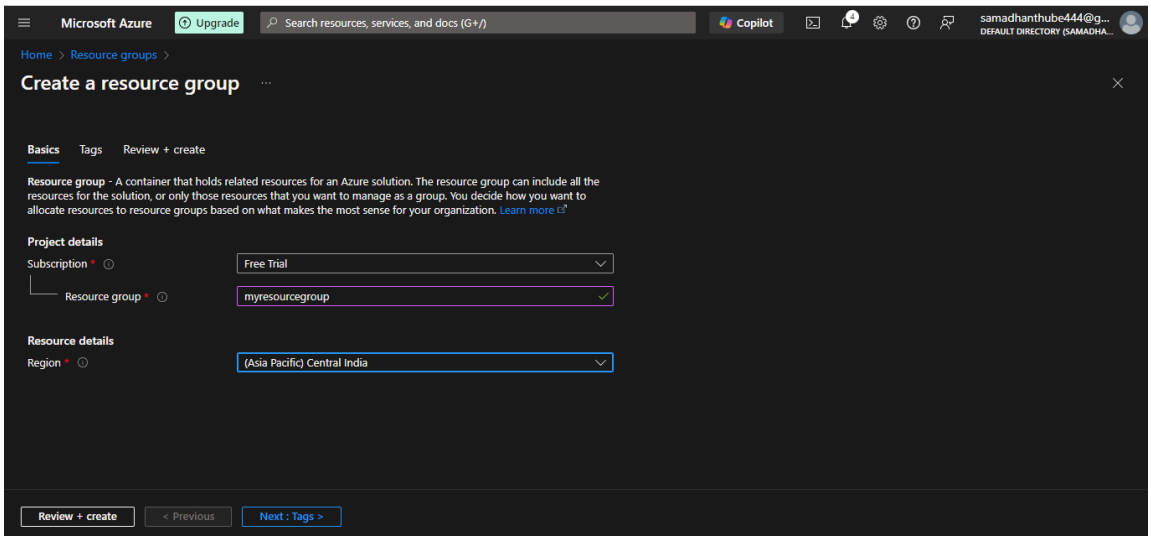
Create an Azure VM for linux(ubuntu) using Azure Portal

Step 1: Log in to the Azure Portal

1. **Open a Web Browser**: Navigate to the Azure Portal.
2. **Sign In**: Log in with your Azure account credentials.

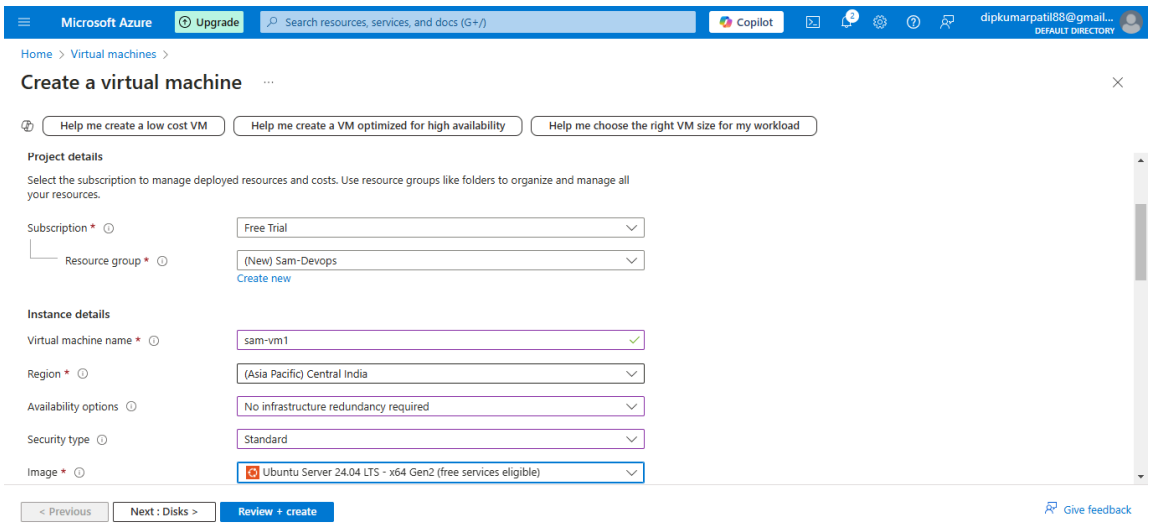
Step 2: Create a Resource Group

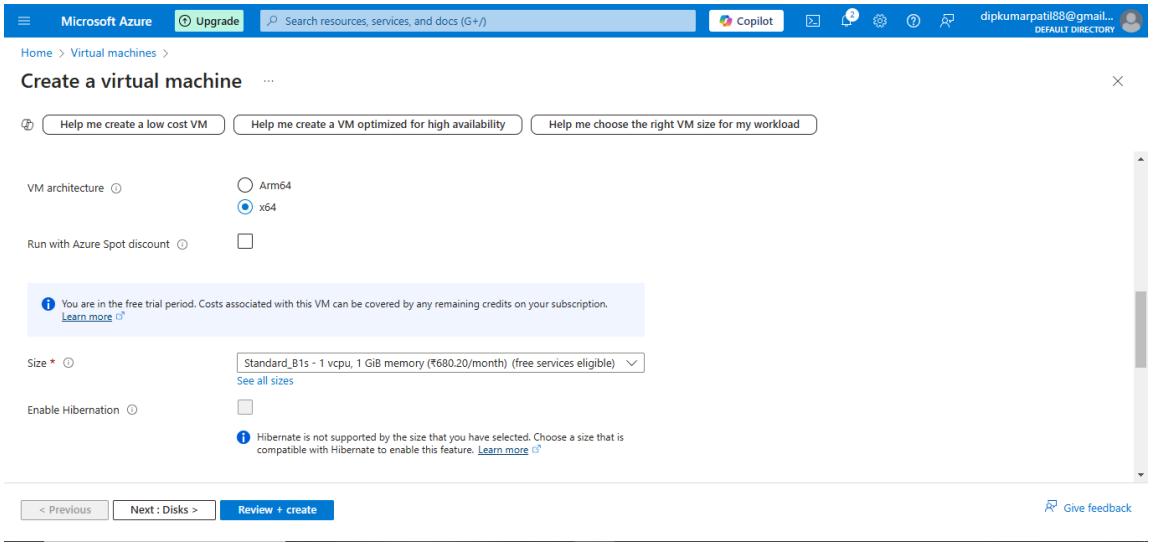
1. **Navigate to Resource Groups**: In the Azure Portal, click on "Resource groups" in the left-hand menu or use the search bar at the top.
2. **Create Resource Group**: Click the "Create" button.
3. **Fill Out the Form**:
   - **Subscription**: Select your Azure subscription.
   - **Resource group**: Enter a name for the resource group (e.g., myResourceGroup).
   - **Region**: Choose the region where you want the resource group to be located.
4. **Review + Create**: Click "Review + create," then "Create."



