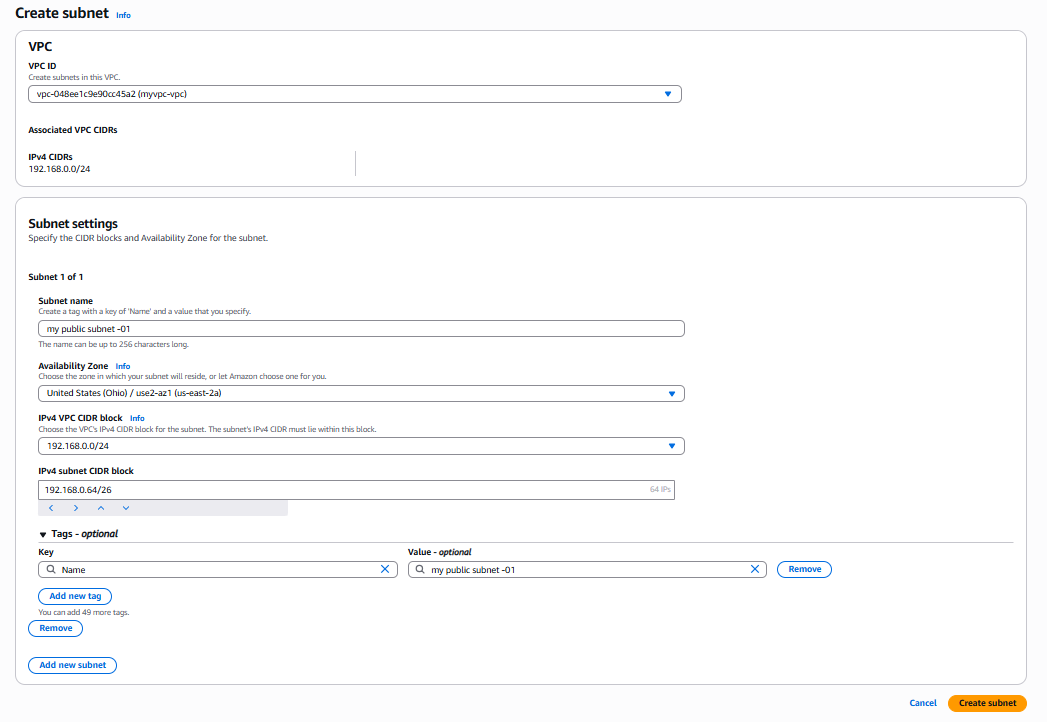
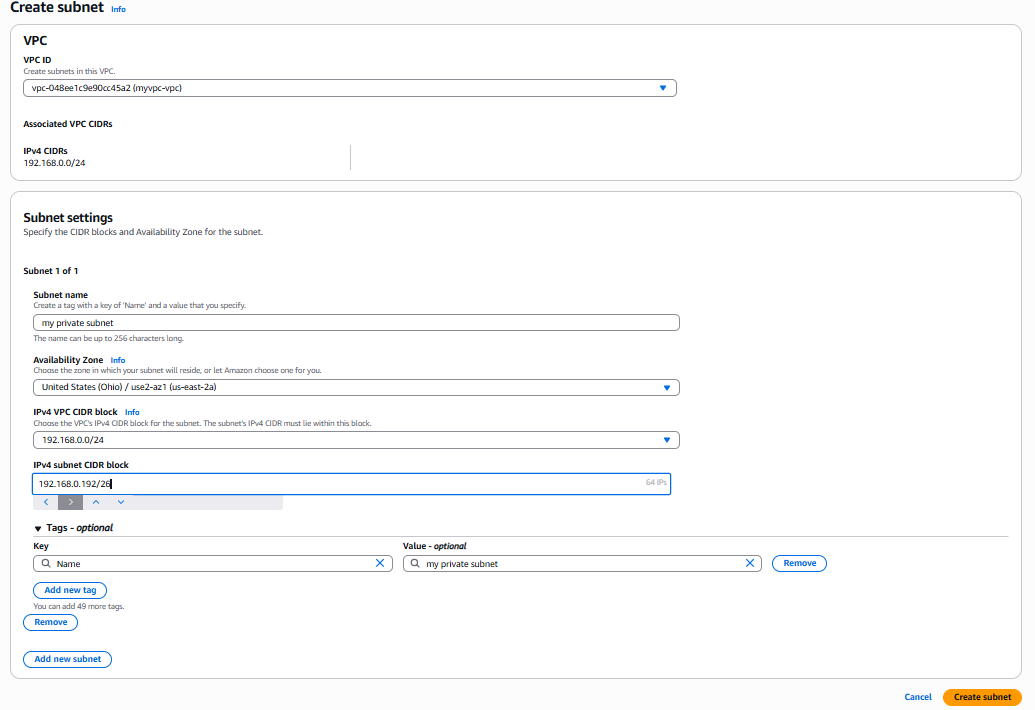
**VPC -01 TASKS**

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

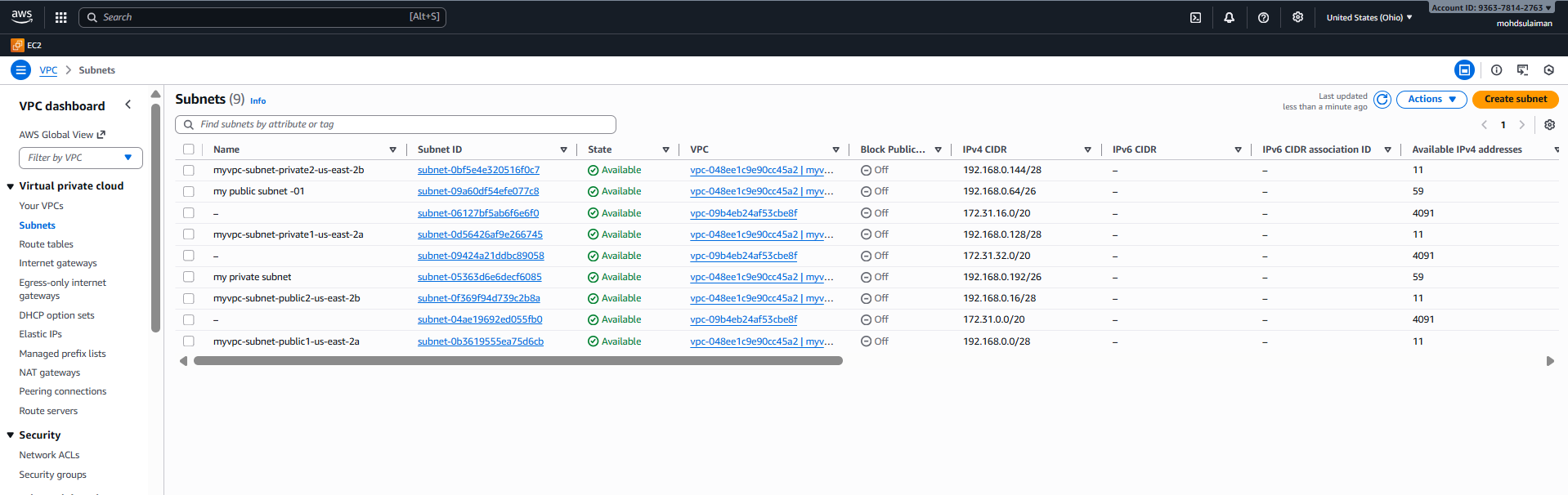
First created a vpc named myvpc



* Public subnet

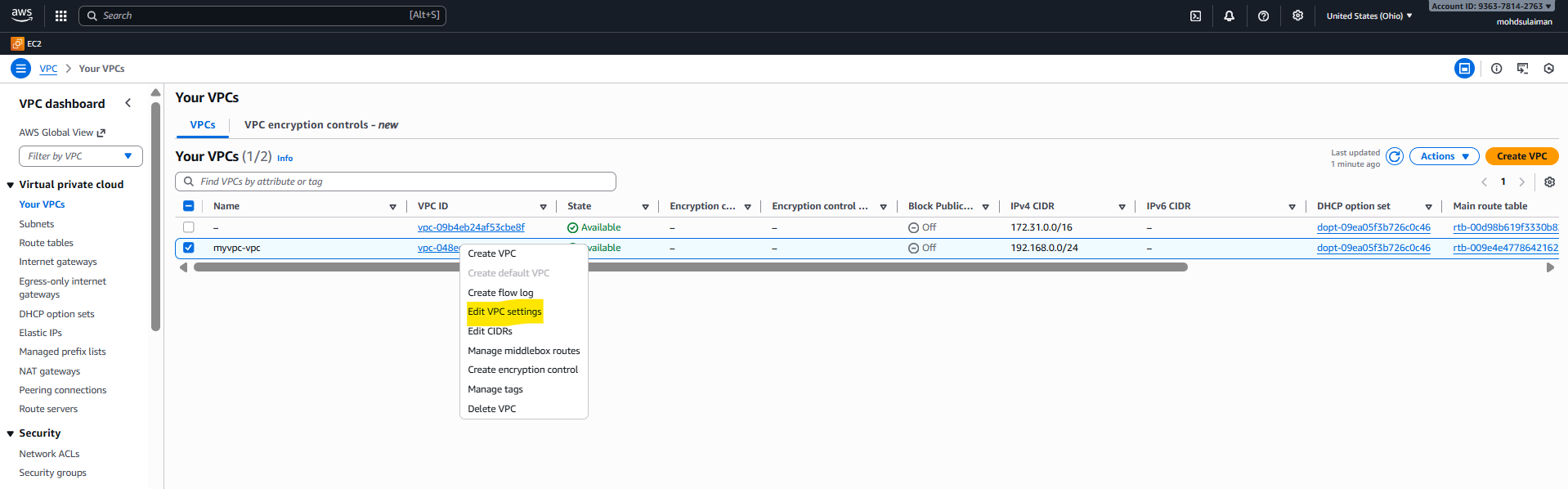


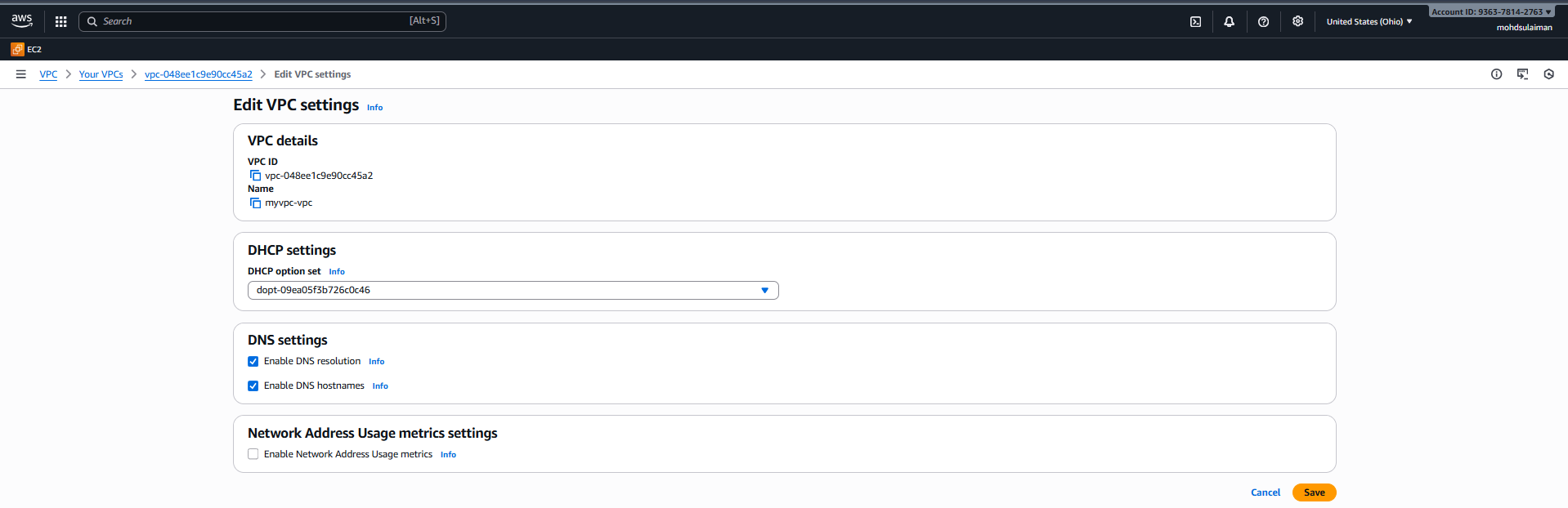
* Private subnet



2. Enable DNS Hostname in vpc

* Go to your vpc what you have created
* Select the vpc and then click to the actions and select the edit VPC
* And enable DNS hostname and click and save it





* Turn **both ON**
* Click **Save**

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

* Go to the
* AWS-VPC-SUBNET
* Select the subnet and go the actions
* Edit subnet setting
* Enable auto assign and public IPV4
* Tick the checkbox:
* And save it
* Repeat it again in subnet 2
* Select the subnet
* Choose subnet
* Go to the actions and edit subnet setting
* Enable auto-assign IPV4
* Click the checkb0x:
* Enable auto-assign public IPV4
* And save it

Usages of auto assign :

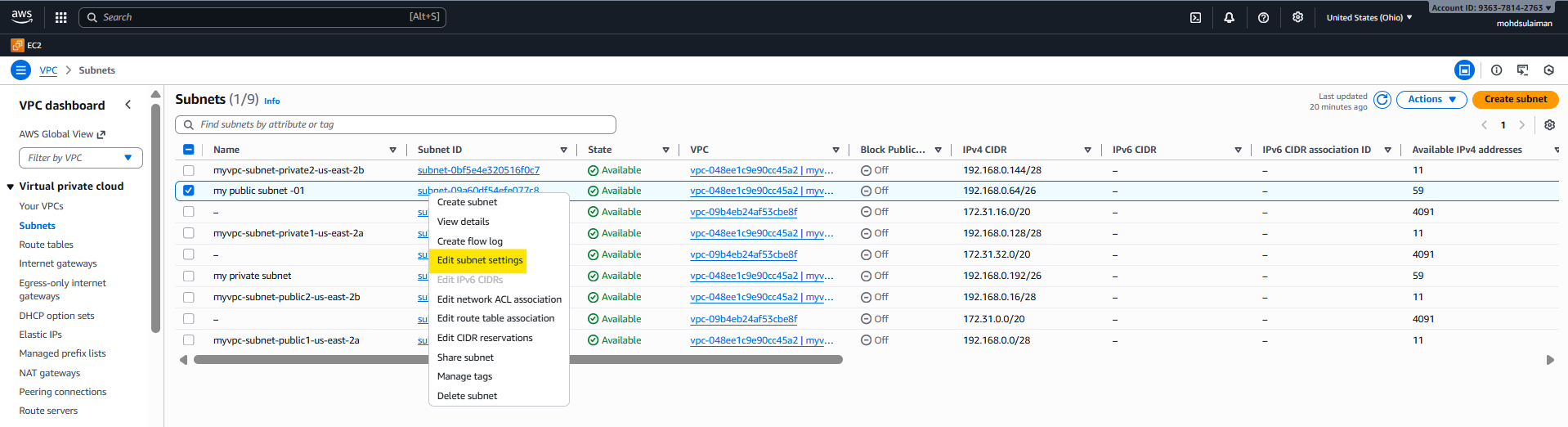
enable auto-assign in public IP give new instance a public IP automatically

