I have created an Ubuntu environment. On this Ubuntu server, we will install wordpress, osticket, and owncloud. Along with that, we will point a domain to the server and enable SSL. So now, we will proceed step by step, starting with wordpress.

**Install WordPress on Ubuntu 22.04**

Since Ubuntu uses “apt” (advanced package tool) for package management, we need to update our repository and upgrade our packages before we get started.

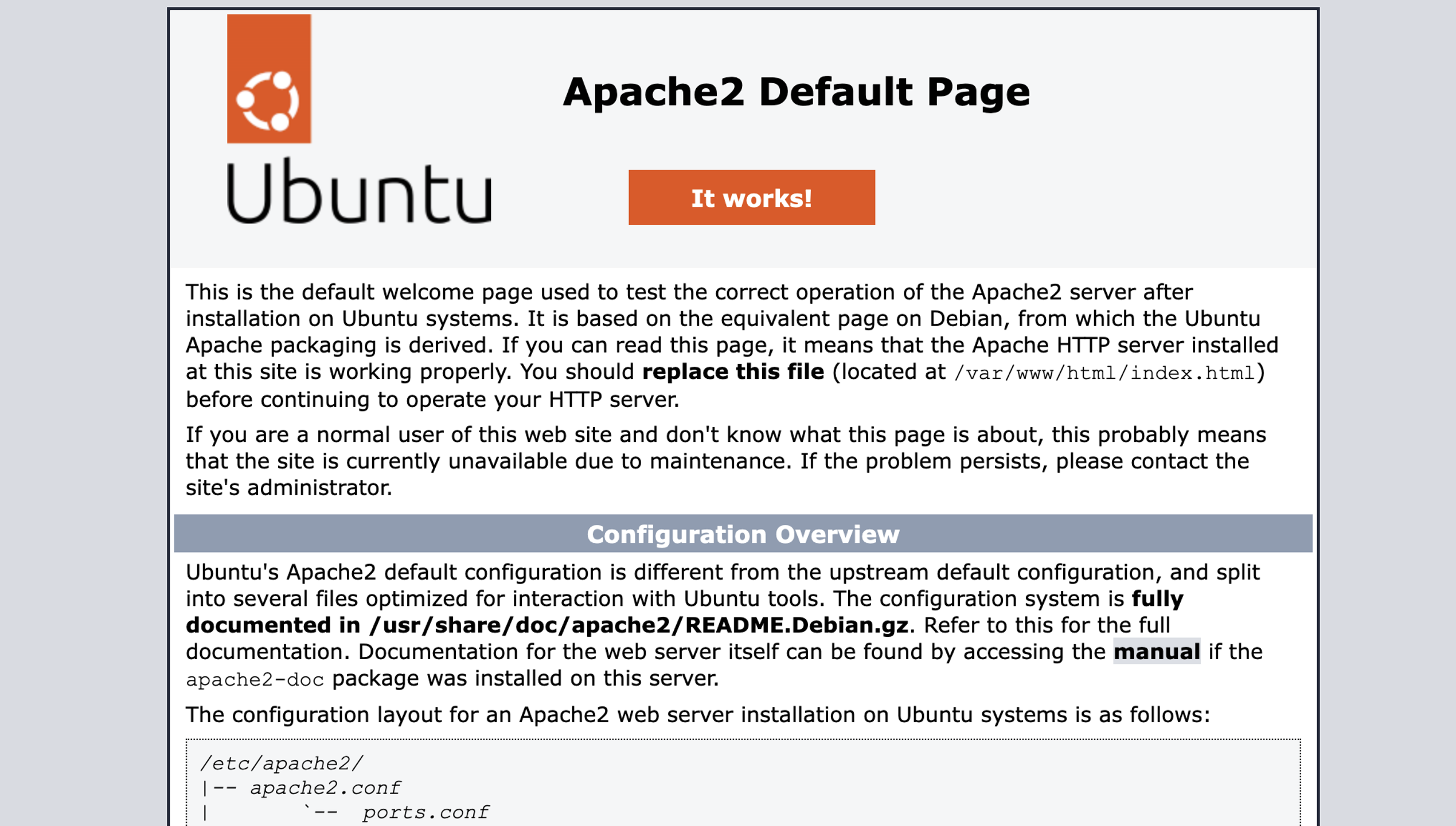
-> sudo –I 🡪for log-in from root to user

-> apt update && sudo apt upgrade –y

Next, you’ll install Apache server and the dependencies you’re going to need.

-> apt install apache2 ghostscript libapache2-mod-php mysql-server php php-bcmath php-curl php-imagick php-intl php-json php-mbstring php-mysql php-xml php-zip –y

Once Apache and related dependencies are finished installing, navigate to <http://domain> name or the IP address where you installed WordPress to verify that the default Apache page is showing.



Next we’re going to make the directory where we’ll host the wordpress files

-> mkdir -p /var/www/html

Change the directory’s ownership to the user www-data

-> sudo chown www-data: /var/www/html

And then curl the zip files from the official wordpress page and unzip them into the directory we created.

-> curl <https://wordpress.org/latest.tar.gz> | sudo -u www-data tar zx -C /var/www/html

When we installed Apache, it created a default configuration in the

/etc/apache2/sites-available directory. That’s where we want to create our

wordpress configuration.

-> sudo touch /etc/apache2/sites-available/wordpress.conf

After that we’ll use the tee command to drop text into our configuration file. We’re only using port 80 for now because it’s HTTP; once we install an SSL certificate from Let’s Encrypt, it will automatically create a configuration file for HTTPS (port 443) that we can further modify later. For now, copy and paste the following into your terminal:

-> vim /etc/apache2/sites-available/wordpress.conf

<< EOF  
<VirtualHost \*:80>  
 DocumentRoot /var/www/html/wordpress  
 <Directory /var/www/html/wordpress>  
 Options FollowSymLinks  
 AllowOverride Limit Options FileInfo  
 DirectoryIndex index.php  
 Require all granted  
 </Directory>  
 <Directory /var/www/html/wordpress/wp-content>  
 Options FollowSymLinks  
 Require all granted  
 </Directory>  
</VirtualHost>  
EOF

Now let’s enable the wordpress site, disable the default Apache site, and enable rewrites.

-> a2ensite wordpress  
-> a2dissite 000-default  
-> a2enmod rewrite

A2ensite and a2dissite stand for “Apache2 enable site” and “Apache2 disable site,” respectively; as you’ve probably guessed, a2enmod means “Apache2 enable modification.”

You can use MariaDB here for the database that stores all your posts, but for simplicity we’re going to use MySQL (which we installed at the beginning). First you need to create the database, which we’ll call “wordpress.” Log into your MySQL server.

-> mysql -u root –p

Next you’ll create a user and select a password. You can also substitute your own desired username and password in these commands to avoid doing it later.

-> CREATE USER wp\_user@localhost IDENTIFIED BY 'wp@123';

After creating the user, we need to give the user privileges to the database.

-> GRANT ALL PRIVILEGES ON \*.\* wp\_user@localhost;

-> FLUSH PRIVILEGES;

-> EXIT;

Now start the MySQL database.

-> service mysql start

You can verify that status of the server with

-> systemctl status mysql

The WordPress files we unzipped includes a sample configuration that we’ll copy into the configuration we’ll actually use.

-> -u www-data cp /var/www/html/wordpress/wp-config-sample.php /var/www/html/wordpress/wp-config.php

Now we need to edit the configuration file and connect our database to our WordPress installation. You can do this one of two ways: edit directly with vim, or use sed (“stream edit”) to edit the file. You can certainly use vim, but since I have the commands listed below, it’ll be faster to copy and paste. Leave database\_name\_here, username\_here, and password\_here alone, but change the db, user, and password name if you’ve changed them when you created your database user and password;

-> -u www-data sed -i 's/wordpress/wordpress/' /var/www/html/wordpress/wp-config.php

-> -u www-data sed -i 's/wp\_user/admin/' /var/www/html/wordpress/wp-config.php

-> -u www-data sed -i 's/wp@123 /defaultpassword/' /var/www/html/wordpress/wp-config.php

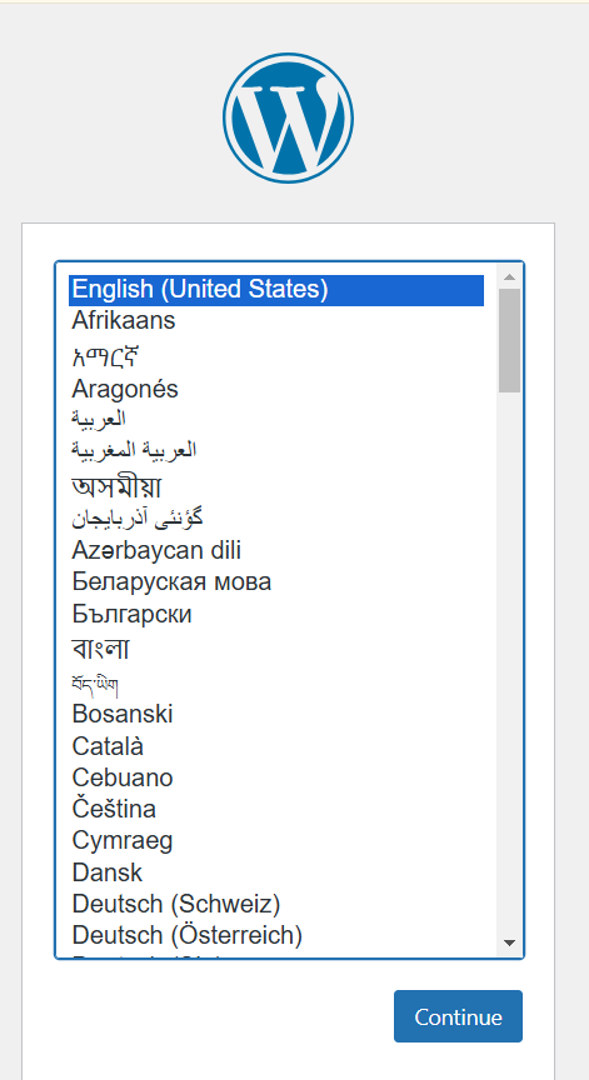
Finally we’ll enable a modification for hardening the website later, and restart Apache.

