

WEB STACK IMPLEMENTATION (LAMP STACK) IN AWS

A technology stack is a set of frameworks and tools used to develop a software product. 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)

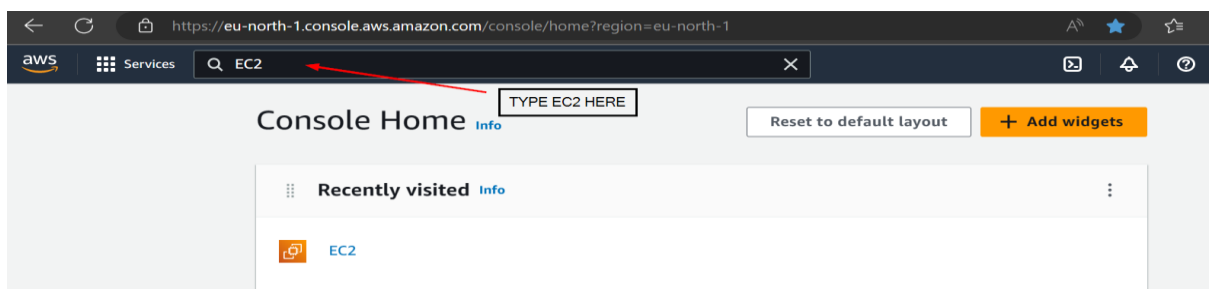
PREREQUISITES

- Aws account running an EC2 instance.
- Internet connection
- Fundamental Knowledge of downloading and installing
- Basics Linux skills

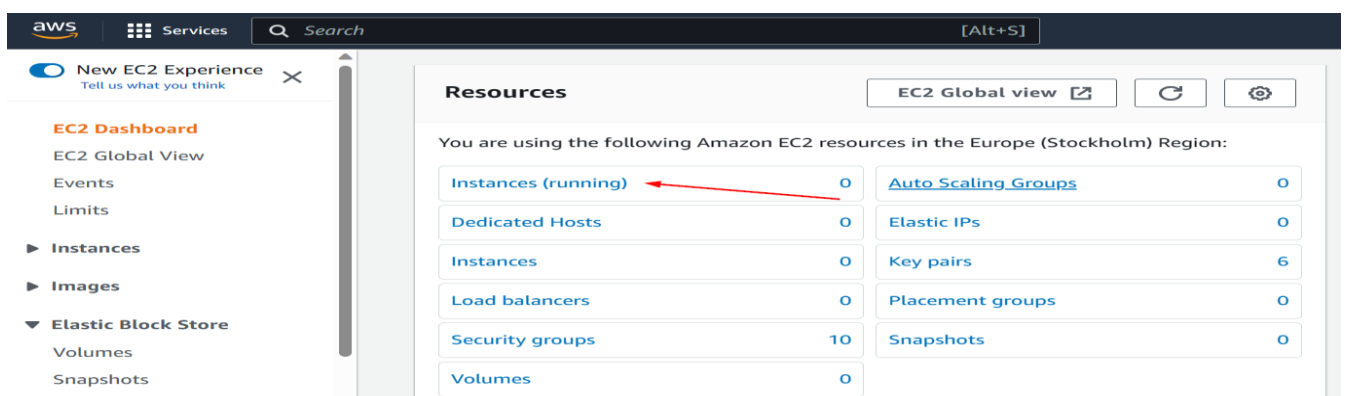
AWS is the biggest Cloud Service Provider, and it offers a free tier account that am going to leverage for this project. Right now, AWS can provide me with a free virtual server called EC2 (Elastic Compute Cloud) for my needs.

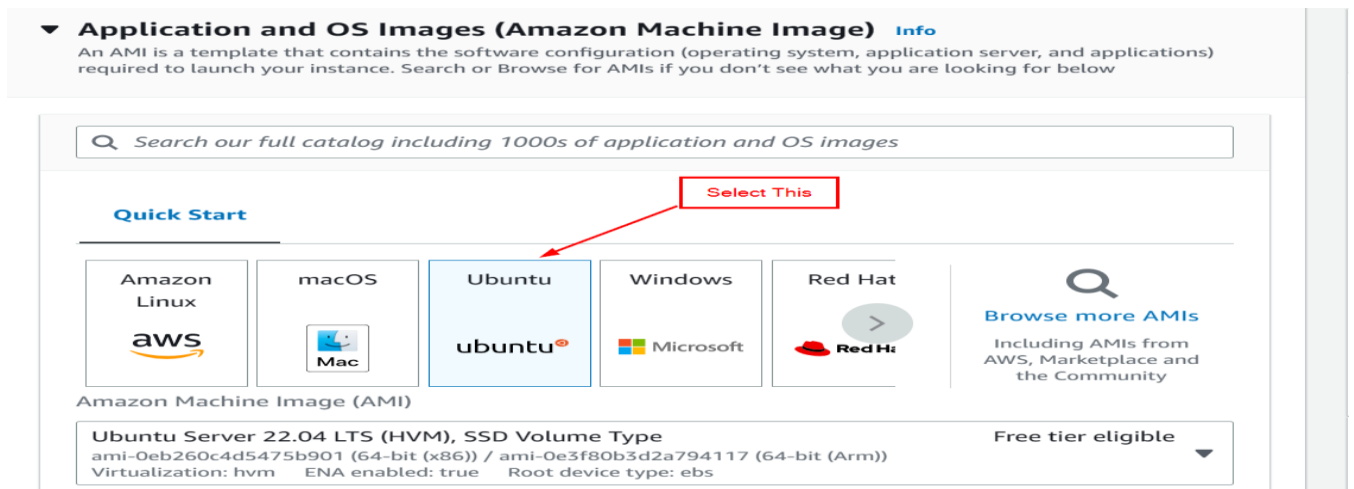
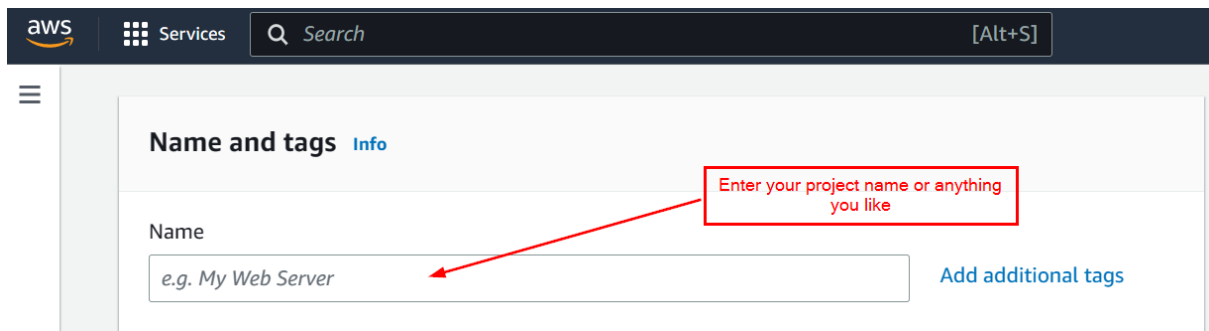
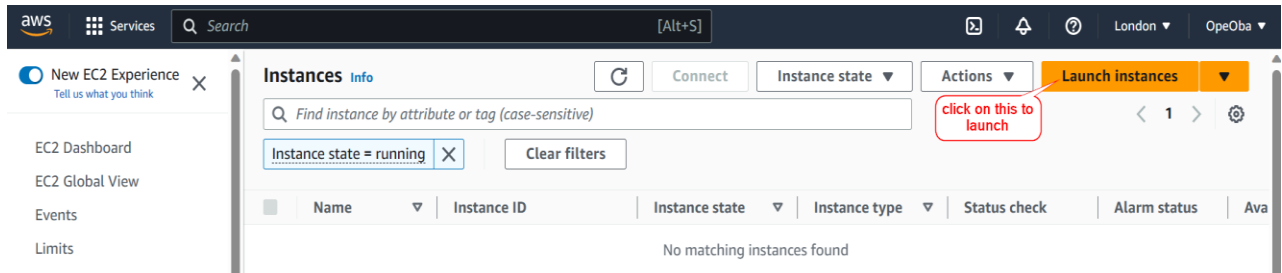
IMPLEMENTATION