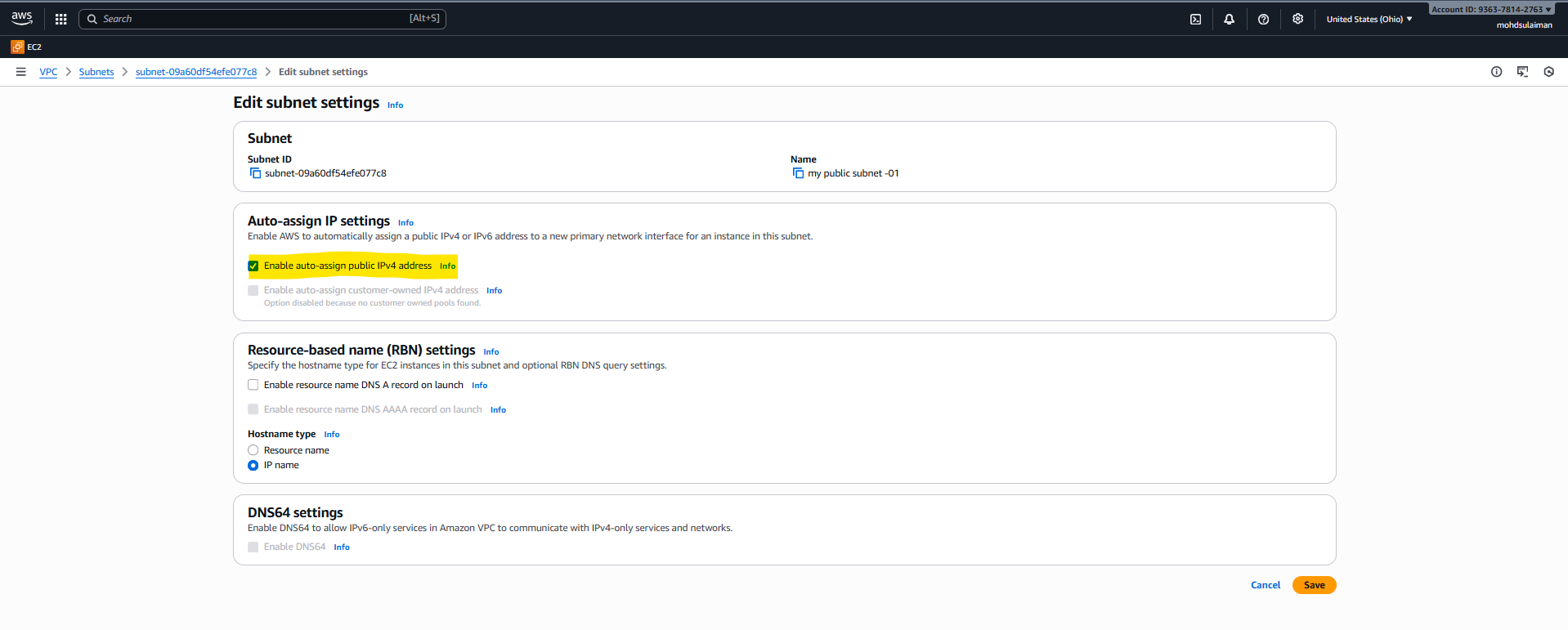
this is allow them directly connect to the internet for updates,downloads,and communication

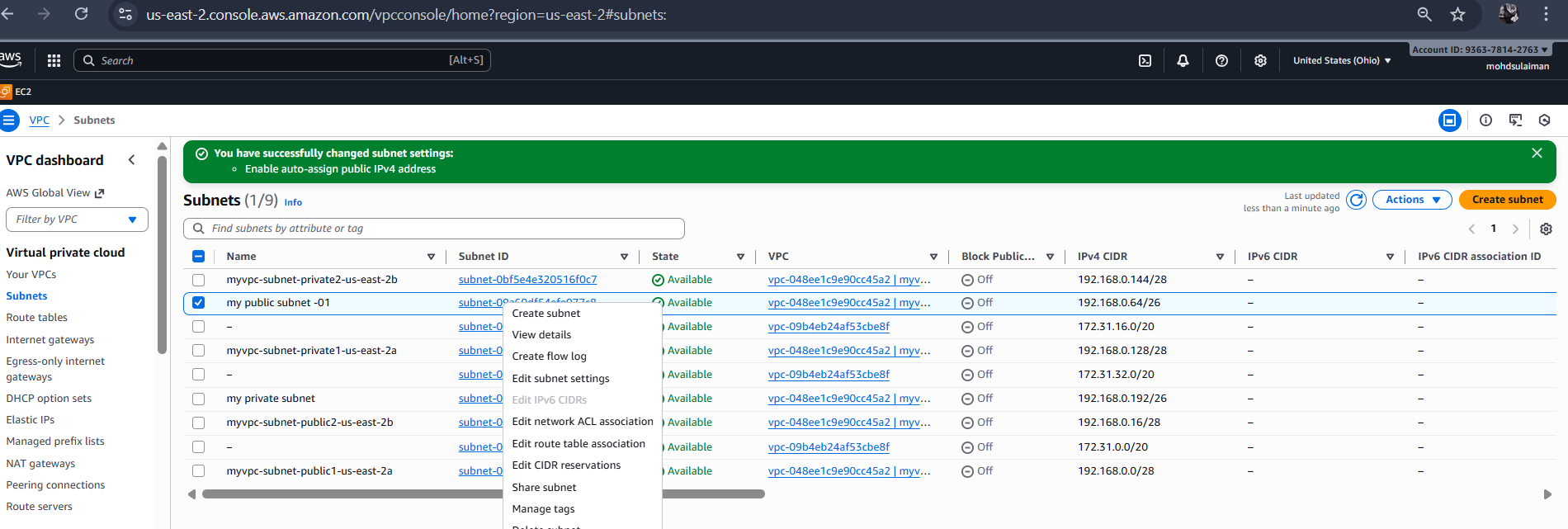
and also let you access those instance from any ssh and browser

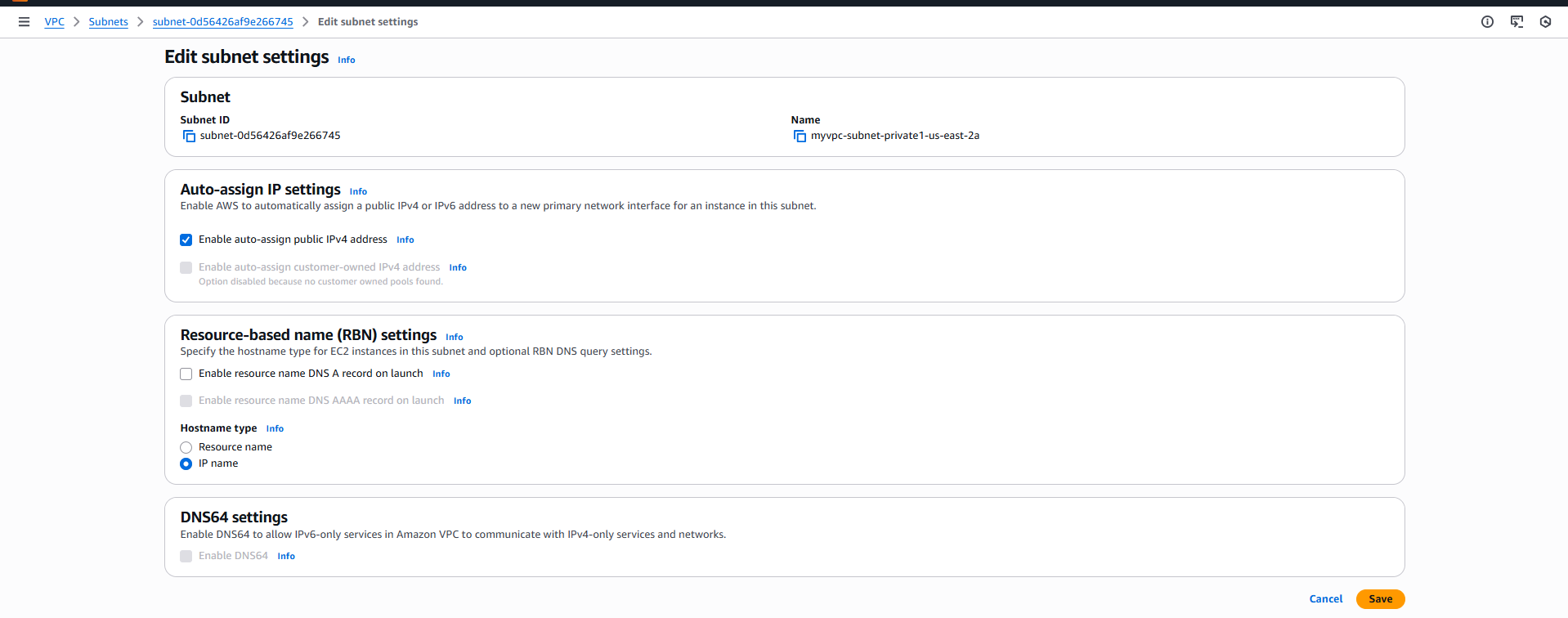
mainly we will use for internet.

Edit subnet settings



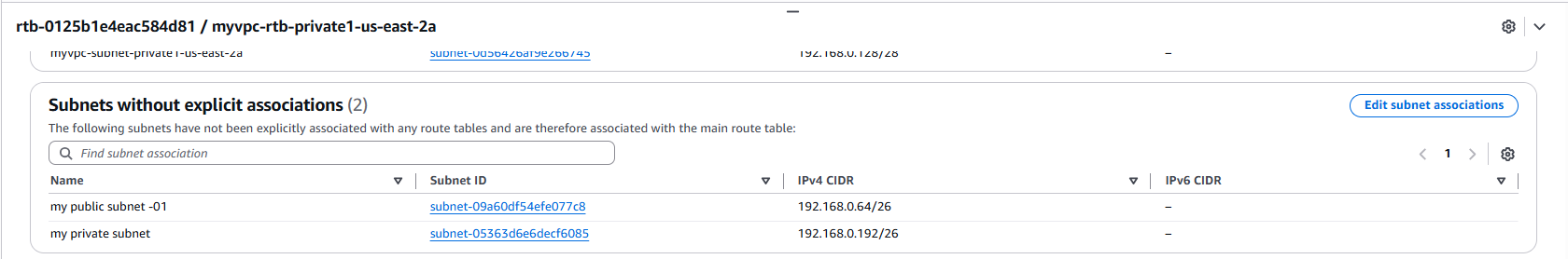


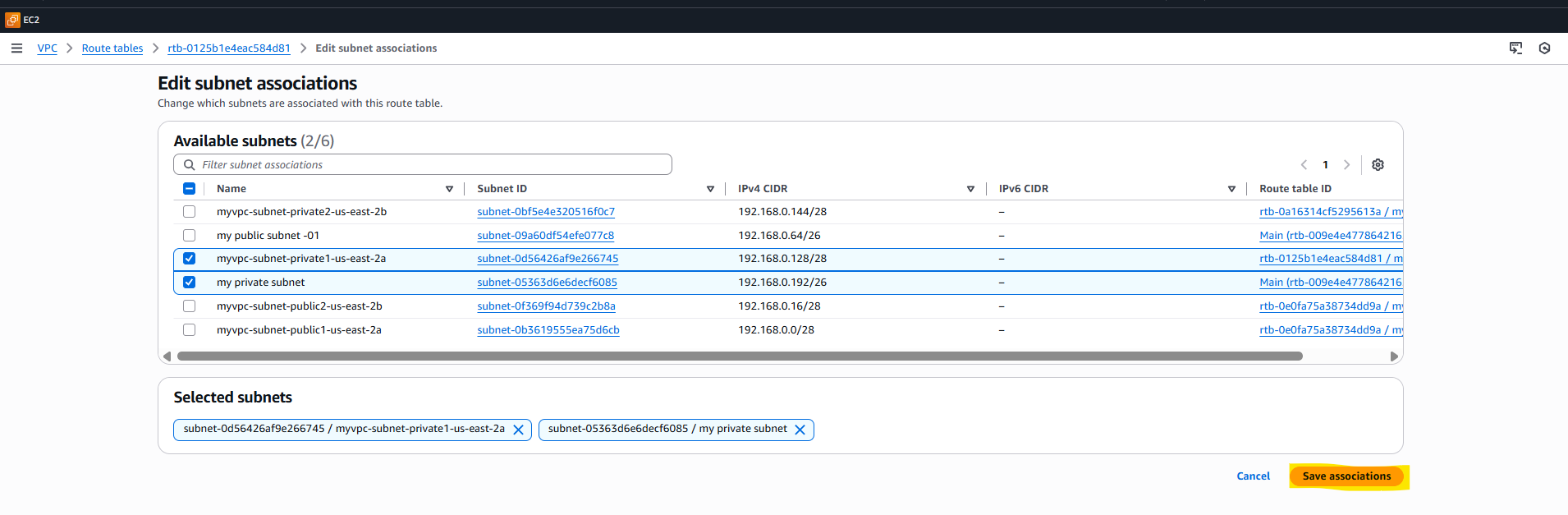


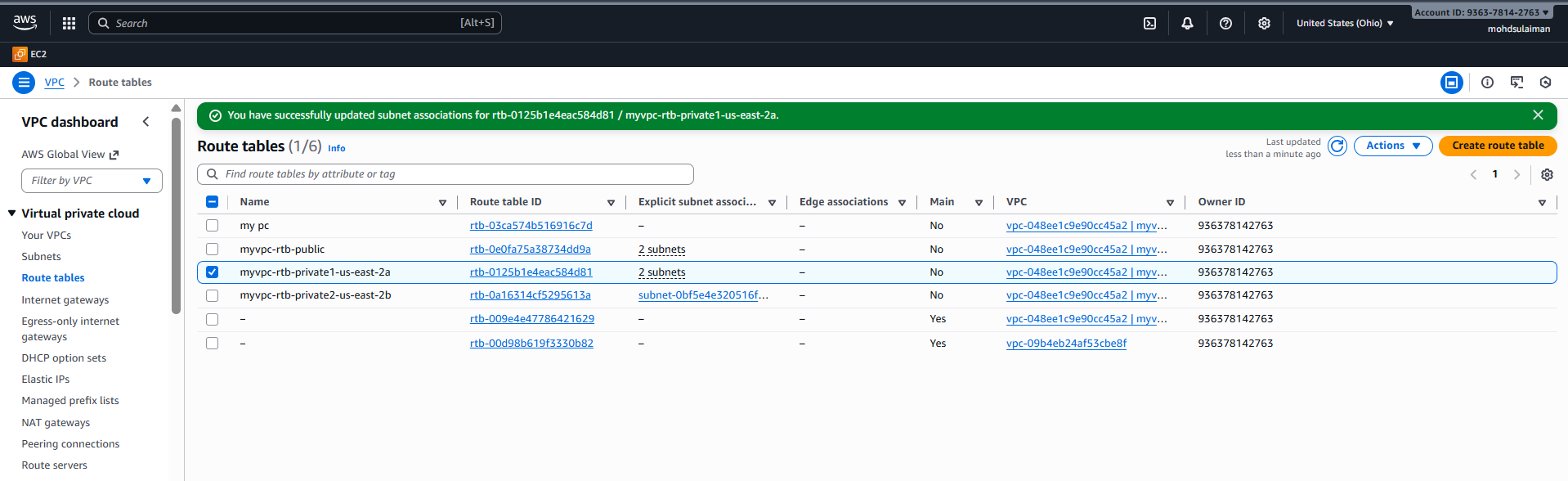


4. Add 2 private subnets in private route table.

* Step 1 : go to rout table
* Open aws consol-VPC-rout table
* Step-2 : select rout table
* Step-3 open subnet associations
* In the bottom panel : select subnet associations tab
* Click edit subnet associations
* Step -4 : select private subnet
* Private IPV4-1
* Private IPV4-2
* Step-5 :save
* Save the assocations
* Subnet private are attached to private rout table

