Deploying Web Services in a Cloud Environment

Submitted for the course:

NETV 379 - Cloud Computing

October 2024

Ryufath Alief Adhyaksa Putera Soepeno (23792097)

Information Systems

Faculty of Engineering & Technology

Sampoerna University

&

Applied Computing

College of Applied Science & Technology

University of Arizona

1. Introduction

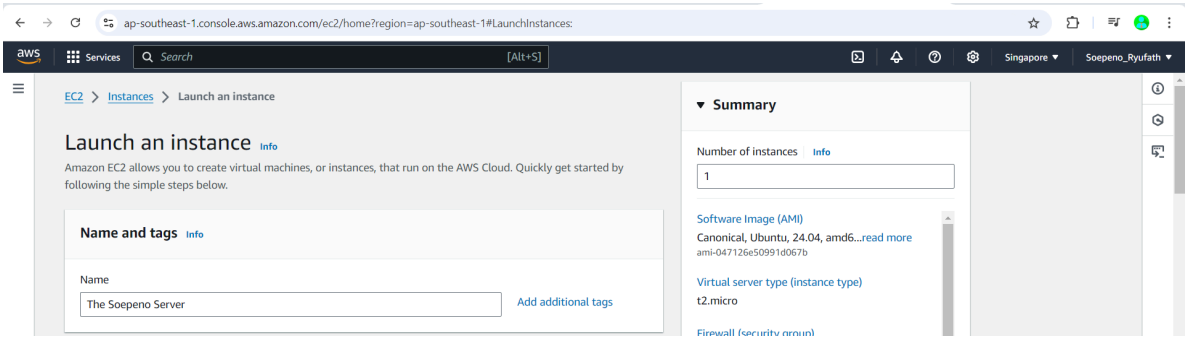
Cloud computing is a technology that provides on-demand access to computing resources over the internet, allowing users to scale their infrastructure and applications without having to manage physical servers. It offers flexibility, cost efficiency, and reliability, making it a key solution for businesses and developers alike. Among the various cloud platforms, Amazon Web Services (AWS) stands out as one of the most preferred providers due to its vast range of services, global infrastructure, security features, and scalability. AWS also provides pay-as-you-go pricing, making it an affordable option for projects of any size.

In this lab case study, I will deploy and create a fully functioning WordPress website using an AWS EC2 instance, an SQL server using the instance, and demonstrate the step-by-step process. The process involves setting up an instance, configuring the necessary environment, and installing WordPress through the instance to serve as the website's content management system. This exercise highlights the simplicity and power of cloud platforms like AWS, where creating scalable web applications becomes more efficient and accessible.

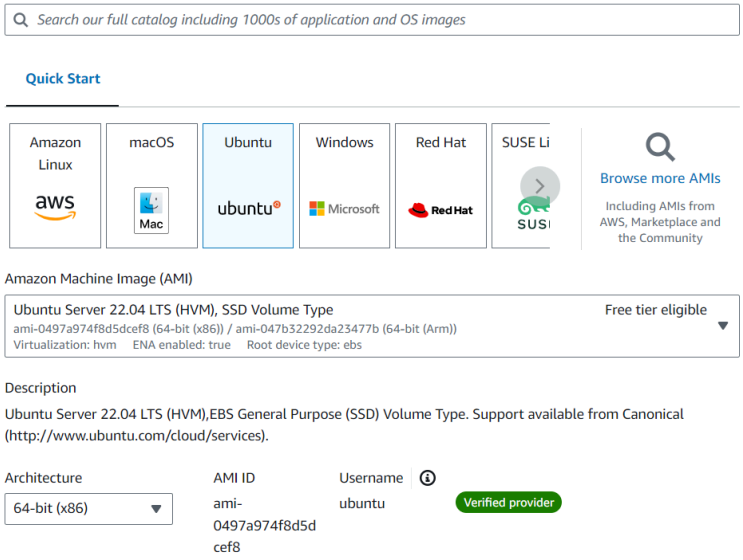
2. Procedure

Part 1: Instances

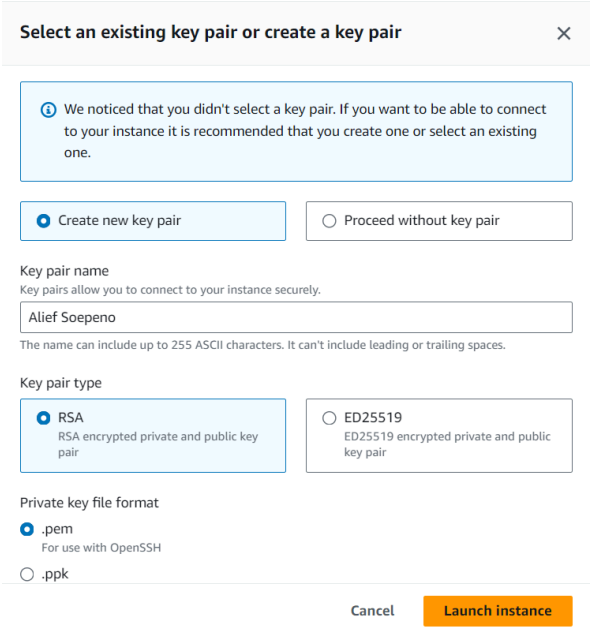
1. First we need to create an instance.



2. We will run the instance with Ubuntu 24.04 LTS as the server

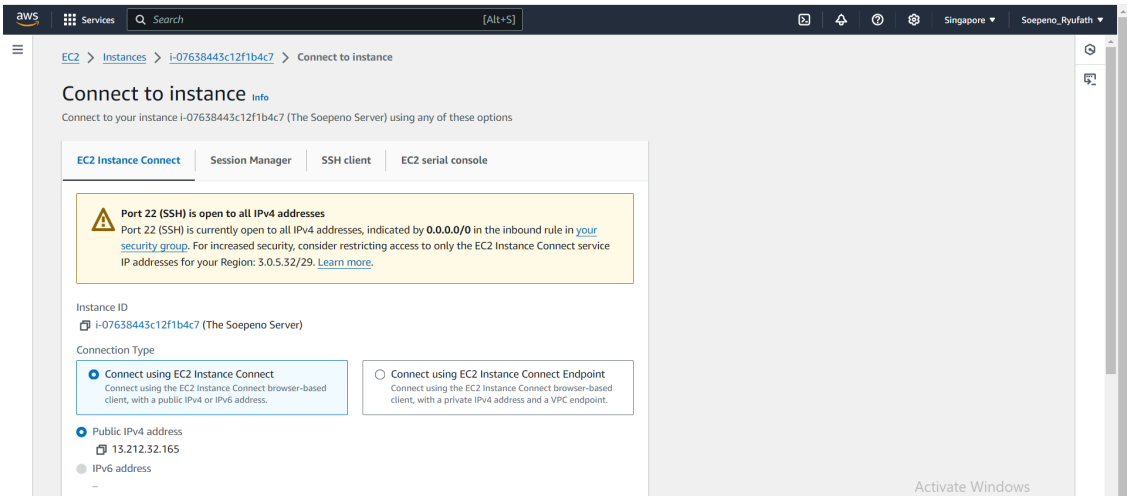


A key pair is needed, we will name it “alief soepeno” and add the other necessary details