- Open your PC browser and login to <https://aws.amazon.com/>
- A region is selected by default (change to your closest location if necessary), and from the search bar type EC2 and click.



- From the EC2 dashboard, click on the button “instances (running)” to launch instance.





An AMI window displays, type “Ubuntu” on the search bar and hit enter, or scroll down to select “Ubuntu Server 22.04 LTS (HVM), SSD Volume Type” based on your system architecture.

- The next step of configuring our EC2 is to select the instance type, preferably a t2 micro - Free tier.
- Create a key pair name, which automatically download into your local system.

aws Services Search [Alt+S]

▼ Instance type Info

Instance type

t2.micro Free tier eligible

Family: t2 1 vCPU 1 GiB Memory Current generation: true

On-Demand Windows pricing: 0.0178 USD per Hour

On-Demand RHEL pricing: 0.0732 USD per Hour

On-Demand SUSE pricing: 0.0132 USD per Hour

On-Demand Linux pricing: 0.0132 USD per Hour

☐ All generations

[Compare instance types](#)

▼ Key pair (login) Info

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Create a key pair name (it can be any name)

Key pair name - required

Select

[Create new key pair](#)

- Leave the Network settings and Configure storage as Default and Launch instance

aws Services Search [Alt+S]

Key pair name - required

Select [Create new key pair](#)

▼ Network settings Info Edit

Network Info

vpc-0415751608806d554

Subnet Info

No preference (Default subnet in any availability zone)

Auto-assign public IP Info

Enable

Firewall (security groups) Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group ☐ Select existing security group

We'll create a new security group called 'launch-wizard-22' with the following rules:

☒ Allow SSH traffic from

Launch...

▼ Summary

Number of instances Info

1

Software Image (AMI)

Canonical, Ubuntu, 22.04 LTS, ...read more

ami-0eb260c4d5475b901

Virtual server type (instance type)

t2.micro

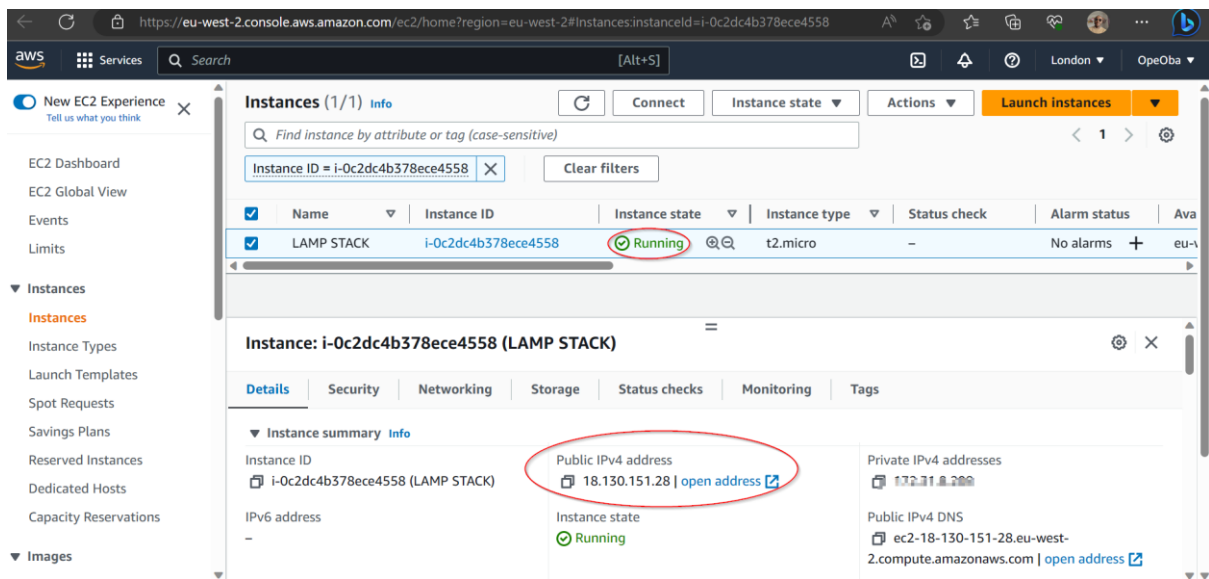
Firewall (security group)

