



Expert Cloud Consulting

Enhance Optimise & Scale

ASCP GPUonCLOUD Pvt Ltd

“Expert Cloud Consulting” -

SOP | Install And Configure LAMP Stack On Ubuntu

28.May.2025

—

Contributed by Dhanshri Ananda Patil

Approved by Akshay Shinde (In Review)

Expert Cloud Consulting

Office #811, Gera Imperium Rise,

Hinjewadi Phase-II Rd, Pune, India – 411057

“Expert Cloud Consulting”

Install and Configure LAMP Stack On Ubuntu

1.0 Contents	1
2.0 General Information:	2
2.1 Document Purpose.....	2
2.2 Document References.....	2
3.0 Document Overview.....	3
4.0 Steps / Procedure.....	4
4.1: Setup the ubuntu server Environment.....	4
4.2: Add Firewall on GPUOnCloud.....	4
4.3: Installing LAMP, WordPress, osTicket, and OwnCloud on Ubuntu 22.04.....	5
.....	19





2.0 General Information:

2.1 Document Purpose

To provide detailed and step by step guide and to install and configure on the lamp stack on the ubuntu 22.04 for web applications such as osticket, wordpress and owncloud

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
26.05.2025	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/
27.05.2025	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



3.0 Document Overview:

Apache VirtualHost Setup for WordPress, OwnCloud, and osTicket

1. Installed LAMP stack on Ubuntu 22.04.
2. Downloaded and extracted WordPress, OwnCloud, and osTicket to /var/www/html/.
3. Created separate Apache config files: wordpress.conf, owncloud.conf, and osticket.conf.
4. Set DocumentRoot and <Directory> permissions in each file.
5. Added PHP-FPM handler for OwnCloud.
6. Disabled the default Apache config to avoid conflicts.
7. Enabled each site using a2ensite and reloaded Apache.
8. Verified each app via http://<IP>/wordpress, /owncloud, and /osticket.

.



4.0 Steps / Procedure

4.1 : Setup the ubuntu server Environment

The following procedure has been done during the installation of ubuntu server:

Add Ubuntu 22.04 on Dashboard

4.2: Add a Firewall on GPUonCloud

Adding a firewall on GPUonCloud (or any cloud infrastructure like AWS, Azure, GCP, etc.) is critical for security and access control.

Overview Inbound Rules Outbound Rules							
Add Edit Remove Enable Refresh							
	Priority	Name	Protocol	Port Range	Source	Action	Last Changed
Elastic VPS							
Ubuntu 22.04							
1	1	Allow Platform Infrastructure	TCP/UDP	All Ports	Platform Infrastructure	ALLOW	
1000	1000	Allow FTP	TCP/UDP	21	All	ALLOW	
1010	1010	Allow SSH	TCP/UDP	22	All	ALLOW	
1020	1020	Allow SMTP	TCP/UDP	25	All	ALLOW	
1030	1030	HTTP	TCP/UDP	80	All	ALLOW	14:53 26 May 2025
1040	1040	HTTPS	TCP/UDP	443	All	ALLOW	14:54 26 May 2025
1050	1050	MySQL	TCP/UDP	3306	All	ALLOW	14:54 26 May 2025
65535	65535	Deny All Inbound	TCP/UDP	All Ports	All	DENY	

4.3: Installing LAMP, WordPress, osTicket, and OwnCloud on Ubuntu 22.04

1. Update System:

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

```
root@node226139-dhanshri14:~# apt update
gn:1 https://repo.virtuozzo.com/ctpreset/deb InRelease
it:2 https://repo.virtuozzo.com/ctpreset/deb Release
it:3 http://archive.ubuntu.com/ubuntu jammy InRelease
et:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]
gn:5 https://repo.virtuozzo.com/ctpreset/deb Release.gpg
et:6 http://archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]
it:7 https://ppa.launchpadcontent.net/ondrej/php/ubuntu jammy InRelease
it:8 http://archive.ubuntu.com/ubuntu jammy-backports InRelease
et:9 http://archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [4,564 kB]
et:10 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [3,264 kB]
etched 8,085 kB in 3s (2,553 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
1 package can be upgraded. Run 'apt list --upgradable' to see it.
root@node226139-dhanshri14:~# systemctl start apache2
root@node226139-dhanshri14:~# systemctl status apache2
```

2. Install Apache2

- `sudo apt install apache2 -y`

```
May 28 07:52:16 node226139-dhanshri14.in1.gpuoncloud.in systemd[1]: Reloaded The Apache HTTP Server.
root@node226139-dhanshri14:~# apt install apache2
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
apache2 is already the newest version (2.4.52-1ubuntu4.14).
The following packages were automatically installed and are no longer required:
  php8.1-apcu php8.1-imagick php8.1-intl
Use 'apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
root@node226139-dhanshri14:~#
```

