**Experiment 4: Deploy load balancer for the application deployed on to distribute the load among multiple instances**

**Aim:** Deploy a load balancer to distribute traffic among multiple instances, ensuring improved application performance and scalability.

**Tools:** AWS

**Learning Objectives:**

* Understand the process of configuring and deploying a load balancer in a cloud environment.
* Learn how to manage traffic distribution across multiple instances to prevent overload and ensure high availability.

**Theory:**

A load balancer is a critical component in modern web architecture, designed to distribute incoming network traffic across multiple servers or instances. The primary goal of load balancing is to ensure that no single server becomes overwhelmed with too much traffic, which can lead to slow performance or even server failure. By spreading the load evenly, load balancers help maintain high availability, reliability, and responsiveness of applications.

**Types of Load Balancers**

There are several types of load balancers, each suited to different needs:

* **Application Load Balancers (ALB):** Operate at the application layer (Layer 7) of the OSI model and can make routing decisions based on content. ALBs are ideal for distributing traffic to web servers, microservices, and API endpoints.
* **Network Load Balancers (NLB):** Operate at the transport layer (Layer 4) and are optimized for handling large amounts of traffic with low latency. NLBs are typically used for load balancing TCP/UDP traffic.
* **Classic Load Balancers:** These provide basic load balancing at both Layer 4 and Layer 7 but with fewer features compared to ALBs and NLBs.

**Benefits of Load Balancing**

* **Scalability:** Load balancers allow you to add or remove servers as needed, enabling your application to scale based on demand.
* **Fault Tolerance:** By distributing traffic across multiple instances, load balancers prevent any single point of failure, enhancing the overall reliability of the application.
* **Improved Performance:** Load balancers ensure that traffic is evenly distributed, preventing any single instance from being overloaded, which helps maintain fast response times.

**Load Balancing Process**

When a client makes a request to an application, the load balancer acts as an intermediary, receiving the request and directing it to one of the available instances based on predefined rules. These rules can be based on factors like round-robin distribution, least connections, or geographic location.

* **Health Checks:** Load balancers continuously monitor the health of instances to ensure that traffic is only directed to servers that are operational. If an instance becomes unhealthy, the load balancer automatically routes traffic to other healthy instances.
* **SSL Termination:** Some load balancers also handle SSL termination, where the secure SSL connection is terminated at the load balancer, and the traffic is forwarded to the backend instances in an unencrypted form. This reduces the processing load on backend servers.

**Auto Scaling and Load Balancers**

Load balancers work hand-in-hand with auto-scaling groups in cloud environments. Auto scaling automatically adjusts the number of active instances in response to traffic demand. When traffic increases, the auto-scaling feature spins up new instances, and the load balancer automatically starts directing traffic to them. This dynamic adjustment ensures cost-efficiency while maintaining high availability and performance.

**Practical Application**

In cloud environments like AWS, load balancers are essential for distributing traffic to instances deployed across multiple availability zones. This not only improves redundancy but also enhances performance by serving user requests from the nearest or least loaded instance.

Deploying a load balancer in this practical session involved configuring it to distribute traffic evenly across multiple instances of the application. This ensures that no single instance is overburdened, and the application can handle increased traffic efficiently.

**Real-World Use Cases**

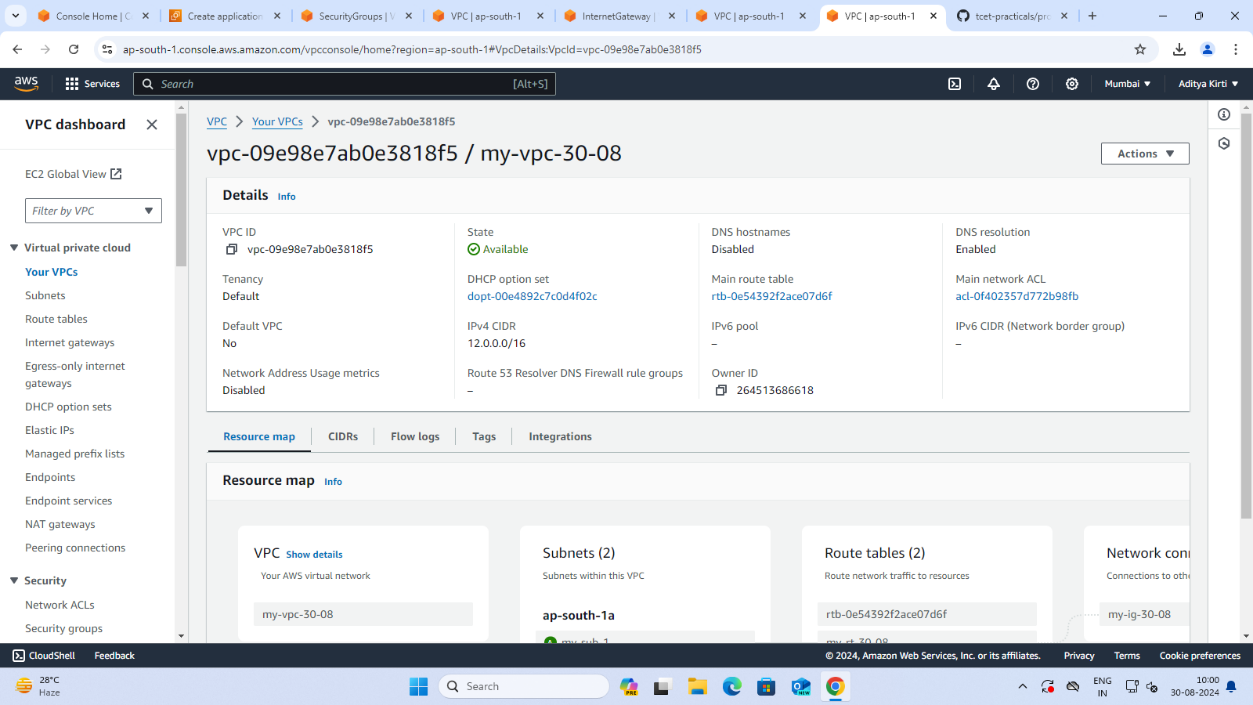
* **E-commerce Platforms:** In high-traffic e-commerce platforms, load balancers are essential for managing large volumes of user requests, especially during peak shopping times like Black Friday. By distributing traffic evenly across multiple servers, load balancers help ensure that the website remains responsive and can handle the surge in demand.
* **Content Delivery Networks (CDNs):** Load balancers are integral to CDNs, where they distribute content requests to servers located in different geographical regions. This reduces latency and ensures that users receive content from the nearest server, improving load times and user experience.