New security group

Storage (volumes)

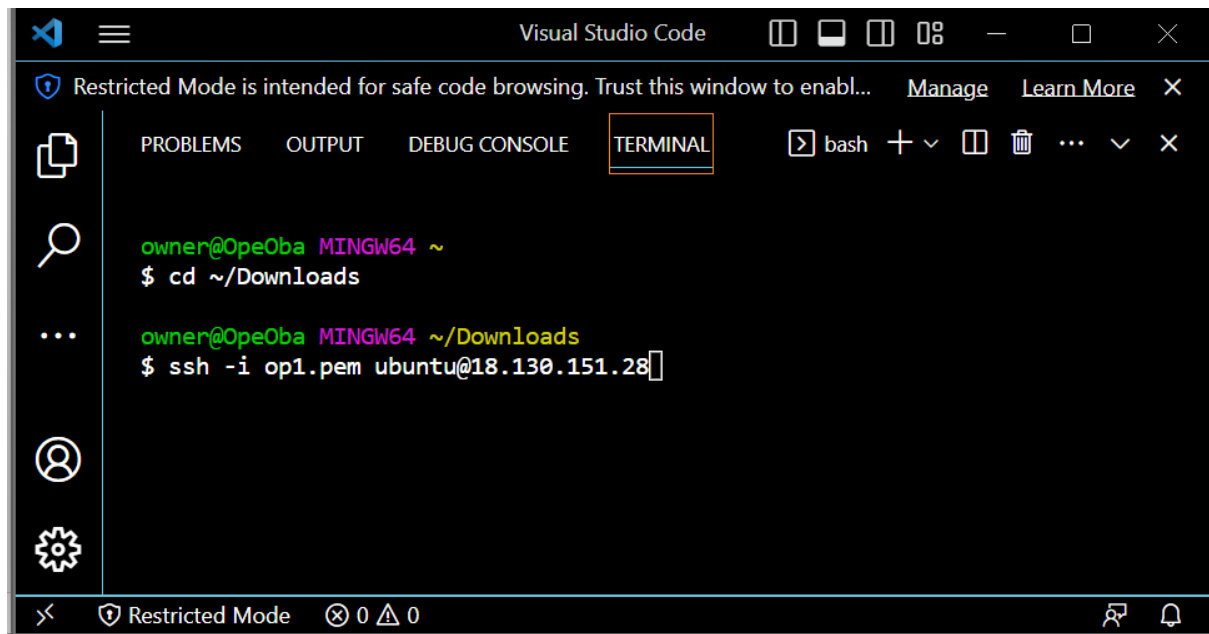
Cancel [Launch instance](#) [Review commands](#)

Done? If you get this, it shows your instance is up and running. Good Job, let's proceed now.



Copy your own Public IP as shown on the above screenshot, now it is time to use the console Yay!!! Open git bash or putty or termius, whichever console works for you, else download.

I am using git bash:



Type yes to connect.

```
The authenticity of host '18.130.151.28 (18.130.151.28)' can't be established.  
ED25519 key fingerprint is SHA256:rRLPmK/OvL7yDiSsuSBg7QCotUDF8JRnr1Voz8Kim  
qw.  
This key is not known by any other names.  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

You have now connected to the EC2 instance via SSH

Type **clear**, to have a neat console and proceed. Now let us follow the steps:

STEP 1 - INSTALLING APACHE AND UPDATING THE FIREWAL

[Apache HTTP Server](#) is the most widely used web server software. Developed and maintained by Apache Software Foundation, Apache is an open-source software available for free. It runs on 67% of all web servers in the world. It is fast, reliable, and secure. It can be highly customized to meet the needs of different environments by using extensions and modules.

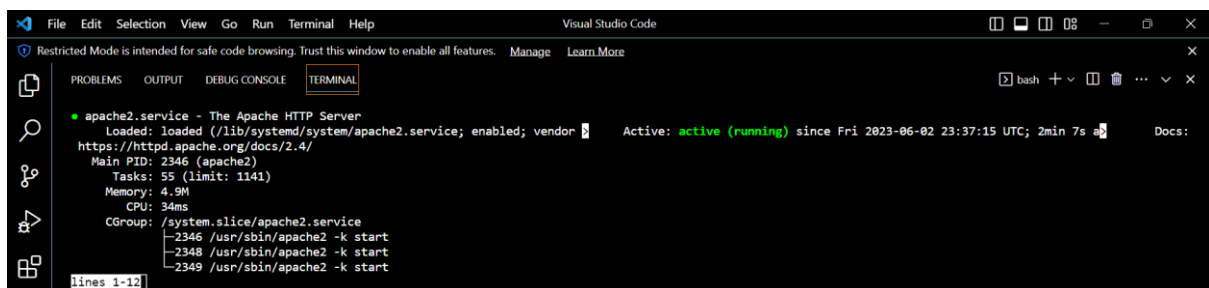
Install Apache using Ubuntu's package manager **'apt'**:

```
#update a list of packages in package manager
sudo apt update

#run apache2 package installation
sudo apt install apache2 -y
```

To verify that apache2 is running as a Service in our OS, use following command.

