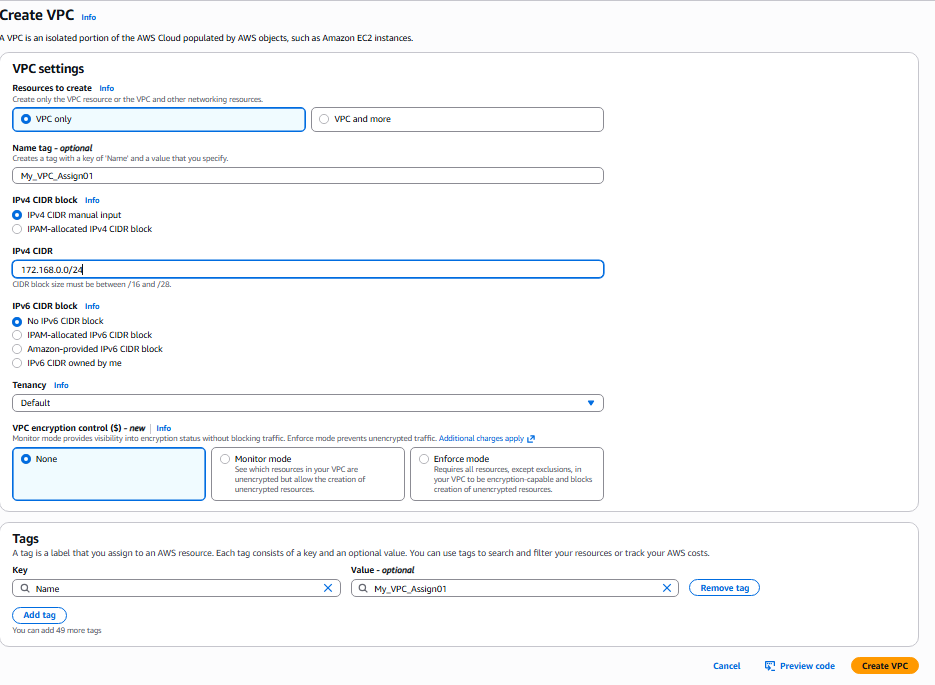
VPC-Assignment-01

1. Create VPC with 2 private and 2 public subnets.

Go to VPC services in the amazon console and select VPC -> create VPC on the top right corner

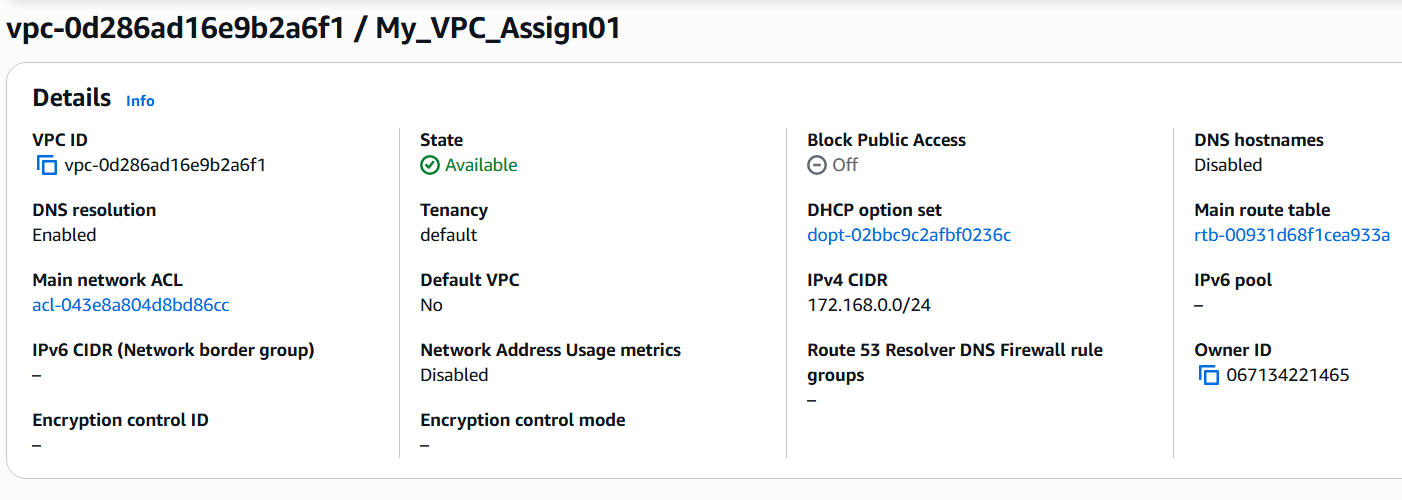
Now enter the name of the VPC (My\_VPC\_Assign01), Resource to create ->VPC only , IPv4 Cidr block ( I have given the range 172.168.0.0/24),

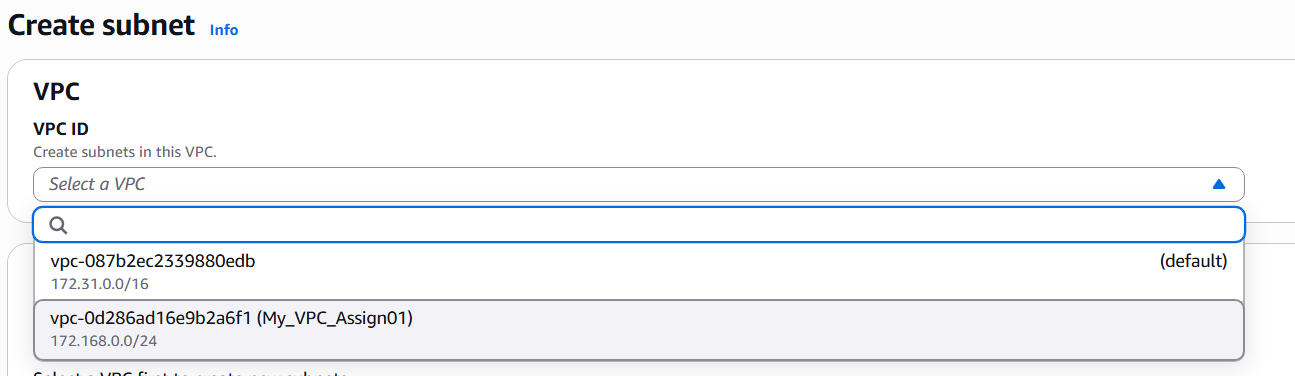


The following are the details of the VPC created

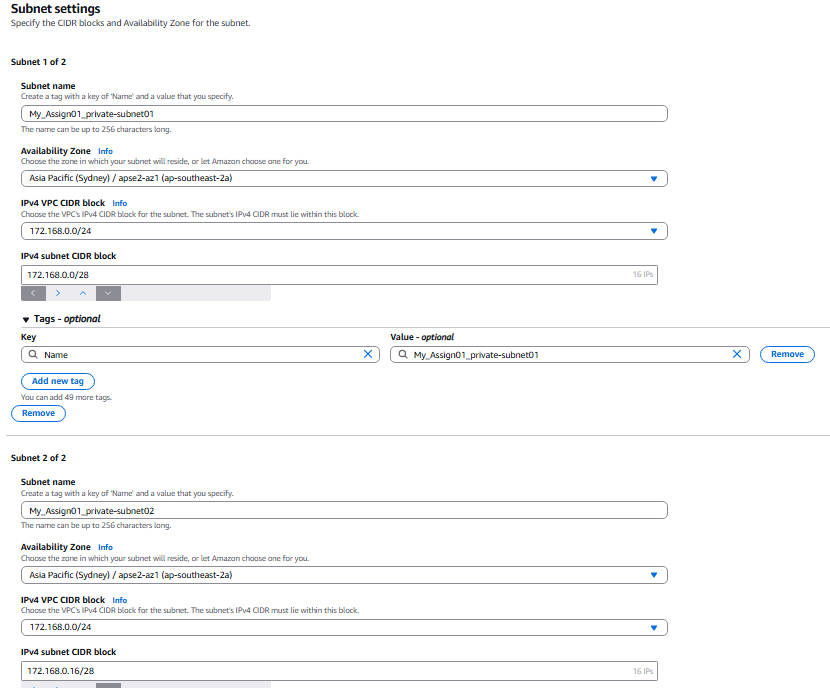
Now create the subnet as follows:

Go to subnets at the left side and create subnet

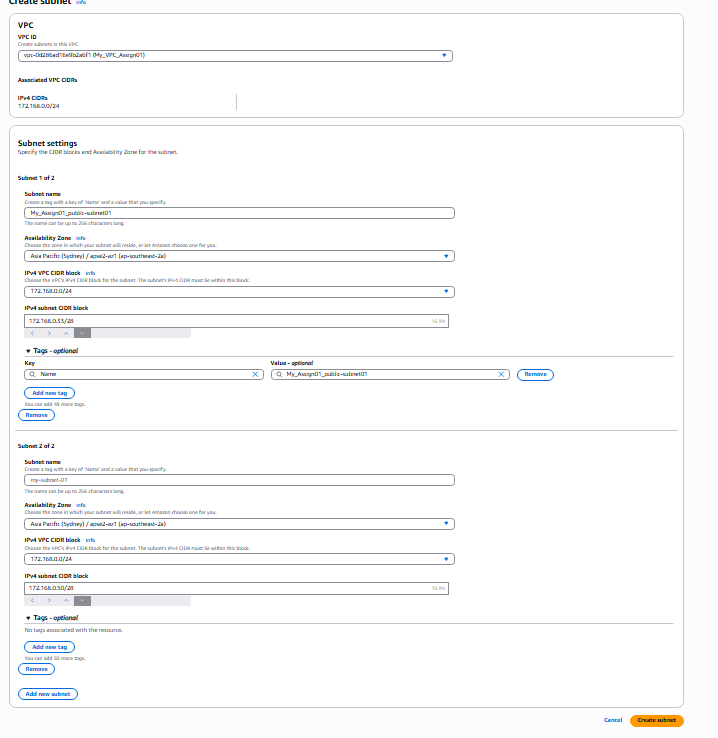
Select VPC ID ( my -VPC-Assign01) 



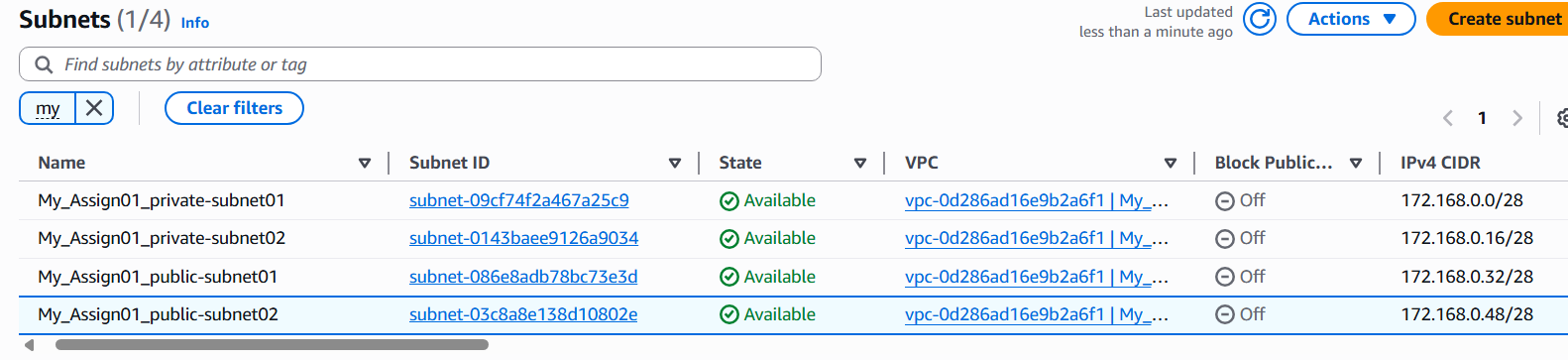
Create subnet by selecting the vpc we have created



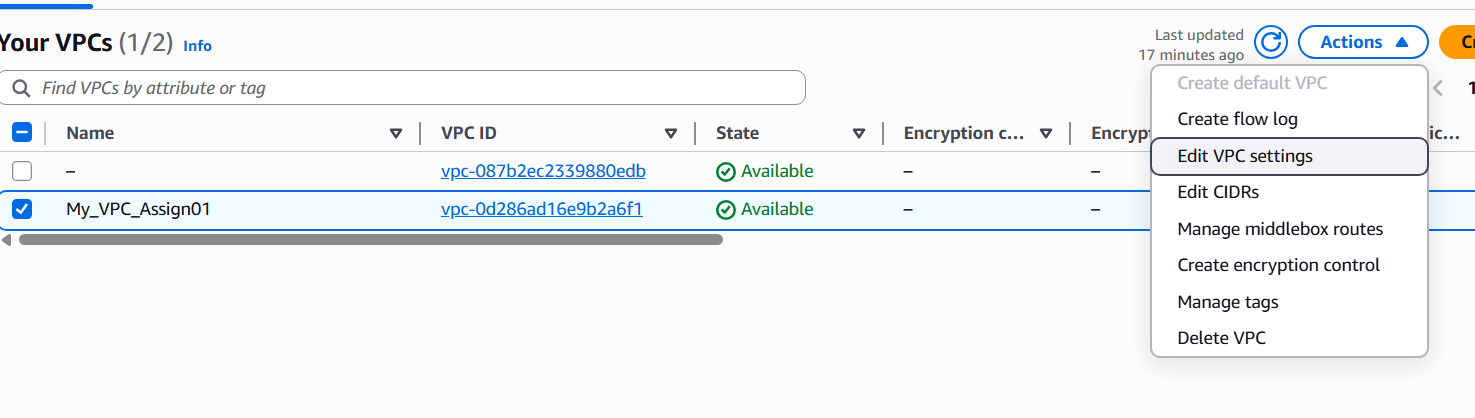
Define the two subnets Ie.. private subnets namely My\_Assign01\_private-subnet01 and My\_Assign01\_private-subnet02 with 172.168.0.0/28 and 172.168.0.16/28 respectively



