Network Mastery with AWS VPC

Author: Oluwaseun Osunsola
Environment: AWS
Project Link: https://github.com/Oluwaseunoa/DevOps-Projects/tree/main

Project Overview

In this session, we explore the core concepts of Amazon Web Services (AWS), focusing specifically on Virtual Private Clouds (VPCs). Our objective is to understand the fundamental components of VPC infrastructure, including subnets, gateways, and routing tables. Through practical demonstrations and interactive exercises, we navigate the AWS Management Console to deploy and manage these critical components effectively.

Before proceeding with setting up VPCs, ensure a solid understanding of cloud networking basics.

Project Goals

- Understand the fundamentals of Virtual Private Cloud (VPC) and its components.
- Gain hands-on experience in setting up and configuring VPC, subnets, Internet Gateway, NAT Gateway, and VPC peering connections.
- Learn how to enable internet connectivity securely within a VPC.
- Implement outbound internet access through the NAT Gateway.
- Establish direct communication between VPCs using VPC peering.

Learning Outcomes

- Acquired knowledge about VPC and its essential components, such as subnets, gateways, and route tables.
- Developed skills in creating and configuring VPC resources using the AWS Management Console.
- Learned how to set up routing tables to enable internet connectivity and outbound internet access securely.
- Gained understanding of VPC peering and its significance in connecting multiple VPCs within the same or different regions.
- Enhanced understanding of network security principles and best practices for cloud environments.

Key Concepts

What is VPC, Subnets, Internet Gateway, and NAT Gateway?

Imagine building a virtual space for the company GatoGrowFast.com so that its computers can communicate securely. That's what a **Virtual Private Cloud (VPC)** is all about—a private room in the cloud for GatoGrowFast.com's use.

Example: Think of GatoGrowFast.com's office building with different departments like HR, Finance, and IT, each with its own area and access rules. In a VPC, GatoGrowFast.com creates different sections, called **subnets**, for different parts of the business.

To connect its office to the internet, GatoGrowFast.com uses a router to control data flow. In a VPC, this is achieved with an **Internet Gateway**, allowing secure communication with the internet.

A **NAT (Network Address Translation) Gateway** acts as a secret agent between GatoGrowFast.com's computers and the internet. When a computer in the virtual office communicates with the internet, the NAT Gateway translates the message and hides the sender's identity, similar to sending a letter without a return address. This keeps GatoGrowFast.com's computers safe and anonymous online.

Note: A **router** directs data packets between networks, acting like a traffic cop for the internet. Data is broken into packets, and the router uses **routing tables**—like maps of the internet—to determine the best path for these packets based on destination IP addresses.

What is an IP Address?

An **IP address** is like a phone number for a computer, a unique set of numbers that helps devices find and communicate with each other on a network, like the internet.

Types of IP Addresses:

- **Public IP Address**: Like a home address, it's unique and allows other computers on the internet to find your device. Assigned by an Internet Service Provider (ISP), it enables global communication. Public IPs can be **dynamic** (changing periodically) or **static** (constant, used for servers or consistent connectivity).
- **Private IP Address**: Like an internal extension number in an office, used for communication within a specific network (e.g., home Wi-Fi or office network). Assigned by a router or DHCP server, private IPs are not routable over the internet and can be reused across different private networks without conflict.

IP Address Versions:

- **IPv4**: The most common type, a 32-bit numeric address written in decimal format (e.g., 192.168.0.1). Each octet ranges from 0 to 255, divided into classes A, B, and C for host addressing.
- **IPv6**: Designed to replace IPv4 due to address exhaustion, a 128-bit hexadecimal address (e.g., 2001:0db8:85a3:0000:0000:8a2e:0370:7334), offering a vast address space.

What is CIDR?

CIDR (Classless Inter-Domain Routing) simplifies talking about groups of IP addresses. Instead of listing each address, CIDR uses a shorthand, like saying "all houses on Main Street" instead of naming each house.

Example: For the IP address 192.168.1.0, CIDR notation like 192.168.1.0/24 refers to all addresses from 192.168.1.0 to 192.168.1.255.

Calculating Available IP Addresses in a CIDR Block: Use the formula: `2^(32 - CIDR notation) - 2` (subtracting 2 for the network and broadcast addresses).

Example: For 192.168.1.0/24:

- `2^(32 - 24) - 2 = 2^8 - 2 = 256 - 2 = 254` available IP addresses.

What is a Gateway?

Gateways are like doorways between networks, enabling data to travel between a local network and others, like the internet. They act as a traffic cop, directing data to its destination.

Example: In a city with neighborhoods, to visit a friend in Neighborhood B from Neighborhood A, you pass through a gateway connecting the two. Similarly, a network gateway connects your local network to the internet.

What is a Route Table?

A **route table** is like a map guiding data around a network. It lists destinations and paths (routes) to reach them. Devices consult the route table to determine where to send data packets.

Example: To send data to a website, a computer checks its route table to find the gateway to the internet. The router then forwards the data to the next stop, ensuring it reaches its destination.

Connection Between Gateway and Route Table

- **Gateways**: Devices (e.g., routers, firewalls) serving as entry/exit points between networks with different IP ranges. They receive packets and determine the next destination based on routing information.
- **Route Tables**: Maintained by networking devices, they list destination networks and the next hop (gateway) to reach them. Devices use route tables to find the best path for packets.
- **Connection**: When a device sends data outside its local network, it checks the route table to identify the gateway. The packet is forwarded to the gateway, which continues routing until the packet reaches its destination.

Difference Between Internet Gateway and NAT Gateway

- **Internet Gateway**: A door to the internet for a subnet, allowing resources (e.g., EC2 instances) to send and receive internet traffic. It enables two-way communication.
- **NAT Gateway**: A one-way street for subnet traffic, allowing resources to access the internet without allowing incoming internet traffic, enhancing security.

What is VPC Peering?

VPC peering connects two virtual offices (VPCs) in the cloud for direct communication, like neighboring offices sharing files without a middleman. By default, EC2 instances in different VPCs cannot communicate. VPC peering establishes a direct network connection between VPCs.

Why VPC Peering?

It enables different parts of a cloud network (e.g., development and marketing VPCs) to share data securely and efficiently.

Key Points:

- VPCs require peering, VPN, or AWS Direct Connect for connectivity.
- Subnets within the same VPC communicate by default via AWS-configured route tables.
- EC2 instances in the same or different subnets within a VPC communicate if security group rules and route tables allow.
- **VPC Peering Basics**:

- Allows direct communication using private IP addresses.
- Supports same or different regions and AWS accounts.
- CIDR blocks must not overlap.
- Requires proper Security Group and Network Access Control List (NACL) configurations.
- No transitive traffic (traffic cannot flow through a peered VPC to another VPC).
- Route tables must include routes for the peer VPC's CIDR block.
- Limits exist on the number of peering connections and route entries.

What is a VPC Endpoint?

A **VPC endpoint** is a secure, dedicated tunnel between a VPC and an AWS service (e.g., S3), bypassing the public internet. It ensures private, safe access to resources.

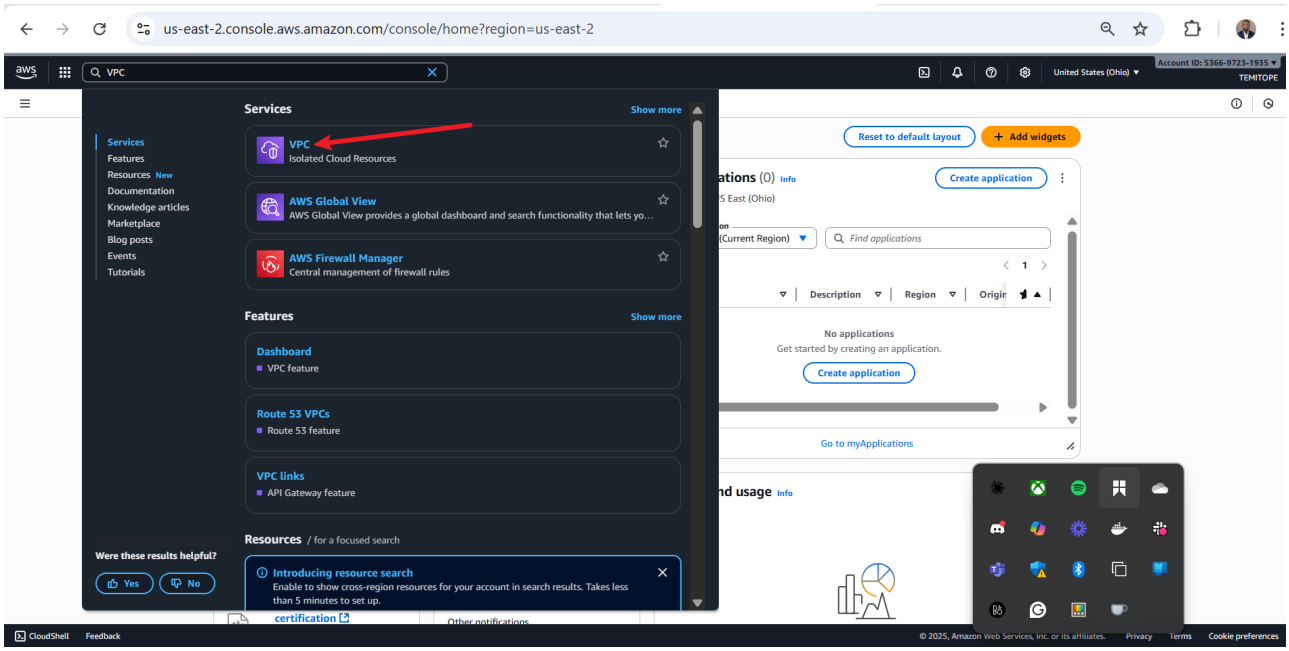
Problem Context: Backing up data from an EC2 instance to an S3 bucket typically goes over the internet, risking sensitive data exposure. A VPC endpoint creates a private connection, keeping data secure.

Note: An **EC2 instance** is a virtual server in AWS for running applications, offering scalable computing power for hosting websites, software, or data processing.

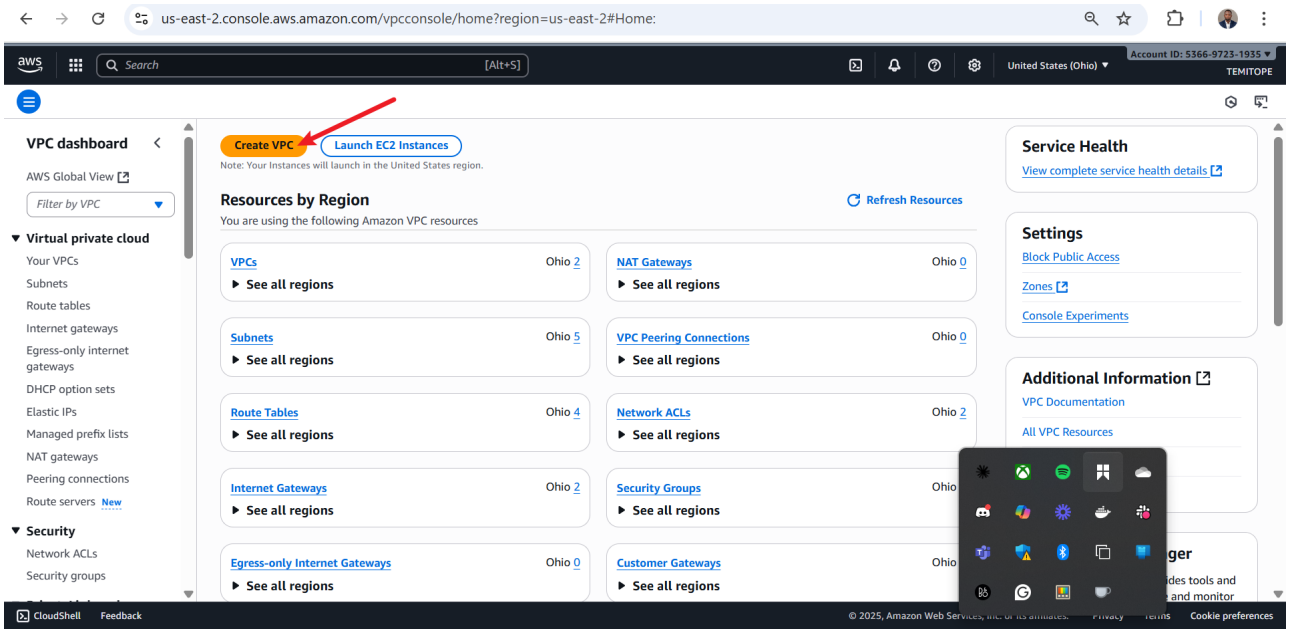
Practical Steps

Part 1: Setting Up a Virtual Private Cloud (VPC)

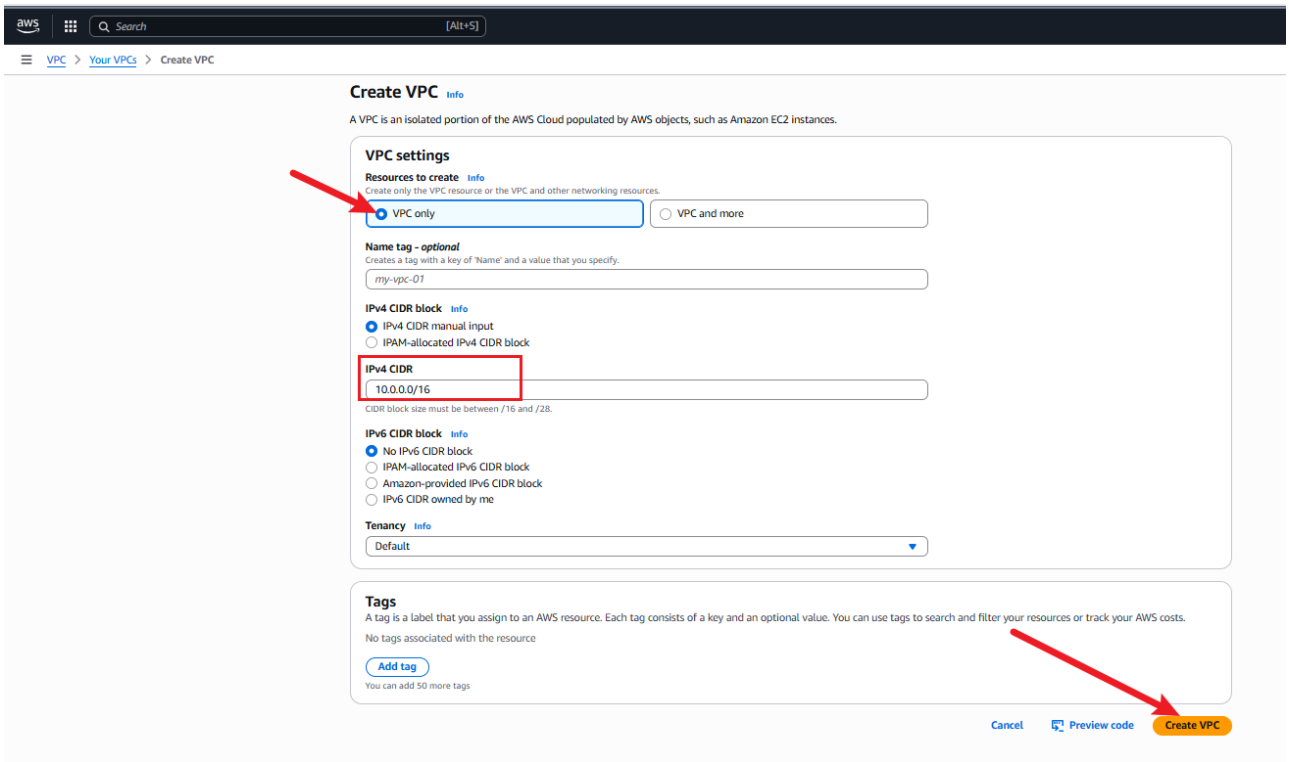
1. Navigate to the search bar, enter 'VPC', and click on the VPC service.



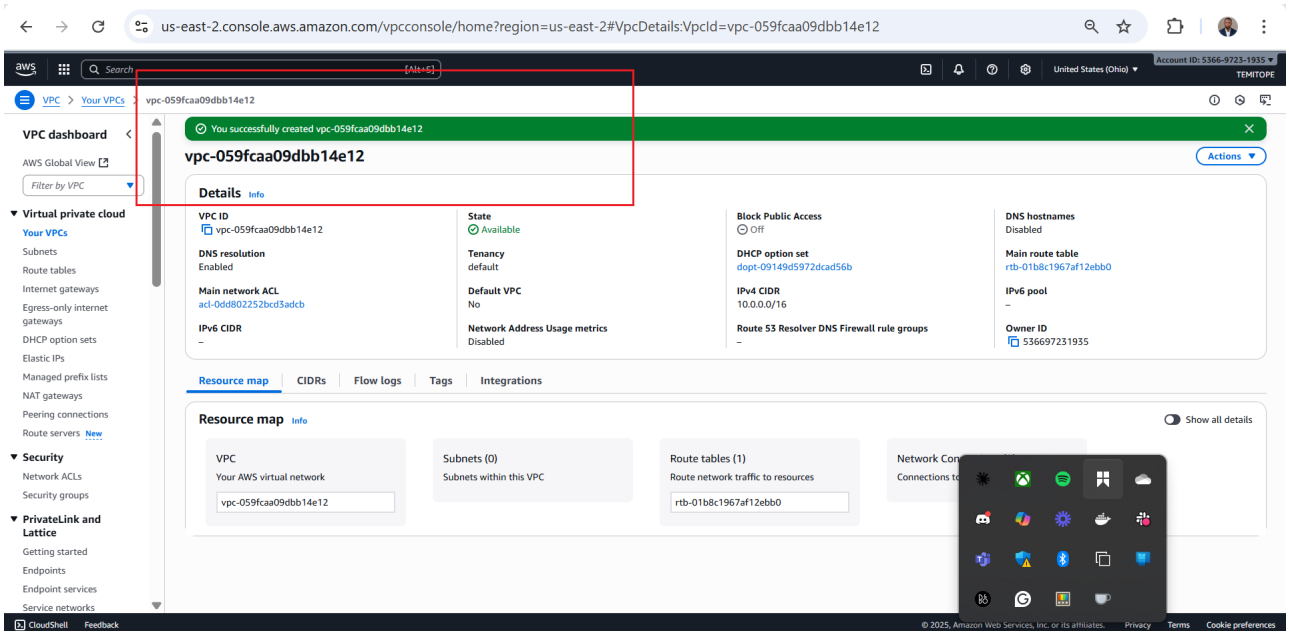
2. On the VPC dashboard, click **Create VPC**.



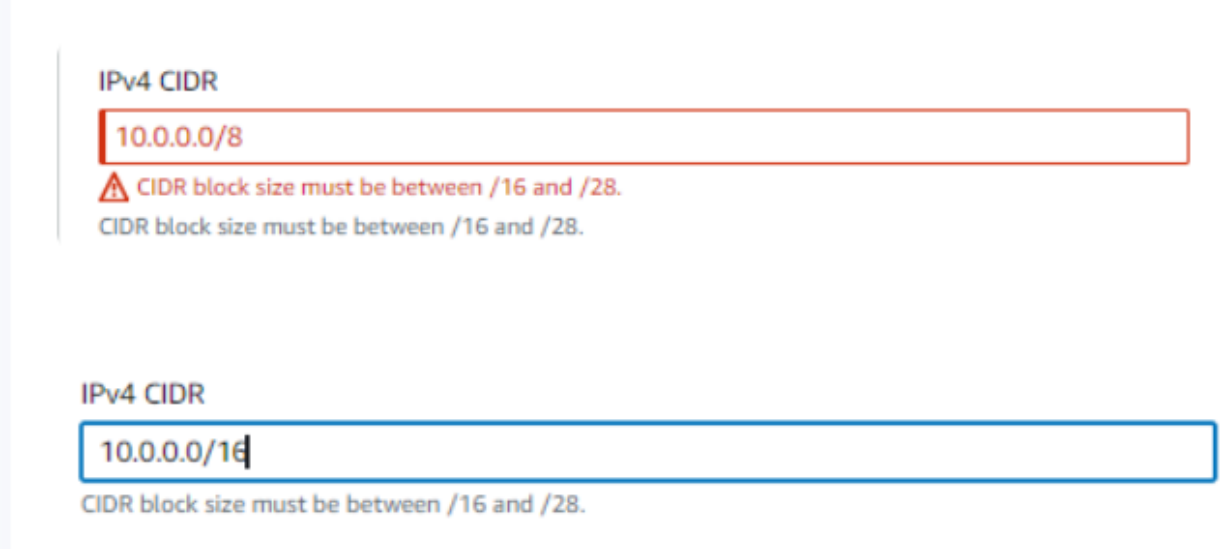
3. Select **VPC only**, set the IPv4 CIDR block (e.g., 10.0.0.0/16), and click **Create VPC**.



4. VPC successfully created.

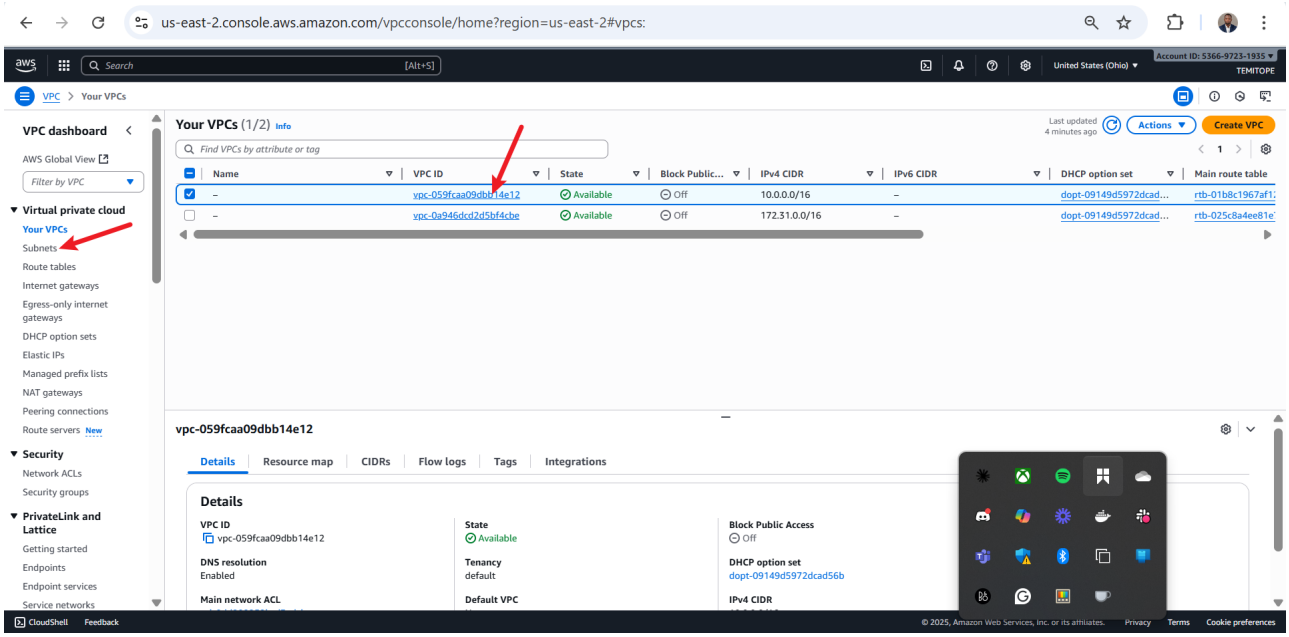


5. **Note**: If a CIDR block size error occurs, ensure the block size is between /16 and /28.

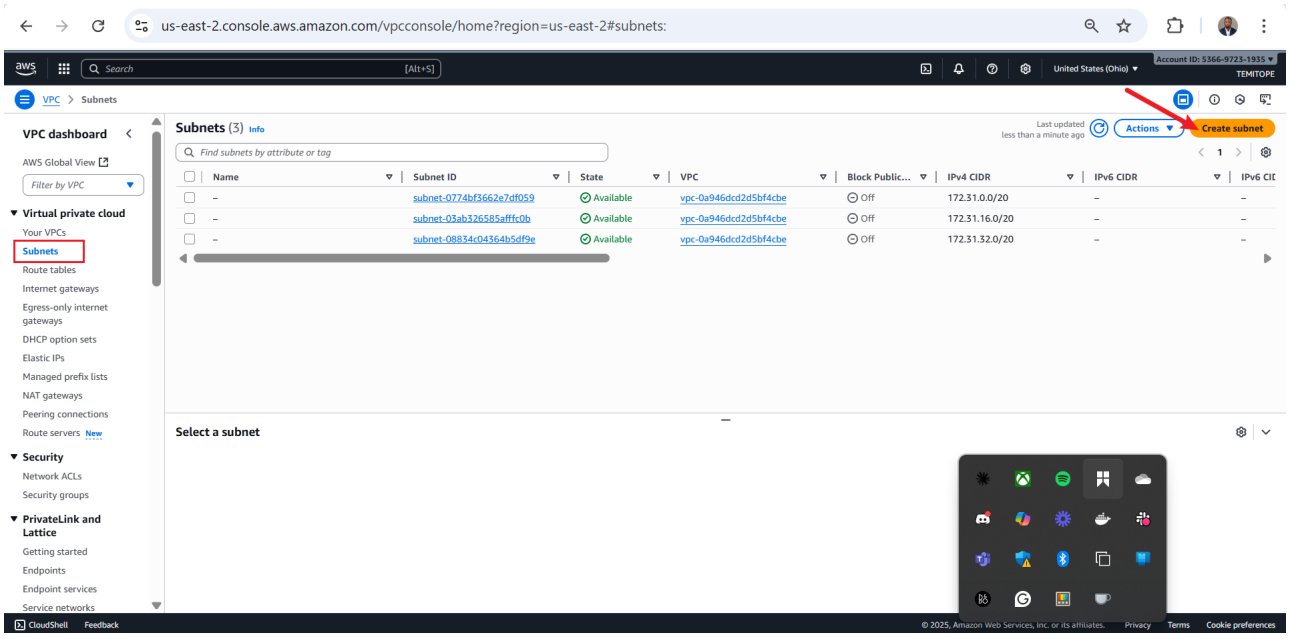


Part 2: Configuring Subnets within the VPC

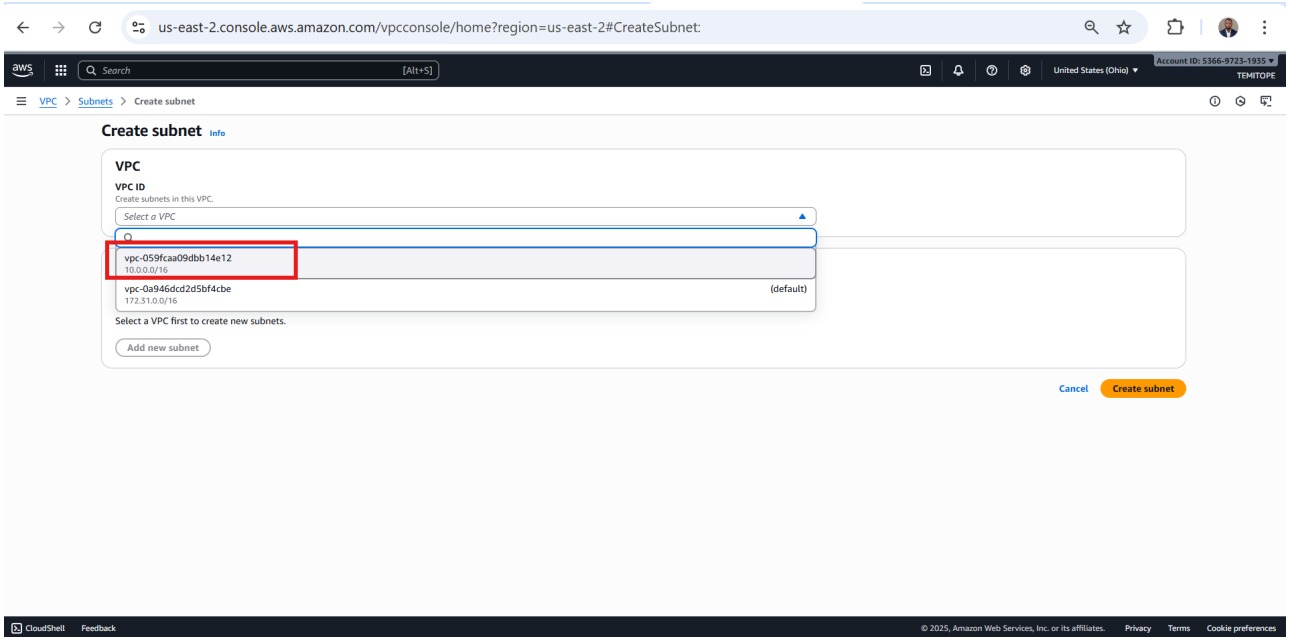
1. From the created VPC, click **Subnets** to create a subnet.



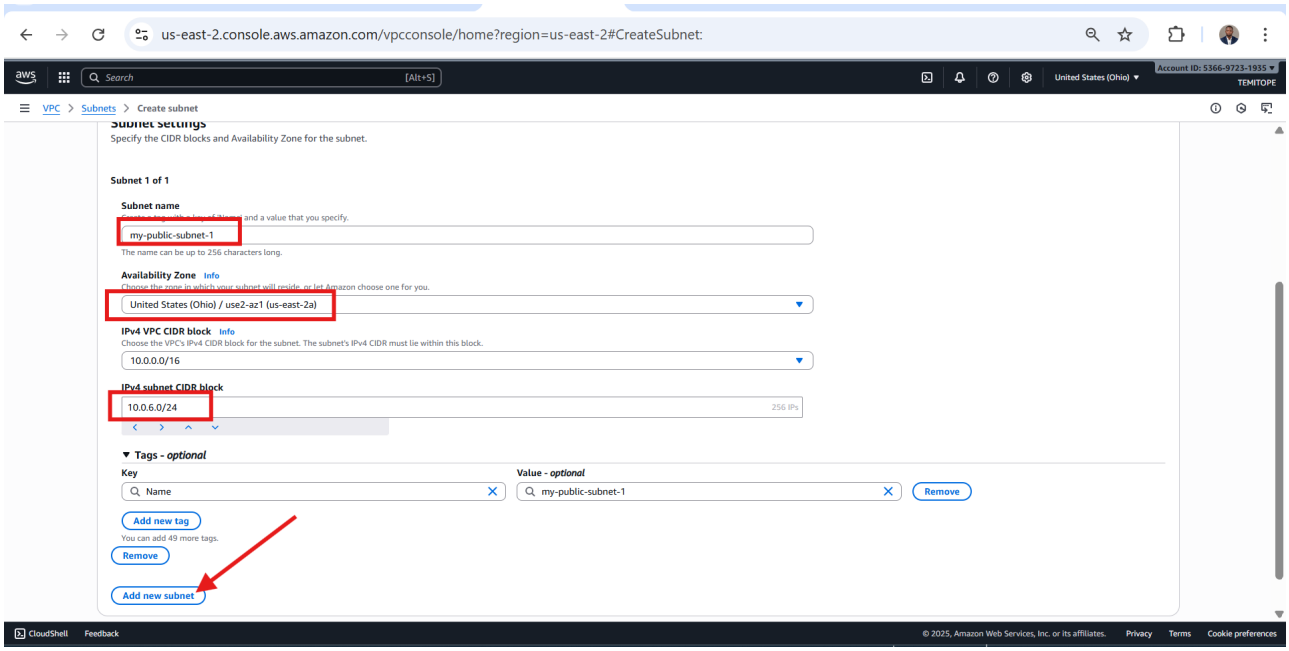
2. On the subnet dashboard, click **Create subnet**.