Step 3: Create the Virtual Machine

1. **Navigate to Virtual Machines**: In the Azure Portal, click on "Virtual machines" in the left-hand menu.
2. **Create a Virtual Machine**: Click the "Create" button and select "Virtual machine" from the dropdown menu.
3. **Configure Basics**:





Microsoft Azure | Upgrade | Search resources, services, and docs (G+/) | Copilot | dipkumarpatil88@gmail... DEFAULT DIRECTORY

Home > Virtual machines >

Create a virtual machine

Help me create a low cost VM | Help me create a VM optimized for high availability | Help me choose the right VM size for my workload

Administrator account

Authentication type: SSH public key / Password

Username *: samthube

Password *: ••••••••••••

Confirm password *: ••••••••••••

Inbound port rules

Select which virtual machine network ports are accessible from the public internet. You can specify more limited or granular network access on the Networking tab.

Public inbound ports *: None / Allow selected ports

Select inbound ports *: SSH (22)

< Previous | Next : Disks > | Review + create | Give feedback

Microsoft Azure | Upgrade | Search resources, services, and docs (G+/) | Copilot | dipkumarpatil88@gmail... DEFAULT DIRECTORY

Home > Virtual machines >

Create a virtual machine

Help me create a low cost VM | Help me create a VM optimized for high availability | Help me choose the right VM size for my workload

OS disk size: Image default (30 GiB)

OS disk type *: Standard HDD (locally-redundant storage)

The selected VM size supports premium disks. We recommend Premium SSD for high IOPS workloads. Virtual machines with Premium SSD disks qualify for the 99.9% connectivity SLA.

Delete with VM: ☑

Key management: Platform-managed key

Enable Ultra Disk compatibility: ☐

Data disks for sam-vm1

You can add and configure additional data disks for your virtual machine or attach existing disks. This VM also comes with a temporary disk.

LUN	Name	Size (GiB)	Disk type	Host caching	Delete with VM

Create and attach a new disk | Attach an existing disk

< Previous | Next : Networking > | Review + create | Give feedback

Microsoft Azure | Upgrade | Search resources, services, and docs (G+/) | Copilot | dipkumarpatil88@gmail... DEFAULT DIRECTORY

Home > Virtual machines >

Create a virtual machine

Help me create a low cost VM | Help me create a VM optimized for high availability | Help me choose the right VM size for my workload

Network interface

When creating a virtual machine, a network interface will be created for you.

Virtual network *: (new) sam-vm1-vnet
Create new

Subnet *: (new) default (10.0.0.0/24)

