

# COS20019 – Assignment 2

## Developing a highly available Photo Album Website

Bui Minh Thuan

104486358

Tutorial Time: Wednesday 10:00AM

Date of Submission: 14/07/2024

### I. INTRODUCTION

This report documents my works for Assignment 2, Unit Cloud Computing Architecture (May 2024). In this assignment, it concentrates on developing a highly available photo album website that allow user upload and view images in their album. The website can be accessed via this link: <http://webserver-elb-784619010.us-east-1.elb.amazonaws.com/photoalbum/album.php>.

### II. INFRASTRUCTURE DEPLOYMENT

In the first part, I start with building and configuring the overall AWS services used in this assignment. Compared to Assignment 1B, the infrastructure for this assignment shares some similarities, but requires some changes to extend functionality and enhance availability for final website.

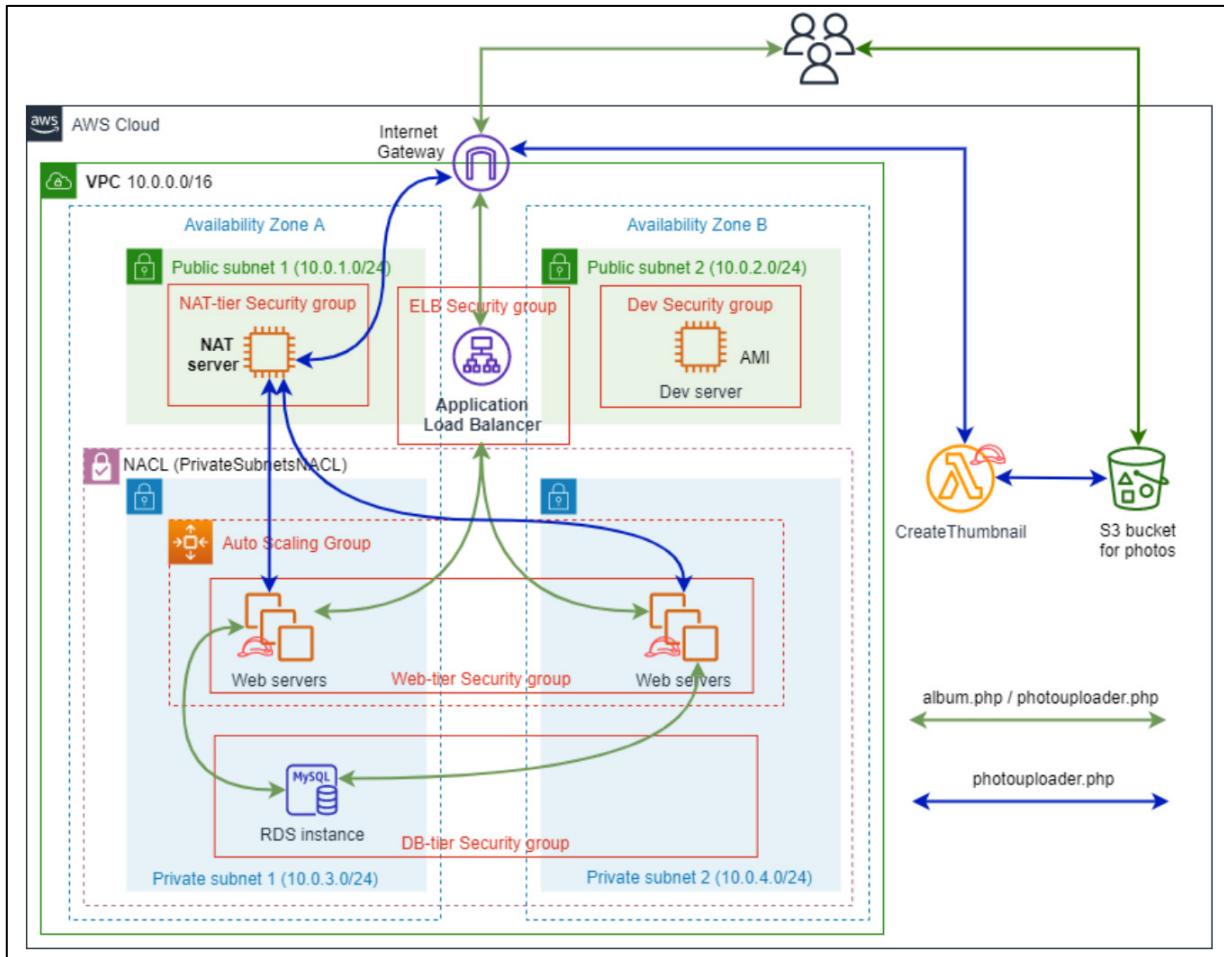


Figure 1: Overall infrastructure

#### A. VPC

For VPC, the configurations stay mostly the same with VPC for assignment 1B. It has “10.0.0.0/16” as CIDR, 2 Availability Zones with 4 subnets ( 1 public and 1 private for each AZ), 2 route tables (1 for public subnets and 1 for private subnets) and an Internet Gateway. However, because web servers in this assignment are deployed in private subnets, I added a NAT gateway which allows instances in private subnets can connect to the internet.

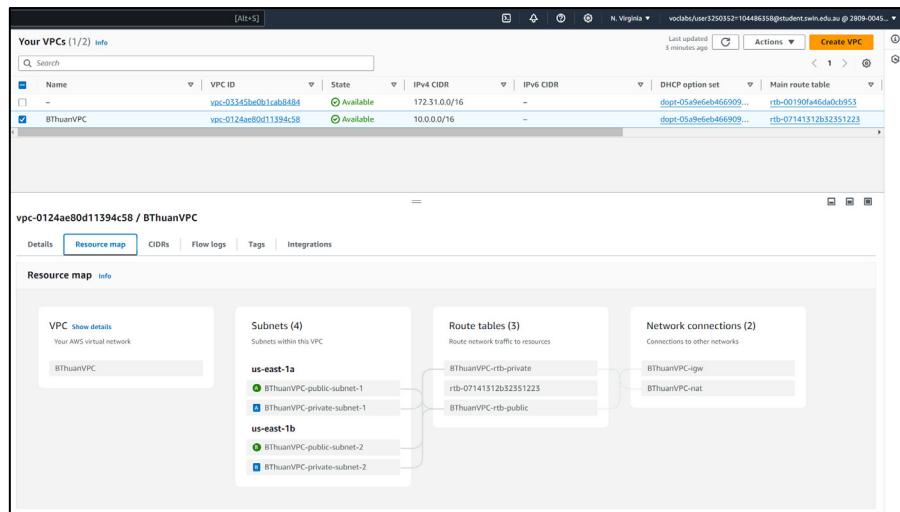


Figure 2: VPC Resources Map

The screenshot shows the Subnets (4) Info page for the BThuanVPC. It lists the following subnets:

