



## COS20019 - Cloud Computing Architecture

### Assignment 2

**\*\*\*Developing a highly Available Photo Album Website\*\*\***

**Student Name:** Lau NgocQuyen

**Student Identification:** 104198996

**Grade Weighting:** 15%

**Deadline: 9 AM (AEST), Monday of Week 10**

**Late submission penalty:** 10% of total available marks per day.

#### 1. Introduction

- To create a reliable picture album website, using Amazon Web Services (AWS) services is my go-to choices. AWS provides a various services comes with superious functions, including storage, hosting, databases, caching, load balancing, monitoring, and so on . Using these services, we can build a strong and high durability picture album website with innovative features that outperform past jobs. AWS services provide the infrastructure and tools needed to assure high availability and optimal performance while storing, retrieving, and managing picture albums.

#### 2. Foundation and Infrastructure

- To prepare for the deployment phase, we must first establish a solid foundation for our website. Configure VPC and subnets flawlessly based on the below Architecture structure. The VPC will provide the necessary networking infrastructure, ensuring isolation, security, and control over our resources. Properly configuring the subnets within the VPC ensures secure communication between website components. This step is crucial for building a reliable photo album website.

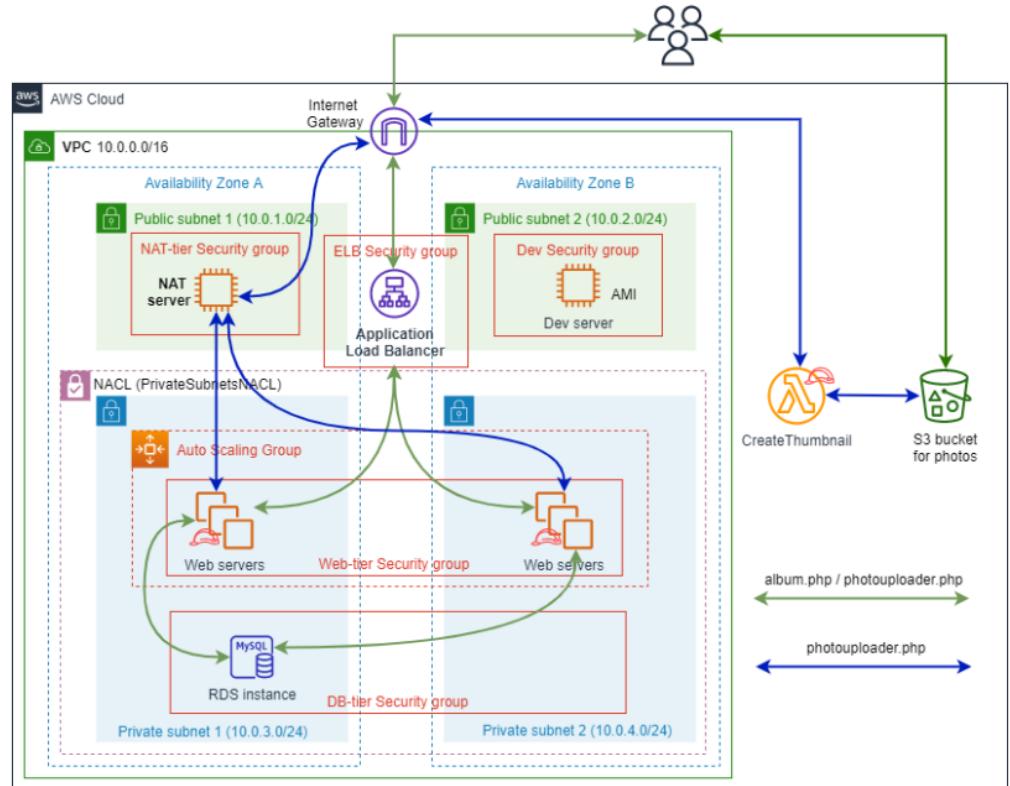


Image 1 – Architecture diagram

- Since the goal has previously been introduced in Assignments 1A and 1B, this work is not too tough. Following configuration, the following VPC road map for our VPC and the corresponding subnets may exist:

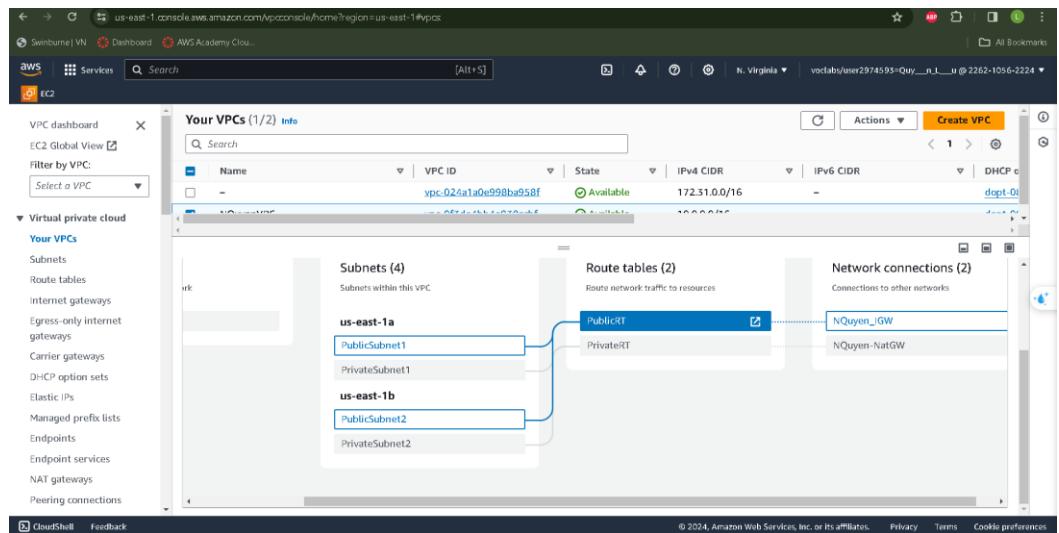


Image 2 – VPC resource linked road map

Name	Subnet ID	State	VPC	IPv4 CIDR
PublicSubnet1	subnet-00e4d78846eeaeaf8	Available	vpc-0f3de4bb4e938acbf	10.0.1.0/24
PublicSubnet2	subnet-0931f0e05da53497b	Available	vpc-0f3de4bb4e938acbf	10.0.2.0/24
PrivateSubnet2	subnet-02ce2bdefb25983a2	Available	vpc-0f3de4bb4e938acbf	10.0.4.0/24
PrivateSubnet1	subnet-05d2fb20bd314cba6	Available	vpc-0f3de4bb4e938acbf	10.0.3.0/24

*Image 3 – IPv4 CIDR for every subnet*

- During the development phase, the PublicSubnet2 Subnet will be sent to the IGW to test website operation. If only our website is completely operational, an Amazon Machine Image (AMI) will be produced **for PublicSubnet2**, allowing for deployment in multiple locations as needed. **Private Subnet1** and **PrivateSubnet2** will be used to deploy an autoscaling group, which will be routed to the NAT Gateway. It's vital to note that this assignment will use a NAT Gateway rather than a NAT instance. The **PublicSubnet1** subnet will operate the NAT Gateway, which will be routed to the Internet Gateway. The graphic below demonstrates the configuration of NAT gateway

The screenshot shows the AWS VPC NAT gateways console. On the left, there's a navigation sidebar with options like Virtual private cloud, NAT gateways, Security, and CloudShell. The main area displays a table titled 'NAT gateways (1/1) info' with one entry:

Name	NAT gateway ID	Connectivity type	State	Primary public IPv4 address	Primary private IPv4 address
NQuyen-NatGW	nat-0053fd9ce859dc08a	Public	Available	44.207.65.240	10.0.1.109

Below the table, there's a detailed view for the selected NAT gateway, showing its ARN, Subnet, and VPC information.

