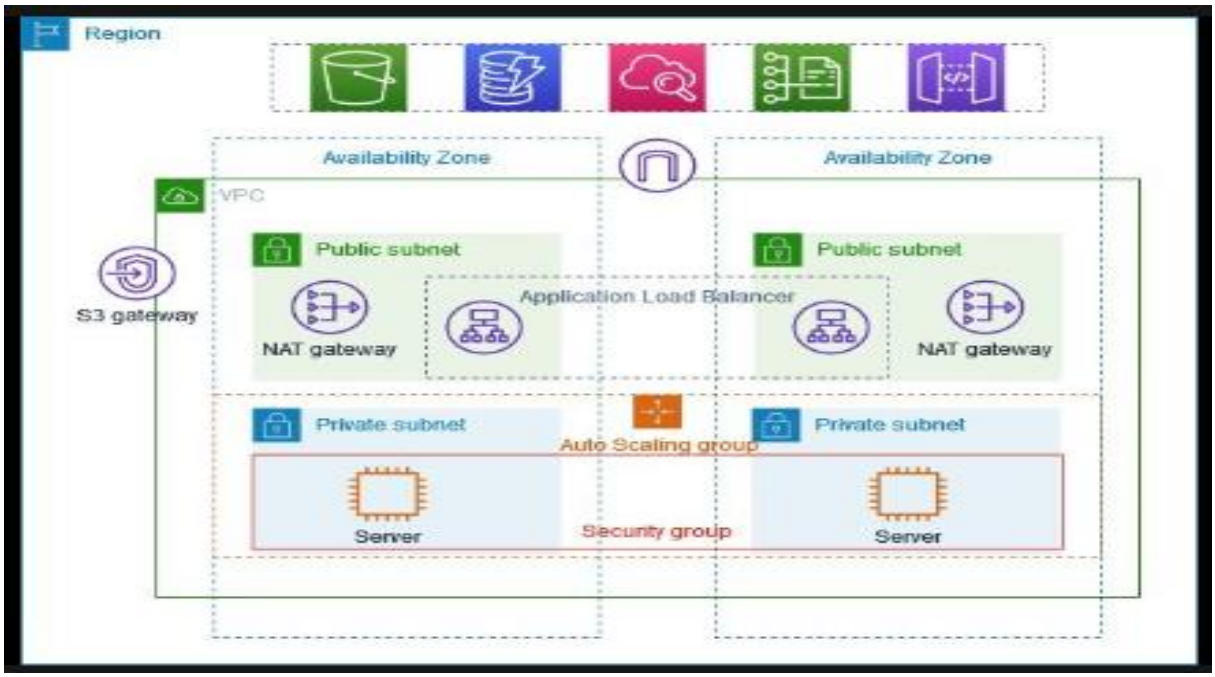
VPC With Public-Private Subnets in Production

About the Project:

This example demonstrates how to create a VPC that you can use for servers in a production environment. To improve resiliency, you deploy the servers in two Availability Zones, by using an Auto Scaling group and an Application Load Balancer. For additional security, you deploy the servers in private subnets. The servers receive requests through the load balancer. The servers can connect to the internet by using a NAT gateway. To improve resiliency, you deploy the NAT gateway in both Availability Zones.

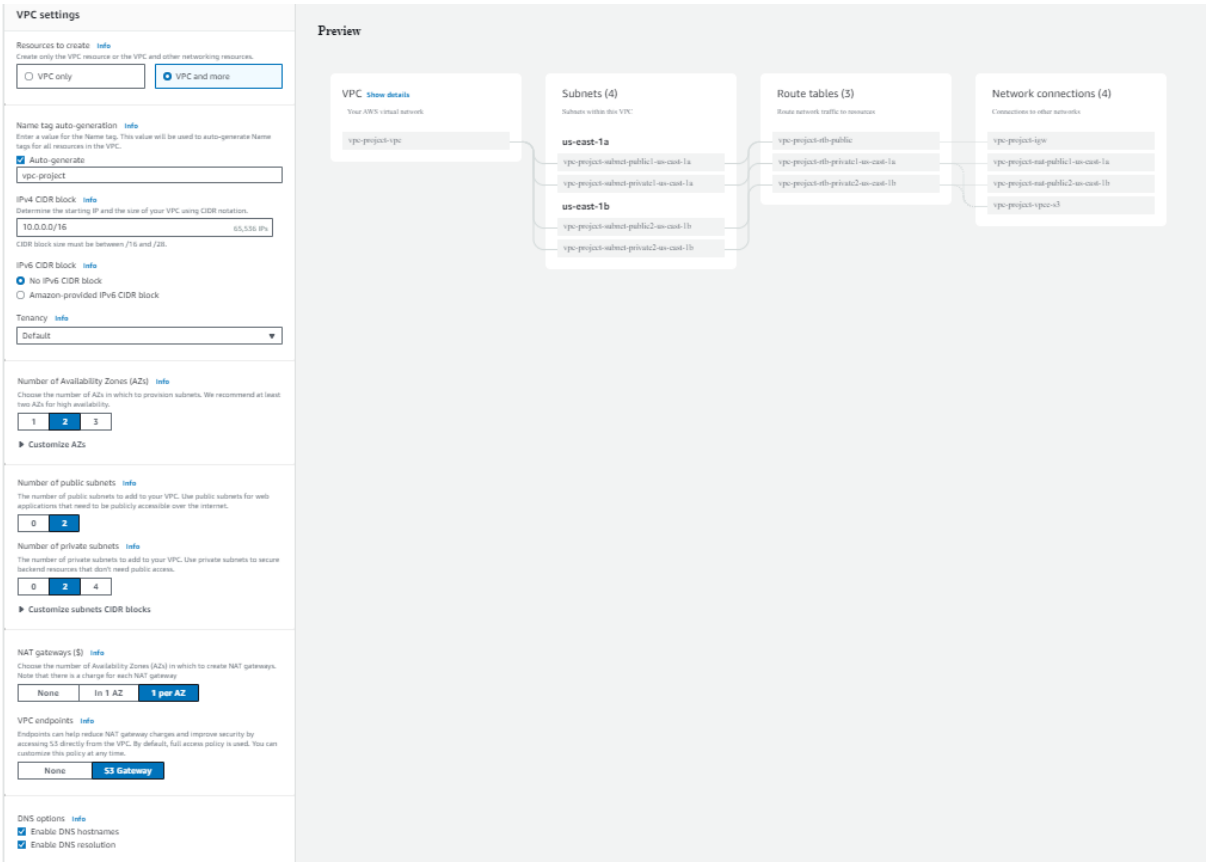
Overview:

