AWS Project-VPC with public-private subnet in Production

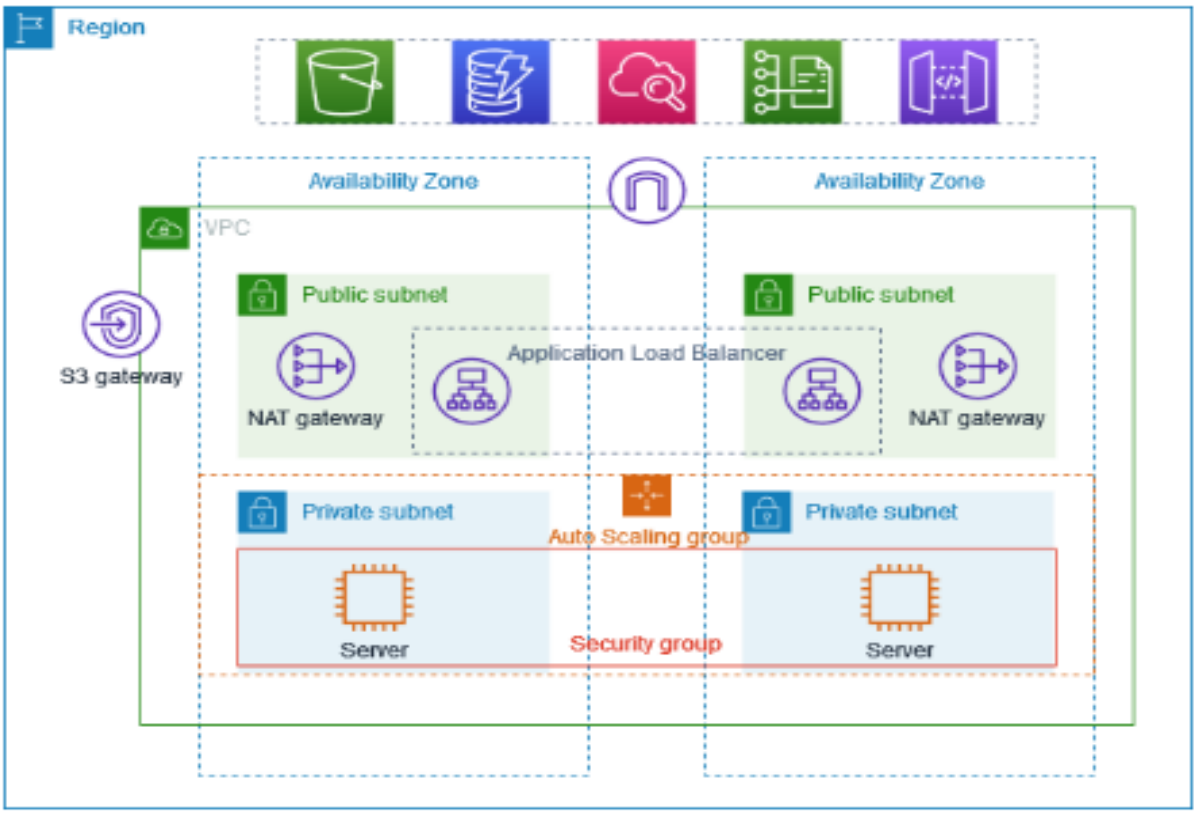
This example demonstrates how to create a VPC that you can use for servers in a production environment. To improve resiliency, you deploy the servers in two Availability Zones, by using an Auto Scaling group and an Application Load Balancer. For additional security, you deploy the servers in private 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.

Contents

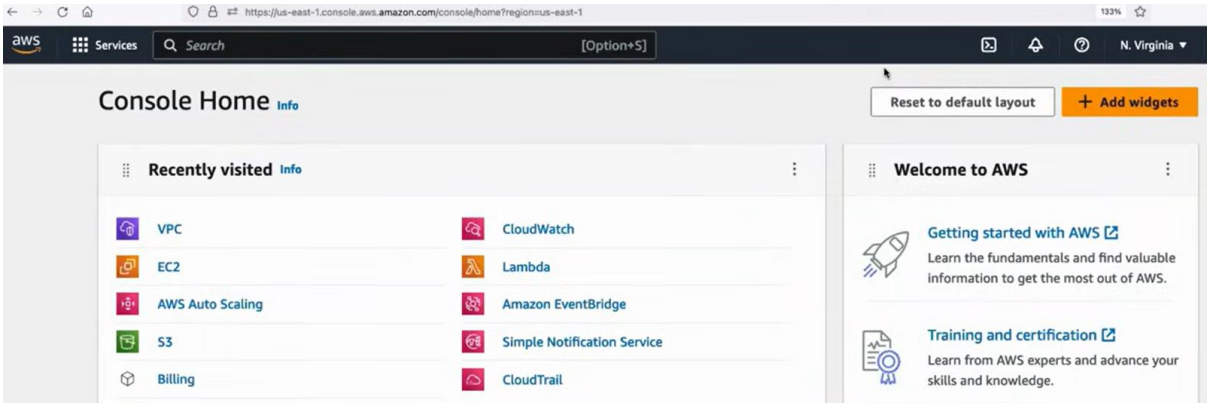
Overview:

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 NAT gateway.

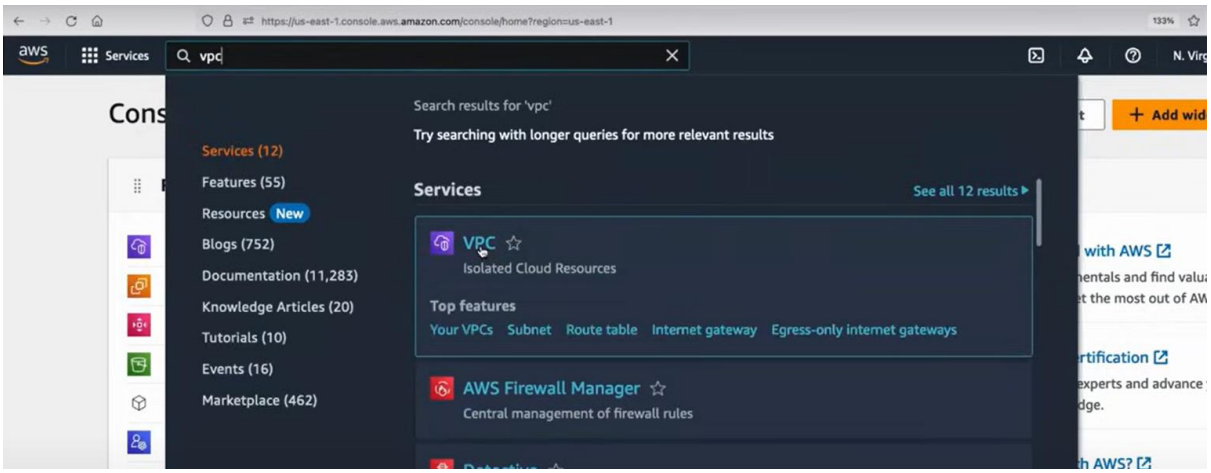
The following diagram provides an overview of the resources included in this example. The VPC has public subnets and private subnets in two Availability Zones. Each public subnet contains a NAT gateway and a load balancer node. The servers run in the private subnets, are launched and terminated by using an Auto Scaling group, and receive traffic from the load balancer. The servers can connect to the internet by using the NAT gateway. The servers can connect to Amazon S3 by using a gateway VPC endpoint.

