Milestones: Use Python Basics for Market Analysis

How to use the milestone document

In this document, you'll find an example of milestones to follow for delivering your project. You'll see:

- Recommendations for accomplishing each milestone.
- Common issues and things to keep in mind.
- The estimated progress on the whole project.

These milestones are only suggestions: you are not required to complete them in this order.

Please note that the project progress for each milestone is a guideline but will vary depending on your pace.

General Recommendations

- While working, store your code in a GitHub repository and make frequent commits.
- Remember to commit a requirements.txt file, but don't store the virtual environment in the repository.
- Make sure that any output files (e.g., CSV or image files) are not committed to the repository either.

Milestone #1: *Set up your development environment*

10% progress once complete

Before working on this phase, I should have:

- Read through the entire project, including the requirements document.

By the end of the phase, I will have:

- Set up the IDE of my choice.
- Installed Python.
- Registered for a GitHub account.

Recommendations:

- Most of the time you spend developing software will be spent working in an IDE (integrated development environment), so it's important to be comfortable with the one you choose.
    - Spend some time exploring different IDEs. Study the resources below related to setting up an IDE and others for installing and trying out different IDEs to help you choose the one you would like to work with.
- You will also need to be familiar with the command line terminal for many software tasks, including running your Python script and managing your Git repository.

What to keep in mind:

- Python is a constantly evolving language and ecosystem.

Resources:

- To help you set up your IDE, the following chapters from the OpenClassrooms course Set Up a Python Environment will be helpful:
    - Choose the Right Editor for You contains details of some available IDEs.
    - From the chapter Download, Install, and Run PyCharm onwards, the course contains details on PyCharm.
- If you're interested in using Visual Studio Code (VS Code) as your IDE, follow the OpenClassrooms Course Set Up Your Front-End Development Environment, especially the first two chapters.
    - While this course is aimed at those setting up a front-end development environment, its content is also applicable to those wanting to use VS Code for Python.
    - If you choose this approach, you might want to explore VS Code extensions relevant to Python development.
- To help you set up your GitHub repository and use it to work on your project, follow the OpenClassrooms course Manage Your Code Project With Git and GitHub
- The OpenClassrooms course Learn the Command Line in Terminal can help you master the basic commands needed to manage files and folders on your computer.

Milestone #2: *Extract a single product's data*

25% progress once complete

Before working on this phase, I should have:

- An understanding of the HTML structure of the product page and the ability to identify each of the required fields in the page's HTML structure.

- An understanding of the CSV file structure:
    - How to define the headers.
    - The use of the comma as a field delimiter and the new line as a row delimiter.
    - Escaping the field text's delimiters, which is particularly important as some of the descriptions might contain commas and new lines.

By the end of the phase, I will have:

- Written a script that can successfully do the following:
    - Extract the data from **the selected page**.
    - Store these data in a local file in CSV format.

Recommendations:

- Start by picking any single product page (i.e., a single book) on Books to Scrape, and write a Python script that visits this page and extracts the information detailed in the requirements document.
- The data should be saved to a CSV file using the extracted fields as column headings.

What to keep in mind:

Make sure you have a good understanding of the following points:

- How to analyze an HTML file to uniquely identify how to extract each field from the page.
    - This will include an understanding of tags, ids, and class names, and selecting children/siblings in the HTML structure and content.
- What a CSV file structure is. Even though the output of this phase is only a two-line CSV file (a header row and a data row), a good understanding of the general CSV file structure will ensure a good foundation for the rest of this project.
- File naming: this is a good opportunity to explore the options of naming output files.
    - While you can choose any name for the file, it is important to use an extension that correctly identifies the file type (.csv). You should also select a name that describes the output (e.g., the item's name, the output type, timestamps).

