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?**

Before the course, I didn't know anything specific about programming. In the Intro and Immersion course, I learned a lot about HTML, CSS, JavaScript, Node.JS, React, Angular, React Native, and a lot of packages that I used within them.

Programming is challenging at times, but I like challenges. For example, I like to play games on the hardest difficulty possible and earn all achievements in them. I used this kind of approach, to go through the course this far, and I'm enjoying it.

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

I knew Python is a programming language, and that it is the best language to start to learn coding. Then, in the Immersion course, I learned, that Python is mostly used at the backend of an application because it needs to be transpiled for the browser to be able to read and utilize it.

I'd like to know how exactly Python works and what it is capable of.

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 the week that might be favorable to your facing challenges and growing. Plan for how to solve challenges that arise.**

The first challenge will be to get used to the new language we learn here, and how differently it will be used. My curiosity will possibly help me through this course, and the thought, that I would like to work as a developer.

At challenges in the tasks firstly I will check the documentation to find a solution to my problem. If it doesn't help, I will search for a similar problem someone had at Stack Overflow. If none of them works, I will ask my Mentor for some advice.

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 deals with the client side of an application. They work on how the application looks and feels to a user, how to send requests to the backend, and how to read the data returned from a request.

Backend development deals with the server side of an application. Their job is to elaborate on what data the application will use, how and where will the data be stored, and how will the client side communicate with the database.

I would be working on a REST API, which will communicate between the client side and the database, create models of how the data should be stored in the database (if the project works with a NoSQL database), validate requests, and let only valid data get stored, and last but not least, creating the logic, which allows only specific origins to communicate with the API.

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?**

It is one of the easiest languages to learn and understand, and much easier to debug or document it due to its readability. It uses simple built-in package management, which is useful in the way we don't need to look for third-party libraries in our project and the most common web operations are available in it as well. It has one of the largest communities, and it's the best language for fast prototyping.

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 do you see yourself working on after you complete this Achievement?**

   1: Learn how to use Python effectively.
   2: Progress through the Exercise
   3: Continue with Achievement 2.

Exercise 1.2: Data Types in Python

1. **Imagine you're having a conversation with a future colleague about whether to use 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 uses color schemas to distinguish line of codes, suggestions, etc.
    - It gives more readability to the shell
- iPython has an autocompletion tool
    - It writes all the possible functions usable with the sequence, considering its data type. It can be chosen by pressing 'Tab' navigating to the right function with the arrow keys, then pressing 'Enter'

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?
string	can store alphanumeric data with special characters surrounded by single or double quotes	Non-Scalar
boolean	can store either true or false, which can be used for example as statements to execute specific code lines	Scalar
integer	can store whole numbers in positive and negative directions as well	Scalar
list	Lists are able to store every kind of data type, even other lists, surrounded by square brackets '[ ]', and they are mutable.	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 response.**
   Tuples are ordered, immutable data storages surrounded by parentheses.
   Lists are ordered, mutable data storages surrounded by square brackets.

Tuples are processed and read faster, but they have the disadvantage of immutability. If we would want to modify its content, we have to add all the information again. They are most useful at working with a huge database.

Lists, on the other hand, have larger memory usage and are slower to read, but their content can be reassigned, modified, and new data can be added to it easily.

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.**

   Since flashcards don't usually need changes, tuples would be the best choice in my opinion. Due to its fast processing behavior, the application would run faster, and when a user would like to change something in a word's data, the frontend would fill out every input field, so the user can update data with minimum effort. When the development of the application continues, the small memory usage will come handy, for example, if a lot more languages are added to the application.

Exercise 1.3: Data Types 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 practive 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."**

Answer:

```
destinations = [ Berlin, Prague, Budapest ]
user_destination = input(“Write your wished travel destination: “)

if user_destination in destinations:
        print(“Enjoy your stay in “ + user_destination + “!”)
else
        print(“Oops, that destination is not currently available.”)
```

2. **Imagine you’re at a job interview for a Python developer role. The interviewer says “Explain logical operations in Python”. Draft how you would respond.**
   Answer: Logical operators are ‘and’, ‘or’, and ‘not’. They will return the statement ‘True’ or ‘False’.

   The ‘and’ operator will only return true if the statements before and after the operator are true. Otherwise, it will return false.

