CAREERFOUNDRY

Python for Web Developers Learning Journal

Objective

We find that the students who do particularly well in our courses are those who practice metacognition. Metacognition is the art of thinking about thinking; developing a deeper understanding of your own

thought processes. With the help of this Learning Journal, you'll broaden your metacognitive knowledge and skills by reflecting on what you learn in this course.

Thanks to this Learning Journal, when you finish the course you'll have a complete and detailed record of your learning journey and progress over time. We really recommend that you take the time to complete this Journal; students do better in CF courses and in the working world as a result!

Directions

First complete the pre-work section before you start your course. Then, once you've begun learning, take time after each Exercise to return to this Journal and respond to the prompts.

There will be 3 to 5 prompts per Exercise, and we recommend spending about 10 to 15 minutes in total answering them. Don't overthink it—just write whatever comes to mind!

Also make sure that, once you've started filling this document in, you upload it as a deliverable on the platform. This is so that your mentor can also see your Journal and how you're progressing over time. Don't worry though—what you write here won't affect how you're graded for the Exercise tasks. The learning journal is mostly for you and your self-evaluation!

Pre-Work: Before You Start the Course

Reflection questions (to complete before your first mentor call)

1. What experiences have you had with coding and/or programming so far? What other experiences (programming-related or not) have you had that may help you as you progress through this course?

   I have learned the MERN stack, applying my knowledge to create full-stack applications like a pokedex app, moviemax app with react and angular versions, react native chat app, and a meet progressive web app. I come from a decade of retail experience, 8+ years of retail management, where I learned many transferrable character skills.

2. What do you know about Python already? What do you want to know?

   I know Python's a very readable language and can be used to complete complex data related tasks. I would like to get a better understanding of how Python is used on the backend to create and maintain scalable applications.

3. What challenges do you think may come up while you take this course? What will help you face them? Think of specific spaces, people, and times of day of week that might be favorable to your facing challenges and growing. Plan for how to solve challenges that arise.

   I'm sure there will be many times I get stuck trying to figure out how to do something and have to rely on finding the answers with the available tools. That being google, forums, blogs, articles, AI language models, and online communities I've joined where I can ask for assistance. I can also ask my tutor/mentor for expert guidance when needed.

Remember, you 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 1.1: Getting Started with Python

Learning Goals

- Summarize the uses and benefits of Python for web development
- Prepare your developer environment for programming with Python

Reflection Questions

1. In your own words, what is the difference between frontend and backend web development? If you were hired to work on backend programming for a web application, what kinds of operations would you be working on?

   Frontend development is what the end user interacts with. It is the client-side of an application. The backend is what goes on behind the scenes that makes it work on the server-side. If I were hired to work on backend programming, I would likely be working on server logic for the application to connect with the client. If the application needs any specific data I would connect the application with relevant APIs or databases. I would also be setting up/testing endpoints that the client would connect to, and handling HTTP responses.

2. Imagine you're working as a full-stack developer in the near future. Your team is asking for your advice on whether to use JavaScript or Python for a project, and you think Python would be the better choice. How would you explain the similarities and differences between the two languages to your team? Drawing from what you learned in this Exercise, what reasons would you give to convince your team that Python is the better option?
   (Hint: refer to the Exercise section "The Benefits of Developing with Python")

   Being that Python and JavaScript are both high-level scripting languages, they both execute line by line. They are dynamically typed, which allows variables to be set to any data type value without causing errors. Python has a leg-up over JavaScript in that it contains many common pre-installed web operations. This allows efficient development with minimal effort. Python is known for its readability. It was designed for quick deployment so that anyone with basic experience in coding can quickly grasp what's happening.

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

   1. I want to gain a better understanding of coding fundamentals.
   2. Become proficient in multiple coding languages (Python and JavaScript).
   3. Gather a deeper understanding of Backend development and how it connects to the Frontend.

Exercise 1.2: Data Types in Python

Learning Goals

- Explain variables and data types in Python
- Summarize the use of objects in Python
- Create a data structure for your Recipe app

Reflection Questions

1. Imagine you're having a conversation with a future colleague about whether to use the iPython Shell instead of Python's default shell. What reasons would you give to explain the benefits of using the iPython Shell over the default one?

