LEMP STACK PROJECT IMPLEMENTATION

The main aim for this project is to explain the DevOps concepts and processes using a LEMP web stack. Some developers use this set of framework and tools to develop software products .We would be carrying out this project in the AWS platform

LEMP is an acronym of sets of technologies used to develop a technical software product.

Linux

NGINX

MySQL

PHP

(P could also stand for Python or Perl)

NGINX is pronounced engineX where the acronym “E” originated from

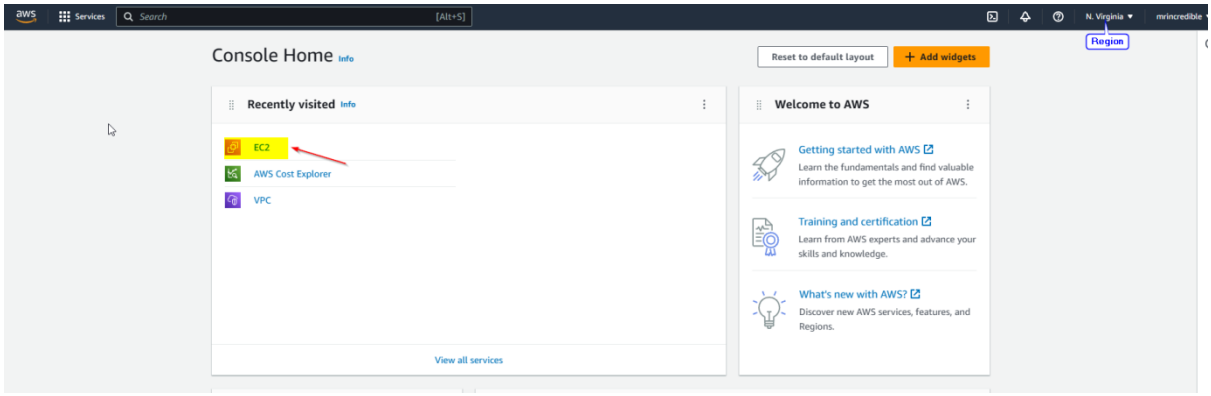
Pre-requisite for the projects is the following.

1) **Fundamental Knowledge of Installing and downloading software**
2) **Basic Understanding of Linux Commands**
3) **AWS account login with EC2 instance**
4) **Internet connection**

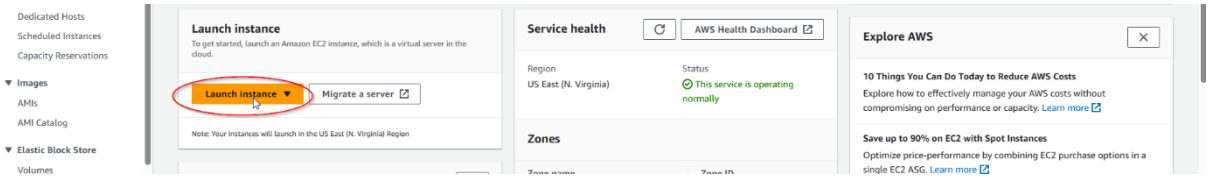
IMPLEMENTATION STEPS:

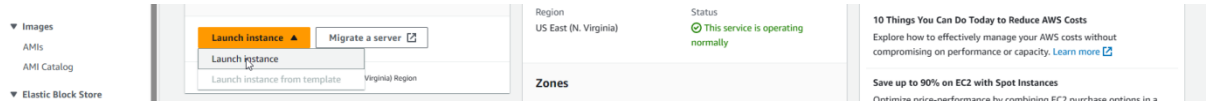
i) **Ensure you login with your details to your AWS console via the https://aws.amazon.com**

ii) **Click on the EC2 link to create instances.**



iii)Click on launch instance dropdown button and select launch instance .

