UNIVERSITI KUALA LUMPUR

MALAYSIAN INSTITUTE OF INFORMATION TECHNOLOGY

Section A: Submission Details

PROGRAMME : BACHELOR IN INFORMATION TECHNOLOGY (HONS) (INTERNET OF THINGS)

COURSE CODE & NAME : IIB43203 - CLOUD COMPUTING

ASSESSMENT : GROUP PROJECT REPORT

DATELINE : 9/1/2026

Section B: Academic Integrity

Tick (√) each box below if you agree:

☑ I have read and understood the UniKL's policy on Plagiarism in University Rules & Regulations.
☑ This submission is my own, unless indicated with proper referencing.
☑ This submission has not been previously submitted or published.
☑ This submission follows the requirements stated in the course.

Section C: Submission Detail

No	Student Name(s)	Student ID(s)
1	HILMI HAIKAL BIN HASAN	52227124091
2	NURISYAM BINTI ABDULLAH	52224224283
3	SHAMIMI BINTI SHAIFUL AFFANDI	52224224046
4	NOOR FATIN AQILAH BINTI AMAT ROSLAN	52224224233

Student Receipt of Submission

The submission receipt of the student task will be based on the **VLE submission date and time**.

AWS Meal Ordering Website

1. INTRODUCTION

1.1 Project Background

Organizations are depending more on cloud platforms to host dependable online applications due to the quick development of cloud computing applications. Amazon Web Services (AWS) is one of the leading cloud service providers that offer a wide range of managed services that make application deployment easier

This project focuses on the development and deployment of a cloud-based Meal Ordering Website using AWS services. The system retrieves food photos from cloud object storage, stores order information in a managed database, and allows users to place meal orders via a web interface. The project replicates a real-world cloud deployment scenario

1.2 Project Objectives

- To deploy a web application using Amazon EC2
- To store structured data using Amazon RDS (MariaDB 10.5)
- To manage media files using Amazon S3
- To demonstrate teamwork in cloud system deployment
- To provide a beginner friendly guide

1.3 Scope of the Project

The scope of this project includes:

- Cloud infrastructure setup (VPC, EC2, Security Groups)
- Database creation and configuration using RDS MariaDB 10.5
- Object storage configuration using Amazon S3
- Web application development using PHP
- System integration and testing of all AWS components

2. SYSTEM ARCHITECTURE OVERVIEW

2.1 Architecture Description

The system consists of an EC2 instance hosting an Apache web server with PHP, an RDS MariaDB database for storing orders, an S3 bucket for meal images, and a VPC with security groups controlling communication.

Amazon EC2 Instance

- Hosts the Apache web server and PHP application
- Handles user requests through a web browser

Amazon RDS (MariaDB 10.5)

- Stores customer order information securely
- Managed database service for reliability and scalability

Amazon S3 Bucket

- Stores meal images
- Enables fast and reliable image retrieval

VPC & Security Groups

- Provides network isolation
- Controls secure communication between EC2, RDS, and S3

