Fundoo Application Deployment (Django)

Database server:

Logged into database server which is present in Private subnet.

Install Postgresql for Database

->sudo apt install postgresql postgresql-contrib

Version of Postgresql

->psql –version

Postgres workbench

->sudo –u postgres psql

postgres=#

Creating database

->CREATE DATABASE pushpadb;

Creating user with password

->CREATE USER pushpa WITH PASSWORD ‘root’;

Grant Permission to user

->GRANT ALL PRIVILEGES ON DATABASE pushpadb TO pushpa;

List of databases

->\l;

List of tables in current database

->\dt;

To display the all credentials from specific table

->select * from <table_name>;

Need to change the Listen address of postgresql.conf file to accept connections on all available IP addresses

->sudo nano /etc/postgresql/16/main/postgresql.conf

listen_addresses = '*'

Need to host-based authentication configuration in pg_hba.conf

->sudo nano /etc/postgresql/16/main/pg_hba.conf

host all all 0.0.0.0/0 md5

->sudo systemctl restart postgresql

->sudo systemctl status postgresql

Backend server

sudo apt update
sudo apt install python3 python3-pip python3-venv –y

Clone the Git hub repository
->https://github.com/Aniket2659/Aws_test

Navigate to Project Directory
->cd /Aws_test

Create Virtual Environment because it isolates our project's dependencies from the global Python environment.
->python3 -m venv myenv
->source myenv/bin/activate

Install the Django framework
->pip install django gunicorn

Install Requirements
->pip install -r requirements.txt

-> (myenv) ubuntu@ip-10-0-2-142:~/Aws_test/fundoo_notes$ pip install python-decouple

Version of python-decouple
->(myenv) ubuntu@ip-10-0-2-142:~/Aws_test/fundoo_notes$ pip freeze | grep python-decouple
python-decouple==3.8

Run the Django application server again
->(myenv) ubuntu@ip-10-0-2-142:~/Aws_test/fundoo_notes$ nohup python3 manage.py runserver 0.0.0.0:8000 &

Now I need to configure my Django for PostgreSQL
-> Need to edit the settings.py file with environment variables for Database because to avoid the hard coded values

```
from decouple import config

from decouple import Config
from decouple import RepositoryEnv

# Load the env.conf file correctly
config = Config(RepositoryEnv('/etc/fundoo/env.conf'))
```

```
DATABASES = {
   'default': {
      'ENGINE': 'django.db.backends.postgresql',
      'NAME': config('DB_NAME'),
      'USER': config('DB_USER'),
      'PASSWORD': config('DB_PASSWORD'),
      'HOST': config('DB_HOST'),
      'PORT': config('DB_PORT', default='5432'),  # Optional default value
   }
}
```

Here I have created one folder as /fundoo in /etc
In /fundoo folder I created one .conf file as **env.conf** for environment variables of database

```
->sudo nano /etc/fundoo/env.conf
DB_NAME=pushpadb
DB_USER=pushpa
DB_PASSWORD=root
DB_HOST=10.0.3.54
DB_PORT=5432
```

Note:

- Here I used python decouple for loading environment variables from env.conf mentioned in settings.py

```
->python3 manage.py makemigrations
```

It applies the migrations to your database

```
->python3 manage.py migrate
```

Then I have .service file in /etc/systemd/system directory

fundoo.service

```
[Unit]
Description=Fundoo Service
After=network.target

[Service]
User=pushpa
Group=pushpa
EnvironmentFile=/etc/fundoo/env.conf
```

WorkingDirectory=/Aws_test/fundoo_notes/
# Execute the Gunicorn server with the specified number of workers and binding
ExecStart=/bin/bash -c "source /home/pushpa/myenv/bin/activate && /home/pushpa/myenv/bin/gunicorn --workers 3 --bind 0.0.0.0:8000 fundoo_notes.wsgi:application"

[Install]
WantedBy=multi-user.target

To show processes which are currently using port 8000 on your server
->sudo lsof -i :8000

To reload its configuration files, we made changes to service files

->sudo systemctl daemon-reload

Starts the fundoo.service

->sudo systemctl start fundoo.service

To start automatically every time the system boots

->sudo systemctl enable fundoo.service

To check current status of the fundoo.service

->sudo systemctl status fundoo.service

If the status of fundoo.service is active(running),then our application is running in the background and no need to start the server manually using
->python3 manage.py runserver 0.0.0.0:8000 (Because it is included in service file only)

Run in Background
nohup gunicorn --workers 3 fundoo_notes.wsgi:application --bind 10.0.2.142:8000 &

Manually we can check whether server is working or not
-> curl http://10.0.2.54:8000

Frontend Instance

->sudo apt update
->sudo apt install nginx

To check the version of nginx
-> nginx –v

Edit /etc/nginx/sites-available/fundoo.conf
->sudo nano /etc/ nginx/sites-available/fundoo.conf

```
server {
        listen 80;
        server_name _default;
location / {
        include proxy_params;
        proxy_pass http://10.0.2.142:8000;
          }
}
```

- Symbolic link to the sites-enabled directory so Nginx can read that file from sites-enabled.
- The configurations present in sites-available are not activated until they are linked to the sites-enabled directory.

->sudo ln -s /etc/nginx/sites-available/fundoo.conf /etc/nginx/sites-enabled/

To test the configuration for Nginx
->sudo nginx –t

If the configuration test is successful, restart Nginx to apply the changes:
->sudo systemctl restart nginx

To verify that Nginx is running without issues:
->sudo systemctl status nginx

Change the hostname for instances of Private ip address
->sudo hostnamectl set-hostname Frontend

Need to edit the /etc/hosts file
->sudo nano /etc/hosts

Add this line below 127.0.0.1 localhost
#127.0.0.1 Frontend