   The ‘or’ operator will return true, if either the former or latter statement is true. If none of them returns true, the operator returns false.

   The ‘not’ operator will return the opposite result, what it normally would be. For example, if the statement is ‘3 + 2 == 5’, normally it would return true. But if we write the ‘not’ operator before the statement, it will return false.

3. **What are functions in Python? When and why are they useful?**
   Answer: Functions can read and modify our data types. There are two function types: built-in and custom.

   Built-in functions are usable for everyone and almost everywhere in our code (it depends on the data types we are using the functions on). We can do every kind of thing with them. For example, converting data to another data type, adding extra content to a variable, modifying, or removing our data.

   Custom functions on the other hand are made manually by the developer, and it will execute the code block that is related to that function after that has been called.

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.**

Answer: I fell behind schedule a bit, but I'm on the track to complete the goals I was set. They were:

1: Learn how to use Python effectively.
2: Progress through the Exercise
3: Continue with Achievement 2.

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?**
   Answer: It is important to store data in files, because otherwise when the script you are running is finished, all data the script created and worked with would be terminated. But if you store the created data in local files, you can access them later.

2. **In this Exercise you learned about the pickling process with the `pickle.dump()` method. What are pickles? In which situation would you choose to use pickles and why?**
   Answer: Pickles are data, that have been disassembled into bytes and will be stored in a file with the '.bin' extension. It is very useful if you would like to store more complex types of data (for example, dictionaries), otherwise, the more complex data should be converted to strings and stored as texts, or it would not be possible to store them.

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?**
   Answer: You have to import 'os' module, then you can use os.getcwd() to see in which directory you're currently working. If you would like to change the directory, you have to use 'os.chdir("<path to desired folder>").

4. **Imagine you're working on 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?**
   Answer: I would use 'try-expect-else-finally' statements. The questionable block of code would go to the 'try' block, I would print a message at the 'except' block, to let me know if there is an error, and then to the 'else' block, I would add what the code should execute if the try block is successful. The 'finally' block can have code lines, what I want to execute anyway, and/or print a "Script end." text.

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 mentor call.
Answer: I feel proud, that I know more and more about Python after every Exercise. Sometimes I struggle to find the error because the error message is often 'blended' with the data I wanted to print. I feel like I have to use more of the functions to really get to know them and use them effectively (possibly after the course as well).

Exercise 1.5: Object-Oriented Programming in Python

1. **In your words, what is object-oriented programming? What are the benefits of OOP?**
   Answer: In object-oriented programming, we strive to create custom classes, and sort every data as instances into one of these classes. The goal is to reduce code repetition and make the data creation more manageable. The benefits are reusable functions and initiations, and the inheritance possibility through classes. Inheritance is to pass defined functions from the parent class to the subclass, so we don't have to repeat our functions.

2. **What are objects and classes in Python? Come up with a real-world example to illustrate how objects and classes work.**
   Answer: Objects are the data, that we are adding as instances to a class. Classes are like a container of objects, where we can define custom functions of what we can do with the data.

   If we didn't use classes, we had to use dictionaries, tuples, or lists to add key-value pairs to every data we wanted to store. With a great amount of data, it would be really hard and time-consuming to handle. Using classes, we can pass all data as attributes when we are creating a new instance of a class, and it will automatically add the attributes as values to the corresponding keys.

3. **In your own words, write brief explanations of the following OOP concepts; 100 to 200 words per method is fine.**
   Inheritance: It is a way to pass functions and initiation from the parent class to the subclass. For example, if we have a parent class called 'Animals' with the function 'get_age()', to get the age of a specific object the Animals class has. If we create a subclass of the 'Animals' class called 'Cat', we can use the 'get_age()' function on every instance the 'Cat' class has. But, if we define a function in the 'Cat' class, we won't be able to use it on objects in the 'Animals' class. The inheritance works only one way. Creating subclasses

under the ‘Cat’ class will inherit the defined methods both from the ‘Animals’ and ‘Cat’ classes.