2.2 Architecture Flow

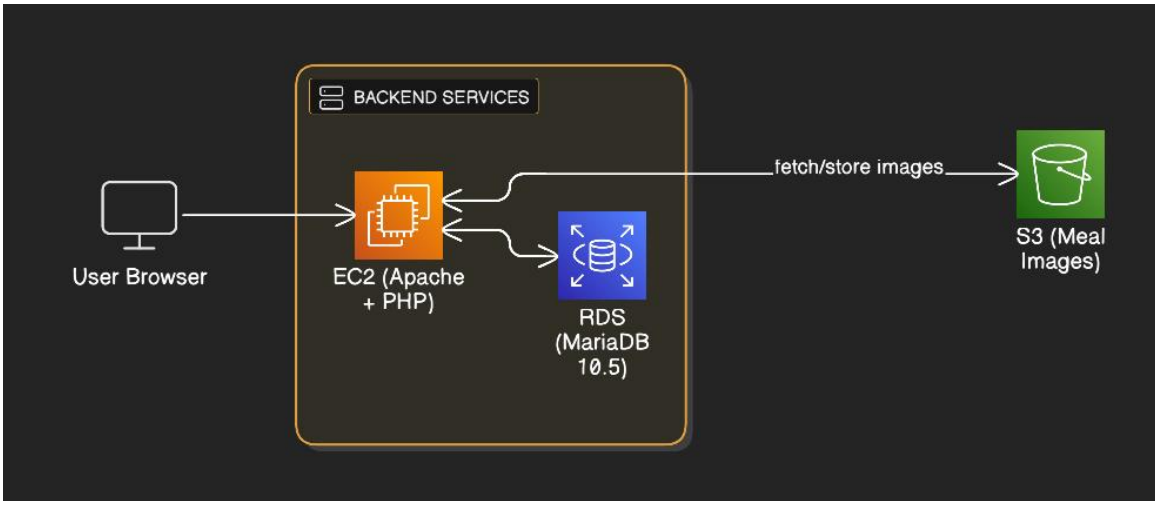


Figure 1.1 Simplified architecture of the AWS Meal Ordering Website showing EC2, RDS, and S3 integration inside the VPC

3. TEAMWORK & ROLE DISTRIBUTION

Team Member	Role	Responsibility
Nurisyam	Infrastructure Engineer	VPC & EC2 Setup
Aqilah	Database Administrator	RDS MariaDB Configuration
Shamimi	Storage Engineer	S3 Bucket & Image Hosting
Hilmi	Application Developer	Website Development & Testing

Youtube link: https://youtu.be/GC2uiezSvTA?si=flU5GV-7GhhyRS4T

4. PROJECT IMPLEMENTATION

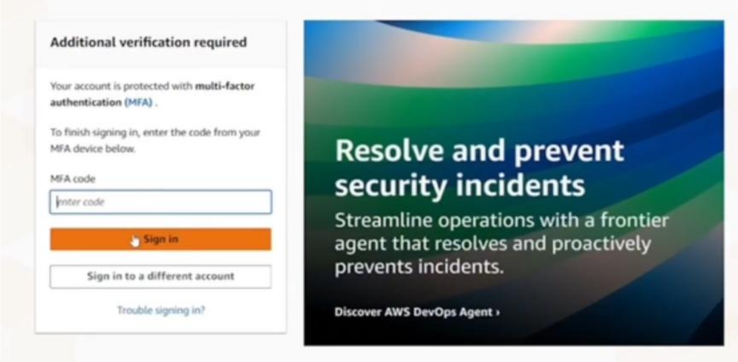
4.1 Create VPC

Purpose

A Virtual Private Cloud (VPC) is required to allow secure communication between EC2 and RDS within the AWS environment.

Steps

1. Login to the AWS Management Console



Additional verification required

Your account is protected with **multi-factor authentication** (MFA).

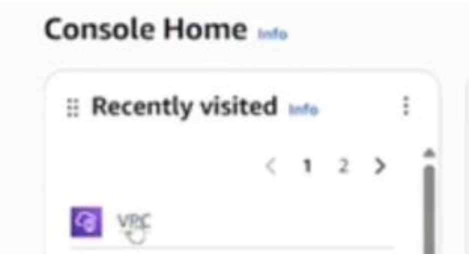
To finish signing in, enter the code from your MFA device below.

MFA code: Enter code

Resolve and prevent security incidents

Streamline operations with a frontier agent that resolves and proactively prevents incidents.

