**PING A PRIVATE SERVER WITH BASTION SEVER WITH MULTIFACTOR AUTHENTICATION(MFA) USING ANSIBLE**

**Prerequisites**:

* Amazon web services Account
* Virtual private cloud
* Ec2 Instance
* Google authenticator
* External device.

# **Steps to create the MFA on Bastion Server**

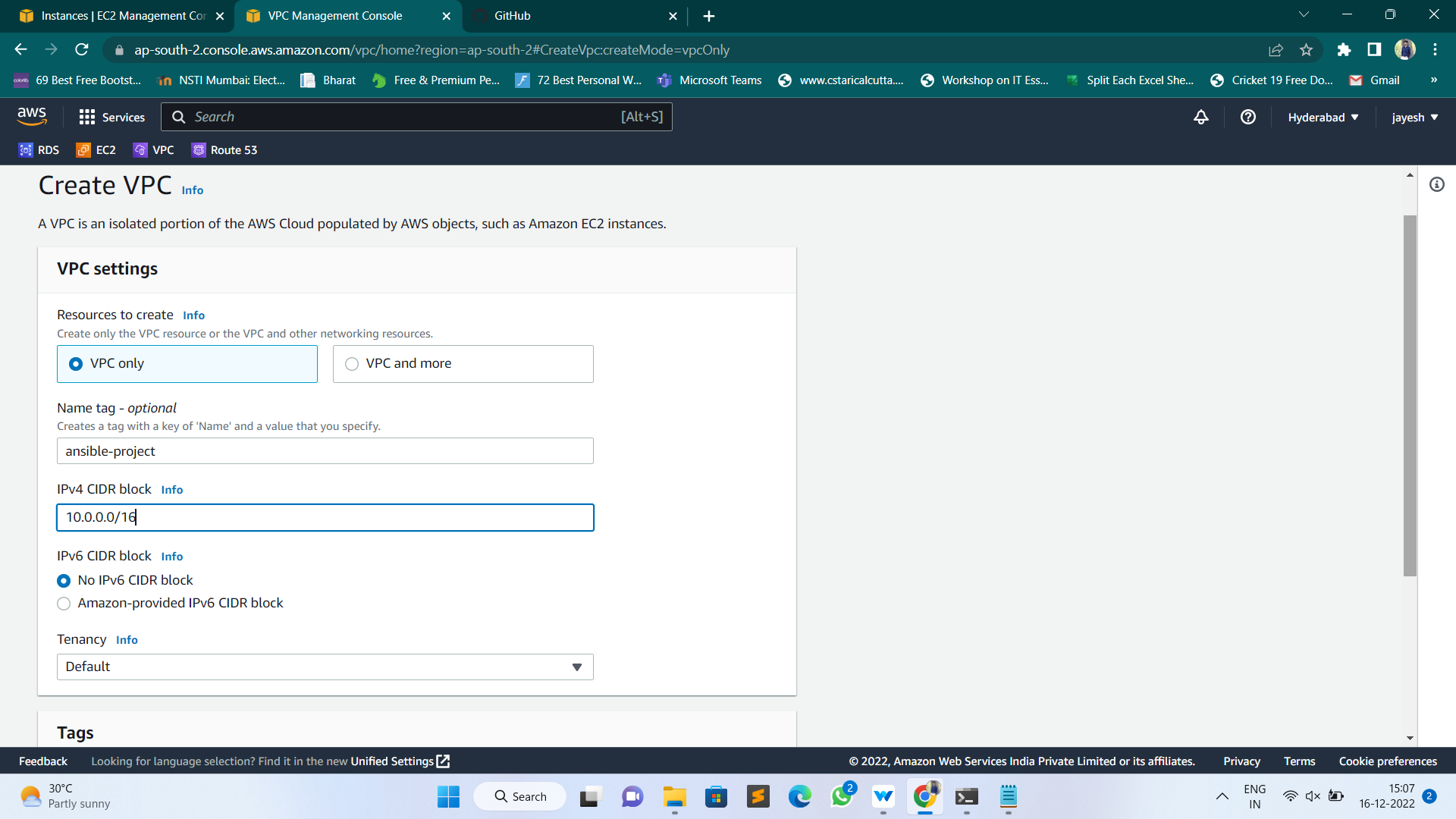
1. create a customized VPC
   * create a VPC with CIDR range 10.0.0.0/16
   * Create IGW and attach with VPC
   * create a public subnet with CIDR range 10.0.0.0/24
   * create private sub 1 with 10.0.128.0/24
   * create a public rt and associate with pulic subnet and create a root in 0.0.0.0/0 with internet gateway
   * create a NAT Gateway
   * create a private rt 1 and associate with private sub 1 and create a root with 0.0.0.0/0 with NAT

1. create, configure and Launch 2 liux2 ec2 instances
   * one is Bastion server
   * Another is private server
2. Install ansible in Bastion server and ping private server form Bastion server.

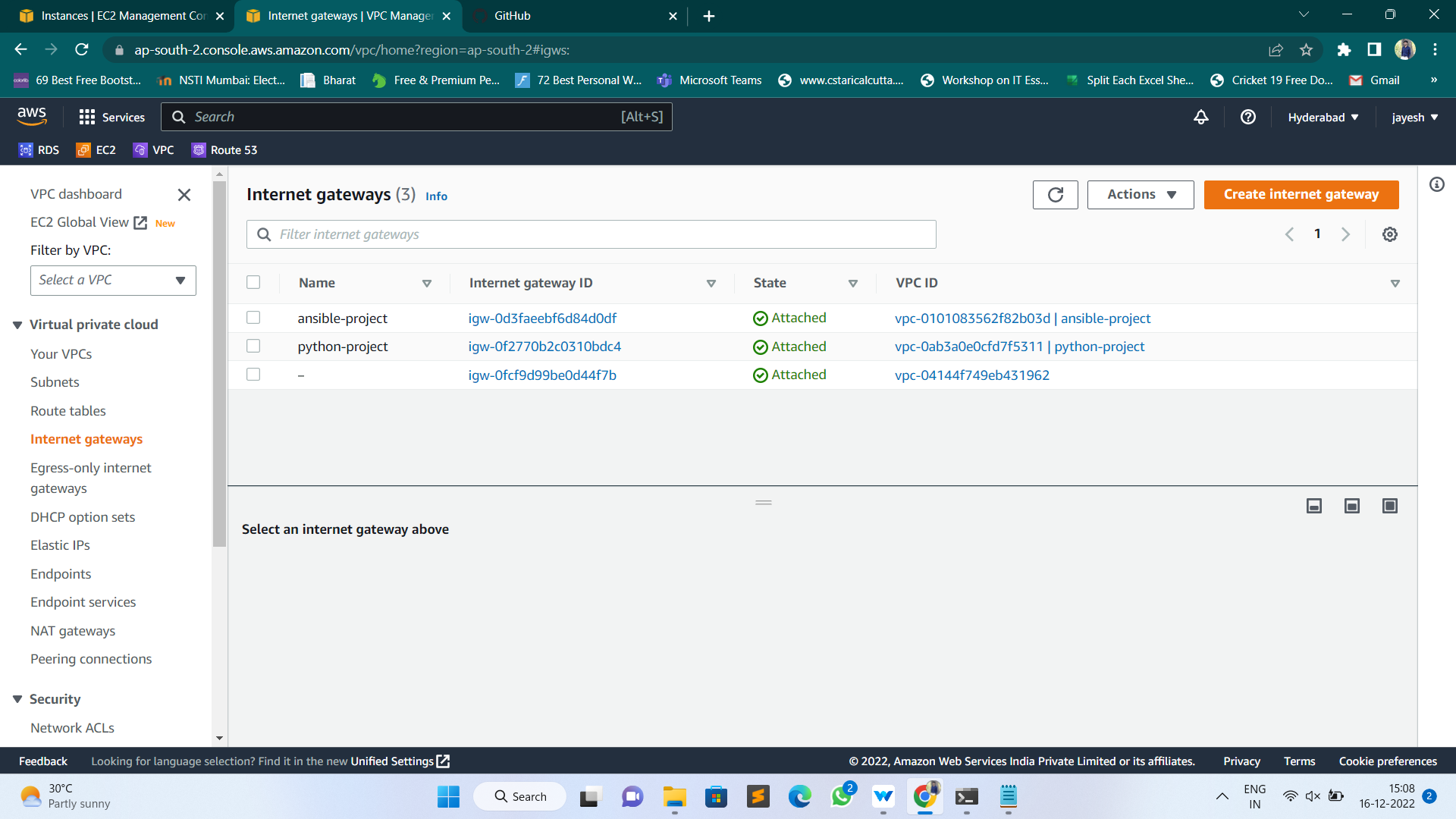
# **Module 1: create a customized VPC**

* Login to Amazon web services account
* Go to virtual private cloud (VPC)
* Click on create VPC

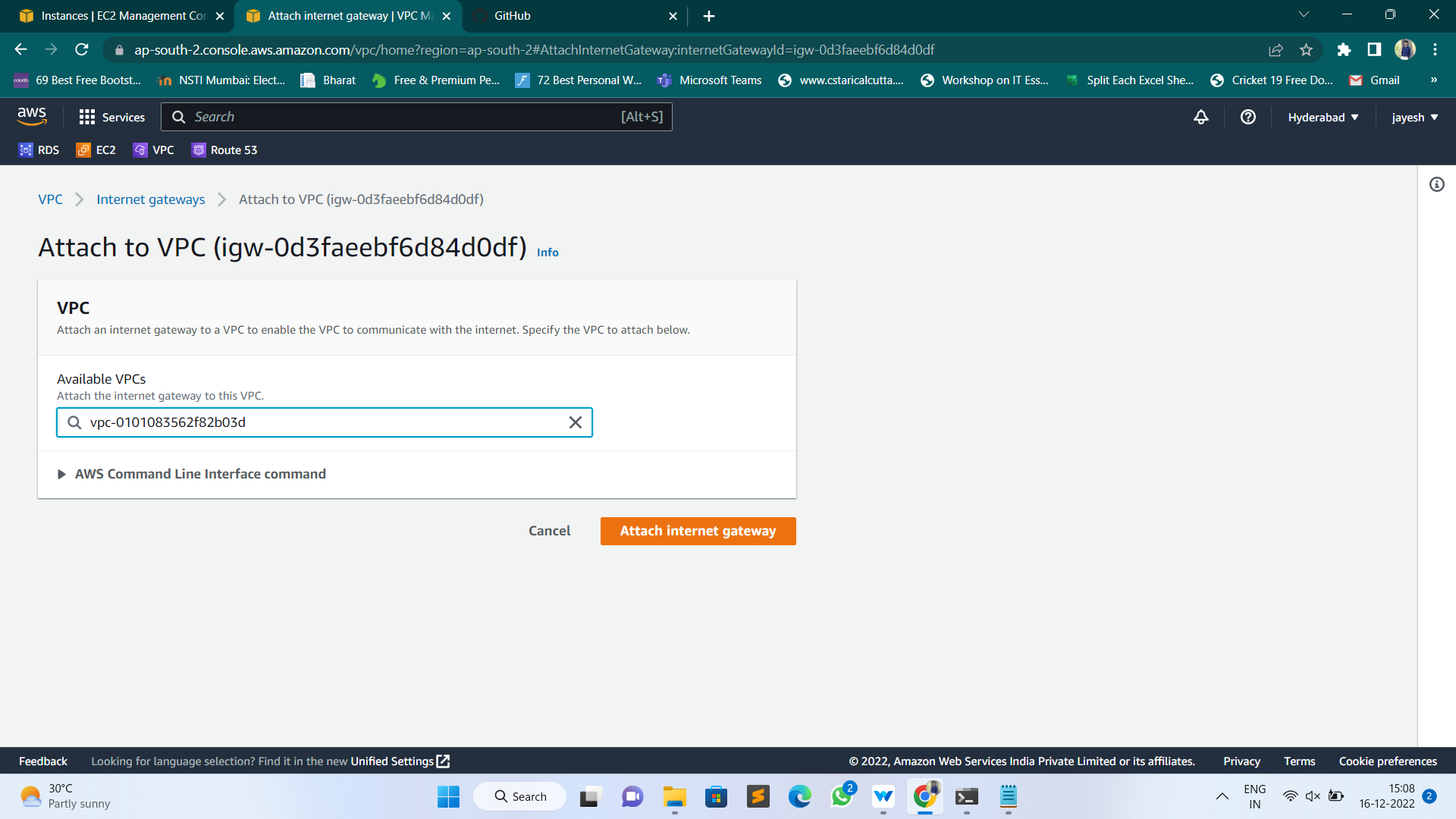
* Enter the VPC Name as Bastion-VPC
* Enter CIDR Range for VPC is 10.0.0.0/16