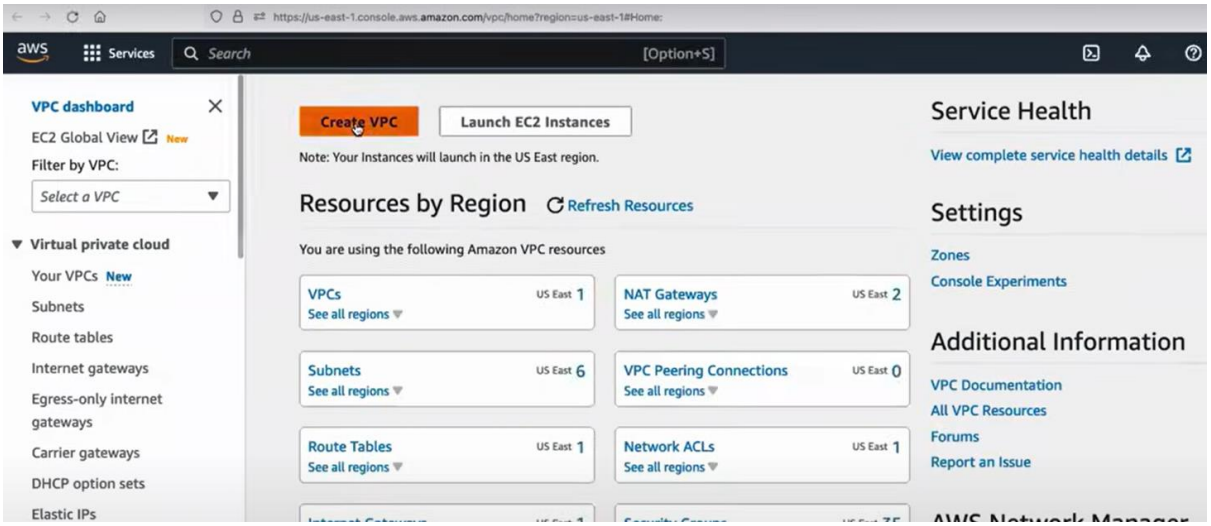


1. Go to Aws console and search for VPC

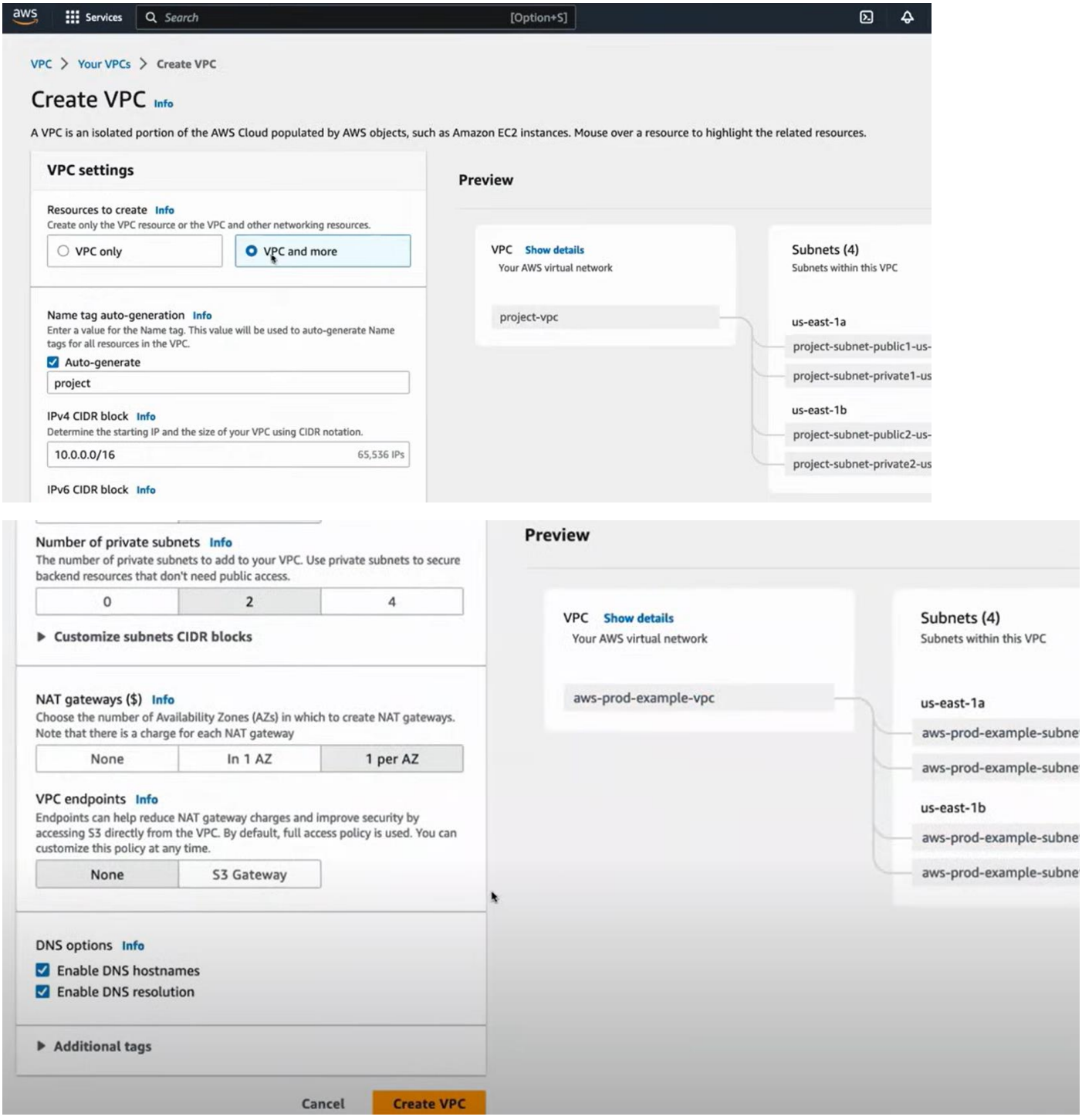


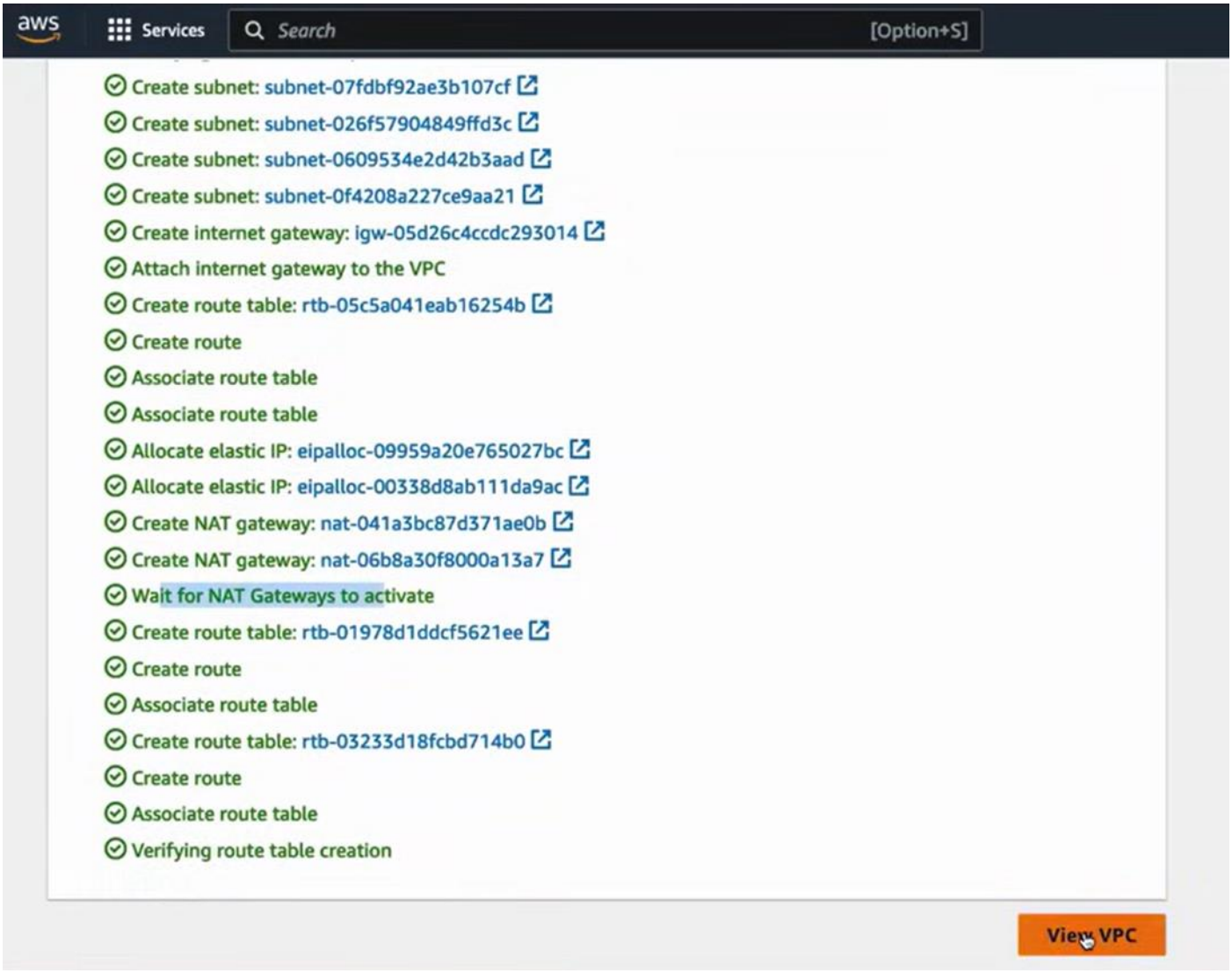
2. Select the VPC and create VPC.



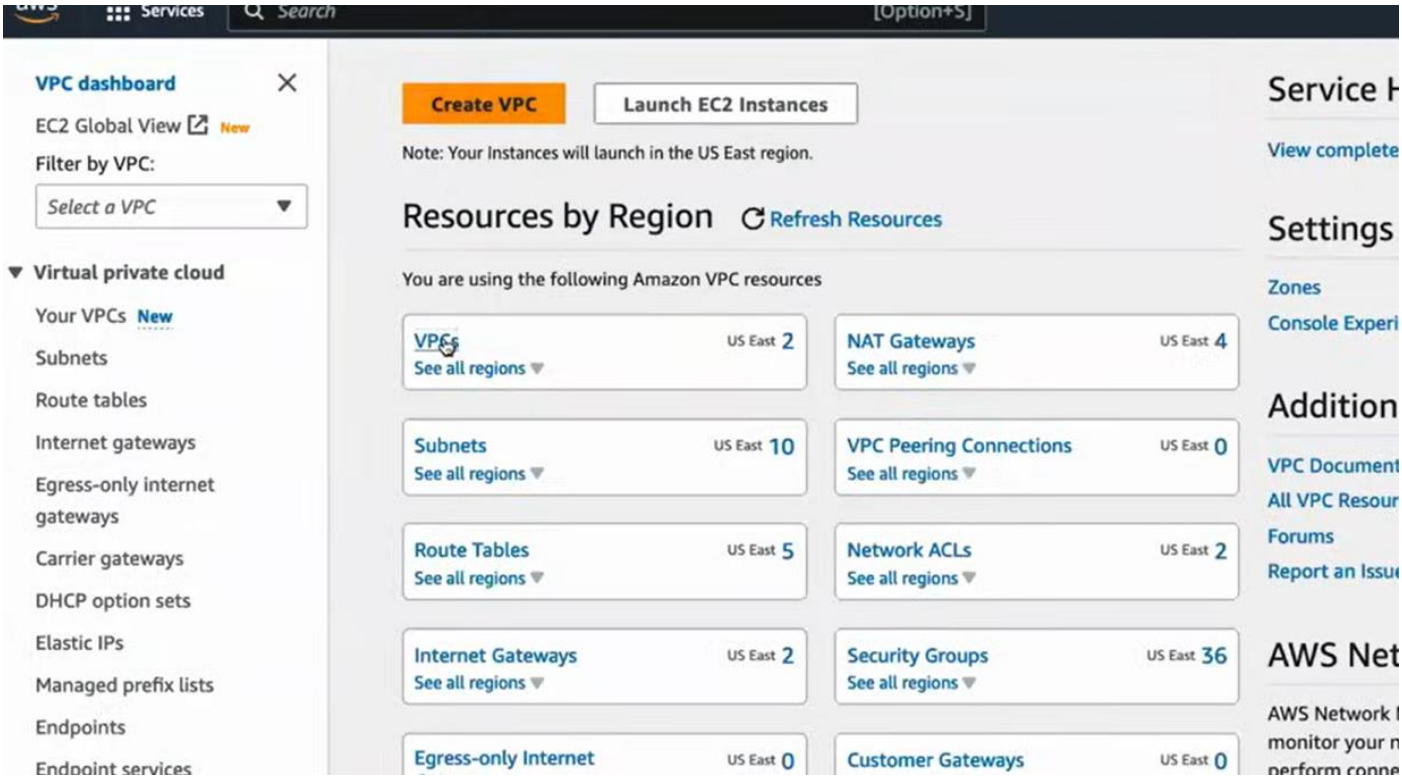


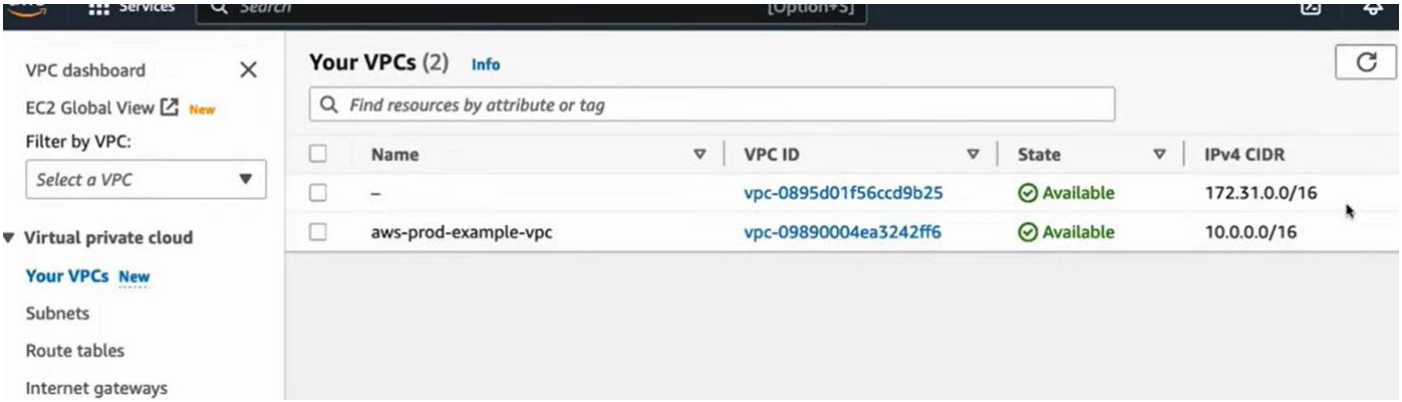
3. Select VPC and more in VPC settings and 1 Per AZ in NAT gateway, leave remaining as default and create VPC.



