Task on VPC

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

**Why Create VPC with Public and Private Subnets?**

**VPC (Virtual Private Cloud):** Your own isolated network inside AWS.

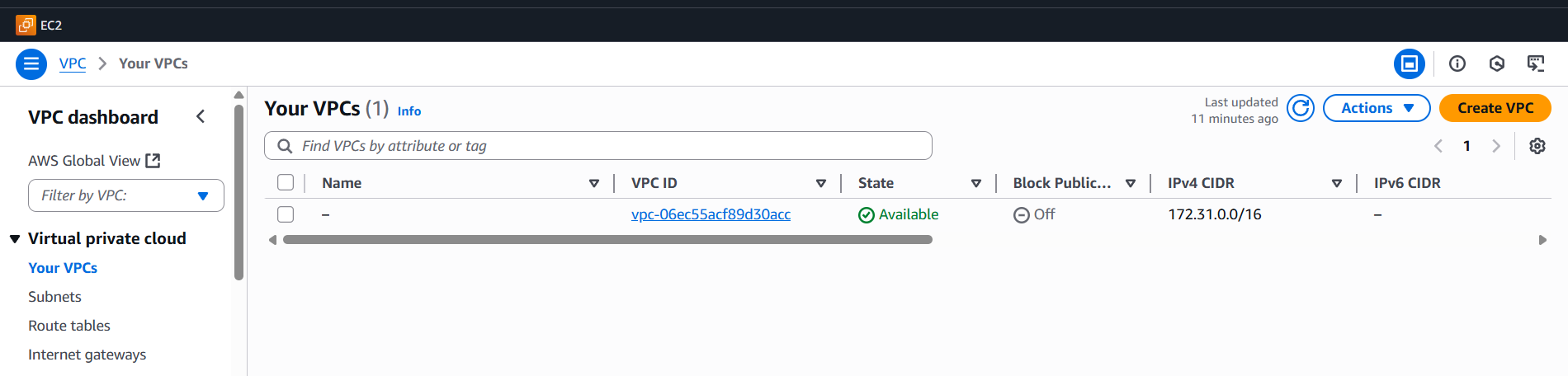
**Public Subnets:** Can access internet directly (via Internet Gateway). Good for web servers, load balancers, bastion hosts.

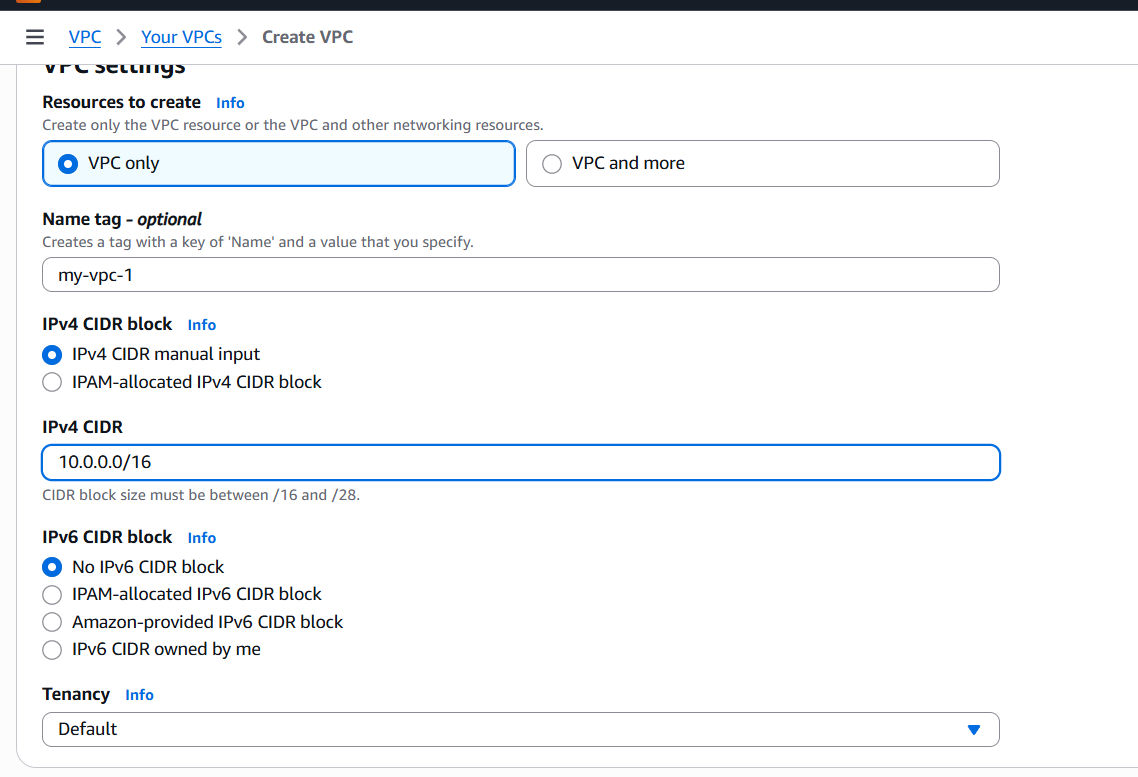
**Private Subnets:** No direct internet access. Good for databases, application servers, or backend systems (safer).

**Architecture Goal:**

* + 2 Availability Zones (for high availability).
  + 2 Public Subnets (for redundancy).
  + 2 Private Subnets (for backend).
  + Public Subnets connect through **Internet Gateway**.
  + Private Subnets connect through **NAT Gateway** for secure outbound internet (patching, updates).

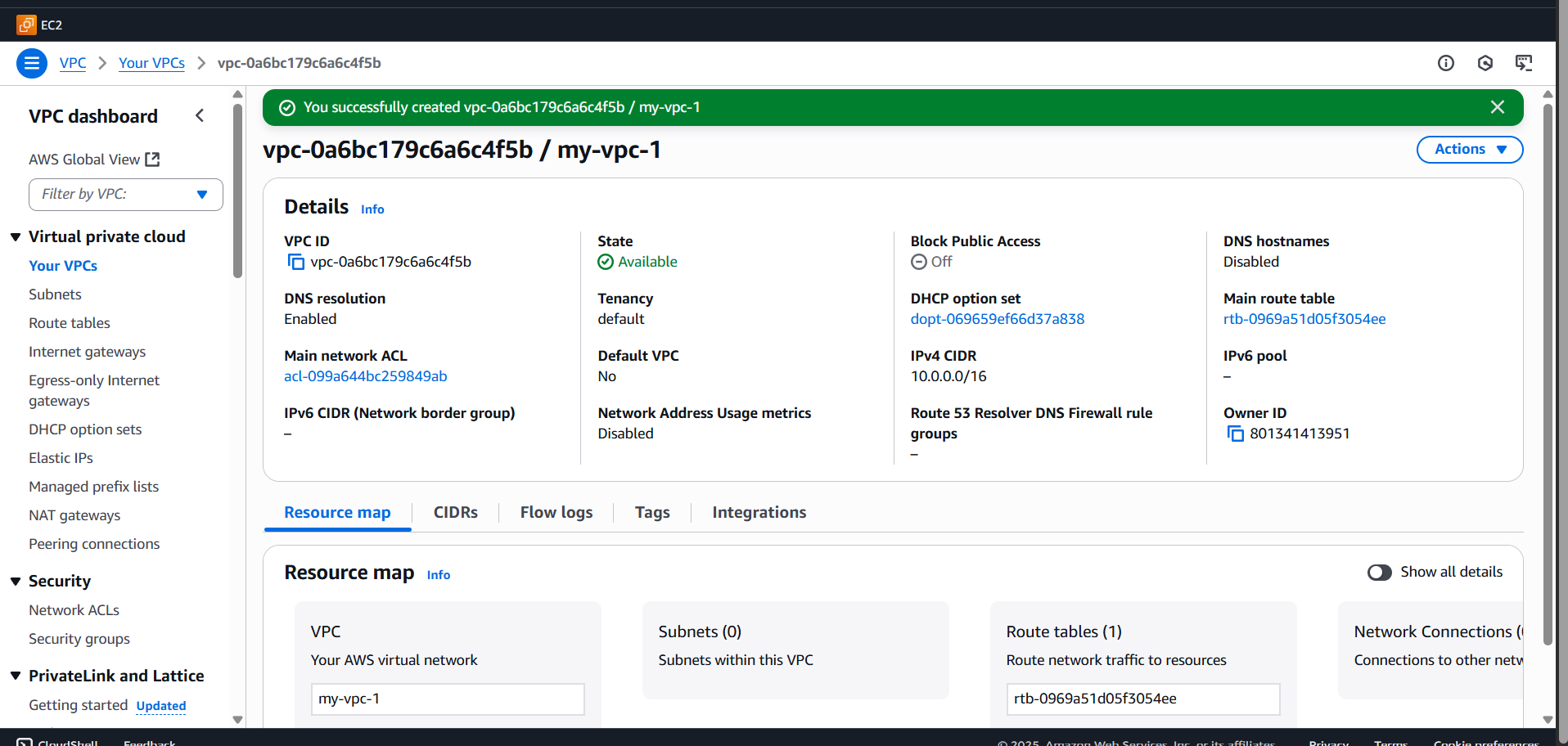
Go to VPC click create vpc





Give to default and choice no ipv6

Successful vpc opened



You’ll create **4 subnets (2 public, 2 private)** across **2 AZs**.

* **Public Subnet 1 (AZ1)** → 10.0.1.0/24
* **Public Subnet 2 (AZ2)** → 10.0.2.0/24
* **Private Subnet 1 (AZ1)** → 10.0.3.0/24
* **Private Subnet 2 (AZ2)** → 10.0.4.0/24

Give vpc id juct we have open one vpc

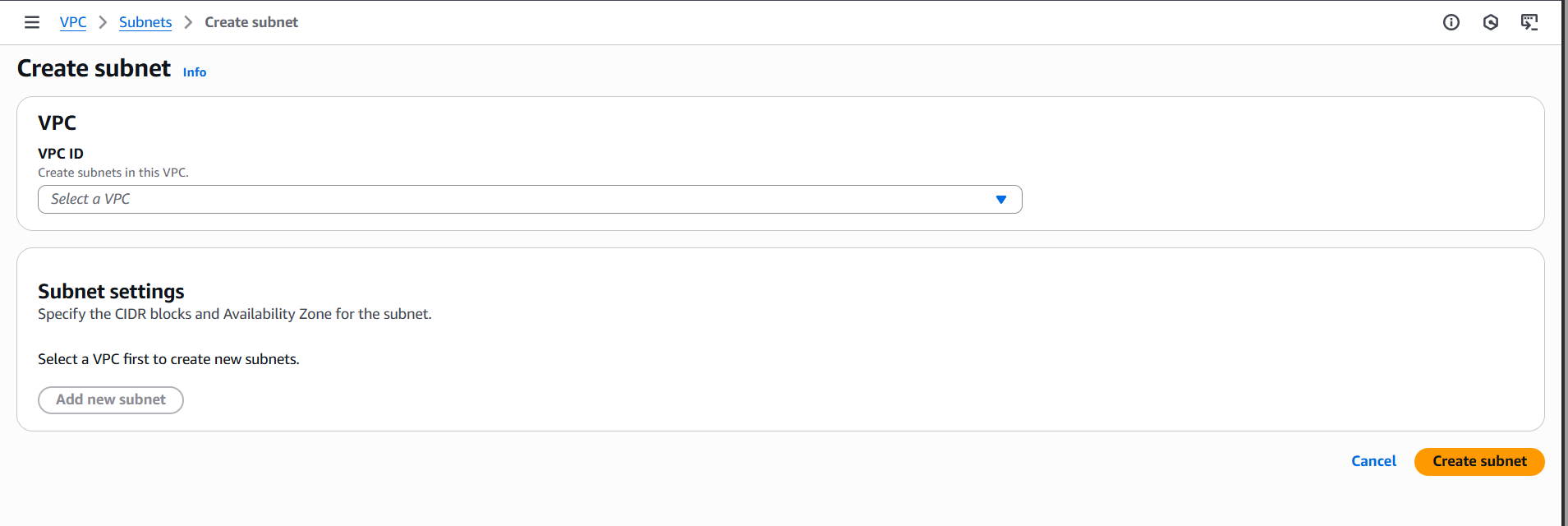
Go to **Subnets → Create Subnet**.

Select your VPC (My-VPC).

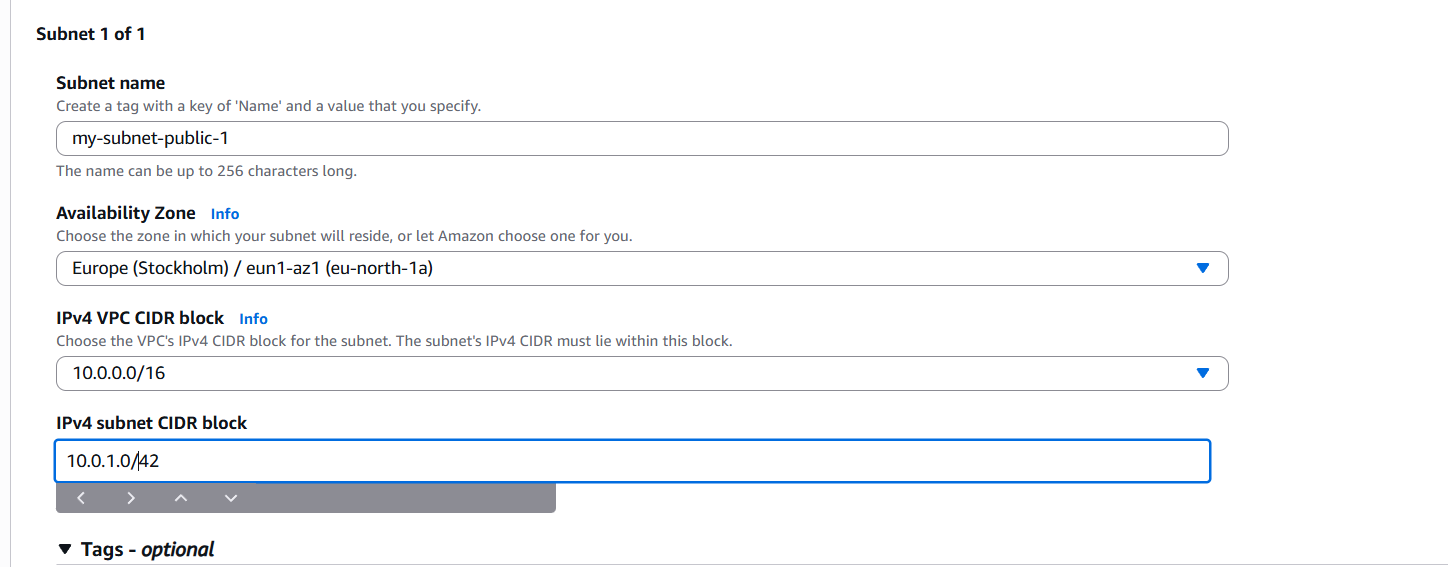
Add 4 subnets (choose different AZs for high availability).

Example:

* Subnet Name: Public-Subnet-AZ1, AZ: ap-south-1a, CIDR: 10.0.1.0/24.
* Repeat for other 3 subnets.

