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 had very little experience with coding and programming before this course, other than the full-stack immersion course I just completed through careerFoundry. However, I did learn a lot from that course and I think It will help me progress more confidently through this course.

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

- I know that Python is a popular coding language used for all kinds of tasks. I think i've heard that it is particularly popular in more data-driven fields. I have also heard that it has pretty simple, easy to understand syntax. I want to know what makes Python unique and how I can use its strengths in web design.

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.

- The biggest challenge I faced in the full-stack immersion course was being introduced to large blocks of code that I didn't understand. When I came upon a situation like that my instinct was to just follow the directions even though I didn't truly understand what they were telling me to do. This would always come back to haunt me later as I inevitably got stuck down the line. I think the best thing I can do to face this problem and most of the problems I might face is to slow down and ensure that I truly understand each step before moving on.

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?

- Front end focuses primarily on what the user sees when they open their browser. It generates items on screen, serves data pulled from the back end, and takes in information from the user like clicks and form submissions. The back end stores information that the front end can pull upon as needed. If I were hired to work on the backend of an application I would be working on things like database management, building API's, and implementing security/authorization.

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")
- JavaScript and Python are both popular languages that can be used to build robust applications. They are both high-level scripting languages with variable types that can be used in a variety of different ways. One advantage of JavaScript is that it is the language of the web and is what the front end code will be based on. Using a common coding language across the front and back-end of our application could help keep things organized. However, I would ultimately recommend Python because it has easy to understand syntax which could make it easier for clients to make use of the code we write. Its simplicity could also make it easier for large teams to collaborate and share code. It also comes with a lot of essential features like package management, routing, form handling and other common web operations.

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?
- Goal 1: To better understand what makes a high-quality backend, no matter the language.
- Goal 2: To understand the basics of Python and how to use it for purposes beyond just web development.
- Goal 3: To learn more about different Python frameworks like Django.

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 has many advantages over the default including the addition of text highlighting, tab-completion, and being able to automatically print to the screen. Text-highlighting makes it easier to read code and spot mistakes, tab-completion can speed up your coding and display available options, and the ability to print to screen without having to type out print(...) everytime helps speed up the process as well.
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	Object that represents an integer like 1, 2, or 3.	Non-Scalar
float	Object that represents decimal numbers like 3.14	Non-Scalar
list	Represents a series of data. Data can be of any type.	Scalar
dictionary	Represents a series of Key-Value Pairs of any type.	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 biggest difference between tuples and lists is that lists are mutable while tuples are not. Therefore tuples may be more useful in situations where you want to store data that is unlikely to change while lists are more useful in situations where you want to store data that is likely to change.
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.
   In this situation I would use a dictionary for the overall data-structure. This is useful because it allows users to store information about each word like its definition and category in key-value pairs. Dictionaries are mutable so users could easily add or remove vocabulary 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!)*

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

If destination == "Japan":
   print("Enjoy your stay in Japan!")
elif destination == "Finland":
   print("Enjoy your stay in Finland!")
elif destination == "Argentina":
   print("Enjoy your stay in Argentina!")
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 are useful situations where some result or outcome is determined by multiple conditions. For instance, in the US one must be over the age of 16 AND hold a valid driver's license in order to be able to drive. In Python the logical operator 'and' would make sure that both of these conditions are true before returning True.

3. What are functions in Python? When and why are they useful?
   Functions are reusable pieces of code that perform a particular function. They are useful when you need to perform certain operations multiple times throughout your code because they prevent you from having to type out the function for each individual use. Instead you can just recall the function whenever you need it to keep code cleaner and speed up the coding process.

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.
   Two of my goals were to learn more about Python frameworks like Django and to better understand what makes a high-quality back-end. The course hasn't covered these two goals in depth yet however my third goal is to understand the basics of Python and how to use it for purposes beyond web design. I feel I have already made a lot of progress in this area. I now have a good understanding of Python data types, for/while loops, if-elif-else statements and how to define and use functions. These skills are important no matter what I might use Python for, web design or otherwise. I'm glad I got to practice with them over the course of the last couple of exercises and I'm looking forward to honing those skills over the rest of this course.

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 it allows you to store data that you use in one script and carry it over to another. If you did not store that data locally, the data would be lost once the script ended.