*Image 4: NatGW in Nat\_Server\_Public1*

- NAT will also bring this assignment's first phase to an end. After the VPC is built, we will establish Organizations that exist on these subnets.

### 3. Functionalities of the Website

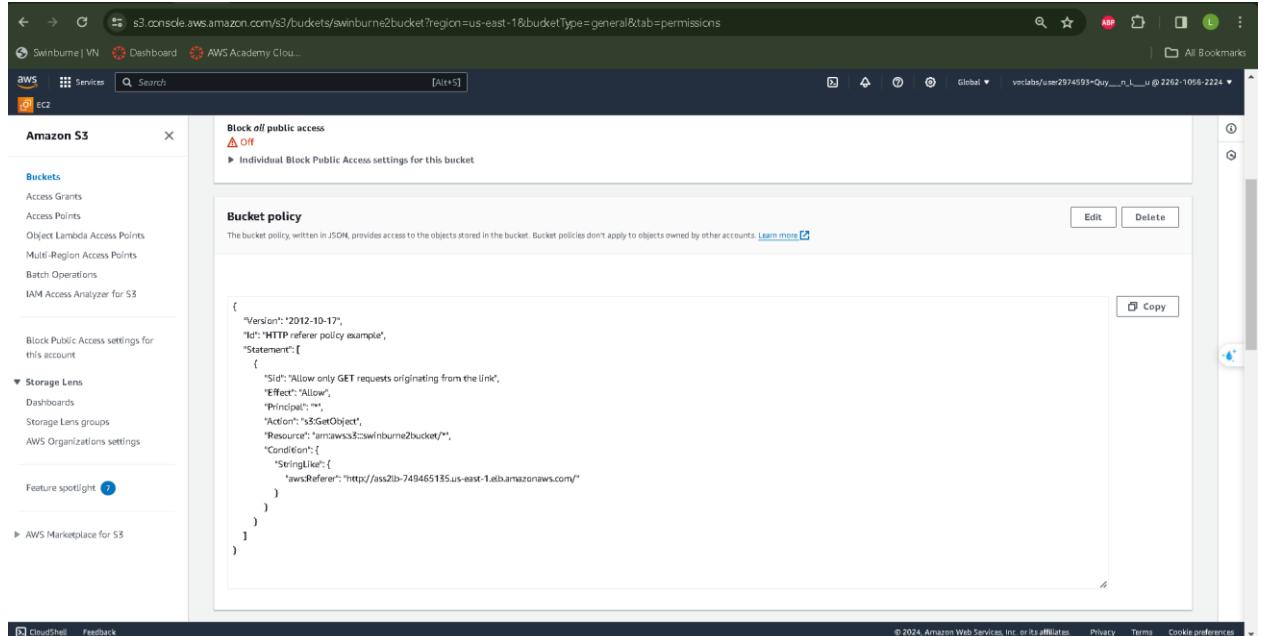
- To make sure that our websites developed in subnet 2, S3 would be mandotarily operated .

The screenshot shows the AWS S3 console. The left sidebar includes options like Buckets, Storage Lens, and Feature spotlight. The main area displays the 'Amazon S3' dashboard with an 'Account snapshot' summary and a table of 'General purpose buckets'.

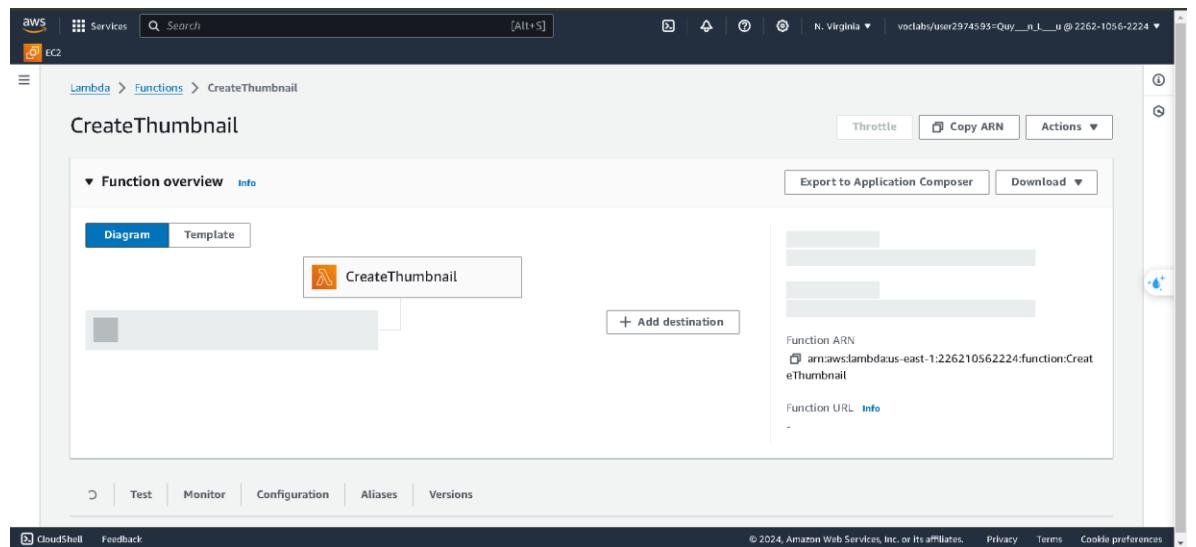
Name	AWS Region	Access	Creation date
swinburne2bucket	US East (N. Virginia) us-east-1	Public	March 1, 2024, 11:42:27 (UTC+07:00)
swinbucketphoto	US East (N. Virginia) us-east-1	Public	February 5, 2024, 23:04:54 (UTC+07:00)

*Image 5: Assignment 2's Amazon S3 bucket*

- Based on the architecture structure, our Amazon S3 bucket will only allow entry from and by the **Load Balancer**. However, configure restriction permission for Amazon S3 bucket policy is required.



*Image 6: Amazon Bucket's Policy*



*Image 7: Creating Lambda function by CreateThumbnail*

```

1 import boto3
2 import os
3 import sys
4 import json
5 import unquote_plus
6 from PIL import Image
7 from botocore.exceptions import ClientError
8
9 """
10 To build an AWS Lambda deployment package, see:
11 https://stackoverflow.com/a/61544380/2599491
12 https://docs.aws.amazon.com/lambda/latest/dg/with-s3-tutorial.html#with-s3-tutorial-create-function-package
13
14 This Lambda function accepts the following input: {"bucketName": "your-photo-bucket-name", "fileName": "your-photo.png"}
15 ONLY PNG FILES ARE SUPPORTED!
16 This Lambda function download your-photo.png from your-photo-bucket-name S3 bucket, resizes the pic, then upload the resized pic (resized-your-photo.png)
17
18
19 s3_client = boto3.client('s3')
20
21 def resize_image(image_path, resized_path):
22     with Image.open(image_path) as image:
23         image.thumbnail(tuple(x / 2 for x in image.size))
24         image.save(resized_path)
25
26
27 def lambda_handler(event, context):
28     try:
29         bucket_name = event["bucketName"]
30         file_name = event["fileName"]
31         key = event['prefix']+file_name
32         tmpkey = key.replace('/', '_')
33         download_path = f"/tmp/{tmpkey}.format(uuid.uuid4()), tmpkey)

```

*Image 8: Upload the provided lambda file to code source*

- The function package is already given, therefore we simply need to test it to ensure accuracy.