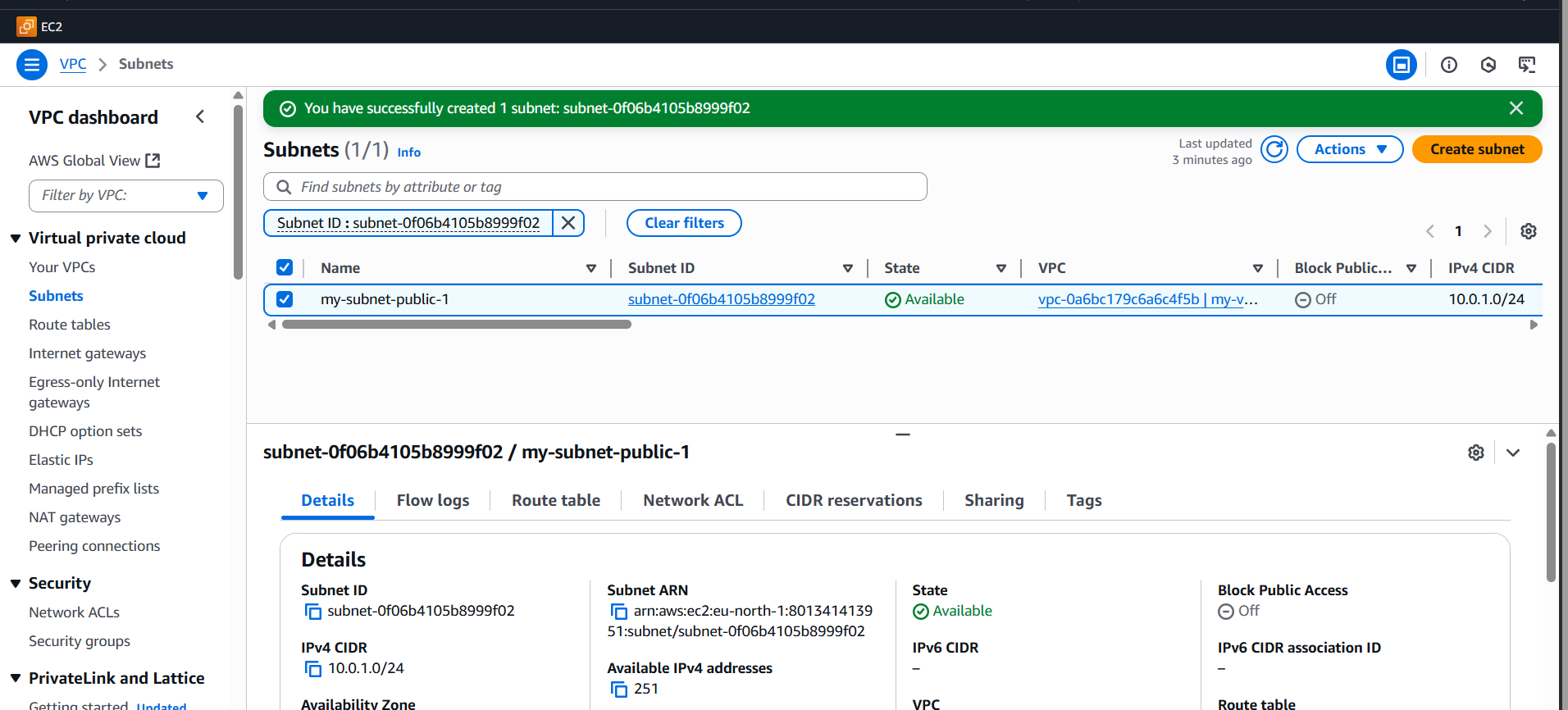


File the details, subnet name and which region you want and ipv4 address strat from

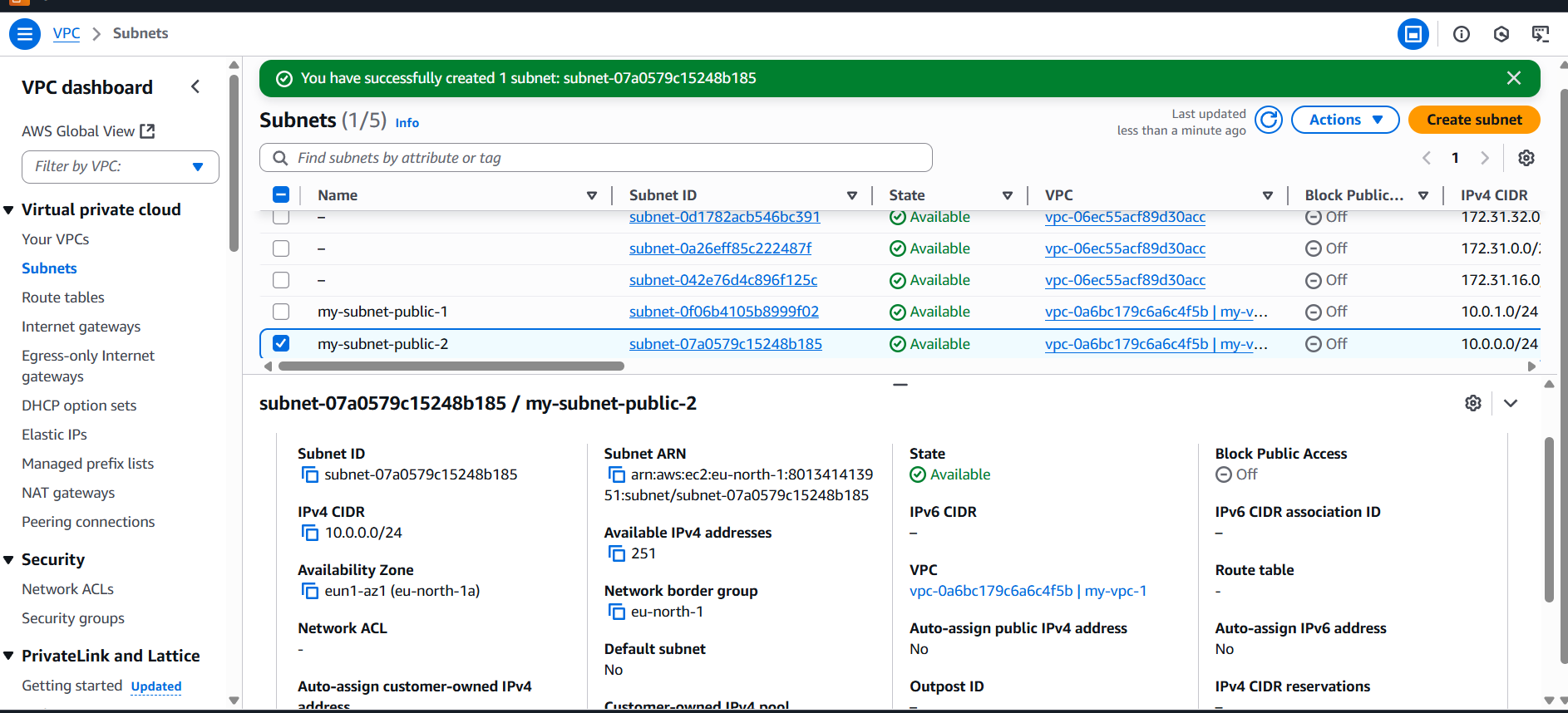


Subnet created successful

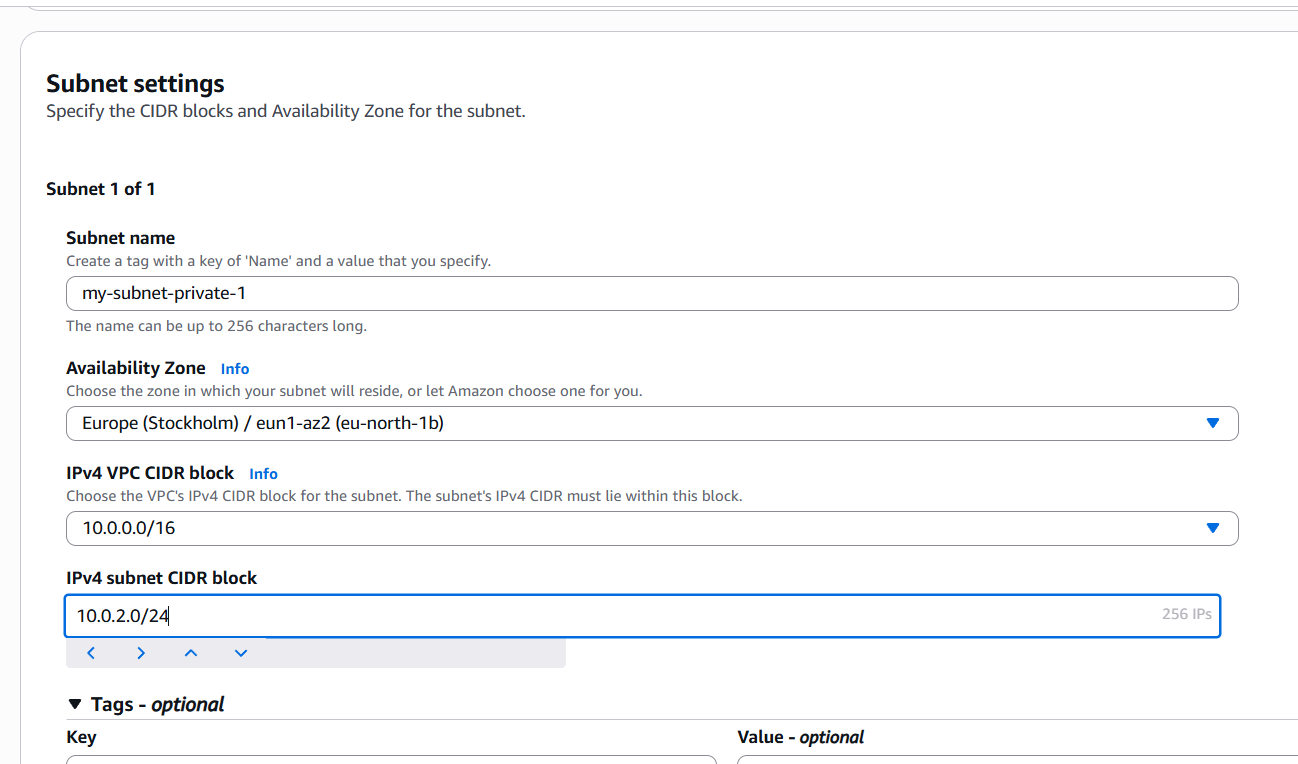
 **Public Subnet 1 (AZ1)** → 10.0.1.0/24