* After that click on create VPC, then VPC is created
* Now create Internet Gateway and attach it with the VPC created
* Go to Internet Gateway, click on Internet Gateways
* Enter the Name tag as Bastion-IGW and click on create Internet Gateway

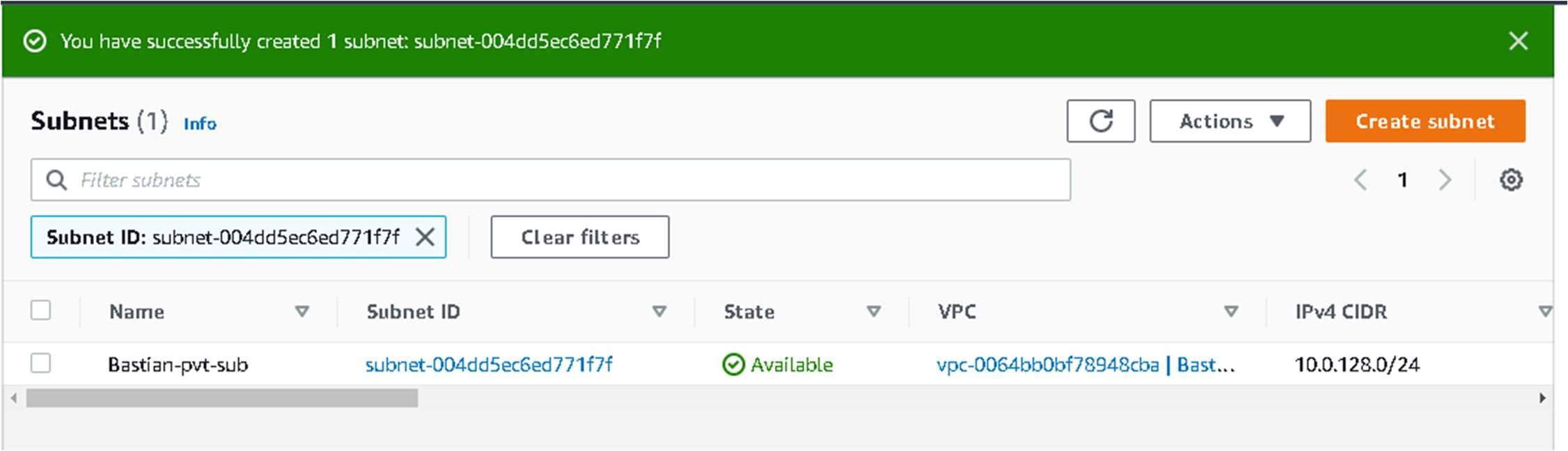


* After Creating Internet Gateway Attach it with VPC

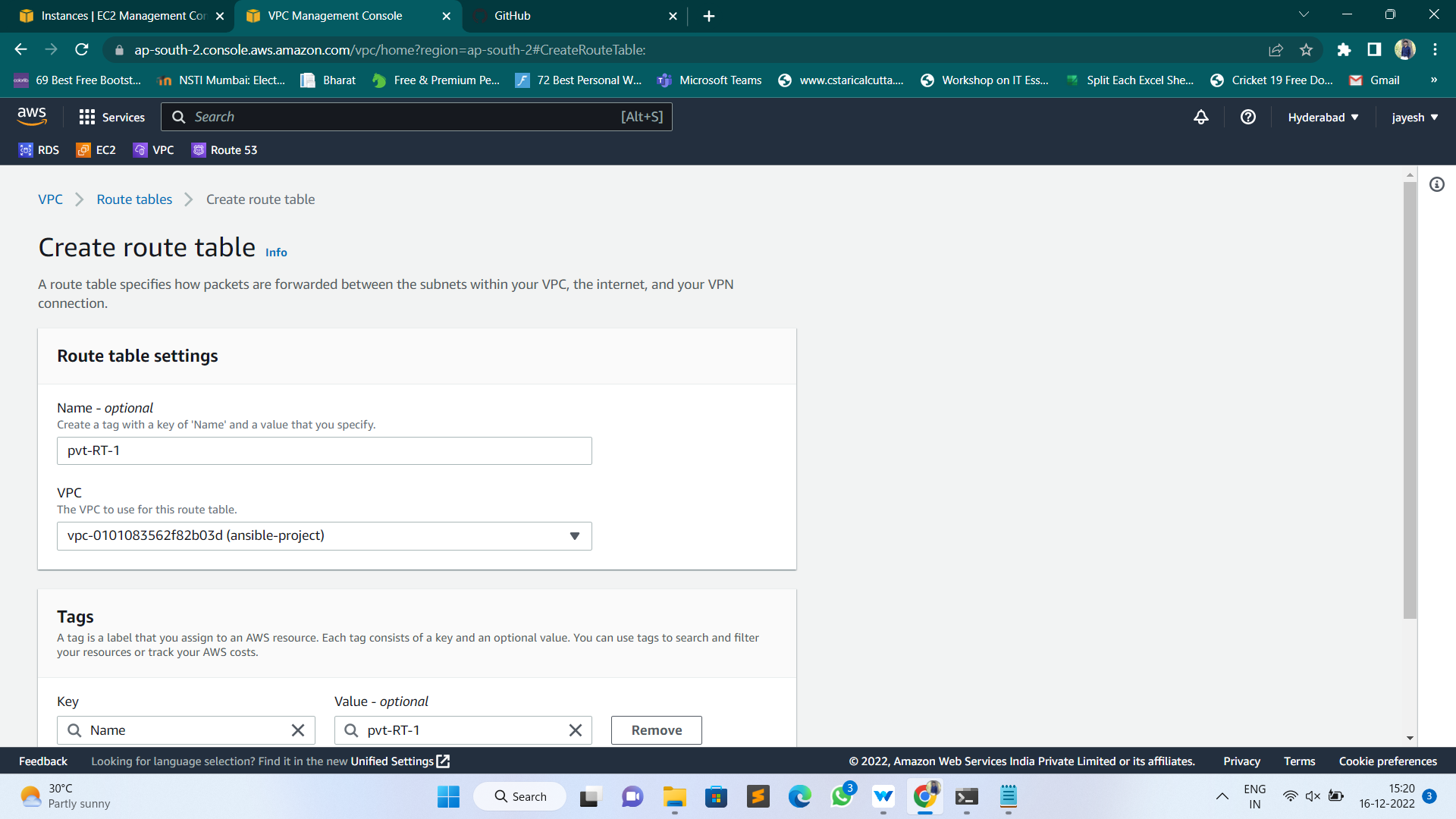


* Create Subnets, go to subnets and click on create Subnets
* Select the VPC Under which you want to create
* Enter Subnet name and Availability zone
* Take CIDR range as 10.0.0.0/24 and click on create subnet.

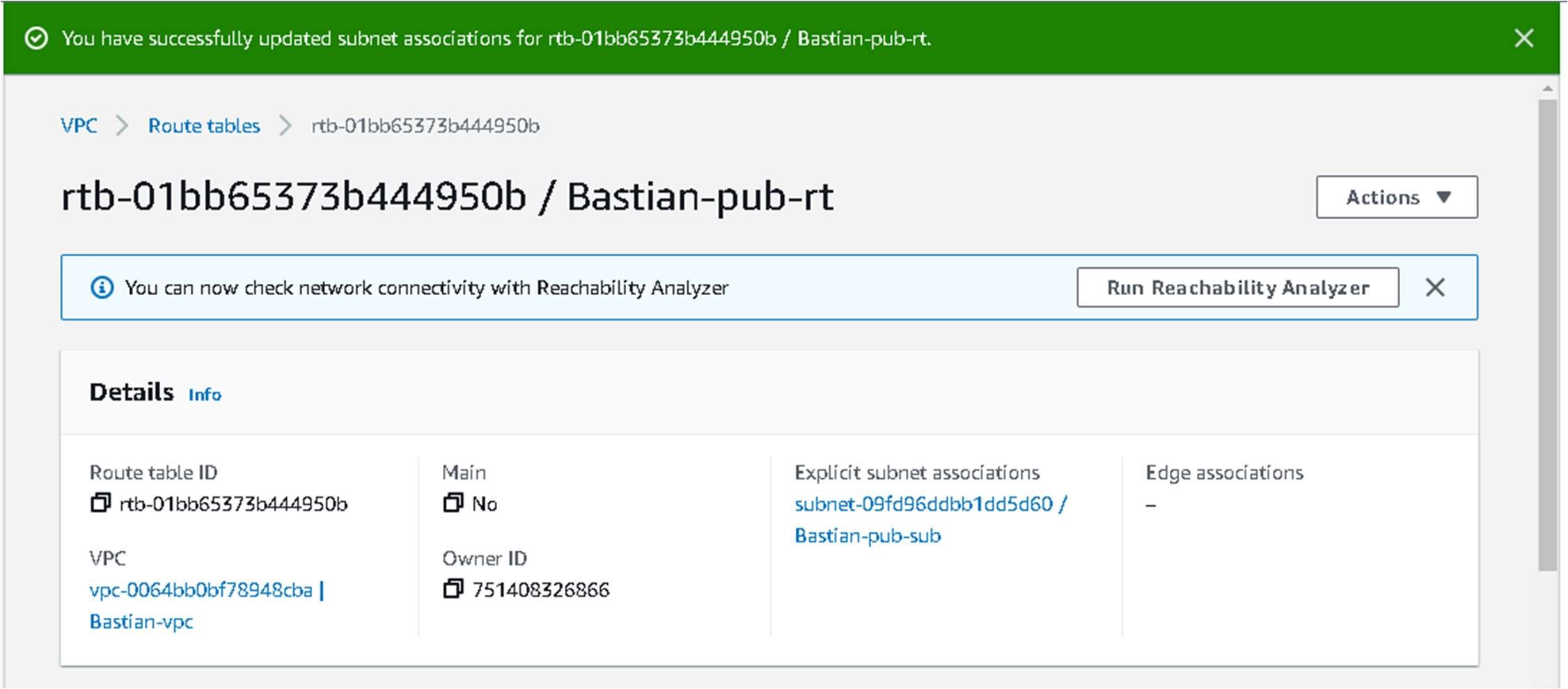
* Next, create Private subnet with CIDR Ranges of 10.0.128.0/24



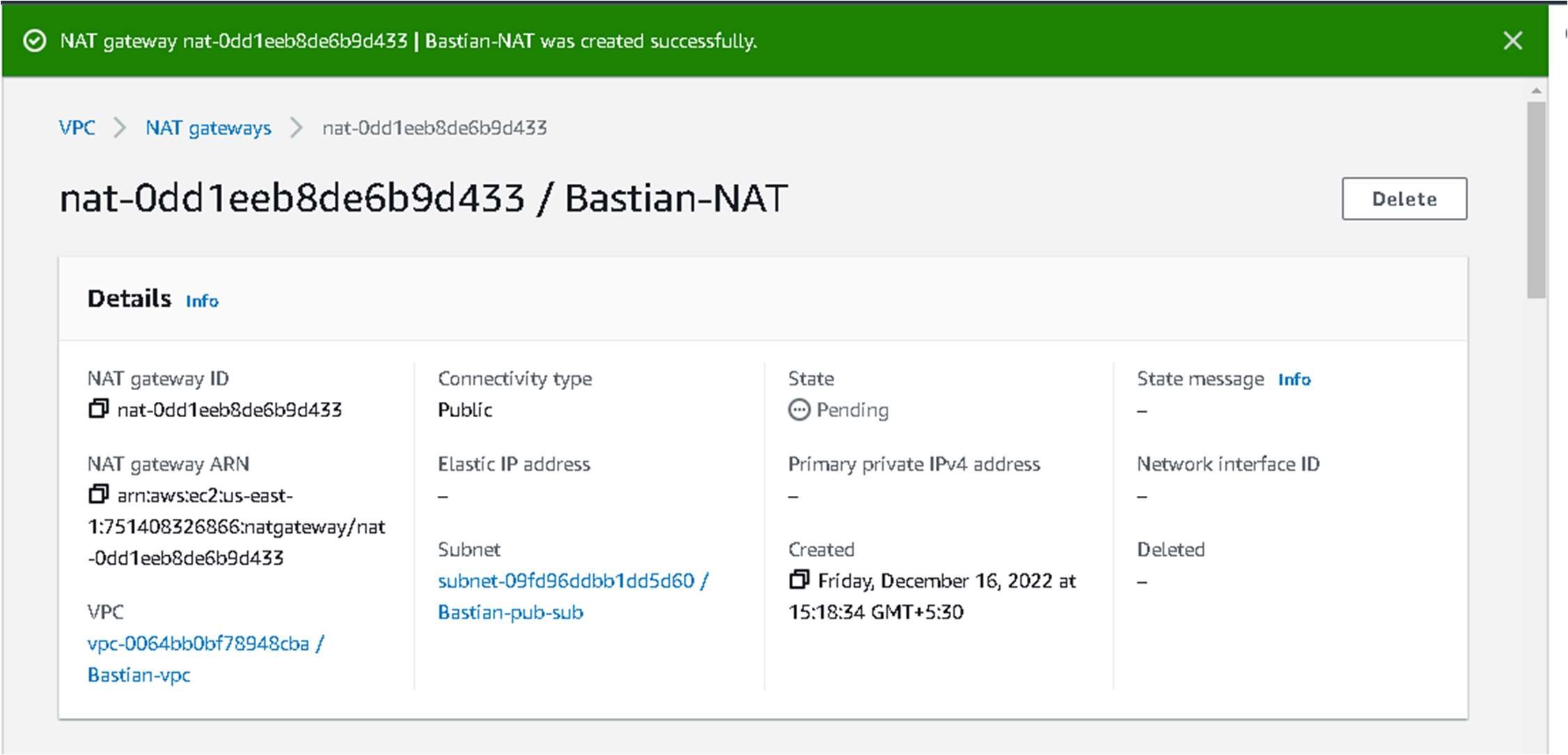
* Now go to Route Tables and click on create Route table
* Create One public route table, enter the name and select the VPC and click on create route table.