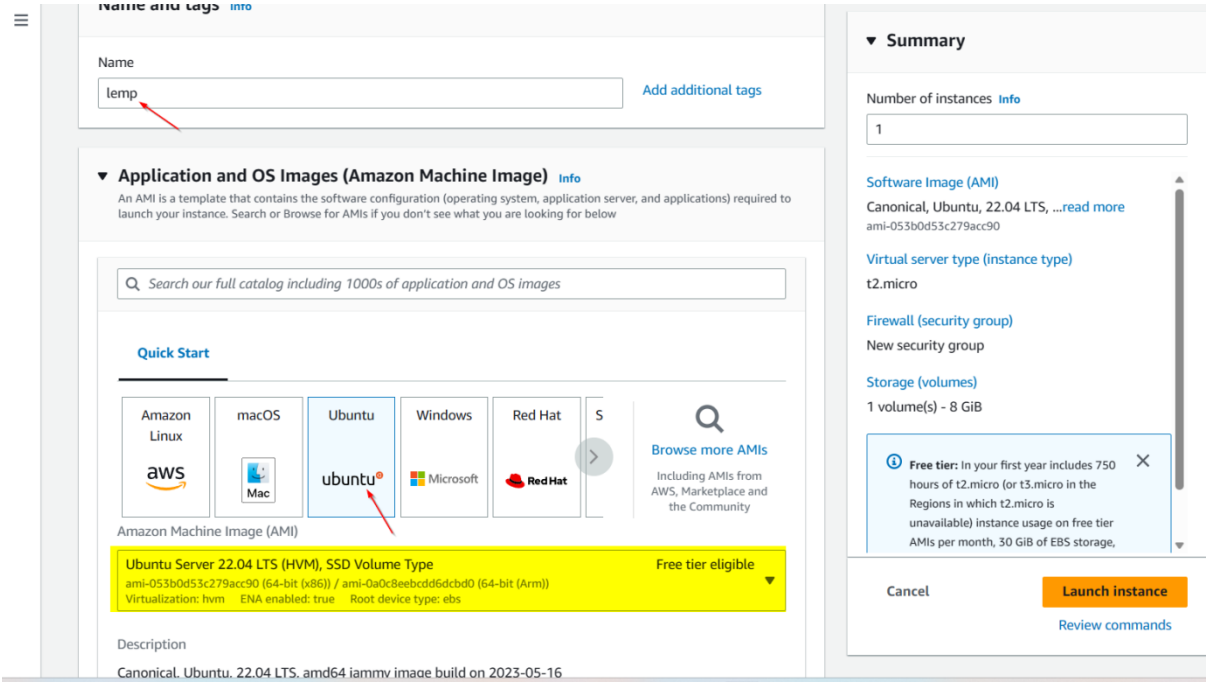




iv)Fill in all relevant details to the LEMP project such as :

Type in the name and additional tag to the project (lemp) .Select ubuntu from the quick start option .Also note that the Amazon machine image selection varies from user to user

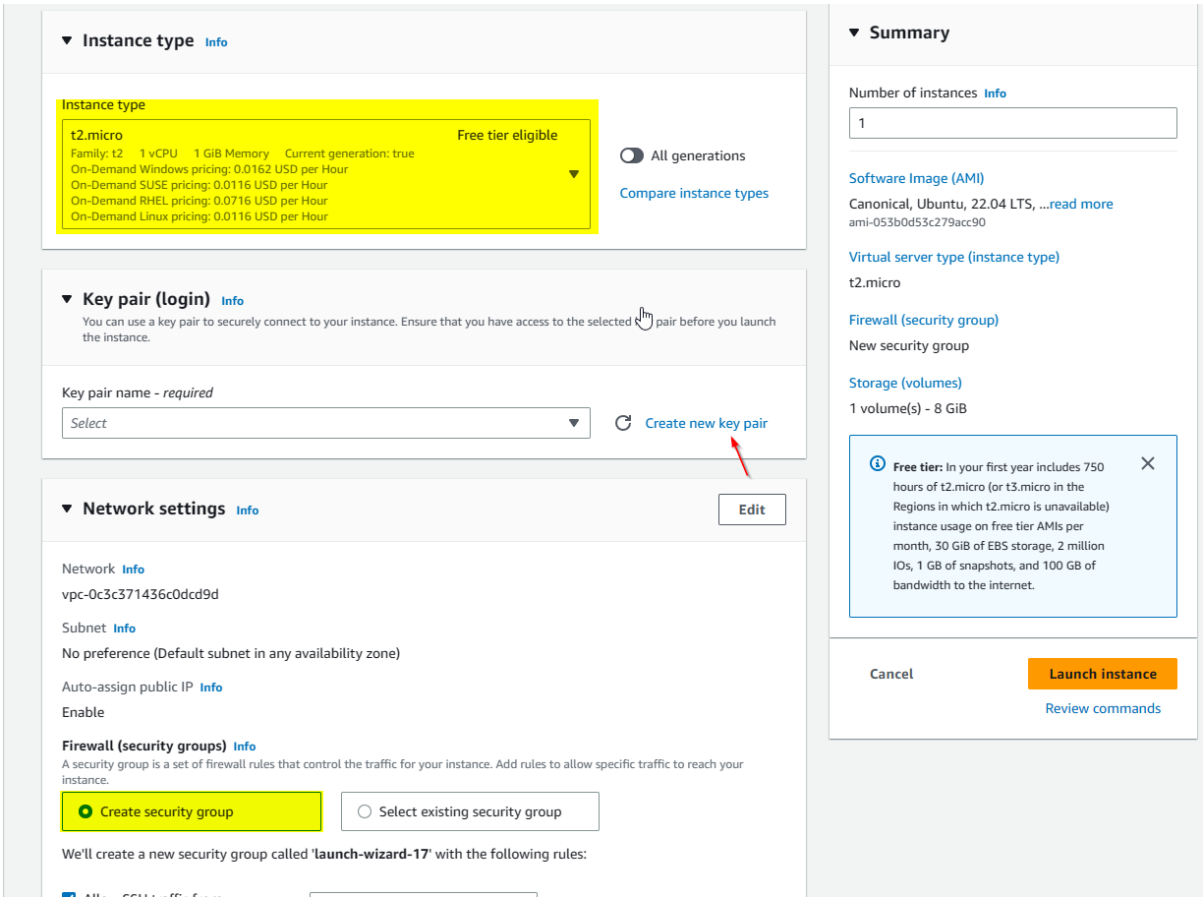
Select Ubuntu server 20.04 LTS (HVM),SSD Volume Type (Free Tier)