-> phpenmod mbstring

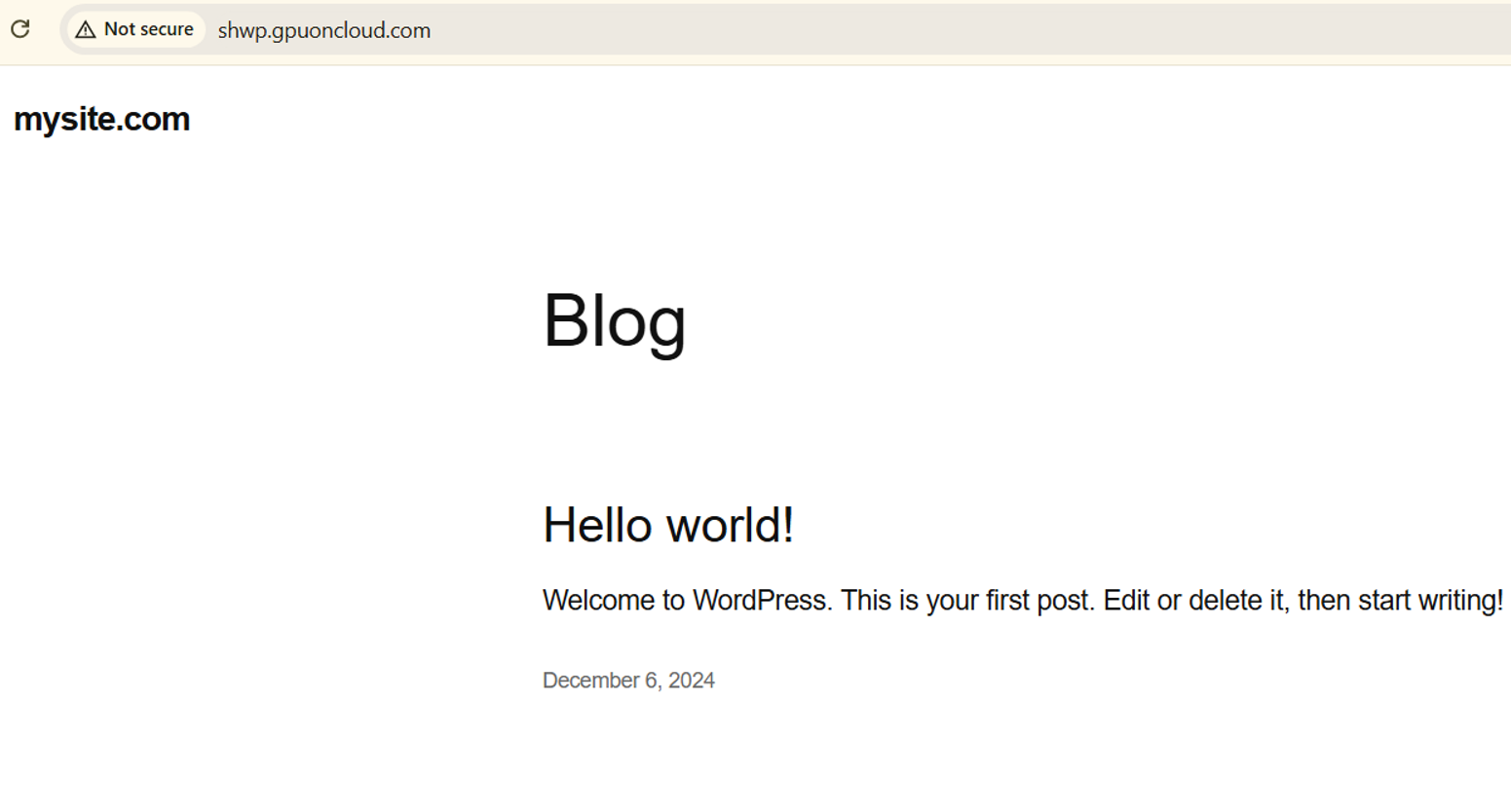
-> a2enmod headers

-> systemctl restart apache2

Navigate to <http://shwp.gpuoncloud.com>



Not Secure:



Once we browse url we got wordpress page like above

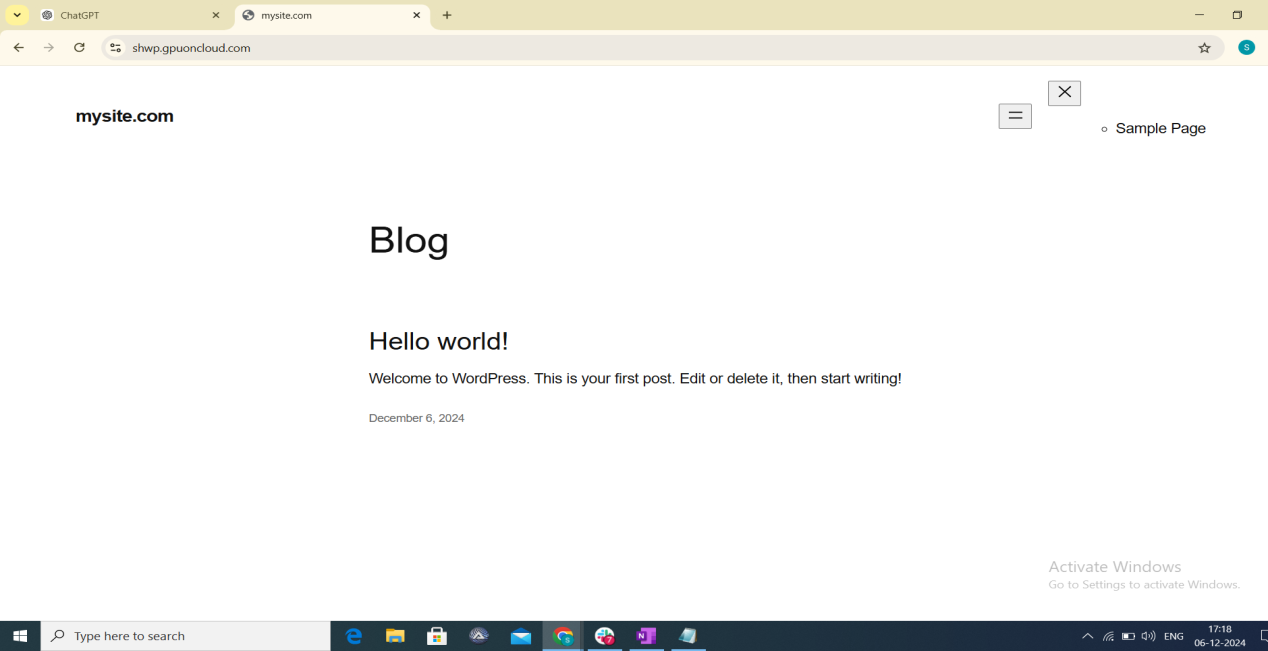
**Enable SSL:**

**Install Certbot**  
-> apt install certbot python3-certbot-apache -y

**Obtain an SSL Certificate** Run the following command to generate and install an SSL certificate:

-> certbot --apache

**Verify SSL** Visit [**https://shwp.gpuoncloud.com**](https://shwp.gpuoncloud.com) to ensure the site is secure.



**2: Installtion of OS-Ticket, also point domain to the server and enablen SSL:**

For OS-Ticket we need to install Apache, PHP with extension as well as mysql, so as I already install this so no required to install again.

