Develop and Deploy Web Application for Amazon Native Database CRUD Operation (LAB-M05-01)

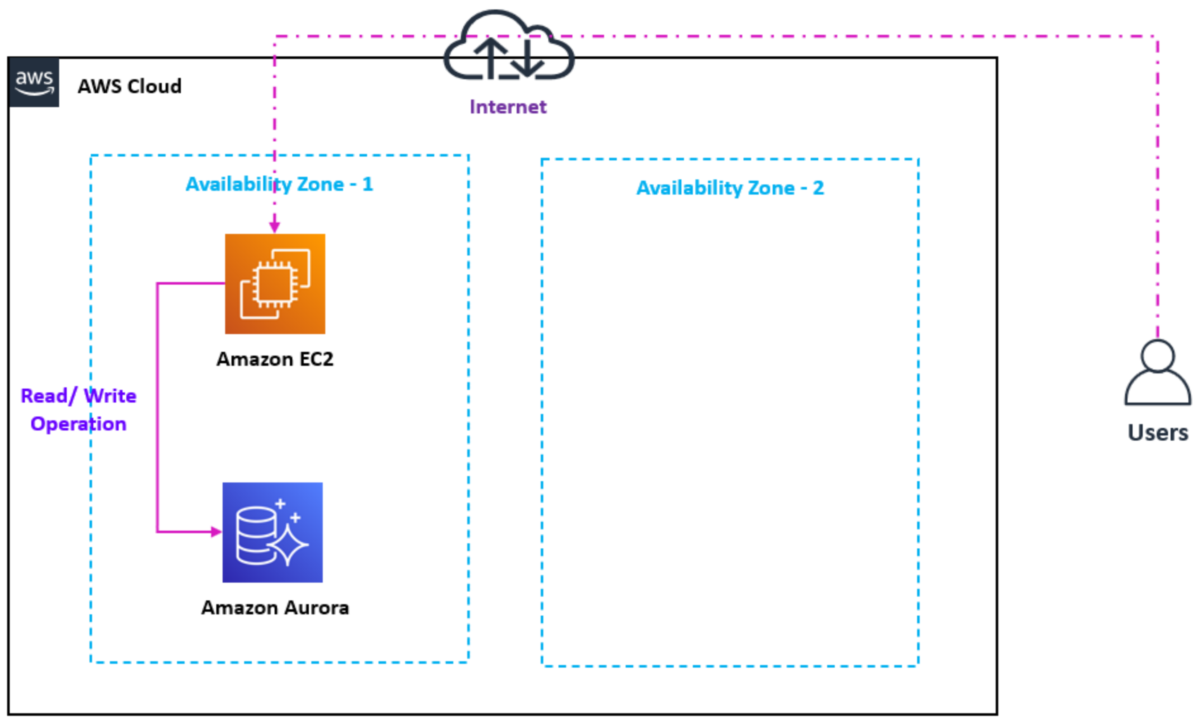
Lab scenario

You're preparing to host a Products web application in AWS that uses to store product details in database. As a development group, your team has decided to host web application in AWS Virtual machine and database in AWS RDS.You also want to explore the AWS Native database for your services.

Objectives

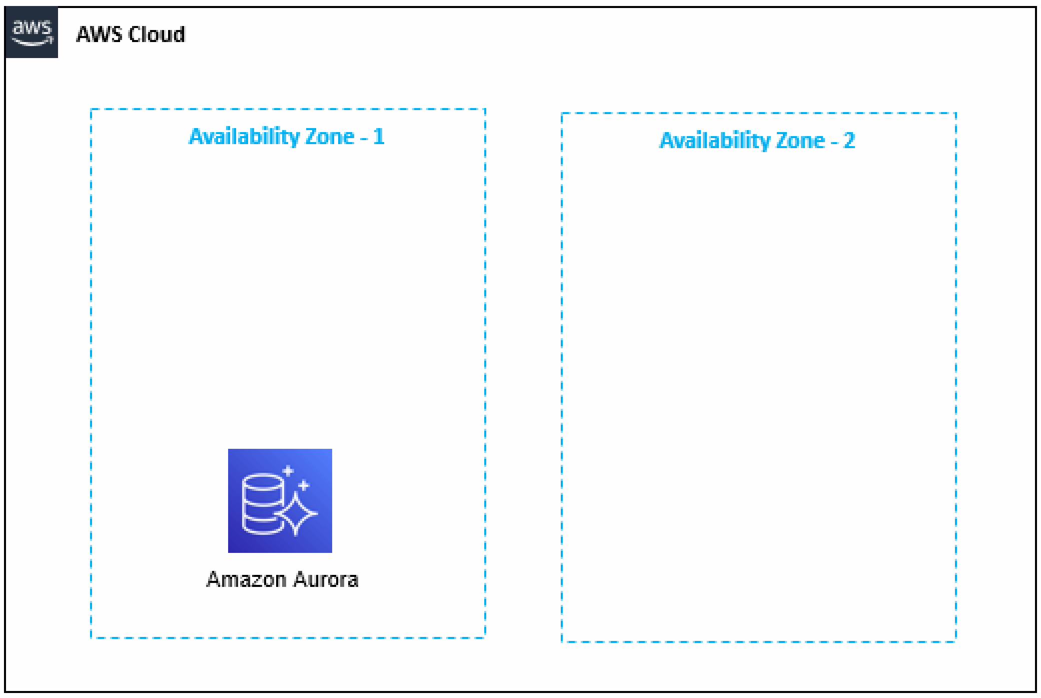
After you complete this lab, you will be able to:

- Create virtual machine using the AWS UI.
- Create SQL database and using AWS UI.
- Develop the Php Code to perform Read and Write operation SQL Servers.



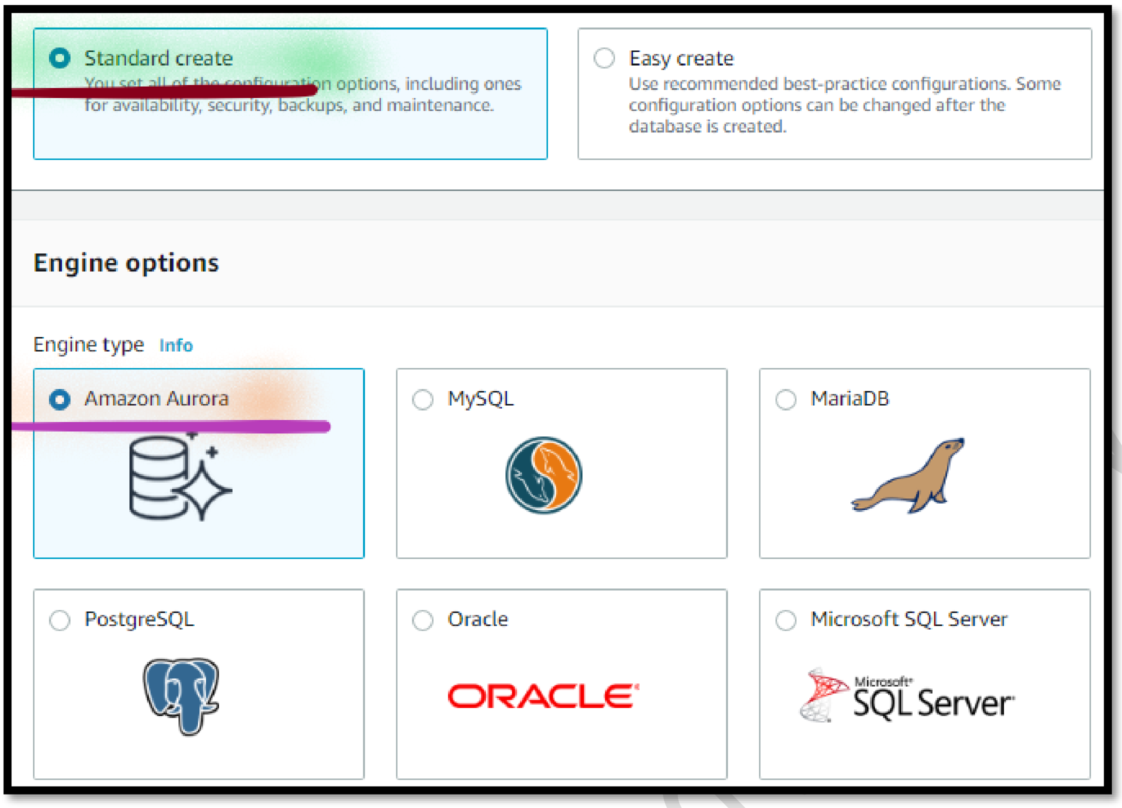
Task 1: Create a Database

In this task, you will create relational (Amazon Aurora) database.



Step 1: Create an Amazon RDS Instance

1. Choose the **US East (N. Virginia)** region list to the right of your account information on the navigation bar.
2. In the **AWS Management Console**, on the **Services** menu, click **RDS**.
3. Click **Create database** under dashboard.
    a. **Choose a database creation method**: Select **Standard create**.
    b. **Engine type**: Select **Amazon Aurora**.



[✓ Standard create]
You set all of the configuration options, including ones for availability, security, backups, and maintenance.

Easy create
Use recommended best-practice configurations. Some configuration options can be changed after the database is created.

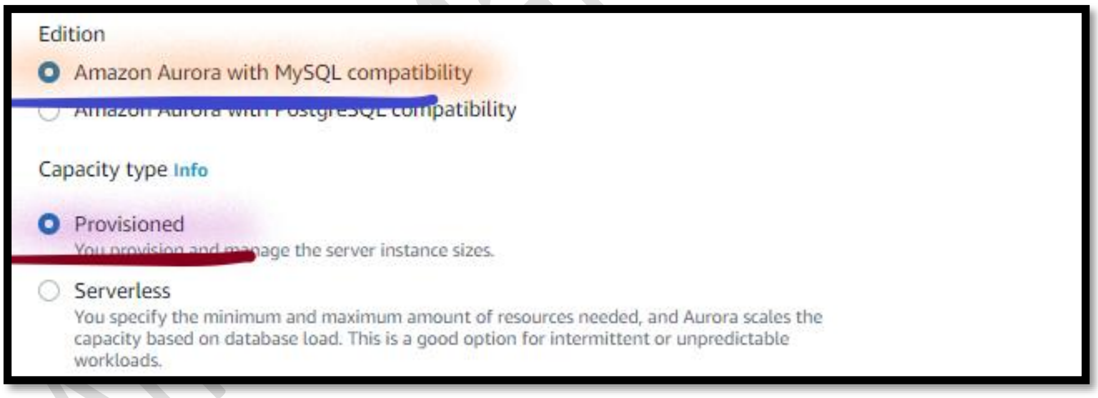
Engine options

Engine type Info

[✓ Amazon Aurora] MySQL MariaDB PostgreSQL Oracle Microsoft SQL Server

c. **Edition**: Select **Amazon Aurora with MySQL compatibility**.

d. **Capacity type**: Select **Provisioned**.



