NETWORKING & SYSTEM ADMINISTRATION LAB

Name: Manya Madhu

Roll No: 17

Batch: S2 RMCA B

Date: 21-03-22

Experiment No.: 5

Aim

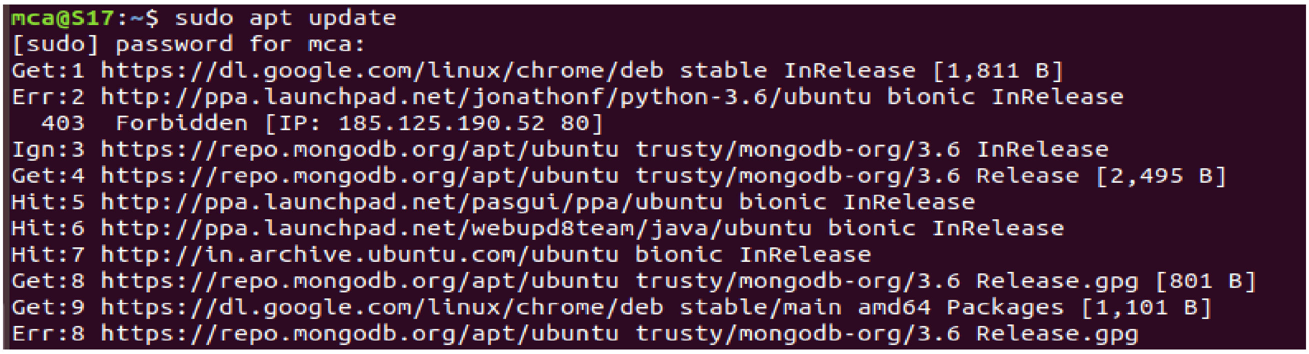
Install WordPress with LAMP on Ubuntu 18.04.

Procedure

Step 1 — Installing Apache and Updating the Firewall

First, make sure your apt cache is updated with:

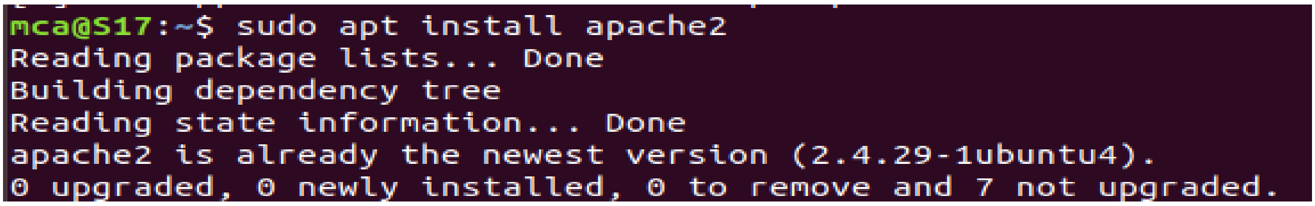
- sudo apt update



```
mca@S17:~$ sudo apt update
[sudo] password for mca:
Get:1 https://dl.google.com/linux/chrome/deb stable InRelease [1,811 B]
Err:2 http://ppa.launchpad.net/jonathonf/python-3.6/ubuntu bionic InRelease
  403  Forbidden [IP: 185.125.190.52 80]
Ign:3 https://repo.mongodb.org/apt/ubuntu trusty/mongodb-org/3.6 InRelease
Get:4 https://repo.mongodb.org/apt/ubuntu trusty/mongodb-org/3.6 Release [2,495 B]
Hit:5 http://ppa.launchpad.net/pasgui/ppa/ubuntu bionic InRelease
Hit:6 http://ppa.launchpad.net/webupd8team/java/ubuntu bionic InRelease
Hit:7 http://in.archive.ubuntu.com/ubuntu bionic InRelease
Get:8 https://repo.mongodb.org/apt/ubuntu trusty/mongodb-org/3.6 Release.gpg [801 B]
Get:9 https://dl.google.com/linux/chrome/deb stable/main amd64 Packages [1,101 B]
Err:8 https://repo.mongodb.org/apt/ubuntu trusty/mongodb-org/3.6 Release.gpg
```

Once the cache has been updated, you can install Apache with:

- sudo apt install apache2

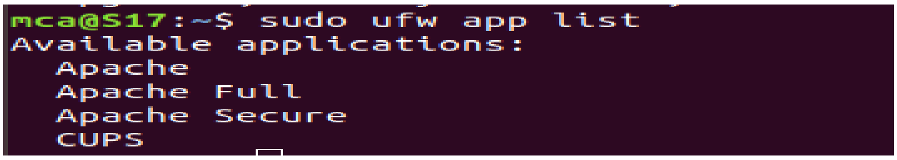


```
mca@S17:~$ sudo apt install apache2
Reading package lists... Done
Building dependency tree
Reading state information... Done
apache2 is already the newest version (2.4.29-1ubuntu4).
0 upgraded, 0 newly installed, 0 to remove and 7 not upgraded.
```

After entering this command, apt will tell you which packages it plans to install and how much extra disk space they'll take up. Press Y and hit ENTER to confirm, and the installation will proceed.

Adjust the Firewall to Allow Web Traffic Next, assuming that you have followed the initial server setup instructions and enabled the UFW firewall, make sure that your firewall allows HTTP and HTTPS traffic. You can check that UFW has an application profile for Apache like so:

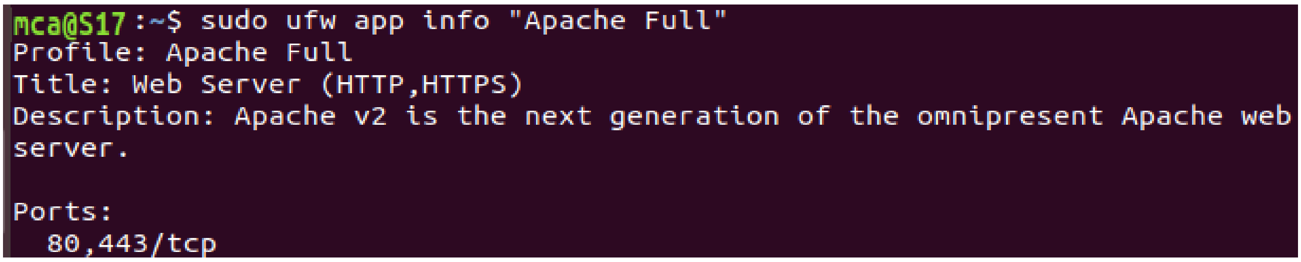
- sudo ufw app list



```
mca@S17:~$ sudo ufw app list
Available applications:
  Apache
  Apache Full
  Apache Secure
  CUPS
```

If you look at the Apache Full profile details, you'll see that it enables traffic to ports 80 and 443:

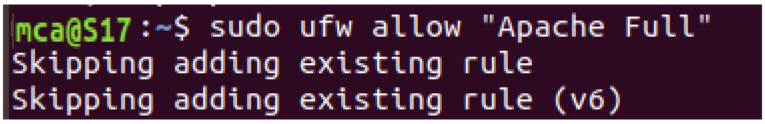
- sudo ufw app info "Apache Full"



```
mca@S17:~$ sudo ufw app info "Apache Full"
Profile: Apache Full
Title: Web Server (HTTP,HTTPS)
Description: Apache v2 is the next generation of the omnipresent Apache web
server.

Ports:
  80,443/tcp
```

To allow incoming HTTP and HTTPS traffic for this server, run:

- sudo ufw allow "Apache Full"

