

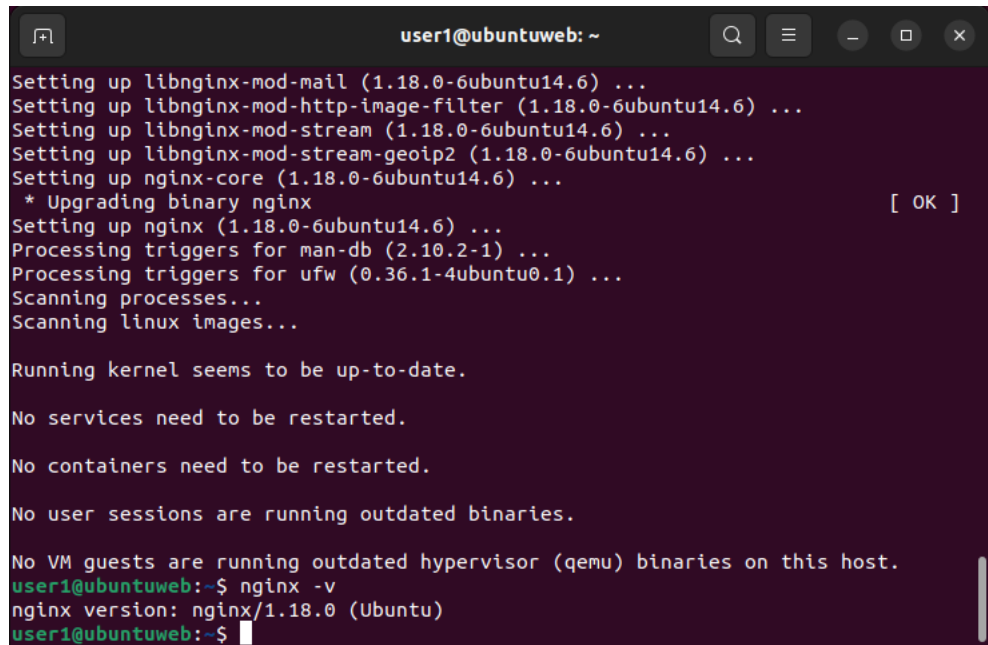
Ubuntu (WebServer) Setup – Nginx, PHP 8.2, MariaDB 10.6 and WordPress

1. Install Nginx

```
sudo apt update && sudo apt install nginx -y
```

Verify installation:

```
nginx -v
```

A terminal window titled 'user1@ubuntuweb: ~' showing the output of 'sudo apt install nginx -y'. The output lists the installation of various Nginx modules (libnginx-mod-mail, libnginx-mod-http-image-filter, libnginx-mod-stream, libnginx-mod-stream-geoip2) and the nginx-core package. It then shows the upgrade of the nginx binary, processing of triggers for man-db and ufw, and scanning of processes and linux images. The terminal concludes with 'Running kernel seems to be up-to-date.', 'No services need to be restarted.', 'No containers need to be restarted.', 'No user sessions are running outdated binaries.', and 'No VM guests are running outdated hypervisor (qemu) binaries on this host.' Finally, it shows the command 'user1@ubuntuweb:~\$ nginx -v' and its output 'nginx version: nginx/1.18.0 (Ubuntu)'.

```
user1@ubuntuweb: ~  
Setting up libnginx-mod-mail (1.18.0-6ubuntu14.6) ...  
Setting up libnginx-mod-http-image-filter (1.18.0-6ubuntu14.6) ...  
Setting up libnginx-mod-stream (1.18.0-6ubuntu14.6) ...  
Setting up libnginx-mod-stream-geoip2 (1.18.0-6ubuntu14.6) ...  
Setting up nginx-core (1.18.0-6ubuntu14.6) ...  
* Upgrading binary nginx [ OK ]  
Setting up nginx (1.18.0-6ubuntu14.6) ...  
Processing triggers for man-db (2.10.2-1) ...  
Processing triggers for ufw (0.36.1-4ubuntu0.1) ...  
Scanning processes...  
Scanning linux images...  
  
Running kernel seems to be up-to-date.  
  
No services need to be restarted.  
  
No containers need to be restarted.  
  
No user sessions are running outdated binaries.  
  
No VM guests are running outdated hypervisor (qemu) binaries on this host.  
user1@ubuntuweb:~$ nginx -v  
nginx version: nginx/1.18.0 (Ubuntu)  
user1@ubuntuweb:~$
```

2. Install PHP

```
sudo apt install php8.2 php8.2-fpm php8.2-mysql php8.2-curl  
php8.2-gd php8.2-mbstring php8.2-xml php8.2-zip -y  
sudo systemctl enable php8.2-fpm  
sudo systemctl start php8.2-fpm
```

If the above command doesn't work then the reason is this:
Ubuntu doesn't have PHP 8.2 by default so if the PHP installation with the
above command doesn't work then do the following.

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

Add PHP Repository:

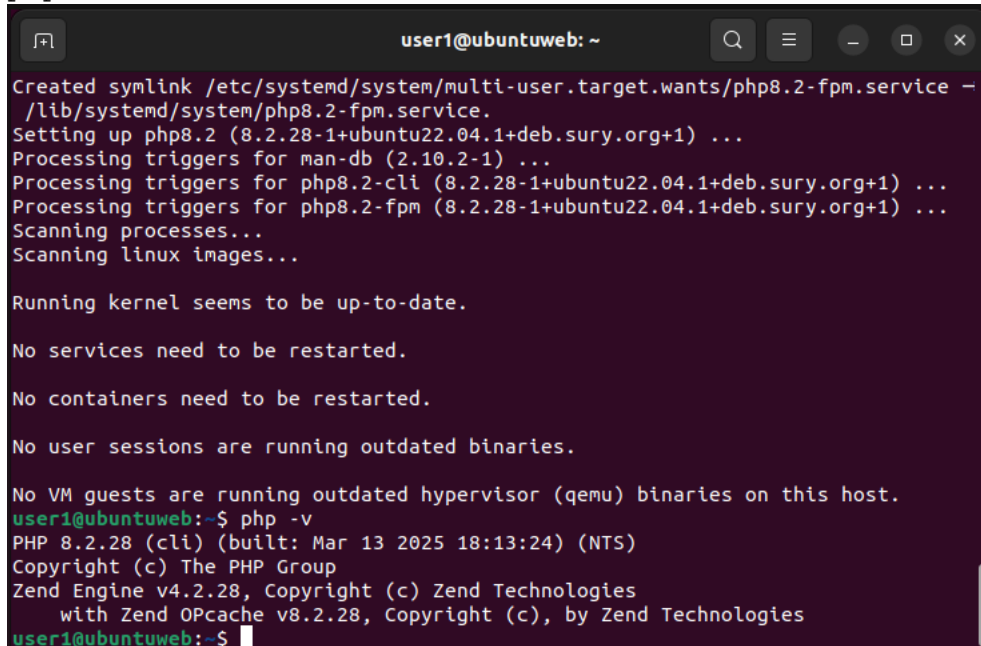
```
sudo add-apt-repository ppa:ondrej/php -y  
sudo apt update
```

Install PHP 8.2 and required extensions:

```
sudo apt install php8.2 php8.2-fpm php8.2-mysql php8.2-curl  
php8.2-gd php8.2-mbstring php8.2-xml php8.2-zip -y  
sudo systemctl enable php8.2-fpm  
sudo systemctl start php8.2-fpm
```

