Deploy Django Backend on EC2 Instance

Architecture Components

- **Instance 1 (Database):** PostgreSQL on EC2 instance
- **Instance 2 (Backend):** Django application on EC2 instance

Configuring Instance 1 (Database)

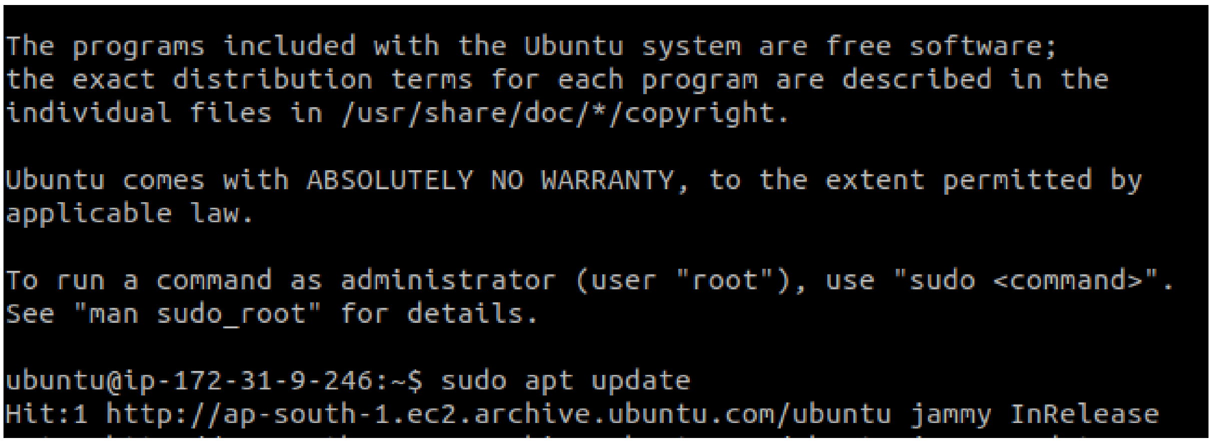
Connect to the Database EC2 Instance

ssh -i path_to_your_key.pem ubuntu@your_database_instance_public_ip

Note: Alternatively, you can use Putty

Update package index

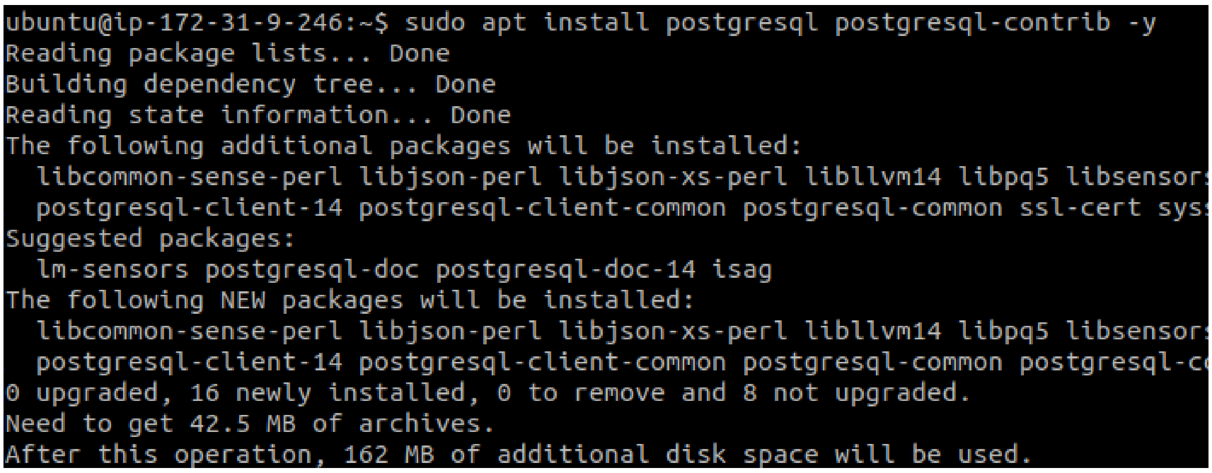
sudo apt update



```
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.

ubuntu@ip-172-31-9-246:~$ sudo apt update
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
```

Install PostgreSQL

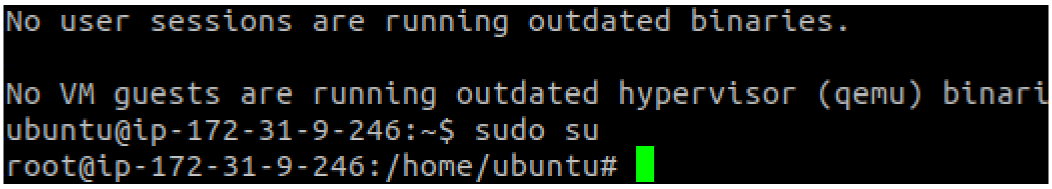
sudo apt update sudo apt install postgresql postgresql-contrib -y



```
ubuntu@ip-172-31-9-246:~$ sudo apt install postgresql postgresql-contrib -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libcommon-sense-perl libjson-perl libjson-xs-perl libllvm14 libpq5 libsensors
  postgresql-client-14 postgresql-client-common postgresql-common ssl-cert sys
Suggested packages:
  lm-sensors postgresql-doc postgresql-doc-14 isag
The following NEW packages will be installed:
  libcommon-sense-perl libjson-perl libjson-xs-perl libllvm14 libpq5 libsensors
  postgresql-client-14 postgresql-client-common postgresql-common postgresql-c
0 upgraded, 16 newly installed, 0 to remove and 8 not upgraded.
Need to get 42.5 MB of archives.
After this operation, 162 MB of additional disk space will be used.
```

Switch to root user

Sudo su

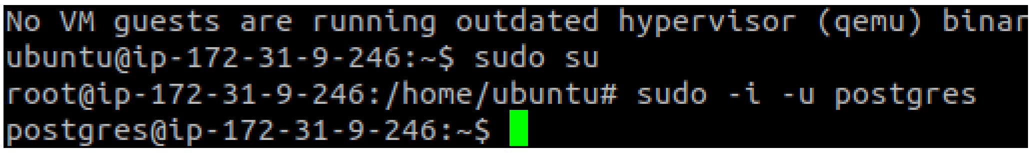


```
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binari
ubuntu@ip-172-31-9-246:~$ sudo su
root@ip-172-31-9-246:/home/ubuntu#
```

Create a new User

Creating a new user for postgresql with the name postgres

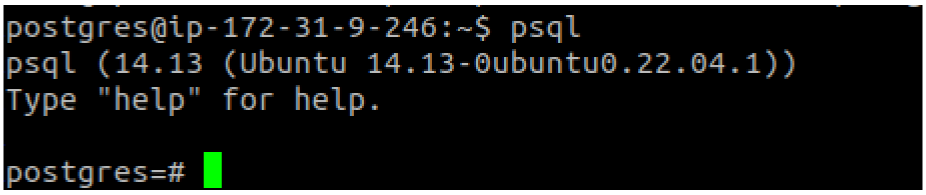
sudo -i -u postgres



```
No VM guests are running outdated hypervisor (qemu) binar
ubuntu@ip-172-31-9-246:~$ sudo su
root@ip-172-31-9-246:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-9-246:~$
```

