



Expert Cloud Consulting

Enhance Optimise & Scale

ASCP GPUonCLOUD Pvt Ltd

“Expert Cloud Consulting”

SOP | LAMP Stack Installation on Ubuntu 22.04

28 May 2025

—

Contributed by: Akshata Ujawane

Approved by : Akshay (In Review)

Expert Cloud Consulting

Office #811, Gera Imperium Rise,

Hinjewadi Phase-II Rd, Pune, India – 411057

“Expert Cloud Consulting”

LAMP Stack Installation on Ubuntu 22.04

1.0 Contents

1.0 Contents	
2.0 General Information	
2.1 Document Purpose	
2.2 Document References	
3.0 Document Overview	
4.0 Steps / Procedure	
4.1 Setup the Ubuntu Server Environment	
4.2 Add Firewall on GPUOnCloud	
4.3 Installing LAMP, WordPress, osTicket, and OwnCloud on Ubuntu 22.04	



2.0 General Information:

2.1 Document Purpose

This document provides a clear, step-by-step guide for installing and configuring the LAMP stack on Ubuntu 22.04, enabling the deployment of applications like WordPress, osTicket, and ownCloud. It ensures a consistent setup process across environments.

2.2 Document References

The following artifacts are referenced within this document. Please refer to the original documents for additional information.

Date	Document	Filename / Url
22.05.2025	Install LAMP Stack On Ubuntu 22.04 Server	https://www.digitalocean.com/community/tutorials/how-to-install-lamp-stack-on-ubuntu
26.05.2025	How to install and download wordpress on Ubuntu 22.04	https://www.digitalocean.com/community/tutorials/install-wordpress-on-ubuntu
27.05.2025	How to install and download osticket on Ubuntu 22.04	https://www.atlantic.net/dedicated-server-hosting/how-to-install-osticket-on-ubuntu-24-04/
28.05.2025	How to install and download owncloud on Ubuntu 22.04	https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-owncloud-on-ubuntu-18-04



Document Overview:

This document outlines the installation of the LAMP stack on Ubuntu 22.04 and the configuration of WordPress, OwnCloud, and osTicket. Each application is deployed under a dedicated Apache VirtualHost for organized access. PHP-FPM is enabled for OwnCloud, and the default Apache configuration is disabled to avoid conflicts.

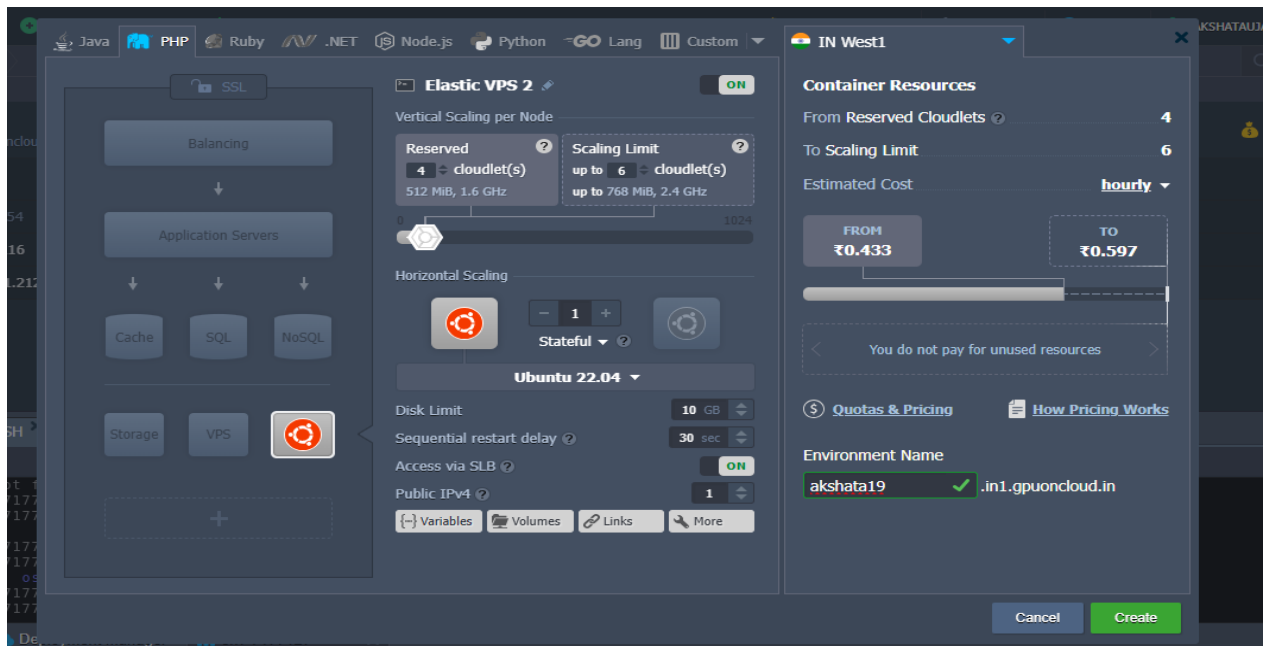
- Installed the LAMP stack on Ubuntu 22.04.
- Downloaded WordPress, OwnCloud, and osTicket into /var/www/html.
- Created separate Apache config files: wordpress.conf, owncloud.conf, and osticket.conf.
- Set appropriate DocumentRoot and <Directory> permissions in each config file.
- Enabled PHP-FPM handling for OwnCloud.
- Disabled the default Apache site to prevent conflicts.
- Enabled all new sites with a2ensite and reloaded Apache.
- Verified the applications via:
 - <http://<your-ip>/wordpress>
 - <http://<your-ip>/owncloud>
 - <http://<your-ip>/osticket>



