2.1

1. Why Django is so popular among developers:

Django is popular among web developers for several reasons. Firstly, it follows the "batteries included" principle, providing a robust and comprehensive set of tools and functionalities out of the box, which speeds up development and reduces the need for external libraries. Secondly, Django emphasizes clean and reusable code, largely due to its adherence to the DRY (Don't Repeat Yourself) principle.

2. List 5 large companies that use Django. Specify what the company's product/service is & what how they use Django:
    a. Pinterest: a bookmarking platform where users can browse and save images of a multitude of interests. Django powers their backend infrastructure such as, user profiles, content categorization, and search functions.
    b. Instagram: a social media platform for sharing photos and videos, and send messages and chat. Django powers their backend for user authentication, content management, and data storage.
    c. Eventbrite: an online platform where users create, promote, and sell tickets for events of varying types. Django powers their backend infrastructure such as, event management and categorization, ticketing functions, and payment processing.
    d. Spotify: a music and podcast streaming platform. Django powers their backend by managing their user accounts, playlists, and content recommendations.
    e. Disqus: a commenting platform that integrates with websites and blogs. Django powers their backend by managing user authentication, comment moderation, and real-time updates for users across a multitude of websites.
3. For each scenario, explain if you would use Django (and, why or why not):

- You need to develop a web application with multiple users - Yes, Django would a great choice for this use case as it has a built-in user authentication and management system, so handling multiple users with varying permissions will be a breeze.

- You need fast development and the ability to make changes as you proceed - Yes, Django is great for fast and iterative development as it has extensive sets of pre-built components which can be employed through the Django Admin interface for rapid development; moreover, since it takes on such a modular design, scaling the software will be easy in the future as well.
- You need to build a very basic application, which doesn't require any database access or file operations - No, Django's best use cases are where a database and file operations are needed, given it has powerful ORMs for database access and interaction. Using something more lightweight would be better.
- You want to build an application from scratch and want a lot of control over how it works - Yes, Django provides a fair bit of autonomy which will allow the developer to define custom models, views, and templates.
- You're about to start working on a big project and are afraid of getting stuck and needing additional support - Yes, Django has a pretty large and active community, extensive documentation, and a plethora of third-party packages.

4. Python Version:

5. Virtual Environment Setup:

6. Django Version:

```
achievement2-practice ~ git:(main) ±67 (0.179s)
python3 -m django --version
4.2.5
```