iGNITE Technologies

WORDPRESS FOR PENTESTER

LAB SETUP

www.hackingarticles.in

Contents

Introduction

In this post, we will demonstrate how to set up a vulnerable WordPress CMS for penetration testing on Ubuntu 20.04, using Docker and XAMPP on Windows.

WordPress Setup on Ubuntu 20.04

To configure WordPress on your Ubuntu platform, certain prerequisites are required for CMS installation.

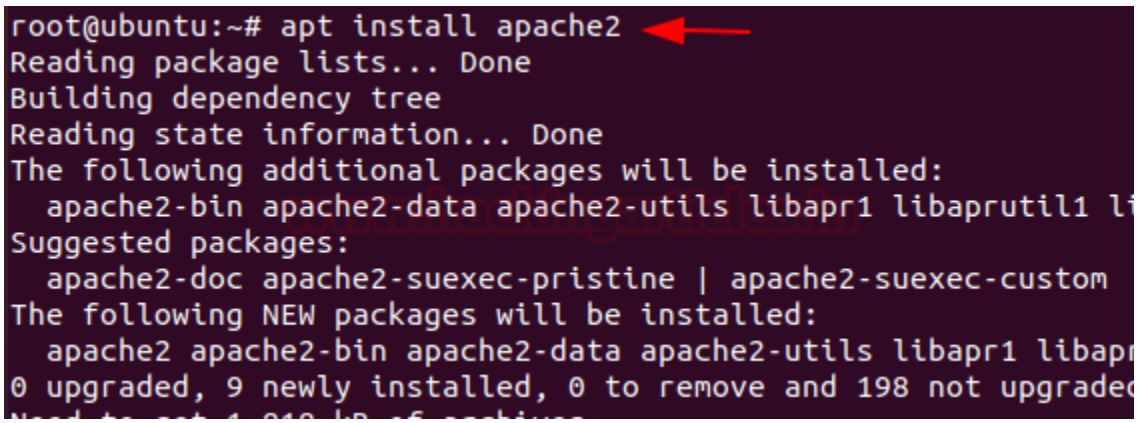
Prerequisites for WordPress

- Apache
- Database (MySQL/MariaDB)
- PHP

Install Apache

Let's start the HTTP service with the help of Apache using a privileged account (as root), execute the following command in the terminal.

```
apt install apache2
```



Install MySQL

To run WordPress, you will also need a database server. The database server is where WordPress content is saved. So, we are going to choose MariaDB-server as the required database for WordPress and execute the following command

```
apt install mariadb-server mariadb-client
```

```
root@ubuntu:~# apt install mariadb-server mariadb-client
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  galera-3 gawk libaio1 libcgi-fast-perl libcgi-pm-perl libconfig-inifiles-
  libterm-readkey-perl mariadb-client-10.3 mariadb-client-core-10.3 mariadb
Suggested packages:
  gawk-doc libclone-perl libmldbm-perl libnet-daemon-perl libsql-statement-
The following NEW packages will be installed:
  galera-3 gawk libaio1 libcgi-fast-perl libcgi-pm-perl libconfig-inifiles-
  libterm-readkey-perl mariadb-client mariadb-client-10.3 mariadb-client-co
0 upgraded, 22 newly installed, 0 to remove and 198 not upgraded.
```

Next, execute the following commands to protect remote root login for the database server.

```
mysql_secure_installation
```

Then respond to questions asked after the command has been executed.

- Enter current password for root (enter for none): **press Enter**
- Set root password? [Y/n]: **Y**
- New password: **Enter password**
- Re-enter new password: **Repeat password**
- Remove anonymous users? [Y/n]: **Y**
- Disallow root login remotely? [Y/n]: **Y**
- Remove the test database and access to it? [Y/n]: **Y**
- Reload privilege tables now? [Y/n]: **Y**

```
root@ubuntu:~# mysql_secure_installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user.  If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

Set 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
```

Install php

And at last, install the PHP MySQL and run the following command to install this application.

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

```
root@ubuntu:~# apt install php php-mysql
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libapache2-mod-php7.4 php-common php7.4 php7.4-cli php7.4-com
Suggested packages:
  php-pear
The following NEW packages will be installed:
  libapache2-mod-php7.4 php php-common php-mysql php7.4 php7.4-
0 upgraded, 11 newly installed, 0 to remove and 198 not upgrade
```

Create a Database for WordPress

To access MySQL, enter the following command, which will create a database for WordPress.

```
mysql -u root -p
CREATE DATABASE wordpress;
CREATE USER 'wp_user'@'localhost' IDENTIFIED BY 'password';
GRANT ALL ON wordpress.* TO 'wp_user'@'localhost' IDENTIFIED BY 'password';
FLUSH PRIVILEGES;
exit
```

```
root@ubuntu:~# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 55
Server version: 10.3.22-MariaDB-1ubuntu1 Ubuntu 20.04

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)]> CREATE USER 'wp_user'@'localhost' IDENTIFIED BY 'password';
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> GRANT ALL ON wordpress.* TO 'wp_user'@'localhost' IDENTIFIED BY 'password';
Query OK, 0 rows affected (0.000 sec)

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

MariaDB [(none)]> exit
Bye
root@ubuntu:~#
```

WordPress Installation & Configuration

Now, it's time to download and install WordPress on our localhost. With the help of the wget command, we have fetched the compressed file of WordPress setup and extracted the folder inside the /var/www/html directory.

```
cd /var/www/html
wget http://www.wordpress.org/latest.tar.gz
tar –xvf latest.tar.gz
```

```
root@ubuntu:/var/www/html# wget https://wordpress.org/latest.tar.gz
--2020-06-30 11:12:06--  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: 12238031 (12M) [application/octet-stream]
Saving to: 'latest.tar.gz'

latest.tar.gz                                       100%[=============================

2020-06-30 11:12:43 (412 KB/s) - 'latest.tar.gz' saved [12238031/12238031]

root@ubuntu:/var/www/html# ls
index.html  latest.tar.gz
root@ubuntu:/var/www/html# tar -xvf latest.tar.gz
wordpress/
wordpress/xmlrpc.php
wordpress/wp-blog-header.php
wordpress/readme.html
wordpress/wp-signup.php
```

Then run the given command to change ownership of the 'wordpress' directory as well permission for the upload directory.

```
chown -R www-data:www-data wordpress/
chmod -R 755  wordpress/
mkdir wordpress/wp-content/uploads
chown -R  www-data:www-data wordpress/wp-content/uploads
```

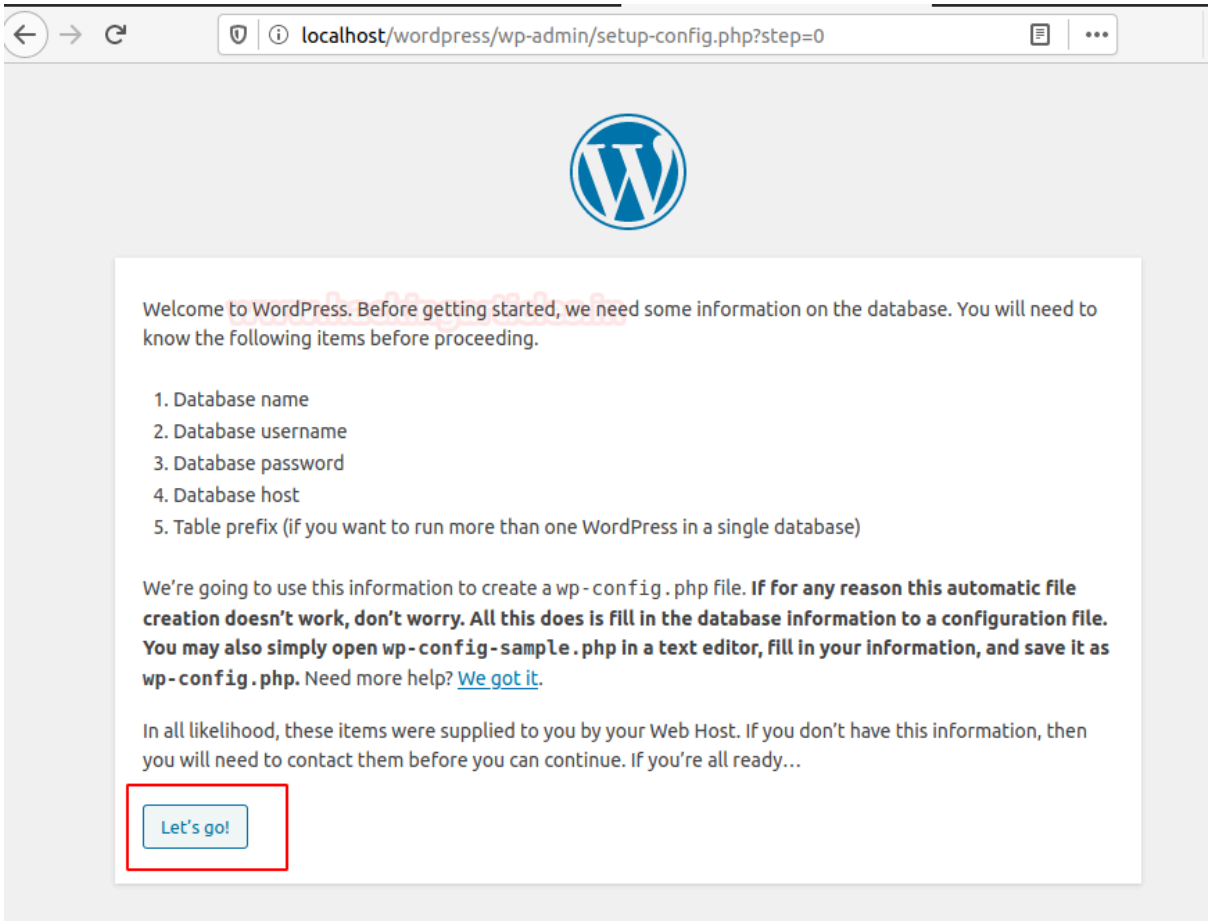
```
root@ubuntu:/var/www/html# chown -R www-data:www-data wordpress/
root@ubuntu:/var/www/html# chmod -R 755 wordpress/
root@ubuntu:/var/www/html# mkdir wordpress/wp-content/uploads
root@ubuntu:/var/www/html# chown -R www-data:www-data wordpress/wp-content/uploads/
root@ubuntu:/var/www/html#
```

Now, we have completed the installation up to this point; to create a WordPress website, we need to access the application over the web browser on localhost by executing the following steps, and then complete the remaining installation process.