Verify Installation:

```
php -v
```



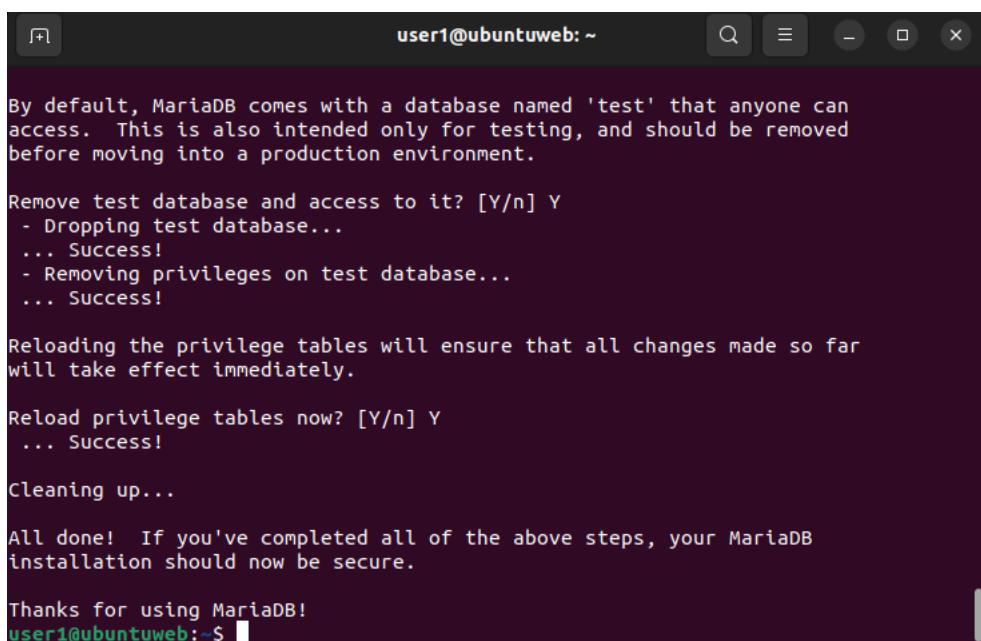
A terminal window titled 'user1@ubuntuweb: ~' showing the output of the 'php -v' command. The output includes system updates for php8.2-fpm.service, man-db, and php8.2-cli, followed by status messages about the kernel, services, containers, and VM guests. The final output shows the PHP version as 8.2.28 (cli) built on Mar 13 2025.

```
Created symlink /etc/systemd/system/multi-user.target.wants/php8.2-fpm.service -  
/lib/systemd/system/php8.2-fpm.service.  
Setting up php8.2 (8.2.28-1+ubuntu22.04.1+deb.sury.org+1) ...  
Processing triggers for man-db (2.10.2-1) ...  
Processing triggers for php8.2-cli (8.2.28-1+ubuntu22.04.1+deb.sury.org+1) ...  
Processing triggers for php8.2-fpm (8.2.28-1+ubuntu22.04.1+deb.sury.org+1) ...  
Scanning processes...  
Scanning linux images...  
  
Running kernel seems to be up-to-date.  
  
No services need to be restarted.  
  
No containers need to be restarted.  
  
No user sessions are running outdated binaries.  
  
No VM guests are running outdated hypervisor (qemu) binaries on this host.  
user1@ubuntuweb:~$ php -v  
PHP 8.2.28 (cli) (built: Mar 13 2025 18:13:24) (NTS)  
Copyright (c) The PHP Group  
Zend Engine v4.2.28, Copyright (c) Zend Technologies  
with Zend OPcache v8.2.28, Copyright (c), by Zend Technologies  
user1@ubuntuweb:~$
```

3. Install MariaDB:

```
sudo apt install mariadb-server mariadb-client -y  
sudo mysql_secure_installation
```

- Set the root password (if prompted).
- Remove anonymous users.
- Disallow remote root login.
- Remove test databases.
- Reload privileges.

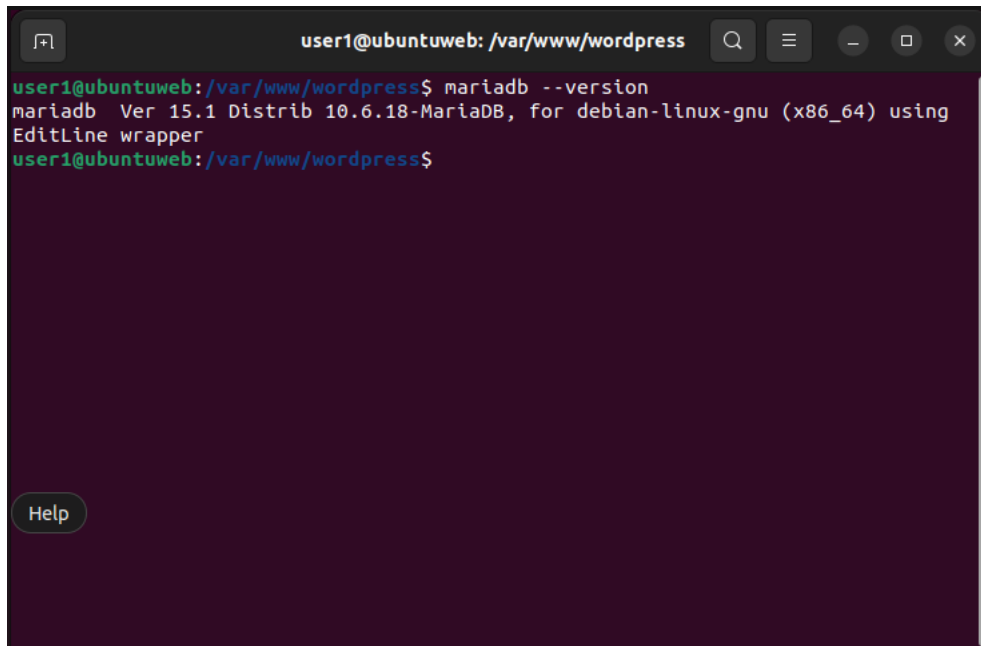


A terminal window titled 'user1@ubuntuweb: ~' showing the output of the 'mysql_secure_installation' command. It prompts to remove the test database and access to it, which is confirmed with 'Y'. It then prompts to reload privilege tables, which is also confirmed with 'Y'. The process concludes with a message that the installation should now be secure.

```
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...  
  
All done! If you've completed all of the above steps, your MariaDB  
installation should now be secure.  
  
Thanks for using MariaDB!  
user1@ubuntuweb:~$
```

