**How to Point Domain and Host Django Project on Nginx Web Server**

* Get Access to Remote Server via SSH

Syntax:- ssh -p PORT USERNAME@HOSTIP

Example:- ssh -p 22 raj@216.32.44.12

#### **Note:- Run Below Commands on Your Remote Server Linux Machine or VPS, Not on Your Local Windows Machine**

* Verify that all required softwares are installed

nginx -v

python --version

pip --version

- SQLite is Included with Python

python -c "import sqlite3; print(sqlite3.sqlite\_version)"

* If Required Softwares and Modules are not Installed then Install them:

sudo apt install nginx

sudo apt install python

sudo apt install libapache2-mod-wsgi-py3

sudo apt install python3-pip

* Install virtualenv

pip list

sudo pip install virtualenv

* Verify Apache2 is Active and Running

sudo systemctl nginx restart

* Verify Web Server Ports are Open and Allowed through Firewall

sudo ufw status verbose

* Go to Your Project Directory

Syntax:- cd /var/www/project\_folder\_name

Example:- cd /var/www/hires\_api

* Create Virtual env

Syntax:- virtualenv env\_name

Example:- virtualenv env

* Activate Virtual env

Syntax:- source virtualenv\_name/bin/activate

Example:- source env/bin/activate

* Install Dependencies

pip install -r requirements.txt

## How to Install PostgreSQL On Ubuntu

**Follow the steps in the sections below to install PostgreSQL from the PostgreSQL repository.**

**Step 1: Add PostgreSQL Repository**

Run the following command to add the PostgreSQL repository to your system:

sudo sh -c 'echo "deb https://apt.postgresql.org/pub/repos/apt $(lsb\_release -cs)-pgdg main" > /etc/apt/sources.list.d/pgdg.list'

**Step 2: Add the Repository Signing Key**

The next step is to fetch the repository's GPG key and add it to APT's trusted keyring. This allows APT to verify the authenticity of packages downloaded from the PostgreSQL repository.

Run the command below:

wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add –

**Step 3: Update the Package List**

After adding the official PostgreSQL repository, update the package list to ensure you install the latest PostgreSQL package.

sudo apt update

**Step 4: Install PostgreSQL**

To install **PostgreSQL** run the following command:

sudo apt install postgresql

## **Verify PostgreSQL Installation**

Verify that PostgreSQL has been installed by checking the PostgreSQL service status. Run the command below:

sudo systemctl status postgresql

## **Connect to PostgreSQL**

To establish a connection with the database, log into the **postgres** account with:

sudo -i -u postgres

Next, open a **postgres** prompt using the command:

psql

Use the following syntax to create a postgres password:

\password postgres

You will be prompted to type a new password. Repeat this for the owner and postgres user, giving each a strong, unique password.

To exit psql, type

\q.

Use the following syntax to create a database:

Syntax: CREATE DATABASE [dbname];

Example: CREATE DATABASE hires\_database;

## Check Connection Information

If you are connected to PostgreSQL and want to see details of the connection, use the command:

\conninfo

Use the following syntax to create a user and password:

Syntax: CREATE USER <username> with ENCRYPTED PASSWORD ‘<password>';

Example: CREATE USER hires with ENCRYPTED PASSWORD ‘postgres@123';

Use the following syntax to create a user and password:

Syntax: ALTER DATABASE <dbname> OWNER TO <username>;

ALTER DATABASE hires\_database OWNER TO hires;

To exit psql, type

\q.

Allow PostgreSQL TCP port 5432 in the Firewall

PostgreSQL default HTTP port is 5432, you’ll need to allow access to this port on the firewall.