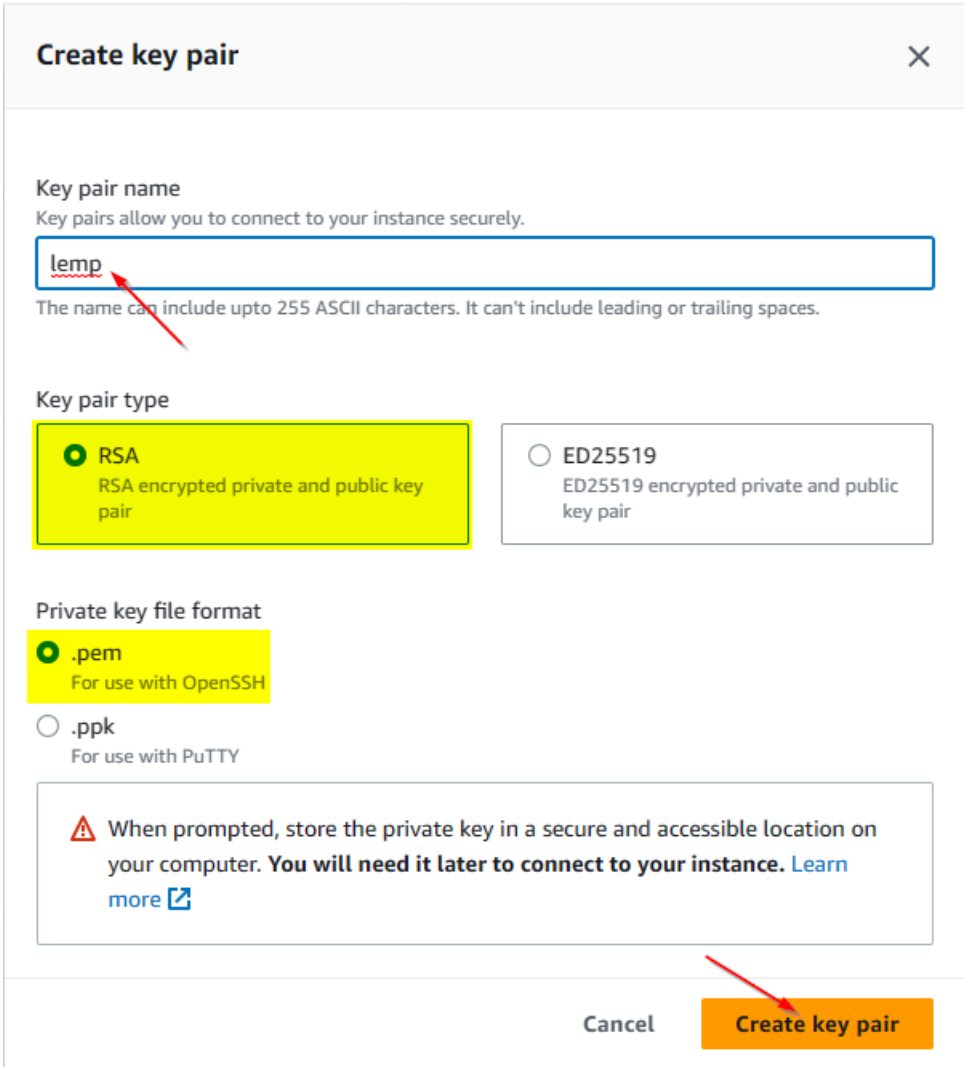
v)The instance type selected in the configuration is the t2 micro -free tier.

Click on the “Create new key pair” link.

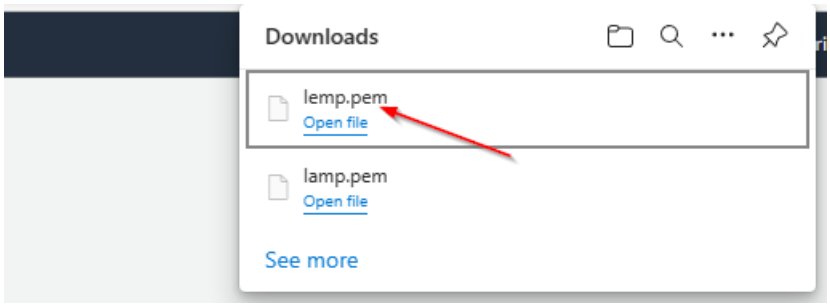
Ensure the Checkbox remains unchanged on the “Create security group”.



vi)Type in the key pair name, chose the default key pair type and private key file format (rsa and .pem) and clicked the “Create key pair button”