**Learning Outcomes:**

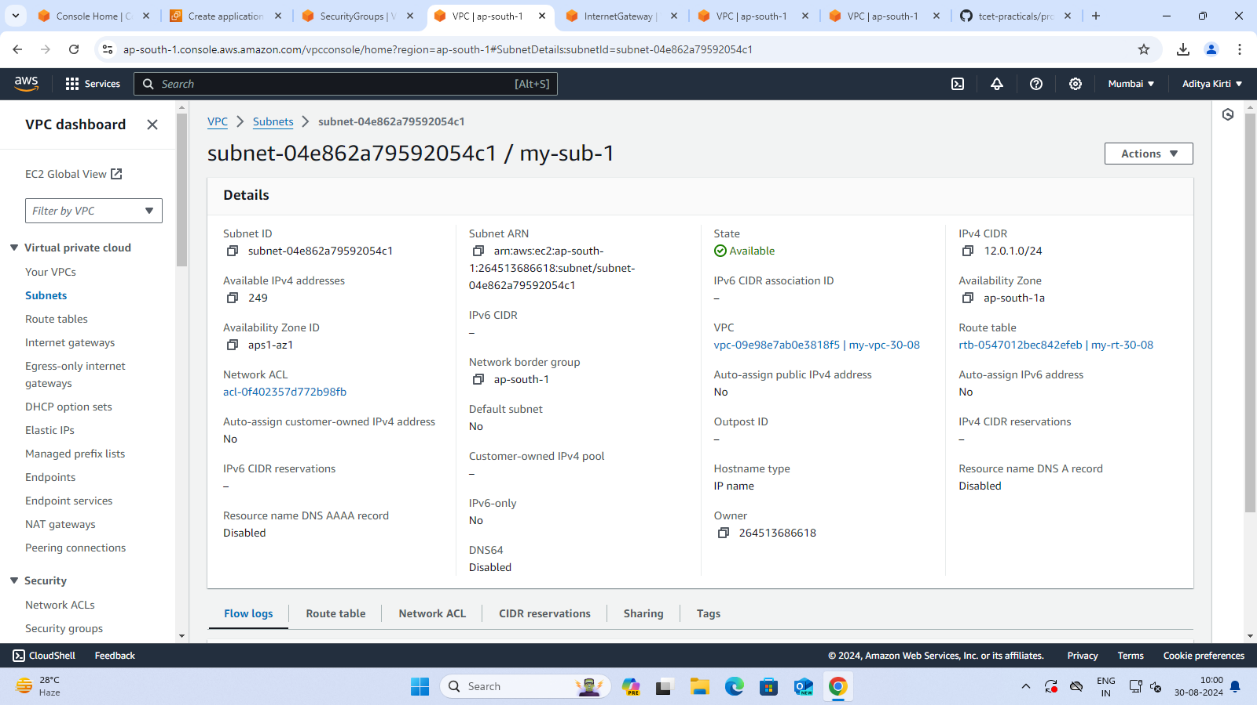
* Ability to deploy and configure a load balancer to effectively distribute traffic.
* Understanding of the benefits of load balancing in enhancing application performance and reliability.