The VPC has public subnets and private subnets in two Availability Zones. Each public subnet contains a NAT gateway and a load balancer node. The servers run in the private subnets, are launched and terminated by using an Auto Scaling group, and receive traffic from the load balancer. The servers can connect to the internet by using the NAT gateway.

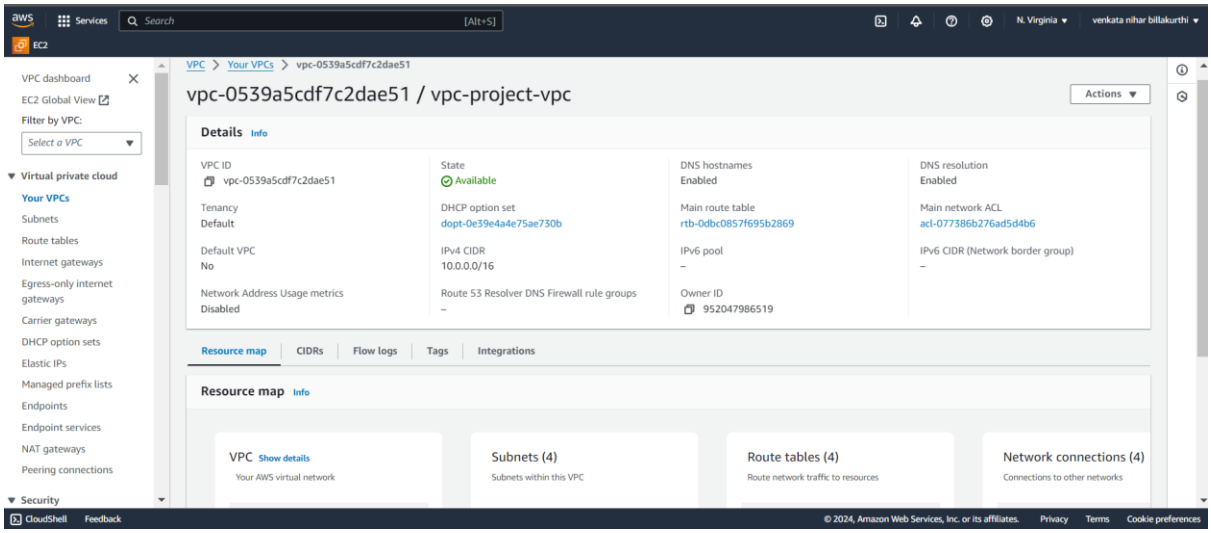


- Deployed servers in two Availability Zones using Auto Scaling group and Application Load Balancer for high availability.
- Configured VPC with 2 AZs, public/private subnets & Bastion host for secure server access.
- Enabled outbound internet connectivity via NAT Gateways for IP masking.
- Implemented Application Load Balancer for optimized performance & user experience.
- Ran application securely within VPC, ensuring compliance & protection.