3. Then we connect and run the instance.

First, we select the instance we want to connect and run



We will be connected to the linux terminal for the instance. Before we move further, we must first update and see if the linux version is relevant

```
ubuntu@ip-172-31-18-247:~$ sudo apt update
Hit:1 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]
Get:3 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [127 kB]
Get:4 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]
Get:5 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy/universe Translation-en [5652 kB]
Get:6 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 c-n-f Metadata [286 kB]
Get:7 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 Packages [217 kB]
Get:8 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse Translation-en [112 kB]
Get:9 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 c-n-f Metadata [8372 B]
Get:10 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [2066 kB]
Get:11 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [357 kB]
Get:12 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [17.8 kB]
Get:13 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [2504 kB]
Get:14 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted Translation-en [432 kB]
Get:15 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 c-n-f Metadata [616 B]
Get:16 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1128 kB]
Get:17 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe Translation-en [263 kB]
Get:18 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 c-n-f Metadata [26.3 kB]
Get:19 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [43.3 kB]
Get:20 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse Translation-en [10.8 kB]
Get:21 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 c-n-f Metadata [444 B]
Get:22 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main amd64 Packages [91.6 kB]
Get:23 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main Translation-en [11.1 kB]
Get:24 http://ap-southeast-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main amd64 c-n-f Metadata [388 B]
```

```
98% [34 Packages store 0 B] [41 Translation-en 6709 B/7588 B 88%]

Get:42 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 c-n-f Metadata [228 B]
98% [34 Packages store 0 B]
98% [Working]
98% [35 Translation-en store 0 B]
98% [Working]
98% [36 Commands-amd64 store 0 B]
99% [Working]
99% [37 Packages store 0 B]
99% [Working]
99% [38 Translation-en store 0 B]
99% [Working]
99% [39 Commands-amd64 store 0 B]
99% [Working]
99% [40 Packages store 0 B]
100% [Working]
100% [41 Translation-en store 0 B]
100% [Working]
100% [42 Commands-amd64 store 0 B]
100% [Working]

Fetched 33.9 MB in 9s (3945 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
74 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

4. Then we install apache2

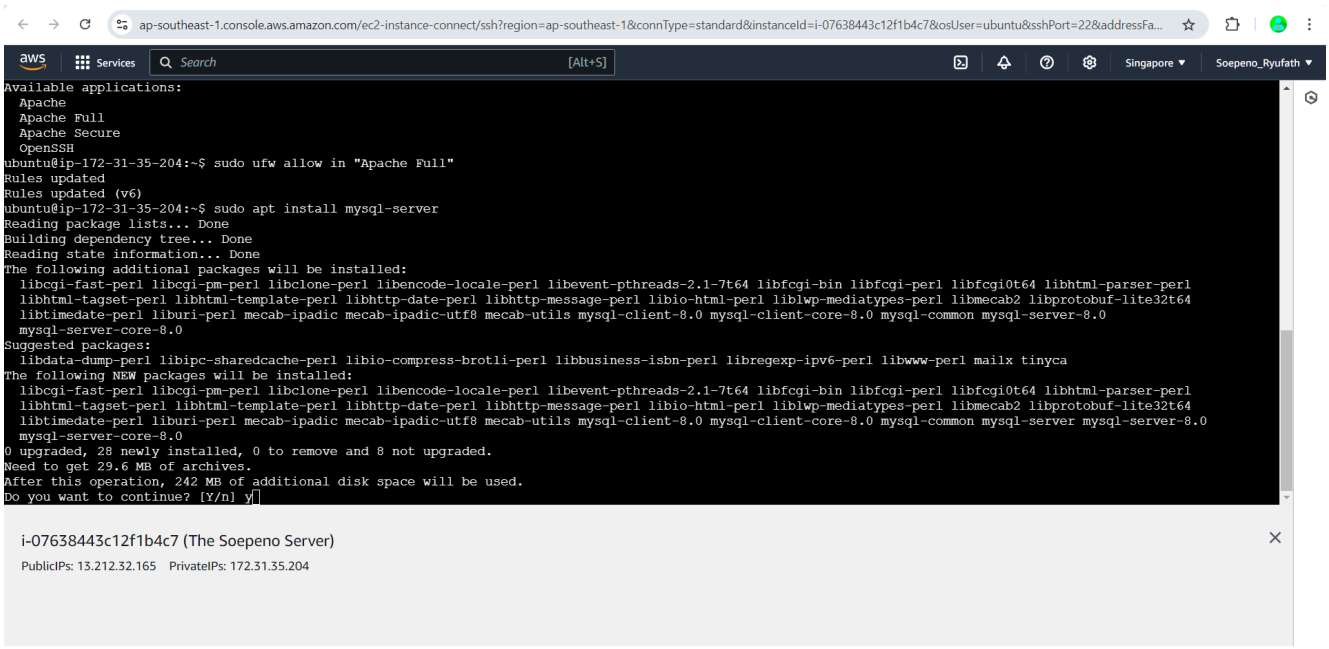
```
aws  Services  Search [Alt+S]  Singapore  Soepeno
99% [39 Commands-amd64 store 0 B]
99% [Working]
99% [40 Packages store 0 B]
100% [Working]
100% [41 Translation-en store 0 B]
100% [Working]
100% [42 Commands-amd64 store 0 B]
100% [Working]

Fetched 33.9 MB in 9s (3945 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
74 packages can be upgraded. Run 'apt list --upgradable' to see them.
ubuntu@ip-172-31-18-247:~$ sudo apt install apache2
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils bzip2 libapr1 libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.3-0 mailcap mime-support ssl-cert
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom www-browser bzip2-doc
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils bzip2 libapr1 libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.3-0 mailcap mime-support ssl-cert
0 upgraded, 13 newly installed, 0 to remove and 74 not upgraded.
Need to get 2141 kB of archives.
After this operation, 8524 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

Then we update the following with the apache rules:

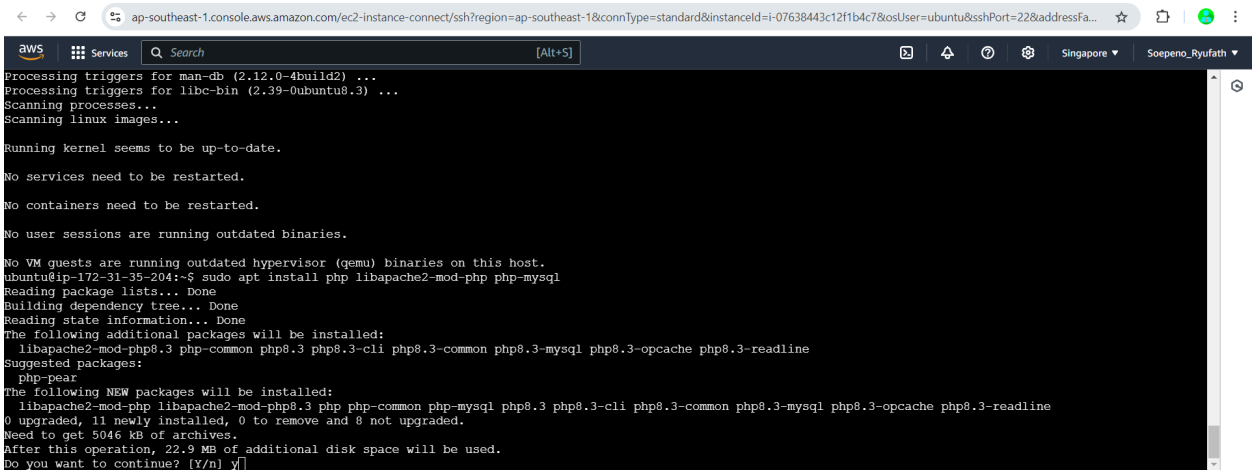
```
ubuntu@ip-172-31-35-204:~$ sudo ufw app list
Available applications:
  Apache
  Apache Full
  Apache Secure
  OpenSSH
ubuntu@ip-172-31-35-204:~$ sudo ufw allow in "Apache Full"
Rules updated
Rules updated (v6)
```

