Build a Private Subnet

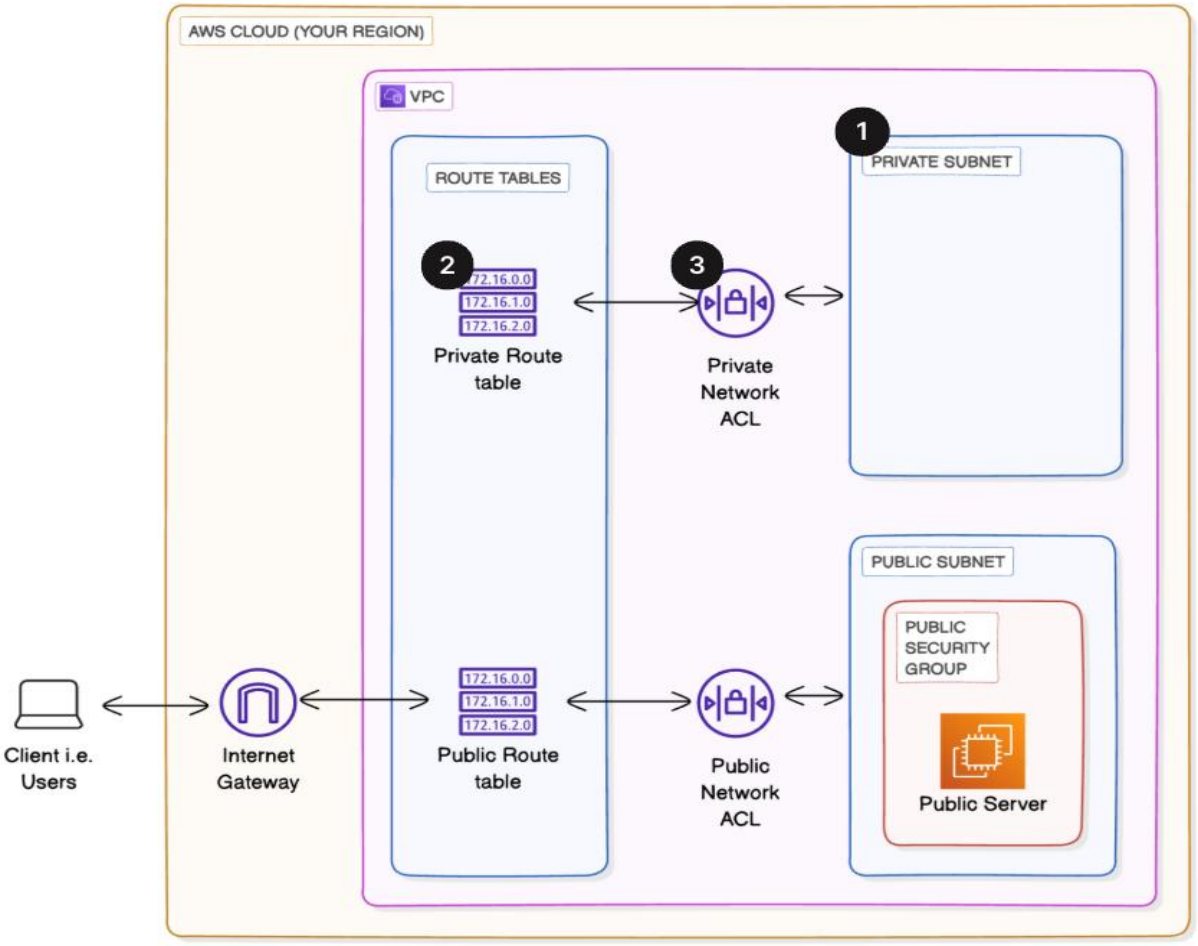
By Haroon Zaman | November 2025

Introduction

I completed this project to deepen my understanding of AWS networking and secure cloud design. After building a public subnet and enabling internet access, my goal was to learn how to create a **private network** inside AWS where resources are **not exposed to the internet** and can only communicate internally.
This step is essential for hosting secure resources such as **databases, backend services, or internal applications**.