3.0 Steps / Procedure

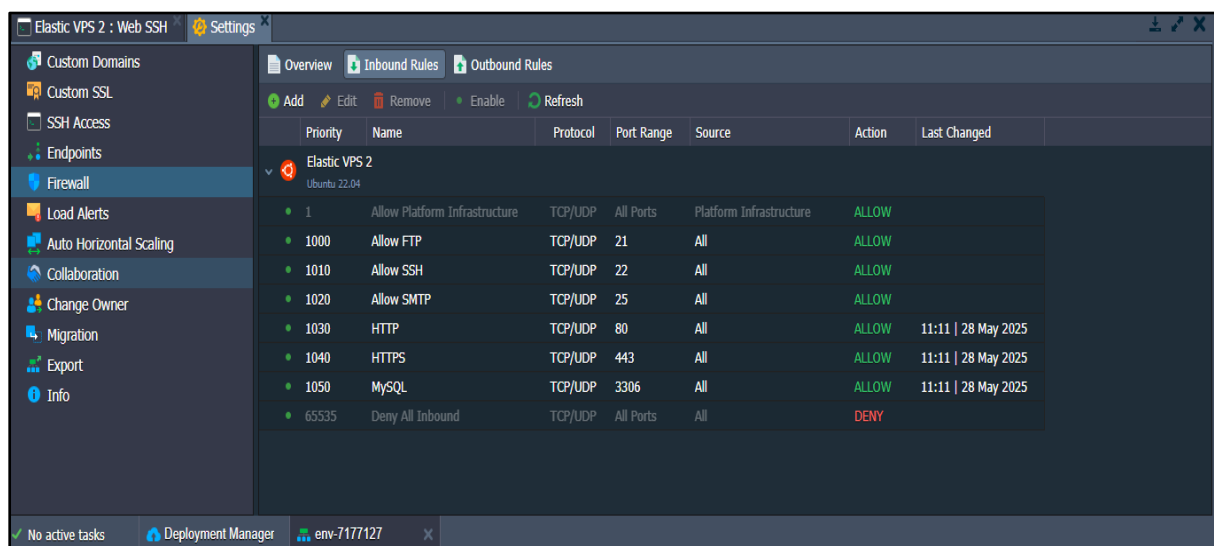
3.1: Setup the Ubuntu Server on GPUonCLOUD

The following steps were performed to set up the Ubuntu 22.04 server environment on **GPUonCLOUD**:



3.3: Configure Firewall Rules

Set up firewall rules on GPUonCLOUD to allow HTTP (80), HTTPS (443), and SSH (22) traffic, ensuring secure and accessible web and server management.



3.4: How To Install Linux, Apache, MySQL, PHP (LAMP) Stack on Ubuntu

```
sudo apt update & sudo apt upgrade
```

```
Elastic VPS 2 : Web SSH
Duplicate Session
Setting up snmp (5.9.1+dfsg-1ubuntu2.8) ...
Setting up binutils (2.38-4ubuntu2.8) ...
Setting up gnupg (2.2.27-3ubuntu2.3) ...
Setting up libpam-systemd:amd64 (249.11-0ubuntu3.15) ...
Setting up packagekit (1.2.5-2ubuntu3) ...
invoke-rc.d: policy-rc.d denied execution of force-reload.
Setting up python3-pkg-resources (59.6.0-1.2ubuntu0.22.04.2) ...
Setting up packagekit-tools (1.2.5-2ubuntu3) ...
Setting up python3-apt (2.4.0ubuntu4) ...
Processing triggers for dbus (1.12.20-2ubuntu4.1) ...
Processing triggers for mailcap (3.70+nmulubuntu1) ...
Processing triggers for libc-bin (2.35-0ubuntu3.9) ...
Processing triggers for ca-certificates (20240203~22.04.1) ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
root@node226422-akshata19:~#
```

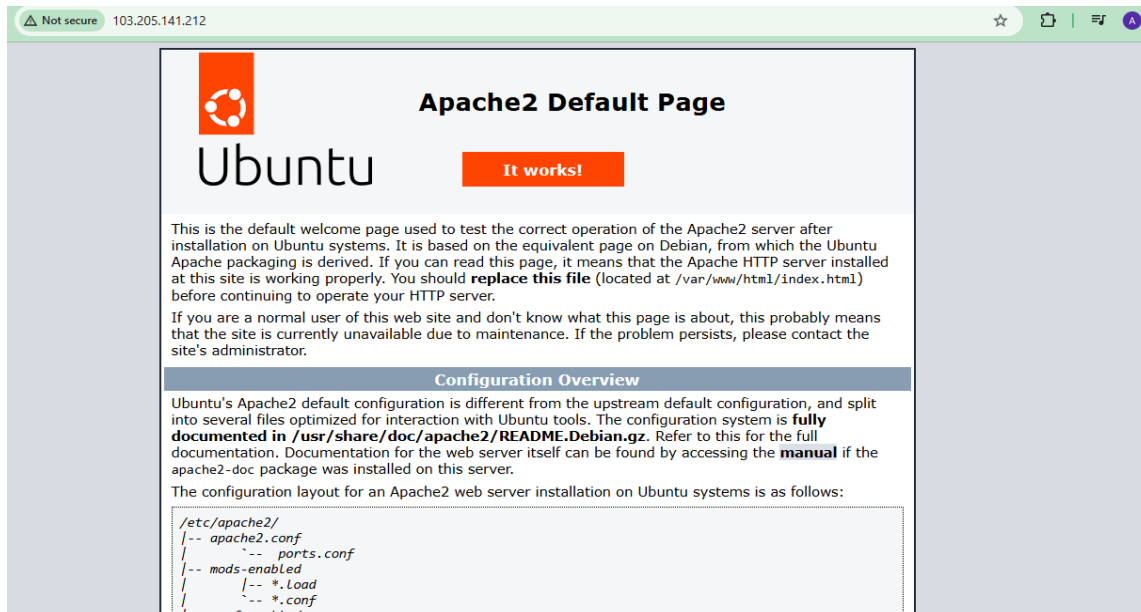
3.5: To install Apache2 on Ubuntu 22.04

```
sudo apt install apache2
sudo systemctl status apache2
sudo systemctl start apache2
sudo systemctl enable apache2
```

```
Duplicate Session
root@node226422-akshata19:~# sudo apt install apache2
root@node226422-akshata19:~# sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; disabled; vendor preset: enabled)
   Active: active (running) since Wed 2025-05-28 10:50:37 UTC; 5s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 16391 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
  Main PID: 16395 (apache2)
    Tasks: 55 (limit: 19660)
   Memory: 5.0M
   CGroup: /system.slice/apache2.service
           └─16395 /usr/sbin/apache2 -k start
             └─16396 /usr/sbin/apache2 -k start
               └─16397 /usr/sbin/apache2 -k start

May 28 10:50:37 node226422-akshata19.in1.gpuoncloud.in systemd[1]: Starting The Apache HTTP Server...
May 28 10:50:37 node226422-akshata19.in1.gpuoncloud.in systemd[1]: Started The Apache HTTP Server.
root@node226422-akshata19:~#
root@node226422-akshata19:~#
root@node226422-akshata19:~#
```

