[2CEIT603: CLOUD COMPUTING]

Practical: 6

AIM- create application load balence to balence http traffic using aws elastic load belencning.

Submitted By: Adeshara Brijesh
Enrollment number: 21012021001

Department of Computer Engineering/Information Technology

Introduction:

- Elastic Load Balancing automatically distributes your incoming traffic across multiple targets, such as EC2 instances, containers, and IP addresses, in one or more Availability Zones.
- It monitors the health of its registered targets, and routes traffic only to the healthy targets. Elastic Load Balancing scales your load balancer as your incoming traffic changes over time.
- It can automatically scale to the vast majority of workloads.

Purpose:

Distribute Traffic

The Application Load Balancer distributes incoming application traffic across multiple targets, such as EC2 instances, to improve the availability and scalability of your applications.

Advanced Routing

It provides advanced routing capabilities, allowing you to route traffic based on the content of the request, such as the URL path or the query string parameters.

Health Checking

The load balancer continuously monitors the health of your targets, automatically routing traffic away from unhealthy instances and ensuring high availability.

Components:

- ### Load balancers

  A load balancer serves as the single point of contact for clients.

  Clients send requests to the load balancer, and the load balancer sends them to targets, such as EC2 instances.

- ### Listeners

  A listener is a process that checks for connection requests, using the protocol and port that you configure.

  Before you start using your Application Load Balancer, you must add at least one listener.

- ### Target groups

- Target groups route requests to individual registered targets, such as EC2 instances, using the protocol and port number that you specify.

You can register a target with multiple target groups.

➢ Architecture:

- **Client**

The client sends a request to the load balancer's DNS name.

- **Load Balancer**

The load balancer receives the request and routes it to the appropriate target group based on the configured rules.

- **Targets**

The load balancer distributes the request to the healthy targets within the target group, such as EC2 instances or containers.

➢ Pros/Cons:

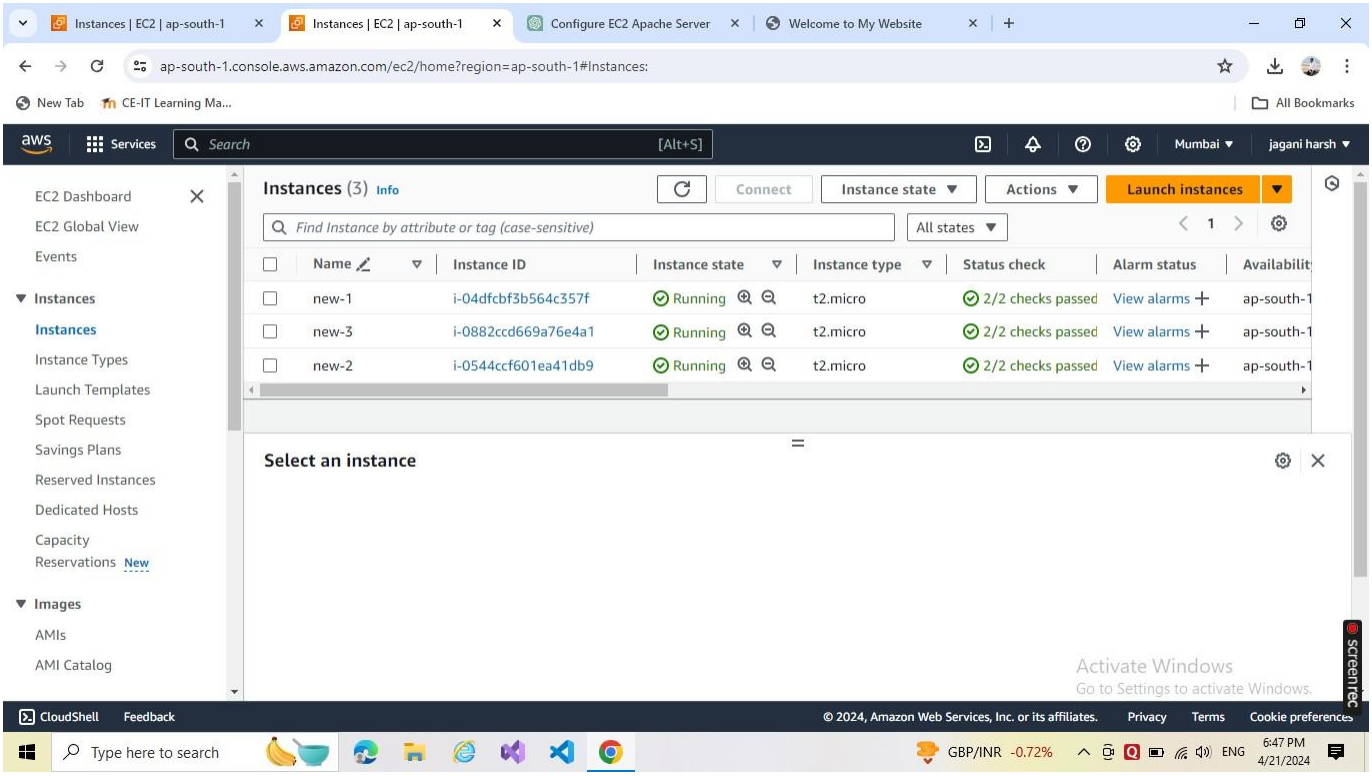
- **Pros**:
  - Improve uptime.
  - Scale Application.
  - Boost network and application performance.
- **Cons**:
  - Geographic limitations in certain load-balancing algorithms.
  - Possibility of assigning only a single point of failure.
  - Delays due to overly complex algorithms.