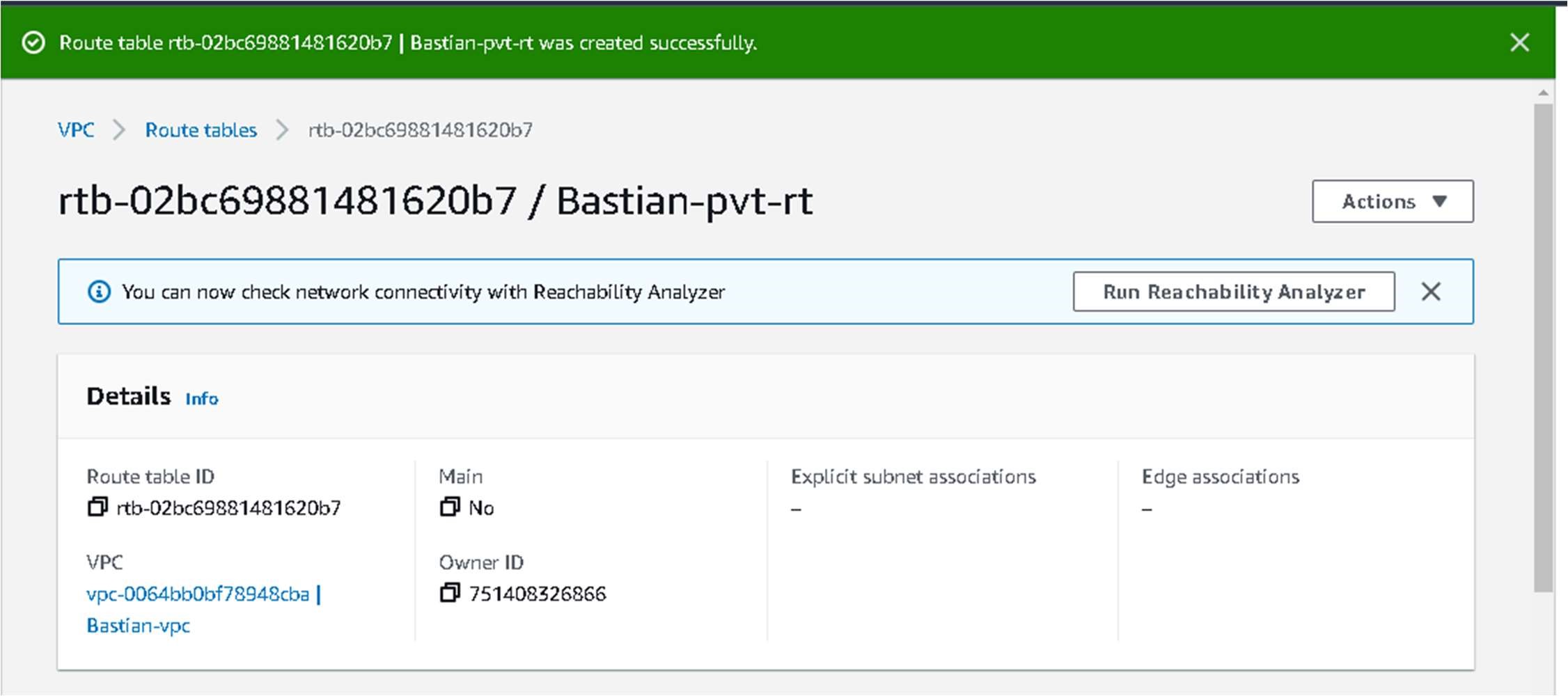
* Click on route, enter into edit routs and create a root in 0.0.0.0/0 with internet gateway  Go to subnet associations and associate with public subnet



* Now create NAT Gateway, go to NAT Gateways click on create NAT Gateway
* Enter the name and subnet (Bastion-pub-sub, click on Allocate elastic ip and click on create NAT Gateway

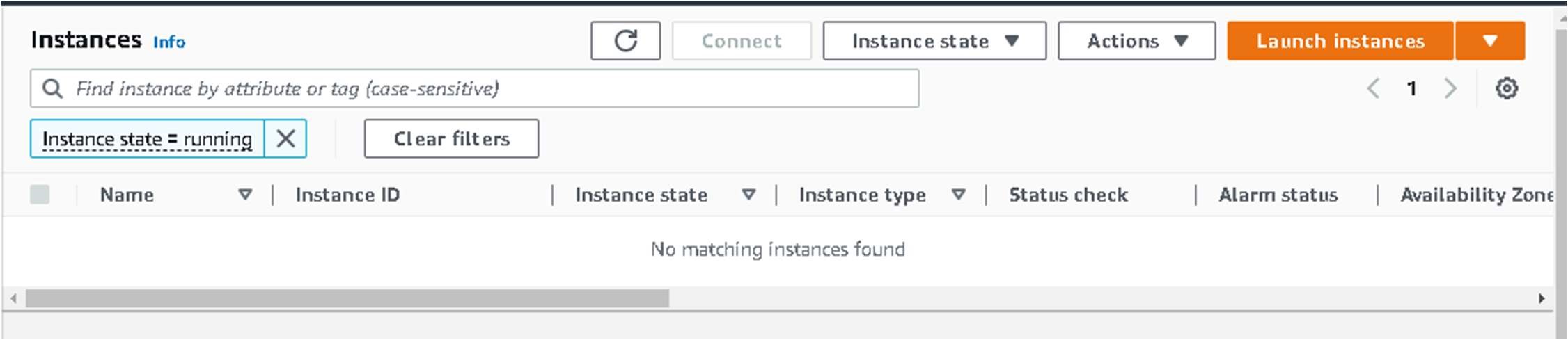


* Again go to route tables and create one private route table similar to public route table .  Click on create route table, Mention name as Bastion-PRIVATE-SUB, select your VPC.
* Go to routs, click on edit routs, create a root with 0.0.0.0/0 with NAT Gateway
* Associate this route table with private subnet with subnet associations.



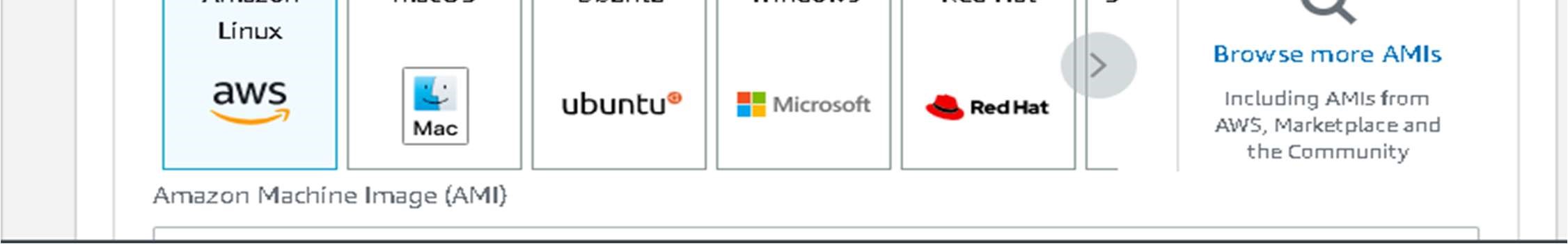
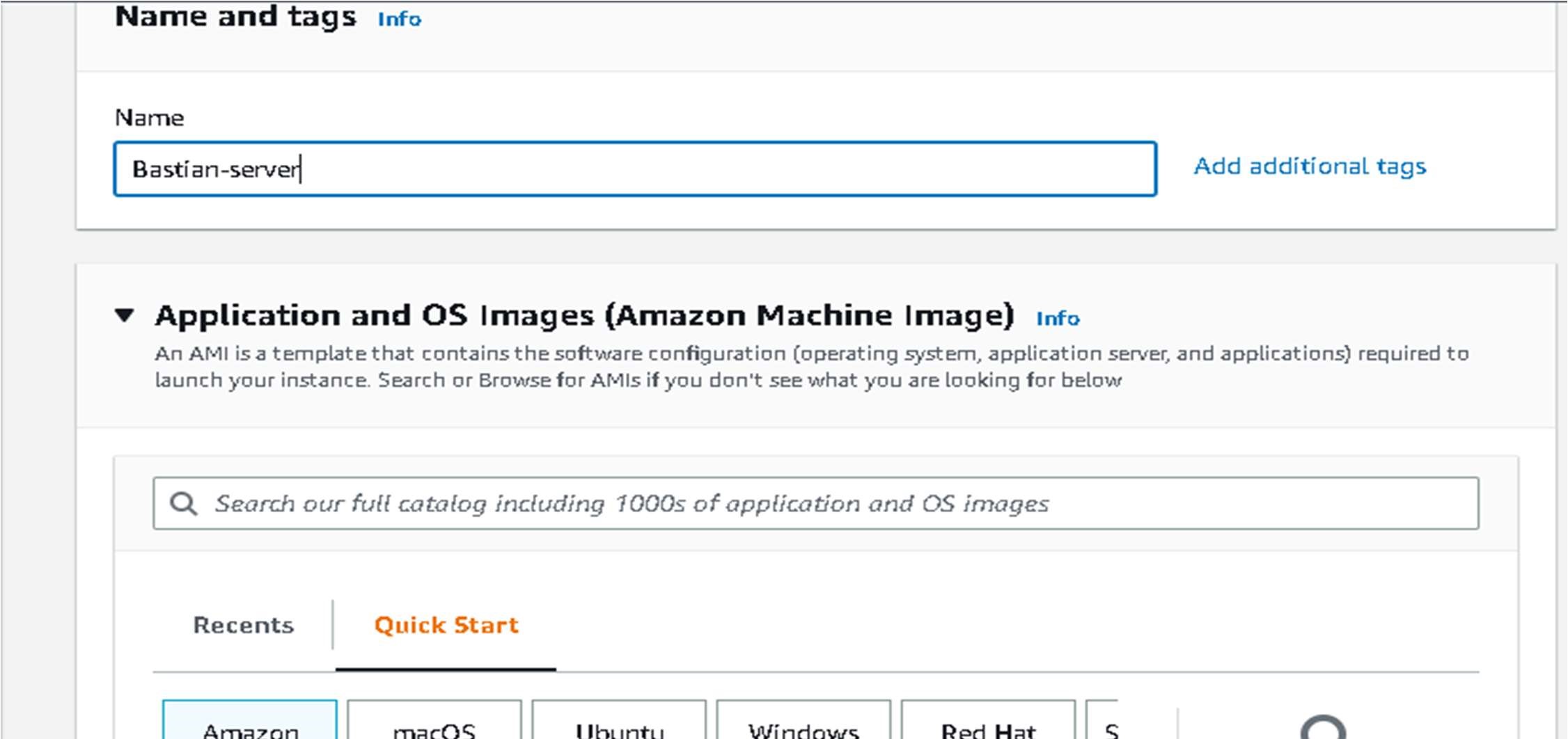
# **Module 2: create, configure and Launch Two linux2 ec2 instances**

EC2: Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the Amazon Web Services (AWS) Cloud. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage.