Edition

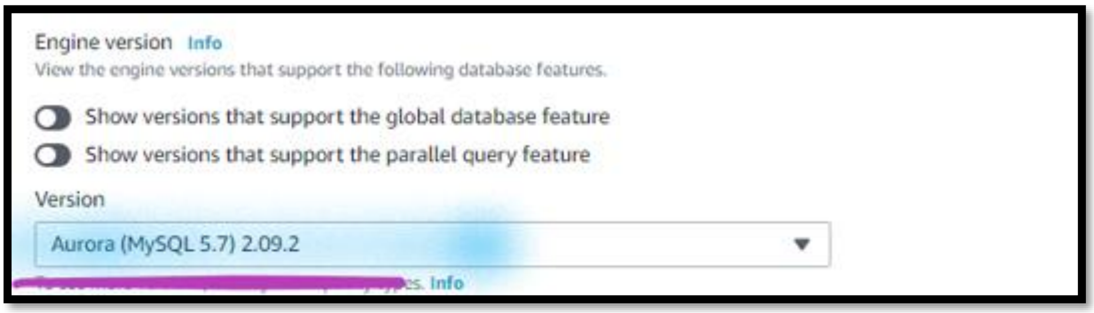
[✓ Amazon Aurora with MySQL compatibility]
Amazon Aurora with PostgreSQL compatibility

Capacity type Info

[✓ Provisioned]
You provision and manage the server instance sizes.

Serverless
You specify the minimum and maximum amount of resources needed, and Aurora scales the capacity based on database load. This is a good option for intermittent or unpredictable workloads.

e. **Engine version**: Dropdown and Select the **Latest version**.



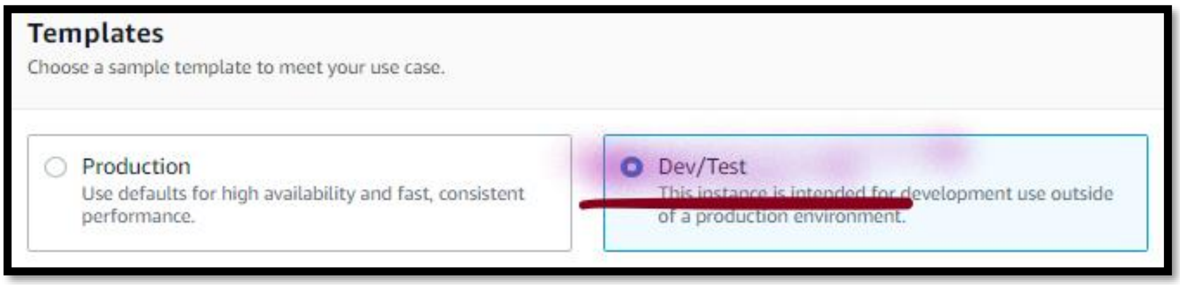
Engine version Info
View the engine versions that support the following database features.

Show versions that support the global database feature
Show versions that support the parallel query feature

Version
Aurora (MySQL 5.7) 2.09.2

[illegible]es. Info

f. **Templates**: Select **Dev/ Test**.


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.

g. On the **Settings** section, *configure*:

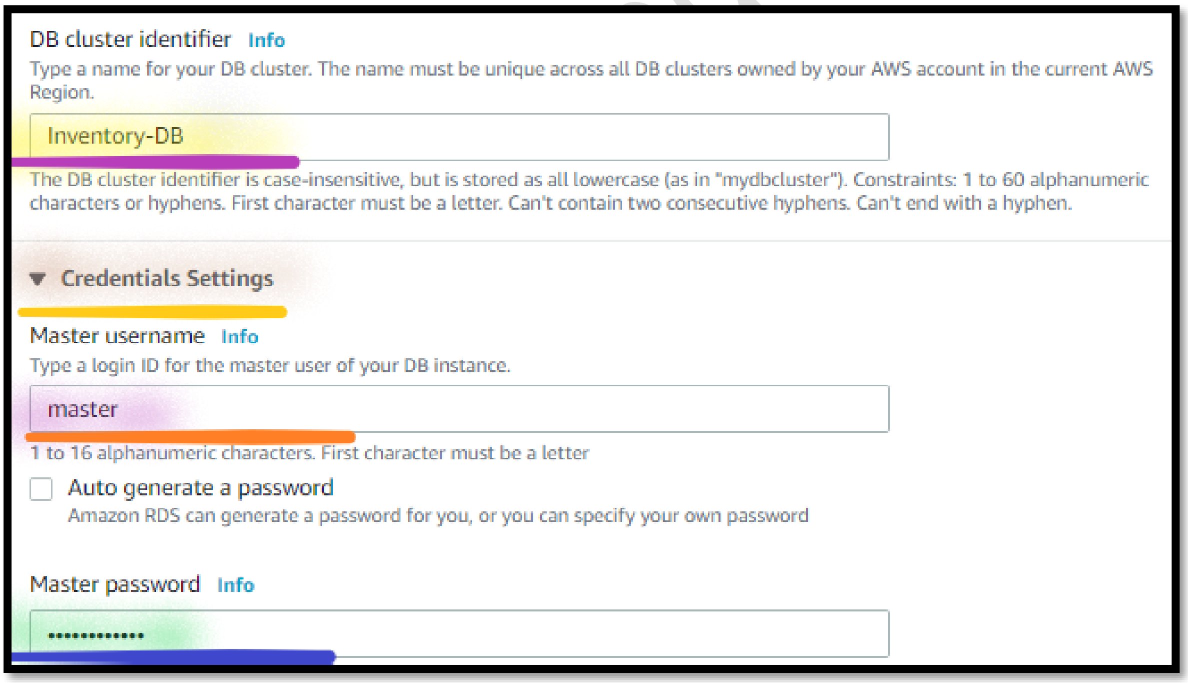
i. **DB cluster identifier**: Write **Inventory-DB**.

ii. **Expand** ***Credentials Settings***.

1) **Master username**: Write **master**.

2) **Master password**: Write **lab-password**.

3) **Confirm password**: Write **lab-password**.


DB cluster identifier Info
Type a name for your DB cluster. The name must be unique across all DB clusters owned by your AWS account in the current AWS Region.
Inventory-DB
The DB cluster identifier is case-insensitive, but is stored as all lowercase (as in "mydbcluster"). 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.
master
1 to 16 alphanumeric characters. First character must be a letter

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

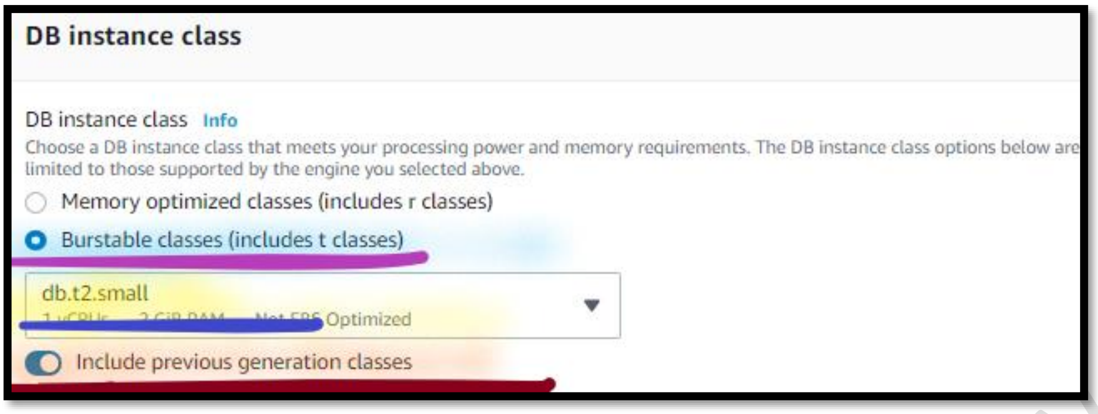
Master password Info
••••••••••••

h. On the **DB instance class** section, *Configure*:

i. **Enable** the ***Include previous generation classes***.

ii. **Select** the **Burstable classes (includes t classes)**.

iii. **Instance size**: Dropdown and Select **db.t2.small**.



DB instance class

DB instance class Info

Choose a DB instance class that meets your processing power and memory requirements. The DB instance class options below are limited to those supported by the engine you selected above.

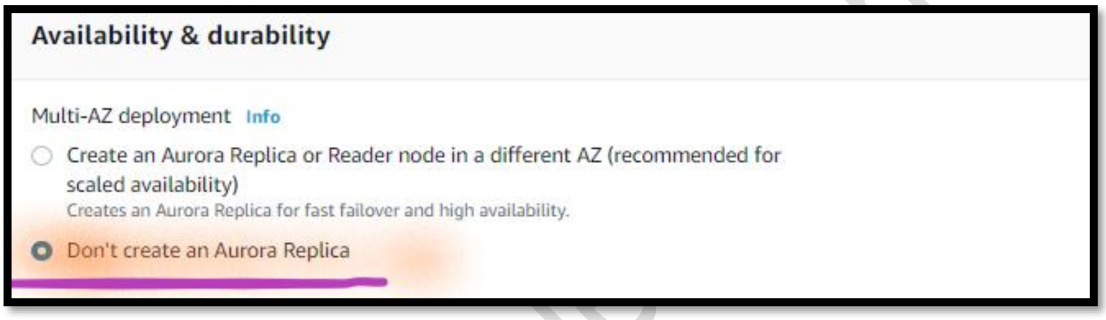
Memory optimized classes (includes r classes) [✓ Burstable classes (includes t classes)]

db.t2.small
1 vCPUs 2 GiB RAM Not EBS Optimized

[✓ Include previous generation classes]

i. On the **Availability & durability** section, *Configure*:

i. **Multi-AZ deployment**: Select **Don't create an Aurora Replica**.



Availability & durability

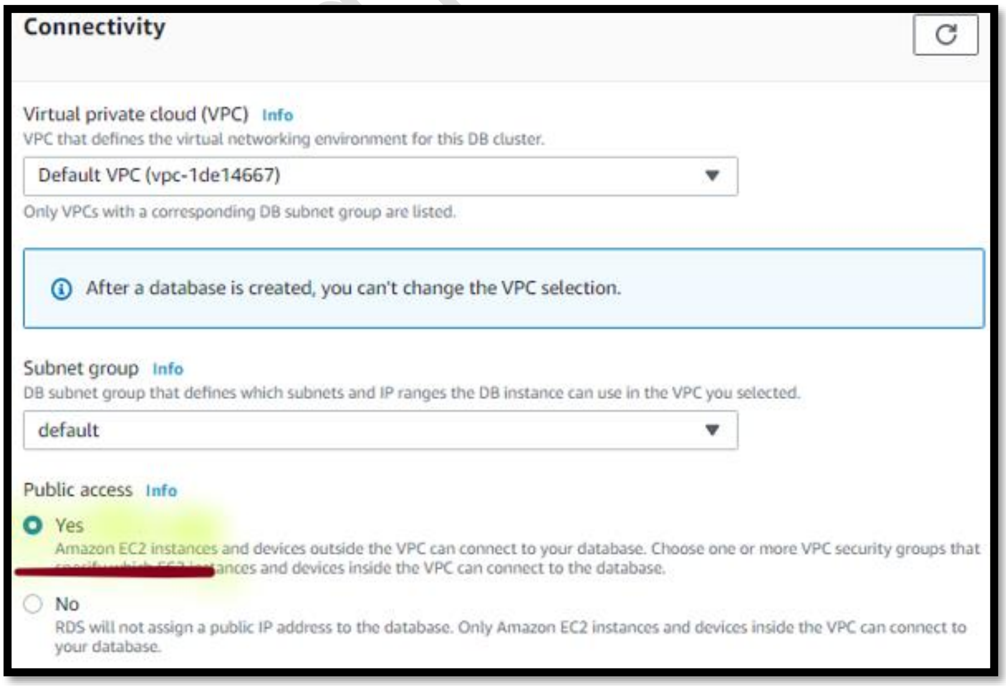
Multi-AZ deployment Info

Create an Aurora Replica or Reader node in a different AZ (recommended for scaled availability)
Creates an Aurora Replica for fast failover and high availability.

