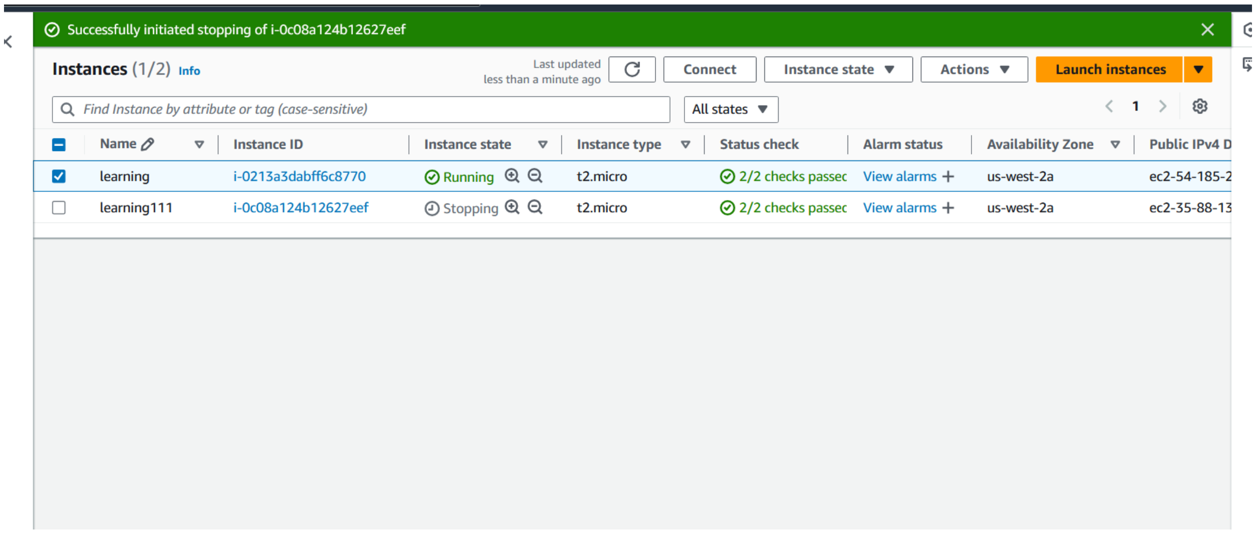
SETTING UP A LAMP STACK ON UBUNTU EC2 INSTANCE

STEP 1: LAUNCHING AN EC2 INSTANCE

The first step was to launch an EC2 instance from the t3.micro family with Ubuntu Server (HMV). I named the instance "learning" and selected the AMI: ubuntu/images/hvm-ssd-gp3/ubuntu-noble-24.04-amd64-server-20240801.



Successfully initiated stopping of i-0c08a124b12627eef

Instances (1/2) Info

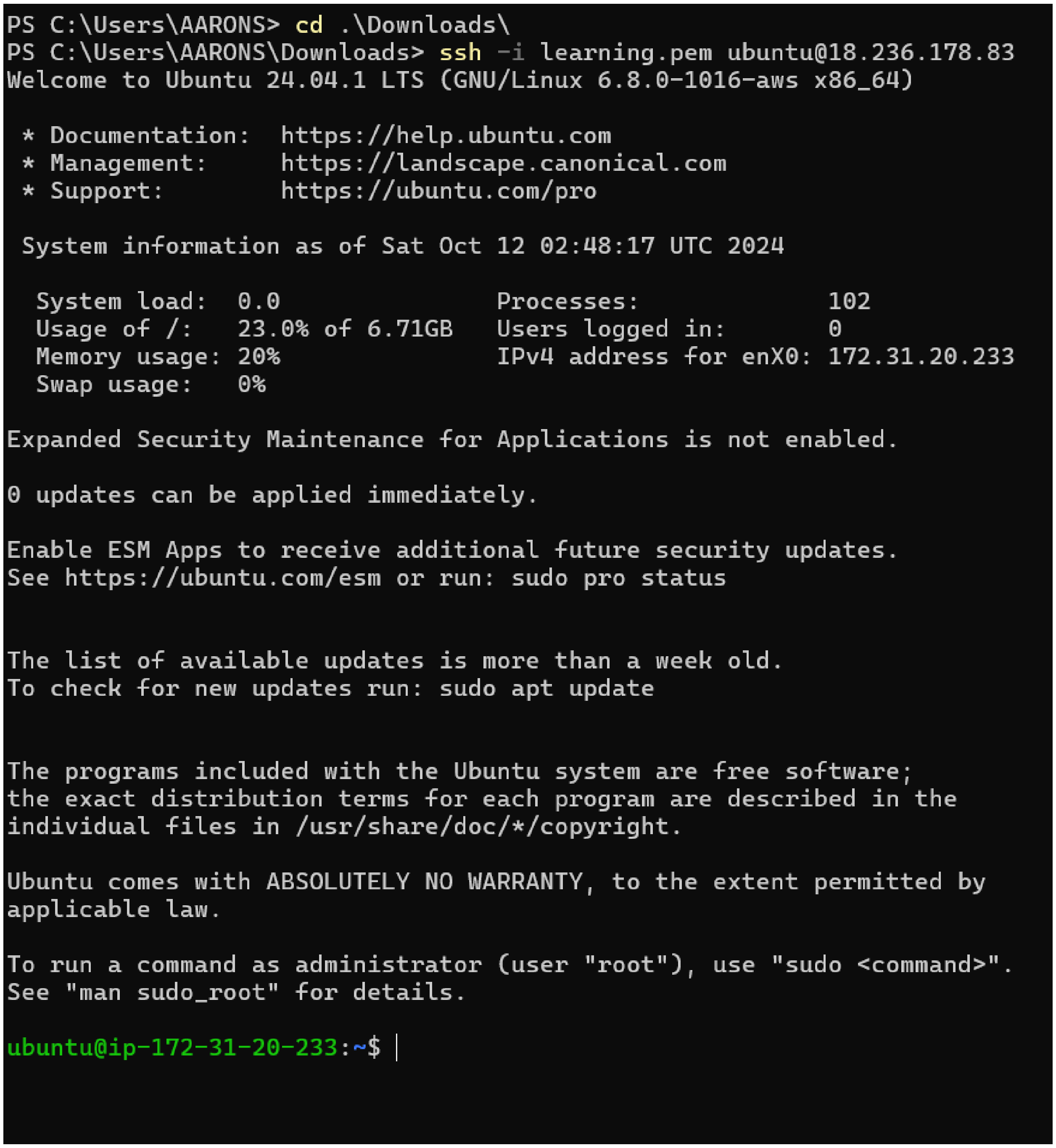
Last updated less than a minute ago

Find Instance by attribute or tag (case-sensitive)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 D
learning	i-0213a3dabff6c8770	Running	t2.micro	2/2 checks passec	View alarms	us-west-2a	ec2-54-185-2
learning111	i-0c08a124b12627eef	Stopping	t2.micro	2/2 checks passec	View alarms	us-west-2a	ec2-35-88-13