Name	Subnet ID	State	VPC	IPv4 CIDR
BThuanVPC-public-subnet-2	subnet-03f555e158cb47cbf	Available	vpc-0124ae80d11394c58   BTh...	10.0.2.0/24
BThuanVPC-public-subnet-1	subnet-0d624181b209cf757	Available	vpc-0124ae80d11394c58   BTh...	10.0.1.0/24
BThuanVPC-private-subnet-1	subnet-0b6e2adb58a82c3b3	Available	vpc-0124ae80d11394c58   BTh...	10.0.3.0/24
BThuanVPC-private-subnet-2	subnet-08602693500b9b4db	Available	vpc-0124ae80d11394c58   BTh...	10.0.4.0/24

Figure 3: Subnets CIDR

The screenshot shows the Route tables (1/1) Info page for the BThuanVPC-rtb-private route table. It displays the following information:

Name	Route table ID	Explicit subnet assoc...	Edge associations	Main
BThuanVPC-rtb-private	rtb-0dc10fd7927c1023f	2 subnets	-	No

Details | Routes | **Subnet associations** | Edge associations | Route propagation | Tags

**Explicit subnet associations (2)**

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
BThuanVPC-private-subnet-1	subnet-0b6e2adb58a82c3b3	10.0.3.0/24	-
BThuanVPC-private-subnet-2	subnet-08602693500b9b4db	10.0.4.0/24	-

Figure 4: Private route tables associate with private subnets

The screenshot shows the Route tables (1/1) Info page for the BThuanVPC-rtb-private route table. It displays the following information:

Name	Route table ID	Explicit subnet assoc...	Edge associations	Main
BThuanVPC-rtb-private	rtb-0dc10fd7927c1023f	2 subnets	-	No

Details | **Routes** | Subnet associations | Edge associations | Route propagation | Tags

**Routes (2)**

Destination	Target	Status	Propagated
0.0.0.0/0	nat-0369f90aab1d77f67	Active	No
10.0.0.0/16	local	Active	No

Figure 5: Private route table routes traffic to NAT

The screenshot shows the AWS Route Tables interface. A public route table named 'BThuanVPC-rtb-public' is selected. It has two explicit subnet associations: 'BThuanVPC-public-subnet-2' (subnet ID: subnet-03f55e158cb47cbf, IPv4 CIDR: 10.0.2.0/24) and 'BThuanVPC-public-subnet-1' (subnet ID: subnet-0d624181b209cf757, IPv4 CIDR: 10.0.1.0/24). The table has no main or edge associations.

Figure 6: Public route tables associate with public subnets

The screenshot shows the AWS Route Tables interface. A public route table named 'BThuanVPC-rtb-public' is selected. It has two routes: one to the target 'igw-0b187da0c3aedc35c' (Status: Active, Propagated: No) and another to the target 'local' (Status: Active, Propagated: No).

Figure 7: Public route table routes traffic to IGW

## B. Security Groups

The next thing I need to configure after setting VPC is Security Groups. Unlike Assignment 1B, when all settings have already provided in the instruction file, in this assignment we need to analyze and decide Security Groups configurations. Below is my settings with explanations for each SG:

Security Group Name	Protocols	Sources
ELBSG	HTTP(80)	Anywhere
WebServerSG	HTTP (80)	ELBSG
	SSH (22)	DevServerSG
DBServerSG	MySQL (3306)	WebServerSG, DevServerSG
DevServerSG	All traffic	Anywhere

Figure 8: Public route table routes traffic to NAT

I) **ELBSG:** The first SG is for ELB (Elastic Load Balancer). Because this ELB will receive web request on behalf of web server, it needs to be able to accept HTTP traffic from anywhere.

The screenshot shows the AWS Security Groups interface. A security group named 'ELBSG' is selected. It has one inbound rule: a TCP port 80 rule from 0.0.0.0/0 to the IP address sgr-01eca5045d9d83... (Protocol: TCP, Port range: 80, IP version: IPv4).

Figure 9: Inbound rules for ELBSG

2) *WebServerSG*: “WebServerSG” is for servers of this website. Here, I allow 2 types of traffic: HTTP from “ELBSG” and SSH from “DevServer”. The first traffic is for web access, while the second one allow I remote access to web server from dev server and test ICMP between dev server and web server.

The screenshot shows the AWS Security Groups console with the search bar set to "WebServerSG". A single security group named "sg-081f0d3a5a4b38ade - WebServerSG" is listed. Under the "Inbound rules" tab, there are two entries:

Security group rule...	IP version	Type	Protocol	Port range	Source
sgr-00a311bb0e7a643...	-	HTTP	TCP	80	sg-06af3d8b2d8500ea...
sgr-0bbcfc1736b11e015	-	SSH	TCP	22	sg-0ee4959cc0dc38a4...

Figure 10: Inbound rules for WebServerSG

3) *DBServerSG*: Database for this website will be in the third Security Group – “DBServerSG”. This SG will accept only MySQL via port 3306, but from 2 sources: “WebServerSG” and “DevServerSG”. First source will be for web server to access its database, while second source allow us remote manage database with phpMyadmin ( see details in RDS section).

The screenshot shows the AWS Security Groups console with the search bar set to "DBServerSG". A single security group named "sg-0cfac8461cc281e57 - DBServerSG" is listed. Under the "Inbound rules" tab, there are two entries:

Security group rule...	IP version	Type	Protocol	Port range	Source
sgr-0e5a00630d156e85f	-	MYSQL/Aurora	TCP	3306	sg-0ee4959cc0dc38a4...
sgr-02754daccf5d59036	-	MYSQL/Aurora	TCP	3306	sg-081f0d3a5a4b38ade...

Figure 11: Inbound rules for DBServerSG

4) *DevServerSG*: Finally, the last needed SG is for Dev Server. As specified in instruction file, this SG doesn't need to follow least-privilege principles, therefore, allowing all traffic from anywhere is enough.

The screenshot shows the AWS Security Groups console with the search bar set to "DevServerSG". A single security group named "sg-0ee4959cc0dc38a44 - DevServerSG" is listed. Under the "Inbound rules" tab, there is one entry:

Security group rule...	IP version	Type	Protocol	Port range	Source
sgr-0e1ca31e37ebdb1fe	IPv4	All traffic	All	All	0.0.0.0/0

Figure 12: Inbound rules for DevServerSG

### C. Network Access Control List (NACL)

Another layer of security for this website is NACL, which filters traffic to subnet. In this scenario, 1 NACL called “PrivateSubnetsNACL” is required for 2 private subnets that contains web servers and databases. This NACL will follow least-privileged principles, block all ICMP traffic from Dev Server to corresponding subnets but still allow the web-related requests so the website runs as expected.