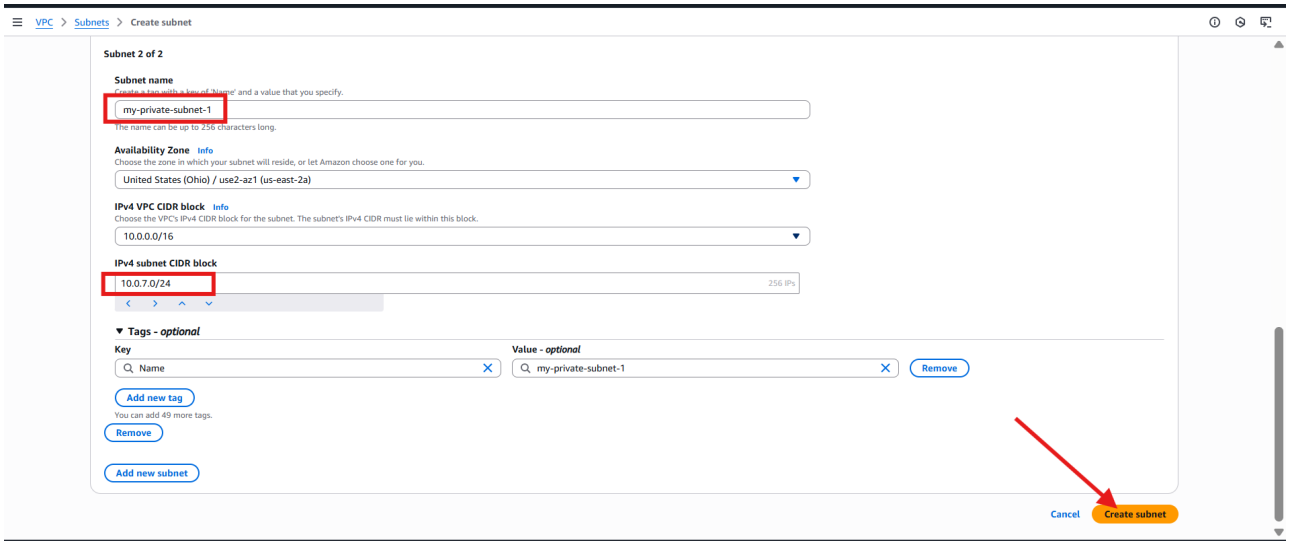
3. Select the VPC created in Part 1.



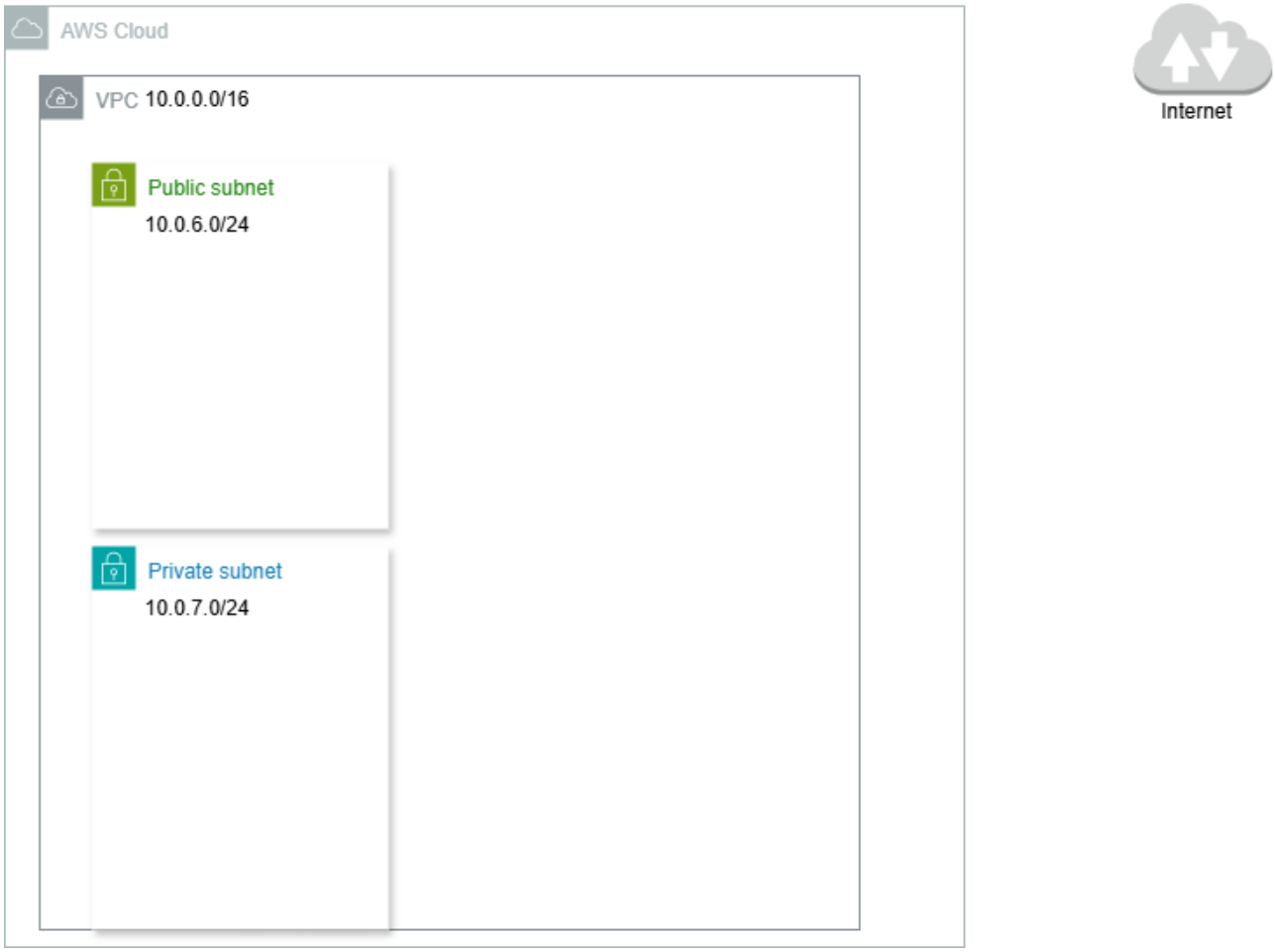
4. Enter subnet name (e.g., `my-public-subnet-1`), set availability zone, specify IPv4 CIDR block (e.g., 10.0.1.0/24), and click **Add new subnet**.



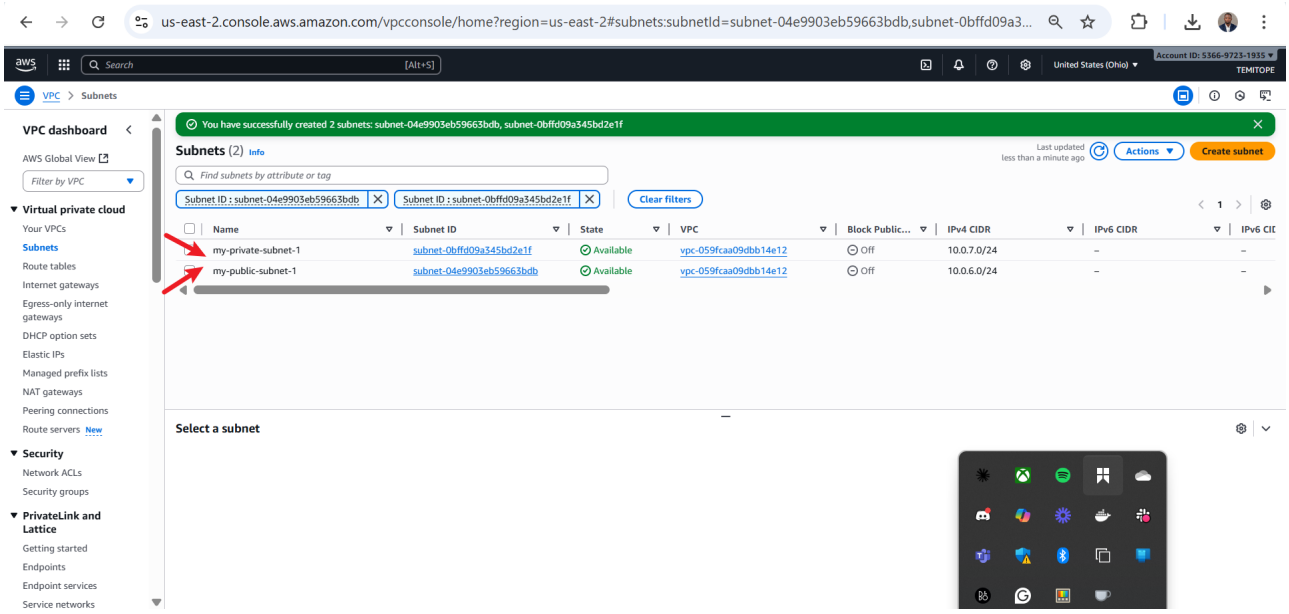
5. Enter subnet name (e.g., `my-private-subnet-1`), set availability zone, specify IPv4 CIDR block (e.g., 10.0.2.0/24), and click **Create subnet**.



6. View the architecture.

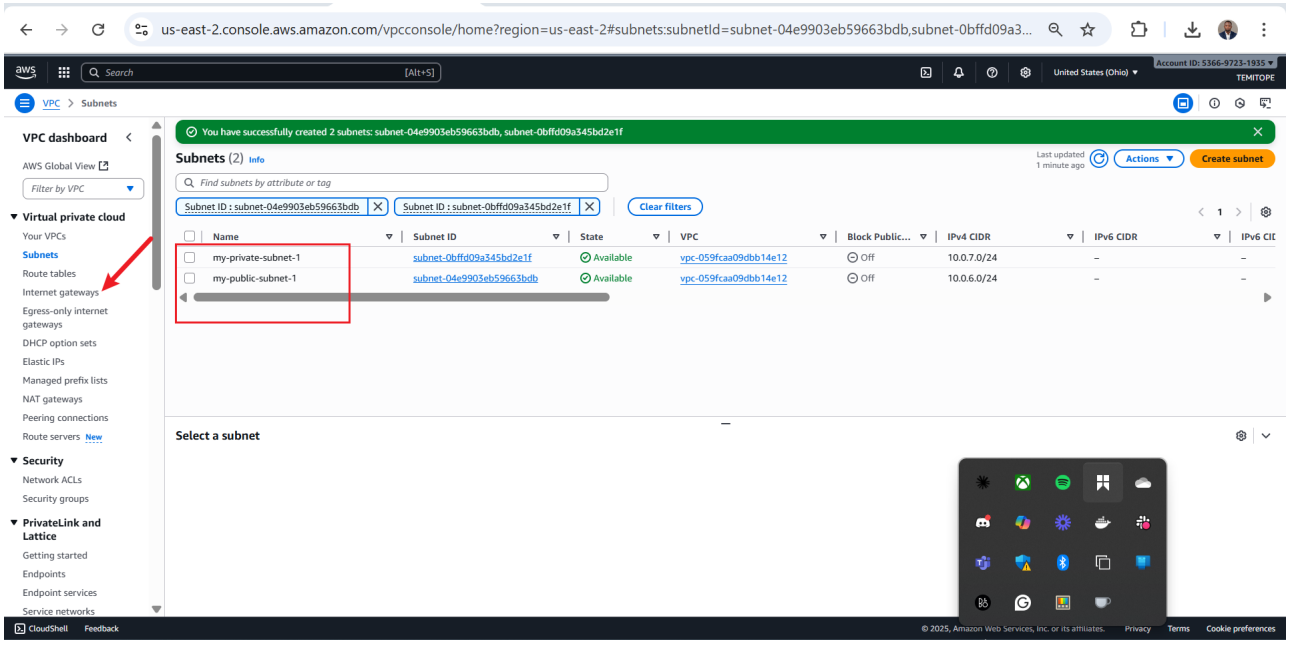


7. Subnets `my-public-subnet-1` and `my-private-subnet-1` successfully created.

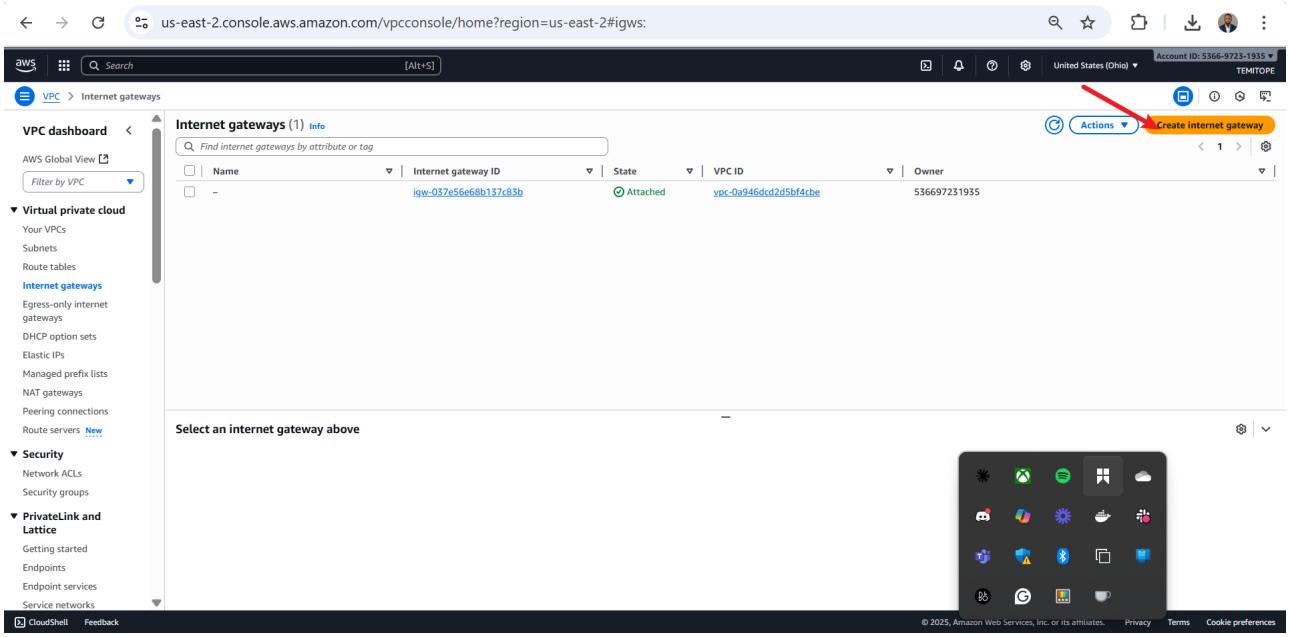


Part 3: Creating Internet Gateway and Attaching it to VPC

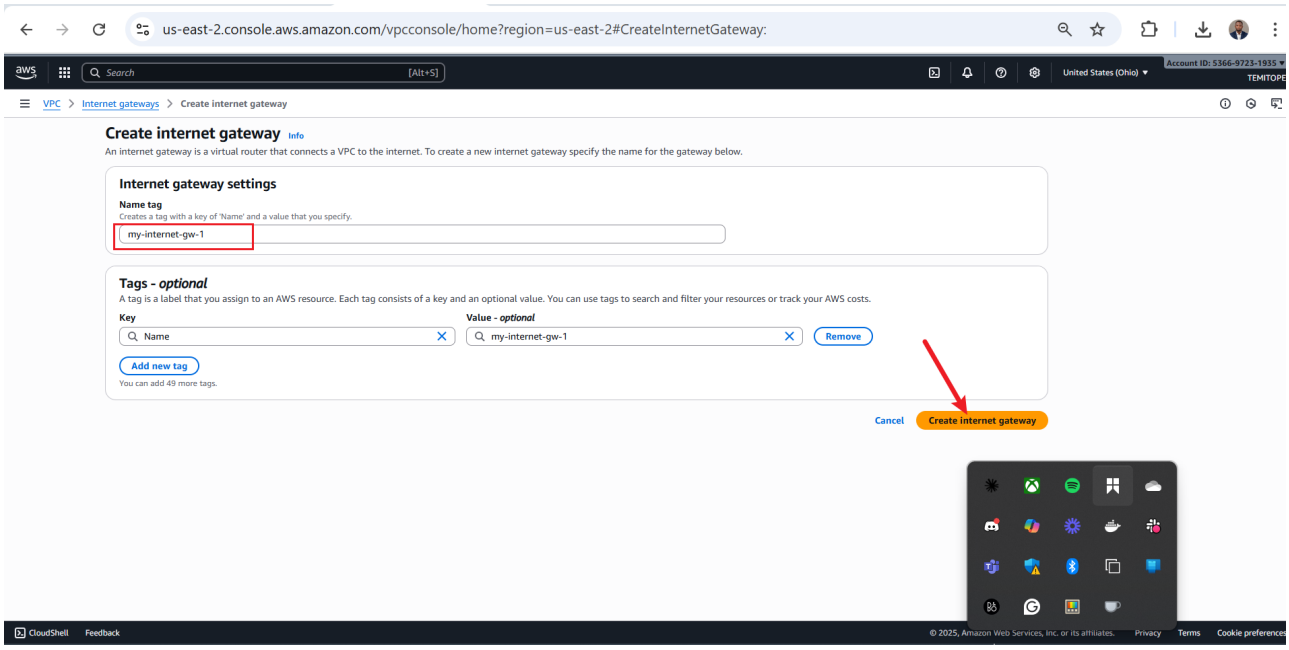
1. Click **Internet Gateways** to connect the public subnet to the internet.



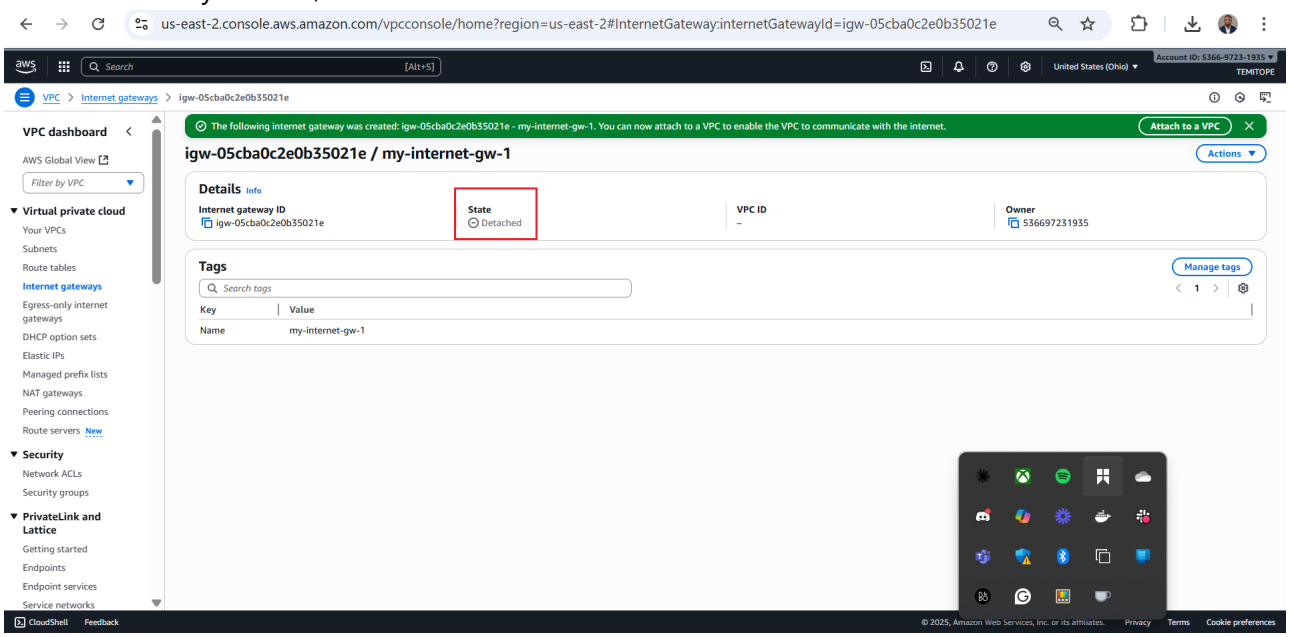
2. Click **Create Internet Gateway**.



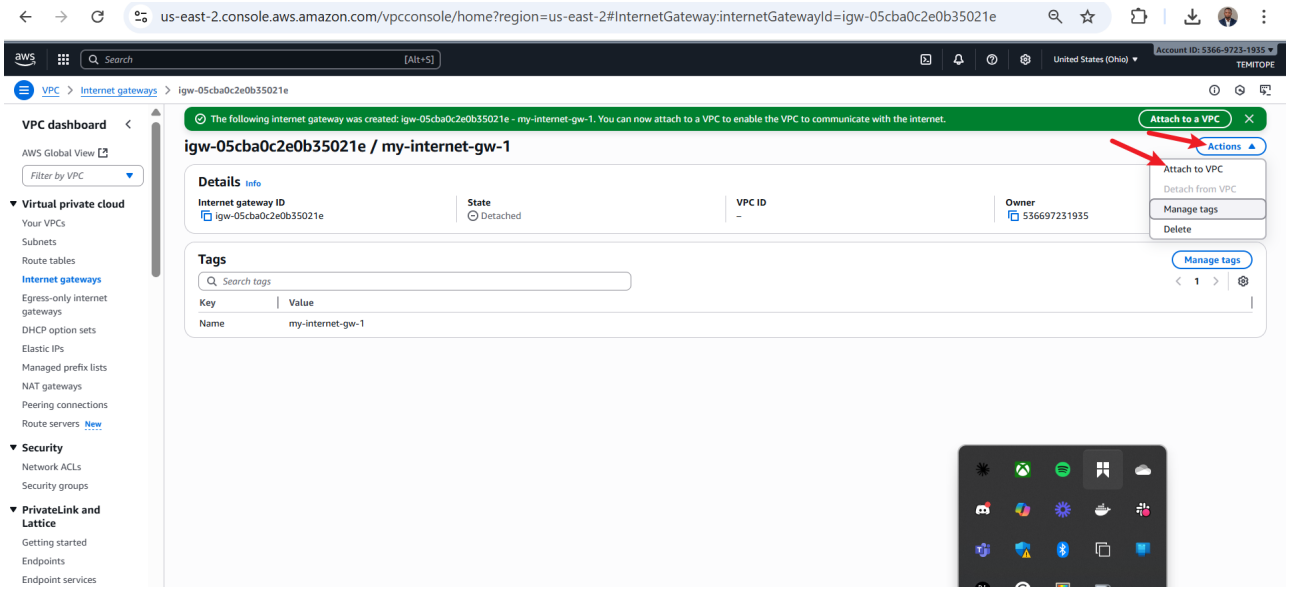
3. Name the Internet Gateway and click **Create Internet Gateway**.



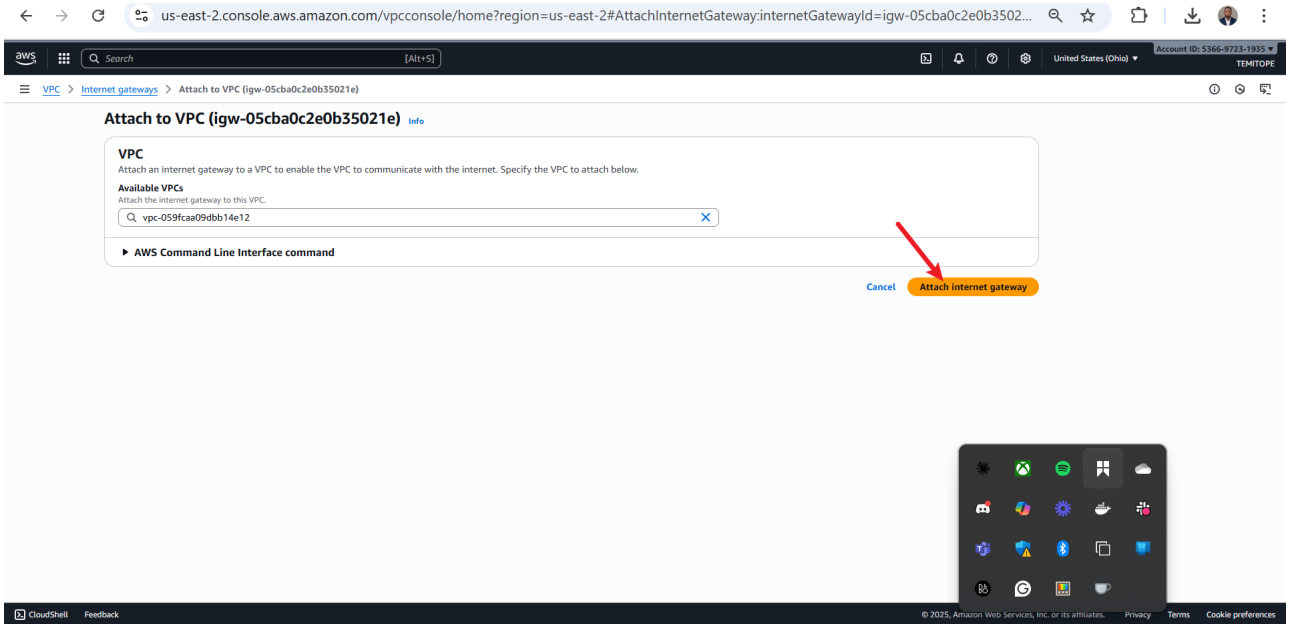
4. Internet Gateway created, note it is detached.



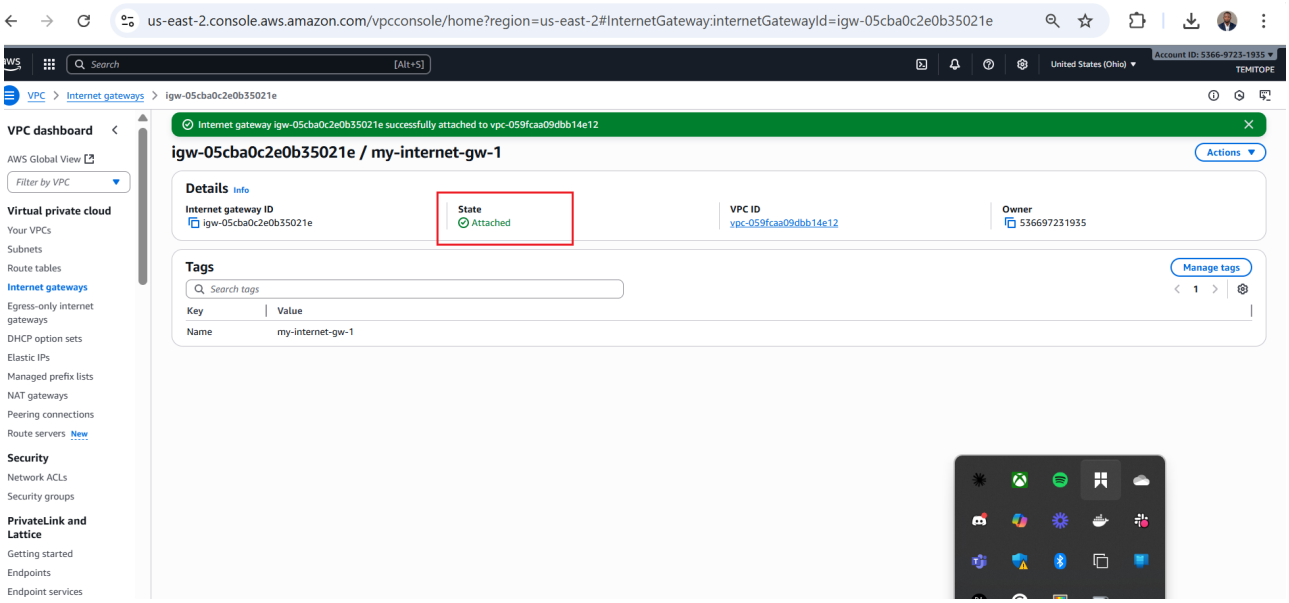
5. Click **Actions** and **Attach to VPC**.



