Reflection Questions Exercise 2.3

1. **Do some research on Django views. In your own words, use an example to explain how Django views work.**
   Views, at their core, are pieces of code that Django uses when a user accesses a URL. This code takes the form of a function that takes in a user request and returns some form of response. Both the request and response can range from simple to complex. It can be something as simple as showing the user an image to a fully fleshed out page that takes user input and returns information from an external database.
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 the case described above, I would use class-based views because they are easy to re-use and extend as the project grows. While they may take a bit longer to put together initially, being able to apply the same code over and over again will save the team more time going forward.
3. **Read Django's documentation on the Django template language and make some notes on its basics.**
   - There are four constructs: variables, tags, filters, comments
   - *Variables* – surrounded by `{{ }}` and used to output values from the context, for example, a logged-in user's name or date of birth that are store in variables like `first.name` and `birth.date`
   - *Tags* – provide a space for more arbitrary logic like loops, and are surrounded by `{% %}`
   - *Filters* – may take an argument, but in general they transform values of variables and tag arguments. Filters are a way to re-format existing data or variables (for example, re-formatting a date from MMDDYY to DDMMYYY
   - *Comments* – notes made by the developer that won't be rendered, surrounded by `{# #}`