**NETWORKING & SYSTEM ADMINISTRATION LAB**

**Name: Vishnu Mohan**

**Roll No: 51**

**Batch:RMCA B**

**Date: 13/06/2022**

**Experiment No.: 5**

**Aim**

Installation and configuration of LAMP stack. Deploy an open source application such as phpMyAdmin and WordPress.

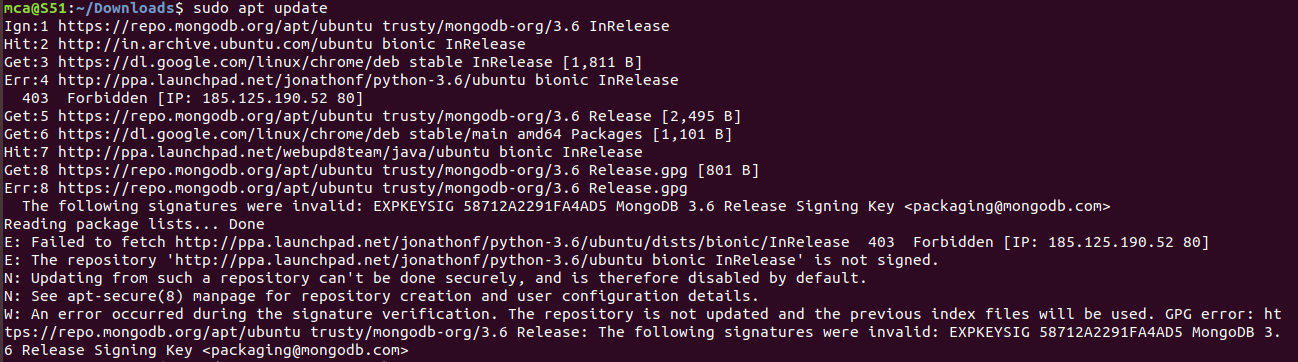
**Procedure**

**Step 1: Installing Apache and updating firewall**

1.Make sure apt class is updated

**$ sudo apt update**

**Output:-**

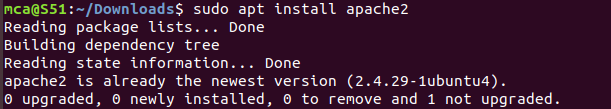


2. Once the apt is updated, install Apache

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.