In this project, I completed the following tasks:

- **Created a Private Subnet** – A subnet that does not provide internet access, ensuring security and isolation for internal resources.
- **Created a Private Route Table** – I set up routing rules so that resources in the private subnet only communicate within the VPC.
- **Created a Private Network ACL** – I added extra security to control inbound and outbound traffic using allow/deny rules.



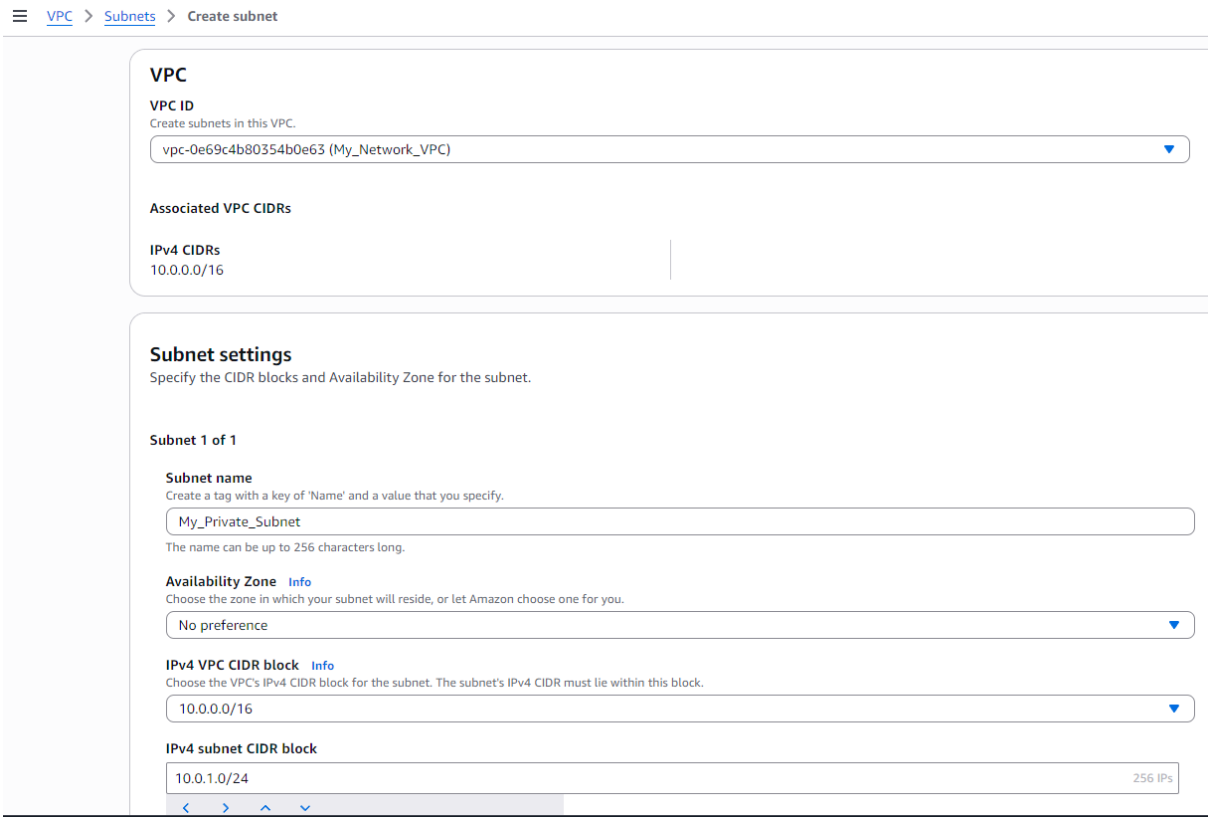
Private Subnet

A **private subnet** is a subnet inside a VPC that **does not have direct access to the internet**. It is used to host services that should remain internal and secure—for example, databases or backend servers that should not be publicly reachable.
Unlike a public subnet, a private subnet **does not route traffic to an internet gateway**.

