Assignment 1 - Part B

Creating and deploying Photo Album website onto a simple AWS infrastructure

Student name: Hai Nam Ngo

Student ID: 103488515

Tutorial session: Friday 04:30PM

URL of Album Page: http://44.194.3.227/cos20019/photoalbum/album.php

I. Introduction

For the assignment, I will be focusing on a set of key objectives using VPC, RDS, S3 and EC2. Firstly, I will create a secure Virtual Private Cloud (VPC) with subnets, routing tables, and security groups. Following that, I will ensure controlled access to and from the VPC via an Internet Gateway. Additionally, I will modify the provided PHP code to create a website that stores metadata information about photos uploaded to Amazon S3 in a MySQL database managed by Amazon RDS. This website will offer users the capability to search for and display photos using metadata. Subsequently, I will deploy and test the PHP website on an Apache web server running on an EC2 virtual machine instance. Lastly, I will add an extra layer of security by implementing a Network ACL to the public subnet responsible for hosting the web server.

II. Infrastructure deployment

The main target is to create an infrastructure looks like this diagram, with web instance attached to public subnet 2, test instance in private subnet 2, RDS instance in private subnet 1, all of them will be attached with the suitable Security Groups, connect with Internet Gateway through web instance. S3 bucket will also be connected to that IGW.

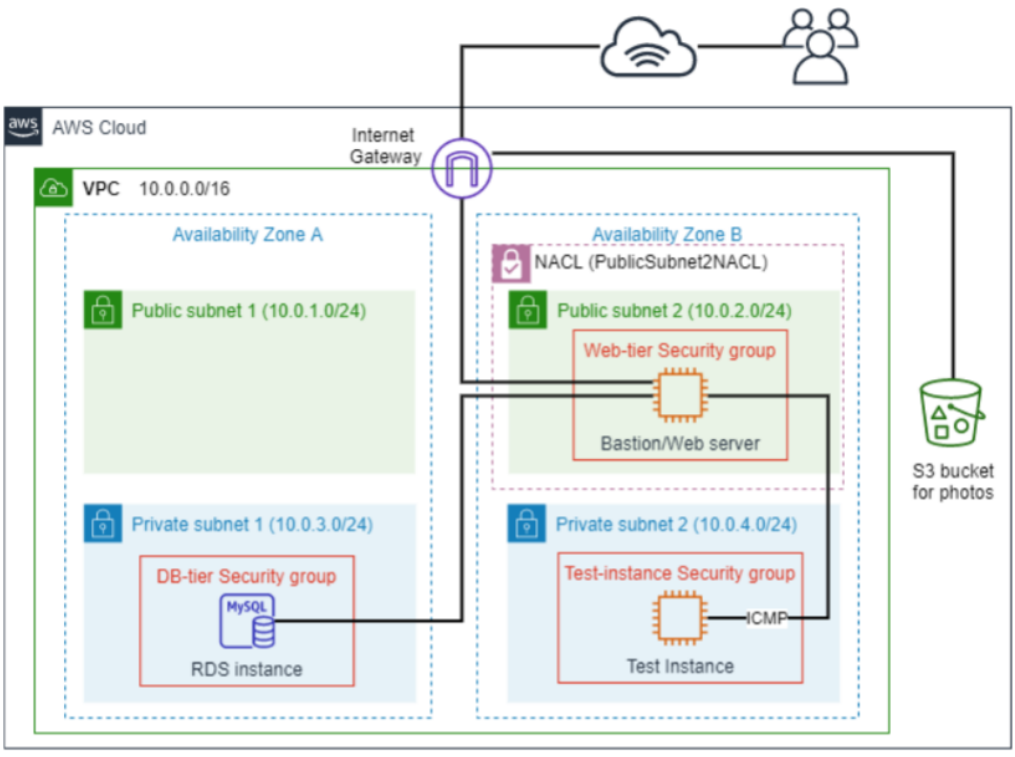

Figure 1 The structure and services in this assignment

1.1 – VPC

This step is creating only VPC named “HNgoVPC” based on my name, with the right IPv4 CIDR and keep all other default setting.

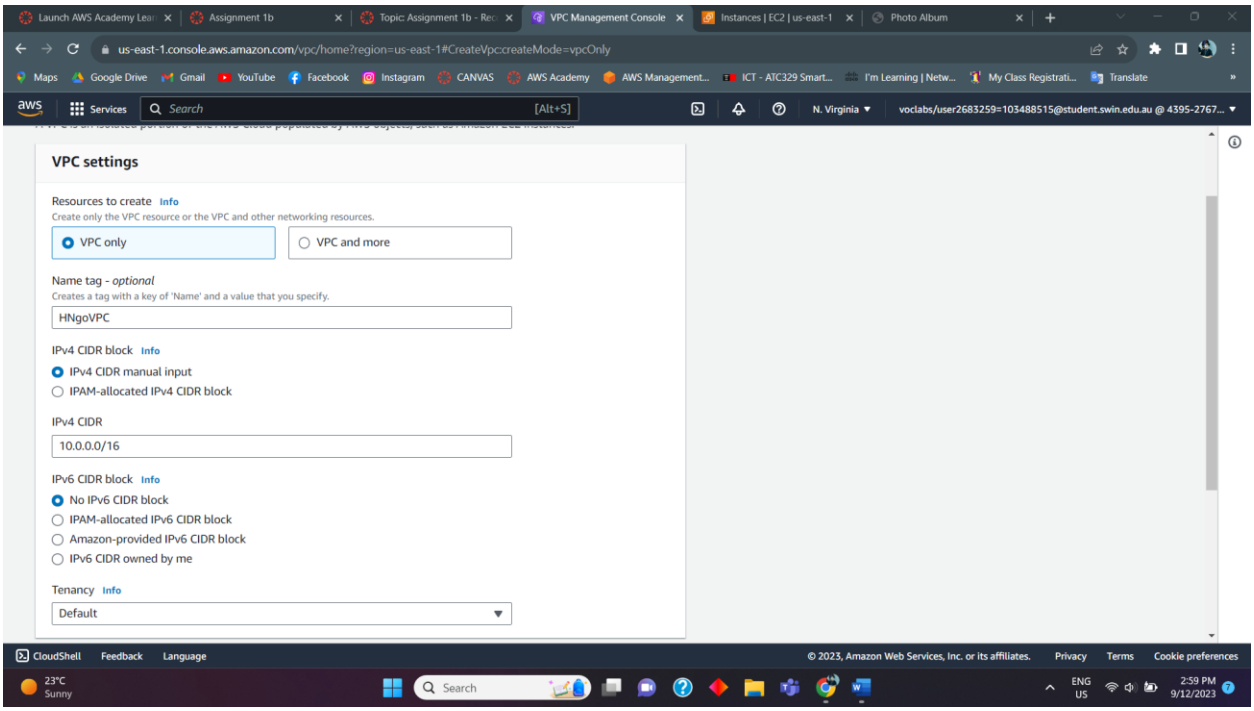

Figure 2 Create VPC (choose "VPC Only" option)

Next step is creating an internet gateway named “HNgo_IGW” and get ready to attach it with the VPC above.

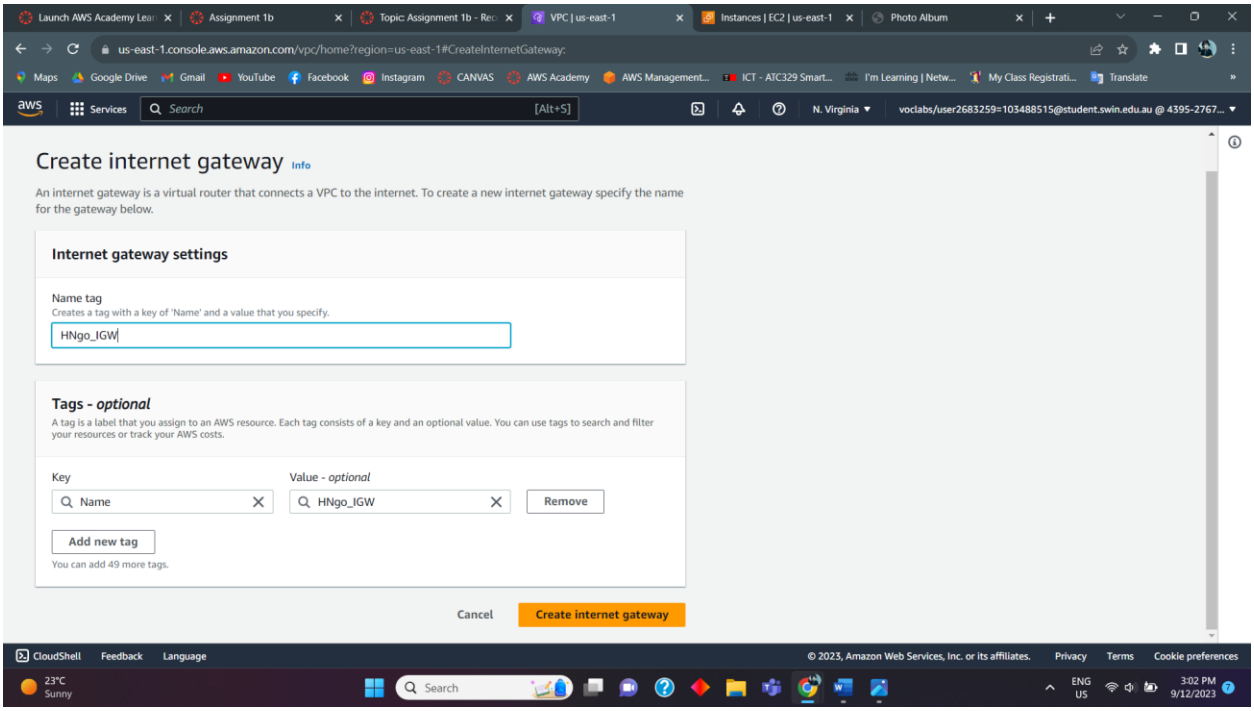

Figure 3 Create Internet gateway in VPC section.

After it is created, click on the “Action” button, and choose to attach, down here is the result. In the VPC ID section, the IGW is already attached to the HNgoVPC.

