### Exercise 2.4: Django Views and Templates

1. Do some research on Django views. In your own words, use an example to explain how Django views work.

Django views are Python functions or class method that takes HTTP request, perform some business logic such as interacting with the database and returns HTTP response.

For instance, when a user access to the URL of Django web application, Django selects the correct view based on the URL to be loaded and corresponding template which defines the output structure will be displayed.

1. 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?

I would use CBVs in this case because even though CBVs are initially hard to write they are easy to reuse or extend them once created. It also helps to avoid code repetition.

1. Read Django’s documentation on the [Django template language](https://docs.djangoproject.com/en/3.2/ref/templates/language/#templates) and make some notes on its basics.

* A template is a text file and can generate any text-based format(HTML, XML etc.)
* “bar” in a template expression like **{{ foo.bar }}** will be interpreted as a literal string and not using the value of the variable “bar”
* Use a pipe (**|**) to apply a filter like **{{ name|lower }}**
* Tags examples:

**{% for … in %} {% endfor %}**

**{% if %} {% elif %} {% else %} {% endif %}**

* Template Inheritance:

**{% block% title} {% endblock %}** : where child templates can override

**{% extends “base.html” %}** : needs to be inserted in the child template