**What is a Technology stack?**  
A technology stack is a set of frameworks and tools used to develop a software product. This set of frameworks and tools are very specifically chosen to work together in creating well-functioning software. They are acronyms for individual technologies used together for a specific technology product. some examples are…

* **LAMP** (Linux, Apache, MySQL, PHP or Python, or Perl)
* **LEMP** (Linux, Nginx, MySQL, PHP or Python, or Perl)
* **MERN** (MongoDB, ExpressJS, ReactJS, NodeJS)
* **MEAN** (MongoDB, ExpressJS, AngularJS, NodeJS

However, on this project, I will be working on LAMP(Linux, Apache, MySQL, and PHP) Stack

Step 1 — Install Apache and update the firewall.

I’m using the AWS platform for this where I have launched an ec2 instance and I can have access to it on my local machine.

Run the following commands to install and verify apache has been installed

sudo apt update

sudo apt install apache2

sudo systemctl status apache2

The screenshot below confirms apache has been installed and its running

A computer screen with white text

Description automatically generated

let us try to check how we can access it locally in our Ubuntu shell, run:

curl <http://localhost:80> or curl <http://127.0.0.1:80>  
  
A screenshot of a computer program

Description automatically generated

Now it is time for us to test how our Apache HTTP server can respond to requests from the Internet.

Open a web browser of your choice and try to access the following URL *http://<Public-IP-Address>:80 or the Public IPv4 DNS(remember to remove the s in 'https' when doing that)*

A screenshot of a computer

Description automatically generated

Step 2 — Installing MySQL

sudo apt install mysql-server -y

Ran this command to install my sql server

sudo mysql

Used this command to log into the sql

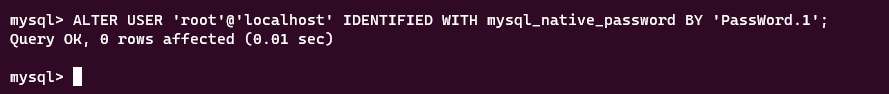
A screenshot of a computer screen

Description automatically generated

This command below will remove some insecure default settings and lock down access to your database system.

We’re defining this user’s password as PassWord.1 for the root user

ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql\_native\_password BY 'PassWord.1';



Then exit the shell by using ‘ctrl d’ or type exit

Then we run this script sudo mysql\_secure\_installation

This script configures security settings and allows you to: Set a password for root accounts and remove the root accounts accessible from outside the local host, remove anonymous user accounts, delete the test database, accessible by anonymous users, and reload the user privileges tables.

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

When you’re finished, test if you’re able to log in to the MySQL console by typing:

sudo mysql -p

A screenshot of a computer program

Description automatically generated

Step 3 — Installing PHP

Apache is set up to deliver your content, while MySQL is installed to oversee your data. PHP is responsible for handling code to present interactive information to users. Besides the php package, you'll require php-mysql, a PHP module facilitating interaction with MySQL databases. Also, libapache2-mod-php is needed to empower Apache in managing PHP files. The essential PHP core packages will be automatically installed as prerequisites.

To install these 3 packages at once, run:

sudo apt install php libapache2-mod-php php-mysql -y and confirm version by running this php -v

A computer screen with white text

Description automatically generated

Step 4 — Create a virtual host for your website using Apache

We will setup a virtual host to test the PHP script, virtual host enables you to setup multiple websites on a single server.

Create the directory for projectlamp using ‘mkdir’ command   
  
**sudo mkdir /var/www/projectlamp**

Create and open a new configuration file in Apache’s sites-available directory.

sudo vi /etc/apache2/sites-available/projectlamp.conf

This will create a new blank file, using ‘i’ to get to insert mode and paste the text below, the use :wq to write,save and quit

<VirtualHost \*:80>

ServerName projectlamp

ServerAlias www.projectlamp

ServerAdmin webmaster@localhost

DocumentRoot /var/www/projectlamp

ErrorLog ${APACHE\_LOG\_DIR}/error.log

CustomLog ${APACHE\_LOG\_DIR}/access.log combined

</VirtualHost>

Use ls command to list the new file in the sites-available directory.

sudo ls /etc/apache2/sites-available

The output of the above command should be similar to this below



Let us enable the new virtual host with the a2ensite command:

sudo a2ensite projectlamp

disable Apache’s default website use a2dissite command:

sudo a2dissite 000-default

To make sure your configuration file doesn’t contain syntax errors, run:

sudo apache2ctl configtest

Finally, reload Apache so these changes take effect:

sudo systemctl reload apache2

A screenshot of a computer program

Description automatically generated

Create an index.html file in that location so that we can test that the virtual host works as expected:

sudo echo 'Hello LAMP from hostname' $(curl -s http://169.254.169.254/latest/meta-data/public-hostname) 'with public IP' $(curl -s http://169.254.169.254/latest/meta-data/public-ipv4) > /var/www/projectlamp/index.html   
  
Then use the command: sudo ls -l /var/www/html/ to confirm the index file has been created

A screenshot of a computer

Description automatically generated

Go to your browser and try to open your website URL using IP address:

http://<Public-IP-Address>:80

A close-up of a computer screen

Description automatically generated

Step 5 — enable PHP on the website.

With the default DirectoryIndex settings on Apache, a file named index.html will always take precedence over an index.php file. This is useful for setting up maintenance pages in PHP applications, by creating a temporary index.html file containing an informative message to visitors. Because this page will take precedence over the index.php page, it will then become the landing page for the application. Once maintenance is over, the index.html is renamed or removed from the document root, bringing back the regular application page.

In case you want to change this behavior, you’ll need to edit the /etc/apache2/mods-enabled/dir.conf file and change the order in which the index.php file is listed within the DirectoryIndex directive:

sudo vim /etc/apache2/mods-enabled/dir.conf

<IfModule mod\_dir.c>

#Change this:

#DirectoryIndex index.html index.cgi index.pl index.php index.xhtml index.htm

#To this:

DirectoryIndex index.php index.html index.cgi index.pl index.xhtml index.htm

</IfModule>

After saving and closing the file, you will need to reload Apache so the changes take effect:

sudo systemctl reload apache2

Finally, we will create a PHP script to test that PHP is correctly installed and configured on your server.

Create a new file named index.php inside your custom web root folder:

vim /var/www/projectlamp/index.php

This will open a blank file. Add the following text, which is valid PHP code, inside the file:

<?php

phpinfo();

A computer screen with white text

Description automatically generated

When you are finished, save and close the file, refresh the page and you will see a page similar to this shown below:

A screenshot of a computer program

Description automatically generated

After checking the relevant information about your PHP server through that page, it’s best to remove the file you created as it contains sensitive information about your PHP environment -and your Ubuntu server. You can use rm to do so:

sudo rm /var/www/projectlamp/index.php