

VPC 2-Tier Architecture: Hosting a College Form Page with LEMP Stack

. Overview

In this project, we set up a two-tier architecture on AWS using a custom VPC. The architecture consists of:

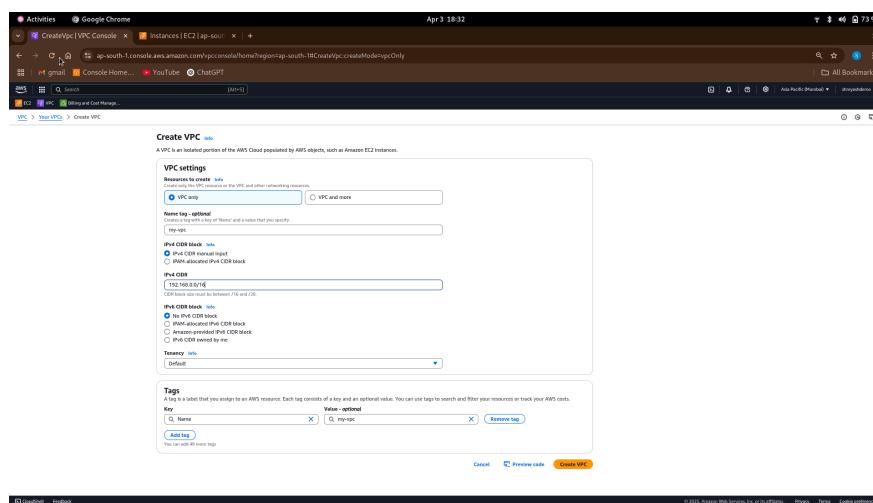
- A Web Server (public subnet) hosting a PHP form.
- A Database Server (private subnet) running MariaDB.
- Secure networking components: VPC, subnets, route tables, Internet Gateway (IGW), and NAT Gateway.

. VPC and Networking Setup

Step 1: Create a Custom VPC

- Go to AWS Management Console → VPC Service → Create VPC.
- Set Name: `my-vpc`.
- Set CIDR Block: `10.0.0.0/16`.
- Click Create VPC.

📌 Screenshot:



The screenshot shows the AWS VPC console interface. On the left, there's a navigation sidebar with options like EC2 Global View, Filter by VPC, Virtual private cloud, Security, and PrivateLink & Lambda. The main area is titled 'Your VPCs (1/2)'. It lists one VPC: 'my-vpc' (VPC ID: vpc-048ac18cdb202b214, State: Available, CIDR: 192.168.0.0/16). Below this, there are tabs for Details, Resource map, CIDs, Flow logs, Tags, and Integrations. The 'Details' tab is active, displaying the VPC's configuration.

Step 2: Create Subnets

- Web Subnet
 - CIDR Block: 10.0.1.0/24
 - Select my-vpc.
 - Enable Auto-assign public IP.
- Database Subnet
 - CIDR Block: 10.0.2.0/24
 - Select my-vpc.
 - No auto-assign public IP.

❤️ Screenshots:

This screenshot shows the 'Create subnet' wizard in the AWS VPC console. The first step, 'Subnet settings', is displayed. It requires specifying the CIDR block for the subnet. The VPC ID is set to 'vpc-048ac18cdb202b214'. A single subnet is being created with the name 'web-subnet' and a CIDR of '192.168.1.0/24'. The 'Associate with existing VPC' checkbox is checked. The 'Auto-assign public IP' checkbox is also checked. At the bottom, there are 'Next Step' and 'Cancel' buttons.

The screenshot shows the AWS VPC Subnets page. The left sidebar includes sections for EC2 Global View, Virtual private cloud, Subnets, Route tables, Internet gateways, Security, PrivateLink, and Lattice. The main content area displays a table of subnets:

Name	Subnet ID	Status	VPC	Block Public Access	IPv4 CIDR	IPv6 CIDR	Actions
db-subnet	subnet-002c95a5462a752081	Available	vpc-046ac18db202b214	Off	192.168.16.0/24	-	Create subnet
-	subnet-0415b377f1b513c1	Available	vpc-056ae95d95c03baed	Off	172.31.16.0/23	-	
-	subnet-0890e49b015b02774	Available	vpc-056ae95d95c03baed	Off	172.31.32.0/23	-	
web-subnet	subnet-04f90a3c30403000	Available	vpc-046ac18db202b214	Off	192.168.0/23	-	
-	subnet-0015520802090230c	Available	vpc-066ac79e5c03baed	Off	172.31.0/23	-	

Below the table, a detailed view for the **subnet-046f804c3504b308c / web-subnet** is shown. It includes tabs for Details, Flow logs, Route table, Network ACL, CIDR reservations, Sharing, and Tags. The Details section contains information such as Subnet ID, Subnet ARN, Availability Zone, IPv4 CIDR, and Route table.