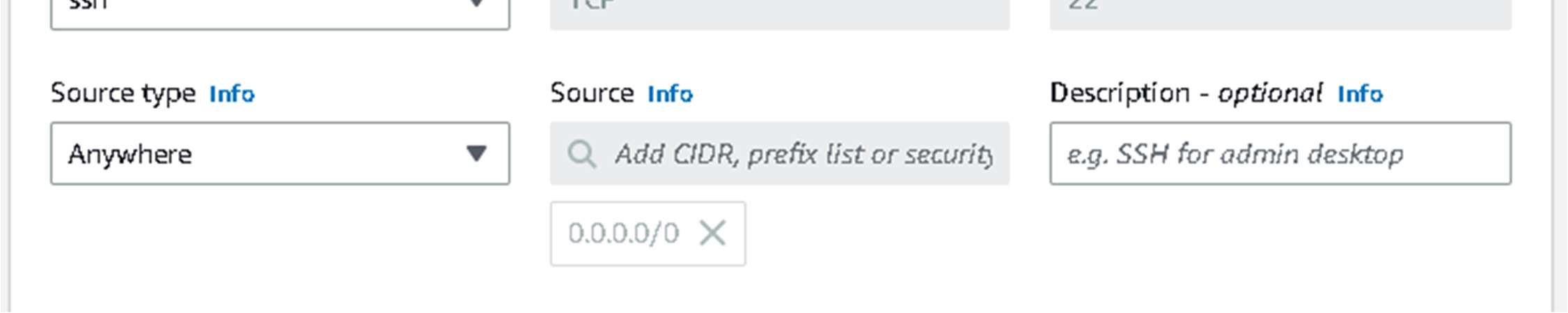
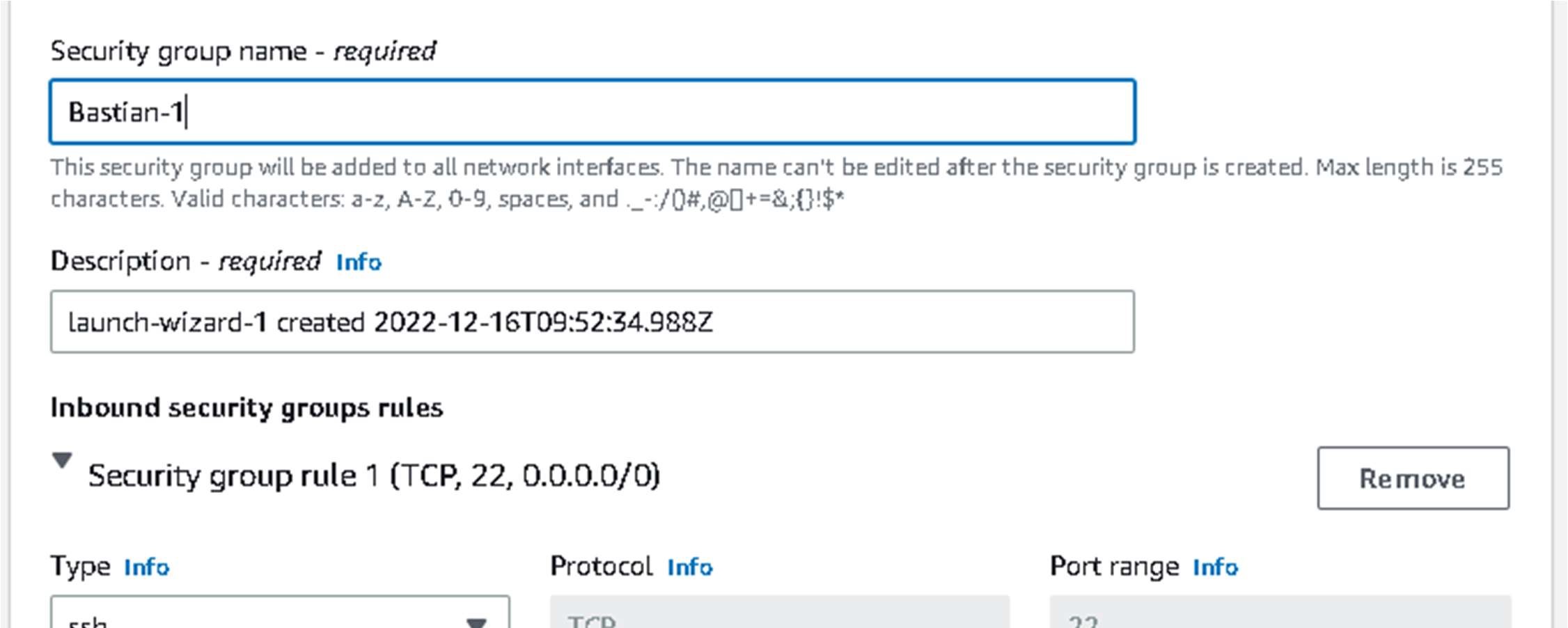
## Choosing an AMI – Ubuntu Linux AMI

An AMI is a virtual image used to create a virtual machine within an EC2 instance. You can also create multiple instances using a single AMI when you need instances with the same configuration. You can also create multiple instances using different AMI when you need instances with a different configuration.



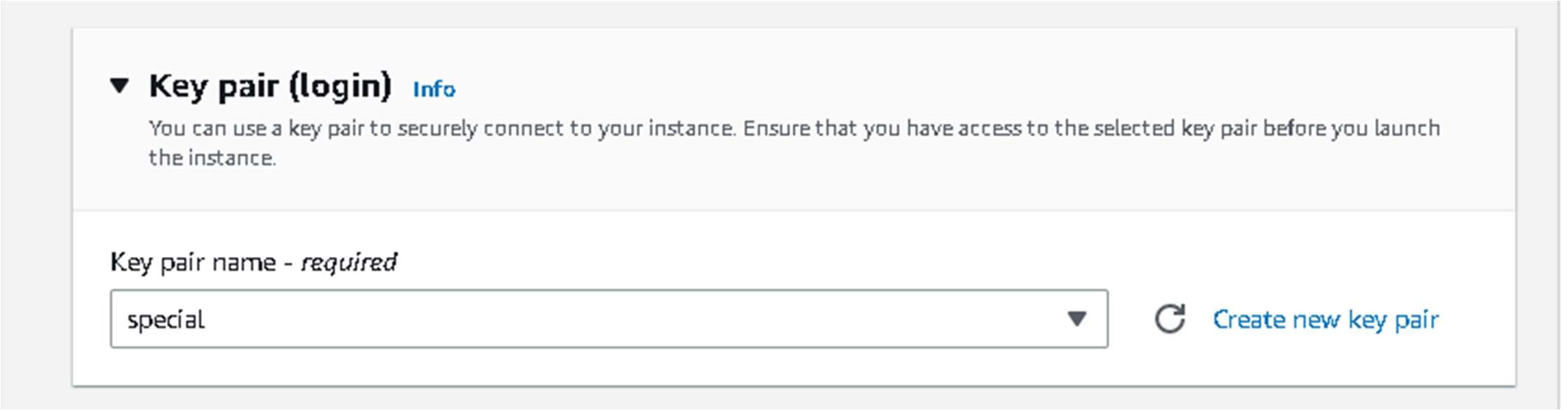
## Configure the Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. Here, I am assigning the port numbers for the inbound rule SSH-22.



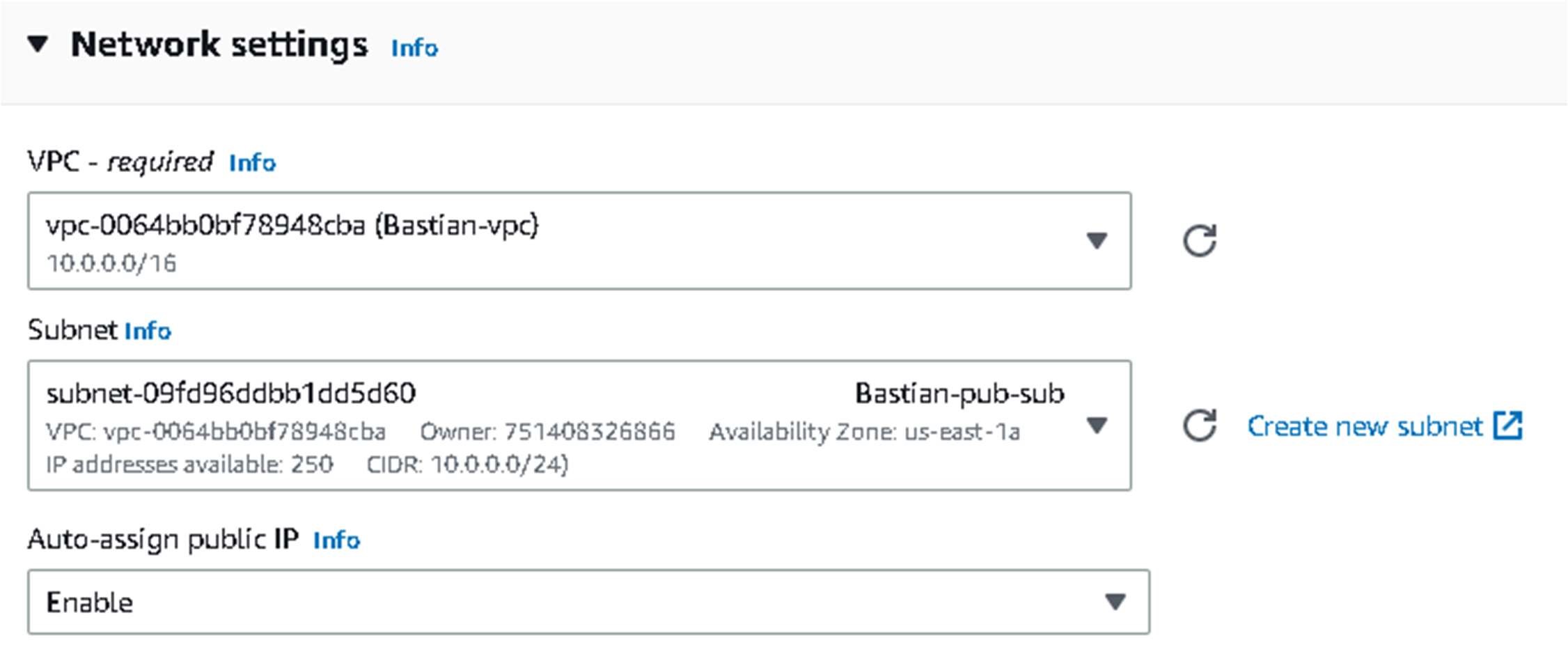
## Select an Existing keypair or create a new key pair

A key pair consists of a public key that AWS stores, and a private key file that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance. Amazon EC2 supports ED25519 and RSA key pair types.

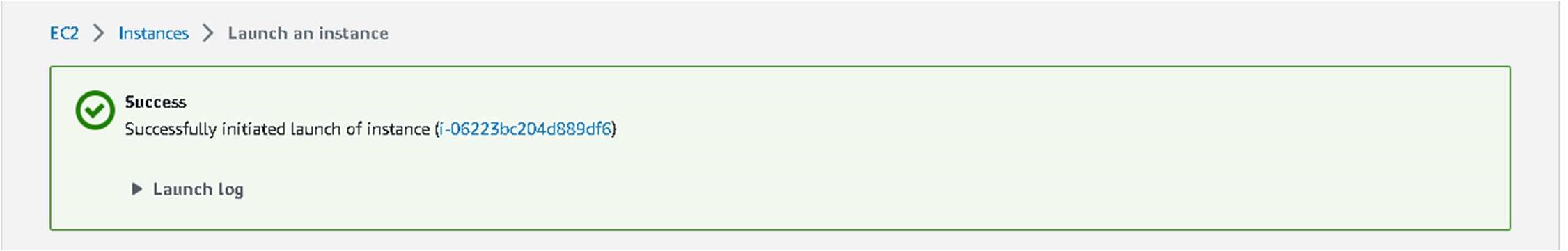


## Select VPC

Amazon Virtual Private Cloud (Amazon VPC) enables you to launch AWS resources into a virtual network that you've defined. This virtual network closely resembles a traditional network that you'd operate in your own data centre, with the benefits of using the scalable infrastructure of AWS.

