CareerFoundry

Python Task 2.1

Django is popular among web developers as it's very useful for fast development, secure deployment, and scalability. It comes with most essentials and allows developers to focus more on writing logic than web development when creating applications.

Five companies that use Django:

- Spotify: music streaming service. Spotify uses Django for user management, scaling, and needs for rapid development-handling user interactions.
- Pinterest: social media service for publishing information and content via pinboards. Pinterest uses Django to handle requests such as retrieving and saving pins, managing user accounts, and authentication.
- Dropbox: file hosting service. Dropbox uses Django for its ability to scale, as was the case with Dropbox early on. Dropbox uses Django to handle file management-letting users upload, view, and manage their files in the cloud.
- NASA: National Aeronautics and Space Administration, an independent agency of the US gov't responsible for space research. NASA uses Django to manage databases related to personnel, inventory, and mission-relevant resources. It also uses Django for media content delivery, such high-resolution images, videos, and streams of space events.
- Eventbrite: event management and ticketing website. Eventbrite uses Django for data management, scaling, and security, such as managing user data and transactions, fraud prevention, and the ability to grow quickly with demand for event tickets.

Scenarios of when to use Django or not:

-Develop a web application with multiple users: in this scenario, I would use Django as having multiple users would require keeping a database. Assuming this is the case, it would also likely require the need for secure data handling, which Django provides.

-Fast deployment and the ability to make changes as I proceed: in this scenario, I would use Django, as it is great for fast deployment and being based on MVT architecture, making changes in a snap is not a problem.

-Very basic application, with no requirement for database access or file operations: in this scenario I would not use Django, as the features necessary in this case do not require everything that comes with Django-much of it isn't necessary.

-Build an application from scratch and want a lot of control over how it works: in this scenario I would not use Django, as Django is written in the "Django way". In this, there is no control on fine details of a system. Django is resolute in how certain things must be done.

-A big project and are afraid of getting stuck and will need additional support: in this scenario, I would use Django, as it's great for "big" projects and there is a large support community of developers who will most likely be able to assist with issues one would experience. There is also excellent documentation that is well structured that contains tutorials, explanations, and examples.