Part 2: Deploy SQL

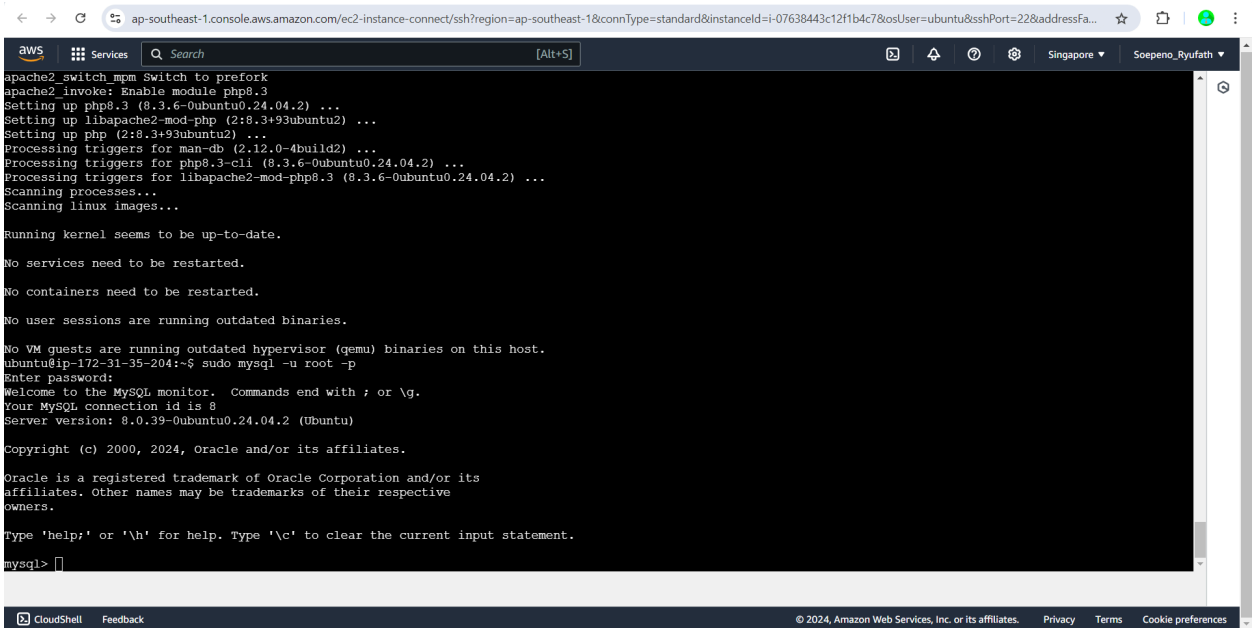
1. First we need to install mysql server with the instance



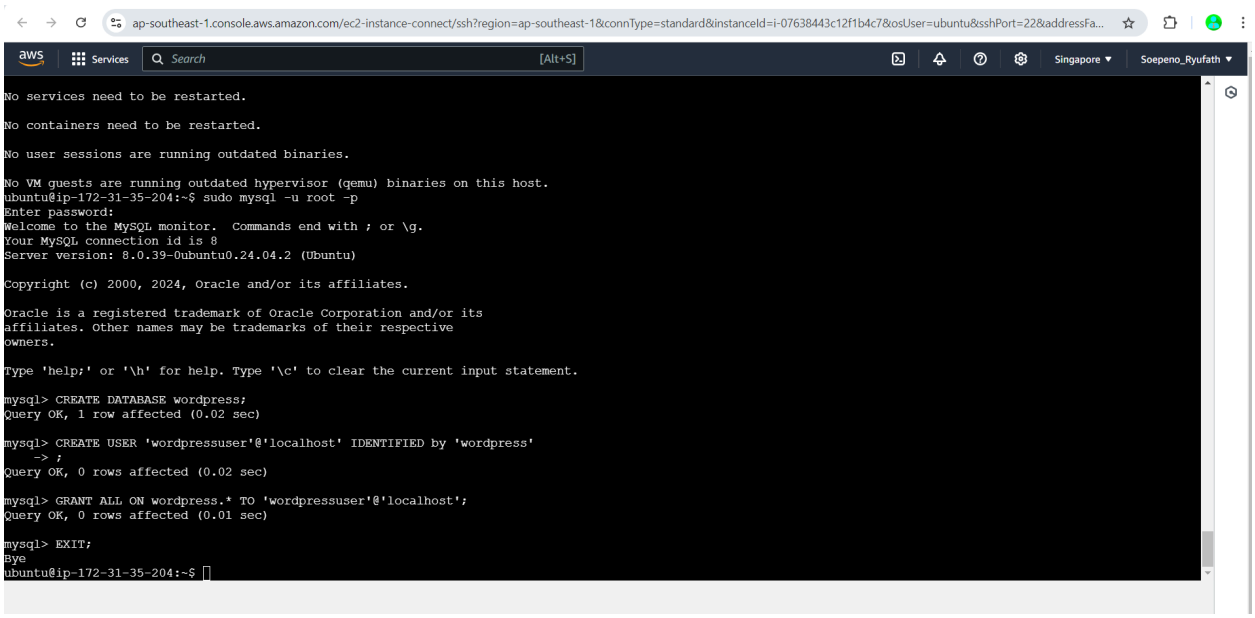
2. Then we install the php host with an apache2 library



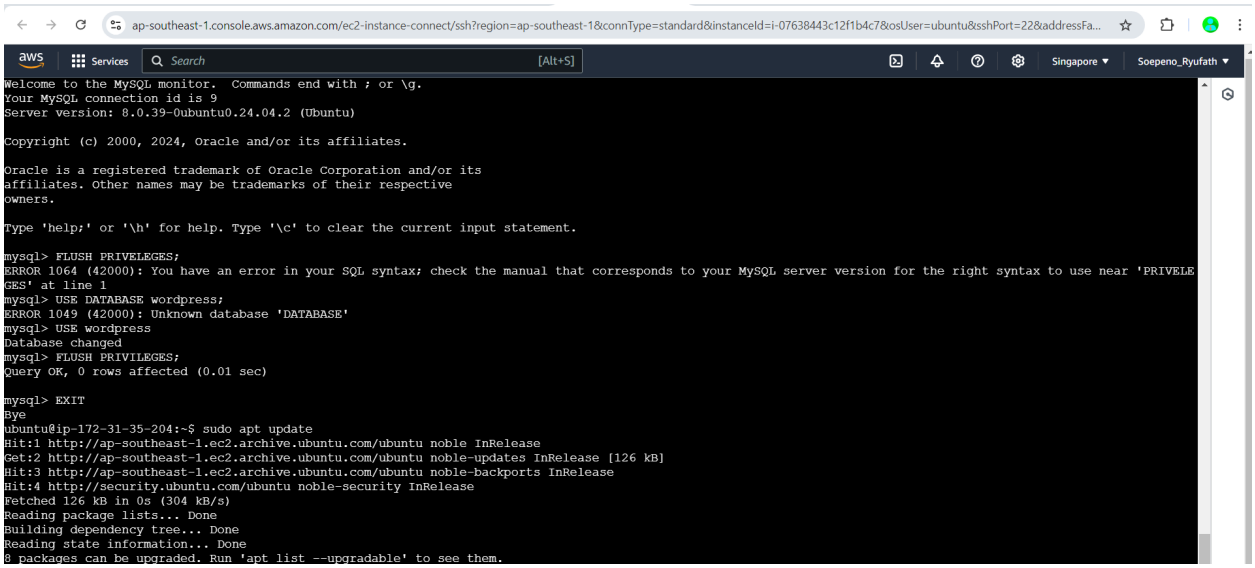
3. Then let's run a mysql server to begin implementing our databases