If your firewall is UFW type the following commands:

sudo ufw allow 5432/tcp

## **Add PostgreSQL Server to pgAdmin**

**Check PostgreSQL Configuration**

1. Verify postgresql.conf:

Locate the postgresql.conf file. Its location can vary, but it's often in

cd /etc/postgresql/16/main/

sudo nano postgresql.conf

listen\_addresses = '\*'

1. Verify pg\_hba.conf:

Locate the pg\_hba.conf file. Its location can vary, but it's often in

cd /etc/postgresql/16/main/

sudo nano pg\_hba.conf

#IPv4 Addresses

host all all 0.0.0.0/0 md5

#IPv6 Addresses

host all all ::0/0 md5

1. Restart PostgreSQL configuration to apply changes:

sudo service restart postgresql

1. Reload PostgreSQL configuration to apply changes:

sudo service reload postgresql

## **Add PostgreSQL Server to pgAdmin**

In pgAdmin, click on the “Servers” menu on the left sidebar, then right-click and choose “Register” then select “Server” Enter a name for your server and switch to the “Connection” tab. Fill in the following details:

**Host name/address:** <server-ip address>  
**Port:** 5432  
**Maintenance database:** postgres  
**Username:** postgres  
**Password:** (Enter the password you set for the PostgreSQL user)

Click “Save” to add the server.

Congratulations! You’ve successfully installed and configured PostgreSQL and pgAdmin on your Ubuntu system. You can now use pgAdmin to manage your databases, tables, and perform various administrative tasks through its intuitive interface.

* Serve Static Files

cd /var/www/hires\_backend

python manage.py collectstatic

* Create Database Tables

python manage.py makemigrations

python manage.py migrate

* Create Superuser

python manage.py createsuperuser

* If Database File throws error Permission Denied then Set below permissions
* Make Webserver as owner for database file. Our Webserver is running as www-data and group is also www-data.

Syntax:-

sudo chown -R www-data:www-data database\_folder

sudo chmod 775 database\_folder

sudo chmod 664 database\_folder/database\_file

Example:-

sudo chown -R www-data:www-data mbdb

sudo chmod 775 mbdb

sudo chmod 664 mbdb/db.sqlite3

* If Media Files (User Uploaded Files) throws error Permission Denied then Set below permissions

sudo chown -R www-data:www-data media

* If needed Deactivate Virtual env

deactivate

* Create hires.ini file
  1. **The file projectname.ini is likely a configuration file for uWSGI. The projectname.ini file extension suggests it’s an INI-style configuration file used   
     to define how uWSGI should run and serve your application.**
  2. **In your case, the path suggests it might be   
     set up to handle the uWSGI configuration for a site called projectname. The sites directory in /etc/uwsgi/ is commonly used for configuration files for different   
     applications or sites.**
* Create a file of hires.ini