   The iPython shell is much more user-friendly compared to the default shell. It comes with auto indentation, syntax highlighting, and auto-fill for methods.

2. Python has a host of different data types that allow you to store and organize information. List 4 examples of data types that Python recognizes, briefly define them, and indicate whether they are scalar or non-scalar.

Data type	Definition	Scalar or Non-Scalar?
Tuples	Immutable collections of elements that can only be overwritten not edited	Non-Scalar
Lists	Mutable ordered collections that can be edited	Non-Scalar
Dictionaries	Functioning similar to objects in JavaScript, they can be used to produce complex data structures using key value pairs	Non-Scalar
Strings	Immutable sequences of str data type characters that are easily queried and edited	Non-Scalar

3. A frequent question at job interviews for Python developers is: what is the difference between lists and tuples in Python? Write down how you would respond.

   The difference between tuples and lists is pretty simple. Once a tuple is created it cannot be changed, where as a list can be. This means tuples are slightly faster to access over lists with increased performance when querying. Tuples would be preferred for fixed collections of items, but if data needs to be modified a list is better suited.

4. In the task for this Exercise, you decided what you thought was the most suitable data structure for storing all the information for a recipe. Now, imagine you're creating a language-learning app that helps users memorize vocabulary through flashcards. Users can input vocabulary words, definitions, and their category (noun, verb, etc.) into the flashcards. They can then quiz themselves by flipping through the flashcards. Think about the necessary data types and what would be the most suitable data structure for this language-learning app. Between tuples, lists, and dictionaries, which would you choose? Think about their respective advantages and limitations, and where flexibility might be useful if you were to continue developing the language-learning app beyond vocabulary memorization.

   Keeping continued development in mind for the language-learning app, I would choose dictionaries and lists as the most suitable data structures. My reason for this is given the requirements of data, it would be easiest to organize the vocabulary words as individual dictionary items that include key value pairs. The keys would include the word, definition, and category with their respective values. Each dictionary could be included in a list of all vocabulary words, which could be easily queried and modified to add or remove words as needed.

Exercise 1.3: Functions and Other Operations in Python

Learning Goals

- Implement conditional statements in Python to determine program flow
- Use loops to reduce time and effort in Python programming
- Write functions to organize Python code

Reflection Questions

1. In this Exercise, you learned how to use **if-elif-else** statements to run different tasks based on conditions that you define. Now practice that skill by writing a script for a simple travel app using an **if-elif-else** statement for the following situation:

    - The script should ask the user where they want to travel.
    - The user's input should be checked for 3 different travel destinations that you define.
    - If the user's input is one of those 3 destinations, the following statement should be printed: "Enjoy your stay in _____!"
    - If the user's input is something other than the defined destinations, the following statement should be printed: "Oops, that destination is not currently available."

Write your script here. *(Hint: remember what you learned about indents!)*

```
available_destinations = ["London", "Japan", "Mexico"]

print("\nEnter your desired destination below\n")

destination = str(input("Destination: "))

if destination in available_destinations:
    print(f"Enjoy your stay in {destination}!")
else:
    print("Oops, that destination is not currently available.")
```

2. Imagine you're at a job interview for a Python developer role. The interviewer says "Explain logical operators in Python". Draft how you would respond.

    Logical operators in Python like "and", "or" and "not" return a boolean value allowing for true false comparisons on operations.

3. What are functions in Python? When and why are they useful?

Functions in Python are used to reduce redundant code by creating reusable components that can be called multiple times. A byproduct of this is more efficient and readable code.

4. In the section for Exercise 1 in this Learning Journal, you were asked in question 3 to set some goals for yourself while you complete this course. In preparation for your next mentor call, make some notes on how you've progressed towards your goals so far.

   Reflecting on the first 2 (of 3) goals that I set for myself, Python has helped me gather a better understanding of dealing with logical operators and writing conditional loops. It's very intuitive compared to JavaScript, which has many nuances or quirks to get familiar with. This has allowed me to focus more on the logic of the code.
   1. I want to gain a better understanding of coding fundamentals.
   2. Become proficient in multiple coding languages (Python and JavaScript).

