Placement Empowerment Program

Cloud Computing and DevOps Centre

Set Up a Private Network in the Cloud : Create a Virtual Private Cloud (VPC) with subnets for your instances. Configure routing for internal communication between subnets.

Name: SAMRAJ K Department: CSE

St. JOSEPH'S COLLEGE OF ENGINEERING | St. JOSEPH'S INSTITUTE OF TECHNOLOGY
AUTONOMOUS INSTITUTIONS, AFFILIATED TO ANNA UNIVERSITY

Introduction

The goal of this Proof of Concept (PoC) was to set up a **Private Network in the Cloud** by creating a **Virtual Private Cloud (VPC)** in AWS, configuring **subnets**, and ensuring **internal communication** between instances within the VPC. This setup focused on isolating cloud resources in a private network, providing a secure environment for communication, and making sure that only internal traffic is allowed, without exposing resources to the public internet.

In this PoC, we created a **private subnet** where EC2 instances could communicate with each other without direct exposure to external networks.

Overview

In this PoC, we:

1. **Created a VPC** in AWS, which serves as the isolated private network.

2. **Created a private subnet** inside the VPC where EC2 instances can reside, ensuring no direct access from the public internet.

3. **Set up routing** to allow communication between the instances within the same VPC and subnet.

4. Launched **EC2 instances** in the private subnet and verified their ability to communicate internally using their private IP addresses.

The setup is designed to simulate a secure cloud environment where resources can interact securely without being exposed to external traffic.

Objective

The primary objectives of this PoC were:

1. Establish a Private Network: Set up a private VPC and subnets for cloud resources to reside in, ensuring they are isolated from the public internet.

2. Internal Communication: Ensure that EC2 instances within the private subnet can communicate with each other using their private IPs.

3. Security: Maintain internal communication only within the VPC, preventing direct exposure of instances to the public internet.

4. Simplify Management: Organize cloud resources into subnets for easier management and scaling, with clear routing between them.

Importance

1. Security: By placing EC2 instances in a private subnet and ensuring that no public IP is assigned, the resources are isolated from external traffic. This is crucial for keeping sensitive data and services protected.

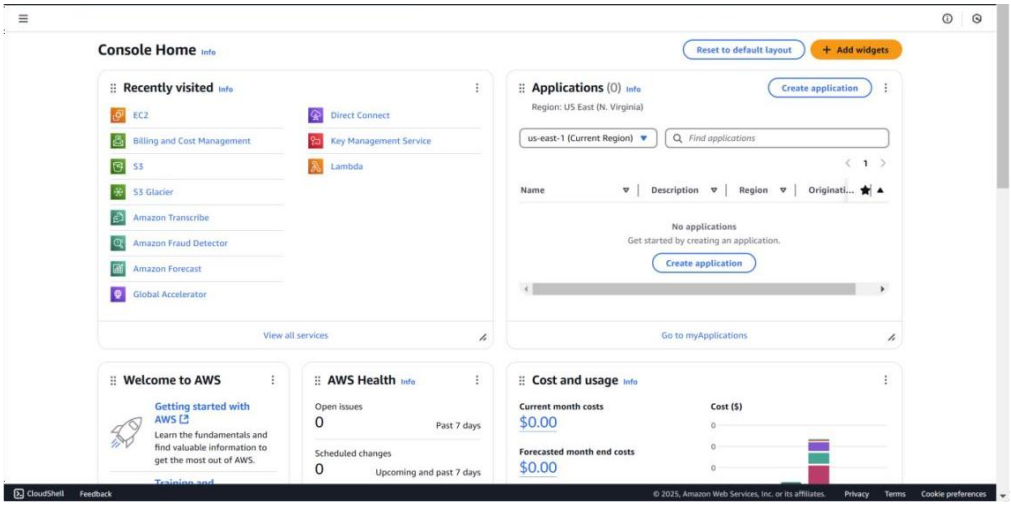
2. Cost Efficiency: Using internal communication and private subnets can help reduce costs related to public internet access and data transfer.

