ICT 171

Assignment 2

Server Setup Documentation

IP Address:
16.16.181.69

Website Link (Domain):
https://villanuevapooltable.store

GitHub Repository:
https://github.com/jerrngs16/ICT171-A2-Documentation

Submitted by: Jerold Ringor | 35441055

TABLE OF CONTENTS

BEFORE SETTING UP:

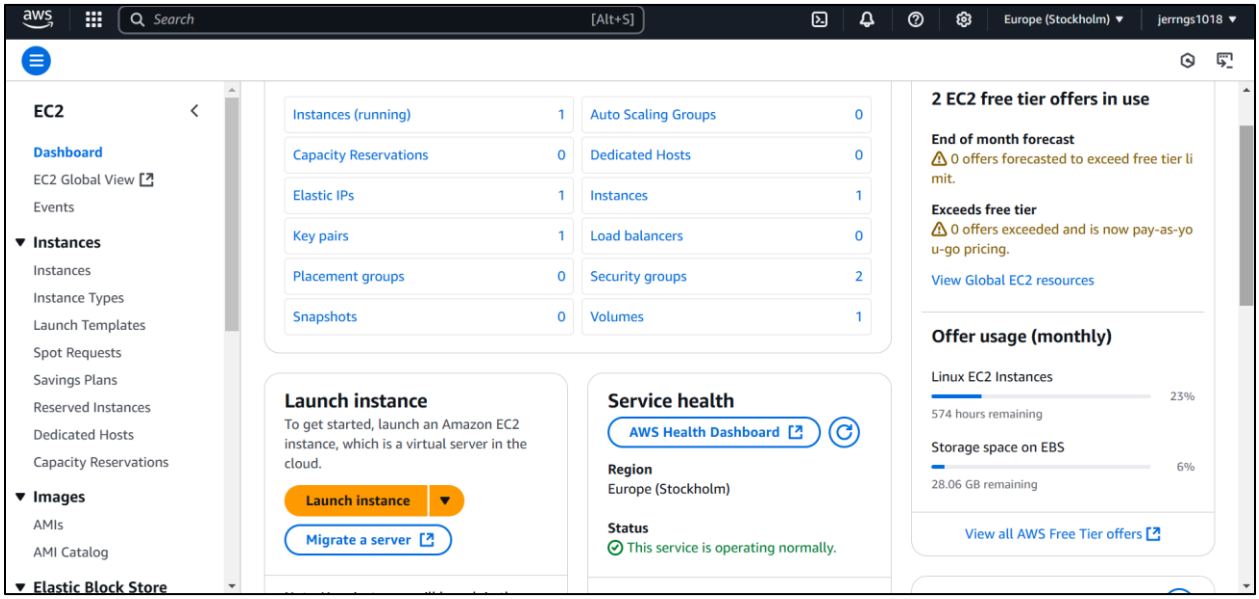
1. Amazon AWS Account
   - Make sure you have an account made to make your instance on

2. Login to AWS (with your account)
   - AWS Management Console (https://aws.amazon.com/console)

Once you have these ready, you can now set up an EC2 instance.

Next Step in the next page.

SETTING UP EC2 INSTANCE:



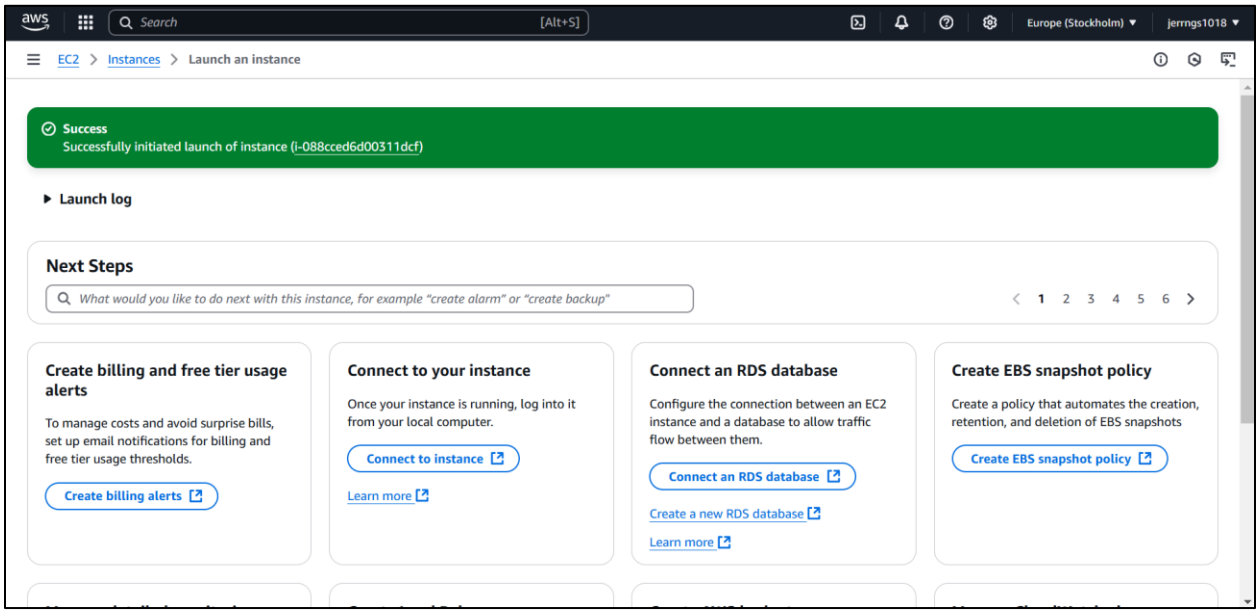
EC2 Console Dashboard

1. In the Console Home Page, select **EC2** from the Services
2. **Launch Instance** (in '*Dashboard*' or '*Instances*' in the sidebar).
3. **Enter a name** for the instance.
4. Under **Application and OS Images, select an AMI** (Amazon Machine Image), for my instance, I used Ubuntu Server 24.04 LTS.
5. **Under Instance type, select an instance type**, for this instance, t3.micro was used.
6. **Under Key pair, create a new key pair.** Enter a name for the key pair and select key pair type as **RSA** and private key file format as **.pem**, then **DOWNLOAD** the key pair (remember the file location).
7. **Under Network settings,** make sure **Create a security group** is selected. Make sure Allow SSH Traffic form is checked, and Anywhere 0.0.0.0/0 is selected for every device to be able to access it.

OPTIONAL or if needed:

- Configure instance details as needed (architecture, storage, etc.) but can also be left as **default** as I did for this server.

After completing all those, **Launch Instance.**



Successful Instance Creation

This is what should appear after creating the instance.
Once you see this, go back to instances (top left; or in the sidebar '*Instances*')

The instance should be found in the Instances list, and it should already be **running**. The status check will show as '**Initializing**'

Refresh the page until the status check shows as '**Checks Passed'.**

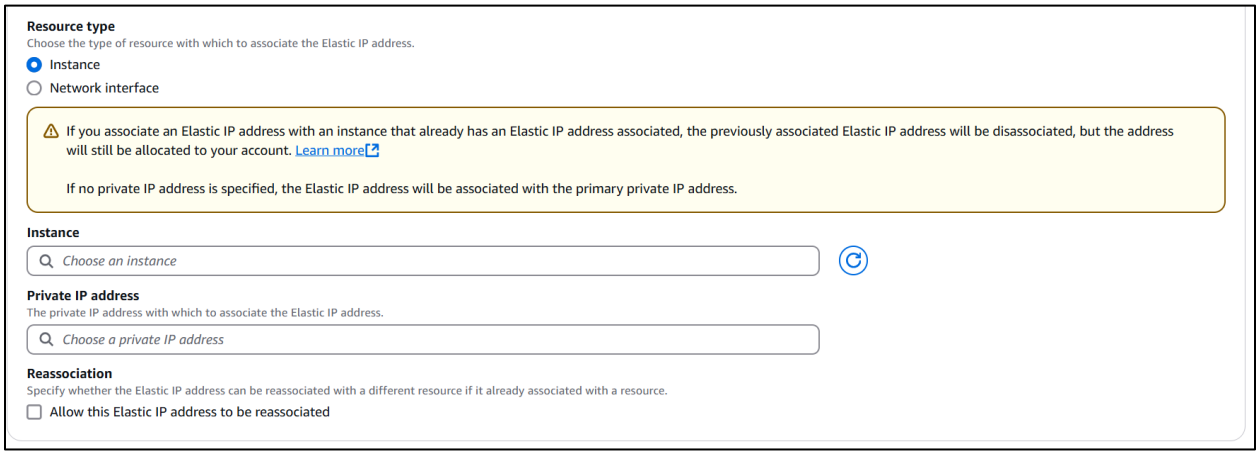
Once you have completed this, you can now associate your instance's IP address to an Elastic IP address.

Next Step in the next page.

ASSOCIATING AN ELASTIC IP ADDRESS:

The IP that you currently have is a **STATIC** IP address, which means that every time the server (or instance) restarts, the IP will change. The **ELASTIC** IP address allows the instance to be linked using an IP that does not change.

1. In the sidebar, navigate to **Elastic IP**.
2. Select your instance by checking the box on the left-hand side of the instances list.
3. Click **Associate Elastic IP**
4. At the bottom of the page, click **Allocate**
5. Once you have created an Elastic IP, right click on the row of the Elastic IP and select **Associate Elastic IP Address**.

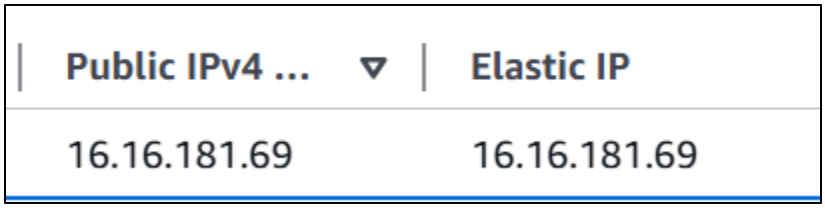


Associating Elastic IP

6. It should take you to the page shown above, **choose your instance**, then **select the private IP address**. Leave **Reassociation unchecked**.

➢ You do not need to change any of the default settings.

After this, you can go back to the '*Instances*' page and check that an elastic IP has been properly associated like so:



Successful Elastic IP Association

Once you have completed this, you can now connect to your instance.

CONNECTING TO THE EC2 INSTANCE:

1. Open **cmd** (terminal)
2. Command to direct to the file location of the key pair (.pem)

   `cd downloads` (or wherever location the key pair is saved)
3. Command to SSH into the instance

   `ssh -i "[yourKeypairName].pem" ubuntu@[your.elastic.ip]`

This will then output:

```
The authenticity of host '51.21.232.73 (51.21.232.73)' can't be
established.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

4. Type in '**yes**' when prompted

After this, you should be connected to the EC2 Instance. **Do not close the terminal yet**.

Once you have completed this, you can now install Apache, MySQL, and PHP.

Next Step in the next page.

INSTALLING ADDITIONAL PACKAGES:

You can install these packages one by one **OR** at once (this is faster).

I will be showing both as a break down for better understanding and clarity of the code.

Installing one at a time:

1. Command to update packages and upgrades installed packages.

