Why Django is Popular Among Web Developers

Django is really popular because it makes building websites easier and faster. It comes with a lot of built-in tools, like a user login system and a way to interact with databases, which saves developers a lot of time. It's also secure, well-documented, and has a big community, so it's easier to find help when you need it.

Five Large Companies Using Django

1. Instagram: Instagram is a social media app for sharing photos and videos. They use Django to handle all the users and to add new features quickly.
2. Pinterest: Pinterest is a platform for sharing images and ideas. They use Django to manage their services and to make their website scalable as it grows.
3. Spotify: Spotify is a music streaming service. They use Django to build the backend of their app, like managing user accounts and suggesting music.
4. Disqus: Disqus is a commenting system for websites. They use Django to handle lots of users interacting with their service at the same time.
5. Mozilla: Mozilla is the company behind the Firefox browser. They use Django for different web services, including support pages, because it's secure and flexible.

Scenarios

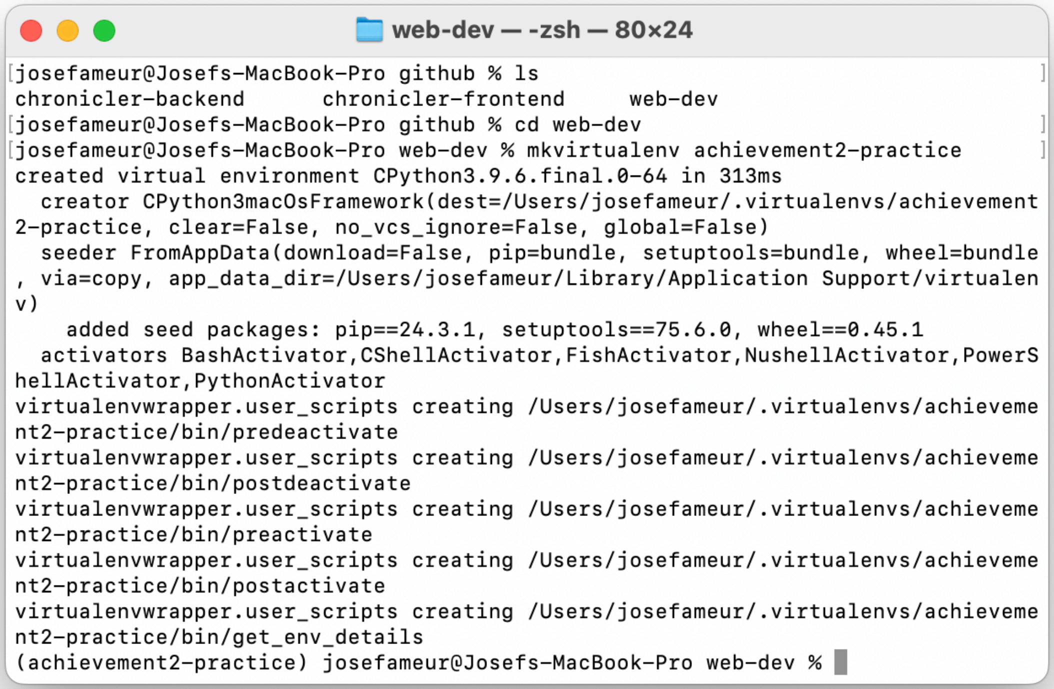
1. You need to develop a web application with multiple users
   Yes, Django is a good choice because it has built-in tools for managing users, like logging in and permissions, which would save you time.

2. You need fast deployment and the ability to make changes as you proceed
   Yes, Django is great for this. It helps you build and deploy your project quickly, and because it's easy to change, you can make updates while you keep working on it.