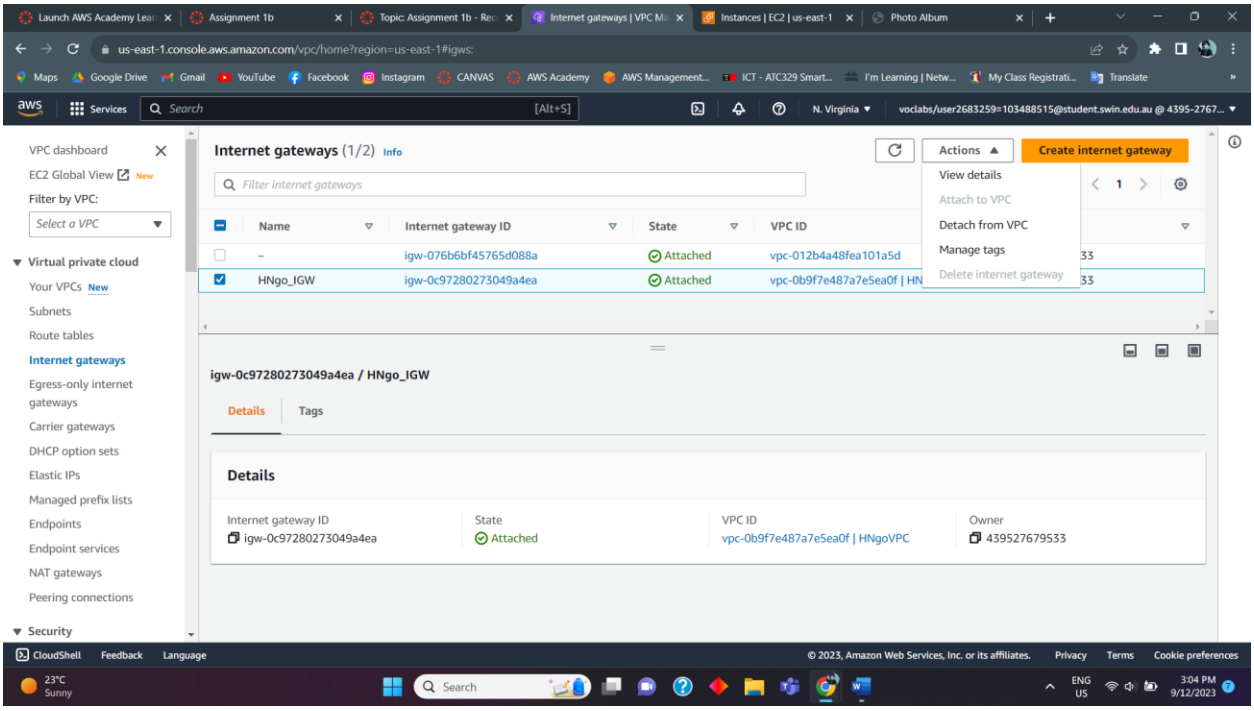

Figure 4 Attach IGW to HNgoVPC

Next step is Creating Route Tables, in this case, the route table will be attached to the HNgoVPC, and add the internet route in Public Route Table, so after this step, there will be 2 routes available in the route section, one is for the internet, and one for local.

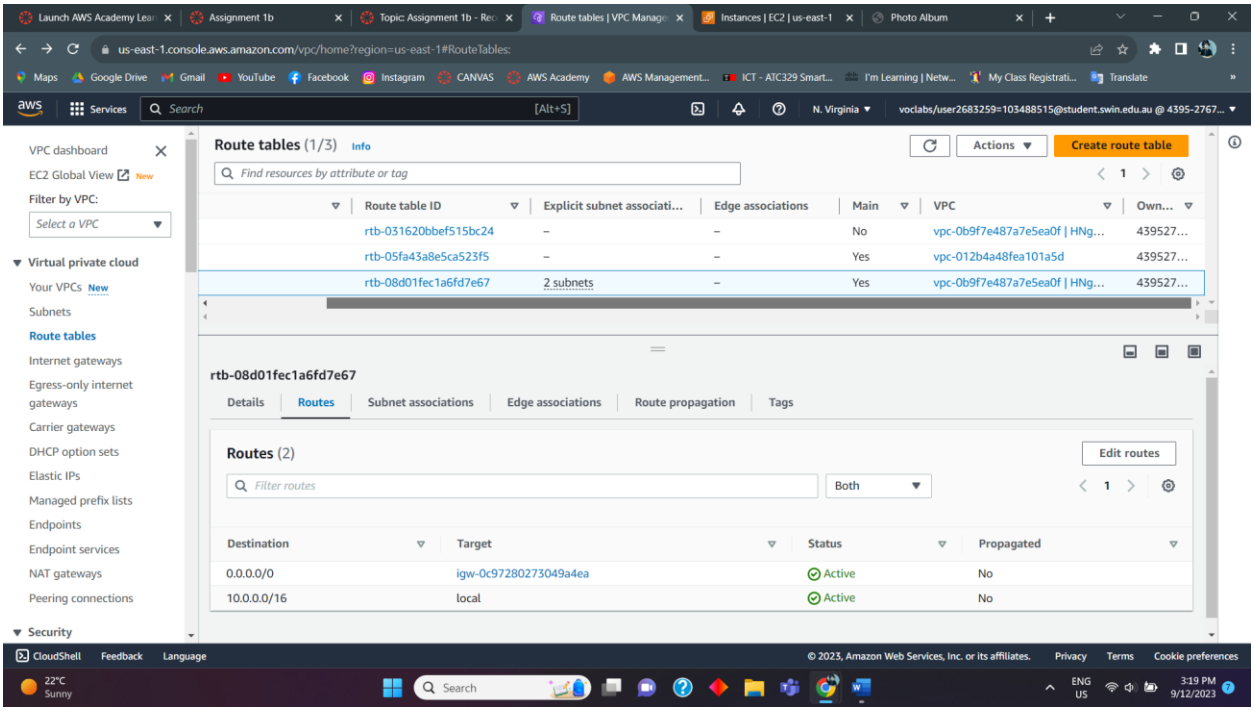

Figure 5 Connect route table with internet route.

Next step is Creating Subnets in right AZs and associate the right Route Table to them. For the assignment, there will be 4 subnets with 2 Availability Zones, they are created correctly with the right IPv4 CIDR.

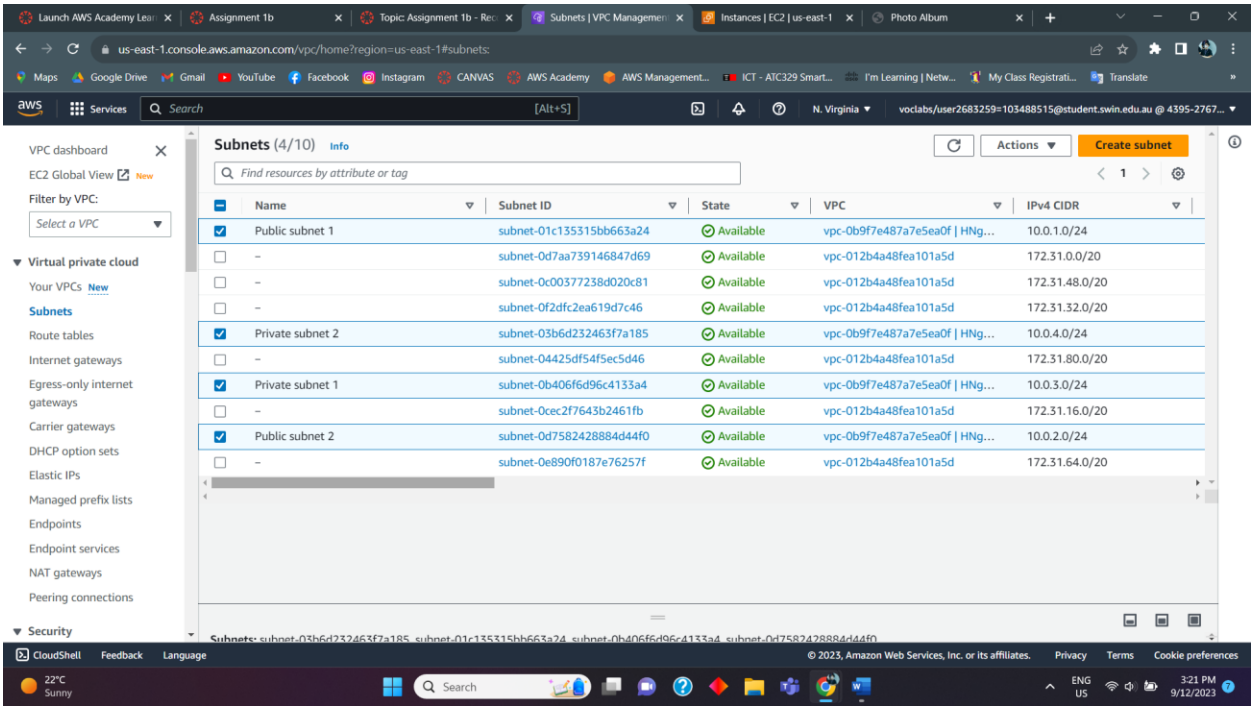

Figure 6 Successfully creating subnets.

Then, associate public subnets with a public route table that routes to the Internet Gateway.

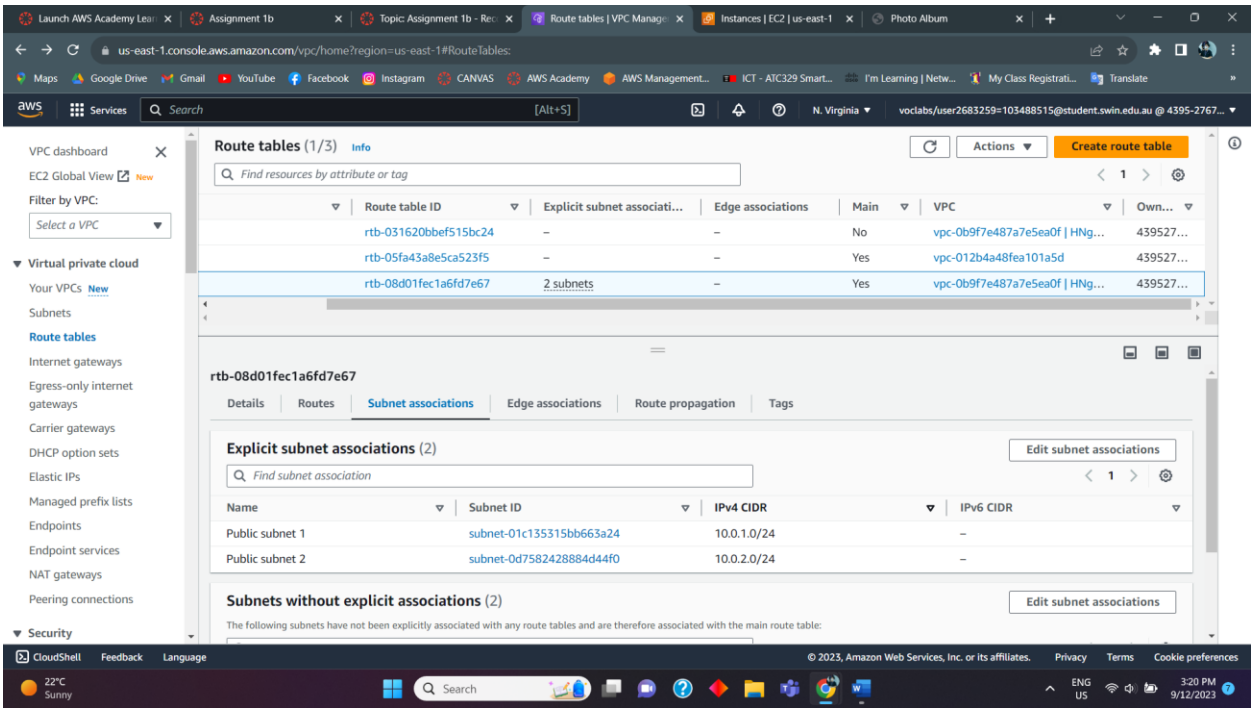

Figure 7 Make public subnets link with the public route table successfully.

