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 just completed the Full-Stack Development course at Career Foundry, and also I worked for over 10 years for a website agency doing first to second level technical support, as well as different tasks that included editing CSS, building websites using a CMS, work on SEO, …

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

   I know it's primarily a backend language and used a lot in AI, also that it is easy to read, has a rich ecosystem and can be used for a wide range of projects. I would like to learn some practical ways to use it alongside my other full-stack skills acquired so far, to use it on real-world project, and maybe expand my skills to projects I didn't think about yet (I like the idea that you can build games with it for instance, even though I'm not really a gamer it is definitely interesting).

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 have ADHD so at times clicking with a logic can be challenging and I need to take a step back to get something properly. Timing maybe also, but it seems manageable looking at the dashboard, more than the previous course which really took all of my time for the past months until late each day.

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 has to do with the user experience, the interface that you see. The backend usually handles the tasks that are done when interacting with the frontend. If hired to work on backend, I would be working on creating functions and algorithms to perform tasks, as well as making sure that everything works properly with the frontend.

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")

   Both are dynamic languages and are using easily understandable keywords to perform tasks, the difference lies in the facts that Python is very easy to read and understand, but is also great for automating tasks, and has a lot of out-of-the-box essentials (routing, form validation, PW encryption, …) which save time. Overall, deployment would be faster, and if the server-side is where most of the magic happens, then it would be a wise choice.

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?

   - learn enough Python to be able to create my own applications organically
   - be able to compare it with other languages to make the best choice for each project
   - I could see myself use Python in the future for projects needing automated tasks

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?

   iPython Shell has a much more intuitive UI (including numbered input and output prompts) and is more versatile. Lines of code are displayed using color which makes them clearer to read, it also offers code completion assistance for any object when we start typing the first characters and press the tab key. You also don't have to worry about Python's indentation syntax, which you need to look for in the default shell.

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?
Int	Integers, basically a whole number (no decimal) that can be either positive or negative, from 0 to infinity	Scalar
Bool	A statement that is either True or False.	Scalar

Tuples	They are linear arrays that can store multiple values of any type.	Non-scalar
Dictionaries	An unordered set of items, storing objects pairing keys and values together. It is useful when we need to store several objects of different unrelated categories under one big category (for instance, a student's ratings in different classes).	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.

   A list is an ordered collection of items, and can be modified. A tuples is an unordered collection of items and cannot be modified after being created. The syntax is also different, lists use square brackets, tuples use parentheses.

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 beyond vocabulary memorization.

   In that case, I would again choose Dictionaries. We need to be able to edit / add / delete data, so it needs to be non-scalar. It's also easy to search for something, which would be essential in a learning app. Since we want to have different types of inputs (word, definition, category) tuples are not an option. If the app happened to be expending to new features later, that data structure of a dictionary would again allow flexibility to add, for instance, several translations of that word. A list could possibly work for the base idea of the app but is less flexible if scaling up and expending beyond memorization, also in term of searching as you need to use indexing which can get more complicated as you add categories.

