Exercise 2.2: Django Project Set Up

Learning Goals

- Describe the basic structure of a Django project
- Summarize the difference between projects and apps
- Create a Django project and run it locally
- Create a superuser for a Django web application

Reflection Questions

1. *Suppose you're in an interview. The interviewer gives you their company's website as an example, asking you to convert the website and its different parts into Django terms. How would you proceed? For this question, you can think about your dream company and look at their website for reference.*

   Considering my dream companies to work for, I picked one of my other interests so as not to repeat software companies likely repeated in this question - Malbon Golf. When considering Malbon's website in Django terms, I would break it into the components:
   a. Models: 'Product' model for golf apparel/ equipment (name, price, description, images, category, stock). 'Category' model for product categories (clothing, accessories, equipment). 'User' model (using Django's built-in) for customer accounts. 'Order' model for purchases (user, products, total, date, status). 'Collection' model for seasonal/ product collections.
   b. Views: Product listing views (filter by category, search); Product detail view; Shopping cart view; Checkout view; User authentication view.
   c. Templates: Base template (header, footer, navigation); Product templates (list, detail, category pages); Cart template; Checkout template; User account templates.

2. *In your own words, describe the steps you would take to deploy a basic Django application locally on your system.*

   1. Set up the environment by creating a virtual environment, activating it, and installing Django.
   2. Create the Django project by creating the project and verifying the structure.
   3. Configure the database by running migrations, which creates the SQLite database and initial tables.
   4. Create a superuser by creating an admin user.
   5. Start the development server by running the server, accessing it on Port 8000 (by default), and admin panel by adding /admin.
   6. Create apps.
   7. Test application.

3. *Do some research about the Django admin site and write down how you'd use it during your web application development.*

Django Admin is a built-in interface that auto-generates management forms from your models, providing CRUD without custom code. During development, I would use it to create test data, manage content, verify model relationships, and test permissions. I could customize it with list_display, list_filter, and search_fields to streamline data management. It speeds up development by handling administrative tasks so I can focus on core application logic, and it's useful for testing model changes, validating data, and managing test data without writing custom views.