Why Django is Popular Among Web Developers?

Django has elegant and clean design, which makes it easy to write maintainable and scalable code. It offers a high level of abstraction and follows the "batteries-included" philosophy, providing many built-in features and libraries for core web development tasks, such as authentication, security, database handling. Django has a large and active community, which means a lot of resources, documentation, and third-party packages are available to help developers.

Five Large Companies That Use Django?

1. **NASA** - The United States' space agency. NASA uses Django for its web applications, data management, and interactive visualizations, particularly in the field of space exploration and research.
2. **Pinterest** - A visual discovery and bookmarking platform. Pinterest relies on Django to power various aspects of its website, handling user data, content recommendation, and search functionality.
3. **Instagram** - A social media platform for sharing photos and videos. Instagram uses Django extensively for its backend services, including user management, content delivery, and more.
4. **Disqus** - A commenting platform used by many websites. Disqus employs Django for its core functionality, managing user comments and discussions on different websites.
5. **Dropbox** - A cloud-based file storage and synchronization service. Dropbox has used Django in various parts of its infrastructure, particularly for web-based administrative and internal tools.

Using Django in Different Scenarios?

Developing a web application with multiple users: Yes, Django is good for this situation because it gives strong user authentication and authorization tools, making it simple to handle many users with different levels of access.

Need for fast deployment and the ability to make changes: Django is good for fast creating and launching of projects because it has many helpful features included and a big group of extra tools. This makes it possible for developers to work on their projects quickly while still keeping the code quality high.

Building a very basic application without database access or file operations: Django may not be the best choice for extremely basic applications that do not require database or file operations.

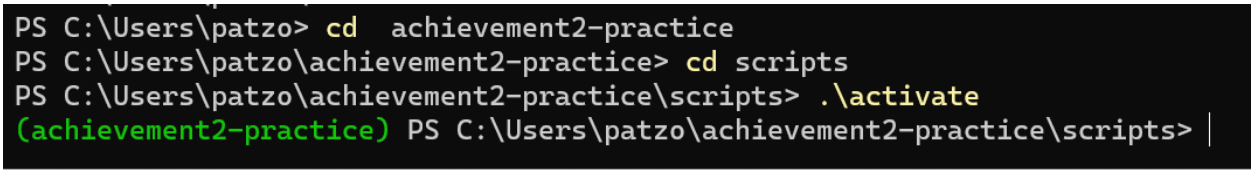
It comes with a lot of features that might not be necessary for such simple projects, and a micro-framework like Flask could be more appropriate.

Building an application from scratch with a lot of control: Django is flexible and allows for a high level of customization within the framework's conventions, making it a good choice when you want control over your application's functionality and architecture as long as you adhere to Django's way of doing things.

Concerns about getting stuck and needing additional support: Django's community and extensive documentation make it a good choice when you're concerned about needing support during a big project. There are many resources and experienced developers available to help troubleshoot issues and give guidance.

- Python's version:

- Virtual environment set up:



- Django version: