Exercise 2.4: Django Views and Templates

Learning Goals

- Summarize the process of creating views, templates, and URLs
- Explain how the "V" and "T" parts of MVT architecture work
- Create a frontend page for your web application

Reflection Questions

1. Do some research on Django views. In your own words, use an example to explain how Django views work.
    - A Django view is a Python function that takes a web request and returns a web response. These responses can be almost anything like the contents of a web page, a redirect, 404 errors, or an image.
    - An example of this would be a basic request for current date and time where first we would import the HttpResponse from django.http and then import datetime from the library. Next we'd build the view function and call it something like "current_datetime" that takes the HttpRequest object as its first parameter, typically named "request." The view then returns an HttpResponse object that contains the generated response.

2. Imagine you're working on a Django web development project, and you anticipate that you'll have to reuse lots of code in various parts of the project. In this scenario, will you use Django function-based views or class-based views, and why?
    - In this case, I think using CBVs would be the better choice due to its reusability and built-in features. CBVs allow for object-orient programming concepts such as inheritance where we can create base views with common functionality and reuse them across the application. For example, if we have a validation form, we can define it in a base class and inherit it from multiple views and reduce redundancy.
    - Additionally, builit-in features like ListView or CreateView that handle common tasks like displaying a list of objects will already have predefined behavior for these tasks and will save time and effort as they can be be extended or customized.

3. Read Django's documentation on the Django template language and make some notes on its basics.
    - A template is a text file that can generate any text-based format (HTML, XML, CSV, etc.).
    - Contains stat parts of the desired HTML outputs as well as special syntax describing how dynamic content will be inserted.
    - It contains variables which get replaced with values when the template is evaluated.
    - It contains tags, which control the logic of the template.