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? Python is my programming mothertongue! I have used it for data science, machine learning and scientific programming.

2. What do you know about Python already? What do you want to know? It is a versatile language, used from machine learning/data engineering to backend web development.

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 think the most challenging things I encounter will not be the coding itself, but all the technical issues that arise like configuring my virtual environment, deploying my backend, etc. Which I have faced in the past.

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?
    a. Citing: Ex. 1.1 Tutorial for reference, my answers are:
    b. Frontend is the interface that users see and interact with, while backend is the collection of operations using servers and APIs.
    c. The operations I would be working on would be interacting with databases, files, servers, compiling web pages, processing user requests and data, communicating with server administrators.

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")

    It is similar to Javascript because it is OOP.

It is different because Python doesn't require you to declare a variable before using it, whereas Javascript does. .

It would be better because it is readable, efficient and has strong community support.

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?
    a. Want to develop Django skills
    b. Want to successfully deploy an API in Django
    c. Want to know capabilities in plain old Python I'd have never known about.

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? One benefit is that ipython allows you to access your code history within the 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?
list	Mutable ordered sequence	non
dictionary	Unordered set of items stored as key value pairs	non
string	immutable array of characters	non
tuple	linear arrays that can store multiple values of any type	non

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. List is mutable, tuple is not.
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. I would choose dictionaries because this lets me store different attributes as keys containing values of different data types, whereas tuples and lists don't. This is a fundamental advantage.

