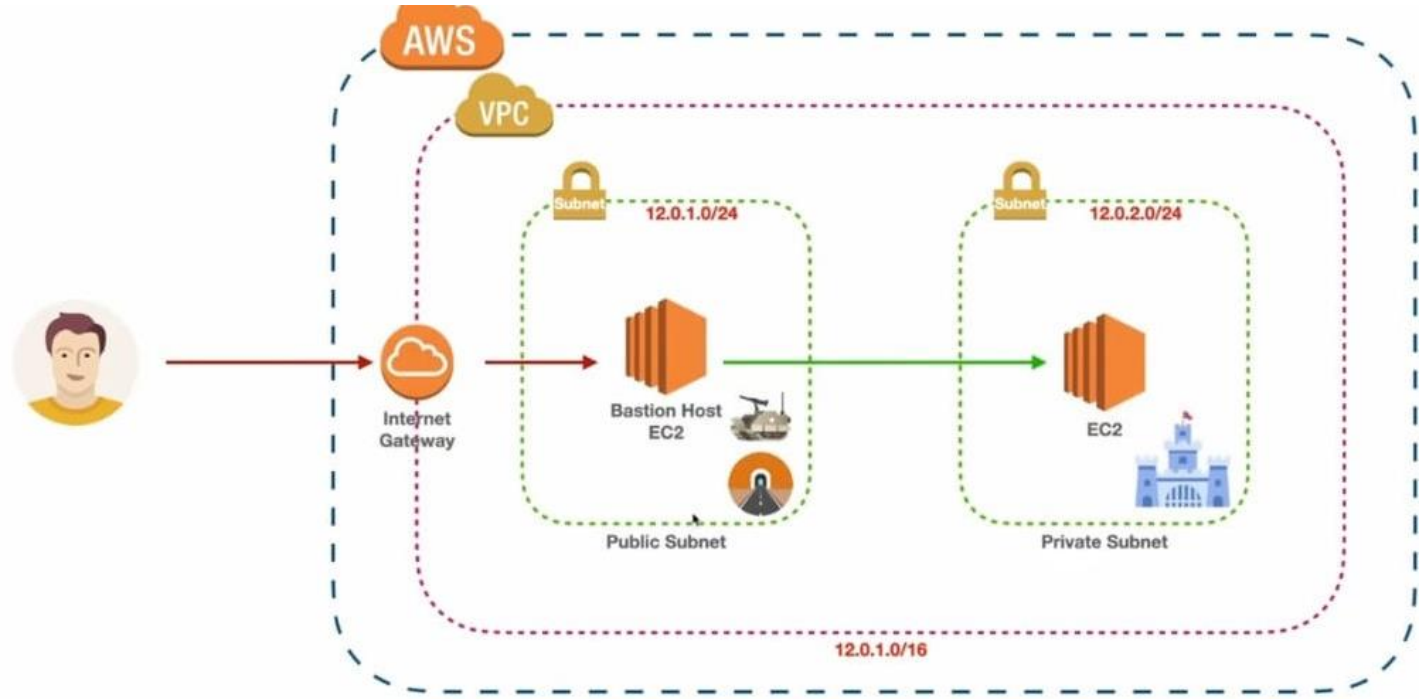
AWS Bastion Host:

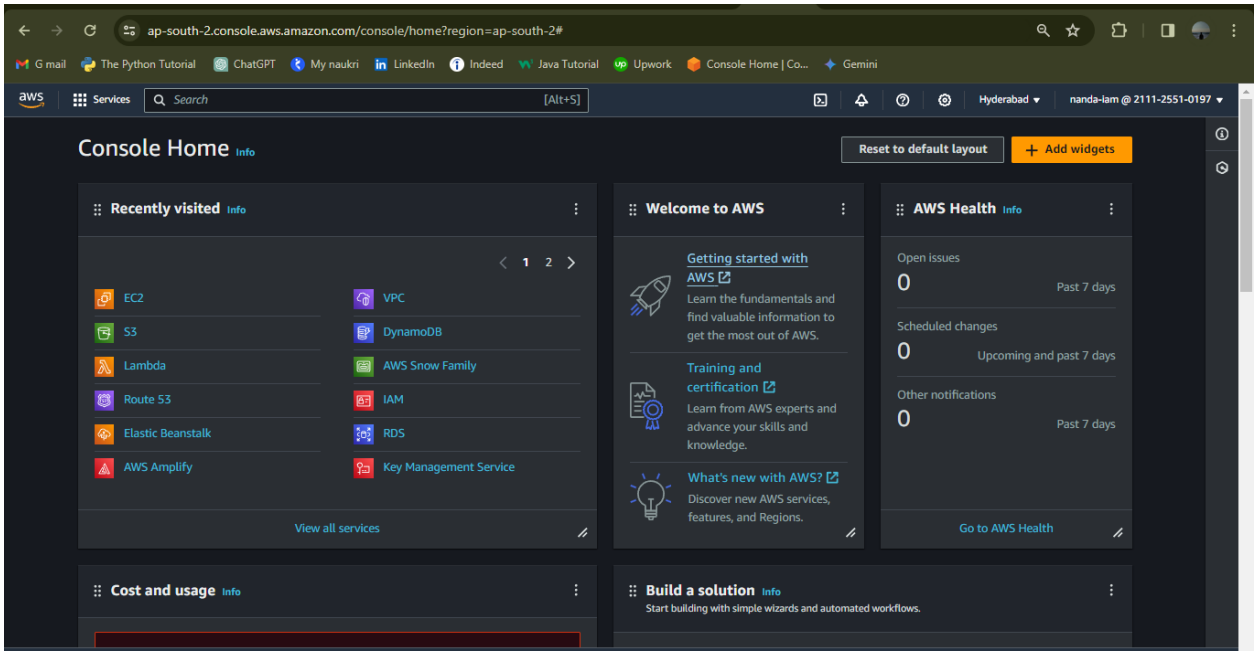
Created By : Nandakumar

This solution sets up the following



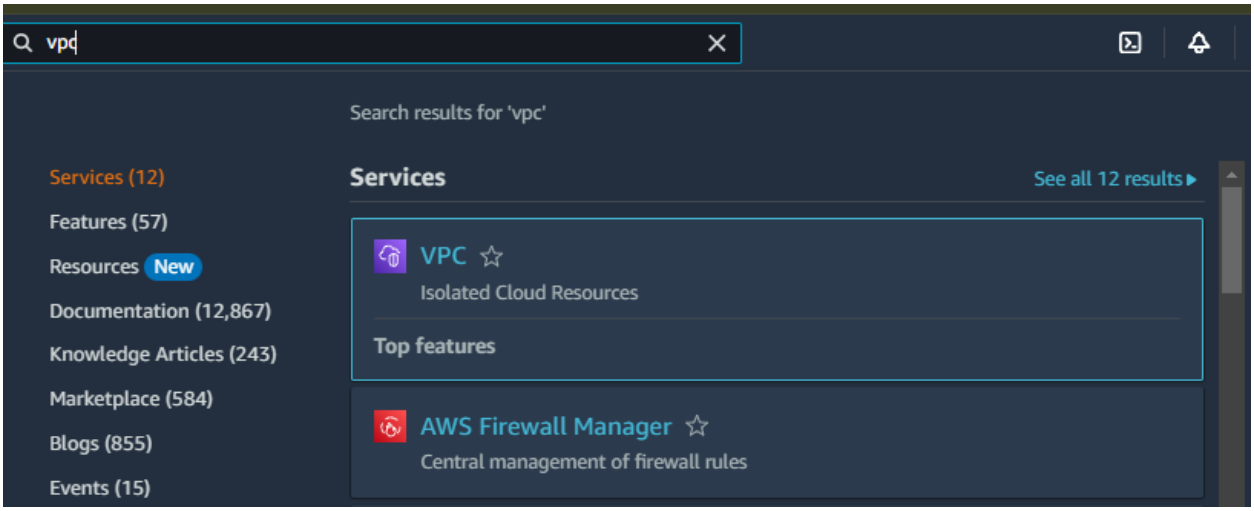
Step 1: Log in to AWS Console

- Navigate to the AWS Management Console at https://console.aws.amazon.com/.
- Log in using your credentials.



Step 2: Go to the VPC Dashboard

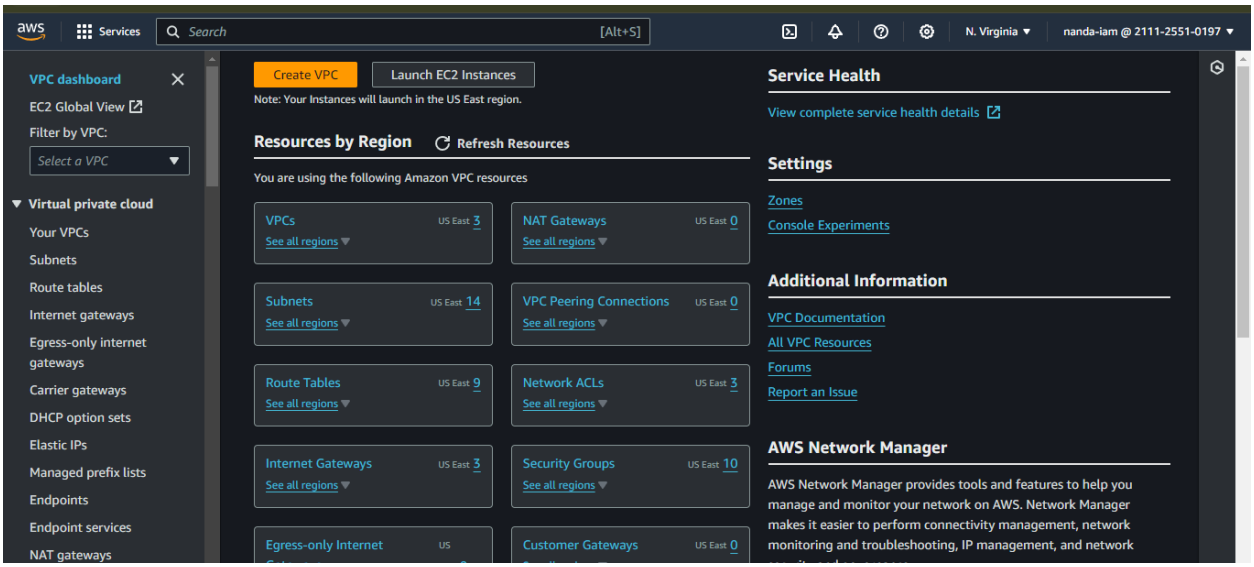
- In the AWS Management Console, search for "VPC" in the services search bar, or navigate to the Networking & Content Delivery section and click on "VPC".