To build my private subnet, I had to:

- Select my existing VPC.
- Define a new IPv4 CIDR block for the private subnet.
- Ensure that **no public IPs are automatically assigned**.
- Create a **private route table** that prevents internet access.
- Add a **custom network ACL** to control inbound and outbound traffic securely.

This setup helps maintain strong security and teaches how real-world cloud architectures separate **public and private resources** for best practices.

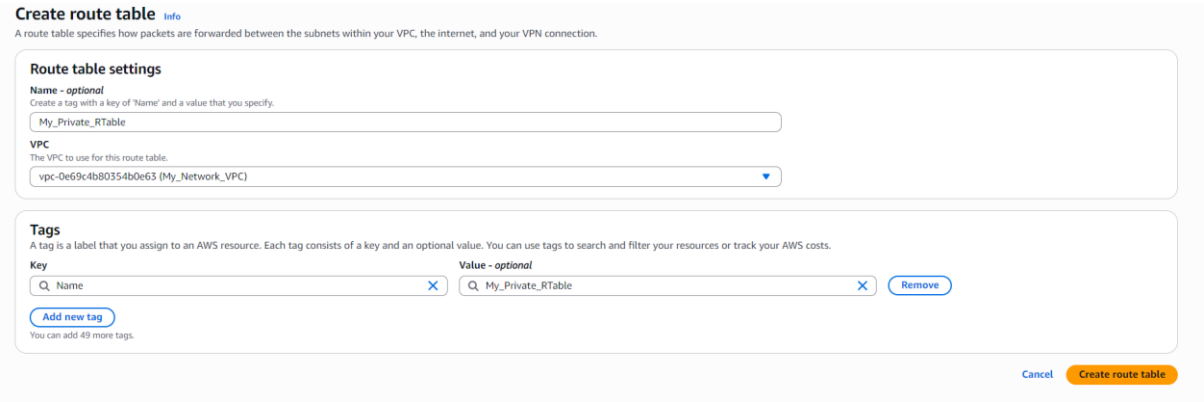


First, I went to the **VPC console** and selected **Subnets** from the left panel. Then I clicked on **Create subnet**.

- I selected my existing VPC: **MY_NETWORK_VPC**.
- I named the subnet **My_Private_Subnet**.
- For **Availability Zone**, I chose **No Preference**, which allows AWS to automatically assign a zone.

- In the **IPv4 subnet CIDR block**, I used **10.0.1.0/24**, which is part of my main VPC CIDR range (10.0.0.0/16), ensuring the subnet stays within the VPC network.

After configuring these settings, I clicked on **Create Subnet** to successfully create my private subnet.