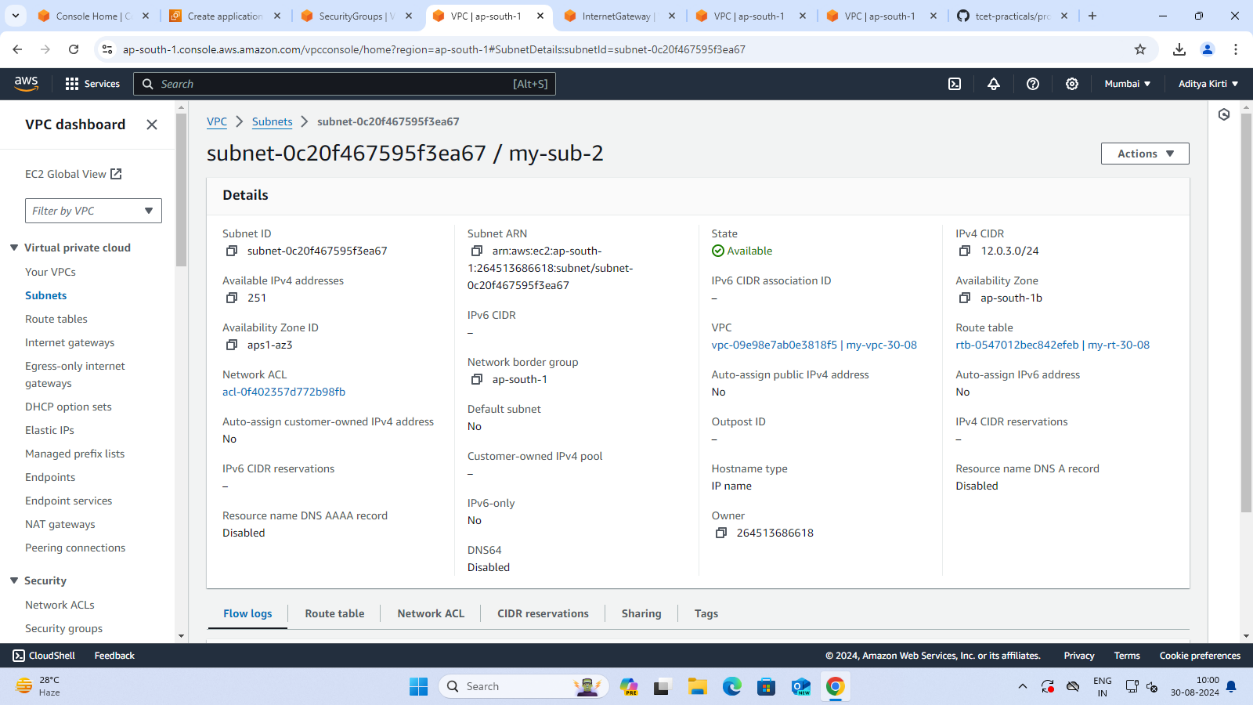
**Implementation:**

1. **VPC Creation:**

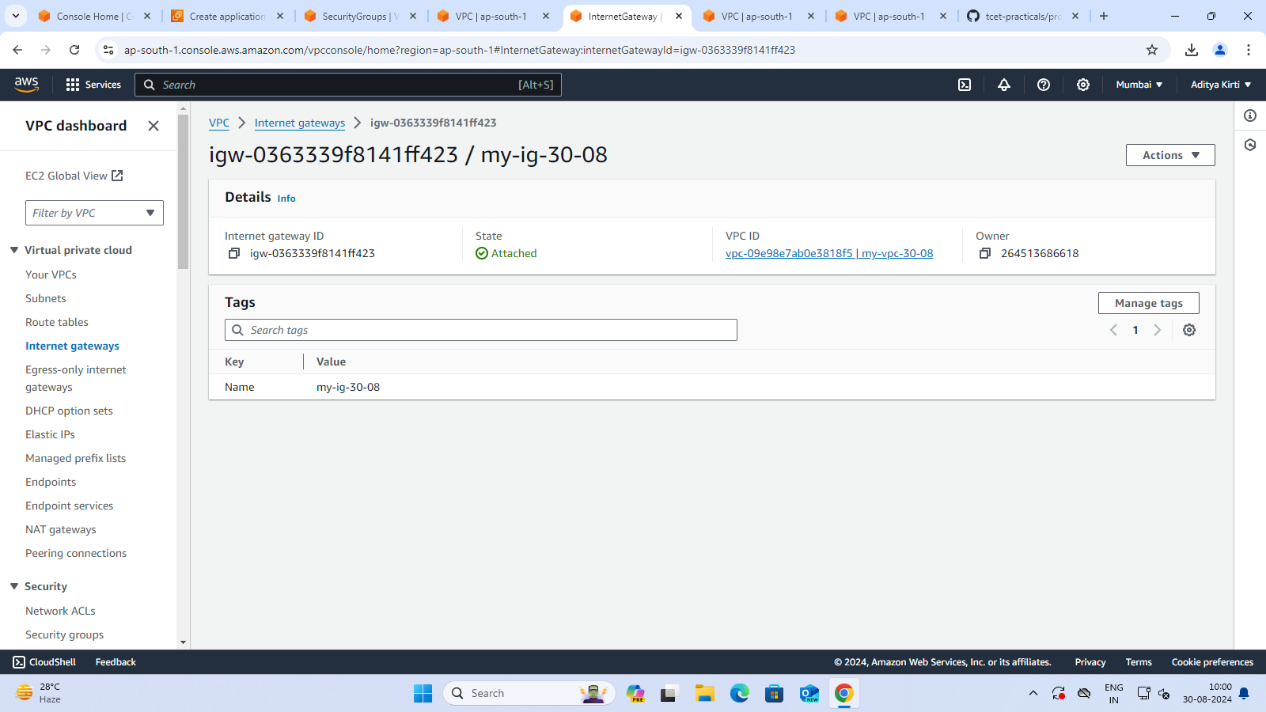
****

1. **Subnets Creation:**

