Python for Web Developers: Learning Journal

Pre-work

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 a friend who went to school for coding initially, and he tried to teach me the basics of Python in Visual Studio Code over online calls. I also had a digital Python textbook from another friend who also went to school for coding, so I tried to use that material to learn as well. I did not get to far with this initial attempt at learning, but it did set me up for pursuing coding in the future, and my prior exposure to the syntax and structure of Python aided in my learning of JavaScript in the Full-Stack Immersion course as I noticed the similarities between the two languages.

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

   I am already pretty familiar with working with strings, variables, conditional statements, and some functions. It has been a couple years since I was last learning any Python material, so I am certainly rusty now.

   What I would like to learn is how to use it for building apps, frontend and back, similarly to how I have learned to use other languages and frameworks to build apps.

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 anticipate I will have different challenges caused by errors on my part in following instructions from the Exercise readings and tasks. I do my best to follow the instructions to the letter, but there can always be something I overlook or misunderstand. In any case, I know I can always look to online forums like Stack Overflow, ask ChatGPT, contact my mentor, or reach out to the Slack community whenever I am stuck with a project.

Achievement 1: Introduction to Python

Exercise 1.1 Getting Started with Python

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

   Frontend development focuses on the client-side of an application, which consists of the design, layout, and routing, whereas backend development pertains to maintaining the functionality of URL endpoints, authentication/authorization, accessing data from external sources, and storing data in databases.

   As a backend developer, I would be hired on to implement authentication/authorization logic to endpoints that require it using different models, testing the functionality of each endpoint using a program like Postman, maintaining an external database for storing application or user data, maintaining a hosting service for the application, or monitoring application performance through a service like Atatus.

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?**

   On the surface, JavaScript and Python have some similarities, particularly when it comes to their syntax and code structure. Compared to JavaScript though, Python is a much more user-friendly with its simplistic syntax, making the code easy to read and understand. It also has a simple package management system that comes built-in, making it easy to access a wide array of functions and operations without having to install them ourselves. Like JavaScript, Python also 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?**

I want to learn more about how Python can be used to create apps and a backend, similar to my past projects. I want to get a better understanding of how this language works similarly to others I have learned while being a simpler syntax. I am not sure what I see myself working on after I complete this achievement, but I know I want to get better at documentation and code structure.

Exercise 1.2: Data Types in Python

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's shell is more user-friendly than the regular Python shell because will display certain things in different fonts and colors, making them stand out more where it counts. It allows for testing pieces of code quickly as each command is executed right after typing and pressing Enter. It also automatically indents when writing nested statements, similarly to regular coding.

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?**
Integer (int)	All whole numbers, positive and negative	Scalar
Float	All decimal numbers, positive and negative	Scalar
String (str)	Array of characters, alphanumeric and symbol	Scalar
Boolean (bool)	Conditional values, True or False	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.**

   They are both considered arrays that can store any data type, but lists are mutable and can be changed. Tuples cannot.

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.**

   Vocabulary

   Each word would be their own dictionary in the following structure:

   vocab_word = {

   "name": <"name">,

   "category": <"noun, verb, etc.">,

   "definition": <"definition">

   }

   Each category could also be selected from a tuple containing the different options, since there should not be a need to modify this data.

   Language

   Each coding language would be a list containing an array of their respective vocabulary word dictionaries. This way, the words can be accessed quickly and easily modified if necessary.

   language1 = [

   vocab_word1,

   vocab_word2,

   ...,

   ]

   language2 = [...]