STEP 2: SSH ACCESS TO THE INSTANCE

After launching the EC2 instance, I learned how to SSH into it using Windows PowerShell CLI. To do this, I used the private key (.pem) that was generated during the instance setup, and connected by running the SSH command with the public IP address of my EC2 instance.


```
PS C:\Users\AARONS> cd .\Downloads\
PS C:\Users\AARONS\Downloads> ssh -i learning.pem ubuntu@18.236.178.83
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1016-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Sat Oct 12 02:48:17 UTC 2024

  System load:  0.0               Processes:             102
  Usage of /:   23.0% of 6.71GB   Users logged in:       0
  Memory usage: 20%               IPv4 address for enX0: 172.31.20.233
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-20-233:~$ 
```

In addition, I learned to SSH into my Ubuntu server using Putty. I retrieved my server’s IP address by running the command "ip a". However, I encountered issues with SSH being inactive on my server.

To resolve the issue, I first ran a series of commands to update the package list, install the OpenSSH server, start and enable SSH, and check its status. I run the following codes in the order that they are listed:

sudo apt update

sudo apt install openssh-server

sudo systemctl start ssh

sudo systemctl enable ssh

sudo systemctl status ssh

Despite these steps, SSH was still inactive. I ran another command to check the installation and re-installed OpenSSH, after which I restarted the SSH service.

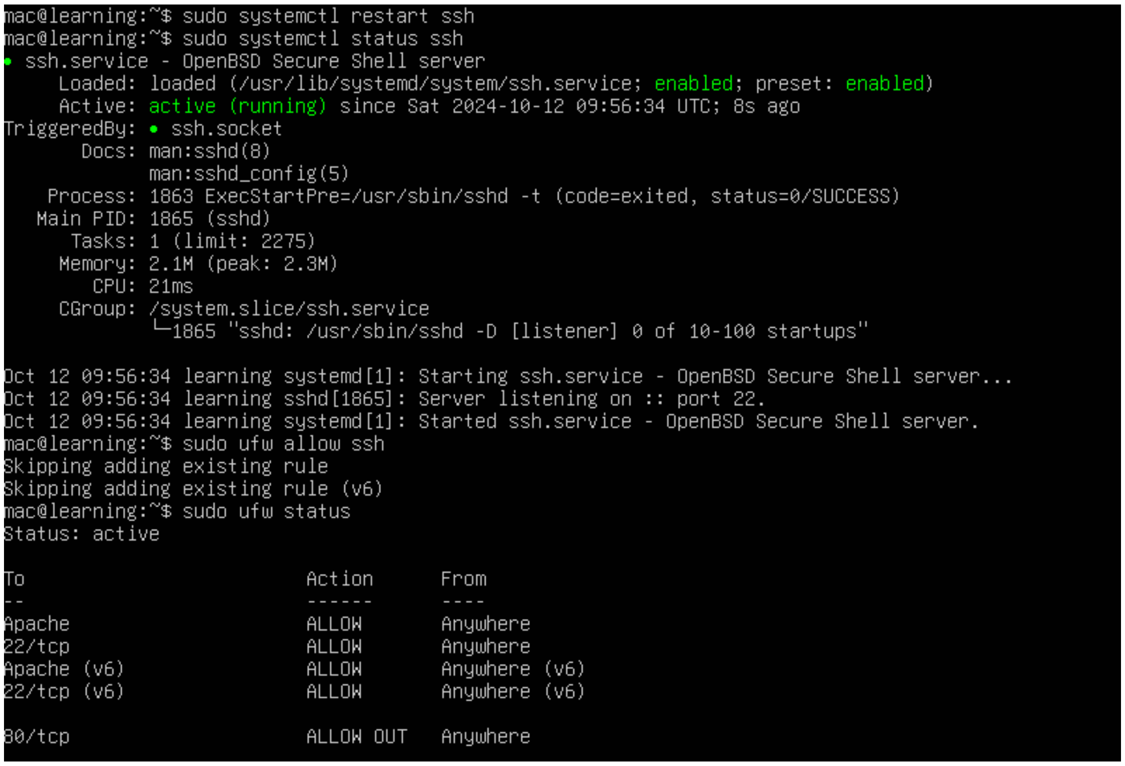
sudo apt list --installed | grep openssh-server

sudo apt install openssh-server

Then I had to restart SSH using the following code:

sudo systemctl restart ssh

Then I checked SSH Stats again using 'sudo systemctl status ssh' and realized it had become active.



