Exercise 2.6: User Authentication in Django

Learning Goals

- Create authentication for your web application
- Use GET and POST methods
- Password protect your web application's views

Reflection Questions

1. In your own words, write down the importance of incorporating authentication into an application. You can take an example application to explain your answer.

    a. Authentication is important for user data protection and being able to differentiate between users, display pertaining data only to them. Unauthenticated users should often see only the public portion of any web application, authentication then ensures that every user can access their personal accounts with the respective data securely.

2. In your own words, explain the steps you should take to create a login for your Django web application.

    a. Create a new login view in the main app (create views.py unless already present)
    b. Import authenticate and login functions from Django.
    c. Defined the login view logic itself (validate form and validate user for logging in).
    d. Create a template - as it doesn't pertain to any particular app, it should be in the app root: src>templates>auth.
    e. Register the view in the main urls.py (project itself) as we don't need an app level registration here.
    f. Also, as the template is in a new location, the TEMPLATES constant should be updated in settings.py.

3. Look up the following three Django functions on Django's official documentation and/or other trusted sources and write a brief description of each.

Function	Description
authenticate()	The `authenticate()` function in Django is used to verify a user's credentials. It takes `username` and `password` as input and returns a `User` object if the credentials are valid. Otherwise, it returns `None`. This function is typically used during the login process to check if the entered details match a registered user.

redirect()	The `redirect()` function is used to send a user to a different URL. It can take a view name, a URL, or a model object as its argument, and Django will automatically handle the URL resolution. This function is commonly used after form submission or login to navigate users to another page.
include()	The `include()` function is used to include other URL configurations in your main `urls.py` file. This allows for better modularity and separation of concerns in large Django projects. It is typically used to include URL patterns from other apps.