---Server Setup---

Setting Up an Ubuntu 20.04 server for Deployment.

When you first create a server from any provider like Linode, Digital Ocean, etc., you will have to secure it by executing several commands. This post will explain what those commands are and why you need to run all those commands to start using your server securely.

I will break down the procedure we will perform into 5 simple steps:

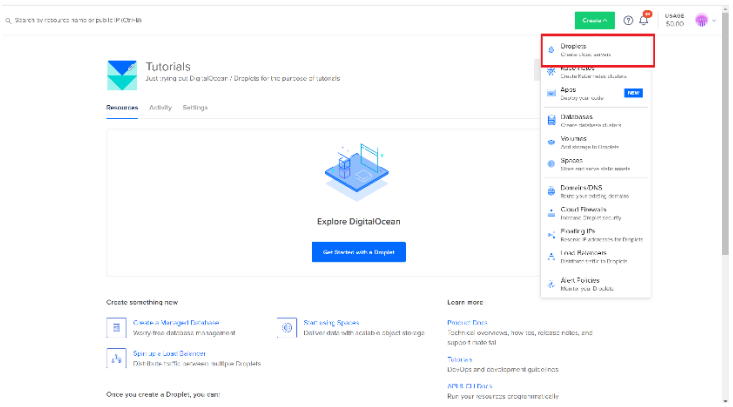
1. Creating a VPS
2. Initial Login
3. Creating a non-root user
4. Firewall setup

Let's get started

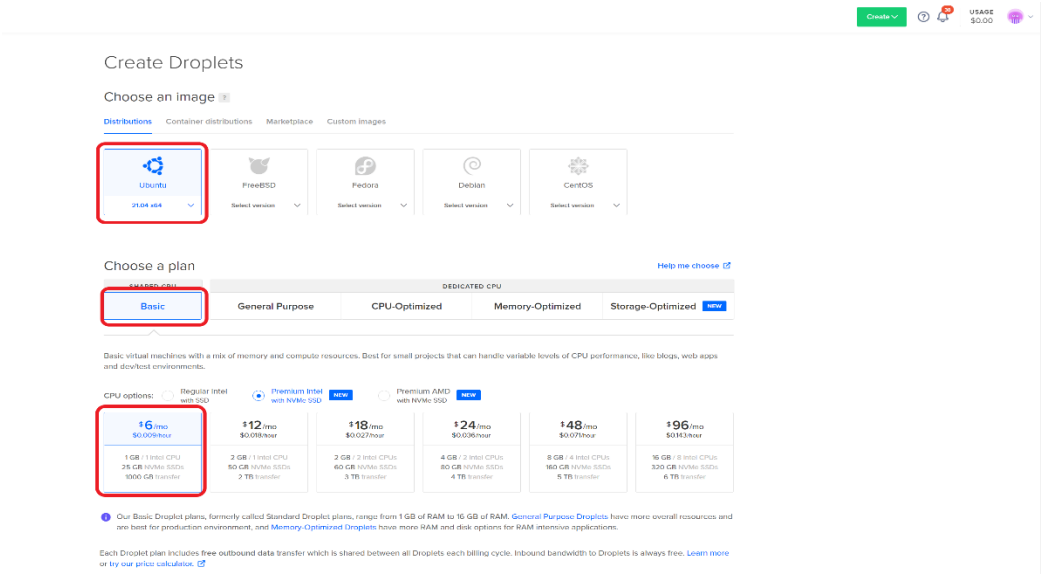
Creating a VPS

For creating a VPS, I will choose DigitalOcean. I have been DigitalOcean for quite a while now, and it never disappointed me. There are other service providers like Linode, Vultr, etc.

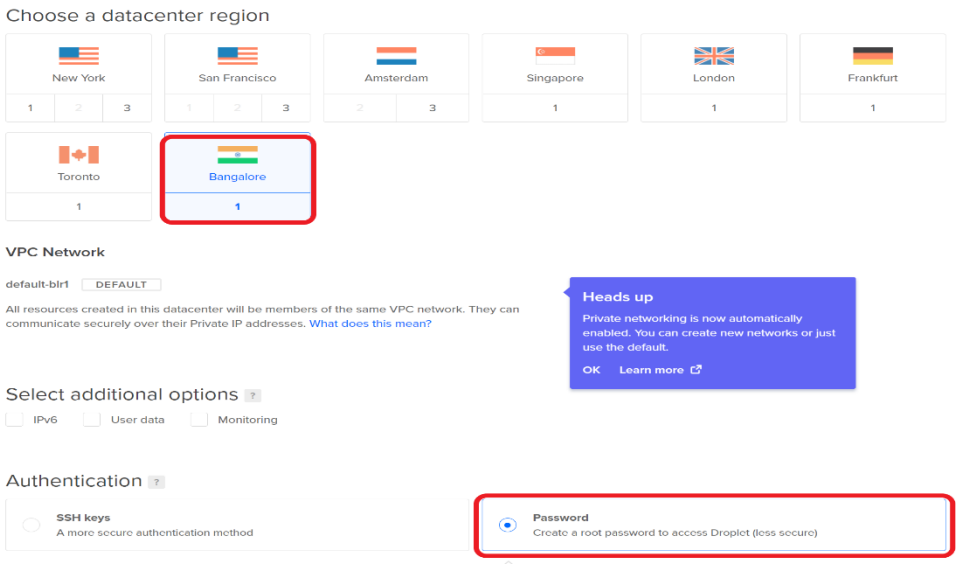
Once you create an account on DigitalOcean and log in, DigitalOcean will present you with this screen. Click on create and then on droplets:



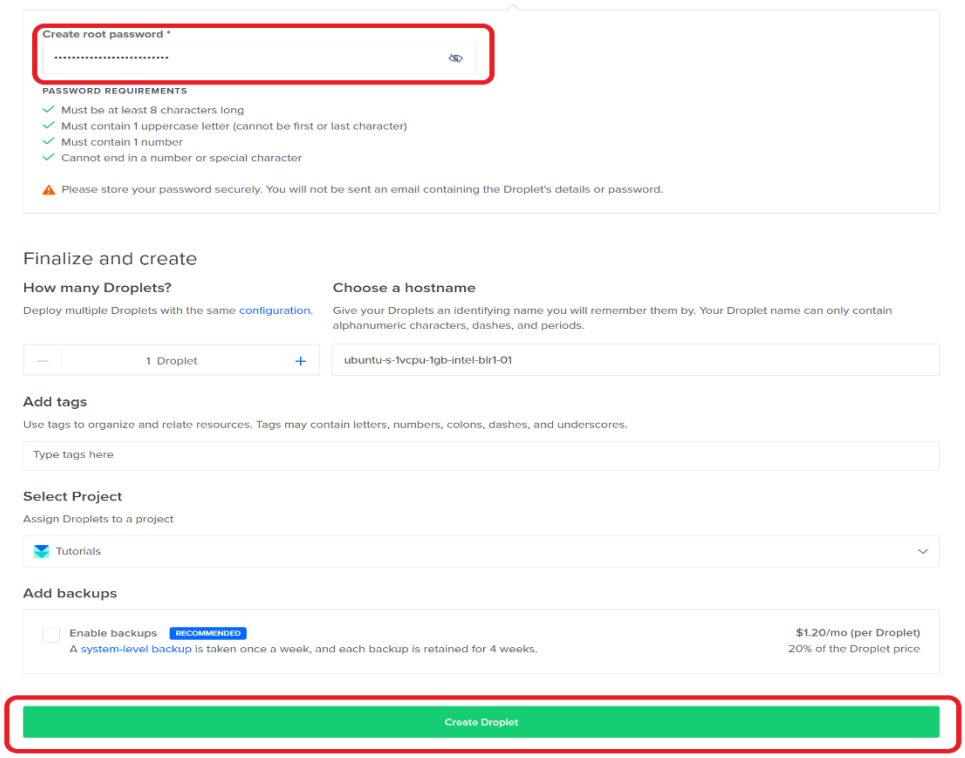
DigitalOcean will present you with the following screen, where you can choose the droplet, Operating system, and plan size. I will choose the options as marked in red below:



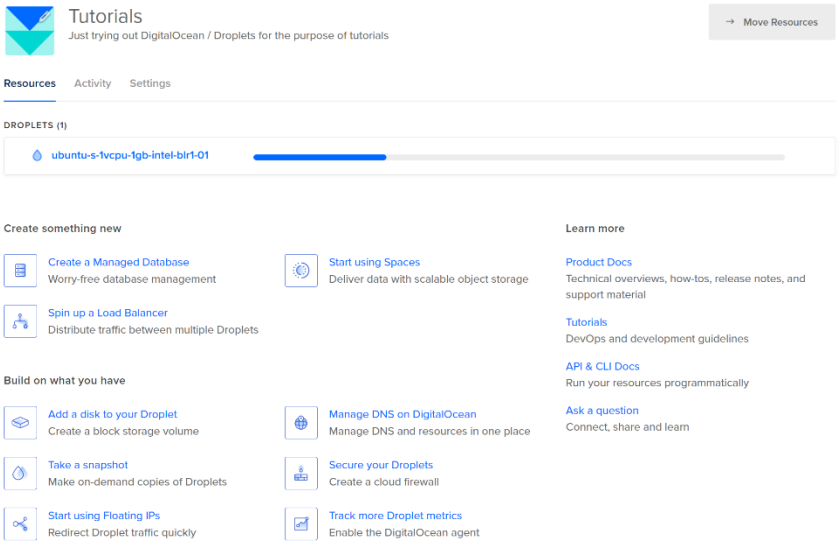
Choose a region. Ideally, this has to be the region closest to your users. This is the physical location where the server will be located:



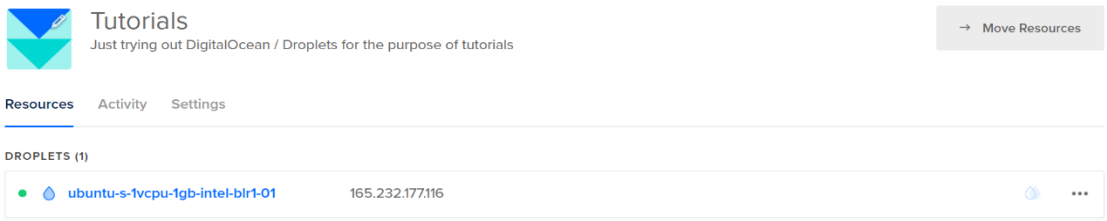
Finally, choose a password and click create droplet:



You will see processing like this



and finally, we will get a public IP address



We will use this public IP address to login into our server

Initial Login

The very first step to start using a server is to log in to it. A web server is just like a computer on the web. We cannot connect a physical keyboard and mouse to it. So how do we use it? We will use an SSH Client to get access to the server's terminal (this remote computer), and once we have access to the terminal, we can start executing commands on this server.

To log in to our server, we need to know its public IP address, just like you need to know the address of a place before visiting it.

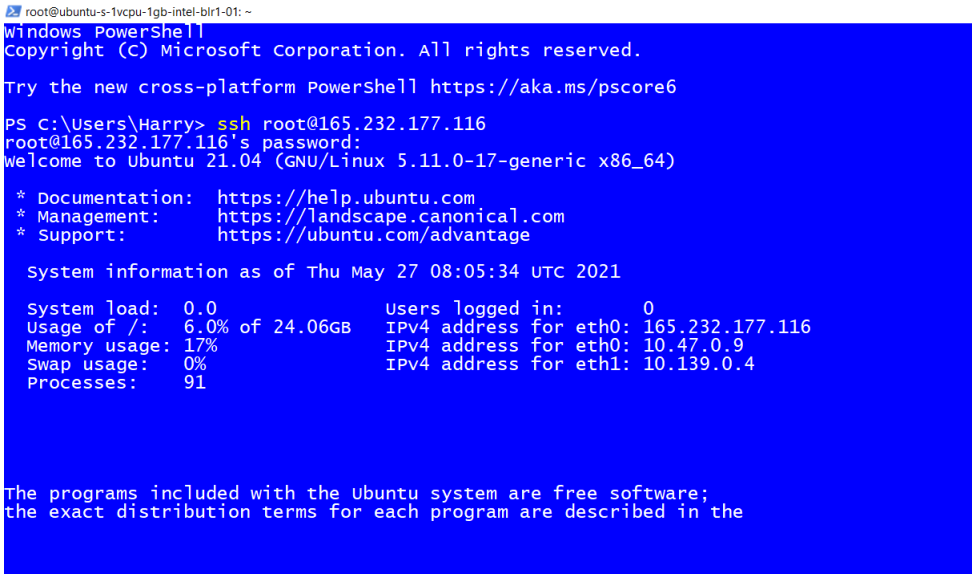
Execute the following command on the PowerShell or terminal

```
ssh root@your_server_ip
```

I will type this:

```
ssh root@165.232.177.116
```

Enter the password provided by your service provider. We have created a password in DigitalOcean, which we will enter here. Once you enter the password when prompted, you will be logged in as root.



The root user is a user having the highest privileges on the server. This means that a root user can do anything and everything on the server. This is why it is strongly discouraged to use a root account as it might accidentally delete files/uninstall programs/install programs etc. Hence we will create a user who will have restricted permissions on the server for day-to-day use.

Creating a non-root user

We will now create a non-root user account that will have restricted access to the machine. This is ideal if you want to have people working for you on the server without doing everything on the server.

Execute the following command:

```
adduser <Your-Username>
```

You will be asked to enter some information and a password. Choose a strong password to avoid getting hacked!

```
root@ubuntu-s-1vcpu-1gb-intel-blr1-01:~#
root@ubuntu-s-1vcpu-1gb-intel-blr1-01:~# adduser harry
Adding user `harry' ...
Adding new group `harry' (1000) ...
Adding new user `harry' (1000) with group `harry' ...
Creating home directory `/home/harry' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for harry
Enter the new value, or press ENTER for the default
        Full Name []: Harry
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
root@ubuntu-s-1vcpu-1gb-intel-blr1-01:~#
```

We have successfully created an account with basic access. We want this user to be able to elevate to root access when required. This will be useful when you want to use your server and sometimes do something which requires root access.

This will save you the time to log out and log back in as the root user.