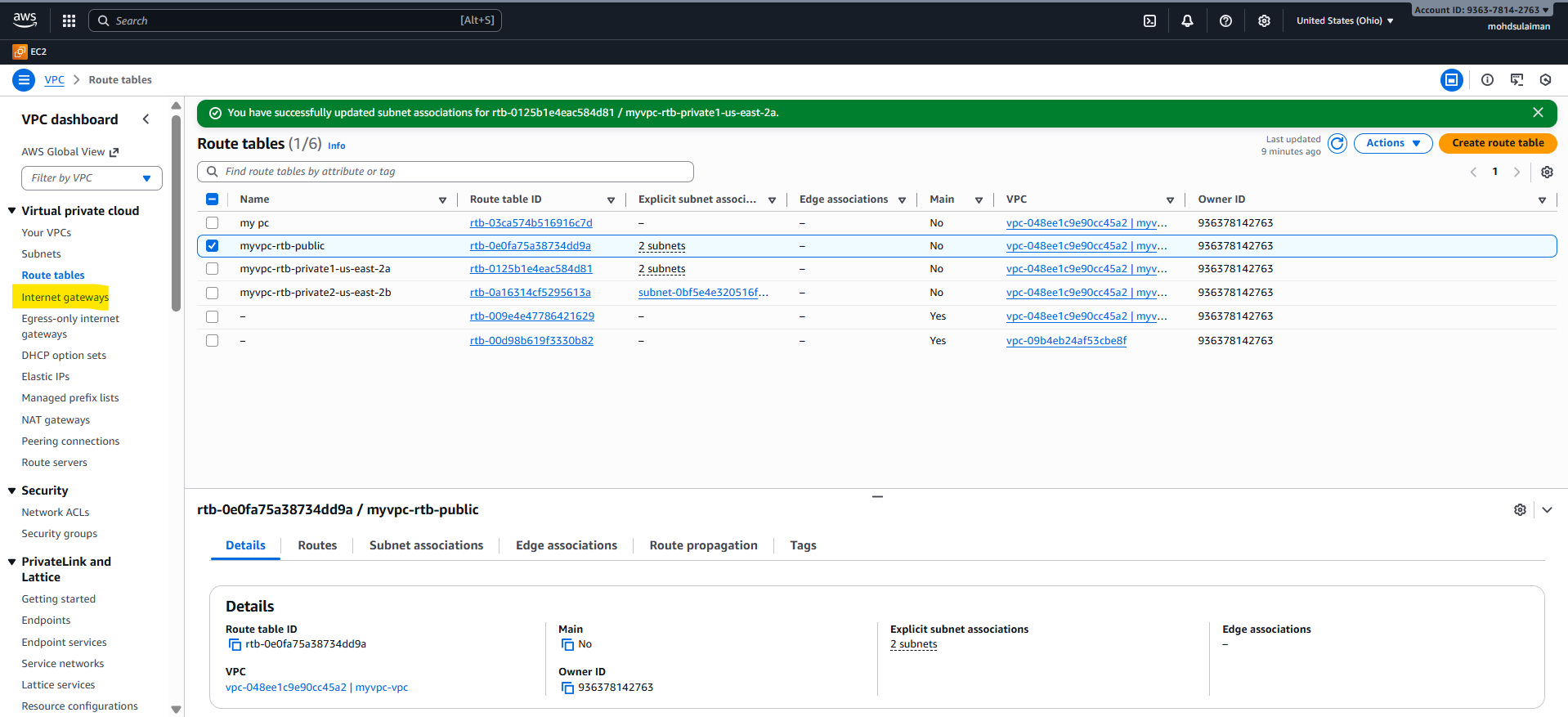


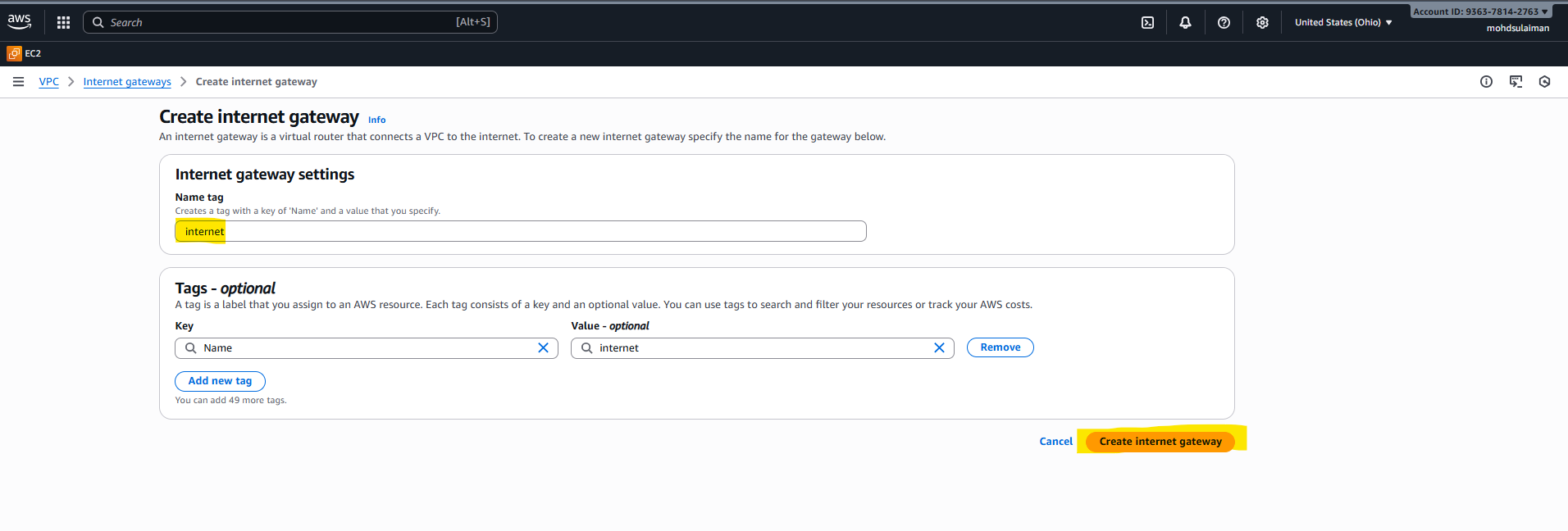




6. Public route table will have the routes to internet and local.

* Go to VPC – internet gateway
* Click create internet gateway
* Give the name of internet
* After creating – select it
* Go to actions-attach VPC
* Choose your VPC-attach’
* VPC-ROUTABLE-PUBLIC-RT-ROUTS-EDIT ROUTS-
* ADD
* Destination
* On target
* Select IP 0.0.0.0/0 : select internet gateway





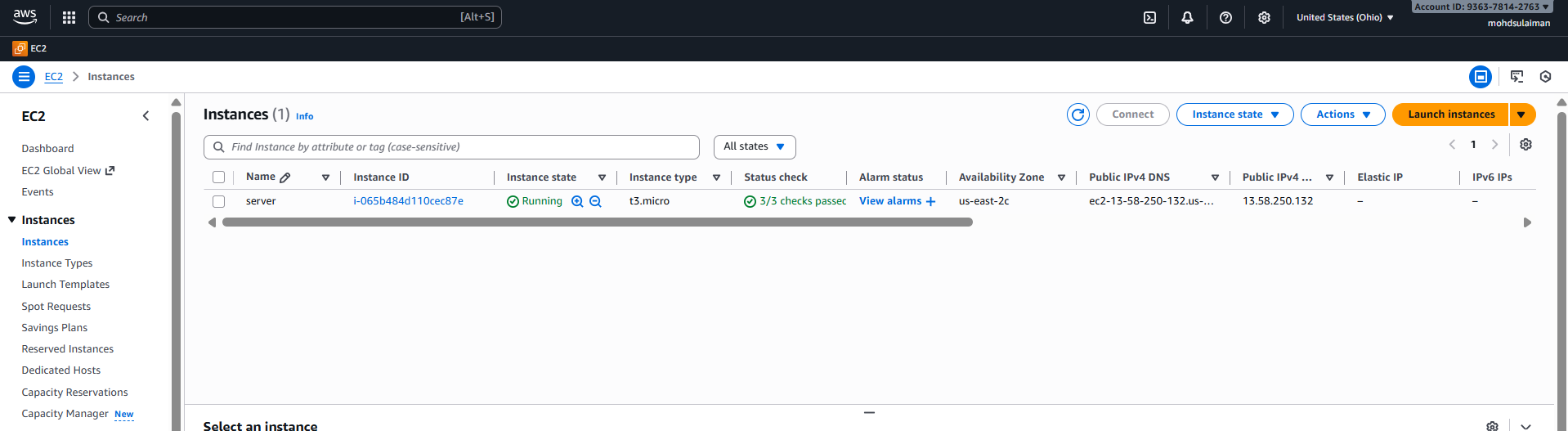


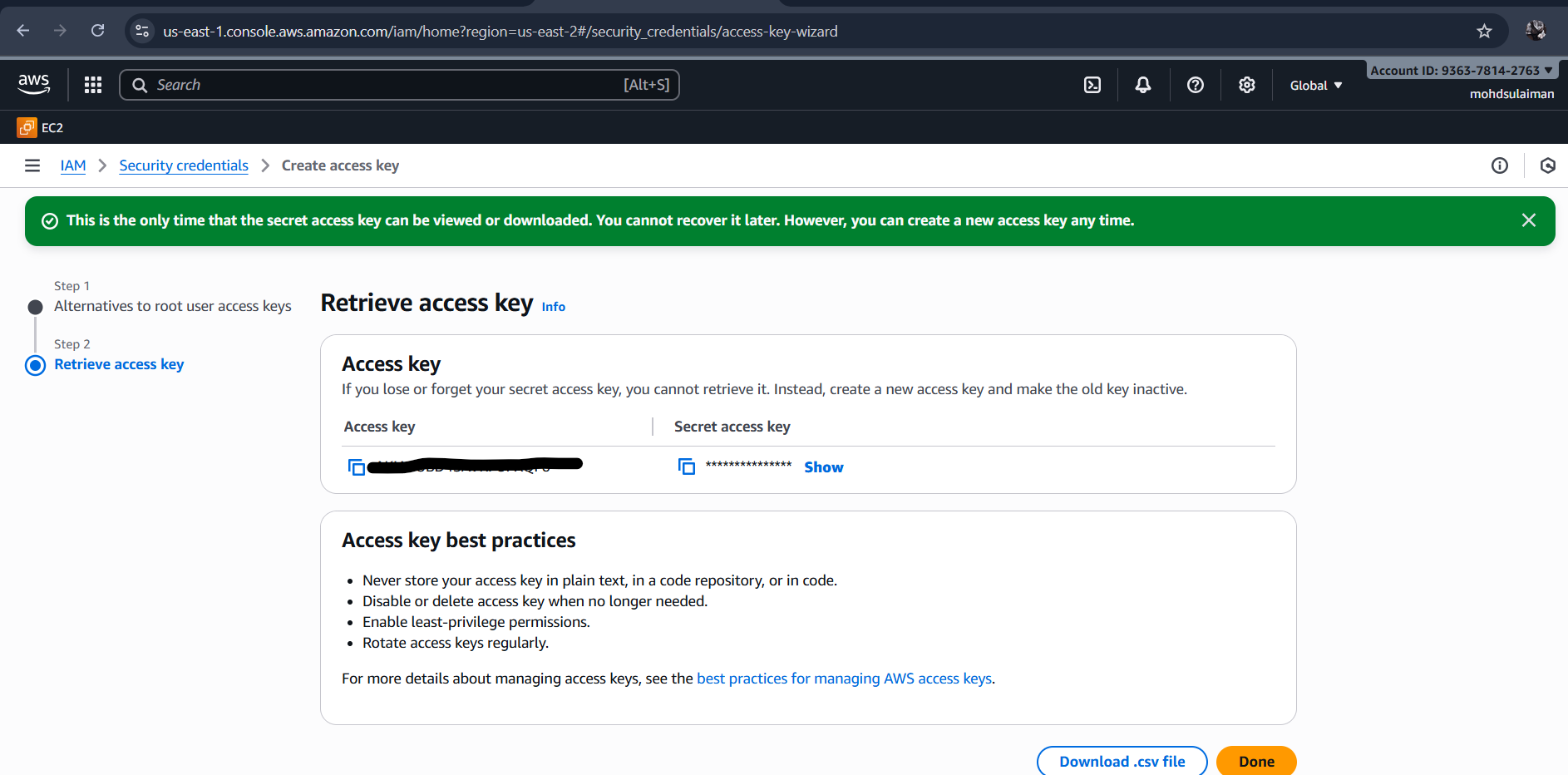


I have already attached my vpc internet getway

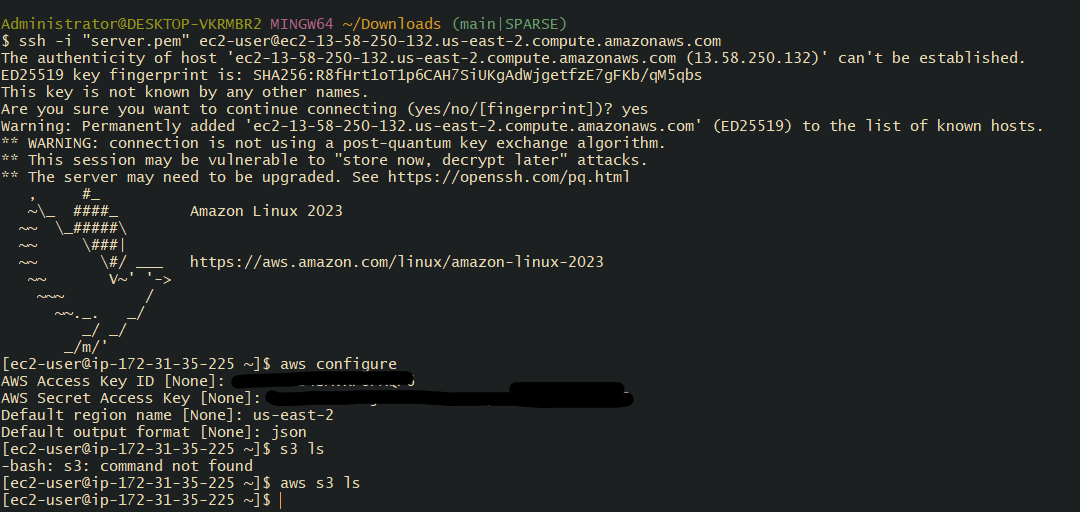
7. Create EC2 in public subnet with t2.micro and install PHP.