[✓ Don't create an Aurora Replica]

j. On the **Connectivity** section, *Configure*:

i. **Public access**: Select **Yes**.



Connectivity

Virtual private cloud (VPC) Info
VPC that defines the virtual networking environment for this DB cluster.
Default VPC (vpc-1de14667)
Only VPCs with a corresponding DB subnet group are listed.

After a database is created, you can't change the VPC selection.

Subnet group Info
DB subnet group that defines which subnets and IP ranges the DB instance can use in the VPC you selected.
default

Public access Info

[✓ Yes]
Amazon EC2 instances and devices outside the VPC can connect to your database. Choose one or more VPC security groups that specify which EC2 instances and devices inside the VPC can connect to the database.

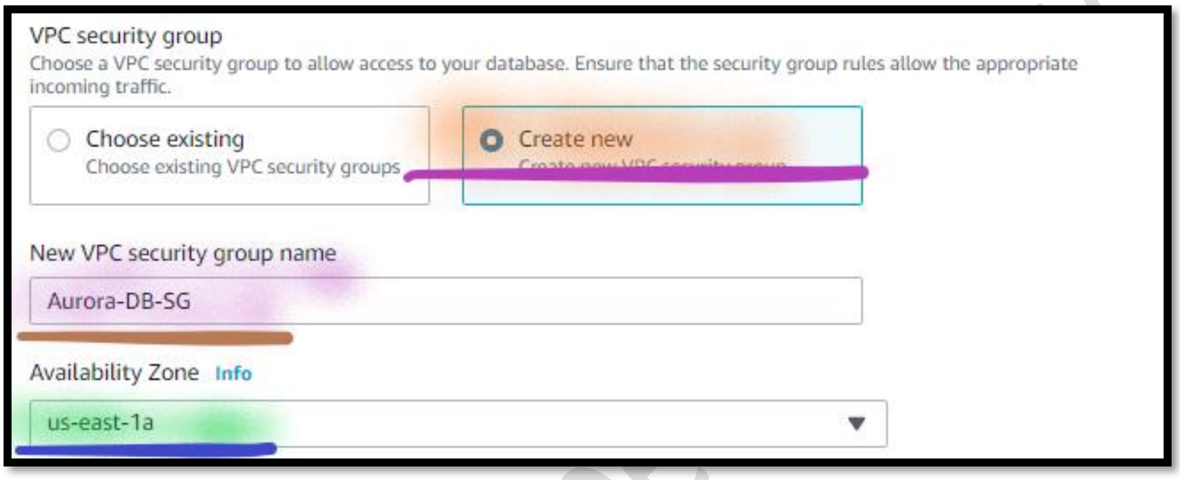
No
RDS will not assign a public IP address to the database. Only Amazon EC2 instances and devices inside the VPC can connect to your database.

ii. **VPC security groups**: Select Create new.

1) **New VPC security group name**: Write Aurora-DB-SG.

2) **Availability zone**: Dropdown and Select First Availability zone (*1a*).

Note: Leave other details as default.

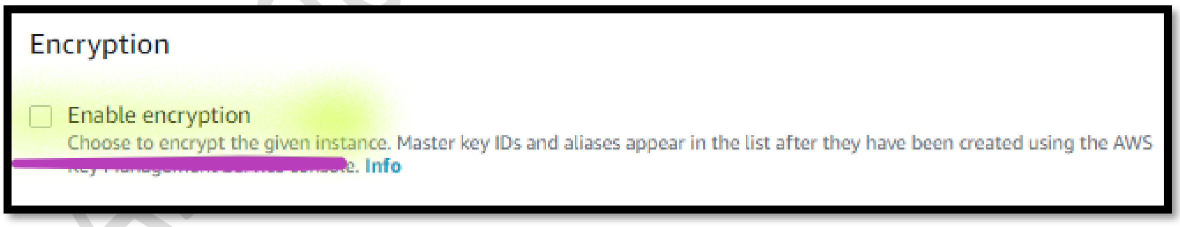


VPC security group
Choose a VPC security group to allow access to your database. Ensure that the security group rules allow the appropriate incoming traffic.
Choose existing — Choose existing VPC security groups
[✓ Create new] — Create new VPC security group
New VPC security group name: Aurora-DB-SG
Availability Zone Info: us-east-1a

k. Expand the ***Additional configuration*** section, *Configure*:

i. **Encryption**: Unselect the **Enable encryption**.

Note: Leave other details as default.



Encryption
☐ Enable encryption
Choose to encrypt the given instance. Master key IDs and aliases appear in the list after they have been created using the AWS Key Management Service console. Info

l. Click Create database (*at the bottom of the page*).

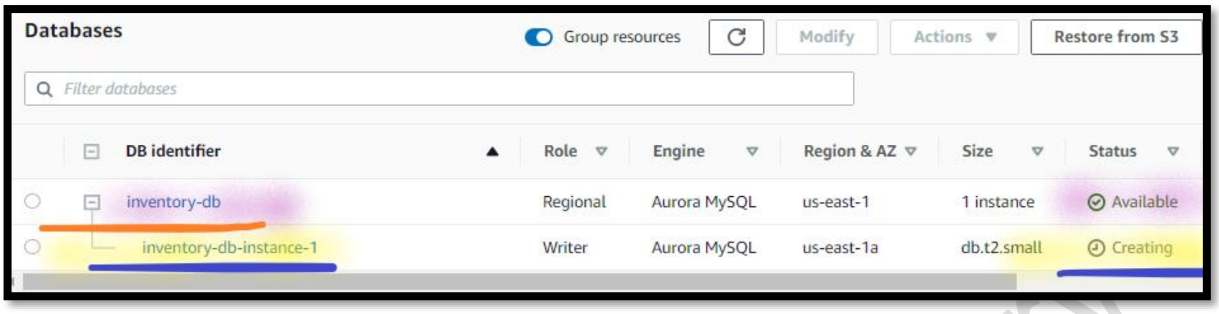
Step 2: View the Amazon Aurora Instance

4. In the **AWS Management Console**, on the Services menu, click RDS.

5. Click Databases.

a. Expand the ***inventory-db***.

Note: You will see the Database instance **status** as **Creating**.



Databases

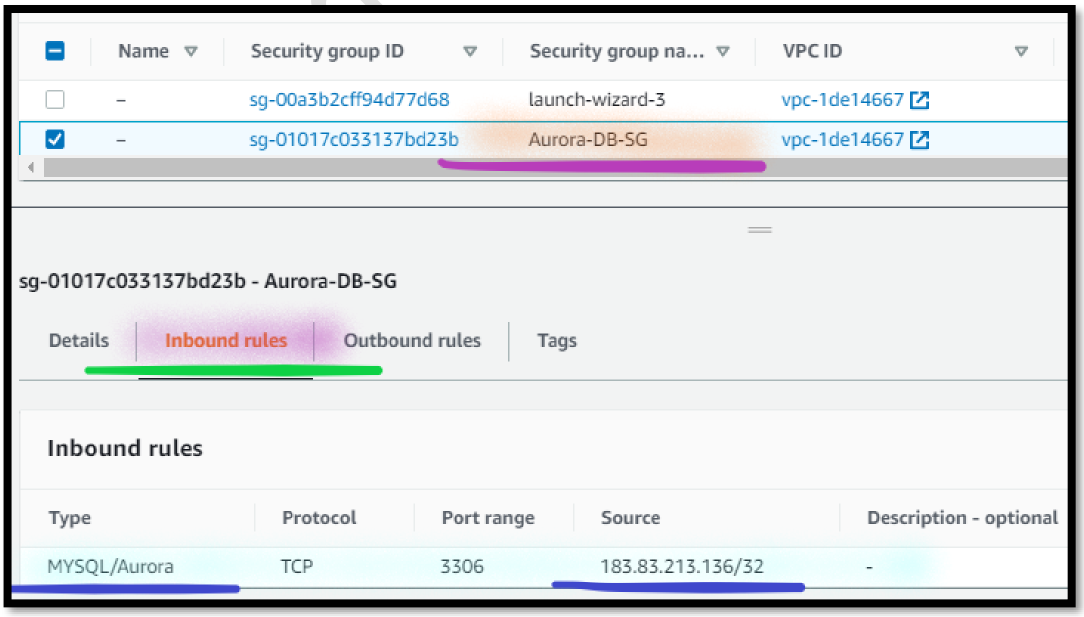
DB identifier	Role	Engine	Region & AZ	Size	Status
inventory-db	Regional	Aurora MySQL	us-east-1	1 instance	Available
inventory-db-instance-1	Writer	Aurora MySQL	us-east-1a	db.t2.small	Creating

Note: Go to the next Task. **Don't wait** for database instance creation.

Step 3: Update Security Group for Database

6. In the **AWS Management Console**, on the **Services** menu, click **EC2**.
7. ***Go to the left*** navigation pane, Click the **Security Groups**.
8. Select the **Aurora-DB-SG** Security group.
    a. ***Go below*** in the console and Select **inbound rules**.

Note: You will see, Public IP address shown under **Source** against MySQL port.