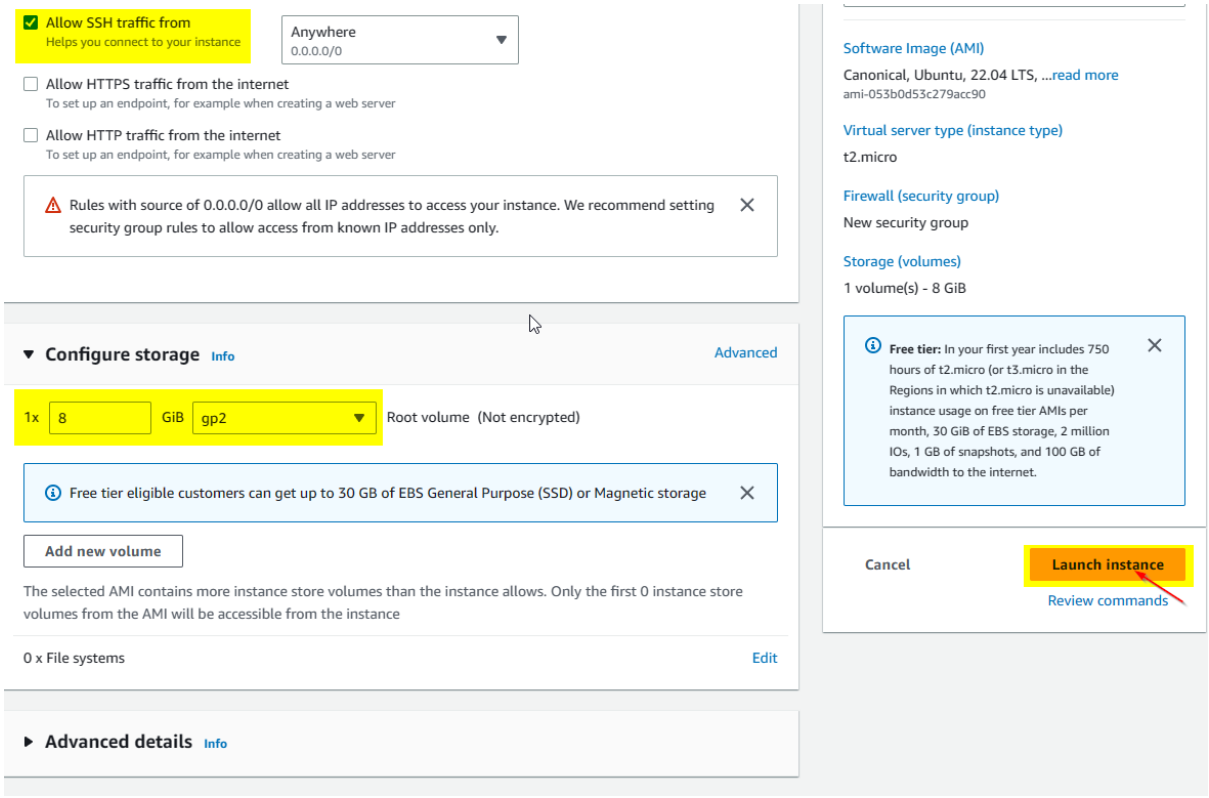


vii) The .pem file was downloaded successfully

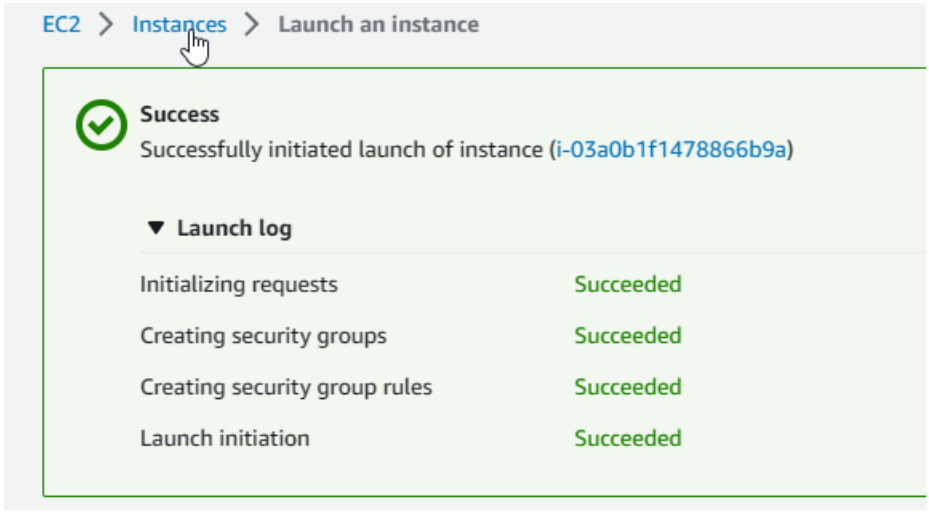


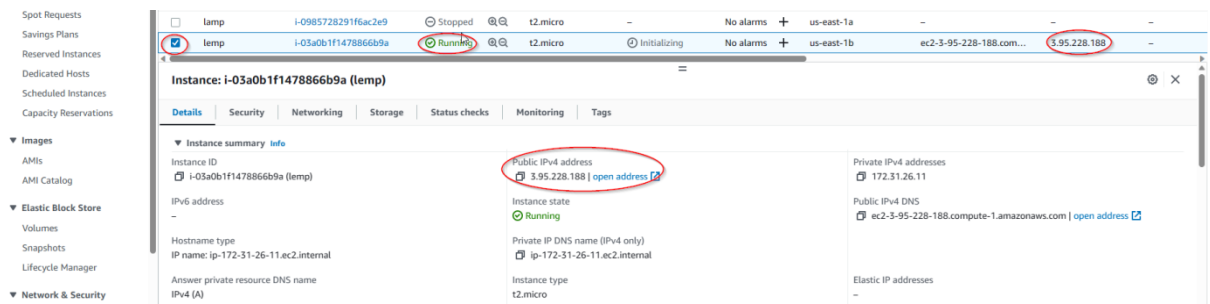
viii) I have deliberately chosen default settings to allow SSH traffic from anywhere as well as the storage volume given by AWS.

Then we proceed to launch our instance finally.

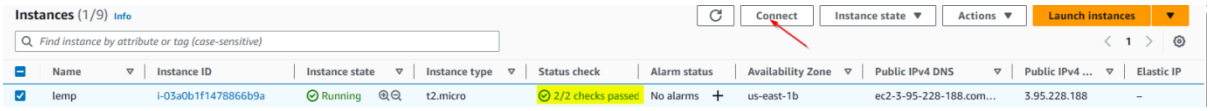


Instance successfully launched and click to view Instance details with the IP address.

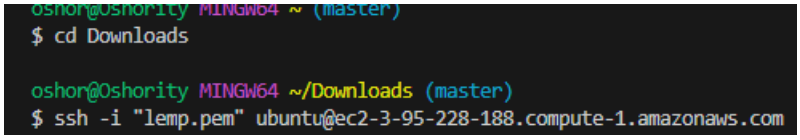




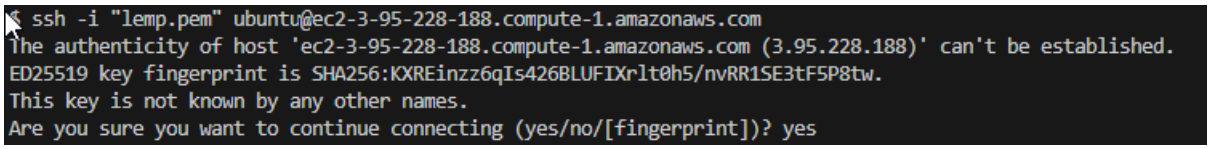
Click the “Connect” button and copy the ssh client details we would be using on the git bash console.