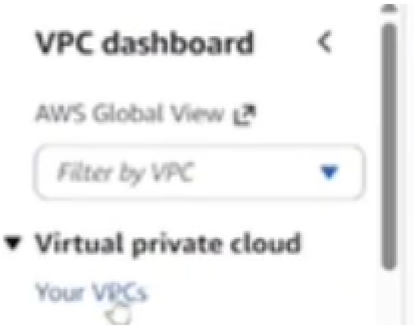
2. Search for VPC



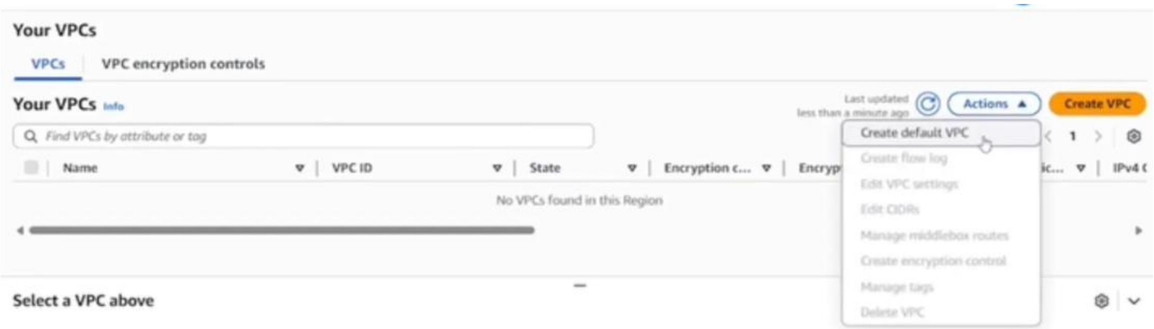
Console Home Info

Recently visited Info

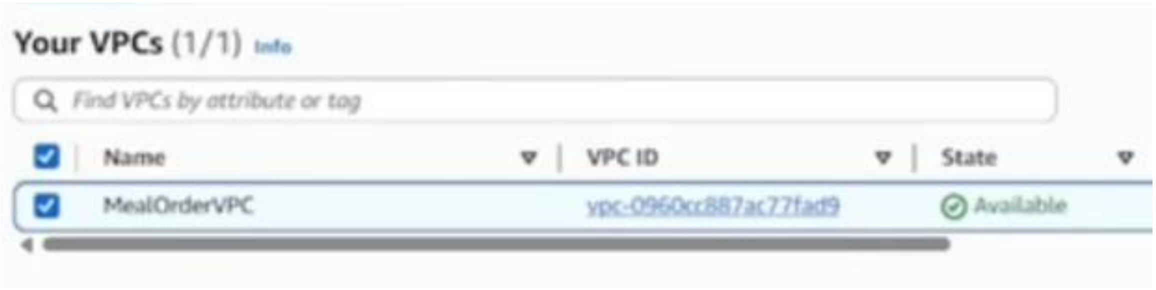
3. Click Your VPCs



4. Click Actions then create default VPC



5. Wait until the VPC status shows Available



☑	Name	VPC ID	State
☑	MealOrderVPC	vpc-0960cc887ac77fad9	✓ Available

4.2 Create EC2 Instance (Web Server)

Purpose

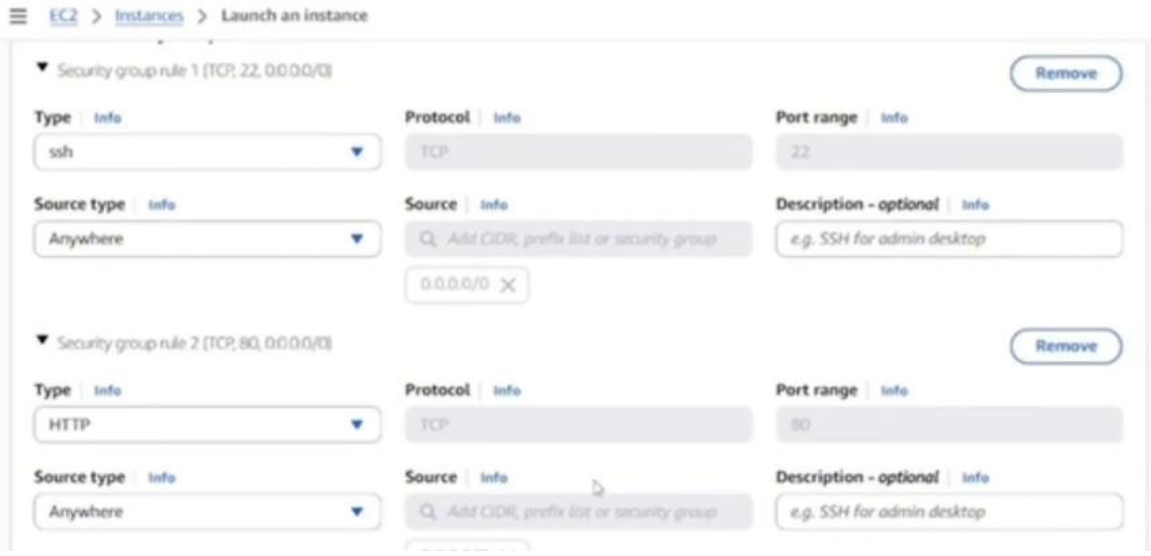
The EC2 instance hosts the web server and processes user requests.

