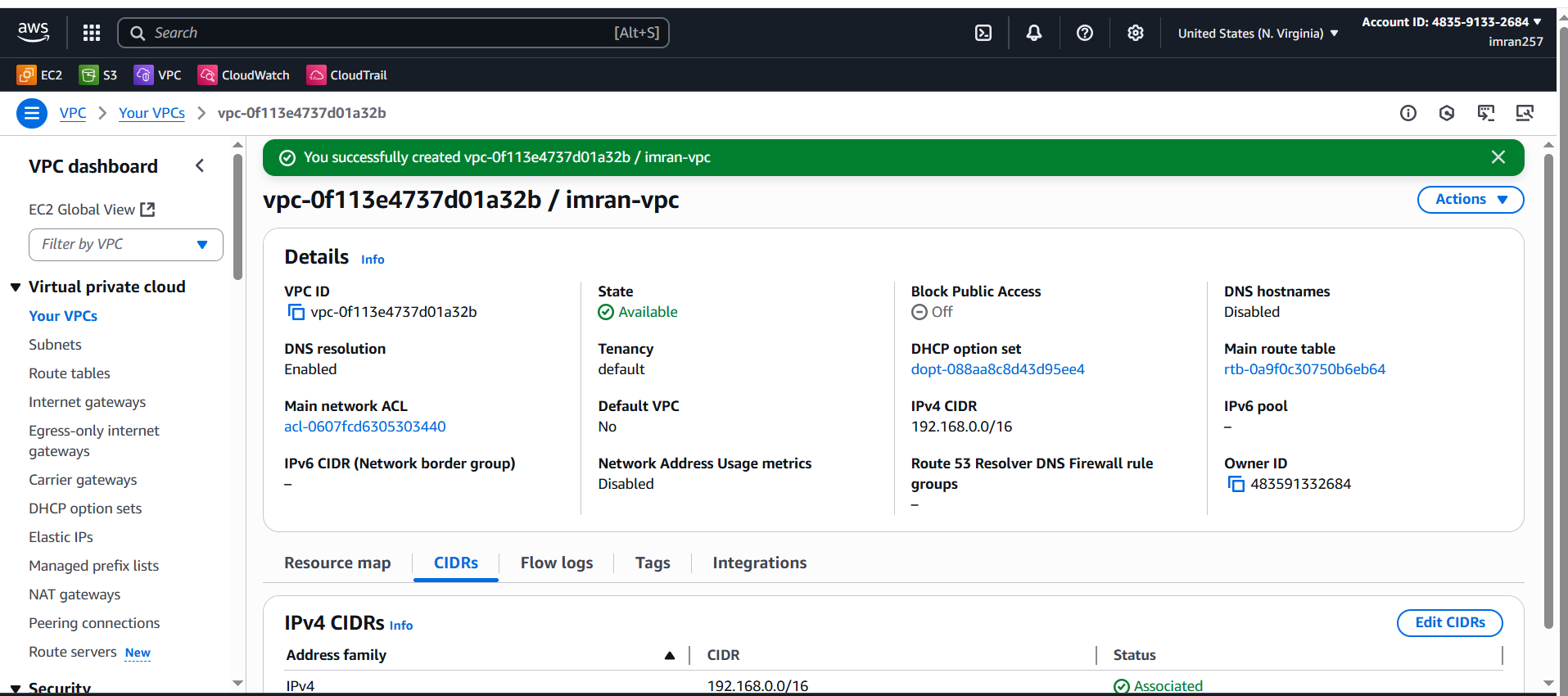
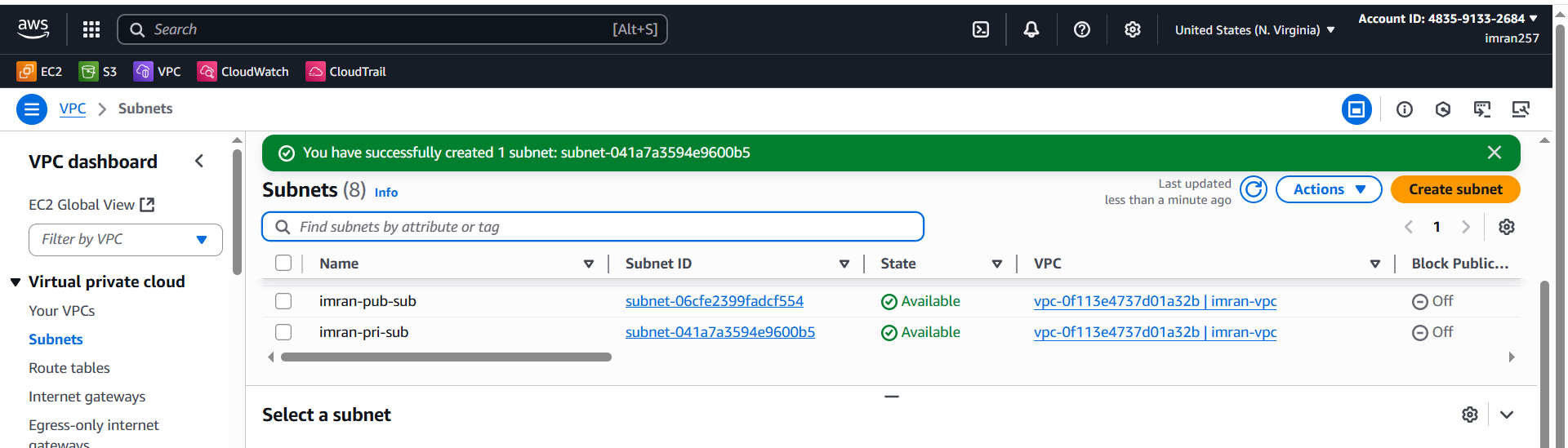
1. Create one VPC in N.virginia region.
2. Create two subnets. One Public subnet and one private subnet.
3. Provide the IGW to the vpc.
4. Create One public RT and one private RT.
5. Deploy NAT gateway on public subnet and attach the NAT gatewat to private subnet.
6. Create Two instances,one in public subnet and one in private subnet.
7. Deploy Apache server on both the ec2 instances with sample index.html file.
8. Create one application load balancer and attach the load balancer to both the ec2 instances.
9. Store Application load balancer logs to s3.
10. Store the VPC flow logs to CloudWatch group.
11. Create Monitoring Dashboards to monitor CPU utilization and to monitor Apache service.
12. CPU utilizations more than 70% then it should triggered Autoscaling and launch new instance.

—-----------------------------------------------------------------------------------------------------------------------

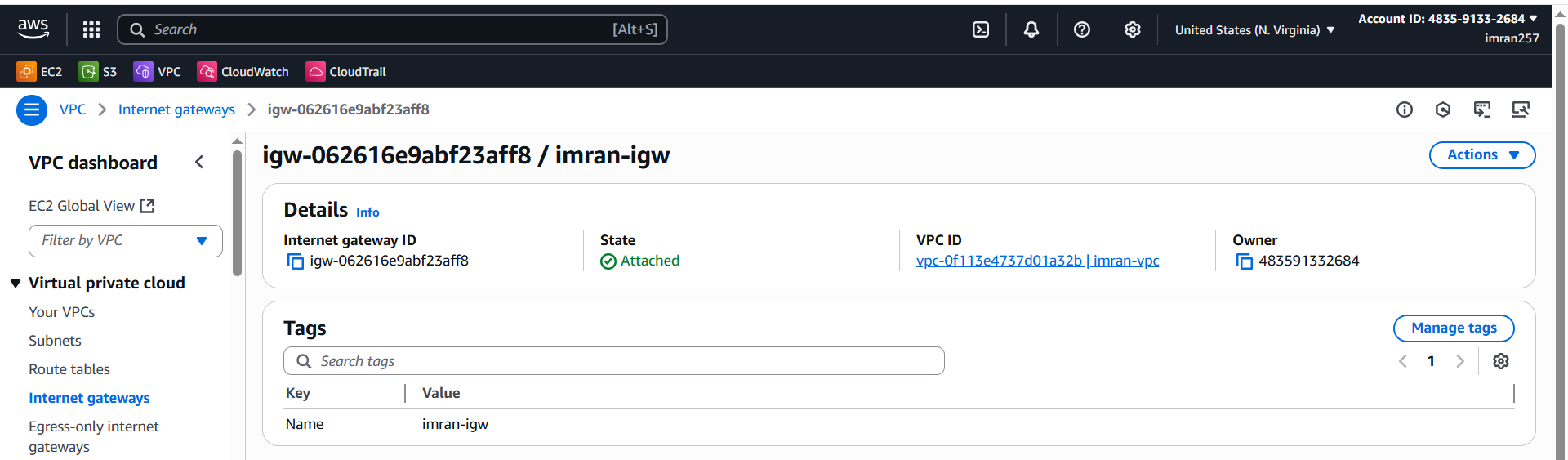
1. Create one VPC in N.virginia region.