Step 3: Create Route Table and Attach to Web Subnet

- Go to Route Tables → Create Route Table.
- Set Name: `my-rt2`.
- Attach to `my-vpc`.
- Select `web-subnet` and associate it.

📌 Screenshots:

The screenshot shows the AWS Create Route Table page. The top navigation bar includes links for VPC, Route tables, and Create route table. The main content area has two main sections: Route table settings and Tags.

Route table settings section:

- Name: `my-rt`
- VPC: `my-vpc`
- Target: `vpc-046ac18db202b214 (my-vpc)`

Tags section:

- A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.
- Key: `Name`, Value: `my-rt`
- Add new tag button

At the bottom right are **Create route table** and **Cancel** buttons.

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
db-subnet	subnet-02b4e5ab27a752603	192.168.10.0/24	-	Main (rt-0277af7800033723a / my-vt)
web-subnet	subnet-046f7004c7504b20c	192.168.0.0/20	-	rtb-0x1d9022cbef0fd1c2 / my-vt-2

Step 4: Create an Internet Gateway

- Go to Internet Gateways → Create IGW.
- Name it `my-igw`.
- Attach it to `my-vpc`.
- Edit Route Table and add:
 - Destination: `0.0.0.0/0`
 - Target: `my-igw`

👉 Screenshots:

Name	Internet gateway ID	State	VPC ID	Owner
Main IGW	rti-02b4e5ab27a752603	Attached	rtb-005ea726ed3b0bed1 / main-VPC	943143228547
my-igw	rti-0185f91924603174	Attached	rtb-046f7004c7504b20c / my-vt	943143228547

The screenshot shows the 'Edit routes' section of the AWS VPC Route Tables page. A new route is being added for destination 192.168.0.0/16, with the target set to 'Internet Gateway' and the status set to 'Active'. The route is not propagated. The 'Save changes' button is highlighted.

The screenshot shows the 'Subnets (1/5)' section of the AWS VPC Subnets page. A success message indicates that subnet settings have been changed. The 'Actions' dropdown for the selected 'web-subnet' is open, showing options like 'Edit', 'Delete', and 'Create subnet copy'.

Step 5: Create a NAT Gateway

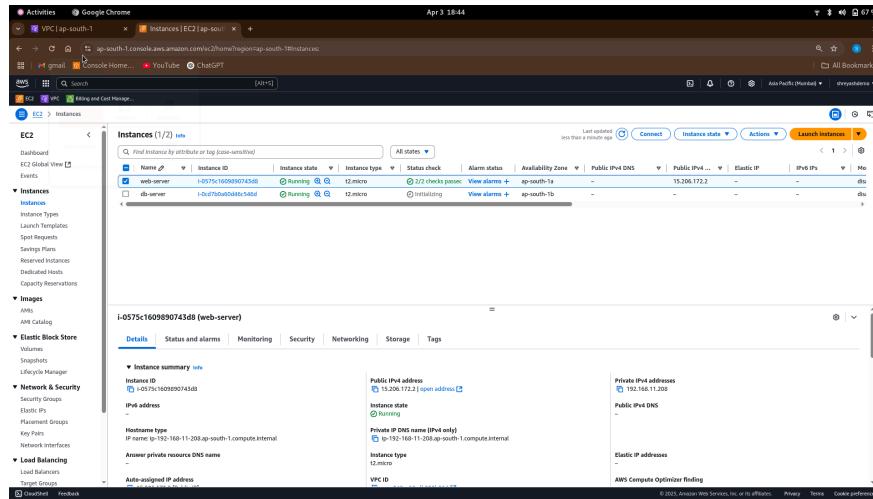
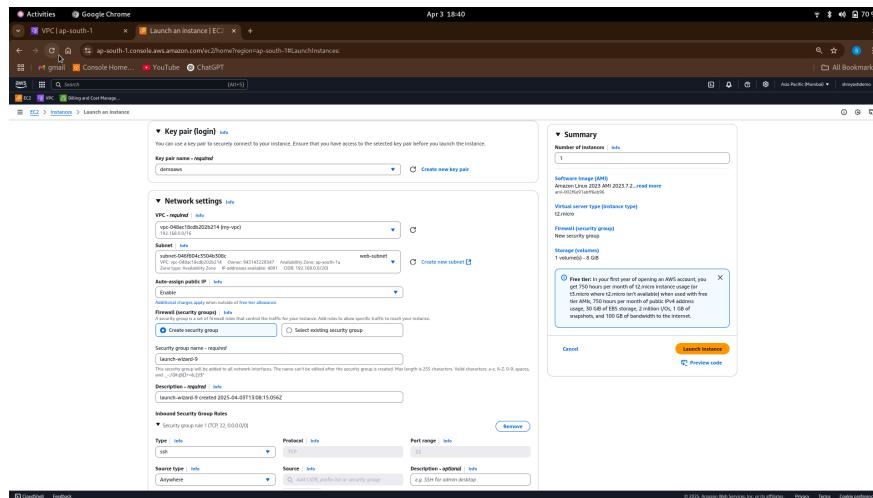
- Go to NAT Gateways → Create NAT Gateway.
- Attach it to `web-subnet`.
- Allocate an Elastic IP.
- Create a new Route Table for `db-subnet`:
 - Destination: `0.0.0.0/0`
 - Target: `NAT Gateway`
 - Do NOT use an Internet Gateway.

. Launching EC2 Instances

Step 6: Create Web and Database Servers

- Web Server
 - Ubuntu 22.04, in `web-subnet` (public)
 - Enable Auto-assign Public IP.
- Database Server
 - Ubuntu 22.04, in `db-subnet` (private)
 - No Public IP.

📌 Screenshot:



Transferring Files and Installing LEMP Stack

Step 7: Send Key Pair & LEMP Script to Web Server



Step 8: Connect to Web Server & Install LEMP Stack

- **LEMP Setup:**
 - Install Nginx, PHP, MySQL Client (not MariaDB)
 - Start and enable Nginx & PHP
 -  Screenshot:

