COS20019 - Cloud Computing Architecture

Assignment 1b

File Manager web site

Due Date: Week 7 (22 October 2024, 11:59 PM)
Weighting: 10%

You must submit the completed template to have your assignment assessed and be eligible to pass the unit.

To complete this Assignment, you will need to…

- Successfully pass Assignment 1a.
- 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.

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 files uploaded to S3 in a MySQL database managed by Amazon RDS. The website should enable the user to view all files and their meta-data.
4. Deploy and test your PHP web site on an Apache web server running on an EC2 virtual machine instance.

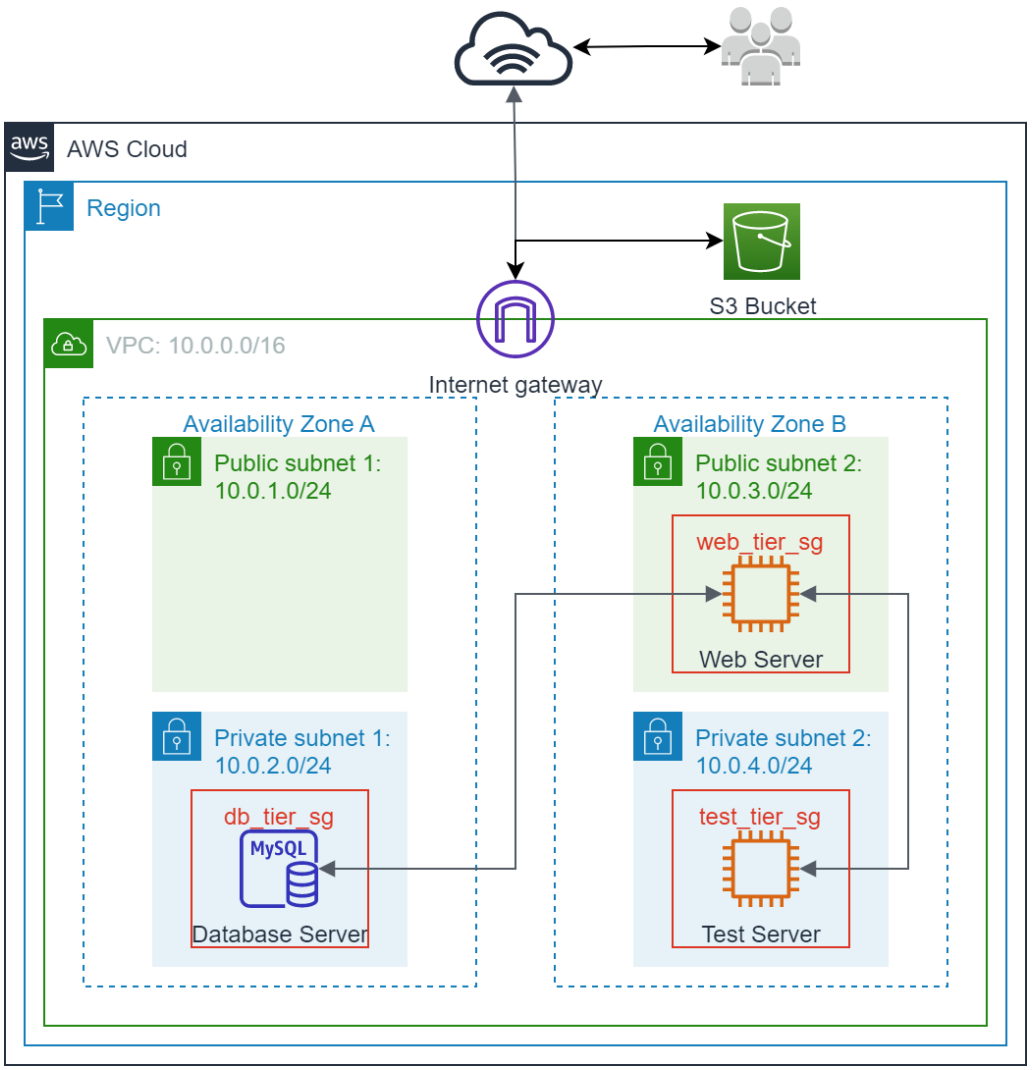
Important:
In your COS20019 assignments, all AWS resources you create (e.g., EC2 instances, Security groups, RDS database instances, etc.) should have the following additional tags added:

- StudentName (with a value of your name)
- StudentID (with a value of your id)

These tags are in addition to any other tags that are appropriate to add to the resource.
These tags will be used to assist in the assessment of your work

Infrastructure requirements

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.

