**VPC Task – 01**

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

**Create VPC**

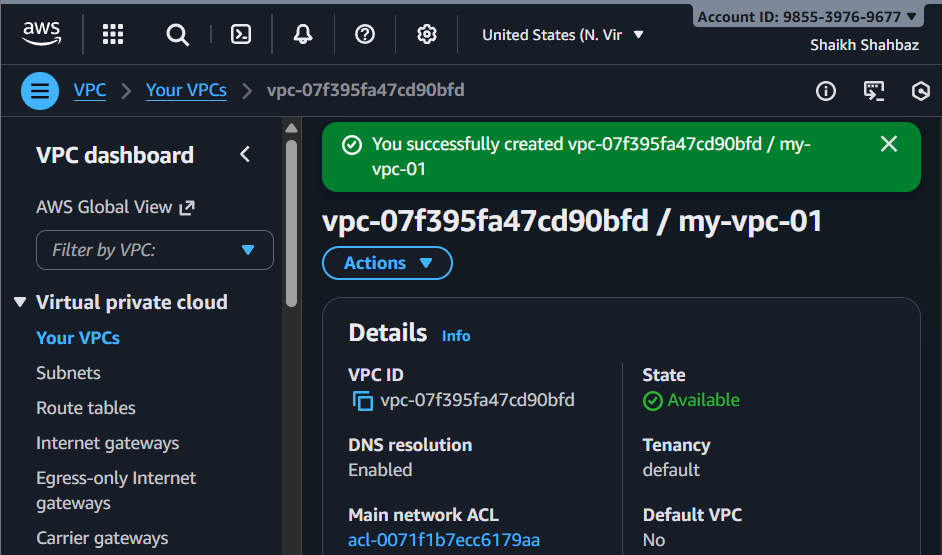
**Open AWS Console → VPC → Your VPCs → Create VPC**

**Select: VPC Only**

**Enter values:**

* + 1. **Name: my-vpc-01**
    2. **IPv4 CIDR: 172.168.0.0/16**
    3. **DNS hostnames Enable**
    4. **Leave others default**

**Click Create VPC**

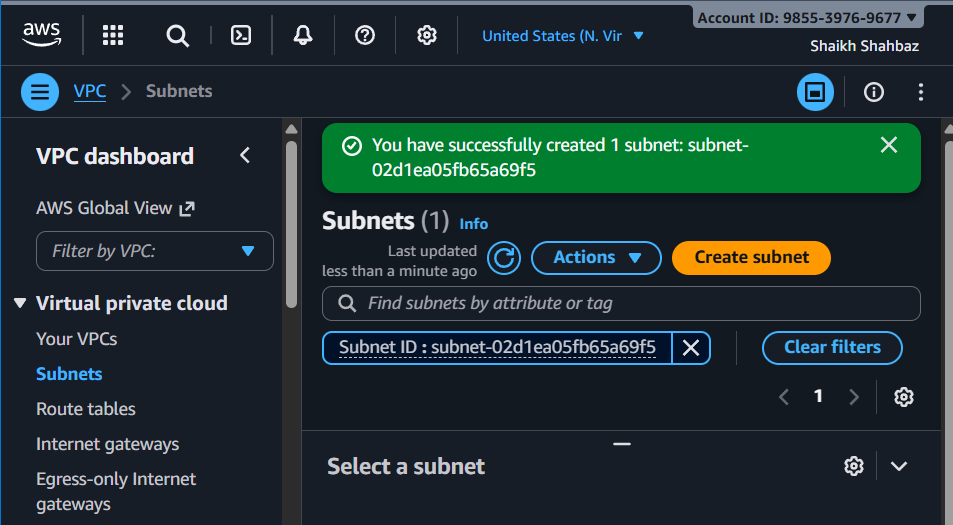
****

**Create Subnets**

**We will create 4 subnets inside the VPC.**

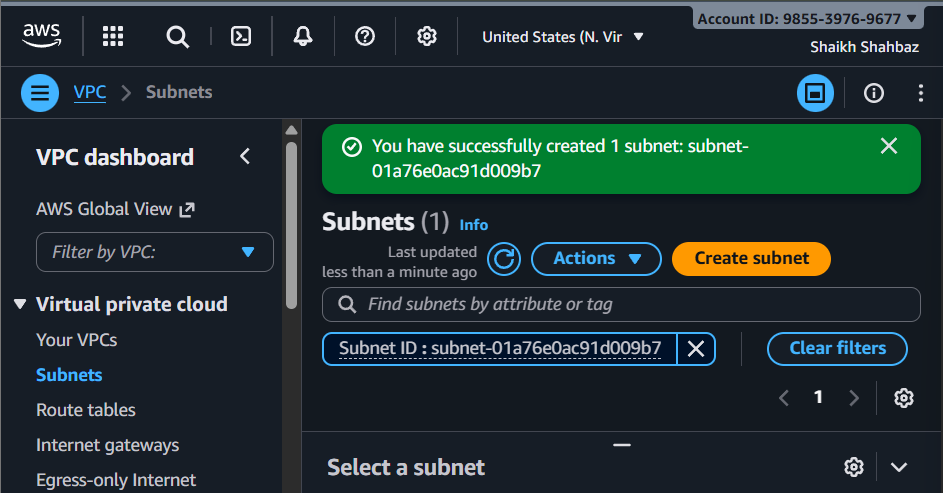
**🔹 Public Subnet 1**

* **Go to Subnets → Create subnet**
* **Select VPC: my-vpc-01**
* **Subnet name: Public-Subnet-01**
* **AZ: eu-east-1a**
* **IPv4 CIDR block: 172.168.0.0/26**
* **Click Create subnet**

****

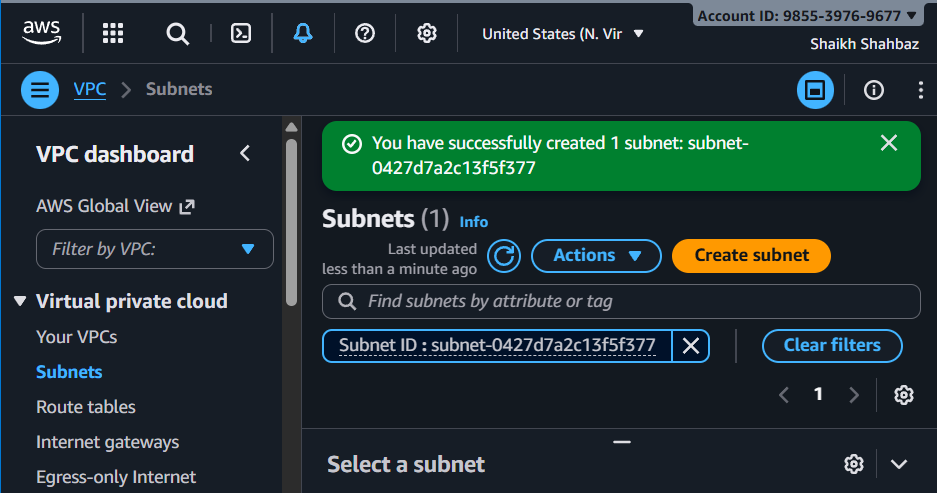
**🔹 Public Subnet 2**

* **Subnet name: Public-Subnet-02**
* **AZ: eu-east-1b**
* **CIDR: 172.168.0.64/26**
* **Create**

****

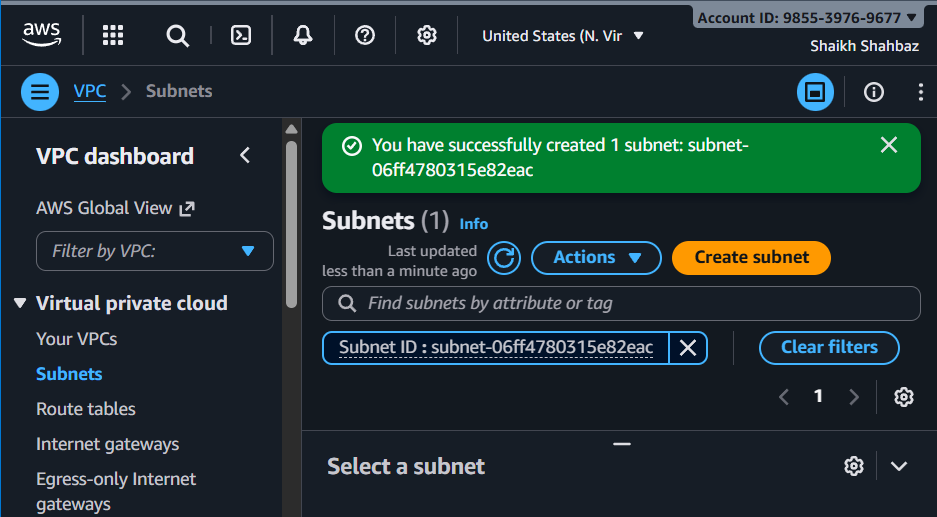
**🔹 Private Subnet 1**

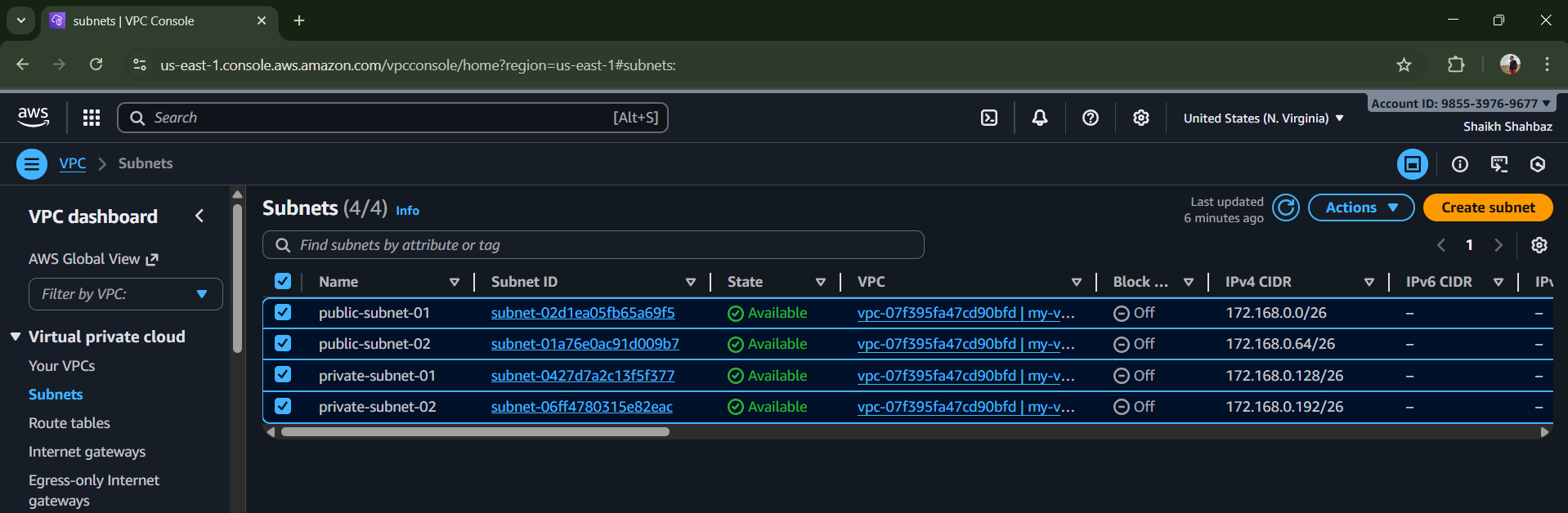
* **Subnet name: Private-Subnet-01**
* **AZ: eu-east-1a**
* **CIDR: 172.168.0.128/26**
* **Create**

****

**🔹 Private Subnet 2**

* **Subnet name: Private-Subnet-02**
* **AZ: eu-east-1b**
* **CIDR: 172.168.0.192/26**
* **Create**