3. Flexibility: This setup provides a foundation for building more complex cloud infrastructures, such as multi-tier applications where only backend servers (databases, app servers) are private, while frontend servers may be public.

Step-by-Step Overview

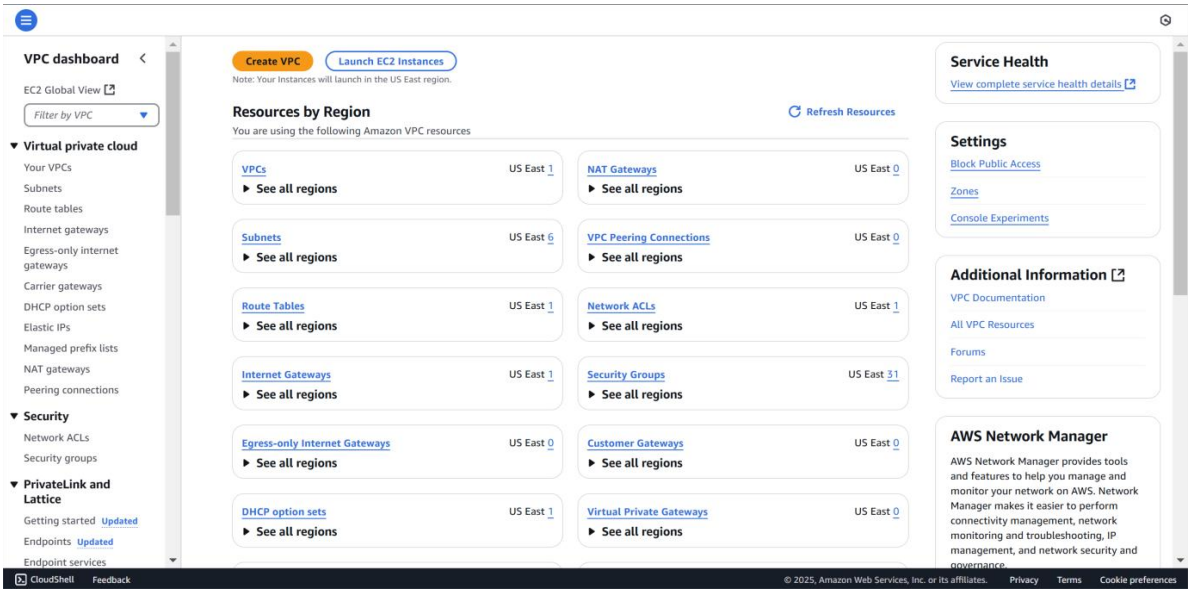
Step 1:

1. Go to AWS Management Console.
2. Enter your username and password to log in.



Step 2:

In the **VPC Dashboard**, click the **Create VPC** button.



Step 3:

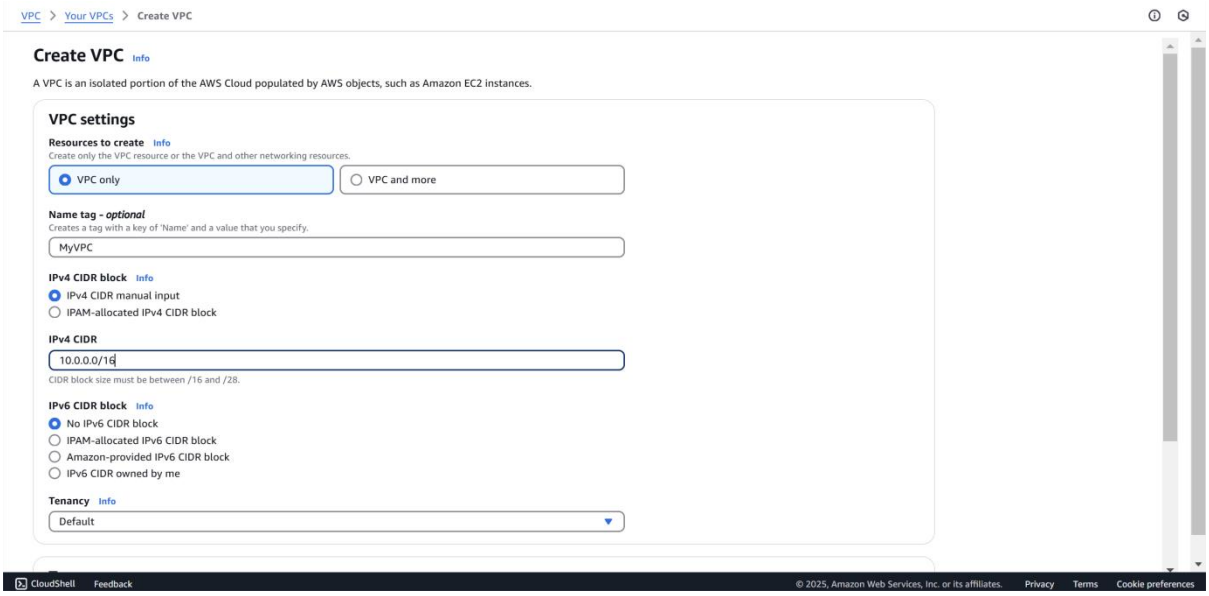
In the VPC creation wizard, select **VPC only**.