1.2 Create Security Groups

Three security groups will be created and each of them will be attached to the correct EC2 and RDS.

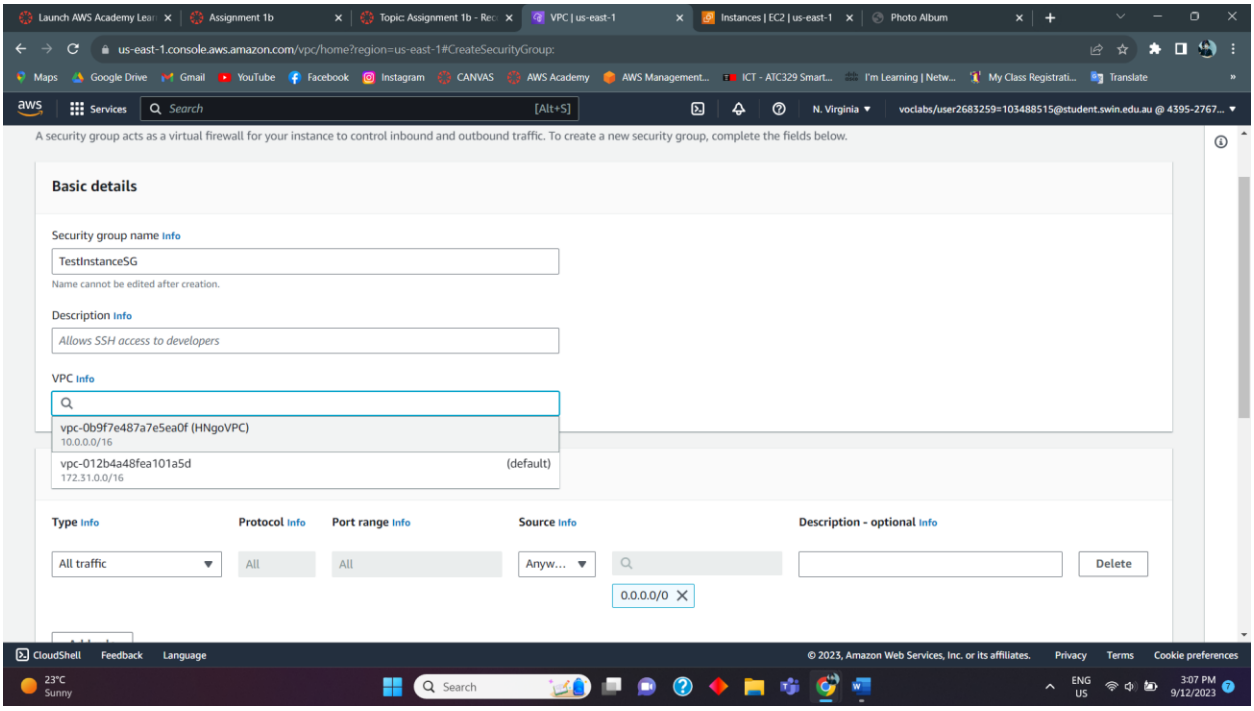

Figure 8 Creating Test Instance SG and include the inbound rule the same as the requirement.

After creating three SG, they will be displayed (highlight section) and ready to be attached.

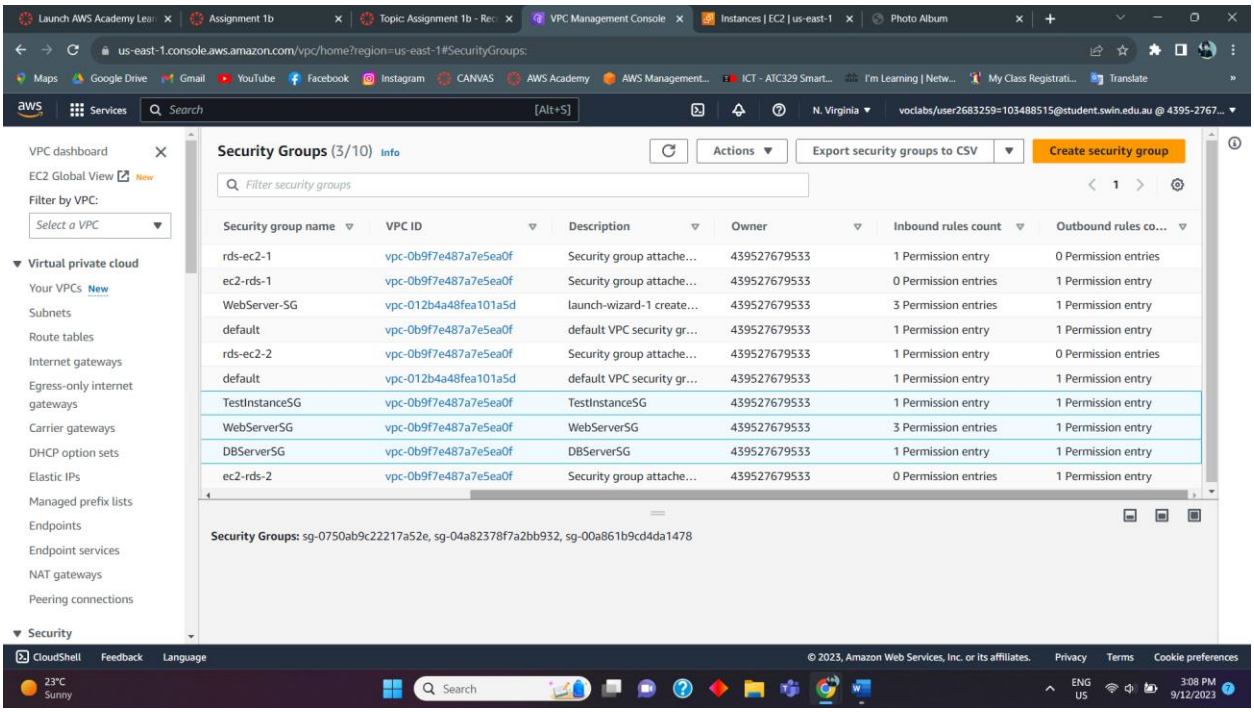

Figure 9 Successfully creating three SG.

1.3. EC2 virtual machine

Public DNS will change every time the webserver instance restarts. To avoid this behavior and to ensure the Webserver URL remains persistent, Elastic IP Address will be added to this instance by allocating an Elastic IP Address.

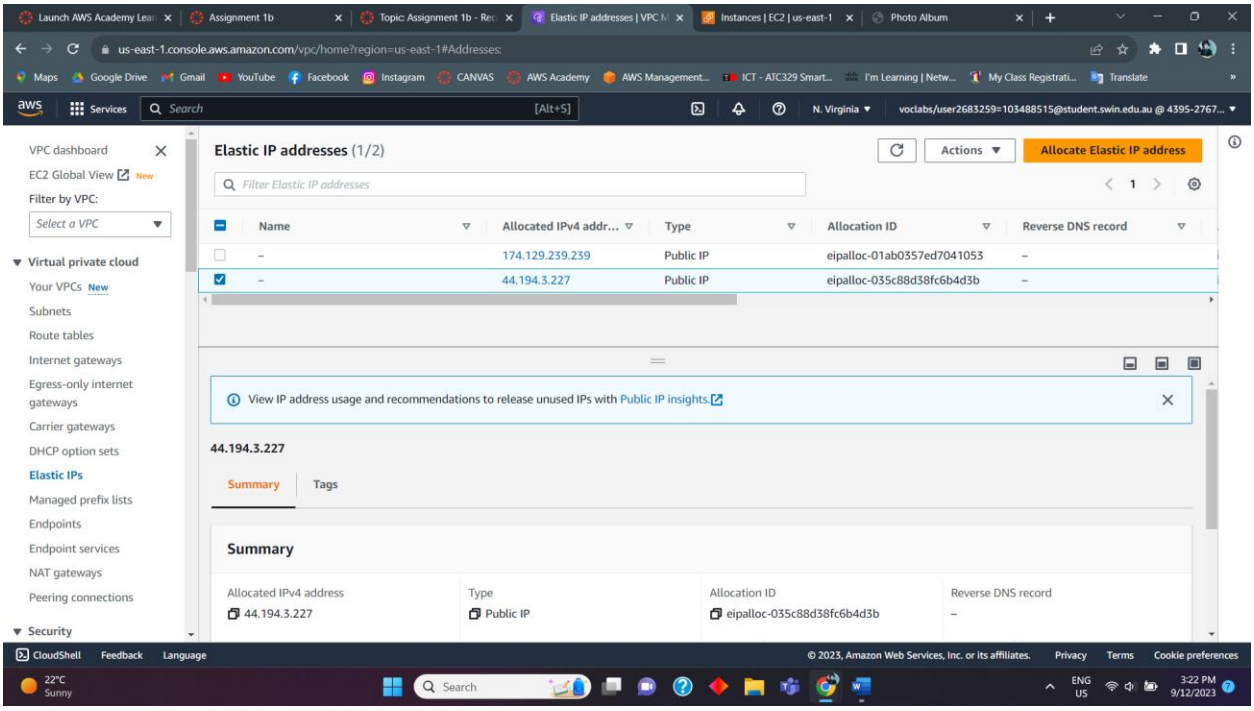

Figure 10 Create an Elastic Public IP address.

After that, Bastion/Web Server instance and test instance is created based on the requirement of the assignment, connect with the right security group, and Elastic IP Address is successfully associated with the instance.

