



CCS6344 T2410 Assignment 2 Submission

Group Name: Group 16

ASYRANI SYAZWAN BIN YUHANIS	1211103222
OOI PUI KEAT	1201101582
DIVYASHREE A/P SELVANYGAM	1221303777

Design Implementation:

- **Client Business Model**

The wedding and catering management system is designed to facilitate the organization, management, and execution of wedding and catering services. Our client, a wedding and catering service provider, needs an efficient system to enhance their operations through a comprehensive management system. The wedding and catering management system aims to streamline the entire process of planning, organizing, and executing wedding and catering events, ensuring efficient management and high customer satisfaction. The key feature of the system is for managing and registration of wedding and catering staff.

- **Model Analysis**

The modules that we used in Amazon Web Services (AWS) for our client's business model are Amazon VPC, Amazon EC2, and Amazon RDS.

Amazon VPC (Virtual Private Cloud): Aims to provide a secure and isolated network environment for the wedding and catering management system. By using a VPC, we get to control the network settings and ensure that the resources are secure and properly configured.

Advantages:

Security: Isolates resources within a private network, enhancing security

Customizability: Allows customization of network configurations

Control: Provides fine-grained control over inbound and outbound network traffic through security groups and network ACLs

Amazon EC2 (Elastic Compute Cloud): This is to provide scalable and flexible compute capacity in the cloud. For the wedding and catering management system, EC2 is used to host the web server and application, ensuring that the system is always available and can handle varying levels of traffic.

Advantages:

Scalability: Easily scale up or down based on demand, ensuring optimal performance during peak times

Flexibility: Choose from a variety of instance types and configurations to meet specific needs and budget constraints

Reliability: Benefit from AWS's infrastructure, which provides high availability and redundancy.

Amazon RDS (Relational Database Service): To provide a managed relational database service that simplifies the setup, operation, and scaling of a relational database. RDS is used for the wedding and catering management system to manage the MySQL database, which stores all the critical data related to staff.

Advantages:

Managed service: Automates routine database tasks such as provisioning, patching, backup, recovery, and scaling

Scalability: Easily scale the database storage and compute resources

Security: Provides encryption at rest and in transit, and integrates with AWS IAM for access control

High availability: Supports Multi A-Z deployments for enhanced availability and durability

Model Implementation:

Design Description

VPC (Virtual Private Cloud):

1. Configure the VPC:
 - Enter details such as name, number of availability zones, subnets, and NAT gateways.
 - Create the VPC.

EC2 Instance:

1. Configure EC2 Instance:
 - Select Amazon Linux as the AMI for compatibility with AWS services.
 - Choose t2.micro instance type for cost-effectiveness and expected load.
 - Select the created VPC for networking control.
2. Configure Security Group:
 - Create a security group named 'WeddingCatering'.
 - Allow HTTP, HTTPS, and SSH access.
3. Key Pair:
 - Create a key pair named 'WeddingCatering.pem' for secure login
4. Configuring the Web Server on EC2:
 - Connect using SSH: 'ssh -i /path/to/WeddingCatering.pem ec2-user@ip-address'.
 - Create a temporary directory: 'mkdir temp'.
 - Copy application files to the temp directory: 'scp -i WeddingCatering.pem user_reg.zip ec2-user@ip-address:temp'.
 - Unzip the application files: 'unzip user_reg.zip'.
 - Moves files to '/var/www/html': 'sudo mv* /var/www/html'.
 - Install Apache: 'yum install -y httpd'
 - Start Apache: 'systemctl enable httpd' and 'systemctl start httpd'

RDS (Relational Database Service):

1. Create RDS Instance:
 - Choose MySQL as the engine type.
 - Use the Free tier template for cost management.
 - Set DB Instance Identifier to 'WeddingCatering'.
 - Set master username and password to 'admin' and 'admin123'

2. Configure Security Group:
 - Use the existing VPC and add a rule to allow MySQL access.
3. Database Endpoint:
 - Connect the application to the RDS instance using the endpoint 'weddingcatering.cpffhoagrni.us-east-1.rds.amazonaws.com'.

Connecting EC2 and RDS:

1. Update Database Configuration:
 - Modify 'database.php' to use the RDS endpoint. Master username, and master password: 'sudo nano database.php'.

Testing:

1. HTTP:
 - Access the web application using '<http://ip-address>'.
 - Some functions may show the PHP code instead of executing.
2. HTTPS:
 - Access using '<https://ip-address>' may not work correctly

Implementation Process

- VPC

1. **Configure the VPC** - Enter the details needed for the VPC settings such as the Name and Number of Availability Zones (AZs), subnets and NAT Gateways. Click “Create VPC” to finish configuring.

VPC settings

Resources to create [Info](#)
Create only the VPC resource or the VPC and other networking resources.

☐ VPC only ☒ VPC and more

Name tag auto-generation [Info](#)
Enter a value for the Name tag. This value will be used to auto-generate Name tags for all resources in the VPC.

☒ Auto-generate
WeddingCatering

IPv4 CIDR block [Info](#)
Determine the starting IP and the size of your VPC using CIDR notation.

10.0.0.0/16 65,536 IPs
CIDR block size must be between /16 and /28.

IPv6 CIDR block [Info](#)

☒ No IPv6 CIDR block
☐ Amazon-provided IPv6 CIDR block

Tenancy [Info](#)
Default

Number of Availability Zones (AZs) [Info](#)
Choose the number of AZs in which to provision subnets. We recommend at least two AZs for high availability.

1 2 3
Customize AZs

Number of public subnets [Info](#)
The number of public subnets to add to your VPC. Use public subnets for web applications that need to be publicly accessible over the internet.

0 2

Number of private subnets [Info](#)
The number of private subnets to add to your VPC. Use private subnets to secure backend resources that don't need public access.

0 2 4
Customize subnets CIDR blocks

NAT gateways (\$) [Info](#)
Choose the number of Availability Zones (AZs) in which to create NAT gateways. Note that there is a charge for each NAT gateway.

None In 1 AZ 1 per AZ

VPC endpoints [Info](#)
Endpoints can help reduce NAT gateway charges and improve security by accessing S3 directly from the VPC. By default, full access policy is used. You can customize this policy at any time.

None S3 Gateway

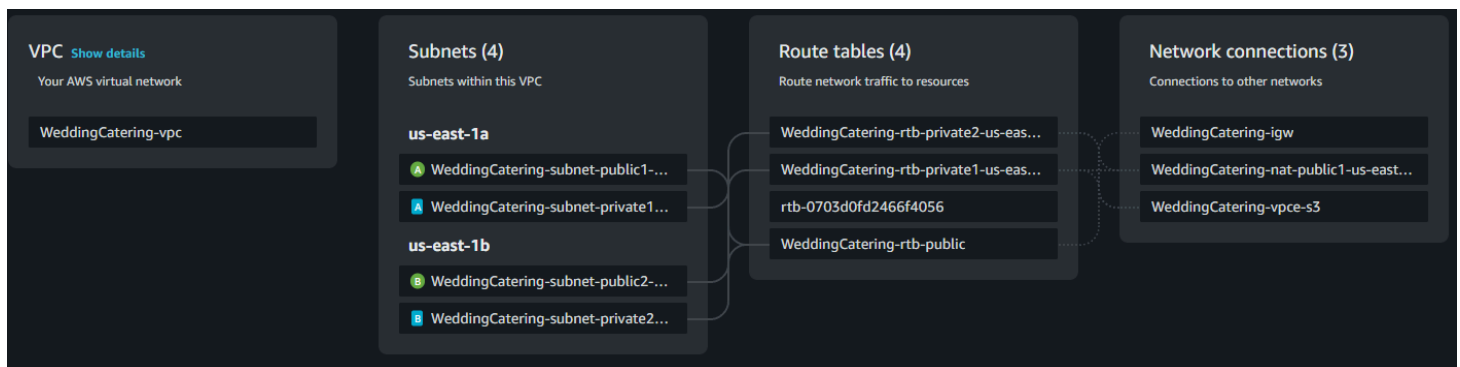
DNS options [Info](#)

☒ Enable DNS hostnames
☒ Enable DNS resolution

Additional tags

Cancel Create VPC

2. VPC is created



- **EC2 Instance**

1. **Configure EC2 Instance** - We choose Amazon Linux as the AMI for its compatibility with AWS services. For the Instance Type, we select t2.micro based on expected load and cost consideration. As for the network, we selected the WeddingCatering-vpc for easier networking control within the application.