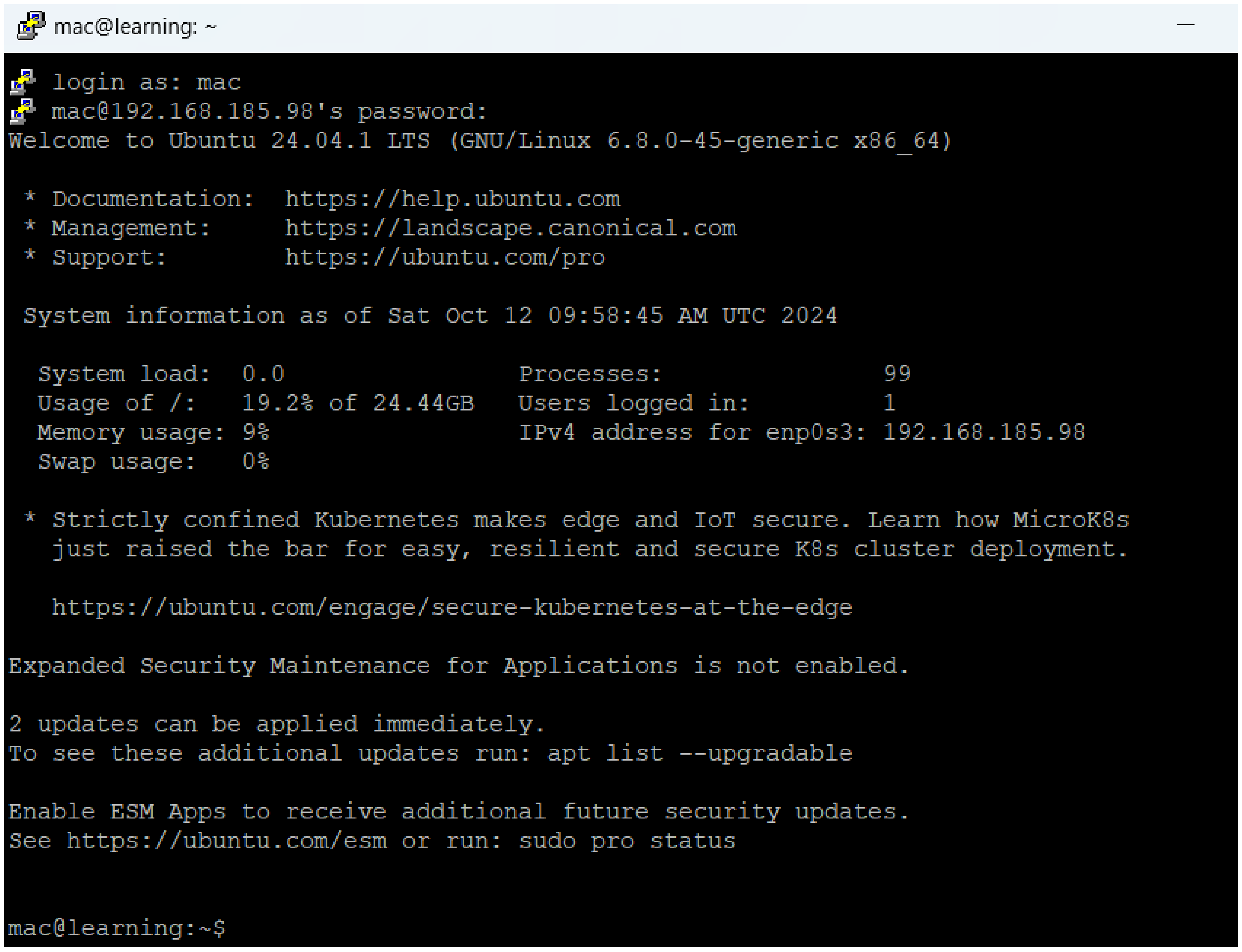
```
mac@learning:~$ sudo systemctl restart ssh
mac@learning:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/usr/lib/systemd/system/ssh.service; enabled; preset: enabled)
     Active: active (running) since Sat 2024-10-12 09:56:34 UTC; 8s ago
TriggeredBy: ● ssh.socket
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 1863 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 1865 (sshd)
      Tasks: 1 (limit: 2275)
     Memory: 2.1M (peak: 2.3M)
        CPU: 21ms
     CGroup: /system.slice/ssh.service
             └─1865 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Oct 12 09:56:34 learning systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...
Oct 12 09:56:34 learning sshd[1865]: Server listening on :: port 22.
Oct 12 09:56:34 learning systemd[1]: Started ssh.service - OpenBSD Secure Shell server.
mac@learning:~$ sudo ufw allow ssh
Skipping adding existing rule
Skipping adding existing rule (v6)
mac@learning:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
Apache                     ALLOW       Anywhere
22/tcp                     ALLOW       Anywhere
Apache (v6)                ALLOW       Anywhere (v6)
22/tcp (v6)                ALLOW       Anywhere (v6)

80/tcp                     ALLOW OUT   Anywhere
```

After this, I finally accessed Putty successfully using SSH


mac@learning: ~

```
login as: mac
mac@192.168.185.98's password:
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-45-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Sat Oct 12 09:58:45 AM UTC 2024

  System load:  0.0                Processes:               99
  Usage of /:   19.2% of 24.44GB   Users logged in:         1
  Memory usage: 9%                 IPv4 address for enp0s3: 192.168.185.98
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

2 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


mac@learning:~$
```

STEP 3: INSTALLING APACHE AND UPDATING FIREWALL

I used Ubuntu package manager 'apt' to install Apache

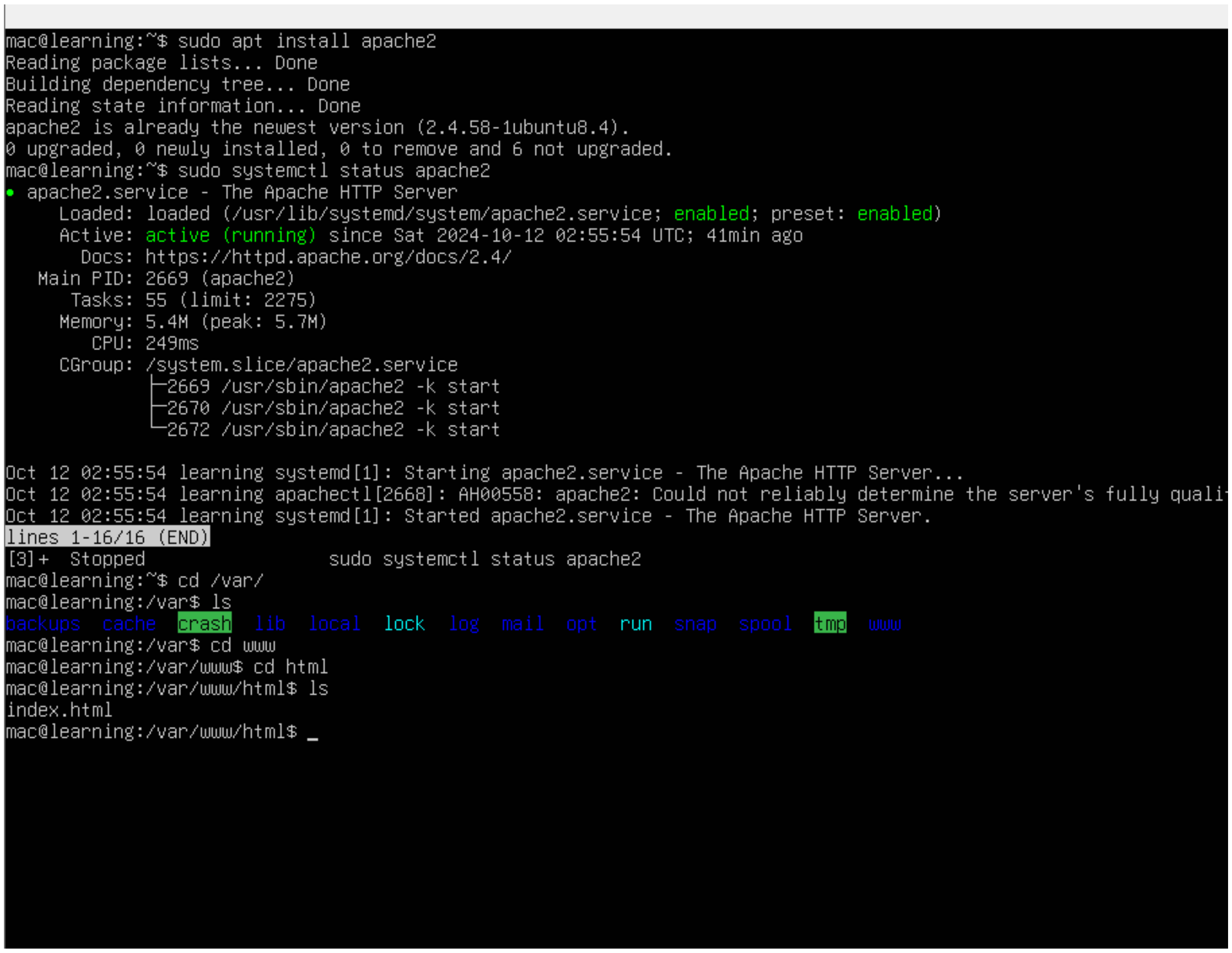
I run the following codes:

$ sudo apt update

$sudo apt install apache2

$sudo systemctl status apache2.