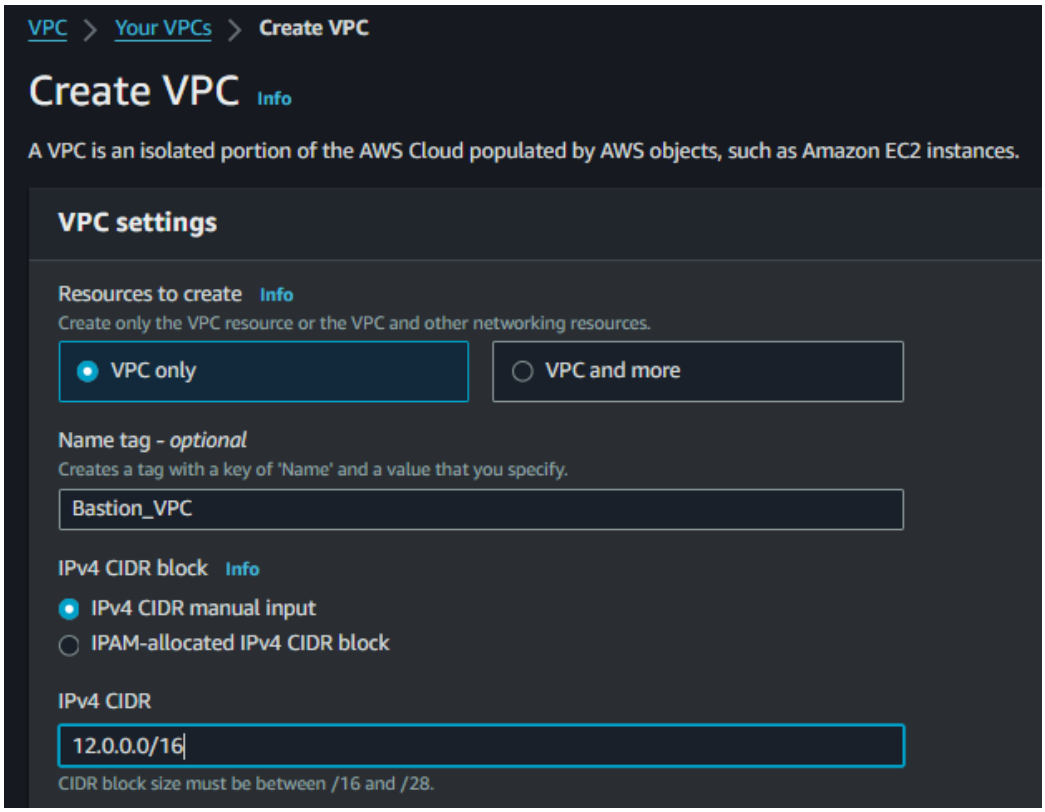
- You'll land on the VPC Dashboard.

Step 3: Create a VPC

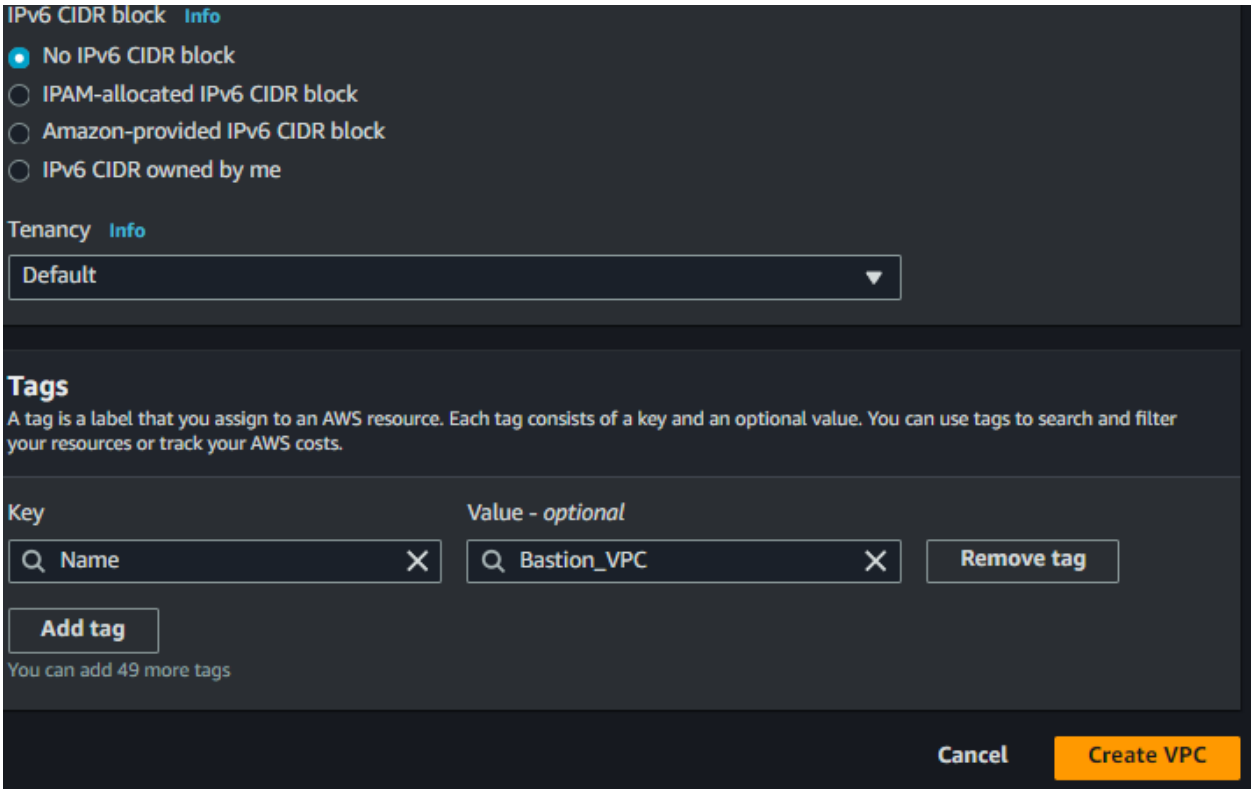
- Click on the "Create VPC" button.



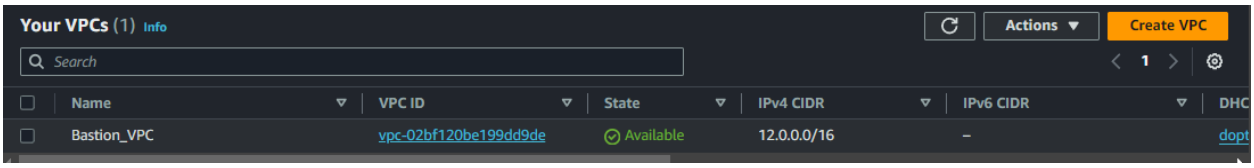
- In the Create VPC wizard, you'll be prompted to fill out details:
- Name tag: Give your VPC a descriptive name of Bastion_VPC
- IPv4 CIDR block: Define the IP address range for your VPC, e.g., 12.0.0.0/16.



- IPv6 CIDR block: Optionally, you can assign an IPv6 CIDR block.
- Tenancy: Choose default unless you have specific requirements.
- Click on "Create".

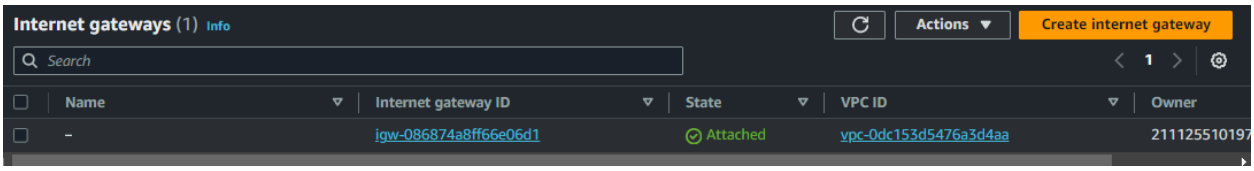


- You will see the status that your VPC is launching below.