<http://103.205.141.212/>



4.1: Install MYSQL

`apt install mariadb-server mariadb-client`

```
root@node226422-akshata19:~# apt install mariadb-server mariadb-client
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
mariadb-client is already the newest version (1:10.6.22-0ubuntu0.22.04.1).
mariadb-server is already the newest version (1:10.6.22-0ubuntu0.22.04.1).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
root@node226422-akshata19:~#
```

Let's now secure our MariaDB database engine and disallow remote root login.

`mysql_secure_installation`

When prompted to change the root password during MySQL/MariaDB setup, type `n` and press Enter to skip if you're confident your current password is strong

```
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

You already have a root password set, so you can safely answer 'n'.

Change the root password? [Y/n]
```



For safety's sake, you will be prompted to remove anonymous users. Type Y.

```
By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them. This is intended only for testing, and to make the installation
go a bit smoother. You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] y
```

Next, disallow remote root login to prevent hackers from accessing your database. However, for testing purposes, you may want to allow log in remotely if you are configuring a virtual server

```
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? [Y/n]
```

Next, remove the test database.

```
By default, MariaDB 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? [Y/n]
```

Finally, reload the database to effect the changes.

```
Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n] y
```

4.2: Install PHP

we will install PHP as the last component of the LAMP stack.

`apt install php php-mysql`

```
root@node226422-akshata19:~# apt install 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
The following NEW packages will be installed:
  libapache2-mod-php8.1 php php-common php-mysql php8.1 php8.1-cli php8.1-common php8.1-mysql php8.1-opcache php8.1-readline
0 upgraded, 10 newly installed, 0 to remove and 0 not upgraded.
Need to get 5,264 kB of archives.
After this operation, 21.8 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```



To confirm that PHP is installed , created a `info.php` file at `/var/www/html/` path

`vim /var/www/html/info.php`


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

Append the following lines

Open your browser and append `/info.php` to the server's URL

<http://103.205.141.212/info.php>

Not secure
103.205.141.212/info.php
☆

PHP Version 7.4.33


System	Linux node226354-env-7177127.in1.gpuoncloud.in 5.2.0 #1 SMP Mon Sep 30 15:36:27 MSK 2024 x86_64
Build Date	May 9 2025 06:44:39
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.4/fpm
Loaded Configuration File	/etc/php/7.4/fpm/php.ini
Scan this dir for additional .ini files	/etc/php/7.4/fpm/conf.d
Additional .ini files parsed	/etc/php/7.4/fpm/conf.d/10-mysqld.ini, /etc/php/7.4/fpm/conf.d/10-opcache.ini, /etc/php/7.4/fpm/conf.d/10-pdo.ini, /etc/php/7.4/fpm/conf.d/15-xml.ini, /etc/php/7.4/fpm/conf.d/20-bcmath.ini, /etc/php/7.4/fpm/conf.d/20-calendar.ini, /etc/php/7.4/fpm/conf.d/20-ctype.ini, /etc/php/7.4/fpm/conf.d/20-curl.ini, /etc/php/7.4/fpm/conf.d/20-dom.ini, /etc/php/7.4/fpm/conf.d/20-exif.ini, /etc/php/7.4/fpm/conf.d/20-fli.ini, /etc/php/7.4/fpm/conf.d/20-fileinfo.ini, /etc/php/7.4/fpm/conf.d/20-ftp.ini, /etc/php/7.4/fpm/conf.d/20-gd.ini, /etc/php/7.4/fpm/conf.d/20-gettext.ini, /etc/php/7.4/fpm/conf.d/20-iconv.ini, /etc/php/7.4/fpm/conf.d/20-imagick.ini, /etc/php/7.4/fpm/conf.d/20-intl.ini, /etc/php/7.4/fpm/conf.d/20-json.ini, /etc/php/7.4/fpm/conf.d/20-mbstring.ini, /etc/php/7.4/fpm/conf.d/20-mysqli.ini, /etc/php/7.4/fpm/conf.d/20-pdo_mysql.ini, /etc/php/7.4/fpm/conf.d/20-phar.ini, /etc/php/7.4/fpm/conf.d/20-posix.ini, /etc/php/7.4/fpm/conf.d/20-readline.ini, /etc/php/7.4/fpm/conf.d/20-shmop.ini, /etc/php/7.4/fpm/conf.d/20-simplexml.ini, /etc/php/7.4/fpm/conf.d/20-sockets.ini, /etc/php/7.4/fpm/conf.d/20-sysvmsg.ini, /etc/php/7.4/fpm/conf.d/20-sysvsem.ini, /etc/php/7.4/fpm/conf.d/20-sysvshm.ini, /etc/php/7.4/fpm/conf.d/20-tokenizer.ini, /etc/php/7.4/fpm/conf.d/20-xmlreader.ini, /etc/php/7.4/fpm/conf.d/20-xmlwriter.ini, /etc/php/7.4/fpm/conf.d/20-xsl.ini, /etc/php/7.4/fpm/conf.d/20-zip.ini
PHP API	20190902
PHP Extension	20190902
Zend Extension	320190902
Zend Extension Build	API320190902.NTS
PHP Extension Build	API20190902.NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled



5.1: Create WordPress Database

Now it's time to log in to our MariaDB database as root and create a database for accommodating our WordPress data.