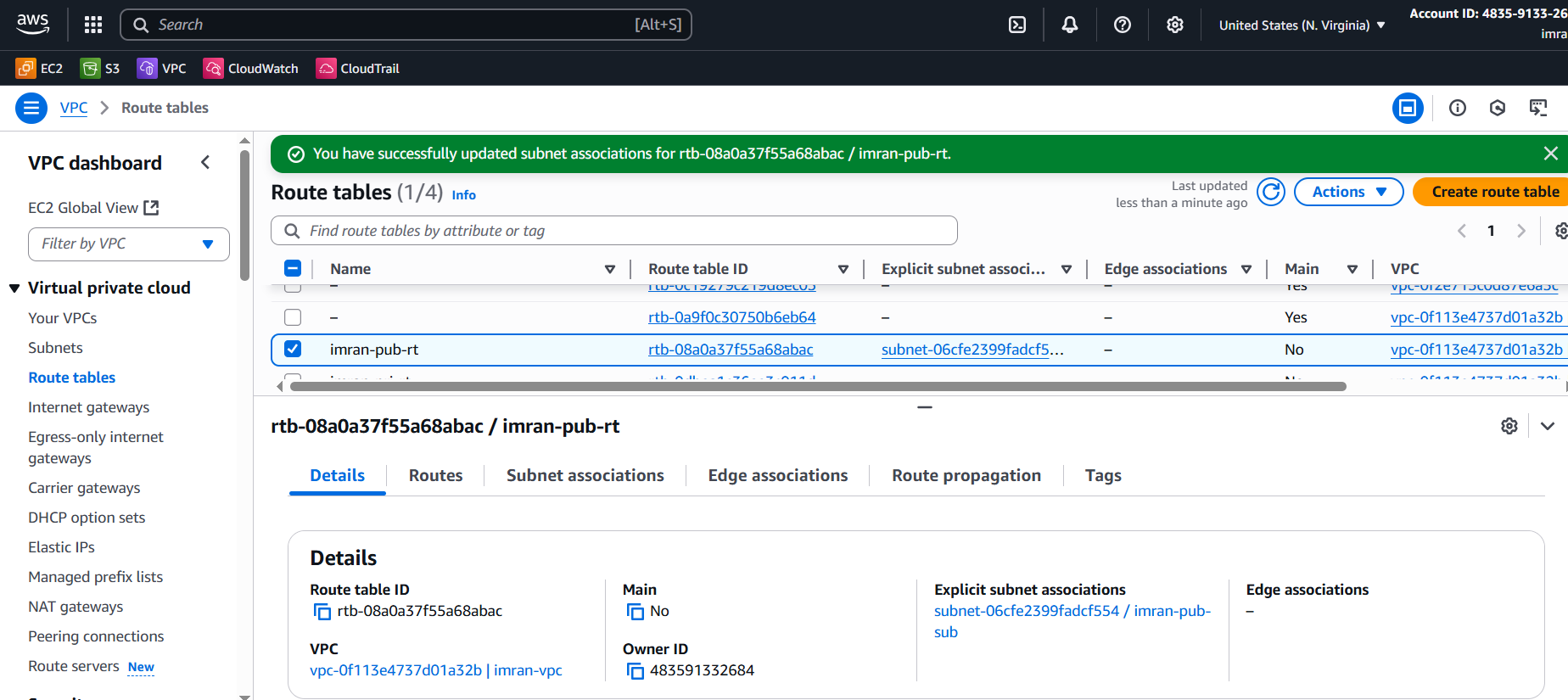
1. Create two subnets. One Public subnet and one private subnet.