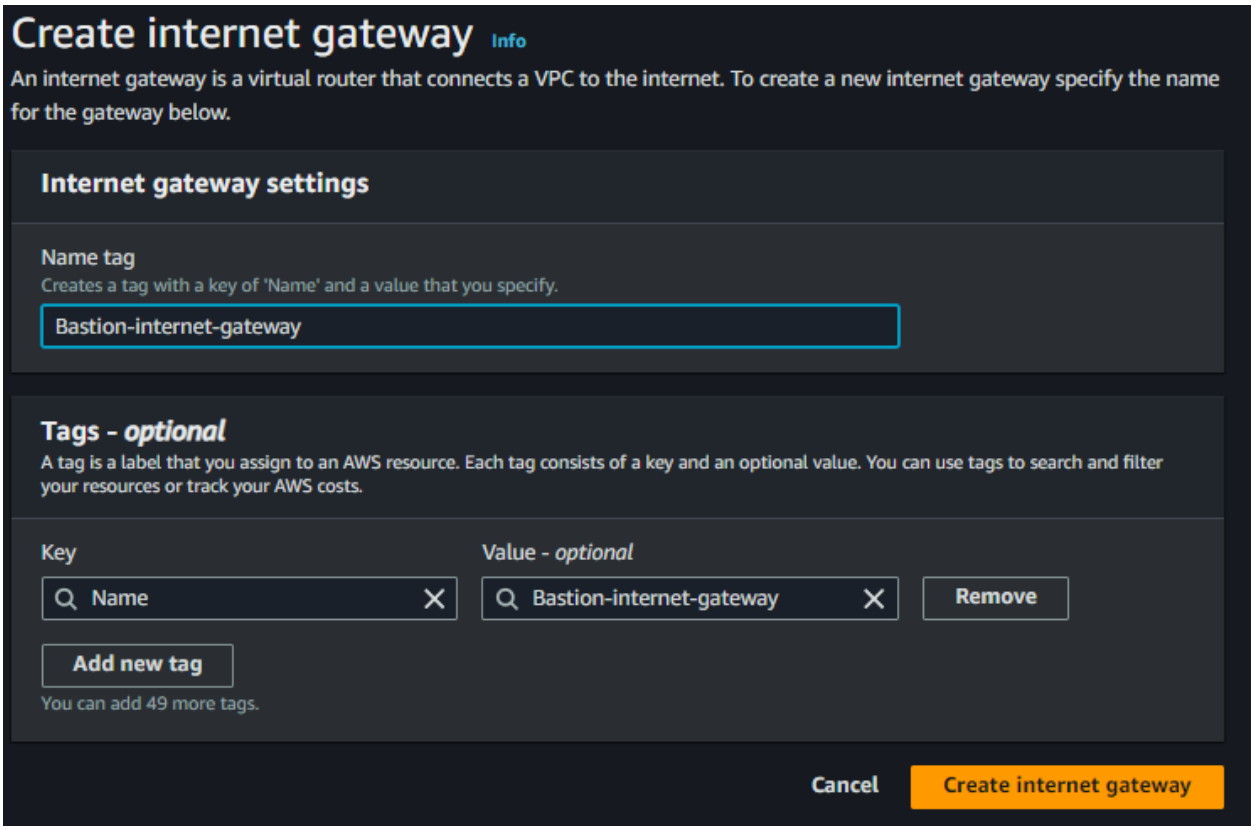


Step 4: Create Internet Gateway (IGW) (if needed)

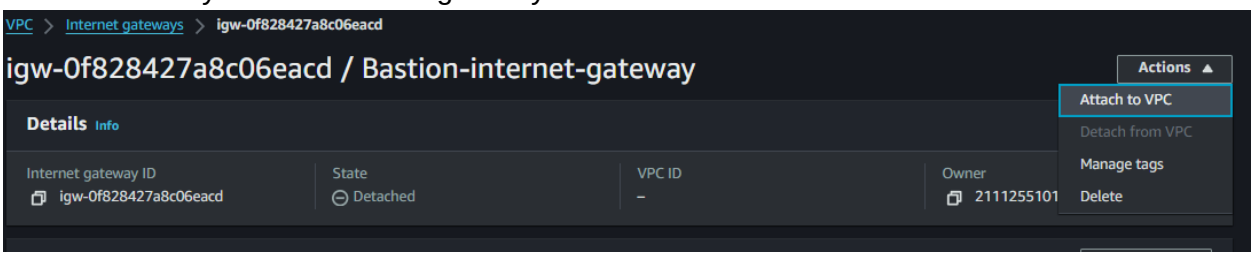
- If you want your VPC to have internet access:
- Go to "Internet Gateways" in the VPC Dashboard.
- Click on "Create internet gateway"



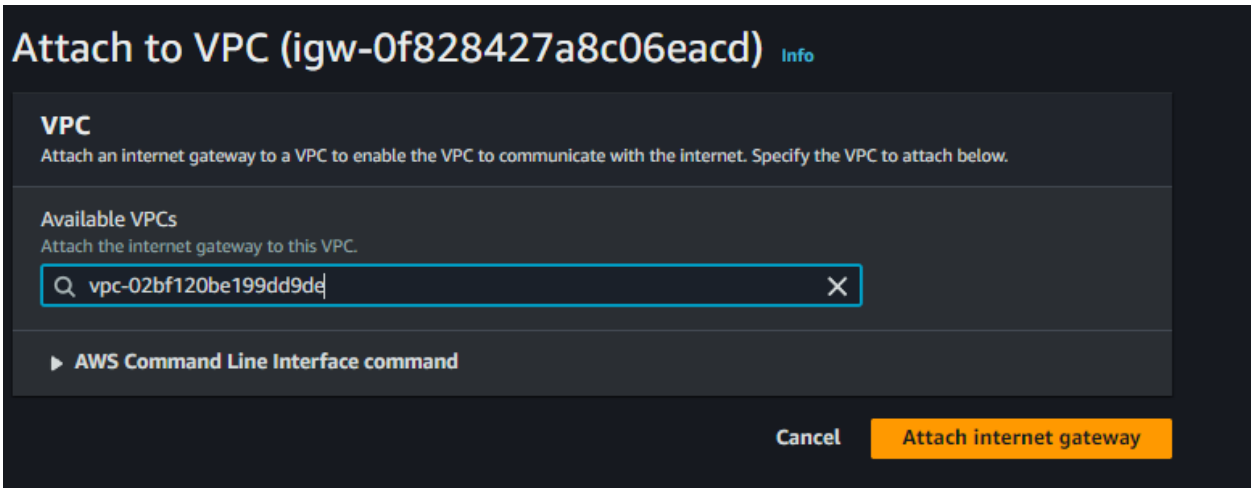
- Provide a name for the internet gateway and click on "Bastion-internet-gatway".



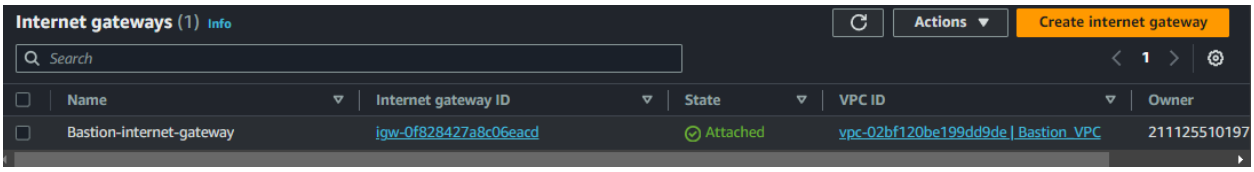
- Select the newly created internet gateway and click on "Attach to VPC".



- Choose your VPC and click on "Attach".

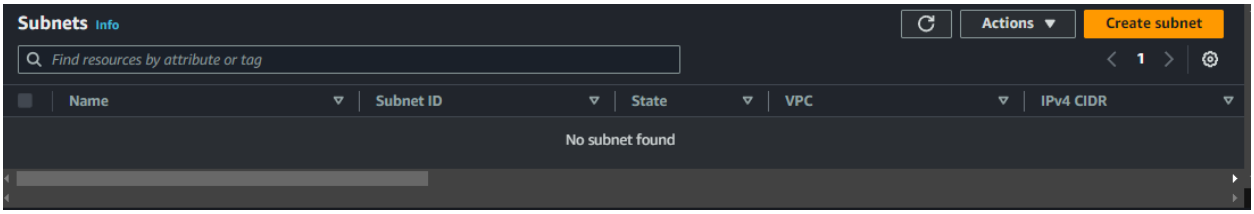


- You will see the status that your internet gateway is launching below.