- `sudo systemctl enable apache2`
- `sudo systemctl start apache2`
- `sudo systemctl status apache2`

```
root@node226139-dhanshri14:~# systemctl start apache2
root@node226139-dhanshri14:~# systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2025-05-28 04:25:19 UTC; 6h ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 3960 ExecReload=/usr/sbin/apachectl graceful (code=exited, status=0/SUCCESS)
  Main PID: 478 (apache2)
    Tasks: 8 (limit: 19660)
   Memory: 89.9M
   CGroup: /system.slice/apache2.service
           └─ 478 /usr/sbin/apache2 -k start
             3964 /usr/sbin/apache2 -k start
             3965 /usr/sbin/apache2 -k start
             3966 /usr/sbin/apache2 -k start
             3967 /usr/sbin/apache2 -k start
             3968 /usr/sbin/apache2 -k start
             3971 /usr/sbin/apache2 -k start
             4967 /usr/sbin/apache2 -k start

May 28 07:19:05 node226139-dhanshri14.in1.gpuoncloud.in systemd[1]: Reloading The Apache HTTP Server...
May 28 07:19:05 node226139-dhanshri14.in1.gpuoncloud.in systemd[1]: Reloaded The Apache HTTP Server.
May 28 07:23:41 node226139-dhanshri14.in1.gpuoncloud.in systemd[1]: Reloading The Apache HTTP Server...
May 28 07:23:41 node226139-dhanshri14.in1.gpuoncloud.in systemd[1]: Reloaded The Apache HTTP Server.
May 28 07:50:49 node226139-dhanshri14.in1.gpuoncloud.in systemd[1]: Reloading The Apache HTTP Server...
May 28 07:50:49 node226139-dhanshri14.in1.gpuoncloud.in systemd[1]: Reloaded The Apache HTTP Server.
May 28 07:52:07 node226139-dhanshri14.in1.gpuoncloud.in systemd[1]: Reloading The Apache HTTP Server...
May 28 07:52:07 node226139-dhanshri14.in1.gpuoncloud.in systemd[1]: Reloaded The Apache HTTP Server.
May 28 07:52:16 node226139-dhanshri14.in1.gpuoncloud.in systemd[1]: Reloading The Apache HTTP Server...
May 28 07:52:16 node226139-dhanshri14.in1.gpuoncloud.in systemd[1]: Reloaded The Apache HTTP Server.
root@node226139-dhanshri14:~#
```

To verify further, open your browser and go to your server's IP address.

<http://103.217.221.253>



3. Install Mysql

Next, we are going to install the MariaDB database engine to hold our Wordpress files. MariaDB is an open-source fork of MySQL and most of the hosting companies use it instead of MySQL.

- apt install mariadb-server mariadb-client

```
root@node226139-dhanshri14:~# 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).
The following packages were automatically installed and are no longer required:
  php8.1-apcu php8.1-imap php8.1-intl
Use 'apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
root@node226139-dhanshri14:~#
```

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

- mysql secure-installation

```
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]
```

Prompted to remove anonymous users

```
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
```

Disallow remote root login to prevent hackers from accessing your database.

```
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] y
```

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
```

3. Install PHP

- apt install php php-mysql

```
root@node226139-dhanshril4:~# apt install php php-mysql
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
php is already the newest version (2:8.4+96+ubuntu22.04.1+deb.sury.org+1).
The following packages were automatically installed and are no longer required:
  php8.1-apcu php8.1-imap php8.1-intl
Use 'apt autoremove' to remove them.
The following NEW packages will be installed:
  php-mysql php8.4-mysql
0 upgraded, 2 newly installed, 0 to remove and 1 not upgraded.
Need to get 141 kB of archives.
After this operation, 480 kB of additional disk space will be used.
Get:1 https://ppa.launchpadcontent.net/ondrej/php/ubuntu jammy/main amd64 php8.4-mysql amd64 8.4.7-1+ubuntu22.04.1+deb.sury.org+1 [133 kB]
Get:2 https://ppa.launchpadcontent.net/ondrej/php/ubuntu jammy/main amd64 php-mysql all 2:8.4+96+ubuntu22.04.1+deb.sury.org+1 [7,502 B]
Fetched 141 kB in 2s (74.8 kB/s)
Selecting previously unselected package php8.4-mysql.
(Reading database ... 40619 files and directories currently installed.)
Preparing to unpack .../php8.4-mysql_8.4.7-1+ubuntu22.04.1+deb.sury.org+1_amd64.deb ...
Unpacking php8.4-mysql (8.4.7-1+ubuntu22.04.1+deb.sury.org+1) ...
Selecting previously unselected package php-mysql.
Preparing to unpack .../php-mysql_2%3a8.4+96+ubuntu22.04.1+deb.sury.org+1_all.deb ...
Unpacking php-mysql (2:8.4+96+ubuntu22.04.1+deb.sury.org+1) ...
Setting up php8.4-mysql (8.4.7-1+ubuntu22.04.1+deb.sury.org+1) ...