First, Create the VPC -name it(vpc-project) and in NAT gatways select the 1per AZ

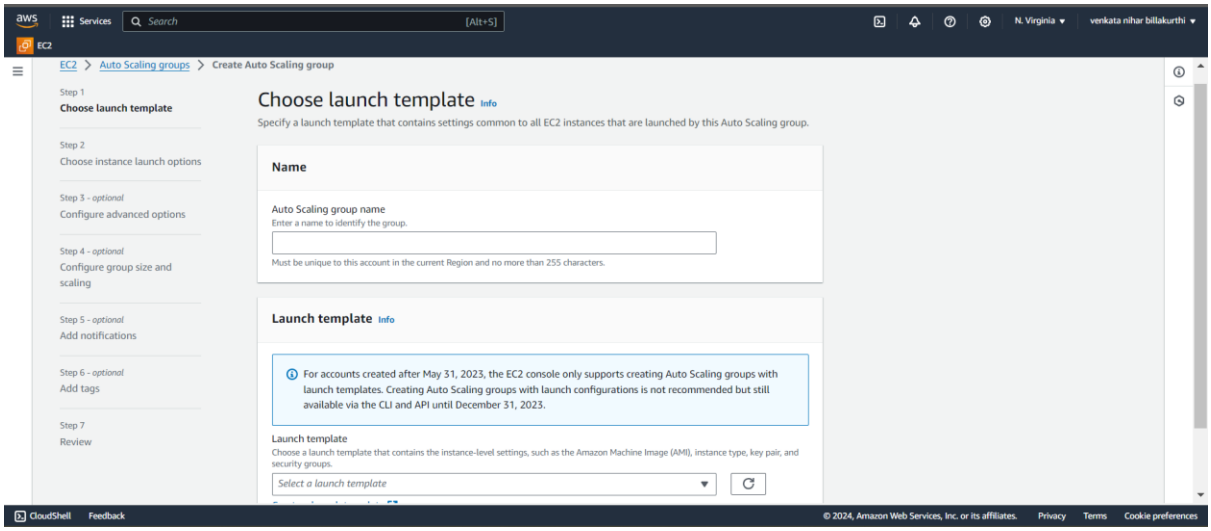


After click on the create it create the four subnets. Two are public and two are private subnets.

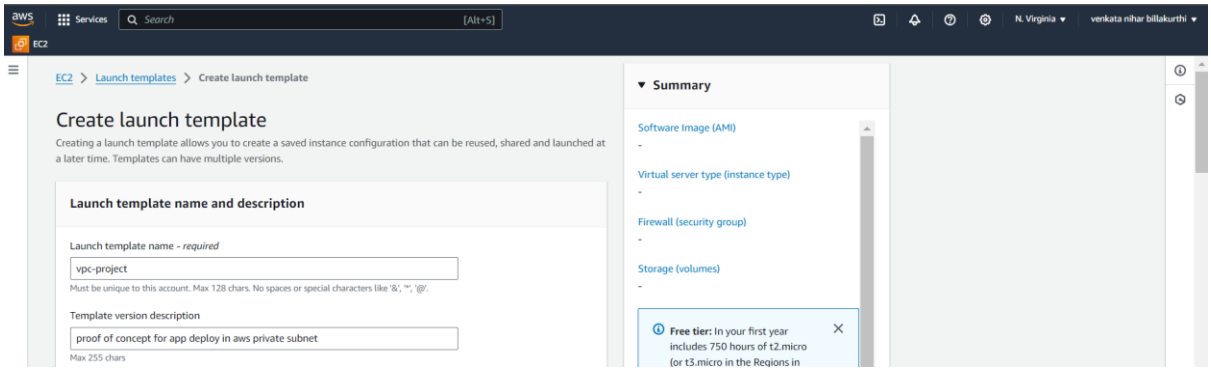


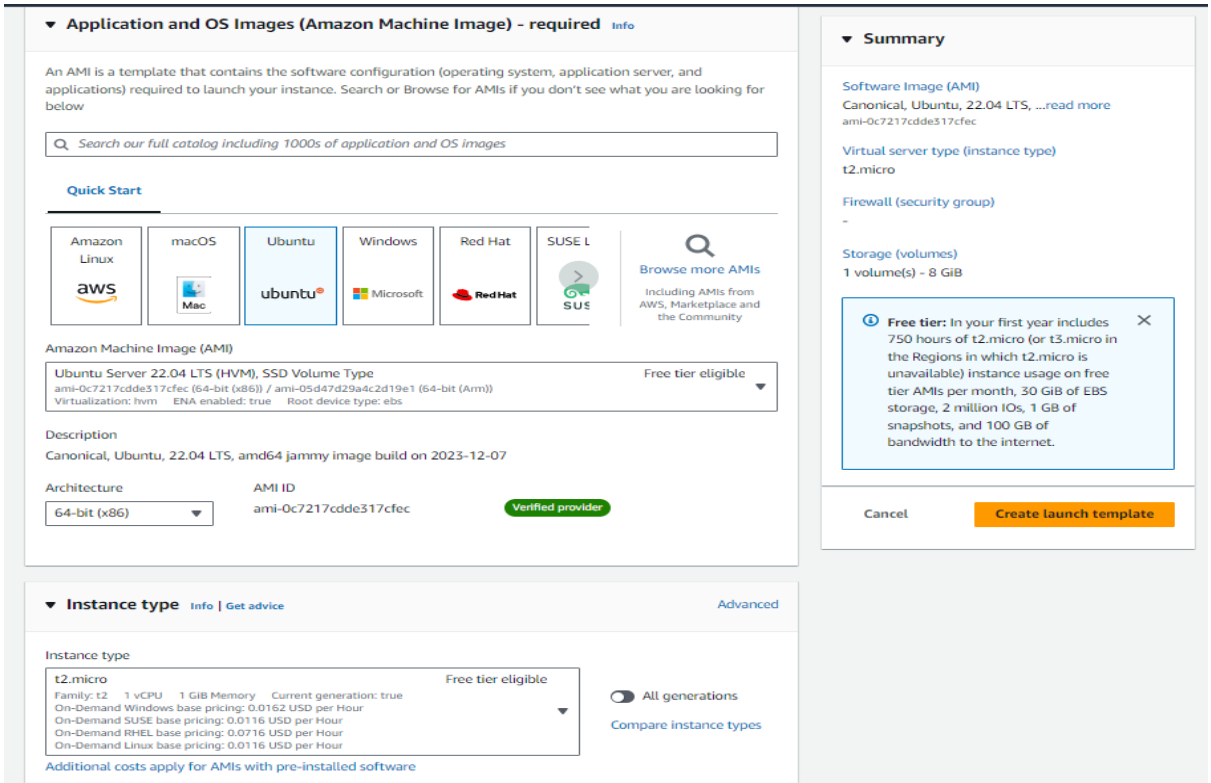
Here, vpc is created successfully.

