COS80001 - Cloud Engineering

Assignment 1 - Part B

Creating and deploying Photo Album website onto a simple AWS infrastructure

Due date: **9:00 AM (AEST) Monday, start of Week 7**
Please refer to **Portfolio Rubric and Progress Tracker** for marking details.

Prerequisite requirements:

- Successfully passed Assignment 1A.
- ACF Labs 2, 3, and 4.
- Know how to set up and manage a MySQL database.
- General understanding of PHP programming language.
- Know how to set up and manage a Web accessible S3 bucket.

All supporting materials mentioned in this document can be found in the corresponding assignment page on Canvas.

The PHP source code has been provided for this assignment. However, you will need to understand how this code works to be able to modify the missing parts. Each student is supposed to add their own specific information in this code; hence, you must not copy someone else's code.

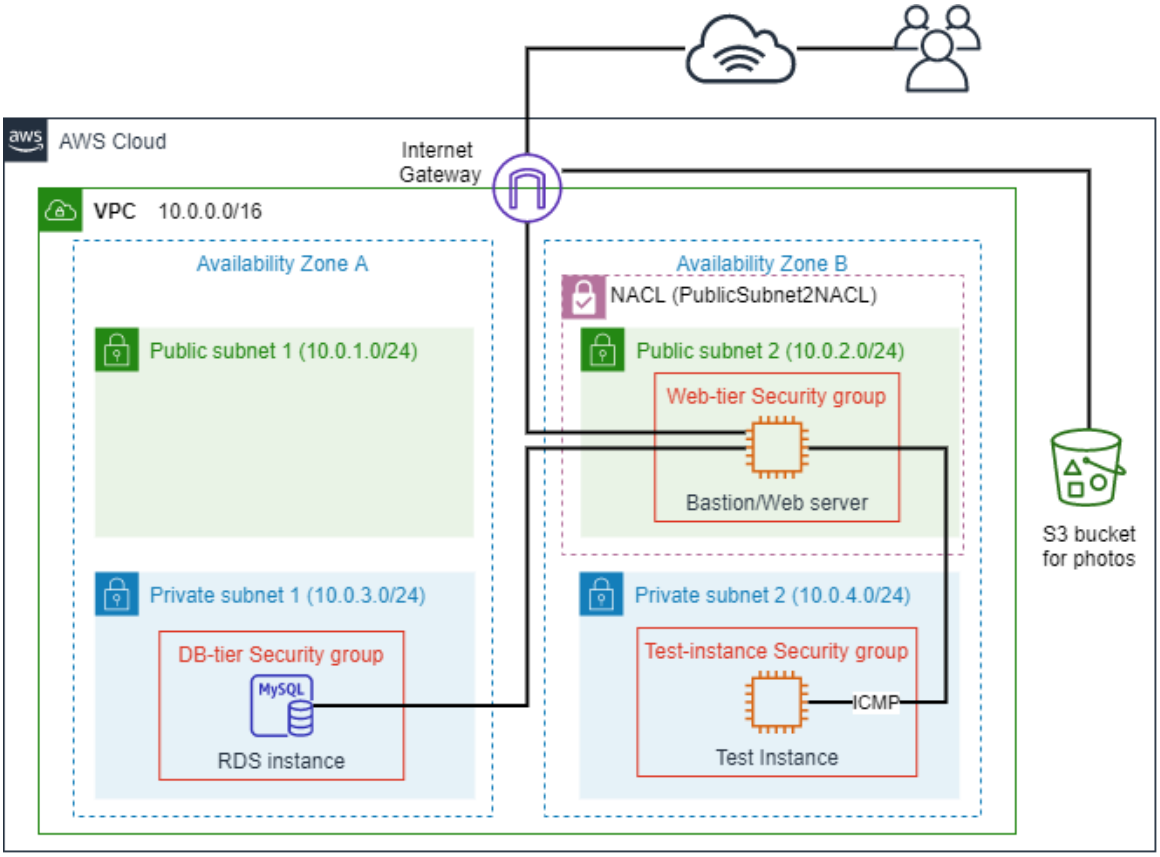
Objectives

This assignment has the following objectives:

1. Create a secure Virtual Private Cloud (VPC) with subnets, routing tables and security groups.
2. Control access to and from your VPC via an Internet Gateway.
3. Modify the provided PHP code to create a website that stores meta-data information about photos uploaded to S3 in a MySQL database managed by Amazon RDS. The website should enable the user to search for and display photos using meta-data.
4. Deploy and test your PHP web site on an Apache web server running on an EC2 virtual machine instance.
5. Add and aditional layer of security by applying a Network ACL to the public subnet that hosts your web server.

1. Infrastructure deployment

You will set up a VPC with the structure and services as illustrated in the diagram below.



NOTE: Do not use the default VPC. All services should be in your custom VPC. Below are the detailed requirements for each service.

1.1 - VPC:

- Name: [*FirstNameInitial*][*LastName*]*VPC*. For example, if your name is Bill Gates, your VPC would be named "*BGatesVPC*".
- Region: us-east-1
- Two availability zones each with a private and public subnet with suitable CIDR as specified in the diagram above.
- Associate public subnets with a public route table that routes to the Internet Gateway.

NOTE: due to some incompatibility issues, it is recommended to create your VPC manually (use the "**Create VPC**" button in VPC tab). Please do NOT use the "**Start VPC Wizard**" button in AWS dashboard.

1.2 - Security groups

Create the following security groups, each is associated with each tier shown in the architecture diagram:

Security group name	Protocols	Source
TestInstanceSG	All traffic	Anywhere
WebServerSG	HTTP (80), SSH (22)	Anywhere
	ICMP	TestInstanceSG
DBServerSG	MySQL (3306)	WebServerSG

1.3 – EC2 virtual machine

You will create two EC2 instances, a test instance and a bastion/web server instance.

1.3.1 – Bastion/Web server instance

Your web server must be deployed on an EC2 instance in Public Subnet 2. This EC2 instance should be configured similar to the EC2 created in Assignment 1A:

- Amazon Machine Image: *Amazon Linux 2023 AMI*
- Instance type: *t2.micro*
- Has Apache web server and other PHP packages installed (you can use the same bash script provided in Assignment 1A to bootstrap your EC2).

This instance will host the "Photo Album" web application, which was created in Assignment 1A – more details are in Section 2 of this specification document. This instance will also act as a bastion host for you to SSH into the Test instance, which resides in a private subnet.

NOTE: [your.public.dns] will change every time the webserver instance restarts. To avoid this behaviour and to ensure your Webserver URL remains persistent, add an Elastic IP Address to this instance by allocating an Elastic IP address in the same region under the Network and Security section in the left menu of the EC2 service page, then associate this new EIP to your Bastion/Web server instance. The public IP address of your instance should now automatically match this Elastic IP address.

1.3.2 – Test instance

This instance will be used for demonstration purposes only. It does not contribute to the functionality of Photo Album website. You will SSH into this instance and ping the web server (using "ping" command in Linux). Please take a screenshot(s) of the Linux terminal to demonstrate that:

- You are able to SSH into an instance in a private subnet (which is this Test instance). For instructions on how to connect to a private EC2 instance through a bastion host, you can refer to https://aws.amazon.com/blogs/security/securely-connect-to-linux-instances-running-in-a-private-amazon-vpc/
- You are able to establish a connection (ICMP ping) between this instance and the Bastion/Web server instance.

The configuration of this instance is entirely your choice. This instance does not host the web application.

1.4 – RDS database instance

Your RDS instance must have the following configs:

- DB engine version: *MySQL 8.0.39*
- Template: *Free tier*
- Public access: *No*
- Resides in private subnets.

NOTE: Your RDS instance needs to be in a private subnet. Only WebServerSG security group can access it. However, you need to be able to access your database over the internet so that you can set it up and maintain it. There are several ways to do this. The easiest way is to install phpMyAdmin (a web-based MySQL administration tool) on your EC2 web server instance and manage your database through phpMyAdmin's UI. Instructions on how to do this are in *Install phpMyAdmin on EC2.pdf* file.

Create a database in your RDS instance with a table called ***photos*** that stores meta-data about the photos stored in the S3 bucket. This table should have the following columns:

- Photo title (*varchar(255)* type)
- Description (*varchar(255)* type)
- Creation date (*date* type)
- Keywords (*varchar(255)* type)
- Reference to the photo object in S3 (*varchar(255)* type)

1.5 – Network ACL

To add an additional layer of security to your web server, you have been asked to design and deploy a Network ACL (named "PublicSubnet2NACL") that limits ICMP and other necessary traffic to the corresponding subnet (Public Subnet 2). This NACL must follow the least-privilege principle. In other words, irrelevant traffic from irrelevant sources must not be allowed. To be specific, the NACL:

- must ALLOW SSH(22) [1]traffic from anywhere so that you can access the WebServer instance.
- must ALLOW ICMP traffic *only* from the subnet that contains the Test instance.
- must ALLOW other necessary traffic so that the Photo Album website is fully functional for users from anywhere.

2. Functional requirements of Photo Album website

Your Photo Album website must have the following functional requirements.

2.1 – Photo storage

Create an S3 bucket to store your photos. Manually upload some photos onto S3 bucket that you just created and ensure they have been successfully uploaded.