```
sudo systemctl status apache2
```



If you have this above active and (running) – you have just launched your first Web Server in the Clouds.

Before we can receive any traffic by our Web Server, we need to open TCP port 80 which is the default port that web browsers use to access web pages on the Internet

As we know, we have TCP port 22 open by default on our EC2 machine to access it via SSH, so we need to add a rule to EC2 configuration to open inbound connection through port 80:

Let us go back to our instance and add new rule which is port 80:

aws Services Search [Alt+S] London OpeOba

New EC2 Experience Tell us what you think

EC2 Dashboard
EC2 Global View
Events
Limits

Instances

Instances

Instance Types
Launch Templates
Spot Requests
Savings Plans

Instances (1/1) Info

Find instance by attribute or tag (case-sensitive)

Instance ID = i-0c2dc4b378ece4558 X Clear filters

<input checked="" type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Ava
<input checked="" type="checkbox"/>	LAMP STACK	i-0c2dc4b378ece4558	Running	t2.micro	-	No alarms	eu-n

Instance: i-0c2dc4b378ece4558 (LAMP STACK)

Details Security Networking Storage Status checks Monitoring Tags

Instance summary Info

Webstack-Implementation: x Launch an instance | EC2 | x Instances | EC2 Managem AWS Solutions Architect C x Step 1 — Installing Apache x

https://eu-west-2.console.aws.amazon.com/ec2/home?region=eu-west-2#Instances:instanceId=i-0c2dc4b378ece4558

aws Services Search [Alt+S] London OpeOba

New EC2 Experience Tell us what you think

EC2 Dashboard
EC2 Global View
Events
Limits

Instances

Instances

Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations

Images

Instances (1/1) Info

Find instance by attribute or tag (case-sensitive)

Instance ID = i-0c2dc4b378ece4558 X Clear filters

<input checked="" type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Ava
<input checked="" type="checkbox"/>	LAMP STACK	i-0c2dc4b378ece4558	Running	t2.micro	-	No alarms	eu-n

Instance: i-0c2dc4b378ece4558 (LAMP STACK)

Security groups

sg-04fa79dae5522564c (launch-wizard-22)

Inbound rules

Filter rules

Name	Security group rule ID	Port range	Protocol	Source
------	------------------------	------------	----------	--------

CloudShell Feedback Language

© 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

10°C Mostly cloudy

00:50 03/06/2023

Inbound rules Outbound rules Tags

You can now check network connectivity with Reachability Analyzer Run Reachability Analyzer X

Inbound rules (1/1)

Filter security group rules

Manage tags Edit inbound rules

<input checked="" type="checkbox"/>	Name	Security group rule...	IP version	Type	Protocol
<input checked="" type="checkbox"/>	-	sgr-03035e290b7724...	IPv4	SSH	TCP

EC2 > Security Groups > sg-04fa79dae5522564c - launch-wizard-22 > Edit inbound rules

Edit inbound rules [Info](#)

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

Inbound rules [Info](#)

Security group rule ID	Type	Protocol	Port range	Source	Description - optional	
sgr-03035e290b77244ba	SSH	TCP	22	Custom 0.0.0.0/0		Delete

Add rule

Cancel Preview changes **Save rules**

aws Services Search [Alt+S] London Op

Edit inbound rules [Info](#)

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

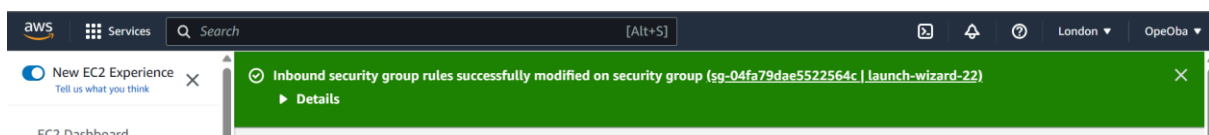
Inbound rules [Info](#)

Security group rule ID	Type	Protocol	Port range	Source	Description - optional	
sgr-03035e290b77244ba	SSH	TCP	22	Custom 0.0.0.0/0		Delete
-	Custom TCP	TCP	80	Anywh... 0.0.0.0/0		Delete

Add rule

Cancel Preview changes **Save rules**

If you have this below, then it is successfully added.



Our server is running and we can access it locally and from the Internet (Source 0.0.0.0/0 means 'from any IP address').

To come out of the page you are on your terminal, press **ctrl + c**

First, 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
```

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 following url

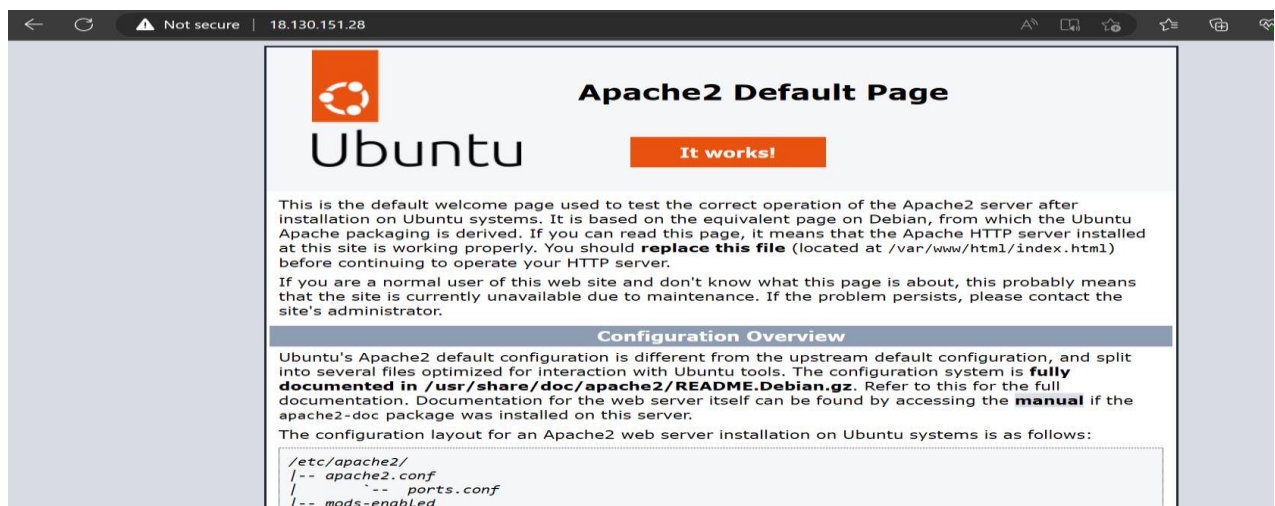
```
http://<Public-IP-Address>:80
```

Another way to retrieve your Public IP address, other than to check it in AWS Web console, is to use following command:

```
curl -s http://169.254.169.254/latest/meta-data/public-ipv4
```

The URL in browser shall also work if you do not specify port number since all web browsers use port 80 by default.