After, then create the Auto Scaling groups – name them(VPC-project) and select our lanuch template.

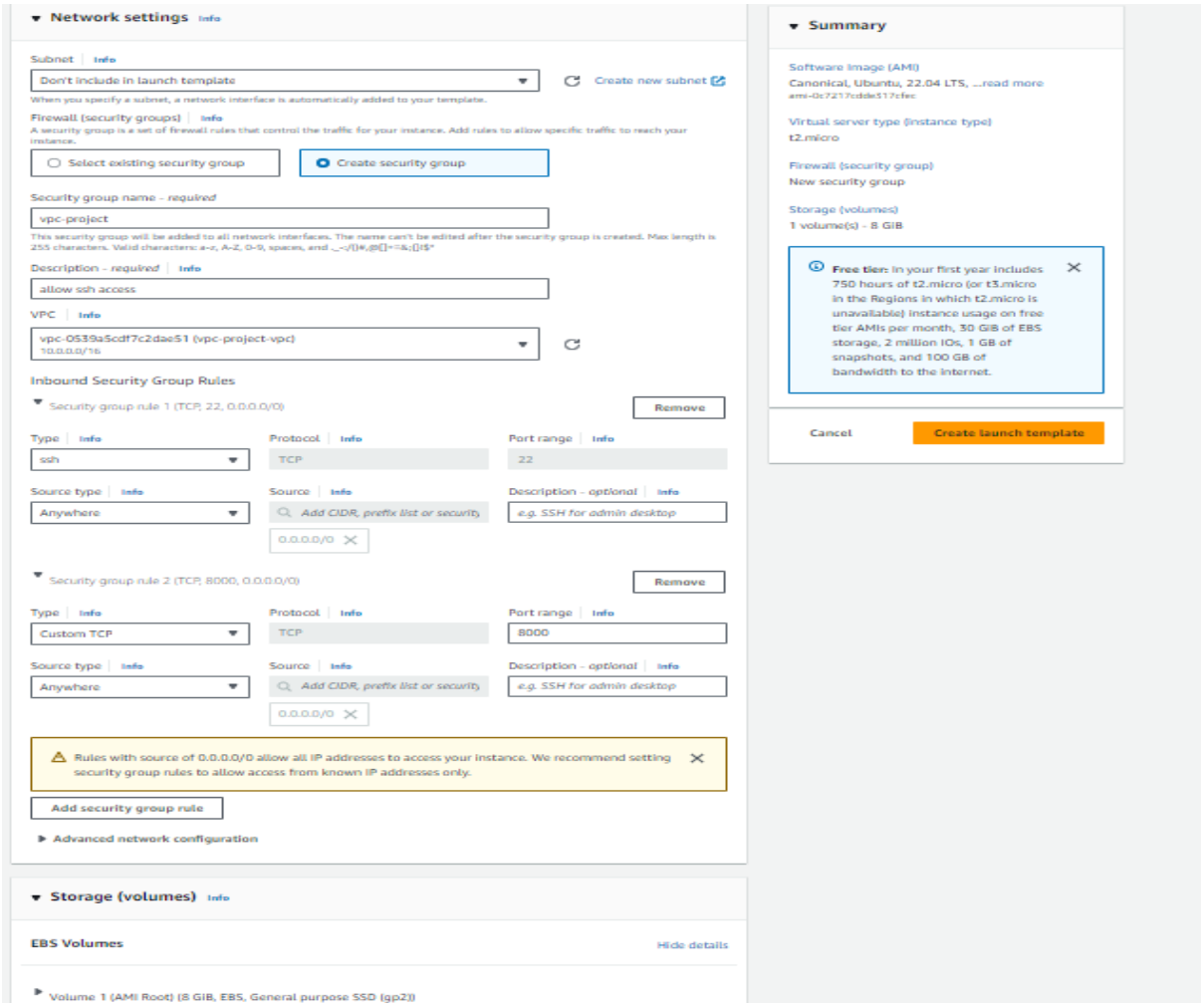


So create the launch template name it(vpc-project) and description (proof of concept for app deployment in aws private subnets) the name and description is our choice.

