WEB STACK IMPLEMENTATION IN AWS

Basically a stack is just a way of keeping things in order one over the other.

Therefore a technology stack keeps or it is a combination of several technologies stacked together in order to build an application.

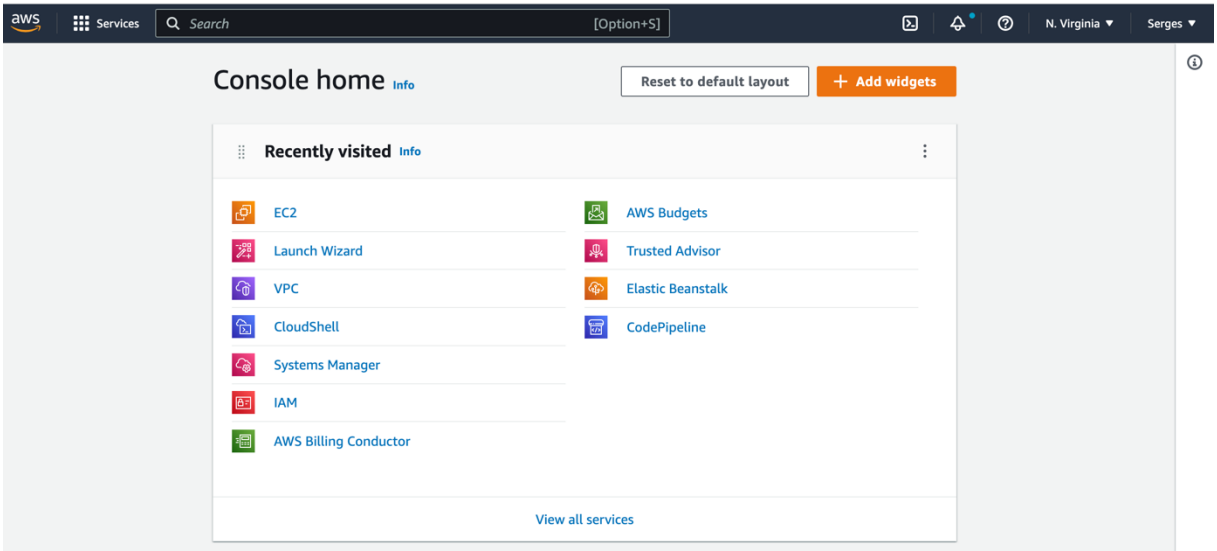
They are acronymns for individual technologies used together for a specific technology product. some examples are…

- 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

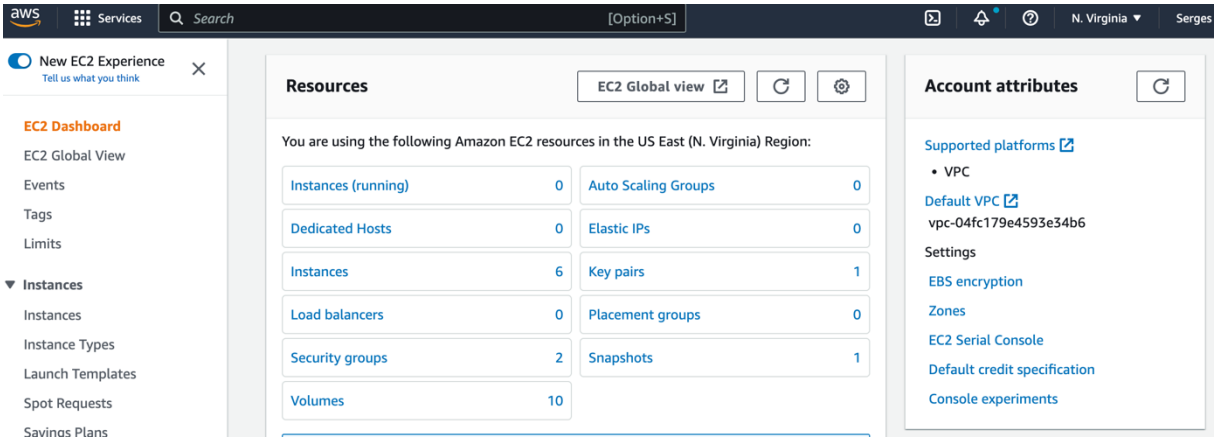
Prerequisites

In order to complete this project you will need an AWS account and a virtual server with Ubuntu Server OS.

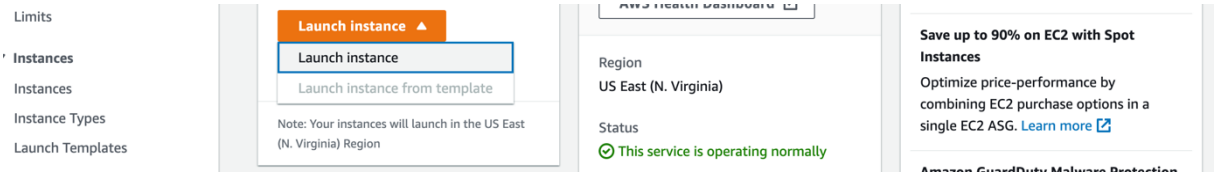
After creating your AWS account, log in to reach the console home as seen below.



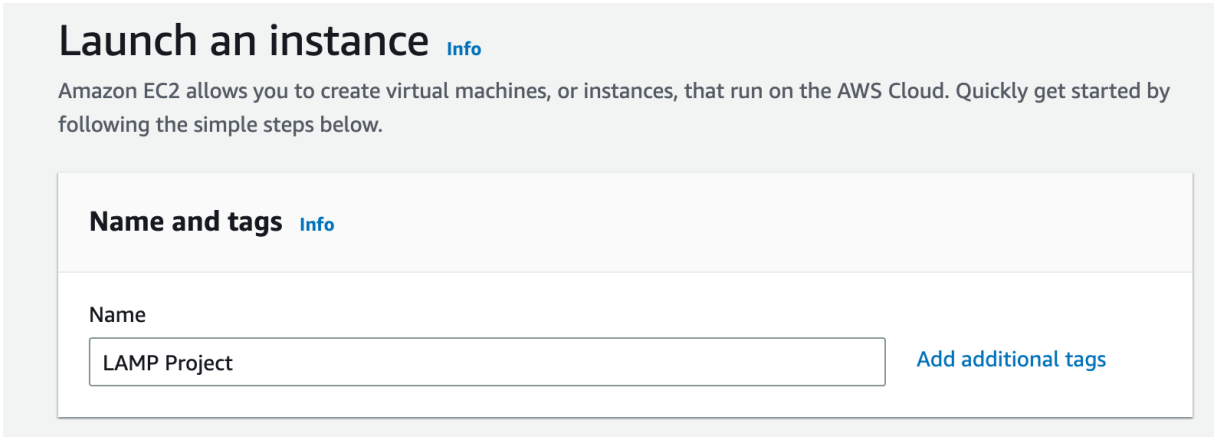
Click on the EC2 service in order to create an instance.



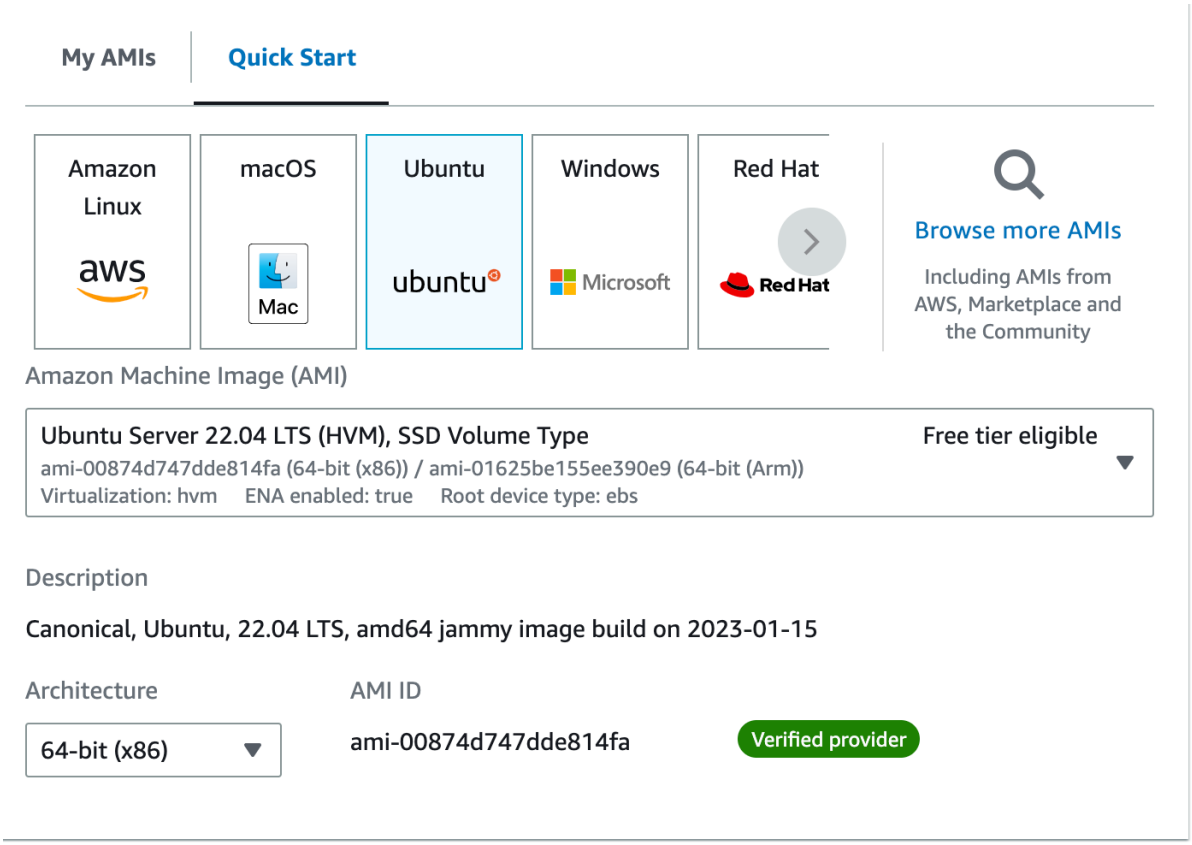
I have already created some instances. So to create one click on launch instance.



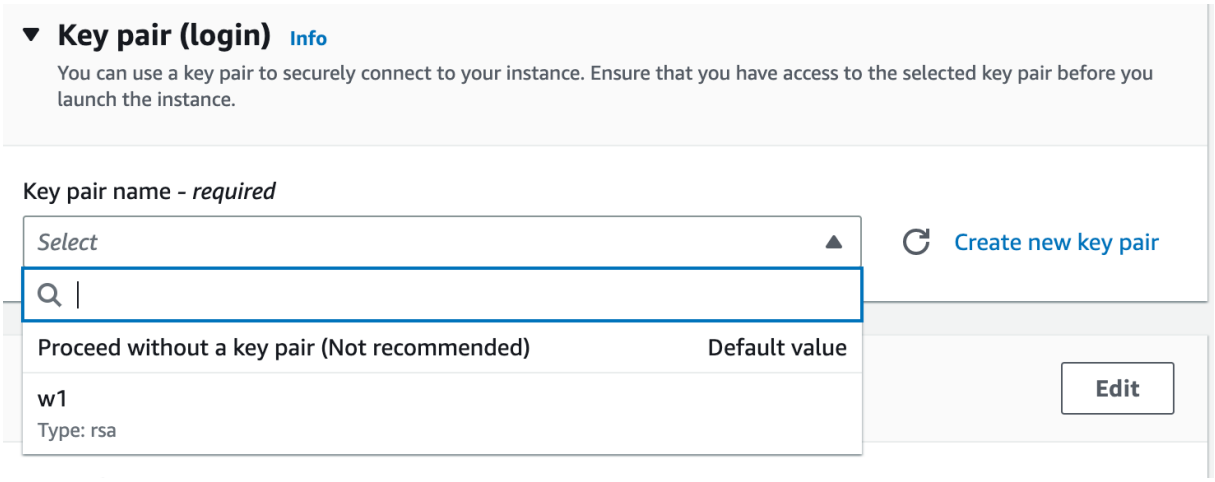
Give a name to your server.