If you see following page, then your web server is now correctly installed and accessible through your firewall.



STEP 2 — INSTALLING MYSQL

Now that we have our web server up and running, we need to install a [Database Management System \(DBMS\)](#) to be able to store and manage data for our site in a [relational database](#). [MySQL](#) is a popular relational database management system used within PHP environments, so we will use it in this project.

Again, use 'apt' to acquire and install this software:

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

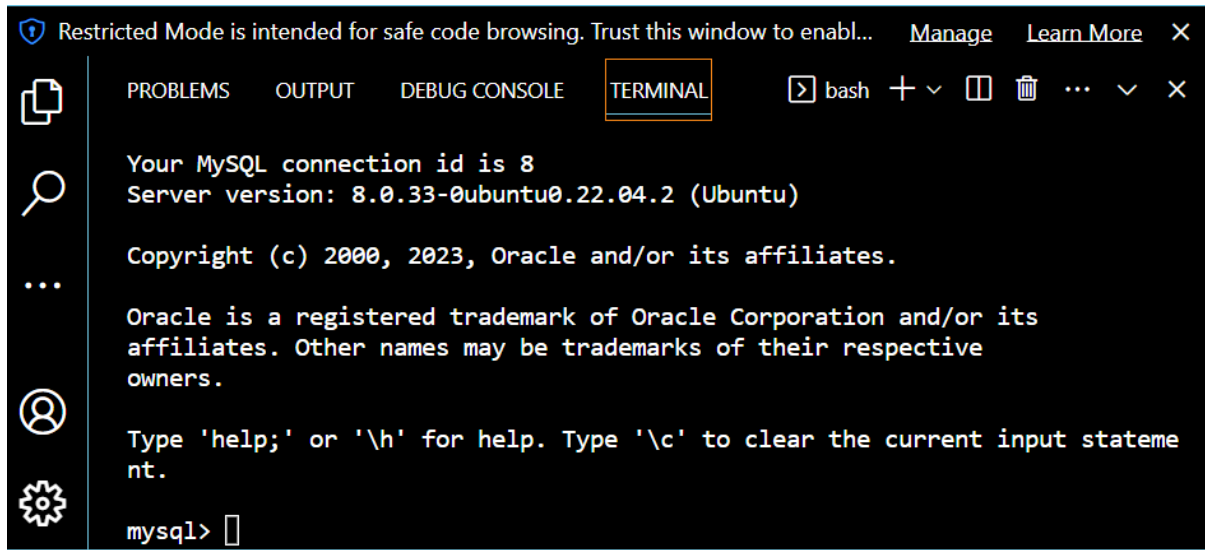
When prompted, confirm installation by typing **Y**, and then **ENTER**.


```
After this operation, 243 MB of additional disk space will be used.  
Do you want to continue? [Y/n]
```

When the installation finished, log in to the MySQL console by typing:

```
$ sudo mysql
```

This will connect to the MySQL server as the administrative database user **root**, which is inferred using sudo when running this command. You should see output like this:

A screenshot of a terminal window. The title bar says "Restricted Mode is intended for safe code browsing. Trust this window to enable...". The terminal shows the output of the 'mysql' command. It displays: "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.", "Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.", and finally the "mysql>" prompt.

```
Restricted Mode is intended for safe code browsing. Trust this window to enable... Manage Learn More X  
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL bash + - [ ] [ ] ... v X  
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.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input state  
ment.  
  
mysql>
```

It's recommended that you run a security script that comes pre-installed with MySQL. This script will remove some insecure default settings and lock down access to your database system. Before running the script you will set a password for the **root** user, using `mysql_native_password` as default authentication method. We're defining this user's password as **Lamp1**.

```
ALTER USER 'root'@'localhost' IDENTIFIED WITH  
mysql_native_password BY 'Lamp1';
```

Exit the MySQL shell with:

```
mysql> exit
```

Start the interactive script by running:

```
$ sudo mysql_secure_installation
```

Note: Enabling this feature is something of a judgment call. If enabled, passwords which don't match the specified criteria will be rejected by MySQL with an error. It is safe to leave validation disabled, but you should always use strong, unique passwords for database credentials.

Answer **Y** for yes, or anything else to continue without enabling.

```
VALIDATE PASSWORD PLUGIN 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 plugin?
```

```
Press y|Y for Yes, any other key for No:
```

If you answer “yes”, you’ll be asked to select a level of password validation. Keep in mind that if you enter **2** for the strongest level, you will receive errors when attempting to set any password which does not contain numbers, upper and lowercase letters, and special characters, or which is based on common dictionary words e.g **Lamp1**.

```
There are three levels of password validation policy:
```

```
LOW      Length >= 8
MEDIUM  Length >= 8, numeric, mixed case, and special characters
STRONG Length >= 8, numeric, mixed case, special characters and
dictionary file
```

```
Please enter 0 = LOW, 1 = MEDIUM and 2 = STRONG: 1
```

Regardless of whether you chose to set up the **VALIDATE PASSWORD PLUGIN**, your server will next ask you to select and confirm a password for the MySQL **root** user. This is not to be confused with the **system root**. The **database root** user is an administrative user with full privileges over the database system. Even though the default authentication method for the MySQL root user dispenses the use of a password, **even when one is set**, you should define a strong password here as an additional safety measure. We’ll talk about this in a moment.

If you enabled password validation, you’ll be shown the password strength for the root password you just entered and your server will ask if you want to continue with that password. If you are happy with your current password, enter **Y** for “yes” at the prompt:

```
Estimated strength of the password: 100
Do you wish to continue with the password provided?(Press y|Y for
Yes, any other key for No) : y
```

For the rest of the questions, press **Y** and hit the **ENTER** key at each prompt. This will prompt you to change the root password, remove some anonymous users and the test database, disable remote root logins, and load these new rules so that MySQL immediately respects the changes you have made.