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 a way of serializing and deserializing data. Serializing turns data into binary which is useful for storing more complex data structures like dictionaries. In this state, the information is stored in such a way that a machine can read it, but humans cannot. You can then use pickle.load() to turn the binary data back into readable text when you need it.

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.getcwd() function to find out your current directory. You can use the os.chdir() to change your current directory.

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?
   In this situation I would use try-except-else-finally. With this text I could attempt to run a bit of code inside the try block. If it runs successfully I could define the next steps inside of an else block and the code could continue like normal. However, I could also define multiple except parameters to catch any possible errors, prevent the code from continuing and possibly provide feedback as to how/why the code failed.

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 have fallen behind a little bit due to sickness but not so much that I can't catch up with a little work so I am still confident in my ability to complete this course. I am proud that I have completed all of the extra code-practices so far and I think it has really helped me to gain a better understanding of the material. The thing I am struggling with most at this moment is probably serializing and deserializing data with pickles. It is definitely a bit more complicated than the basic data manipulation I was doing in the first few tasks. However it is also a new topic and something I am confident I can improve upon with practice.

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?
   In object oriented programming, programs are organized around data and objects. One can define any number of objects, each with unique attributes. We can then use these objects in conjunction to build a larger program. OOP has some great advantages like the ability to reuse code through inheritance and it's modular nature allows for easier scaling as well.

2. What are objects and classes in Python? Come up with a real-world example to illustrate how objects and classes work.
   Classes are archetypes that define the structure of an object. In a real-world example a class might be 'Human'. 'Human' defines a general structure for what an object in that class could be. An object of the class 'Human' would have a name, age, height, weight, etc. An object is a specific instance of a class, like a person named John who is age 30, 6 feet 2 inches, and 180 pounds.

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

Method	Description
Inheritance	Through inheritance you can create subclasses that inherit the attributes of their parent class but allow you to add new attributes specific to the new class.
Polymorphism	Polymorphism allows you to define methods of the same name across different classes without them contradicting each other.
Operator Overloading	When creating a new class we must define how operators like +, -, >, <, etc will function for that class. This is called operator overloading.

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 tools for storing data in a structured way. They are useful because data stored in a database will not disappear once your script finishes running and can be accessed from multiple scripts. Databases also allow you to update and alter values within as your data changes overtime.
2. List 3 data types that can be used in MySQL and describe them briefly:

Data type	Definition
int	Int is for whole numbers like 1, 2, 3.. etc, just like python.
varchar()	Varcha() allows you to input strings of various lengths
float	Float is for numbers with decimal values like 3.14

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

SQLite is great for smaller applications that do not need to handle a whole lot of data. You don't need to install or set up SQLite so its also great for quick builds like prototypes or testing.

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?
   So far I find Python a bit more straightforward and easy to understand, though perhaps that is because I spent so much time with javascript before learning python. Python requires less special characters in its syntax and relies mainly on indentation, parenthesis, and colons for the most part. Python also seems more geared toward OOP than javascript.

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?
   One thing that has tripped me up a little bit is that Python will not allow you to initialize a variable without defining it which I am used to doing in javascript. I feel like this makes my code a little more verbose sometimes than it would be in javascript. One other limitation when it comes to web design is that Python is not the language of the web like javascript is. This has not been too big of an issue so far as this task focused on the back end but I imagine that is something I will have to work around when learning front end development with python.

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?
   ORMs are tools that translate information between OOPs and databases. They are useful because they allow you to interact with the database directly through your OOP application and save time since you don't have to manually translate data between the database and your application.

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 thought it went well. I certainly hit a few bumps along the way but I think I learned from those mistakes. I think something I did well with, or at least improved with, was using all the types of loops. I felt like I could structure them and type them out pretty naturally, almost like a real spoken language. One thing I would change would be to better implement DRY principles. I feel like there is some code that I could have used more efficiently and reused in more places.

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 have experience implementing the back-end of a web application. I used python and SQLAlchemy to communicate with a mysql database and created a command line interface through which a user could make queries to and retrieve data from the database. In the course of creating this app I gained a lot of experience in using basic Python operations and developing Object Oriented Programs.

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 think most of the basics of learning Python went well. I feel like I have a good understanding of data types, operators, functions and OOP in python. I also feel like I understand what Python's strengths and weaknesses are.
   b. What's something you're proud of?
      The thing I am most proud of is my improved familiarity with programming in general. What I mean is that writing code is beginning to feel more like second nature rather than a completely foreign language.
   c. What was the most challenging aspect of this Achievement?
      The most challenging part for me was learning how to interact with mySQL through Python. SQLAlchemy certainly made it easier but mySQL and Python operate so differently that translating between them could be confusing at times.
   d. Did this Achievement meet your expectations? Did it give you the confidence to start working with your new Python skills?
      It definitely met my expectations. I am confident in my basic Python skills and am ready to try out Django.
   e. What's something you want to keep in mind to help you do your best in Achievement 2?
      I think the most important thing for me to remember is to make sure to work at least a little bit everyday. I've learned a lot already and I need to keep those skills sharp going into the next achievement.

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 would say it was mildly effective. I think I learned a lot but I need to pick up the pace in order to stay on schedule for this course.

- 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 most proud of how well I was able to translate what I learned from JavaScript over to Python. They are quite different but I didn't have too much trouble repurposing those skills. I will build upon this achievement by taking the basics of Python that I have learned and applying them to a web framework in Django.
- 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?
  The biggest difficulty I faced was in translating data between the mySQL database and my Python application. SQLAlchemy helped a lot but the differences between how each system operates made things a bit tricky. However, the practice I got in this regard will be a great help in achievement too as I will need to create a front-end that communicates with that back end.

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?
   Using Django allows you to get a web-application up and running quickly as it comes with a lot of built-in features like authentication and security. Django applications can also scale easily thanks to its modular design. However, in order to use the features you must give up some control over how your application works because Django requires certain things to be done certain ways. On the other hand, using vanilla Python affords more freedom to you as a developer but takes more time to develop and would be harder to scale because you wouldn't be able to use all of the features that Django has right out of the box.

2. In your own words, what is the most significant advantage of Model View Template (MVT) architecture over Model View Controller (MVC) architecture?
   MVT is more “batteries-included” architecture. This means there are a lot of common features that can be used straight away without having to build any custom functionality
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:
   - Goal 1 - To improve my intuitive understanding of Python so that writing it will be more natural
   - Goal 2 - To be able to build a full-stack application with the Django framework.
   - Goal 3 - To improve my coding skills specifically related to creating databases and making queries to them.

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.*)
   I would look at any distinct functions that I could find on the website. For instance on this google docs page the function bar at the top may be an app while the main text-input area may be another app. These apps could be reused in other similar applications like google slides.

2. In your own words, describe the steps you would take to deploy a basic Django application locally on your system.
   First you must set up a virtual environment in which Django can run and then install Django there. Then using django-admin you can initialize the django project, run migrations, set up a superuser and run a server so you can access the application from the web.

3. Do some research about the Django admin site and write down how you'd use it during your web application development.
   The admin site has many useful features that I could use throughout development. I could use the permissions and security tools to determine who has access to my app and how much access they receive. I can also use the admin site to register and keep track of any models that I might use in my project.

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 translate python classes into tables in a database. This is useful because it allows developers to write their code in python and let Django handle translating it into SQL instead of having to hardcode everything into an SQL database ourselves.

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 test cases early can act kind of like a guide, as you are defining key functions of your app which you hope to achieve. They also of course help you to make sure that your application is working as intended. Finally, robust testing can help you generate documentation for your app.

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 work by taking a request from the web application through the url and returning a template that corresponds to that request. A developer maps their templates to different url paths to ensure that the user receives the view that they have requested.

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 this scenario I would use class-based views. Although function-based views can be easier to read and implement, at least initially, class-based views are easier to reuse which makes them a better fit for this scenario even though they may be a bit trickier to implement at first.

3. Read Django's documentation on the Django template language and make some notes on its basics.
   - Not simply python embedded into HTML
   - Tags control the logic of the template
   - Can be used for any text-based format
   - Use pipes (|) to apply filters
   - Tags are more complicated than variables and their function can vary slightly
   - Comment with #
   - Templates can be inherited with the extends tag

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.