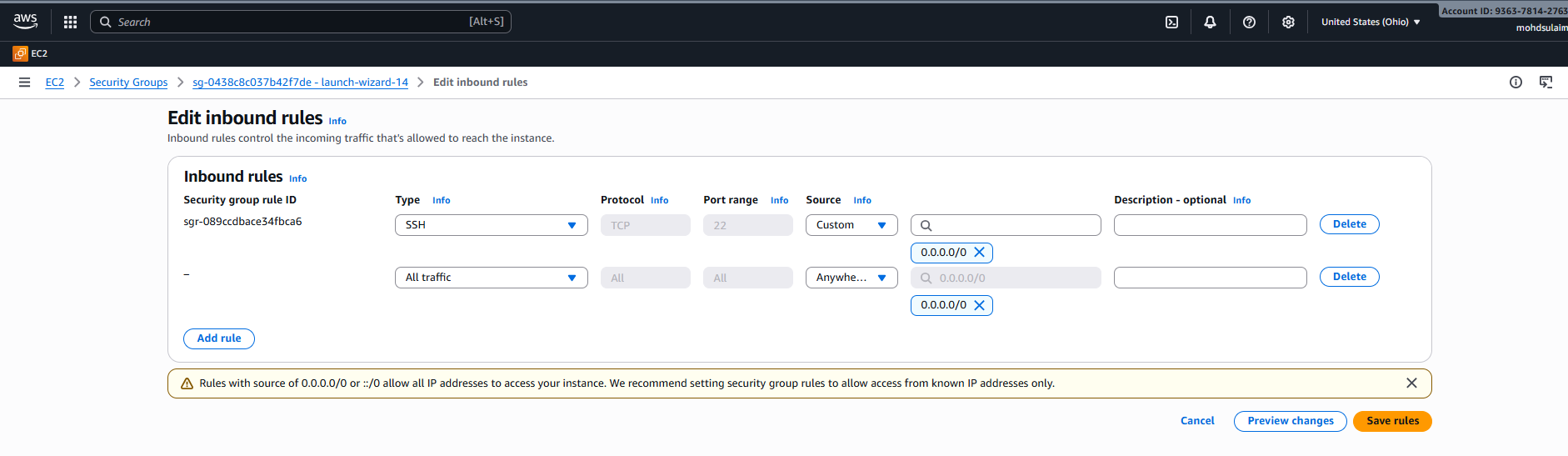
* Go to Aws
* Launch one instance : select the one name to the instance
* Give operating system (AMI)
* In the networking settiings .choose VPC and select the public IP
* In security groups add the inbounds rules
* SSH (PORT:22)
* HTTP(PORT:80)
* And connect to the terminal
* Successfully connected
* Go to the root user
* Sudo update -y
* Sudo yum install httpd -y
* Sudo systemctl status httpd
* Sudo syste,ctl status httpd (activated httpd)
* Sudo yum install php php-cli php-mysqlnd -y
* Sudo systemctl restart httpd
* Them browse with your IP address
* Its working



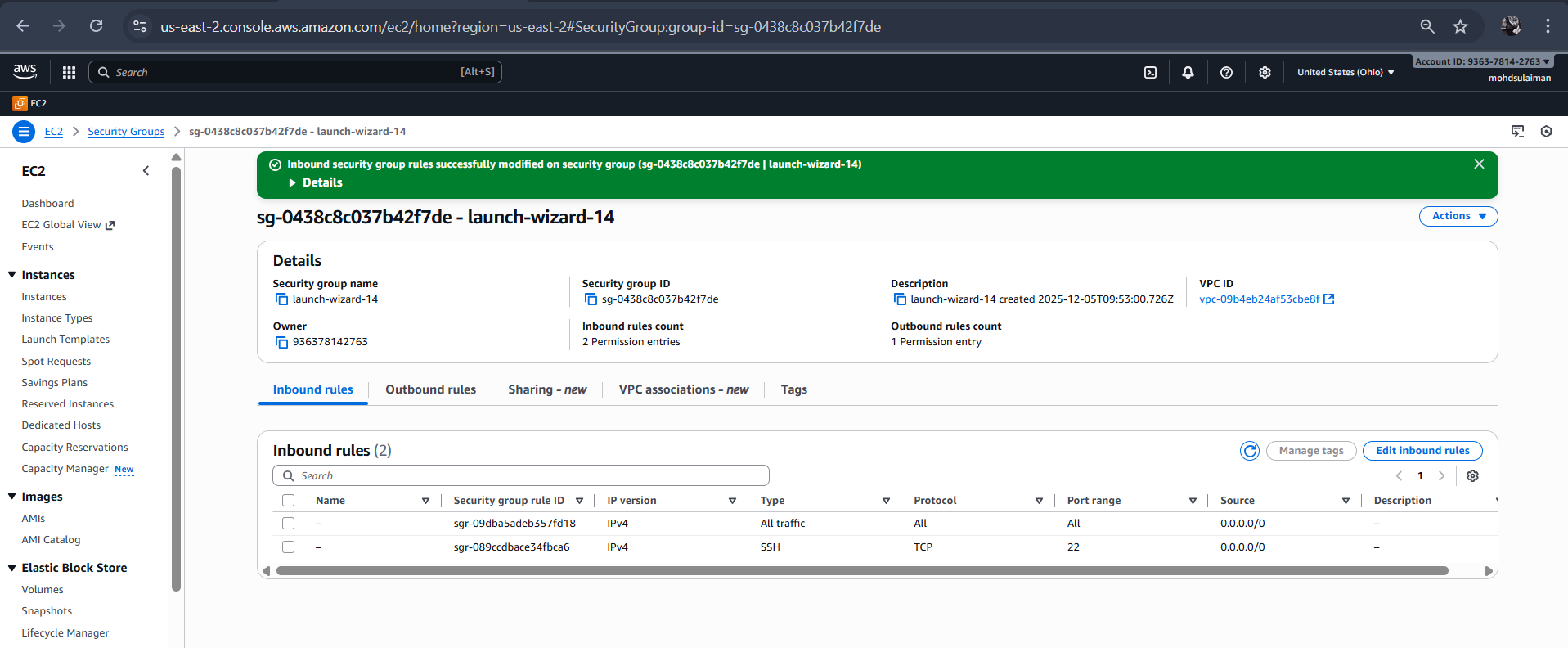


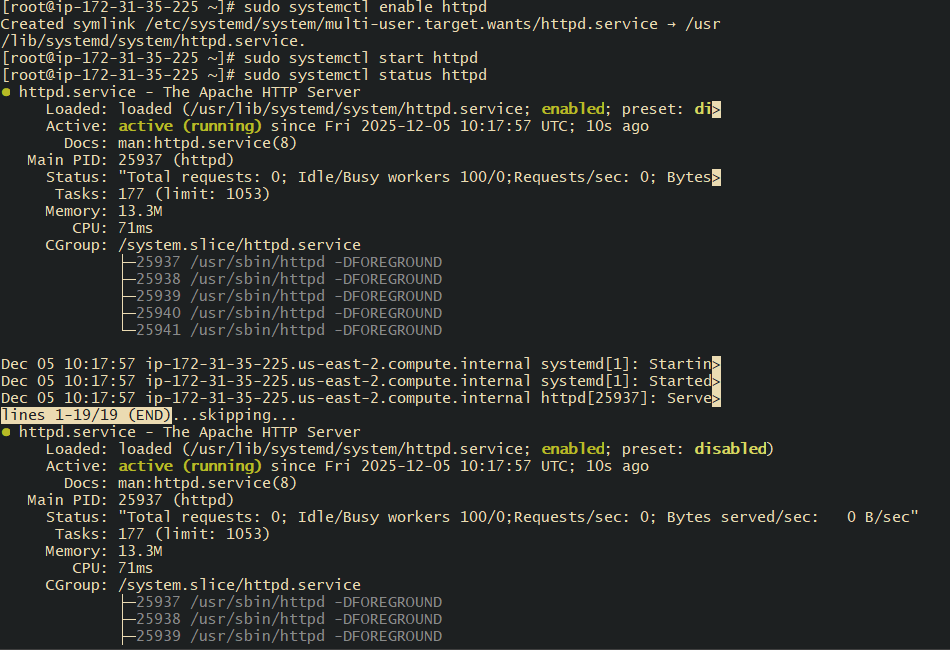
Now configure your aws using access key and secret access key

