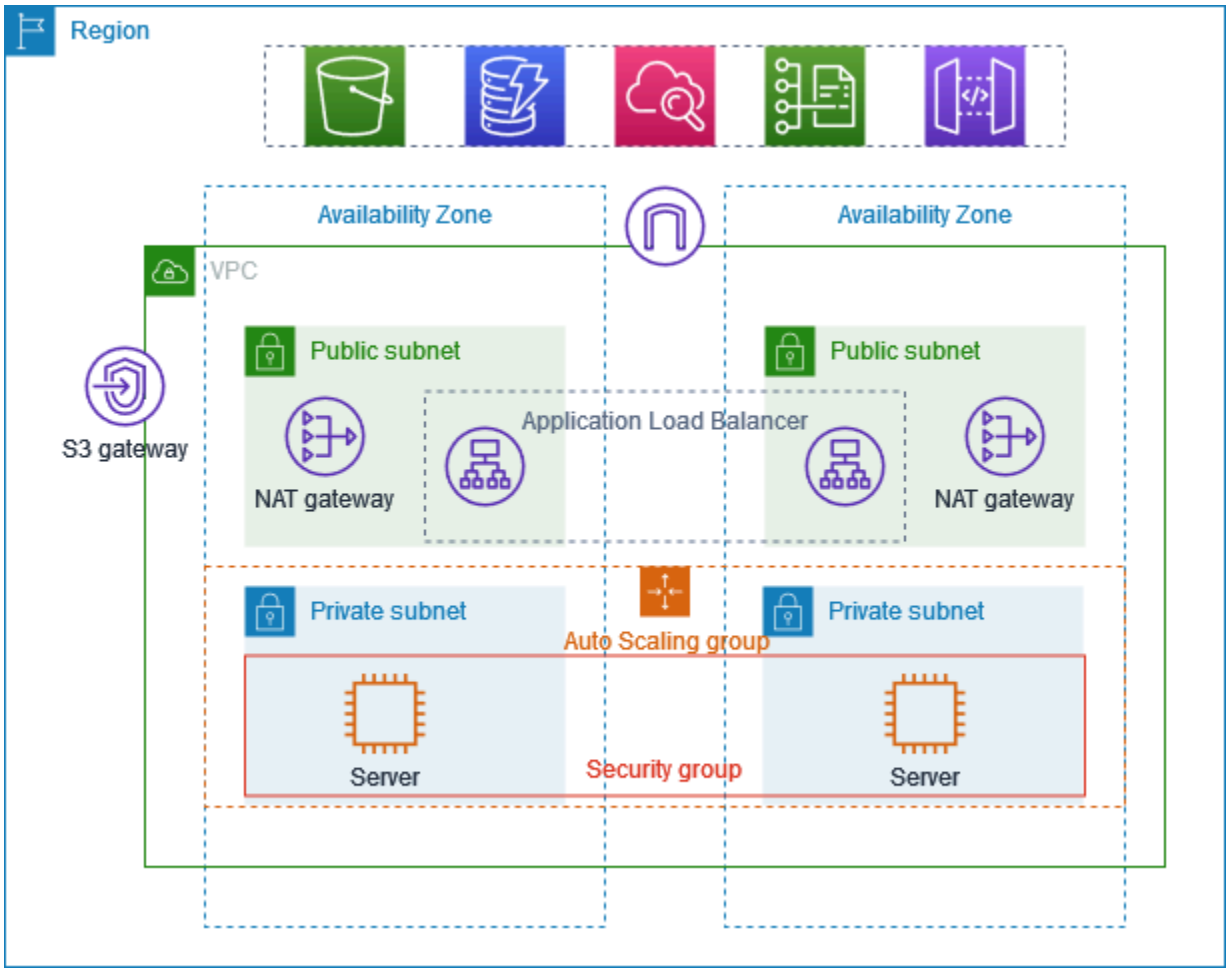
VPC with servers in private subnets and NAT



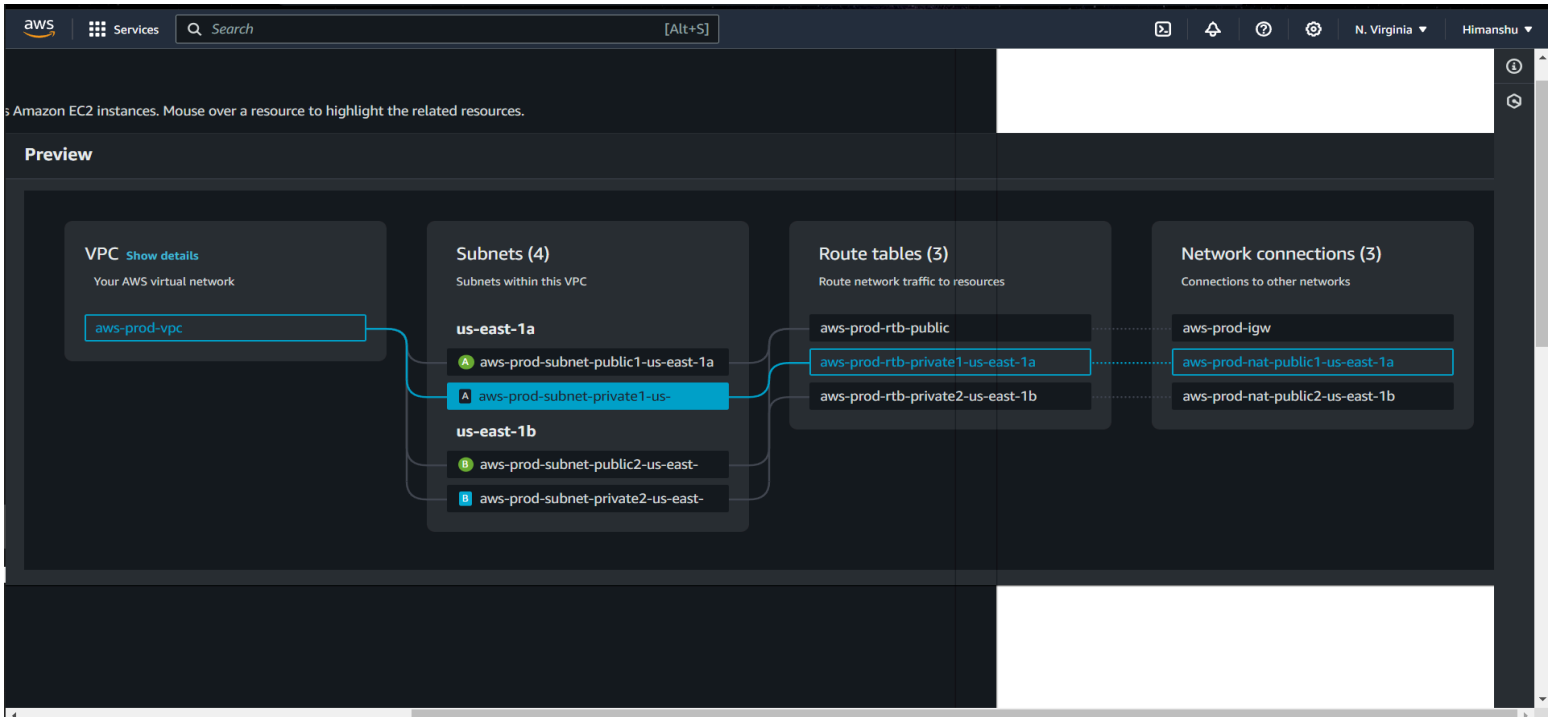
🌟 A Heartfelt Thank You to Abhishek Veeramalla! 🌟

I completed this VPC project,The insights and detailed explanations provided in this video were incredibly helpful.

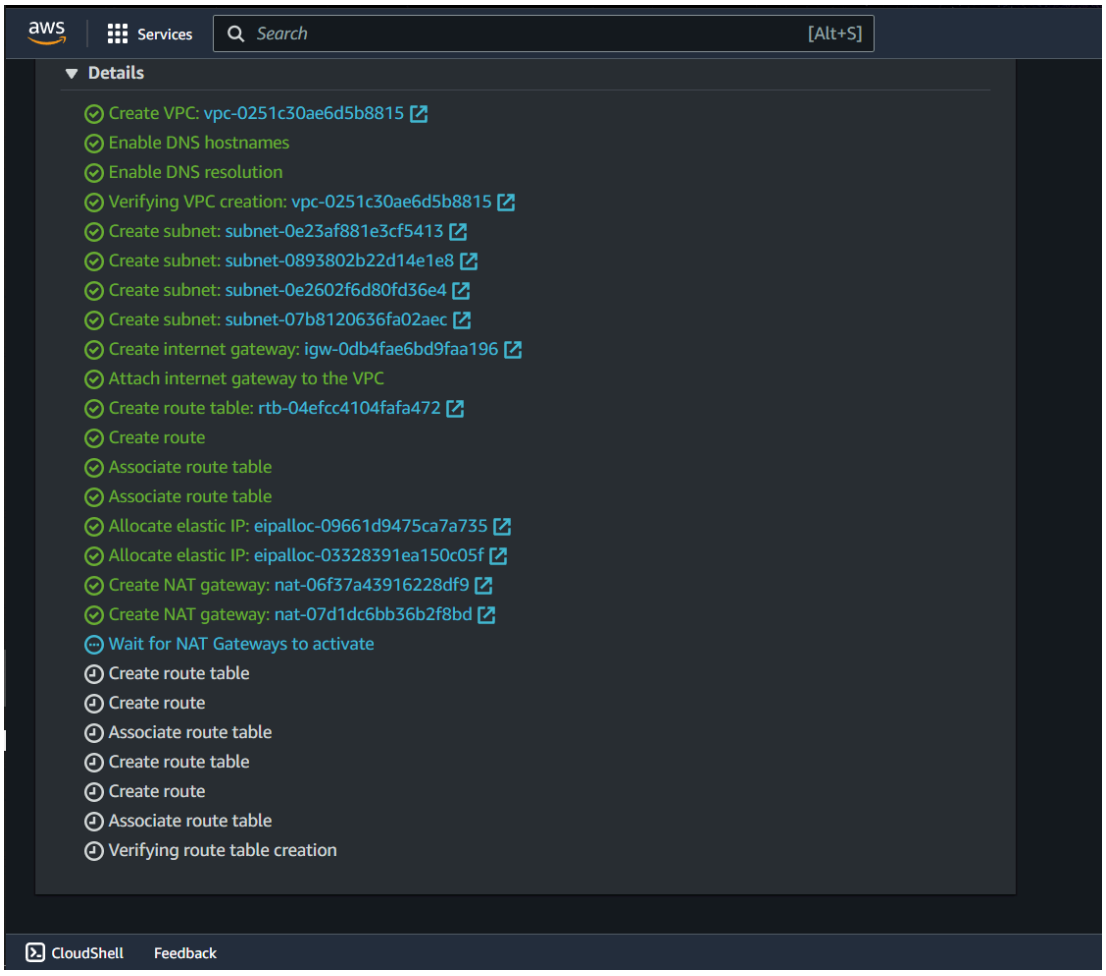
Through this video, I learned so many new things about VPCs and cloud networking. Abhishek, your ability to break down complex concepts and make them accessible is truly remarkable.

Thank you for sharing your knowledge and for making such valuable content available to the community. Your work is making a significant impact, and I'm grateful for the learning opportunity you provided.