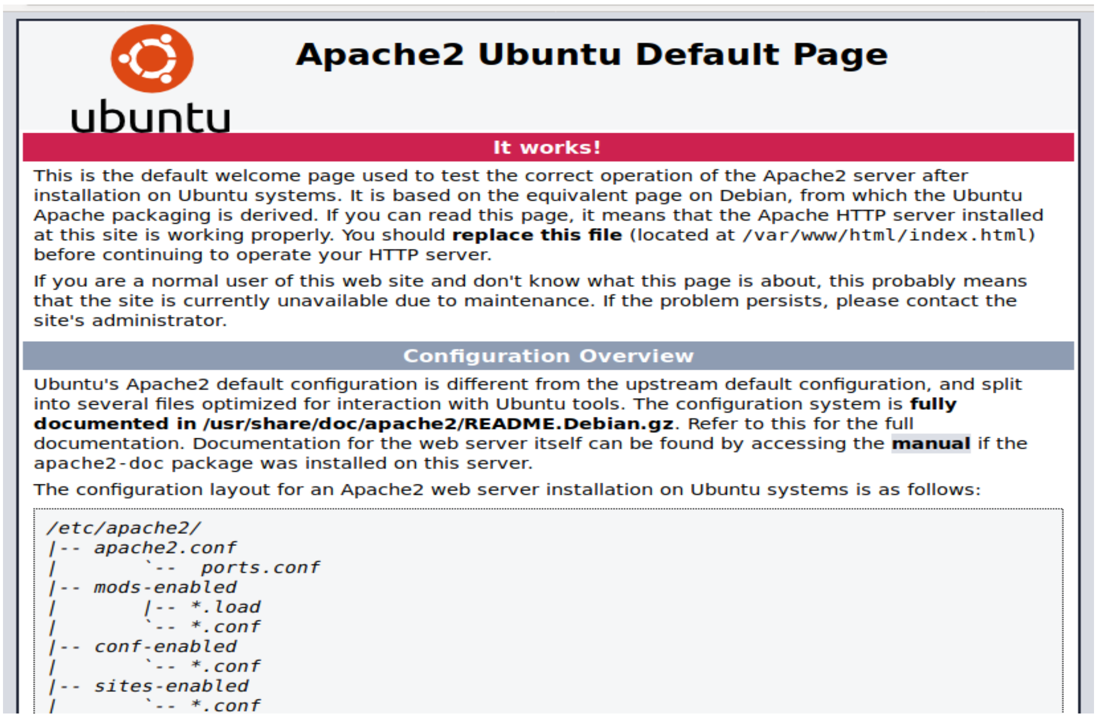


```
mca@S17:~$ sudo ufw allow "Apache Full"
Skipping adding existing rule
Skipping adding existing rule (v6)
```

You can do a spot check right away to verify that everything went as planned by visiting your server's public IP address in your web browser:

- http://your_server_ip

You will see the default Ubuntu 18.04 Apache web page, which is there for informational and testing purposes. It should look something like this:



Apache2 Ubuntu Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at /var/www/html/index.html) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|        `--  ports.conf
|-- mods-enabled
|        |-- *.load
|        `-- *.conf
|-- conf-enabled
|        `-- *.conf
|-- sites-enabled
|        `-- *.conf
```

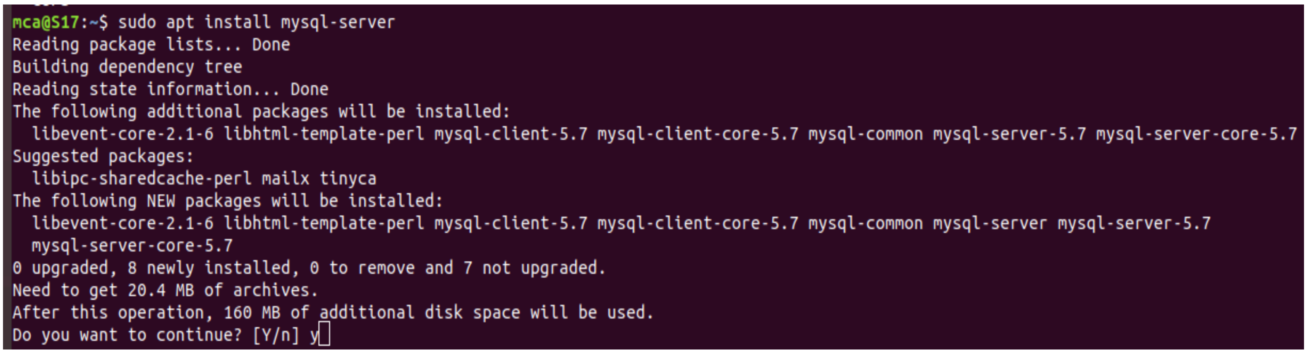
The Apache2 default index page will be displayed in case the webserver is up and running.

Root directory is /var/www/html

Step 2 — Installing MySQL

Again, use apt to acquire and install this software:

- sudo apt install mysql-server

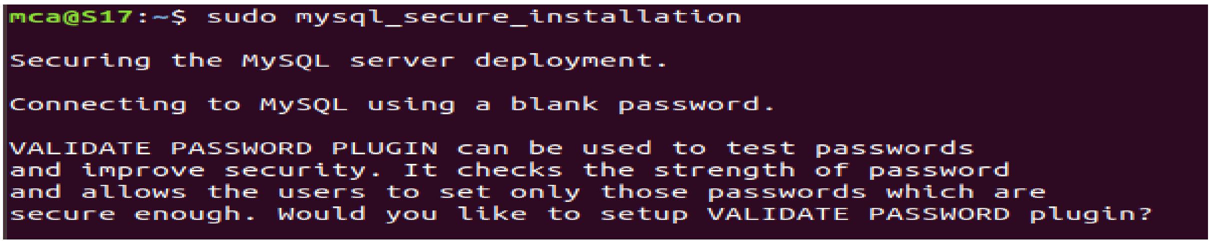


```
mca@S17:~$ sudo apt install mysql-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libevent-core-2.1-6 libhtml-template-perl mysql-client-5.7 mysql-client-core-5.7 mysql-common mysql-server-5.7 mysql-server-core-5.7
Suggested packages:
  libipc-sharedcache-perl mailx tinyca
The following NEW packages will be installed:
  libevent-core-2.1-6 libhtml-template-perl mysql-client-5.7 mysql-client-core-5.7 mysql-common mysql-server mysql-server-5.7
  mysql-server-core-5.7
0 upgraded, 8 newly installed, 0 to remove and 7 not upgraded.
Need to get 20.4 MB of archives.
After this operation, 160 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

This command, too, will show you a list of the packages that will be installed, along with the amount of disk space they'll take up. Enter Y to continue.

When the installation is complete, run a simple security script that comes pre-installed with MySQL which will remove some dangerous defaults and lock down access to your database system. Start the interactive script by running:

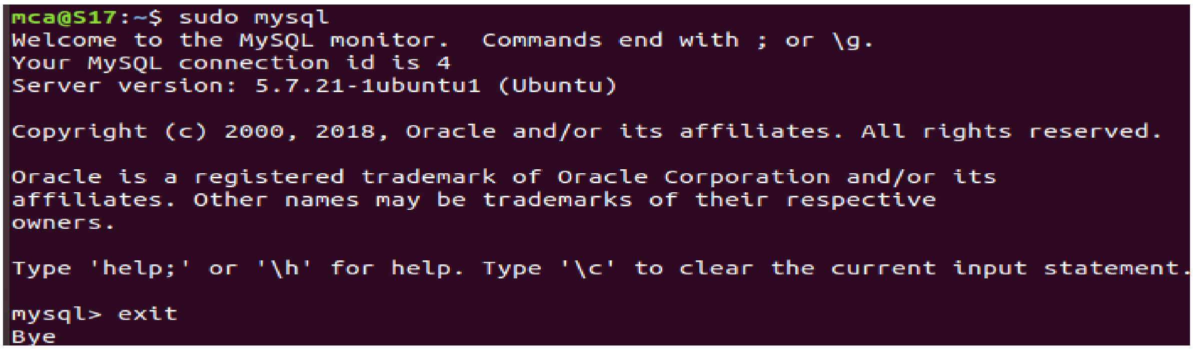
- sudo mysql_secure_installation



```
mca@S17:~$ sudo mysql_secure_installation

Securing the MySQL server deployment.

Connecting to MySQL using a blank password.

VALIDATE PASSWORD PLUGIN can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD plugin?
```

