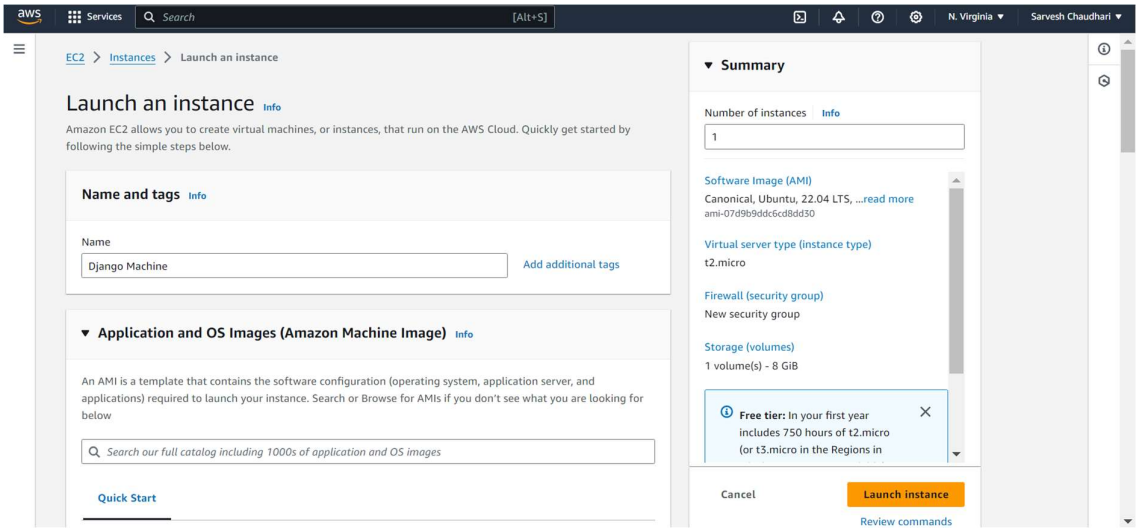
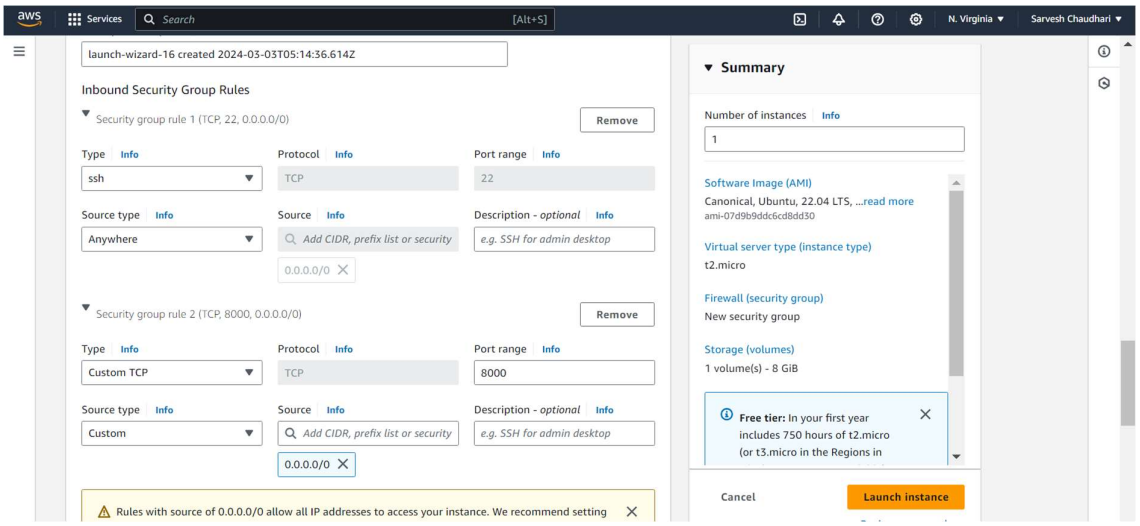
Deploy Django, Flask-app & LampStack on EC2