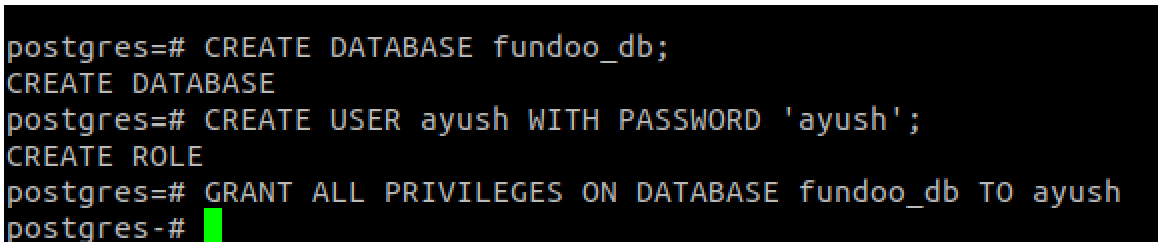
Access the Postgresql

psql

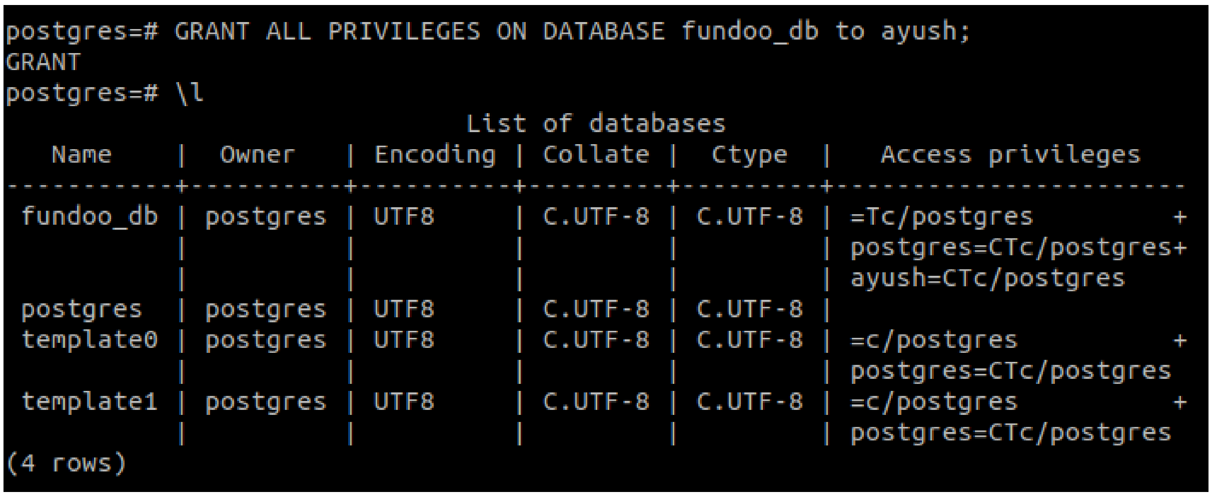


```
postgres@ip-172-31-9-246:~$ psql
psql (14.13 (Ubuntu 14.13-0ubuntu0.22.04.1))
Type "help" for help.

postgres=#
```

Create Database, User and Grant Privileges



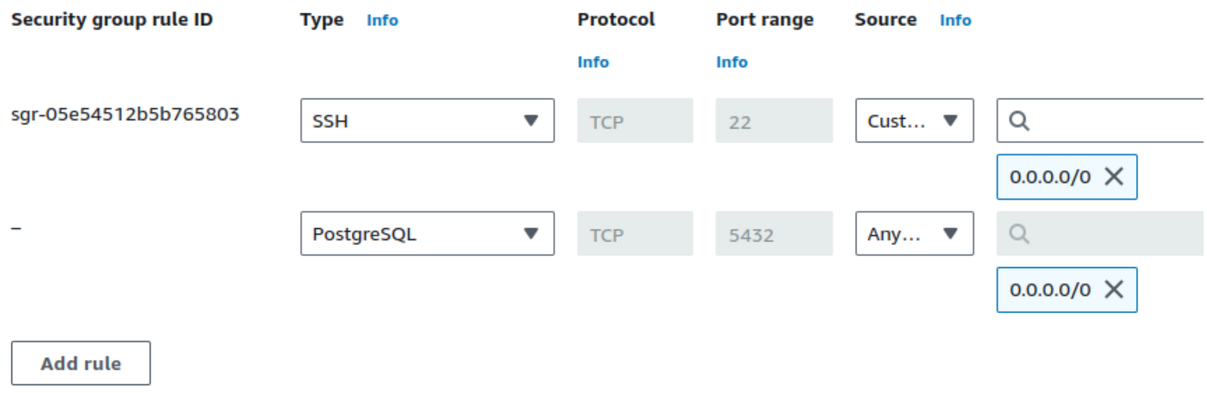
```
postgres=# CREATE DATABASE fundoo_db;
CREATE DATABASE
postgres=# CREATE USER ayush WITH PASSWORD 'ayush';
CREATE ROLE
postgres=# GRANT ALL PRIVILEGES ON DATABASE fundoo_db TO ayush
postgres-#
```



```
postgres=# GRANT ALL PRIVILEGES ON DATABASE fundoo_db to ayush;
GRANT
postgres=# \l
                              List of databases
   Name    |  Owner   | Encoding | Collate  |  Ctype   |   Access privileges
-----------+----------+----------+----------+----------+-----------------------
 fundoo_db | postgres | UTF8     | C.UTF-8  | C.UTF-8  | =Tc/postgres         +
           |          |          |          |          | postgres=CTc/postgres+
           |          |          |          |          | ayush=CTc/postgres
 postgres  | postgres | UTF8     | C.UTF-8  | C.UTF-8  |
 template0 | postgres | UTF8     | C.UTF-8  | C.UTF-8  | =c/postgres          +
           |          |          |          |          | postgres=CTc/postgres
 template1 | postgres | UTF8     | C.UTF-8  | C.UTF-8  | =c/postgres          +
           |          |          |          |          | postgres=CTc/postgres
(4 rows)
```

Configure EC2 Security Group

Open the TCP port Postgresql which is the port 5432

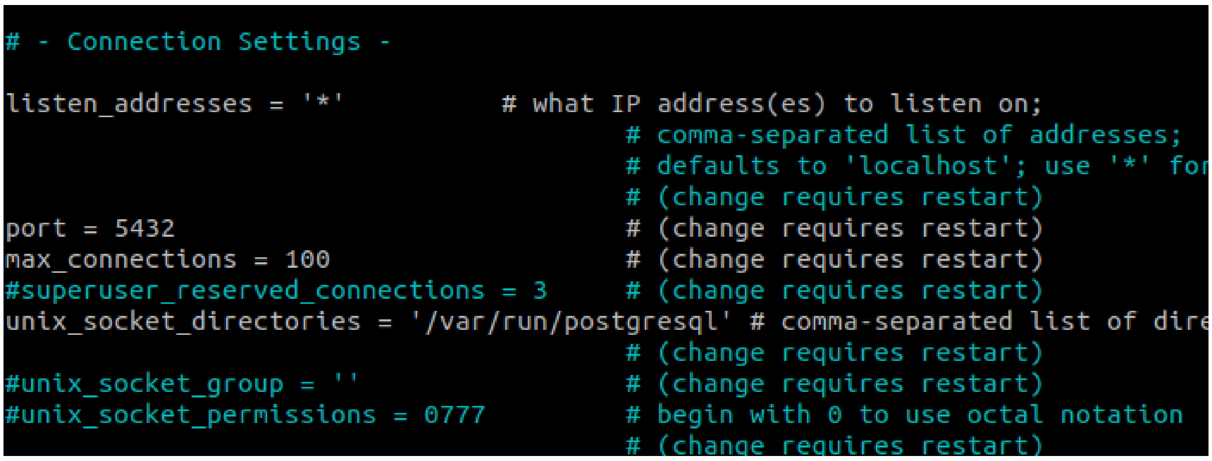


Security group rule ID	Type	Protocol	Port range	Source	
sgr-05e54512b5b765803	SSH	TCP	22	Cust...	0.0.0.0/0
–	PostgreSQL	TCP	5432	Any...	0.0.0.0/0