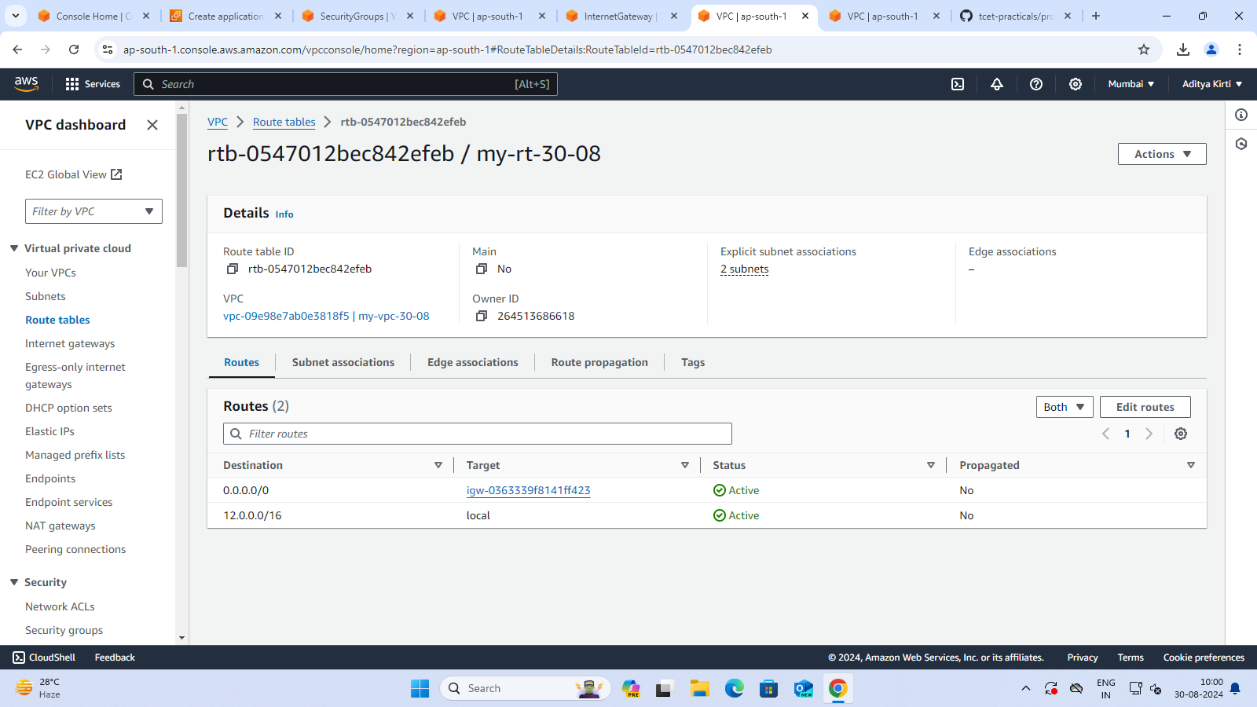
****

****

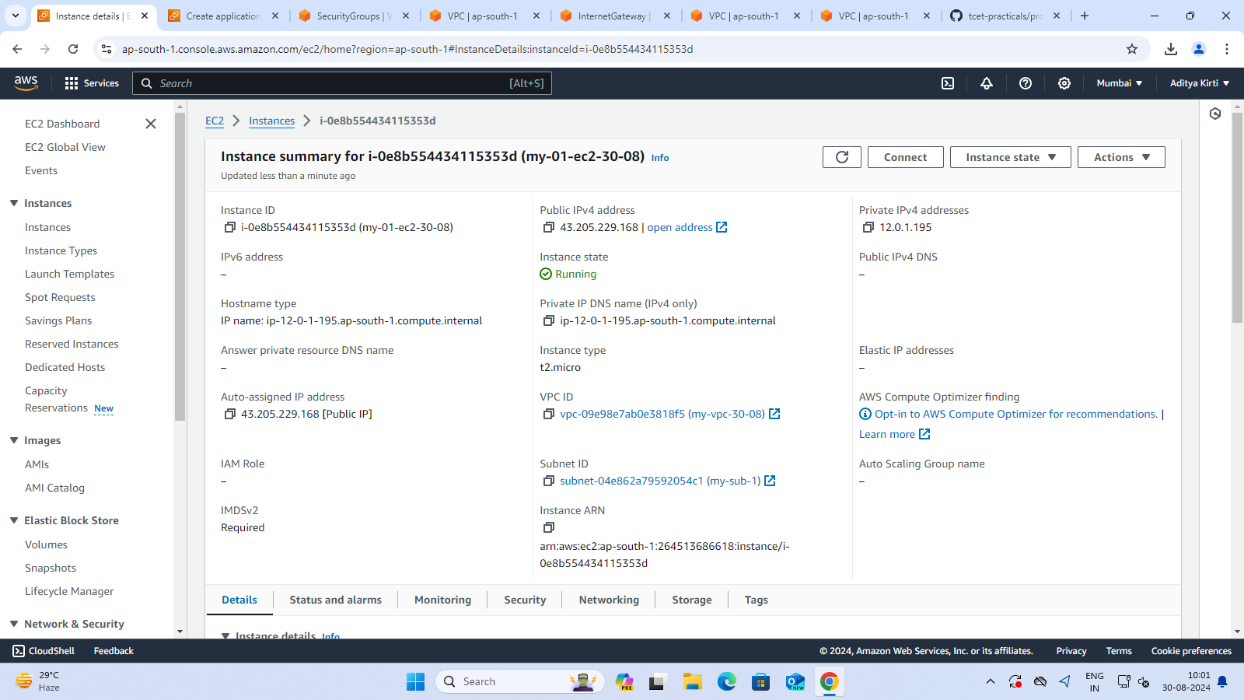
1. **Internet Gateway:**

