1. Describe Load Balancing and its significance in Cloud Environment

Load Balancing and its Significance in Cloud Environment

Load Balancing:

Load balancing is the process of distributing network or application traffic across multiple servers to ensure no single server becomes overwhelmed, which helps maintain system reliability, availability, and performance. By evenly distributing the load, it ensures that resources are used efficiently, minimizes response time, and avoids overload on any single resource.

Significance in Cloud Environment:

1. **Scalability:**
    - **Horizontal Scaling:** Load balancers facilitate horizontal scaling by distributing traffic across multiple instances of applications or services. This allows cloud environments to handle increasing loads by adding more servers without degrading performance.
2. **Availability and Reliability:**
    - **High Availability:** By distributing traffic, load balancers prevent single points of failure, ensuring continuous availability of services. If one server fails, the load balancer redirects traffic to healthy servers.
    - **Failover:** Load balancers can detect unhealthy servers and automatically reroute traffic to healthy ones, maintaining service reliability.
3. **Performance:**
    - **Optimal Resource Utilization:** Load balancing ensures that no server is overburdened, leading to better utilization of available resources and improved application performance.
    - **Reduced Latency:** By directing traffic to the nearest or least busy servers, load balancers can reduce response times and improve user experience.
4. **Flexibility and Maintenance:**
    - **Zero-Downtime Maintenance:** Load balancers enable rolling updates and maintenance of servers without downtime. Traffic can be directed away from servers that are being updated or maintained.
    - **Session Persistence:** Load balancers can ensure that user sessions are consistently routed to the same server, which is important for applications requiring session persistence.
5. **Security:**
    - **DDoS Mitigation:** Load balancers can distribute the effects of Distributed Denial of Service (DDoS) attacks across multiple servers, mitigating the impact on any single server.
    - **SSL Termination:** Load balancers can offload SSL decryption from application servers, improving performance and simplifying certificate management.

2. List the Load Balancing Service available in AWS, Azure and GCP.

Load Balancing Services in AWS, Azure, and GCP

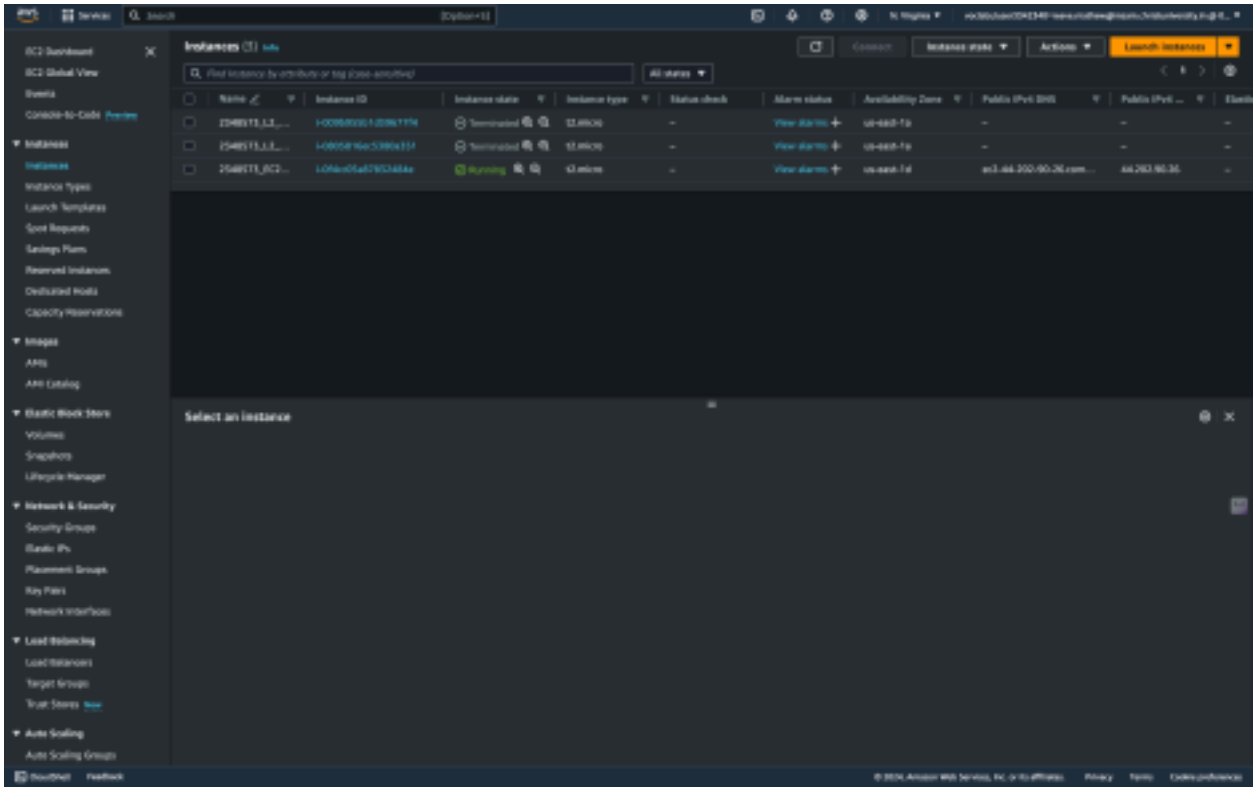
1. **Amazon Web Services (AWS):**
    - **Elastic Load Balancing (ELB):**
        - **Application Load Balancer (ALB):** Designed for web applications, it operates at the application layer (Layer 7) and provides advanced routing based on HTTP/HTTPS requests.
        - **Network Load Balancer (NLB):** Operates at the transport layer (Layer 4) and is optimized for handling high-volume TCP/UDP traffic with ultra-low latency.
        - **Classic Load Balancer (CLB):** Supports both Layer 4 and Layer 7 traffic and is used for applications that were built within the EC2-Classic network.
    - **Global Accelerator:** Provides static IP addresses that act as a fixed entry point to your applications, improving availability and performance by routing traffic through AWS global network.
2. **Microsoft Azure:**
    - **Azure Load Balancer:**
        - **Basic Load Balancer:** Suitable for small-scale, non-critical applications, supporting up to 100 endpoints.
        - **Standard Load Balancer:** Provides higher scale, resiliency, and features like zone redundancy and diagnostic insights.
    - **Azure Application Gateway:** A web traffic load balancer that operates at Layer 7 and includes features such as SSL termination, cookie-based session affinity, and application firewall capabilities.
    - **Azure Traffic Manager:** A DNS-based traffic load balancer that enables distribution of traffic across multiple regions for high availability and responsiveness.
3. **Google Cloud Platform (GCP):**
    - **Google Cloud Load Balancing:**
        - **HTTP(S) Load Balancing:** A global load balancer that distributes HTTP and HTTPS traffic across multiple backend services. It supports advanced features like URL map-based routing and SSL offloading.
        - **SSL Proxy Load Balancing:** Provides global load balancing for SSL traffic.
        - **TCP Proxy Load Balancing:** Provides global load balancing for non-HTTPS TCP traffic.
        - **Internal Load Balancing:** Distributes traffic within a Google Cloud VPC, used for internal services within the cloud environment.

Summary

Load balancing is crucial in cloud environments for ensuring scalability, availability, performance, flexibility, and security of applications and services. Each major cloud provider (AWS, Azure, GCP) offers a range of load balancing services tailored to different needs and traffic types, enabling efficient and reliable management of workloads in the cloud.

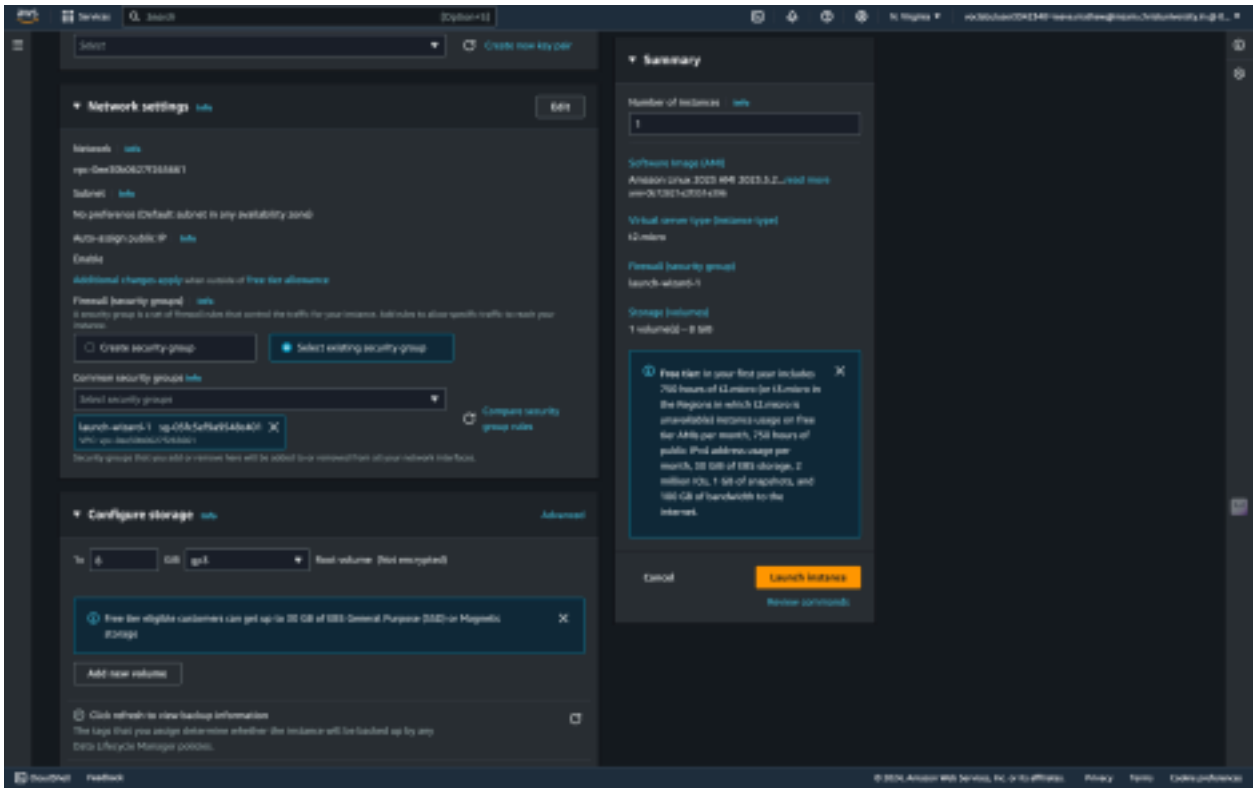
3) Create an AWS EC2 / GCP VM Instances (Instance Name: Regno_EC2_VM1, Regno_EC2_VM2) and install a
webserver of your choice in each of the instances to host web site of your organization globally.
4) Create an Application Load Balancer to ensure the fare allocation of tasks among the web servers deployed on the
Virtual machine instances.

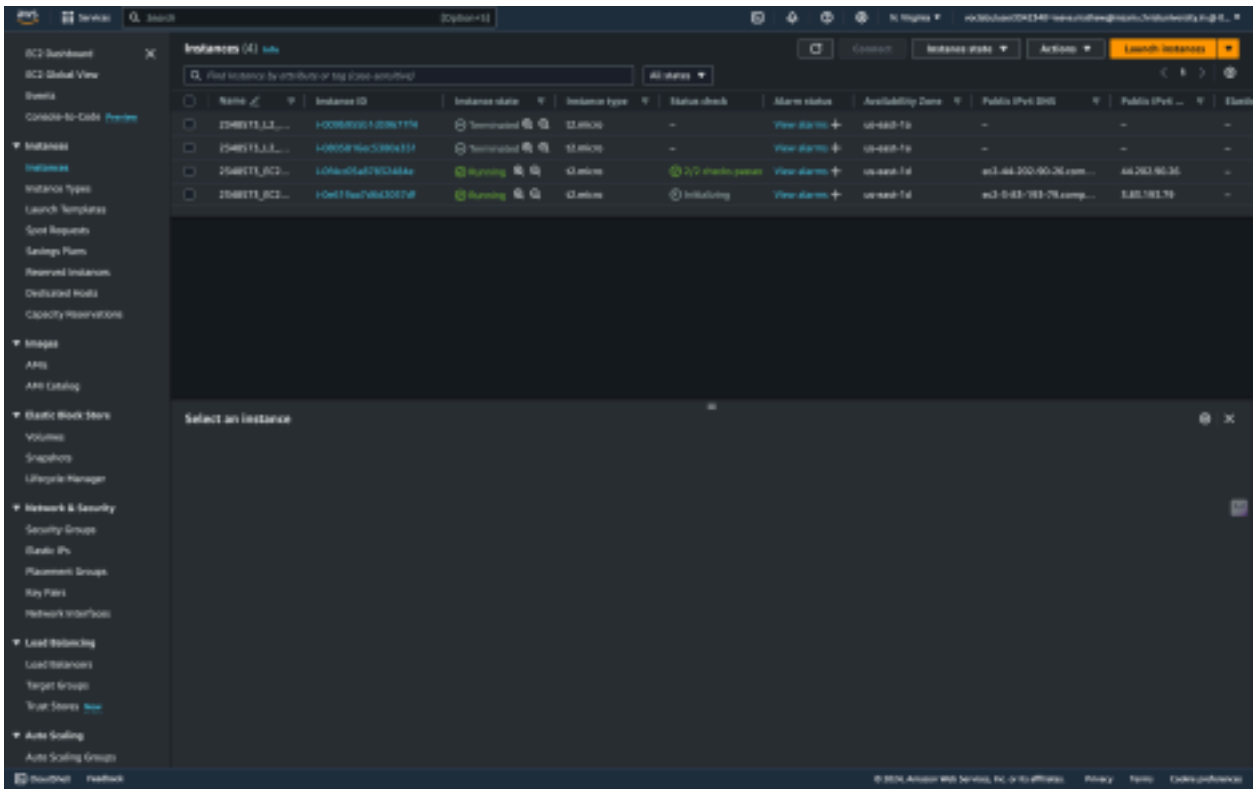
Solution:
Step 1: Navigate to EC2 homepage of AWS console.



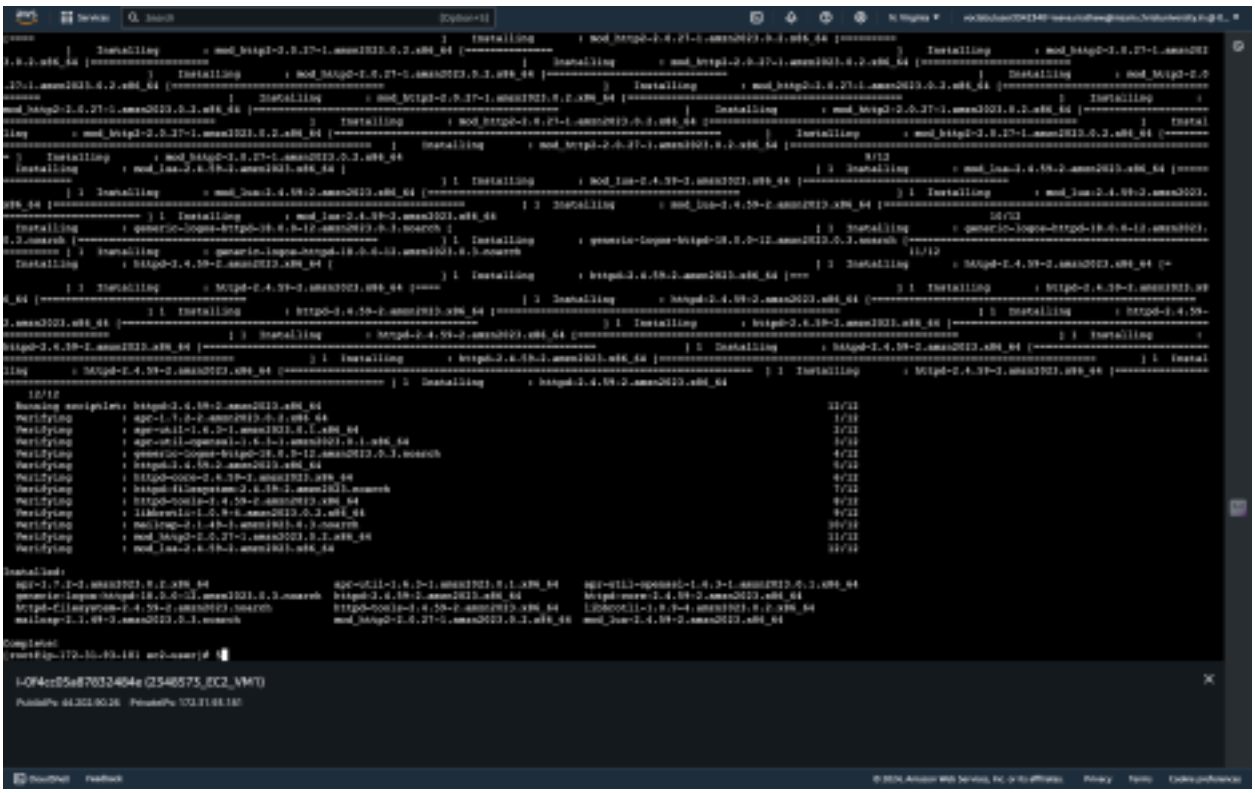
Step 2: Click on Launch Instances and Create 2 identical EC2 instances with having the same security group
EC2 Configurations:

i) No. of instances = 2
ii) Name: 2348573_EC2_VM1 and 2348573_EC2_VM2
iii) AMI: Amazon Linux 2023 AMI
iv) Instance Type: t2.micro
v) Key Pair: vockey
vi) Network Settings
a. Click on Existing Security Group
b. Select launch wizard 1
vii) Click on Launch Instances

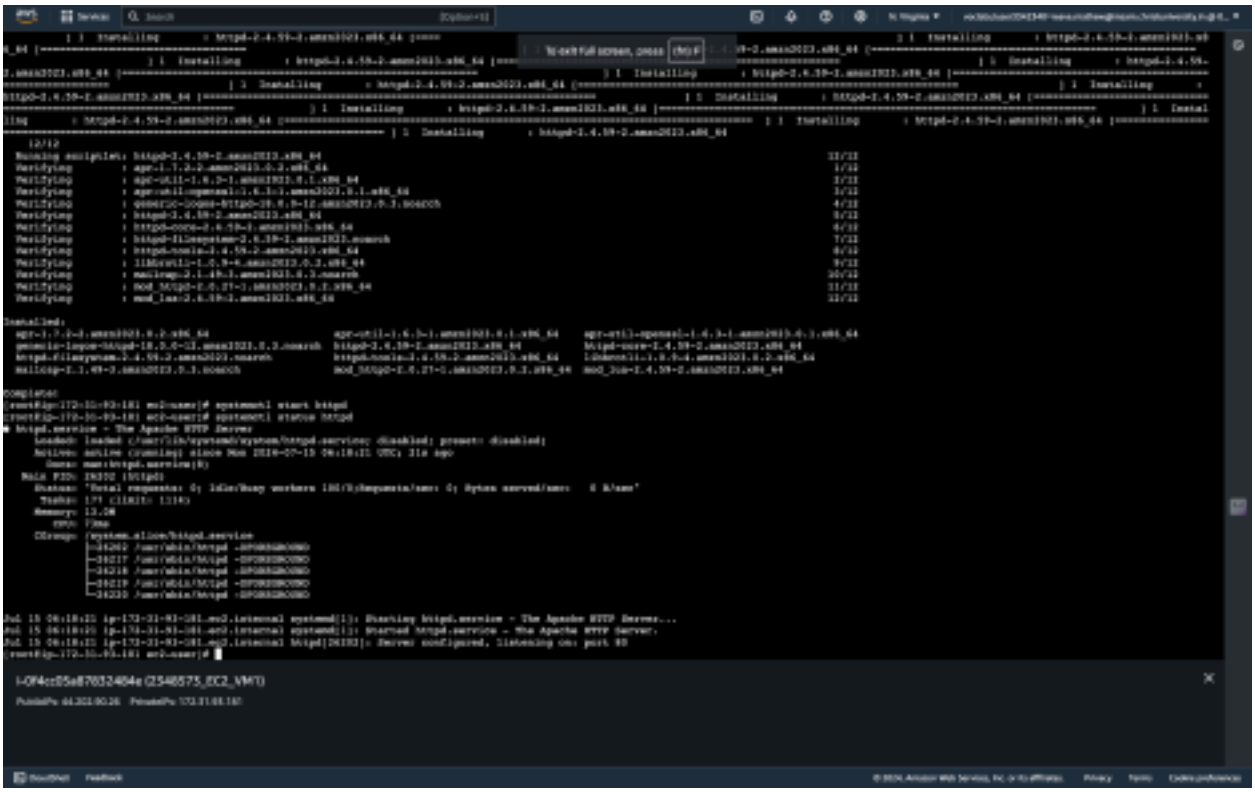




Step 3: Connect both the instances via amazon connect and run a static website in it. To do it run the below
commands
i) Sudo su
ii) Yum update -y



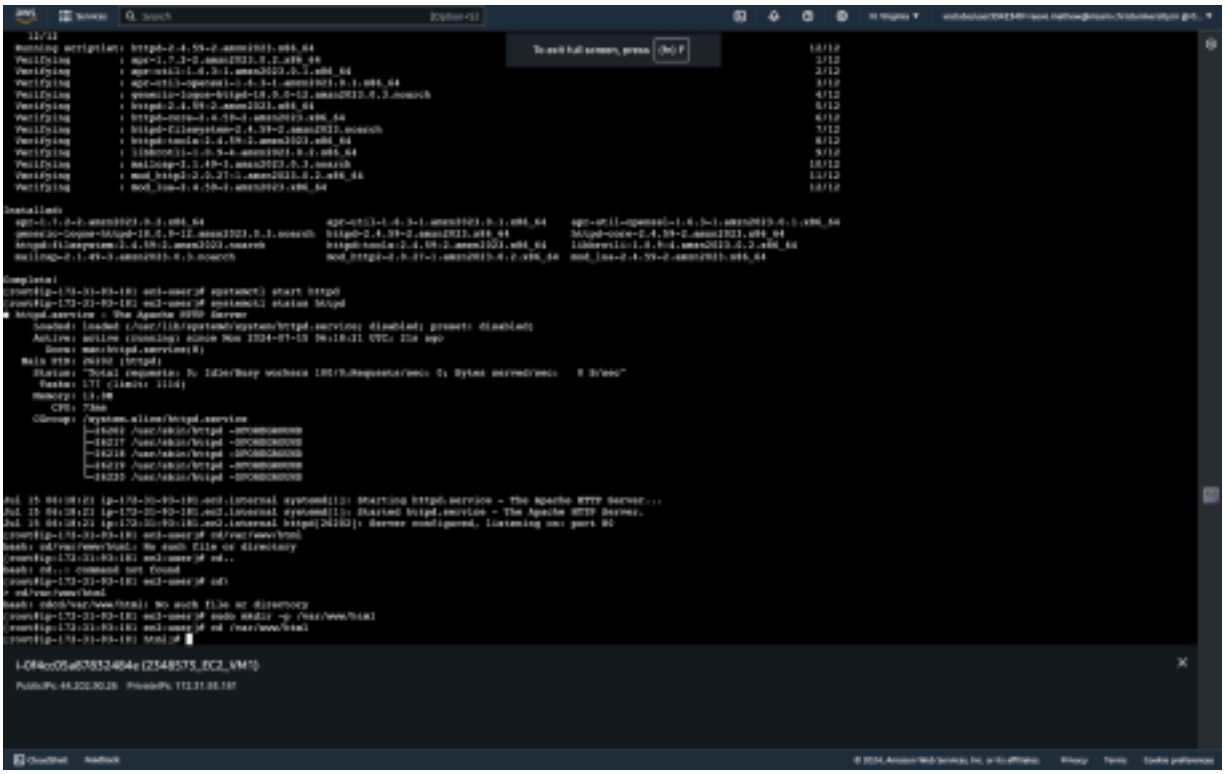
iii) Yum install httpd -y



iv) Systemctl start httpd

v) Systemctl status httpd

vi) Cd/var/www/html



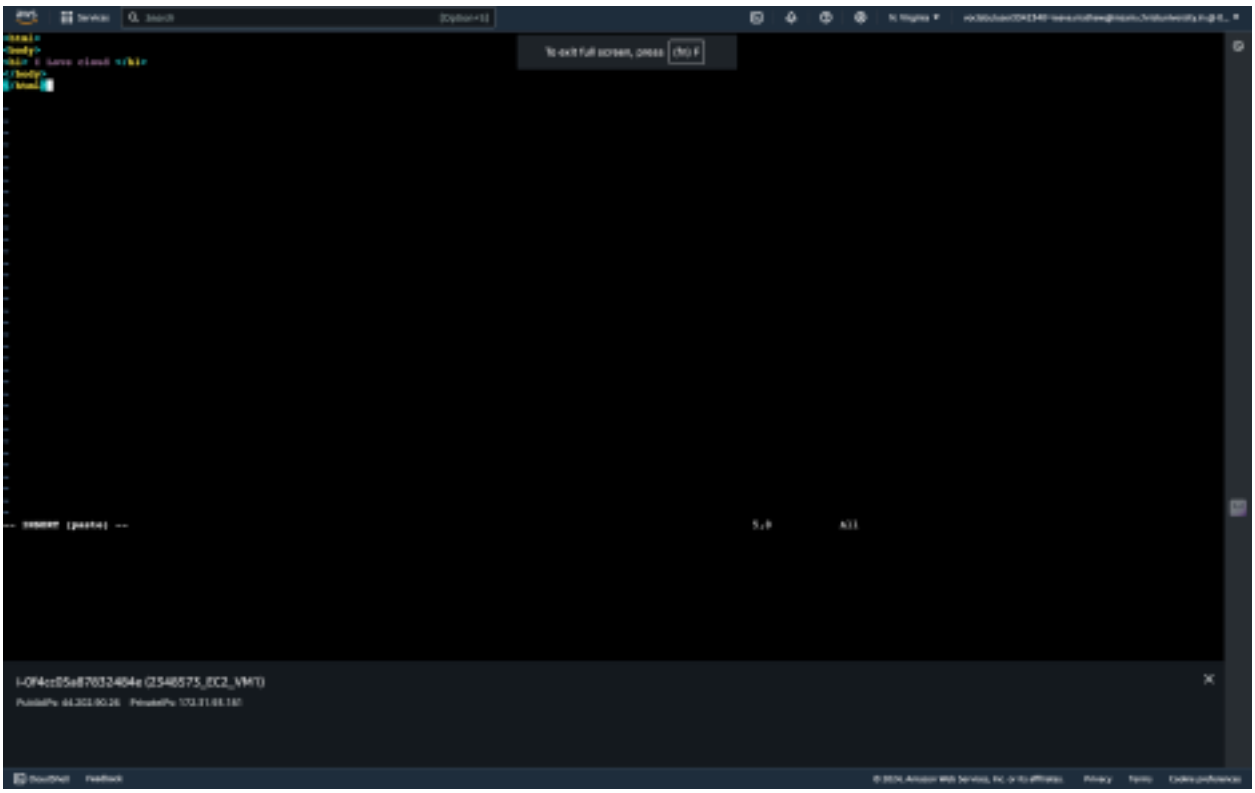
vii) Vim index.html

a. For Instance 1

```
<html>
<body>
<h1> My name is Reeve r </h1>
</body>
</html>
```

b. For instance 2

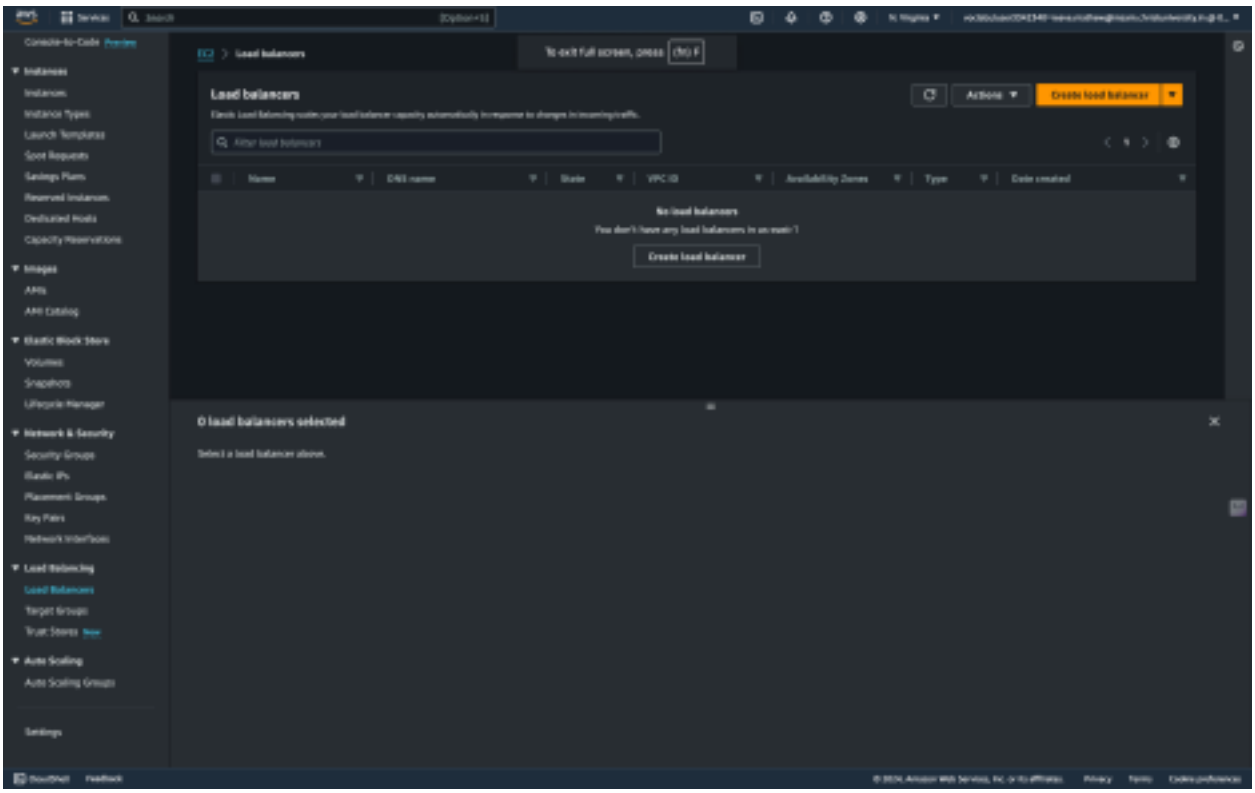
```
<html>
<body>
<h1> I Love cloud </h1>
</body>
</html>
```



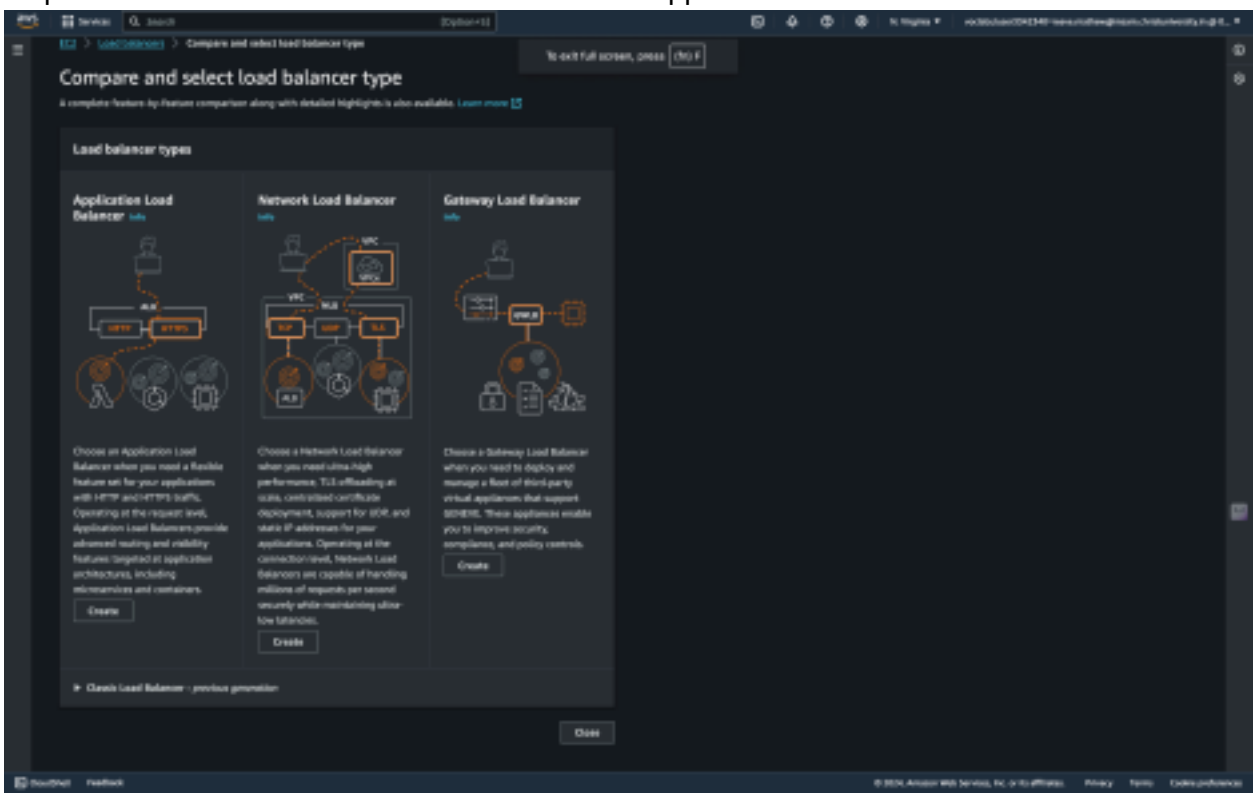
viii) Systemctl start httpd

Step 4: Exit the Cloudshell of both the instances and navigate back to EC2 Step 5: Copy the public IPv4DNS address of both the instances and paste it in anew tab and verify the html file is
running or not.

Step 6: Navigate to AWS search panel and search for EC2 Load Balancer



Step 7: Click on Create Load Balancer and select Application Load Balancer



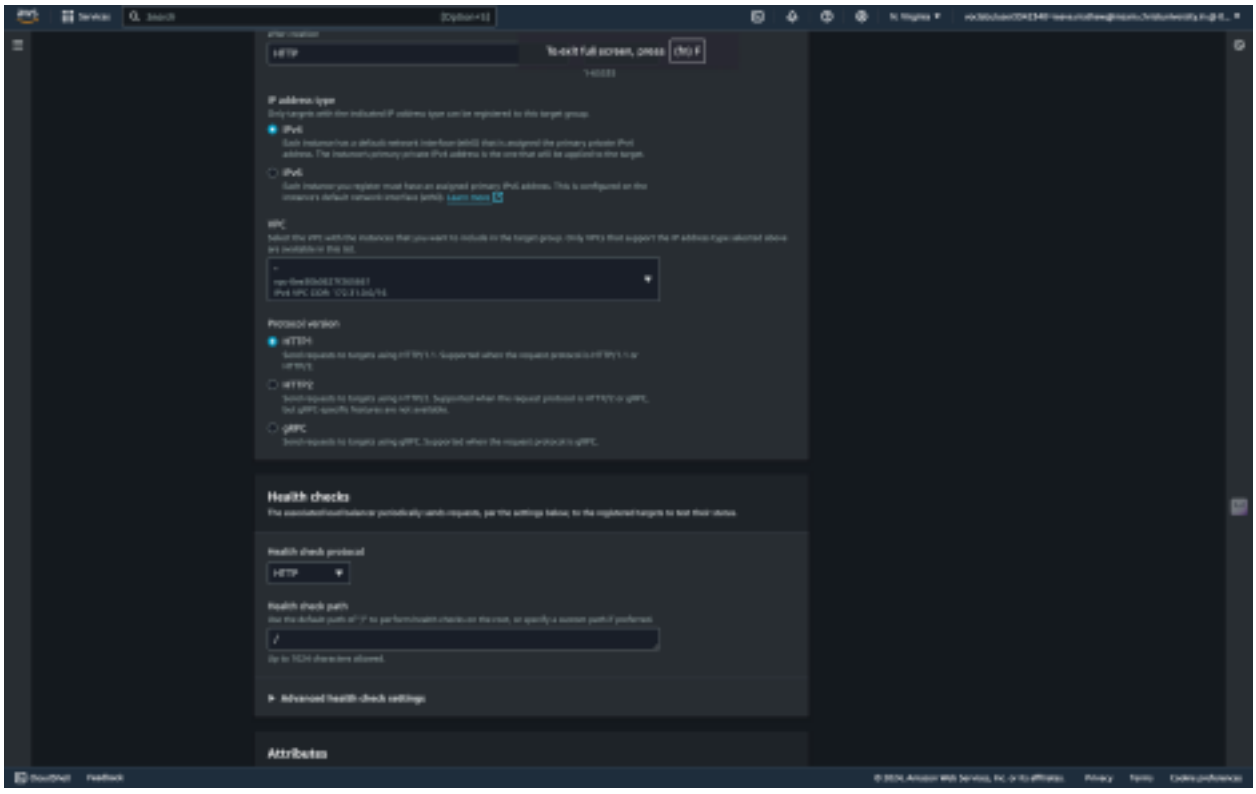
Step 8: Application Load Balancer Configuration

i) Load Balancer Name: 2348554LB

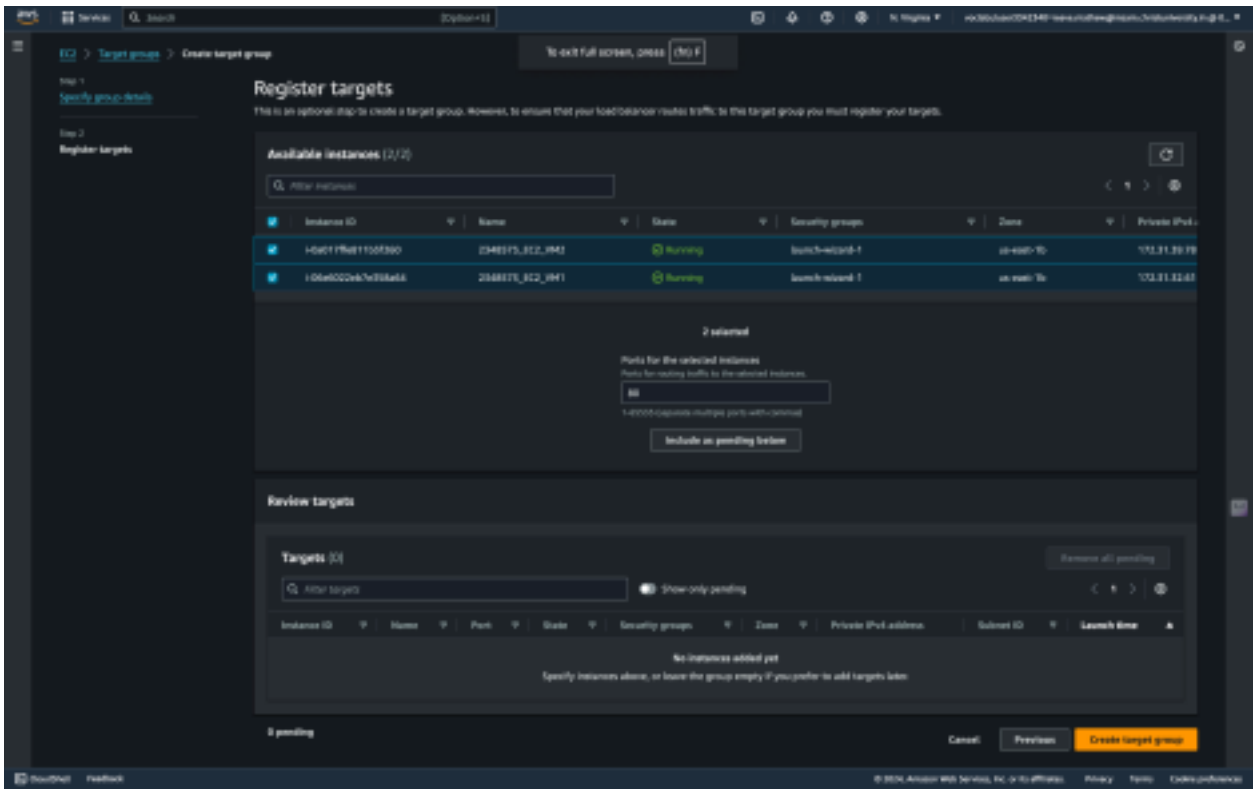
ii) Scheme: Internet-facing (Default)
iii) Load Balancer IP address type: IPv4 (Default)
iv) Network Mapping: Select the default VPC and make sure to select the availability zone in which your
instance is running on. In my case it is “us-east-1d”. We will have to select two availability zones, so
select any other of your choice. Leave the subnet by default.

v) Security Group: Select the same security group of your launched instances. In my case it is “launch
wizard-1”.

vi) Listeners and Routing: Click on create a target group
a. Target Group Configurations
b. Target Group Name: 2348554TG
c. IP address Type: IPv4 (Default)
d. Protocol Version HTTP1 (Default)

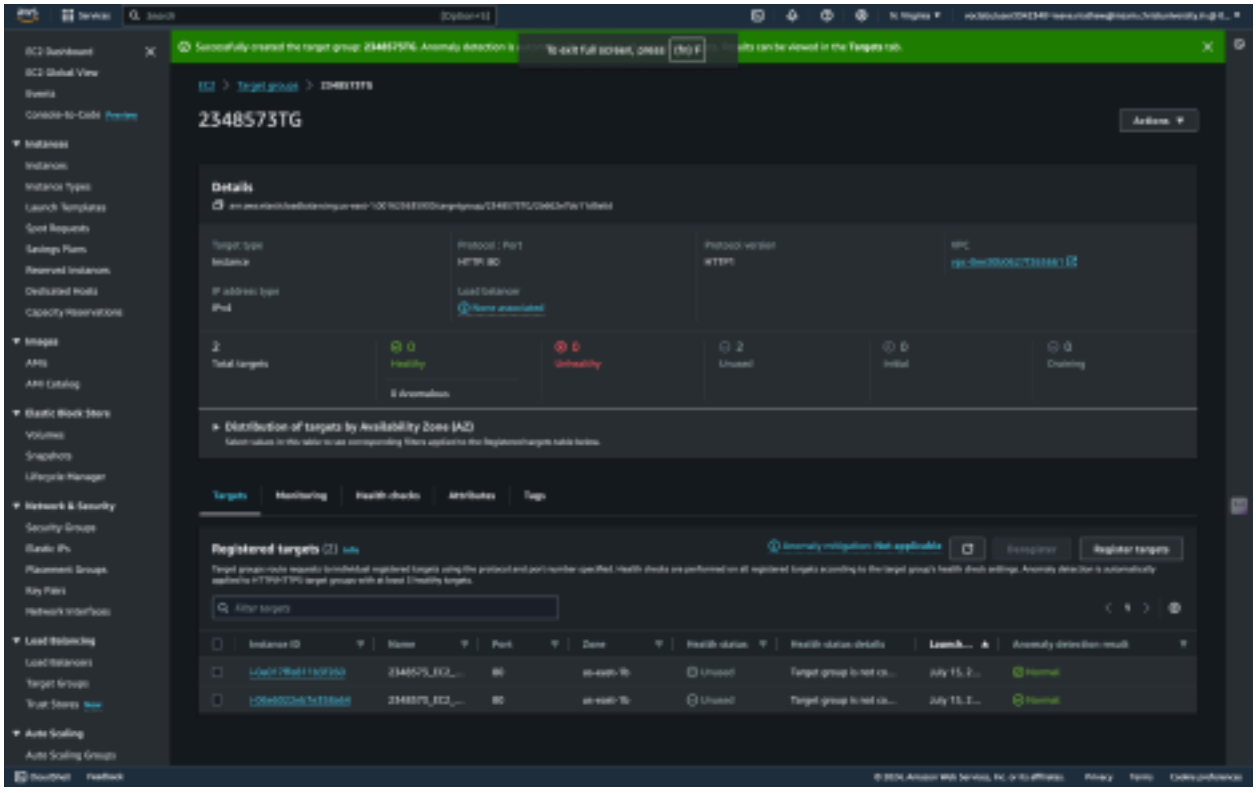


e. Click on Next
f. Select both the available instances and click on include as pending below
g. Click on Create Target Group

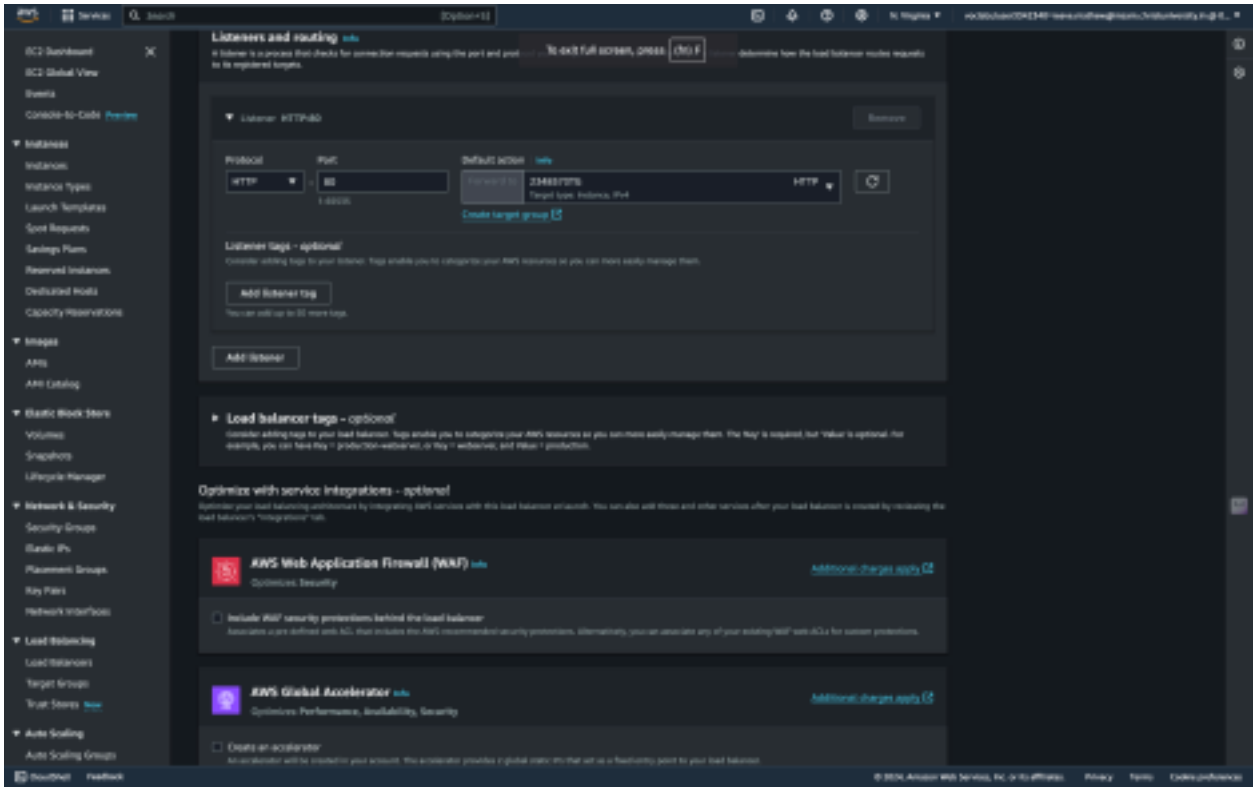


vii) Navigate back to load balancer and select the created target group

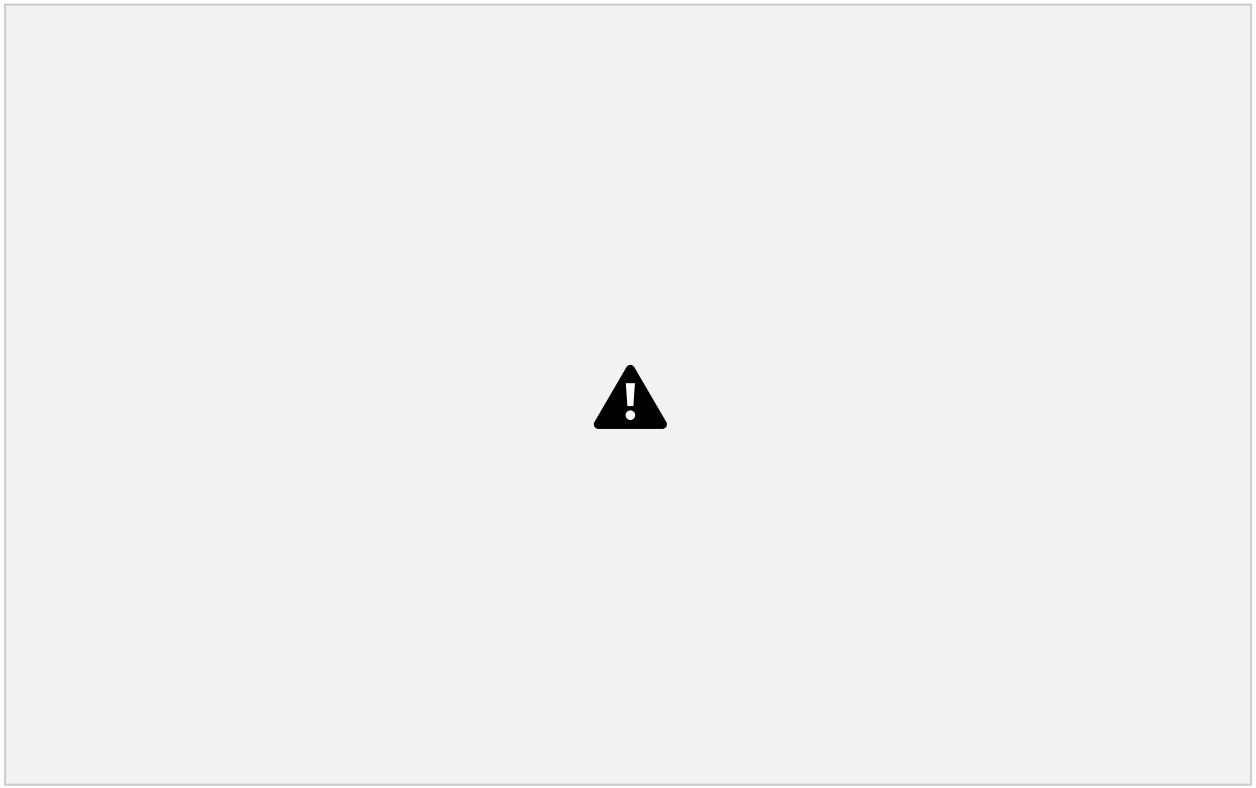
viii) Click on Create Load Balancer



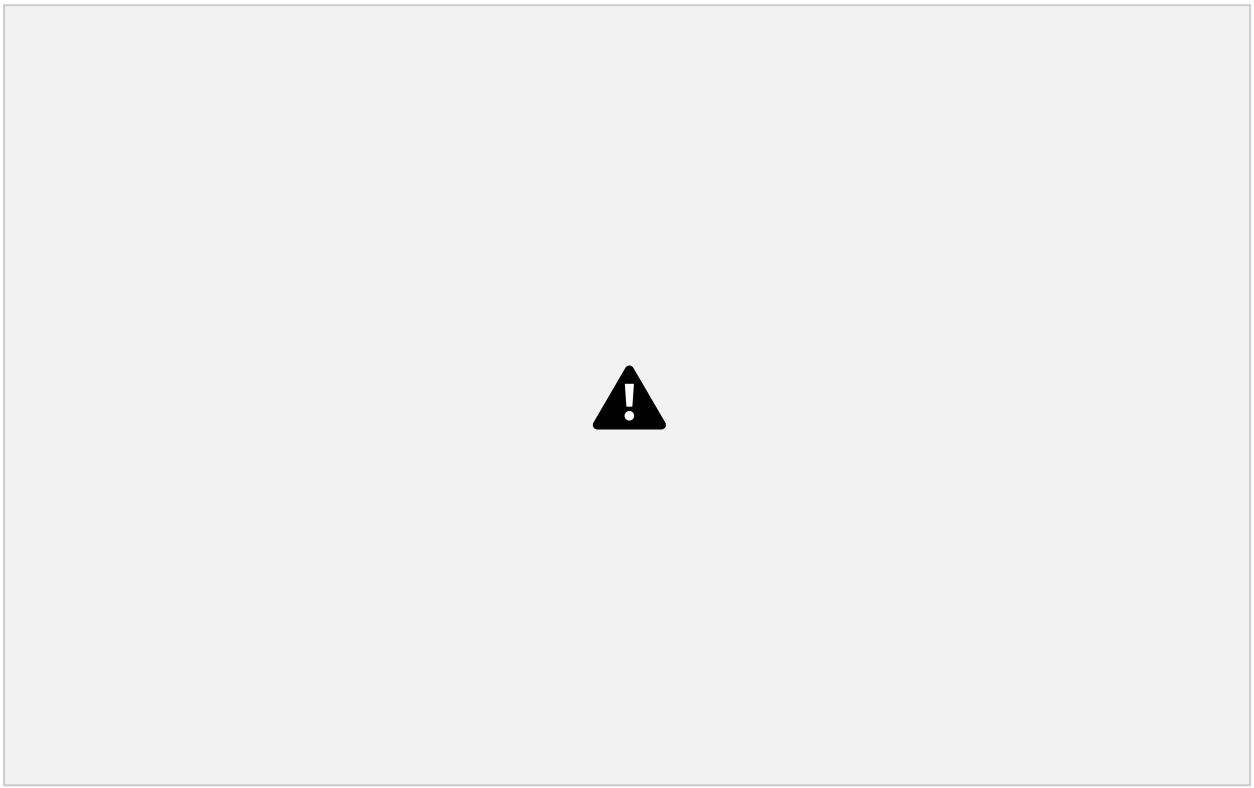
Step 9: Wait for the Load Balancer Status to be active. Might take 2-5 minutes.

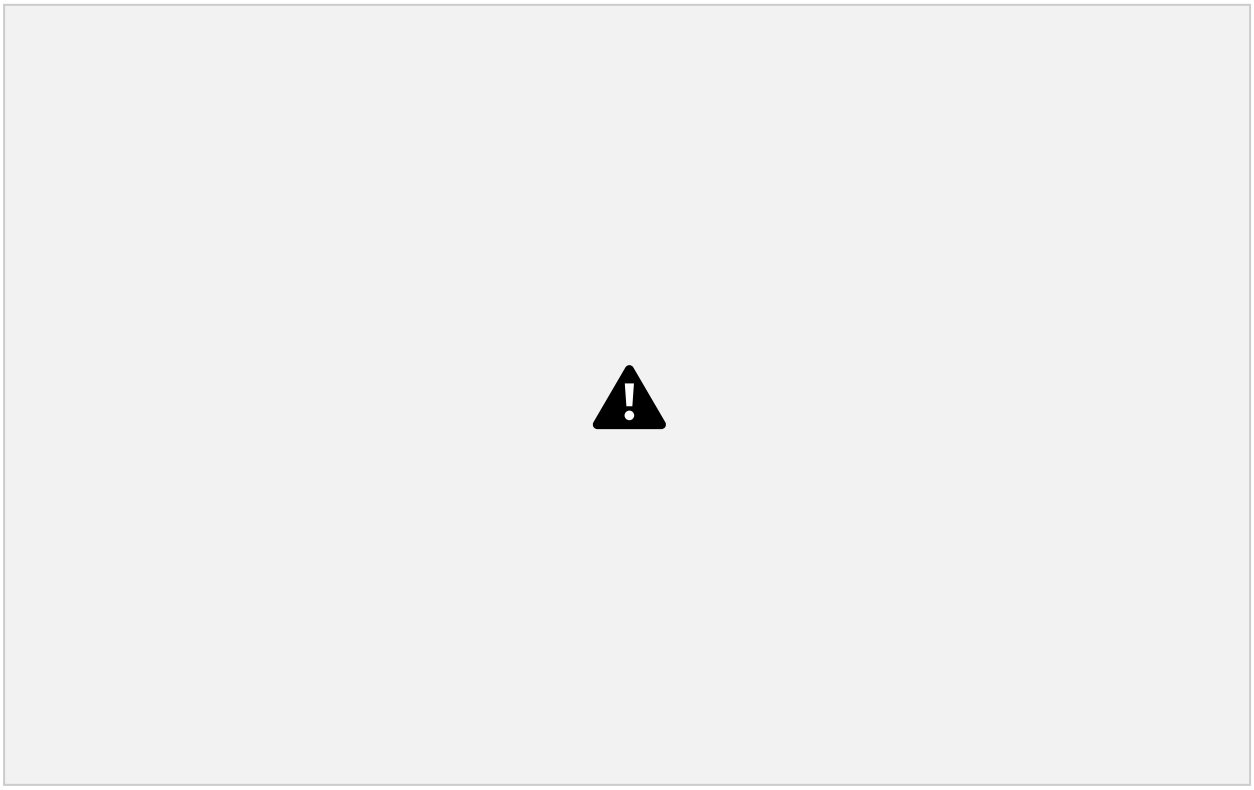


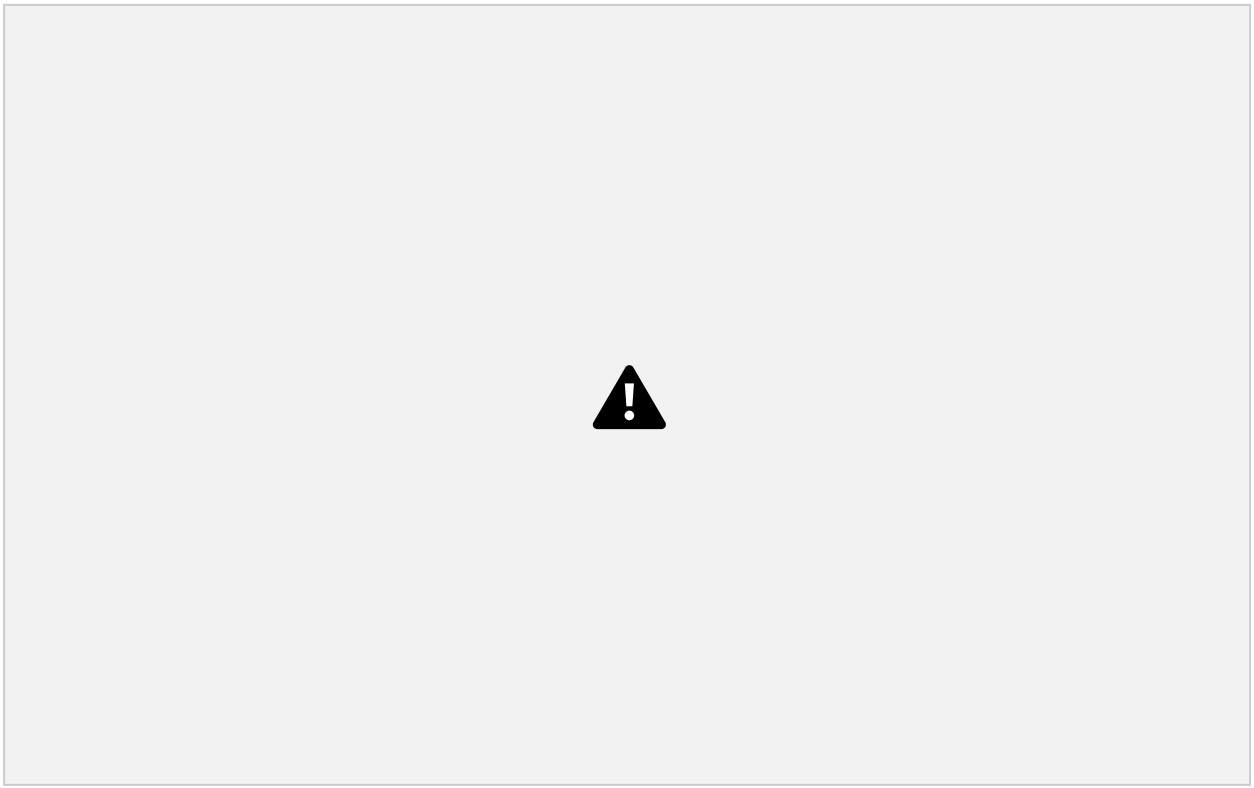
Step 10: Navigate back to Target Group and check the health status of both the instances, it should so “healthy”.



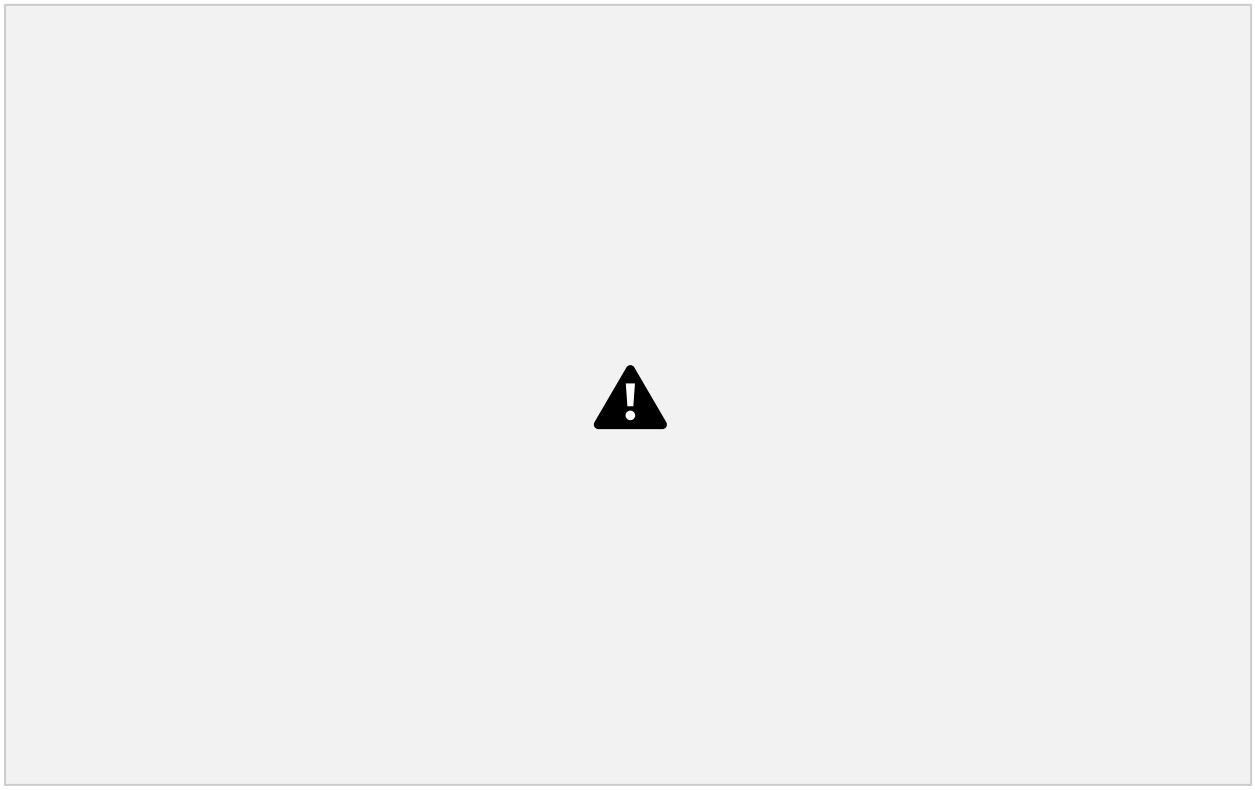
Step 11: Copy the Load Balancer DNS name and paste it on a browser. We will see the index.html of first instance. On
refreshing we will see the output of index.html of second instance.







NavigatetoLoadBalancerandcheckthemonitoringrequestsgraphandwewill findasurge in thegraph.
Navigate to EC2 and select both the instances and check the monitoring graphs.



Delete every services step by step to avoid costs