*Image 9: Create test case for lamda provie*

- After completing the test case, we will set up a Relational Database Service and create a database instance within a second.

The screenshot shows the Amazon RDS console with the URL <https://us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#database-id=database-2;is-cluster=false>. The left sidebar is titled "Amazon RDS" and includes sections for Dashboard, Databases (selected), Query Editor, Performance insights, Snapshots, Exports in Amazon S3, Automated backups, Reserved instances, Proxies, Subnet groups, Parameter groups, Option groups, Custom engine versions, Zero-ETL integrations, and Events. The main content area is titled "database-2" and has tabs for "Summary" (selected), Connectivity & security, Monitoring, Logs & events, Configuration, Zero-ETL integrations, Maintenance & backups, and Tags. The "Summary" tab displays details such as DB identifier (database-2), Status (Available), Role (instance), Engine (MySQL Community), and Recommendations (2 informational). The "Connectivity & security" tab shows the Endpoint (database-2.cb6gi6wksph.us-east-1) and Networking (Availability Zone us-east-1a). The VPC security groups listed are DBSG2 (sg-026b1d2a26c084ccb5).

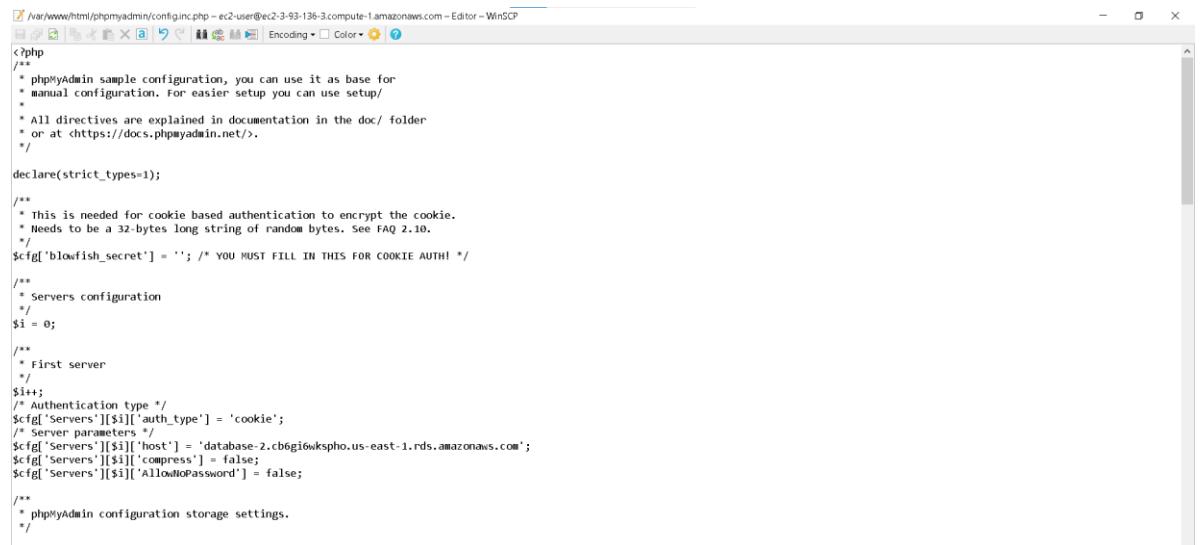
*Image 10: database-2 in Amazon RDS for image's database*

The screenshot shows the Amazon RDS console with the URL <https://us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#database-id=database-2;is-cluster=false;tab=configuration>. The left sidebar is identical to Image 10. The main content area is titled "Instance" and displays detailed configuration settings for the database instance. The "Configuration" section includes DB instance ID (database-2), Engine version (8.0.35), DB name (empty), License model (General Public License), Option groups (default:mysql-8-0 In sync), Amazon Resource Name (ARN) (arnaws:rds:us-east-1:226210562224:db:database-2), Resource ID (db-B3H7GVPNUNE7SVI3Q2FFR6OEAQ), Created time (March 03, 2024, 01:51 (UTC+07:00)), and DB instance parameter group (empty). The "Instance class" is db.t3.micro, with vCPU (2), RAM (1 GB), and Storage type (General Purpose SSD (gp2)). The "Storage" section includes Encryption (Enabled), AWS KMS key (aws/rds), Storage (20 GiB), Provisioned IOPS (-), Storage throughput (-), Storage autoscaling (Enabled), Maximum storage threshold (1000 GiB), and Storage file system configuration (Current). The "Performance Insights" section indicates that Performance Insights are enabled but turned off.

*Image 11: RDS Database detailed*

- Before uploading our website to the development server, we need to complete all missing code. We will need to alter the constant.php and Config.inc.php files in the

given ZIP package.



```

<?php
/**
 * phpMyAdmin sample configuration, you can use it as base for
 * manual configuration. For easier setup you can use setup/
 *
 * All directives are explained in documentation in the doc/ folder
 * or at <https://docs.phpmyadmin.net/>.
 */
declare(strict_types=1);

/**
 * This is needed for cookie based authentication to encrypt the cookie.
 * Needs to be a 32-bytes long string of random bytes. See FAQ 2.10.
 */
$cfg['blowfish_secret'] = ''; /* YOU MUST FILL IN THIS FOR COOKIE AUTH! */

/**
 * Servers configuration
 */
$i = 0;

/**
 * First server
 */
$i++;
/* Authentication type */
$cfg['Servers'][$i]['auth_type'] = 'cookie';
/* Server parameters */
$cfg['Servers'][$i]['host'] = 'database-2.cb6gi6wkspho.us-east-1.rds.amazonaws.com';
$cfg['Servers'][$i]['compress'] = false;
$cfg['Servers'][$i]['AllowNoPassword'] = false;

/**
 * phpMyAdmin configuration storage settings.
 */

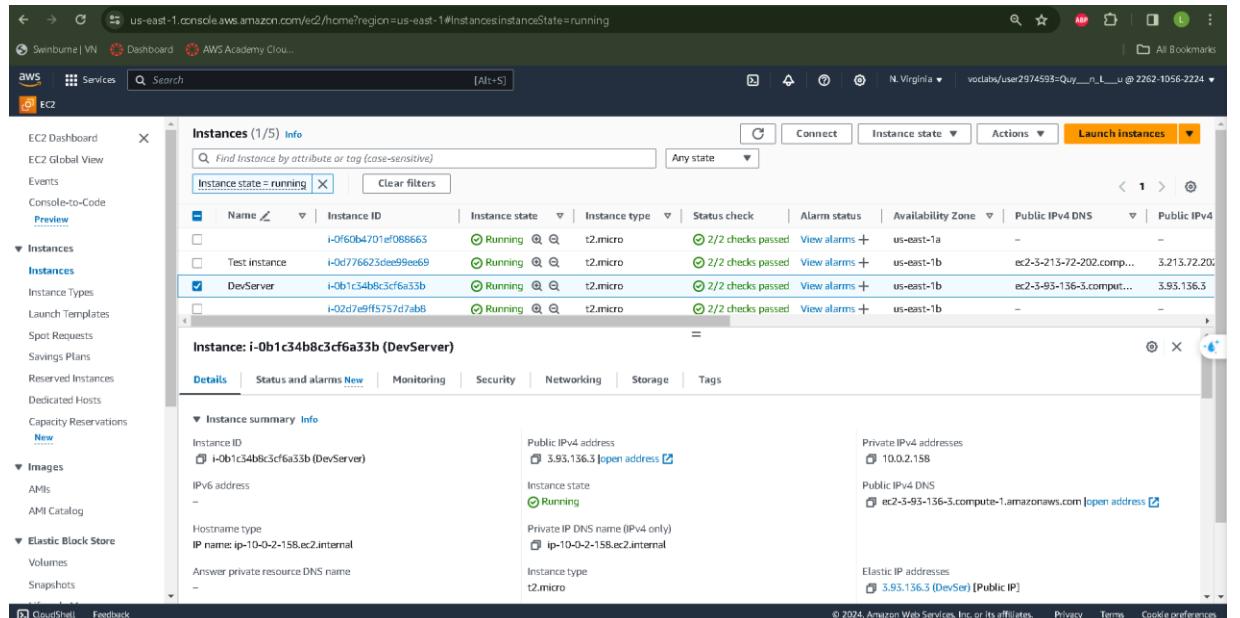
```