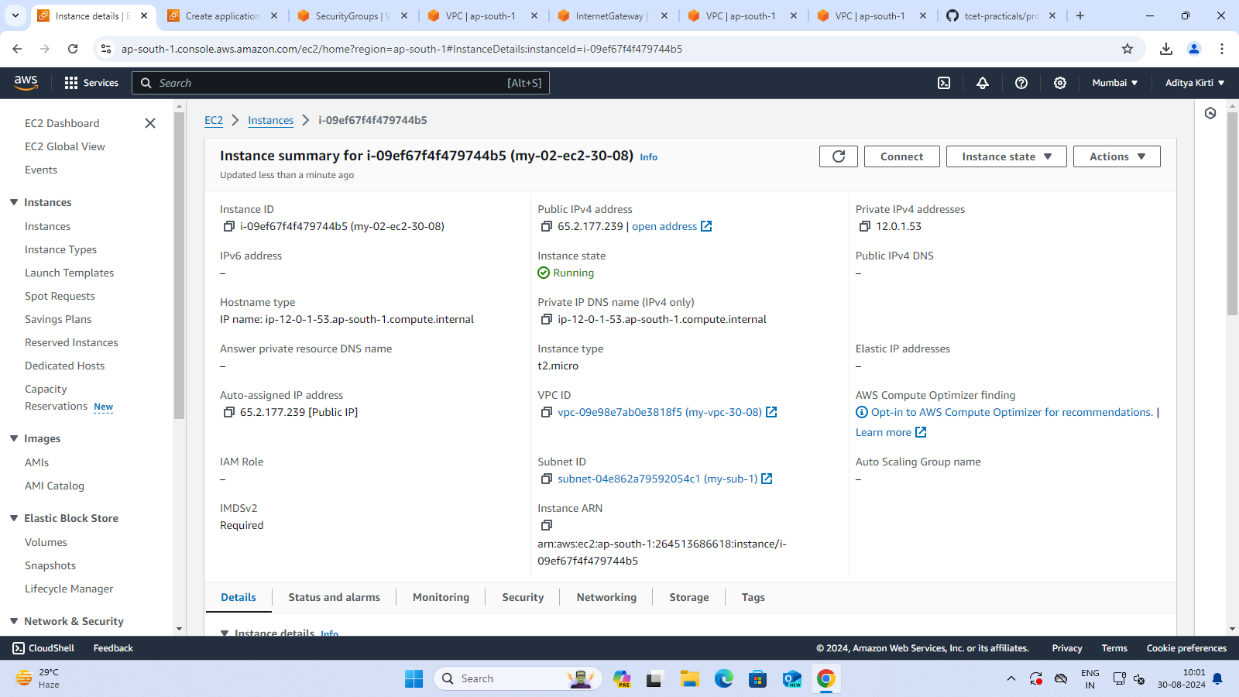
****

1. **Route Table:**

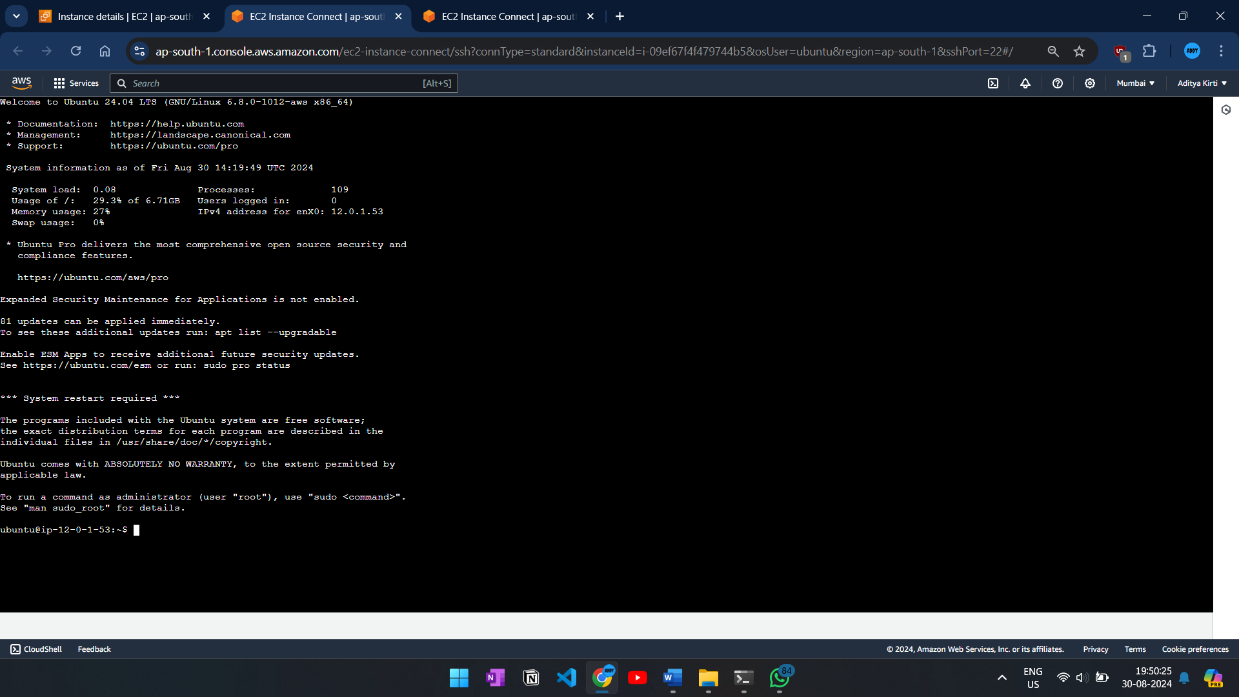
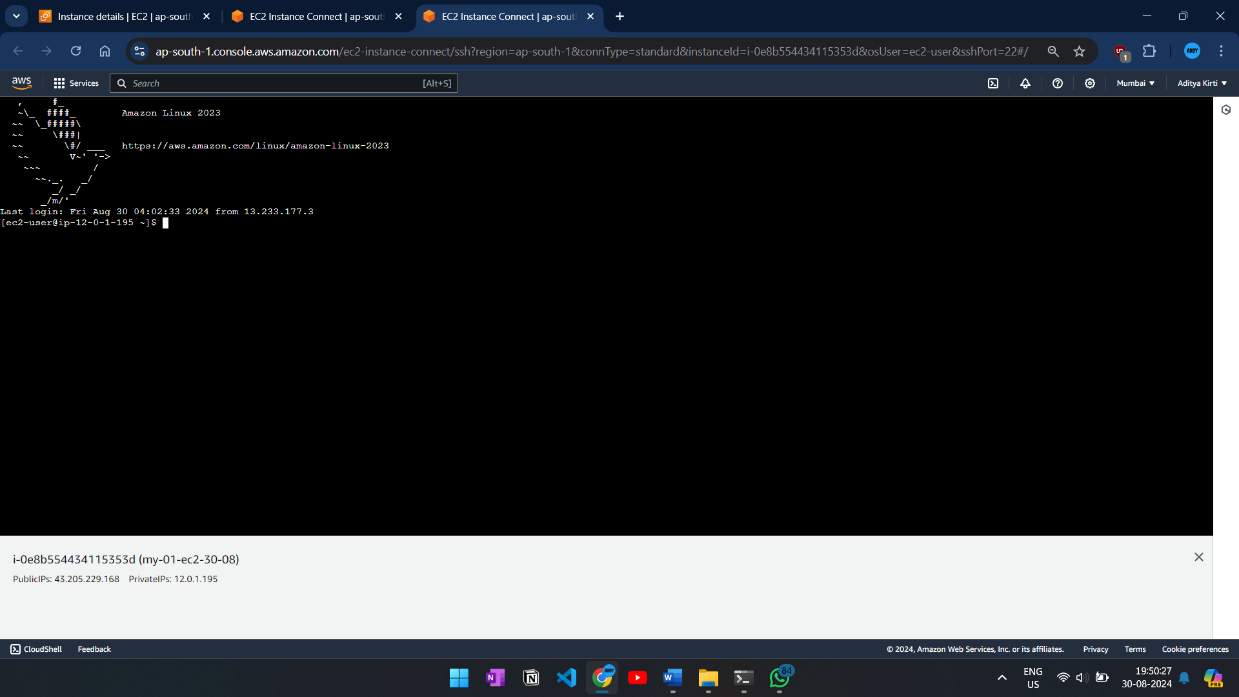
****

