

COS20019 – Assignment 2

Developing a highly available Photo Album website

Matthew Gibbons, 105335503, 11/5/25
Tutorial Class Monday 4:30 PM

I INTRODUCTION

Throughout Assignment 2, I extended upon the infrastructure I developed in Assignment 1b via implementing lambda functions and IAM roles. I also developed two web instances through the use of an auto-scaling group and a launch template containing an AMI that I created within a Dev instance. Furthermore I added security features by restricting S3 access using a security policy, used security groups for all instances/ELB'S, and controlled traffic through the use of NACL's.

Deployed Website:

<http://photoloadbalancer-164214825.us-east-1.elb.amazonaws.com/photoalbum/album.php>

II. Creating VPC

Firstly, I created the Virtual Private Cloud for the infrastructure and assigned its name as “”, region, and two availability zones, “Zone A” and “Zone B”, with a public and a private subnet within each of the zones. I then created a route table and an internet gateway and connected them.

The screenshot shows the AWS VPC dashboard with the URL <https://us-east-1.console.aws.amazon.com/vpc/>. The left sidebar shows navigation options like EC2 Global View, Virtual private cloud (selected), Security, and PrivateLink and Lattice. Under 'Your VPCs', there are two entries:

Name	VPC ID	State	Block Public...	IPv4 CIDR
-	vpc-085221d3640f90afc	Available	Off	172.31.0.0/16
MGibbonsVPC	vpc-05da8d5344d663854	Available	Off	10.0.0.0/16

A message at the bottom says "Select a VPC above".

Figure 2 - VPC Creation

The screenshot shows the AWS Subnets dashboard with the URL <https://us-east-1.console.aws.amazon.com/vpc/subnets/>. The left sidebar shows navigation options like EC2 Global View, Virtual private cloud (selected), Subnets (selected), Security, and PrivateLink and Lattice. Under 'Subnets (10) Info', there are ten entries:

Name	Subnet ID	State	VPC
Private subnet 2	subnet-0d2a5374c468c2f66	Available	vpc-05da8d5344d663854 MGI...
-	subnet-093dac735bd3140d2	Available	vpc-085221d3640f90afc
-	subnet-098e4b469b0c9bb01	Available	vpc-085221d3640f90afc
Private subnet 1	subnet-073809060bf3ab485	Available	vpc-05da8d5344d663854 MGI...
-	subnet-055ff77ec3ca5eb17	Available	vpc-085221d3640f90afc
-	subnet-093887ab521432b3f	Available	vpc-085221d3640f90afc
Public subnet 2	subnet-03ef53a9b335cebe7	Available	vpc-05da8d5344d663854 MGI...
Public subnet 1	subnet-0e2e09b72a8007211	Available	vpc-05da8d5344d663854 MGI...

A message at the bottom says "Select a subnet".

Figure 1 - All Subnets Created within VPC

The screenshot shows the AWS VPC dashboard with the 'Route tables' section selected. The table lists two route tables: 'PrivateRT' and 'PublicRT'. Both route tables are associated with 2 subnets and have 'Explicit subnet associations' set to 'Yes'. The 'Create route table' button is visible at the top right.

Name	Route table ID	Explicit subnet assoc...	Edge associations	
-	rtb-0e7955b92963ccca1	-	-	Yes
-	rtb-0f9cc3c9cf4d89fb	-	-	Yes
<input checked="" type="checkbox"/> PrivateRT	rtb-0456da689d1cec4f9	2 subnets	-	No
<input checked="" type="checkbox"/> PublicRT	rtb-0302906e68a102231	2 subnets	-	No

Figure 3 - Route Tables For Subnets

III. Creating NAT Gateway

I then created a NAT Gateway in the public subnet in Availability Zone 1a, which will be used later in order to enable the web instances in the private subnets to access the internet.

The screenshot shows the AWS VPC dashboard with the 'NAT gateways' section selected. A single NAT gateway named 'NAT Gateway' is listed, with its status as 'Available'. The 'Create NAT gateway' button is visible at the top right.

Name	NAT gateway ID	Connectivity...	State	State message	Prim...
NAT Gateway	nat-00b10f7c7ef622087	Public	Available	-	54.2

Figure 4 - The NAT Gateway

IV. Creating Elastic Load Balancer

I then created the Elastic Load Balancer and configured it to run health checks on the “/photoalbum/album.php” pathway in order to ensure the ELB is running health checks on all instances.

The screenshot shows the AWS EC2 Load Balancers console with the 'PhotoLoadBalancer' configuration page. The left sidebar includes links for Dashboard, EC2 Global View, Events, Instances (with sub-links for Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations), Images (AMIs, AMI Catalog), Elastic Block Store (Volumes, Snapshots, Lifecycle Manager), and Network & Security (CloudShell, Feedback). The main content area displays the 'PhotoLoadBalancer' details, including:

- Details:** Load balancer type: Application, Status: Provisioning, Scheme: Internet-facing, Hosted zone: Z355XD0TRQ7X7K.
- VPC:** vpc-05da8d5344d663854.
- Availability Zones:** subnet-03ef53a9b335cebe7 (us-east-1b (use1-az1)), subnet-0e2e09b72a8007211 (us-east-1a (use1-az6)).
- Load balancer ARN:** arn:aws:elasticloadbalancing:us-east-1:649193435357:loadbalancer/app/PhotoLoadBalancer/f7e213cd8591c7ed.
- DNS name:** PhotoLoadBalancer-164214825.us-east-1.elb.amazonaws.com (A Record).

Below this, the 'Listeners and rules' tab is selected, showing one listener rule (1) with options to Manage rules, Manage listener, or Add listener. A note states: "A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules." A search bar labeled 'Filter listeners' is present.

Figure 5 - Elastic Load Balancer for the PhotoAlbum site

V. Configuring S3 Bucket

I then utilised the same S3 bucket that was created in assignment 1b, however, I altered the policies of the bucket in order to only allow access from the web server and to deny access to all other HTTP traffic.

The screenshot shows the AWS S3 Buckets console with the 'Buckets' page. The left sidebar includes links for General purpose buckets (with sub-links for Directory buckets, Table 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 a Feature spotlight section. The main content area displays the 'General purpose buckets' section, which shows one bucket named 'photos-bucket-assignment1b'. The bucket details are as follows:

Name	AWS Region	IAM Access Analyzer	Creation date
photos-bucket-assignment1b	US East (N. Virginia) us-east-1	View analyzer for us-east-1	April 13, 2025, 15:53:15 (UTC+10:00)

Figure 6 - S3 Bucket from assignment 1b

VI. Creating the Dev Instance

Next, I created the Dev instance, which will later be used to create the AMI for the Web instances. I also SSH into the dev instance and downloaded AWS SDK for PHP, Apache, PHP, and Mysql.

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with navigation links like Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images (AMIs, AMI Catalog), Elastic Block Store (Volumes, Snapshots, Lifecycle Manager), and Network & Security. The main content area shows a table titled 'Instances (1/1)'. The table has columns for Name, Instance ID, Instance state, Instance type, Status check, and Alarm status. A single row is selected: 'Dev Instance' with Instance ID 'i-02bd324fd69eba0dd', Instance state 'Running', Instance type 't2.micro', Status check 'Initializing', and Alarm status 'View alarms+'. Below the table, a detailed view for 'i-02bd324fd69eba0dd (Dev Instance)' is shown with tabs for Details, Status and alarms, Monitoring, Security, Networking, Storage, and Tags. Under the Details tab, there's an 'Instance summary' section with fields for Instance ID (i-02bd324fd69eba0dd), Public IPv4 address (empty), Private IPv4 addresses (10.0.2.76), IPv6 address (empty), Instance state (Running), Public IPv4 DNS (empty), and Hostname type (IP name: ip-10-0-2-76.ec2.internal). At the bottom of the page, there are links for CloudShell and Feedback, and a footer with copyright information.

Figure 8 - Dev Instance I Created

The screenshot shows an SSH session window titled '103828 — ec2-user@ip-10-0-2-76:/var/www/html — ssh -i ~/Desktop/AssignmentKey.pem'. The session starts with two blank lines, followed by the command 'ssh -i ~/Desktop/AssignmentKey.pem ec2-user@3.208.231.191'. This is followed by a series of ASCII art characters forming a stylized logo or banner. The text continues with 'Last login: Wed May 7 04:01:52 2025 from 49.190.21.75', '[ec2-user@ip-10-0-2-76 ~]\$ sudo yum update -y', 'Last metadata expiration check: 0:48:26 ago on Wed May 7 03:22:49 2025.', 'Dependencies resolved.', 'Nothing to do.', 'Complete!', '[ec2-user@ip-10-0-2-76 ~]\$ sudo yum install -y httpd', 'Last metadata expiration check: 0:48:33 ago on Wed May 7 03:22:49 2025.', 'Dependencies resolved.' The session ends with a series of '=' characters.

Figure 7 - SSH into Dev Server

```
103828 — ec2-user@ip-10-0-2-76:/var/www/html — ssh -i ~/Desktop/As...
Last metadata expiration check: 0:48:26 ago on Wed May 7 03:22:49 2025.
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-10-0-2-76 ~]$ sudo yum install -y httpd
Last metadata expiration check: 0:48:33 ago on Wed May 7 03:22:49 2025.
Dependencies resolved.
=====
Package           Arch      Version       Repository   Size
=====
Installing:
httpd            x86_64    2.4.62-1.amzn2023      amazonlinux 48 k
Installing dependencies:
apr               x86_64    1.7.5-1.amzn2023.0.4    amazonlinux 129 k
apr-util          x86_64    1.6.3-1.amzn2023.0.1    amazonlinux 98 k
generic-logos-httpd noarch   18.0.0-12.amzn2023.0.3  amazonlinux 19 k
httpd-core        x86_64    2.4.62-1.amzn2023      amazonlinux 1.4 M
httpd-filesystem  noarch   2.4.62-1.amzn2023      amazonlinux 14 k
httpd-tools        x86_64    2.4.62-1.amzn2023      amazonlinux 81 k
libbrotli         x86_64    1.0.9-4.amzn2023.0.2    amazonlinux 315 k
mailcap           noarch   2.1.49-3.amzn2023.0.3   amazonlinux 33 k
Installing weak dependencies:
apr-util-openssl x86_64    1.6.3-1.amzn2023.0.1    amazonlinux 17 k
mod_http2         x86_64    2.0.27-1.amzn2023.0.3   amazonlinux 166 k
```

Figure 9 - Downloading Required Files

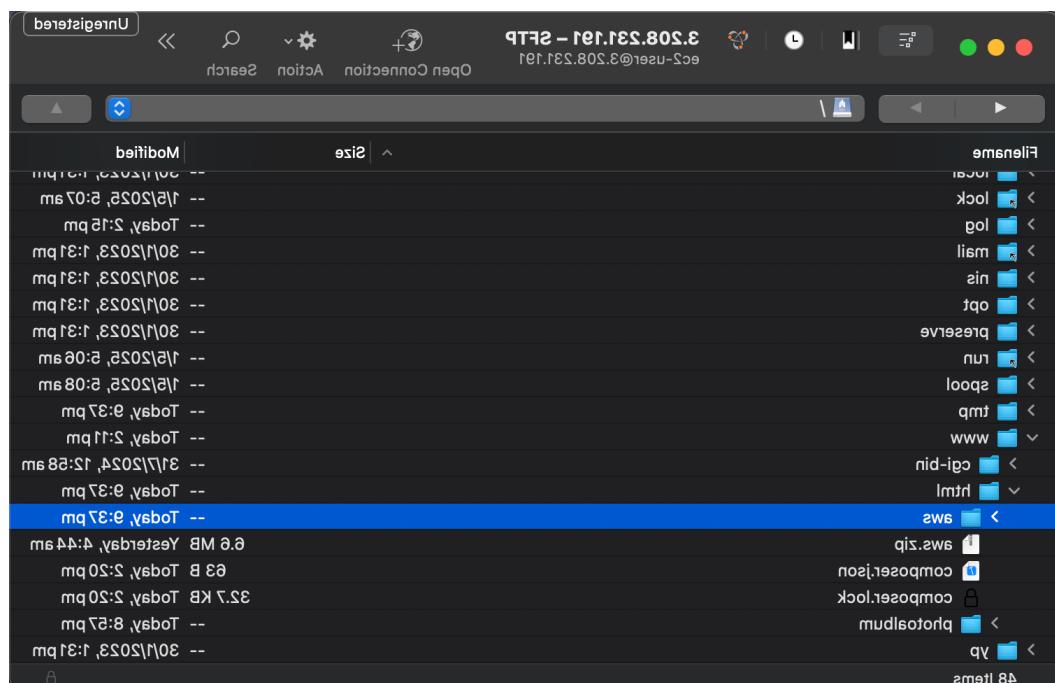


Figure 10 - Directory Structure on Cyberduck

VII. Configuring Security Groups

I then created the security groups needed for the Dev server, Web Servers, Load balancer, and Database. After creating and editing the respective AWS utilised and associated them with their security group.