1. Deploy Django instance using EC2

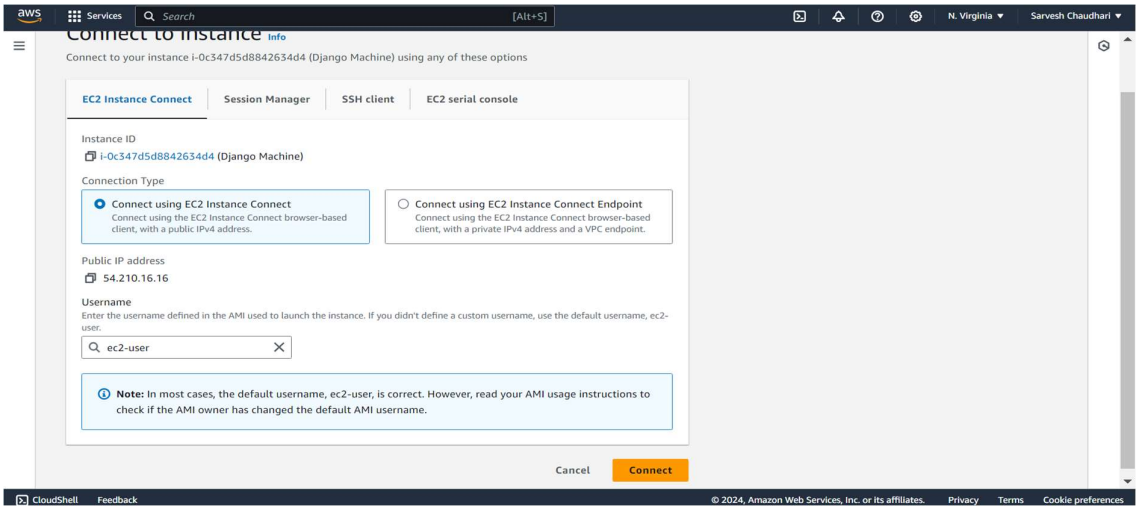
1st create Ubuntu machine



In SG add one Custom TCP with port range 8000



Now connect to our instance.



Now connect Instance

SSH into your AWS EC2 instance and update it using the following command:

Next, clone your Django project repository onto the instance using git

```
ubuntu@ip-172-31-31-101:~$ apt-get update
Reading package lists... Done
E: Could not open lock file /var/lib/apt/lists/lock - open (13: Permission denied)
E: Unable to lock directory /var/lib/apt/lists/
W: Problem unlinking the file /var/cache/apt/pkgcache.bin - RemoveCaches (13: Permission denied)
W: Problem unlinking the file /var/cache/apt/srcpkgcache.bin - RemoveCaches (13: Permission denied)
ubuntu@ip-172-31-31-101:~$ git clone https://github.com/yeshwanthlm/django-on-ec2.git
Cloning into 'django-on-ec2'...
remote: Enumerating objects: 304, done.
remote: Counting objects: 100% (74/74), done.
remote: Compressing objects: 100% (23/23), done.
remote: Total 304 (delta 51), reused 51 (delta 51), pack-reused 230
Receiving objects: 100% (304/304), 124.20 KiB | 6.54 MiB/s, done.
Resolving deltas: 100% (164/164), done.
```

Now install python3-pip and Django

```
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-31-101:~$ sudo apt install python3-pip -y
```

```
ubuntu@ip-172-31-31-101:~$ pip install django
Defaulting to user installation because normal site-packages is not writeable
Collecting django
  Downloading Django-5.0.2-py3-none-any.whl (8.2 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.2/8.2 MB 34.4 MB/s eta 0:00:00
Collecting sqlparse>=0.3.1
  Downloading sqlparse-0.4.4-py3-none-any.whl (41 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 41.2/41.2 KB 4.8 MB/s eta 0:00:00
Collecting asgiref<4,>=3.7.0
  Downloading asgiref-3.7.2-py3-none-any.whl (24 kB)
Collecting typing-extensions>=4
  Downloading typing_extensions-4.10.0-py3-none-any.whl (33 kB)
Installing collected packages: typing-extensions, sqlparse, asgiref, django
  WARNING: The script sqlformat is installed in '/home/ubuntu/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
  WARNING: The script django-admin is installed in '/home/ubuntu/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed asgiref-3.7.2 django-5.0.2 sqlparse-0.4.4 typing-extensions-4.10.0
ubuntu@ip-172-31-31-101:~$
```

