Project: Configuring an Amazon VPC

Amazon Virtual Private Cloud (Amazon VPC) gives the ability to provision a logically isolated section of the Amazon Web Services (AWS) Cloud where one can launch AWS resources in a virtual network that one defines. One has complete control over their virtual networking environment, including selecting IP address ranges, creating subnets, and configuring route tables and network gateways.

In this project, I have built a virtual private cloud (VPC) and other network components required to deploy resources, such as an Amazon Elastic Compute Cloud (Amazon EC2) instance**.**

Objectives

End of this project we will be able to do the following:

- Create a VPC with a private and public subnet, an internet gateway, and a NAT gateway.
- Configure route tables associated with subnets to local and internet-bound traffic by using an internet gateway and a NAT gateway.
- Launch a bastion server in a public subnet.

Task 1: Creating a VPC

In this task, I have created a new VPC.

5. On the **AWS Management Console**, in the **Search** bar, enter and choose VPC to go to the **VPC Management Console**.
6. In the left navigation pane, for **Virtual private cloud**, choose **Your VPCs**.

In every Region, a default VPC with a Classless Inter-Domain Routing (CIDR) block of 172.31.0.0/16 has already been created for you. Even if you haven't created anything in your account yet, you will see some pre-existing VPC resources already there.

7. Choose **Create VPC** and configure the following options:

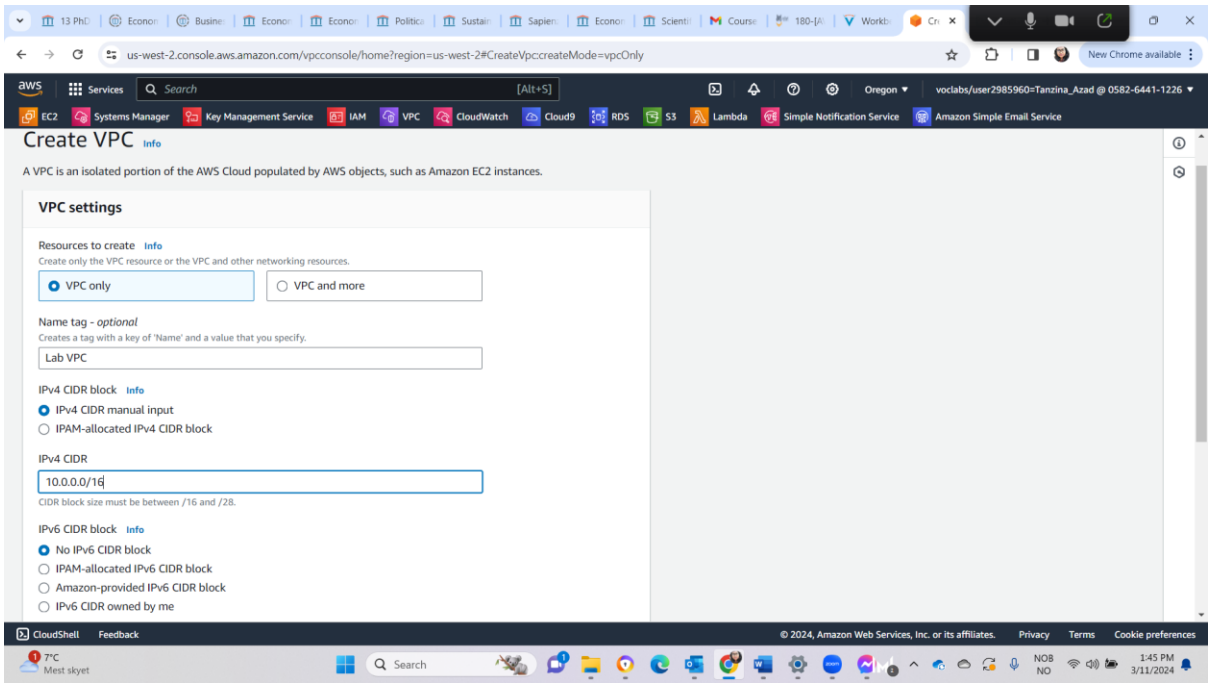
- **Resources to create:** Choose **VPC only**.
- **Name tag:** Enter Lab VPC.
- **IPv4 CIDR block:** Choose **IPv4 CIDR manual input**.
- **IPv4 CIDR:** Enter 10.0.0.0/16.
- **IPv6 CIDR block:** Choose **No IPv6 CIDR block**.
- **Tenancy:** Choose **Default**.
- **Tags:** Leave the suggested tags as is.

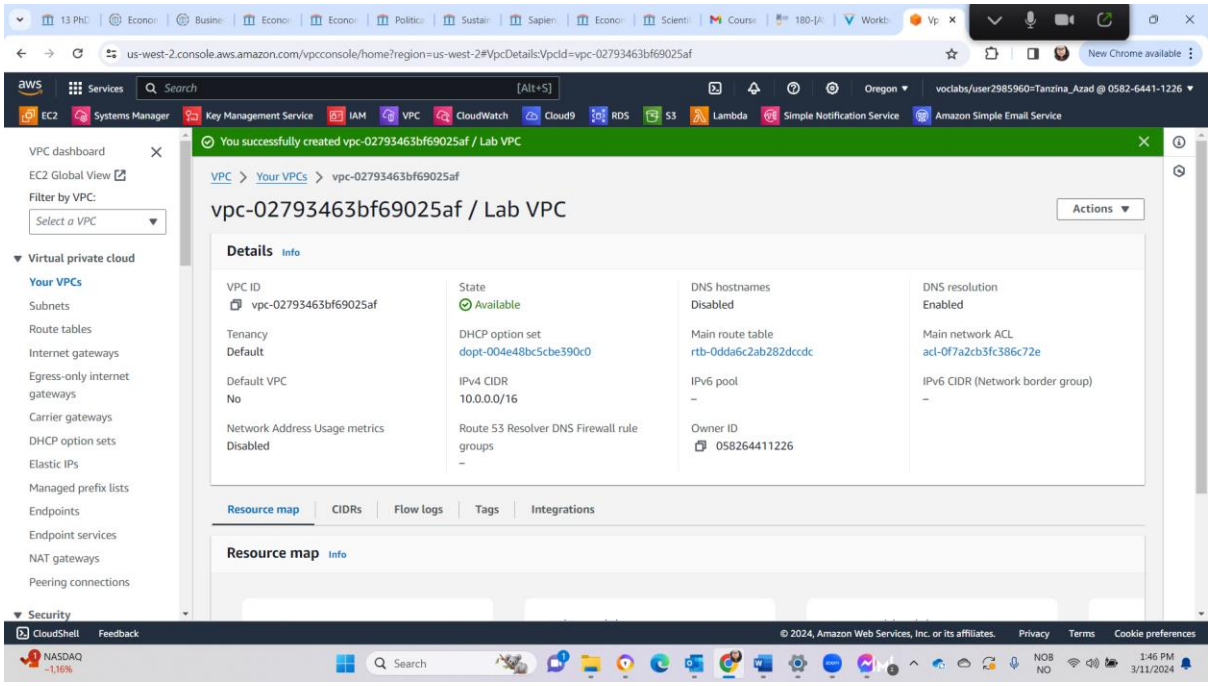
8. Choose **Create VPC**.