**$ sudo apt install apache2**

**Output:-**

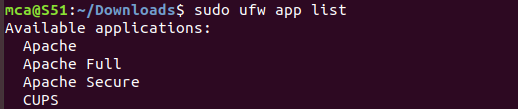
****

3. 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**

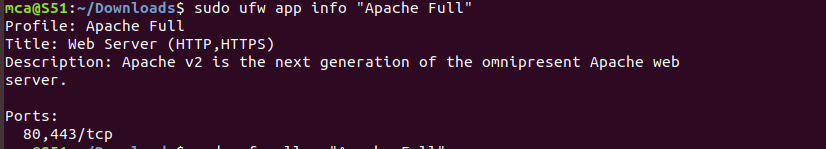
**Output:-**



4. 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”**

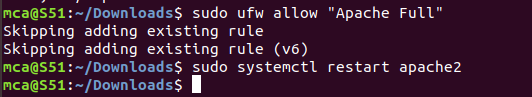
**Output:-**

****

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

**$ sudo ufw allow “Apache full”**

**Output:-**

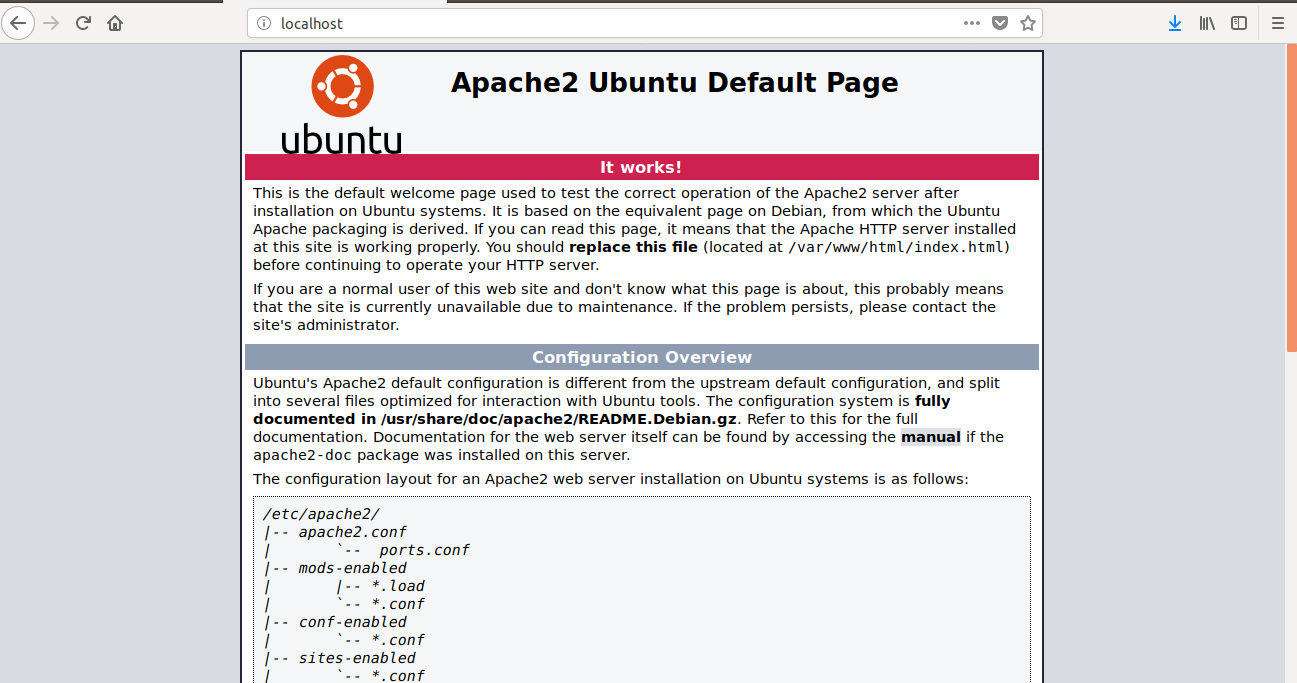
****

6. 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**

**http://localhost**

**Output:-**

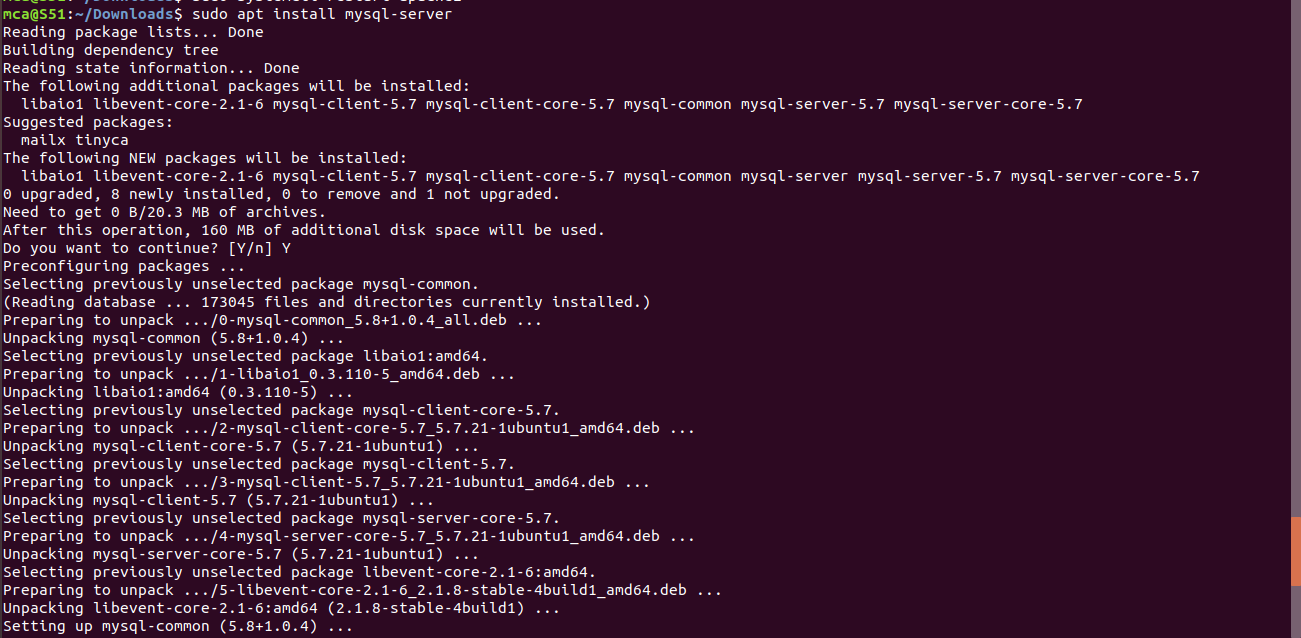


**Step 2: Installing MySQL**

1. Again , apt to acquire and install this software

**$ sudo apt install mysql-server**

**Output:-**

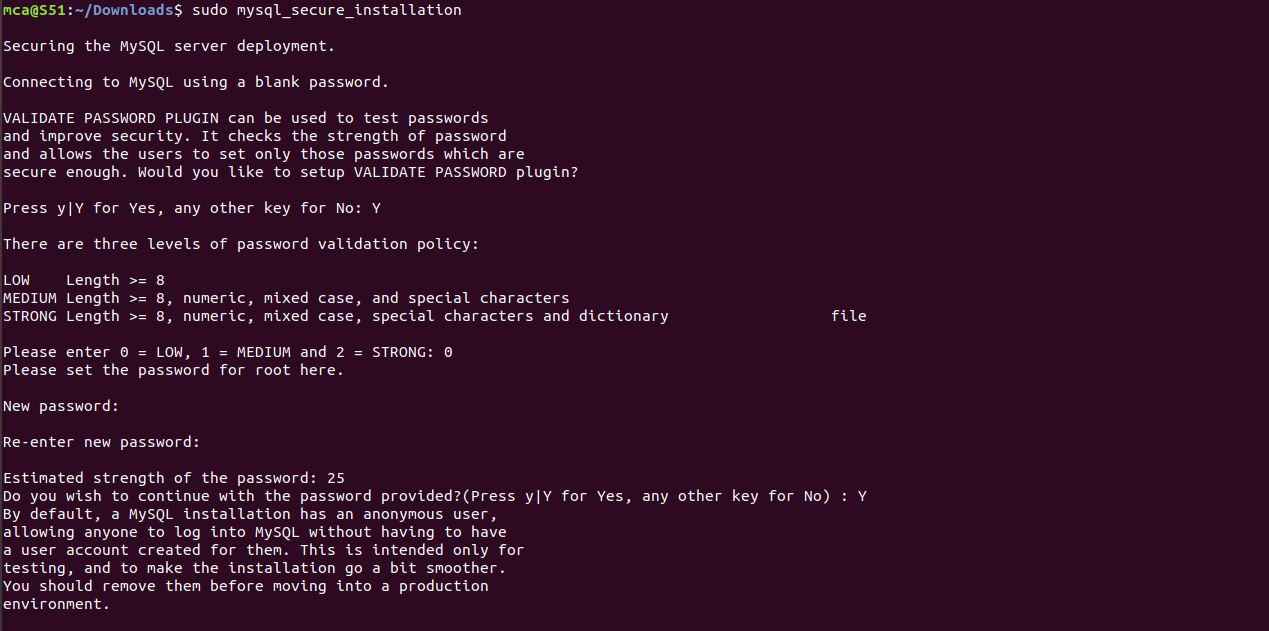
****

1. 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.

This will ask if you want to configure the VALIDATE PASSWORD PLUGIN. Answer Y for yes, or anything else to continue without enabling.