*Image 12: RDS end point replacement for server*

- Once all components are successfully conImaged, create an EC2 instance on the development server.

## 4. Built of the Website using EC2 and AMI

- EC2 instance function should be choose correctly in order to hosting the website



Metric	Value
InstanceId	i-0b1c34b8c3cf6a33b
InstanceType	t2.micro
PublicIP	3.93.136.3
PrivateIP	10.0.2.158
Region	N. Virginia
Timestamp	2024-01-15T10:00:00Z

*Image 13: DevServer Instance*

- To use the CreateThumbnail Lambda function to upload items and re-size the image to an S3 bucket, an IAM role is required.

The screenshot shows the AWS EC2 Instances page with the search bar set to 'Instance state = running'. A table lists five instances, with 'DevServer' selected. The 'Instance: i-0b1c34b8c3cf6a33b (DevServer)' details pane is open, showing the following information:

Hostname type	Private IP DNS name (IPv4 only)
IP name	ip-10-0-2-158.ec2.internal
Answer private resource DNS name	-
Auto-assigned IP address	ip-10-0-2-158.ec2.internal
IAM Role	LabRole
IMDSv2 Required	-
VPC ID	vpc-0f3d04b4e938acbf (hQuyenVPC)
Subnet ID	subnet-0931f0e05da53497b (PublicSubnet2)
Elastic IP addresses	3.95.136.3 (DevSet) [Public IP]
AWS Compute Optimizer finding	Opt-in to AWS Compute Optimizer for recommendations.
Learn more	-
Auto Scaling Group name	-

*Image 14: Additional settings of WebDevServerInstance*

- Verifying if an EC2 instance is accessible, associate Elastic IP address and see the website throughout public IPv4 DNS.

The screenshot shows a browser window with the URL 'ec2-3-93-136-3.compute-1.amazonaws.com'. The title bar says 'Test Page'. The page content is as follows:

This page is used to test the proper operation of the Apache HTTP server after it has been installed. If you can read this page, it means that the Apache HTTP server installed at this site is working properly.

If you are a member of the general public:  
The fact that you are seeing this page indicates that the website you just visited is either experiencing problems, or is undergoing routine maintenance.

If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.

For example, if you experienced problems while visiting www.example.com, you should send e-mail to "webmaster@example.com".

If you are the website administrator:  
You may now add content to the directory /var/www/html/. Note that until you do so, people visiting your website will see this page, and not your content. To prevent this page from ever being used, follow the instructions in the file /etc/httpd/conf.d/welcome.conf.

You are free to use the image below on web sites powered by the Apache HTTP Server:

Powered by APACHE 2.4

*Image 15: Deploying EC2 and Page Testing Flawlessly*

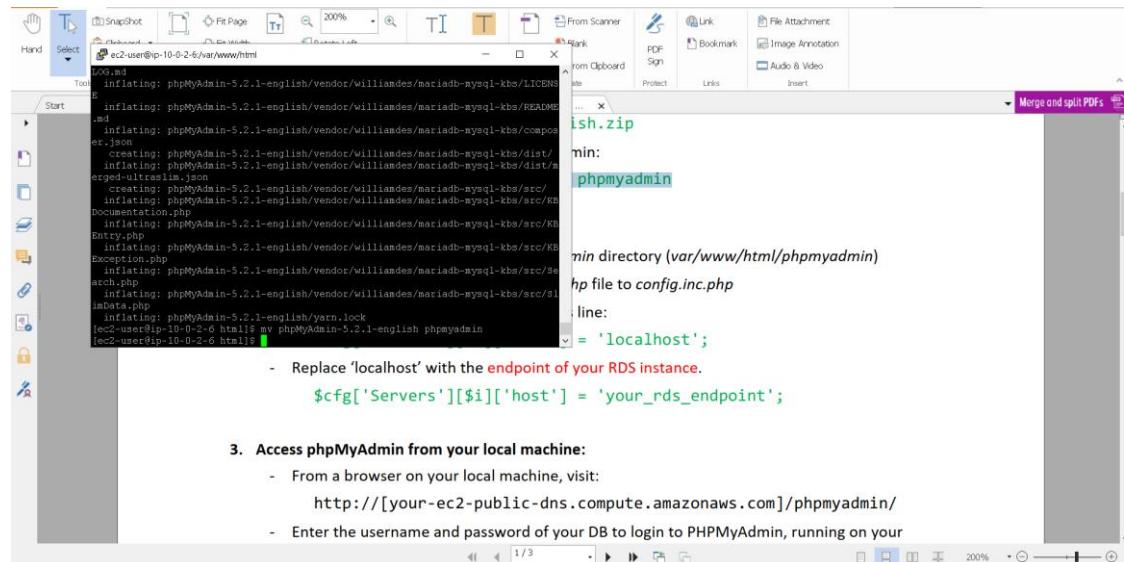


Image 16: WebDevServer SSH terminal using Putty

- Phpmyadmin creates meta-data for our database, monitors its operation. To connect our phpMyAdmin console to the RDS instance generated in stage 3, make a little change to config.inc.php.