Exercise 1.4: File Handling in Python

Learning Goals

- Use files to store and retrieve data in Python

Reflection Questions

1. Why is file storage important when you're using Python? What would happen if you didn't store local files?

   File storage is important because without it, data stored in-memory would be lost when the program is terminated.

2. In this Exercise you learned about the pickling process with the **`pickle.dump()`** method. What are pickles? In which situations would you choose to use pickles and why?

   The pickles module is used to read and convert python objects to and from binary format. This allows for efficient storage and loading.

3. In Python, what function do you use to find out which directory you're currently in? What if you wanted to change your current working directory?

   You can use the os module to get and change your current working directory. The os.getcwd() function is used to find out the current directory. The os.chdir() is used to change the current directory to a new path.

4. Imagine you're working on a Python script and are worried there may be an error in a block of code. How would you approach the situation to prevent the entire script from terminating due to an error?

Implement a try-except-else block, putting the execution code in try, printing an error message in except, and any conditional code to run if the try block completes successfully inside the else. This would prevent the entire script from terminating abruptly.

5. You're now more than halfway through Achievement 1! 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? Feel free to use these notes to guide your next mentor call.

   I'm feeling much more confident in finding solutions to logical problems and really understanding the code. Before I could understand the flow of the program, but not exactly what the code is doing at each step and required a lot of review and practice to get it. I'm able to grasp these concepts much faster in Python compared to when I learned them with JavaScript.

Exercise 1.5: Object-Oriented Programming in Python

Learning Goals

- Apply object-oriented programming concepts to your Recipe app

Reflection Questions

1. In your own words, what is object-oriented programming? What are the benefits of OOP?

   The basic concepts of OOP include classes and objects. It gives structure to data and allows for inheritance and polymorphism. OOP allows for more scalable and organized code bases, utilizing reusable code.

2. What are objects and classes in Python? Come up with a real-world example to illustrate how objects and classes work.

   Classes in Python are like a blueprint for objects. Classes define a structure that includes attributes and methods. For example, a Library class could contain attributes like a list of books, library name, and operating hours. Methods are defined within the class and can take specific attributes from the class initialization. Actions such as borrowing or returning a book, or checking book availability, would all be methods of the Library class.

3. In your own words, write brief explanations of the following OOP concepts; 100 to 200 words per method is fine.

Method	Description

Inheritance	Inheritance allows data (attributes) and methods (functions) to be passed from a parent class to a child class. Using the example from my practice tasks, a parent class could be Animal. It's child classes could be Dog and Cat. They can inherit attributes like their name but also have their own attributes and methods.
Polymorphism	Polymorphism allows for methods to be used and overwritten between multiple classes, which allows for reusable code. For example if I have a speak method in the Animal class, I can specify the method's attributes based on it's subclass. Such as a dog (Woof) or cat (Meow).
Operator Overloading	Allows for the overwriting of built in Python operator functions. The default behavior does not work for all use cases, so this allows us to write methods with modified operating logic when necessary.

Exercise 1.6: Connecting to Databases in Python

Learning Goals

- Create a MySQL database for your Recipe app

Reflection Questions

1. What are databases and what are the advantages of using them?

   Databases are structured systems for managing data. They allow data to be stored and retrieved so that it can persist beyond temporary local storage. Databases keep data organized which supports efficient management and security.

2. List 3 data types that can be used in MySQL and describe them briefly:

Data type	Definition
INT - Integer	An INT is a numeric data type used to store whole numbers (positive or negative). It cannot contain letters, decimals, or special characters.
VARCHAR – Variable length string	A VARCHAR() is a flexible data type used to store

	alphanumeric characters. It has a maximum character limit defined by the value passed in its parentheses.
PRIMARY KEY – Unique identifier	A PRIMARY KEY is a unique identifier used for each record in a table. Usually represented as an integer ID.

3. In what situations would SQLite be a better choice than MySQL?

   When working with very simple databases, SQLite is the better choice. Since SQLite is a portable version, it allows for faster setup with no installation required. In situations with more complex databases, MySQL would provide better functionality.