Name

WeddingCatering

Add additional tags

▼ 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

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

Recents

Quick Start

Amazon Linux

aws

macOS

Mac

Ubuntu

ubuntu

Windows

Microsoft

Red Hat

Red Hat

SUSE Li

SUSE

Q

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2023 AMI

ami-06c68f701d8090592 (64-bit (x86), uefi-preferred) / ami-07832e309d3f756c8 (64-bit (Arm), uefi)

Free tier eligible

Virtualization: hvm

ENA enabled: true

Root device type: ebs

Description

Amazon Linux 2023 is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud applications.

Architecture

64-bit (x86)

Boot mode

uefi-preferred

AMI ID

ami-06c68f701d8090592

Verified provider

▼ Instance type Info | Get advice

Instance type

t2.micro

Free tier eligible

Family: t2

1 vCPU

1 GiB Memory

Current generation: true

On-Demand Windows base pricing: 0.0162 USD per Hour

On-Demand SUSE base pricing: 0.0116 USD per Hour

On-Demand RHEL base pricing: 0.0716 USD per Hour

On-Demand Linux base pricing: 0.0116 USD per Hour

Additional costs apply for AMIs with pre-installed software

▼ Network settings Info

VPC - required Info

vpc-0c0ab93134f2e37eb (WeddingCatering-vpc)

10.0.0.0/16

Subnet Info

subnet-06d0130978b962712 WeddingCatering-subnet-public2-us-east-1b

VPC: vpc-0c0ab93134f2e37eb Owner: 762192338171 Availability Zone: us-east-1b

IP addresses available: 4091 CIDR: 10.0.16.0/20

Auto-assign public IP Info

Enable

Additional charges apply when outside of free tier allowance

2. Configure Security Group - Create a security group to control inbound and outbound traffic. We called the security group's name as WeddingCatering. This will allow HTTP, HTTPS and SSH to be accessed through the IP Address obtained from EC2 Instance.

Firewall (security groups) | [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group ☐ Select existing security group

Security group name - required

WeddingCatering

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and ._-:/()#,@[]+=&;{}!\$*

Description - required | [Info](#)

Secure access to Wedding & Catering server

Inbound Security Group Rules

▼ Security group rule 1 (TCP, 22, 0.0.0.0/0) [Remove](#)

Type Info	Protocol Info	Port range Info
ssh	TCP	22
Source type Info	Source Info	Description - optional Info
Anywhere	<input type="text" value="0.0.0.0/0"/> Add CIDR, prefix list or security	<input type="text" value="e.g. SSH for admin desktop"/>

▼ Security group rule 2 (TCP, 443, 0.0.0.0/0) [Remove](#)

Type Info	Protocol Info	Port range Info
HTTPS	TCP	443
Source type Info	Source Info	Description - optional Info
Anywhere	<input type="text" value="0.0.0.0/0"/> Add CIDR, prefix list or security	<input type="text" value="e.g. SSH for admin desktop"/>

▼ Security group rule 3 (TCP, 80, 0.0.0.0/0) [Remove](#)

Type Info	Protocol Info	Port range Info
HTTP	TCP	80
Source type Info	Source Info	Description - optional Info
Anywhere	<input type="text" value="0.0.0.0/0"/> Add CIDR, prefix list or security	<input type="text" value="e.g. SSH for admin desktop"/>

[Add security group rule](#)

► [Advanced network configuration](#)

- ### 3. **Key Pair** - By having a key pair, it can provide a secure login information for the instance. The key pair name is WeddingCatering.pem

▼

Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

WeddingCatering

↕

↻

Create new key pair

- #### 4. **Configuring the Web Server on EC2** - First, we need to connect with SSH so that we can access the EC2 Instance. We can do so by running “ssh -i (path to WeddingCatering.pem) ec2-user@ip-address”.

```
C:\Users\Asyer>ssh -i "C:\Users\Asyer\OneDrive\Desktop\MMU\DEGREE\Semester 5\Database and Cloud Security\Assignment\Ass 2\WeddingCatering.pem" ec2-user@ec2-54-159-133-108.compute-1.amazonaws.com
```

```
#_
~\_ ##### Amazon Linux 2023
~~ \#####\
~~~ \###|
~~~~ \|/_--- https://aws.amazon.com/linux/amazon-linux-2023
      V^' ^->
       / 
    ~~~ / 
        _.-./-/
         |_/_-/-/
          m/'
```

