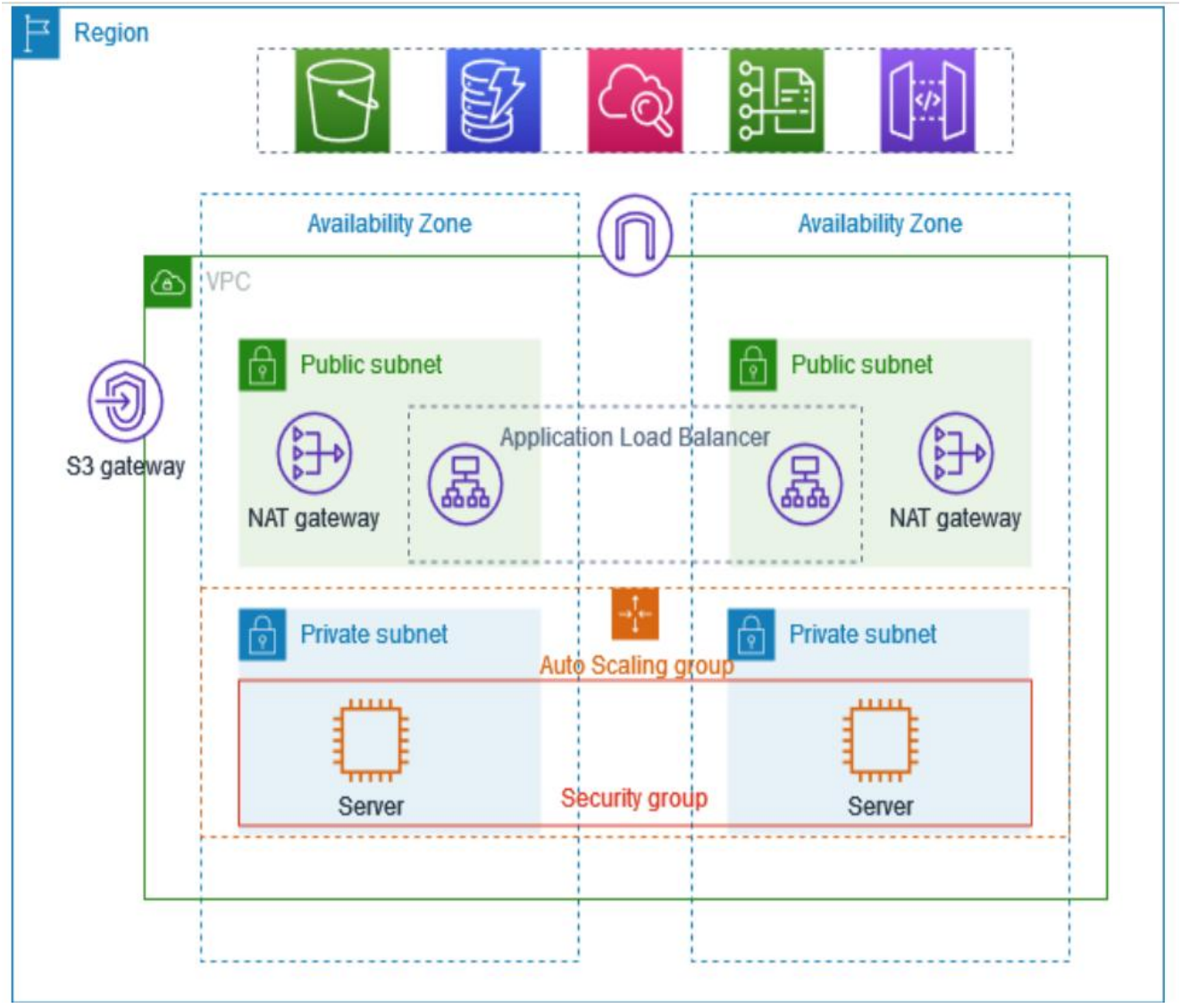
VPC WITH PUBLIC PRIVATE SUBNET IN PRODUCTION

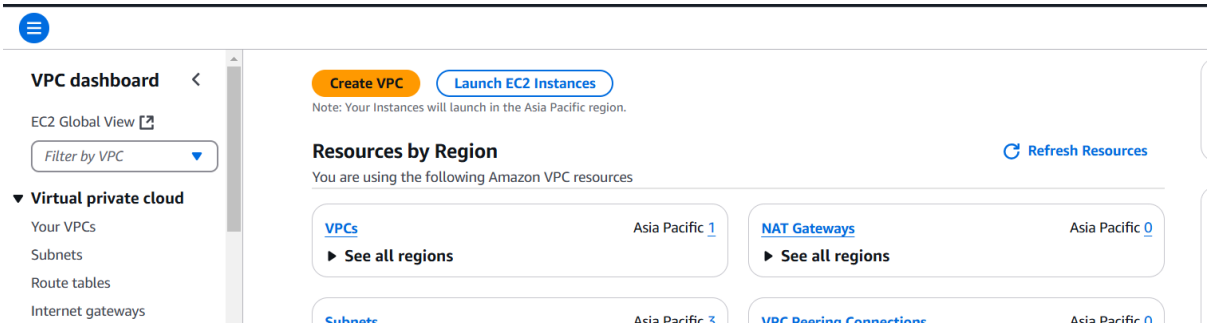
This example demonstrates how to create a VPC that you can use for servers in a production environment. To improve resiliency, you deploy the servers in two Availability Zones, by using an Auto Scaling group and an Application Load Balancer. For additional security, you deploy the servers in private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

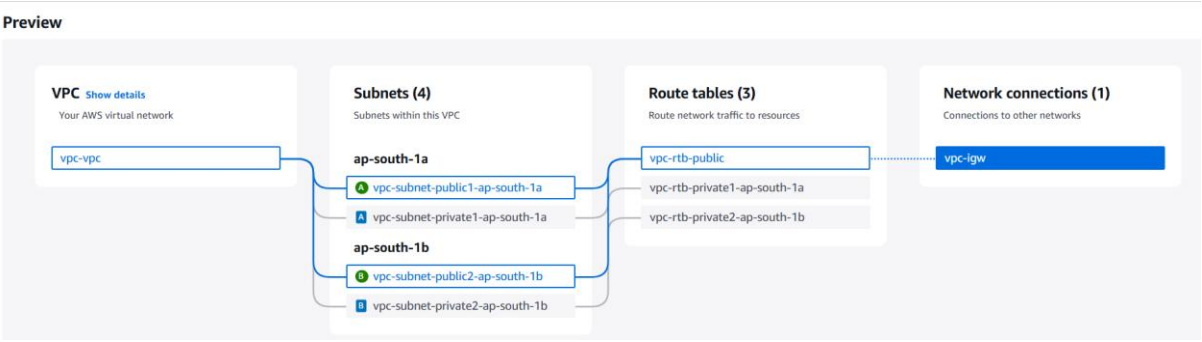
The following diagram provides an overview of the resources included in this example. The VPC has public subnets and private subnets in two Availability Zones. Each public subnet contains a NAT gateway and a load balancer. The servers run in the private subnets, are launched and terminated by using an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