At the top of the page, a message displays similar to the following: "You successfully created vpc-NNNNNNNNNNN / Lab VPC."

9. Choose **Actions**, and choose **Edit VPC settings**.
10. In the **DNS settings** section, select **Enable DNS hostnames**.
11. Choose **Save**.

EC2 instances launched into the VPC now automatically receive a public IPv4 Domain Name System (DNS) hostname.





Task 2: Creating subnets

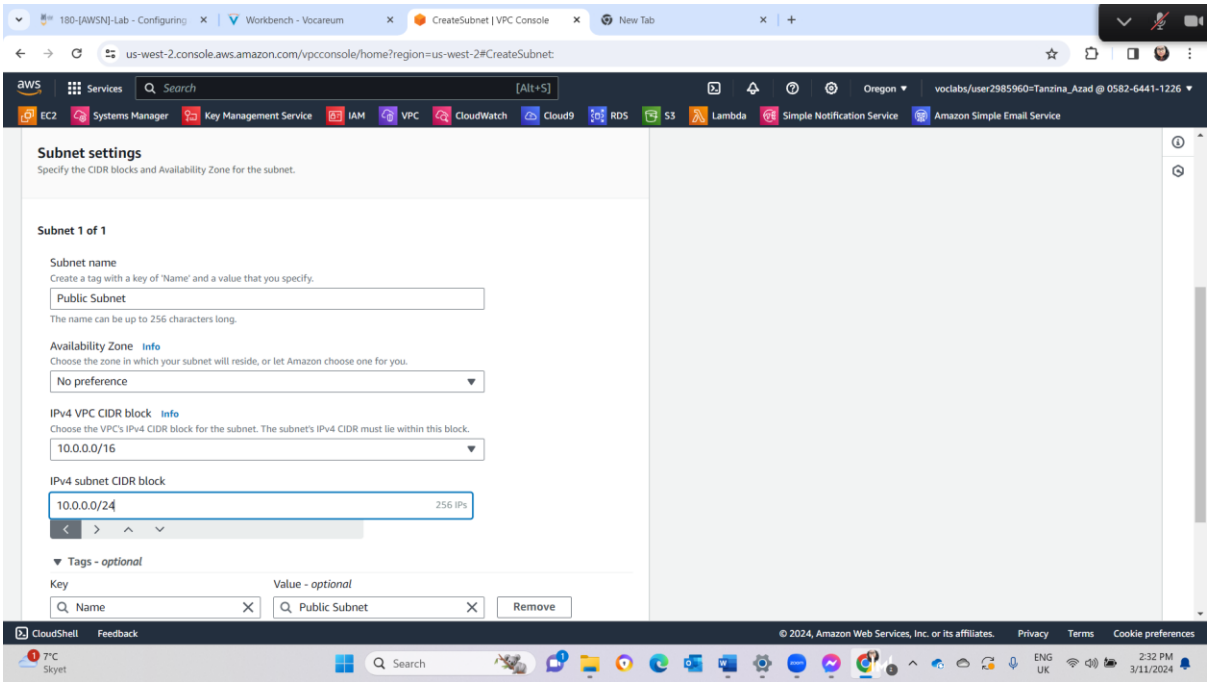
In this task, I will create a public subnet and a private subnet.

Task 2.1: Creating a public subnet

12. In the left navigation pane, for **Virtual private cloud**, choose **Subnets**.
13. Choose **Create subnet** and configure the following options:

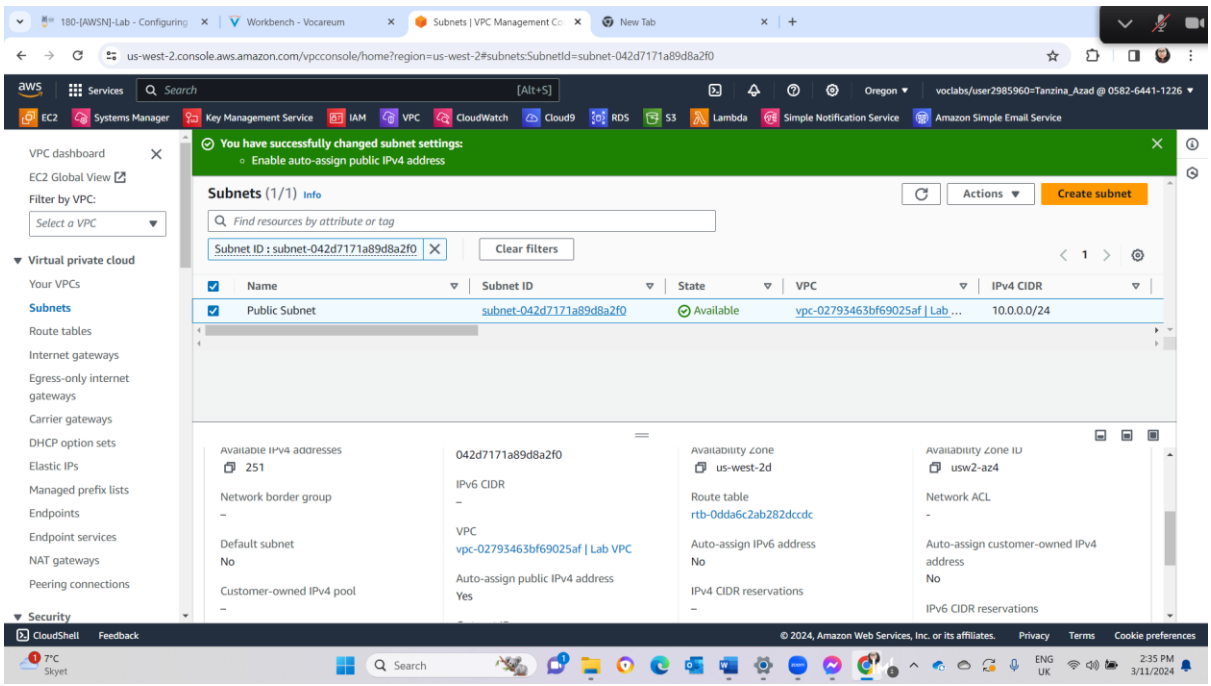
- **VPC ID:** Choose **Lab VPC**.
- **Subnet name:** Enter Public Subnet.
- **Availability Zone:** Choose the first Availability Zone in the list. Do not choose **No preference**.
- **IPv4 CIDR block:** Enter 10.0.0.0/24.

14. Choose **Create subnet**.



You now configure the public subnet to automatically assign a public IP address for all EC2 instances that are launched within it.

