**Chapter -1**

**Commands to setup Nginx and PhpMyAdmin in Ubuntu EC2 Instance :**

***Note:*** *At first create the Ubuntu EC2 instance in your Amazon Web Service as shown in the video.*

**Install Linux, Nginx, MySQL, PHP (LEMP stack)**

-----------------------------------------------------------------------------------------------------------

1. **Update and Upgrade Ubuntu**

**sudo apt-get update && sudo apt-get upgrade**

------------------------------------------------------------------------------------------------------------

1. **Install the Nginx Web Server**

In order to display web pages to our site visitors, we are going to employ

 Nginx, a modern, efficient web server.

**sudo apt-get install nginx**

------------------------------------------------------------------------------------------------------------

1. **Now, if you give url in your browser as:**

**http://server\_domain\_or\_IP**

**You will see:**



-------------------------------------------------------------------------------------------------------------

1. **Install MySQL Server**

**sudo apt-get install mysql-server**

--------------------------------------------------------------------------------------------------------------

1. **Type Command to open mysql as:**

**sudo mysql**

-------------------------------------------------------------------------------------------------------------

1. **Create User using command as:-**

There will be appearing as mysql> and then type below command

**CREATE USER 'admin'@'%' IDENTIFIED BY 'admin';**

  It means that the user name is ‘admin’ and password is also ‘admin’

-----------------------------------------------------------------------------------------------------------

1. **Grant Permission using the command as:-**

**GRANT ALL PRIVILEGES ON \*.\* TO 'admin'@'%' WITH GRANT OPTION;**

---------------------------------------------------------------------------------------------------------------

1. **Install PHP for Processing**

**sudo apt-get install php-fpm php-mysql**

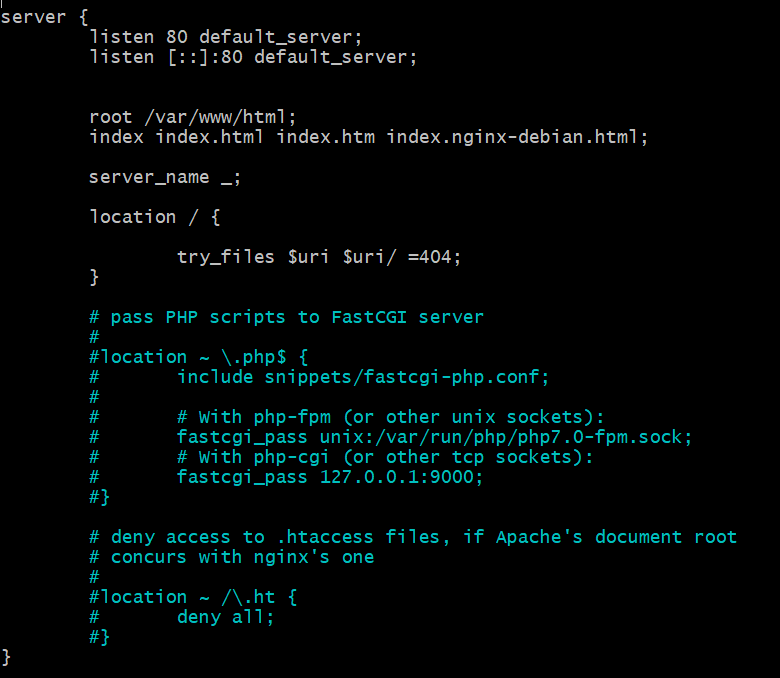
---------------------------------------------------------------------------------------------------------------

1. **Configure Nginx to Use the PHP Processor**

The only configuration change we still need is to tell Nginx to use our PHP processor for dynamic content. We do this on the server block level (server blocks are similar to Apache’s virtual hosts). Open the default Nginx server block configuration file by typing:

**sudo nano /etc/nginx/sites-available/default**

Initially you can see these in configuration



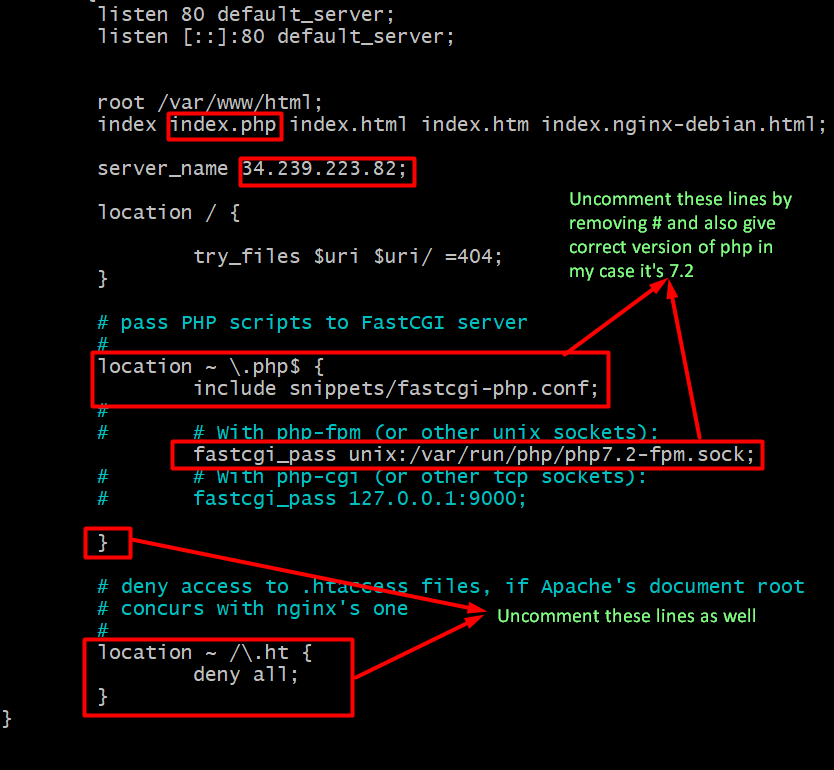
you need to change some of the above commented section to uncomment and add some of the lines as shown in below screenshot:

Check the php version before configuring as we need this in the below configuration process. To check the php version type the command as:

**php --version**

In below configuration, you have to add **index.php** as shown in screenshot below and **server\_name** is your ec2 instance Public IPv4 DNS or Public IPv4 address:

In my case it's an elastic ip address i have created i.e., **34.239.223.82.**

****

After changing into this above file type **ctrl+o** to write and save and **ctrl+x** to exit.

------------------------------------------------------------------------------------------------------------

1. **Type below command to check whether configuration files have error or not:**

**sudo nginx -t**

It should say:

**nginx: the configuration file /etc/nginx/nginx.conf syntax is ok**

**nginx: configuration file /etc/nginx/nginx.conf test is successful**

If it’s giving an error then re-check the configuration file in a calm manner.

-----------------------------------------------------------------------------------------------------------

1. **Reload the nginx:**

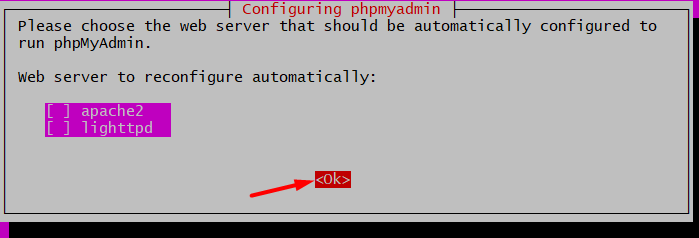
**sudo systemctl reload nginx**

----------------------------------------------------------------------------------------------------

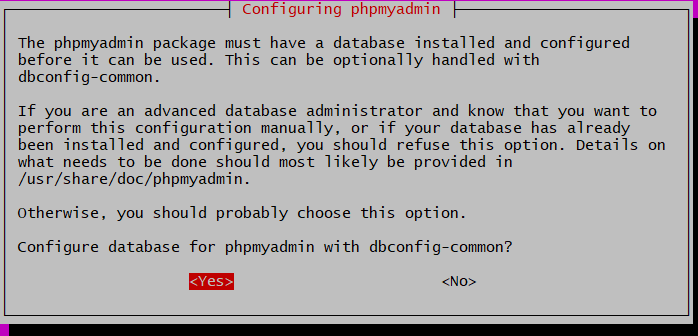
1. **Install PhpMyAdmin using the command as:**

**sudo apt-get install phpmyadmin**

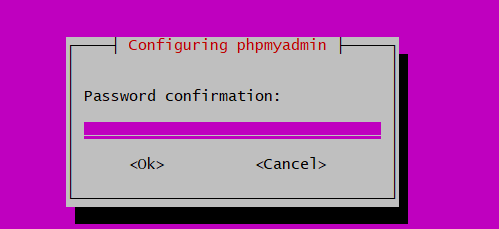
During the installation process of Phpmyadmin you will be asked to choose web server as we are using ‘Nginx’ as a web server so don’t choose any of them press ‘tab’ and press enter on ‘ok’.



After that you will be asked for configure database for phpmyadmin with dbconfig-common hit enter on ‘yes’ option’



After that you will be asked for a password. Give the password that is used to create the user as in **step 7** in my case it’s ‘admin’. I will give the same password.



Give password and confirm the password as well.

----------------------------------------------------------------------------------------------------

1. **Finally, Create symbolic link:**

For the Nginx web server to find and serve the phpMyAdmin files correctly, we’ll need to create a symbolic link. Use the command as:

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

This will create the symbolic link also known as symlink to the /var/www/html directory.

After that restart nginx

**sudo systemctl restart nginx**

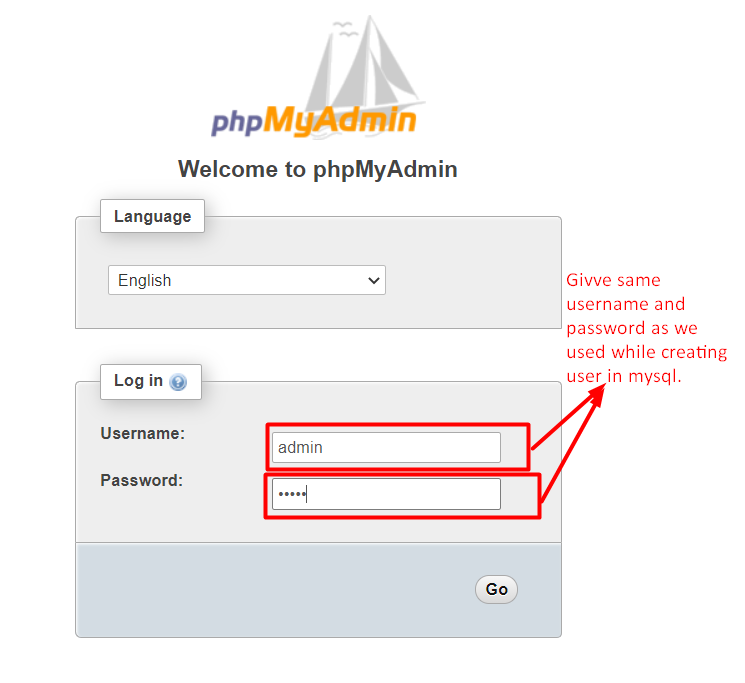
----------------------------------------------------------------------------------------------------

1. **Now all configuration Completed!! Go to your browser and give url as**

<http://you_public_ipv4_address/phpmyadmin/>

In my case it’s:

<http://34.239.223.82/phpmyadmin/>

****

We have successfully configured **phpMyAdmin** with **Nginx** web server.

**Chapter-2**

**Setup Django Project With Gunicorn and Supervisor In AWS**

*[Note: use sudo Infront of all commands for ubuntu]*

1. **At first setup the project in ubuntu EC2 instance**

Install virtualenv with the command as:

sudo apt-get install python3-venv

To create virtualenv use:

python3 -m venv env\_name

------------------------------------------------------------------------------------------------------------

1. **Install mysqlclient inside the virtual environment of EC2 instance:**

pip install mysqlclient

If error occurred then just install some of the file as ‘.whl’ file won’t be supported in linux platform.

***[Note: If error occurred while installing mysqlclient give command in new terminal as:]***

sudo apt-get install python3 python3-dev python3-pip libxml2-dev libxslt1-dev zlib1g-dev libffi-dev libssl-dev

Also Run:

sudo apt install libmysqlclient-dev

------------------------------------------------------------------------------------------------------------

1. **­Install Gunicorn:**

Gunicorn 'Green Unicorn' is a Python WSGI HTTP Server for UNIX. Gunicorn knows how to run a web application based on the hook between the WSGI server and the WSGI-compliant web app.

pip install gunicorn

To learn more about Gunicorn follow below link:

<https://www.fullstackpython.com/green-unicorn-gunicorn.html>

----------------------------------------------------------------------------------------------------------------------------------

1. **Bind the application with gunicorn:**

Specify a server socket to bind.

