Getting Started with Dijango

Write two to three sentences on why Django is so popular among web developers.

Django framework gained its popularity mainly through its fast paced development, the DRY (Don't repeat yourself) being its core principle to enhance speed and efficiency. In fact it brags as being the "web framework for perfectionists with deadlines". Furthermore, its admin panel adds the ability of non-technical persons to log in and manage web or app contents and comes with wide-range of tools, libraries and features ready to use. Also scalability, robust security and strong community support contributes to its fame.

List five large companies that use Django. Specify what the company's product or service is and what they use Django for.

1. Instagram: Django framework helps Instagram process large amount of data and perform user interaction fast every minute of the day.
2. Youtube: Django ensures that new features and updates are added in short period of time and eliminates the risk of making massive flaws in implementing.
3. Spotify: spotify has a large library with huge amount of data accessible everywhere in any device and to handle these data it uses django.
4. Pinterest: pinterest uses django to ensure best performance to its heavy load users of monthly 250 million people.
5. Mozilla: it uses Django to manage high traffic and improve efficiency from millions of searches daily.

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

- **You need to develop web application with multiple users.** I would use Django in this case as it would help deploy the app quickly and scale it up in future to improve user experience.
- **You need fast deployment and the ability to make changes as you proceed.** One of the main features of Django is its fast deployment speed and easy flawless scalability so I would use it.
- **You need to build a very basic application, which doesn't require any database access or file operations.** I wouldn't use django here either, because its strength is seen when most of its features is used. In small applications like this though it will in fact be unnecessary heavy burden for the server.
- **You want to build an application from scratch and want a lot of control over how it works.** Again I wouldn't use django here as, django being batteries-included framework, I will definitely lose some control on how I can build it.
- **You are about to start working on a big project and are afraid of getting stuck and needing additional support.** I would use Django here. It would help with its built in libraries and tools if I got stuck and I could always scale down without/less flaws.

Download and install Python

```
(web-dev) C:\Users\Moses>python --version
Python 3.9.13

(web-dev) C:\Users\Moses>
```

Install Django and verify the installation by checking the version.

```
(django) C:\Users\Moses>py -m pip install Django
Collecting Django
  Downloading Django-4.1.4-py3-none-any.whl (8.1 MB)
     ---------------------------------------- 8.1/8.1 MB 61.5 kB/s eta 0:00:00
Collecting tzdata
  Downloading tzdata-2022.7-py2.py3-none-any.whl (340 kB)
     ---------------------------------------- 340.1/340.1 kB 59.1 kB/s eta 0:00:00
Collecting asgiref<4,>=3.5.2
  Downloading asgiref-3.5.2-py3-none-any.whl (22 kB)
Collecting sqlparse>=0.2.2
  Downloading sqlparse-0.4.3-py3-none-any.whl (42 kB)
     ---------------------------------------- 42.8/42.8 kB 69.4 kB/s eta 0:00:00
Installing collected packages: tzdata, sqlparse, asgiref, Django
Successfully installed Django-4.1.4 asgiref-3.5.2 sqlparse-0.4.3 tzdata-2022.7

(django) C:\Users\Moses>django-admin --version
4.1.4

(django) C:\Users\Moses>
```