15. Select **Public Subnet**.
16. Choose **Actions**, and then choose **Edit subnet settings**.
17. In the **Auto-assign IP settings** section, select **Enable auto-assign public IPv4 address**.
18. Choose **Save**.



Even though this subnet has been named **Public Subnet**, it is not yet public. A public subnet must have an internet gateway, which you attach in a task later in the lab.

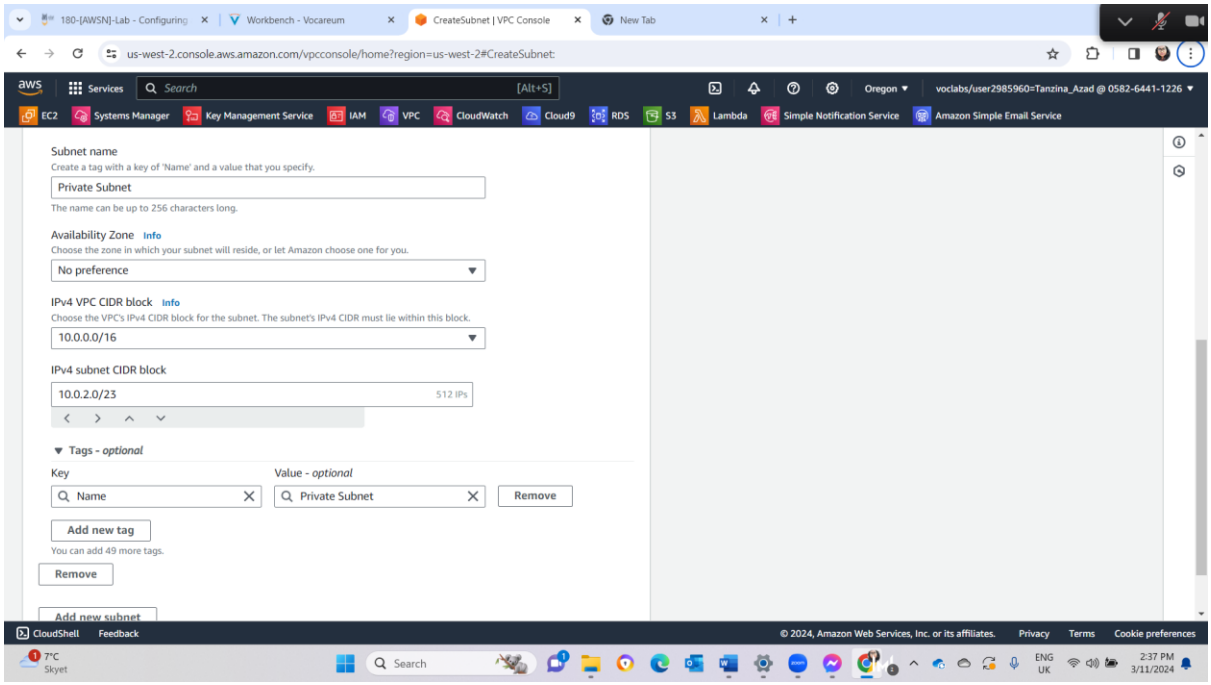
Task 2.2: Creating a private subnet

In this task, I create the private subnet, which is used for resources that are to remain isolated from the internet.

19. To create the private subnet, repeat the steps from the previous task, and choose the following options:

- **VPC ID:** Choose **Lab VPC**.
- **Subnet name:** Enter Private Subnet.
- **Availability Zone:** Choose the first Availability Zone on the list. Do not choose **No preference**.
- **IPv4 CIDR block:** Enter 10.0.2.0/23.

20. Choose **Create subnet**.



The CIDR block of 10.0.2.0/23 includes all IP addresses that start with 10.0.2.x and 10.0.3.x. This range is twice as large as the public subnet because most resources should be kept in private subnets unless they specifically need to be accessible from the internet.

The VPC now has two subnets. However, the VPC is totally isolated and cannot communicate with resources outside the VPC.

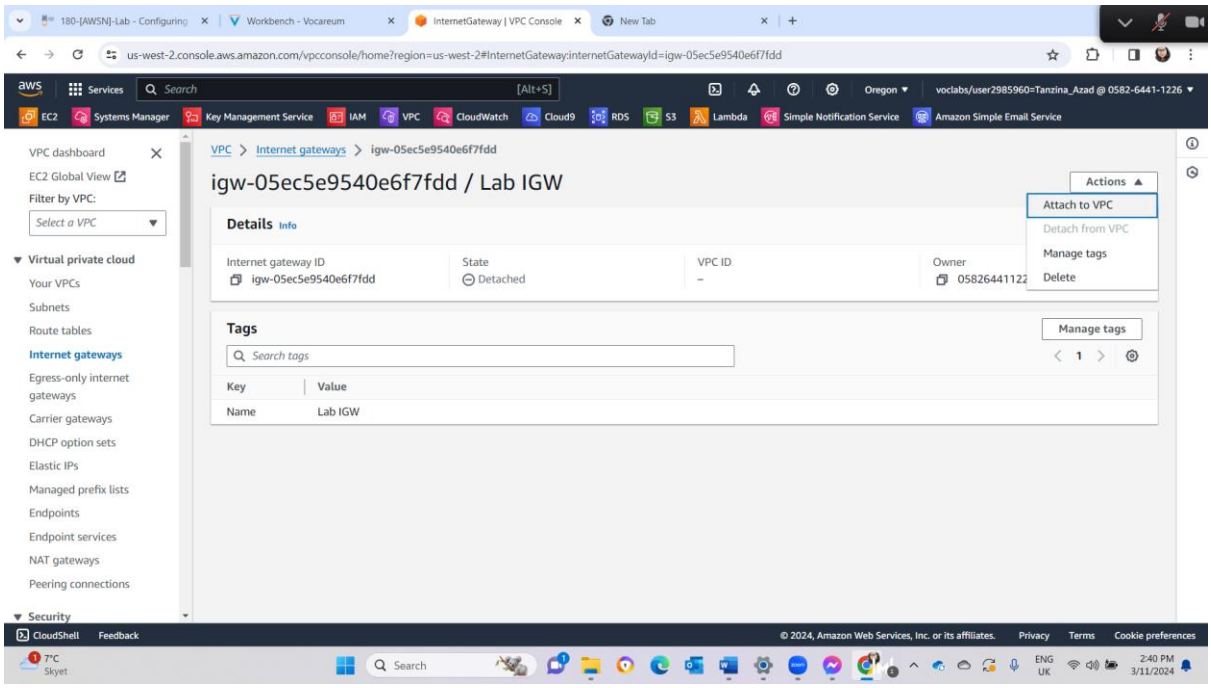
Next, need to configure the public subnet to connect to the internet through an internet gateway.

Task 3: Creating an internet gateway

In this task, I create an internet gateway for your VPC. I need an internet gateway to establish outside connectivity to EC2 instances in VPCs.