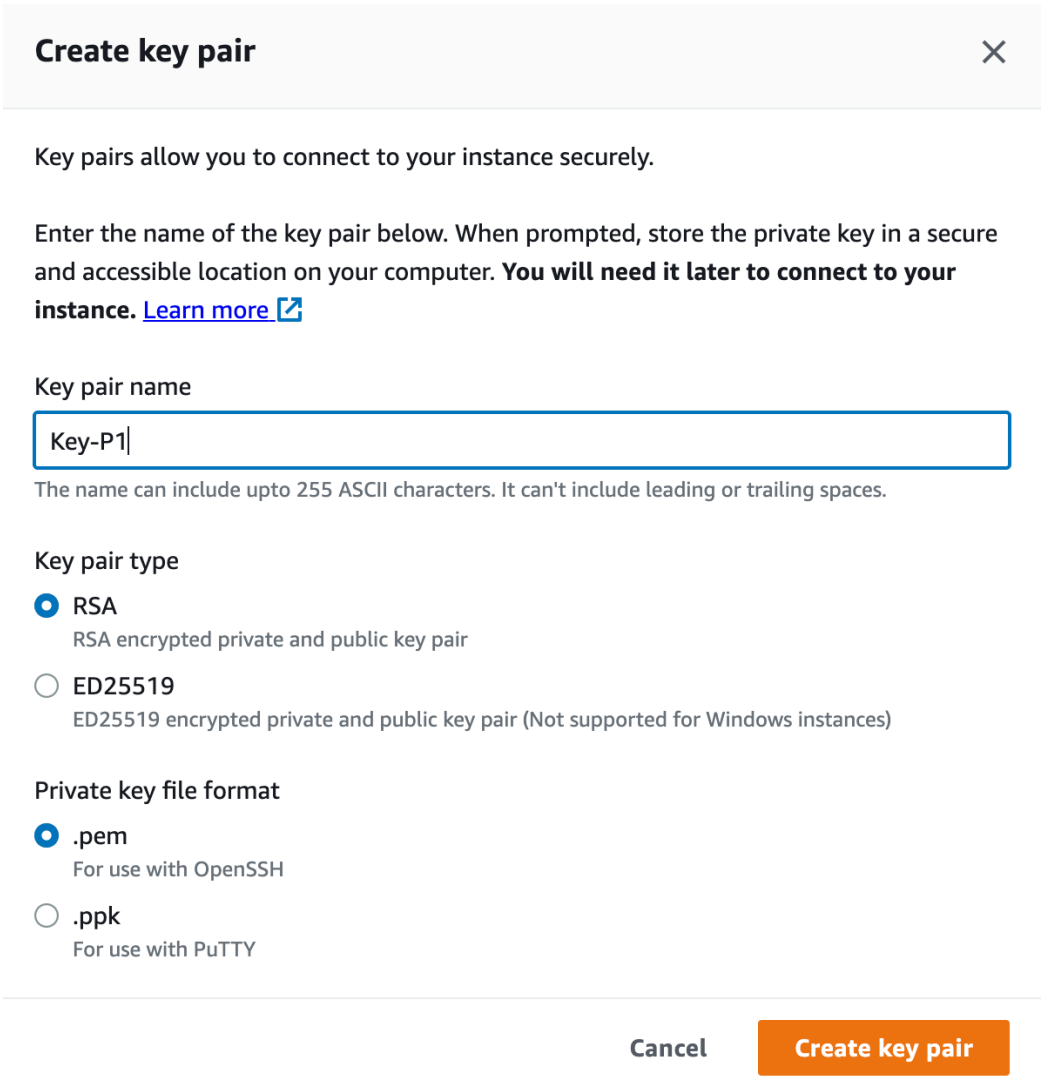
Launch a new "EC2" instance of t2.micro family with Ubuntu Server 22.04 LTS(HVM) 64 bit



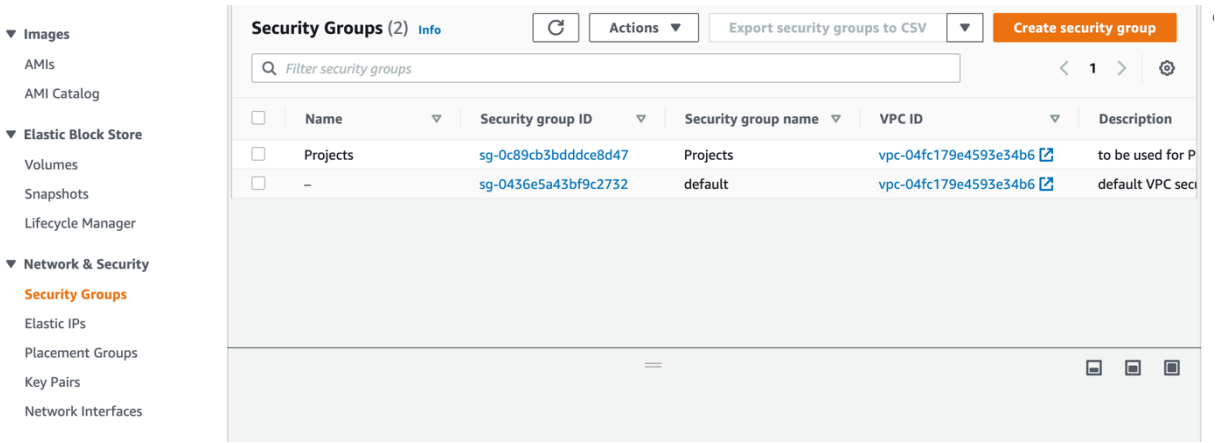
You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.



Click on Create new key pair then click on Create key pair.

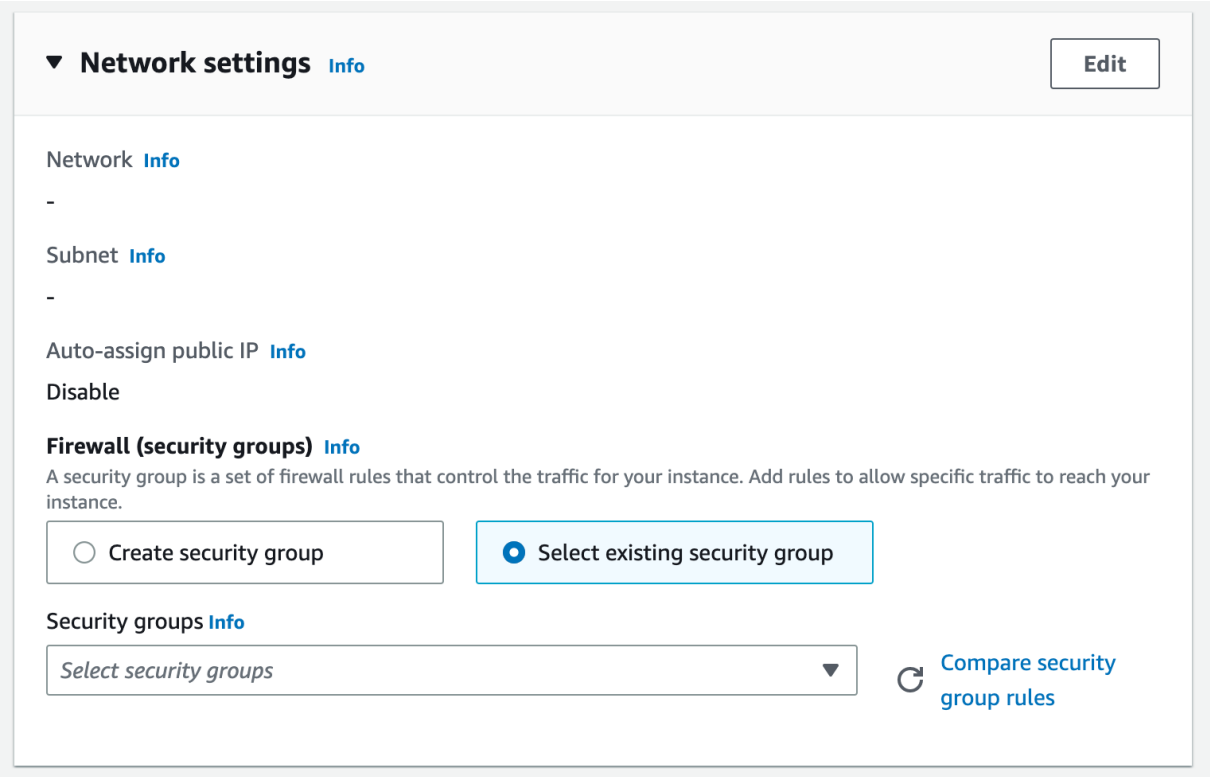


Configure Security Group; A security group acts as a virtual firewall for your EC2 instances to control incoming and outgoing traffic. Inbound rules control the incoming traffic to your instance, and outbound rules control the outgoing traffic from your instance. When you launch an instance, you can specify one or more security groups.

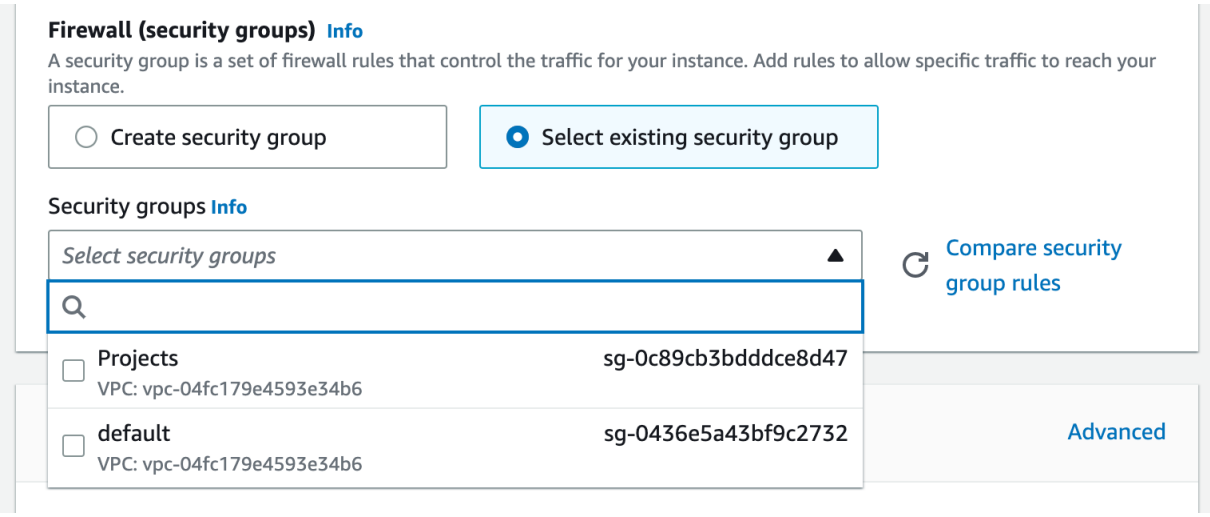


Click on create security group.

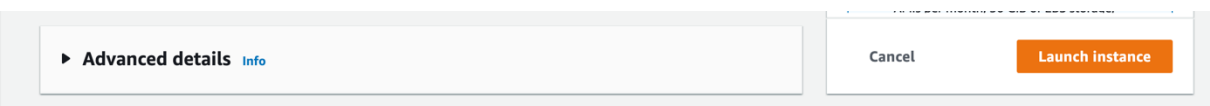
Under the Network settings, select existing security group.



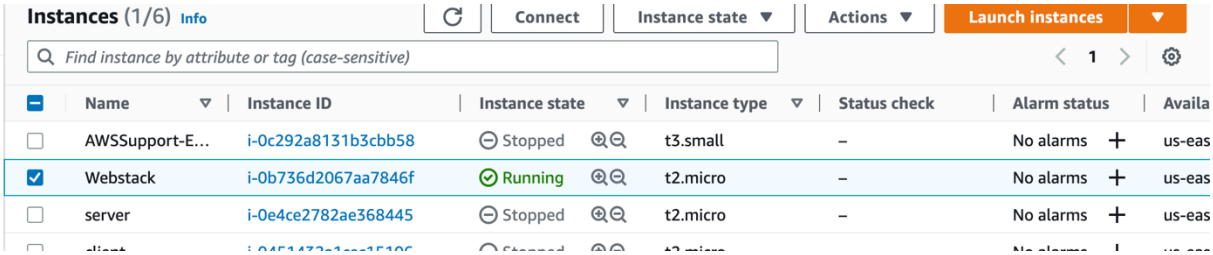
Click on security groups to select the security groups to select the security group created.