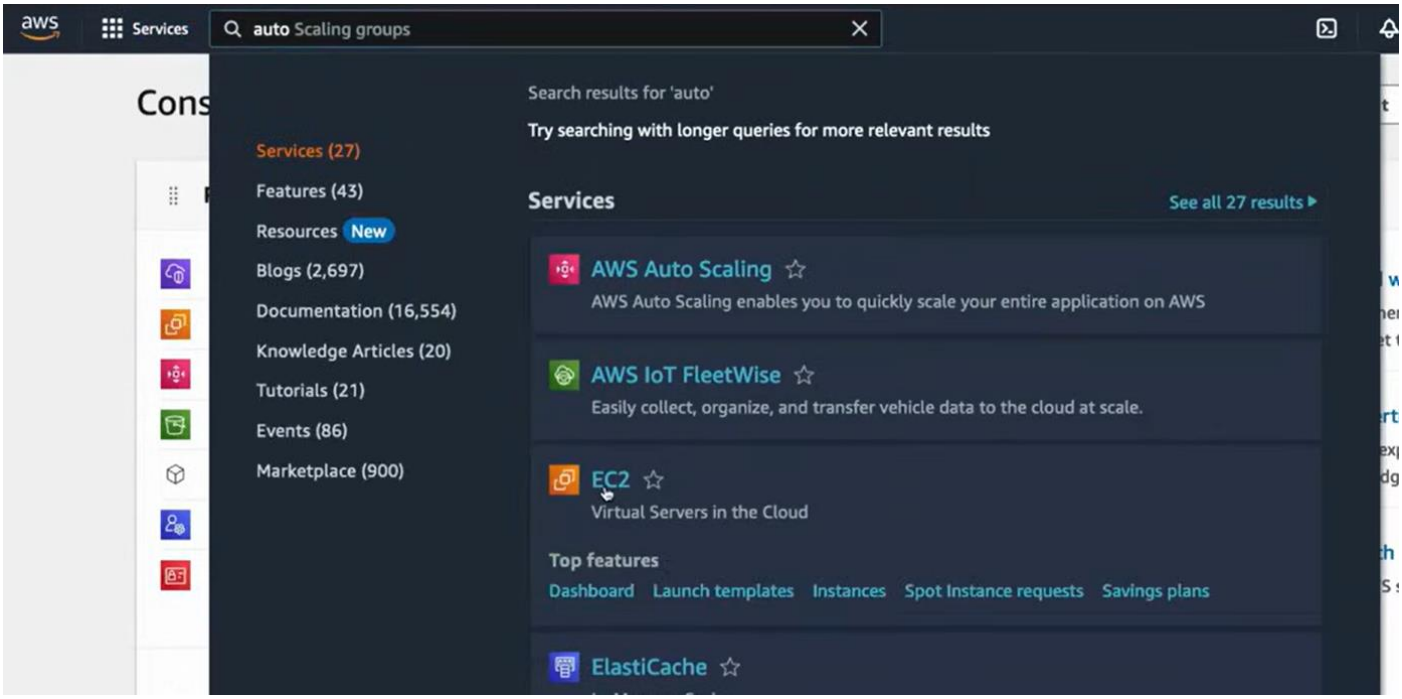


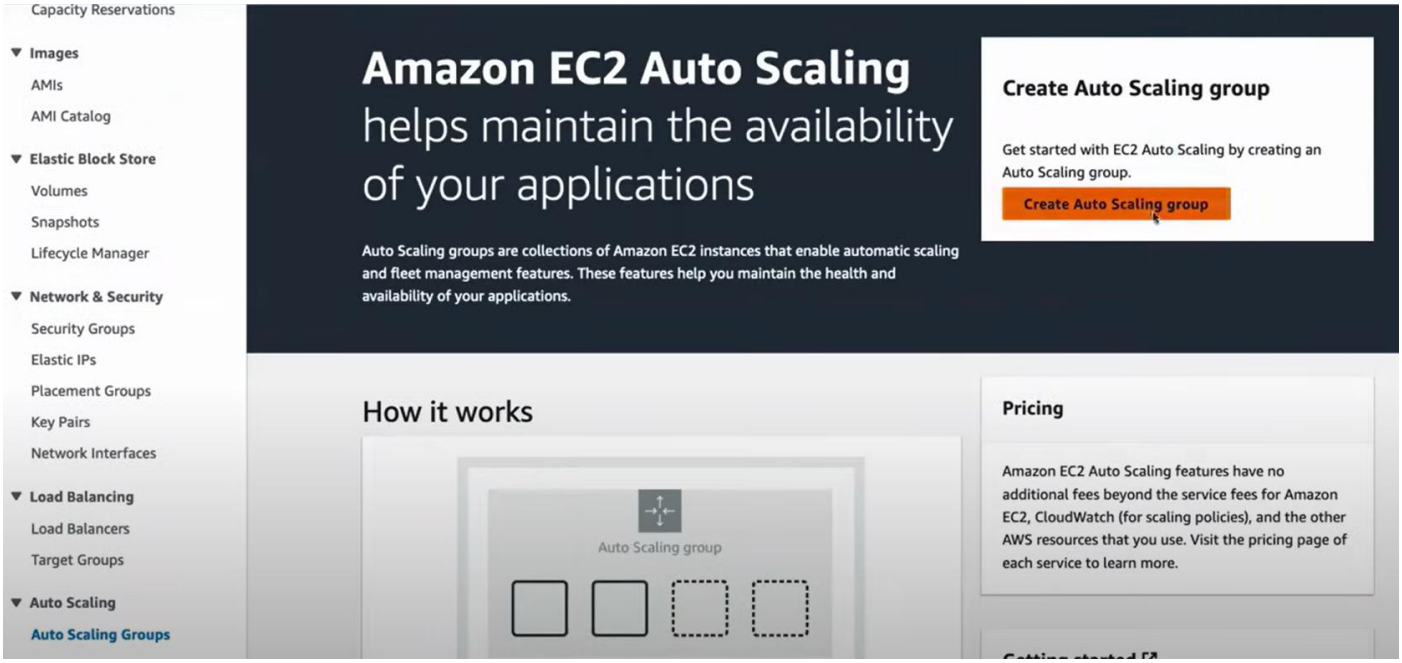
4. Click on view VPC you can see your created VPC