**Syntax:**

gunicorn --bind 0.0.0.0:8000 Project\_Name.wsgi:application

**Example:**

gunicorn --bind 0.0.0.0:8000 DemoProject.wsgi:application

----------------------------------------------------------------------------------------------------------------------------------

1. **Install supervisor:**

Supervisor is a process manager to ensure gunicorn starts, restarts, and runs when we need it. Supervisor will look after the Gunicorn process and make sure that they are restarted if anything goes wrong, or to ensure the processes are started at boot time.

sudo apt-get install -y supervisor

----------------------------------------------------------------------------------------------------------------------------------

1. **Configure supervisor**

Create configuration so that supervisor could read from that configuration.

**Step 1:** Go to the conf.d directory

cd /etc/supervisor/conf.d/

**Step 2:** Create gunicorn.conf file

sudo touch gunicorn.conf

**Step 3:** Configure the gunicorn.conf file with following lines of code:

sudo nano gunicorn.conf

configuration file will be open in nano text editor and inside that file copy below configurations:

**Syntax:**

[program:gunicorn]

Directory=path\_to\_django\_project

command=path\_to\_gunicorn –workers 3 --bind unix:/path\_to\_project/app.sock Project\_Name.wsgi:application

autostart=true

autorestart=true

stderr\_logfile=/var/log/gunicorn/gunicorn.err.log

stdout\_logfile=/var/log/gunicorn.out.log

[group:guni]

programs:gunicorn

**Example:**

[program:gunicorn]

Directory=/var/www/html/Django

command=/var/www/html/demo\_env/bin/gunicorn --workers 3 --bind unix:/var/www/html/Django/app.sock DemoProject.wsgi:application

autostart=true

autorestart=true

stderr\_logfile=/var/log/gunicorn/gunicorn.err.log

stdout\_logfile=/var/log/gunicorn.out.log

[group:guni]

programs:gunicorn

Now, press ctrl+o to write and save and ctrl+x to exit.

----------------------------------------------------------------------------------------------------------------------------------

1. **Create gunicorn directory in following path:**

mkdir /var/log/gunicorn

----------------------------------------------------------------------------------------------------------------------------------

1. **Re-read and then update supervisor to configuration file:**

sudo supervisorctl reread

It must say guni:available if not then you might have made some mistakes in gunicorn.conf file.

sudo supervisorctl update

----------------------------------------------------------------------------------------------------------------------------------

1. **Start supervisor to start at background:**

sudo supervisorctl status

It must say RUNNING pid …..

----------------------------------------------------------------------------------------------------------------------------------

1. **Go to the sites-available directory and configure below code:**

**sudo nano /etc/nginx/sites-available/default**

Inside **location** area add followings:

**Syntax:**

location / {

include proxy\_params;

proxy\_pass http://unix:/path\_to\_project/app.sock;

}

**Example:**

location / {

include proxy\_params;

proxy\_pass http://unix:/var/www/html/Django/app.sock;

}

Check whether the configuration file is ok or not by giving the command as:

sudo nginx -t

----------------------------------------------------------------------------------------------------------------------------------

1. **Configure the static file to be load by nginx:**

Inside same nginx .conf file add following configuration to configure ‘static’ file for the project.

location /static/ {

autoindex on;

alias /var/www/html/Django/static/;

}

**It should say:**

nginx: the configuration file /etc/nginx/nginx.conf syntax is ok

nginx: configuration file /etc/nginx/nginx.conf test is successful

----------------------------------------------------------------------------------------------------------------------------------

1. **Restart nginx and the supervisor using the command as:**

sudo service nginx restart

sudo supervisorctl restart all

----------------------------------------------------------------------------------------------------------------------------------

**[Not included in Video Tutorial]**

**NOTE:** If you want to go to the phpmyadmin directory again after all these above configurations by giving only your\_public\_ip/phpmyadmin will give **you url not found error** for that go to the directory as:

cd /etc/nginx/sites-available/default

Then add following configuration below and save it ctrl+o and ctrl+x

location /phpmyadmin/

{

root /usr/share/;

index index.php;

try\_files $uri $uri/ =404;

}

Then restart nginx:

sudo service nginx restart

-----------------------------------------------------\*\*\* The End \*\*\*--------------------------------------------------------