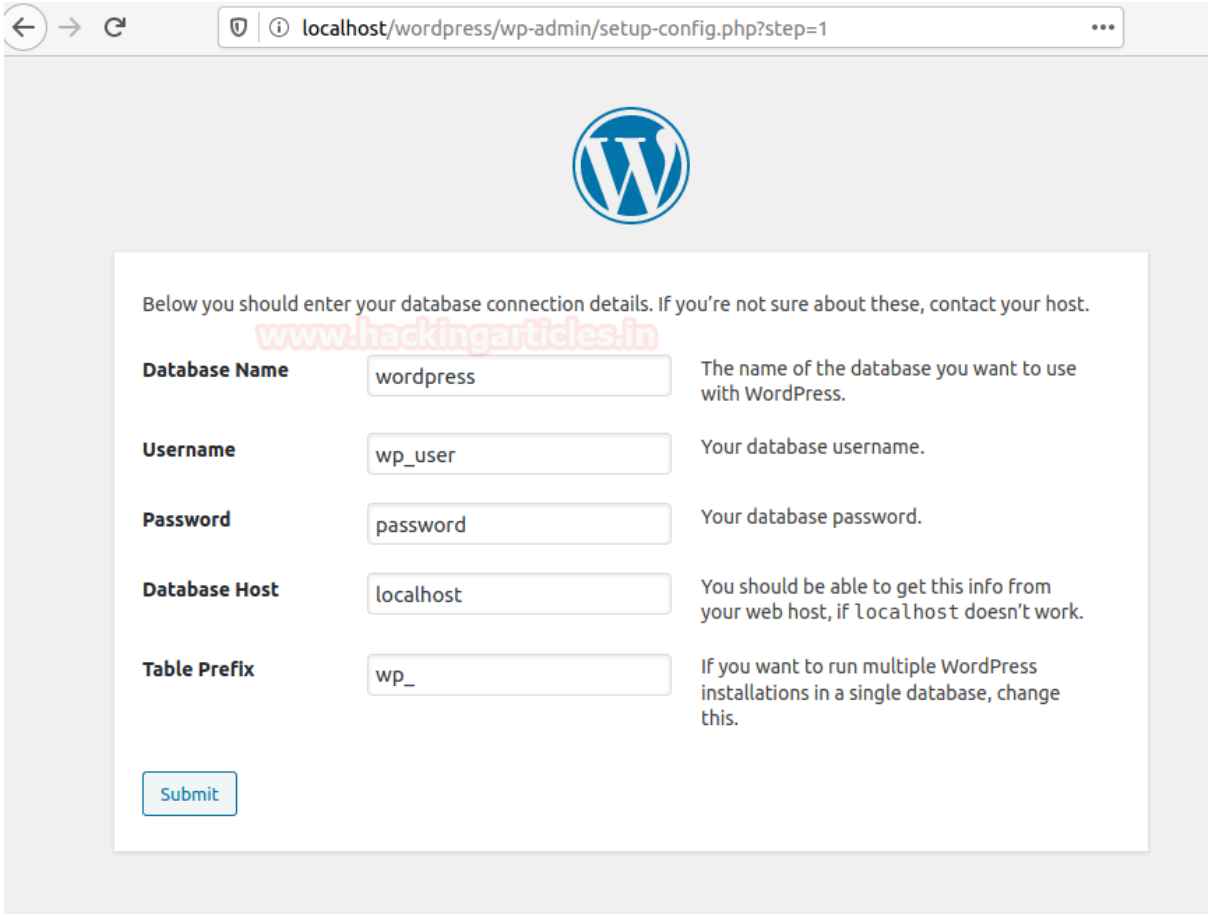
```
http://localhost/wordpress/
```

This will open the setup file and ask you to choose your preferred language. I select **English** and then press the **continue** button.

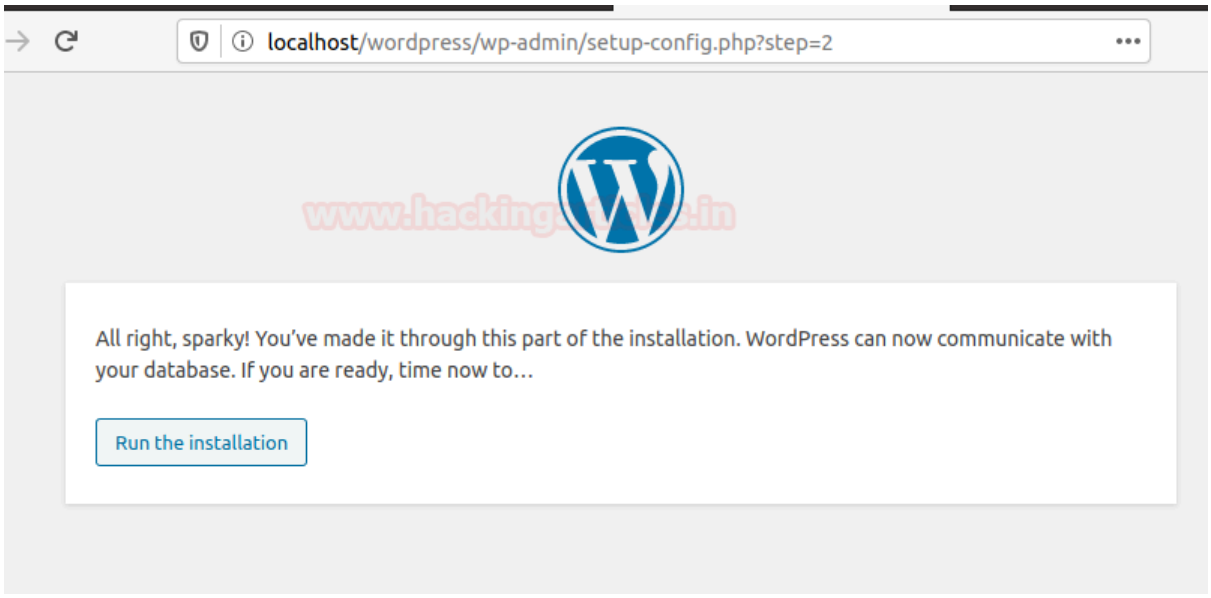
Read the given content and press Let's go to continue the activity.



To continue the activity, we need to enter the required details that will help the application to connect with the database, thus it should be the same information that we have entered above at the time of database we have created for WordPress.

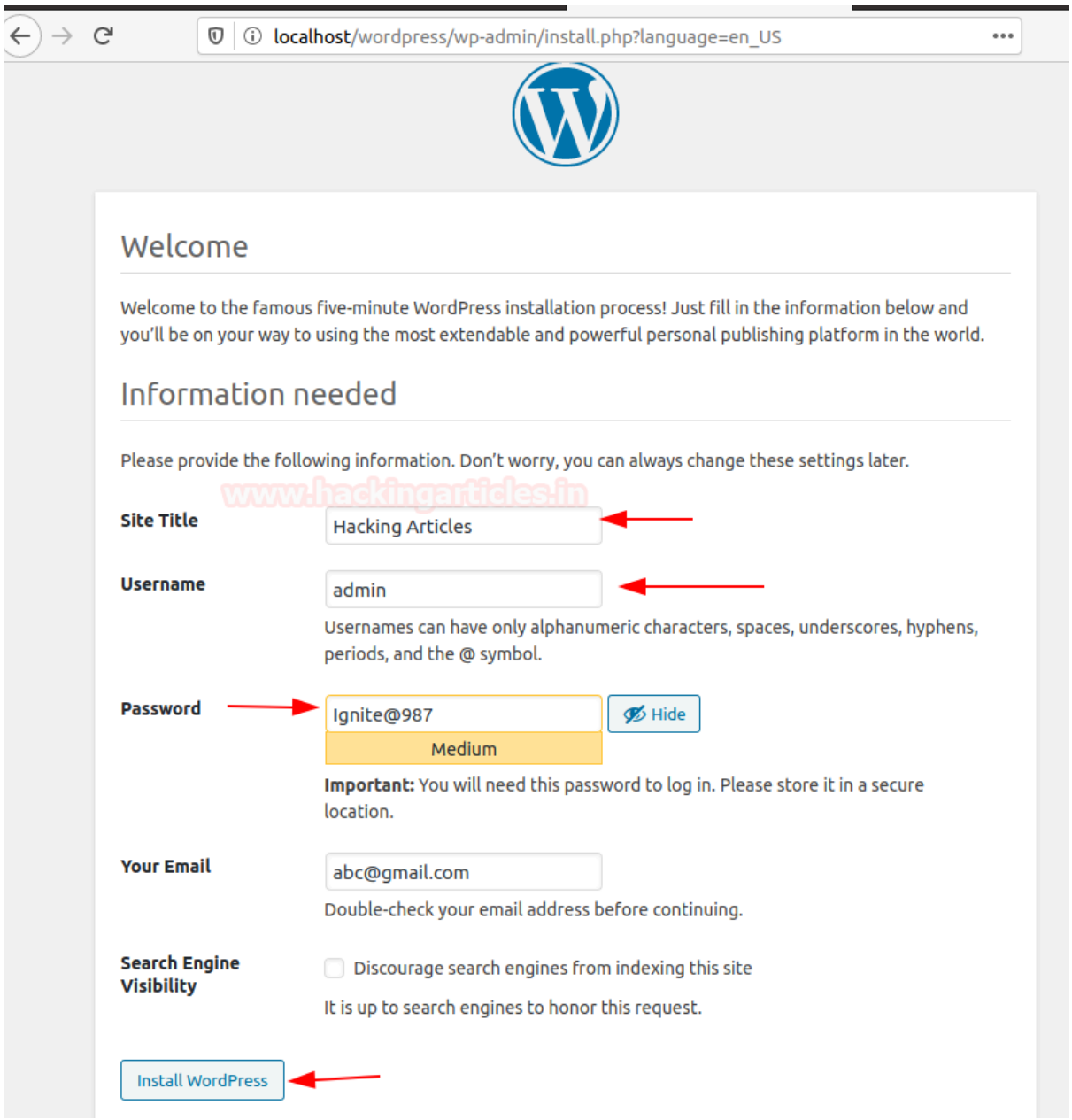


And if your above-given detail is correct, you will get the Installation page as we have here.