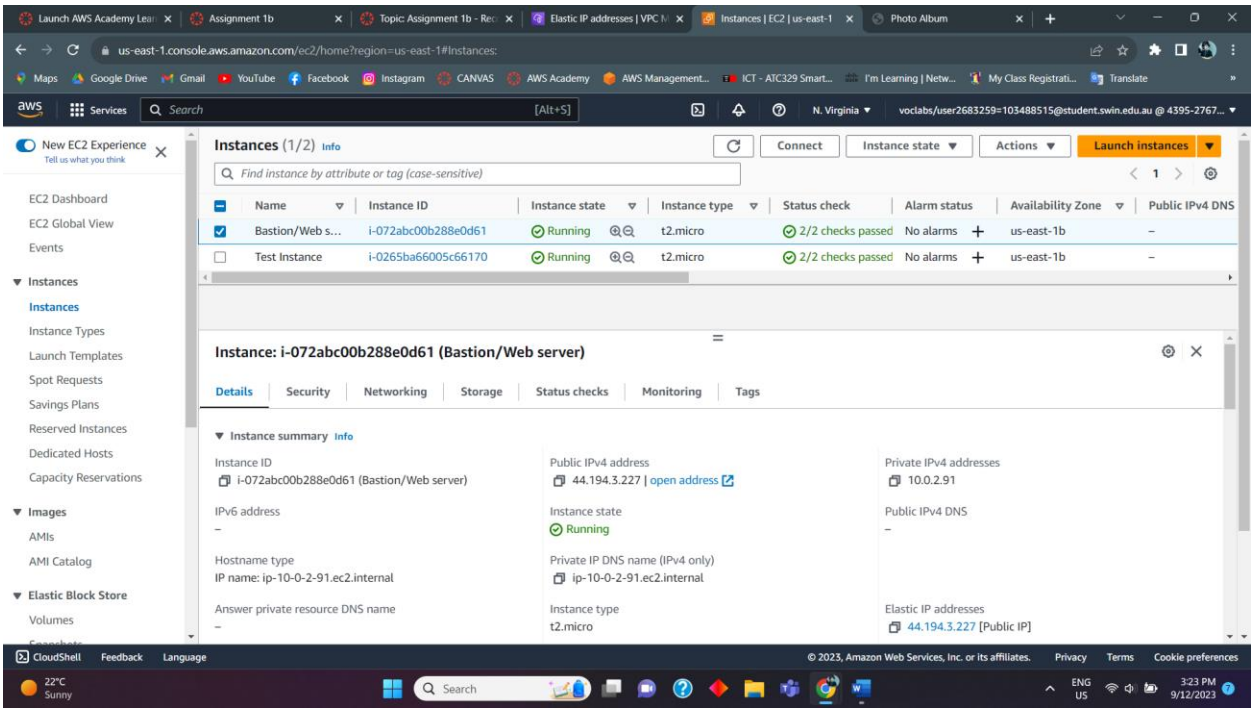

Figure 11 Create the two necessary instances.

1.4. RDS database instance

This step will create an RDS database, put it in the private subnet 1 and connect it with the Bastion/Web Server Instance. Because I have done the part of connecting RDS with EC2, so the security group will also have the rds-ec2-2 group (I delete the first rds-ec2-1 by accident so I need to connect them again).

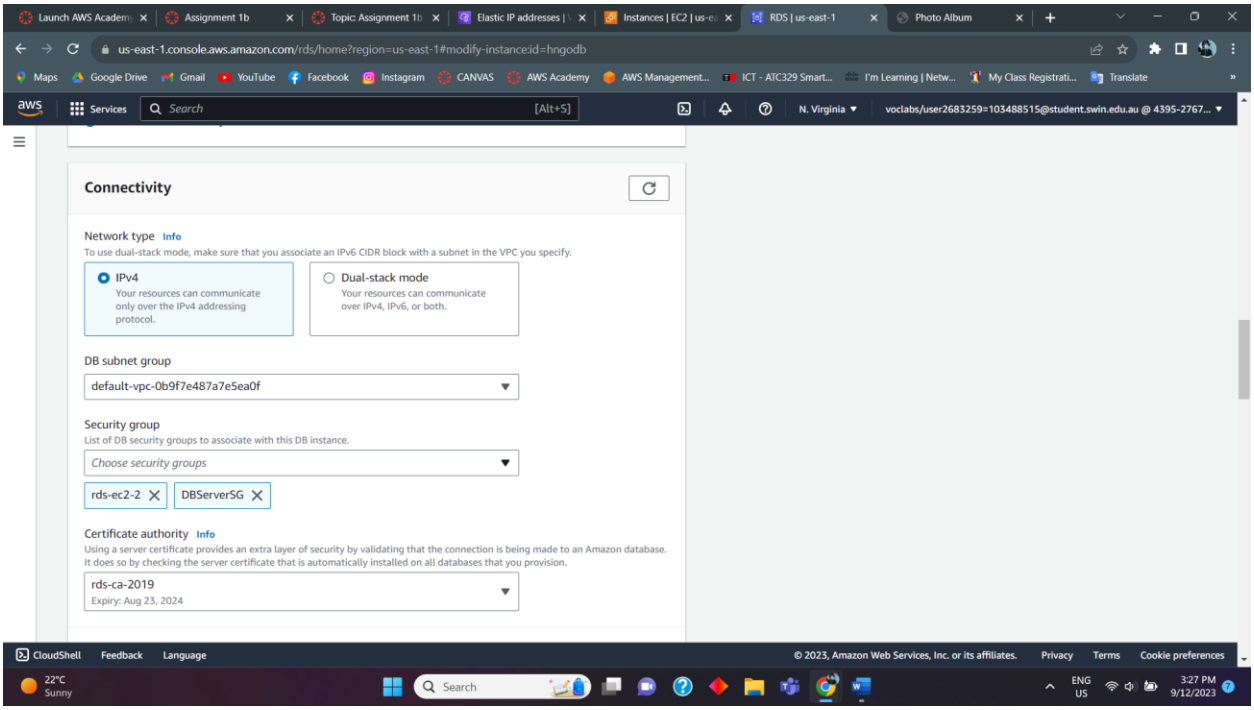

Figure 12 Modify it with the right security group.

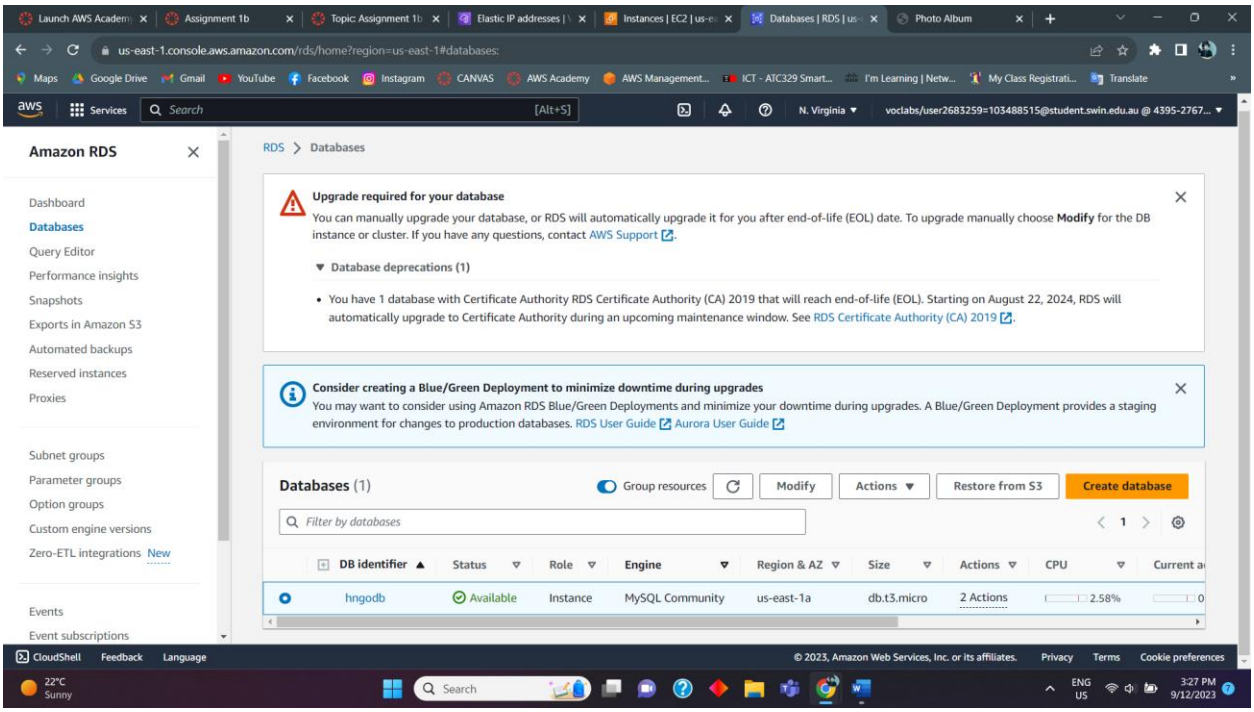

Figure 13 Create RDS named "hngodb".

After creating RDS, create and configure the right Database Table on RDS, using Web Server instance. To do that, PuTTY is required. The private key is downloaded from the learner lab. The host name will be the Elastic IP Address of the Bastion/Web Server Instance.

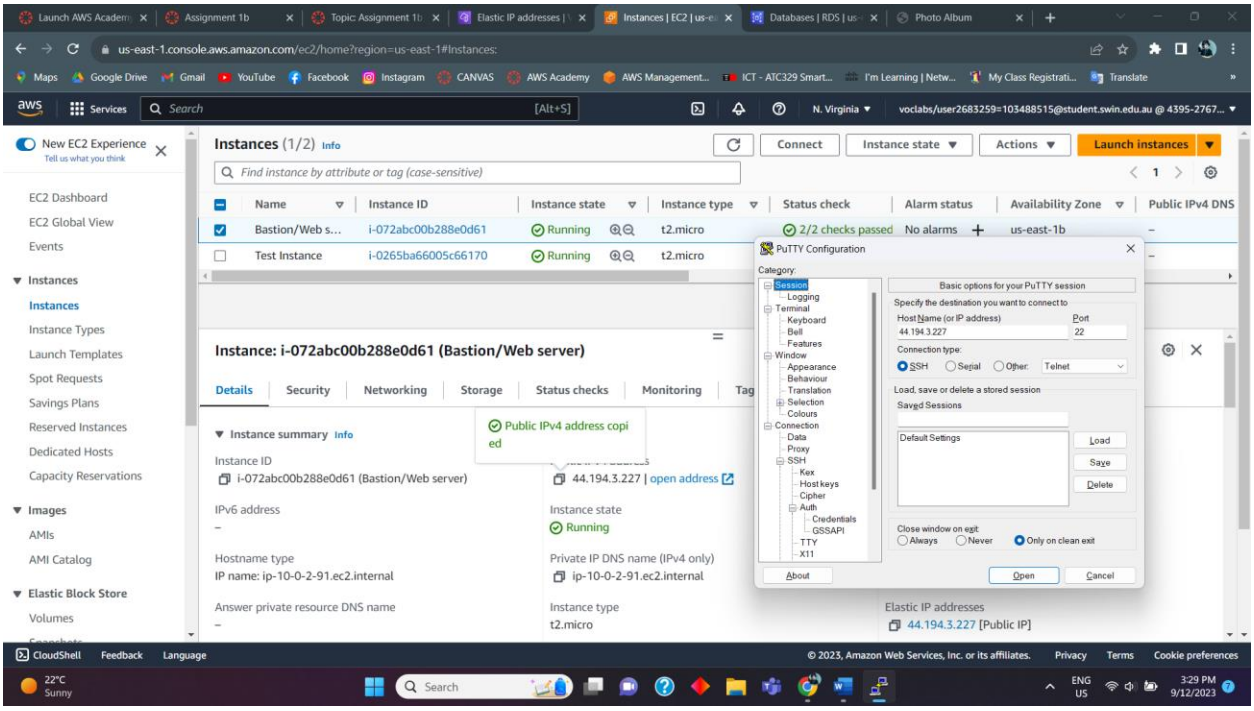

Figure 14 Access to Web Server Instance using PuTTY.