	Architecture	Version	Repository	Size
Packages:				
install-nginx-server	x86_64	3:10.5-25.1.amzn2023.0.1	amazonlinux	11 M
nginx	x86_64	1:1.16.26-1.amzn2023.0.1	amazonlinux	33 k
nginx-ingress	x86_64	8.4.5-1.amzn2023.0.1	amazonlinux	17 k
Installing dependencies:				
apr	x86_64	1.7.1-0.1.amzn2023.0.4	amazonlinux	129 k
apr-util	x86_64	1.6.1-0.1.amzn2023.0.4	amazonlinux	93 k
generic-logos-httpsd	x86_64	n/a	amazonlinux	39 k
openssl-libs	x86_64	18.0.12-0.2.amzn2023.0.3	amazonlinux	200 k
openssl-tls	x86_64	2.5.1-0.1.amzn2023.0.3	amazonlinux	1.4 M
httpd-core	x86_64	2.4.6-0.2.amzn2023	amazonlinux	14 k
httpd-fs-filesystem	x86_64	2.4.6-0.2.amzn2023	amazonlinux	0.1 k
httpd-tools	x86_64	2.4.6-0.2.amzn2023	amazonlinux	0.1 k
libbrotli	x86_64	1.0.5-0.4.amzn2023.0.2	amazonlinux	315 k
libodium	x86_64	1.0.19-0.4.amzn2023	amazonlinux	176 k
libxml2	x86_64	2.9.10-0.1.amzn2023.0.2	amazonlinux	155 k
libxslt	x86_64	1.1.34-0.1.amzn2023.0.2	amazonlinux	183 k
mailcap	x86_64	1.1.4-0.1.amzn2023.0.3	amazonlinux	211 k
mariadb-connector-c	x86_64	3.1.0-0.1.amzn2023.0.1	amazonlinux	9.9 k
mariadb-connector-c-config	x86_64	n/a	amazonlinux	1.6 k
mariadb105-common	x86_64	3:10.5.25-1.amzn2023.0.1	amazonlinux	29 k
mariadb105-ermsg	x86_64	3:10.5.25-1.amzn2023.0.1	amazonlinux	213 k
mariadb105-thin	x86_64	3:10.5.25-1.amzn2023.0.1	amazonlinux	39 k
nginx	x86_64	1.16.26-1.amzn2023.0.1	amazonlinux	678 k
nginx-fs-filesystem	x86_64	1:1.16.26-1.amzn2023.0.1	amazonlinux	97 k
nginx-ingress-types	x86_64	1.16.26-1.amzn2023.0.1	amazonlinux	21 k
perl-B	x86_64	1.88.477-0.2.amzn2023.0.6	amazonlinux	153 k
perl-Crypt-RSA	x86_64	1.4647-7.amzn2023.0.3	amazonlinux	709 k
perl-DBI	x86_64	2.174.468.amzn2023.0.8	amazonlinux	55 k
perl-Dumper	x86_64	2.33.1-0.1.amzn2023.0.1	amazonlinux	29 k
perl-File-Copy	x86_64	2.20.0-0.1.amzn2023.0.1	amazonlinux	16 k
perl-File-Footer	x86_64	2.03.477.amzn2023.0.6	amazonlinux	202 k
perl-Math-BigInt	x86_64	1.11.13-0.1.amzn2023.0.2	amazonlinux	39 k
perl-Math-BigRat	x86_64	0.09.04-0.1.amzn2023.0.2	amazonlinux	47 k
perl-Math-Complex	x86_64	1.59.477.amzn2023.0.6	amazonlinux	10 k
perl-Sys-Hostname	x86_64	1.33.0-0.1.amzn2023.0.6	amazonlinux	17 k
perl-Time-Local	x86_64	2.27.477.amzn2023.0.6	amazonlinux	17 k
php80-4-clis	x86_64	8.4.5-1.amzn2023.0.1	amazonlinux	3.8 M
php80-common	x86_64	8.4.5-1.amzn2023.0.1	amazonlinux	703 k
php80-process	x86_64	8.4.5-1.amzn2023.0.1	amazonlinux	53 k
php80-zip	x86_64	8.4.5-1.amzn2023.0.1	amazonlinux	972 k
Installing weak dependencies:				
apr-util-openssl	x86_64	1.6.3-0.1.amzn2023.0.1	amazonlinux	17 k
curl	x86_64	2.4.1-0.1.amzn2023.0.1	amazonlinux	46 k
mariadb105-backup	x86_64	3:10.5.25-1.amzn2023.0.1	amazonlinux	6.3 M
mariadb105-cracklib-password-check	x86_64	3:10.5.25-1.amzn2023.0.1	amazonlinux	0.1 k
openssl	x86_64	1.1.34-0.1.amzn2023.0.2	amazonlinux	1.6 k

```

Apr 3 18:51 Activities Terminal Apr 3 18:51
php8.4-pdo-8.4.5-1.amzn2023.0.1.x86_64 php8.4-process-8.4.5-1.amzn2023.0.1.x86_64 php8.4-sodium-8.4.5-1.amzn2023.0.1.x86_64
[ec2-user@ip-192-168-11-208 ~]$ sudo service nginx start
Redirecting to /usr/bin/systemctl start nginx.service
[ec2-user@ip-192-168-11-208 ~]$ sudo systemctl enable nginx.service
[ec2-user@ip-192-168-11-208 ~]$ sudo systemctl enable nginx.service
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service -> /usr/lib/systemd/system/nginx.service.
[ec2-user@ip-192-168-11-208 ~]$ sudo systemctl status nginx
Created symlink /etc/systemd/system/multi-user.target.wants/php-fpm.service -> /usr/lib/systemd/system/php-fpm.service.

● nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
   Active: active (running) since THU 2025-04-03 13:19:25 UTC; 1min 34s ago
     Main PID: 23524 (nginx)
        Tasks: 6 (limit: 1111)
       Memory: 2.5M
          CPU: 61ms
         CGroup: /system.slice/nginx.service
             └─ 23524 /usr/sbin/nginx

● php-fpm.service - The PHP FastCGI Process Manager
   Loaded: loaded (/usr/lib/systemd/system/php-fpm.service; enabled; preset: disabled)
   Active: active (running) since Thu 2025-04-03 13:19:25 UTC; 1min 43s ago
     Main PID: 23452 (php-fpm)
        Tasks: 6 (limit: 1111)
       Memory: 2.5M
          CPU: 57ms
         CGroup: /system.slice/php-fpm.service
             ├─ 23454 /usr/sbin/php-fpm master process (/etc/php-fpm.conf)
             ├─ 23468 /usr/sbin/php-fpm pool www*
             ├─ 23470 /usr/sbin/php-fpm pool www*
             ├─ 23471 /usr/sbin/php-fpm pool www*
             ├─ 23472 /usr/sbin/php-fpm pool www*
             └─ 23473 /usr/sbin/php-fpm pool www*

Apr 03 13:19:25 ip-192-168-11-208.ap-south-1.compute.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Apr 03 13:19:25 ip-192-168-11-208.ap-south-1.compute.internal nginx[23417]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Apr 03 13:19:25 ip-192-168-11-208.ap-south-1.compute.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy server.

[ec2-user@ip-192-168-11-208 ~]$ sudo systemctl status php-fpm
● php-fpm.service - The PHP FastCGI Process Manager
   Loaded: loaded (/usr/lib/systemd/system/php-fpm.service; enabled; preset: disabled)
   Active: active (running) since Thu 2025-04-03 13:19:25 UTC; 1min 43s ago
     Main PID: 23452 (php-fpm)
        Tasks: 6 (limit: 1111)
       Memory: 2.5M
          CPU: 57ms
         CGroup: /system.slice/php-fpm.service
             ├─ 23454 /usr/sbin/php-fpm master process (/etc/php-fpm.conf)
             ├─ 23468 /usr/sbin/php-fpm pool www*
             ├─ 23470 /usr/sbin/php-fpm pool www*
             ├─ 23471 /usr/sbin/php-fpm pool www*
             ├─ 23472 /usr/sbin/php-fpm pool www*
             └─ 23473 /usr/sbin/php-fpm pool www*

Apr 03 13:19:25 ip-192-168-11-208.ap-south-1.compute.internal systemd[1]: Starting php-fpm.service - The PHP FastCGI Process Manager...
Apr 03 13:19:25 ip-192-168-11-208.ap-south-1.compute.internal systemd[1]: Started php-fpm.service - The PHP FastCGI Process Manager.

[ec2-user@ip-192-168-11-208 ~]$