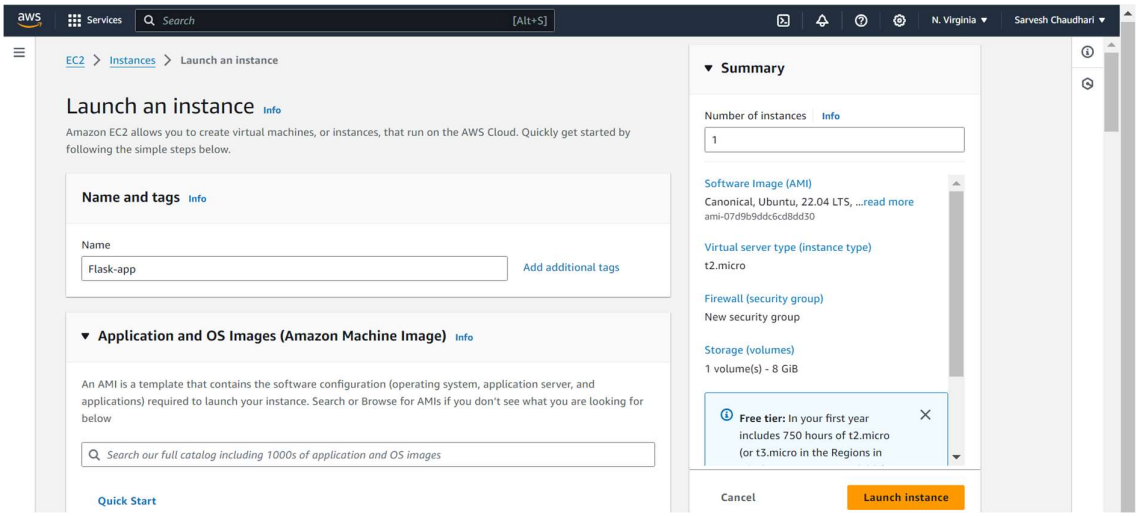
aws Services Search [Alt+S] N. Virginia Sarvesh Cha

```
ubuntu@ip-172-31-31-101:~$ cd django-on-ec2
ubuntu@ip-172-31-31-101:~/django-on-ec2$ python3 manage.py makemigrations
System check identified some issues:

WARNINGS:
todos.Todo: (models.W042) Auto-created primary key used when not defining a primary key type, by default 'django.db.models.AutoField'.
        HINT: Configure the DEFAULT_AUTO_FIELD setting or the TodosConfig.default_auto_field attribute to point to a subclass of AutoField, e.g. 'django.db.models.B
gAutoField'.
No changes detected
ubuntu@ip-172-31-31-101:~/django-on-ec2$ python3 manage.py migrate
System check identified some issues:

WARNINGS:
todos.Todo: (models.W042) Auto-created primary key used when not defining a primary key type, by default 'django.db.models.AutoField'.
        HINT: Configure the DEFAULT_AUTO_FIELD setting or the TodosConfig.default_auto_field attribute to point to a subclass of AutoField, e.g. 'django.db.models.B
gAutoField'.
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions, todos
Running migrations:
  No migrations to apply.
ubuntu@ip-172-31-31-101:~/django-on-ec2$ python3 manage.py runserver
Watching for file changes with StatReloader
Performing system checks...
```

```
ubuntu@ip-172-31-31-101:~/django-on-ec2$ python3 manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified some issues:

WARNINGS:
todos.Todo: (models.W042) Auto-created primary key used when not defining a primary key type, by default 'django.db.models.AutoField'.
        HINT: Configure the DEFAULT_AUTO_FIELD setting or the TodosConfig.default_auto_field attribute to point to a subclass of AutoField, e.g. 'django.db.models.B
gAutoField'.

System check identified 1 issue (0 silenced).
March 03, 2024 - 12:00:10
Django version 5.0.2, using settings 'todoApp.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

2. Deploy Flask app using EC2

Steps:- 1st create Ubuntu server



In SG allow http and https and create & launch instance.

2nd Install Required Packages

Update the package manager and install necessary packages like Python

```
Sudo apt-get update
```

```
sudo apt-get install python3-venv
```

3rd setup virtual environment

Create new directory

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-29-254:~$ mkdir helloworld
ubuntu@ip-172-31-29-254:~$ ls
helloworld
ubuntu@ip-172-31-29-254:~$ cd helloworld
ubuntu@ip-172-31-29-254:~/helloworld$
```

Create virtual environment

```
ubuntu@ip-172-31-29-254:~/helloworld$ python3 -m venv venv
ubuntu@ip-172-31-29-254:~/helloworld$
```

activate the virtual environment

```
ubuntu@ip-172-31-29-254:~/helloworld$ source venv/bin/activate
(venv) ubuntu@ip-172-31-29-254:~/helloworld$
```

Now install flask

```
ubuntu@ip-172-31-29-254:~/helloworld$ source venv/bin/activate
(venv) ubuntu@ip-172-31-29-254:~/helloworld$ pip install flask
Collecting flask
  Downloading flask-3.0.2-py3-none-any.whl (101 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 101.3/101.3 KB 2.4 MB/s eta 0:00:00
Collecting itsdangerous>=2.1.2
  Downloading itsdangerous-2.1.2-py3-none-any.whl (15 kB)
Collecting Jinja2>=3.1.2
  Downloading Jinja2-3.1.3-py3-none-any.whl (133 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 133.2/133.2 KB 14.5 MB/s eta 0:00:00
Collecting blinker>=1.6.2
  Downloading blinker-1.7.0-py3-none-any.whl (13 kB)
Collecting Werkzeug>=3.0.0
  Downloading werkzeug-3.0.1-py3-none-any.whl (226 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 226.7/226.7 KB 33.6 MB/s eta 0:00:00
Collecting click>=8.1.3
  Downloading click-8.1.7-py3-none-any.whl (97 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 97.9/97.9 KB 9.7 MB/s eta 0:00:00
Collecting MarkupSafe>=2.0
  Downloading MarkupSafe-2.1.5-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x
86_64.whl (25 kB)
Installing collected packages: MarkupSafe, itsdangerous, click, blinker, Werkzeu
g, Jinja2, flask
Successfully installed Jinja2-3.1.3 MarkupSafe-2.1.5 Werkzeug-3.0.1 blinker-1.7.
0 click-8.1.7 flask-3.0.2 itsdangerous-2.1.2
(venv) ubuntu@ip-172-31-29-254:~/helloworld$
```

