Exercise 2.3: Django Models

Learning Goals

- Discuss Django models, the “M” part of Django’s MVT architecture
- Create apps and models representing different parts of your web application
- Write and run automated tests

Reflection Questions

1. Do some research on Django models. In your own words, write down how Django models work and what their benefits are.
   Django models serve as a bridge between your Python code and the database, allowing you to define data structures using Python classes instead of SQL. Each model represents a table in the database, and the fields of the model correspond to the table's columns. This approach abstracts away the complexity of SQL queries, enabling developers to interact with the database in a more intuitive way.
   For example, instead of writing raw SQL to query data, you can use Django's ORM (Object-Relational Mapping) to perform database operations directly through Python. This includes creating, reading, updating, and deleting records. The ORM handles the translation of your Python code into SQL queries, ensuring that interactions with the database are secure and efficient.
   Moreover, Django models come with built-in methods and tools that simplify data validation, relationship management (such as foreign keys and many-to-many relationships), and even the automatic generation of database schema through migrations. This makes it easier to evolve your database schema over time without having to manually write migration scripts.
   In essence, Django models provide a high level of abstraction, allowing you to focus on the logic of your application rather than the details of database management.

2. In your own words, explain why it is crucial to write test cases from the beginning of a project. You can take an example project to explain your answer.
   It is crucial to write test cases from the beginning of the project because they save you time so you do not have to keep checking the app manually throughout the project and can focus on the code. They also prevent future problems from occurring in addition to checking for current problems. Additionally, they make code more reliable and strengthen code, so it is not broken by others or by self.