For inbound rules, here is my rules:

1. Rule 100: Deny ICMP from Dev Server, so instances inside NACL can not receive ping from Dev Server instance
2. Rule 110: Allow MySQL from DevServerSG (10.0.2.0/24)
3. Rule 120, 130: Allow HTTP traffic from ELB, which can be in public subnet 1 (10.0.1.0/24) or public subnet 2 (10.0.2.0/24)
4. Rule 140: Allow SSH from Dev Server, so we can remote access from Dev Server and test ICMP ping
5. Rule 150: Allow MySQL from 10.0.4.0/24 because the RDS is deployed in 10.0.3.0/24, while ASG might create instance in 10.0.4.0/24, so we need to enable this traffic
6. Rule 160: Allow response from Lambda function. Even though Lambda doesn't have an associated IP but its response still uses ephemeral ports 1024 – 65535

Rule number	Type	Protocol	Port range	Source	Allow/Deny
100	All ICMP - IPv4	ICMP (1)	All	10.0.2.0/24	Deny
110	MySQL/Aurora (3306)	TCP (6)	3306	10.0.2.0/24	Allow
120	HTTP (80)	TCP (6)	80	10.0.1.0/24	Allow
130	HTTP (80)	TCP (6)	80	10.0.2.0/24	Allow
140	SSH (22)	TCP (6)	22	10.0.2.0/24	Allow
150	MySQL/Aurora (3306)	TCP (6)	3306	10.0.4.0/24	Allow
160	Custom TCP	TCP (6)	1024 - 65535	0.0.0.0/0	Allow
	All traffic	All	All	0.0.0.0/0	Deny

Figure 13: Inbound rules for “PrivateSubnetsNACL”

For outbound rules, here is my rules:

1. Rule 100: Deny ICMP to Dev Server, so instances inside NACL cannot send ping from Dev Server instance
2. Rule 110, 120: Enable TCP in ephemeral ports to 10.0.1.0/24 or 10.0.2.0/24 so web server can reply to web request from ELB
3. Rule 130: Allow send MySQL request to database from web server in 10.0.4.0/24 because sometimes a webserver can be deployed in subnet 10.0.4.0/24
4. Rule 140: Allow sending reply from database in 10.0.3.0/24 (port 3306) to web server in 10.0.4.0/24 (port 1024 – 65535), because webserver can be deployed in subnet 10.0.4.0/24 by ASG
5. Rule 150: Allow HTTPS, so the web server can upload image to S3 bucket and invoke Lambda function.

Rule number	Type	Protocol	Port range	Destination	Allow/Deny
100	All ICMP - IPv4	ICMP (1)	All	10.0.2.0/24	Deny
110	Custom TCP	TCP (6)	1024 - 65535	10.0.1.0/24	Allow
120	Custom TCP	TCP (6)	1024 - 65535	10.0.2.0/24	Allow
130	MySQL/Aurora (3306)	TCP (6)	3306	10.0.3.0/24	Allow
140	Custom TCP	TCP (6)	1024 - 65535	10.0.4.0/24	Allow
150	HTTPS (443)	TCP (6)	443	0.0.0.0/0	Allow
	All traffic	All	All	0.0.0.0/0	Deny

Figure 14: Outbound rules for “PrivateSubnetsNACL”

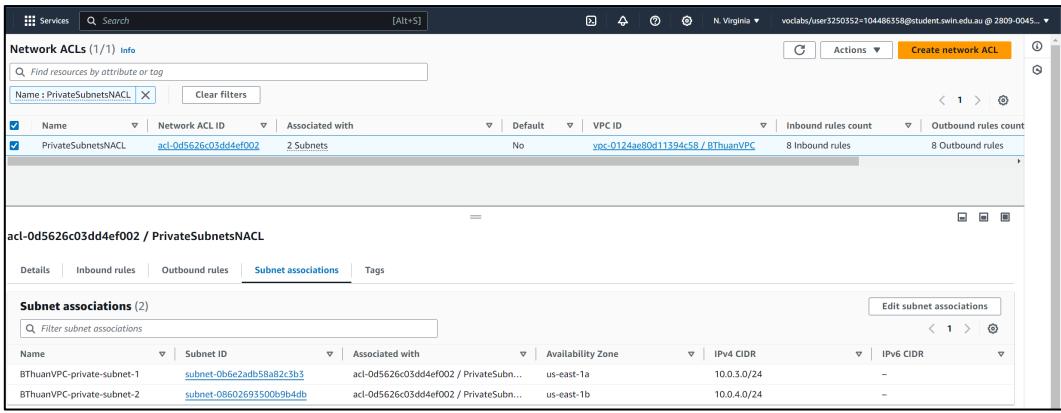


Figure 15: “PrivateSubnetsNACL” is associated with 2 private subnets

These rules ensure the web still function as expected, and block ICMP from/to Dev Server.

```
ec2-user@ip-10-0-4-217:~$ login as: ec2-user
Authenticating with public key "asm2"
Last login: Fri Jul 12 16:06:17 2024 from 118.71.222.82
  _\ _###_ Amazon Linux 2
  ~\_\###\ AL2 End of Life is 2025-06-30.
  ~~ \###!
  ~~ \#/
  ~~ V~' '-->
  ~~~ / A newer version of Amazon Linux is available!
  ~~~ . / Amazon Linux 2023, GA and supported until 2028-03-15.
  ~~~ / / https://aws.amazon.com/linux/amazon-linux-2023/
  _/m,'

[ec2-user@ip-10-0-2-214 ~]$ ping 10.0.4.217
PING 10.0.4.217 (10.0.4.217) 56(84) bytes of data.
^C
--- 10.0.4.217 ping statistics ---
9 packets transmitted, 0 received, 100% packet loss, time 8189ms

[ec2-user@ip-10-0-2-214 ~]$ ssh ec2-user@10.0.4.217
The authenticity of host '10.0.4.217 (10.0.4.217)' can't be established.
ECDSA key fingerprint is SHA256:mR/8+I3KNRseuYaiqRX74A613od90UGQTHSpMwAckb0.
ECDSA key fingerprint is MD5:ee:3:b7:f7:96:dd:fe:0b:a1:fb:70:22:0e:22:dc:c0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.4.217' (ECDSA) to the list of known hosts.
Last login: Fri Jul 12 15:15:55 2024 from ec2-18-206-107-29.compute-1.amazonaws.com
  _\ _###_ Amazon Linux 2
  ~\_\###\ AL2 End of Life is 2025-06-30.
  ~~ \###!
  ~~ \#/
  ~~ V~' '-->
  ~~~ / A newer version of Amazon Linux is available!
  ~~~ . / Amazon Linux 2023, GA and supported until 2028-03-15.
  ~~~ / / https://aws.amazon.com/linux/amazon-linux-2023/
  _/m,'

[ec2-user@ip-10-0-4-217 ~]$ ping 10.0.2.214
PING 10.0.2.214 (10.0.2.214) 56(84) bytes of data.
^C
--- 10.0.2.214 ping statistics ---
9 packets transmitted, 0 received, 100% packet loss, time 8178ms
```

Figure 16: Cannot perform ping between Web Server and Dev Server

#### D. IAM Roles

In this assignment, we need EC2, Lambda and S3 being able to interact with each other, but still ensure security because we don't want all services in our account are publicly used. That's why we need IAM roles – which will restrict access of resources to only what we want and needed for the web to function.

Because this assignment is done in Learner Lab, we don't have any permissions to create new role, so I need to use a provided role called “LabRole”. This role has all needed and no additional policy to resources in our account.

