Objective:

To create a public and private EC2 instance in AWS where the public instance acts as a web server and the private server is responsible for holding a database which is to be fetched and displayed when accessing the webpage.

Setup/Integration:

The public and private subnets that were previously created are used to launch new EC2 instances according to the requirements.

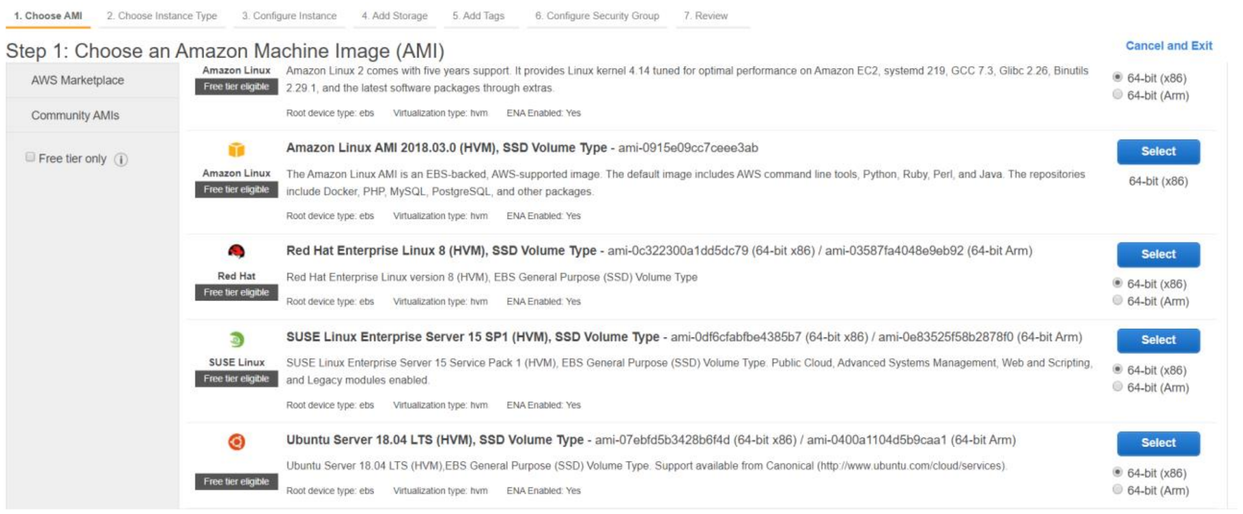
OS: Ubuntu 18.04 LTS

Type: T2 micro

RAM: 1GB

STORAGE: 8GB General Purpose SSD

We select the required Ubuntu Image



Step 1: Choose an Amazon Machine Image (AMI)

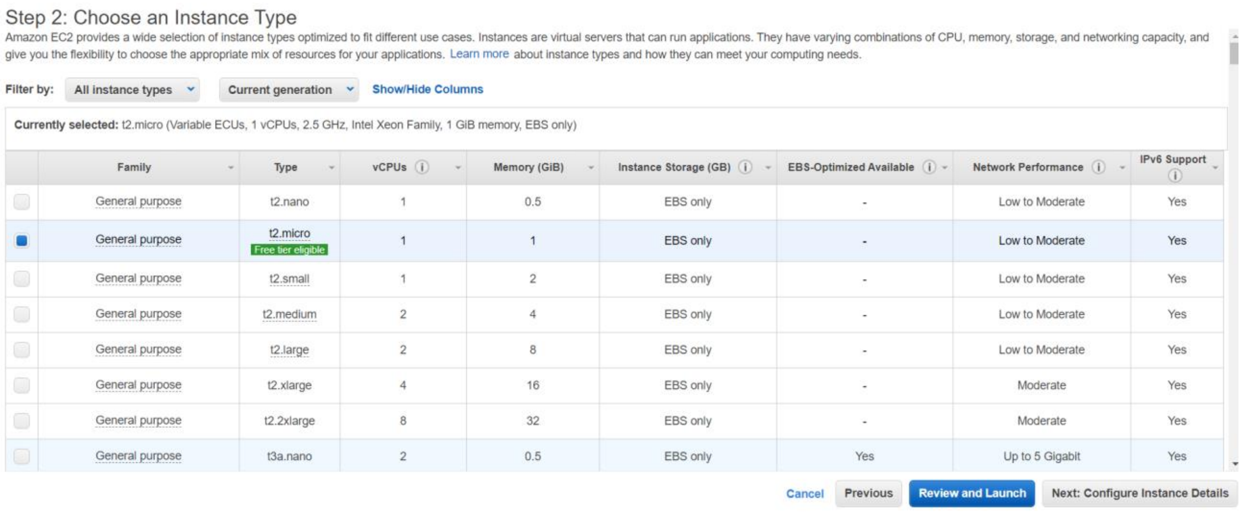
Amazon Linux — Free tier eligible
Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras.
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes
[✓ 64-bit (x86)] 64-bit (Arm)

Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type - ami-0915e09cc7ceee3ab
Amazon Linux — Free tier eligible
The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes
64-bit (x86)

Red Hat Enterprise Linux 8 (HVM), SSD Volume Type - ami-0c322300a1dd5dc79 (64-bit x86) / ami-03587fa4048e9eb92 (64-bit Arm)
Red Hat — Free tier eligible
Red Hat Enterprise Linux version 8 (HVM), EBS General Purpose (SSD) Volume Type
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes
[✓ 64-bit (x86)] 64-bit (Arm)

SUSE Linux Enterprise Server 15 SP1 (HVM), SSD Volume Type - ami-0df6cfabfbe4385b7 (64-bit x86) / ami-0e83525f58b2878f0 (64-bit Arm)
SUSE Linux — Free tier eligible
SUSE Linux Enterprise Server 15 Service Pack 1 (HVM), EBS General Purpose (SSD) Volume Type. Public Cloud, Advanced Systems Management, Web and Scripting, and Legacy modules enabled.
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes
[✓ 64-bit (x86)] 64-bit (Arm)

Ubuntu Server 18.04 LTS (HVM), SSD Volume Type - ami-07ebfd5b3428b6f4d (64-bit x86) / ami-0400a1104d5b9caa1 (64-bit Arm)
Free tier eligible
Ubuntu Server 18.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes
[✓ 64-bit (x86)] 64-bit (Arm)



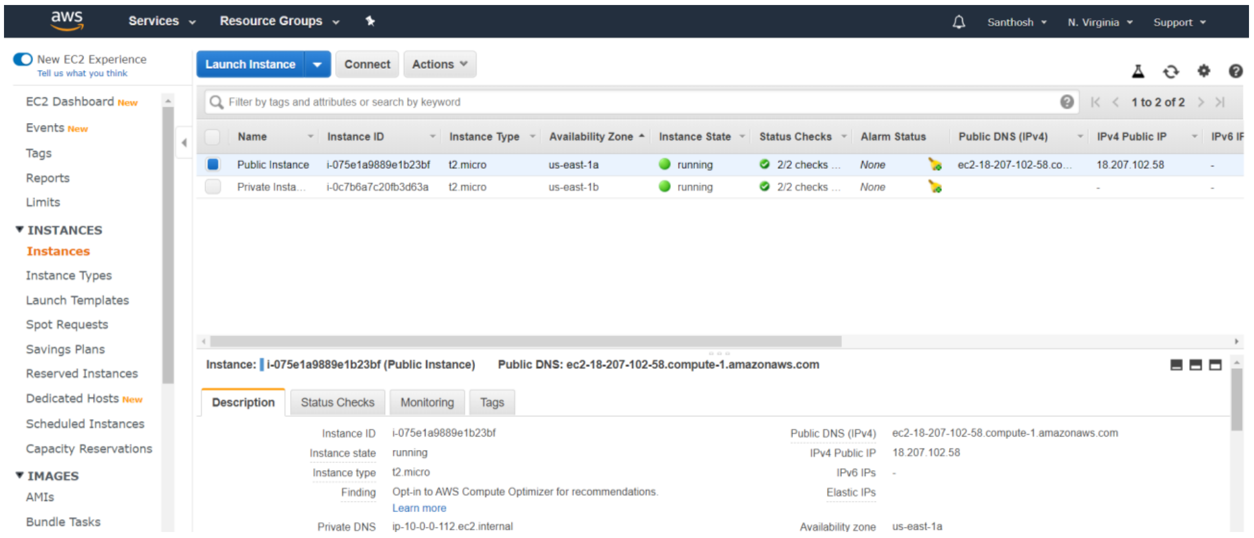
Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. Learn more about instance types and how they can meet your computing needs.

Filter by: All instance types | Current generation | Show/Hide Columns

Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance	IPv6 Support
	General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
■	General purpose	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate	Yes
	General purpose	t2.small	1	2	EBS only	-	Low to Moderate	Yes
	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate	Yes
	General purpose	t2.large	2	8	EBS only	-	Low to Moderate	Yes
	General purpose	t2.xlarge	4	16	EBS only	-	Moderate	Yes
	General purpose	t2.2xlarge	8	32	EBS only	-	Moderate	Yes
	General purpose	t3a.nano	2	0.5	EBS only	Yes	Up to 5 Gigabit	Yes

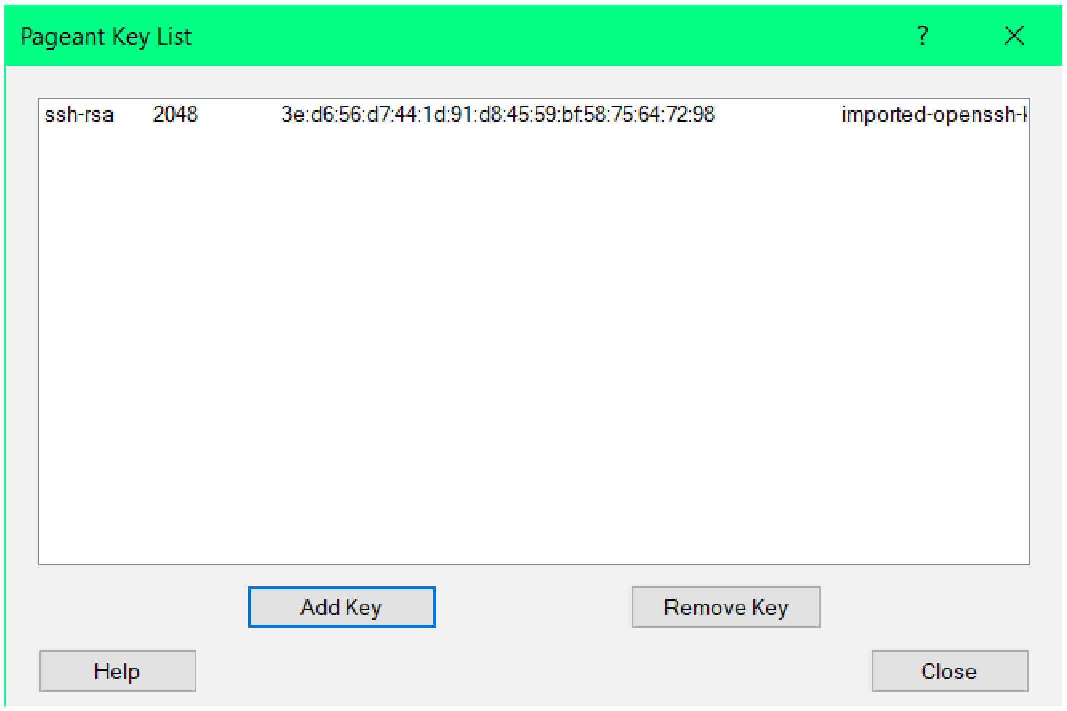


	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP	IPv6 IP
■	Public Instance	i-075e1a9889e1b23bf	t2.micro	us-east-1a	● running	✓ 2/2 checks ...	None	ec2-18-207-102-58.co...	18.207.102.58	-
	Private Insta...	i-0c7b6a7c20fb3d63a	t2.micro	us-east-1b	● running	✓ 2/2 checks ...	None		-	-

Instance: i-075e1a9889e1b23bf (Public Instance) **Public DNS:** ec2-18-207-102-58.compute-1.amazonaws.com

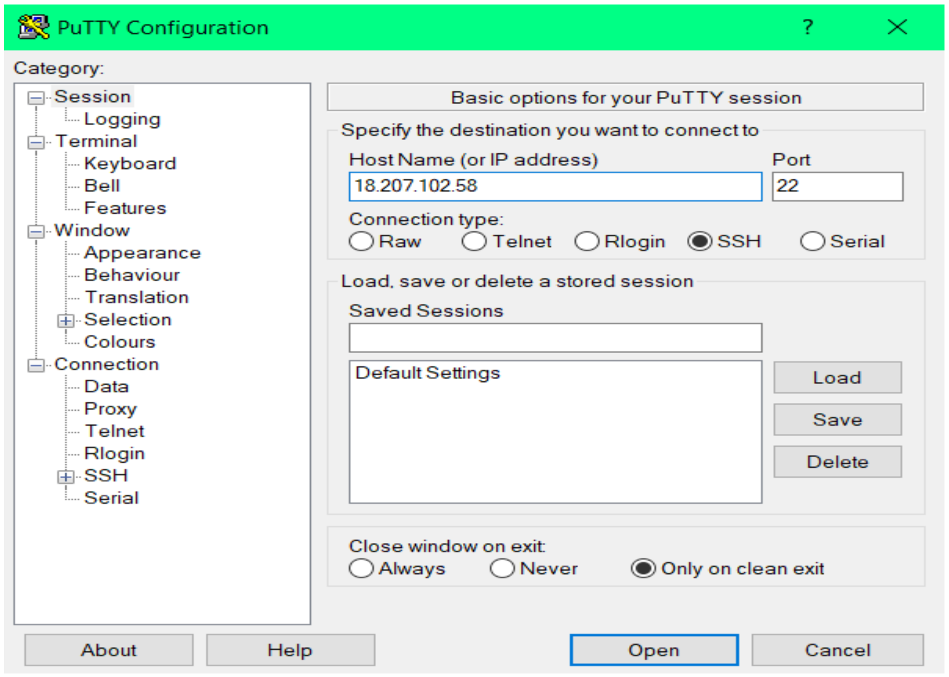
Instance ID	i-075e1a9889e1b23bf	Public DNS (IPv4)	ec2-18-207-102-58.compute-1.amazonaws.com
Instance state	running	IPv4 Public IP	18.207.102.58
Instance type	t2.micro	IPv6 IPs	-
Finding	Opt-in to AWS Compute Optimizer for recommendations. Learn more	Elastic IPs	
Private DNS	ip-10-0-0-112.ec2.internal	Availability zone	us-east-1a

In order to login to these instance SSH is done using the Putty tool by the utilizing the Public and Private key pair. However, our private EC2 instance does not have a public IP thus, we use the Pageant tool to upload the private key of our private EC2 instance. Before uploading, we convert our private key from pem format to ppk format using the PuttyGen tool.