```
sudo apt update && sudo apt upgrade -y
```

2. Command to install Apache.

```
sudo apt install apache2 -y
```

3. Command to install a MySQL Server.

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

4. Command to install PHP.

```
sudo apt install php -y
```

5. Command to install Apache PHP.

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

6. Command to install MySQL PHP.

```
sudo apt install php-mysql -y
```

7. Command to install Unzip.

```
sudo apt install unzip -y
```

OR installing all at once (faster):

1. Command to install all packages at once.

```
sudo apt install apache2 mysql-server php libapache2-mod-php php-
mysql unzip -y
```

After doing these, you will see that all of these have been downloaded.

You can check if all of these have been downloaded correctly (or for troubleshooting) and are working by running these commands:

- For Apache:
  ```
  sudo systemctl status apache2
  ```
- For MySQL:
  ```
  sudo systemctl status mysql
  ```
- For PHP:
  ```
  php -v
  ```

Once you have completed this, you can now create a database for WordPress.

Next Step in the next page.

SECURING MYSQL:

1. Command to install Secure MySQL.

```
sudo mysql_secure_installation
```

This will output:

```
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:
```

2. Type in '**y**' when prompted.

This will then output:

```
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:
```

3. Type in whichever option would seem fitting when prompted. (for my server, I selected 1)

4. More prompts will then be asked afterwards, all of which you can simply type in **'y'**

After all of those has been completed, we will need to set a password. You will know that the prompts are finished after you see '*Success. All done!*'

5. Command to open MySQL command-line client.

```
sudo mysql
```

6. Command to set a password

```
ALTER USER 'root'@'localhost' IDENTIFIED WITH
mysql_native_password BY 'YOUR OWN PASSWORD HERE';
```

Make sure your password satisfies the requirements you chose in step 3.

7. Command to reload privilege tables
   ```
   FLUSH PRIVILEGES;
   ```

8. Command to exit MySQL command-line
   ```
   EXIT;
   ```

After doing this, it will now require the password that you have created when opening the SQL command-line, so don't forget it!

Once you have completed this, you can now create the database for WordPress.

Next Step in the next page.

CREATING A DATABASE FOR WORDPRESS:

This step is to set up the database for WordPress before we install it into the server.

1. Command to open the MySQL command-line client.
   ```
   sudo mysql -u root -p
   ```

This will output:
```
Enter password:
```

2. Enter the password that you set from Securing MySQL.
   NOTE: It will appear as if you are not typing anything. This is a security feature to prevent anyone from seeing your password. Type your password SLOWLY and make sure it is correct.

If the password is correct, it will then output:
```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.41-0ubuntu0.24.04.1 (Ubuntu)

Copyright (c) 2000, 2025, 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>` (this is now the MySQL command-line)

3. Command to create a database.
   ```
   CREATE DATABASE wordpress;
   ```

4. Command to create a user.
   ```
   CREATE USER 'wordpressuser'@'localhost' IDENTIFIED BY 'YOUR OWN PASSWORD HERE';
   ```

   NOTE: This password can be different from the password from Securing MySQL. I would recommend that the passwords for here and Securing MySQL are **NOT THE SAME**.

5. Command to grant the Wordpress user to the Wordpress database
   ```
   GRANT ALL PRIVILEGES ON wordpress.* TO
   'wordpressuser'@'localhost';
   ```

6. Command to reload privilege tables
   ```
   FLUSH PRIVILEGES;
   ```

7. Command to exit MySQL command-line
   ```
   EXIT;
   ```

After doing this, a database would have been created for WordPress.

Once you have completed this, you can now install the additional PHP extensions needed.

Next Step in the next page.

INSTALLING ADDITIONAL PHP EXTENSIONS:

1. Command to install all extensions at once.
   ```
   sudo apt install php-curl php-gd php-mbstring php-xml php-xmlrpc
   php-soap php-intl php-zip -y
   ```

2. Command to restart Apache.
   ```
   sudo systemctl restart apache2
   ```

What are these extensions for?

1. php-curl:
   - Enables PHP to interact with external servers using the cURL library.
   - Required for making HTTP requests, such as fetching data from APIs or downloading files.
2. php-gd:
   - Provides image processing capabilities.
   - Required for tasks like resizing, cropping, or generating images (e.g., thumbnails in WordPress).
3. php-mbstring:
   - Adds support for multibyte strings, which are essential for handling non-ASCII characters (e.g., Chinese, Japanese, or Cyrillic).
   - Required for proper encoding and decoding of text in multilingual applications.
4. php-xml:
   - Enables PHP to parse and manipulate XML data.
   - Required for handling XML-based APIs, RSS feeds, or configuration files.
5. php-xmlrpc:
   - Adds support for the XML-RPC protocol, which allows remote procedure calls over HTTP.
   - Required for certain WordPress features, such as remote publishing or pingbacks.
6. php-soap:
   - Enables PHP to interact with SOAP-based web services.
   - Required for applications that rely on SOAP APIs.
7. php-intl:
   - Provides internationalization (i18n) support, including locale-aware string comparison, date formatting, and number formatting.
   - Required for multilingual websites or applications.
8. php-zip:
   - Adds support for ZIP file compression and extraction.
   - Required for tasks like installing plugins, themes, or updates in WordPress.

(PHP: Hypertext Preprocessor, 2025; Wikipedia Contributors, n.d.)

Once you have completed this, you can now download Wordpress into the server.

Next Step in the next page.

DOWNLOADING WORDPRESS:

1. Commands to install Wordpress (run this all at once)

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

(tmp is a temp directory, wget is for downloading from the web, tar extracts files)

2. Command to move the Wordpress files to the Web Directory

```
sudo mv wordpress /var/www/html/
```

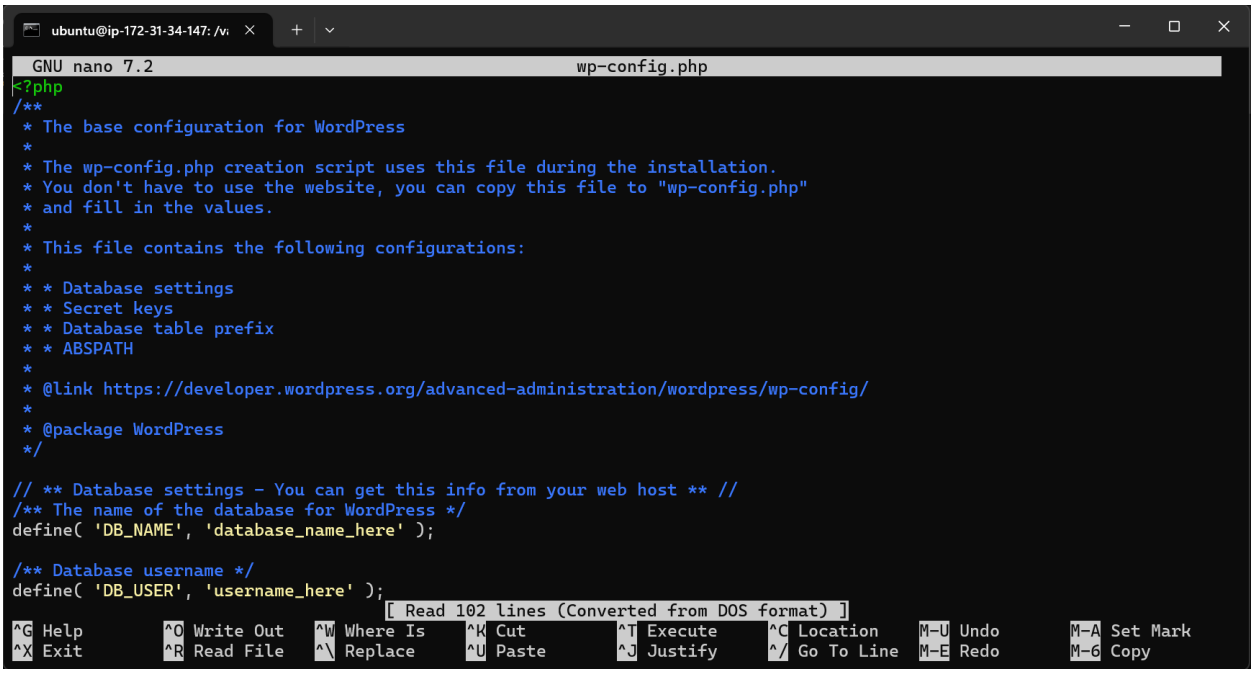
3. Commands to set permissions (run this line by line)

```
sudo chown -R www-data:www-data /var/www/html/wordpress
sudo chmod -R 755 /var/www/html/wordpress
```

4. Commands to configure Wordpress (run this line by line)

```
cd /var/www/html/wordpress
sudo cp wp-config-sample.php wp-config.php
sudo nano wp-config.php
```

After doing that last command, it will bring you to this php file:



In this file, you will see this: (use arrow buttons to navigate in the file)

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

/** Database username */
define( 'DB_USER', 'username_here' );

/** Database password */
define( 'DB_PASSWORD', 'password_here' );

/** Database hostname */
define( 'DB_HOST', 'localhost' );
```

We have to update the contents of those lines using the things we did in the previous steps (Creating a database for Wordpress):

It should look like this:

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

/** Database username */
define( 'DB_USER', 'wordpressuser' );

/** Database password */
define( 'DB_PASSWORD', 'YOUR OWN PASSWORD HERE' );

/** Database hostname */
define( 'DB_HOST', 'localhost' );
```

After you have done this, you can either do:

- CTRL + X (close file), Y (confirm save), Enter (confirm close)
- CTRL + O (save file), Enter (confirm save), CTRL + X (close file)

Either of them will save the changes to wp-config.php, then close the file.

Once you have completed this, you can now configure Apache for Wordpress.

Next Step in the next page.

CONFIGURING APACHE FOR WORDPRESS:

1. Command to create the virtual host file.
   `sudo nano /etc/apache2/sites-available/wordpress.conf`

After doing that command, it will open a blank file.