1. **EC2 Instances:**

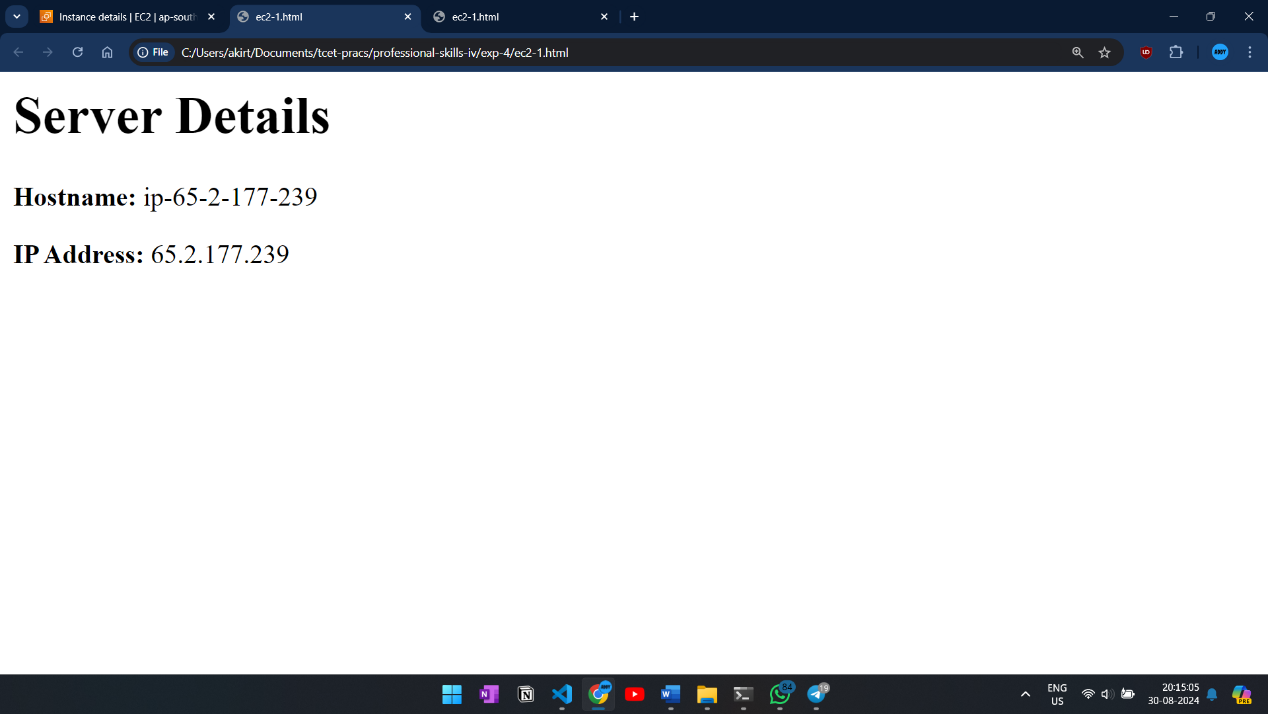
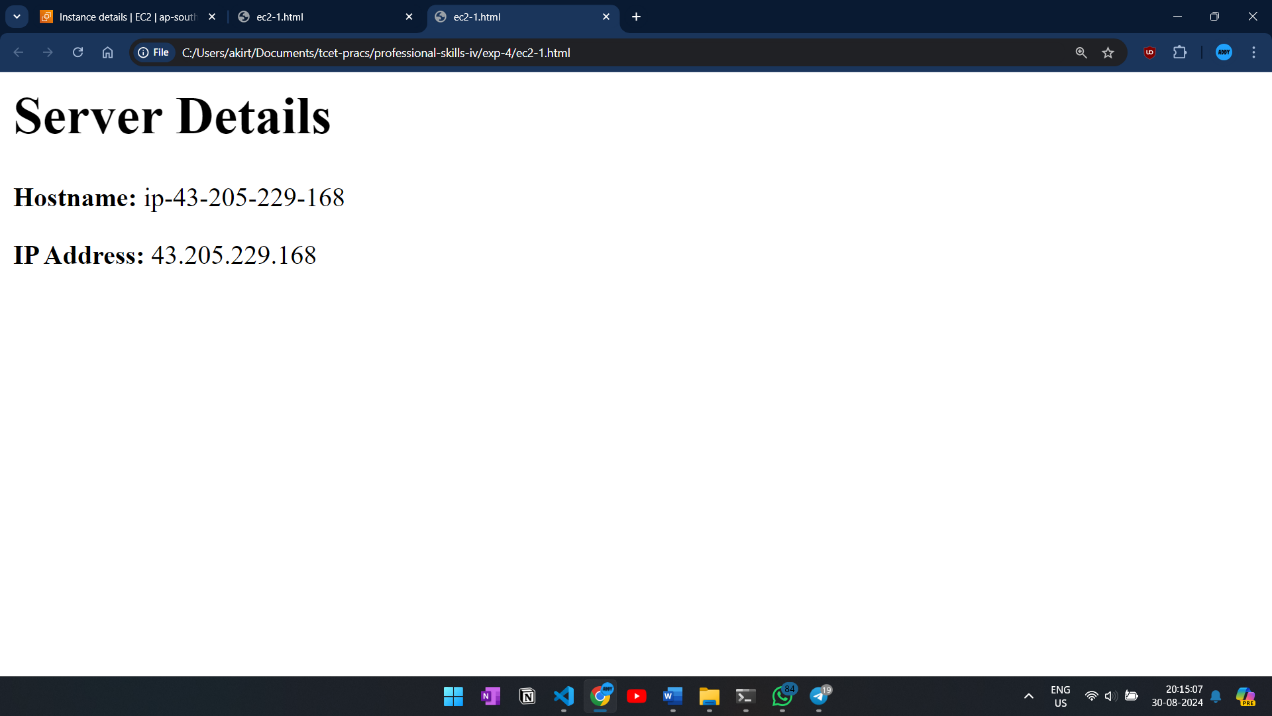
****

