

Exercise 2.1

2.

Django is popular among web developers because it offers fast processing, and allows for rapid development as provides authentication systems, ORM, URL routing, forms, session management, ... all out of the box. It also handles security, offering several features from user authentication and password hashing to protection for SQL injections or cross-site requests.

It makes scaling up easier thanks to loosely coupled architecture and the fact it has a strict rules can make it harder but also takes a lot of things off the developer with a lot of precoding.

And of course, it has a strong community which is a major plus for support.

3.

- Instagram (social media platform to share pictures and video content) : They use Django on the backend to handle large-scale traffic, user authentication, feeds, API for mobile/web apps).
- Mozilla (creator of the Firefox browser as well as various web services) : Used for parts of their web services like the support site or add-ons site in order to manage content and handle a large user base.
- Dropbox (cloud storage and file sharing services) : Used in their web-app layer to handle file sharing, and the communication between the user interface and backend.
- NASA (US space and aeronautic program) : Used for their web and intranet services as well as scientific web apps.
- The Washington Post (newspaper company) : Used for content-heavy parts online, like publishing, article management and high volume traffic.

4. Scenarios in which whether to use Django or not :

- *You need to develop a web application with multiple users :*

Absolutely, especially if you need user accounts and features like moderator/admin permissions, Django offers all the user management tools that would make it easier at any scale.

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

Absolutely, Django allows for fast deployment thanks to all the built-in tools and ORM.

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

In that case, no. One of the main interest of Django is making DB communication and various features easy thanks to precoding, if a DB is not required and if it is a small project, it loses interest as Django is still heavy, so it wouldn't be needed for a basic app that doesn't require heavy coding, Flask would work better here.

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

The precoding would help in the build, but the rigid structure of Django wouldn't give much control because of hard rules, so in that case, no.

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

Django would be a good choice here, as it's great for scalability and support because of a strong community. On the other hand, it depends of the project.

5. Python version :



A screenshot of a terminal window in a dark-themed code editor. The tabs at the top are PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (which is underlined), and PORTS. The terminal output shows:

```
● (web-dev) PS C:\Users\theho\Documents\cf-python-course\Exercise 2.1> python --version
Python 3.14.0
❖ (web-dev) PS C:\Users\theho\Documents\cf-python-course\Exercise 2.1> █
```

7. Created and activated virtual environment :



A screenshot of a terminal window in a dark-themed code editor. The tabs at the top are PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (which is underlined), and PORTS. The terminal output shows:

```
● PS C:\Users\theho\Documents\cf-python-course> python -m venv achievement2-practice
● PS C:\Users\theho\Documents\cf-python-course> .\achievement2-practice\Scripts\Activate.ps1
❖ (achievement2-practice) PS C:\Users\theho\Documents\cf-python-course> █
```

8. Django version :

The screenshot shows a terminal window with the following content:

```
(achievement2-practice) PS C:\Users\theho\Documents\cf-python-course> pip install django
Collecting asgiref>=3.8.1 (from django)
  Using cached asgiref-3.10.0-py3-none-any.whl.metadata (9.3 kB)
Collecting sqlparse>=0.3.1 (from django)
  Using cached sqlparse-0.5.3-py3-none-any.whl.metadata (3.9 kB)
Collecting tzdata (from django)
  Using cached tzdata-2025.2-py2.py3-none-any.whl.metadata (1.4 kB)
Using cached django-5.2.7-py3-none-any.whl (8.3 MB)
Using cached asgiref-3.10.0-py3-none-any.whl (24 kB)
Using cached sqlparse-0.5.3-py3-none-any.whl (44 kB)
Using cached tzdata-2025.2-py2.py3-none-any.whl (347 kB)
Installing collected packages: tzdata, sqlparse, asgiref, django
Successfully installed asgiref-3.10.0 django-5.2.7 sqlparse-0.5.3 tzdata-2025.2

[notice] A new release of pip is available: 25.2 -> 25.3
[notice] To update, run: python.exe -m pip install --upgrade pip
● (achievement2-practice) PS C:\Users\theho\Documents\cf-python-course> python -m django --version
5.2.7
❖ (achievement2-practice) PS C:\Users\theho\Documents\cf-python-course>
```