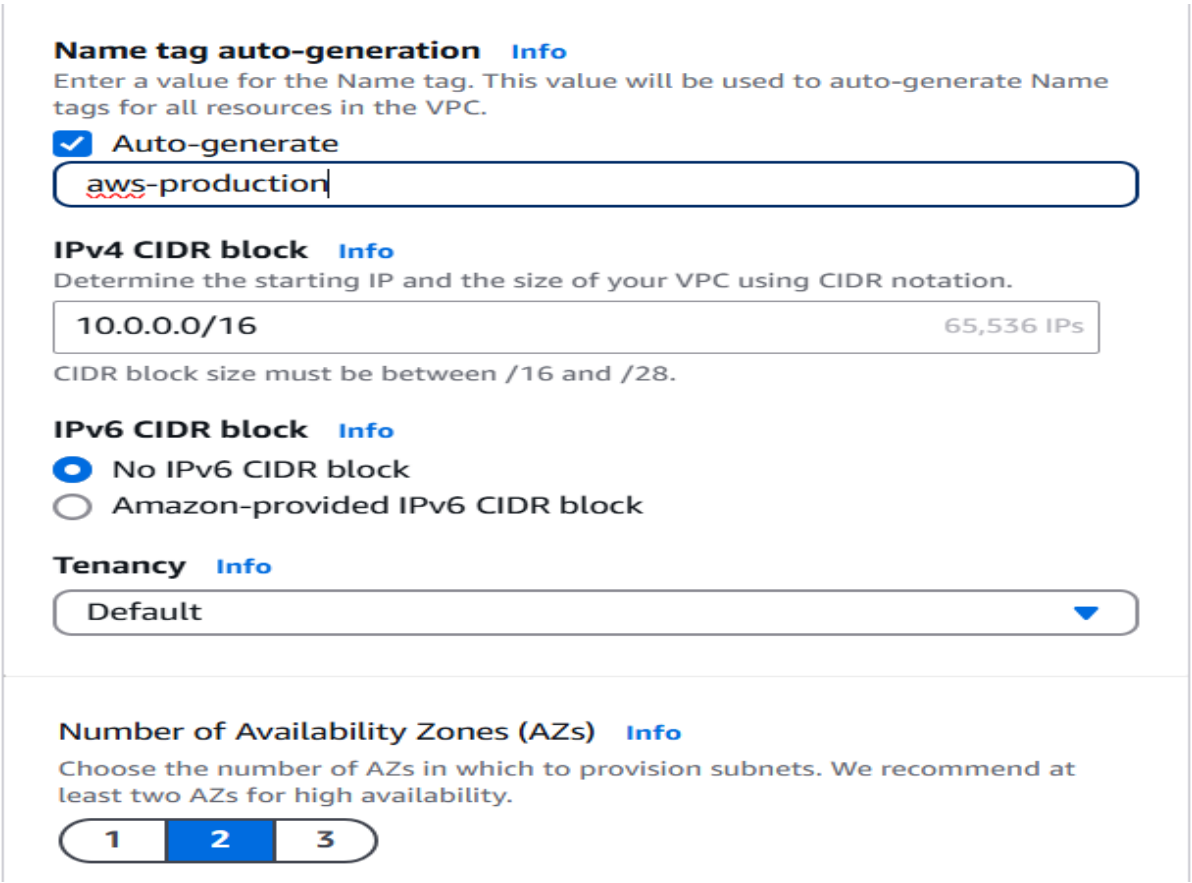
>Goto vpc > create vpc

>Create vpc and more



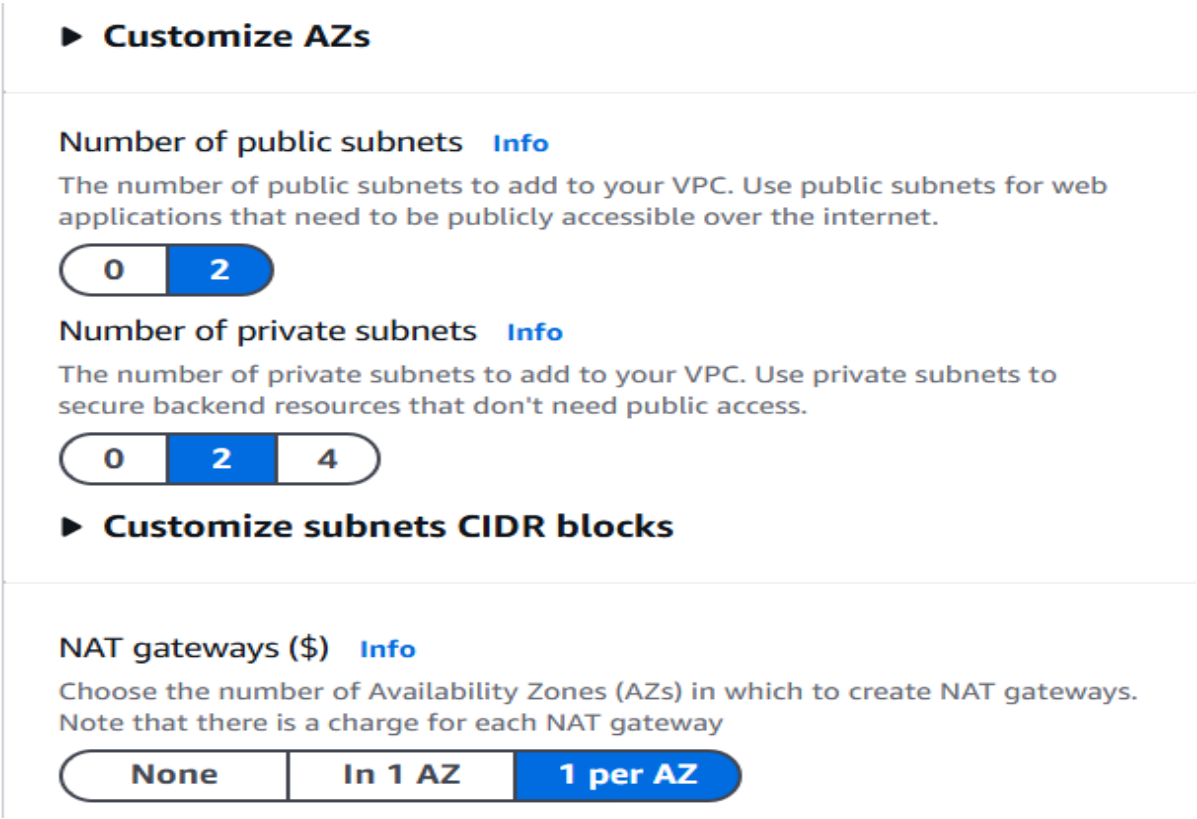


>name > number of AZ >2

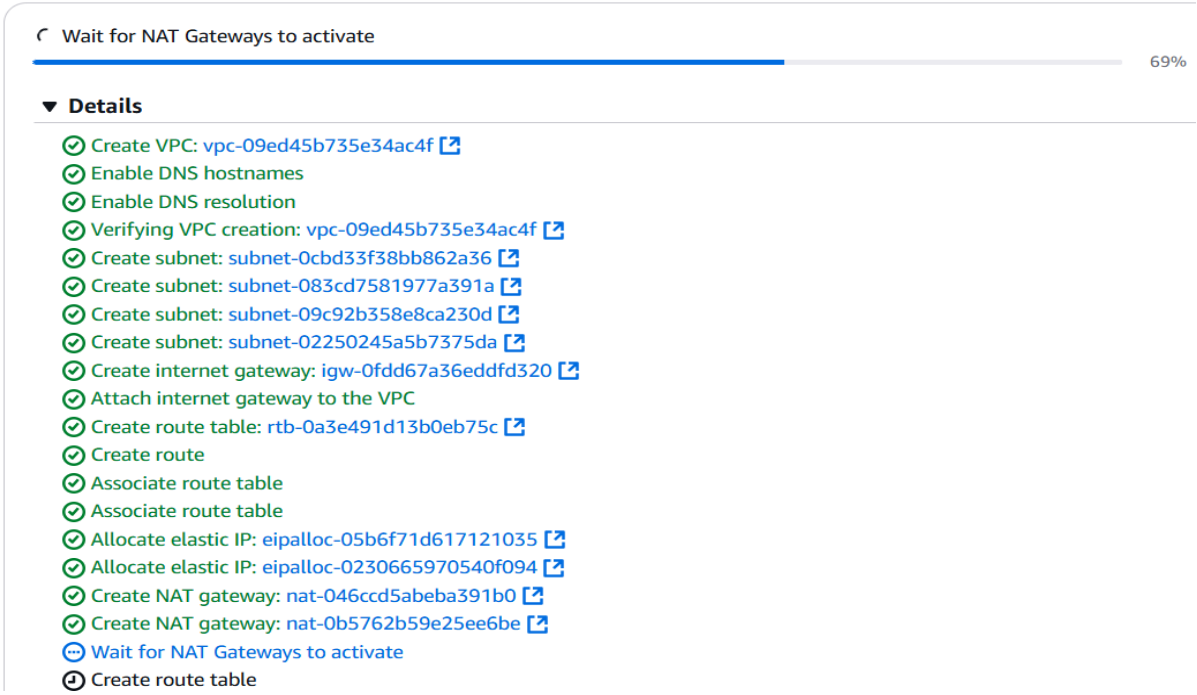


>Number of public subnets > 2 > Number of private subnets > 2

> Number of NAT Gateway > 1 per AZ



>Create



You can see all components are create automatically.

>Create 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*

temp

Must be unique to this account. Max 128 chars. No spaces or special characters like '&', '*', '@'.

Template version description

for production

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

>select ubuntu image

Recents | Quick Start