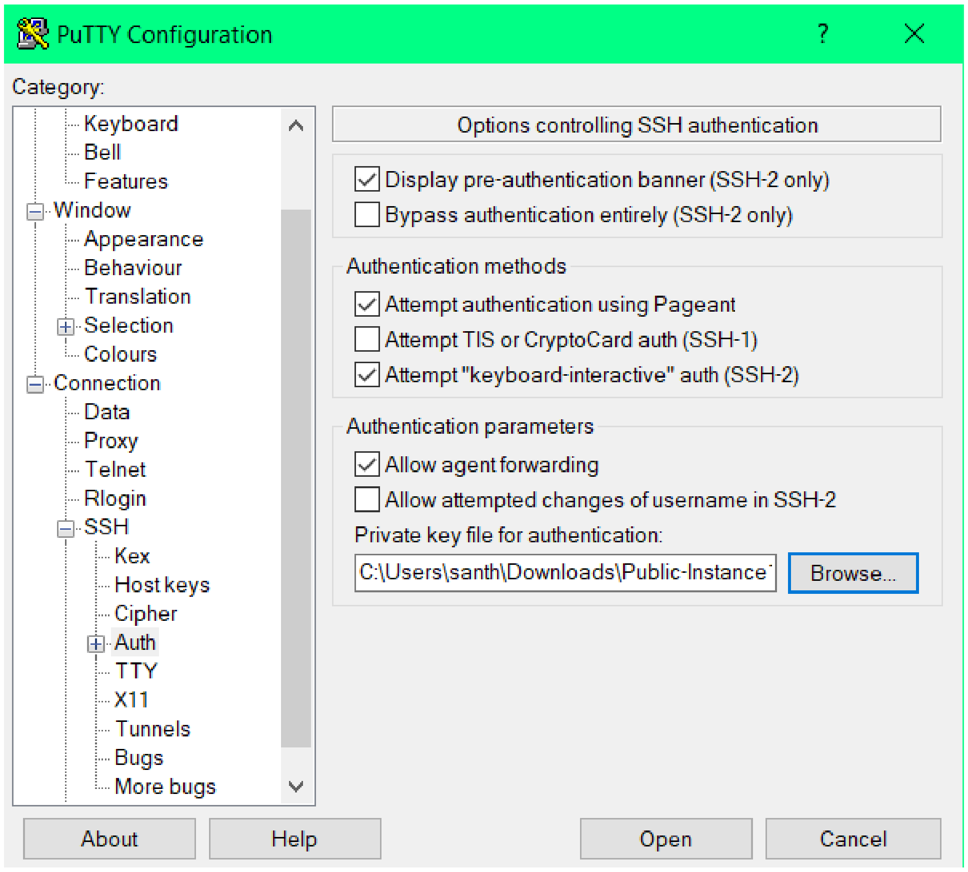
```
ssh-rsa    2048    3e:d6:56:d7:44:1d:91:d8:45:59:bf:58:75:64:72:98    imported-openssh-k
```

After uploading, we open the Putty tool and enter the public IP of the public EC2 instance. Under SSH authorization we enable agent forwarding and upload the private key of our public EC2 instance.



Host Name (or IP address): 18.207.102.58
Port: 22
Connection type: Raw Telnet Rlogin [✓ SSH] Serial
Close window on exit: Always Never [✓ Only on clean exit]

Under SSH→Auth, we enable Agent Forwarding and upload the private key of our public EC2 instance.



Options controlling SSH authentication

- [✓] Display pre-authentication banner (SSH-2 only)
- [ ] Bypass authentication entirely (SSH-2 only)

Authentication methods

- [✓] Attempt authentication using Pageant
- [ ] Attempt TIS or CryptoCard auth (SSH-1)
- [✓] Attempt "keyboard-interactive" auth (SSH-2)