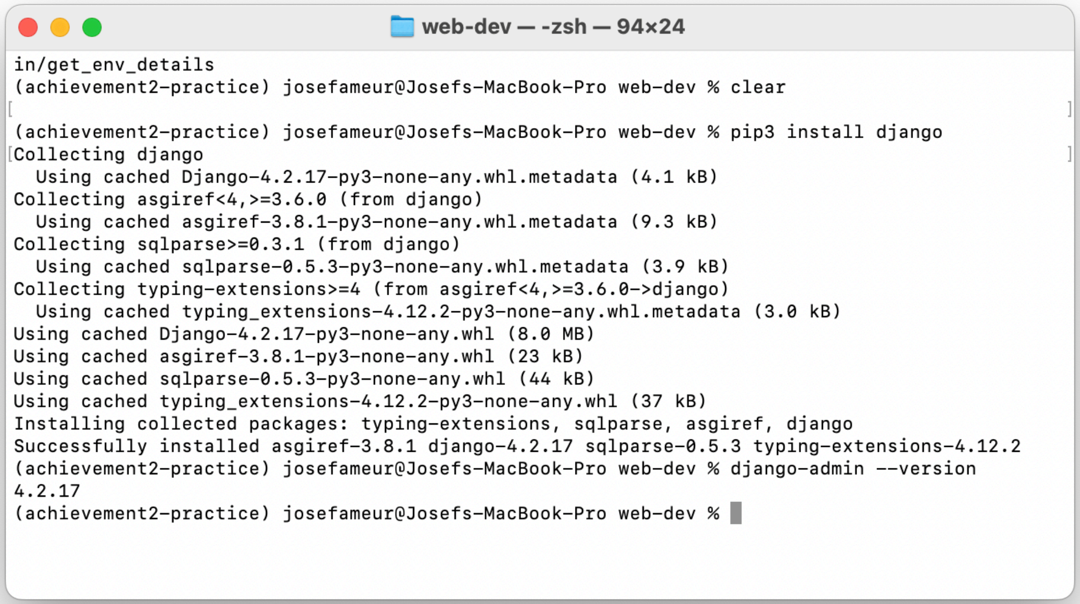
3. You need to build a very basic application, which doesn't require any database access or file operations
   No, Django might be too much for this. If you don't need a database or complex features, something simpler like Flask might work better for your needs.

4. You want to build an application from scratch and want a lot of control over how it works
   Maybe not. Django comes with a lot of built-in features that you may not need if you want complete control. In that case, something like Flask would give you more flexibility.

5. You're about to start working on a big project and are afraid of getting stuck and needing additional support
   Yes, Django is a great choice. There's a large community and lots of helpful resources, so if you run into problems, there's a lot of support to help you out.



```
[josefameur@Josefs-MacBook-Pro github % ls                                     ]
chronicler-backend      chronicler-frontend     web-dev
[josefameur@Josefs-MacBook-Pro github % cd web-dev                             ]
[josefameur@Josefs-MacBook-Pro web-dev % mkvirtualenv achievement2-practice    ]
created virtual environment CPython3.9.6.final.0-64 in 313ms
  creator CPython3macOsFramework(dest=/Users/josefameur/.virtualenvs/achievement
2-practice, clear=False, no_vcs_ignore=False, global=False)
  seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle
, via=copy, app_data_dir=/Users/josefameur/Library/Application Support/virtualen
v)
    added seed packages: pip==24.3.1, setuptools==75.6.0, wheel==0.45.1
  activators BashActivator,CShellActivator,FishActivator,NushellActivator,PowerS
hellActivator,PythonActivator
virtualenvwrapper.user_scripts creating /Users/josefameur/.virtualenvs/achieveme
nt2-practice/bin/predeactivate
virtualenvwrapper.user_scripts creating /Users/josefameur/.virtualenvs/achieveme
nt2-practice/bin/postdeactivate
virtualenvwrapper.user_scripts creating /Users/josefameur/.virtualenvs/achieveme
nt2-practice/bin/preactivate
virtualenvwrapper.user_scripts creating /Users/josefameur/.virtualenvs/achieveme
nt2-practice/bin/postactivate
virtualenvwrapper.user_scripts creating /Users/josefameur/.virtualenvs/achieveme
nt2-practice/bin/get_env_details
(achievement2-practice) josefameur@Josefs-MacBook-Pro web-dev %
```



```
in/get_env_details
(achievement2-practice) josefameur@Josefs-MacBook-Pro web-dev % clear
[                                                                                              ]
(achievement2-practice) josefameur@Josefs-MacBook-Pro web-dev % pip3 install django
[Collecting django                                                                             ]
  Using cached Django-4.2.17-py3-none-any.whl.metadata (4.1 kB)
Collecting asgiref<4,>=3.6.0 (from django)
  Using cached asgiref-3.8.1-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 typing-extensions>=4 (from asgiref<4,>=3.6.0->django)
  Using cached typing_extensions-4.12.2-py3-none-any.whl.metadata (3.0 kB)
Using cached Django-4.2.17-py3-none-any.whl (8.0 MB)
Using cached asgiref-3.8.1-py3-none-any.whl (23 kB)
Using cached sqlparse-0.5.3-py3-none-any.whl (44 kB)
Using cached typing_extensions-4.12.2-py3-none-any.whl (37 kB)
Installing collected packages: typing-extensions, sqlparse, asgiref, django
Successfully installed asgiref-3.8.1 django-4.2.17 sqlparse-0.5.3 typing-extensions-4.12.2
(achievement2-practice) josefameur@Josefs-MacBook-Pro web-dev % django-admin --version
4.2.17
(achievement2-practice) josefameur@Josefs-MacBook-Pro web-dev %
```