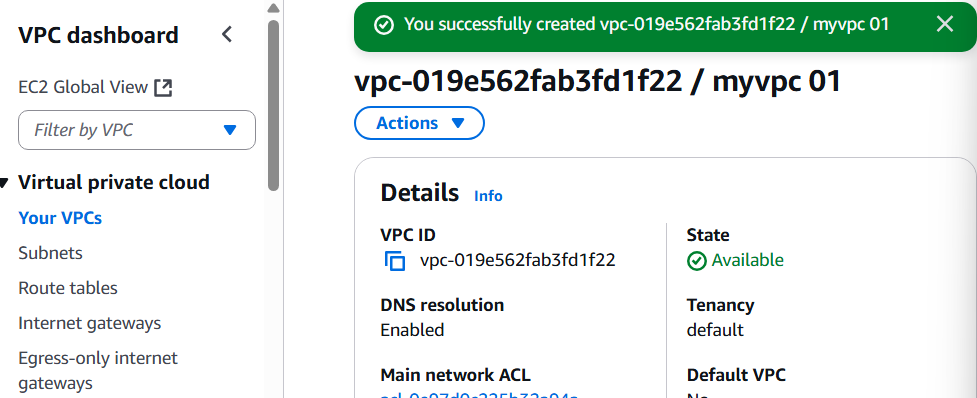
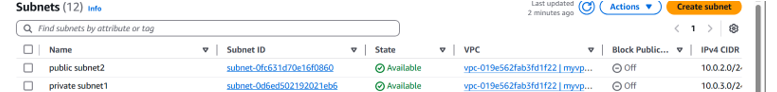
1)Create one vpc in N.virginia region.

* Go to the VPC Dashboard
* In the search bar, type "VPC" and go to the VPC Dashboard.
* Click “Create VPC”
* Name tag: MyVPC01
* IPv4 CIDR block: 10.0.0.0/16



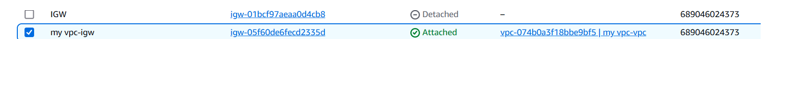
2) Create two subnets. One Public subnet and one private subnet. - Create vpc

* Create Subnets
* Public Subnet 1
* Go to Subnets > Create subnet
* VPC ID: Choose MyCustomVPC
* Subnet name: Public-Subnet-1
* Availability Zone: Pick one (e.g., us-east-1a)
* IPv4 CIDR block: 10.0.1.0/24
* Private Subnet 1
* Name: Private-Subnet-1
* AZ: us-east-1a
* CIDR: 10.0.3.0/24



3) Provide the IGW to the vpc.

* In the VPC dashboard, select **Internet Gateways**, then click **Create internet gateway**.
* Enter a name (e.g., MyInternetGateway) and create the gateway.
* Select the newly created internet gateway, click **Actions**, then **Attach to VPC**.
* Choose your VPC and attach the gateway.



4) Create One public RT and one private RT.

**1. Create a VPC**

1. **Navigate to the VPC Dashboard**:
   * Open the [AWS Management Console](https://console.aws.amazon.com/).
   * Go to **Services** > **VPC**.
2. **Create a New VPC**:
   * Click on **"Your VPCs"** in the left navigation pane.
   * Click **"Create VPC"**.
   * **Name tag**: Enter a name for your VPC (e.g., MyVPC).
   * **IPv4 CIDR block**: Enter a CIDR block (e.g., 10.0.0.0/16).
   * Leave other settings as default or configure as needed.
   * Click **"Create VPC"**.

**2. Create Subnets**

1. **Public Subnet**:

* Click on **"Subnets"** in the left navigation pane.
* Click **"Create subnet"**.
* **Name tag**: Enter a name (e.g., PublicSubnet).
* **VPC**: Select the VPC you created (MyVPC).
* **Availability Zone**: Choose an AZ (e.g., us-east-1a).
* **IPv4 CIDR block**: Enter a subnet CIDR (e.g., 10.0.1.0/24).
* Click **"Create subnet"**.

1. **Private Subnet**:

* Repeat the above steps with:
* **Name tag**: PrivateSubnet.
* **IPv4 CIDR block**: 10.0.2.0/24.

**3. Create and Attach an Internet Gateway**

1. **Create Internet Gateway (IGW)**:

* Click on **"Internet Gateways"** in the left navigation pane.
* Click **"Create internet gateway"**.
* **Name tag**: Enter a name (e.g., MyIGW).
* Click **"Create internet gateway"**.

1. **Attach IGW to VPC**:

* Select the newly created IGW.
* Click **"Actions"** > **"Attach to VPC"**.
* Select your VPC (MyVPC) and click **"Attach internet gateway"**.

**4. Create Route Tables**

1. **Public Route Table**:

* Click on **"Route Tables"** in the left navigation pane.
* Click **"Create route table"**.
* **Name tag**: Enter a name (e.g., PublicRT).
* **VPC**: Select your VPC (MyVPC).
* Click **"Create route table"**.

1. **Private Route Table**:

* Repeat the above steps with:
* **Name tag**: PrivateRT.

**5. Configure Routes**

1. **Public Route Table**:

* Select PublicRT.
* Click on the **"Routes"** tab.
* Click **"Edit routes"**.
* Click **"Add route"**.
* **Destination**: 0.0.0.0/0.
* **Target**: Select **Internet Gateway** and choose MyIGW.
* Click **"Save routes"**.

1. **Private Route Table**:

* For now, the private route table will have only the local route (10.0.0.0/16 local).
* If you plan to allow instances in the private subnet to access the internet, you'll need to set up a NAT Gateway and update the route accordingly.

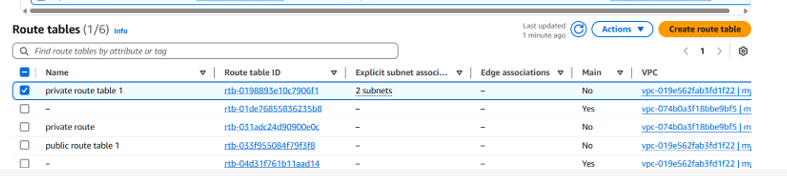
**6. Associate Subnets with Route Tables**

1. **Public Subnet**:

* Select PublicRT.
* Click on the **"Subnet associations"** tab.
* Click **"Edit subnet associations"**.
* Select PublicSubnet.
* Click **"Save associations"**.

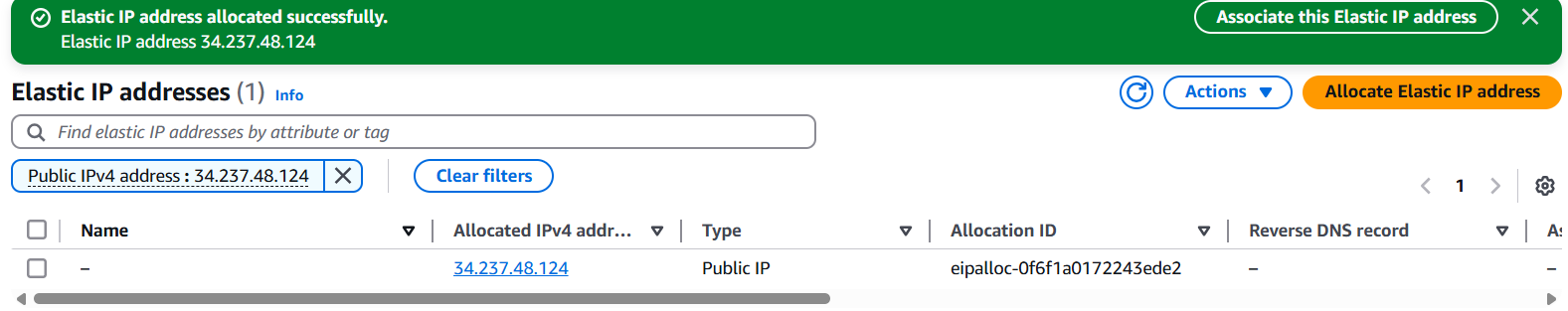
1. **Private Subnet**:

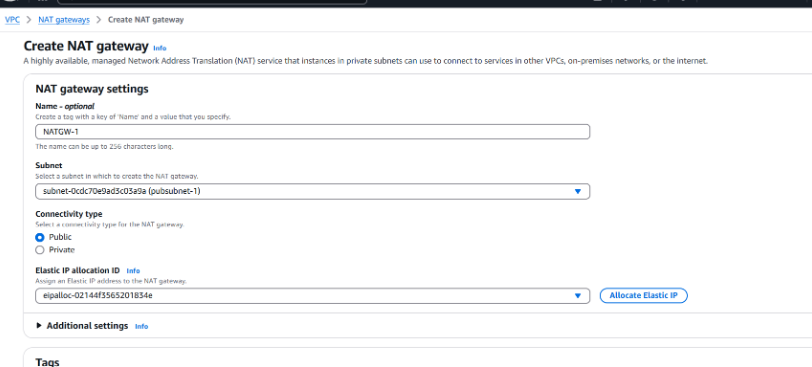
* Select PrivateRT.
* Click on the **"Subnet associations"** tab.
* Click **"Edit subnet associations"**.
* Select PrivateSubnet.
* Click **"Save associations"**.