```

Step 9: Create Web Form Files

Create form.html:

Screenshot:

```

Apr 3 19:04 Activities Terminal Apr 3 19:04
[ec2-user@ip-192-168-11-208 ~]$ nano form.html
[ec2-user@ip-192-168-11-208 ~]$ cat form.html
<!DOCTYPE HTML>
<html>
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Contact Form</title>
<style>
body {
    font-family: Arial, sans-serif;
    background-color: #f4f4f4;
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
    margin: 0;
}
.container {
    background: white;
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0px 0px 10px rgba(0, 0, 0, 1);
    width: 350px;
}
.h2 {
    text-align: center;
    color: #333;
}
label {
    font-weight: bold;
    display: block;
    margin-top: 10px;
}
input, textarea {
    width: 100%;
    padding: 5px;
    margin-top: 5px;
    border: 1px solid #ccc;
    border-radius: 5px;
    font-size: 14px;
}
input:focus, textarea:focus {
    border-color: #8000ff;
    outline: none;
}
.form-group {
    display: flex;
    justify-content: space-between;
    margin-top: 10px;
}
</style>
</head>
<body>


## Contact Form


<form>
    <div class="form-group">
        <label>Name:</label>
        <input type="text" name="name" required="required" />
    </div>
    <div class="form-group">
        <label>Email:</label>
        <input type="email" name="email" required="required" />
    </div>
    <div class="form-group">
        <label>Message:</label>
        <textarea name="message" required="required" style="height: 100px;" />
    </div>
    <div style="text-align: center; margin-top: 20px;">
        <button type="submit" style="background-color: #8000ff; color: white; border: none; padding: 10px; border-radius: 5px; font-size: 16px; width: fit-content;">Submit</button>
    </div>
</form>
</div>
</body>
</html>