```
root@node226354-env-7177127:/var/www/html/wordpress# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 141
Server version: 10.6.22-MariaDB-0ubuntu0.22.04.1 Ubuntu 22.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

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

MariaDB [(none)]> █
```

Create a database for our WordPress installation

```
CREATE DATABASE wordpress_db;
```

```
MariaDB [(none)]> CREATE DATABASE wordpress_db;
Query OK, 1 row affected (0.00 sec)
```

Next, create a database user for our WordPress setup

```
CREATE USER 'wp_user'@'localhost' IDENTIFIED BY 'Password';
```

```
MariaDB [(none)]> CREATE USER 'wp_user'@'localhost' IDENTIFIED BY 'password';
Query OK, 0 rows affected (0.00 sec)
```

Output

```
MariaDB [(none)]> GRANT ALL ON wordpress_db.* TO 'wp_user'@'localhost' IDENTIFIED BY 'password';
Query OK, 0 rows affected (0.00 sec)
```

Grant privileges to the user Next, grant the user permissions to access the database

```
GRANT ALL ON wordpress_db.* TO 'wp_user'@'localhost' IDENTIFIED BY 'password';
```



```
FLUSH PRIVILEGES;
```

```
Exit;
```

Show DataBase;

```

Your MariaDB connection id is 133
Server version: 10.6.22-MariaDB-0ubuntu0.22.04.1 Ubuntu 22.04
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| osticket_db |
| performance_schema |
| sys |
| wordpress_db |
+-----+
6 rows in set (0.000 sec)

MariaDB [(none)]>

```

5.2: Install WordPress CMS

Go to your temp directory and download the latest WordPress File

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

Output

```

Elastic VPS 2 : Web SSH x
Duplicate Session
root@node226422-akshata19:/var/www# cd /tmp && wget https://wordpress.org/latest.tar.gz
--2025-05-29 04:53:20-- 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: 26926501 (26M) [application/octet-stream]
Saving to: 'latest.tar.gz'

latest.tar.gz           100%[=====] 25.68M  7.93MB/s  in 3.2s

2025-05-29 04:53:24 (7.93 MB/s) - 'latest.tar.gz' saved [26926501/26926501]

root@node226422-akshata19:/tmp#

```

Next, Uncompress the tarball which will generate a folder called “wordpress”.

```
tar -xvf latest.tar.gz
```



Output

```
root@wordpress:~# tar -xvf latest.tar.gz
wordpress/
wordpress/xmlrpc.php
wordpress/wp-blog-header.php
wordpress/readme.html
wordpress/wp-signup.php
wordpress/index.php
wordpress/wp-cron.php
wordpress/wp-config-sample.php
wordpress/wp-login.php
wordpress/wp-settings.php
wordpress/license.txt
wordpress/wp-content/
wordpress/wp-content/themes/
wordpress/wp-content/themes/twentynineteen/
wordpress/wp-content/themes/twentynineteen/footer.php
wordpress/wp-content/themes/twentynineteen/template-parts/
wordpress/wp-content/themes/twentynineteen/template-parts/content/
wordpress/wp-content/themes/twentynineteen/template-parts/content/content-excerpt.php
```

Copy the wordpress folder to `/var/www/html/` path.

```
cp -R wordpress /var/www/html/
```

Run the command below to change ownership of 'wordpress' directory.

```
chown -R www-data:www-data /var/www/html/wordpress/
```

change File permissions of the WordPress folder

```
chmod -R 755 /var/www/html/wordpress/
```

Create 'uploads' directory.

```
mkdir /var/www/html/wordpress/wp-content/uploads
```

Finally, change permissions of 'uploads' directory

```
chown -R www-data:www-data /var/www/html/wordpress/wp-content/uploads/
```

Sample Apache Virtual Host Config for WordPress