Verify installation:

```
mariadb --version
```

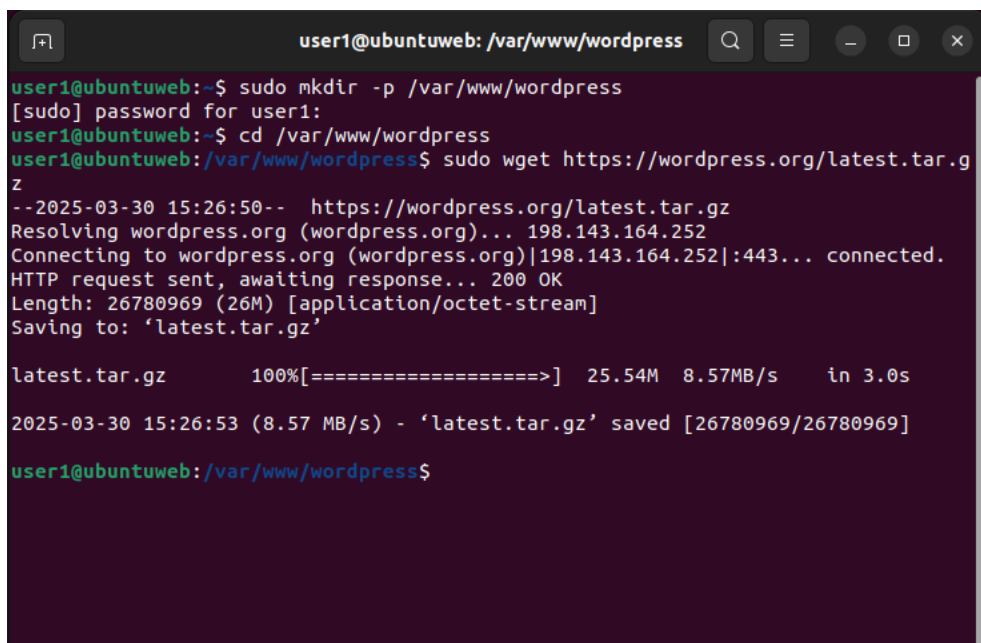
A terminal window titled 'user1@ubuntuweb: /var/www/wordpress' showing the command 'mariadb --version' and its output. The output is 'mariadb Ver 15.1 Distrib 10.6.18-MariaDB, for debian-linux-gnu (x86_64) using Editline wrapper'. The prompt returns to 'user1@ubuntuweb: /var/www/wordpress\$'. A 'Help' button is visible in the bottom left corner.

```
user1@ubuntuweb:/var/www/wordpress$ mariadb --version
mariadb Ver 15.1 Distrib 10.6.18-MariaDB, for debian-linux-gnu (x86_64) using
Editline wrapper
user1@ubuntuweb:/var/www/wordpress$
```

4. Install WordPress & Configure Database

a) Download WordPress:

```
sudo mkdir -p /var/www/wordpress
cd /var/www/wordpress
sudo wget https://wordpress.org/latest.tar.gz
```

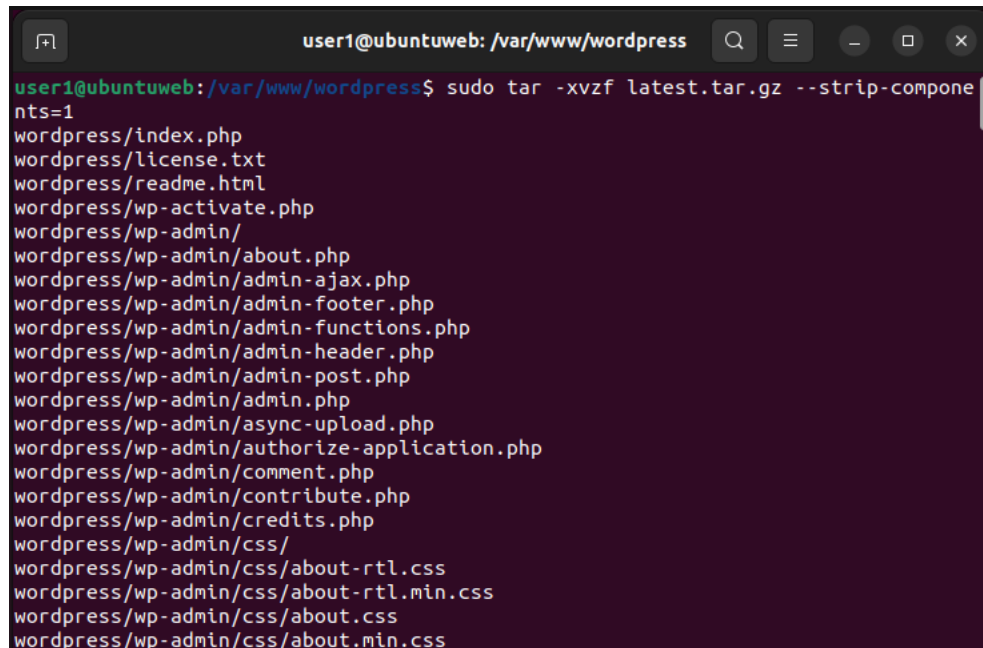
A terminal window titled 'user1@ubuntuweb: /var/www/wordpress' showing the steps to download WordPress. It includes 'mkdir', 'cd', and 'wget' commands. The 'wget' output shows the file being downloaded from 'https://wordpress.org/latest.tar.gz' at 8.57 MB/s. The prompt returns to 'user1@ubuntuweb: /var/www/wordpress\$'.

```
user1@ubuntuweb:~$ sudo mkdir -p /var/www/wordpress
[sudo] password for user1:
user1@ubuntuweb:~$ cd /var/www/wordpress
user1@ubuntuweb:/var/www/wordpress$ sudo wget https://wordpress.org/latest.tar.gz
z
--2025-03-30 15:26:50-- 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: 26780969 (26M) [application/octet-stream]
Saving to: 'latest.tar.gz'

latest.tar.gz      100%[=====>] 25.54M  8.57MB/s   in 3.0s
2025-03-30 15:26:53 (8.57 MB/s) - 'latest.tar.gz' saved [26780969/26780969]
user1@ubuntuweb:/var/www/wordpress$
```

b) Extract WordPress tar gzip file:

```
sudo tar -xvzf latest.tar.gz --strip-components=1
```

A terminal window titled 'user1@ubuntuweb: /var/www/wordpress' showing the execution of the command 'sudo tar -xvzf latest.tar.gz --strip-components=1'. The output lists various WordPress files and directories being extracted, including 'index.php', 'license.txt', 'readme.html', 'wp-activate.php', 'wp-admin/' directory, and several CSS files in the 'wp-admin/css/' directory.

```
user1@ubuntuweb: /var/www/wordpress$ sudo tar -xvzf latest.tar.gz --strip-components=1
wordpress/index.php
wordpress/license.txt
wordpress/readme.html
wordpress/wp-activate.php
wordpress/wp-admin/
wordpress/wp-admin/about.php
wordpress/wp-admin/admin-ajax.php
wordpress/wp-admin/admin-footer.php
wordpress/wp-admin/admin-functions.php
wordpress/wp-admin/admin-header.php
wordpress/wp-admin/admin-post.php
wordpress/wp-admin/admin.php
wordpress/wp-admin/async-upload.php
wordpress/wp-admin/authorize-application.php
wordpress/wp-admin/comment.php
wordpress/wp-admin/contribute.php
wordpress/wp-admin/credits.php
wordpress/wp-admin/css/
wordpress/wp-admin/css/about-rtl.css
wordpress/wp-admin/css/about-rtl.min.css
wordpress/wp-admin/css/about.css
wordpress/wp-admin/css/about.min.css
```