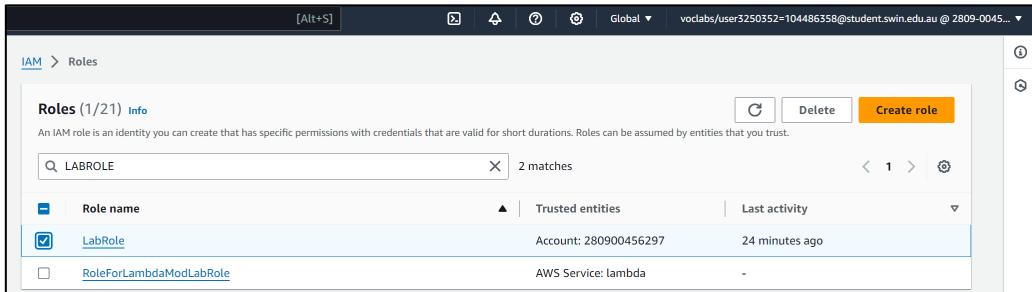


Figure 17: “LabRole” already created in Learner Lab account

This role will be used when I create the instance profile for the launch template, so it will allow the Web Server perform photo uploading with AWS PHP SDK, and when I set up execution role when create the Lambda function “CreateThumbnail”. Details about these 2 usages will be discussed in sections about “Launch Template” and “Lambda”.

#### E. S3 bucket

Same with Assignment 1B, photo uploaded are stored in an S3 bucket. However, this assignment required stricter access policy: the image should not be accessed publicly via internet, instead, it should be accessed by my website only.

Name	Type	Last modified	Size	Storage class
chelsea.png	png	July 14, 2024, 16:08:44 (UTC+07:00)	108.8 KB	Standard
cole.png	png	July 14, 2024, 16:38:10 (UTC+07:00)	42.3 KB	Standard
images.jpg	jpg	July 14, 2024, 21:58:18 (UTC+07:00)	101.4 KB	Standard
resized-chelsea.png	png	July 14, 2024, 16:08:45 (UTC+07:00)	28.7 KB	Standard
resized-cole.png	png	July 14, 2024, 16:38:12 (UTC+07:00)	12.5 KB	Standard

Figure 18: All images are stored in an S3 bucket

Because the image is only accessible through my website, I set up a bucket policy that restricts access to the bucket from a specific HTTP referrer [1]. In this case, only request from my website with DNS “webserver-elb-784619010.us-east-1.elb.amazonaws.com” or “www.webserver-elb-784619010.us-east-1.elb.amazonaws.com” is allowed to access my bucket.

```
{
  "Version": "2012-10-17",
  "Id": "S3 policy",
  "Statement": [
    {
      "Sid": "Only allow request from website",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::asm2-bucket-may2024/*",
      "Condition": {
        "StringLike": {
          "aws:Referer": [
            "http://webserver-elb-784619010.us-east-1.elb.amazonaws.com/*",
            "http://www.webserver-elb-784619010.us-east-1.elb.amazonaws.com/*"
          ]
        }
      }
    }
  ]
}
```

Figure 19: Updated bucket policy

```
<Error>
<Code>AccessDenied</Code>
<Message>Access Denied</Message>
<RequestId>DHQ4S0GJH16CJV1J</RequestId>
<HostId>OKUR0BVupTZ73VY64SdwacjR2GrDR1TtYCbrgH3rHm8j410z60SpFwXy4Y7T9weib/dUI7OU2zU=</HostId>
</Error>
```

Figure 20: Image is not accessible publicly

#### F. RDS

Similar to Assignment 1B, in this assignment, we also need an RDS database to store meta-data for uploaded images. The settings for this database stays exactly the same with Assignment 1B, so there is no for changing in configurations. The difference is now instead of access to this database from web server, we need to access it from Dev Server. Therefore, I need to install phpMyAdmin in Dev Server and configure DBServerSG to accept MySQL from DevServerSG.

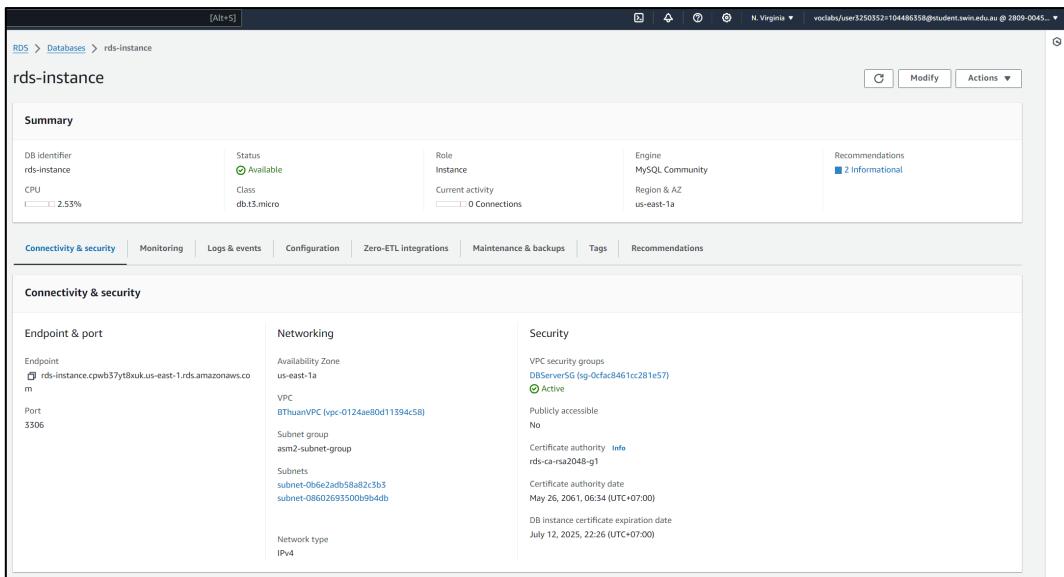


Figure 21: Database configurations similar to assignment 1B: MySQL, locates in “BThuanVPC” and associate with “DBServerSG”