This ensured that Apache HTTP server was active (running)

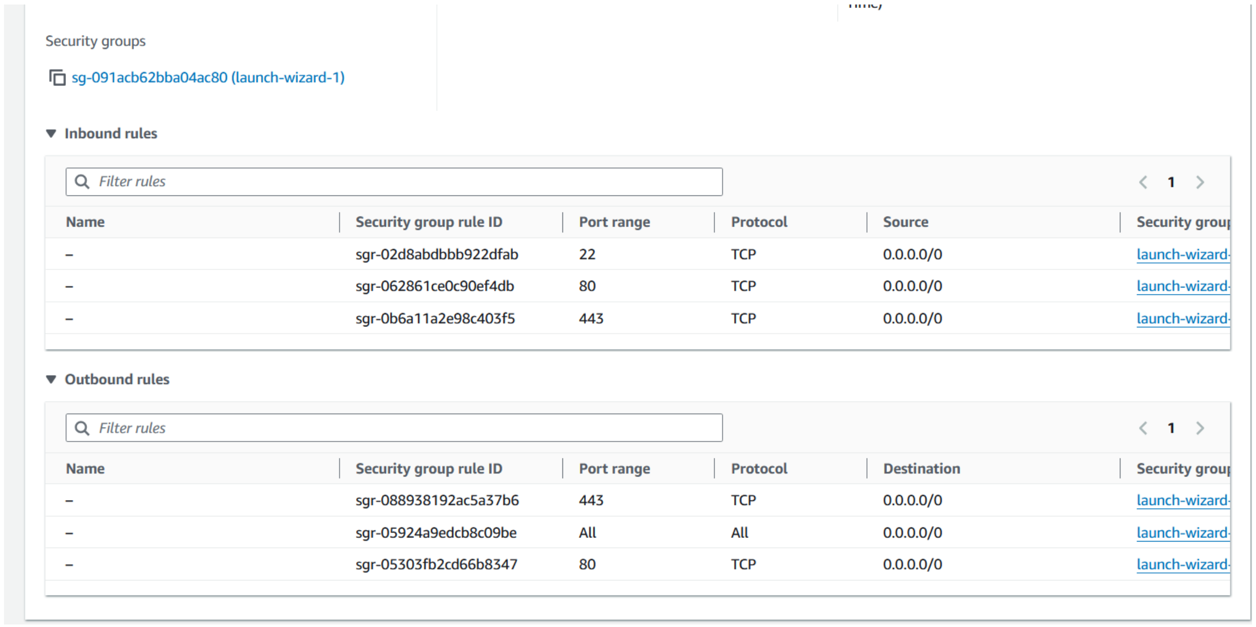


```
mac@learning:~$ sudo apt install apache2
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
apache2 is already the newest version (2.4.58-1ubuntu8.4).
0 upgraded, 0 newly installed, 0 to remove and 6 not upgraded.
mac@learning:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
     Active: active (running) since Sat 2024-10-12 02:55:54 UTC; 41min ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 2669 (apache2)
      Tasks: 55 (limit: 2275)
     Memory: 5.4M (peak: 5.7M)
        CPU: 249ms
     CGroup: /system.slice/apache2.service
             ├─2669 /usr/sbin/apache2 -k start
             ├─2670 /usr/sbin/apache2 -k start
             └─2672 /usr/sbin/apache2 -k start

Oct 12 02:55:54 learning systemd[1]: Starting apache2.service - The Apache HTTP Server...
Oct 12 02:55:54 learning apachectl[2668]: AH00558: apache2: Could not reliably determine the server's fully quali
Oct 12 02:55:54 learning systemd[1]: Started apache2.service - The Apache HTTP Server.
lines 1-16/16 (END)
[3]+  Stopped                 sudo systemctl status apache2
mac@learning:~$ cd /var/
mac@learning:/var$ ls
backups  cache  crash  lib  local  lock  log  mail  opt  run  snap  spool  tmp  www
mac@learning:/var$ cd www
mac@learning:/var/www$ cd html
mac@learning:/var/www/html$ ls
index.html
mac@learning:/var/www/html$ _
```

After that I went to my EC2 instance and edited inbound rules to receive request from Port 80. Next, I accessed the Apache server via DNS by using the curl command and then updated the inbound rules on my EC2 instance to allow HTTP requests through Port 80.

Then I used the $ curl 'http://localhost:80' command to access the Apache Server via my IP-Address.