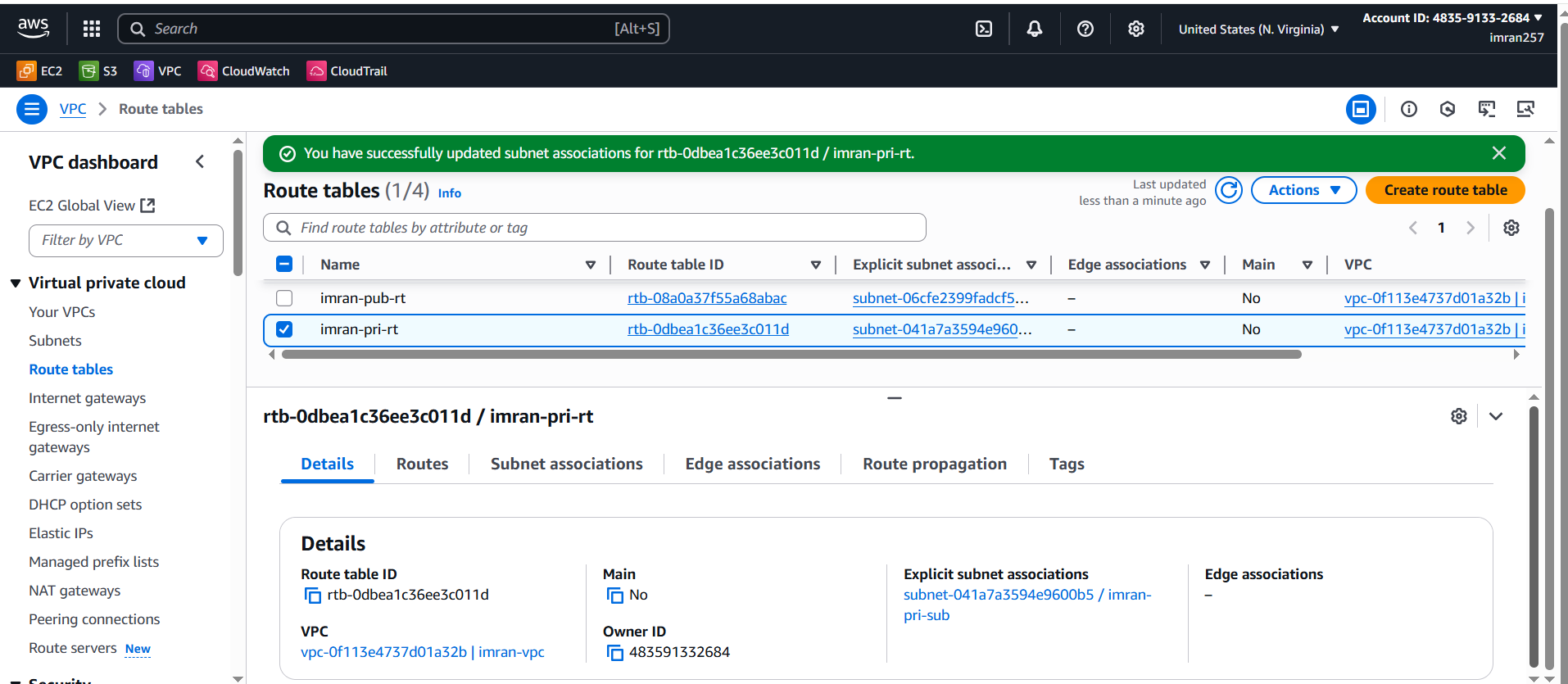


1. Provide the IGW to the vpc.

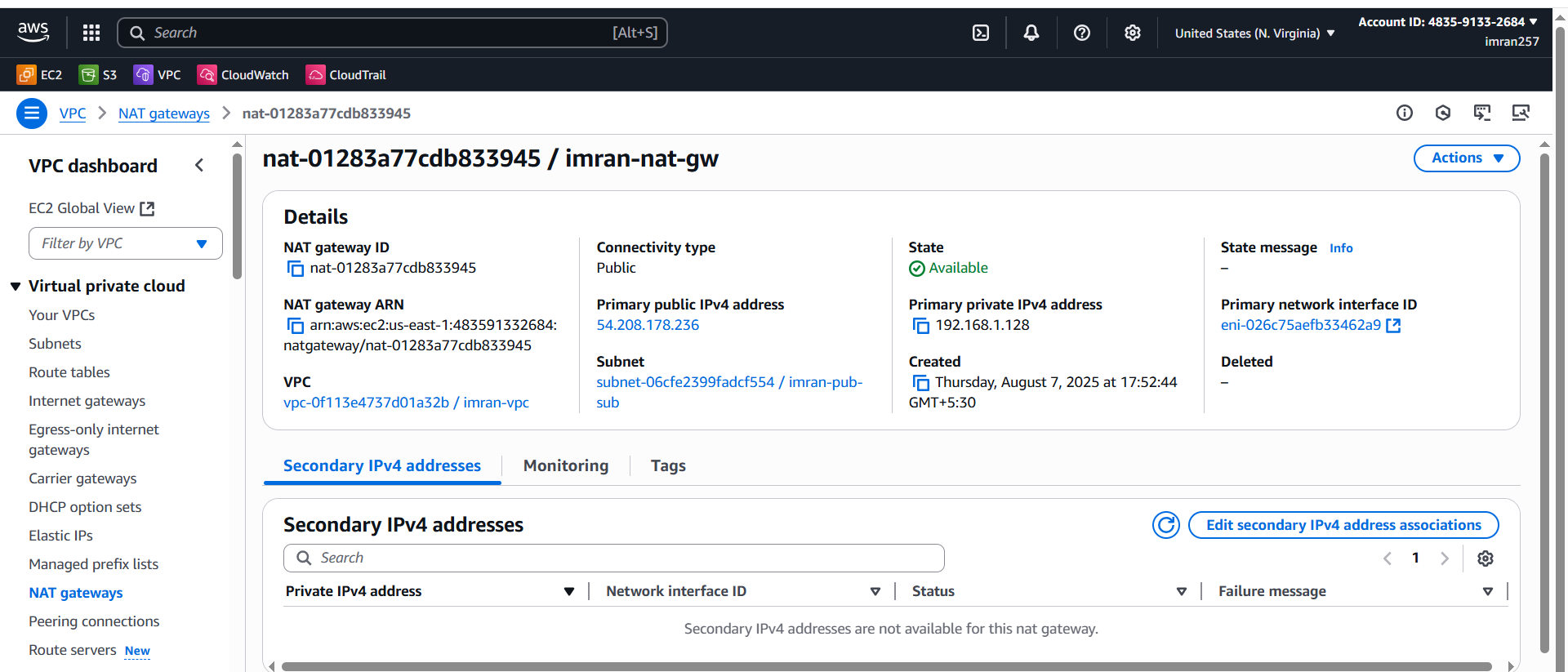


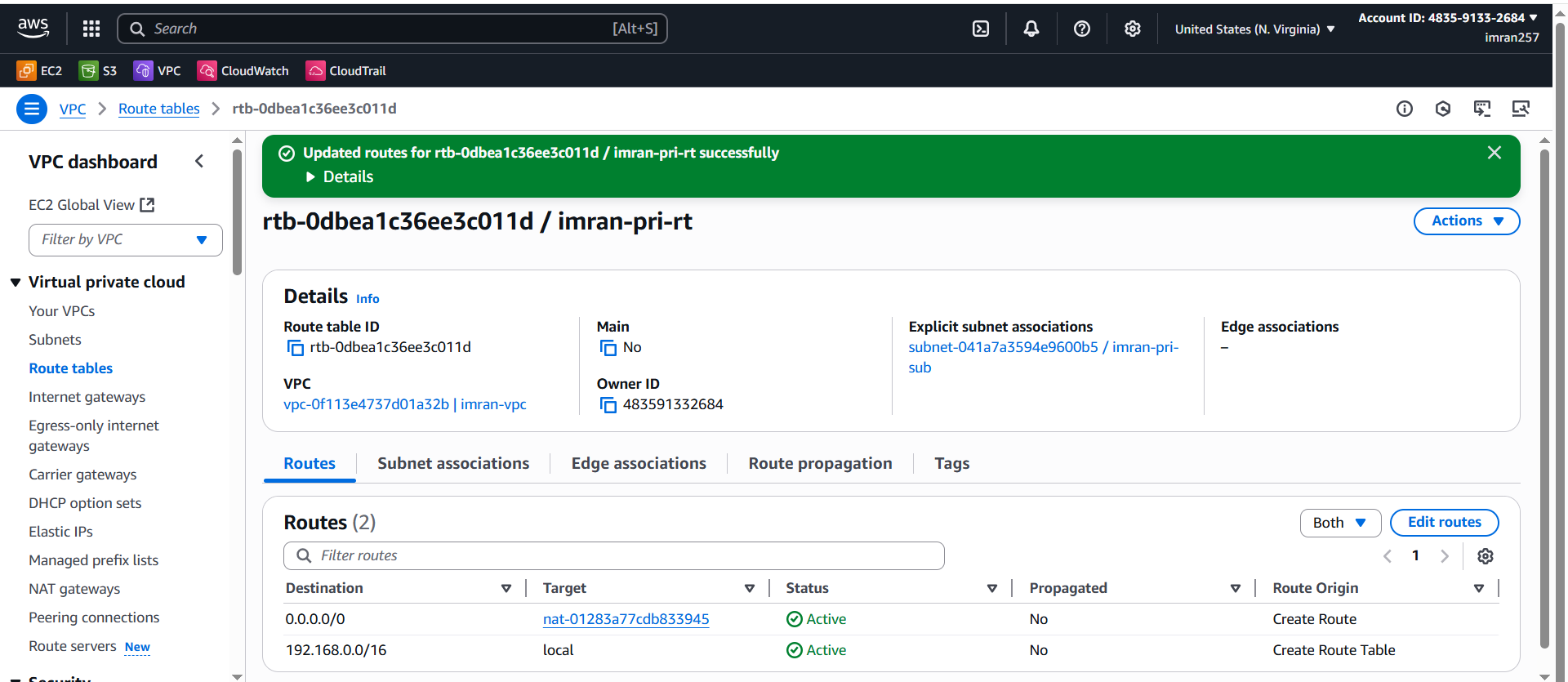
1. Create One public RT and one private RT.