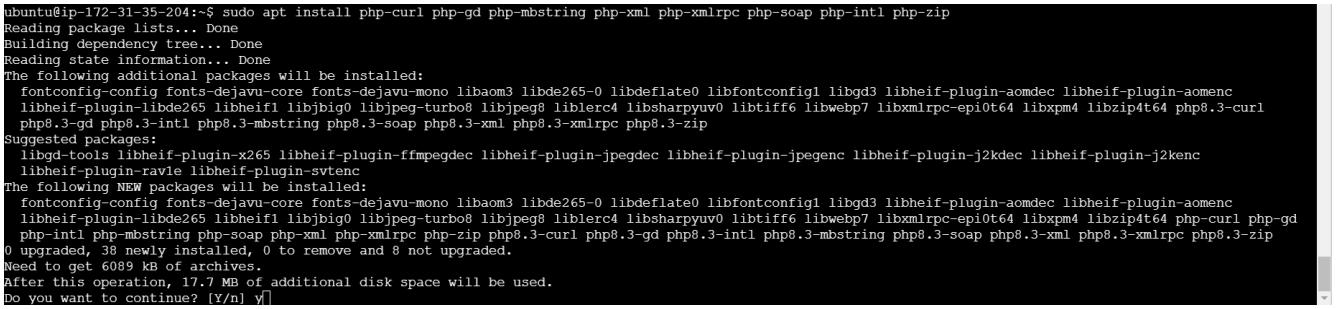
4. Let's create a database named "wordpress" with all the permissions and priveleges required