The screenshot shows the AWS Management Console interface for managing security groups. The left sidebar contains navigation links for Images, Elastic Block Store, Network & Security (Security Groups selected), Load Balancing, Auto Scaling, and Settings. The main content area is titled "Security Groups (7) Info". It displays a table with columns: Name, Security group ID, Security group name, and VPC ID. The table lists the following entries:

Name	Security group ID	Security group name	VPC ID
-	sg-038d77f8bd8dc05e	launch-wizard-3	vpc-05da8d5344d6f
-	sg-00de4ebd5b1d775e9	ELBSG	vpc-05da8d5344d6f
-	sg-04d56c92350c9df19	default	vpc-05da8d5344d6f
-	sg-04d60a5b10e6f3a03	default	vpc-085221d3640f9
-	sg-0973b4da15a56e546	DBServerSG	vpc-05da8d5344d6f
-	sg-0e5230d44678ef61a	DevServerSG	vpc-05da8d5344d6f
-	sg-0ea0184a29390b8b8	WebServerSG	vpc-05da8d5344d6f

Below the table, there is a section titled "Select a security group" with a dropdown menu. The footer of the page includes links for CloudShell, Feedback, and various AWS terms like Privacy, Terms, and Cookie preferences.

Figure 11 - All Security Groups

VIII. Creating a Lambda Function

I then created the Lambda Function that will be used to resize the images, I also configured the Lambda Function with one of the pre-made IAM Roles in order to improve security.

The screenshot shows the AWS Lambda function configuration page for "CreateThumbnail". The top navigation bar indicates the function is located at "Lambda > Functions > CreateThumbnail". The main area is titled "CreateThumbnail" and contains the "Function overview" tab. It shows a diagram of the function structure with a single layer named "CreateThumbnail". Below the diagram are buttons for "Add trigger" and "Add destination". To the right, there are sections for "Description", "Last modified" (50 minutes ago), "Function ARN" (arn:aws:lambda:us-east-1:649193435337:function:CreateThumbnail), and "Function URL". At the bottom of the configuration pane, there are tabs for "Code", "Test", "Monitor", "Configuration", "Aliases", and "Versions".

At the bottom of the page, there is a "Code source" section with a code editor containing the Python script "lambda_function.py". The script has a single line of code: `print("Hello from Lambda")`. There is also an "Upload from" button and a file browser icon. The footer of the page includes links for CloudShell, Feedback, and various AWS terms like Privacy, Terms, and Cookie preferences.

Figure 12 - Create Thumbnail Lambda Function

```

import boto3
import os
import sys
import uuid
import json
from urllib.parse import unquote_plus
from PIL import Image
from botocore.exceptions import ClientError

def lambda_handler(event, context):
    # This Lambda function accepts the following input: {"bucketName": "your-photo-bucket-name"}
    # ONLY PNG FILES ARE SUPPORTED!!
    # This Lambda function downloads your-photo.png from your-photo-bucket-name S3
    # ...
    s3_client = boto3.client('s3')

    if resize_image(image_path, resized_path):
        with Image.open(image_path) as image:
            ...

```

Figure 13 - Code Source for Lambda Function

IX. Creating RDS Database

I then created the RDS Database, which will be used to store the image metadata. I then configured the Database's structure via phpMyAdmin, after SSH into the Web server.