Amazon Linux | macOS | Ubuntu | Windows | Red Hat | SUSE Linux | Debian

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type — Free tier eligible

ami-00bb6a80f01f03502 (64-bit (x86)) / ami-09773b29dffbef1f2 (64-bit (Arm))

Virtualization: hvm ENA enabled: true Root device type: ebs

>Create new security group in production vpc

Firewall (security groups) | Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

○ Select existing security group ◉ Create security group

Security group name - *required*

new-sg

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and ._-:/()#,@[]+=&;{}!$*

Description - *required* | Info

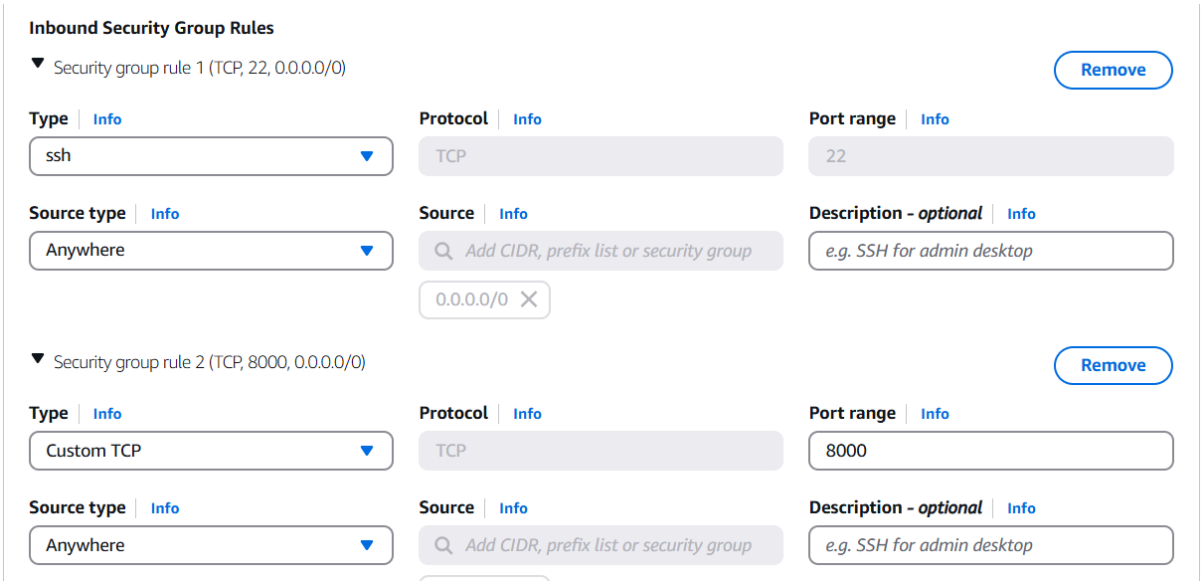
new-sg

VPC | Info

vpc-09ed45b735e34ac4f (aws-production-vpc)