5. Update the instance again



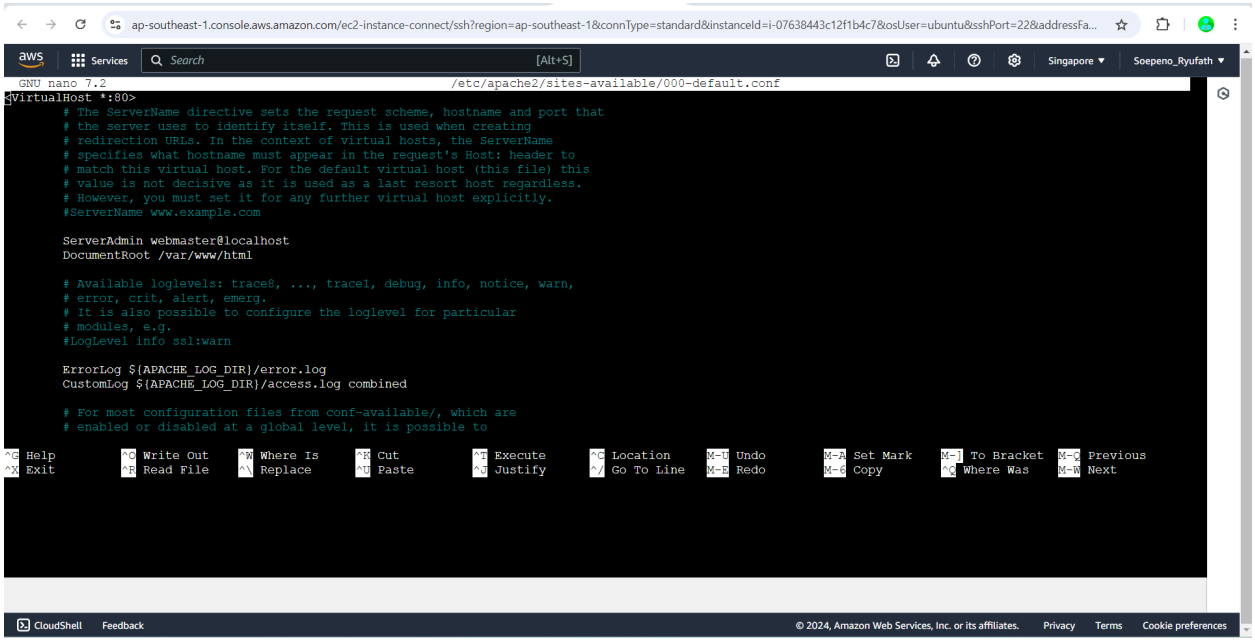
6. Install the php extensions



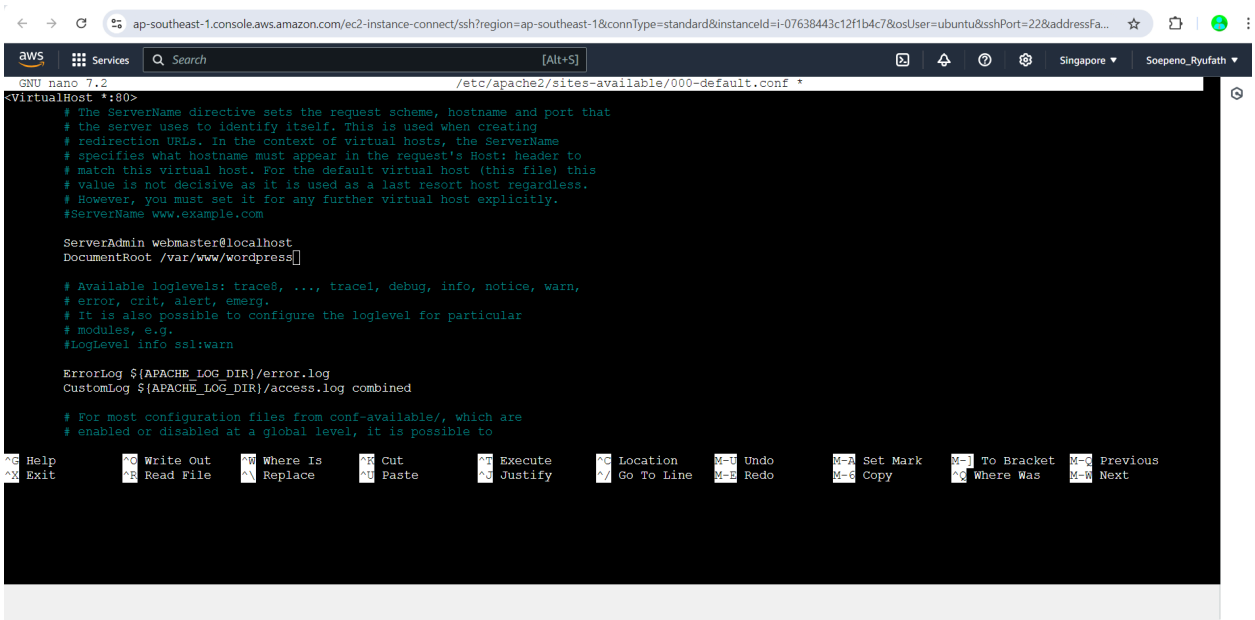
Part 3: Launch Wordpress Website

1. First, we need to run the GPU terminal in linux

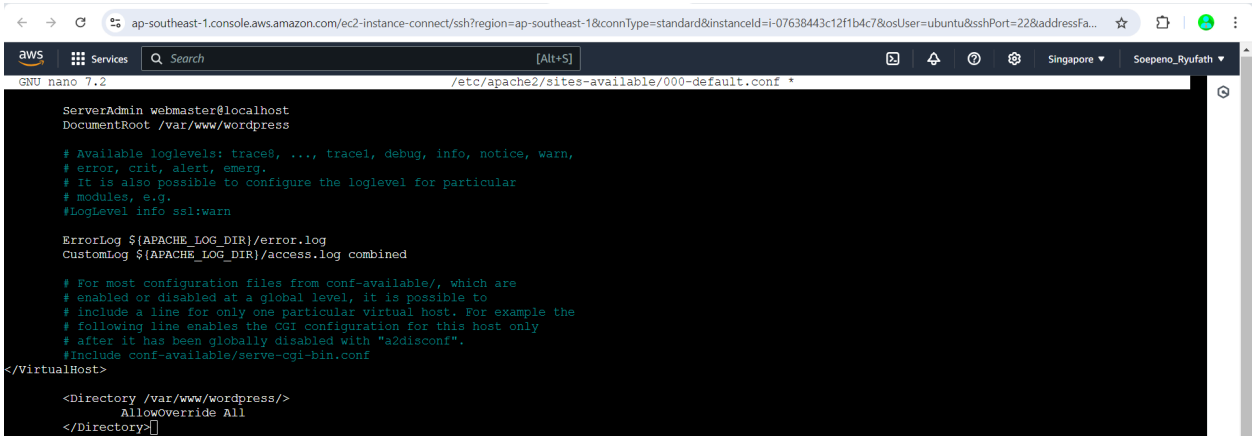
We will see a GPU like this



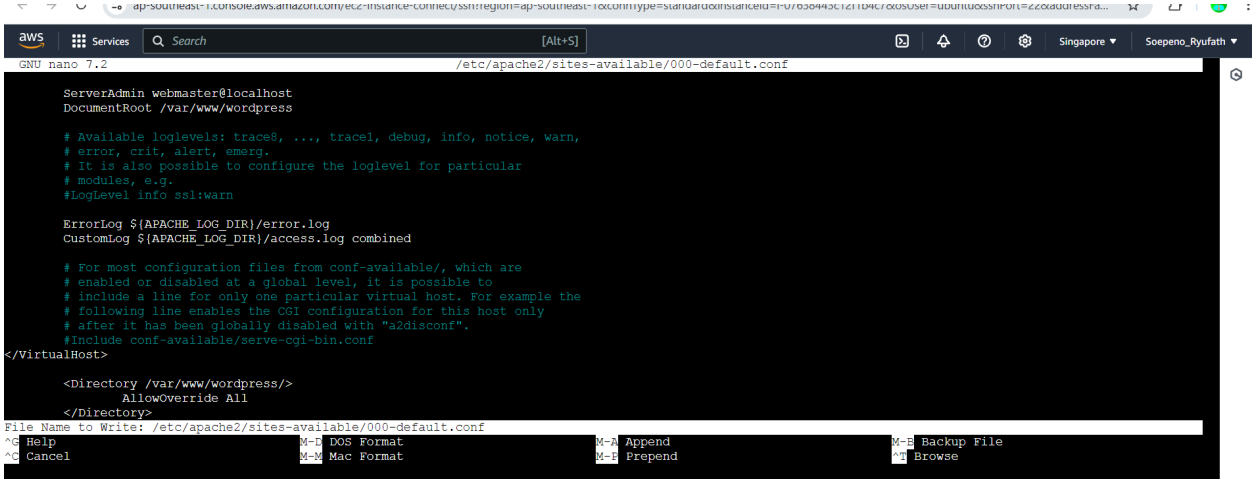
2. Change the document root into our wordpress deployment. To navigate to DocumentRoots, we need to use our page keys on our computers