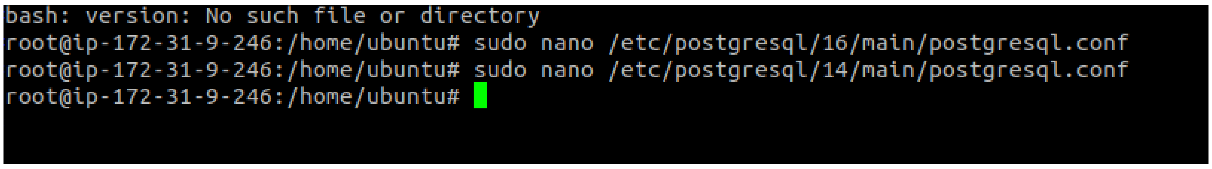
Configure postgresql.conf

sudo nano /etc/postgresql/14/main/postgresql.conf

By default, PostgreSQL listens on localhost only. To allow remote connections, Find the line with `listen_addresses` and change it to listen_addresses = '*'



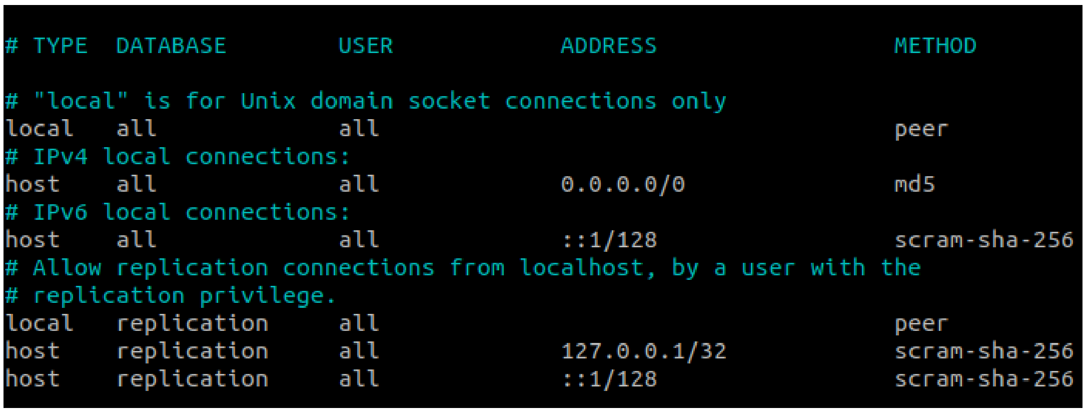
```
# - Connection Settings -

listen_addresses = '*'          # what IP address(es) to listen on;
                                        # comma-separated list of addresses;
                                        # defaults to 'localhost'; use '*' for
                                        # (change requires restart)
port = 5432                             # (change requires restart)
max_connections = 100                   # (change requires restart)
#superuser_reserved_connections = 3     # (change requires restart)
unix_socket_directories = '/var/run/postgresql' # comma-separated list of dire
                                        # (change requires restart)
#unix_socket_group = ''                 # (change requires restart)
#unix_socket_permissions = 0777         # begin with 0 to use octal notation
                                        # (change requires restart)
```



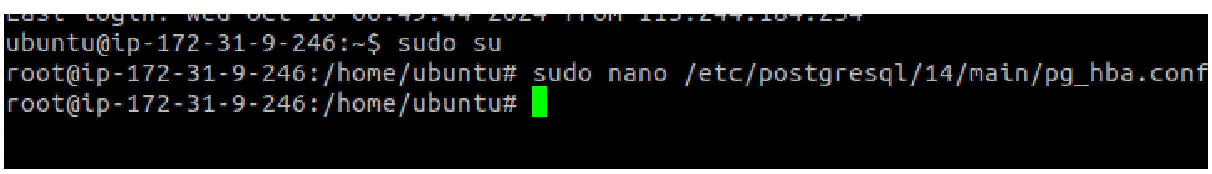
```
bash: version: No such file or directory
root@ip-172-31-9-246:/home/ubuntu# sudo nano /etc/postgresql/16/main/postgresql.conf
root@ip-172-31-9-246:/home/ubuntu# sudo nano /etc/postgresql/14/main/postgresql.conf
root@ip-172-31-9-246:/home/ubuntu#
```

Configure pg_hba.conf

sudo nano /etc/postgresql/14/main/pg_hba.conf
Add the following line at the end of the file to allow connections from any IP:
host   all   all   0.0.0.0/0   md5



```
# TYPE  DATABASE        USER            ADDRESS                 METHOD

# "local" is for Unix domain socket connections only
local   all             all                                     peer
# IPv4 local connections:
host    all             all             0.0.0.0/0               md5
# IPv6 local connections:
host    all             all             ::1/128                 scram-sha-256
# Allow replication connections from localhost, by a user with the
# replication privilege.
local   replication     all                                     peer
host    replication     all             127.0.0.1/32            scram-sha-256
host    replication     all             ::1/128                 scram-sha-256
```



```
ubuntu@ip-172-31-9-246:~$ sudo su
root@ip-172-31-9-246:/home/ubuntu# sudo nano /etc/postgresql/14/main/pg_hba.conf
root@ip-172-31-9-246:/home/ubuntu#
```

Enable PostgreSQL to start on boot

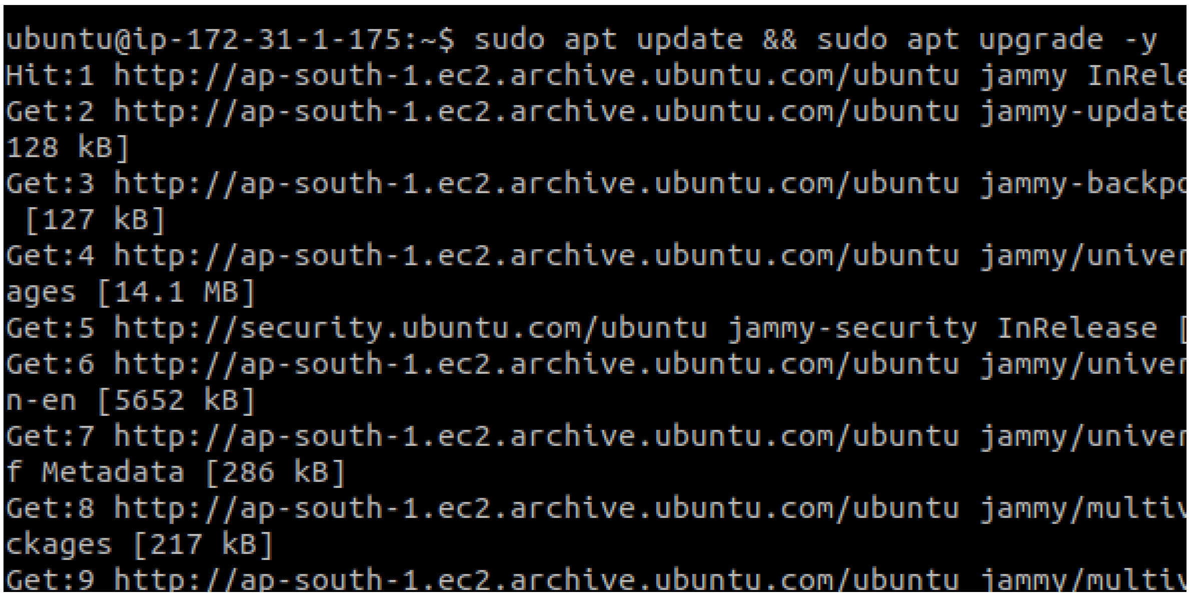
To Enable PostgreSQL to run on ec2 instance startup

```
sudo systemctl enable postgresql
```