**$ sudo mysql\_secure\_installation**

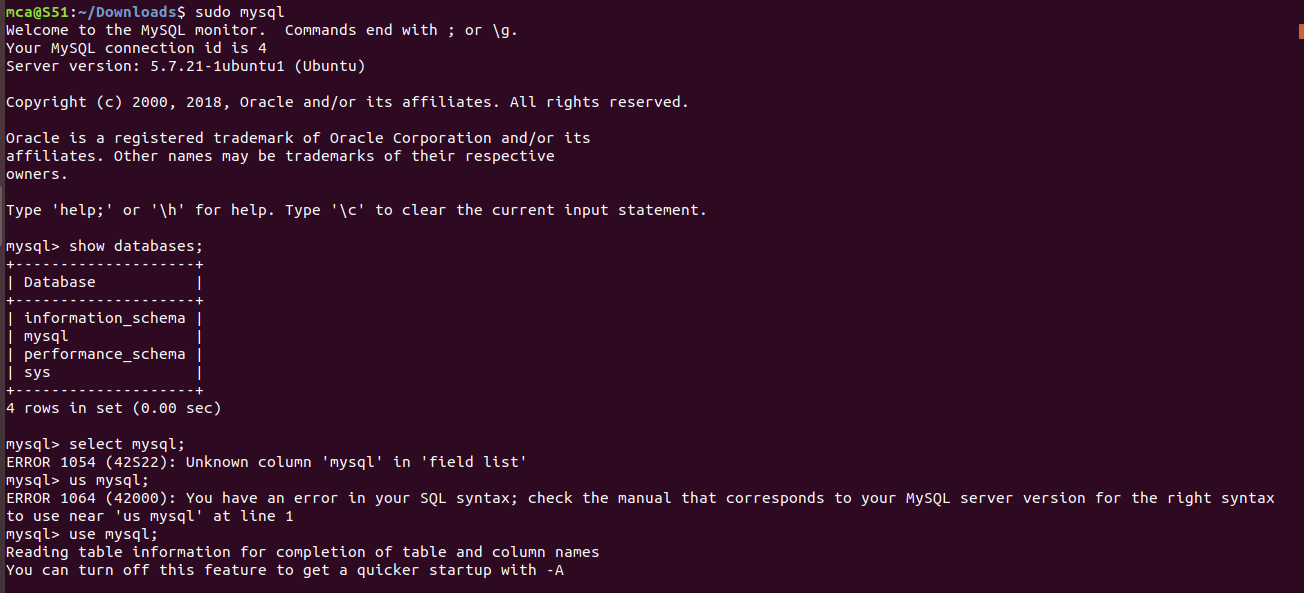
**Output:-**

****

1. When you’re finished, test if you’re able to log in to the MySQL console by typing sudo mysql,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:

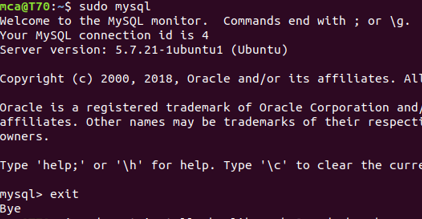
**$ sudo mysql**

**Output:-**

****

1. To exit MySQL console,

**$ exit**

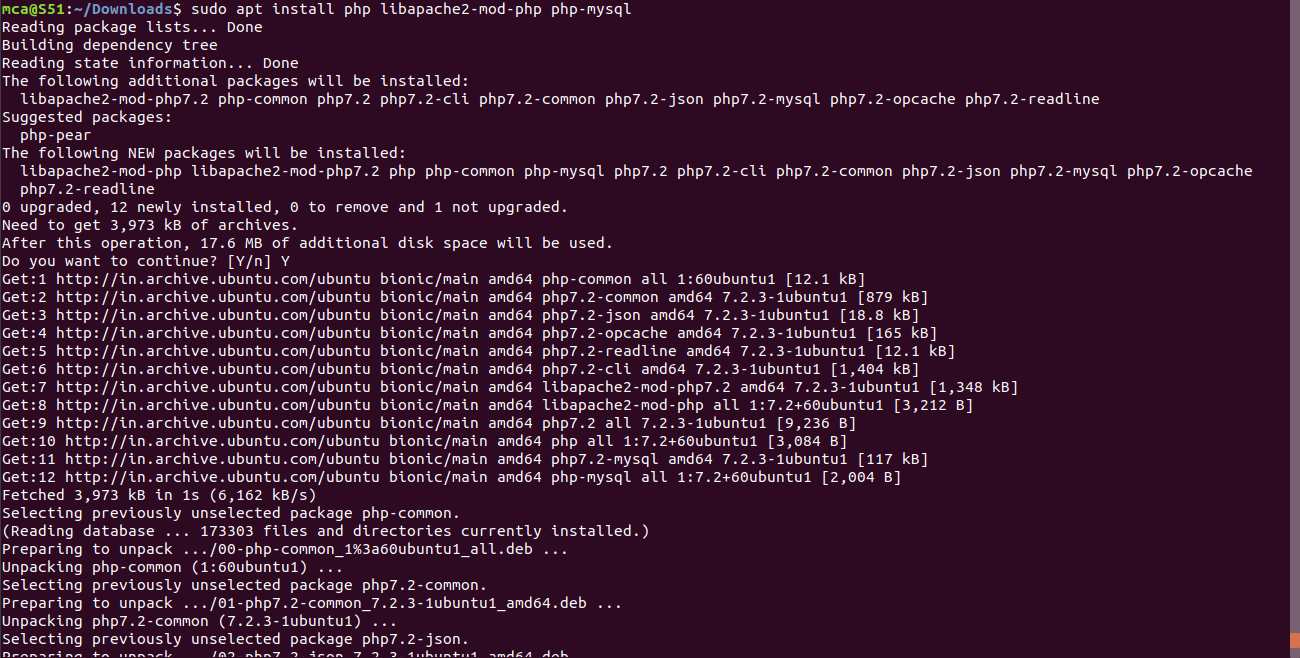


**Step 3: Installing PHP**

1. 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**

**Output:-**

****

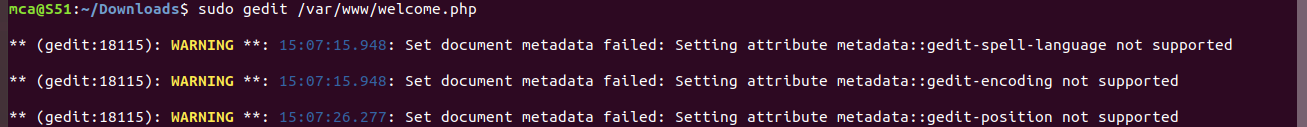
2. Testing The PHP processing on your web server