6. Select the VPC and click **Attach Internet Gateway**.

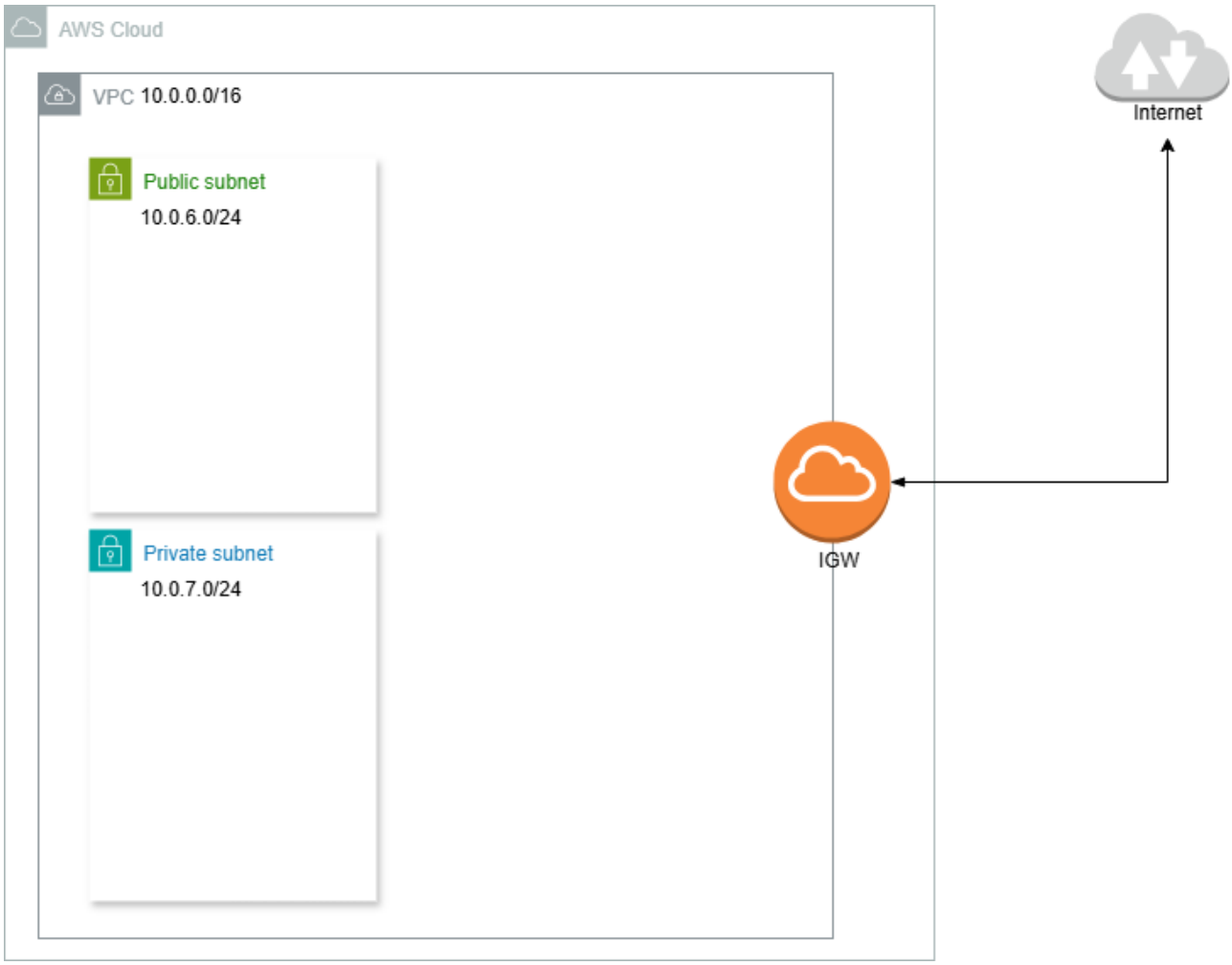


7. Internet Gateway successfully attached to VPC.

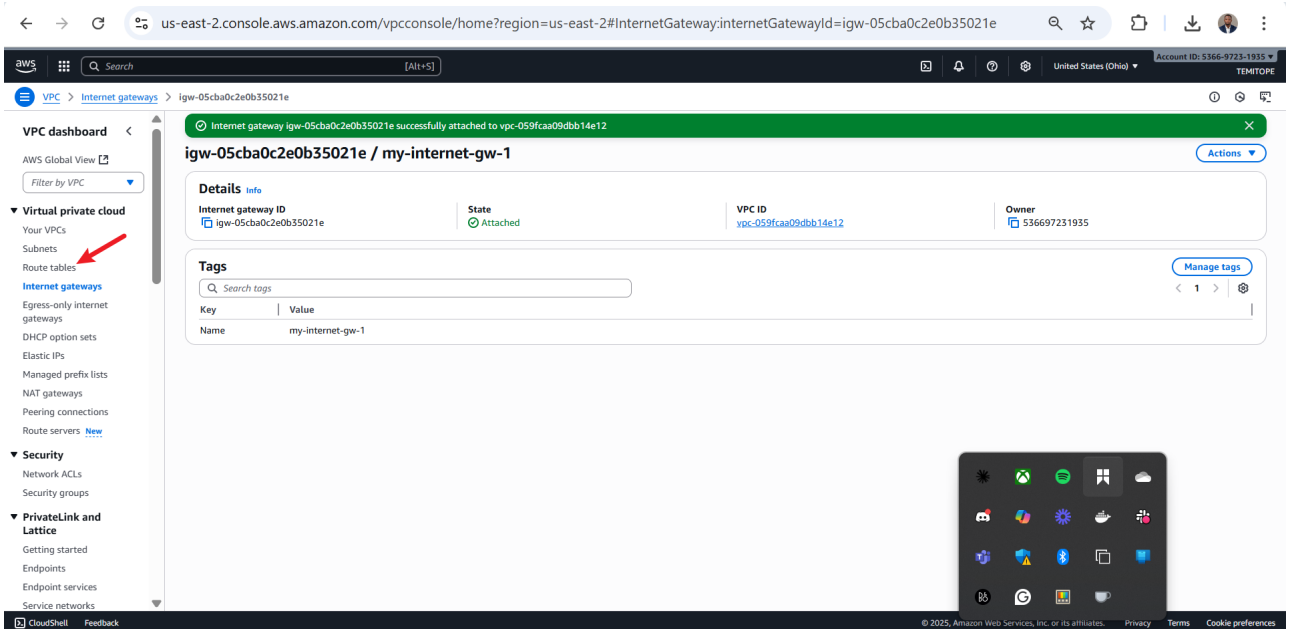


Part 4: Enabling Internet Connectivity with the Internet Gateway by Setting Up Routing Tables

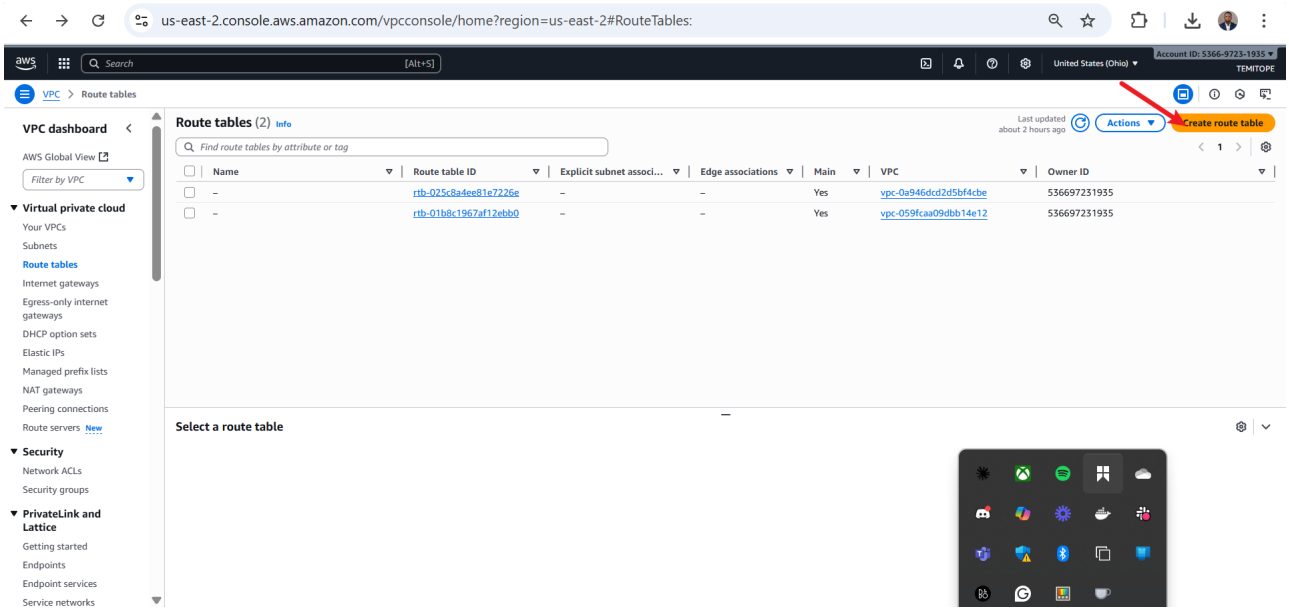
1. View the current architecture.



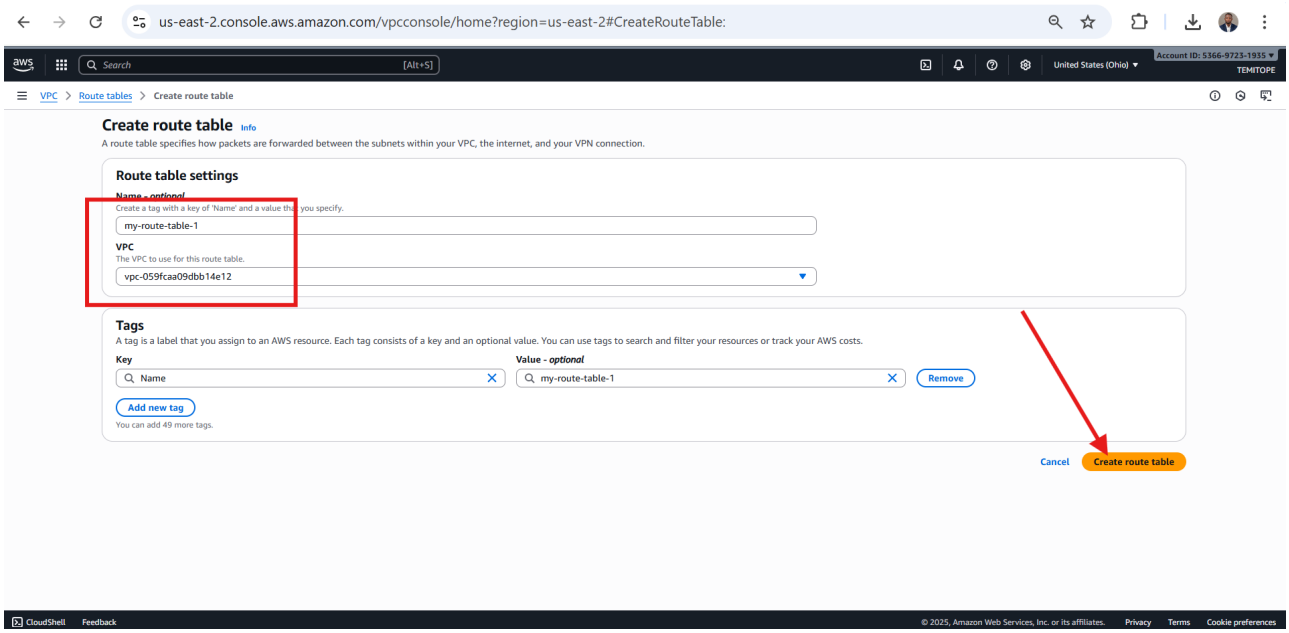
2. Click **Route Tables**.



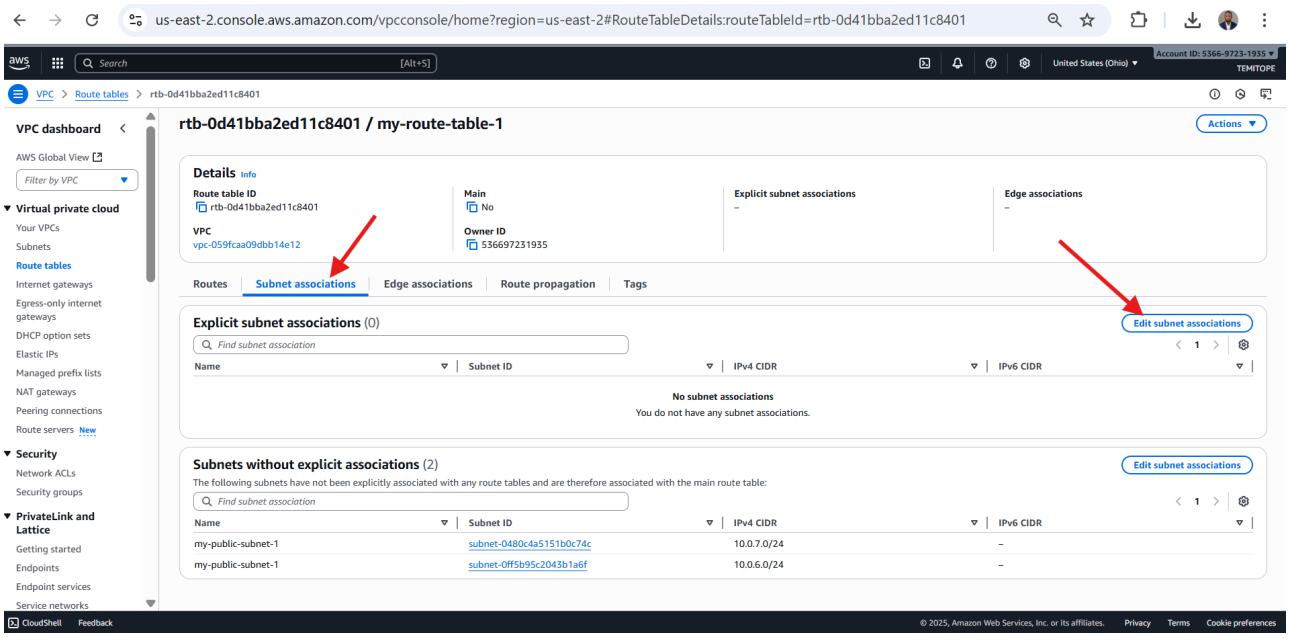
3. Click **Create route table**.



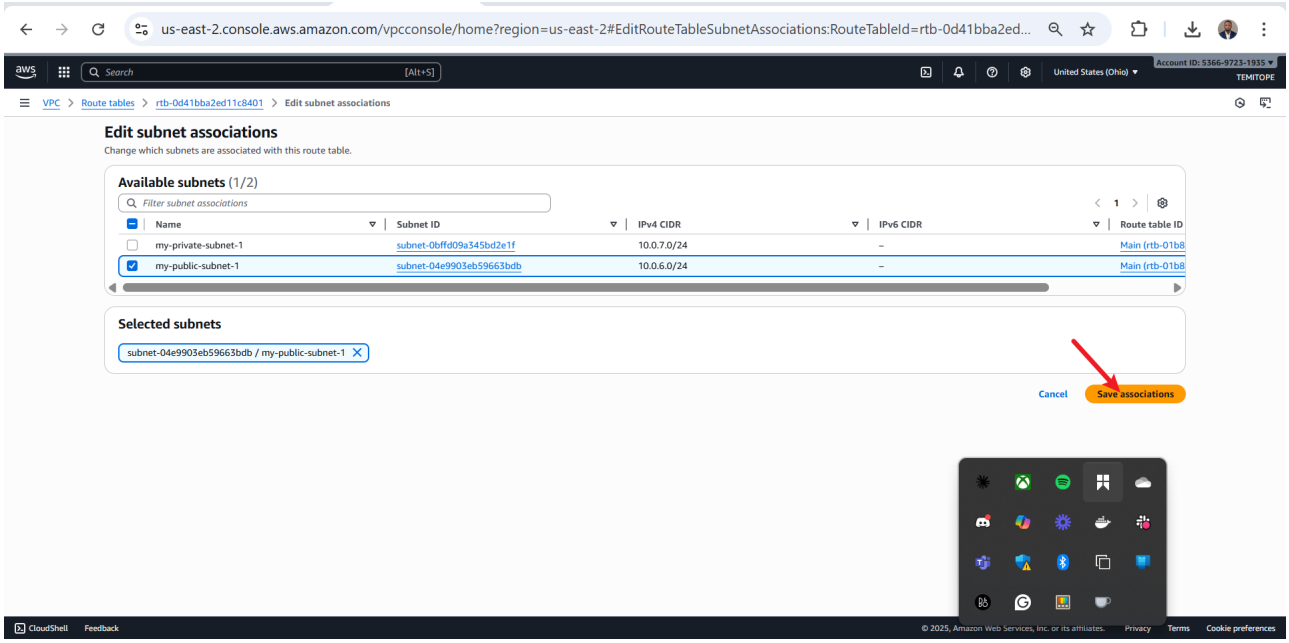
4. Name the route table, select the VPC, and click **Create route table**.



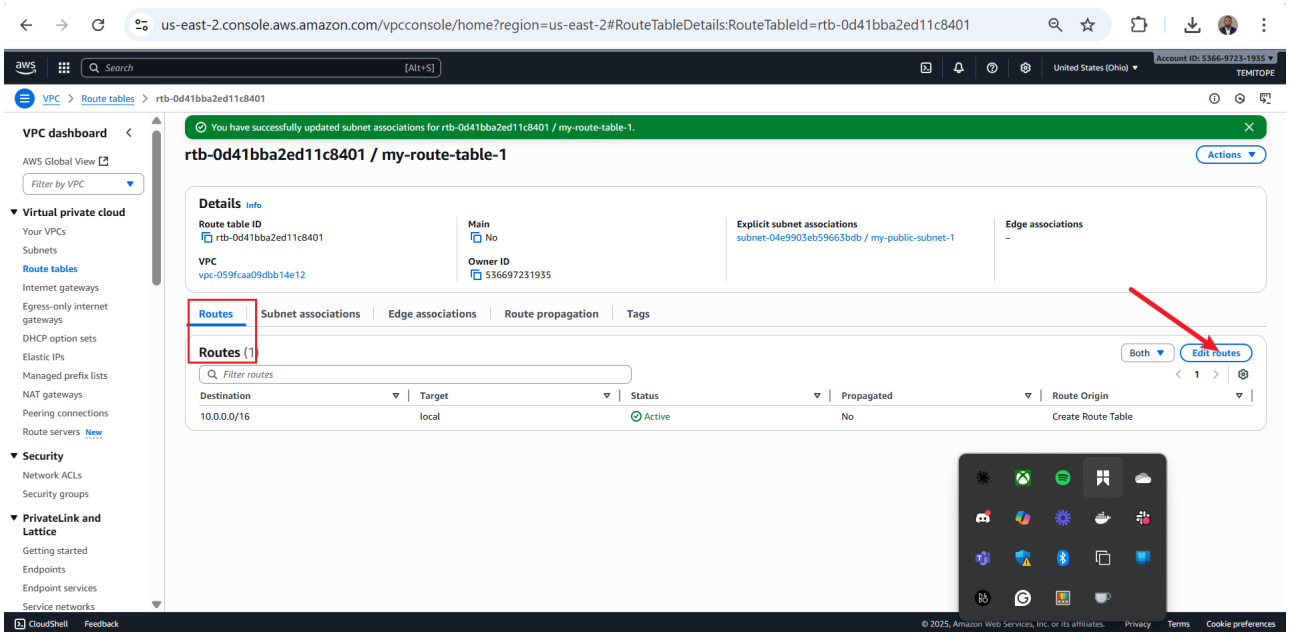
5. Click **Subnet associations**, then **Edit subnet associations**.



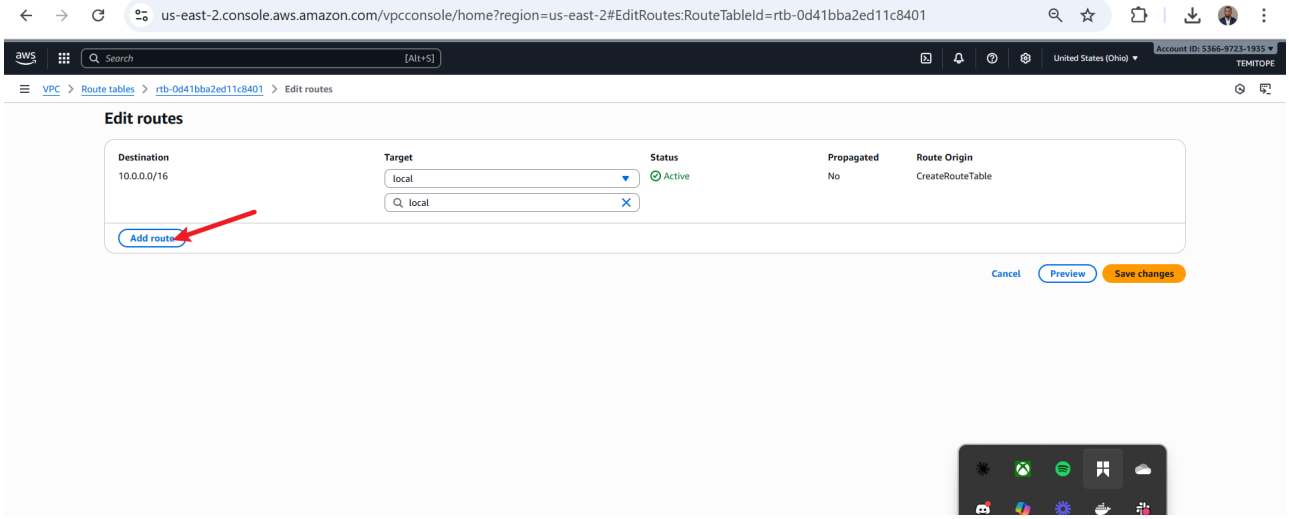
6. Select the public subnet and click **Save associations**.



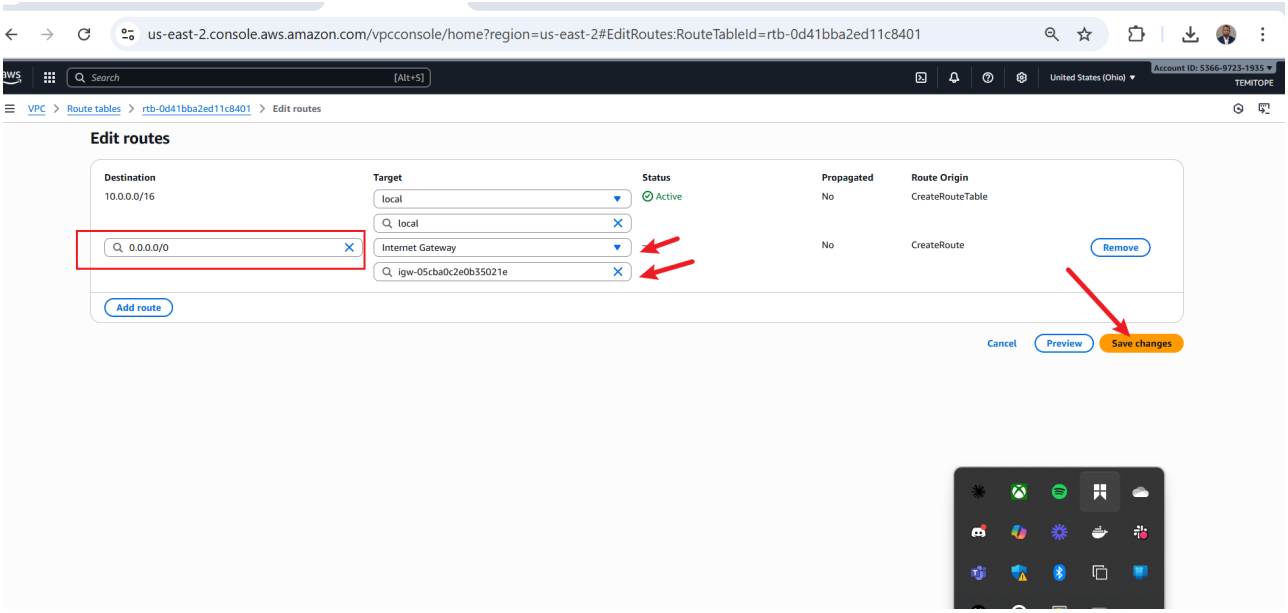
7. Under **Routes** tab, click **Edit routes**.



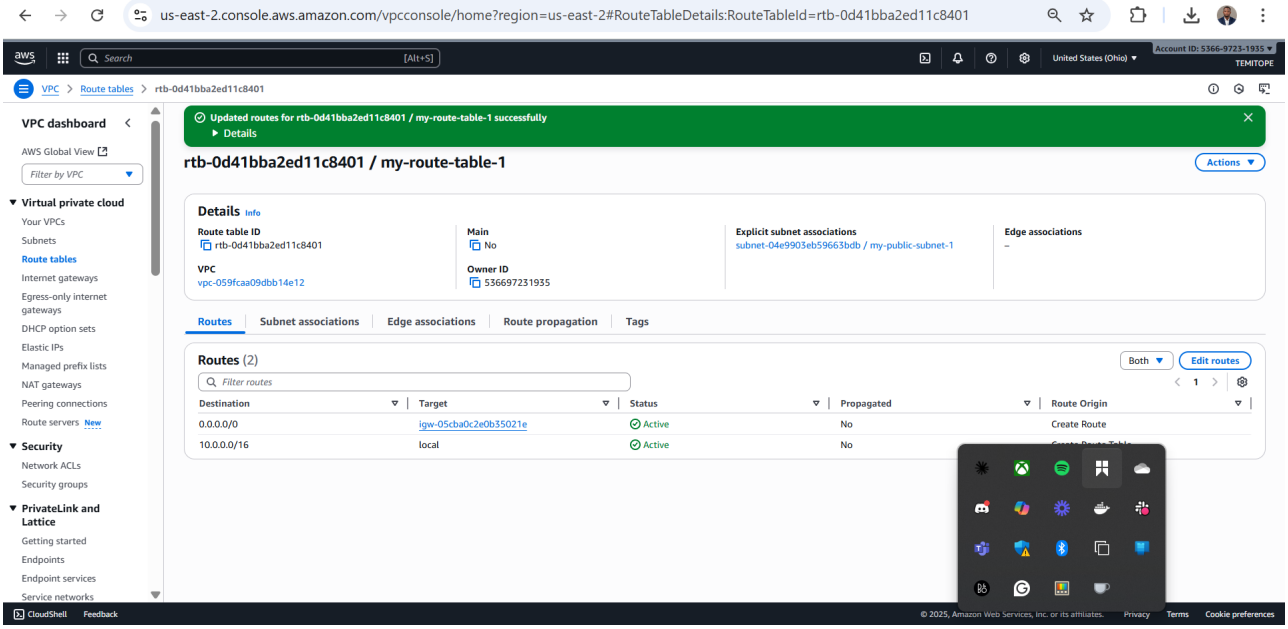
8. Click **Add route**.



9. Set **Destination** to `0.0.0.0/0`, **Target** to the created Internet Gateway, and save changes.

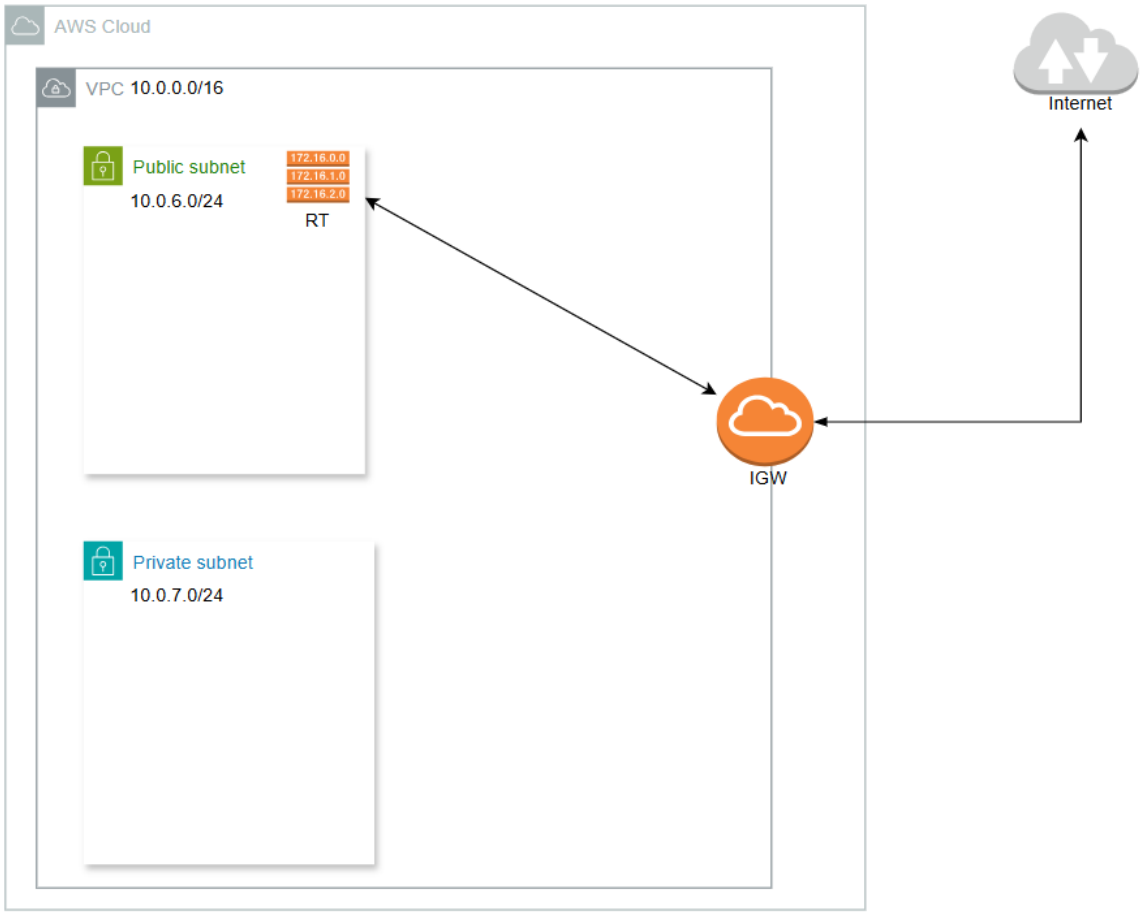


10. Route table updated successfully.

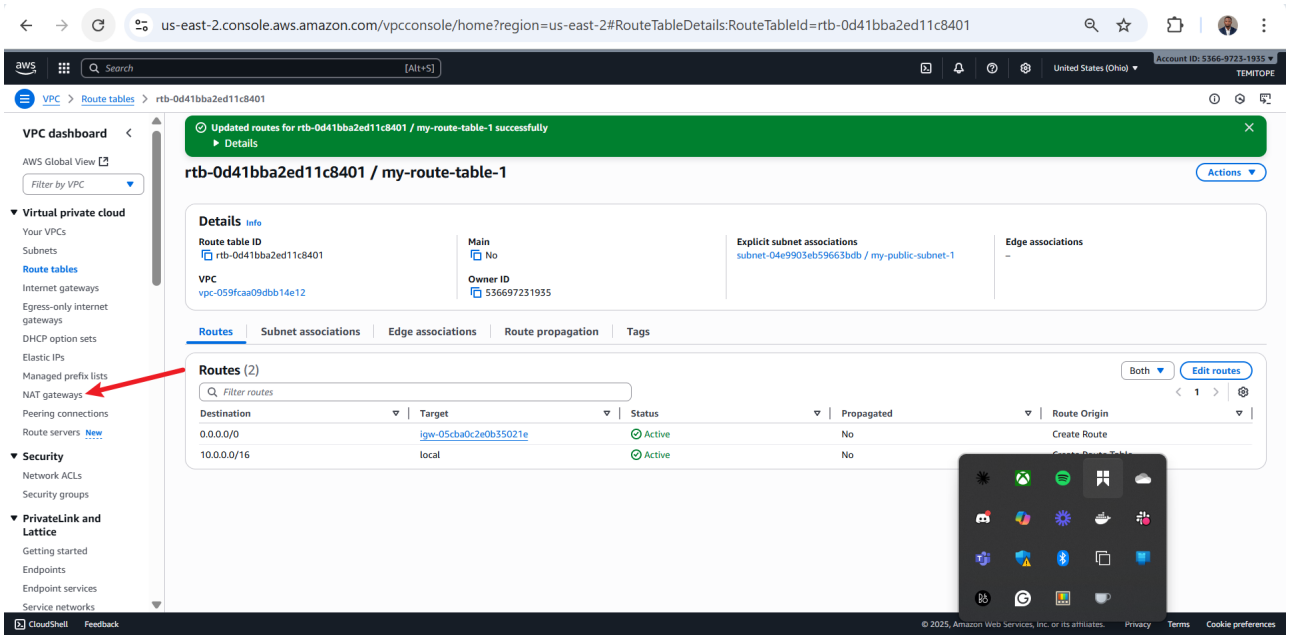


Part 5: Enabling Outbound Internet Access through NAT Gateway

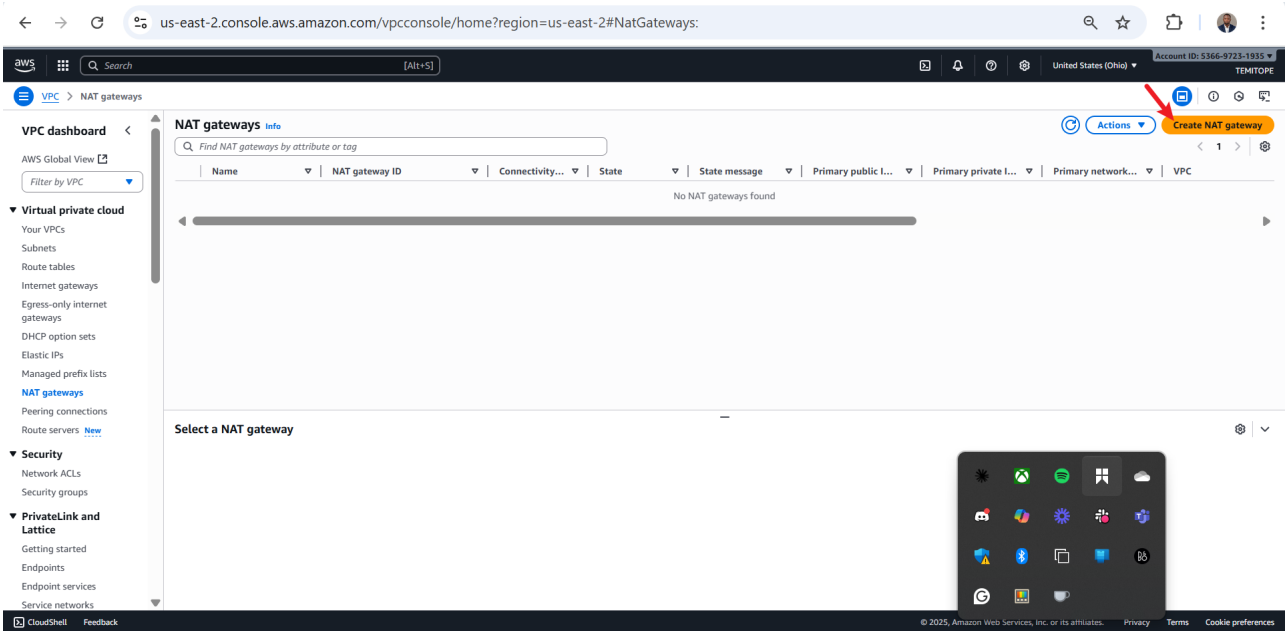
1. View the current VPC architecture.