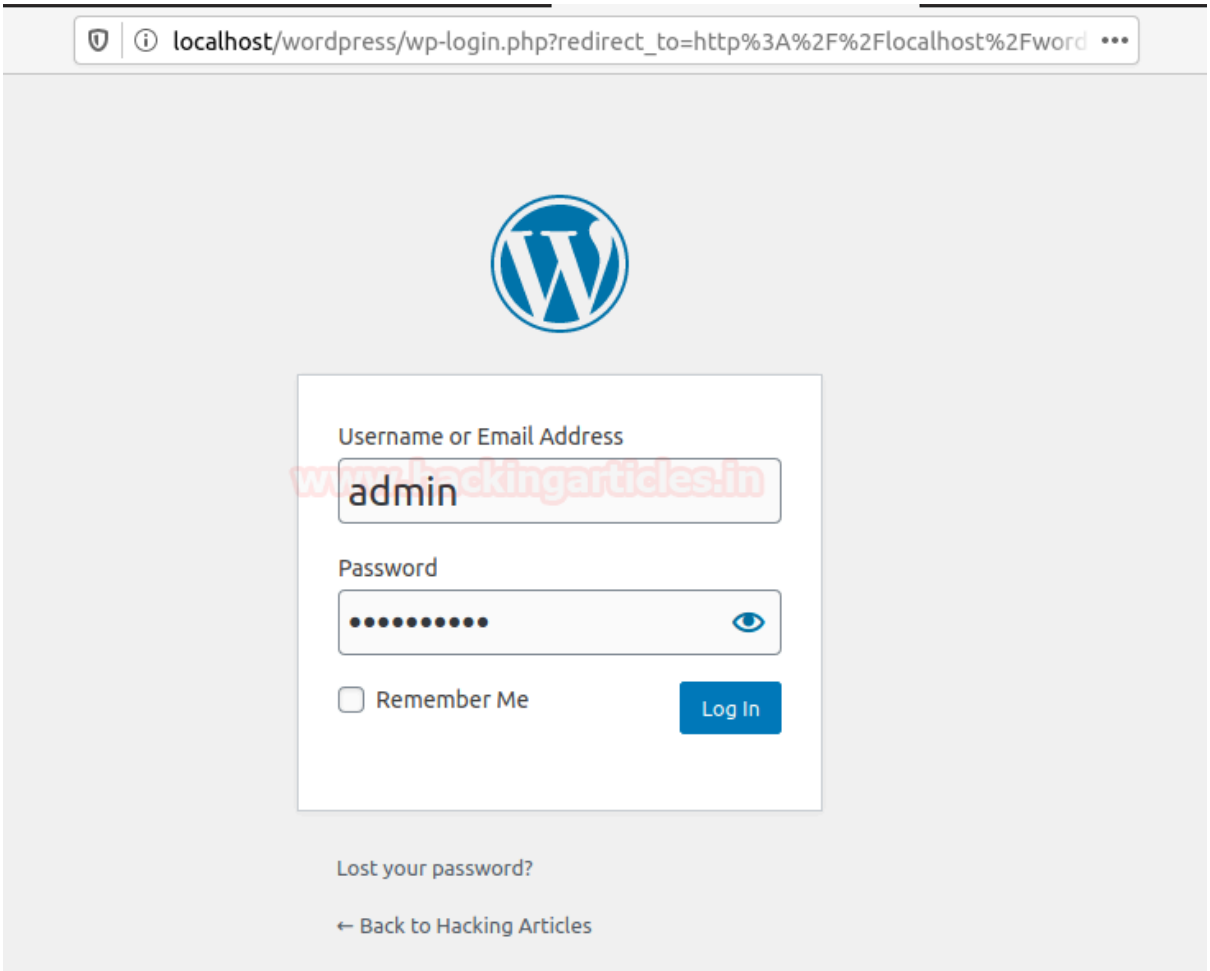


Now, after that, it will ask you to enter details for your Website which you want to host using WordPress CMS as shown in the image below, and then finally click on the install Tab.

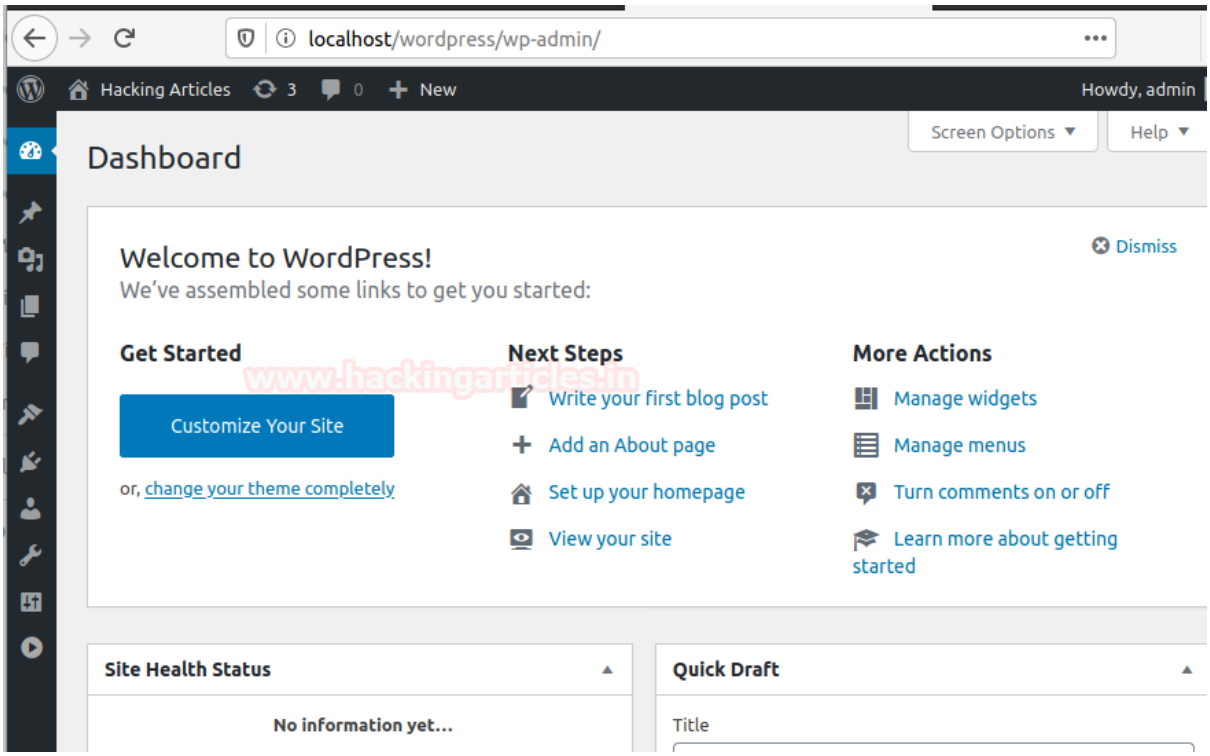
Note: The User and Password asked before the installation is referred to your Database information, and the username and password asked after installed are referred to your application (CMS).



And once it is done, you will get the application login page where you must enter credentials to access the dashboard of your CMS.

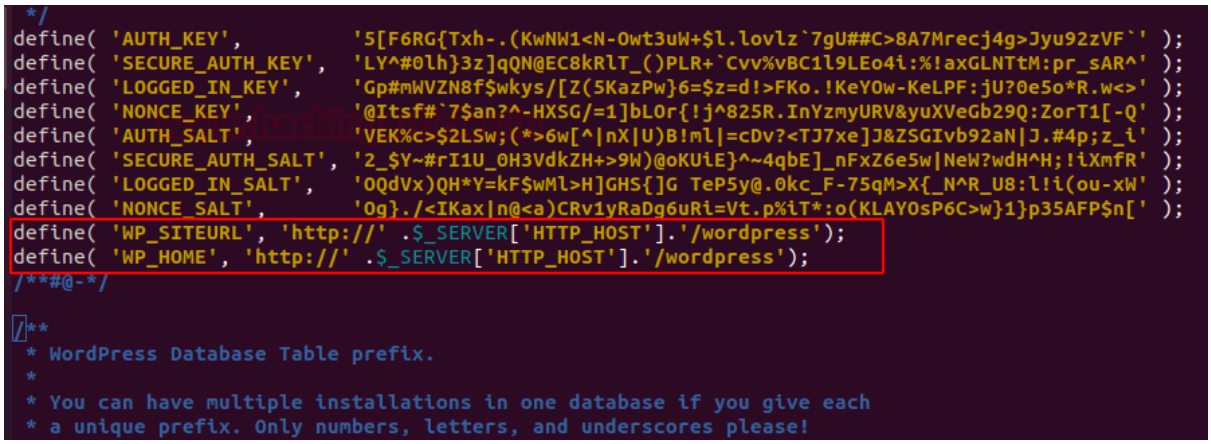


You will get the dashboard where you can write your content to be posted on the website.