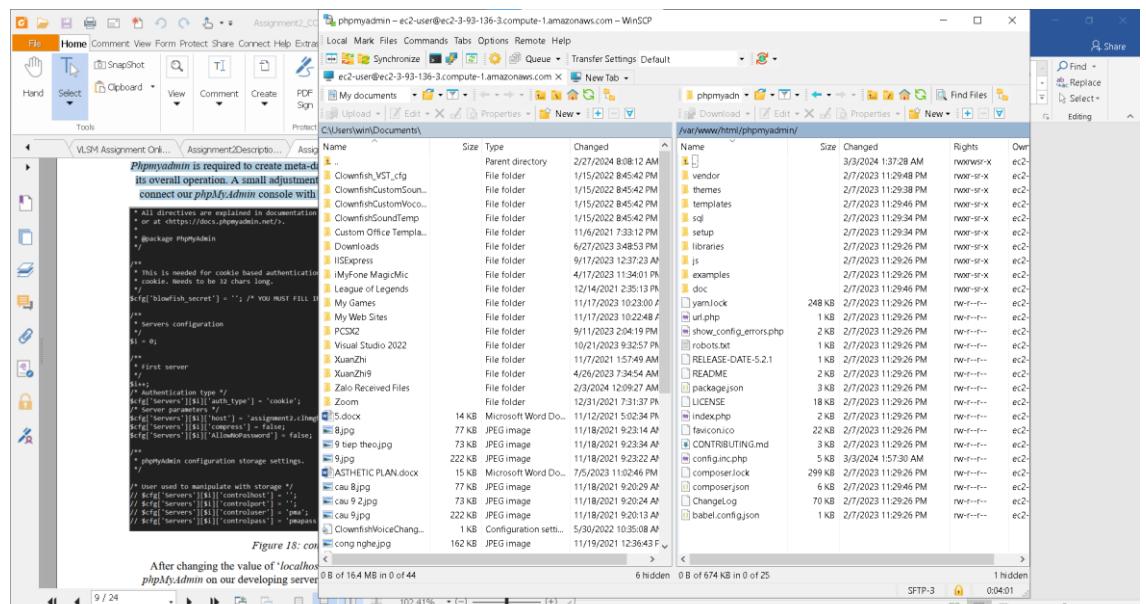


Figure 18: config.inc.php

*Image 17: Change the config.sample.inc.php into config.inc.php*

- Modifying 'localhost' option in config.inc.php allows us to utilize phpMyAdmin on our development server website to generate database metadata.

The screenshot shows the phpMyAdmin interface connected to a MySQL database named 'photos'. The left sidebar shows the database structure with tables like 'information\_schema', 'mysql', 'performance\_schema', 'Photos', and 'sys'. The main panel displays the 'photos' table with 9 rows. The SQL query at the top is:

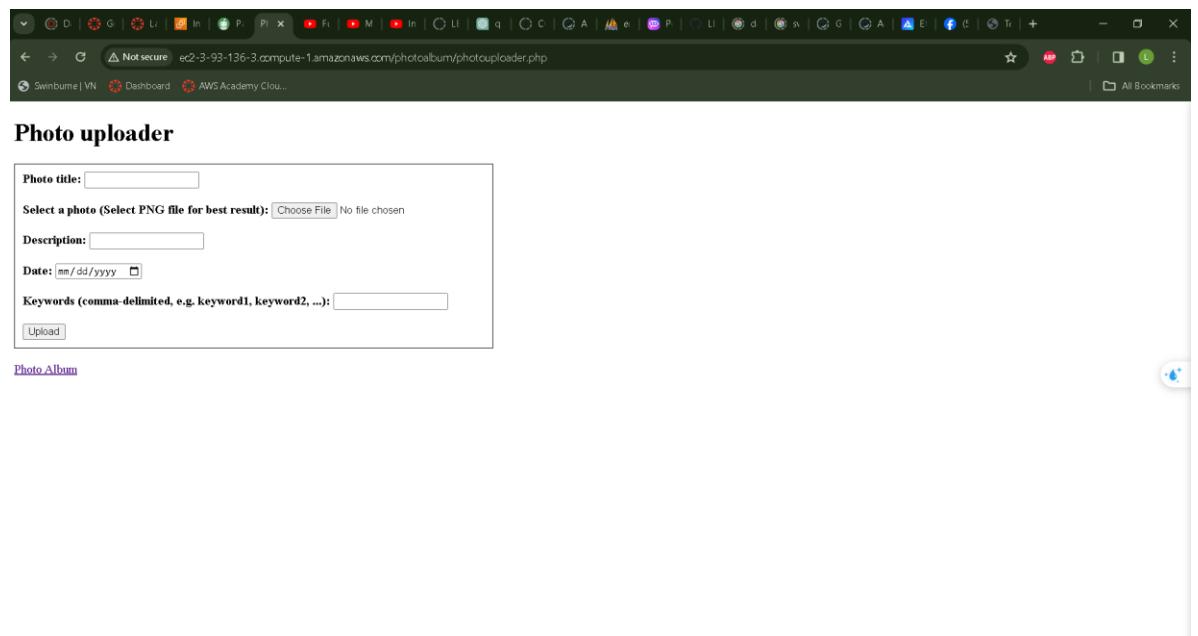
```
SELECT * FROM `photos`
```

*Image 18: Table photos in phpMyAdmin*

The screenshot shows the WinSCP interface with two panes. The left pane shows a local directory structure under 'Assignment2\_COS2001...' and a remote directory under '/var/www/html/photoalbum/'. The right pane shows a detailed list of files in the remote directory, including PHP scripts, CSS files, and image files. A message in the center says 'We can then upload the provided files...'. The status bar at the bottom indicates 'Our album.php site on the development server'.

*Image 19: Important php file for the website*

- The **album.php** webpage on the development server will look like this.



*Image 20: photouploader page*

- We may upload images to the website for testing.

Photo	Name	Description	Creation date	Keywords
	Cat	Swinburne Logo	2024-02-28	val1233
	Cat	little cat	2024-03-06	val1233
	Cat	little cat	2024-03-06	val1233
	cgh	cgh	2024-03-06	hvj
	gh	gh	2024-03-07	drg
	Valorant	Valorant Logo	2024-03-07	val1233
	Valorant	little cat	2024-03-07	swinburne logo
	Cat	little cat	2024-03-06	Harvard
	Cat	little cat	2024-03-06	Harvard

*Image 21: photoalbum page*

- After uploading, the image may be seen in both phpMyAdmin and S3. phpMyAdmin keeps the metadata, whereas S3 stores the resized picture. To enhance security, we have created a limitation that only displays photos on the album.php page when viewed using our Load Balancer. As a consequence, while accessing the

album.php page from the server, no photographs will be displayed. However, this behavior will change if we deploy our Web server from the Auto Scaling Group.

The screenshot shows the phpMyAdmin interface for a MySQL database named 'Photos'. The left sidebar shows the database structure with tables like 'information\_schema', 'mysql', 'performance\_schema', 'Photos', and 'sys'. The main area displays the 'photos' table with the following data:

PhotoTitle	Description	CreationDate	Keywords	Reference
Cat	Swinburne Logo	2024-02-28	val1233	https://swinburne2bucket.s3.amazonaws.com/Harvard...
Cat	little cat	2024-03-06	val1233	https://swinburne2bucket.s3.amazonaws.com/Valorant...
Cat	little cat	2024-03-06	val1233	https://swinburne2bucket.s3.amazonaws.com/Valorant...
gh	gh	2024-03-06	hj	https://swinburne2bucket.s3.amazonaws.com/Untitled...
gh	fgh	2024-03-07	drg	https://swinburne2bucket.s3.amazonaws.com/asdadfsf...
Valorant	Valorant Logo	2024-03-07	val1233	https://swinburne2bucket.s3.amazonaws.com/rezzed...
Valorant	little cat	2024-03-07	swinburne logo	https://swinburne2bucket.s3.amazonaws.com/rezzed...
Cat	little cat	2024-03-06	Harvard	https://swinburne2bucket.s3.amazonaws.com/asdadfsf...
Cat	little cat	2024-03-06	Harvard	https://swinburne2bucket.s3.amazonaws.com/asdadfsf...