Name tag: Enter MyVPC .

IPv4 CIDR block: Enter 10.0.0.0/16 (this defines the IP range for your VPC).

Tenancy: Leave it as **Default**.

Click **Create VPC**.



Step 4:

In the **VPC Dashboard**, click on **Subnets** in the left-hand menu.

Click the **Create subnet** button.

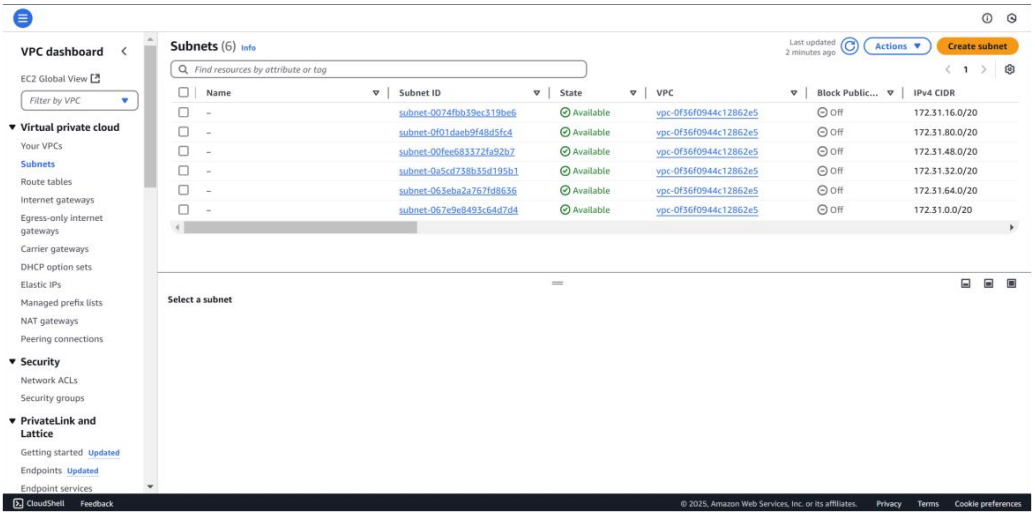
VPC: Select MyVPC (the one you just created).