This will ask if you want to configure the VALIDATE PASSWORD PLUGIN. Answer Y for yes, or anything else to continue without enabling. When you're finished, test if you're able to log in to the MySQL console by typing:

- sudo mysql



```
mca@S17:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
Server version: 5.7.21-1ubuntu1 (Ubuntu)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

mysql> exit
Bye
```

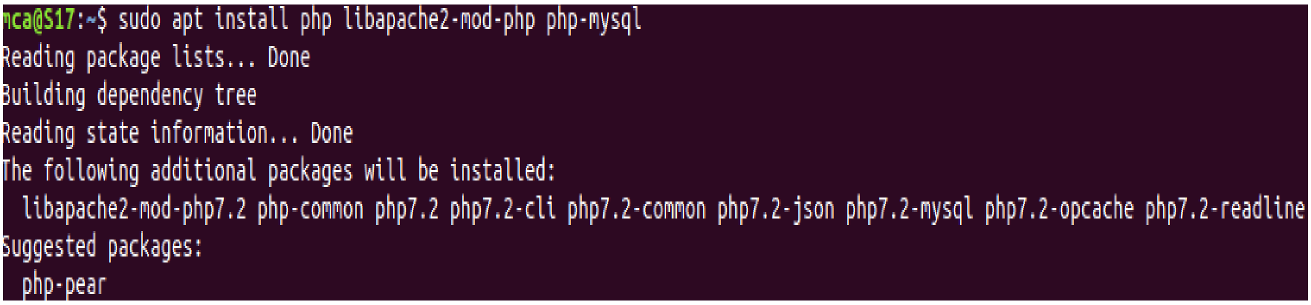
This will connect to the MySQL server as the administrative database user root, which is inferred by the use of sudo when running this command. To exit the MySQL console, type:

- exit

Step 3 — Installing PHP

In addition to the php package, you'll also need libapache2-mod-php to integrate PHP into Apache, and the php-mysql package to allow PHP to connect to MySQL databases. Run the following command to install all three packages and their dependencies:

- sudo apt install php libapache2-mod-php php-mysql



```
mca@S17:~$ sudo apt install php libapache2-mod-php php-mysql
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libapache2-mod-php7.2 php-common php7.2 php7.2-cli php7.2-common php7.2-json php7.2-mysql php7.2-opcache php7.2-readline
Suggested packages:
  php-pear
```

After this, restart the Apache web server in order for your changes to be recognized. You can do that with the following command:

- sudo systemctl restart apache2

```
mca@S17:~$ sudo systemctl restart apache2
```

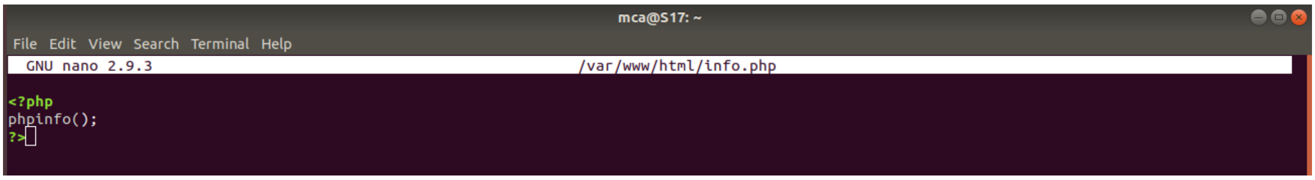
Step 5 — Testing PHP Processing on your Web Server

In order to test that your system is properly configured for PHP, create a PHP script called info.php. In order for Apache to find this file and serve it correctly, it must be saved to your web root directory. Create the file at the web root you created in the previous step by running:

- sudo nano /var/www/your_domain/info.php

This will open a blank file. Add the following text, which is valid PHP code, inside the file:

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



```
  GNU nano 2.9.3                    /var/www/html/info.php

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

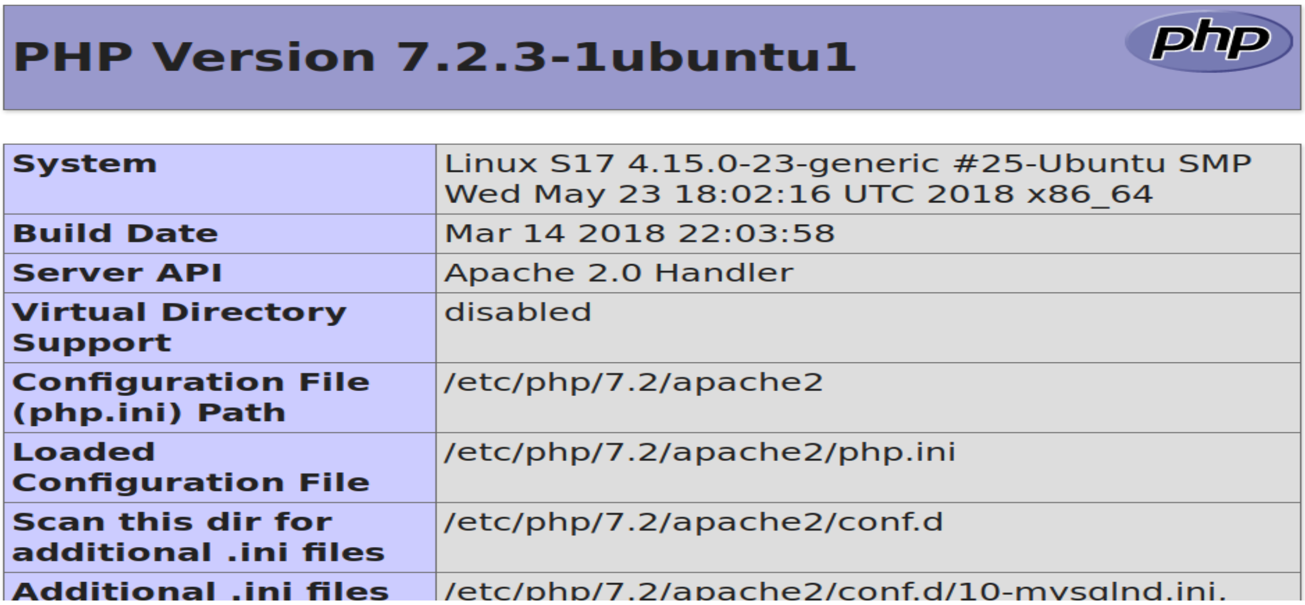
When you are finished, save and close the file.

Now you can test whether your web server is able to correctly display content generated by this PHP script. To try this out, visit this page in your web browser. You'll need your server's public IP address or domain name again.

The address you will want to visit is:

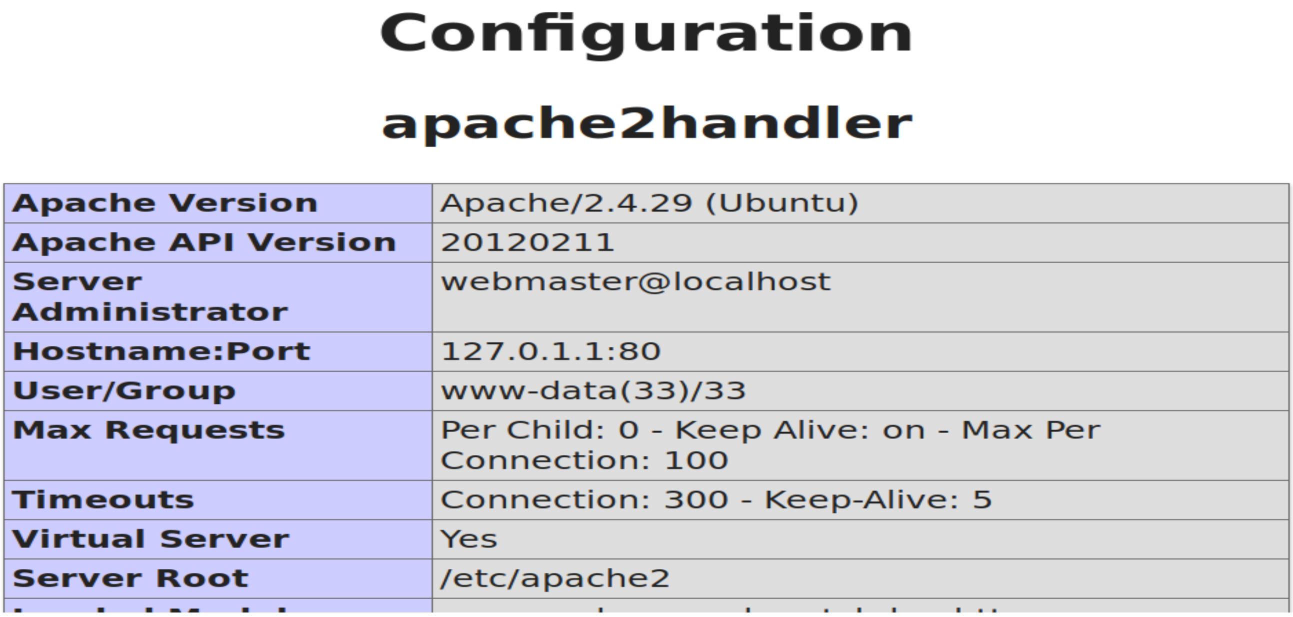
http://your_domain/info.php

The page that you come to should look something like this:



PHP Version 7.2.3-1ubuntu1

System	Linux S17 4.15.0-23-generic #25-Ubuntu SMP Wed May 23 18:02:16 UTC 2018 x86_64
Build Date	Mar 14 2018 22:03:58
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.2/apache2
Loaded Configuration File	/etc/php/7.2/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.2/apache2/conf.d
Additional .ini files	/etc/php/7.2/apache2/conf.d/10-mysqlnd.ini,



Configuration

apache2handler

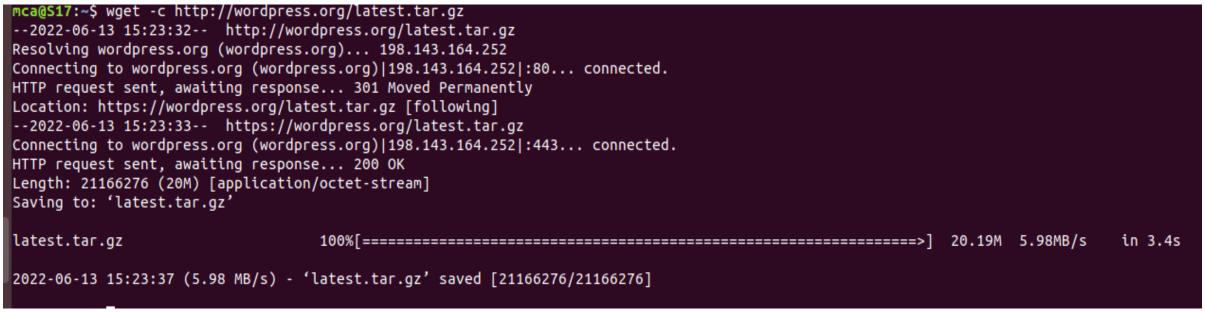
Apache Version	Apache/2.4.29 (Ubuntu)
Apache API Version	20120211
Server Administrator	webmaster@localhost
Hostname:Port	127.0.1.1:80
User/Group	www-data(33)/33
Max Requests	Per Child: 0 - Keep Alive: on - Max Per Connection: 100
Timeouts	Connection: 300 - Keep-Alive: 5
Virtual Server	Yes
Server Root	/etc/apache2

Install WordPress with LAMP on Ubuntu 18.04

Step 1 – Download WordPress

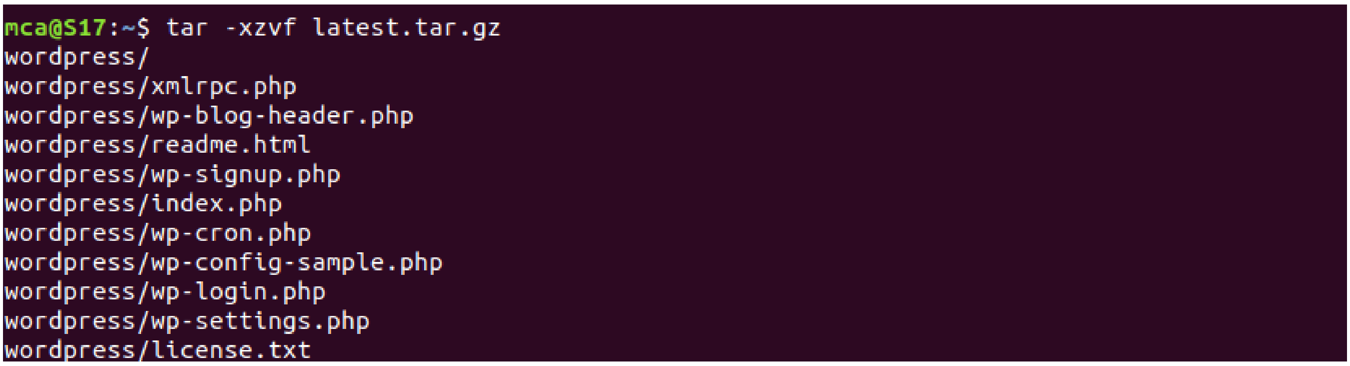
Download the latest version of the WordPress package and extract it by issuing the commands below on the terminal:

- wget -c http://wordpress.org/latest.tar.gz



```
mca@S17:~$ wget -c http://wordpress.org/latest.tar.gz
--2022-06-13 15:23:32--  http://wordpress.org/latest.tar.gz
Resolving wordpress.org (wordpress.org)... 198.143.164.252
Connecting to wordpress.org (wordpress.org)|198.143.164.252|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://wordpress.org/latest.tar.gz [following]
--2022-06-13 15:23:33--  https://wordpress.org/latest.tar.gz
Connecting to wordpress.org (wordpress.org)|198.143.164.252|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 21166276 (20M) [application/octet-stream]
Saving to: ‘latest.tar.gz’

latest.tar.gz                 100%[=================================================================>]  20.19M  5.98MB/s    in 3.4s

2022-06-13 15:23:37 (5.98 MB/s) - ‘latest.tar.gz’ saved [21166276/21166276]
```

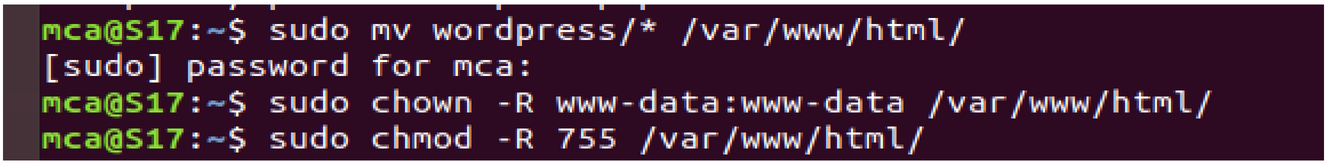
- tar -xzvf latest.tar.gz