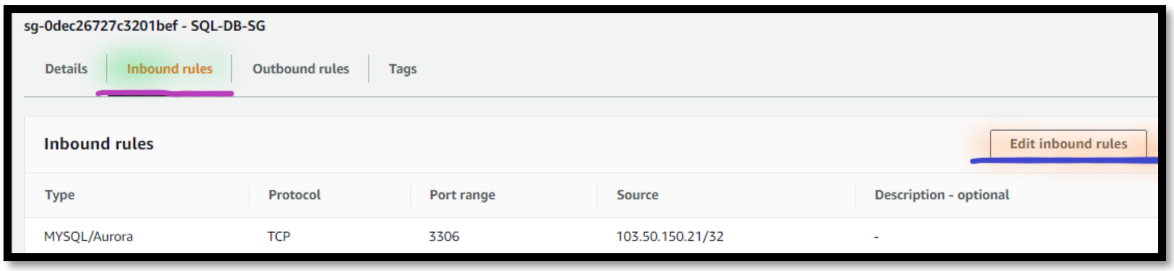
Name	Security group ID	Security group na...	VPC ID
–	sg-00a3b2cff94d77d68	launch-wizard-3	vpc-1de14667
–	sg-01017c033137bd23b	Aurora-DB-SG	vpc-1de14667

sg-01017c033137bd23b - Aurora-DB-SG

Inbound rules

Type	Protocol	Port range	Source	Description - optional
MYSQL/Aurora	TCP	3306	183.83.213.136/32	-

b. Select **Edit inbound rules**.



sg-0dec26727c3201bef - SQL-DB-SG

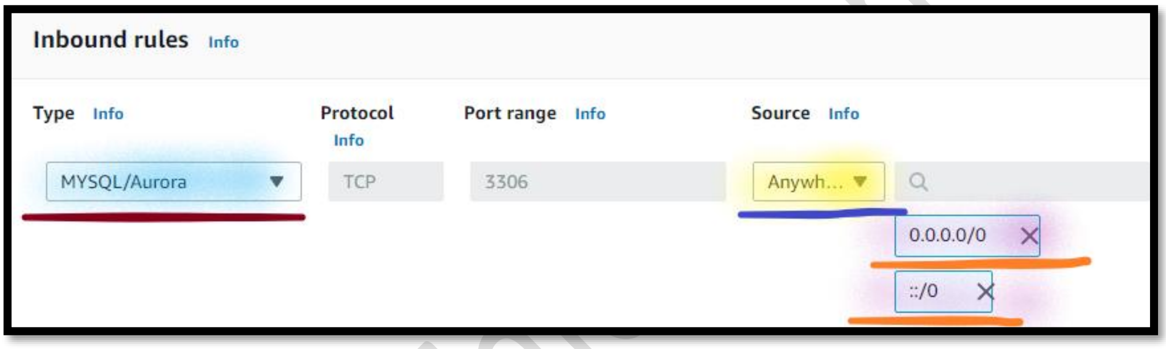
Details | Inbound rules | Outbound rules | Tags

Inbound rules — Edit inbound rules

Type	Protocol	Port range	Source	Description - optional
MYSQL/Aurora	TCP	3306	103.50.150.21/32	-

i. **Source**: Dropdown and select **Anywhere Ipv4**.

Note: You will see, Public IP address shown under **Source** against Aurora port is removed and IP address **0.0.0.0/0** added.



Inbound rules Info

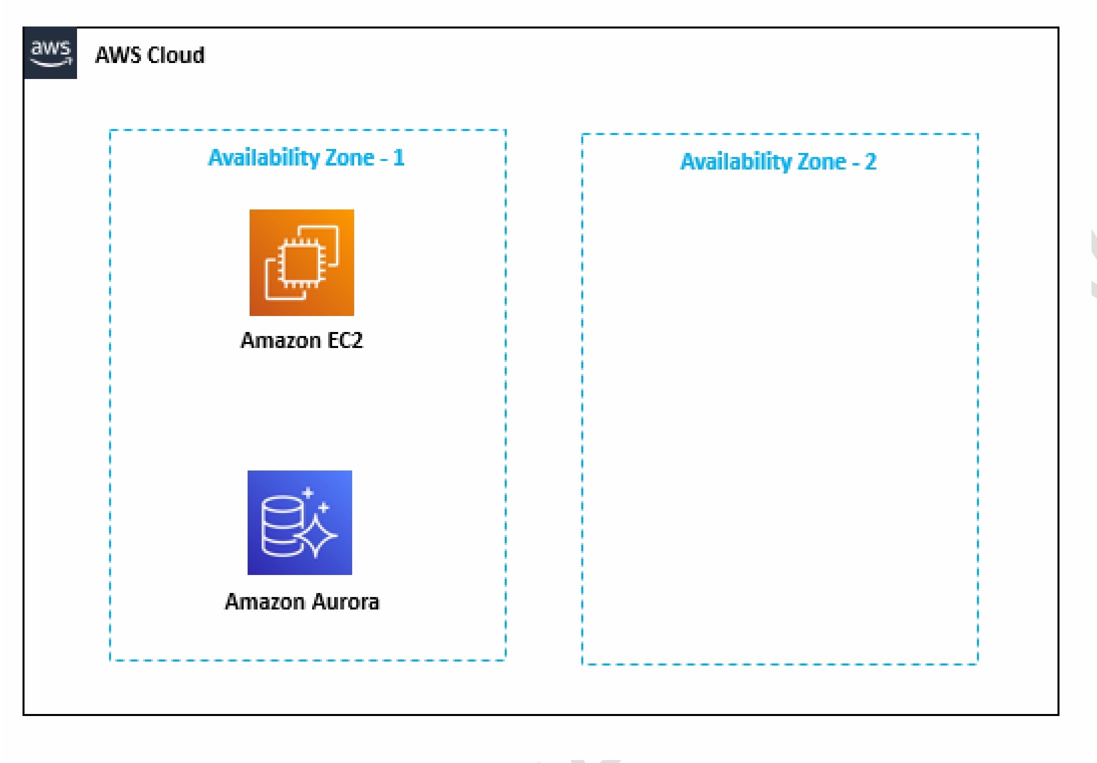
Type	Protocol	Port range	Source
MYSQL/Aurora	TCP	3306	Anywh... 0.0.0.0/0 ::/0

ii. Select **Save rules**

Note: Now the **WebApp instance** can receive traffic from **Anywhere**.

Task 2: Deploy the WebApp Server

In this task, you will create Ubuntu virtual machine (Amazon EC2).



Step 1: Create EC2 Instance

9. Choose the **US East (N. Virginia)** region list to the right of your account information on the navigation bar.
10. In the **AWS Management Console**, on the **Services** menu, click **EC2**.
11. Select **Instances**.
12. Select **Launch Instances**.
    a. In the **Name and tags** section:
        i. **Name**: Write **Web Server**.
    b. In the **Application and OS Images** section:
        i. In the **Search box**:
            a) Type **Ubuntu Server 18.04 LTS**.
            b) Press **Enter** ***key***.

Note: You can see the **Choose an Amazon Machine Image** page.

c) **From** the **Choose an Amazon Machine Image** page:

1) Select **Ubuntu Server 18.04 LTS**

Note: You can see the **Launch an Instance** page.

c. In the **Instance Type** section:

   i. **Instance type**: Dropdown and in the **Search box**:

      a) Type **t2.micro**.

      b) Select **t2.micro**.

d. In the **Key pair (login)** section:

   i. **Key pair name**: Dropdown and select **My-Dev-LAB-KP**.

e. In the **Network setting** section:

   i. Select **Edit**.

      a) **Firewall**: Select **Create security group**.

         1) **Security group name**: Write **Web-Server-SG**.

         2) **Description**: Write **Web Server Group**.

         3) **Inbound security groups rules**:

            I. In the **Security group rule 1**:

               1) **Type**: Dropdown and select **SSH**.

               2) **Source type**: Dropdown and select **Anywhere**.

            II. Select **Add Security group rule**.

            III. In the **Security group rule 2**:

               1) **Type**: Dropdown and select **HTTP**.

2) **Source type**: Dropdown and select **Anywhere**.

Note: Leave the other details as default.

f. In the **Summary** section:

i. Select **Launch Instances**.

Note: **Wait**, till you can see the **message** "**Successfully initiated launch of instance**".

g. Select **View all instances**

Note: **Wait**, till you can see the **Web Server** Instance **State** is **Running**.

Note: **Wait**, till you can see the **Web Server** Instance **Status check** is **2/2 check passed**.

Step 2: Check the WebApp Server Status

13. In the **AWS Management Console**, on the **Services** menu, click **EC2**.
14. Click **Instance**.
15. Select the **Web Server**.

i. **Wait** for the **Instance State** to change to **Running** state.

ii. **Wait** for the **Status check** to change to **2/2 checks passed**.

Task 3: Create Database Table

In this task, you will create SQL database, table and structure.

Step 1: Copy the Amazon Aurora Instance

16. In the **AWS Management Console**, on the **Services** menu, click **RDS**.
17. Click **Databases**.

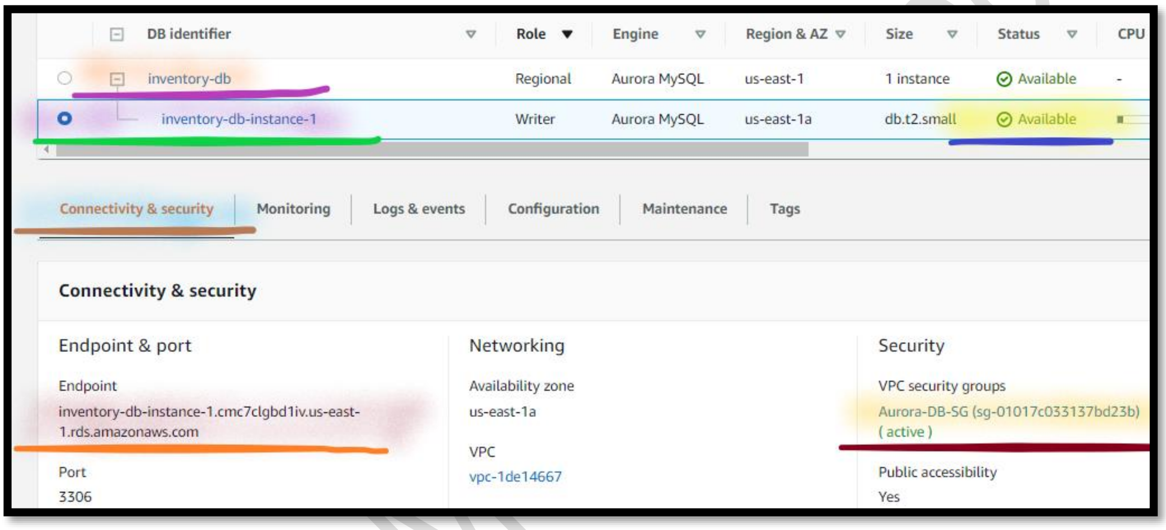
a. **Expand** the ***inventory-db***.

Note: You will see the Database instance **status** as **Available**, If database instance status is not showing as available. Keep **Refresh**, until database instance status shown as available.

b. Select and Open the **inventory-db-instance-1** whose **Role** as ***Writer***.

i. ***Go below*** in the console, Select the **Connectivity & security**.

ii. **Copy** the ***inventory-db-instance-1*** database instance **Endpoint** in the **Notepad**.



DB identifier	Role	Engine	Region & AZ	Size	Status	CPU
inventory-db	Regional	Aurora MySQL	us-east-1	1 instance	Available	-
inventory-db-instance-1	Writer	Aurora MySQL	us-east-1a	db.t2.small	Available	

Connectivity & security

Endpoint & port

Endpoint
inventory-db-instance-1.cmc7clgbd1iv.us-east-1.rds.amazonaws.com

Port
3306

Networking

Availability zone
us-east-1a

VPC
vpc-1de14667

Security

VPC security groups
Aurora-DB-SG (sg-01017c033137bd23b)
(active)

Public accessibility
Yes

Note: Ensure, in the VPC security group, You will see only one Security group Aurora-DB-SG.