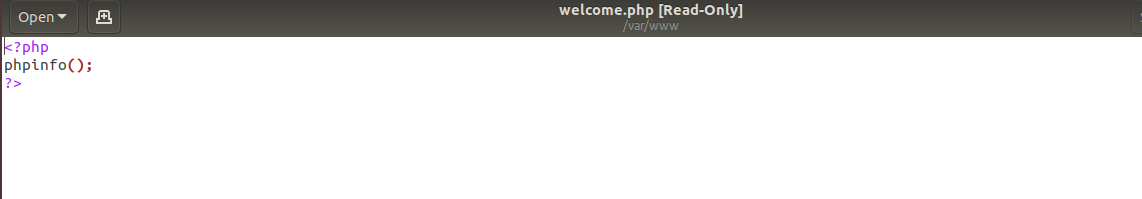
In order to test that your system is properly configured for PHP, create a PHP script called info.php. In order for Apache to find this file and serve it correctly, it must be saved to your web root directory. Create the file at the web root you created in the previous step by running:

**$ sudo gedit /var/www/welcome.php**

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

**Output:-**

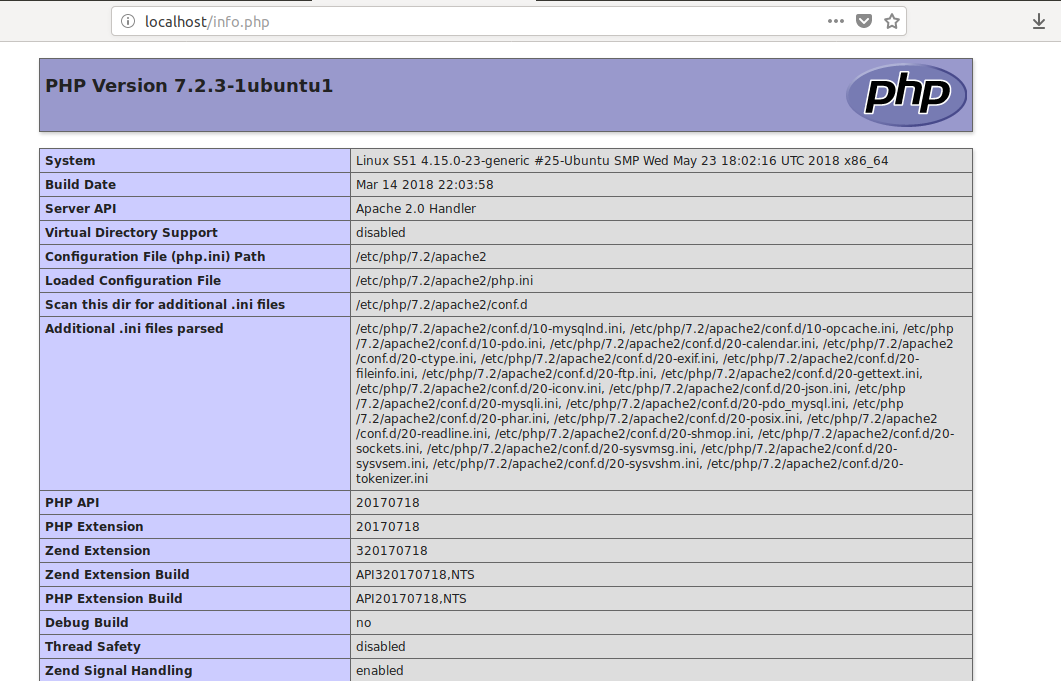
****

****

3.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:

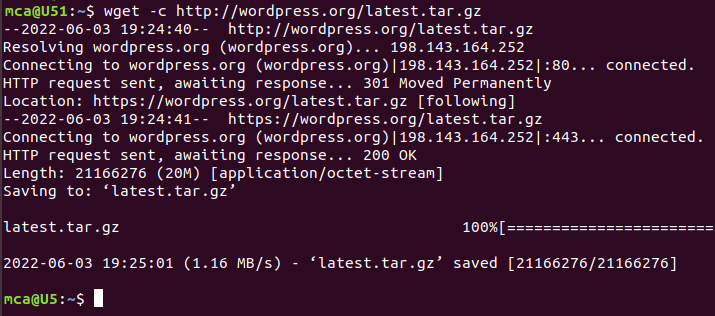
****

**Step 4: Install wordpress with LAMP on Ubuntu 18.04**

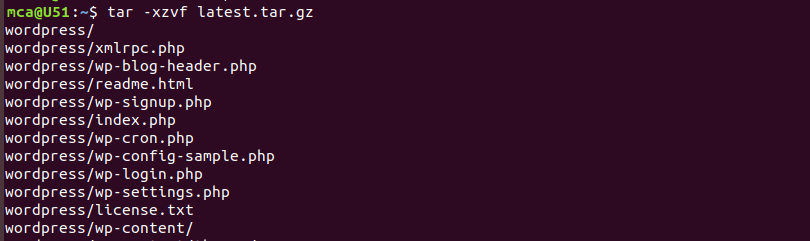
Download the latest version of the WordPress package and extract it by issuing the commands below on the terminal:

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

**Output:**

****

**$ tar -xzvf latest.tar.gz**

****

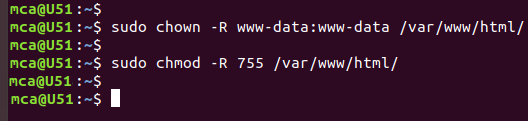
Then move the WordPress files from the extracted folder to the Apache default root directory, /var/www/html/:

**sudo mv wordpress/\* /var/www/html/**

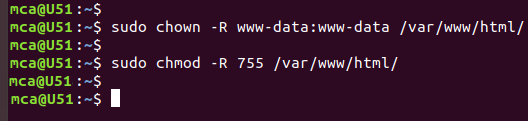
****

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

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

****

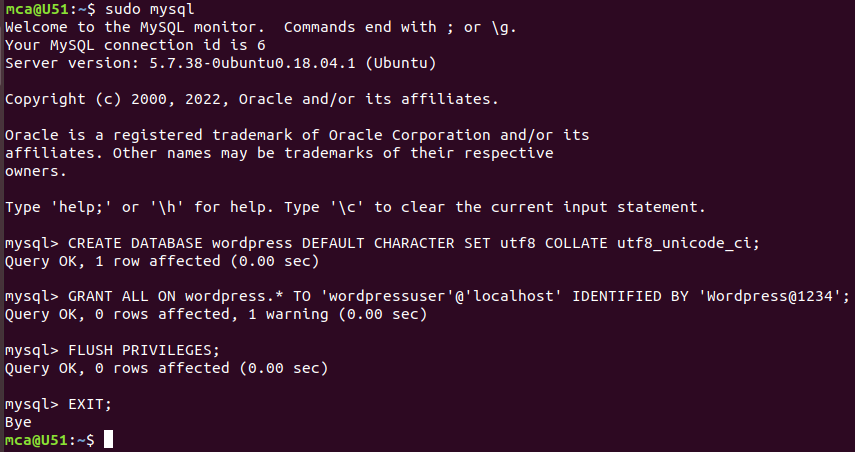
**sudo chmod -R 755 /var/www/html/**

****

**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**

****

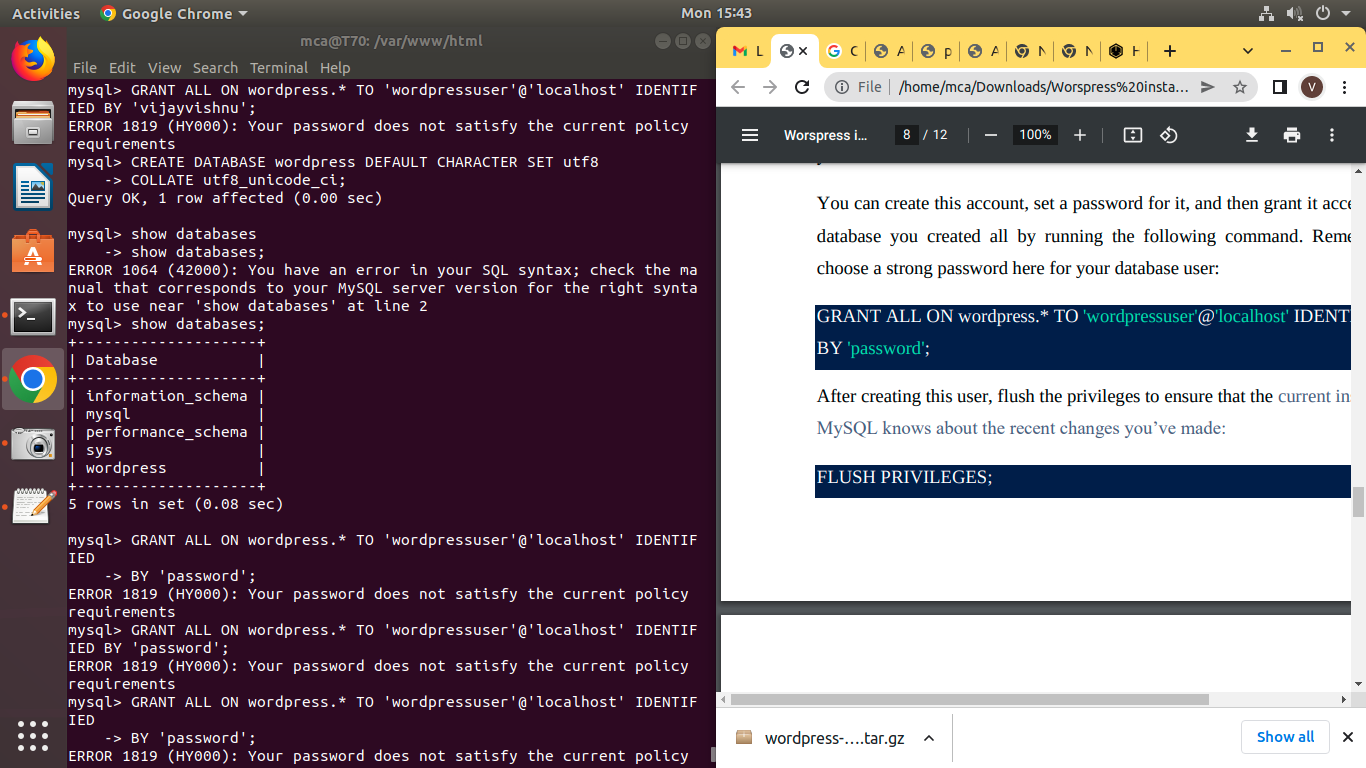
You will be prompted for the password you set for the MySQL root account when you installed the software. However, if you have password authentication enabled for your root user, you can run the following command and enter your password information when prompted:

**mysql -u root –p**



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:

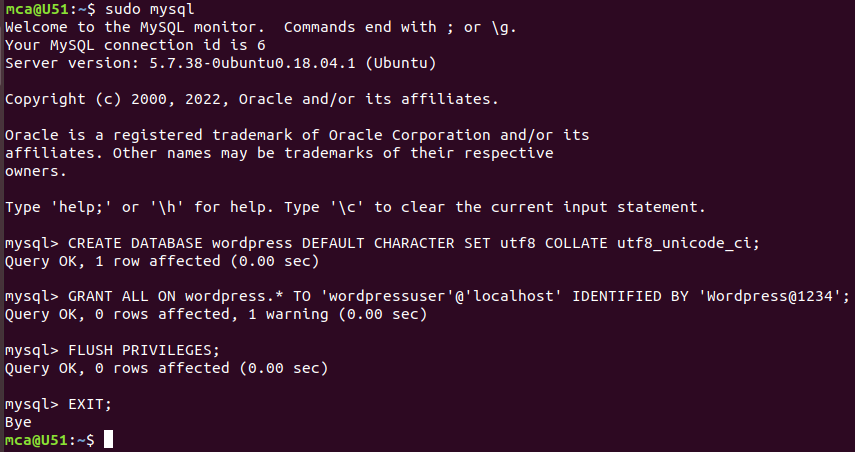
**CREATE DATABASE wordpress DEFAULT CHARACTER SET utf8 COLLATE utf8\_unicode\_ci;**



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.