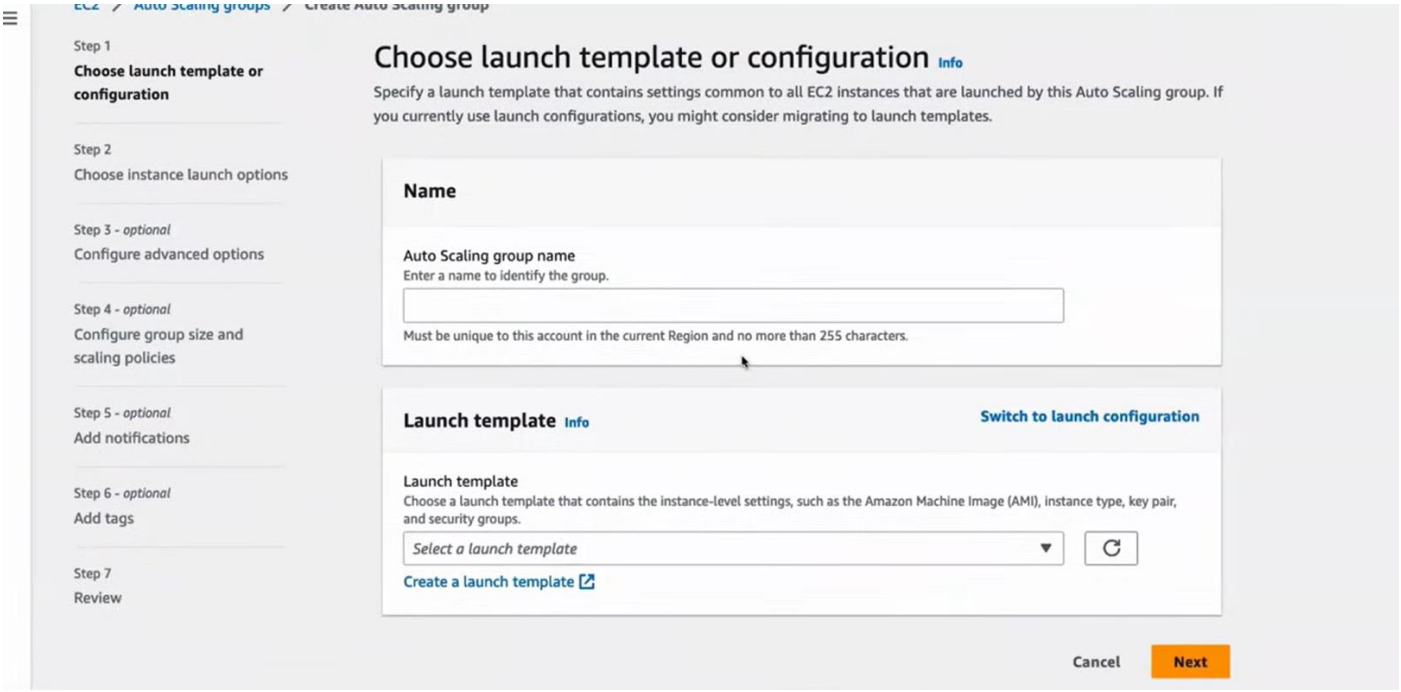


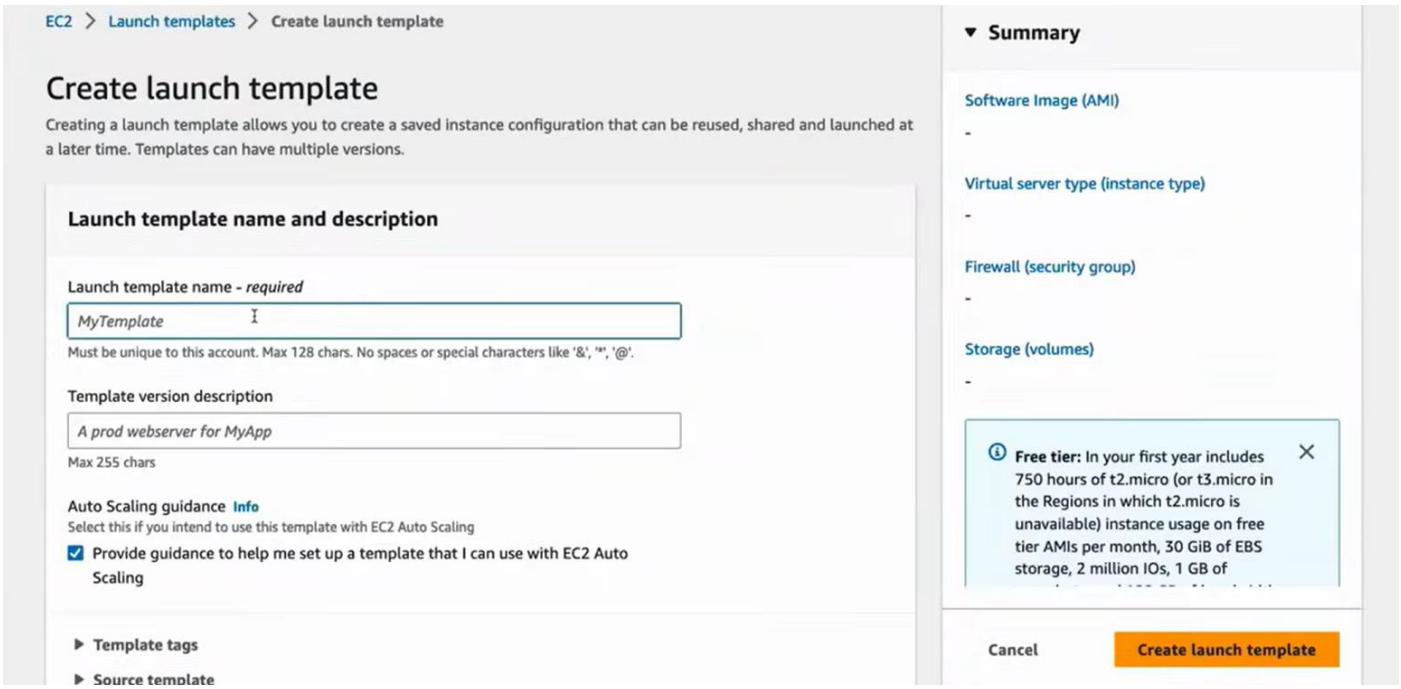


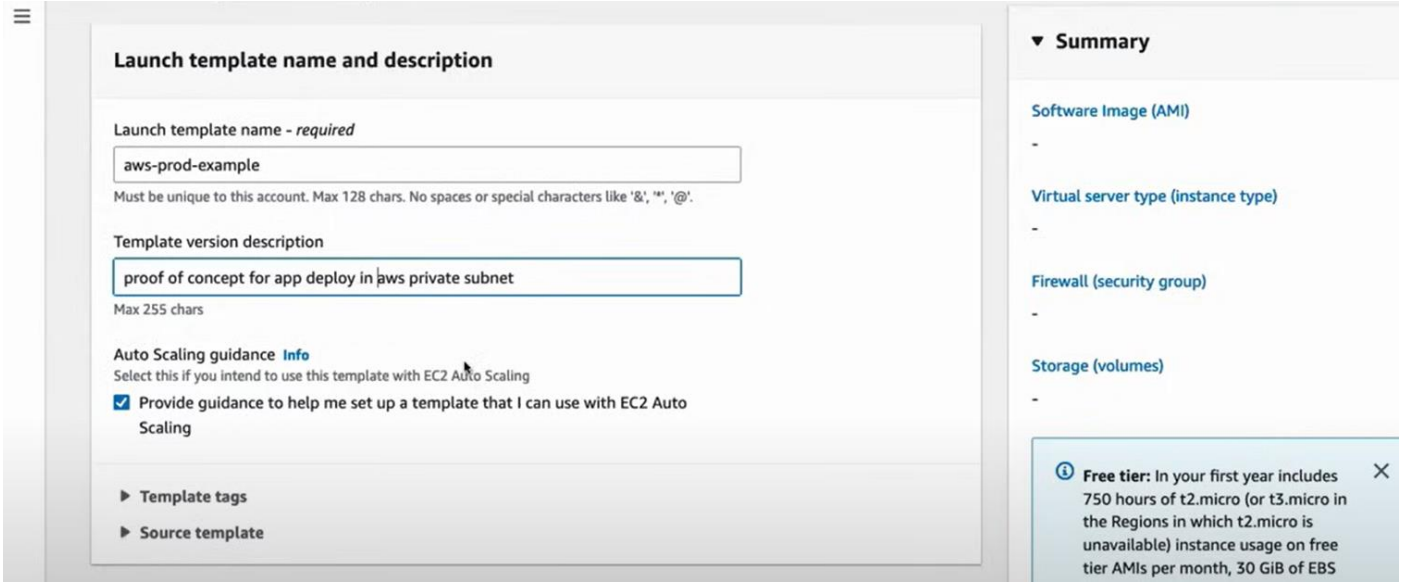
5. Now create a Auto Scaling as shown

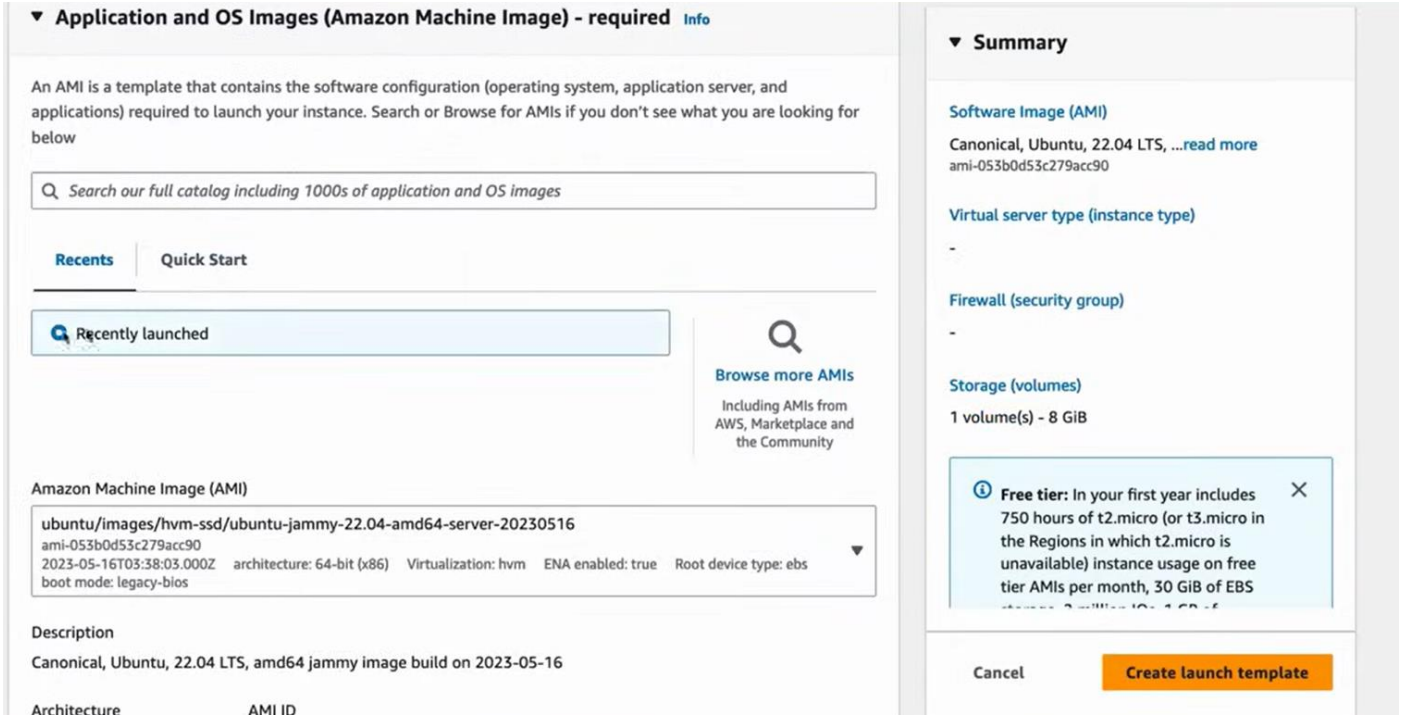




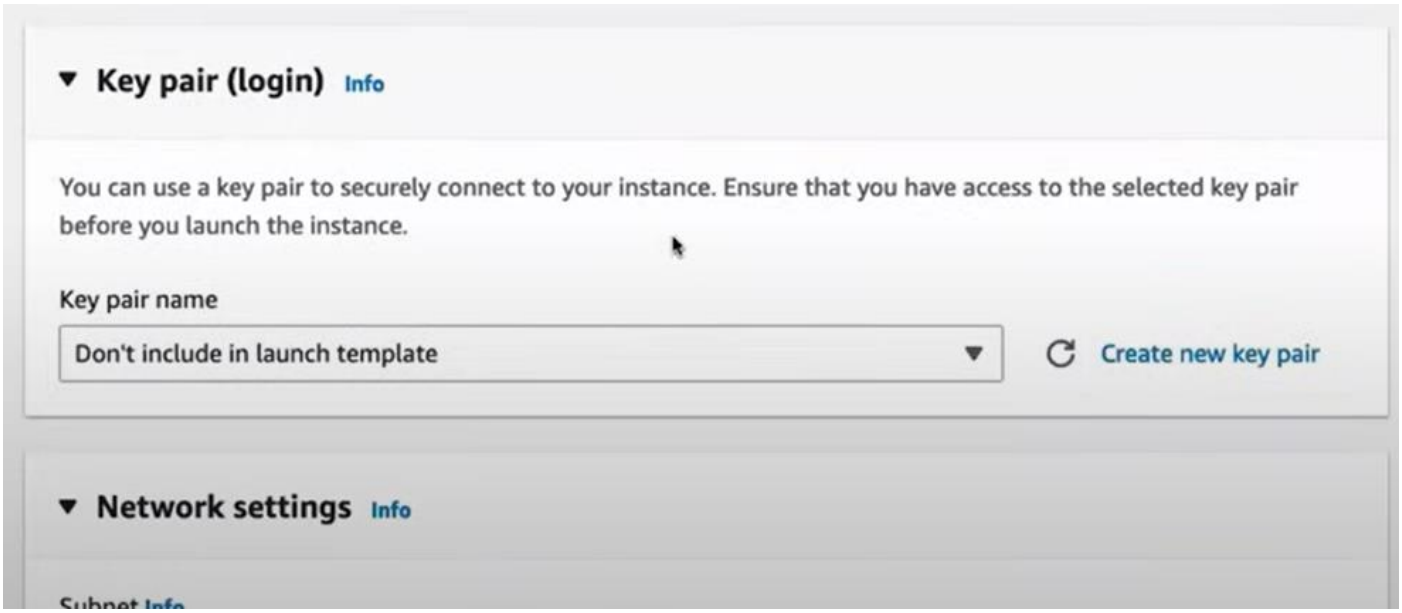

Step 1
Choose launch template or configuration
Step 2
Choose instance launch options
Step 3 - optional
Configure advanced options
Step 4 - optional
Configure group size and scaling policies
Step 5 - optional
Add notifications
Step 6 - optional
Add tags
Step 7
Review
Choose launch template or configuration Info
Specify a launch template that contains settings common to all EC2 instances that are launched by this Auto Scaling group. If you currently use launch configurations, you might consider migrating to launch templates.
Name
Auto Scaling group name
Enter a name to identify the group.
Must be unique to this account in the current Region and no more than 255 characters.
Launch template Info
Switch to launch configuration
Launch template
Choose a launch template that contains the instance-level settings, such as the Amazon Machine Image (AMI), instance type, key pair, and security groups.
Select a launch template
Create a launch template
Cancel
Next


EC2 > Launch templates > Create launch template
Create launch template
Creating a launch template allows you to create a saved instance configuration that can be reused, shared and launched at a later time. Templates can have multiple versions.
Launch template name and description
Launch template name - required
MyTemplate
Must be unique to this account. Max 128 chars. No spaces or special characters like '&', '*', '@'.
Template version description
A prod webserver for MyApp
Max 255 chars
Auto Scaling guidance Info
Select this if you intend to use this template with EC2 Auto Scaling
Provide guidance to help me set up a template that I can use with EC2 Auto Scaling
Template tags
Source template
Summary
Software Image (AMI)
-
Virtual server type (instance type)
-
Firewall (security group)
-
Storage (volumes)
-
Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of
Cancel
Create launch template

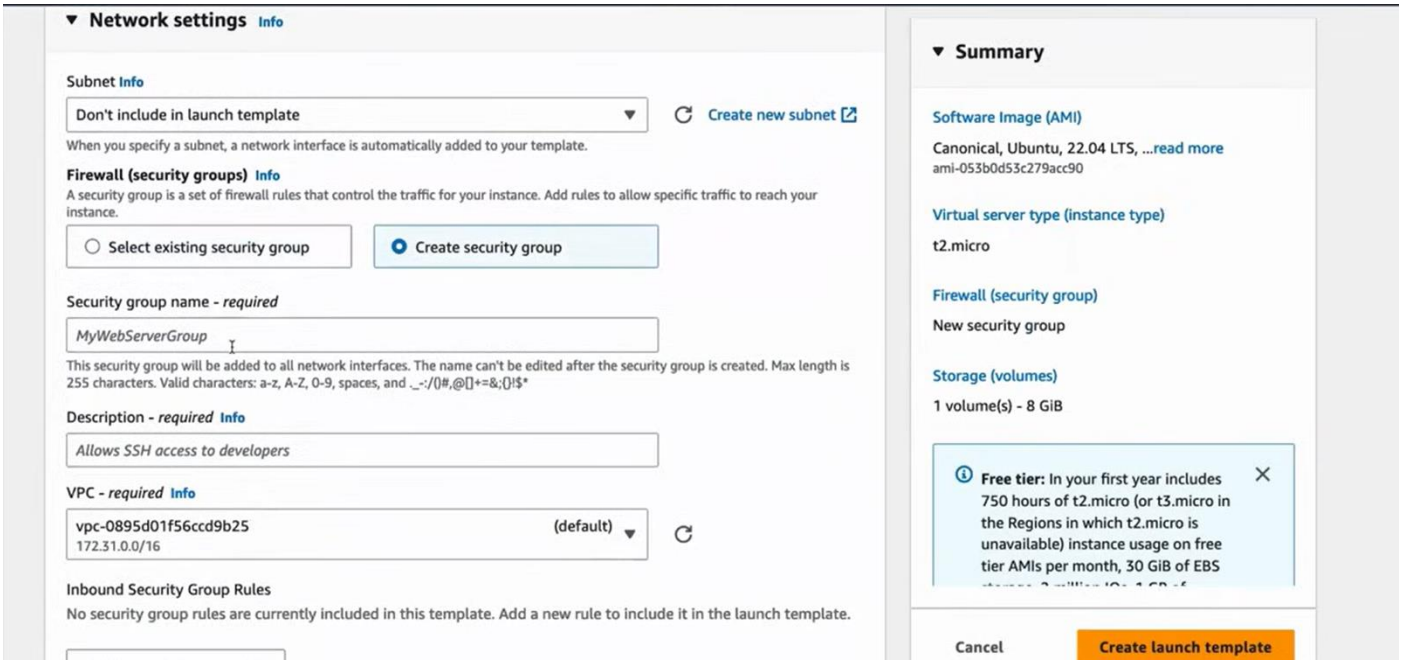

Launch template name and description
Launch template name - required
aws-prod-example
Must be unique to this account. Max 128 chars. No spaces or special characters like '&', '*', '@'.
Template version description
proof of concept for app deploy in aws private subnet
Max 255 chars
Auto Scaling guidance Info
Select this if you intend to use this template with EC2 Auto Scaling
Provide guidance to help me set up a template that I can use with EC2 Auto Scaling
Template tags
Source template
Summary
Software Image (AMI)
-
Virtual server type (instance type)
-
Firewall (security group)
-
Storage (volumes)
-
Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS



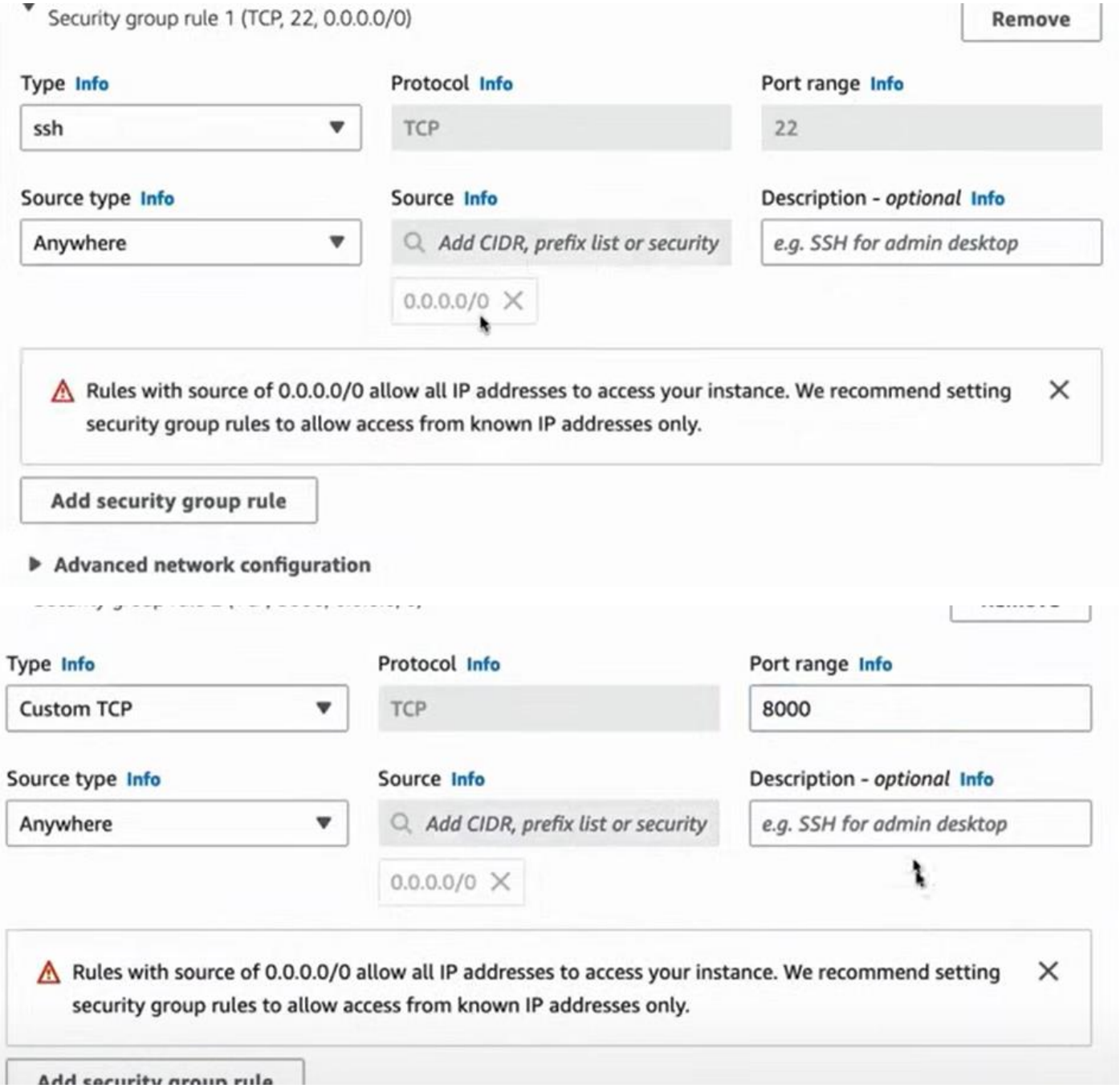
6. Add your key pair to this, if not create a new one.



7. In Network settings select create security group, give VPC that is created previously and create template as shown below.

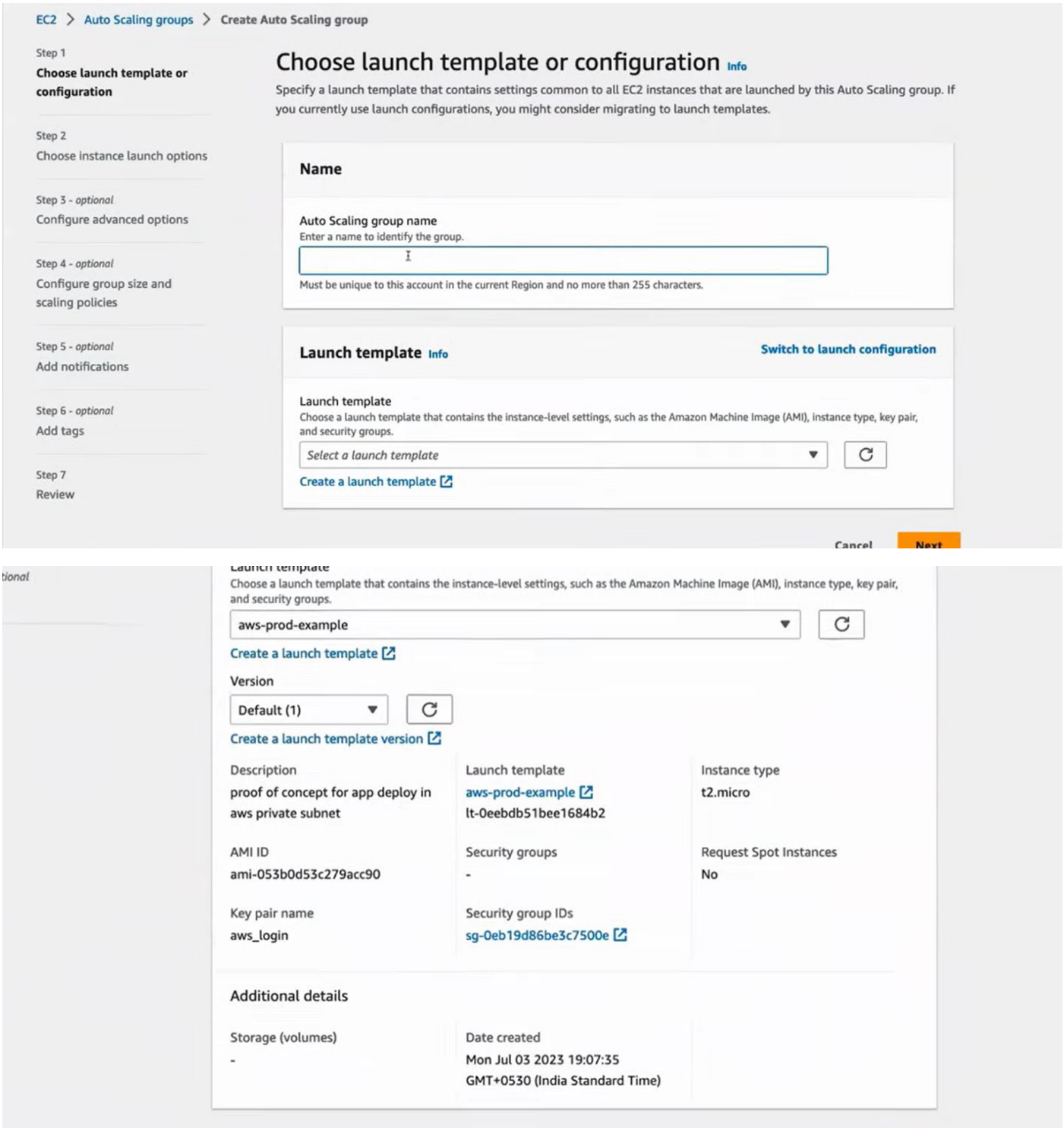


8. Click on Inbound Security Groups Rules and configure as follow.



9. Click Create template

10. Now Name Auto Scaling Group and add this created template to Auto Scaling Group. Click on next.



11. Select the created VPC that is created and also the Availability Zones and subnets. Click on next and create Auto Scaling Group.

Choose instance launch options Info

Choose the VPC network environment that your instances are launched into, and customize the instance types and purchase options.

Network Info

For most applications, you can use multiple Availability Zones and let EC2 Auto Scaling balance your instances across the zones. The default VPC and default subnets are suitable for getting started quickly.

VPC

Choose the VPC that defines the virtual network for your Auto Scaling group.

vpc-09890004ea3242ff6 (aws-prod-example-vpc)
10.0.0.0/16

Create a VPC

Availability Zones and subnets

Define which Availability Zones and subnets your Auto Scaling group can use in the chosen VPC.

Select Availability Zones and subnets

Create a subnet

Configure advanced options - *optional* Info

Integrate your Auto Scaling group with other services to distribute network traffic across multiple servers using a load balancer or to establish service-to-service communications using VPC Lattice. You can also set options that give you more control over health check replacements and monitoring.

Load balancing Info

Use the options below to attach your Auto Scaling group to an existing load balancer, or to a new load balancer that you define.

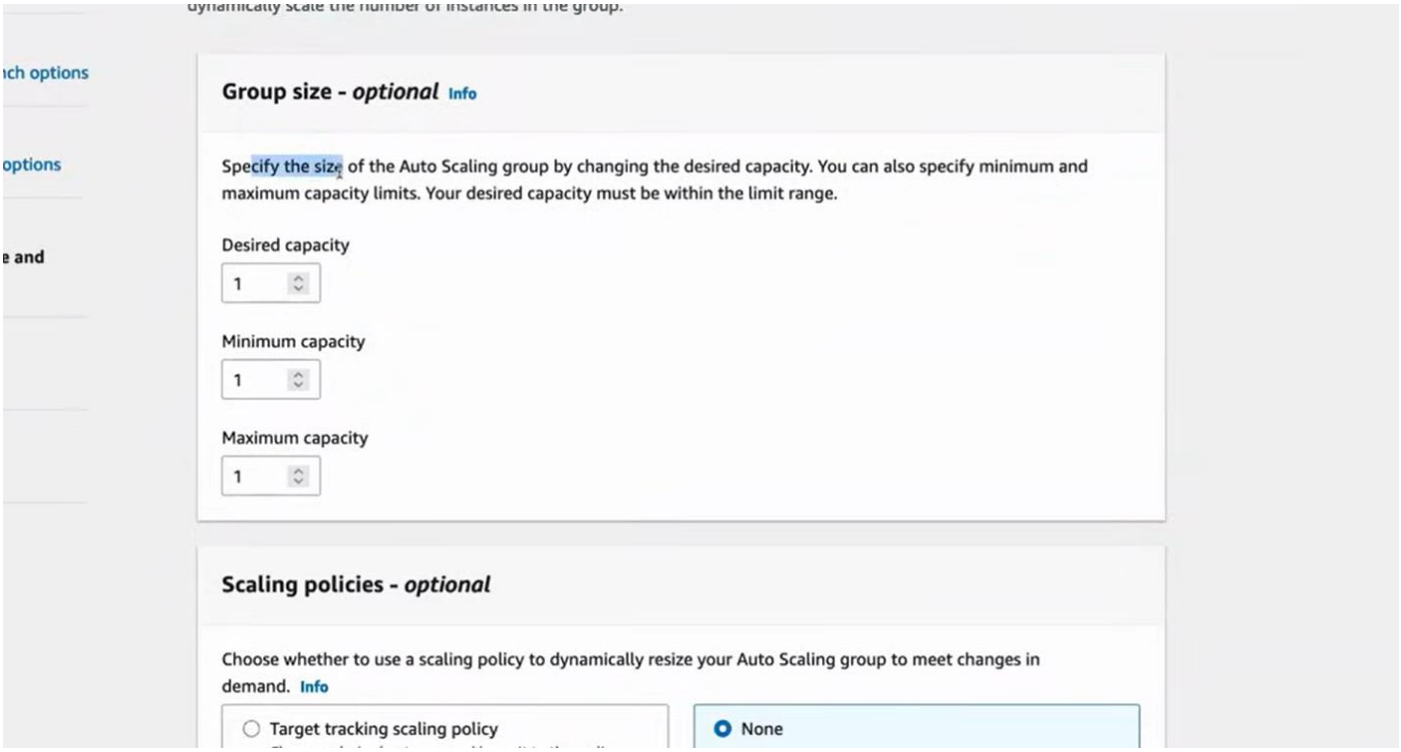
- (●) **No load balancer**
  Traffic to your Auto Scaling group will not be fronted by a load balancer.
- ( ) **Attach to an existing load balancer**
  Choose from your existing load balancers.
- ( ) **Attach to a new load balancer**
  Quickly create a basic load balancer to attach to your Auto Scaling group.

