

Deploying a WordPress Server on AWS (Ubuntu EC2)

ASWIN VTK

1. Introduction

WordPress is one of the most popular open-source Content Management Systems (CMS) used to create websites and blogs.

In this mini project, we deploy **WordPress on an AWS EC2 instance using Ubuntu**, following industry-standard practices.

This project demonstrates:

- AWS EC2 provisioning
- LAMP stack installation
- Database configuration
- Secure WordPress deployment

Reference guide used: DigitalOcean WordPress on Ubuntu tutorial
(Implementation adapted for AWS environment)

2. Project Architecture

Components Used

- **Amazon Web Services (AWS)**
 - EC2 (Ubuntu Server 22.04)
 - Apache Web Server
 - MySQL Database Server
 - PHP
 - WordPress CMS
 - Security Group (Firewall)
-

3. Prerequisites

Before starting, ensure you have:

- An AWS account
- Basic Linux commands knowledge
- SSH client (Terminal / Git Bash / PuTTY)
- Key Pair (.pem file)

4. Step 1: Launch an EC2 Instance

4.1 Create EC2 Instance

1. Login to AWS Console
2. Navigate to **EC2 → Launch Instance**
3. Configure:
 - o **Name:** wordpress-server
 - o **AMI:** Ubuntu Server 22.04 LTS
 - o **Instance Type:** t2.micro (Free Tier)
 - o **Key Pair:** Create or select existing
 - o **Network:** Default VPC

4.2 Configure Security Group

Allow the following inbound rules:

Type	Port	Source
SSH	22	My IP
HTTP	80	Anywhere

5. Install WordPress on Ubuntu

Before we begin, let's update and upgrade the system. Login as the root user to your system and update the system to update the repositories.

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

Next, we are going to install the LAMP stack for WordPress to function. LAMP is short for Linux Apache MySQL and PHP.

Step 1: Install Apache

Let's jump right in and install Apache first. To do this, execute the following command.

```
apt install apache2
```

To confirm that Apache is installed on your system, execute the following command.

```
systemctl status apache2
```

Output

```
root@ip-172-31-15-253:/home/ubuntu# systemctl status apache2
● apache2.service - The Apache HTTP Server
  Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
  Active: active (running) since Thu 2026-02-05 08:01:33 UTC; 17s ago
    Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 2863 (apache2)
     Tasks: 55 (limit: 1121)
    Memory: 5.2M (peak: 5.4M)
       CPU: 31ms
      CGroup: /system.slice/apache2.service
              └─2863 /usr/sbin/apache2 -k start
                  ├─2865 /usr/sbin/apache2 -k start
                  ├─2866 /usr/sbin/apache2 -k start

Feb 05 08:01:33 ip-172-31-15-253 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Feb 05 08:01:33 ip-172-31-15-253 systemd[1]: Started apache2.service - The Apache HTTP Server.
root@ip-172-31-15-253:/home/ubuntu#
```

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

<https://ip-address>



Output

Step 2: 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
```

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

```
mysql_secure_installation
```

The first step will prompt you to change the root password to login to the database. You can opt to change it or skip if you are convinced that you have a strong password. To skip changing type n

Follow the steps and give Y/N

```
Enter current password for root (enter for none):  
OK, successfully used password, moving on...  
  
Setting the root password or using the unix_socket ensures that nobody  
can log into the MariaDB root user without the proper authorisation.  
  
You already have your root account protected, so you can safely answer 'n'  
  
Switch to unix_socket authentication [Y/n] y  
Enabled successfully!  
Reloading privilege tables..  
... Success!  
  
You already have your root account protected, so you can safely answer 'n'  
  
Change the root password? [Y/n] y  
New password:  
Re-enter new password:  
Password updated successfully!  
Reloading privilege tables..  
... Success!  
  
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  
... Success!  
  
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  
... Success!  
  
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] y  
- Dropping test database...  
... Success!  
- Removing privileges on test database...  
... Success!  
  
Reloading the privilege tables will ensure that all changes made so far  
will take effect immediately.  
  
Reload privilege tables now? [Y/n] y  
... Success!  
  
