Task 2.1 - Getting Started with Django

<u>1. Write two or three sentences on why Django is so popular among web developers.</u>

Like other web frameworks, Django *simplifies* and *streamlines* the development process for developers, allowing them to focus more time and effort on creating great features and logic for its users. More specifically to Django, it offers great *security features*, offers a *powerful ORM* to simplify database interaction, is great for *scalability*, *fast deployment*, and has a massive *support community* as well as great *documentation*.

<u>2. List five large companies that use Django. Specify what the company's product or service is and what they use Django for.</u>

Instagram: A photo/video-sharing social network.
  - They use Django because of it's scalability (their need to process massive amounts of data and even more interactions between users), security features for massive amounts of user sign-ins and sensitive info, it's ORM to simplify the constant back and forth with the database for user content. Also, Instagram began as a startup, so they were able to quickly build a prototype in two weeks.

Spotify: A music and podcast streaming service.
  - They use Django for data analysis and backend services, specifically speed, machine learning and AI, and the community that comes along with Django and Python. They are an active contributor in the community as well.

YouTube: A video-sharing and streaming service.
  - They use Django because the amount of content that moves through YouTube requires strong performance and constant development for adding and updating features.

Eventbrite: A ticketing and event management platform.
  - Eventbrite started using Django when the platform audience and interaction volumes started increasing and the in-house frameworks couldn't keep up. Now, Eventbrite can adjust events to any audience size.

Dropbox: A cloud-storage service for documents, videos, pictures, etc.
  - They use Django for their backend and client desktop software. Dropbox developed a user history option, account synchronization, and a file sharing option with Django. The creator created a prototype in two weeks.

<u>3. For each of the following scenarios, explain if you would use Django (and why or why not):</u>

You need to develop a web application with multiple users.

- **Yes**. Django's scalability is one of the reasons Instagram and Pinterest use it. Django offers great security features, allowing secure-by-design implementation with built-in automated encryption. It also has fast processing, which is good for an app with a lot of users.

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

- **Yes**. Django is based on MVT (Model View Template) architecture, making it suitable for fast prototyping, a lot of changes, and high-speed apps. MVT architecture ensures development is fast and easy.

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

- **No**. Django offers a lot of out-of-the-box solutions. These solutions are best utilized in projects that need them. If your app is simple , it might not be necessary. Furthermore, if your app doesn't need a database or if it isn't web-based, you don't need Django.

You want to build an application from scratch and want a lot of control over how it works.
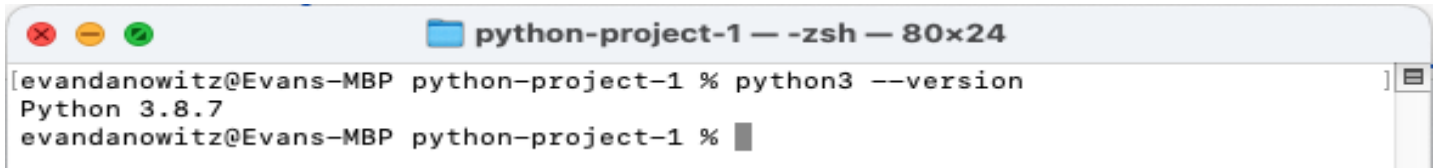
- **No**. Django requires that things be done a certain way. You cannot diverge from the rules. For some, this is a good thing, but you will lose some control over the internals of your system. If you need to have control over the fine details of your system, Django might not be suitable for your project.

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

- **Yes**. Django is a good choice for apps that need to be scalable. Additionally, it has a huge support community with plenty of in-depth documentation.
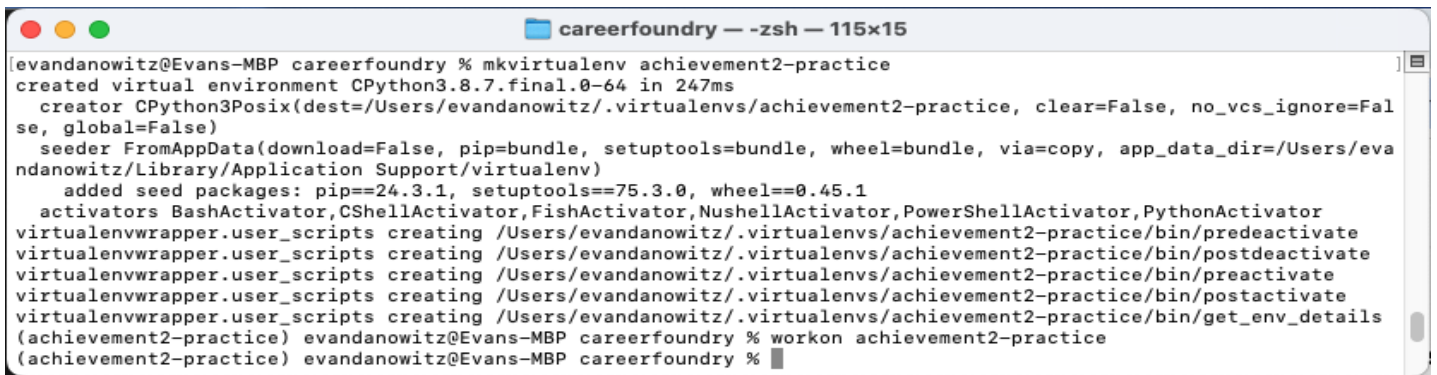
Screenshots