****

****

1. **Enable DNS Hostname in VPC.**

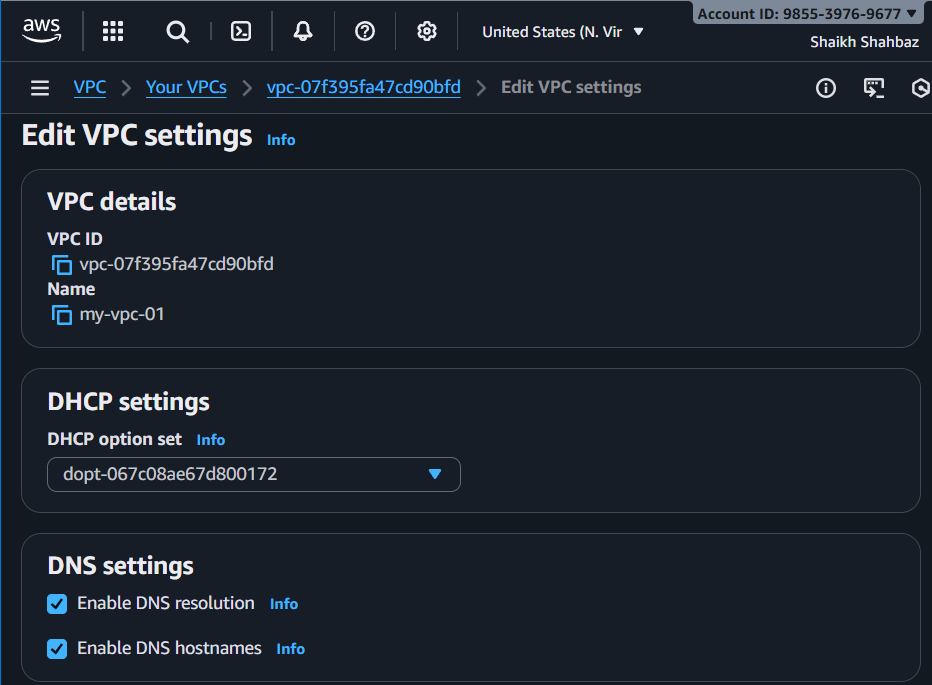
**Go to your VPCs**

**Select my-vpc-01**

**Click actions 🡪 Edit DNS hostnames**

**Enable : DNS Hostnames**

**Save changes**

****

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

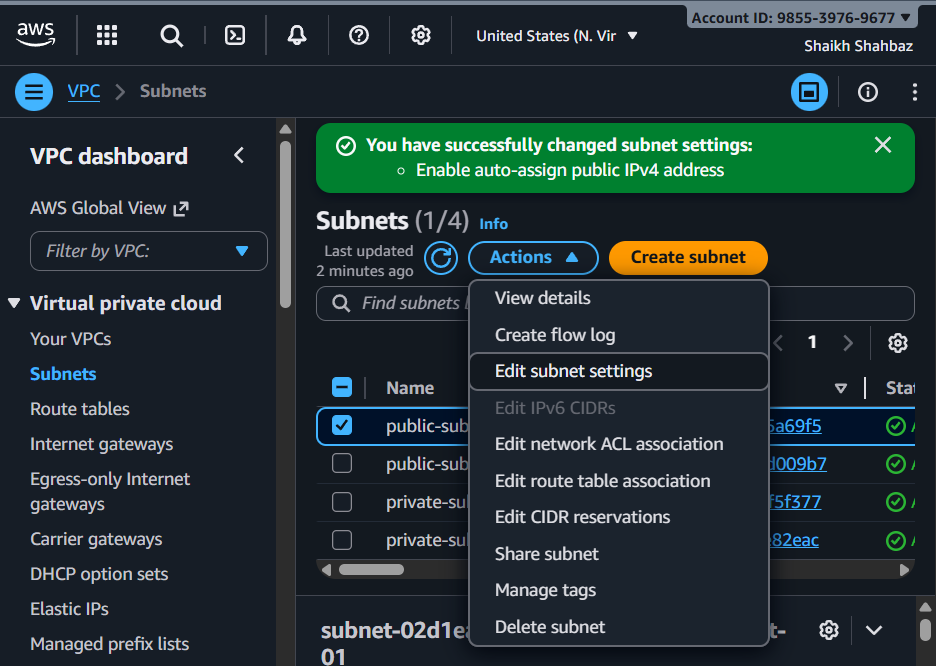
**Go to Subnets**

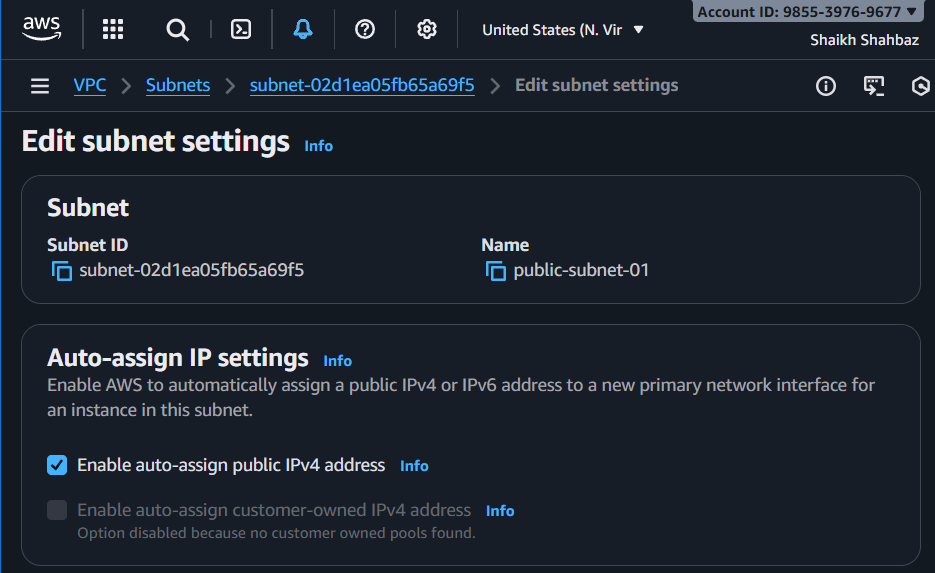
**Select public-subnet-01**

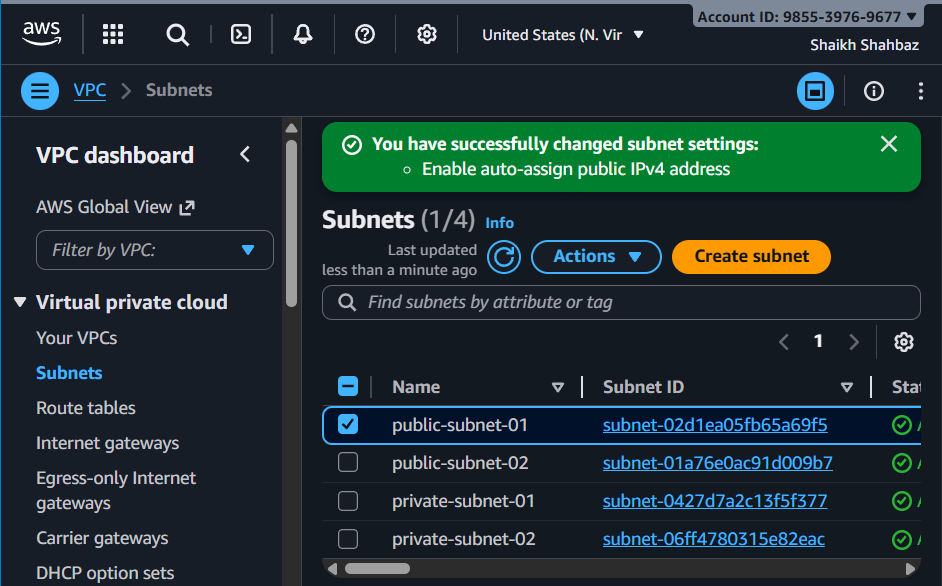
**Click Actions 🡪 Edit subnet settings**

**Enbale : Auto-assign IPv4 public IP**

**Save**

****

****

****

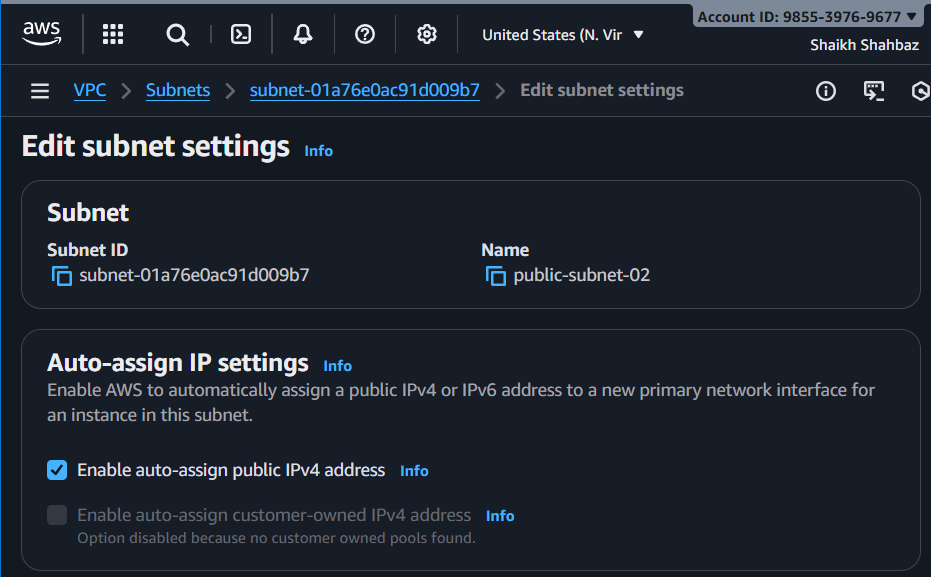
**Go to Subnets**

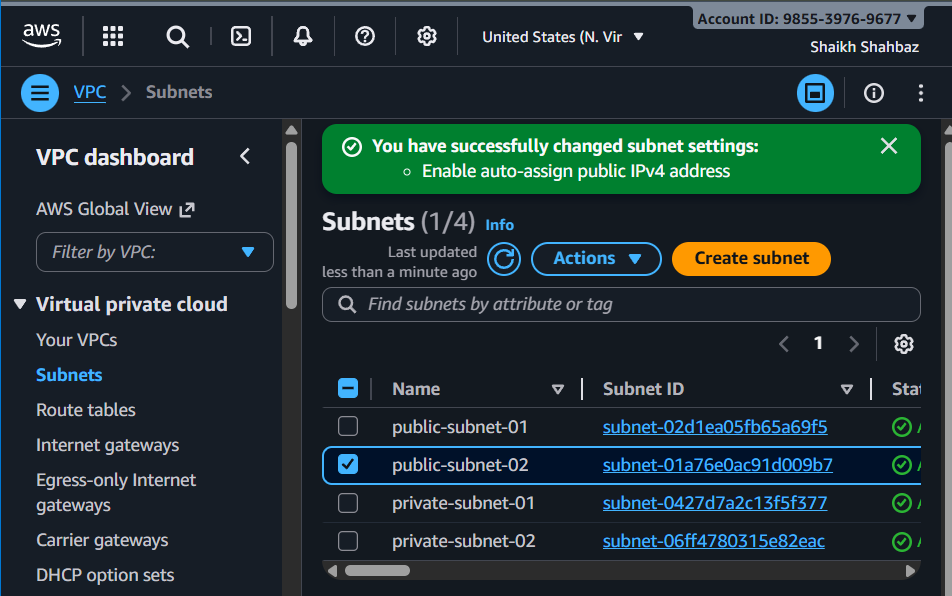
**Select public-subnet-01**

**Click Actions 🡪 Edit subnet settings**

**Enbale : Auto-assign IPv4 public IP**

**Save**

****

****

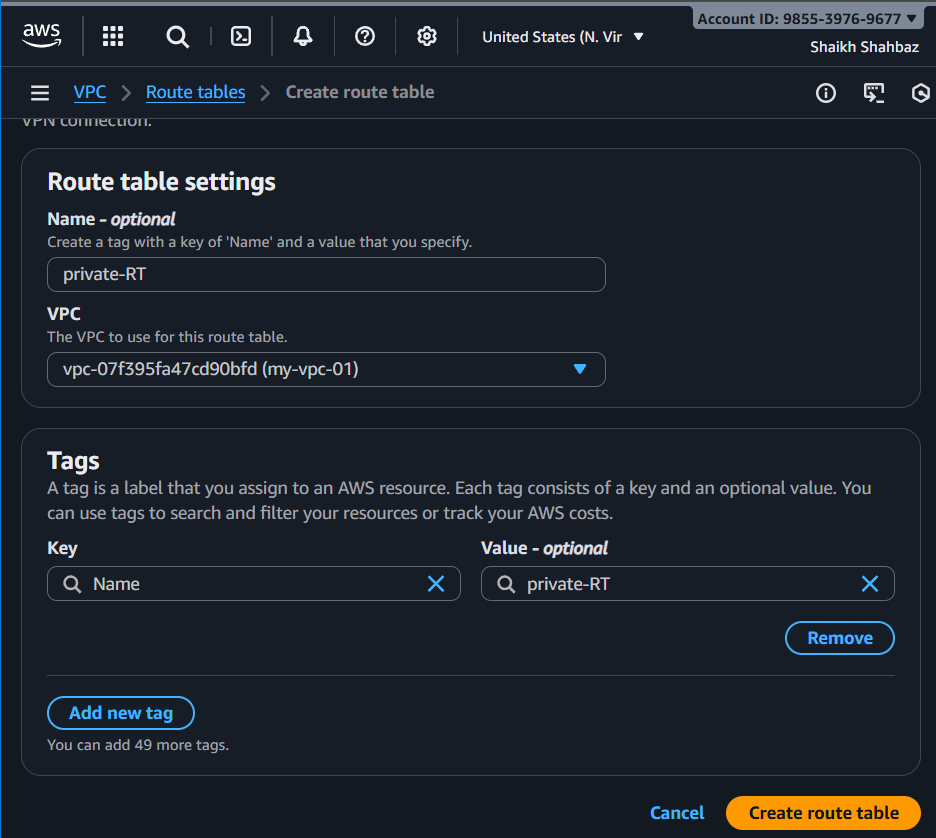
1. **Add 2 private subnets in private route table.**

**Go to Route Tables 🡪 Create route table**

**Name : private-RT**

**Choose VPC : my-vpc-01**

**Create**

****

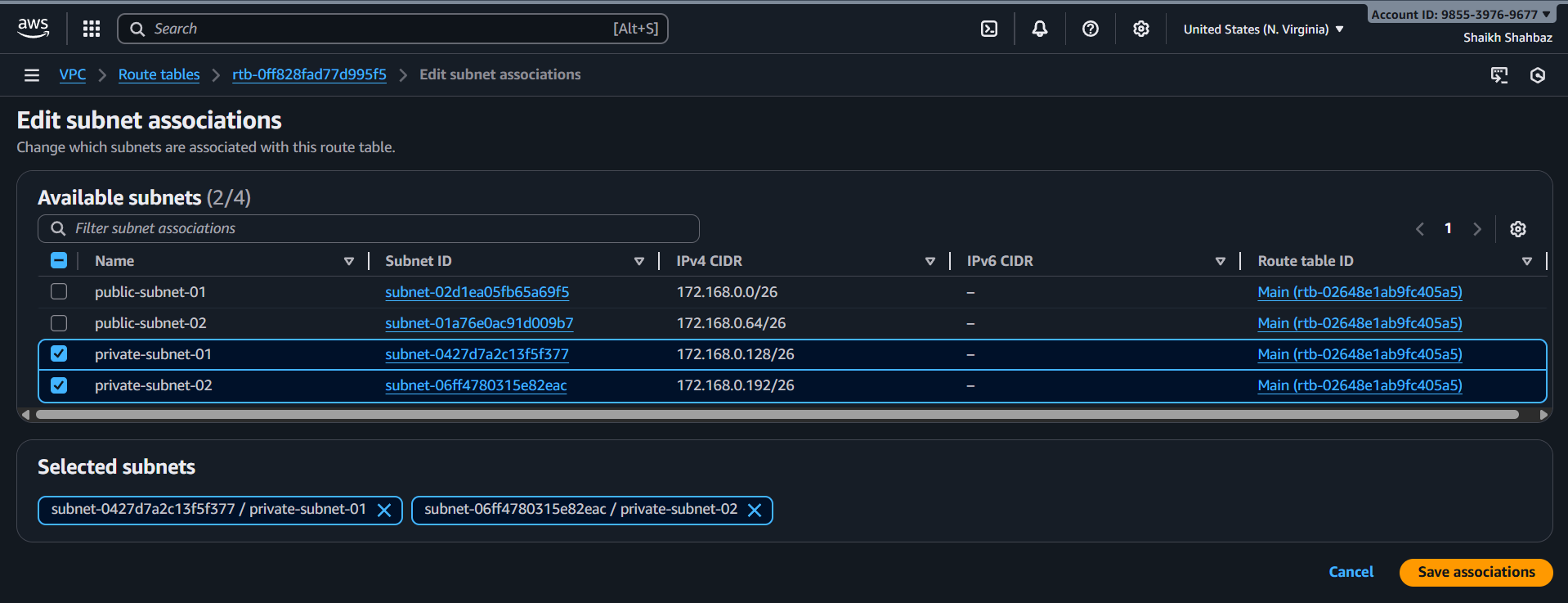
**🡪 Now associate**

**Select private-RT 🡪 Subnet association 🡪 Edit**

**Select : private-subnet-01**

**Private-subnet-02**

**Save association**

****

****

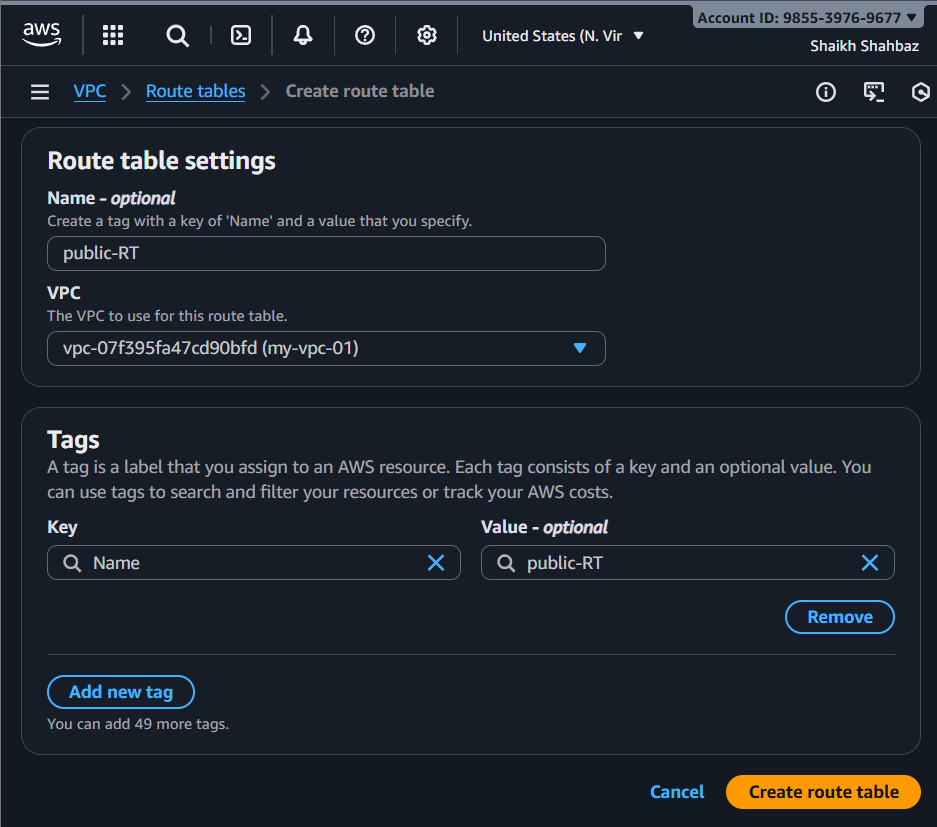
1. **Add 2 public subnets in public route table.**

**Route Tables 🡪 Create route table**

**Name : public-RT**

**VPC : my-vpc-01**

**Create**

****

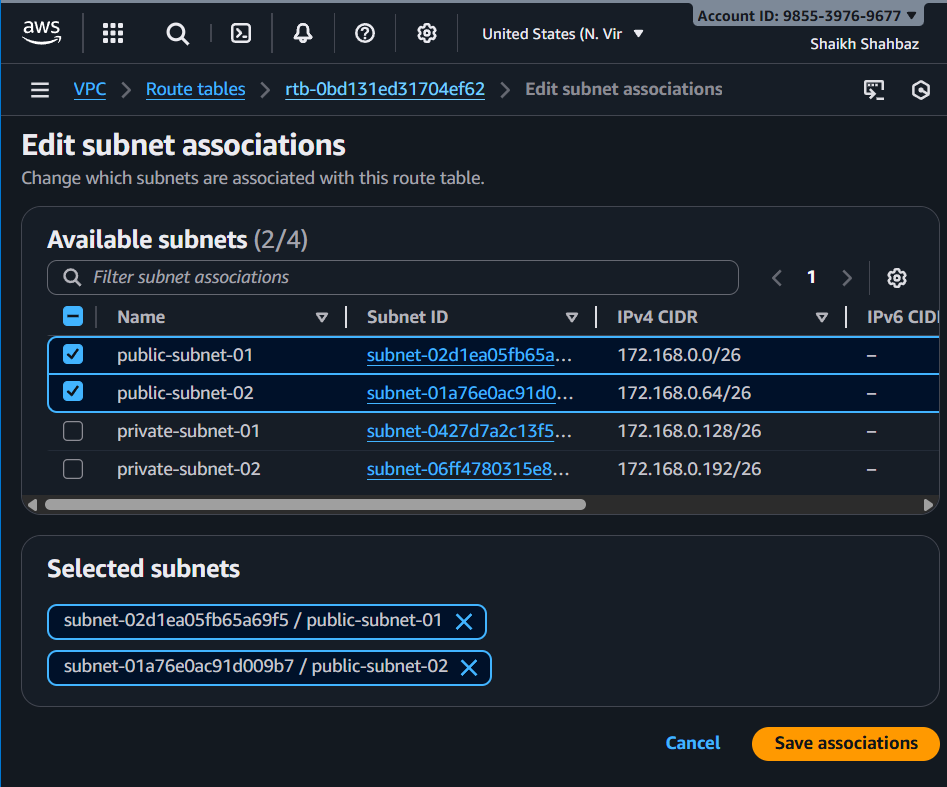
**🡪 Associate**

**Select public-RT 🡪 Subnet association 🡪 Edit**

**Select : public-subnet-01**

**Public-subnet-02**

**Save**

****

****

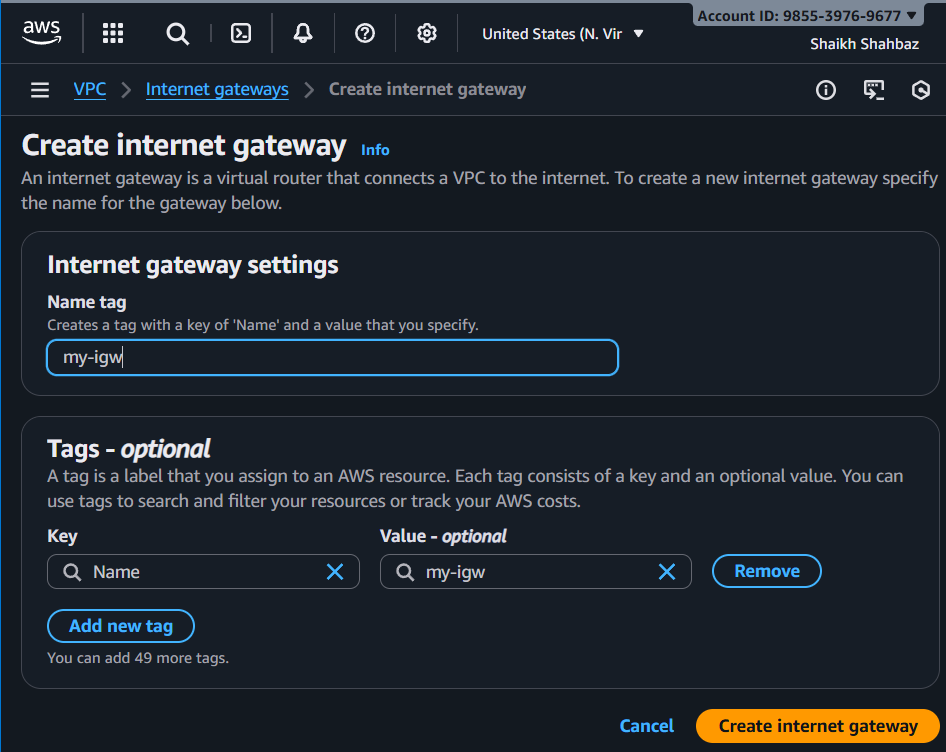
1. **Public route table will have the routes to internet and local.**

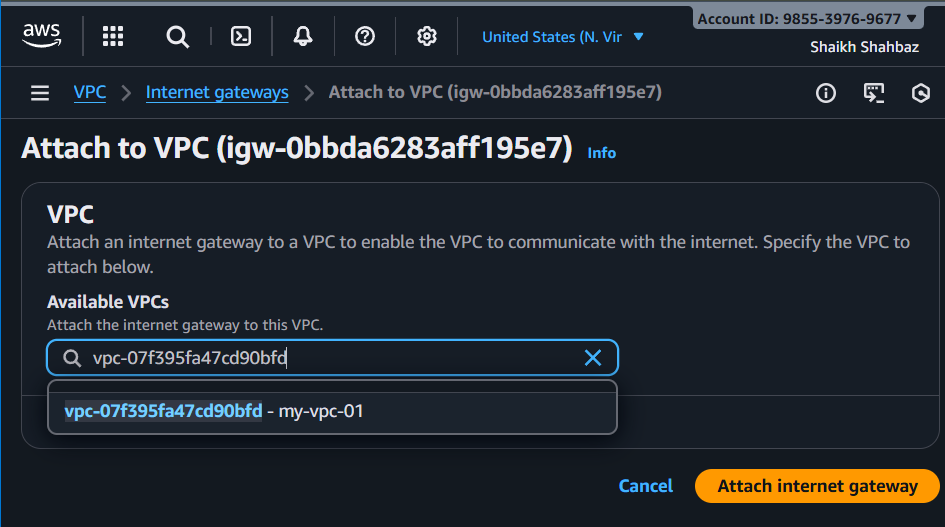
**Go to Internet Gateways 🡪 Create IGW**

**Name : my-igw**

**Create 🡪 Actions 🡪 Attach to VPC**

**Select my-vpc-01**

****

****

**🡪 Add route to Public Route Table**

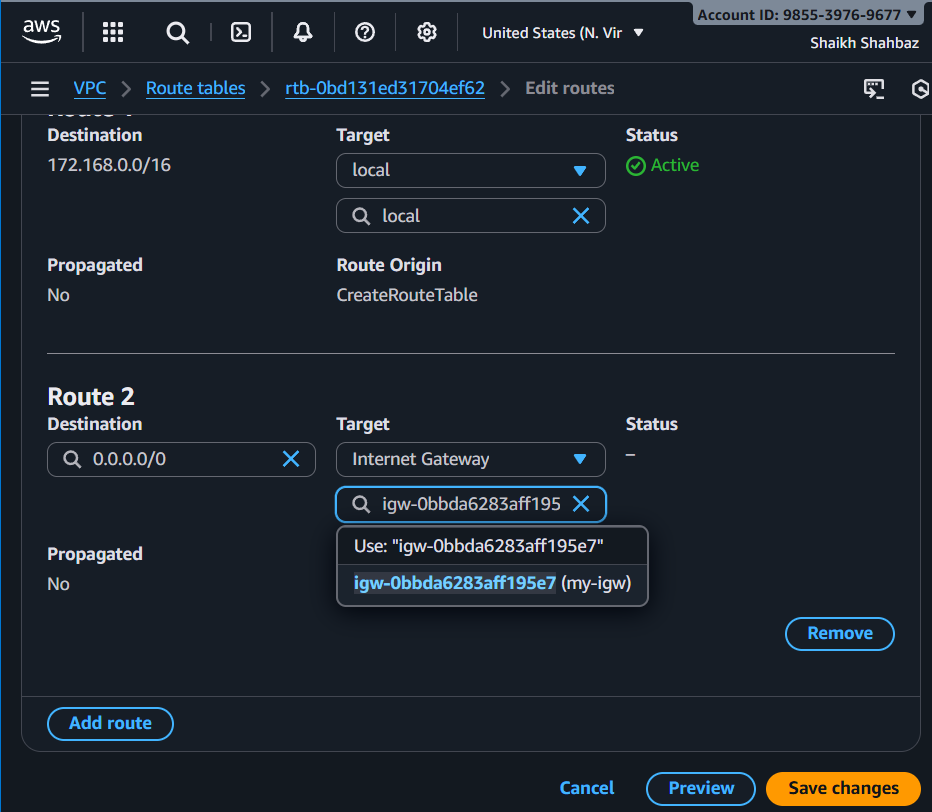
**Select public-RT**

**Routes 🡪 Edit routes**

**Add : Destination : 0.0.0.0/0**

**Target : Internet Gateway (my-igw)**

**Save**

****

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

**Go to ec2 🡪 Launch Instance**

**Name : public-server**

**AMI : Amazon Linux 2**

**Instance type : t2.micro**

**Key pair : select existing (avengers)**

**Network :**

**VPC : my-vpc-01**

**Subnet : public-subnet-01**

**Auto-assign public IP : Enabled**

**Security group :**

**Allow SSH (22)**

**Allow HTTP (80)**

**User data :**  #!/bin/bash

yum update -y

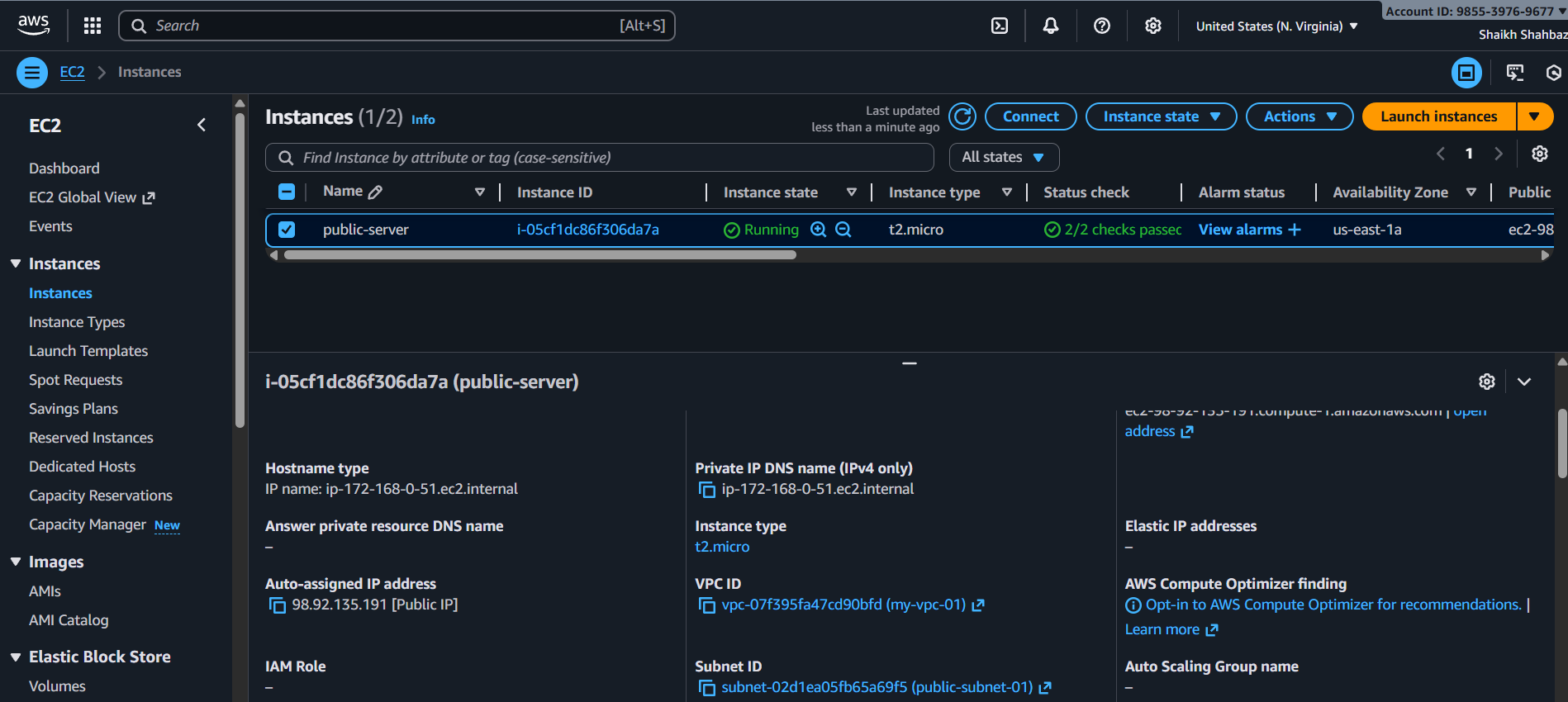
yum install -y httpd php

systemctl start httpd

systemctl enable httpd

echo "<?php phpinfo(); ?>" > /var/www/html/index.php

**Launch Instance**

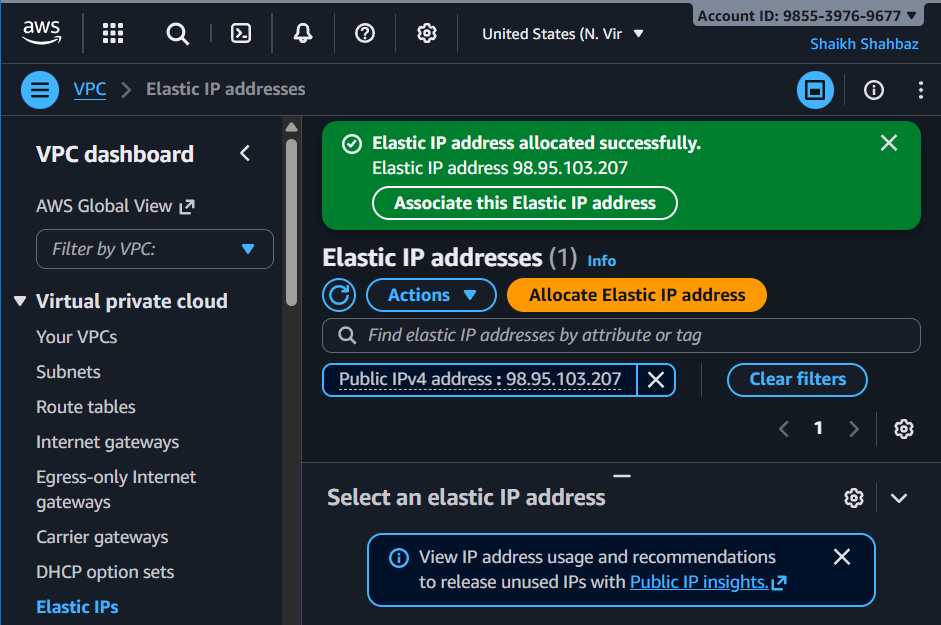
****

**Test in browser : https://** **98.92.135.191**

****

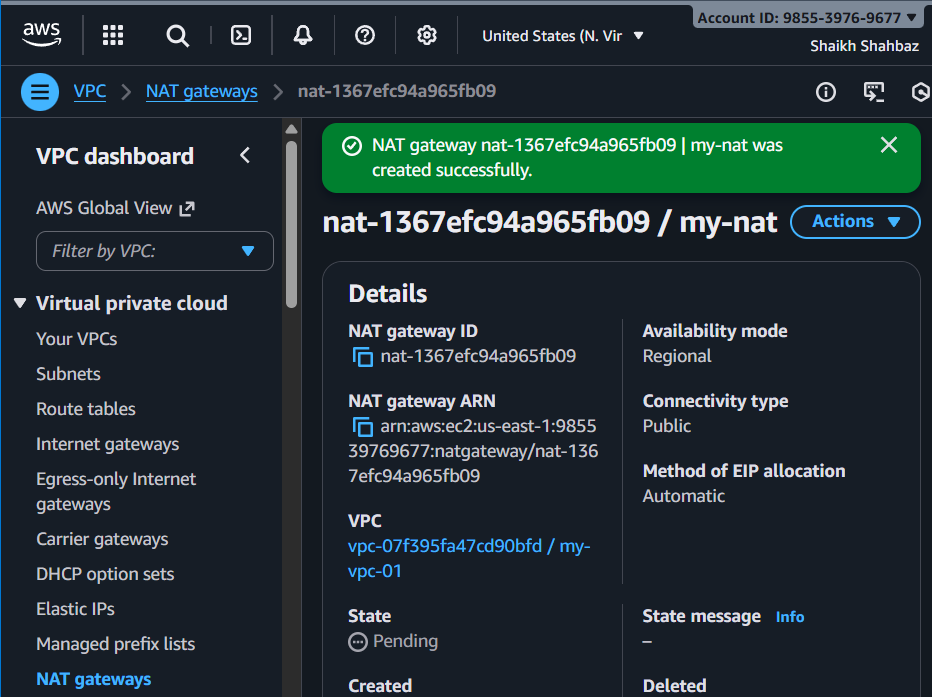
1. **Configure NAT gateway in public subnet and connect to private instance.**

