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.
   - Views are functions or classes in a Django app that process requests and return responses. Views handle logic and render the webpage

     Example:
     # views.py
     from django.http import HttpResponse

     def home(request): (This is the request)
         return HttpResponse("Hello, world! This is the home page.") (This is what is returned)

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?
   - CBVs are more suited for reuse over FBVs due to their OOP ways that allows inheritance and mixins. This can reduce duplication and improve maintainability. If there are several view handling similar tasks the code can be inherited and the logic shared.

3. Read Django’s documentation on the Django template language and make some notes on its basics.
   - Variables: In DTL, you can insert variables into the template using double curly braces.
   - Tags: Provide arbitrary logic into the rendering process. They are enclosed in { % %}.
   - Filters: Modify the display of variables. They are applied with a pipe | symbol.
   - Inheritance: Allows you to create a base template with common layout and extend it in other templates.