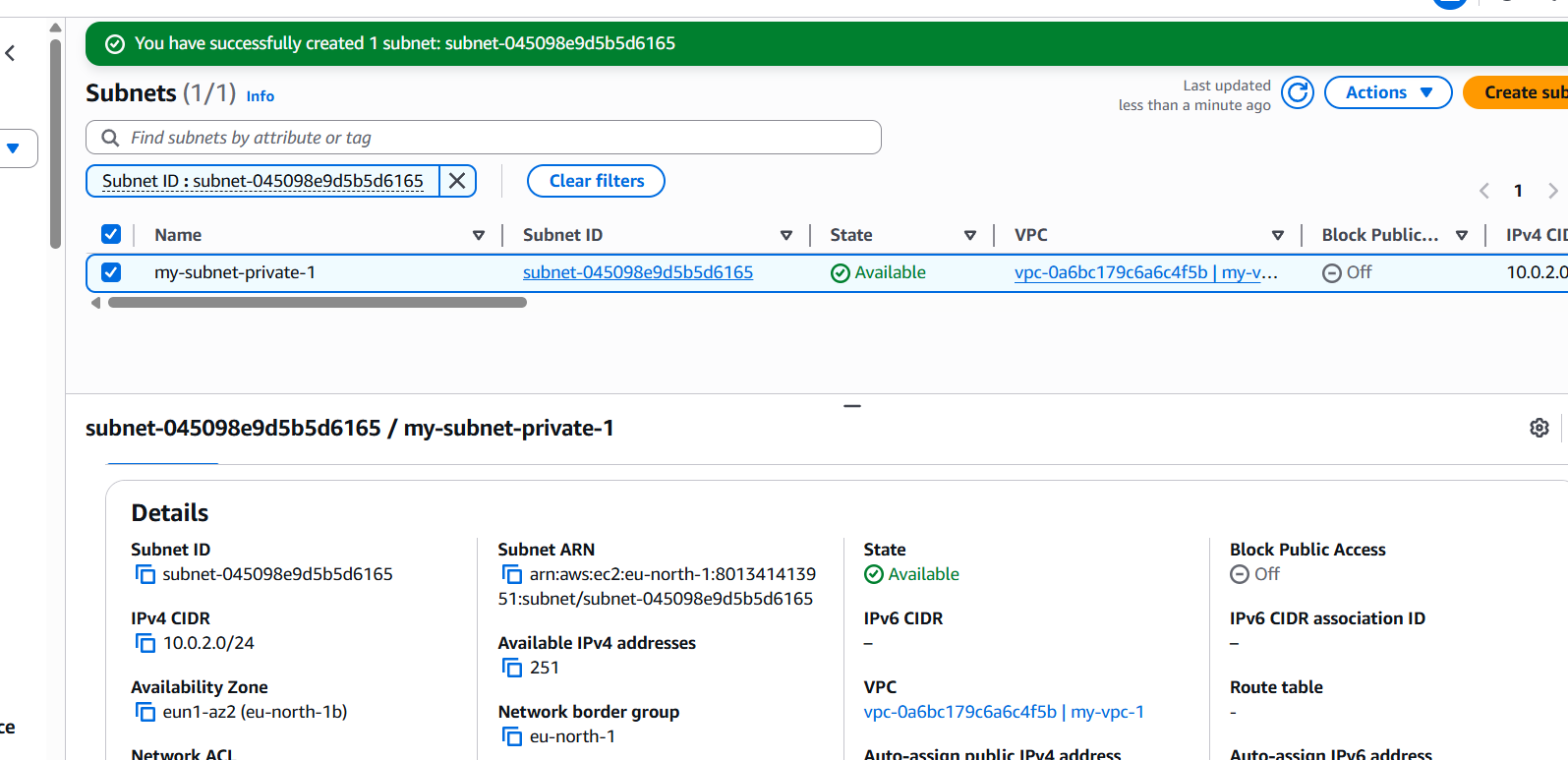
**Public Subnet 2 (AZ2)** → 10.0.2.0/24

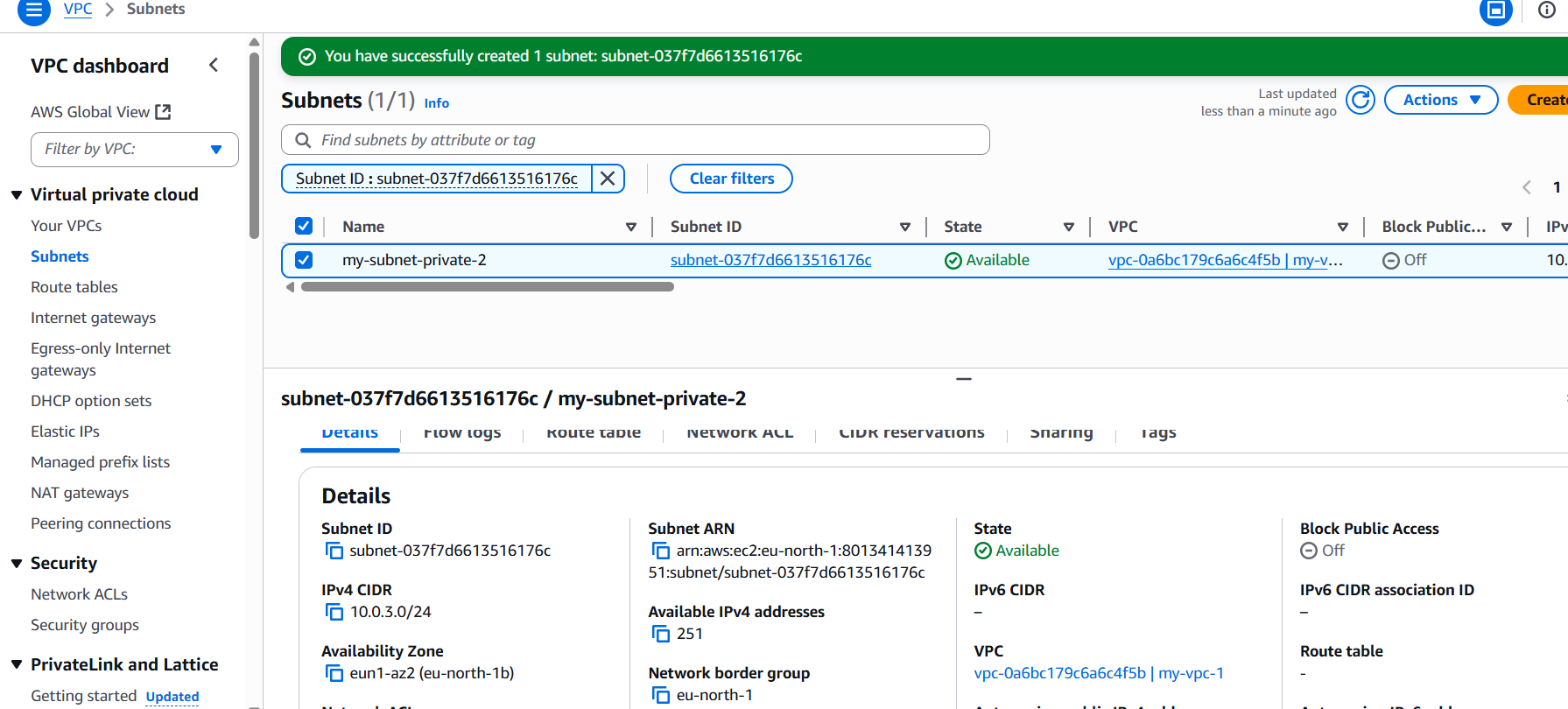


Subnet private 1



* **Private Subnet 1 (AZ1)** → 10.0.3.0/24
* **Private Subnet 2 (AZ2)** → 10.0.4.0/24

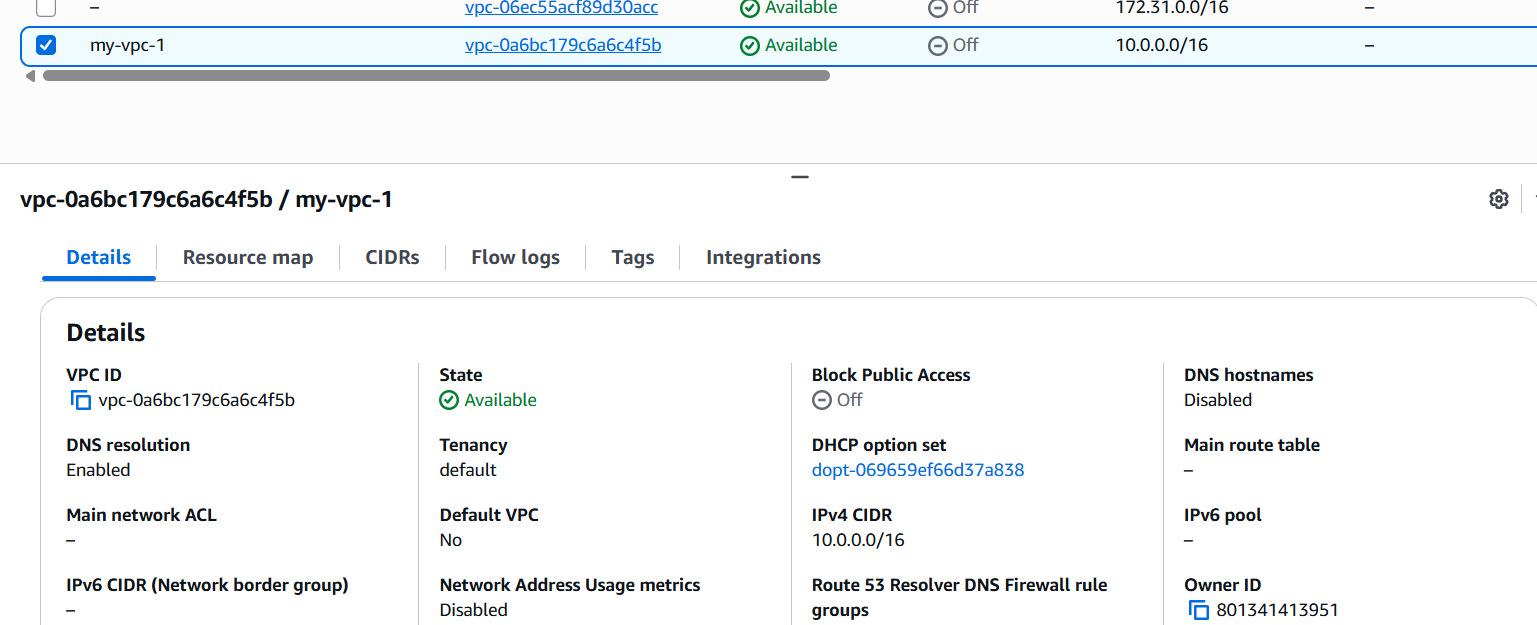




**Why Enable DNS Hostnames?**

* By default, VPC gives only **private IPs**.
* If you launch an instance in a public subnet, it will get a **public IP**, but without DNS hostnames you can’t connect using an easy name (only IP).
* Enabling it helps with:
  + SSH using public DNS names
  + Load balancers, Route 53, custom DNS setups
  + Easier management

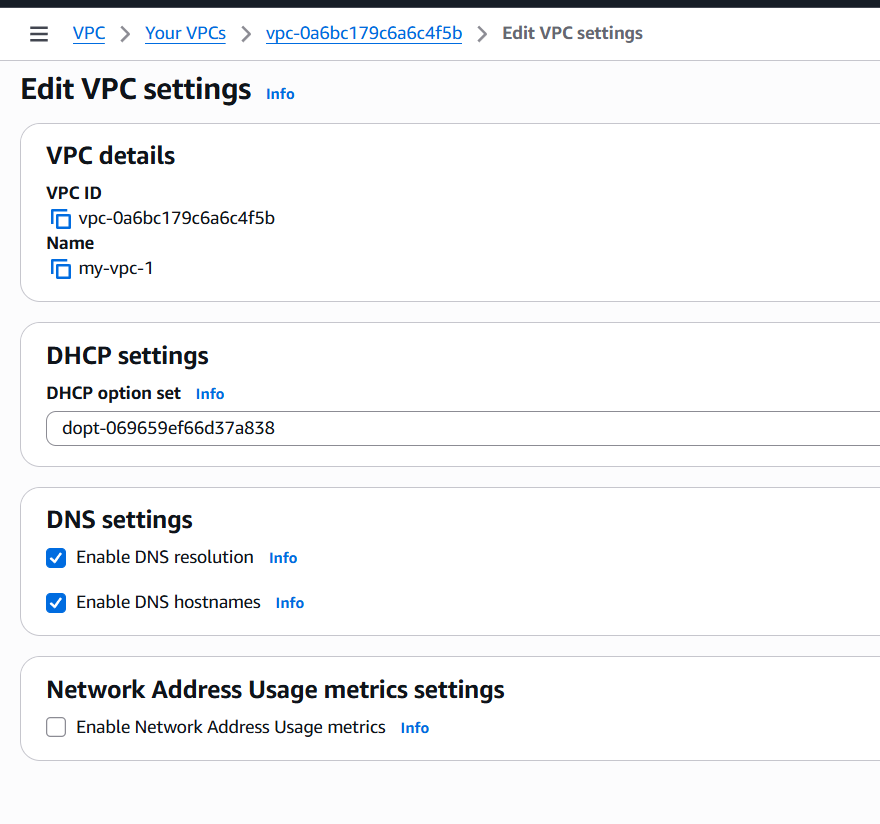
1. Enable DNS Hostname in VPC.



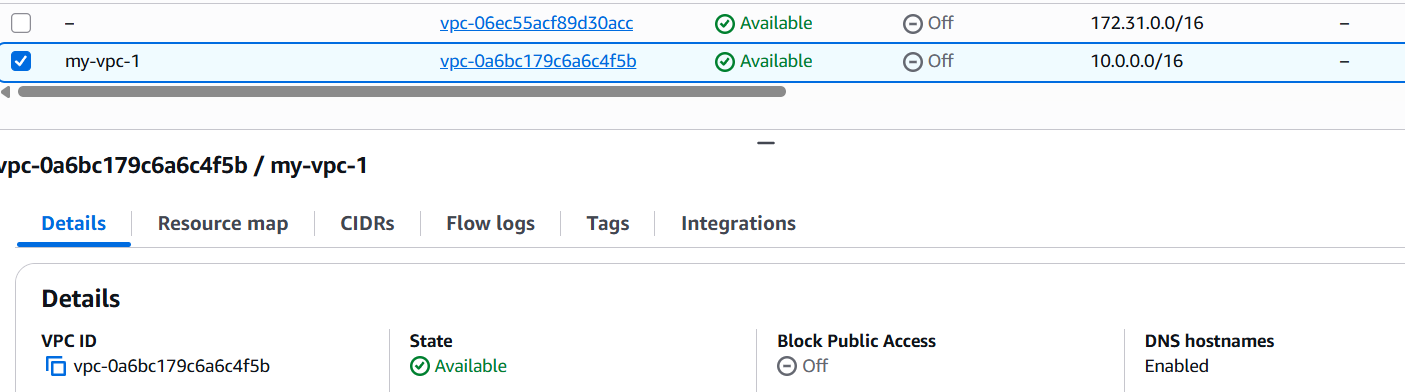
Click **Actions → Edit VPC settings**.

Tick:

* ✅ Enable DNS resolution
* ✅ Enable DNS hostnames
* Click and



Now Is enabled



Successfully enabled hostname

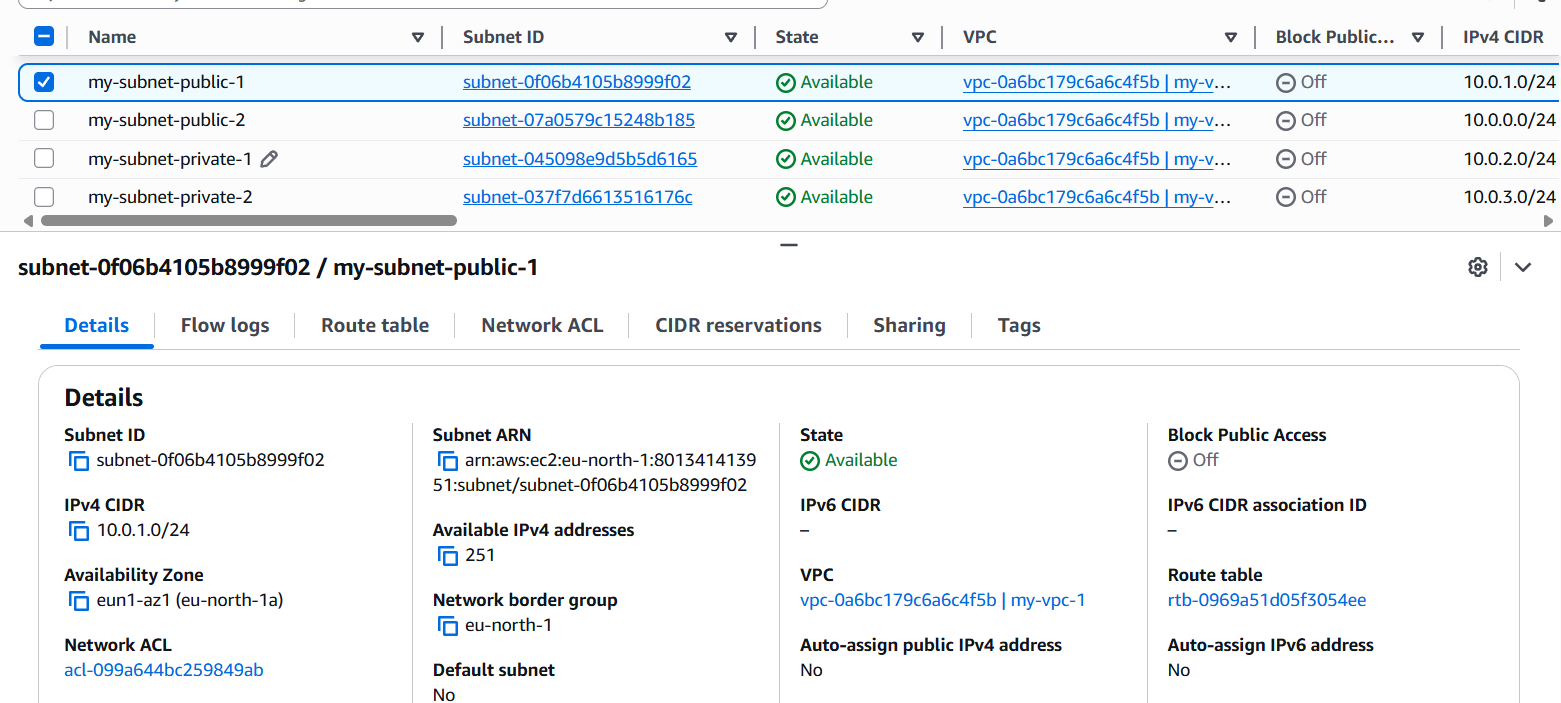
1. Enable Auto Assign Public IP in 2 public subnets.