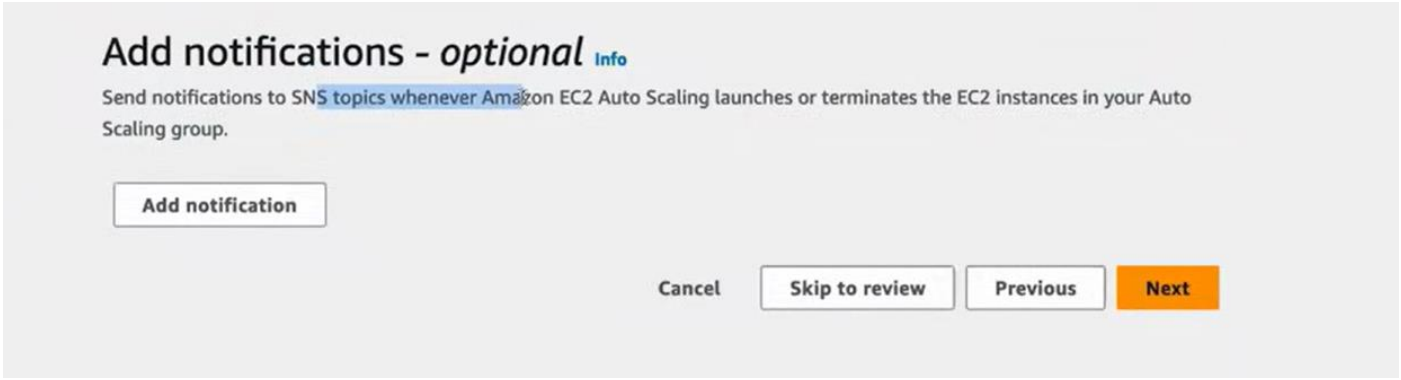
VPC Lattice integration options Info

To improve networking capabilities and scalability, integrate your Auto Scaling group with VPC Lattice. VPC Lattice facilitates communications between AWS services and helps you connect and manage your applications across compute services in AWS.

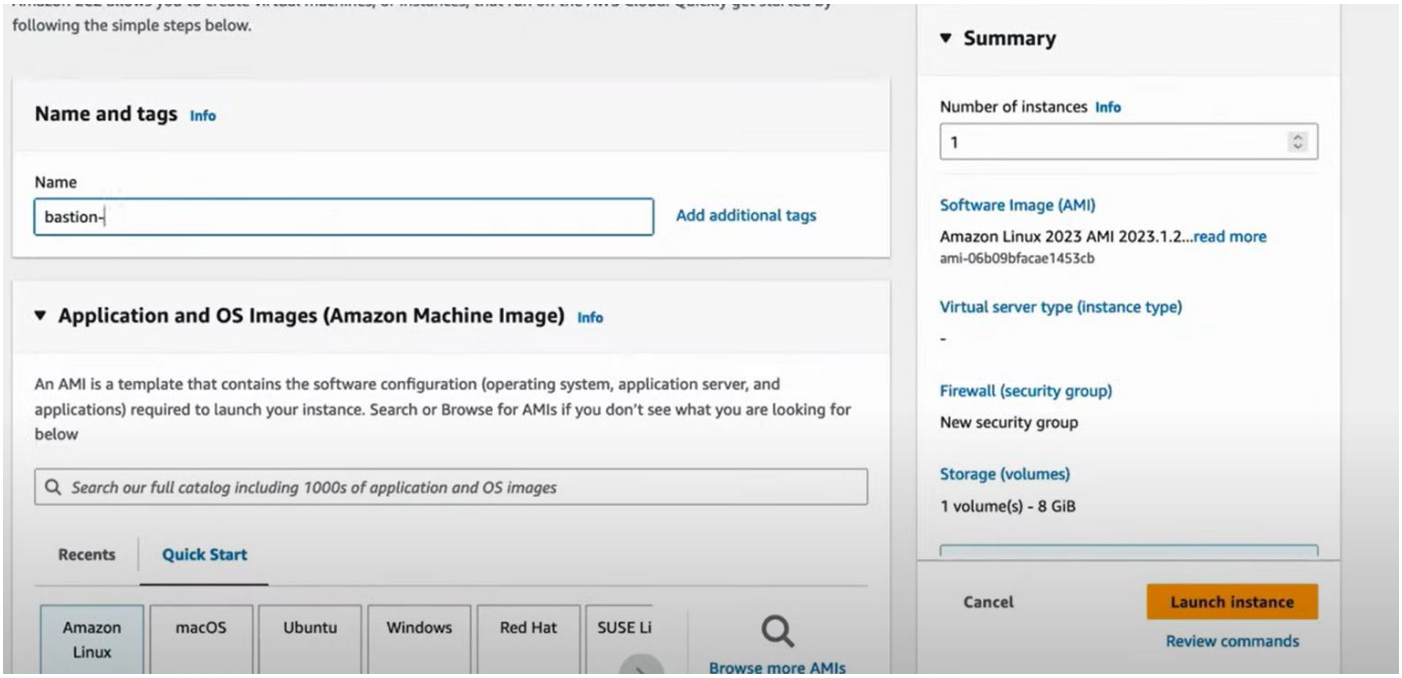
Select VPC Lattice service to attach

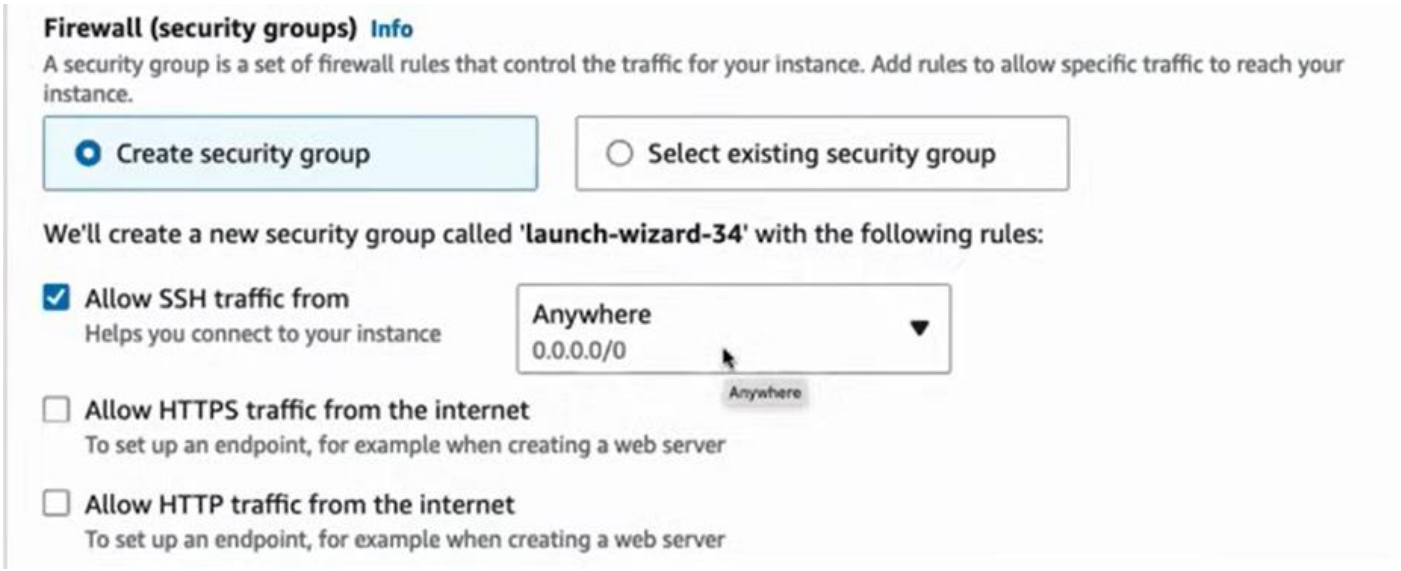
- (●) **No VPC Lattice service**
  VPC Lattice will not manage your Auto Scaling group's network access and connectivity with other services.
- ( ) **Attach to VPC Lattice service**
  Incoming requests associated with specified VPC Lattice target groups will be routed to your Auto Scaling group.

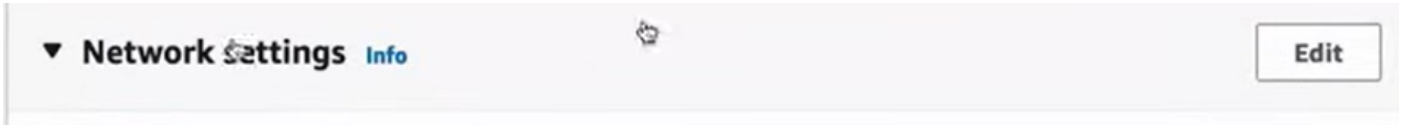




12. Check and verify the Two Instances are created in desired region.
13. Now create a new EC2 instance and launch as shown below.