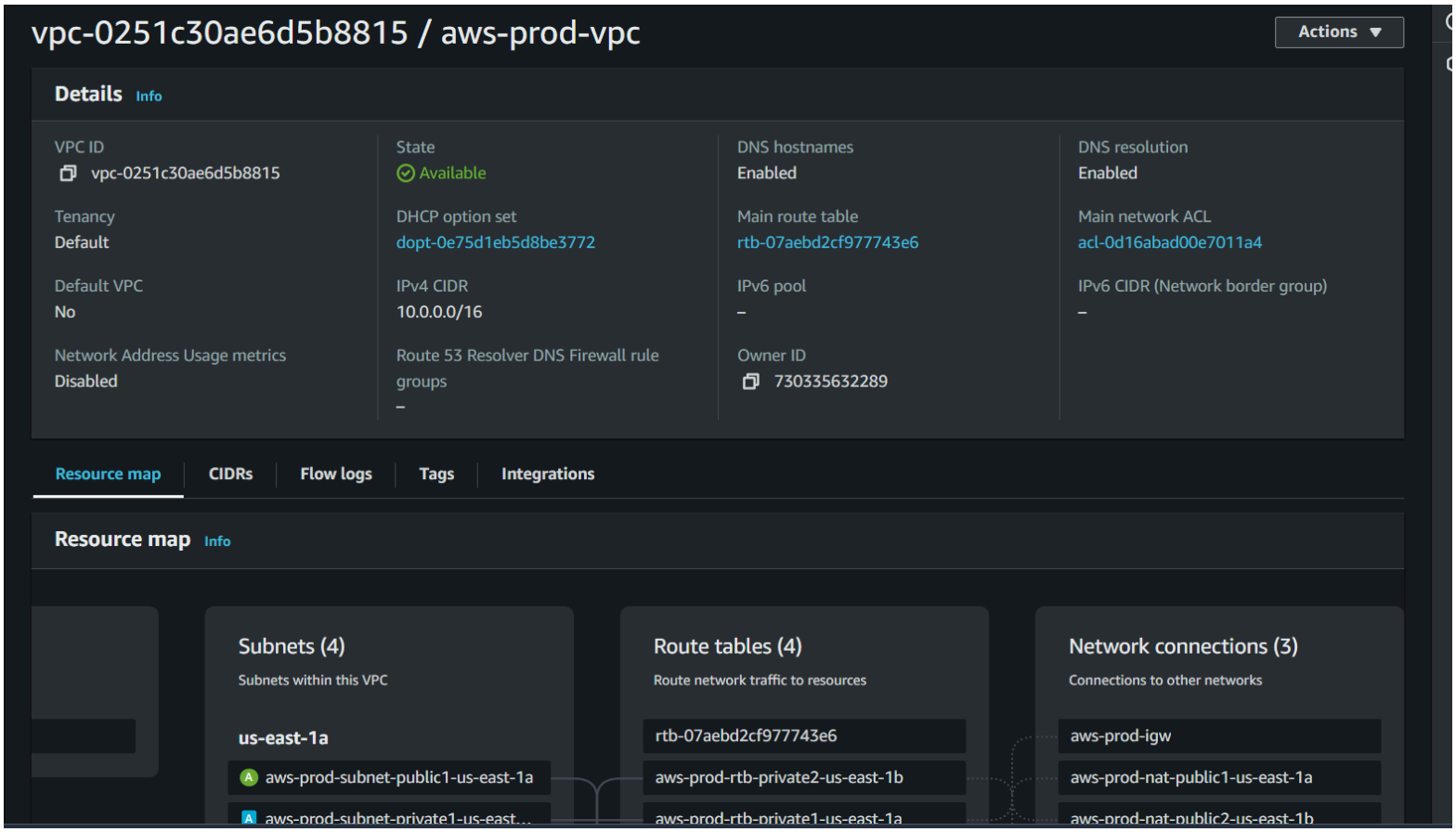
So in this project we are going to create a VPC(Virtual Private Cloud) first.

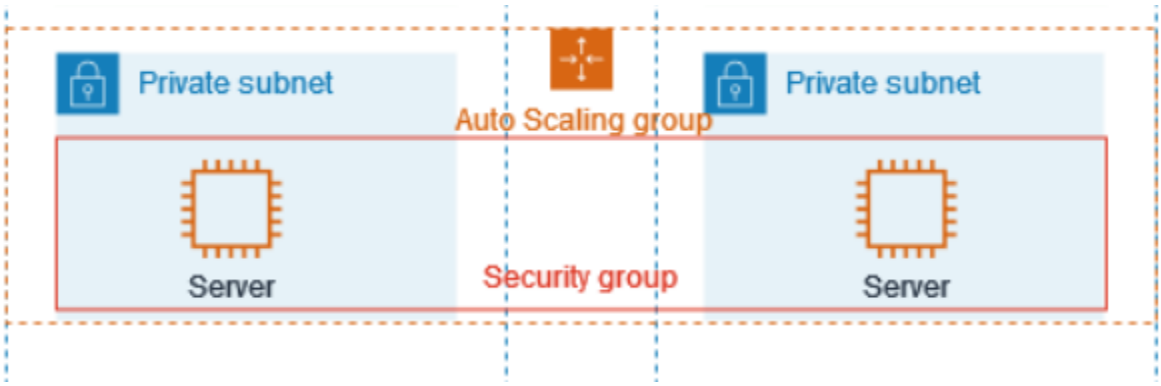


We have to create in 2 AZ with each having one public and private subnet .



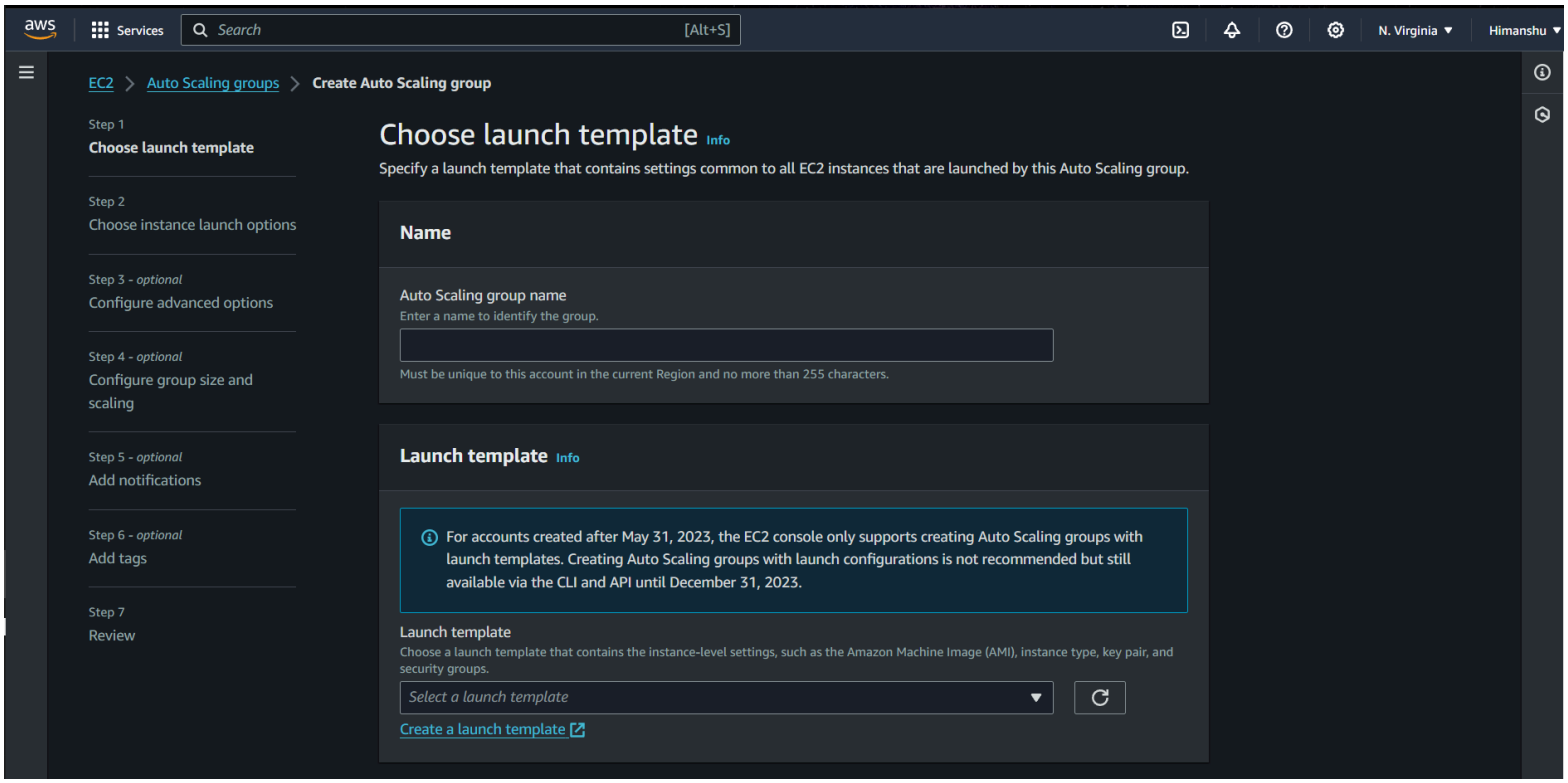
So , we can see in the above screenshot what all things VPC creates for us.



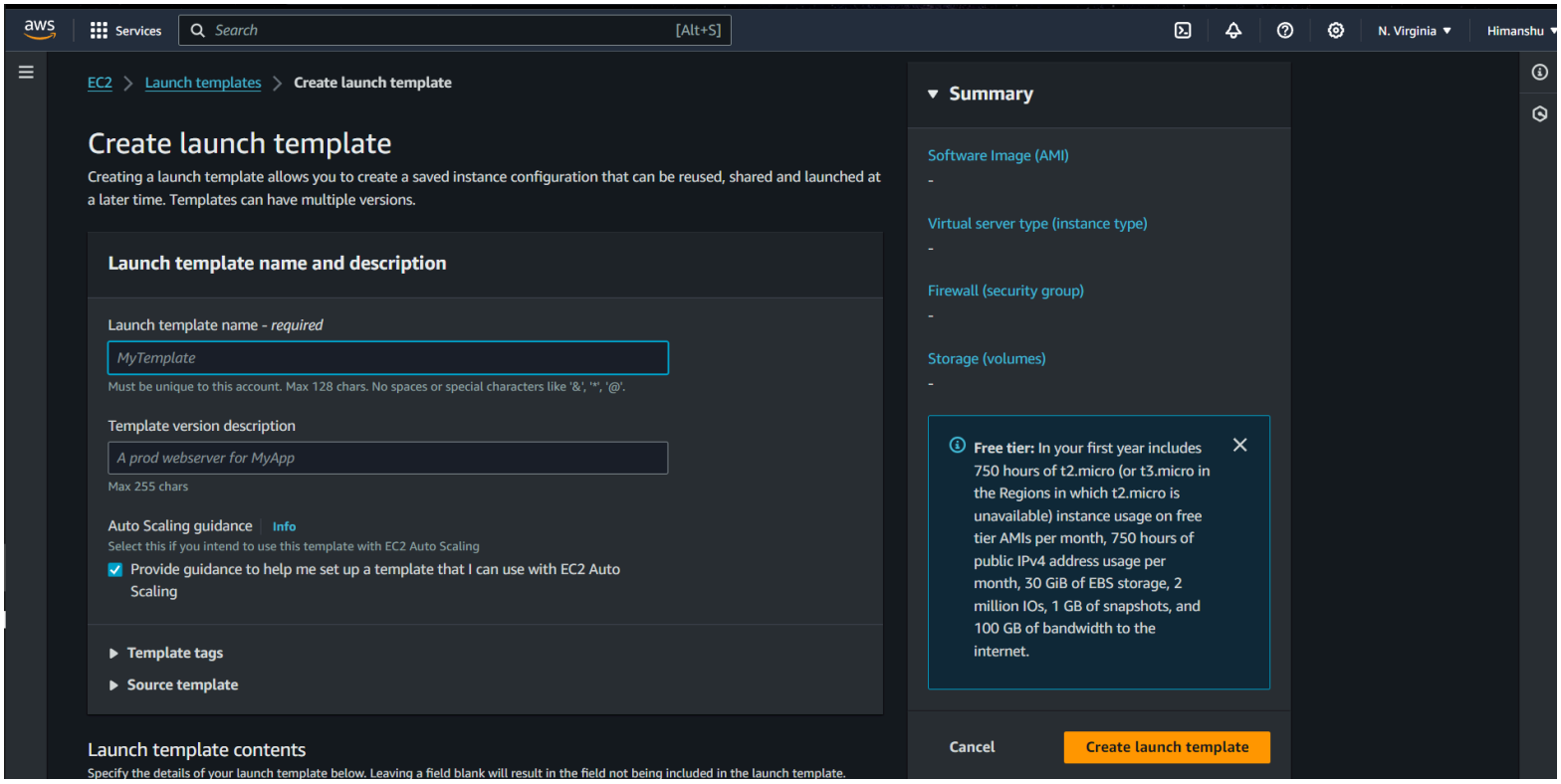


- Now we need to create the Auto Scaling Group and for creating ASG we need to create the launch template .