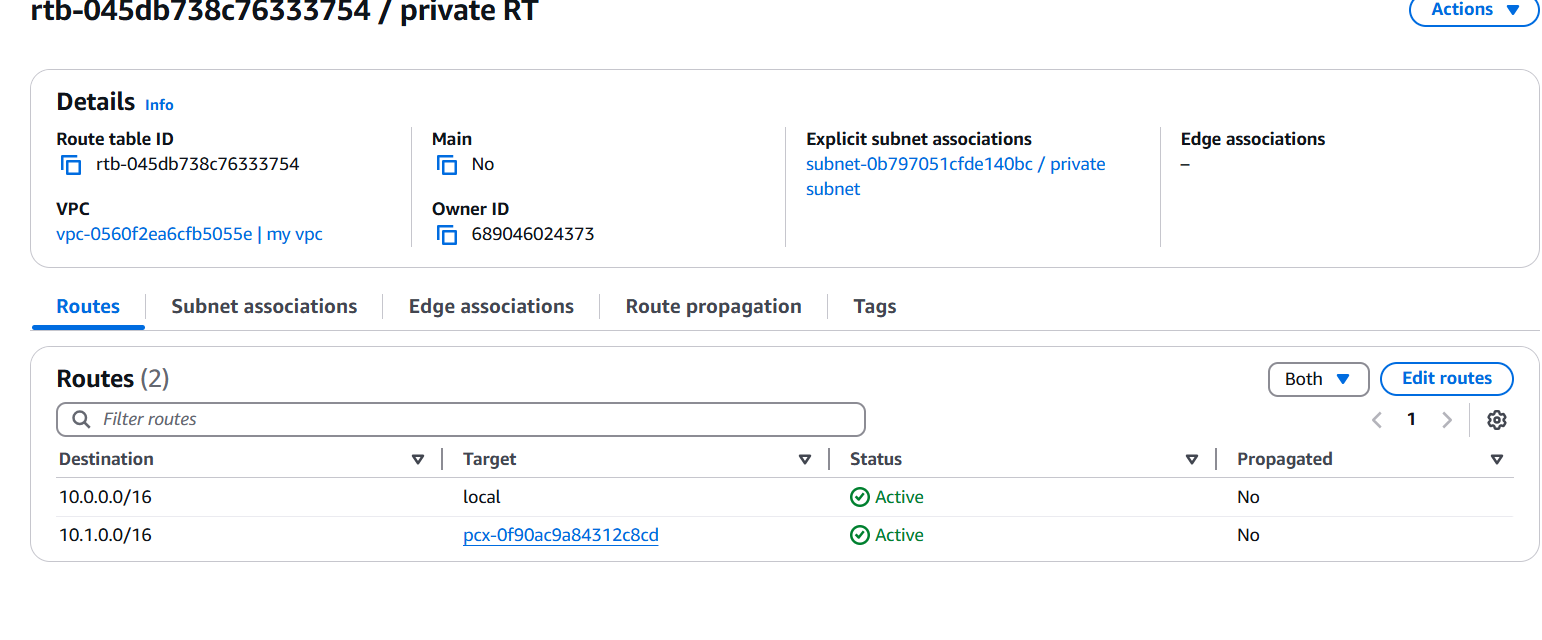


5) Deploy NAT gateway on public subnet and attach the NAT gateway to private subnet.

* Create a NAT Gate way Go to the AWS Console → Navigate to VPC.
* Create an Elastic IP: Go to Elastic IPs and allocate a new one.
* Launch a NAT Gateway:
* Go to NAT Gateways → Click Create NAT Gateway
* Select the public subnet.
* Attach the Elastic IP.
* Click Create.
* Update Route Table for Private Subnet
* Go to Route Tables → Select the private subnet’s route table.
* Edit Routes:
* Add a new route: 0.0.0.0/0 → Target: NAT Gateway.
* Save changes



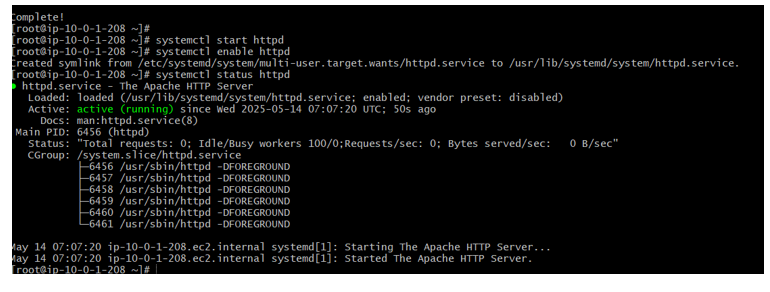
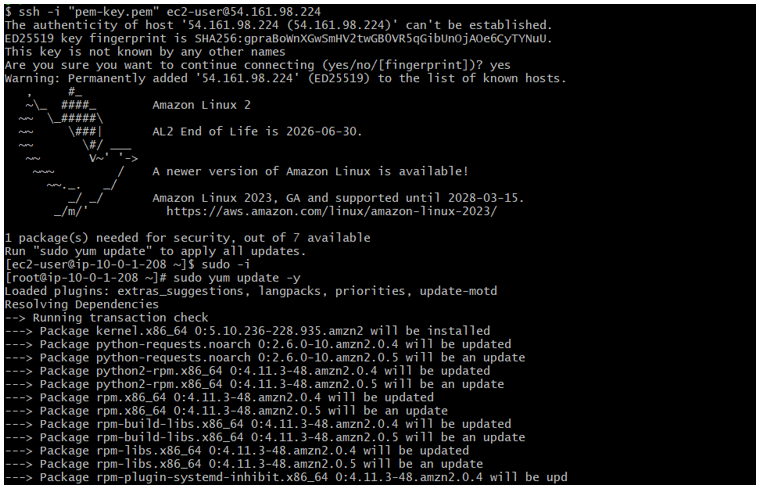


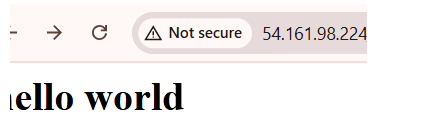


* Test Connectivity
* Launch an EC2 instance in the private subnet.
* SSH into a public instance (if needed)

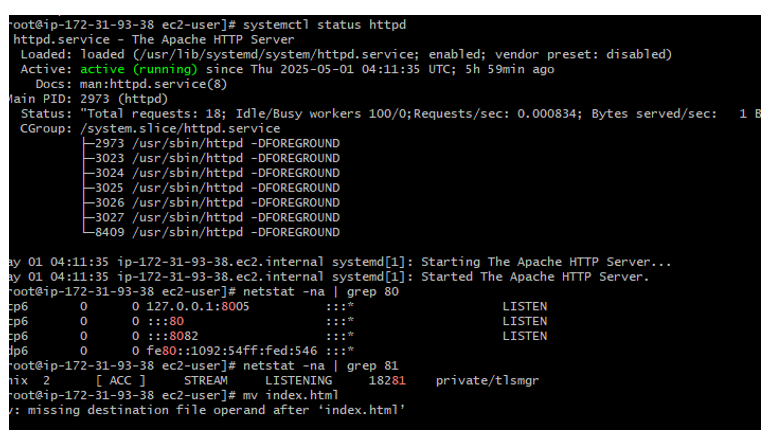
6) Create Two instances, one in public subnet and one in private subnet.