Execute the following command on the server:

```
usermod -aG sudo <Your-Username>
```

Firewall setup

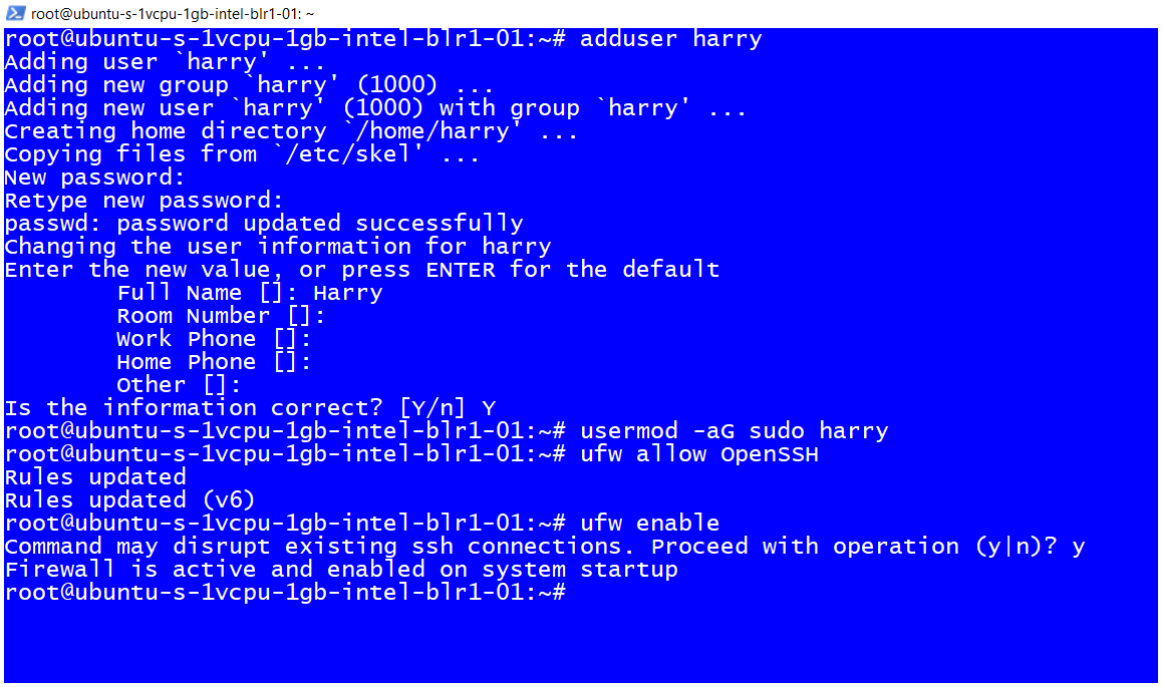
Our server is public, and we want to protect it from external unwanted connections. We want only selected services exposed to the public on our server.

We will set up the UFW firewall on our Ubuntu 20.04 server and allow OpenSSH, which allows us to connect to our server.
Execute the following command to allow OpenSSH:

```
ufw allow OpenSSH
```

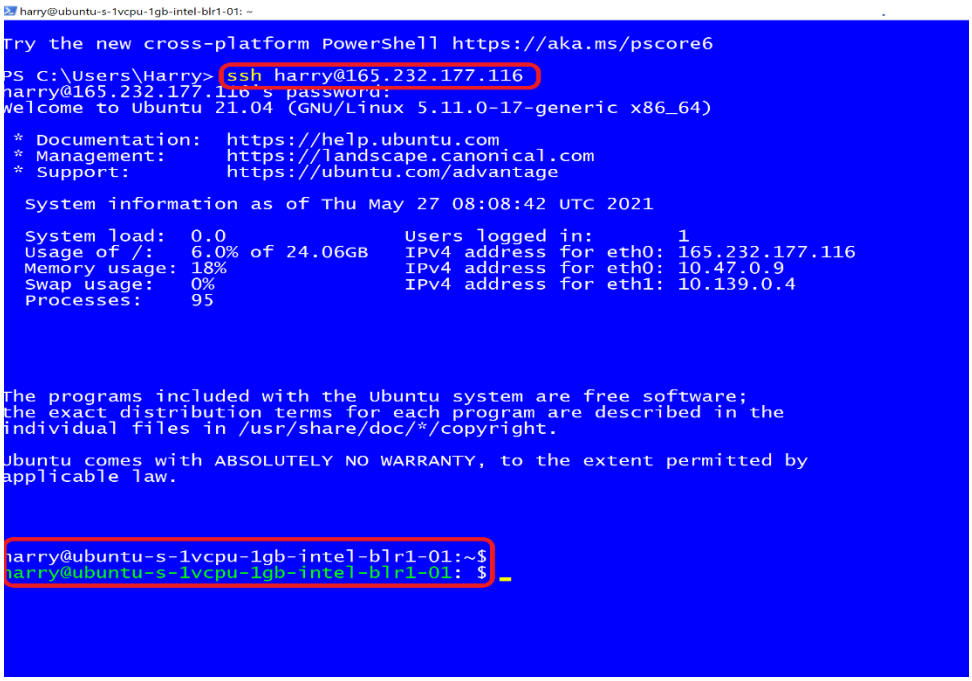
Enable the firewall by executing the following command:

```
ufw enable
```



The firewall will now block all the connections except SSH

You have successfully created a new user. Try to log in through this user in a new terminal tab and verify everything works before logging out from the current session.



You can type the following command to elevate to the root access as and when required.

```
sudo su
```

To go back to the existing user, execute the following command:

```
exit
```

```
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01: $ ls
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01: $ cd ~
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01: $ ls
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01: $ pwd
/home/harry
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01: $ sudo su
[sudo] password for harry:
root@ubuntu-s-1vcpu-1gb-intel-blr1-01:/home/harry# ls
root@ubuntu-s-1vcpu-1gb-intel-blr1-01:/home/harry# exit
exit
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01: $
```

2.Transferring Files, Passwordless login & Managing multiple servers..

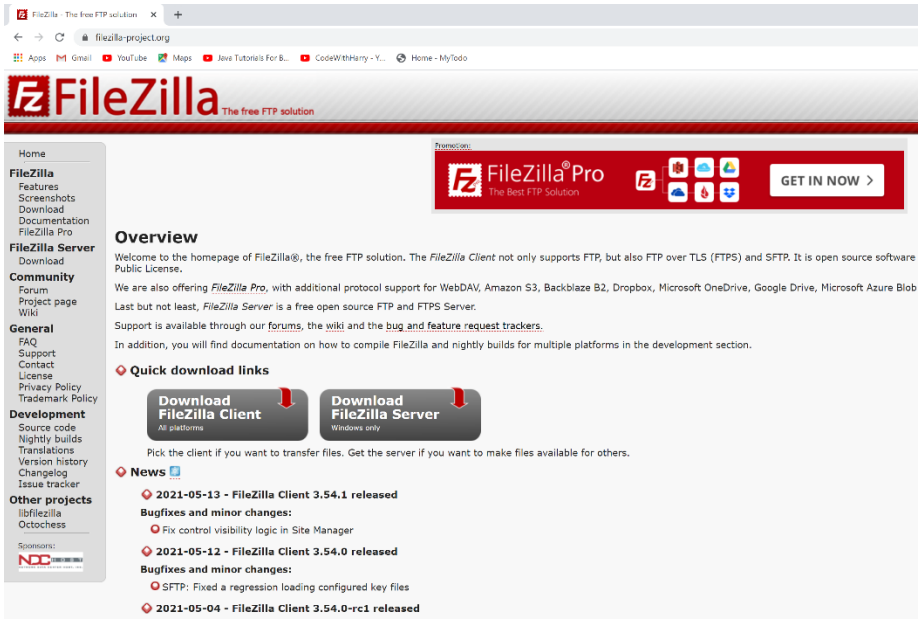
I will tell you few tricks I use to manage multiple servers efficiently. Let's start with uploading files to the server.

Uploading files to your server

In order to upload files to your server, you can use an SFTP client like Filezilla which simply can be used by entering your server IP, username, and password. Follow the steps below:

Step 1 - Download and install Filezilla

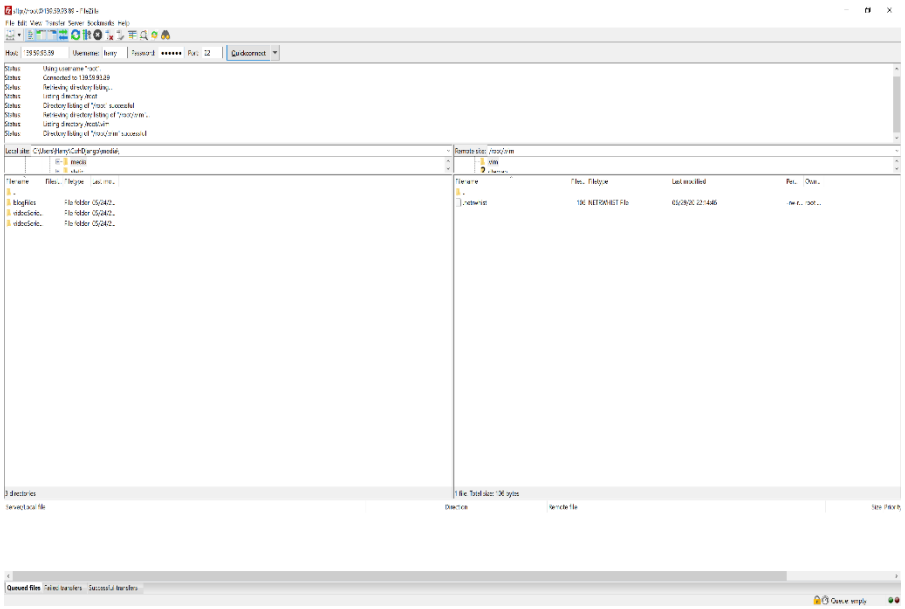
FileZilla Client is a fast and reliable cross-platform FTP, FTPS, and SFTP client with lots of useful features and an intuitive graphical user interface. Download this free software [from here]



Install Filezilla on your computer and open it.

Step 2 - Login to your server using FileZilla

Enter your server IP, username, and password inside the boxes at the top in FileZilla. Make sure your enter 22 as the port. Press enter and wait for some time.



Step 3 - Uploading files to your server

You can now upload files to your server by navigating to the directory of your choice and dragging/dropping your files into the folder(on your server) of your choice.

Enabling passwordless login to your server

Every time we need to log in to our server, we type 'ssh user@ip-address' command followed by the strong password which is quite cumbersome.

We can avoid typing this strong password without having to compromise security by authorizing our machine to log

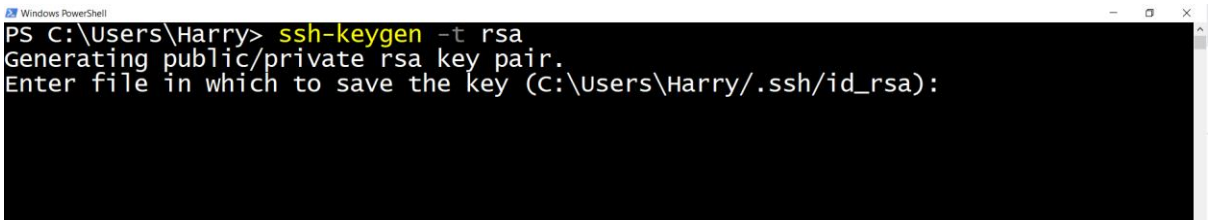
in to the server without the password every time we type a single (short) command. Follow the steps below:

Step 1 - Create a private-public key pair

We can create a private-public key pair using ssh-keygen. Execute the following command on your Cmd:

```
ssh-keygen -t rsa
```

This generates a private-public key pair on your computer. Press enter 3 times to choose the default options.



Step 2 - Deploy ssh key on your server

Login to your server using ssh(preferably non-root user) and password and execute the following command:

```
cd ~
mkdir .ssh
```

This will create a .ssh in the home directory ('/home/<YourSystemUser>/' in my case) of your user. DigitalOcean droplet ships with a root account so creating this '.ssh' directory is not needed if you want to configure serverless login for root.

Execute the following command on your host computer:

```
scp vikas@165.232.182.74:~/.ssh/authorized_keys
C:\Users\hp\.ssh\id_rsa.pub
```

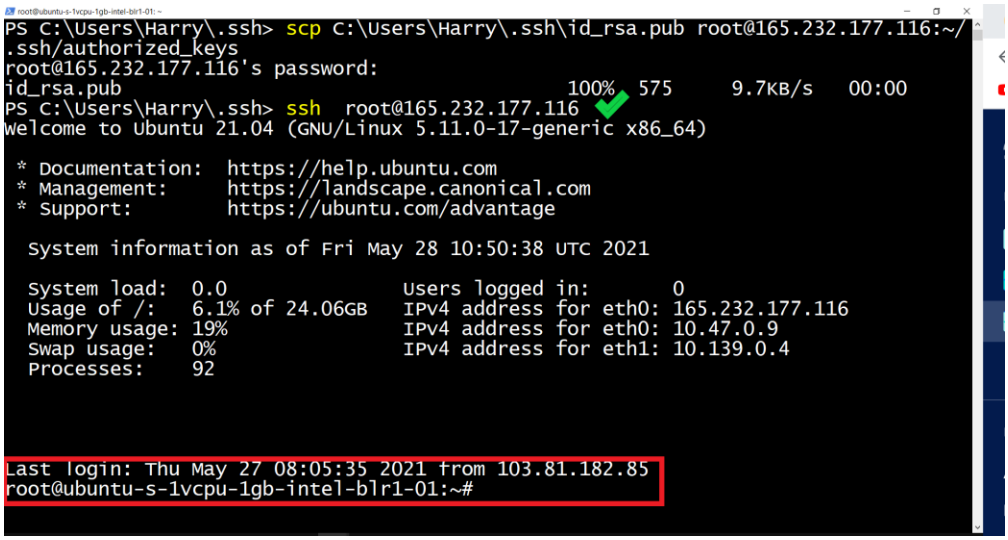
```
C:\Users\Harry\.ssh\id_rsa.pub
```

Replace 'C:\Users\Harry' with your user home directory on Windows. You will be prompted for the password. Enter your password and your ssh key will be deployed.