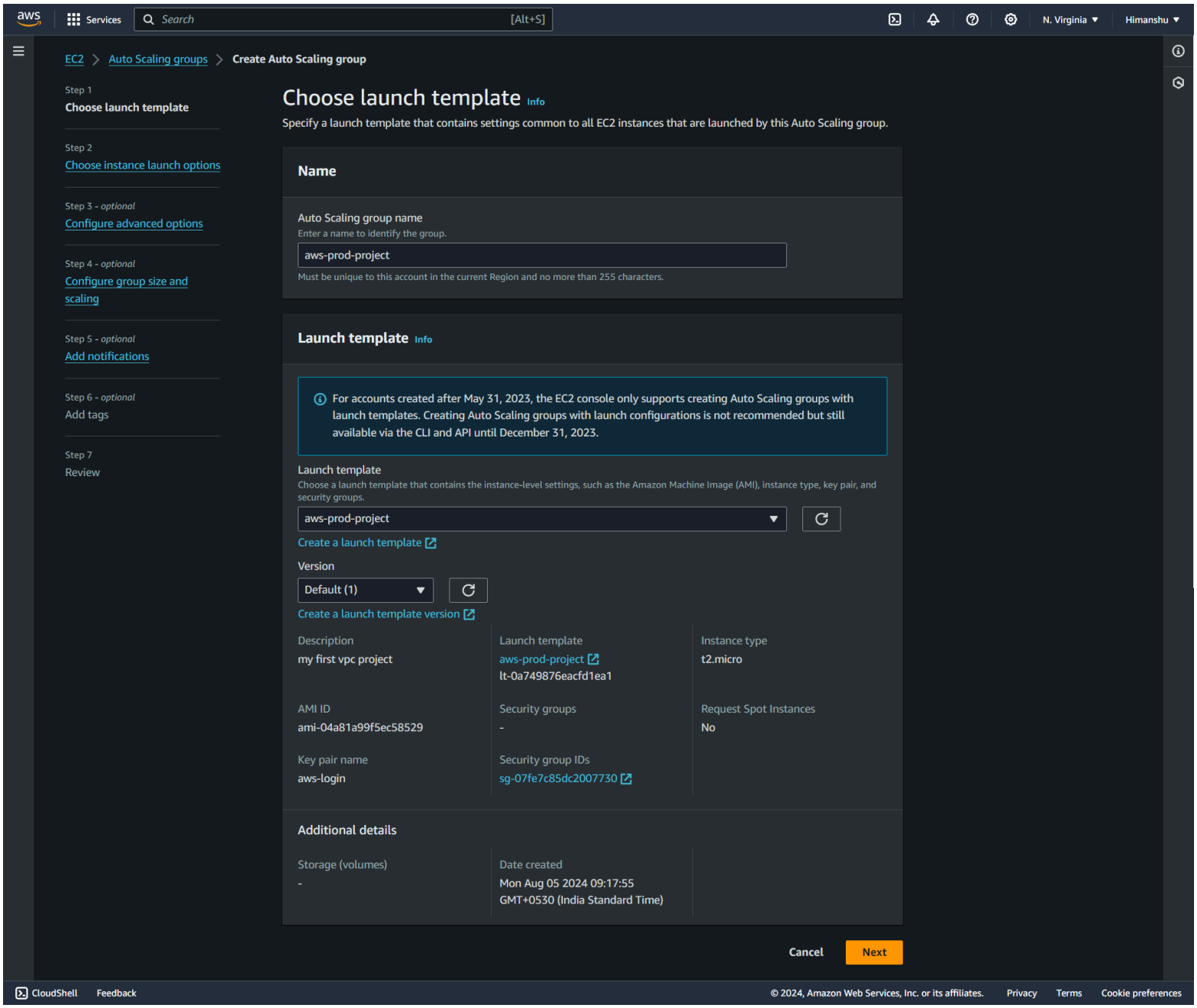
- A launch template in an Auto Scaling Group (ASG) in AWS specifies information for configuring Amazon EC2 instances. It includes details like the Amazon Machine Image (AMI) ID, instance type, key pair, and security groups. Launch templates are similar to launch configurations.



Creating a launch template :

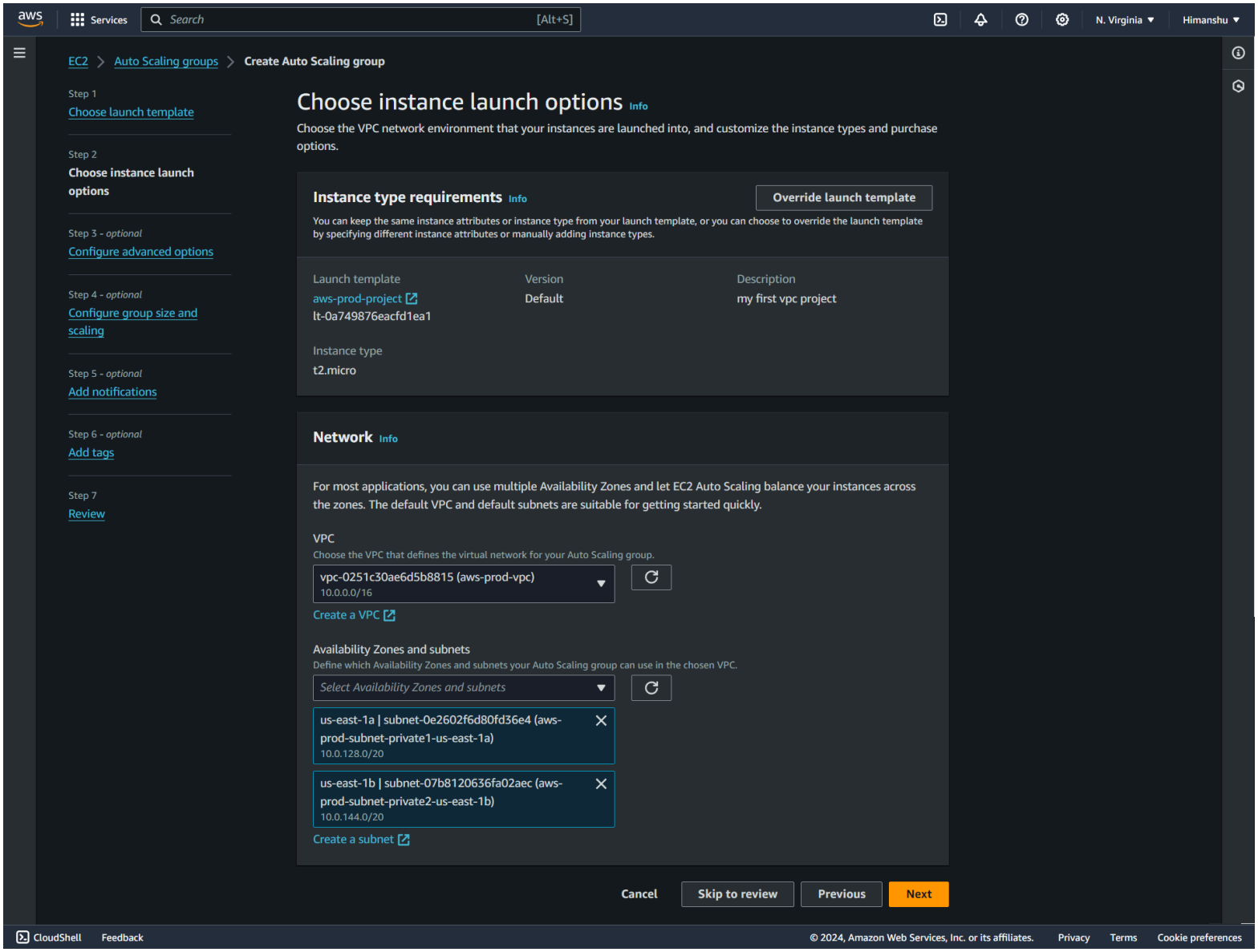


After creating a launch template you can select it in ASG.

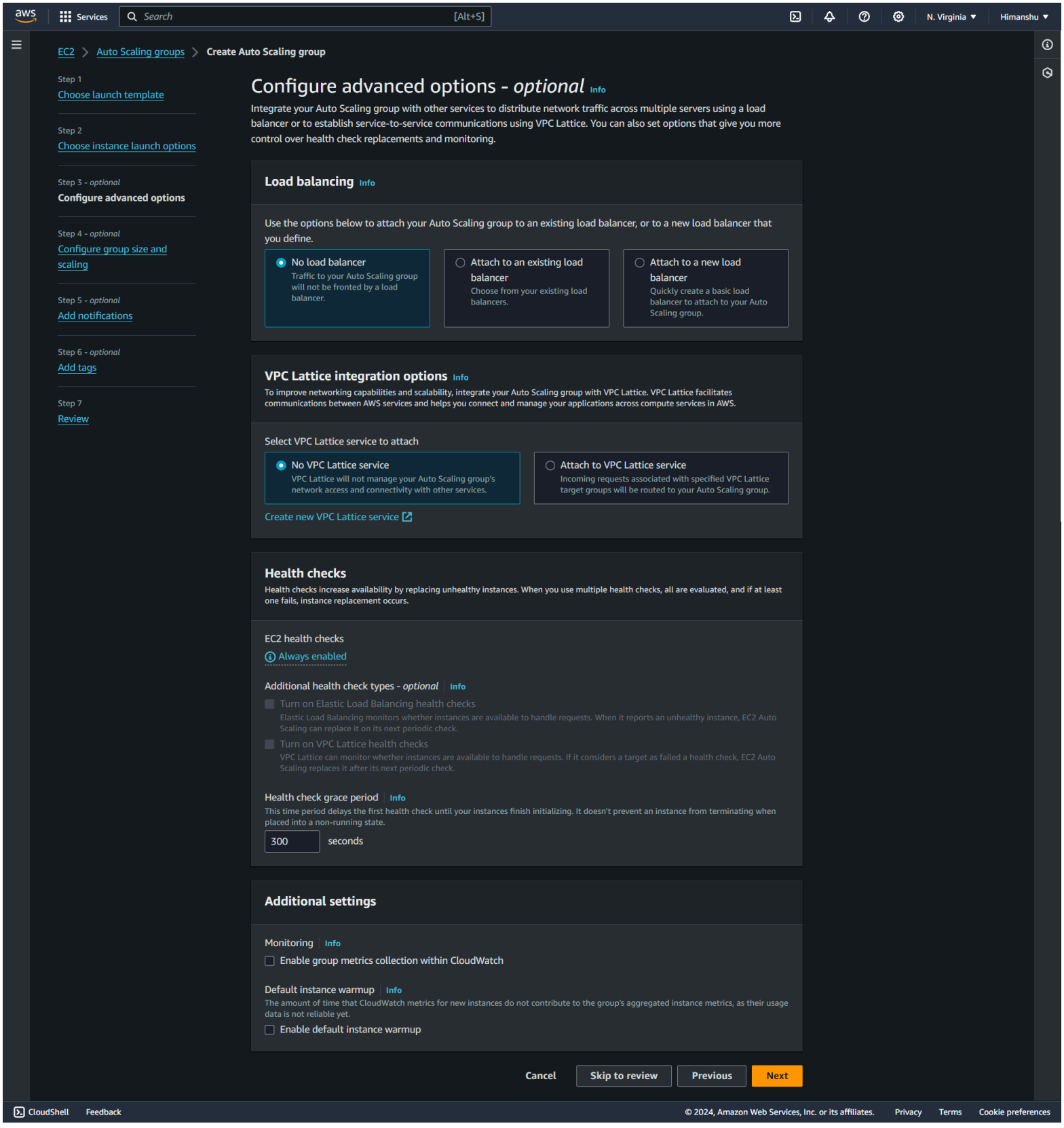


Here you can see , we selected the launch Template in ASG. After that you need to select in which VPC and AZ you want to launch the instance.

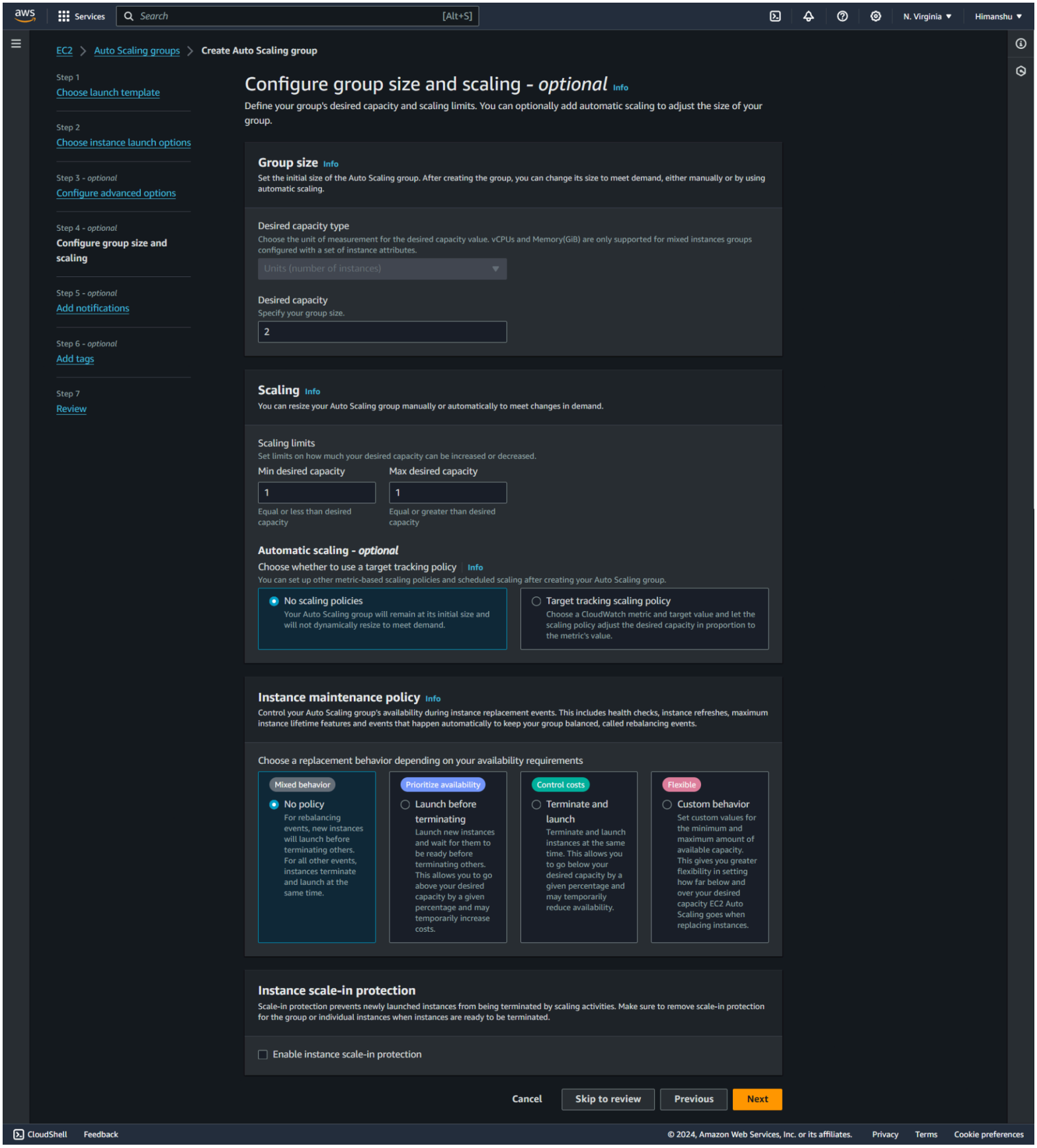
So, we can see I selected the VPC and the AZ with a subnet .