Polymorphism: Polymorphism means we can use the name of predefined or already defined functions in a class, and it won’t have the same result unless we don’t define it ourselves. For example, we can define the ‘sort()’ function, but it will add predefined integers together if we want it to (but it would be a bad practice to do so). Or we can have two classes, that use the same function name, but the end result is different. For example, the ‘Cat’ and ‘Dog’ classes can have the same ‘speak()’ function, the ‘Cat’ class will return ‘Meow’, but the ‘Dog’ class will return ‘Bark’.

Operator Overloading: Normally, we can’t use any operation on a class, so we have to define them ourselves, using the corresponding name of the operation (for example ‘add’ for ‘+’), surrounding with two underlines at both sides (‘def __add__’), and we have to write all the functionalities we want it to do. After defining it, we can add two objects of this class together (for example: ‘class_instance_1 + class_instance_2). We have to use the same method to compare objects within the class, we can then modify them accordingly. For example, if we would like to sort the instances of a class, we have to check which instance is less than the other, and then sort them appropriately.

Exercise 1.6: Connecting to Databases in Python

1. **What are databases and what are the advantages of using them?**
   Answer: Databases are where we can store our website’s or program’s data. Databases can be used locally, and they can be hosted, for example on the Cloud. The advantage of using them is that we can have an organized data storage where we can create new data, search for them, and update or delete them.

   The advantage of a local database is that we don’t need any external storage to store our data, the user’s device will be used to solve this problem. The disadvantage of this approach is, that if something is happening to the device or to the database on the device, all data will be lost.

   The advantage of a hosted database is that our data will be accessible to any device we use, it can be account-bound or globally accessible. The disadvantage is that the data will be exposed to hackers with a much higher probability (It can happen with the locally stored data as well, sadly, I don’t have the right knowledge to say exactly, which one occurs more often).

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

Data type	Definition
VARCHAR	It can store alphanumeric data. (e.g. “asd02.3”)
INT	It can store only integers. (e.g. 52)
FLOAT	It can store whole numbers with a decimal point. (e.g. 35.23)

3. **In what situations would SQLite be a better choice than MySQL?**
   Answer: If we would like to test something, or we would like to use a very simple database to store our data.

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?**
   Answer: So far, I think JavaScript has a better use on the client side since the browser can read it right away, and we don’t need to transpile our code. On the other hand, if we’re looking at the server side of programming, I think Python is easier and more understandable to create a connection with the database. With the use of IPython, we can easily test smaller parts of our code with immediate response. The package manager of the Python side (pip) seems to be much faster and more reliable than the Node.js side (npm). I don’t have experience yet with APIs created with Python. If I remember it right, here is where Node.js is stronger than Python because of its async behavior.

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?**
   Answer: For web development, we have to transpile our code, so that the browser can understand it. Python doesn’t have async behavior, that’s why it is a disadvantage if we would like to build an API with it. We have to remember which working environment we used to create a certain project, and which package versions we used in this environment, in case we lose all the data from our device (we can export to a text file, so, in this case, we have all the information on GitHub). In my opinion, JavaScript is stronger with its “package.json” file and the globally installable packages.

Exercise 1.7: Finalizing Your Python Program

1. **What is an Object Relational Mapper and what are the advantages of using one?**
   Answer: It is a package, that helps developers to add entries in their database using classes and functions to add, read, edit, and delete data inside their databases without directly using the language of the used database.

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?**
   Answer: It was a long journey to build this app with various approaches, with and without databases as well. In my opinion, I did a great job of finding and preventing the possibility of breaking my application. Things that I would do otherwise are:
   - If the user makes a mistake with the input, the program should let them try it again immediately, instead of throwing them back to the main menu after every mistake and making them do things from the beginning.
   - Checking if a recipe name is already taken, and ask the user to give another name or do they want to make another recipe with the same name.

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 this question.**
   Answer: I used IPython to test smaller parts of the code, to see, how would they respond to a specific kind of argument or function. As Python is an object-oriented language, I built a recipe application using classes and subclasses. Then, I used MySQL as a database to save what I created inside the application. Last, but not least, I used the SQL Alchemy package to be able to use my database through the Python way (so using object-oriented programming).

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?**
      Answer: I had to get used to the language after JavaScript, but after getting confident using it, I could build the apps pretty fast in my opinion.
   b. **What's something you're proud of?**
      Answer: I am proud to have learned two more languages, what is Python and MySQL during this achievement.
   c. **What was the most challenging aspect of this Achievement?**

Answer: To get used to the language after using JavaScript, and use object-oriented programming.

d. **Did this Achievement meet your expectations? Did it give you the confidence to start working with your new Python skills?**
   Answer: Yes, I am grateful that I had the opinion to get to know Python, and be able to create a program with it. After this Achievement, I will be able to work with Python with a lot more confidence.
e. **What's something you want to keep in mind to help you do your best in Achievement 2?**
   Answer: To learn consistently next to work as well, since in the end, I would like to work in the development/IT industry. For that, I have to reach the end of this course and begin to work in the field.

Pre-Work: Before You Start Achievement 2

1. **Reflect on these questions:**
   a. **Was your study routine effective during Achievement 1? If not, what will you do differently during Achievement 2?**
      Yes, it was. The hardest factor was at my study routine, that I began to work 40 hours a week. It is really hard to manage my private life and keep up with the timeline.

   b. **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 proud that with less time invested in the course, I was able to finish the tasks I aimed for most of the time. I am proud that I was able to quickly adjust to Python from JavaScript, and that's why at most of the tasks I didn't have much problem completing the requirements.

   c. **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?**
      Mostly, I had a problem at first switching from JavaScript, which I used for seven to eight months before I began to learn Python. Now, I feel a lot more confident using the new language, and I feel like this will help me a lot in Achievement 2.

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?**

Advantages:

- Security
    - Secure-by-design implementation
    - Automated encryption
- Fast prototyping and development
- Fast processing
- Easy communication with the database
- Don’t Repeat Yourself method
    - Helps to keep the code clean and better readable
- Scalability
    - Works well with high bandwidth, so the project can handle a lot of users at once
- Large, supportive community
    - It comes handy if the developer stuck in a project and needs help to proceed.

Drawbacks:

- A project can only be built with the “Django Way”
    - The developer has to give up control over some part of the code
- It works poorly if no database will be used, or it will be used with low bandwidth.

2. **In your words, what is the most significant advantage of Model View Template (MVT) architecture over Model View Controller (MVC) architecture?**
   The low-level operations will be handled by the framework instead of the developer itself. The views and models are loosely coupled, and the operation flow seems to process faster than with the MVC method. Last, but not least, MVT is suitable for small projects as well, MVC is too complicated to build at small applications.

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:**
    a. **What do you want to learn about Django?**
       I would like to learn how to work with Django to get a web application, and I would like to see the performance difference between my projects built with JavaScript, React, or Angular.
    b. **What do you want to get out of this Achievement?**
       I would like to confidently use Python and Django at the end of the Achievement so that I can use them in a professional environment, should it be a workplace or a private project.

c. **Where or what do you see yourself working on after you complete this Achievement?**
First, I would like to refine my resume and portfolio, so that I can look for a developer job. Next, I will think about a project, which I can showcase in my portfolio, or I can use in my private life.

Exercise 2.1: Getting Started with Django

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 to look at their website for reference.**
Answer: First, the project is the whole website with all the web pages. The project consists of applications. Applications are a smaller, specified part of a project. Applications are at this website the main page, 'About Me', 'My Works' and 'Contact' pages. All the applications are highlighted in the 'urls.py' file, where they are bound to a specific URL. For example, after clicking on the 'Contact' link, it will open the "website.com/contact" URL, which will tell Django to generate the view of the 'Contact' app.

2. **In your own words, describe the steps you would take to deploy a basic Django application locally on your system.**
Answer: I would begin with the creation of a new virtual environment, using 'mkvirtualenv <environment name>. After that, I would install django using 'pip'. When it finished installing, in the terminal I would enter 'django-admin –version' to make sure it has been successfully installed. I would move to the folder where I want to create the new application and enter 'django-admin.exe startproject <project name>' to create the new project folder with the basic files and sub-folders. Then, I would rename the first generated folder, to avoid the misunderstanding of the parent and the child folder, because at default they have both the name of the project. After moving to the renamed folder in the terminal, I would enter "py manage.py migrate", to create a database (it can be specified, which database type we would like to create, but for now, we can use the default SQLite). At last, I would enter "py manage.py runserver", which will return the URL of the test website. With this URL, I can open my website in the browser.