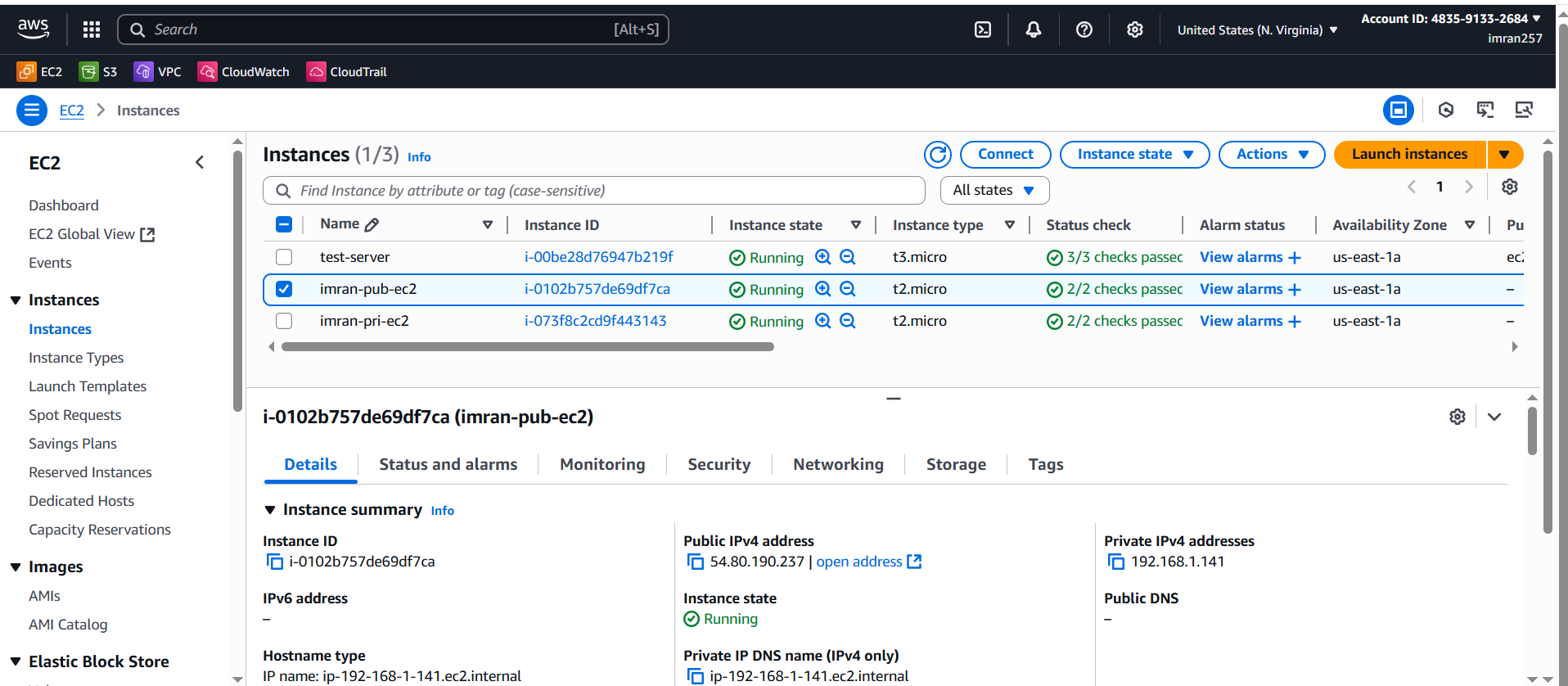


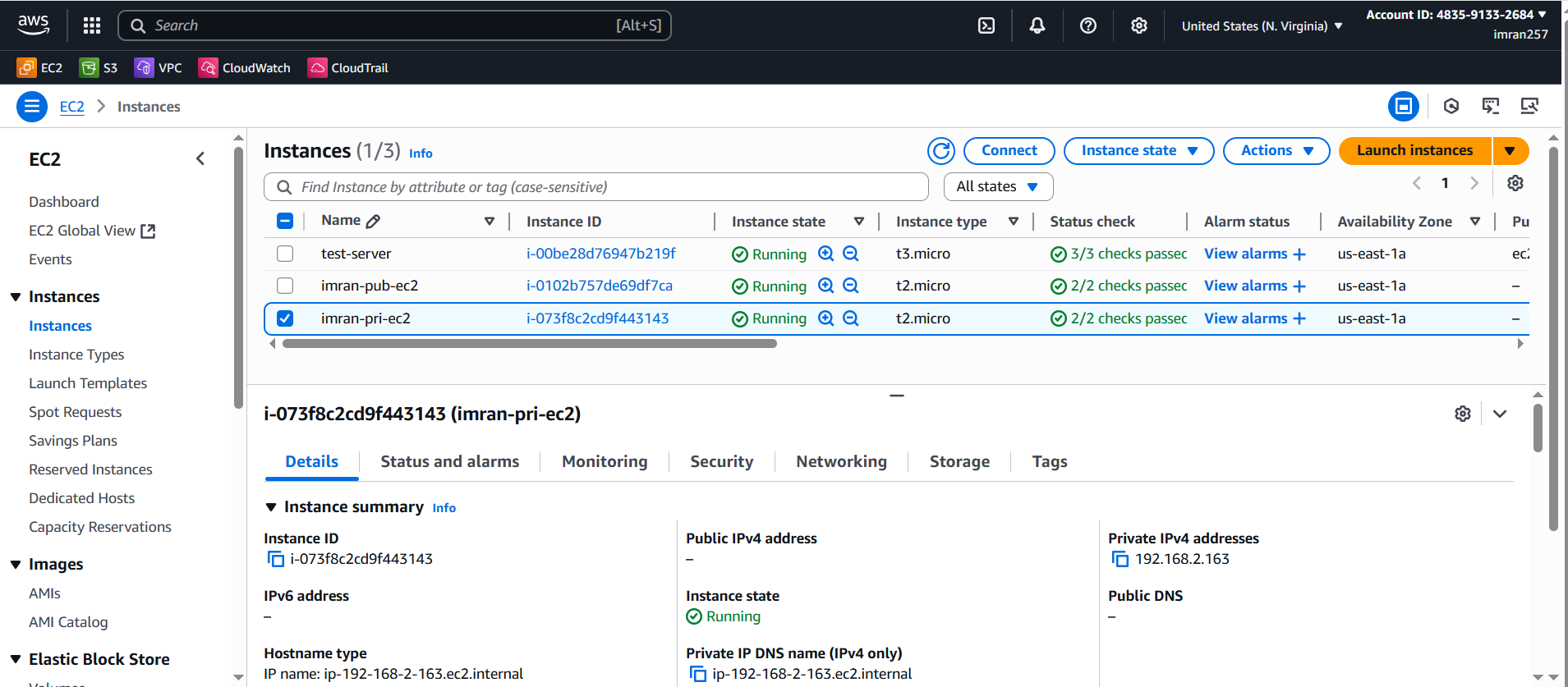
1. Deploy NAT gateway on public subnet and attach the NAT gateway to private subnet.





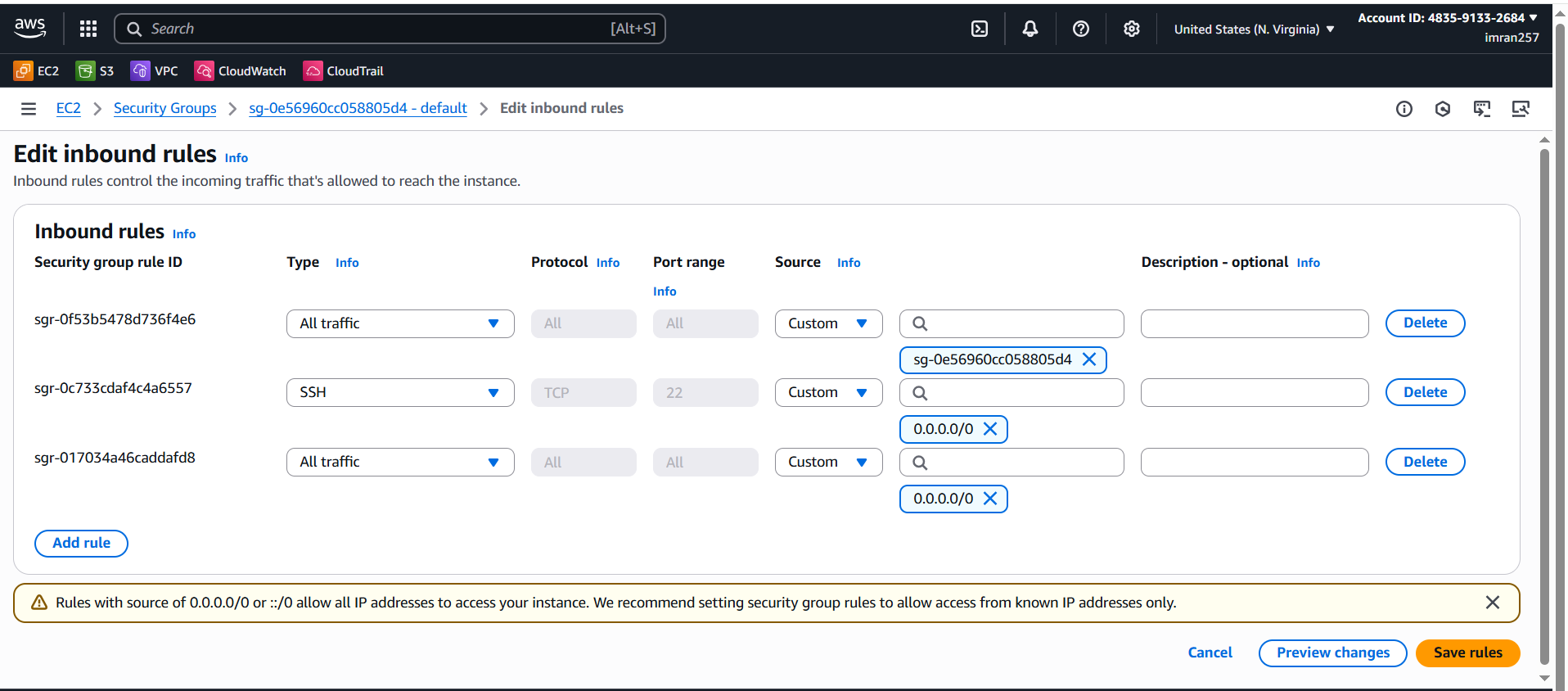
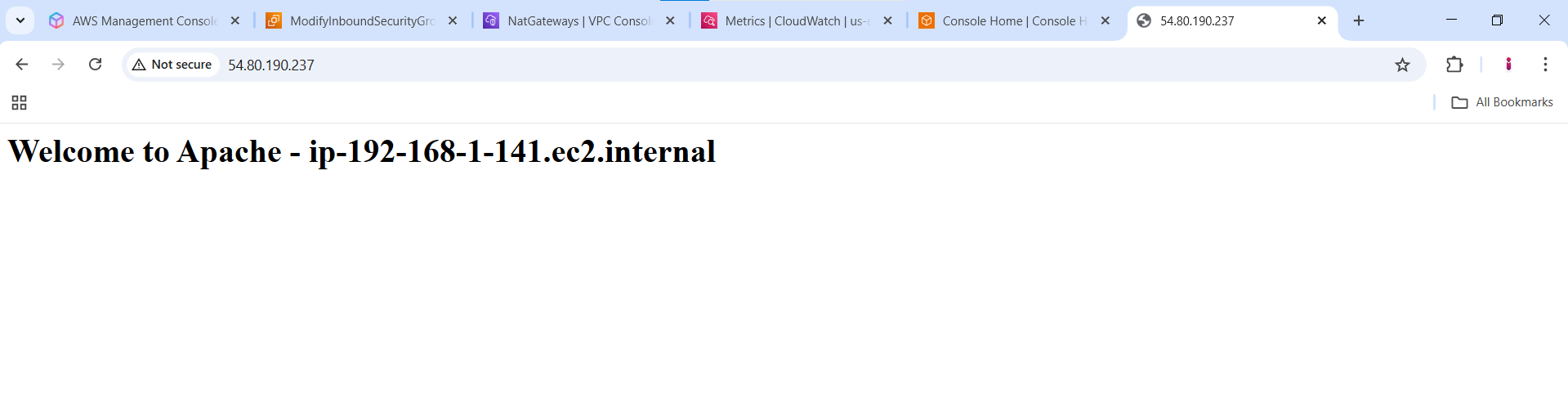
1. Create Two instances,one in public subnet and one in private subnet.



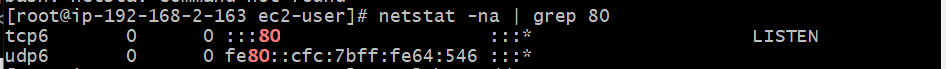
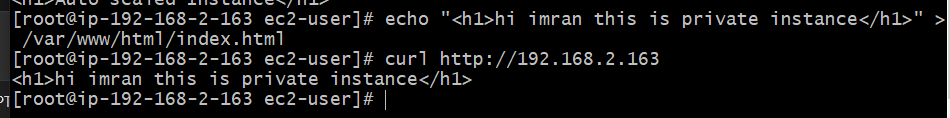


1. Deploy Apache server on both the ec2 instances with sample index.html file.

**APACHE on PUBLIC EC2**

1. Created pub ec2 and pri ec2(disabled public IP)
2. 
3. sudo netstat -tulnp | grep :80
4. sudo vi /etc/httpd/conf/httpd.conf
5. Listen 0.0.0.0:80 need to insert in above file
6. sudo systemctl restart httpd
7. sudo netstat -tulnp | grep :80
8. 
9. 