Cleaning up...
```

Step 3: Install PHP

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

```
apt install php php-mysql
```

Output

```
root@ip-172-31-15-253:/home/ubuntu# 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-php8.3 php-common php8.3 php8.3-cli php8.3-common php8.3-mysql php8.3-opcache php8.3-readline
Suggested packages:
 php-pear
The following NEW packages will be installed:
 libapache2-mod-php8.3 php php-common php-mysql php8.3 php8.3-cli php8.3-common php8.3-mysql php8.3-opcache php8.3-readline
0 upgraded, 10 newly installed, 0 to remove and 78 not upgraded.
Need to get 5047 kB of archives.
After this operation, 22.9 MB of additional disk space will be used.
```

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

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

Append the following lines:

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

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

<https://ip-address/info.php>

Output:

The screenshot shows a browser window with the URL 65.2.142.158/info.php. The page title is "PHP Version 8.3.6". It displays detailed PHP configuration information in a table format. Key entries include:

System	Linux ip-172-31-15-253 6.14.0-1018-aws #18~24.04.1-Ubuntu SMP Mon Nov 24 19:46:27 UTC 2025 x86_64
Build Date	Jan 7 2026 08:40:32
Build System	Linux
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/8.3/apache2
Loaded Configuration File	/etc/php/8.3/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/8.3/apache2/conf.d
Additional .ini files parsed	/etc/php/8.3/apache2/conf.d/10-mysqlind.ini, /etc/php/8.3/apache2/conf.d/10-opcache.ini, /etc/php/8.3/apache2/conf.d/10-pdo.ini, /etc/php/8.3/apache2/conf.d/20-calendar.ini, /etc/php/8.3/apache2/conf.d/20-crypt.ini, /etc/php/8.3/apache2/conf.d/20-exif.ini, /etc/php/8.3/apache2/conf.d/20-fil.ini, /etc/php/8.3/apache2/conf.d/20-fileinfo.ini, /etc/php/8.3/apache2/conf.d/20-ftp.ini, /etc/php/8.3/apache2/conf.d/20-gd.ini, /etc/php/8.3/apache2/conf.d/20-gettext.ini, /etc/php/8.3/apache2/conf.d/20-iconv.ini, /etc/php/8.3/apache2/conf.d/20-mbstring.ini, /etc/php/8.3/apache2/conf.d/20-phar.ini, /etc/php/8.3/apache2/conf.d/20-posix.ini, /etc/php/8.3/apache2/conf.d/20-readline.ini, /etc/php/8.3/apache2/conf.d/20-shmop.ini, /etc/php/8.3/apache2/conf.d/20-sockets.ini, /etc/php/8.3/apache2/conf.d/20-sysvmsg.ini, /etc/php/8.3/apache2/conf.d/20-sysvsem.ini, /etc/php/8.3/apache2/conf.d/20-sysvshm.ini, /etc/php/8.3/apache2/conf.d/20-tokenizer.ini
PHP API	20230831
PHP Extension	20230831
Zend Extension	420230831
Zend Extension Build	API420230831,NTS
PHP Extension Build	API20230831,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
Zend Max Execution Timers	disabled
IPv6 Support	enabled
DTrace Support	disabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2, tlsv1.3
Registered Stream Filters	zlib.*, string.rot13, string.toupper, string.tolower, convert*, consumed, dechunk, convert.iconv.*

This program makes use of the Zend Scripting Language Engine:
Zend Engine v4.3.6, Copyright (c) Zend Technologies with Zend OPcache v8.3.6, Copyright (c), by Zend Technologies

zendengine

Step 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
```

Output

```
root@wordpress:~# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 2
Server version: 10.1.37-MariaDB-0+deb9u1 Debian 9.6

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;
```

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;
```

Output

```
MariaDB [(none)]> CREATE DATABASE wordpress_db;
Query OK, 1 row affected (0.001 sec)

MariaDB [(none)]> CREATE USER 'wp_user'@'localhost' IDENTIFIED BY 'password';
Query OK, 0 rows affected (0.001 sec)

MariaDB [(none)]> GRANT ALL ON wordpress_db.* TO 'wp_user'@'localhost' IDENTIFIED BY 'password';
Query OK, 0 rows affected (0.002 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]>
MariaDB [(none)]> Exit;
Bye
root@ip-172-31-15-253:/home/ubuntu#
```

Exit;

Step 5: Install WordPress CMS

Go to your temp directory and download the latest WordPress File

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

Output

```
root@ip-172-31-15-253:/home/ubuntu# cd /tmp && wget https://wordpress.org/latest.tar.gz
--2026-02-05 08:10:42-- https://wordpress.org/latest.tar.gz
Resolving wordpress.org (wordpress.org)... 198.143.164.252, 2607:f978:5:8002::c68f:a4fc
Connecting to wordpress.org (wordpress.org)|198.143.164.252|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 27062929 (26M) [application/octet-stream]
Saving to: 'latest.tar.gz'

latest.tar.gz                                              100%[=====] 27062929  5.28 MB/s