Configuration

- Instance Name: MealOrderEC2
- AMI: Amazon Linux 2023
- Instance Type: t2.micro (Free Tier eligible)
- Key Pair: meal-key.pem

Security Group Rule

Type	Port	Source
SSH	22	My IP
HTTP	80	Anywhere

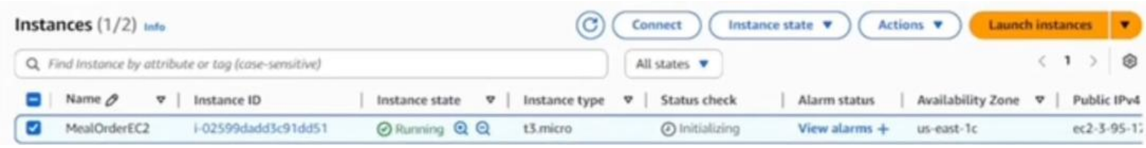


▼ Security group rule 1 (TCP, 22, 0.0.0.0/0)

Type: ssh
Protocol: TCP
Port range: 22
Source type: Anywhere
Source: Add CIDR, prefix list or security group
0.0.0.0/0
Description - optional: e.g. SSH for admin desktop

▼ Security group rule 2 (TCP, 80, 0.0.0.0/0)

Type: HTTP
Protocol: TCP
Port range: 80
Source type: Anywhere
Source: Add CIDR, prefix list or security group
Description - optional: e.g. SSH for admin desktop



☐	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
☑	MealOrderEC2	i-02599dadd3c91dd51	Running	t3.micro	Initializing	View alarms +	us-east-1c	ec2-3-95-1

Figure 1.2: MealOrderEC2 is running

Steps

1. Connect to EC2 (Using SSH)

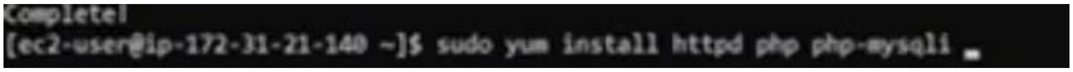
```
cd C:/Users/cryto/Documents/AWS

ssh -i C:/Users/cryto/Documents/AWS/meal-key.pem ec2-user@3.95.174.39
```

Successful connection will display:

```
[ec2-user@ip-xx-xx-xx-xx ~]$
```

2. Install APACHE & PHP



```
Complete!
[ec2-user@ip-172-31-21-140 ~]$ sudo yum install httpd php php-mysqli
```

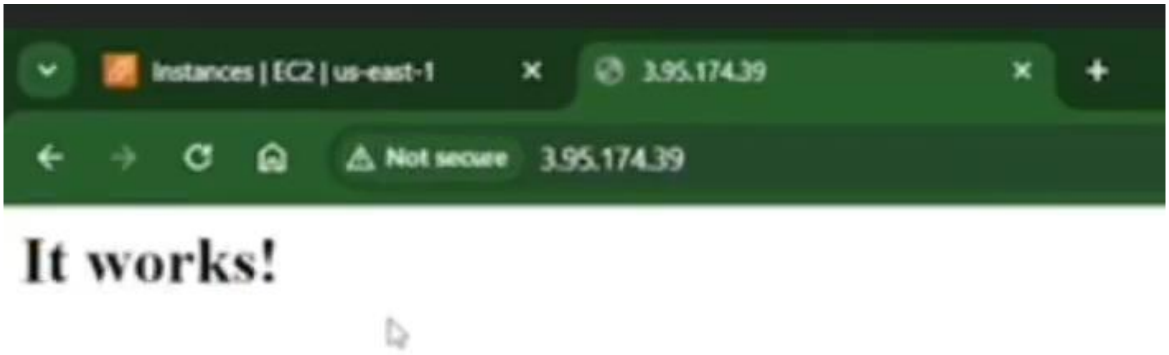
```
sudo yum update -y

sudo yum install httpd php php-mysqli -y

sudo systemctl start httpd

sudo systemctl enable httpd
```

3. Test Web Server



It works!