```

Create submit.php:



```
[Activities Terminal] Apr 3 15:18:52 2025
[ec2-user@ip-192-168-11-288 ~]$ sudo nano httpd.conf
[ec2-user@ip-192-168-11-288 ~]$ sudo service nginx reload
Redirecting to /bin/systemctl reload nginx.service
[ec2-user@ip-192-168-11-288 ~]$ sudo nano subunit.php
[ec2-user@ip-192-168-11-288 ~]$ sudo service php7.4-fpm reload
Redirecting to /bin/systemctl reload nginx.service
[ec2-user@ip-192-168-11-288 ~]$ sudo yum install httpd-mod-spdy
Last metadata expiration check: 0:23:20 on Apr 3 15:18:52 2025.
Dependencies resolved.
=====


| Package       | Architecture | Version              | Repository  | Size  |
|---------------|--------------|----------------------|-------------|-------|
| php7.4-mysqld | x86_64       | 8.4.5-1.amzn2023.0.1 | amazonlinux | 157 k |


Transaction Summary
=====
Install 1 Package

Total download size: 157 k
Is this ok? [y/n]: y
Download Packages:
php7.4-mysqld-8.4.5-1.amzn2023.8.1.x86_64.rpm ..... 1.0 MB/s | 157 kB 00:00
1.0 MB/s | 157 kB 00:00

Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing : ..... 1/1
  Enabling : php7.4-mysqld-8.4.5-1.amzn2023.0.1.x86_64 1/1
  Running script: php7.4-mysqld-8.4.5-1.amzn2023.0.1.x86_64 1/1
  Verifying : php7.4-mysqld-8.4.5-1.amzn2023.0.1.x86_64 1/1

Installed:
  php7.4-mysqld-8.4.5-1.amzn2023.8.1.x86_64

Complete!
[ec2-user@ip-192-168-11-288 ~]$
```

. Configuring Database Server

Step 10: Connect from Web Server to Database Server

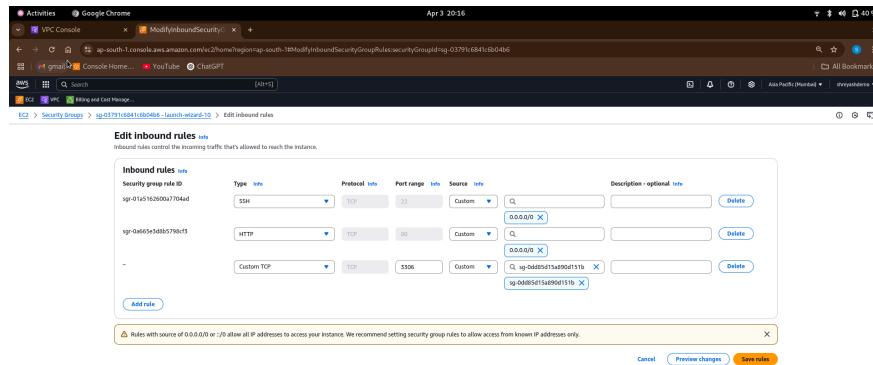
Install MariaDB on Database Server

📌 Screenshot:

The screenshot shows a terminal session on an Amazon Linux 2023 instance. The user runs `sudo dnf install mariadb105-server` and `sudo dnf install mariadb105-server-samples` to install the MariaDB server and its samples. The terminal output shows the package manager listing dependencies and installing 22 packages. The packages installed include `mariadb105-server` (version 3.10.5.25.1), `mariadb105-libs` (version 3.10.5.25.1), and various MySQL client libraries like `libmysqlclient` and `libmysqld`. The total size of the packages is approximately 210 MB.