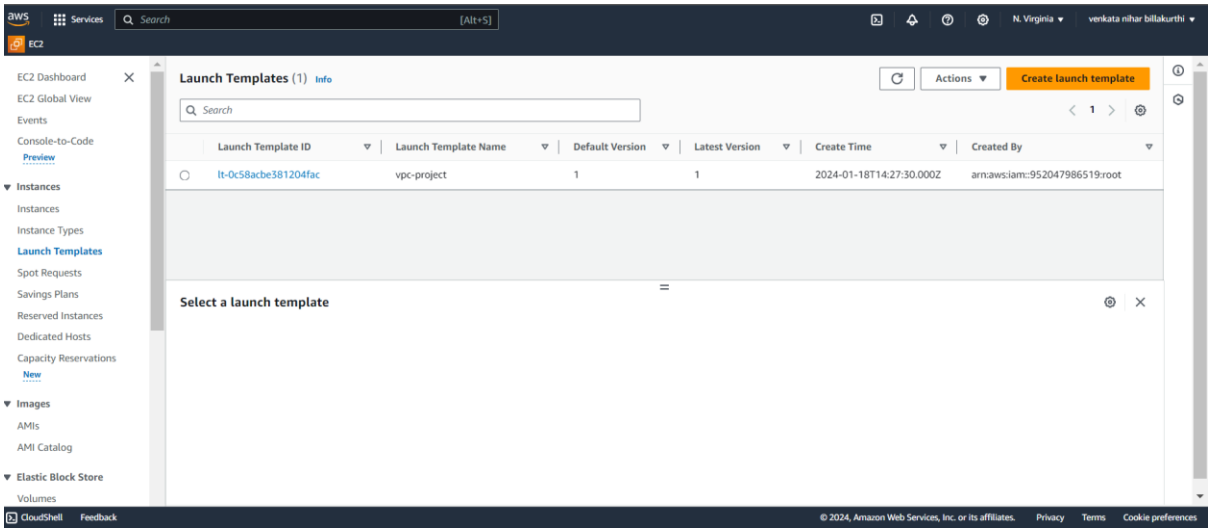




Add the custom TCP port:8000 in security group

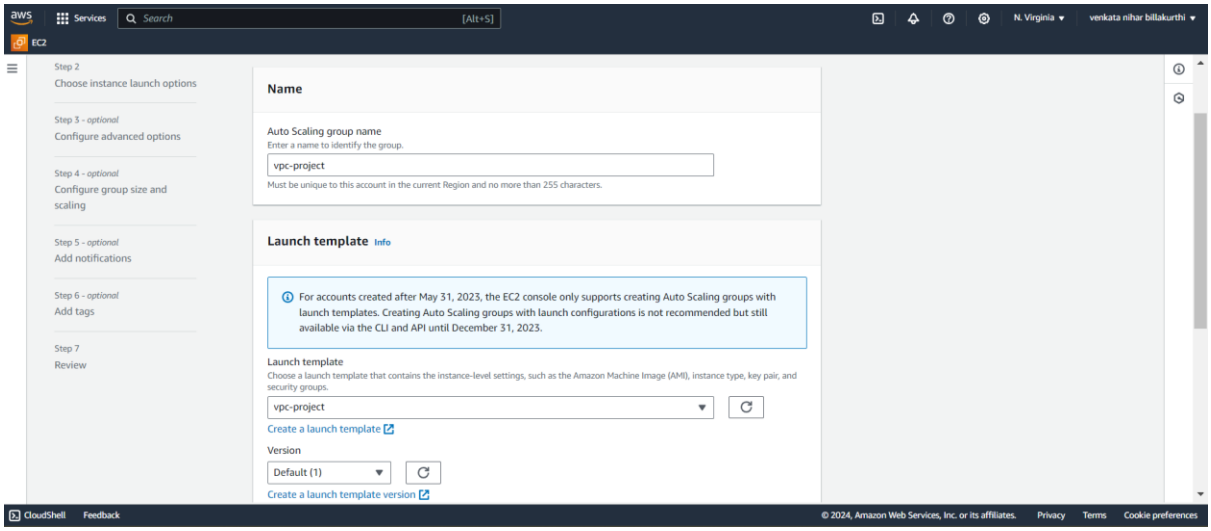


Then click on the create.