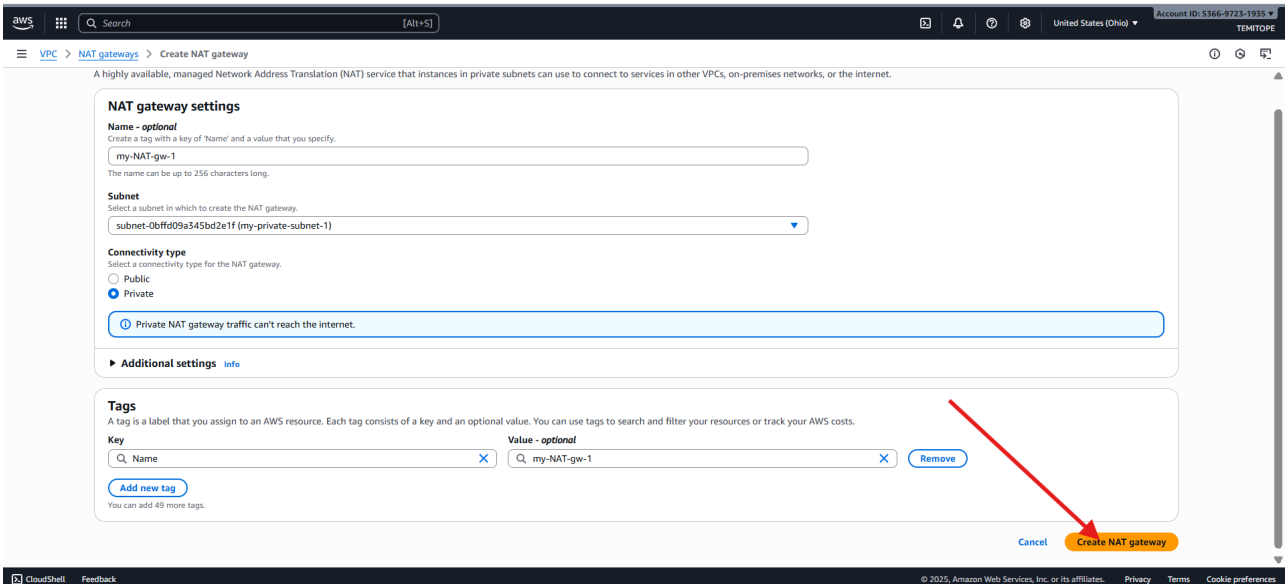
2. Click **NAT Gateways**.



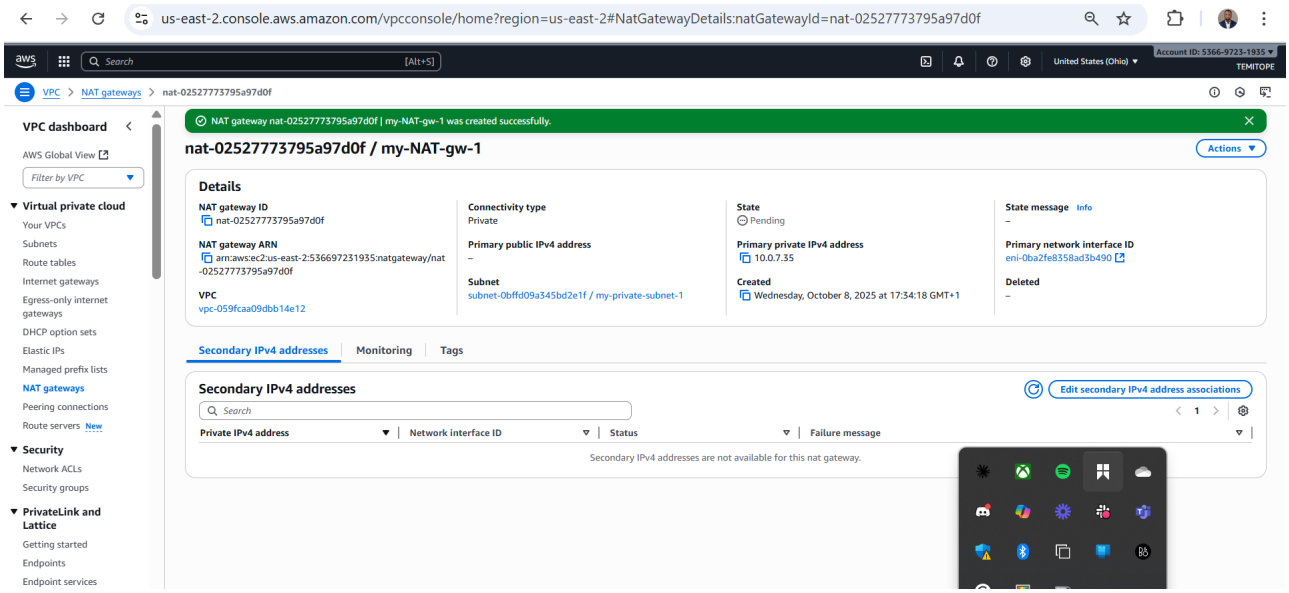
3. Click **Create NAT Gateway**.



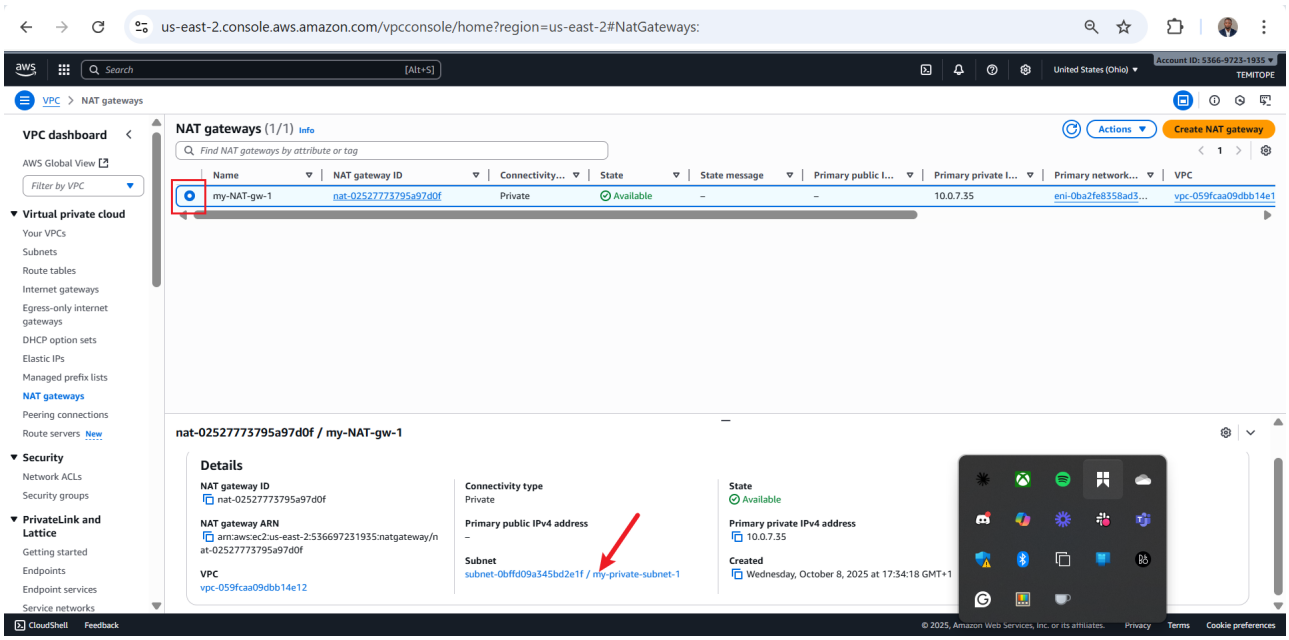
4. Name the NAT Gateway, select the private subnet, set connectivity type to **Private**, and click **Create NAT Gateway**.



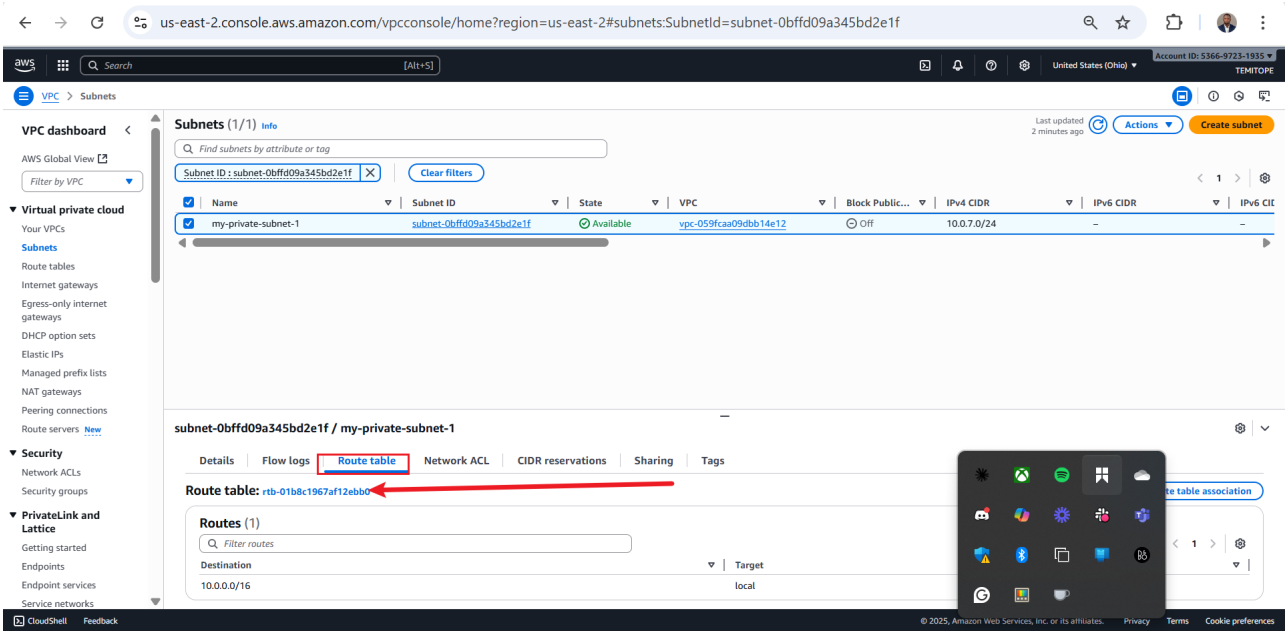
5. NAT Gateway created successfully.



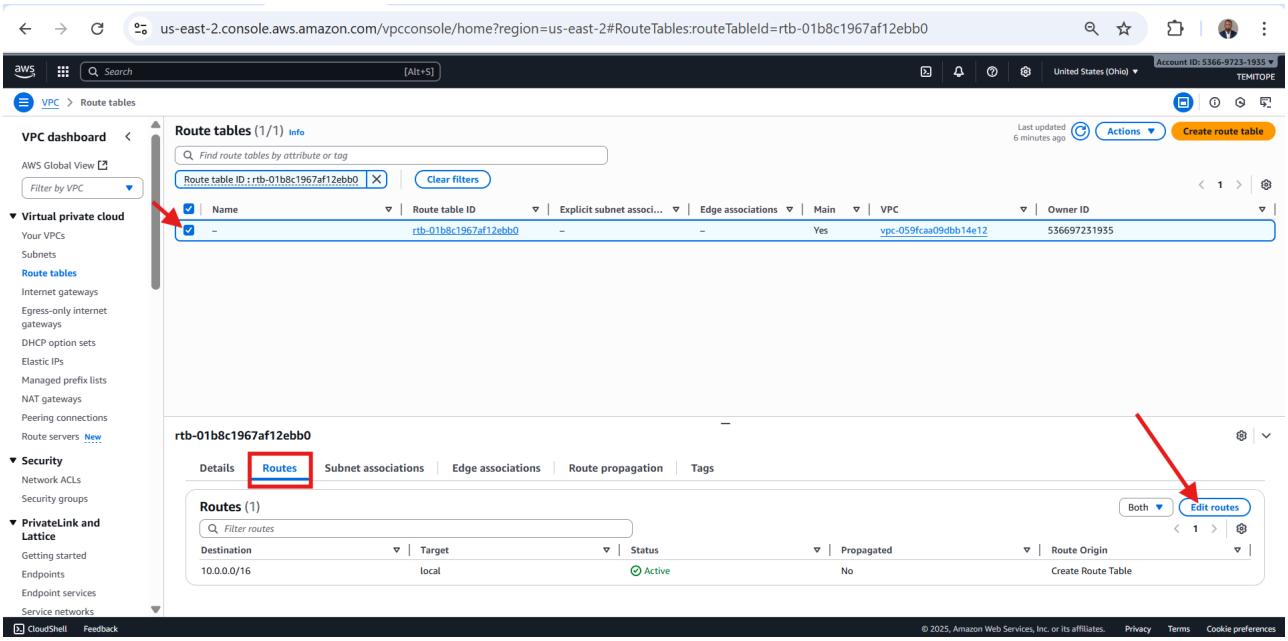
6. Select the NAT Gateway, locate the subnet ID in the **Details** tab, and click it.



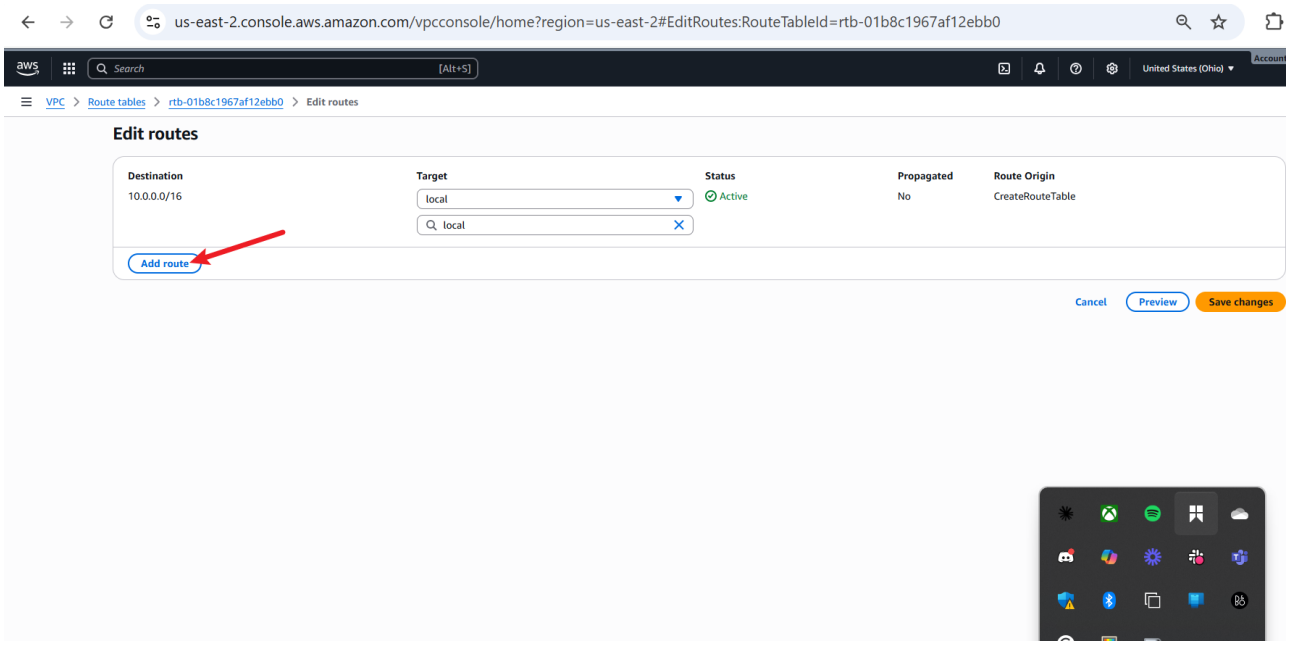
7. Navigate to the **Route Table** section and click the route table ID.



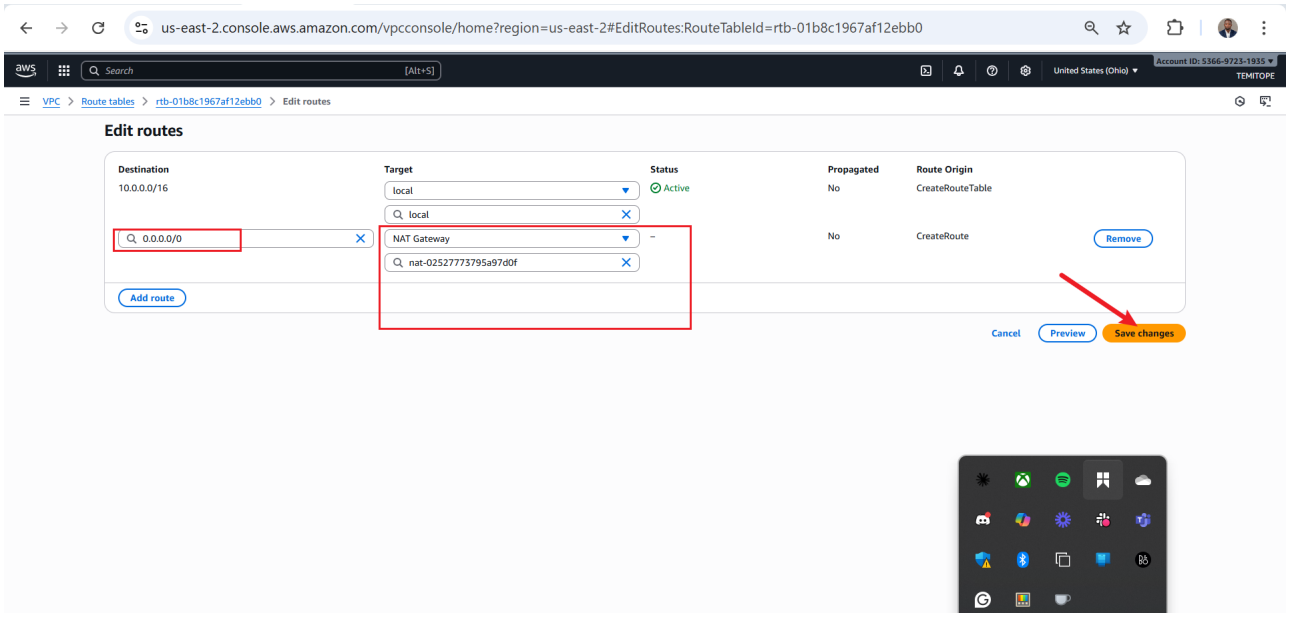
8. Click **Routes**, then **Edit routes**.



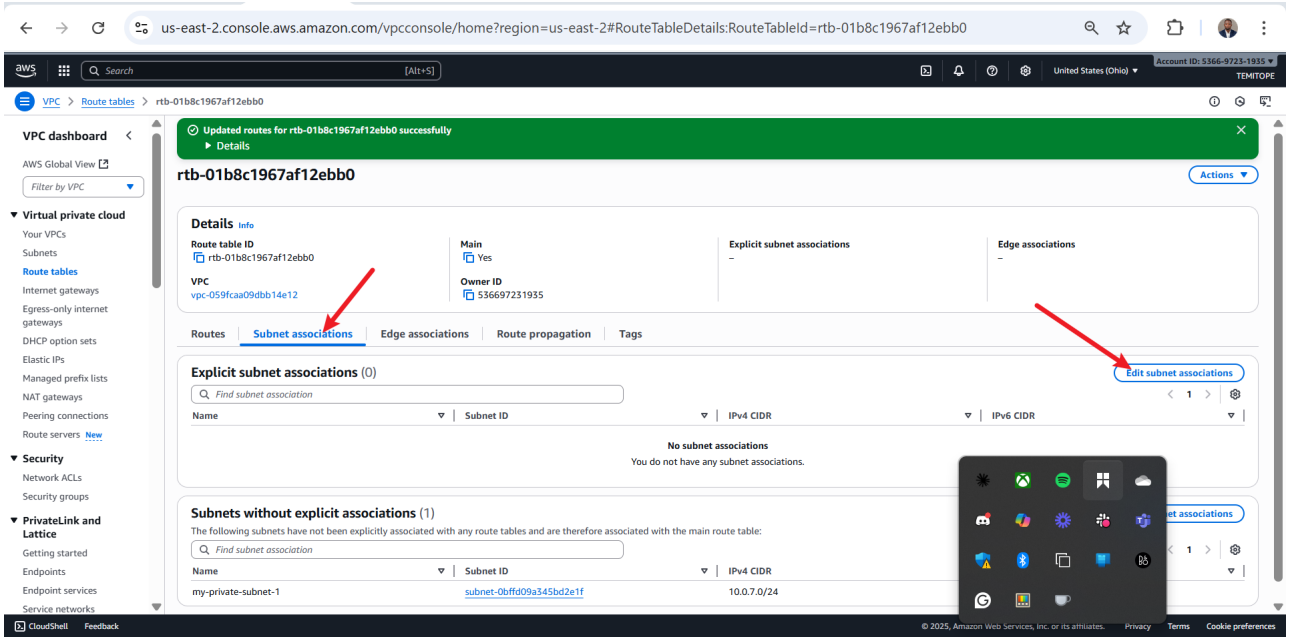
9. Click **Add route**.



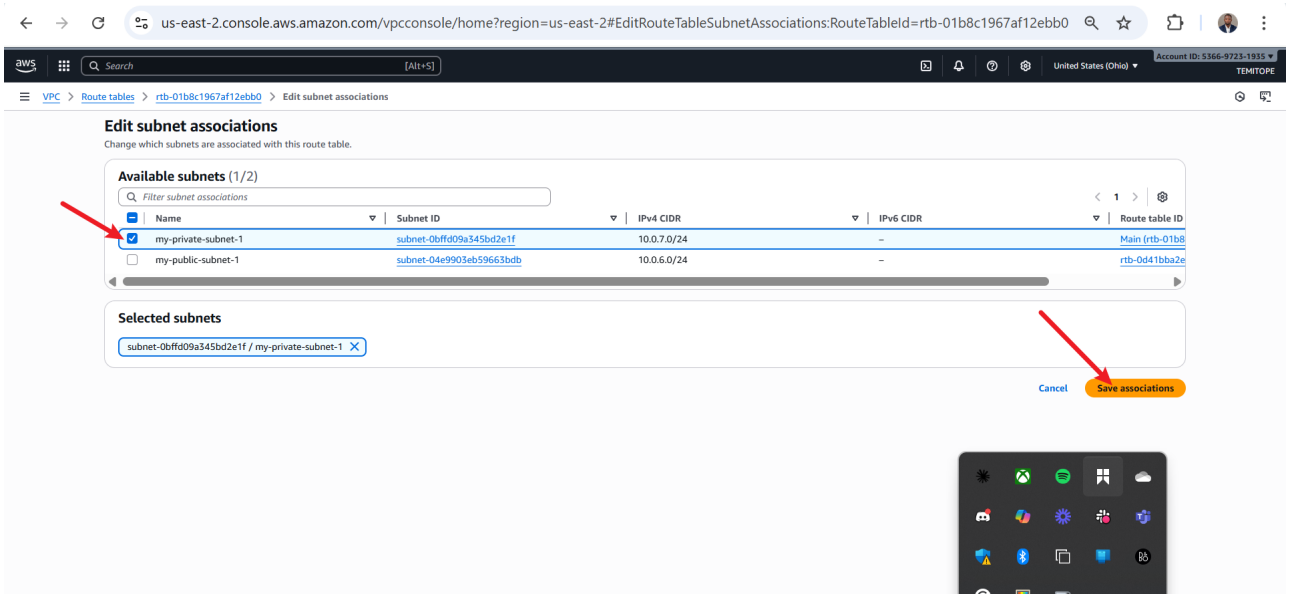
10. Set **Destination** to `0.0.0.0/0`, **Target** to **NAT Gateway**, select the created NAT Gateway, and save changes.



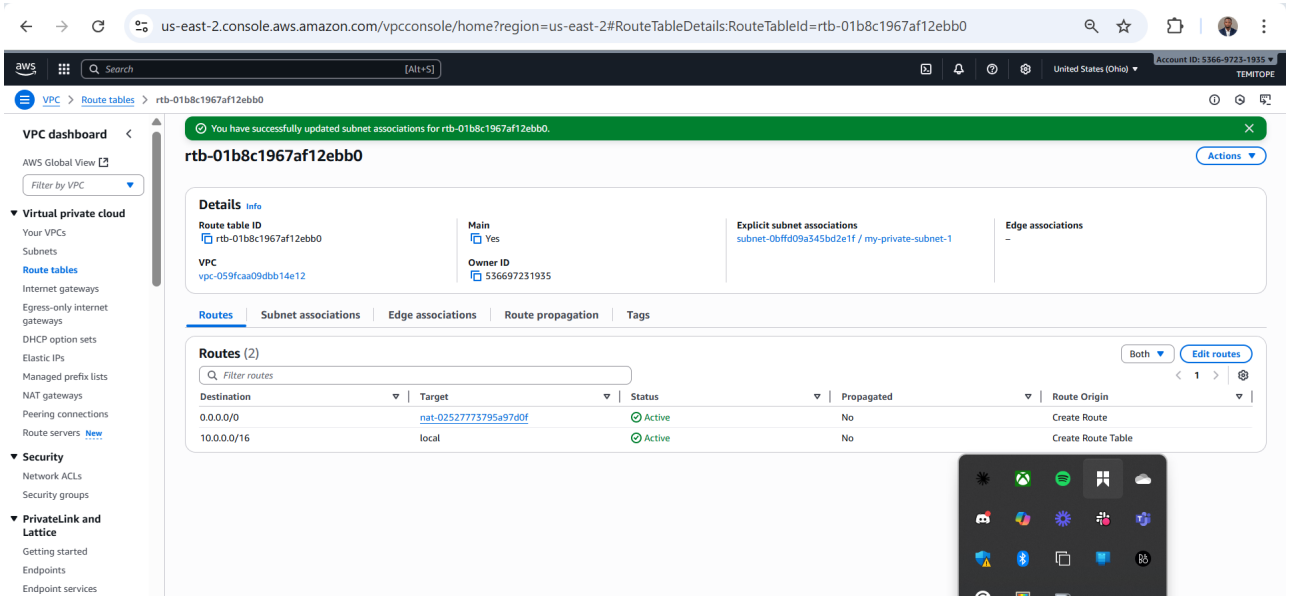
11. Route table updated successfully, click **Subnet associations**, then **Edit subnet associations**.



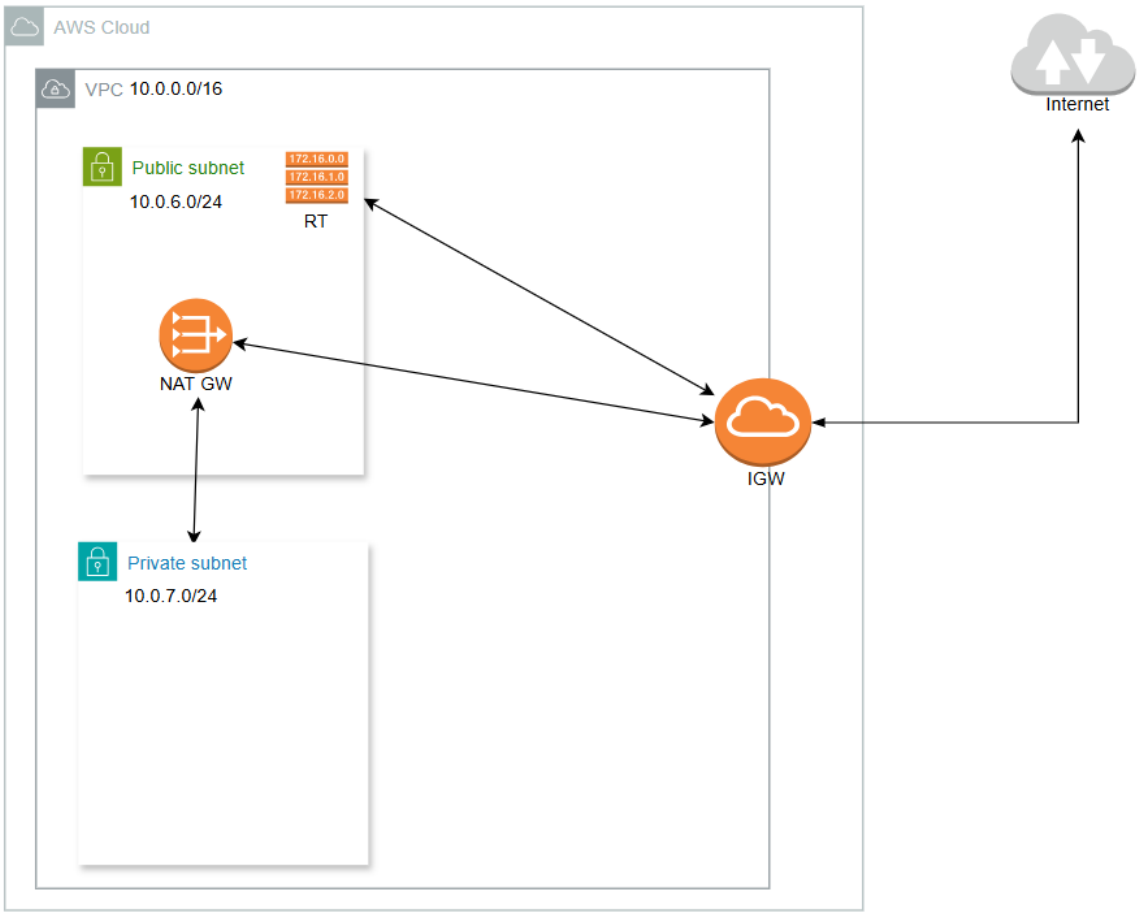
12. Select the private subnet and click **Save associations**.