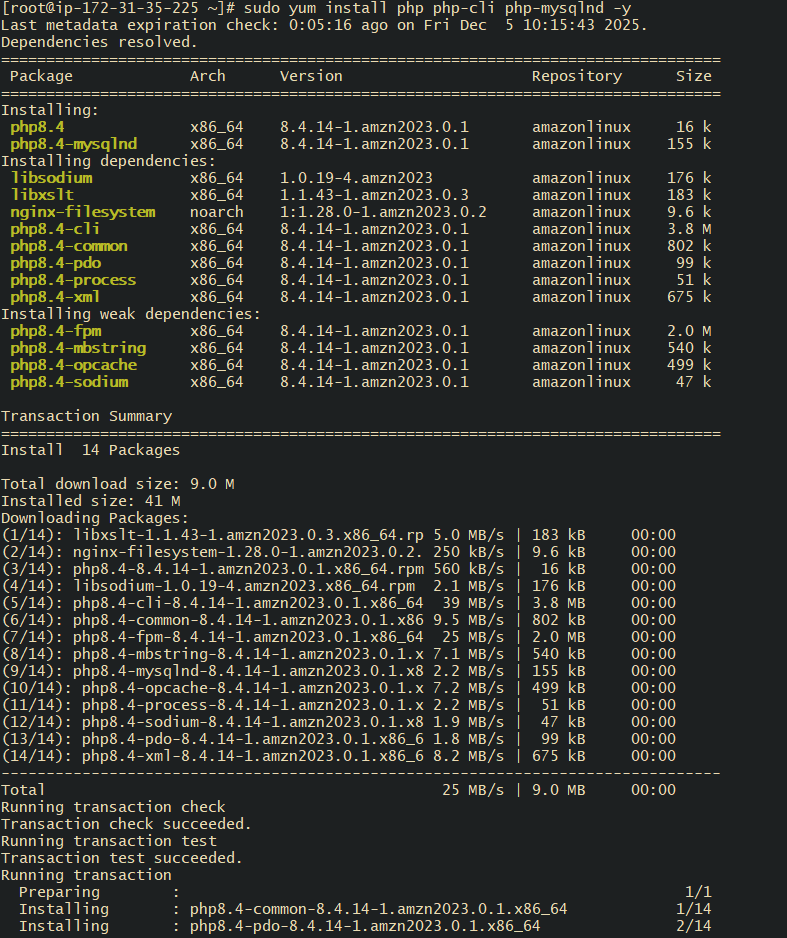




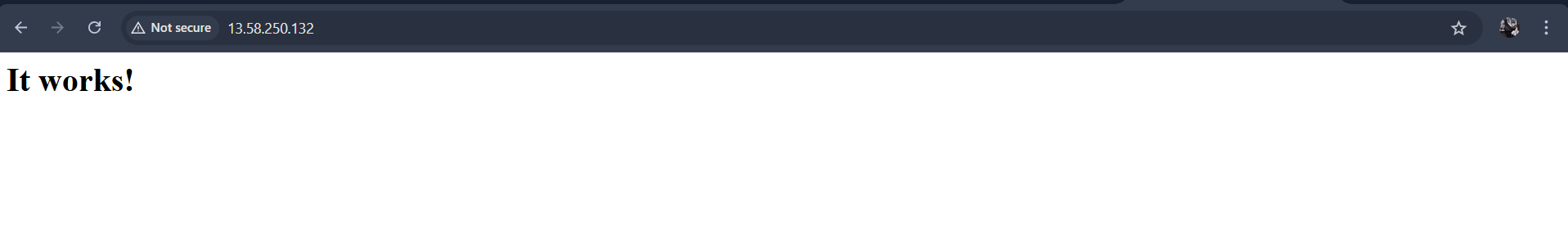
Make sure you allow all traffic in it





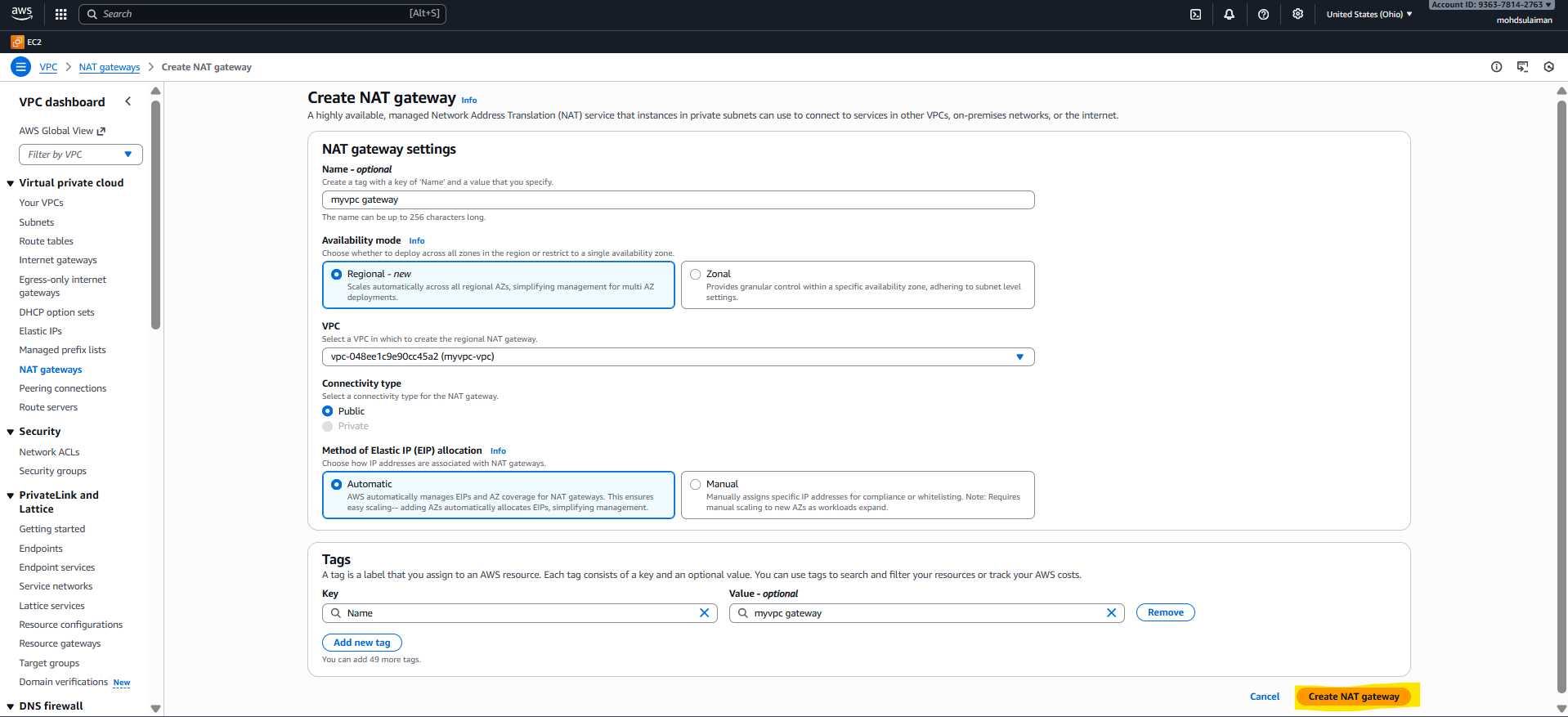


Now we successfully install mysql which is actually database



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

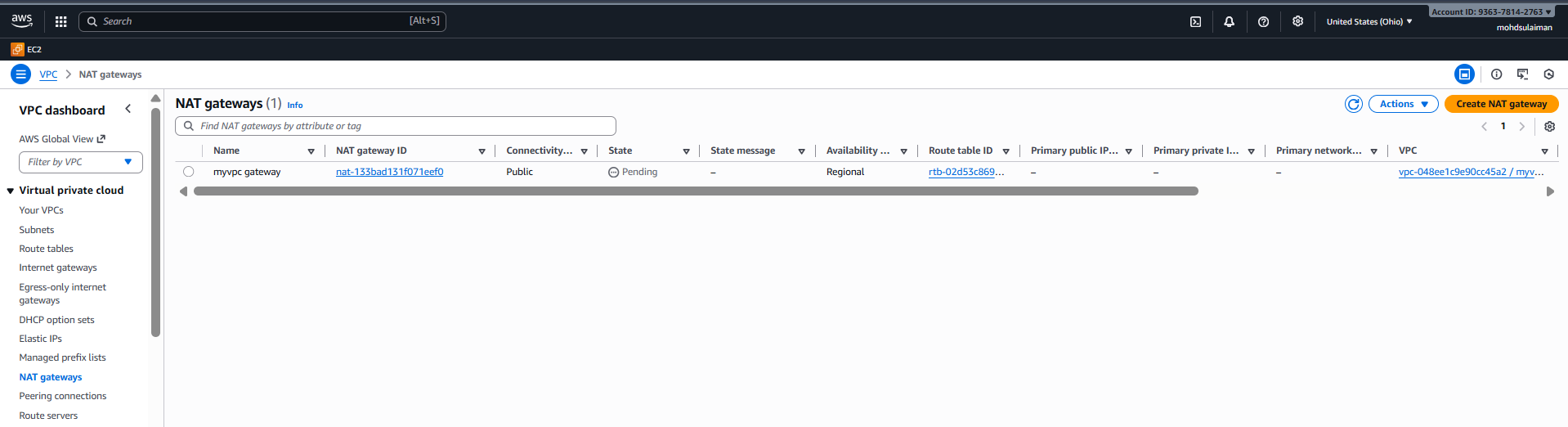
* **Allocate an Elastic IP (EIP)**
* Go to **VPC Dashboard** → **Elastic IPs**.
* Click **Allocate Elastic IP address** → Choose
* **Create the NAT Gateway**
* Navigate to **VPC Dashboard** → **NAT Gateways**.
* Click **Create NAT Gateway**.
* Select the **public subnet** where the NAT Gateway will be deployed.
* Assign the previously allocated **Elastic IP** to the NAT Gateway.
* Click **Create NAT Gateway** and wait for its status to become "Available."
* **Update the Private Route Table**
* Go to **VPC Dashboard** → **Route Tables**.
* Select the route table associated with your private subnets.
* Click **Edit Routes** and add a route with the following:
  + **Destination:** 0.0.0.0/0 (all internet traffic)
  + **Target:** Select the **NAT Gateway** you created.
* **Associate Private Subnets with the Route Table**
* In the route table, go to **Subnet Associations**.
* Click **Edit Subnet Associations**.
* Select the private subnets to associate with this route table and save
* **Verify and Test Connectivity**
* Launch or use an existing EC2 instance in a private subnet.
* Connect to the instance via a Bastion host or AWS Systems Manager.
* Test outbound connectivity by running a command like curl to an external website or updating software packages.
* **Verification Checklist**
* **NAT Gateway Status:** Ensure the NAT Gateway is "Available."
* **Route Table Configuration:** Confirm that the route table for private subnets has the default route (0.0.0.0/0) pointing to the NAT Gateway.
* **Private Instance Access:** Verify that private instances can access the internet but do not have public IPs.