Creating config file /etc/php/8.4/mods-available/mysqlnd.ini with new version

Creating config file /etc/php/8.4/mods-available/mysqli.ini with new version

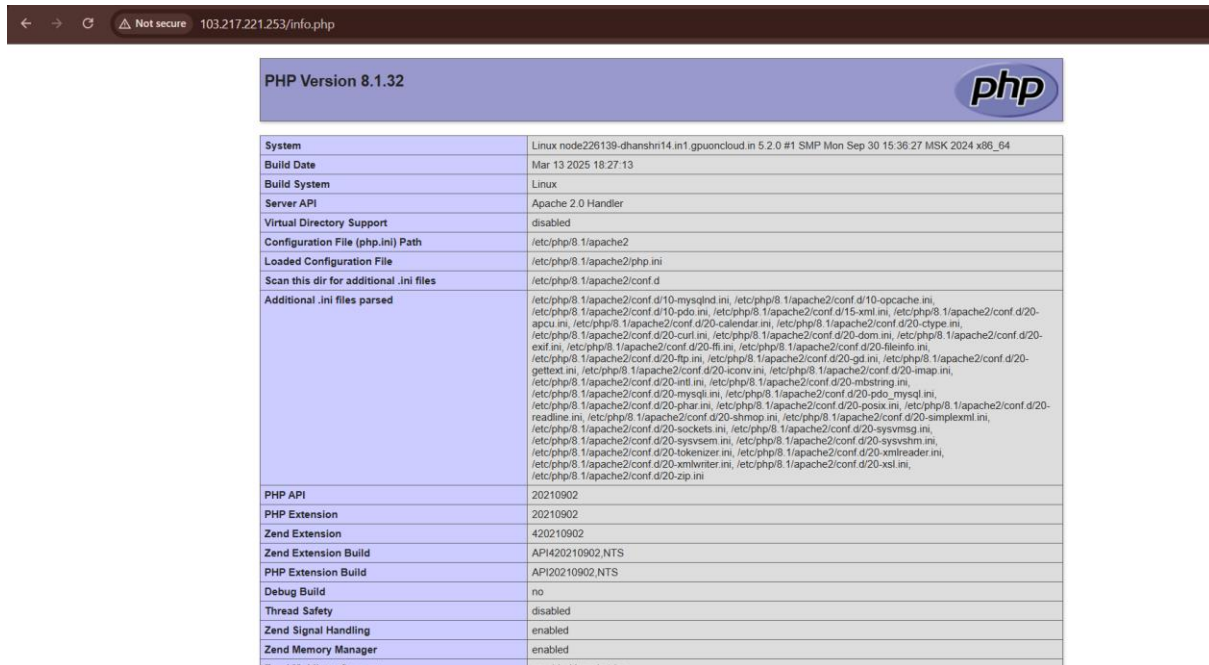
Creating config file /etc/php/8.4/mods-available/pdo_mysql.ini with new version
Setting up php-mysql (2:8.4+96+ubuntu22.04.1+deb.sury.org+1) ...
Processing triggers for libapache2-mod-php8.4 (8.4.7-1+ubuntu22.04.1+deb.sury.org+1) ...
invoke-rc.d: policy-rc.d denied execution of restart.
Processing triggers for php8.4-cli (8.4.7-1+ubuntu22.04.1+deb.sury.org+1) ...
root@node226139-dhanshril4:~#
```



```
sudo nano /var/www/html/info.php
```

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

<http://103.217.221.253/info.php>



PHP Version 8.1.32	
System	Linux node226139-dhanshri14.in1.gpcloud.in 5.2.0 #1 SMP Mon Sep 30 15:36:27 MSK 2024 x86_64
Build Date	Mar 13 2025 18:27:13
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-mysqld.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/15-xml.ini, /etc/php/8.1/apache2/conf.d/20-apcu.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-curl.ini, /etc/php/8.1/apache2/conf.d/20-dom.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-gd.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-imap.ini, /etc/php/8.1/apache2/conf.d/20-intl.ini, /etc/php/8.1/apache2/conf.d/20-mbstring.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-simplexml.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-sysvshm.ini, /etc/php/8.1/apache2/conf.d/20-sysvsem.ini, /etc/php/8.1/apache2/conf.d/20-tokenizer.ini, /etc/php/8.1/apache2/conf.d/20-xmlreader.ini, /etc/php/8.1/apache2/conf.d/20-xmlwriter.ini, /etc/php/8.1/apache2/conf.d/20-xsl.ini, /etc/php/8.1/apache2/conf.d/20-zip.ini
PHP API	20210902
PHP Extension	20210902
Zend Extension	420210902
Zend Extension Build	API420210902.NTS
PHP Extension Build	API20210902.NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by extension

4. 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.