```
mca@S17:~$ tar -xzvf 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
```

Then move the WordPress files from the extracted folder to the Apache default root directory, /var/www/html/:

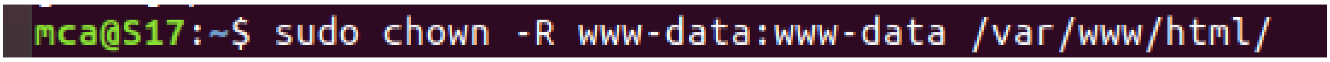
- ➢ sudo mv wordpress/* /var/www/html/



```
mca@S17:~$ sudo mv wordpress/* /var/www/html/
[sudo] password for mca:
mca@S17:~$ sudo chown -R www-data:www-data /var/www/html/
mca@S17:~$ sudo chmod -R 755 /var/www/html/
```

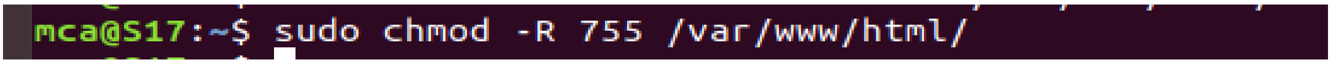
Next, set the correct permissions on the website directory, that is give wnership of the WordPress files to the webserver as follows:

- ➢ sudo chown -R www-data:www-data /var/www/html/



```
mca@S17:~$ sudo chown -R www-data:www-data /var/www/html/
```

- ➢ sudo chmod -R 755 /var/www/html/

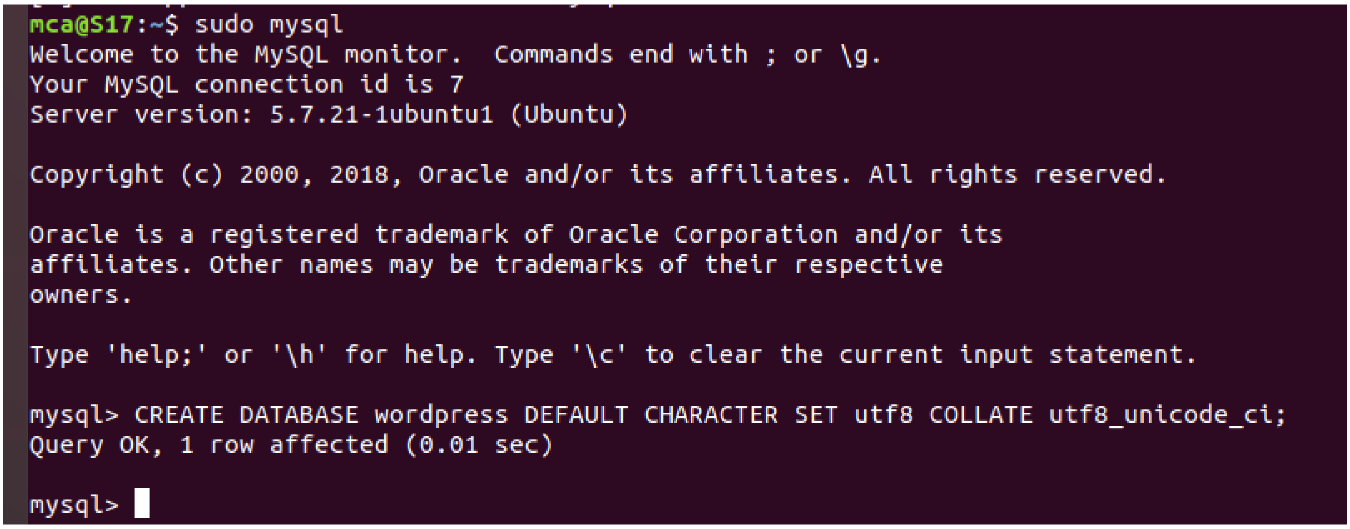


```
mca@S17:~$ sudo chmod -R 755 /var/www/html/
```

Step 2 – Creating a MySQL Database and User for WordPress

The first step you'll take is a preparatory one. Even though MySQL is already installed, you still need to create a database to manage and store the user information for WordPress to use. To get started, log into the MySQL root (administrative) account by issuing the following command:

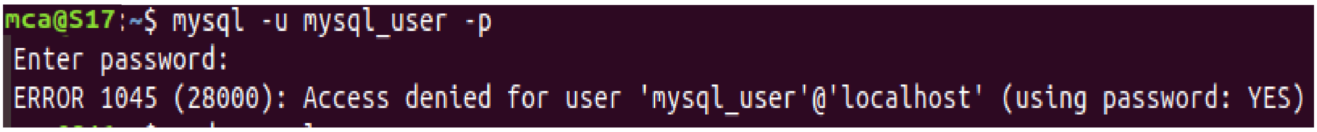
sudo mysql



```
mca@S17:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7
Server version: 5.7.21-1ubuntu1 (Ubuntu)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

mysql> CREATE DATABASE wordpress DEFAULT CHARACTER SET utf8 COLLATE utf8_unicode_ci;
Query OK, 1 row affected (0.01 sec)

mysql> 
```

You will be prompted for the password you set for the MySQL root account when you installed the software. However, if you have password authentication enabled for your root user, you can run the following command and enter your password information when prompted:

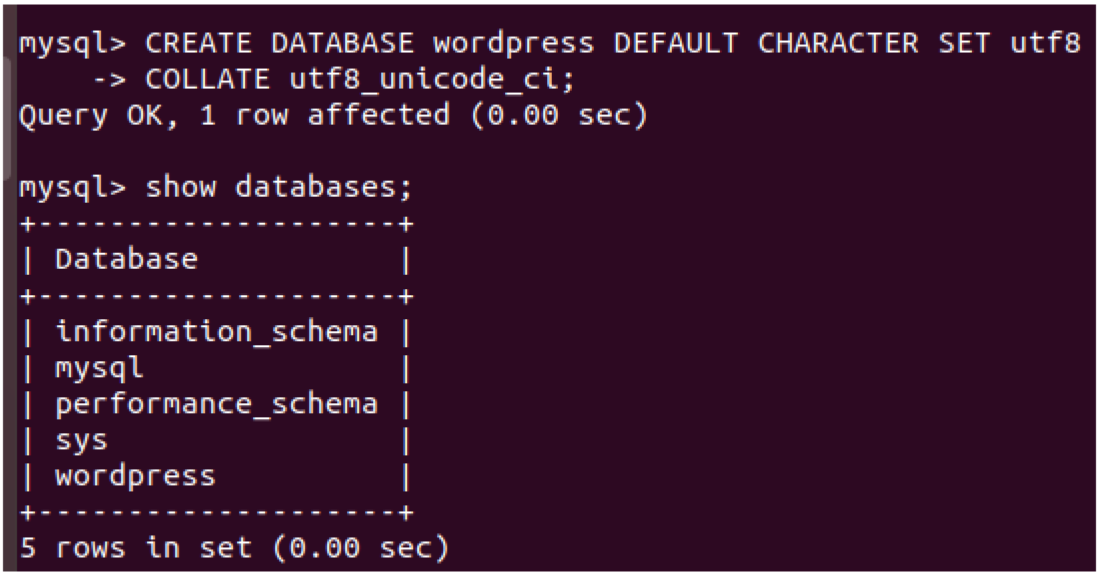
- ➢ mysql -u root –p



```
mca@S17:~$ mysql -u mysql_user -p
Enter password:
ERROR 1045 (28000): Access denied for user 'mysql_user'@'localhost' (using password: YES)
```

From there, you'll create a new database that WordPress will control. You can call this whatever you would like, but we will be using wordpress in this guide as an example. Create the database for WordPress by writing the following:

- ➢ CREATE DATABASE wordpress DEFAULT CHARACTER SET utf8 COLLATE utf8_unicode_ci;

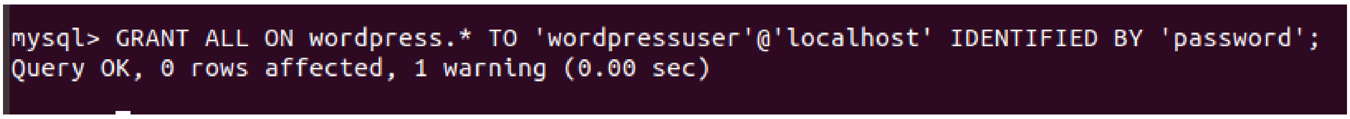