photo_title	description	creation	keywords	reference
Chelsea	My image	2024-07-12	test, football team	<a href="https://asm2-bucket-may2024.s3.amazonaws.com/chels...">https://asm2-bucket-may2024.s3.amazonaws.com/chels...</a>
Chelsea > Barca	doibongtioyeu	2024-07-12	rich, wealthy	<a href="https://asm2-bucket-may2024.s3.amazonaws.com/th.pn...">https://asm2-bucket-may2024.s3.amazonaws.com/th.pn...</a>
Mountain	My image	2024-07-12	mount, beauty	<a href="https://asm2-bucket-may2024.s3.amazonaws.com/chels...">https://asm2-bucket-may2024.s3.amazonaws.com/chels...</a>
Kazim	My teacher smile	2024-07-12	smile, happy	<a href="https://asm2-bucket-may2024.s3.amazonaws.com/Scree...">https://asm2-bucket-may2024.s3.amazonaws.com/Scree...</a>
Mind Map	My map	2024-07-12	proposal, mind map	<a href="https://asm2-bucket-may2024.s3.amazonaws.com/Scree...">https://asm2-bucket-may2024.s3.amazonaws.com/Scree...</a>
Test 1	test image	2024-07-12	hello, test	<a href="https://asm2-bucket-may2024.s3.amazonaws.com/Scree...">https://asm2-bucket-may2024.s3.amazonaws.com/Scree...</a>
Test 1	test image	2024-07-12	hello, test	<a href="https://asm2-bucket-may2024.s3.amazonaws.com/Scree...">https://asm2-bucket-may2024.s3.amazonaws.com/Scree...</a>
picture	test icmp	2024-07-13	aws, putty	<a href="https://asm2-bucket-may2024.s3.amazonaws.com/Scree...">https://asm2-bucket-may2024.s3.amazonaws.com/Scree...</a>
Ronaldo	Brother 7 hardword	2024-07-14	goat, euro, portugal	<a href="https://asm2-bucket-may2024.s3.amazonaws.com/ronal...">https://asm2-bucket-may2024.s3.amazonaws.com/ronal...</a>
Chelsea	test image	2024-07-14	mount, beauty	<a href="https://asm2-bucket-may2024.s3.amazonaws.com/chels...">https://asm2-bucket-may2024.s3.amazonaws.com/chels...</a>
Chelsea	test image	2024-07-14	mount, beauty	<a href="https://asm2-bucket-may2024.s3.amazonaws.com/chels...">https://asm2-bucket-may2024.s3.amazonaws.com/chels...</a>
Messi	Second goat	2024-07-14	test, football team	<a href="https://asm2-bucket-may2024.s3.amazonaws.com/ronal...">https://asm2-bucket-may2024.s3.amazonaws.com/ronal...</a>
Palmer	Cole Winger	2024-07-14	chelsea, england	<a href="https://asm2-bucket-may2024.s3.amazonaws.com/cole...">https://asm2-bucket-may2024.s3.amazonaws.com/cole...</a>
Lucky money	A tradition on Tet holiday	2015-01-01	lucky, money, tradition, Tet	<a href="https://asm2-bucket-may2024.s3.amazonaws.com/image...">https://asm2-bucket-may2024.s3.amazonaws.com/image...</a>

Figure 22: Meta-data are recorded correctly in table “photos”, similar to Assignment 1B

#### G. Lambda function

In this assignment, it is required to implement a serverless function (Lambda) to resize the newly-uploaded image. Whenever a new image is uploaded, this function will be invoked to resize this image and upload the resized version to the same S3 bucket. As the code for this function is already written in the instruction folder, what I need to do is just upload it to Lambda and configure function settings so that it works properly. This function is called “CreateThumbnail”, run in “Python 3.11”, “arm64” as architecture and has “LabRole” as execution role (see IAM Roles section for details of this role)

```

1 import os
2 import boto3
3 import json
4 import uuid
5 import base64
6 import io
7 from PIL import Image
8 from PIL.ExifTags import getexif
9 ...
10 # To build an AWS Lambda deployment package, see:
11 # https://docs.aws.amazon.com/lambda/latest/dg/build-with-v3-tutorial.html#build-with-v3-tutorial-create-function-package
12 # https://docs.aws.amazon.com/lambda/latest/dg/with-v3-tutorial.html#with-v3-tutorial-create-function-package
13 # ONLY PNG FILES ARE SUPPORTED
14 # This Lambda Function shrinksizes your photo.png from your photo-bucket-name S3 bucket, resizes the pic, then upload the resized pic (resized-your-photo.png) to the same bucket
15 ...
16 ...
17 ...
18 ...
19 ...
20 ...
21 ...
22 def resize_image(image_path, resized_path):
23     with Image.open(image_path) as image:
24         image.thumbnail([x / 2 for x in image.size])
25         image.save(resized_path)
26
27 def lambda_handler(event, context):
28     try:
29         bucket_name = event['bucketName']
30         file_name = event['filename']
31         key = event['key']
32         file_size = event['size']
33         file_type = event['type']
34         file_ext = file_type.split('.')[-1]
35         tmpkey = key.replace(file_ext, '_resized.' + file_ext)
36         uploaded_path = '/tmp/' + tmpkey
37         resized_image_downloaded_path = '/tmp/' + file_name
38         resized_image_downloaded_name = file_name + '_resized.' + file_ext
39         resized_image_downloaded_size = file_size
40         resized_image_downloaded_type = file_type
41         resized_image_downloaded_ext = file_ext
42
43         client = boto3.client('s3')
44
45         client.download_file(bucket_name, key, uploaded_path)
46         client.upload_file(uploaded_path, bucket_name, tmpkey)
47         client.upload_file(resized_image_downloaded_path, bucket_name, resized_image_downloaded_name)
48
49     except ClientError as e:
50         print(e)
51         return {"LambdaError": "Error: " + str(e)}
52
53     return {"LambdaError": "Success: " + str(e)}
54
55