**: Open VPC Service**

1. Go to **AWS Console → VPC service**.
2. In the left menu, click **Subnets**.

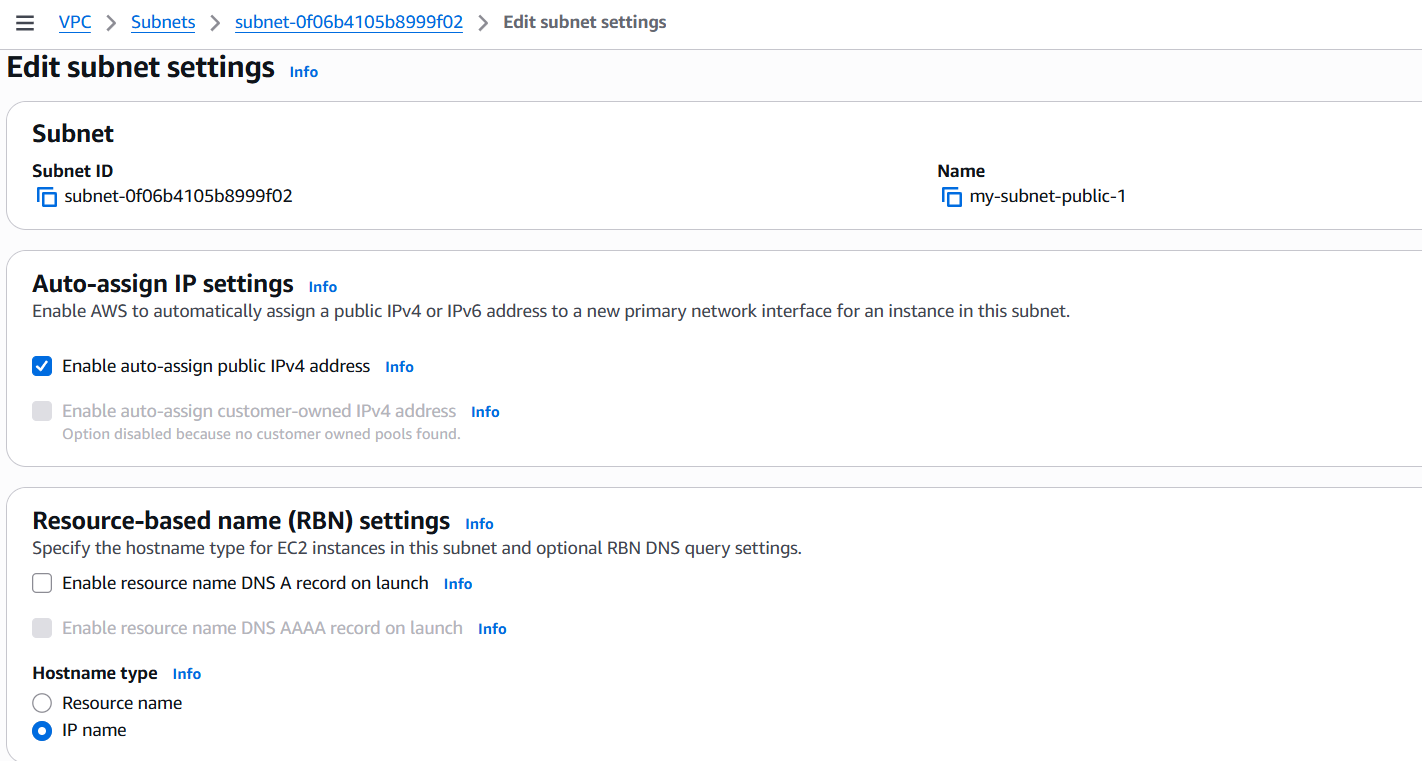
**Step 2: Select Public Subnet**

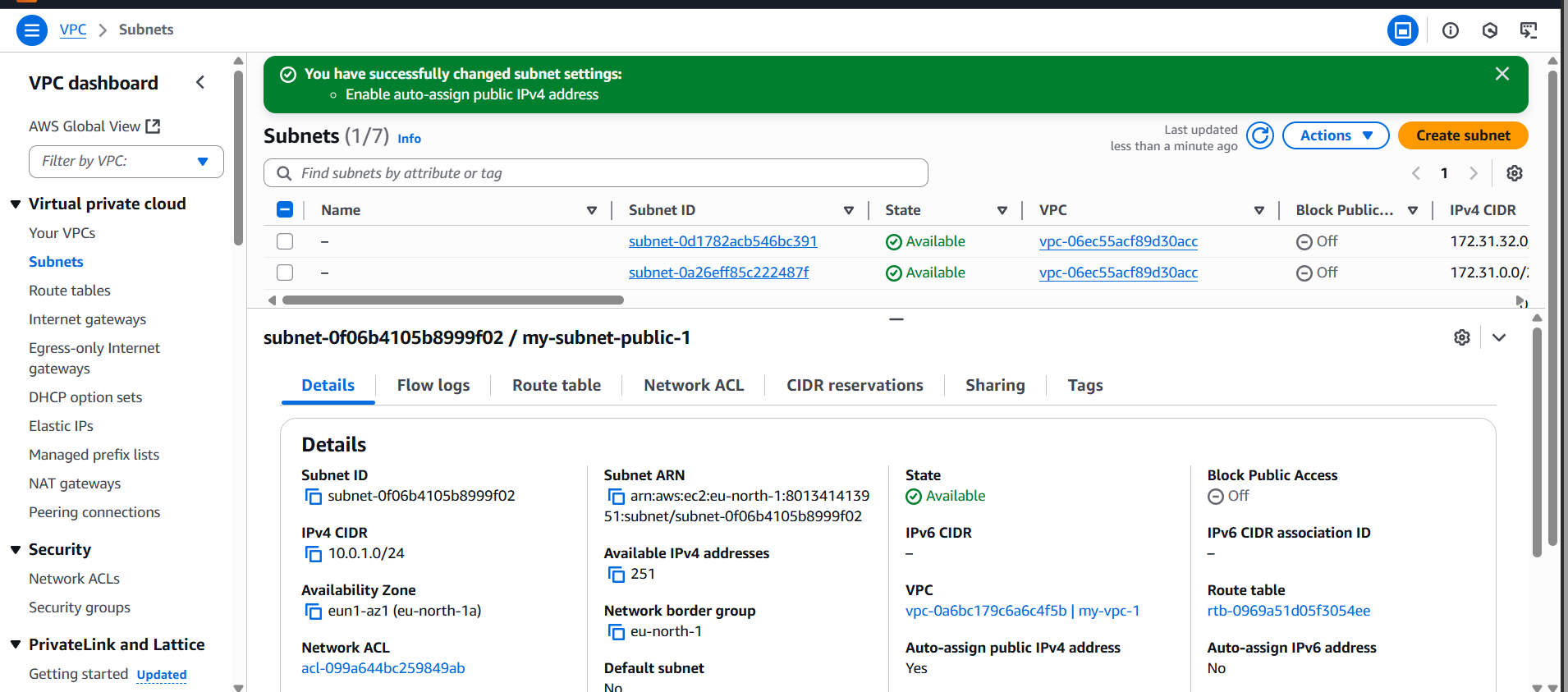
1. From the list, select your **first public subnet** (e.g., Public-Subnet-AZ1).
2. At the bottom panel, go to the **Subnet details** tab.



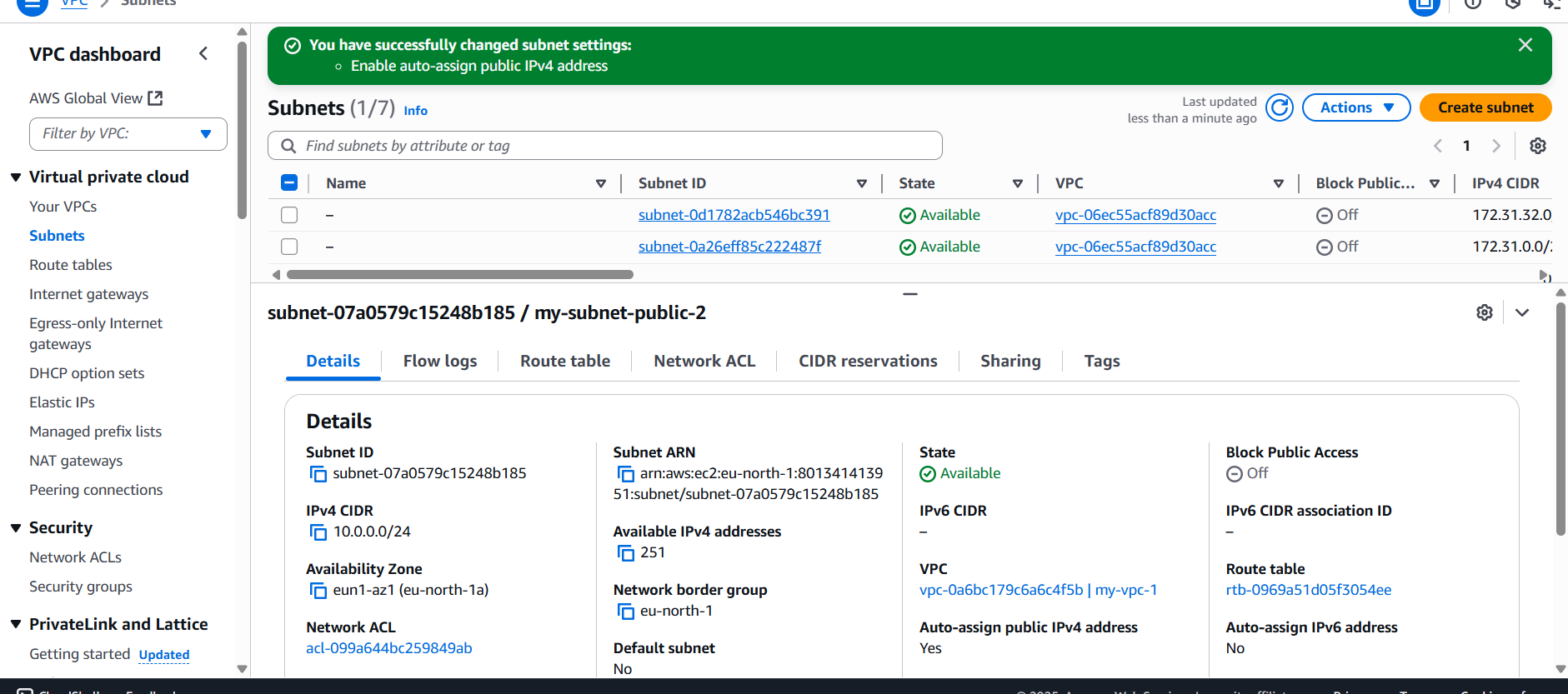
**Step 3: Modify Auto-assign Public IP**

1. Click **Actions → Edit subnet settings**.
2. Look for **Auto-assign IP settings**.
3. Enable: ✅ **Auto-assign IPv4**.
4. Save changes.





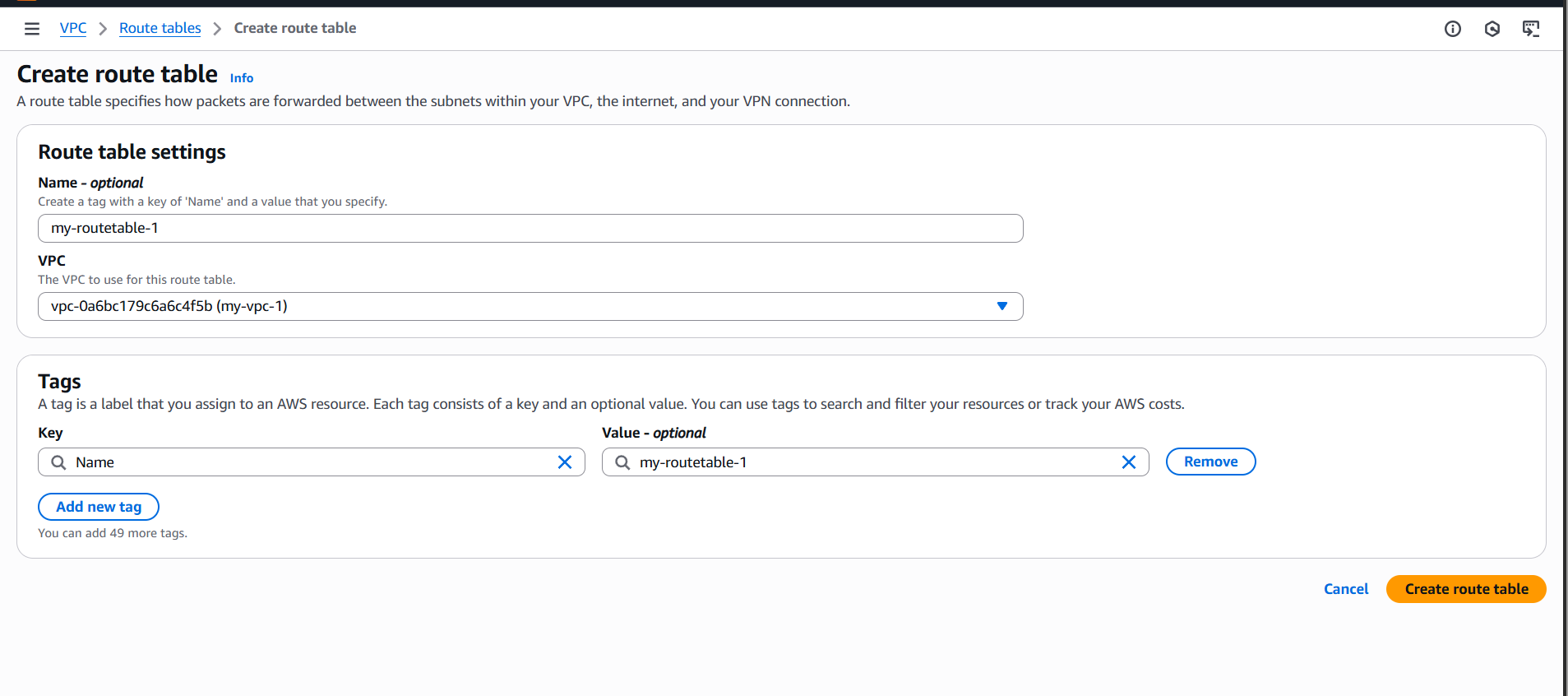
Same as 2 my subnet-public-2 done changed auto-assgin ipv4 address



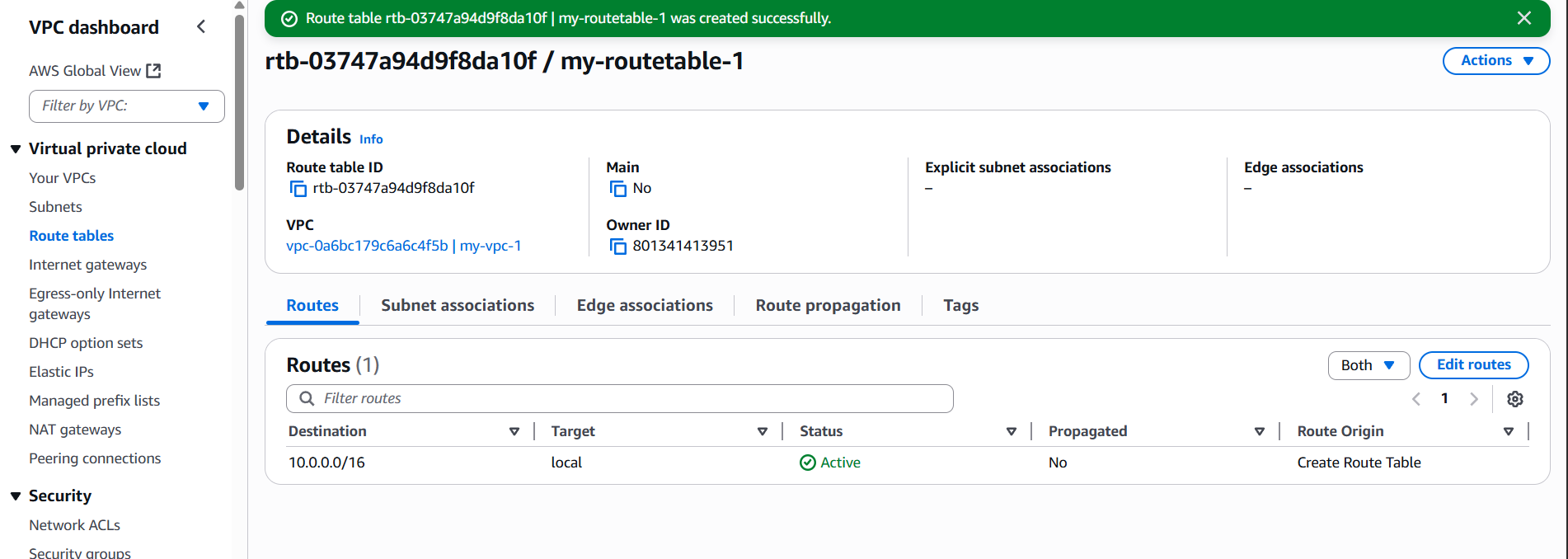
1. Add 2 private subnets in the private route table.