2. Add this code inside that file.

```
<VirtualHost *:80>
    DocumentRoot /var/www/html/wordpress
    ServerName yourdomain.com
    ServerAlias www.yourdomain.com
    <Directory /var/www/html/wordpress>
        AllowOverride All
        Require all granted
    </Directory>
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

It should look like this after adding the code:

Save and close the file
(By either:

- CTRL + X (close file), Y (confirm save), Enter (confirm close)
- CTRL + O (save file), Enter (confirm save), CTRL + X (close file)

)

3. Command to enable the site.
   `sudo a2ensite wordpress.conf`

4. Command to activate new configuration.
   `sudo systemctl reload apache2`

5. Command to rewrite module.
   `sudo a2enmod rewrite`

6. Command to activate new configuration.
   `sudo systemctl restart apache2`

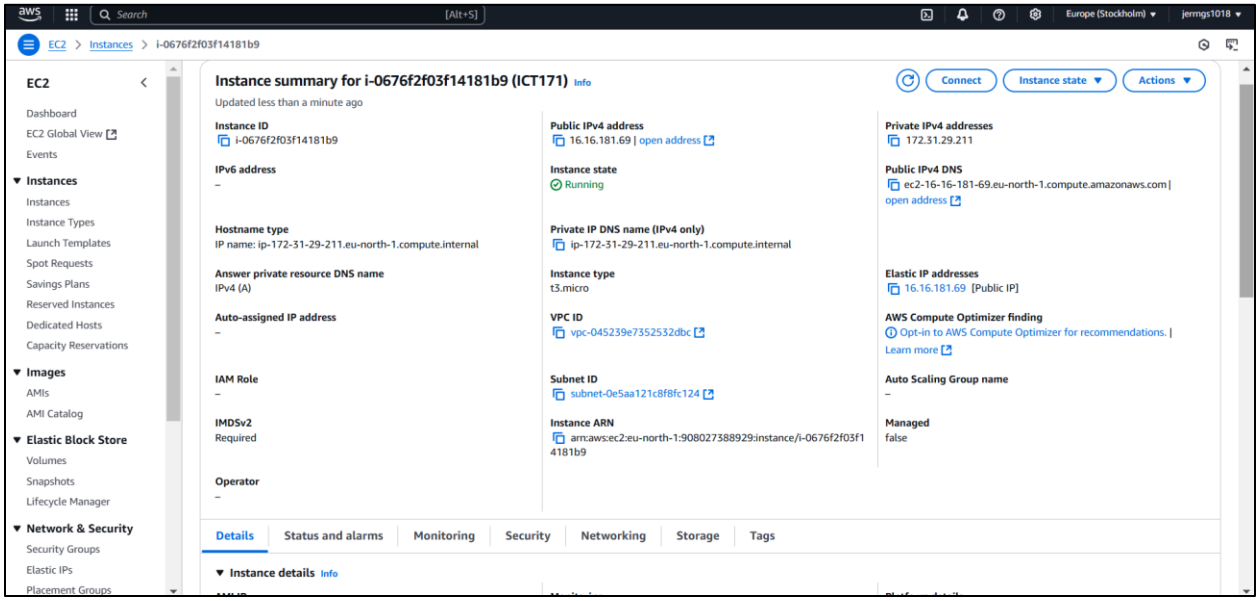
Once you have completed this, you can now add more inbound rules in EC2.