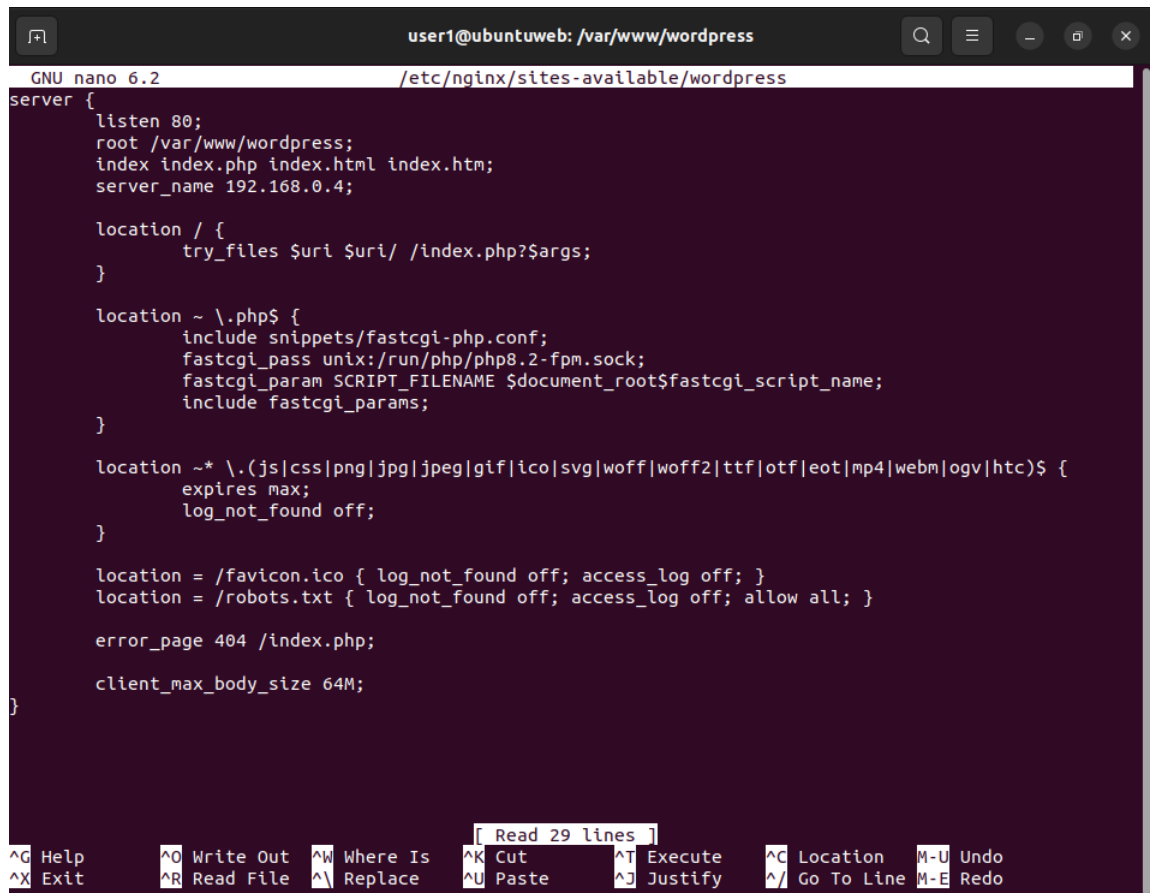
c) Change Owner/User Permissions:

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

5. Configure Nginx for WordPress

a) Create an nginx configuration file for wordpress:

```
sudo nano /etc/nginx/sites-available/wordpress
```



```
GNU nano 6.2 /etc/nginx/sites-available/wordpress
server {
    listen 80;
    root /var/www/wordpress;
    index index.php index.html index.htm;
    server_name 192.168.0.4;

    location / {
        try_files $uri $uri/ /index.php?$args;
    }

    location ~ \.php$ {
        include snippets/fastcgi-php.conf;
        fastcgi_pass unix:/run/php/php8.2-fpm.sock;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;
    }

    location ~* \.(js|css|png|jpg|jpeg|gif|ico|svg|woff|woff2|ttf|otf|eot|mp4|webm|ogv|htc)$ {
        expires max;
        log_not_found off;
    }

    location = /favicon.ico { log_not_found off; access_log off; }
    location = /robots.txt { log_not_found off; access_log off; allow all; }

    error_page 404 /index.php;

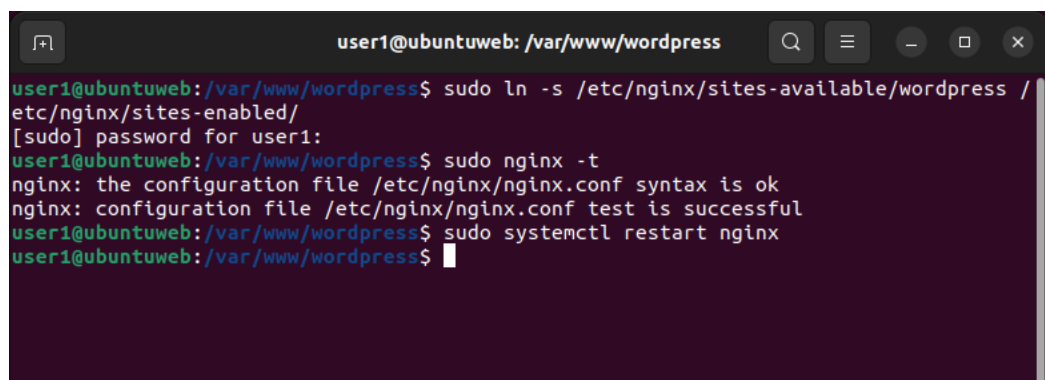
    client_max_body_size 64M;
}
```

b) Create symbolic link to enable the site:

```
sudo ln -s /etc/nginx/sites-available/wordpress
/etc/nginx/sites-enabled/
```

c) Test Nginx Configuration:

```
sudo nginx -t
sudo systemctl restart nginx
```



```
user1@ubuntuweb: /var/www/wordpress
user1@ubuntuweb:/var/www/wordpress$ sudo ln -s /etc/nginx/sites-available/wordpress /
etc/nginx/sites-enabled/
[sudo] password for user1:
user1@ubuntuweb:/var/www/wordpress$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
user1@ubuntuweb:/var/www/wordpress$ sudo systemctl restart nginx
user1@ubuntuweb:/var/www/wordpress$
```

6. Configure WordPress Database

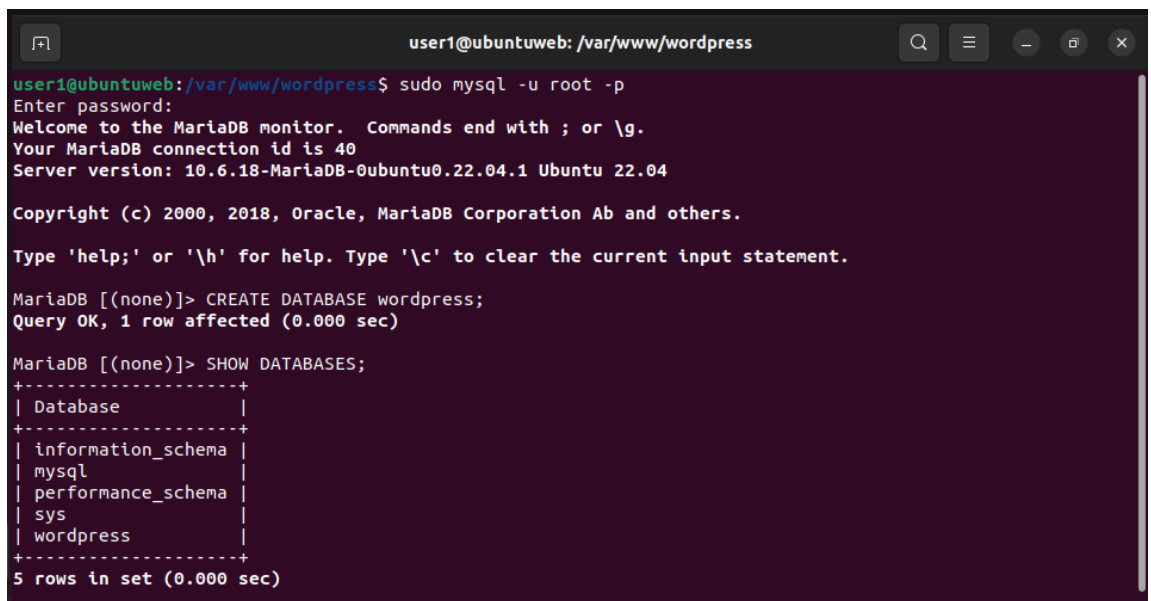
a) Login to MariaDB:

```
sudo mysql -u root -p
Enter password: [YOUR_PASSWORD]
```