21. In the left navigation pane, for **Virtual private cloud**, choose **Internet gateways**.
22. Choose **Create internet gateway**, and then for **Name tag**, enter Lab IGW.
23. Choose **Create internet gateway**.
24. Choose **Actions**, then choose **Attach to a VPC**.

My public subnet now has a connection to the internet. However, to route traffic to the internet, I must also configure the public subnet's route table so that it uses the internet gateway.



Task 4: Configuring route tables

In this task, I will do the following:

- Create a public route table for internet-bound traffic.
- Add a route to the route table to direct internet-bound traffic to the internet gateway.
- Associate the public subnet with the new route table.

25. In the left navigation pane, for **Virtual private cloud**, choose **Route tables**.

Several route tables are listed.

26. Select the route table that includes **Lab VPC** in the **VPC** column.

Tip: If you cannot see the VPC column, scroll to the right.

27. In the **Name** column, choose the edit icon, enter Private Route Table for **Edit Name**, and then choose **Save**.

28. Choose the **Routes** tab.

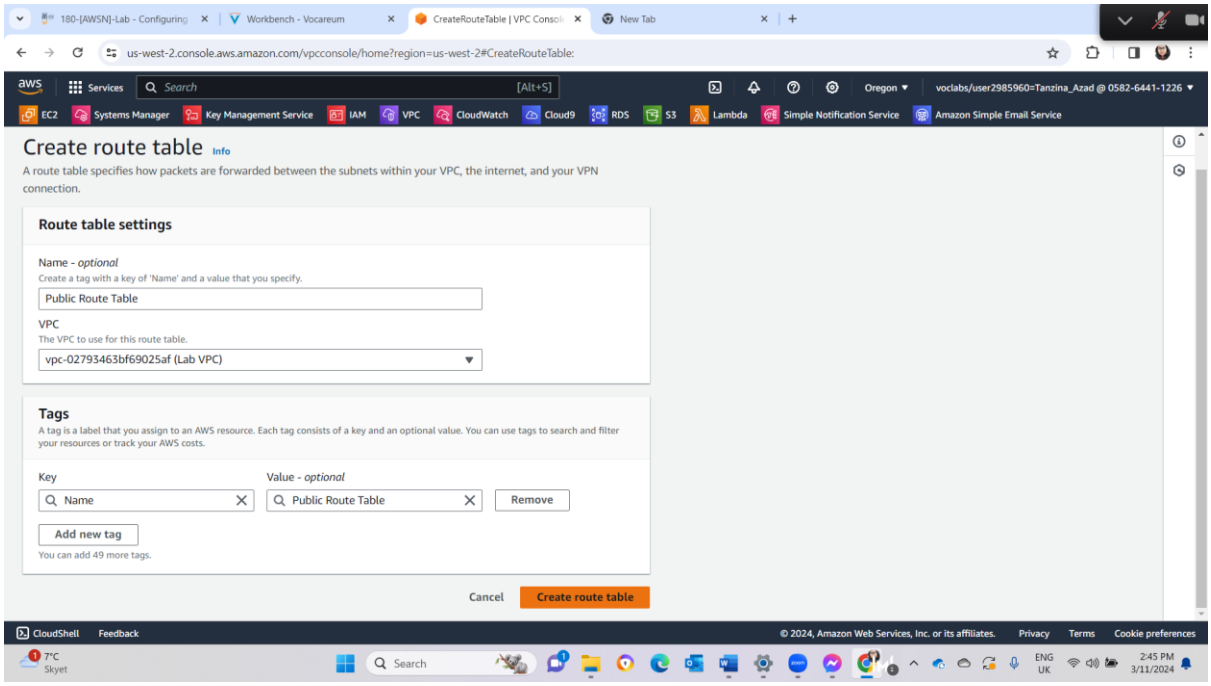
There is currently only one route. It shows that all traffic destined for 10.0.0.0/16 (which is the range of the Lab VPC) will be routed locally. This option allows all subnets within a VPC to communicate with each other.

Now create a new public route table to send public traffic to the internet gateway.

29. Choose **Create route table** and configure the following options:

- **Name - *optional*:** Enter Public Route Table.
- **VPC:** Choose **Lab VPC**.

30. Choose **Create route table**.



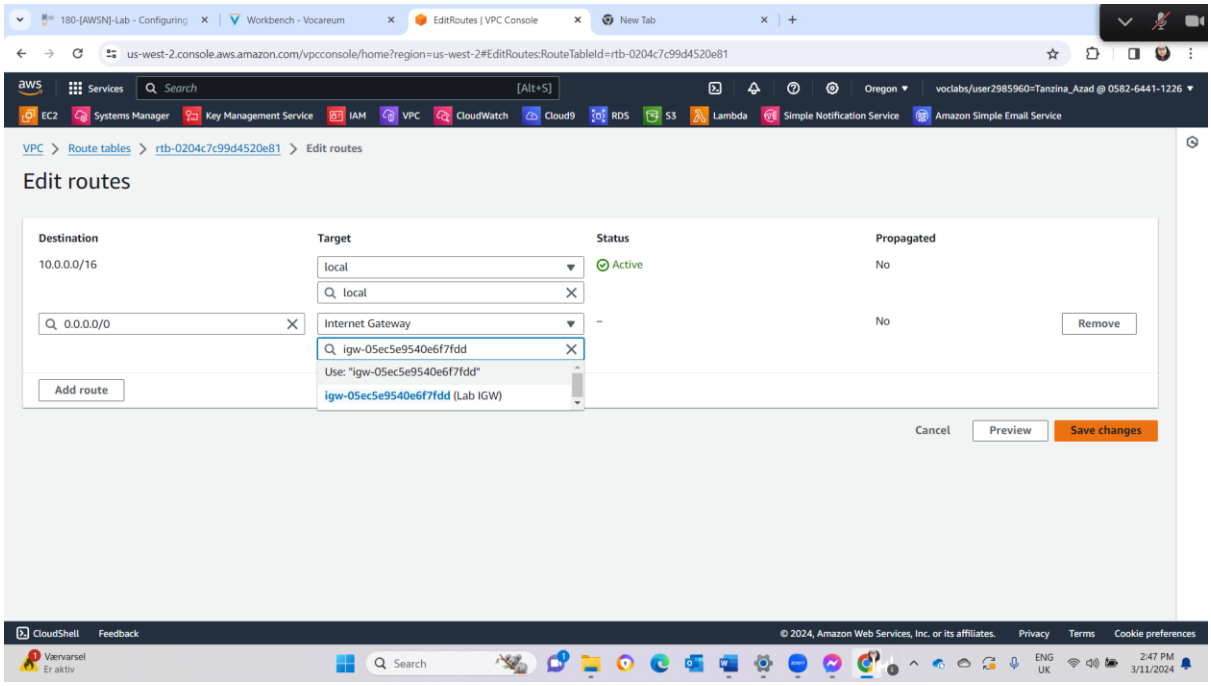
31. After the route table is created, in the **Routes** tab, choose **Edit routes**.