3. Below **</VirtualHost>,** add a **<directory>** tag to override the instance and read the file



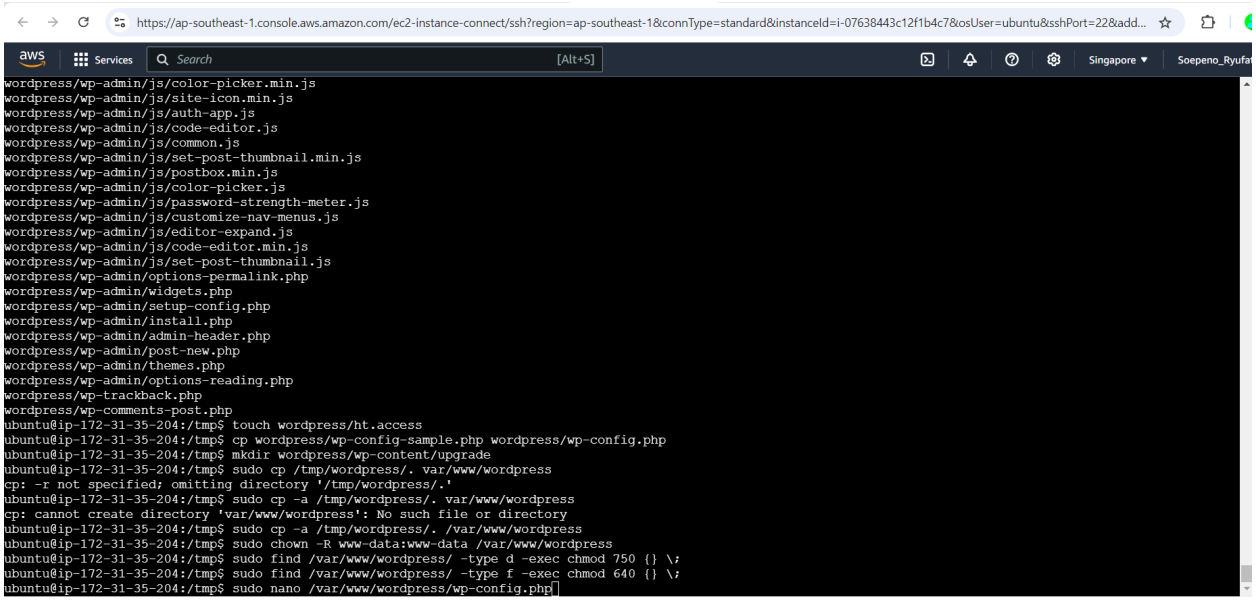
4. Write out the code and exit the GPU



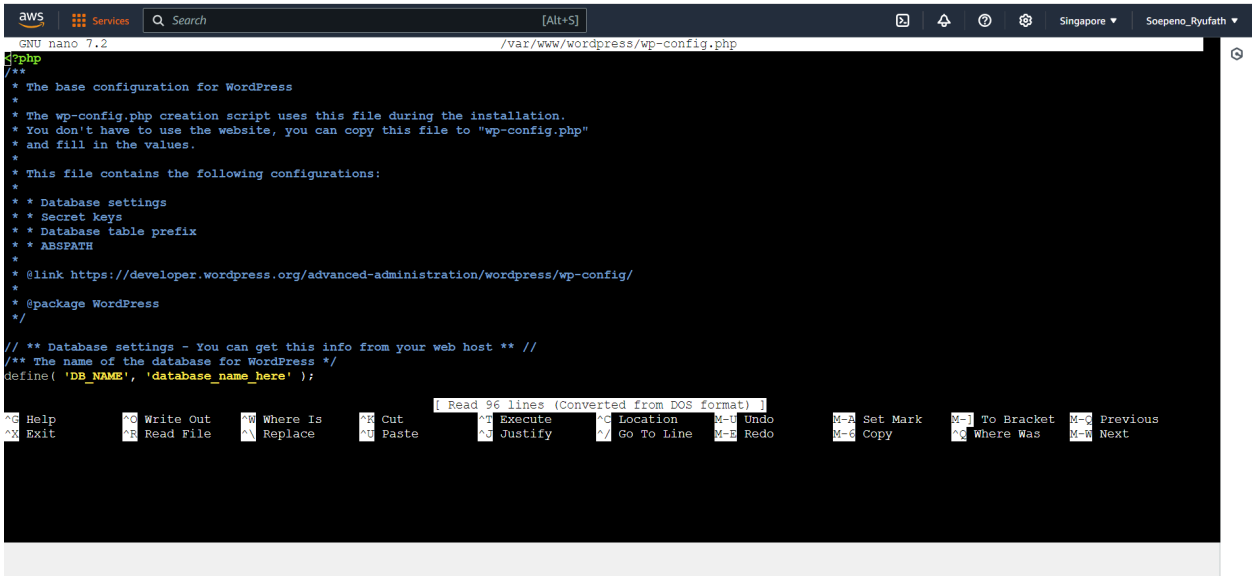
Then write it in the linux terminal

5. Run the wordpress

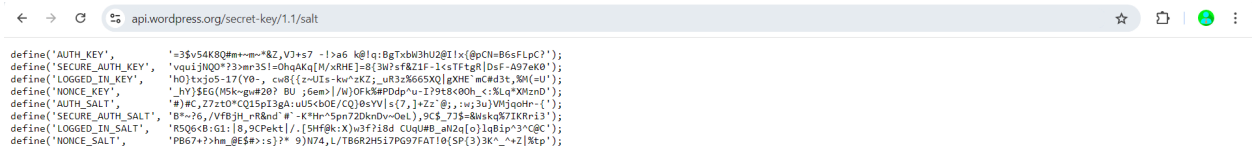
```
ubuntu@ip-172-31-13-222:~$ sudo a2enmod rewrite
Enabling module rewrite.
To activate the new configuration, you need to run:
  systemctl restart apache2
ubuntu@ip-172-31-13-222:~$ sudo apachectl configtest
AH00112: Warning: DocumentRoot [/var/www/wordpress] does not exist
Syntax OK
ubuntu@ip-172-31-13-222:~$ sudo systemctl restart apache2
ubuntu@ip-172-31-13-222:~$ cd /tmp
ubuntu@ip-172-31-13-222:/tmp$ wget https://wordpress.org/latest.tar.gz
--2022-09-24 15:46:21--  https://wordpress.org/latest.tar.gz
Resolving wordpress.org (wordpress.org)... 198.143.164.252
Connecting to wordpress.org (wordpress.org)|198.143.164.252|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 21172479 (20M) [application/octet-stream]
Saving to: 'latest.tar.gz'

latest.tar.gz                     100%[=============================================================>]  20.19M  20.6MB/s    in 1.0s

2022-09-24 15:46:23 (20.6 MB/s) - 'latest.tar.gz' saved [21172479/21172479]

ubuntu@ip-172-31-13-222:/tmp$ tar xzvf latest.tar.gz
```