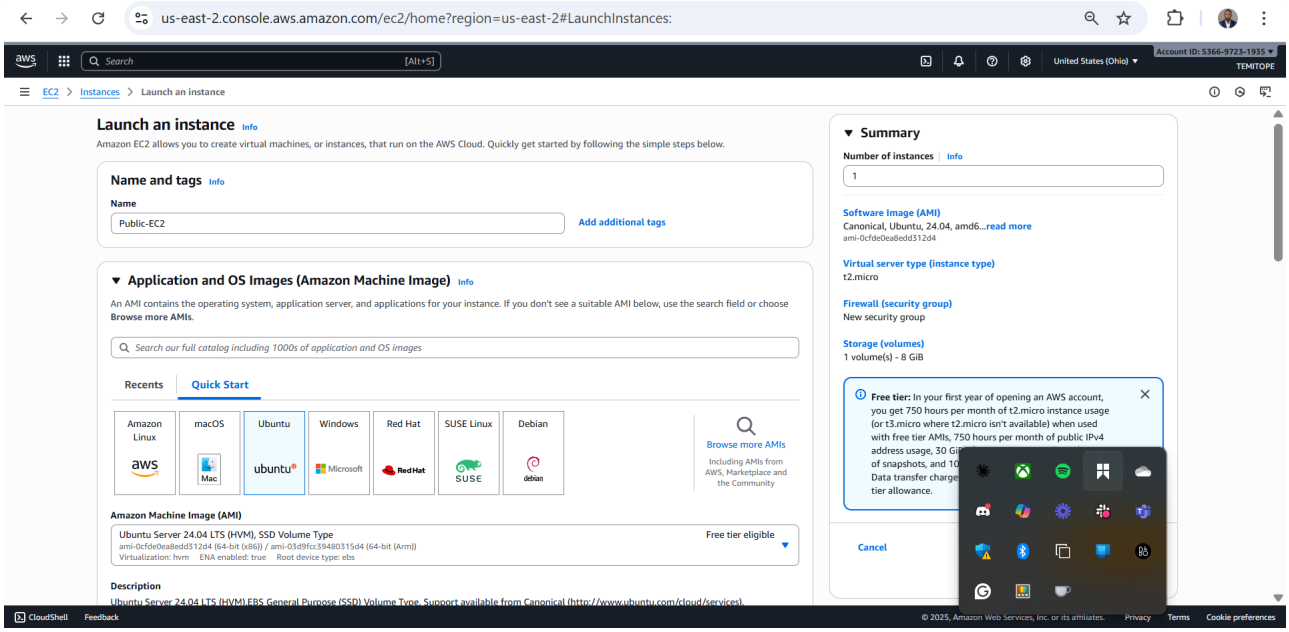
13. Subnet successfully attached to the route table.



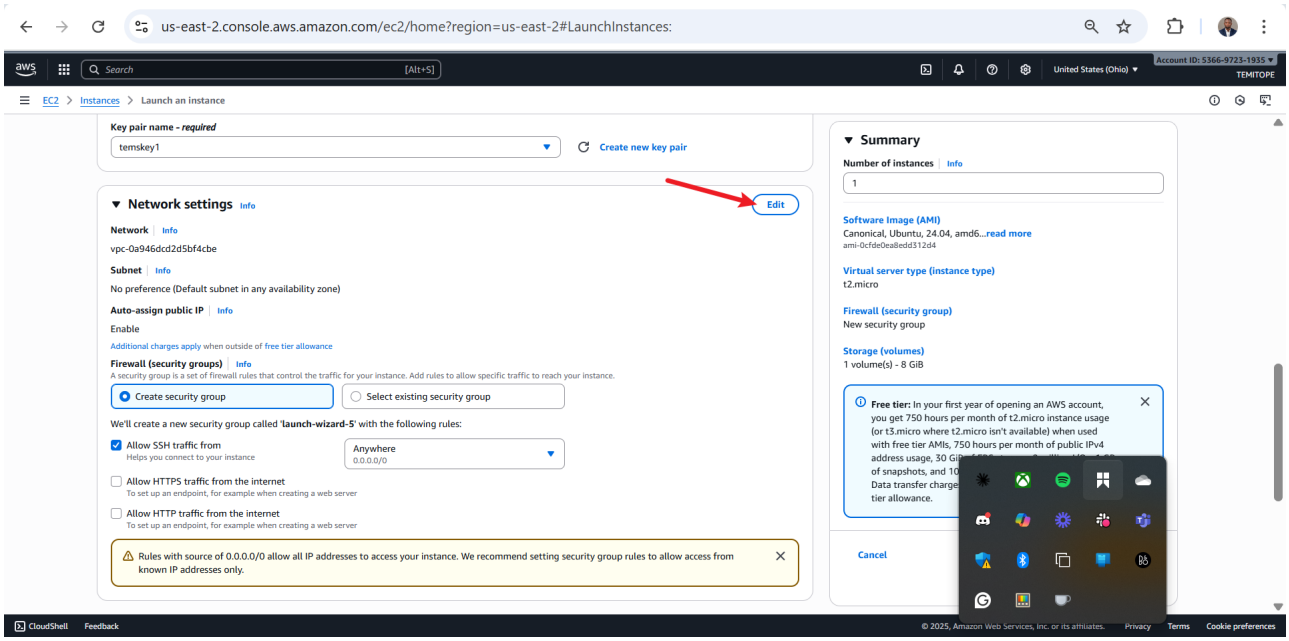
14. View the current VPC architecture.



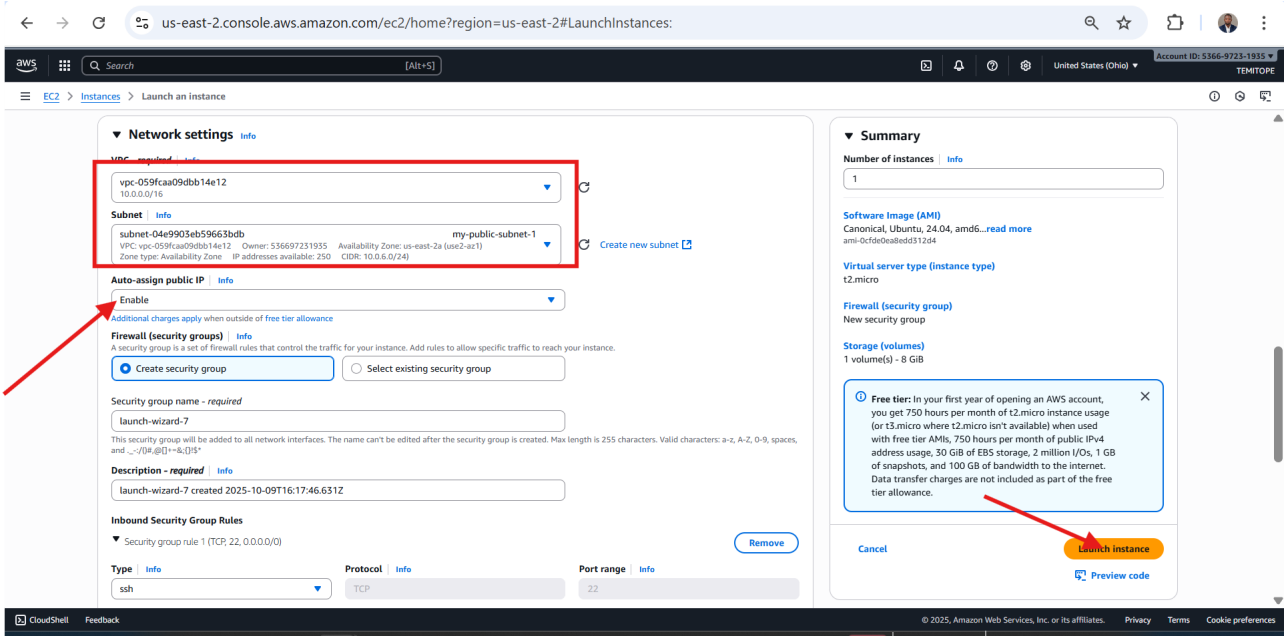
15. Create an Ubuntu EC2 instance named `Public-EC2`.



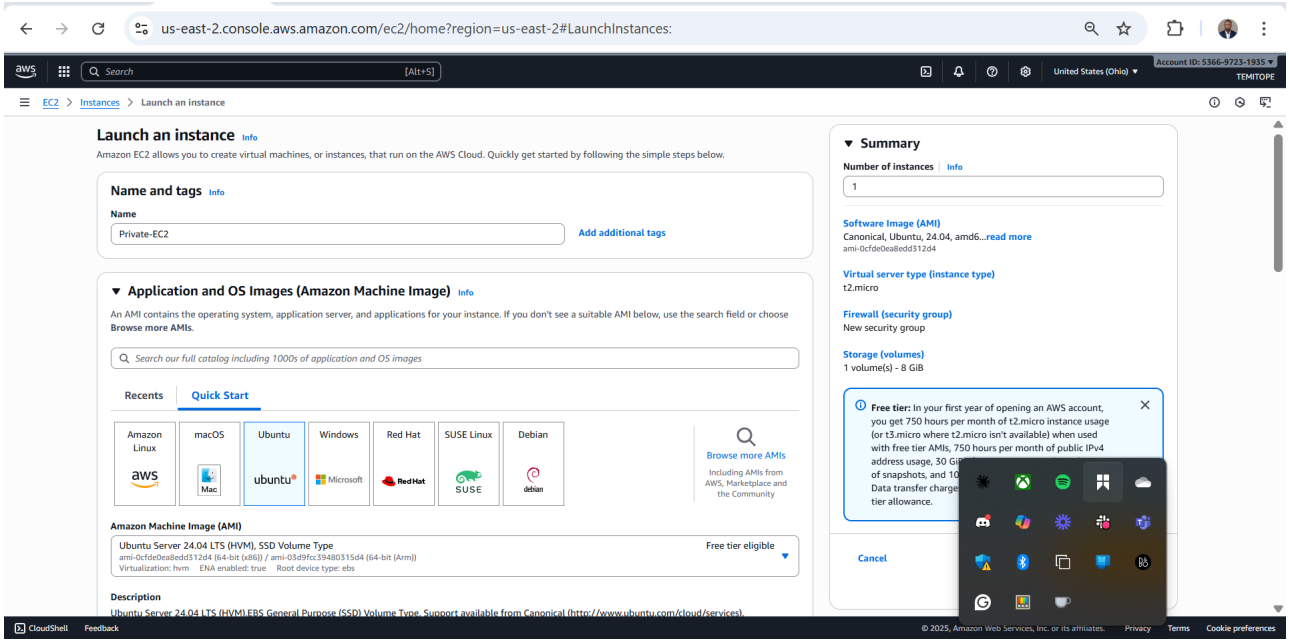
16. In network settings, click **Edit**.



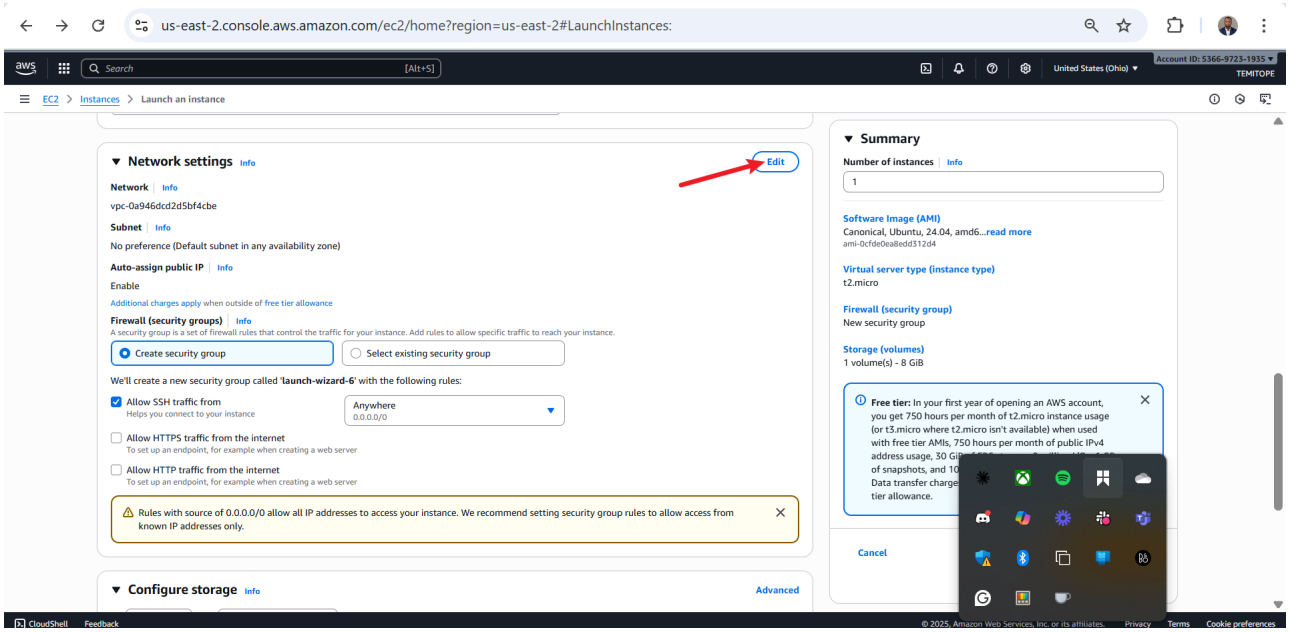
17. Set VPC to 10.0.0.0/16, subnet to the public subnet, enable **Auto-assign public IP**, and launch instance.



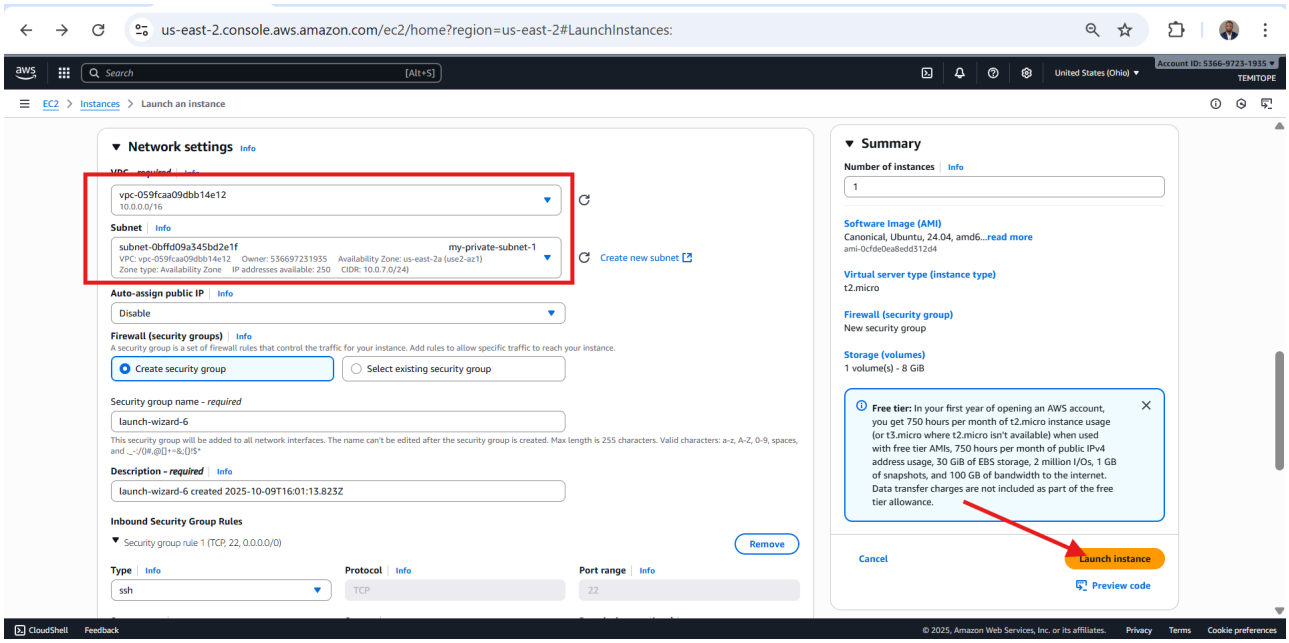
18. Create an Ubuntu EC2 instance named `Private-EC2`.



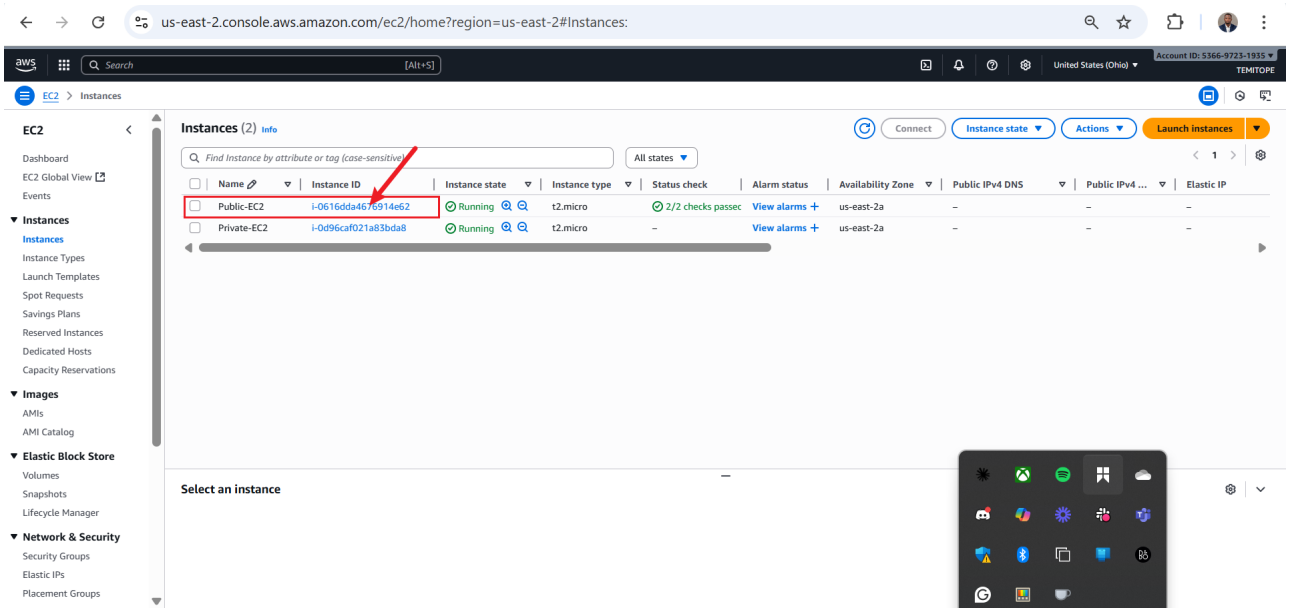
19. In network settings, click **Edit**.



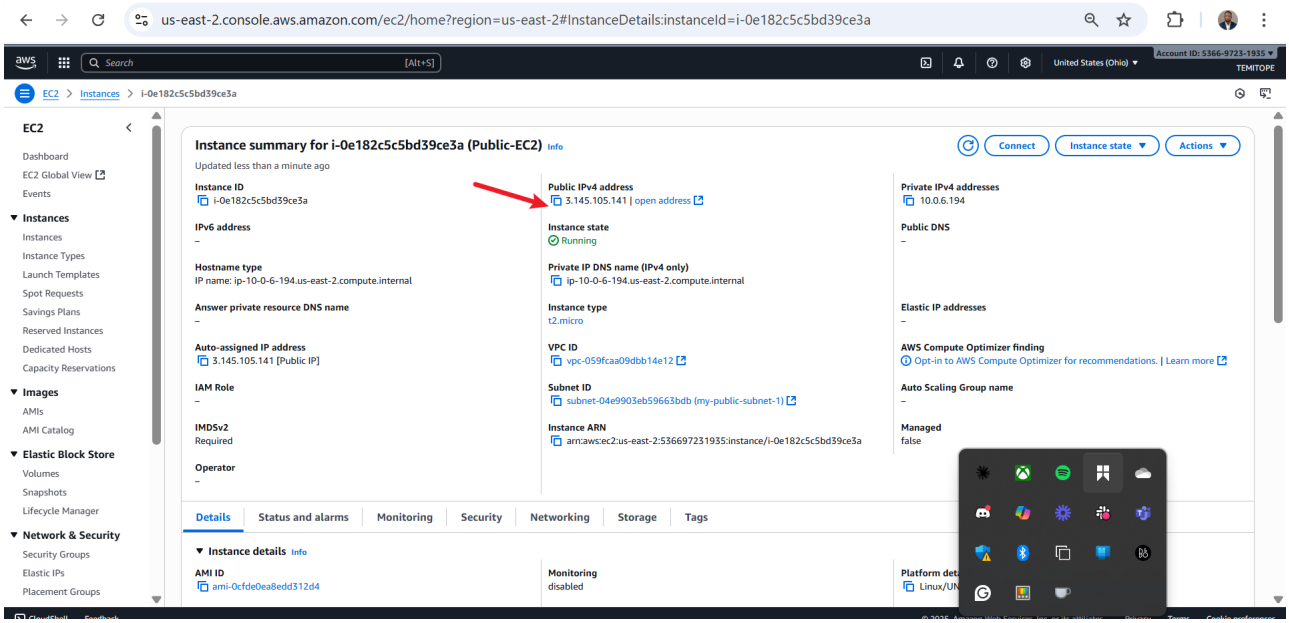
20. Set VPC to 10.0.0.0/16, subnet to the private subnet, and launch instance.



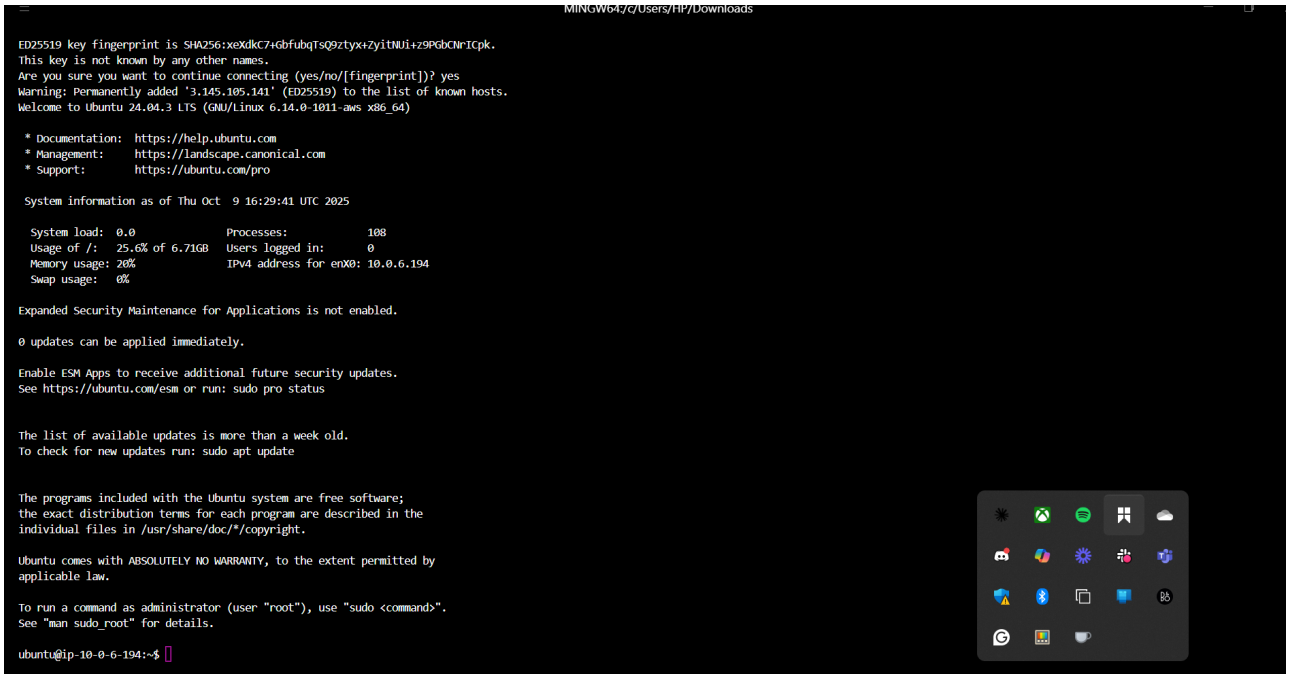
21. Both EC2 instances created, click the `Public-EC2` ID.



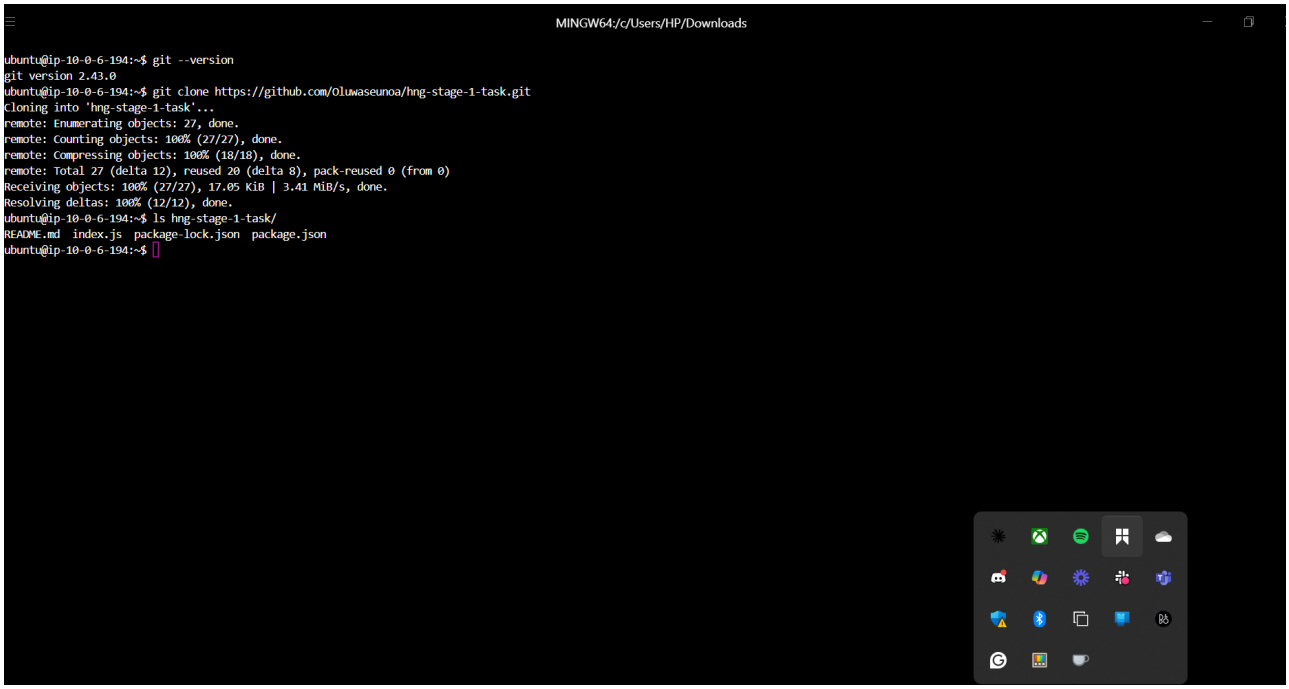
22. Copy the public IP for remote connection.



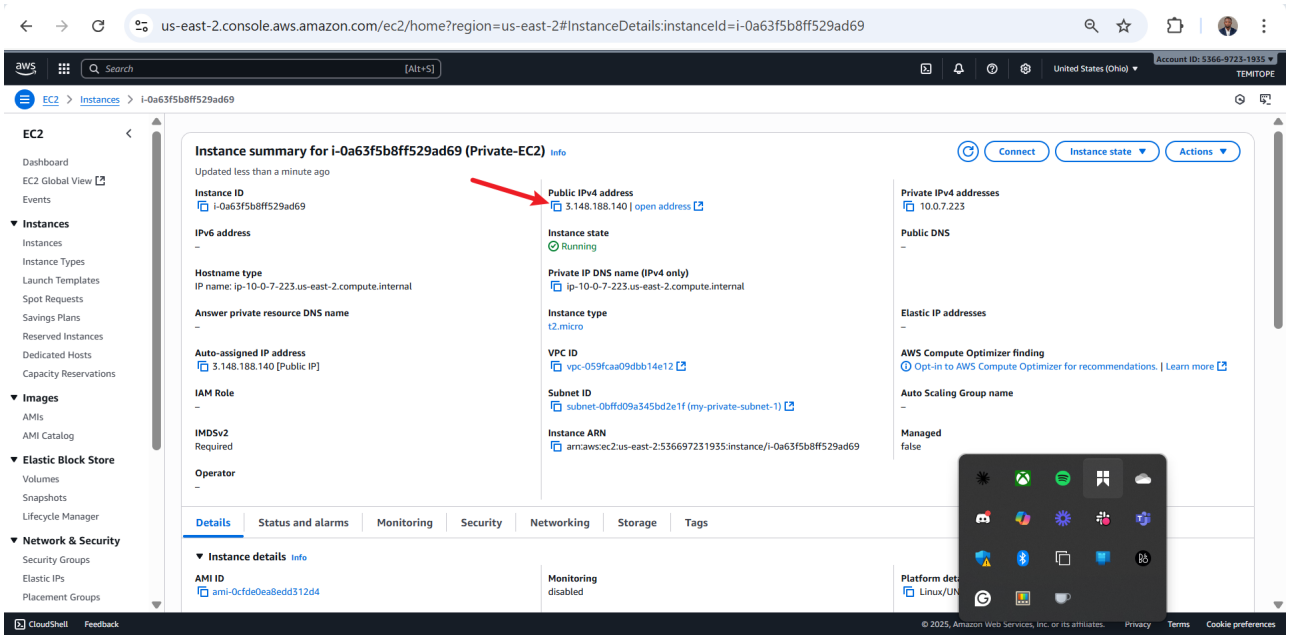
23. Connect via SSH.



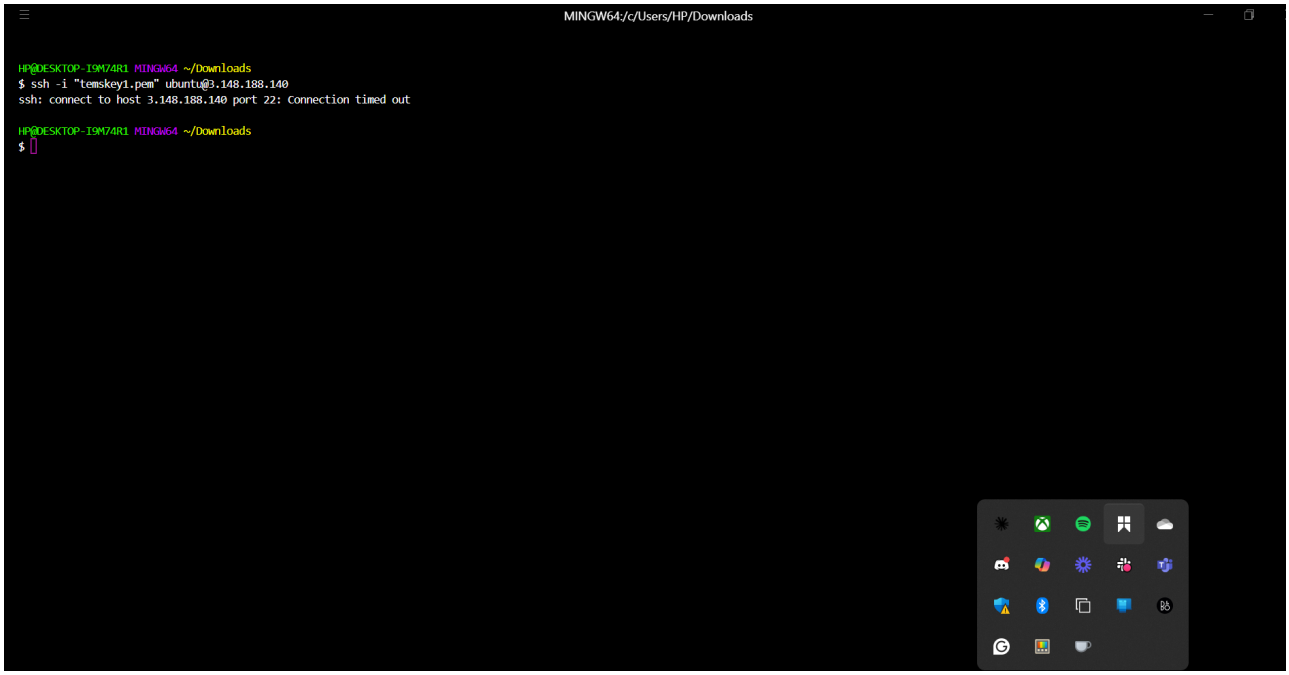
24. Test outbound traffic with `git clone`.