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

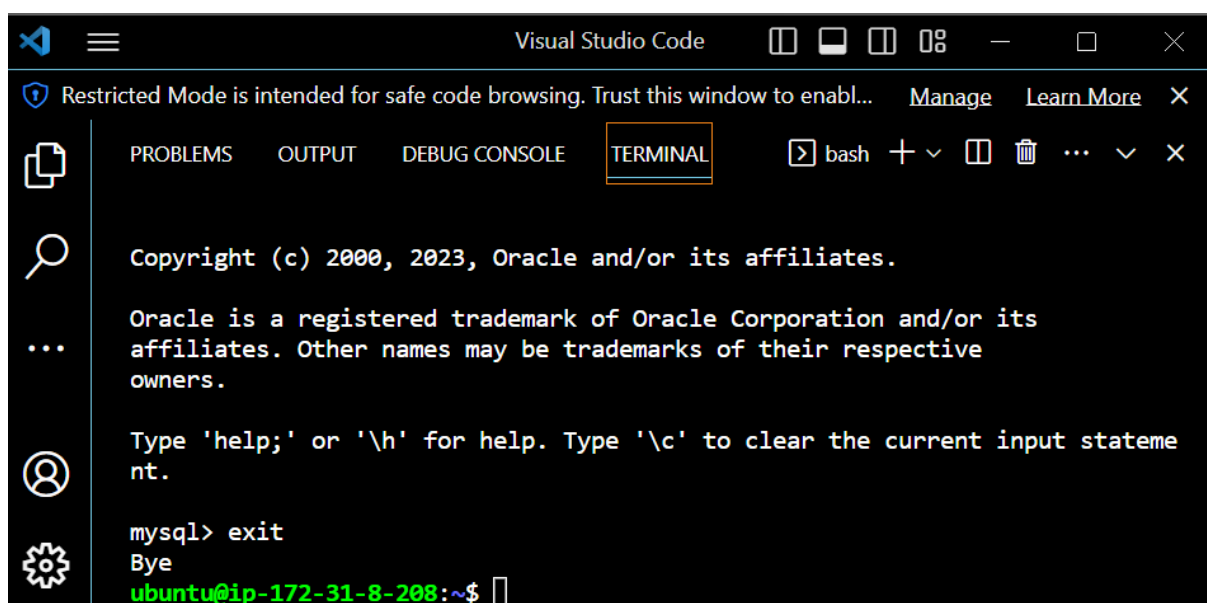
```
$ sudo mysql -p
```

Notice the **-p** flag in the command, which will prompt you for the password used after changing the **root** user password.

To exit the MySQL console, type:

```
mysql> exit
```

Notice that you need to provide a password to connect as the **root** user.

A screenshot of a Visual Studio Code window with a terminal pane open. The terminal shows the output of the 'mysql' command. It displays copyright information for Oracle, instructions on how to use help and clear the input, and then shows the user typing 'mysql> exit' and receiving a 'Bye' response. The prompt changes from 'mysql>' to 'ubuntu@ip-172-31-8-208:~\$'. The terminal title bar says 'Visual Studio Code' and the tab is labeled 'TERMINAL'.

```
Visual Studio Code
Restricted Mode is intended for safe code browsing. Trust this window to enable... Manage Learn More X
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL bash + v [ ] [ ] ... v X
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-172-31-8-208:~$
```

Our MySQL server is now installed and secured. Next, we will install PHP, the final component in the LAMP stack.

STEP 3 — INSTALLING PHP

we have Apache installed to serve our content and MySQL installed to store and manage our data. [PHP](#) is the component of our setup that will process code to display dynamic content to the end user. In addition to the **php** package, we will need **php-mysql**, a PHP module that allows PHP to communicate with MySQL-based databases. we will also need **libapache2-mod-php** to enable Apache to handle PHP files. Core PHP packages will automatically be installed as dependencies.

To install these 3 packages at once, run:

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

Once the installation finishes, we can run the following command to confirm our PHP version:

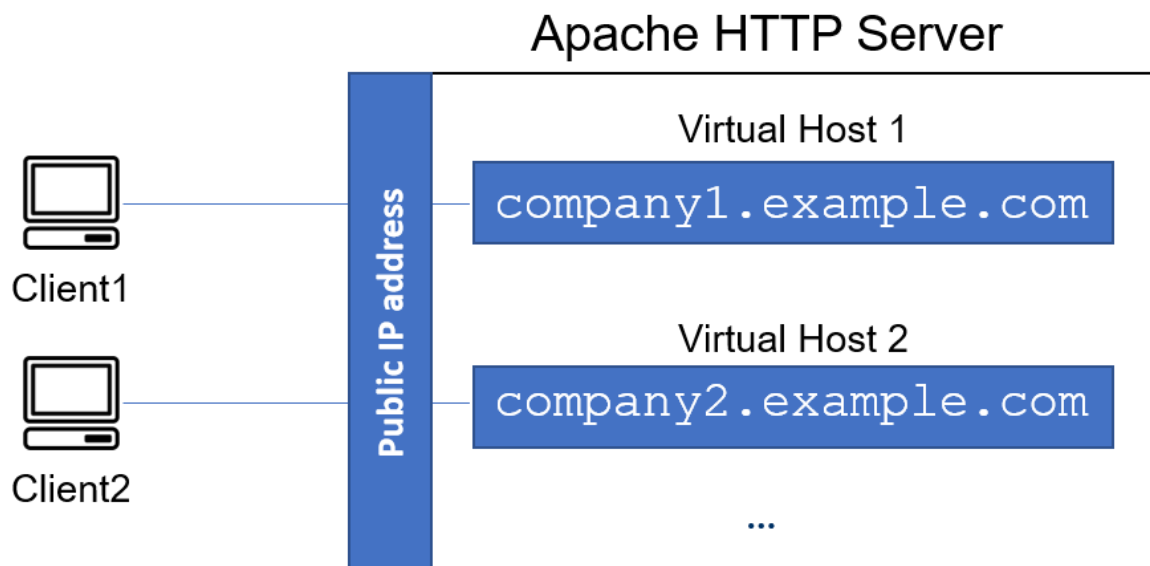
```
php -v
```

```
ubuntu@ip-172-31-8-208:~$ php -v
PHP 8.1.2-1ubuntu2.11 (cli) (built: Feb 22 2023 22:56:18) (NTS)
Copyright (c) The PHP Group
Zend Engine v4.1.2, Copyright (c) Zend Technologies
    with Zend OPcache v8.1.2-1ubuntu2.11, Copyright (c), by Zend Technologies
ubuntu@ip-172-31-8-208:~$
```

At this point, your LAMP stack is completely installed and fully operational.