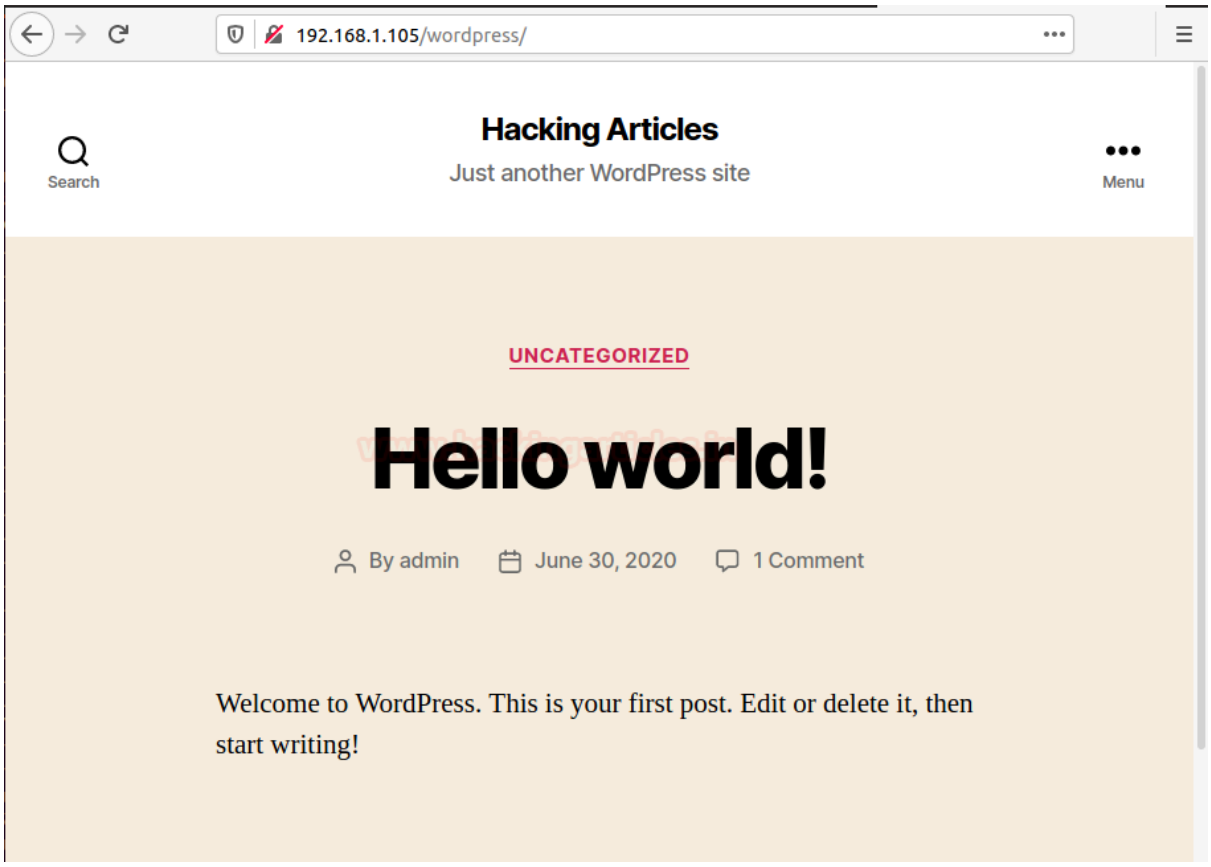


Open the wp-config.php file in WordPress directory and paste the following lines in it to access the website page.

```
define( 'WP_SITEURL', 'http://' .$_SERVER['HTTP_HOST'].'/wordpress');
define( 'WP_HOME', 'http://' .$_SERVER['HTTP_HOST'].'/wordpress');
```



And finally, it is over here, and your WordPress is completely ready to go 😊 .

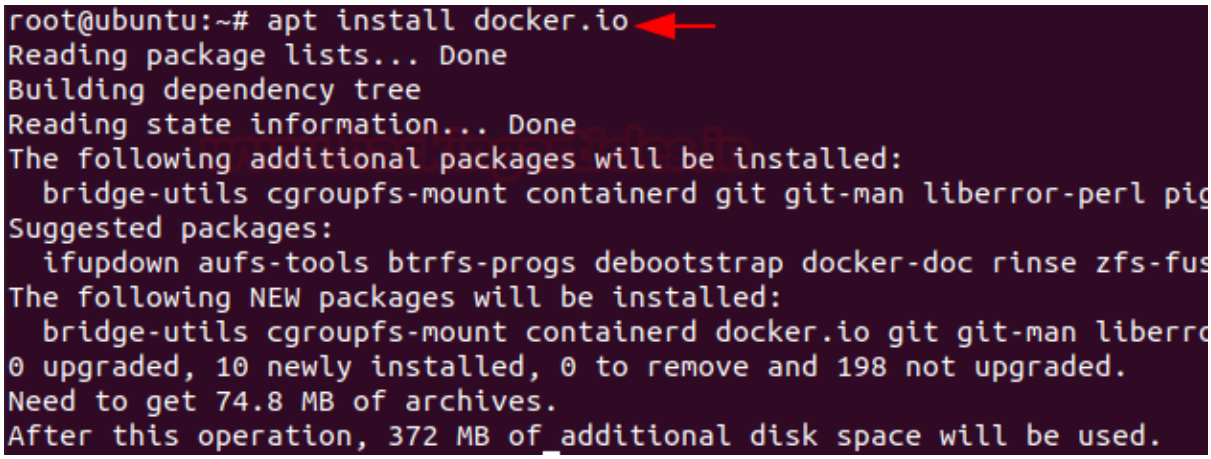


Install WordPress using Docker

Installing WordPress through Docker will release your effort of installing prerequisites for WordPress setup. It is a very easy and quick technique to configured WordPress. All you need to have some basic knowledge of Docker and its functionalities.

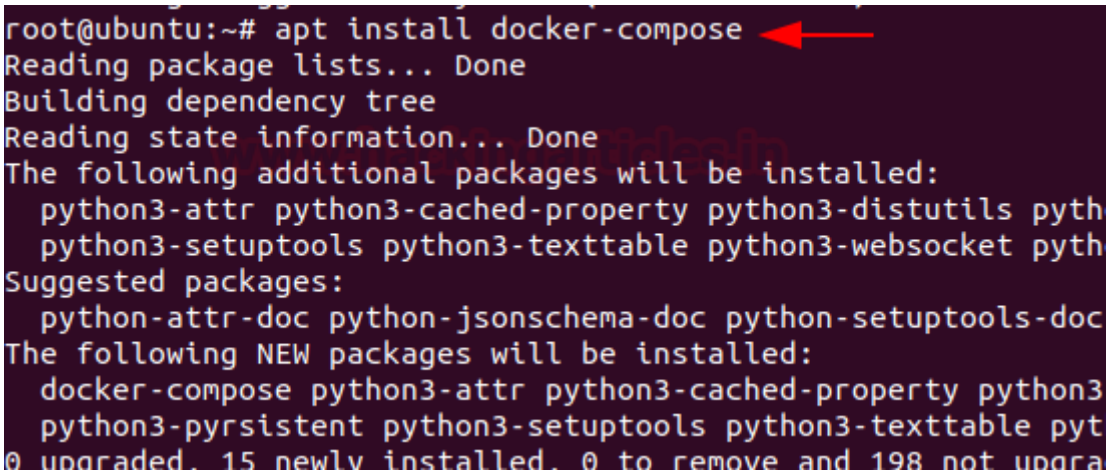
To install wordpress using docker, first, we will update the Ubuntu repository and then install the latest version of docker.io. Let's start the installation of docker packages with the apt command as below:

```
apt install docker.io
```



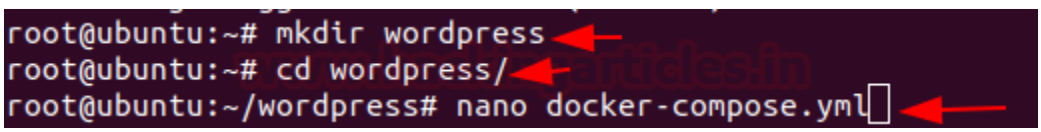
Docker Compose is used to run multiple containers as a single service. Let's begin the installation of docker-compose with the help of apt by entering the following command.

```
apt install docker-compose
```



After installing the composer for the Docker, we must create a directory by the name of Wordpress. After creating the directory, we will create a .yml file that will contain the service definitions for your setup.

```
mkdir wordpress
cd wordpress/
nano docker-compose.yml
```



Now, paste the following text in the .yml and save the configuration. Source Code From **here**

```
version: '3.3' services:
db:
image: mysql:5.7
volumes:
- db_data:/var/lib/mysql
restart: always
environment:
MYSQL_ROOT_PASSWORD: somewordpress
MYSQL_DATABASE: wordpress
MYSQL_USER: wordpress
MYSQL_PASSWORD: wordpress

wordpress:
depends_on:
- db
image: wordpress:latest
ports:
- "8000:80"
restart: always
environment:
WORDPRESS_DB_HOST: db:3306
WORDPRESS_DB_USER: wordpress
WORDPRESS_DB_PASSWORD: wordpress
WORDPRESS_DB_NAME: wordpress
volumes:
db_data: {}
```