Step 2: Copy the IP Address of Web Server

18. In the **AWS Management Console**, on the **Services** menu, click **EC2**.

19. Select the **Web Server**.

a. Select the **Details**.

Note: **Copy** the **Public IP address** of **Web Server** in the **Notepad**.

Step 3: Connect to Web Server Instance

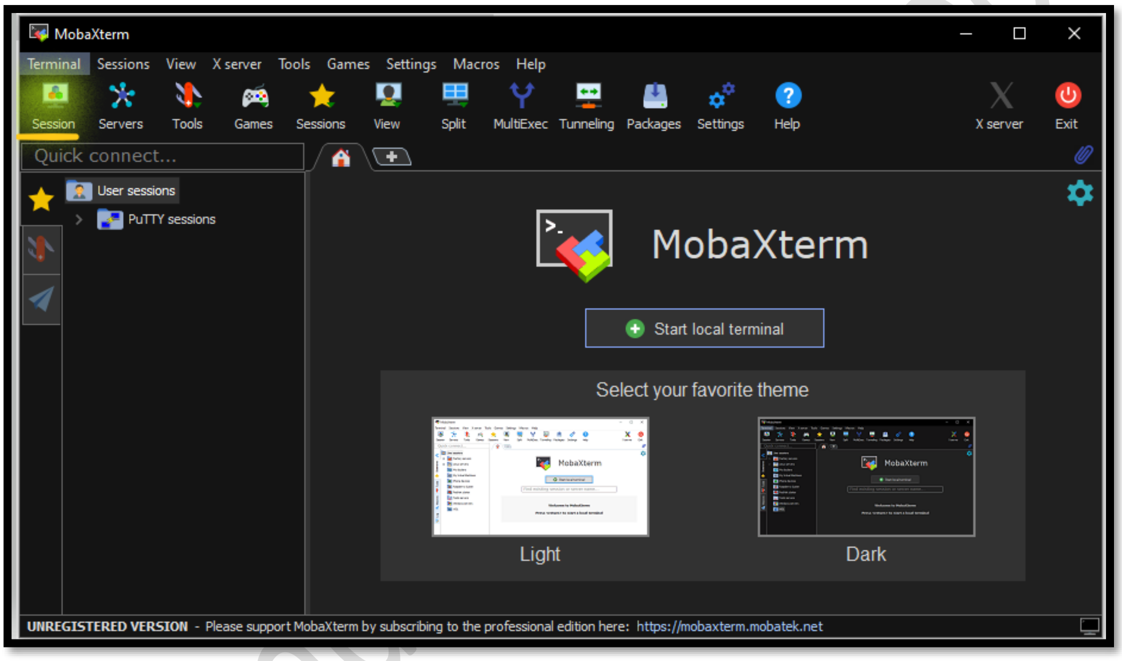
20. From the **Local Desktop/ Laptop** (Windows Desktop), **Download** the **MobaXterm** (**Portable edition**).

    https://mobaxterm.mobatek.net/download-home-edition.html

21. From the **Local Desktop/ Laptop** (Windows Desktop), **Open** the **MobaXterm**.

22. From the **MobaXterm**.

    a. Select **Session**.


MobaXterm
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help X server Exit
Quick connect...
User sessions
PuTTY sessions
MobaXterm
Start local terminal
Select your favorite theme
Light
Dark

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

    b. Select **SSH**.

        i. Select **Advanced SSH settings**.

            1. **Remote host**: Write **Public IP address** of the **Linux Web Server**.

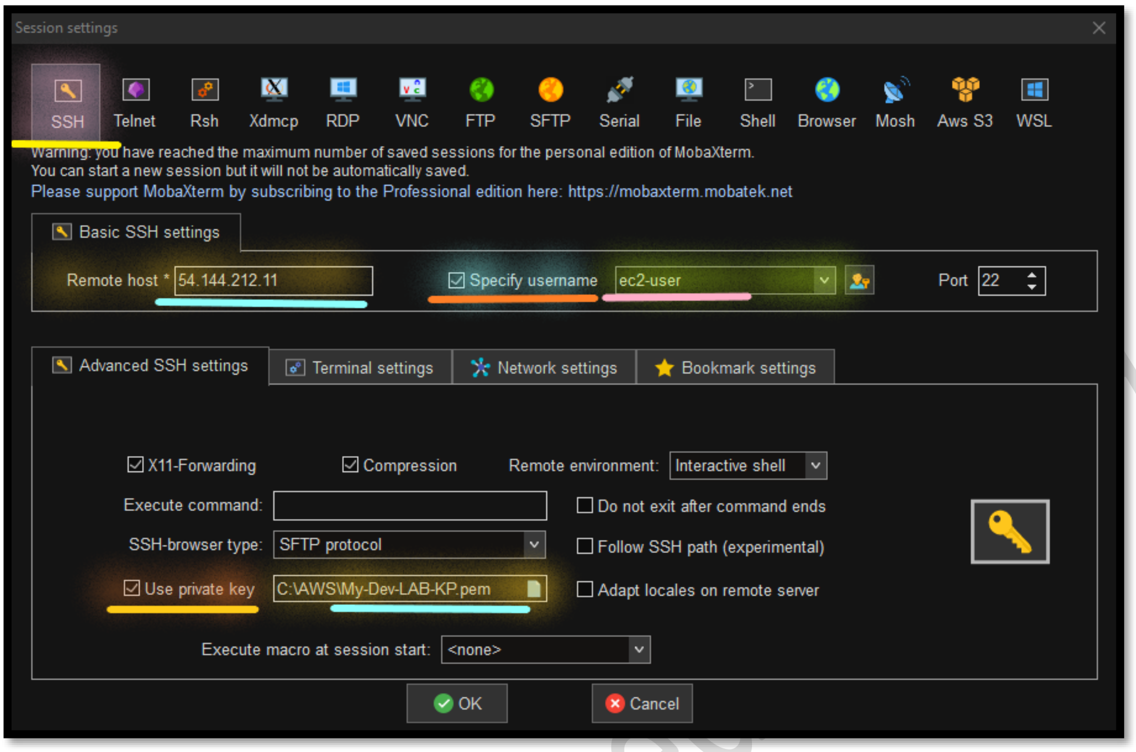
            2. **Specify username**: **Enable** the **Checkmark**.

            3. **Specify username**: Write **ubuntu**.

            4. **Use Private key**: **Enable** the **Checkmark**.

            5. **Use Private key**: Click on the **Search box**:

1) **Navigate** and **select** the **My-Dev-LAB-KP**.pem.



Session settings

Warning: you have reached the maximum number of saved sessions for the personal edition of MobaXterm.
You can start a new session but it will not be automatically saved.
Please support MobaXterm by subscribing to the Professional edition here: https://mobaxterm.mobatek.net

Basic SSH settings

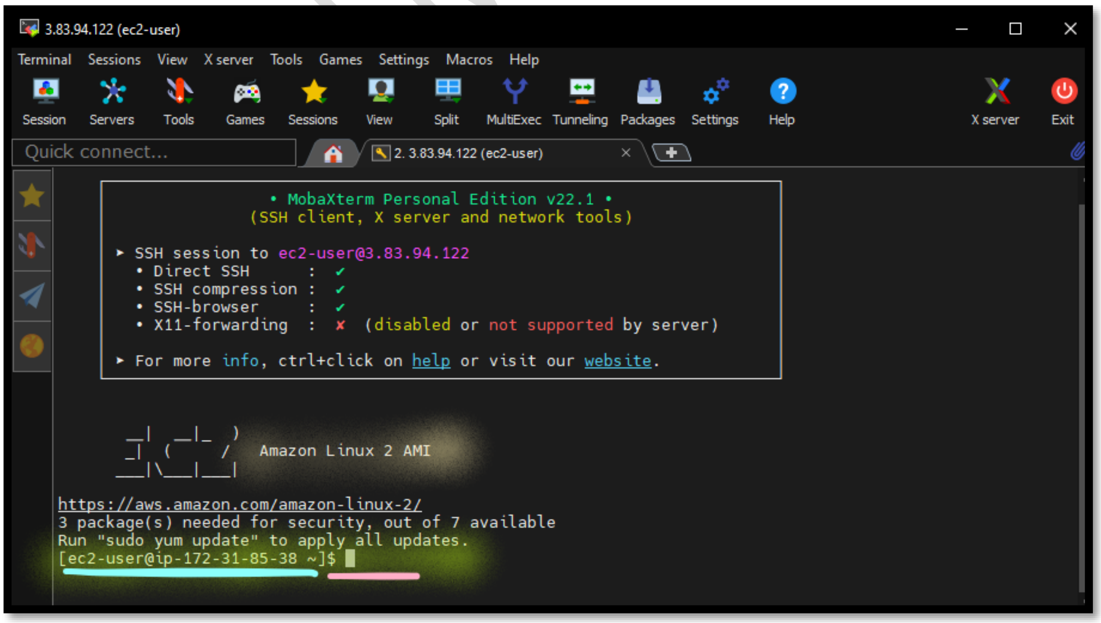
Remote host * 54.144.212.11 ☑ Specify username ec2-user Port 22

Advanced SSH settings | Terminal settings | Network settings | Bookmark settings

☑ X11-Forwarding ☑ Compression Remote environment: Interactive shell
Execute command: ☐ Do not exit after command ends
SSH-browser type: SFTP protocol ☐ Follow SSH path (experimental)
☑ Use private key C:\AWS\My-Dev-LAB-KP.pem ☐ Adapt locales on remote server
Execute macro at session start: <none>

6. Select **Ok**.

Note: You can see the **Linux Console**.



3.83.94.122 (ec2-user)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help X server Exit
Quick connect...
2. 3.83.94.122 (ec2-user)

```
            • MobaXterm Personal Edition v22.1 •
          (SSH client, X server and network tools)

 ▸ SSH session to ec2-user@3.83.94.122
   • Direct SSH      :  ✔
   • SSH compression :  ✔
   • SSH-browser     :  ✔
   • X11-forwarding  :  ✘  (disabled or not supported by server)

 ▸ For more info, ctrl+click on help or visit our website.

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
3 package(s) needed for security, out of 7 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-85-38 ~]$
```

Step 4: Install the MySQL client

23. **From** the **Ubuntu terminal**, execute the below commands:

   a. Update the **packages**.

   ```
   sudo apt-get -y update
   ```

   b. Install the **mysql client**.

   ```
   sudo apt-get install -y mysql-client
   ```

Step 5: Create the Database

24. **From** the **Ubuntu terminal**, execute the below commands:

   a. **Connect** to **Amazon Aurora** Instance.