```
- mysql -u root -p
```

```
root@node226139-dhanshri14:~# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MariaDB connection id is 30
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_database;
```

Next, create a database user for our WordPress setup.

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

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';
```

Great, now you can exit the database.

```
FLUSH PRIVILEGES;
```

```
Exit;
```

OUTPUT :

```

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

```

5. Install Wordpress CMS

Go to your temp directory and download the latest WordPress File

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

```

root@node226139-dhanshi14:~# cd /tmp && wget https://wordpress.org/latest.tar.gz
--2025-05-29 04:42:30-- 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%[=====>] 25.60M  7.93MB/s  in 3.2s

2025-05-29 04:42:34 (7.93 MB/s) - 'latest.tar.gz' saved [26926501/26926501]

root@node226139-dhanshi14:/tmp#

```

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

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

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/
```

Create Separate Apache Config Files

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

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

```
ServerAdmin admin@103.217.221.253
```

```
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
```

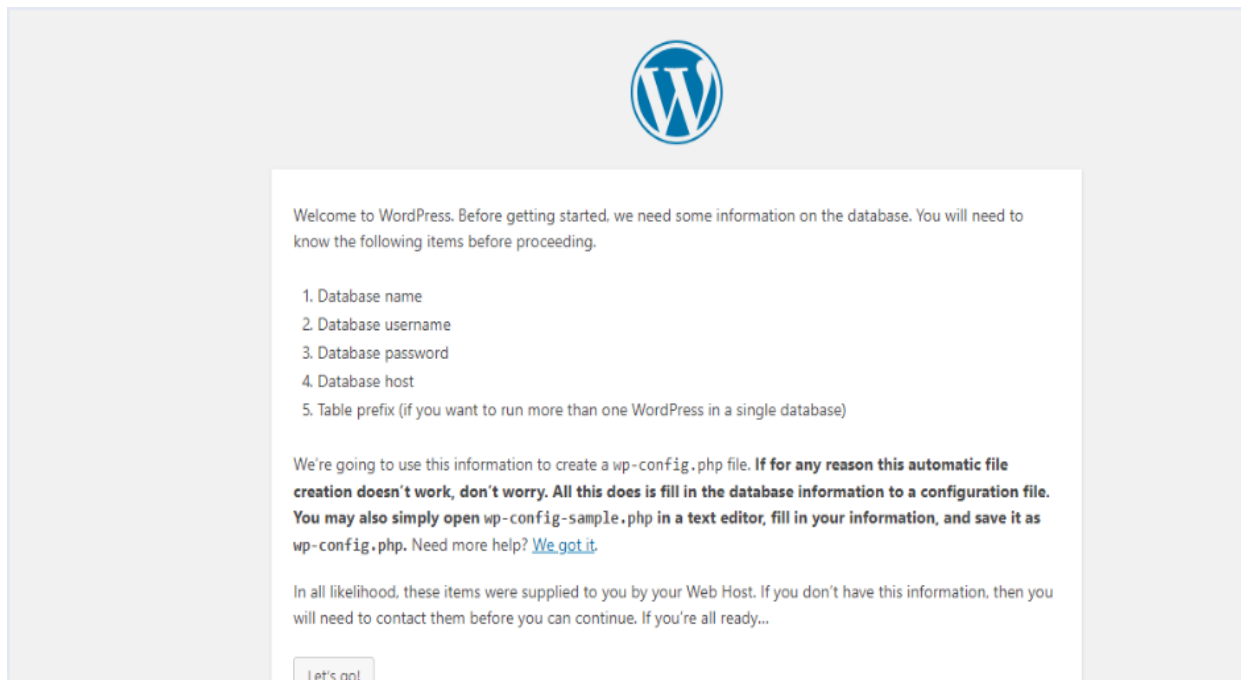
```
CustomLog ${APACHE_LOG_DIR}/wordpress_access.log combined
```