****

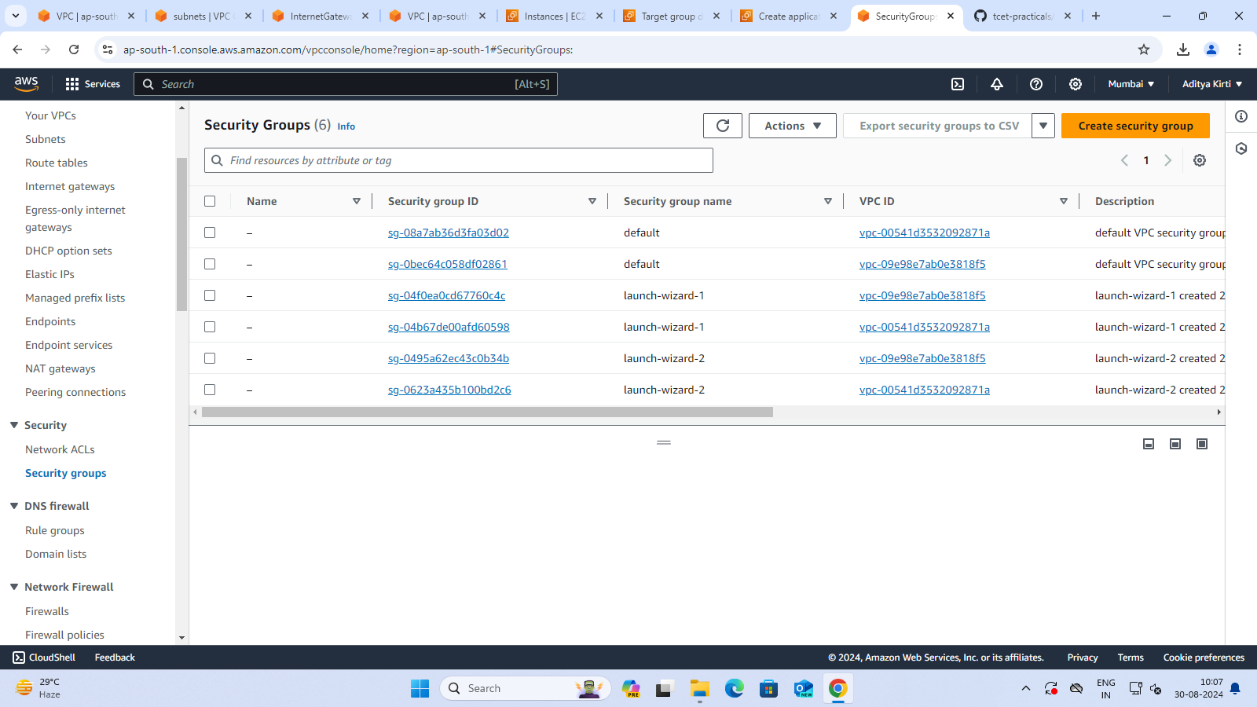
1. **EC2 Instance Connections:**

****

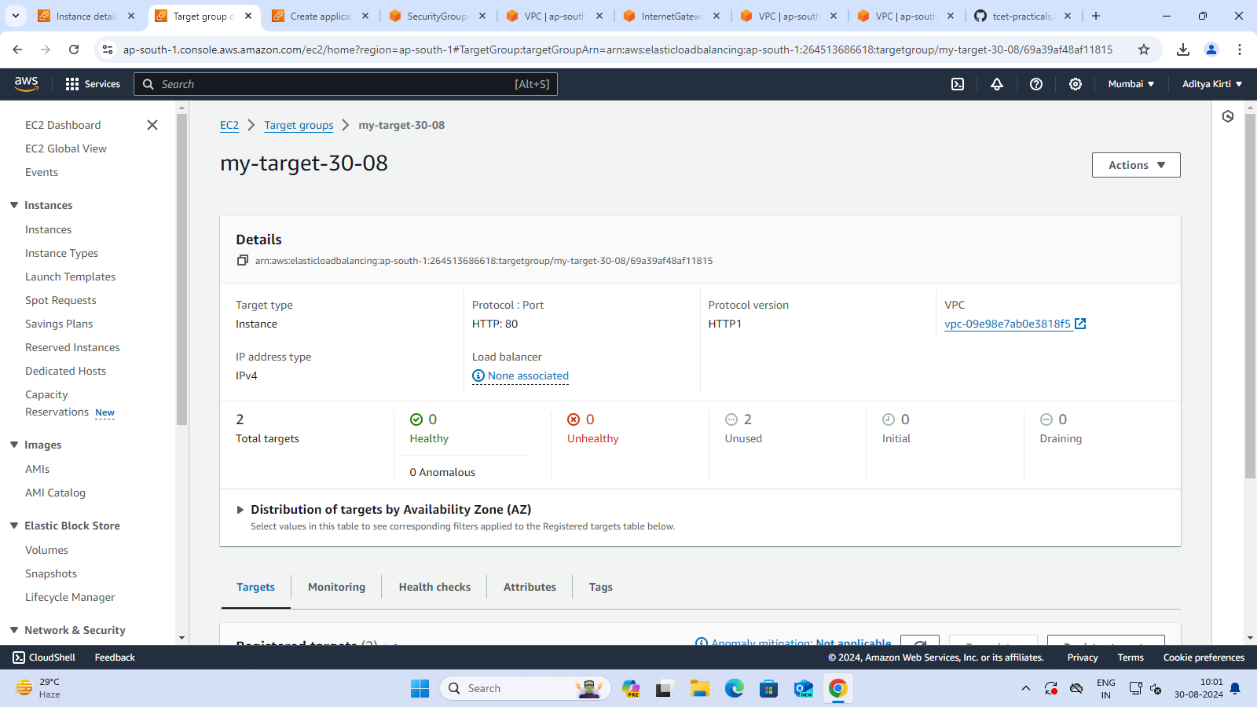
1. **EC2 Instance Webpage:**

****

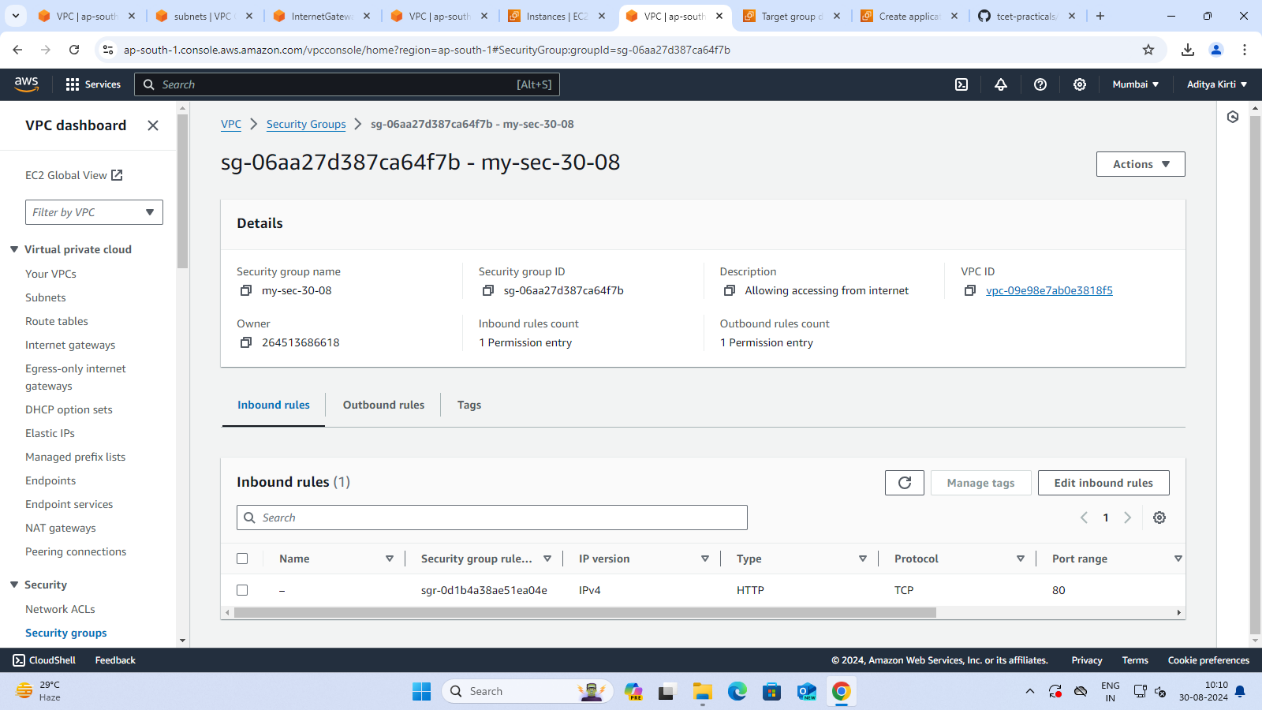