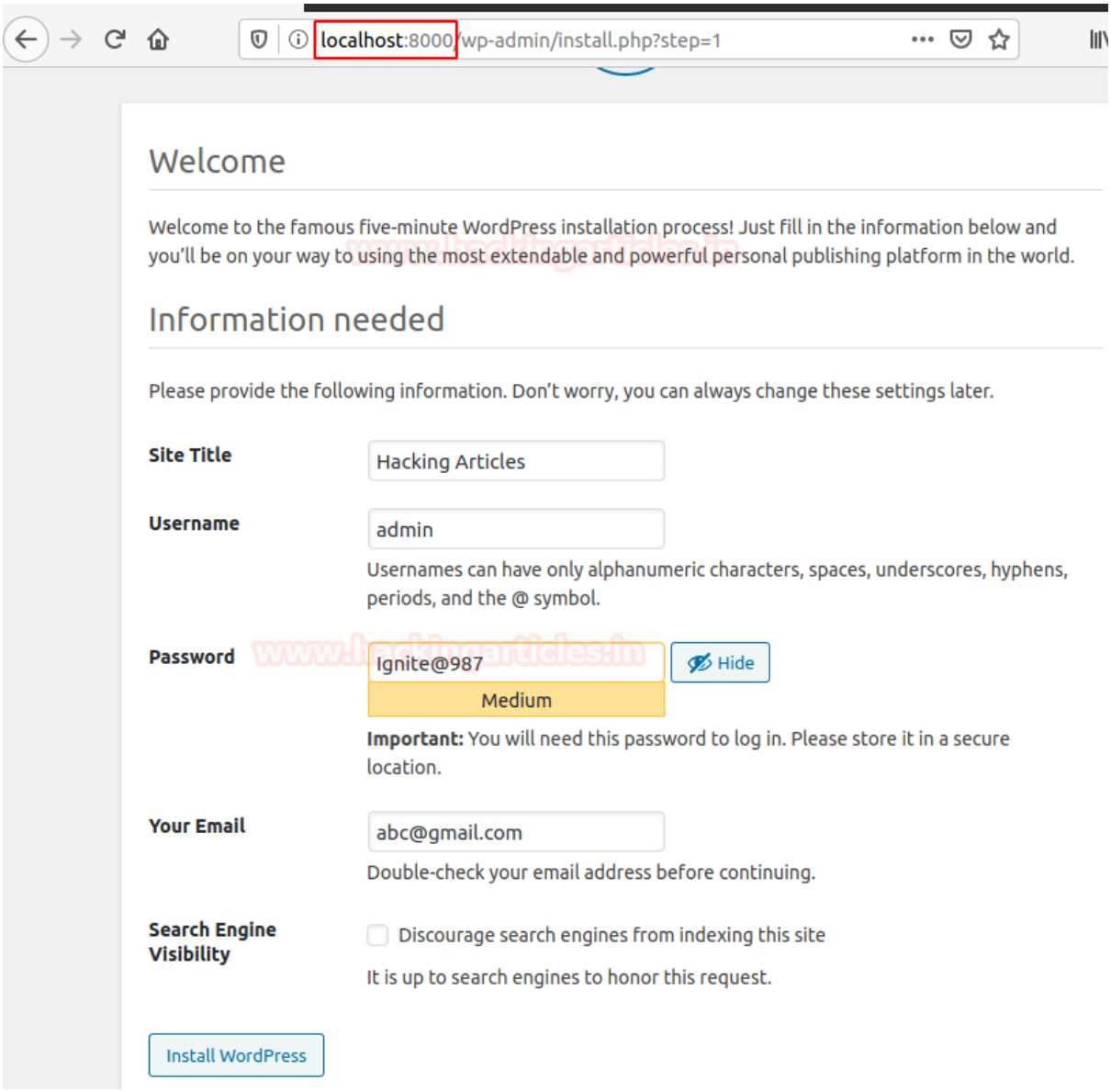
```
  GNU nano 4.8
version: '3.3'

services:
   db:
     image: mysql:5.7
     volumes:
       - db_data:/var/lib/mysql
     restart: always
     environment:
       MYSQL_ROOT_PASSWORD: somewordpress
       MYSQL_DATABASE: wordpress
       MYSQL_USER: wordpress
       MYSQL_PASSWORD: wordpress

   wordpress:
     depends_on:
       - db
     image: wordpress:latest
     ports:
       - "8000:80"
     restart: always
     environment:
       WORDPRESS_DB_HOST: db:3306
       WORDPRESS_DB_USER: wordpress
       WORDPRESS_DB_PASSWORD: wordpress
       WORDPRESS_DB_NAME: wordpress
volumes:
    db_data: {}
```

Now run the Docker image in detached mode using the following command

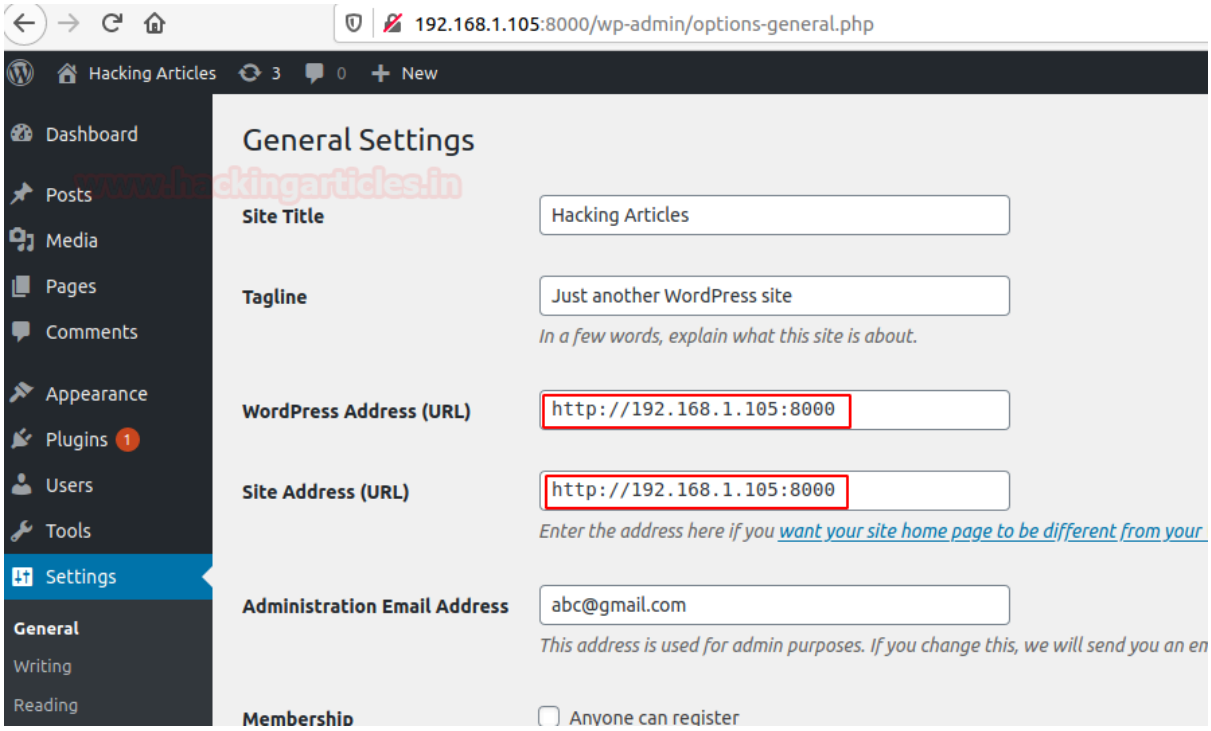
```
docker–compose up -d
```

```
root@ubuntu:~/wordpress# docker-compose up -d
Creating network "wordpress_default" with the default driver
Creating volume "wordpress_db_data" with default driver
Pulling db (mysql:5.7)...
5.7: Pulling from library/mysql
8559a31e96f4: Downloading [======================>
d51ce1c2e575: Download complete
c2344adc4858: Download complete
fcf3ceff18fc: Download complete
16da0c38dc5b: Download complete
b905d1797e97: Downloading [=========================================>
4b50d1c6b05c: Download complete
d85174a87144: Download complete
a4ad33703fa8: Downloading [==>
f7a5433ce20d: Waiting
3dcd2a278b4a: Waiting
```

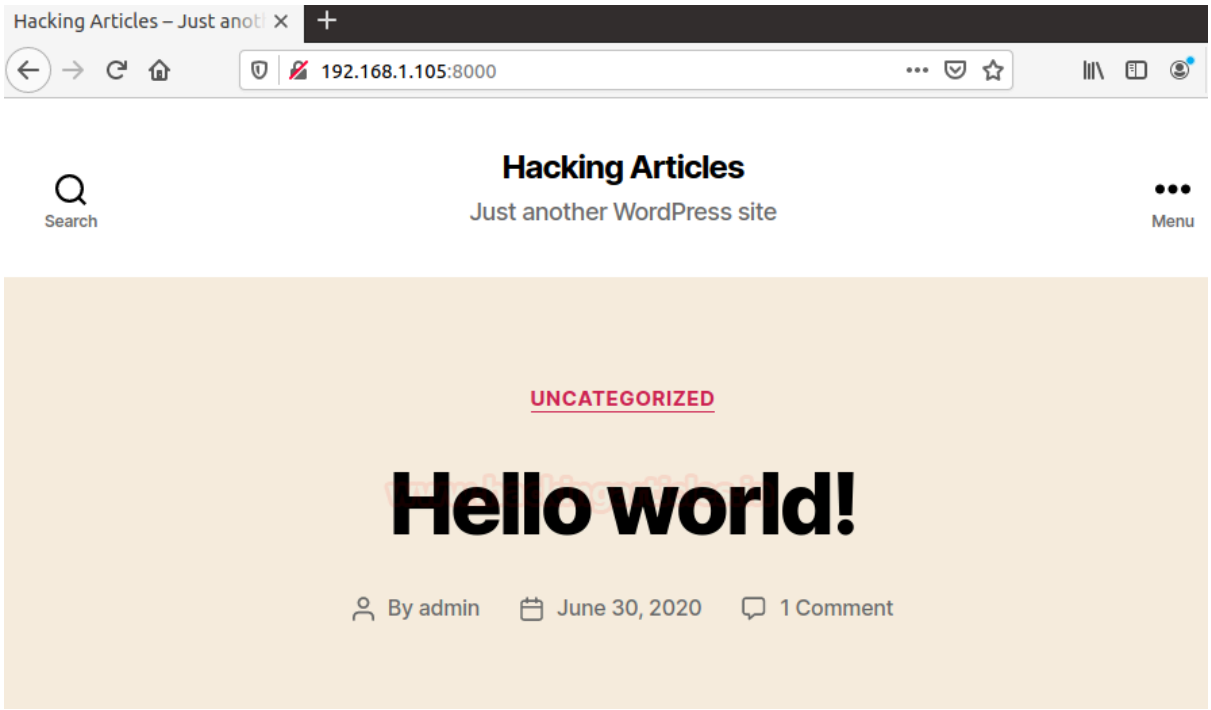
After doing all the configuration step-by-step, now access the localhost on port 8000 that will be hosting your WordPress Docker image and configure your WordPress site as done in the previous section.