4.3 Create S3 Bucket (Image and Storage)

Purpose

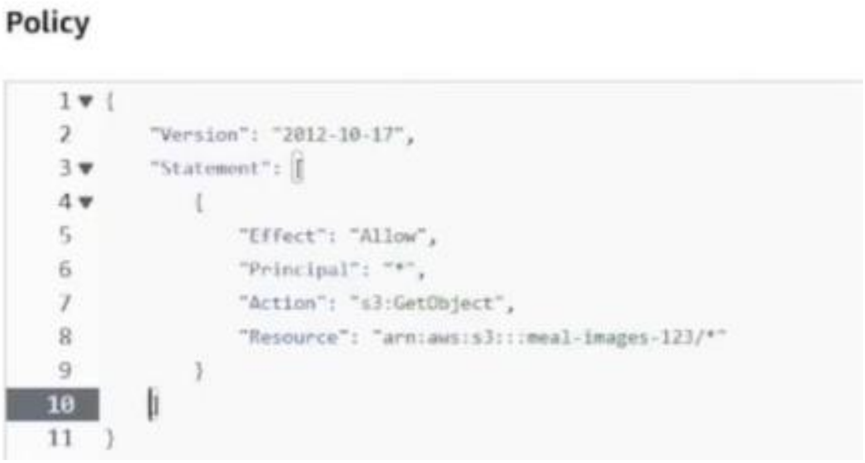
Amazon S3 stores meal images separately from the EC2 server.

Steps

- Bucket Name: meal-images-123
- Disable Block all public access
- Upload:
    - pizza.jpg
    - burger.jpg

Bucket Policy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::meal-images-123/*"
    }
  ]
}
```



```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::meal-images-123/*"
        }

}
```

4.4 Create RDS (MariaDB 10.5)

Configuration

- Engine: MariaDB 10.5
- DB Identifier: mealdb
- Username: admin
- Password: SecurePass!123
- Public Access: No
- Port: 3306 (EC2 access only)

RDS Endpoint

```
mealdb.c8xa824ic05d.us-east-1.rds.amazonaws.com
```

4.4 Connect EC2 to RDS

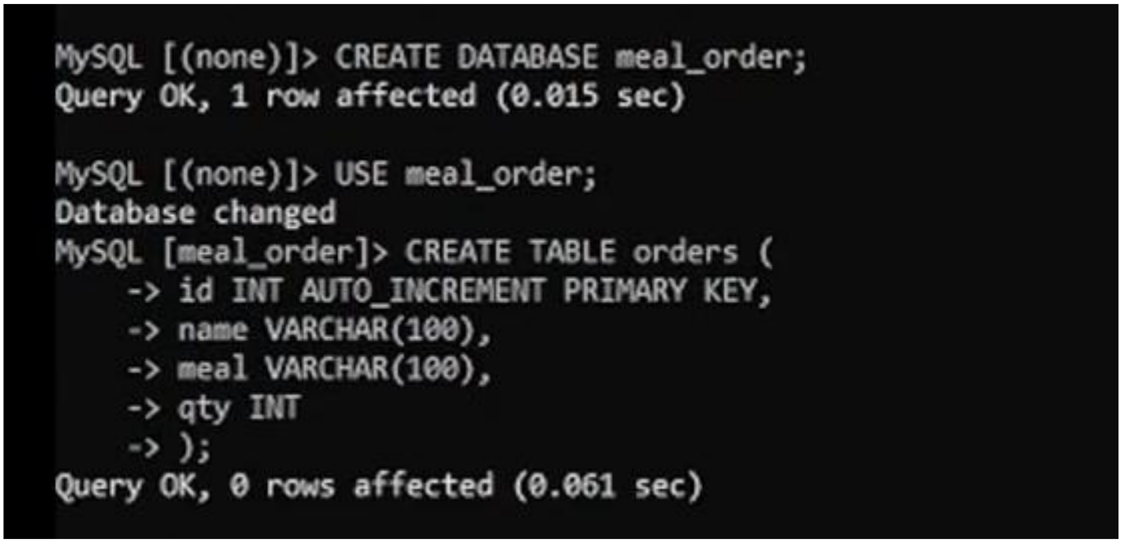
Install MariaDB client:

```
sudo yum install mariadb105 -y
```

Connect to database:

```
mariadb -h mealdb.c8xa824ic05d.us-east-1.rds.amazonaws.com -u admin -p
```

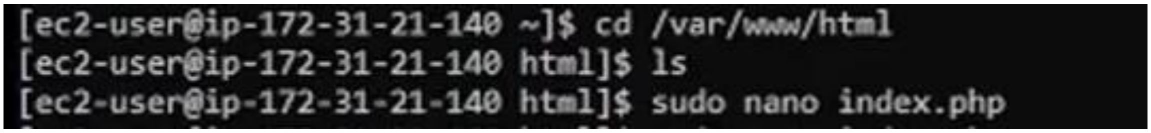
Create database and table:



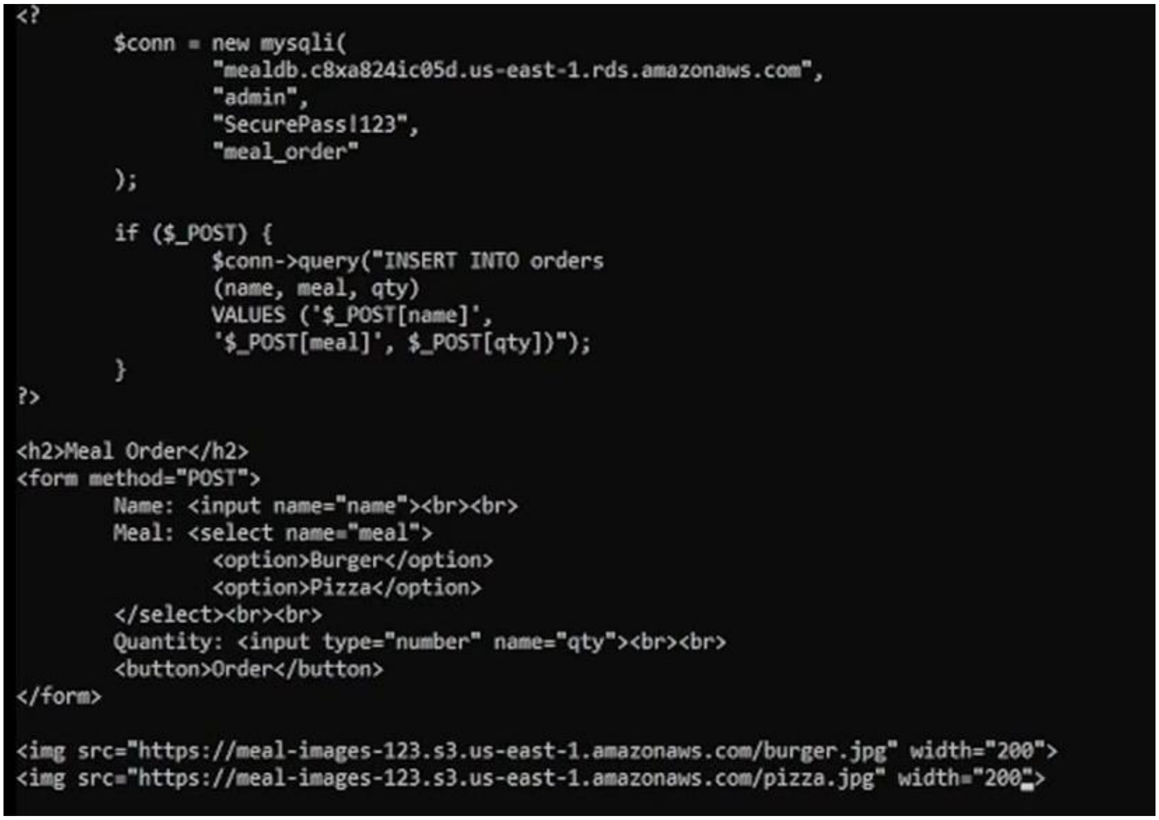
```
MySQL [(none)]> CREATE DATABASE meal_order;
Query OK, 1 row affected (0.015 sec)

MySQL [(none)]> USE meal_order;
Database changed
MySQL [meal_order]> CREATE TABLE orders (
    -> id INT AUTO_INCREMENT PRIMARY KEY,
    -> name VARCHAR(100),
    -> meal VARCHAR(100),
    -> qty INT
    -> );
Query OK, 0 rows affected (0.061 sec)
```

4.5 Create Website Application