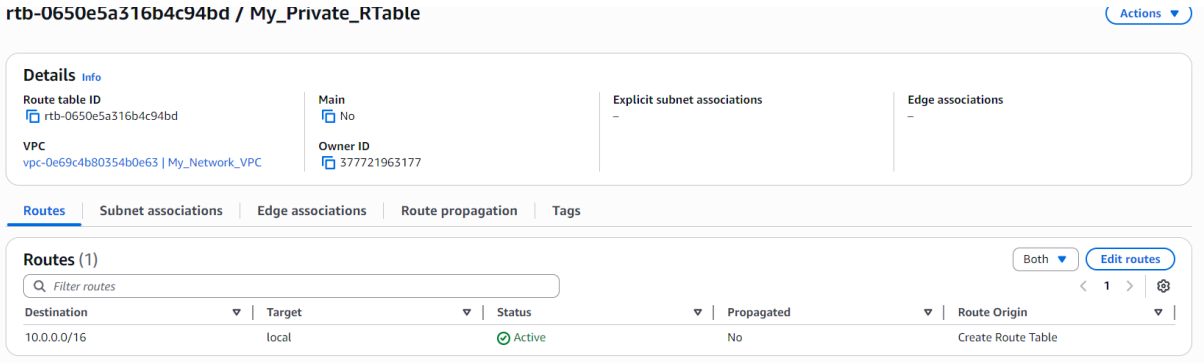


Creating a Private Route Table

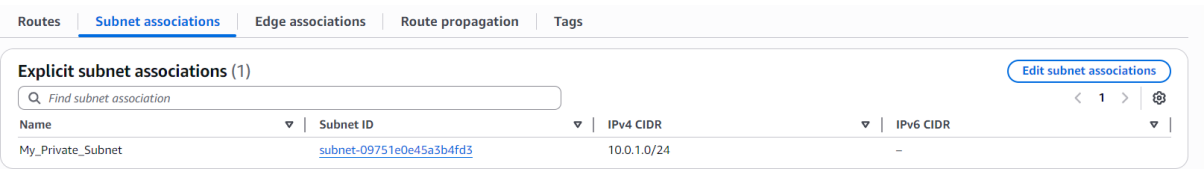
After creating **My_Private_Subnet**, I proceeded to set up a private route table for it.
I opened the **VPC console** and selected **Route Tables** from the left-side panel. Then I clicked on **Create route table**.

- I named it **My_Private_RTable**.
- I selected my VPC: **MY_NETWORK_VPC**.

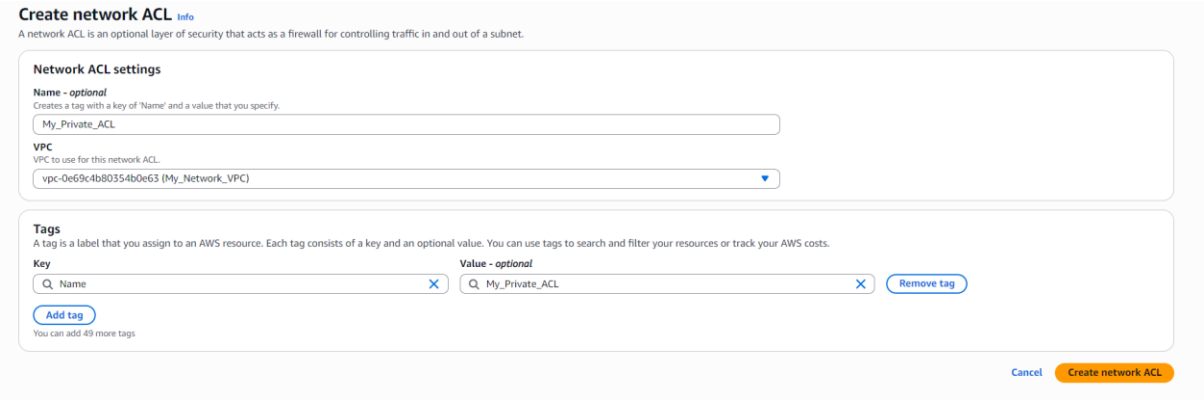
Once the details were filled, I clicked **Create route table** to complete the setup.



After creating **My_Private_RTable**, I checked the routes inside it. I noticed that there was already a default route with **Destination: 10.0.0.0/16** and **Target: local**.