Next, we need to create a temp directory by using “mkdir temp”.

```
[ec2-user@ip-10-0-24-185 ~]$ mkdir temp
[ec2-user@ip-10-0-24-185 ~]$ ls
temp
[ec2-user@ip-10-0-24-185 ~]$ cd temp
```

After that, in another powershell tab, we need to copy the Application files that contains the php files into the temp directory inside the EC2 Instance. To do this, we can use “scp -i WeddingCatering.pem user_reg.zip ec2-user@IP-address:temp”.

```
PS C:\Users\Asyer\OneDrive\Desktop\MMU\DEGREE\Semester 5\Database and Cloud Security\Assignment\Ass 2> scp -i WeddingCatering.pem user_reg.zip ec2-user@ec2-54-159-133-108.compute-1.amazonaws.com:temp user_reg.zip
100% 10KB 36.4KB/s 00:00
PS C:\Users\Asyer\OneDrive\Desktop\MMU\DEGREE\Semester 5\Database and Cloud Security\Assignment\Ass 2>
```

Now, we can see in the temp directory that we have the application zip file. From here, we can unzip the file using “unzip user_reg”

```
[ec2-user@ip-10-0-24-185 temp]$ ls
user_reg.zip
[ec2-user@ip-10-0-24-185 temp]$ unzip user_reg
Archive:  user_reg.zip
  inflating: user_reg/add_staff.php
  inflating: user_reg/catering_staff.php
  inflating: user_reg/database.php
  inflating: user_reg/delete_staff.php
  inflating: user_reg/index.php
  inflating: user_reg/login.php
  inflating: user_reg/logout.php
  inflating: user_reg/style.css
  inflating: user_reg/update_staff.php
  inflating: user_reg/user_registration.sql
  inflating: user_reg/wedding_staff.php
[ec2-user@ip-10-0-24-185 temp]$ ls
user_reg  user_reg.zip
```

After unzipping the file, we need to create another directory called /var/www/html and also move all the application files into this directory using the command “sudo mv * /var/www/html”.

```
[ec2-user@ip-10-0-24-185 user_reg]$ sudo mkdir -p /var/www/html
```

```
[ec2-user@ip-10-0-24-185 user_reg]$ sudo mv * /var/www/html
```

```
[ec2-user@ip-10-0-24-185 user_reg]$ cd /var/www/html
[ec2-user@ip-10-0-24-185 html]$ ls
add_staff.php      database.php      index.php      logout.php      update_staff.php      wedding_staff.php
catering_staff.php delete_staff.php  login.php      style.css      user_registration.sql
```

Now that all the application files are in the EC2 Instance, we can proceed on installing Apache by using the command “yum install -y httpd”. Make sure to be in the root user by running the “sudo su -”.

```
[ec2-user@ip-10-0-24-185 html]$ sudo su -
Last login: Thu Jul  4 22:40:11 UTC 2024 on pts/2
[root@ip-10-0-24-185 ~]# yum update -y
Last metadata expiration check: 0:16:07 ago on Thu Jul  4 22:24:34 2024.
Dependencies resolved.
Nothing to do.
Complete!
[root@ip-10-0-24-185 ~]# yum install -y httpd
Last metadata expiration check: 0:16:42 ago on Thu Jul  4 22:24:34 2024.
Dependencies resolved.
```

Package	Architecture	Version	Repository	Size
Installing:				
httpd	x86_64	2.4.59-2.amzn2023	amazonlinux	47 k
Installing dependencies:				
apr	x86_64	1.7.2-2.amzn2023.0.2	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	2.4.59-2.amzn2023	amazonlinux	1.4 M


After install, we can proceed to check start the web server by running the “systemctl enable httpd” and “systemctl start httpd”. We can also view the status by running “systemctl status httpd”.


```
Complete!
[root@ip-10-0-24-185 ~]# systemctl status httpd
○ httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: disabled)
   Active: inactive (dead)
     Docs: man:httpd.service(8)
[root@ip-10-0-24-185 ~]# systemctl enable httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
[root@ip-10-0-24-185 ~]# systemctl start httpd
[root@ip-10-0-24-185 ~]# systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)
   Active: active (running) since Thu 2024-07-04 22:43:55 UTC; 6s ago
     Docs: man:httpd.service(8)
  Main PID: 26226 (httpd)
    Status: "Started, listening on: port 80"
   Tasks: 177 (limit: 1114)
  Memory: 12.9M
    CPU: 66ms
  CGroup: /system.slice/httpd.service
          └─26226 /usr/sbin/httpd -DFOREGROUND
            └─26227 /usr/sbin/httpd -DFOREGROUND
              └─26228 /usr/sbin/httpd -DFOREGROUND
                └─26229 /usr/sbin/httpd -DFOREGROUND
                  └─26230 /usr/sbin/httpd -DFOREGROUND
```


- **RDS**


1. **Create RDS Instance** - We will be using MySQL as the engine type, since the application also used MySQL database.