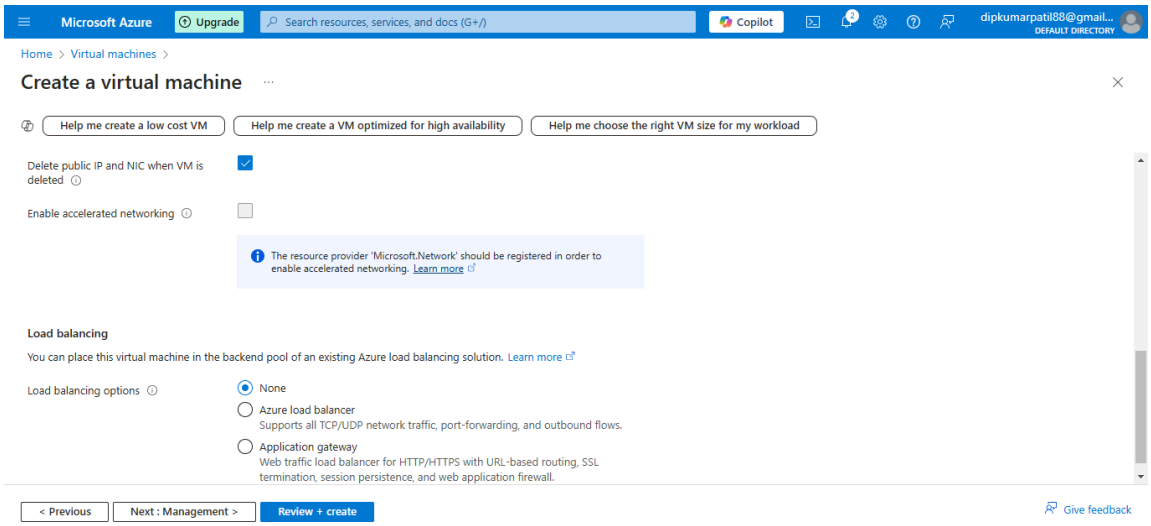
Public IP: (new) sam-vm1-ip
Create new

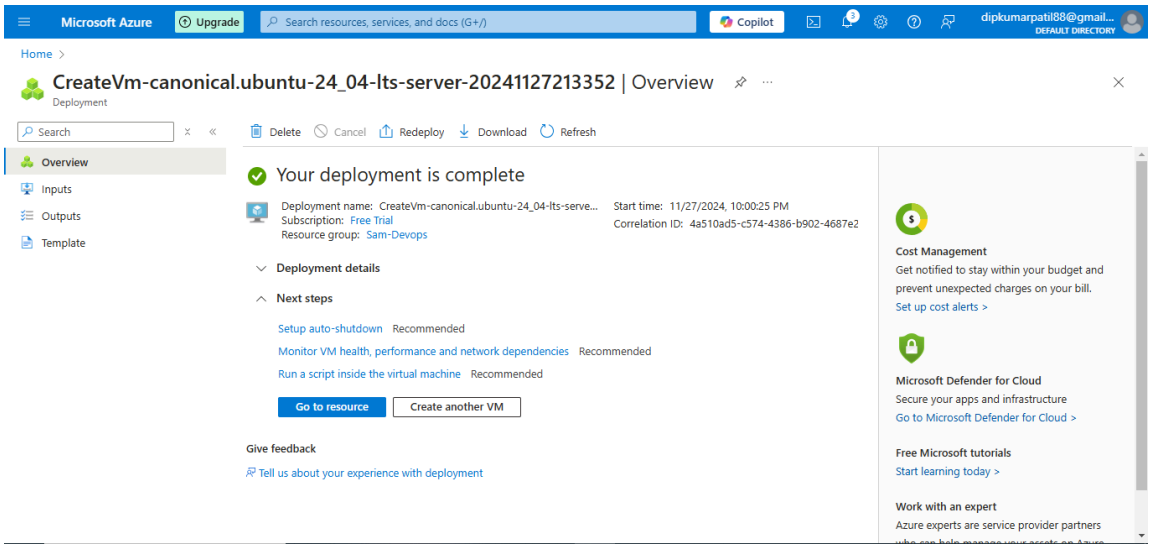
NIC network security group: None / Basic / Advanced

Public inbound ports *: None / Allow selected ports

Select inbound ports *: SSH (22)

< Previous | Next : Management > | Review + create | Give feedback

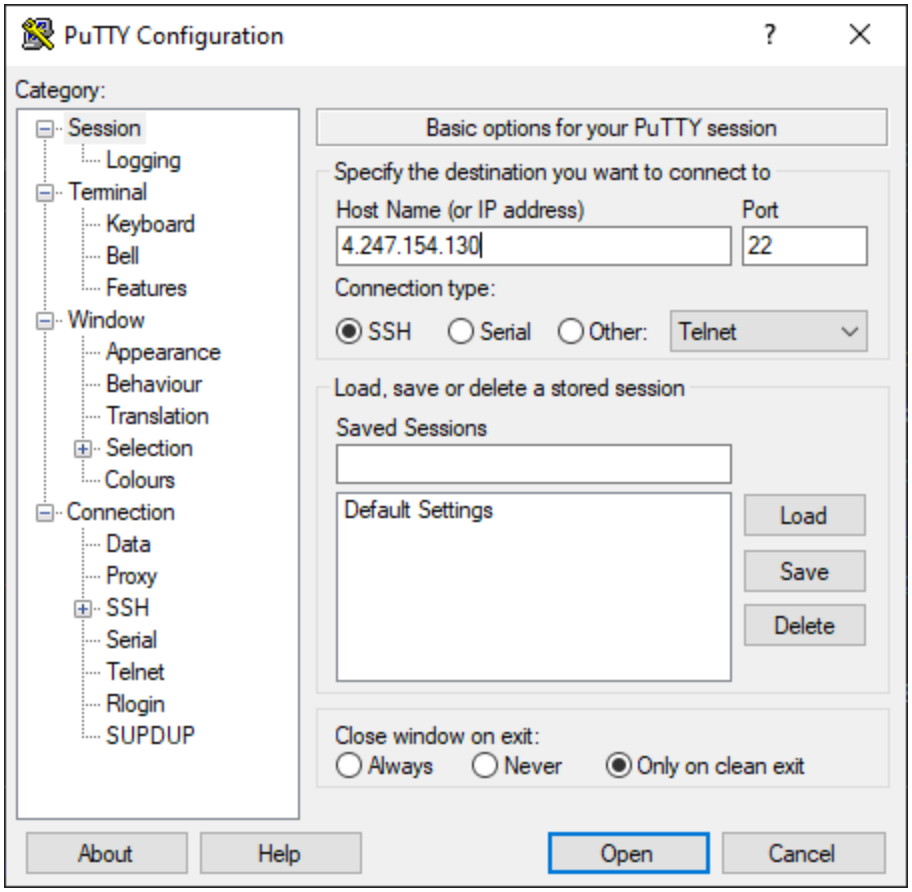




Step 4: Access the Virtual Machine

1. **Install Putty** : Install Putty using https://the.earth.li/~sgtatham/putty/latest/w64/putty-64bit-0.81-installer.msi

2. Enter the host name or IP address with connection type SSH and then enter the credential to connect and access the virtual machine


PuTTY Configuration
Category:
Session
Logging
Terminal
Keyboard
Bell
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
SSH
Serial
Telnet
Rlogin
SUPDUP
Basic options for your PuTTY session
Specify the destination you want to connect to
Host Name (or IP address)
Port
4.247.154.130
22
Connection type:
SSH
Serial
Other:
Telnet
Load, save or delete a stored session
Saved Sessions
Default Settings
Load
Save
Delete
Close window on exit:
Always
Never
Only on clean exit
About
Help
Open
Cancel

```
samthube@sam-vm1: ~
login as: samthube
samthube@52.172.202.200's password:
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1017-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Thu Nov 28 05:07:44 UTC 2024

  System load:  0.22               Processes:             113
  Usage of /:   5.4% of 28.02GB    Users logged in:       0
  Memory usage: 26%                IPv4 address for eth0: 10.0.0.4
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: 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.

samthube@sam-vm1:~$
```

For CMD Use

ssh username@ip

Step 3: Update Package Index

Update the package list:

```
sudo apt update
```

Step 4: Install PostgreSQL

Install PostgreSQL:

```
sudo apt install postgresql postgresql-contrib -y
```

Step 5: Switch to PostgreSQL User

Switch to the PostgreSQL user:

```
sudo -i -u postgres
```

```
samthube@sam-vm1:~$ sudo -i -u postgres
postgres@sam-vm1:~$
```