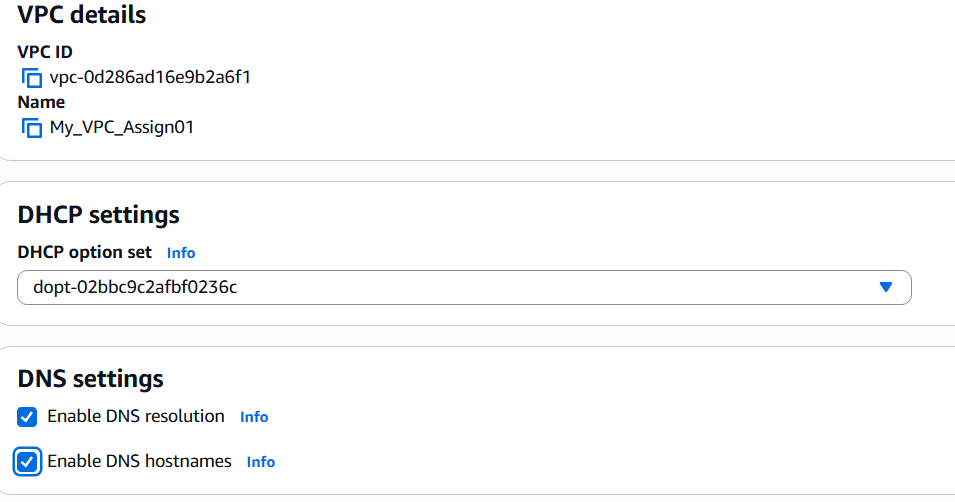
Similarly, Define the two subnets Ie.. public subnets namely My\_Assign01\_public-subnet01 and My\_Assign01\_private-subnet02 with 172.168.0.32/28 and 172.168.0.48/28 respectively



1. Enable DNS Hostname in VPC.

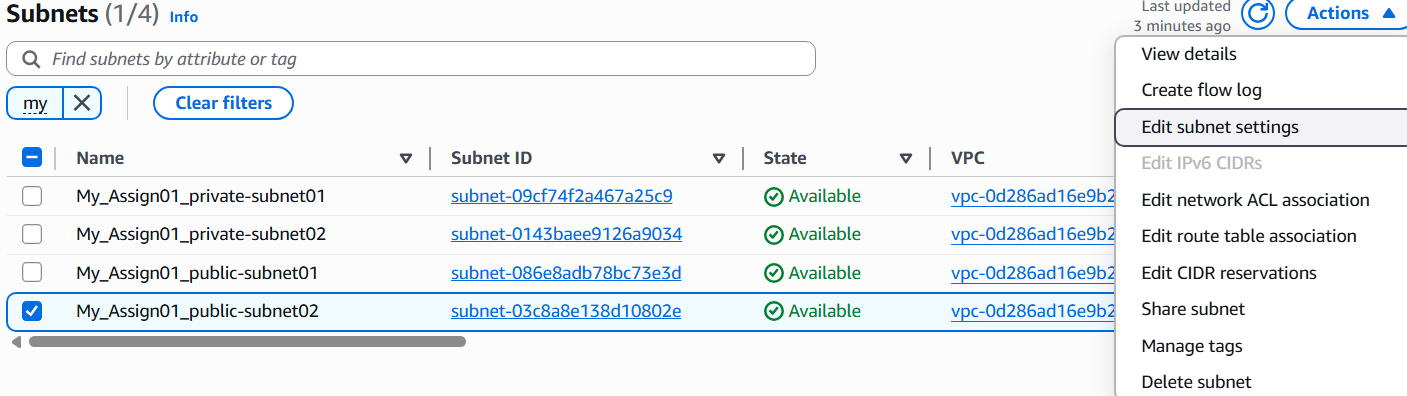


Go to actions in your vpcs, by selecting your vpc and select ‘Edit VPC Settings’

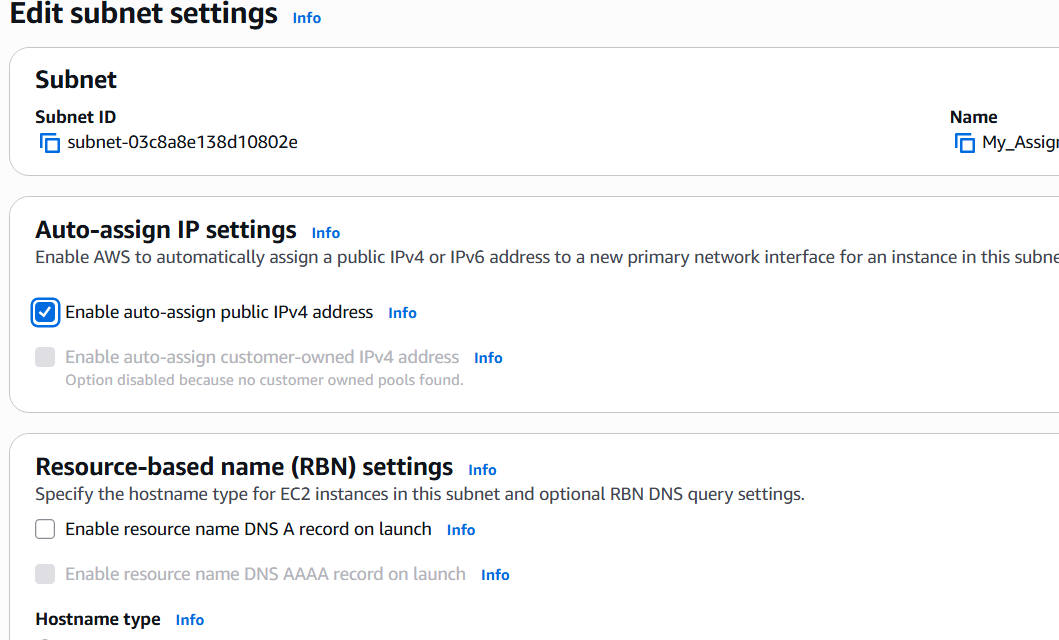


Toggle enable DNS hostnames to enable DNS hostname in VPC

1. Enable Auto Assign Public IP in 2 public subnets.



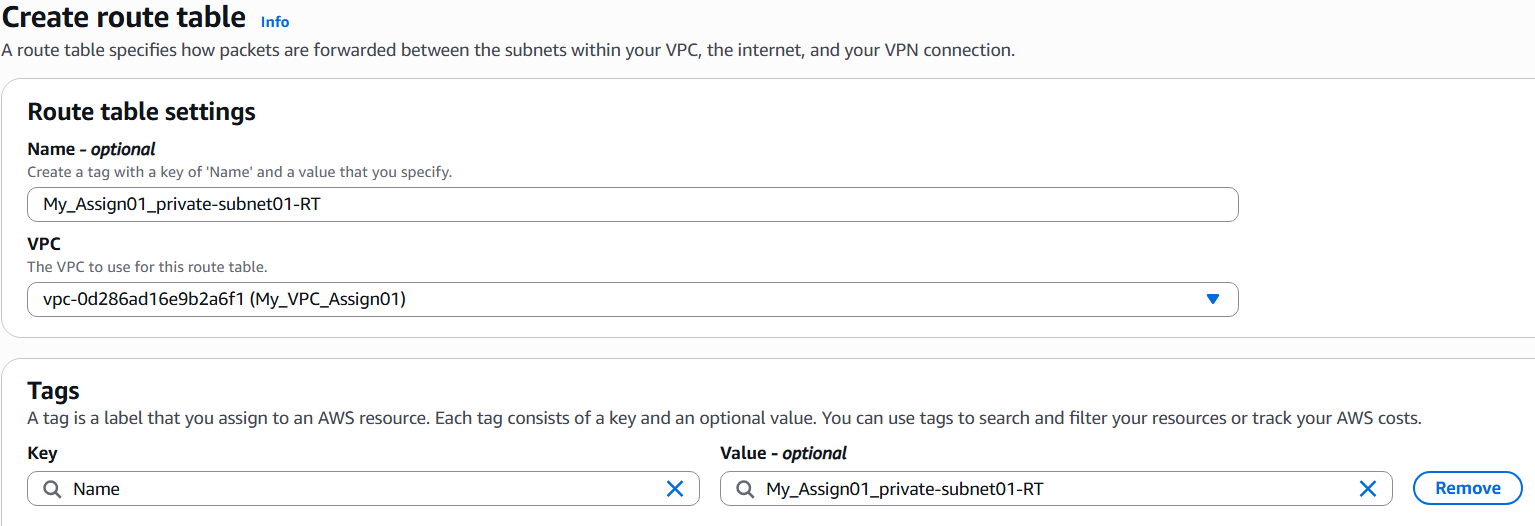
Go to subnets and select the public subnets created earlier, select ‘Edit subnet settings’ in the Action menu



Toggle ‘Enable auto-assign Public IPV4 address’

Repeat the same process for the public subnet created earlier and repeat the process to enable auto assign public ipv4 address.

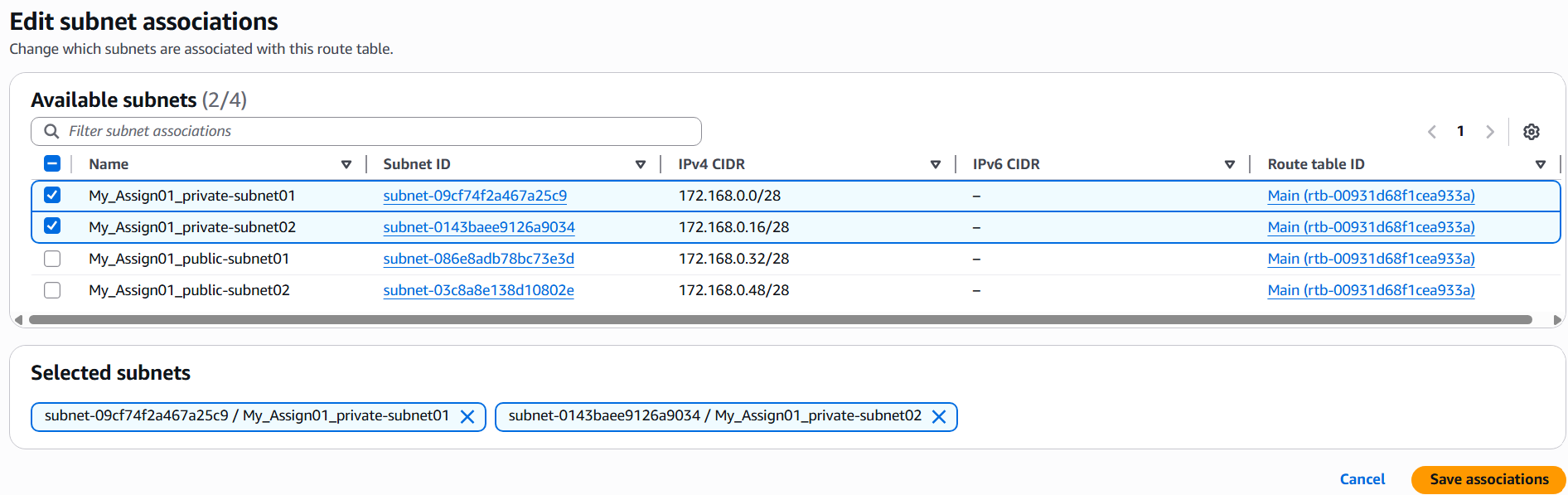
1. Add 2 private subnets in private route table.



Create a route table and give the name (My\_assign01\_private\_subnet01-RT) and select the vpc we created.

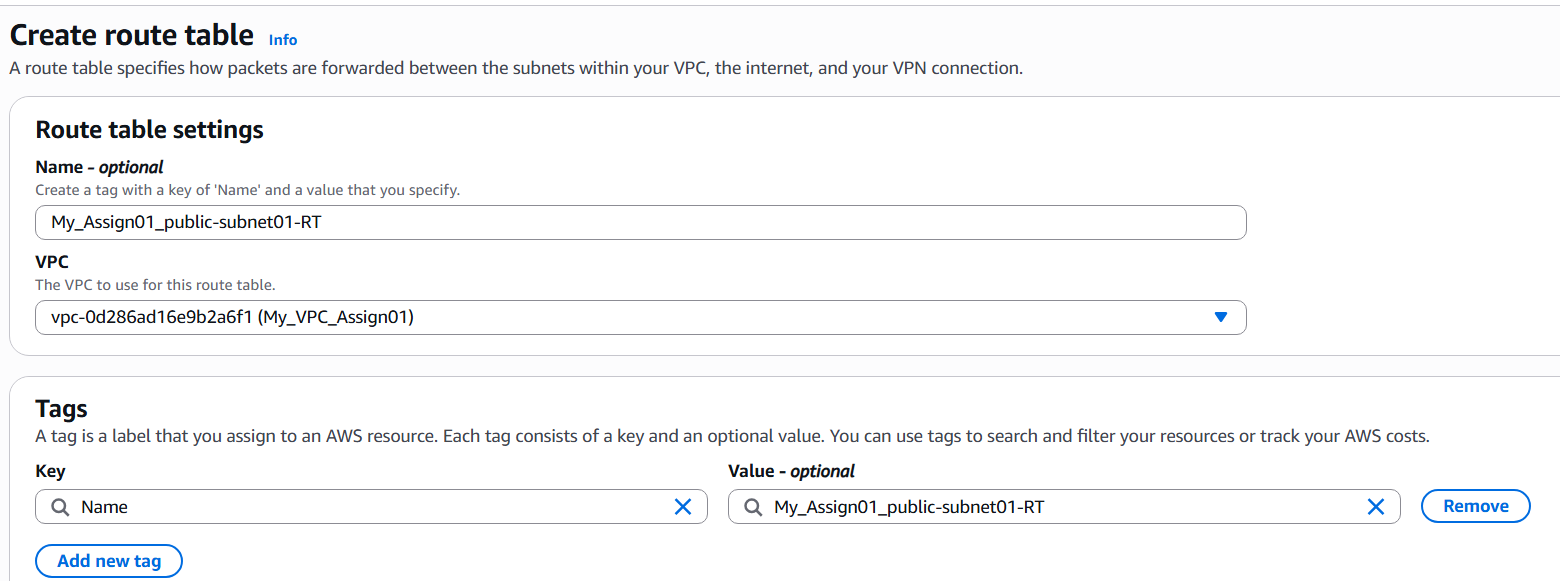


Associate the private subnet with the created routing table by pressing on Edit subnet associations