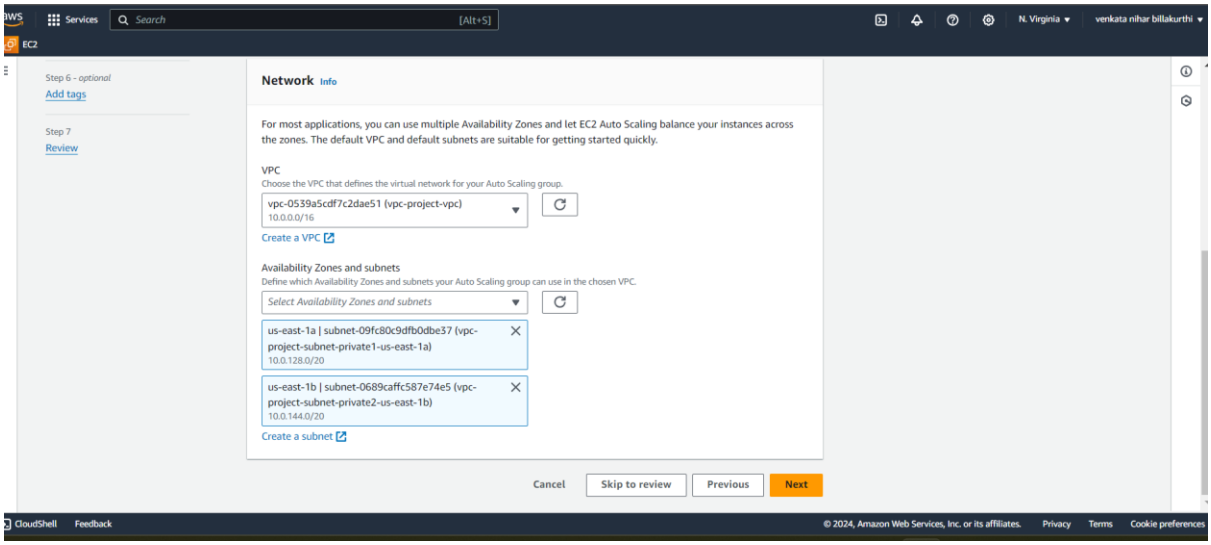


Here the launch template is created.

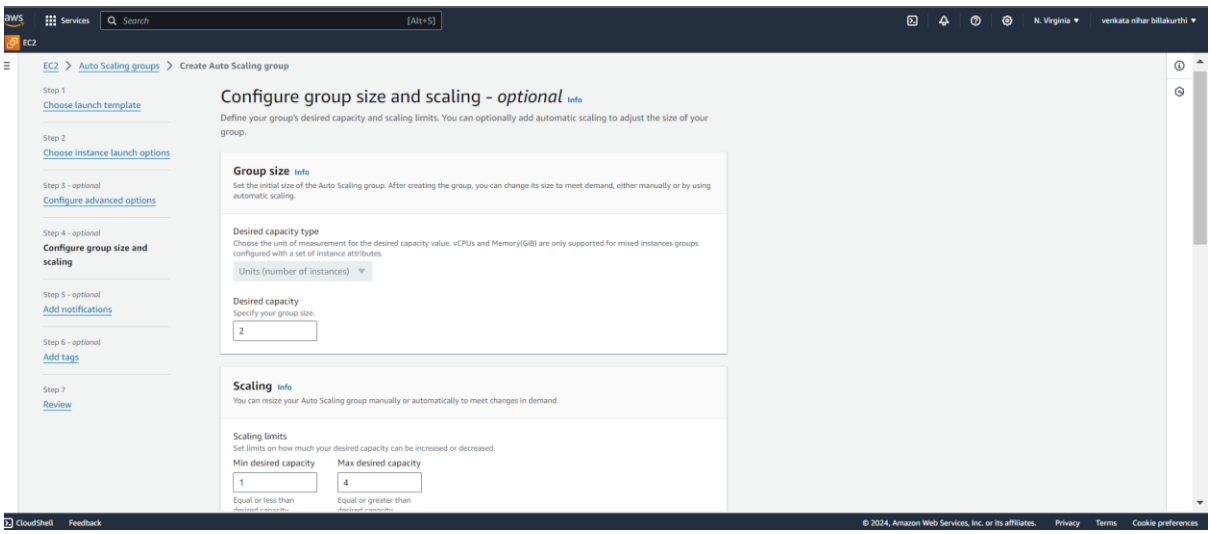
Now, select our launch template in auto scaling