4th Create a simple Flask API using vim command

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def hello_world():
 return 'Hello World!'

if __name__ == "__main__":
 app.run()
~
~
~
~
```

Then Ctrl + C to exit

```
(venv) ubuntu@ip-172-31-29-254:~/helloworld$ gunicorn -b 0.0.0.0:8000 app:app
[2024-03-03 14:35:30 +0000] [3215] [INFO] Starting gunicorn 21.2.0
[2024-03-03 14:35:30 +0000] [3215] [INFO] Listening at: http://0.0.0.0:8000 (3215)
[2024-03-03 14:35:30 +0000] [3215] [INFO] Using worker: sync
[2024-03-03 14:35:30 +0000] [3216] [INFO] Booting worker with pid: 3216
^C[2024-03-03 14:35:43 +0000] [3215] [INFO] Handling signal: int
[2024-03-03 14:35:44 +0000] [3215] [INFO] Shutting down: Master
(venv) ubuntu@ip-172-31-29-254:~/helloworld$
```

Now install Gunicorn

Pip install gunicorn

gunicorn -b 0.0.0.0:8000 [replace with your app name]:app

```
^C[2024-03-03 14:35:43 +0000] [3215] [INFO] Handling signal: int
[2024-03-03 14:35:44 +0000] [3215] [INFO] Shutting down: Master
(venv) ubuntu@ip-172-31-29-254:~/helloworld$ sudo nano /etc/systemd/system/he
lloworld.se

sudo: na: command not found
(venv) ubuntu@ip-172-31-29-254:~/helloworld$ sudo nano /etc/systemd/system/he
lloworld.service
(venv) ubuntu@ip-172-31-29-254:~/helloworld$
```

Enable the service with top 3 command

```
aws  Services  N. Virginia ▼  Sarvesh Chaudhari ▼
(venv) ubuntu@ip-172-31-29-254:~/helloworld$ sudo systemctl daemon-reload
(venv) ubuntu@ip-172-31-29-254:~/helloworld$ sudo systemctl start helloworld
(venv) ubuntu@ip-172-31-29-254:~/helloworld$ sudo systemctl enable helloworld
Created symlink /etc/systemd/system/multi-user.target.wants/helloworld.servic
e → /etc/systemd/system/helloworld.service.
(venv) ubuntu@ip-172-31-29-254:~/helloworld$
```

Now install nginx

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
(venv) ubuntu@ip-172-31-29-254:~/helloworld$ sudo systemctl start nginx
(venv) ubuntu@ip-172-31-29-254:~/helloworld$ sudo systemctl enable nginx
Synchronizing state of nginx.service with SysV service script with /lib/syste
md/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable nginx
(venv) ubuntu@ip-172-31-29-254:~/helloworld$
```

create a new Nginx configuration file and restart machine.