Security groups

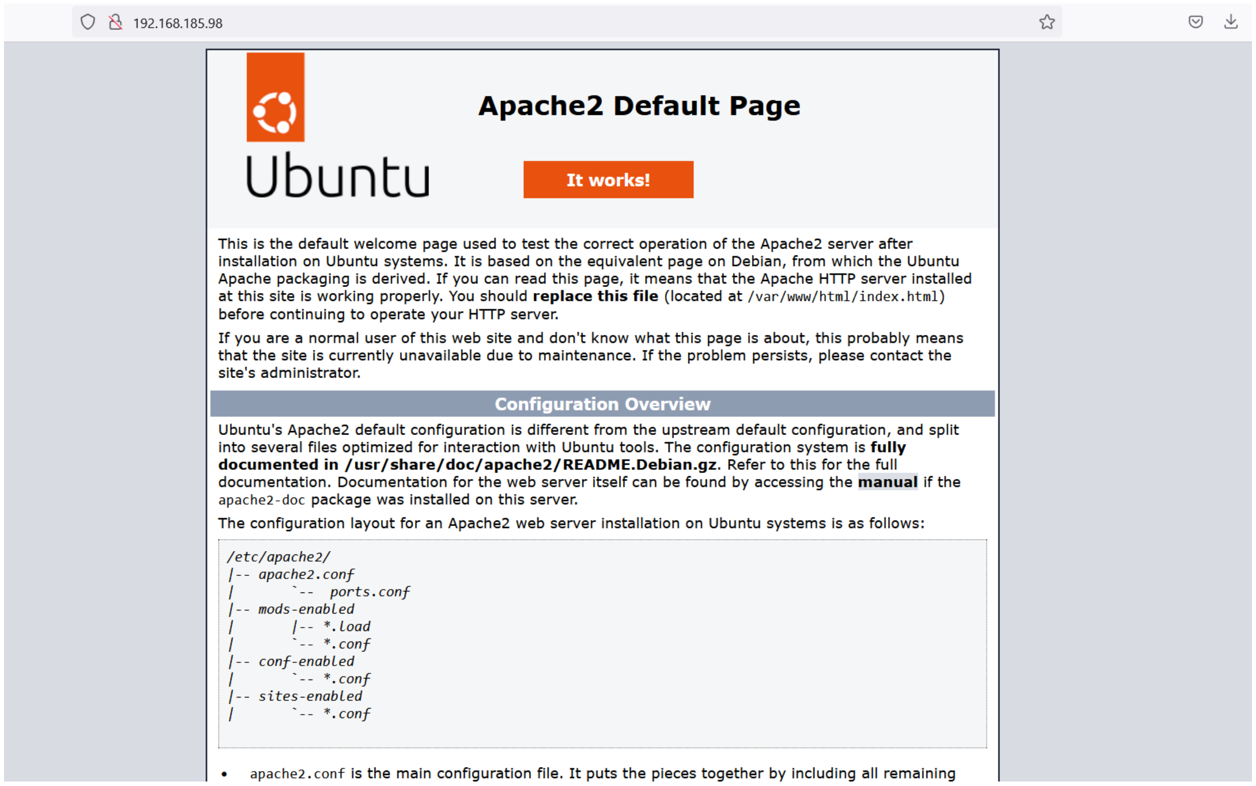
sg-091acb62bba04ac80 (launch-wizard-1)

Inbound rules

Name	Security group rule ID	Port range	Protocol	Source	Security group
–	sgr-02d8abdbbb922dfab	22	TCP	0.0.0.0/0	launch-wizard
–	sgr-062861ce0c90ef4db	80	TCP	0.0.0.0/0	launch-wizard
–	sgr-0b6a11a2e98c403f5	443	TCP	0.0.0.0/0	launch-wizard

Outbound rules

Name	Security group rule ID	Port range	Protocol	Destination	Security group
–	sgr-088938192ac5a37b6	443	TCP	0.0.0.0/0	launch-wizard
–	sgr-05924a9edcb8c09be	All	All	0.0.0.0/0	launch-wizard
–	sgr-05303fb2cd66b8347	80	TCP	0.0.0.0/0	launch-wizard



192.168.185.98

Apache2 Default Page

Ubuntu

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.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-enabled
|       `-- *.conf
```

- `apache2.conf` is the main configuration file. It puts the pieces together by including all remaining

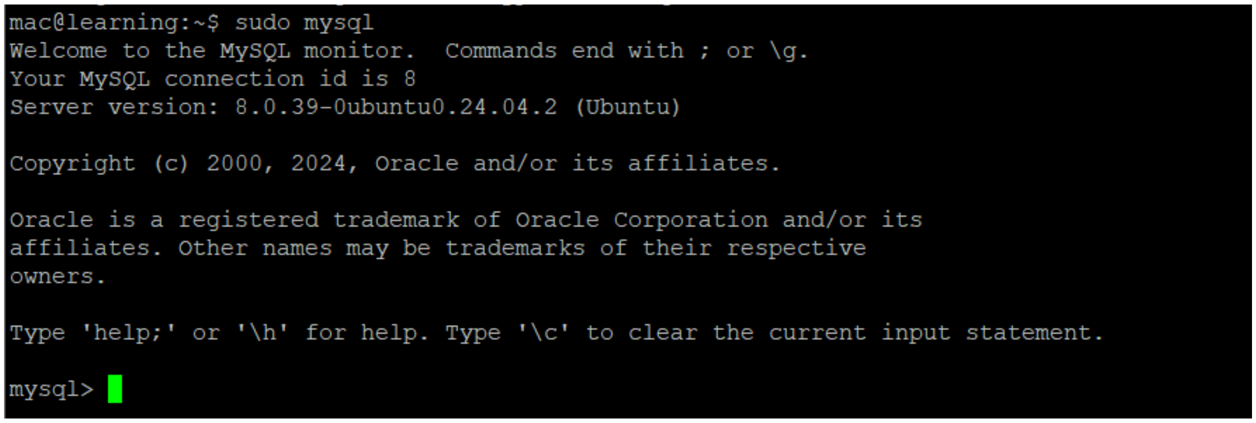
STEP 4: INSTALLING MYSQL-SERVER

I proceeded to install mysql-server using the code:

sudo ap install mysql-server

then i confirmed if it had been properly installed using

sudo mysql



```
mac@learning:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.39-0ubuntu0.24.04.2 (Ubuntu)

Copyright (c) 2000, 2024, 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> 
```

Then I set root password for MySQL using

ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'PassWord.1';

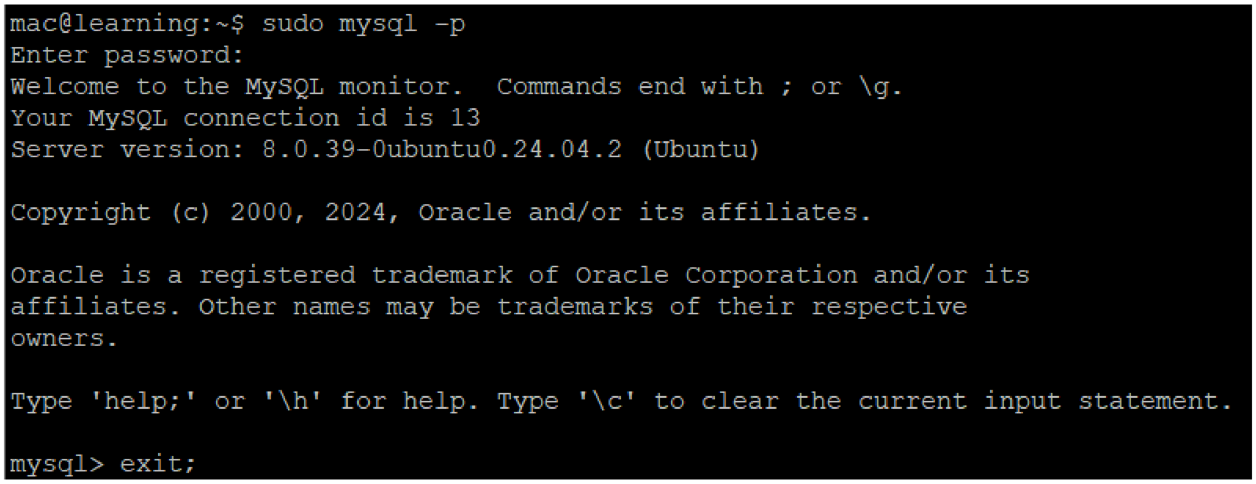
Next, I ran an interactive script that allowed me to set a root password and improve security settings. Afterward, I tested MySQL by logging in with the newly set password and confirmed that everything was working properly.

sudo mysql_secure_installation

After setting up the password for MySQL I tested it using

sudo MySQL -p

I was able to log in successfully.



