Pre-Work: Before You Start Achievement 2

In the final part of the learning journal for Achievement 1, you were asked if there's anything—on reflection—that you'd keep in mind and do similarly or differently during Achievement 2. Think about these questions again:

- Was your study routine effective during Achievement 1? If not, what will you do differently during Achievement 2?
  Yes, my new study routine was actually a lot more effective than for my full stack immersion course! It helped me get through the achievement a lot faster.
- Reflect on your learning and project work for Achievement 1. What were you most proud of? How will you repeat or build on this in Achievement 2?
  I am just proud of the fact that I was able to learn Python and its syntax and how quickly I learned and got familiar with it.
- What difficulties did you encounter in the last Achievement? How did you deal with them? How could this experience prepare you for difficulties in Achievement 2?
  Some of the difficulties I encountered were due to the content given being outdated / old and the frameworks are being updated. I solved these by searching up the solutions from Google to stack overflow, and got the solutions that I needed.

Note down your answers and discuss them with your mentor in a call if you like.

Remember that can always refer to Exercise 1.4 of the Orientation course if you're not sure whom to reach out to for help and support.

Exercise 2.1: Getting Started with Django

Learning Goals

- Explain MVT architecture and compare it with MVC
- Summarize Django's benefits and drawbacks
- Install and get started with Django

Reflection Questions

1. Suppose you're a web developer in a company and need to decide if you'll use vanilla (plain) Python for a project, or a framework like Django instead. What are the advantages and drawbacks of each?

The advantages of using Python is that it has a simple easy to read syntax. Django itself helps a lot with quick prototyping, fast deployment, scalability, and more. With Django developers don't have to worry too much about the database setup and such, they can focus on more important feature implementation. Django and Python also have a large supportive community backing them if you ever run into an issue. The drawbacks however, is that Django has its own way of doing things that you must follow, also known as "The Django Way". Which decreases the amount of control a developer has over parts of their project, if you want more control over fine details and such, Django may not be the best option.

2. In your own words, what is the most significant advantage of Model View Template (MVT) architecture over Model View Controller (MVC) architecture?
   The advantage of MVT architecture over MVC is that with MVT you don't have to write the code to fetch the data from the database and then map it to a URL. That is all taken care of with the MVT components. Unlike with MVC, where you would have to take the time to code all of that.

3. Now that you've had an introduction to the Django framework, write down three goals you have for yourself and your learning process during this Achievement. You can reflect on the following questions if it helps:
   - What do you want to learn about Django?
   - What do you want to get out of this Achievement?
   - Where or what do you see yourself working on after you complete this Achievement?

I want to learn how Django works, how to set up a project with it, and what other functionalities I can create with it. With this achievement I want to have a fully working Django project that I can showcase in my portfolio. I want the knowledge that I've gained from this achievement to be enough for me to be able to work on other little projects made with Django.

Exercise 2.2: Django Project Set Up

Learning Goals

- Describe the basic structure of a Django project
- Summarize the difference between projects and apps
- Create a Django project and run it locally
- Create a superuser for a Django web application

Reflection Questions

1. Suppose you're in an interview. The interviewer gives you their company's website as an example, asking you to convert the website and its different parts into Django terms. How would you

proceed? For this question, you can think about your dream company and look at their website for reference.
(*Hint: In the Exercise, you saw the example of the CareerFoundry website in the Project and Apps section.*)

For a movie/shoe website like HBO Max the whole website is a project. Inside that project it will have individual modules that perform different tasks. Such as login to log into the website and get authenticated. Then you can see another module such as a profile to see your information. On the homepage for logged out users there is a Plan module, showing users the different plans available for their service.

2. In your own words, describe the steps you would take to deploy a basic Django application locally on your system.

   To deploy a basic Django application locally on my system I will first create and activate a virtual environment and install Django. Then I will create the django project with django-admin.exe startproject projectname. Then I will rename the project directory folder (in this case project-name) to src so there's no confusion with duplicate names. Next I will create the database by running the first migration with py manage.py migrate and then run the server to check for the success message in the browser. After that I need to create a superuser which will have access to everything such as all CRUD operations and the ability to manage other users. To do that I would use the command py manage.py createsuperuser and then follow the prompts. Afterwards, to confirm the creation I would run the server again and add /admin to the end of the url to log into the new super user that I just created. Once I'm finished I will use CTRL + C to stop running my server and deactivate to quit my virtual environment.

3. Do some research about the Django admin site and write down how you'd use it during your web application development.
   The Django admin site is a powerful, built-in tool that Django provides to facilitate database management tasks for your web application. It offers an out-of-the-box web interface to manage the content of your database models, without having to build CRUD (Create, Read, Update, Delete) interfaces from scratch.

Exercise 2.3: Django Models

Learning Goals

- Discuss Django models, the "M" part of Django's MVT architecture
- Create apps and models representing different parts of your web application
- Write and run automated tests

Reflection Questions

1. Do some research on Django models. In your own words, write down how Django models work and what their benefits are.

2. In your own words, explain why it is crucial to write test cases from the beginning of a project. You can take an example project to explain your answer.

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.

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?

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

Exercise 2.5: Django MVT Revisited

Learning Goals

- Add images to the model and display them on the frontend of your application
- Create complex views with access to the model
- Display records with views and templates

Reflection Questions

1. In your own words, explain Django static files and how Django handles them.

2. Look up the following two Django packages on Django's official documentation and/or other trusted sources. Write a brief description of each.

Package	Description
ListView	
DetailView	

3. You're now more than halfway through Achievement 2! Take a moment to reflect on your learning in the course so far. How is it going? What's something you're proud of so far? Is there something you're struggling with? What do you need more practice with? You can use these notes to guide your next mentor call.

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.

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

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()	
redirect()	
include()	

Exercise 2.7: Data Analysis and Visualization in Django

Learning Goals

- Work on elements of two-way communication like creating forms and buttons
- Implement search and visualization (reports/charts) features
- Use QuerySet API, DataFrames (with pandas), and plotting libraries (with matplotlib)

Reflection Questions

1. Consider your favorite website/application (you can also take CareerFoundry). Think about the various data that your favorite website/application collects. Write down how analyzing the collected data could help the website/application.

2. Read the Django official documentation on QuerySet API. Note down the different ways in which you can evaluate a QuerySet.

3. In the Exercise, you converted your QuerySet to DataFrame. Now do some research on the advantages and disadvantages of QuerySet and DataFrame, and explain the ways in which DataFrame is better for data processing.

Exercise 2.8: Deploying a Django Project

Learning Goals

- Enhance user experience and look and feel of your web application using CSS and JS
- Deploy your Django web application on a web server
- Curate project deliverables for your portfolio

Reflection Questions

1. Explain how you can use CSS and JavaScript in your Django web application.

2. In your own words, explain the steps you'd need to take to deploy your Django web application.

3. (Optional) Connect with a few Django web developers through LinkedIn or any other network. Ask them for their tips on creating a portfolio to showcase Python programming and Django skills. Think about which tips could help you improve your portfolio.

4. You've now finished Achievement 2 and, with it, the whole course! Take a moment to reflect on your learning:
    a. What went well during this Achievement?
    b. What's something you're proud of?
    c. What was the most challenging aspect of this Achievement?
    d. Did this Achievement meet your expectations? Did it give you the confidence to start working with your new Django skills?

Well done—you've now completed the Learning Journal for the whole course.