Step 6: Access PostgreSQL

Access the PostgreSQL shell:

```
Psql
```

```
postgres@sam-vm1:~$ psql
psql (16.4 (Ubuntu 16.4-0ubuntu0.24.04.2))
Type "help" for help.
```

Step 7: Create Database, User, and Grant Privileges

Create a new database and user:

```
CREATE DATABASE your_database_name;

CREATE USER your_user_name WITH PASSWORD 'your_password';

GRANT ALL PRIVILEGES ON DATABASE your_database_name TO your_user_name;
```

```
postgres=# CREATE DATABASE fundoo_db;
CREATE DATABASE
postgres=# CREATE USER samthube WITH PASSWORD Samadhan@123;
ERROR:  syntax error at or near "Samadhan"
LINE 1: CREATE USER samthube WITH PASSWORD Samadhan@123;
                                           ^
postgres=# CREATE USER samthube WITH PASSWORD 'Samadhan@123';
CREATE ROLE
postgres=# GRANT ALL PRIVILEGES ON DATABASE fundoo_db TO samthube;
GRANT
postgres=# \l
                                                  List of databases
   Name    |  Owner   | Encoding | Locale Provider | Collate |  Ctype  | ICU Locale | ICU Rules |   Access privileges
-----------+----------+----------+-----------------+---------+---------+------------+-----------+-----------------------
 fundoo_db | postgres | UTF8     | libc            | C.UTF-8 | C.UTF-8 |            |           | =Tc/postgres         +
           |          |          |                 |         |         |            |           | postgres=CTc/postgres+
           |          |          |                 |         |         |            |           | samthube=CTc/postgres
 postgres  | postgres | UTF8     | libc            | C.UTF-8 | C.UTF-8 |            |           |
 template0 | postgres | UTF8     | libc            | C.UTF-8 | C.UTF-8 |            |           | =c/postgres          +
           |          |          |                 |         |         |            |           | postgres=CTc/postgres
 template1 | postgres | UTF8     | libc            | C.UTF-8 | C.UTF-8 |            |           | =c/postgres          +
           |          |          |                 |         |         |            |           | postgres=CTc/postgres
(4 rows)

postgres=#
```

```
ALTER DATABASE your_database_name OWNER TO nagashree;
```

```
postgres=# ALTER DATABASE fundoo_db OWNER TO samthube;
ALTER DATABASE
postgres=# \l
                                                  List of databases
   Name    |  Owner   | Encoding | Locale Provider | Collate |  Ctype  | ICU Locale | ICU Rules |   Access privileges
-----------+----------+----------+-----------------+---------+---------+------------+-----------+-----------------------
 fundoo_db | samthube | UTF8     | libc            | C.UTF-8 | C.UTF-8 |            |           | =Tc/samthube         +
           |          |          |                 |         |         |            |           | samthube=CTc/samthube
 postgres  | postgres | UTF8     | libc            | C.UTF-8 | C.UTF-8 |            |           |
 template0 | postgres | UTF8     | libc            | C.UTF-8 | C.UTF-8 |            |           | =c/postgres          +
           |          |          |                 |         |         |            |           | postgres=CTc/postgres
 template1 | postgres | UTF8     | libc            | C.UTF-8 | C.UTF-8 |            |           | =c/postgres          +
           |          |          |                 |         |         |            |           | postgres=CTc/postgres
(4 rows)
```

Grant Privileges to `samthube`

Grant Privileges on Existing Tables: Allow `samthube` to perform `SELECT`, `INSERT`, `UPDATE`, and `DELETE` on all existing tables in the `public` schema:

sql

Copy code

```
GRANT SELECT, INSERT, UPDATE, DELETE ON ALL TABLES IN SCHEMA public TO samthube;
```

1.

Grant Privileges on Future Tables: Ensure `samthube` automatically gets these privileges on any tables created in the future within the `public` schema:

sql

Copy code

```
ALTER DEFAULT PRIVILEGES IN SCHEMA public GRANT SELECT, INSERT,
UPDATE, DELETE ON TABLES TO samthube;
```

2.

Grant Privileges on the Database: Allow `samthube` to connect to and work with the `fundoo_db` database:

sql

Copy code

```
GRANT CONNECT ON DATABASE fundoo_db TO samthube;
```

3.

Grant Privileges on Schemas: If `samthube` needs to create objects (like tables or sequences) in the `public` schema, grant the required privileges:

sql

Copy code

```
GRANT USAGE ON SCHEMA public TO samthube;
```

4. `GRANT CREATE ON SCHEMA public TO samthube;`

```
postgres=# GRANT SELECT, INSERT, UPDATE, DELETE ON ALL TABLES IN SCHEMA public TO samthube;
GRANT
postgres=# ALTER DEFAULT PRIVILEGES IN SCHEMA public GRANT SELECT, INSERT, UPDATE, DELETE ON TABLES TO samthube;
ALTER DEFAULT PRIVILEGES
postgres=# GRANT CONNECT ON DATABASE fundoo_db TO samthube;
GRANT
postgres=# GRANT USAGE ON SCHEMA public TO samthube;
GRANT
postgres=# GRANT CREATE ON SCHEMA public TO samthube;
GRANT
postgres=# _
```

Step 8: Configure Azure Network Security Group

11. **Modify the Network Security Group (NSG)**:
    - In the Azure Portal, navigate to the NSG associated with your VM.
    - Add an inbound security rule to allow TCP traffic on port 5432 from your backend VM's IP address or specific IPs.

Microsoft Azure | Upgrade | Search resources, services, and docs (G+/) | Copilot | dipkumarpatil88@gmail... DEFAULT DIRECTORY (DIPKUMA...