Next Step in the next page.

ADDING INBOUND RULES FOR HTTP AND HTTPS

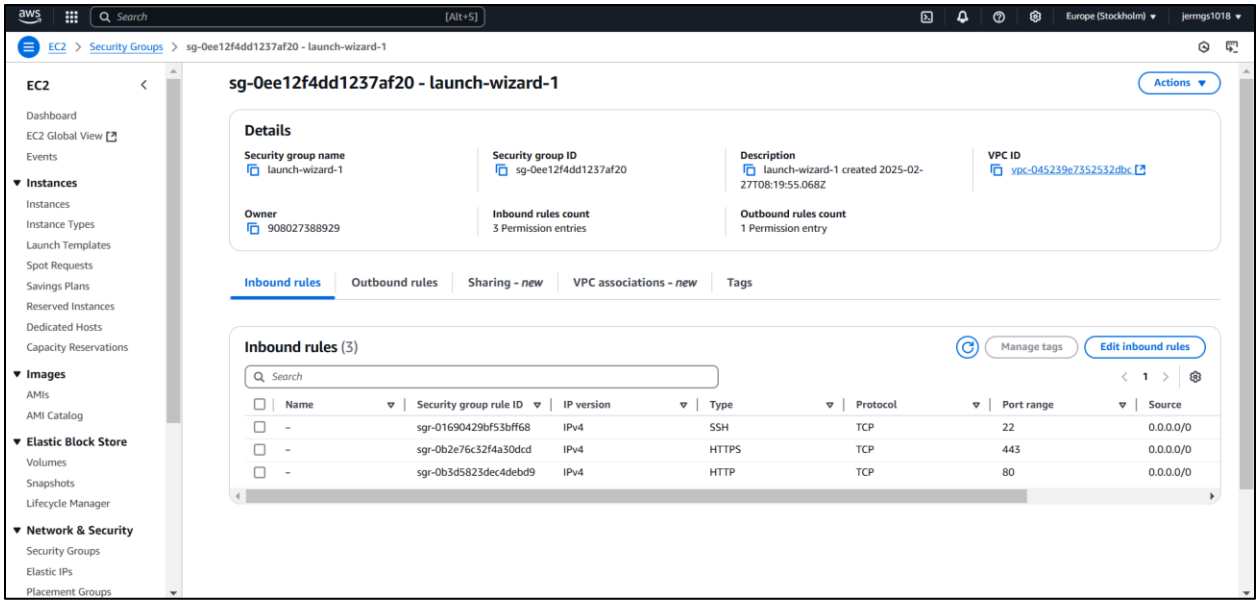
1. In the Instances Page (Amazon EC2), click on the instance ID.

Doing that will bring you to this page:



2. Click on '*Security*' (scroll down if not visible) and click on '*Security Groups*'

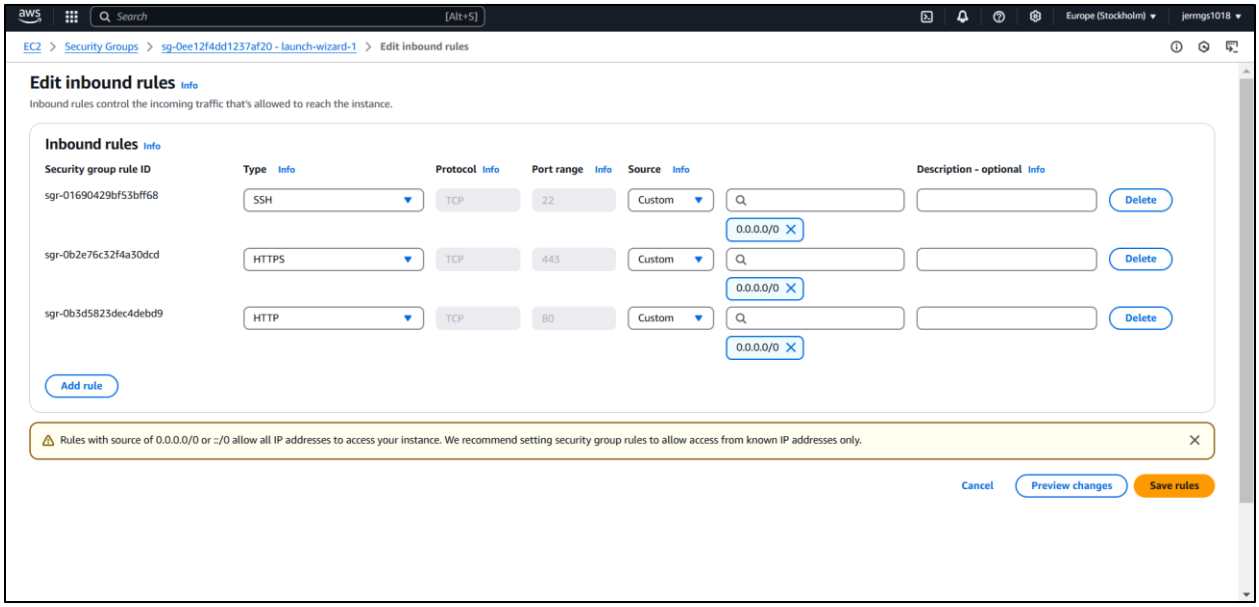
Doing that will then bring you to this page:



(I already have 3 inbound rules in this image which I will show how to add in the next page)

3. Click on '*Edit Inbound Rules*'. Then click on '*Add Rules*'. Make sure you add HTTP (port 80) and HTTPS (port 443).

It should look like this:



4. Don't forget to '*Save Rules*'

After doing all these, you should have three inbound rules.

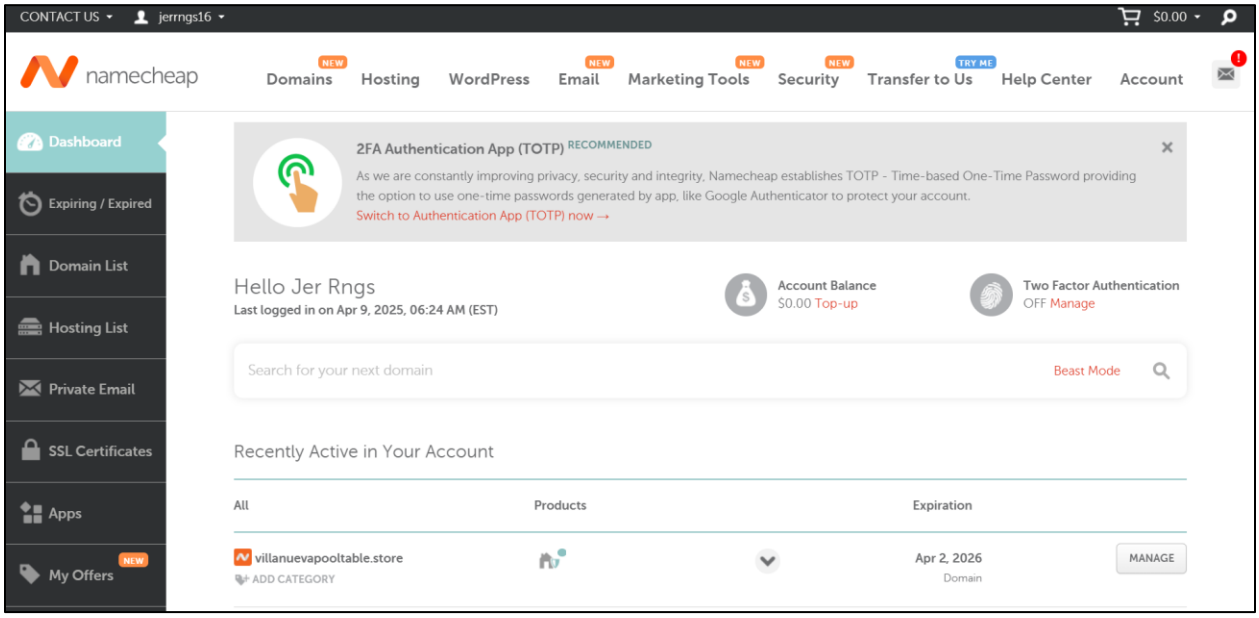
<u>Once you have completed this, you can now configure your DNS.</u>

Next Step in the next page.

CONFIGURING DNS (NAMECHEAP):