b) Create a Database for WordPress:

```
CREATE DATABASE wordpress;
```

Check database: `SHOW DATABASES;`



```
user1@ubuntuweb: /var/www/wordpress
user1@ubuntuweb:/var/www/wordpress$ sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 40
Server version: 10.6.18-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 DATABASE wordpress;
Query OK, 1 row affected (0.000 sec)

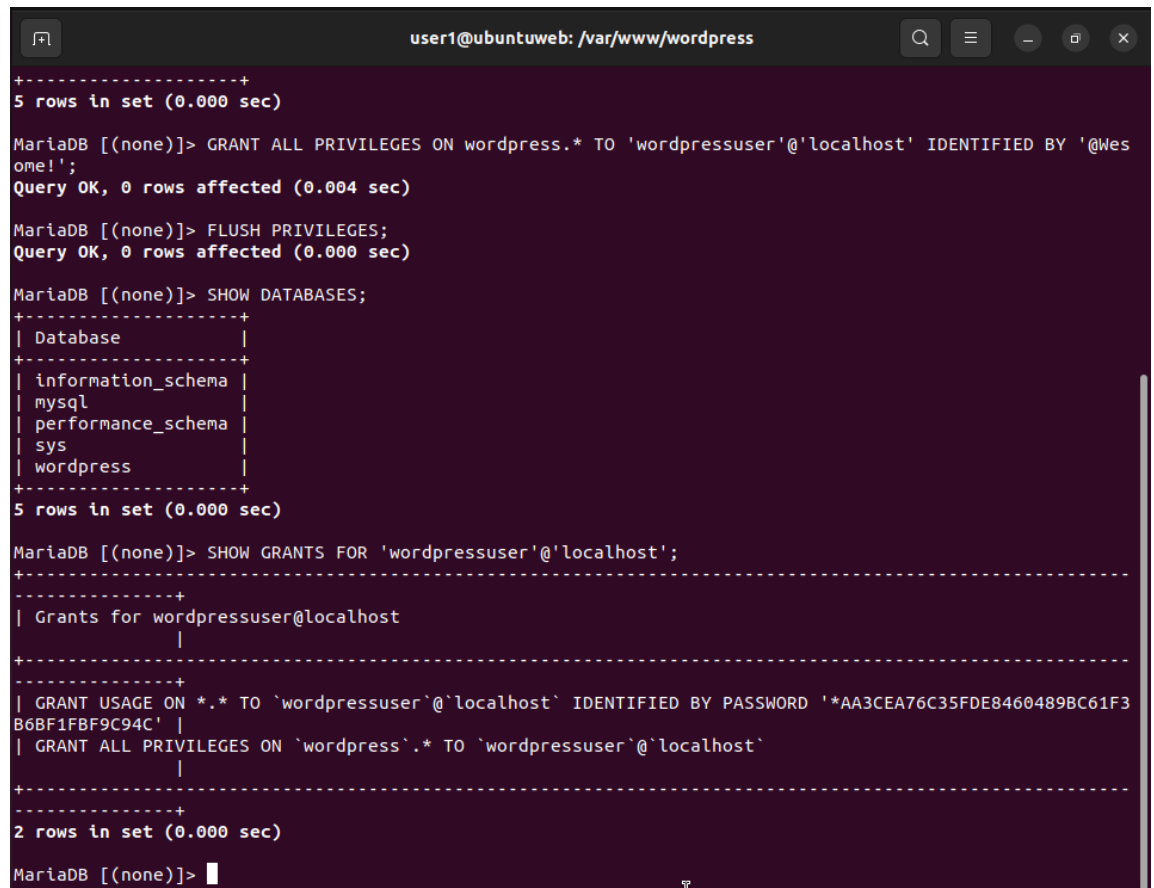
MariaDB [(none)]> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| wordpress |
+-----+
5 rows in set (0.000 sec)
```

c) Provide Grant access for WordPress admin user:

```
GRANT ALL PRIVILEGES ON wordpress.* TO
'wordpressuser'@'localhost' IDENTIFIED BY 'your_password';
FLUSH PRIVILEGES;
EXIT;
```

Check the grant:

```
SHOW GRANTS FOR 'wordpressuser'@'localhost';
EXIT;
```



```
user1@ubuntuweb: /var/www/wordpress
+-----+
5 rows in set (0.000 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON wordpress.* TO 'wordpressuser'@'localhost' IDENTIFIED BY '@Wes
ome!';
Query OK, 0 rows affected (0.004 sec)

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

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

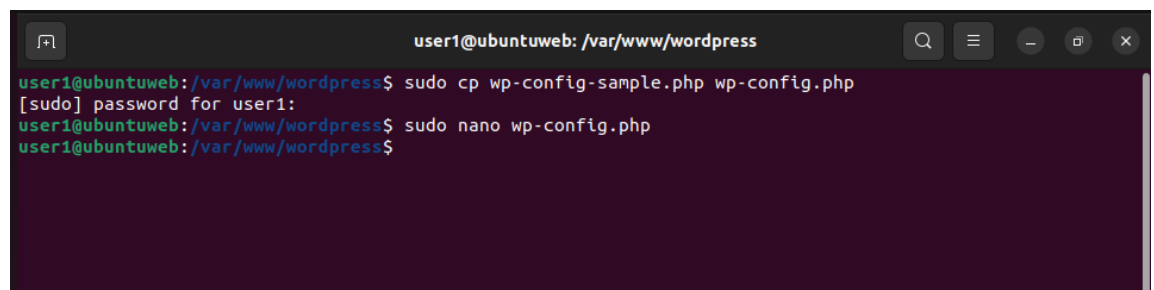
MariaDB [(none)]> SHOW GRANTS FOR 'wordpressuser'@'localhost';
+-----+
| Grants for wordpressuser@localhost |
+-----+
| GRANT USAGE ON *.* TO 'wordpressuser'@'localhost' IDENTIFIED BY PASSWORD '*AA3CEA76C35FDE8460489BC61F3
B6BF1FBF9C94C' |
| GRANT ALL PRIVILEGES ON `wordpress`.* TO 'wordpressuser'@'localhost' |
+-----+
2 rows in set (0.000 sec)

MariaDB [(none)]>
```