**Why Associate Private Subnets with Private Route Table?**

* **Route table = set of rules that decide how traffic flows in/out of subnets.**
* **By default, all subnets in a VPC use the main route table. That’s not best practice.**
* **We create a separate private route table for private subnets so:**
  + **They don’t get direct internet access.**

****

**First,** create one route table

****

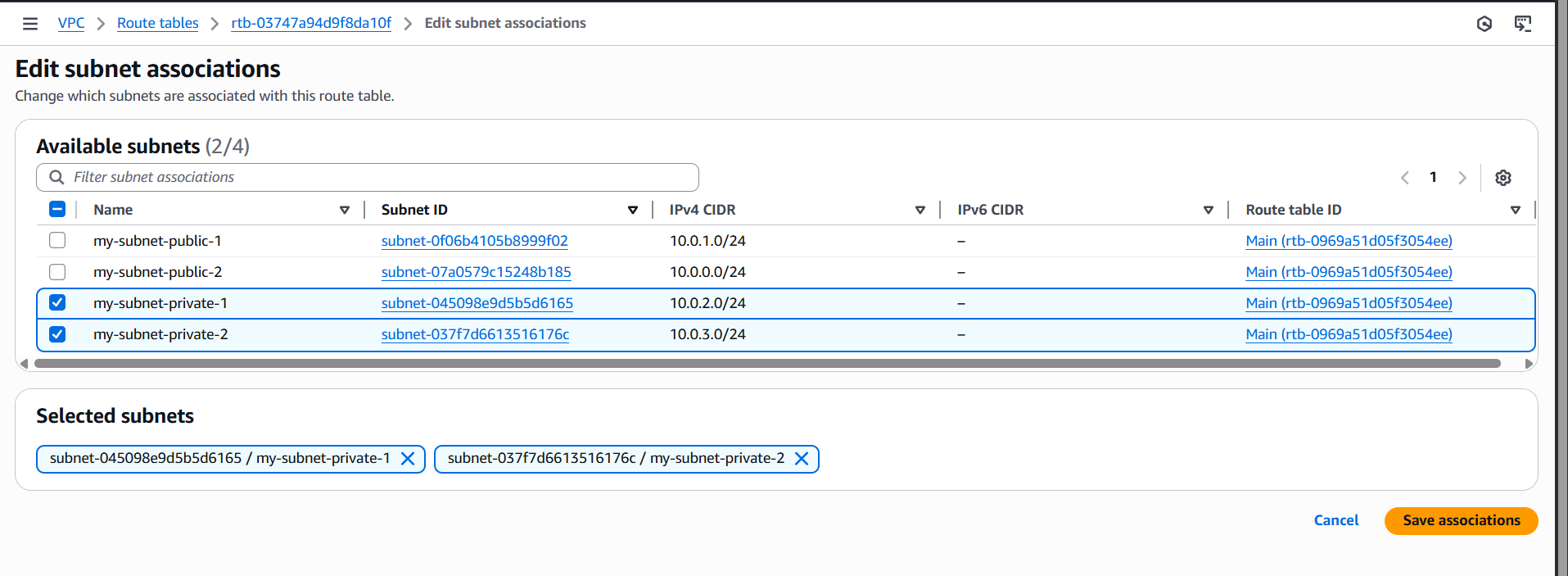
**Edit Subnet Associations**

**With Private-RT selected, go to the Subnet associations tab (bottom panel).**

**Click Edit subnet associations.**

**Select your 2 private subnets (e.g., Private-Subnet-AZ1, Private-Subnet-AZ2).**

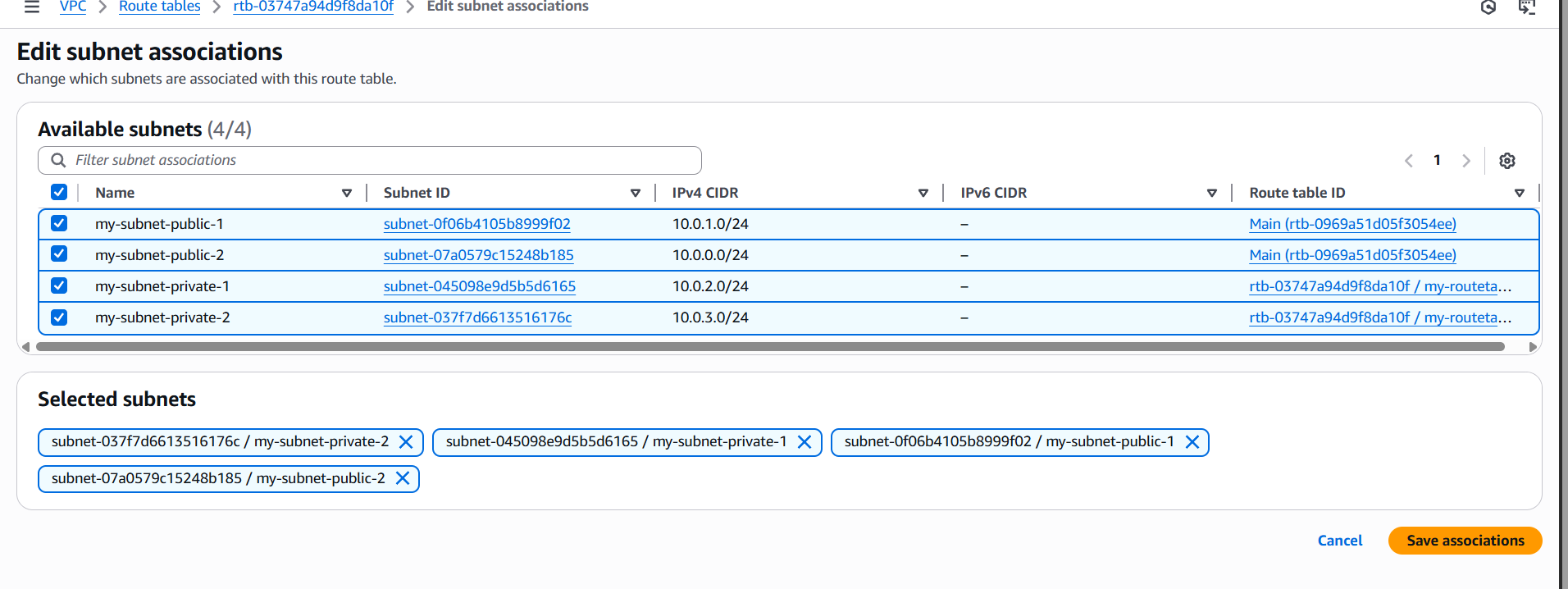
**Save associations.**

****

1. Add 2 public subnets in public route table.

**Associate Public Subnets**

1. With the Public Route Table still selected → go to **Subnet associations** tab.
2. Click **Edit subnet associations**.
3. Tick your **2 public subnets** (e.g., Public-Subnet-AZ1, Public-Subnet-AZ2).
4. Click **Save associations**.



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

A route to **Local (VPC internal traffic)** → automatically added by AWS.

A route to the **Internet Gateway (IGW)** → you must add manually.

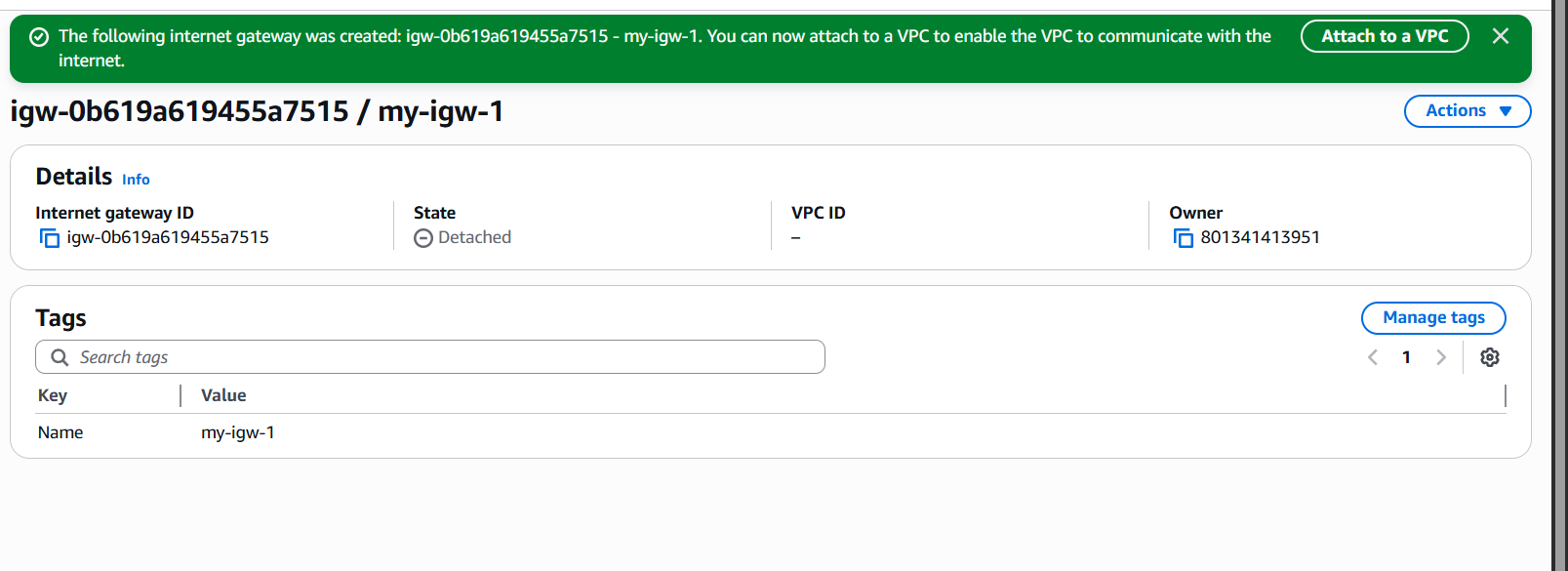
**Check Existing Routes**

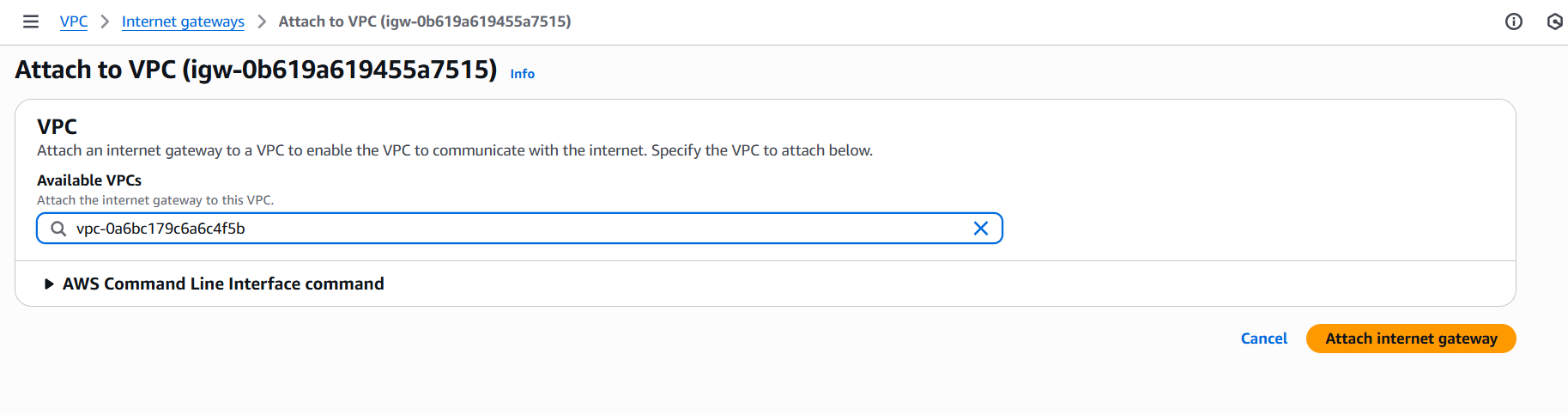
1. At the bottom panel, click the **Routes** tab.
2. You’ll see one default route:
   * **Destination:** 10.0.0.0/16 (your VPC CIDR)
   * **Target:** local  
     👉 This is automatically there for internal communication.

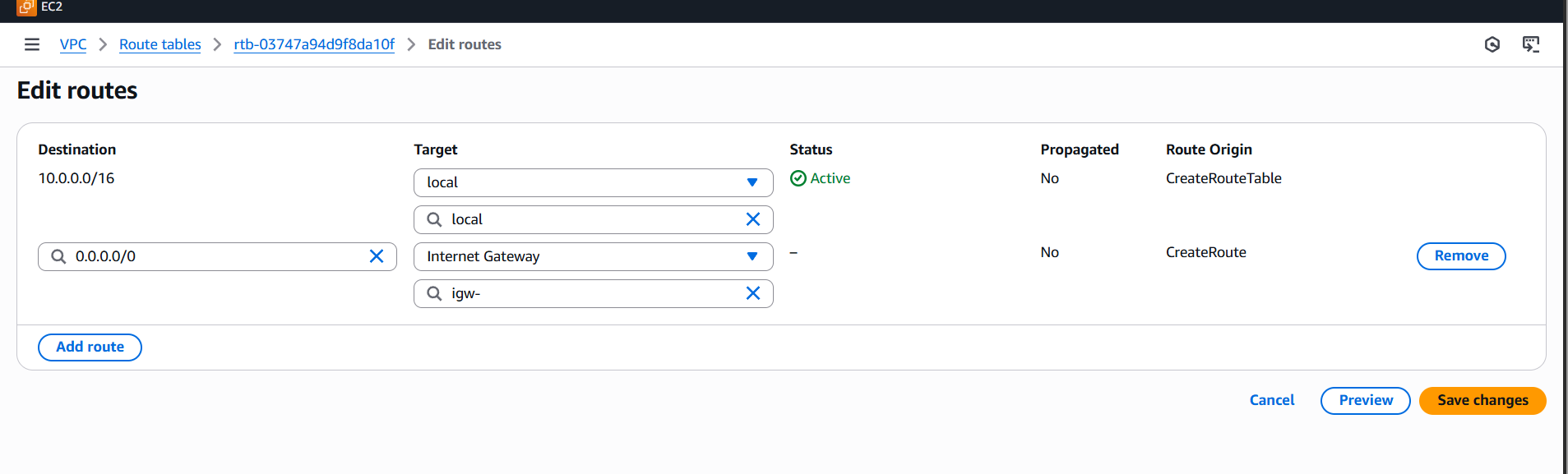
**Add Internet Route**

1. In the same **Routes tab**, click **Edit routes**.
2. Click **Add route**.
3. Enter:
   * **Destination:** 0.0.0.0/0
   * **Target:** Select your **Internet Gateway (IGW)** (e.g., igw-xxxxxx).
4. Click **Save changes**.

