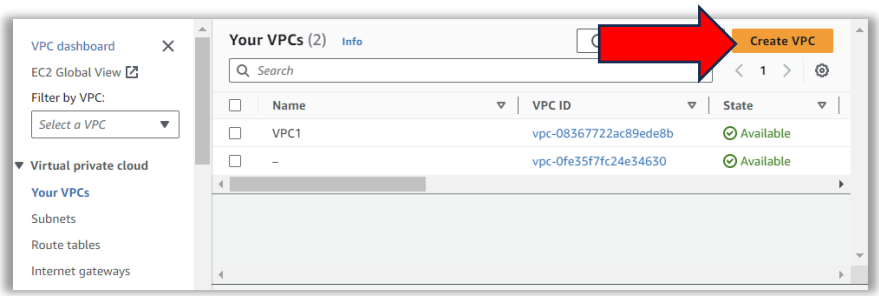
AWS Elastic Compute Cloud Lab (EC2)

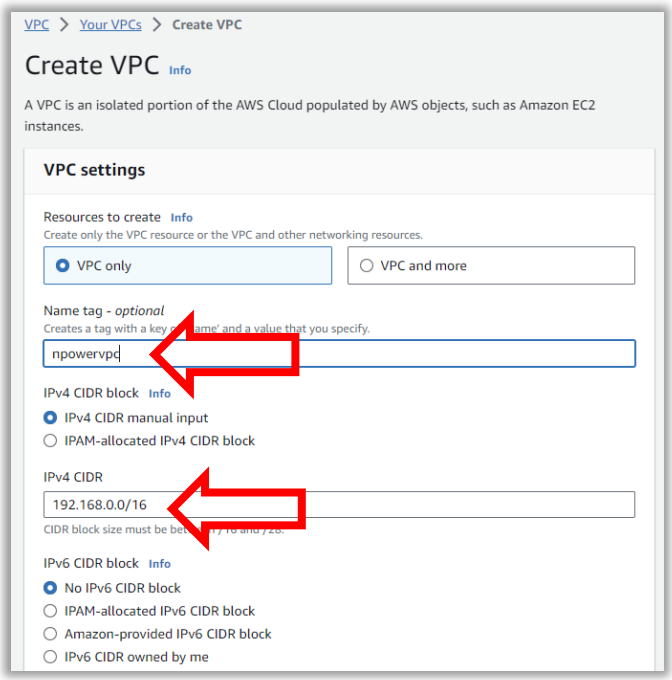
In this lab, you are going to learn about the EC2 Instance which is the virtual server in the AWS cloud. Amazon Elastic Compute Cloud (Amazon EC2) provides on-demand, scalable computing capacity in the Amazon Web Services (AWS) Cloud. Using Amazon EC2 reduces hardware costs so you can develop and deploy applications faster. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage.

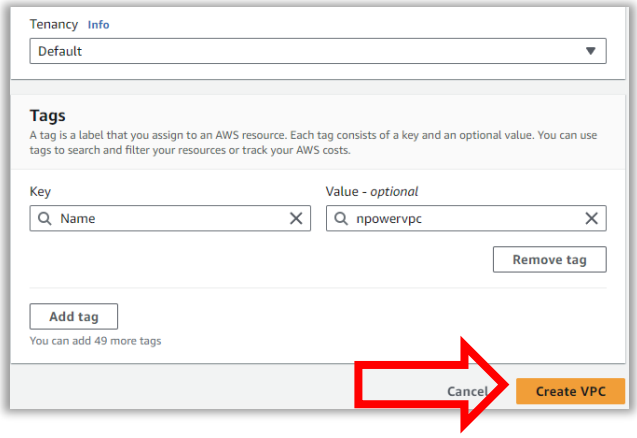
Virtual Private Cloud (VPC) is a logically isolated virtual network. You can launch AWS resources in a logically isolated virtual network that you've defined. This virtual network closely resembles a traditional network that you'd operate in your own data center, with the benefits of using the scalable infrastructure of AWS.

Step 1. Create a VPC.



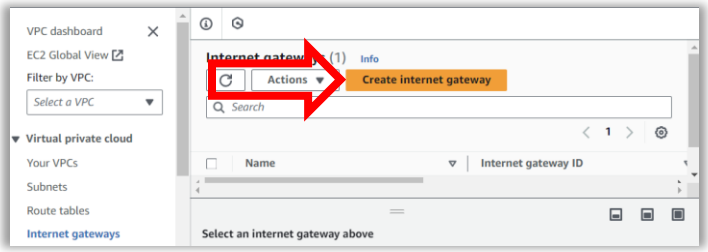
Step 1a. Name the VPC, and give an IPv4 CIDR block of **192.168.0.0/16**

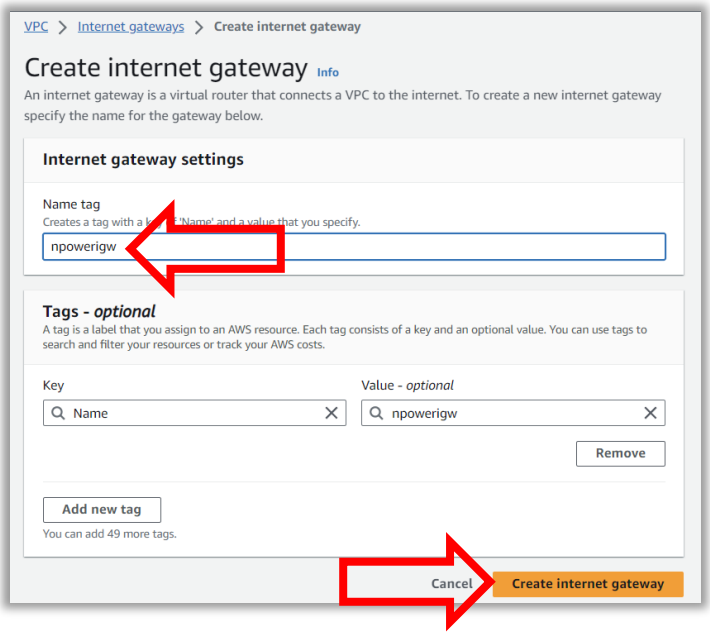




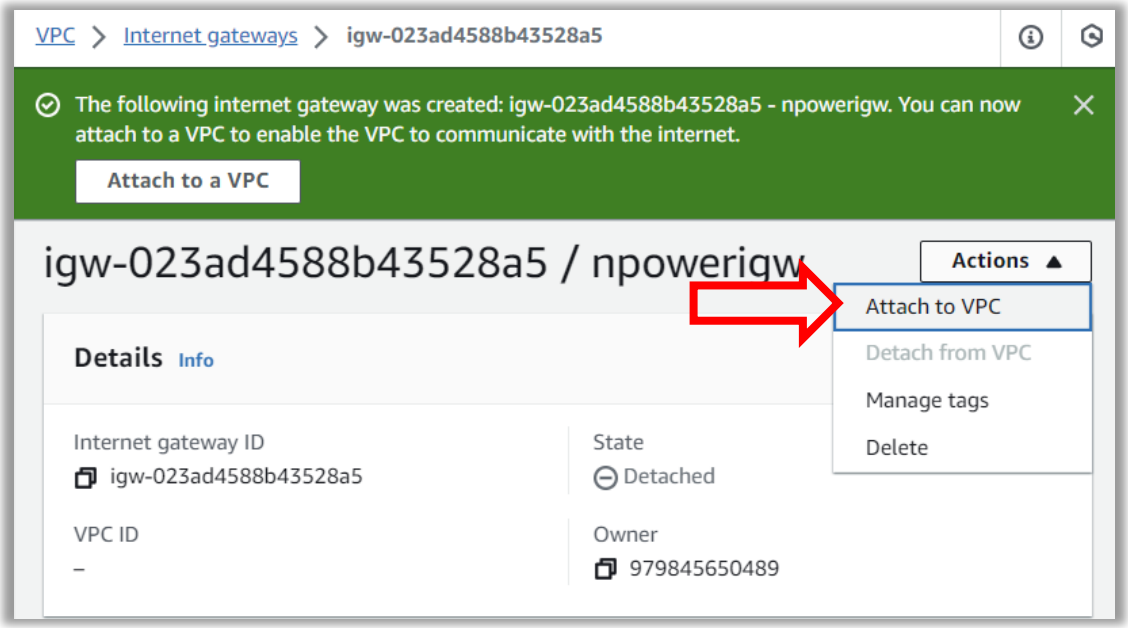
Step 2. An Internet Gateway is a VPC component that allows communication between your VPC and the internet. It supports IPv4 and IPv6 traffic. An internet gateway enables resources in your public subnets (such as EC2 instances) to connect to the internet if the resource has a public IPv4 address or an IPv6 address. Similarly, resources on the internet can initiate a connection to resources in your subnet using the public IPv4 address or IPv6 address.

