

Django is quite popular among web developers because it is broad, powerful, scalable and has the “whole package” built in: ORM, authentication, an admin interface and URL routing. It maintains clean code by adhering to the DRY principle. Furthermore, it has a really robust and active community and thorough documentation which makes Django a solid choice for developers particularly of large apps that rely on solid databases.

Spotify: Spotify is a music streaming service that uses Django for its web app development. Django allows Spotify to have the most highly functional app it can have.

Dropbox: Dropbox is a cloud storage service primarily used for photos, videos and documents. It has millions of users and Django allows it to synchronize users’ items and facilitate sharing, storage and access.

Pinterest: Pinterest allows its users a social media platform for “visual discovery” where they can share photos, videos, articles, recipes, etc. across an incredibly broad spectrum of interests. Django makes the performance of Pinterest’s web app as smooth and as stable as it can possibly be.

National Geographic: National Geographic prides itself on being at the forefront of education, storytelling, science and exploration, and it uses the Django framework to create a CMS to handle all aspects of their web application including photos, videos, advertisements, donations.

Instagram: Instagram is a wildly popular photo and video sharing social media platform. They use Django as their web app framework as it can handle and process massive amounts of data and user interactions.

For each of the following scenarios, explain if you would use Django (and why or why not):

- You need to develop a web application with multiple users.

Django would work well in this situation as it is a framework made to create web apps and to handle complex data smoothly. It also has the ability to scale up for future growth.

- You need fast deployment and the ability to make changes as you proceed.

Django is going to work well for this situation – these are two of its primary selling points: the ability to deploy quickly and the flexibility to make changes.

- You need to build a very basic application, which doesn’t require any database access or file operations.

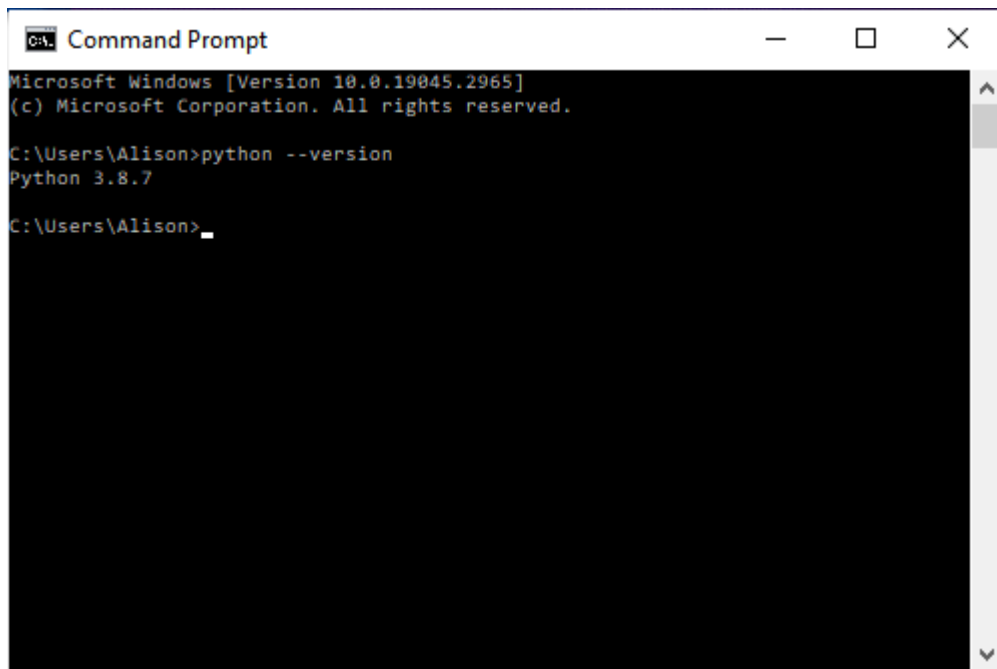
This is probably not a case where I would use Django. It’s strength is in its complex data management, and its robustness would be burdensome on a small and basic app.

- You want to build an application from scratch and want a lot of control over how it works.

This is again probably not the app for Django as it has a lot of its functionality preset as a “batteries-included” framework. It maintains a lot of the control over an app.

- You’re about to start working on a big project and are afraid of getting stuck and needing additional support.

Django could be solid here because of its robust documentation and community support.



```
Command Prompt
Microsoft Windows [Version 10.0.19045.2965]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Alison>python --version
Python 3.8.7

C:\Users\Alison>_
```

```
Command Prompt
Microsoft Windows [Version 10.0.19045.2965]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Alison>python --version
Python 3.8.7

C:\Users\Alison>cd documents\careerfoundry\python\achievement2

C:\Users\Alison\Documents\CareerFoundry\Python\achievement2>mkvirtualenv achieve
ment2-practice
created virtual environment CPython3.8.7.final.0-64 in 774ms
  creator CPython3Windows(dest=C:\Users\Alison\Envs\achievement2-practice, clear
=False, no_vcs_ignore=False, global=False)
  seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle
, via=copy, app_data_dir=C:\Users\Alison\AppData\Local\pypa\virtualenv)
    added seed packages: pip==23.1.2, setuptools==67.7.2, wheel==0.40.0
  activators BashActivator,BatchActivator,FishActivator,NushellActivator,PowerSh
ellActivator,PythonActivator

(achievement2-practice) C:\Users\Alison\Documents\CareerFoundry\Python\achieveme
nt2>_
```

```
Command Prompt

(achievement2-practice) C:\Users\Alison\Documents\CareerFoundry\Python\achieveme
nt2>py -m pip install Django
Collecting Django
  Using cached Django-4.2.2-py3-none-any.whl (8.0 MB)
Collecting asgiref<4,>=3.6.0 (from Django)
  Using cached asgiref-3.7.2-py3-none-any.whl (24 kB)
Collecting sqlparse>=0.3.1 (from Django)
  Using cached sqlparse-0.4.4-py3-none-any.whl (41 kB)
Collecting backports.zoneinfo (from Django)
  Using cached backports.zoneinfo-0.2.1-cp38-cp38-win_amd64.whl (38 kB)
Collecting tzdata (from Django)
  Using cached tzdata-2023.3-py2.py3-none-any.whl (341 kB)
Collecting typing-extensions>=4 (from asgiref<4,>=3.6.0->Django)
  Using cached typing_extensions-4.6.3-py3-none-any.whl (31 kB)
Installing collected packages: tzdata, typing-extensions, sqlparse, backports.zo
neinfo, asgiref, Django
Successfully installed Django-4.2.2 asgiref-3.7.2 backports.zoneinfo-0.2.1 sqlpa
rse-0.4.4 typing-extensions-4.6.3 tzdata-2023.3

(achievement2-practice) C:\Users\Alison\Documents\CareerFoundry\Python\achieveme
nt2>django-admin --version
4.2.2

(achievement2-practice) C:\Users\Alison\Documents\CareerFoundry\Python\achieveme
nt2>
```