- ☒ Linux (Ubuntu)
- ☒ Apache HTTP Server
- ☒ MySQL
- ☒ PHP

To test our setup with a PHP script, it's best to set up a proper [Apache Virtual Host](#) to hold our website's files and folders. Virtual host allows us to have multiple websites located on a single machine and users of the websites will not even notice it.



STEP 4 - CREATING A VIRTUAL HOST FOR OUR WEBSITE

USING APACHE

In this project, we will set up a domain called **projectlamp**, but you can replace this with any domain of your choice.

Apache on Ubuntu 22.04 has one server block enabled by default that is configured to serve documents from the **/var/www/html** directory.

We will leave this configuration as is and will add our own directory next to the default one.

Create the directory for **projectlamp** using **'mkdir'** command as follows:

```
sudo mkdir /var/www/projectlamp
```

Next, assign ownership of the directory with your current system user:

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

Then, create and open a new configuration file in Apache's **sites-available** directory using your preferred command-line editor. Here, we'll be using **vi** or **vim** (They are the same by the way):

```
sudo nano /etc/apache2/sites-available/projectlamp.conf
```

This will create a new blank file. Paste in the following bare-bones configuration by hitting on **i** on the keyboard to enter the insert mode, and type

the text:

```
File Edit View

<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>
```

Press **ctrl + x** to save and Enter

You can use the **ls** command to show the new file in the **sites-available** directory

```
sudo ls /etc/apache2/sites-available
```

you will see something like this;

```
ubuntu@ip-172-31-8-208:~$ sudo ls /etc/apache2/sites-available
000-default.conf default-ssl.conf projectlamp.conf
ubuntu@ip-172-31-8-208:~$
```

With this VirtualHost configuration, we are telling Apache to serve **projectlamp** using **/var/www/projectlamp** as its web root directory.

We can now use **a2ensite** command to enable the new virtual host:

```
sudo a2ensite projectlamp
```

We might want to disable the default website that comes installed with Apache. This is required if we are not using a custom domain name, because in this case Apache's default configuration would overwrite our virtual host. To disable Apache's default website use **a2dissite** command, type:

```
sudo a2dissite 000-default
```

```
ubuntu@ip-172-31-8-208:~$ sudo a2dissite 000-default
Site 000-default disabled.
To activate the new configuration, you need to run:
    systemctl reload apache2
ubuntu@ip-172-31-8-208:~$
```

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

```
sudo apache2ctl configtest
```

```
ubuntu@ip-172-31-8-208:~$ sudo apache2ctl configtest
Syntax OK
ubuntu@ip-172-31-8-208:~$
```

Finally, reload Apache so these changes take effect:

```
sudo systemctl reload apache2
```

Your new website is now active, but the web root **/var/www/projectlamp** is still empty. 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
```

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

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

You will see something like this;



You can leave this file in place as a temporary landing page for your application until you set up an `index.php` file to replace it. Once you do that, remember to remove or rename the `index.html` file from your document root, as it would take precedence over an `index.php` file by default.

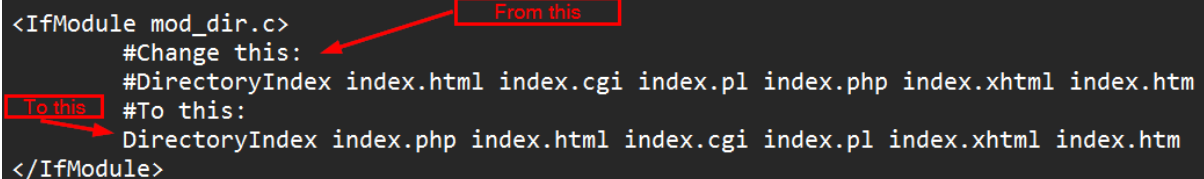
STEP 5 — ENABLE PHP ON THE WEBSITE

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 we want to change this behavior, we will 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
```