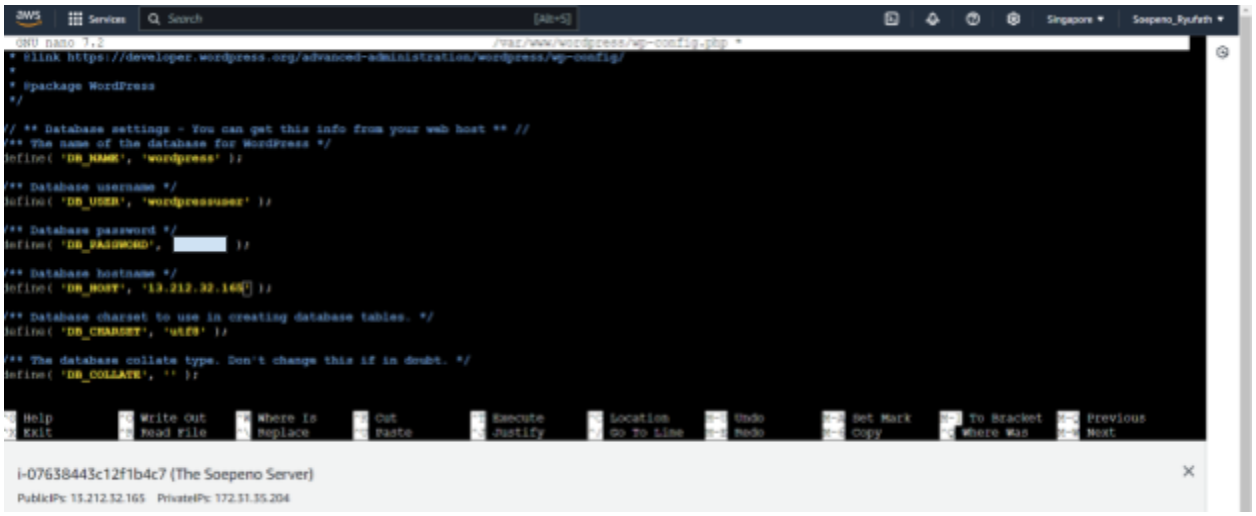


This GPU will be shown



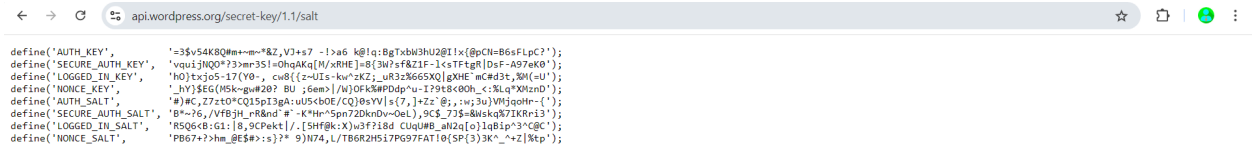
6. Define the database functions



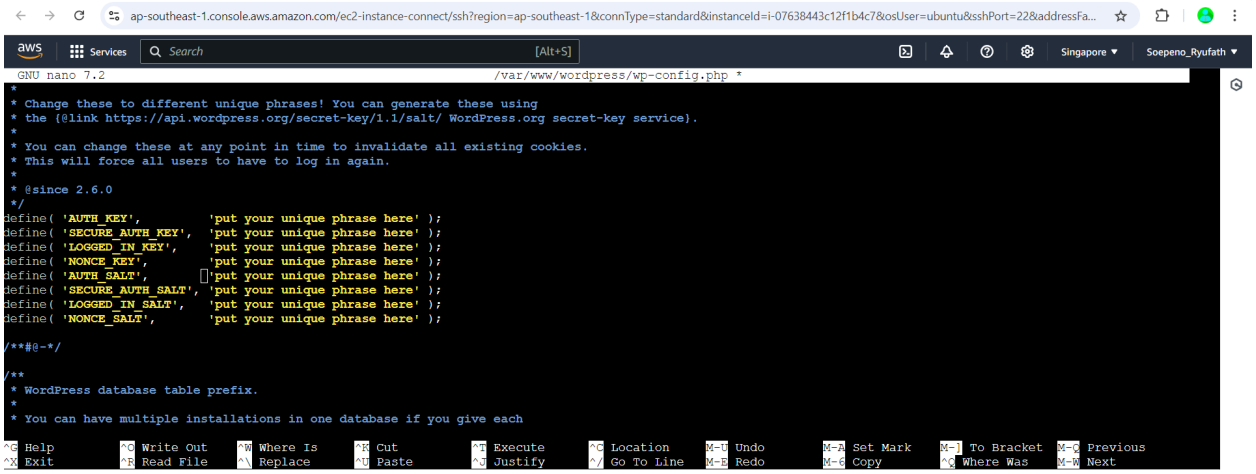


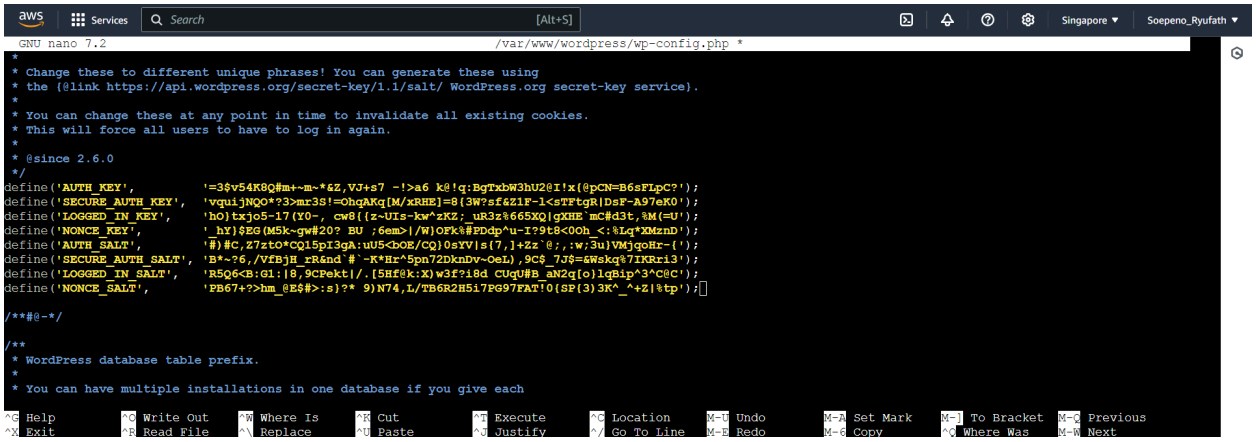
7. Copy the digits of the keys:

Add a new tab and see if the constraints have been added:



Once that, input it to this:

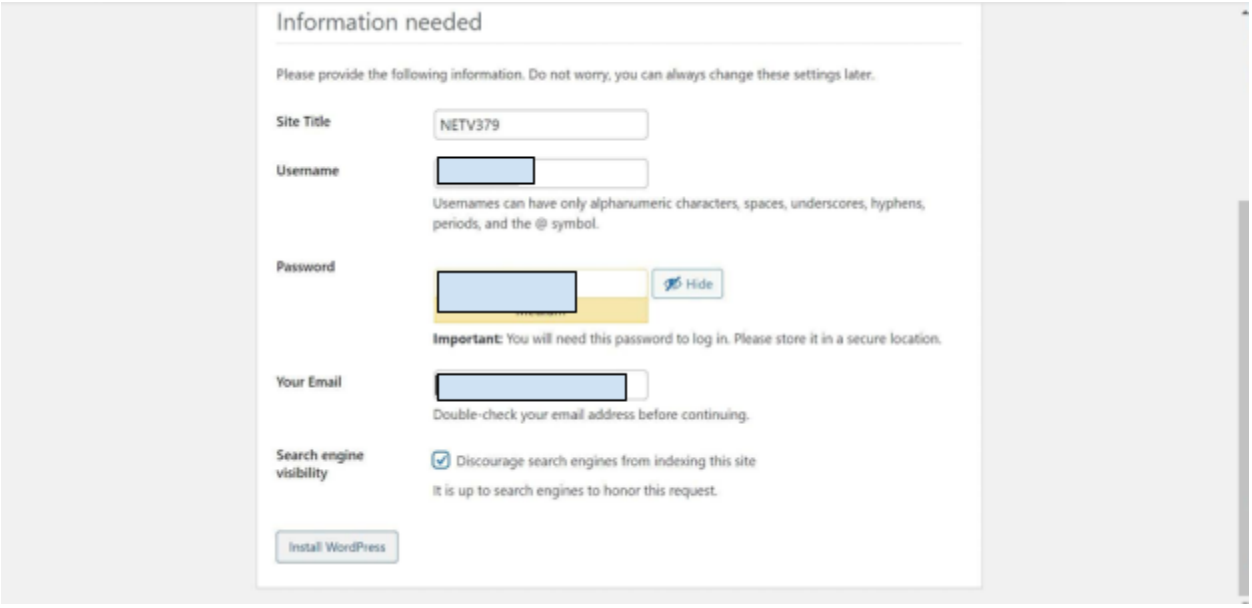




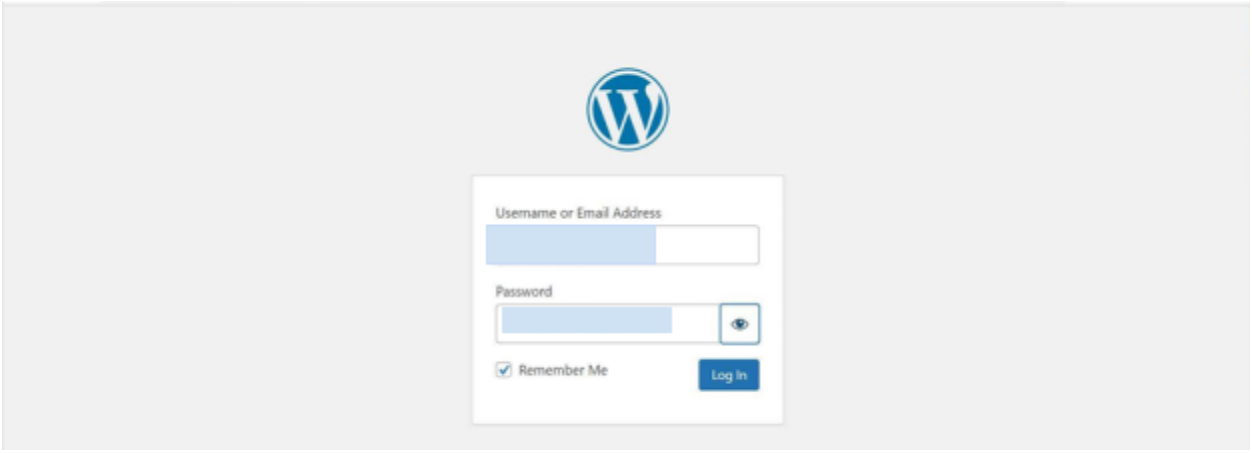
Part 4: Running Wordpress:

1. Input your ip address as your wordpress website adress, in my case, it is 13.212.32.165

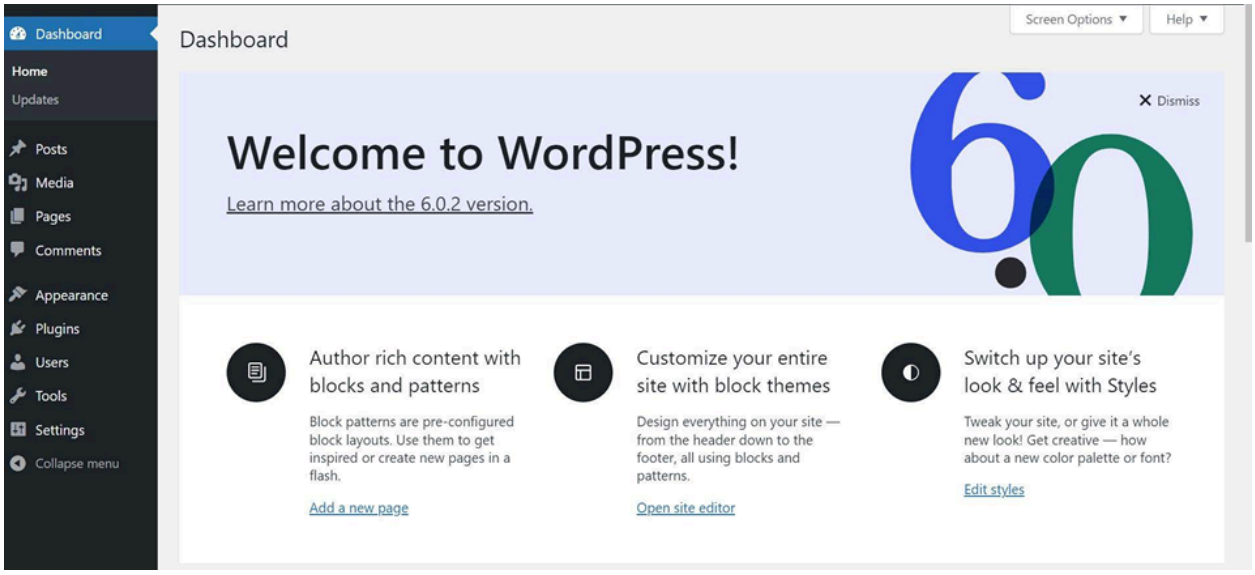
   Once that, you will be lead to this window, where you will need to input your information



2. Once the information is filled, the wordpress site will ask you to log in again

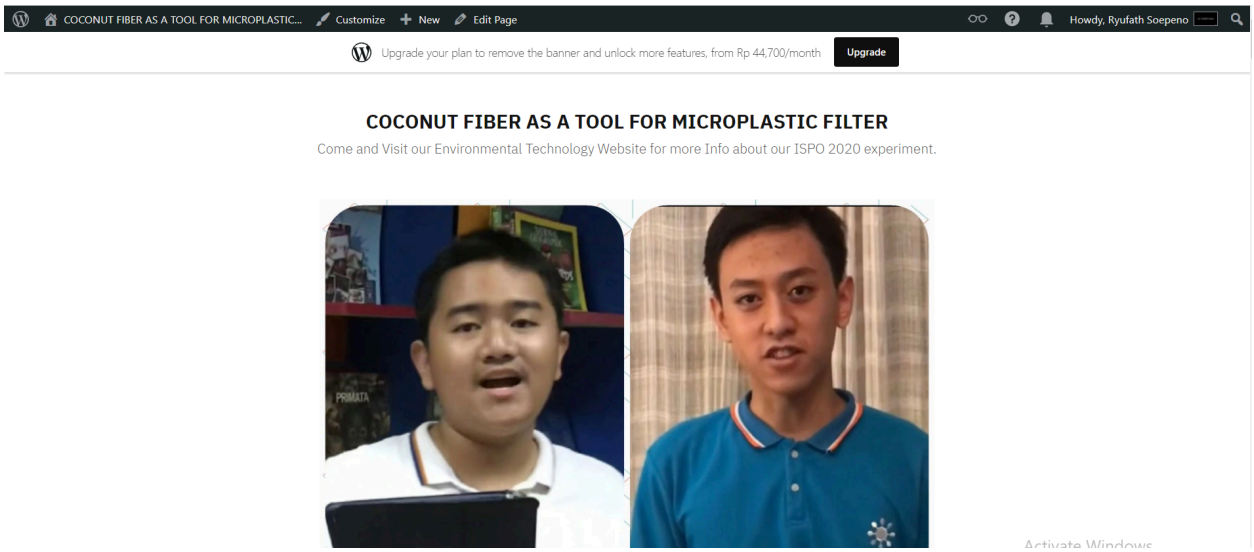


And you will be lead to the wordpress dashboard



3. Output

I wanted to do a little experiment, I actually have already build a wordpress website before for my highschool project olympiad, and so for this output, I exported my wordpress website to my aws dashboard.



Here is the link to my wordpress website:

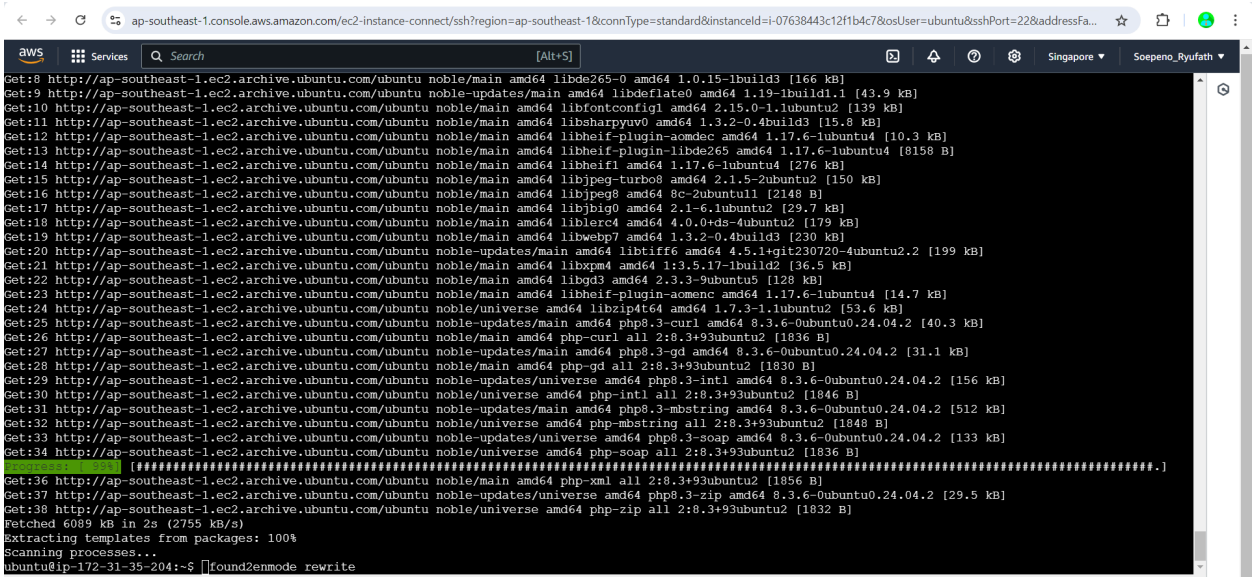
http:/13.212.32.165/2024/10/20/coconutfiberasatoolformicroplasticfilter

However if for some reason, the AWS wasn't able to connect, click this link: https://coconutfiberasatoolformicroplasticfilter.wordpress.com/?_gl=1*4k7r4i*_gcl_au*MTA0MjUwMjEyOS4xNzI4ODg5Njc4

4. Insights & Analysis

During this lab study, I noticed a few errors in my side and my version of the AWS instance that I was working on:

- I noticed that an attempt to input the password, the terminal hides it without showing any word or asterisk, which shows the security of the instance terminal. This was something new for me.
- Sometimes I get errors where the output is shown on the linux prompt and it hinders my input sometimes



However, this was solved by pressing enter, since will help navigate the code out of the output

- When I DB-HOST, localhost does not work, so I changed it to my IP Address instead

5. Conclusion

Overall, this lab case study has equipped me with the basic knowledge and experience in using AWS as an instance to manage my activities, shared and managed under a cloud computing service. And although there were errors or inconsistencies on my side, I managed to solve and mitigate most of them in many different ways.