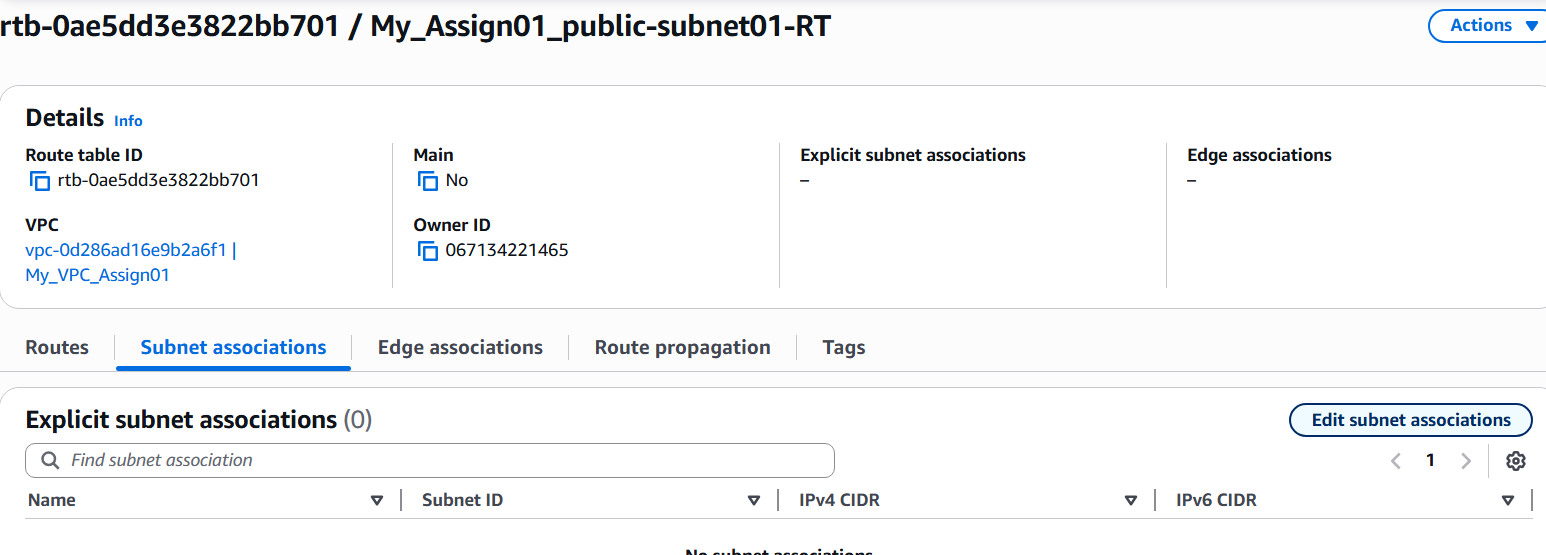


Select the available private subnets and save associations

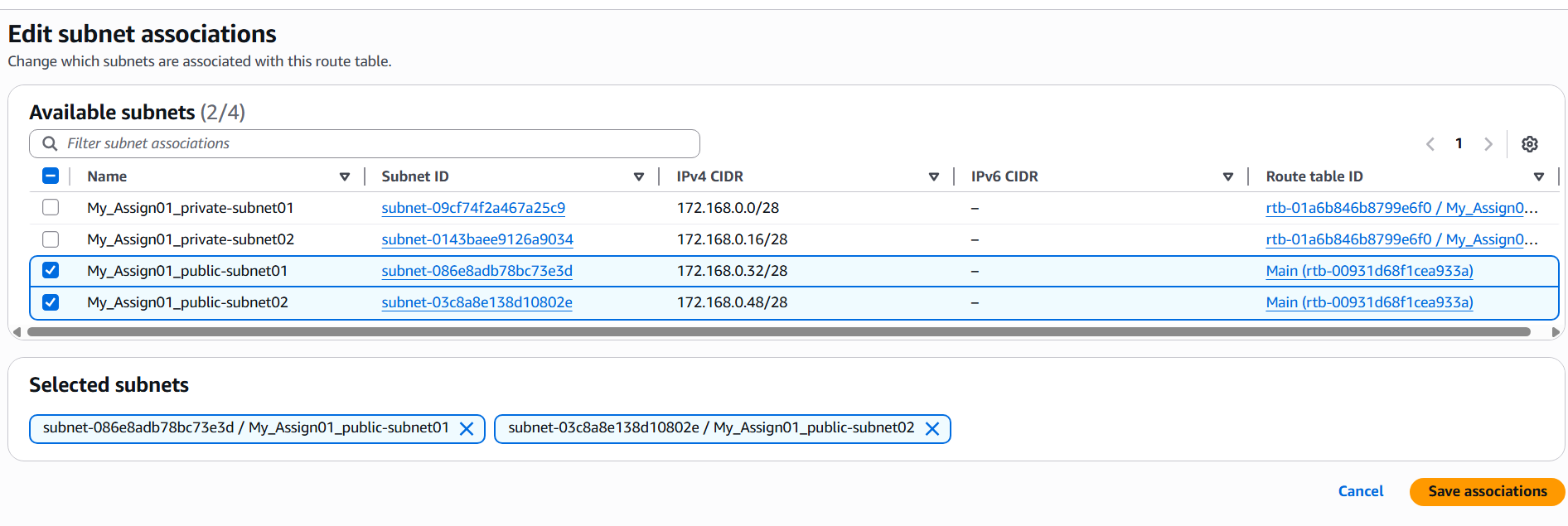
1. Add 2 public subnets in public route table.



Creae a route table with name (My\_Assign01\_public-subnet01-RT) and select the vpn created earlier

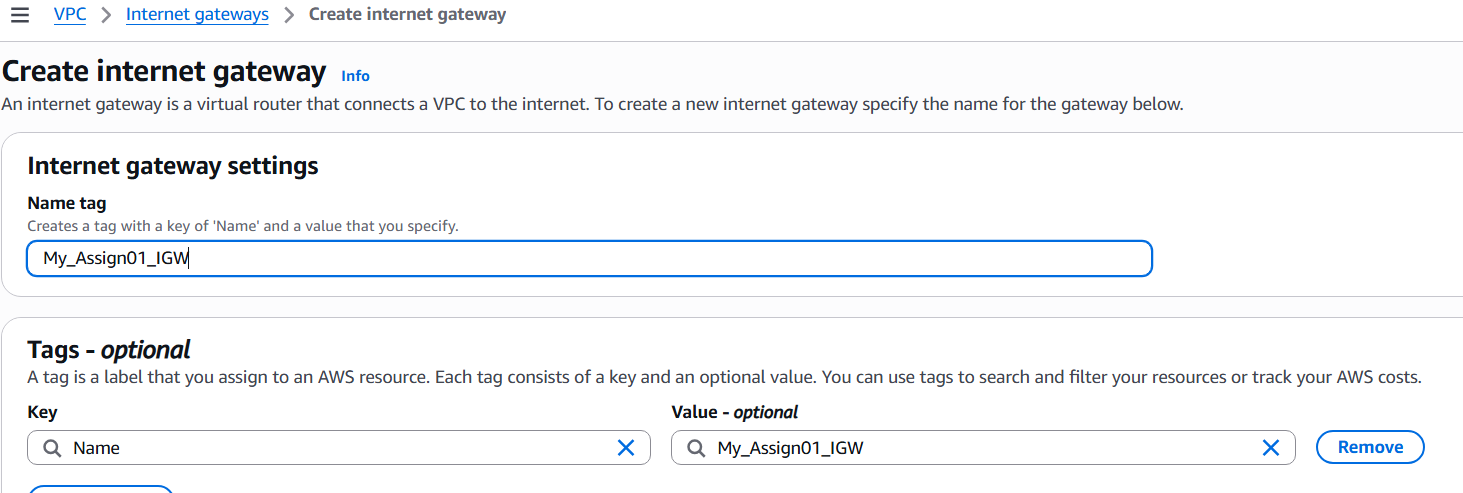


Make the subnet association of the public subnets with the newly created routing table.

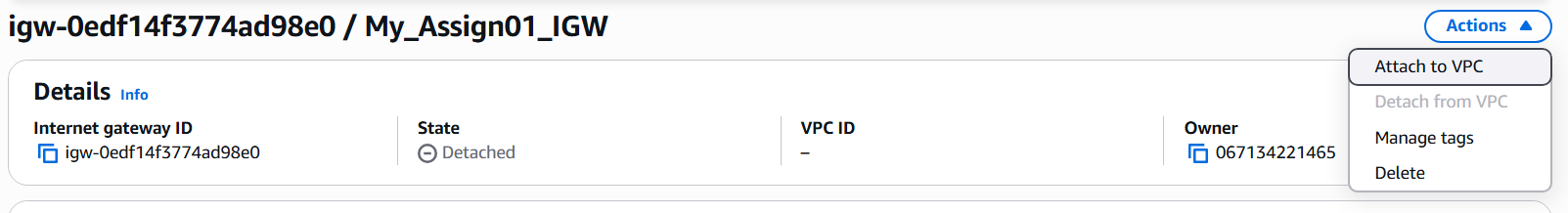


Select the public subnets and save associations.

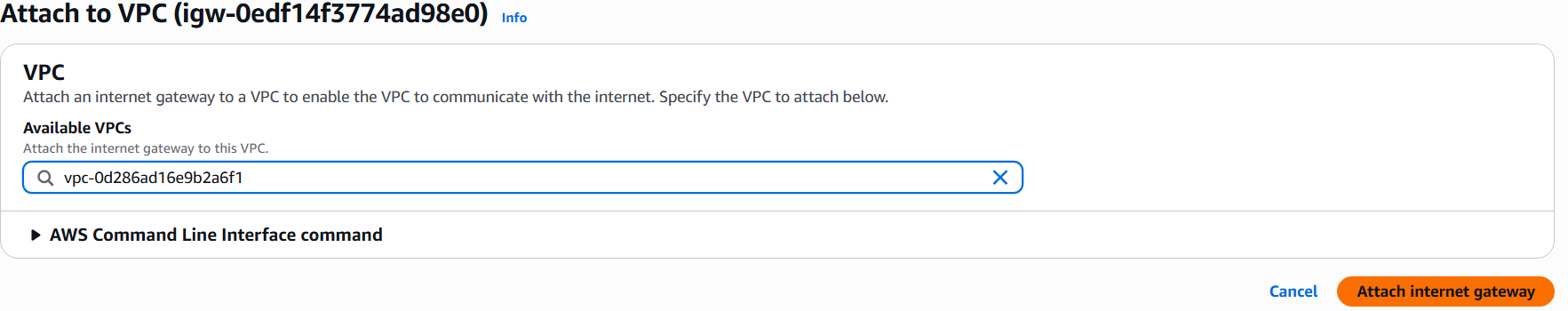
1. Public route table will have the routes to internet and local.

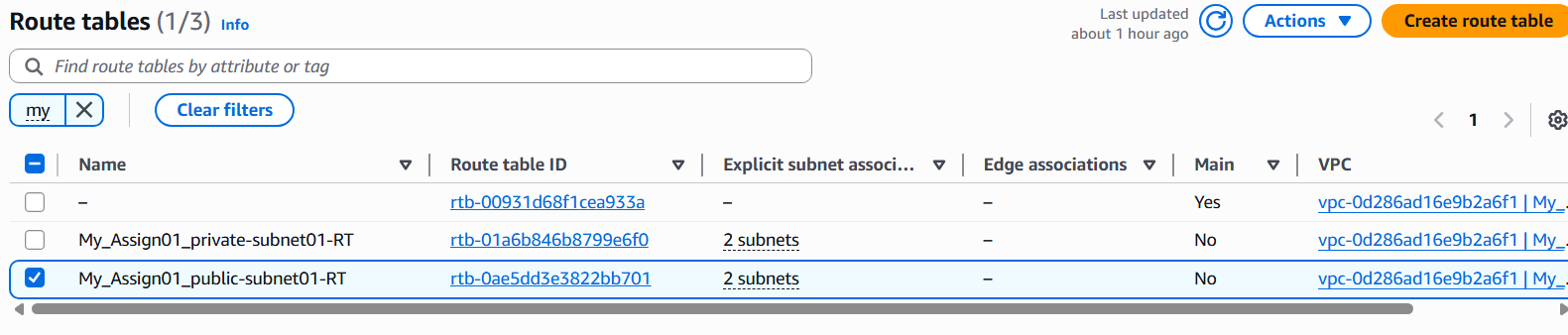


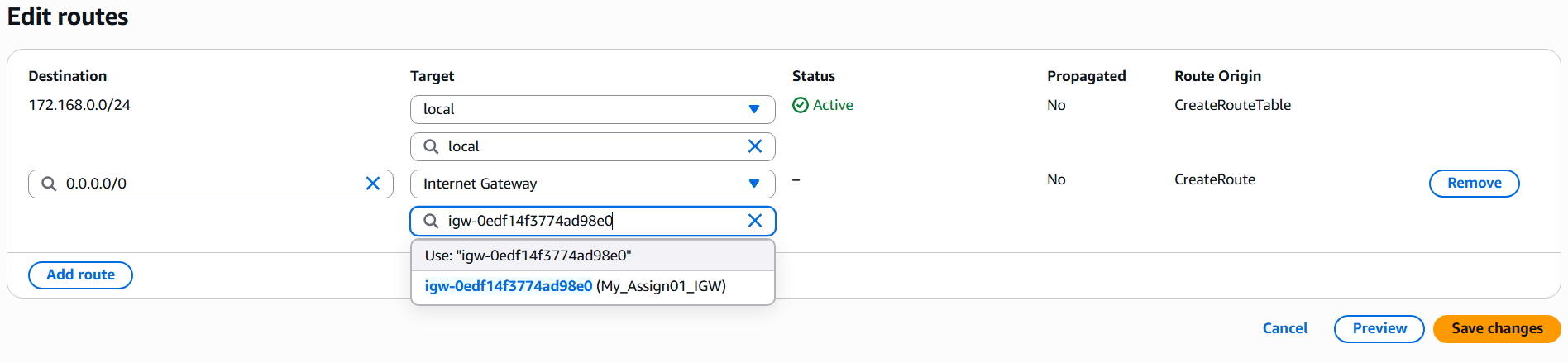
Create internet gateway by giving the name (My\_Assign01\_IGW)