2026-02-05 08:10:48 (5.28 MB/s) - 'latest.tar.gz' saved [27062929/27062929]
root@ip-172-31-15-253:/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.s

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

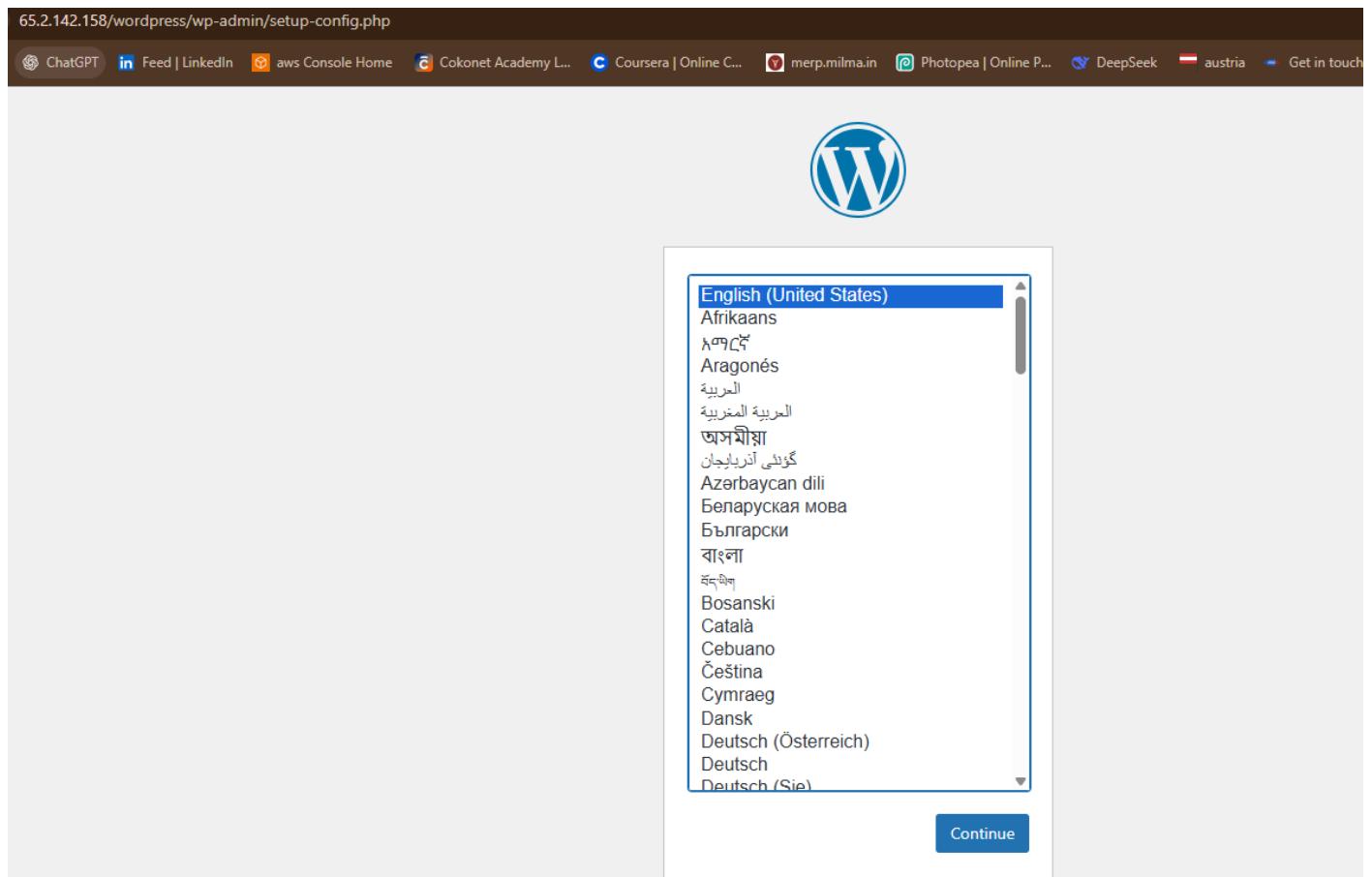
Finally, change permissions of ‘uploads’ directory.

