CAREERFOUNDRY

Python for Web Developers Learning Journal

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.

   Django views are the business logic that Django runs when a user accesses a URL. It takes in a web request and returns a web response. The response is the template which can be the HTML contents of a web page, a redirect, image, 404 error, etc.

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?

   Class-based views would be the best option in this case because they are easy to reuse and extend due to their class-based nature and avoid code duplication. Function-based views are more suited for to projects with multiple views that require customized logic.

3. Read Django's documentation on the Django template language and make some notes on its basics.

   Variables: Variables look like this: **{{ variable }}**. When the template engine encounters a variable, it evaluates that variable and replaces it with the result.
   Filters: You can modify variables for display by using **filters**. Filters can be chained and also take arguments.
   Tags: Tags look like this: **{% tag %}**. Tags are more complex than variables: Some create text in the output, some control flow by performing loops or logic, and some load external information into the template to be used by later variables.
   Template inheritance: Template inheritance allows you to build a base "skeleton" template that contains all the common elements of your site and defines **blocks** that child templates can override.