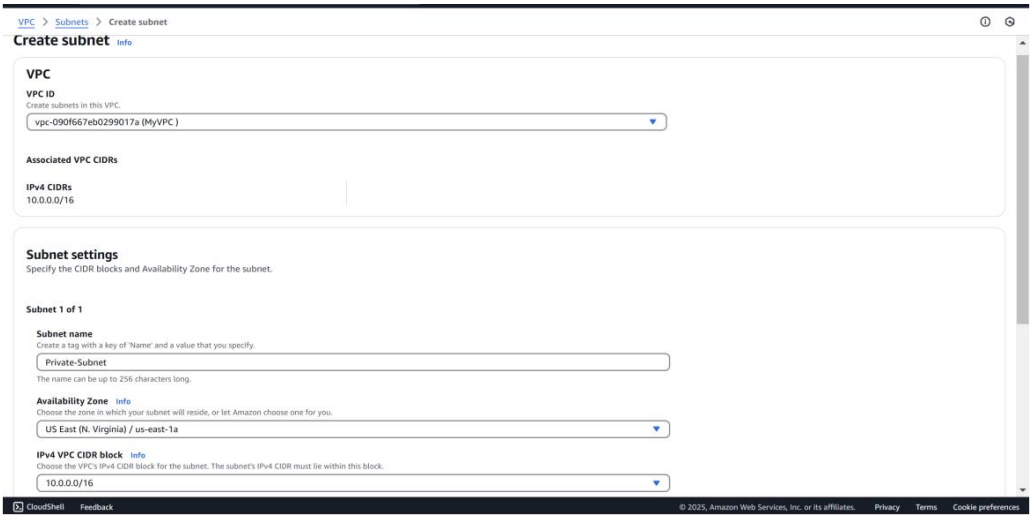
Subnet name: Enter Private-Subnet.

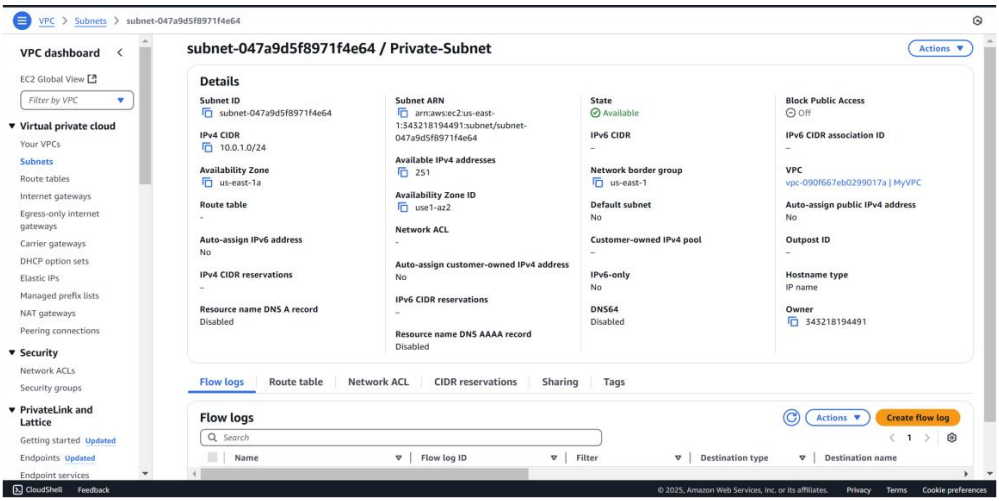
Availability Zone: Pick any (e.g., us-east-1a or any zone from your region).

IPv4 CIDR block: Enter 10.0.1.0/24 (this is a smaller range within the VPC's IP range).

Click **Create subnet**.