7. Configure WordPress:

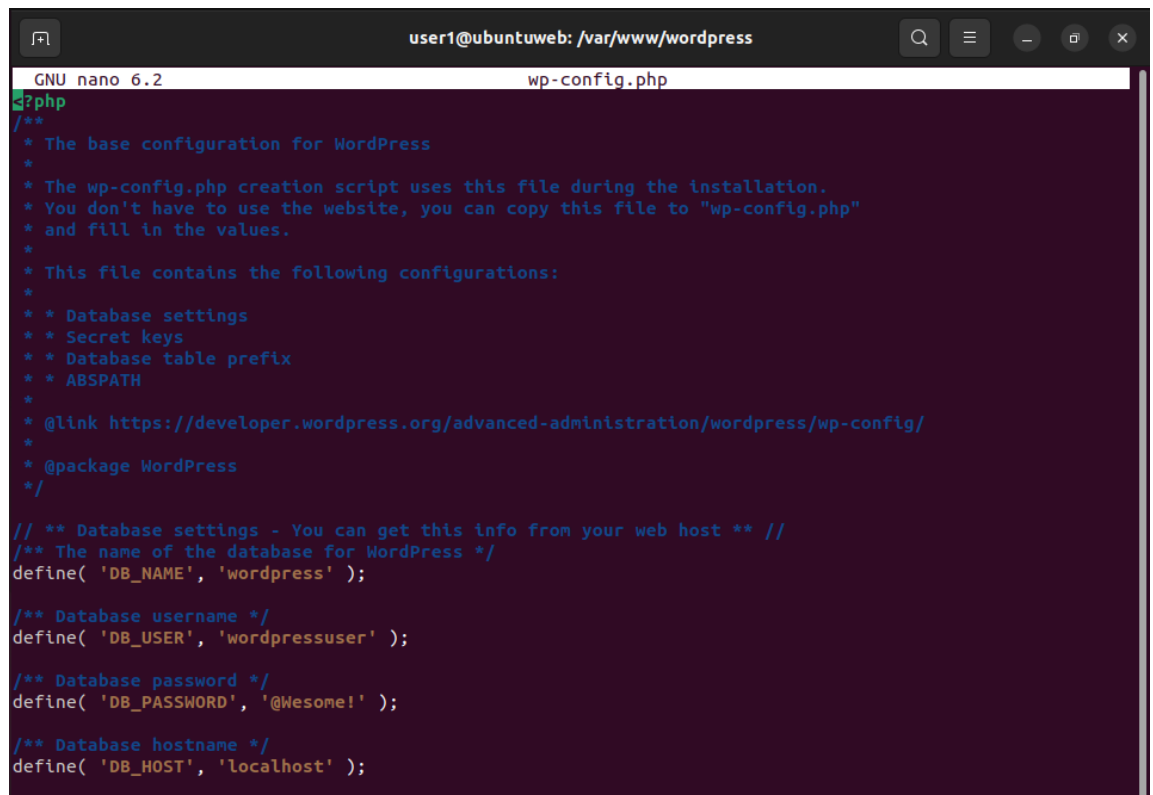
a) Rename and Edit WordPress Config File:

```
sudo cp wp-config-sample.php wp-config.php
```



```
user1@ubuntuweb: /var/www/wordpress
user1@ubuntuweb:/var/www/wordpress$ sudo cp wp-config-sample.php wp-config.php
[sudo] password for user1:
user1@ubuntuweb:/var/www/wordpress$ sudo nano wp-config.php
user1@ubuntuweb:/var/www/wordpress$
```

```
sudo nano wp-config.php
define('DB_NAME', 'wordpress');
define('DB_USER', 'wordpressuser');
define('DB_PASSWORD', 'StrongPassword');
define('DB_HOST', 'localhost');
```



```
GNU nano 6.2 wp-config.php
?php
/**
 * The base configuration for WordPress
 *
 * The wp-config.php creation script uses this file during the installation.
 * You don't have to use the website, you can copy this file to "wp-config.php"
 * and fill in the values.
 *
 * This file contains the following configurations:
 *
 * * Database settings
 * * Secret keys
 * * Database table prefix
 * * ABSPATH
 *
 * @link https://developer.wordpress.org/advanced-administration/wordpress/wp-config/
 *
 * @package WordPress
 */

// ** 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', '@Wesome!' );

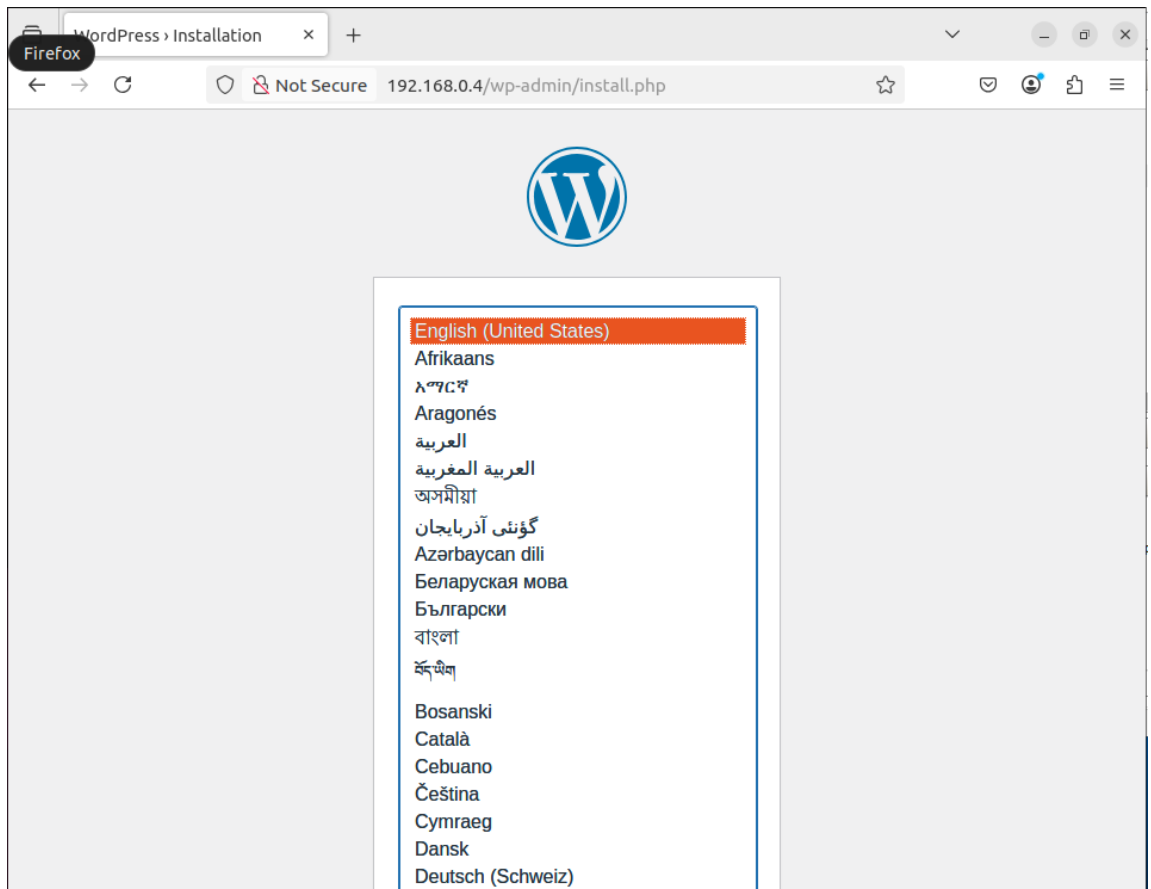
/** Database hostname */
define( 'DB_HOST', 'localhost' );
```


b) Finalize WordPress Installation:

Open Firefox browser,

Go to **http://[server_ip]/wp-admin/install.php**

Select Language > **English** > Next



Create admin username, password and email

WordPress › Installation

Not Secure 192.168.0.4/wp-admin/install.php?step=1

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 OzCazual WordPress Sandbox

Username user66
Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.

Password [Show](#)
Weak

Important: You will need this password to log in. Please store it in a secure location.

Confirm ☒ Confirm use of weak password

[Show Applications](#)

Check Discourage search engines from indexing this site > Install WordPress

WordPress › Installation

Not Secure 192.168.0.4/wp-admin/install.php?step=1

Information needed

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

Site Title OzCazual WordPress Sandbox

Username user66
Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.

Password [Show](#)
Weak

Important: You will need this password to log in. Please store it in a secure location.