Note: Now add a route to direct internet-bound traffic (0.0.0.0/0) to the internet gateway.

32. Choose **Add route** and then configure the following options:

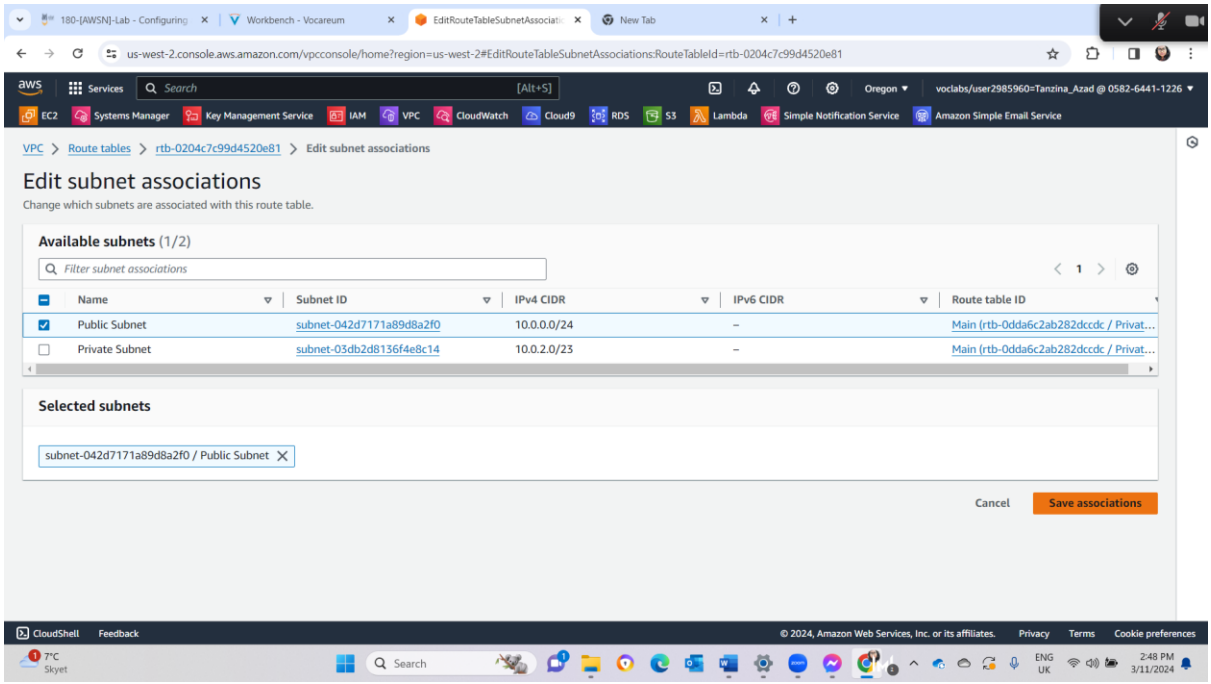
- **Destination:** Enter 0.0.0.0/0.
- **Target:** Choose **Internet Gateway**, and then choose **Lab IGW** from the list.

33. Choose **Save changes**.



The final step is to associate this new route table with the public subnet.

34. Choose the **Subnet associations** tab.
35. Choose **Edit subnet associations**.
36. Select **Public Subnet**.
37. Choose **Save associations**.



The public subnet is now public because it has a route table entry that sends traffic to the internet through the internet gateway.

In the previous tasks, I created a VPC and attached an internet gateway. Then you created subnets and a route table and associated a public route table to the public subnet. now launch resources in the subnets as required.

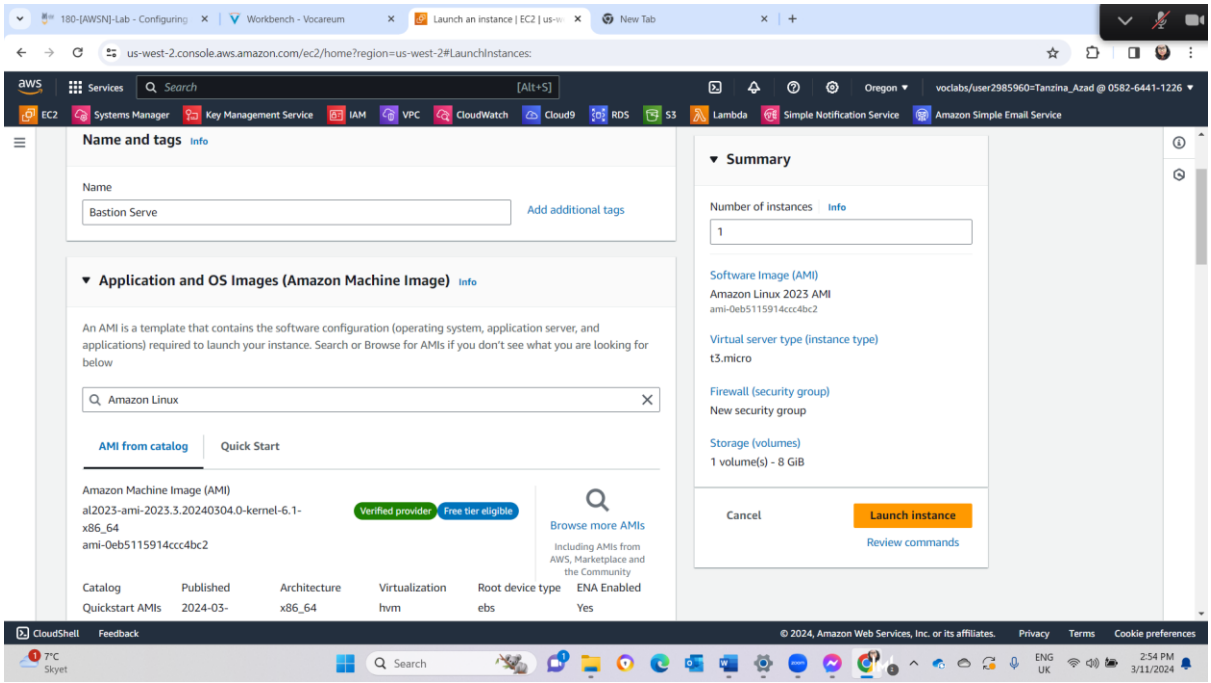
Task 5: Launching a bastion server in the public subnet

A bastion server (also known as a jump box) is an EC2 instance in a public subnet that is securely configured to provide access to resources in a private subnet. Systems operators can connect to the bastion server and then jump into resources in the private subnet.

In this task, launch an EC2 instance bastion server in the public subnet that was created earlier.