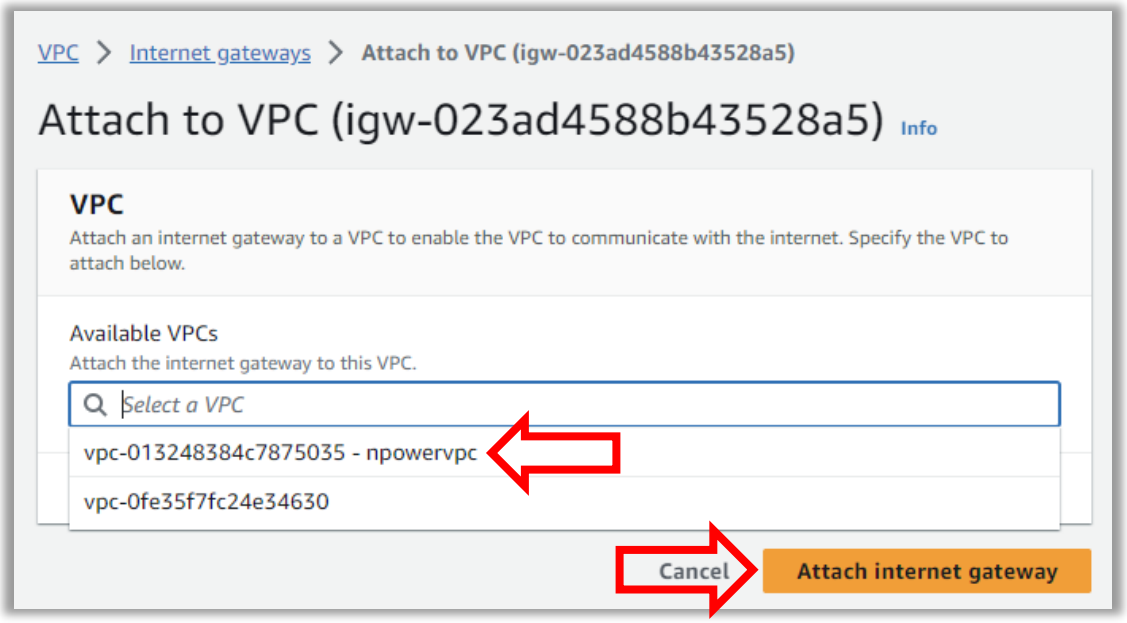
- Create an Internet Gateway
- Name the Internet gateway (npowerigw) and click create.





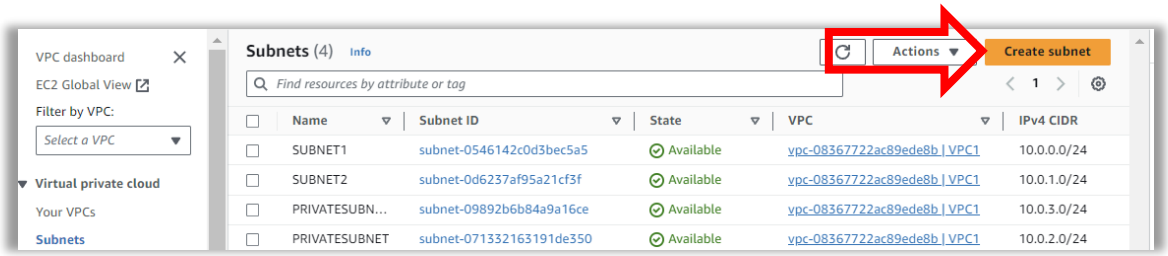
Step 3. Attach the Internet Gateway to the VPC that was just created.



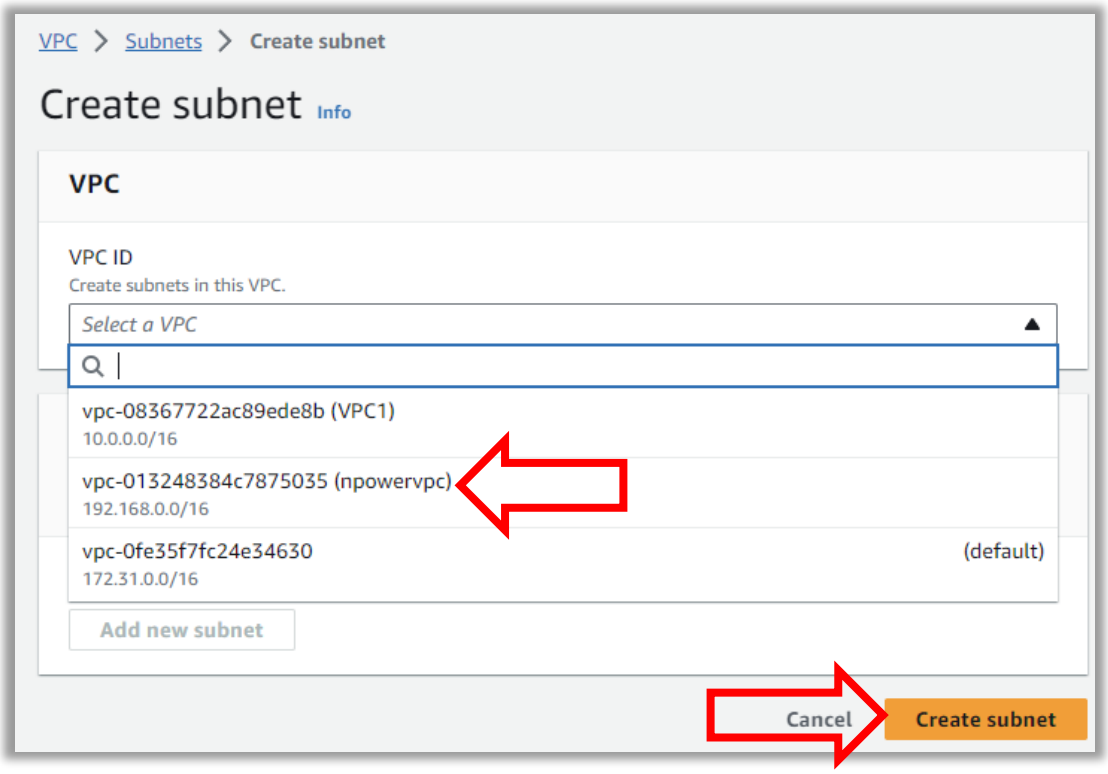


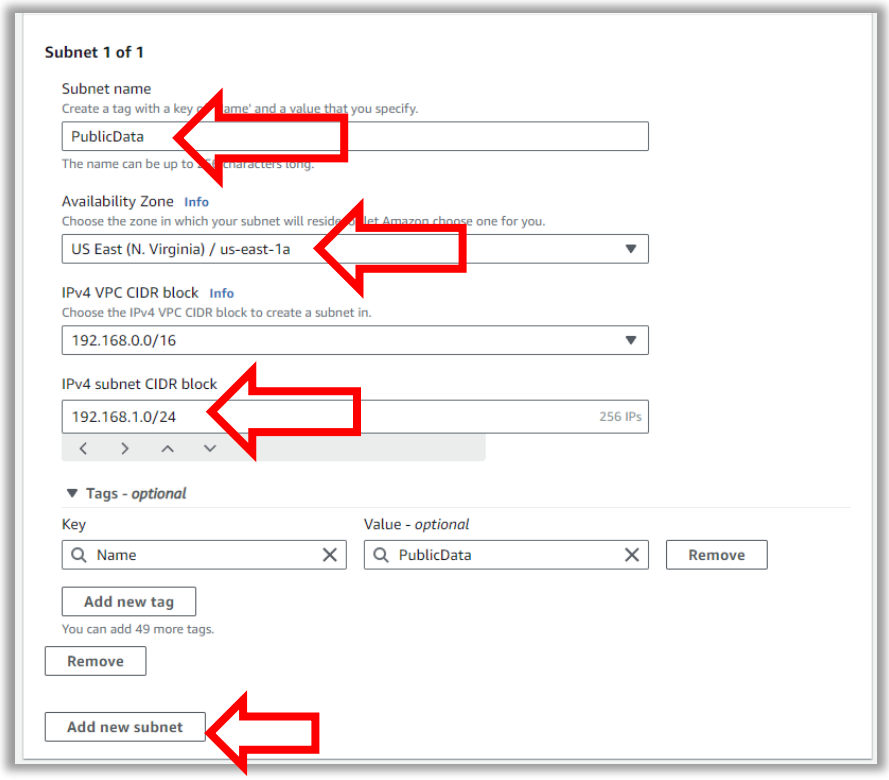
Step 4. A *subnet* is a range of IP addresses in your VPC. You can create AWS resources, such as EC2 instances, in specific subnets. Each subnet must reside entirely within one Availability Zone and cannot span zones. By launching AWS resources in separate Availability Zones, you can protect your applications from the failure of a single Availability Zone.