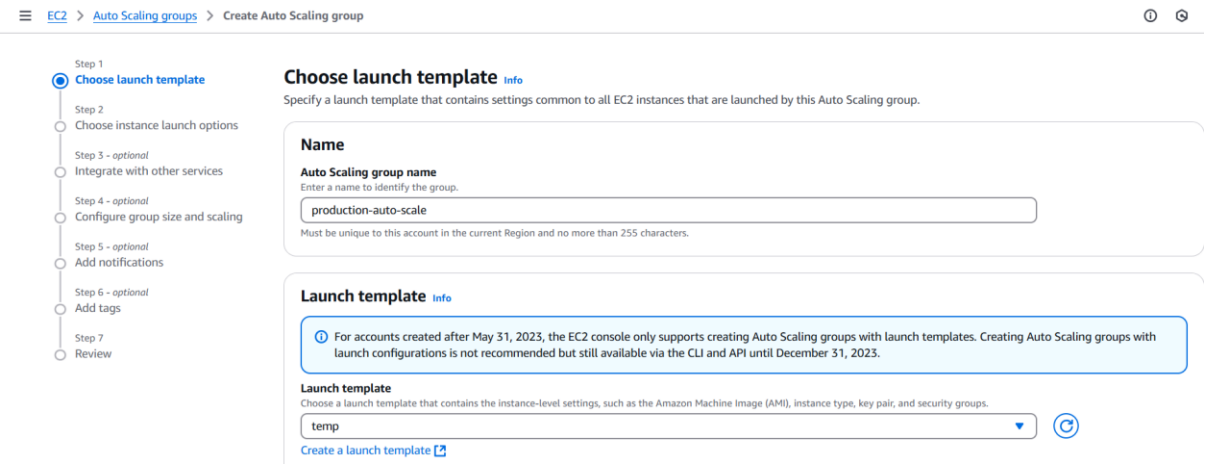
10.0.0.0/16

>in security group allow SSH and allow 8000 port

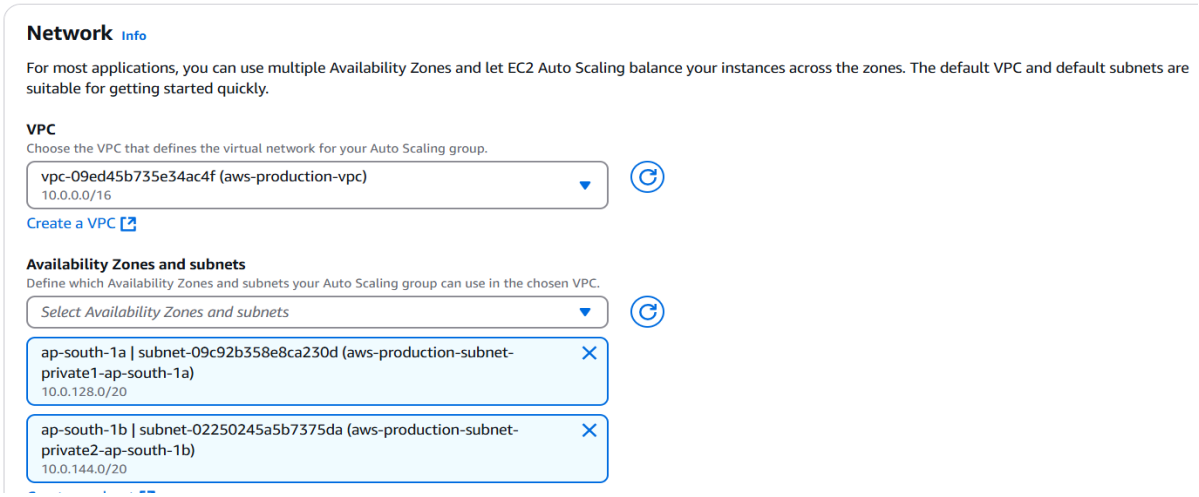


>done

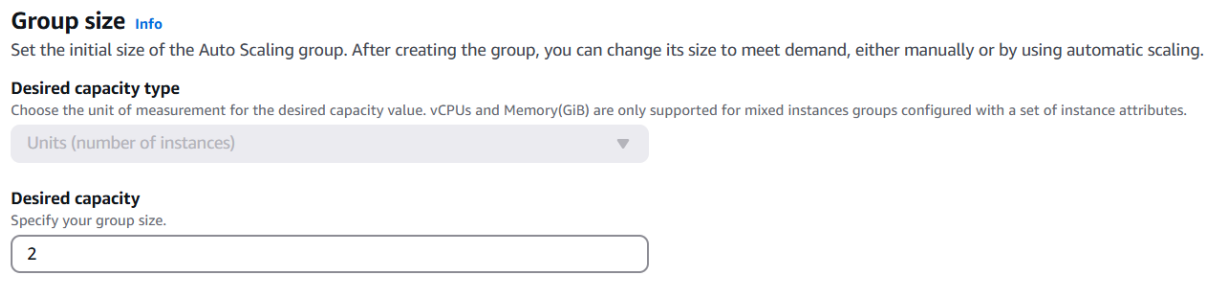
>Create Auto-scaling

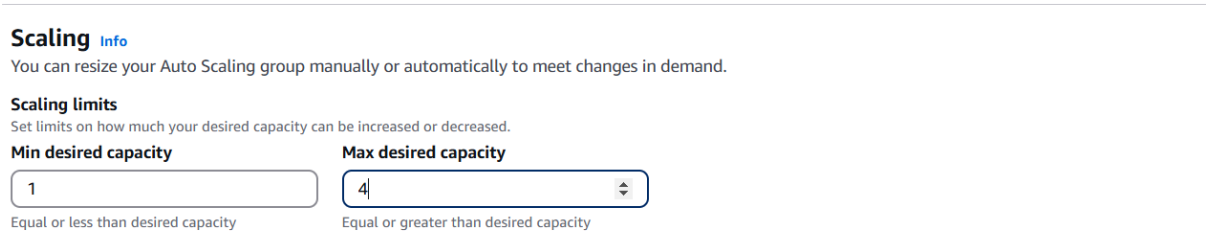


>name > select temp

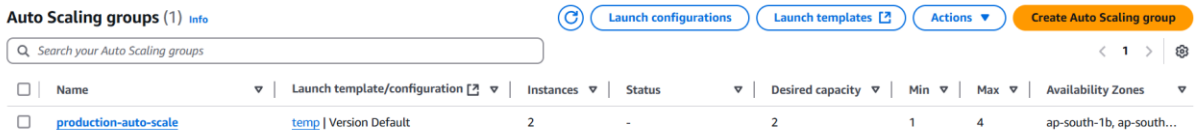


>select production vpc >select 2 private subnet

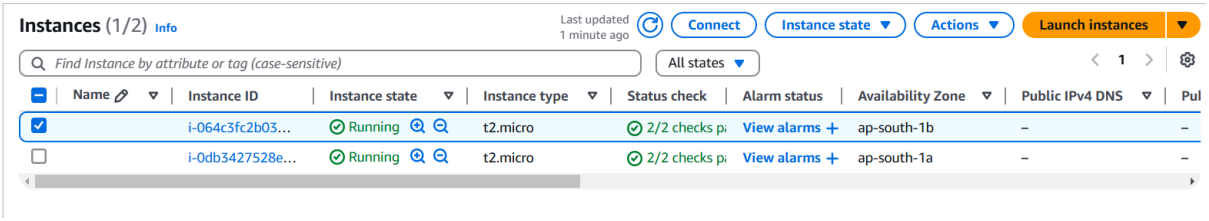




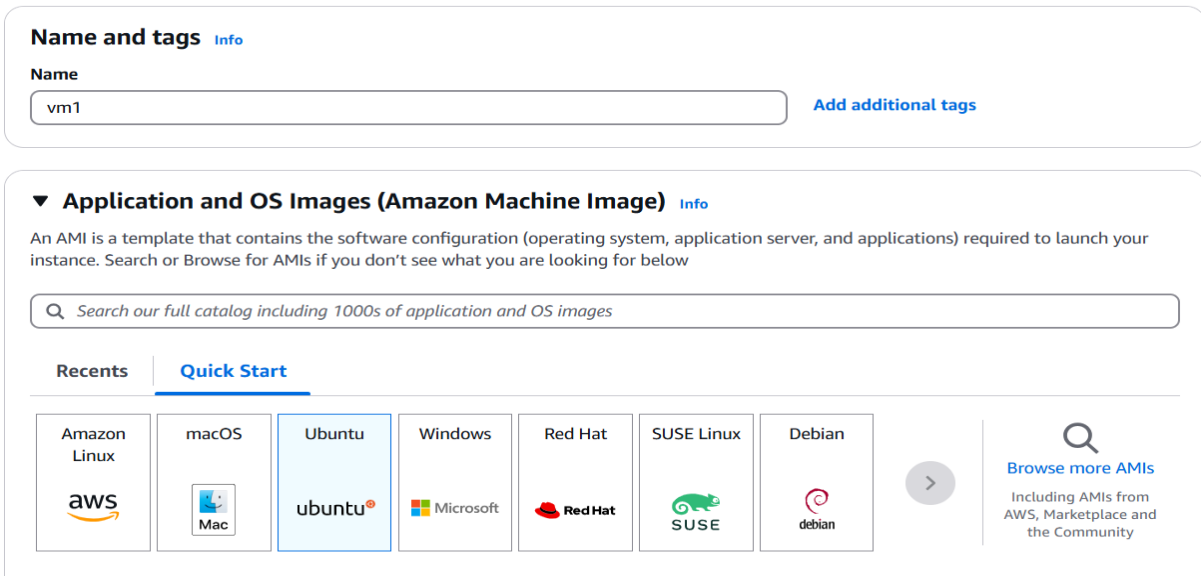
>desired capacity=2 >min desired capacity=1 >max desired capacity=4