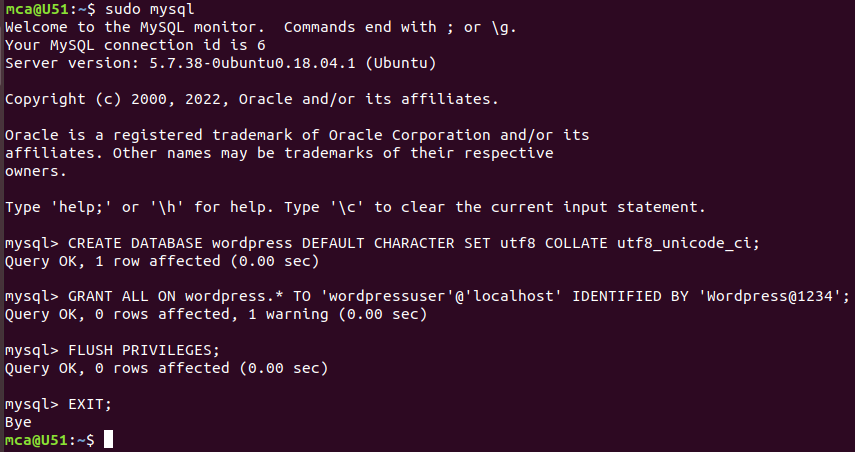
You can create this account, set a password for it, and then grant it access to the database you created all by running the following command. Remember to choose a strong password here for your database user:

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

****

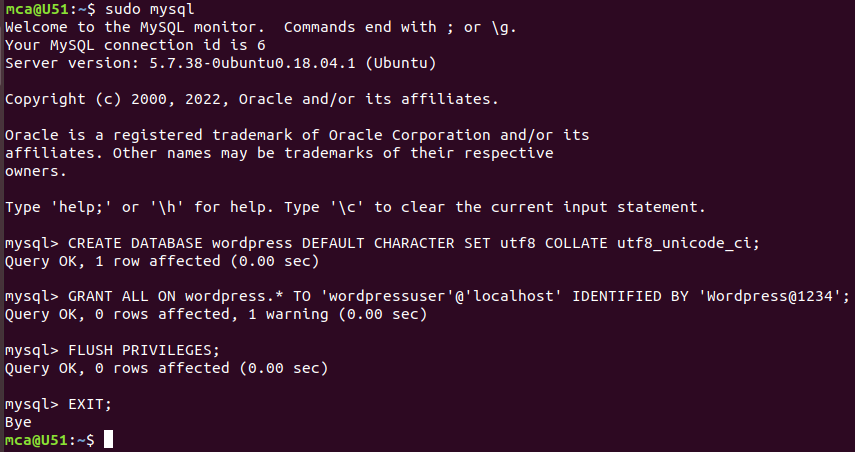
After creating this user, flush the privileges to ensure that the current instance of MySQL knows about the recent changes you’ve made:

**FLUSH PRIVILEGES;**



Exit out of MySQL:

**EXIT**

****

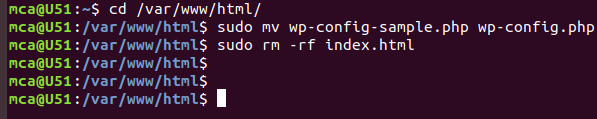
You now have a database and user account in MySQL, each made specifically for WordPress.

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/**

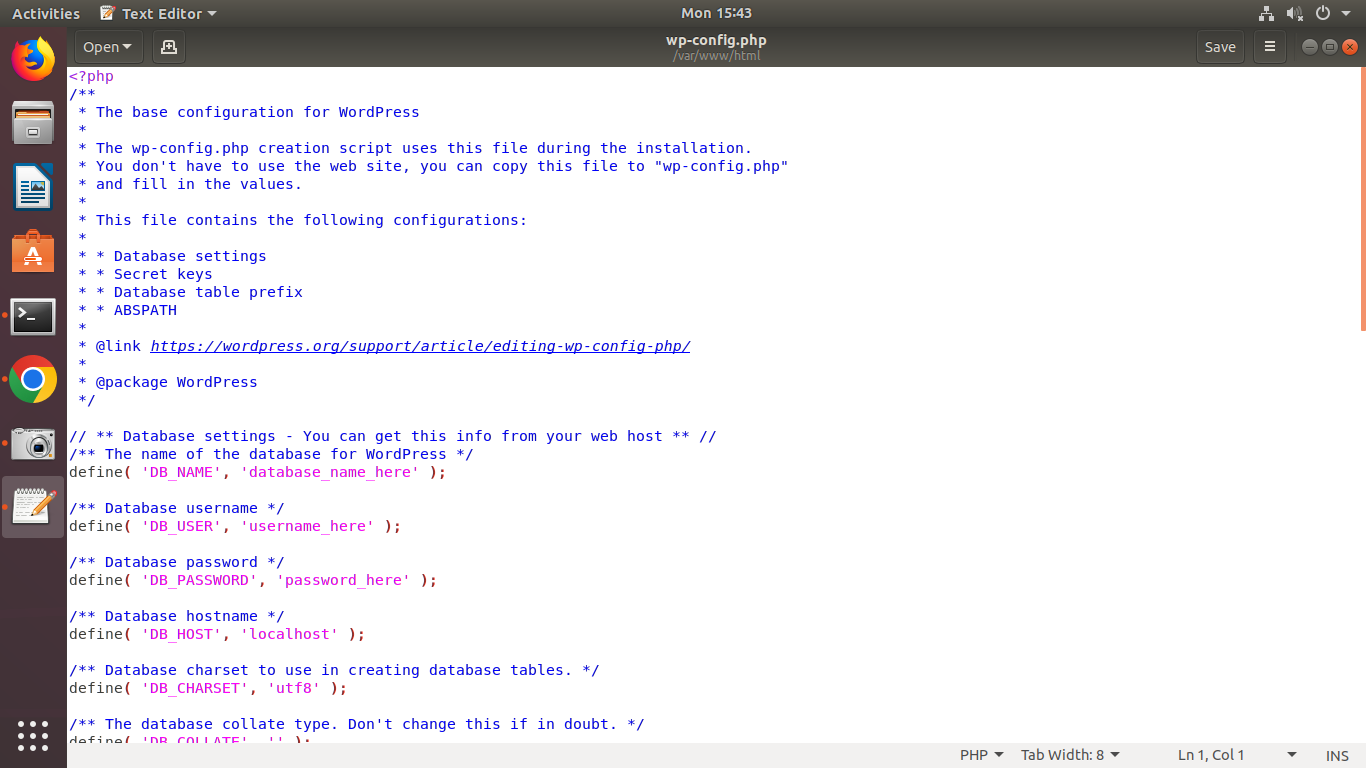
**sudo mv wp-config-sample.php wp-config.php**

**sudo rm -rf index.html**

****

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:

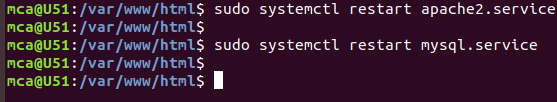


Save and close the file when you are finished.