Configuring Instance 2 (Backend)

Update package index

sudo apt update && sudo apt upgrade -y

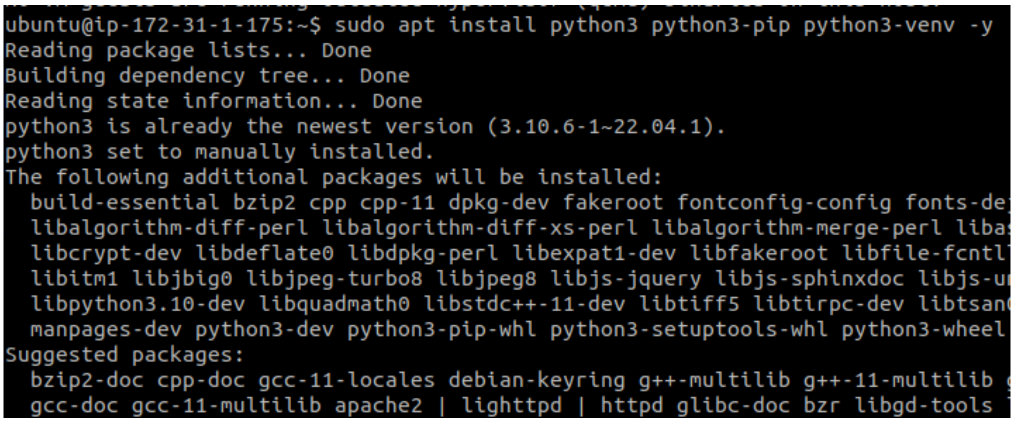


```
ubuntu@ip-172-31-1-175:~$ sudo apt update && sudo apt upgrade -y
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy InRele
Get:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-update
128 kB]
Get:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backpo
 [127 kB]
Get:4 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/univer
ages [14.1 MB]
Get:5 http://security.ubuntu.com/ubuntu jammy-security InRelease [
Get:6 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/univer
n-en [5652 kB]
Get:7 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/univer
f Metadata [286 kB]
Get:8 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/multiv
ckages [217 kB]
Get:9 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/multiv
```

Install Python and pip

Django requires Python, so install Python and pip (Python's package installer)

```
sudo apt install python3 python3-pip python3-venv -y
```

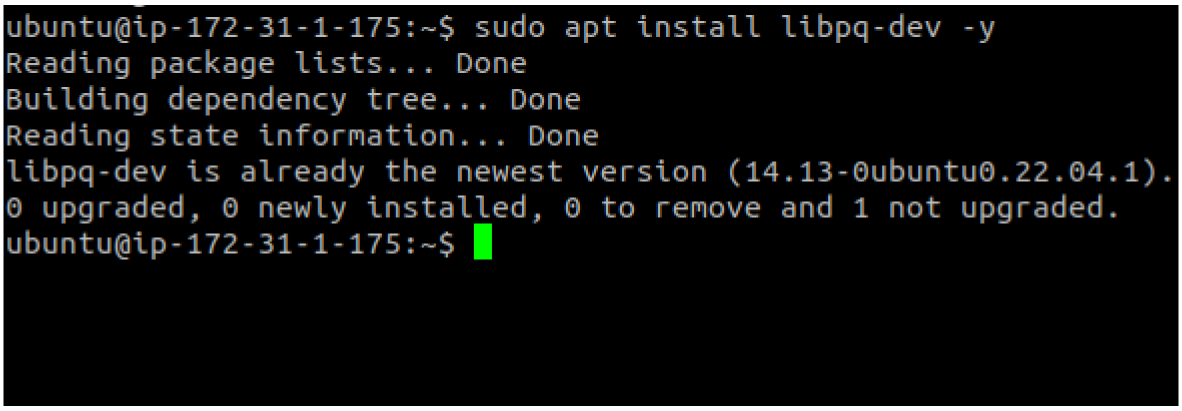


```
ubuntu@ip-172-31-1-175:~$ sudo apt install python3 python3-pip python3-venv -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
python3 is already the newest version (3.10.6-1~22.04.1).
python3 set to manually installed.
The following additional packages will be installed:
  build-essential bzip2 cpp cpp-11 dpkg-dev fakeroot fontconfig-config fonts-de
  libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl liba
  libcrypt-dev libdeflate0 libdpkg-perl libexpat1-dev libfakeroot libfile-fcntl
  libitm1 libjbig0 libjpeg-turbo8 libjpeg8 libjs-jquery libjs-sphinxdoc libjs-u
  libpython3.10-dev libquadmath0 libstdc++-11-dev libtiff5 libtirpc-dev libtsan
  manpages-dev python3-dev python3-pip-whl python3-setuptools-whl python3-wheel
Suggested packages:
  bzip2-doc cpp-doc gcc-11-locales debian-keyring g++-multilib g++-11-multilib
  gcc-doc gcc-11-multilib apache2 | lighttpd | httpd glibc-doc bzr libgd-tools
```

Install PostgreSQL Development Libraries

Install PostgreSQL development headers and libraries (necessary for connecting Django to PostgreSQL)

```
sudo apt install libpq-dev -y
```

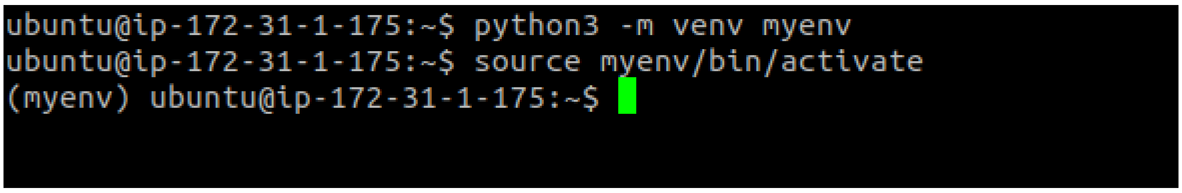


```
ubuntu@ip-172-31-1-175:~$ sudo apt install libpq-dev -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
libpq-dev is already the newest version (14.13-0ubuntu0.22.04.1).
0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
ubuntu@ip-172-31-1-175:~$
```

Set Up a Python Virtual Environment

It’s best practice to use a virtual environment for your Django app to manage dependencies

```
python3 -m venv myenv
source myenv/bin/activate
```