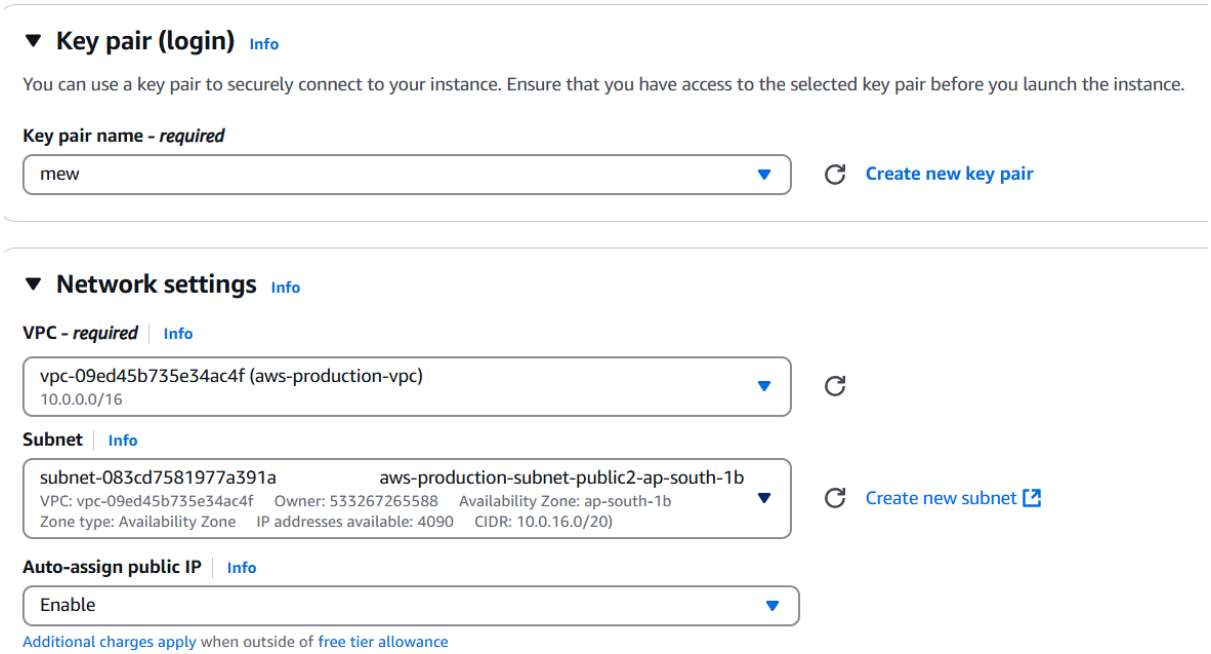
You can see auto scaling group are create



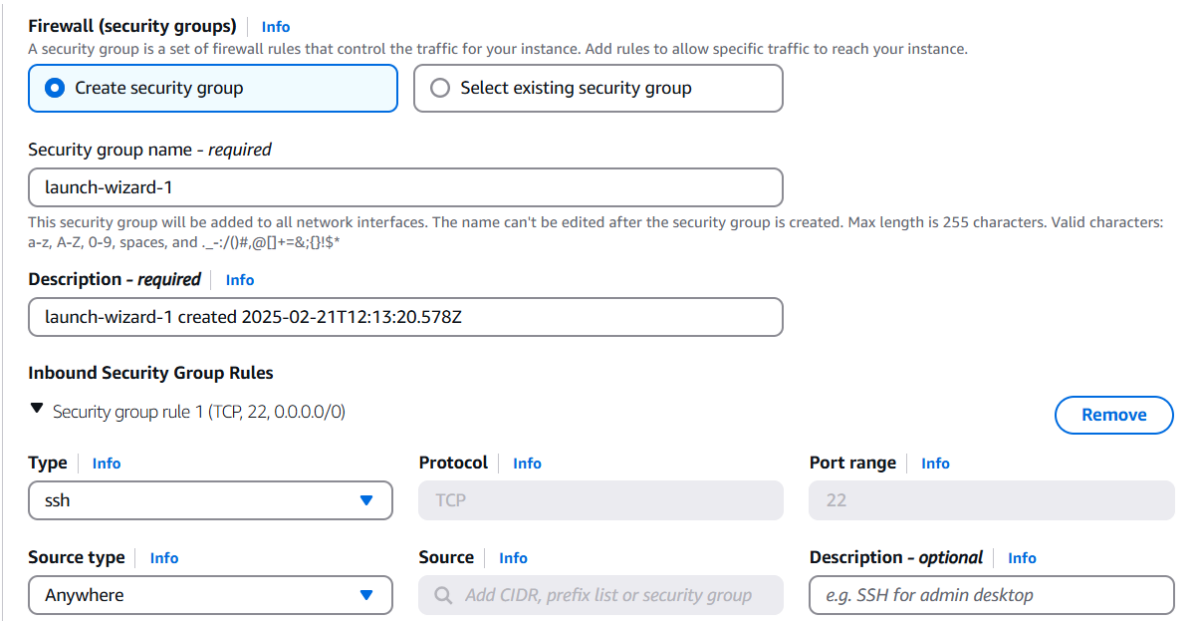
You can see 2 instance are deploy automatically.



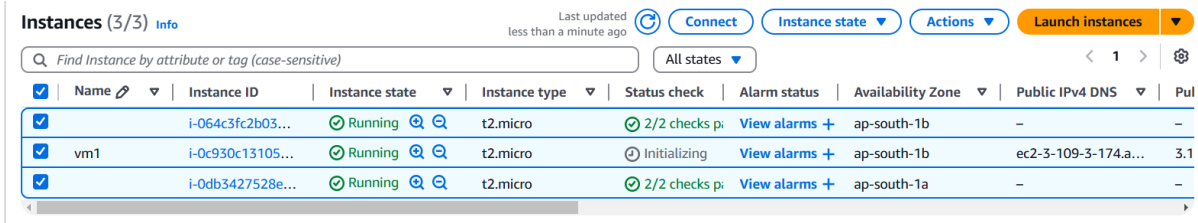
>Also create 1 instance for jump server



>With same production Vpc with public subnet and enable public Ip



>create security group with SSH rule



you can see new instances are created

>scp -i /users\downloads\abc.pem c:\ users\downloads\abc.pem ubuntu@1.24.23.5: /home/ubuntu

```
C:\Users\kharv\Documents>scp -i /Users\kharv\Downloads\pem C:\Users\kharv\Downloads\pem ubuntu@3.109.3.174
        1 file(s) copied.
```

Copy in ubuntu machine

```
C:\Users\kharv\Downloads>ssh -i  pem ubuntu@3.109.3.174
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1021-aws x86_64)
```