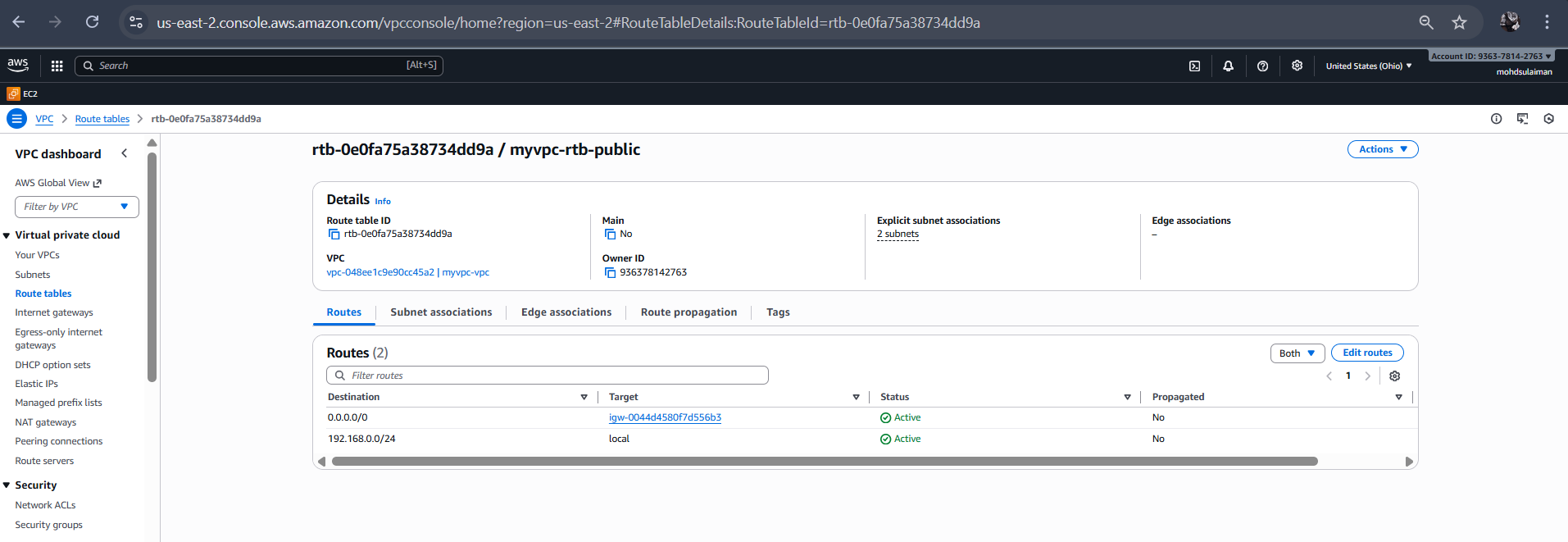


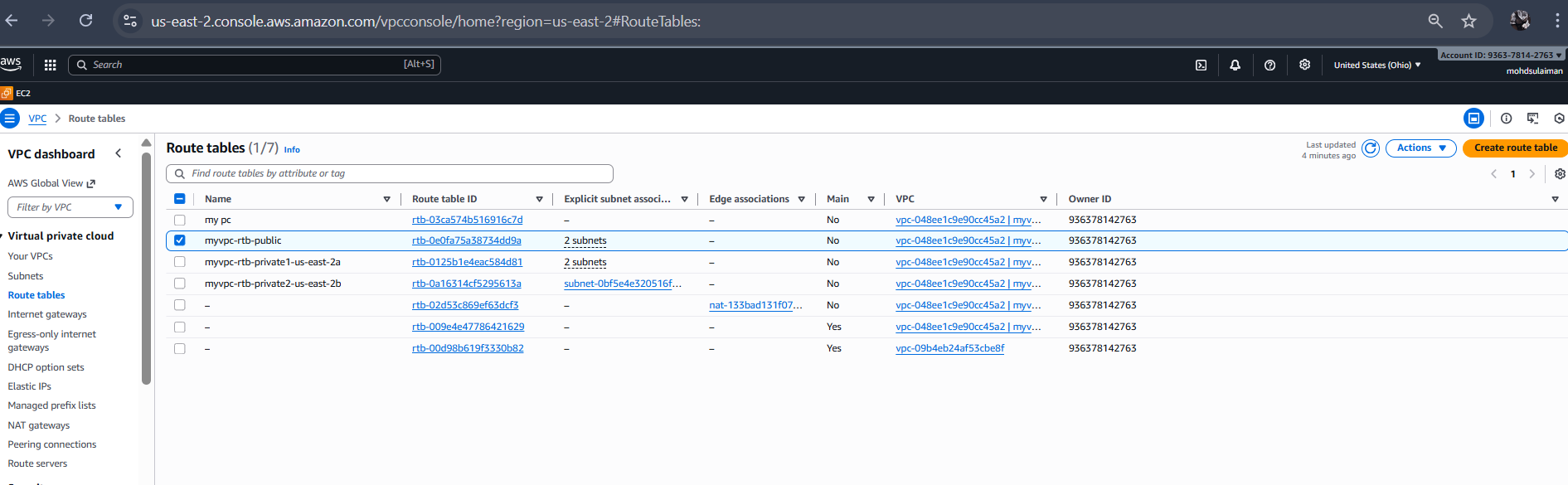
Important step:-

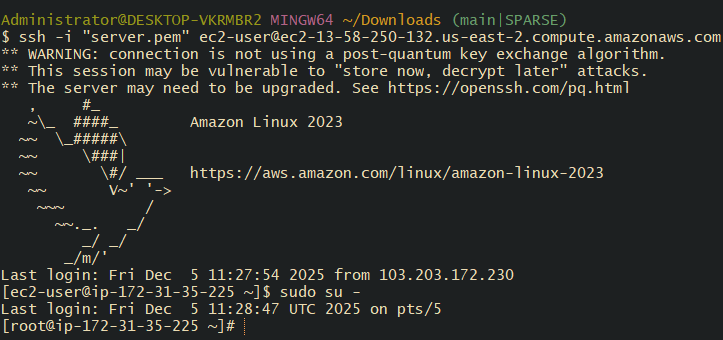
Connectivity type

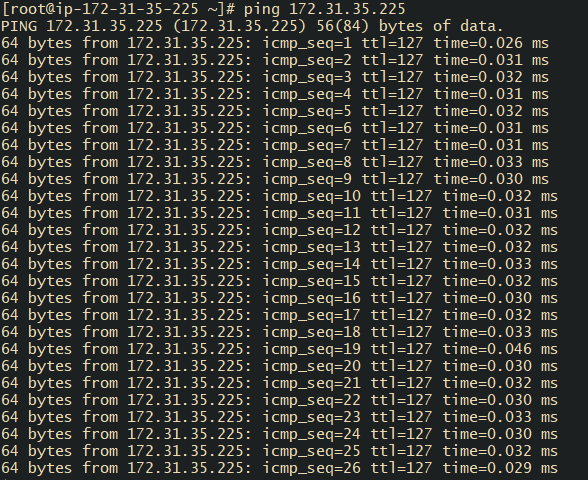
Public





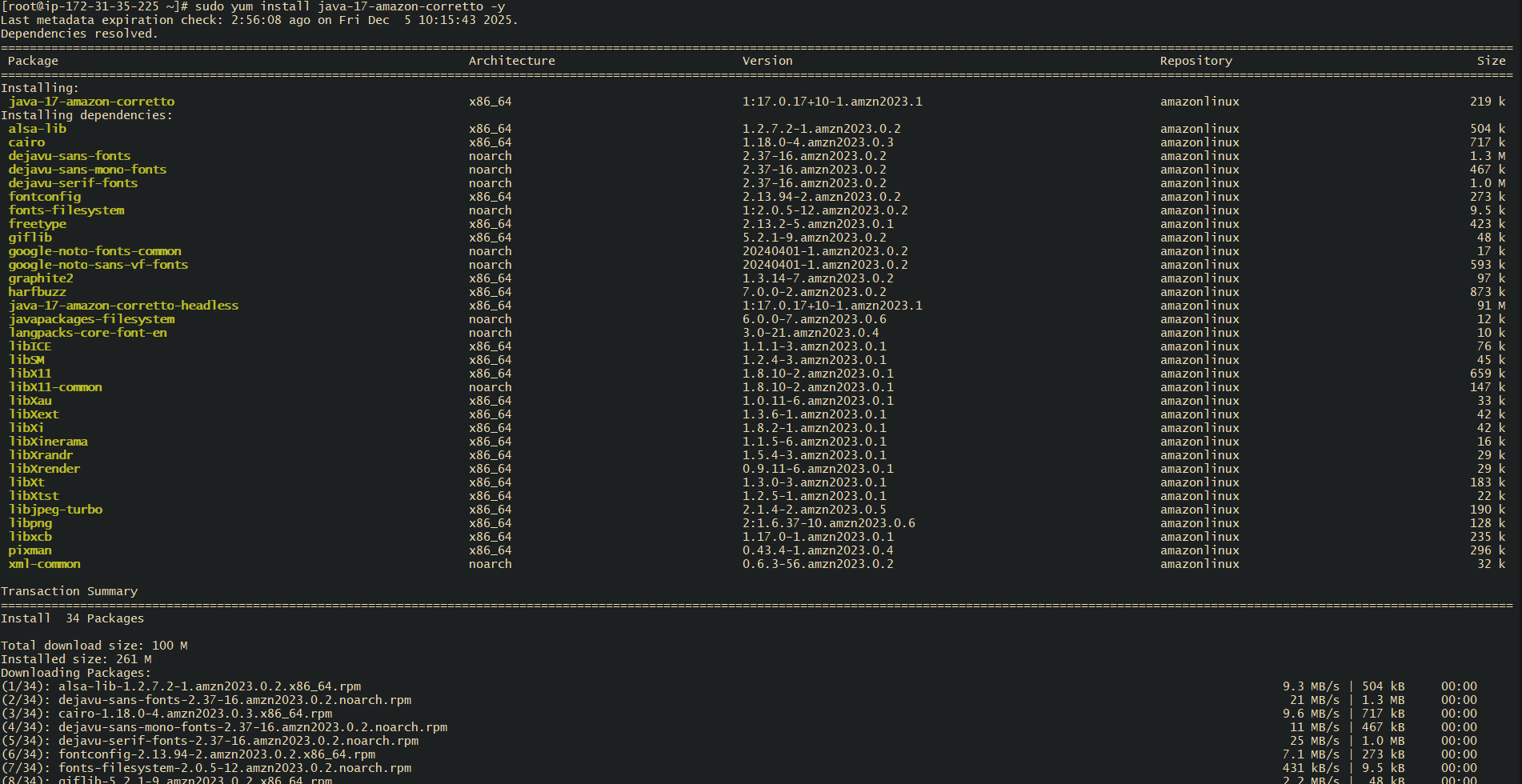




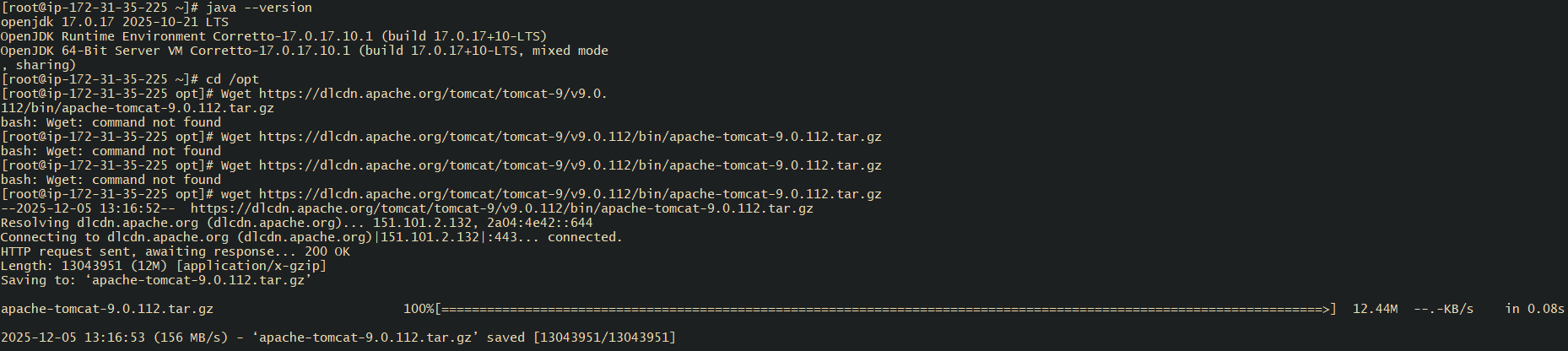


9. Install Apache Tomcat in private EC2 and deploy a sample app.

Connect to the local

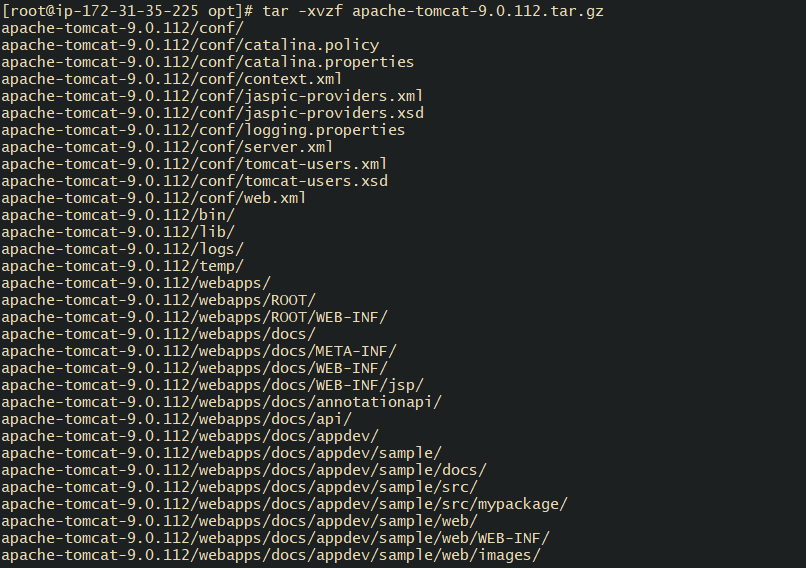


Successfully installed the java



wget <https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.112/bin/apache-tomcat-9.0.112.tar.gz>

extract it :- tar -xvzf apache-tomcat-9.0.112.tar.gz



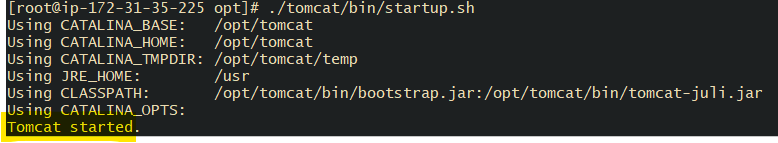
Rename it

mv apache-tomcat-9.0.112 tomcat

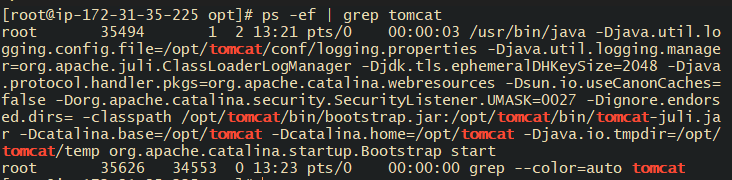


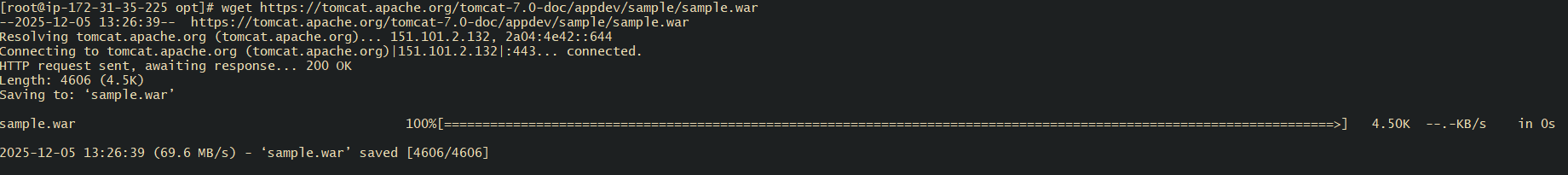
Make sure you give it permission

Start Tomcat:



Ps -ef | grep tomcat : checking the system running status





Now wget <https://tomcat.apache.org/tomcat-7.0-doc/appdev/sample/sample.war>

Just type wget <https://tomcat.apache.org/tomcat-7.0-doc/appdev/sample/sample.war>



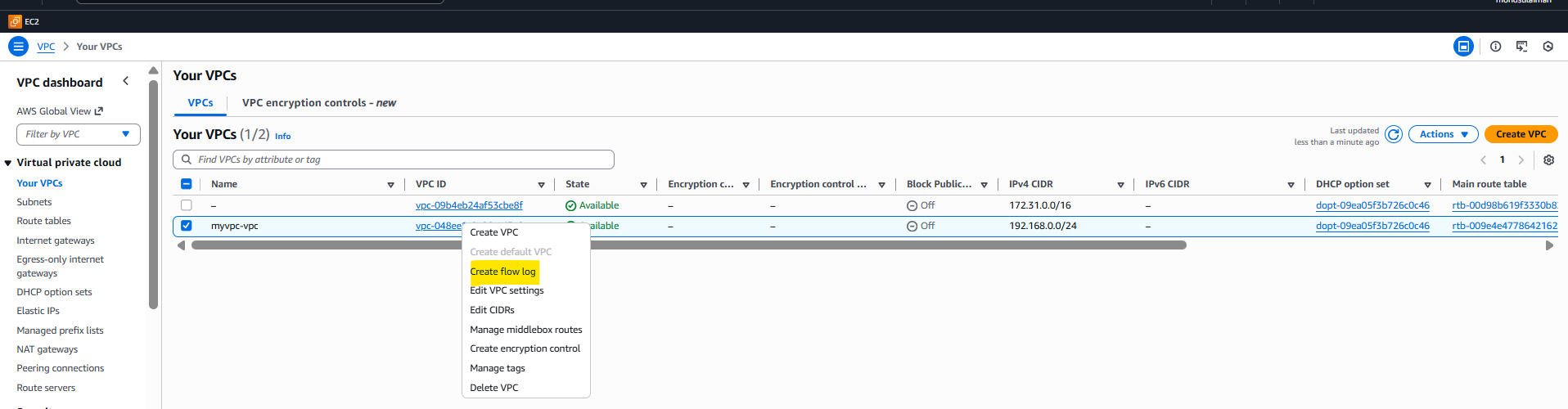
Change directory : cd sample

Then type : ls

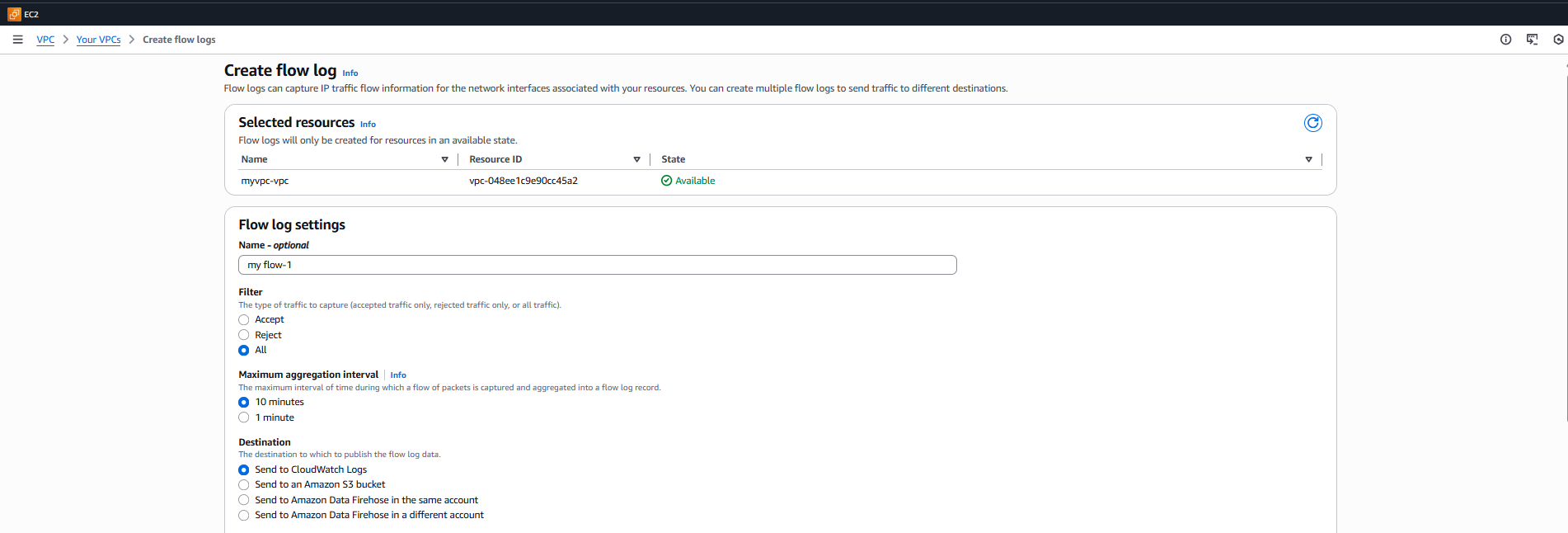


10. Configure VPC flow logs and store the logs in S3 and CloudWatch.

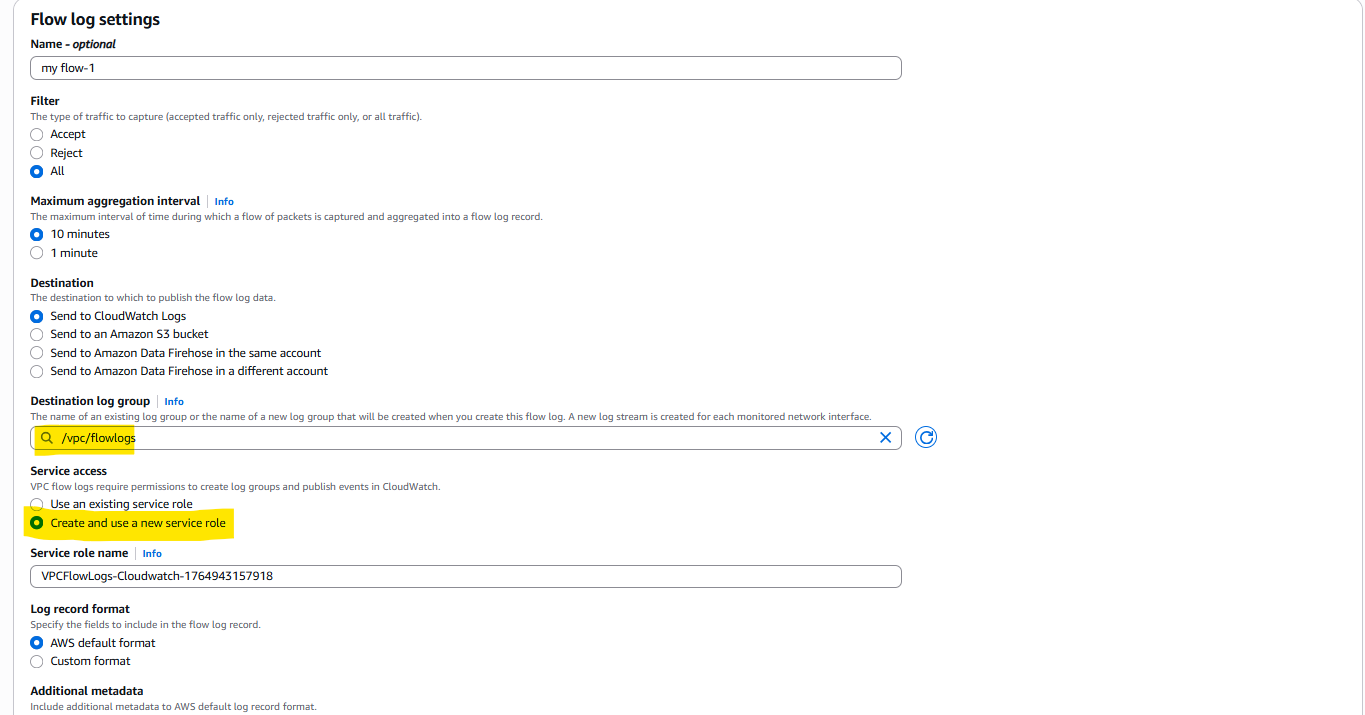
* Go to the VPC
* Select the vpc

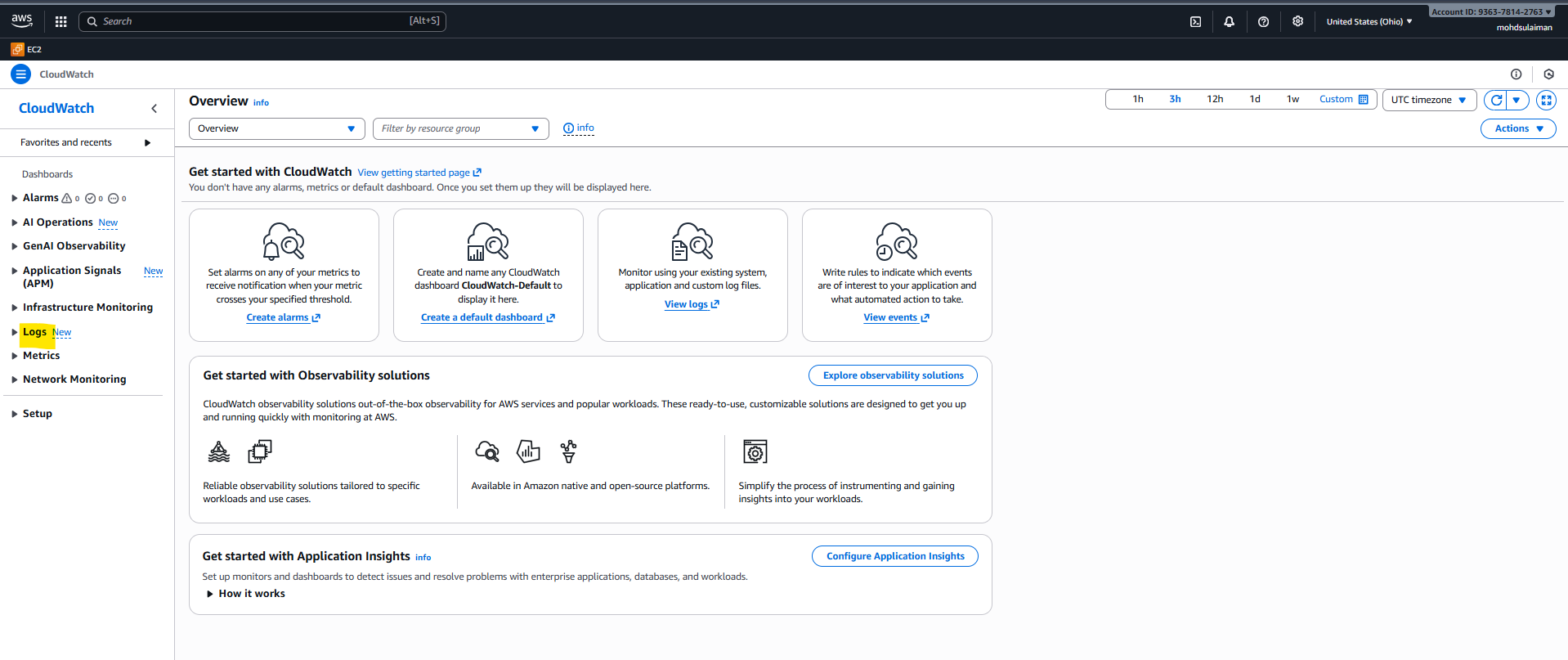


Now goes to inside create flow log

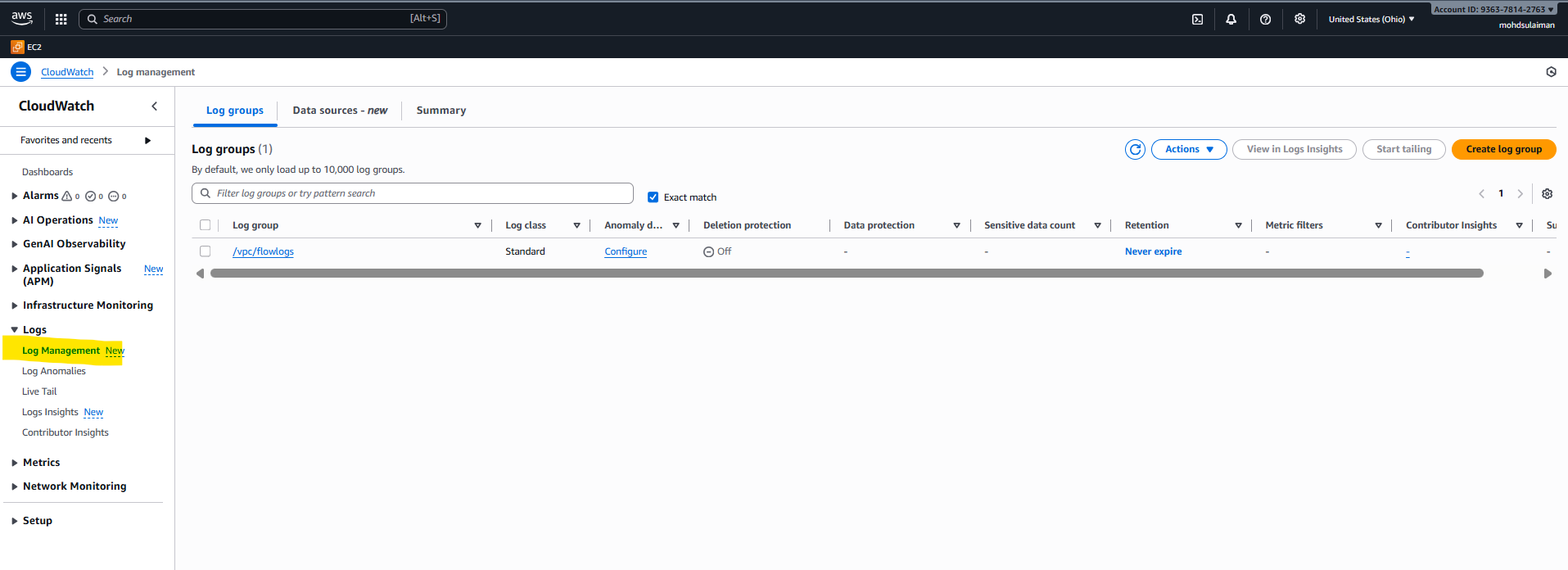


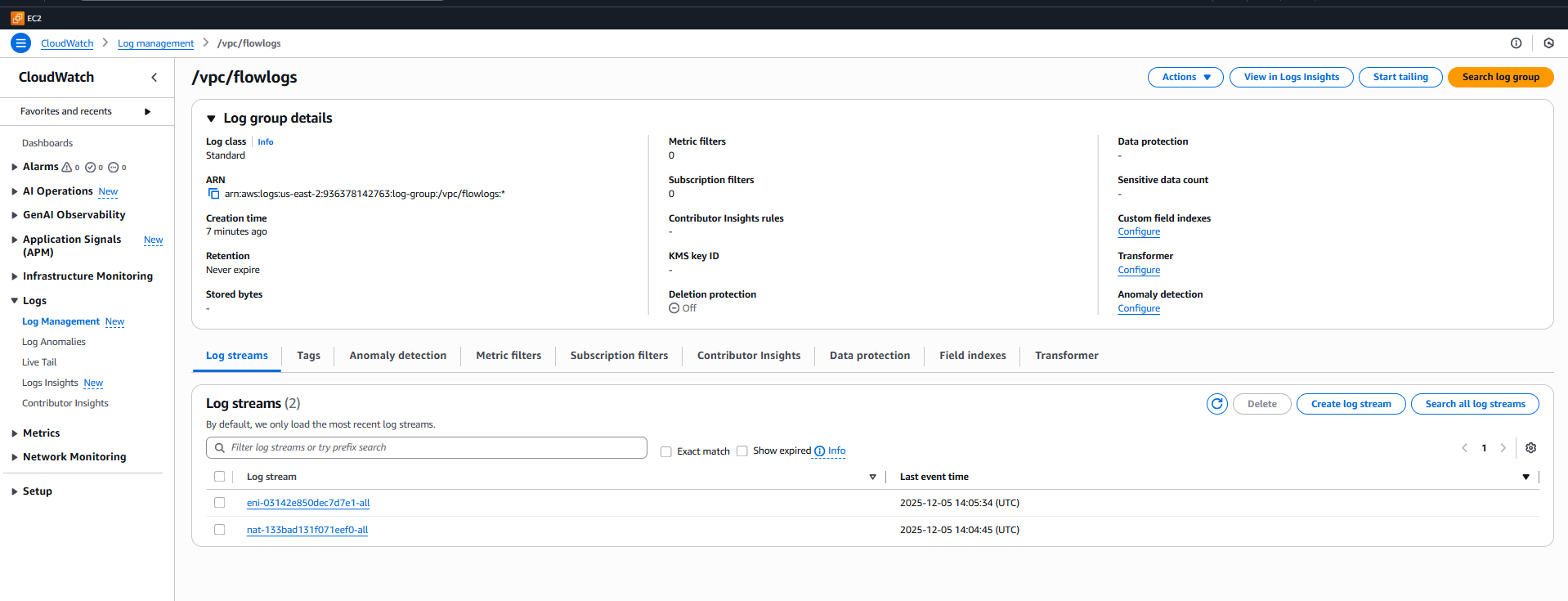
Now write name ( my flow-01)



