Assignment 2
COS20019
Developing a Highly available Photo Album website

Student name: Tran Thien Thao Vy

Student ID: 104991221

Swinburne University of Technology

School of Science, Computing and Engineering Technologies

I. Introduction

This paper describes an advanced AWS infrastructure designed to enhance the interactions between EC2, Lambda, and S3 services. Key objectives include the development of a Lambda function, the use of custom AMIs, the implementation of auto scaling with launch configurations, the application of elastic load balancers, and the enforcement of access control through AWS Network Access Control Lists (NACLs) and S3 bucket policies. By achieving these objectives, the infrastructure demonstrates significant improvements in security, scalability, and performance, thereby enabling more effective collaboration among EC2, Lambda, and S3 services within the AWS cloud environment.

Keywords — Cloud Computing, System Architecture, EC2, S3, VPC, Lambda.

II. Website Infrastructure

1. Basic Infrastructure requirements:

The VPC is as per Assignment 1b.

Instead of using the NAT instance, I deploy the NAT gateway as my Learner Lab environment also includes NAT gateway. It is way more convenient yet brings the same effect to the web functionality.

There are two AZs to configure. In specific, NAT is resided in the public-1 subnet, public-2 is for Dev Instance, we can use private-1 and two for auto-scaling webservers which will be employed in the later part of the assignment.

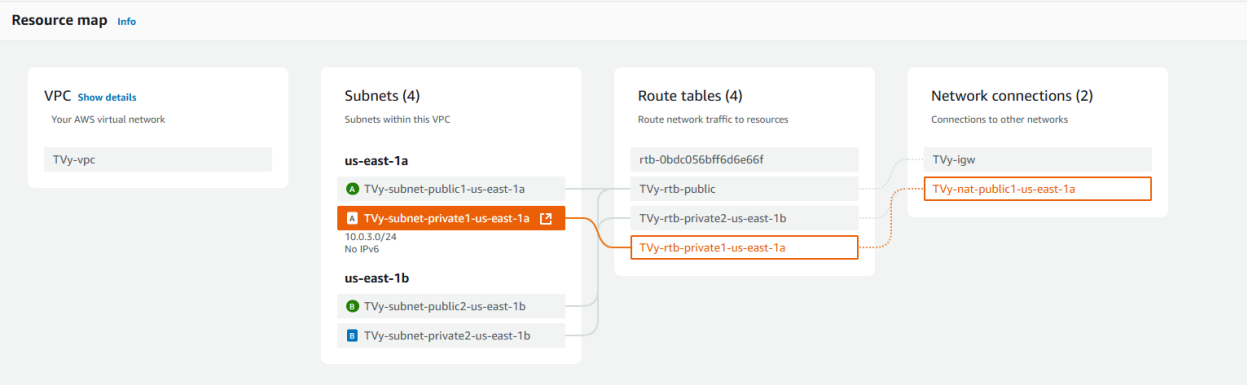


Figure 1: VPC resource map

Make sure that the NAT gateway will route to the Internet Gateway.

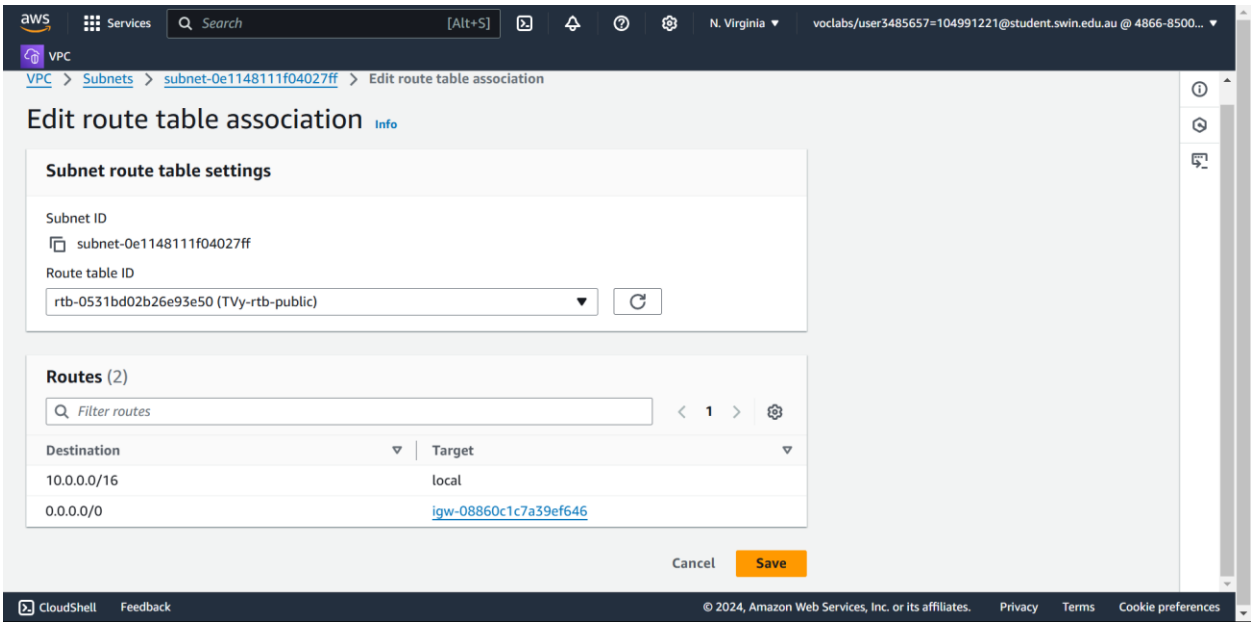


Figure 2: NAT subnet route table association

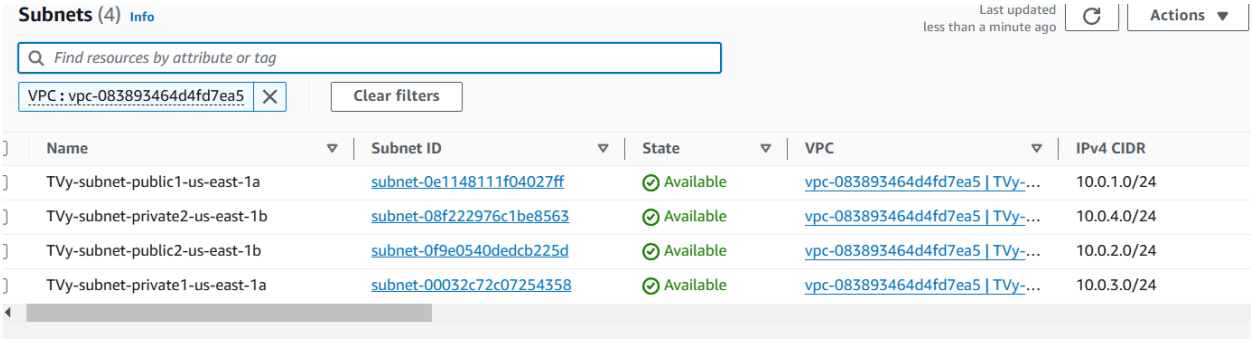


Figure 3: subnets of TVy-VPC

Continue to configure the Dev Instance, which stores most AWS PHP SDK (configured by SSH), Apache web server (configured by user-data), source code of the website (configured by transferred through WinSCP).

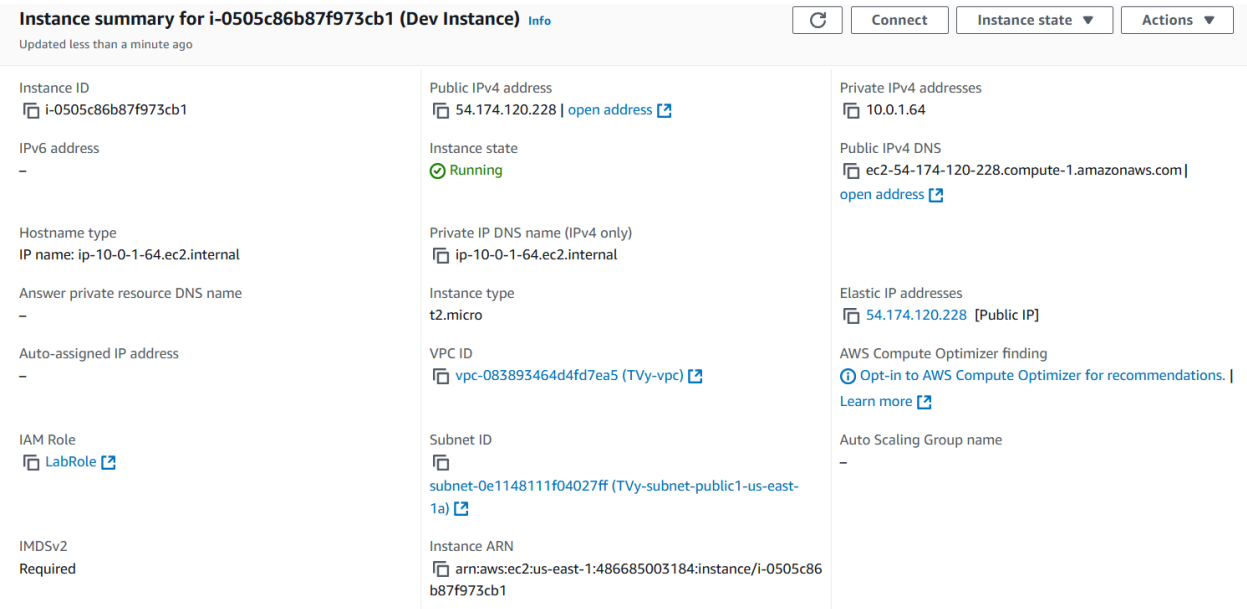


Figure 4: Dev Instance configuration

Make sure the public DNS and Apache works by access to the browser through public DNS.

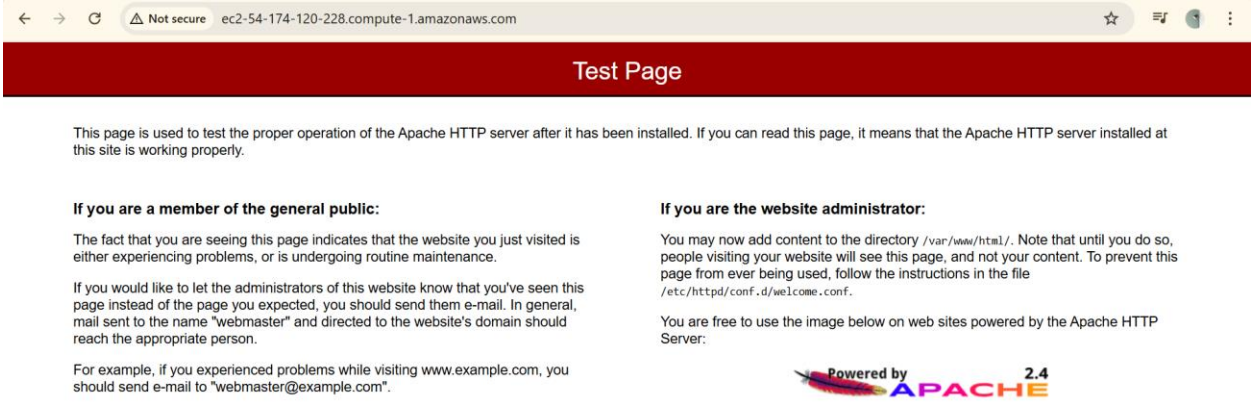


Figure 5: Test page of Dev instance

As things are settled, we can continue to configure the database section. The first thing is to SSH to Dev Instance and download “phpmyadmin” by following the same instruction as per assignment 1a.

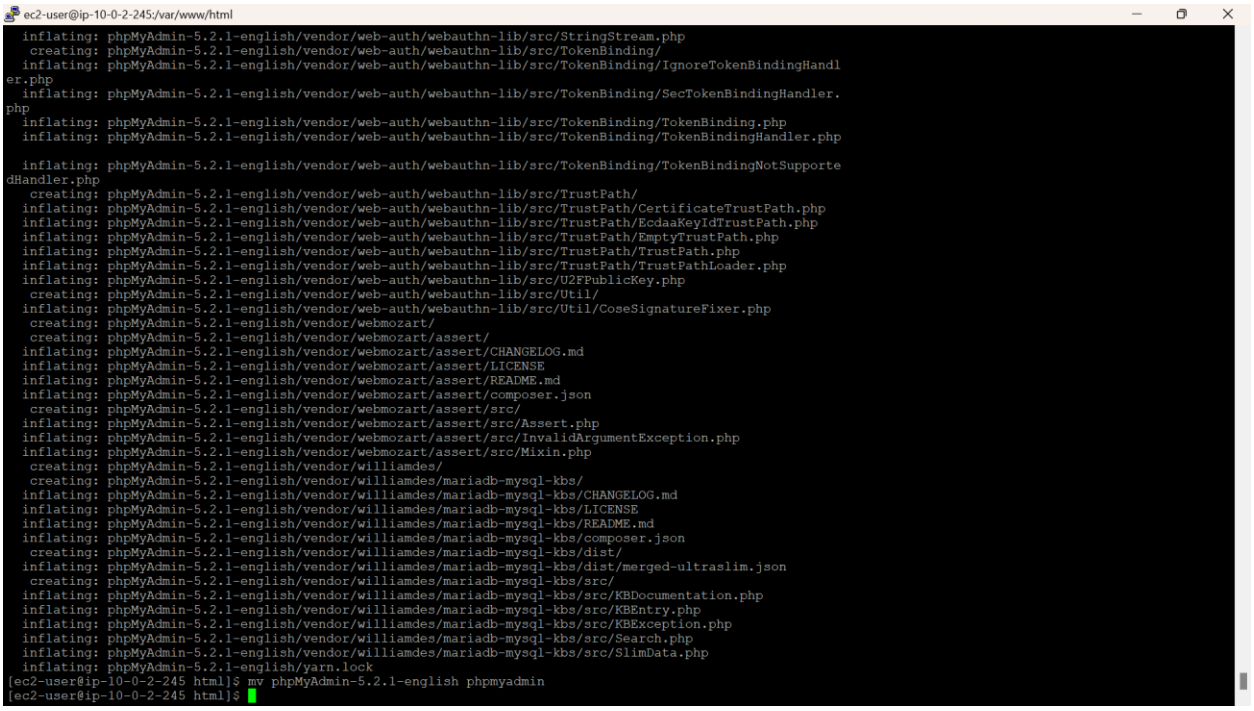
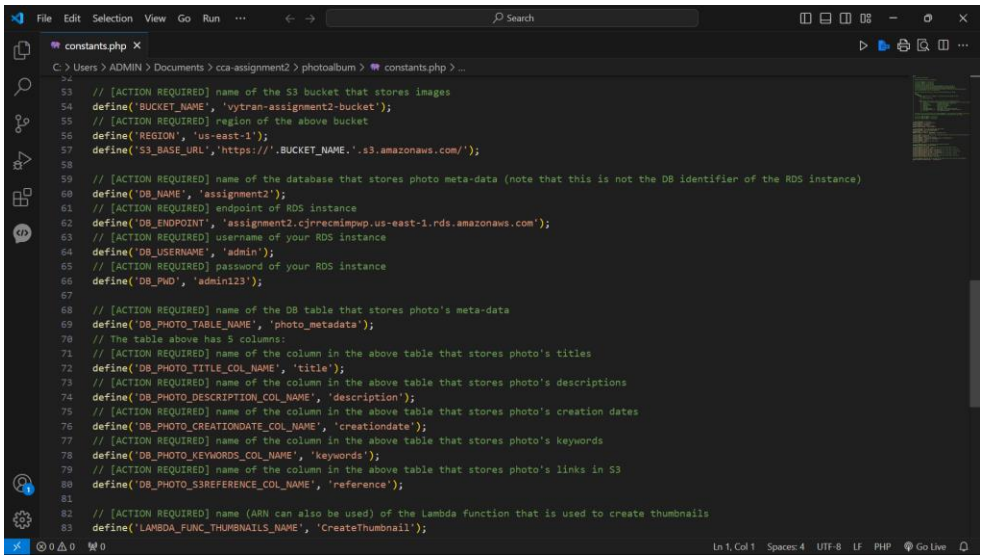


Figure 6: PuTTY screen of phpmyadmin download

After that, we need to specialize some data in constants.file.



Log into phpMyAdmin page through Dev Instance EC2 DNS and create the photos' metadata table.

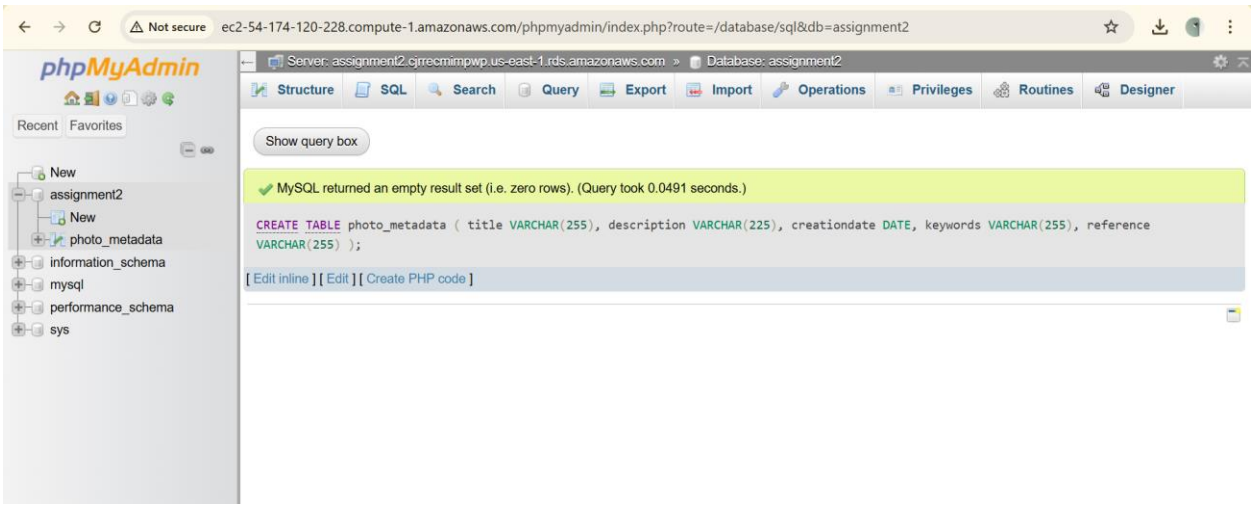


Figure 7: phpMyAdmin screen

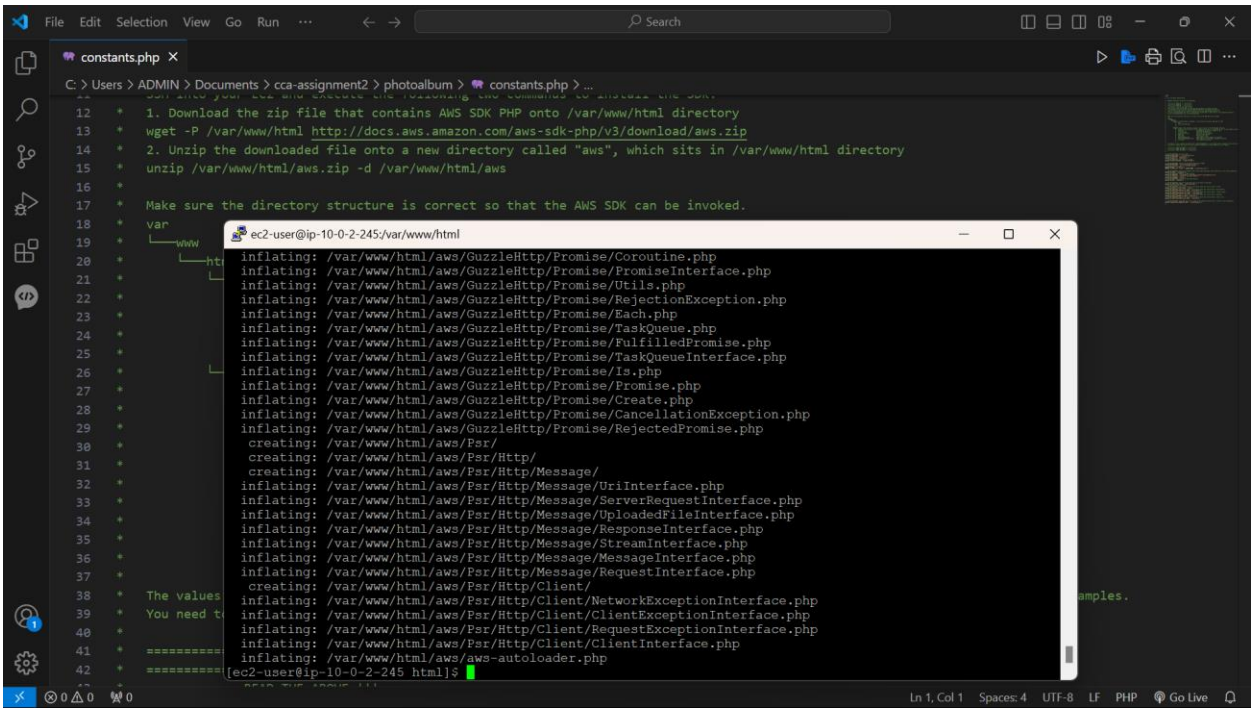


Figure 8: AWS SDK package through SSH

For the website to work, we need to download AWS SDK package through SSH into the Dev Instance. Then transfer the files to WinSCP.

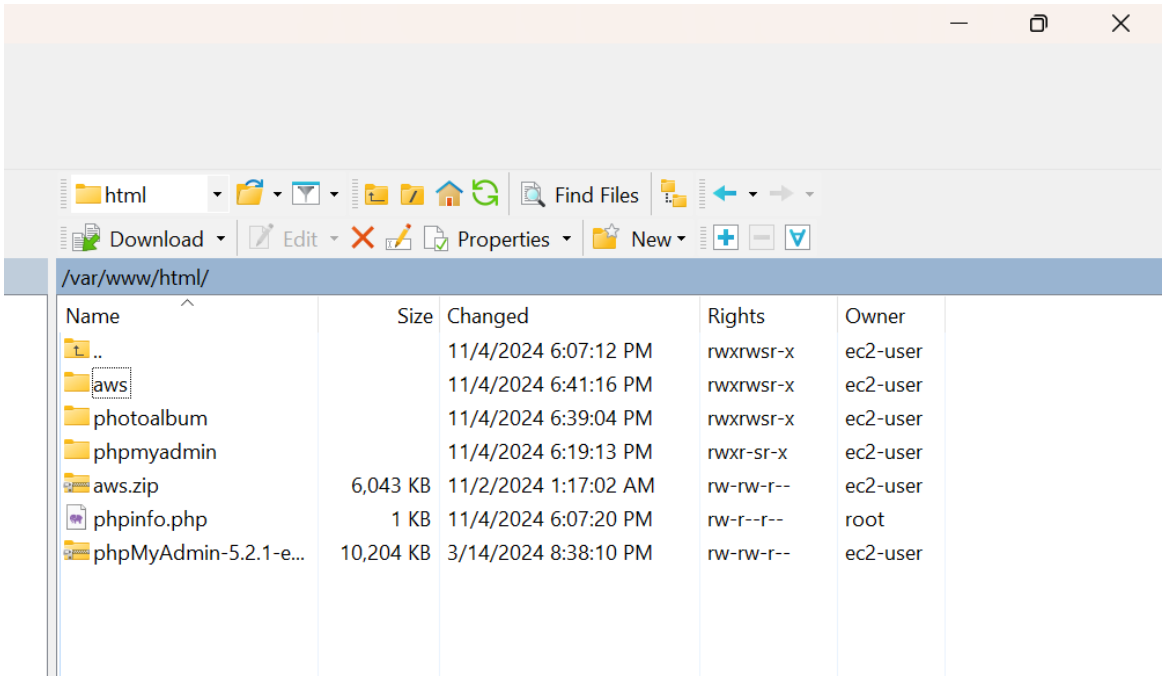


Figure 9: WinSCP directories in the correct arrangement.

After the photoalbum php files and AWS SDK files and phpMyAdmin are settled in WinSCP. We can connect to the website through Dev Instance public DNS (make sure the instance is associated with an elastic IP address for further use).