➢ Implementation Steps:

Step 1: Design Your Architecture

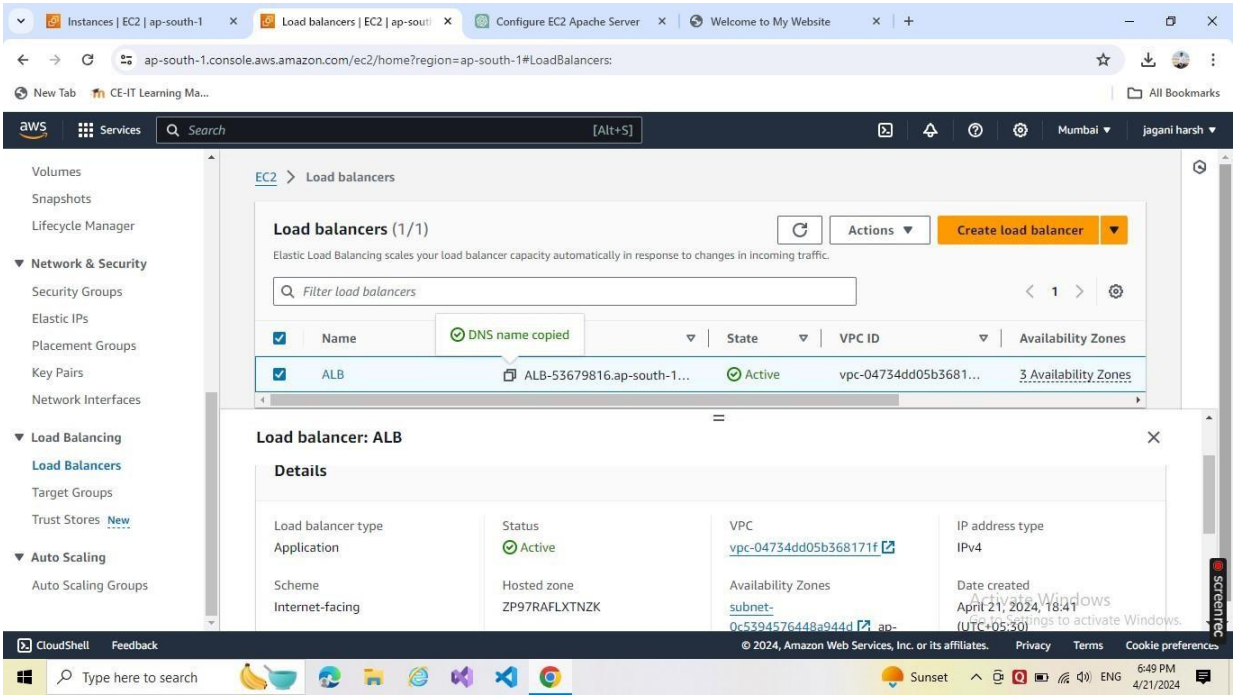
➢ **Define Your Requirements**: Determine what you need from your load balancer, including routing strategies (path-based, host-based, etc.), health check requirements, and target types (EC2 instances, microservices, containers, etc.).

- **Identify Your Resources**: List the resources that will be placed behind the load balancer, including details about compute resources, network configurations, and the ports and protocols used.