After successfully logging in to the terminal, the next step is to download the phpMyAdmin file into the EC2, because I have already done this, the result will be displayed down below.

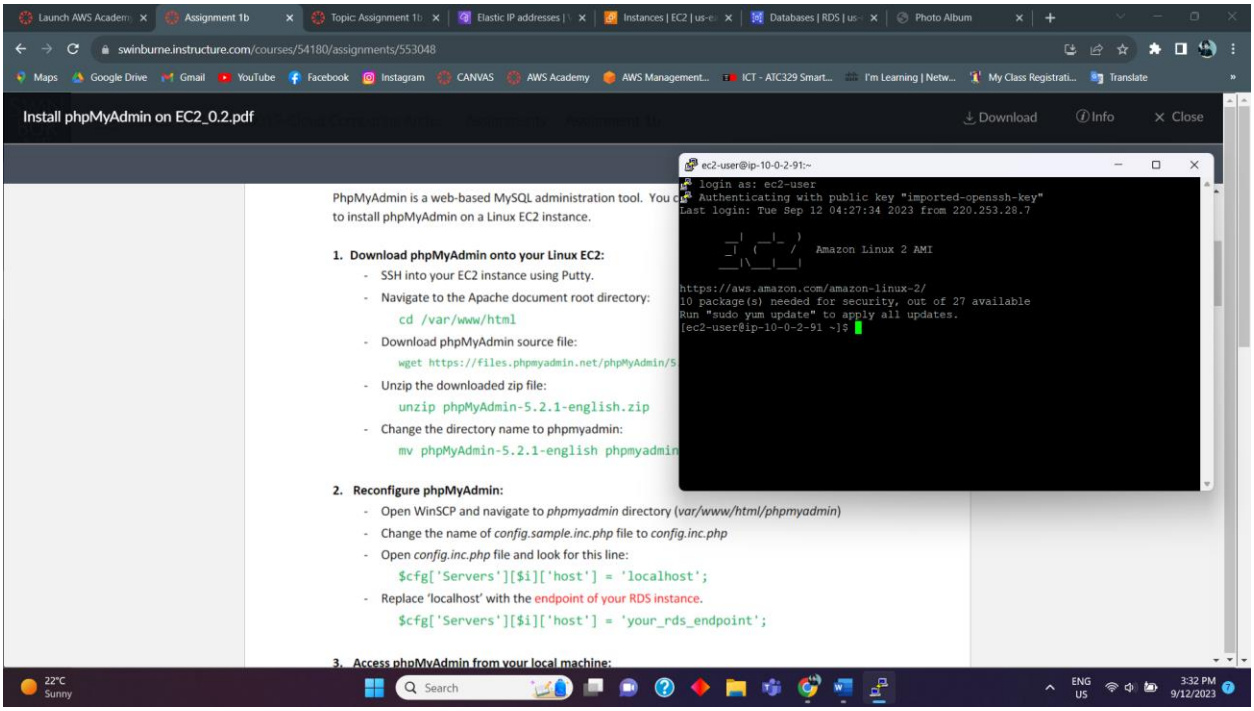

Figure 15 Log into Web Instance Linux

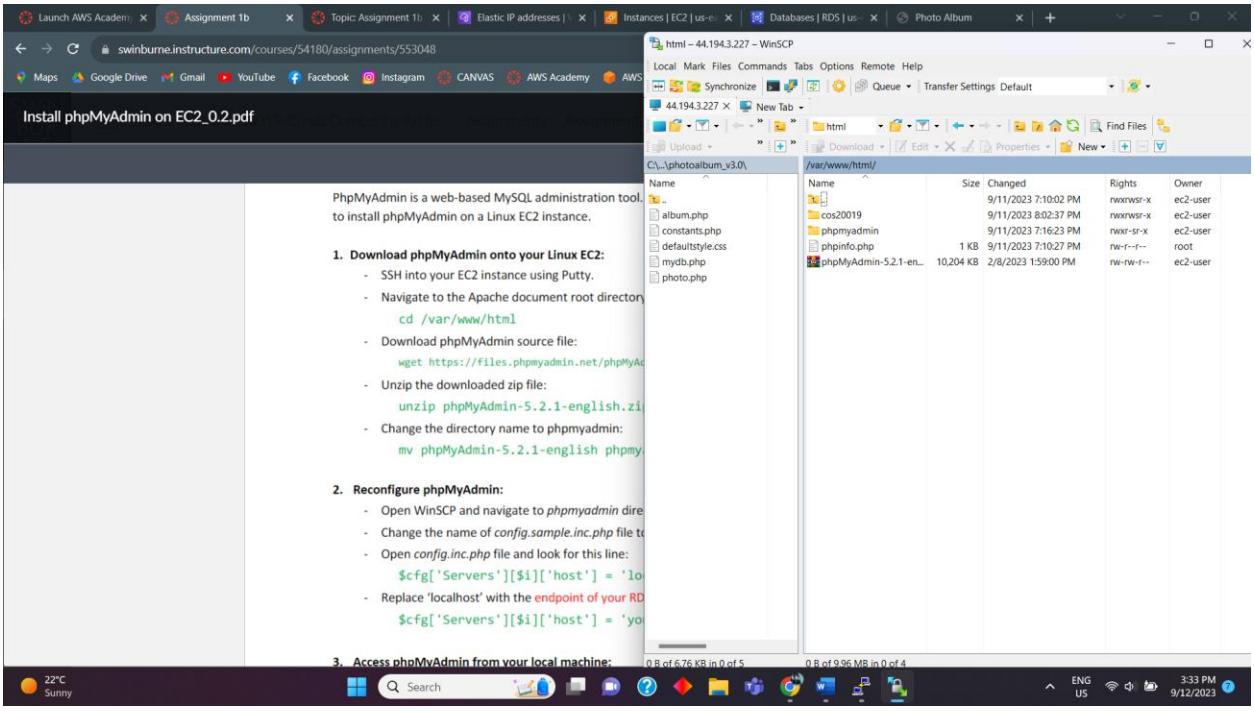

Figure 16 PhpMyAdmin file is downloaded and unzip successfully.

After that, the config.inc.php must be modified by replacing the host server with the link to the hngodb RDS.

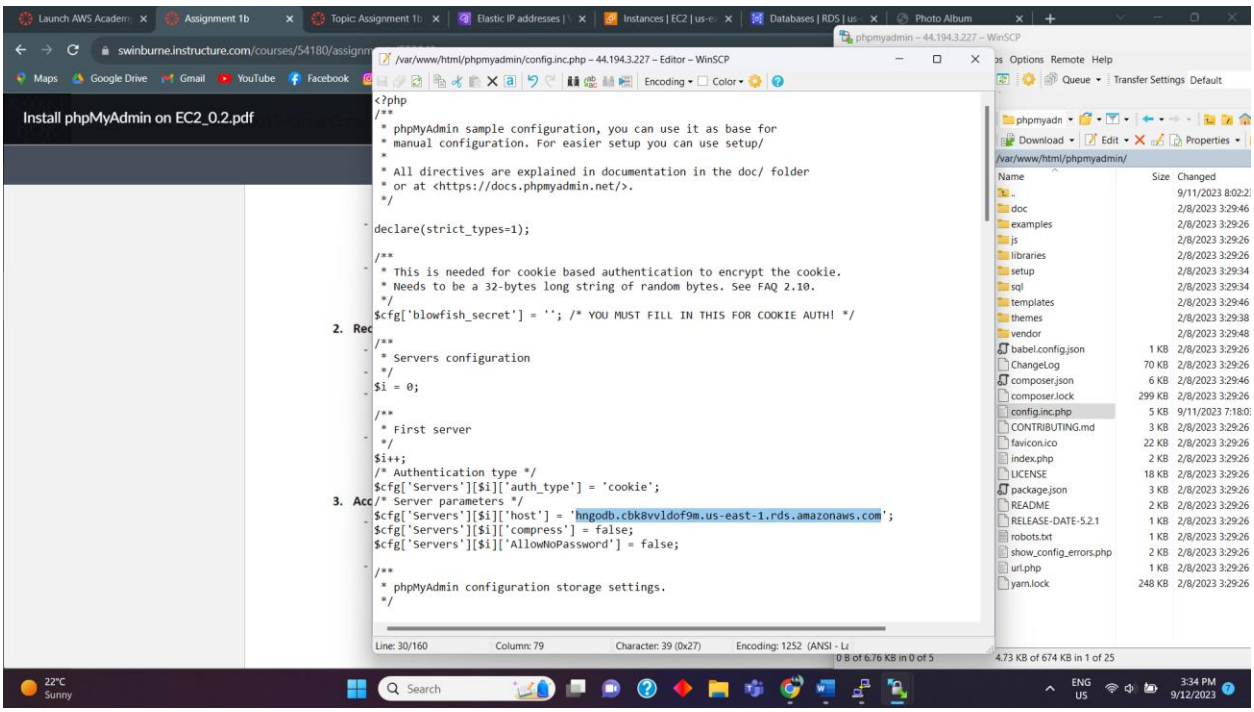

Figure 17 Modify the host server link.

For the next step, I went to the phpMyAdmin website and typed in the username and password to log in. The password and username have been written inside the constant file.

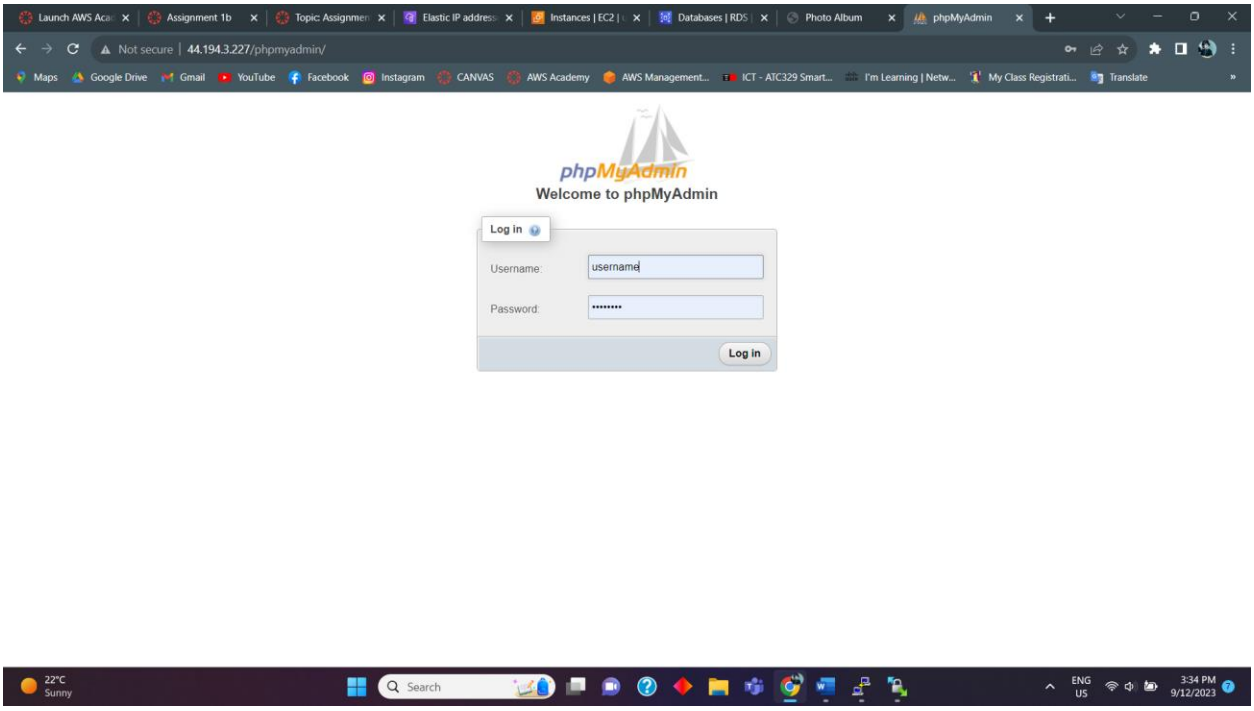

Figure 18 Log into phpMyAdmin

The next step is creating a photo database, where will keep the table “photos”.