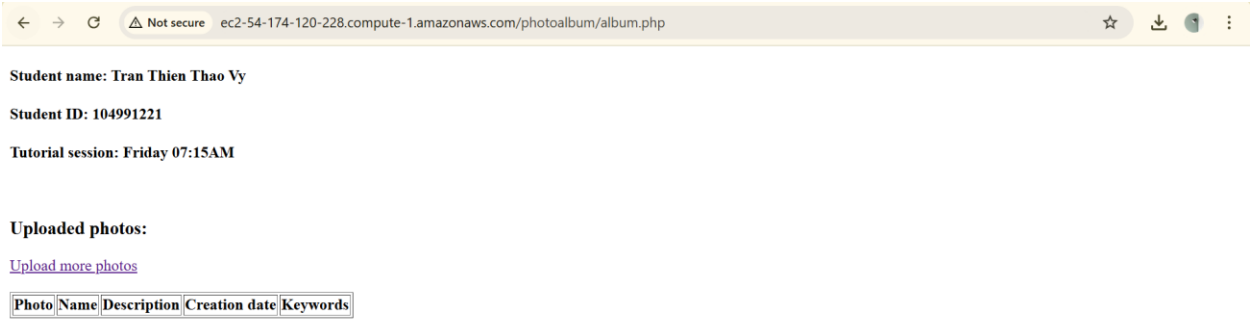


Figure 10: Website screen through ec2 public DNS access.

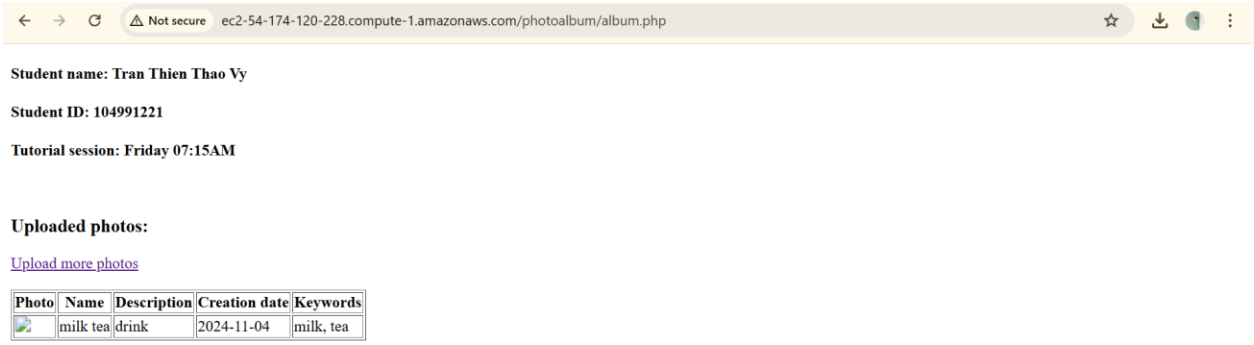


Figure 11: Meta-data polulated.

S3 is used for photo storage. It is built the same way as Assignment 1b, however we need to make sure that bucket policy and permissions are modified correctly.

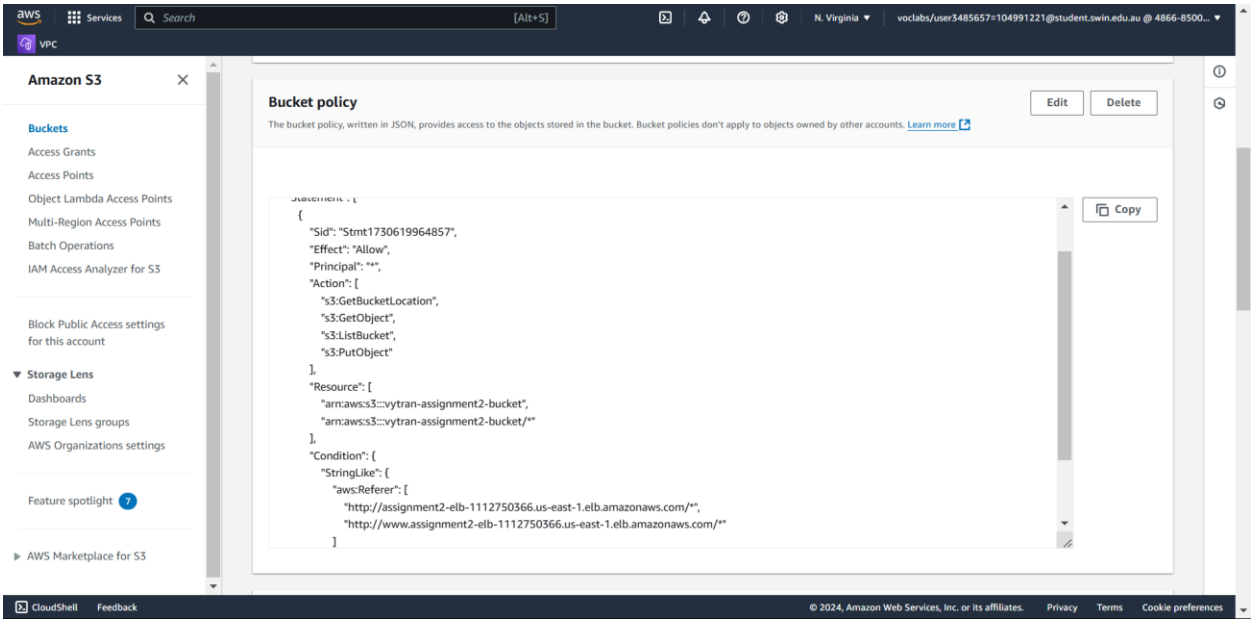


Figure 12: S3 bucket policy

After configuring Lambda function by uploading the zip file and configure its IAM role, we can run a test to check if S3 and Lambda are working well.

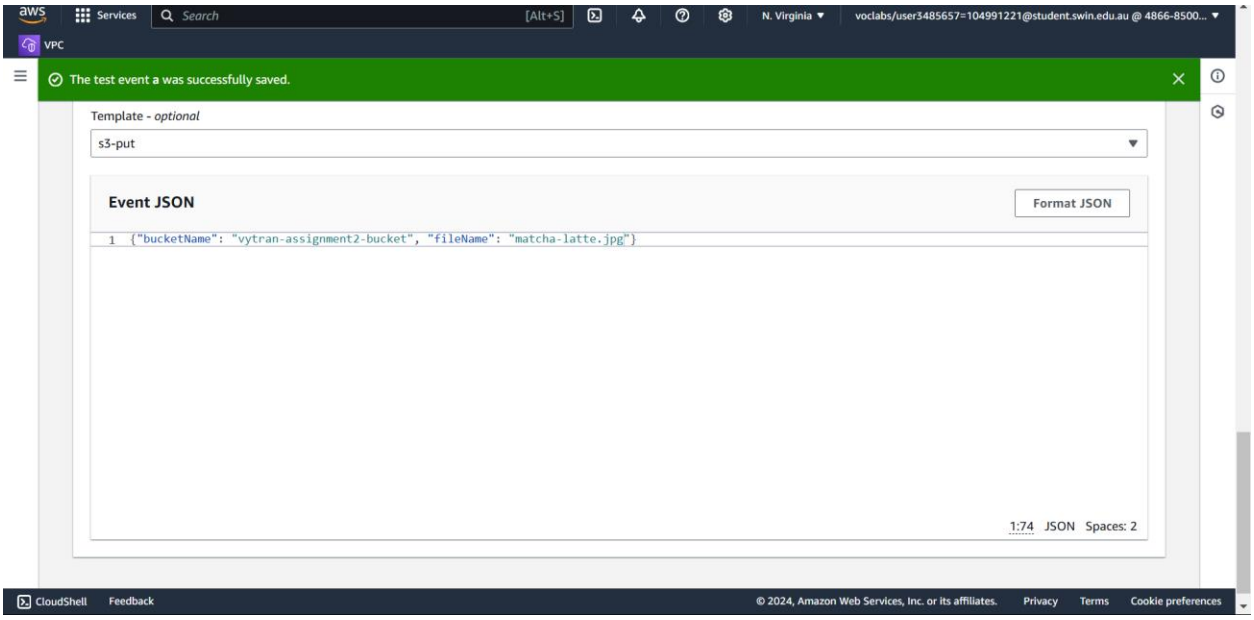


Figure 13: Lambda code to test a case

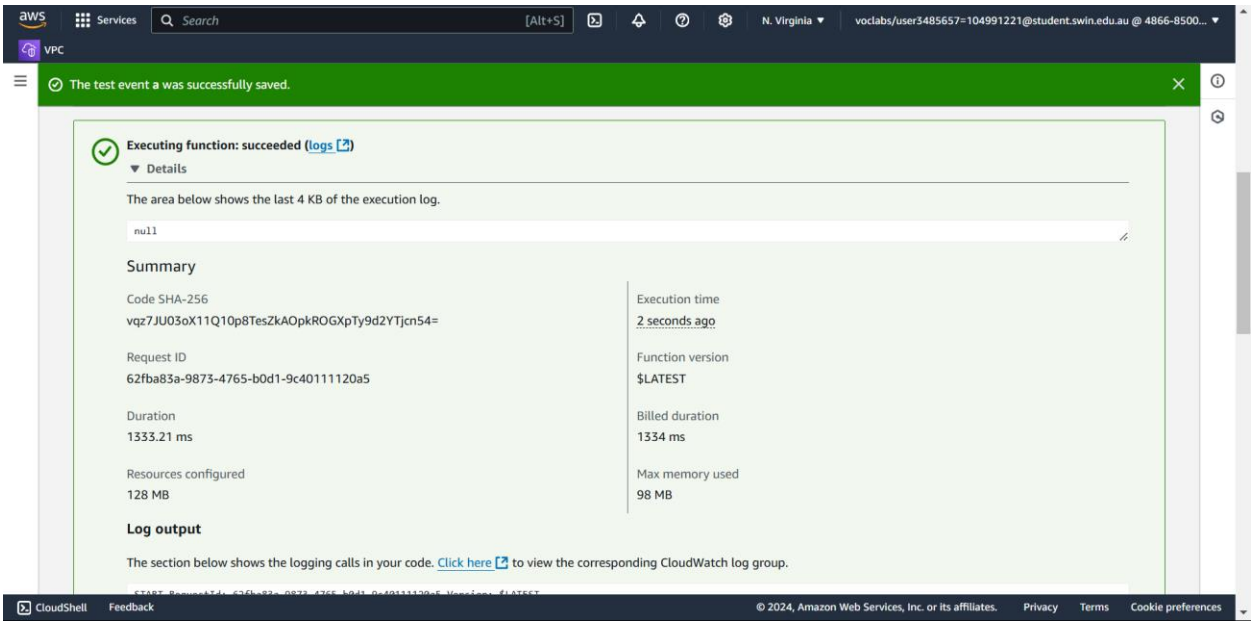


Figure 14: test succeeded.

Therefore, the meta-data of the “milk tea” object photo will be a resized jpg.

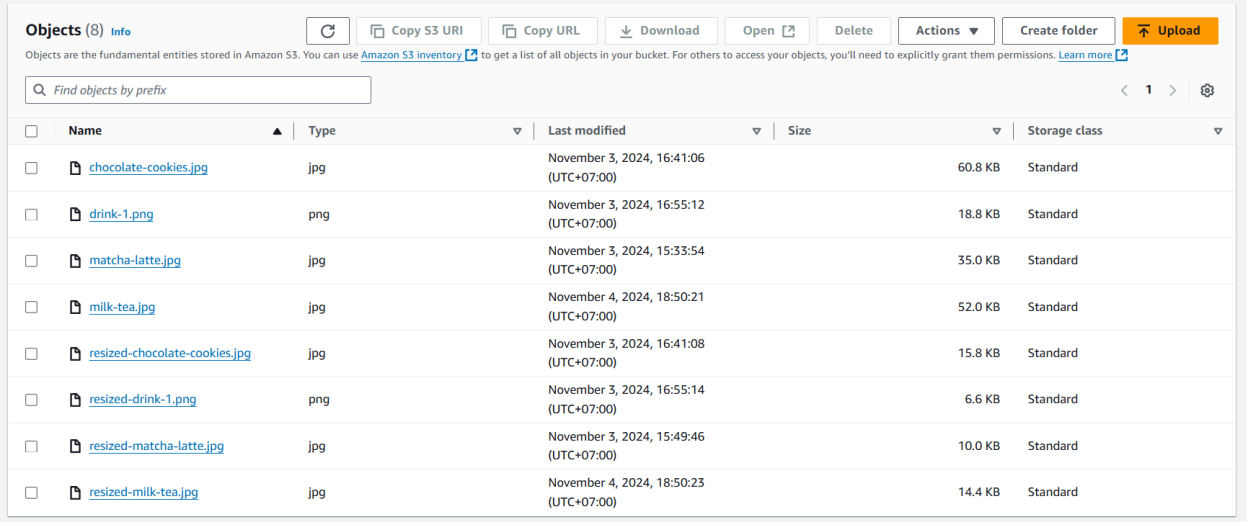


Figure 15: S3 objects.

After the Dev Instance configuration, which works properly, we can continue create the custom AMI, which can be used for the webservers.

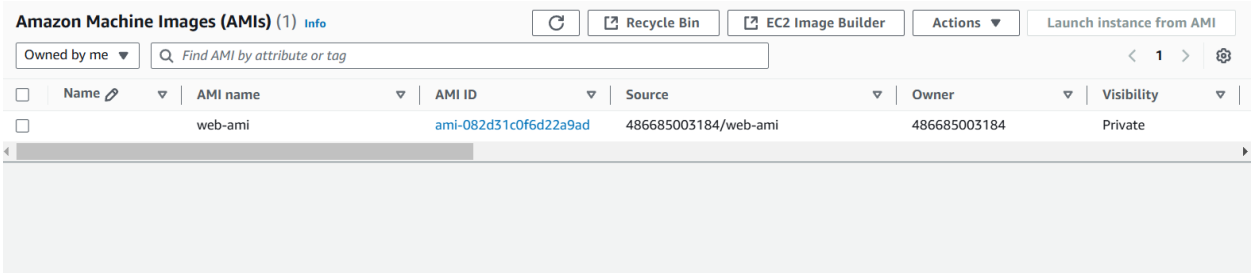
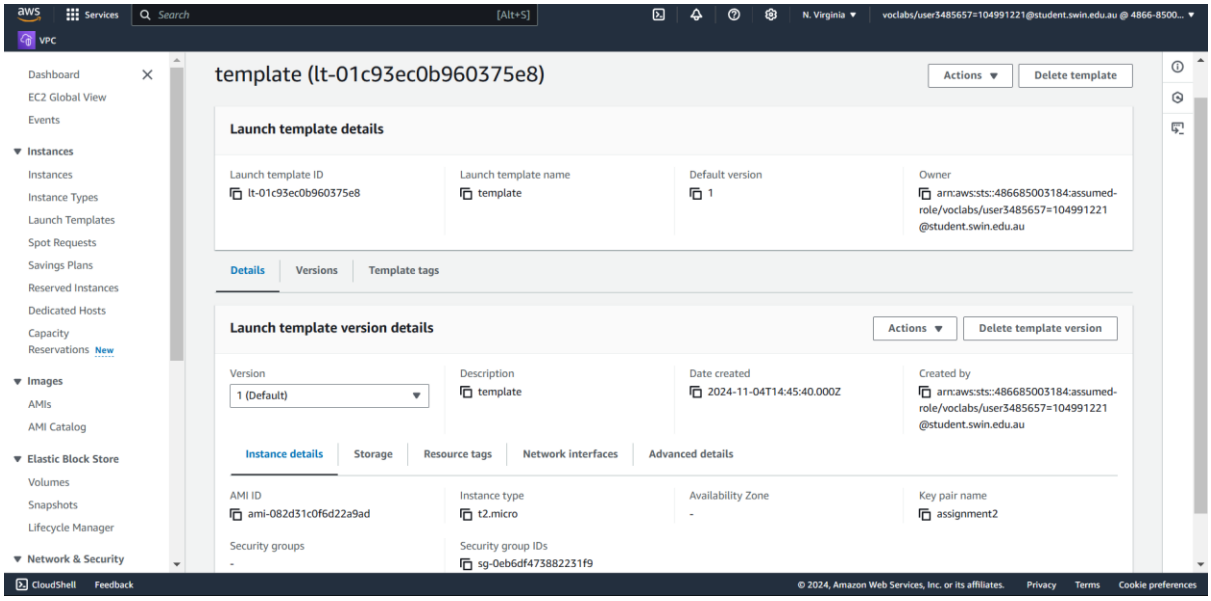


Figure 16: custom AMI for web servers.

2. Developing a highly available website by configuring a load balancer and an auto scaling group.

We firstly need to launch a template for the web servers, in which the template has to employ the customed AMI.



Details of the launched template.

After that, we need to configure a load balancer and an auto scaling group as well.

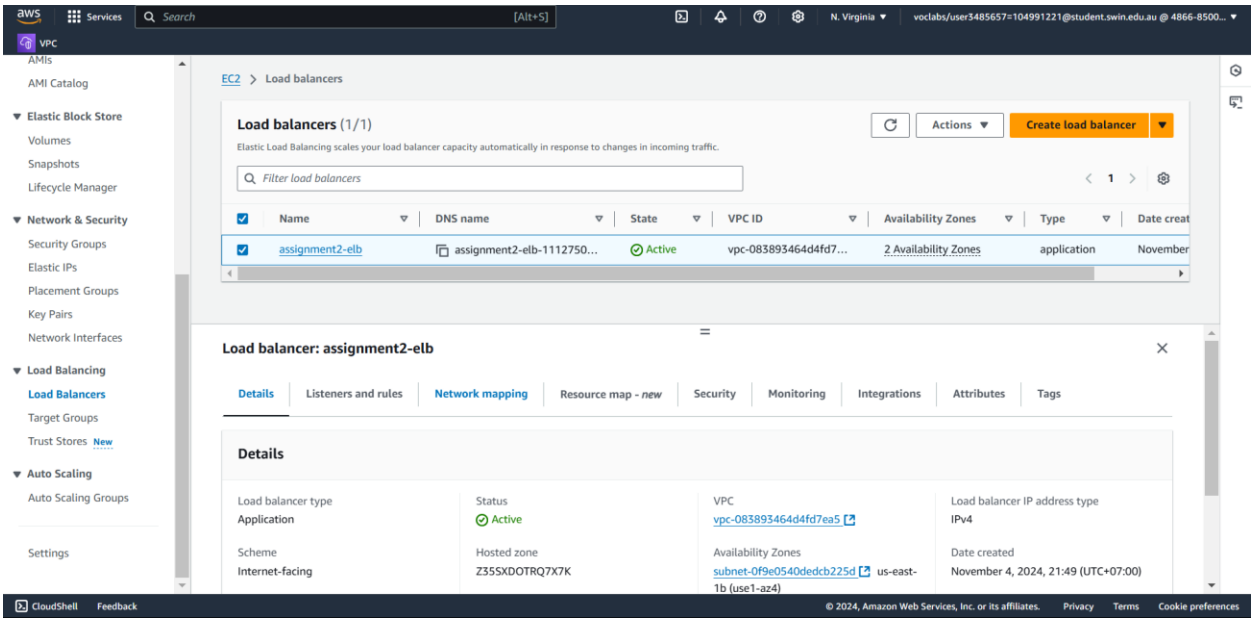


Figure 18: load balancer for web servers.

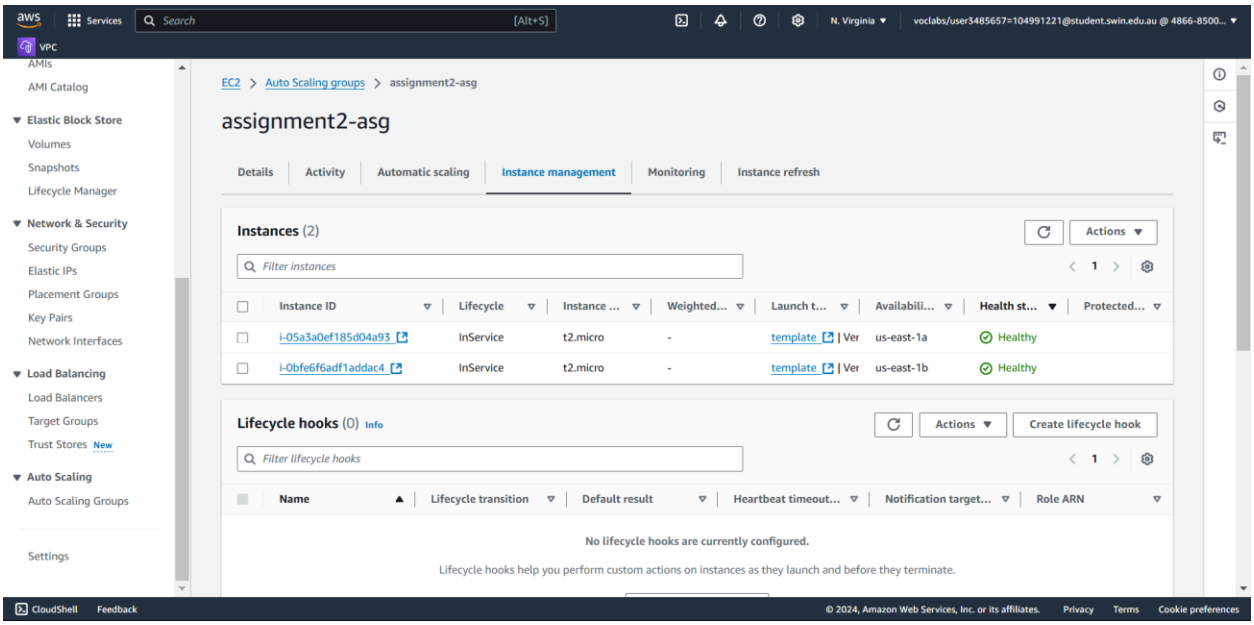


Figure 19: Make sure the registered targets are healthy.

After all configurations, which are supposing-ly correct, we can access the website through the ELB public ARN.

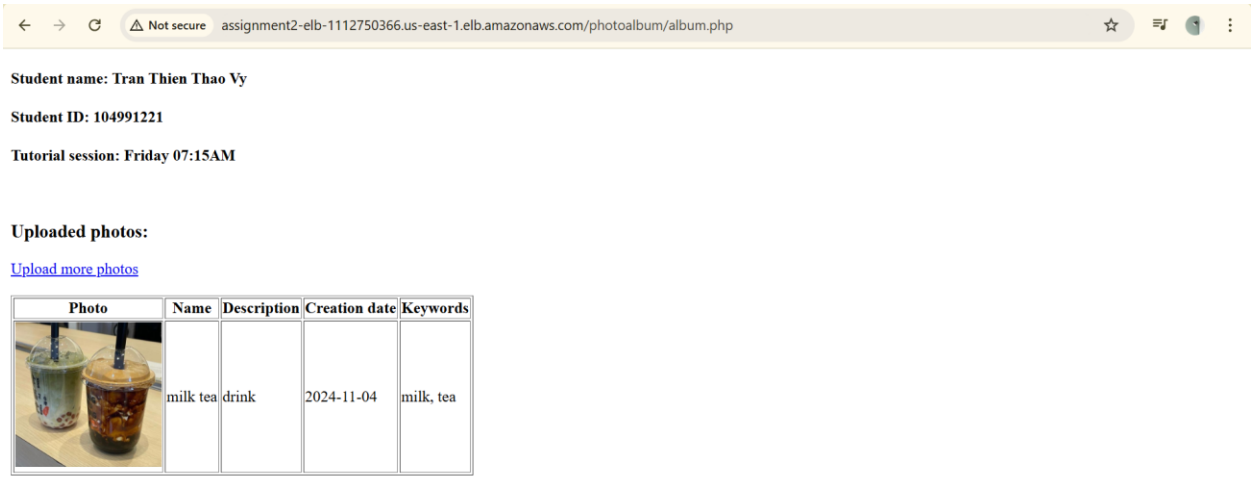


Figure 20: album.php Webpage

We can upload photos and photos' meta-data in the uploader page.

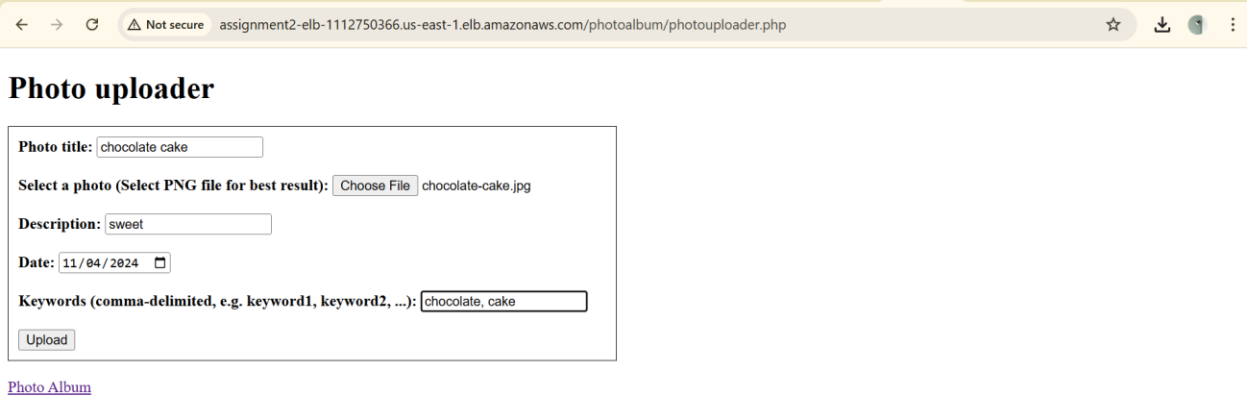


Figure 21: photouploader.php webpage

After uploading some photos, here is the webpage.

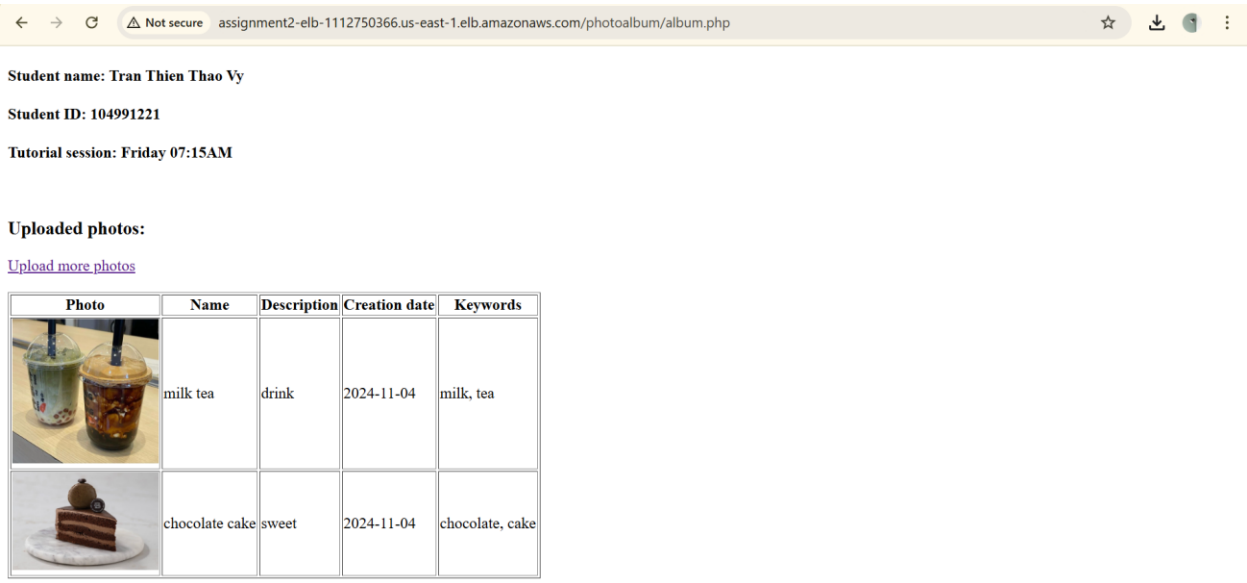
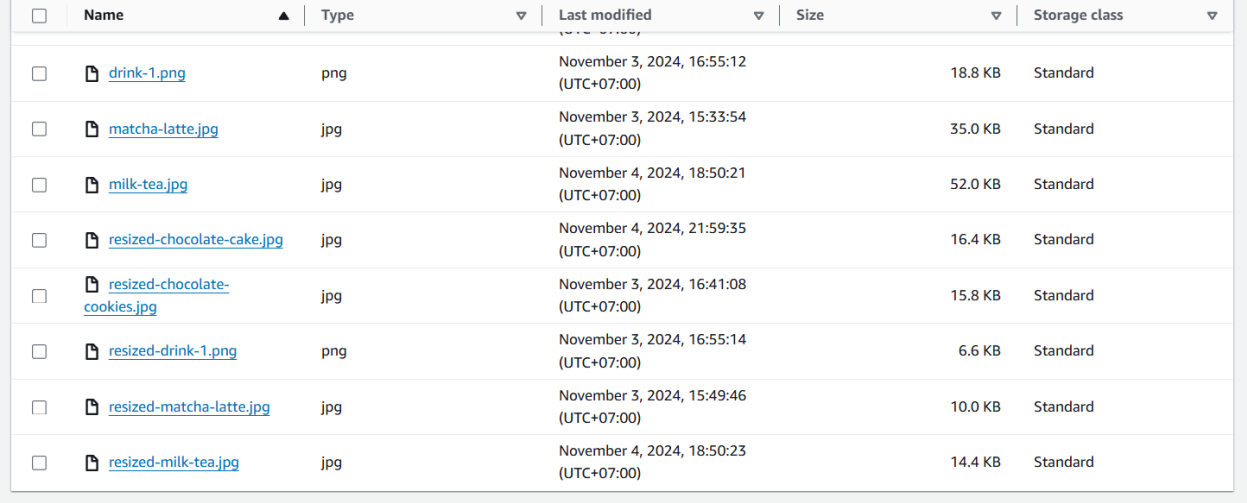


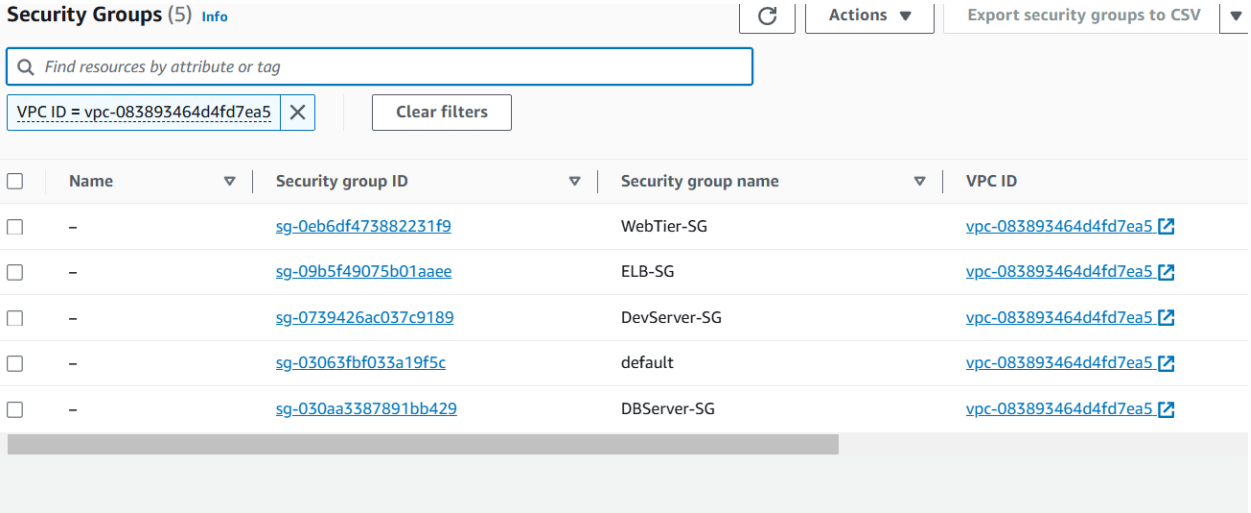
Figure 21: album.php webpage



Name	Type	Last modified	Size	Storage class
drink-1.png	png	November 3, 2024, 16:55:12 (UTC+07:00)	18.8 KB	Standard
matcha-latte.jpg	jpg	November 3, 2024, 15:33:54 (UTC+07:00)	35.0 KB	Standard
milk-tea.jpg	jpg	November 4, 2024, 18:50:21 (UTC+07:00)	52.0 KB	Standard
resized-chocolate-cake.jpg	jpg	November 4, 2024, 21:59:35 (UTC+07:00)	16.4 KB	Standard
resized-chocolate-cookies.jpg	jpg	November 3, 2024, 16:41:08 (UTC+07:00)	15.8 KB	Standard
resized-drink-1.png	png	November 3, 2024, 16:55:14 (UTC+07:00)	6.6 KB	Standard
resized-matcha-latte.jpg	jpg	November 3, 2024, 15:49:46 (UTC+07:00)	10.0 KB	Standard
resized-milk-tea.jpg	jpg	November 4, 2024, 18:50:23 (UTC+07:00)	14.4 KB	Standard

Figure 22: S3 bucket objects

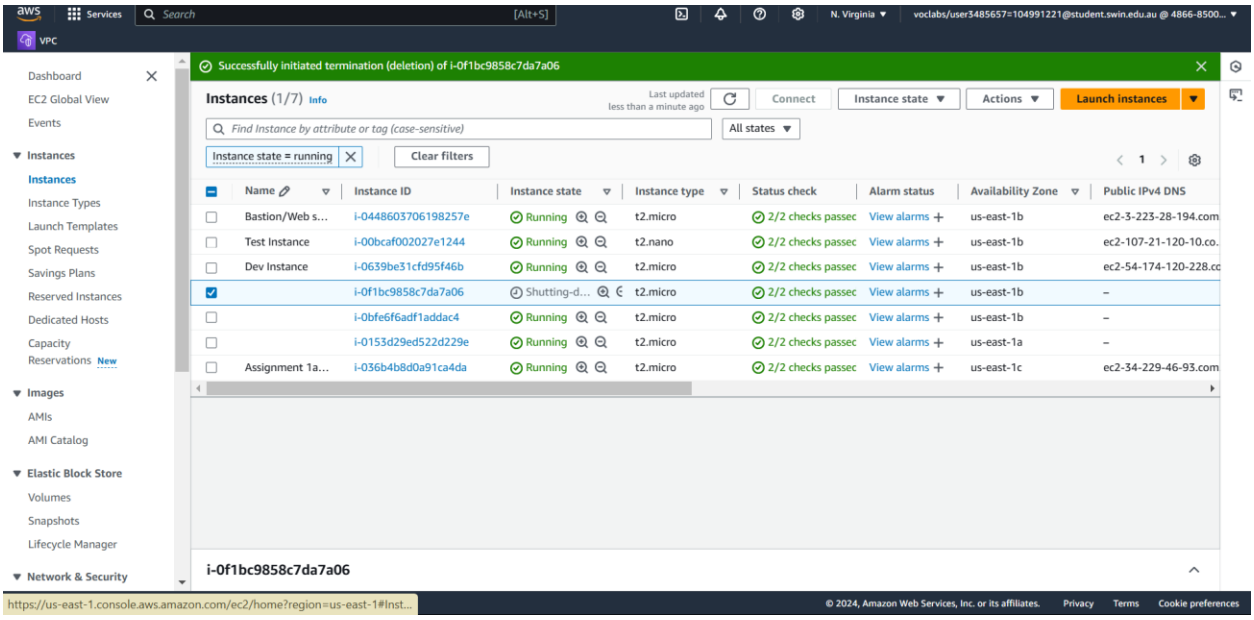
The jpg photo data are resized after being forwarded to S3 by the Lambda function from the EC2 instance.

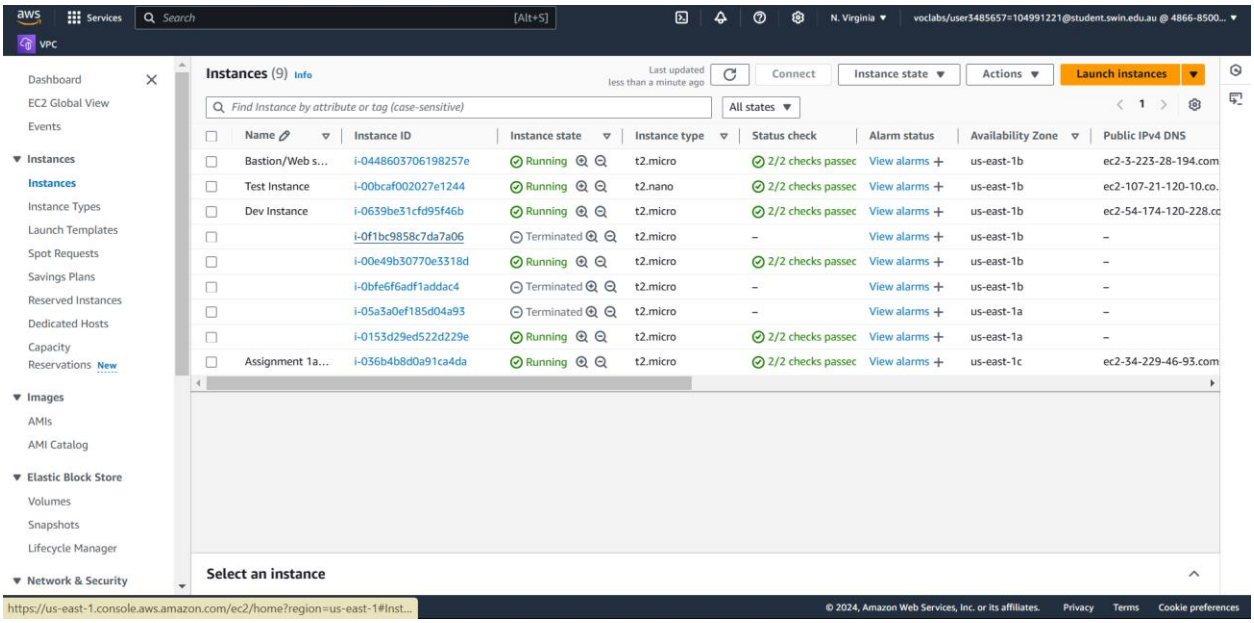


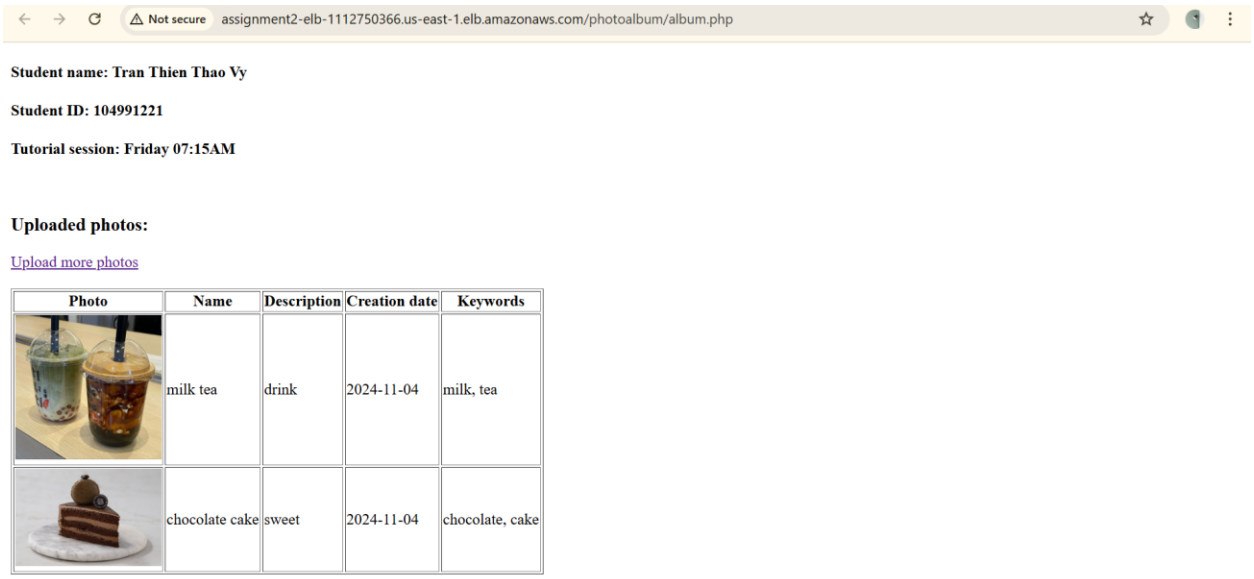
Rules of security groups are configure as following:

- DevServer-SG: Inbounds from all traffic and outbounds to all traffic.
- WebTier-SG: Inbounds from ELB-SG and outbound to the NAT gateway.
- ELB-SG: Inbounds from Internet gateway and outbounds to IGW.
- DBServer-SG: Inbounds from/outbounds toWebTier and DevServer.

We can test the high availability of the instances by terminating some instances and the other instances will be initialized and the website will continue to work:


Successfully initiated termination (deletion) of i-0f1bc9858c7da7a06
Instances (1/7) Info
Launch instances
Instance state = running
Clear filters
i-0f1bc9858c7da7a06


Instances (9) Info
Launch instances
Select an instance



III. Additional information for marking

Album.php ELB link: http://assignment2-elb-1112750366.us-east-1.elb.amazonaws.com/photoalbum/album.php

Dev Instance EC2 link: http://ec2-54-174-120-228.compute-1.amazonaws.com/

PhpMyAdmin link: http://ec2-54-174-120-228.compute-1.amazonaws.com/phpmyadmin