Home > Resource groups > Sam-Devops > sam-vm1

sam-vm1 | Network settings

Virtual machine

Search

- Overview
- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems
- Connect
- Networking
  - Network settings
  - Load balancing
  - Application security groups
  - Network manager
- Settings
- Availability + scale
- Security

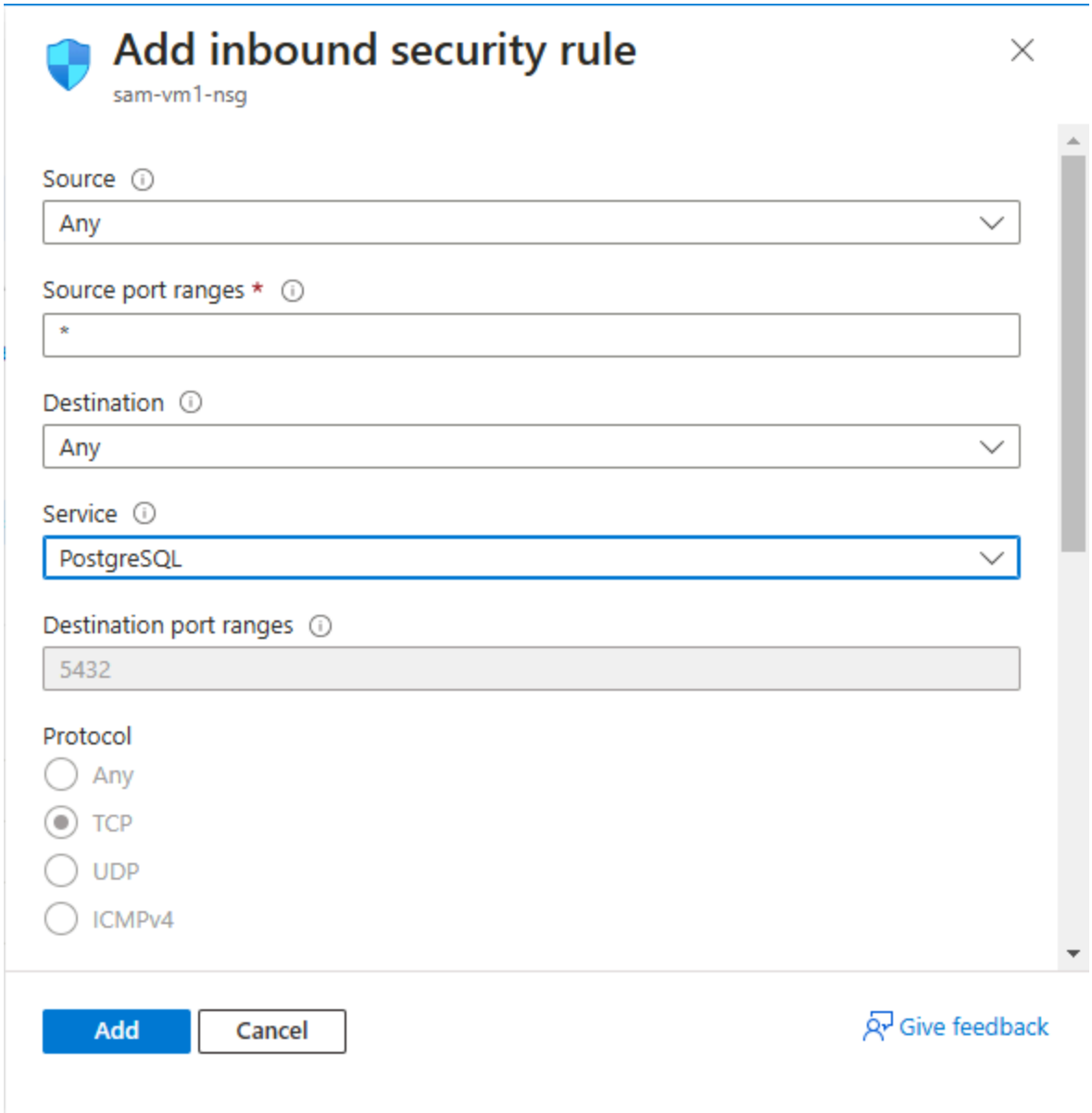
This is a new experience. Please provide feedback

Network security group sam-vm1-nsg (attached to networkInterface: sam-vm1462)
Impacts 0 subnets, 1 network interfaces

+ Create port rule

Search rules | Source == all | Destination == all | Protocol == all | Action == all

Priority ↑	Name	Port	Protocol	Source	Destination	Action
Inbound port rules (5)						
300	SSH	22	TCP	Any	Any	Allow
310	AllowAnyPostgreSQLInbound	5432	TCP	Any	Any	Allow
65000	AllowVnetInBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowAzureLoadBalancerInBound	Any	Any	AzureLoadBalancer	Any	Allow
65500	DenyAllInBound	Any	Any	Any	Any	Deny
Outbound port rules (3)						



Step 9: Configure PostgreSQL to Allow Remote Connections

```
postgres@sam-vm:~$ exit
logout
samthube@sam-vm:~$ sudo passwd postgres
New password:
Retype new password:
passwd: password updated successfully
```

Edit `postgresql.conf`:

Check for the version:

```
postgres@sam-vm1:~$ ls
16
postgres@sam-vm1:~$ sudo nano /etc/postgresql/16/main/postgresql.conf
[sudo] password for postgres:
```

`sudo nano /etc/postgresql/16/main/postgresql.conf`

```
#------------------------------------------------------------------------------
# CONNECTIONS AND AUTHENTICATION
#------------------------------------------------------------------------------

# - Connection Settings -

listen_addresses = '*'          # what IP address(es) to listen on;
                                        # comma-separated list of addresses;
                                        # defaults to 'localhost'; use '*' for all
                                        # (change requires restart)
port = 5432                             # (change requires restart)
max_connections = 100                   # (change requires restart)
#reserved_connections = 0               # (change requires restart)
#superuser_reserved_connections = 3     # (change requires restart)
unix_socket_directories = '/var/run/postgresql' # comma-separated list of directories
                                        # (change requires restart)
```