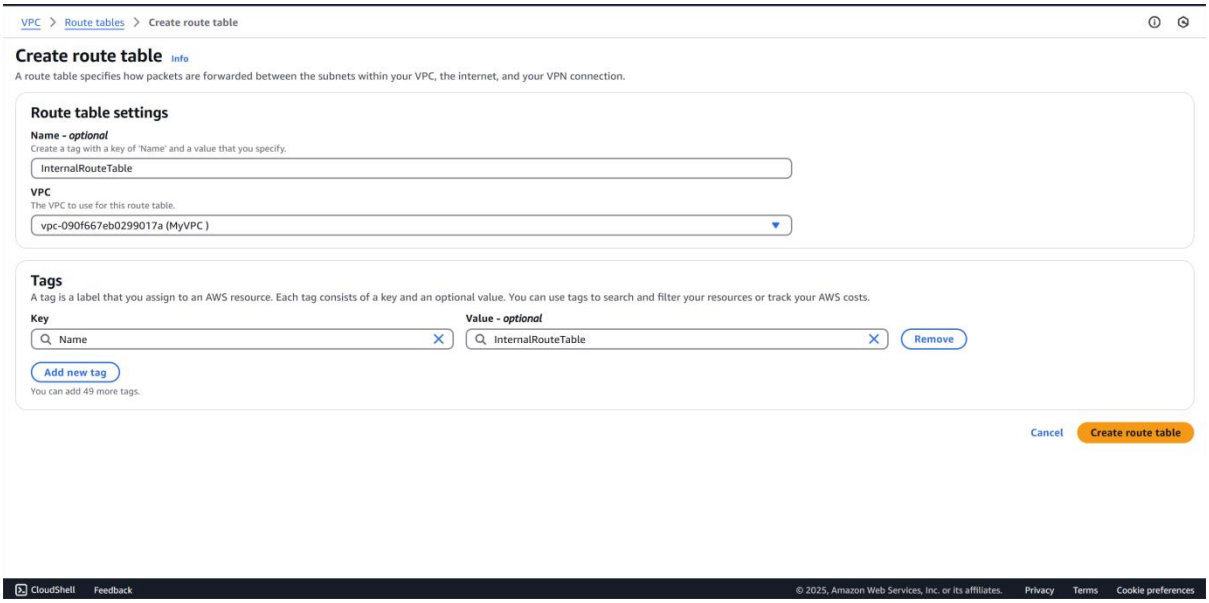
Step 5:

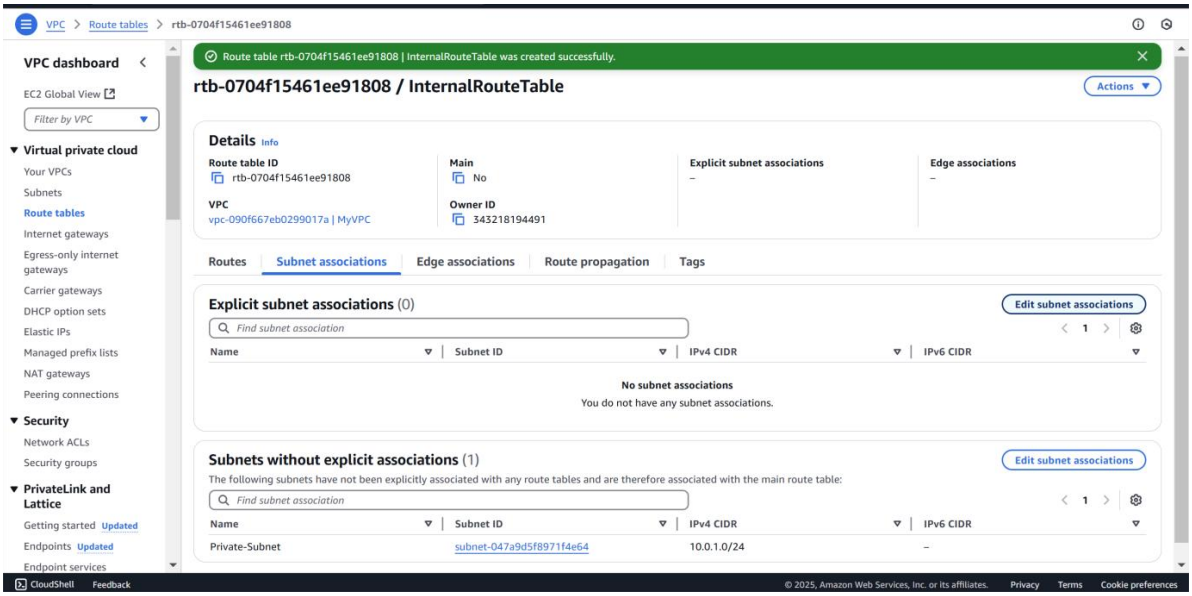
In the **VPC Dashboard**, click on **Route Tables** in the left-hand menu. Click **Create route table**.

Name tag: Enter InternalRouteTable.

VPC: Select MyVPC (the one you created earlier).