**Download osTicket**

Navigate to /var/www/html and download the osTicket package.

->cd /var/www/html

->wget <https://github.com/osTicket/osTicket/releases/download/v1.17.1/osTicket-v1.17.1.zip>

**Extract osTicket**

Unzip the downloaded osTicket package.

->apt install unzip -y # Ensure unzip is installed

-> unzip osTicket-v1.17.1.zip -d /var/www/html

**Rename the Extracted Directory**

Rename the extracted folder to osticket for easier access.

->mv /var/www/html/upload /var/www/html/osticket

**Set Permissions**

Set the appropriate permissions for the osticket directory.

->chown -R www-data:www-data /var/www/html/osticket

->chmod -R 755 /var/www/html/osticket

**Create a MySQL Database and User for osTicket**

Log in to MySQL and create the database and user.

 ->mysql -u root –p

Once logged in, execute the following SQL commands:

sql

Copy code

CREATE DATABASE osticket;  
CREATE USER 'osticketuser'@'localhost' IDENTIFIED BY 'os@123';  
GRANT ALL PRIVILEGES ON osticket\_db.\* TO 'osticketuser'@'localhost';  
FLUSH PRIVILEGES;  
EXIT;

To enable the recommended extensions for osTicket on your Ubuntu 22.04 server, follow these steps:

**Step 1: Install Missing PHP Extensions**

Run the following command to install the required extensions:

->apt install php-gd php-imap php-xml php-mbstring php-json php-intl php-apcu php-zip php-bcmath php-curl php-phar php-opcache –y

**Step 2: Enable Installed Extensions**

Ensure that the installed extensions are enabled. Restart the Apache server to load the new extensions:

-> systemctl restart apache2

**Step 3: Verify Installed Extensions**

You can verify if the extensions are enabled by creating a PHP info file:

Create the file in the web directory:  
  
vim /var/www/html/phpinfo.php

Add the following content to the file:  
  
<?php  
phpinfo();  
?>

Save the file and access it in your browser:

<http://shot.gpuoncloud.com.com/phpinfo.php>

**Configure Apache for osTicket**

Create a new Apache configuration file for osTicket.

vim /etc/apache2/sites-available/osticket.conf

Add the following content:

<VirtualHost \*:80>  
 ServerAdmin webmaster@yourdomain.com  
 ServerName shot.gpuoncloud.com  
 ServerAlias shot.gpuoncloud.com  
 DocumentRoot /var/www/html/osticket

<Directory /var/www/html/osticket>  
 Options FollowSymLinks  
 AllowOverride All  
 Require all granted  
 </Directory>

ErrorLog ${APACHE\_LOG\_DIR}/error.log  
 CustomLog ${APACHE\_LOG\_DIR}/access.log combined  
</VirtualHost>

**Enable the Apache Configuration**

Enable the new site and the required Apache modules.

-> a2ensite osticket.conf  
-> a2enmod rewrite  
-> systemctl reload apache2

**Secure Apache with SSL**

To enable SSL, install and run Certbot to get a free SSL certificate.