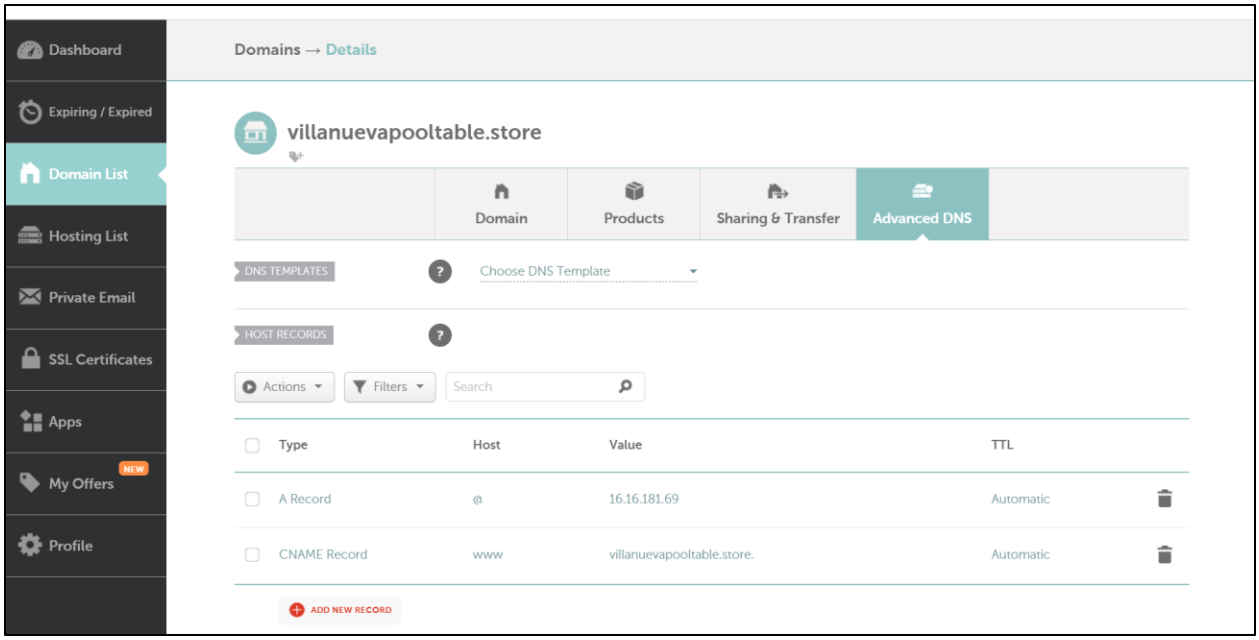
1. Before proceeding, you must first have a domain from Namecheap. I bought my domain here (https://www.namecheap.com/domains/domain-name-search/)
2. After buying the domain, navigate to the Namecheap Dashboard (clicking on account name on top left).

Doing that will bring you to this page:



3. Click on '*Manage*' beside your domain name and navigate to '*Advanced DNS*'.

Doing that will then bring you to this page:



4. Change or delete (preferrably this) the records and add two records:

- A Record
  ⇨ Host: @, Value: IP Address, TTL: Automatic
- CNAME Record
  ⇨ Host: www, Value: your.domain, TTL: Automatic

5. Don't forget to save all changes. DNS Changes may take up to 24-48 hours but does not necessarily take that long. For me, it was only a few minutes.

You can see if your DNS is propagating across servers using this (https://dnschecker.org/).