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!)*

```
destination = input("Where do you want to travel? ")

if destination in ["Dubai", "Paris", "Mumbai"]:
    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 are keywords used to combine or modify statements (like inside an if statement) in order to perform an operation by adding/detailing conditions. E.g. : or, and, not.

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

   Functions are reusable blocks containing sets of instructions that tells python what to do with the code. They are used when you want to perform a particular task, and are useful when a task is repetitive as it allows automating this task, for instance looping through a sequence of items to perform this task on each of them, instead of rewriting all of the code for each item. It also makes the code cleaner and easier to read.

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.

   So far I became more familiar with Python so I can already see the difference with other languages and why I would go for Python instead (simplicity for one). I also started thinking of possible projects (one would be an informative app where you enter a list of criterias and get answers/directions based on your choices) to build using Python, though I am still exploring options as I am learning more.

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?

   It allows data to persist after the program stops running, so you can use it again. Without it, using the shell it only exists in memory.

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?

   Pickles are complex data objects converted into binary files, without having to convert any of the different structures into a string first. They are perfect in a testing situation (or for small projects), avoiding to use a database.

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 use os.getcwd() to find out which directory you are currently in, and os.chdir() to switch to another one.

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?

   I would add safety features in the form of a try/except/else block.

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 love it so far! Probably the most intuitive language so far (but I guess being more familiars with others helps as well) which makes it very fun and some of the features are really interesting. I'm definitely proud of how smooth my understanding is at this point, though I definitely need to practice everything more to really keep it in mind and use it more organically. I don't feel like a particular topic is a struggle at this point, rather my own ADHD mind thinking too much at times which causes hesitation (order of things / when to place something in or out of a function, how to use an argument

or another, …) but it usually goes away when I take a break to think clearly, or do a little bit of research on the logic.

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?

   OOP is a way of organizing code using objects (opposed to functioning programming which revolves around functions). Everything is like a real-world object/box containing attributes (data) and methods (actions), where you can define structures for each object that keeps the datas and related actions together. The benefits are that you can reuse code very easily using inheritance, and it keeps related data and methods together making it very clear and organized.

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

   A class is a template defining which types of data and methods will be held in the actual objects. Objects are the actual concrete examples of what the template is referring to, they have their own attributes and values defined by the rules described in the 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 in Python is the capacity to pass on methods to another class, using a one-way system of parentclass and subclass. The subclass can call on the parent class (and only in that direction, as in real-life inheritance only goes from parent to child) when it is defined, using *Class NewClass(OldClass)* to access the methods defined in the parent class without having to rewrite each method again. It is important to notice that if you define a new similarly named method in a new subclass, it will have priority and will be used instead of the parent class' method. Also, if new attributes have to be added to the class in the __init__() initialization method, a new init method needs to be created in the new class, before calling the parent's init method inside of it.**
Polymorphism	**Polymorphism is when different data or methods have the same name but represent/do different things. Since they are all defined in their own space, the same names can be used without interfering with each other. For**

	instance if I have two different classes using both a method with the same name *method()*, I could instruct each of these methods to give different inputs, and it wouldn't cause any issue as long as they are defined separately (same principle as local functions not being available on a global scale in functional programming).
Operator Overloading	Operator overloading is the process of defining custom methods to use operators on different types of data. Indeed, an operator will react differently on different types (for instance, using the + operator on a string will result in concatenating both string together, while using it on integers will give you the sum of these integers). In a custom class, operators are not directly supported and their way to function needs to be defined in special methods like __add__() so they know how to behave.

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 collections of data stored together following a defined structure. Their advantages are that they allow to store, organize, and large amounts of data, also the fact that they are structured makes them very convenient to use as access and modification to specific data is easier.

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

Data type	Definition
VARCHAR	**Stores string values with a defined limited length (declared as *VARCHAR(n)*, n being the maximal number of characters allowed)**
FLOAT	**Stores decimal numbers**
INT	**Stores regular integers**

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

   If you are in testing phase or have a small simple project that will basically only use a single process. Also if you are not familiar with SQL but wants a stepping stone to get familiar with it.

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?

Python is much more adapted to OOP, and is also much more intuitive. It is more backend oriented, while JS is more of a full-stack language adapted for web apps/interactive websites, it can run directly in the browser without to install anything too.

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?

It cannot run natively in the browser, takes a lot of memory to run and is slower as it's not compiled. It's dynamically typed so type errors are only shown when you run the program which can be a pain to debug. Also, it's not really tailored for mobile development.

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 a tool that helps make database conversions easier by transforming the content and structure of the database into classes and objects, so we don't have to worry about SQL syntax. It makes it easier to switch database management systems.

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?

It went great ! I find Python very intuitive (structure, ways to pass attributes, logic, …), I'm getting pretty comfortable with loops, if-elif or if-else statements, normalization, … Something I would improve would probably be conversion from one type of data to another going through different steps (an output as an object needing to be turned into a list, string and so on to be reused later), I get it but it still isn't fully intuitive yet and I need to double check it when using it.

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've built a Command Line recipe app (at first without ORM, using raw SQL queries, pickles and connectors, then with ORM – SQLAlchemy - using PyMySQL as a DB) so I'm familiar with the Python and OOP structure, including handling errors, well-commenting my code with docstrings and regular comments for clarity, managing connections/sessions, good practices in general.

I can use the MySQL command line to check the details of my tables (structure or content), update it and so on.

The app allows a user to create a new recipe, view all recipes available at once, search recipes by user-defined ingredients (displaying first an entire list of ingredients used throughout all the recipes, all normalized through creation or edit methods), edit a recipe by selecting the different fields to edit before adding new values, delete a recipe or quit the program – again all including error handling like type errors, invalid inputs, empty fields, …

Overall it gave me experience building a full CRUD-App on Python that handles data persistence and maintains a clean code.

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?
    Most of it to be honest, instructions were clear and Python feels easier (at this level at least) when coming out of other languages in my opinion. I like OOP a lot and everything is much clearer than functional programming when it comes to quick readability.
    b. What's something you're proud of?
    How smooth this achievement went ! I still took my time to make sure I did things right but there wasn't really a moment where I got stuck so it gives me confidence for sure.
    c. What was the most challenging aspect of this Achievement?
    I'd say all the little details regarding outputs and how they will either display or be used later needing conversions. It's not very challenging but requires attention and, at times, to pause for a second and take a step back to see the bigger picture.
    d. Did this Achievement meet your expectations? Did it give you the confidence to start working with your new Python skills?
    It absolutely did. Python is the most comfortable I have been in my journey so far, though I am curious now about more complicated aspects like AI use or routing, but I feel like I start with solid bases which is as good as it gets.
    e. What's something you want to keep in mind to help you do your best in Achievement 2?
    To relax and take the time to take a step back when needed. Python is pretty clear and even when it gets a little more "wordy", taking a breath to look at it is usually the key to get it.

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?
- 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?
- 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?

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?

2. In your own words, what is the most significant advantage of Model View Template (MVT) architecture over Model View Controller (MVC) architecture?

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?

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.*)

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

3. Do some research about the Django admin site and write down how you'd use it during your web application development.

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

- 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.