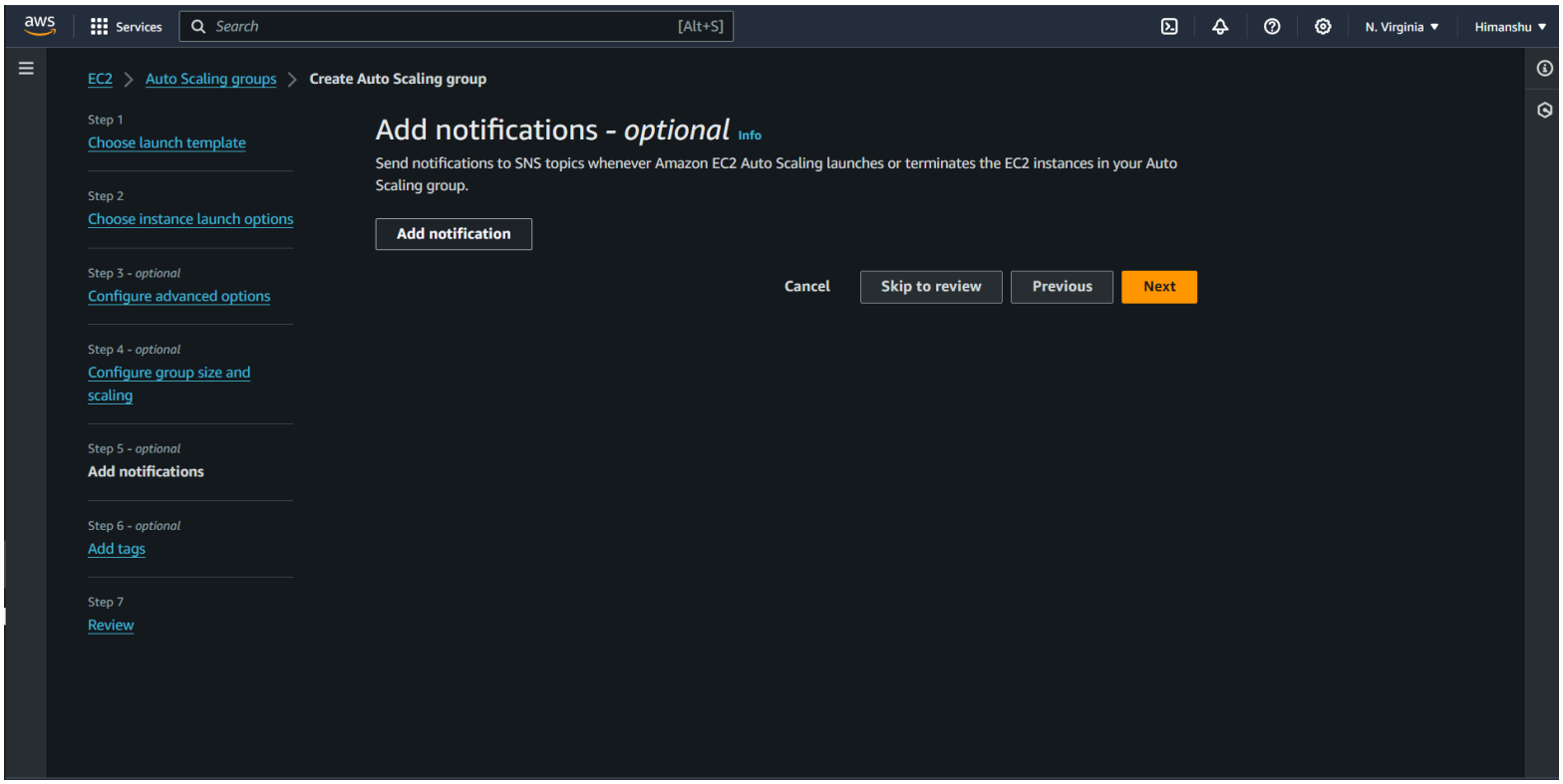


We are not configuring the Load balancer here .

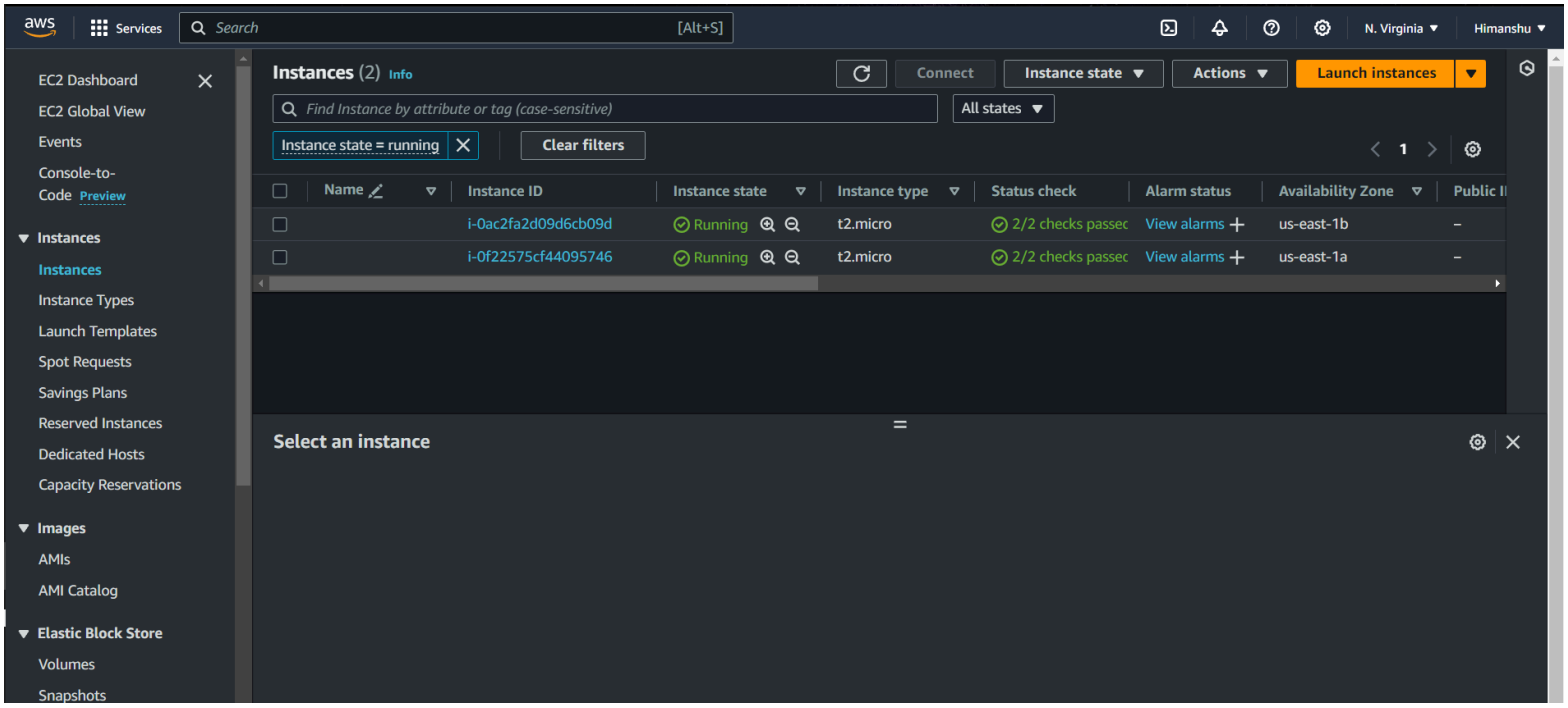


Here we select the size and scaling of ASG.

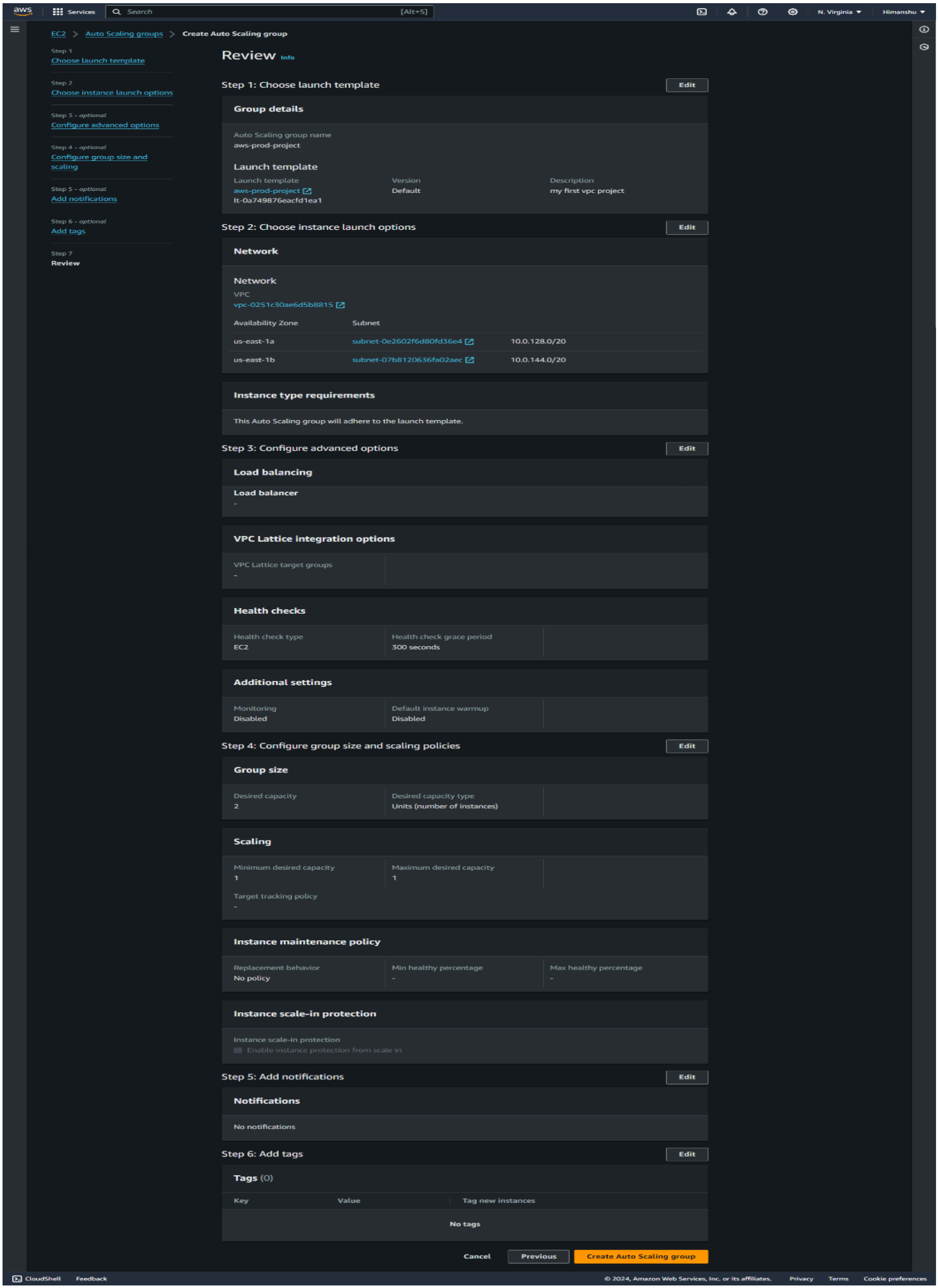




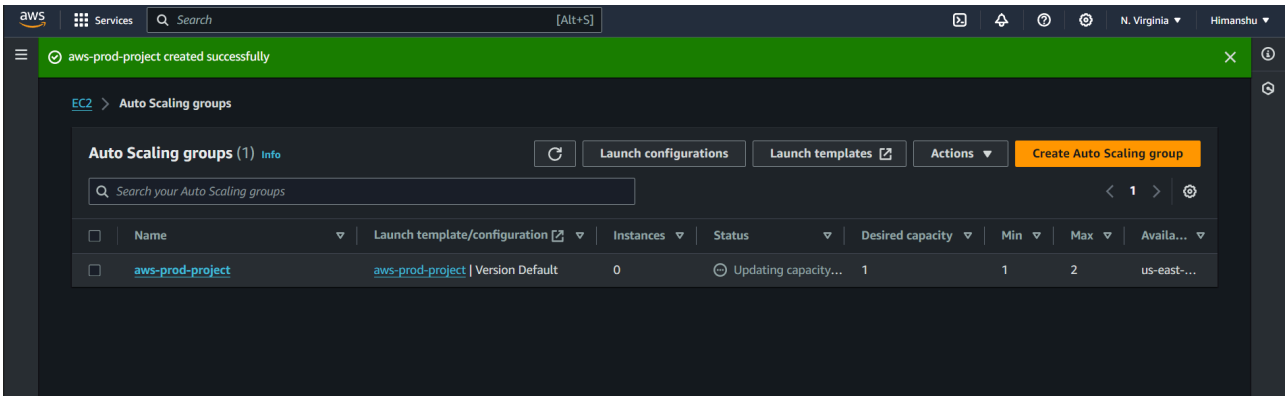
Auto Scaling Group created two instances in the two different AZ us-east-1b and us-east-1a .