Open git bash on visual studio code or whichever console is convenient to use. We are using git bash here with Visual Studio Code

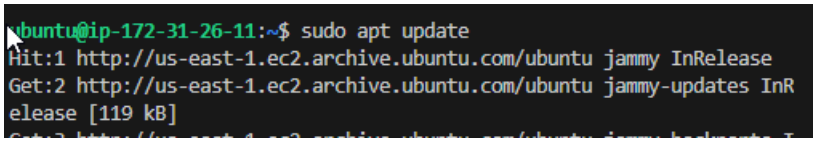


Type YES to connect.

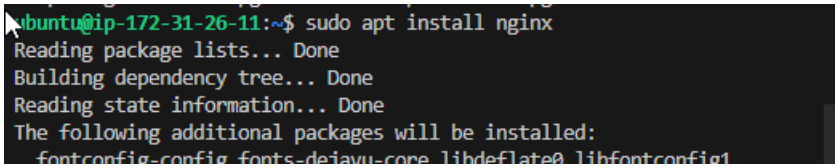


You have successful connected to the EC2 instance launched on AWS via ssh

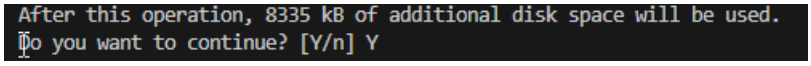
Type clear to have a clear console and proceed to updating the lists of packages in the package manager.



Then we run apache2 installation and click yes to complete installation

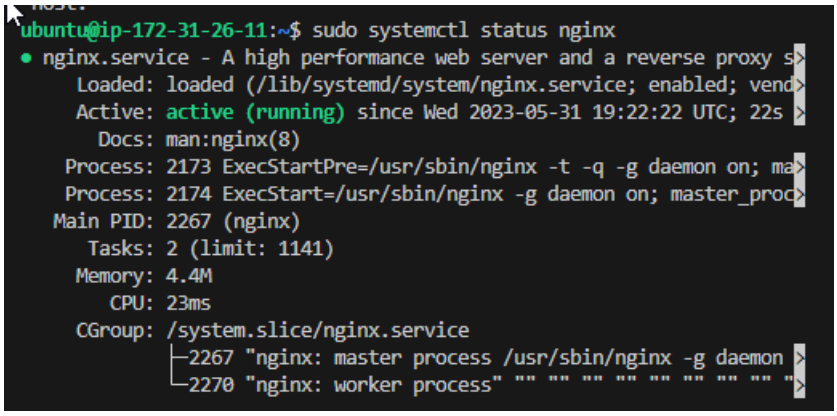


Type YES to continue NGINX installation.



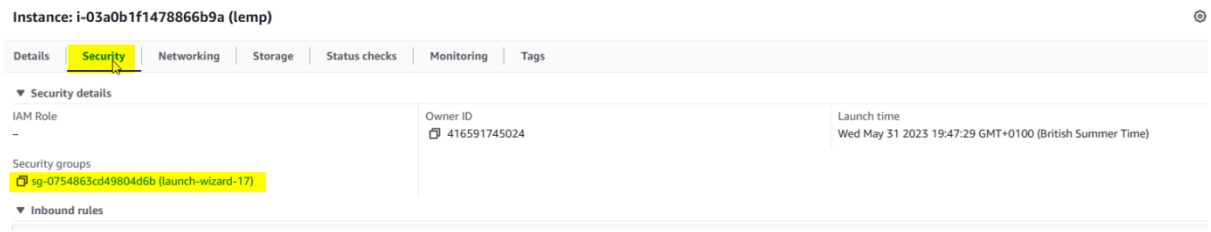
We have to verify that Nginx is running in our Operating System and press the

Ctl + C button to get the ubuntu root.

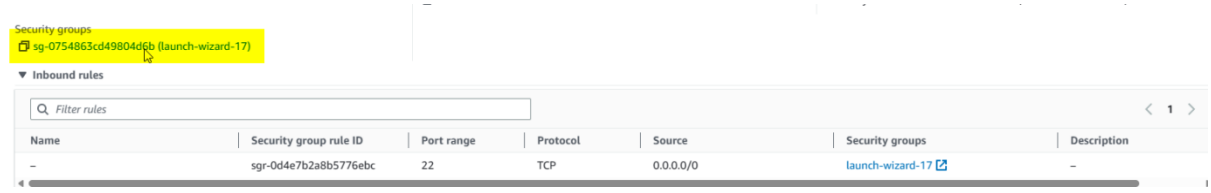


To proceed by launching the web server in the AWS Cloud, we need to navigate back to the security group on the platform to add a new rule for TCP port 80 which is the default for web browsers. Once done we can access the web page on internet.

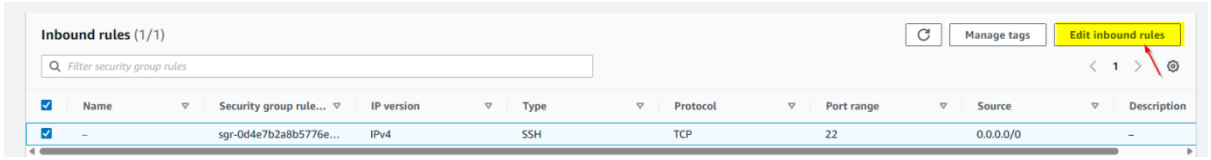
Click on security button.



