**GITHUB link**

|  |  |
| --- | --- |
| GITHUB repository | https://github.com/Prashanth6782/Monitoring\_Scaling\_and\_automation.git |
| Boto3 | Present in the github |
| Python Scripts folder | Contains the boto3 python script which was executed |
| HTML folder | Contains the static HTML which was executed |

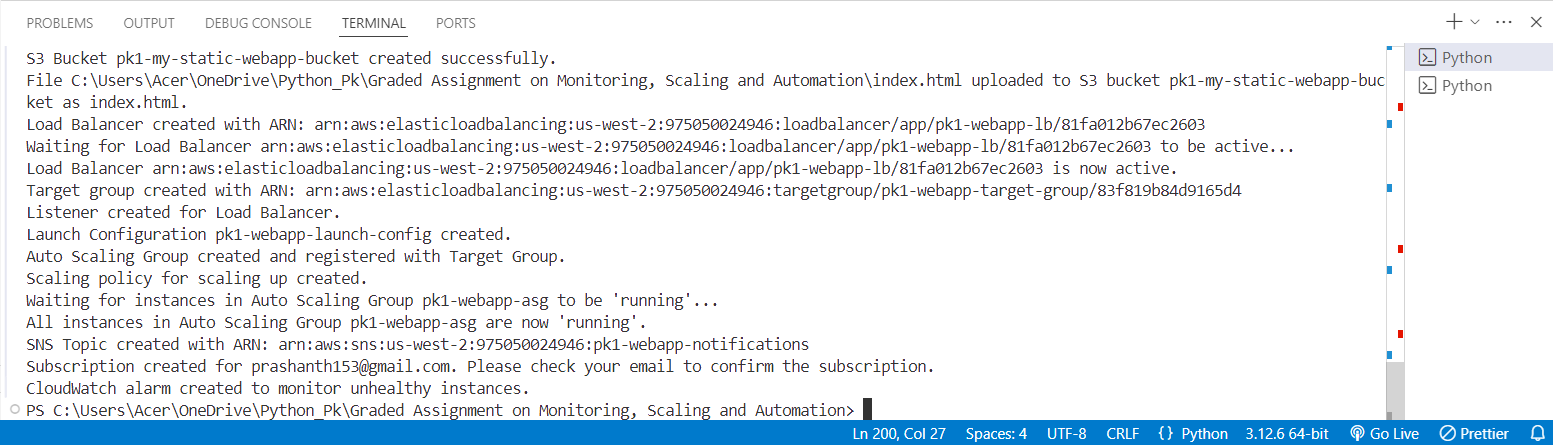
**User Training and Resources Document (UTR)**

**Project Overview**

This project automates the lifecycle management of a web application hosted on AWS using various services such as Amazon EC2, S3, Application Load Balancer (ALB), Auto Scaling Groups (ASG), Amazon SNS, and CloudWatch. The goal is to create a resilient infrastructure that automatically scales based on traffic and monitors its health, while keeping administrators informed through notifications.

***Resources Created:***

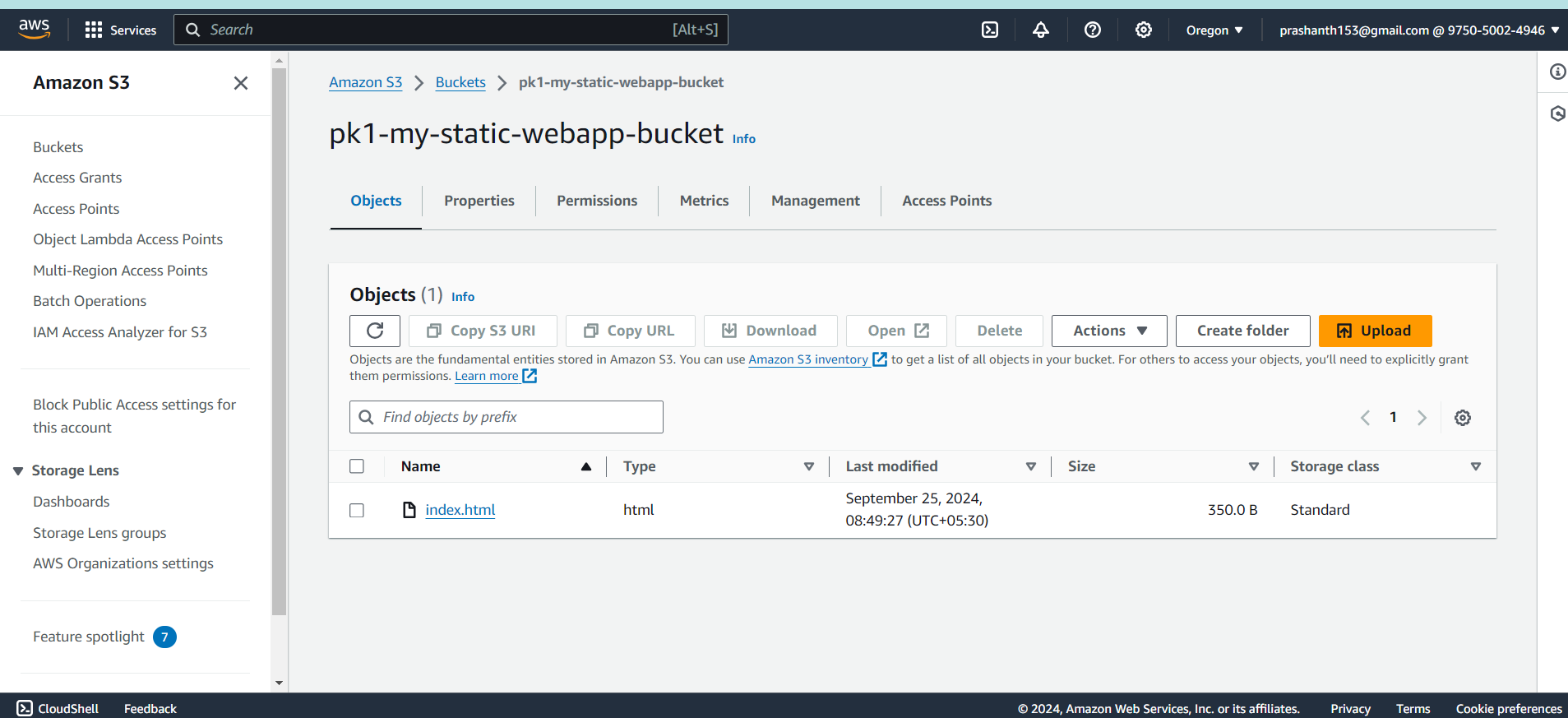
***After executing the boto3 script the resources were created successfully as shown in the boto3 terminal output from windows***

****

**1. S3 Bucket**

* **Bucket Name:** pk1-my-static-webapp-bucket  
  This S3 bucket is used to store static files for the web application, including the main HTML file (index.html). By using S3, the application can serve these files directly, ensuring high availability and durability.

***S3 bucket was created successfully with the index.html the static file***



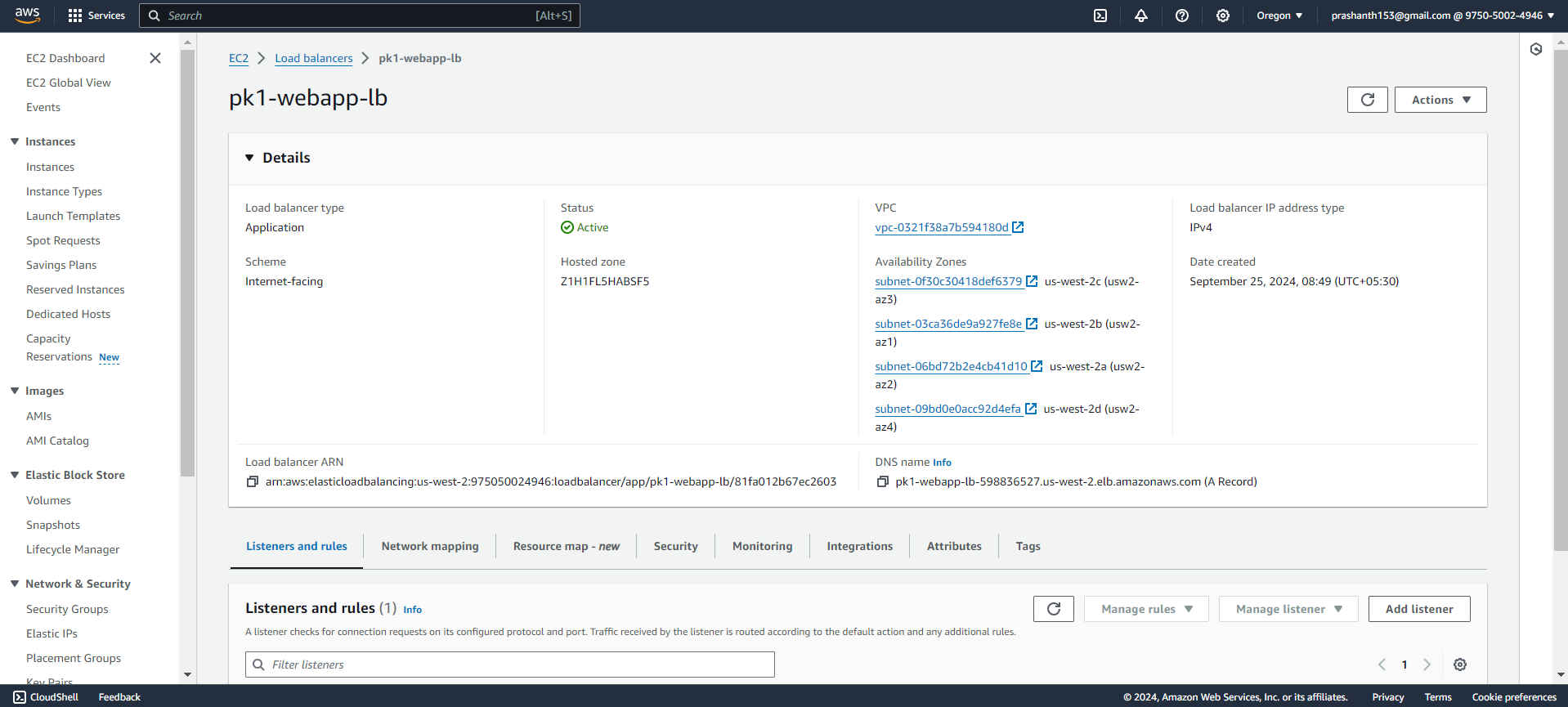
**2. EC2 Instances**

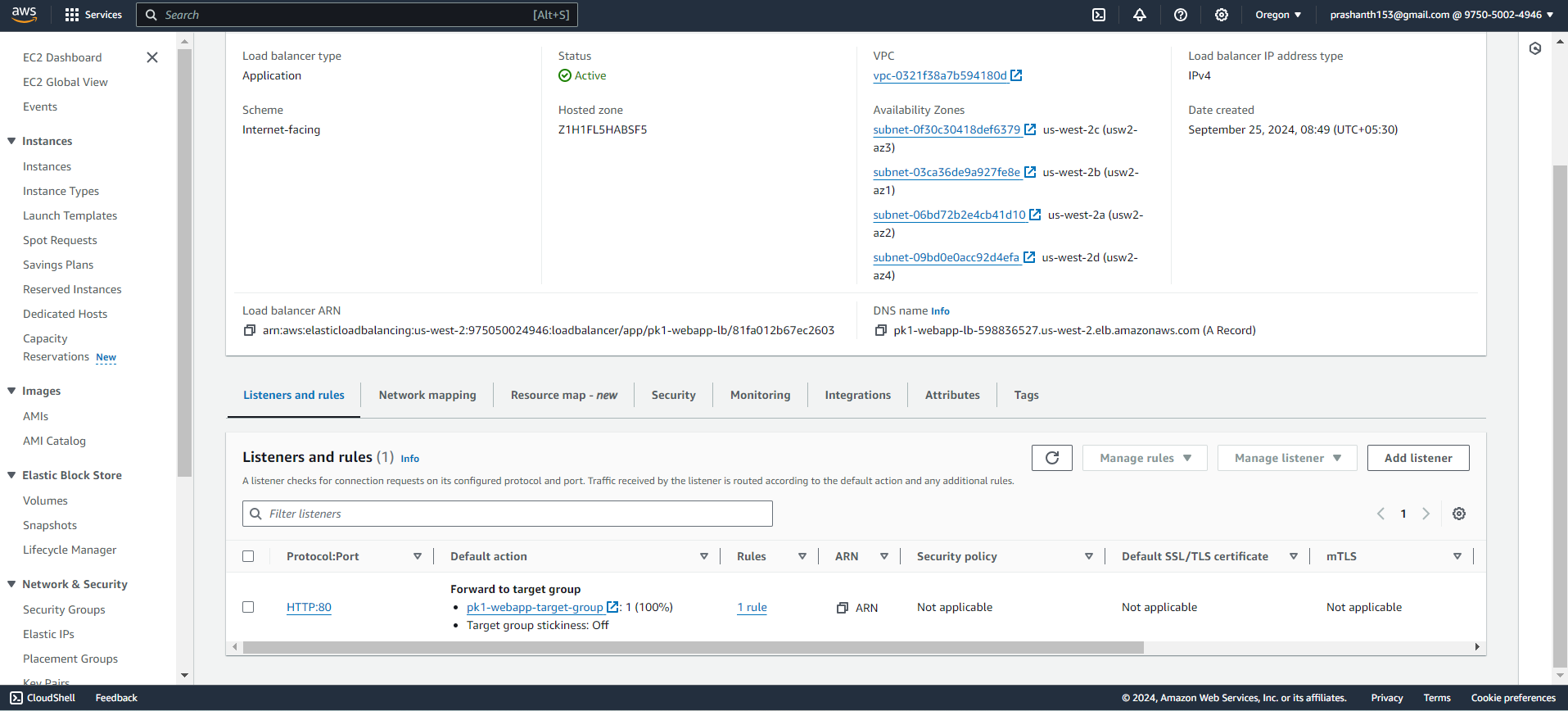
* **Launch Configuration Name:** pk1-webapp-launch-config  
  This configuration defines the EC2 instances that will host the web application. The instances are set up with a specified instance type (t4g.micro), associated security groups, and user data that installs necessary software and retrieves the application files from S3.

***I created an AMI with nginx installed which can be used for launching the ASG’s***

**3. Application Load Balancer (ALB)**

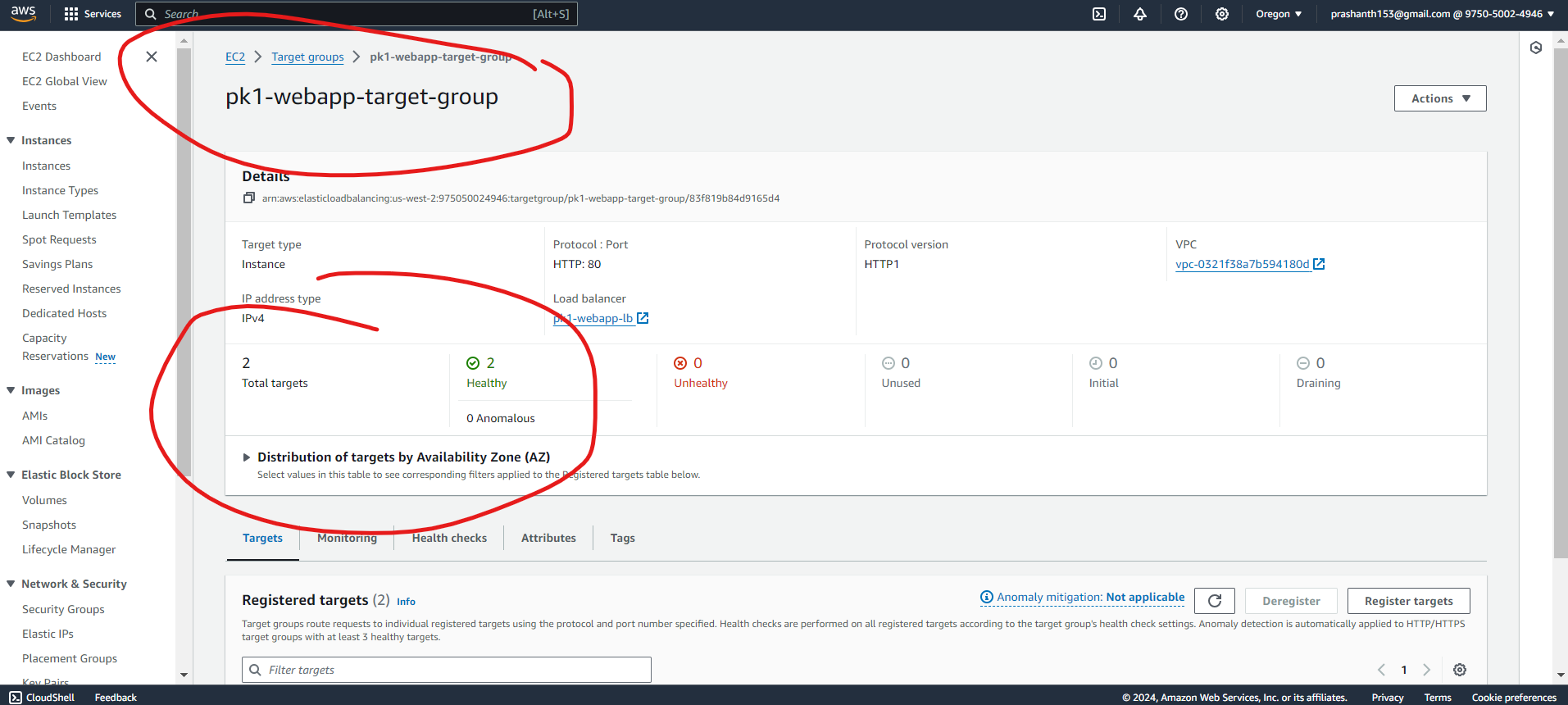
* **Load Balancer Name:** pk1-webapp-lb  
  The ALB distributes incoming traffic across multiple EC2 instances, enhancing the application’s availability and fault tolerance. It is configured to operate in specific subnets and is designed to handle HTTP requests efficiently.
* ***Load balancer was created successfully***

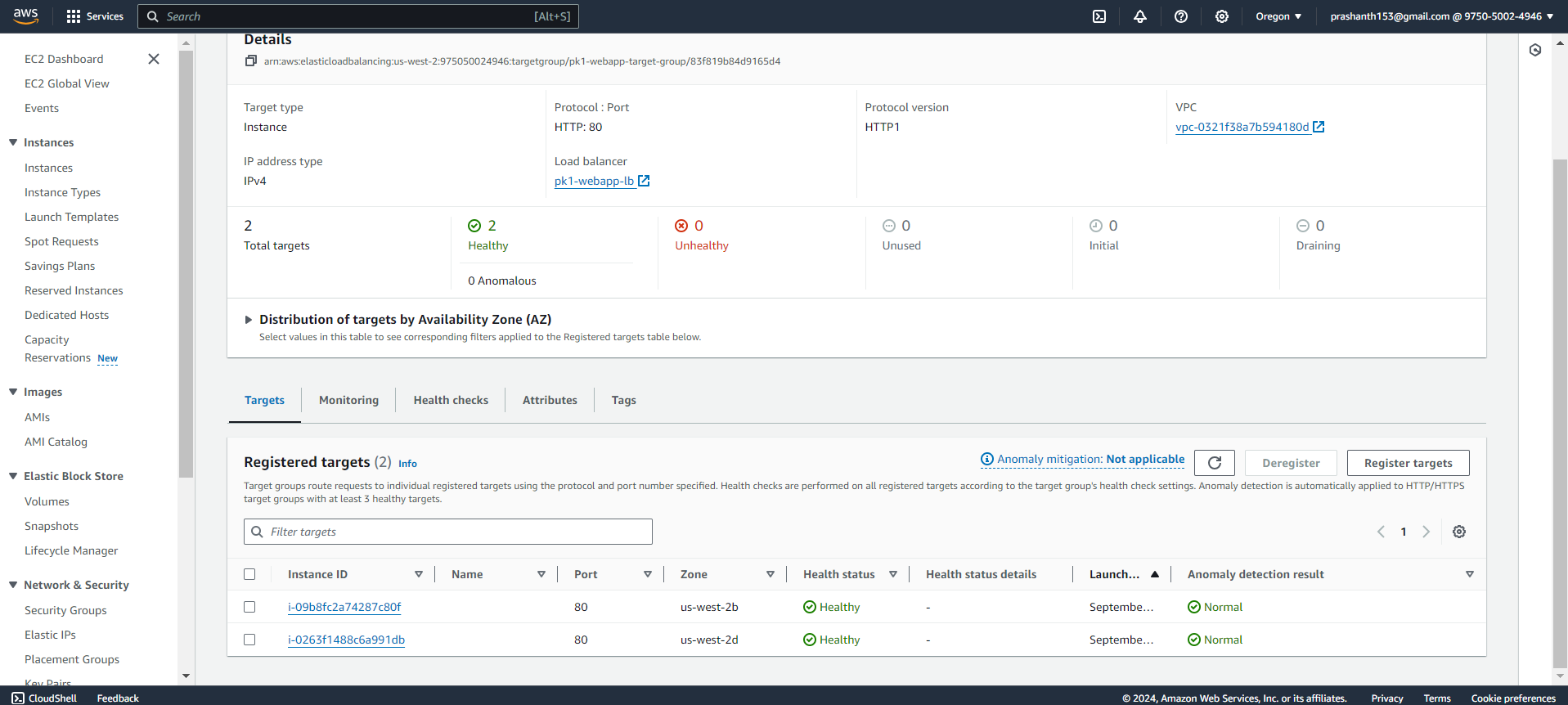


* ***Load balancer is attached to the target group as well as shown in the below screenshot***

**4. Target Group**

* **Target Group Name:** ***pk1-webapp-target-group***This target group ***registers*** the EC2 instances to receive traffic from the ALB. It is set to monitor the health of these instances and route traffic accordingly, ensuring that only healthy instances serve user requests.

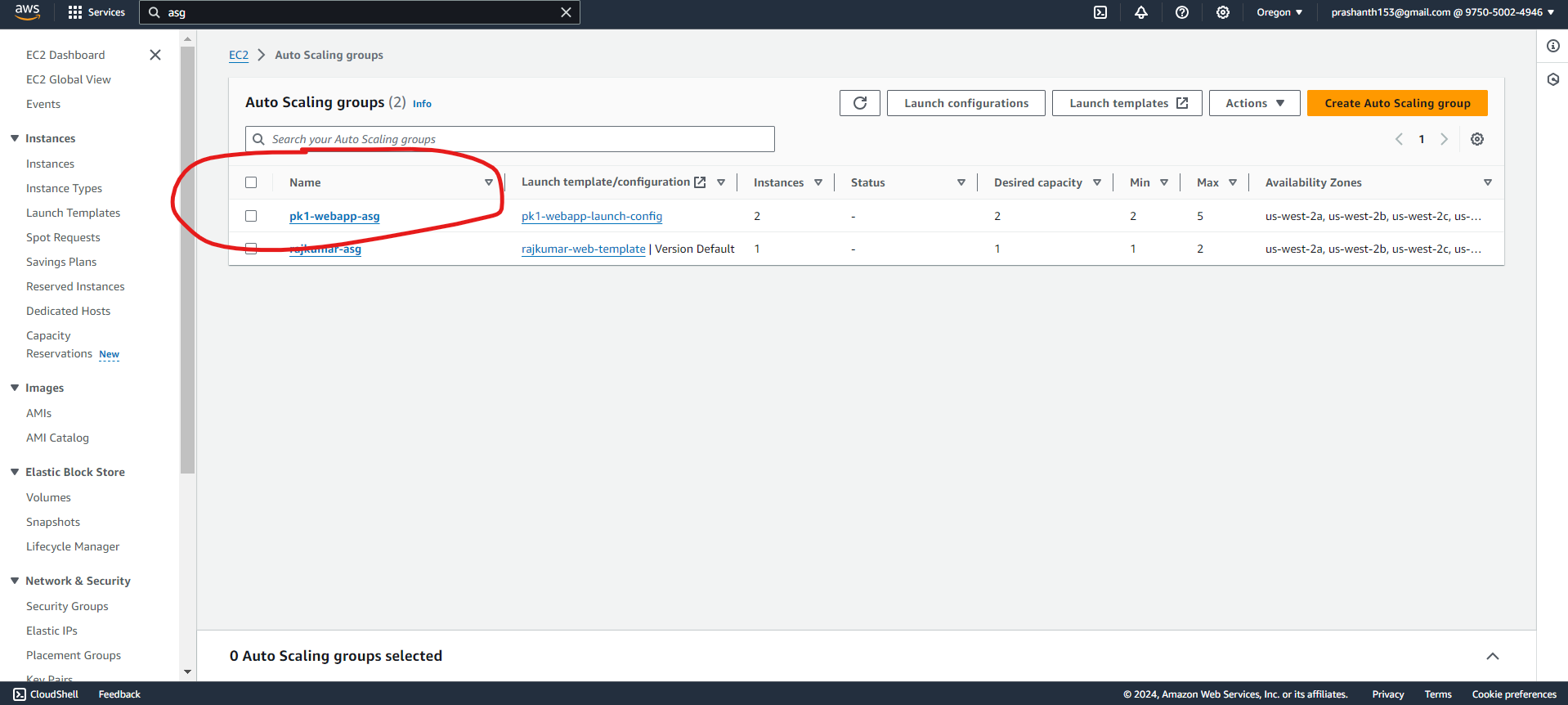




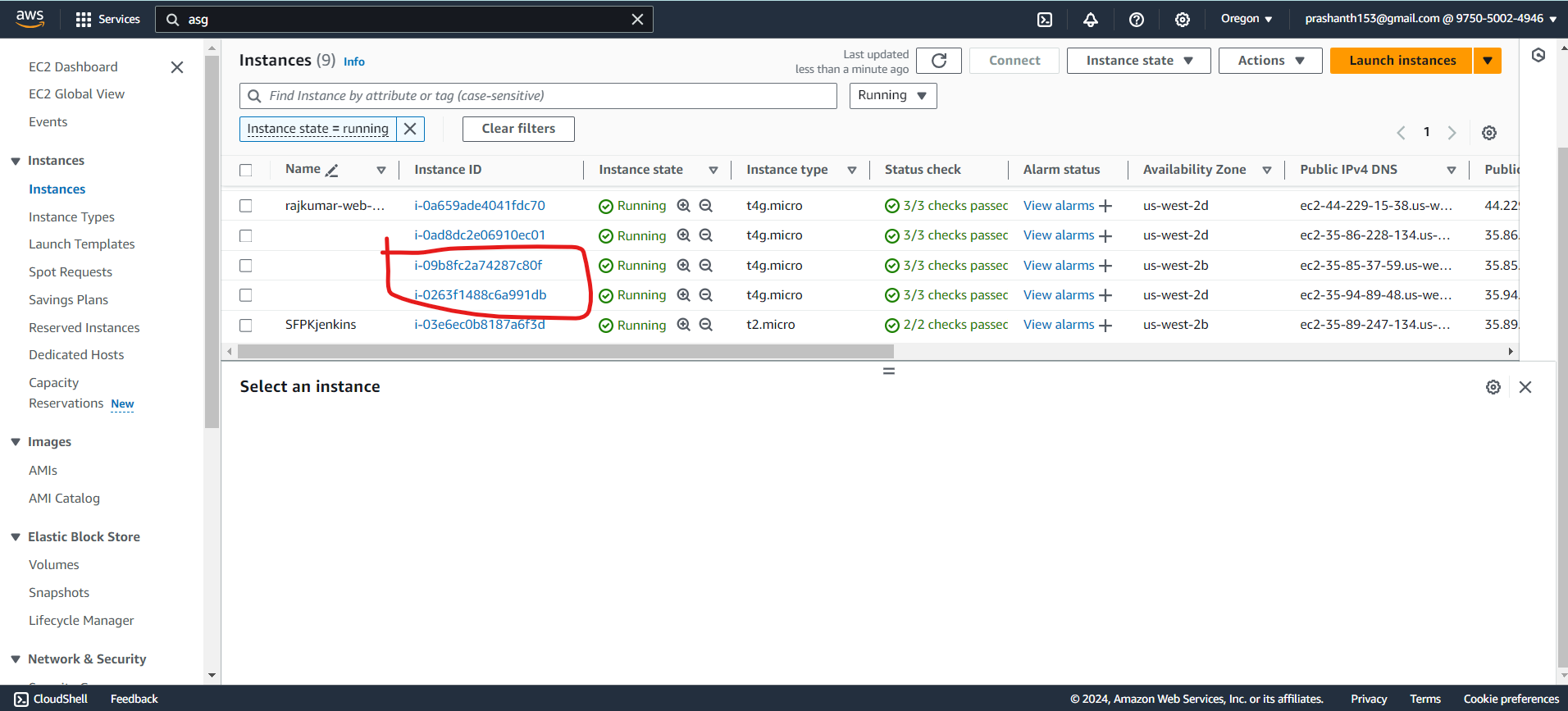
**5. Auto Scaling Group (ASG)**

* **Auto Scaling Group Name:** pk1-webapp-asg  
  The ASG automatically adjusts the number of running EC2 instances based on current traffic and load conditions. It maintains a specified minimum and maximum number of instances, ensuring efficient resource usage while providing scalability for the application.

***The ASG is created with the desired capacity as 2 and they are created from a Ubuntu image with already the webserver nginx running on the same…***

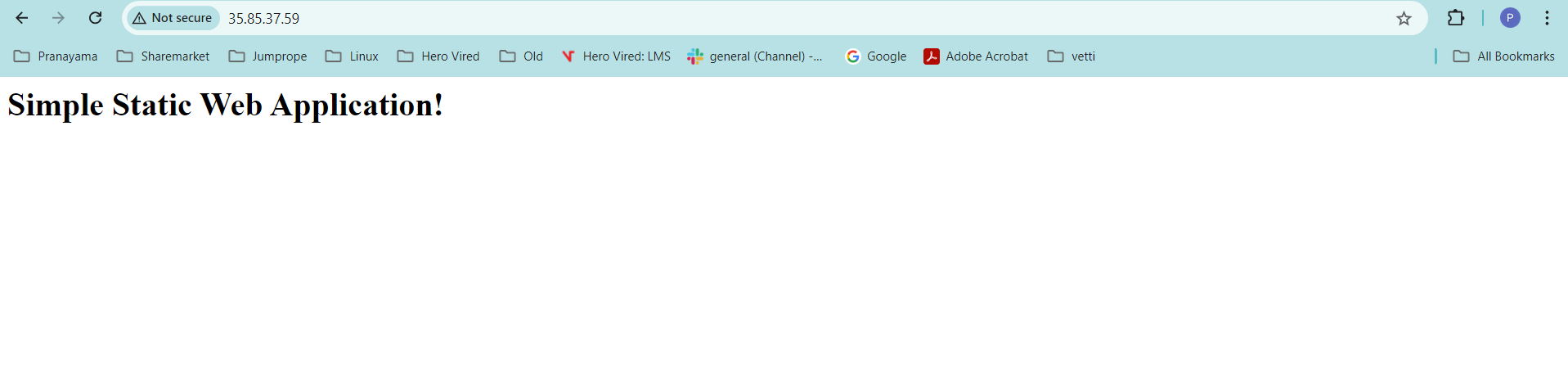


***Ec2 instances are active and running in the asg as shown in the below screenshot***

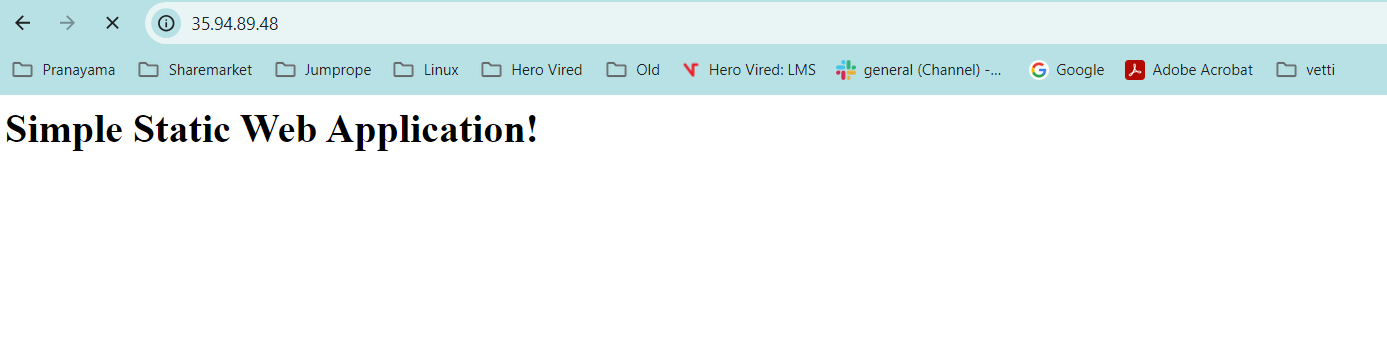
******

***When I go into the instances and use the public ip the webapp should be displayed***

***Instance 1: webapp is displayed***

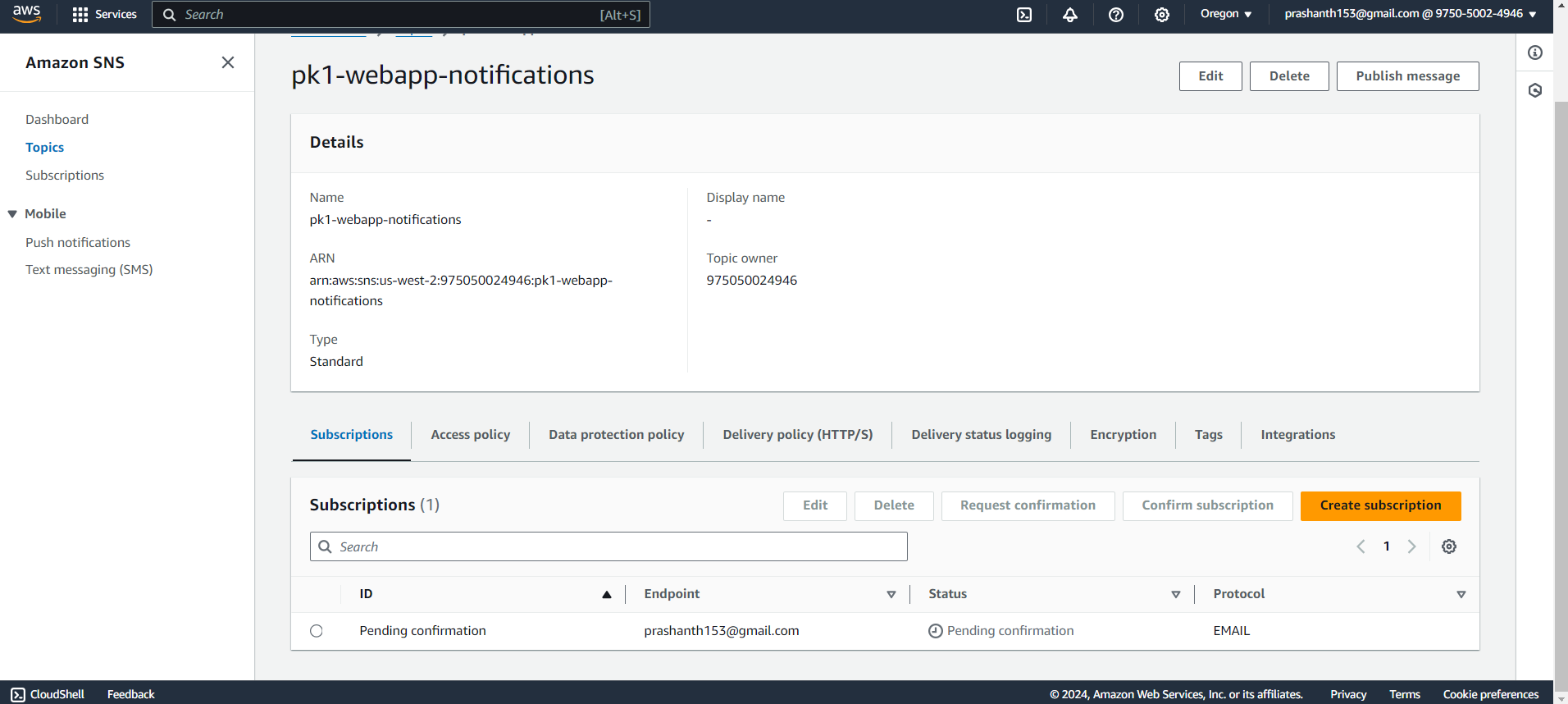
******

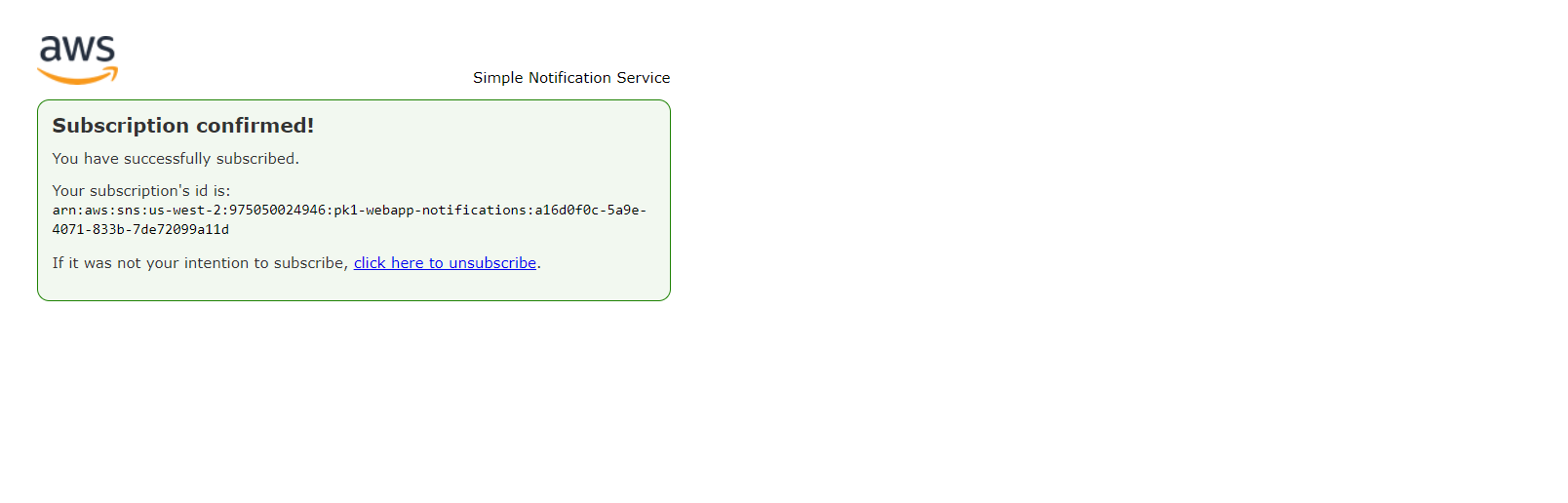
***Instance 2: webapp is displayed***

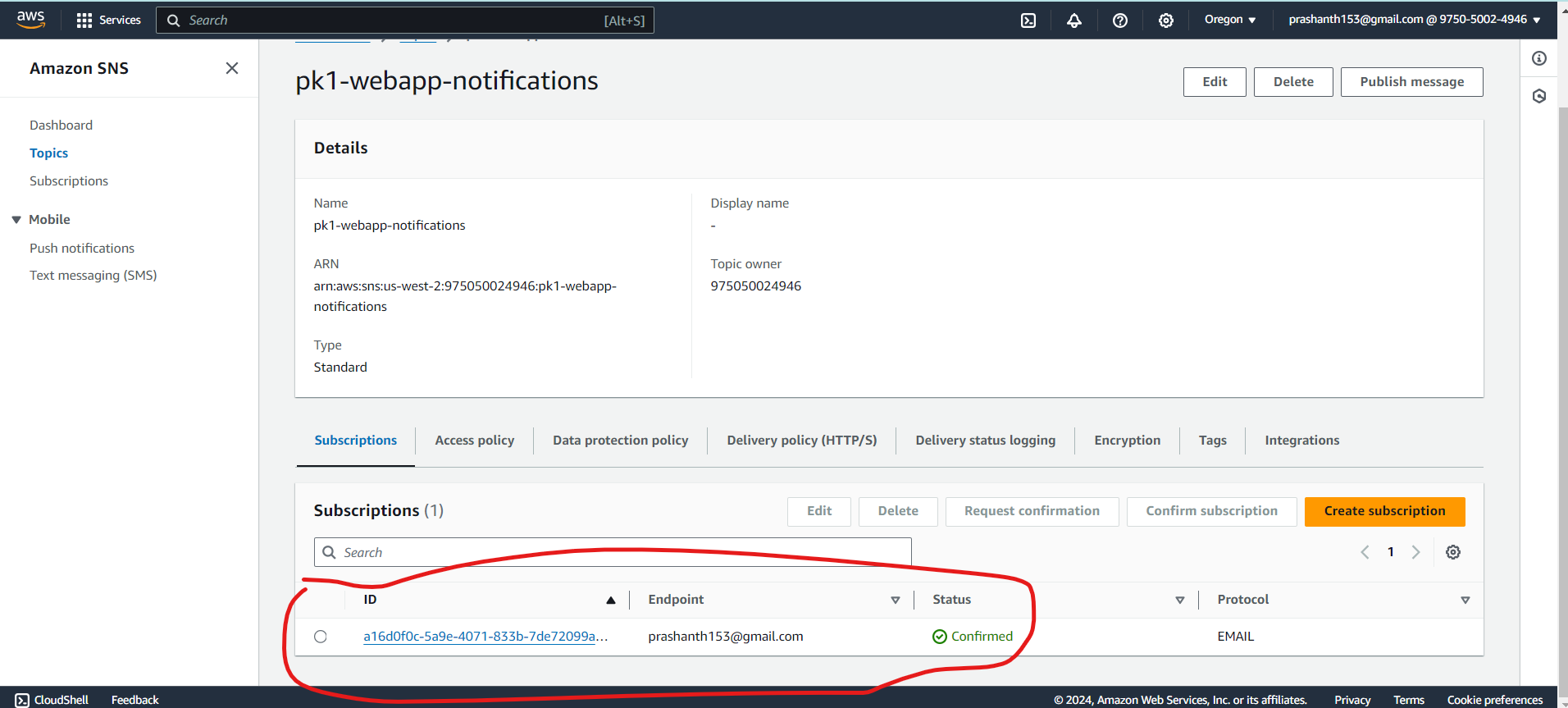
***nstan***

***So the ASG is setup with the Ec2 instance…even if I terminate the instances it is recreated again due to this.