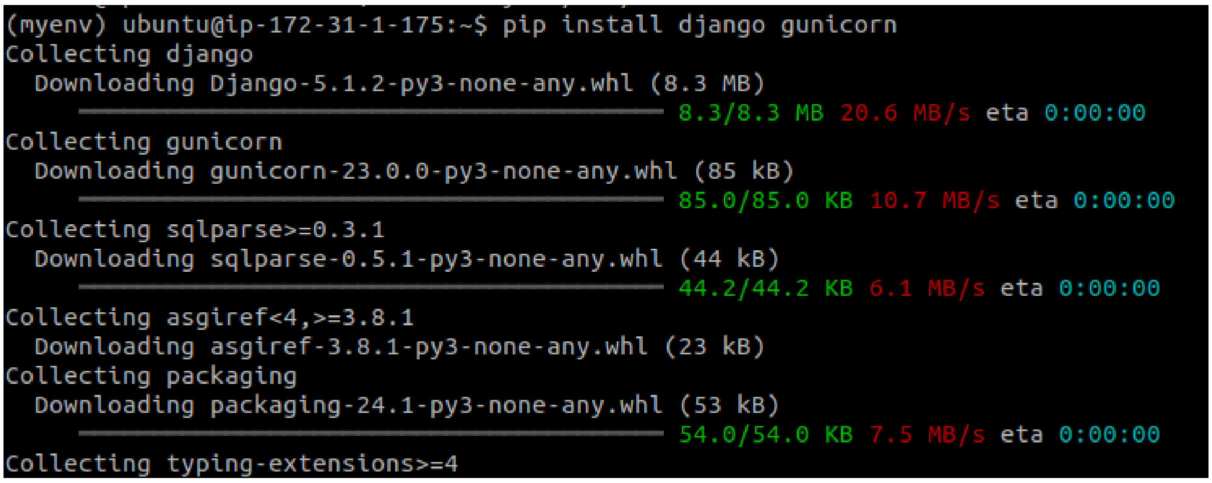


```
ubuntu@ip-172-31-1-175:~$ python3 -m venv myenv
ubuntu@ip-172-31-1-175:~$ source myenv/bin/activate
(myenv) ubuntu@ip-172-31-1-175:~$
```

Install Django and Gunicorn

Install Django and Gunicorn (the production WSGI server)

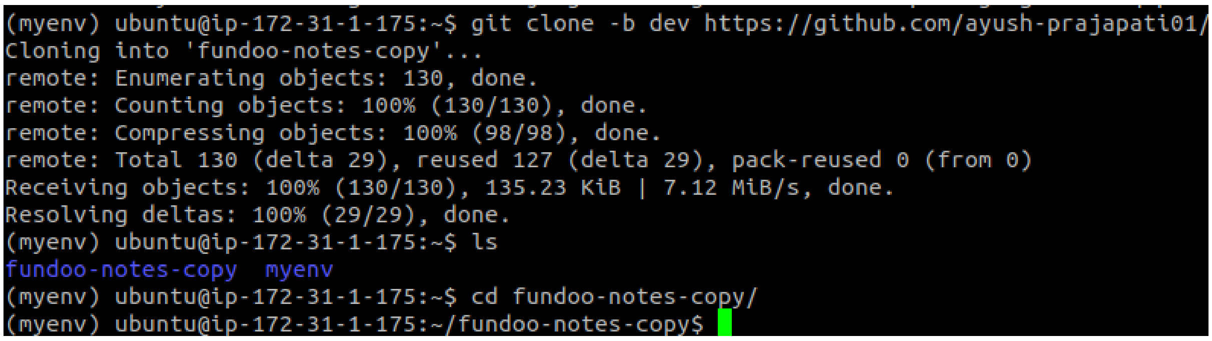
```
pip install django gunicorn
```



```
(myenv) ubuntu@ip-172-31-1-175:~$ pip install django gunicorn
Collecting django
  Downloading Django-5.1.2-py3-none-any.whl (8.3 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.3/8.3 MB 20.6 MB/s eta 0:00:00
Collecting gunicorn
  Downloading gunicorn-23.0.0-py3-none-any.whl (85 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 85.0/85.0 KB 10.7 MB/s eta 0:00:00
Collecting sqlparse>=0.3.1
  Downloading sqlparse-0.5.1-py3-none-any.whl (44 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 44.2/44.2 KB 6.1 MB/s eta 0:00:00
Collecting asgiref<4,>=3.8.1
  Downloading asgiref-3.8.1-py3-none-any.whl (23 kB)
Collecting packaging
  Downloading packaging-24.1-py3-none-any.whl (53 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 54.0/54.0 KB 7.5 MB/s eta 0:00:00
Collecting typing-extensions>=4
```

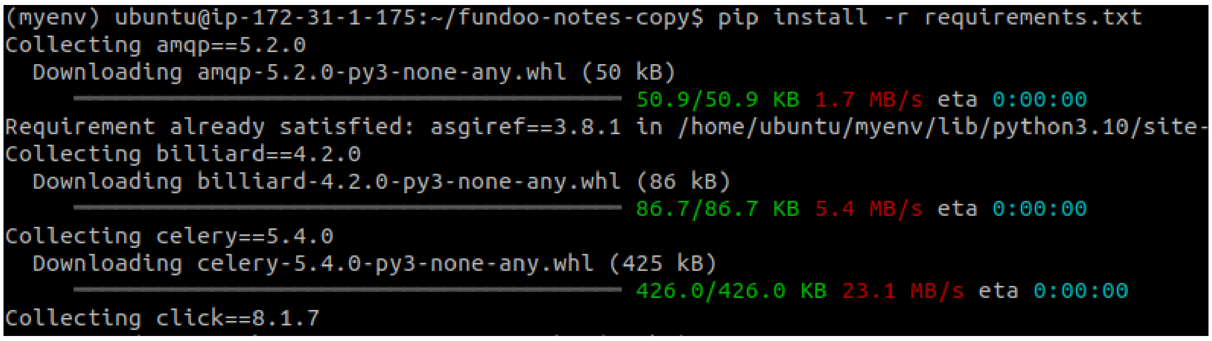
Clone the Django project from Github

git clone -b <branch-name> <repo-link>



```
(myenv) ubuntu@ip-172-31-1-175:~$ git clone -b dev https://github.com/ayush-prajapati01/
Cloning into 'fundoo-notes-copy'...
remote: Enumerating objects: 130, done.
remote: Counting objects: 100% (130/130), done.
remote: Compressing objects: 100% (98/98), done.
remote: Total 130 (delta 29), reused 127 (delta 29), pack-reused 0 (from 0)
Receiving objects: 100% (130/130), 135.23 KiB | 7.12 MiB/s, done.
Resolving deltas: 100% (29/29), done.
(myenv) ubuntu@ip-172-31-1-175:~$ ls
fundoo-notes-copy  myenv
(myenv) ubuntu@ip-172-31-1-175:~$ cd fundoo-notes-copy/
(myenv) ubuntu@ip-172-31-1-175:~/fundoo-notes-copy$
```