Engine type [Info](#)


☐ Aurora (MySQL Compatible)



☐ Aurora (PostgreSQL Compatible)


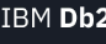
☒ MySQL


☐ MariaDB


☐ PostgreSQL


☐ Oracle


☐ Microsoft SQL Server


☐ IBM Db2


Edition

☒ MySQL Community

Engine version [Info](#)

View the engine versions that support the following database features.

▼ Hide filters

☒ Show versions that support the Multi-AZ DB cluster [Info](#)
Create a Multi-AZ DB cluster with one primary DB instance and two readable standby DB instances. Multi-AZ DB clusters provide up to 2x faster transaction commit latency and automatic failover in typically under 35 seconds.

☐ Show versions that support the Amazon RDS Optimized Writes [Info](#)
Amazon RDS Optimized Writes improves write throughput by up to 2x at no additional cost.

Engine Version

MySQL 8.0.35 ▼

☐ Enable RDS Extended Support [Info](#)

Amazon RDS Extended Support is a [paid offering](#). By selecting this option, you consent to being charged for this offering if you are running your database major version past the RDS end of standard support date for that version. Check the end of standard support date for your major version in the [RDS for MySQL documentation](#).

For the templates, we use the Free tier for easier cost management. This will also automatically provide the Single DB Instance for the availability and durability.

Templates

Choose a sample template to meet your use case.

☐ Production

Use defaults for high availability and fast, consistent performance.

☐ Dev/Test

This instance is intended for development use outside of a production environment.

☒ Free tier

Use RDS Free Tier to develop new applications, test existing applications, or gain hands-on experience with Amazon RDS.

[Info](#)

Availability and durability

Deployment options [Info](#)

The deployment options below are limited to those supported by the engine you selected above.

☐ Multi-AZ DB Cluster

Creates a DB cluster with a primary DB instance and two readable standby DB instances, with each DB instance in a different Availability Zone (AZ). Provides high availability, data redundancy and increases capacity to serve read workloads.

☐ Multi-AZ DB instance (not supported for Multi-AZ DB cluster snapshot)

Creates a primary DB instance and a standby DB instance in a different AZ. Provides high availability and data redundancy, but the standby DB instance doesn't support connections for read workloads.

☒ Single DB instance (not supported for Multi-AZ DB cluster snapshot)

Creates a single DB instance with no standby DB instances.

As for the settings, we use WeddingCatering as the DB Instance Identifier for easier management. For the credential settings, we assign a master username and master password called “admin” and “admin123” to make the database more secure.

Settings

DB instance identifier [Info](#)

Type a name for your DB instance. The name must be unique across all DB instances owned by your AWS account in the current AWS Region.

WeddingCatering

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 60 alphanumeric characters or hyphens. First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

▼ Credentials Settings

Master username [Info](#)

Type a login ID for the master user of your DB instance.

admin

1 to 16 alphanumeric characters. The first character must be a letter.

Credentials management

You can use AWS Secrets Manager or manage your master user credentials.

☐ Managed in AWS Secrets Manager - *most secure*

RDS generates a password for you and manages it throughout its lifecycle using AWS Secrets Manager.

☒ Self managed

Create your own password or have RDS create a password that you manage.

☐ Auto generate password

Amazon RDS can generate a password for you, or you can specify your own password.

Master password [Info](#)

Password strength [Weak](#)

Minimum constraints: At least 8 printable ASCII characters. Can't contain any of the following symbols: / ' " @

Confirm master password [Info](#)

2. Configure Security Group - We choose the existing VPC as the security group for easier management and also add a new rule for the Inbound Rules to allow MySQL to be accessed by the IP address.

