# Python Learning Journal

**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 web development is what the user sees and interacts with. Backend is where operations that occur for handling requests, interacting with databases and files the server. If I was hired as a backend programmer, I would anticipate working on CRUD operations, working with databases, security concerns, and working with servers.

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?**

   JavaScript and Python are both scripting languages that use dynamic typing. JavaScript is not optimized for backend development, however Python's straightforward syntax and high readability and range of libraries and frameworks may make more sense in this situation.

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 basic Python syntax.
   • In this achievement I want to be able to setup an environment to get started using Python.
   • After this achievement I want to continue to build my knowledge and skillset to become marketable as a backend developer.

**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 is more user-friendly. It utilizes different colors making reading the code easier. The lines are numbered. There is syntax highlighting and indenting is done automatically.

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.**

| <u>Data type</u> | <u>Definition</u> | <u>Scalar or Non-Scalar?</u> |
|---|---|---|
| String str | immutable, can be composed of alphanumeric characters and symbols contained in single or double quotes | Non-Scalar |
| Numeric int | integer (whole number) negative or positive (to infinity) | Scalar |
| Boolean | True or False statement | Scalar |
| Sequence list | mutable, ordered sequence of data using square brackets and commas | 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.**

While both are a sequence of data, a tuple is immutable while a list is mutable. Data in a tuple is write-protected. A list can be more easily manipulated. In the code, a tuple is written using parenthesis and commas while a list utilizes brackets and commas.

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.**

For each input they would be stored as strings. Having words, definitions, and category may get confusing if they were just stored as a list. If you used a dictionary you can use key-value pairs to distinguish between categories. Also

with a dictionary you can always extract the keys or values and place those into a list if needed in the future.

**Exercise 1.3: Functions and Other Operations in Python**

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:**

   ```python
   destination = input("Where would you like to travel to? ")
   if destination == 'Hawaii':
       print("Enjoy your stay in Hawaii!")
   elif destination == 'Alaska':
       print("Enjoy your stay in Alaska!")
   elif destination == 'New York':
       print("Enjoy your stay in New York!")
   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 operations are used on conditional statements and will return True or False. The and operator will return true if both statements are true and false if either statement is false. The or operator will return true is either statement is true and false only if both statements are false. The not operator will return the opposite of the statement (if statement is true it will return false but a false statement will return true).

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

   Functions are blocks of code run only when they are called. They are useful for manipulating data or code. They cut down on repetitive code and makes it easier to read. They are many builtin functions in Python or you can create your own to carry out a specified task.

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 am learning the language by completing exercises and looking up Python documentation. My mentor has given me resources to explore online as well. I am getting comfortable with using the Python Shell and executing a script in the

terminal. After meeting with my mentor for the first time, I am gaining industry insight on what I need to become marketable as a backend developer. This will entail learning another coding language to expand my knowledge base.

**Exercise 1.4: File Handling in Python**

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

   When you use variables to assign and keep track of values that data doesn't exist when the script stops running. All the data is lost and cannot be used later. Creating local files allows you to store data permanently.

2. **In this Exercise you learned about the pickling process with the <span style="color:red">pickle.dump()</span> method. What are pickles? In which situations would you choose to use pickles and why?**

   Pickles convert data into a byte stream. This binary file can store complex information. Pickles need to be used for anything that is not basic data, which can be stored as a simple text file.

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?**

   The os module using the os.getcwd() command to find out what file directory you're currently in. The os.chdir('<path to desired folder>' ) command is used to navigate to another 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?**

   By using a try-except block. The try block is where you place the code that the error may occur. If there is no error than the rest of the code is executed as normal. If there is an error then the except block handles the error. After which, the code continues to run until the end.

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 really enjoying this achievement. I had to take a week off and that is not ideal in learning any new skill but fortunately I was able to jump back in. I find the

hardest part for me is starting writing the code for a task. Once I figure out where to start I can logically think my way through it. I hope to get over this hurdle of "I don't know where to start" by just consistently doing the assignments and becoming more familiar with Python.

**Exercise 1.5: Object-Oriented Programming in Python**

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

   OOP is when you bundle data and methods into individual objects. The benefit of OOP is that it keeps code simple and non-repetitive. You can pass information to the "template" and it will give you the functionality of that template for use with your particular object.

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

   Everything is an object in Python. Each object has a corresponding class. A class is a template that provides an internal structure. One class can have many objects. An example is a class, shoe. You can then create an individual shoe object, like a sneaker. You could describe color, function, and other attributes for that individual shoe. Another shoe object could be a dress shoe with its individual characteristics that are setup in the shoe 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 | This is when you pass the properties and methods of a class to an instance of the class for use. This is done when declaring your class you pass the inherited class into the parentheses at the beginning. Then it is available to the subclass you just created for use without having to redefine those attributes. |
| Polymorphism | This is where a data attribute or method has the same name across different classes or data types but performs different operations. This can be built in or custom. |
| Operator Overloading | This is when you use operators on a custom class. The method for the operator use has to be a function defined in the class. An example is __add__() would need to be a defined function in your class and then the + operator can be used to call this function. |

**Exercise 1.6: Connecting to Databases in Python**

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

   Databases are a way to store data and keep it organized in collections. The advantages are to keep the data in a standardized format that can be accessed by other applications than Python. They can also be password protected. Databases create an easy way to read, store and modify data.

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

   | Data type | Definition |
   |---|---|
   | String | This is a text value or string of characters |
   | Numeric | Integer or decimal number |
   | Date and Time | Can be date or time or both together |

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

   Very simple databases or to use a test database would make SQLite a good choice. It has a simple setup with no installation required.

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 seems to have a more straightforward and simple syntax and code block structure. Python seems to have a little more structure which I like. Javascript can be used for frontend and backend. While Python is used on the backend. It is

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 is used mainly on the server-side. It has a lower execution speed and higher memory consumption. The simplicity of Python can be a disadvantage.

**Exercise 1.7: Finalizing Your Python Program**

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

   An ORM converts the contents and structure of the database into classes and objects that you can interact with directly. It saves time not having to worry about SQL syntax.

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 think I got comfortable using conditionals in relation to using the data. If I was to start over I would get fancier with the look of the user interface. I usually find that to be an after thought.

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 used Python to create a recipe app. In this process I was able to touch on different aspects of Python. I used a variety of data types and implemented operators and functions for the application. I was able to use Object-Oriented Programming, databases, and Object-Relational Mapping. This has expanded my knowledge and skillset in regards to using Python.

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. I found python to be an easy language to start using right away. So the simplicity helped to feel comfortable faster.
   b. What's something you're proud of?
      i. I am proud of the fact that I have had several personal challenges arise and I am was able to get back on track.
   c. What was the most challenging aspect of this Achievement?
      i. The search_ingredients function was challenging with all the steps involved. The other portion that took more consideration was search_recipe function. Having to manipulate the data from the database and then present it in a useful format.
   d. Did this Achievement meet your expectations? Did it give you the confidence to start working with your new Python skills?
      i. I thought it might be more in depth. I like all the topics covered.
   e. What's something you want to keep in mind to help you do your best in Achievement 2?
      i. I think when learning anything new you need to keep revisiting the material. It is easy to forget the basics or skills that you did previously. So I think reviewing what has already been presented is helpful.

**Exercise 2.1: Getting Started with Django**

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 javascript would be useful for simple applications. This would be good for webpages that do not use a database or the need for users to login. The disadvantage is in the future if you the need for these things happens it is harder

to scale the project for growing and changing needs. It truly depends on the goal and intention of the application.

Using Django is good for large-scale systems. It is useful in applications that have a backend and frontend. It can handle a database easily. The advantages of Django are rapid implementation and the ability to handle scaling of a project in the future, i.e. the need to add new hardware or components. It is also built with security in mind with its design. The disadvantages would that you have to stick with the Django way and you don't have control over certain aspects of the finer details of the application.

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

   I think the rapid setup is the most significant advantage. It takes away the need to write a lot of code initially. I like that it that it uses the don't repeat yourself principles.

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.**

   1) I want to learn how to get started using Django and the implementation in a project.
   2) In this achievement I want to get more comfortable with Python and know when and where using it would be applicable.
   3) After this achievement I want to create applications using Python to further my knowledge and show employers my skillset.

**Exercise 2.2: Django Project Set Up**

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**

   The website itself would be a project. Each part that has a separate functionality would be an app. A blog I frequent has a login, daily looks, and features. All of these would be considered apps.

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

To deploy an application you want to run migrations, which sets up and saves changes the database tables. This is done via terminal: % python manage.py migrate

Then you are going to run the server. In terminal: % python manage.py runserver
It will be running on port 8000. You will go to your browser and enter '**http://127.0.0.1:8000/**'. You should see a rocket with the words "The install worked successfully! Congratulations" if it was successful. You will control C in terminal to quit.

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

From the Django documentation, the admin interface reads metadata from the models to provide an interface where you can manage content on the site. You can create actions, functions, that can be called on a list of objects on change list page. You can customize the admin interface as well. It also has an admin documentation generator.
I would register my models so I can manage the data. I would create users for the project.

## Exercise 2.3: Django Models

1. **Do some research on Django models. In your own words, write down how Django models work and what their benefits are**

Models are the source of information about the data. It has all the fields and behaviors that the data can have. You can access your data and manage it as well with a model. They really simplify how you manage your data.

2. **In your own words, explain why it is crucial to write test cases from the beginning of a project. You can take an example project to explain your answer.**

Writing tests in the beginning saves time and not only does is show errors it helps prevent problems from occurring during the development of your application. It creates reliable code from the start.

## Exercise 2.4: Django Views and Templates

1. **Do some research on Django views. In your own words, use an example to explain how Django views work.**

Views are Python function take http requests and returns http response. Let's consider a login button. The views.py will have the function that directs to a

specified file, like html. A template folder will contain the file that the function will direct you. The view is call by a URL.

2. **Imagine you're working on a Django web development project, and you anticipate that you'll have to reuse lots of code in various parts of the project. In this scenario, will you use Django function-based views or class-based views, and why?**

Class-based views would be a better choice. They are specifically for ease of reusability and eliminate the need to rewrite code. Function-based views are for a customized purpose that is needed.

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

Django's template language is fairly opinionated.

Syntax - four constructs: variables, tags, filters, and comments
1. variable - output value from the context using double brackets {{ }}
2. tags - provide arbitrary logic in rendering using {% %}
3. filter - transform values of variables and tag arguments {{ | }}
4. comments - noted by {# #}
Components
1. Engine is the instance of a Django template system
2. Template django.template.Template is a compiled template
3. Context holds metadata as well as context data
4. Loaders locate templated and return Template objects
5. Context processors are functions that receive http request and return dict of data to rendered context

**Exercise 2.5: Django MVT Revisited**

1. **In your own words, explain Django static files and how Django handles them**

Static files are not dynamically generated by the server. These files can be images, JavaScript or CSS. These files are managed with django.contrib.staticfiles in installed apps and that is defined with STATIC_URL in settings file. The files are saved in static folder. A tag used in the template to give path.

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 | This is a class that inherits from an existing view. It generates data as a list to be used as object_list or <model name>_list. You map the view via url path and it will look for the corresponding html file in templates folder. |
| DetailView | This package is also class based. It helps display data details. You can modify the data or add extra fields inside the class. The data is accessed in templates/detail.html |

3. **You're now more than halfway through Achievement 2! Take a moment to reflect on your learning in the course so far. How is it going? What's something you're proud of so far? Is there something you're struggling with? What do you need more practice with? You can use these notes to guide your next mentor call.**

   I am enjoying Python. The course itself is well laid out and gives good descriptions. I am able to find resolutions to problems quicker and easier as time goes on in my learning. I need more practice on memorizing all the different skills and information presented. I would like to go more in depth into Django.

## Exercise 2.6: User Authentication in Django

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.**

   Authentication allows the information displayed to be customized for the specific user. It also protects user specific information that you do not want know to other users of the site.

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

   You create a view in a views.py file. This is done by adding a function for login and authentication. Then you create a template. This is a html file that contains a check for error message, login form, security token, and submit button. We did not put the login within an app so you do not need to map on the app level. Next you register the url to the project in the urls.py under urlpatterns.

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 user's credentials against a database. It takes two arguments, username and password. |
| redirect() | Returns an HttpResponseRedirect to the url for the arguments passed. Its default returns a temporary redirect |
| include() | A function that takes a full Python import path to another URLconf module that should be "included" in this place. |

**Exercise 2.7: Data Analysis and Visualization in Django**

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.**

   The data can indicate what the user is interested in. Take a blog that is tailored to women who are interested in fashion, home decor, etc. You could track if they are saving things to a database under their user profile. You can track the links they are clicking through. It could help you create content that is more relevant to your audience. Thereby creating a more liked user experience in the future.

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

   -Iteration
   -Slicing
   -Pickling/Caching
   -repr()
   -len()
   -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.**

   DataFrame is better for data processing because it was created to provide fast and efficient data analysis especially for large data sets. It is similar to a spreadsheet, but are more powerful and easier to use.

**Exercise 2.8: Deploying a Django Project**

1. **Explain how you can use CSS and JavaScript in your Django web application.**

   You can do styling such as set background colors, images, spacing, font styles and more. You can also create transitions or animations. By incorporating widgets in fields you can modify attributes of your elements.

2. **In your own words, explain the steps you'd need to take to deploy your Django web application.**

   I used render.com. I tried multiple ways to get free account with Heroku but I kept getting a contact your school administration.
   I copied my repo and created a new repository on GitHub. Before I added anything on GitHub I updated the SECRET_KEY and settings.py with correct DEBUG info from **render.com**. I also changed the ALLOWED_HOSTS to specified content.
   I installed dj-database-url and psycopg2-binary via CL and updated DATABASES in settings to reflect postgres. I installed whitenoise and specified it in MIDDLEWARE in settings and updated the STATIC files ROOT and STORAGE.
   I installed gunicorn as well.
   I ran pip freeze > requirements.txt to create a file with dependencies. I added a script file build.sh at root which gives pip install and decencies. A render.yaml file was added to root as well. It indicates the databases and services, along with any environment variables that are needed.
   On **render.com** I created a new blueprint instance for web service and database. I updated my GitHub repo and allowed permission for **render.com** to access the repo. The site was deployed.
   I had to create a superuser  in the build since render didn't allow use of their shell with free accounts. I added recipes on /admin portion of the site.

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?
   I think understanding the basic functionality of the models, views and url implementation went well.

b. What's something you're proud of?

I am proud that I finally was able to get all the pictures working and uploading. I am proud that I deployed it in two ways and was able to figure out all the requirements for each.

c. What was the most challenging aspect of this Achievement?

CSS and styling can take a long time for me to implement. I try different ideas and implement them and then decide I do not like them. So the whole process takes an inordinate amount of time and then I still end up with something very simple.

The biggest challenge after I finished styling was trying to get media files to function and when that wasn't getting results I tried saving the images as static files and that presented major challenges as well.

d. Did this Achievement meet your expectations? Did it give you the confidence to start working with your new Django skills?

I feel that it did meet my expectations. I was able to figure out how to create an add recipe page and create a form that adds user input to the database. I was able to familiarize myself with Django in a general way. I had to spend a lot of time figuring out render.com since Heroku wasn't available at first. I then was able to setup Heroku and get my site deployed there as well. Along with configuring AWS S3 bucket for my application.