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!)*

```
a = input("where do you want to travel")
b = ['a','b','c']
If a in b:
    print("Enjoy your stay in ", a, '!')
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. They allow you to check if a condition is true or false.

3. What are functions in Python? When and why are they useful? They are a block of statements that run a task. They are reusable and can be used when the same task is performed multiple times on different values.

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. I practiced using functions, python shell, data structures and types. I am getting closer to starting work on building a Django API.

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 is important because you need to save files that you write to and need to read. If you didn't store local files you would have to write all the data you wanted to again.

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 structures that convert complex data files into a packaged stream of bytes. You use them when you need to serialize and de-serialize a Python object structure and need to save it on to disk and reuse it.

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

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? Use exception handling with try, except, else, finally to throw an error.

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 am honing skills that I already know and am getting more practice and understanding of concepts I have used less or didn't previously understand as well.

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 form of programming that organizes software around data or objects rather than functions. Its benefits are that you can reuse data structures or objects and allows programs to be broken into smaller parts so it can be debugged easily. We can use inheritance so objects or classes can be reused while extending its features.

2. What are objects and classes in Python? Come up with a real-world example to illustrate how objects and classes work. Objects are data structures with unique attributes and behavior. Example is: a person. It has attributes like name, weight, height, etc.. Class is an organization of data so elements can be reused. Example is: a car. You can create instances of a car class like Toyota, Mercedes, etc.

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

Method	Description
Inheritance	Method of allowing class to inherit attributes and methods of another class.
Polymorphism	Method that performs different things according to the object's class.
Operator Overloading	Allowing the same operator to be used for multiple operations.

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? Database is an organized collection of structured data. Advantages: can make data valuable resources, can efficiently and quickly manage lots of data, can be used to dynamically analyze data.

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

Data type	Definition
int	Used to store numbers
varchar	Used to store strings
date	Used to store dates

3. In what situations would SQLite be a better choice than MySQL? When using smaller datasets

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 pure OOP but Javascript is not.

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? Python has limitations like speed and memory management.

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? ORM maps data within the structure of objects in the programming code and manages the data using classes and structures within the code. Advantages: Speeds up development for teams and decreases cost.

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 well, I think. The thing I did well is that I got the functionalities to work. I think I would improve it by adding a better user interface to it like creating more alerts when the user deletes or updates a recipe.

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 build an app for allowing users to create, search, update and delete recipes within a database. I used classes and its functions to do so, allowing reusability of attributes efficiently.

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 got the code to work and do what the instructions wanted.
   b. What's something you're proud of? I was efficient and focused on the coding parts so I could stay motivated.
   c. What was the most challenging aspect of this Achievement? Technical issues like setting up the database and Python so my code would run properly.
   d. Did this Achievement meet your expectations? Did it give you the confidence to start working with your new Python skills? Yes, to both.
   e. What's something you want to keep in mind to help you do your best in Achievement 2? Don't get demotivated by technical issues and code requirement ambiguity, because this will prepare me for a real job in software engineering.

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? Yes, it was not just successful, but efficient. I would do this differently by going even faster and scheduling each subtask on paper to get more focused.
- 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 was most proud of not just getting the work done, but also efficiently. I will repeat this by using the same work scheduling strategy as I think

Django will have many subtle technical challenges that will take up more time and I am prepared for this.

- 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? I think the most difficult things were technical issues like setting up the database and Python so my code would properly run. Thanks to google I found help and was able to resolve them quickly. This prepared me for the next achievement as many of these instances will happen there.s

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? Vanilla: Advantages: Better for small scale projects. Disdvantages: less flexible, scalable and secure. Django: Advantages: More flexible, scalable and secure. Disdvantages: Not as viable for small scale projects.

2. In your own words, what is the most significant advantage of Model View Template (MVT) architecture over Model View Controller (MVC) architecture? In mvt, the template is responsible for rendering the View while in mvc the View handles the rendering.

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? How to build scalable, flexible large scale APIs.
   - What do you want to get out of this Achievement? To build a backend for a web app in django.
   - Where or what do you see yourself working on after you complete this Achievement? On a highly scalable REST API with Django.

Exercise 2.2: Django Project Set Up

Learning Goals

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

Reflection Questions

1. Suppose you're in an interview. The interviewer gives you their company's website as an example, asking you to convert the website and its different parts into Django terms. How would you proceed? For this question, you can think about your dream company and look at their website for reference.
(*Hint: In the Exercise, you saw the example of the CareerFoundry website in the Project and Apps section.*)

One example is where I have to take user inputs. Instead of using a console I would use a form and post inputs to my API. Examples of doing this including signing up for the app and logging in, creating and deleting recipes, deleting account, etc.

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

Run migrations (make changes to database schema of application.) Run the server (deploy application to localhost and give weblink to access application from browser.)

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

I would use it to allow a user to sign up, log in, create data structures (e.g. recipes), search, update, delete them and eve log out or delete the account.

Exercise 2.3: Django Models

- 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 web apps access and manage data through models.Each model maps to a single database table. They define the structure of the data stored in the database. It is scalable, versatile and easy to work with.

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.

They indicate how and app is supposed to behave. They also help in creating documentation of a project. For example in the bookstore app you can test the length of a book title.

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.

A view is the logic that Django runs when a user accesses a URL. For example in our app when we specify the url endpoint /admin/, this specifies the corresponding view as admin.

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?

Class based views. Because they are easy to reuse and extend, and this is because with classes you can reuse not just functions inside the classes but also data structures and they let you reuse views.

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

A template is a text file. It can generate any text-based format. It contains variables which get replaced with values when the template is evaluated, and tags, which control the logic of the template.

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.

Static files are not dynamically generated by the Django server and don't change while the web application is running. They are provided by the developer during project creation. They are loaded as {% load static %}.

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	Displays data as a list. Allows you to display a list of items with item text and, optionally, an icon to identify the type of item.
DetailView	Displays data details. Used when you want to present detail of a single model instance.

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.

It is going great. I have managed to successfully set up the precursor for building an API in Django. My only real struggles have been technical issues like running commands properly and syntax errors. I need more practice with building REST API endpoints and data modeling.

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.

It defends the app from risk of unauthorized access and sensitive data. Example: online bank account. Lots of sensitive data like social security number, bank id, etc that strangers can access and steal money from.

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

Create view, Create the template, Specify the URL mapping, Register the URL to the project.

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()	Used to verify a set of credentials. It takes credentials as keyword arguments, username and password for the default case, checks them against each authentiction backend, and returns a User object if the credentials are valid for a backend.

redirect()	It lets you store redirects in a database and handles the redirecting for you.
include()	It chops off whatever part of the URL matched up to that point and sends the remaining string to the included URLconf for further processing.

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.

Let's say Facebook. I see that data collected includes name, password, location, age, etc. They can use location to determine recommending events, profile pages and friend requests. They can use age to recommend movies, songs, etc.

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

Iteration, slicing, pickling, len(), repr(), list(), bool().

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.

Queryset: advantages: safe to run in asynchronous code, and do not have separate asynchronous versions. Disadvantages: if you refer to some table alias of the queryset in the raw SQL, then it is possible that Django might change that alias.
DataFrame: advantages: easy to use, efficient data structure, handles missing values. Disadvantages: memory intensive, compromised performance.

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.