4. Think back to what you learned in the Immersion course. What do you think about the differences between JavaScript and Python as programming languages?

   JavaScript can leverage existing frameworks to be used on both frontend and backend development, whereas Python is mainly used for the backend. Python has greater application for data science and machine learning. JavaScript is optimized for web browsers.

5. Now that you're nearly at the end of Achievement 1, consider what you know about Python so far. What would you say are the limitations of Python as a programming language?

   While Python is very good for computational data analysis, it is not ideal for frontend or mobile development. Python's memory management can lead to higher memory usage, which is limiting for larger applications. Since browsers are designed to execute JavaScript, it would be less efficient to use Python. Python has a smaller ecosystem for web development libraries/frameworks compared to JavaScript.

Exercise 1.7: Finalizing Your Python Program

Learning Goals

- Interact with a database using an object-relational mapper
- Build your final command-line Recipe application

Reflection Questions

1. What is an Object Relational Mapper and what are the advantages of using one?

   An ORM is used to convert python to SQL syntax allowing interaction with databases to be much easier and more streamlined.

2. By this point, you've finished creating your Recipe app. How did it go? What's something in the app that you did well with? If you were to start over, what's something about your app that you would change or improve?

I enjoyed the way the exercises were structured, building up and putting all the components together. I struggled a lot with trying to perfect handling all edge cases that I could think of, causing me to spend too much time testing. In the end I learned a lot about handling loops for a better user experience and the methods involved in querying data from an SQL database.

3. Imagine you're at a job interview. You're asked what experience you have creating an app using Python. Taking your work for this Achievement as an example, draft how you would respond to this question.

   I learned a lot about the fundamentals of Python and it's relevance with handling data. Lots of trial and error has taught me how to effectively write classes, methods, and functions to query data from a MySQL database. I also learned how to use an ORM toolkit called SQLAlchemy, making the process much easier. I read documentation and updated deprecated syntax and constantly refactored my code to more robustly handle errors and user interfaces.

4. You've finished Achievement 1! Before moving on to Achievement 2, take a moment to reflect on your learning in the course so far:
   a. What went well during this Achievement?
      I feel confident in my understanding of Python fundamentals. The syntax was easy to learn and I was able to solve problems quickly once I had it down.
   b. What's something you're proud of?
      The fully functional Recipe CLI Program. Went through many different iterations in a short time but it was cool to see what I'm able to build.
   c. What was the most challenging aspect of this Achievement?
      Going over all the edge cases when taking in a user input. This will be something I want to review more deeply.
   d. Did this Achievement meet your expectations? Did it give you the confidence to start working with your new Python skills?
      It did meet my expectations and I look forward to the next Achievement!
   e. What's something you want to keep in mind to help you do your best in Achievement 2?
      Break down problems into more manageable steps.

Well done—you've now completed the Learning Journal for Achievement 1. As you'll have seen, a little metacognition can go a long way!

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?
  I need to block out time during my busy schedule to form a study habit.
- 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?
  Putting together a fully functional CLI with Python. I learned a lot building the project.
- 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?
  Finding solutions to problems that aren't the most "efficient". Discussing what I did to solve the problem and looking for improved alternatives will better prepare me for Achievement 2.

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?
   Depending on the needs, I'd choose Python for a small scale project that did not require any database operations. This would allow me full control without adding unnecessary bulk, increasing performance. I'd choose Django for a large scale project that required database operations, multiple users, and frequent updates. This would allow me to get a structured MVP out quickly and have robust data handling capabilities and security when scaling.

2. In your own words, what is the most significant advantage of Model View Template (MVT) architecture over Model View Controller (MVC) architecture?
   MVT switches the pattern for what the view and controller are responsible for. In MVT, the view receives user input, processes it, and returns a response rather than the controller in MVC. In MVT, the template renders UI elements rather than the view in MVC.

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?

    Build a consistent study habit using tested habit forming principles.

    Be able to understand and participate in backend discussions as a frontend developer.

    Use my newfound knowledge to contribute to a meaningful project.