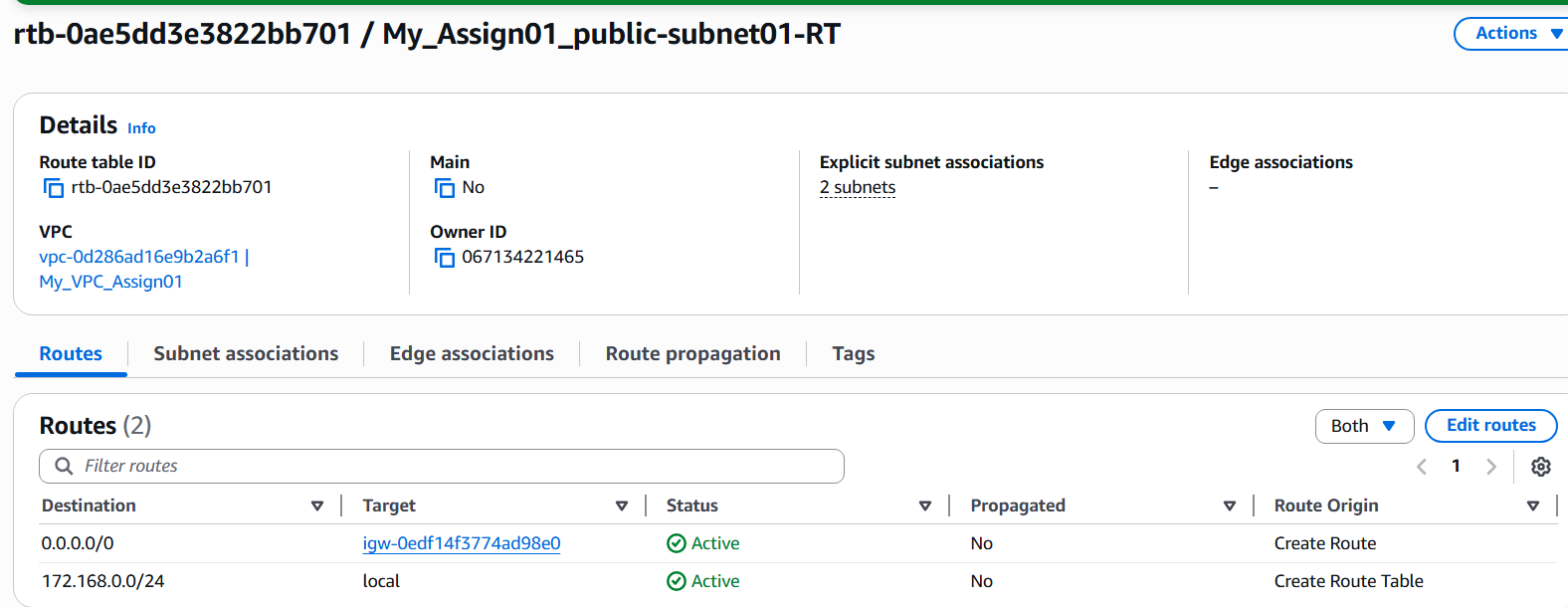
Go to actions and Attach to vpc …. Attach it the vpn





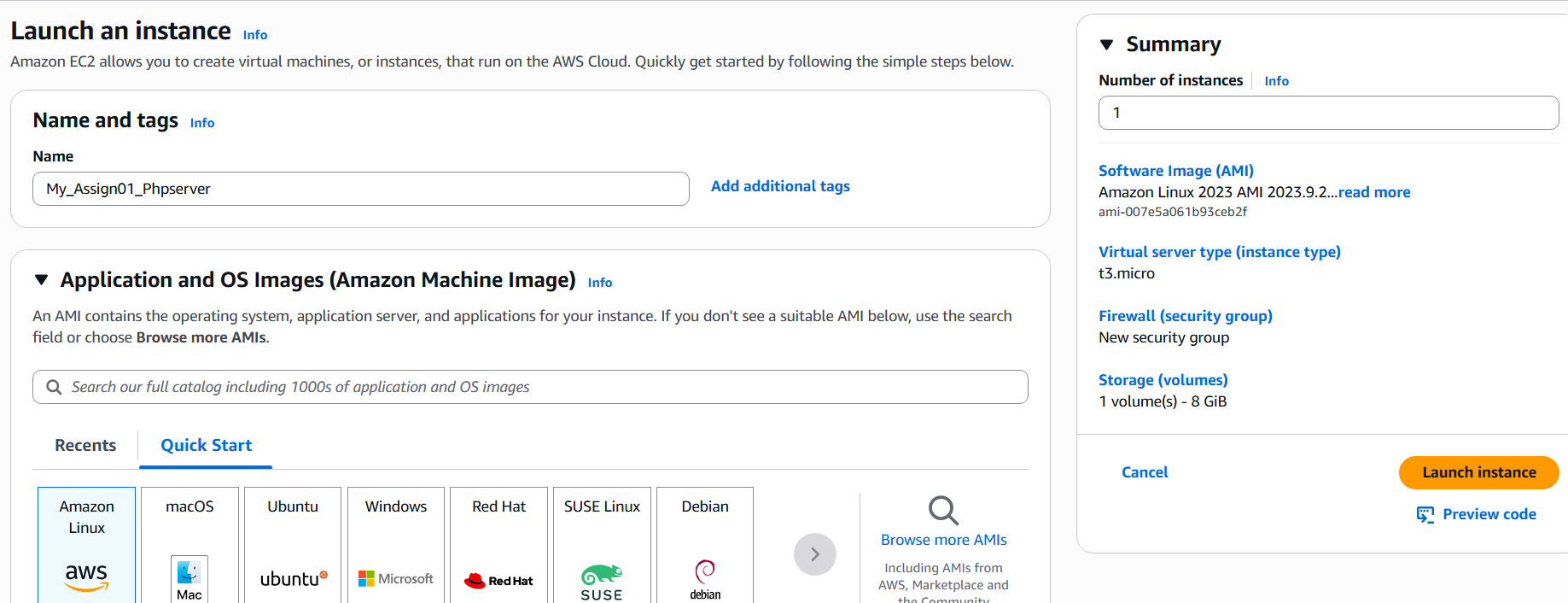


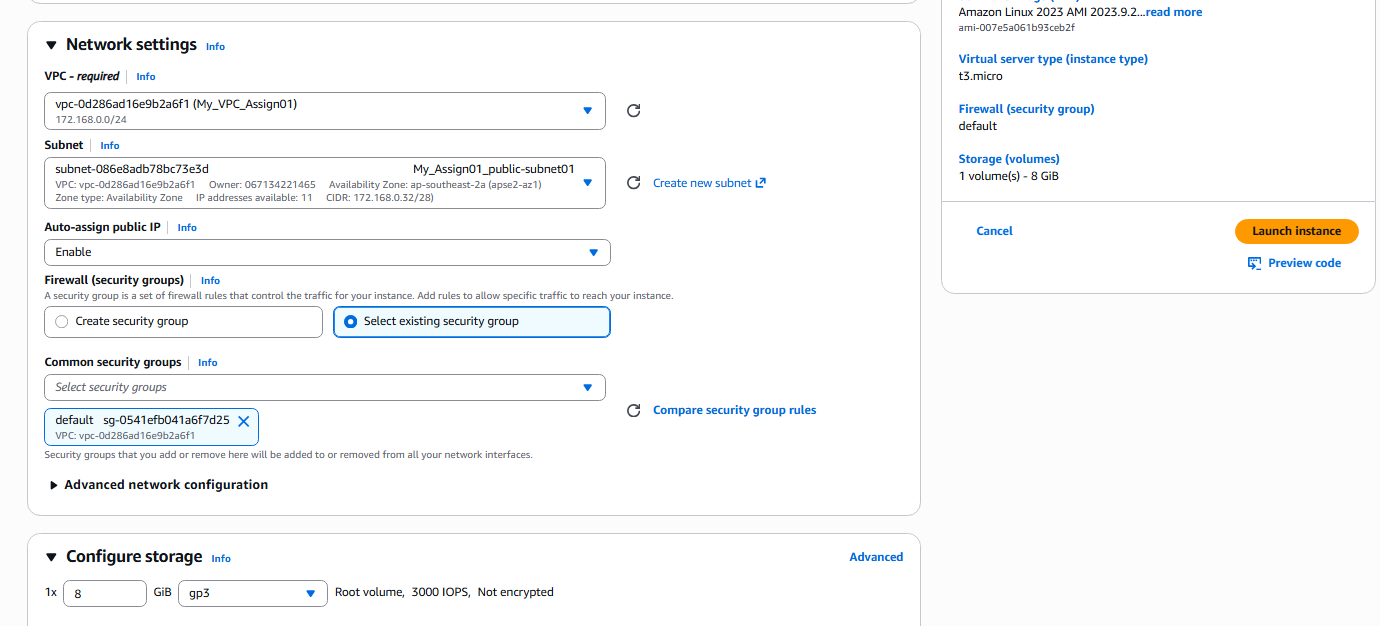
Edit routing table route in the public with Internet gateway with destination defined to 0.0.0.0



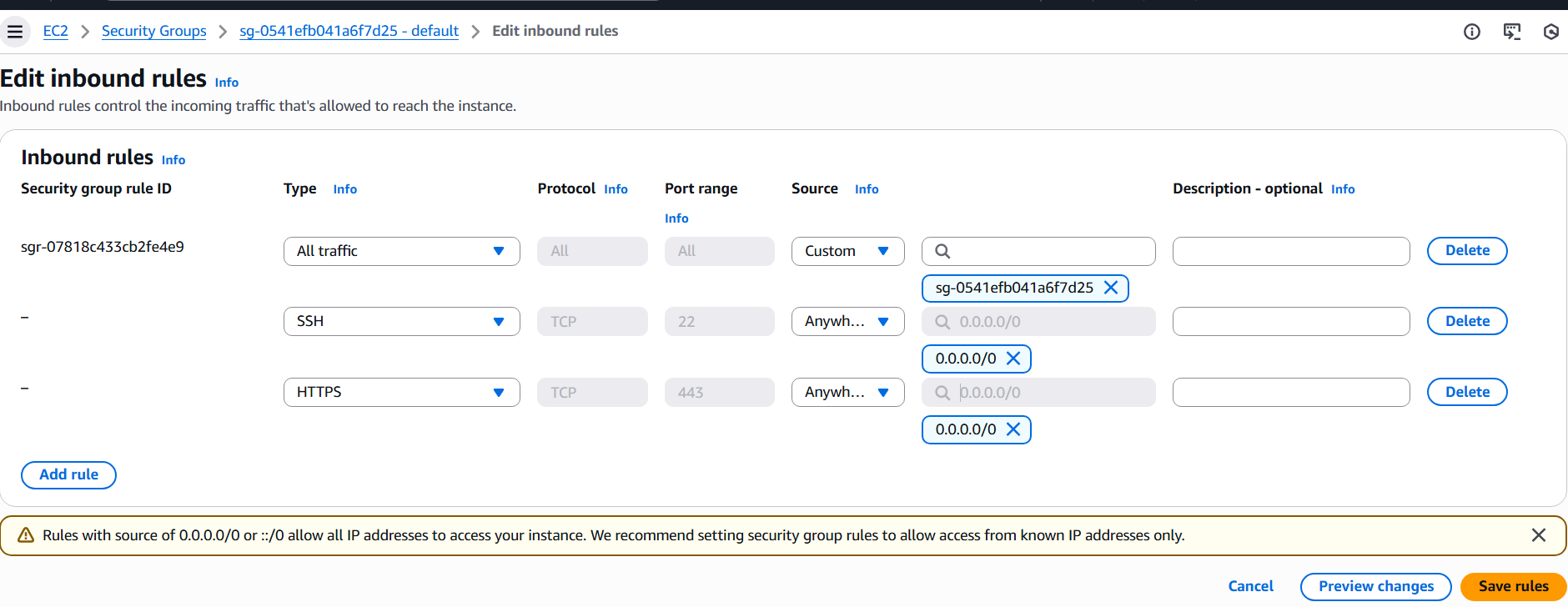
Now we can see the routes with local as well as the IGW

1. Create EC2 in public subnet with t2.micro and install PHP.

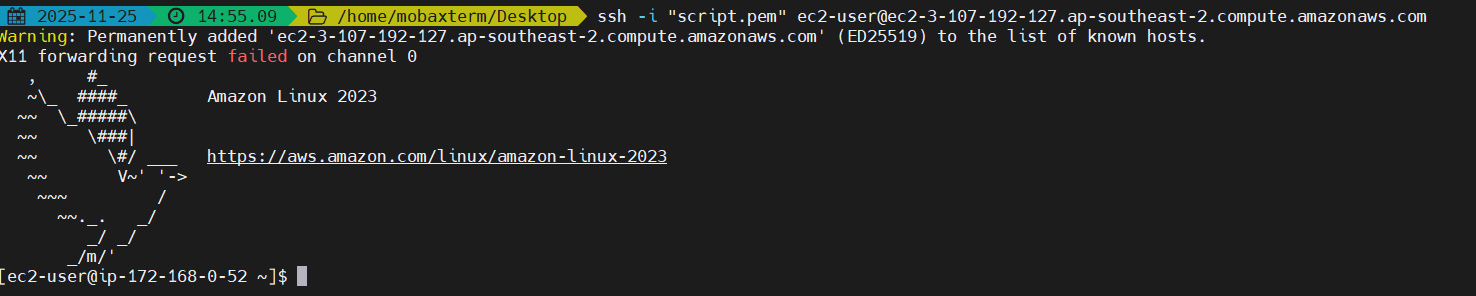


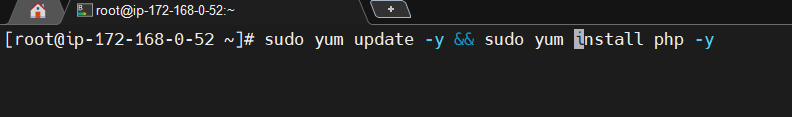


Create a EC2 Linux instance and select t3.micro instance type, go to the network setting select the vpn, public subnet and also the default vpn security group.

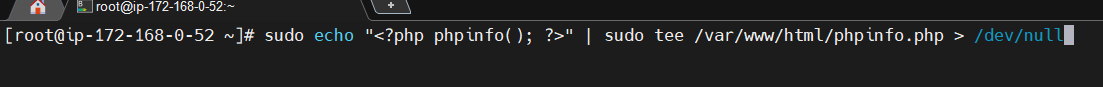


Edit the security group attached with SSH, HTTP, Http enabled.