**Go to Elastic IPs 🡪 Allocate Elastic IP**

****

**Go to NAT Gateway 🡪 Create NAT Gateway**

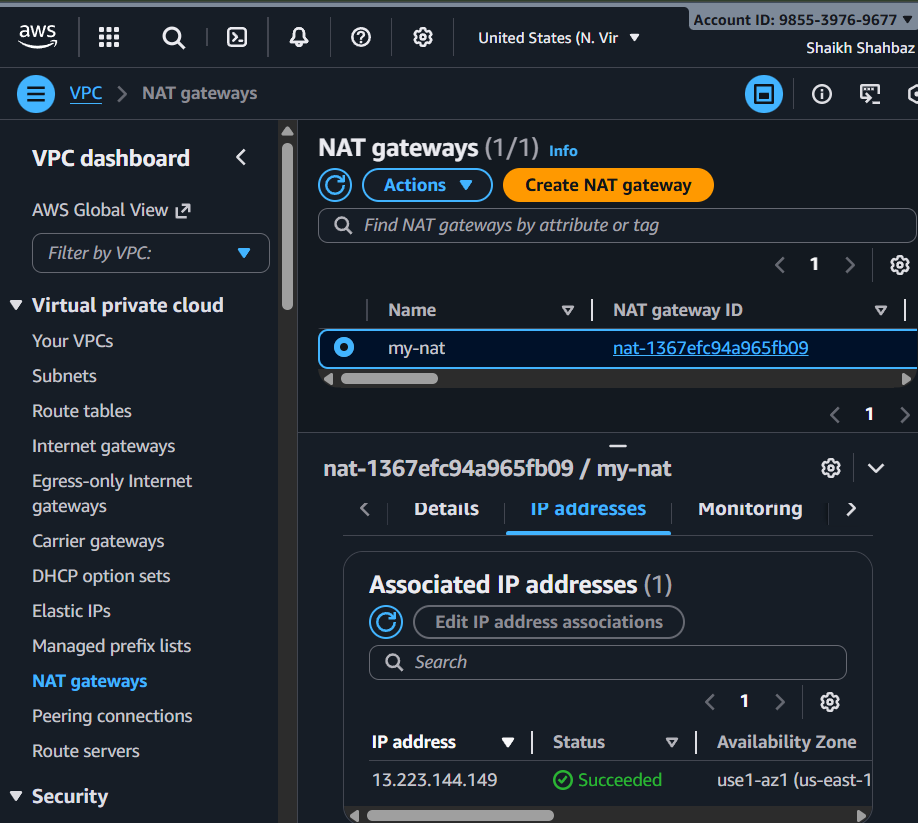
**Name : my-nat**

****

**Subnet : public-subnet-01**

**Elastic IP : Select allocated**

**Create**

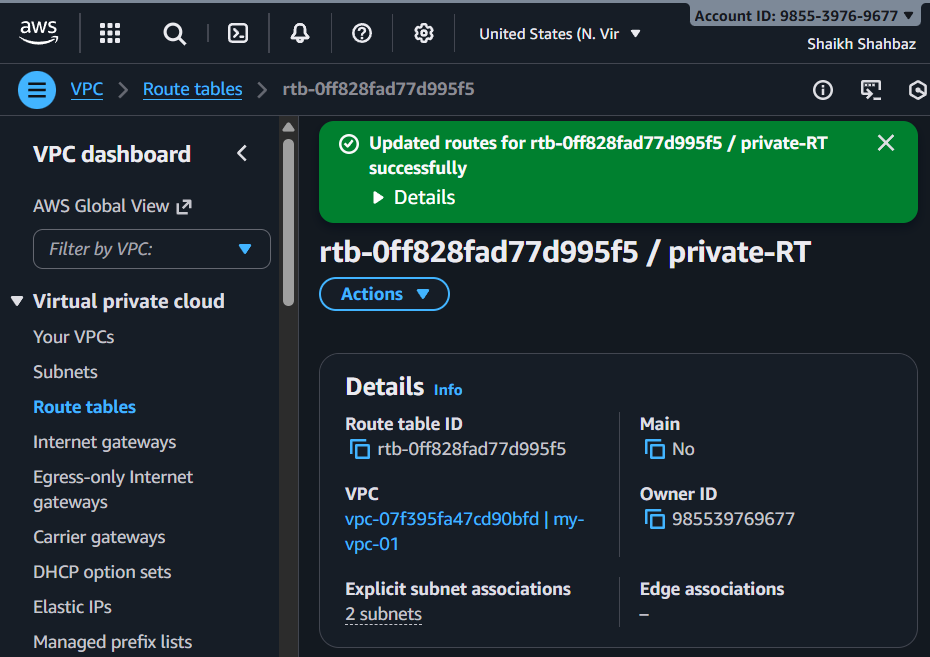


**🡪 Update Private Route Table**

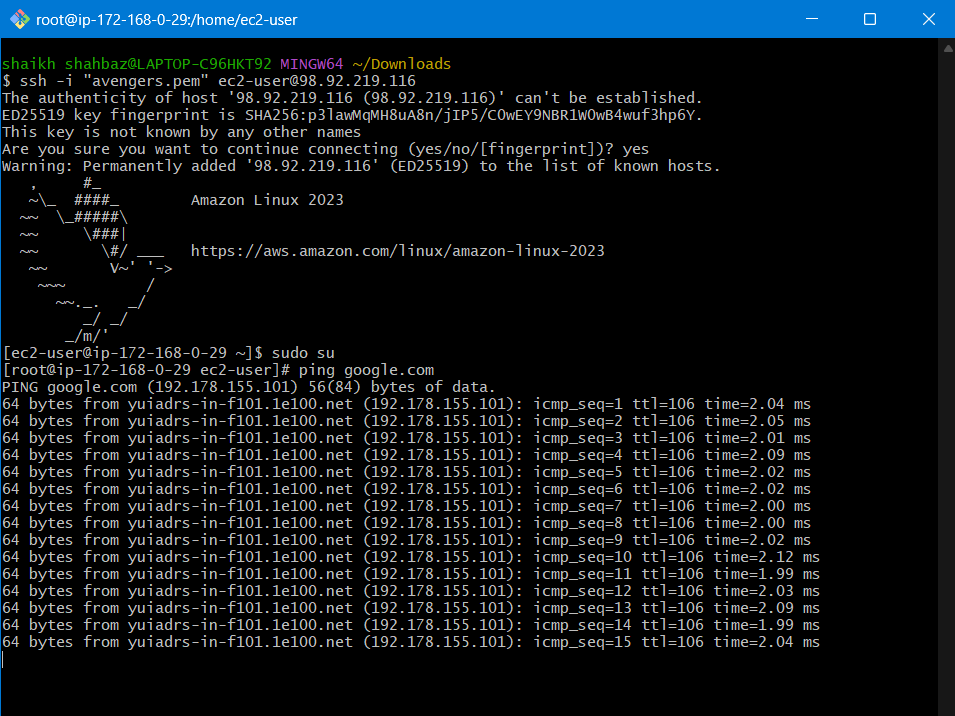
**Route Tables 🡪 private-RT**

**Routes 🡪 Edit routes 🡪 Add : Destination : 0.0.0.0/0**

**Target : NAT Gateway 🡪 my-nat**

****

**Now private servers can access the internet**

****

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

**🡪 Launch Private ec2 (Tomcat Server)**

**Ec2 🡪 Launch instance**

**Name : private-tomcat**

**AMI : Amacon Linux 2**

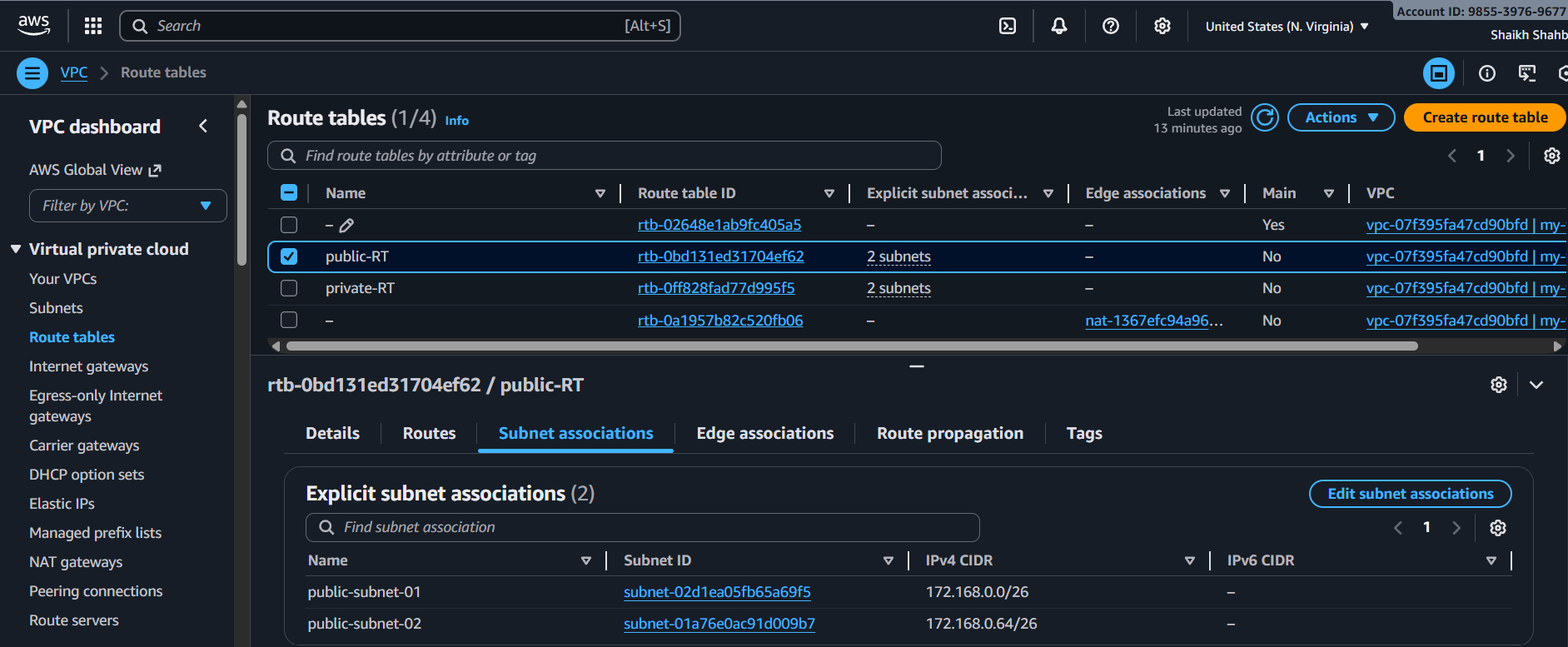
**Instance type : t2.micro**

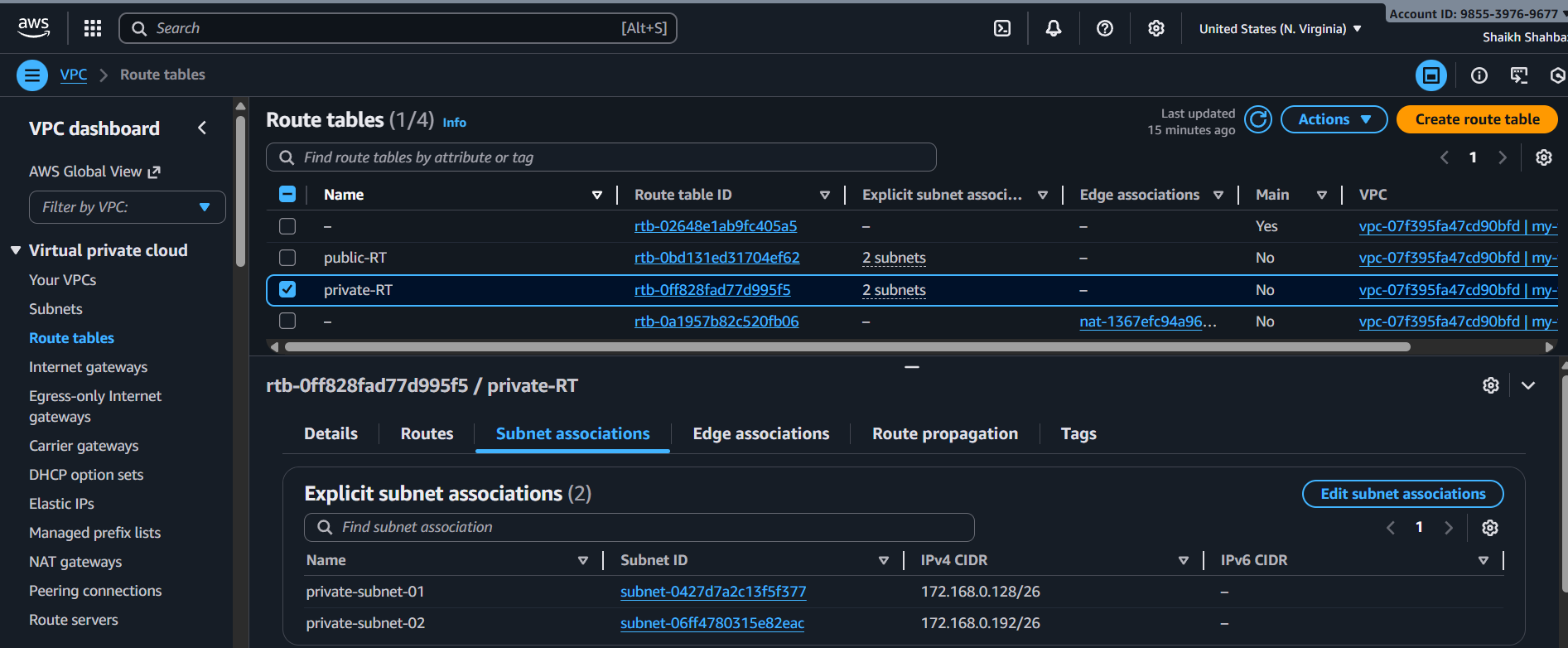
**Key pair : avengers.pem (select exixting)**

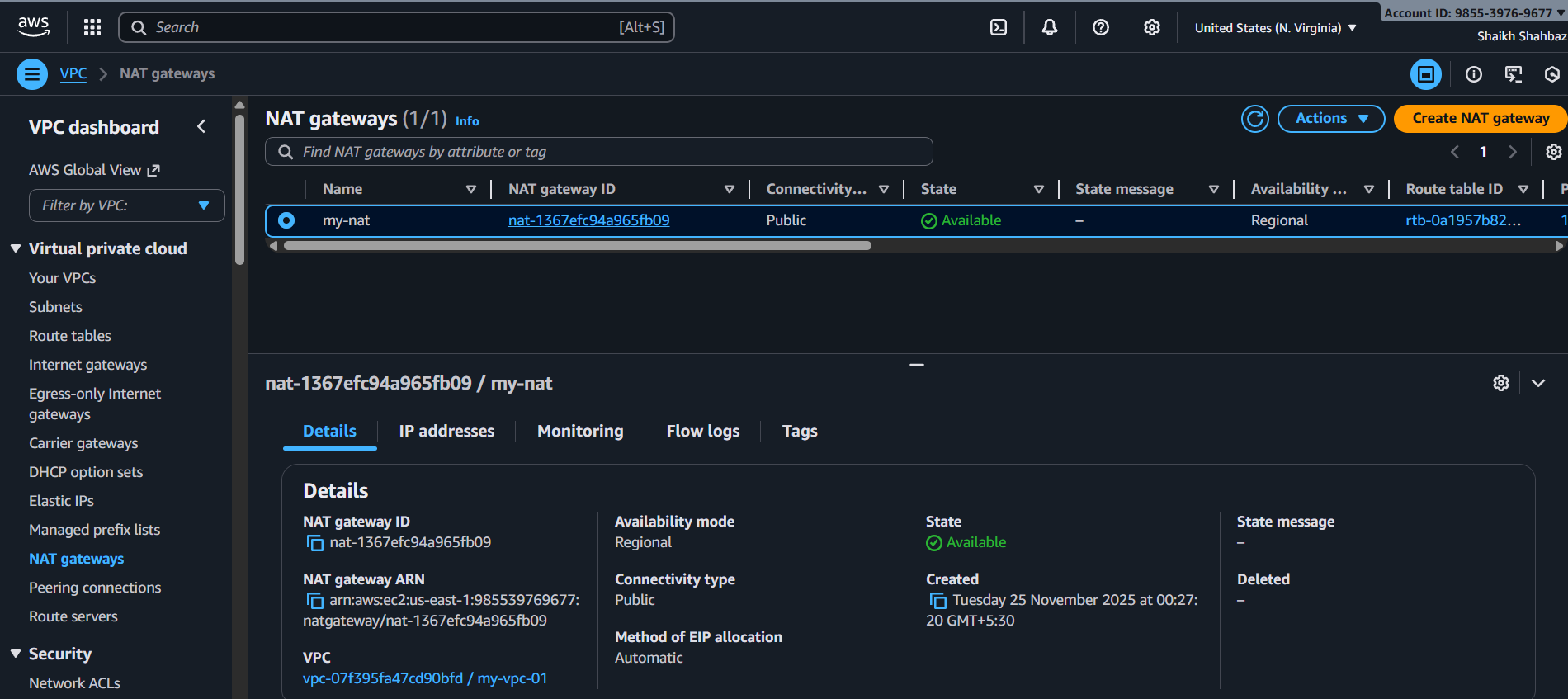
**VPC : my-vpc-01**

**Subnet : public-subnet-01**

**Auto Assign Public IP : Disabled**

****

****

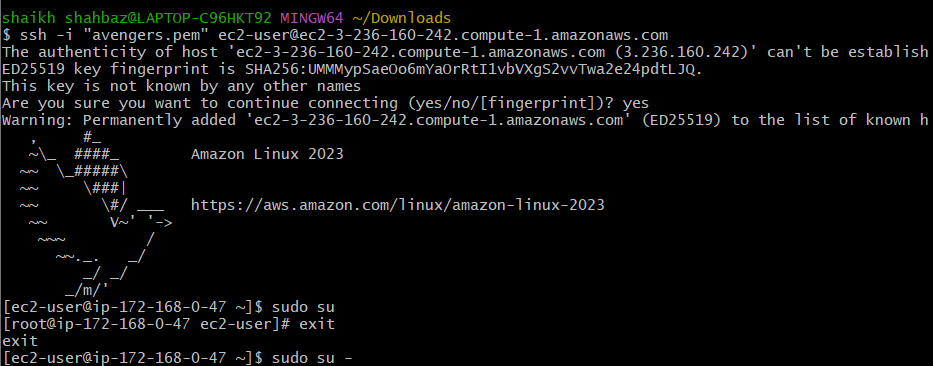
****

**Security Group : SSH only from internal VPC (10.0.0.0/16)**

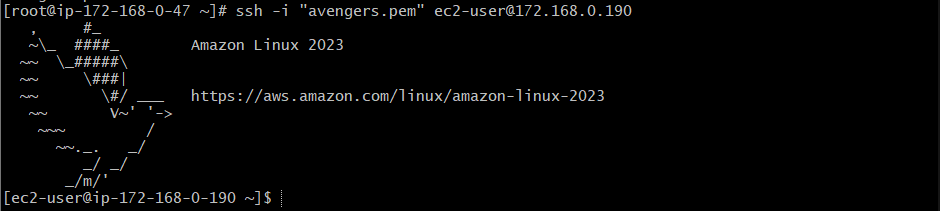
**Tomcat Port (8080) allow internal network only**

**🡪 SSH into Private ec2 via Public ec2**

**From laptop 🡪 ssh into public server : ssh -I “avengers.pem” ec2-user@<public-ip>**

****

**Then from public to private : ssh -I “avengers.pem” ec2-user@<private-ip>**

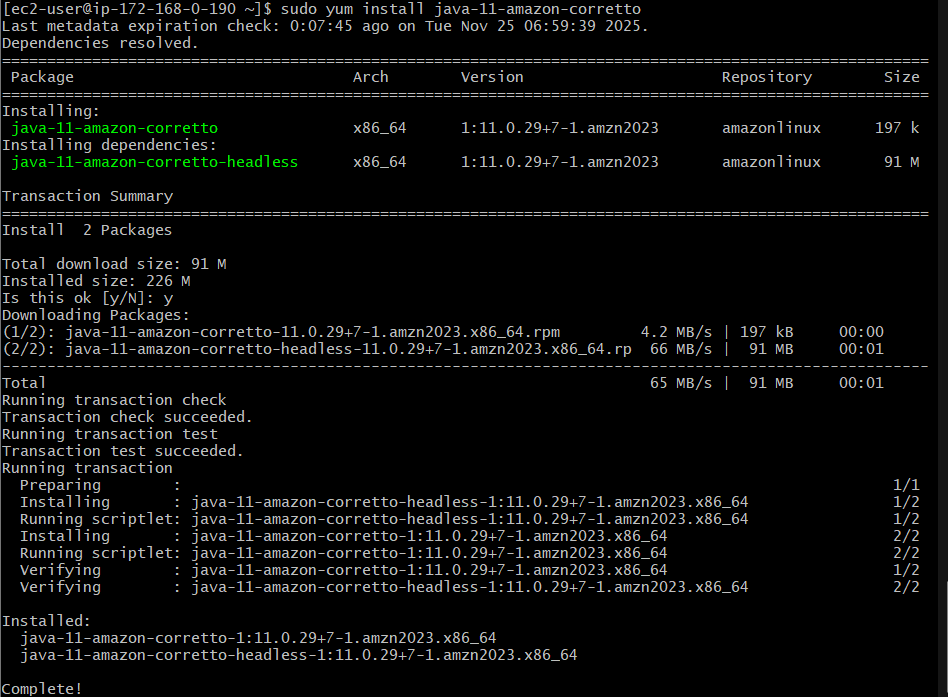
****

**🡪 Install java and tomcat on private ec2**

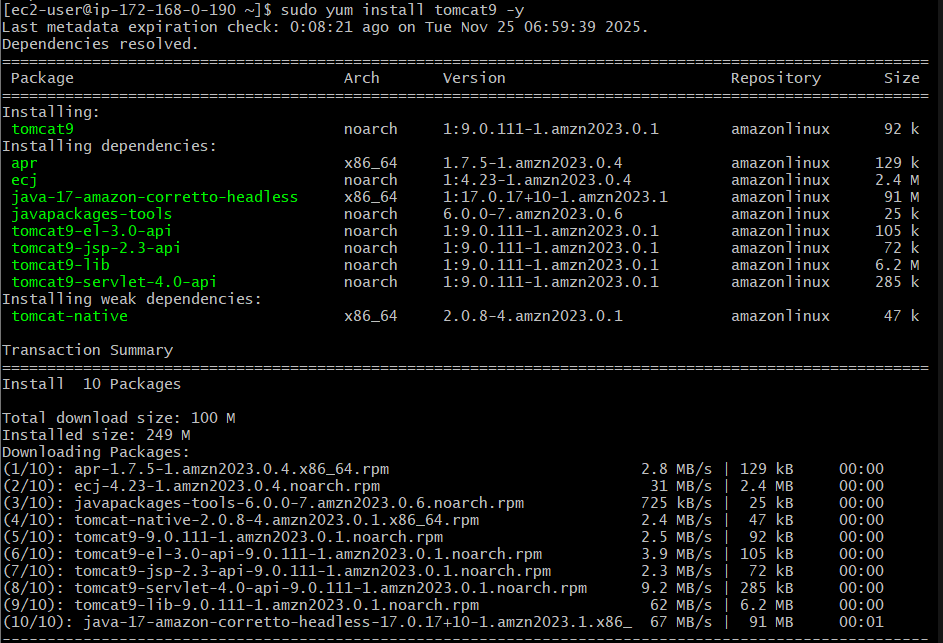
sudo yum update -y



sudo yum install -y java-11-amazon-corretto



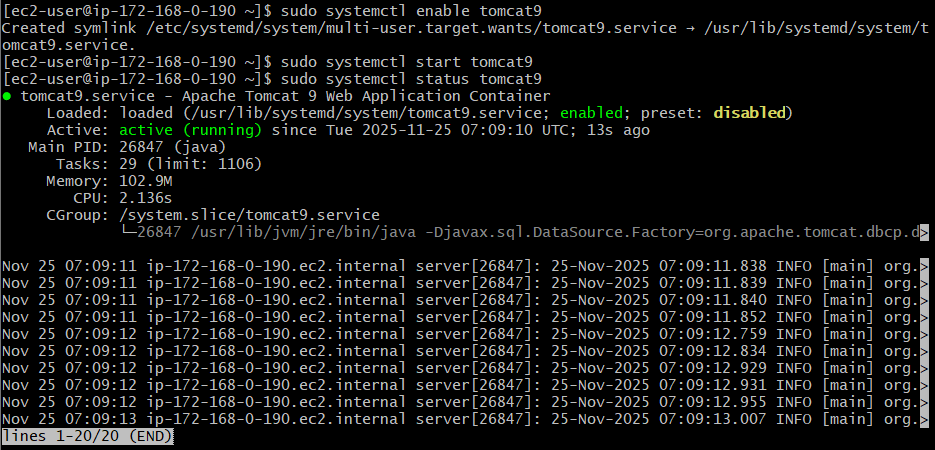
sudo yum install tomcat9 -y



sudo systemctl enable tomcat9

sudo systemctl start tomcat9

sudo systemctl status tomcat9

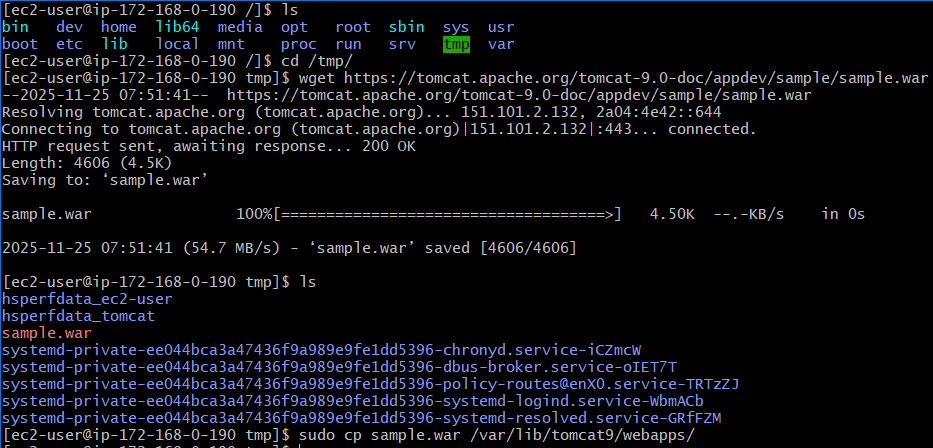


**🡪 Deploy sample web app in tomcat :**

**cd /tmp**

**wget** [**https://tomcat.apache.org/tomcat-9.0-**](https://tomcat.apache.org/tomcat-9.0-)**doc/appdev/sample/sample.war**

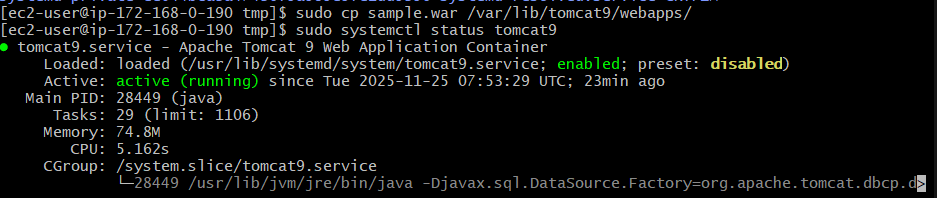
sudo cp sample.war /var/lib/tomcat9/webapps/

****

**Restart tomcat :**

**sudo systemctl restart tomcat9**

**sudo systemctl status tomcat9**

****

**Test deployment : curl** [**http://localhost:8080/**](http://localhost:8080/)sample

**Output : Welcome to Private Tomcat Server**

****

**🡪 Test NAT Gateway Internet Access : ping -c 4 google.com**

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

**Create two flow logs for your VPC:  
🡪 One that sends logs to CloudWatch Logs  
🡪 One that sends logs to S3 Bucket**

**01. Create CloudWatch Log Group**

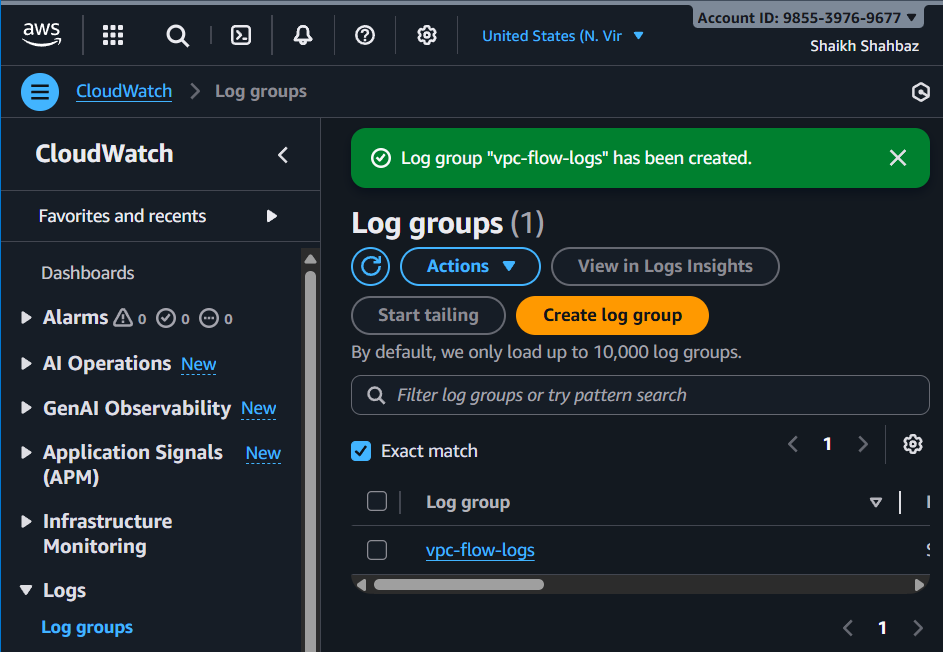
**Before creating the flow log, CloudWatch needs a log group.**

**🡪 Open CloudWatch**

* **Go to AWS Console**
* **Search CloudWatch**
* **Select CloudWatch**

**🡪 Create Log Group**

1. **In left menu → choose Logs → Log groups**
2. **Click Create log group**
3. **Enter:**
   * **Log group name: vpc-flow-logs**
4. **Click Create**

****

**02. Create S3 Bucket for Flow Logs**

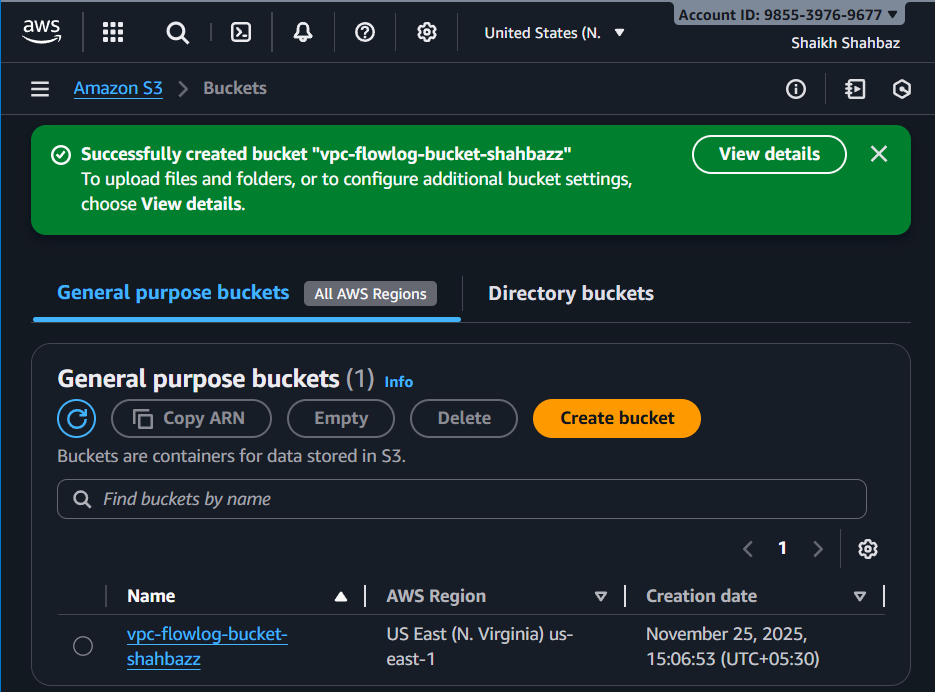
**Now create an S3 bucket to store raw flow log files.**

**🡪 Open S3**

* **Go to AWS console → S3 service**

**🡪 Create bucket**

1. **Click Create bucket**
2. **Enter:**
   * **Bucket name: vpc-flowlog-bucket-shahbazz**
3. **Region: Same region as your VPC**
4. **Keep all default settings**
5. **Click Create bucket**

****

**03. Create VPC Flow Logs (CloudWatch Destination)**

**🡪 Go to VPC Dashboard**

* **AWS Console → VPC**

**🡪 Select Your VPC**

1. **From left menu click Your VPCs**
2. **Select the VPC you created (my-vpc-01)**