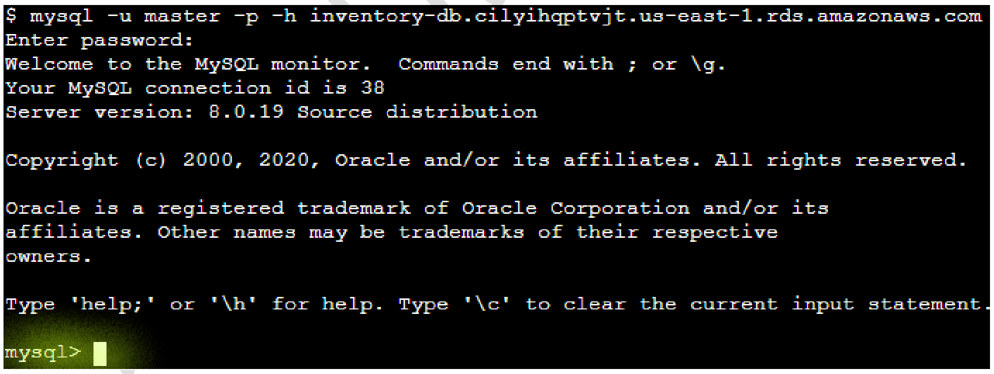
   ```
   mysql -u master -p -h Hostname.
   ```

Note: **Replace** the **Hostname** with **RDS Amazon Aurora Writer** instance **endpoint**,which you have copied in the previous step.

   b. You will be **Prompted** to enter the password.

   Type the password as **lab-password**.

   c. You will be shown a brief introduction message and then be placed at the **mysql>** ***prompt***.

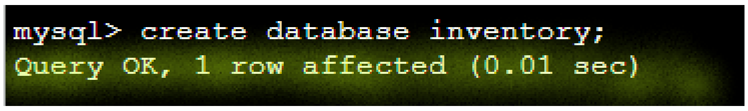


```
$ mysql -u master -p -h inventory-db.cilyihqptvjt.us-east-1.rds.amazonaws.com
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 38
Server version: 8.0.19 Source distribution

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

   d. **Create** the **inventory** database.

   ```
   create database inventory;
   ```

Note: In the Output you can see "**Query OK, 0 rows affected**" message.

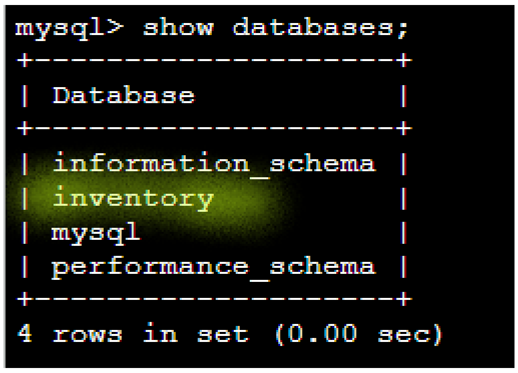


```
mysql> create database inventory;
Query OK, 1 row affected (0.01 sec)
```

e. **List** the ***existing*** databases.

show databases;

Note: In the Output, you can see the **inventory** database.

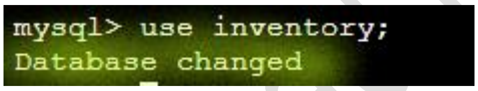


```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| inventory          |
| mysql              |
| performance_schema |
+--------------------+
4 rows in set (0.00 sec)
```

f. **Use** the **inventory** database.

use inventory;

Note: In the output, should show "**database change**" message.

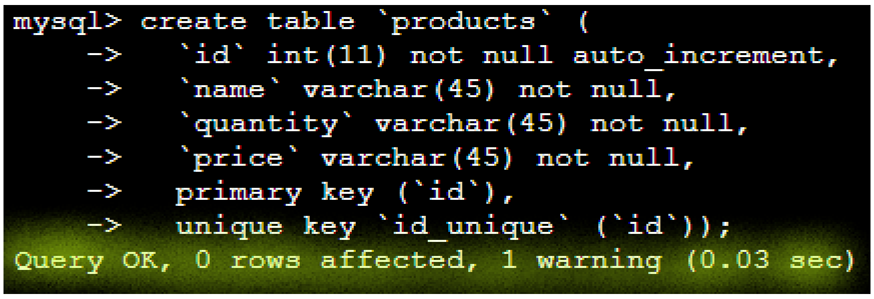


```
mysql> use inventory;
Database changed
```

g. **Create** the **products** table in the **inventory** database.

```
create table `products` (
 `id` int(11) not null auto_increment,
 `name` varchar(45) not null,
 `quantity` varchar(45) not null,
 `price` varchar(45) not null,
 primary key (`id`),
 unique key `id_unique` (`id`));
```

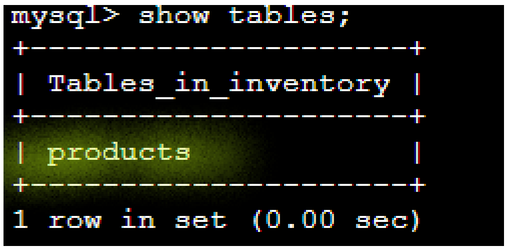
Note: In the Output you can see "**Query OK, 0 rows affected**" message.



```
mysql> create table `products` (
    ->   `id` int(11) not null auto_increment,
    ->   `name` varchar(45) not null,
    ->   `quantity` varchar(45) not null,
    ->   `price` varchar(45) not null,
    ->   primary key (`id`),
    ->   unique key `id_unique` (`id`));
Query OK, 0 rows affected, 1 warning (0.03 sec)
```

h. **List** the ***tables*** created under the **inventory** databases.
   show tables;

Note: In the Output, you can see the **products** table.