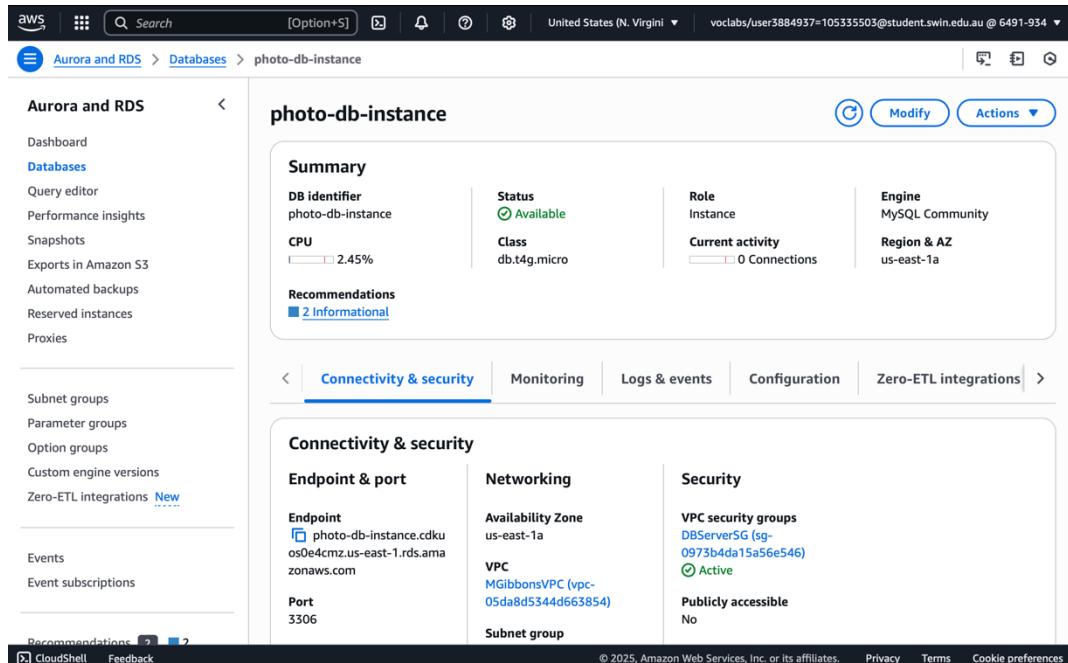


Figure 14 - Photo RDS Database

Figure 15 - Database Structure in phpMyAdmin

X. Creating AMI In Web Instance

I then utilised the Web Instance in order to create an image, which I will use later to create the Launch Template for the Auto Scaling group.

Figure 16 - AMI of Web Server

XI. Creating the Auto scaling group & Launch template

I then used the AMI I created in order to create the launch template. I then used the Launch template to create an Auto scaling group in order to create two Web Instances in the private subnets.