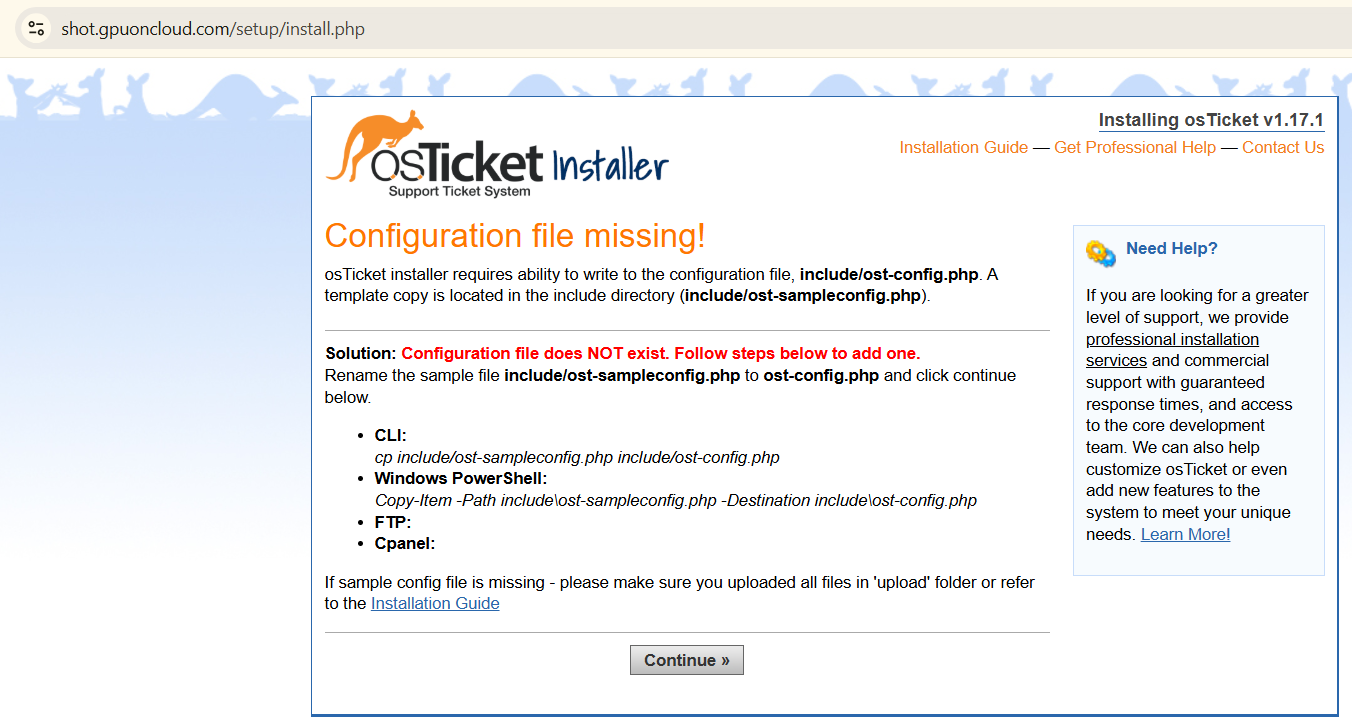
-> apt install certbot python3-certbot-apache –y

-> certbot –apache

Follow the prompts to set up SSL. When asked for your email and whether to share data with EFF, answer accordingly. Certbot will automatically configure SSL for your site.

**Step 10: Finalize osTicket Installation**

1. Open your web browser and navigate to [https://shot.gpuoncloud.com](https://yourdomain.com) or <https://www.shot.gpuoncloud.com>.



**3: Installtion of owncloud, also point domain to the server and enablen SSL:**

For owncloud we need to install Apache, PHP with extension as well as mysql, so as I already install this so no required to install again.

**->Install OwnCloud**

**Add the OwnCloud Repository:**

First, add the OwnCloud repository to your server:

**-**>wget -q [https://download.owncloud.org/download/repositories/stable/owncloud/owncloud.list -O /etc/apt/sources.list.d/owncloud.list](https://download.owncloud.org/download/repositories/stable/owncloud/owncloud.list%20-O%20/etc/apt/sources.list.d/owncloud.list)

1. **Import the Repository Key:**

-> wget -q https://download.owncloud.org/download/repositories/stable/owncloud/Release.key -O- | sudo apt-key add –

**Update the Package List:**

-> apt update -y

**Install OwnCloud:**

-> apt install owncloud

**Configure the Web Server:**

OwnCloud comes with Apache2 configuration. To make it work, create an Apache virtual host configuration:

-> nano /etc/apache2/sites-available/owncloud.conf

Add the following configuration (replace your\_domain.com with your actual domain):

<VirtualHost \*:80>

ServerName shoc.gpuoncloud.com

DocumentRoot /var/www/owncloud

<Directory /var/www/owncloud>

Options +FollowSymlinks

AllowOverride All

<IfModule mod\_dav.c>

Dav off

</IfModule>

</Directory>

ErrorLog ${APACHE\_LOG\_DIR}/error.log

CustomLog ${APACHE\_LOG\_DIR}/access.log combined

</VirtualHost>

### ****Configure MySQL Database for OwnCloud****

1. **Log into MySQL:**

-> mysql -u root -p

**Create a Database and User for OwnCloud:**

CREATE DATABASE owncloud;

CREATE USER 'oc\_user'@'localhost' IDENTIFIED BY 'wp@123';