```
mysql> show tables;
+---------------------+
| Tables_in_inventory |
+---------------------+
| products            |
+---------------------+
1 row in set (0.00 sec)
```

i. **Exit** the database ***instance***.
   exit

Note: Go to the next task, but **Don't close the Linux terminal**.

Task 4: Develop the Php Application

In this task, you will develop the Php code who can perform read and write operation from single database server.

Step 1: Develop the Code to Perform CRUD Operation on Amazon Aurora Database

25.**Unzip** the **LAB-05-01-code.zip** (**Php code**).

Note: **Lab-05-01-code.zip** code file is available with the Lab manual.

26.Open the **data.php** in the **Notepad**.

27. **Update** the ***Amazon Aurora database*** details in the code:

    a. **Replace** the **TO DO 1** with the ***inventory-db-instance-1 database*** instance **endpoint**, which you have copied in the previous step.

Note: Don't remove the starting and end quote (**' '**)and semicolon (**;**).

    b. **Replace** the **TO DO 2** with the database instance ***user name*** **master**.

    c. **Replace** the **TO DO 3** with the database instance ***password*** **lab-password**.

    d. **Replace** the **TO DO 4** with the ***database name*** **inventory**.

    e. **Replace** the **TO DO 5** with the ***inventory database table name*** **products**.

28. Select **File**.

29. Select **Save**.

Task 5: Deploy the Php Application

In this task, you will deploy the Php code into Aws virtual machine and configure the runtime environment.

Step 2: Upload the Code to GitHub Repository

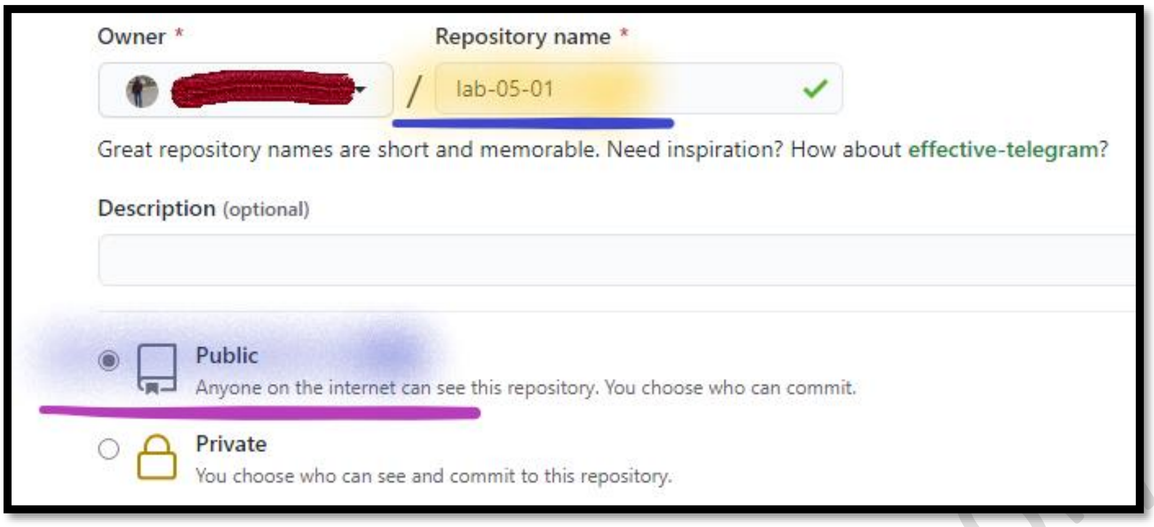
30. **Login** into your **GitHub account**.

31. Select **New repository** and **configure**:

    a. **Repository name**: Write **lab-05-01**.

    b. Select **Public**.

Note: Leave the other details as default.



Owner * Repository name *

[redacted] / lab-05-01 ✓

Great repository names are short and memorable. Need inspiration? How about effective-telegram?

Description (optional)

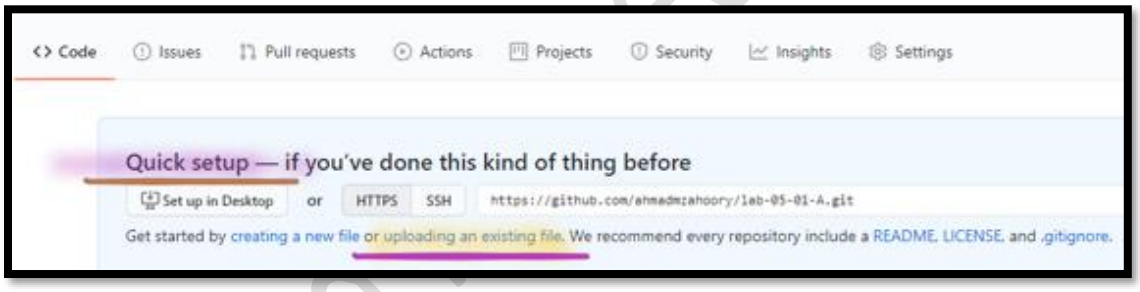
● Public
Anyone on the internet can see this repository. You choose who can commit.

○ Private
You choose who can see and commit to this repository.

c. Select **Create repository**.

32. **From** the **lab-05-01** repository:

a. Select the **uploading an existing file** under ***Quick setup***.

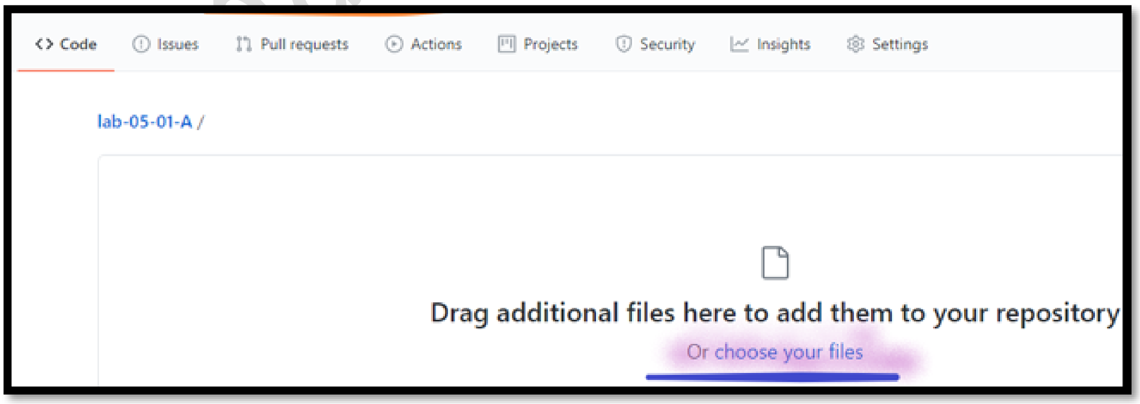


Quick setup — if you've done this kind of thing before

Set up in Desktop or HTTPS SSH `https://github.com/ahmadmzahoory/lab-05-01-A.git`

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

b. Select **Choose your files**.



lab-05-01-A /

Drag additional files here to add them to your repository

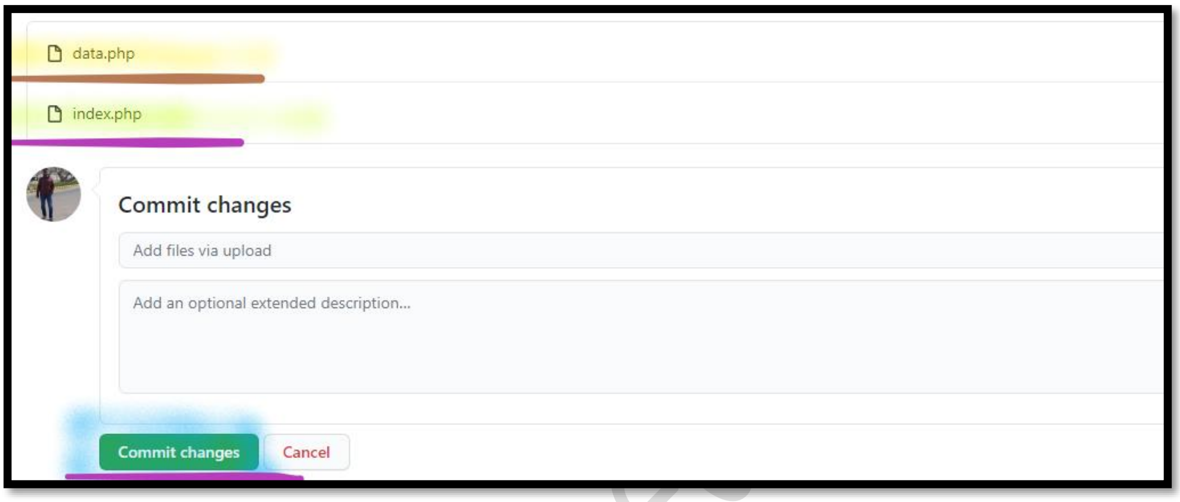
Or choose your files

i. **Navigate** and Select **index.php** and **data.php** files.

Note: **Select** the **data.php** and **index.php** files. **Don't upload the zip folder**.

ii. Select **Open**.

Note: In the github repository you can see two files (**data.php** and **index.php**) files.



data.php

index.php

Commit changes

Add files via upload

Add an optional extended description...

iii. Select **Commit changes**.

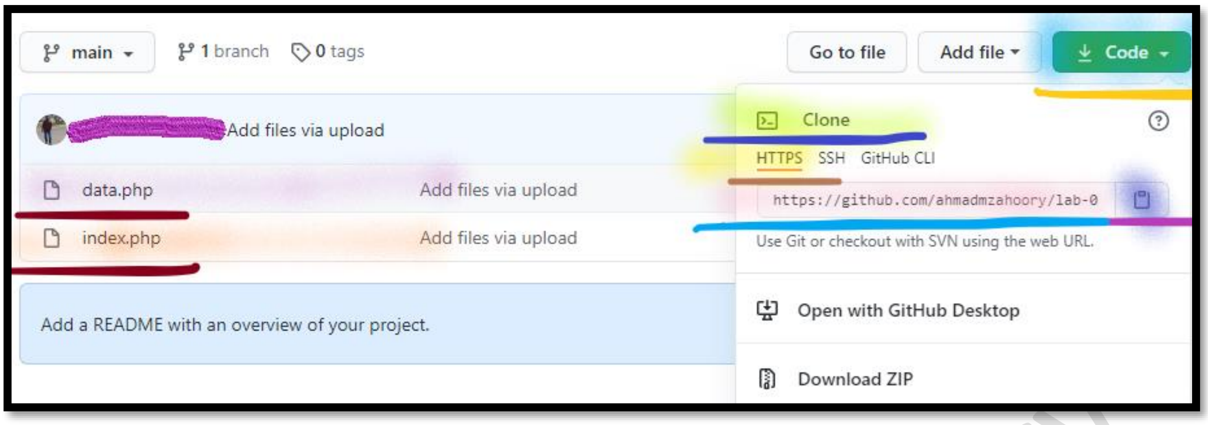
Note: Once code **uploaded successfully**, you can see them in repository.

Step 3: Clone the GitHub repository

33. **Return** to the **lab-05-01** ***GitHub repository***.

a. ***Go to the right***, Click on **Code**.

b. **Copy** the **Clone URL** in the **Notepad**.



[redacted] Add files via upload

data.php Add files via upload

index.php Add files via upload

Add a README with an overview of your project.

https://github.com/ahmadmzahoory/lab-0

Use Git or checkout with SVN using the web URL.

Step 4: Install Php Runtime environment to Deploy the PHP Amazon Aurora Code

34. **Return** to the **Ubuntu terminal**, execute the below commands:

   a. **Install** the ***Apache***.
      sudo apt-get install -y apache2