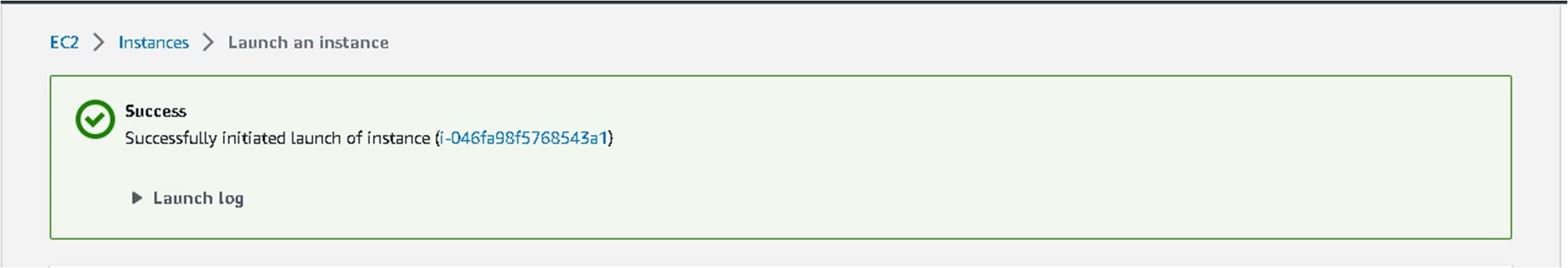


* Finally, Instance launched a WordPress-Server Instance.

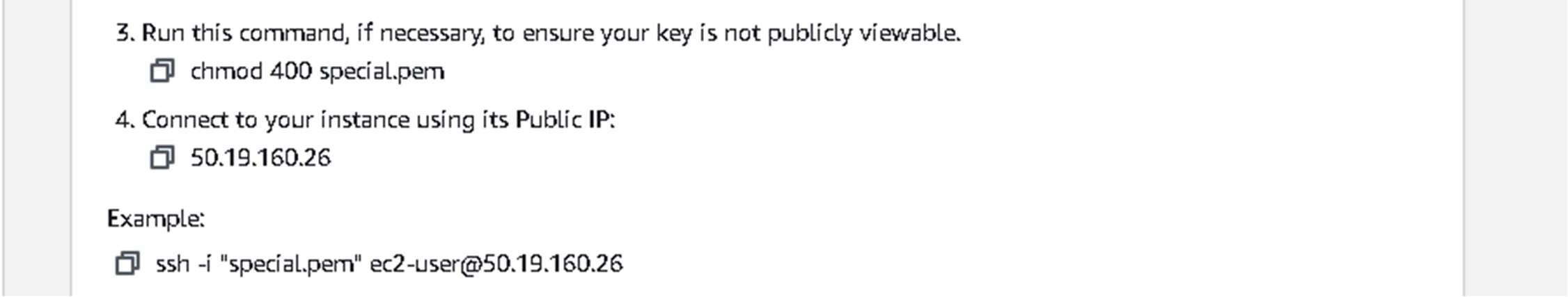
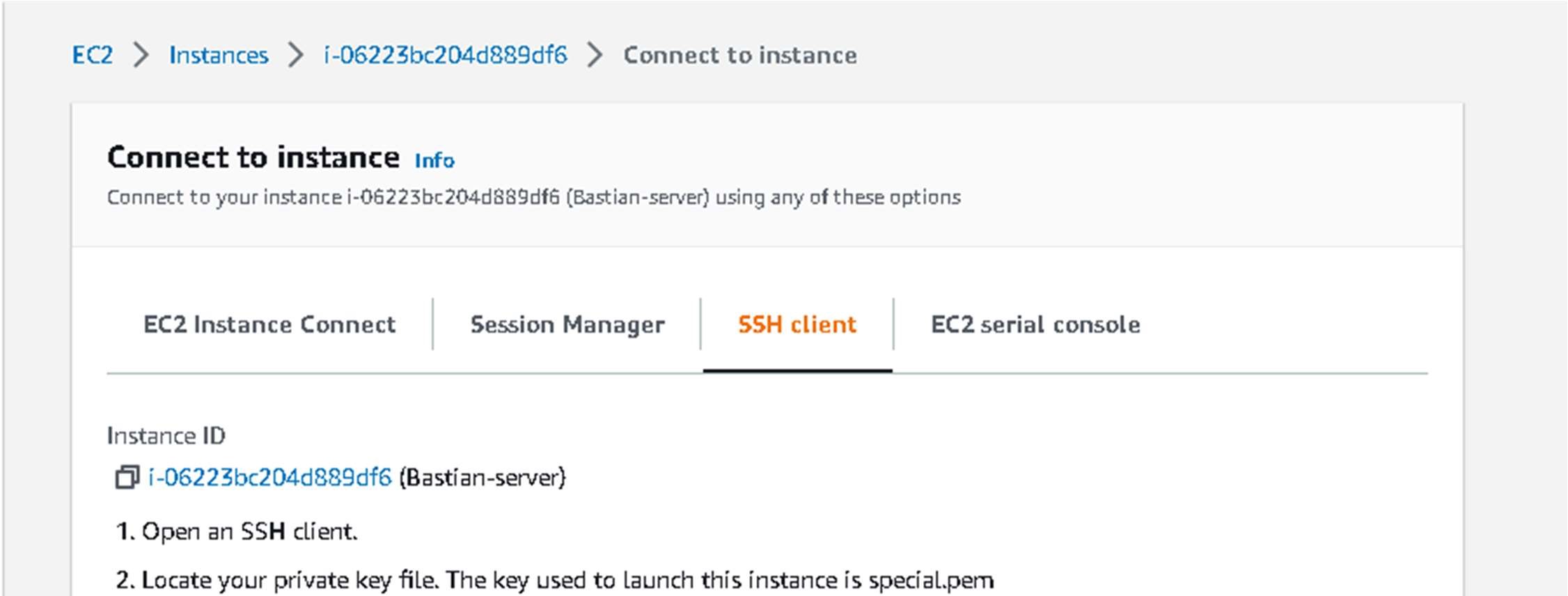


### Create second Instance

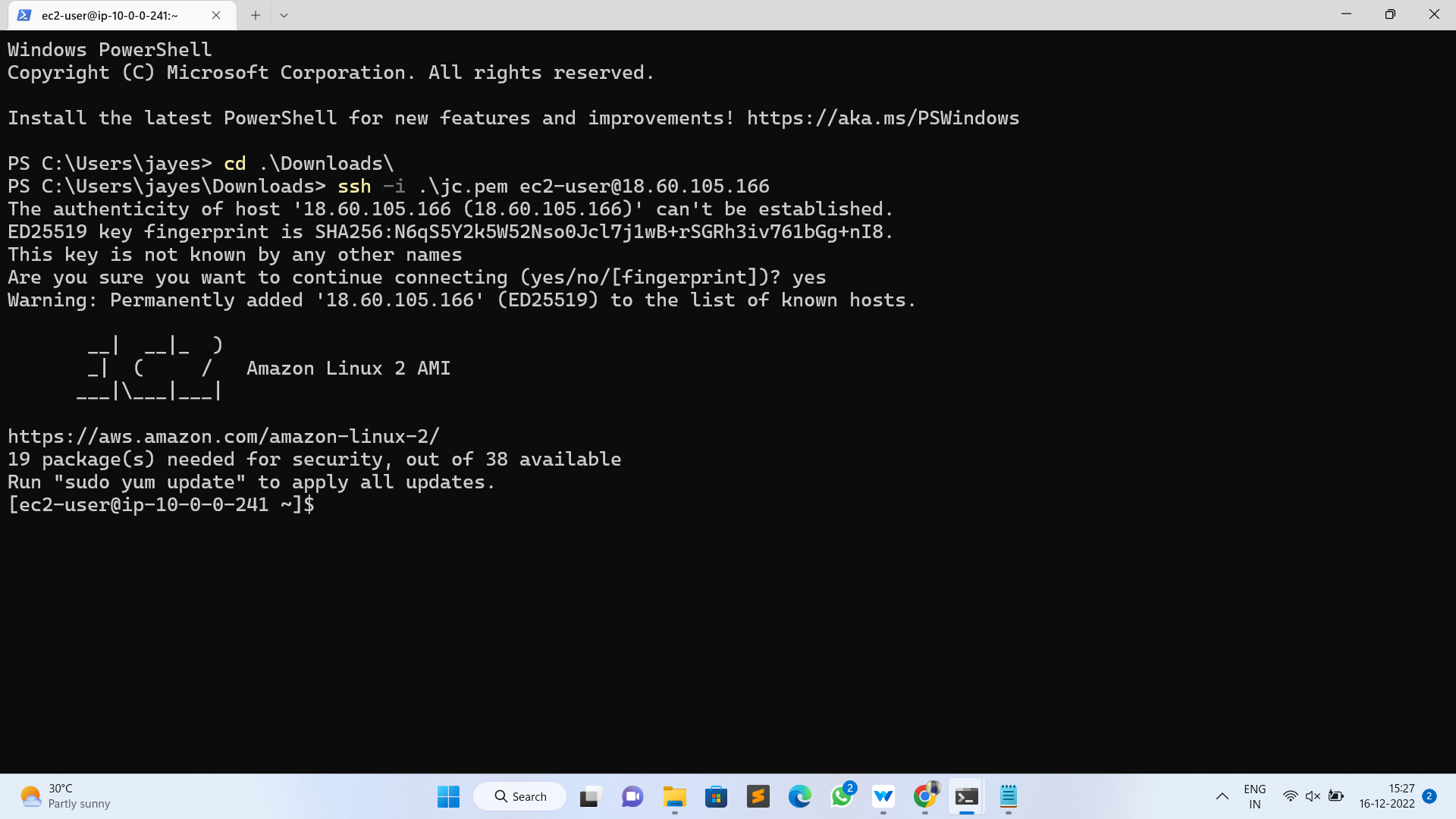
* Similarly, create another instance with the name of Bastion-private-server and connect it with VPC and private subnet. And launch.



