Django Blog Cheat Sheet

How to set up a basic Django Project and deploy to Heroku

Step 1: Installing Django and supporting libraries

Note:
It is recommended when you are still learning this content that you type out each line of code, rather than copying and pasting. This will help you learn!

Key:
For this project, ***PROJ_NAME*** is "codestar2021" and ***APP_NAME*** is "blog". You will need to change these names for your own projects.

In the Terminal:

#	Step	Code
1.	Install Django and gunicorn:	pip3 install django gunicorn
2.	Install supporting libraries:	pip3 install dj_database_url psycopg2
3.	Install Cloudinary Libraries	pip3 install dj3-cloudinary-storage
4.	Create requirements file	pip3 freeze --local > requirements.txt
5.	Create Project	django-admin startproject PROJ_NAME **.** *(Don't forget the .)*
6.	Create App	python3 manage.py startapp APP_NAME

settings.py

#	Step	Code
7.	Add to **installed apps**	'APP_NAME',

In the Terminal:

#	Step	Code
8.	Migrate Changes	python3 manage.py migrate
9.	Run Server to Test	python3 manage.py runserver

Step 2: Deploying an app to Heroku

4 stages:

1. Create the Heroku app
2. Attach the database
3. Prepare our environment and settings.py file
4. Get our static and media files stored on Cloudinary.

2.1 Create the Heroku app

In Heroku:

#	Step	Code
1.	Create new Heroku App	APP_NAME, Location = Europe
2.	Add Database to App Resources	Located in the Resources Tab, Add-ons, search and add e.g. ‘Heroku Postgres’
3.	Copy DATABASE_URL	Located in the Settings Tab, in Config Vars, Copy Text

2.2 Attach the Database:

In gitpod

#	Step	Code
4.	Create new env.py file on top level directory	E.g. env.py

In env.py

#	Step	Code
5.	Import os library	import os
6.	Set environment variables	os.environ["DATABASE_URL"] = "Paste in Heroku DATABASE_URL Link"
7.	Add in secret key	os.environ["SECRET_KEY"] = "Make up a randomSecretKey"

In heroku.com

#	Step	Code
8.	Add Secret Key to Config Vars	SECRET_KEY, “randomSecretKey”

2.3 Prepare our environment and settings.py file:

In settings.py

#	Step	Code

9.	Reference env.py	*from pathlib import Path*
		import os
		import dj_database_url
		
		if os.path.isfile("env.py"):
		   import env

10.	Remove the insecure secret key and replace - *links to the secret key variable on Heroku*	SECRET_KEY = os.environ.get('SECRET_KEY')

11.	**Replace** DATABASES Section (Comment out the old DataBases Section)
	- *links to the DATATBASE_URL variable on Heroku*
		DATABASES = {
		   'default':
		dj_database_url.parse(os.environ.get("DATABASE_URL"))
		}

In the Terminal

#	Step	Code
12.	Make Migrations	python3 manage.py migrate

2.4 Get our static and media files stored on Cloudinary:

In Cloudinary:

#	Step	Code
1.	Copy your CLOUDINARY_URL e.g. API Environment Variable.	From Cloudinary Dashboard

In env.py:

#	Step	Code
2.	Add Cloudinary URL to env.py - *be sure to paste in the correct section of the link*	os.environ["CLOUDINARY_URL"] = "cloudinary://9444:SUZi@dbhyuj5mc"

In Heroku:

#	Step	Code
3.	Add Cloudinary URL to Heroku Config Vars - *be sure to paste in the correct section of the link*	Add to Settings tab in Config Vars e.g. COUDINARY_URL, cloudinary://9444:SUZi@dbhyuj5mc
4.	Add DISABLE_COLLECTSTATIC to Heroku Config Vars **(temporary step for the moment, must be removed before deployment)**	e.g. DISABLE_COLLECTSTATIC, 1

In settings.py:

#	Step	Code
5.	Add Cloudinary Libraries to installed apps	... 'cloudinary_storage', ***'django.contrib.staticfiles',*** 'cloudinary', ... **(note: order is important)**

6.	Tell Django to use Cloudinary to store media and static files
	Place under the Static files Note
		STATIC_URL = '/static/'
		STATICFILES_STORAGE = 'cloudinary_storage.storage.StaticHashedCloudinaryStorage'
		STATICFILES_DIRS = [os.path.join(BASE_DIR, 'static')]
		STATIC_ROOT = os.path.join(BASE_DIR, 'staticfiles')
		
		MEDIA_URL = '/media/'
		DEFAULT_FILE_STORAGE = 'cloudinary_storage.storage.MediaCloudinaryStorage'

7.	Link file to the templates directory in Heroku *Place under the BASE_DIR line*	TEMPLATES_DIR = os.path.join(BASE_DIR, 'templates')
8.	Change the templates directory to TEMPLATES_DIR *Place within the TEMPLATES array*	'DIRS': [TEMPLATES_DIR]
9.	Add Heroku Hostname to ALLOWED_HOSTS	ALLOWED_HOSTS = ["PROJ_NAME.herokuapp.com", "localhost"]

In Gitpod:

#	Step	Code
10.	Create 3 new folders on top level directory	media, static, templates
11.	Create procfile on the top level directory	Procfile

In Procfile

#	Step	Code
12.	Add code	web: gunicorn PROJ_NAME.wsgi

In the Terminal:

#	Step	Code
13.	Add, Commit and Push	git add . git commit -m “Deployment Commit” git push

In Heroku:

#	Step	Code
14.	Deploy Content manually through heroku/	E.g Github as deployment method, on main branch