Step 3 - Test your passwordless login

Try to log into your server by entering the following command. Replace user with your username and ip-address with your server's ip address

```
ssh user@ip-address
```



Quick login using Powershell profiles

If you are using Windows Powershell, you can use PowerShell profiles to efficiently log in to multiple servers. Personally, I manage multiple servers including the one

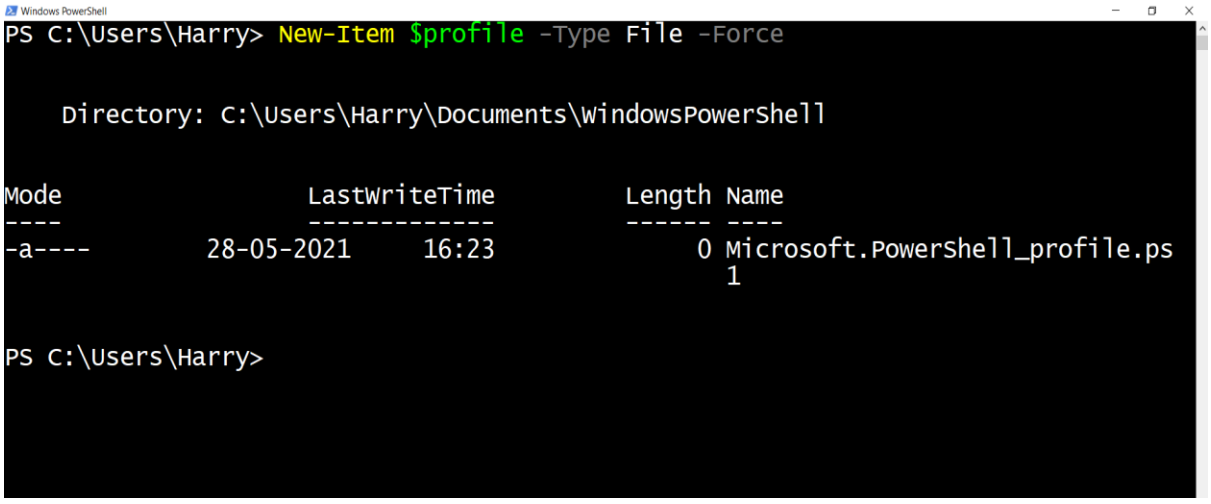
which hosts this site. I need to switch from a personal server to my client's server on a very regular basis. It gets hectic to remember multiple passwords and type them repeatedly. Hence I use PowerShell functions to manage multiple servers. If you are overwhelmed, don't! This is easier than you think and once you configure it you will never look back. Follow the below steps:

Step 1 - Create a PowerShell profile

Create a PowerShell profile using the command below:

```
New-Item $profile -Type File -Force
```

This creates a PowerShell profile that will execute whenever you start PowerShell on your computer.



Step 2 - Open and add a function to the PowerShell profile

Open your PowerShell profile by executing the following command on Windows:

```
notepad $profile
```

As a part of the next step, we will add few functions to our PowerShell profile. Paste the following code to your profile

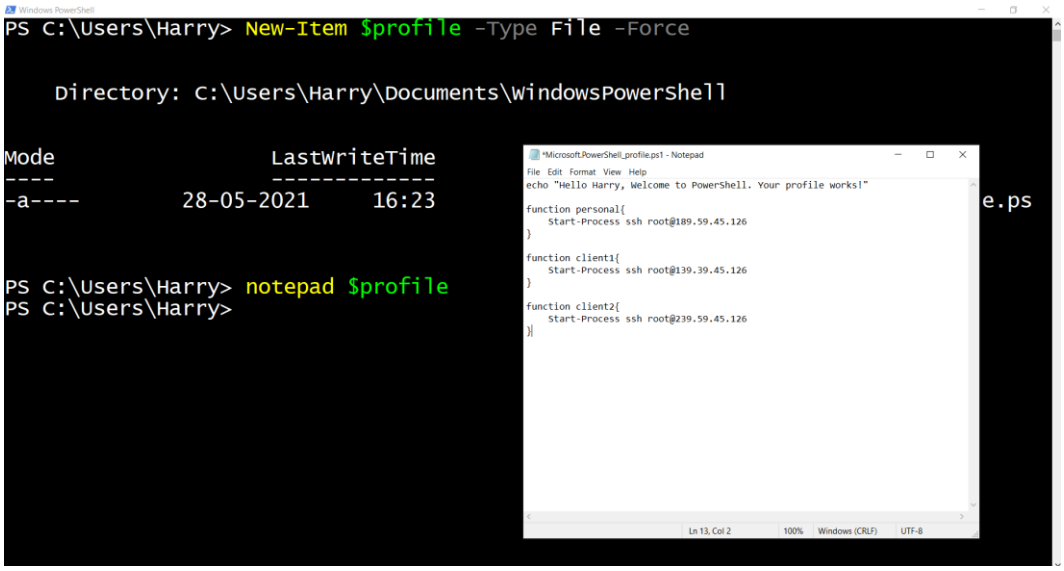
```
echo "Hello Sir, Welcome to PowerShell. Your profile works!"

function personal{
   Start-Process ssh harry@189.59.45.126
}

function client1{
   Start-Process ssh harry@139.39.45.126
}

function client2{
   Start-Process ssh root@239.59.45.126
}
```

Replace the usernames and IP addresses with the actual values of your servers. Finally, save and close the file.



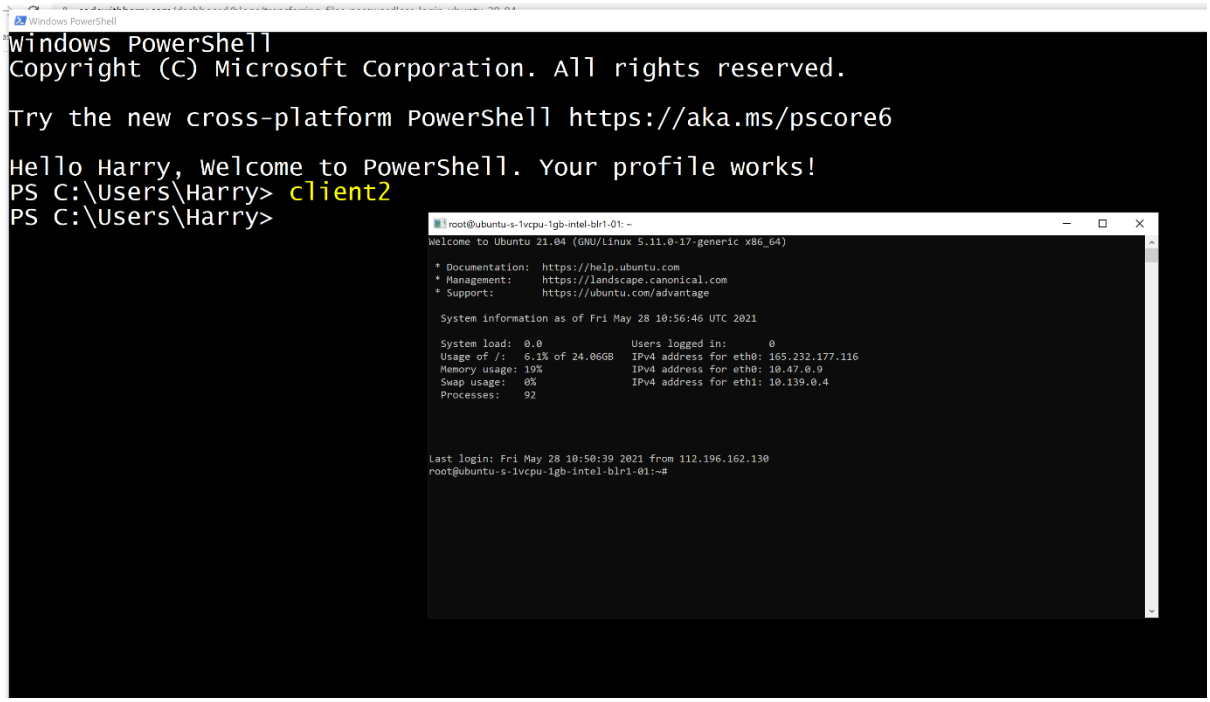
Step 3 - Test your profile

If you don't restart PowerShell you will see an error like this:

```
PS C:\Users\Harry> notepad $profile
PS C:\Users\Harry> client2
client2 : The term 'client2' is not recognized as the name of a cmdlet,
function, script file, or operable program. Check the spelling of the name, or
if a path was included, verify that the path is correct and try again.
At line:1 char:1
+ client2
+ ~~~~~~~
    + CategoryInfo          : ObjectNotFound: (client2:String) [], CommandNotFou
   ndException
```

```
    + FullyQualifiedErrorId : CommandNotFoundException
```

Restart your PowerShell and type client2:



Your profile should now work. Close all the PowerShell instances and reopen PowerShell. Now try to execute any one of the following commands:

```
personal
```

```
client1


client2
```

Wow we are now able to open the server we created for our client by issuing a single word command ie 'client1'

Note: You can always configure serverless login for a root user but using a root account for day-to-day activities is not recommended.

You can configure as many servers as you want of your choice.

3.Installing LAMP stack on Ubuntu 20.04 in 5 Minutes.

Install LAMP stack on Ubuntu 20.04. LAMP stack consists of the following components:

1. Linux - Any Linux based operating system
2. Apache server - Apache is an open-source webserver
3. MySQL - MySQL database
4. PHP - PHP as a server-side programming language

These components work together to host single or multiple dynamic websites that are stable in production.

I will be using DigitalOcean for creating the VPS but you can use any service provider of your choice. As long as the operating system of your server is Ubuntu, this guide will work.

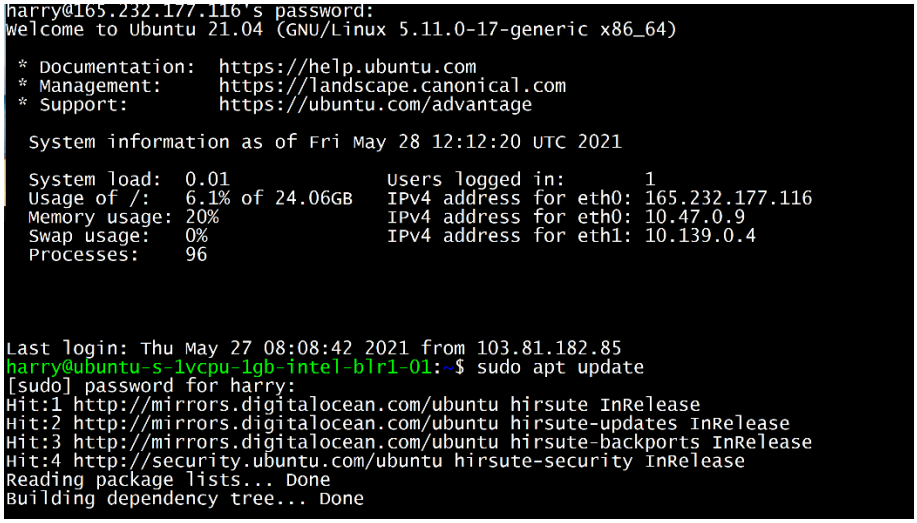
Before you go ahead and start installing the LAMP stack on Ubuntu, make sure you have a non-root user configured along with a server firewall for security purposes. If you don't know what I mean, have a look at this article to configure a non-root sudo user and a firewall on your server before you proceed further.

Installing Apache and allowing it through the firewall

Apache is an amazing open-source web server. Let's install it using the following commands:

```
sudo apt update
```

This command updates the package lists for upgrades. Let's run it to ensure that we install the up-to-date versions of the software later.




```
sudo apt install apache2
```

Press Y when prompted and the installation will finish after a while.

```
Last login: Thu May 27 08:08:42 2021 from 103.81.182.85
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$ sudo apt update
[sudo] password for harry:
Hit:1 http://mirrors.digitalocean.com/ubuntu hirsute InRelease
Hit:2 http://mirrors.digitalocean.com/ubuntu hirsute-updates InRelease
Hit:3 http://mirrors.digitalocean.com/ubuntu hirsute-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu hirsute-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$ sudo apt install apache2
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils bzip2 libapr1 libaprutil1 libaprutil1-dbd-sqlite3
  libaprutil1-ldap libjansson4 liblua5.3-0 mailcap mime-support ssl-cert
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom www-browser bzip2-doc
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils bzip2 libapr1 libaprutil1 libaprutil1-dbd-sqlite3
  libaprutil1-ldap libjansson4 liblua5.3-0 mailcap mime-support ssl-cert
0 upgraded, 14 newly installed, 0 to remove and 0 not upgraded.
Need to get 1968 kB of archives.
After this operation, 8463 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

Let's allow Apache through the firewall using the following command:

```
sudo ufw allow in "Apache Full"
```

You can always issue the "sudo ufw status" command to verify the changes:

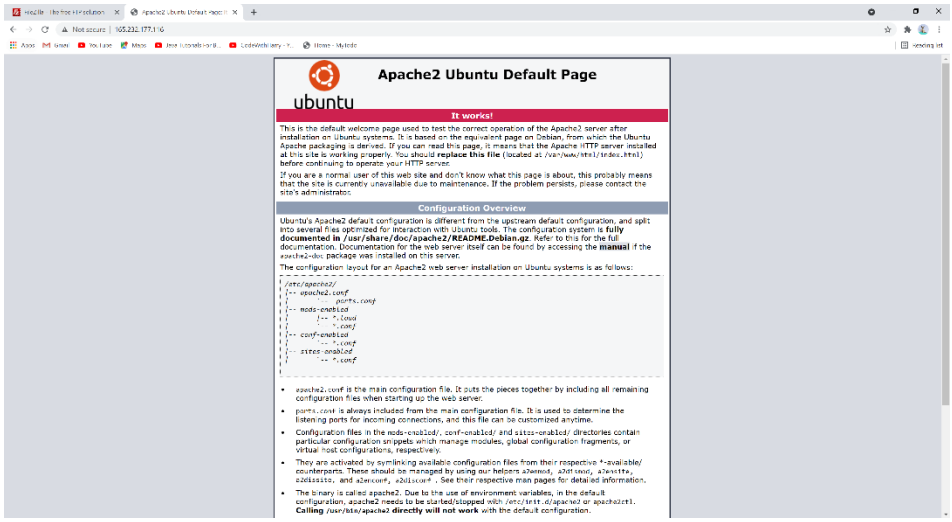
```
sudo ufw status
```

```
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$ sudo ufw allow in "Apache Full"
Rule added
Rule added (v6)
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
OpenSSH                    ALLOW       Anywhere
Apache Full                ALLOW       Anywhere
OpenSSH (v6)               ALLOW       Anywhere (v6)
Apache Full (v6)           ALLOW       Anywhere (v6)

harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$
```

You can now go to the server URL (IP address) and the apache2 default page will be displayed on the screen



Congratulations! You have successfully installed the apache web server on your server

Installing MySQL on Ubuntu

MySQL is a very popular and open source database that can be used with almost any application to store huge amounts of data effectively.