VPC security group (firewall) [Info](#)
Choose one or more VPC security groups to allow access to your database. Make sure that the security group rules allow the appropriate incoming traffic.

☒ **Choose existing**
Choose existing VPC security groups

☐ **Create new**
Create new VPC security group

Existing VPC security groups

WeddingCatering

Availability Zone [Info](#)

RDS Proxy
RDS Proxy is a fully managed, highly available database proxy that improves application scalability, resiliency, and security.

☐ **Create an RDS Proxy** [Info](#)
RDS automatically creates an IAM role and a Secrets Manager secret for the proxy. RDS Proxy has additional costs. For more information, see [Amazon RDS Proxy pricing](#).

Certificate authority - optional [Info](#)
Using a server certificate provides an extra layer of security by validating that the connection is being made to an Amazon database. It does so by checking the server certificate that is automatically installed on all databases that you provision.

Expiry: May 26, 2061

If you don't select a certificate authority, RDS chooses one for you.

► Additional configuration

Edit inbound rules [Info](#)
Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules [Info](#)

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info	Description - optional Info	
sgr-017a69f4cb9915bb9	HTTPS	TCP	443	Custom	<input type="text" value="0.0.0.0"/>	<input type="button" value="Delete"/>
sgr-0108a128b50640c68	SSH	TCP	22	Custom	<input type="text" value="0.0.0.0"/>	<input type="button" value="Delete"/>
sgr-0479cfc74aea36c9c	HTTP	TCP	80	Custom	<input type="text" value="0.0.0.0"/>	<input type="button" value="Delete"/>
-	MySQL/Aurora	TCP	3306	Anywhere-IPv4	<input type="text" value="0.0.0.0"/>	<input type="button" value="Delete"/>

Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

- 3. Database Endpoint** - The database endpoint is used to connect the application to the database;
"weddingcatering.cpffhyoagrni.us-east-1.rds.amazonaws.com"

- **Connecting EC2 and RDS**

- 1. Update Database Configuration** - To ensure the application can connect the RDS instance, we need to modify the "database.php" file to use the RDS endpoint, master username and master password. We can edit the php file by using "sudo nano database.php" and change the details needed.

```
[ec2-user@ip-10-0-24-185 html]$ ls
add_staff.php      database.php      index.php  logout.php  update_staff.php  wedding_staff.php
catering_staff.php delete_staff.php  login.php  style.css   user_registration.sql
[ec2-user@ip-10-0-24-185 html]$ sudo nano database.php
```

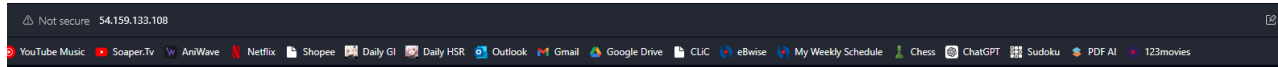
```
GNU nano 5.8 database.php
<?php

$hostName = "weddingcatering.cpffhyoagrni.us-east-1.rds.amazonaws.com";
$dbUser = "admin";
$dbPassword = "admin123";
$dbName = "user_registration";
$conn = mysqli_connect($hostName, $dbUser, $dbPassword, $dbName);
if (!$conn) {
    die("Something went wrong;");
}

?>
```


- **Testing**

1. **HTTP (Slightly work)** - We test the web application by entering "<http://ip-address>" in the browser. We manage to enter application.



However, when we clicked the functions which will bring us to another php file, it broke and just shows the code inside the php file.

A screenshot of a web browser window displaying PHP code. The address bar shows 'Not secure 54.159.133.108/add_staff.php'. The browser's tab bar contains various tabs including YouTube, YouTube Music, Soaper.Tv, AniWave, Netflix, Shopee, Daily GI, Daily HSR, Outlook, and Gmail. The main content area displays the following PHP code:

```
<?php
session_start();
if (!isset($_SESSION["user"]) || $_SESSION["role"] !== "Admin") {
    header("Location: index.php");
    exit; // Ensure to exit after redirection
}

// Include your database connection file
require_once "database.php";

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = $_POST["username"];
    $contact = $_POST["contact"];
    $age = $_POST["age"];
    $salary_plain = $_POST["salary"];
    $duty = $_POST["duty"];
    $role = $_POST["role"];
```

2. **HTTPS (Error)** - We also tested using "<https://ip-address>" but that entirely didn't let us enter the web application.