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
(venv) ubuntu@ip-172-31-29-254:~/helloworld$ sudo systemctl start nginx
(venv) ubuntu@ip-172-31-29-254:~/helloworld$ sudo systemctl enable nginx
Synchronizing state of nginx.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable nginx
(venv) ubuntu@ip-172-31-29-254:~/helloworld$ sudo nano /etc/nginx/sites-available/default
(venv) ubuntu@ip-172-31-29-254:~/helloworld$ sudo systemctl restart nginx
(venv) ubuntu@ip-172-31-29-254:~/helloworld$
```

When we copy and paste public IP we can see nginx website.

Not secure 54.166.202.203

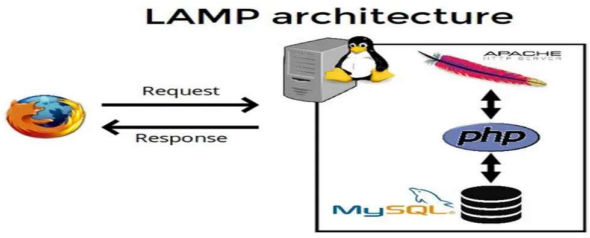
Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

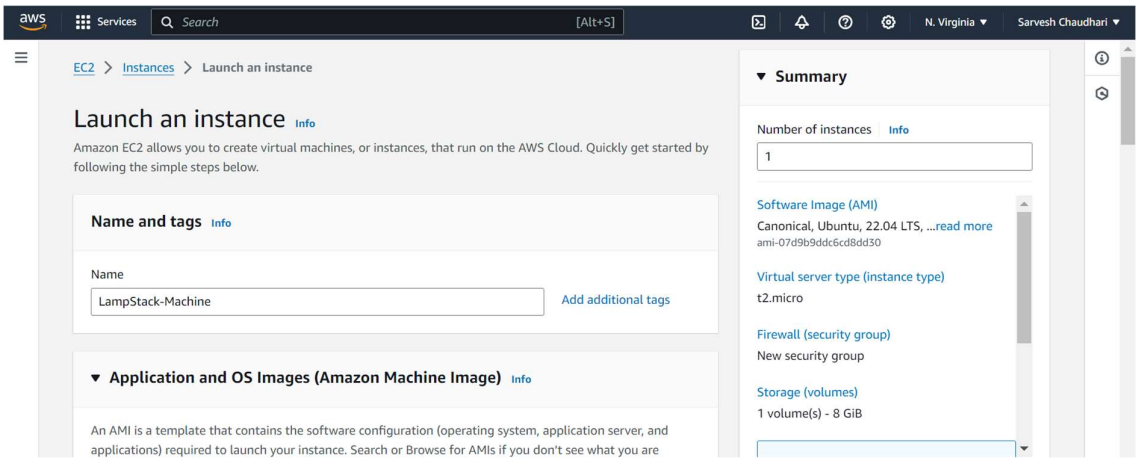
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

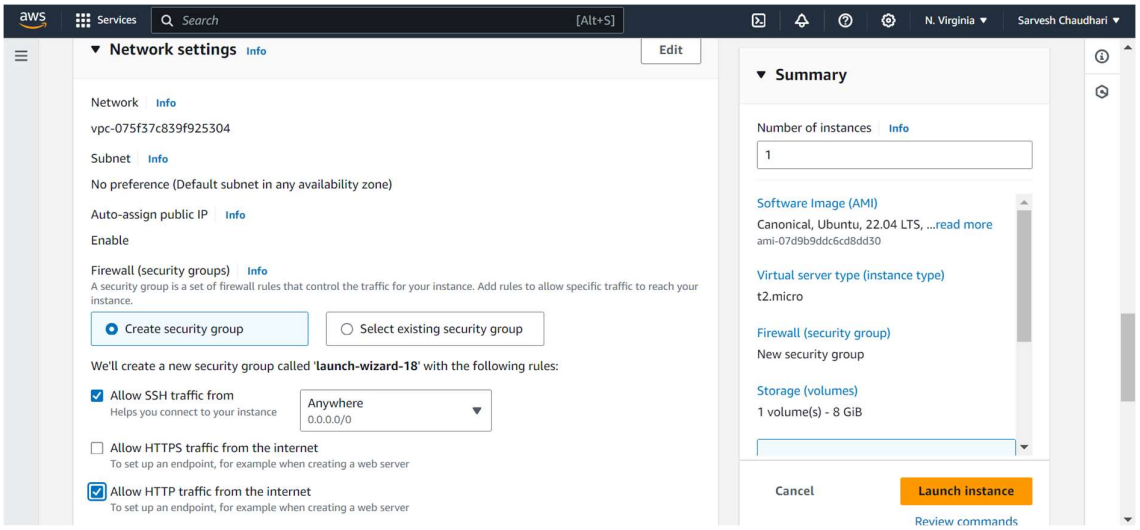
3. Deploy Lamp Stack on EC2 instance



1st Create the ubuntu instance



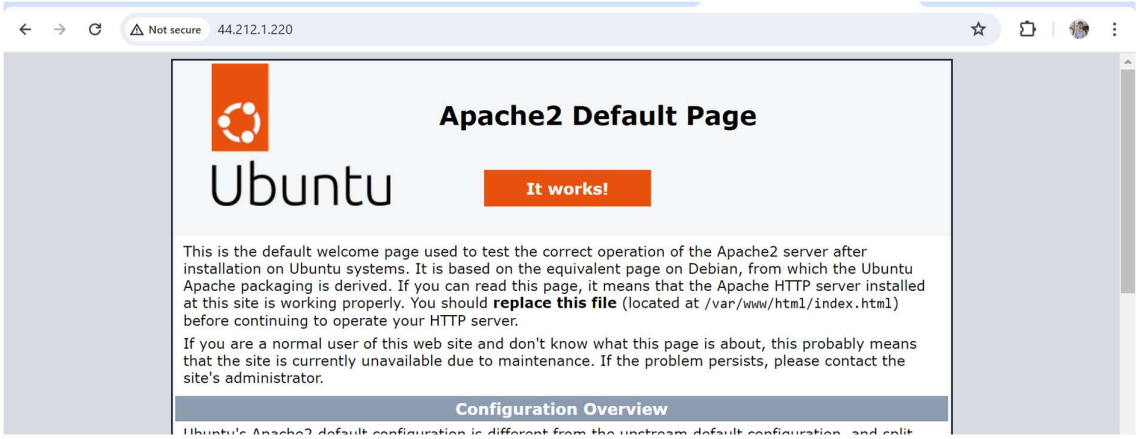
In SG give ssh and http



2nd Installing Apache Web Server

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

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

Now lets check it is working or not



Run the command to download MySQL server & log into the MySQL console

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