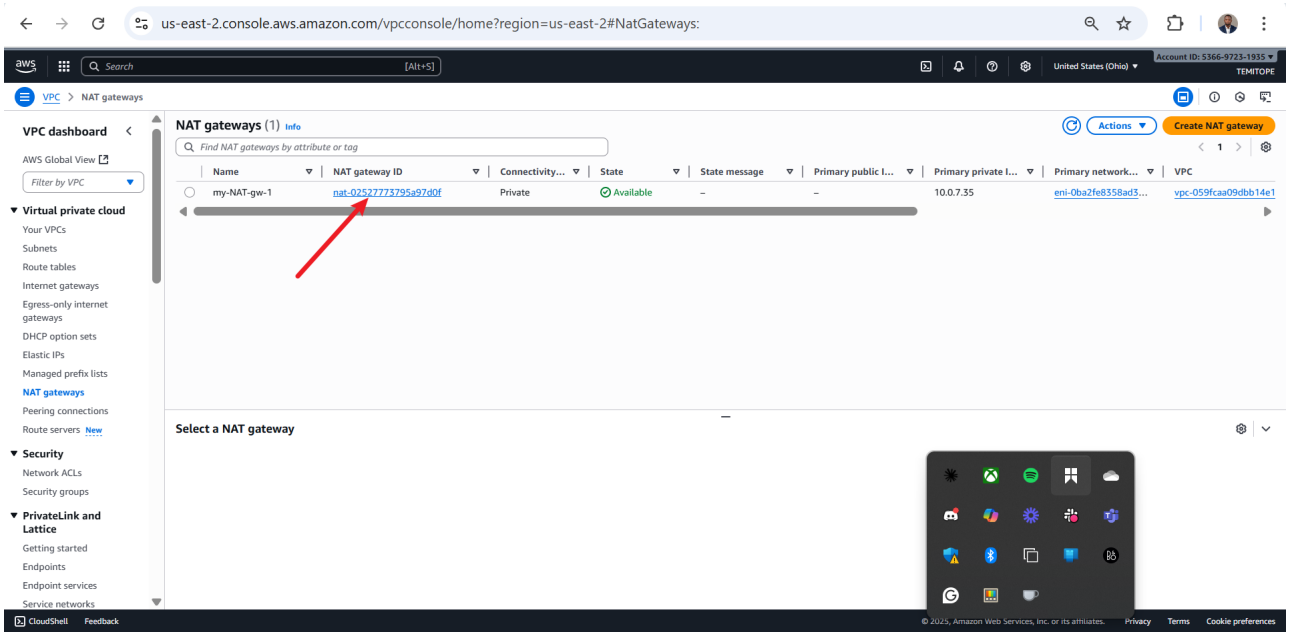
25. Copy the public IP of `Private-EC2` for remote connection (note: connection should fail).



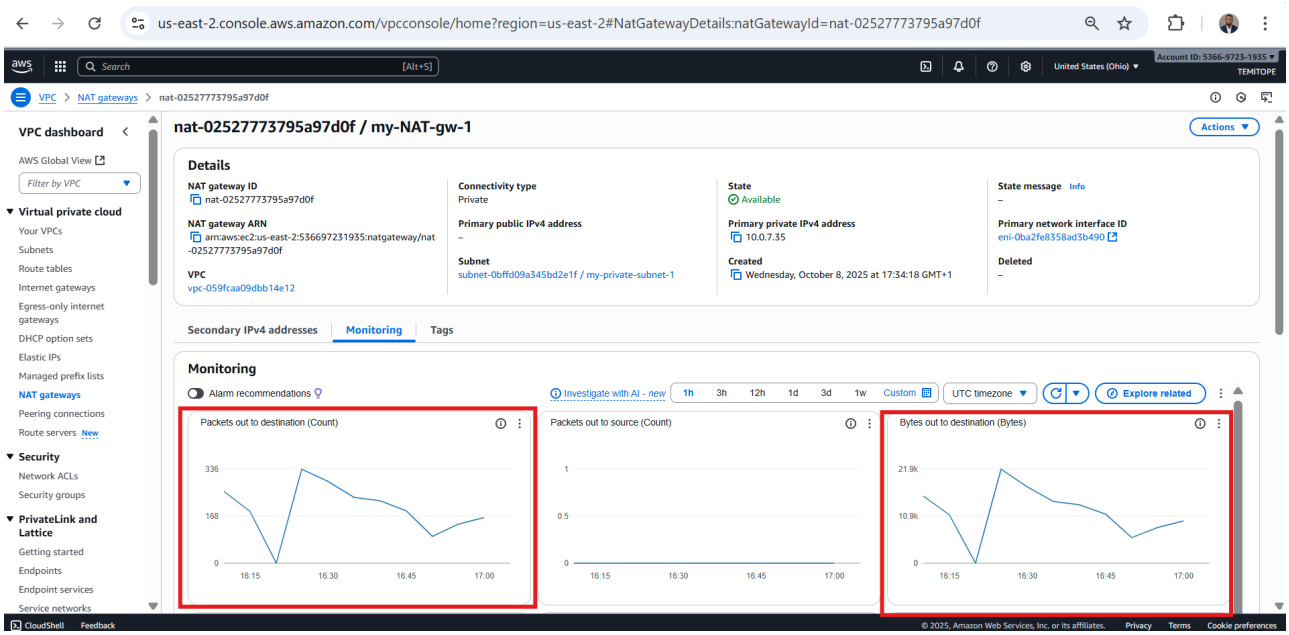
26. Unable to connect to `Private-EC2` over the internet.



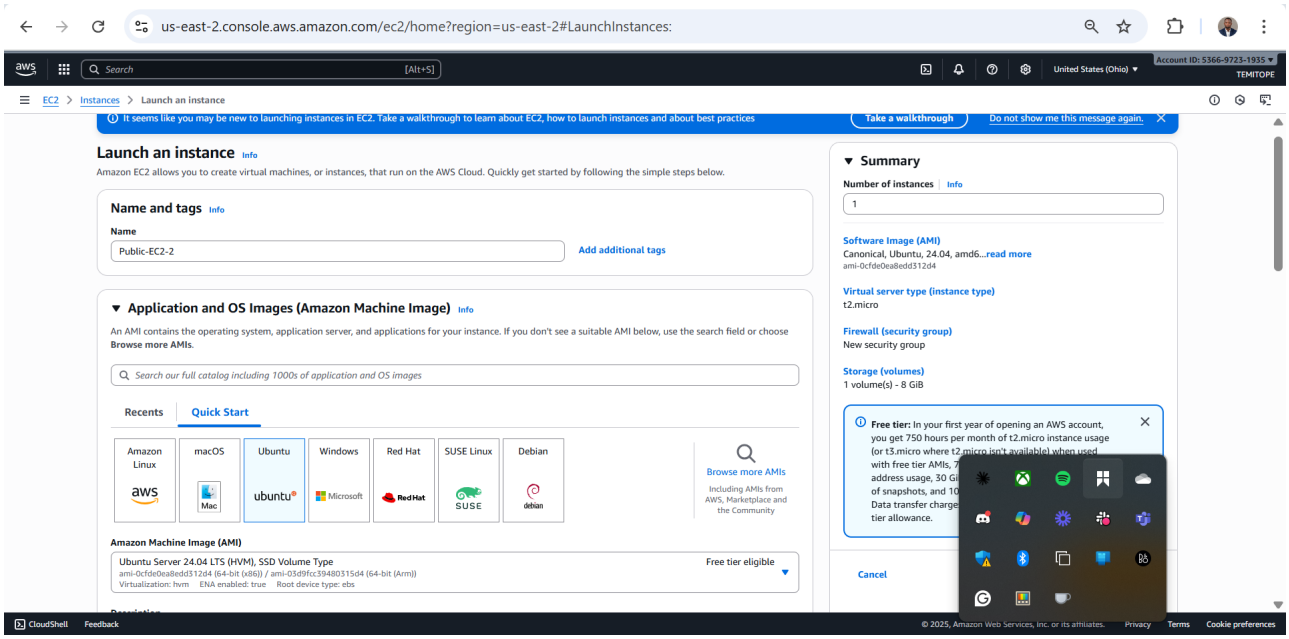
27. Verify NAT Gateway functionality by navigating to **NAT Gateways** and selecting the created NAT Gateway.



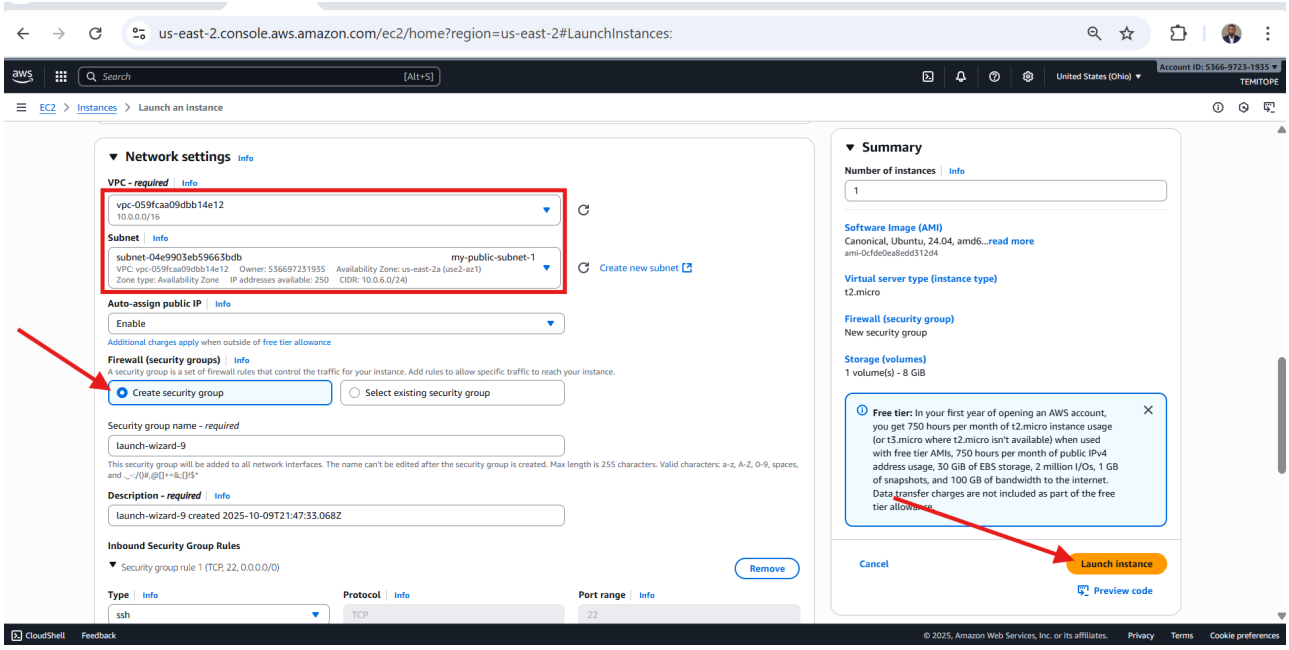
28. Observe outbound traffic data in **Packet Out to Destination** and **Byte Out to Destination**.



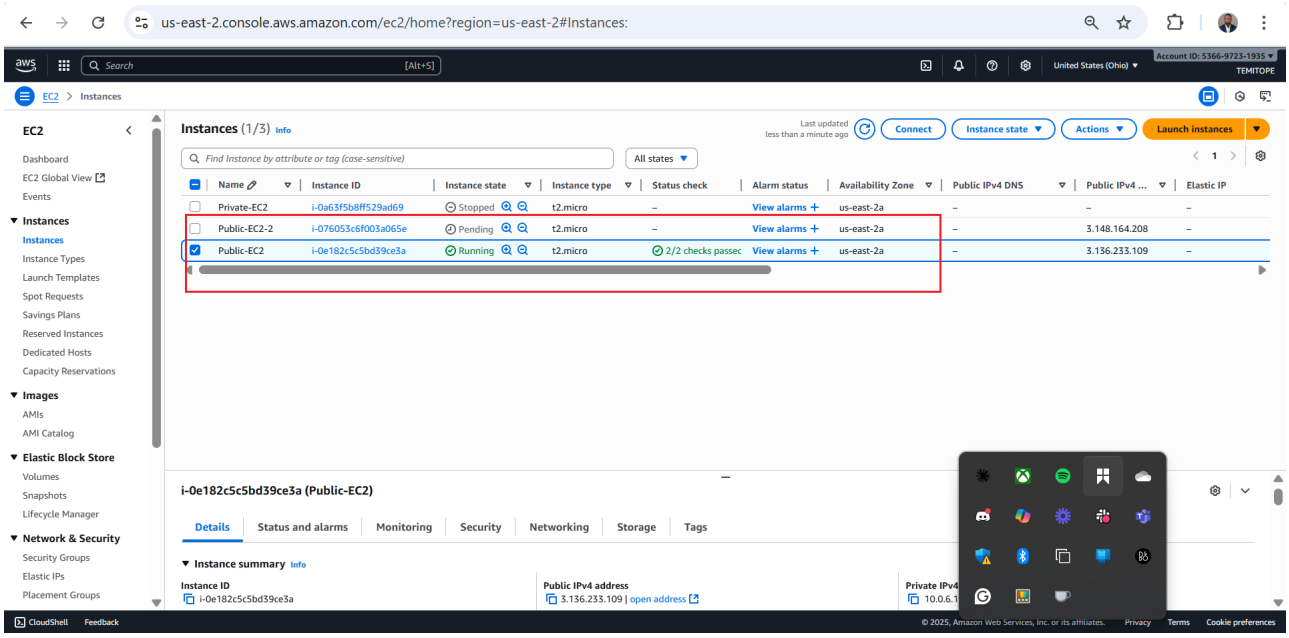
29. Create another EC2 instance to test connectivity within the same subnet.



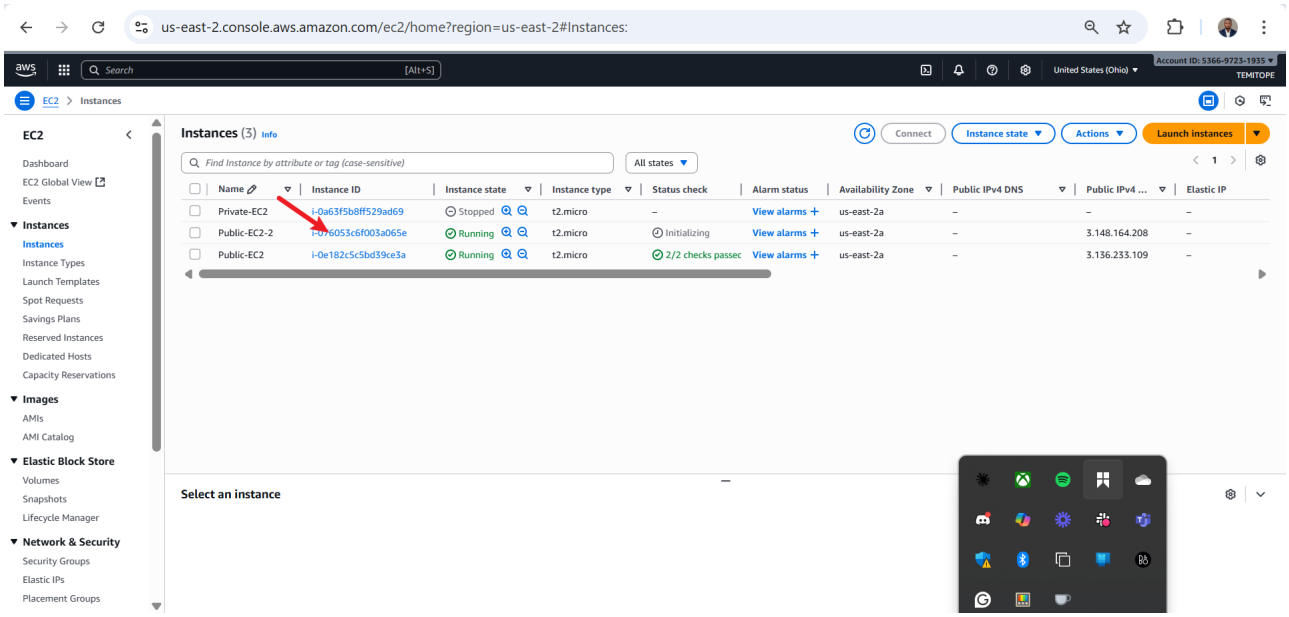
30. Set VPC to 10.0.0.0/16, subnet to the public subnet, and launch instance.



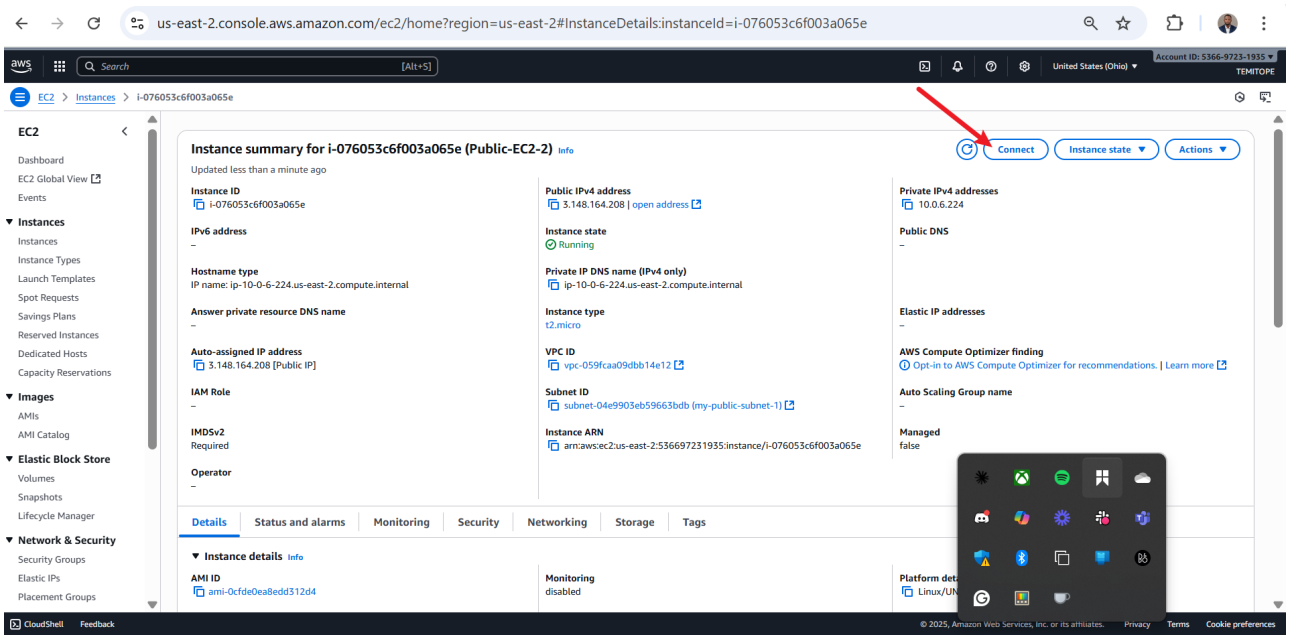
31. Two EC2 instances now in the public subnet.



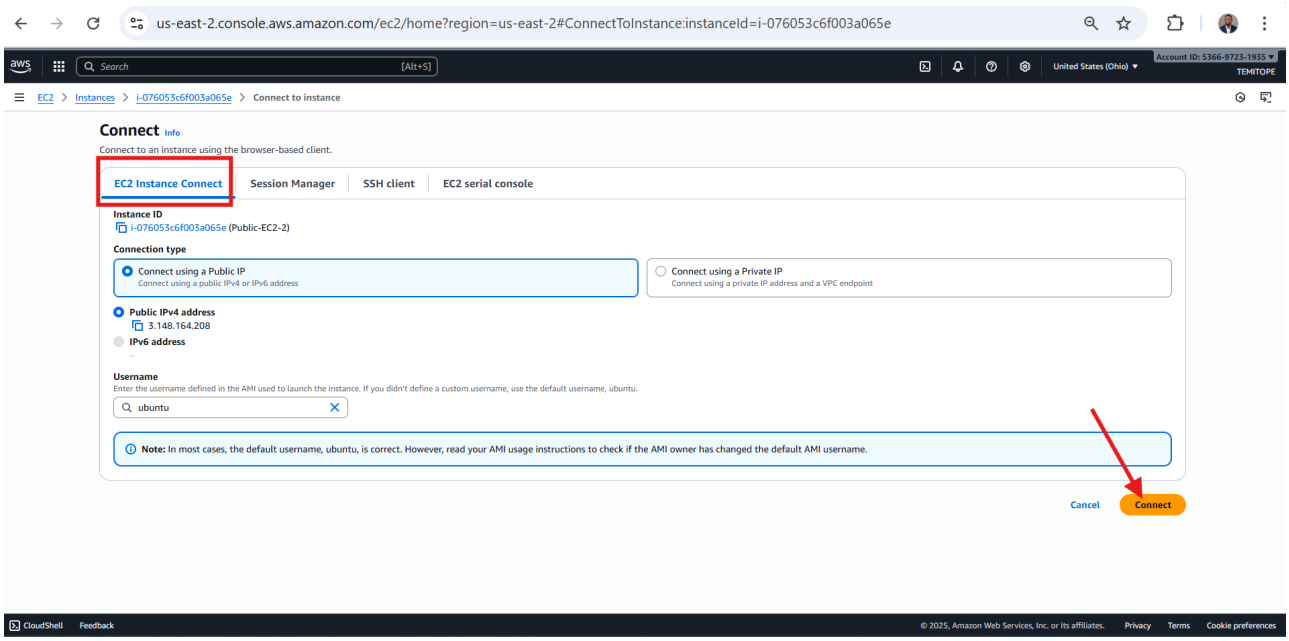
32. Click the new `Public-EC2`.



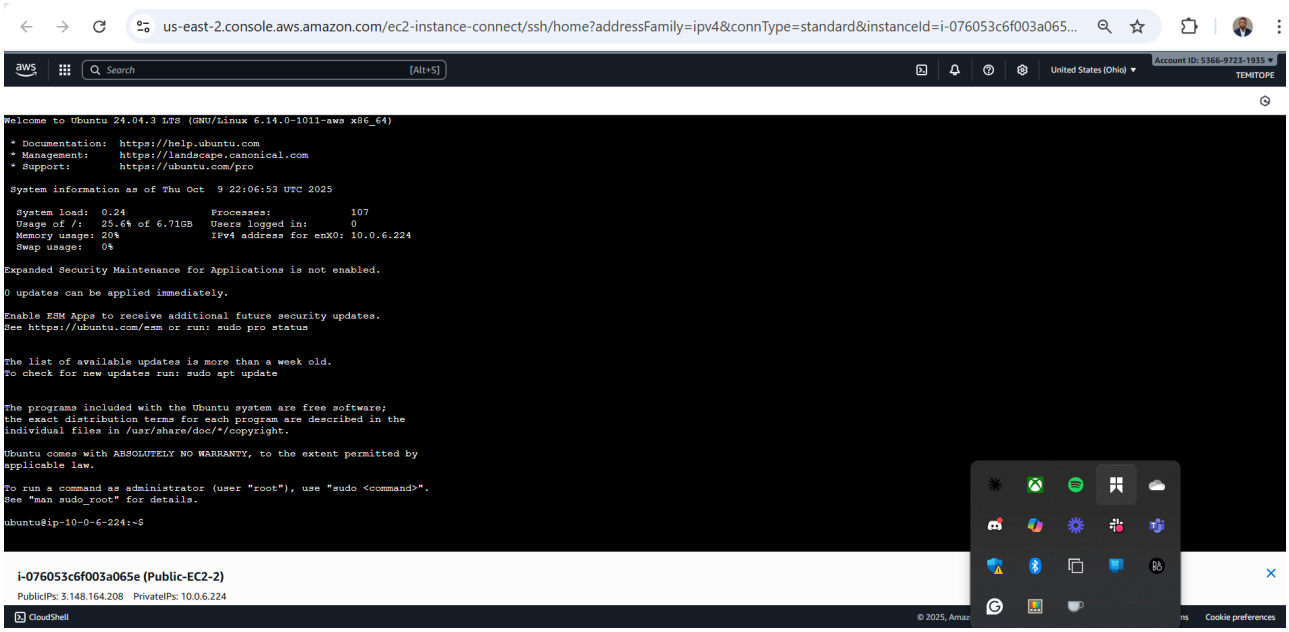
33. Click **Connect**.



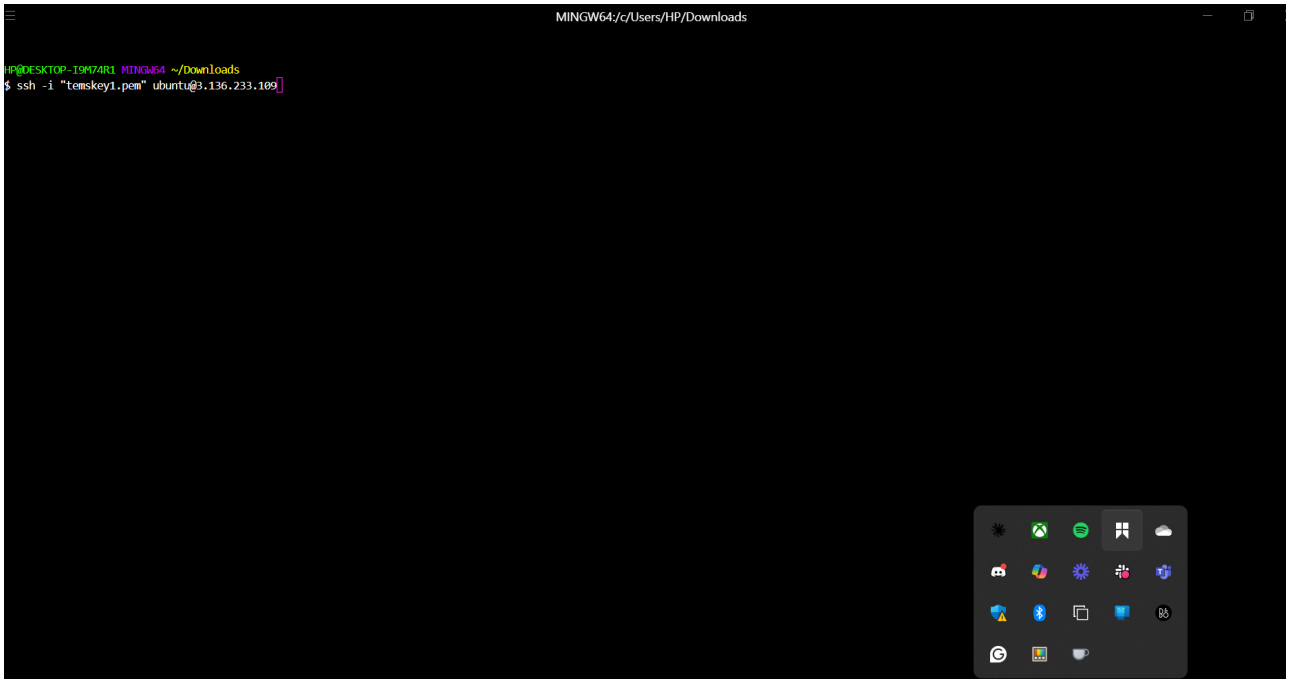
34. Select **EC2 Instance Connect** tab and click **Connect**.



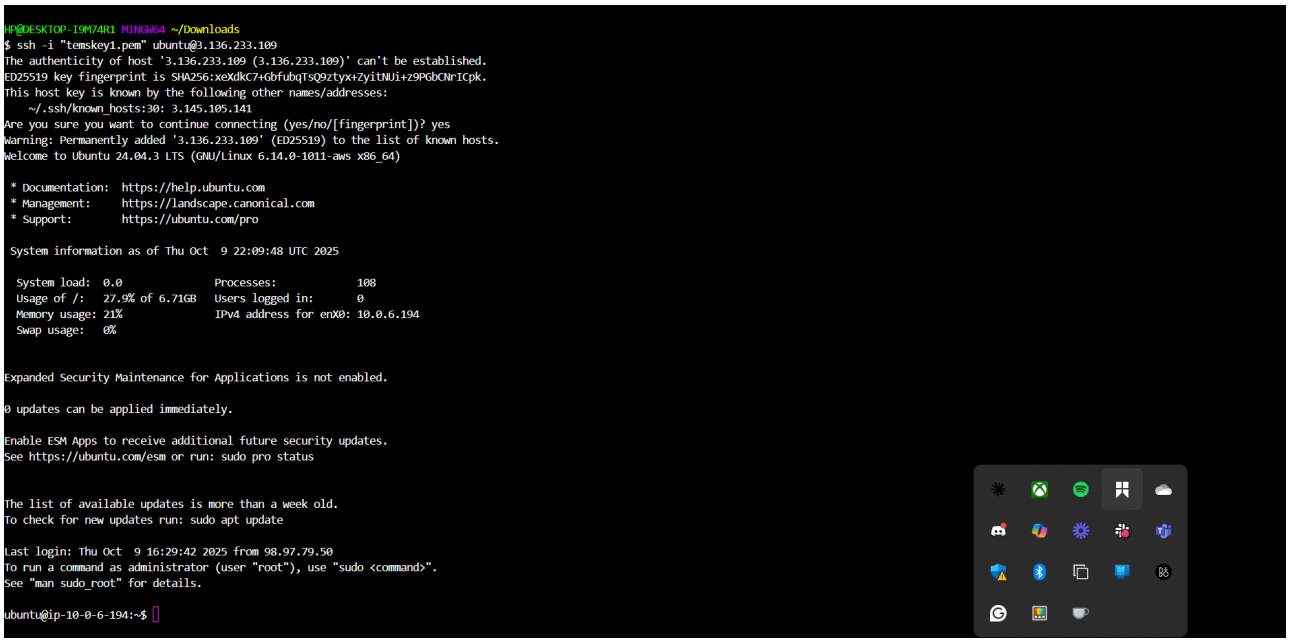
35. Successfully connected via browser.