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

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

<https://server-ip/wordpress>

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





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

Database Name	<input type="text" value="wordpress_db"/>	The name of the database you want to use with WordPress.
Username	<input type="text" value="wp_user"/>	Your database username.
Password	<input type="text" value="password"/>	Your database password.
Database Host	<input type="text" value="localhost"/>	You should be able to get this info from your web host, if localhost doesn't work.
Table Prefix	<input type="text" value="wp_"/>	If you want to run multiple WordPress installations in a single database, change this.



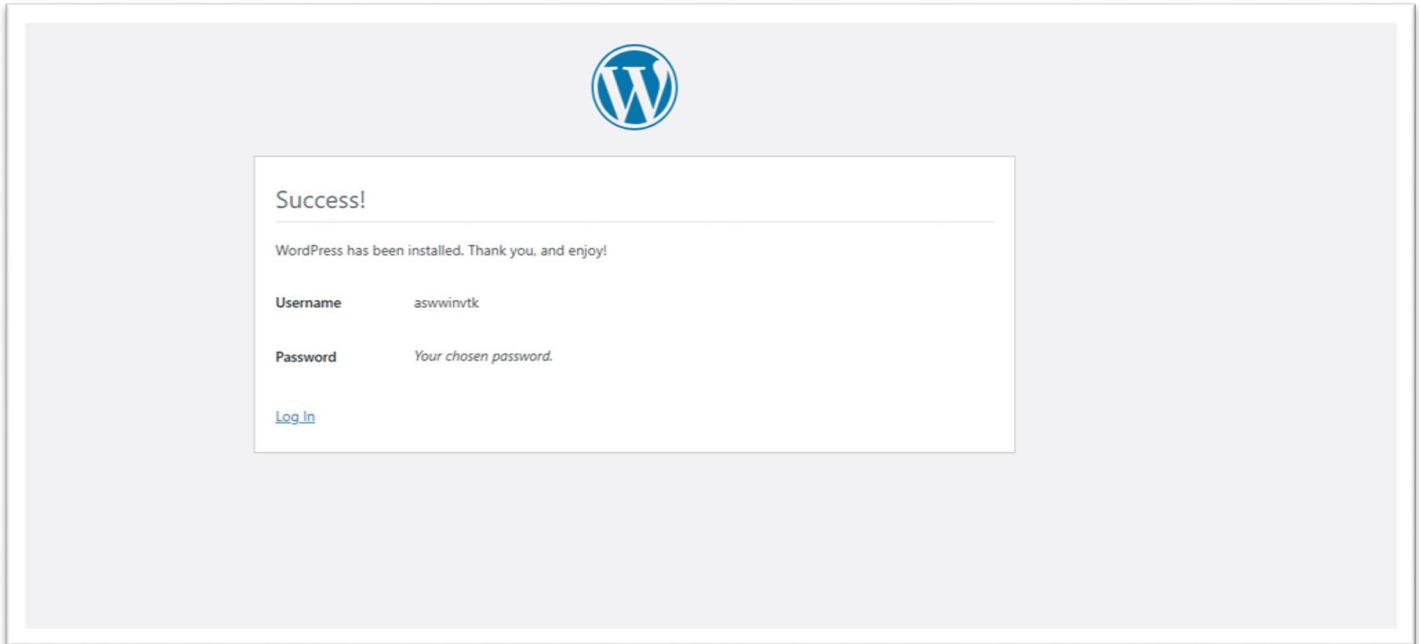
Welcome

Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Please provide the following information. Do not worry, you can always change these settings later.

Site Title	<input type="text" value="ASWIN WP PROJECT"/>
Username	<input type="text" value="aswwinvtk"/> Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.
Password	<input type="password" value="*****"/> <small>Medium</small> <input type="button" value="Show"/>
Important: You will need this password to log in. Please store it in a secure location.	
Your Email	<input type="text" value="aswinvtk97@gmail.com"/> Double-check your email address before continuing.
Search engine visibility	<input type="checkbox"/> Discourage search engines from indexing this site It is up to search engines to honor this request.



The image shows the WordPress dashboard. The URL in the browser bar is '65.2.142.158/wordpress/wp-admin/'. The dashboard sidebar on the left is titled 'Dashboard' and contains links for Home, Updates, Posts, Media, Pages, Comments, Appearance, Plugins, Users, Tools, Settings, and Collapse Menu. The main content area features a large white 'Welcome to WordPress!' heading, followed by the text 'Learn more about the 6.9.1 version.' Below this, there are two sections: one with a pencil icon and the text 'Author rich content with blocks and patterns', and another with a theme icon and the text 'Customize your entire site with block themes'. Both sections include descriptive text and a 'Read more' link.