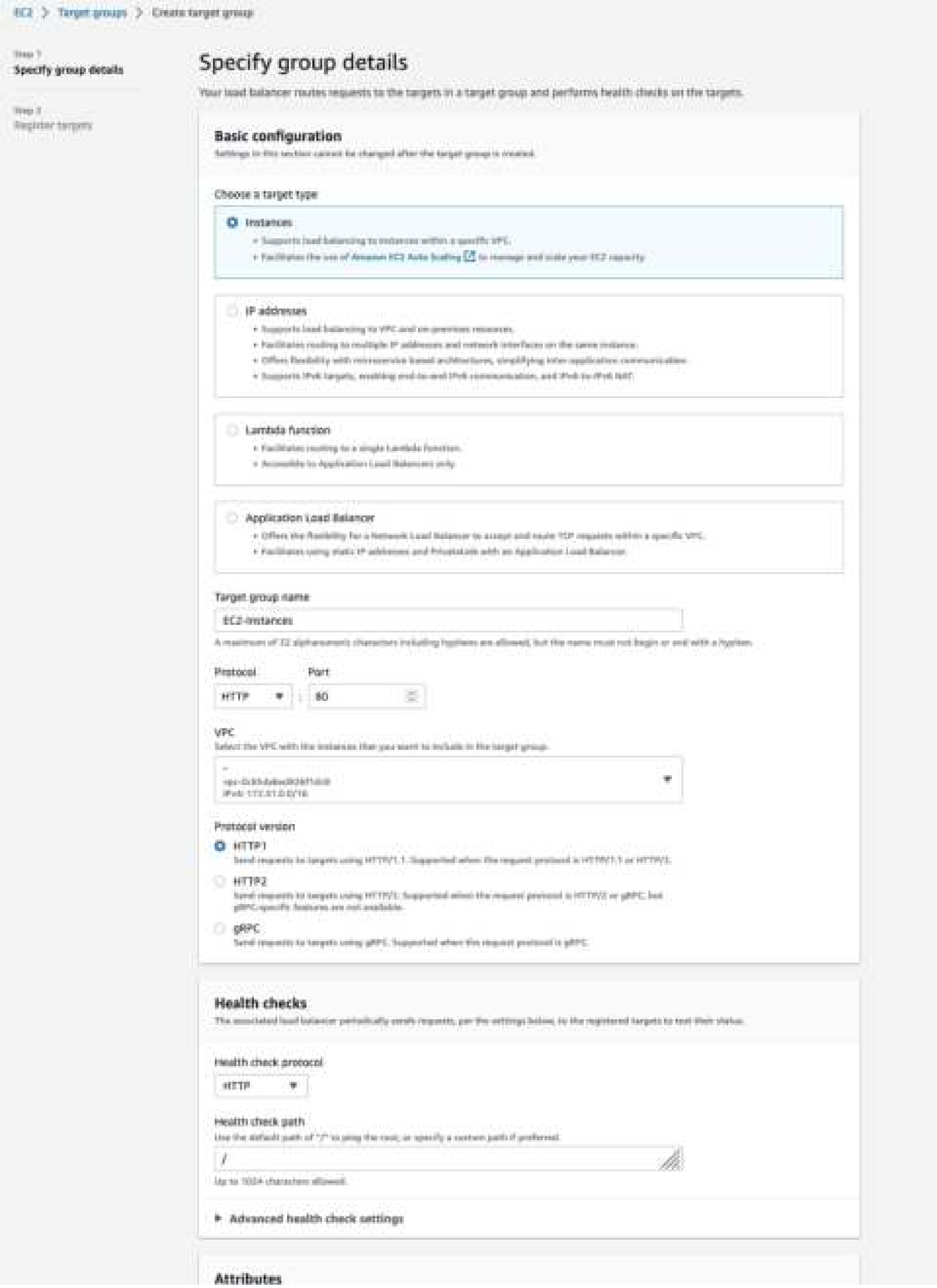



EC2 > Load balancers > Create Application Load Balancer
Create Application Load Balancer info
The Application Load Balancer distributes incoming HTTP and HTTPS traffic across multiple targets such as Amazon EC2 instances, microservices, and containers, based on request attributes. When the load balancer receives a connection request, it evaluates the listener rules in priority order to determine which rule to apply, and if applicable, it selects a target from the target group for the rule action.
How Elastic Load balancing works
Basic configuration
Load balancer name
Name must be unique within your AWS account and cannot be changed after the load balancer is created.
uday_aws_app_loadbalancer
A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.
Scheme info
Scheme cannot be changed after the load balancer is created.
Internet-facing
An internet-facing load balancer routes requests from clients over the internet to targets. Requires a public subnet. Learn more
Internal
An internal load balancer routes requests from clients to targets using private IP addresses.
IP address type info
Select the type of IP addresses that your subnets use.
IPv4
Recommended for internal load balancers.
Dualstack
Includes IPv4 and IPv6 addresses.


Network mapping info
The load balancer routes traffic to targets in the selected subnets, and in accordance with your IP address settings.
VPC info
Select the virtual private cloud (VPC) for your targets. Only VPCs with an internet gateway are enabled for selection. The selected VPC cannot be changed after the load balancer is created. To confirm the VPC for your targets, view your target groups
Mappings info
Select at least two Availability Zones and one subnet per zone. The load balancer routes traffic to targets in these Availability Zones only. Availability Zones that are not supported by the load balancer or the VPC are not available for selection.
ap-south-1a
Subnet
subnet-021d0615f7e45ca5d
IPv4 settings
Assigned by AWS
ap-south-1b
Subnet
subnet-05f94c3356f9be7bf
IPv4 settings
Assigned by AWS
ap-south-1c


Security groups info
A security group is a set of firewall rules that control the traffic to your load balancer.
Security groups
Select up to 5 security groups
Create new security group
default sg-07de06485612f176b