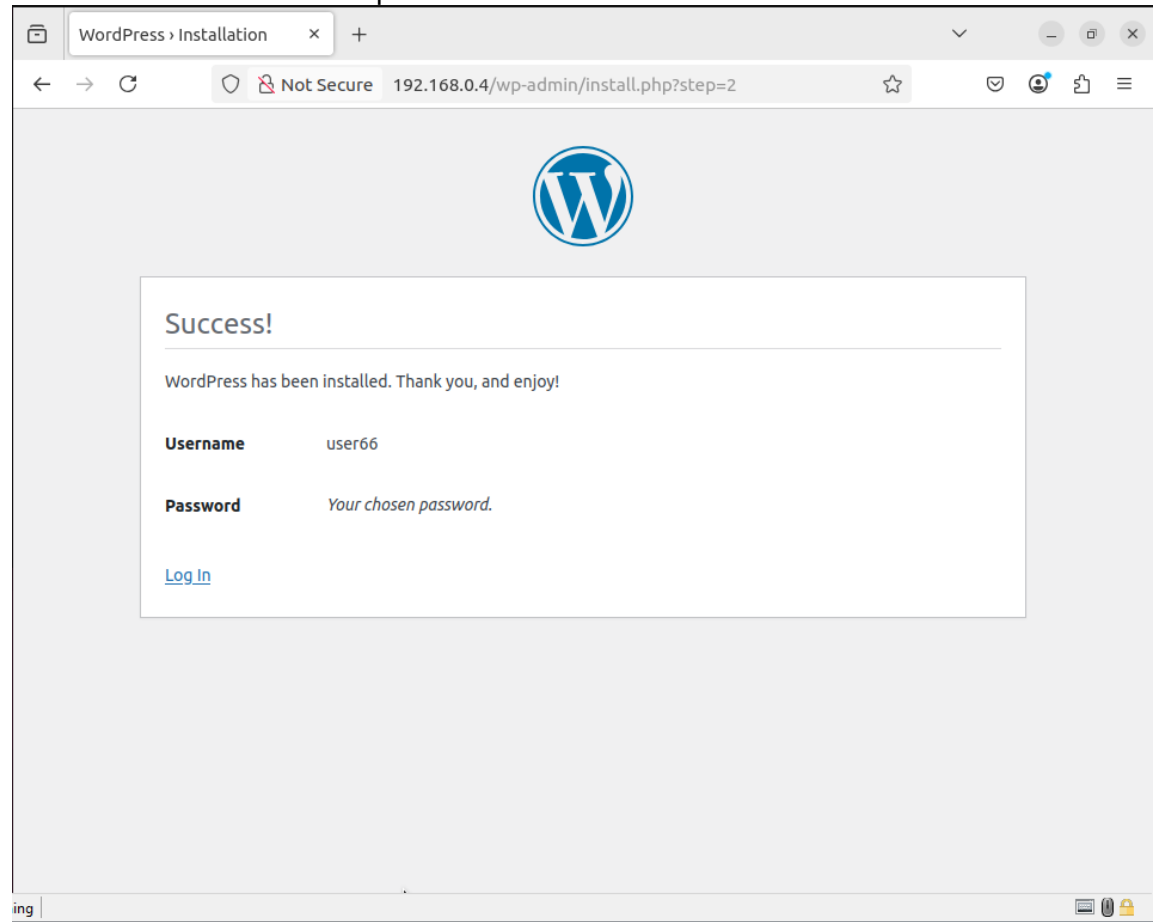
Confirm Password ☒ Confirm use of weak password

Your Email user66@gmail.com
Double-check your email address before continuing.

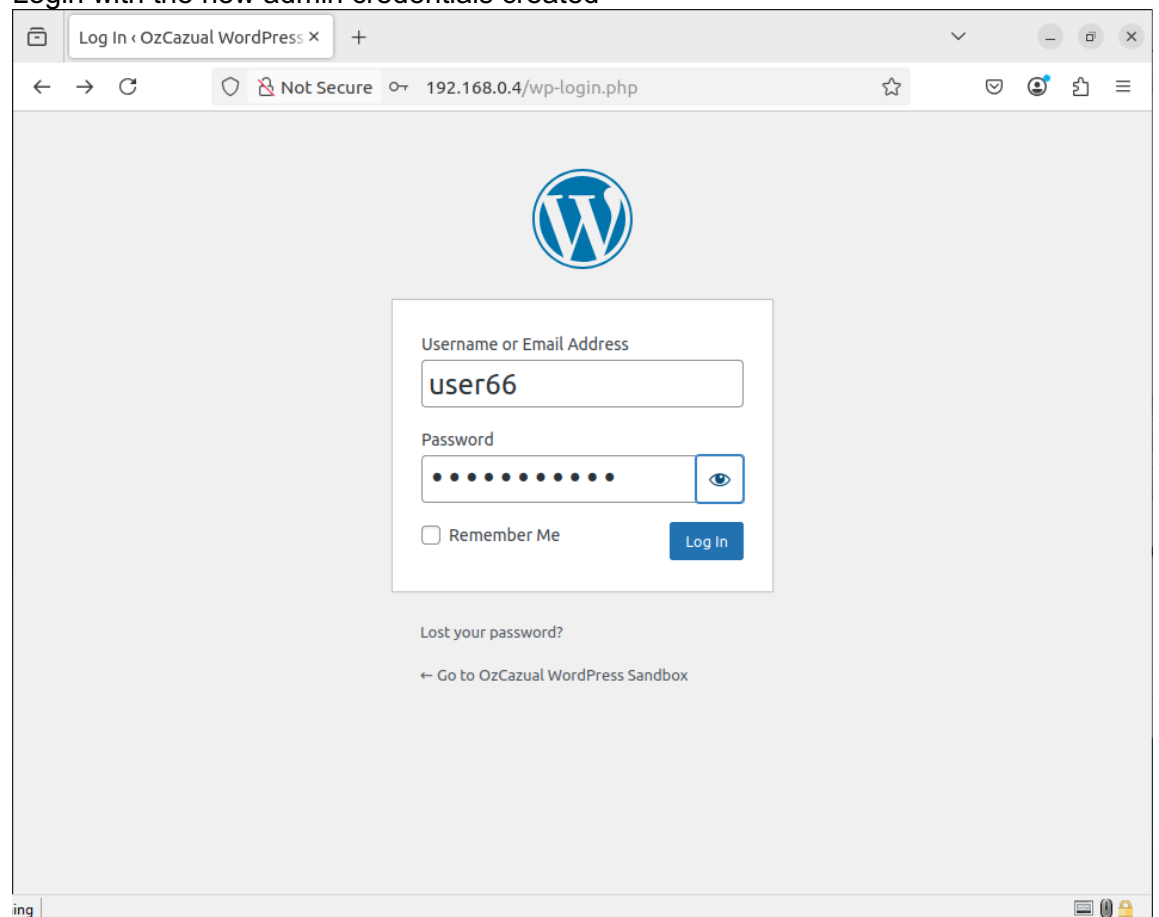
Search engine visibility ☒ Discourage search engines from indexing this site
It is up to search engines to honor this request.