* Create and launce an instance like usual, just select your   
  VPC and public subnet in the settings
* and shown in the browser
* systemctl start httpd
* System status httpd





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



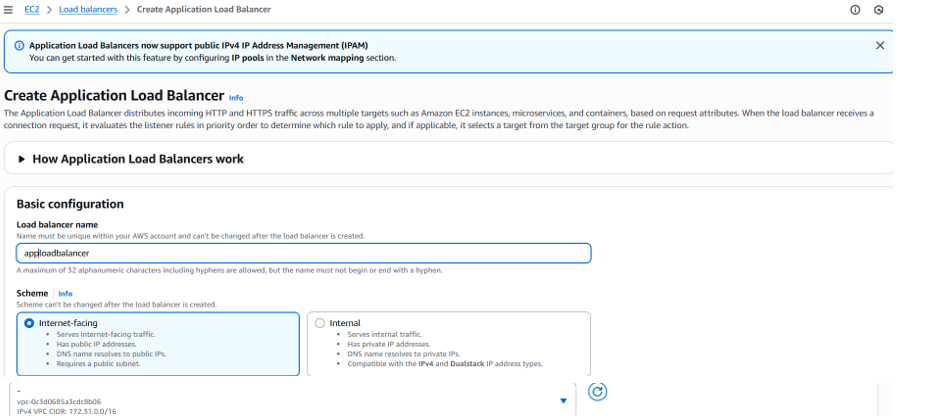
8) Create one application load balancer and attach the load balancer to both the ec2 instances.

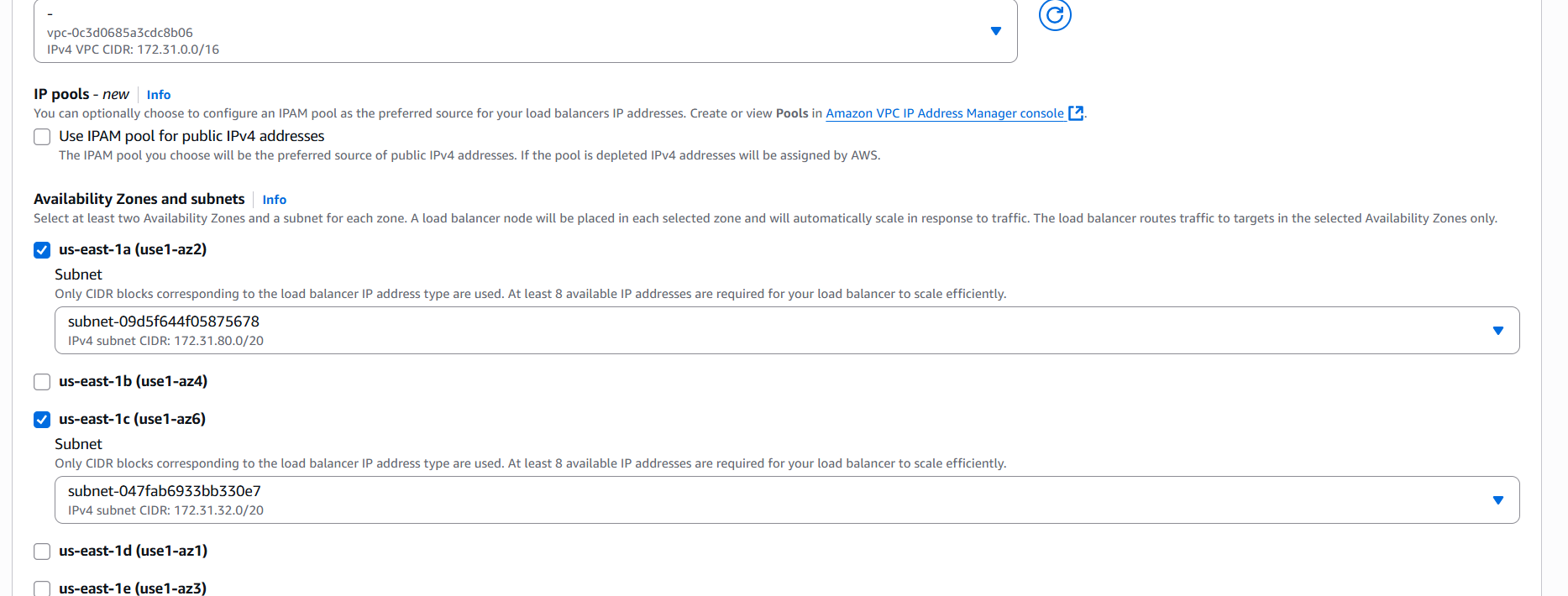
**Step 1: Create a Target Group**

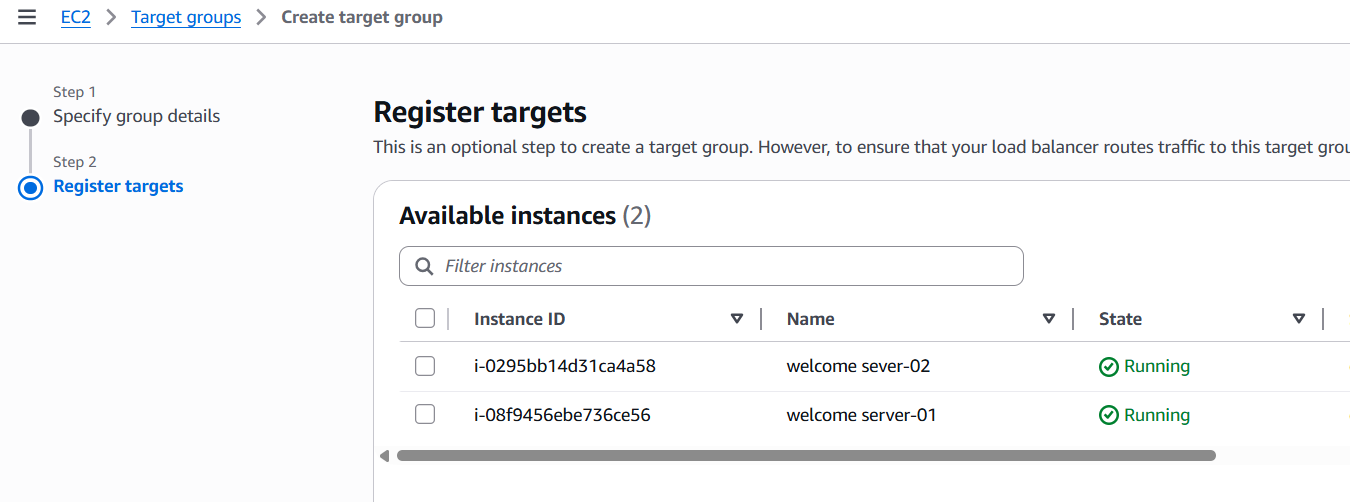
1. **Access the EC2 Console**: Navigate to the [Amazon EC2 console](https://console.aws.amazon.com/ec2/).
2. **Create Target Group**:
   * In the navigation pane, under **Load Balancing**, choose **Target Groups**.
   * Click **Create target group**.
   * For **Target type**, select **Instances**.
   * Enter a **Target group name** (e.g., my-target-group).
   * Set the **Protocol** to **HTTP** and **Port** to **80**.
   * Choose the appropriate **VPC**.
   * Click **Next**.
3. **Register Targets**:
   * Select the two EC2 instances you want to include.
   * Click **Include as pending below**.
   * Click **Create target group**.