Image 22: Metadata in phpMyAdmin

The screenshot shows the AWS S3 console for the 'swinburne2bucket' bucket. The left sidebar has sections for Buckets, Access Grants, Access Points, Object Lambda Access Points, Multi-Region Access Points, Batch Operations, IAM Access Analyzer for S3, Block Public Access settings for this account, Storage Lens, Dashboards, Storage Lens groups, AWS Organizations settings, and Feature spotlight. The main area shows a list of objects with the following details:

Name	Type	Last modified	Size	Storage class
asdadfsfhdg.png	png	March 4, 2024, 13:48:18 (UTC+07:00)	140.4 KB	Standard
aws.png	png	March 4, 2024, 13:22:46 (UTC+07:00)	3.5 KB	Standard
dumb cat.png	png	March 4, 2024, 13:24:29 (UTC+07:00)	244.9 KB	Standard
Harvard.png	png	March 4, 2024, 13:29:21 (UTC+07:00)	22.9 KB	Standard
resized-asdadfsfhdg.png	png	March 4, 2024, 13:48:17 (UTC+07:00)	159.7 KB	Standard
resized-aws.png	png	March 4, 2024, 13:22:47 (UTC+07:00)	1.3 KB	Standard
resized-dumb cat.png	png	March 4, 2024, 13:36:16 (UTC+07:00)	70.2 KB	Standard
resized-Harvard.png	png	March 4, 2024, 13:35:16 (UTC+07:00)	8.1 KB	Standard
resized-resized-dumb cat.png	png	March 4, 2024, 13:36:17 (UTC+07:00)	19.5 KB	Standard
resized-resized-Harvard.png	png	March 4, 2024, 13:35:17 (UTC+07:00)	3.1 KB	Standard
resized-Untitled.png	png	March 4, 2024, 13:32:58 (UTC+07:00)	77.8 KB	Standard
Untitled.png	png	March 4, 2024, 22:40:06 (UTC+07:00)	263.1 KB	Standard
Valorant.png	png	March 4, 2024, 13:32:51 (UTC+07:00)	73.9 KB	Standard
Valorant.png	png	March 4, 2024, 13:30:31 (UTC+07:00)	874.2 KB	Standard

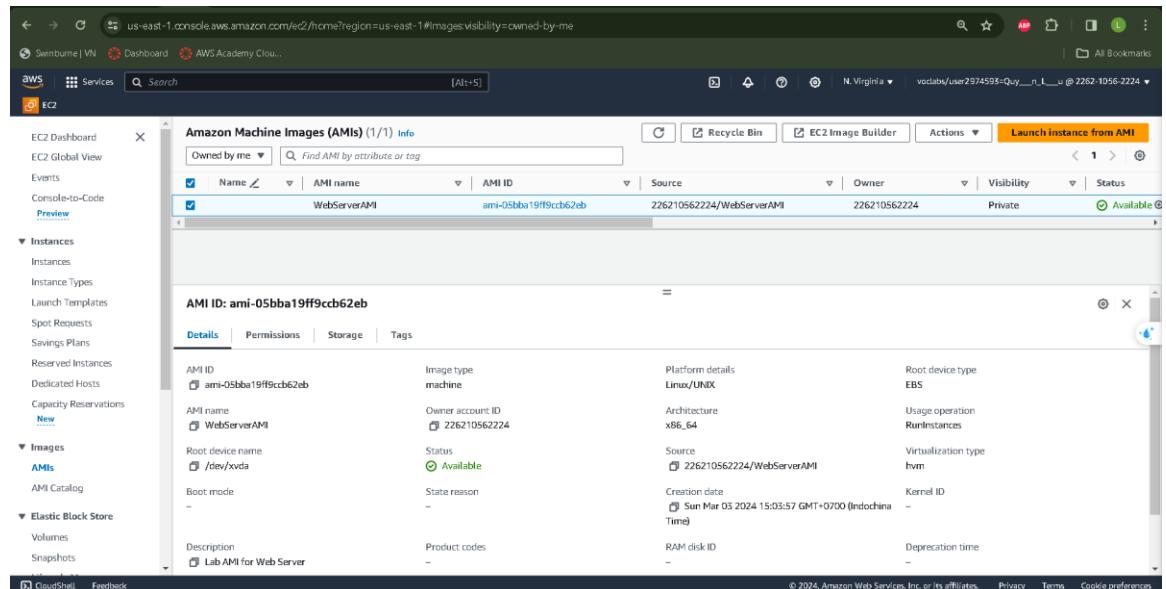
Image 23: All photos and image resizing objects are stored in an S3 bucket.

- With our website completely built and operational, the following step is to generate an Amazon Machine Image (AMI) for the WebDevServer. This AMI will provide a snapshot of our server's present state, including installed software, configurations, and

any changes done.

Creating an AMI allows us to capture the entire server setup, making it easier to replicate and deploy in different environments. By deploying the AMI to our Auto Scaling Group, we ensure that new instances launched by the group will have the same configuration as the WebDevServer, ensuring consistency and scalability.

This process involves creating the AMI from the running WebDevServer instance and then updating the Launch Configuration or Launch Template used by the Auto Scaling Group to use the