### Now to connect an public EC2 instance, we can connect with console



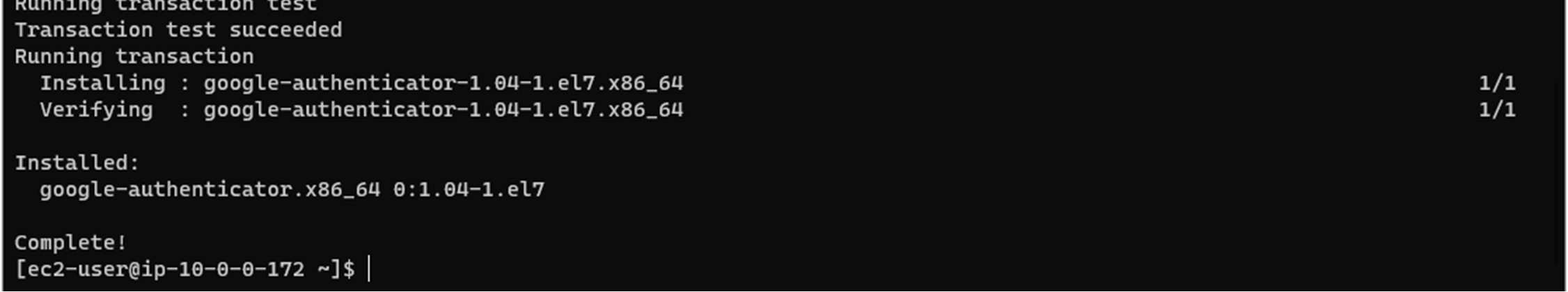
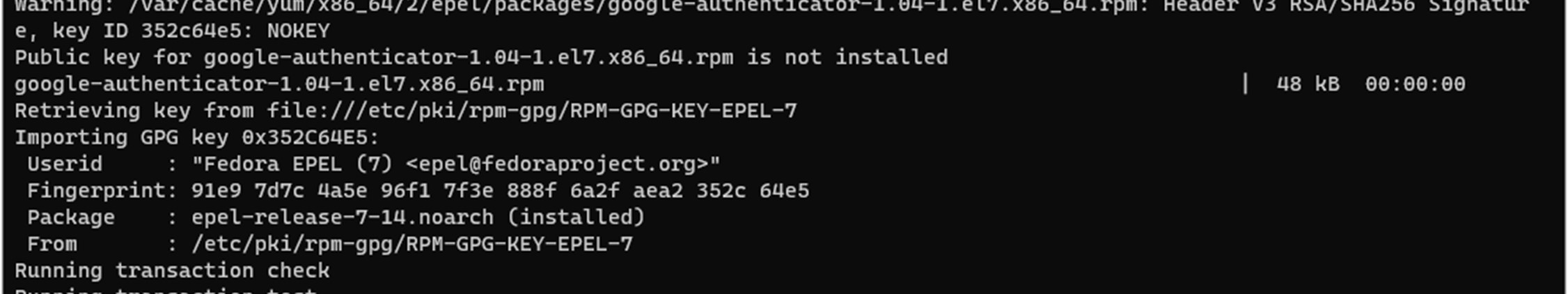
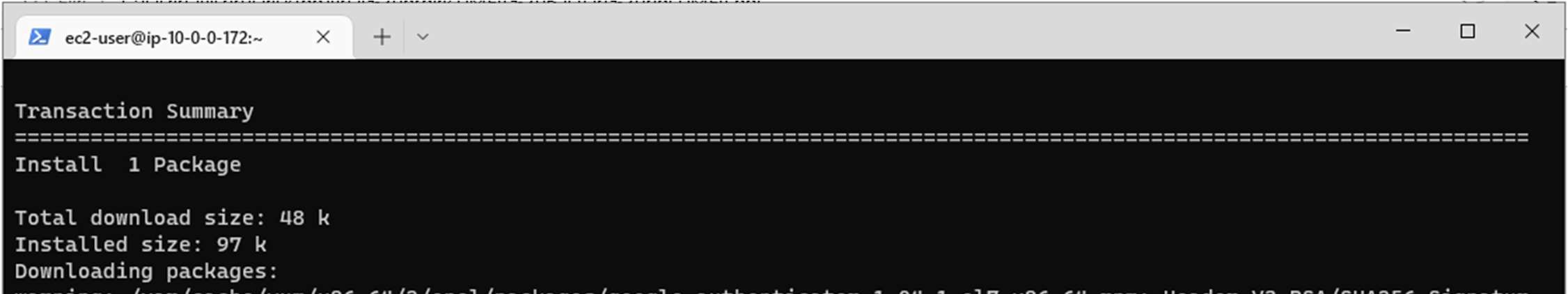
* Go to the instance page and click on connect then go to SSH Client/EC2 instance connect.
* By using the SSH command we are accessing our instance in Command Line Interface Terminal.



**Module 3: Assigning Multifactor Authentication to private instance.**

* Google Authenticator is part of the EPEL repo and you should install the EPEL repo in your EC2 instance.
* In this case, I am using Amazon EC2 Instance (OS) and I can download my EPEL 7 and install it like this

* sudo yum install -y https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm  sudo yum install google-authenticator.x86\_64 -y



### Configuring Google Authenticator

* Run the initialization app:
  + google-authenticator
* You will be asked if the authentication tokens used should be time-based. In this example, we will use time-based tokens.
  + Do you want authentication tokens to be time-based (y/n) y
* After this it will show QR code, You have to scan it with your external device.



* + Do you want me to update your "/home/ec2-user/.google\_authenticator" file (y/n) y
  + Do you want to disallow multiple uses of the same authentication token? This restricts you to one login about every 30s, but it increases your chances to notice or even prevent man-in-the-middle attacks (y/n) y
  + Do you want to disallow multiple uses of the same authentication token? This restricts you to one login about every 30s, but it increases your chances to notice or even prevent man-in-the-middle attacks (y/n) y
  + By default, tokens are good for 30 seconds and in order to compensate for possible time-skew between the client and the server, we allow an extra token before and after the current time. If you experience problems with poor time synchronization, you can increase the window from its default size of 1:30min to about 4min.

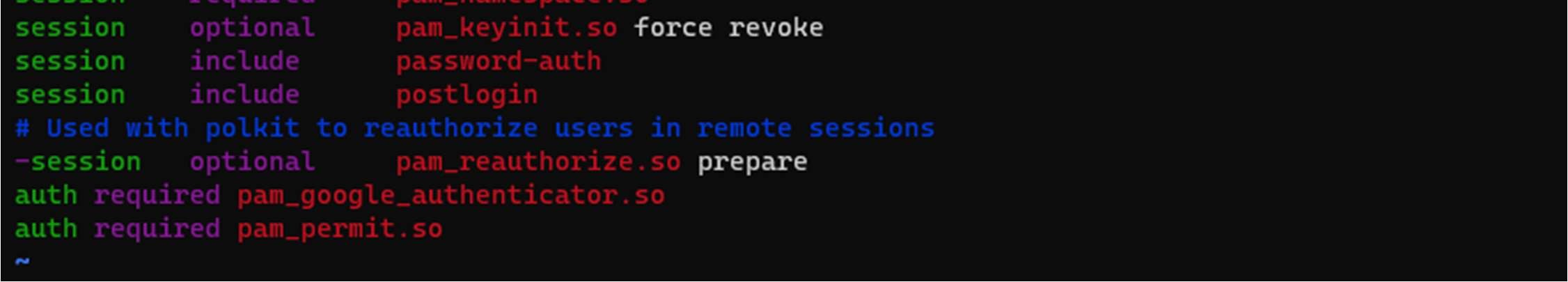
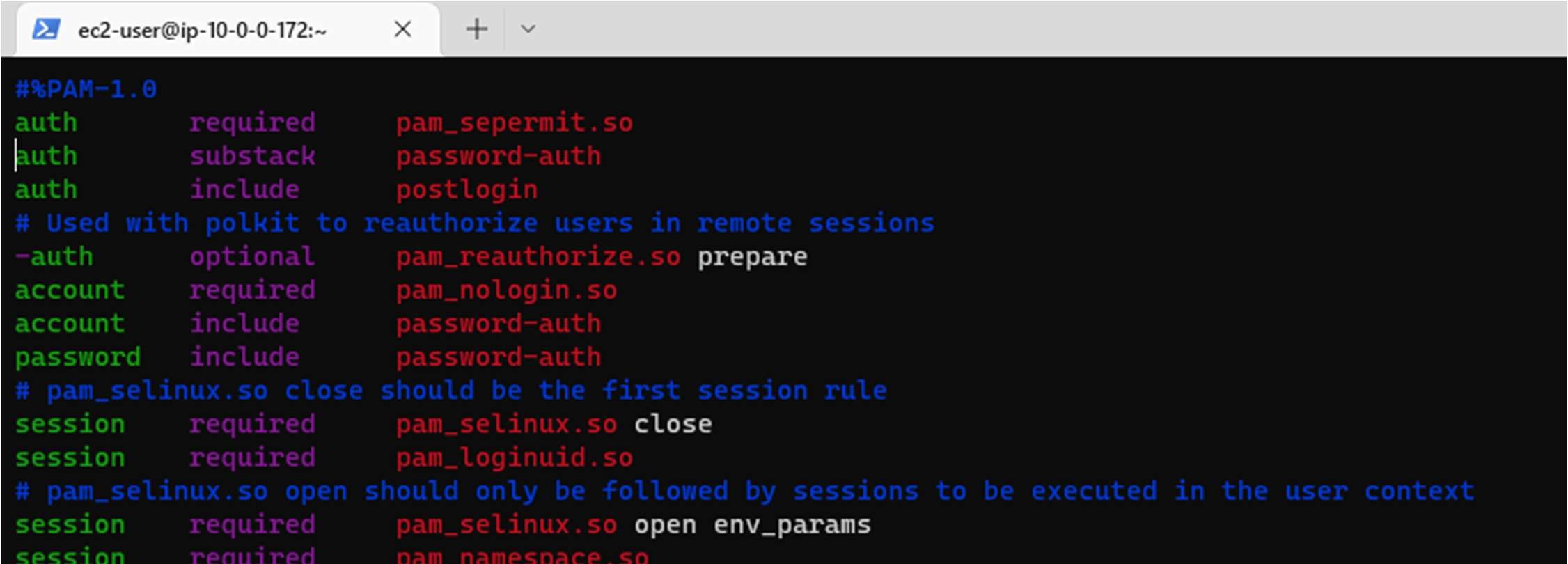
Do you want to do so (y/n) n

* If the computer that you are logging into isn't hardened against brute-force login attempts, you can enable rate-limiting for the authentication module. By default, this limits attackers to no more than 3 login attempts every 30s. Do you want to enable rate-limiting (y/n) y

### Configure SSH to use the Google Pluggable Authentication Module

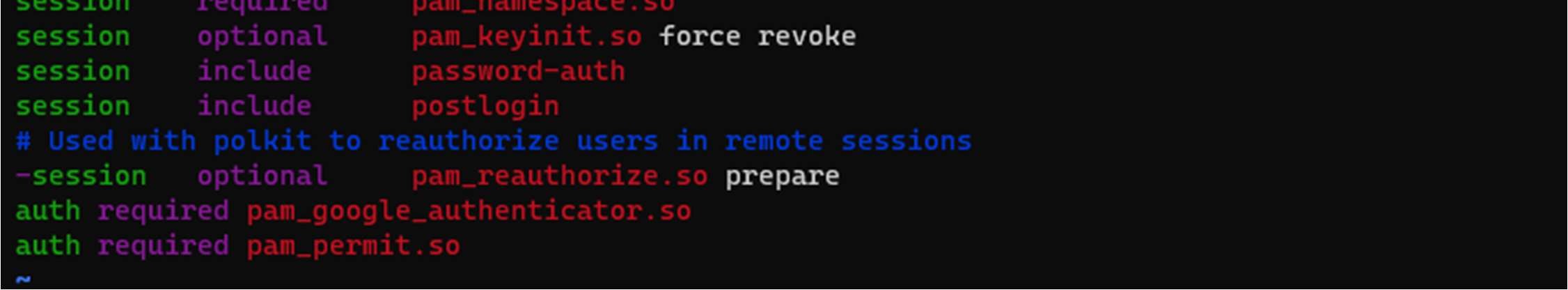
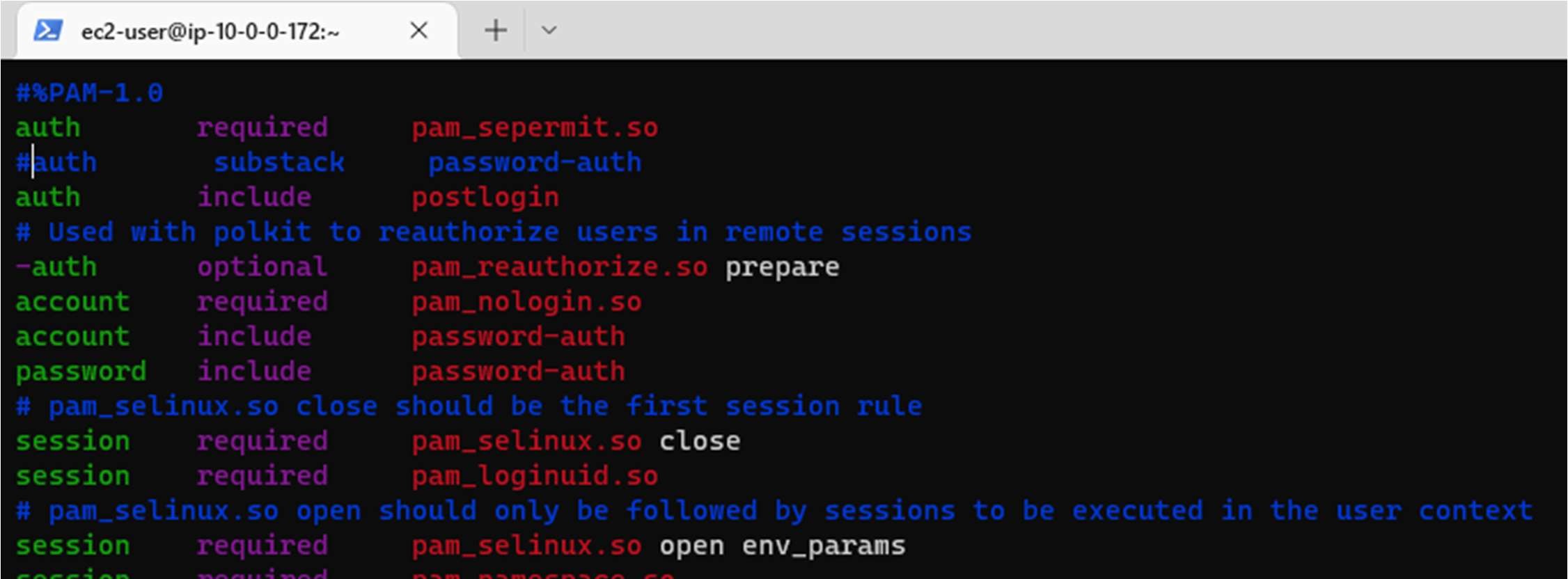
* Run the following command to make changes to the PAM configuration.  sudo vi /etc/pam.d/sshd
* Add the following to the bottom of the file to use Google Authenticator. If there are service accounts or users who should be able to log in without MFA, add nullok at the end of the following statement. This will mean that users who don’t run Google Authenticator initialization won’t be asked for a second authentication.

auth required pam\_google\_authenticator.so auth required pam\_permit.so



* Comment out the password requirement. We just want to use the key pair and the verification code generated on the Google Authenticator app.