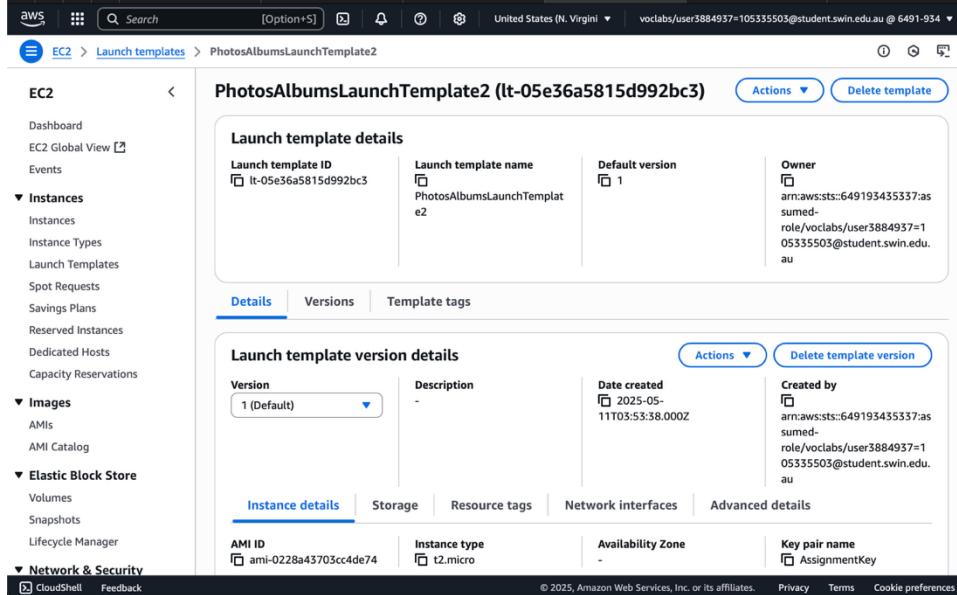


Figure 17 - Launch Template

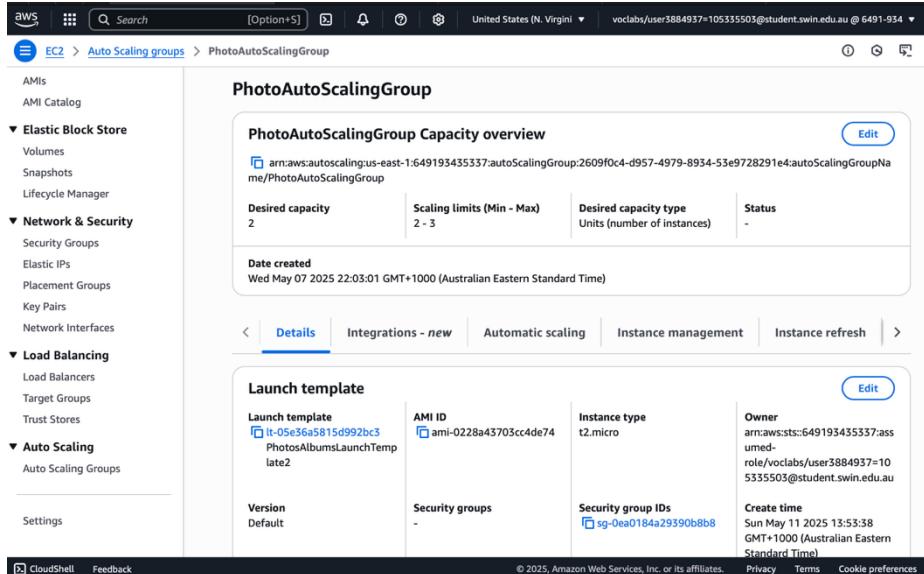


Figure 18 - Auto Scaling Group

XII. Configuring NACL's

I then created NACL policies in order to add an additional security feature via blocking ICMP traffic to and from the Dev Server.

The screenshot shows the AWS VPC dashboard. On the left, there's a sidebar with options like 'Your VPCs', 'Subnets', 'Route tables', etc. The main area is titled 'Network ACLs (1/3)'. It shows a table with three rows:

Name	Network ACL ID	Associated with	Default	VPC
acl-02b5c0790e2f48d1c	3 Subnets	Yes	vpc-0	
acl-07a9e7dab44bed6	6 Subnets	Yes	vpc-0	
PrivateSubnetsNACL	acl-0057931f62b5c40fd	subnet-03ef53a9b335cebe7 / Public subnet 2	No	vpc-0

Below this, there's a section for 'acl-0057931f62b5c40fd / PrivateSubnetsNACL' with tabs for 'Details', 'Inbound rules' (which is selected), 'Outbound rules', 'Subnet associations', and 'Tags'. The 'Inbound rules' table has four rows:

Rule number	Type	Protocol	Port range	Source	Action
100	SSH (22)	TCP (6)	22	0.0.0.0/0	Allow
120	All ICMP - IPv4	ICMP (1)	All	0.0.0.0/0	Deny
130	All traffic	All	All	0.0.0.0/0	Allow

Figure 19 - NACL Policies

XIII. Testing Website

Lastly I Tested the websites functionality, determining whether it could successfully enable me to upload a png with its metadata and store, resize the image into a thumbnail and print the thumbnail and metadat in the table on the album.php page.

Student name: Matthew Gibbons

Student ID: 105335503

Tutorial session: Monday 16:30AM

Uploaded photos:

[Upload more photos](#)

Photo	Name	Description	Creation date	Keywords
	Truck	orange truck	2025-05-11	truck, orange, cool
	Truck2	red truck	2025-05-11	truck, red, ok

Figure 20 - Album.php Page in Browser

Photo uploader

Photo title:

Select a photo (Select PNG file for best result): no file selected

Description:

Date:

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

[Photo Album](#)

Figure 21 - photouploader.php page in Browser

XIV. Learner Lab Credit Usage

Throughout the assignment, I was cautious not to spend unnecessary credits on my Learner Lab account. In order to limit my credit usage, I ensured no unnecessary instances were running. I also removed/terminated resources used from assignment 1 or assignment 1b.

XV. Conclusion

Ultimately, I was successfully able to complete and fulfil the functional and infrastructure requirements of Assignment 2. Although I had difficulties with the use of the auto scaling group and launch template, I was able to overcome these difficulties by reviewing instructions from previous labs. I am proud of my achievements throughout this assignment and am significantly more confident in my ability to create highly available websites.