change from what you have in the image below first part to second part, i.e



```
<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 our server.

Now that we have a custom location to host our website's files and folders, we'll create a PHP test script to confirm that Apache is able to handle and process requests for PHP files.

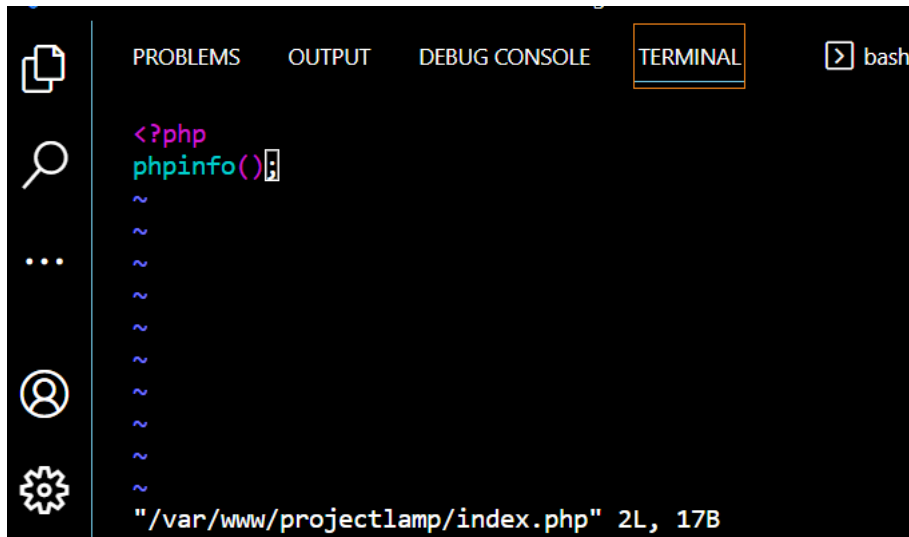
Create a new file named **index.php** inside your custom web root folder:

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



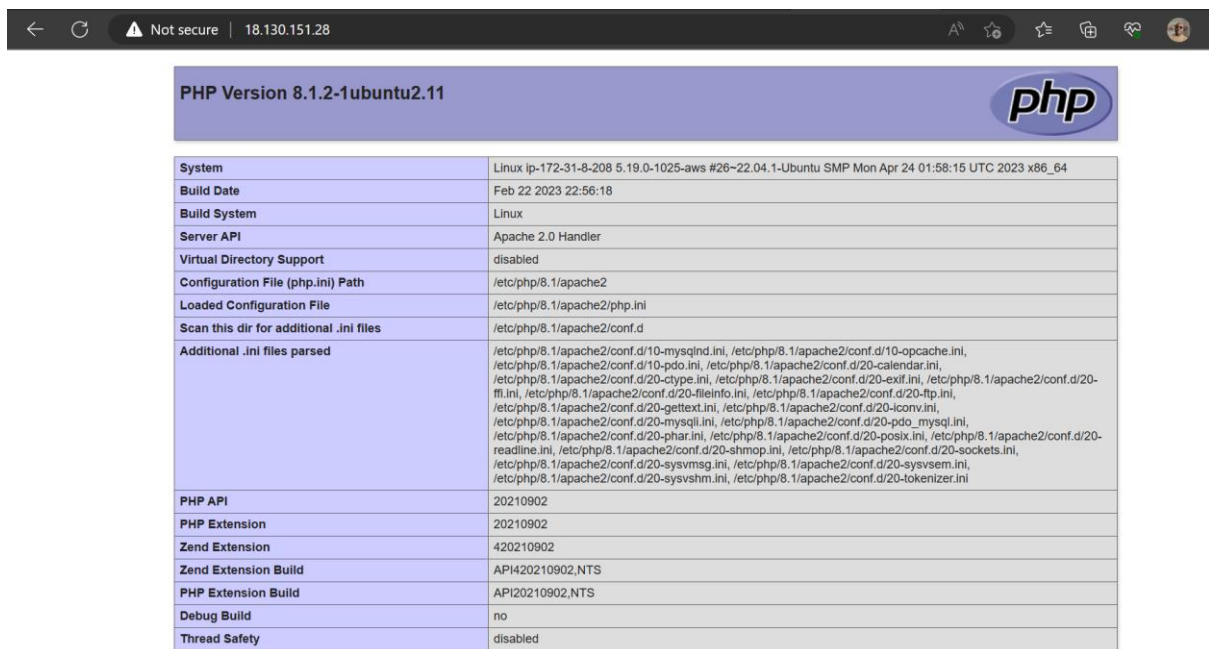
```
<?php
phpinfo();
```

It will look like this;



The screenshot shows a code editor with a dark theme. On the left is a sidebar with icons for file explorer, search, and settings. The main area has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab is active, showing a bash prompt. The code <?php phpinfo(); is being typed. Below the code, there are several tilde (~) characters representing line numbers. At the bottom, the file path "/var/www/projectlamp/index.php" is shown with line 2L and column 17B.

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



The screenshot shows a web browser window with the address bar displaying "18.130.151.28". The page title is "PHP Version 8.1.2-1ubuntu2.11". The page content is a table with system and configuration information.

PHP Version 8.1.2-1ubuntu2.11	
System	Linux ip-172-31-8-208 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	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/8.1/apache2
Loaded Configuration File	/etc/php/8.1/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/8.1/apache2/conf.d
Additional .ini files parsed	/etc/php/8.1/apache2/conf.d/10-mysqlnd.ini, /etc/php/8.1/apache2/conf.d/10-opcache.ini, /etc/php/8.1/apache2/conf.d/10-pdo.ini, /etc/php/8.1/apache2/conf.d/20-calendar.ini, /etc/php/8.1/apache2/conf.d/20-ctype.ini, /etc/php/8.1/apache2/conf.d/20-exif.ini, /etc/php/8.1/apache2/conf.d/20-ffi.ini, /etc/php/8.1/apache2/conf.d/20-fileinfo.ini, /etc/php/8.1/apache2/conf.d/20-ftp.ini, /etc/php/8.1/apache2/conf.d/20-gettext.ini, /etc/php/8.1/apache2/conf.d/20-iconv.ini, /etc/php/8.1/apache2/conf.d/20-mysql.ini, /etc/php/8.1/apache2/conf.d/20-pdo_mysql.ini, /etc/php/8.1/apache2/conf.d/20-phar.ini, /etc/php/8.1/apache2/conf.d/20-posix.ini, /etc/php/8.1/apache2/conf.d/20-readline.ini, /etc/php/8.1/apache2/conf.d/20-shmop.ini, /etc/php/8.1/apache2/conf.d/20-sockets.ini, /etc/php/8.1/apache2/conf.d/20-sysvmsg.ini, /etc/php/8.1/apache2/conf.d/20-sysvsem.ini, /etc/php/8.1/apache2/conf.d/20-sysvshm.ini, /etc/php/8.1/apache2/conf.d/20-tokenizer.ini
PHP API	20210902
PHP Extension	20210902
Zend Extension	420210902
Zend Extension Build	API420210902,NTS
PHP Extension Build	API20210902,NTS
Debug Build	no
Thread Safety	disabled

If you can see this page in your browser, then your PHP installation is working as expected.

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
```

You can always recreate this page if you need to access the information again later.

Congratulations! You have succeeded in deploying a LAMP stack website in AWS Cloud!

PS: Remember to terminate your EC2 instance.