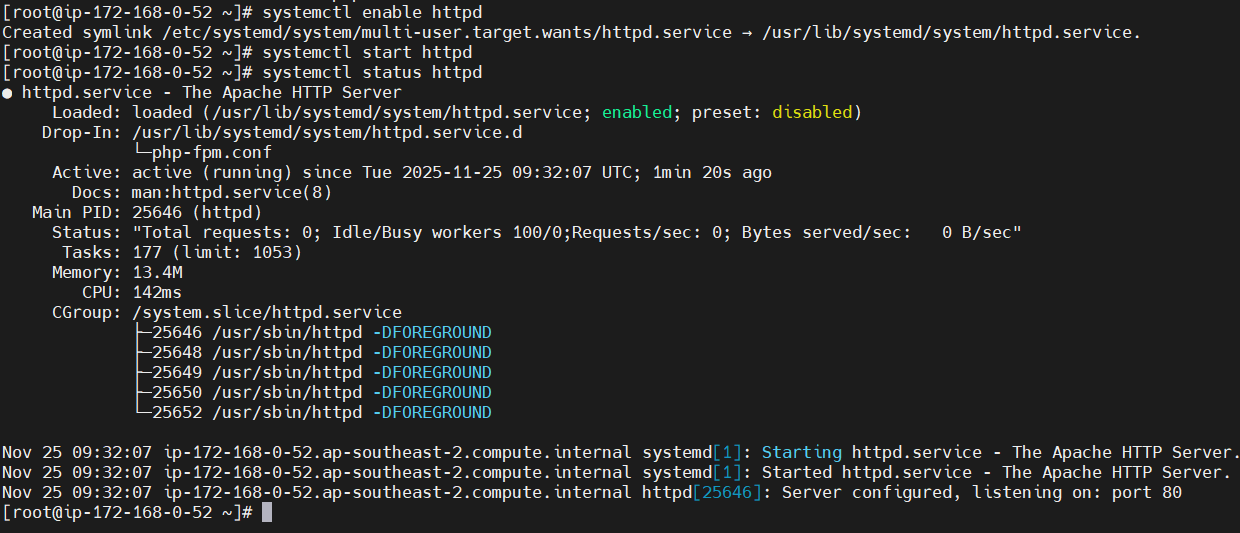




Logon to the EC2 insane and update the instance and also install php as shown in the above.



The above command securely creates a file named phpinfo.php in the web server's document root, containing the PHP code necessary to display a phpinfo() page, while suppressing any terminal output from the operation

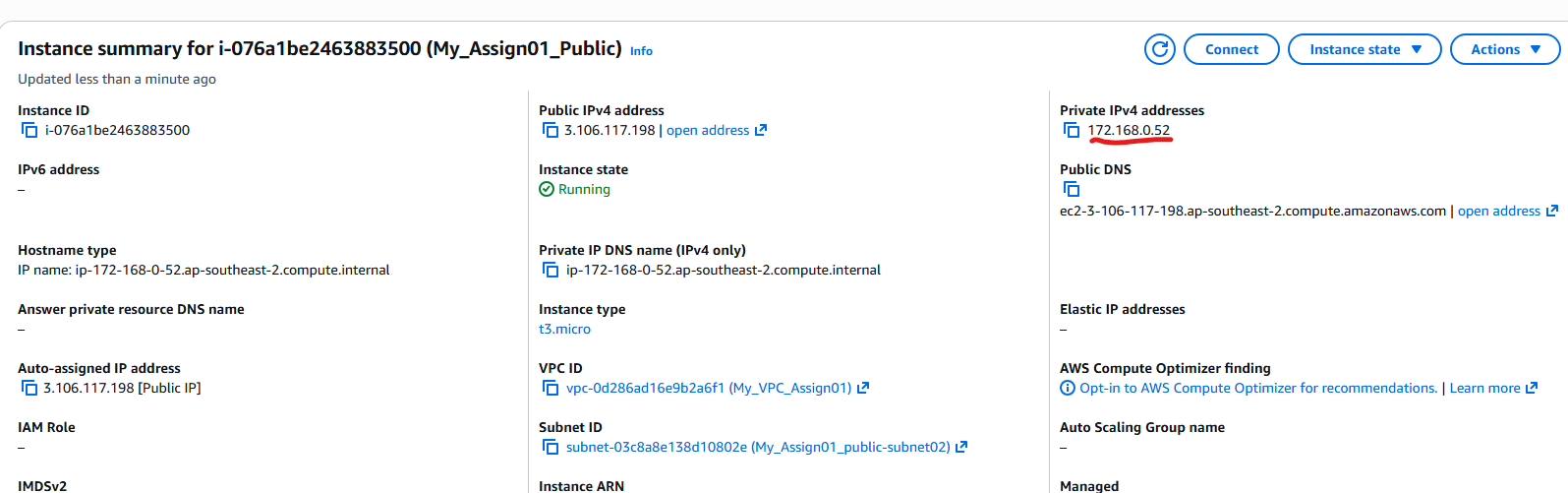


Start the httpd server and check the status if it is working.

Go to <public\_address>/phinfo.php to access the php default page.



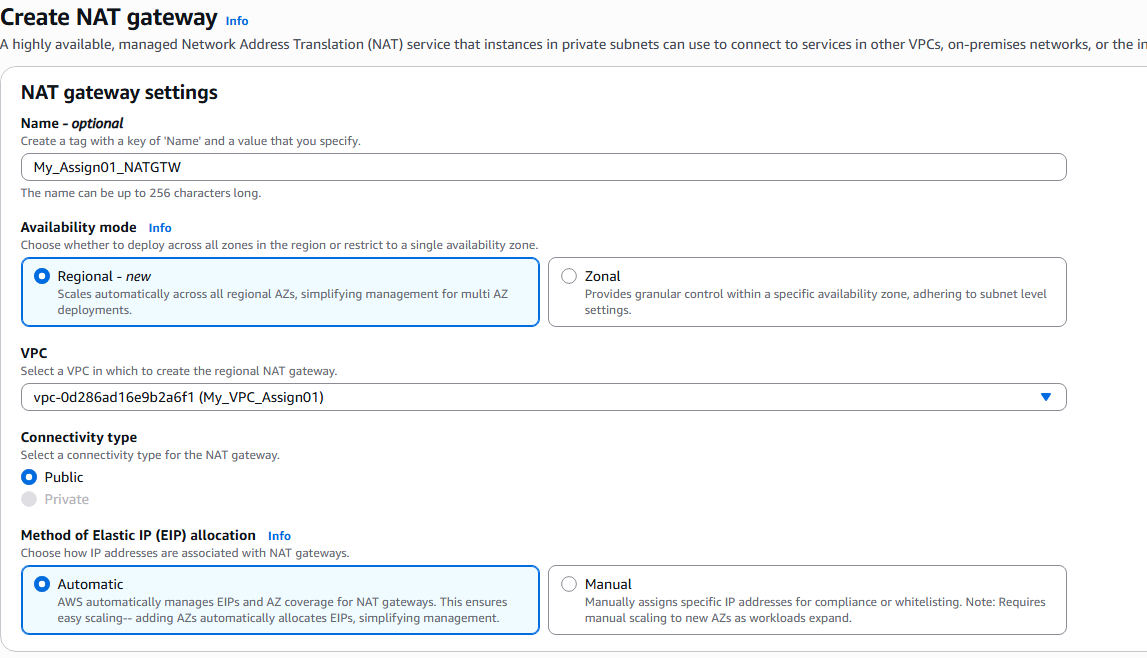
1. Configure NAT gateway in public subnet and connect to private instance.



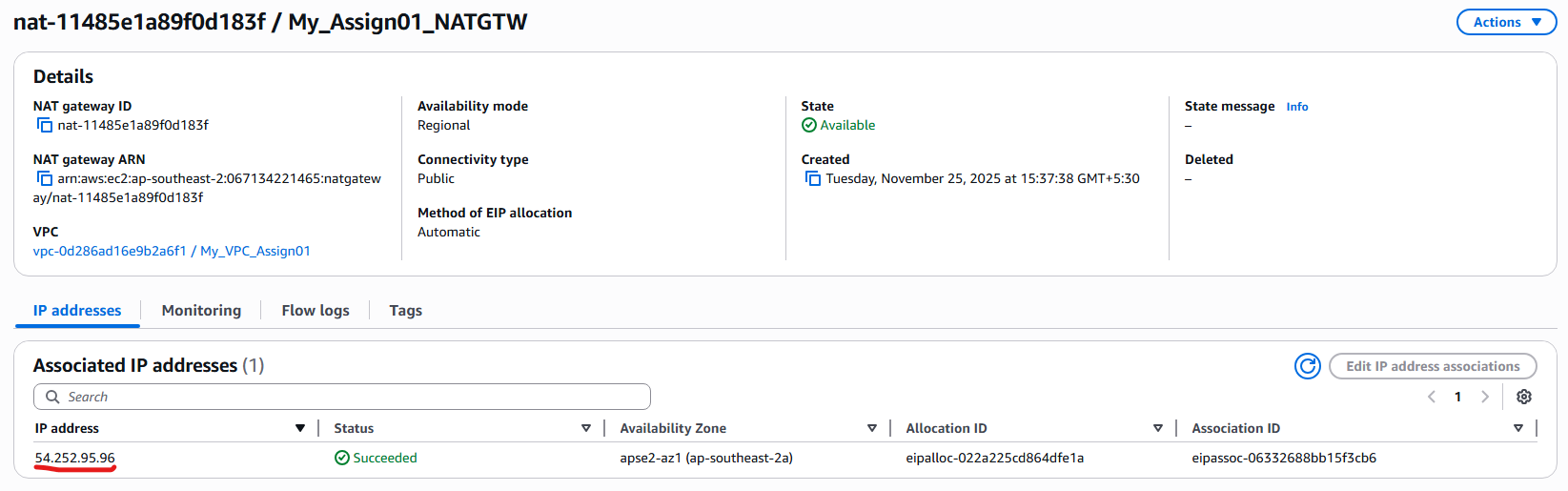
Create the instance in the public subnet by selecting the vpn and also the public subnet as shown in the above.



Create the instance in the private subnet by selecting the vpn and also the private subnet as shown in the above

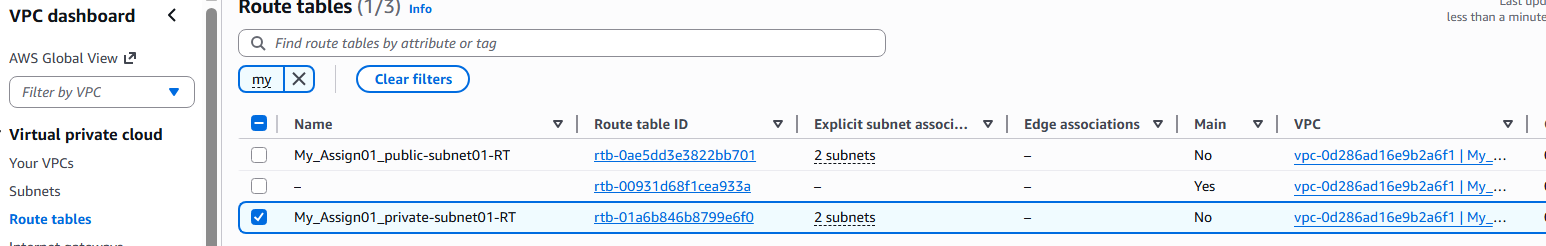


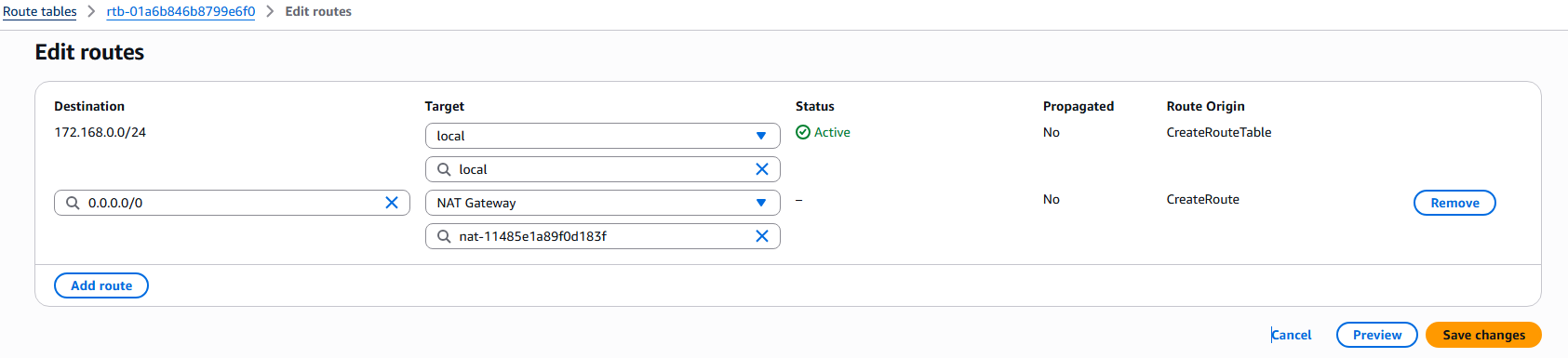
Create the NAT gate, give name and select the VPC