38. On the AWS Management Console, in the **Search** bar, enter and choose EC2 to go to the **EC2 Management Console**.
39. In the left navigation pane, choose **Instances**.
40. Choose **Launch instances** and configure the following options:

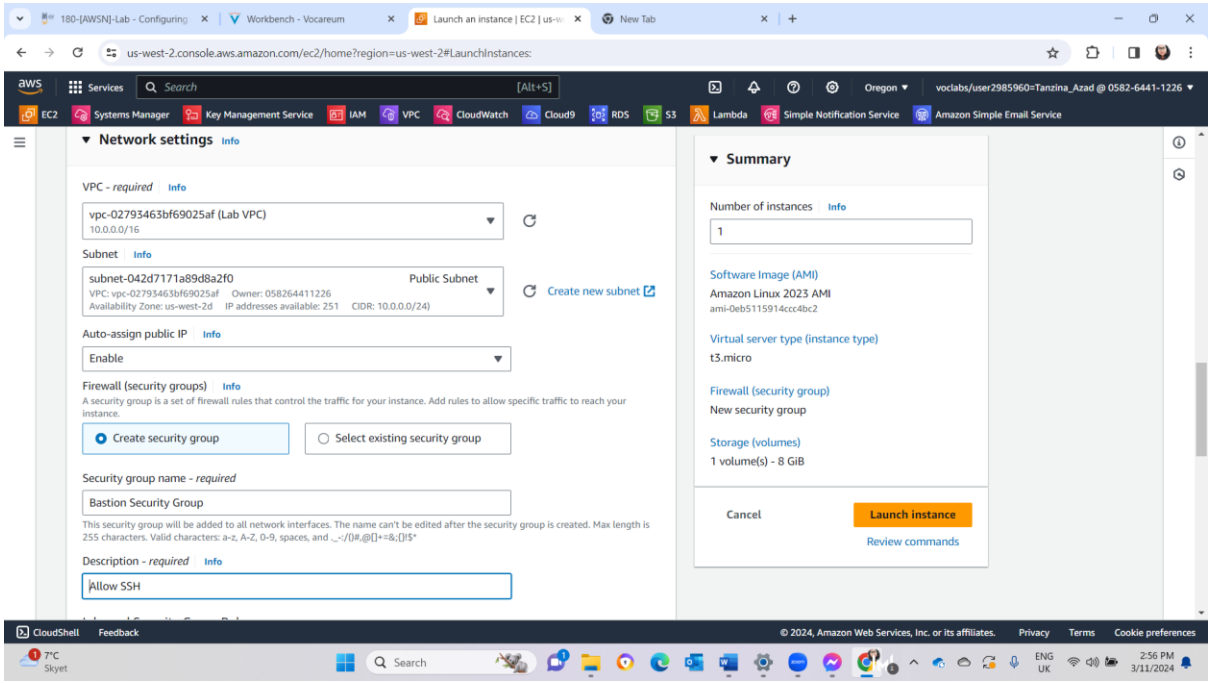
- In the **Name and tags** section, enter Bastion Server.
- In the **Application and OS Images (Amazon Machine Image)** section, configure the following options:
    - **Quick Start:** Choose **Amazon Linux**.
    - **Amazon Machine Image (AMI):** Choose **Amazon Linux 2023 AMI**.
- In the **Instance type** section, choose **t3.micro**.
- In the **Key pair (login)** section, choose **Proceed without a key pair (Not recommended)**.



I use EC2 Instance Connect to access the shell running on the EC2 instance, so a key pair is not needed in the lab.

41. In the **Network settings** section, choose Edit and configure the following options:

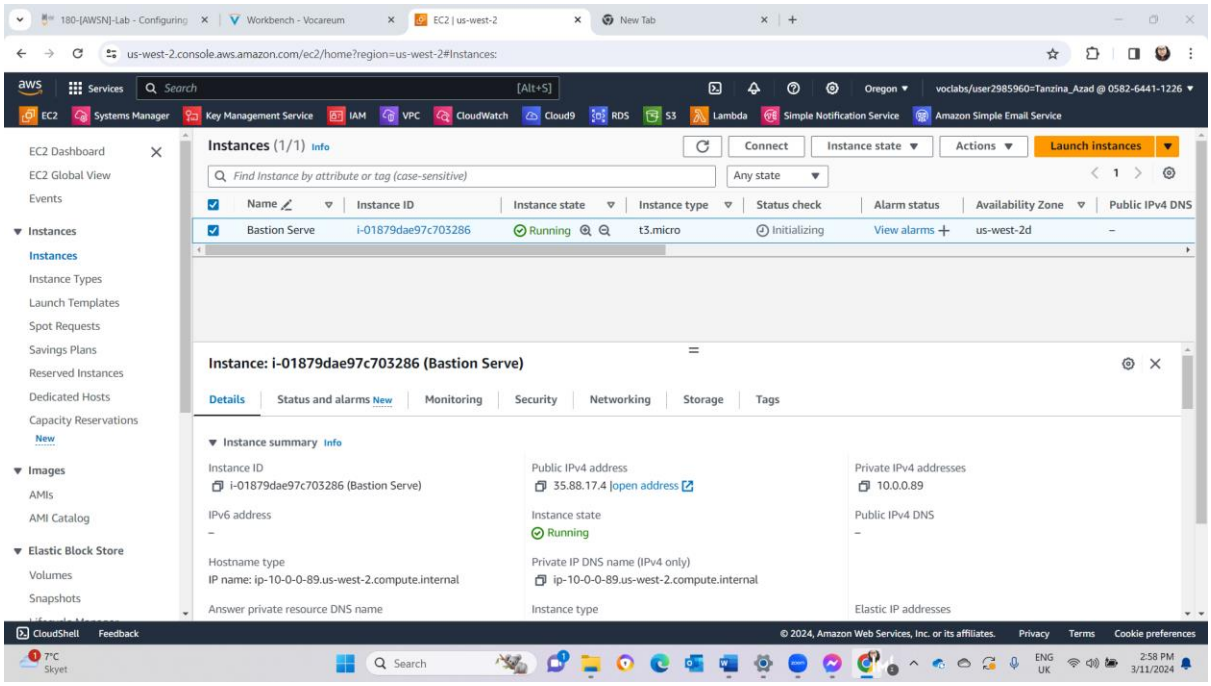
- **VPC - *required*:** Choose **Lab VPC**.
- **Subnet:** Choose **Public Subnet**.
- **Auto-assign public IP:** Choose **Enable**.
- **Firewall (security groups):** Choose **Create security group**.
    - **Security group name - *required*:** Enter Bastion Security Group.
    - **Description - *required*:** Enter Allow SSH.



- **Inbound security groups rules:**
    - **Type:** Choose **ssh**.
    - **Source type:** Choose **Anywhere**.

42. Choose **Launch instance**.

43. To display the launched instance, choose **View all instances**.



The EC2 instance named **Bastion Server** is initially in a *Pending* state. The **Instance state** then changes to *Running* to indicate that the instance has finished booting.

The bastion server will be launched in the public subnet.