VPC

- The name of your VPC **must** be in the format [*StudentID*]VPC. For example, if your student id is 12345 your VPC would be named **`12345VPC`**.
- Region: us-east-1
- Two availability zones, each with a private and public subnet with specified CIDR
- Associate public subnets with a route table that routes to an Internet Gateway

Security Groups

Create the following security groups

SG Name	Protocols	Source	Assign To
test_tier_sg	All traffic	Anywhere	Test Server
web_tier_sg	HTTP (80), SSH (22)	Anywhere	Web Server
	All ICMP - IPv4	test_tier_sg	Web Server
db_tier_sg	MySQL (3306)	web_tier_sg	Database Server

EC2 Instance (Web Server)

- Amazon Machine Image: Amazon Linux 2 AMI (HVM)
- Instance type: t2.micro
- User data: install Apache Web Server and PHP (as in Assignment 1a)

EC2 Instance (Test Server)

- This instance will be used for demonstration purposes only. It does not contribute to the functionality of the website. You will SSH into this instance and ping the web server (using "ping" command in Linux).
- 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/

RDS Instance (Database Server)

- DB engine version: MariaDB 10.11.8
- Template: Free tier
- DB instance class: db.t4g.micro
- Storage autoscaling: disabled
- Public accessibility: No
- Backup retention period: 0 days

RDS Hint #1:

RDS requires that you have at least two Availability Zones. However, while it is desirable to have a master-slave Multi-AZ RDS in a production deployment, this is **not** available as part of the AWS Free-Tier.

Before creating your RDS instance, ensure that you have created a **DB Subnet Group**. This subnet group specifies which subnets the RDS instance will be launched in.

RDS Hint #2:

In production, you need your RDS instance to be in a private subnet with only the web tier security group being able to access it. This means that you cannot directly connect to the database server. However, there are a number of ways for you to manage your database. It is up to you to choose your method:

1. Install phpMyAdmin on your Web Server instance, then create and maintain the database through its UI (least secure method).
2. Use your Web Server instance as a 'Bastion Host' to SSH tunnel into the RDS instance (most secure method).
3. Create an "Admin" interface in PHP that allows a user with the correct credentials to run create, insert and delete SQL operations on the database, so that tables and records can be managed.

Functional requirements

Storage

Create a S3 bucket to store your files. This bucket should be configured to be accessible publicly. Manually upload a number of files into the S3 bucket and record their URLs so they can be referenced in the database. All objects (photos, videos, documents, etc.) 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: In **this** assignment you **will not** be required to add functionality to the PHP web page (*upload.php*) you created in Assignment 1a.

Database

Create a database in your RDS instance with the following table:

Files
Stores meta-data of each file
• Filename (varchar) • Description (varchar) • Creation date (date) • Keywords (varchar) • S3 URL (varchar)

Web pages

You can choose your own design for the web site. The website must be able to list all files (linked to S3) along with their meta-data (from the database). Sample source code has been provided (file_manager.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.

The directory structure of your website is described below. You can create additional HTML, CSS, JavaScript files if needed. The Apache HTTP server serves files located in a directory called Apache document root (/var/www/html).

```
COS20019/
        . . . AWS SDK, other support libraries
        file_manager /
                upload.php (not yet fully implemented)
                get_files.php
                file.php
                . . . other PHP, HTML, CSS, JavaScript files
```

Testing

Upload a number of files to S3 and input their meta-data and keywords into the RDS database. Some of the files should have keywords in common. Thoroughly test that the files and their meta-data are correctly displayed.

Marking Scheme

Infrastructure Requirements (10 marks)	**Marks**
VPC with 2 public and 2 private subnets	2
Correct Public and Private Routing tables with correct subnet associations	1
Security groups properly configured	2
Web, Test, and Database server properly setup and running in correct subnet	2
S3 bucket properly configured and contains at least 5 files inside (e.g. jpg, docx, pdf)	2
Database schema properly created	1

Functional Requirements (5 marks)	**Marks**
Web page (get_files.php) can be accessed on EC2 instance	1
Files loaded from S3 and information shown from database	2
Web Server reachable from Test Server via ICMP ping	1
Website theme	1

Deductions	**Marks**
Resources not properly tagged	-1 to -5
Plagiarism	-1 to -15