Before you open an igw







**Verify**

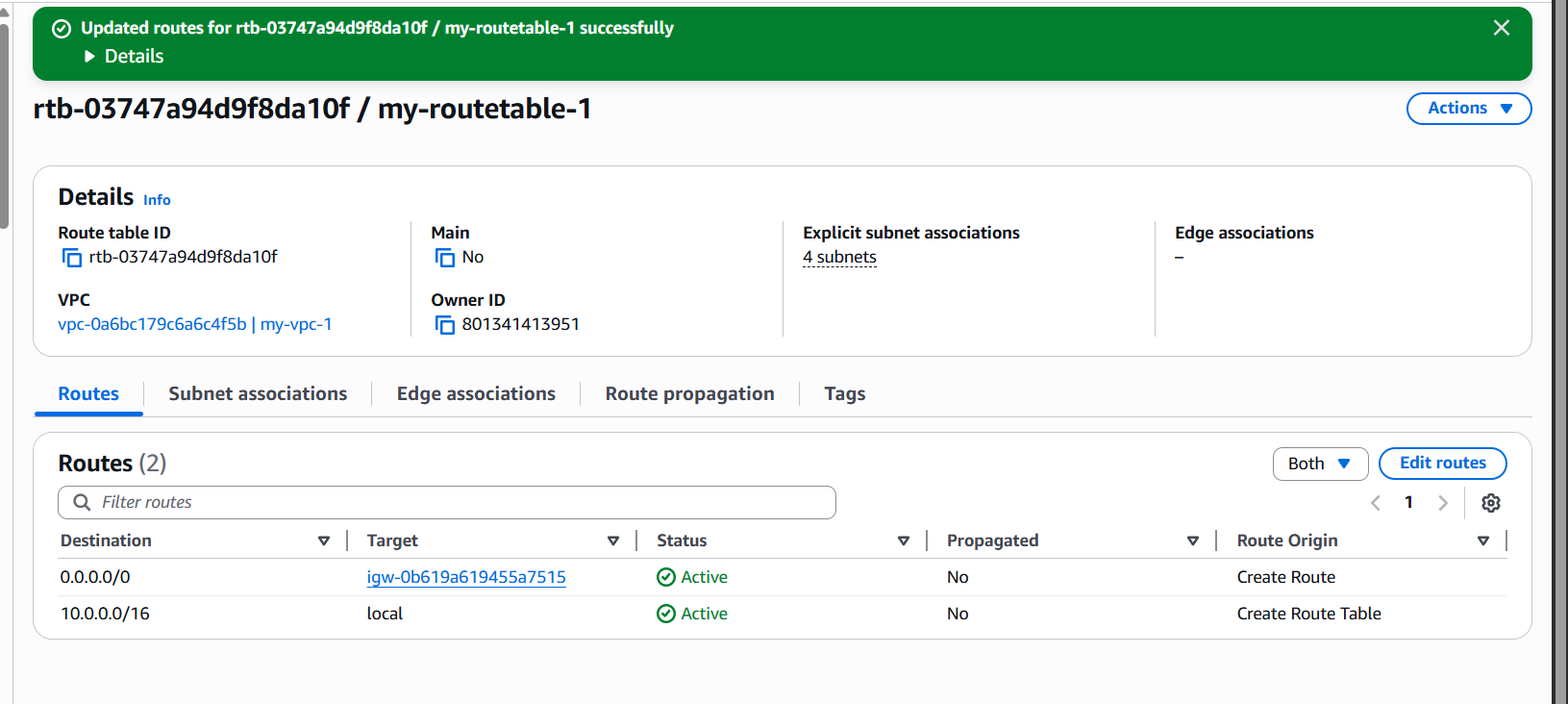
Now the Public Route Table should have two routes:

* + 10.0.0.0/16 → local
  + 0.0.0.0/0 → igw-xxxxxx

This means:

* + All internal traffic stays inside VPC.
  + Any external (internet) traffic goes to the **Internet Gateway**.

1. Done! Now your **public subnets** associated with this Public Route Table can reach the internet (if Auto-assign Public IP is enabled).

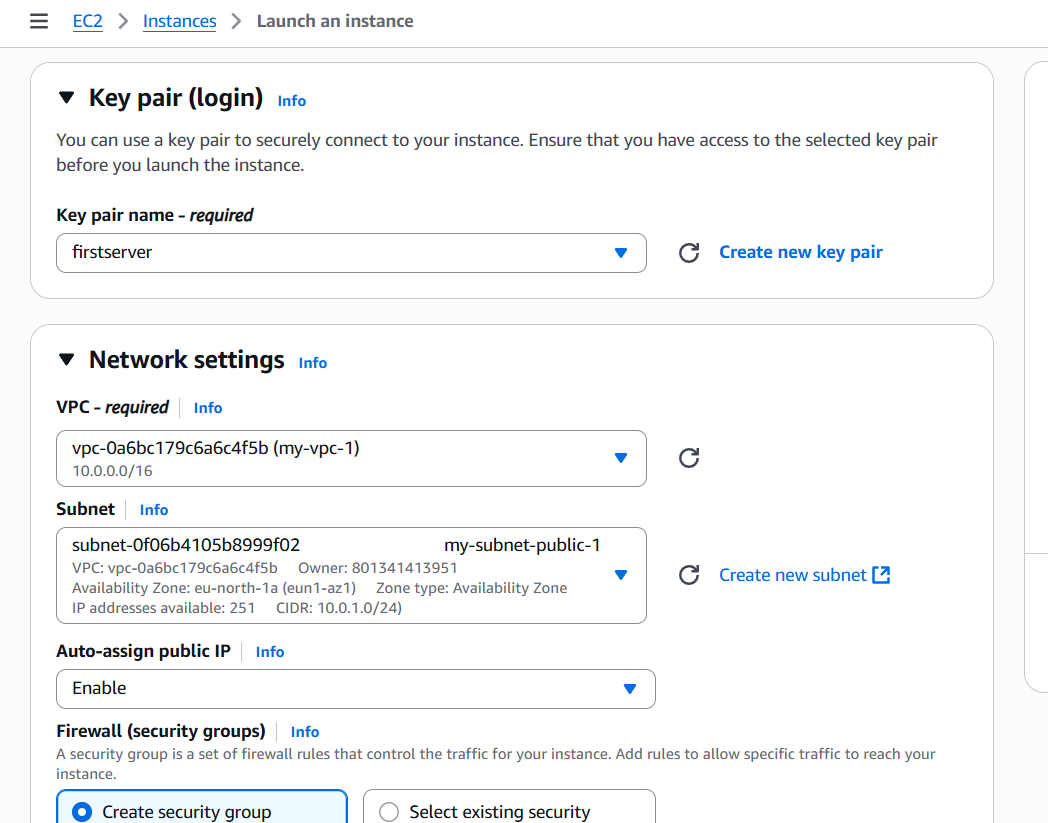
****

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

Create one ec2 instance while creating a choice as our wish, then go to network configuration

**Network Settings**:

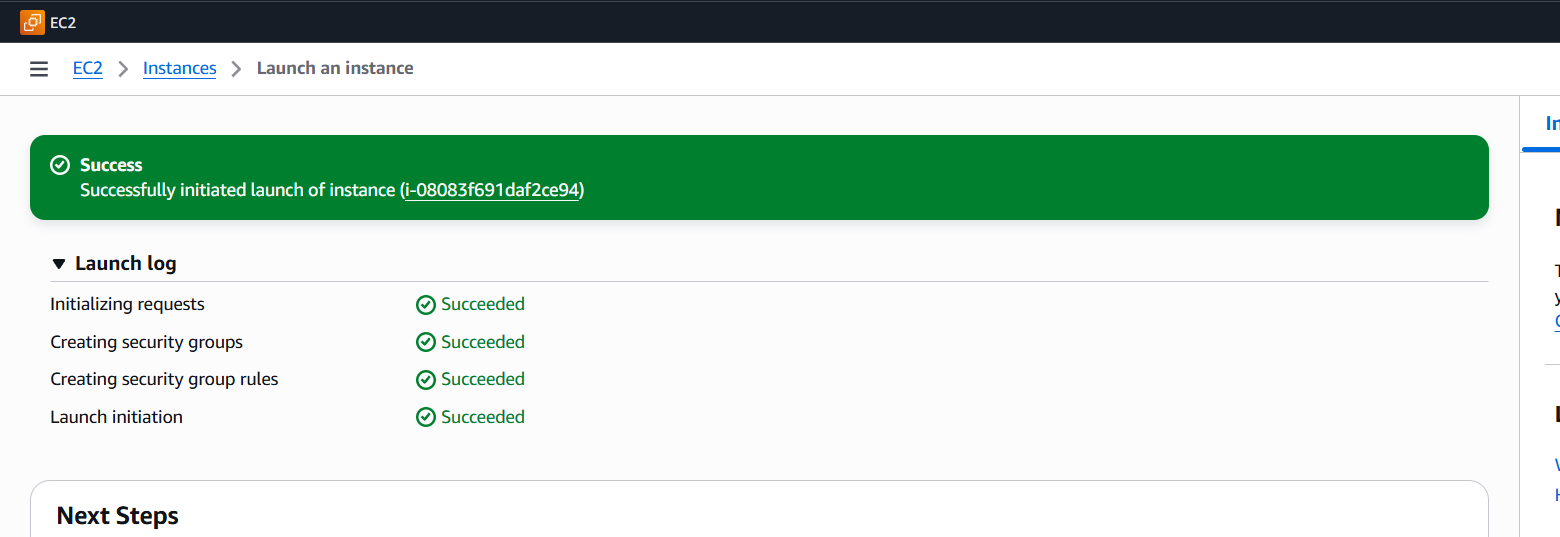
* **VPC:** Select your VPC (My-VPC).
* **Subnet:** Select one of your **Public Subnets** (e.g., Public-Subnet-AZ1).
* **Auto-assign Public IP:** Must be **Enabled** (so you get a public IP)

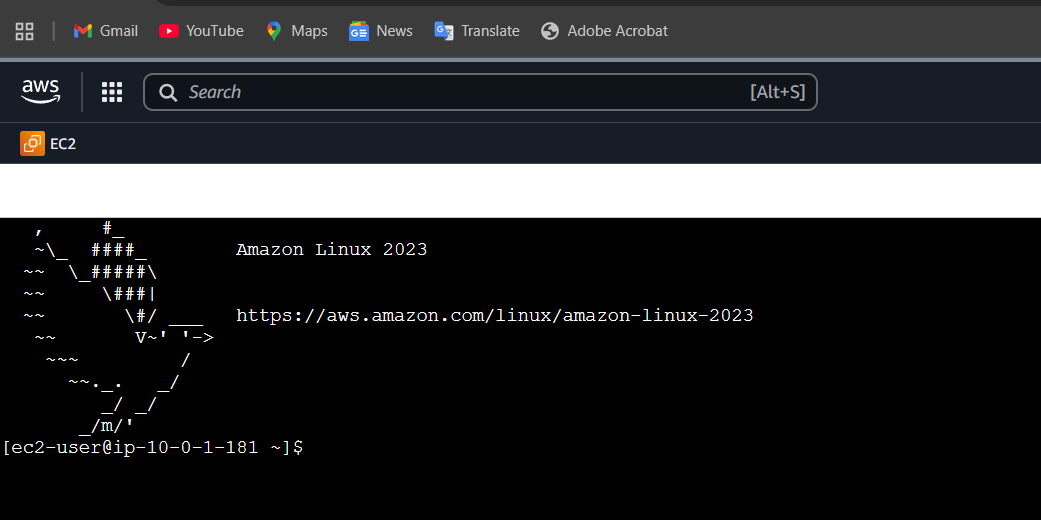


Review settings → Click **Launch Instance**.

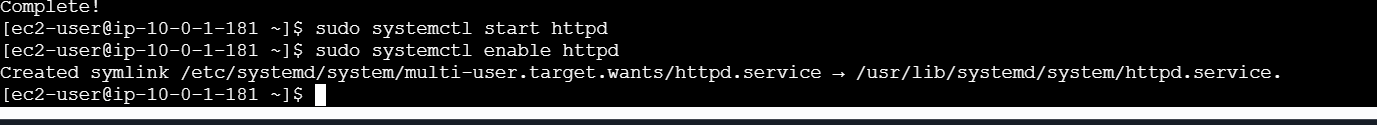
Wait until **Instance State = Running**.