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.*)
   Considering instacart's website, in Django terms the whole website can be regarded as a project. The individual modules like search, login, location, navigation, individual retailers, cart, and more can be considered apps. Each has its own particular function and can be reused across multiple projects. For example, the search app persists across multiple areas of the project. Following the DRY method this logic can be reused for different retailer products.

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

I would create a virtual environment and use pip to install Django. Using the django-admin commands I would create a project directory for the app, migrate the project to create the database schema, and run the server. I would then create a superuser login to access the admin dashboard for handling CRUD operations.

3. Do some research about the Django admin site and write down how you'd use it during your web application development.
   Django automatically creates an admin site that makes it easy to perform and test CRUD operations on models. It can also be useful for managing data in production. This tool would save me a lot of time as a developer, not having to create an admin dashboard to view, modify, and test relevant information.

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.
   Django models are represented as a class that correspond to a table (project) in the database. Models use ORM to convert Python objects into database queries, which allows interaction with the database and mitigates the risk of SQL injection. By keeping the logic for each model separate, it keeps projects organized, allowing for easier customization, testing, and debugging.

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.
   Writing tests from the start of a project gives you a better understanding of the apps core functionality. It acts as a blueprint, where the tests cover what the code should do. It also allows bugs to be caught early, ensuring code quality. When a project scales in complexity, it speeds up debugging and helps prevent new features breaking existing functionality.

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 render templates with http requests, utilizing Django’s built in render function. In the views.py file of my recipes app, I defined a home view that takes a request parameter and returns a custom recipes_home.html template. This links the view (home) with template (html), which can then be mapped to a URL.

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?
   I would use Django’s class-based views because they are designed to reuse code via inheritance. Depending on the complexity of the project, CBV’s offer better ways to handle common patterns in methods.

3. Read Django’s documentation on the Django template language and make some notes on its basics.
   Django’s template language has many methods to display templates with differing complexities. The template itself is a parent text file which can include variables with dot notation, filters for variable modification, tags for loops and ‘if-else’ statements, single-line and multi-line comments, template inheritance to allow overriding content with child templates, and html escaping to avoid malicious exploits.

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.
   Django static files are used by developers to reference files that are not generated dynamically, such as CSS, image stores and others for use within the app. Django understands the folder structure of static files so they can be easily linked to templates using Django syntax during development.

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	Used to display a list of objects from a queryset, typically all objects of a specific model (Recipe in my case).
DetailView	Used to display details of a single object. Fetches the object based on its primary key provided in the URL.

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.
   I took a break for the holidays to spend time with my family. Instead of focusing on my project, my spare time was spent solving some leetcode questions as this is a weak area of mine. My focus was on fundamental problem solving and translating that into programming. When I came back to my project it took me a while to get back into it, but once I did it all came back quickly. I feel like Django finally clicked for me. At first it seemed unnecessarily complex with its quirks but after seeing my project come together, I understand how it can be used to speed up development.

Frontend Inspirations

- https://www.americastestkitchen.com/
  I like the welcoming landing page, with a large hero image and slogan present. It also has an immediate call to action, search bar, and recipe development process.
- https://www.hellofresh.com/recipes
  I like the scrollable recipe categories and popular meals immediately displayed. The layout is not too cluttered like other recipe apps, making it easier to browse.
- https://www.budgetbytes.com/
  I like the grid layout, separating different featured recipes between category browsing. They make it easy to see an average cost per recipe, as the name suggests.

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.
   Without authentication, it would be hard to discern what information should be private or publicly available to your users. The ability to manage user-specific data requires some form of authentication, which is common-place in many apps. This allows for better user experience and security.

2. In your own words, explain the steps you should take to create a login for your Django web application.
   First initialize a login view and template structure. Then include necessary imports for built in Django auth and forms in the applications views.py file. Create a login view that takes an HTTP request parameter and check if input from the form is valid, redirecting accordingly. Add a url pattern for login to the urls.py file. Lastly, configure a logout view and url pattern so users can logout.

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()	Verifies users credentials against the database, returning either a user object or None if invalid.
redirect()	Sends a user to the desired page using a specified url or view name.
include()	Allows the reference of other URL configurations, keeping each app's urls self-contained. This allows for better organization and scalability.

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.