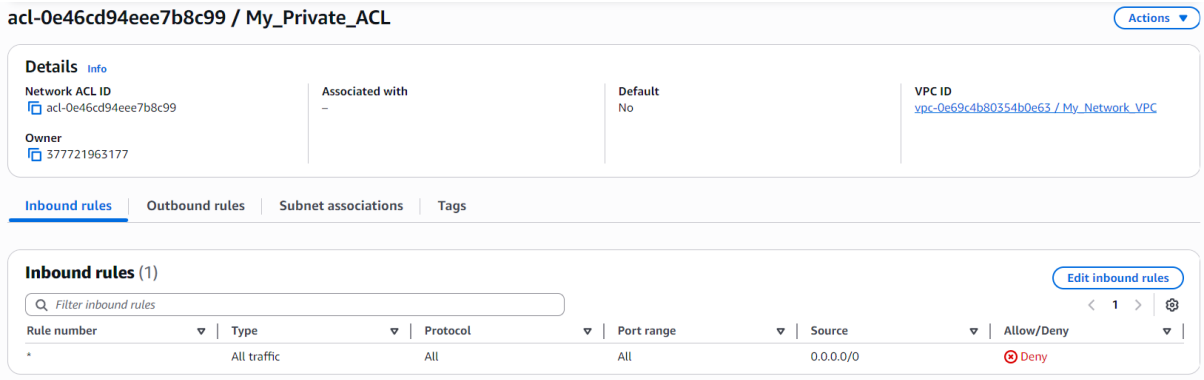


I selected **My_Private_RTable**, went to the **Subnet Associations** tab, clicked **Edit associations**, and selected **My_Private_Subnet**. After saving, the subnet was officially using this private route table — confirming it as a **private subnet** with no internet access.

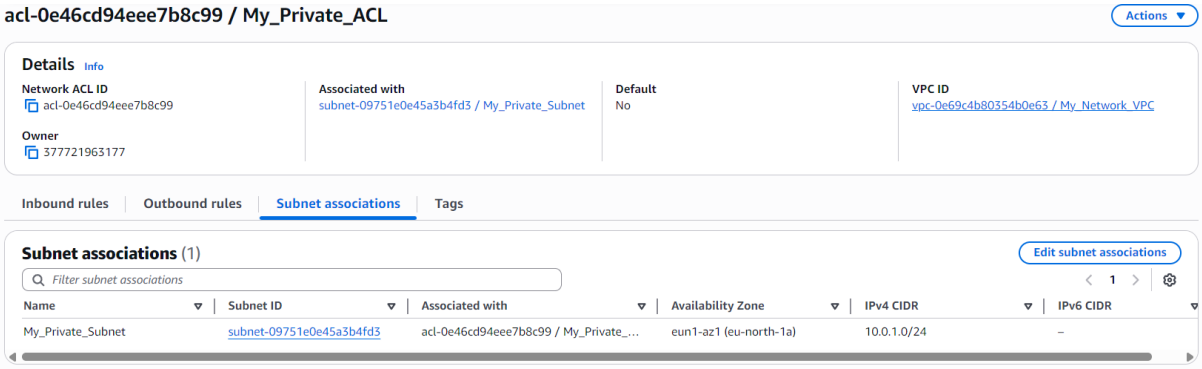


Creating a Private Network ACL

- Next, I created the **private ACL** for my subnet.
- In the **VPC console**, I went to **Network ACLs** and clicked **Create Network ACL**.
- I entered the name **My_Private_ACL**.
- I selected the appropriate **VPC (MY_Network_VPC)**.
- Finally, I clicked **Create** to finish the setup.



- After creating the **My_Private_ACL**, I reviewed the rules section.
- Both the **inbound and outbound rules were set to DENY by default**.
- I did **not modify any rules** because this subnet is meant to be fully private.
- Later, if I deploy resources inside this subnet, I will **add rules according to my needs**.
- For now, the subnet remains **fully isolated and not accessible from the internet** — which is the purpose of a private subnet.



- After creating the **My_Private_ACL**, I went to the **Subnet Associations tab**.
- I clicked on **Edit subnet associations**.
- A list of subnets appeared — from there, I selected **My_Private_Subnet**.
- Then I clicked **Save** to confirm the association.
- Now, **My_Private_Subnet is protected by my custom Private NACL**, keeping it fully secure and isolated.

Conclusion

Through this project, I successfully created a **private subnet**, a **private route table**, and a **custom network ACL**, strengthening the security of my VPC. This helped me understand how resources can be **fully isolated from the internet** while still existing inside the same VPC. I also learned the difference between **public and private subnets**, and how routing and access rules control communication in AWS.

Working with custom ACLs and route tables gave me hands-on experience with **network segmentation, access control, and secure architecture design**. This project taught me how to build a **secure cloud environment**, preparing me for more advanced AWS networking and real-world cloud deployments.