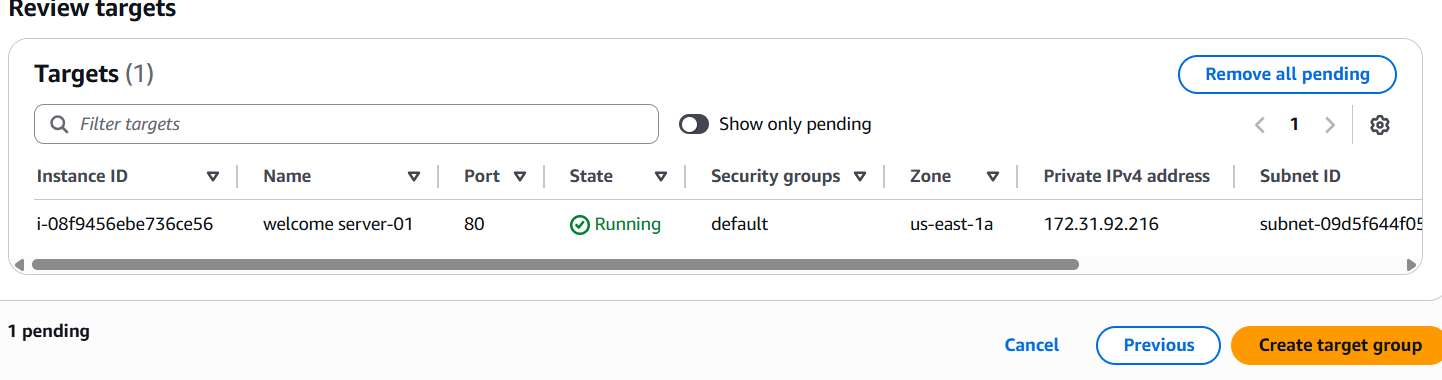
**Step 2: Create an Application Load Balancer**

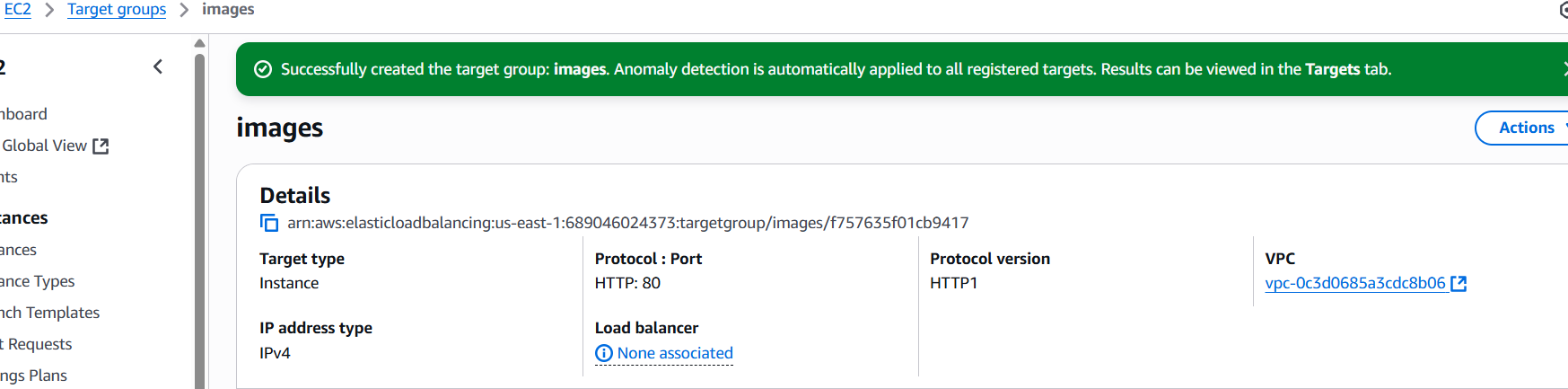
1. **Access Load Balancers**:
   * In the EC2 console, under **Load Balancing**, choose **Load Balancers**.
   * Click **Create Load Balancer**.
   * Select **Application Load Balancer**.
2. **Configure Load Balancer**:
   * Enter a **Name** for your load balancer (e.g., my-alb).
   * Choose **Scheme** as **Internet-facing** if you want it accessible from the internet.
   * Select the appropriate **IP address type** (IPv4 or dualstack).
   * Under **Network mapping**, select at least two **Availability Zones** and their corresponding **Subnets**.
3. **Configure Security Settings**:
   * Under **Security groups**, select an existing security group or create a new one that allows inbound HTTP (port 80) traffic.
4. **Configure Listeners and Routing**:
   * Under **Listeners**, ensure a listener is set for **HTTP** on port **80**.
   * For **Default action**, select **Forward to** and choose the target group you created earlier.
5. **Review and Create**:
   * Review all settings.
   * Click **Create load balancer**