Click **Create route table**.





Step 6:

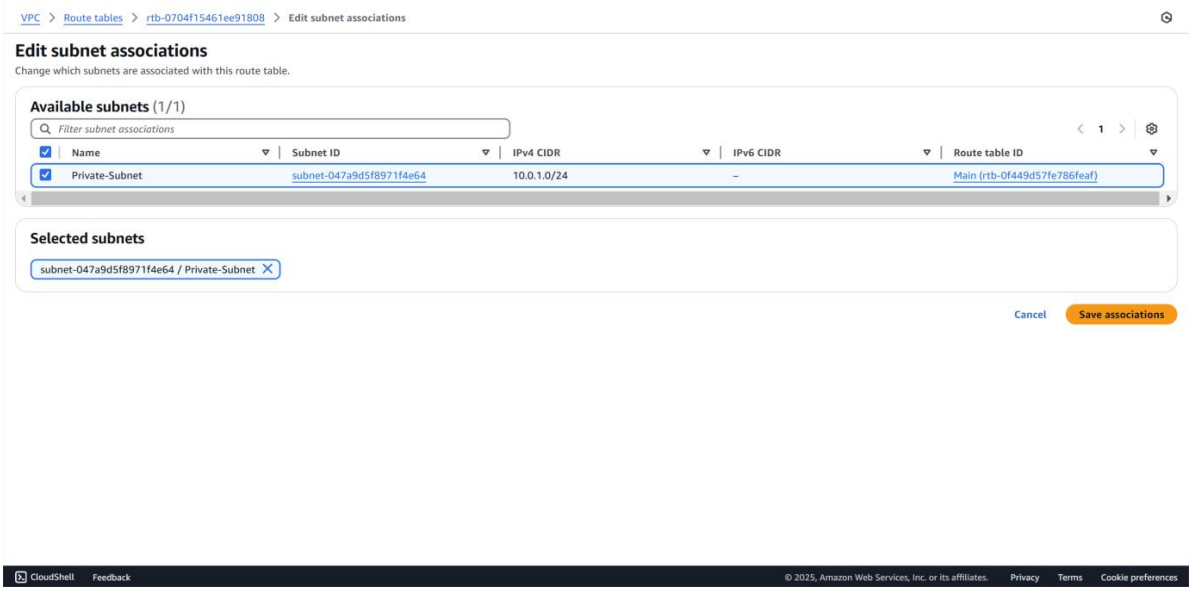
Select the InternalRouteTable you just created.

Go to the **Subnet Associations** tab (it's near the bottom).

Click **Edit subnet associations**.

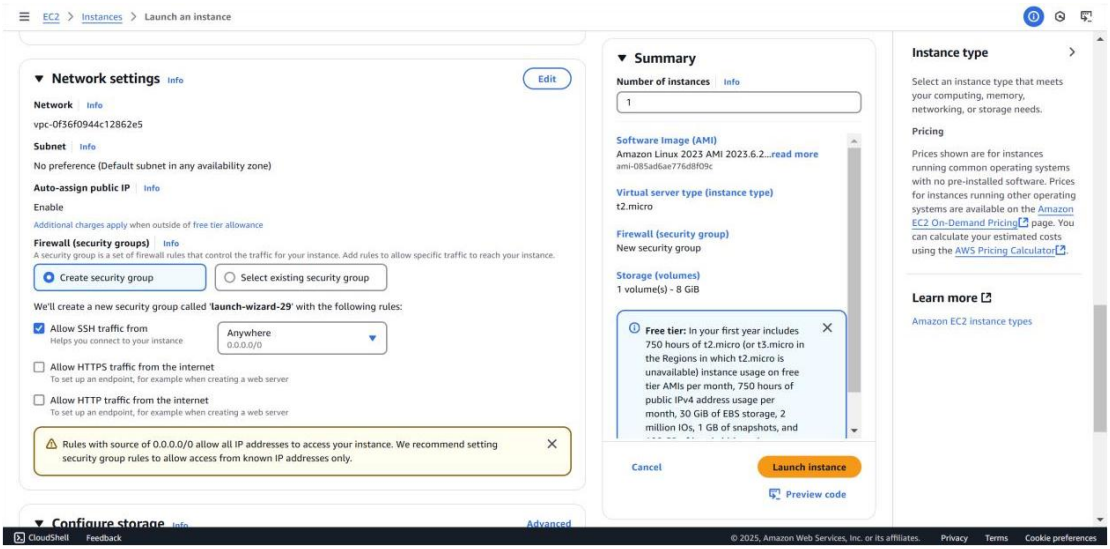
Select Private-Subnet (the subnet you created earlier).