Change `listen_addresses` to: `listen_addresses = '*'`

Edit `pg_hba.conf`:

`sudo nano /etc/postgresql/16/main/pg_hba.conf`

```
postgres@sam-vm1:~$ sudo nano /etc/postgresql/16/main/pg_hba.conf
```

Add the following line at the end of the file:

`host  all  all  0.0.0.0/0  md5`

```
# Database administrative login by Unix domain socket
local   all             postgres                                peer

# 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
# Database administrative login by Unix domain socket
local   all             postgres                                peer

# TYPE  DATABASE        USER            ADDRESS                 METHOD
```

Step 10: Enable PostgreSQL to Start on Boot

14. **Enable PostgreSQL to run on startup**:
    `sudo systemctl enable postgresql`

```
postgres@sam-vm1:~$ sudo systemctl enable postgresql
```

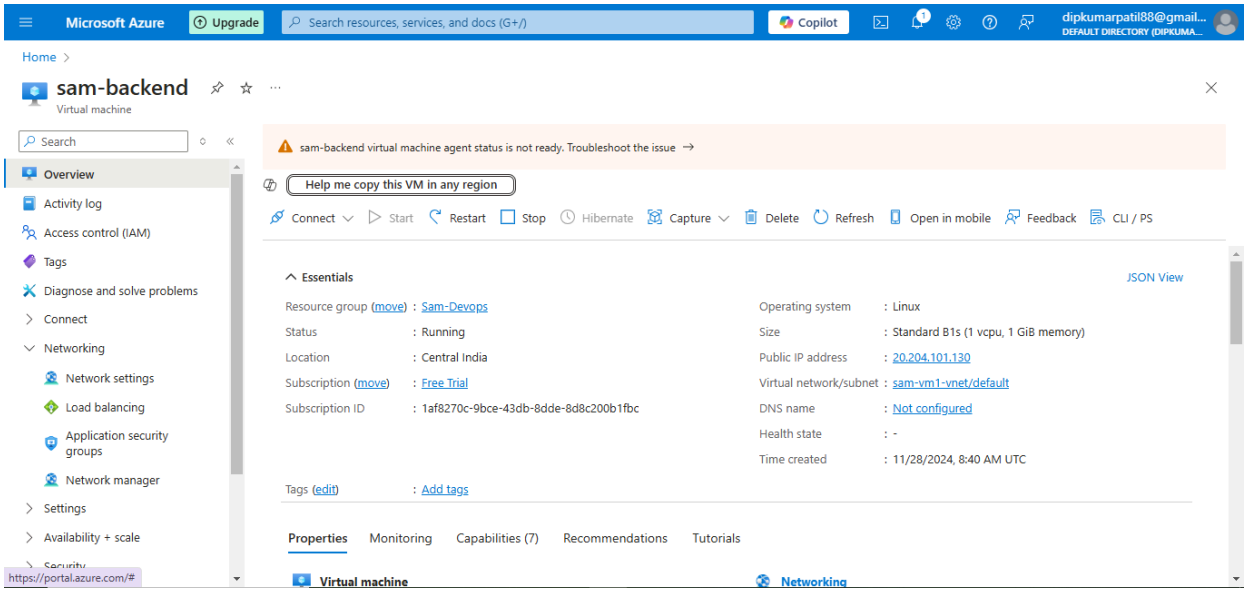
```
postgres@sam-vm1:~$ sudo systemctl reload postgresql
postgres@sam-vm1:~$ psql -U samthube -d fundoo_db
Password for user samthube:
psql (16.4 (Ubuntu 16.4-0ubuntu0.24.04.2))
Type "help" for help.

fundoo_db=> _
```

Logout and close connection

```
fundoo_db=> exit
postgres@sam-vm1:~$ exit
logout
samthube@sam-vm1:~$ exit
logout
Connection to 52.172.202.200 closed.
```

Configuring VM 2 Backend:

Create new VM



Log in to vm

```
C:\WINDOWS\system32>az vm start --resource-group Sam-Devops --name sam-backend --subscription 1af8270c-9bce-43db-8dde-8d8c200b1fbc

C:\WINDOWS\system32>az vm list-ip-addresses --resource-group Sam-Devops --name sam-backend --subscription 1af8270c-9bce-43db-8dde-8d8c200b1fbc
[
  {
    "virtualMachine": {
      "name": "sam-backend",
      "network": {
        "privateIpAddresses": [
          "10.0.0.5"
        ],
        "publicIpAddresses": [
          {
            "id": "/subscriptions/1af8270c-9bce-43db-8dde-8d8c200b1fbc/resourceGroups/Sam-Devops/providers/Microsoft.Network/publicIPAddresses/sam-backend-ip",
            "ipAddress": "20.204.101.130",
            "ipAllocationMethod": "Static",
            "name": "sam-backend-ip",
            "resourceGroup": "Sam-Devops",
            "zone": null
          }
        ]
      },
      "resourceGroup": "Sam-Devops"
    }
  }
]

C:\WINDOWS\system32>ssh samthube@20.204.101.130
The authenticity of host '20.204.101.130 (20.204.101.130)' can't be established.
ED25519 key fingerprint is SHA256:pVMVqJuTUoalEn3NxYhBgPe7R8pHW+aW1H8ZxflDw98.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '20.204.101.130' (ED25519) to the list of known hosts.
samthube@20.204.101.130's password:
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1017-azure x86_64)
```

Update package index

sudo apt update && sudo apt upgrade -y

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

Install PostgreSQL Development Libraries

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

sudo apt install libpq-dev -y

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

```
samthube@sam-backend:~$ python3 -m venv myenv
samthube@sam-backend:~$ source myenv/bin/activate
(myenv) samthube@sam-backend:~$
```