Note the **Public IPv4 address or Public DNS**.









sudo amazon-linux-extras enable php8.0

sudo yum install -y php php-cli php-mysqlnd => use this command for install for PHP

echo "<?php phpinfo(); ?>" | sudo tee /var/www/html/index.php=> execute this command

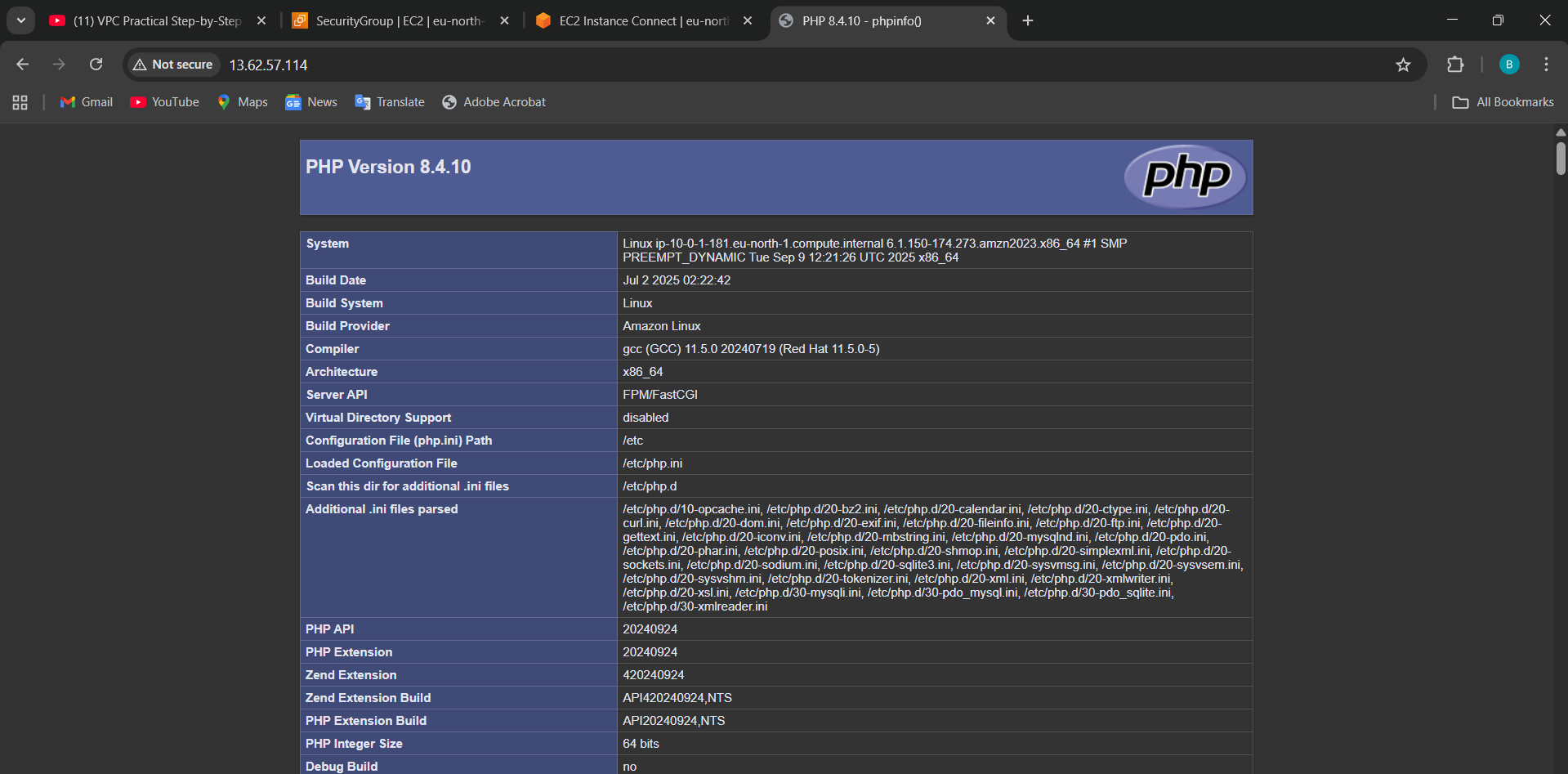
 Go to **EC2 → Instances → Select your instance → Security tab**.

 Click on the Security Group.

 In **Inbound rules**, check:

* ✅ **SSH** (Port 22, My IP).
* ✅ **HTTP** (Port 80, 0.0.0.0/0 and ::/0).

 If missing → **Edit inbound rules → Add rule → HTTP → Source = Anywhere (0.0.0.0/0)**.



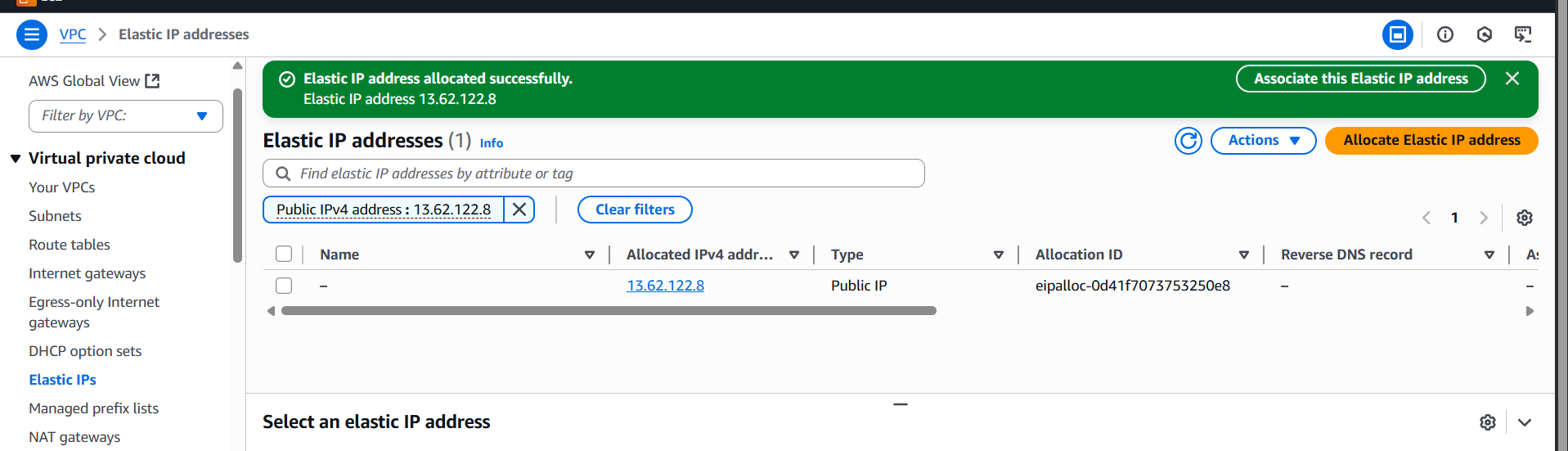
**What is PHP?**

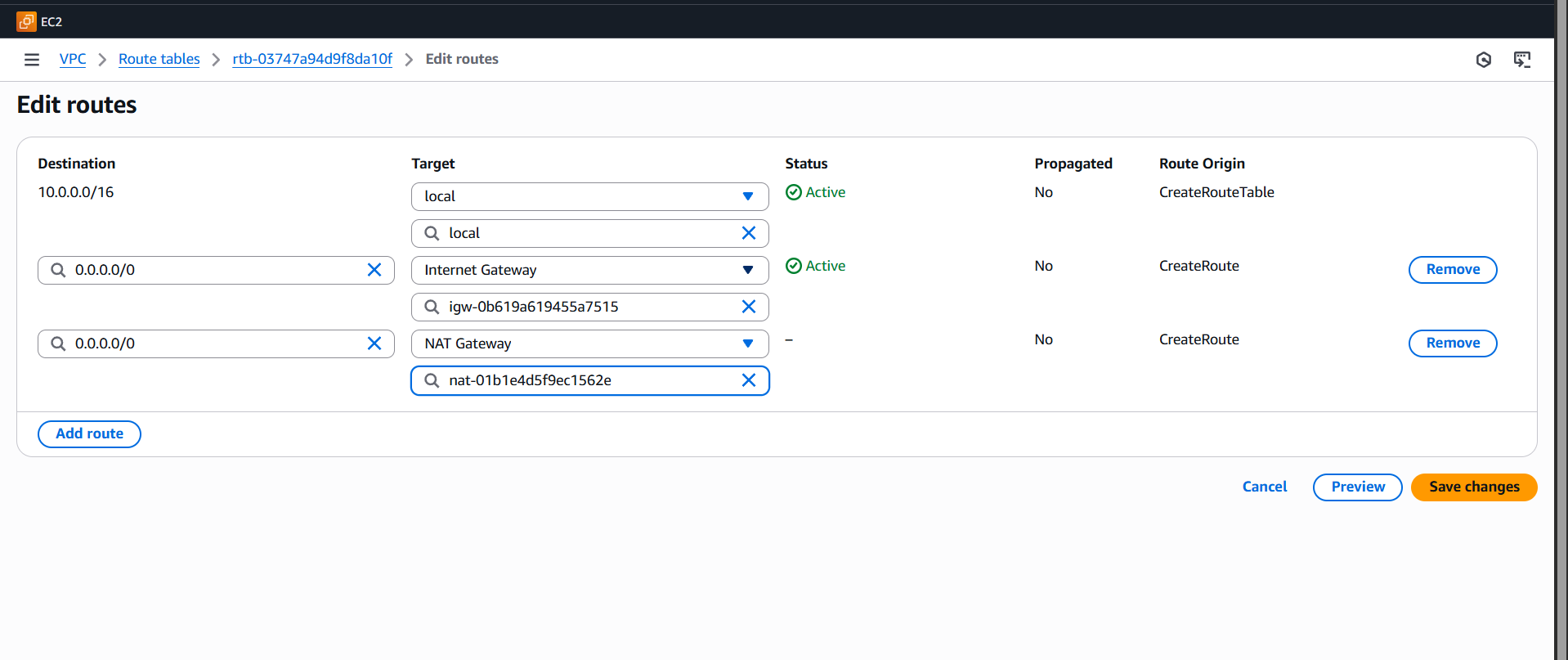
* **PHP** = Hypertext Preprocessor (a **server-side scripting language**).
* It runs on the **server** and generates **dynamic content** for web pages.
* Unlike HTML (static content), PHP can **interact with databases, process forms, and display content based on user actions**.

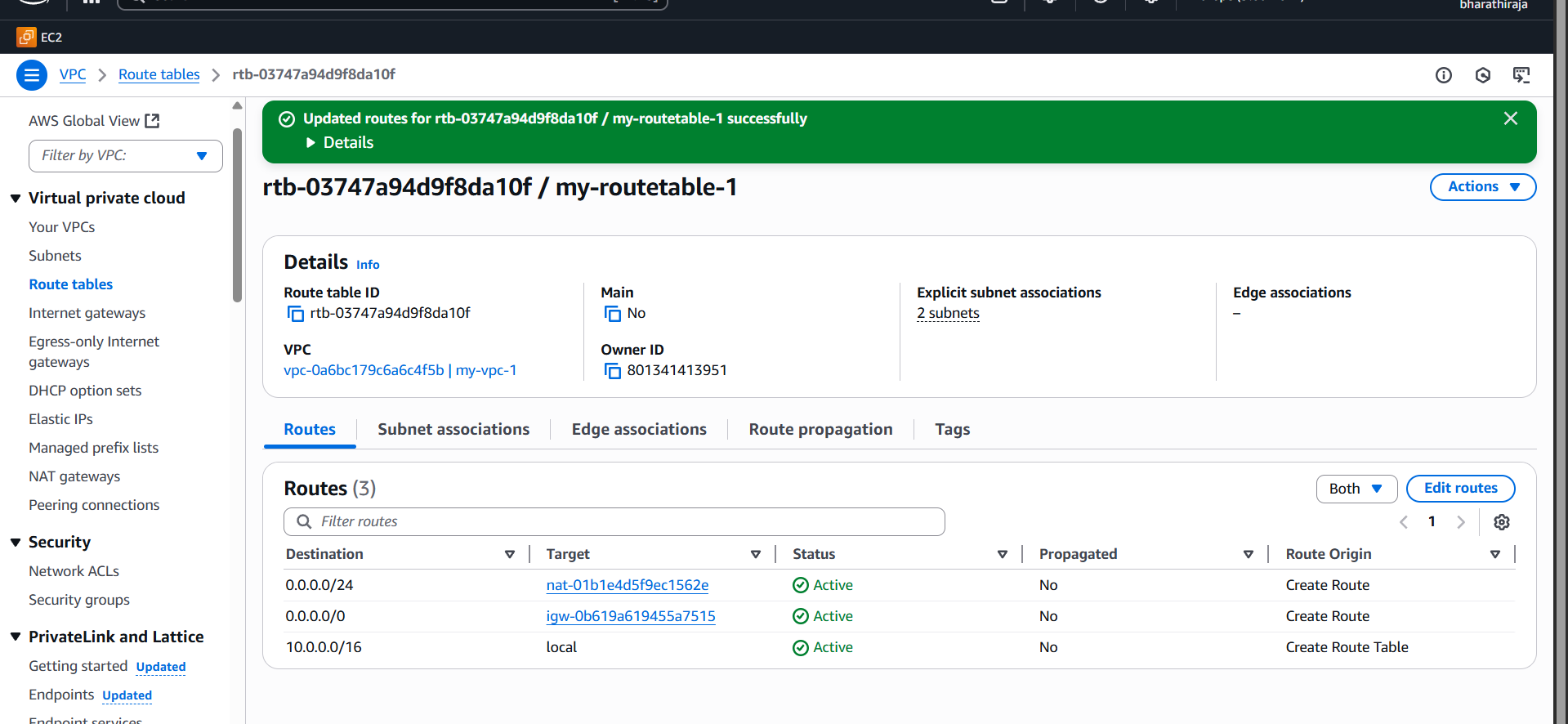
1. Configure NAT gateway in public subnet and connect to private instance.

**Why Use a NAT Gateway?**

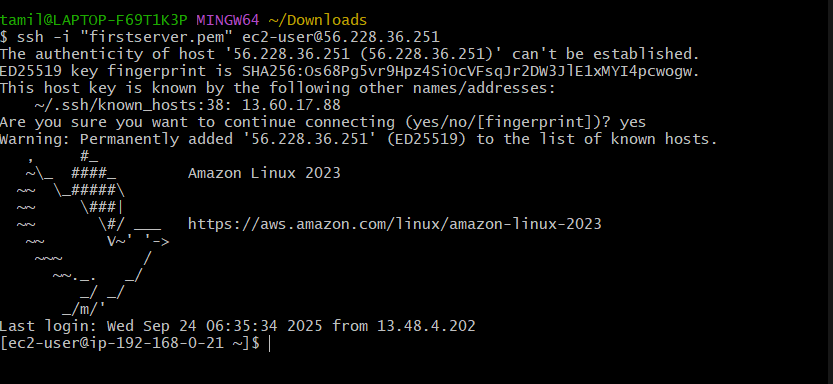
* Private subnets **cannot access the internet directly**.
* NAT Gateway in a public subnet allows **outbound internet traffic** for private instances.