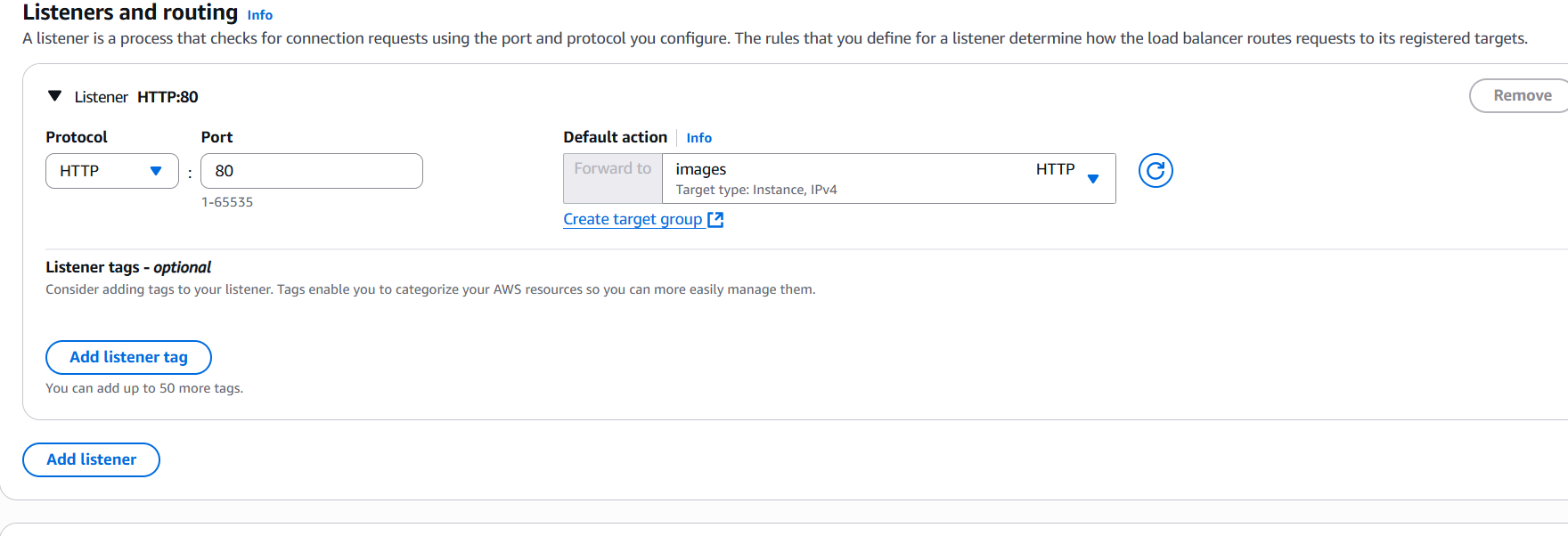


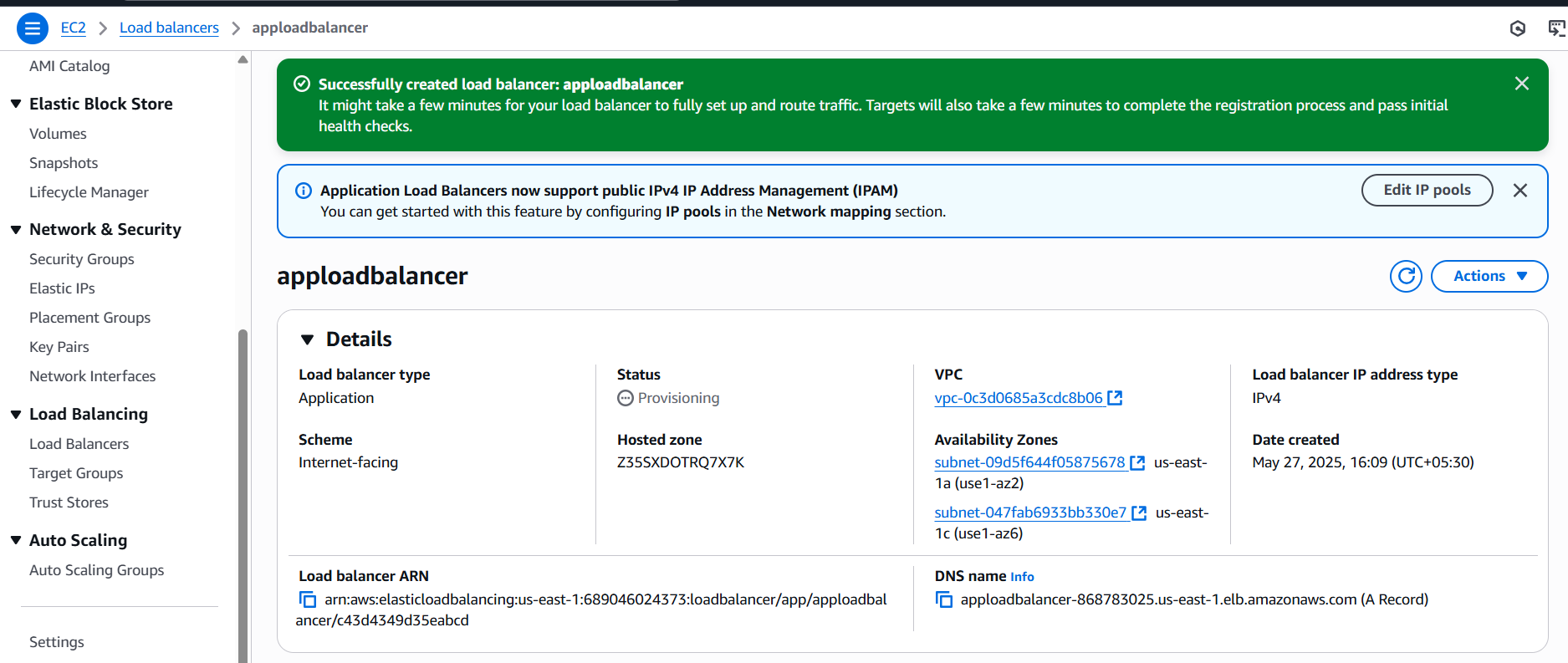


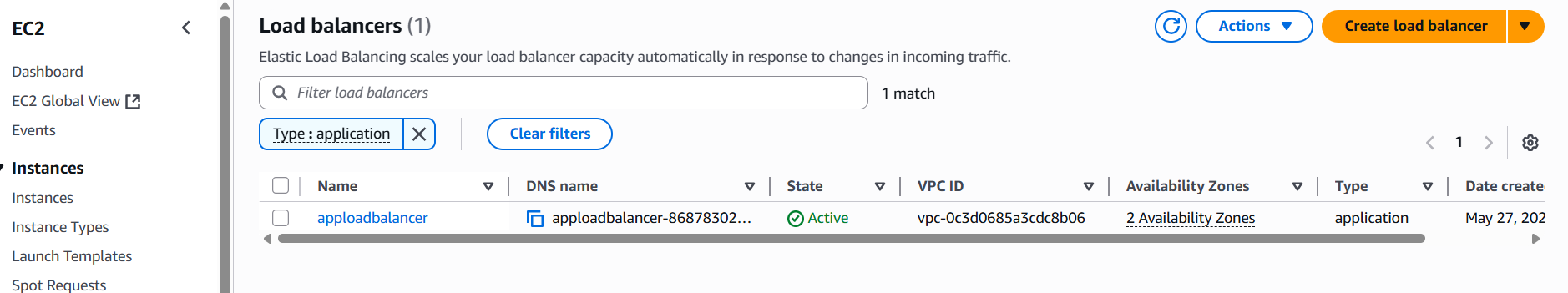












9) Store Application load balancer logs to s3.

**Step 1: Create an S3 Bucket**

1. Navigate to the [Amazon S3 console](https://console.aws.amazon.com/s3/).
2. Click **Create bucket**.
3. Enter a unique **Bucket name**.
4. Select the **AWS Region** where your ALB is deployed.
5. Under **Default encryption**, choose **Amazon S3-managed keys (SSE-S3)**.
6. Click **Create bucket**. *Note: The bucket must be in the same Region as your ALB.*

**✅ Step 2: Attach a Bucket Policy**

Grant Elastic Load Balancing permission to write logs to your bucket:

1. In the S3 console, select your bucket.
2. Go to the **Permissions** tab and click **Bucket policy**.
3. Add the following policy, replacing your-bucket-name, your-region, and your-account-id with your details:

json

CopyEdit

{

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

"Statement": [

{

"Sid": "ELBAccessLogsPolicy",

"Effect": "Allow",

"Principal": {

"Service": "elasticloadbalancing.amazonaws.com"

},

"Action": "s3:PutObject",

"Resource": "arn:aws:s3:::your-bucket-name/AWSLogs/your-account-id/\*",

"Condition": {

"StringEquals": {

"AWS:SourceAccount": "your-account-id"

},

"ArnLike": {

"AWS:SourceArn": "arn:aws:elasticloadbalancing:your-region:your-account-id:loadbalancer/app/\*"

}

}

}

]

}

1. Click **Save changes**.

*Note: Ensure the bucket policy grants the necessary permissions for Elastic Load Balancing to write logs.*

**Step 3: Enable Access Logs on Your ALB**

1. Navigate to the [Amazon EC2 console](https://console.aws.amazon.com/ec2/).
2. In the left navigation pane, choose **Load Balancers**.
3. Select your ALB.
4. Go to the **Description** tab and click **Edit attributes**.
5. Under **Access logs**, select **Enable**.
6. Enter the **S3 bucket name** and an optional **prefix** (e.g., my-alb-logs/).
7. Click **Save**.

**Create an S3 Bucket**

* **Region**: Ensure the bucket is in the same AWS Region as your ALB.
* **Encryption**: Use Amazon S3-managed keys (SSE-S3) for server-side encryption.
* **Naming**: Choose a unique bucket name.

**2. Attach a Bucket Policy**

Grant Elastic Load Balancing (ELB) permission to write logs to your S3 bucket by adding a bucket policy.

Here's an example policy for the Asia Pacific (Hyderabad) Region

json

CopyEdit

{

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

"Statement": [

{

"Effect": "Allow",

"Principal": {

"AWS": "arn:aws:iam::127311923021:root"

},

"Action": "s3:PutObject",

"Resource": "arn:aws:s3:::your-bucket-name/prefix/AWSLogs/your-account-id/\*"

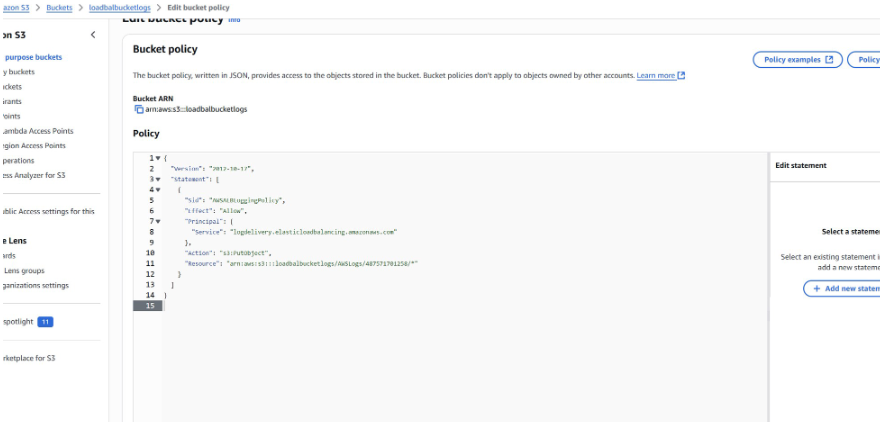
}

]