Execute the following command to install MySQL on ubuntu:

```
sudo apt install mysql-server
```

confirm the installation by typing 'y' followed by the 'Enter'.

```
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$ sudo apt install mysql-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libcgi-fast-perl libcgi-pm-perl libclone-perl libencode-locale-perl libevent-core-2.1-7
  libevent-pthreads-2.1-7 libfcgi-bin libfcgi-perl libfcgi0ldbl libhtml-parser-perl
  libhtml-tagset-perl libhtml-template-perl libhttp-date-perl libhttp-message-perl libio-html-perl
  liblwp-mediatypes-perl libmecab2 libprotobuf-lite23 libtimedate-perl liburi-perl mecab-ipadic
  mecab-ipadic-utf8 mecab-utils mysql-client-8.0 mysql-client-core-8.0 mysql-common mysql-server-8.0
  mysql-server-core-8.0
Suggested packages:
  libdata-dump-perl libipc-sharedcache-perl libwww-perl mailx tinyca
The following NEW packages will be installed:
  libcgi-fast-perl libcgi-pm-perl libclone-perl libencode-locale-perl libevent-core-2.1-7
  libevent-pthreads-2.1-7 libfcgi-bin libfcgi-perl libfcgi0ldbl libhtml-parser-perl
  libhtml-tagset-perl libhtml-template-perl libhttp-date-perl libhttp-message-perl libio-html-perl
  liblwp-mediatypes-perl libmecab2 libprotobuf-lite23 libtimedate-perl liburi-perl mecab-ipadic
  mecab-ipadic-utf8 mecab-utils mysql-client-8.0 mysql-client-core-8.0 mysql-common mysql-server
```

You should be able to login to MySQL console by typing the following command:

```
sudo mysql
```

```
User sessions running outdated binaries:
 harry @ user manager service: systemd[6385]
 root @ user manager service: systemd[5636]
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$ sudo apt install mysql-server^C
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$ ^C
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$ ^C
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.25-0ubuntu0.21.04.1 (Ubuntu)

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

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>
```

To exit the MySQL console type exit in the MySQL console

```
exit
```

Installing PHP

We can install PHP by firing the following commands:

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

```
mysql> exit
Bye
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$ sudo apt install php libapache2-mod-php php-mysql
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libapache2-mod-php7.4 libsodium23 php-common php7.4 php7.4-cli php7.4-common php7.4-json
  php7.4-mysql php7.4-opcache php7.4-readline
Suggested packages:
  php-pear
The following NEW packages will be installed:
  libapache2-mod-php libapache2-mod-php7.4 libsodium23 php php-common php-mysql php7.4 php7.4-cli
  php7.4-common php7.4-json php7.4-mysql php7.4-opcache php7.4-readline
0 upgraded, 13 newly installed, 0 to remove and 0 not upgraded.
```

This will install the following 3 packages

- php - installs PHP

- libapache2-mod-php - Used by apache to handle PHP files
- php-mysql - PHP module that allows PHP to connect to MySQL

Confirm the PHP installation by executing the below command:

```
php -v
```

```
Last login: Fri May 28 12:12:20 2021 from 112.196.162.130
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$ php -v
PHP 7.4.16 (cli) (built: Mar 23 2021 16:15:03) ( NTS )
Copyright (c) The PHP Group
Zend Engine v3.4.0, Copyright (c) Zend Technologies
    with Zend OPcache v7.4.16, Copyright (c), by Zend Technologies
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$
```

Congratulations! You have successfully installed LAMP stack on your server….

Write This Command In Your Server Machine----

Cd /var/www/html/

Then Type ls In Your Terminal To Show All Files In Your Terminal…

Sudo vim index.html For Edit Your Html File…

Now We Have To Restart Our Server Using

sudo service apache2 restart

4.How to Host Multiple Websites on Ubuntu VPS?

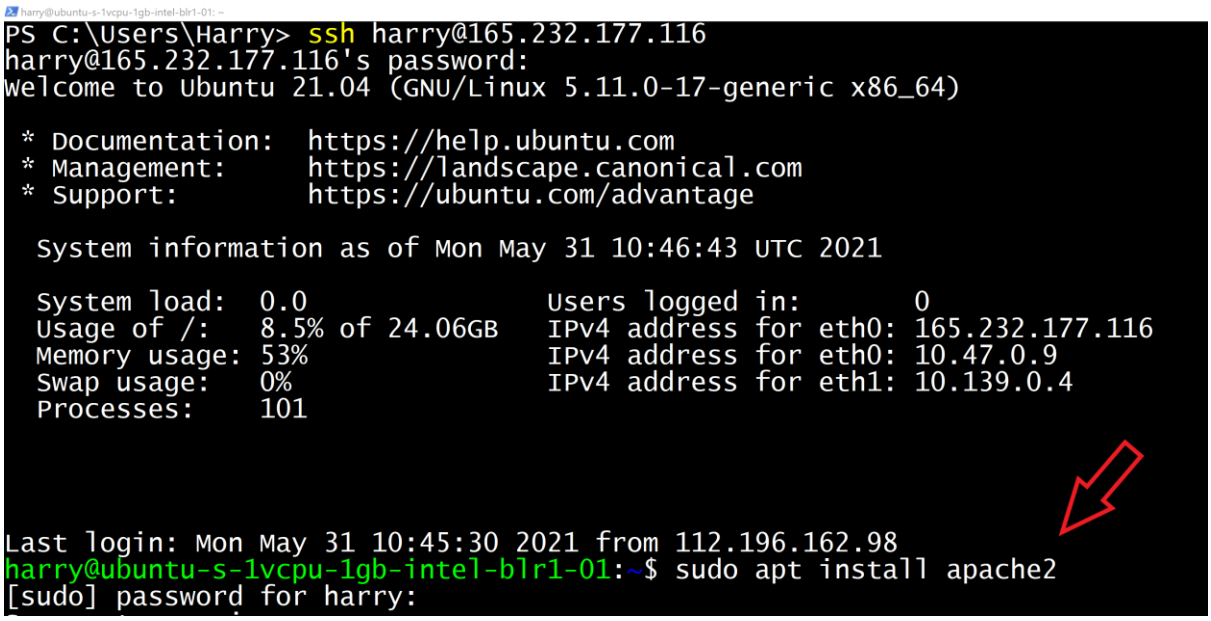
An apache2 web server provides robustness and scalability for hosting multiple websites on your Ubuntu VPS. This means you can utilize the power of your VPS hardware without any downtime on your website.

We can configure apache2 to host multiple websites on the same VPS. To achieve this using apache, we create virtual hosts corresponding to each domain or individual site. A virtual host is a way to direct traffic corresponding to an IP to a specific directory on our VPS. This way we will be able to host multiple websites on a single Ubuntu server.

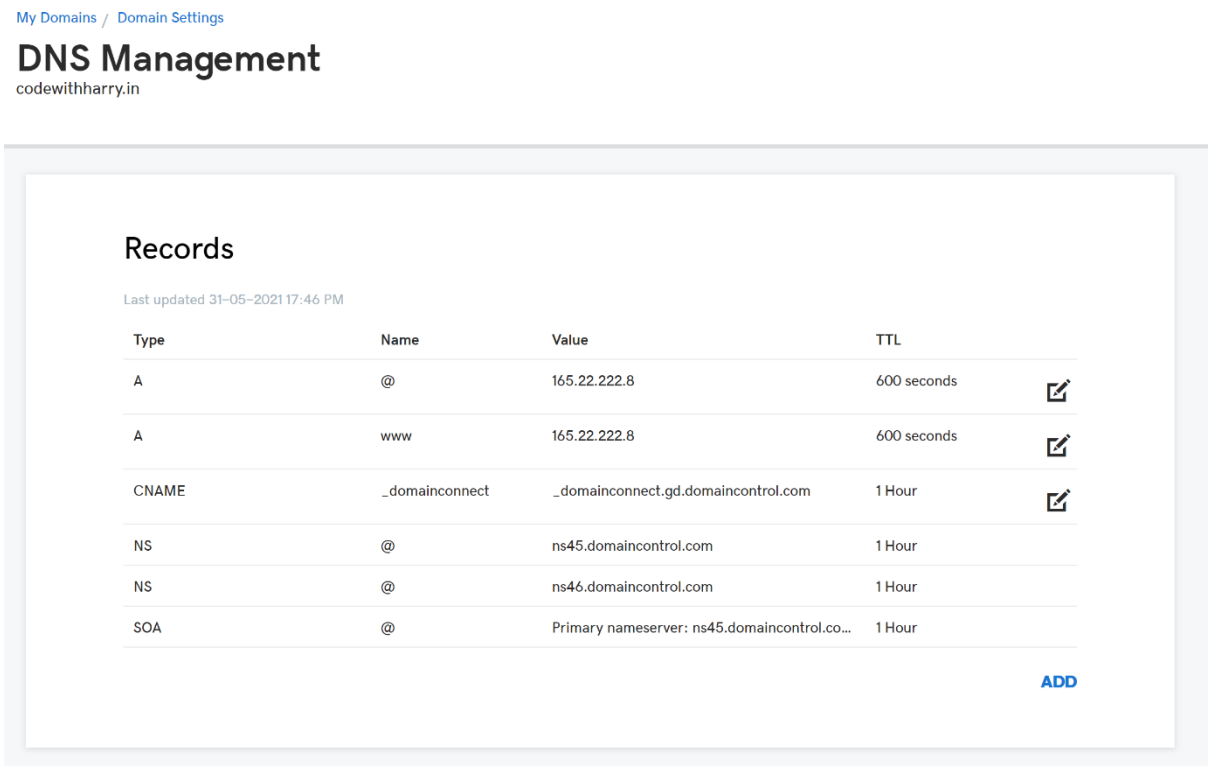
How to setup a virtual host in apache2

If you haven't already installed apache2, you can do so by entering the below command in the ubuntu terminal

```
sudo apt install apache2
```



Assuming you have apache2 installed on your machine, all you need to do now is to follow the steps outlined in the next section to host multiple websites. For the sake of this tutorial, I will assume that we have to host 2 websites codewithharry.com and programmingwithharry.com on our ubuntu VPS. We will point the domains to the IP address of our droplets. Here is how A record is set up in GoDaddy.



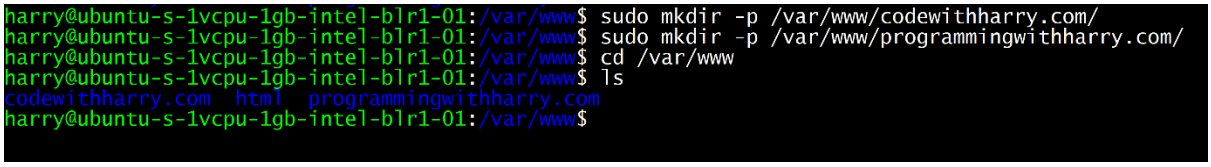
The process is pretty similar for other domain providers

Let's set up our virtual hosts now

Step 1 - Creating Directories for individual sites

Let's create individual directories to store the contents of codewithharry.com and programmingwithharry.com. Execute the commands below to create these directories inside the /var/www folder

```
sudo mkdir -p /var/www/codewithharry.com/
sudo mkdir -p /var/www/programmingwithharry.com/
```



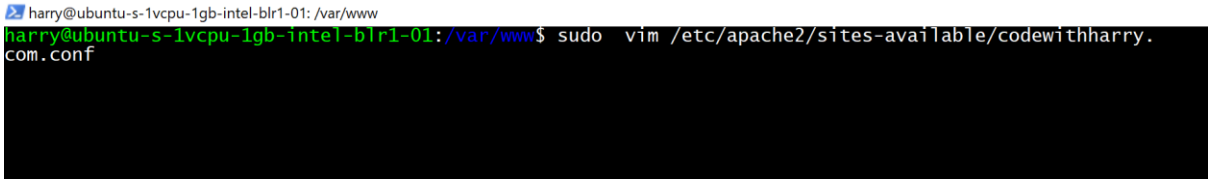
Step 2 - Transfer the site contents

Transfer the index.html and related files to the individual site directories using Filezilla

Step 3 - Creating the VirtualHost files

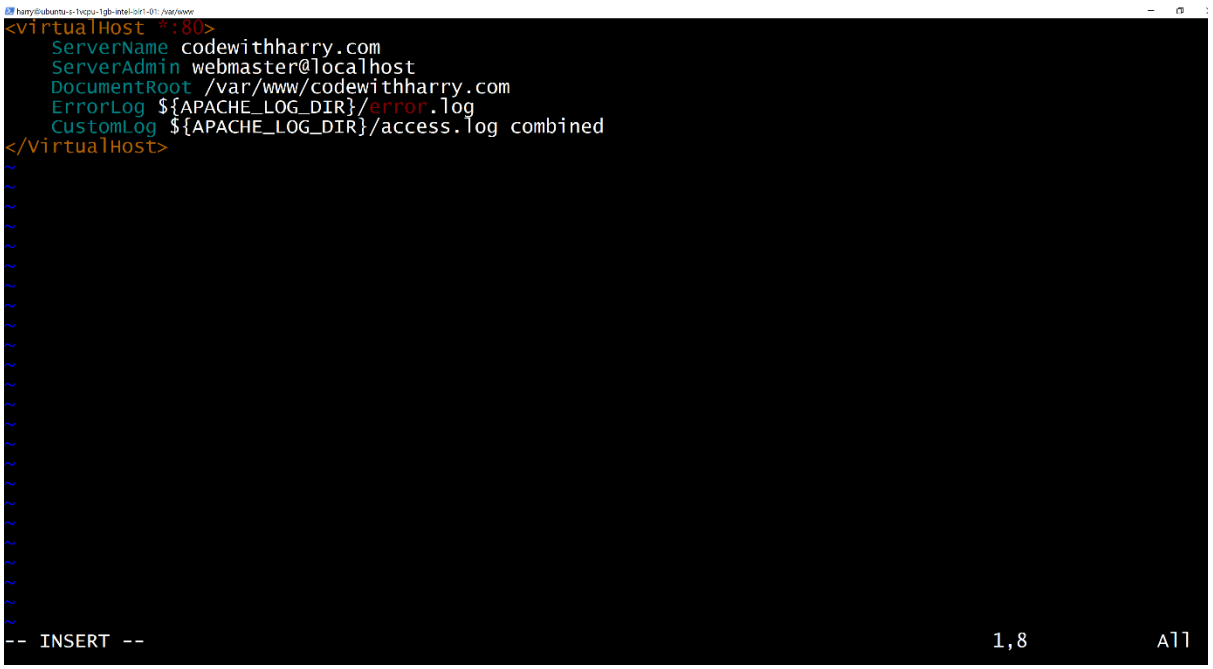
Create a new file inside the /etc/apache2/sites-available/ directory by firing the following commands below:

```
sudo vim /etc/apache2/sites-available/codewithharry.com.conf
```



Once the codewithharry.com.conf file is created. paste the below contents inside it

```
<VirtualHost *:80>
    ServerName codewithharry.com
    ServerAdmin yourPublicEmail@email.com
    DocumentRoot /var/www/codewithharry.com
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```



Repeat the same for programmingwithharry.com by executing the command below:

```
vim /etc/apache2/sites-available/programmingwithharry.com.conf
```

and paste the below contents:

```
<VirtualHost *:80>
```

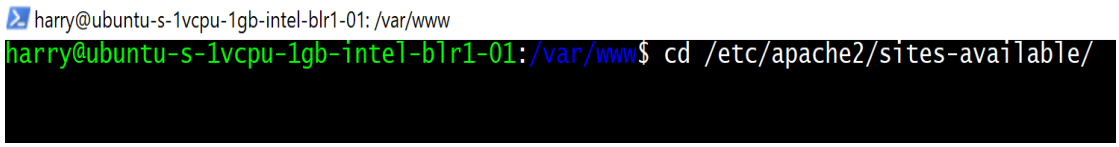
```
    ServerName programmingwithharry.com
    ServerAdmin yourPublicEmail@email.com
    DocumentRoot /var/www/programmingwithharry.com
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Step 4 - Enable the VirtualHosts

In order for these virtual host files to function correctly, we need to enable them.

Enter the directory where we have created virtual hosts:

```
cd /etc/apache2/sites-available/
```

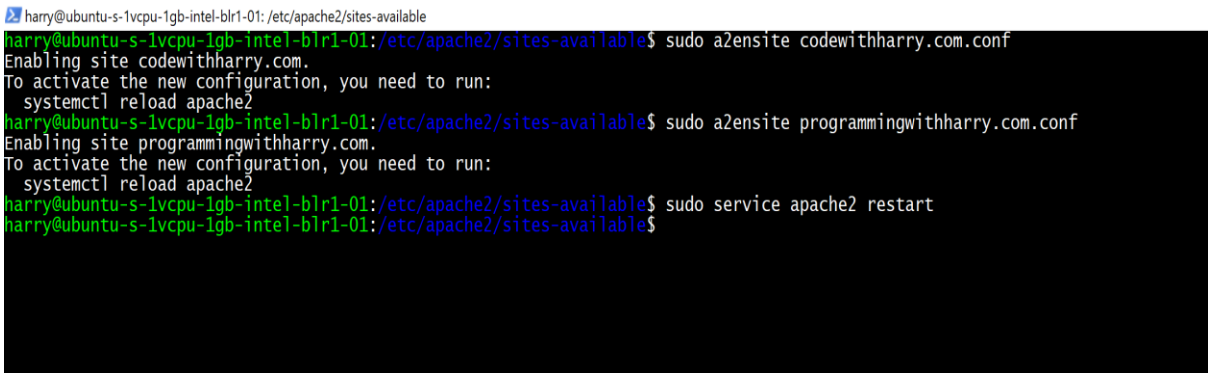


Execute the following commands to enable the virtual hosts:

```
sudo a2ensite codewithharry.com.conf
sudo a2ensite programmingwithharry.com.conf
```

Finally, you will have to restart the apache server:

```
sudo service apache2 restart
```



Step 5 - Test the configuration

Test the configuration of these virtual hosts by visiting your domains. You can configure as many virtual hosts as you want for your domains and this technique can help you power many websites on a single VPS.

5. Installing phpMyAdmin and adding password authentication to MySQL on Ubuntu

Loading...

In this video, we will see how to install and secure phpMyAdmin on Ubuntu 20.04. phpMyAdmin was created to easily interact with MySQL databases using a web-based interface. I will assume that you have configured a non-root user on your ubuntu machine and installed the lamp stack.

We will also see how to enable password authentication on MySQL which supports socket-based authentication by default.

Step 1 - Installing PHPMyAdmin

Let's update our server's package index

```
sudo apt update
```

```
Last login: Mon May 31 10:46:44 2021 from 112.196.162.98
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$ sudo apt update
[sudo] password for harry:
Hit:1 http://mirrors.digitalocean.com/ubuntu hirsute InRelease
Get:2 http://mirrors.digitalocean.com/ubuntu hirsute-updates InRelease [109 kB]
Hit:3 http://security.ubuntu.com/ubuntu hirsute-security InRelease
Hit:4 http://mirrors.digitalocean.com/ubuntu hirsute-backports InRelease
Get:5 http://mirrors.digitalocean.com/ubuntu hirsute-updates/main amd64 Packages [170 kB]
Fetched 278 kB in 1s (277 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$
```

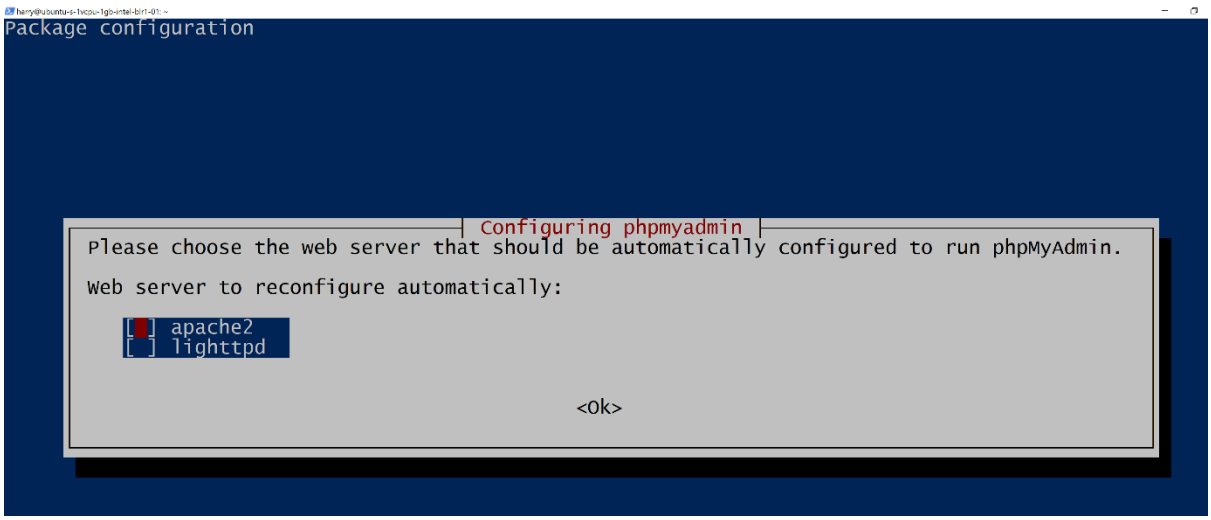
Run the following command to install phpMyAdmin and related packages

```
sudo apt install phpmyadmin php-mbstring php-zip php-gd php-json php-curl
```

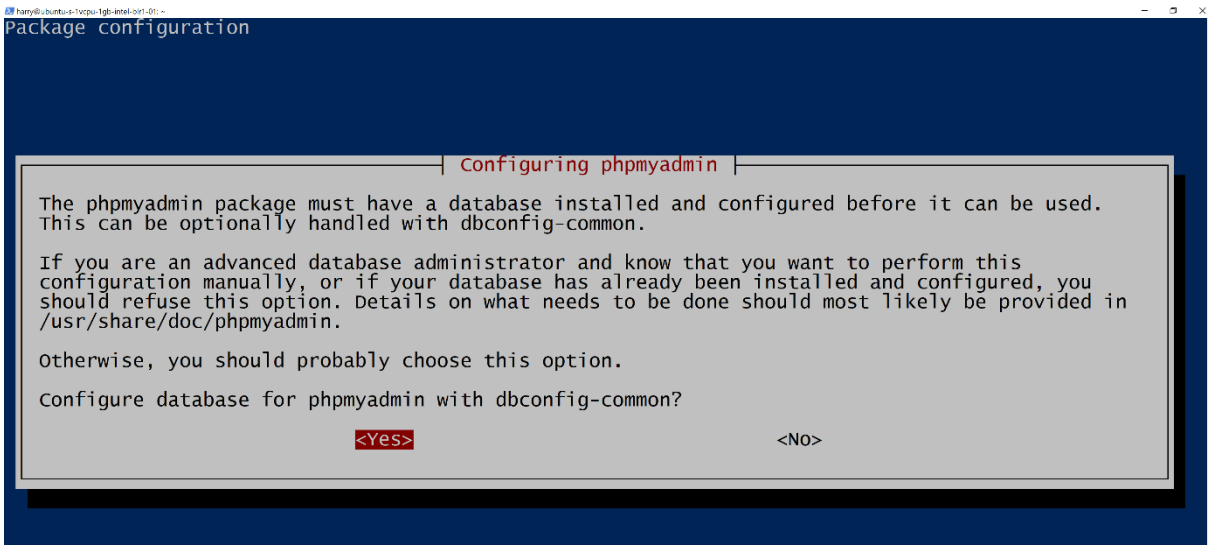
Using this command we installed the following packages:

- php-mbstring: A module for managing non-ASCII strings with different encodings
- php-zip: An extension that facilitates uploading .zip files to phpMyAdmin
- php-gd: Enables support for GD graphics library
- php-json: Provides support for JSON serialization
- php-curl: Allows PHP to communicate with other servers

While installing PHPMyAdmin you will see a prompt that will ask you to select the webserver. Press <Spacebar> <Tab> and then enter key to continue the installation.



You will also be asked to continue with dbconfig-common setup. Press enter to confirm.

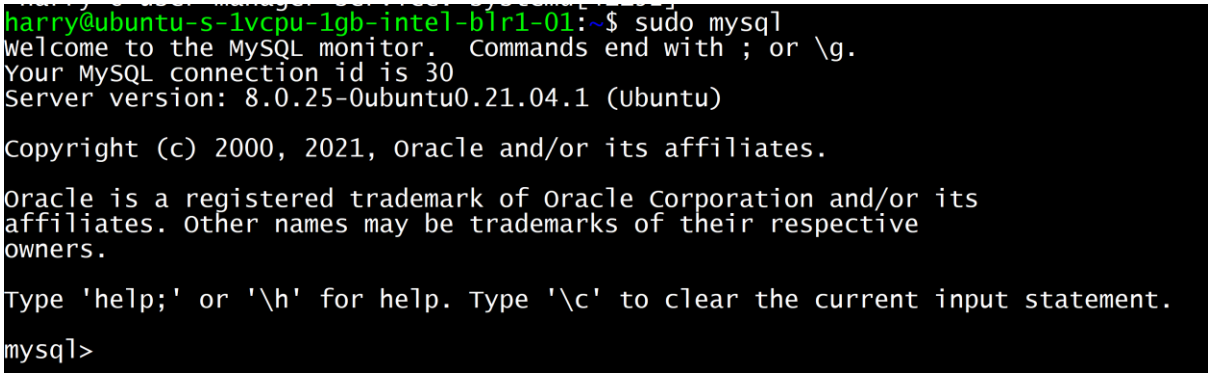


Finally, choose and confirm a strong password for PHPMyAdmin. If you get a prompt asking for the services which need to be restarted, simply press the "tab" and then "enter" key to continue. Phpmyadmin is now installed on your Ubuntu server

Step 2 - Configuring password access in MySQL

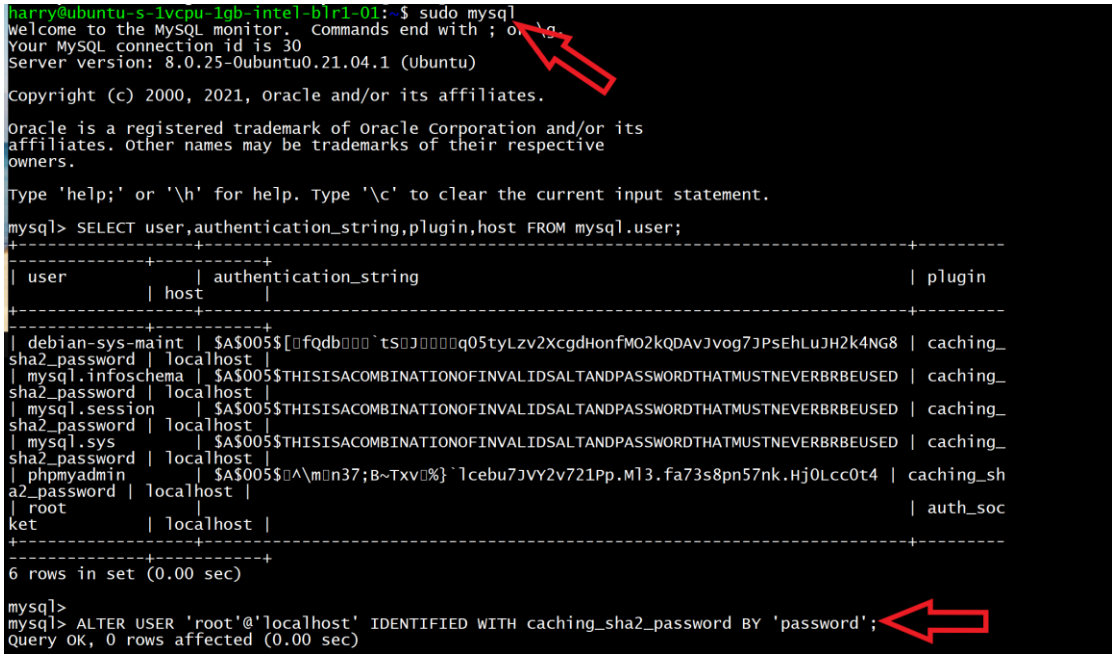
to use MySQL we need to enable password login in MySQL. enter the following command to launch the MySQL console:

```
sudo mysql
```



Now enter the following query to enable password authentication in MySQL

```
ALTER USER 'root'@'localhost' IDENTIFIED WITH caching_sha2_password BY 'MyStrongPassword1234$';
```



Please make sure you choose your own password and not the one which I chose ('MyStrongPassword1234$')

You can always check the authentication method used by using the following MySQL query:

```
SELECT user,authentication_string,plugin,host FROM mysql.user;
```

Exit from the MySQL console by typing exit.

Now you can log in using your MySQL password by entering the following command to the MySQL console

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

Step 3 - Configuring a non-root user in MySQL

Since its not a good idea to use root as your login user, we will create another MySQL user named harry to login to the PHPMyAdmin console:

Execute the following query:

```
CREATE USER 'harry'@'localhost' IDENTIFIED WITH caching_sha2_password BY 'MyStrongPassword1234$';
```

This creates a user named harry with the password - 'MyStrongPassword1234$'

Let's provide this user all the privileges so that we can use it to access to the PHPMyAdmin console. Execute the query below

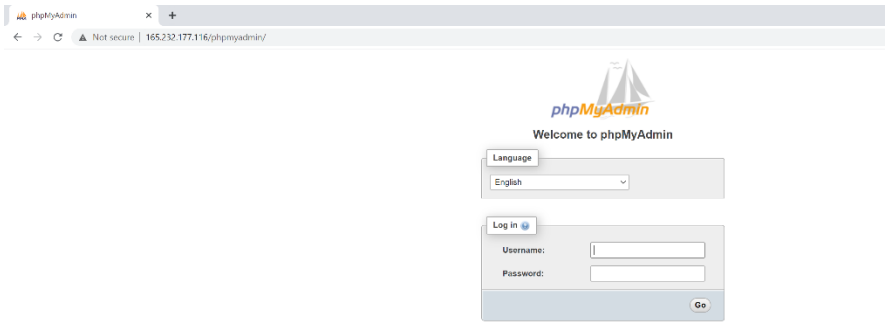
```
GRANT ALL PRIVILEGES ON *.* TO 'harry'@'localhost' WITH GRANT OPTION;
```

You can now exit the MySQL shell:

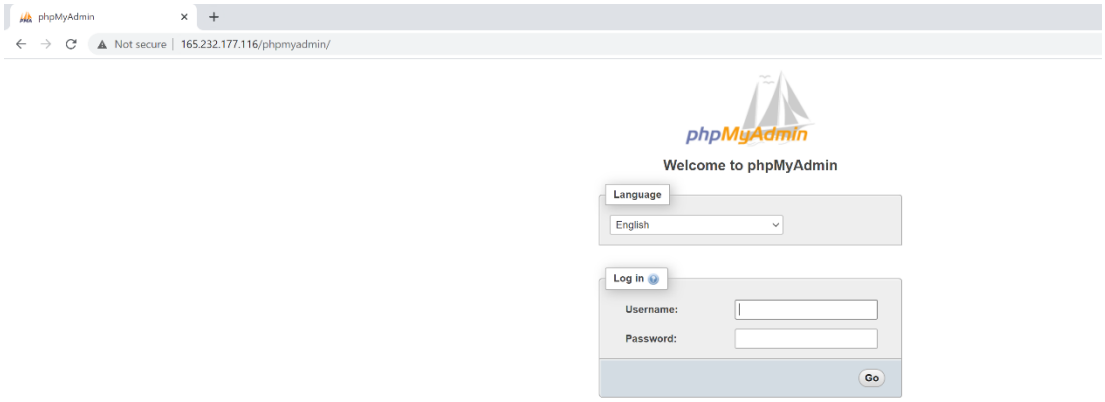
```
exit
```

Step 4 - Accessing MySQL

Go to the browser and type 'http://your_domain_or_IP/phpmyadmin' in your URL bar. You will see an option to log into your PHPMyAdmin console. Enter the username (harry in my case) and password you chose for this user.



You will be logged in to PHPMyAdmin.



6. How to get free https on your site using Let's Encrypt SSL on Ubuntu 20.04

Loading...

Have you ever seen that green padlock at the top in the URL bar which reads "Connection is secure"? In this post, we will discuss how to enable HTTPS for your website hosted on ubuntu 20.04. We will use let's encrypt for this purpose which is a non-profit certificate authority used to get SSL certificates for your website.

Not only that we can use certbot to automatically renew these certificates which makes it a one-time task for a single domain.

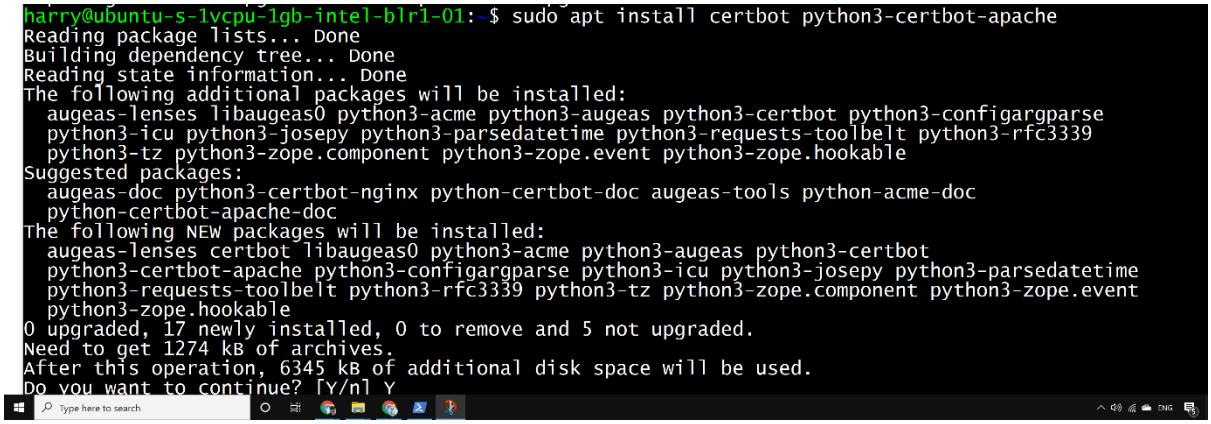
We will assume that you already have a domain pointed to the IP address of your VPS. Follow the steps below:

Step 1 - Install lets encrypt

Execute the following commands to install let's encrypt on ubuntu:

```
sudo apt install certbot python3-certbot-apache
```

Copy



Step 2 - Allow HTTPS through the firewall and configure apache2 virtual hosts

If you are using apache, enter the following commands to allow apache2 over the firewall:

```
sudo ufw allow 'Apache Full'
```

Copy

Step 3 - Obtain an SSL certificate

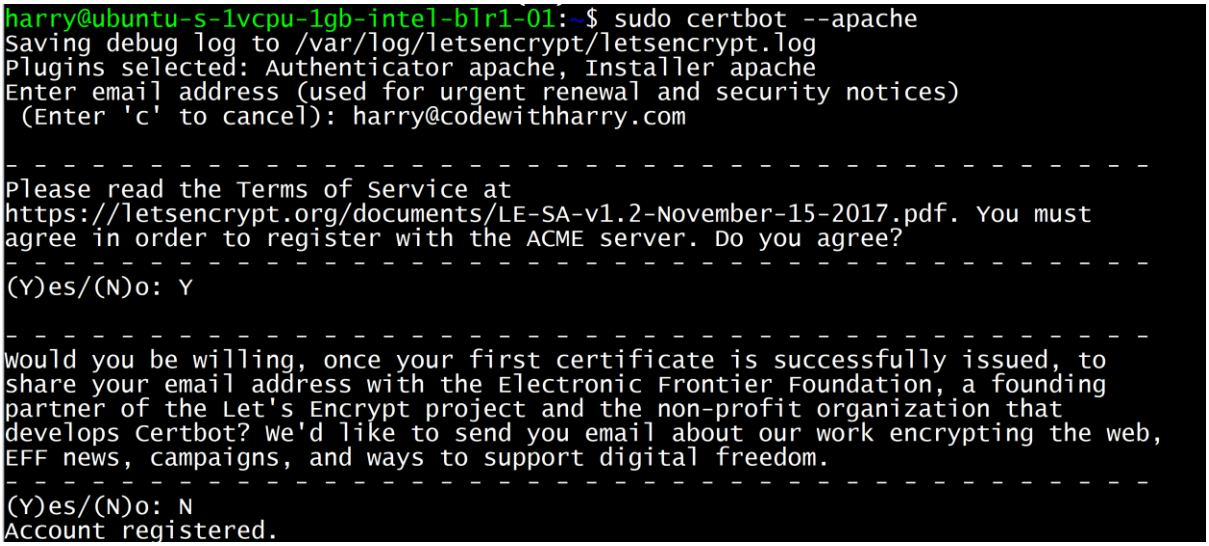
Run the following command to obtain an SSL certificate

```
sudo certbot --apache
```

Copy

```
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$ sudo certbot --apache
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Plugins selected: Authenticator apache, Installer apache
Enter email address (used for urgent renewal and security notices)
 (Enter 'c' to cancel): harry@codewithharry.com
```

This command shows you some prompts and finally installs an SSL certificate on the domain of your choice.



Step 4 - Test the auto-renewal process

Certbot, by itself, takes care of the auto-renewal process of your SSL certificate. You can test the auto-renewal of your SSL certificate by firing the following commands:

```
sudo systemctl status certbot.timer
```

You can even dry run the auto-renewal by using the below command:

```
sudo certbot renew --dry-run
```

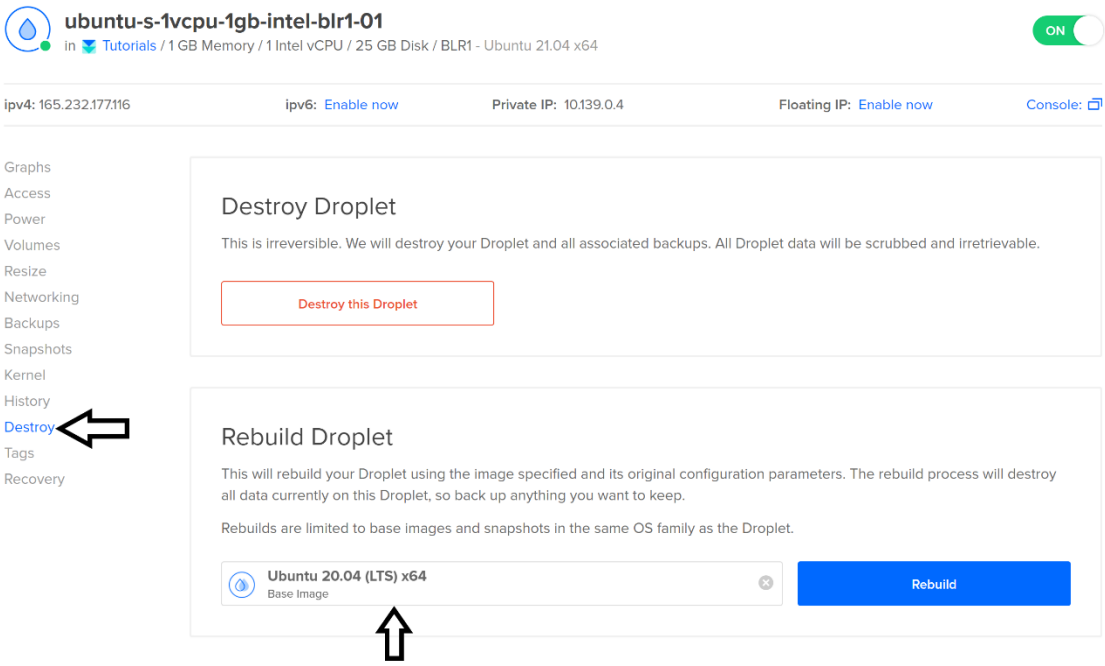
7. How to install LEMP stack (Linux, Nginx, MySQL, PHP) on Ubuntu 20.04

Loading...

In this tutorial, we will install the LEMP stack on the ubuntu 20 based server. LEMP consists of Linux, Nginx(pronounced as Engine-x), MySQL for database, and PHP as a backend programming language. Together you can use this combination to host a set of high-performing sites on a single server. I will assume that you have finished the initial server setup as

described here on your ubuntu server before installing the LEMP stack on Ubuntu 20.04.

If you are already using some other web server like apache2, it is recommended that you uninstall it or rebuild your server from scratch using the console provided by your hosting provider. DigitalOcean gives a convenient way to achieve this in its dashboard.



Let's install the LEMP stack now. Follow the steps below:

Step 1 - Update the server's package index

Update the server's package index by executing the command below:

```
sudo apt update
```

```
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$ sudo apt update
[sudo] password for harry:
Sorry, try again.
[sudo] password for harry:
Hit:1 http://mirrors.digitalocean.com/ubuntu hirsute InRelease
Get:2 http://mirrors.digitalocean.com/ubuntu hirsute-updates InRelease [109 kB]
Get:3 http://security.ubuntu.com/ubuntu hirsute-security InRelease [101 kB]
Hit:4 http://mirrors.digitalocean.com/ubuntu hirsute-backports InRelease
Get:5 http://mirrors.digitalocean.com/ubuntu hirsute-updates/main amd64 Packages [188 kB]
Get:6 http://mirrors.digitalocean.com/ubuntu hirsute-updates/universe amd64 Packages [216 kB]
Get:7 http://security.ubuntu.com/ubuntu hirsute-security/main amd64 Packages [117 kB]
Get:8 http://security.ubuntu.com/ubuntu hirsute-security/main Translation-en [32.9 kB]
Get:9 http://security.ubuntu.com/ubuntu hirsute-security/main amd64 c-n-f Metadata [2036 B]
Get:10 http://security.ubuntu.com/ubuntu hirsute-security/restricted amd64 Packages [100 kB]
Get:11 http://security.ubuntu.com/ubuntu hirsute-security/restricted Translation-en [14.4 kB]
Get:12 http://security.ubuntu.com/ubuntu hirsute-security/universe amd64 Packages [188 kB]
Get:13 http://security.ubuntu.com/ubuntu hirsute-security/universe Translation-en [31.0 kB]
Get:14 http://security.ubuntu.com/ubuntu hirsute-security/universe amd64 c-n-f Metadata [3928 B]
Fetched 1103 kB in 2s (480 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
4 packages can be upgraded. Run 'apt list --upgradable' to see them.
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$
```

Step 2 - Install Nginx

Install Nginx using the command below:

```
sudo apt install nginx
```

Step 3 - Allow Nginx through the firewall

Allow Nginx through the firewall using the command below:

```
sudo ufw app list
sudo ufw allow 'Nginx Full'
```

```
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$ sudo ufw app list
Available applications:
  Nginx Full
  Nginx HTTP
  Nginx HTTPS
  OpenSSH
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$ sudo ufw allow 'Nginx Full'
Rule added
Rule added (v6)
```

You can now go to your server's IP address to check if Nginx is installed successfully. You should see a page like this:

Step 4 - Installing MySQL

Let's install MySQL using the command below:

```
sudo apt install mysql-server
```

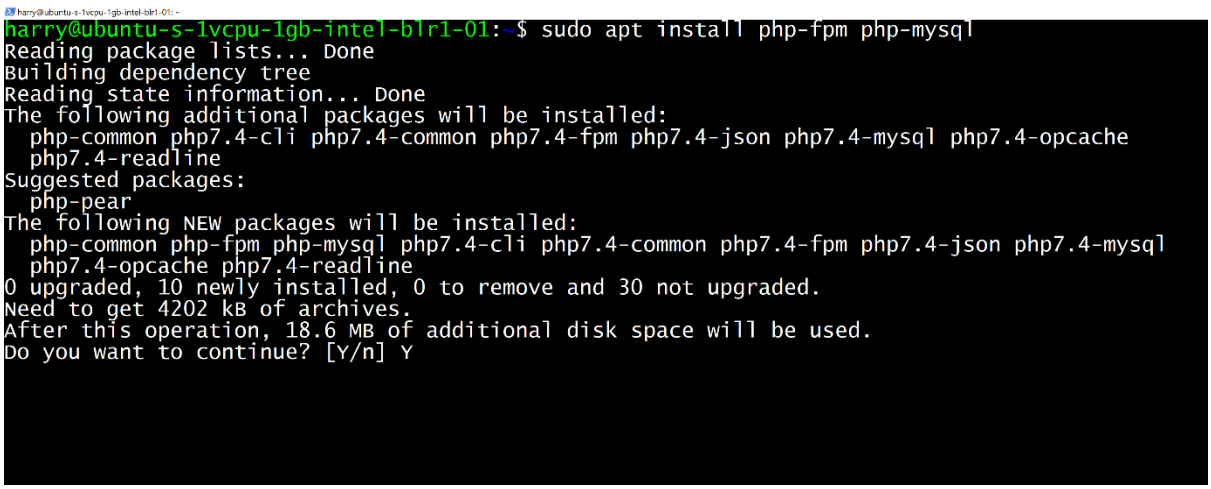
This command will install MySQL, and you will be able to see the console by entering "sudo mysql" in the terminal.

```
sudo mysql
```

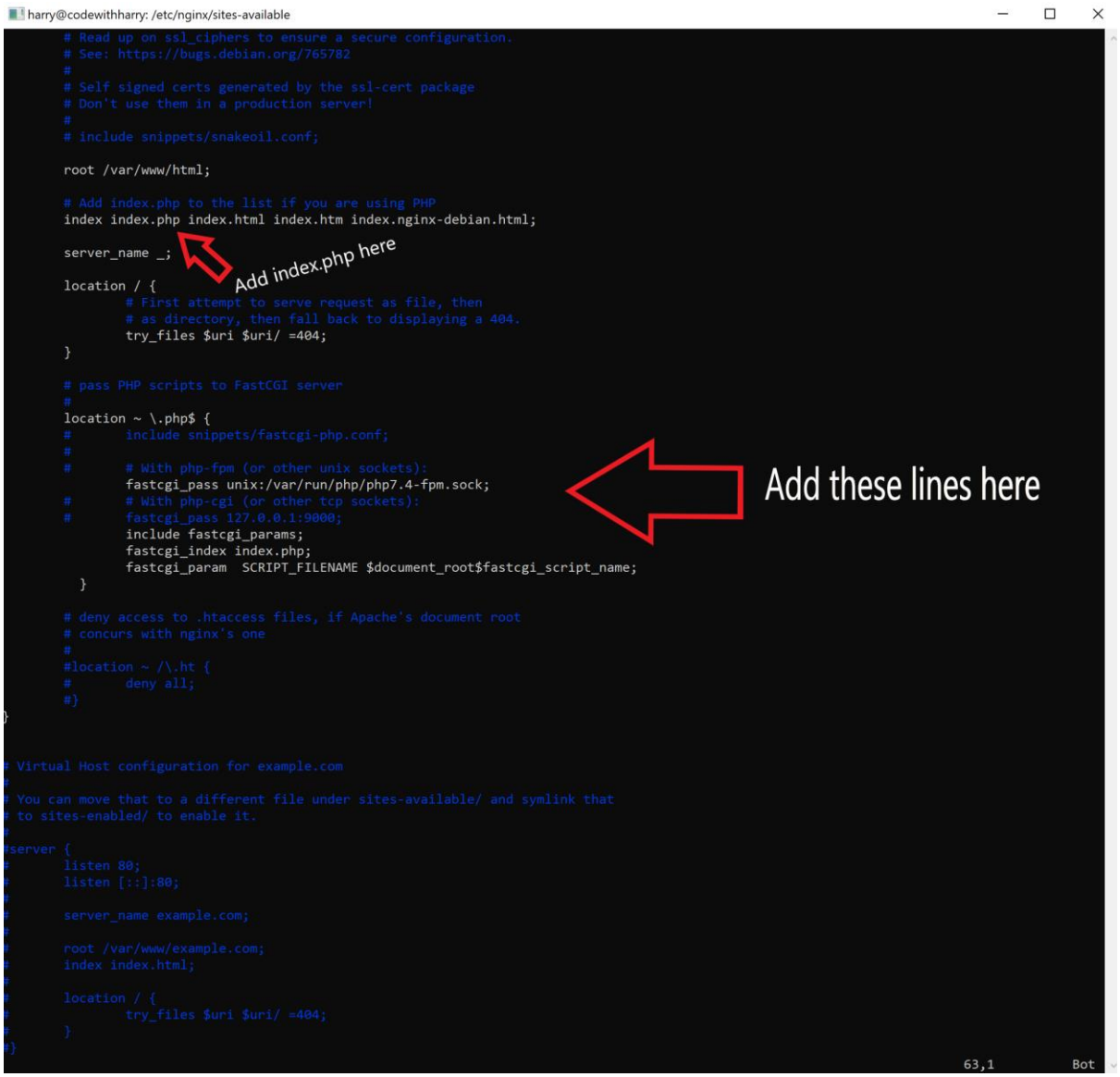
Step 5 - Installing PHP

The last component in the LEMP stack is PHP. Let's install it using the command below:

```
sudo apt install php-fpm php-mysql
```



If you want to host a PHP website, you will have to copy your files to '/var/www/html' and modify the file located at '/etc/nginx/sites-available/default' to look something like this:



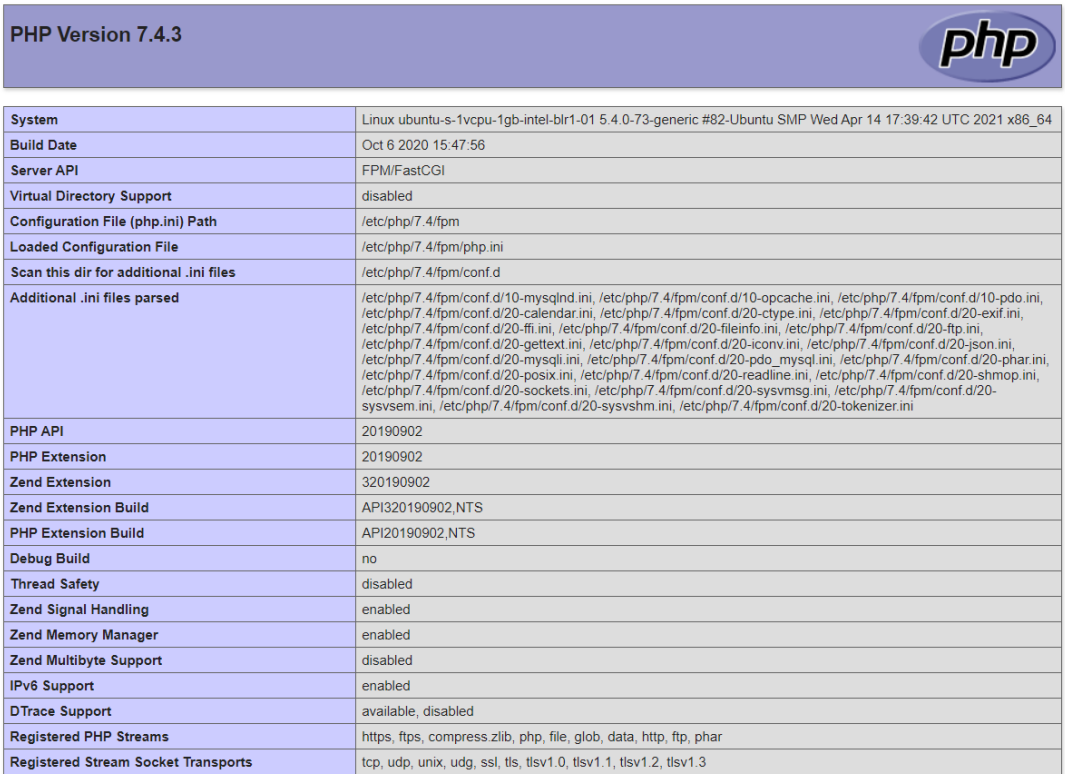
these are the additional lines added inside the "location ~ \.php$ {" block

```
include fastcgi_params;
fastcgi_index index.php;
fastcgi_param  SCRIPT_FILENAME
$document_root$fastcgi_script_name;
```

Here is a sample index.php site for you to try out:

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

Once you visit the IP of your droplet, you will see a page like this:



PHP Version 7.4.3

System	Linux ubuntu-s-1vcpu-1gb-intel-blr1-01 5.4.0-73-generic #82-Ubuntu SMP Wed Apr 14 17:39:42 UTC 2021 x86_64
Build Date	Oct 6 2020 15:47:56
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.4/fpm
Loaded Configuration File	/etc/php/7.4/fpm/php.ini
Scan this dir for additional .ini files	/etc/php/7.4/fpm/conf.d
Additional .ini files parsed	/etc/php/7.4/fpm/conf.d/10-mysqlnd.ini, /etc/php/7.4/fpm/conf.d/10-opcache.ini, /etc/php/7.4/fpm/conf.d/10-pdo.ini, /etc/php/7.4/fpm/conf.d/20-calendar.ini, /etc/php/7.4/fpm/conf.d/20-ctype.ini, /etc/php/7.4/fpm/conf.d/20-exif.ini, /etc/php/7.4/fpm/conf.d/20-ffi.ini, /etc/php/7.4/fpm/conf.d/20-fileinfo.ini, /etc/php/7.4/fpm/conf.d/20-ftp.ini, /etc/php/7.4/fpm/conf.d/20-gettext.ini, /etc/php/7.4/fpm/conf.d/20-iconv.ini, /etc/php/7.4/fpm/conf.d/20-json.ini, /etc/php/7.4/fpm/conf.d/20-mysqli.ini, /etc/php/7.4/fpm/conf.d/20-pdo_mysql.ini, /etc/php/7.4/fpm/conf.d/20-phar.ini, /etc/php/7.4/fpm/conf.d/20-posix.ini, /etc/php/7.4/fpm/conf.d/20-readline.ini, /etc/php/7.4/fpm/conf.d/20-shmop.ini, /etc/php/7.4/fpm/conf.d/20-sockets.ini, /etc/php/7.4/fpm/conf.d/20-sysvmsg.ini, /etc/php/7.4/fpm/conf.d/20-sysvsem.ini, /etc/php/7.4/fpm/conf.d/20-sysvshm.ini, /etc/php/7.4/fpm/conf.d/20-tokenizer.ini
PHP API	20190902
PHP Extension	20190902
Zend Extension	320190902
Zend Extension Build	API320190902,NTS
PHP Extension Build	API20190902,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
IPv6 Support	enabled
DTrace Support	available, 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

8. How to install PhpMyAdmin on Ubuntu running Nginx (LEMP stack)

Loading...

In this post, we will see how to install phpMyAdmin on servers running Nginx. Follow the steps below:

Step 1 - Installing phpMyAdmin

Enter the following command to install PHPMyAdmin

```
sudo apt install phpmyadmin php-mbstring php-zip php-gd php-json php-curl
```

```
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$ sudo apt install phpmyadmin php-mbstring php-zip php-gd php-
json php-curl
[sudo] password for harry:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  dbconfig-common dbconfig-mysql icc-profiles-free javascript-common libjs-jquery libjs-openlayers
  libjs-sphinxdoc libjs-underscore libonig5 libzip5 php php-bz2 php-google-recaptcha
  php-phpmyadmin-motranslator php-phpmyadmin-shapefile php-phpmyadmin-sql-parser php-phpseclib
  php-psr-cache php-psr-container php-psr-log php-symfony-cache php-symfony-cache-contracts
  php-symfony-expression-language php-symfony-service-contracts php-symfony-var-exporter php-tcpdf
  php-twig php-twig-extensions php-xml php7.4 php7.4-bz2 php7.4-curl php7.4-gd php7.4-mbstring
  php7.4-xml php7.4-zip
Suggested packages:
  php-dbase php-libsodium php-mcrypt php-gmp php-symfony-service-implementation php-imagick
  php-twig-doc php-symfony-translation www-browser php-recode php-gd2 php-pragmarx-google2fa
  php-bacon-qr-code php-samyoul-u2f-php-server
Recommended packages:
  php-mcrypt
The following NEW packages will be installed:
  dbconfig-common dbconfig-mysql icc-profiles-free javascript-common libjs-jquery libjs-openlayers
  libjs-sphinxdoc libjs-underscore libonig5 libzip5 php php-bz2 php-curl php-gd php-google-recaptcha
```

Step 2 - Configuring MySQL to use a password

Now we will configure password login to MySQL for logging into PHPMyAdmin

Open MySQL console

```
sudo mysql
```

```
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01: ~
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$ sudo mysql
[sudo] password for harry:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 21
Server version: 8.0.25-0ubuntu0.20.04.1 (Ubuntu)

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

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 USER 'harry'@'localhost' IDENTIFIED WITH caching_sha2_password BY 'MyStrongPassword1234$';
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT ALL PRIVILEGES ON *.* TO 'harry'@'localhost' WITH GRANT OPTION;
Query OK, 0 rows affected (0.01 sec)

mysql> exit
Bye
harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~$
```

and execute the following query to create a new user to be used for PHPMyAdmin.

```
CREATE USER 'harry'@'localhost' IDENTIFIED WITH caching_sha2_password BY 'MyStrongPassword1234$';
```

This creates a new MySQL user named harry with the password - 'MyStrongPassword1234$'

Let's provide this user all the privileges so that we can use it to access the PHPMyAdmin console. Execute the query below

```
GRANT ALL PRIVILEGES ON *.* TO 'harry'@'localhost' WITH GRANT OPTION;
```

Let's exit the MySQL console now

```
exit
```

Step 3 - Create a symlink

You'll now need to create a symbolic link from the PHPMyAdmin files to Nginx's document root directory. This will tell Nginx where PhpMyAdmin files are and how to serve them!