```
sudo nano /etc/apache2/sites-available/wordpress.conf
```

```
<VirtualHost *:80>
```

```
    ServerAdmin webmaster@gpuoncloud.com
```

```
    ServerName 103.205.141.212/
```

```
    DocumentRoot /var/www/html/wordpress
```

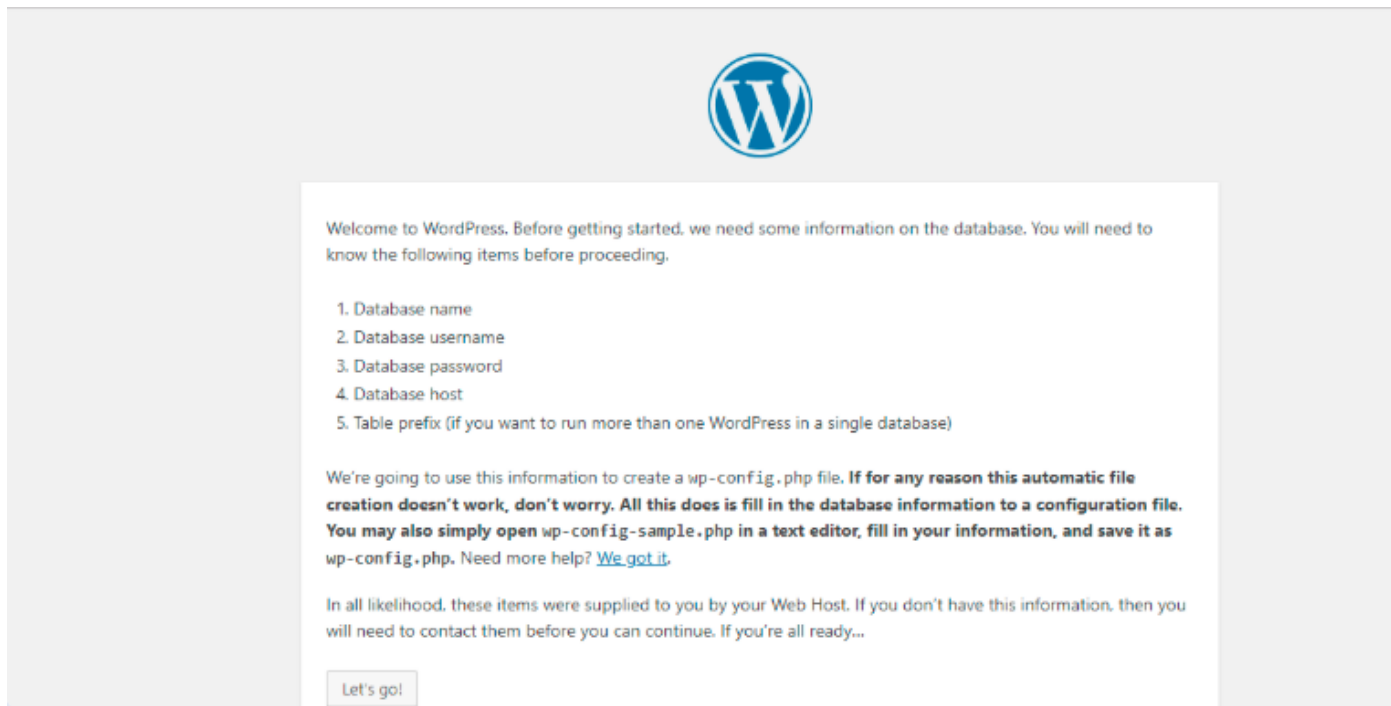


```
<Directory /var/www/html/wordpress>
    Options Indexes FollowSymLinks
    AllowOverride All
    Require all granted
</Directory>