[1] *Ideally, SSH(22) traffic should only be allowed from your home network's public IPv4 address range since common users do not need to access the web server. But for simplicity, you can allow SSH from anywhere in this assignment.*

All objects (photos) in this S3 bucket must become publicly available. To accomplish this task, you MUST use an appropriate access policy to enable public access to all available objects in this S3 bucket.

NOTE: marks will be deducted if S3 bucket objects have been individually configured to be publicly available.

2.2 – Photo meta-data in RDS Database

The meta-data of the photos stored in the S3 bucket is stored in a database table, which has been created in Section 1.4. You need to populate the table with a few records. Below is an example of a record:

- Photo title: *Swinburne Logo*
- Description: *Logo of Swinburne uni*
- Creation date: *2021-08-09*
- Keywords: *logo, university*
- Object URL in S3: *https://photo-bucket.s3.amazonaws.com/swinburnelogo.jpg*

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). The full source code has been provided to you (*photoalbum_v3.0.zip*). Modify the *constants.php* file in the provided code (carefully read the comments in the file) using available information from the S3 bucket and RDS database that you created in the previous steps.

NOTE: in constants.php file, variables/names that you add must not include space. You may use underscore "_" instead of space.

Examples:

```
define('DB_PHOTO_CREATIONDATE_COL_NAME', 'creationdate');
define('DB_PHOTO_KEYWORDS_COL_NAME', 'keywords_column');
```

The website should be accessible through http://[*your.public.dns*].amazonaws.com/cos80001/photoalbum/album.php if the directory structure in your web server is correctly created.

Testing

Manually upload several photos to the S3 bucket and insert their meta-data into the database. Thoroughly test to make sure the photos and their meta-data are correctly displayed.

Ensure the Network ACL satisfies the additional security requirement, by login into the Test instance (e.g. via SSH) and run a ping to the web server's IP address.

Submission

Make sure your website is functional from the due date - check you can start the web server EC2 instance and stop it. (No need to start the Test instance).

Submission is a single PDF document to Canvas. No demonstration is required. The document must contain the following:

1. A **single PDF document**, **maximum 15 pages,** in **IEEE Conference Style** in either one or two column mode submitted to Canvas by the due date.
2. Title page with your name, student ID, and tutorial class.
3. URL of the album.*php* pages on your EC2 so your marker can view your website from their browser using the URL that you have provided (Elastic IP address to be used).
4. *If your assignment is done in your personal AWS account instead of Vocareum*, you will need to create an IAM user with proper permissions and provide us with the credentials so that your tutor can access your AWS management console.
5. Well formatted Screenshot(s) of the data records in your database.
6. Well formatted Screenshot(s) of Linux terminal showing you have been able to ping the Web server Instance from your Test instance.
7. Well formated screenshot(s) and a brief explanation for each step that you have taken, problems that you faced and achievements during your deployment for this assignment.
8. Each screenshot must include have your AWS Management Console username/student ID visible.

NOTE:

This assignment is to be completed in a managed AWS Lab environment (e.g. AWS Learner Lab), which is accessible through your AWS Canvas page. For further information of how to access this environment please refer to your Swinburne Canvas Page "**Accessing AWS Resources**".

This environment is time-limited until the end of semester and comes with $US100 credit. It is your responsibility to use and manage this credit correctly to ensure there will be enough remaining credits for all assignments.

Marks will be deducted if your assignment resources are not accessible due to insufficient credits.

Assignment 1B Checklist

Make sure all the following are completed.

Submission Checklist

Student Name: ……

Student Id: ……

Tutorial time: ……

Date of submission: ……

Submit to Canvas:

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

Marking Scheme

Infrastructure Requirements	
VPC with 2 public and 2 private subnets	Rubric on Canvas
Correct Public and Private Routing tables with correct subnet associations	Rubric on Canvas
Security groups properly configured and attached.	Rubric on Canvas
Network ACL properly configured and attached	Rubric on Canvas
Correct Web server and Test instances running in correct subnets	Rubric on Canvas
Database schema as specified	Rubric on Canvas
Database running in correct subnets	Rubric on Canvas
S3 objects publicly accessible, using proper access policy	Rubric on Canvas
Functional Requirements	
album.php page displayed from EC2 Web server	Rubric on Canvas
Provided URL is persistent (Elastic IP Association)	Rubric on Canvas
Photos loaded from S3 with matching metadata from RDS	Rubric on Canvas
Web server instance reachable from Test instance via ICMP	Rubric on Canvas
Deductions	
Documentation not as specified or poorly presented (up to minus 100)	
Serious misconfigurations of AWS services being used (up to minus 100)	

Comments