3. **Do some research about the Django admin site and write down how you'd use it during your web application development.**
Answer: I would use it to grant some users more authority (mostly they would be my alt accounts to test the functionality of the website as all account ranks possible), creating, updating, and sometimes deleting a few accounts, what I

don’t need anymore. I would create new User models, where I can set what they can access on my website. Then changing the display of the Module for a more user-friendly approach. After that, I can modify, add, and delete users in the model.

Exercise 2.3: Django Models

1. **Do some research on Django models. In your words, write down how Django models work and what their benefits are.**
   Answer: Django models determine what credentials the data has to fulfill to be uploaded to the database, and the model forms a unified database. It means, every data will have the same keys, and the values have to satisfy the rules the developer set in the key of the model.

   Django came up with the solution to design our models as a Class, and the Objects of the Class will be our data. After defining our data, Django will transpile the code to a language the database will understand (depending on which database we use in our project.) The benefit of this is, that we don’t have to learn different database languages, it is sufficient to know how to create a database with Django.

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.**
   Answer: It is essential to write tests from the beginning of the application because that way we can immediately see if something is wrong with our project. There are more type of testing that we can use:

   Unit testing: It is used to test the core part of every chunk of code. If there will go something wrong, it has a high risk of breaking our whole application. With unit testing, we can prevent it.

   Integration testing: Here, we are testing the units, when they are used together. For example, if the first unit shows some information, and the second unit is a button with “See more” text. If we press the button, more information should appear on the screen.

   There are more testing types, but these two are the most core part of testing. Without testing, we have to test our application every time we change our code. Tests make our code higher quality and save us a lot of time.

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.**
   Answer: Basically, Django views are like an API part of the server side of the application. It communicates for example between the client side of the application and the database. If a specific URL request is made on the frontend, Django will check, which view is associated with that URL, and execute the code inside this file. The view will then accordingly return data to the client side. It can be a static or dynamic HTML file, an image, a response from the database, etc.

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?**
   Answer: I would like to use class-based views, exactly because it has the advantage to reuse specific code lines easily. It has the disadvantage of worse read- and writeability, but in the long run, the advantage will overcome the disadvantages of the class-based views.

3. **Read Django's documentation on the Django template language and make some notes on its basics.**
   Answer: Django template language (DTL) makes the web application available to use dynamical HTML files. This way, you can write static HTML files with dynamic parts, such as variables, tags, and filters. This way, you can connect your database with the frontend of your application.

Exercise 2.5: Django MVT Revisited

1. **In your own words, explain Django static files and how Django handles them.**
   Answer: Static files are files that don't change during the server's runtime. Such files are for example images, which are stored in the Django project in the media folder (or another folder name, it is customarily given by the developer, but the best practice is media). Other static files are CSS or static HTML files (HTML files not written with Django Template Language).

   Django handles them through the project's `settings.py` file. The developer has to define where the static files will be stored, and they need to be imported to the file we are using them (or just define its exact position within the project's folder.)

2. **Look up the following two Django packages on Django's official documentation and/or other trusted sources. Write a brief description of each.**
   ListView
   - This package gives us the possibility to use `object_list`, which contains all the instances of a Model. With the help of ListView, we can loop through `object_list`, and can create a dynamic HTML view.

   DetailView
   - This package will contain the instance from the Model the `views.py` is currently working with, and we can access it through Django Template Language. Like this, we can generate a dynamic HTML view, which shows the desired object's details.

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.**
   Answer: It is going well. As I'm beginning to understand the concept of Django, I'm able to solve more complex problems with it. The biggest struggle so far is to create a well-looking web application. Sadly, I need more time to check on the contents, what are better-looking UIs, and practice executing the acquired information into practice.

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.**
   Answer: There is a lot of advantage if we authenticate a user.
   - We can show them specific data, what is bound to their account, and only the person who logged in can see these data. (except for the admin, of course, unless the whole data is encrypted.)
   - We can add more access to specific accounts, so they will be authorized to see more data than the casual users.
   - We can protect our backend against hackers, especially if we are using more types of accounts on our website, for example:
     - casual user
     - admin
     - moderator
     - etc.