```
</VirtualHost>
```

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

<http://103.217.221.253/wordpress>

You'll be presented with a WordPress wizard and a list of credentials required to successfully set it up.



The image shows the first screen of the WordPress installation wizard. 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.

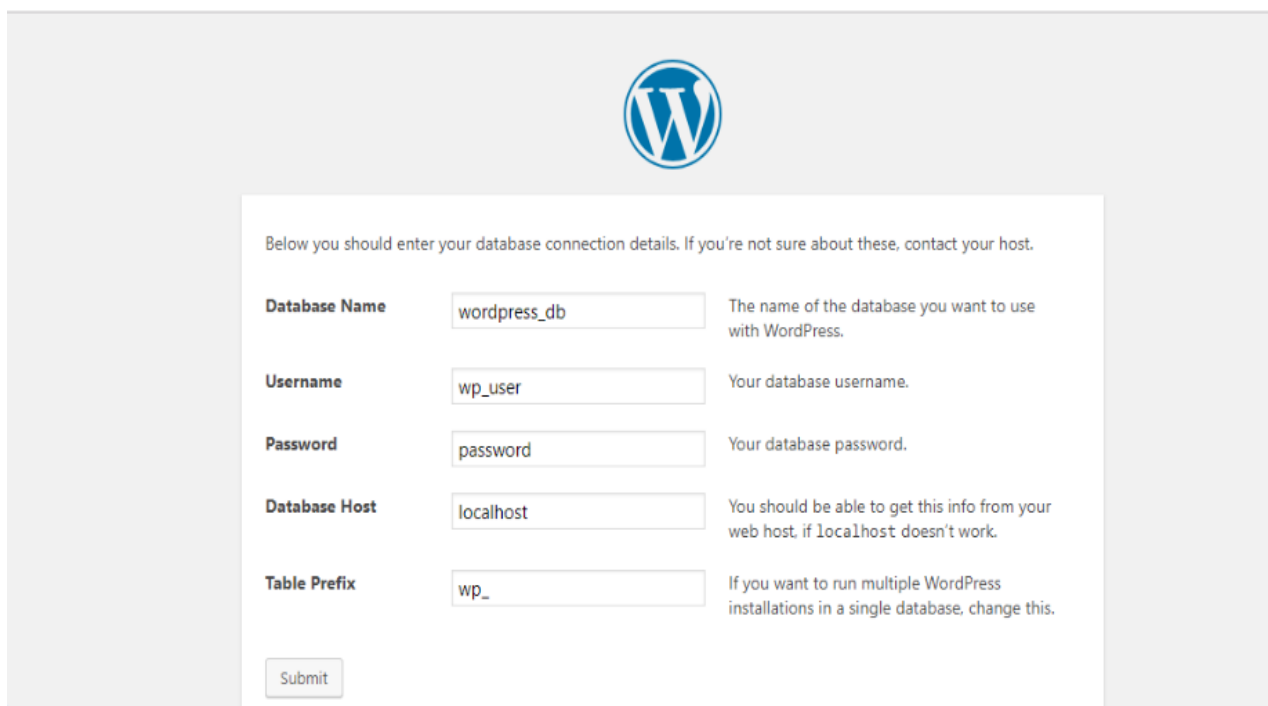
1. Database name
2. Database username
3. Database password
4. Database host
5. Table prefix (if you want to run more than one WordPress in a single database)

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...

At the bottom left of the white box is a button labeled "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 second screen of the WordPress installation wizard. At the top center is the WordPress logo. Below it, a white box contains the following text:

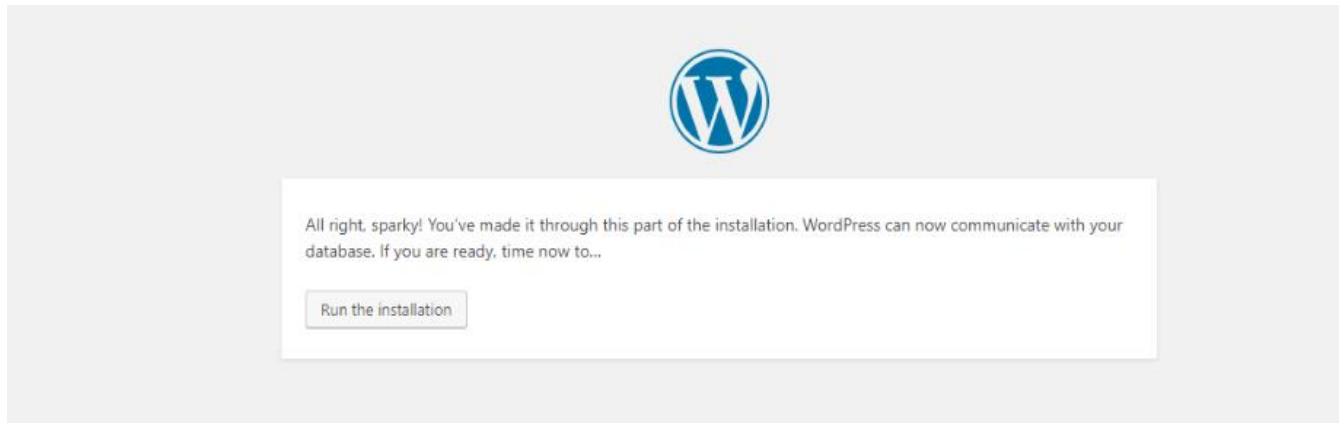