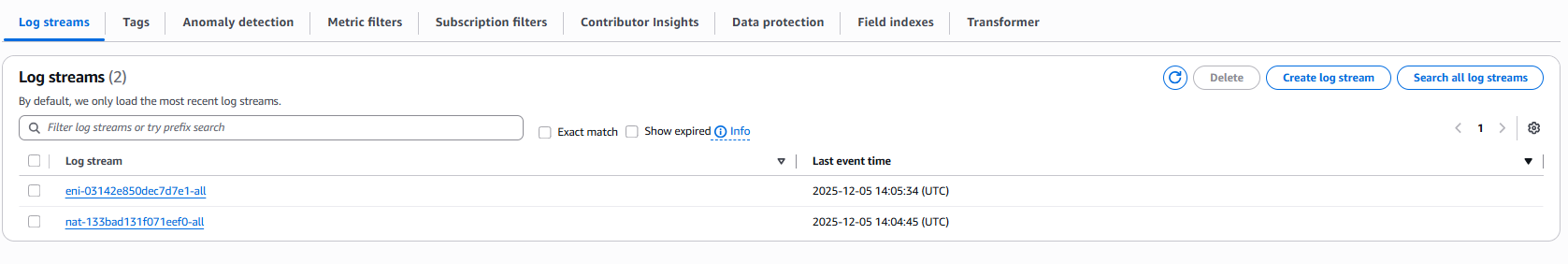


Go to the logs

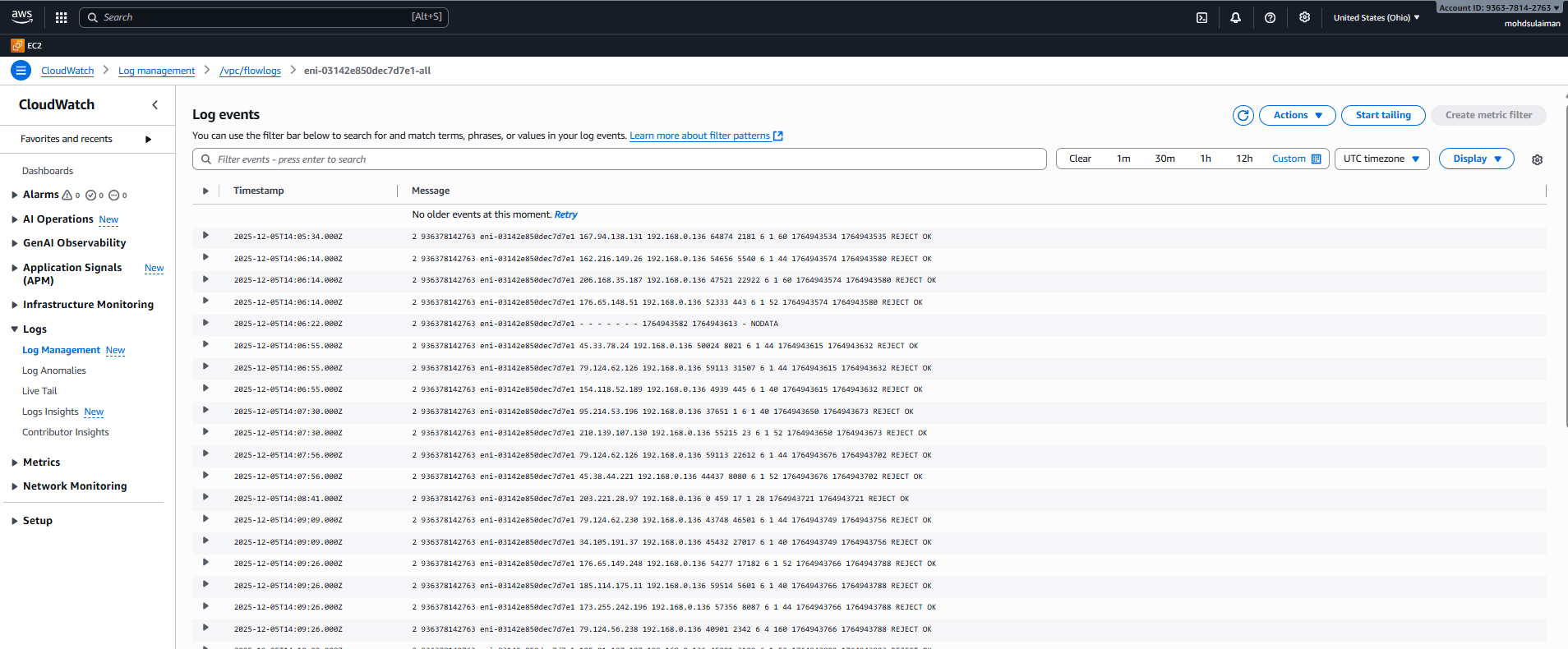




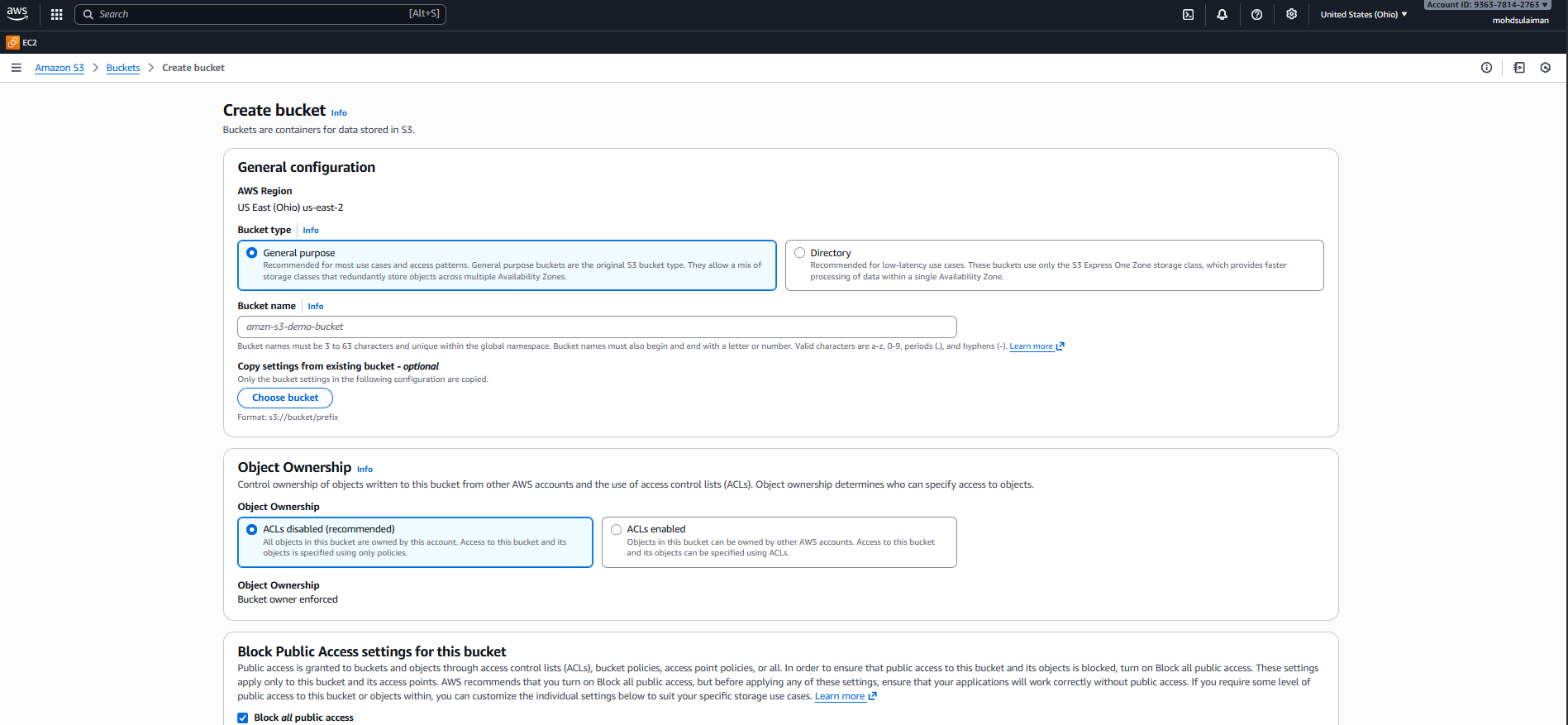
Choose any one

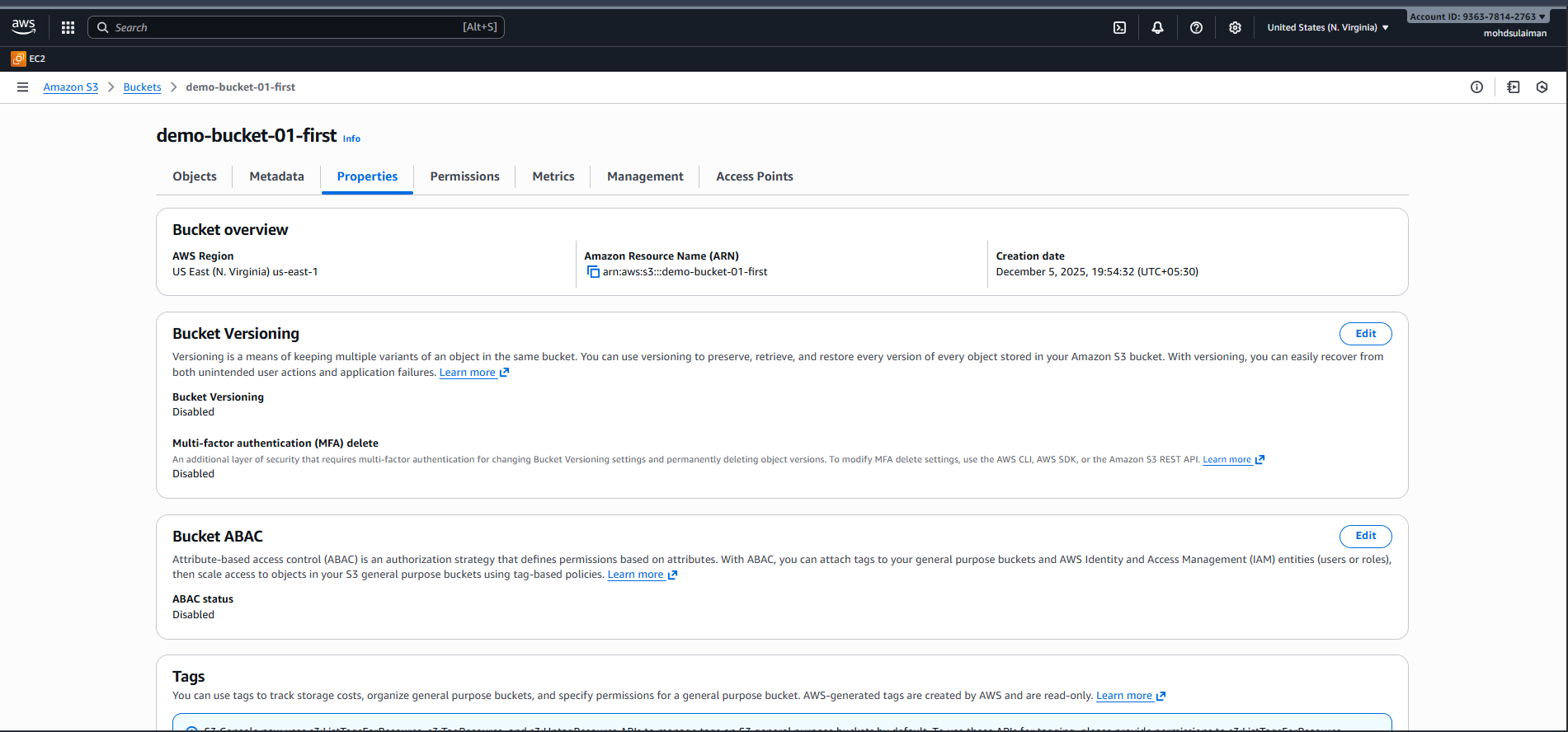


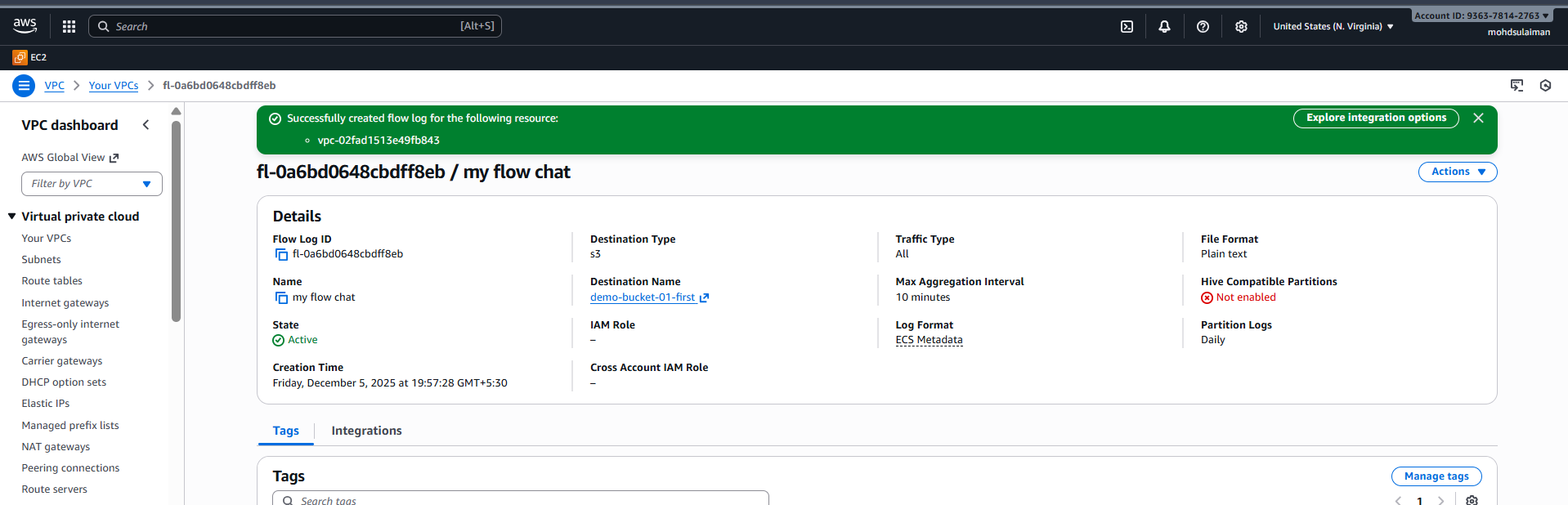
Hear the log events



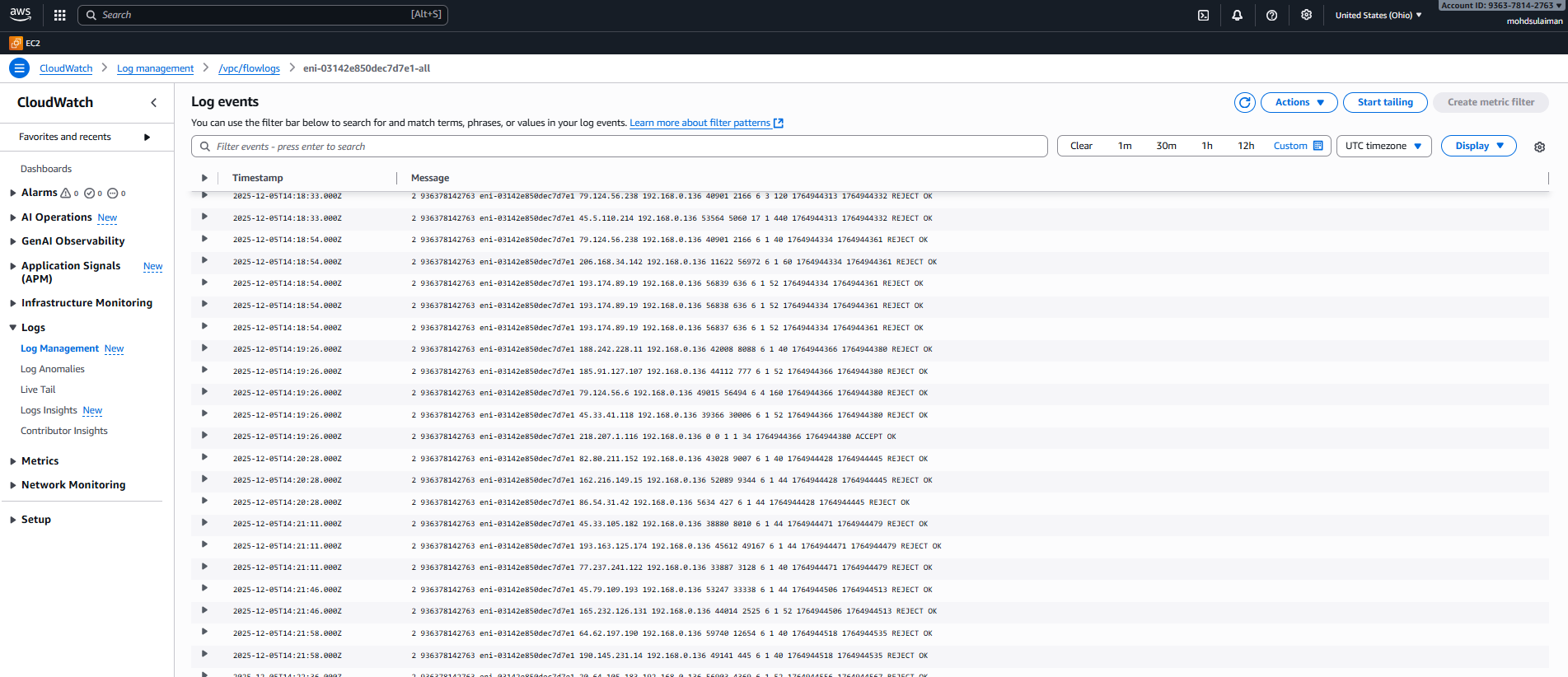
Lets create our bucket







Now successfully create bucket in our aws



Logs: