**Placement Empowerment Program**

***Cloud Computing and DevOps Centre***

**Set Up a Load Balancer in the Cloud**Configure a load balancer to distribute traffic across multiple VMs hosting your web application.

Name: Kowshica N Department : Biotech



**Introduction and Overview**

A **Load Balancer** is a crucial component in cloud architecture that helps distribute incoming network traffic across multiple virtual machines (VMs) to ensure **high availability, fault tolerance, and scalability** of a web application. Cloud providers like **AWS, Google Cloud (GCP), and Microsoft Azure** offer managed load balancer services that optimize performance and reliability.

**Objective**

 Set up a **Load Balancer** in a cloud environment (AWS, GCP, or Azure).

 Configure it to distribute traffic among multiple **Virtual Machines (VMs)** hosting a web application.

 Ensure **high availability, fault tolerance, and optimized performance**.

 Test the setup by simulating traffic distribution.

**Importance**

 **Ensures High Availability:** Prevents a single point of failure by distributing traffic across multiple servers.

 **Improves Performance:** Reduces response time by routing traffic to the least loaded or nearest server.

 **Scalability:** Supports auto-scaling by dynamically adjusting the number of VMs based on demand.

 **Redundancy and Fault Tolerance:** If one server fails, the load balancer redirects traffic to healthy servers.

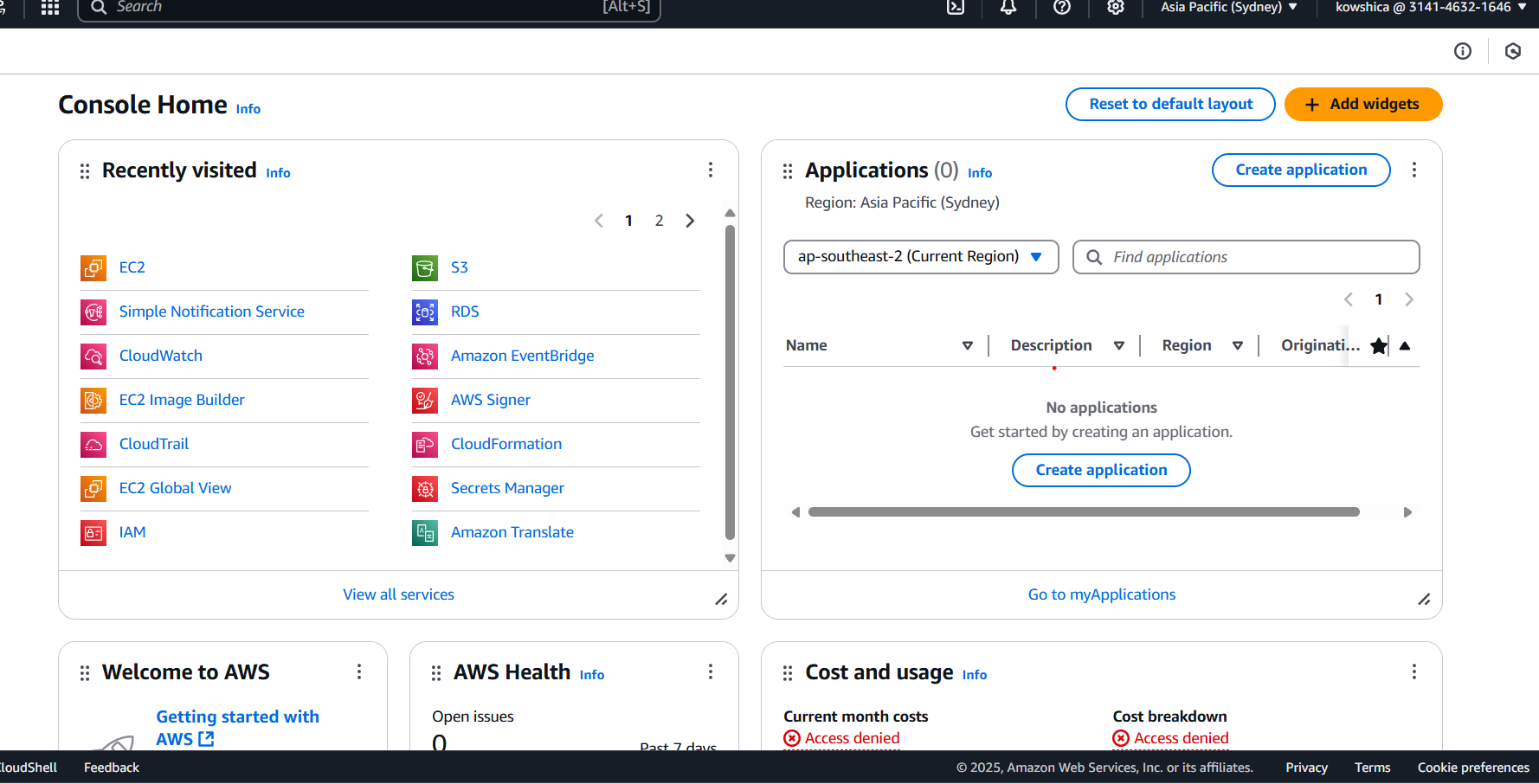
 **Security Benefits:** Many cloud load balancers offer built-in DDoS protection and SSL termination.

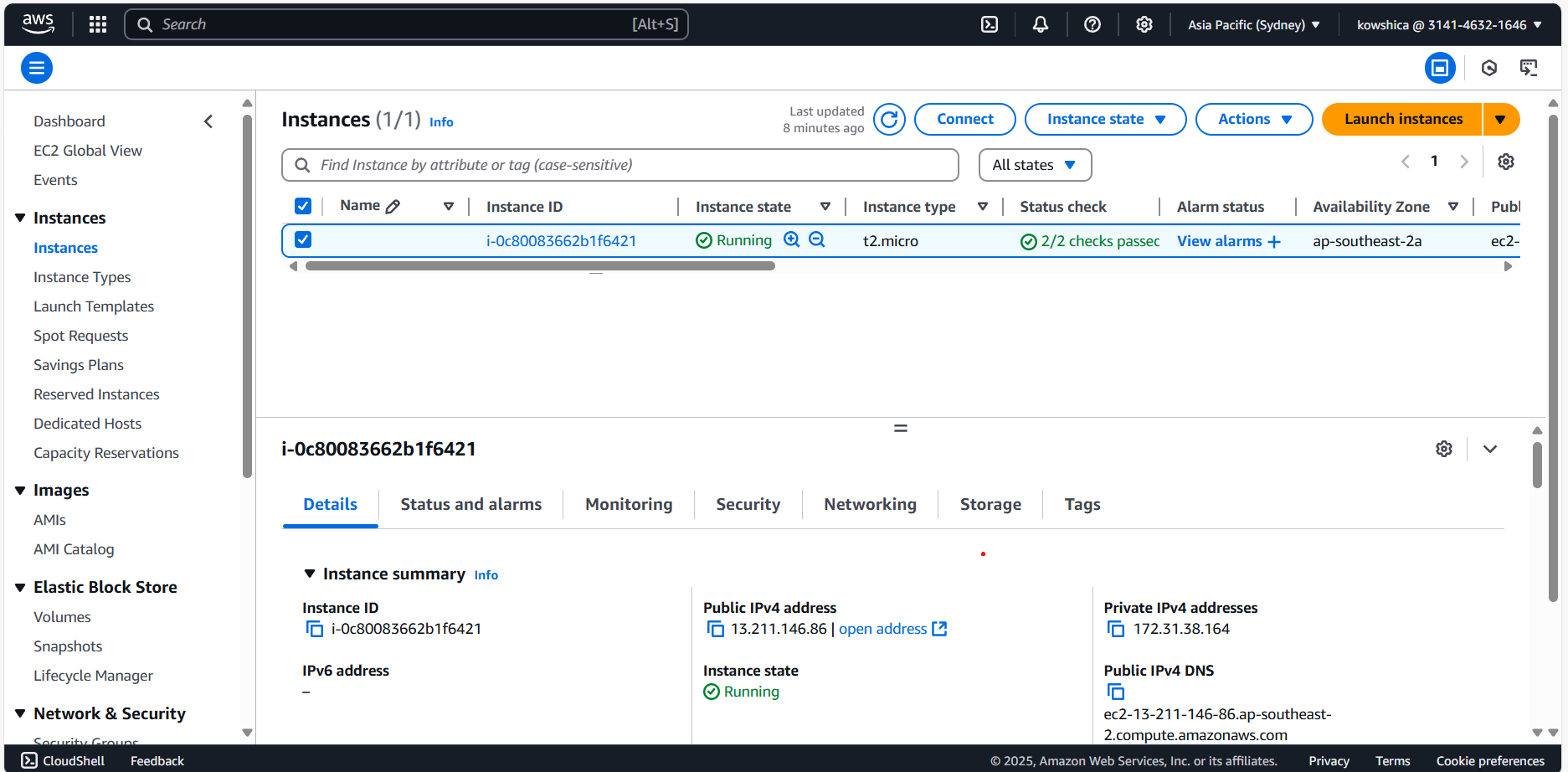
**Step-by-Step Overview**

Step1:

## =**Set Up Virtual Machines (EC2 Instances)**

1. **Log in to AWS Console** → Navigate to **EC2 Dashboard**.
2. **Launch multiple EC2 instances** (Ubuntu/Linux or Windows) to host your web application.
   * Choose an instance type (e.g., t2.micro for free-tier).
   * Configure security groups to allow HTTP (port 80) and HTTPS (port 443).
   * Assign a **public IP or Elastic IP** if needed.

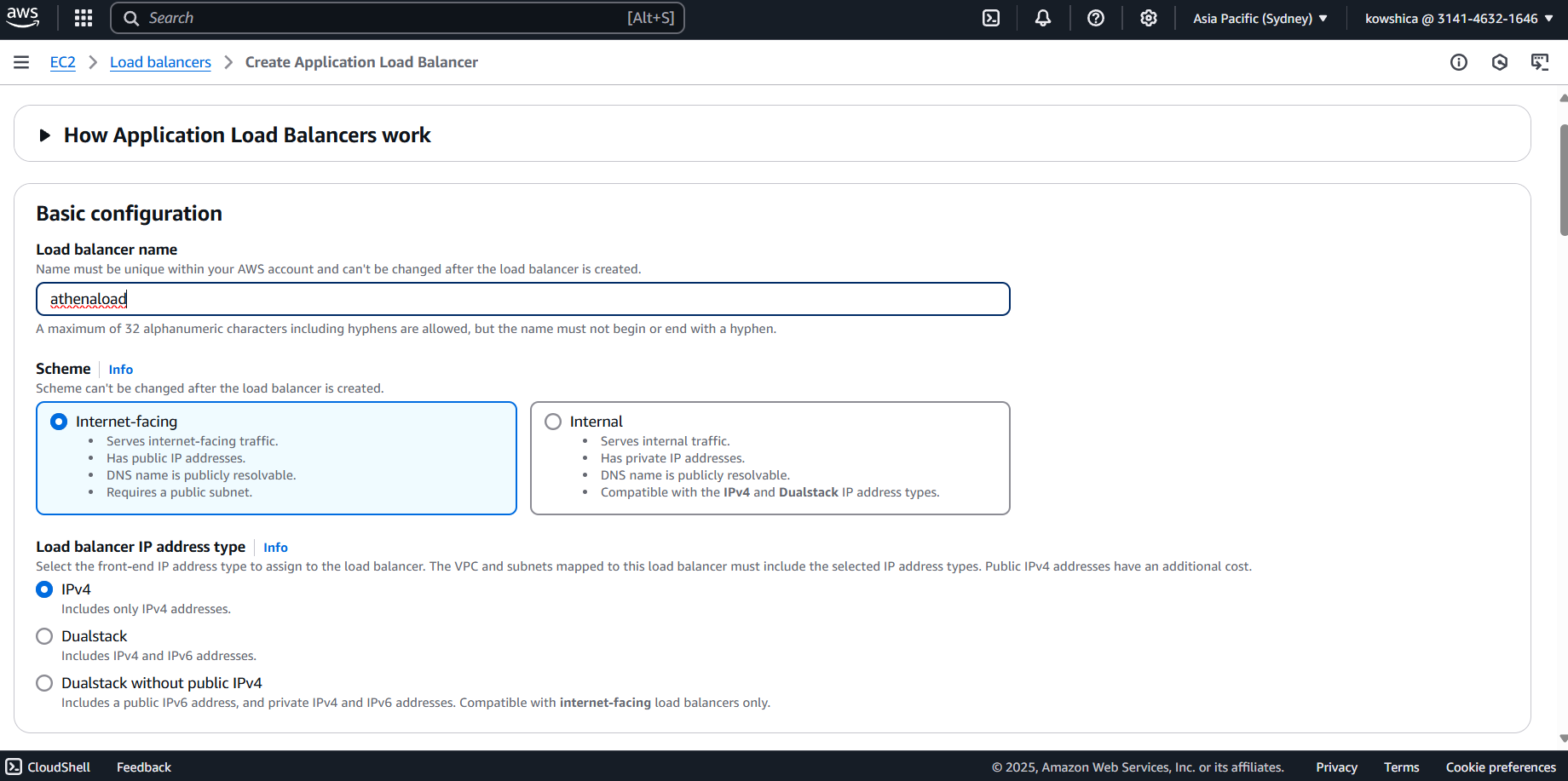




Step 2 :

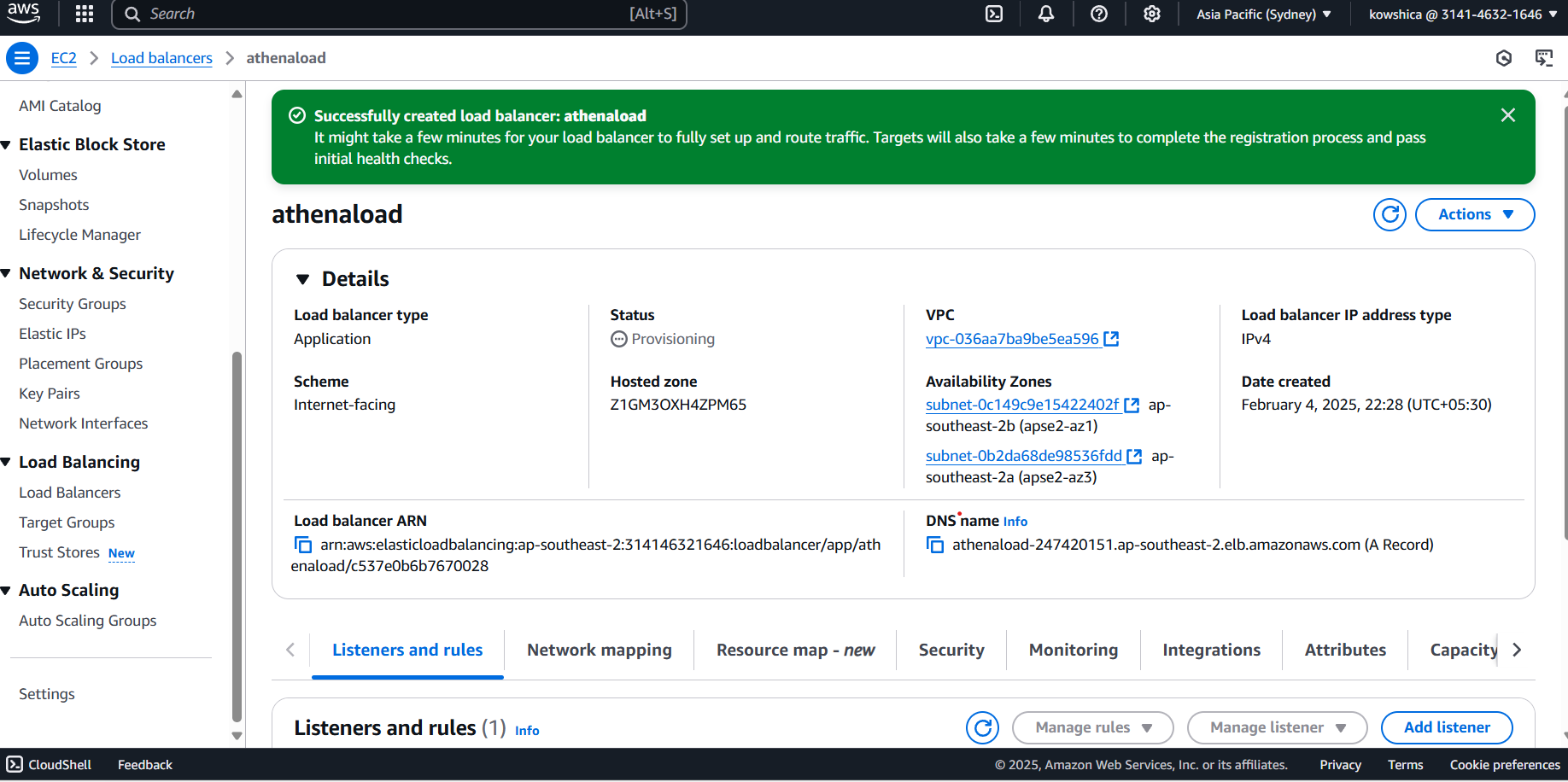
**Create a Load Balancer**

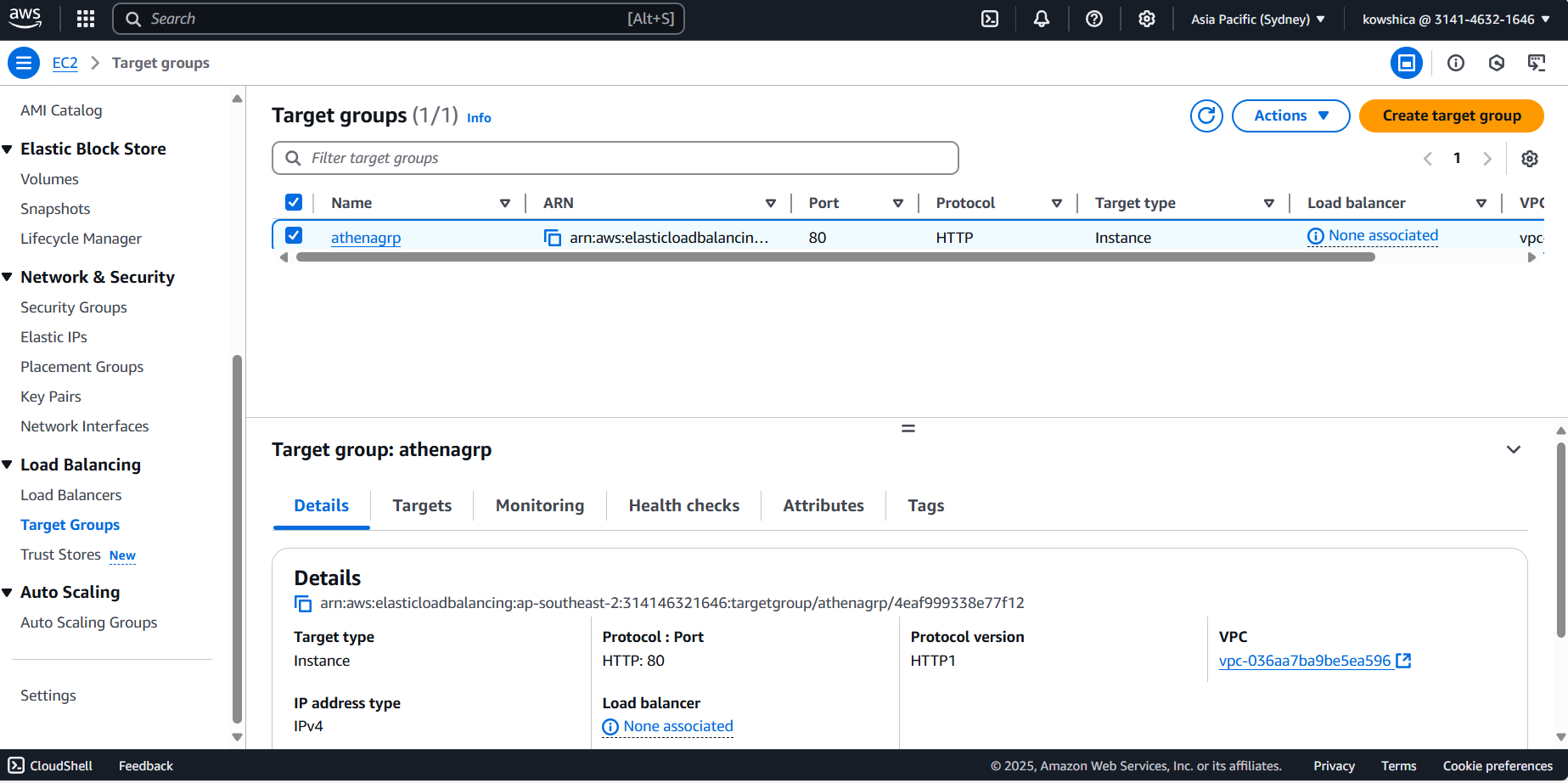
1. **Navigate to EC2 Dashboard** → Click on **Load Balancers**.
2. Click **Create Load Balancer** → Select **Application Load Balancer (ALB)**.
3. **Configure Load Balancer:**
   * Name: MyWebAppLoadBalancer
   * Scheme: Internet-facing
   * IP Type: IPv4
   * Listeners: Add **HTTP (80)** and **HTTPS (443) (Optional)**
   * VPC: Select an existing **VPC and Subnets**.



Step 3 :

**Create a Target Group**

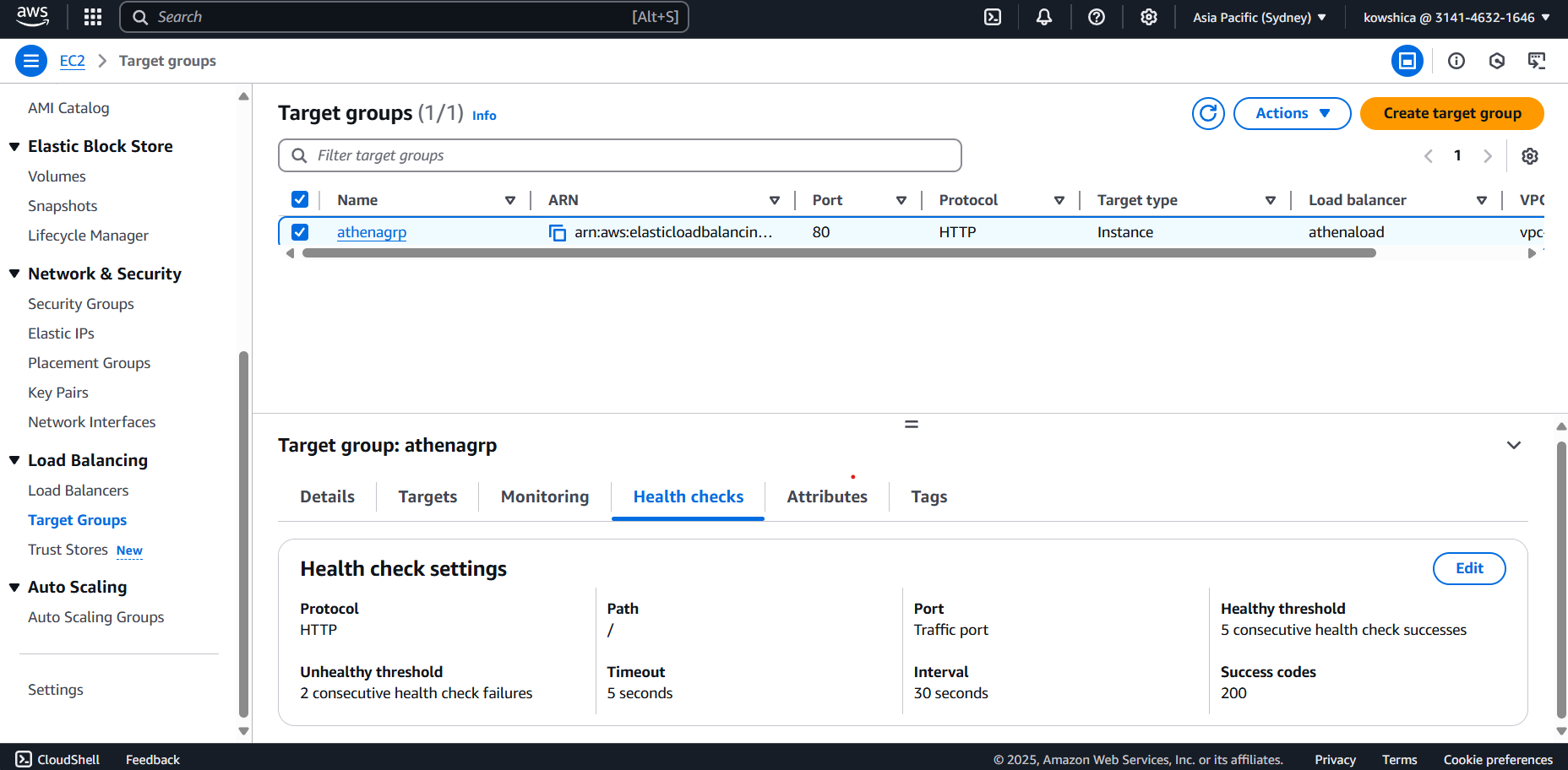
1. **Go to "Target Groups"** → Click **Create Target Group**.
2. Choose **Target Type: Instances**.
3. Set protocol as **HTTP (80)**.
4. Register the **EC2 instances** as targets.
5. Click **Create** and attach it to the Load Balancer. 



Step 4 :

**Configure Health Checks**

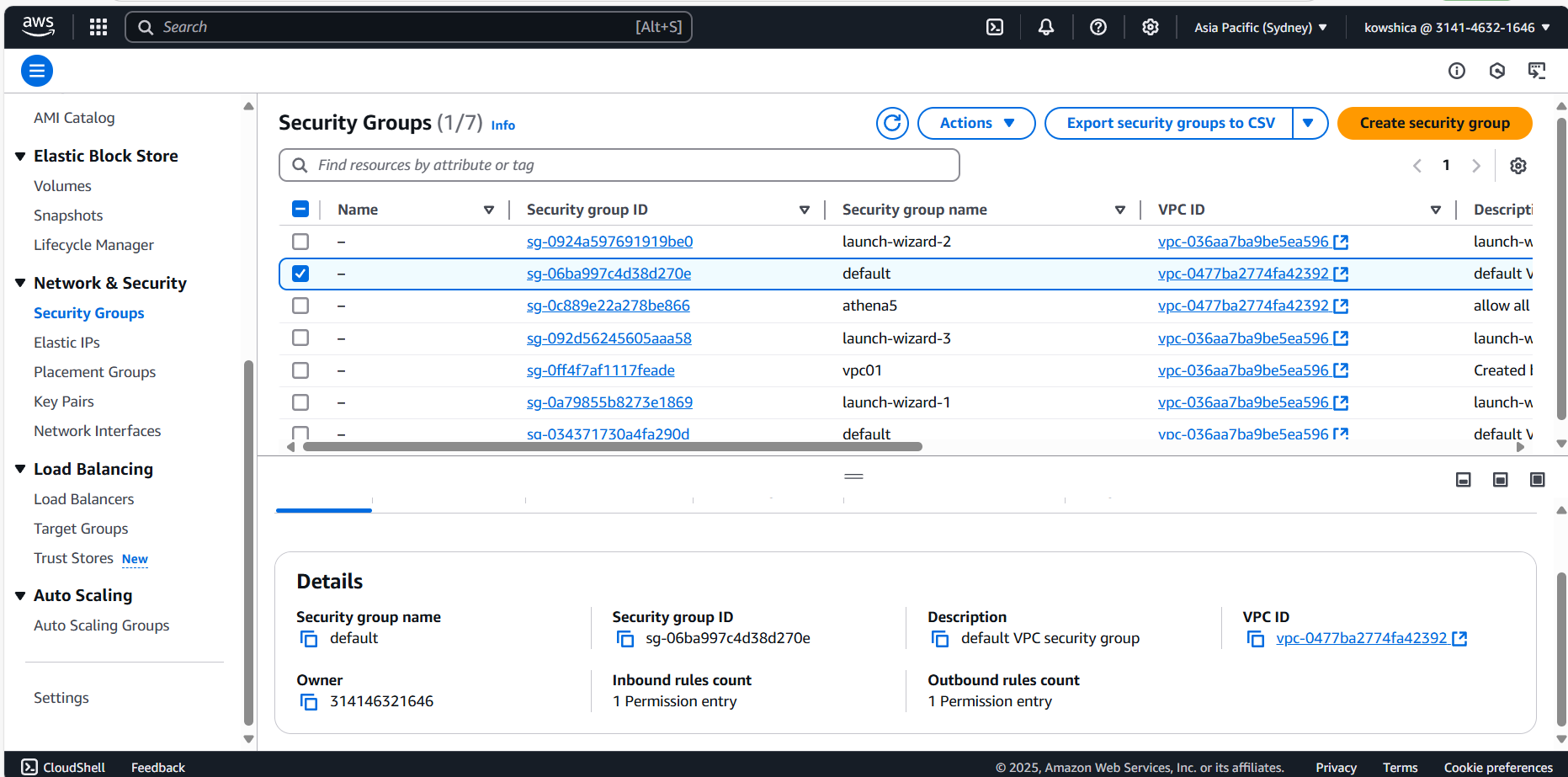
1. Set up a **health check endpoint** to monitor server health.
2. Configure:
   * Path: /index.html
   * Healthy threshold: 3
   * Unhealthy threshold: 2
   * Response timeout: 5 sec



Step 5 :

**Add Security Rules**

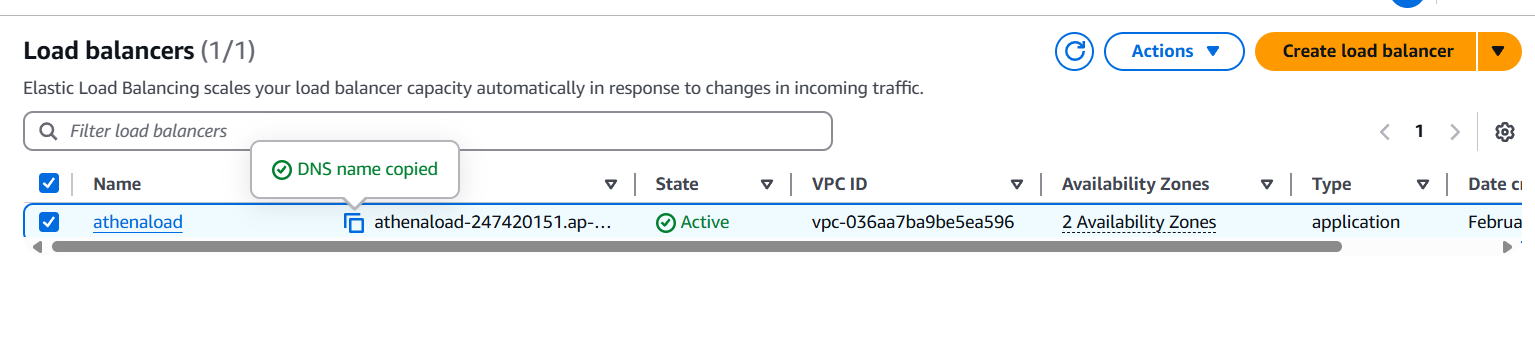
1. Edit **Security Groups**:
   * Allow **HTTP (80)** and **HTTPS (443)** for public access.
   * Restrict **SSH (22)** to your IP.
2. Enable **SSL termination** (if using HTTPS).



Step 6 :

**Test Load Balancer**

1. Copy the **Load Balancer DNS name** from the AWS console.
2. Paste it into the browser. You should see responses from different servers as you refresh the page.
3. **Simulate failure:** Stop one EC2 instance and check if traffic is rerouted.



Step 7 :

**Enable Auto Scaling (Optional)**

1. **Go to Auto Scaling Groups** → Click **Create Auto Scaling Group**.
2. Set a **desired instance count** and scale-out/in conditions.
3. Attach it to the **Load Balancer** to automatically adjust instances.

.

**Expected Outcome**

 A functional **load-balanced web application** with automatic traffic distribution.

 **Increased reliability** with reduced downtime.

 Efficient **scaling of resources** based on incoming traffic.

 Secure and optimized **network traffic management**.

Setting up a **Load Balancer** in the cloud significantly enhances the **availability, scalability, and performance** of web applications. By distributing traffic effectively and handling failures automatically, organizations can ensure a seamless user experience.