```
mac@learning:~$ sudo mysql -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 13
Server version: 8.0.39-0ubuntu0.24.04.2 (Ubuntu)

Copyright (c) 2000, 2024, 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;
```

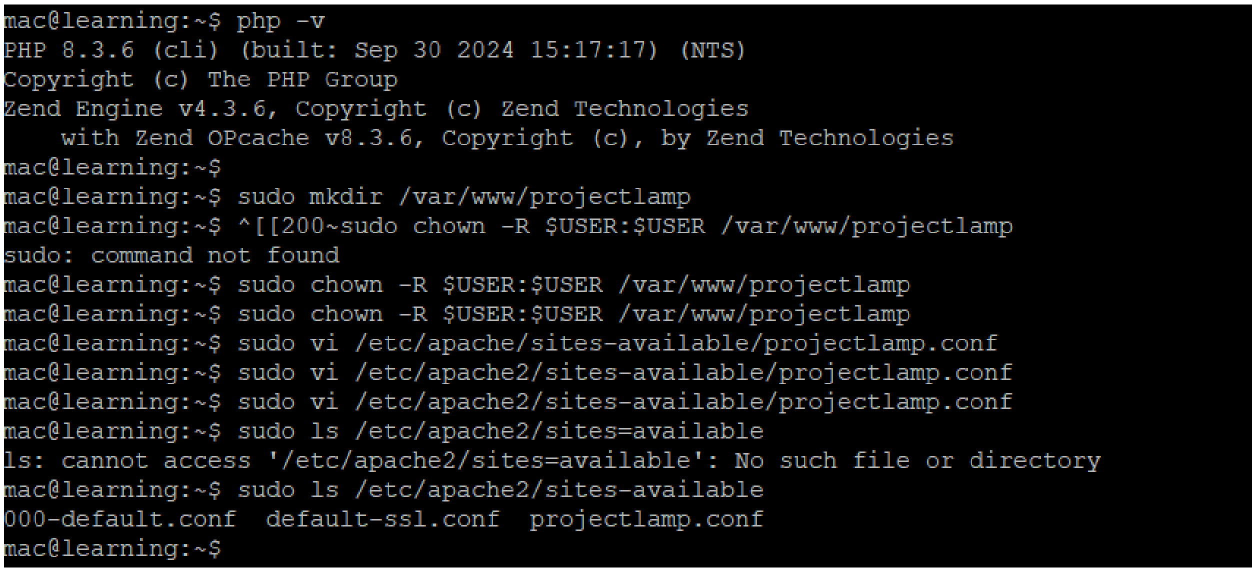
STEP 5: INSTALLING PHP

I understand that PHP is needed to process dynamic web content, while php-mysql allows PHP to communicate with MySQL-based databases, and libapache2-mod-php enables Apache to handle PHP files.

So, I run the command below to install PHP along with the necessary Apache and MySQL modules:

sudo apt install php libapache2-mod-php php-mysql

After installation, I run : *php -v* to confirm the version.



```
mac@learning:~$ php -v
PHP 8.3.6 (cli) (built: Sep 30 2024 15:17:17) (NTS)
Copyright (c) The PHP Group
Zend Engine v4.3.6, Copyright (c) Zend Technologies
    with Zend OPcache v8.3.6, Copyright (c), by Zend Technologies
mac@learning:~$
mac@learning:~$ sudo mkdir /var/www/projectlamp
mac@learning:~$ ^[[200~sudo chown -R $USER:$USER /var/www/projectlamp
sudo: command not found
mac@learning:~$ sudo chown -R $USER:$USER /var/www/projectlamp
mac@learning:~$ sudo chown -R $USER:$USER /var/www/projectlamp
mac@learning:~$ sudo vi /etc/apache/sites-available/projectlamp.conf
mac@learning:~$ sudo vi /etc/apache2/sites-available/projectlamp.conf
mac@learning:~$ sudo vi /etc/apache2/sites-available/projectlamp.conf
mac@learning:~$ sudo ls /etc/apache2/sites=available
ls: cannot access '/etc/apache2/sites=available': No such file or directory
mac@learning:~$ sudo ls /etc/apache2/sites-available
000-default.conf  default-ssl.conf  projectlamp.conf
mac@learning:~$
```

AT THIS POINT I UNDERSTAND THAT LAMP IS COMPLETELY INSTALLED AND FULLY OPERATIONAL ON MY UBUNTU SERVER.

STEP 6: CREATING A VIRTUAL HOST FOR THE WEBSITE

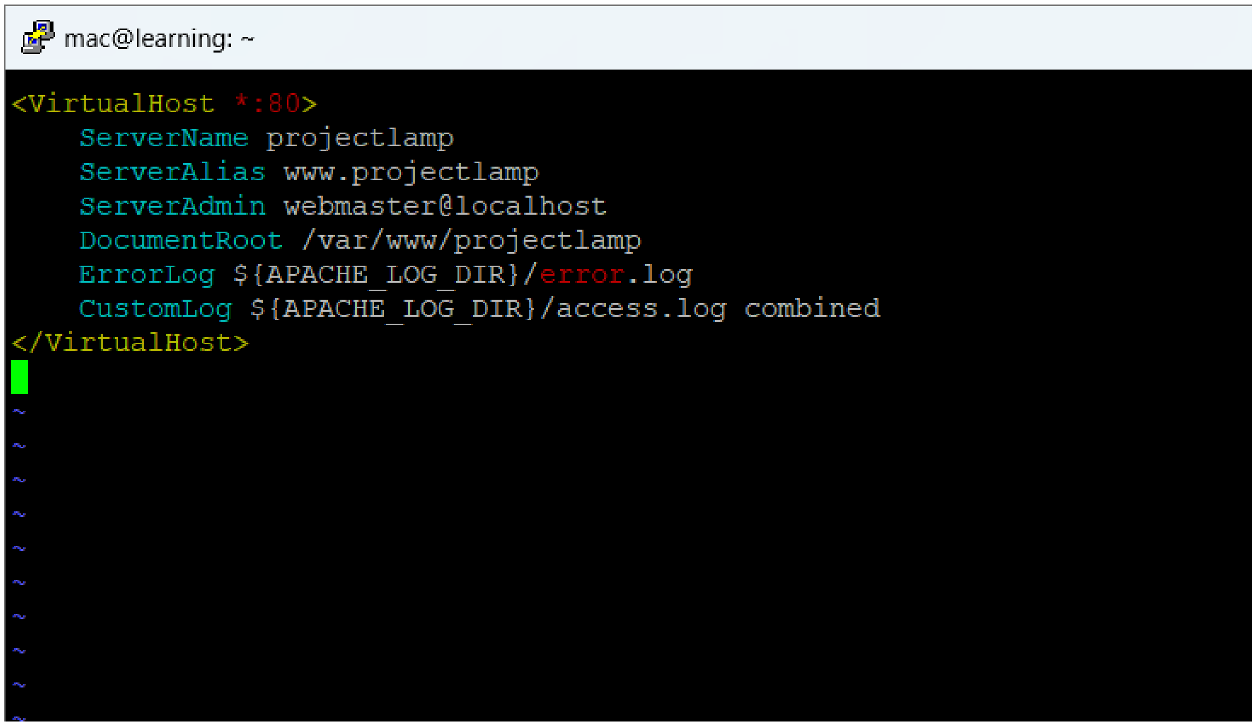
I first created a directory for my project "project lamp" using the command:

mkdir /var/www/projectlamp

Then I assigned ownership to the directory using the $USER environment variable. The following command was used to do that:

$ sudo chown -R $USER:$USER /var/www/projectlamp

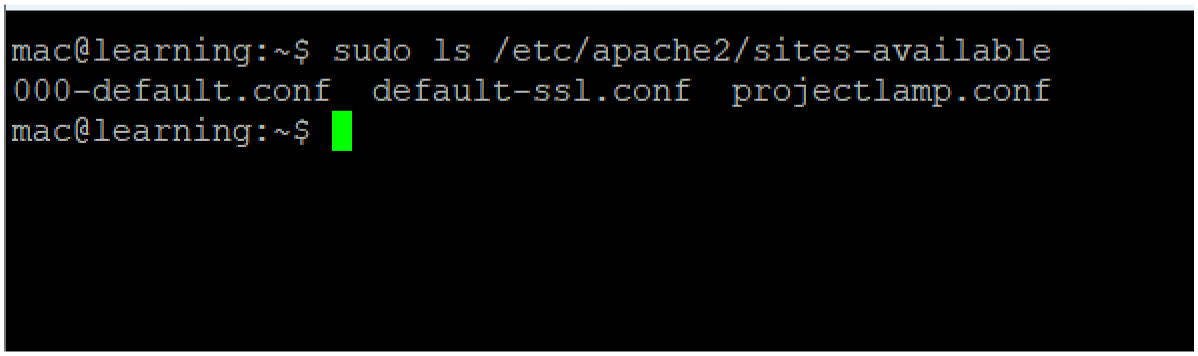
Then I opened a new configuration file in Apache's sites-available directory (sudo vi /etc/apache2/sites-available/projectlamp.conf) using vi or vim.