}

**Replace:**

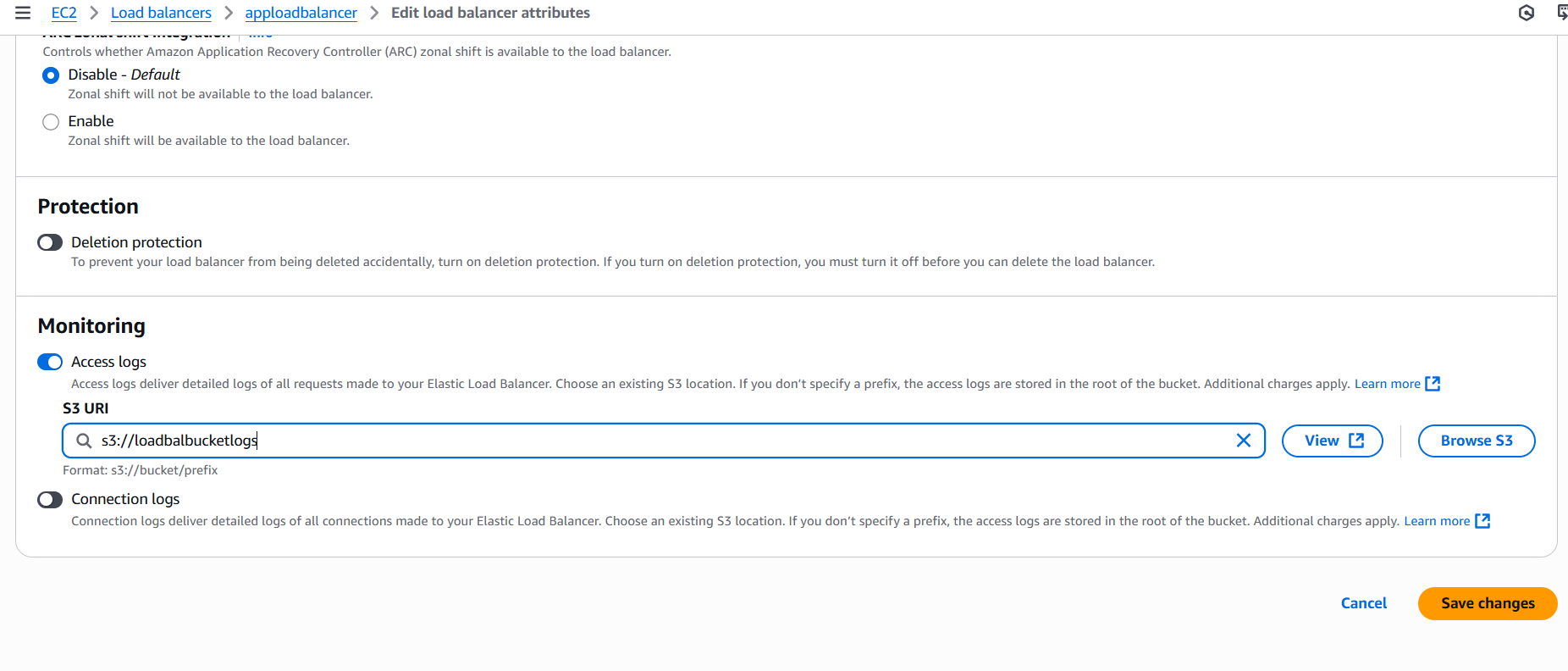
* your-bucket-name with your S3 bucket name.
* prefix with your desired log prefix (optional).
* your-account-id with your AWS account ID.



**3. Enable Access Logs on Your ALB**

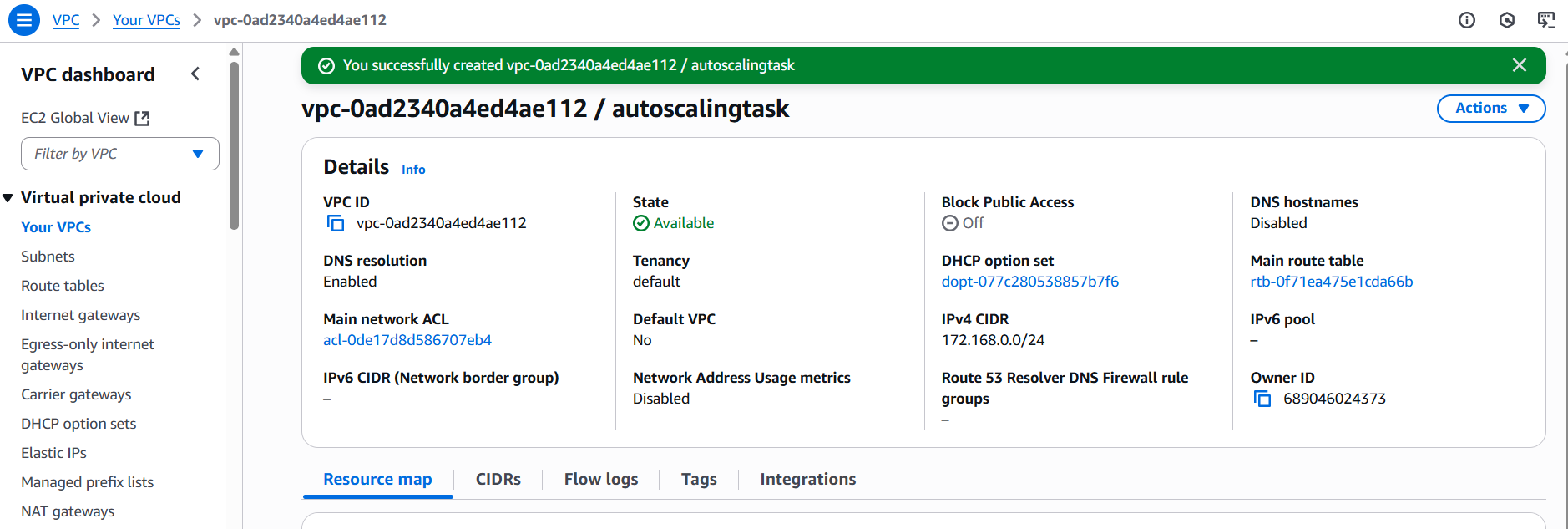
**Using the AWS Console:**

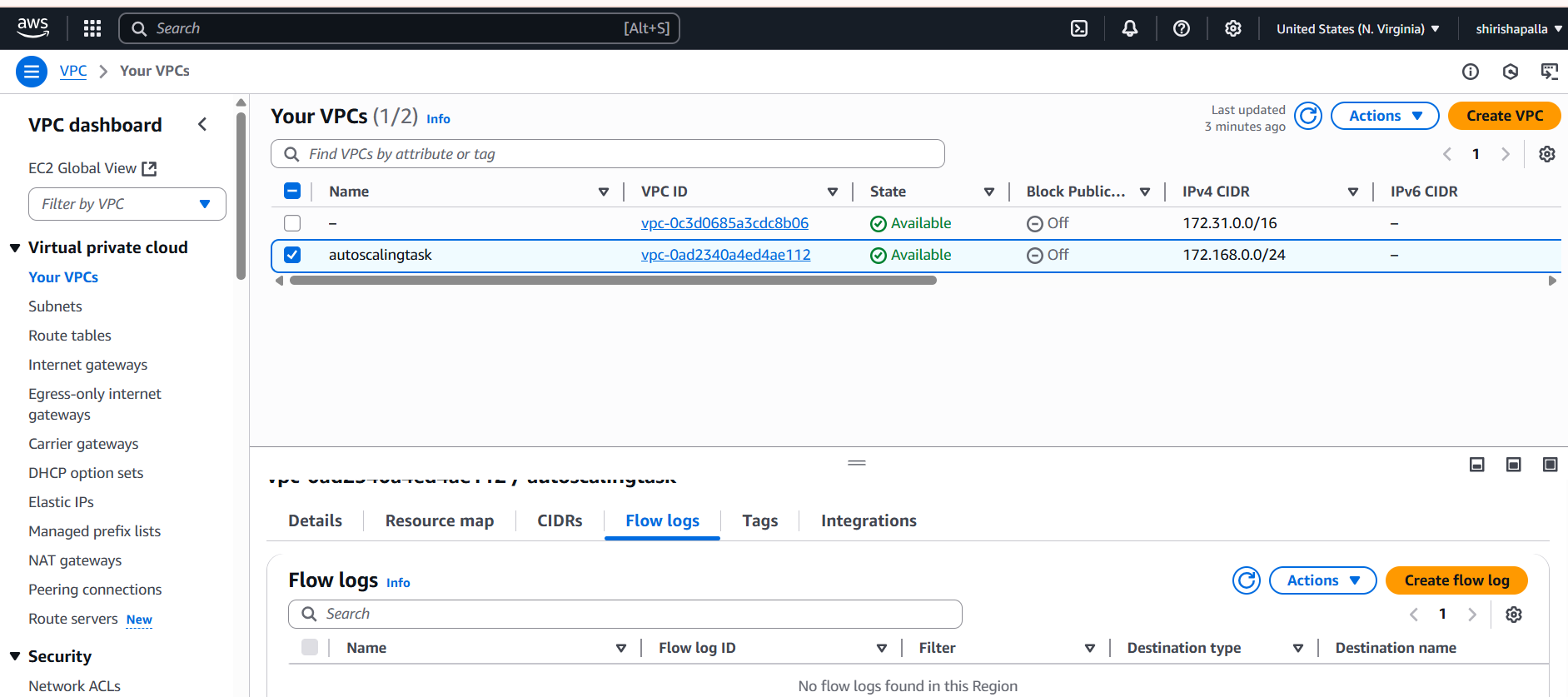
1. Navigate to the EC2 service.
2. In the left pane, select **Load Balancers**.
3. Choose your ALB and go to the **Attributes** tab.
4. Click **Edit attributes**.
5. Enable **Access logs**.
6. Specify the S3 URI in the format:
   * With prefix: s3://your-bucket-name/prefix
   * Without prefix: s3://your-bucket-name
7. Save the changes.