```

Figure 23: Lambda code and runtime settings

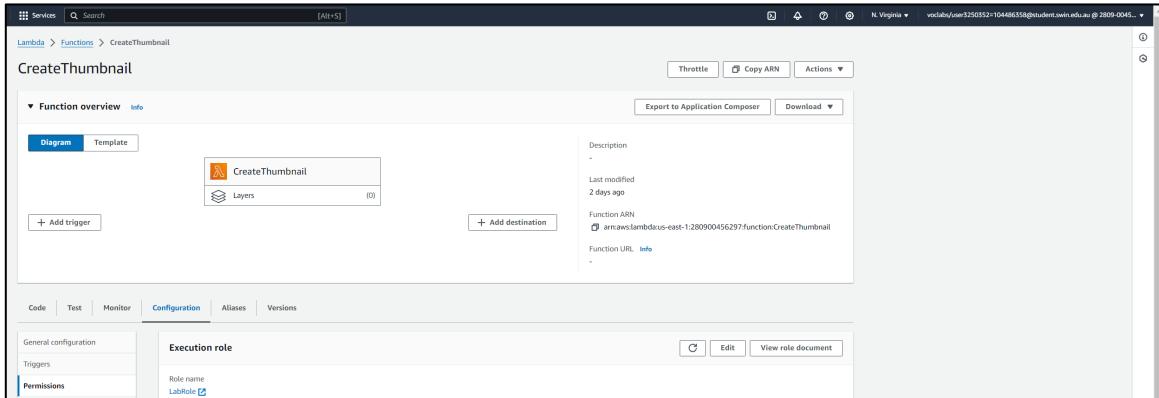


Figure 24: “LabRole” as execution role

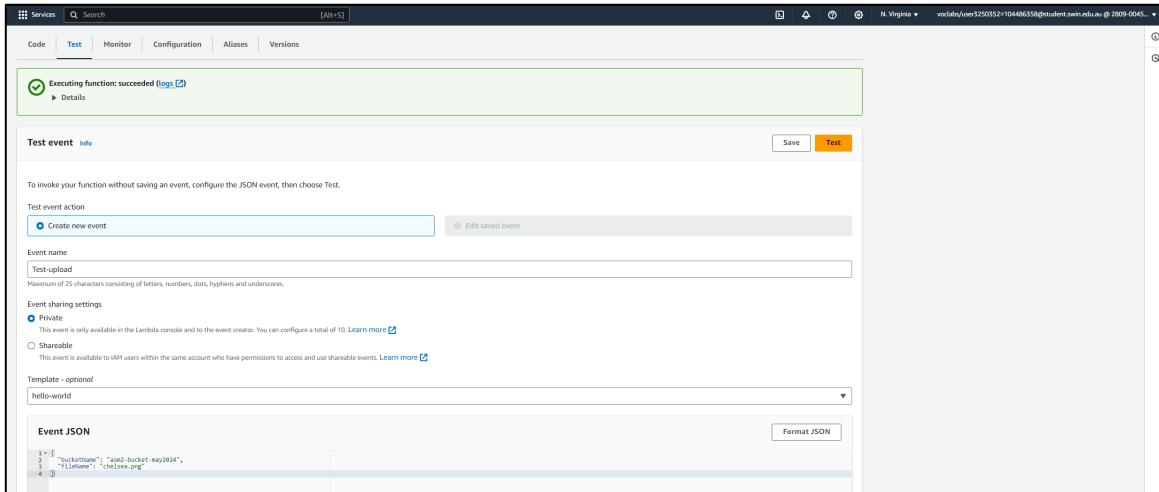


Figure 25: Function runs successfully when testing

#### H. Elastic Load Balancer (ELB)

To guarantee the high availability of this website, an Application Load Balancer is required to forward traffic to different EC2 instances. This ELB requires a target group contains a list of instances to distribute traffic into and perform health check.

Figure 26: Target Group with health check protocol and path

With this target group, I define listener rule for the ELB. It will listen and perform health check to instances in this target group in port 80 (HTTP).

Figure 27: Load Balancer with correct rules and listener

The final ELB will be associated with an Elastic Public IP address and ELB SG as Security Group.

Figure 28: Load Balancer associate with elastic network interface

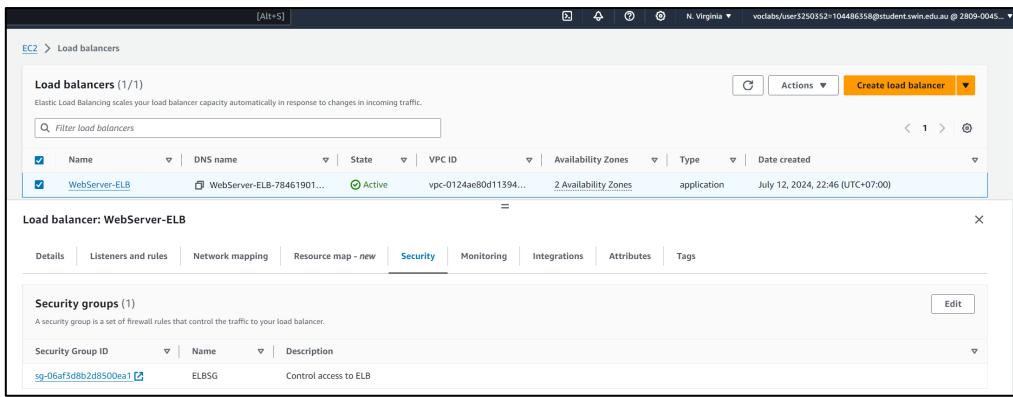


Figure 29: ELB is protected by ELBSG

### I. Auto Scaling Groups (ASG)

After setting up the ELB, I will add an Auto Scaling Group to my infrastructure. The idea of using this ASG is to automatically deploy enough instances for the web server, so that my website is always available. In order to function properly, an ASG needs a Load Balancer (which is already set up) and a Launch Template.

Launch Template is the template containing configurations and settings, which the ASG will refer to when it sets up new instances for my web servers. The Launch Template is created from a custom AMI, or instance image. Just imagine it as an image that captures all settings, software,.. of an instance. This means, the first thing we need to set up is an EC2 instance as sample instance, then create an AMI from it, which later will be used to set up the Launch Template. In this scenario, this EC2 instance is the Dev Server.

First, I deploy the Dev Server in my VPC, in subnet public 2. This instance has similar configuration for a web server instance. I will also install needed software for this website such as Apache, AWS PHP SDK, phpMyAdmin,... into it.

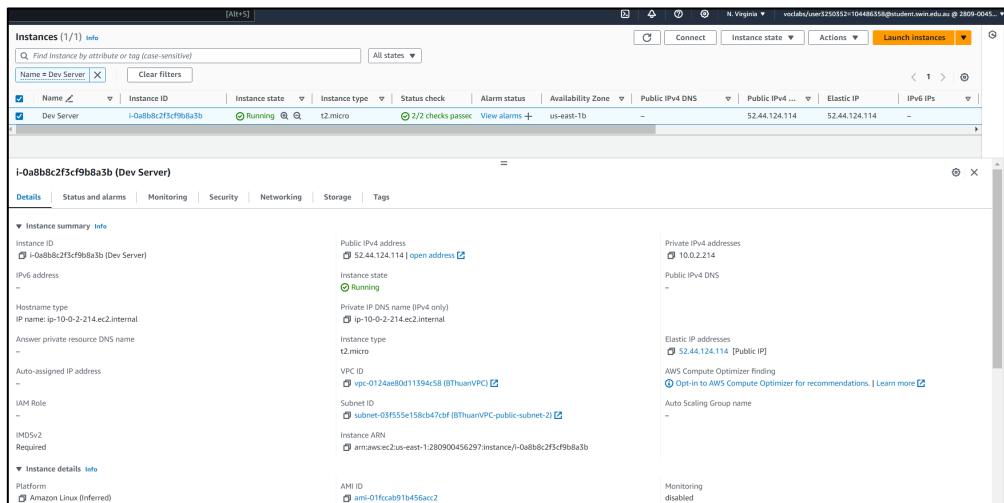


Figure 30: Instance configurations

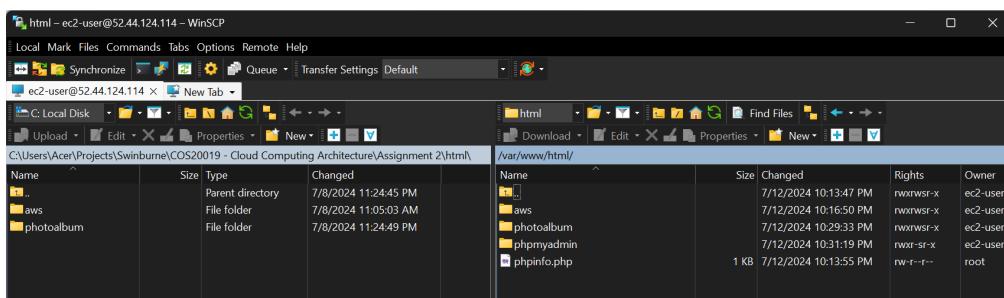


Figure 31: Need software installed on Dev Server

From this instance, I create an AMI, then used this AMI to set up the Launch Template for Web Servers. One thing to remember is setting instance role as "LabRole", so it can function as expected.