Install requirements.txt

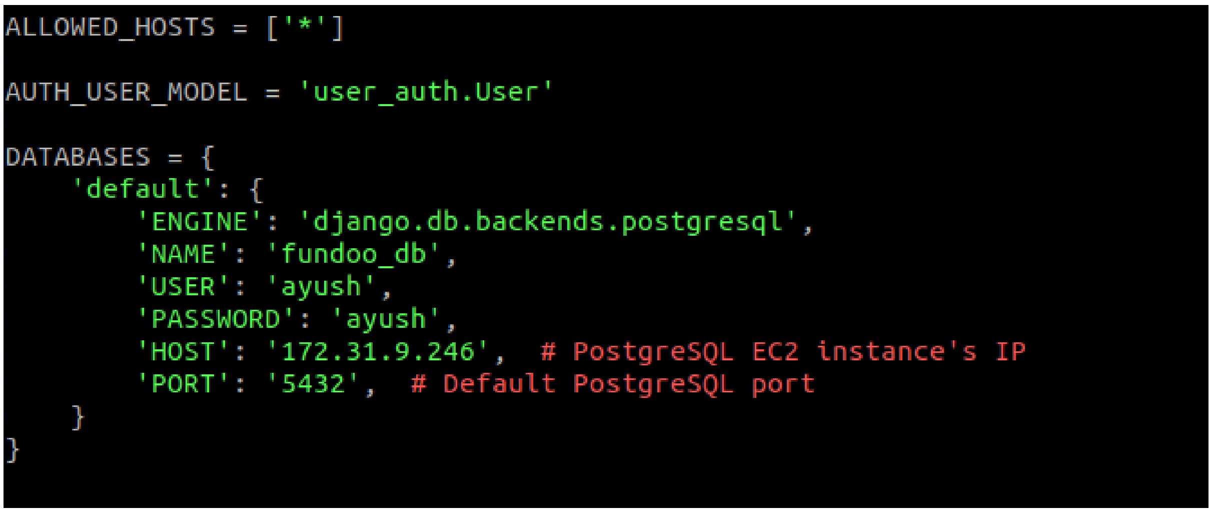


```
(myenv) ubuntu@ip-172-31-1-175:~/fundoo-notes-copy$ pip install -r requirements.txt
Collecting amqp==5.2.0
  Downloading amqp-5.2.0-py3-none-any.whl (50 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 50.9/50.9 KB 1.7 MB/s eta 0:00:00
Requirement already satisfied: asgiref==3.8.1 in /home/ubuntu/myenv/lib/python3.10/site-
Collecting billiard==4.2.0
  Downloading billiard-4.2.0-py3-none-any.whl (86 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 86.7/86.7 KB 5.4 MB/s eta 0:00:00
Collecting celery==5.4.0
  Downloading celery-5.4.0-py3-none-any.whl (425 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 426.0/426.0 KB 23.1 MB/s eta 0:00:00
Collecting click==8.1.7
```

Configure PostgreSQL in Django Settings

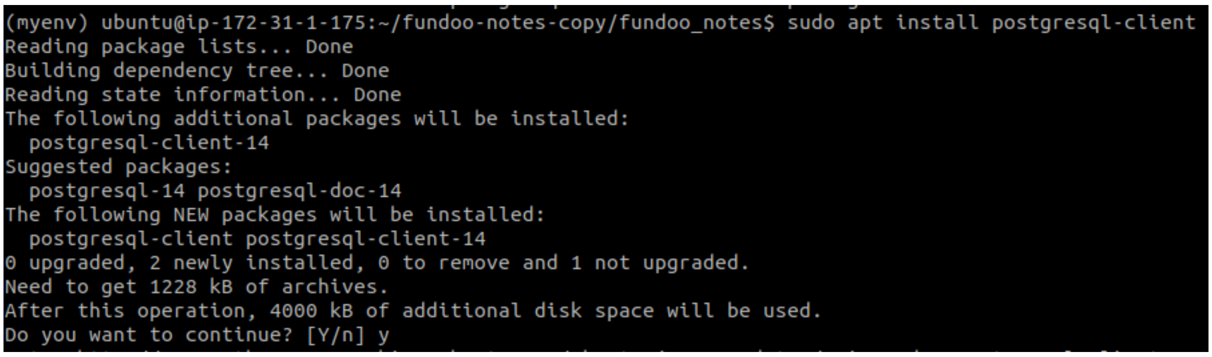
(myenv) ubuntu@ip-172-31-1-175:~/fundoo-notes-copy/fundoo_notes/fundoo_notes$ nano settings.py

Allow all host and Change databases settings



```
ALLOWED_HOSTS = ['*']

AUTH_USER_MODEL = 'user_auth.User'

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'fundoo_db',
        'USER': 'ayush',
        'PASSWORD': 'ayush',
        'HOST': '172.31.9.246',  # PostgreSQL EC2 instance's IP
        'PORT': '5432',  # Default PostgreSQL port
    }
}
```

Install Postgresql Client

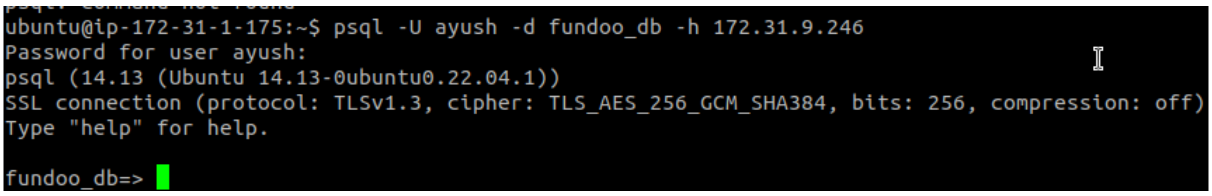


```
(myenv) ubuntu@ip-172-31-1-175:~/fundoo-notes-copy/fundoo_notes$ sudo apt install postgresql-client
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  postgresql-client-14
Suggested packages:
  postgresql-14 postgresql-doc-14
The following NEW packages will be installed:
  postgresql-client postgresql-client-14
0 upgraded, 2 newly installed, 0 to remove and 1 not upgraded.
Need to get 1228 kB of archives.
After this operation, 4000 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

Test the Connection with Database

Test the database connection with the following command

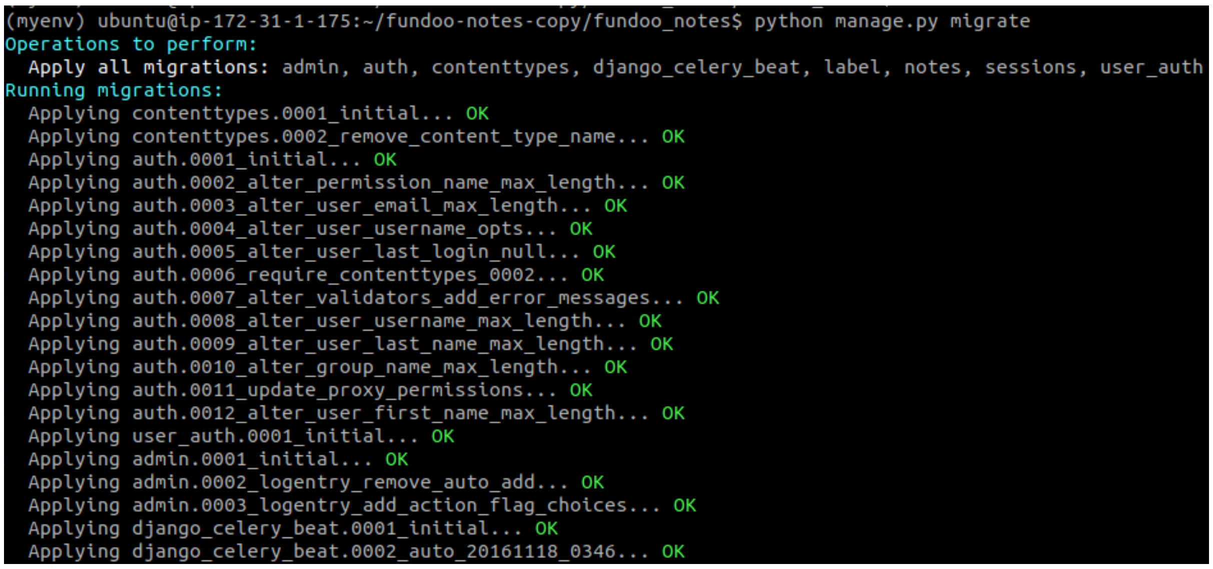
psql -U ayush -d fundoo_db -h 172.31.9.246