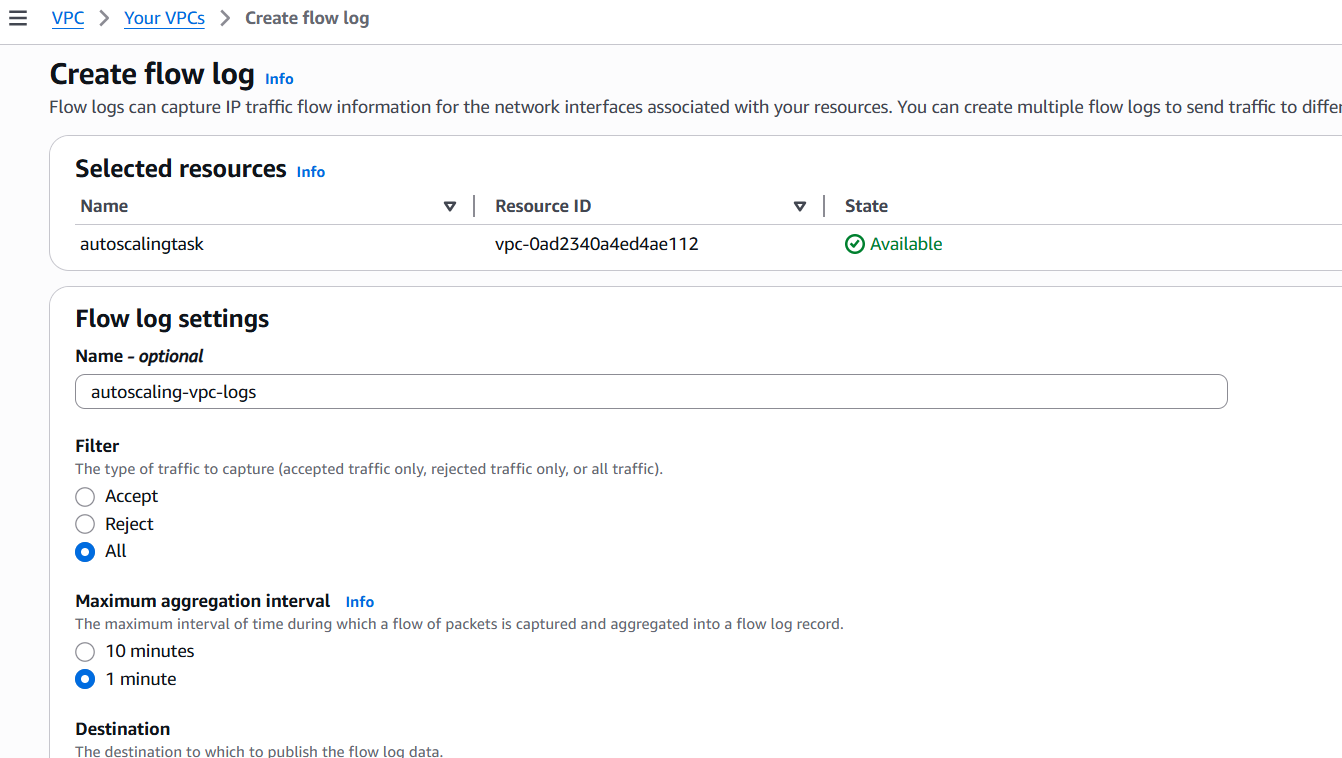


10) Store the vpc flow logs to CloudWatch group.