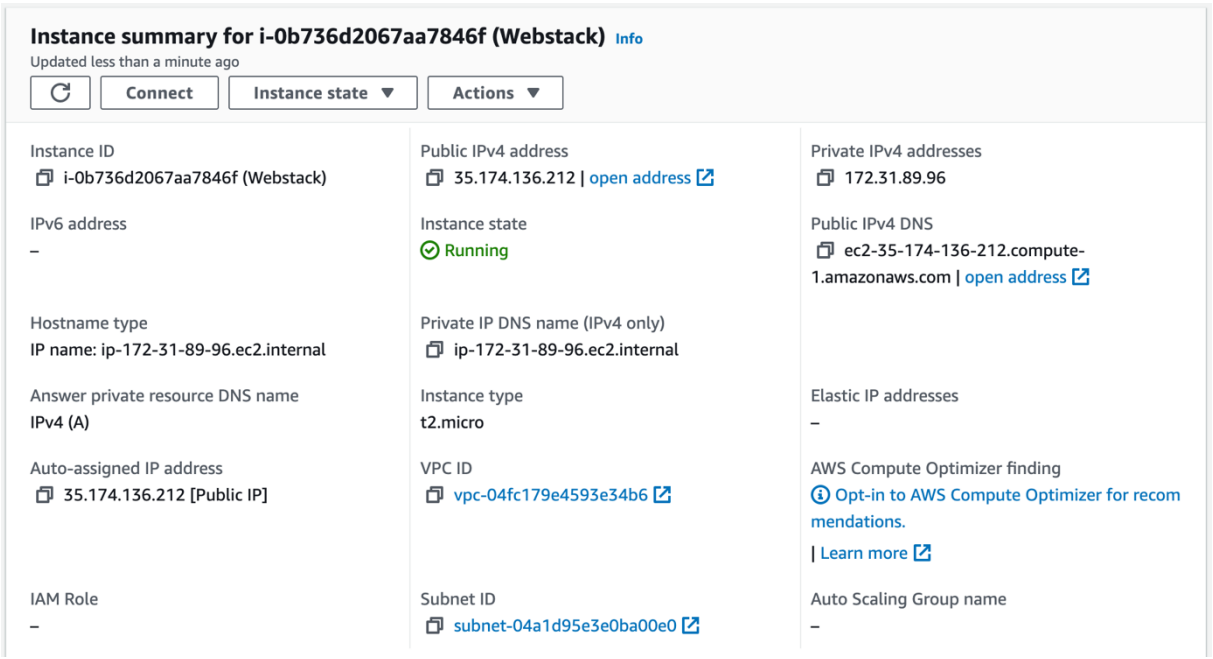
Then click on launch instance.



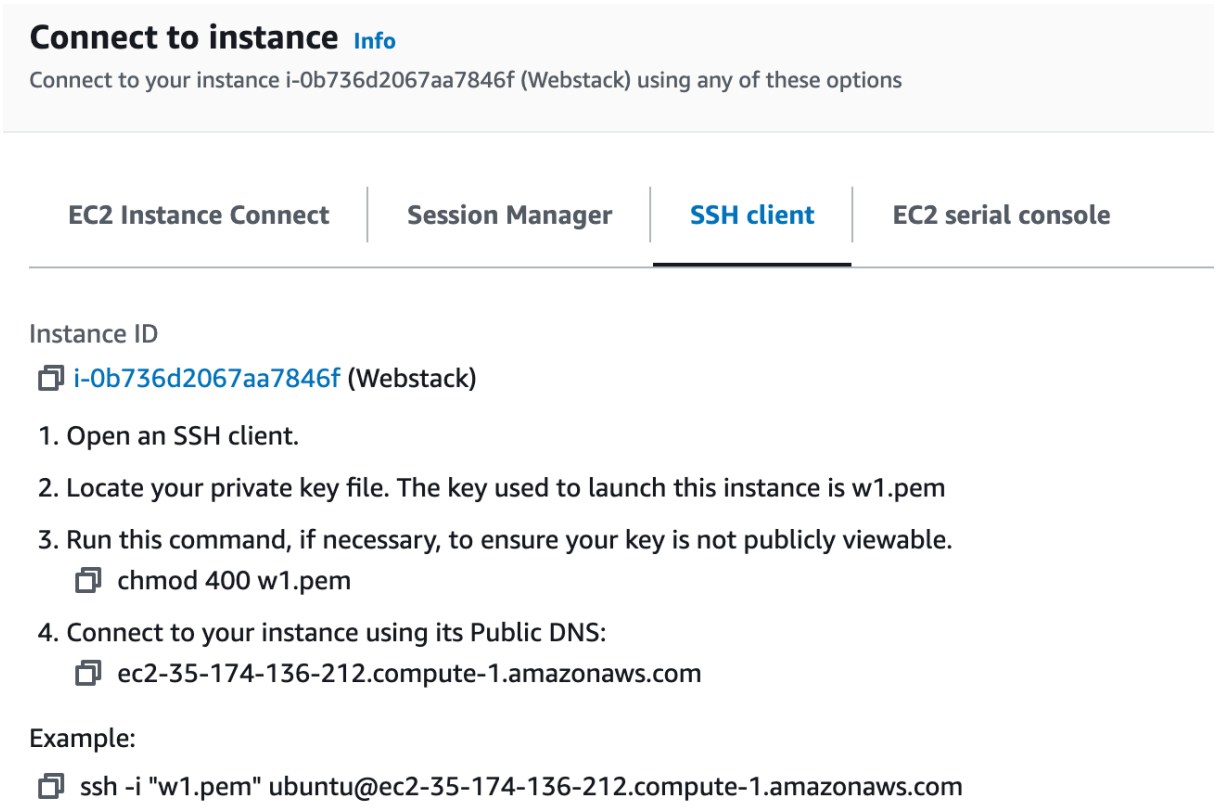
Click on instance to see if the new created one is running.



Click on the running instance in order to see the instance details.



Next is to connect my instance to an SSH Client by clicking on connect.



Connecting to EC2 terminal

Using the terminal on MAC/Linux

- The terminal is already installed by default. You just need to open it up.
- You do not need to convert to a .ppk file. Just use the same key as downloaded from AWS.
- Change directory into the location where your PEM file is. Most likely will be in the Downloads folder

  cd ~/Downloads

- Change premissions for the private key file (.pem), otherwise you can get an error "Bad permissions"

  sudo chmod 0400 <private-key-name>.pem

- Connect to the instance by running

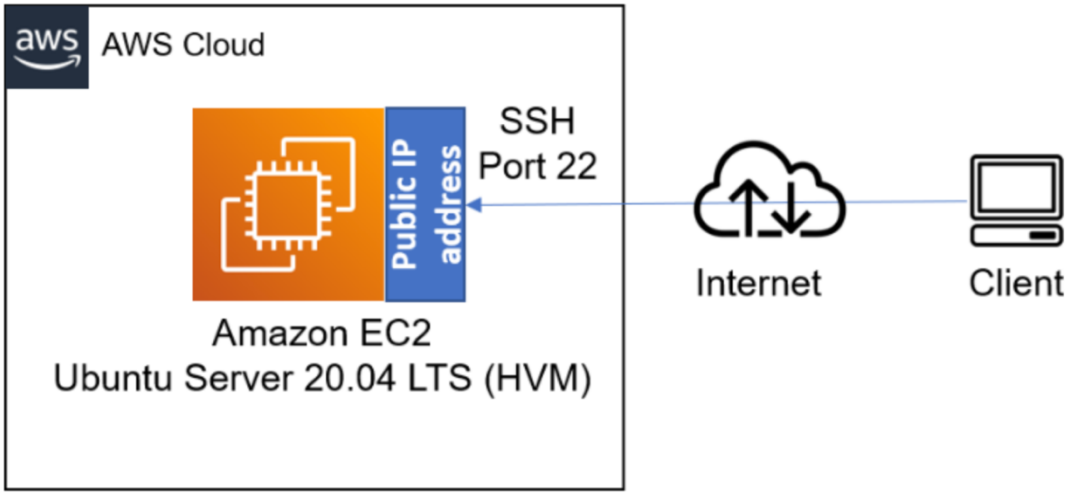
  ssh -i <private-key-name>.pem ubuntu@<Public-IP-address>

```
(base) [redacted] Downloads % chmod 400 w1.pem
(base) [redacted] Downloads %  ssh -i "w1.pem" ubuntu@ec2-35-174-136-212.compute-1.amazon
aws.com
The authenticity of host 'ec2-35-174-136-212.compute-1.amazonaws.com (35.174.136.212)' can't be est
ablished.
ED25519 key fingerprint is SHA256:xlnxO3n014jF9J5o9ObWP1Px5/f2I+taEd1yIoT8tUw.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:13: 34.226.152.237
    ~/.ssh/known_hosts:16: ec2-34-226-152-237.compute-1.amazonaws.com
    ~/.ssh/known_hosts:17: ec2-54-144-182-224.compute-1.amazonaws.com
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

Answer yes to carry on

```
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-35-174-136-212.compute-1.amazonaws.com' (ED25519) to the list of kn
own hosts.
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-1028-aws x86_64)

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

  System information as of Mon Jan 30 13:36:25 UTC 2023

  System load:  0.0               Processes:             100
  Usage of /:   61.3% of 7.57GB   Users logged in:       0
  Memory usage: 58%               IPv4 address for eth0: 172.31.89.96
  Swap usage:   0%

 * Ubuntu Pro delivers the most comprehensive open source security and
   compliance features.

   https://ubuntu.com/aws/pro

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


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

Last login: Tue Jan 17 09:49:57 2023 from 88.175.121.87
ubuntu@ip-172-31-89-96:~$
```

Congratulations! You have just created your very first Linux Server in the Cloud and our set up looks like this now: (You are the client)



Step 1 — Installing Apache and Updating the Firewall

- We need to Install Apache using Ubuntu's package manager, apt:

  $ sudo apt update

  $ sudo apt install apache2

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

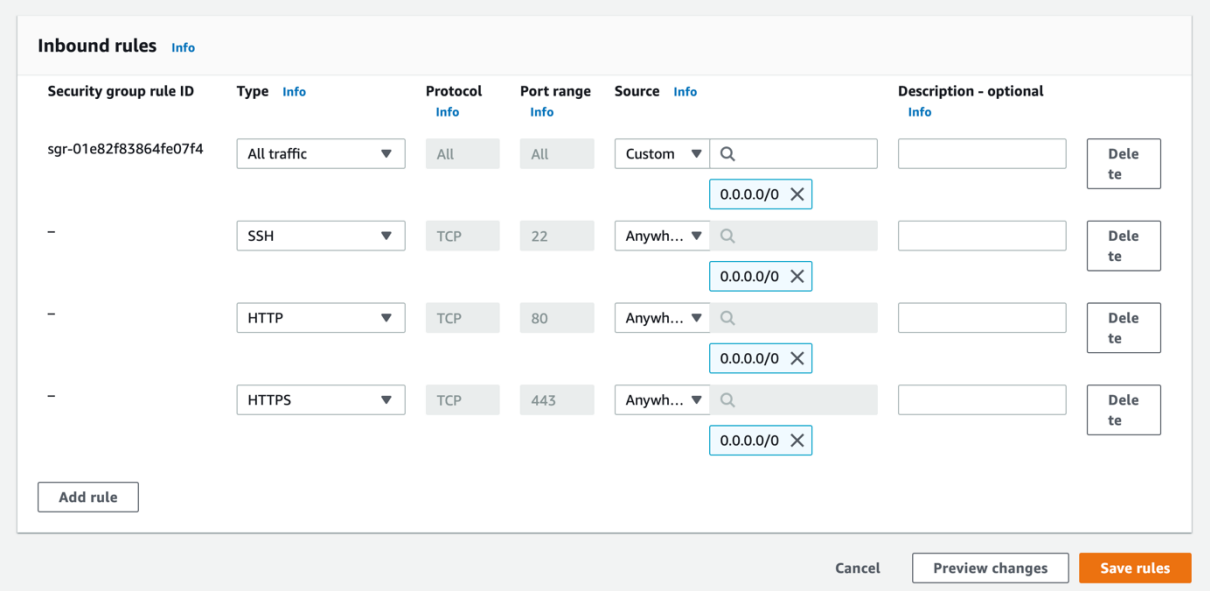
  $ sudo systemctl status apache2

```
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-49-178:~$  sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2023-02-02 23:56:56 UTC; 4min 58s ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 2591 (apache2)
      Tasks: 55 (limit: 1143)
     Memory: 4.8M
        CPU: 42ms
     CGroup: /system.slice/apache2.service
             ├─2591 /usr/sbin/apache2 -k start
             ├─2593 /usr/sbin/apache2 -k start
             └─2594 /usr/sbin/apache2 -k start