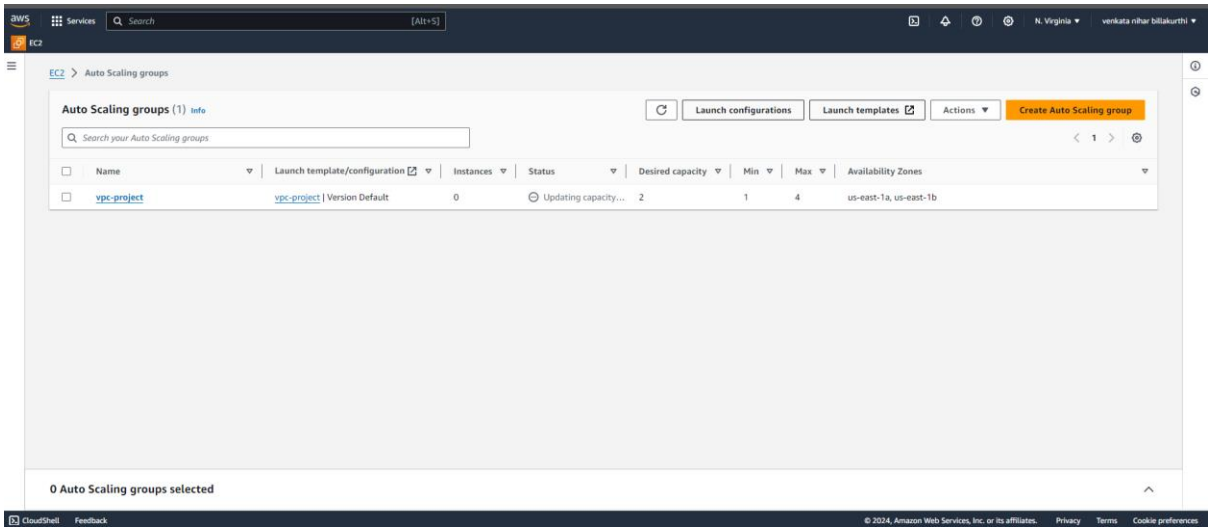
And select out vpc and private subnets also.

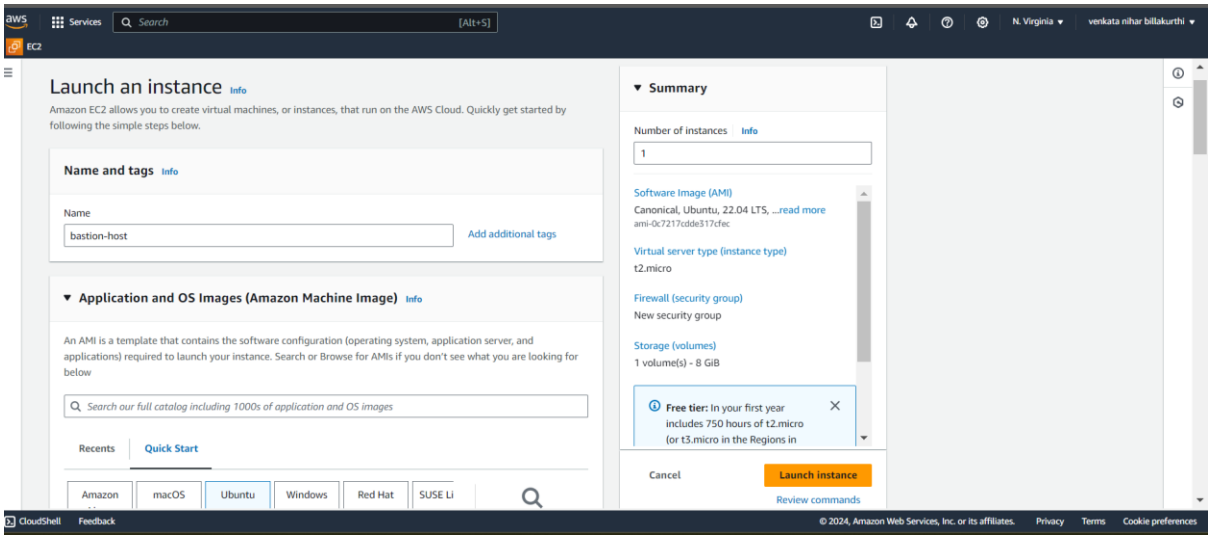


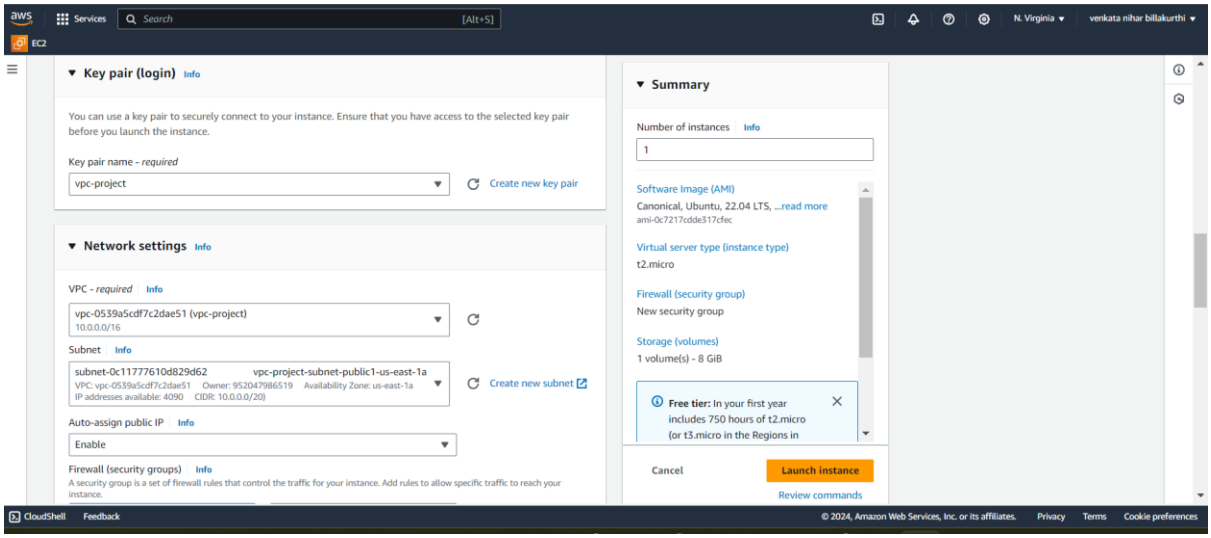
In the configure group size and scaling, Desired capacity is 2 and max is 4 this our choices.