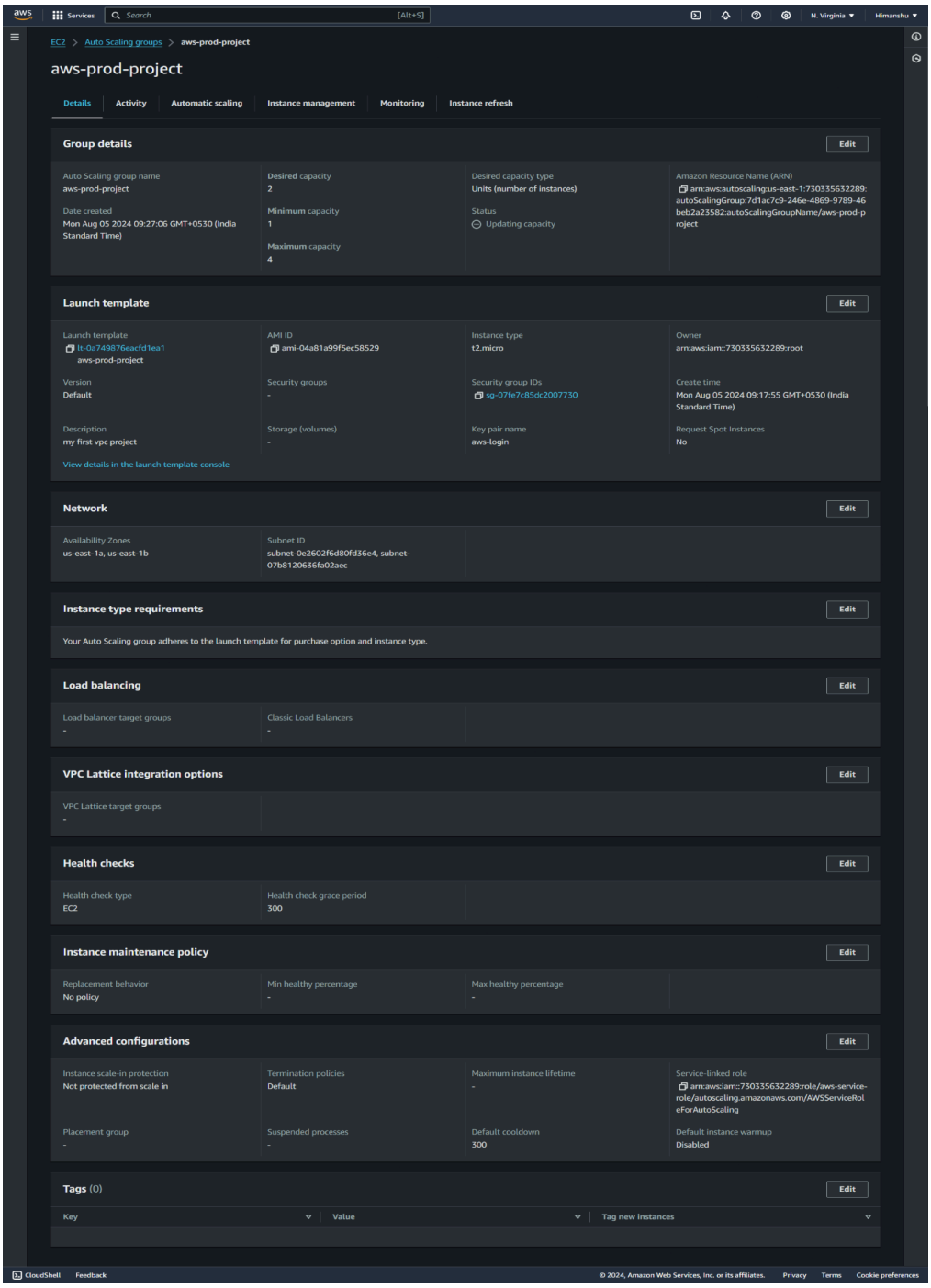


We can review all the settings which we did to create ASG.


aws
Services
Search
[Alt+S]
N. Virginia
Himanshu
EC2 > Auto Scaling groups > Create Auto Scaling group
Step 1
Choose launch template
Step 2
Choose instance launch options
Step 3 - optional
Configure advanced options
Step 4 - optional
Configure group size and scaling
Step 5 - optional
Add notifications
Step 6 - optional
Add tags
Step 7
Review
Review Info
Step 1: Choose launch template
Edit
Group details
Auto Scaling group name
aws-prod-project
Launch template
Launch template
aws-prod-project
lt-0a749876eacfd1ea1
Version
Default
Description
my first vpc project
Step 2: Choose instance launch options
Edit
Network
Network
VPC
vpc-0251c30ae6d5b8815
Availability Zone
Subnet
us-east-1a
subnet-0e2602f6d80fd36e4
10.0.128.0/20
us-east-1b
subnet-07b8120636fa02aec
10.0.144.0/20
Instance type requirements
This Auto Scaling group will adhere to the launch template.
Step 3: Configure advanced options
Edit
Load balancing
Load balancer
-
VPC Lattice integration options
VPC Lattice target groups
-
Health checks
Health check type
EC2
Health check grace period
300 seconds
Additional settings
Monitoring
Disabled
Default instance warmup
Disabled
Step 4: Configure group size and scaling policies
Edit
Group size
Desired capacity
2
Desired capacity type
Units (number of instances)
Scaling
Minimum desired capacity
1
Maximum desired capacity
1
Target tracking policy
-
Instance maintenance policy
Replacement behavior
No policy
Min healthy percentage
-
Max healthy percentage
-
Instance scale-in protection
Instance scale-in protection
Enable instance protection from scale in
Step 5: Add notifications
Edit
Notifications
No notifications
Step 6: Add tags
Edit
Tags (0)
Key
Value
Tag new instances
No tags
Cancel
Previous
Create Auto Scaling group
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

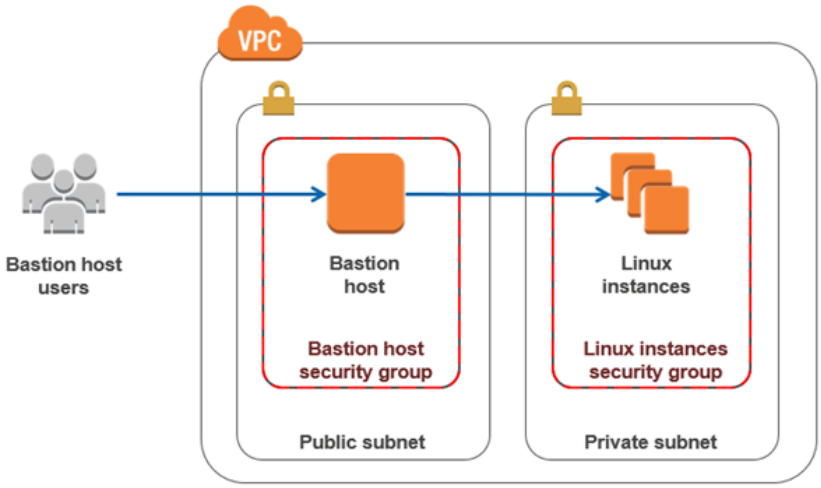
Here you can see the AGS is created.



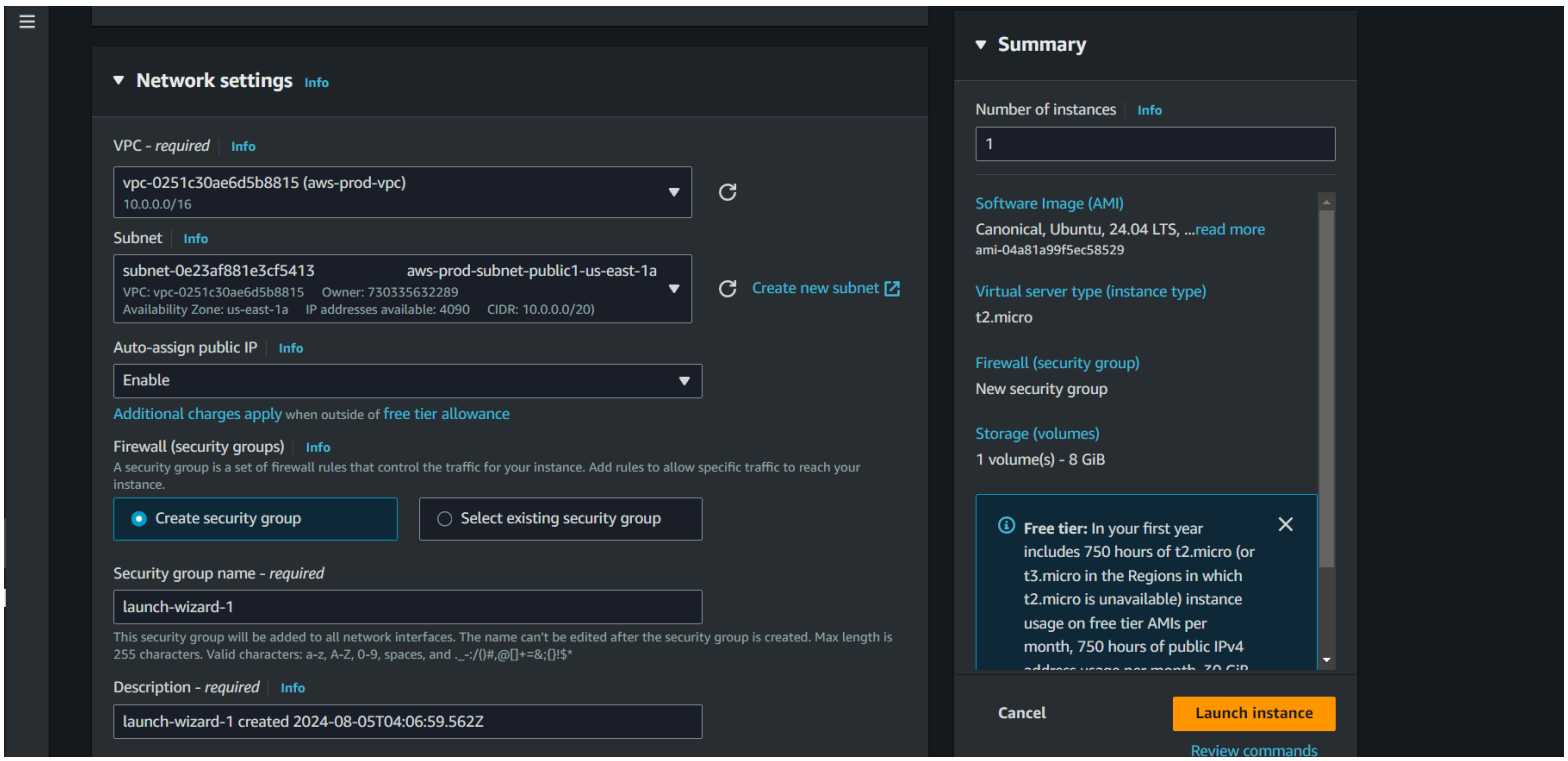


- Why do we need to create a Bastion Host ?

➔ An AWS bastion host is a secure computer that acts as a gateway to access other resources within a virtual private cloud (VPC). It's designed to be the SSH entrypoint for a VPC, and it can provide secure access to Linux instances in both private and public subnets.



- Here we are creating a bastion host so that we can connect to our private Ec2 instance, we have to do network settings so that this bastion host should be inside our VPC publicly available .