Below you should enter your database connection details. If you're not sure about these, contact your host.

The form consists of five rows, each with a label, an input field, and a description:

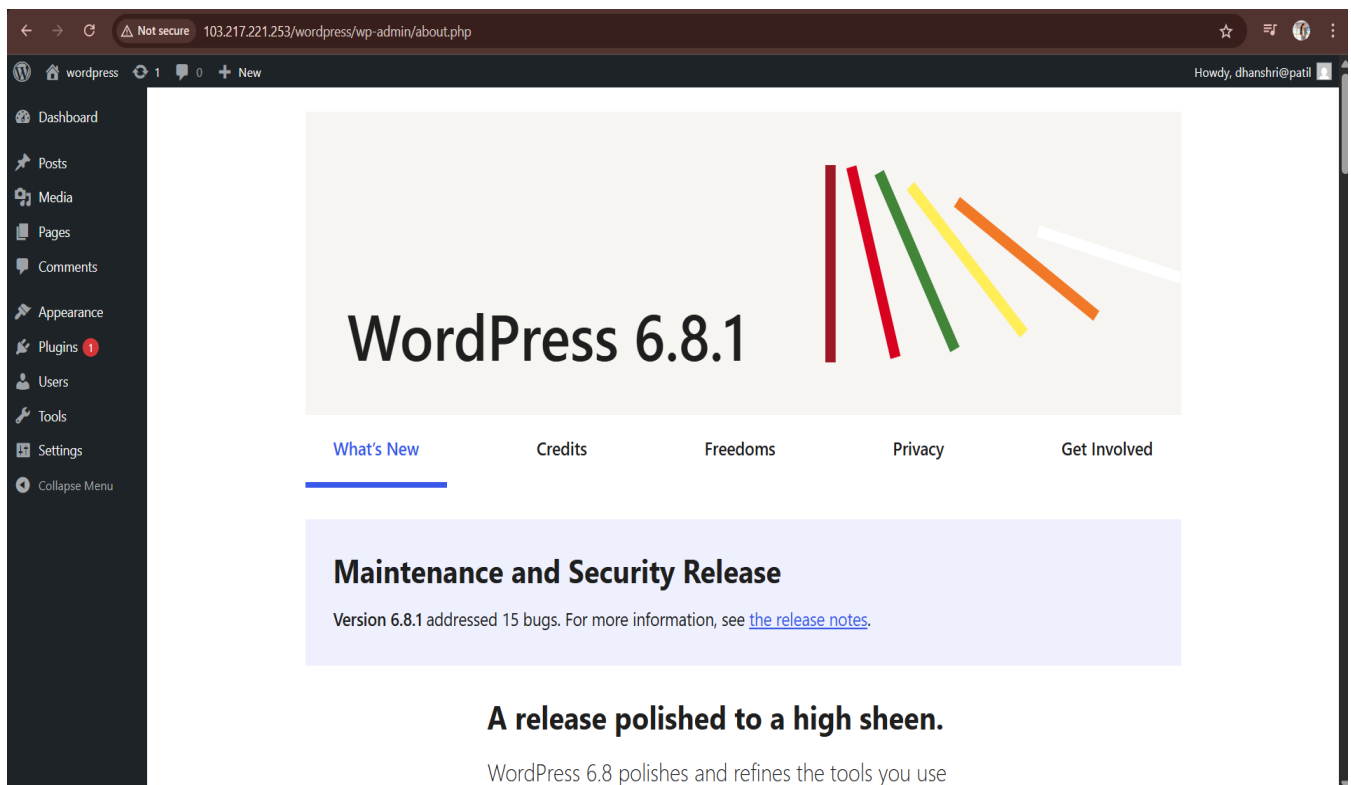
Label	Input Field	Description
Database Name	wordpress_db	The name of the database you want to use with WordPress.
Username	wp_user	Your database username.
Password	password	Your database password.
Database Host	localhost	You should be able to get this info from your web host, if localhost doesn't work.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

At the bottom left of the white box is a button labeled "Submit".

If all the details are correct, you will be given the green light to proceed. Run the installation.



Provide your login credentials and hit 'Login'.



5. Create Database On OsTicket

-- osTicket

Create a database for our osticket installation.