36. Connect to the other EC2 in the public subnet using a terminal.

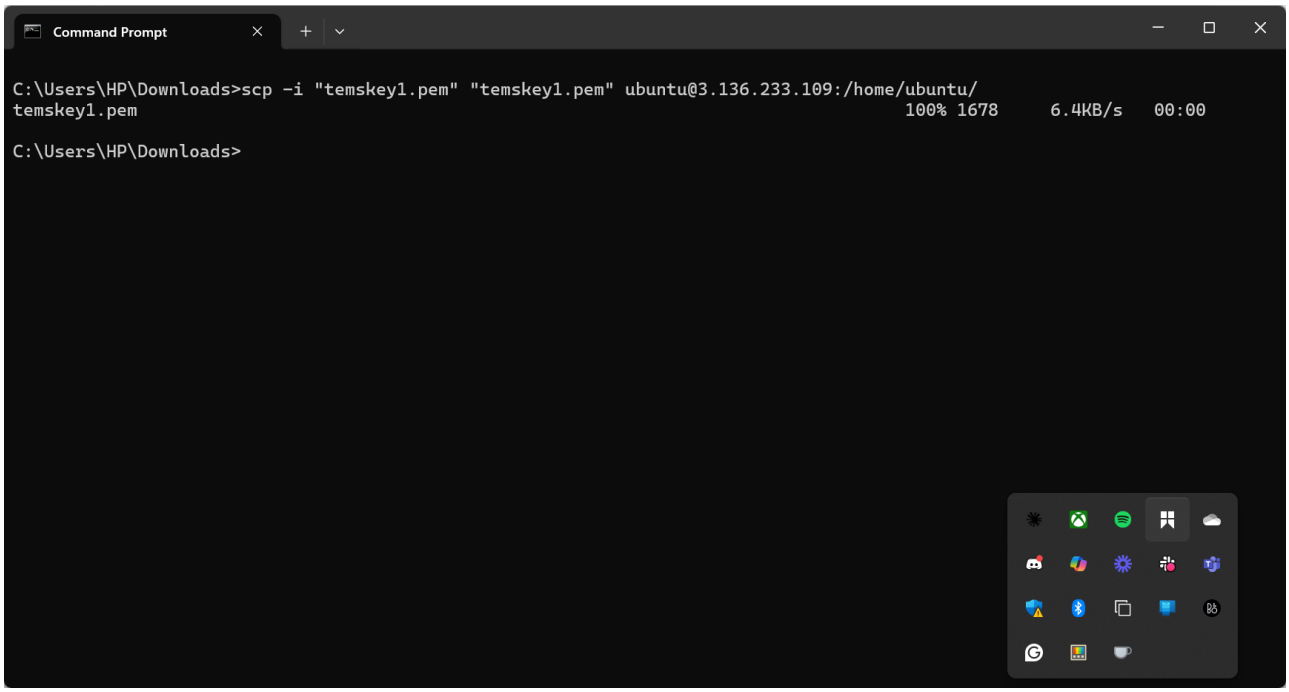


37. Successfully connect via terminal.



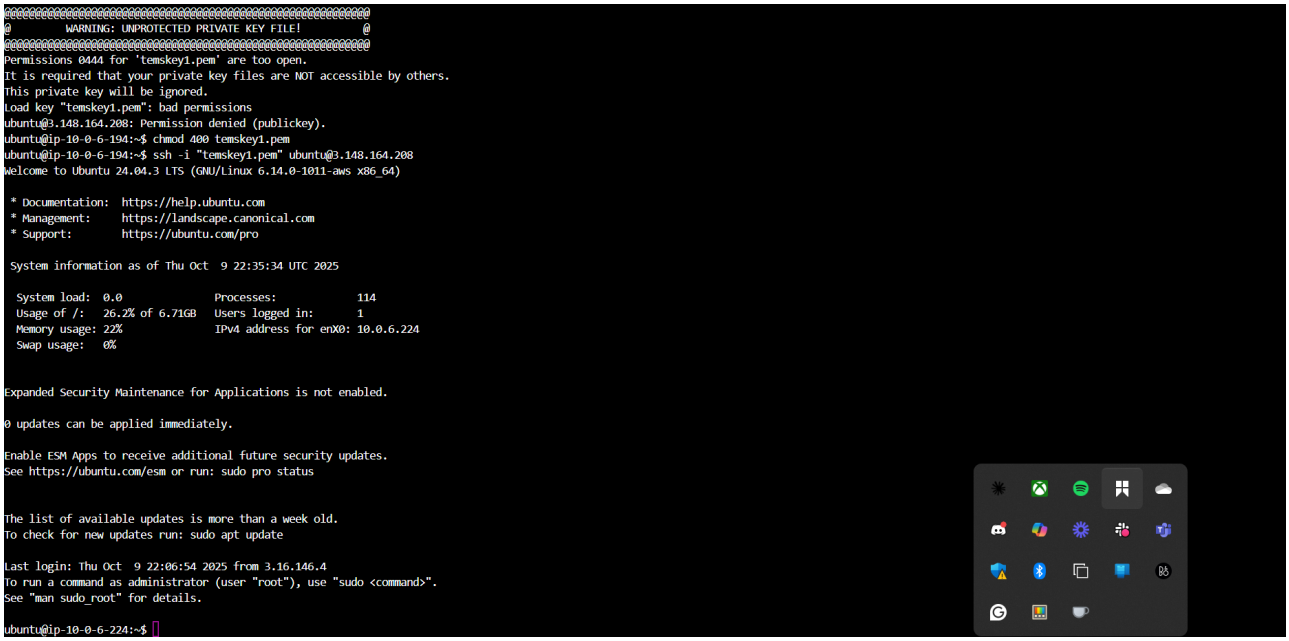
38. Ping the second EC2 to ensure reachability.

39. Move the key pair to EC2 using SCP to SSH into the second EC2.

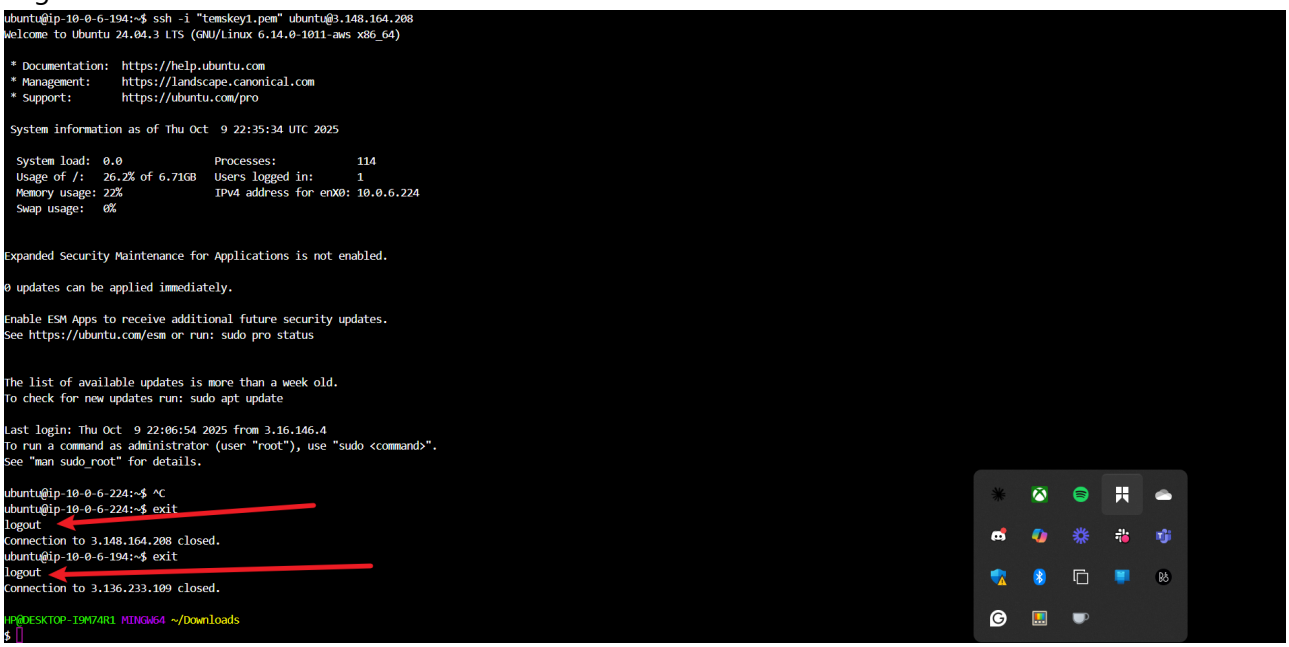


40. Confirm key pair copied and SSH into `Public-EC2-2` from `Public-EC2`.

41. Successfully SSH into `Public-EC2-2`.

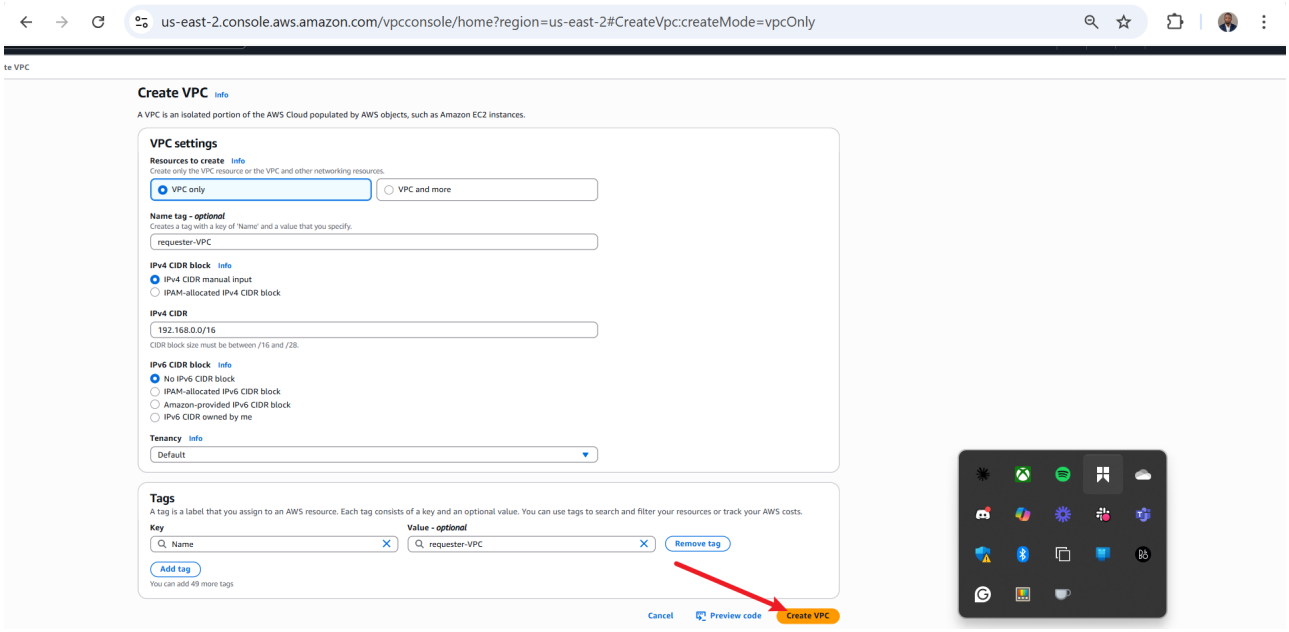


42. Log out of both instances.

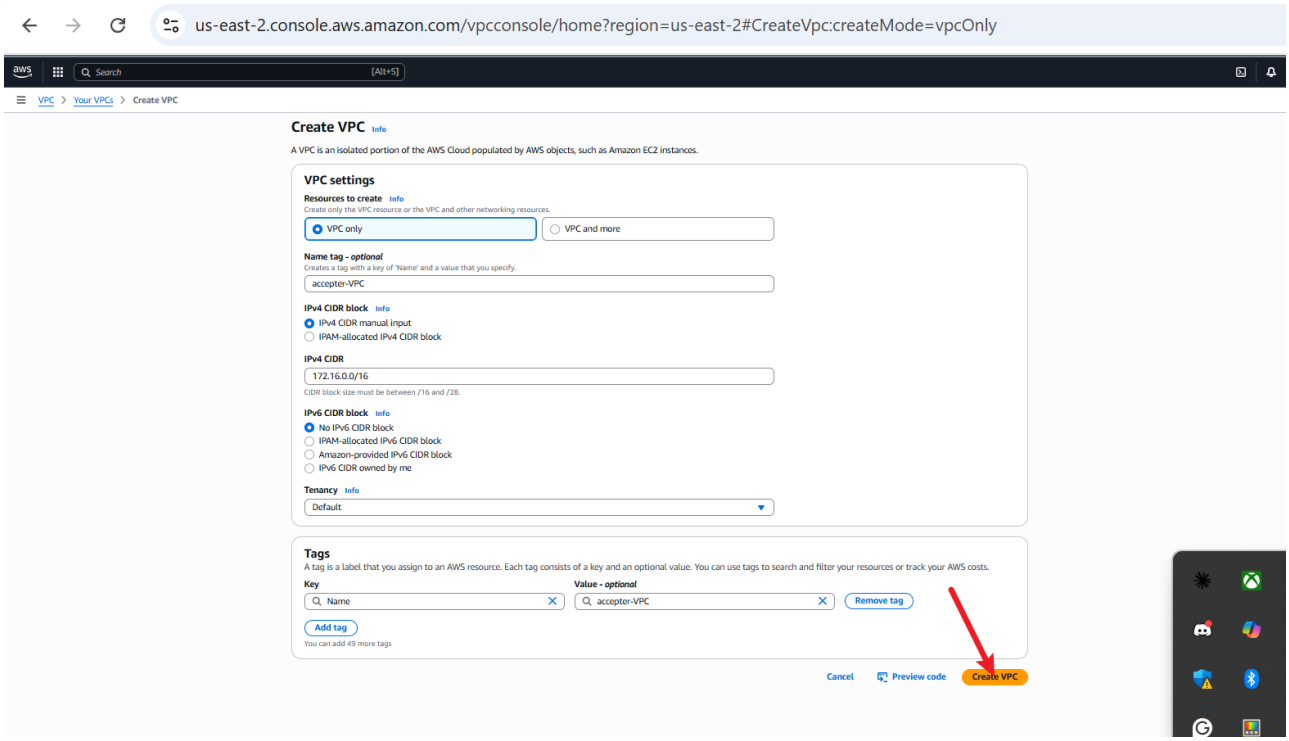


Part 6: Establishing VPC Peering Connections

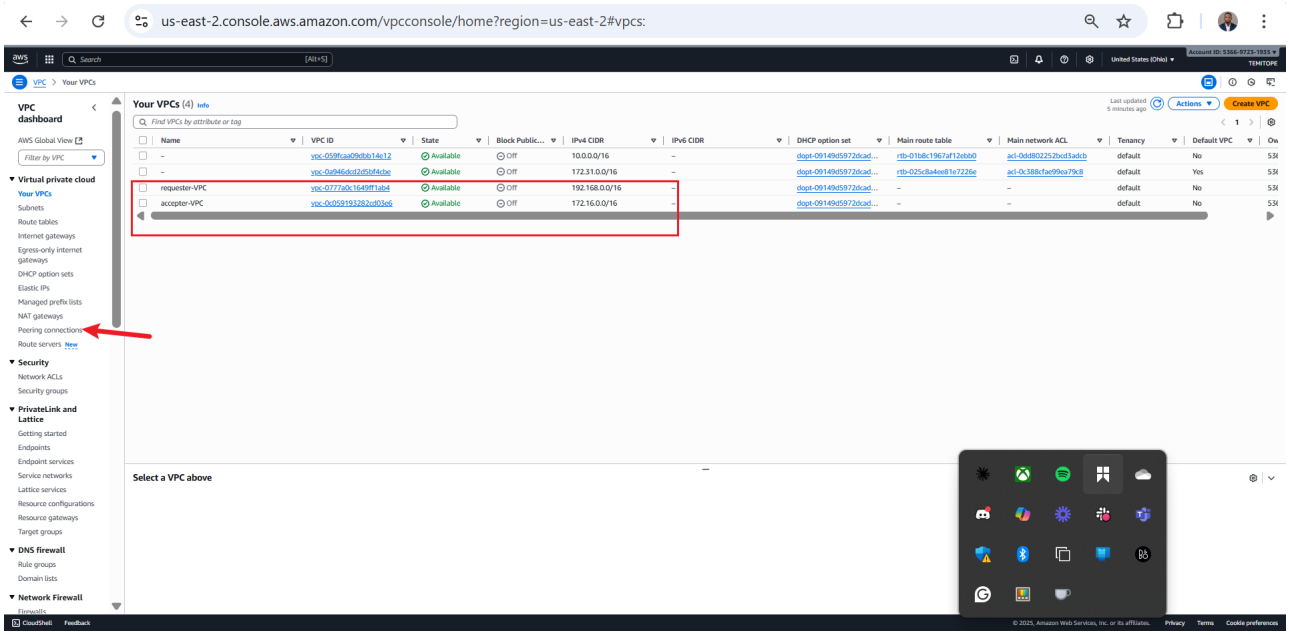
1. Create the requester VPC (192.168.0.0/16).



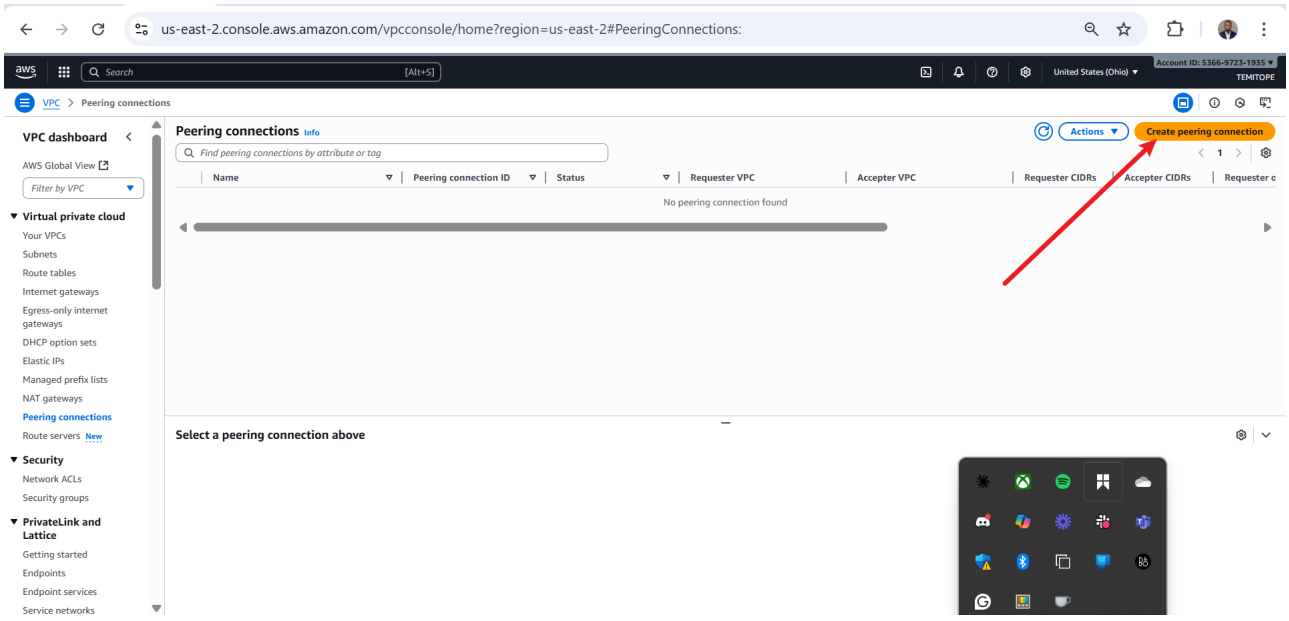
2. Create the accepter VPC (172.16.0.0/16).



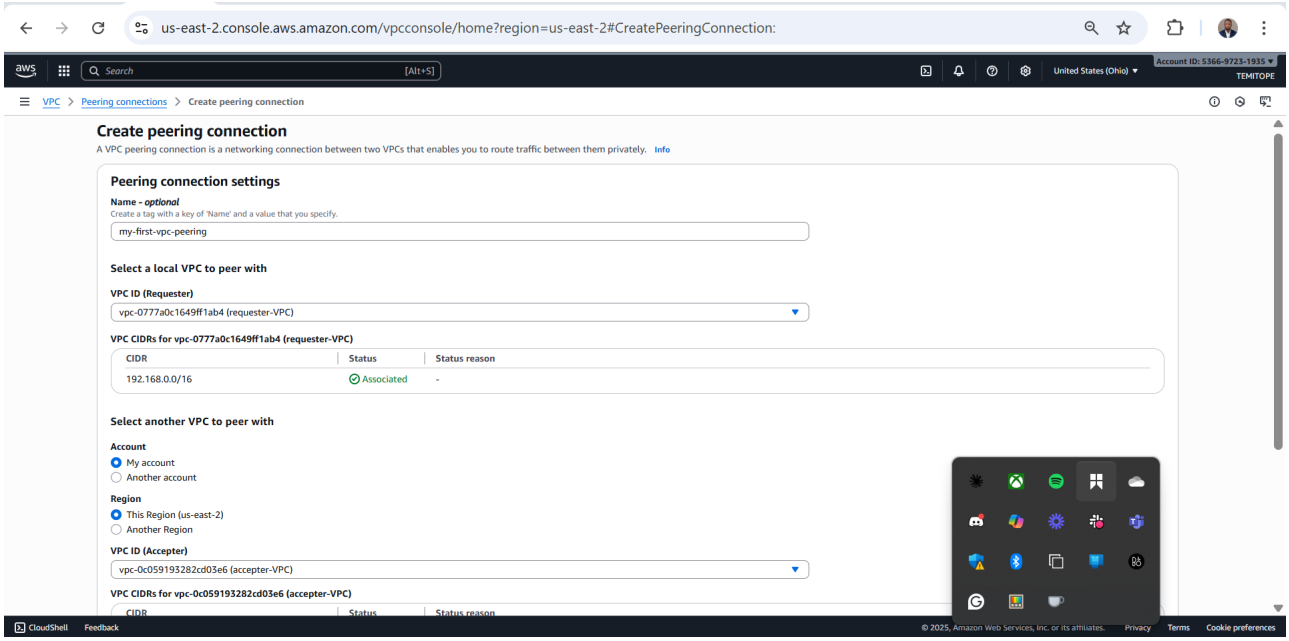
3. Both VPCs created, navigate to **Peering Connections**.



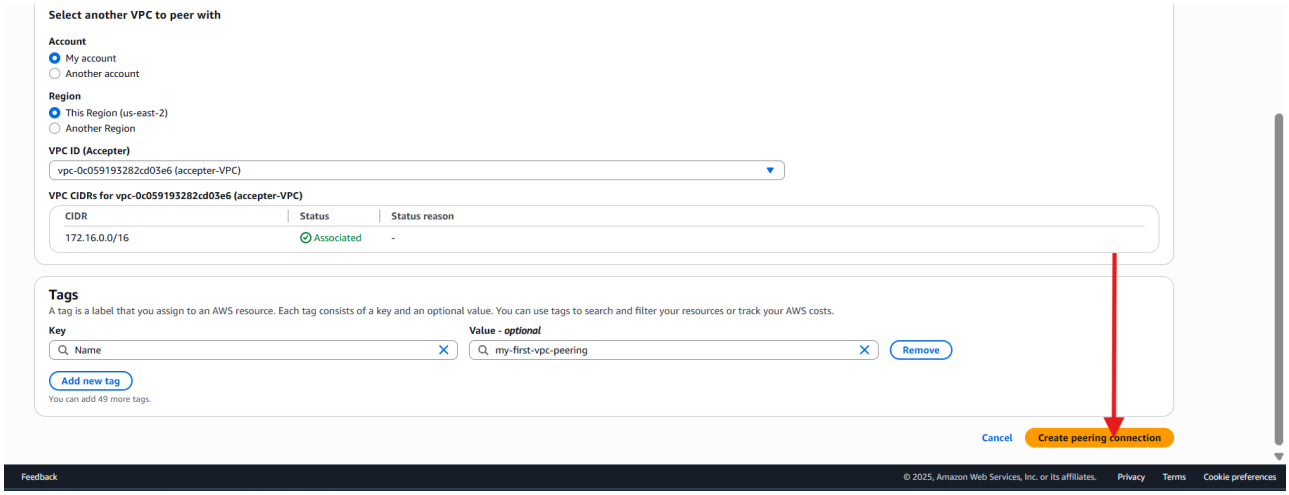
4. Click **Create Peering Connection**.



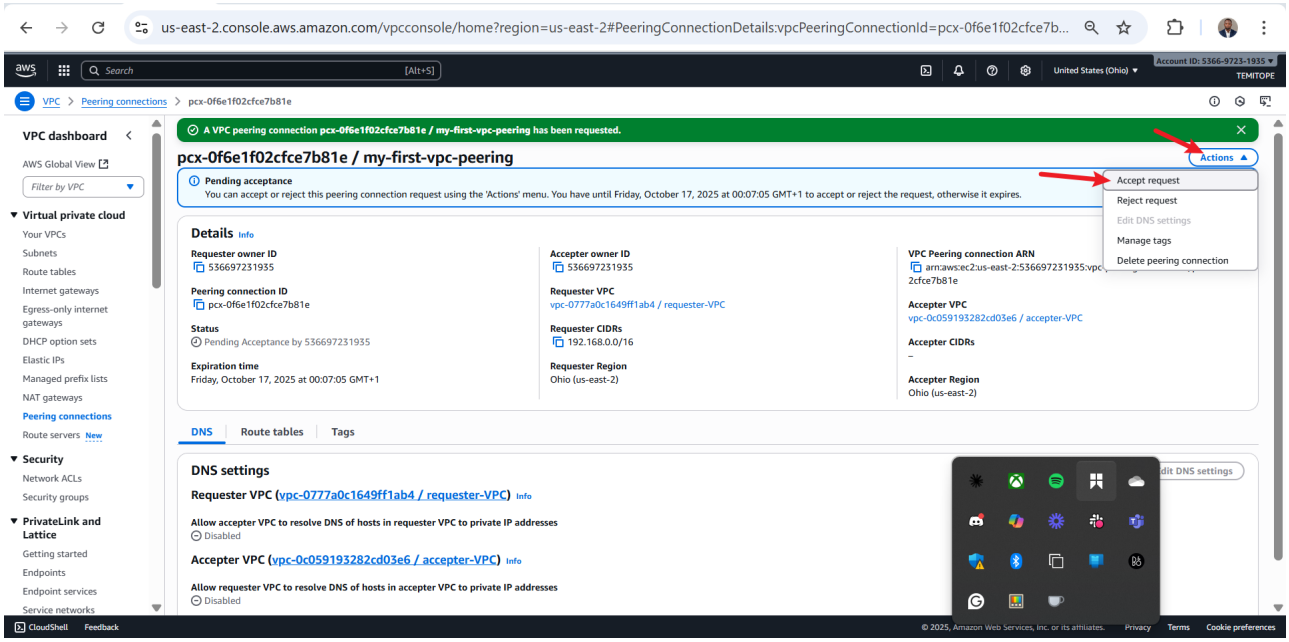
5. Name the peering connection, select requester VPC, choose **My account**, select accepter VPC, and click **Create Peering Connection**.



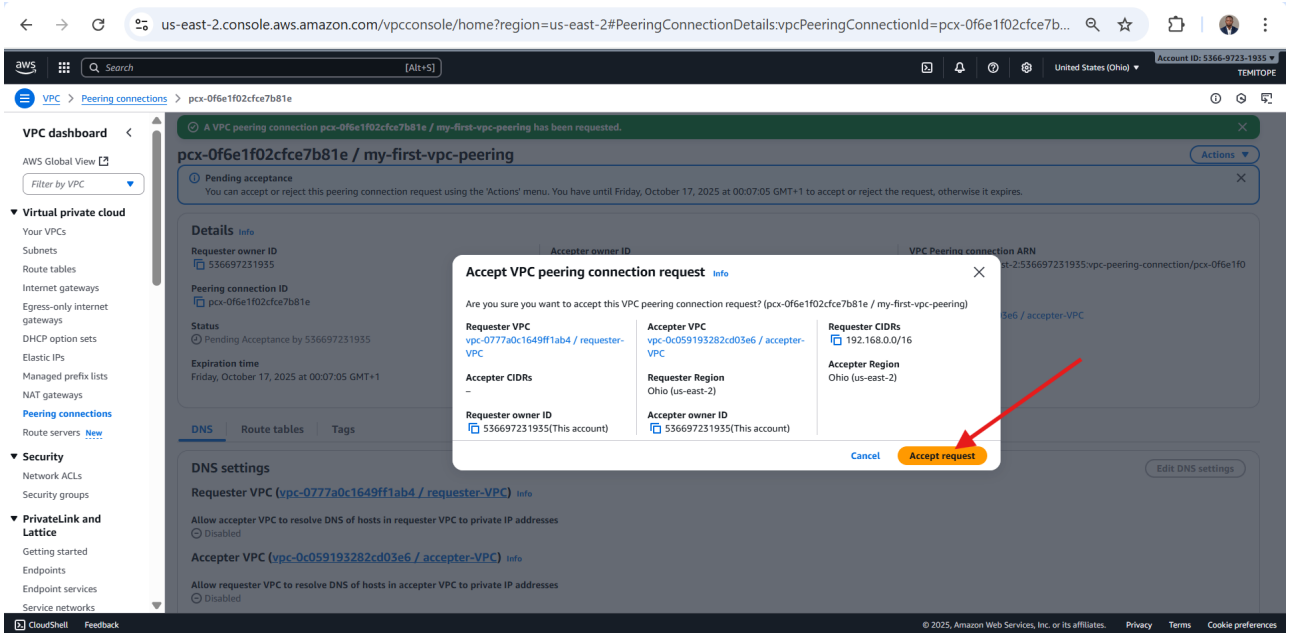
6. Scroll down and create the peering connection.



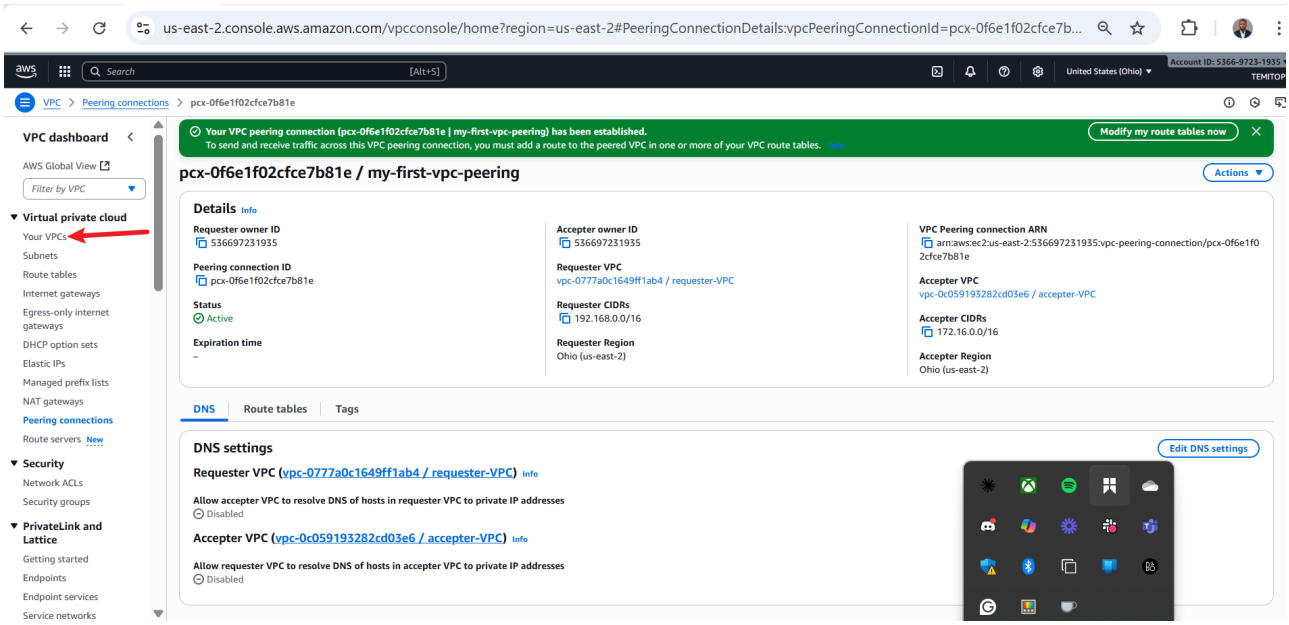
7. VPC peering connection created, click **Actions** and **Accept Request**.