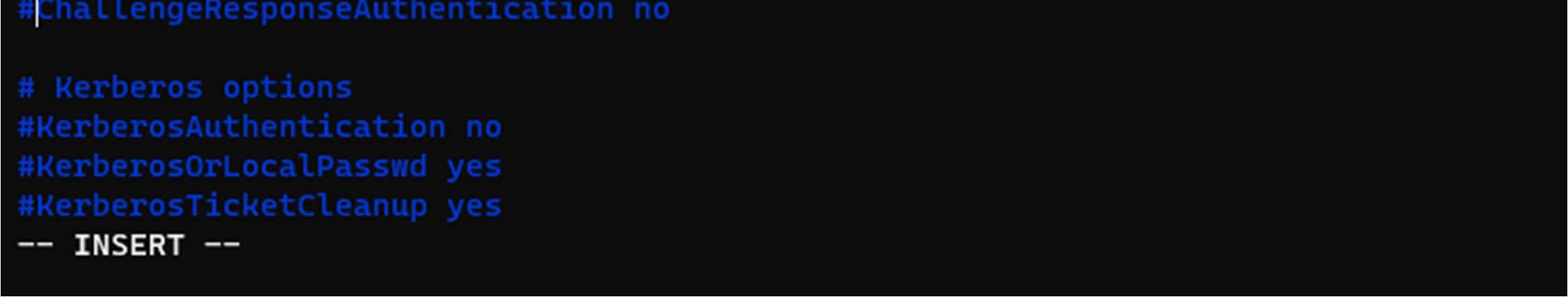
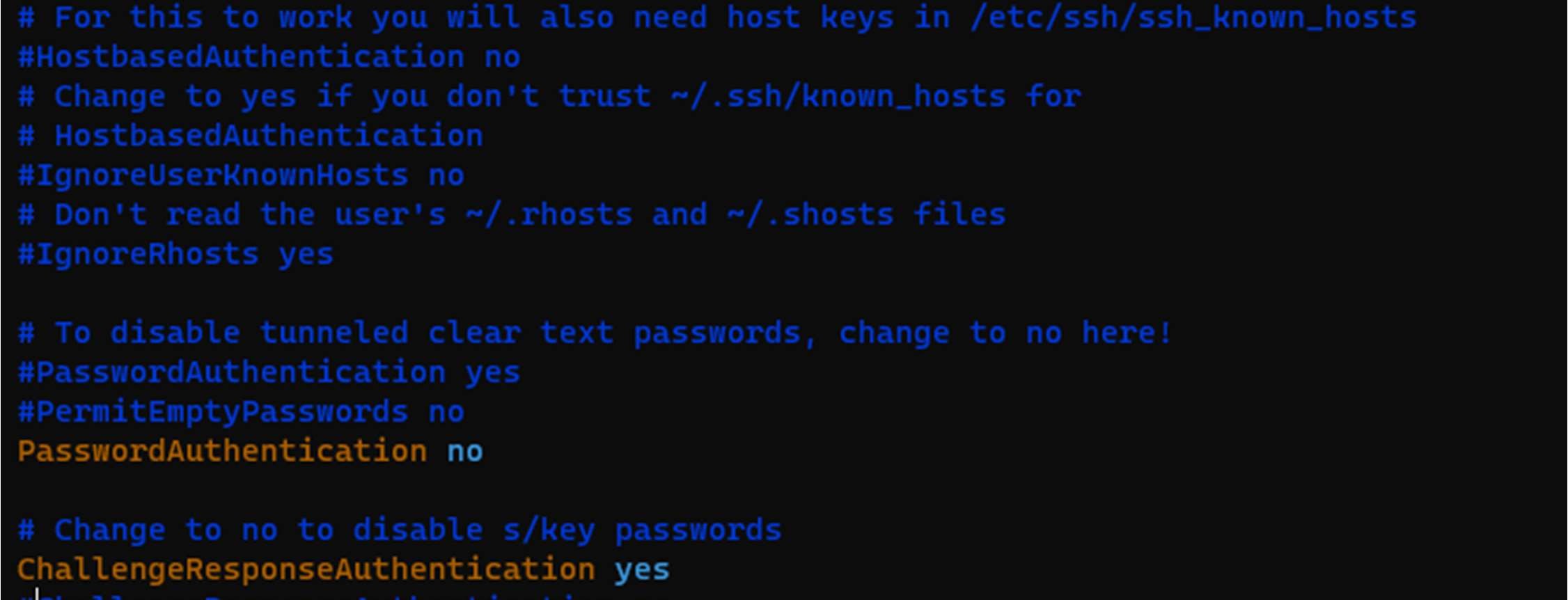
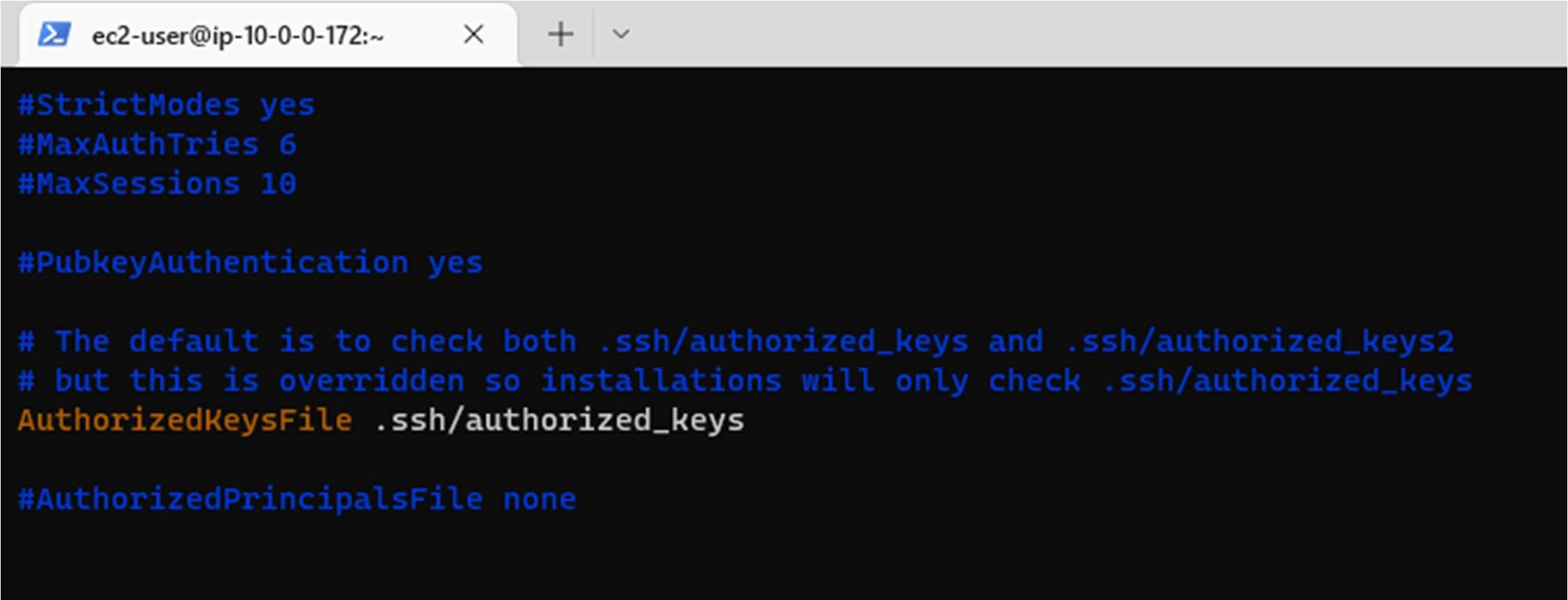
#auth substack password-auth



* Save the file. Next, we need to change the SSH configuration to make it prompt for a second authentication
* Now Run
  + sudo vi /etc/ssh/sshd\_config
* Comment out the line which says ChallengeResponseAuthentication ‘no’ and uncomment the line which says ‘yes’.

#ChallengeResponseAuthentication yes

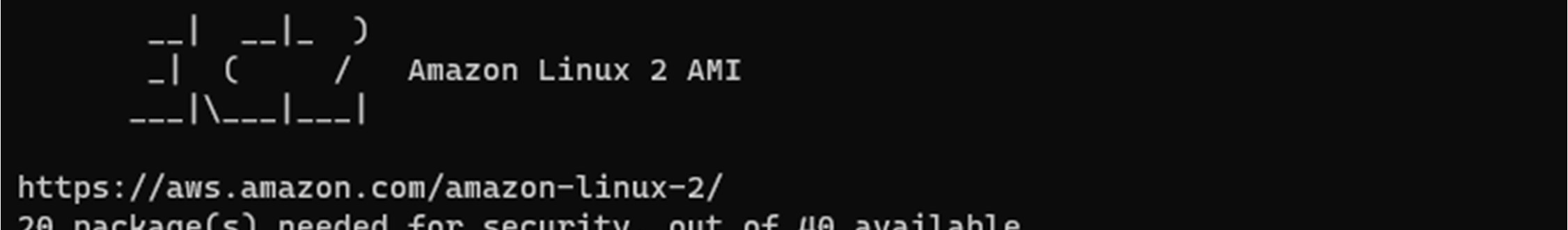
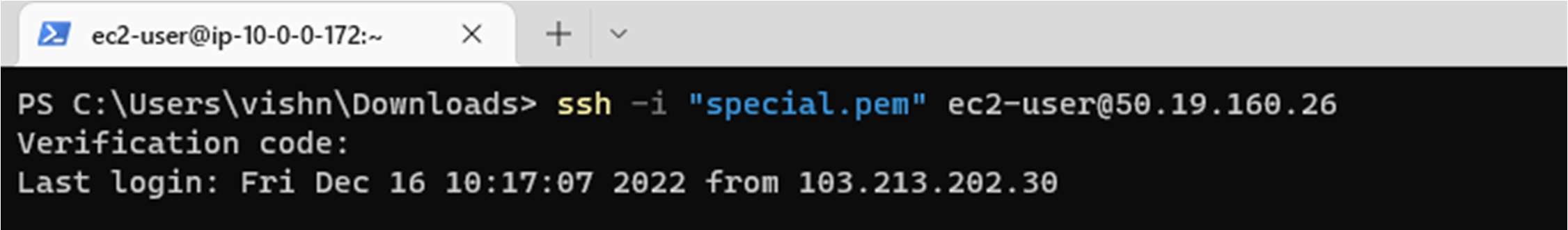
ChallengeResponseAuthentication no



* Lastly, we need to let SSH know that it should ask for SSH key and a verification code to let us in. At the bottom of the file add:

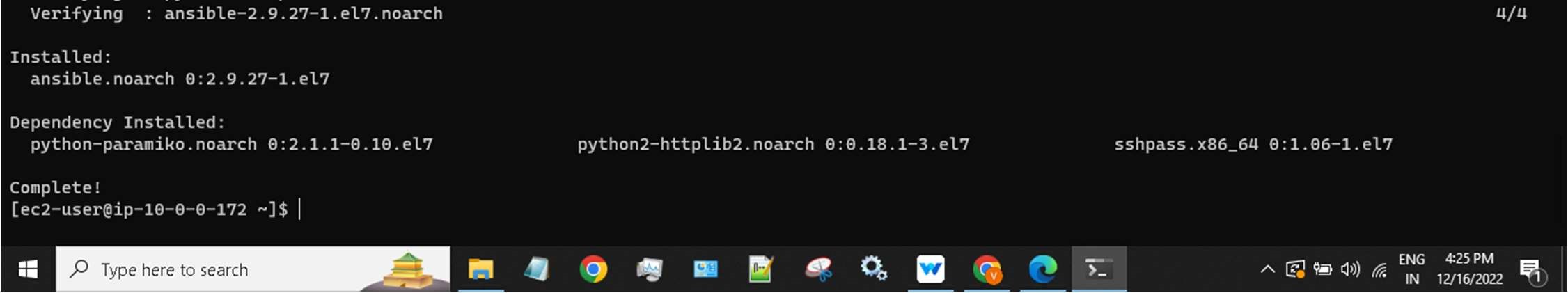
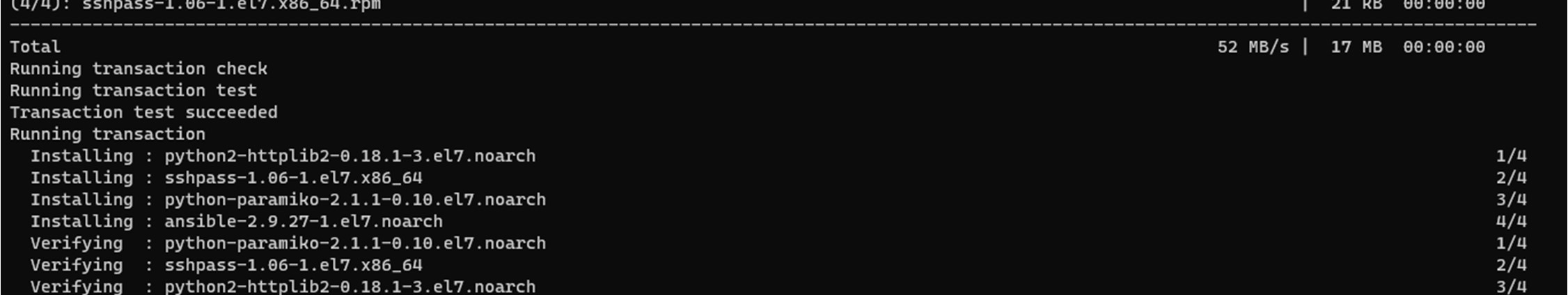
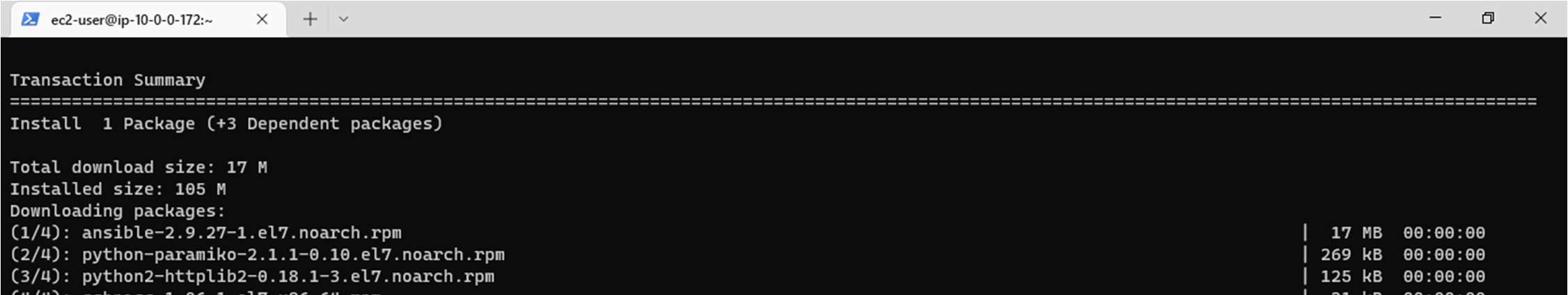
AuthenticationMethods publickey,keyboard-interactive

* Save the file. Restart the SSH to let the changes take effect
  + sudo systemctl restart sshd
* Exit from server, again connect with the server it will ask for verification code.

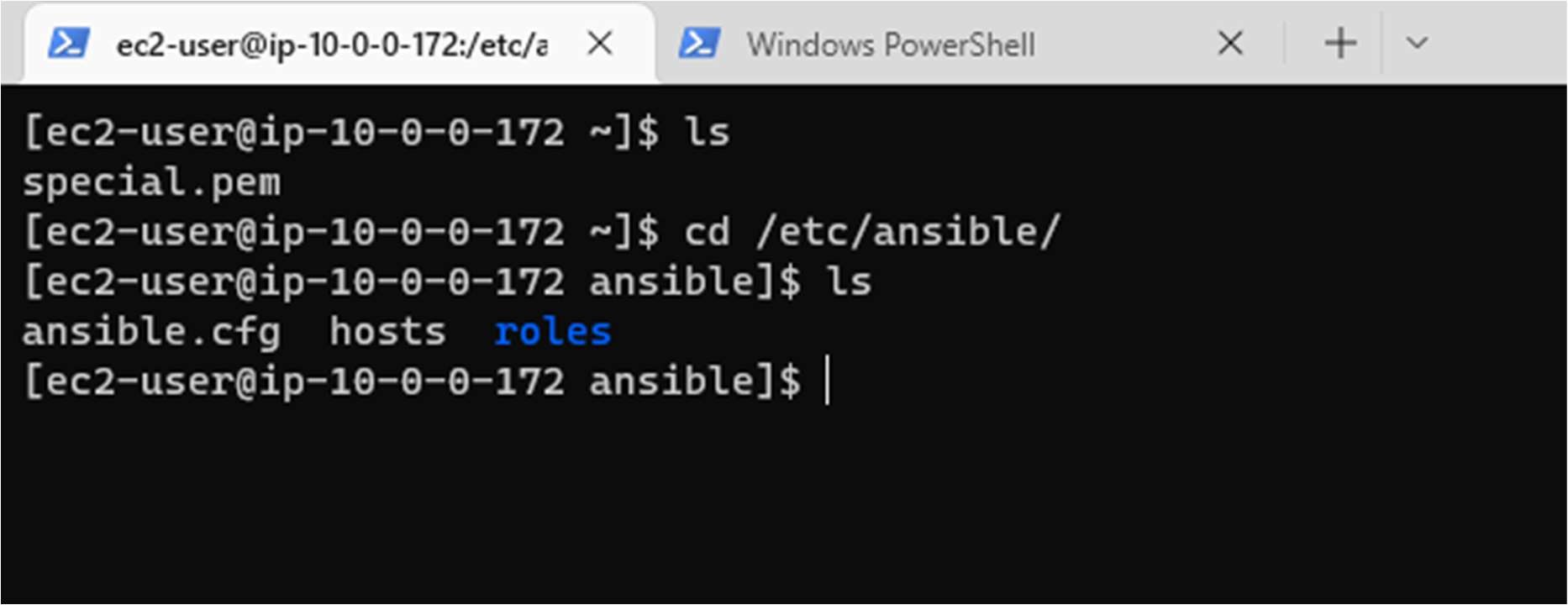


Module 3: Ping a private server

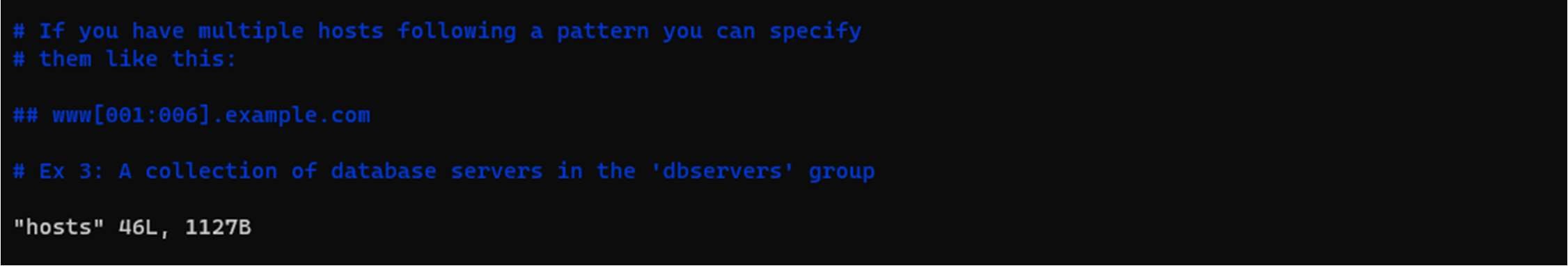
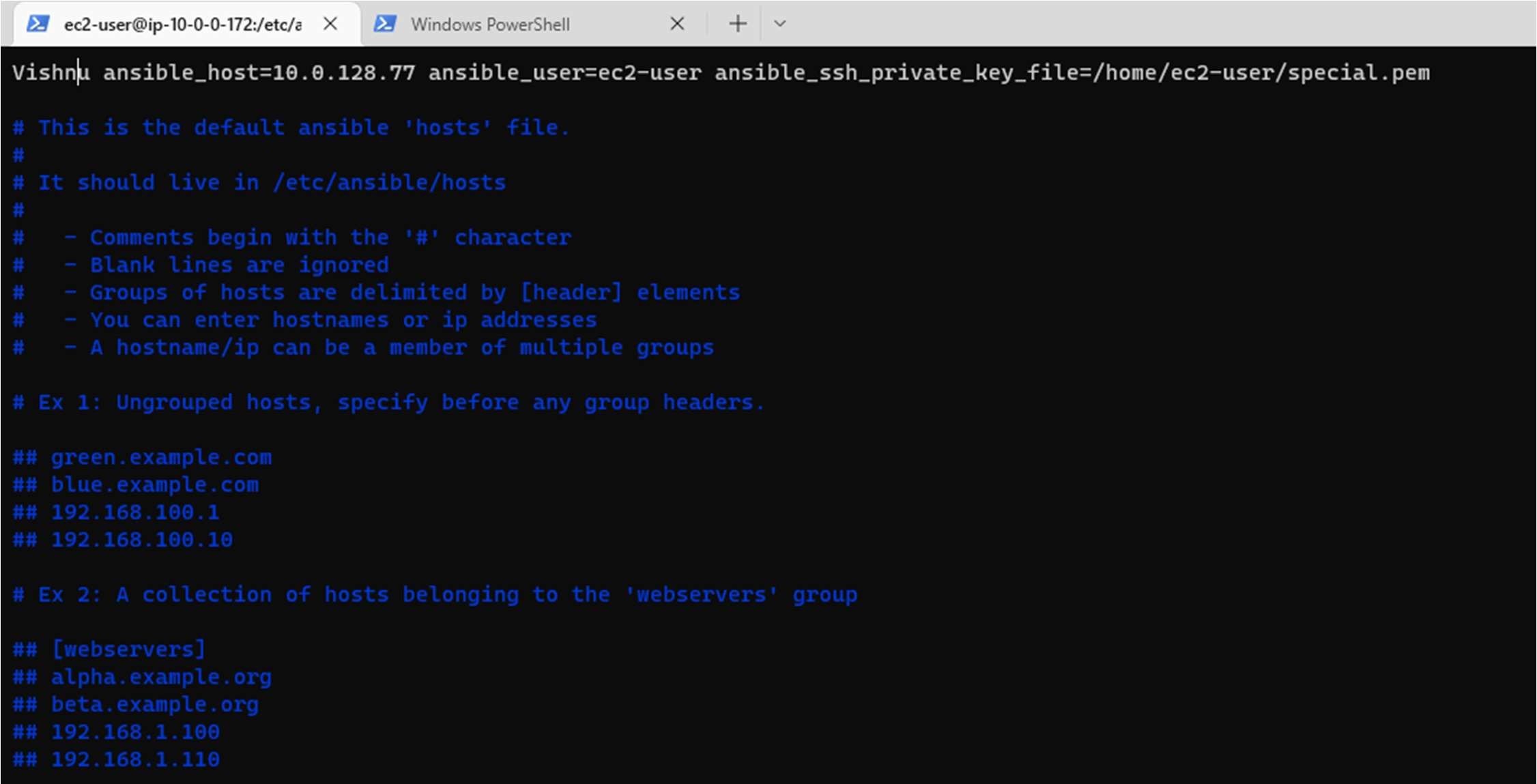
* Connect with Bastion server with verification code.
* Install ansible in Bastion server
  + sudo yum -y install ansible



* Now go to ansible with the help of below command
  + cd /etc/ansible/



* Edit the host with the command
  + Vishnu ansible\_host=10.0.128.77 ansible\_user=ec2-user ansible\_ssh\_private\_key\_file=/home/ec2-user/special.pem



* Give read permissions to the key file with below command.
  + sudo chmod 400 special.pem
* Now ping the private server with below command.
  + ansible all -m ping