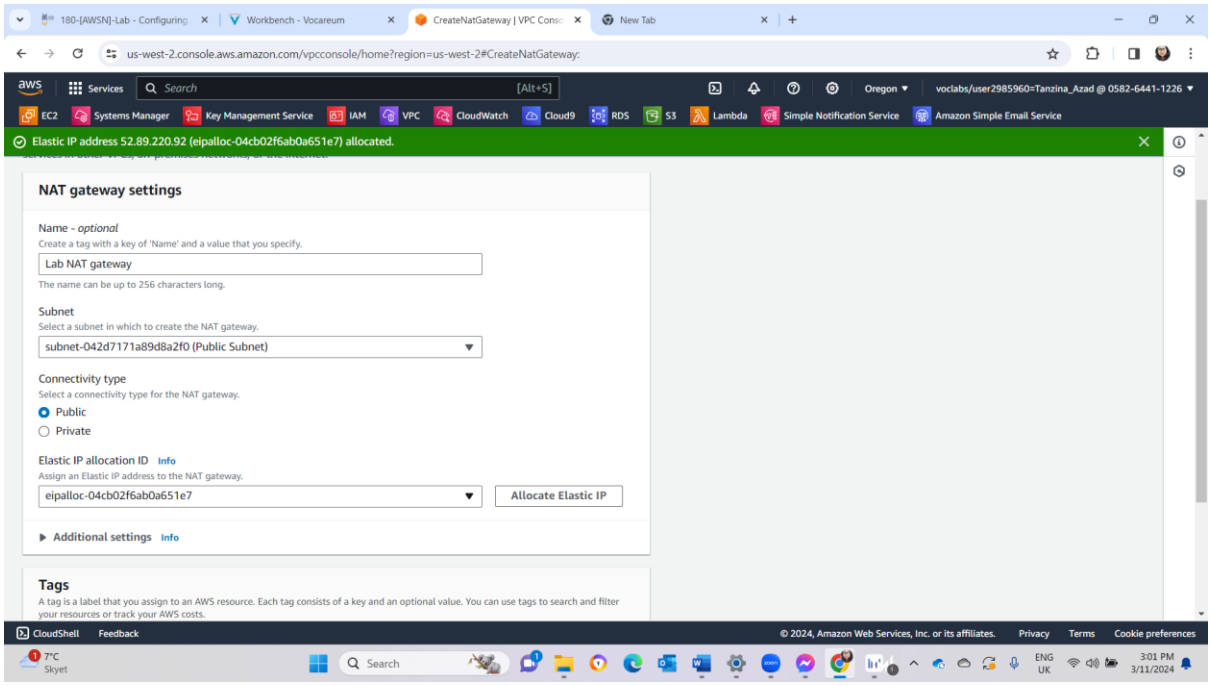
Task 6: Creating a NAT gateway

In this task, you launch a NAT gateway in the public subnet and configure the private route table to facilitate communication between resources in the private subnet and the internet.

44. On the AWS Management Console, in the **Search** bar, enter NAT gateways, choose the **Features** list, and choose **NAT gateways**.
45. Choose **Create NAT gateway** and configure the following options:

- **Name:** Enter Lab NAT gateway.
- **Subnet:** From the dropdown list, choose **Public Subnet**.

46. Choose **Allocate Elastic IP**.
47. Choose **Create a NAT gateway**.



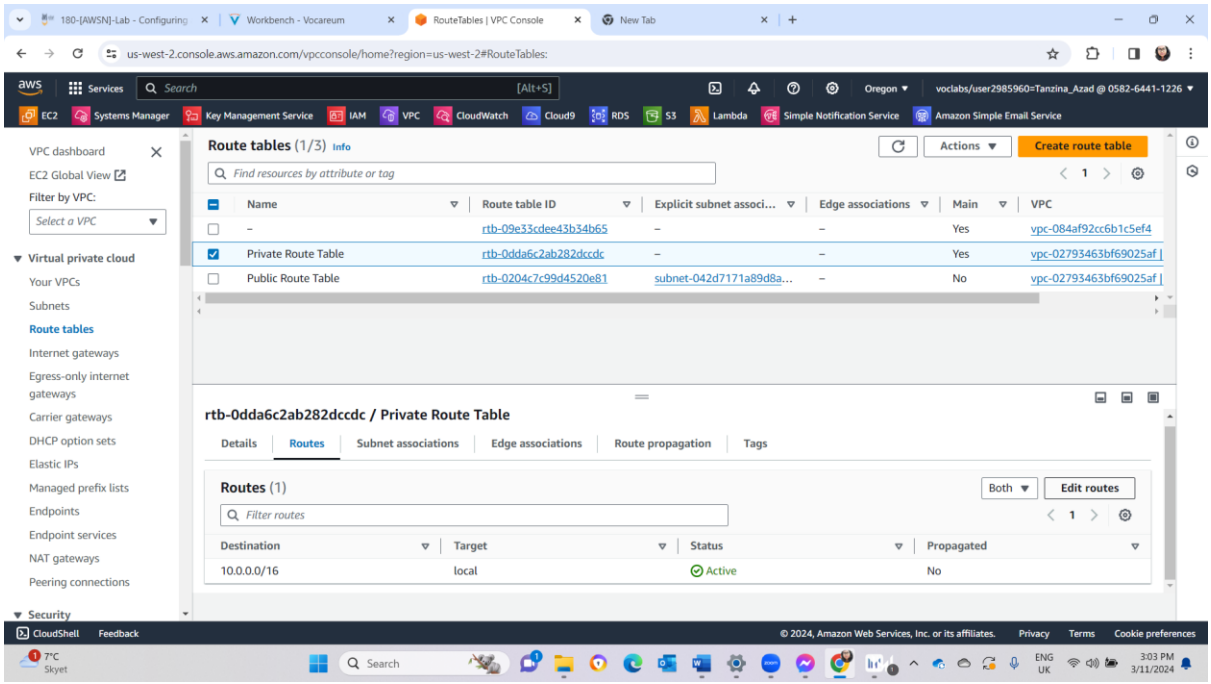
You now configure the private subnet to send internet-bound traffic to the NAT gateway.