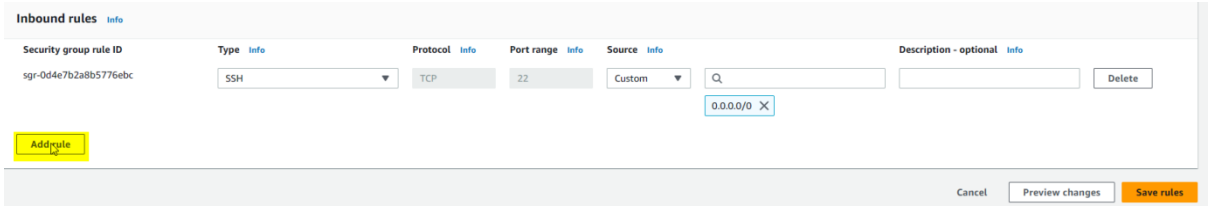
And click the security group link.



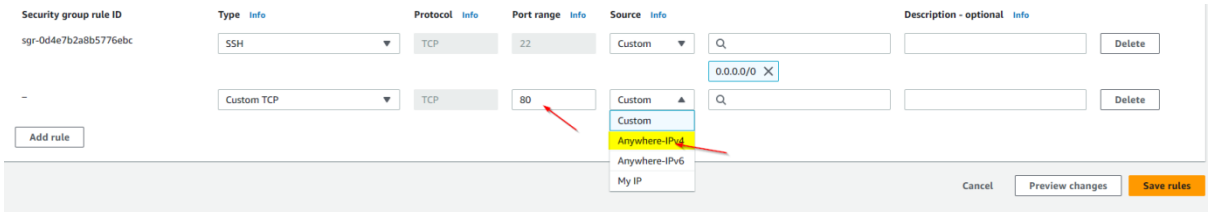
Click on “Edit inbound rules “in order to add a new rule for port 80



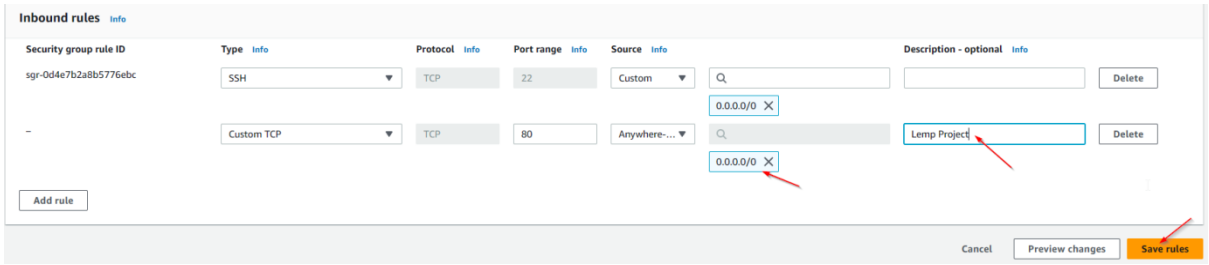
Add a new rule.



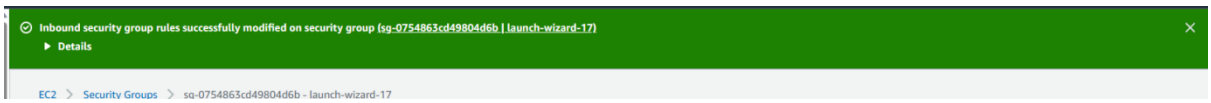
Type in the port range and click “Anywhere ipv4”



Click the “Save rules” Button



Inbound rule successfully modified



Open any browser of your choice and access the URL http://3.95.228.188:80

Nginx default page successfully displayed.

From the LEMP stack, we have implemented with Linux and now have Nginx ready .

Next step would be to get the MySQL installed .

MYSQL INSTALLATION

Now that our web server is running, we need a relational database uses within the PHP environment hence we install MySQL server
Type “Y” and press Enter

```
ubuntu@ip-172-31-26-11:~$ sudo apt install mysql-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

When installation is finished, Log in to connect to the MySQL server as the administrator user root so that you can have access to the sudo command.

```
No VM guests are running outdated hypervisor (qemu) binaries on this
 host.
ubuntu@ip-172-31-26-11:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.33-0ubuntu0.22.04.2 (Ubuntu)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

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

It is important to set up a password for the user root using mysql_native_password as a default authentication method.
Please note, Password not revealed for security purpose
Exit MySQL

```
mysql> ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'PassWord.1';
Query OK, 0 rows affected (0.01 sec)

mysql> exit
Bye
```

Interactive script is started and all modifications are answered with a Y/N response .
Root user password was set Validate password: No
Change password: No
Remove anonymous user: No

```
buntu@ip-172-31-26-11:~$ sudo mysql_secure_installation

Securing the MySQL server deployment.

Enter password for user root:

VALIDATE PASSWORD COMPONENT can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD component?

Press y|Y for Yes, any other key for No: No
Using existing password for root.
Change the password for root ? ((Press y|Y for Yes, any other key for No) : No

 ... skipping.
By default, a MySQL installation has an anonymous user,
allowing anyone to log into MySQL without having to have
a user account created for them. This is intended only for
testing, and to make the installation go a bit smoother.
You should remove them before moving into a production
environment.

Remove anonymous users? (Press y|Y for Yes, any other key for No) : No

 ... skipping.
```

Disallow remote login: No

Remove test data base and access to it: No

Reload Privilege tables: Yes

```
Normally, root should only be allowed to connect from
'localhost'. This ensures that someone cannot guess at
the root password from the network.

Disallow root login remotely? (Press y|Y for Yes, any other key for No) : No

 ... skipping.
By default, MySQL comes with a database named 'test' that
anyone can access. This is also intended only for testing,
and should be removed before moving into a production
environment.
ey for No) : No

 ... skipping.
Reloading the privilege tables will ensure that all changes
made so far will take effect immediately.

Reload privilege tables now? (Press y|Y for Yes, any other key for No) : Yes
Success.

All done!
```

Verify login details to ensure all details were inputted correctly and exiting MySQL

```
ubuntu@ip-172-31-26-11:~$ sudo mysql -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 15
Server version: 8.0.33-0ubuntu0.22.04.2 (Ubuntu)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

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> exit
Bye
ubuntu@ip-172-31-26-11:~$
```

MySQL server is now installed and secured .