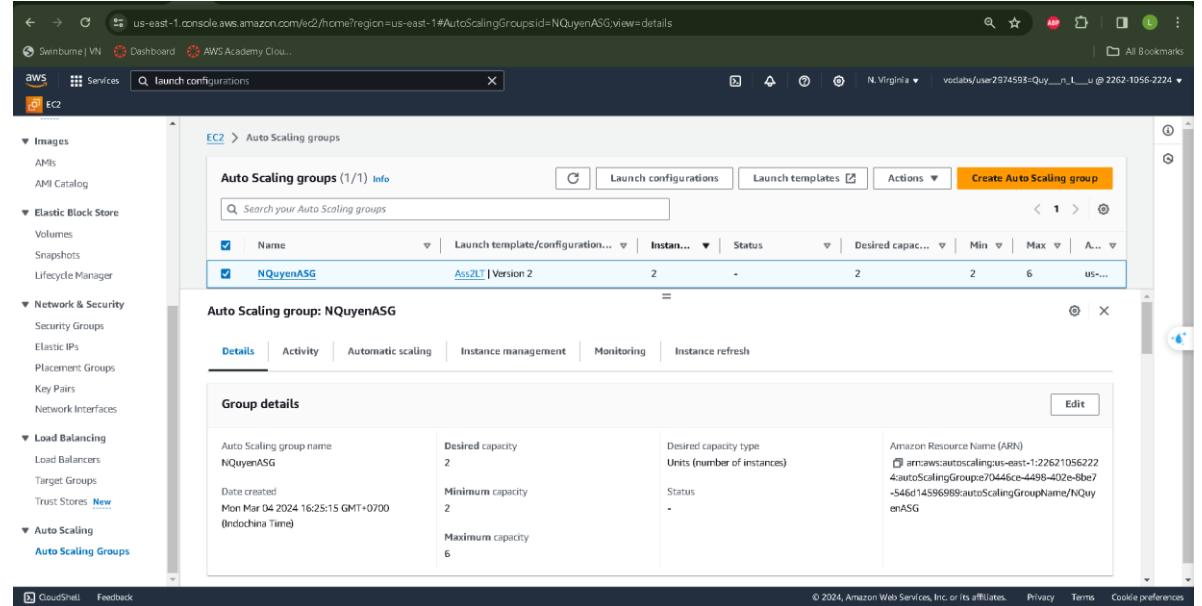


*Image 24: Our WebDevServer's AMI.*

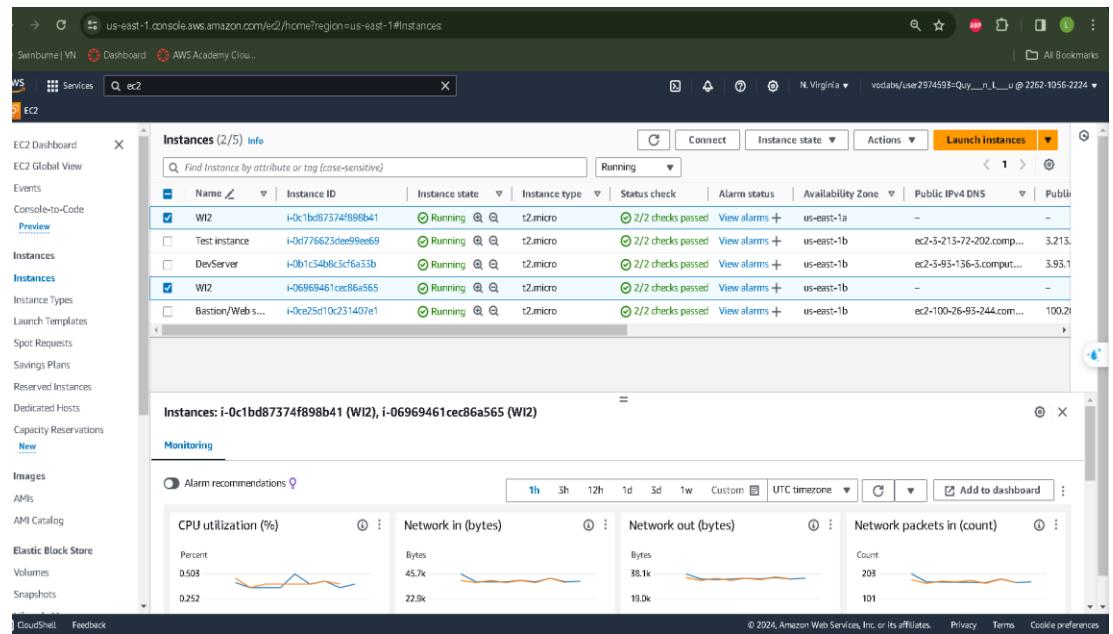
## 5. Load Balancer and Auto Scaling Group

- To improve website availability and scalability after development, we may use Elastic Load Balancer (ELB) and Auto Scaling Group (ASG). Setting up an ASG

allows us to automatically grow the number of EC2 instances based on predetermined circumstances. Integrating it with an ELB allows for effective load balancing and enhances fault tolerance by spreading traffic over numerous instances. This combination ensures that our website can withstand variable traffic levels while maintaining excellent availability.

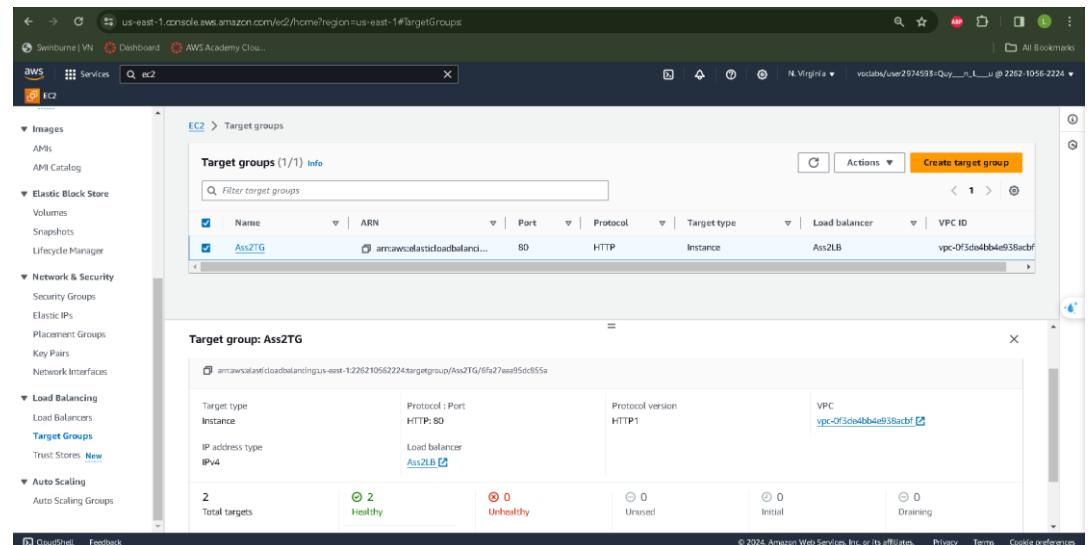


*Image 25: Auto Scaling Group's Detail*



*Image 26: 2 WI2 creating from Auto Scaling Group*

- Launching the load balancer requires a target group.



*Image 27: Loadbalancers attached by Target group*

- Our Load Balancer after attach Target Group will demonstrate like this:

The screenshot shows the AWS CloudFront console with the following details:

- Load balancers (1/1)**: A table listing one load balancer named "Ass2LB".
- Load balancer: Ass2LB** (Details):
  - Type: Application
  - Status: Active
  - VPC: vpc-0f3de4bb4e938acbf
  - Availability Zones: us-east-1a (use1-az2), us-east-1b (use1-az1)
  - IP address type: IPv4
  - Date created: March 4, 2024, 13:45 (UTC+07:00)

Image 28: Ass2LB is creating

The screenshot shows a web browser displaying an album of uploaded photos. The photos are:

Photo	Name	Description	Creation date	Keywords
	Valorant	Valorant Logo	2024-03-05	Valorant, logo, image
	Cat	little cat	2024-03-05	cat, cute, small
	Aws	Aws logo	2024-03-05	Aws logo, cloud computing

Image 29: Album.php may be accessed via the load balancer.

## 6. Security Group and Network ACL

- Once our Web Server Instances are operational, we can focus on ensuring their security and accessibility.

Name	Security group ID	Security group name	VPC ID	Description	Owner
ec2-rds-1	sg-05e52a18fa5750a5	vpc-0f5cdebb4e938acbf		Security group attached to instances to all...	226210562224
ELBSG	sg-0fcdb9ef0291494	vpc-0f5cdebb4e938acbf		ELBSG	226210562224
TestInstanceSG	sg-05862ecf7ecf011765	vpc-0f5cdebb4e938acbf		TestInstanceSG	226210562224
DevServerSG	sg-0cfa047797b173d2c	vpc-0f5cdebb4e938acbf		DevServerSG	226210562224
rds-ec2-1	sg-062d5e0360a6e536fb	vpc-0f5cdebb4e938acbf		Security group attached to dbassignment2...	226210562224
default	sg-00191666393945f1	vpc-0f5cdebb4e938acbf		default VPC security group	226210562224
	sg-0a822942a8539a19	vpc-024a1a0e09909b95ff		default VPC security group	226210562224
DBServerSG	sg-05a95c1415c4f9e1	vpc-0f5cdebb4e938acbf		DBServerSG	226210562224
WebServerSG	sg-04dd0177f760a1334	vpc-0f5cdebb4e938acbf		WebServerSG	226210562224
launch-wizard-1	sg-058674d567641a2d	vpc-024a1a0e09909b95ff		launch-wizard-1 created 2024-01-17T17:4...	226210562224
WebServerSG2	sg-071c8db6797add0988	vpc-0f5cdebb4e938acbf		WebServerSG2	226210562224
DBSG2	sg-03080fd22609e1b5	vpc-0f5cdebb4e938acbf		DBSG2	226210562224

Image 30: Security Groups

- **Web Server Security Group** should only allow access for inbound traffic from Elastic Load Balancer and outbound traffic to the NAT gateway.
  - Inbound and Outbound from All Traffics may accepted by **DevServer Security Group**
  - The **Database Server Security Group** may receive from both inbound and outbound traffic from the Webserver and Devserver.
  - The **Elastic Load Balancer Security Group** controls all inbound and outgoing connections from the Internet gateway.
  - In the final setup step, we will construct an ACL to prevent DevServer from transmitting ICMP packets to the WebServer.
- The configured inbound and outbound rules of our ACL will look like this.