Check previously installed Python version: Check version: `python3 --version`

```
● ● ●                    📁 python-project-1 — -zsh — 80×24
[evandanowitz@Evans-MBP python-project-1 % python3 --version
Python 3.8.7
evandanowitz@Evans-MBP python-project-1 % ▊
```

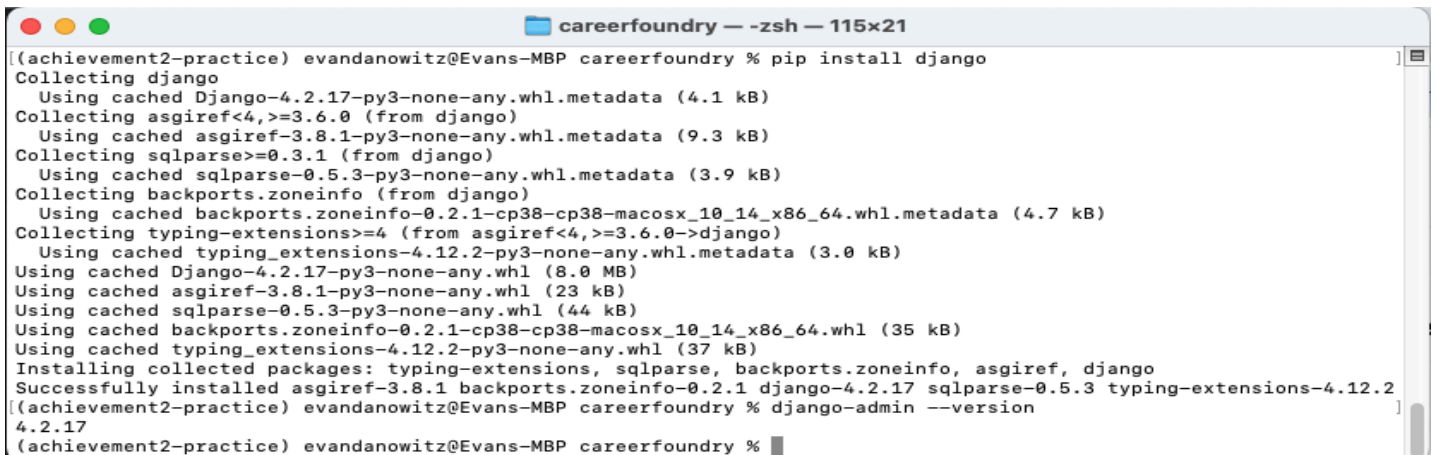Setup and create a virtual environment named "achievement2-practice". Then, activate it:
  - Create: `mkvirtualenv achievement2-practice` | Activate: `workon achievement2-practice`

```
● ● ●                    📁 careerfoundry — -zsh — 115×15
[evandanowitz@Evans-MBP careerfoundry % mkvirtualenv achievement2-practice
created virtual environment CPython3.8.7.final.0-64 in 247ms
   creator CPython3Posix(dest=/Users/evandanowitz/.virtualenvs/achievement2-practice, clear=False, no_vcs_ignore=Fal
se, global=False)
   seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy, app_data_dir=/Users/eva
ndanowitz/Library/Application Support/virtualenv)
      added seed packages: pip==24.3.1, setuptools==75.3.0, wheel==0.45.1
   activators BashActivator,CShellActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActivator
virtualenvwrapper.user_scripts creating /Users/evandanowitz/.virtualenvs/achievement2-practice/bin/predeactivate
virtualenvwrapper.user_scripts creating /Users/evandanowitz/.virtualenvs/achievement2-practice/bin/postdeactivate
virtualenvwrapper.user_scripts creating /Users/evandanowitz/.virtualenvs/achievement2-practice/bin/preactivate
virtualenvwrapper.user_scripts creating /Users/evandanowitz/.virtualenvs/achievement2-practice/bin/postactivate
virtualenvwrapper.user_scripts creating /Users/evandanowitz/.virtualenvs/achievement2-practice/bin/get_env_details
(achievement2-practice) evandanowitz@Evans-MBP careerfoundry % workon achievement2-practice
(achievement2-practice) evandanowitz@Evans-MBP careerfoundry % ▊
```

Install Django and verify the installation by checking the version:
  - Install: `pip install django` | Check the version: `django-admin --version`

```
● ● ●                    📁 careerfoundry — -zsh — 115×21
[(achievement2-practice) evandanowitz@Evans-MBP careerfoundry % pip 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 backports.zoneinfo (from django)
   Using cached backports.zoneinfo-0.2.1-cp38-cp38-macosx_10_14_x86_64.whl.metadata (4.7 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 backports.zoneinfo-0.2.1-cp38-cp38-macosx_10_14_x86_64.whl (35 kB)
Using cached typing_extensions-4.12.2-py3-none-any.whl (37 kB)
Installing collected packages: typing-extensions, sqlparse, backports.zoneinfo, asgiref, django
Successfully installed asgiref-3.8.1 backports.zoneinfo-0.2.1 django-4.2.17 sqlparse-0.5.3 typing-extensions-4.12.2
[(achievement2-practice) evandanowitz@Evans-MBP careerfoundry % django-admin --version
4.2.17
(achievement2-practice) evandanowitz@Evans-MBP careerfoundry % ▊
```