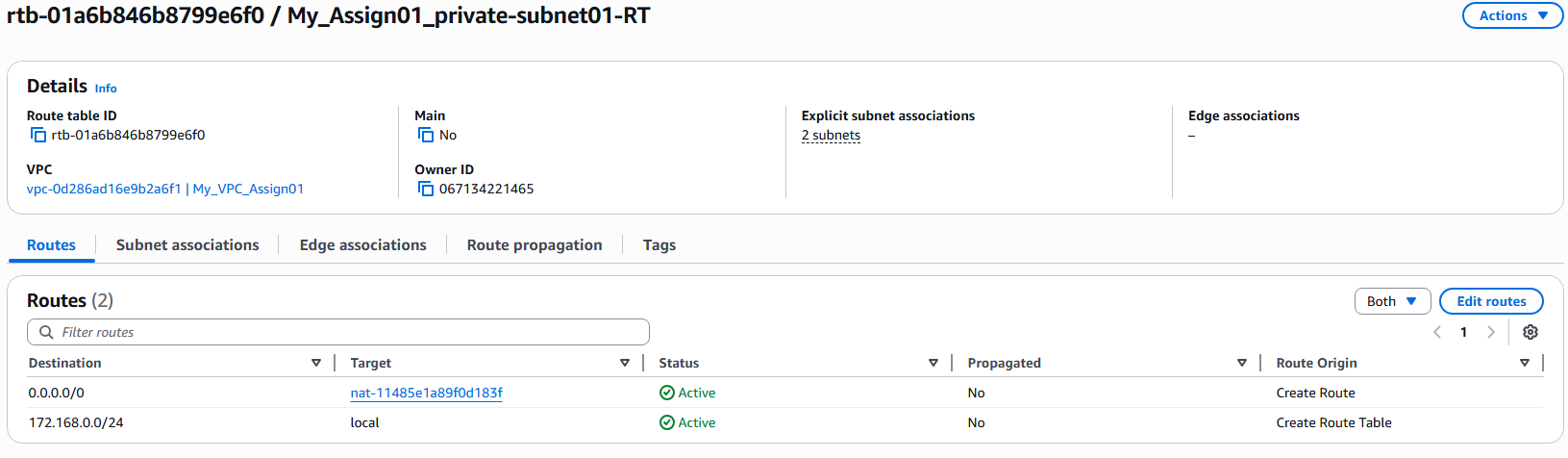


We need to deploy the NAT Gateway in the public subnet inorder for the instances in the private to access the internet (52.252.95.96)

So go to the private subnets and edit the routes to the NAT Gateway with the destination address defines to 0.0.0.0



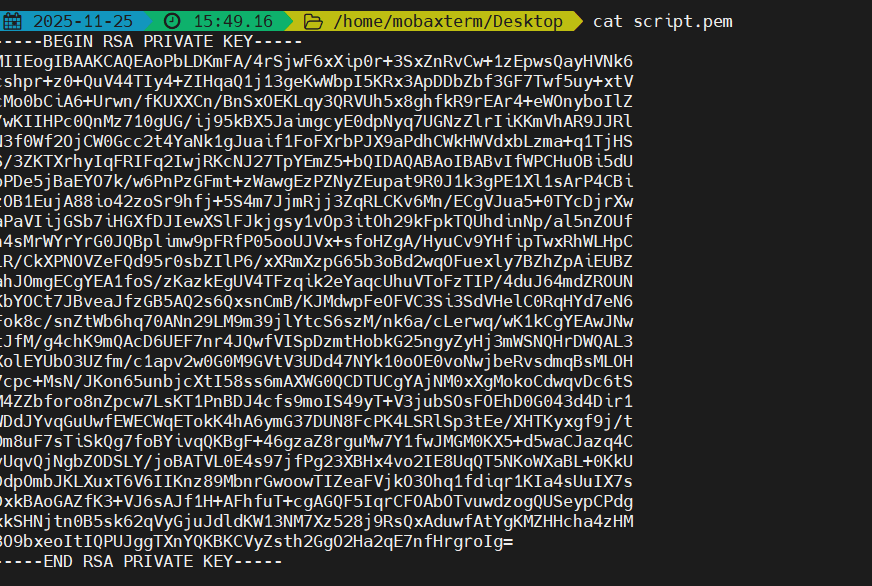


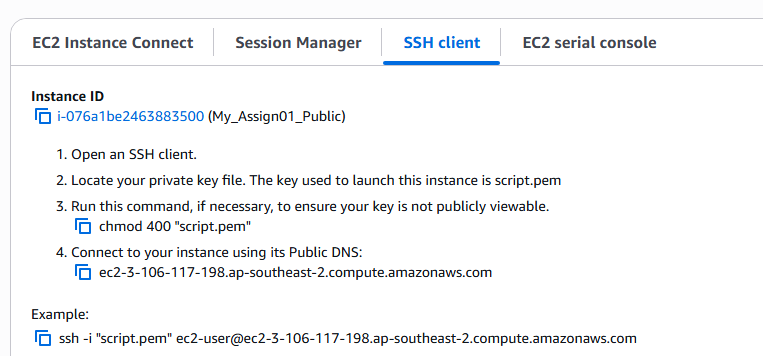


Check the routing table to see if the NAT is configured

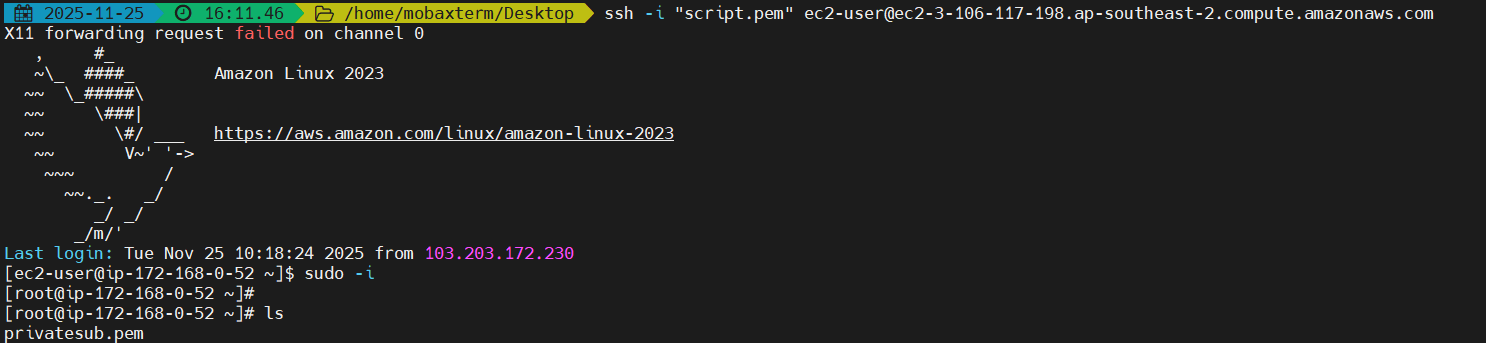
Inorder for the access to the instances in the private server, we need to access the instances in the public server to jump server to the instances.

So to do that first copy the pem file contents of the instance in private server



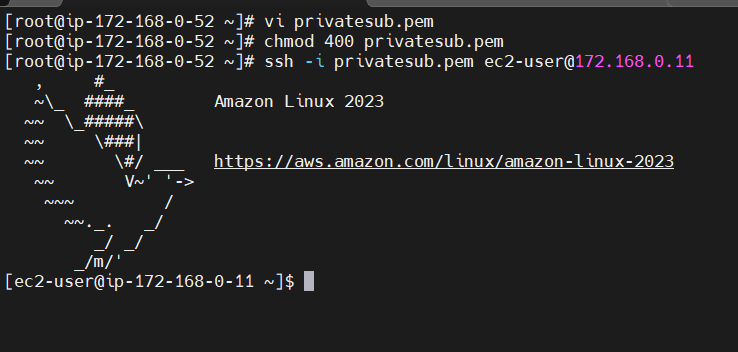


Logon to the public instances and create a pem file( privatesub.pem) , paste the content of the pem file and save the file



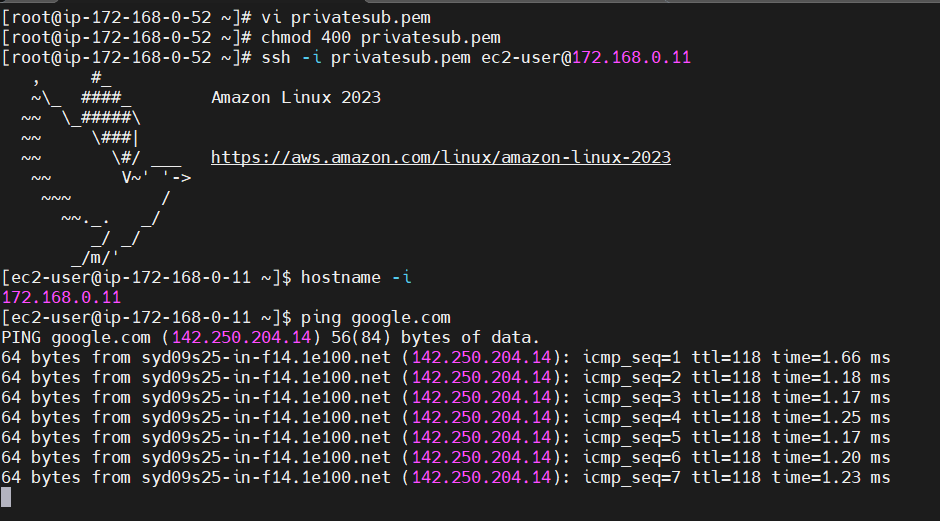
Give readonly permissions to the pem file before accessing the instance

Type “ ssh -i <pemfile> ec2-user@<Private\_ip\_address> (172.168.0.11)

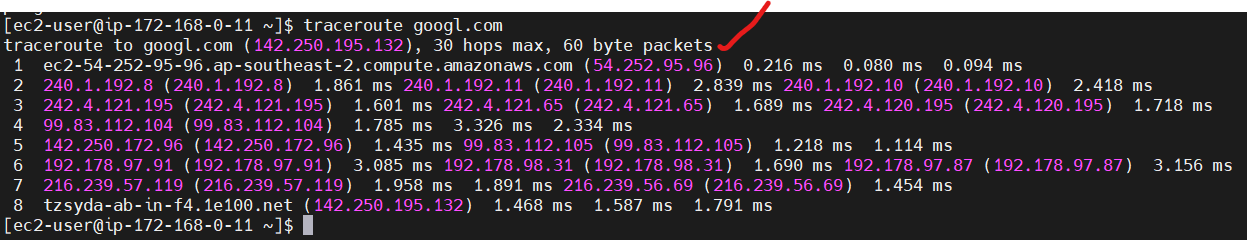


After successful logon to the private instance

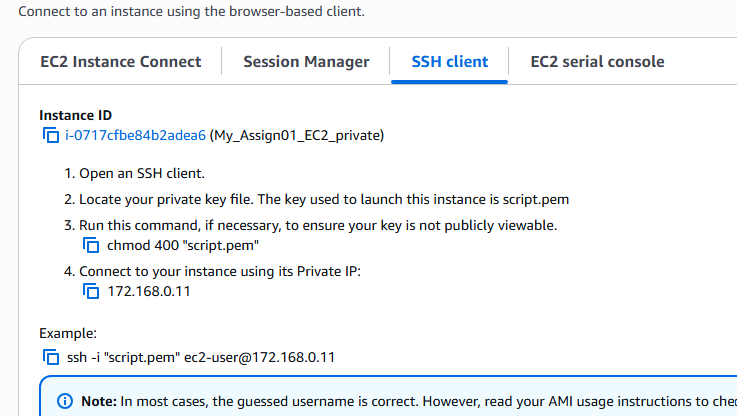
Initiate ping command to check if we are accessing the internet.

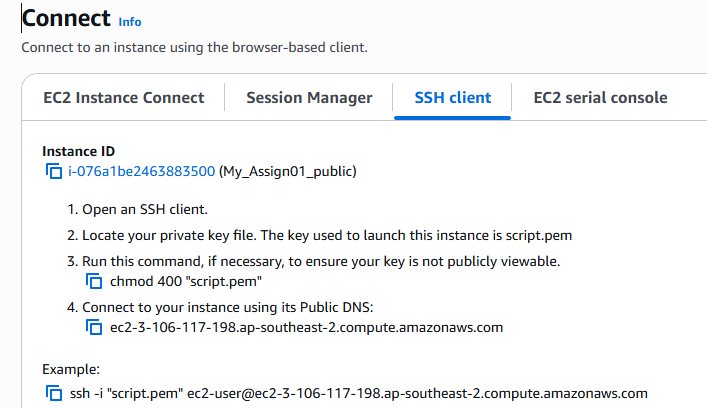


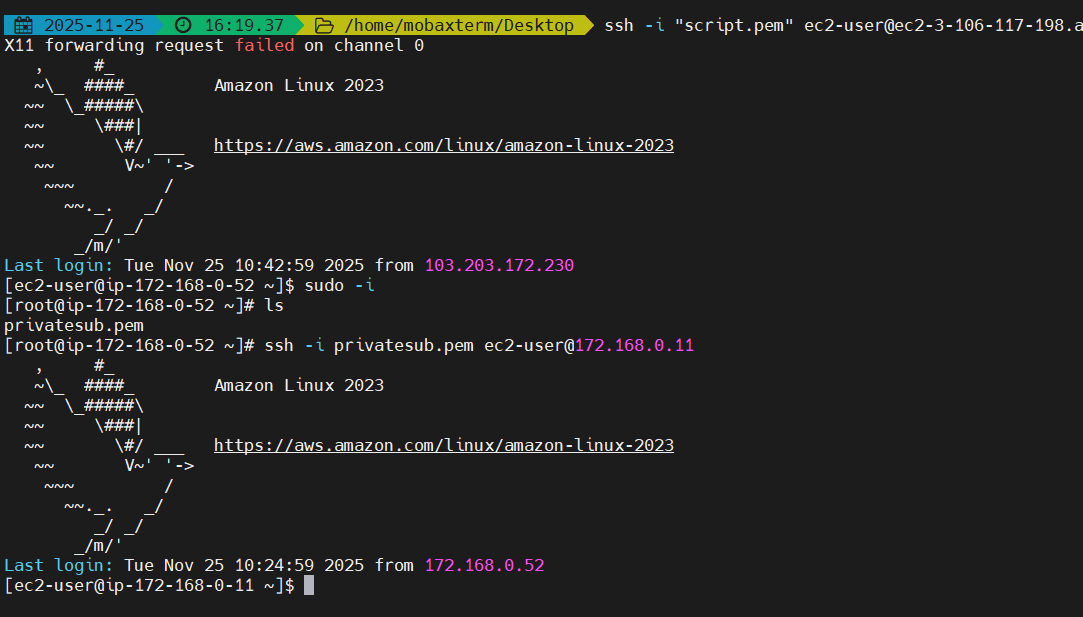
The Traceroute command shows the routing is happening from the NAT gateway we defined in th public subnet (ie..54.252.95.96)



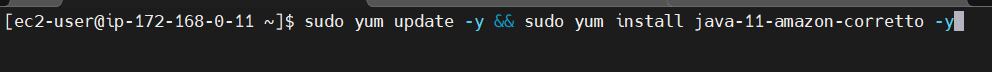
1. Install Apache Tomcat in private EC2 and deploy a sample app.