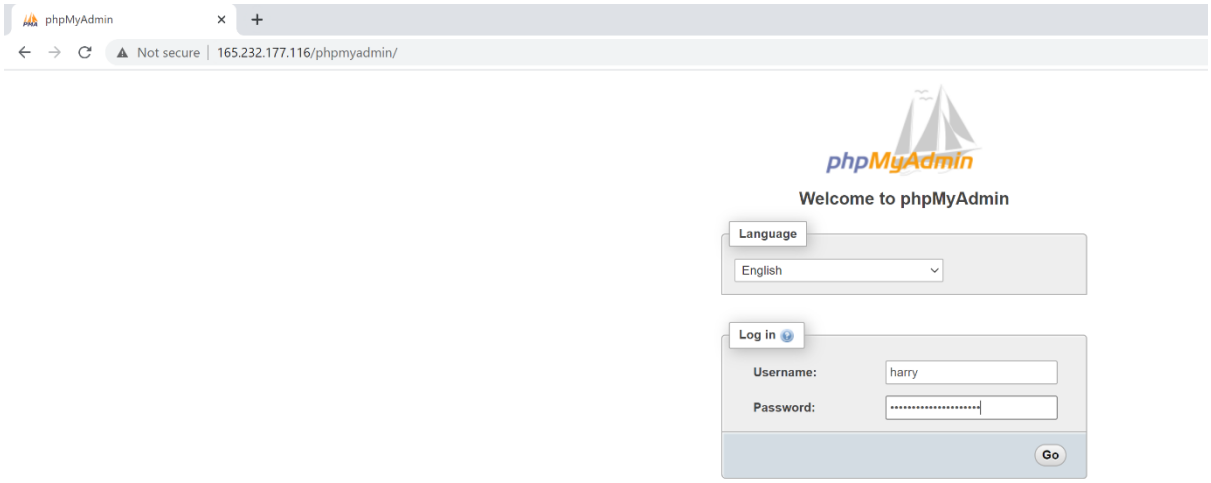
```
sudo ln -s /usr/share/phpmyadmin /var/www/html/phpmyadmin
```

Step 4 - Check the installation

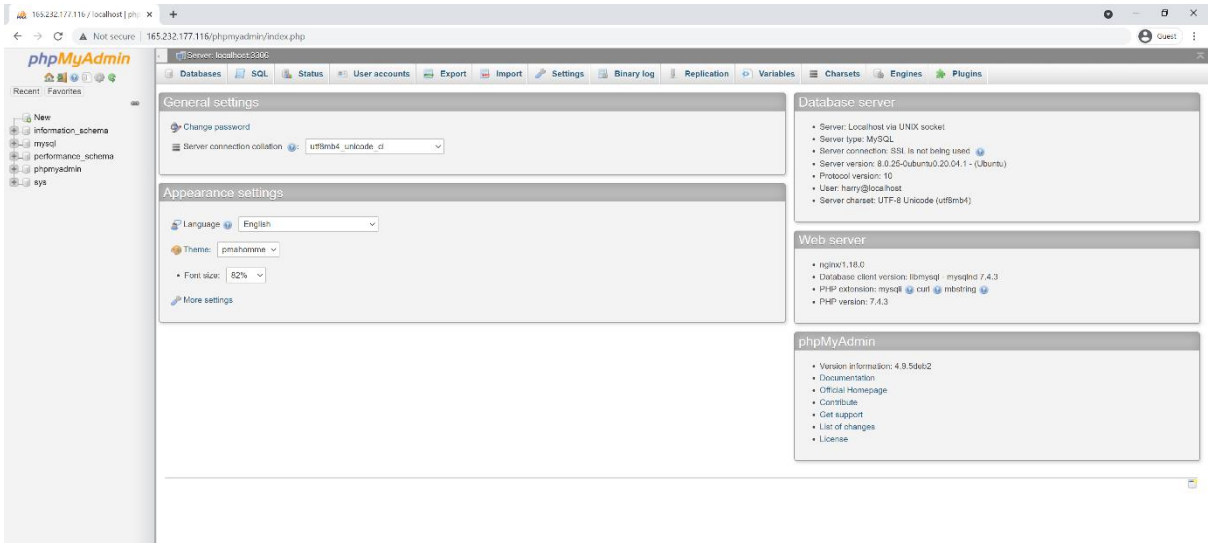
Restart Nginx using the command below:

```
sudo service nginx restart
```

Now go to the http://<your-server-ip>/phpmyadmin from your browser and login using the username and password you used in step 2.



You should be able to see the PHPMyAdmin interface.



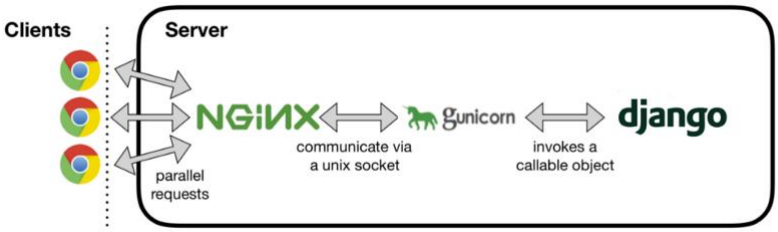
You should now have phpmyadmin configured on your server. You can use this interface to easily create, manage and update your databases.

9.How to host Django Application using gunicorn & nginx in Production

Loading...

In this post, we will see how to use nginx with gunicorn to serve django applications in production.

Django is a very powerful web framework and ships with a server which is able to facilitate development. This development server is not scalable and is not suited for production. Hence we need to configure gunicorn to get better scalability and nginx can be used as a reverse proxy and as a web server to serve static files. Let's get started



Before you follow the steps outlined below, I will assume that you have already configured your Ubuntu server with a non root user and firewall as outlined here.

Step 1 - Installing python and nginx

Let's update the server's package index using the command below:

```
sudo apt update
```

```
sudo apt install python3-pip python3-dev nginx
```

This will install python, pip and nginx server

Step 2 - Creating a python virtual environment

```
sudo pip3 install virtualenv
```

This will install a virtual environment package in python. Let's create a project directory to host our Django application and create a virtual environment inside that directory.

```
mkdir ~/projectdir
cd ~/projectdir
```

```
virtualenv env
```

A virtual environment named env will be created. Let's activate this virtual environment:

```
source env/bin/activate
```

Step 3 - Installing Django and gunicorn

```
pip install django gunicorn
```

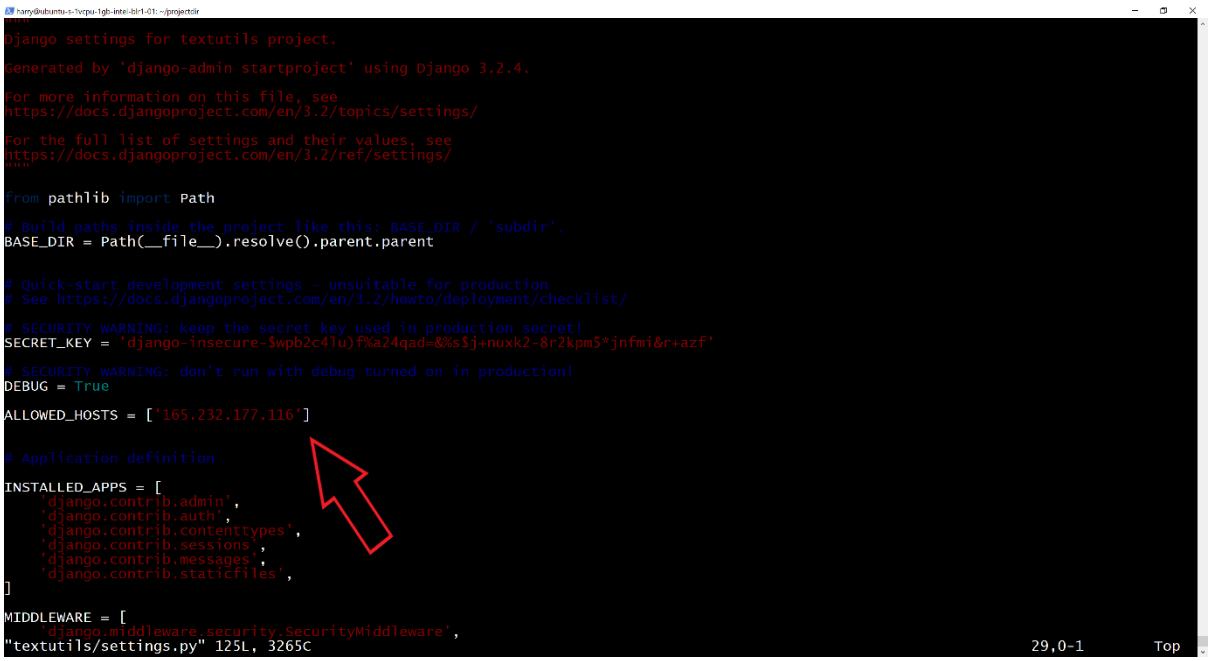
This installs Django and gunicorn in our virtual environment

Step 4 - Setting up our Django project

At this point you can either copy your existing Django project into the projectdir folder or create a fresh one as shown below:

```
django-admin startproject textutils ~/projectdir
```

Add your IP address or domain to the ALLOWED_HOSTS variable in settings.py.



If you have any migrations to run, perform that action:

```
~/projectdir/manage.py makemigrations
~/projectdir/manage.py migrate
```

Let's test this sample project by running the following commands:

```
sudo ufw allow 8000
```

This opens port 8000 by allowing it over the firewall. Let's start our Django development server to test the setup so far:

```
~/projectdir/manage.py runserver 0.0.0.0:8000
```

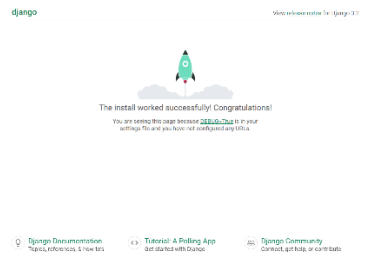
Step 5 - Configuring gunicorn

Lets test gunicorn's ability to serve our application by firing the following commands:

```
gunicorn --bind 0.0.0.0:8000 textutils.wsgi
```

```
(env) harry@ubuntu-s-1vcpu-1gb-intel-blr1-01:~/projectdir$ gunicorn --bind 0.0.0.0:8000 textutils.wsgi
[2021-06-06 06:26:24 +0000] [49911] [INFO] Starting gunicorn 20.1.0
[2021-06-06 06:26:24 +0000] [49911] [INFO] Listening at: http://0.0.0.0:8000 (49911)
[2021-06-06 06:26:24 +0000] [49911] [INFO] Using worker: sync
[2021-06-06 06:26:24 +0000] [49913] [INFO] Booting worker with pid: 49913
```

This should start gunicorn on port 8000. We can go back to the browser to test our application. Visiting http://<ip-address>:8000 shows a page like this:



Deactivate the virtualenvironment by executing the command below:

```
deactivate
```

Let's create a system socket file for gunicorn now:

```
sudo vim /etc/systemd/system/gunicorn.socket
```

Paste the contents below and save the file

```
[Unit]
Description=gunicorn socket

[Socket]
ListenStream=/run/gunicorn.sock

[Install]
WantedBy=sockets.target
```

Next, we will create a service file for gunicorn

```
sudo vim /etc/systemd/system/gunicorn.service
```

Paste the contents below inside this file:

```
[Unit]
Description=gunicorn daemon
Requires=gunicorn.socket
After=network.target

[Service]
User=harry
Group=www-data
WorkingDirectory=/home/harry/projectdir
ExecStart=/home/harry/projectdir/env/bin/gunicorn \
        --access-logfile - \
        --workers 3 \
        --bind unix:/run/gunicorn.sock \
        textutils.wsgi:application

[Install]
WantedBy=multi-user.target
```

Lets now start and enable the gunicorn socket

```
sudo systemctl start gunicorn.socket
sudo systemctl enable gunicorn.socket
```

Step 6 - Configuring Nginx as a reverse proxy

Create a configuration file for Nginx using the following command

```
sudo vim /etc/nginx/sites-available/textutils
```

Paste the below contents inside the file created

```
server {
    listen 80;
    server_name www.codewithharry.in;

    location = /favicon.ico { access_log off; log_not_found off; }
    location /static/ {
        root /home/harry/projectdir;
    }

    location / {
        include proxy_params;
        proxy_pass http://unix:/run/gunicorn.sock;
    }
}
```

Activate the configuration using the following command:

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

Restart nginx and allow the changes to take place.

```
sudo systemctl restart nginx
```

10.How to install Ubuntu Desktop with Full GUI on an Ubuntu VPS

Loading...

This post will explain to you how to install ubuntu-desktop on Ubuntu VPS. Follow the steps below:

Step 1 - Installing Ubuntu desktop using tasksel

Let's update the server's package index using the command below:

```
sudo apt update
```

Install tasksel using the command below:

```
sudo apt install tasksel
```

Now run tasksel and choose ubuntu-desktop using the spacebar

```
sudo tasksel
```

Step 2 - Reboot the system

Execute the following command to reboot the VPS:

```
sudo reboot
```

Step 3 - Install xrdp

Let's install xrdp using the command below:

```
apt install xrdp
```

Finally, enable xrdp using the command below:

```
systemctl enable xrdp
```

You should now be able to connect using the Windows Remote Desktop Connection. Enter the IP of your VPS followed by the username and password of your non-root account and start using Ubuntu Desktop

Step 4 - Customize Ubuntu Desktop like Windows (Optional)

You can customize your Ubuntu desktop installation to get a taskbar and taskbar icons like windows. Execute the command below to install

```
sudo apt install gnome-shell-extensions gnome-shell-extension-dash-to-panel gnome-tweaks adwaita-icon-theme-full
```

Reboot your Ubuntu server. Next, log in using RDP and search for Tweaks, and click extension.

Under extensions, enable "Dash to panel" extension and you will see the taskbar and application icons on your taskbar.

THAT'S ALL ABOUT SERVER Deployment.

Make a Website & Link and Manage Server…..