   b. **Install** the ***Php 7.2***.
      sudo apt-get install -y php7.2

   c. **Install** the ***MySQL Module for Php 7.2***.
      sudo apt-get install -y php7.2-mysql

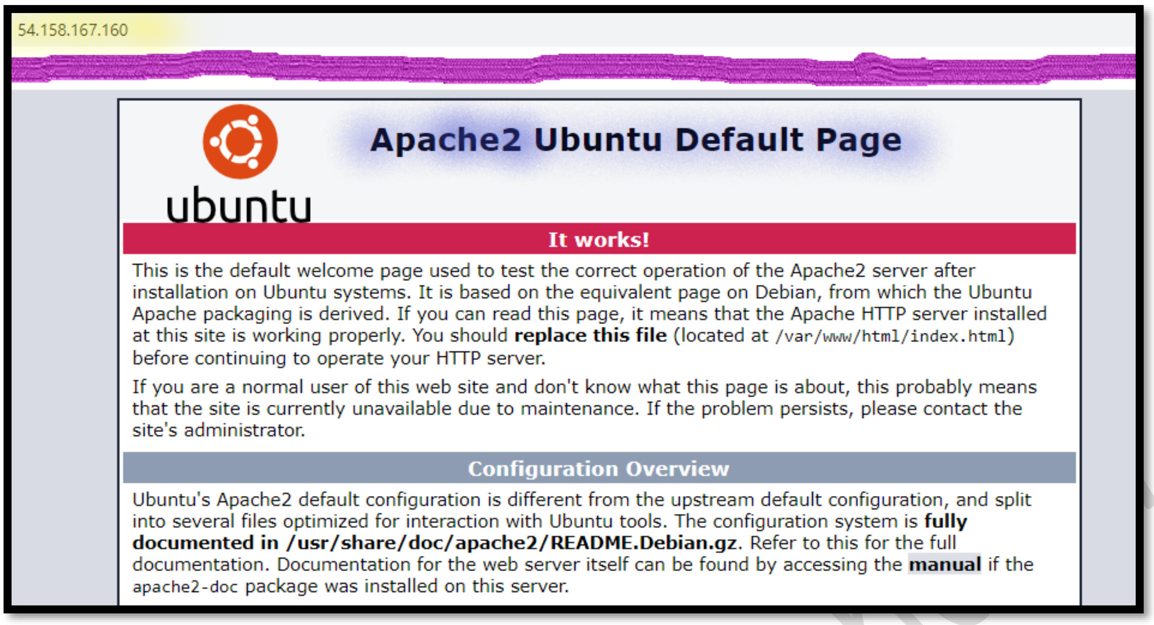
   d. **Install** the ***Git***.
      sudo apt-get install -y git

Note: Go to the next task, but **Don't close the Linux terminal**.

Step 5: Access the Web App Server

35. From your local desktop/ laptop **Web browser**, type **Public IP Address** of **WebApp Server** (Ubuntu virtual machine) and access your **website**.

Note: You will see the **Default Apache Ubuntu page**.


54.158.167.160 [redacted]

Apache2 Ubuntu Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

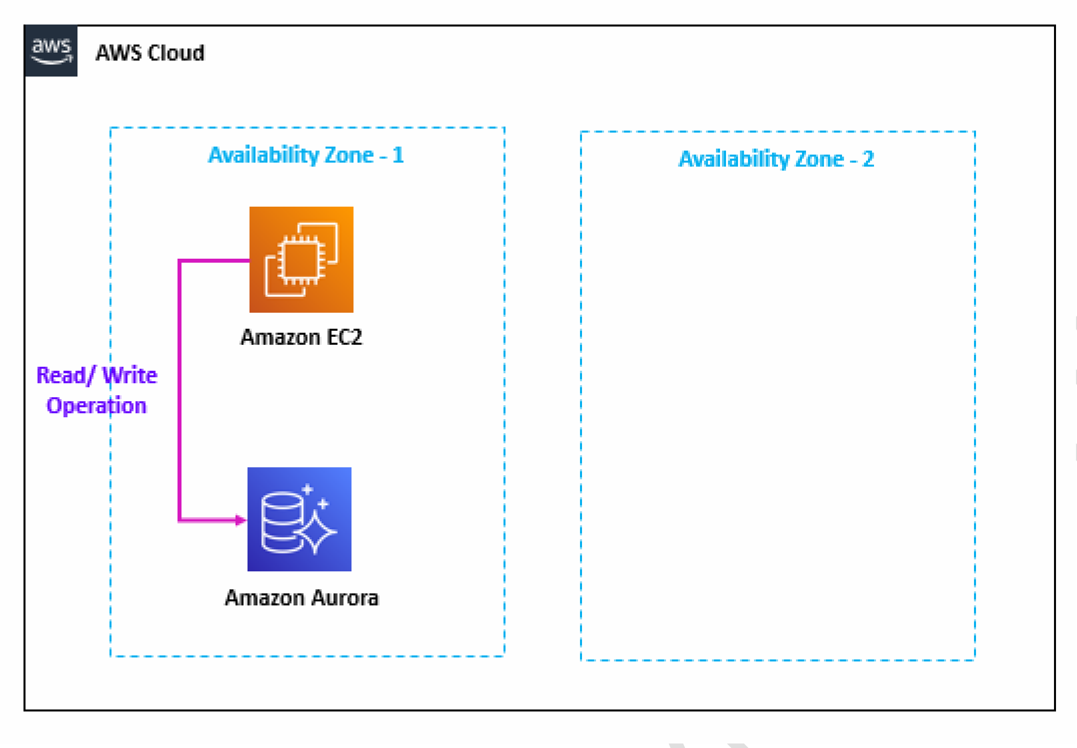
If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

Note: Go to the next task, but **Don't close the website**.

Step 6: Deploy the Php Code



36. **Return** to the **Ubuntu terminal**, execute the below commands:
   a. **Change** to ***/var/www/html***.
      cd /var/www/html/

   b. **List** the ***file & folders***.
      ls -l

Note: You can see the **index.html** (default page) file.

   c. **Remove** the default ***index.html***.
      sudo rm index.html

   d. **Clone** the lab-05-01 ***GitHub repository***.
      sudo git clone **CLONE-URL**

Note: **Replace** the **CLONE-URL** with the **GitHub URL** you have copied in the previous step.

   e. **List** the ***file & folders***.
      ls -l

Note: You can see the **lab-05-01** folder.

f. **Change** to ***lab-05-01 folder***.
```
cd lab-05-01
```

g. **Move** all the contents to ***/var/www/html***.
```
sudo mv -v /var/www/html/lab-05-01/* /var/www/html/
```

h. **Change** to ***parent directory***.
```
cd ..
```

g. **List** the ***file & folders***.
```
ls -l
```

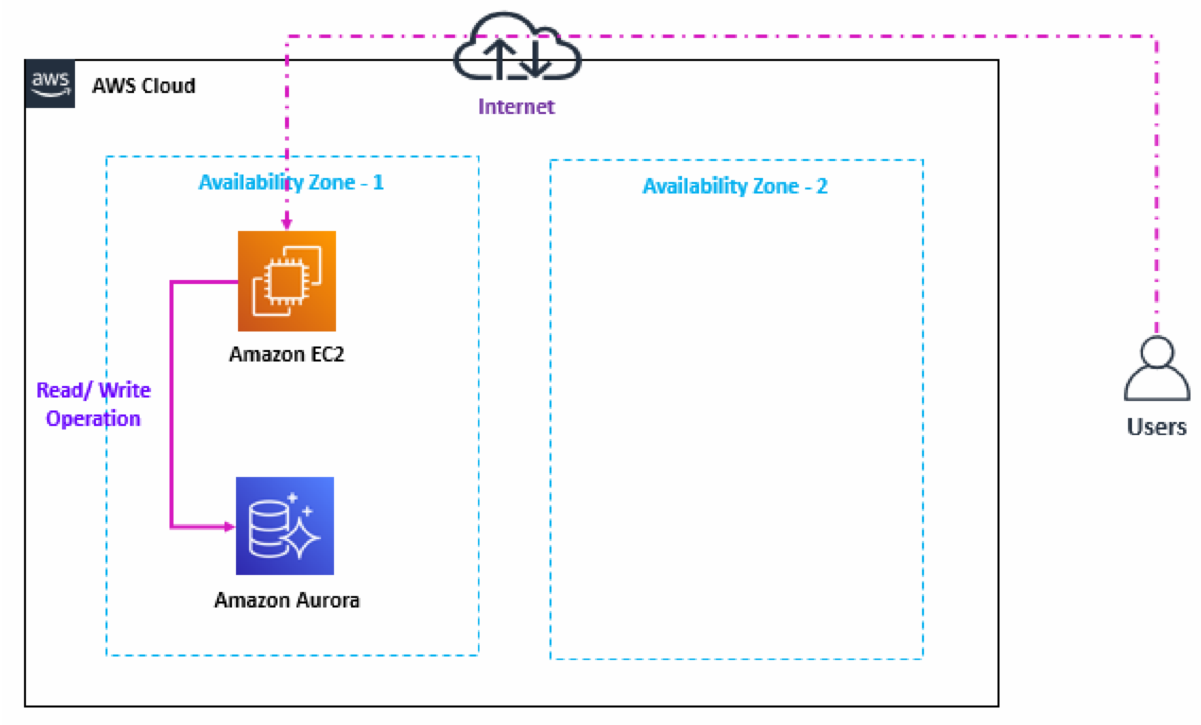
Note: You can see the **data.php** and **index.php** files.

h. **Restart** the ***apache service***.
```
sudo systemctl restart apache2.service
```

Note: Go to the next task, but **Don't close the Linux terminal**.

Task 6: Access the Deployment

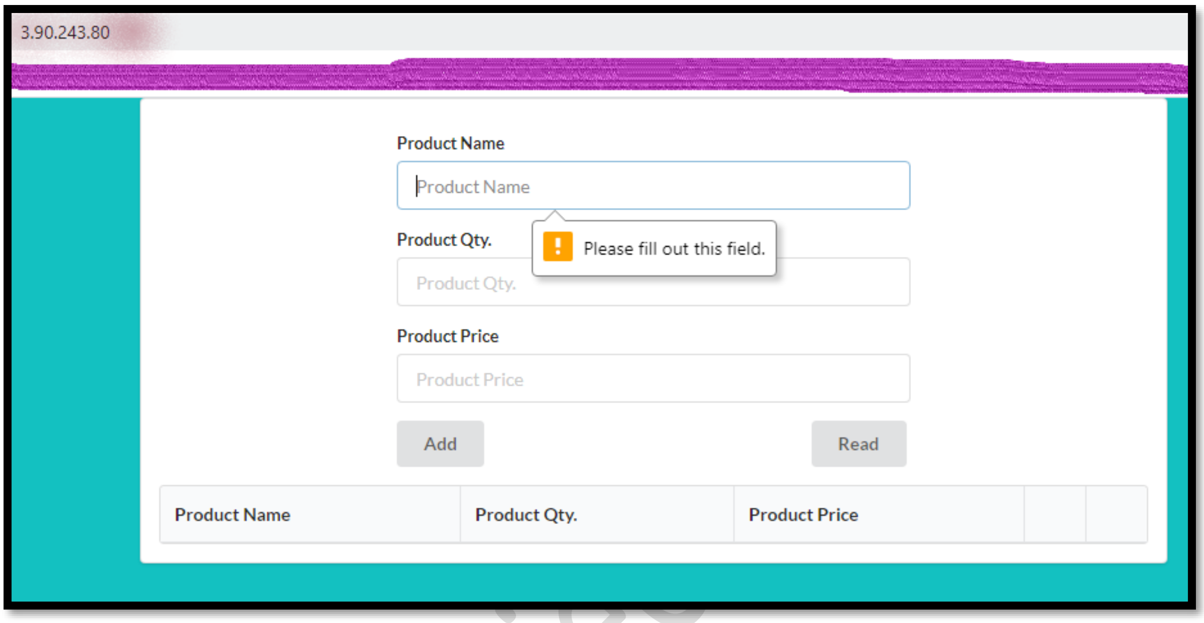
In this task, you will test your deployment by performing the CRUD operation.



Step 1: Access the Php App Server

37. **Return** to the **Web browser**, from where you have opened the **Web App Server** (Linux virtual machine) and **Refresh** your **website.**

Note: You will see the Web Application page.


3.90.243.80

[redacted]

Product Name

Product Name

Please fill out this field.

Product Qty.

Product Qty.

Product Price

Product Price

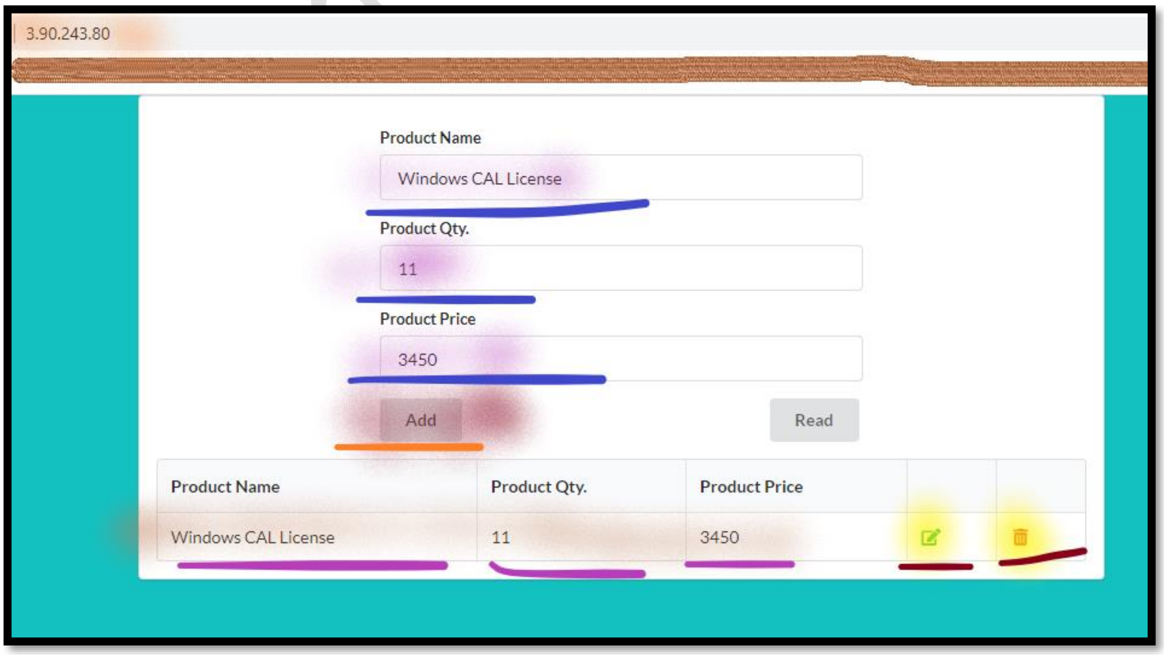
Add Read

Product Name	Product Qty.	Product Price		

Step 2: Perform the CRUD Operation

38. You can **perform** the **CRUD operation**.

a. **Add** the ***Inventory Data***.


3.90.243.80

[redacted]

Product Name

Windows CAL License

Product Qty.

11

Product Price

3450

Add Read

Product Name	Product Qty.	Product Price		
Windows CAL License	11	3450		

b. You can also **Update** the ***Inventory* Data**.

c. You can also **Delete** the ***Inventory* Data**.

Note: Go to the next task, but **Don't close the website**.

Task 12: Clean up the Environment

Step 1: Terminate EC2 Instances

39. In the **AWS Management Console**, on the **Services** menu, click **EC2**.
40. Select **Instances**.
41. Select **Web Server**.
    a. Select **Instance State**.
    b. Select **Terminate instance**.
    c. Select **Terminate**.

Step 2: Terminate an RDS

42. In the **AWS Management Console**, on the **Services** menu, click **RDS**.
43. Select the **Databases**.
44. **Expand** the ***inventory-db***.
    a. Select and Open the **inventory-db-instance-2**.
        i. Select **Actions**.
        ii. Select **Delete**.
        iii. **Unselect** the **Create final snapshot**.
        iv. Select the **I acknowledge ………**.
        v. To confirm deletion, type **delete me** in the field.
        vi. Choose **Delete**.