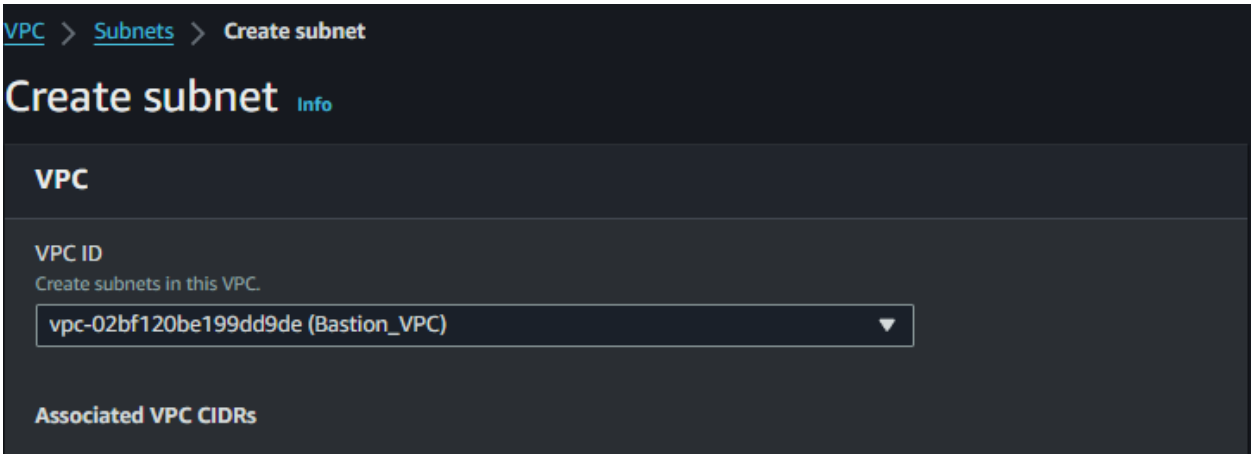


Step 5: Create Subnets

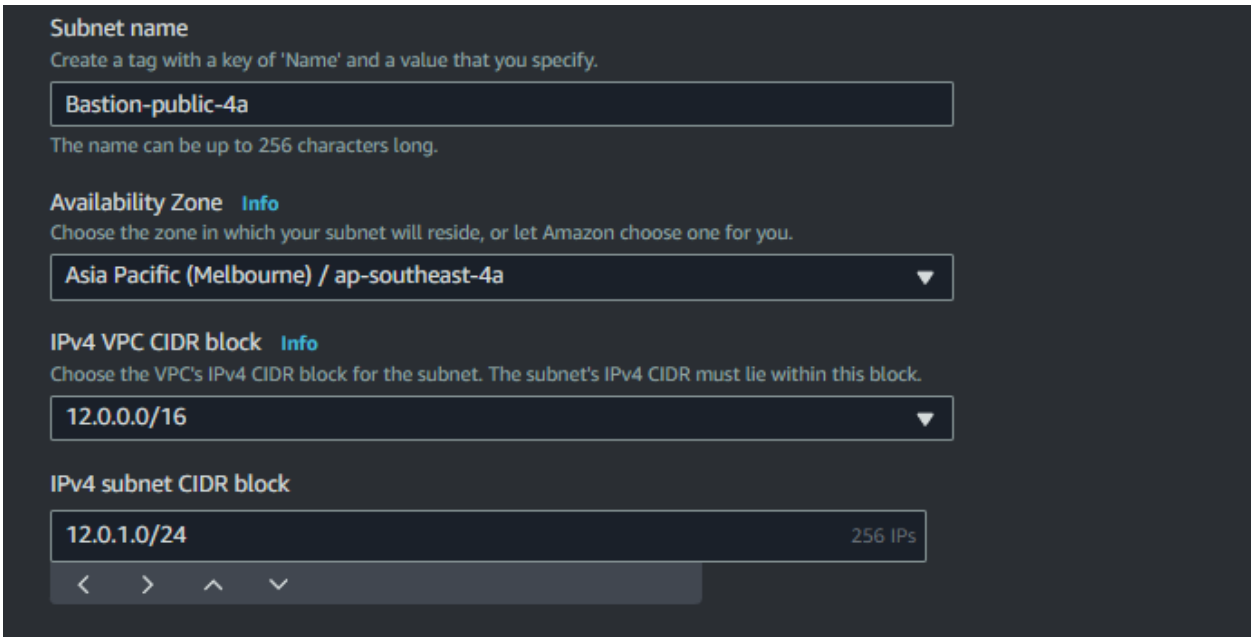
- Sure, let's create both public and private subnets within the VPC.
- After creating the VPC, click on "Subnets" in the VPC Dashboard.
- Click on "Create subnet"



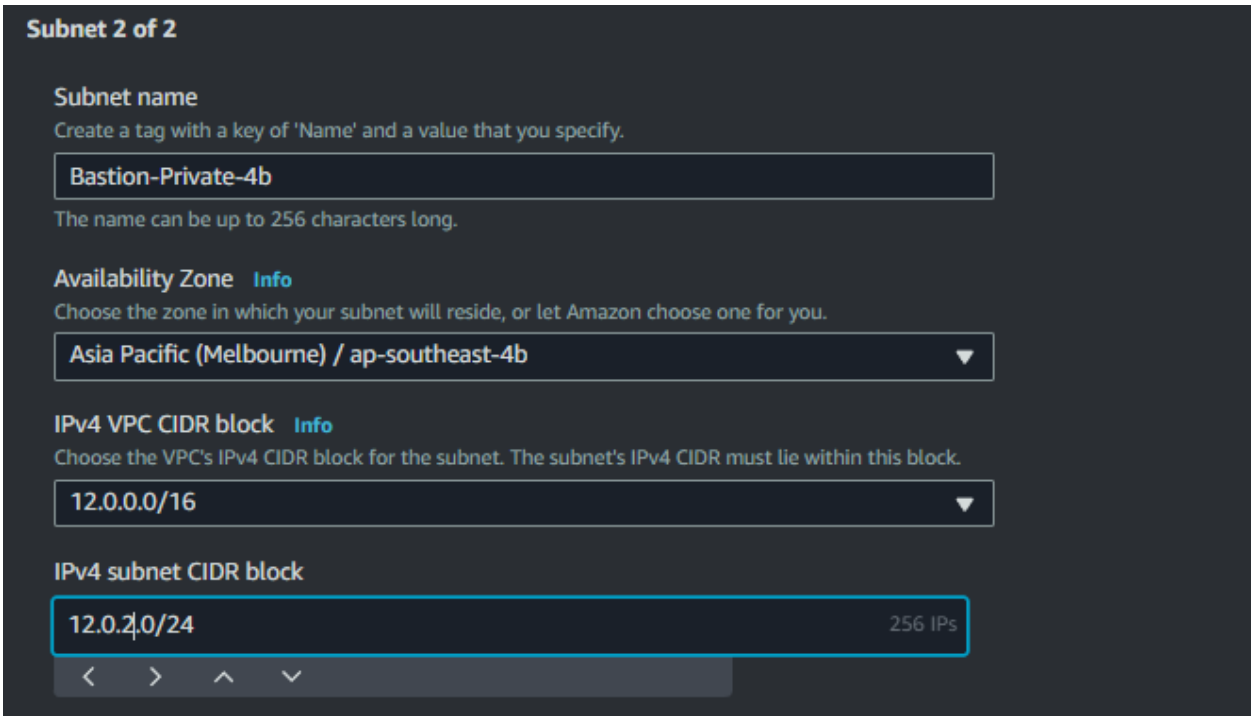
- VPC: Choose the VPC you created earlier.



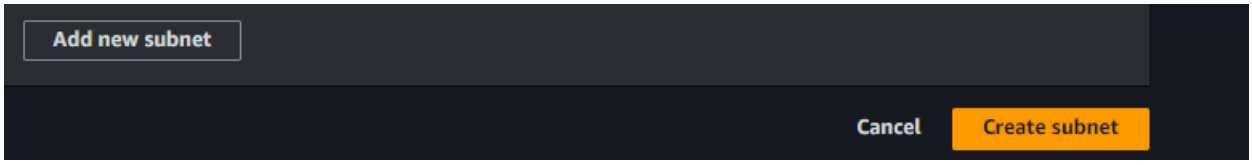
- Name Tag: Bastion-public-4a
- Availability Zone: Select an availability zone. For example, ap-southeast-4a.
- IPv4 CIDR block: Define the IP address range for the subnet within the VPC CIDR block. For instance, if your VPC CIDR block is 12.0.0.0/16, you can define your public subnet as 12.0.1.0/24.



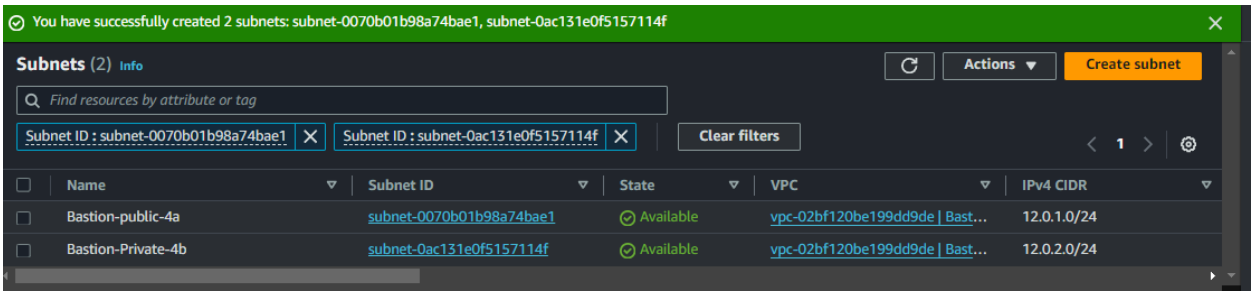
- Name Tag: Bastion-private-4b
- Availability Zone: Select an availability zone. For example, ap-southeast-4b.
- IPv4 CIDR block: Define the IP address range for the subnet within the VPC CIDR block. For instance, if your VPC CIDR block is 12.0.0.0/16, you can define your public subnet as 12.0.2.0/24.



- Click on "Create subnet".

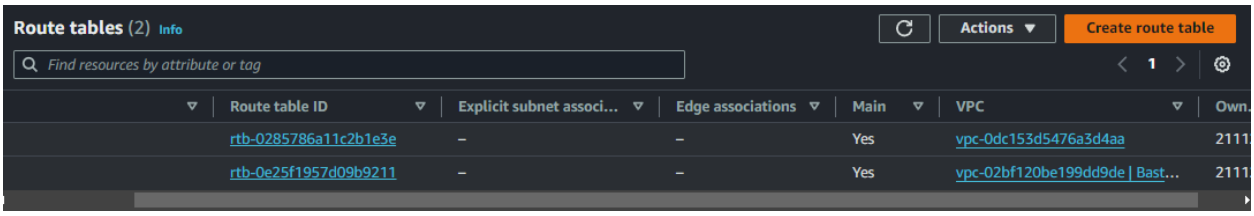


- After completing the steps, AWS will create the public and private subnet, and you'll see it listed in the Subnets section of the VPC Dashboard