```
<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>

~
~
~
~
~
~
~
~
~
```

Then I used the list command to show new files in the sites-available directory



```
mac@learning:~$ sudo ls /etc/apache2/sites-available
000-default.conf  default-ssl.conf  projectlamp.conf
mac@learning:~$
```

I proceeded to enable new virtualhost by using

sudo a2ensite projectlamp

and also disabled default website that comes with Apache by using

sudo a2dissite 000-default

I checked to see if configuration did not contain errors by using

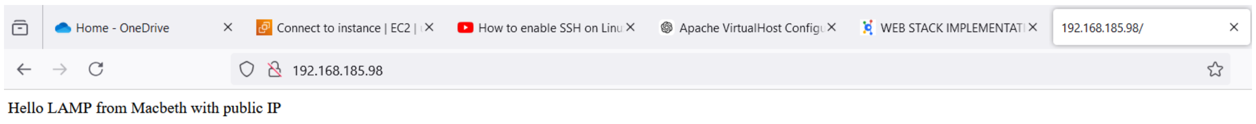
sudo apache2ctl configtest

I then created an html file by pastig in this command:

(NB: I modified the command a bit by inserting my name as the hostname.)

sudo echo 'Hello LAMP from Macbeth' $(TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` && curl -H "X-aws-ec2-metadata-token: $TOKEN" -s http://169.254.169.254/latest/meta-data/public-hostname) 'with public IP' $(TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` && curl -H "X-aws-ec2-metadata-token: $TOKEN" -s http://169.254.169.254/latest/meta-data/public-ipv4) > /var/www/projectlamp/index.html

Afterward, I listed the files in the sites-available directory to confirm the new virtual host configuration was in place. Next, I enabled the virtual host using the appropriate command and disabled the default website that comes with Apache. I also checked the configuration for any errors and ensured that everything was set up correctly. Finally, I created a custom HTML file to display a test message on my webpage. This message included my hostname and the public IP address of my server. After reloading my webpage, I verified that the setup was successful and the message was displayed correctly.


Home - OneDrive
Connect to instance | EC2 |
How to enable SSH on Linu
Apache VirtualHost Config
WEB STACK IMPLEMENTAT
192.168.185.98/
192.168.185.98

Hello LAMP from Macbeth with public IP

STEP 7: ENABLING PHP ON THE WEBSITE

The first step in the process was to modify the Apache configuration file to prioritize the loading of PHP files when a directory is accessed. Apache, by default, looks for specific file types (e.g., index.html, index.php) to load when a directory is requested in a browser. To modify this behavior, I needed to edit the directory index settings. I began by opening the dir.conf file, which is located in /etc/apache2/mods-enabled/. This file controls how Apache handles directory indexing. Using the vim editor, I ran the following command in the terminal:

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

This command opens the configuration file for editing with root privileges, which are necessary to make changes to system-level files. Once inside the file, I located the <IfModule mod_dir.c> section, which lists the files Apache looks for by default. Initially, the configuration listed multiple file types in a specific order, with index.html as the first file. My goal was to prioritize index.php as the first file in this list. The original configuration looked like this:

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

I edited the configuration to make index.php the first file Apache would look for:

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

This change ensures that whenever a directory is accessed, Apache will first look for a file named index.php before checking for other file types like index.html or index.cgi. After making this change, I saved the file and closed the vim editor by pressing Esc and typing *:wq*.

Once the configuration was edited, the next step was to apply the changes by reloading the Apache service. Instead of fully restarting Apache, which could temporarily interrupt the web server, I reloaded the service using the following command:

sudo systemctl reload apache2