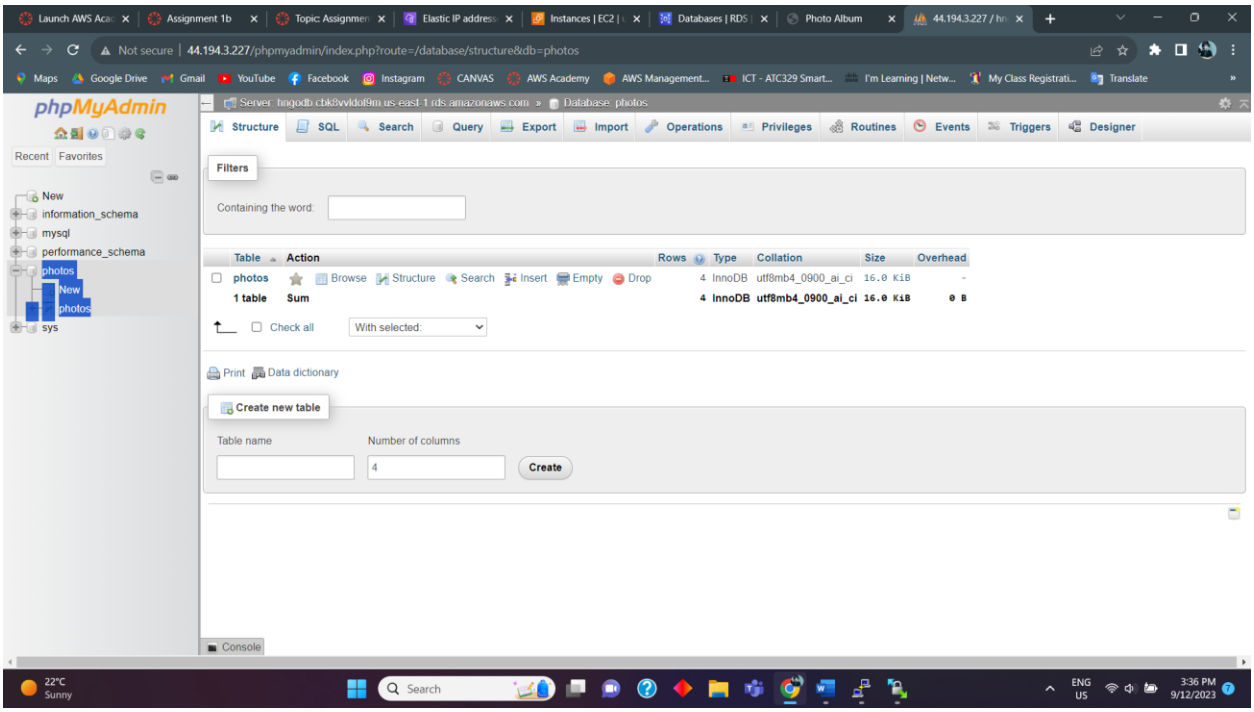

Figure 19 Create a database "photos" and table called "photos" as well.

According to the requirement, the table has been created with 5 columns: photo title, description, creation date, keywords, and reference, four varchar (255) columns and one date column.

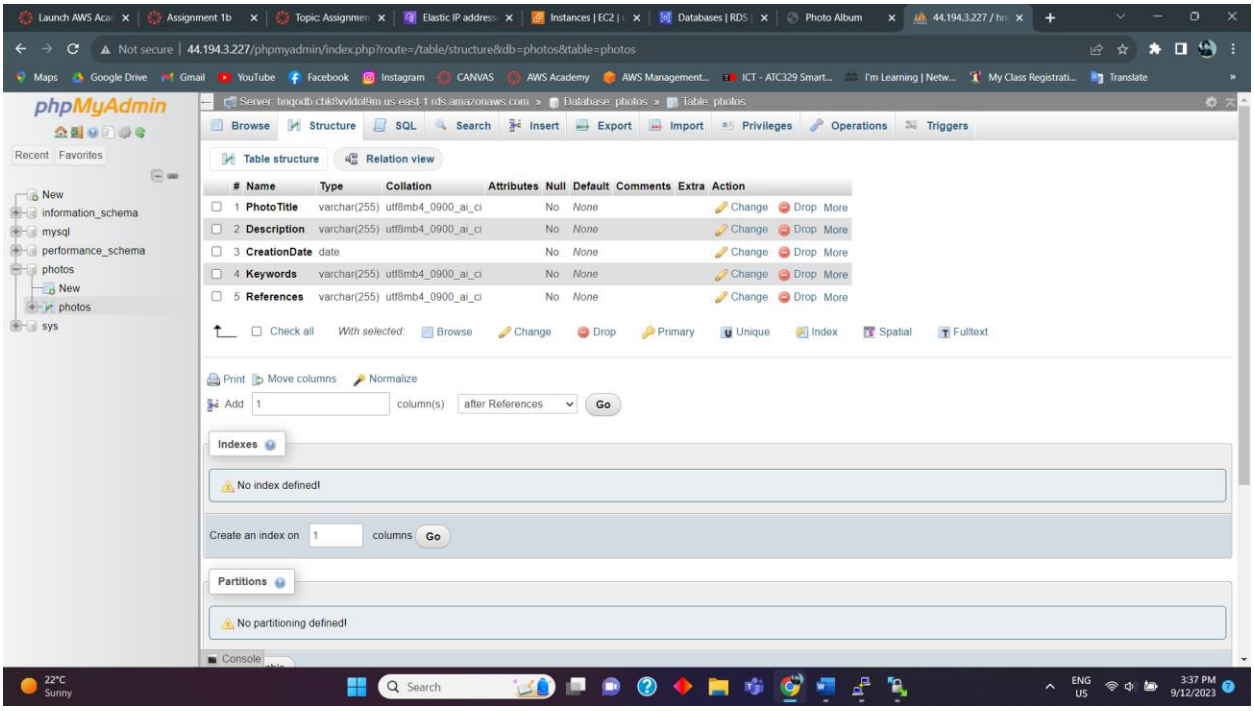

Figure 20 Table with 5 columns has been created.

1.5. Network ACL

To enhance the server's security, the next step is creating and implementing a Network ACL named "PublicSubnet2NACL" for Public Subnet 2. This ACL should strictly adhere to the principle of least privilege, allowing only essential traffic. Specifically, it should permit SSH (port 22) traffic from any source for Webserver instance access, allow ICMP traffic exclusively from the Test instance's subnet, and enable other required traffic to ensure full functionality of the Photo Album website for users everywhere.

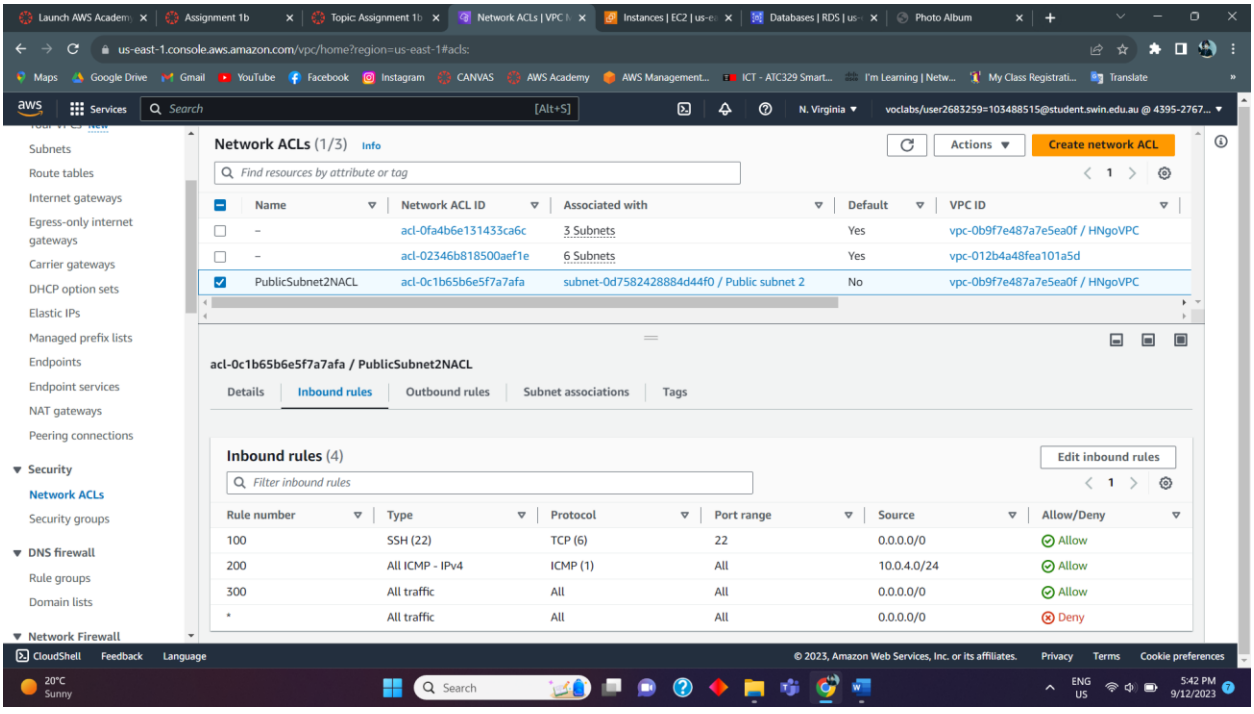

Figure 21 PublicSubnet2NACL is created with inbound and outbound rules, and connect it with public subnet 2

III. Functional requirements of Photo Album website

2.1. Photo storage

For this step, it is required to create an S3 Bucket with appropriate access policy.