Authentication parameters

- [✓] Allow agent forwarding
- [ ] Allow attempted changes of username in SSH-2

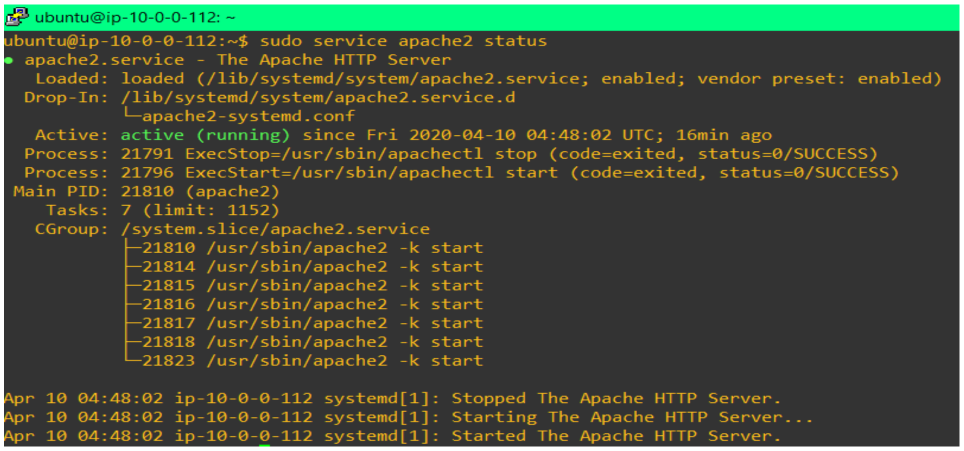
Private key file for authentication:

C:\Users\santh\Downloads\Public-Instance

We enter the username as "ubuntu" and login to our public instance. In order to deploy a web server, we use the following command:

```
$sudo apt-get install apache2 libapache2-mod-php php
```

After installation, we can check to see if the status is active



```
ubuntu@ip-10-0-0-112: ~
ubuntu@ip-10-0-0-112:~$ sudo service apache2 status
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
  Drop-In: /lib/systemd/system/apache2.service.d
           └─apache2-systemd.conf
   Active: active (running) since Fri 2020-04-10 04:48:02 UTC; 16min ago
  Process: 21791 ExecStop=/usr/sbin/apachectl stop (code=exited, status=0/SUCCESS)
  Process: 21796 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
 Main PID: 21810 (apache2)
    Tasks: 7 (limit: 1152)
   CGroup: /system.slice/apache2.service
           ├─21810 /usr/sbin/apache2 -k start
           ├─21814 /usr/sbin/apache2 -k start
           ├─21815 /usr/sbin/apache2 -k start
           ├─21816 /usr/sbin/apache2 -k start
           ├─21817 /usr/sbin/apache2 -k start
           ├─21818 /usr/sbin/apache2 -k start
           └─21823 /usr/sbin/apache2 -k start

Apr 10 04:48:02 ip-10-0-0-112 systemd[1]: Stopped The Apache HTTP Server.
Apr 10 04:48:02 ip-10-0-0-112 systemd[1]: Starting The Apache HTTP Server...
Apr 10 04:48:02 ip-10-0-0-112 systemd[1]: Started The Apache HTTP Server.
```

We now SSH into the private instance by using the following command.

```
$ssh ubuntu@ip-10-0-3-199.ec2.internal
```

To install the MySQL server, the following command is executed.

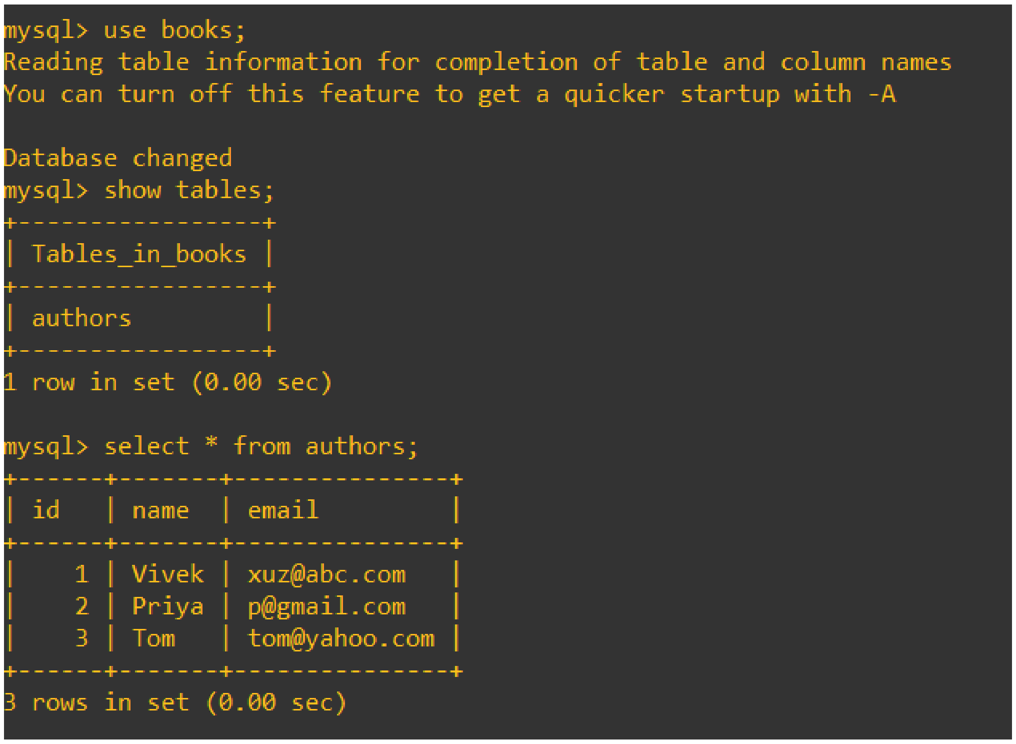
```
$sudo apt-get install mysql-server
```