**APACHE on PRIVATE EC2**

1. Now copy the pem key which is used in private EC2 and create test.pem file in public ec2 and paste the key and save.
2. Now you can connect → ssh -i test.pem ec2-user@PrivateInstancePrivateIP.
3. Now install the httpd
4. 
5. 
6. Create one application load balancer and attach the load balancer to both the ec2 instances.

### **Steps Performed**

#### **1. Prerequisites**

Before creating the ALB, we ensured:

* A **VPC** was already created (imran-vpc).
* Two **EC2 instances** were launched:  
  + **Public EC2 Instance** in Public Subnet 1 (192.168.1.0/24)
  + **Private EC2 Instance** in Public Subnet 2 (192.168.2.0/24)  
     (in one EC2 Apache HTTP Server installed and running.)
* Security Group allowed **HTTP (port 80)** inbound from the internet.

#### **2. Creating the ALB**

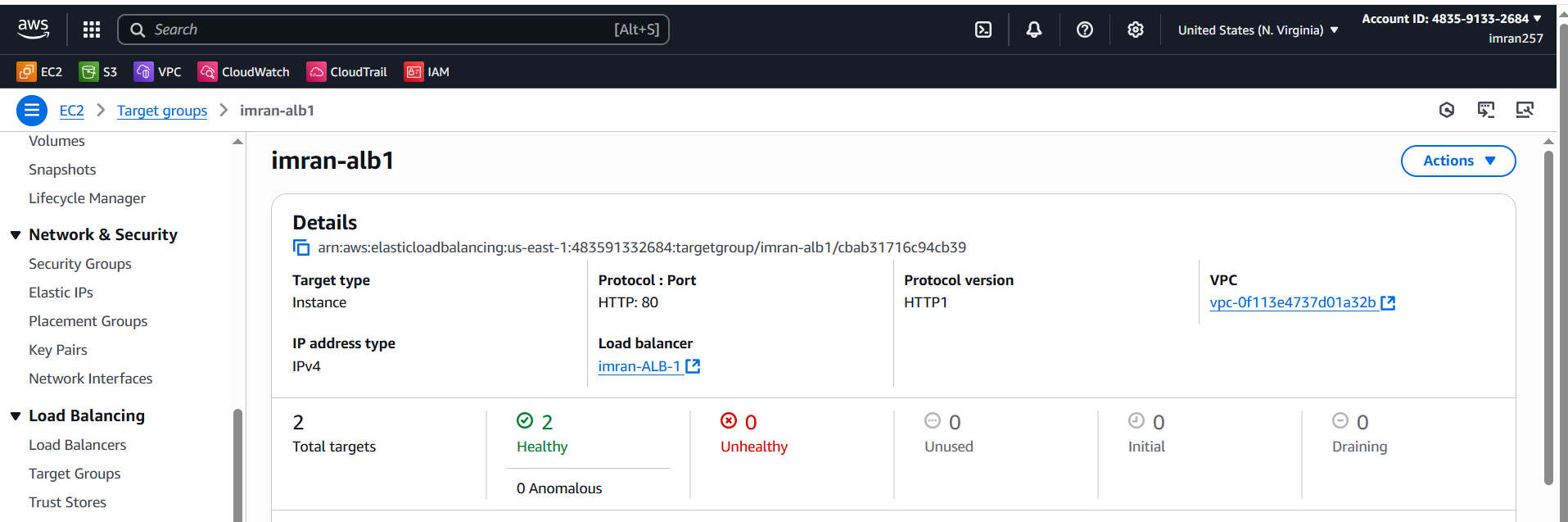
1. **Go to** AWS Management Console → EC2 → **Load Balancers** → *Create Load Balancer*.
2. **Choose** Application Load Balancer.
3. **Name:** imran-alb-1.
4. **Scheme:** internet-facing (so it can receive public traffic).
5. **IP address type:** IPv4.
6. **Network Mapping:**
   * **VPC:** imran-vpc.
   * **Subnets:** Selected **two public subnets** in different Availability Zones (required for ALB).
7. **Security Group:** Selected/created one that allows **HTTP (port 80)** inbound from 0.0.0.0/0.

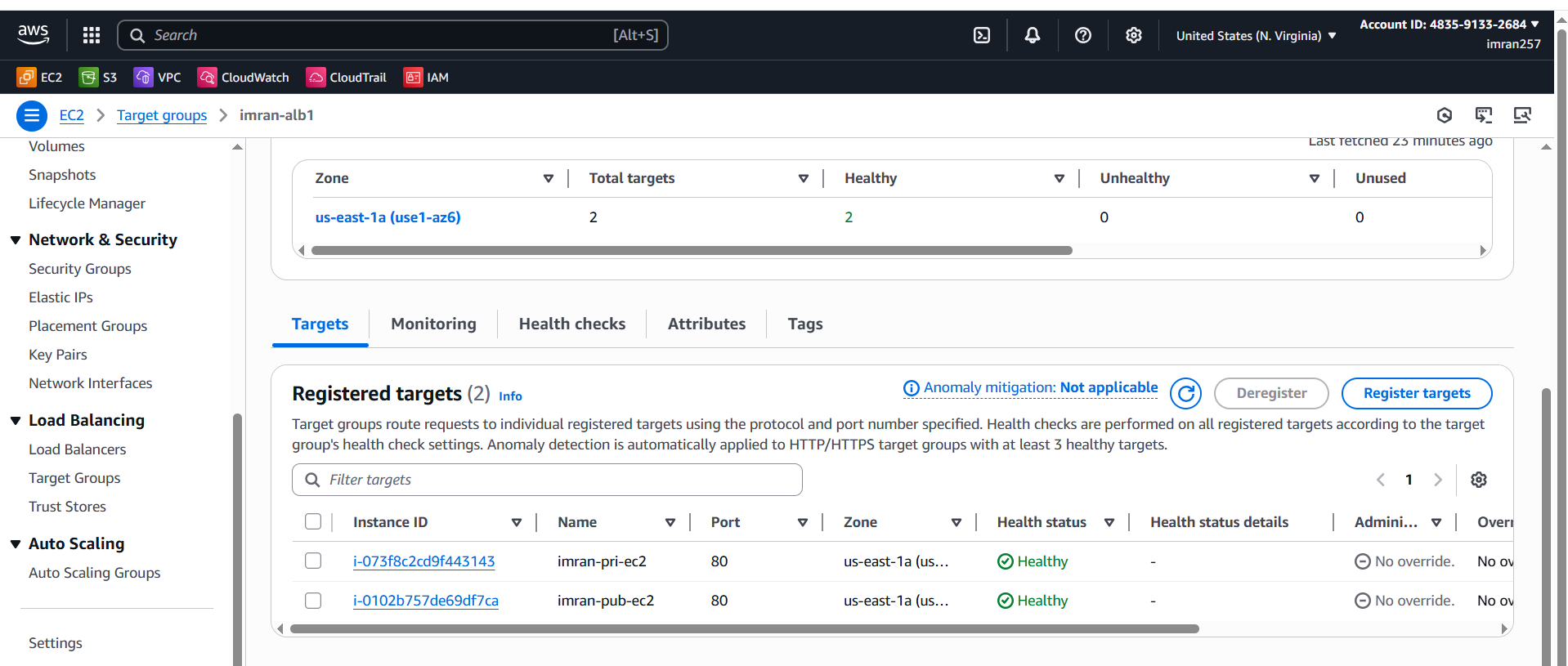
#### **3. Target Group Creation**

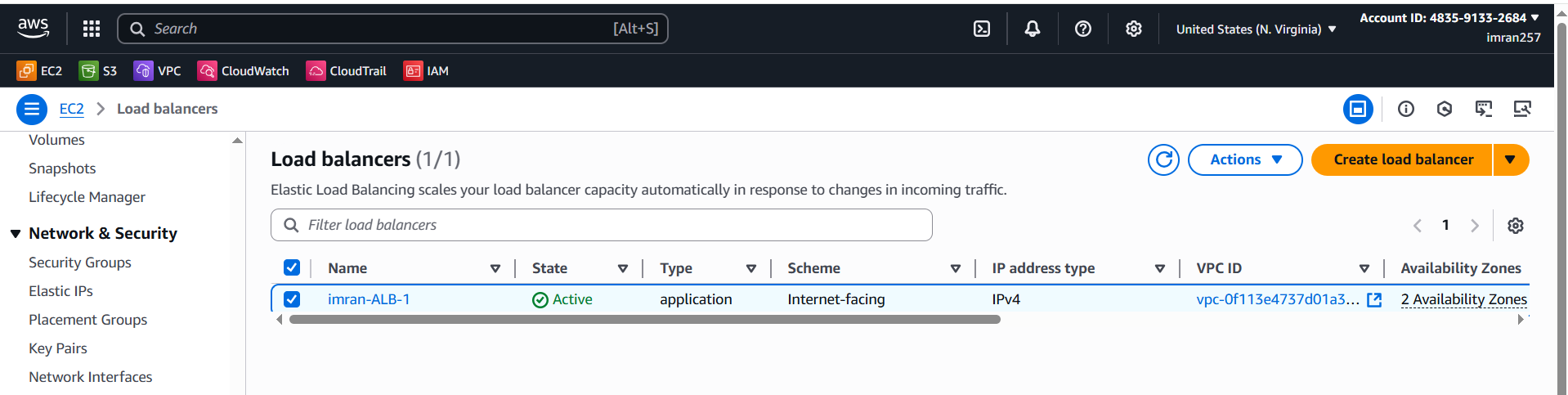
1. While creating the ALB, clicked **Create a target group**.
2. **Target type:** Instance.
3. **Protocol:** HTTP.
4. **Port:** 80.
5. **Target Group Name:** alb-targets.
6. Registered both EC2 instances:  
   * **Public EC2** (Apache running).
   * **Private EC2**
7. Saved target group.