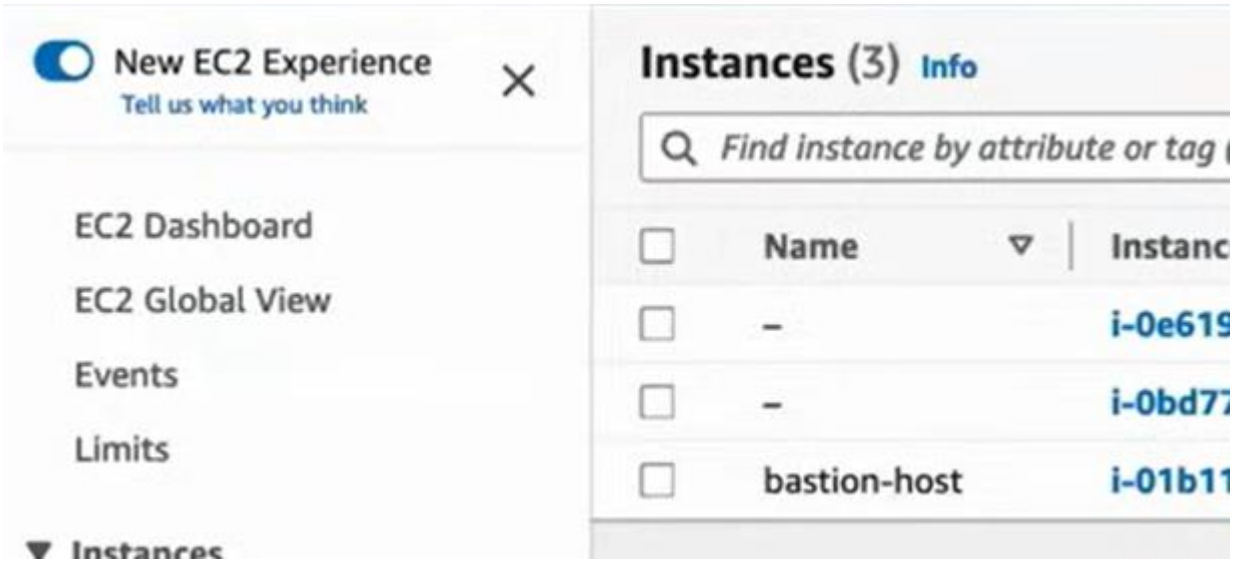






14. Make sure that the Bastion VPC is created in same VPC .

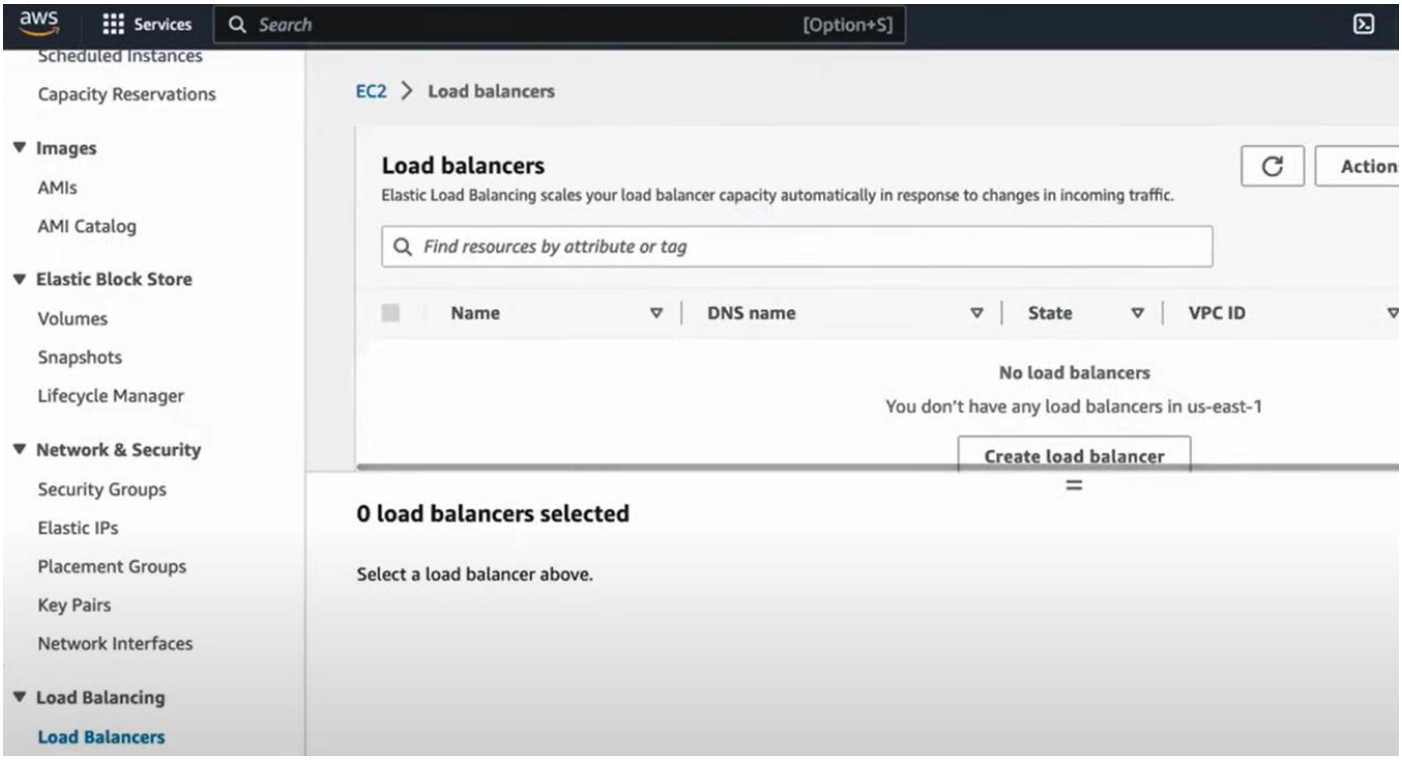
15. You can see instances are created.

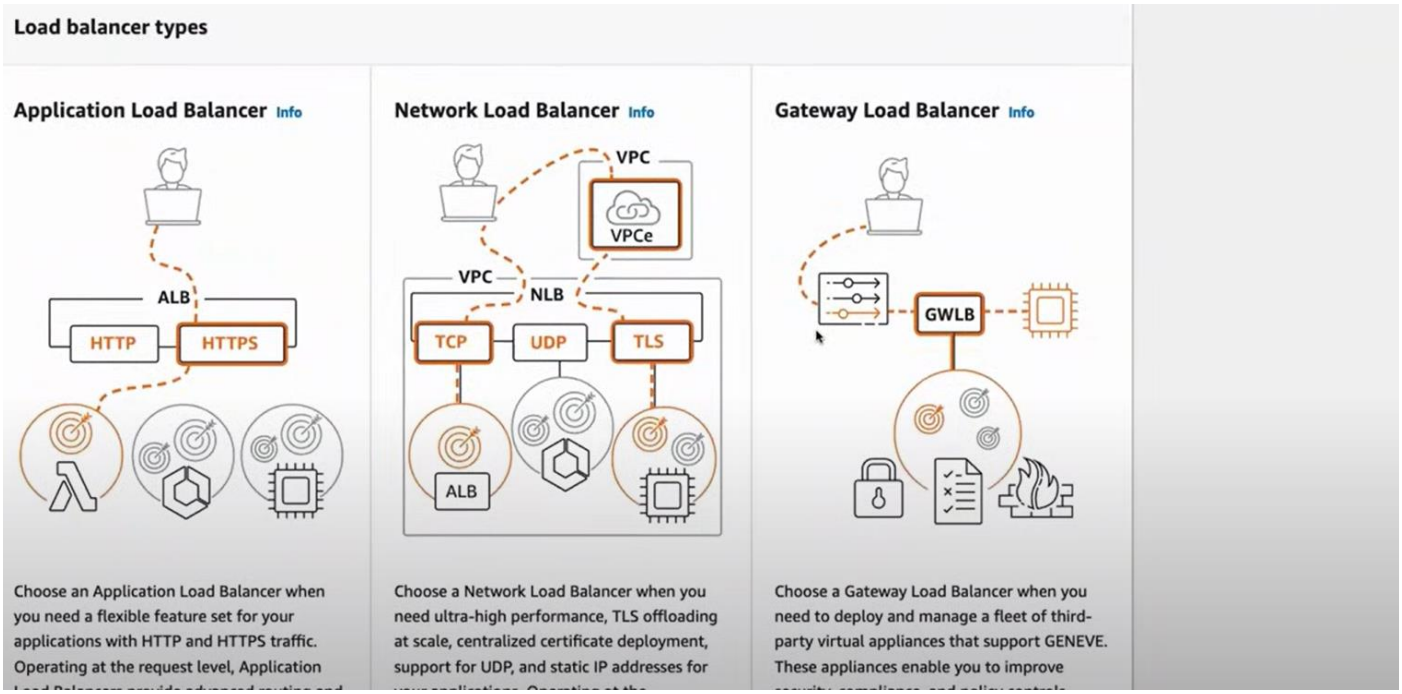


16. Use your Terminal to login and give the SSH key to it using Scp command to the Bastion instance using Ip address of it.
17. Use the below command with your bastion Ip.
18. Do ls command you will see your Aws pem file. **ssh – i your pem file ubuntu@your ip address**
19. Now we want to install our application in any one of our two instances, login to any one instance using private ip address using this command **ssh – i your pem file ubuntu@your ip address**
20. Now your are login in a instance using private ip. Install/add your application here within this instance.
21. I am using some basic html page and python3 with server.

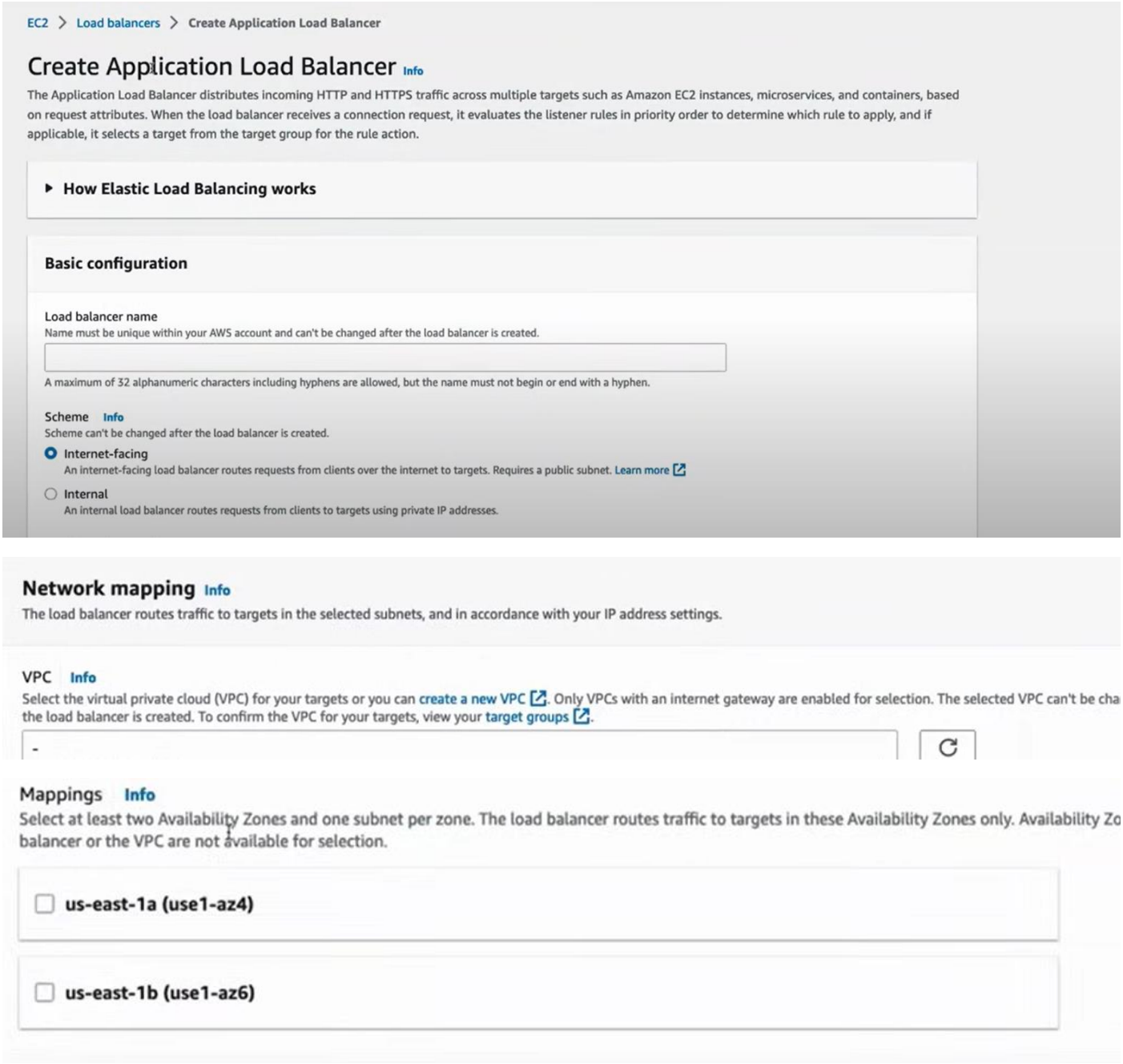
```
vim index.html
python3 -m http.server 8000
```