[Install WordPress](#)

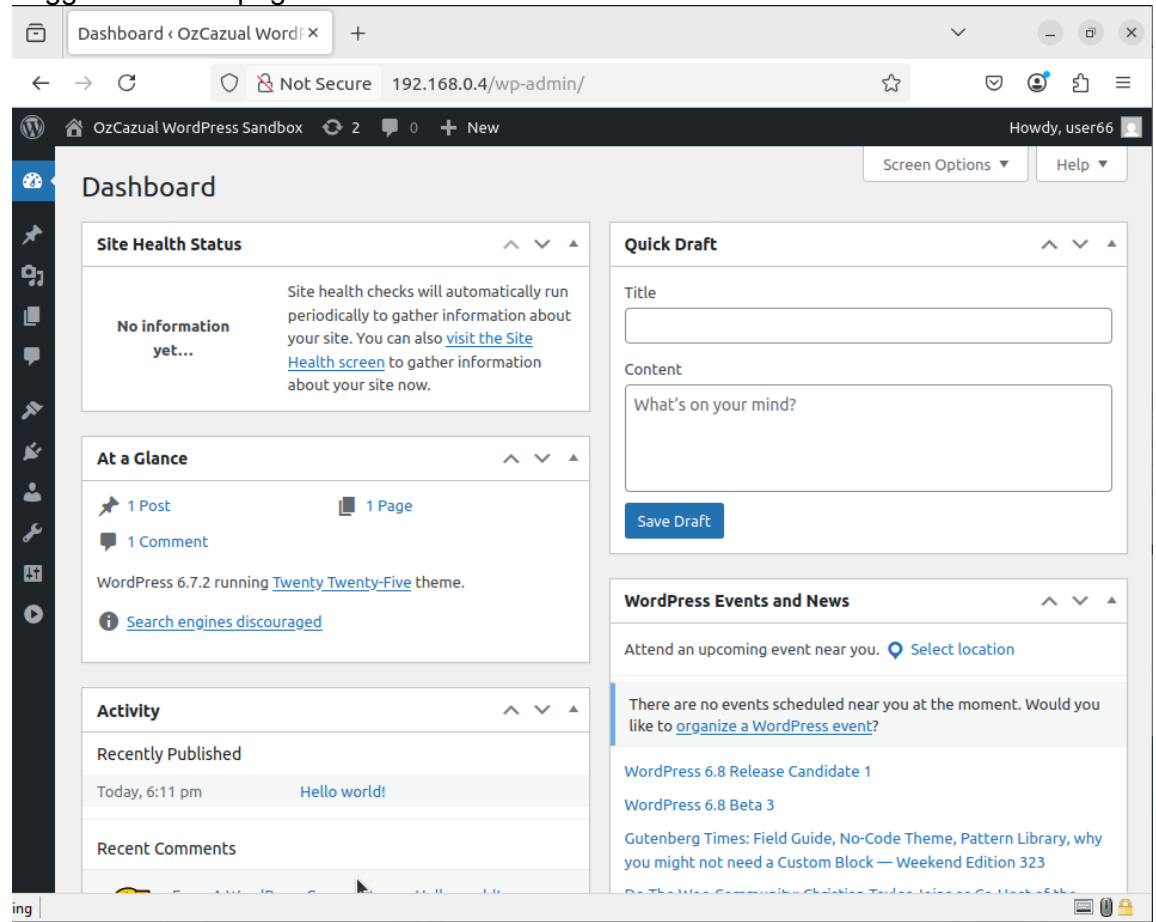
WordPress Installation Complete



Login with the new admin credentials created



Logged in admin page



If you changed your IP after connecting to pfSense,
Say like this,

```
sudo nano /etc/netplan/50-cloud-init.yaml
```

```
GNU nano 6.2 /etc/netplan/50-cloud-init.yaml
network:
  ethernets:
    eth0:
      dhcp4: no
      addresses:
        - 192.168.1.20/24
      routes:
        - to: 0.0.0.0/0
          via: 192.168.1.1
      nameservers:
        addresses:
          - 192.168.1.1
          - 8.8.8.8
          - 8.8.4.4
      version: 2
```

```
sudo netplan try
sudo netplan apply
```

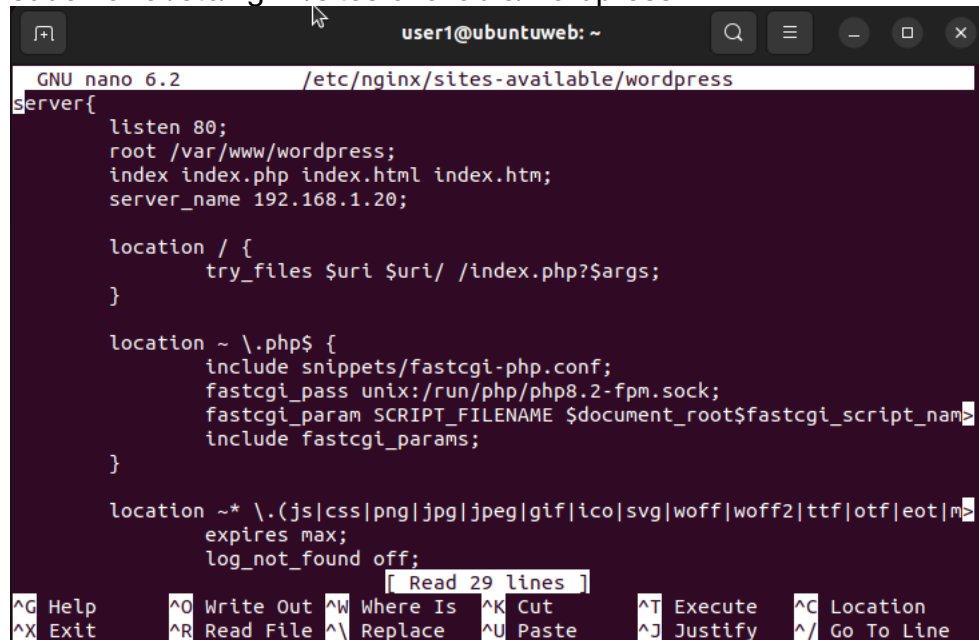
sudo reboot

Check:

ip a

Configure the nginx to change the server name:

sudo nano /etc/nginx/sites-available/wordpress



```
GNU nano 6.2 /etc/nginx/sites-available/wordpress
server{
    listen 80;
    root /var/www/wordpress;
    index index.php index.html index.htm;
    server_name 192.168.1.20;

    location / {
        try_files $uri $uri/ /index.php?$args;
    }

    location ~ \.php$ {
        include snippets/fastcgi-php.conf;
        fastcgi_pass unix:/run/php/php8.2-fpm.sock;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;
    }

    location ~* \.(js|css|png|jpg|jpeg|gif|ico|svg|woff|woff2|ttf|otf|eot|m
    expires max;
    log_not_found off;
}
Read 29 lines
^G Help    ^O Write Out  ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit    ^R Read File  ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line
```

sudo nginx -t

sudo systemctl reload nginx

Check if you have wp-cli installed:

wp --info

If not installed then install it:

curl -O <https://raw.githubusercontent.com/wp-cli/builds/gh-pages/phar/wp-cli.phar>

chmod +x wp-cli.phar

sudo mv wp-cli.phar /usr/local/bin/wp

wp --info

Make changes in the wordpress tables using mariadb:

sudo mysql -u root -p

[password]

SHOW DATABASES;

USE wordpress;

SHOW TABLES;

UPDATE wp_options SET option_value = 'http://<new_IP>' WHERE option_name = 'siteurl';

UPDATE wp_options SET option_value = 'http://<new_IP>' WHERE option_name = 'home';

FLUSH PRIVILEGES;

EXIT;

If it still shows old_IP in any links of wordpress website, then search the database tables like this:

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

```
[password]
```

```
USE wordpress;
```

```
SELECT * FROM wp_options WHERE option_value LIKE '%<old_IP>%';
```

```
SELECT * FROM wp_posts WHERE post_content LIKE '%<old_IP>%';
```

```
SELECT * FROM wp_postmeta WHERE meta_value LIKE '%<old_IP>%';
```

```
SELECT * FROM wp_usermeta WHERE meta_value LIKE '%<old_IP>%';
```

```
EXIT;
```

After updating the wordpress database tables, if you still see your old_IP in any of the wordpress links in the website:

```
wp search-replace 'http://<old_IP>' 'http://<new_IP>' --all-tables
```