Feb 02 23:56:56 ip-172-31-49-178 systemd[1]: Starting The Apache HTTP Server...
Feb 02 23:56:56 ip-172-31-49-178 systemd[1]: Started The Apache HTTP Server.
```

If it is green and running, then you did everything correctly – 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

To open our port 80, i went back to the instance, clicked on security, clicked on edit inbound rules and i add rules. i added port 80 for http and port 443 https and i cliked on save rules



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').

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

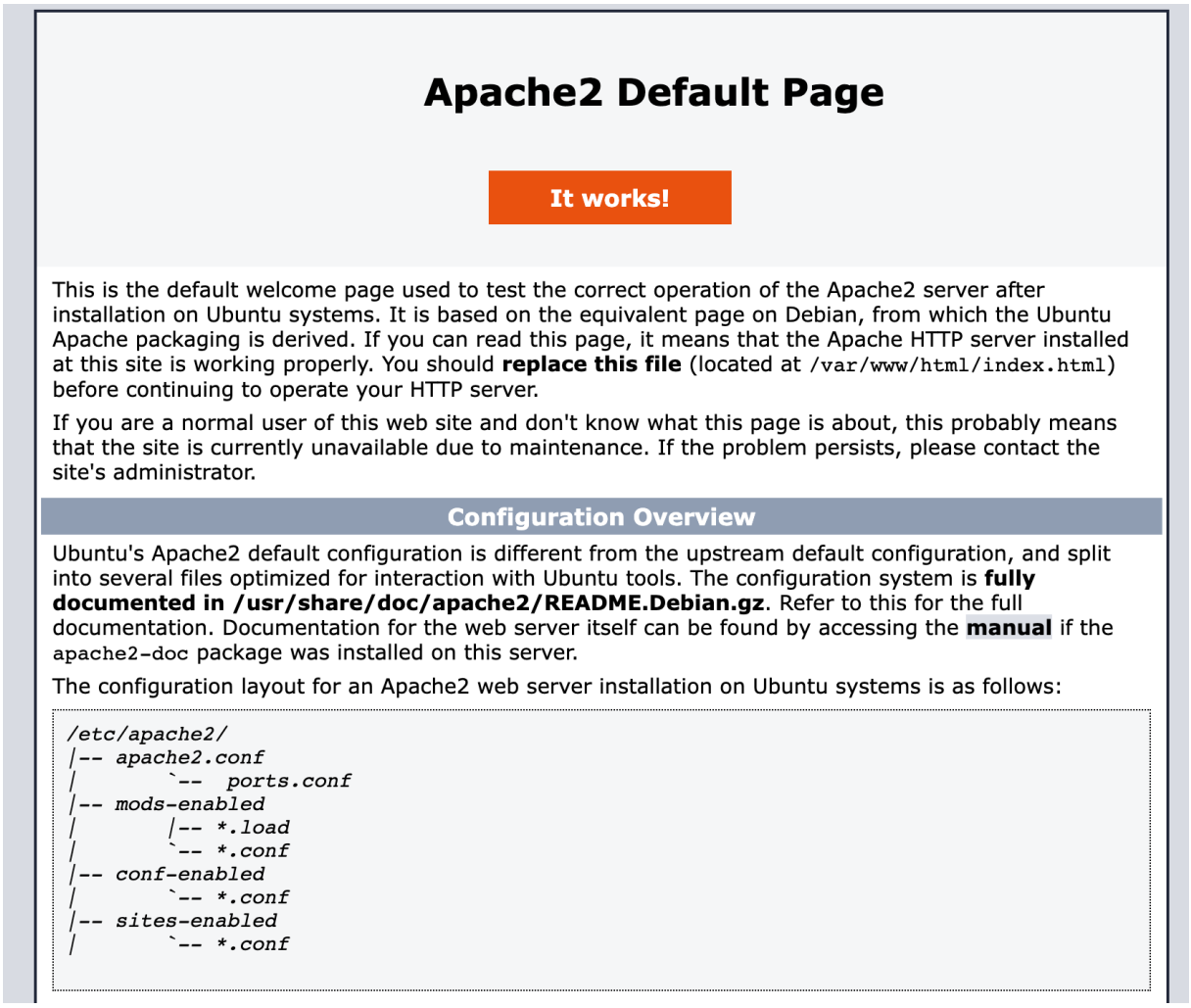
Now it is time for us to test how our Apache HTTP server can respond to requests from the Internet. Open a web browser of your choice and try to access following url

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

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



STEP 2 — INSTALLING MYSQL

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

Again, use ‘apt’ to acquire and install this software:

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

When the installation is 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 by the use of sudo when running this command. You should see output like this:

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-49-178:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.32-0ubuntu0.22.04.2 (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>
```

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 PassWord.1.

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

Exit the MySQL shell with:

mysql> exit

- After the installation it is recommended that we run a security script that comes pre-installed with MySQL. This script will remove some insecure default settings and lockdown access to our database system.
- Start the interactive Script by running:

  $ sudo mysql_secure_installation

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

```
ubuntu@ip-172-31-49-178:~$ sudo mysql_secure_installation

Securing the MySQL server deployment.

Connecting to MySQL using a blank password.

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:
```

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

when you are done, you will get this **output:**

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


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) : y
Success.

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.


Remove test database and access to it? (Press y|Y for Yes, any other key for No) : y
 - Dropping test database...
Success.

 - Removing privileges on test database...
Success.

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) : y
Success.

All done!
```

- When you're finished, test if you're able to login to the MySQL Console by typing:

  $ sudo mysql

```
ubuntu@ip-172-31-49-178:~$ sudo mysql -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 21
Server version: 8.0.32-0ubuntu0.22.04.2 (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>
```

- To exit the MySQL Console, type: exit

  mysql> exit

STEP 3 — INSTALLING PHP

You have Apache installed to serve your content and MySQL installed to store and manage your 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, you'll need php-mysql, a PHP module that allows PHP to communicate with MySQL-based databases. You'll 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 is finished, you can run the following command to confirm your PHP version:

$ php -v

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-49-178:~$ php -v
PHP 8.1.2-1ubuntu2.10 (cli) (built: Jan 16 2023 15:19:49) (NTS)
Copyright (c) The PHP Group
Zend Engine v4.1.2, Copyright (c) Zend Technologies
    with Zend OPcache v8.1.2-1ubuntu2.10, Copyright (c), by Zend Technologies
ubuntu@ip-172-31-49-178:~$
```

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

We will configure our first Virtual Host in the next step.

Step 4 — Creating a Virtual Host for your Website using Apache

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

Apache on Ubuntu 20.04 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 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 the $USER environment variable, which will reference your current user.

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

- We can now open a new configuration file in Apache's sites-available directory using our preferred command-line editor. Here, we'll be using vi

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 paste the text:

```
<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>
~
```

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;

000-default.conf default-ssl.conf projectlamp.conf

```
ubuntu@ip-172-31-49-178:~$ sudo ls /etc/apache2/sites-available
000-default.conf  default-ssl.conf  projectlamp.conf
```

With this VirtualHost configuration, we're telling Apache to serve projectlamp using /var/www/projectlampl as its web root directory. If you would like to test Apache without a domain name, you can remove or comment out the options ServerName and ServerAlias by adding a # character in the beginning of each option's lines. Adding the # character there will tell the program to skip processing the instructions on those lines.