```
CREATE DATABASE osticket_db;
```

Next, create a database user for our osticket setup.

```
CREATE USER 'osticket_user'@'localhost' IDENTIFIED BY 'osticket_password';
```

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

```
GRANT ALL PRIVILEGES ON osticket_db.* TO 'osticket_user'@'localhost';
```

Great, now you can exit the database.

```
FLUSH PRIVILEGES;
```

```
EXIT;
```

OUTPUT:

```
ERROR 1064 (42000): You have an error in your SQL syntax; check the
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)]> █
```

Download and Install osTicket:

Change the directory to Apache web root and download the latest osTicket version inside that directory.

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

```
-sudo wget https://github.com/osTicket/osTicket/releases/download/v1.18/osTicket-v1.18.zip
```

Next, unzip the downloaded file.

```
-sudo unzip osTicket-v1.18.zip -d osticket
```

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

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

```
-sudo chmod -R 755 osticket
```

Apache Configuration (VirtualHost)

Create config:

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

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

```
ServerAdmin admin@103.217.221.253
```

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

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

```
Options Indexes FollowSymLinks
```

```
AllowOverride All
```

```
Require all granted
```

```
</Directory>
```

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

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

```
</VirtualHost>
```

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

<http://103.217.221.253/osticket>

Access osTicket Web UI

Now, open your web browser and access the osTicket using the URL <http://osticket.example.com>. You will see the osTicket prerequisites page.

Thank You for Choosing osTicket!

We are delighted you have chosen osTicket for your customer support ticketing system!

The installer will guide you every step of the way in the installation process. You're minutes away from your awesome customer support system!

Prerequisites

Before we begin, we'll check your server configuration to make sure you meet the minimum requirements to run the latest version of osTicket.

Required:

These items are necessary in order to install and use osTicket.

- ✓ PHP v8.1 or greater — (8.3.6)
- ✓ MySQLi extension for PHP — module loaded

Recommended:

You can use osTicket without these, but you may not be able to use all features.

- ✓ Gdlib Extension
- ✓ PHP ICONV Extension — Useful for email processing
- ✓ PHP IMAP Extension — Useful for email processing
- ✓ PHP CTYPE Extension — Required for email fetching
- ✓ PHP XML Extension — Required for XML API
- ✓ PHP XML-DOM Extension — Useful for HTML email processing
- ✓ PHP JSON Extension — Recommended for faster performance
- ✓ Mbstring Extension — Recommended for all installations
- ✓ Phar Extension — Recommended for plugins and language packs
- ✓ Intl Extension — Recommended for improved localization
- ✓ APCu Extension — Recommended for faster performance
- ✓ Zend OPcache Extension — Recommended for faster performance

Need Help?

If you are looking for a greater level of support, we provide [professional installation services](#) and commercial support with guaranteed response times, and access to the core development team. We can also help customize osTicket or even add new features to the system to meet your unique needs. [Learn More!](#)

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

Database Settings

Database connection information

MySQL Table Prefix:

MySQL Hostname:

MySQL Database:

MySQL Username:

MySQL Password:

[Install Now](#)

Need Help? We provide [professional installation services](#) and commercial support. [Learn More!](#)

Copyright © 2024 osTicket.com

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.

osTicket Installer
Support Ticket System

Installing osTicket v1.18.2
[Installation Guide](#) — [Get Professional Help](#) — [Contact Us](#)

Congratulations!

Your osTicket installation has been completed successfully. Your next step is to fully configure your new support ticket system for use, but before you get to it please take a minute to cleanup.

Config file permission:

Change permission of ost-config.php to remove write access as shown below.

- CLI:**
`chmod 0644 include/ost-config.php`
- Windows PowerShell:**
`icacls include/ost-config.php /reset`
- FTP:**
Using WS_FTP this would be right hand clicking on the file, selecting chmod, and then remove write access
- Cpanel:**
Click on the file, select change permission, and then remove write access.

Below, you'll find some useful links regarding your installation.

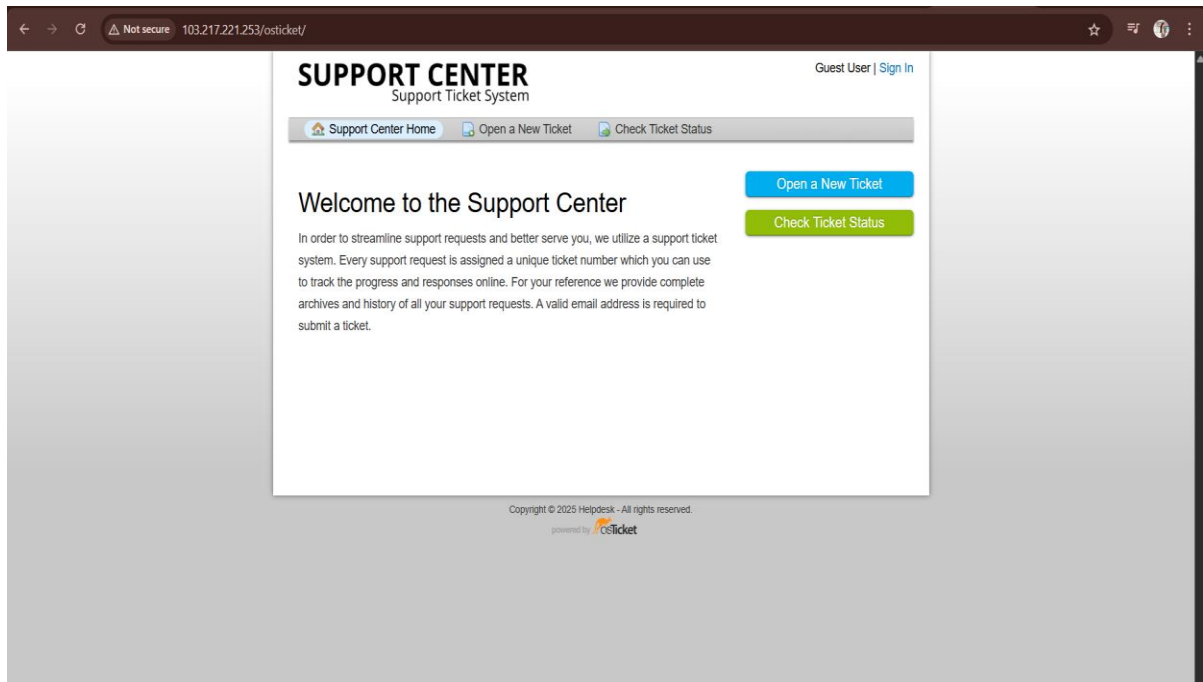
Your osTicket URL: http://osticket.example.com/	Your Staff Control Panel: http://osticket.example.com/scp
osTicket Forums: https://forum.osticket.com/	osTicket Documentation: https://docs.osticket.com/

PS: Don't just make customers happy, make happy customers!

What's Next?

Post-Install Setup: You can now log in to [Admin Panel](#) with the username and password you created during the install process. After a successful log in, you can proceed with post-install setup. For complete and upto date guide see [osTicket wiki](#)

Commercial Support Available: Don't let technical problems impact your osTicket implementation. Get guidance and hands-on expertise to address unique challenges and make sure your osTicket runs smoothly, efficiently, and securely. [Learn More!](#)



Create Database on OwnCloud

-- OwnCloud

Create a database for our owncloud installation.