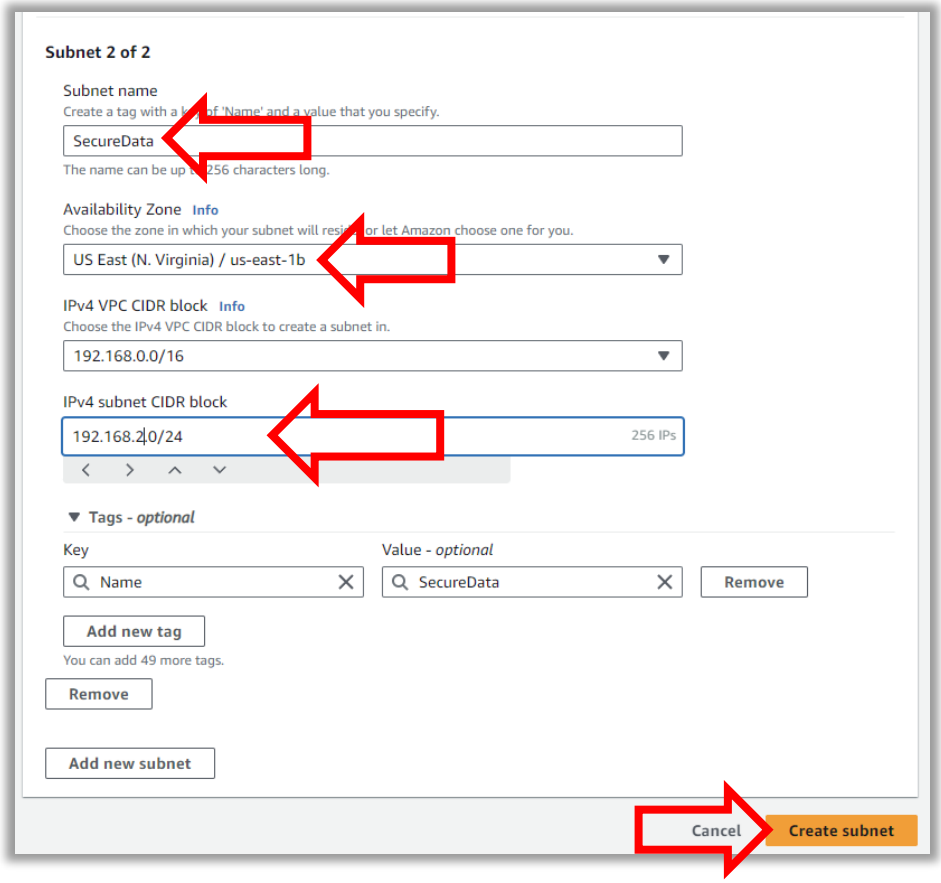
- Create two subnets Named (PublicData, SecureData), and attach to your VPC. Give each one a different Availability Zone (1a, 1b) with CIDR Blocks (192.168.1.0/24, & 192.168.2.0/24) respectively.



- Choose the VPC created in step 1.

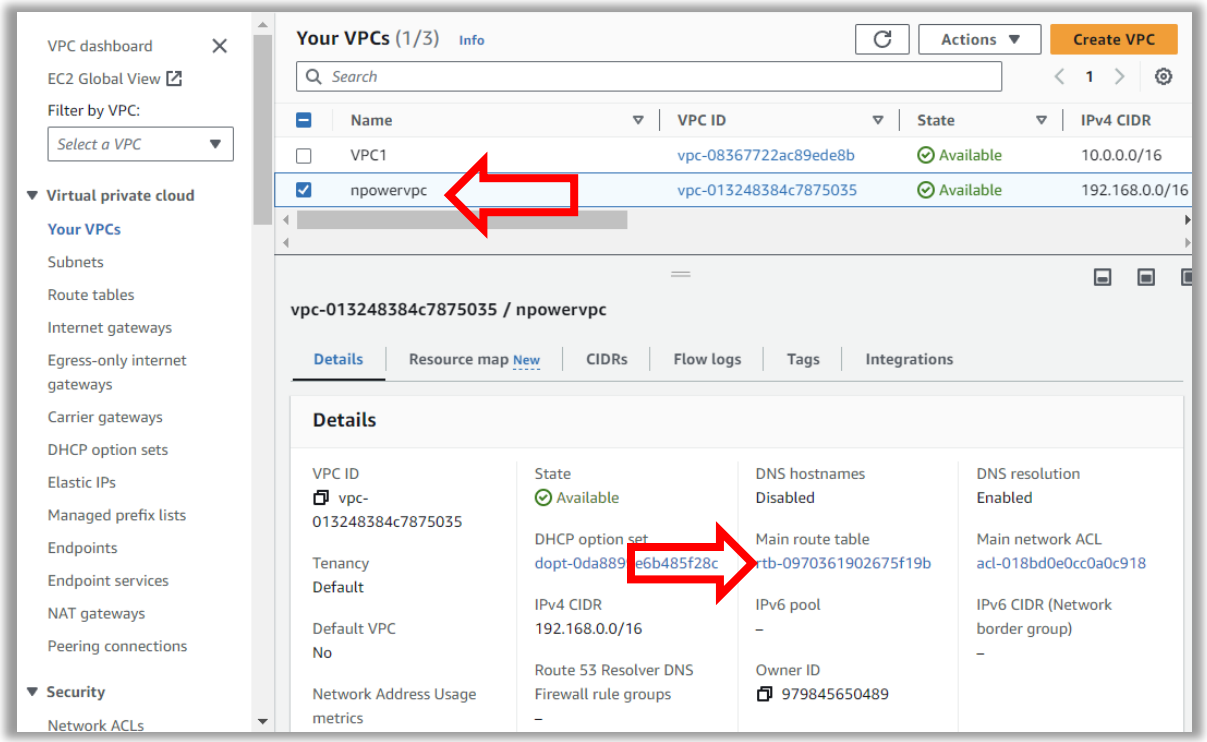



Subnet 1 of 1
Subnet name
Create a tag with a key of 'Name' and a value that you specify.
PublicData
The name can be up to 256 characters long.
Availability Zone Info
Choose the zone in which your subnet will reside, or let Amazon choose one for you.
US East (N. Virginia) / us-east-1a
IPv4 VPC CIDR block Info
Choose the IPv4 VPC CIDR block to create a subnet in.
192.168.0.0/16
IPv4 subnet CIDR block
192.168.1.0/24
256 IPs
Tags - optional
Key
Value - optional
Name
PublicData
Remove
Add new tag
You can add 49 more tags.
Remove
Add new subnet


Subnet 2 of 2
Subnet name
Create a tag with a key of 'Name' and a value that you specify.
SecureData
The name can be up to 256 characters long.
Availability Zone Info
Choose the zone in which your subnet will reside, or let Amazon choose one for you.
US East (N. Virginia) / us-east-1b
IPv4 VPC CIDR block Info
Choose the IPv4 VPC CIDR block to create a subnet in.
192.168.0.0/16
IPv4 subnet CIDR block
192.168.2.0/24
256 IPs
Tags - optional
Key
Value - optional
Name
SecureData
Remove
Add new tag
You can add 49 more tags.
Remove
Add new subnet
Cancel
Create subnet

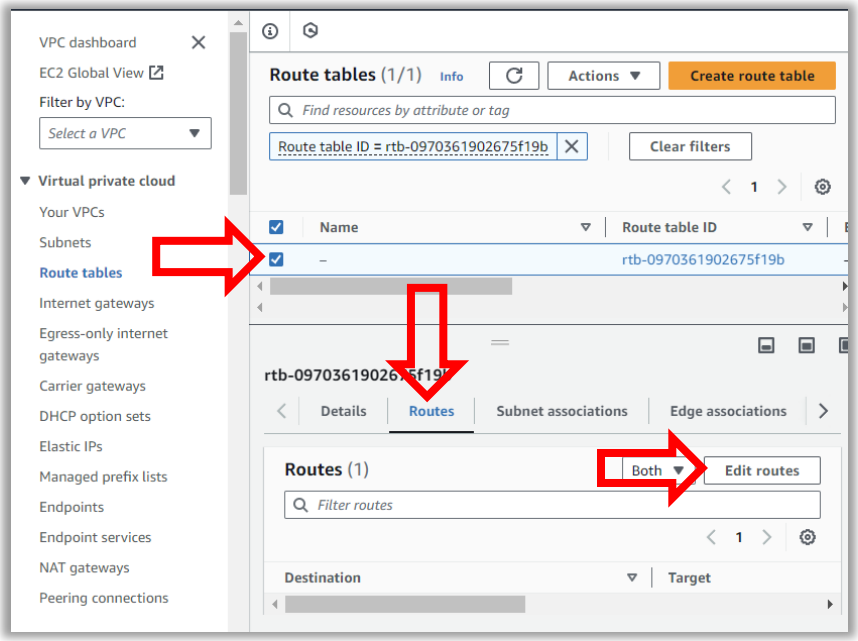
Step 5. A *route table* contains a set of rules, called *routes*, that determine where network traffic from your subnet or gateway is directed. The **Main route table**—The route table that automatically comes with your VPC.