2. **In your own words, explain the steps you should take to create a login for your Django web application.**
   Answer:
   - Create a user/superuser, if none is created so far
   - Create a `views.py` in the project folder (src/<project name>) for login view, and create the business logic there
   - Create a template for login view.
     - First, create a template folder in the most outer folder of the project
     - Create an auth folder within
     - Create `login.html` template with the method `POST`
   - In `settings.py`, set the `TEMPLATES / DIRS` to `[ BASE_DIR / 'templates' ]`
   - In the `<project folder>/urls.py`declare the path of the login view.

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.**
   - **authenticate()**
     It is used to verify user credentials. This function takes the arguments `username` and `password`, searches for the user in the backend, and compares the backend value with the value passed to the `authenticate` function. If the credentials are valid, the function returns the `User` object, else returns `None`.

   - **redirect()**
     Returns an `HTTPResponseRedirect` based on the passed argument.

   - **include()**
     Takes a full Python import path to another URL config module, that should be included in the file we are using this function.

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.**
   Answer: In this reflection, I will consider talking about YouTube as a website. When a user is logged in, the data about what kind of videos the user clicks on or watches entirely is very powerful information, what uses the site to suggest similar videos to watch, and the website will have a better understanding, of what kind of advertisements the user will with high possibility like or even buy/use the advertised product.

On top of that, if the website has access to all the cookies from the browser, this effect takes place a lot more.

Another use of information is the age, nationality, country, and gender of all the users, so they can map what part of the users watch which kind of videos, and what demographic statistics the website has. This information is very useful for them, the third-party services, who are making the advertisements, and for the content creators as well, who will have a better understanding of which type of videos have more views, etc.

2. **Read the Django official documentation on QuerySet API. Note down the different ways in which you can evaluate QuerySet.**
   Answer:
   - Iteration
   - Slicing
   - Picking/Catching
   - `repr()`
   - `len()`
   - `list()`

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.**
   Answer: DataFrame uses in-memory data, and QuerySet makes a request through the network. That's why DataFrame is more performant when working with data, but first, we need that data to be in memory, that's why we make the database query with QuerySet first, then convert it into DataFrame.

   Advantages of QuerySet:
   - Abstraction: allows developers to work with data in a more Pythonic way, rather than writing raw SQL queries.

   - ORM Integration

   - Chaining: you can apply multiple filters, orderings, and other operations in a single line of code. This makes it easy to build complex queries in a readable and concise manner.

   - Lazy Evaluation: the database is not queried until the data is actually needed. This can improve performance by avoiding unnecessary database calls.

- Compatibility with various database backends supported by Django
- Caching: reduces the need to repeatedly query the database for the same data.

Disadvantages:

- Learning Curve: For developers new to Django or ORM concepts, there can be a learning curve in understanding how QuerySets work and how to use them effectively.

- Limited Control: there may be scenarios where you need more fine-grained control over the generated SQL queries. In this case, you have to use raw SQL queries.

- Potential performance overhead compared to writing optimized SQL queries directly.

- Complexity: As queries become more complex, chaining multiple filters and annotations can make the code harder to read and maintain.

- Debugging Challenges: When debugging, it might be less straightforward to inspect the actual QuerySets queries, especially in complex scenarios.

Advantages of DataFrames:

- Tabular Structure: DataFrames organizes data similar to a spreadsheet or a SQL table. This structure is intuitive and makes it easy to represent and analyze data.

- Easy to use: It has high-level abstractions for data manipulation, which makes it easy to perform operations like filtering, grouping, aggregating, and merging.

- Interoperability: DataFrames often support easy integration with other data analysis libraries and tools, allowing seamless transitions between different stages of the data analysis process.

- Labeling: the possibility of the labeling of both rows and columns, making it easy to reference and manipulate specific elements of the dataset.

- Handling missing or incomplete data, allowing users to fill, drop, or interpolate missing values.

- Statistical operations: simplify the calculation of common statistical metrics and measures.

- Wide range of formats: This makes it easy to import and export data from different sources.

- Visualization support via plotting libraries, enabling quick exploration and understanding of the data through charts and graphs.

Disadvantages:

- Memory consumption

- Performance overhead

- Limited to in-memory data: Handling extremely large datasets may require additional considerations or alternative approaches.

- Not Always Optimized for Specific Operations

- Dependencies: Potentially leading to compatibility issues.

- Overhead for small datasets: For relatively small datasets or simple data manipulations, using DataFrames may be too complex