>ssh -i abc.pem ubuntu@1.24.23.5

```
ubuntu@ip-10-0-29-140:~$ ssh -i pem ubuntu@10.0.152.146
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1021-aws x86_64)
```

you can see you access the machine

```
ubuntu@ip-10-0-152-146:~$ sudo apt install apache2
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libapr1t64 libaprutil1-dbd-sqlite3 libaprutil1-ldap libaprutil1t64
  liblua5.4-0 ssl-cert
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom www-browser
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapr1t64 libaprutil1-dbd-sqlite3 libaprutil1-ldap libaprutil1t64
```

>sudo apt install apache2

```
ubuntu@ip-10-0-139-110:/var/www/html$ cd /var/www/html
ubuntu@ip-10-0-139-110:/var/www/html$ sudo rm -rf *
ubuntu@ip-10-0-139-110:/var/www/html$ ls
ubuntu@ip-10-0-139-110:/var/www/html$ sudo nano index.html
ubuntu@ip-10-0-139-110:/var/www/html$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.0.25.174 - - [21/Feb/2025 13:53:10] "GET / HTTP/1.1" 200 -
10.0.25.174 - - [21/Feb/2025 13:53:12] "GET / HTTP/1.1" 304 -
10.0.25.174 - - [21/Feb/2025 13:53:12] "GET / HTTP/1.1" 304 -
10.0.25.174 - - [21/Feb/2025 13:53:13] "GET / HTTP/1.1" 304 -
10.0.25.174 - - [21/Feb/2025 13:53:14] "GET / HTTP/1.1" 304 -
10.0.25.174 - - [21/Feb/2025 13:53:14] "GET / HTTP/1.1" 304 -
```

>cd /var/www/html/

>sudo rm -rf *

>sudo nano index.html link - krishna-20802/-VPC-with-Public-Private-Subnet-in-Production

>python3 -m http.server 8000

>get ssh in another instance

```
ubuntu@ip-10-0-29-140:~$ ssh -i pem ubuntu@10.0.152.146
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1021-aws x86_64)
```

>cd /var/www/html/

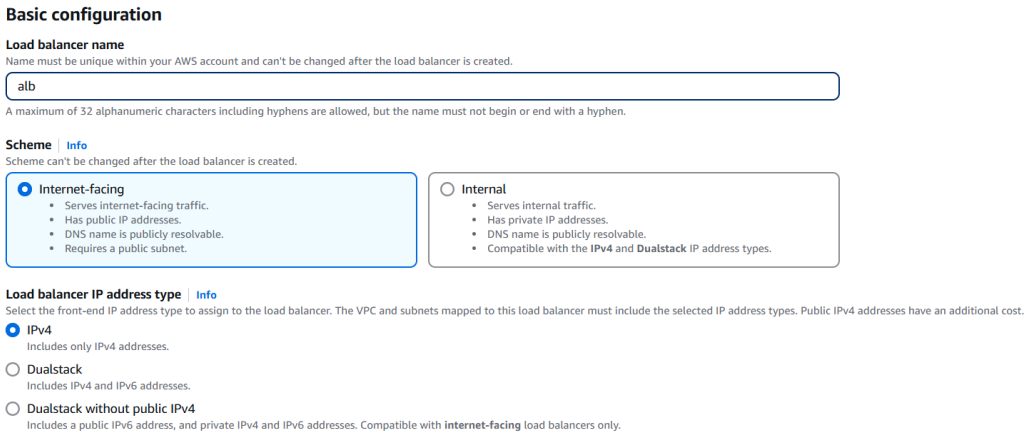
>sudo rm -rf *

>sudo nano index.html

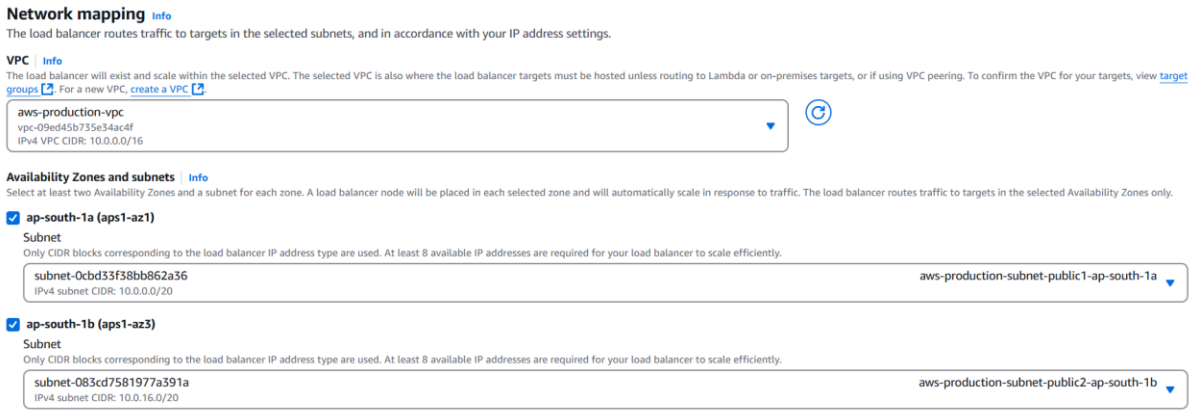
>python3 -m http.server 8000

```
ubuntu@ip-10-0-152-146:/var/www/html$ sudo rm -rf *
ubuntu@ip-10-0-152-146:/var/www/html$ ls
ubuntu@ip-10-0-152-146:/var/www/html$ sudo nano index.html
ubuntu@ip-10-0-152-146:/var/www/html$ ubuntu@ip-10-0-152-146:/var/www/html$
ubuntu@ip-10-0-152-146:/var/www/html$ ls
index.html
ubuntu@ip-10-0-152-146:/var/www/html$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.0.25.174 - - [21/Feb/2025 13:51:47] "GET / HTTP/1.1" 200 -
10.0.3.180 - - [21/Feb/2025 13:51:57] "GET / HTTP/1.1" 200 -
10.0.25.174 - - [21/Feb/2025 13:52:17] "GET / HTTP/1.1" 200 -
10.0.3.180 - - [21/Feb/2025 13:52:27] "GET / HTTP/1.1" 200 -
10.0.25.174 - - [21/Feb/2025 13:52:47] "GET / HTTP/1.1" 200 -
10.0.3.180 - - [21/Feb/2025 13:52:57] "GET / HTTP/1.1" 200 -
```

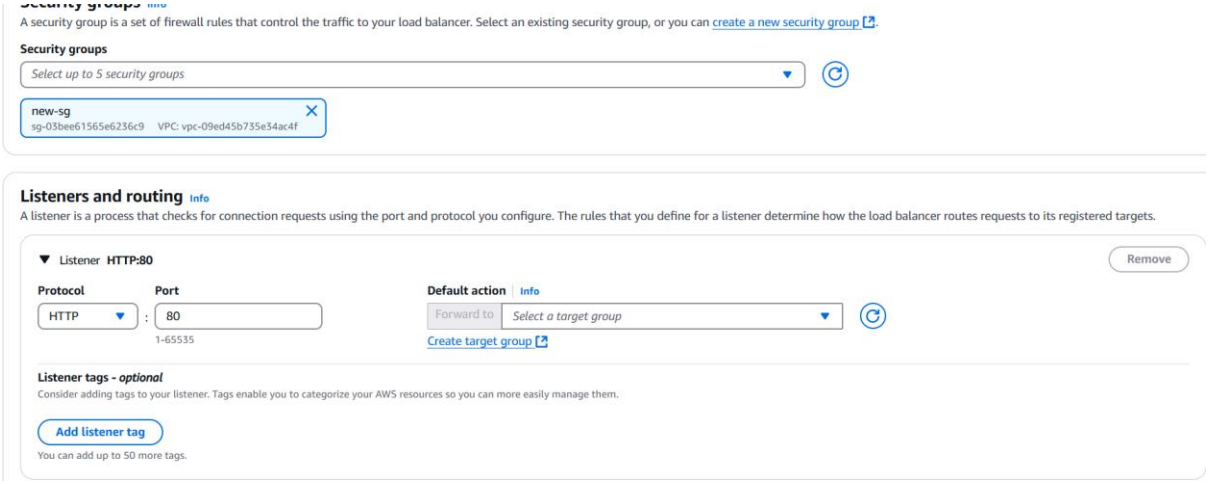
>Create Application Load Balancer with internet facing