Install Django and Gunicorn

Install Django and Gunicorn (the production WSGI server)

pip install django gunicorn

```
(myenv) samthube@sam-backend:~$ pip install django gunicorn
Collecting django
  Downloading Django-5.1.3-py3-none-any.whl.metadata (4.2 kB)
Collecting gunicorn
  Downloading gunicorn-23.0.0-py3-none-any.whl.metadata (4.4 kB)
Collecting asgiref<4,>=3.8.1 (from django)
  Downloading asgiref-3.8.1-py3-none-any.whl.metadata (9.3 kB)
Collecting sqlparse>=0.3.1 (from django)
  Downloading sqlparse-0.5.2-py3-none-any.whl.metadata (3.9 kB)
Collecting packaging (from gunicorn)
  Downloading packaging-24.2-py3-none-any.whl.metadata (3.2 kB)
Downloading Django-5.1.3-py3-none-any.whl (8.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.3/8.3 MB 18.4 MB/s eta 0:00:00
Downloading gunicorn-23.0.0-py3-none-any.whl (85 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 85.0/85.0 kB 7.0 MB/s eta 0:00:00
Downloading asgiref-3.8.1-py3-none-any.whl (23 kB)
Downloading sqlparse-0.5.2-py3-none-any.whl (44 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 44.4/44.4 kB 3.5 MB/s eta 0:00:00
Downloading packaging-24.2-py3-none-any.whl (65 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 65.5/65.5 kB 4.5 MB/s eta 0:00:00
Installing collected packages: sqlparse, packaging, asgiref, gunicorn, django
Successfully installed asgiref-3.8.1 django-5.1.3 gunicorn-23.0.0 packaging-24.2 sqlparse-0.5.2
(myenv) samthube@sam-backend:~$
```

Clone the Django project from Github

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

```
(myenv) samthube@sam-backend:~$ git clone -b dev https://github.com/antimaYAD/FUNDOO-NOTES
Cloning into 'FUNDOO-NOTES'...
remote: Enumerating objects: 240, done.
remote: Counting objects: 100% (240/240), done.
remote: Compressing objects: 100% (180/180), done.
remote: Total 240 (delta 147), reused 126 (delta 58), pack-reused 0 (from 0)
Receiving objects: 100% (240/240), 67.90 KiB | 202.00 KiB/s, done.
Resolving deltas: 100% (147/147), done.
(myenv) samthube@sam-backend:~$
```

Install requirements.txt

```
(myenv) samthube@sam-backend:~$ ls
FUNDOO-NOTES  myenv
(myenv) samthube@sam-backend:~$ cd FUNDOO-NOTES/
(myenv) samthube@sam-backend:~/FUNDOO-NOTES$ ls
README.md  fundoonote  label  manage.py  notes  pytest.ini  requirements.txt  user
(myenv) samthube@sam-backend:~/FUNDOO-NOTES$ pip install -r requirements.txt
Collecting amqp==5.2.0 (from -r requirements.txt (line 1))
  Downloading amqp-5.2.0-py3-none-any.whl.metadata (8.9 kB)
Collecting anyio==3.6.2 (from -r requirements.txt (line 2))
  Downloading anyio-3.6.2-py3-none-any.whl.metadata (4.7 kB)
Collecting arrow==1.2.3 (from -r requirements.txt (line 5))
  Downloading arrow-1.2.3-py3-none-any.whl.metadata (6.9 kB)
Requirement already satisfied: asgiref==3.8.1 in /home/samthube/myenv/lib/python3.12/site-packages (from -r requirements.txt (line 6)) (3.8.1)
```

Configure PostgreSQL in Django Settings

```
(myenv) samthube@sam-backend:~$ cd FUNDOO-NOTES/
(myenv) samthube@sam-backend:~/FUNDOO-NOTES$ cd fundoonote/
(myenv) samthube@sam-backend:~/FUNDOO-NOTES/fundoonote$ nano settings.py
```

```
# Database
# https://docs.djangoproject.com/en/5.1/ref/settings/#databases

DATABASES = {
     "default": {
        "ENGINE": "django.db.backends.postgresql",
        "NAME": 'fundoo_db',#os.environ.get('DATABASE_NAME'),
        "USER": "samthube",#os.environ.get('DATABASE_USER'),
        "PASSWORD": "Samadhan@123",#os.environ.get('DATABASE_PASSWORD'),
        "HOST" : "52.172.202.200",
        "PORT" : "5432",
    }
}
```

Install Postgresql Client

```
(myenv) samthube@sam-backend:~/FUNDOO-NOTES/fundoonote$ 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-16 postgresql-client-common
Suggested packages:
  postgresql-16 postgresql-doc-16
The following NEW packages will be installed:
  postgresql-client postgresql-client-16 postgresql-client-common
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 1319 kB of archives.
After this operation, 4235 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 samthube -d fundoo_db -h 52.172.202.200

```
(myenv) samthube@sam-backend:~$ psql -U samthube -d fundoo_db -h 52.172.202.200
Password for user samthube:
psql (16.4 (Ubuntu 16.4-0ubuntu0.24.04.2))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression: off)
Type "help" for help.

fundoo_db=>
```

Install Python-dotenv

```
(myenv) samthube@sam-backend:~/FUNDOO-NOTES$ pip install python-dotenv
Collecting python-dotenv
  Using cached python_dotenv-1.0.1-py3-none-any.whl.metadata (23 kB)
Downloading python_dotenv-1.0.1-py3-none-any.whl (19 kB)
Installing collected packages: python-dotenv
Successfully installed python-dotenv-1.0.1
```

Migrate the Database

python manage.py migrate

```
(myenv) samthube@sam-backend:~/FUNDOO-NOTES$ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, django_celery_beat, django_celery_results, label, notes, sessions, user
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.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
  Applying django_celery_beat.0003_auto_20161209_0049... OK
  Applying django_celery_beat.0004_auto_20170221_0000... OK
  Applying django_celery_beat.0005_add_solarschedule_events_choices... OK
```

Install redis server

Sudo apt install redis-server

```
(myenv) nagashree@BackendVMachine:~/Fundoo/fundoo_notes$ sudo apt install redis-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libjemalloc2 liblzf1 redis-tools
Suggested packages:
  ruby-redis
The following NEW packages will be installed:
  libjemalloc2 liblzf1 redis-server redis-tools
0 upgraded, 4 newly installed, 0 to remove and 1 not upgraded.
Need to get 1481 kB of archives.
After this operation, 7558 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
```

Run the server:
Run Django Locally to Test -
python manage.py runserver 0.0.0.0:8000