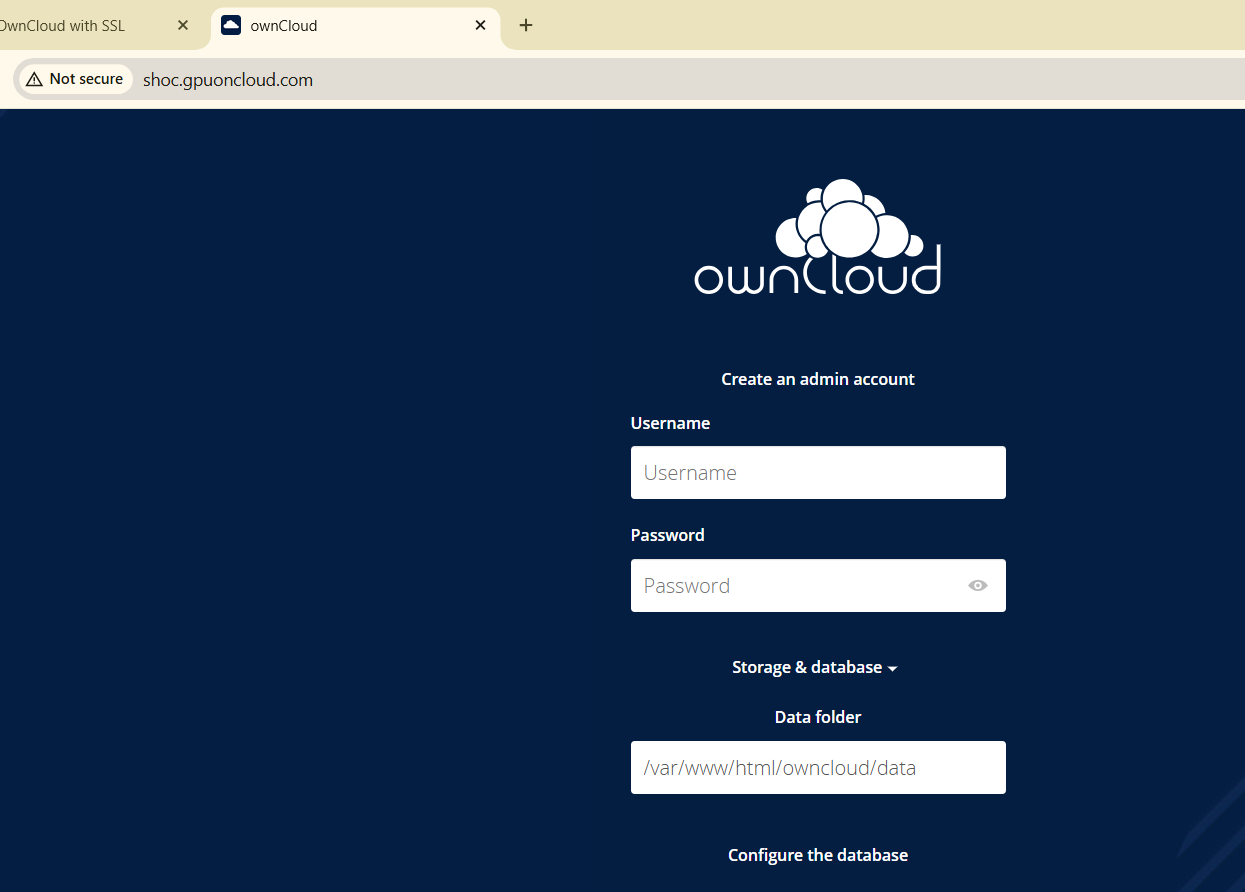
GRANT ALL PRIVILEGES ON owncloud.\* TO 'oc\_user'@'localhost';

FLUSH PRIVILEGES;

EXIT;

### ****Complete OwnCloud Setup in Web Browser****

Open a web browser and go to <http://shoc.gpuoncloud.com>.

Not Secure:

### ****Enable SSL for OwnCloud Using Let's Encrypt****

1. **Install Certbot:**

-> apt install certbot python3-certbot-apache

**Obtain an SSL Certificate:**

-> certbot --apache -d your\_domain.com

-> certbot –apache

**Finalize owncloud Installation:**

1. Open your web browser and navigate to <https://shoc.gpuoncloud.com> or <https://www.shoc.gpuoncloud.com>.

This one is secure with SSL