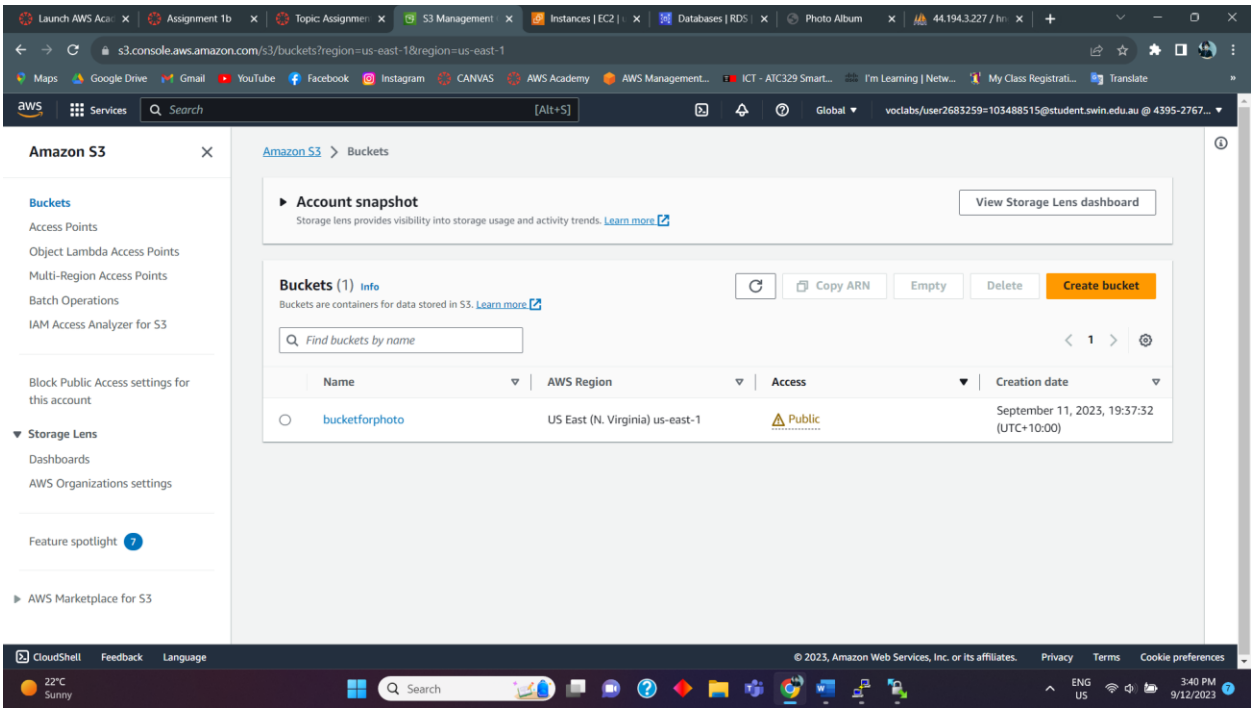

Figure 22 S3 bucket called "bucketforphoto" is created.

All objects (photos) in this S3 bucket must become publicly available. The policy below enables public access to all available objects in this S3 bucket.

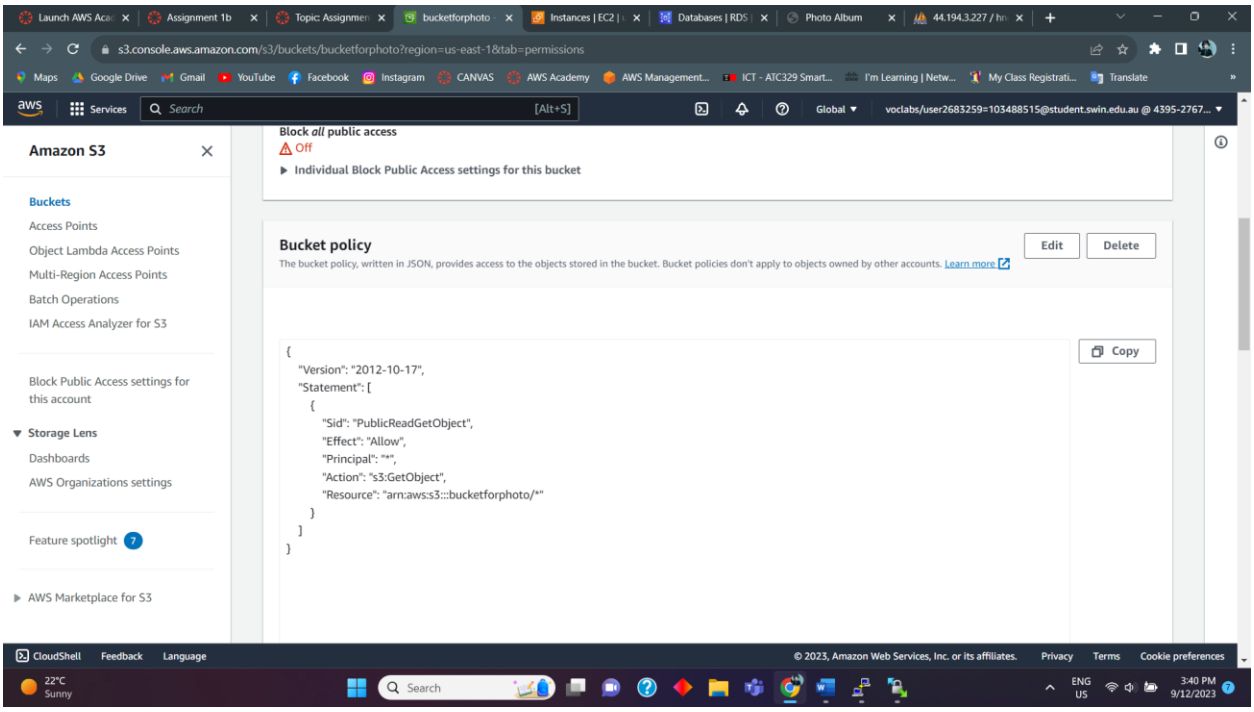

Figure 23 Policy for public access object in S3

2.2. Photo meta-data in RDS Database

Four logo photos of four universities have been uploaded in s3 bucket and get ready to transfer to phpMyAdmin.

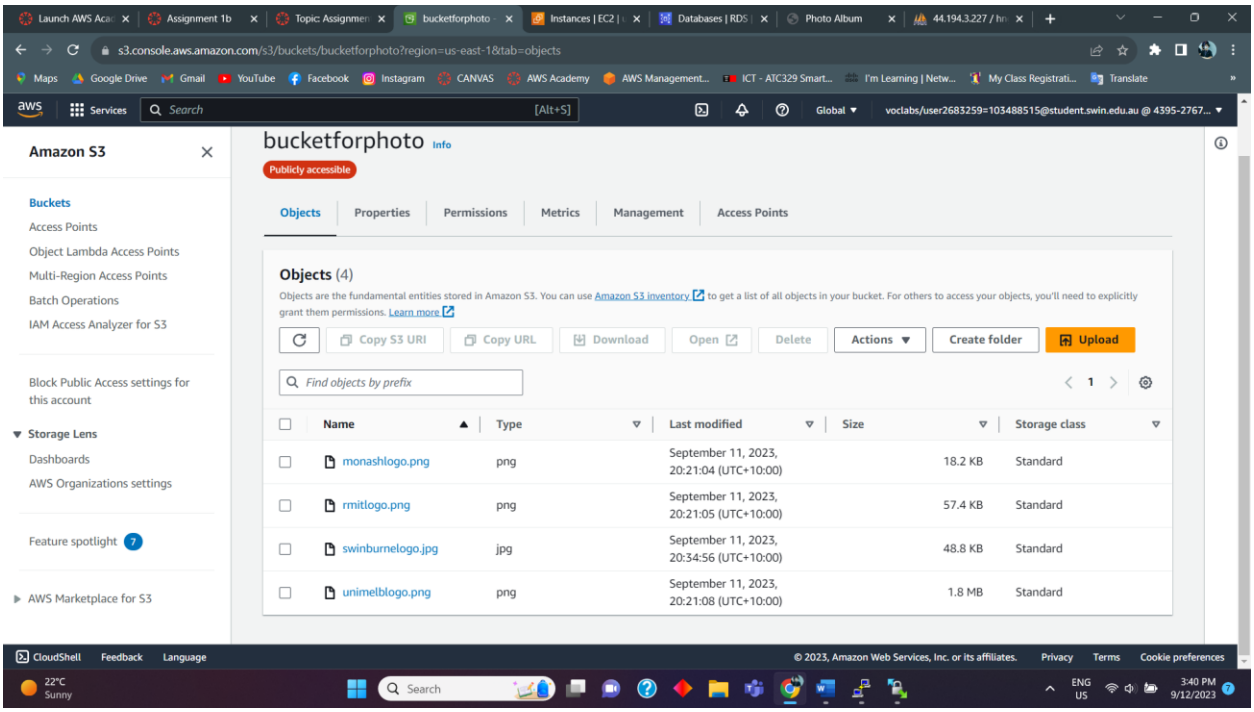

Figure 24 Upload four photos to S3 "bucketforphoto"

For the next step, the information of each photo must be inserted correctly, and the reference should be pre-signed URL of the photo in the S3 bucket, so the system will know where to get the photo and display it correctly to the website.

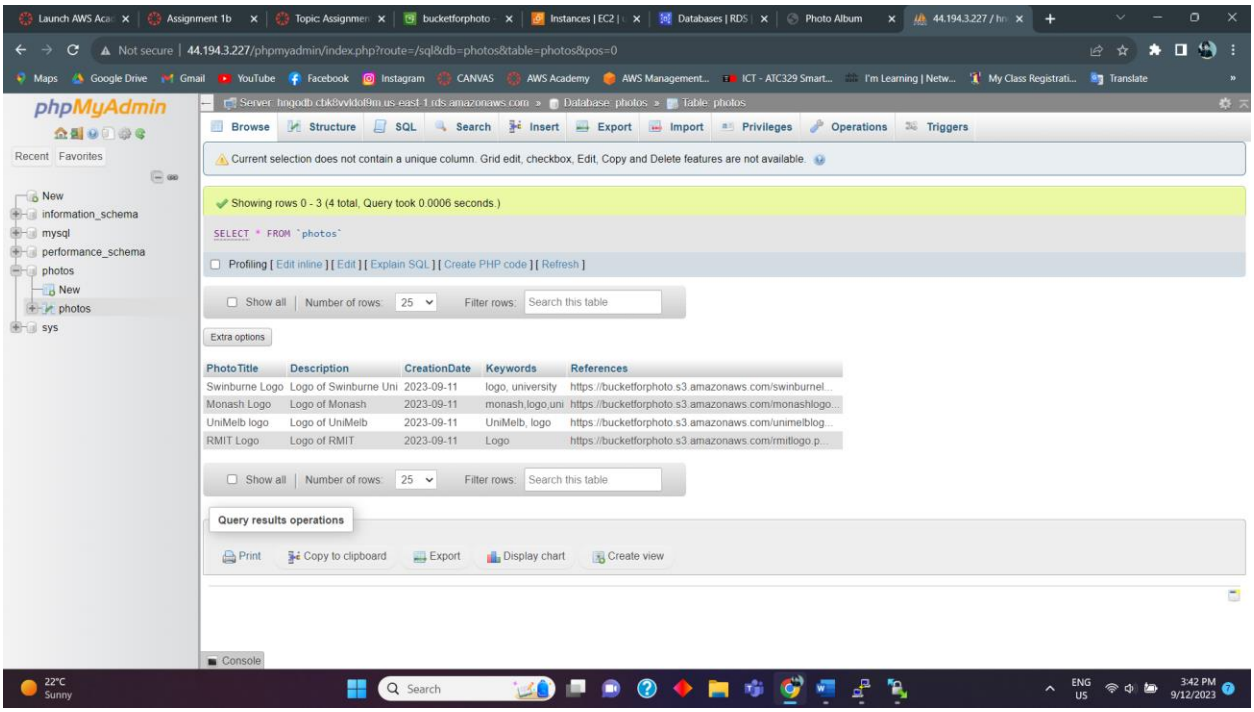

Figure 25 Successfully upload four photos to phpMyAdmin.