```
(myenv) samthube@sam-backend:~/Aws_test_django/fundoo_notes$ python manage.py runserver 0.0.0.0:8000
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
November 29, 2024 - 09:16:03
Django version 5.1, using settings 'fundoo_notes.settings'
Starting development server at http://0.0.0.0:8000/
Quit the server with CONTROL-C.
```

Edit Home.html file:

```
(myenv) samthube@sam-backend:~/Aws_test_django/fundoo_notes/templates$ ls
home.html  signin.html  signup.html
(myenv) samthube@sam-backend:~/Aws_test_django/fundoo_notes/templates$ nano home.html
```

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

```
(myenv) samthube@sam-backend:~$ cd  /etc/systemd/system/
(myenv) samthube@sam-backend:/etc/systemd/system$ ls
chronyd.service                          emergency.target.wants     iscsi.service                  redis.service                        sysinit.target.wants
cloud-config.target.wants                final.target.wants         mdmonitor.service.wants        rescue.target.wants                  syslog.service
cloud-final.service.wants                fundoo_notes.service       multi-user.target.wants        sleep.target.wants                   sysstat.service.wants
cloud-init.target.wants                  getty.target.wants         network-online.target.wants    sockets.target.wants                 timers.target.wants
dbus-org.freedesktop.ModemManager1.service graphical.target.wants   network.target.wants           ssh.service.requires                 vmtoolsd.service
dbus-org.freedesktop.resolve1.service    hibernate.target.wants     open-vm-tools.service.requires suspend-then-hibernate.target.wants
default.target.wants                     hybrid-sleep.target.wants  paths.target.wants             suspend.target.wants
(myenv) samthube@sam-backend:/etc/systemd/system$ sudo nano fundoo_notes.service
(myenv) samthube@sam-backend:/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.

```
samthube@sam-backend: /etc/systemd/system
  GNU nano 7.2                                                    fundoo_notes.service *
[Unit]
Description=Fundoo Notes Django Application
After=network.target

[Service]
User=samthube
Group=samthube
WorkingDirectory=/home/samthube/Aws_test_django/fundoo_notes
ExecStart=/home/samthube/myenv/bin/python /home/samthube/Aws_test_django/fundoo_notes/manage.py runserver 0.0.0.0:8000
Restart=always
Environment=PYTHONUNBUFFERED=1

[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

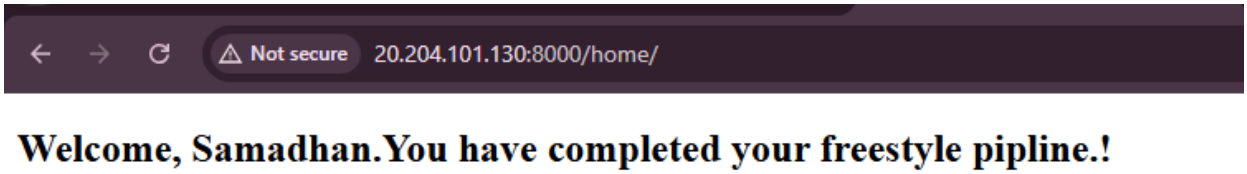
Verify that the service is running correctly sudo

systemctl status fundoo-service

```
(myenv) samthube@sam-backend:/etc/systemd/system$ sudo nano fundoo_notes.service
(myenv) samthube@sam-backend:/etc/systemd/system$ sudo systemctl daemon-reload
(myenv) samthube@sam-backend:/etc/systemd/system$ sudo systemctl start fundoo_notes.service
(myenv) samthube@sam-backend:/etc/systemd/system$ sudo systemctl status fundoo_notes.service
● fundoo_notes.service - Fundoo Notes Django Application
     Loaded: loaded (/etc/systemd/system/fundoo_notes.service; enabled; preset: enabled)
     Active: active (running) since Sat 2024-11-30 05:29:40 UTC; 1min 33s ago
   Main PID: 79402 (python)
      Tasks: 3 (limit: 1004)
     Memory: 113.5M (peak: 115.5M)
        CPU: 2.689s
     CGroup: /system.slice/fundoo_notes.service
             ├─79402 /home/samthube/myenv/bin/python /home/samthube/Aws_test_django/fundoo_notes/manage.py runserver 0.0.0.0:8000
             └─79405 /home/samthube/myenv/bin/python /home/samthube/Aws_test_django/fundoo_notes/manage.py runserver 0.0.0.0:8000

Nov 30 05:29:40 sam-backend systemd[1]: Started fundoo_notes.service - Fundoo Notes Django Application.
Nov 30 05:29:41 sam-backend python[79405]: Watching for file changes with StatReloader
Nov 30 05:29:41 sam-backend python[79405]: Performing system checks...
Nov 30 05:29:41 sam-backend python[79405]: System check identified no issues (0 silenced).
Nov 30 05:29:41 sam-backend python[79405]: November 30, 2024 - 05:29:41
Nov 30 05:29:41 sam-backend python[79405]: Django version 5.1, using settings 'fundoo_notes.settings'
Nov 30 05:29:41 sam-backend python[79405]: Starting development server at http://0.0.0.0:8000/
Nov 30 05:29:41 sam-backend python[79405]: Quit the server with CONTROL-C.
(myenv) samthube@sam-backend:/etc/systemd/system$
```

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.



Perform API testing

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

Not secure 20.204.101.130:8000/swagger/

Swagger

http://20.204.101.130:8000/swagger/?format=openapi

Explore

Fundoo_notes v1

[Base URL: 20.204.101.130:8000/]
http://20.204.101.130:8000/swagger/?format=openapi

Test description

Terms of service
Contact the developer
BSD License

Schemes
HTTP

Authorize

Filter by tag

Labels

GET /label/labels/ list_labels