Next, we proceed to the PHP installation which is the final component of the LEMP STACK

PHP INSTALLATION

PHP is the component that would process the codes to display dynamic content to the end user. Nginx requires an external program to handle the PHP processing and this act as a bridge between PHP interpreter and the web server .This enhances the overall performance for most PHP web based site .It is called the PHP fastCGI process manager)

Hence ,we would need to install 2 packages namely :

1)php-fpm 2) php-mysql .

```
ubuntu@ip-172-31-26-11:~$ sudo apt install php-fpm php-mysql
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

Installation continues.

```
Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this
 host.
ubuntu@ip-172-31-26-11:~$
```

At this point the PHP component is installed but we need to configure the Nginx to use the PHP processor

CONFIGURING NGINX TO USE PHP PROCESSOR

We need to create server block to encapsulate the Nginx configuration details and it can host more than one domain on a single server .Therefore we would create a direct structure with the /var/www for the domain website .

Root web directory created for our domain below.

```
ubuntu@ip-172-31-26-11:~$ sudo mkdir /var/www/projectLEMP
ubuntu@ip-172-31-26-11:~$ sudo chown -R $USER:$USER /var/www/projectLE
MP
```

Then proceed to open new nginx in sites-available directory using the vim editor

```
ubuntu@ip-172-31-26-11:~$ sudo vim /etc/nginx/sites-available/projectL
EMP
```

Put the edited file in an insert mode by typing “i” without quotes and add the bare-bones configuration files ,press ESC ,save and exit with “ :wq” command

```
#/etc/nginx/sites-available/projectLEMP

server {
    listen 80;
    server_name projectLEMP www.projectLEMP;
    root /var/www/projectLEMP;

    index index.html index.htm index.php;

    location / {
        try_files $uri $uri/ =404;
    }

    location ~ \.php$ {
        include snippets/fastcgi-php.conf;
        fastcgi_pass unix:/var/run/php/php8.1-fpm.sock;
     }

    location ~ /\.ht {
        deny all;
    }

}
~
~
~
~
~
:wq
```

Please observe the following functions the location block carries out:

listen: Port Nginx listen to port 80

root: defines where the document root is served and stored by the website

index: defines how Nginx prioritize the index files. Index .html file gets the highest priority than the rest of the indexes.

Server-name: The domain name or Ip address the server block is responsible for .

Location: Includes try file directives that checks the files that correlates with the matching URI request. If not found, it will return Error 404

Location ~\.php$: Handles the PHP processing and ensures it points Nginx to the fastcgi-php.conf files and declares what socket is associated with php-fpm

Location ~ /\.ht : deals with .htaccess files that Nginx does not process by denying all directives and ensuring they are not served to users .

Now we have to activate our configuration by linking the config files from Nginx site –enabled directory and also test the configuration to know if the syntax are OK

```
ubuntu@ip-172-31-26-11:~$ sudo ln -s /etc/nginx/sites-available/projec
tLEMP /etc/nginx/sites-enabled/
ubuntu@ip-172-31-26-11:~$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
ubuntu@ip-172-31-26-11:~$
```

Test successful and syntax are okay.

Next step is disabling default nginx host that is currently configured to listen to port 80.

```
ubuntu@ip-172-31-26-11:~$ sudo unlink /etc/nginx/sites-enabled/default

ubuntu@ip-172-31-26-11:~$
```

Proceed to reload Nginx to apply all changes.

```
ubuntu@ip-172-31-26-11:~$ sudo systemctl reload nginx
ubuntu@ip-172-31-26-11:~$
```

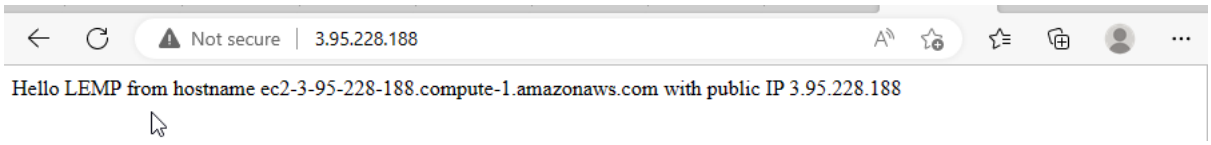
Hence , Our new website is active but note that the /var/www/projectLemp is empty.

We should create a file in that location so that we can test that the blocks is working as expected .