You will get the dashboard where you can write your content that to be posted on the website. But here we need to make some changes inside the **setting** so that the wordpress after installation it will work properly. Thus, enter your localhost IP address with a port number on which your docker image is running.

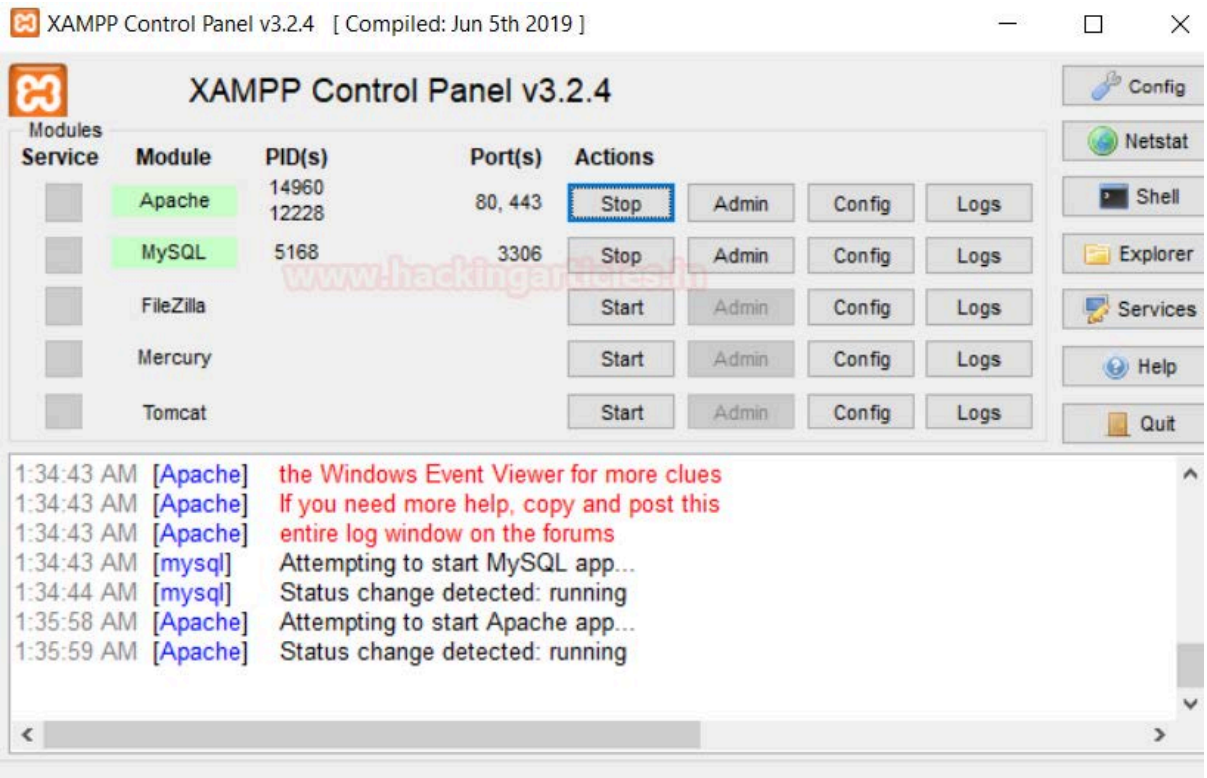


And finally, it is over here, and your WordPress is completely ready to go but over port 8000 as shown here 😊.

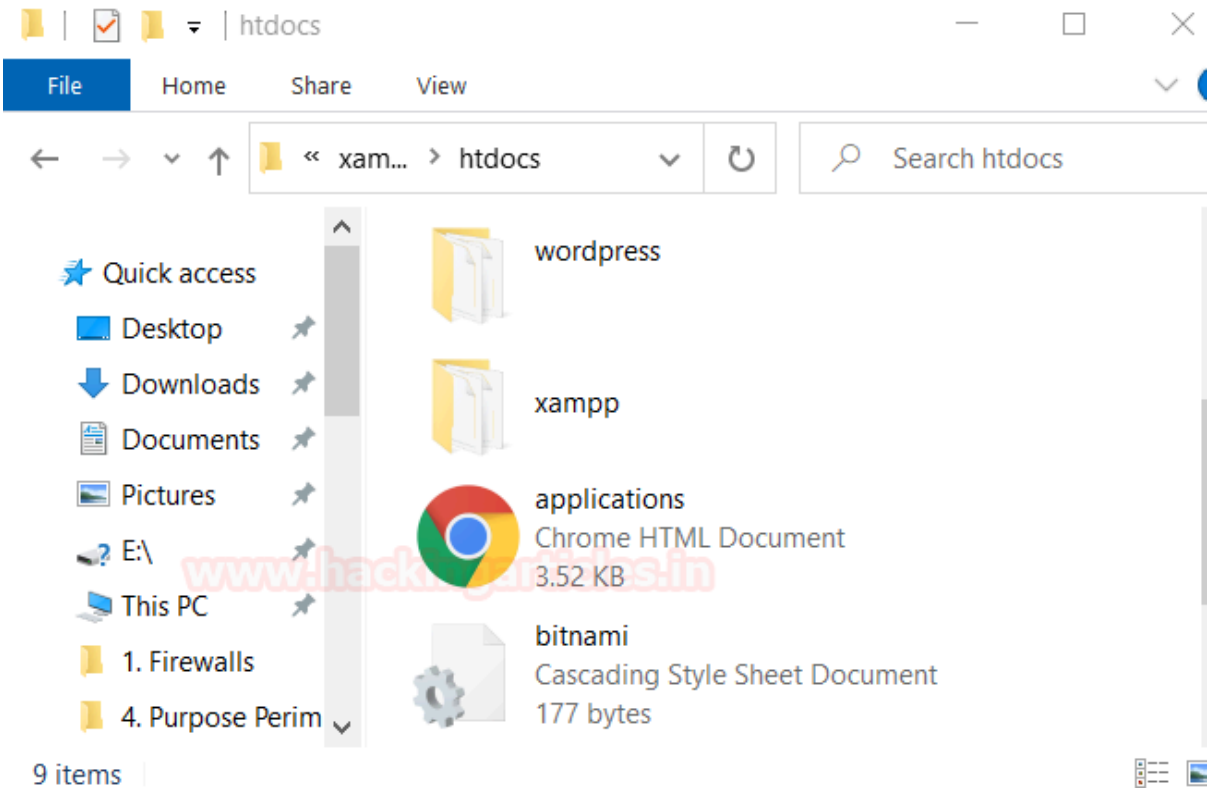


Install WordPress on Windows Platform

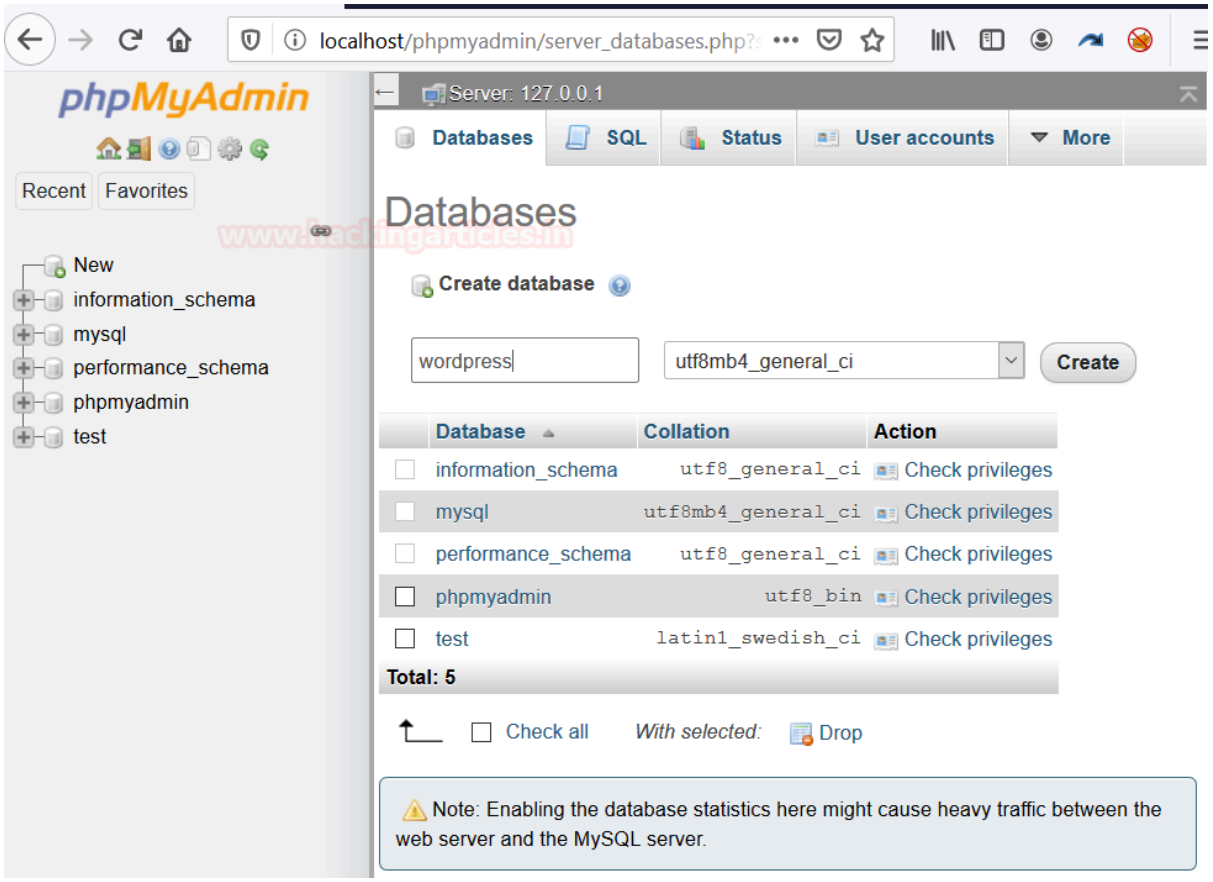
Installation of WordPress is also very easy as compared to Ubuntu because to fulfil the prerequisites of LAMP Server, we can use XAMPP, which will complete all the required dependencies like Apache and MySQL for WordPress.



Now download the extracted zip file of WordPress inside the /htdocs folder in /xampp folder in C-Drive.



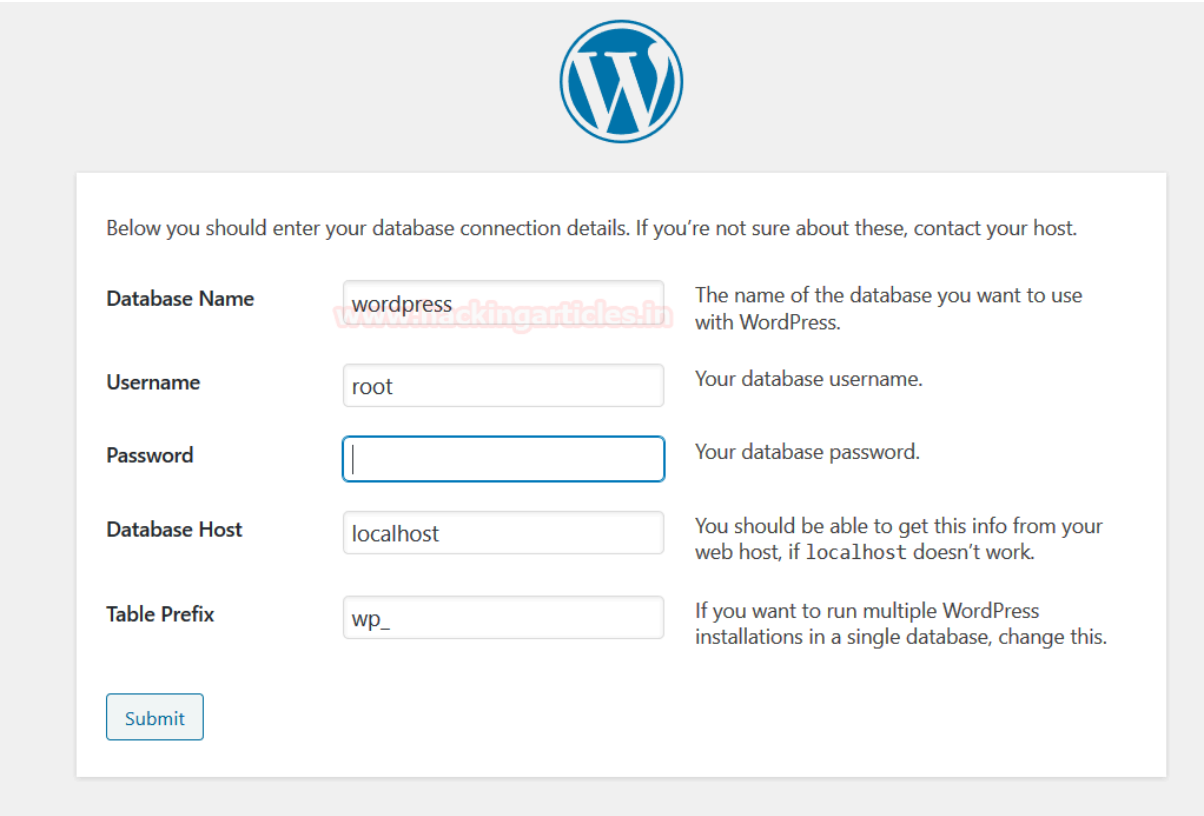
Then, open the PHPMYADMIN in a web browser by accessing /localhost/phpMyAdmin and create the database for Wordpress to store its data.



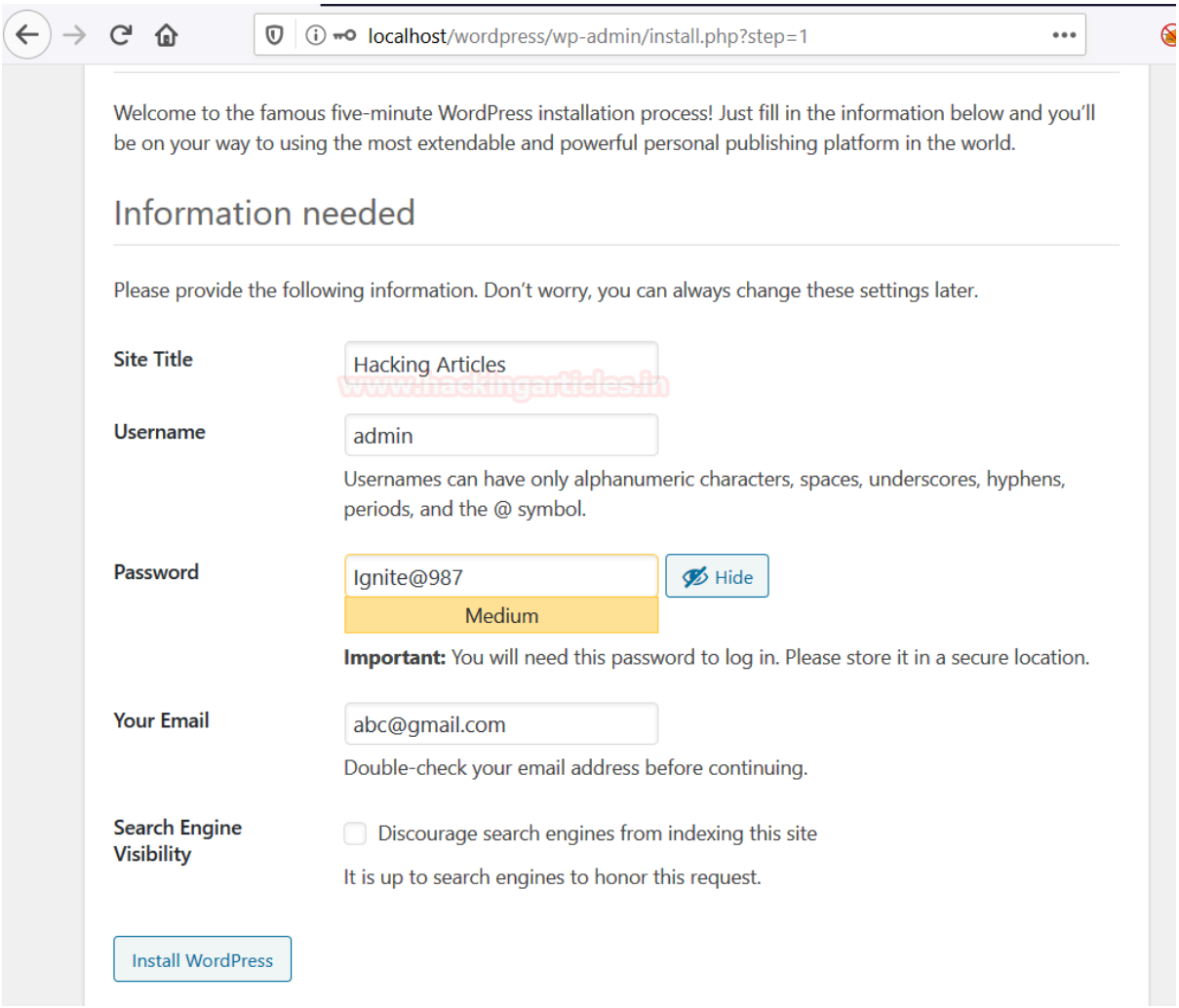
Now to configure wordpress, explore the /localhost/wordpress/ and then enter the detail for the database.