```
sudo mysql
```

```
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-172-31-92-76:~$ sudo mysql
```

i-0918501d0d84267ad (LampStack-Machine)

Entering and exiting from MySQL

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> select * from db
    -> exit
    ->
    ->
    ->
    ->
    ->
    ->
    -> \c
mysql> exit
Bye
ubuntu@ip-172-31-92-76:~$
```

i-0918501d0d84267ad (LampStack-Machine)

Now install php

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

```
    ->
    -> \c
mysql> exit
Bye
ubuntu@ip-172-31-92-76:~$ sudo apt install php libapache2-mod-php php-mysql
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libapache2-mod-php8.1 php-common php8.1 php8.1-cli php8.1-common php8.1-mysql php8.1-opcache php8.1-readline
Suggested packages:
  php-pear
```

create a directory for “projectlamp”

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

We will assign ownership of the directory with the user

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

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-92-76:~$ sudo mkdir /var/www/projectlamp
ubuntu@ip-172-31-92-76:~$ ls
ubuntu@ip-172-31-92-76:~$ sudo chown -R $USER:$USER /var/www/projectlamp
ubuntu@ip-172-31-92-76:~$
```

i-0918501d0d84267ad (LampStack-Machine)

Next up, we are creating a new configuration file that will reside in the Apache directory using vi/vim

```
ubuntu@ip-172-31-92-76:~$ ls
ubuntu@ip-172-31-92-76:~$ sudo chown -R $USER:$USER /var/www/projectlamp
ubuntu@ip-172-31-92-76:~$ sudo vi /etc/apache2/sites-available/projectlamp.conf
ubuntu@ip-172-31-92-76:~$ cat ^C
ubuntu@ip-172-31-92-76:~$ cat projectlamp.conf
cat: projectlamp.conf: No such file or directory
ubuntu@ip-172-31-92-76:~$ cat projectlamp.conf
cat: projectlamp.conf: No such file or directory
ubuntu@ip-172-31-92-76:~$ cat /etc/apache2/sites-available/projectlamp.conf
<VirtualHost *:80>
    ServerName projectlamp
    ServerAlias www.projectlamp
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/projectlamp
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
ubuntu@ip-172-31-92-76:~$
```

Command to enable virtual host

```
sudo a2ensite projectlamp
```

```
ubuntu@ip-172-31-92-76:~$ sudo a2ensite projectlamp
Enabling site projectlamp.
To activate the new configuration, you need to run:
  systemctl reload apache2
ubuntu@ip-172-31-92-76:~$
```

i-0918501d0d84267ad (LampStack-Machine)

Command to disable the default Apache website so that we can replace it with our custom website:

```
sudo a2dissite 000-default
```

Command that checks and makes sure that our configuration file doesn't contain any errors:

```
sudo apache2ctl configtest
```

Command that reloads the Apache server to have the changes take effect:

```
sudo systemctl reload apache2
```

```
ubuntu@ip-172-31-92-76:~$ sudo a2ensite projectlamp
Enabling site projectlamp.
To activate the new configuration, you need to run:
  systemctl reload apache2
ubuntu@ip-172-31-92-76:~$ sudo a2dissite 000-default
Site 000-default disabled.
To activate the new configuration, you need to run:
  systemctl reload apache2
ubuntu@ip-172-31-92-76:~$ sudo apache2ctl configtest
Syntax OK
ubuntu@ip-172-31-92-76:~$ sudo systemctl reload apache2
ubuntu@ip-172-31-92-76:~$
```

i-0918501d0d84267ad (LampStack-Machine)

Now create Index.html file in /var/www/projectlamp location