1. Install Apache Tomcat in a private EC2 and deploy a sample app.  
   first step: connect the instance



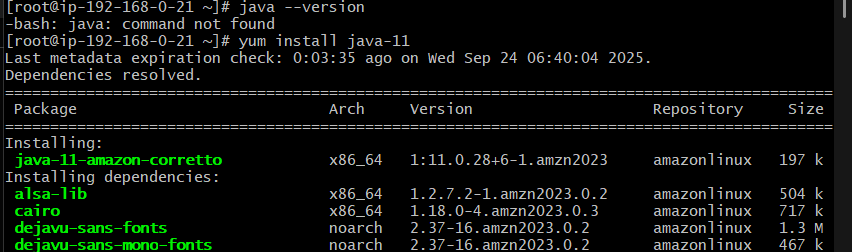
sudo yum update -y

sudo amazon-linux-extras enable java-openjdk11

sudo yum install java-11-openjdk -y

java -version  
use these commands to install in our applicaton

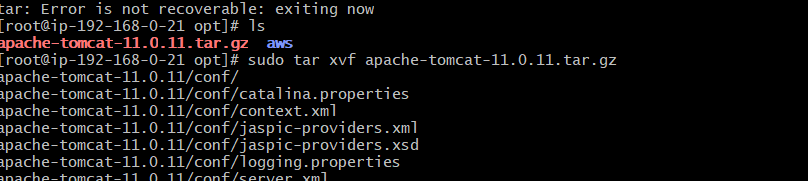
sudo yum install java-11 => use this command install java



Wget (link )=> download and install

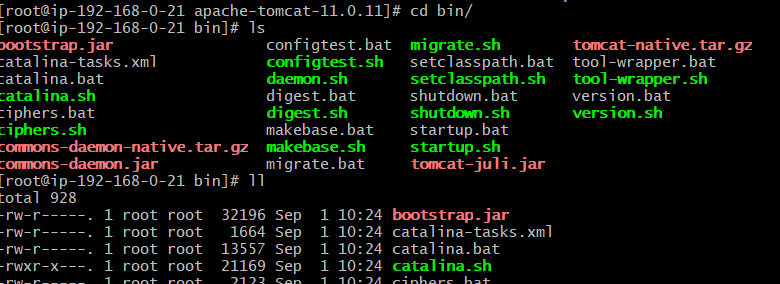


sudo tar xvf apache-tomcat-10.1.26.tar.gz =>use extract the file



cd /opt/tomcat/bin=> go to the file location

you may see the result below nect page if you have reached the exact location

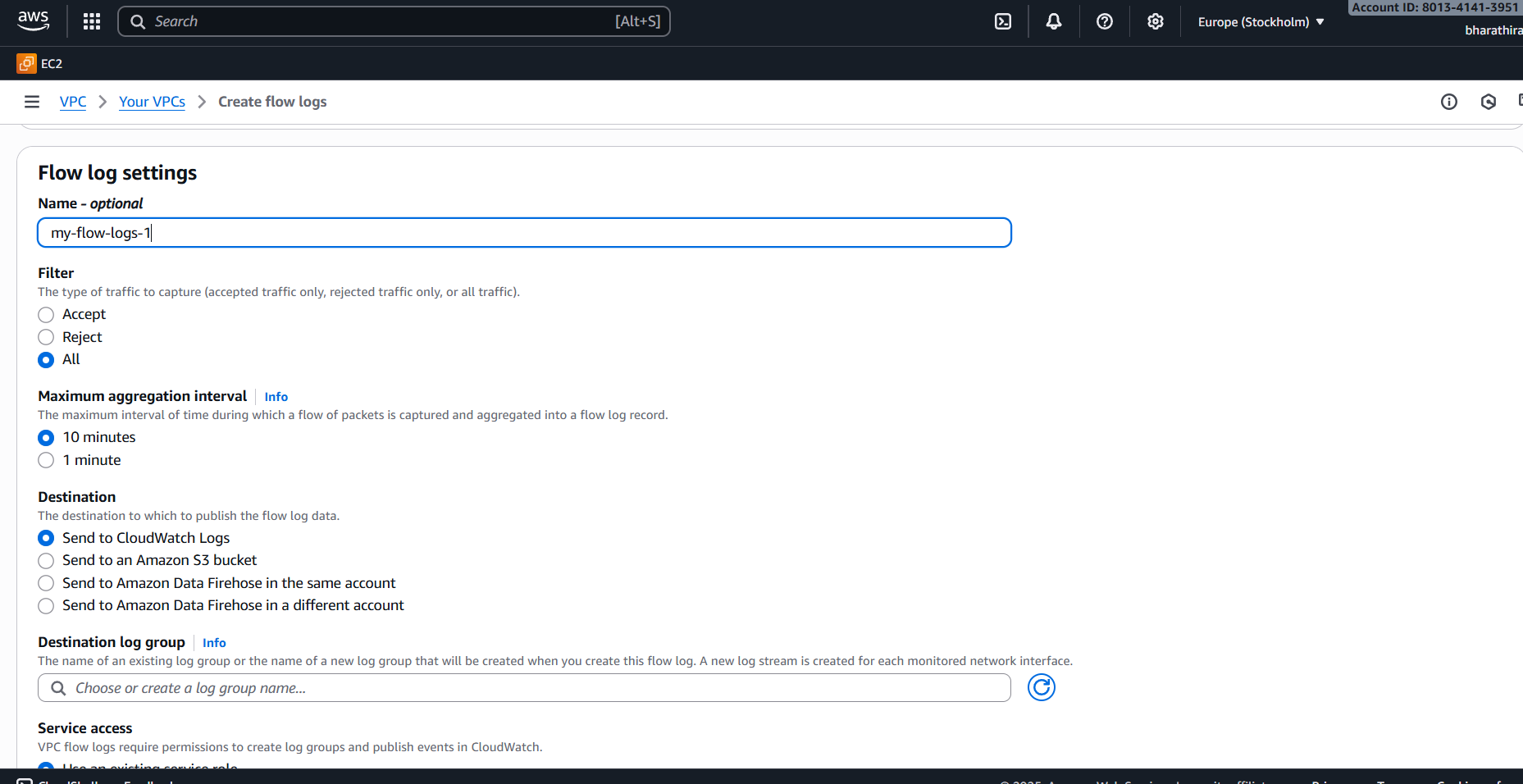


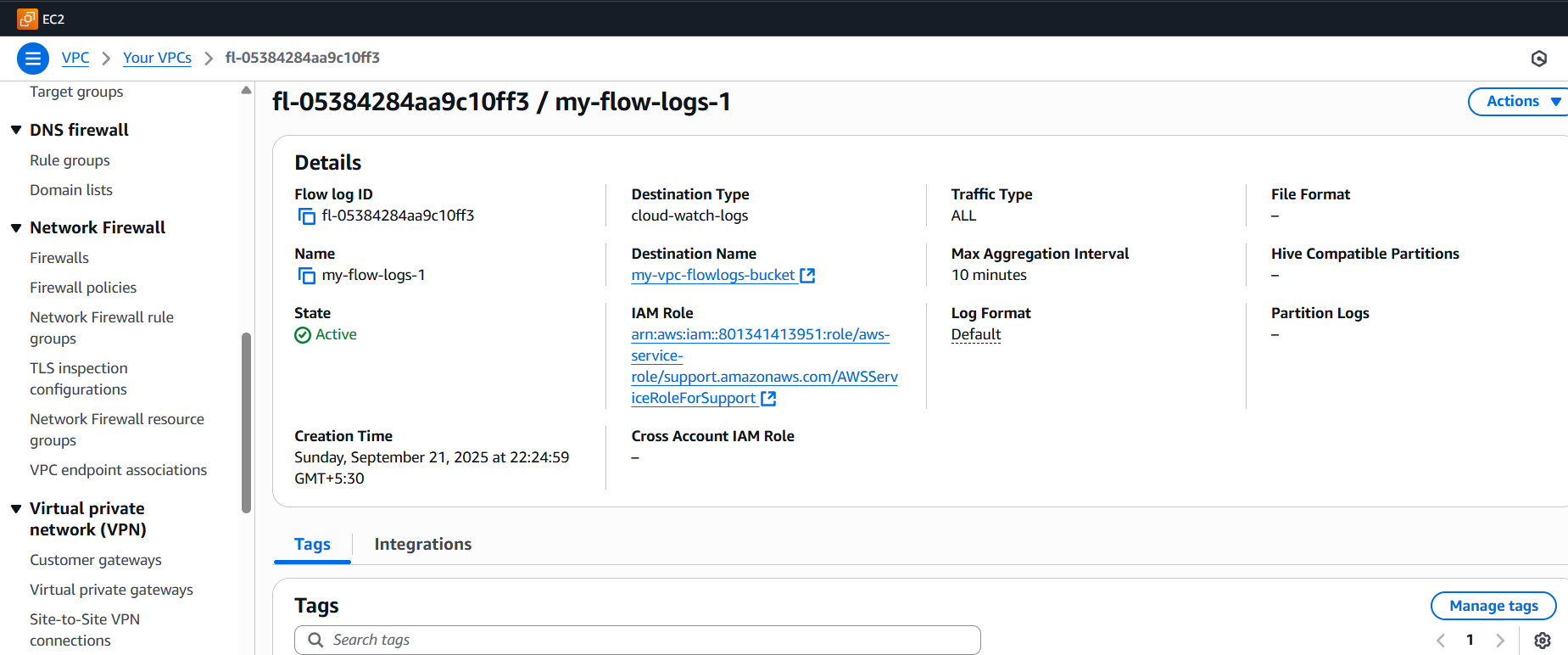
1. Configure VPC flow logs and store the logs in S3 and CloudWatch.



**Create Flow Log**

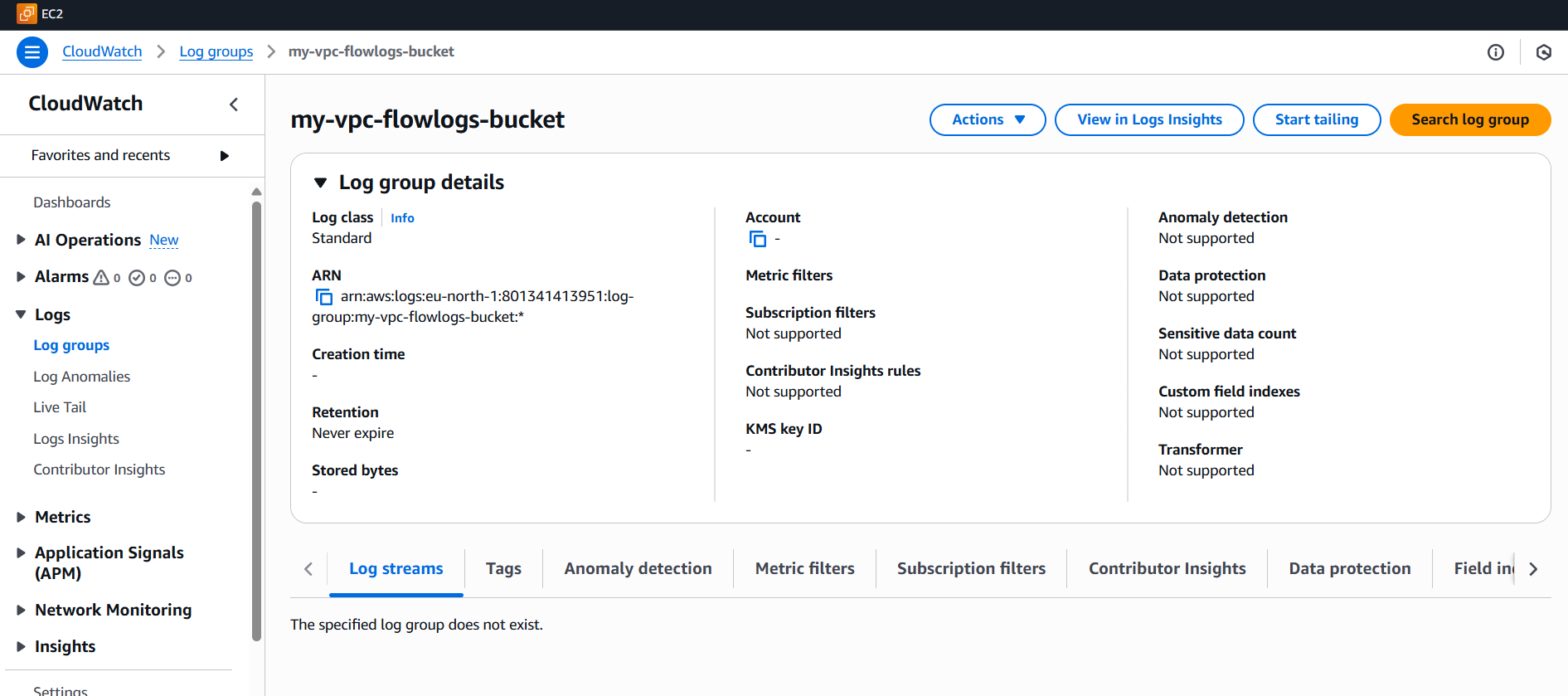
1. With your VPC selected, click **Actions → Create flow log**.
2. Configure:
   * **Filter:** Choose what traffic to log
     + ALL (both accepted/rejected)
     + ACCEPT (only accepted traffic)
     + REJECT (only rejected traffic)
   * **Maximum aggregation interval:** 1 minute (for detailed logs) or 10 minutes
   * **Destination log group:** Choose **Send to CloudWatch Logs**.

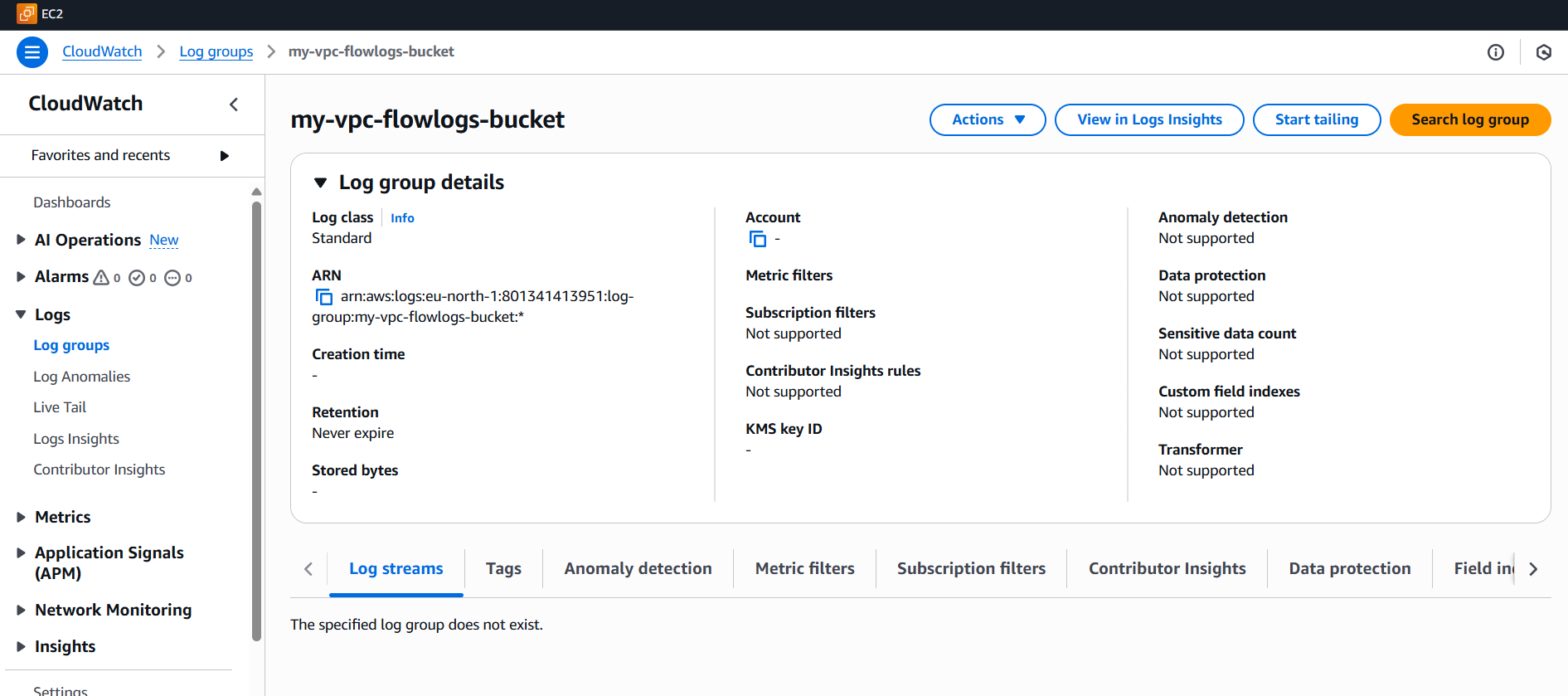




**Send Logs to S3 Bucket**

1. **Destination log group:** Choose **Send to Amazon S3 bucket**.





**How to Verify Logs**

**For CloudWatch Logs:**

1. Go to **CloudWatch → Logs → Log Groups**.
2. Open your log group (VPCFlowLogs).
3. Expand → you’ll see log streams with records like: