COS20019-Cloud Computing Architecture

Assignment 2

Developing a highly available Photo Album website

Student Name: Doan Minh Hieu

Student ID: 104168106

2023-HX01-COS20019-Cloud Computing Architecture (Hanoi)-H1

I. Introduction

In this assignment, the infrastructure and program I developed for Assignment 1b will be improved upon in order to extend a highly available Photo Album website. I will also take a look at AWS PHP SDK and AWS Network ACLs so that I can integrate AWS services with a PHP application and apply a network security feature to manage the traffic flows in and out of a Virtual Private Cloud.

II. Creating a VPC (Virtual Private Cloud):

Firstly, I created a VPC that followed the requirements so as to launch other AWS resources.

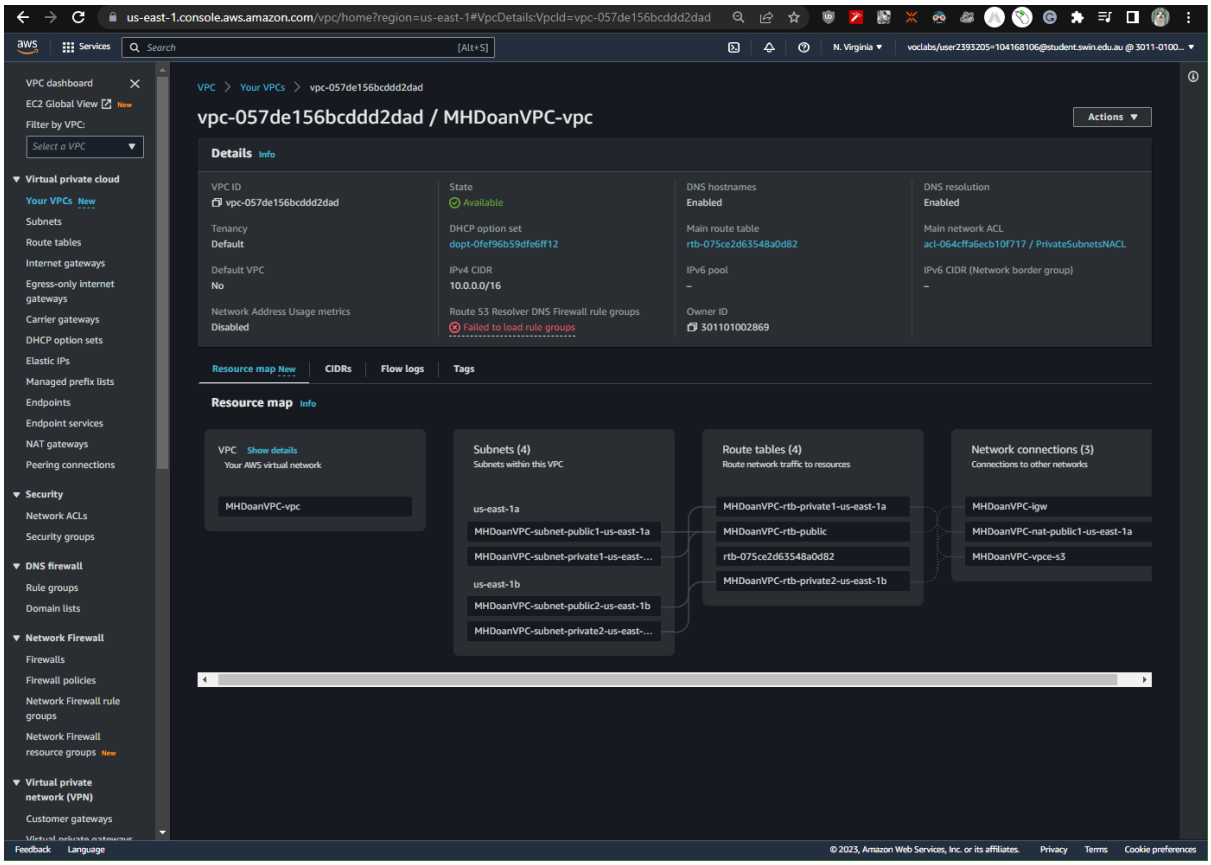

Figure 1 – MHDOANVPC-vpc

III. CREATING SECURITY GROUPS AND CONFIGURING:

After that, I needed to set up five security groups, each of which corresponds to a tier in the architecture diagram in the PDF file. I also had to make sure that these security groups followed the least-privilege principle.

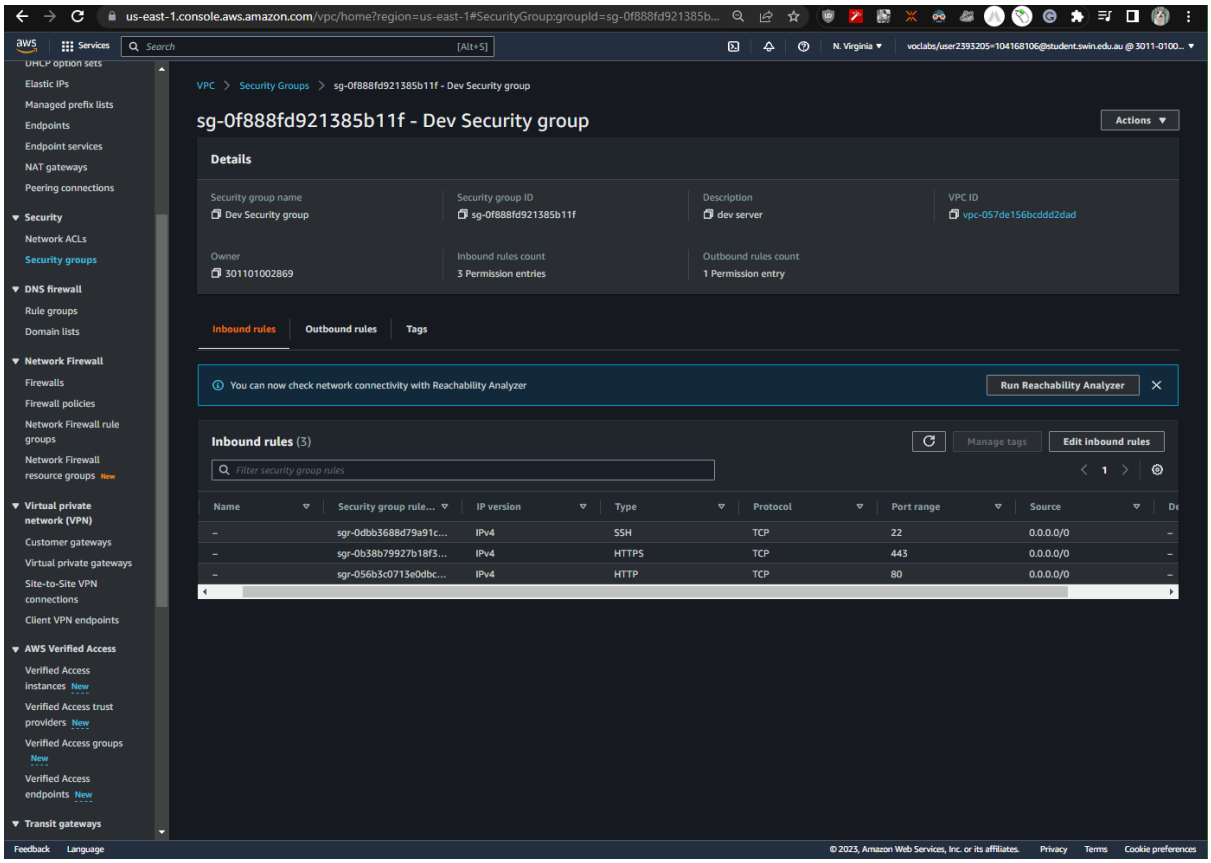

Figure 2 – Dev Security group

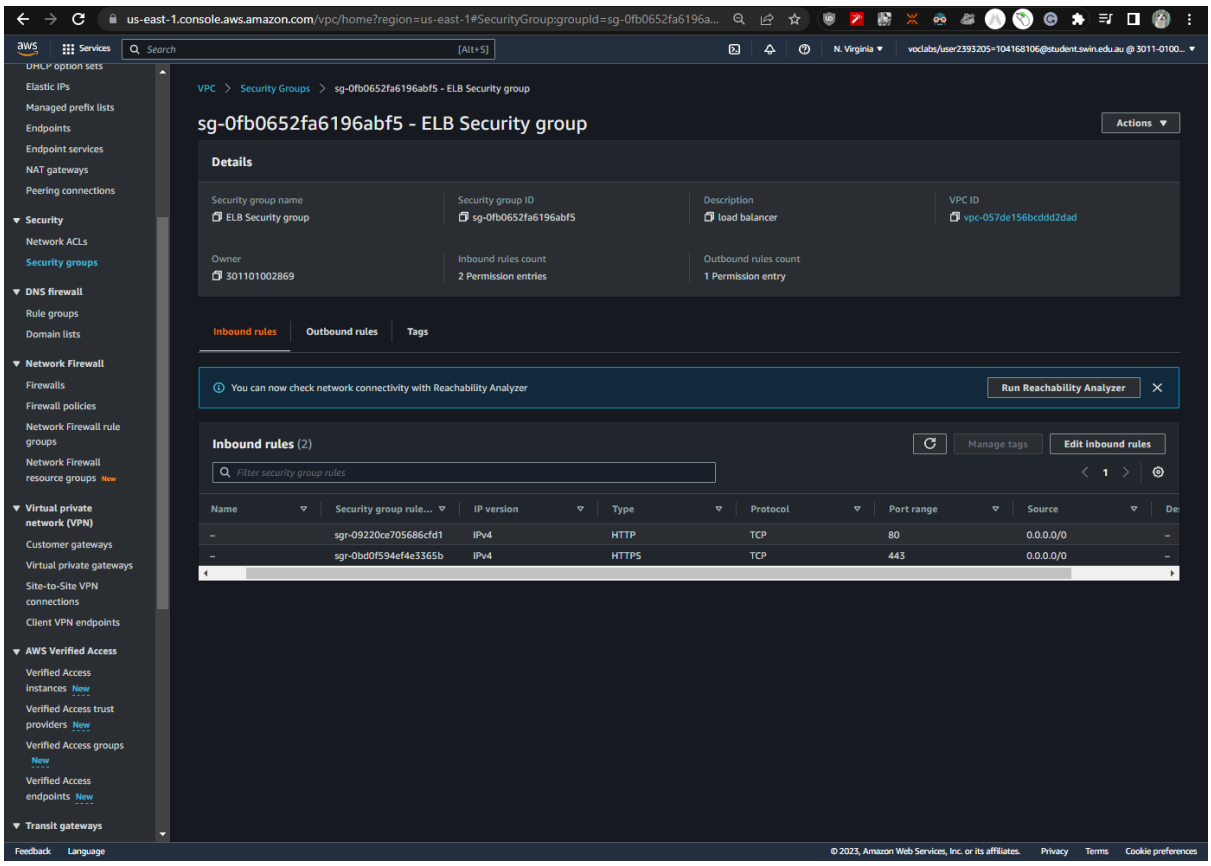

Figure 3 – ELB Security group

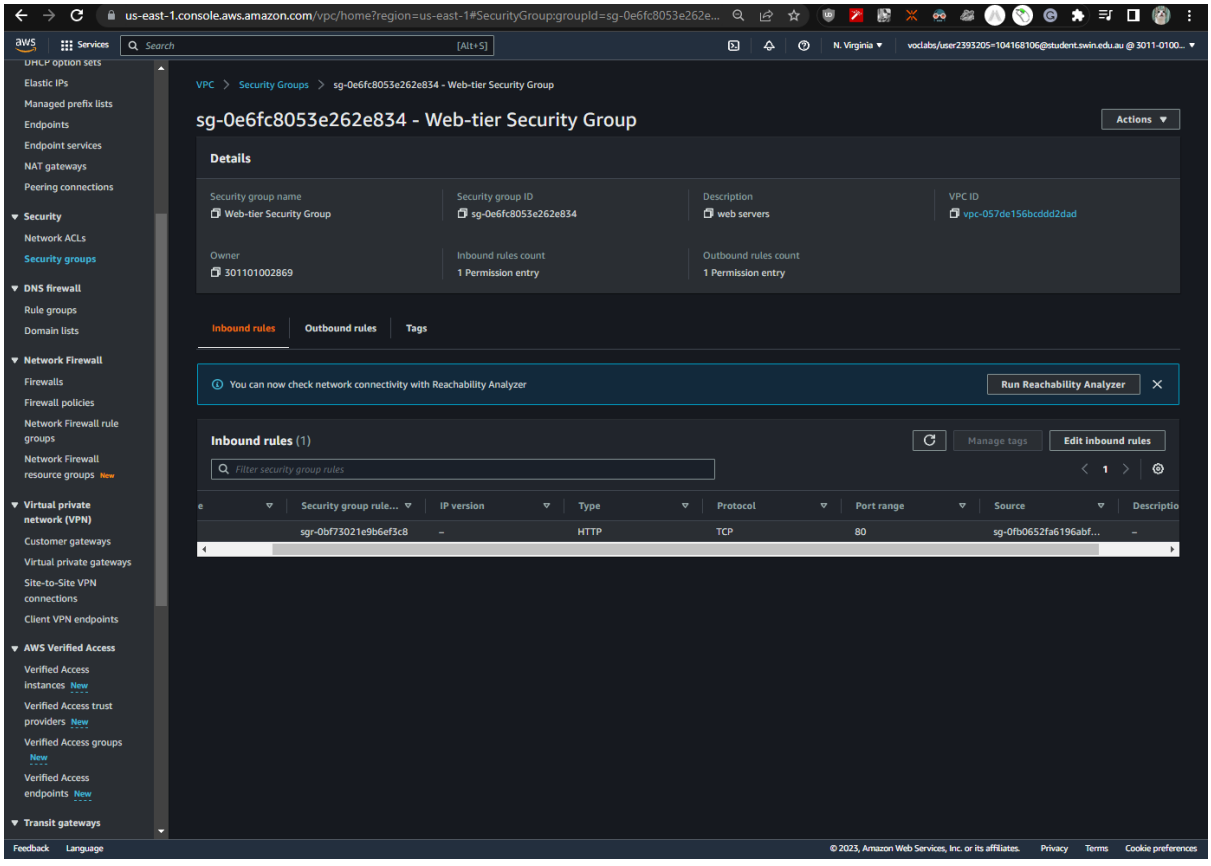

Figure 4 – Web-tier Security group

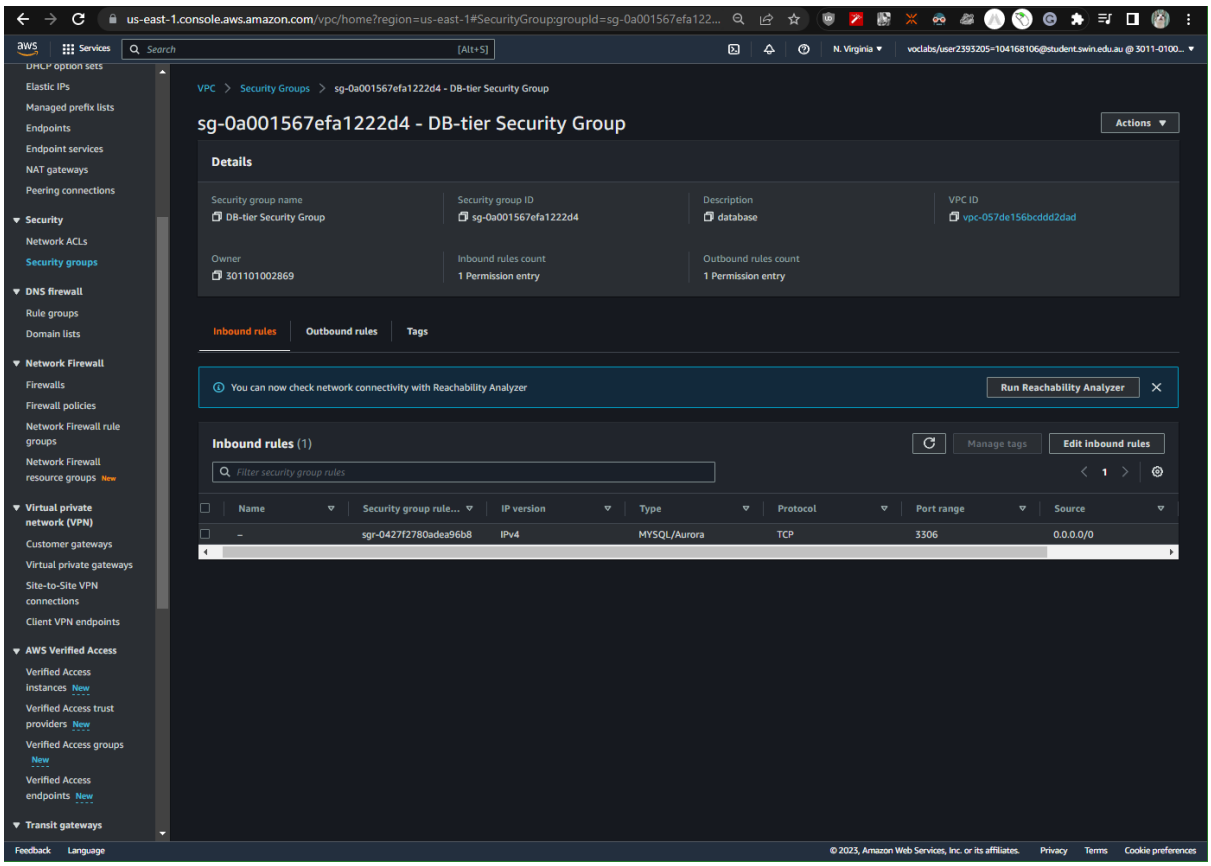

Figure 5 – DB-tier Security group

IV. CONFIGURING A NETWORK ACL (NACL):

The next step is to implement a Network ACL that restricts ICMP traffic to the appropriate subnets in order to provide my web servers with an additional layer of security.

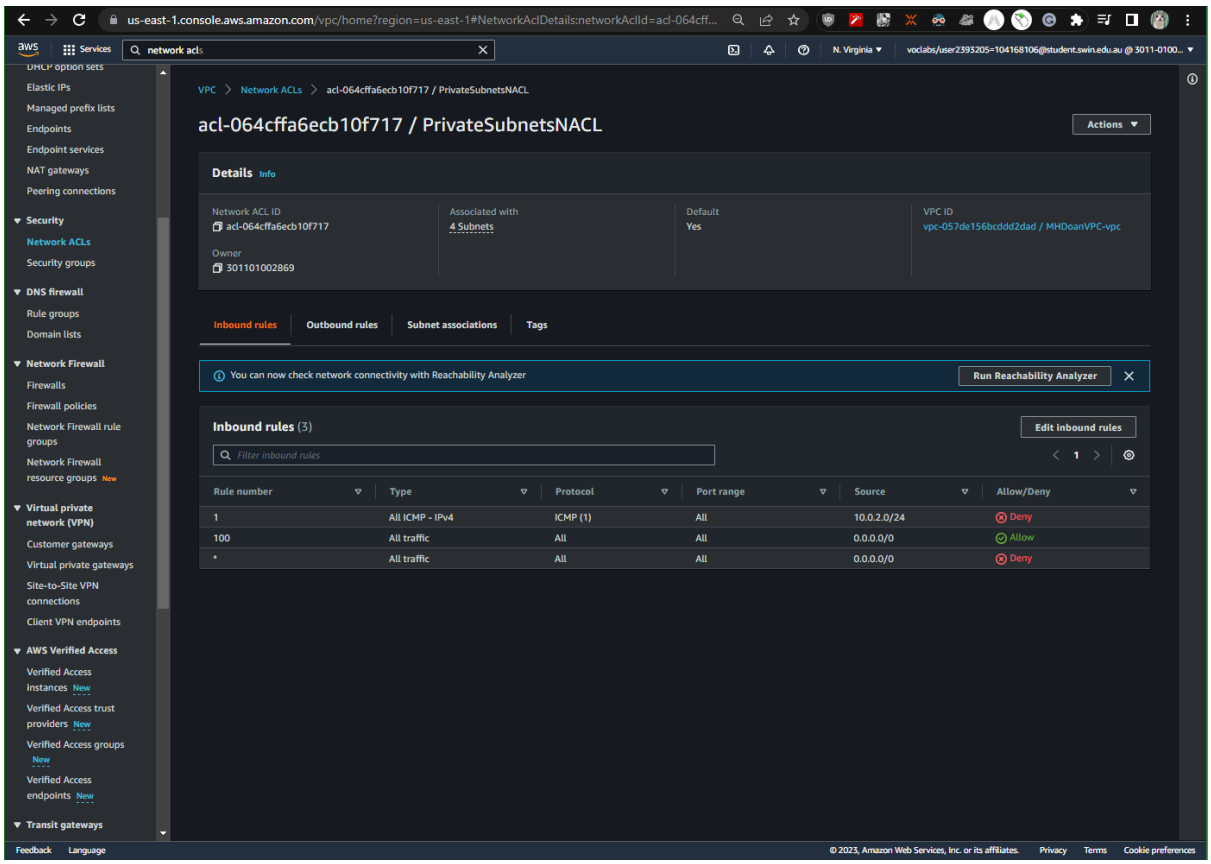

Figure 6 – PrivateSubnetsNACL

V. CONFIGURING IAM ROLES:

Because of AWS academy restrictions, I was not able to create my own IAM roles in AWS Learner Lab environment. Instead, I used the existed IAM roles “LabRole” with required permissions.

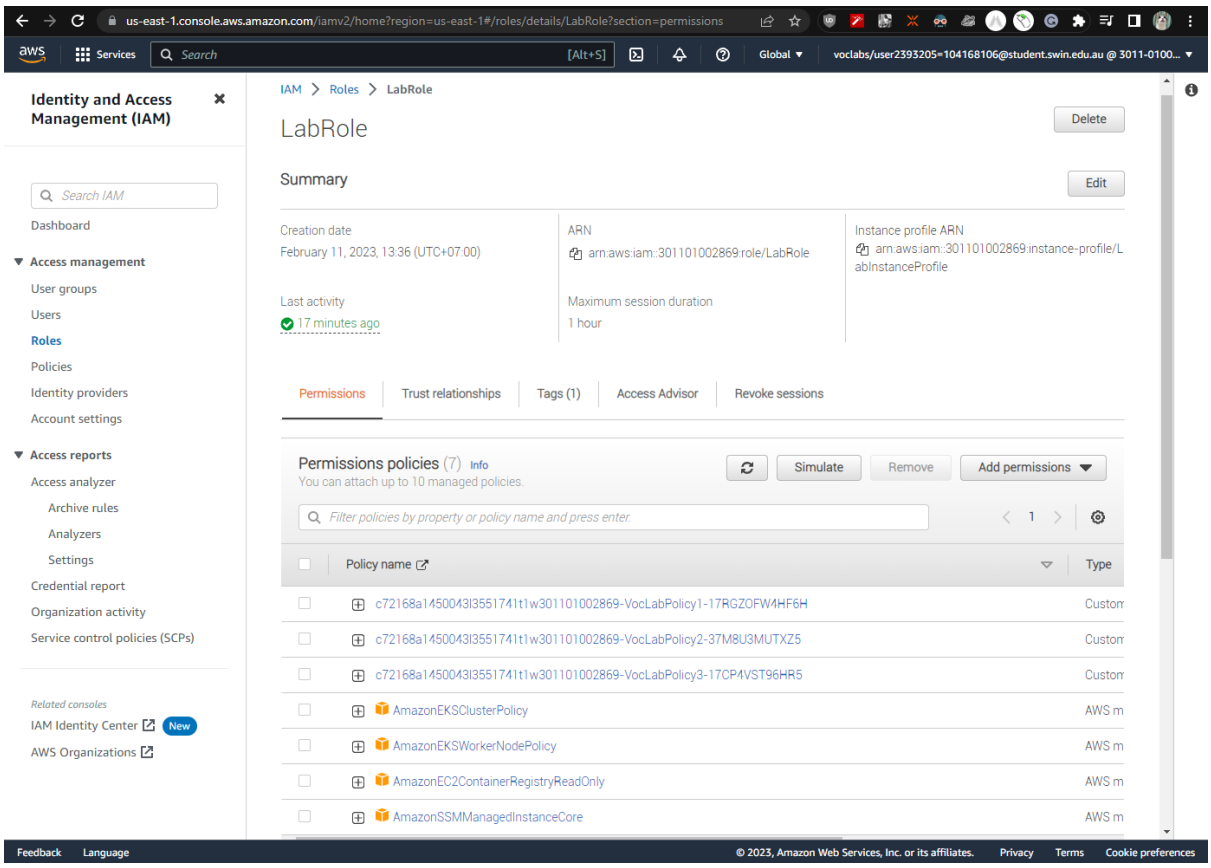

Figure 7 – LabRole

VI. CONFIGURING AN AUTO-SCALING GROUP (ASG):

According to the requirements, I needed to ensure that my auto-scaling group is scaled dynamically based on the traffic to maintain a certain number of requests per target in the ELB target group. I also need to ensure that instances are launched in private subnets.

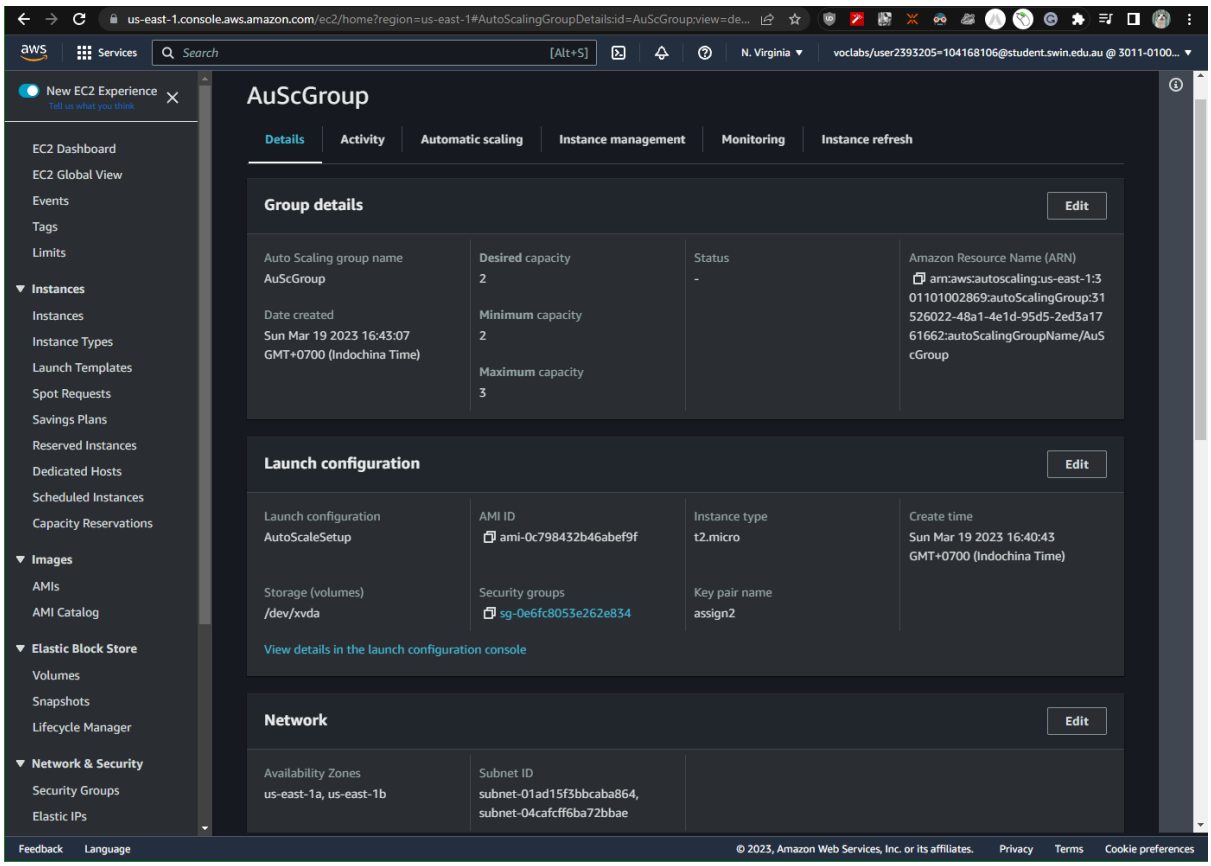

Figure 8 – My auto-scaling group

VII. CREATING AN APPLICATION LOAD BALANCER (ELB):

Before proceeding to the next step, I created an application load balancer, which must be used to distribute the load of web requests among the web servers in the auto-scaling group. Another point to ensure is that all instances must have health checks conducted by my ELB.

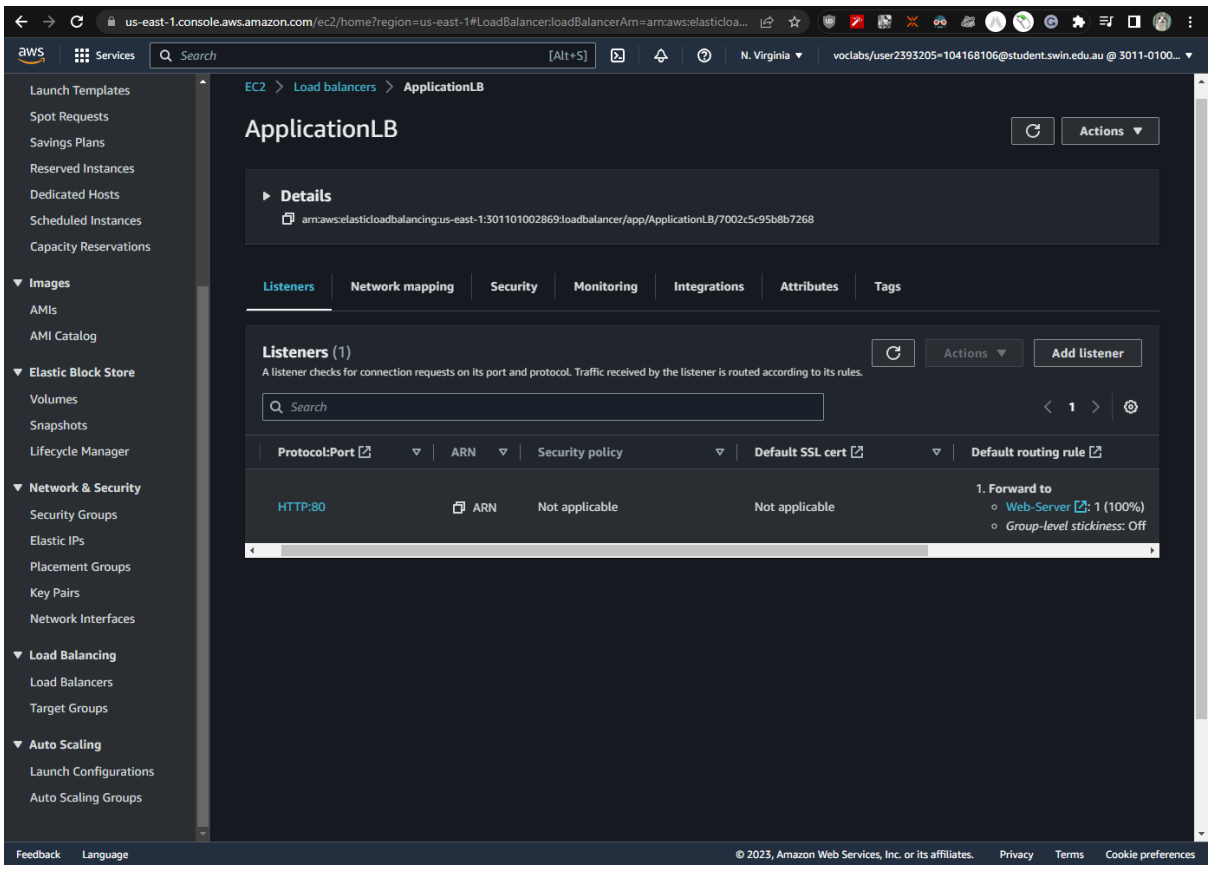

Figure 9 – My application load balancer

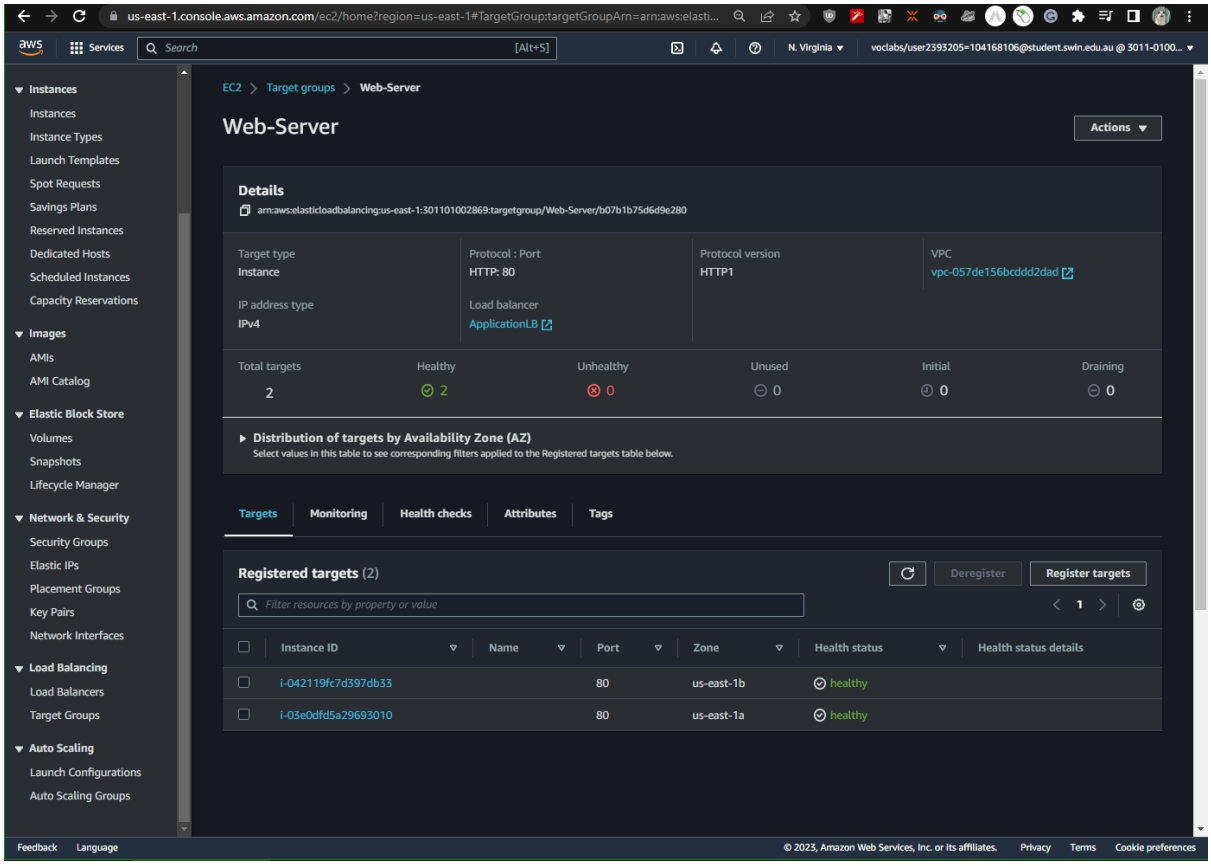

Figure 10 – Instances launched by the auto scaling group

VIII. CREATING AN ELASTIC COMPUTE CLOUD (EC2) INSTANCE:

The ELB does not send any traffic to the Dev server. The PhotoAlbum website can be operated on this server. Moreover, my database management can also be done using it.

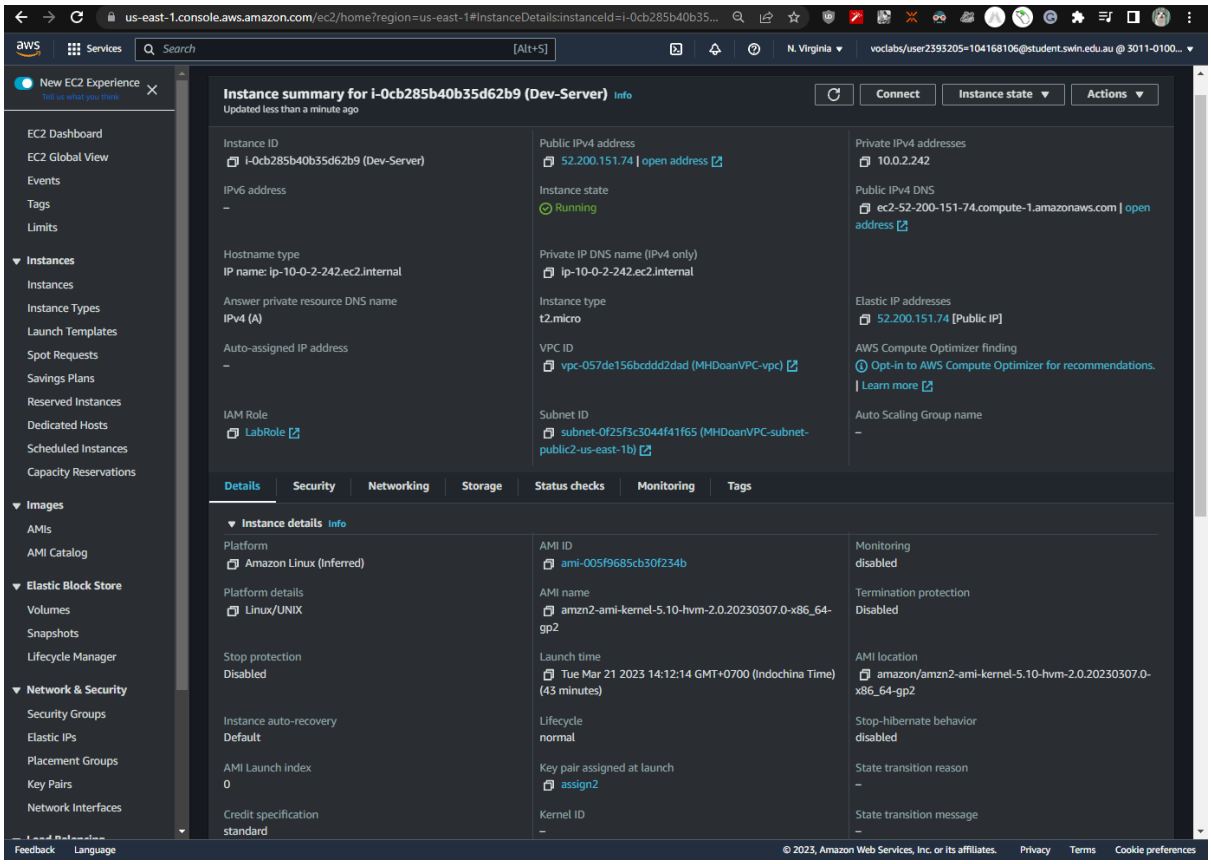

Figure 11 – Dev-Server with IAM role

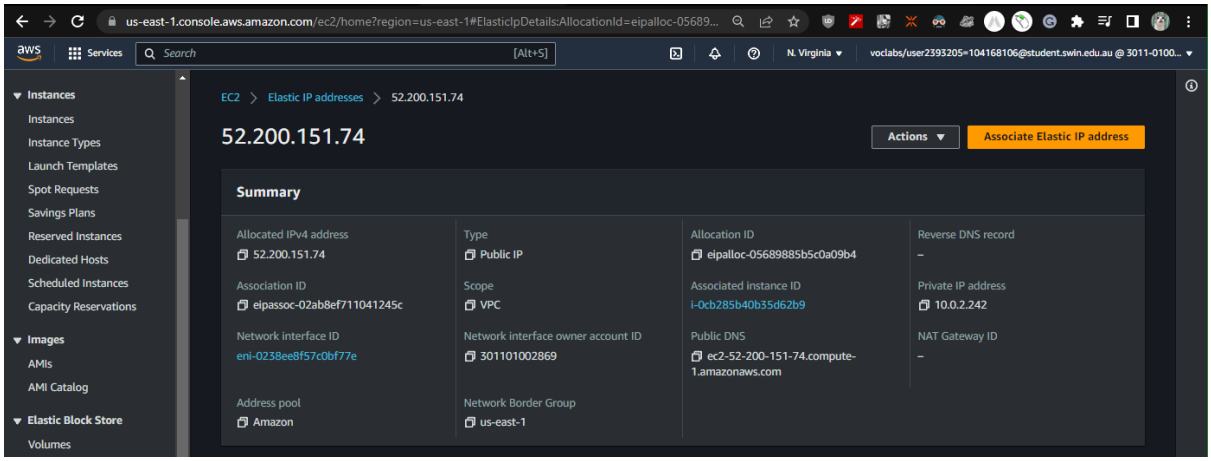

Figure 12 – Elastic IP for Dev-Server

IX. ACCESSING TO PHOTOS IN S3 BUCKET AND CONFIGURING ITS POLICY:

By setting up a S3 bucket policy that restricts access to a specific HTTP referrer, I made sure the objects kept in this S3 bucket are correctly accessible.