- For accessing bastion host i copied my .pem to my WSL

```
himanshu@DESKTOP-LVJ61KE: ~

┌──(himanshu㉿ DESKTOP-LVJ61KE)-[~]
└─$ cp /mnt/d/AWS_RSA_KEY/aws-login.pem ~/

┌──(himanshu㉿ DESKTOP-LVJ61KE)-[~]
└─$ ls
aws  aws-login.pem  awsStatus.sh  awsStatus.txt  first-shell-script.sh  linux.txt

┌──(himanshu㉿ DESKTOP-LVJ61KE)-[~]
└─$
```

I got an error here when i am copy .pem file using scp to bastion host,

- Actually I didn't give the correct permission to the .pem file so I gave it .

```
┌──(himanshu㉿ DESKTOP-LVJ61KE)-[~]
└─$ scp -i aws-login.pem aws-login.pem ubuntu@44.195.77.144:/home/ubuntu
The authenticity of host '44.195.77.144 (44.195.77.144)' can't be established.
ED25519 key fingerprint is SHA256:hlZp8KLKDCiO6JNCG/VRn/CVo0cYOOZxNLcw9maz9xo.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '44.195.77.144' (ED25519) to the list of known hosts.
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@         WARNING: UNPROTECTED PRIVATE KEY FILE!          @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions 0755 for 'aws-login.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "aws-login.pem": bad permissions
ubuntu@44.195.77.144: Permission denied (publickey).
scp: Connection closed
```

- If you get this error first run chmod 400 "aws-login.pem" this for the permission.
- After that I transferred .pem file to my bastion host so that i can connect to the private instance with that .pem file.

```
┌──(himanshu㉿ DESKTOP-LVJ61KE)-[~]
└─$ scp -i aws-login.pem aws-login.pem ubuntu@44.195.77.144:/home/ubuntu
aws-login.pem                                          100% 1678     7.3KB/s   00:00
```

- Login into the the bastion host machine

```
┌──(himanshu DESKTOP-LVJ61KE)-[~]
└─$ ssh -i aws-login.pem ubuntu@44.195.77.144
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1009-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Mon Aug  5 04:36:33 UTC 2024

  System load:  0.0               Processes:             105
  Usage of /:   22.9% of 6.71GB   Users logged in:       0
  Memory usage: 19%               IPv4 address for enX0: 10.0.1.60
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Mon Aug  5 04:34:44 2024 from 14.143.179.90
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-10-0-1-60:~$
```

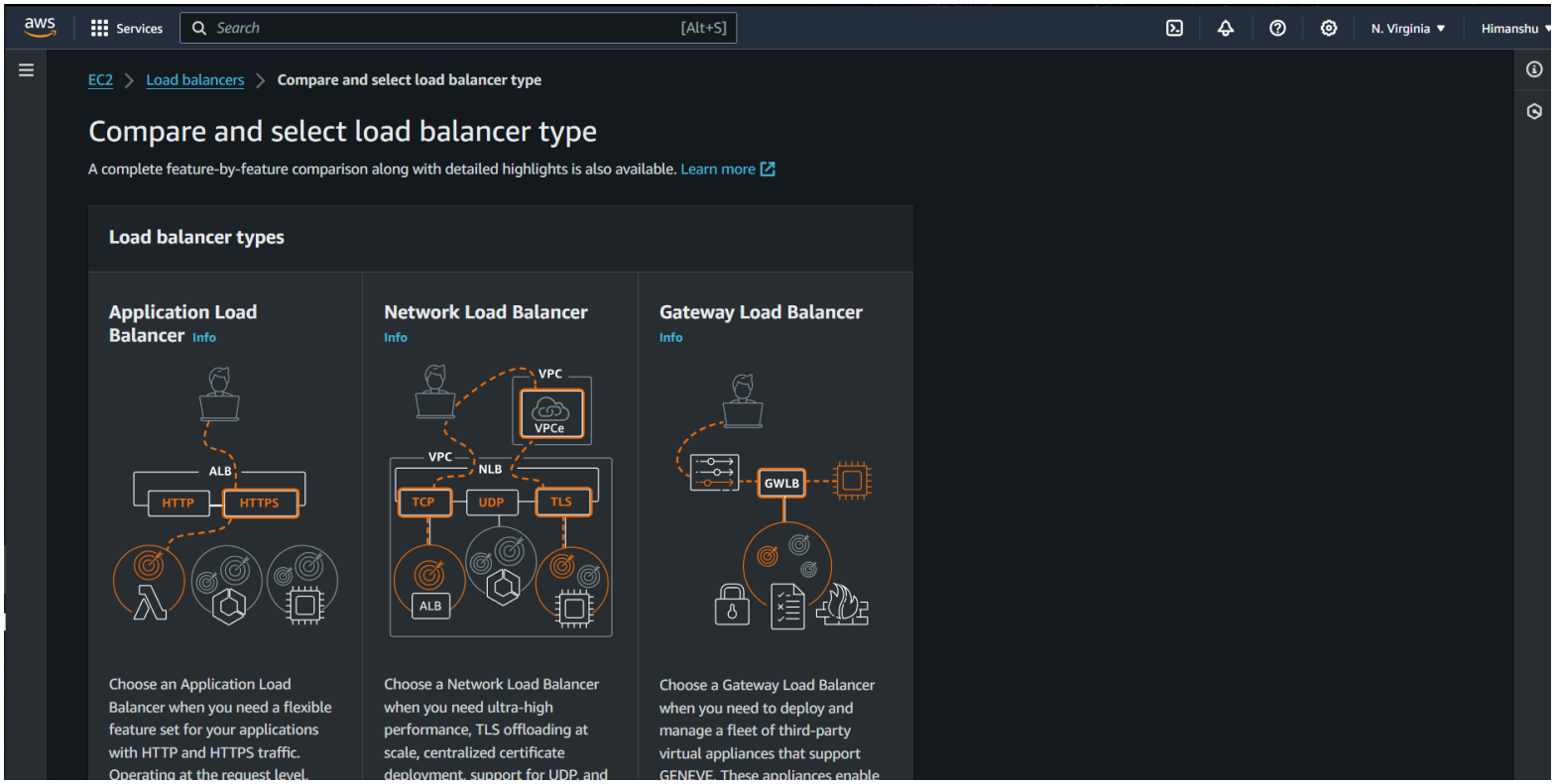
- After login to bastion host did ssh to ec2 instance in private subnet

```
ubuntu@ip-10-0-1-60:~$ ssh -i aws-login.pem ubuntu@10.0.153.11
The authenticity of host '10.0.153.11 (10.0.153.11)' can't be established.
ED25519 key fingerprint is SHA256:33dnzrxLo9qCwQD+Q6cKnpyuHwoKnEIhqIgsZs/5GRY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.0.153.11' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1009-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Mon Aug  5 04:38:27 UTC 2024

  System load:  0.0               Processes:             105
  Usage of /:   22.8% of 6.71GB   Users logged in:       0
  Memory usage: 30%               IPv4 address for enX0: 10.0.153.11
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-10-0-153-11:~$
```