ErrorLog ${APACHE_LOG_DIR}/wordpress_error.log
CustomLog ${APACHE_LOG_DIR}/wordpress_access.log combined
</VirtualHost>
```

Open your browser and go to the server's URL. In my case it's

<http://103.205.141.212/wordpress/>

The image shows the WordPress installation database configuration screen. At the top center is the WordPress logo. Below it, a white box contains the following text: "Welcome to WordPress. Before getting started, we need some information on the database. You will need to know the following items before proceeding." This is followed by a numbered list: 1. Database name, 2. Database username, 3. Database password, 4. Database host, and 5. Table prefix (if you want to run more than one WordPress in a single database). Below the list, a paragraph states: "We're going to use this information to create a wp-config.php file. If for any reason this automatic file creation doesn't work, don't worry. All this does is fill in the database information to a configuration file. You may also simply open wp-config-sample.php in a text editor, fill in your information, and save it as wp-config.php. Need more help? [We got it.](#)" Another paragraph follows: "In all likelihood, these items were supplied to you by your Web Host. If you don't have this information, then you will need to contact them before you can continue. If you're all ready..." At the bottom of the white box is a button labeled "Let's go!".

Welcome to WordPress. Before getting started, we need some information on the database. You will need to know the following items before proceeding.

1. Database name
2. Database username
3. Database password
4. Database host
5. Table prefix (if you want to run more than one WordPress in a single database)

We're going to use this information to create a wp-config.php file. If for any reason this automatic file creation doesn't work, don't worry. All this does is fill in the database information to a configuration file. You may also simply open wp-config-sample.php in a text editor, fill in your information, and save it as wp-config.php. Need more help? [We got it.](#)

In all likelihood, these items were supplied to you by your Web Host. If you don't have this information, then you will need to contact them before you can continue. If you're all ready...

Let's go!

Fill out the form as shown with the credentials specified when creating the WordPress database in the MariaDB database. Leave out the database host and table prefix and Hit 'Submit' button.



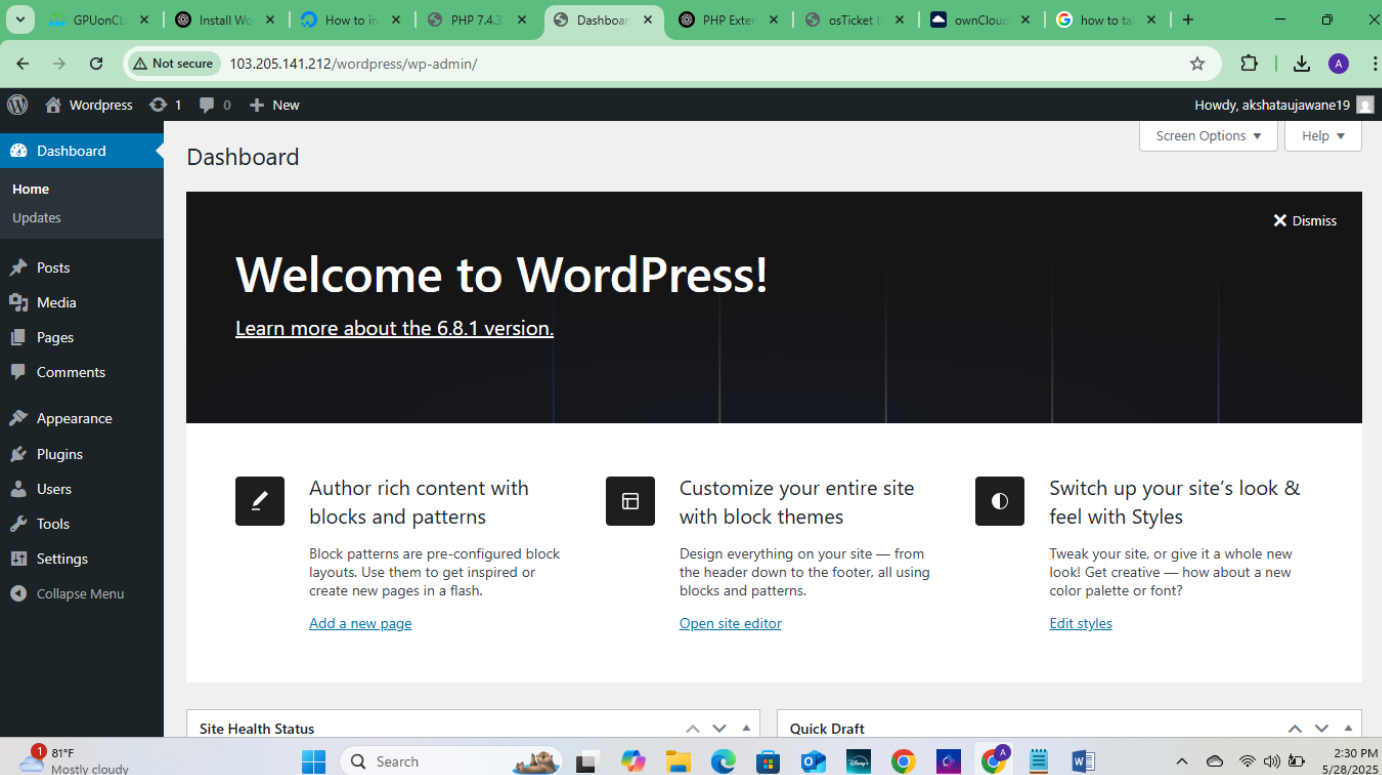


The image shows the WordPress installation screen for database configuration. At the top is the WordPress logo. Below it, a message states: "Below you should enter your database connection details. If you're not sure about these, contact your host." There are five input fields with labels and descriptions:

- Database Name:** The name of the database you want to use with WordPress.
- Username:** Your database username.
- Password:** Your database password.
- Database Host:** You should be able to get this info from your web host, if localhost doesn't work.
- Table Prefix:** If you want to run multiple WordPress installations in a single database, change this.

At the bottom left is a "Submit" button.

Provide your login credentials and hit 'Login'.



The image shows the WordPress dashboard in a web browser. The address bar shows "103.205.141.212/wordpress/wp-admin/". The dashboard has a dark sidebar on the left with a menu: Dashboard, Home, Updates, Posts, Media, Pages, Comments, Appearance, Plugins, Users, Tools, Settings, and Collapse Menu. The main content area has a "Welcome to WordPress!" message with a link to "Learn more about the 6.8.1 version." Below this are three cards: "Author rich content with blocks and patterns" (with a link "Add a new page"), "Customize your entire site with block themes" (with a link "Open site editor"), and "Switch up your site's look & feel with Styles" (with a link "Edit styles"). At the bottom, there is a "Site Health Status" section and a "Quick Draft" section. The browser's taskbar at the bottom shows various application icons and the system clock indicating 2:30 PM on 5/28/2025.

Success! Your WordPress dashboard is now live. You're all set to start building your first blog or website. Explore themes, plugins, and settings to make it your own. Congrats on getting this far!



5.2: Install osTicket

`sudo apt update`

start and enable the Apache service.

`systemctl start apache2`

`systemctl enable apache2`

Install and Configure MariaDB Database

First, install the MariaDB database server using the following command:

`apt install mariadb-server -y`

Next, connect to the MariaDB shell

MySQL

After connecting to the MariaDB, create a database and user for osTicket.

`CREATE DATABASE osticket;`

`GRANT ALL PRIVILEGES ON osticket.* TO osticket@localhost IDENTIFIED BY "admin";`

Next, flush the privileges and exit from the MariaDB.

`FLUSH PRIVILEGES;`

`EXIT;`

`SHOW DATABASES;`

```
root@MariaDB:~# mysql
mysql: [Warning] Using a password on the command line interface can be insecure.
Server version: 10.6.22-MariaDB-0ubuntu0.22.04.1 Ubuntu 22.04
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| osticket_db |
| performance_schema |
| sys |
| wordpress_db |
+-----+
6 rows in set (0.000 sec)

MariaDB [(none)]>
```



5:3 Install osTicket

First, change the directory to Apache web root and download the latest osTicket version inside that directory.

```
cd /var/www/html
curl -s https://api.github.com/repos/osTicket/osTicket/releases/latest | grep
browser_download_url | cut -d '"' -f 4 | wget -i -
```

Next, unzip the downloaded file.

```
unzip osTicket*.zip -d osTicket
```

Next, copy the osTicket sample configuration file.

```
Cp
```

```
/var/www/html/osTicket/upload/include/ost-sampleconfig.php
```

```
/var/www/html/osTicket/upload/include/ost-config.php
```

Then, set the necessary permissions and ownership to the osTicket directory.