Once you have completed this, you can now setup HTTPS.

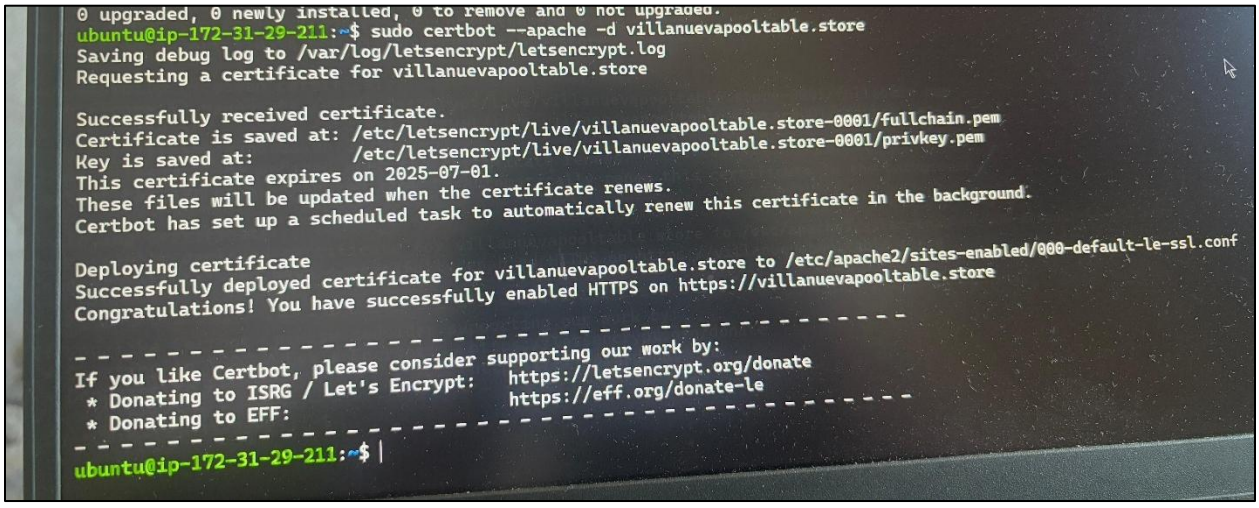
Next Step in the next page.

SETTING UP HTTPS:

1. Command to install Certbot
   `sudo apt install certbot python3-certbot-apache -y`

2. Command to obtain an SSL Certificate
   `sudo certbot --apache -d domain.name`

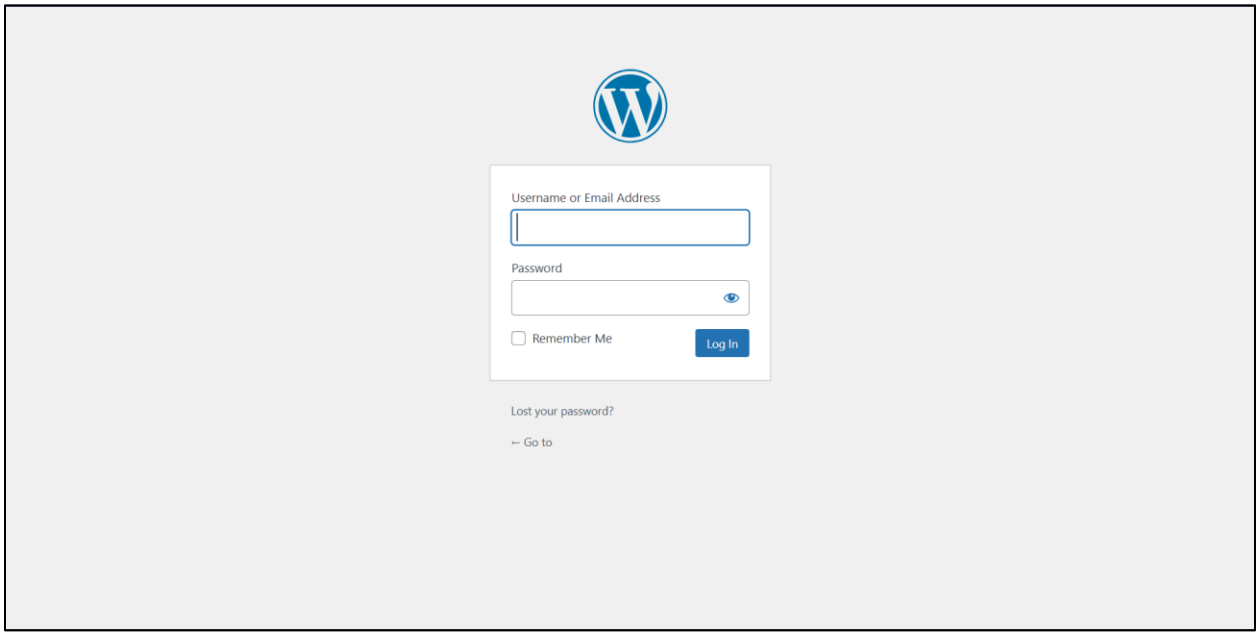
It should then show this with your domain



WORDPRESS SETUP

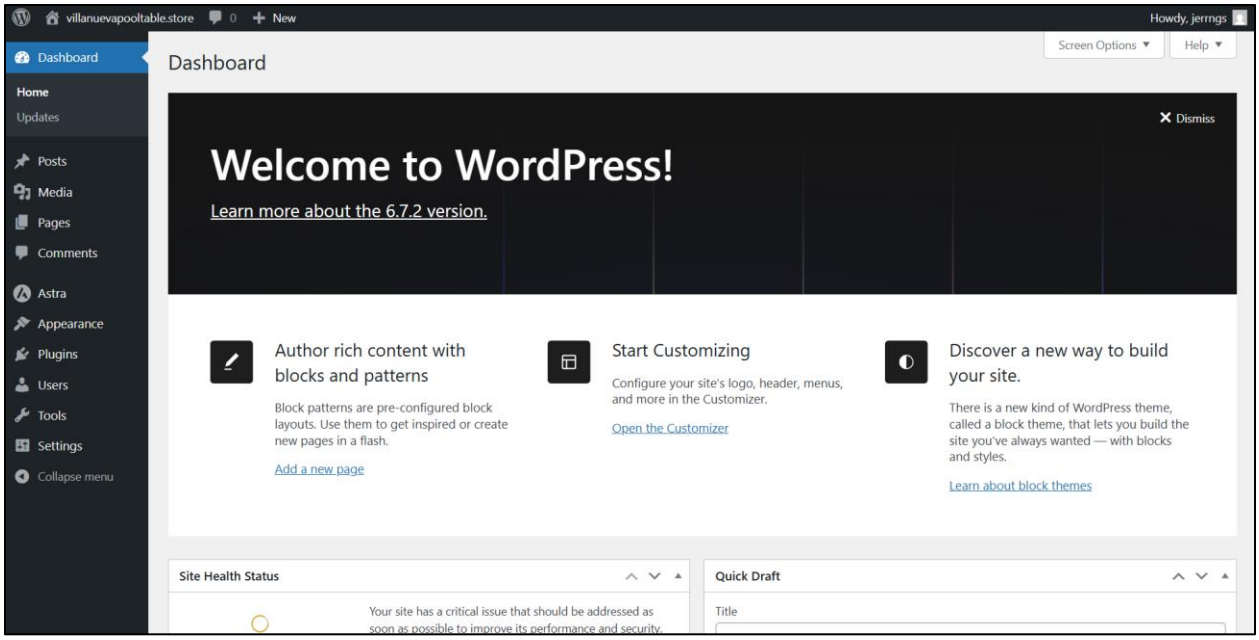
1. Go to browser and enter '*https://yourdomain.com/wp-admin*'

You should see this:



Log-in using the credentials you have made in the previous steps

You would then see this:



From there, you can edit the page design and functionality.

REFERENCES:

PHP: Hypertext Preprocessor. (2025). *PHP: Alphabetical - Manual*. Php.net. https://www.php.net/manual/en/extensions.alphabetical.php

Wikipedia Contributors. (n.d.). *List of PHP Extensions*. Wikipedia; Wikimedia Foundation.