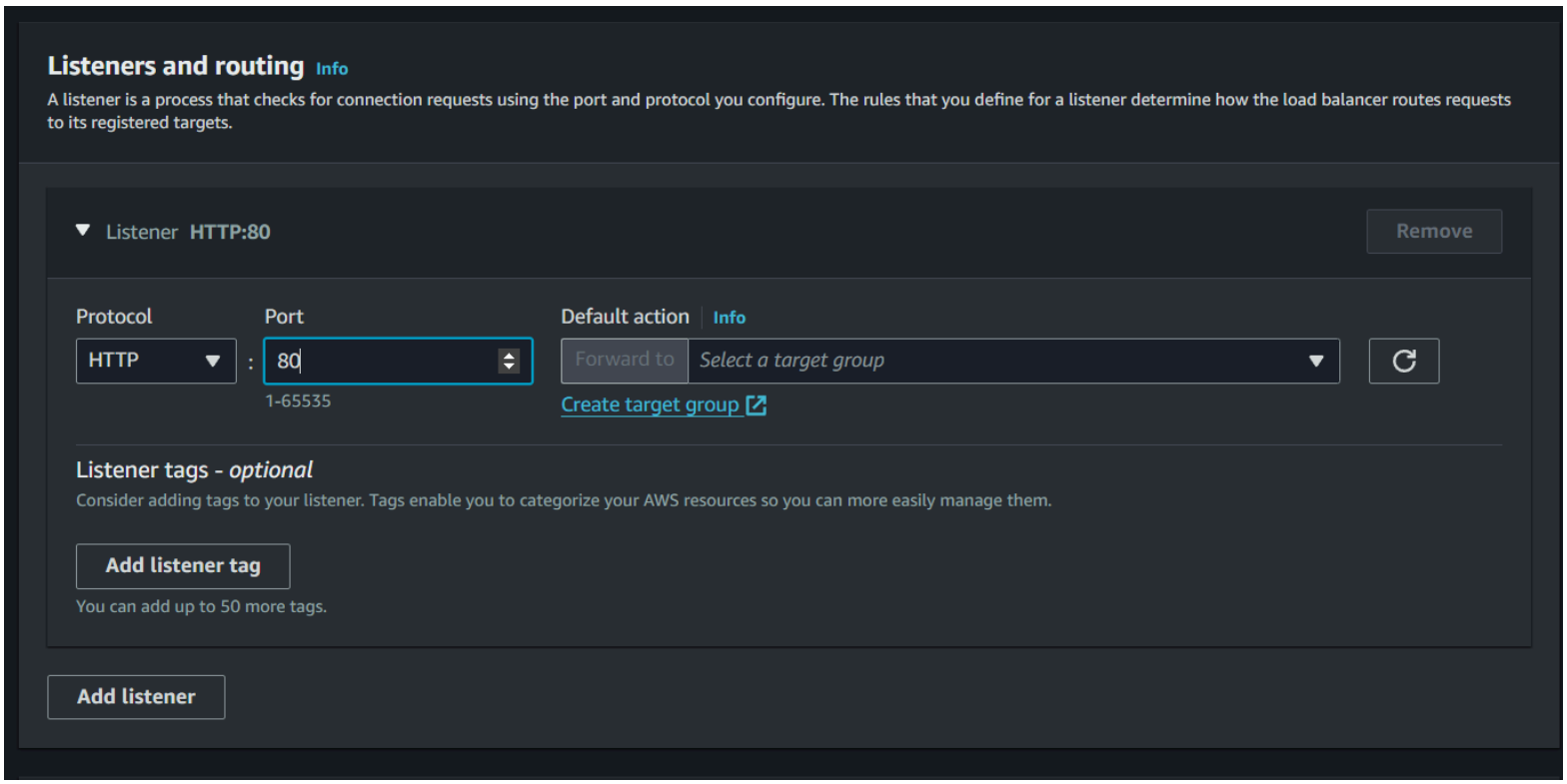
- After this i made a index.html file and run the python server

```
ubuntu@ip-10-0-153-11:~$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

We didn’t created the load balancer After this we need to create load balancer

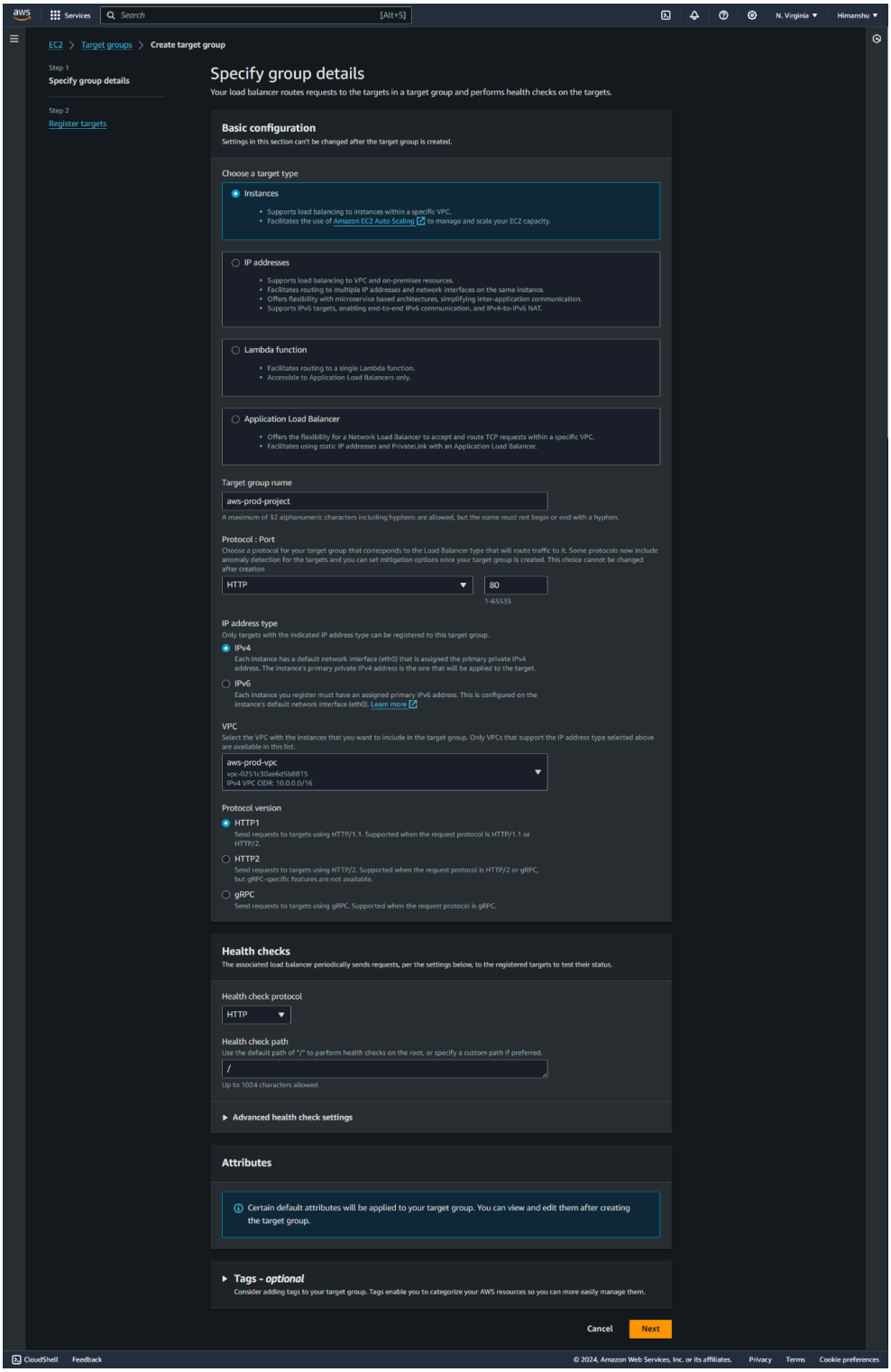


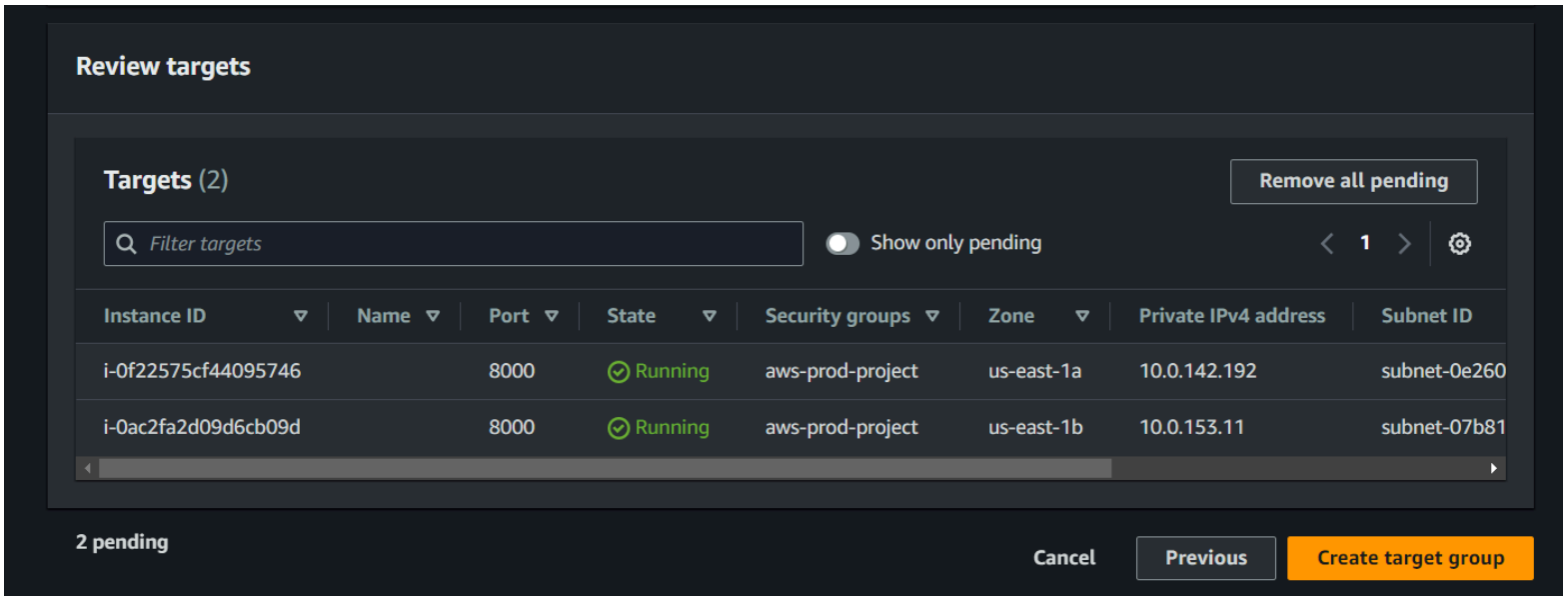
And will go with ALB(Application load Balancer), basically it is OSI Layer 7 load balancer which mostly works on HTTP and HTTPS.

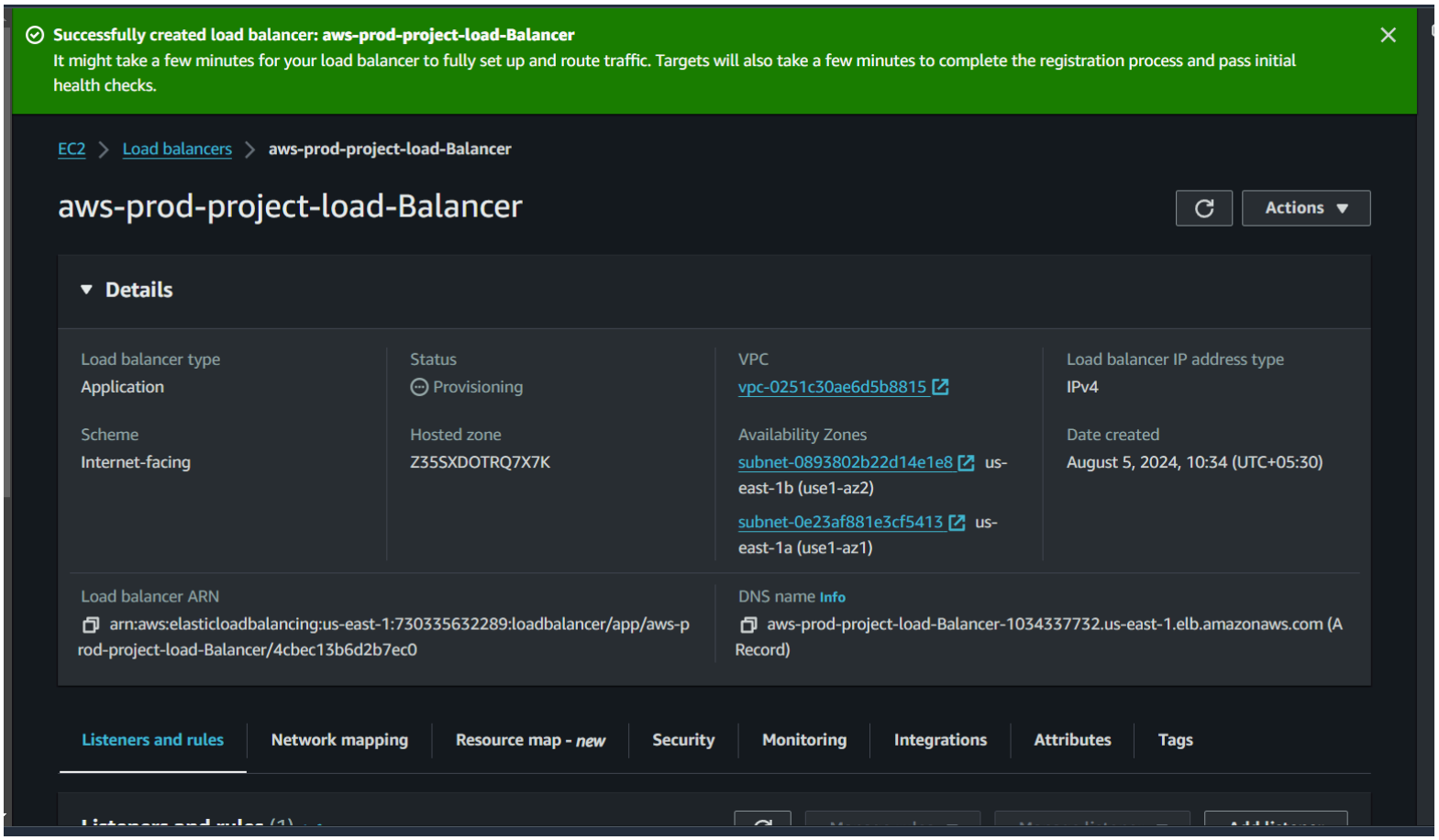


- What is Target Group ?

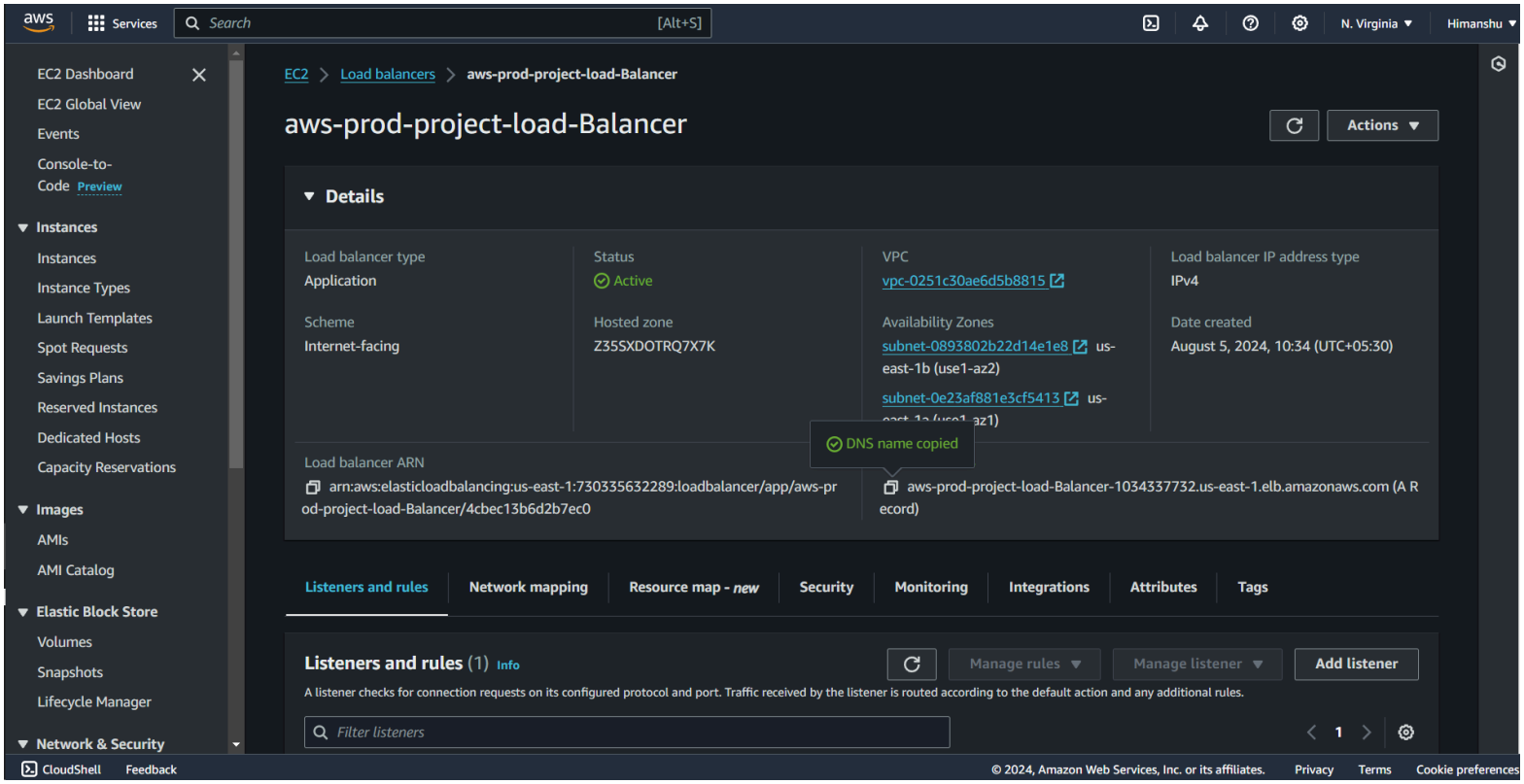
➔ Target groups route requests to individual registered targets, such as EC2 instances, using the protocol and port number that you specify. You can register a target with multiple target groups. You can configure health checks on a per target group basis. Health checks are performed on all targets registered to a target group that is specified in a listener rule for your load balancer.



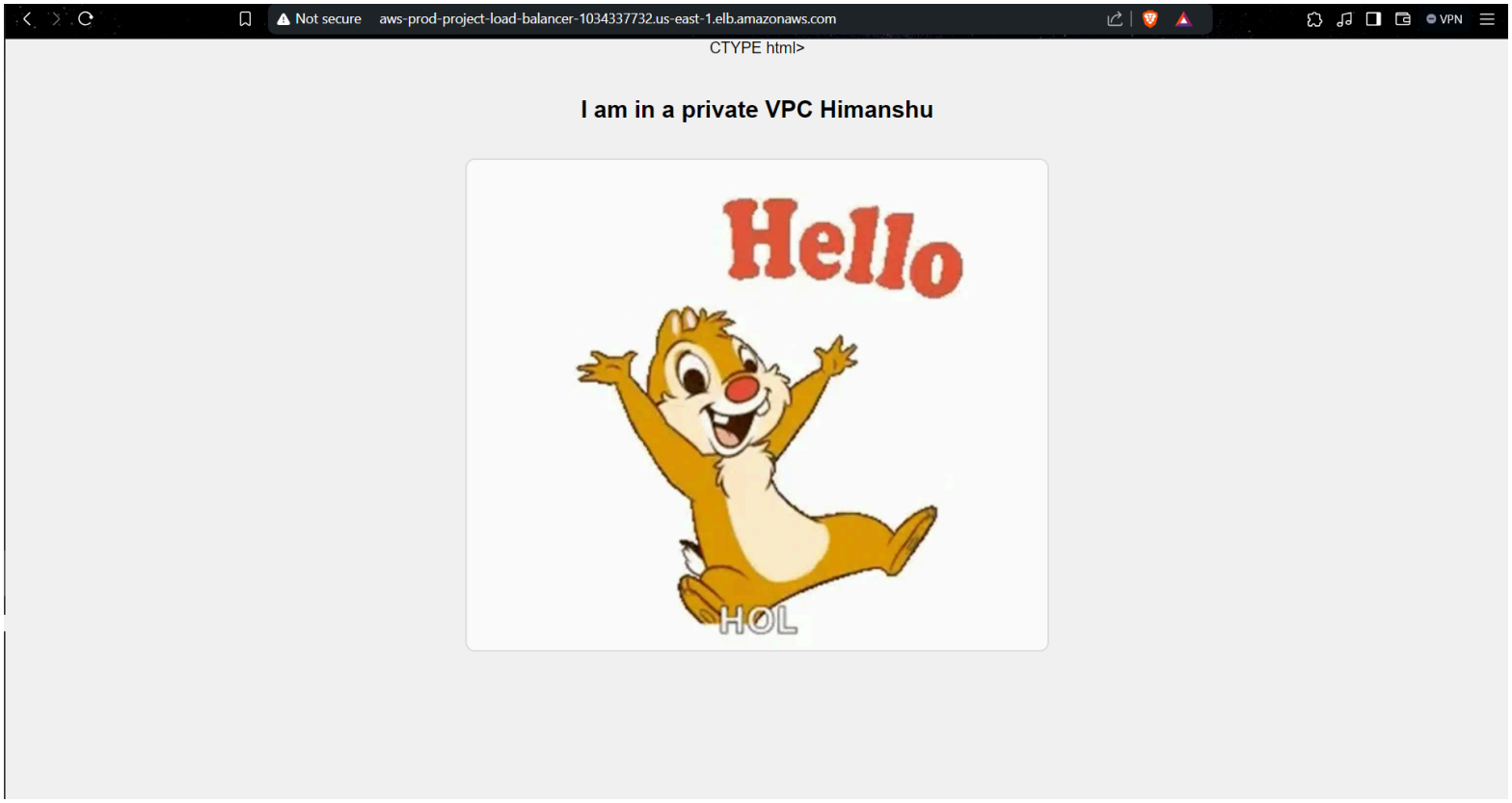

Review targets
Targets (2)
Remove all pending
Filter targets
Show only pending
1
Instance ID
Name
Port
State
Security groups
Zone
Private IPv4 address
Subnet ID
i-0f22575cf44095746
8000
Running
aws-prod-project
us-east-1a
10.0.142.192
subnet-0e260
i-0ac2fa2d09d6cb09d
8000
Running
aws-prod-project
us-east-1b
10.0.153.11
subnet-07b81
2 pending
Cancel
Previous
Create target group


Successfully created load balancer: aws-prod-project-load-Balancer
It might take a few minutes for your load balancer to fully set up and route traffic. Targets will also take a few minutes to complete the registration process and pass initial health checks.
EC2
Load balancers
aws-prod-project-load-Balancer
aws-prod-project-load-Balancer
Actions
Details
Load balancer type
Application
Status
Provisioning
VPC
vpc-0251c30ae6d5b8815
Load balancer IP address type
IPv4
Scheme
Internet-facing
Hosted zone
Z35SXDOTRQ7X7K
Availability Zones
subnet-0893802b22d14e1e8 us-east-1b (use1-az2)
subnet-0e23af881e3cf5413 us-east-1a (use1-az1)
Date created
August 5, 2024, 10:34 (UTC+05:30)
Load balancer ARN
arn:aws:elasticloadbalancing:us-east-1:730335632289:loadbalancer/app/aws-prod-project-load-Balancer/4cbec13b6d2b7ec0
DNS name Info
aws-prod-project-load-Balancer-1034337732.us-east-1.elb.amazonaws.com (A Record)
Listeners and rules
Network mapping
Resource map - new
Security
Monitoring
Integrations
Attributes
Tags

Load balancer is created with target group



- So load balance is ready now and I also opened port 80 in the security group and now we can see the application is running .

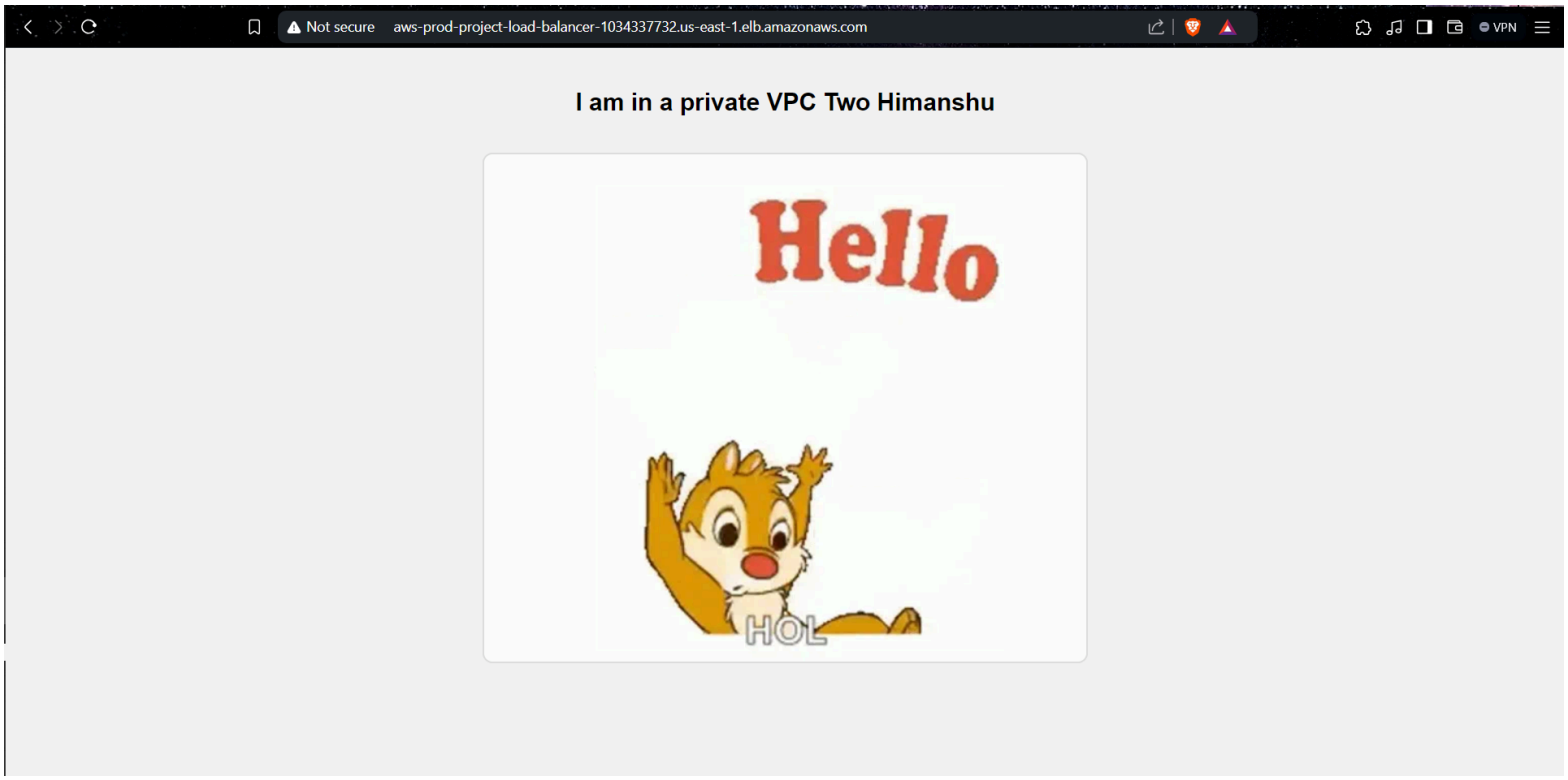