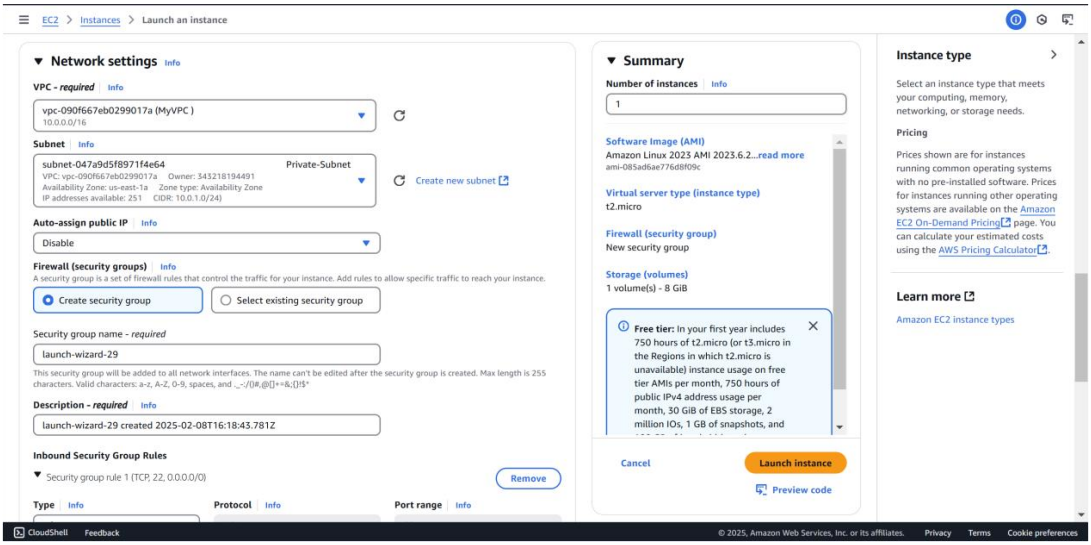
Click **Save associations**.



Step 7:

To launch a new EC2 instance in your private subnet, go to the EC2 Dashboard, click **Launch Instance**, and fill in the details: Name it "Private-Instance", choose an Amazon Linux 2 AMI (or another free-tier eligible image), select the **t2.micro** instance type, and either choose an existing key pair or create a new one for SSH access. Under **Network settings**, select your **MyVPC** and **Private-Subnet**, and make sure **Auto-assign Public IP** is disabled to keep it private. Leave all other settings as default, then click **Launch Instance**.





Step 8: Verify Internal Communication

1. **Find the private IP of your instance**:

   Go to the **EC2 Dashboard**.

   Select your instance in Private-Subnet.

   Note the **Private IPv4 address** (e.g., 10.0.1.x).

2. **Ping the Private IP**:

   If you have only one instance, you can skip this. If you have multiple instances in the private subnet, SSH into one instance and try pinging the private IP of the other instance.

Outcome

By completing this PoC of setting up a Private Network in AWS, you will:

1. Deploy a VPC with a private subnet to isolate cloud resources securely from the public internet.

2. Launch EC2 instances within the private subnet and ensure internal communication between them using private IPs.

3. Configure routing tables to enable efficient communication within the VPC while maintaining the isolation of private resources.

4. Implement security groups to allow only internal traffic between instances while restricting external access.

5. Gain practical experience in designing secure cloud architectures and foundational AWS services like VPC, EC2, and private networking.