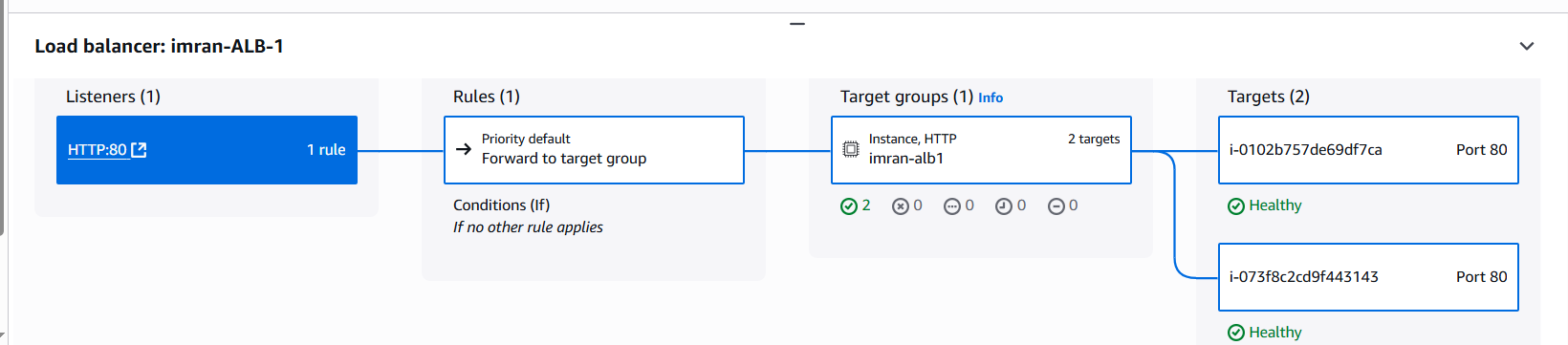
#### **4. Finalizing the ALB**

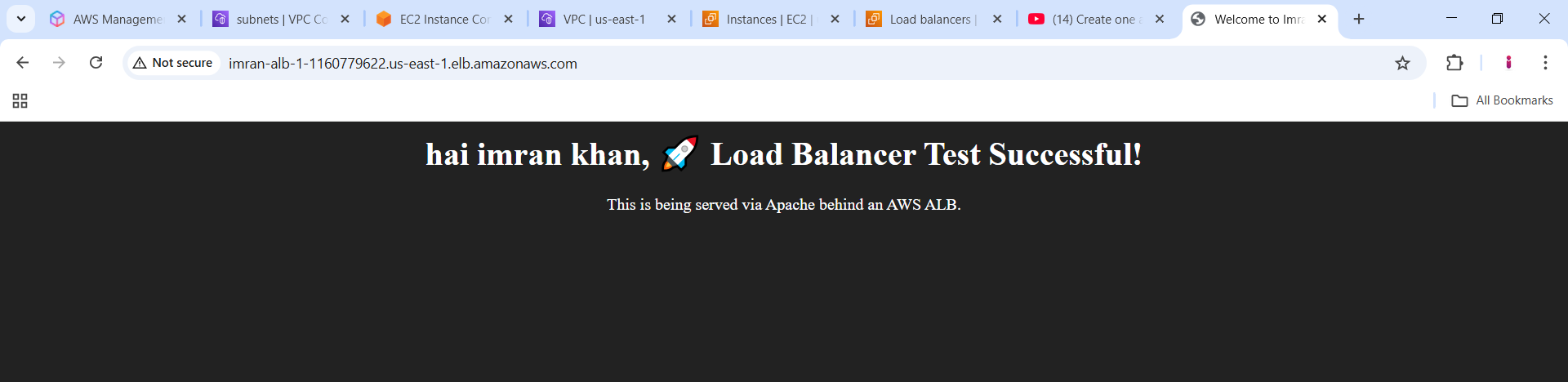
1. Linked the **target group** (alb-targets) to the ALB listener on port 80.
2. Reviewed and created the ALB.
3. Waited for the **ALB status** to become **Active**.

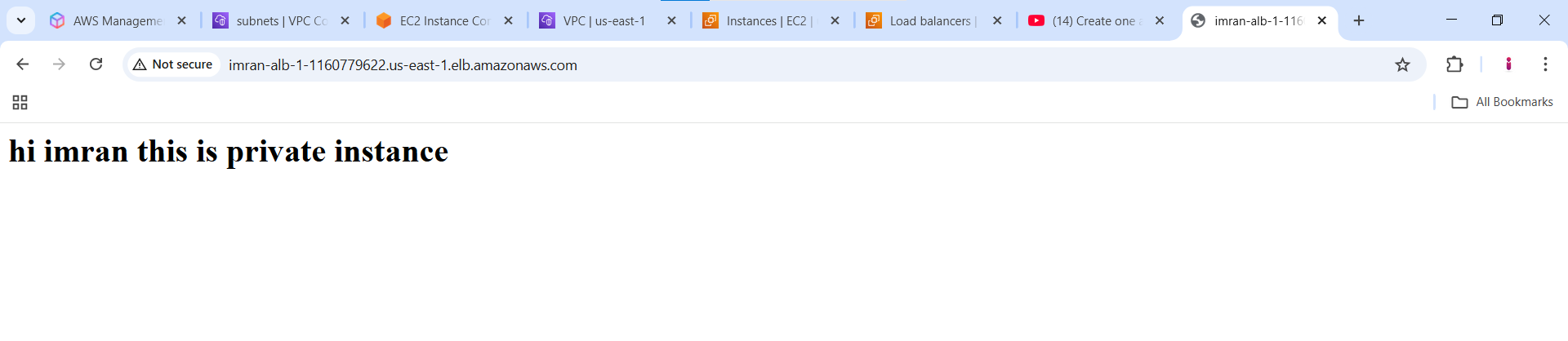




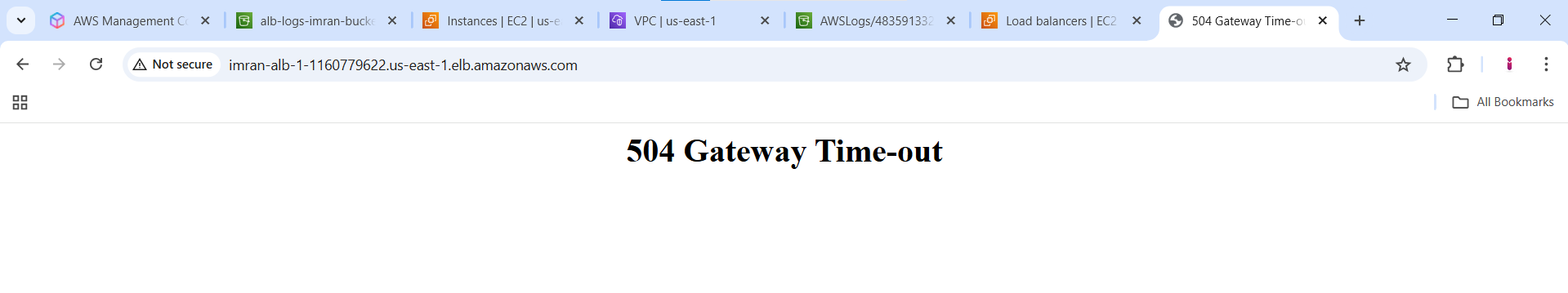








I stopped the public instance then i got as ,



1. **Store Application load balancer logs to s3.**

### **Steps**

#### **1. Create an S3 bucket for logs**

1. Go to the **Amazon S3 Console** → Click **Create bucket**.
2. **Bucket name:** Must be unique globally (example: alb-logs-imran-bucket).
3. **Region:** Select the **same region** as your ALB.
4. **Block Public Access:** Keep **enabled** for security.
5. Click **Create bucket**.