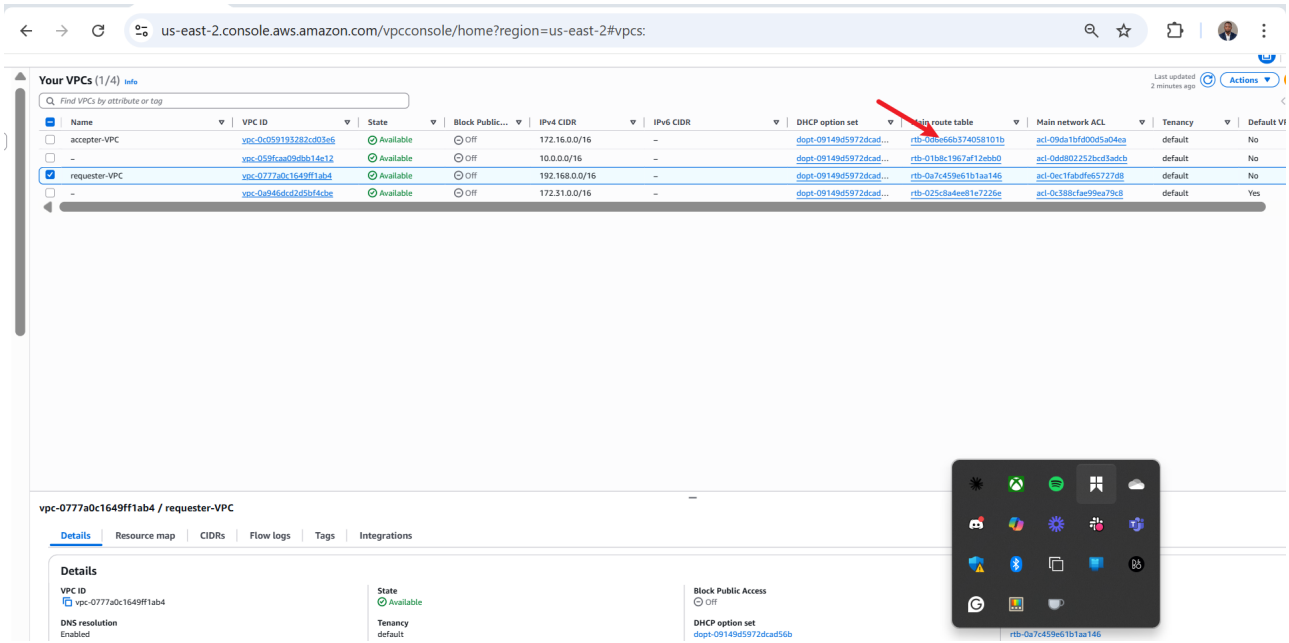
8. Click **Accept Request** in the pop-up.



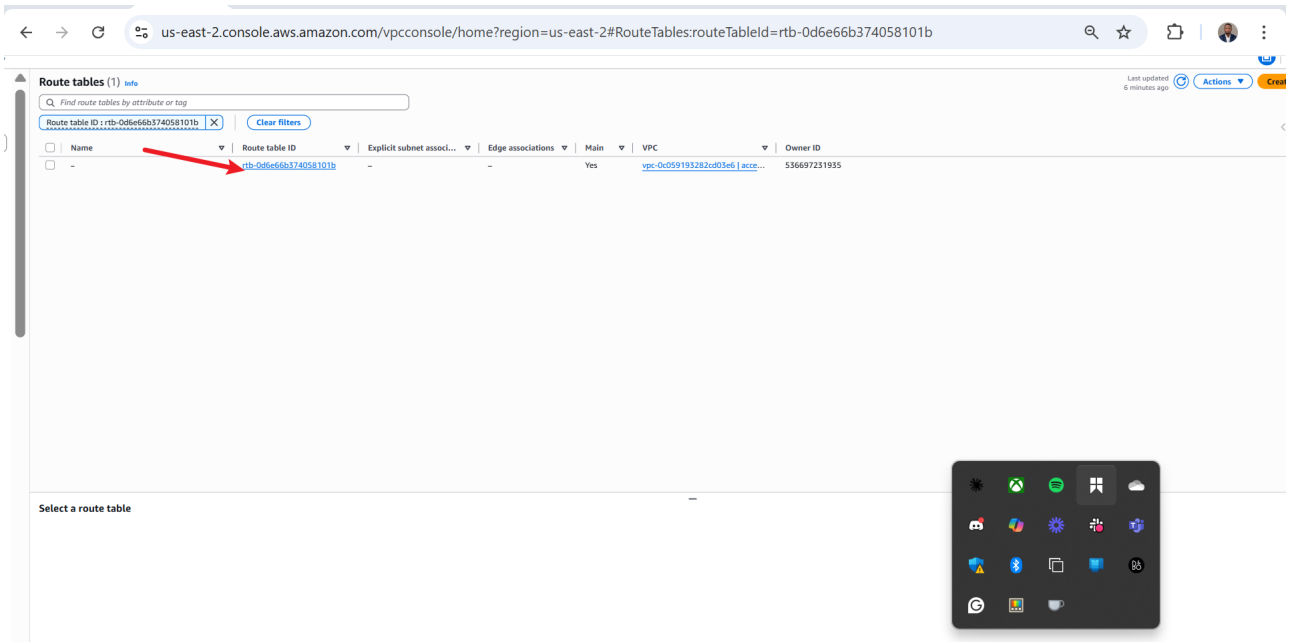
9. VPC peering connection established, click **VPC**.



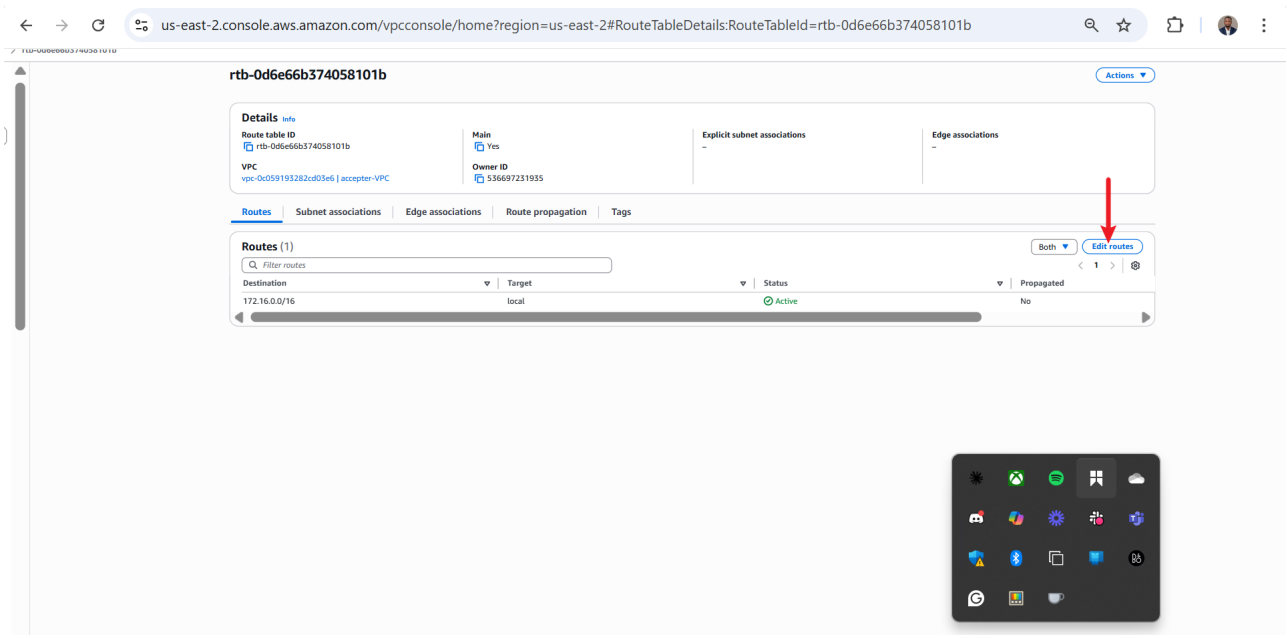
10. Click the accepter VPC's main route table ID.



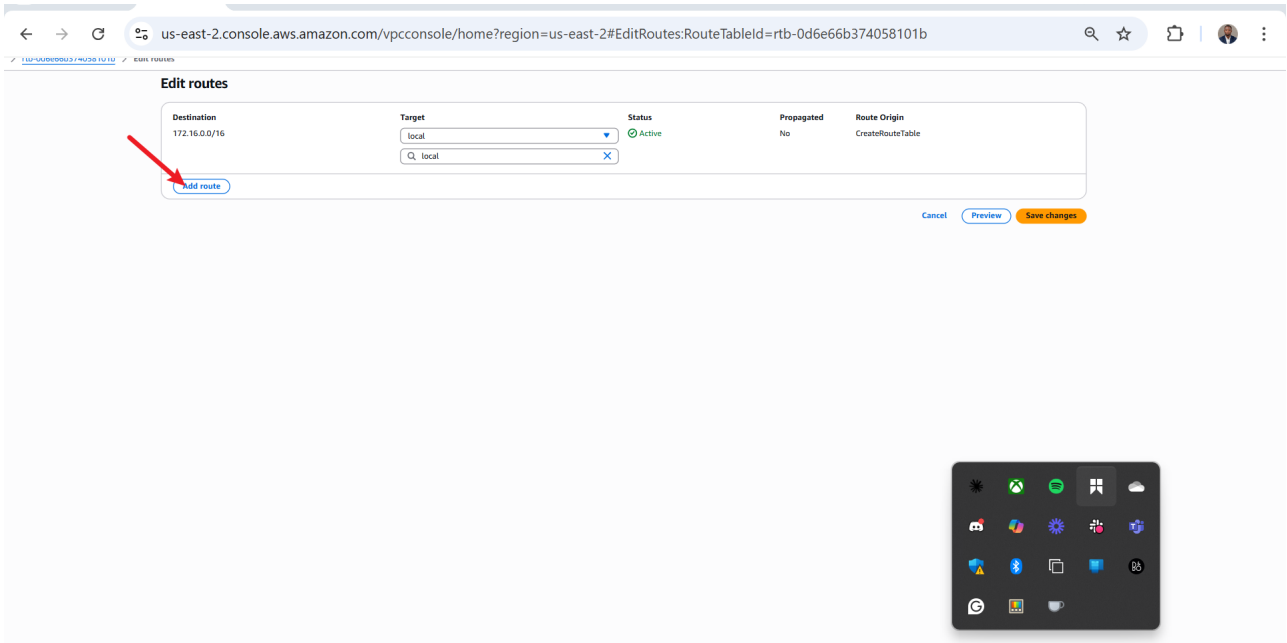
11. Click the route table ID.



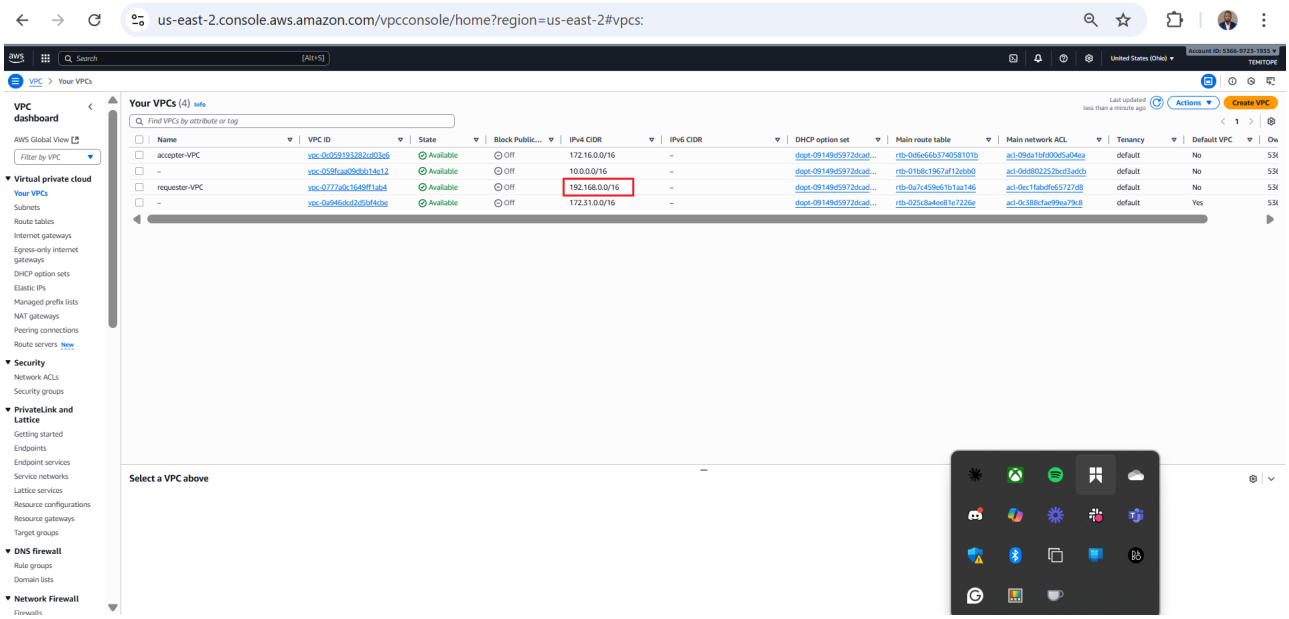
12. Click **Edit routes**.



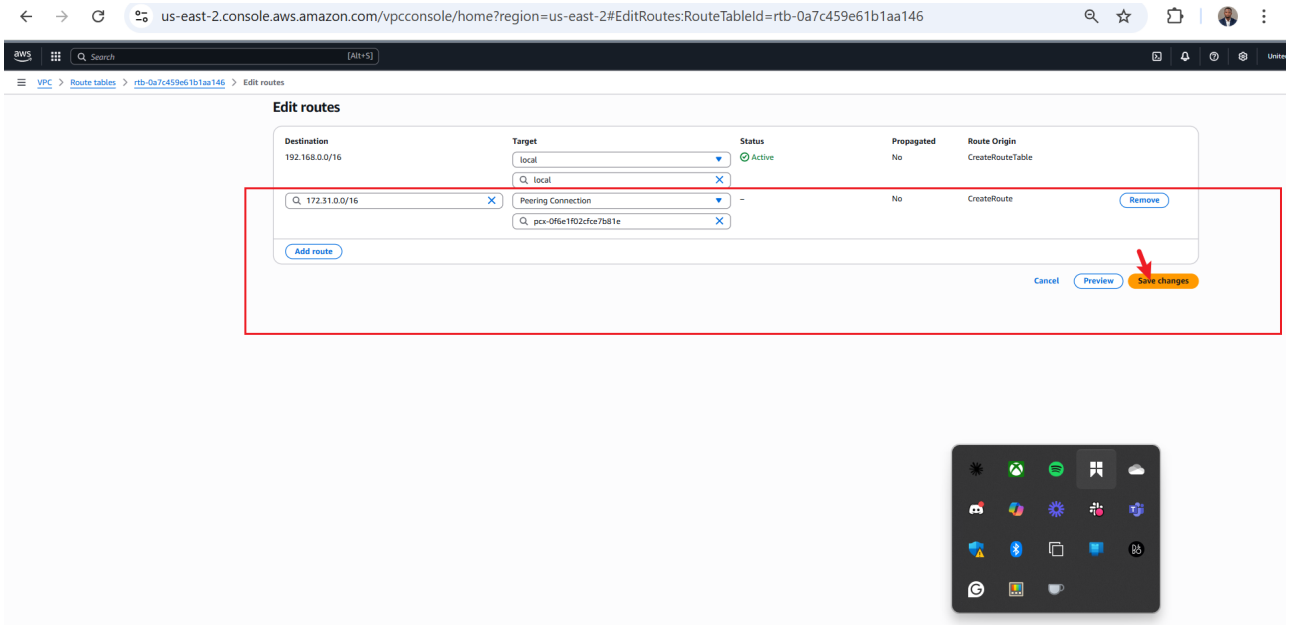
13. Click **Add route**.



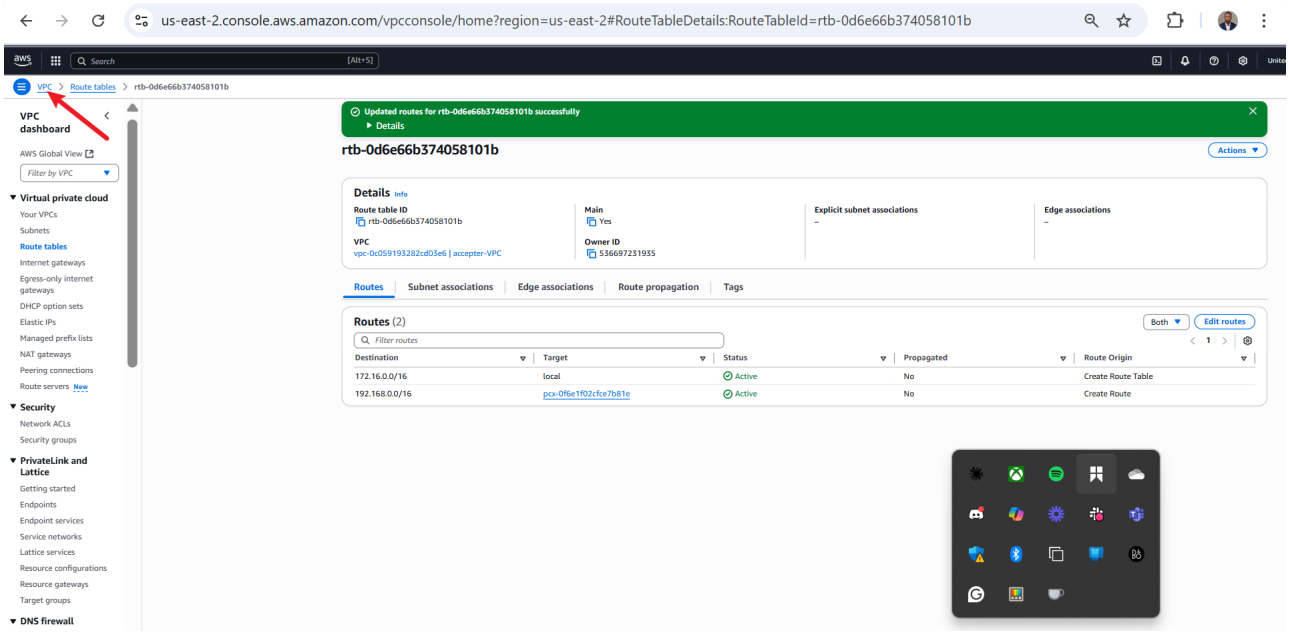
14. Copy the IPv4 CIDR of the requester VPC (192.168.0.0/16).



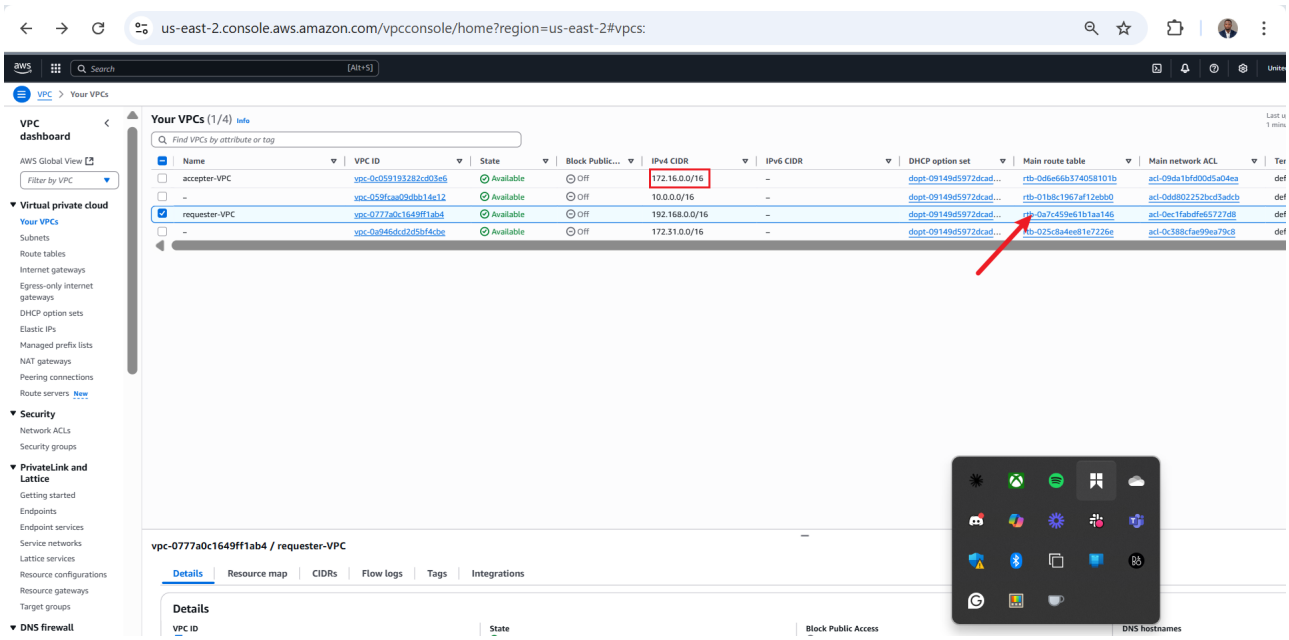
15. Add the requester VPC's CIDR to the **Destination**, set **Target** to the peering connection, and save changes.



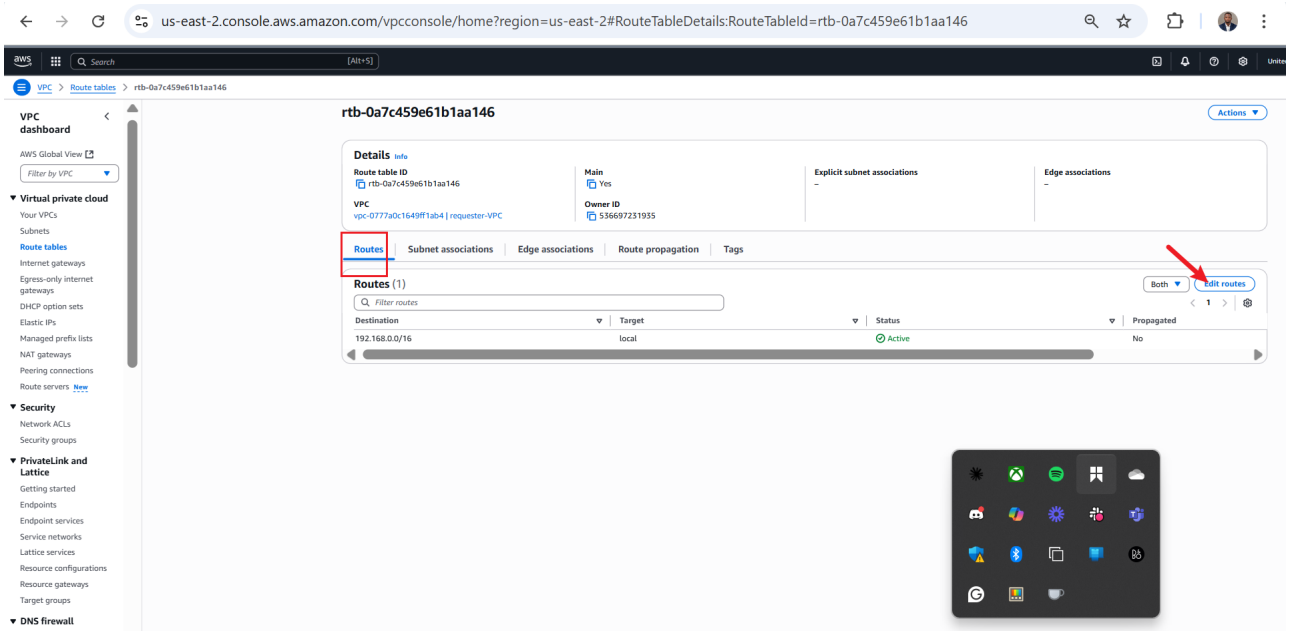
16. Route table updated successfully, return to **VPC**.



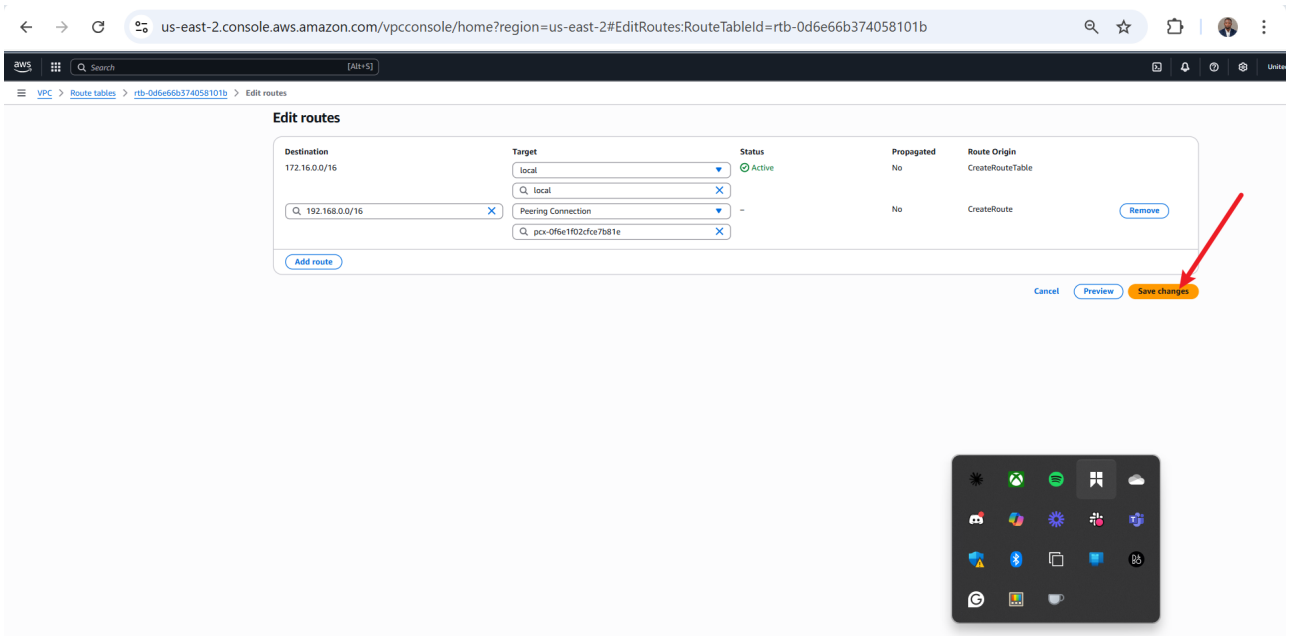
17. Copy the IPv4 CIDR of the accepter VPC (172.16.0.0/16) and click the requester VPC's main route table ID.



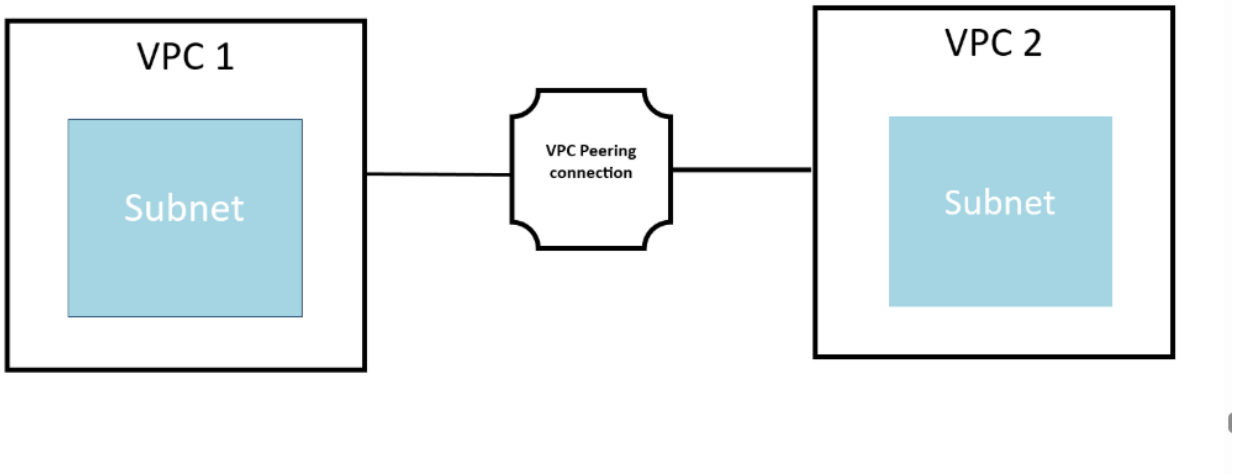
18. Click **Edit route**.



19. Add the accepter VPC's CIDR (172.16.0.0/16) to the **Destination**, set **Target** to the peering connection, select the created peering connection, and save changes.



20. Peering connection successfully established.



The VPC peering connection has been successfully established. Resources in the accepter VPC (172.16.0.0/16) can now communicate with resources in the requester VPC (192.168.0.0/16), and vice versa, using private IP addresses, ensuring secure and direct inter-VPC communication.

Project Validation

To validate the setup, we performed connectivity tests:

- **Public Subnet Connectivity**: Connected to the `Public-EC2` instance via SSH using its public IP and tested outbound internet access by cloning a repository, confirming the Internet Gateway's functionality.
- **Private Subnet Isolation**: Attempted to connect to the `Private-EC2` instance using its public IP, which failed as expected, verifying that the private subnet is not directly accessible from the internet.
- **NAT Gateway Verification**: Observed outbound traffic metrics (Packet Out to Destination and Byte Out to Destination) in the NAT Gateway dashboard, confirming that the private subnet can access the internet securely.
- **Intra-Subnet Communication**: Created a second EC2 instance (`Public-EC2-2`) in the public subnet, established SSH connectivity between the two public EC2 instances, and verified reachability using ping, confirming default intra-subnet communication within the VPC.
- **VPC Peering**: Configured route tables for both VPCs to include each other's CIDR blocks, enabling direct communication between resources in the requester and accepter VPCs.

Project Reflection

- **Achievements**:
    - Successfully completed all project tasks, including setting up a VPC, configuring subnets, attaching an Internet Gateway, enabling outbound access via a NAT Gateway, and establishing VPC peering.
    - Gained hands-on expertise in navigating the AWS Management Console to deploy and manage VPC resources.
    - Demonstrated the ability to configure secure network architectures, ensuring resources in public subnets have internet access while private subnets remain isolated yet capable of outbound

connectivity.
    - Established and validated VPC peering, enabling seamless communication between two VPCs.

- **Challenges and Solutions**:

    - Encountered CIDR block size limitations during VPC creation, resolved by adjusting the CIDR block to fall within the recommended range (/16 to /28).
    - Ensured non-overlapping CIDR blocks for VPC peering to avoid conflicts, reinforcing the importance of proper IP address planning.
    - Configured security group rules and route tables meticulously to enable connectivity while maintaining security best practices.

- **Key Learnings**:

    - Developed a deeper understanding of cloud networking concepts, including the roles of subnets, gateways, and route tables in AWS VPCs.
    - Learned the significance of network security measures like NAT Gateways and VPC endpoints for protecting sensitive data.
    - Gained practical experience in troubleshooting network configurations and validating connectivity through real-world tests.

Conclusion

This project provided a comprehensive exploration of AWS VPC fundamentals, from creating and configuring a VPC to enabling secure internet access and inter-VPC communication. The hands-on exercises reinforced theoretical knowledge with practical application, enhancing proficiency in cloud network architecture. The successful setup of public and private subnets, Internet and NAT Gateways, and VPC peering connections demonstrates the ability to design secure and scalable cloud environments. The project also highlighted the importance of network security and proper configuration to prevent unauthorized access while enabling necessary connectivity.

Overall, this mini-project offered valuable insights into cloud networking and practical skills in deploying VPC infrastructure on AWS, equipping participants with the knowledge and confidence to tackle more complex cloud networking challenges in the future.