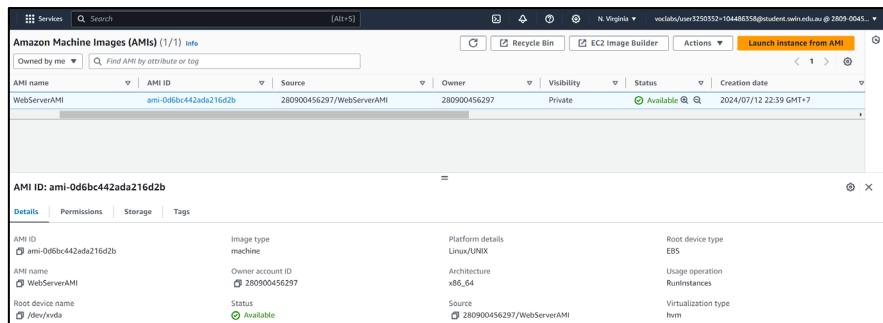


Figure 32: WebServerAMI is created from Dev Server

**Application and OS Images (Amazon Machine Image) Info**

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Recents | My AMIs | Quick Start

Don't include in launch template  Owned by me  Shared with me

Browse more AMIs  
Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

WebServerAMI  
ami-0d6bc442ada216d2b  
2024-07-12T15:39:44.000Z Virtualization: hvm ENA enabled: true Root device type: ebs

Figure 33: Specify WebServerAMI as AMI when create Launch Template

IAM instance profile | Info

LabInstanceProfile  
arn:aws:iam::280900456297:instance-profile/LabInstanceProfile

Figure 34: Specify role for this Launch Template as “LabRole”

After setting up the Launch Template, we can use this to create the Auto Scaling Group. This Auto Scaling Group will take the newly-created Launch Template as template to create new instances when any instances become unhealthy. It ensures that there are always at least 2 and maximum 3 instances running at the same time, so that the ELB can distribute traffic properly.

Instance type requirements Info

Launch template: WebServer-LT

Instance type: t2.micro

Network Info

VPC: vpc-0124ae0bd11394c58 (BThuanVPC)

Availability Zones and subnets

us-east-1a | subnet-066e2ad5b5882c3b3 (BThuanVPC-private-subnet-1) 10.0.3.0/24

us-east-1b | subnet-08602693500b04db (BThuanVPC-private-subnet-2) 10.0.4.0/24

Create a subnet

Figure 35: Deploy ASG to private subnets in my VPC

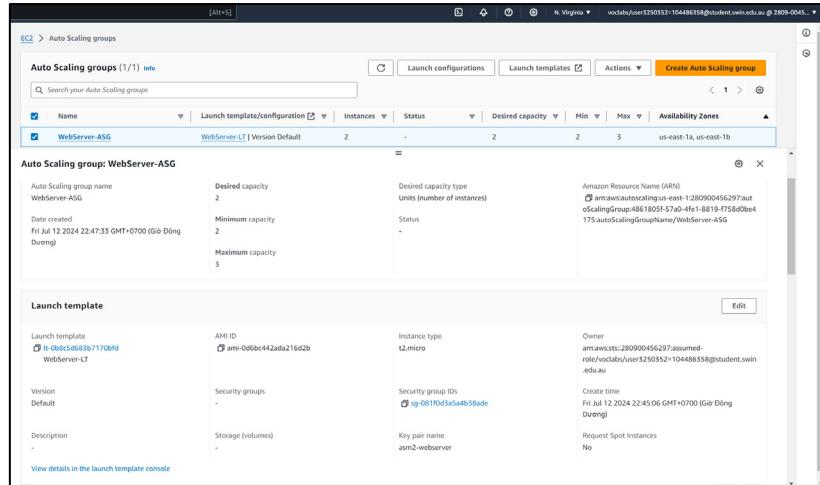


Figure 36: WebServer-ASG is created

Another thing we need to concern about is target tracking policy. In this assignment, it is required to track the request count part target to 30 for this Auto Scaling Group. I achieve that by configuring the target tracking policy when setting up the ASG.

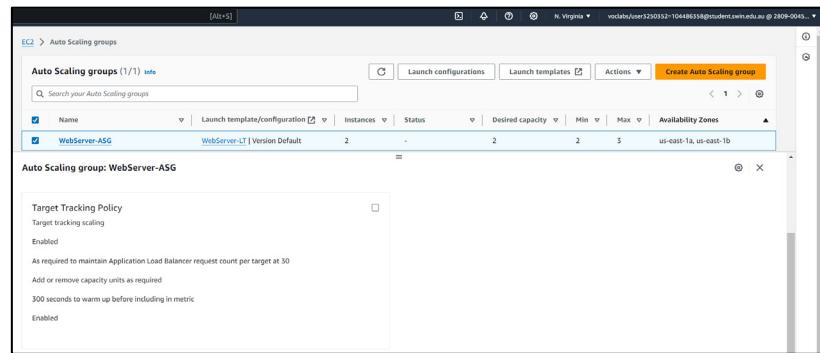


Figure 37: Target tracking policy is configured

After the ASG is set up successfully, the main web servers used for this website will be automatically deployed. These servers have the correct settings created based on the Launch Template, having all needed software and functionalities with “LabRole” as instance role. It is ready to serve the Photo Album users right after launched without any further configurations.