#### **2. Configure bucket policy to allow ALB to write logs**

1. Go to the S3 bucket → **Permissions** tab → **Bucket Policy** → Click **Edit**.
2. Paste the following policy (replace with your **bucket name** and **AWS region**):

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "AWSALBLogsPolicy",

"Effect": "Allow",

"Principal": {

"Service": "logdelivery.elasticloadbalancing.amazonaws.com"

},

"Action": "s3:PutObject",

"Resource": "arn:aws:s3:::alb-logs-imran-bucket/AWSLogs/\*"

}

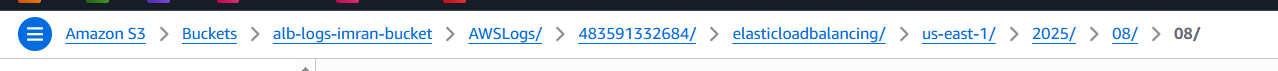
]

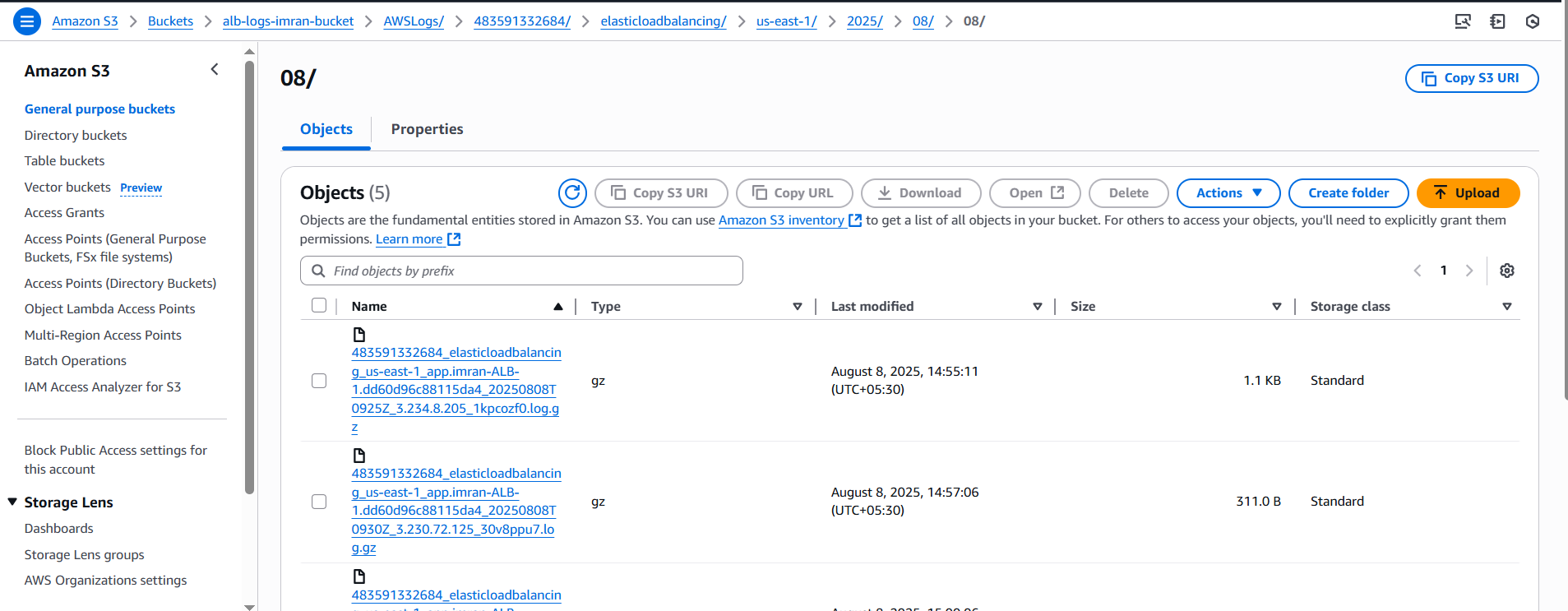
}

#### **3. Enable Access Logs in the ALB**

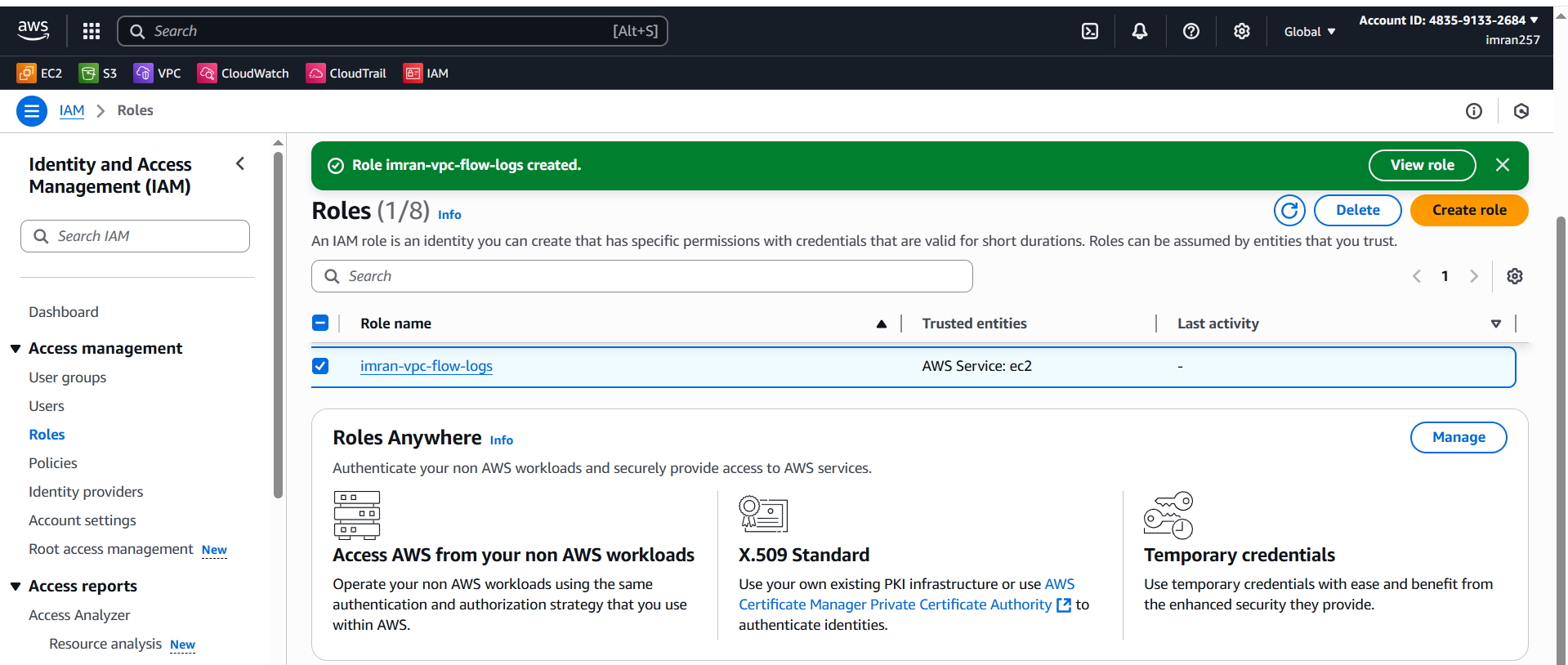
1. Go to **EC2 Console** → **Load Balancers**.
2. Select your **Application Load Balancer**.
3. Go to the **Attributes** tab.
4. Under **Access logs**, click **Edit**.
5. **Enable Access Logs**.
6. Select the **S3 bucket** you created.
7. (Optional) Specify a **prefix** for organizing logs (e.g., alb-logs/).
8. Click **Save changes**.