```
ubuntu@ip-172-31-26-11:~$ sudo echo 'Hello LEMP from hostname' $(curl
-s http://169.254.169.254/latest/meta-data/public-hostname) 'with publ
ic IP' $(curl -s http://169.254.169.254/latest/meta-data/public-ipv4)
> /var/www/projectLEMP/index.html
ubuntu@ip-172-31-26-11:~$
```

Now go and launch your browser URL using the IP address.

http://3.95.228.188:80



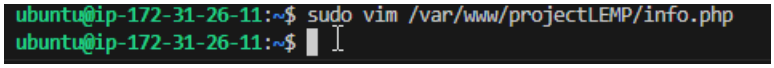
Our LEMP stack is fully configured successfully.

Next would be to create a PHP Script to test that our Nginx can handle .php files within the new configured website.

TESTING PHP WITH NGINX

Our LEMP stack is set up and completely installed and fully operational. We would test to validate that Nginx can handle .php files off to our PHP processor.

Create a new file called info.php and paste the file



Paste the simplest valid php code that would return information about your server

Let us access this page on our web browser with the endpoint /info.php

http://3.95.228.188/info.php

PHP Version 8.1.2-1ubuntu2.11

System	Linux ip-172-31-26-11 5.19.0-1025-aws #26~22.04.1-Ubuntu SMP Mon Apr 24 01:58:15 UTC 2023 x86_64
Build Date	Feb 22 2023 22:56:18
Build System	Linux
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/8.1/fpm

A webpage containing details information about our server should be successfully displayed .

Please note: After checking the relevant information about your php .Its best to remove the file created as it contains sensitive information about your PHP environment and your ubuntu server by the command below

sudo rm /var/www/your_domain/info.php

Next ,we would be retrieving data from MySQL database with PHP

RETRIEVING DATA FROM MYSQL DATABASE WITH PHP

We would be creating a test database with a simple To-do list and configure to access it ,so the Nginx would be able to query data from the database and display it .

A new user would be created with the “mysql_native_password” in order for it to connect to the MySQL database from PHP

Our new user: example_user

Our new database: example_database

We would connect to the root account by the command below

```
ubuntu@ip-172-31-26-11:~$ sudo mysql -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 17
Server version: 8.0.33-0ubuntu0.22.04.2 (Ubuntu)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

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 st
atement.

mysql>
```

Then we create a new database called “example_database” and password.

```
mysql> CREATE DATABASE `example_database`;
Query OK, 1 row affected (0.05 sec)

mysql> CREATE USER 'example_user'@'%' IDENTIFIED WITH mysql_native_pas
sword BY `[redacted]`
Query OK, 0 rows affected (0.04 sec)

mysql>  GRANT ALL ON example_database.* TO 'example_user'@'%';
Query OK, 0 rows affected (0.01 sec)
```

We grant all permissions to the new user over the new “example_database” in order to give the new user full privileges.

We exit the shell and test if the new user has actually been granted permission to be able to login to the MySQL console again and display the database.

```
mysql> exit
Bye
ubuntu@ip-172-31-26-11:~$ mysql
```

```
ubuntu@ip-172-31-26-11:~$ mysql -u example_user -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 19
Server version: 8.0.33-0ubuntu0.22.04.2 (Ubuntu)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

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 st
atement.

mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| example_database   |
| information_schema |
| performance_schema |
+--------------------+
3 rows in set (0.05 sec)

mysql>
```

Next, we create a test table named “todo_list” and input 4 different values on each section

```
mysql> CREATE TABLE example_database.todo_list (item_id INT AUTO_INCRE
MENT,content VARCHAR(255),PRIMARY KEY(item_id));
Query OK, 0 rows affected (0.04 sec)
```

```
mysql> INSERT INTO example_database.todo_list (content) VALUES ("My fi
rst important item");
Query OK, 1 row affected (0.02 sec)

mysql> INSERT INTO example_database.todo_list (content) VALUES ("My se
cond interesting sport");
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO example_database.todo_list (content) VALUES ("My th
ird  important decision");
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO example_database.todo_list (content) VALUES ("My fo
urth  important motivation");
Query OK, 1 row affected (0.01 sec)
```

We confirm that the data were successfully saved to our table below and exit the table.

```
mysql> SELECT * FROM example_database.todo_list;
+---------+--------------------------------+
| item_id | content                        |
+---------+--------------------------------+
|       1 | My first important item        |
|       2 | My second interesting sport    |
|       3 | My third  important decision   |
|       4 | My fourth  important motivation |
+---------+--------------------------------+
4 rows in set (0.00 sec)
```

```
mysql> exit
Bye
ubuntu@ip-172-31-26-11:~$
```

Next, we create a PHP Script that would connect to the MySQL database for our content in out root directory using vim editor to input the PHP script in the todo_list.php

```
Bye
ubuntu@ip-172-31-26-11:~$ sudo vim  /var/www/projectLEMP/todo_list.php
ubuntu@ip-172-31-26-11:~$
```

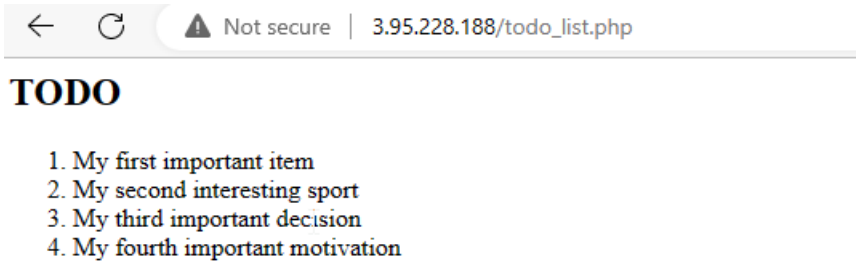
Save and close file when done editing.

```php
<?php
$user = "example_user";
$password = "password";
$database = "example_database";
$table = "todo_list";

try {
  $db = new PDO("mysql:host=localhost;dbname=$database", $user, $passw
ord);
  echo "<h2>TODO</h2><ol>";
  foreach($db->query("SELECT content FROM $table") as $row) {
    echo "<li>" . $row['content'] . "</li>";
  }
  echo "</ol>";
} catch (PDOException $e) {
    print "Error!: " . $e->getMessage() . "<br/>";
    die();
}
~
~
~
```

We can now access the page in the browser by visiting the IP address configured for the website follow by the endpoint /todo_list.php

http://3.95.228.188/todo_list.php



Test table have been successfully displayed on the webpage and our PHP environment has connected and was able interact with our MySQL server

Congratulations.