Image 31: Network ACLs

- Using Putty to SSH into our DevServer to ensure created ACL is working as we expected.

Image 32: ICMP testing

- We can ping the NAT gateway at 0.0.0.0 gate, but ICMP messages sent to private subnets 1 (10.0.3.0) and 2 (10.0.4.0) are unreachable. As a result, our Network ACL is operating perfectly.

## 7. Testing

- We will test our website's functionality to verify proper setting. This section will focus on the remaining exams, as many have already been completed.

### Test 1: Termination( Stop) of EC2 and ASG

The screenshot shows the AWS EC2 Instances page. A modal dialog box at the top center says "Successfully stopped i-0b1c34b8c3cf6a33b". Below it, the "Instances (1/5) Info" table lists five instances. The fourth row, labeled "DevServer", has a checkbox next to its name which is checked. The "Actions" column for this row contains a "Stop" button. The rest of the table shows other instances like "Test instance" and "Bastion/Web s...". At the bottom of the page, there is a detailed view for the selected instance "i-0b1c34b8c3cf6a33b (DevServer)" with tabs for Details, Status and alarms, Monitoring, Security, Networking, Storage, and Tags. The "Details" tab is active, showing the Instance ID, Public IPv4 address (3.93.136.3), Private IP4 address (10.0.2.158), and Public IPv4 DNS (ec2-3-93-136-3.compute-1.amazonaws.com).

Image 33: A DevServer instance in the Auto Scaling Group is terminated.

The screenshot shows a web application interface for managing photos. At the top, it displays student information: "Student name: Lau Ngoc Quyen", "Student ID: 104198996", and "Tutorial session: Tuesday 7:AM". Below this, there is a section titled "Uploaded photos:" with a link "Upload more photos". A table lists three uploaded photos:

Photo	Name	Description	Creation date	Keywords
	Valorant	Valorant Logo	2024-03-05	Valorant, logo, image
	Cat	little cat	2024-03-05	cat, cute, small
	Aws	Aws logo	2024-03-05	Aws logo, cloud computing

*Image 34: The Load Balancer DNS remains available even after the Web Instance is terminated and initialized.*

Timestamp	Log Stream	Message
2024-03-05T10:00:00Z	lambda_function/2024-03-05T10:00:00Z	START RequestId: 12345678901234567890123456789012 Version: \$LATEST
2024-03-05T10:00:00Z	lambda_function/2024-03-05T10:00:00Z	END RequestId: 12345678901234567890123456789012 Version: \$LATEST
2024-03-05T10:00:00Z	lambda_function/2024-03-05T10:00:00Z	Function execution succeeded

*Image 35: Two DevServer Instances are healthy*

Name	Subnet ID	CIDR Range	Route Table	Network ACL
PublicSubnet1	subnet-00e4c7b846ce00f8	10.0.0.0/24	rtb-00c73929b7424b44   PublicRT	ad-0351cd0907a672f0   PublicSubnetACL
PublicSubnet2	subnet-0931f0e05d53497b	10.0.1.0/24	rtb-00c73929b7424b44   PublicRT	ad-0351cd0907a672f0   PublicSubnetACL
PrivateSubnet2	subnet-02ce2bdefb23983a2	10.0.3.0/24	rtb-00c73929b7424b44   PrivateRT	ad-0351cd0907a672f0   PrivateSubnetACL
PrivateSubnet1	subnet-05cd2f820bd314cb6	10.0.4.0/24	rtb-00c73929b7424b44   PrivateRT	ad-0351cd0907a672f0   PrivateSubnetACL

*Image 36: Successfully transmit ICMP traffic between the web servers and the development server.*

Amazon S3 > Buckets > swinburne2bucket

**Objects (8) Info**

Name	Type	Last modified	Size	Storage class
aws.png	png	March 5, 2024, 11:21:51 (UTC+07:00)	3.3 KB	Standard
dumb_cat.png	png	March 5, 2024, 11:39:37 (UTC+07:00)	244.9 KB	Standard
resized-aws.png	png	March 5, 2024, 11:21:52 (UTC+07:00)	1.3 KB	Standard
resized-dumb_cat.png	png	March 5, 2024, 11:39:38 (UTC+07:00)	70.2 KB	Standard
resized-Swinburne.png	png	March 5, 2024, 11:39:39 (UTC+07:00)	2.0 KB	Standard
resized-Valorant.png	png	March 5, 2024, 01:54:29 (UTC+07:00)	263.1 KB	Standard
Swinburne.png	png	March 5, 2024, 11:35:09 (UTC+07:00)	5.7 KB	Standard
Valorant.png	png	March 5, 2024, 01:54:25 (UTC+07:00)	874.2 KB	Standard

*Image 37: Check the S3 bucket to ensure photographs are existed and resized perfectly.*

phpMyAdmin

Database: Photos > Table: photos

Showing rows 0 - 7 (total, Query took 0.0011 seconds.)

SELECT \* FROM `photos`

PhotoTitle	Description	CreationDate	Keywords	Reference
Valorant	Valorant Logo	2024-03-05	Valorant, logo, image	<a href="https://swinburne2bucket.s3.amazonaws.com/Valorant.png">https://swinburne2bucket.s3.amazonaws.com/Valorant...</a>
Cat	little cat	2024-03-05	cat, cute, small	<a href="https://swinburne2bucket.s3.amazonaws.com/dumb_cat.png">https://swinburne2bucket.s3.amazonaws.com/dumb_cat...</a>
Aws	Aws logo	2024-03-05	Aws logo, cloud computing	<a href="https://swinburne2bucket.s3.amazonaws.com/aws.png">https://swinburne2bucket.s3.amazonaws.com/aws.png</a>
Aws	Aws logo	2024-03-05	Aws logo, cloud computing	<a href="https://swinburne2bucket.s3.amazonaws.com/aws.png">https://swinburne2bucket.s3.amazonaws.com/aws.png</a>
Quyen	Aws logo	2024-03-05	Aws logo	<a href="https://swinburne2bucket.s3.amazonaws.com/quyen.png">https://swinburne2bucket.s3.amazonaws.com/quyen...</a>
Swinburne	Swinburne University	2024-05-23	Swinburne	<a href="https://swinburne2bucket.s3.amazonaws.com/Swinburne.png">https://swinburne2bucket.s3.amazonaws.com/Swinburn...</a>
Swinburne	Swinburne University	2024-05-23	Swinburne	<a href="https://swinburne2bucket.s3.amazonaws.com/Swinburne.png">https://swinburne2bucket.s3.amazonaws.com/Swinburn...</a>
Stupid cat	little cat	2024-03-05	cat	<a href="https://swinburne2bucket.s3.amazonaws.com/dumb_cat.png">https://swinburne2bucket.s3.amazonaws.com/dumb_cat...</a>

*Image 38: Metadata in phpMyAdmin*

*Link of album.php:*

<http://ass2lb-749465135.us-east-1.elb.amazonaws.com/photoalbum/album.php>

*Link of ELB photouploader.php:*

<http://ass2lb-749465135.us-east-1.elb.amazonaws.com/photoalbum/photouploader.php>

*Link of DevServer Instance:*

<http://ec2-3-93-136-3.compute-1.amazonaws.com/>

-----*The End*-----