🚀 Steps to Deploy a Django Project on PythonAnywhere

✅ 1. Prepare Your Django Project (On Local Machine)

a. Create `requirements.txt`

Run the following command in your terminal (inside the project root):

pip freeze > requirements.txt

This will create a file listing all the installed packages your project depends on.

b. Zip your Django project

Zip the entire Django project folder (excluding virtual environment) before uploading.

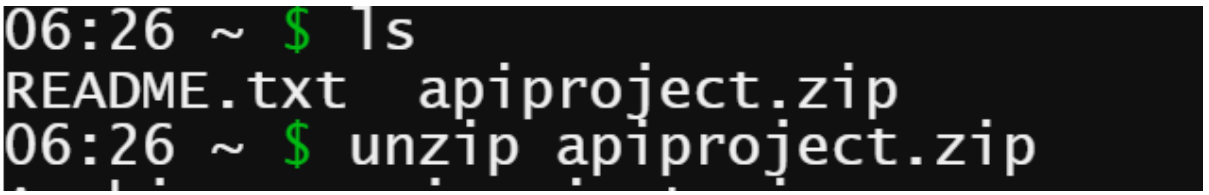
✅ 2. Upload Project to PythonAnywhere

1. **Login** to https://www.pythonanywhere.com
2. Go to the **"Files"** section.
3. Upload the **.zip** file of your project.
4. Once uploaded, go to the **"Consoles"** tab and open a **Bash console**.

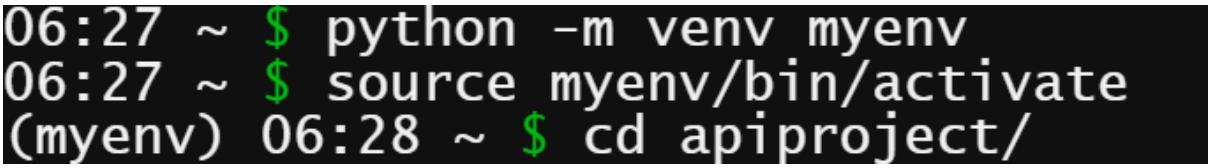
✅ 3. Set Up Environment in PythonAnywhere

a. Unzip your project

Inside the Bash console, run:



b. Create a virtual environment



c. Install dependencies

pip install -r requirements.txt

d. Perform migrations

python manage.py makemigrations
python manage.py migrate

e. Additional commands

1. For generating current project path – pwd
2. For generating virtual env path – echo $VIRTUAL_ENV

✅ 4. Run `collectstatic`

Now that your virtual environment is active and dependencies are installed, you need to collect static files:

python manage.py collectstatic

- `Type 'yes' to continue, or 'no' to cancel: yes`

If error shows :

a. Set `STATIC_ROOT` in `settings.py`

Add the following at the bottom of your `settings.py` file:

```
import os
```

```
STATIC_ROOT = os.path.join(BASE_DIR, 'staticfiles') # Collects all static files here
```

Note: Ensure to click on “ save” then run command again in console

This is required so you can later run `collectstatic` and serve static files in production.

🔹 This will gather all static files (CSS, JS, images) into the `staticfiles/` directory, as specified in your `settings.py`.

✅ 5. Apply Database Migrations

Run the following commands to set up your database schema:

python manage.py makemigrations

python manage.py migrate

✅ 6. Configure Web App on PythonAnywhere

- Go to the **"Web"** tab on PythonAnywhere.
- Click **"Add a new web app"**.
- Choose **Manual configuration**, and select the correct **Python version** (must match your virtualenv)--(3.13.1)
- Set the **Source code directory** (where your `manage.py` is located).
- Set the **Virtualenv path** to something like:

/home/yourusername/myenv

6. Scroll to the **WSGI configuration file** link and click it.

✅ 7. Edit the WSGI File

Uncomment and modify like this:

```
import sys

import os


path = '/home/yourusername/your_project_folder'

if path not in sys.path:

    sys.path.append(path)


os.environ['DJANGO_SETTINGS_MODULE'] =
'your_project_name.settings'


from django.core.wsgi import get_wsgi_application

application = get_wsgi_application()
```

> Replace `yourusername`, `your_project_folder`, and `your_project_name` accordingly.
>
> **Note: Ensure to click on “ save”**

✅ 8. Reload Your Web App

1. Go back to the **"Web"** tab.

2. Click **Reload** at the top right.

✅ 9. AllowedHost and Debug

Go to setting.py and update it

- DEBUG=False
- ALLOWED_HOST=[‘specify url provided by pythonanywhere’]

Note: Ensure to click on “ save”

Go back to the **"Web"** tab.

Click **Reload** at the top right.

✅ 10. Done! 🎉

Visit your website using the PythonAnywhere-provided domain like:

https://yourusername.pythonanywhere.com

✅ 10. Static file attachment(Optional) 🎉

Configure Static Files in PythonAnywhere Web Tab

Go to the **Web tab** on PythonAnywhere and scroll to the **Static files** section. Add this:

- **URL:** `/static/`
- **Directory:** `/home/yourusername/yourprojectfolder/staticfiles`

Replace `yourusername` and `yourprojectfolder` accordingly.

Click the **"Reload"** button at the top of the Web tab.

✅ Steps to Delete a Deployed Project on PythonAnywhere

1. Delete the Project Files (Zip and Unzipped)

Go to the Files tab.

Navigate to the location where you uploaded your Django project.

Delete:

The uploaded .zip file

The unzipped project folder (usually created when you extracted the zip).

2. Stop Running Processes

Go to the Consoles tab.

If any Bash, Python, or server processes related to your project are running, click "Kill" next to them to stop them.

3. Delete the Web App

Go to the Web tab.

Scroll to the bottom and click "Delete this web app".

Confirm the deletion.

✅ For Flask project deployement 🎉

Follow same step as django project deployment except step-4 (**python manage.py collectstatic)**