```
mysql> CREATE DATABASE wordpress DEFAULT CHARACTER SET utf8
    -> COLLATE utf8_unicode_ci;
Query OK, 1 row affected (0.00 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| wordpress          |
+--------------------+
5 rows in set (0.00 sec)
```

Next, you're going to create a separate MySQL user account that you'll use exclusively to operate on the new database. Creating one-function databases and accounts is a good idea from a management and security standpoint. We will use the name wordpressuser as an example in this guide. Feel free to change this if you'd like.

You can create this account, set a password for it, and then grant it access to the database you created all by running the following command. Remember to choose a strong password here for your database user:

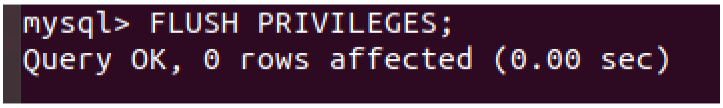
- ➢ GRANT ALL ON wordpress.* TO 'wordpressuser'@'localhost' IDENTIFIED BY 'password';



```
mysql> GRANT ALL ON wordpress.* TO 'wordpressuser'@'localhost' IDENTIFIED BY 'password';
Query OK, 0 rows affected, 1 warning (0.00 sec)
```

After creating this user, flush the privileges to ensure that the current instance of MySQL knows about the recent changes you've made:

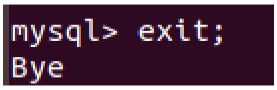
- ➢ FLUSH PRIVILEGES;



```
mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.00 sec)
```

Exit out of MySQL:

- EXIT

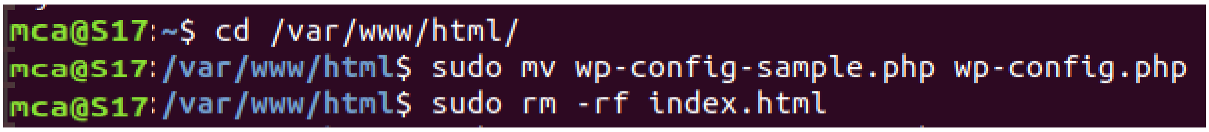


```
mysql> exit;
Bye
```

You now have a database and user account in MySQL, each made specifically for WordPress.

Go the /var/www/html/ directory and rename existing wp-config-sample.php to wpconfig.php. Also, make sure to remove the default Apache index page.

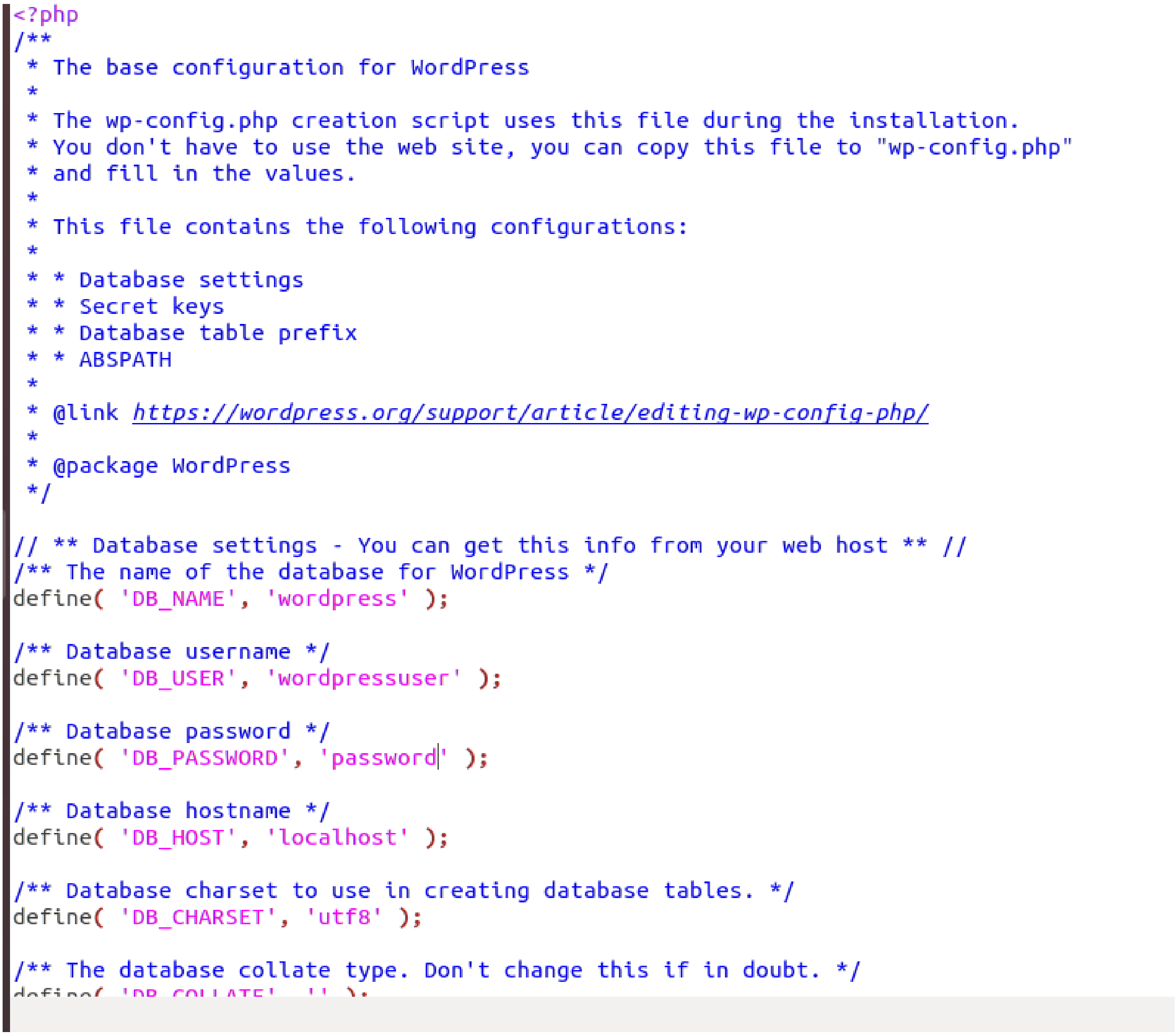
- cd /var/www/html/
- sudo mv wp-config-sample.php wp-config.php
- sudo rm -rf index.html



```
mca@S17:~$ cd /var/www/html/
mca@S17:/var/www/html$ sudo mv wp-config-sample.php wp-config.php
mca@S17:/var/www/html$ sudo rm -rf index.html
```

Then update it with your database information under the MySQL settings section (refer to the highlighted boxes in the image below):

This setting can be added after the database connection settings, or anywhere else in the file:

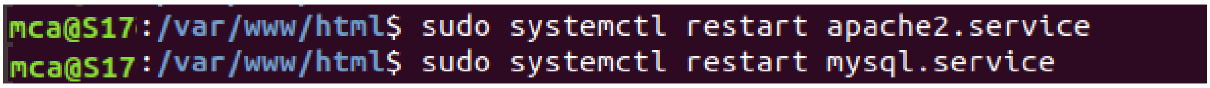