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

$ sudo a2ensite projectlamp

```
ubuntu@ip-172-31-49-178:~$ sudo a2ensite projectlamp
Enabling site projectlamp.
To activate the new configuration, you need to run:
  systemctl reload apache2
```

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

$ sudo a2dissite 000-default

```
ubuntu@ip-172-31-49-178:~$ sudo a2dissite 000-default
Site 000-default disabled.
To activate the new configuration, you need to run:
  systemctl reload apache2
```

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

$ sudo apache2ctl configtest

```
ubuntu@ip-172-31-49-178:~$ sudo apache2ctl configtest
Syntax OK
```

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:

Type and run :

$ sudo vi /var/www/projectlamp/index.html

see my input:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN">
<html>
<head>
  <title>Ma première page avec du style</title>
</head>

<body>

<!-- Menu de navigation du site -->
<ul class="navbar">
  <li><a href="index.html">Home page</a>
  <li><a href="reflexions.html">Réflexions</a>
  <li><a href="ville.html">Ma ville</a>
  <li><a href="liens.html">Liens</a>
</ul>

<!-- Contenu principal -->
<h1>Ma première page avec du style</h1>

<p>Bienvenue sur ma page avec du style!

<p>Il lui manque des images, mais au moins, elle a du style. Et elle a desliens, même s'ils ne mènent null
e part...
&hellip;

<p>Je devrais étayer, mais je ne sais comment encore.

<!-- Signer et dater la page, c'est une question de politesse! -->
<address>Fait le 5 avril 2004<br>
  par moi.</address>

</body>
</html>
```

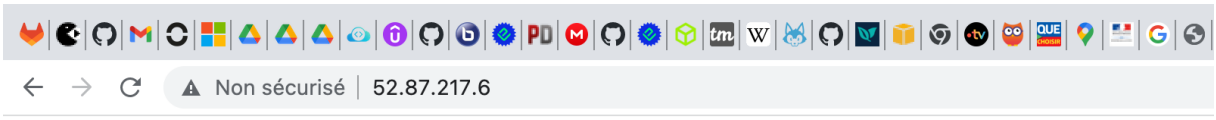
Now go to your browser and try to open your website URL using IP address:

http://<public-ip-address>:80

or you can also browse using your public dns. the result is the same

http://<Public-DNS-Name>:80

see my output :



- Home page
- RÃ©flexions
- Ma ville
- Liens

Ma premiÃ¨re page avec du style

Bienvenue sur ma page avec du style!

Il lui manque des images, mais au moins, elle a du style. Et elle a desliens, mÃªme s'ils ne mÃ¨nent nulle part... …

Je devrais Ã©tayer, mais je ne sais comment encore.

Fait le 5 avril 2004
par moi.

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.

To check your Public IP from the Ubuntu shell, run :

$(curl -s http://169.254.169.254/latest/meta-data/public-hostname)

$(curl -s http://169.254.169.254/latest/meta-data/public-ipv4)

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

#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.html

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.

Now that you have a custom location to host your 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.

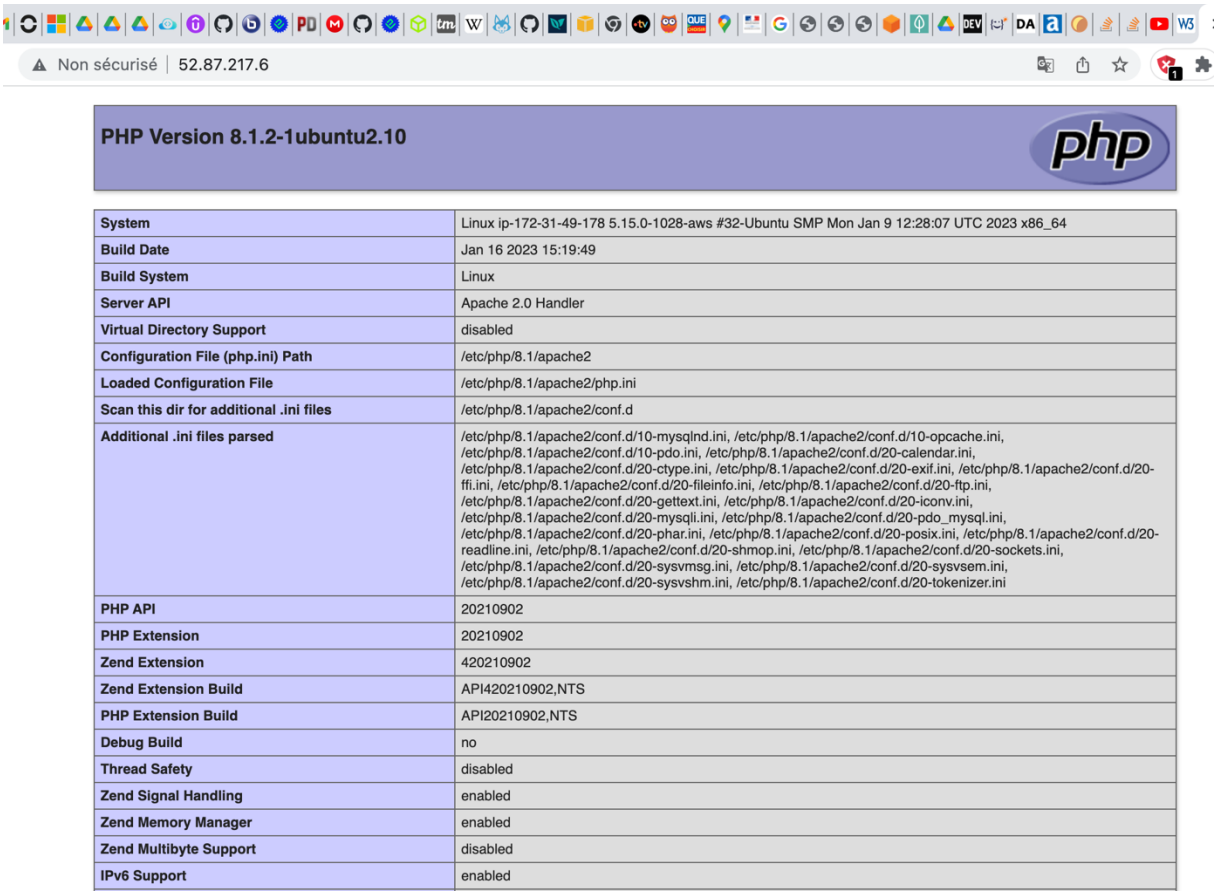
Create a new file named index.php inside your custom web root folder:

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

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

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

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



System	Linux ip-172-31-49-178 5.15.0-1028-aws #32-Ubuntu SMP Mon Jan 9 12:28:07 UTC 2023 x86_64
Build Date	Jan 16 2023 15:19:49
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-mysqli.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
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
IPv6 Support	enabled

- This page provides information about your server from the perspective of PHP. It is useful for debugging and to ensure that your settings are being applied correctly.
- 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