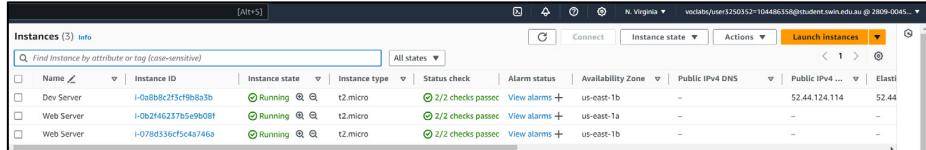


Figure 38: 2 Web Server instances are deployed

### III. FUNCTIONAL REQUIREMENTS

Based on the infrastructure deployed above, the website is set up. The code file has already provided in the assignment instruction file, so all I need to do is updating the constants in “constants.php” to match with the newly-configured infrastructure.

```
// [ACTION REQUIRED] your Auto Scaling group name
define('STUDENT_NAME', 'Bùi Minh Thuận');
// [ACTION REQUIRED] your student ID
define('STUDENT_ID', '104486358');
// [ACTION REQUIRED] your tutorial session
define('TUTORIAL_SESSION', 'Wednesday 10:00AM');

// [ACTION REQUIRED] name of the S3 bucket that stores images
define('BUCKET_NAME', 'am2-bucket');
// [ACTION REQUIRED] region of the above bucket
define('REGION', 'us-east-1');
define('DB_BUCKET_URL', 'https://'.BUCKET_NAME.'.s3.amazonaws.com/');

// [ACTION REQUIRED] name of the database that stores photo meta-data (note that this is not the DB identifier of the RDS instance)
define('DB_NAME', 'cos2019-may-2024-asg');
// [ACTION REQUIRED] endpoint of RDS instance
define('DB_ENDPOINT', 'WebServer-ASG-t1bku.us-east-1.rds.amazonaws.com');
// [ACTION REQUIRED] username of your RDS instance
define('DB_USERNAME', 'main');
// [ACTION REQUIRED] password of your RDS instance
define('DB_PWD', 'buiminhtuan12345');

// [ACTION REQUIRED] name of the DB table that stores photo's meta-data
define('DB_PHOTO_TABLE_NAME', 'photos');
// The table above has 5 columns:
// [ACTION REQUIRED] name of the column in the above table that stores photo's titles
define('DB_PHOTO_TITLE_COL_NAME', 'photo_title');
// [ACTION REQUIRED] name of the column in the above table that stores photo's descriptions
define('DB_PHOTO_DESCRIPTION_COL_NAME', 'description');
// [ACTION REQUIRED] name of the column in the above table that stores photo's creation dates
define('DB_PHOTO_CREATE_DATE_COL_NAME', 'creation');
// [ACTION REQUIRED] name of the column in the above table that stores photo's keywords
define('DB_PHOTO_KEYWORDS_COL_NAME', 'keywords');
// [ACTION REQUIRED] name of the column in the above table that stores photo's links in S3
define('DB_PHOTO_SREFERENCE_COL_NAME', 'reference');

// [ACTION REQUIRED] name (ARN can also be used) of the Lambda function that is used to create thumbnails
define('LAMBDA_FUNC_THUMBNAILS_NAME', 'CreateThumbnail');
```

Figure 39: Updated constants for website

#### A. Website accessible via ELB and photos, metadata are displayed correctly.

My website is now accessible via ELB's DNS with the link: <http://webserver-elb-784619010.us-east-1.elb.amazonaws.com/photoalbum/album.php>. Because ELB has its own Elastic IP, the DNS stays the same, so this link is always consistent. With accurate ASG settings, the website is always available and shows consistent content every time we use it. In my website, we can see all uploaded photos and its meta-data displayed in a table.

Photo	Name	Description	Creation date	Keywords
	Chelsea	My image	2024-07-12	test, football team
	Chelsea > Barca	doibongtoiyen	2024-07-12	rich, wealthy
	Mountain	My image	2024-07-12	mount, beauty

Figure 40: Website accessible with correct information displayed

#### B. Photo are uploaded properly to S3 and RDS, and also resized by RDS

A significant features of this website is being able to upload images user want to its database. The images will be stored in S3 bucket, its meta-date, include image link are stored in RDS, while resized version of this photo created by Lambda will be in the same S3 bucket. Details about S3, RDS, and Lambda have already mentioned in above sections.

Photo title:

Select a photo (Select PNG file for best result):  griziz.png

Description:

Date:

Keywords (comma-delimited, e.g. keyword1, keyword2, ...):

[Photo Album](#)

Figure 41: Upload images with a form in “photouploader.php”

	picture	test icmp	2024-07-13	aws, putty
	Ronaldo	Brother 7 hardword	2024-07-14	goat, euro, portugal
	Chelsea	test image	2024-07-14	mount, beauty
	Chelsea	test image	2024-07-14	mount, beauty
	Messi	Second goat	2024-07-14	test, football team
	Palmer	Cole Winger	2024-07-14	chelsea, england
	Lucky money	A tradition on Tet holiday	2015-01-01	lucky, money, tradition, Tet
	Griezmann	My favorite football player	2024-07-14	france, madrid

Figure 42: Newly added image is displayed in the bottom

photo_title	description	creation	keywords	reference
Chelsea	My image	2024-07-12	test, football team	https://asm2-bucket-may2024.s3.amazonaws.com/chels...
Chelsea > Barca	doibongtroyeu	2024-07-12	rich, wealthy	https://asm2-bucket-may2024.s3.amazonaws.com/fh.pn...
Mountain	My image	2024-07-12	mount, beauty	https://asm2-bucket-may2024.s3.amazonaws.com/chels...
Kazim	My teacher smile	2024-07-12	smile, happy	https://asm2-bucket-may2024.s3.amazonaws.com/Scree...
Mind Map	My map	2024-07-12	proposal, mind map	https://asm2-bucket-may2024.s3.amazonaws.com/Scree...
Test 1	test image	2024-07-12	hello, test	https://asm2-bucket-may2024.s3.amazonaws.com/Scree...
Test 1	test image	2024-07-12	hello, test	https://asm2-bucket-may2024.s3.amazonaws.com/Scree...
picture	test kmp	2024-07-13	aws, putty	https://asm2-bucket-may2024.s3.amazonaws.com/Scree...
Ronaldo	Brother 7 hardword	2024-07-14	goat, euro, portugal	https://asm2-bucket-may2024.s3.amazonaws.com/ronal...
Chelsea	test image	2024-07-14	mount, beauty	https://asm2-bucket-may2024.s3.amazonaws.com/chels...
Chelesa	test image	2024-07-14	mount, beauty	https://asm2-bucket-may2024.s3.amazonaws.com/chels...
Messi	Second goat	2024-07-14	test, football team	https://asm2-bucket-may2024.s3.amazonaws.com/ronal...
Palmer	Cole Winger	2024-07-14	chelsea, england	https://asm2-bucket-may2024.s3.amazonaws.com/cole...
Lucky money	A tradition on Tet holiday	2015-01-01	lucky, money, tradition, Tet	https://asm2-bucket-may2024.s3.amazonaws.com/image...
Griezmann	My favorite football player	2024-07-14	france, madrid	https://asm2-bucket-may2024.s3.amazonaws.com/grizi...

Figure 43: Meta-data is recorded in RDS database

Objects (20) <a href="#">Info</a>						
<a href="#">Actions</a> <a href="#">Create folder</a> <a href="#">Upload</a>						
<input type="text" value="Find objects by prefix"/> <a href="#">Copy S3 URI</a> <a href="#">Copy URL</a> <a href="#">Download</a> <a href="#">Open</a> <a href="#">Delete</a>						
Name	Type	Last modified	Size	Storage class		
resized-griziz.png	png	July 14, 2024, 23:10:37 (UTC+07:00)	20.5 KB	Standard		
griziz.png	png	July 14, 2024, 23:10:35 (UTC+07:00)	70.7 KB	Standard		

Figure 44: Images are stored in S3.

The photo I just uploaded, “griziz.png”, is automatically taken to create a resized version called “resized-cole.png” and uploaded to the same bucket. You can see the image above to verify that both 2 images exist, and resized version is significantly lighter (12.3 KB compared to 42.3KB)

#### IV. CONCLUSION

Throughout this report, all my works in Assignment 2 – Unit COS20019 Cloud Computing Architecture are reported. Based on the given requirements of the website, I have deployed everything from scratch, from setting up infrastructure to website functionalities deployment.

#### REFERENCES

- [1] Amazon Web Services. (2012, Oct 17). “Examples of Amazon S3 bucket policies”. Amazon Simple Storage Service Documentation. <https://docs.aws.amazon.com/AmazonS3/latest/userguide/example-bucket-policies.html> (accessed July. 14, 2024).