***

**6. SNS Notifications**

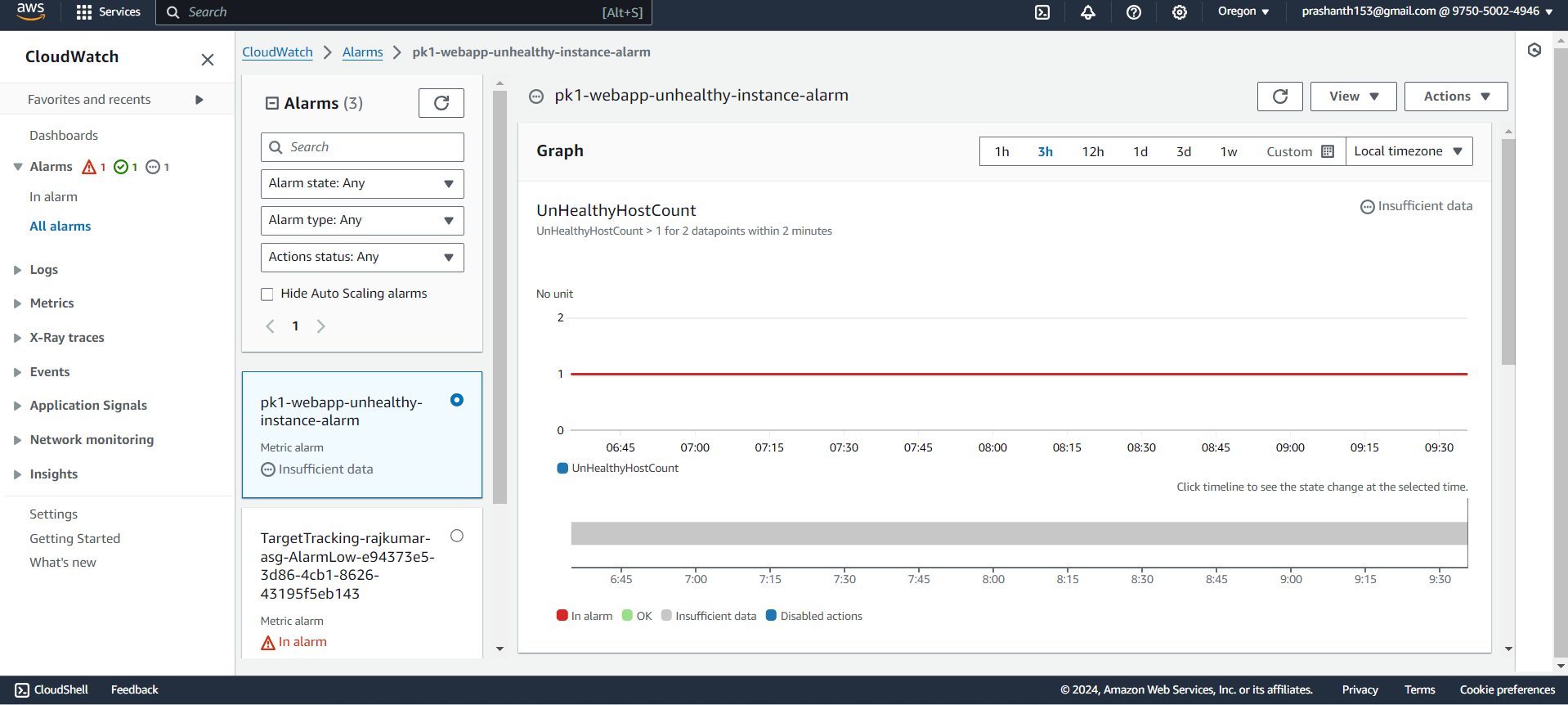
* **Topic Name:** pk1-webapp-notifications  
  This SNS topic is set up to send notifications regarding critical events, such as instance health changes and scaling actions. Administrators subscribed to this topic will receive email alerts, helping them respond quickly to infrastructure issues.
* ***SNS topic has been setup as below***
* ******
* ***Currently the confirmation is pending and let me do the same.. I have confirmed on the subcription***

******

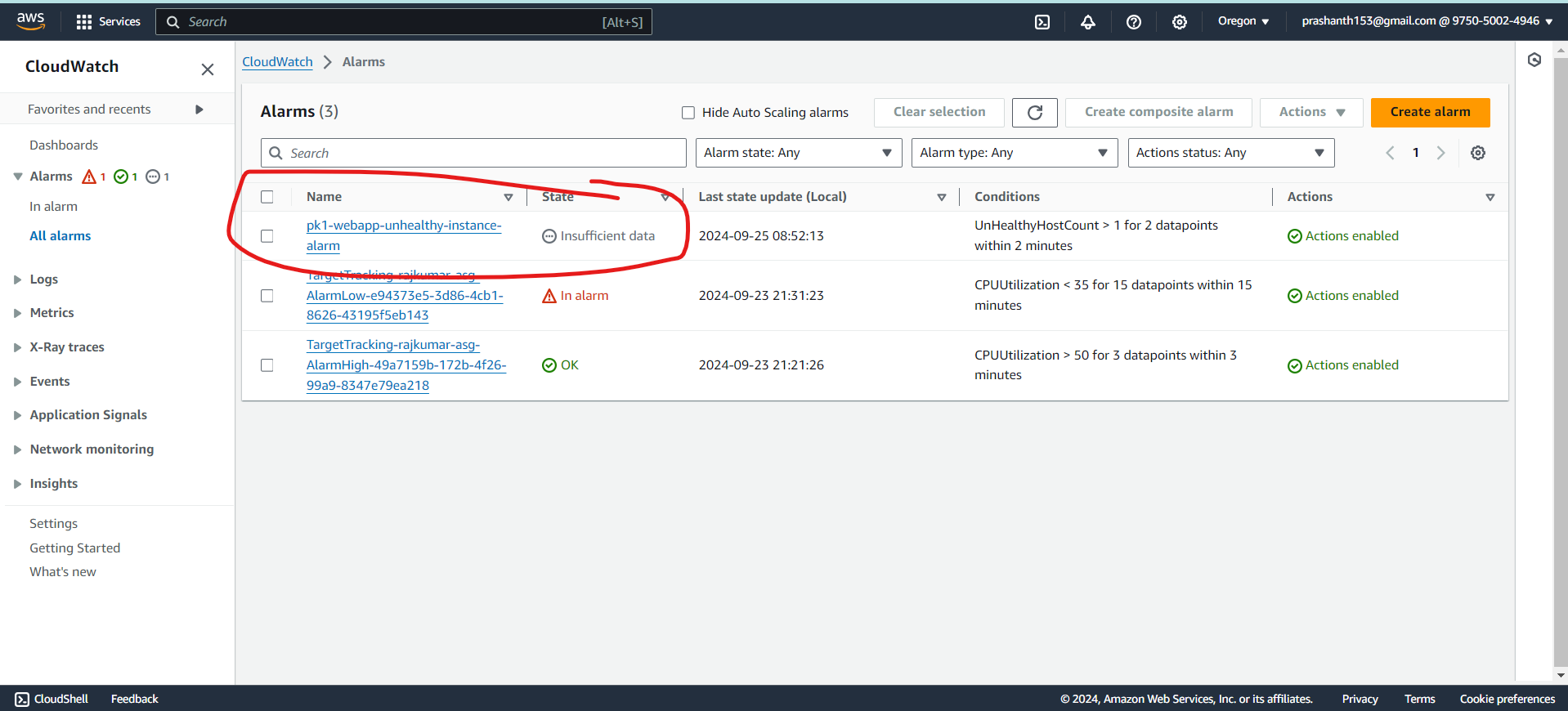
* ***Subscription is confirmed now as shown below***
* 

**7. CloudWatch Alarm**

* **Alarm Name:** pk1-webapp-unhealthy-instance-alarm  
  The CloudWatch alarm monitors the health of instances in the target group, triggering notifications when the number of unhealthy hosts exceeds a defined threshold. This proactive monitoring helps maintain application reliability.
* ***Unhealthy alarm is created for the same so when the instance in the ASG is triggered the alarm will go on immediately***

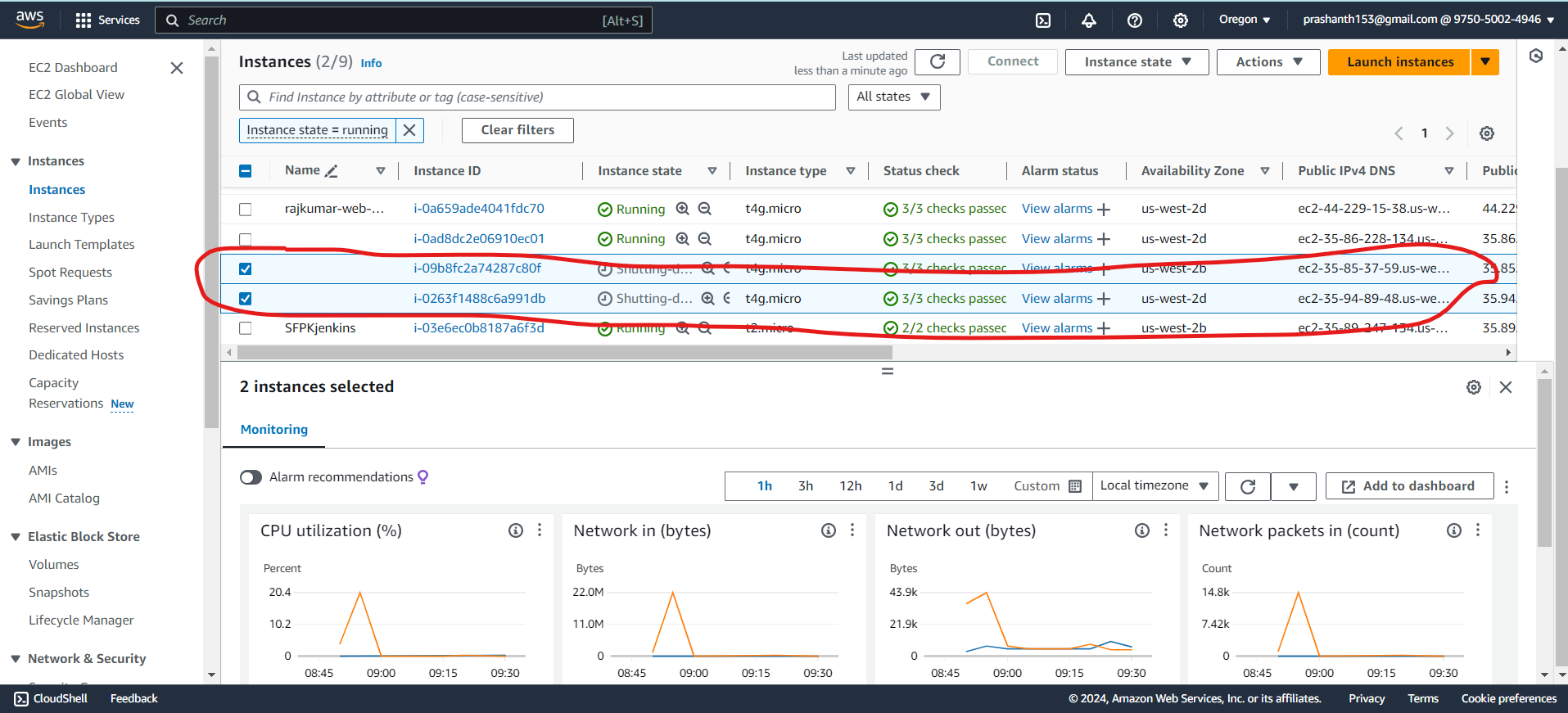


***Since it is only few minutes the alarm didn’t trigger yet and it says insufficient data as shown below.***

******

***IM going to manually terminate the instances to see if the alarm becomes active as the host will be inactive for sometime till the ASG recreates them.***

***Terminating the instances manually now. As shown below I have terminated the instances manually to trigger the alarm on.***

******

**Tear down script**

***I have separately created this script to tear down any resources which have been created and is working fine.***