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

**sudo systemctl restart apache2.service**

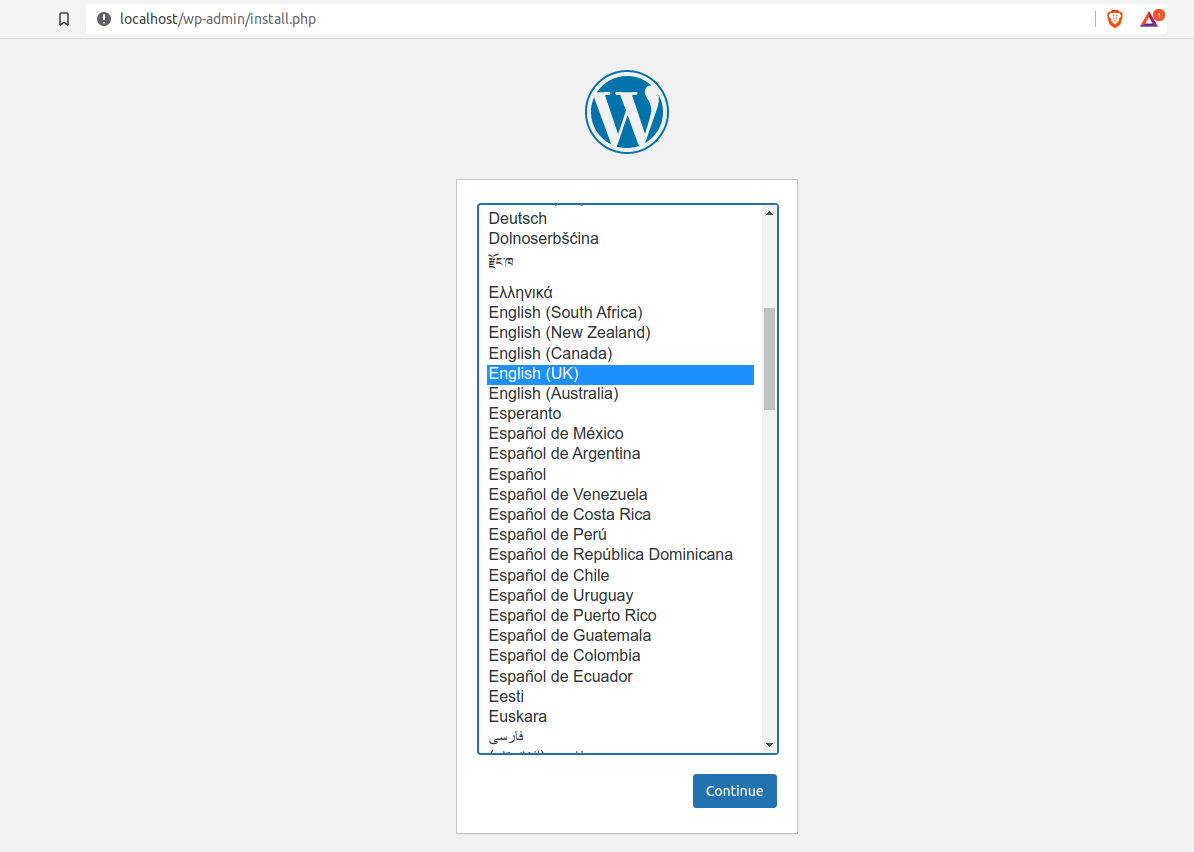
**sudo systemctl restart mysql.service**



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**

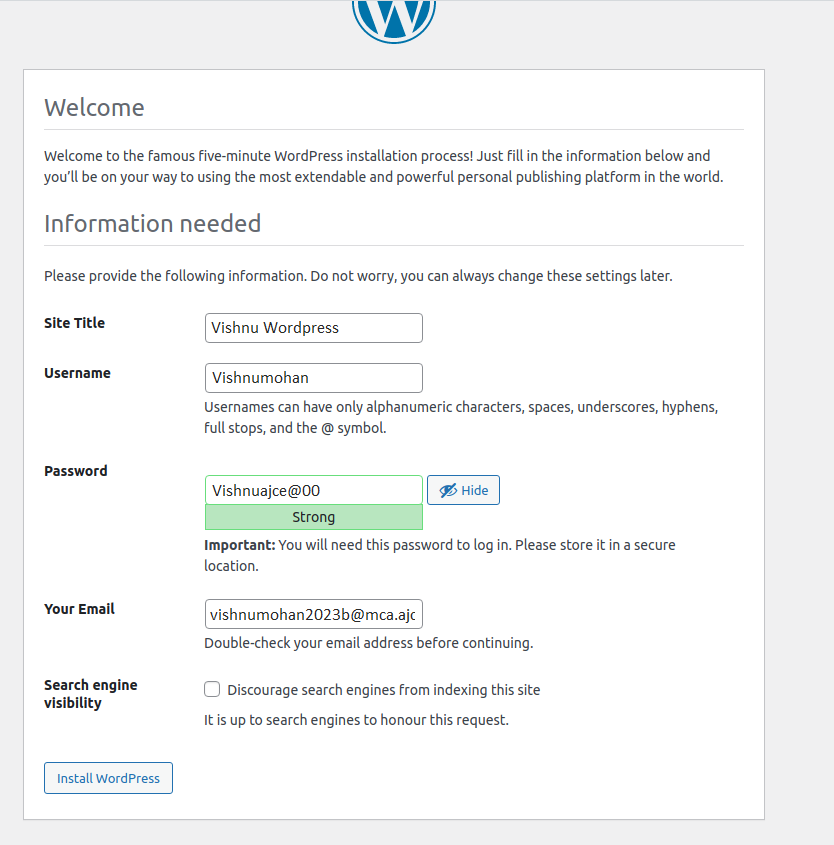
Select the language you would like to use:

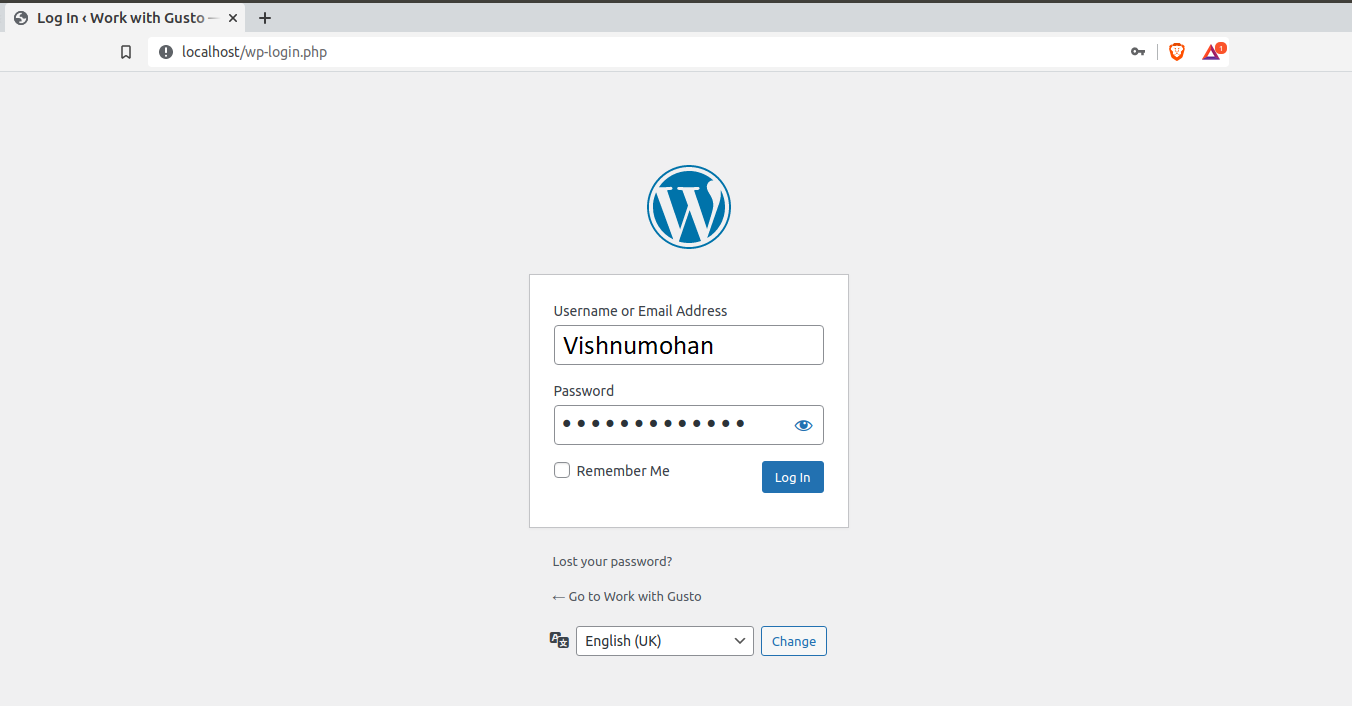


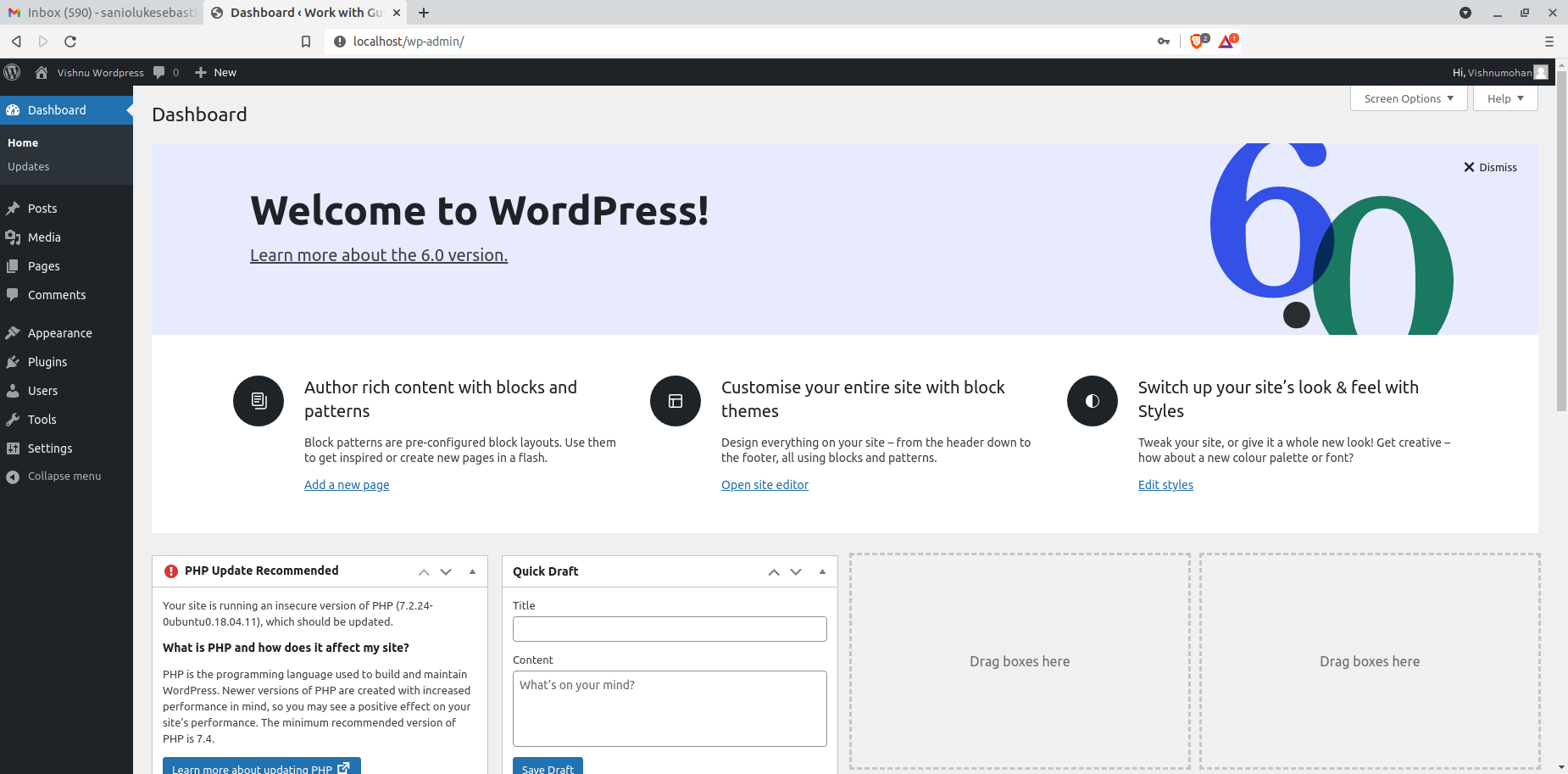
Next you will be directed to the main setup page. Select a name for your

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:

****

****

****