1. Go to the VPC dashboard.  
2. Select the VPC.  
3. Click on "Flow logs" in the sidebar.  
4. Click "Create flow log"





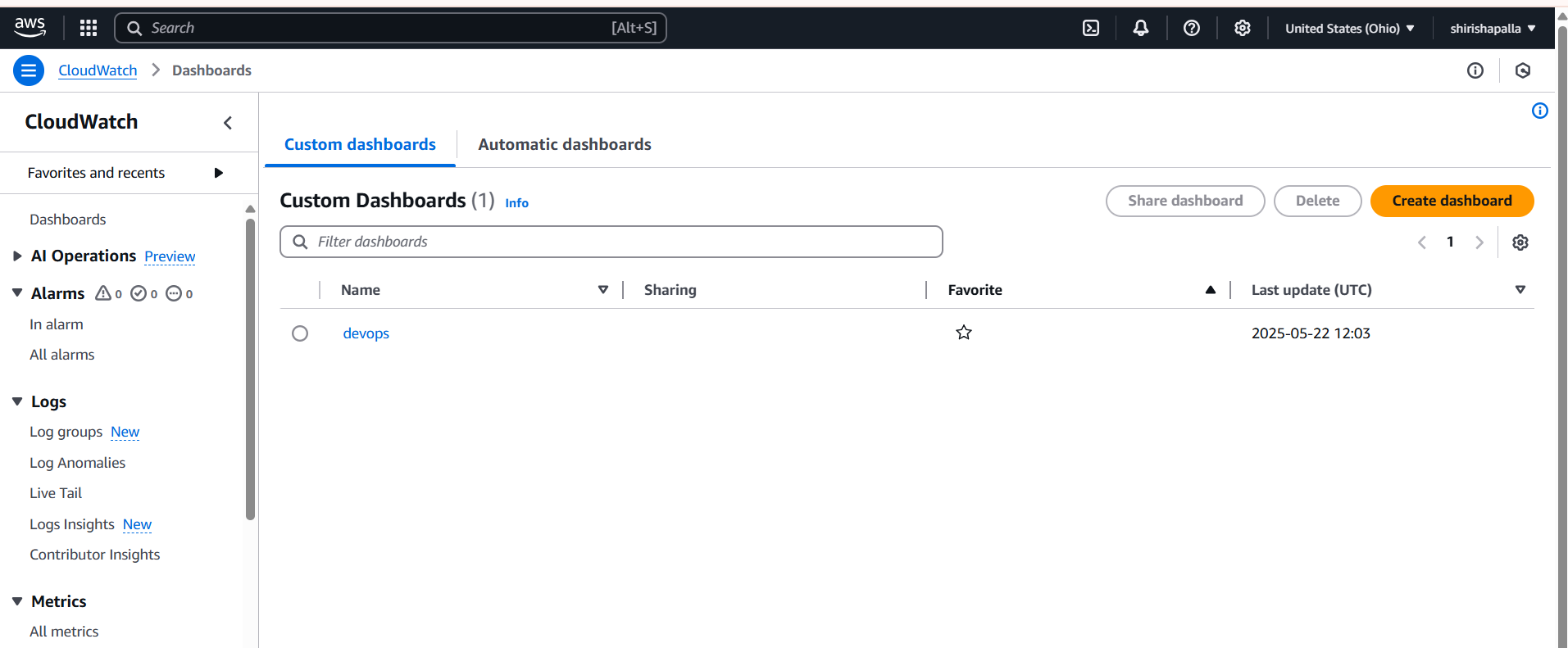


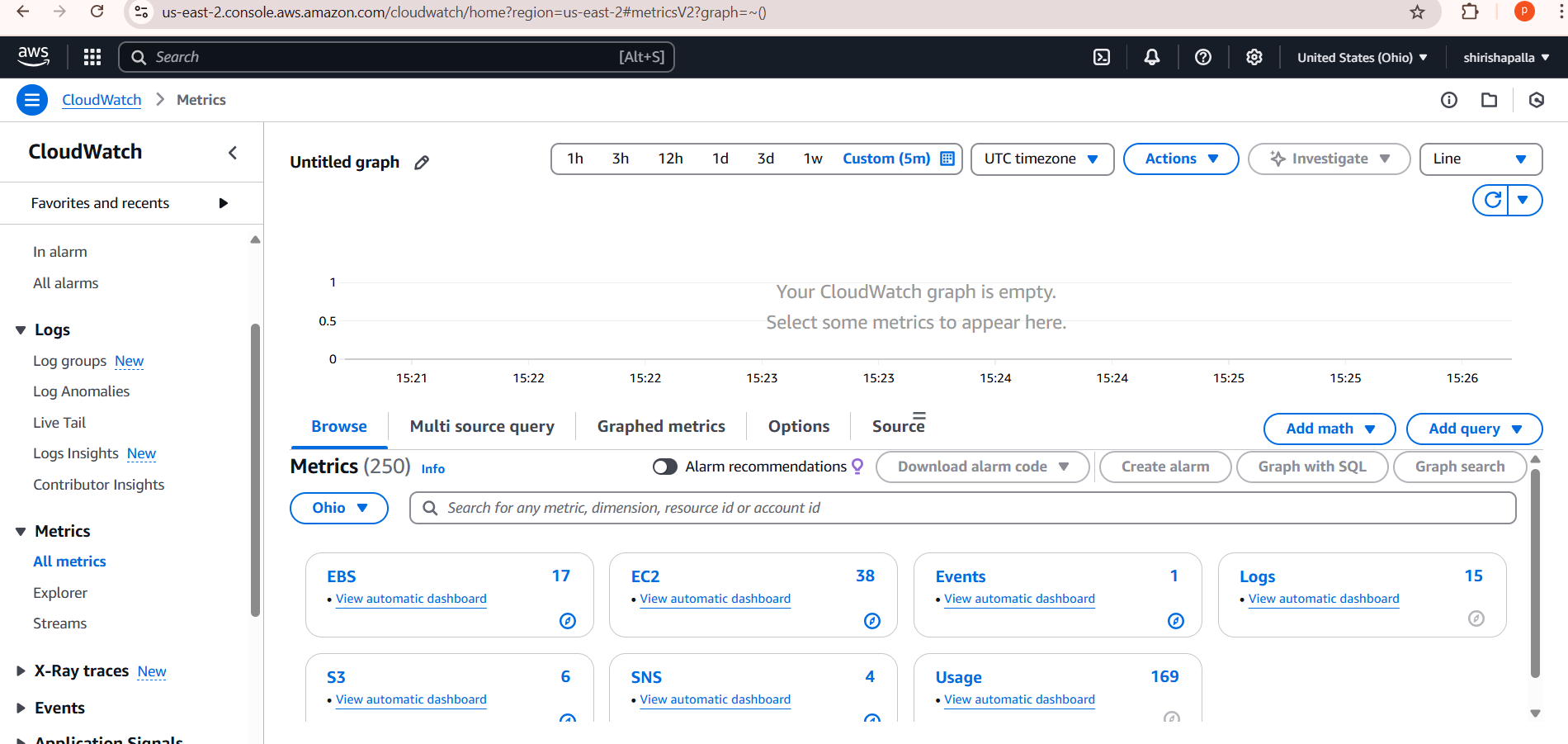
11) Create Monitoring Dashboards to monitor cpu utilization and to monitor apache service.

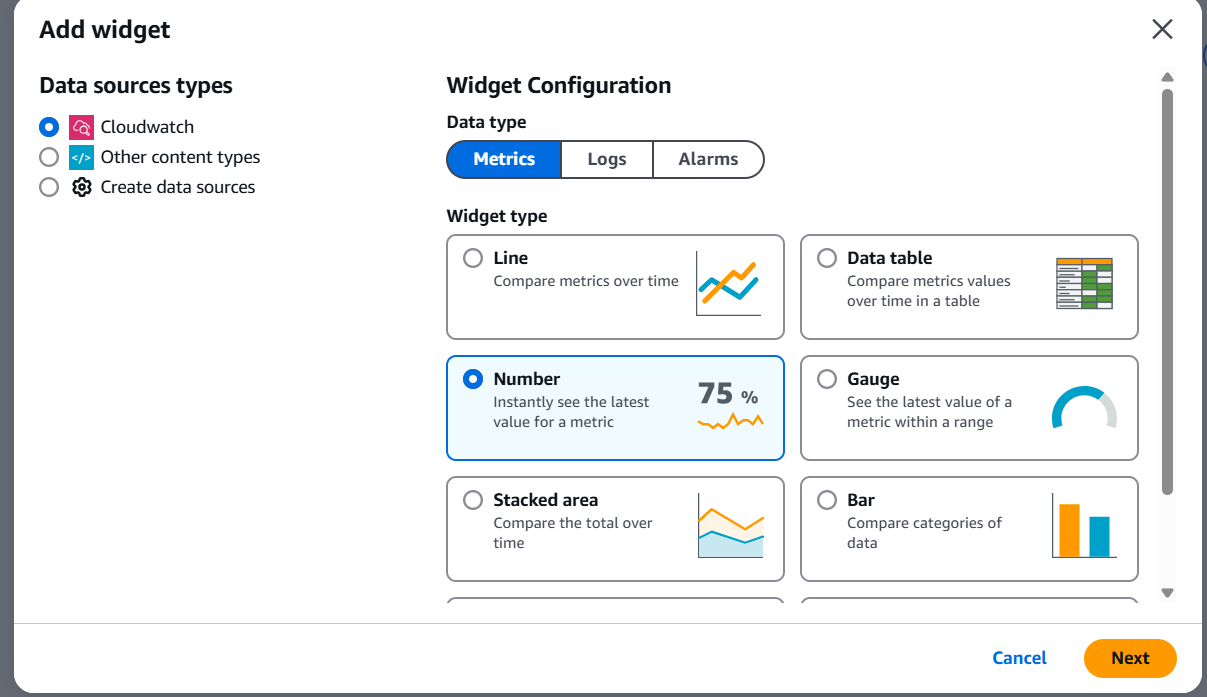
Launch an EC2 Instance (If Not Already Done)  
If you already have an EC2 instance, skip to Step 2.  
1. Go to the EC2 Console →  
2. Click Launch Instance  
3. Choose an AMI (Amazon Linux 2 recommended)  
4. Choose an instance type (e.g., t2.micro)  
5. Configure and launch it.  
To enable detailed monitoring:  
1. Go to the EC2 Console  
2. Click on your instance

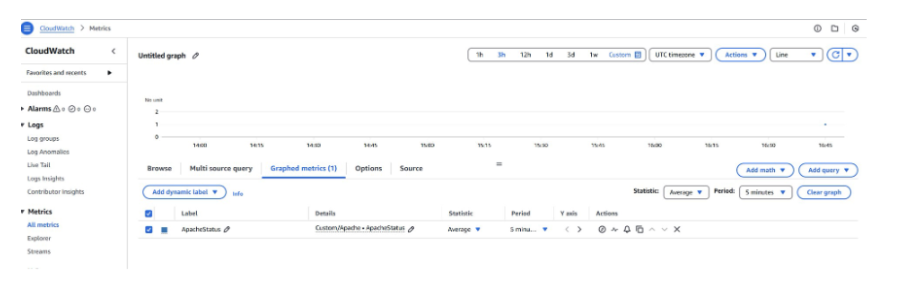
3. Click Actions → Monitor and troubleshoot → Modify   
monitoring  
4. Check Enable detailed monitoring  
5. Click Apply

To view them:  
1. Go to the CloudWatch Console →  
2. In the left menu, choose Metrics  
3. Click EC2 → Per-Instance Metrics  
4. Select your instance ID to see available metrics











12) CPU utilizationis more than 70% then it should triggere Autoscaling and launch new instance.

**Step-by-Step Guide:**

1. **Create a Launch Template:**
   * Navigate to the EC2 console.
   * Define the AMI, instance type, key pair, security groups, and other configurations.
2. **Create an Auto Scaling Group (ASG):**
   * Use the launch template created above.
   * Specify the desired number of instances, minimum, and maximum capacity.
   * Select the appropriate VPC and subnets.
3. **Set Up a CloudWatch Alarm:**
   * Go to the CloudWatch console.
   * Create a new alarm based on the CPU Utilization metric.
   * Set the threshold to trigger when CPU utilization exceeds 70% for a specified period (e.g., 5 minutes).
4. **Define a Scaling Policy:**
   * In the ASG settings, add a scaling policy.
   * Choose between **Step Scaling** or **Target Tracking Scaling**:
     + **Step Scaling:** Define steps to add instances when CPU utilization crosses certain thresholds. For example, add 1 instance when CPU > 70%.
     + **Target Tracking Scaling:** Set a target CPU utilization (e.g., 70%), and AWS will automatically adjust the number of instances to maintain this target.
5. **Attach the Alarm to the Scaling Policy:**
   * Link the CloudWatch alarm to the scaling policy so that when the alarm is triggered, the scaling action is executed.
6. **Configure Instance Warm-Up Time:**
   * Set the warm-up time to ensure new instances are fully initialized before being considered in subsequent scaling decisions.