```
chown -R www-data:www-data /var/www/html/osTicket/
```

```
chmod -R 775 /var/www/html/osTicket/
```

Configure Apache for osTicket

```
sudo nano /etc/apache2/sites-available/osticket.conf
```

```
<VirtualHost *:80>
```

```
    ServerAdmin admin@yourdomain.com
```

```
    DocumentRoot /var/www/html/osticket
```

```
    ServerName 103.205.141.38/osticket
```

```
<Directory /var/www/html/osticket/>
```

```
    Options +FollowSymlinks
```

```
    AllowOverride All
```

```
    Require all granted
```

```
</Directory>
```

```
    ErrorLog ${APACHE_LOG_DIR}/osticket_error.log
```

```
    CustomLog ${APACHE_LOG_DIR}/osticket_access.log combined
```

```
</VirtualHost>
```



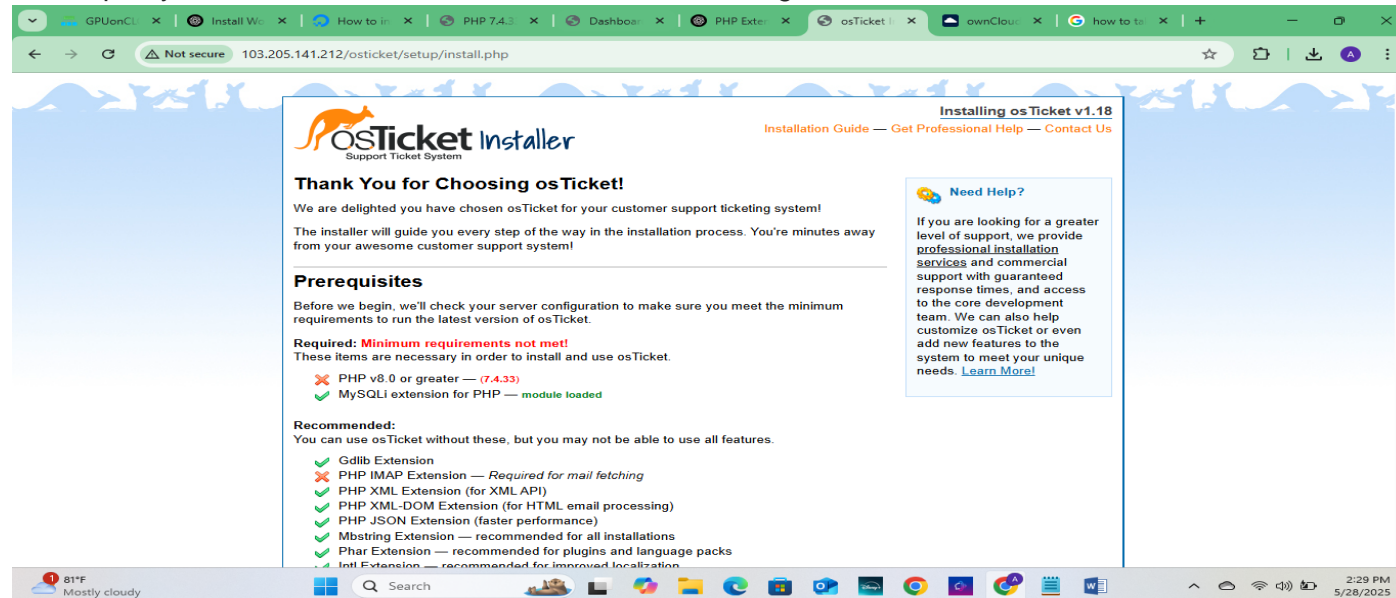
5.4: Enable the site and restart Apache

```
sudo a2ensite osticket.conf  
sudo systemctl reload apache2
```

Finally, restart the Apache service to apply the changes

Access osTicket Web UI

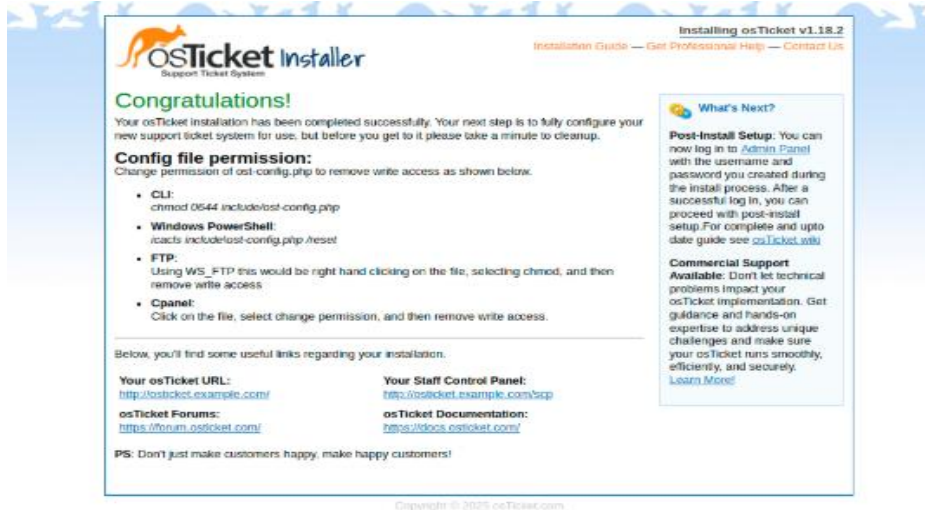
Now, open your web browser and access the osTicket using the URL



Click on **Continue**. You will see the osTicket configuration page.



Provide your helpdesk name, URL, admin username, password, email, and database credentials, then click on Install Now. Once the osTicket is installed, you will see the following page.



6:1: Install ownCloud:

Create a database for our owncloud installation

```
CREATE DATABASE owncloud;
```

Next, create a database user for our owncloud setup.

```
GRANT ALL PRIVILEGES ON owncloud.* TO owncloud@localhost IDENTIFIED BY "admin";
```

Grant privileges to the user Next, grant the user permissions to access the database

```
GRANT ALL PRIVILEGES ON owncloud_db.* TO 'owncloud_user'@'localhost';
```

Great, now you can exit the database.