```
ubuntu@ip-10-0-153-11:~$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.0.16.65 - - [05/Aug/2024 05:06:33] "GET / HTTP/1.1" 200 -
10.0.14.70 - - [05/Aug/2024 05:06:48] "GET / HTTP/1.1" 200 -
10.0.16.65 - - [05/Aug/2024 05:07:03] "GET / HTTP/1.1" 200 -
10.0.14.70 - - [05/Aug/2024 05:07:18] "GET / HTTP/1.1" 200 -
10.0.16.65 - - [05/Aug/2024 05:07:33] "GET / HTTP/1.1" 200 -
10.0.14.70 - - [05/Aug/2024 05:07:48] "GET / HTTP/1.1" 200 -
10.0.16.65 - - [05/Aug/2024 05:08:03] "GET / HTTP/1.1" 200 -
10.0.14.70 - - [05/Aug/2024 05:08:18] "GET / HTTP/1.1" 200 -
10.0.16.65 - - [05/Aug/2024 05:08:33] "GET / HTTP/1.1" 200 -
10.0.14.70 - - [05/Aug/2024 05:08:48] "GET / HTTP/1.1" 200 -
10.0.16.65 - - [05/Aug/2024 05:09:03] "GET / HTTP/1.1" 200 -
10.0.16.65 - - [05/Aug/2024 05:09:11] "GET / HTTP/1.1" 200 -
10.0.16.65 - - [05/Aug/2024 05:09:12] code 404, message File not found
10.0.16.65 - - [05/Aug/2024 05:09:12] "GET /favicon.ico HTTP/1.1" 404 -
10.0.14.70 - - [05/Aug/2024 05:09:18] "GET / HTTP/1.1" 200 -
10.0.16.65 - - [05/Aug/2024 05:09:33] "GET / HTTP/1.1" 200 -
10.0.14.70 - - [05/Aug/2024 05:09:48] "GET / HTTP/1.1" 200 -
10.0.16.65 - - [05/Aug/2024 05:10:03] "GET / HTTP/1.1" 200 -
10.0.14.70 - - [05/Aug/2024 05:10:18] "GET / HTTP/1.1" 200 -
10.0.16.65 - - [05/Aug/2024 05:10:33] "GET / HTTP/1.1" 200 -
10.0.14.70 - - [05/Aug/2024 05:10:48] "GET / HTTP/1.1" 200 -
10.0.16.65 - - [05/Aug/2024 05:11:03] "GET / HTTP/1.1" 200 -
10.0.14.70 - - [05/Aug/2024 05:11:18] "GET / HTTP/1.1" 200 -
10.0.16.65 - - [05/Aug/2024 05:11:33] "GET / HTTP/1.1" 200 -
10.0.14.70 - - [05/Aug/2024 05:11:48] "GET / HTTP/1.1" 200 -
10.0.16.65 - - [05/Aug/2024 05:12:03] "GET / HTTP/1.1" 200 -
10.0.14.70 - - [05/Aug/2024 05:12:18] "GET / HTTP/1.1" 200 -
10.0.16.65 - - [05/Aug/2024 05:12:33] "GET / HTTP/1.1" 200 -
```

- So now what i did i ran the server in both EC2 in private VPC

```
ubuntu@ip-10-0-142-192:~$ vim index.html
ubuntu@ip-10-0-142-192:~$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.0.16.65 - - [05/Aug/2024 05:22:34] "GET / HTTP/1.1" 200 -
10.0.14.70 - - [05/Aug/2024 05:22:49] "GET / HTTP/1.1" 200 -
10.0.16.65 - - [05/Aug/2024 05:23:04] "GET / HTTP/1.1" 200 -
```

- We can see now load balance is transferring the traffic to both VPC one and VPC Two



This is step by step process to delete the every things which we created :

Clean-Up Steps:

1. Delete the Auto Scaling Group:
    - Remove the scaling configurations.

2. Delete the Load Balancer:
    - Remove the ALB you created.

3. Delete the Target Group:
    - Remove the associated target group.

4. Delete the NAT Gateway:
    - Remove the NAT gateway from your VPC.

5. Delete All EC2 Instances:
    - Terminate the three instances you created.

6. Release the Elastic IP:
    - Release any Elastic IP addresses associated with the instances.

7. Delete the VPC:
    - Remove the VPC and all associated resources