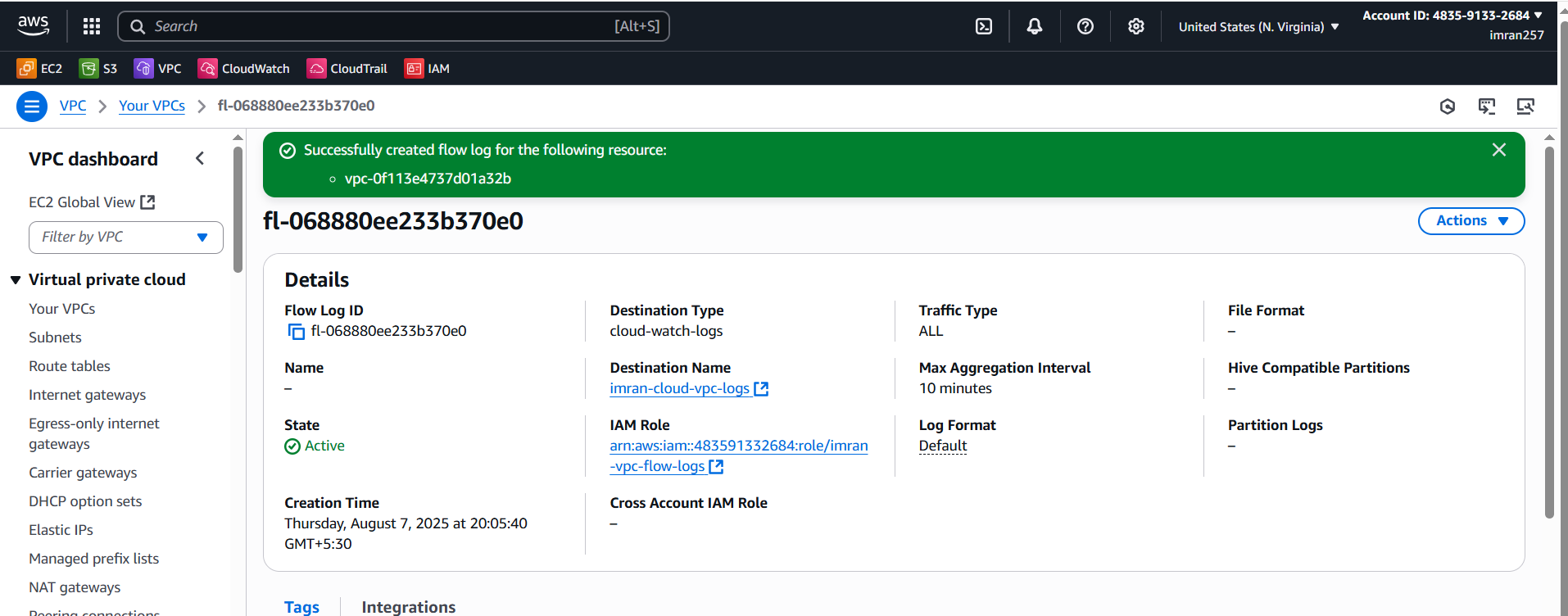
#### **4. Verify Logs**

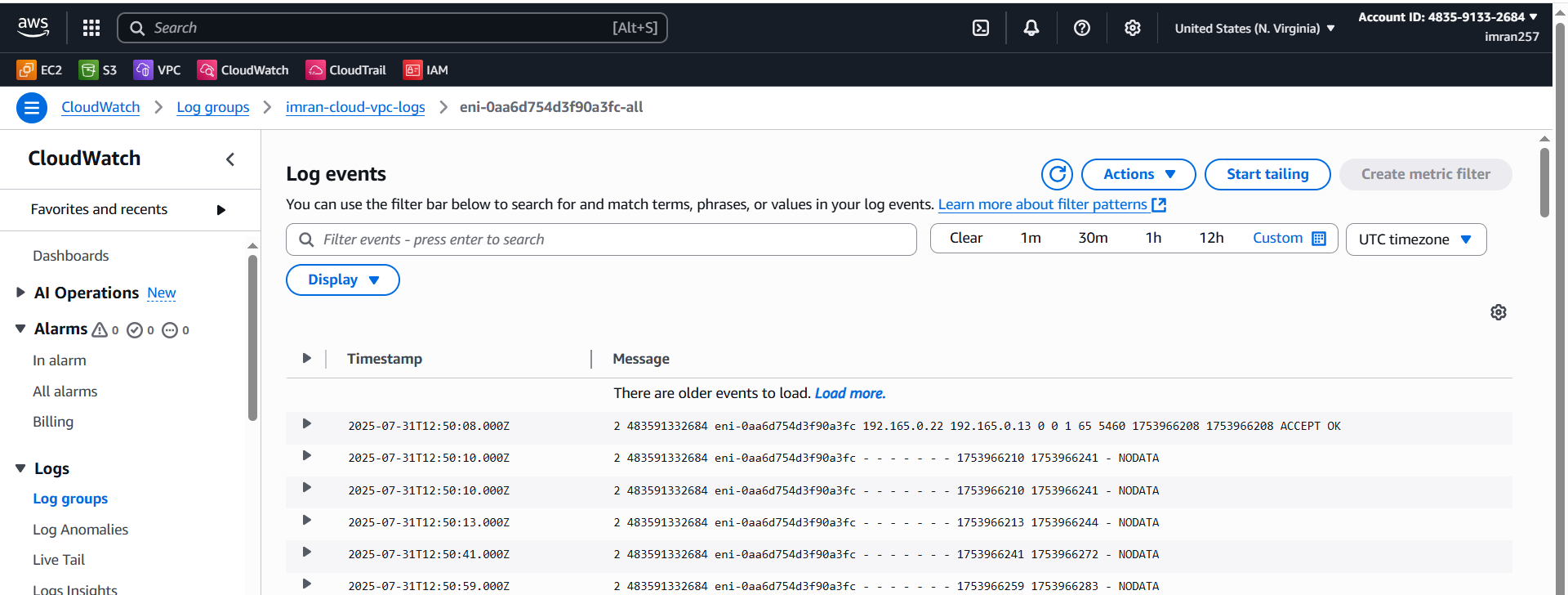
* ALB logs are delivered **every 5 minutes** to the S3 bucket.
* 



1. Store the VPC flow logs to CloudWatch group.







1. **Create Monitoring Dashboards to monitor CPU utilization and to monitor Apache service.**

#### **Step 1: Go to CloudWatch → Dashboards → Create dashboard**

* Name: imran-dashboard

#### **Step 2: Add Widget for CPU**

* Click **Add widget → Number**
* Select **EC2** → **Per-Instance Metrics**
* EC2 instance
* Metric: CPUUtilization
* Click **Create Widget**

Now for apache write a script in EC2. vi apache-monitor.bash

#!/bin/bash

status=$(systemctl is-active httpd)

if [ "$status" == "active" ]; then

value=1

else

value=0

fi

aws cloudwatch put-metric-data \

--metric-name ApacheStatus \

--namespace Apache \

--value $value \

--region us-east-1

**Give permissions:**

chmod +x [apache-monitor.sh](http://apache-monitor.sh)

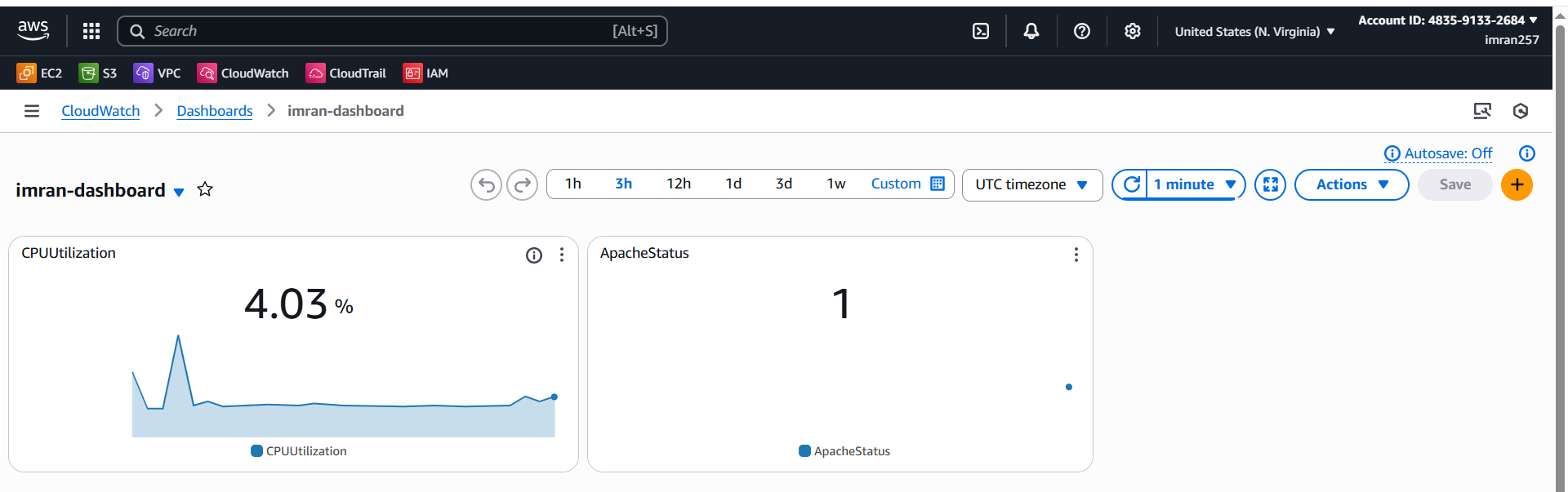
Install crontab then

Crontab -e

\*/1 \* \* \* \* /root/[apache-monitor.sh](http://apache-monitor.sh)

./apache-monitor.bash

Now go to imran-dashboard in cloudwatch and you can see apache and select widget with number and save.



1. CPU utilizations more than 70% then it should triggered Autoscaling and launch new instance.
2. Create a template with EC2 configuration and write script to install nginx and create 1 index.html page.

#!/bin/bash

Yum -y install httpd

Touch /var/www/html/index.html

Echo “welcome to new template from ASG” >> /var/www/html/index.html

1. Create autoscaling group with take or attach the template and in automatic scaling select Target Tracking Policy → metric type avg CPU utilization → if Target value 70 → 100 seconds and then create.
2. Once your EC2 reached CPU load above 70 automatically one EC2 will be launched you can check EC2 Instances.