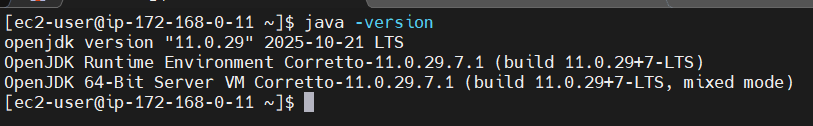




Logon to the public subnet instance which will act like a jump server, ssh to the private subnet instance by ssh connection as we did the process in the previous question



Install java-11 which is prerequisite for the apache tomacat 9



Verify the version by executing command ‘java -version’



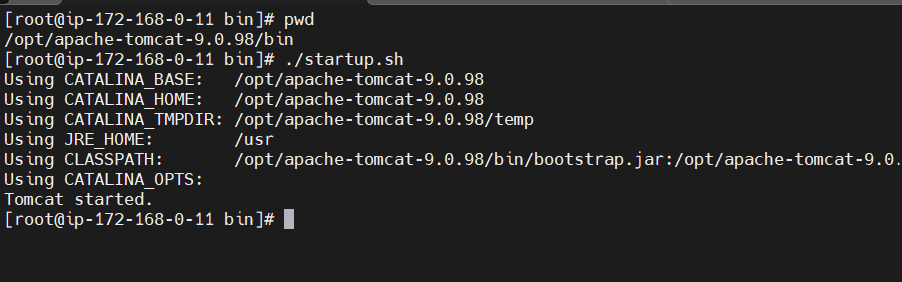
Move to folder /opt and download the official tomcat binaries from the apache tomcat official site

Cd /opt/

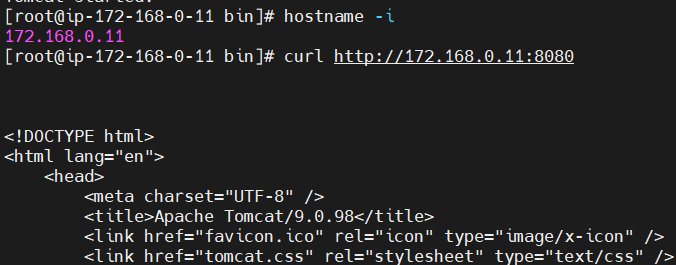
Sudo Wget <https://archive.apache.org/dist/tomcat/tomcat-9/v9.0.98/bin/apache-tomcat-9.0.98.tar.gz>



Extract the contents using the tar xvf apache-tomcat-9.0.98.tar.gz

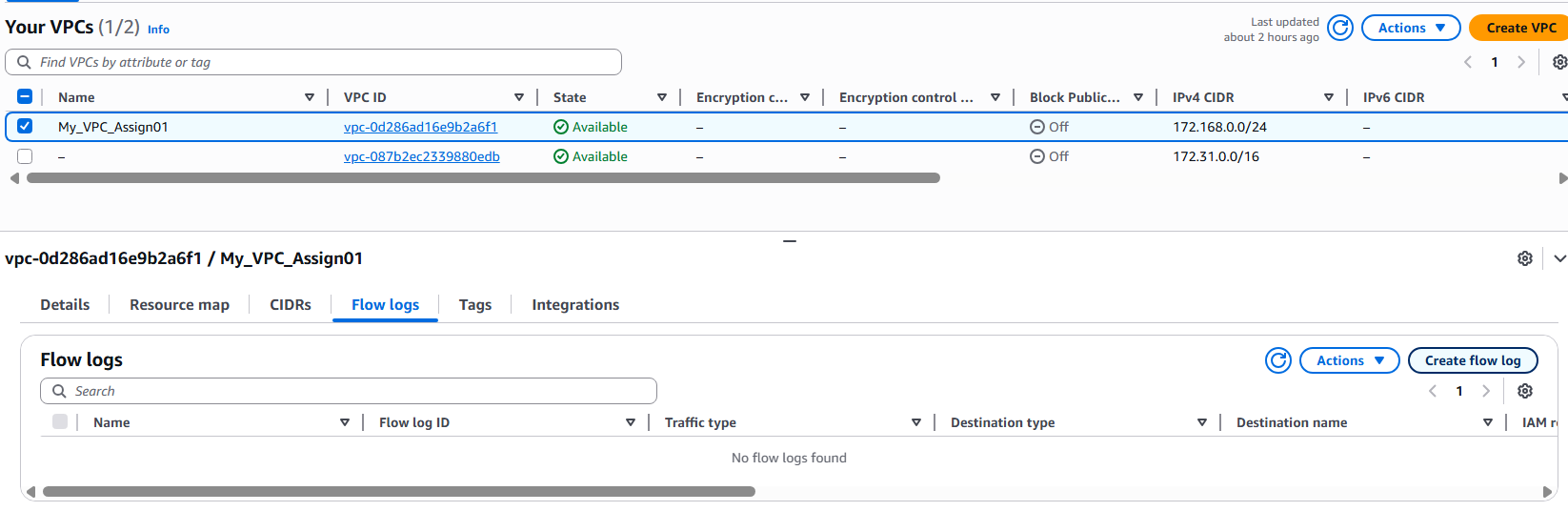


Go to folder bin and run the script ./start.sh

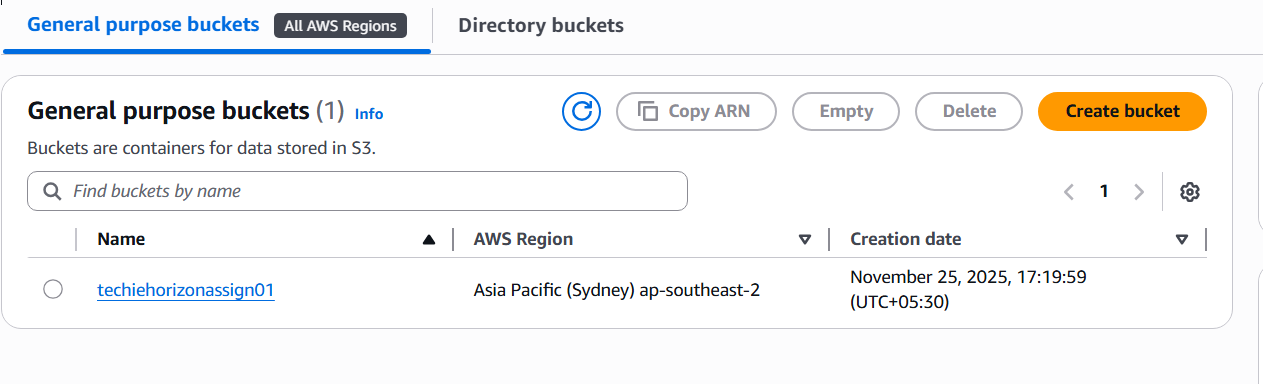


Curl to the private ip address to check if the raw content is displayed, in case if it is not successful we will see 404 error(resource not found).

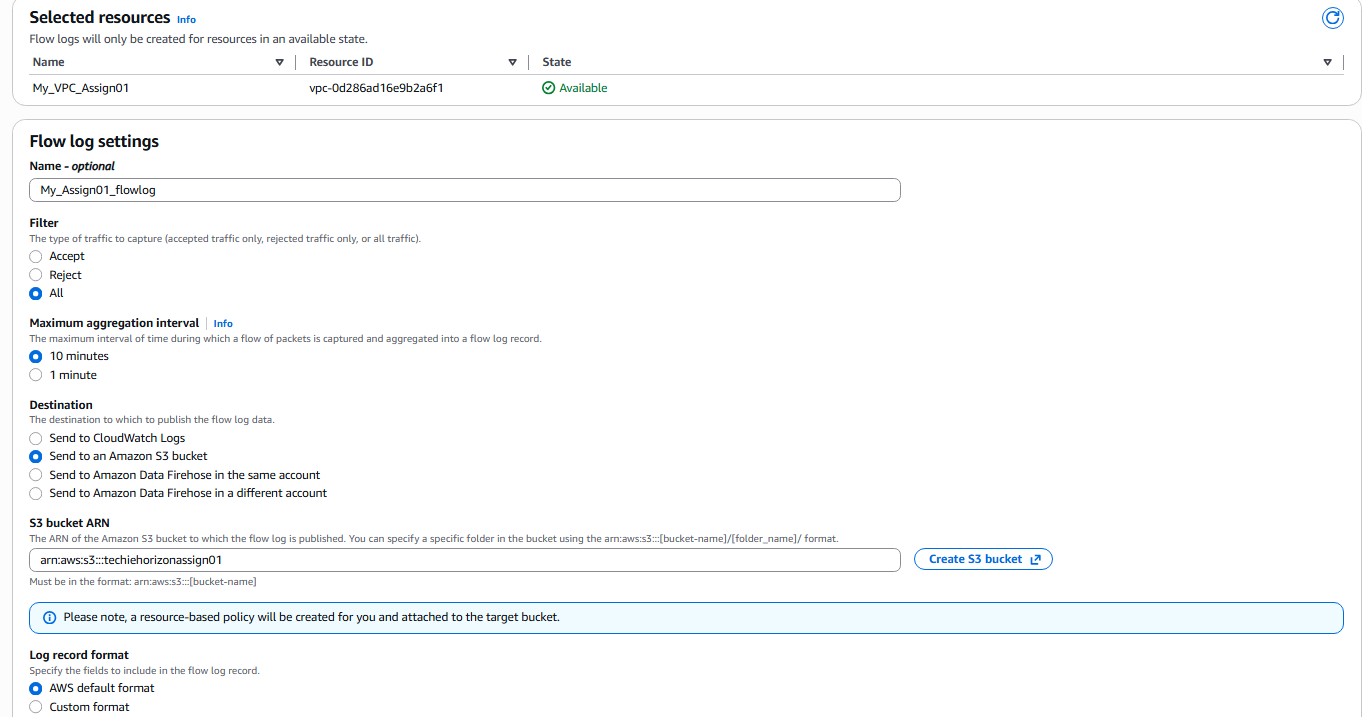
1. Configure VPC flow logs and store the logs in S3 and CloudWatch.



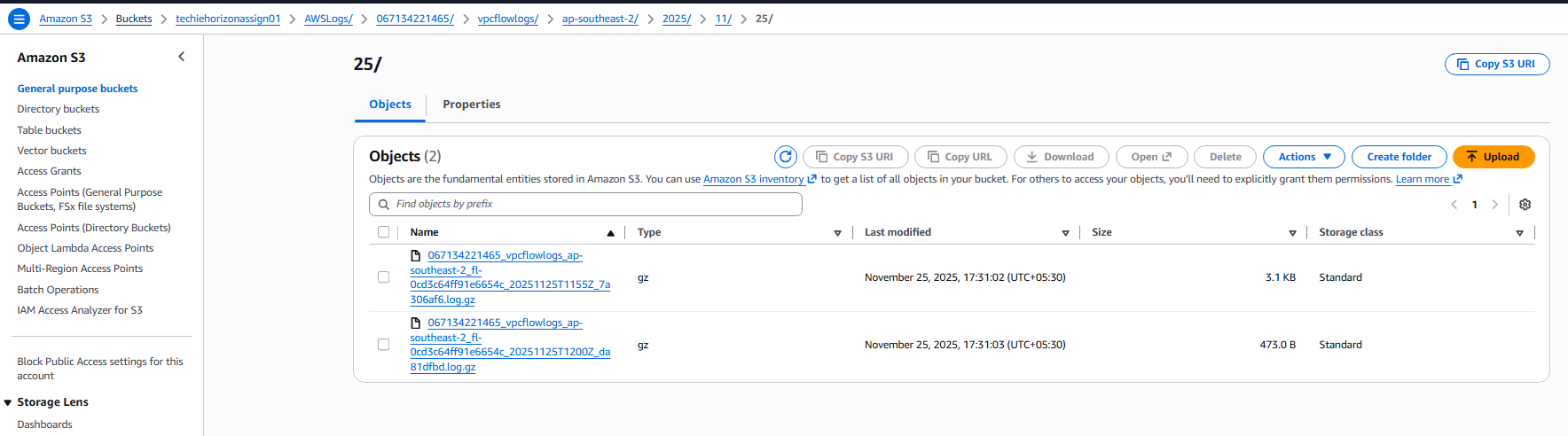
Go to vpc and select the vpc and go the flow logs option at the bottom, select flow log option



Before creating flow log create a bucket in s3 and copy the ARN number of the bucket created.