Note: By Default, XAMPP DB_User is root and DB_Pass is empty <blank>

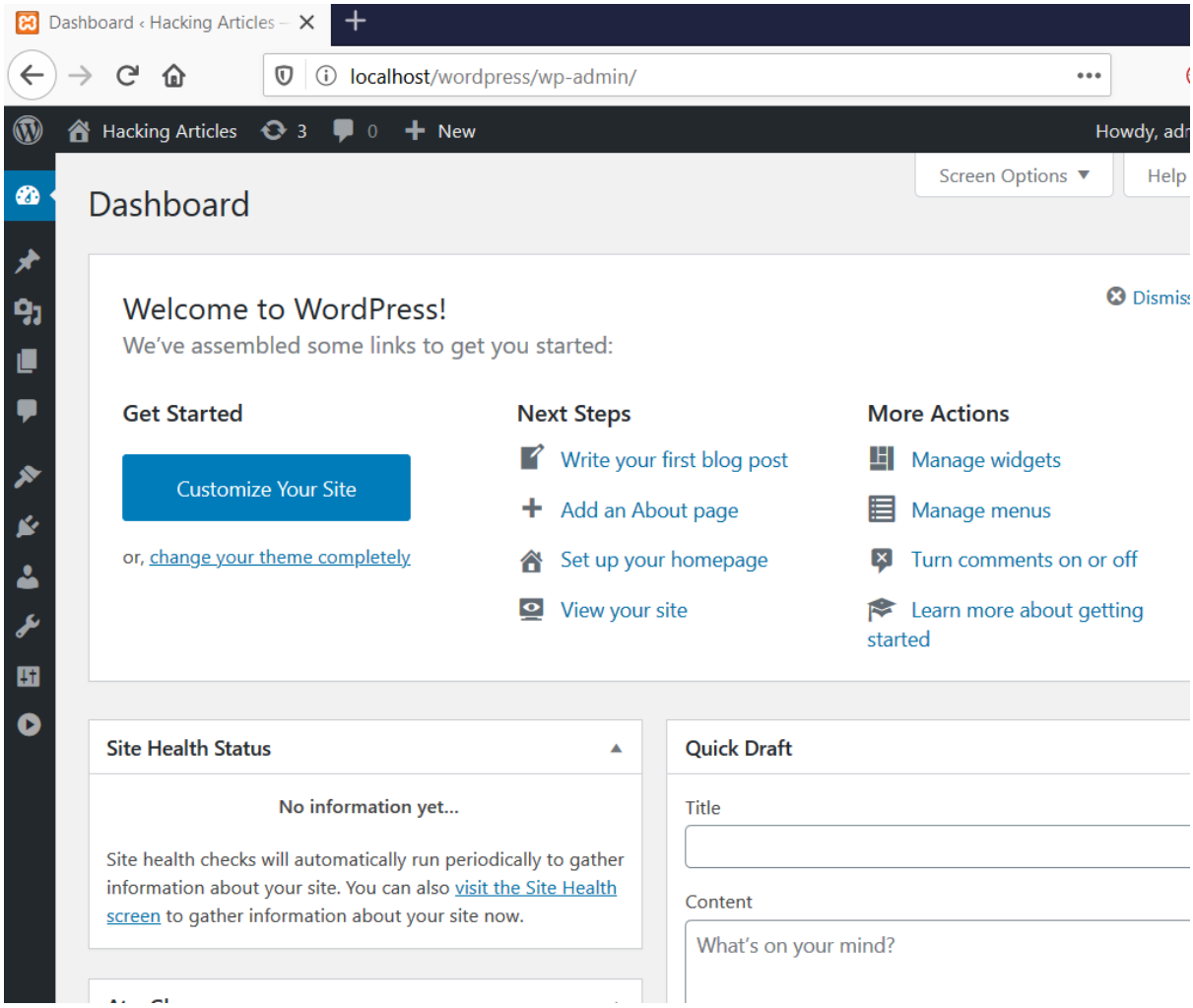
So as per XMAPP database configuration, we entered the following details in the given record.



Now again repeat the same step as done in the above section.



You will get the dashboard where you can write your content that to be posted on the website.

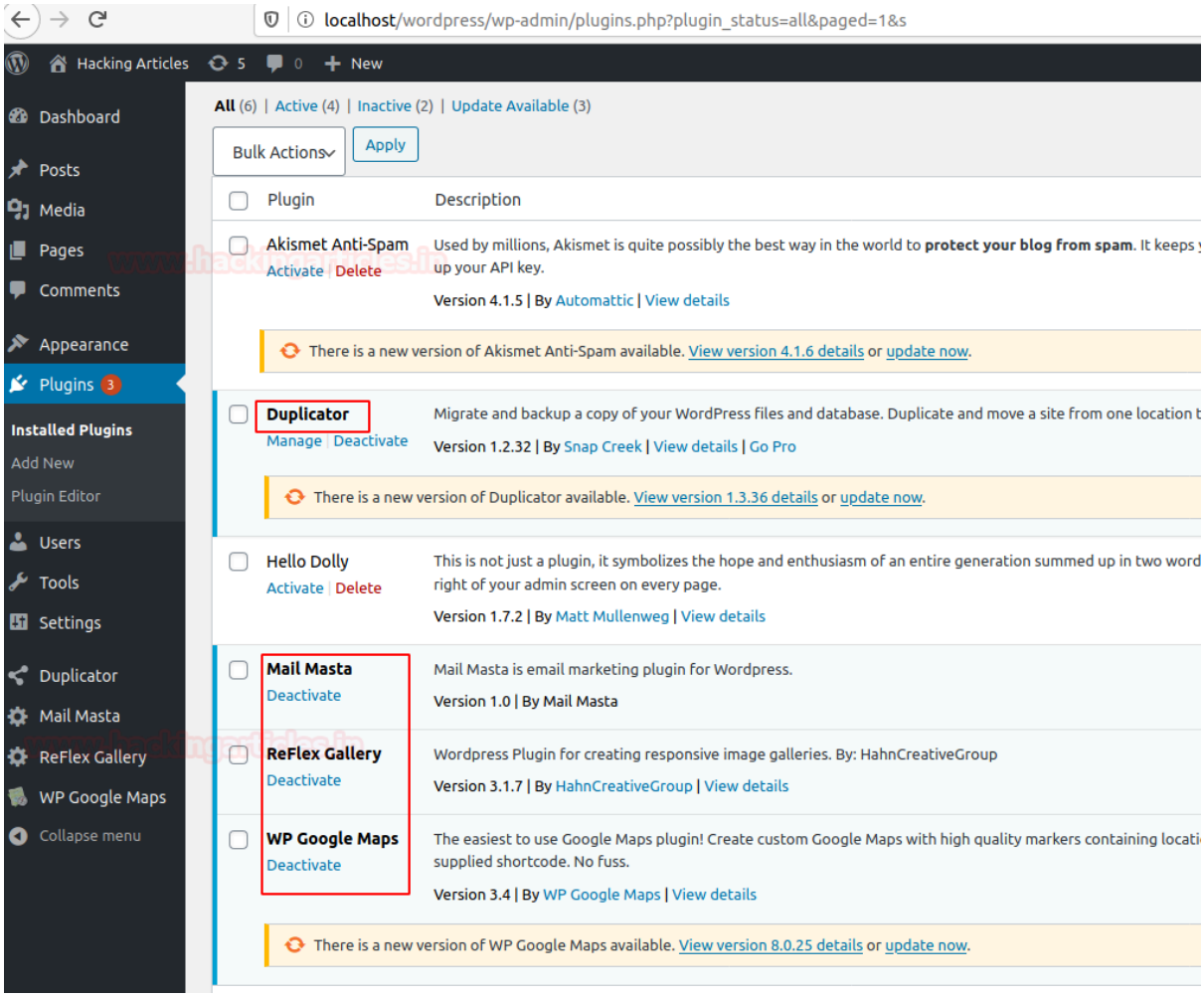


To make it vulnerable WordPress platform in order to perform penetration testing I have installed some vulnerable plugin as highlighted in the image.

To know how we can go do WordPress Penetration testing, read **this** article.

WordPress Vulnerable Plugin

- **https://www.exploit-db.com/exploits/40290**
- **https://www.exploit-db.com/exploits/36374**
- **https://www.exploit-db.com/exploits/44883**


localhost/wordpress/wp-admin/plugins.php?plugin_status=all&paged=1&s
Hacking Articles
5
0
New
Dashboard
Posts
Media
Pages
Comments
Appearance
Plugins 3
Installed Plugins
Add New
Plugin Editor
Users
Tools
Settings
Duplicator
Mail Masta
ReFlex Gallery
WP Google Maps
Collapse menu
All (6) | Active (4) | Inactive (2) | Update Available (3)
Bulk Actions
Apply
Plugin
Description
Akismet Anti-Spam
Activate | Delete
Used by millions, Akismet is quite possibly the best way in the world to protect your blog from spam. It keeps
up your API key.
Version 4.1.5 | By Automattic | View details
There is a new version of Akismet Anti-Spam available. View version 4.1.6 details or update now.
Duplicator
Manage | Deactivate
Migrate and backup a copy of your WordPress files and database. Duplicate and move a site from one location t
Version 1.2.32 | By Snap Creek | View details | Go Pro
There is a new version of Duplicator available. View version 1.3.36 details or update now.
Hello Dolly
Activate | Delete
This is not just a plugin, it symbolizes the hope and enthusiasm of an entire generation summed up in two word
right of your admin screen on every page.
Version 1.7.2 | By Matt Mullenweg | View details
Mail Masta
Deactivate
Mail Masta is email marketing plugin for Wordpress.
Version 1.0 | By Mail Masta
ReFlex Gallery
Deactivate
Wordpress Plugin for creating responsive image galleries. By: HahnCreativeGroup
Version 3.1.7 | By HahnCreativeGroup | View details
WP Google Maps
Deactivate
The easiest to use Google Maps plugin! Create custom Google Maps with high quality markers containing locati
supplied shortcode. No fuss.
Version 3.4 | By WP Google Maps | View details
There is a new version of WP Google Maps available. View version 8.0.25 details or update now.

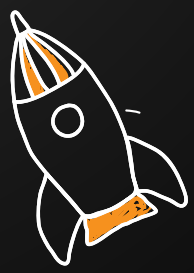
iGNITE Technologies

JOIN OUR TRAINING PROGRAMS

CLICK HERE

BEGINNER

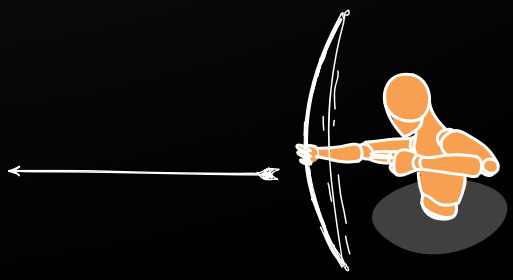
- Ethical Hacking
- Network Pentest
- Bug Bounty
- Wireless Pentest
- Network Security Essentials



ADVANCED

- Burp Suite Pro
- Android Pentest
- Web Services-API
- Advanced Metasploit
- Pro Infrastructure VAPT
- CTF
- Computer Forensics

EXPERT



- Red Team Operation
  - APT's - MITRE Attack Tactics
  - Active Directory Attack
  - MSSQL Security Assessment
- Privilege Escalation
  - Windows
  - Linux