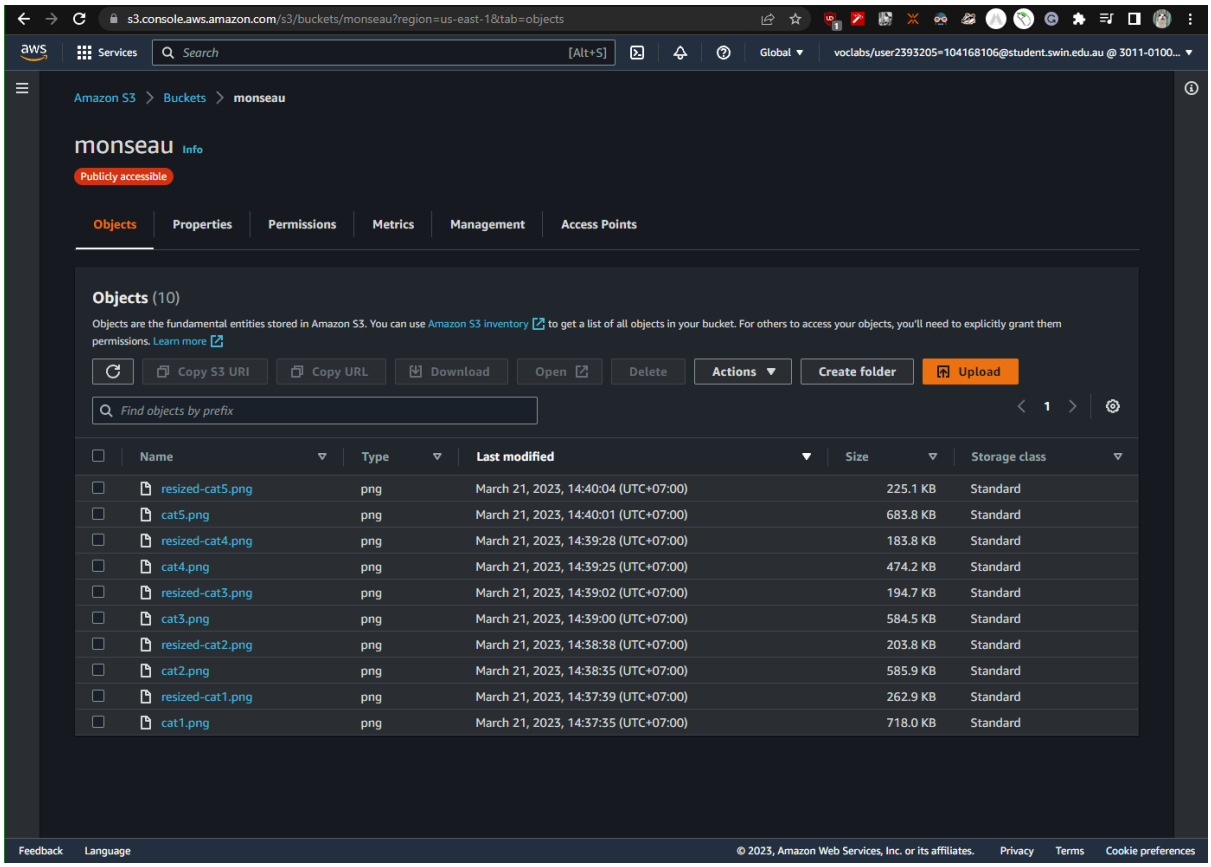

Figure 13 – my S3 bucket named monseau

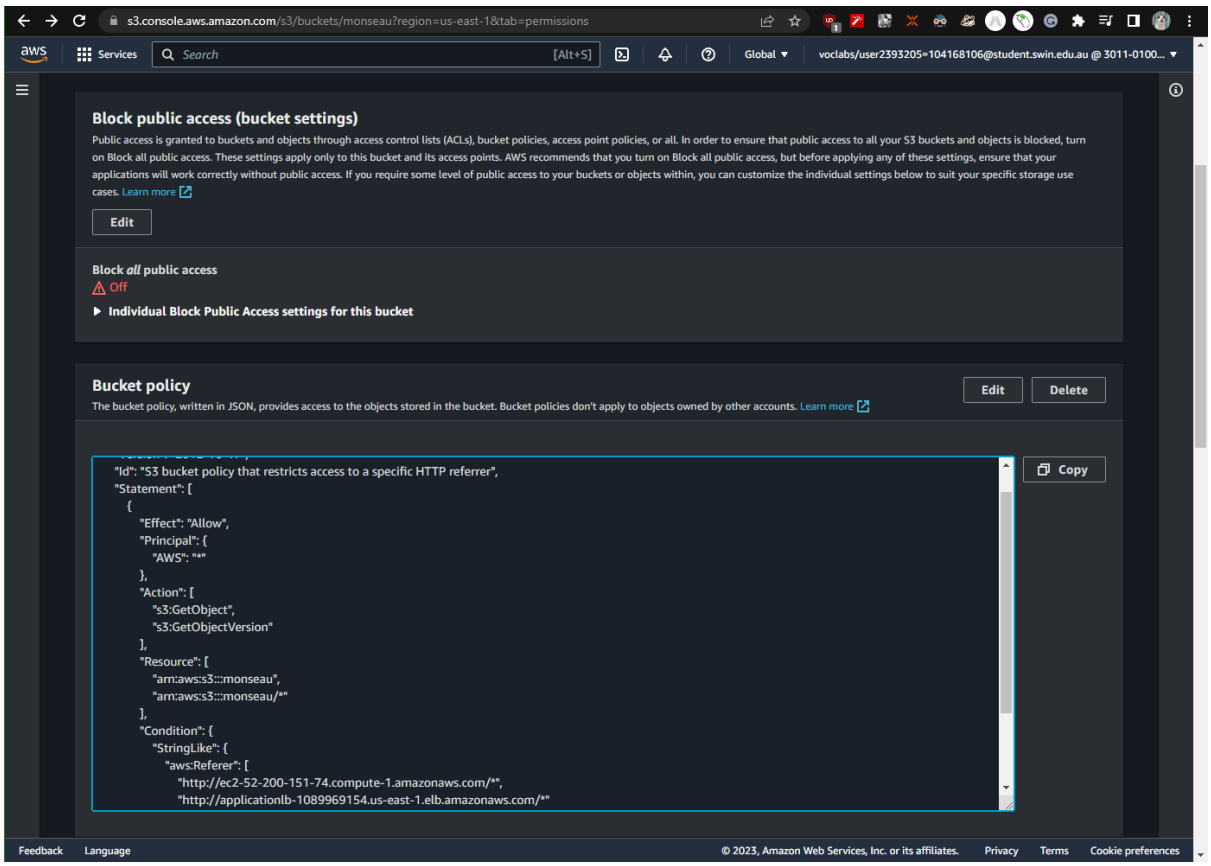

Figure 14 – S3 bucket policy

Figure 15 – Direct access to S3 photos

X. CREATING A LAMBDA FUNCTION:

To continue, I uploaded a provided deployment package to add functionality to the Lambda function that I had created. The package doesn't need to be modified in order to function.

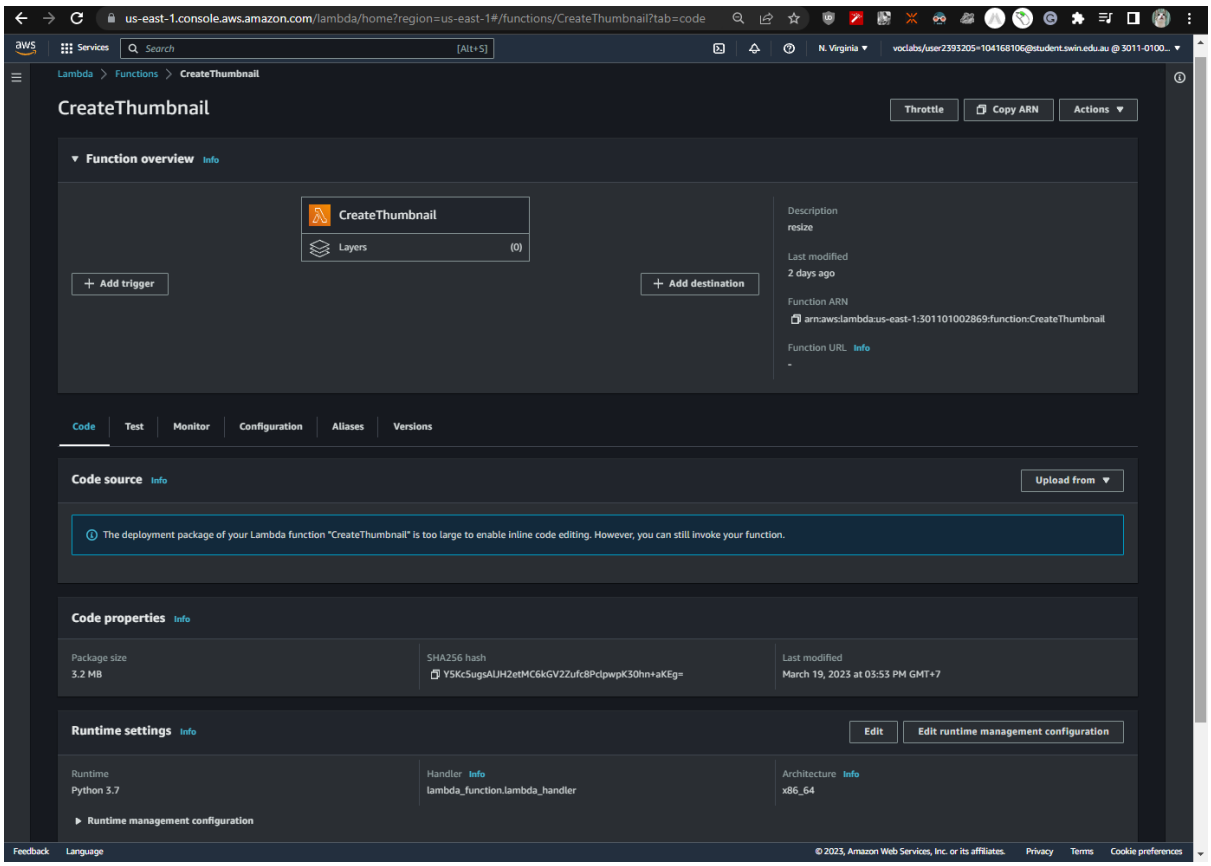

Figure 16 – Lambda function

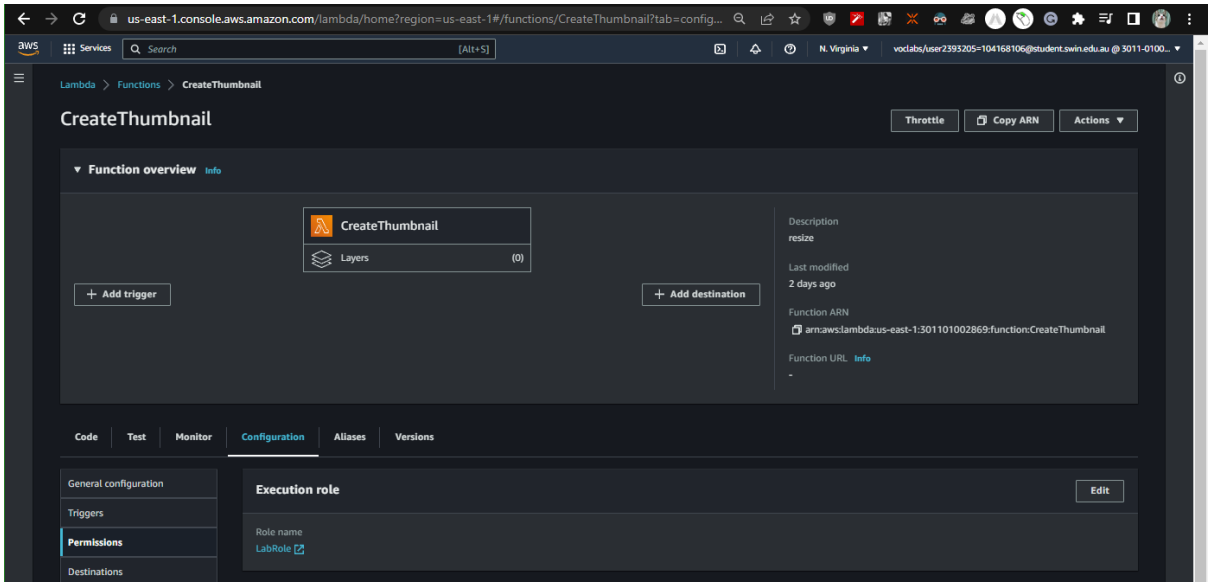

Figure 17 – Lambda function's IAM role

XI. CONFIGURING A RELATIONAL DATABASE SERVICE (RDS) DB:

Once I had completed the previous step, I could move on to create and manage my database through the Dev server by accessing to phpMyAdmin.