Name the flow log , paste the s3 bucket ARN number by selecting ‘send to amazon s3 bucket’

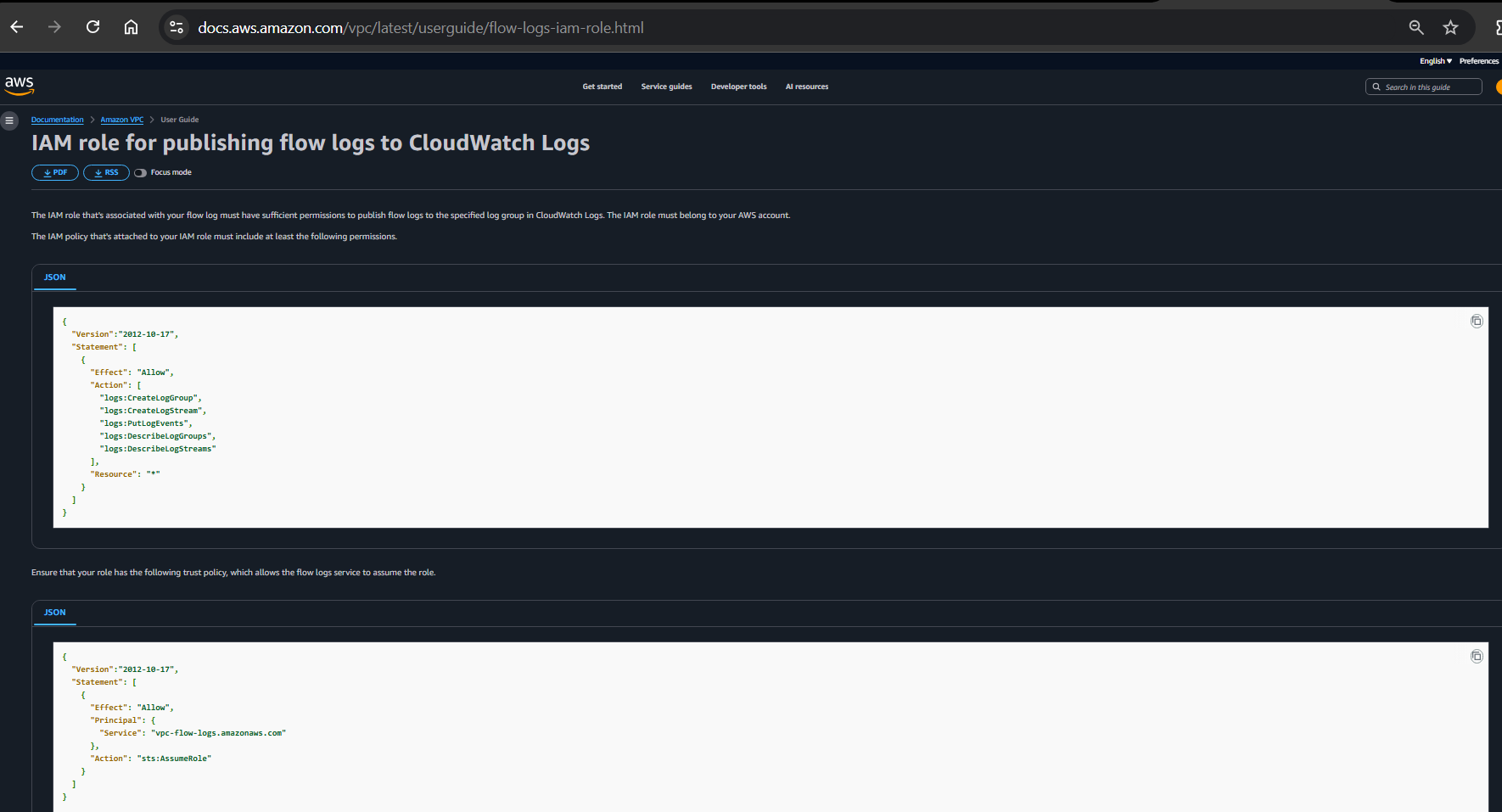


Go to bucket and at the end of the directories created we will find the logs.

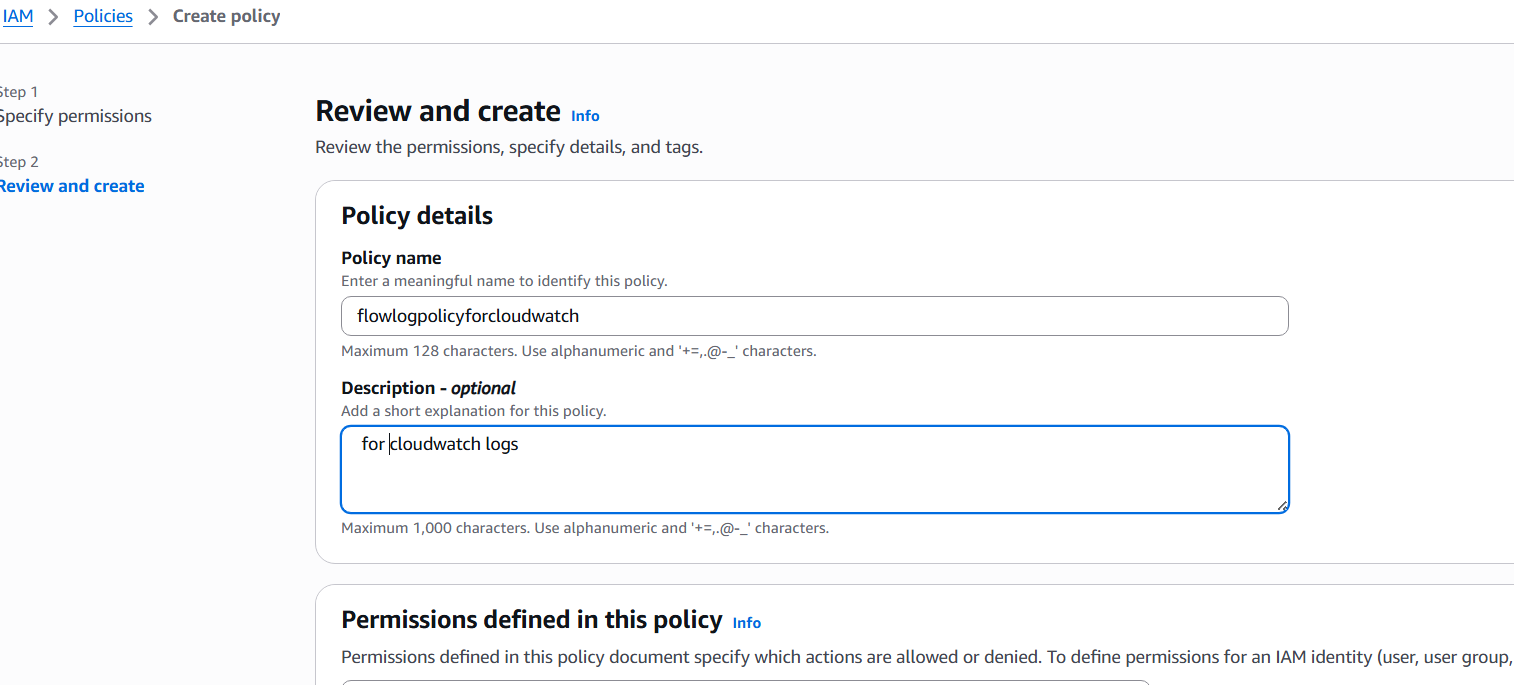
Creating cloud watch logs

Go to the aws official site on IAM role for publishing flow logs to cloud watch

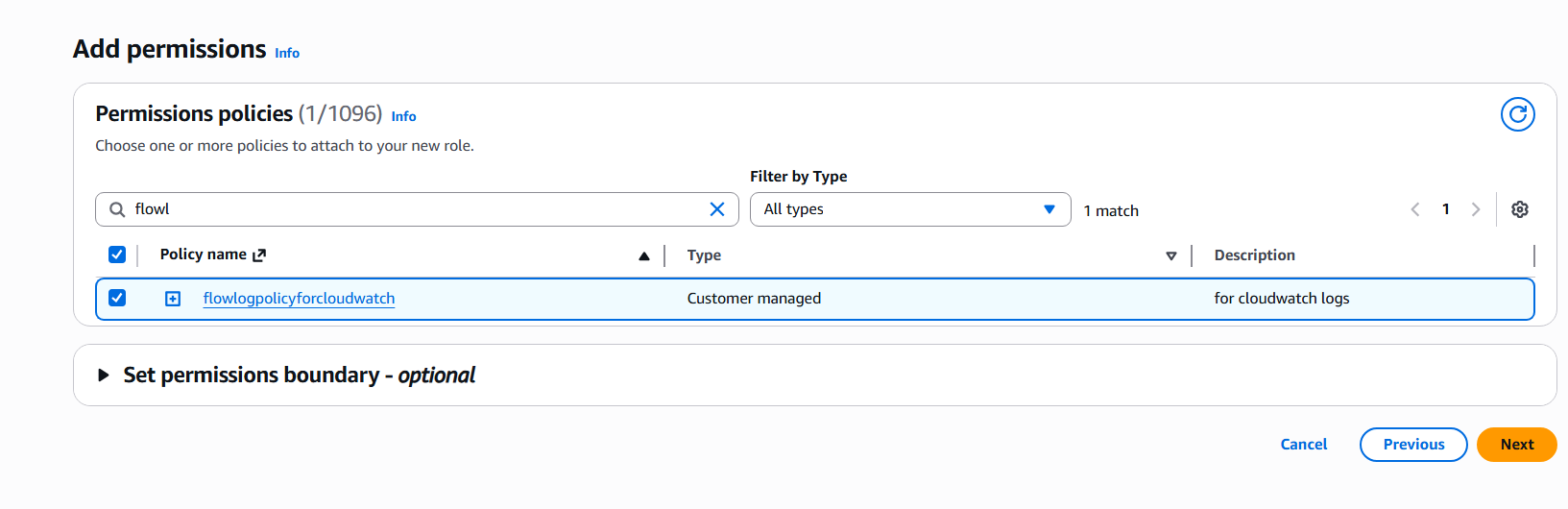
https://docs.aws.amazon.com/vpc/latest/userguide/flow-logs-iam-role.html



Got to IAM and create a policy by copying the json content on the link for the policy content



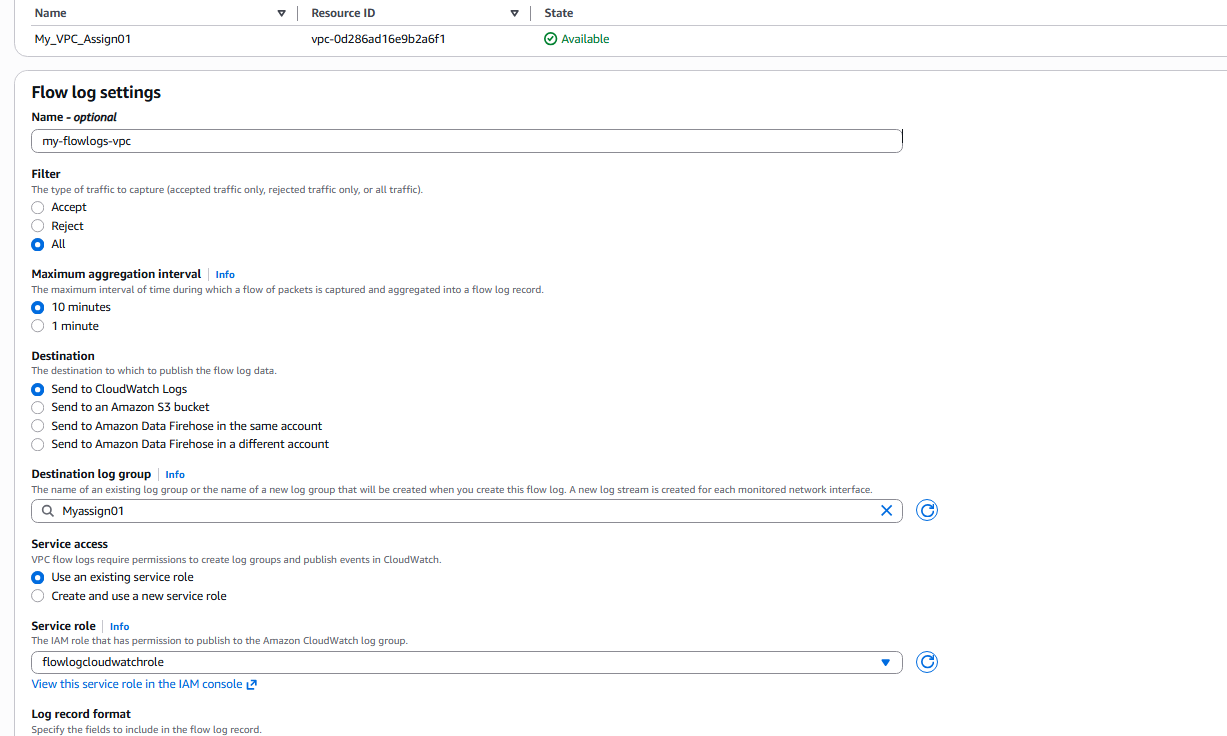
Go to IAM role and create a role, select custom thrust policy and paste the thrust policy json content and click next



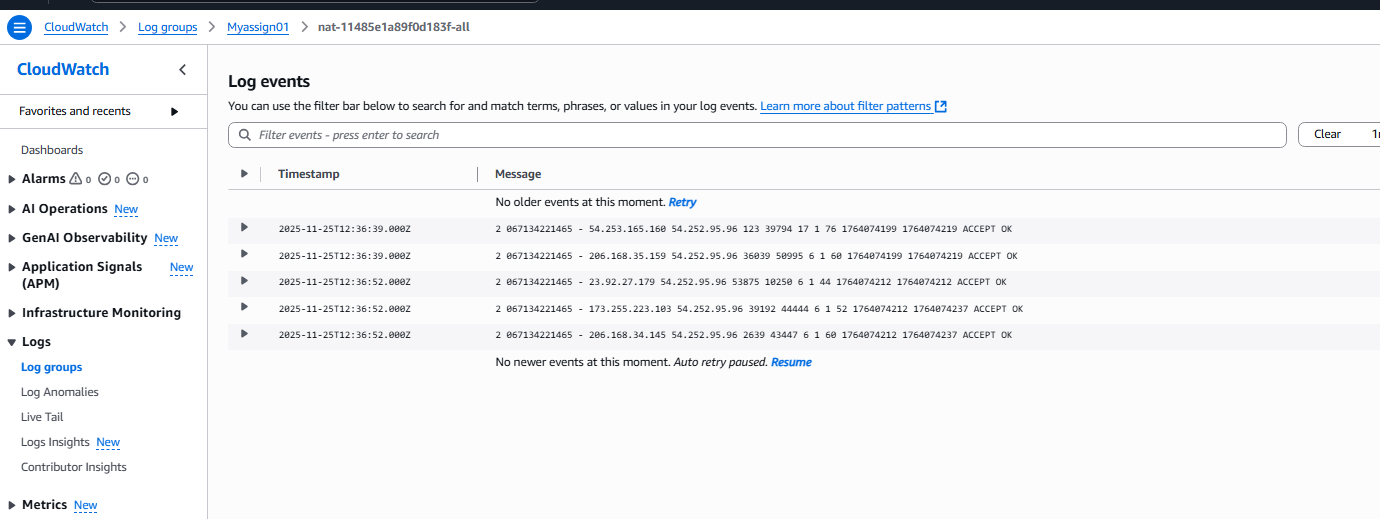
Add permissions by giving the policy name we create earlier



Give the role name and description and create a role



Go to your vpc and select create flow logs at the bottom and select the destination to send to cloudwatch logs and also give the name of the log group, select the role we created earlier



Go to the cloudwatch and go to log group, select the log group we created at the flowlog configuration, go to log events to see the logs inside the cloudwatch.