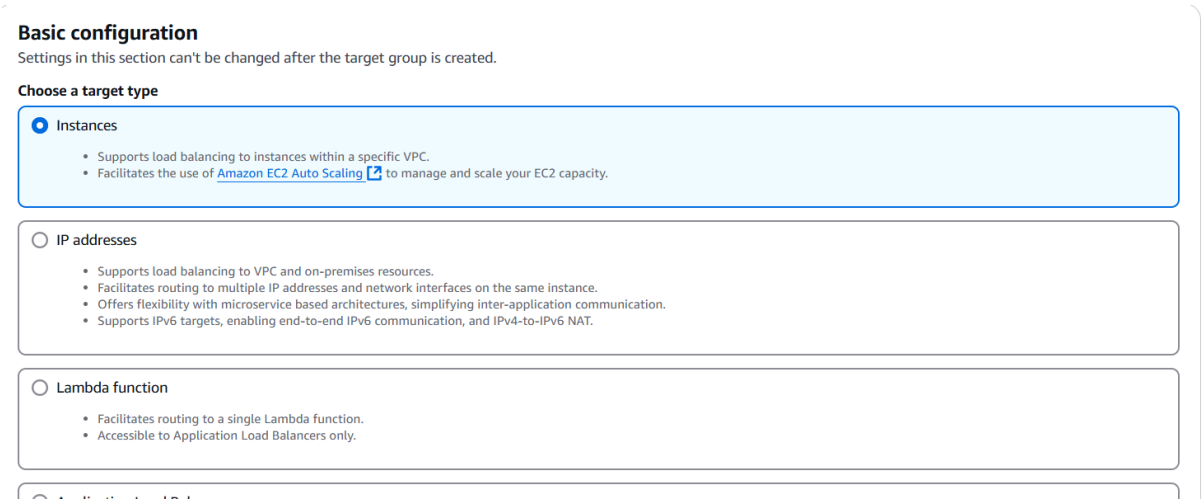
>select production vpc and public subnet