22. Now go to EC2 Load balancers.

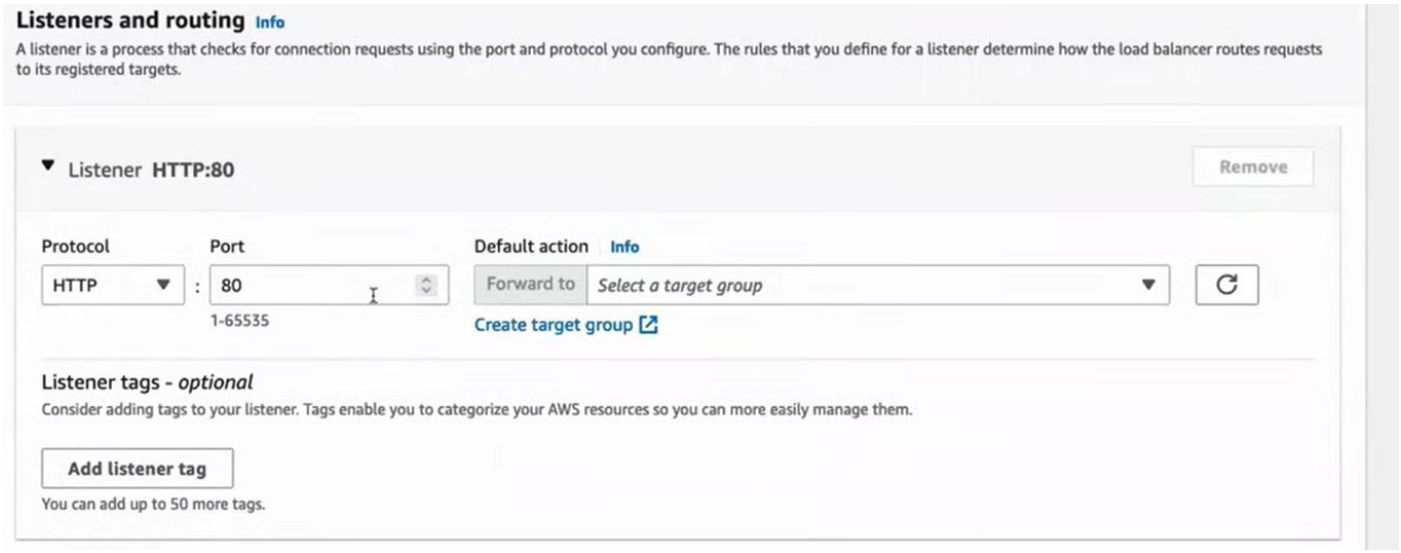




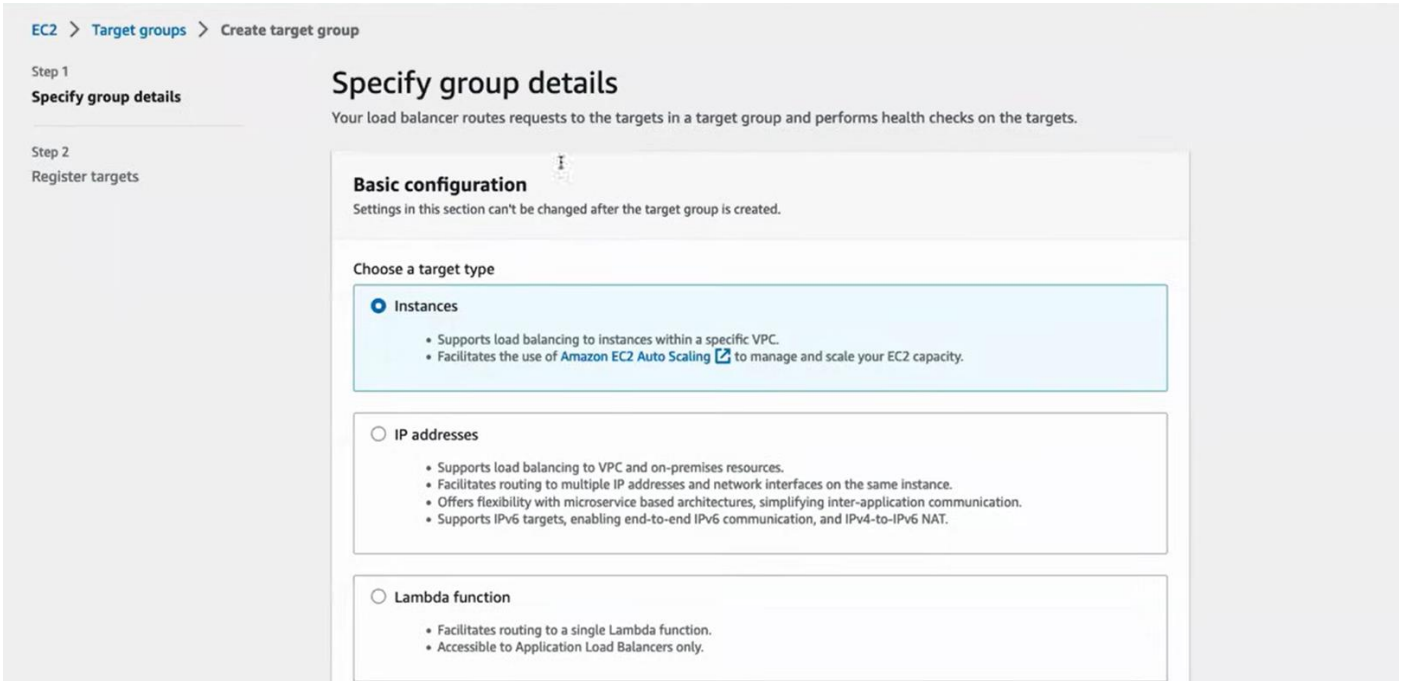
23. Select Application Load Balancer and create.

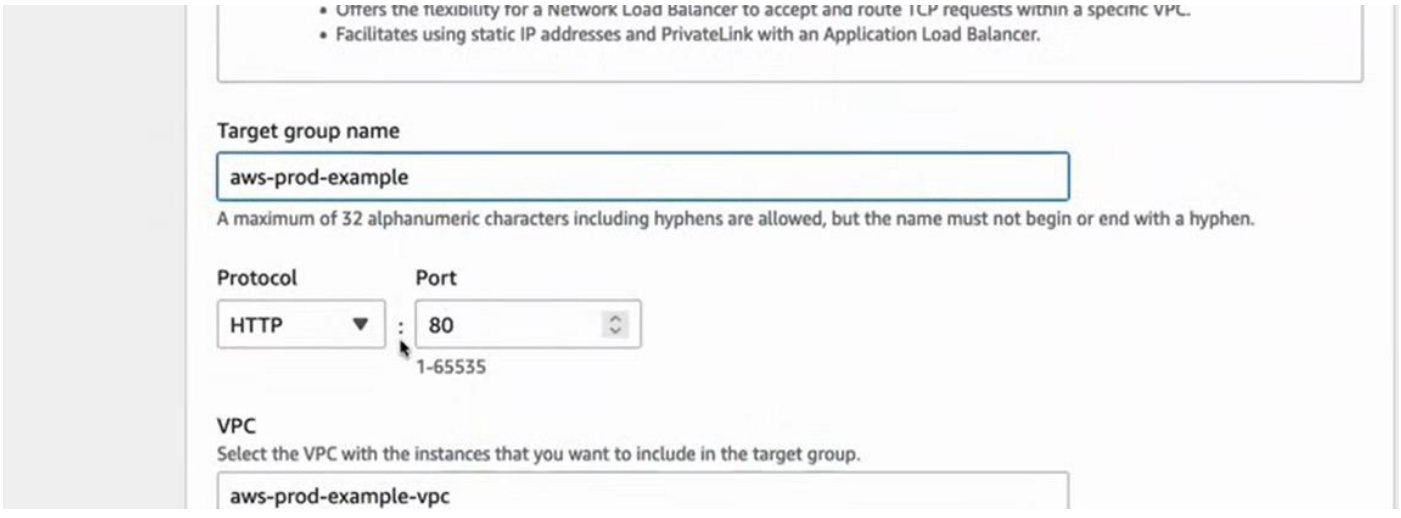


24. Provide the VPC that is Created now and pick up the both Mappings change to public subnet.
25. Now select security groups already created or you can create custom one also.

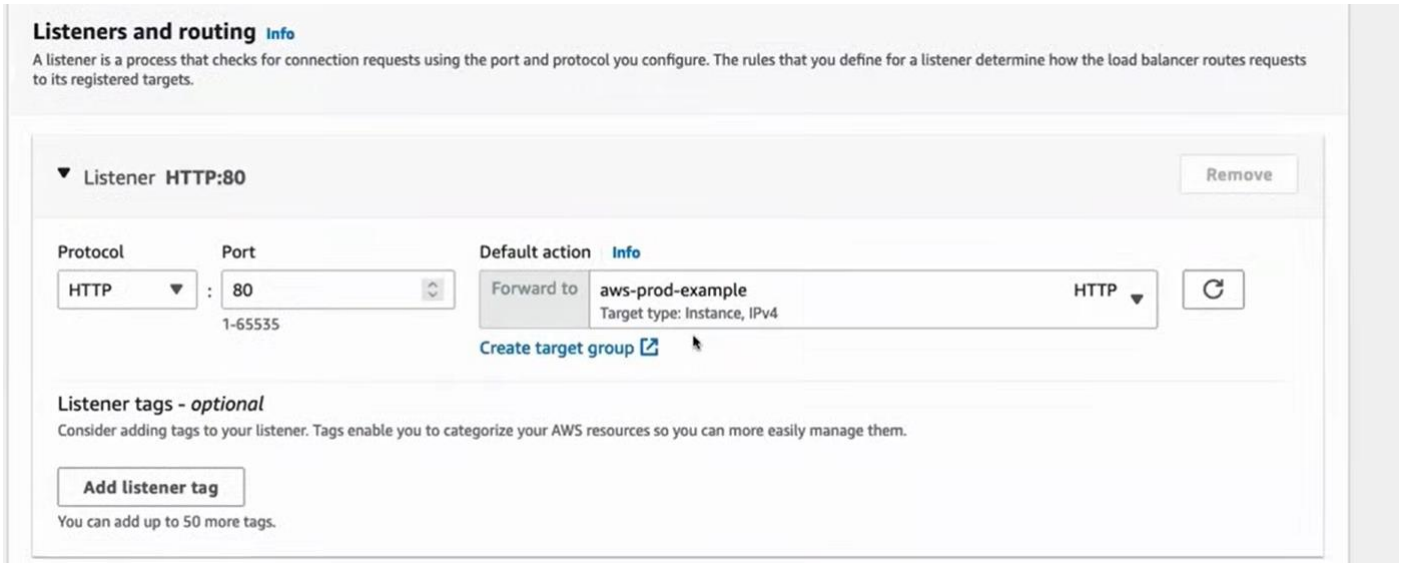


26. Create target group using port 8000

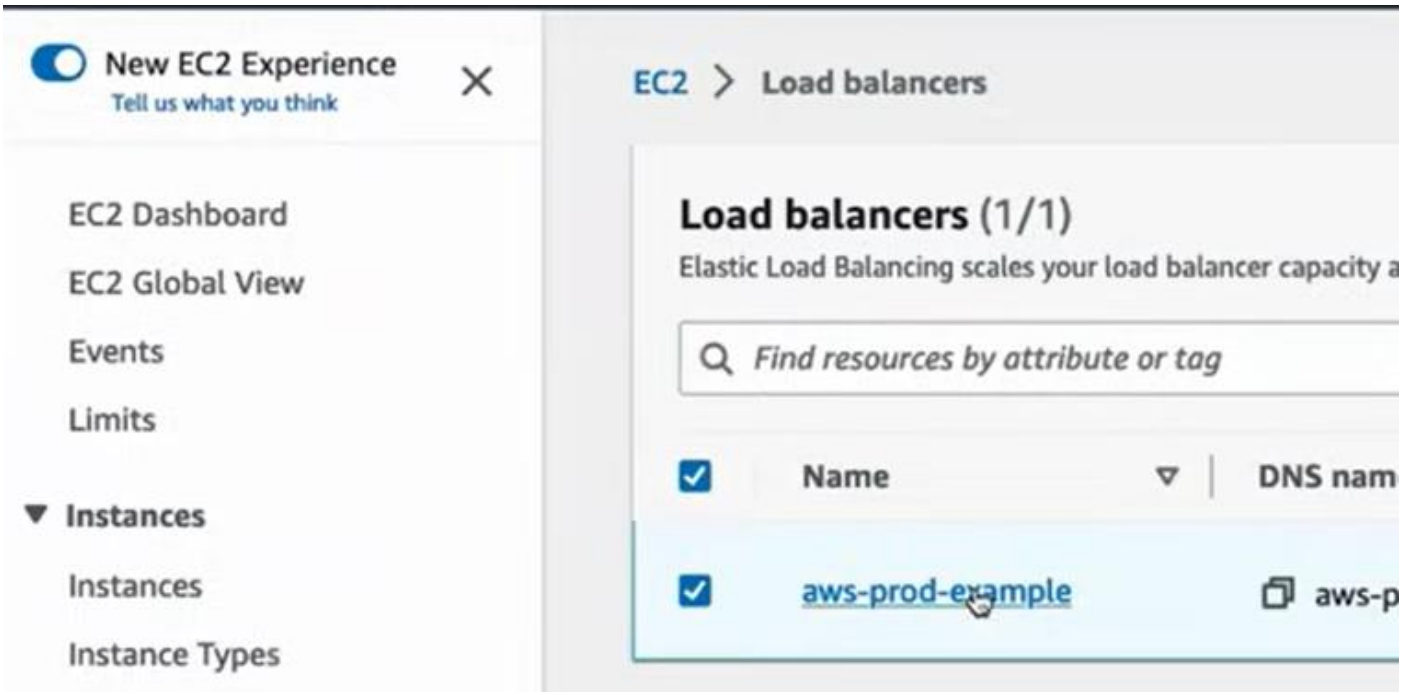




27. Now select the both instances and create and add this to the load balancer.



28. Now create the load balancer.



29. Click on the load balancer and copy DNS name in details section and search in your browser, you can see your application.

30. The above one is my basic application which I used. The project is done.

NOTE: For more information please refere to this amazon official doc.

https://docs.aws.amazon.com/vpc/latest/userguide/vpc-example-private-subnets-nat.html