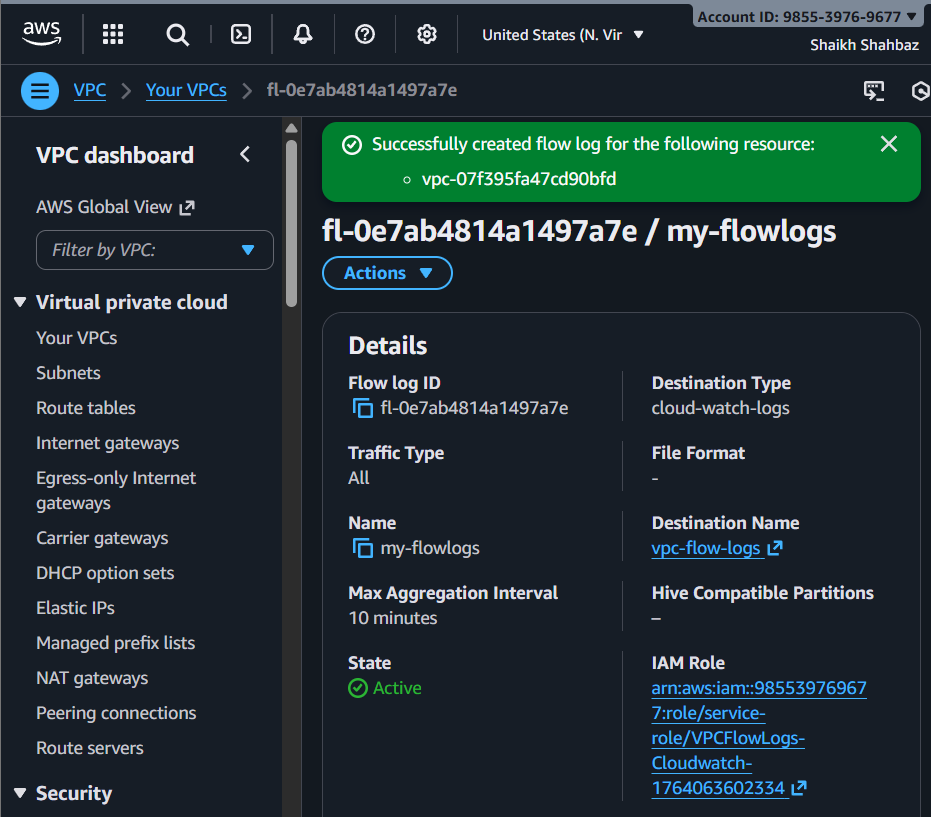
**🡪 Create Flow Log for CloudWatch**

1. **Select Create flow log**

**🡪 Configure Log Settings**

| **Field** | **Value** |
| --- | --- |
| **Filter** | **ALL (captures Accept + Reject + All traffic)** |
| **Max aggregation interval** | **10 minutes (default)** |
| **Destination** | **CloudWatch Logs** |
| **Log group** | **Choose vpc-flow-logs** |
| **IAM role** | **Click Create new role (AWS automatically creates one)** |

1. **Click Create flow log**

****

**04. Create VPC Flow Logs (S3 Destination)**

**You must repeat the steps to create a second flow log.**

**🡪 Select the Same VPC Again**

* **VPC Console → Your VPCs**
* **Select my-vpc-01**

**🡪 Actions → Create Flow Log**

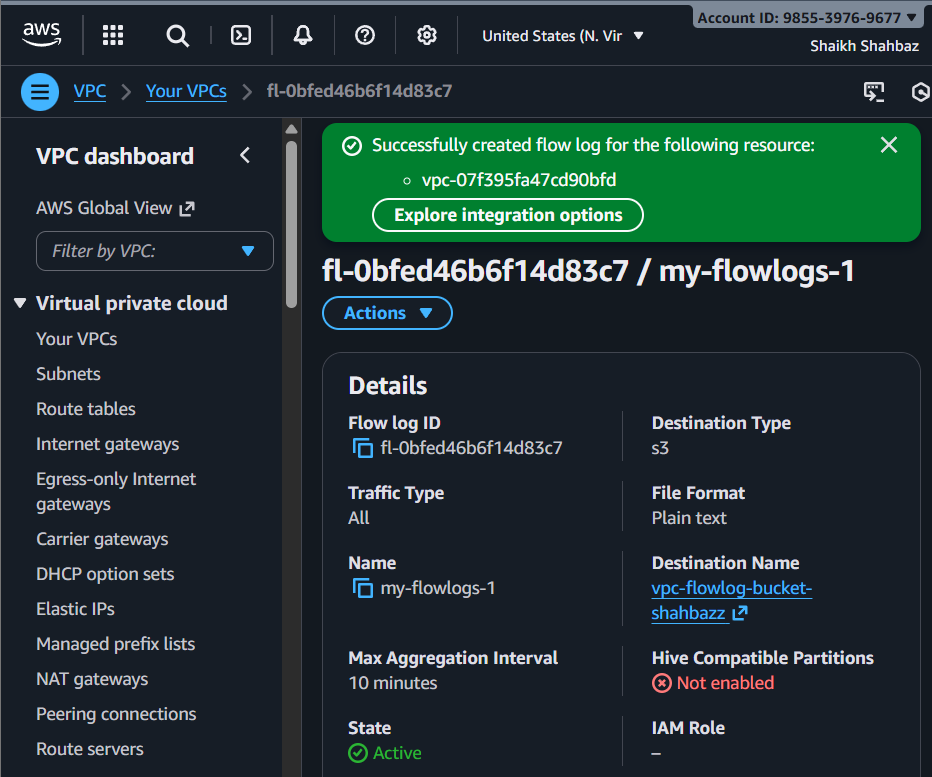
**Repeat same as before.**

**🡪 Configure for S3**

**This time change the Destination.**

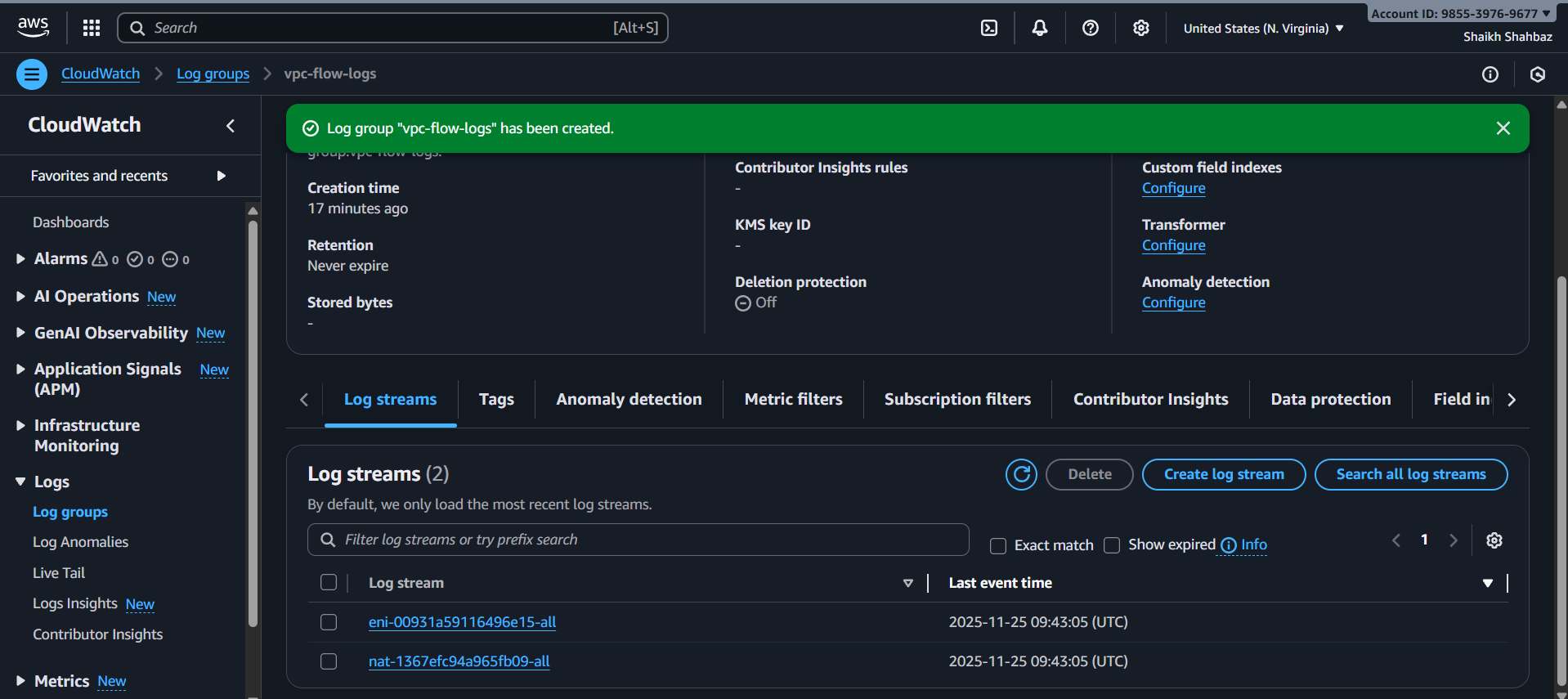
| **Field** | **Value** |
| --- | --- |
| **Filter** | **ALL** |
| **Max aggregation interval** | **10 minutes** |
| **Destination** | **S3** |
| **S3 bucket ARN** | **Choose the bucket created earlier: arn : aws : s3 : : : vpc-flowlogs-bucket-shahbazz** |

1. **Leave rest as default.**
2. **Click Create flow log**

****

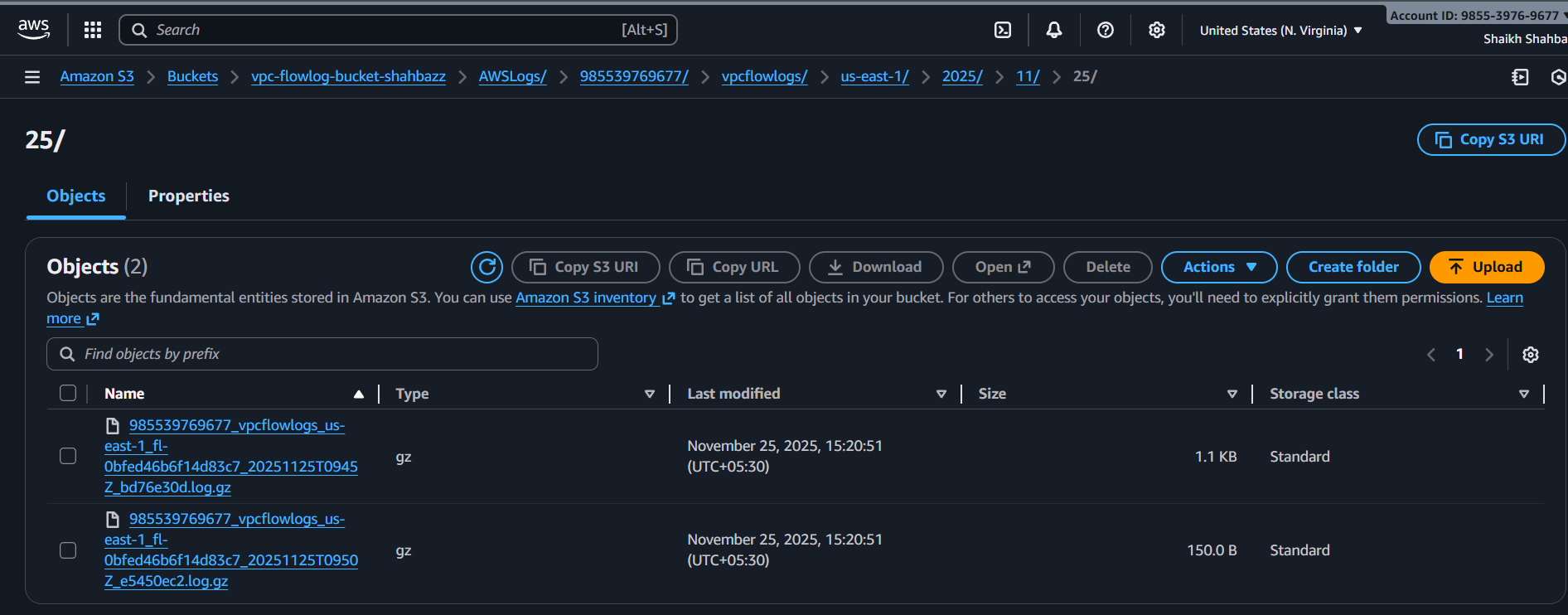
**🡪 Flow Log #1**

* **Destination: CloudWatch**
* **Log Group: vpc-flow-logs**

****

**🡪 Flow Log #2**

* **Destination: S3**
* **Bucket: vpc-flowlogs-bucket-shahbazz**

****