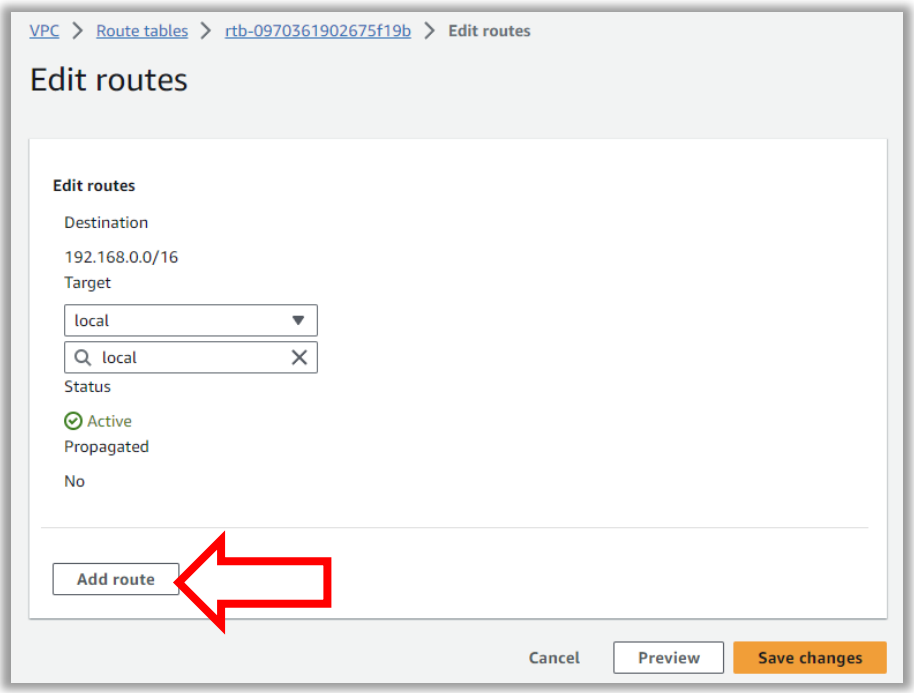
- Select your VPC and click on the Main route table

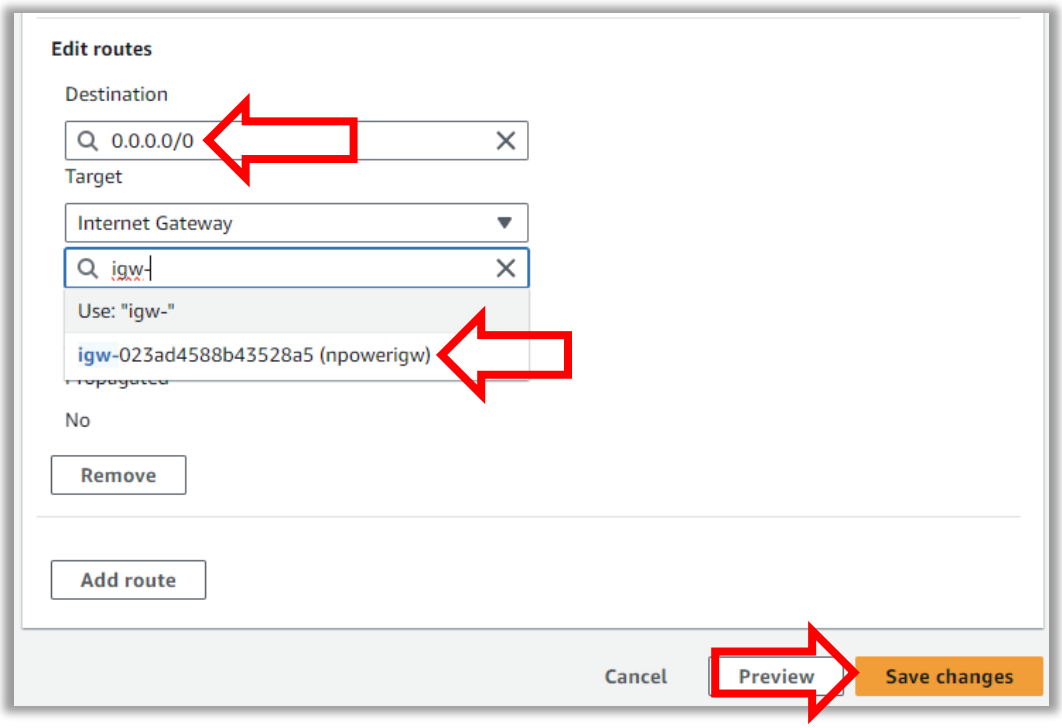


Step 6. Each route in a table specifies a destination and a target. For example, to enable your subnet to access the internet through an internet gateway, add the following route to your subnet route table. The destination for the route is 0.0.0.0/0, which represents all IPv4 addresses. The target is the internet gateway that's attached to your VPC.

- Edit routes, add universal IP address 0.0.0.0/0, with target set with the Internet Gateway we created from Step 2.

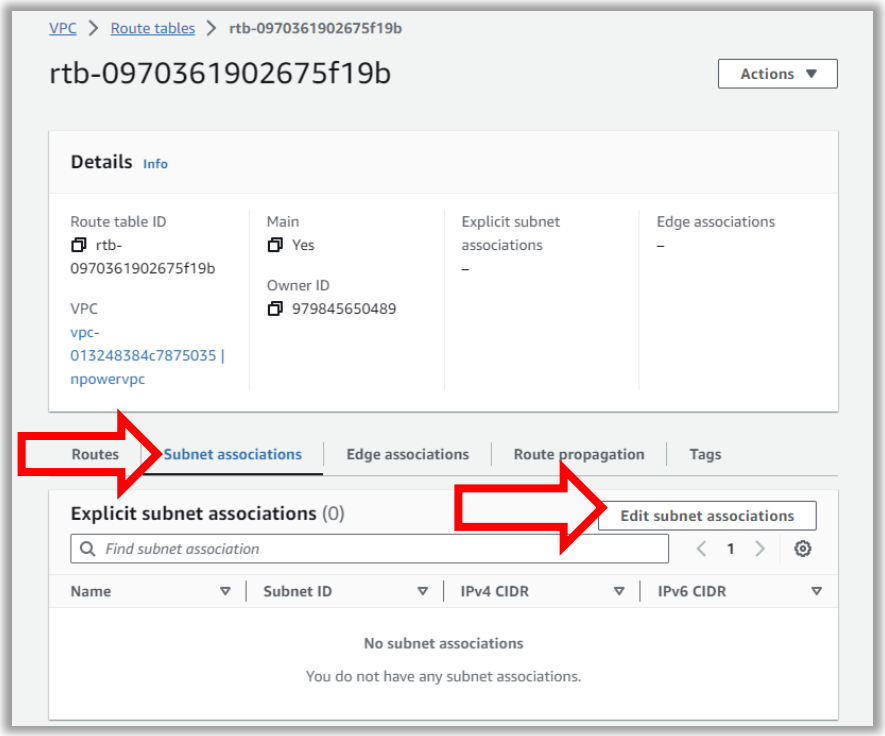




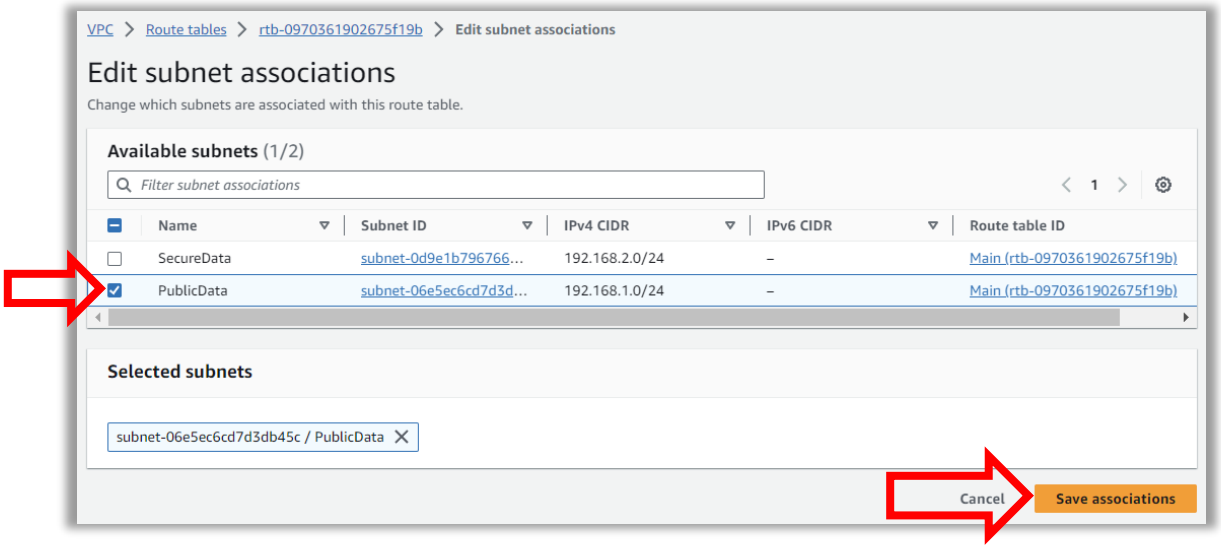


Step 7. Each subnet in your VPC must be associated with a route table. A subnet can be explicitly associated with custom route table, or implicitly or explicitly associated with the main route table.

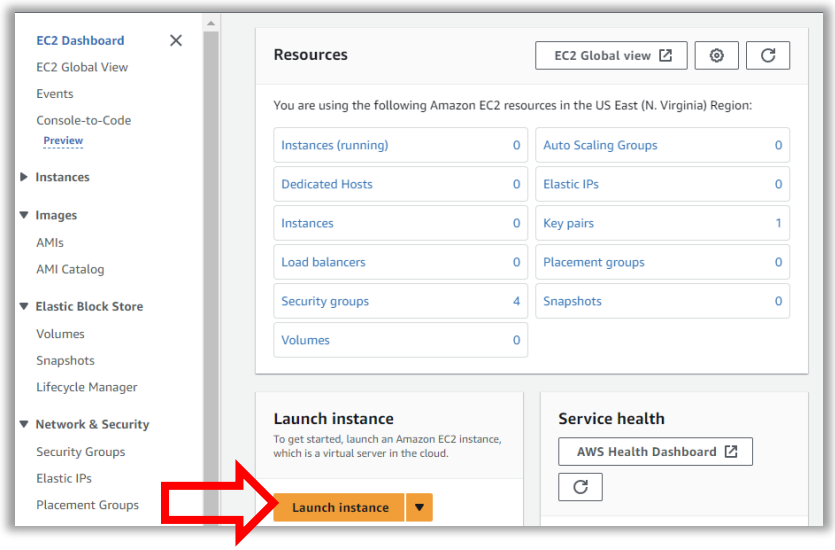
- Edit subnet associations of the route table.



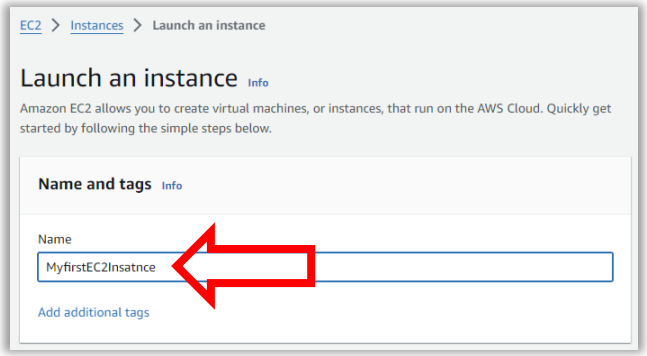
Step 7a. Associate Subnet: PublicData with the Rotatable and save.



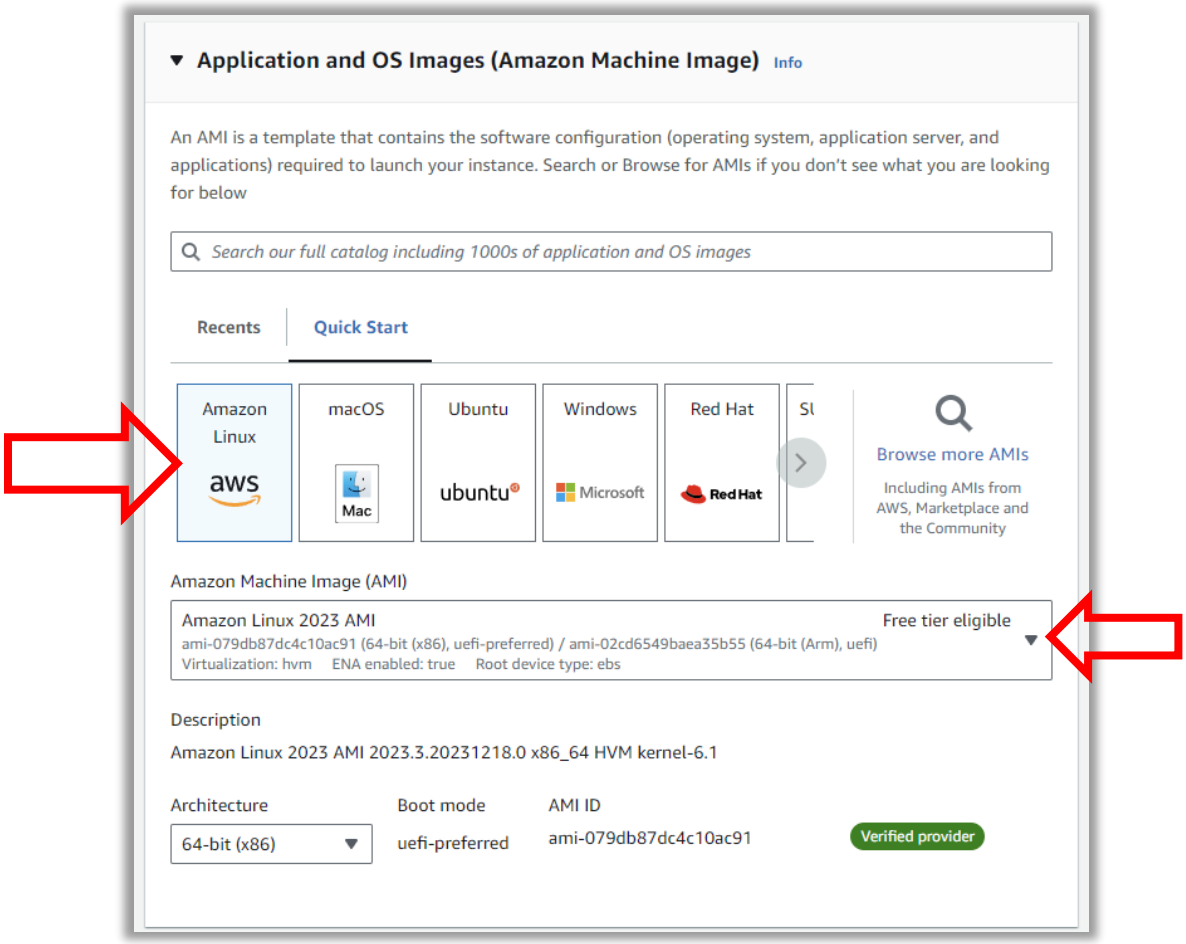
Step 8. Launch an EC2 Instance



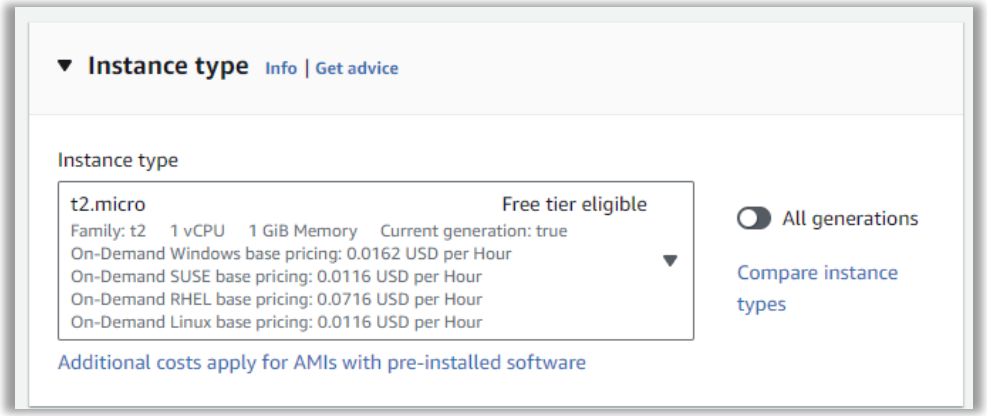
Step 8a. Name - MyFirstEC2Instance



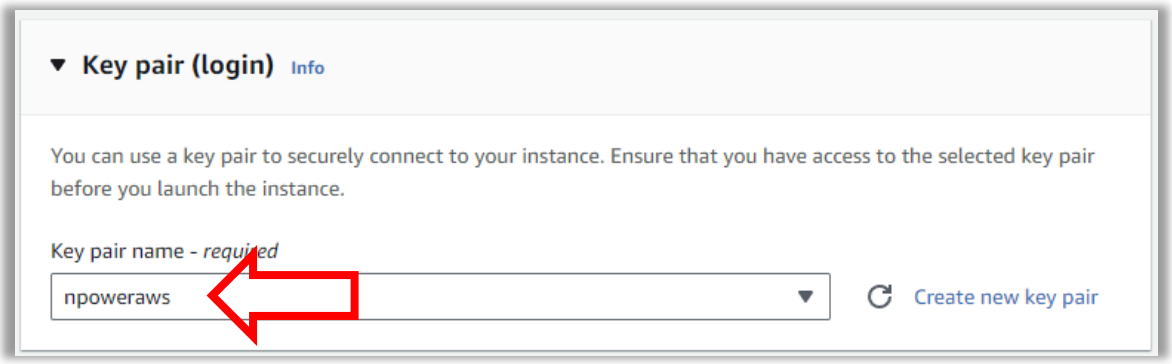
- **Choose the Amazon Linux 2 AMI (HVM), SSD Volume Type**



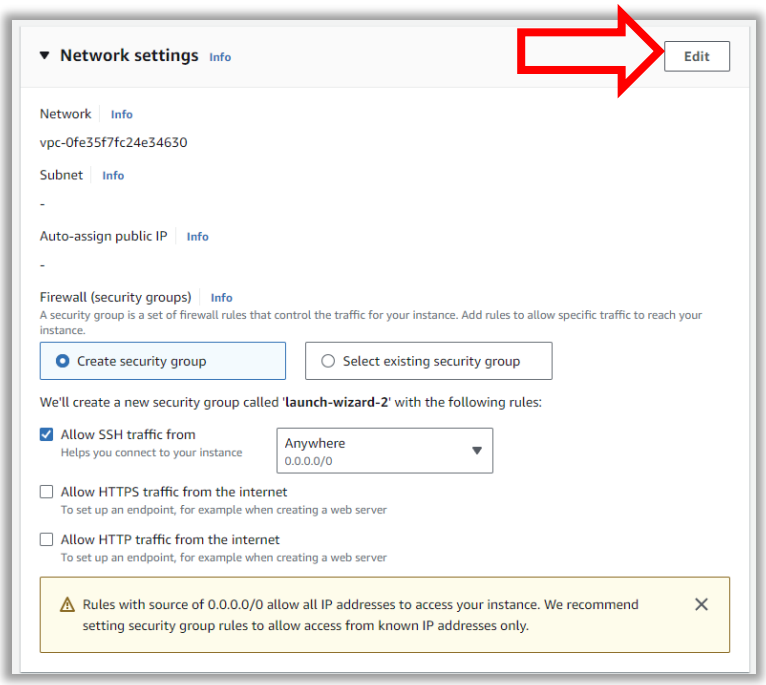
Step 8b. Leave instance type as default.



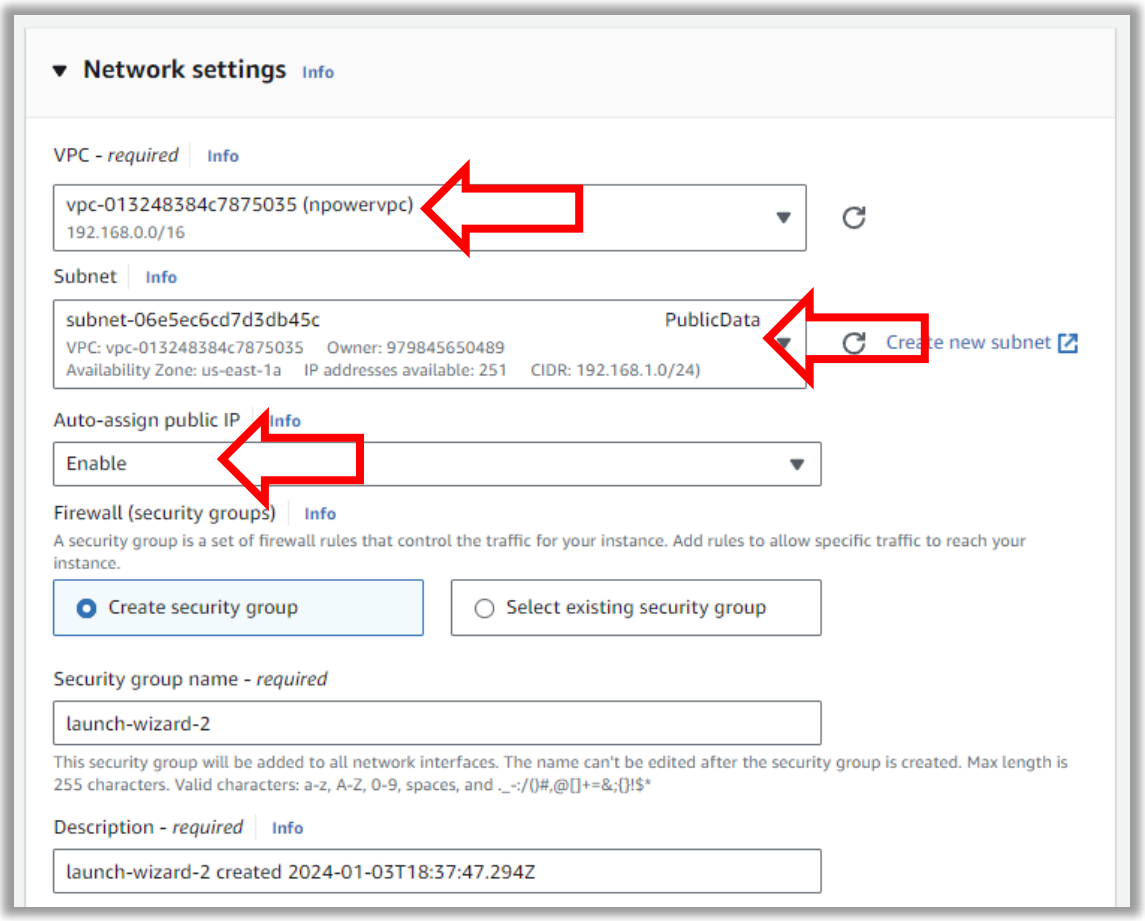
Step 8c. Choose the existing Keypair created from Lab 1.



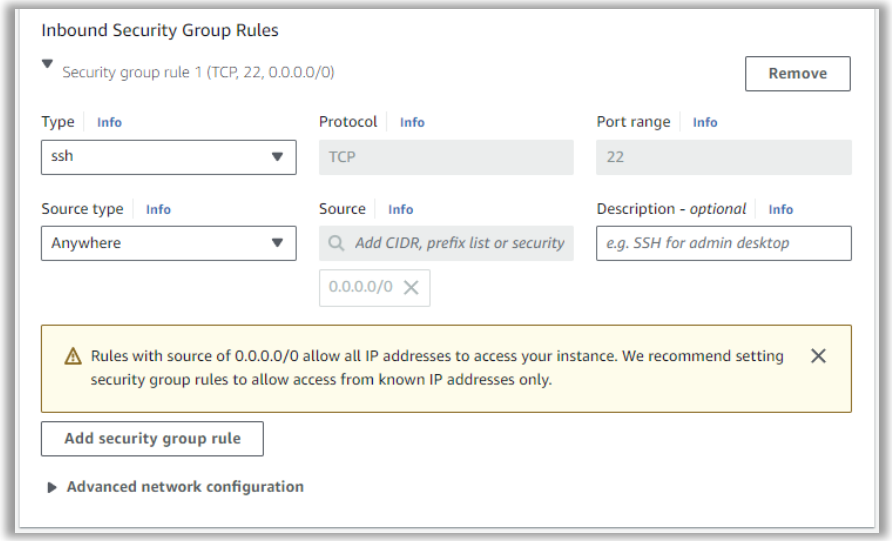
Step 8d. Configure the Network Settings



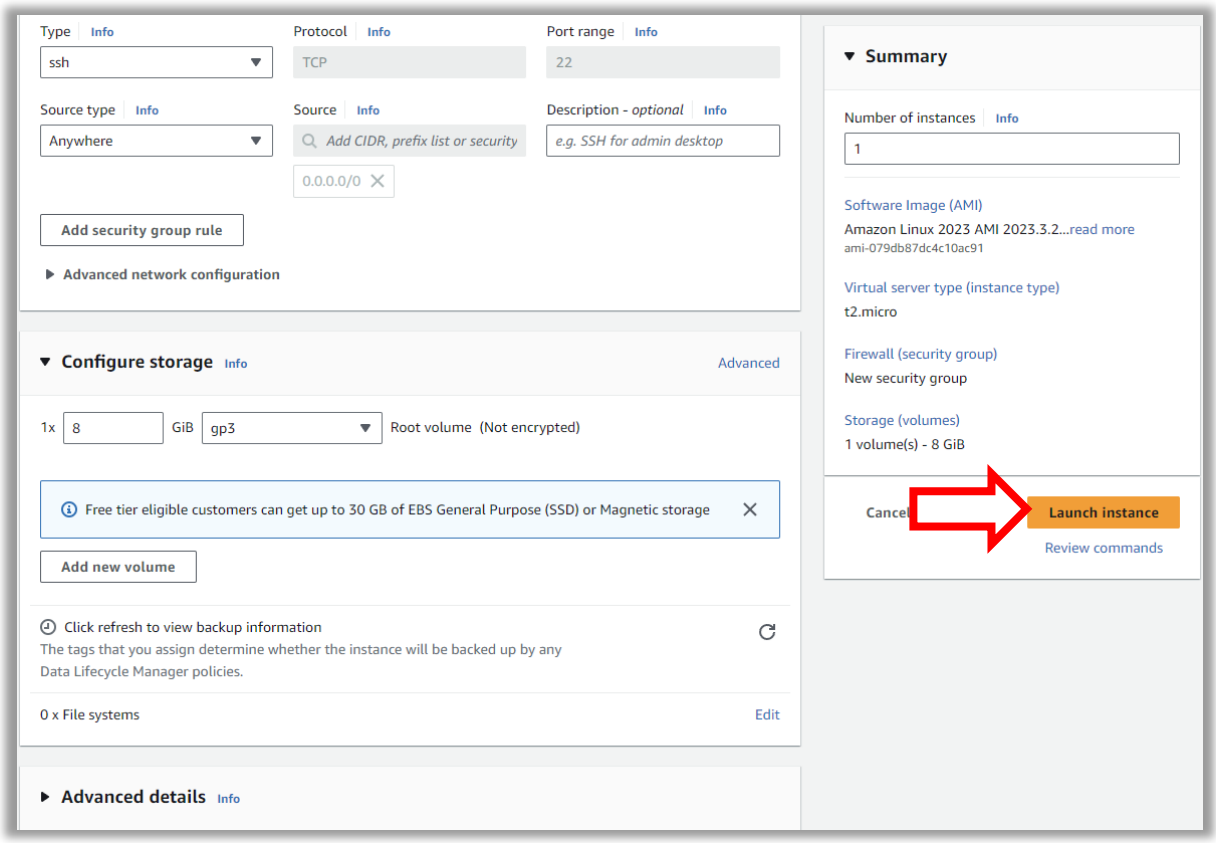
- VPC: **npowervpc** | Subnet: **PublicData** | Auto-assign public-IP: **Enabled**



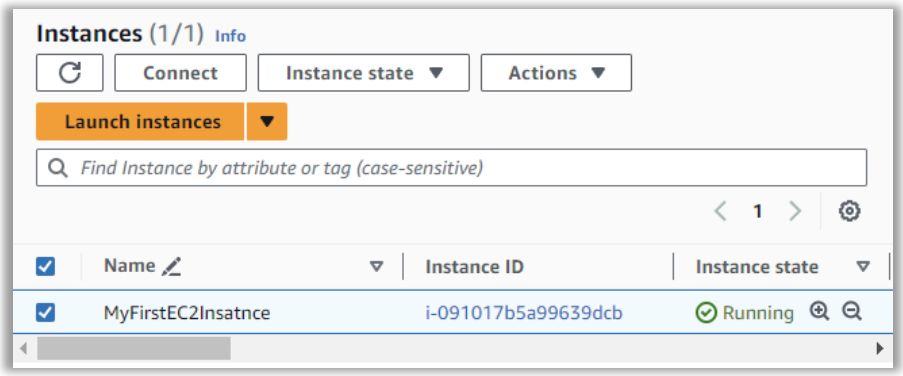
- Inbound Security Group rules leave as default.

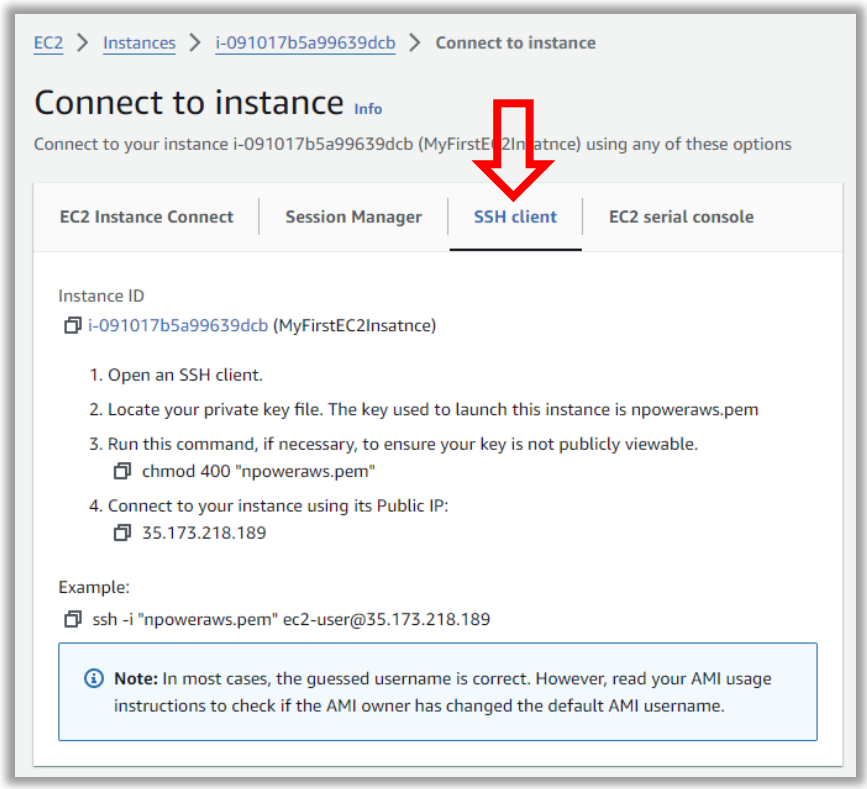


- Configure storage leave it as default. Click **Launch instance.**

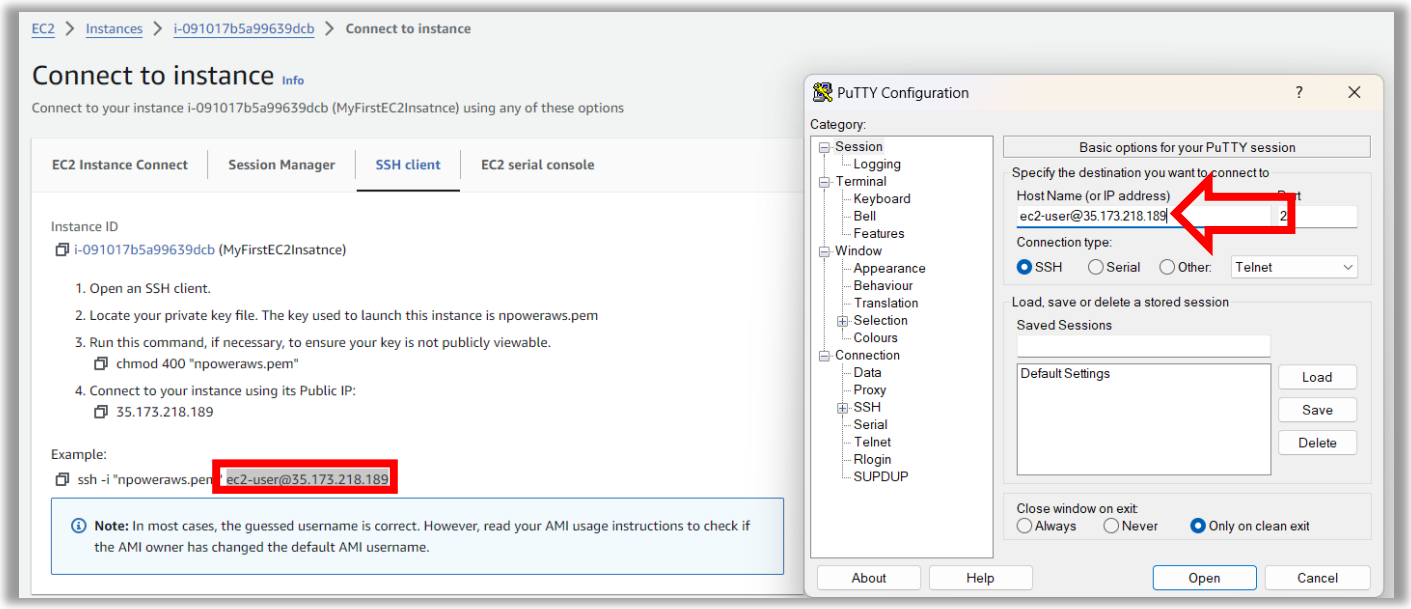


Step 9. Connect onto your EC2 Instance using SSH Client



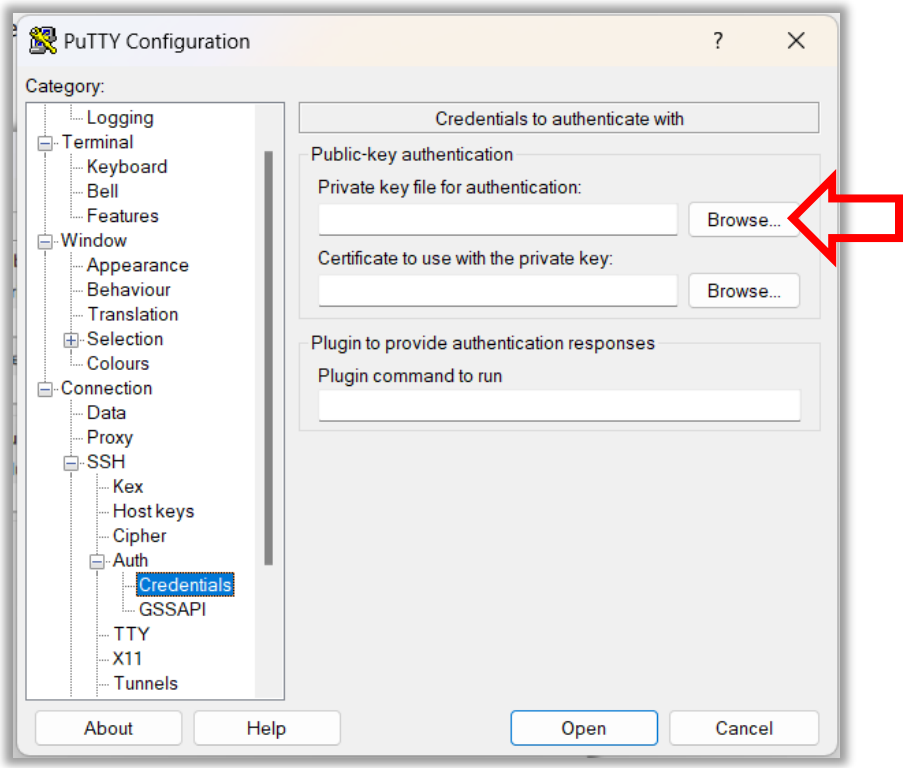


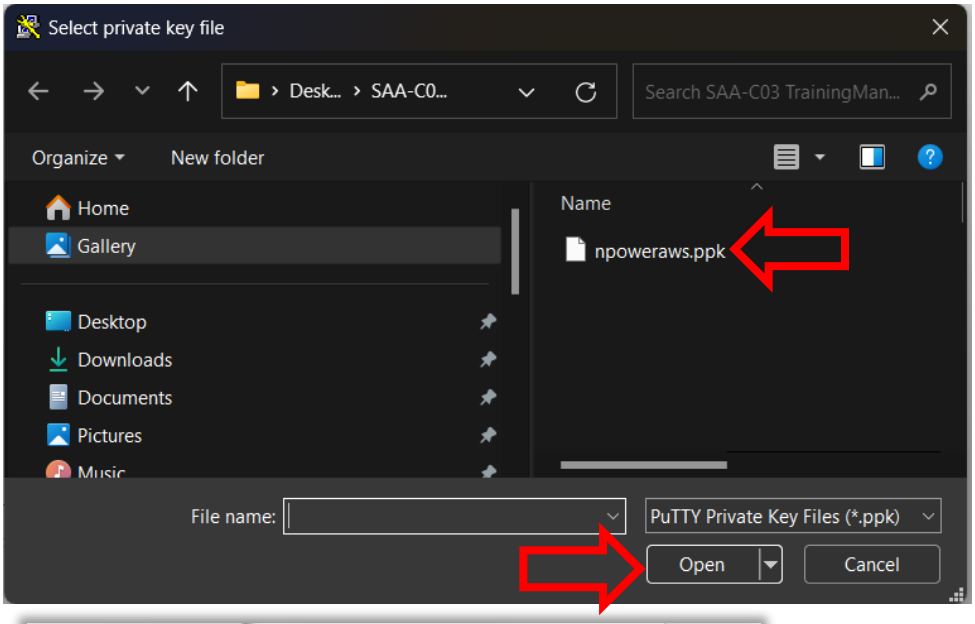
Step 9a. Open Putty, Copy the Host Name and paste it into Putty, under Host Name (or IP address)

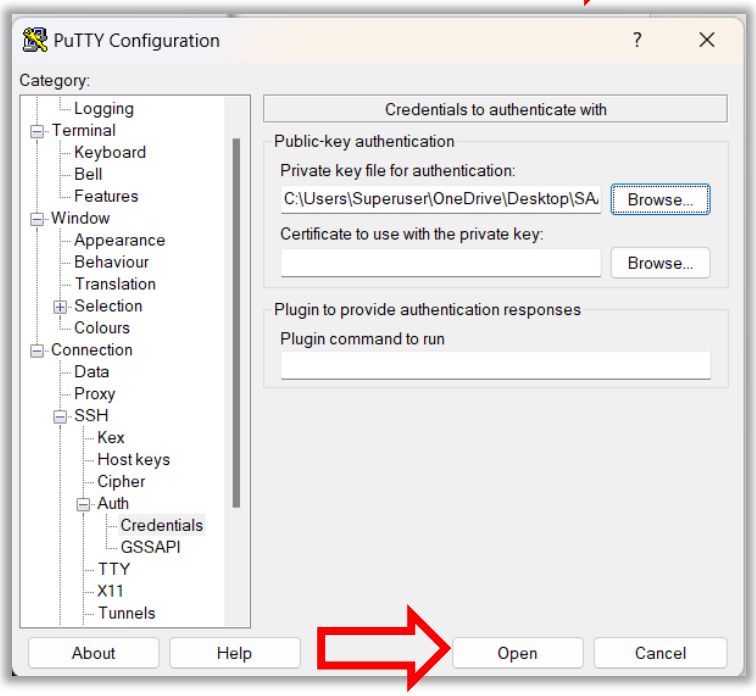


Step 9b. Authenticate your SSH Connection and Choose the Private Key that was selected with the EC2 instance.

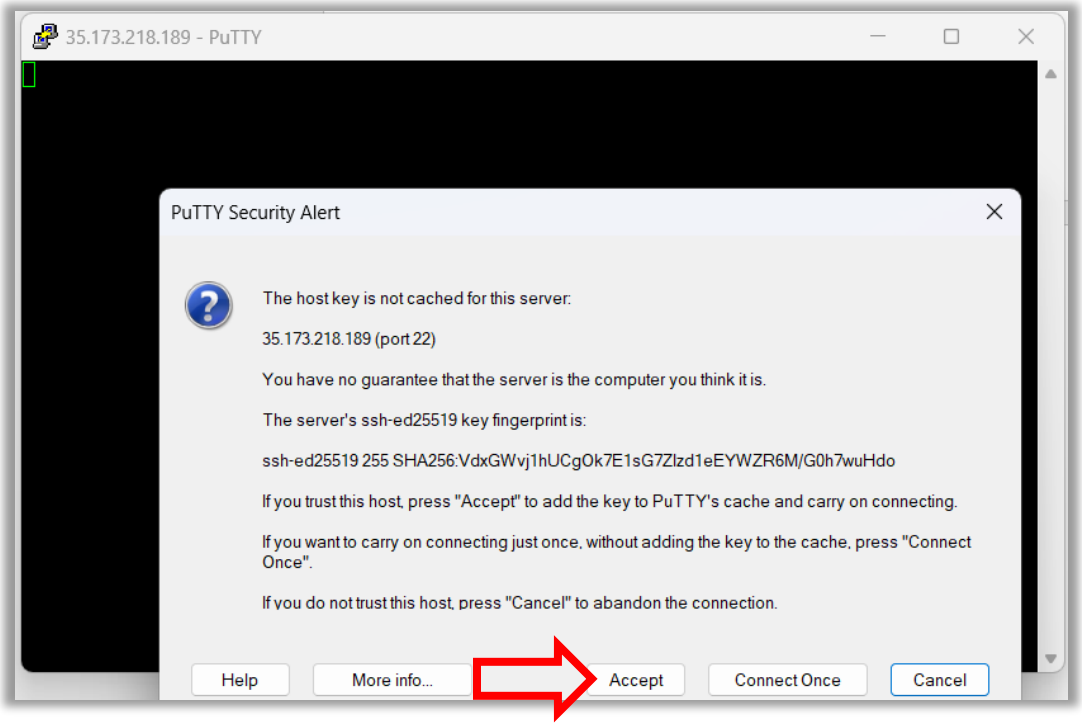
- Expand the SSH tab by clicking on +
- Select the AUTH tab and click the +
- Choose **credentials**, click on Browse.




Select private key file
Desk... › SAA-C0...
Search SAA-C03 TrainingMan...
Organize
New folder
Home
Gallery
Desktop
Downloads
Documents
Pictures
Music
Name
npoweraws.ppk
File name:
PuTTY Private Key Files (*.ppk)
Open
Cancel

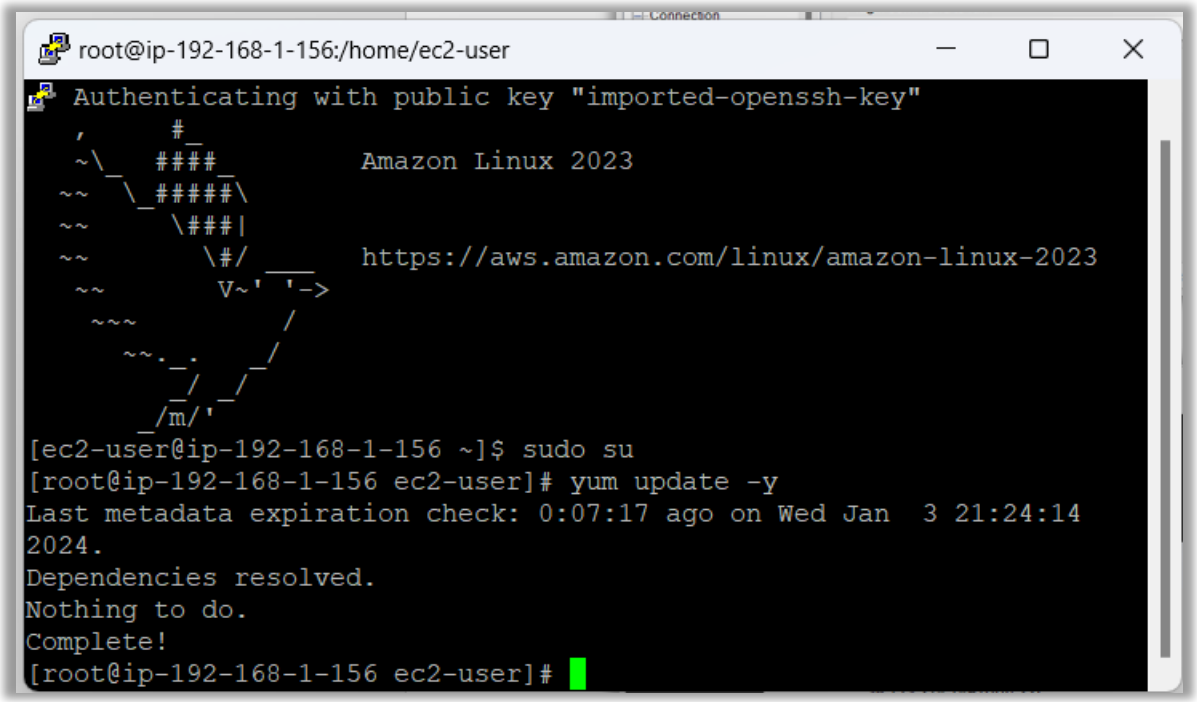

PuTTY Configuration
Category:
Logging
Terminal
Keyboard
Bell
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
SSH
Kex
Host keys
Cipher
Auth
Credentials
GSSAPI
TTY
X11
Tunnels
Credentials to authenticate with
Public-key authentication
Private key file for authentication:
C:\Users\Superuser\OneDrive\Desktop\SA
Browse...
Certificate to use with the private key:
Browse...
Plugin to provide authentication responses
Plugin command to run
About
Help
Open
Cancel

Step 9c. Click Accept



Step 10. Type following Linux commands.

- **sudo su** to become a root user **| yum update -y** to update the instance.



CONGRATULATIONS!! You have completed the EC2 Lab!