1. **Security Groups for EC2 Instances:**

****

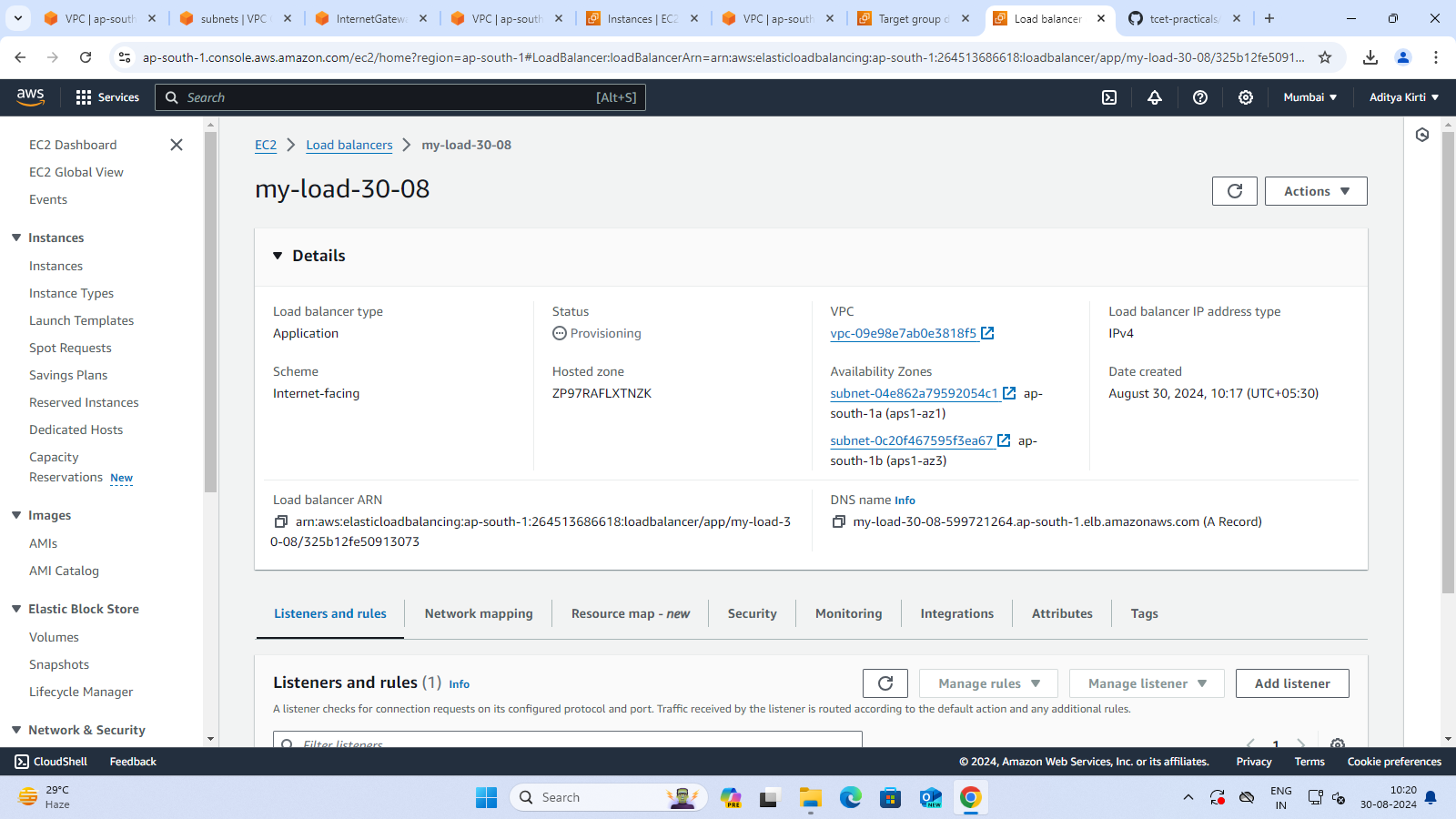
1. **Target Group:**

****

1. **Security Group for Load Balancer:**

****

1. **Load Balancer Creation:**

****