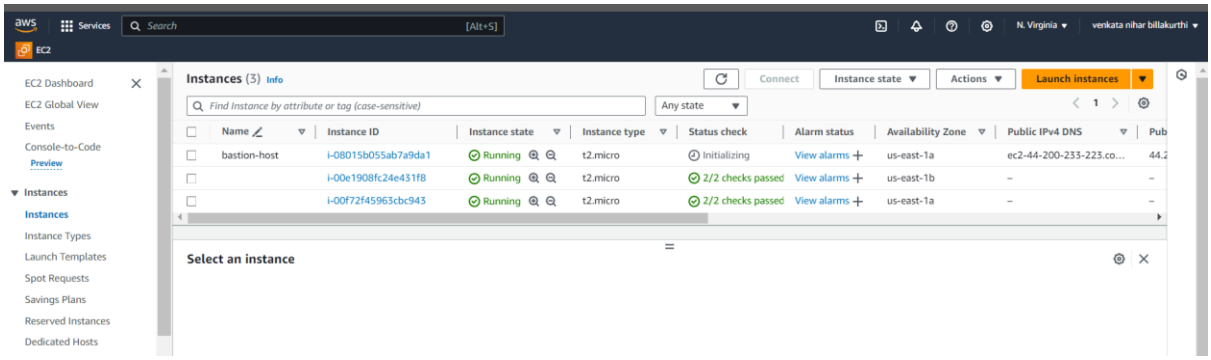


Here, the auto scaling group and two aws instances also created.










```
PS C:\Users\hp> cd downloads
PS C:\Users\hp\downloads> scp -i /Users/hp/Downloads/vpc-project.pem   /Users/hp/Downloads/vpc-project.pem ubuntu@44.200
.233.223:/home/ubuntu
vpc-project.pem                                                100% 1674     6.0KB/s   00:00
```

```
ubuntu@ip-10-0-8-134: ~

PS C:\Users\hp\downloads> ssh -i vpc-project.pem ubuntu@44.200.233.223
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 6.2.0-1017-aws x86_64)

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

  System information as of Thu Jan 25 05:45:22 UTC 2024

  System load:  0.0               Processes:             98
  Usage of /:   20.8% of 7.57GB   Users logged in:       0
  Memory usage: 21%               IPv4 address for eth0: 10.0.8.134
  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

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

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

ubuntu@ip-10-0-8-134:~$ ls
vpc-project.pem
ubuntu@ip-10-0-8-134:~$
```

Now, we try to login in private instances.

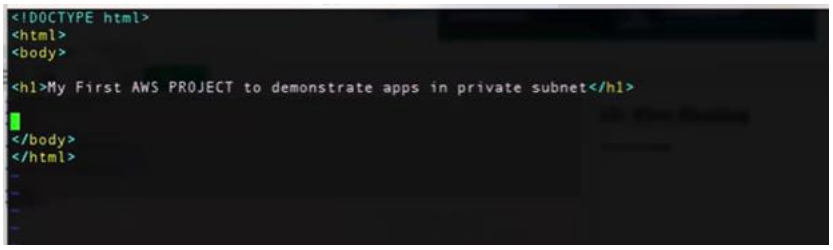
Then, copy the private IP Address of a one instances

- ➔ Ssh -i (passkey file name) ubuntu@privateIP

Then, we can able to login to the private instances.

Now, create a file

- ➔ Vim index.html
- ➔ Enter the sample code and save it

➔ 

➔ Python3 -m http.server 8000
It is running now.

Now, go to EC2
-> Load Balancers
-> Application l.b
-> Select the our Vpc
-> Select the both subnets
-> Select the our security group
-> Listeners & routing – create target group -name it and port 8000,next -select the private instances -click on include as pending below -create.
-> Select the our target group
-> create.

Then go to security group -> edit inbound rules -> HTTP/anywhere ->create

Go to load balancer >>>copy the (DNS) and paste it on browser.