Syntax:- sudo nano /etc/uwsgi/sites/projectname.ini

Example:- sudo nano /etc/uwsgi/sites/hires.ini

* Write a below code on hires.ini



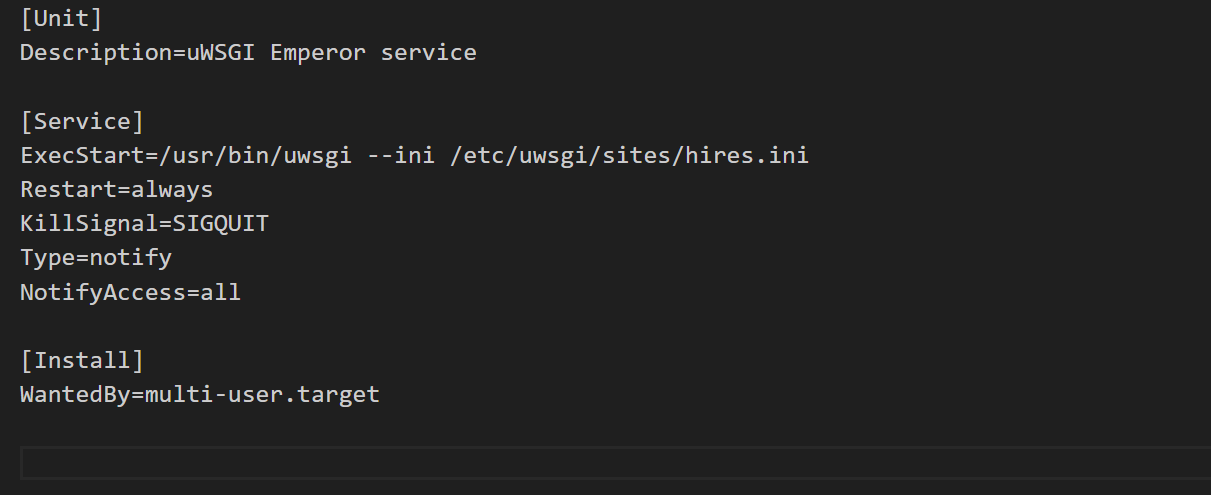
* Create uwsgi.service file

The uwsgi.service is a systemd service unit file that is used to manage uWSGI as a service on a Linux system. uWSGI is a popular application server for deploying Python web applications. It can communicate with various web servers using protocols like WSGI, FastCGI, and others. When you install uWSGI, you often create a uwsgi.service file to configure how uWSGI should start, stop, and restart with the system.

Syntax:- sudo nano /etc/systemd/system/uwsgi.service

Example:- sudo nano /etc/systemd/system/uwsgi.service

* Write a below code on uwsgi.service



* Relaod deamon

sudo systemctl daemon-reload

* Start uwsgi

sudo systemctl start uwsgi

* Enable uwsgi

sudo systemctl enable uwsgi

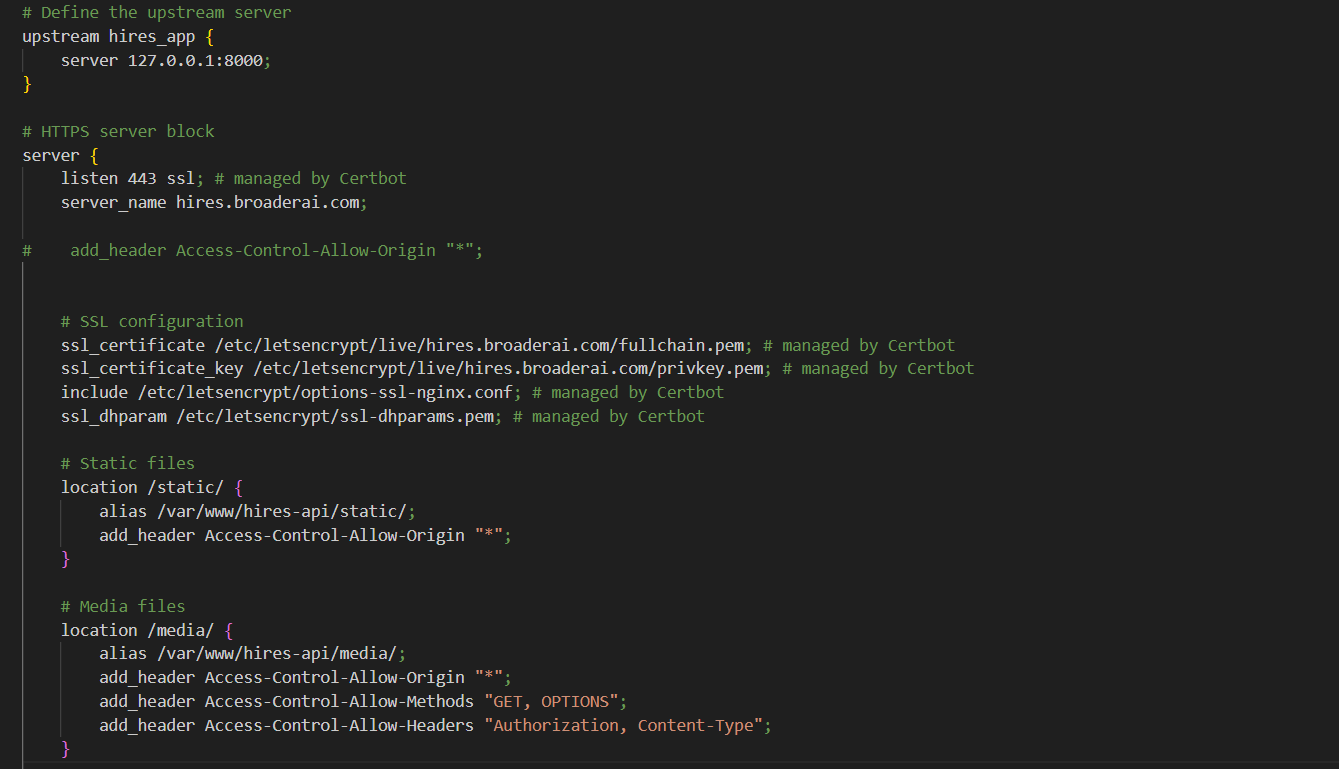
* Restart nginx

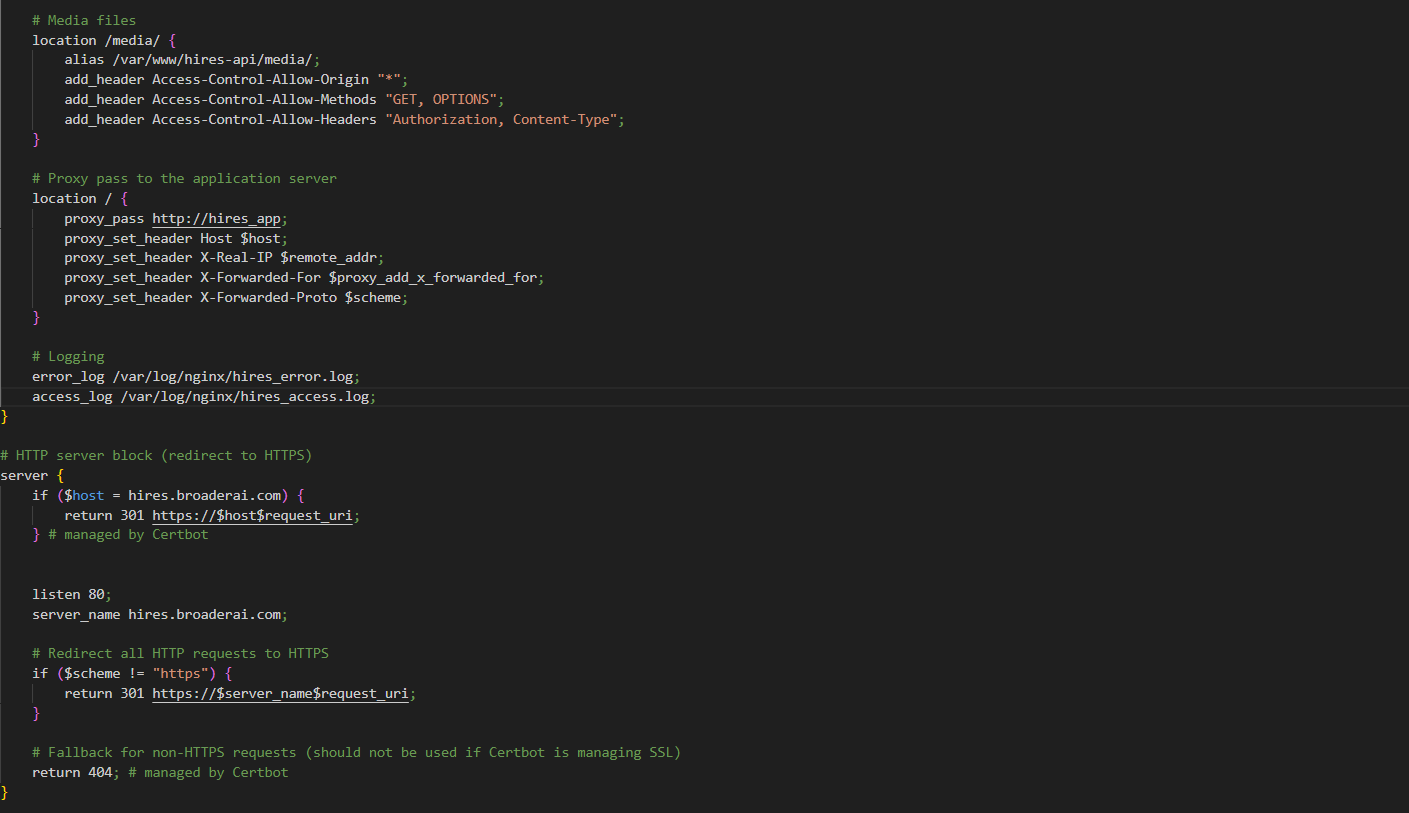
sudo systemctl restart nginx

* Create Virtual Host File

Syntax:- sudo nano /etc/nginx/conf.d/your\_domain.conf

Example:- sudo nano /etc/nginx/conf.d/hires.broaderai.com.conf





* Check Configuration is correct or not

sudo nginx -t

* Enable Virtual Host

cd /etc/nginx/conf.d  
sudo a2ensite hires.broaderai.com.conf

* Restart nginx

sudo systemctl restart nginx

* Reload nginx

sudo systemctl reload nginx

**Supervisor configuration on linux**

* Install Supervisor

sudo apt-get install supervisor

* To Configure our supervisor, we will have to create few configuration files and set them up

Example:

# open editer

sudo nano /etc/supervisor/conf.d/myproject.conf

# copy paste the command and modify your project location

[program:myproject]

command=/path/to/your/venv/bin/gunicorn myproject.wsgi:application - workers 3 - bind 0.0.0.0:8000

directory=/path/to/your/django/project

autostart=true

autorestart=true

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

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

# directory: Specifies the working directory of your Django project.

# autostart and autorestart: Ensure that Supervisor starts and restarts the process automatically.

# stderr\_logfile and stdout\_logfile: Log files for standard error and standard output.

* Create Supervisor configuration on linux

Syntax:- sudo nano /etc/supervisor/conf.d/myproject.conf

Example:- sudo nano /etc/supervisor/conf.d/hires.conf

#example of your django project

sudo nano /etc/supervisor/conf.d/hires.conf

[program:hires]

command=/var/www/hires-api/env/bin/gunicorn hires.wsgi:application - workers 3 - bind 0.0.0.0:8000

directory=/var/www/hires-api

autostart=true

autorestart=true

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

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

* Reread the supervisor

sudo supervisorctl reread

* Update the supervisor

sudo supervisorctl update

* Status about supervisor

sudo supervisorctl status

* If you have any change on Django file on your python project then you restart the supervisor via gunicorn

Run the following command to restart the specific process:

Syntax: sudo supervisorctl restart app\_name

Example: sudo supetvisorctl restart hires

* Reread the supervisor

sudo supervisorctl reread

* Update the supervisor

sudo supervisorctl update

* Status about supervisor

sudo supervisorctl status

### How to Point Domain and Host ReactJS Project using Nginx Web Server

* To Access Remote Server via SSH

Syntax:- ssh -p PORT USERNAME@HOSTIP

Example:- ssh -p 22 raj@216.32.44.12

#### **Note:- Run Below Commands on Your Remote Server Linux Machine or VPS, Not on Your Local Windows Machine**

* Verify that all required software are installed

nginx -v

node -v

npm -v

git --version

* Install Apache

sudo apt install nginx

* Install Node and npm

curl -fsSL https://deb.nodesource.com/setup\_18.x | sudo -E bash - &&\

sudo apt-get install -y nodejs

* Verify Nginx is Active and Running

sudo systemctl status nginx

* Go to Your Project Directory

Syntax:- cd /var/www/project\_folder\_name

Example:- cd /var/www/hires-recruiters

* Install Dependencies

npm install

* Create Production Build

npm run build

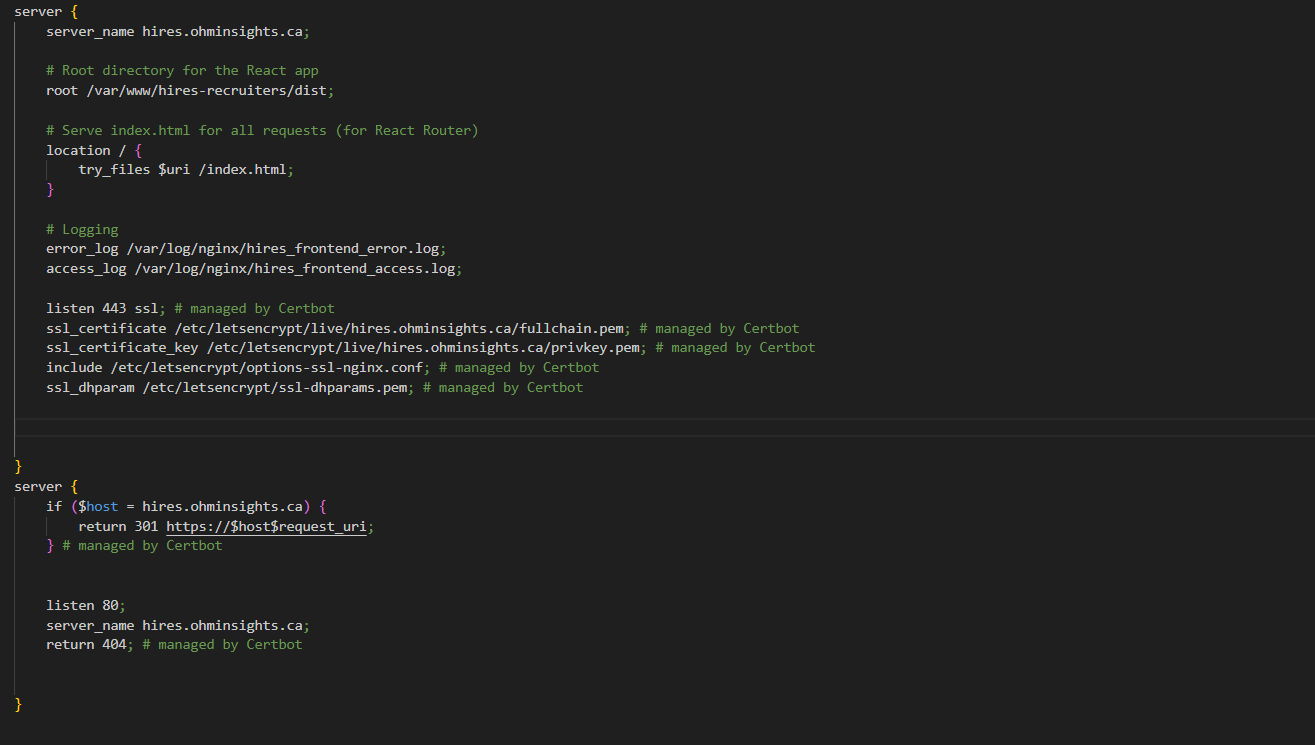
// OR

npm run export

* Create Virtual Host File

Suntax: sudo nano /etc/nginx/conf.d/your\_domain.conf

Example: sudo nano /etc/nginx/conf.d/hires.ohminsights.ca.conf



* Enable Virtual Host

cd /etc/nginx/conf.d

sudo a2ensite hires.ohminsights.ca.conf

* Check Configuration is correct or not

sudo nginx -t

* Restart and Reload Nginx

sudo systemctl nginx restart

sudo systemctl nginx reload

### How to Enable HTTPS in Your Domain Hosted on Linux Remote Server or VPS

#### Let's Encrypt is a non-profit certificate authority run by Internet Security Research Group that provides X.509 certificates for Transport Layer Security encryption at no charge.

* To Access Remote Server via SSH

Syntax:- ssh -p PORT USERNAME@HOSTIP

Example:- ssh -p 22 root@216.32.44.12

* Install Certbot and python3-certbot-apache

sudo apt install certbot python3-certbot-nginx

- Certbot is a free, open source software tool for automatically using Let’s Encrypt certificates on manually-administrated websites to enable HTTPS.

- python3-certbot-nginx is a nginx plugin for Certbot.

* Verify Web Server Ports are Open and Allowed through Firewall

sudo ufw status verbose

* Obtain an SSL certificate

sudo certbot --nginx

* Check Status of Certbot

sudo systemctl status certbot.timer

* Dry Run SSL Renewal

sudo certbot renew --dry-run

### Example:-

### 