```
<?php
/**
 * The base configuration for WordPress
 *
 * The wp-config.php creation script uses this file during the installation.
 * You don't have to use the web site, you can copy this file to "wp-config.php"
 * and fill in the values.
 *
 * This file contains the following configurations:
 *
 * * Database settings
 * * Secret keys
 * * Database table prefix
 * * ABSPATH
 *
 * @link https://wordpress.org/support/article/editing-wp-config-php/
 *
 * @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', 'password' );

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

/** Database charset to use in creating database tables. */
define( 'DB_CHARSET', 'utf8' );

/** The database collate type. Don't change this if in doubt. */
define( 'DB_COLLATE', '' );
```

Save and close the file when you are finished.

Restart the web server and mysql service using the commands below:

- sudo systemctl restart apache2.service
- sudo systemctl restart mysql.service



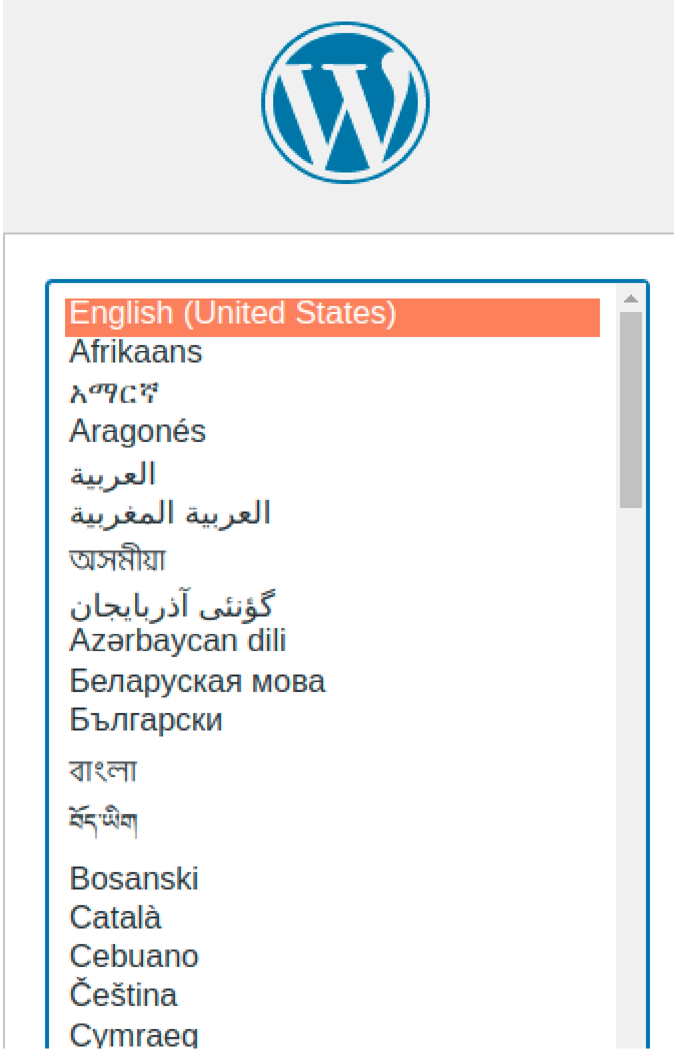
```
mca@S17:/var/www/html$ sudo systemctl restart apache2.service
mca@S17:/var/www/html$ sudo systemctl restart mysql.service
```

Step 3 – Completing the Installation Through the Web Interface

Now that the server configuration is complete, you can complete the installation through the web interface. In your web browser, navigate to your server’s domain name or public IP address:

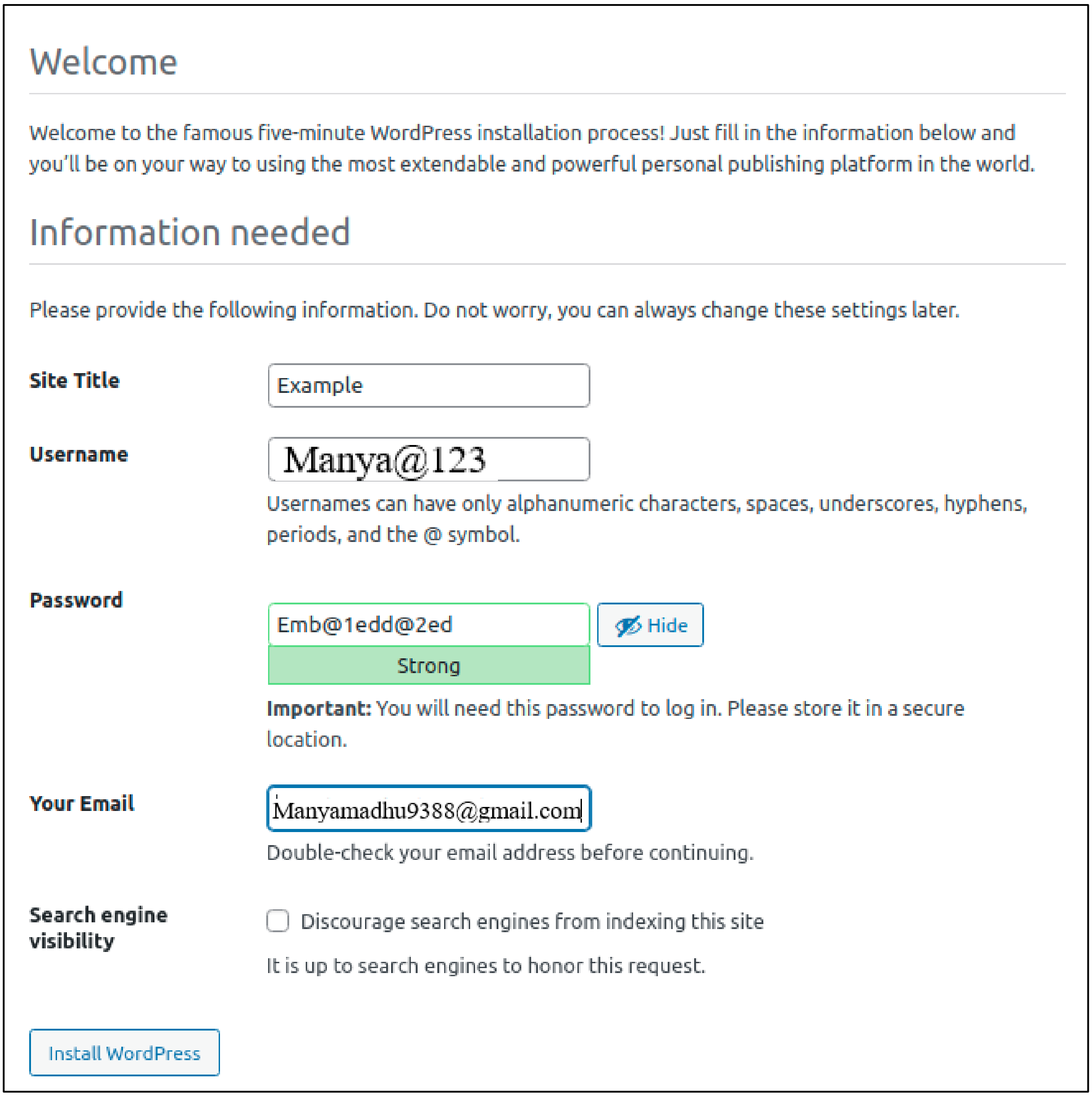
- https://server_domain_or_IP

Select the language you would like to use:



Next you will be directed to the main setup page. Select a name for your WordPress site and choose a username (it is recommended not to choose something like “admin” for security purposes). A strong password is generated automatically. Save this password or select an alternative strong password.