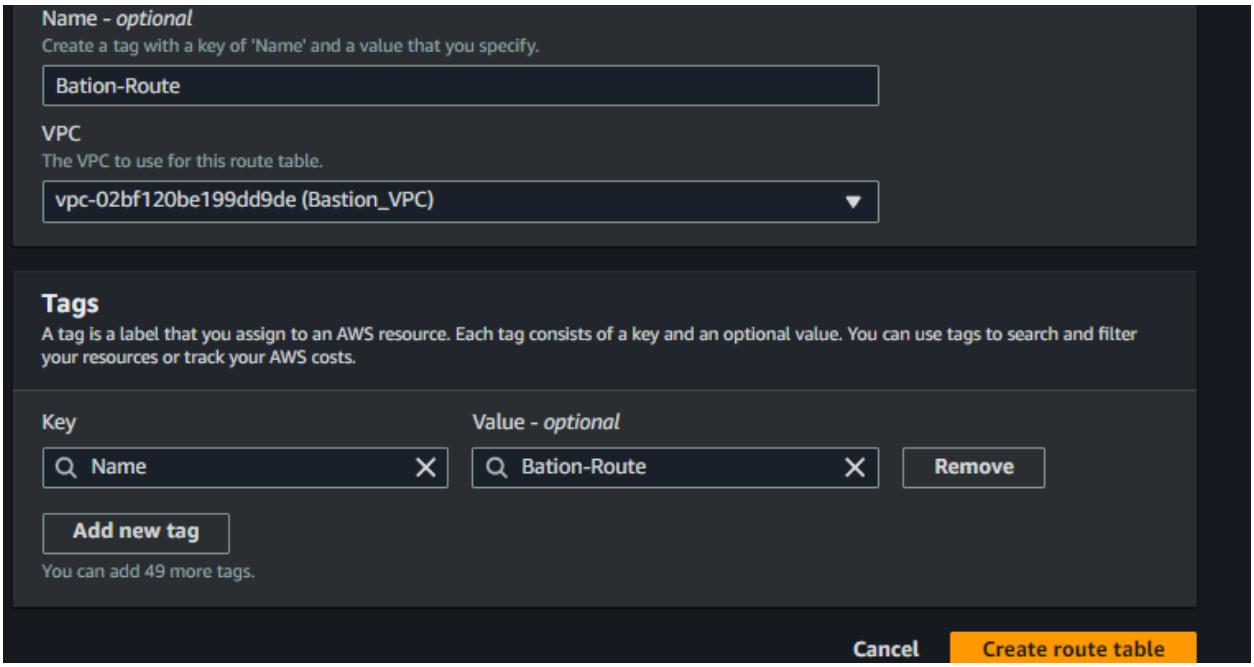


Step 6: Create Route Table of Public and Private

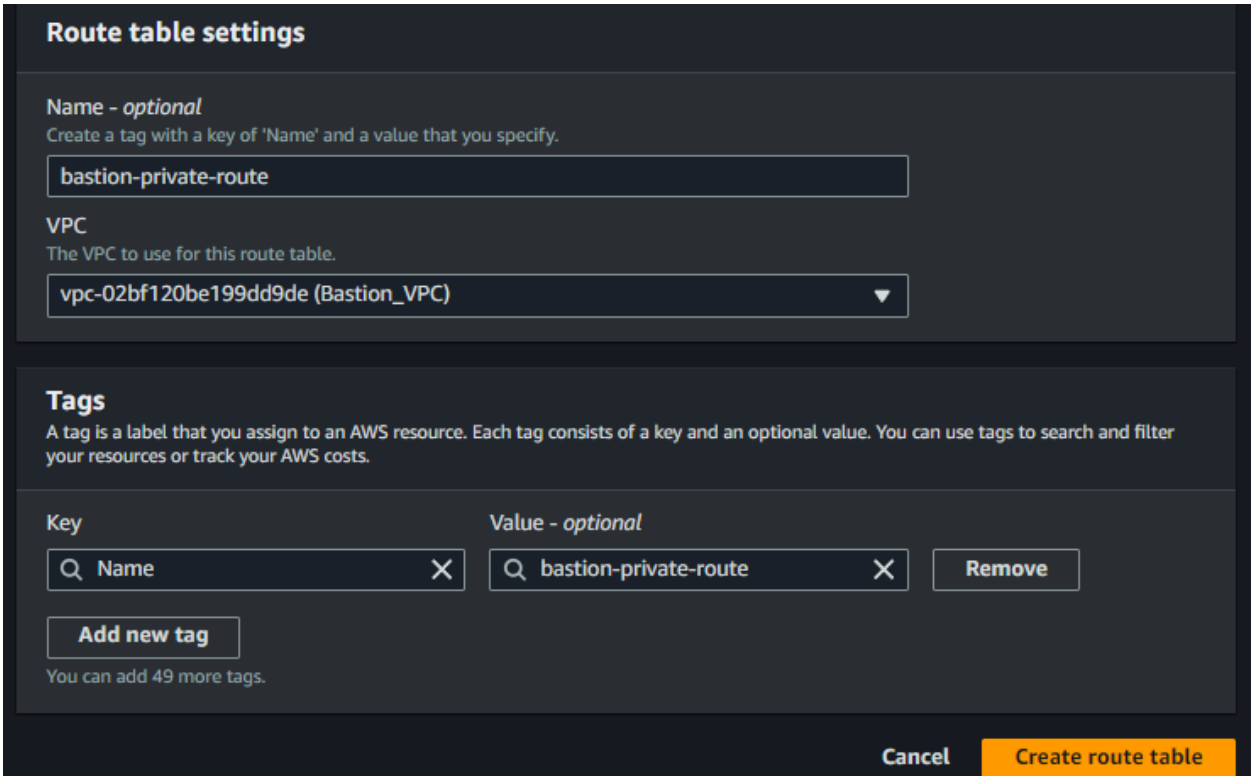
- Click on "Create route table".



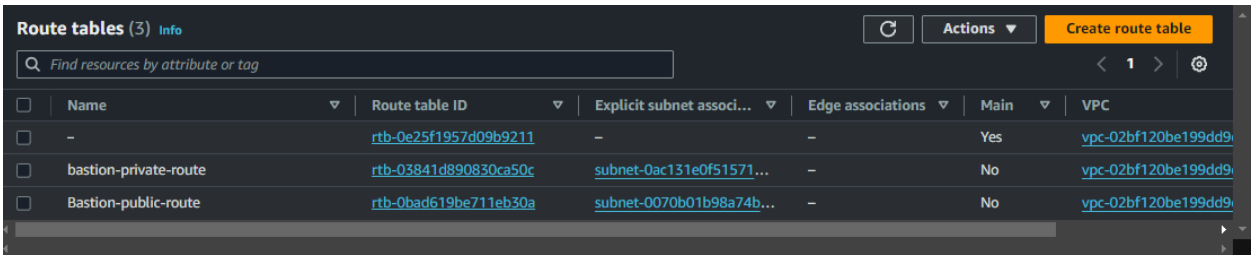
- Provide a name for the of Bastion-public-route
- Then select your VPC.
- Click on "Create".



- Provide a name for the of Bastion-Private-route
- Then select your VPC.
- Click on "Create".

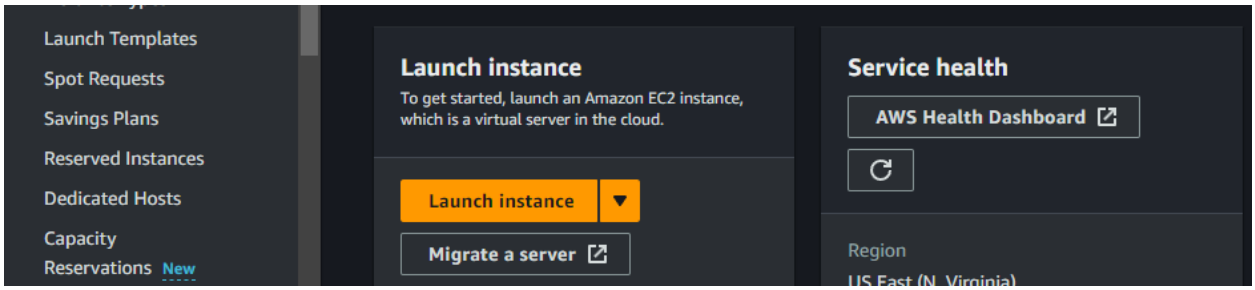


You will see the status that your Creating Route table is below.

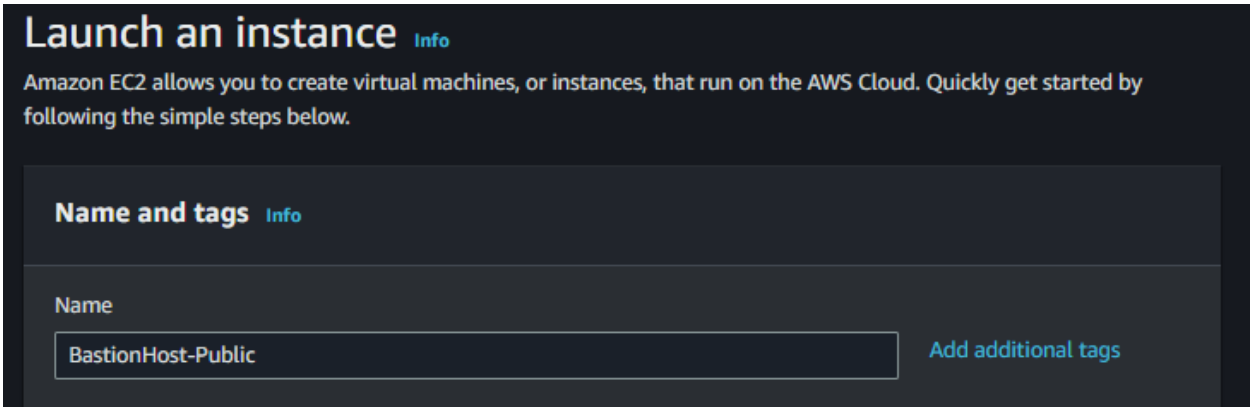


Step 7: Create instance Public and Private

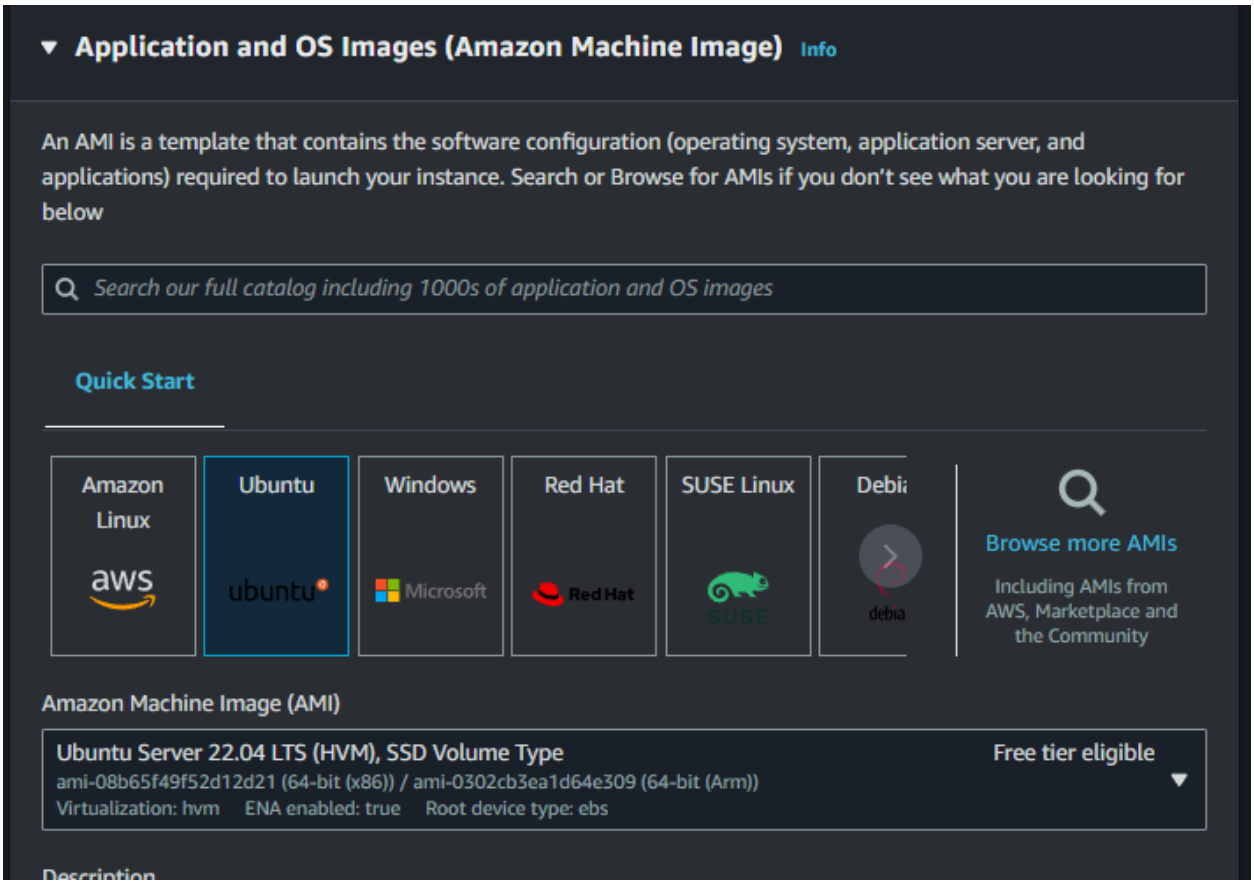
- Go to the EC2 Dashboard by searching for "EC2" in the services search bar or navigating to Compute > EC2.
- In the EC2 Dashboard, click on the "Launch Instance" button.



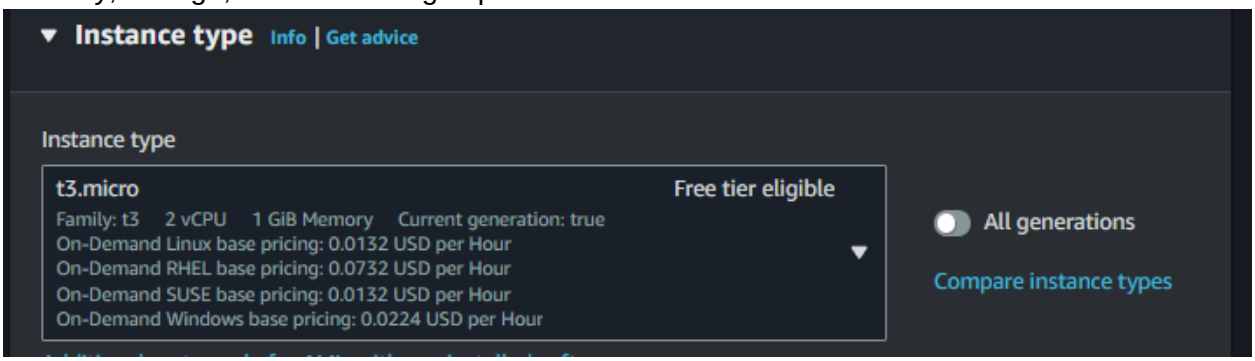
- Provide a name for the of BastionHost-Public



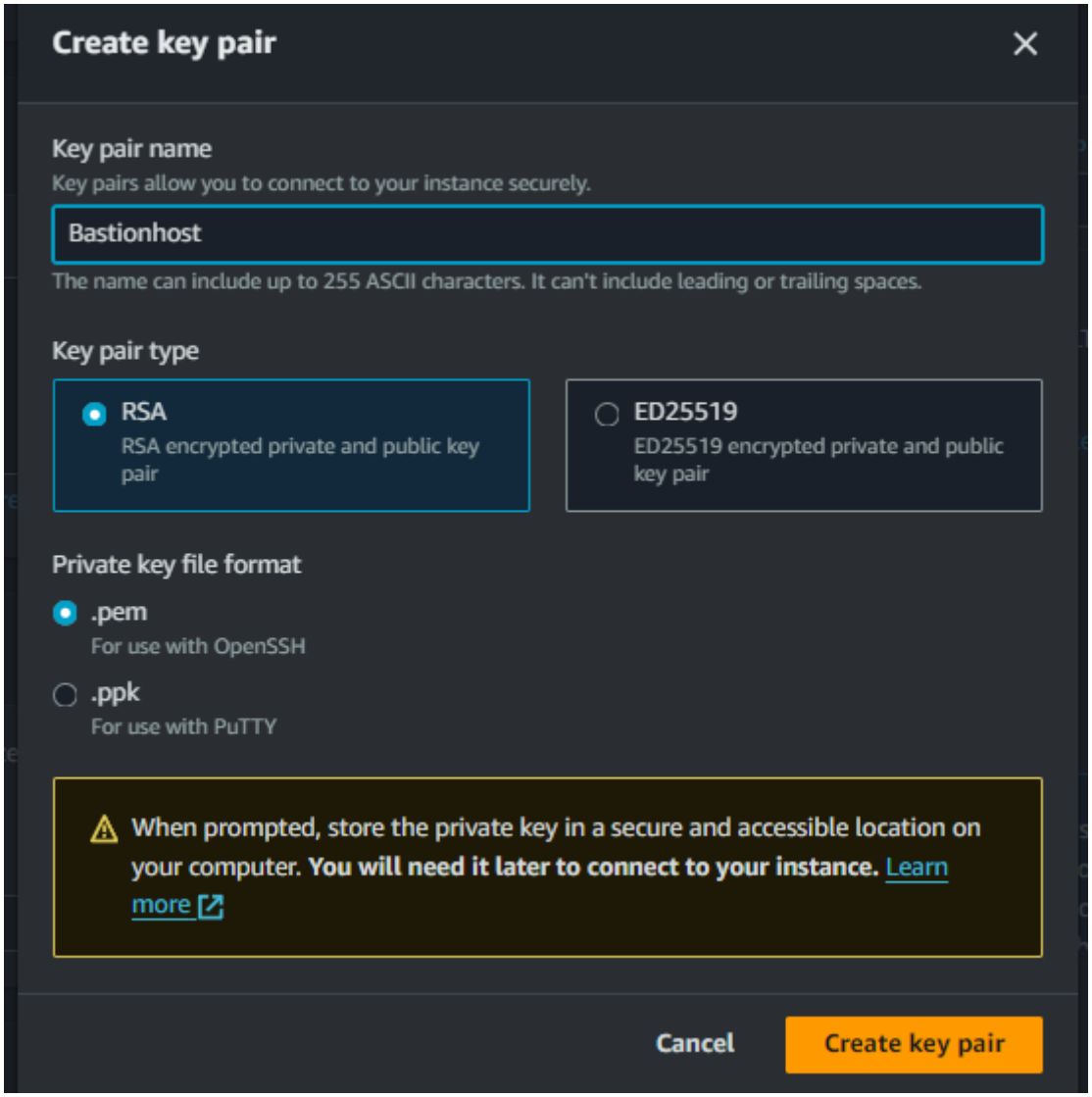
- Choose an Amazon Machine Image (AMI) based on your requirements. You can select from the AWS Marketplace, AWS Community AMIs, or your own custom AMIs.
- Click on "Select" once you've chosen an Ubuntu AMI

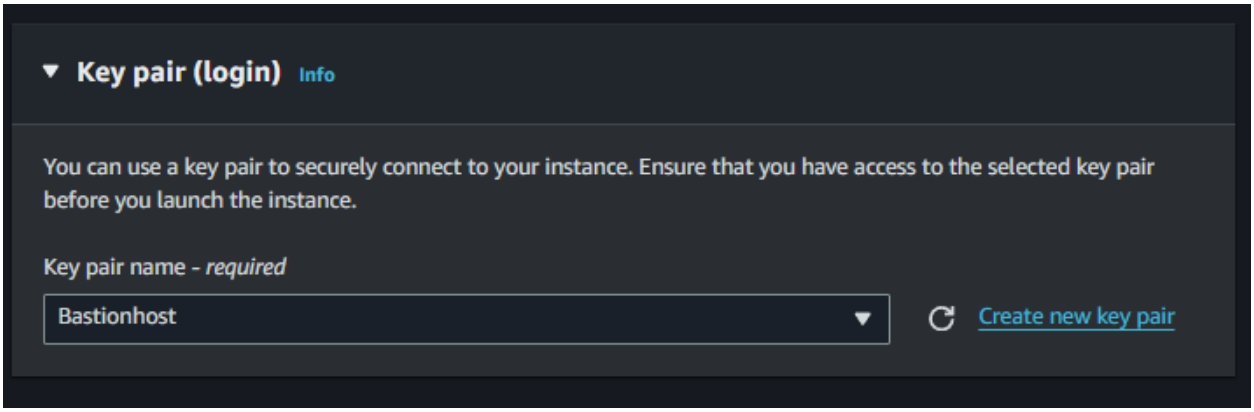


- Select an instance type based on your workload requirements. Instances vary in terms of CPU, memory, storage, and networking capabilities.

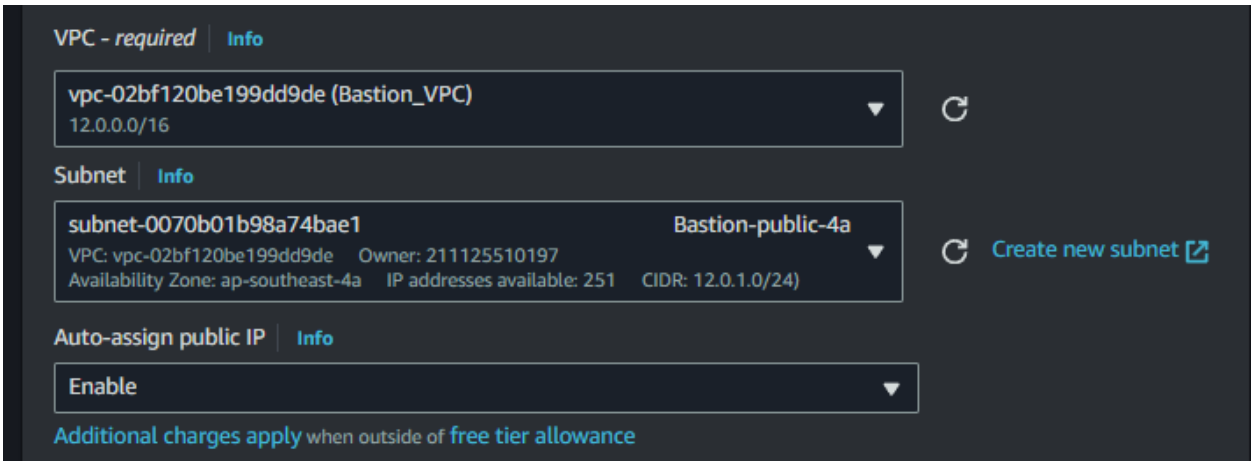


- Create a new key pair to securely connect to your instance via SSH.
- Provide a Key pair name is Bastionhost
- Key Pair type is RSA
- Key file format is .pem
- Click on "Create Key pair"

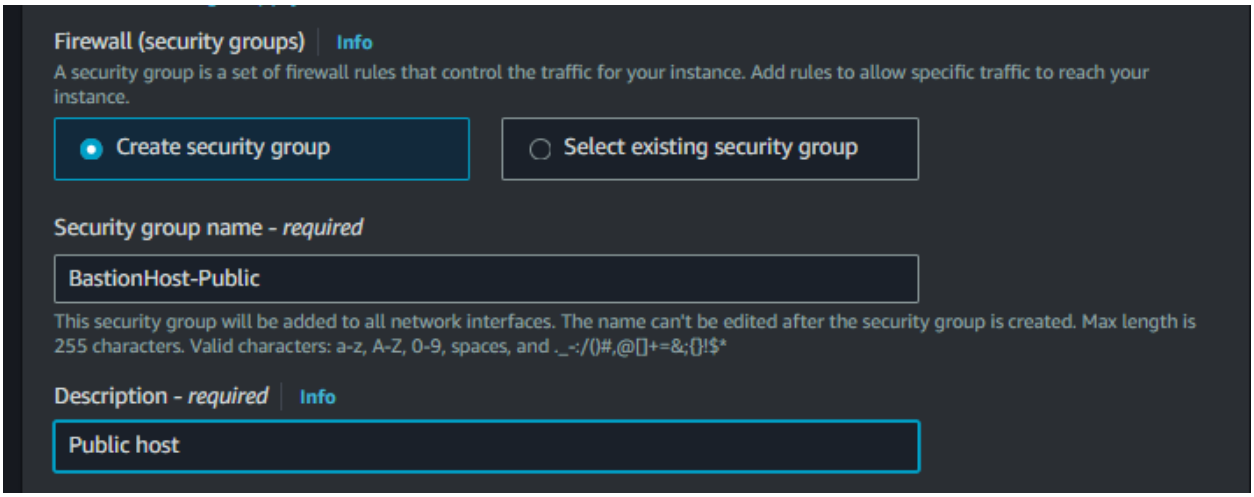




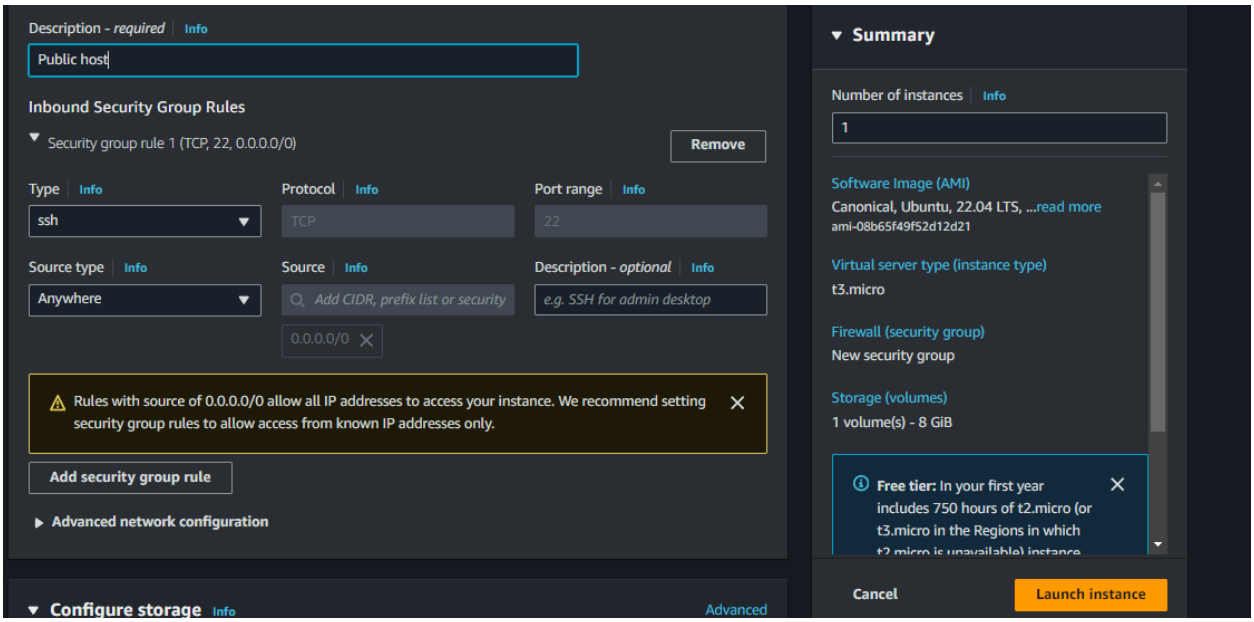
- Configure Network Settings
- Select the VPC you want to use from the Bastion_VPC.
- Choose the subnet within the selected the Bastion-public-4a
- Auto-assign Public IP: Choose whether you want to assign a public IP address to the instance. Options are "Use subnet setting (Enable or Disable)" or "Enable" or "Disable" specifically for this instance.



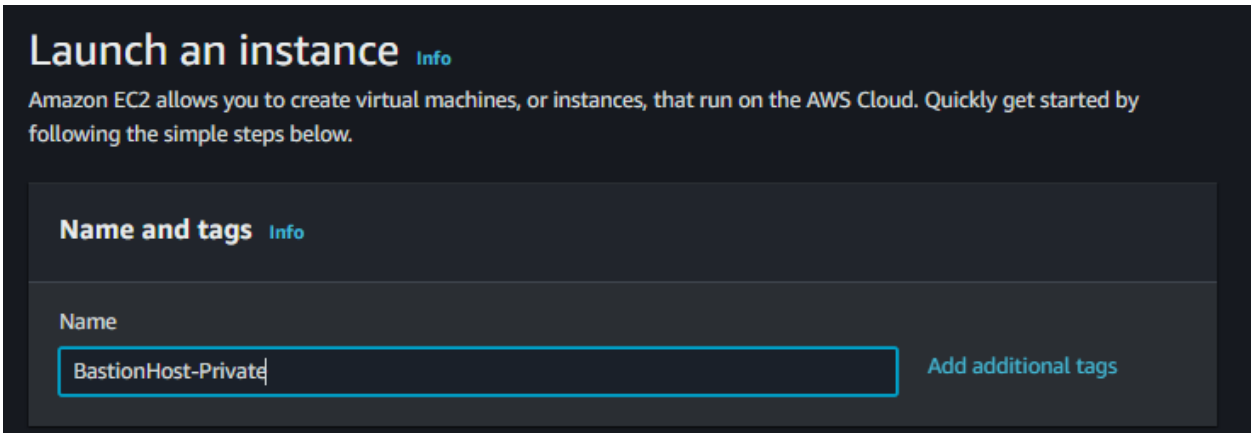
- Create a new security group for your instance Name is BastionHost-public.



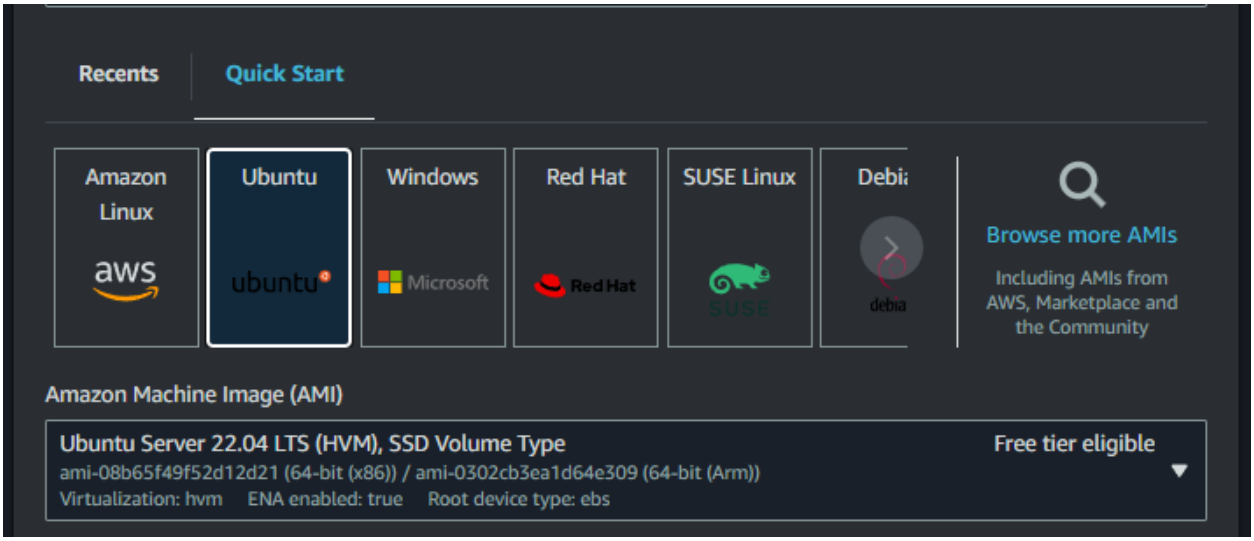
- This defines the inbound and outbound traffic rules.
- Review the configuration of your instance to ensure everything is correct.
- Click on "Launch".