2.3. Photo Album website functionality

The website must be able to list all the photos (stored in the S3 bucket) along with their meta-data (stored in the database). In this case, I have Modified the constants.php file in the provided code using available information from the S3 bucket and RDS database.

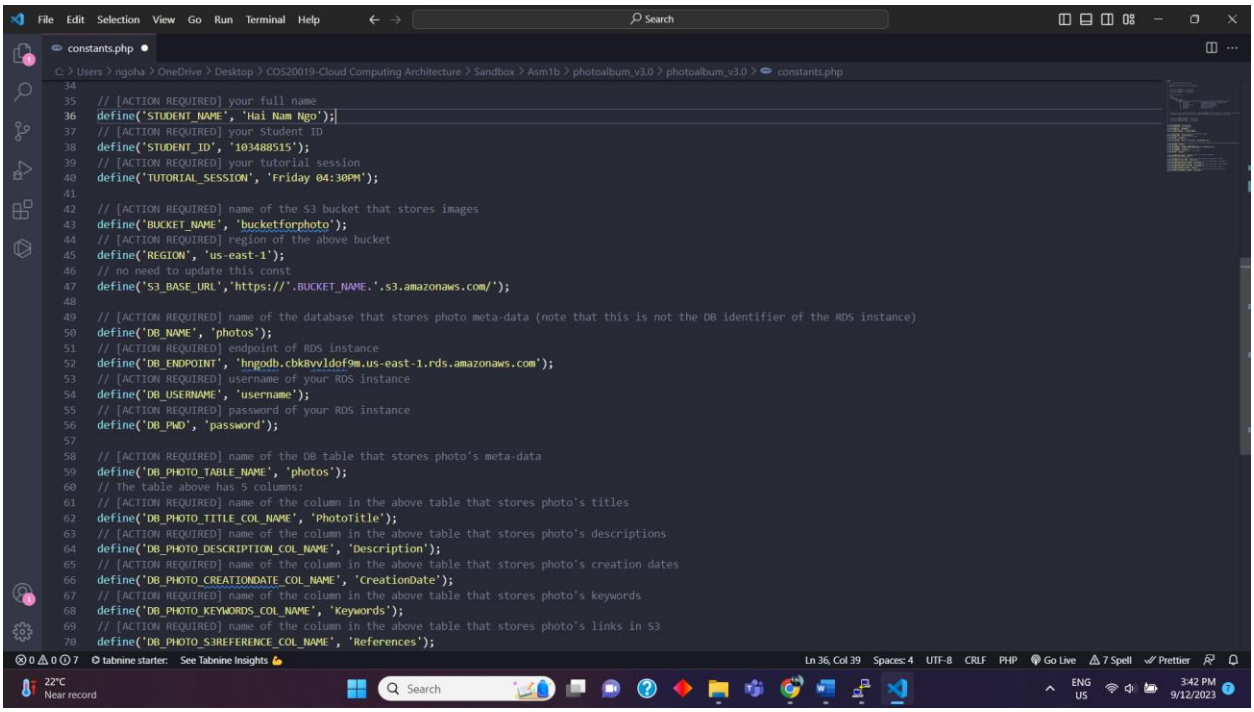

Figure 26 Modify the given file with correct information.

Next step is Using WinSCP to upload five sources code for the website.

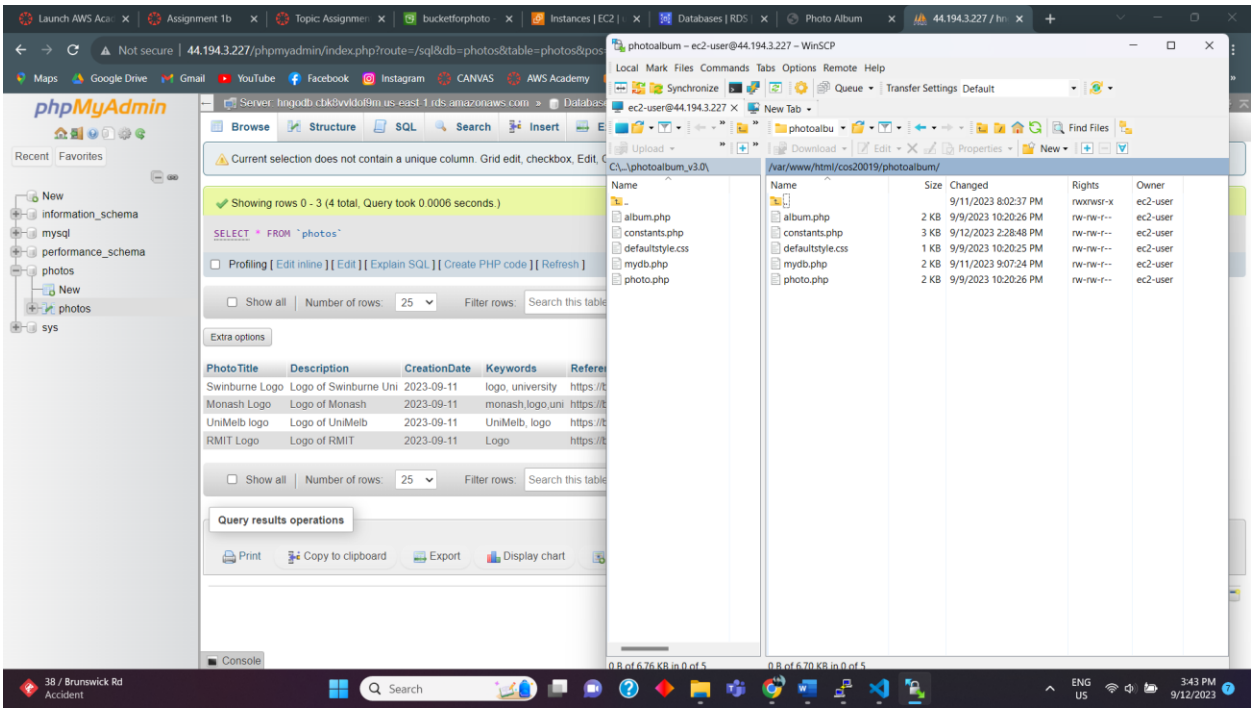

Figure 27 Upload source code files to the server

After all these steps, the website should successfully display four photos with additional information, along with my student information. (The link to access is already given on the first page).

Figure 28 Final result of album website

IV. Testing

This part is created to test if it is possible to ping Bastion/ Web Server Instance from Test Instance.

First, we can put the private key into the Pageant, so when we access Test Instance, it is unnecessary to give the private key again. When we have accessed Test Instance EC2 Linux, we can use “ping” command to ping the Bastion/Web Server Instance. We need to ping the private IPv4 address to make it work. In this case, the ping command will be “ping 10.0.2.91”.

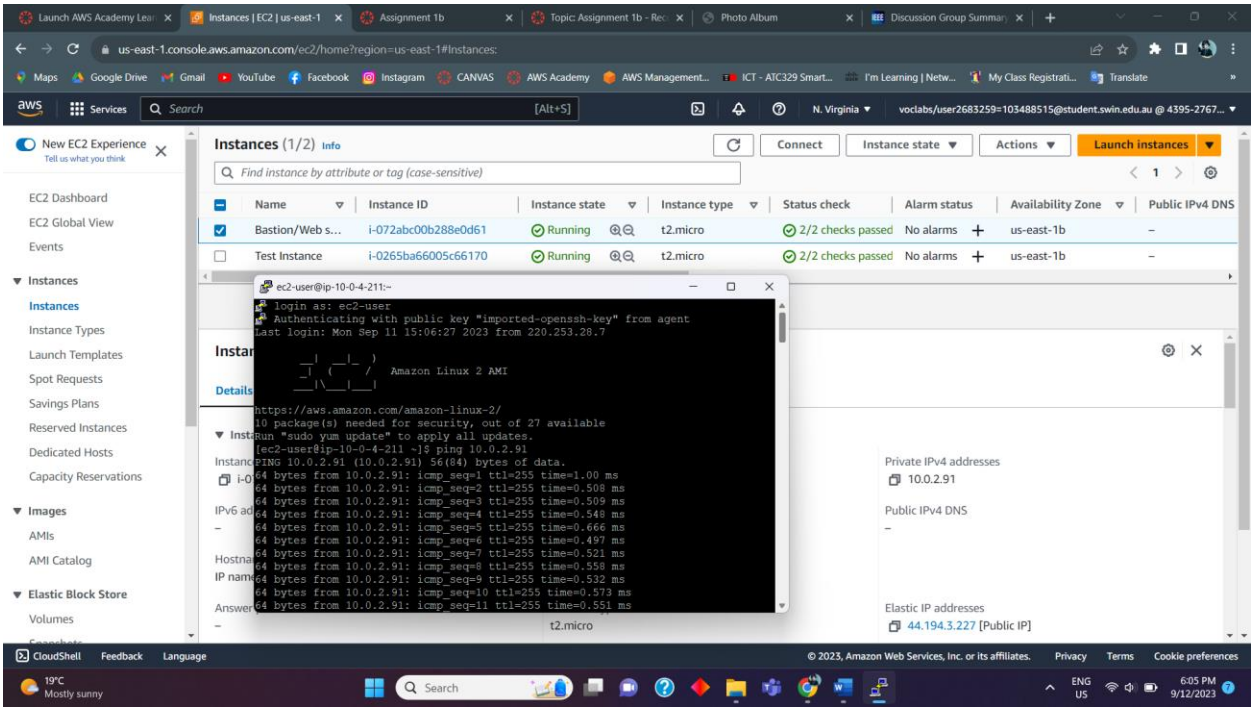

Figure 29 Test Instance successfully ping Bastion/Web Server Instance