Listeners and routing info
A listener is a process that checks for connection requests using the port and protocol you configure. The rules that you define for a listener determine how the load balancer routes requests to its registered targets.
Listener HTTP:80
Remove
Protocol
Port
Default action info
HTTP
80
1-65535
Forward to
Select a target group
Create target group
Listener tags - optional
Consider adding tags to your listener. Tags enable you to categorize your AWS resources so you can more easily manage them.
Add listener tag
You can add up to 50 more tags.
Add listener


Add-on services - optional
Additional AWS services can be integrated with this load balancer at launch. You can also add these and other services after your load balancer is created by reviewing the "Integrated Services" tab for the selected load balancer.
AWS Global Accelerator Info
Create an accelerator to get static IP addresses and improve the performance and availability of your applications. Additional charges apply
Tags - optional
Consider adding tags to your load balancer. Tags enable you to categorize your AWS resources so you can more easily manage them. The 'Key' is required, but 'Value' is optional. For example, you can have Key = production-webserver, or Key = webserver, and Value = production.
Summary
Review and confirm your configurations. Estimate cost
Basic configuration Edit
utlay_aws_app_loadbalancer
Internet-facing
IPv4
Security groups Edit
default
Network mapping Edit
VPC
ap-south-1a
ap-south-1b
Listeners and routing Edit
HTTP:80
Add-on services Edit
None
Tags Edit
None
Attributes
Certain default attributes will be applied to your load balancer. You can view and edit them after creating the load balancer.
Cancel
Create load balancer


EC2
Target groups
Create target group
Specify group details
Specify group details
Your load balancer routes requests to the targets in a target group and performs health checks on the targets.
Basic configuration
Choose a target type
Instances
IP addresses
Lambda function
Application Load Balancer
Target group name
EC2-instances
Protocol
Port
HTTP
80
VPC
Protocol version
HTTP1
HTTP2
gRPC
Health checks
Health check protocol
HTTP
Health check path
/
Advanced health check settings
Attributes

Application Load Balancer is created and registered available EC2 instances successfully