48. In the left navigation pane, choose **Route tables**, and then select **Private Route Table**.

49. Choose the **Routes** tab.

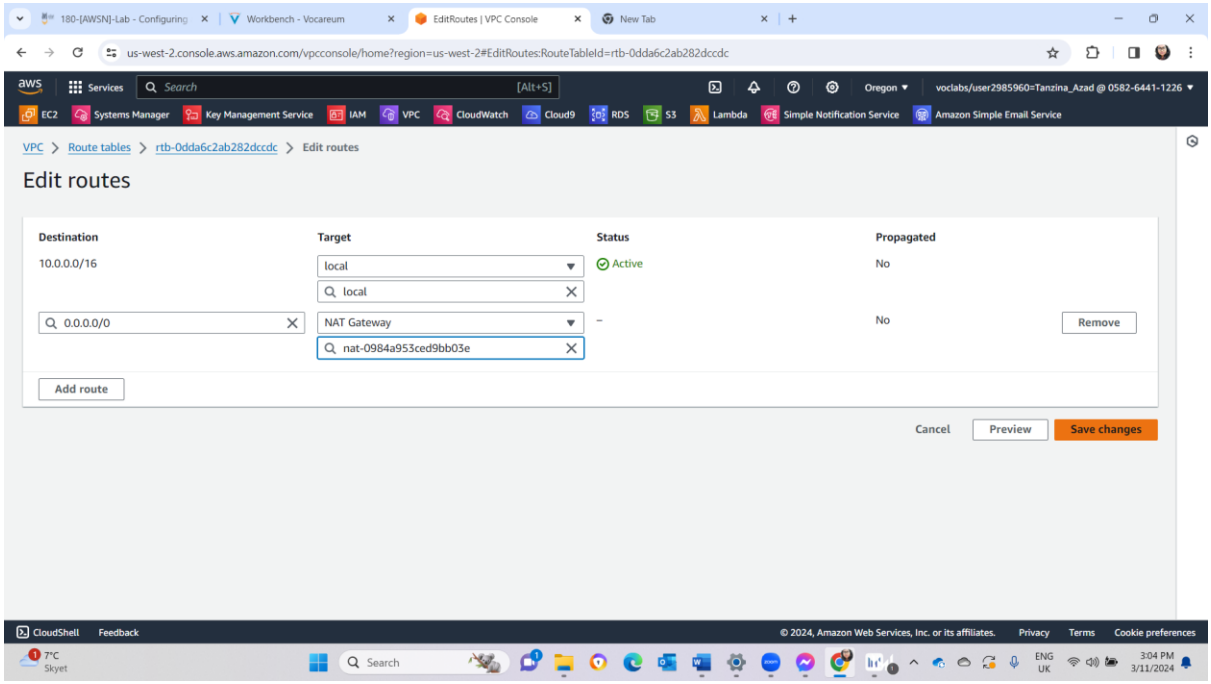
The route table is currently showing only a single entry, which routes traffic locally within the VPC. You add an additional route to send internet-bound traffic through the NAT gateway.

50. Choose **Edit routes**.

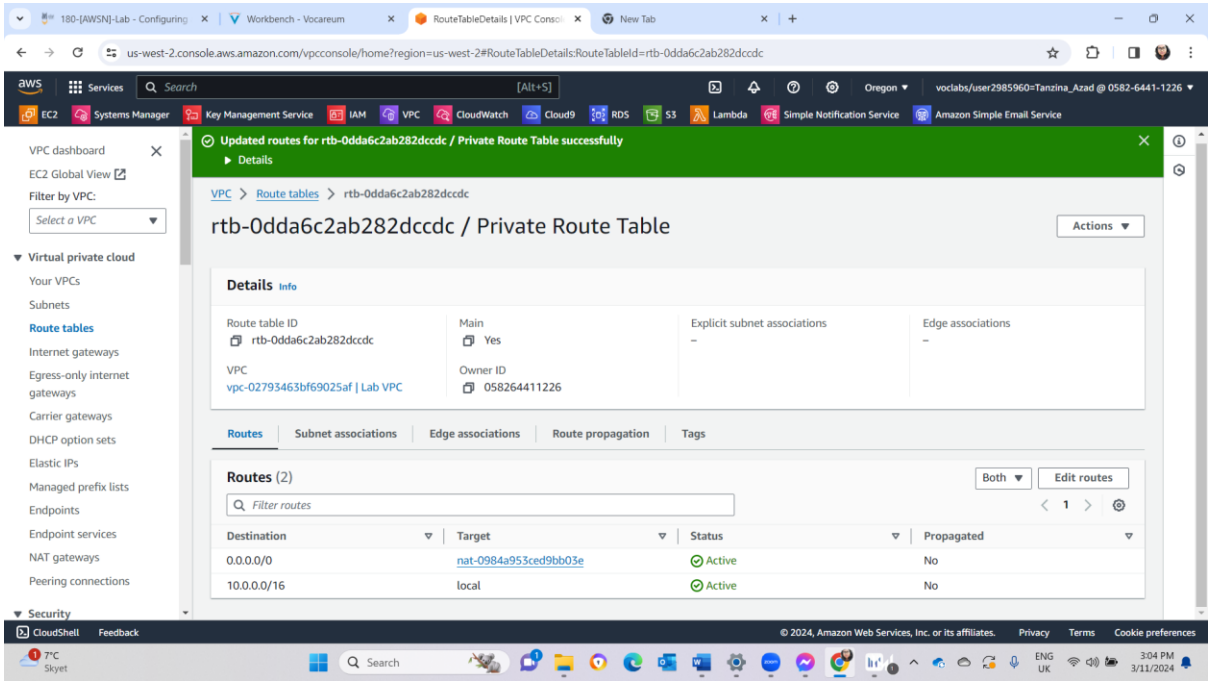


51. Choose **Add route** and configure the following options:

- **Destination:** Enter 0.0.0.0/0.
- **Target:** Choose **NAT Gateway**, and then choose **nat-** from the list.



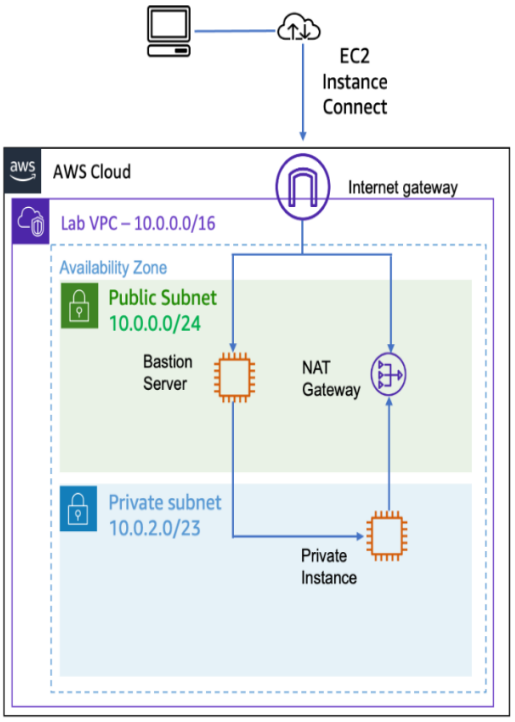
53. Choose **Save changes**.



Resources in the private subnet that wish to communicate with the internet now have their network traffic directed to the NAT gateway, which forwards the request to the internet. Responses flow through the NAT gateway back to the private subnet.

Finally,

- Created a Lab VPC with a private and public subnet, an internet gateway, and a NAT gateway.
- Configured route tables associated with subnets to local and internet-bound traffic by using an internet gateway and a NAT gateway.
- Launched a Bastion server in a public subnet.



Public Route Table

Destination	Target
10.0.0.0/16	Local
0.0.0.0/0	Internet Gateway

Private Route Table

Destination	Target
10.0.0.0/16	Local
0.0.0.0/0	NAT Gateway

End of the project we will get this AWS structure.