```
CREATE DATABASE owncloud_db;
```

Next, create a database user for our owncloud setup

```
CREATE USER 'owncloud_user'@'localhost' IDENTIFIED BY 'owncloud_password';
```

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;
```

OUTPUT:

```
ERROR 1064 (42000): You have an error in your SQL syntax; check the
MariaDB [(none)]> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| oosticket_db |
| owncloud_db |
| performance_schema |
| sys |
| wordpress_database |
+-----+
7 rows in set (0.001 sec)

MariaDB [(none)]> █
```

Download and Install OwnCloud:

This is where Apache serves web files. You're moving here to install OwnCloud in this location

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

Brings the compressed OwnCloud software to your server.

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

Apache runs as `www-data`. This gives it permission to read/write the OwnCloud files for proper operation.

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

Ensures proper access permissions so Apache can serve the application without security issues.

```
-sudo chown -R www-data:www-data owncloud
-sudo chmod -R 755 owncloud
```

Apache Configuration (VirtualHost)

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

```
<VirtualHost *:80>
  ServerAdmin admin@103.217.221.253
  DocumentRoot /var/www/html/owncloud

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

    <FilesMatch \.php$>
      SetHandler "proxy:unix:/run/php/php7.4-fpm.sock|fcgi://localhost/"
    </FilesMatch>
  </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

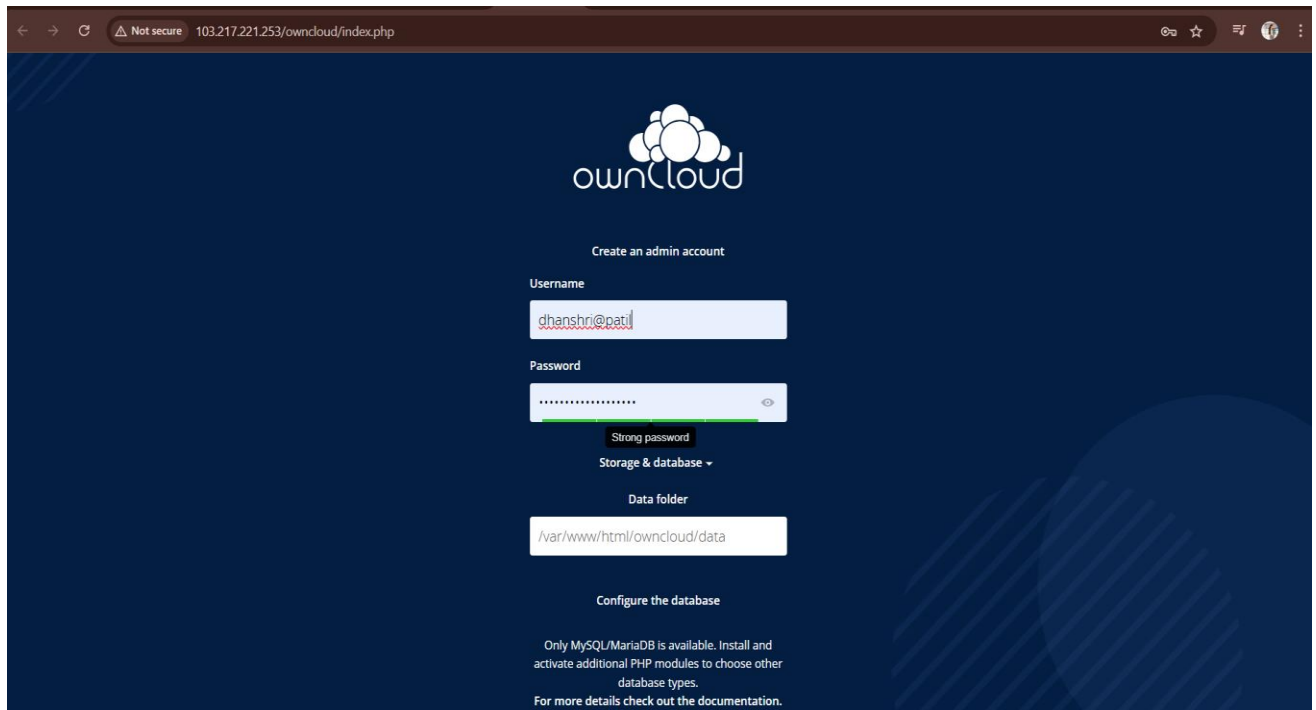
<http://103.217.221.253/owncloud>



Access osTicket Web UI :

You should see the ownCloud web configuration page in your browser.

Create an admin account by choosing a username and a password. For security purposes it is not recommended to use something like “admin” for the username:

**Enable All Configs :**

```
-sudo a2ensite wordpress.conf
-sudo a2ensite owncloud.conf
-sudo a2ensite osticket.conf
```

Reload Apache :

```
- sudo systemctl reload apache2
```

Apache VirtualHost Setup (Short Overview)

1. **Created separate config files** for each web app:
 - wordpress.conf
 - owncloud.conf
 - osticket.conf
2. **Defined VirtualHost settings** in each file:

- Set DocumentRoot to point to the correct folder (/var/www/html/...)
 - Allowed access and overrides (Require all granted, AllowOverride All)
 - For OwnCloud, added PHP handler with FastCGI.
3. **Enabled the new sites** using a2ensite.
 4. **Reloaded Apache** to apply changes.