```
[ec2-user@ip-172-31-21-140 ~]$ cd /var/www/html
[ec2-user@ip-172-31-21-140 html]$ ls
[ec2-user@ip-172-31-21-140 html]$ sudo nano index.php
```



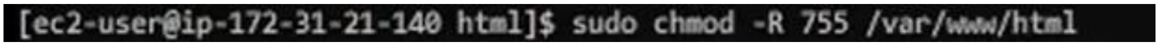
```
<?
        $conn = new mysqli(
                "mealdb.c8xa824ic05d.us-east-1.rds.amazonaws.com",
                "admin",
                "SecurePass!123",
                "meal_order"
        );

        if ($_POST) {
                $conn->query("INSERT INTO orders
                (name, meal, qty)
                VALUES ('$_POST[name]',
                '$_POST[meal]', $_POST[qty])");
        }
?>

<h2>Meal Order</h2>
<form method="POST">
        Name: <input name="name"><br><br>
        Meal: <select name="meal">
                <option>Burger</option>
                <option>Pizza</option>
        </select><br><br>
        Quantity: <input type="number" name="qty"><br><br>
        <button>Order</button>
</form>

<img src="https://meal-images-123.s3.us-east-1.amazonaws.com/burger.jpg" width="200">
<img src="https://meal-images-123.s3.us-east-1.amazonaws.com/pizza.jpg" width="200">
```

Set permissions:



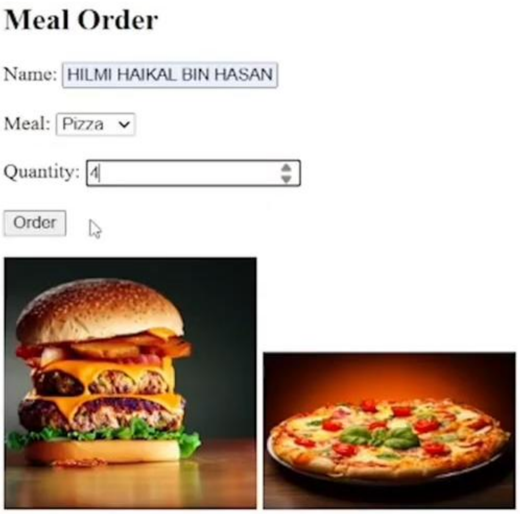
```
[ec2-user@ip-172-31-21-140 html]$ sudo chmod -R 755 /var/www/html
```

4.6 Final Testing & Verification

Open browser:

```
http://3.95.174.39/index.php
```

Verify database records:

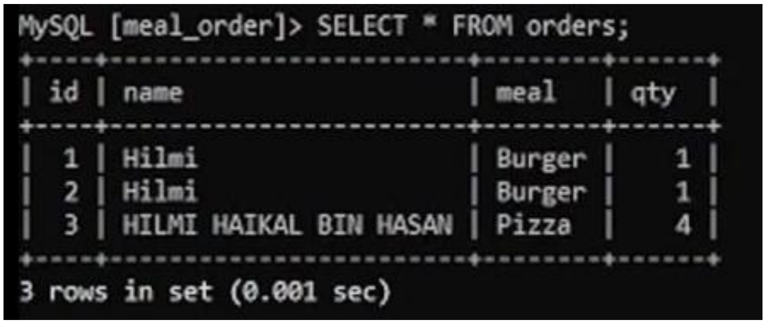


Meal Order

Name: HILMI HAIKAL BIN HASAN

Meal: Pizza

Quantity: 4



```
MySQL [meal_order]> SELECT * FROM orders;
+----+-----------------------+--------+-----+
| id | name                  | meal   | qty |
+----+-----------------------+--------+-----+
|  1 | Hilmi                 | Burger |   1 |
|  2 | Hilmi                 | Burger |   1 |
|  3 | HILMI HAIKAL BIN HASAN | Pizza  |   4 |
+----+-----------------------+--------+-----+
3 rows in set (0.001 sec)
```

5. OUTCOME & TEAMWORK RESULTS

The project successfully delivered a fully functional AWS based meal ordering system with proper teamwork and service integration.

6. CONCLUSION

This project successfully demonstrates the integration of several AWS cloud services to produce a safe, scalable, and effective web-based application