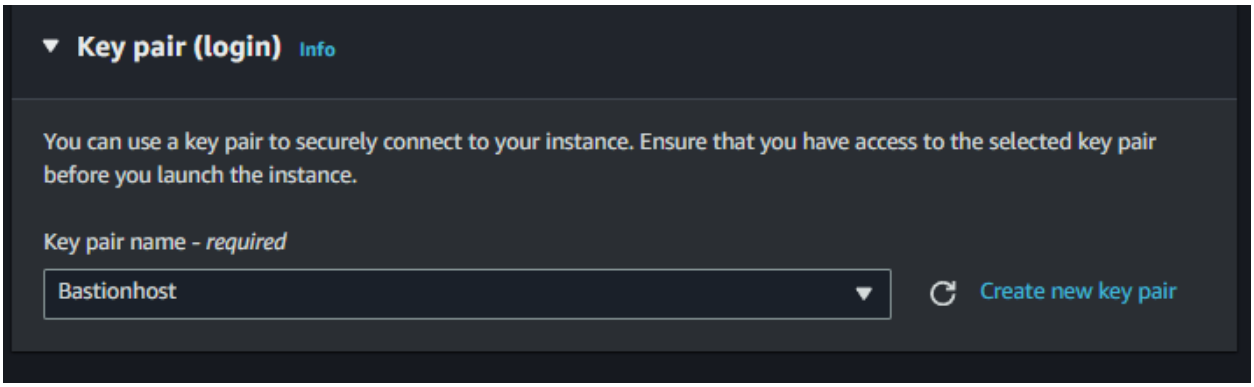
- Provide a name for the of BastionHost-Private



- Choose an Amazon Machine Image (AMI) based on your requirements. You can select from the AWS Marketplace, AWS Community AMIs, or your own custom AMIs.
- Click on "Select" once you've chosen an Ubuntu AMI

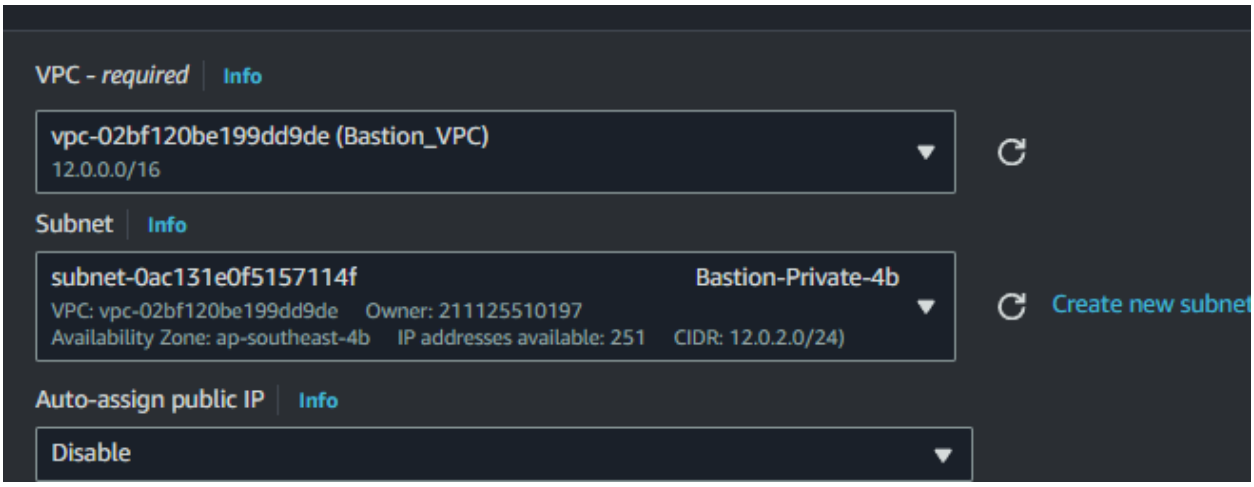


- Choose an existing key pair of “Bastionhost” to securely connect to your instance via SSH

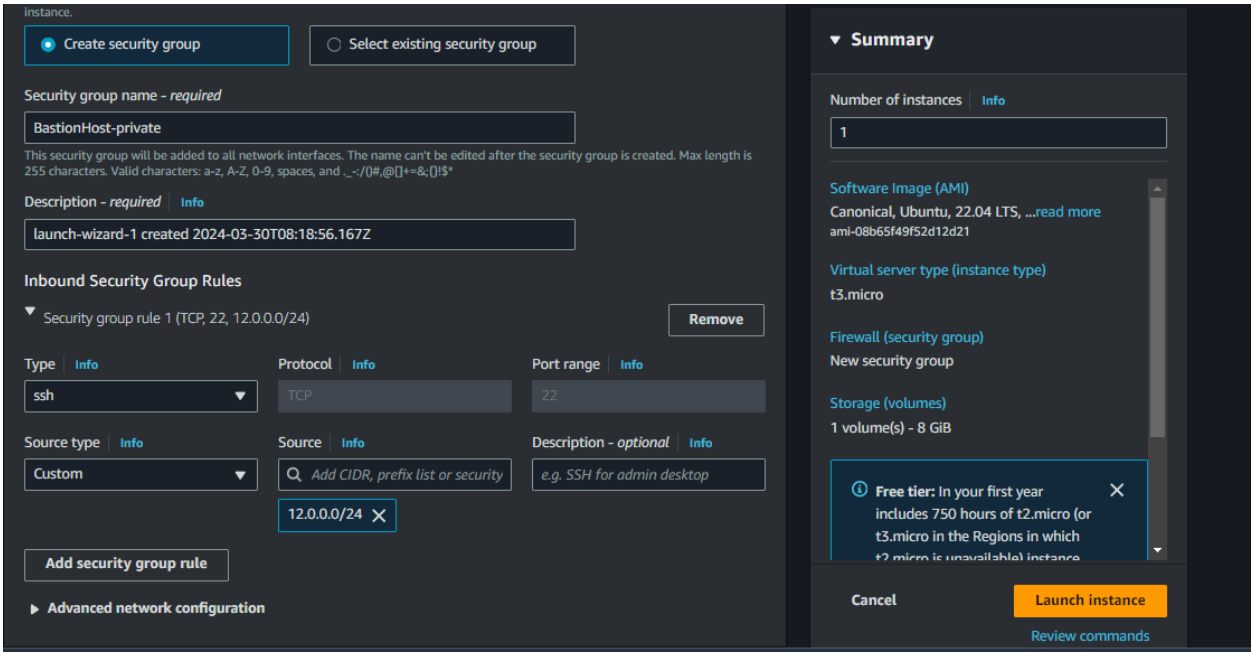


- Configure Network Settings

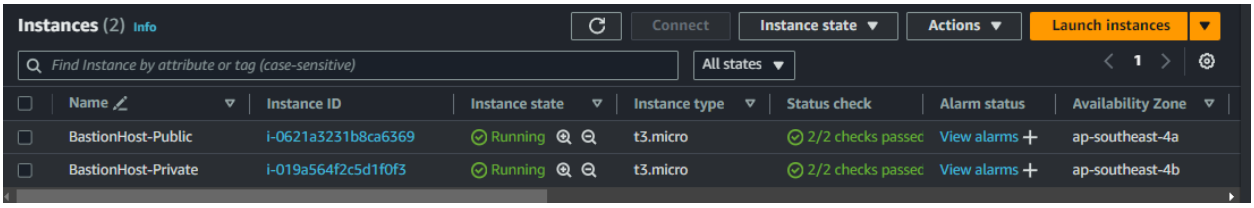
- Select the VPC you want to use from the Bastion_VPC.
- Choose the subnet within the selected the Bastion-Private-4b.
- Auto-assign Public IP: Choose whether you want to assign a public IP address to the instance. Options are "Use subnet setting (Enable or Disable)" or "Enable" or "Disable" specifically for this instance.



- Create a new security group for your instance Name is BastionHost-Private
- This defines the inbound and outbound traffic rules.
- Review the configuration of your instance to ensure everything is correct.
- Click on "Launch".



- click on Instances You will see like below. Instance state is Running.
- You can click on the Instance ID to see more details about your instance.



Step 8**:** Connect the Bastion Host

- Open a terminal window on your local machine.
- Creating a key pair (bastionhost.pem):
- The first line vi bastionhost.pem uses the vi text editor to create a new file named "bastionhost.pem". You can use any text editor you are comfortable with.
- Setting permissions on the key pair:
- The line chmod 400 "bastionhost.pem" changes the permissions of the "bastionhost.pem" file. The chmod command is used to modify file permissions in Unix-based systems. Here, "400" sets the permissions so that only the owner of the file has read, write, and execute permissions. This restricts access to the private key file for enhanced security.

```
ubuntu@ip-12-0-1-39:~$ vi Bastionhost.pem
ubuntu@ip-12-0-1-39:~$ chmod 400 "Bastionhost.pem"
```

- Connecting to the bastion host:
- The line ssh -i "bastionhost.pem" ubuntu@12.0.2.227 initiates an SSH connection to the server with the IP address 12.0.2.227. Here's a breakdown of the options used:
- -i "bastionhost.pem" : This option specifies the path to the private key file "bastionhost.pem" that you created in step 1.
- ubuntu@ : This specifies the username to use for login. In this case, it's "ubuntu".
- 12.0.2.227 : This is the IP address of the server you want to connect to.

```
ubuntu@ip-12-0-1-39:~$ ssh -i "Bastionhost.pem" ubuntu@12.0.2.227
The authenticity of host '12.0.2.227 (12.0.2.227)' can't be established.
ED25519 key fingerprint is SHA256:CVe3AhqZM2j4EPL51e0fmV617gujEBgku8vfiXACbRU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

- you should be connected to the public EC2 instance from your local machine via the bastion host and the Private EC2 instance.

```
ubuntu@ip-12-0-2-227:~$ exit
logout
Connection to 12.0.2.227 closed.
ubuntu@ip-12-0-1-39:~$
```