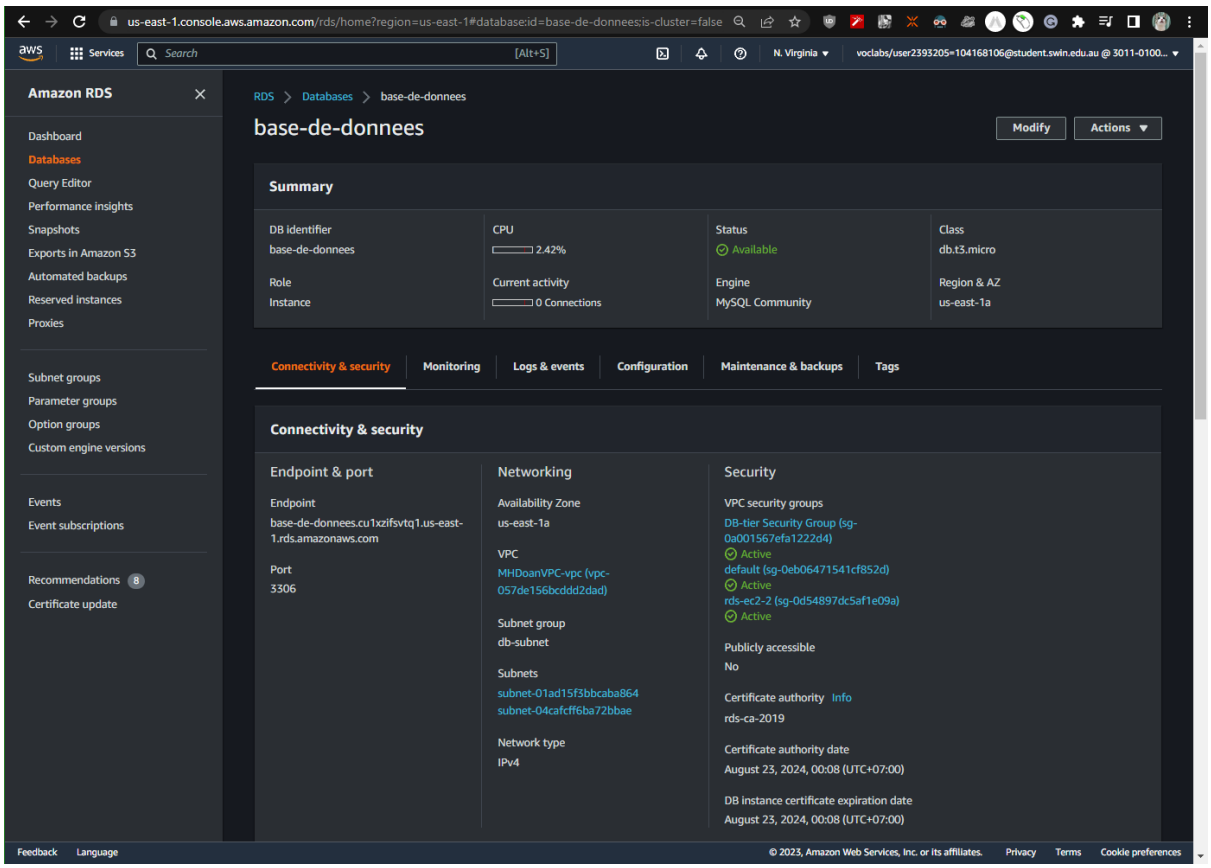

Figure 18 – My database

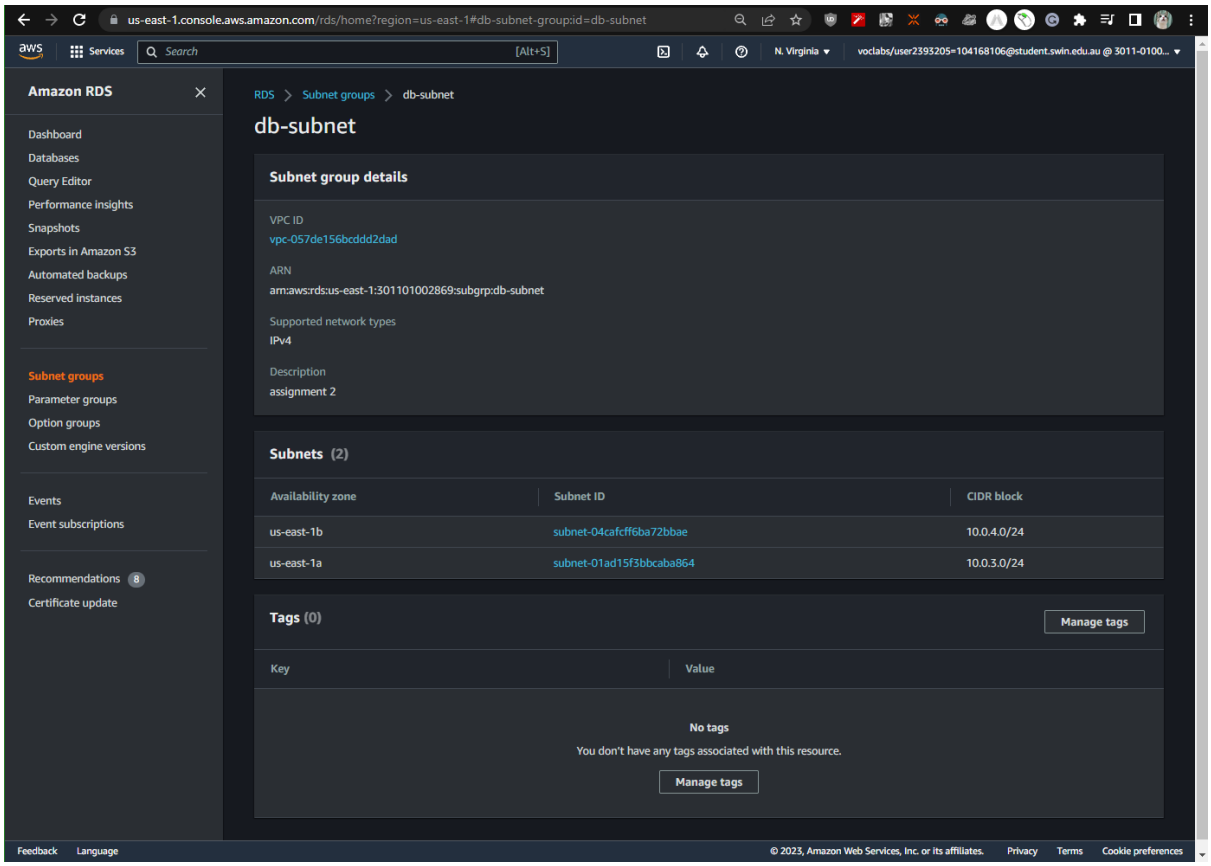

Figure 19 – My database's subnet group

XII. WEBSITE ACCESSIBLE VIA ELB:

http://applicationlb-1089969154.us-east-1.elb.amazonaws.com/photoalbum/album.php

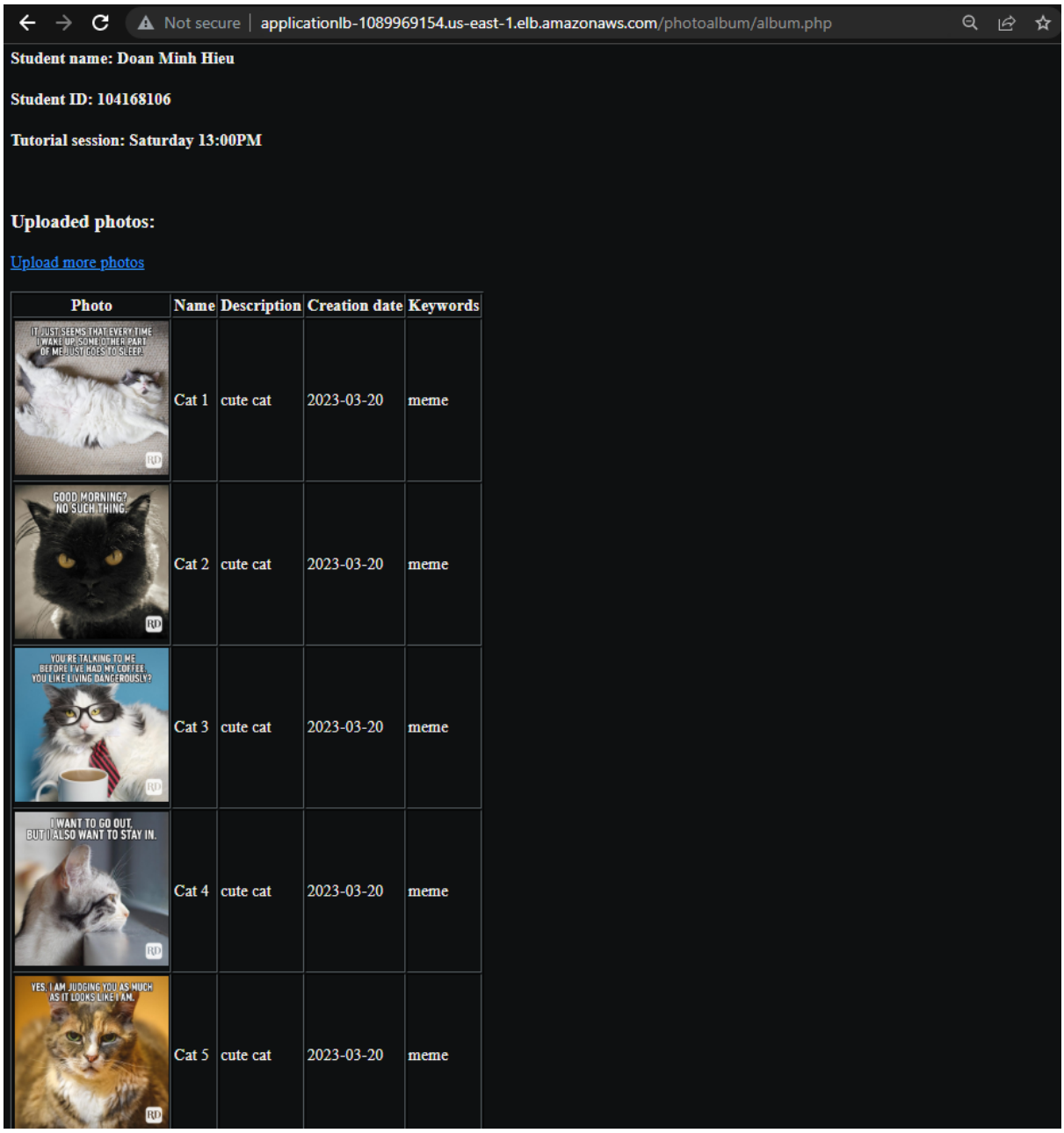

Figure 20 – Website on ELB

XIII. PHOTOS AND THEIR META-DATA DISPLAYED ON ALBUM.PHP PAGE:

http://ec2-52-200-151-74.compute-1.amazonaws.com/photoalbum/album.php

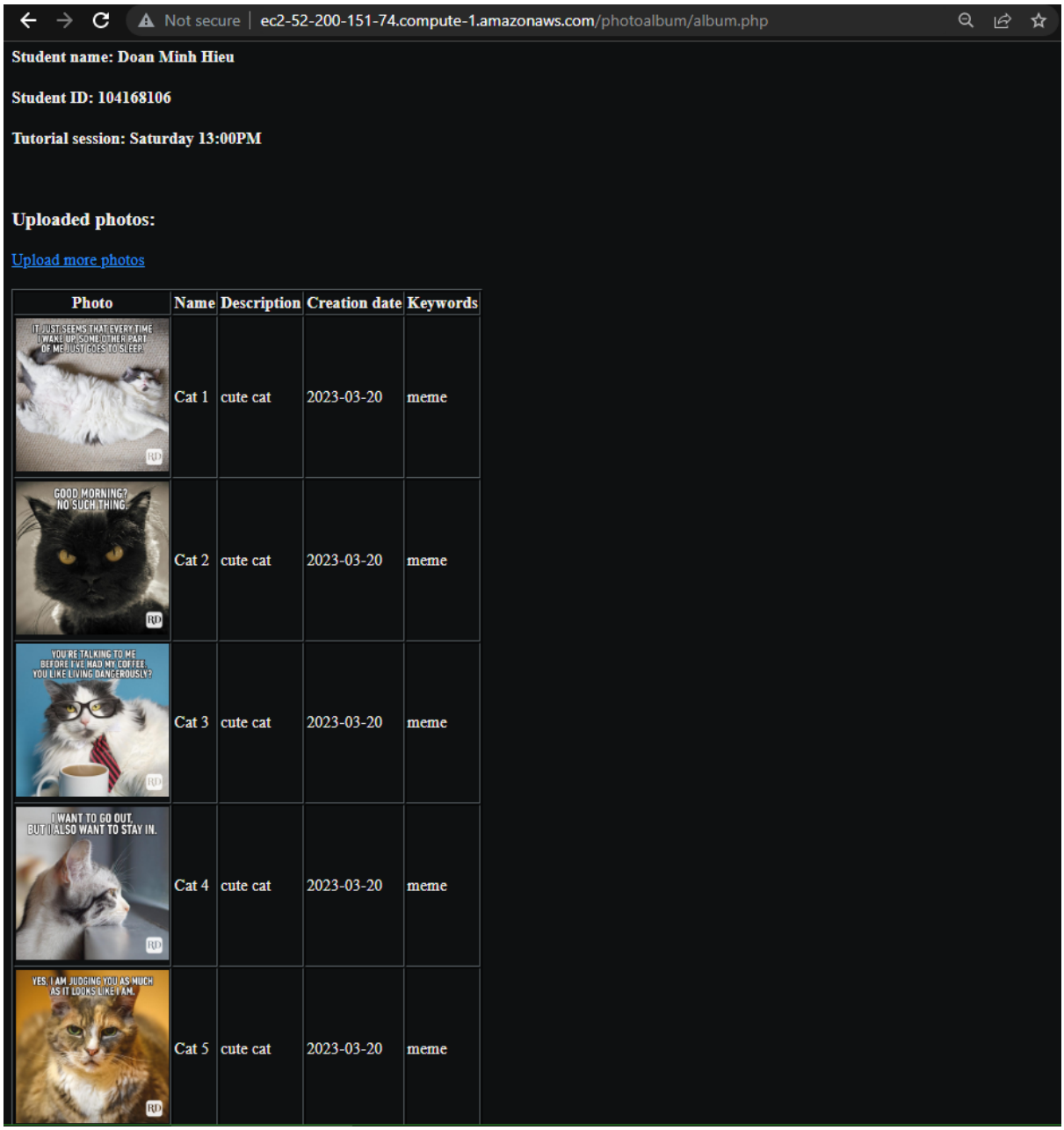

Figure 21 – Website on Dev-Server

XIV. PHOTOS AND THEIR META-DATA ON THE RDS DATABASE:

Finally, I could check whether the photos and their meta-data were correctly uploaded through album.php by checking the RDS database and also the S3 bucket (Fig.13).

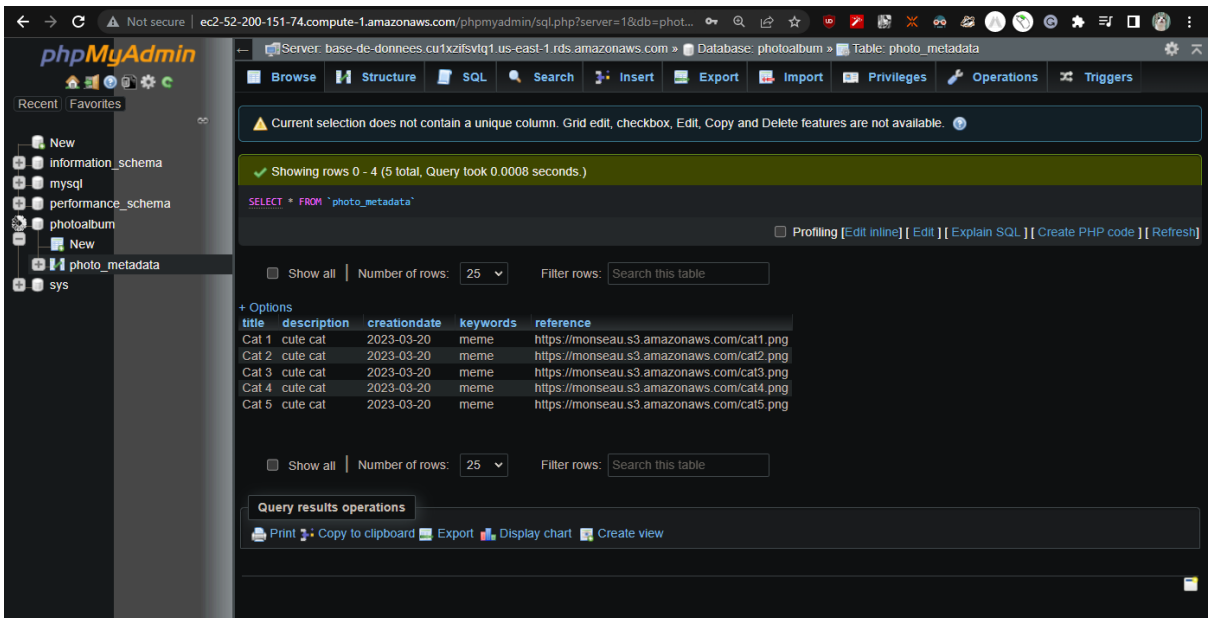

Figure 22 – Photos and their meta-data on the RDS database

XV. CHECKLIST:

COS80001 CCA: Assignment 2 Checklist

Make sure all the following are completed.

Submission Checklist

Student Name: ……Doan Minh Hieu……

Student Id: ……104168106……

Tutorial time: ……Saturday 13:00PM……

Date of submission: ...21/03/2023……

Submit to Canvas:

☒ A PDF document file as specified in the Submission section of the assignment specification.

Marking Scheme

Infrastructure Requirements (10 marks)		
VPC configured with 2AZs both with public and private subnets. Public and private route tables route to IGW and NAT, respectively.	0.5	Fig.1
Security groups created and properly configured.	2	Fig.2, 3, 4, 5
NACL correctly configured.	1	Fig.6
IAM roles properly configured	1.5	Fig.7, 11, 17
ASG configured and working correctly.	1	Fig.8
ELB configured and working correctly with associated Elastic Public IP address.	1	Fig.9, 10
Photos stored in S3 are correctly accessible. S3 bucket policy is correct.	1.5	Fig.13, 14, 15
Lambda configured and working correctly.	1	Fig.16
RDS configured and working correctly.	0.5	Fig.18, 19, 22
Functional Requirements (5 marks)		
Website accessible via ELB.	0.5	Fig.20
Photos and their meta-data displayed on album.php page	1.5	Fig.21
Photos and their meta-data can be uploaded to the S3 bucket and RDS database, respectively.	1.5	Fig.13, 22
Photos are resized by the Lambda function.	1.5	Fig.13
Deductions		
Documentation not as specified or poorly presented (up to minus 15)		
Serious misconfigurations of AWS services being used (up to minus 15)		

Comments