```
ubuntu@ip-172-31-1-175:~$ psql -U ayush -d fundoo_db -h 172.31.9.246
Password for user ayush:
psql (14.13 (Ubuntu 14.13-0ubuntu0.22.04.1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
Type "help" for help.

fundoo_db=>
```

Migrate the Database

python manage.py migrate



```
(myenv) ubuntu@ip-172-31-1-175:~/fundoo-notes-copy/fundoo_notes$ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, django_celery_beat, label, notes, sessions, user_auth
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0001_initial... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying user_auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying django_celery_beat.0001_initial... OK
  Applying django_celery_beat.0002_auto_20161118_0346... OK
```

Run Django Locally to Test

```
python manage.py runserver 0.0.0.0:8000
```

Configure the daemon service file

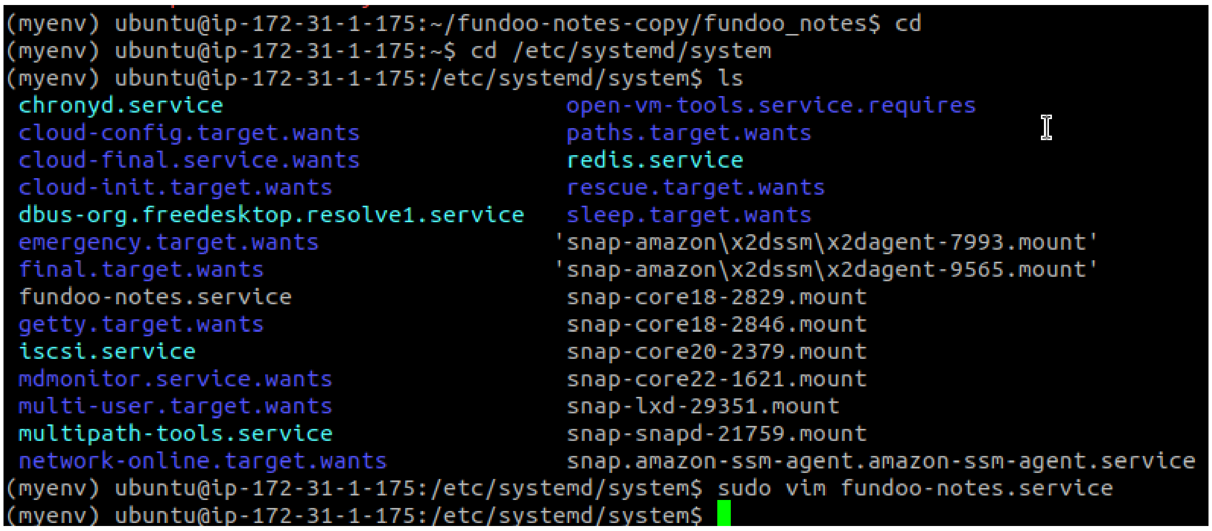
We will create a service file so that the django app can run in the background

Create a Service File:
The service files are usually located in `/etc/systemd/system/`. You'll create your custom service file there.

```
sudo nano /etc/systemd/system/<name>.service
```

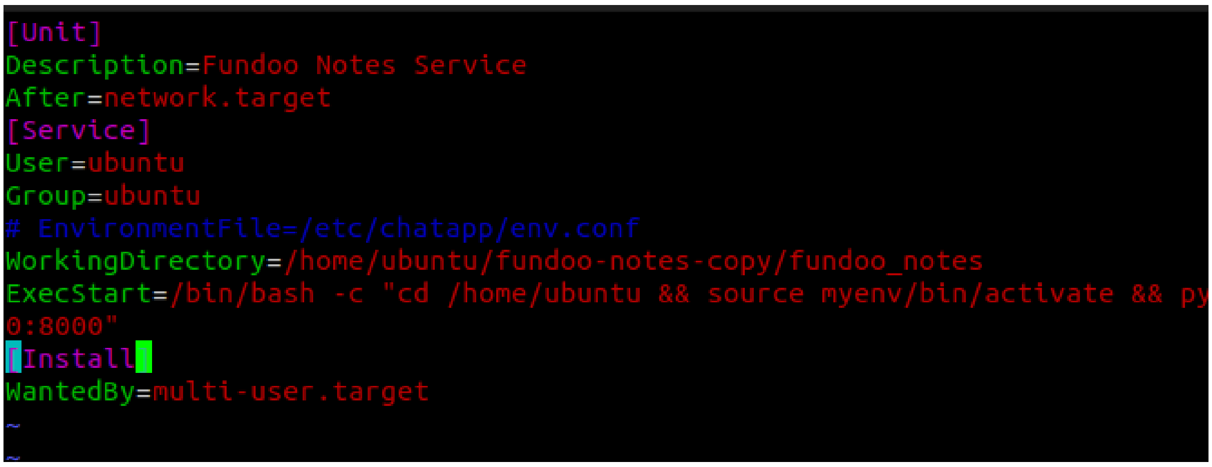
Define the Service Configuration

```
sudo vim fundoo-service.service
```



```
(myenv) ubuntu@ip-172-31-1-175:~/fundoo-notes-copy/fundoo_notes$ cd
(myenv) ubuntu@ip-172-31-1-175:~$ cd /etc/systemd/system
(myenv) ubuntu@ip-172-31-1-175:/etc/systemd/system$ ls
 chronyd.service                        open-vm-tools.service.requires
 cloud-config.target.wants              paths.target.wants
 cloud-final.service.wants              redis.service
 cloud-init.target.wants                rescue.target.wants
 dbus-org.freedesktop.resolve1.service  sleep.target.wants
 emergency.target.wants                'snap-amazon\x2dssm\x2dagent-7993.mount'
 final.target.wants                    'snap-amazon\x2dssm\x2dagent-9565.mount'
 fundoo-notes.service                   snap-core18-2829.mount
 getty.target.wants                     snap-core18-2846.mount
 iscsi.service                          snap-core20-2379.mount
 mdmonitor.service.wants                snap-core22-1621.mount
 multi-user.target.wants                snap-lxd-29351.mount
 multipath-tools.service                snap-snapd-21759.mount
 network-online.target.wants            snap.amazon-ssm-agent.amazon-ssm-agent.service
(myenv) ubuntu@ip-172-31-1-175:/etc/systemd/system$ sudo vim fundoo-notes.service
(myenv) ubuntu@ip-172-31-1-175:/etc/systemd/system$
```

Description: A short description of your service.
After: Defines when the service should start, such as after the network is up.
User: The user that will run the service (typically your system user).
Group: The group for file permissions.
WorkingDirectory: The location where your project files reside.
ExecStart: The command to start your application (in this case, Gunicorn).
Restart=always: Automatically restarts the service if it crashes.
Environment: Use to define environment variables like Django settings.