After logging in to the MySQL server we create a database and table using the following commands:

```
mysql> CREATE DATABASE books;
```

```
mysql> CREATE TABLE authors;
```

We insert required data into the table.



```
mysql> use books;
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
mysql> show tables;
+-----------------+
| Tables_in_books |
+-----------------+
| authors         |
+-----------------+
1 row in set (0.00 sec)

mysql> select * from authors;
+------+-------+---------------+
| id   | name  | email         |
+------+-------+---------------+
|    1 | Vivek | xuz@abc.com   |
|    2 | Priya | p@gmail.com   |
|    3 | Tom   | tom@yahoo.com |
+------+-------+---------------+
3 rows in set (0.00 sec)
```

We create a user and grant permissions using the following commands:

```
mysql> CREATE USER “testing”@”10.0.0.112” IDENTIFIED BY “******”;
```

```
mysql> GRANT ALL PRIVILEGES ON *.* TO “testing”@”10.0.0.112”;
```

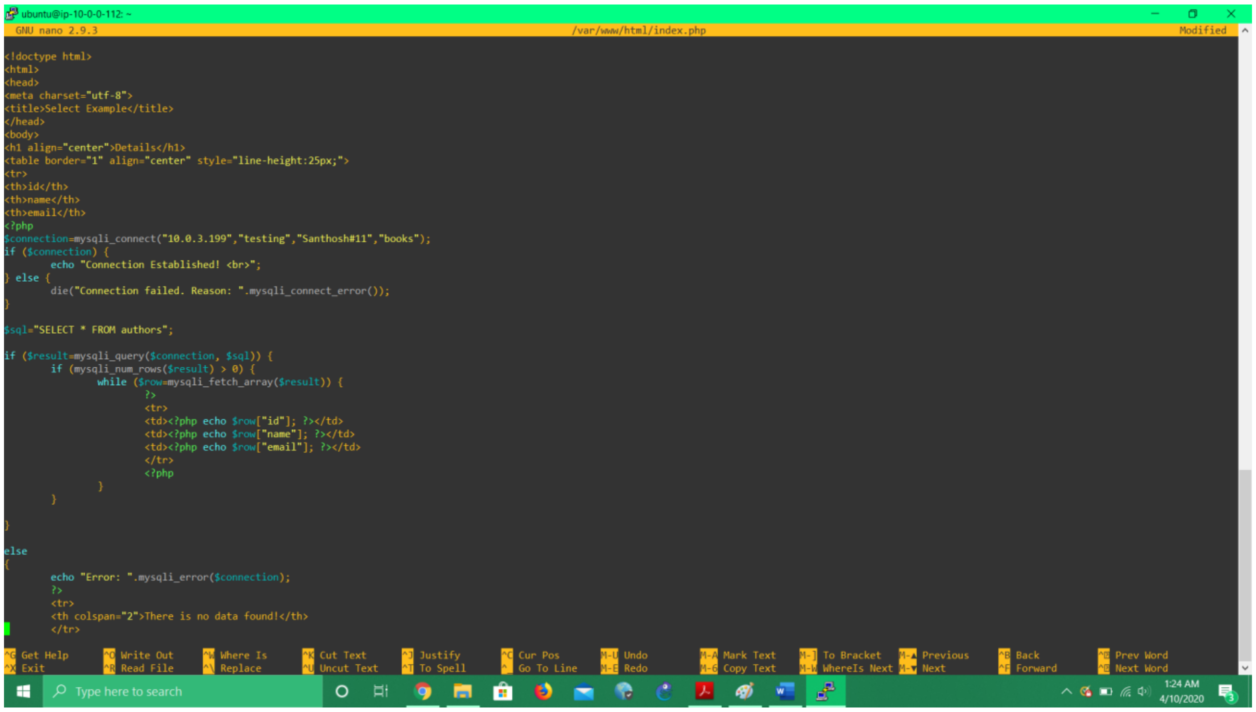
```
mysql> FLUSH PRIVILEGES;
```

We open the file /etc/mysql/mysql.conf.d/mysqld.cnf and comment the bind-address.