Enter your email address and select whether you want to discourage search engines from indexing your site:



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 Example

Username Manya@123

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

Password Emb@1edd@2ed

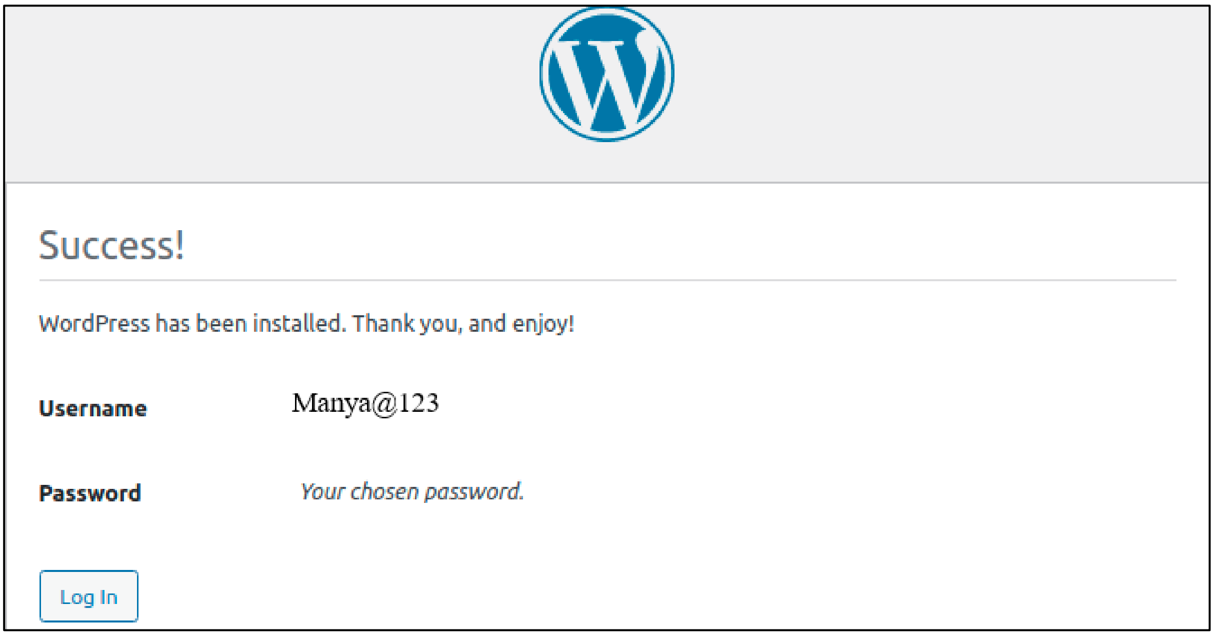
Important: You will need this password to log in. Please store it in a secure location.

Your Email Manyamadhu9388@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.

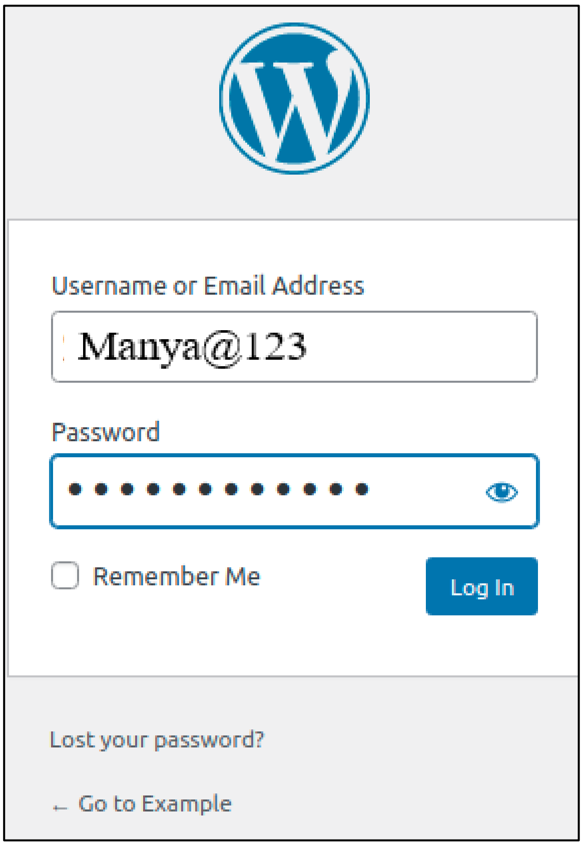


Success!

WordPress has been installed. Thank you, and enjoy!

Username Manya@123

Password *Your chosen password.*



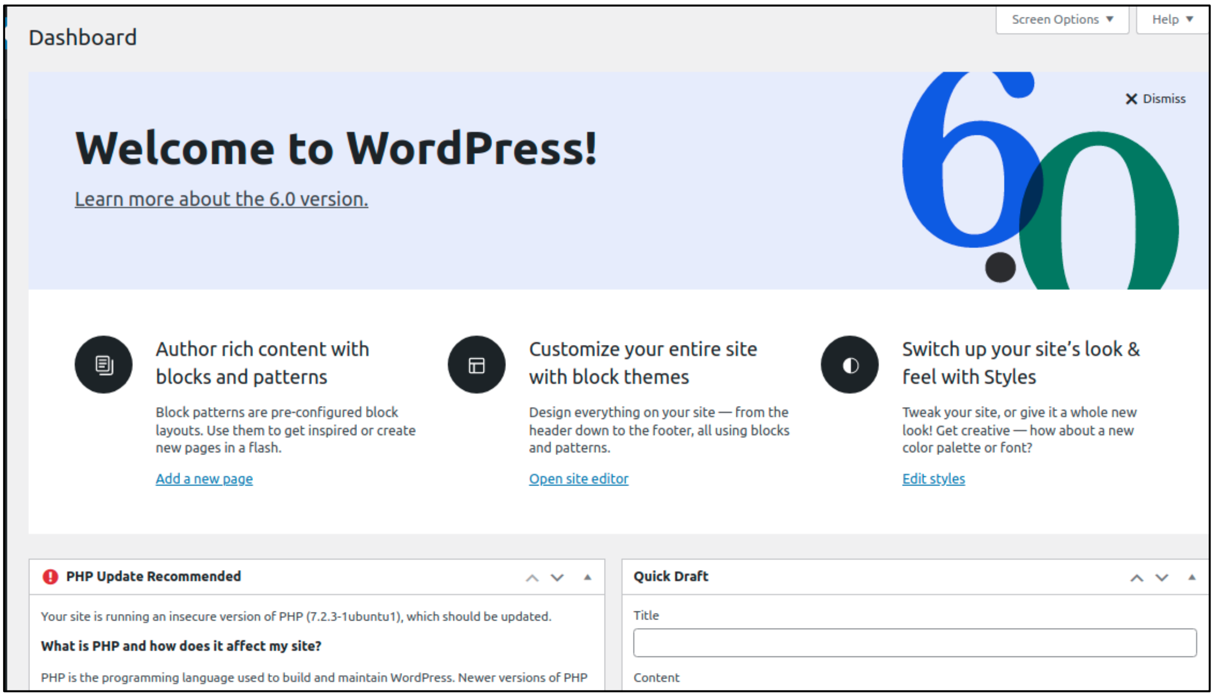
Username or Email Address

Manya@123

Password

••••••••••••

Once you log in, you will be taken to the WordPress administration dashboard:



Dashboard

Welcome to WordPress!

Learn more about the 6.0 version.

Author rich content with blocks and patterns

Block patterns are pre-configured block layouts. Use them to get inspired or create new pages in a flash.

Add a new page

Customize your entire site with block themes

Design everything on your site — from the header down to the footer, all using blocks and patterns.

Open site editor

Switch up your site's look & feel with Styles

Tweak your site, or give it a whole new look! Get creative — how about a new color palette or font?

Edit styles

PHP Update Recommended

Your site is running an insecure version of PHP (7.2.3-1ubuntu1), which should be updated.

What is PHP and how does it affect my site?

PHP is the programming language used to build and maintain WordPress. Newer versions of PHP

Quick Draft

Title

Content

From there, you can begin using and customizing your WordPress site.