**Django project deployment**

**Project deployment on aws (ec2) instance –**

**Step 1: Create an EC2 Instance**

**Step 2: Configure Security Group**

* SSH (port 22) for remote access to your instance
* HTTP (port 80) for web traffic
* HTTPS (port 443) for secure web traffic

**Step 3: Connect to your EC2 Instance**

Now that you have created and configured your EC2 instance, it’s time to connect to it. You will need to use an SSH client to connect to your instance.

**Step 4: 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

**Step 5: Creating a Python virtual environment**

sudo pip3 install virtualenv  
sudo apt install python3-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.

git clone <https://github.com/yupt1234/chatApp.git>

cd ~/chatApplication

then

virtualenv env

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

source env/bin/activate

pip install -r requirements.txt

**Step 6: Installing Django and gunicorn**

pip install django gunicorn

This installs Django and gunicorn in our virtual environment

**Step 7: Setting up our Django project**

Add your IP address or domain to the ALLOWED\_HOSTS variable in settings.py.

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

python manage.py makemigrations

python manage.py migrate

python manage.py collectstatic

Import Thing about static files, You must make sure to add few lines in your seeting.py file.

1. Add this line, “**whitenoise.runserver\_nostatic**”, into your **Installed\_apps** of setting file.
2. Add ‘whitenoise.middleware.WhiteNoiseMiddleware’, into MiddleWare of your setting File.
3. Also, add these lines at the bottom of the *blog/urls. py file.*

if settings.DEBUG:  
 urlpatterns += static(settings.MEDIA\_URL, document\_root = settings.MEDIA\_ROOT)  
 urlpatterns += static(settings.STATIC\_URL, document\_root = settings.STATIC\_URL)

1. Also, add these imports lines at the top of the *blog/urls. py file.*

from django.conf import settings # new  
from django.conf.urls.static import static #new

1. Run this command to install the whitenoise

pip install whitenoise

**Step 8: Configuring gunicorn**

Deactivate the virtual environment 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=ubuntu  
Group=www-data  
WorkingDirectory=/home/ubuntu/chatApp/chatApplication  
ExecStart=/home/ubuntu/ chatApp/chatApplication/env/bin/gunicorn \  
 --access-logfile - \  
 --workers 3 \  
 --bind unix:/run/gunicorn.sock \  
 websock.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

with this command, you can check if already a file exists.

cd /etc/nginx/sites-enabled/

**Step 9: Configuring Nginx as a reverse proxy**

Create a configuration file for Nginx using the following command

sudo vim /etc/nginx/sites-available/websock

Paste the below contents inside the file created

server {  
 listen 80 default\_server;  
 server\_name \_; location = /favicon.ico { access\_log off; log\_not\_found off; }  
 location /static/ {  
 root /home/ubuntu/chatApp/chatApplication;  
 } 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/websock /etc/nginx/sites-enabled/

Run this command to load a static file

$ sudo gpasswd -a www-data username

Restart nginx and allow the changes to take place.

sudo systemctl restart nginx  
sudo service gunicorn restart

sudo service nginx restart

**Additionally in case of errors**

To check error logs

$ sudo tail -f /var/log/nginx/error.log

to check nginx working fine

$ sudo systemctl status nginx

sudo fuser -k 8000/tcp

sudo lsof -t -i tcp:8000 | xargs kill -9. # to kill terminal

# <https://amalgjose.com/2020/02/27/gunicorn-connection-in-use-0-0-0-0-8000/>