Step 11: Add Web Server Sg to DB Server

📌 Screenshot:



Step 12: Secure and Configure MySQL



```
Activities Terminal Apr 3 20:01
ec2-user@ip-192-168-16-38:~
```

Type 'help' or '?' for help. Type 'c' to clear the current input statement.

```
MariaDB [(none)]> create user 'shreyash'@'192.168.11.208' identified by 'Pass@l23';
Query OK, 0 rows affected (0.002 sec)

MariaDB [(none)]> ^C
ERROR: No query specified

MariaDB [(none)]> alter user 'root'@'localhost' identified by 'Pass@l23';
ERROR 1664 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'by 'Pass@l23'' at line 1
MariaDB [(none)]> alter user 'root'@'localhost' identified by 'Pass@l23';
:
> ^C
Query OK, 0 rows affected (0.002 sec)

ERROR 1664 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'Pass@l23';

:~ at line 1
MariaDB [(none)]> alter user 'root'@'localhost' identified by 'Pass@l23';
Query OK, 0 rows affected (0.002 sec)

MariaDB [(none)]> create database mydb;
Query OK, 1 row affected (0.000 sec)

MariaDB [(none)]> grant all privileges on mydb.* to 'shreyash'@'192.168.11.208';
Query OK, 0 rows affected (0.002 sec) - waiting for privilege refresh in the user table

MariaDB [(none)]> grant all privileges on mybd.* to 'shreyash'@'92.168.11.208';
ERROR 1133 (28000): Cannot find any matching user for user name 'mybd'

MariaDB [(none)]> CREATE USER 'shreyash'@'192.168.11.208' IDENTIFIED BY 'your_password';
Query OK, 0 rows affected (0.002 sec)

MariaDB [(none)]> grant all privileges on mybd.* to 'shreyash'@'92.168.11.208';
Query OK, 0 rows affected (0.002 sec)

MariaDB [(none)]> CREATE USER 'shreyash'@'192.168.11.208' IDENTIFIED BY 'Pass@l23';
ERROR 1396 (HY000): Operation CREATE USER failed for 'shreyash'@'192.168.11.208'
Query OK, 0 rows affected (0.001 sec) - waiting for privilege refresh in the user table

MariaDB [(none)]> grant all privileges on mybd.* to 'shreyash'@'192.168.11.208';
Query OK, 0 rows affected (0.001 sec)

MariaDB [(none)]> create table student(id int primary key auto_increment, name varchar(100), email varchar(100), website varchar(100), comment varchar(100), gender varchar(100));
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> show tables;
Tables: changed

MariaDB [(mydb)]> create table student(id int primary key auto_increment, name varchar(100), email varchar(100), website varchar(100), comment varchar(100), gender varchar(100));
Query OK, 0 rows affected (0.001 sec)

MariaDB [(mydb)]>
```

. Final Validation and Testing

Step 15: Access the Web Form

- Open the browser and go to: <http://web-server-public-ip/form.html>
 - Submit the form and check if data is stored in mydb.



A screenshot of a web browser showing a contact form. The browser's address bar displays the URL: '15.206.172.21form.html'. The title bar of the browser window says 'Contact Form'. The form itself has a dark background with white text and input fields. It includes fields for Name, E-mail, Website, Comment, Gender (with options Female, Male, and Other), and a blue 'Submit' button at the bottom.



Step 16: Verify Data in Database

📌 Screenshot:

```

Activities Terminal Apr 3 20:24 ec2-user@ip-192-168-11-208-
[ec2-user@ip-192-168-11-208 ~]$ sudo mysql -u shreyash -p -h192.168.16.38
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 10.5.25-MariaDB MariaDB Server
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
MariaDB [(none)]> show database;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'database' at line 1
MariaDB [(none)]> select database;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
+-----+
2 rows in set (0.00 sec)

MariaDB [(none)]> use mydb;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
Database changed
MariaDB [mydb]> select * from student;
+----+-----+-----+-----+-----+
| id | name | email | website | comment | gender |
+----+-----+-----+-----+-----+
| 1 | shreyash | shreyash@gmail.com | Hellooooo | male |
| 2 | shreyash | shreyash@gmail.com | Hellooooo | male |
| 3 | rohit | shreyash@gmail.com | Hellooooo | male |
+----+-----+-----+-----+-----+
3 rows in set (0.001 sec)

MariaDB [mydb]>

```

. Conclusion

In this project, we establish a two-tier architecture on AWS using a custom VPC to host a college form page powered by the LEMP stack. The architecture includes a public subnet for the web server running PHP and a private subnet for the database server with MariaDB. Key steps involve setting up secure networking components, launching EC2 instances, transferring files, and installing necessary software. We conclude by validating the setup through form submission and database verification.