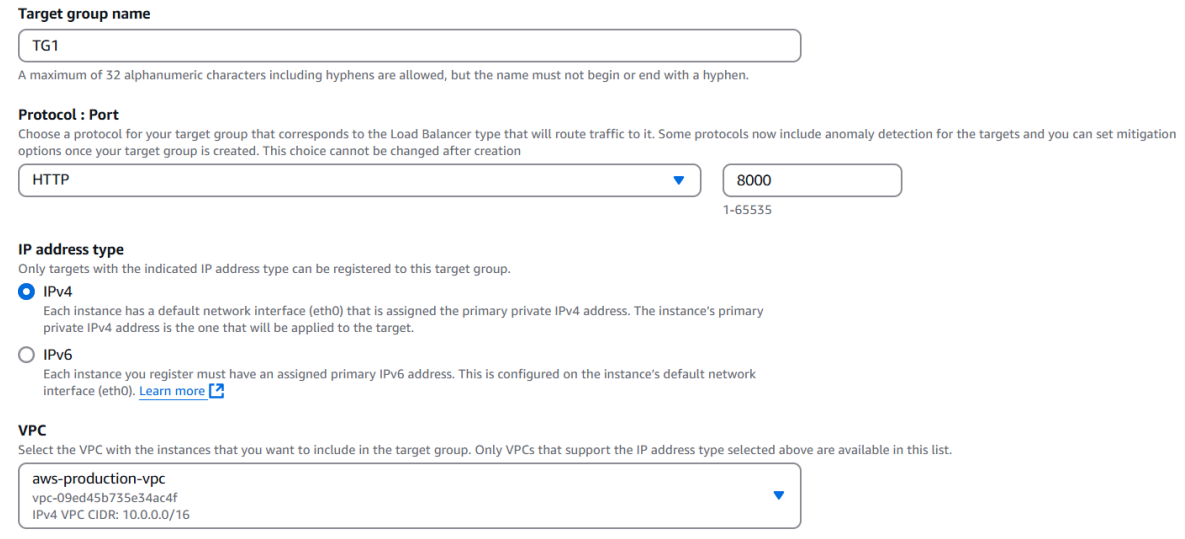


>Select Security group

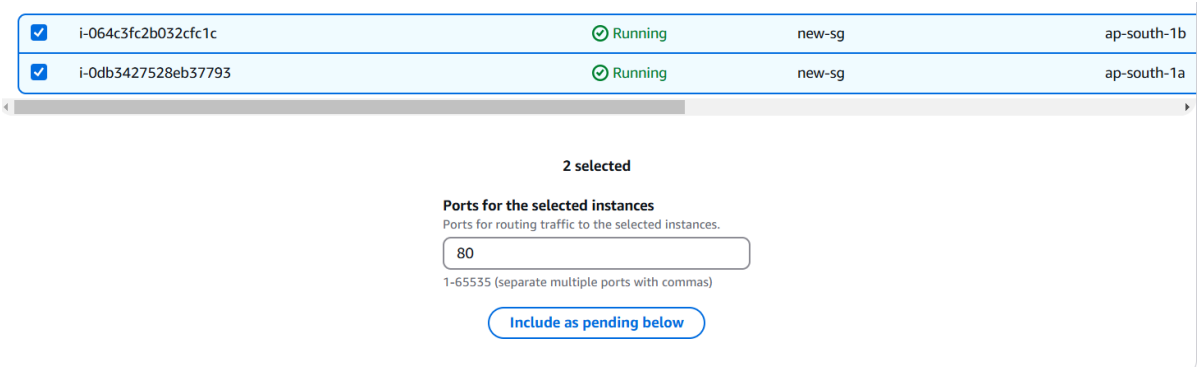


>create target group with port 8000





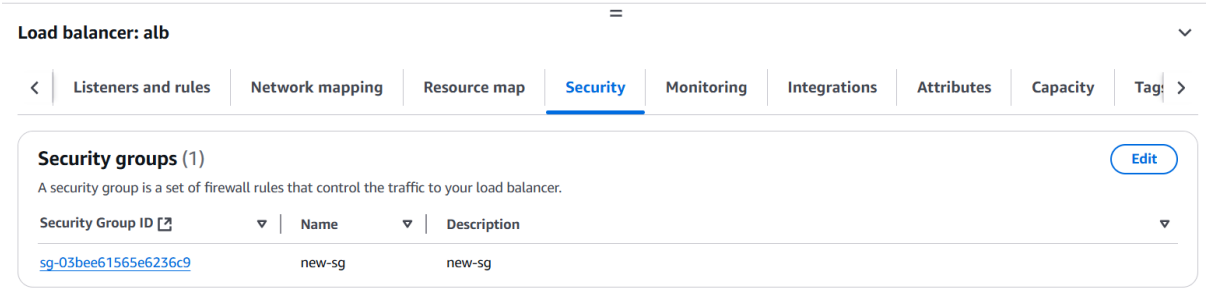
>select 2 and include as pending below



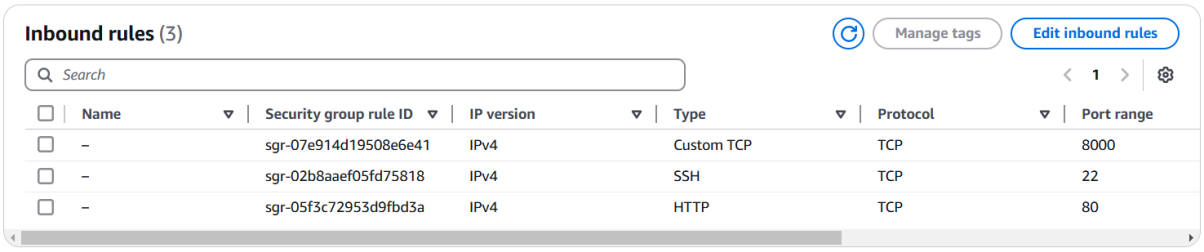
>copy the alb dns and paste it on chrome

You don’t see any website

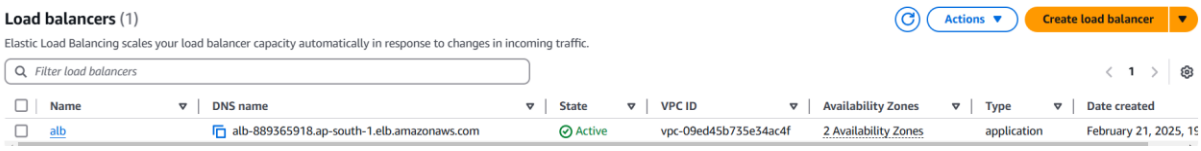
>go to alb security click on that



>add inbound rule 8000 port

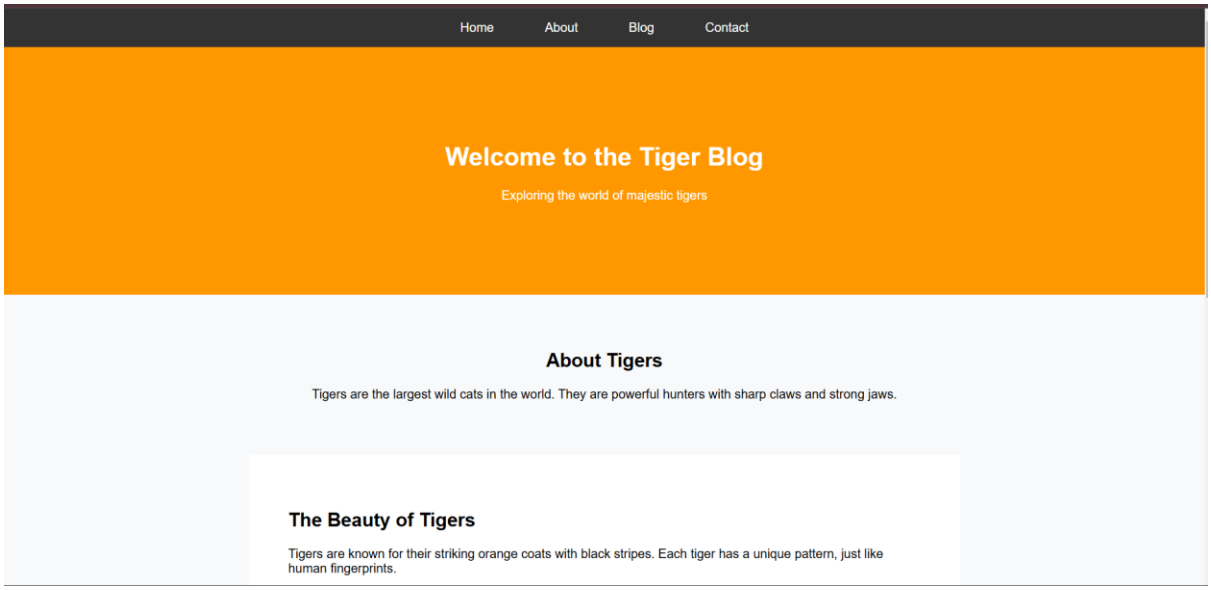


>copy the alb DNS

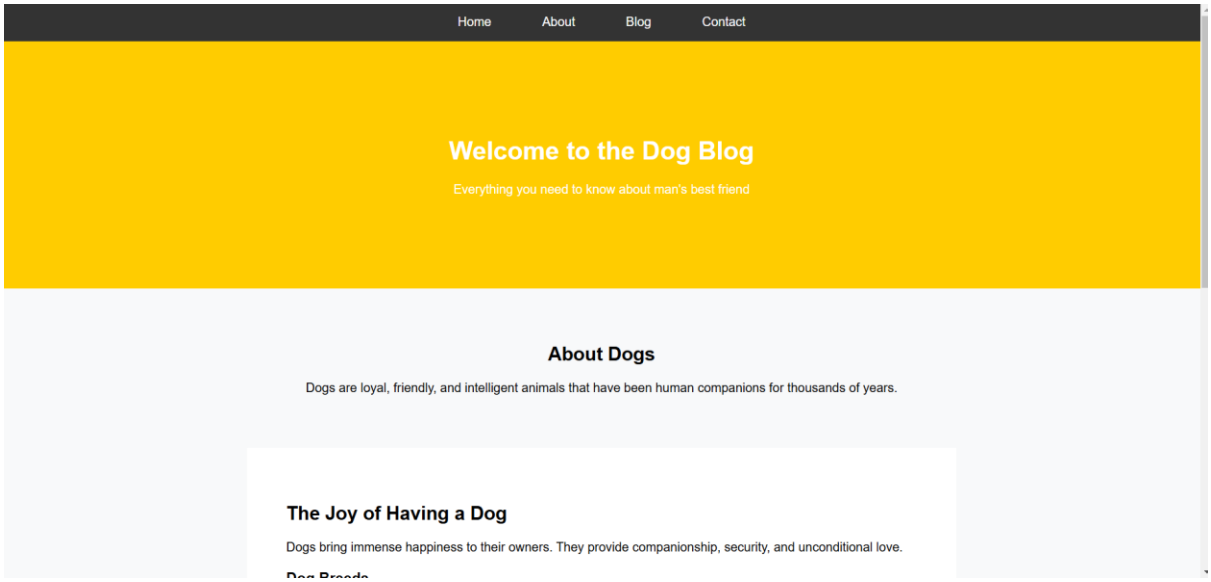


>paste it on chrome

You see your web-site is LIVE



>Refresh the website you can see your another website



Well done!!!!!

Our project is done!!!!!!

Your web site work on private subnet.

We deploy our instance in private subnet. The private subnet does not face internet directly.

In This Project I Use EC2, VPC, Internet Gateway, Route Table, Nat Gateway, Elastic IP, HTML Code, Auto-Scaling Group, Target Group, Application Load Balancer, Launch Templates.