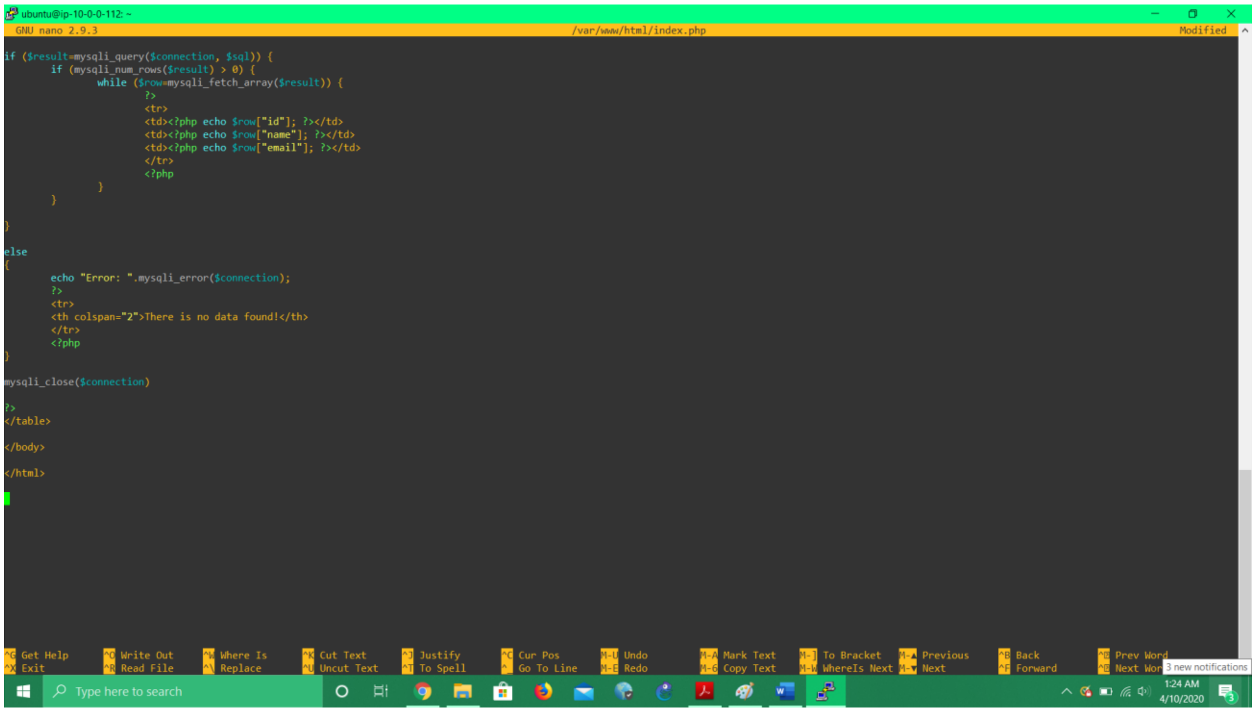
Restart the MySQL server using the following command:

$sudo service mysql restart

We then get into our public instance and write a PHP code to fetch and display data from the MySQL table in the path /var/www/html/index.php

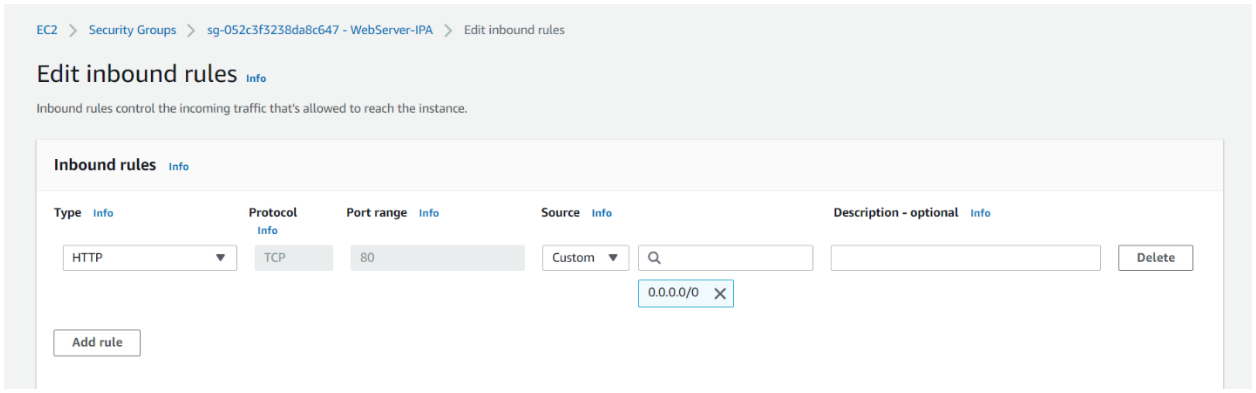


```
ubuntu@ip-10-0-0-112: ~
  GNU nano 2.9.3                              /var/www/html/index.php                              Modified

<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Select Example</title>
</head>
<body>
<h1 align="center">Details</h1>
<table border="1" align="center" style="line-height:25px;">
<tr>
<th>id</th>
<th>name</th>
<th>email</th>
<?php
$connection=mysqli_connect("10.0.3.199","testing","Santhosh#11","books");
if ($connection) {
        echo "Connection Established! <br>";
} else {
        die("Connection failed. Reason: ".mysqli_connect_error());
}

$sql="SELECT * FROM authors";

if ($result=mysqli_query($connection, $sql)) {
        if (mysqli_num_rows($result) > 0) {
                while ($row=mysqli_fetch_array($result)) {
                        ?>
                        <tr>
                        <td><?php echo $row["id"]; ?></td>
                        <td><?php echo $row["name"]; ?></td>
                        <td><?php echo $row["email"]; ?></td>
                        </tr>
                        <?php
                }
        }

}

else
{
        echo "Error: ".mysqli_error($connection);
        ?>
        <tr>
        <th colspan="2">There is no data found!</th>
        </tr>
```



```
ubuntu@ip-10-0-0-112: ~
  GNU nano 2.9.3                              /var/www/html/index.php                              Modified

if ($result=mysqli_query($connection, $sql)) {
        if (mysqli_num_rows($result) > 0) {
                while ($row=mysqli_fetch_array($result)) {
                        ?>
                        <tr>
                        <td><?php echo $row["id"]; ?></td>
                        <td><?php echo $row["name"]; ?></td>
                        <td><?php echo $row["email"]; ?></td>
                        </tr>
                        <?php
                }
        }

}

else
{
        echo "Error: ".mysqli_error($connection);
        ?>
        <tr>
        <th colspan="2">There is no data found!</th>
        </tr>
        <?php
}

mysqli_close($connection)

?>
</table>

</body>

</html>
```