```
ubuntu@ip-172-31-92-76:/var/www$ cd projectlamp
ubuntu@ip-172-31-92-76:/var/www/projectlamp$ ls
ubuntu@ip-172-31-92-76:/var/www/projectlamp$ vi index.html
ubuntu@ip-172-31-92-76:/var/www/projectlamp$ cd /
ubuntu@ip-172-31-92-76:/$ sudo vim /etc/apache2/mods-enabled/dir.conf
```

Now create new file index.php

`vim /var/www/projectlamp/index.php and enter the following lines`

aws Services Search

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

Now refresh the Public Ip you will see PHP

Instances | EC2 | us-east-1 | EC2 Instance Connect | us-east-1 | PHP 8.1.2-1ubuntu2.14 - phpinfo()

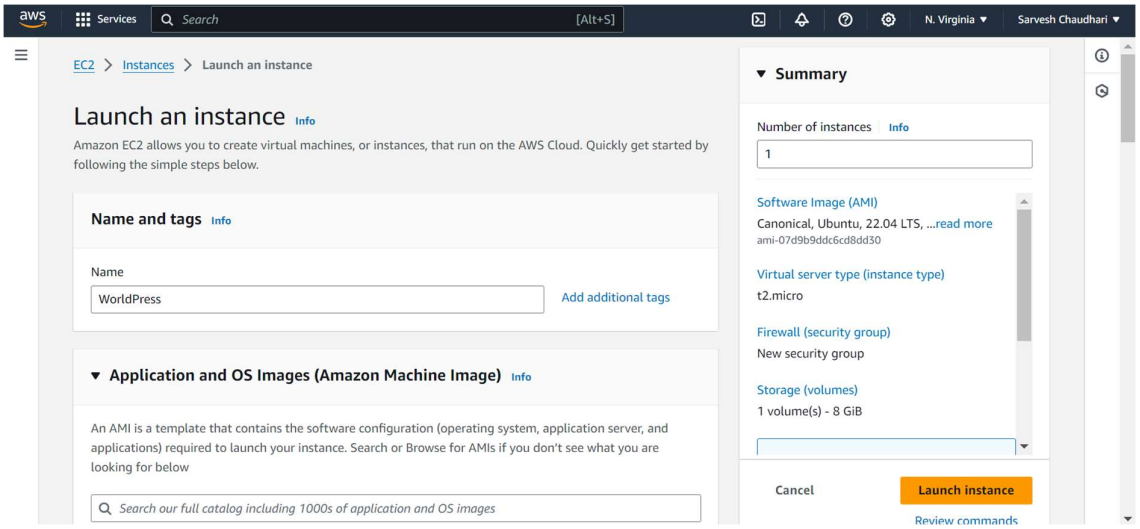
Not secure 44.212.1.220

PHP Version 8.1.2-1ubuntu2.14

System	Linux ip-172-31-92-76 6.2.0-1018-aws #18~22.04.1-Ubuntu SMP Wed Jan 10 22:54:16 UTC 2024 x86_64
Build Date	Aug 18 2023 11:41:11
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

4. Deploy a WordPress website using Lamp Stack on EC2.

1st create instance with name WorldPress



In step 1 install Apache Web-server

```
sudo add-apt-repository ppa:ondrej/apache2
sudo apt update
sudo apt-get install apache2 -y
```

Install PHP

sudo add-apt-repository ppa:ondrej/php

sudo apt install -y php7.2

Apart from this, you need to install the following PHP modules that required for your WordPress application.

sudo apt install -y libapache2-mod-php7.2 php7.2-common php7.2-mbstring php7.2-xmlrpc php7.2-gd php7.2-xml php7.2-mysql php7.2-cli php7.2-zip php7.2-curl php-imagick

To install PHP 7.2, type command below:

sudo apt install -y php7.2Copy

Apart from this, you need to install the following PHP modules that required for your WordPress application.

sudo apt install -y libapache2-mod-php7.2 php7.2-common php7.2-mbstring php7.2-xmlrpc php7.2-gd php7.2-xml php7.2-mysql php7.2-cli php7.2-zip php7.2-curl php-imagickCopy

After installation completed, you will make some changes from php.ini configuration file, run command:

sudo vim /etc/php/7.2/apache2/php.ini

- Install MySQL Server

```
sudo apt-get install mysql-server -y
```

- Install the latest WordPress in Ubuntu 18.04 Server

```
cd /tmp && wget https://wordpress.org/latest.tar.gz
```

Next, extract the compressed files.

```
tar -zxvf latest.tar.gzCopy
```

Next, move the entire extracted files to your root document directory, type the command:

```
sudo mv wordpress /var/www/wordpressCopy
```

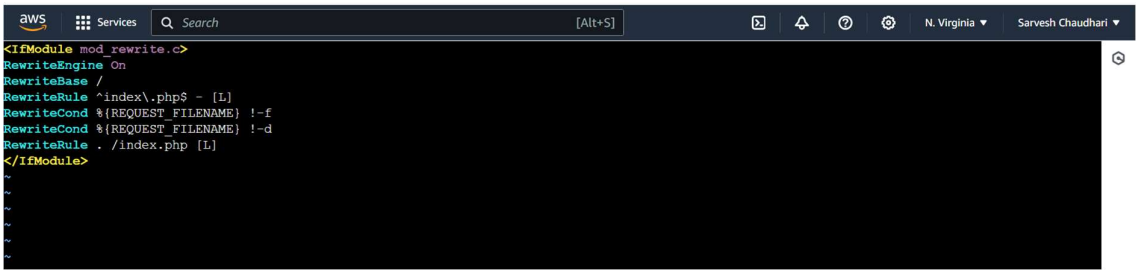
Then open root document directory using:

```
cd /var/www/wordpress/
```

Create an .**htaccess** file, type command:

```
sudo vim .htaccessCopy
```

Add following lines into the .**htacess** file.



Save and close the file.

Next, Rename the **wp-config-sample.php** using the following command:

```
sudo mv wp-config-sample.php wp-config.php
```

To change WordPress ownership, type command:

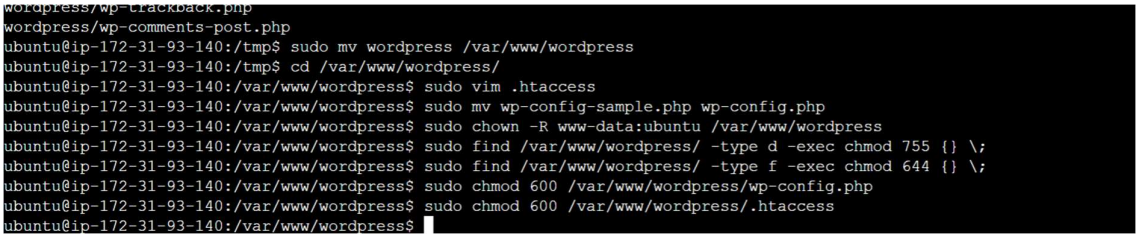
```
sudo chown -R www-data:ubuntu /var/www/wordpressCopy
```

Next, Change the right permission for the particular files and directories, type command:

```
sudo find /var/www/wordpress/ -type d -exec chmod 755 {} \;
sudo find /var/www/wordpress/ -type f -exec chmod 644 {} \;Copy
```

Apart from this, set the following important files to `chmod 600` so that only the owner can fully read and write access to these files.

```
sudo chmod 600 /var/www/wordpress/wp-config.php
sudo chmod 600 /var/www/wordpress/.htaccess
```



- **Setup MySQL Database in WordPress Configuration File**

open the WordPress configuration file, type command:

```
sudo vim /var/www/wordpress/wp-config.phpCopy
```

Therefore, look for the MySQL database section and define the following values, similar to this:

```
// ** MySQL settings - You can get this info from your web host
** //
/** The name of the database for WordPress */
define( 'DB_NAME', 'database_name_here' );

/** MySQL database username */
define( 'DB_USER', 'username_here' );
```

```
/** MySQL database password */
define( 'DB_PASSWORD', 'password_here' );

/** MySQL hostname */
define( 'DB_HOST', 'localhost' );

/** Database Charset to use in creating database tables. */
define( 'DB_CHARSET', 'utf8' );
Copy
```

After modifying the database connection, then save and close the file.

Step 8. Setup WordPress Security Key

WordPress Security Key is a set of random variables that improve encryption of information stored in the user's cookies.

To grab your own `WordPress Security Key`, type following command on your terminal console:

```
curl -s https://api.wordpress.org/secret-key/1.1/salt/
```

Step 9. Creating Apache Virtual Host for WordPress

To add an Apache virtual host for WordPress, Go to the Apache2 virtual configuration directory, type command:

```
cd /etc/apache2/sites-availableCopy
```

Next, Disable the default configuration file, type command:

```
sudo a2dissite 000-default.confCopy
```

Then reload the Apache2 configuration file.

```
sudo systemctl reload apache2Copy
```

Now, Create a new WordPress virtual host configuration file, type command:

```
sudo vim wordpress.confCopy
```

Copy the following virtual host configuration file below and change the domain name and also the path location for root document directory.

```
<VirtualHost *:80>
     ServerAdmin admin@example.com
     DocumentRoot /var/www/wordpress
     ServerName example.com
     ServerAlias www.example.com

     <Directory /var/www/wordpress/>
        Options +FollowSymlinks
        AllowOverride All
        Require all granted
     </Directory>

     ErrorLog ${APACHE_LOG_DIR}/error.log
     CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```


Save and close the file.

Next, Enable the virtual host configuration file to available on public access, type command:

```
sudo a2ensite wordpress.confCopy
```

Also, Enable the Apache rewrite module to enable .**htaccess** file on rewrite modules , type command:

```
sudo a2enmod rewriteCopy
```

Note: The rewrite module can be used to redirect one URL to another URL. In order to provides a rule-based rewriting engine to rewrite requested URLs on the fly.

Next, Verify your WordPress virtual host configuration using:

```
sudo apache2ctl configtestCopy
```

The output similar to this:

```
Syntax OK Copy
```

Therefore, If everything is fine. Reload your Apache2 service to implement the changes, type command:

```
sudo service apache2 restart
```