```
[Unit]
Description=Fundoo Notes Service
After=network.target
[Service]
User=ubuntu
Group=ubuntu
# EnvironmentFile=/etc/chatapp/env.conf
WorkingDirectory=/home/ubuntu/fundoo-notes-copy/fundoo_notes
ExecStart=/bin/bash -c "cd /home/ubuntu && source myenv/bin/activate && py
0:8000"
[Install]
WantedBy=multi-user.target
~
~
```

Reload the systemd Daemon

After creating the service file, reload systemd to recognize the new service.

```
sudo systemctl daemon-reload
```

Start the Service

sudo systemctl start fundoo-service

Enable the Service to Start on Boot

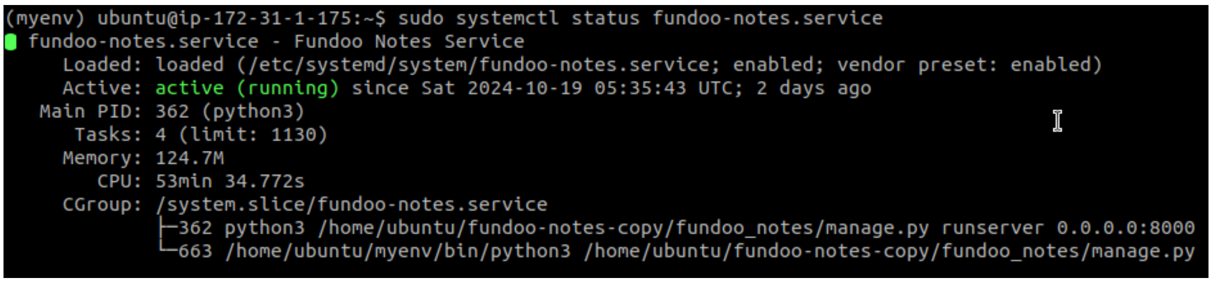
To ensure the service starts automatically at boot

sudo systemctl enable fundoo-service

Check the Status of the Service

Verify that the service is running correctly

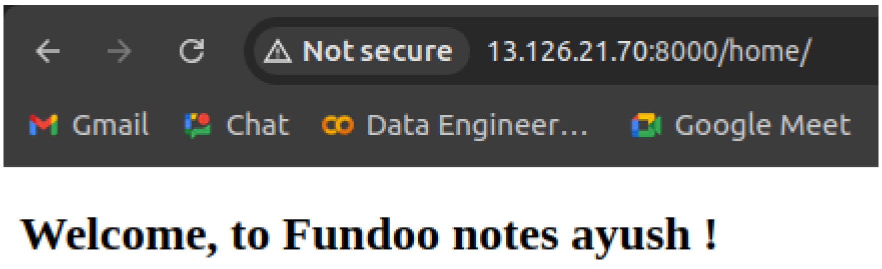
sudo systemctl status fundoo-service



```
(myenv) ubuntu@ip-172-31-1-175:~$ sudo systemctl status fundoo-notes.service
● fundoo-notes.service - Fundoo Notes Service
     Loaded: loaded (/etc/systemd/system/fundoo-notes.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2024-10-19 05:35:43 UTC; 2 days ago
   Main PID: 362 (python3)
      Tasks: 4 (limit: 1130)
     Memory: 124.7M
        CPU: 53min 34.772s
     CGroup: /system.slice/fundoo-notes.service
             ├─362 python3 /home/ubuntu/fundoo-notes-copy/fundoo_notes/manage.py runserver 0.0.0.0:8000
             └─663 /home/ubuntu/myenv/bin/python3 /home/ubuntu/fundoo-notes-copy/fundoo_notes/manage.py
```

Verify Deployment

Once the setup is complete, verify that your Django application is running correctly by accessing it via its public IP address or domain name.

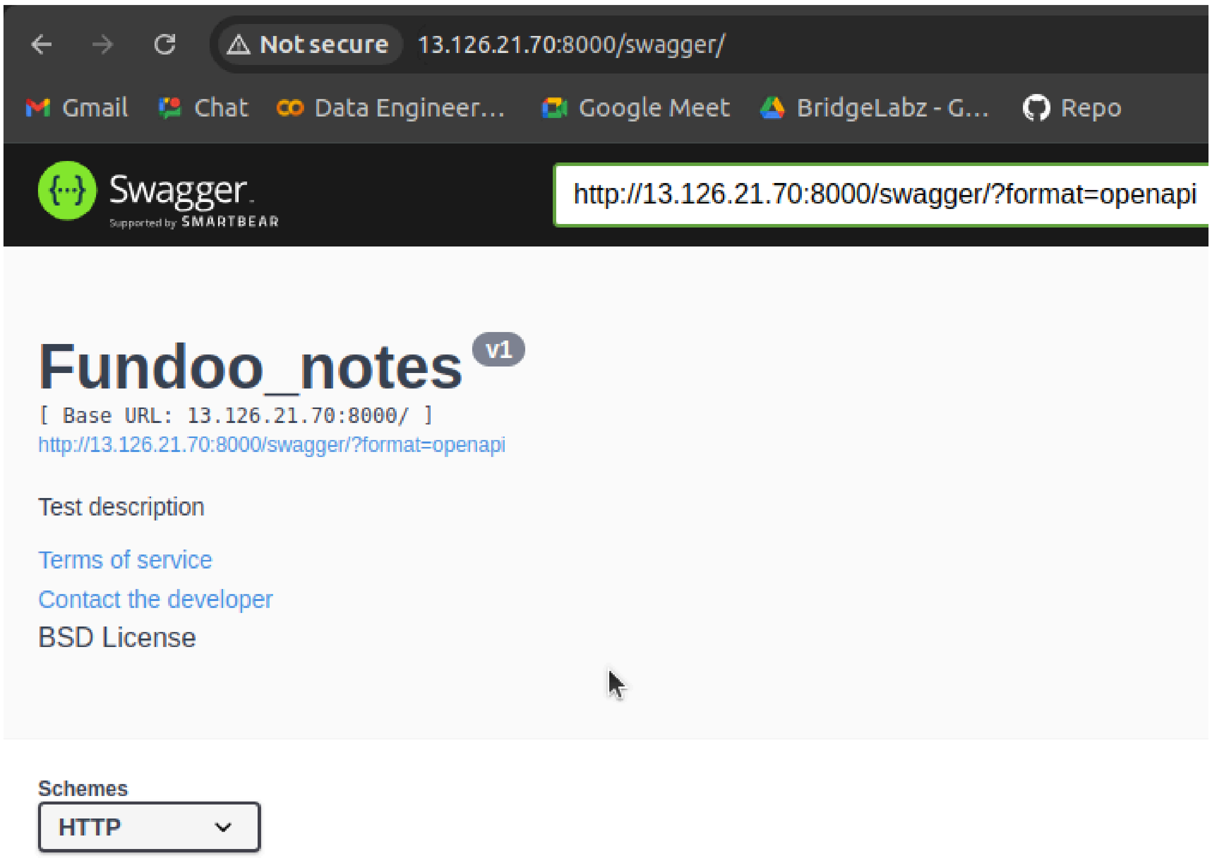


Not secure 13.126.21.70:8000/home/

Welcome, to Fundoo notes ayush !

Perform API testing

We can perform api testing using swagger to confirm our applications is running perfectly



http://13.126.21.70:8000/swagger/?format=openapi

Fundoo_notes v1

[Base URL: 13.126.21.70:8000/]

http://13.126.21.70:8000/swagger/?format=openapi

Test description

Terms of service

Contact the developer

BSD License

Schemes: HTTP