Reloading the service ensures that the new configuration takes effect without affecting the server's availability. The next part of the task involved creating a PHP test script to verify that Apache could properly handle and process PHP files. To do this, I created a new file named index.php in the custom web root folder where the website's files are stored. I executed the following command to open a blank file using vim:

$ vim /var/www/projectlamp/index.php

This path points to the directory where the website's files are located. In this case, /var/www/projectlamp/ is the root folder for the web project. Once the file was opened, I added a simple PHP script to confirm the server's ability to handle PHP. The script I used calls the phpinfo() function, which provides detailed information about the PHP environment on the server. The code I added to index.php was:

<?php

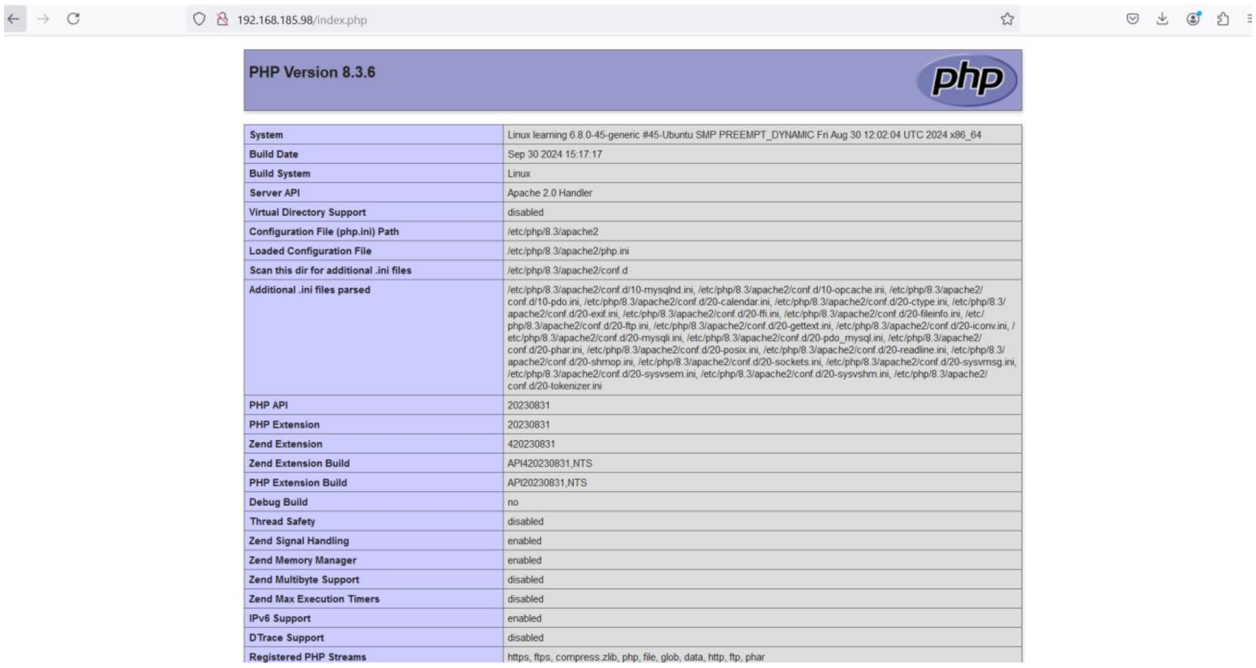
phpinfo();

?>

The phpinfo() function is commonly used to test PHP installations because it displays essential information, such as the PHP version, installed modules, and configuration settings. After adding the PHP code, I saved and closed the file by pressing Esc and typing :wq in the vim editor. The next step was to test whether Apache was correctly processing PHP files. To do this, I opened a web browser and navigated to the server's IP address or domain, followed by /index.php.

http://http://192.168.185.98/index.php

The page displayed PHP details, including the version, enabled modules, and server configuration. This successful display of the PHP info page indicated that both Apache and PHP were properly configured. The changes I made to prioritize index.php ensured that the server would load PHP files first when a directory is accessed, which is critical for PHP-based web applications.



PHP Version 8.3.6

System	Linux learning 6.8.0-45-generic #45-Ubuntu SMP PREEMPT_DYNAMIC Fri Aug 30 12:02:04 UTC 2024 x86_64
Build Date	Sep 30 2024 15:17:17
Build System	Linux
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/8.3/apache2
Loaded Configuration File	/etc/php/8.3/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/8.3/apache2/conf.d
Additional .ini files parsed	/etc/php/8.3/apache2/conf.d/10-mysqlnd.ini, /etc/php/8.3/apache2/conf.d/10-opcache.ini, /etc/php/8.3/apache2/conf.d/10-pdo.ini, /etc/php/8.3/apache2/conf.d/20-calendar.ini, /etc/php/8.3/apache2/conf.d/20-ctype.ini, /etc/php/8.3/apache2/conf.d/20-exif.ini, /etc/php/8.3/apache2/conf.d/20-ffi.ini, /etc/php/8.3/apache2/conf.d/20-fileinfo.ini, /etc/php/8.3/apache2/conf.d/20-ftp.ini, /etc/php/8.3/apache2/conf.d/20-gettext.ini, /etc/php/8.3/apache2/conf.d/20-iconv.ini, /etc/php/8.3/apache2/conf.d/20-mysqli.ini, /etc/php/8.3/apache2/conf.d/20-pdo_mysql.ini, /etc/php/8.3/apache2/conf.d/20-phar.ini, /etc/php/8.3/apache2/conf.d/20-posix.ini, /etc/php/8.3/apache2/conf.d/20-readline.ini, /etc/php/8.3/apache2/conf.d/20-shmop.ini, /etc/php/8.3/apache2/conf.d/20-sockets.ini, /etc/php/8.3/apache2/conf.d/20-sysvmsg.ini, /etc/php/8.3/apache2/conf.d/20-sysvsem.ini, /etc/php/8.3/apache2/conf.d/20-sysvshm.ini, /etc/php/8.3/apache2/conf.d/20-tokenizer.ini
PHP API	20230831
PHP Extension	20230831
Zend Extension	420230831
Zend Extension Build	API420230831,NTS
PHP Extension Build	API20230831,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
Zend Max Execution Timers	disabled
IPv6 Support	enabled
DTrace Support	disabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar

CHALLENGES AND SOLUTIONS

Throughout this process, I faced some challenges, particularly with SSH access and ensuring the proper configuration of the LAMP stack. One significant issue was that SSH was initially inactive, requiring me to reinstall and restart the SSH service to resolve the problem. Another challenge involved configuring Apache to prioritize PHP files, which I overcame by editing the correct configuration file.

To overcome these challenges, I reached out to colleagues for assistance, and my colleague Fisayo was particularly helpful. I am grateful for this opportunity.