We also have to update Security Groups to our EC2 instances to allow the necessary connections.

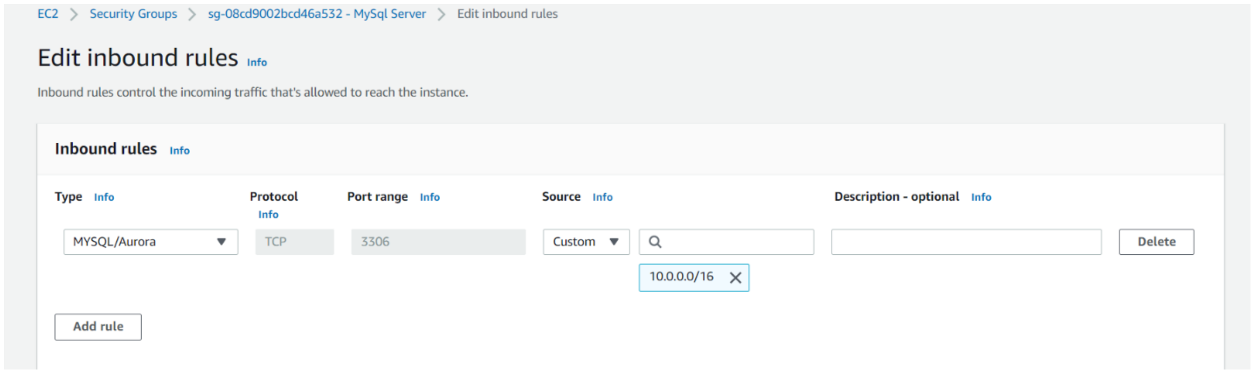

EC2 > Security Groups > sg-052c3f3238da8c647 - WebServer-IPA > Edit inbound rules

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Type	Protocol	Port range	Source		Description - optional
HTTP	TCP	80	Custom	0.0.0.0/0	


EC2 > Security Groups > sg-08cd9002bcd46a532 - MySql Server > Edit inbound rules

Edit inbound rules Info

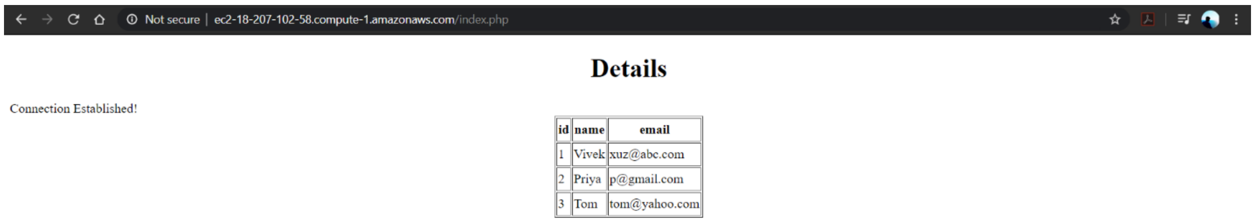
Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Type	Protocol	Port range	Source		Description - optional
MYSQL/Aurora	TCP	3306	Custom	10.0.0.0/16	

After making all the necessary changes we copy the public IP or public DNS of our public EC2 instance provided by AWS and paste it in a web browser

Output:


Not secure | ec2-18-207-102-58.compute-1.amazonaws.com/index.php

Details

Connection Established!

id	name	email
1	Vivek	xuz@abc.com
2	Priya	p@gmail.com
3	Tom	tom@yahoo.com