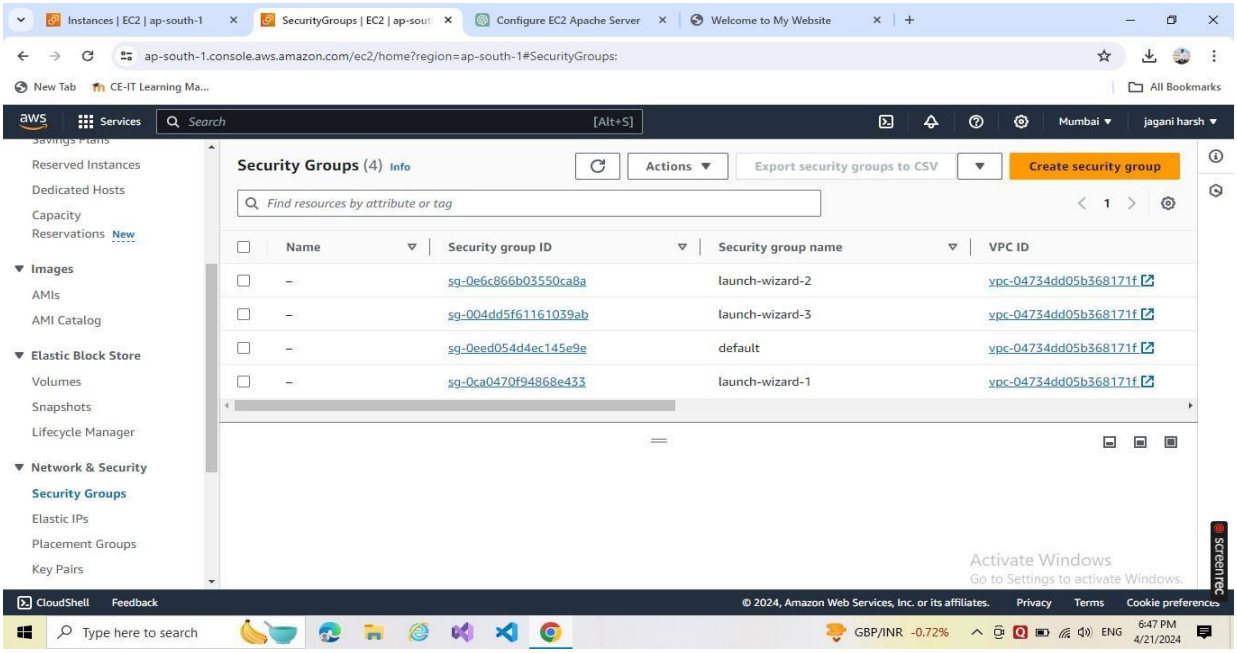
Step 2: Create the Application Load Balancer

Log in to the AWS Management Console.Navigate to the EC2 Dashboard, and under the Load Balancing section, choose Load Balancers.
Click Create Load Balancer and select Application Load Balancer.
Configure Basic Settings: Name your load balancer.



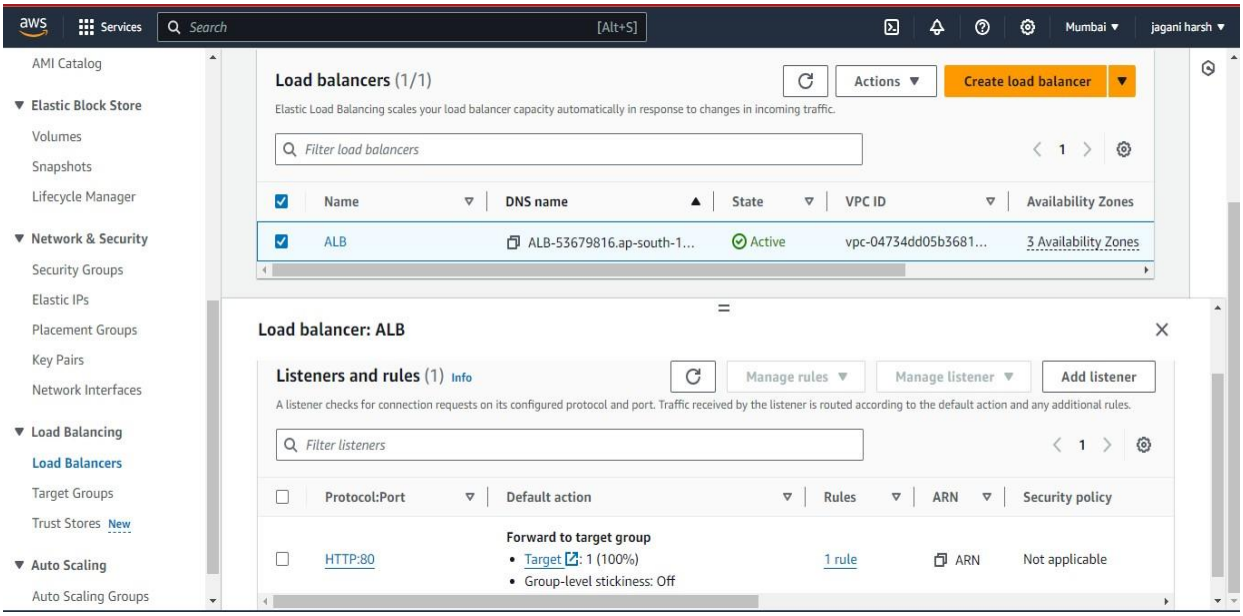
Step 3: Configure Security Groups

Assign Security Groups: Create or select a security group that allows the appropriate inbound traffic (usually HTTP on port 80 and HTTPS on port 443) and the necessary outbound rules.



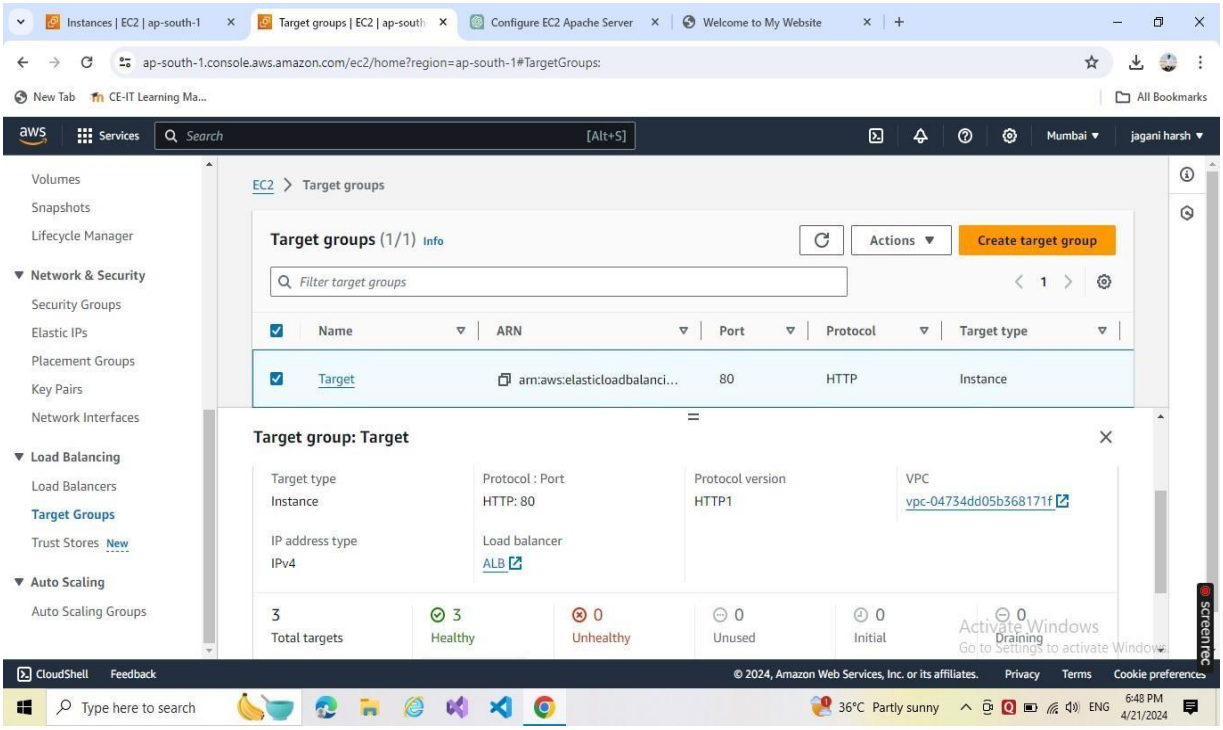
Step 4: Configure Listeners and Routing

- **Set Up Listeners**: A listener checks for connection requests, using the protocol and port that you configure. Set up listeners for HTTP and, if needed, HTTPS.
- **Define Routing Rules**: Create rules that define how the incoming traffic should be routed to the target groups.
- **Target Groups:** Create target groups, which are sets of similar targets. Specify protocol and port for targets.
- **Routing Rules**: Specify path or host-based rules that direct the traffic to different target groups based on the URL path or hostname.



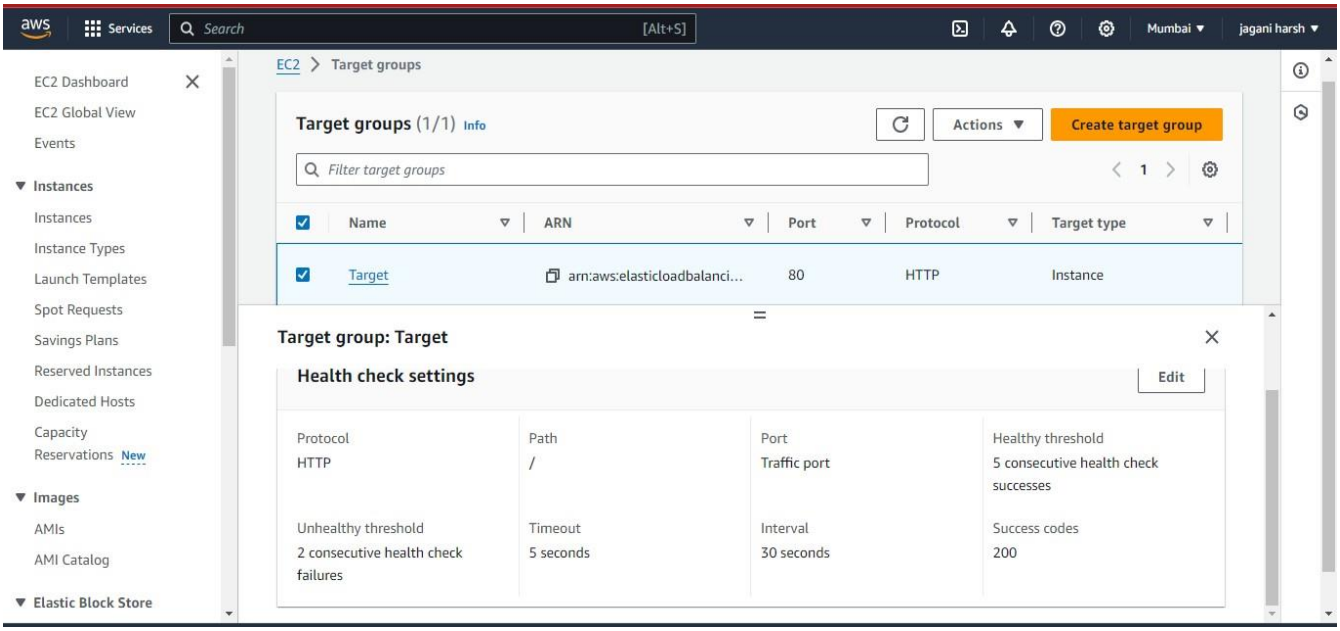
Step 5: Register Targets

- **Add Targets to the Target Groups**: Manually add EC2 instances or other supported targets to each target group. Alternatively, if using services like ECS, these might be managed automatically.



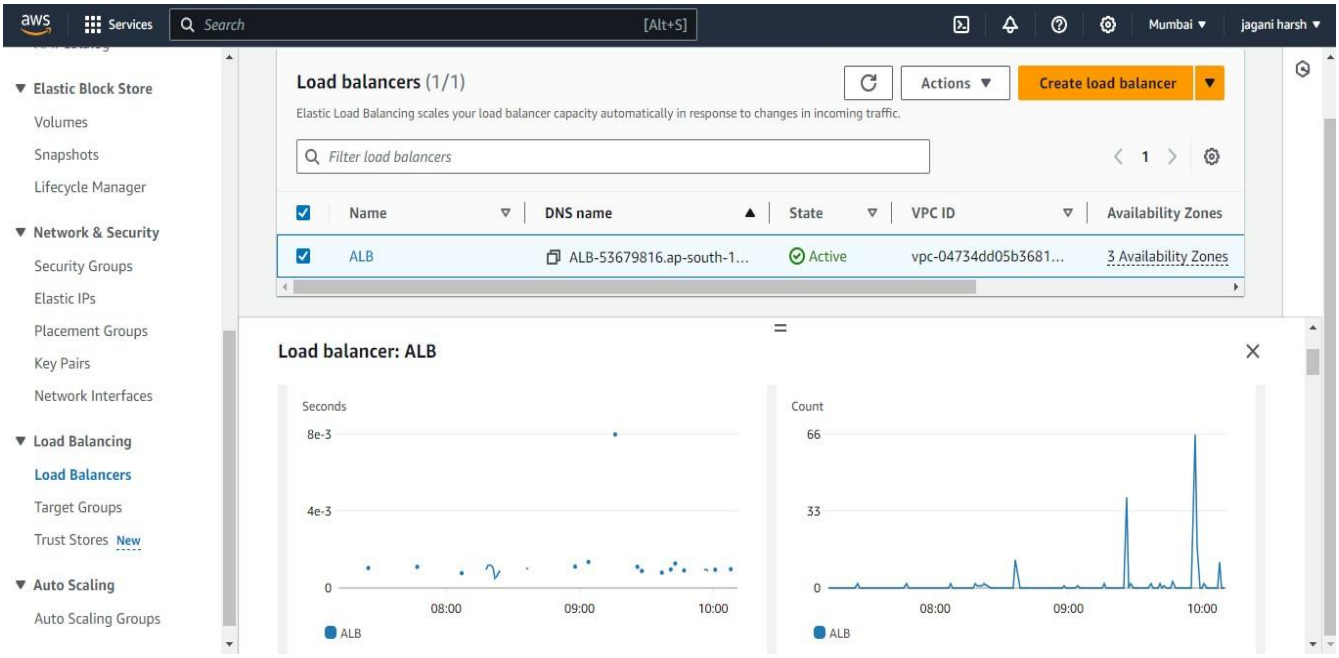
Step 6: Configure Health Checks

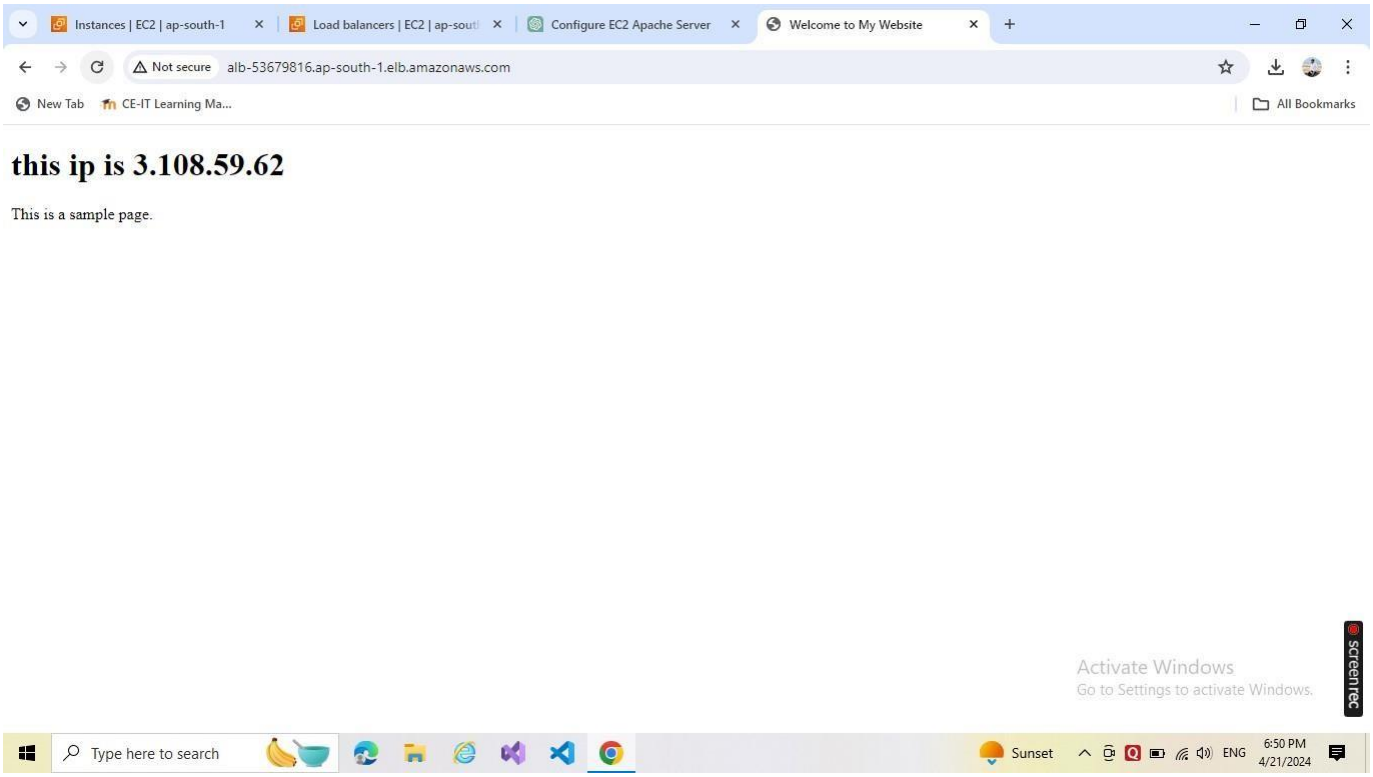
- **Set Up Health Checks for Target Groups:** Configure health checks to ensure traffic is routed only to healthy targets. Define the health check protocol, path, and other parameters.

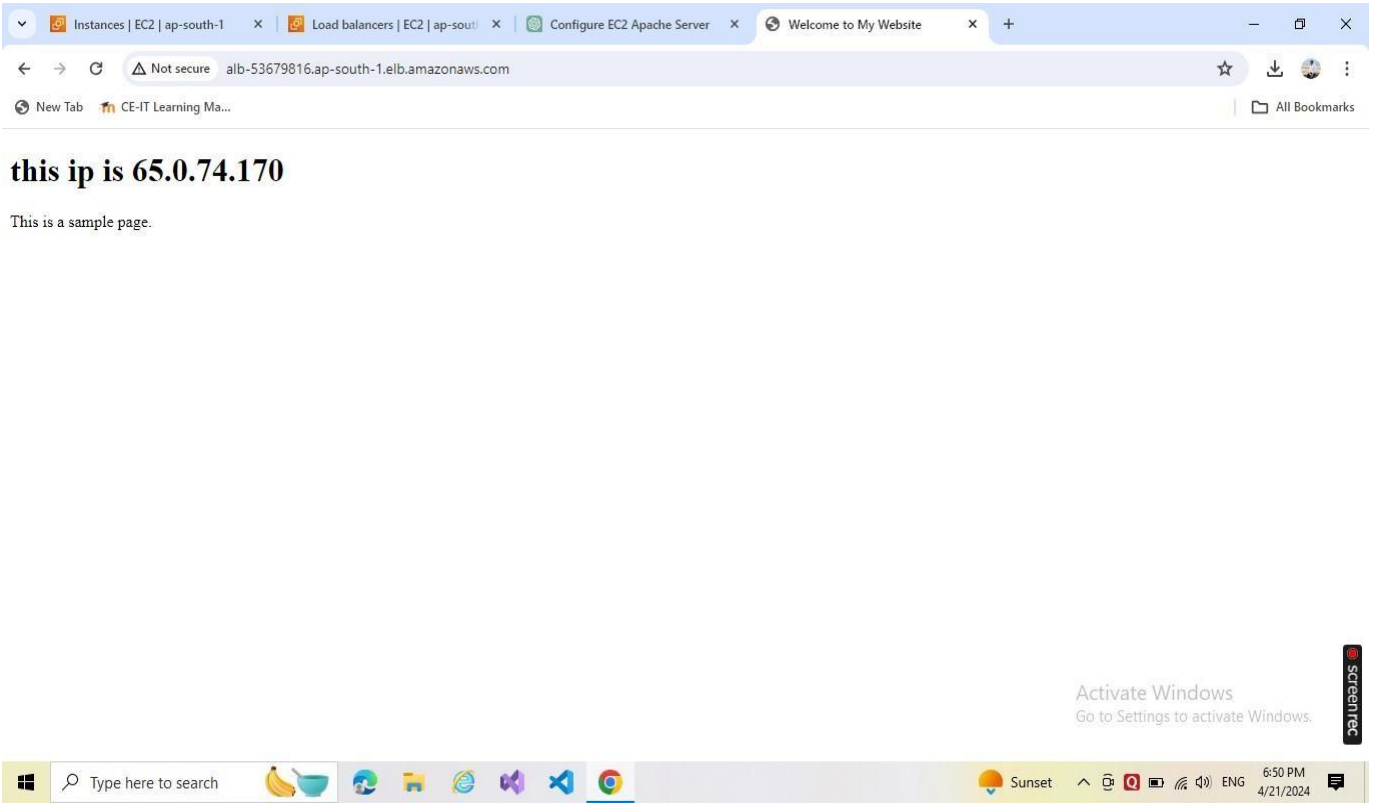


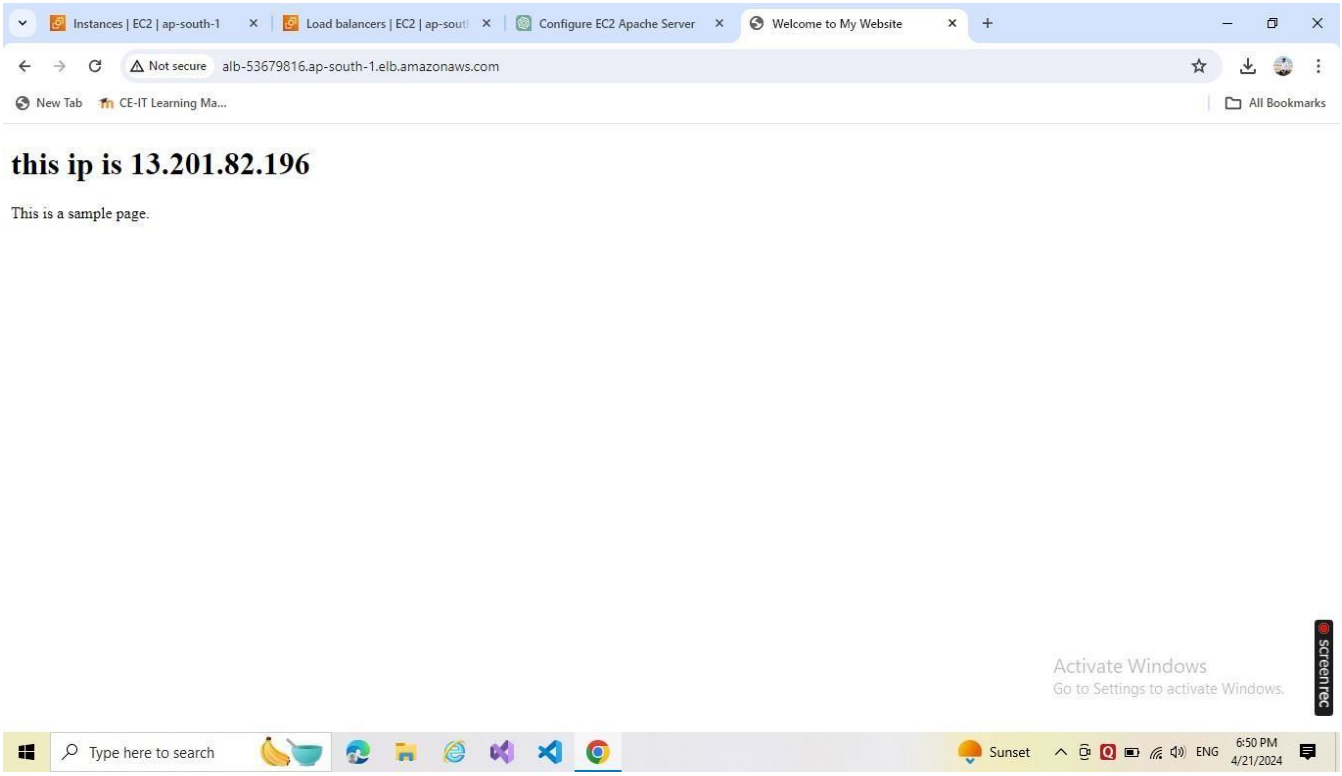
Step 7: Test and Validate

- **Test the Load Balancer**: After setting up the load balancer, send requests to its DNS name to ensure it properly routes traffic to the target groups according to your rules.
- **Monitor and Optimize**: Utilize AWS CloudWatch to monitor performance and logs to troubleshoot and optimize the configuration.




Instances | EC2 | ap-south-1
Load balancers | EC2 | ap-sout
Configure EC2 Apache Server
Welcome to My Website
Not secure
alb-53679816.ap-south-1.elb.amazonaws.com
New Tab
CE-IT Learning Ma...
All Bookmarks
this ip is 3.108.59.62
This is a sample page.
Activate Windows
Go to Settings to activate Windows.
screenrec
Type here to search
Sunset
ENG
6:50 PM
4/21/2024


Instances | EC2 | ap-south-1
Load balancers | EC2 | ap-sout
Configure EC2 Apache Server
Welcome to My Website
Not secure
alb-53679816.ap-south-1.elb.amazonaws.com
New Tab
CE-IT Learning Ma...
All Bookmarks
this ip is 65.0.74.170
This is a sample page.
Activate Windows
Go to Settings to activate Windows.
screenrec
Type here to search
Sunset
ENG
6:50 PM
4/21/2024



Step 8: Deployment and Maintenance

Deploy: Once tested, deploy your application and begin directing production traffic to the ALB.

Maintain and Scale: Regularly update the health checks, security policies, and add/remove targets based on demand and capacity.

- **Comparison:**

  - **Application Load Balancer**

  - Offers advanced routing capabilities, HTTP/HTTPS support, and health monitoring, making it suitable for modern, sophisticated web applications.

  - **Classic Load Balancer**

  - Provides basic load balancing capabilities, suitable for simpler web applications or legacy systems, with limited routing options.

  - **Network Load Balancer**

  - Specialized for high-performance, low-latency traffic, ideal for TCP-based applications

    or microservices, but with less advanced routing options.

➢ Conclusion:

- AWS Elastic Load Balancing with the Application Load Balancer offers a powerful and flexible solution for distributing traffic to your web applications, providing advanced routing, health monitoring, and automatic scaling to ensure high availability and optimal performance.