```
FLUSH PRIVILEGES;
```

```
EXIT;
```

```
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'MariaDB [(none)]> SHOW DATABASES;' at line 1
MariaDB [(none)]> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| osticket_db |
| owncloud_db |
| performance_schema |
| sys |
| wordpress_database |
+-----+
7 rows in set (0.001 sec)

MariaDB [(none)]> █
```



6.2: Download and Install OwnCloud:

Download the latest version of ownCloud from the official website by executing the following command on your server

Navigate to Apache's web root

```
cd /var/www/html
```

Download OwnCloud

```
sudo wget https://download.owncloud.org/community/owncloud-latest.tar.bz2
```

Extract the downloaded file

```
sudo tar -xvf owncloud-latest.tar.bz2
```

Set correct permissions

```
sudo chown -R www-data:www-data owncloud
```

```
sudo chmod -R 755 owncloud
```

Create a new Apache config file for OwnCloud

```
sudo nano /etc/apache2/sites-available/owncloud.conf
```

Paste the following configuration **inside the file**:

```
<VirtualHost *:80>
```

```
    ServerAdmin admin@example.com
```

```
    DocumentRoot /var/www/html/owncloud
```

```
    ServerName your-domain.com
```

```
<Directory /var/www/html/owncloud>
```

```
    Options +FollowSymlinks
```

```
    AllowOverride All
```

```
    Require all granted
```

```
</Directory>
```

```
    ErrorLog ${APACHE_LOG_DIR}/owncloud_error.log
```

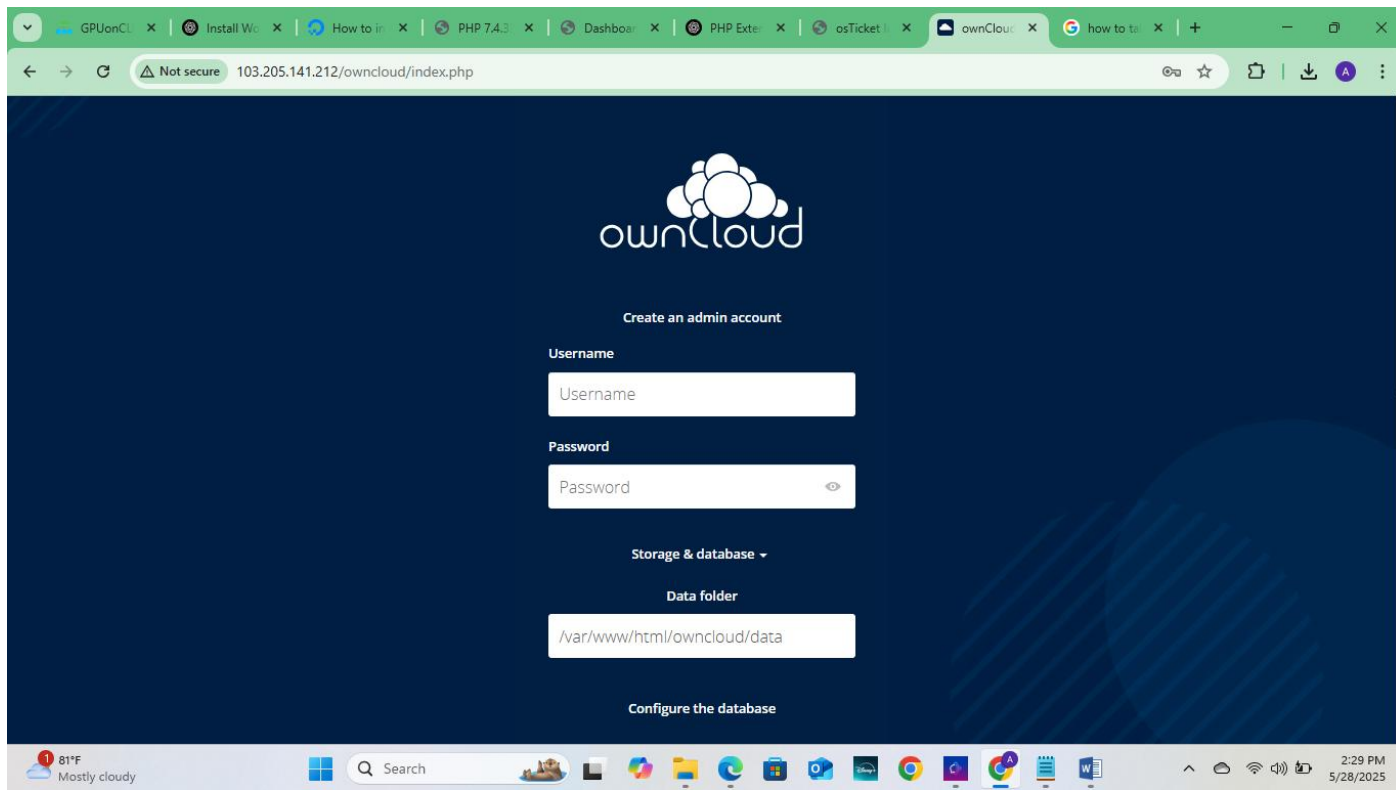
```
    CustomLog ${APACHE_LOG_DIR}/owncloud_access.log combined
```

```
</VirtualHost>
```



Open your browser and go to the server's URL. In my case it's

<http://103.205.141.212/owncloud/index.php>



Set your ownCloud admin username, password, data folder, database name, database username, and password, and click on the Finish setup button. You should see the ownCloud login page

Enable the site and restart Apache

```
sudo a2ensite owncloud.conf
```

```
sudo systemctl reload apache2
```