Resources:

- This course provides some guidance on using the browser dev tools for analyzing the HTML content of a page: Optimize your website with DevTools, in particular the following chapters:
    - Set Up Developer Tools
    - Break Down the Tools View
    - Elements

Milestone #3: *Extract all the product data for a category*

50% progress once complete

Before working on this phase, I should have:

- Completed the script that extracts the data of a single product and saves it to a local CSV file.

By the end of the phase, I will have:

- Written a script that can successfully do the following:
  - Extract the data from **all of the books from the selected category**.
  - Store these data in a local file in CSV format.

Recommendations:

- You should pick any book category (listed in the left column of the home page) on Books to Scrape, and write a Python script that visits this category page and extracts the product page URL for each book in the category.
- Combine this script with the work you completed in the previous milestone to extract the product data for each book in the chosen category and write the data to a single CSV file.
  - Don't reinvent the wheel! Successful code doesn't need to start from scratch—most does not.

What to keep in mind:

- Some category pages have more than 20 books listed, spread across different pages.
  - You will need to understand pagination and how to handle it by writing code that detects if there is a "next" link on the page.
  - If there is a "next" link on the page, the script should visit that link and extract the information from that page.
- File naming (as with the previous milestone) is an opportunity to explore options for naming output files.

Resources:

- Read the following resources to learn more about working with CSV files, which you will use to store the data that your code extracts:
  - How-to Geek: What Is a CSV File, and How Do I Open It?
  - Wikipedia page on CSVs

Milestone #4: *Extract all the products from all the categories*

75% progress once complete

Before working on this phase, I should have:

- Completed the script that extracts the data related to all the products in a single category and saves them to a local CSV file.

By the end of the phase, I will have:

- Written a script that can successfully do the following:
    - Extract the data from **all of the books in all of the categories**.
    - Generate a CSV file for **each category**.

Recommendations:

- Write a script that visits the Books to Scrape home page and extracts the links to all the available book categories.
- Combine this script with the work you completed in the previous milestone. Your code should visit each category's page and extract the product information for all books in each category.
- Write the data for each book category to a separate CSV file.

What to keep in mind:

- You may need to review the file naming conventions for the script. Will the previously chosen file naming convention work in terms of content clarity of multiple files, or do you need to revise it?

Milestone #5: *Extract and save the image files*

90% progress once complete

Before working on this phase, I should have:

- Completed the script that extracts all categories and their books and saves the results in a separate CSV file for each.

By the end of the phase, I will have:

- Added the following functionalities to this script:
    - **Download the respective images** for all of the books in all categories.
    - **Save the image files** locally.

Recommendations:

- You should extend the part of the script that extracts the details of a specific book to download and save the image file for the book.
- You will need to choose a Python module that you can use to download the image files.
- Pay special attention here to relative paths and how you need to convert them to absolute paths so the script can download them.

What to keep in mind:

- Remember that you can transform the data you extract (in this case, the image's filename) before saving it.
- File naming image downloads: Revisit the importance of choosing a file naming convention. It should be easy to know which image belongs to which book (e.g., will they all be in one folder, or have a nested folder structure? Which attribute(s) will be used for the file name?).

Milestone #6: *Complete your written deliverables*

100% progress once complete

Before working on this phase, I should have:

- Written all the code.
- Completed the data ZIP file with cleaned and prepared data (extracted data and associated images).

By the end of the phase, I will have:

- Written the email to the team leader, Sam, describing how the program can establish an ETL pipeline.
- Written a README.md file and added it to the repository, giving instructions on how to run the code successfully and output some data.

Recommendations:

- Ensure that you have committed a **requirements.txt** file to the repository and not added the virtual environment.

What to keep in mind:

- You will need to know how to set up a .gitignore file for the repository.
- You might want to consider Markdown formatting for styling the content of the README, particularly the basic styles such as headings and code blocks.
- The draft email can be brief and to the point. Don't be afraid to keep things simple.

Resources:

- This basic Markdown styles guide will help you format your writing in the repository.
- To help you set up a .gitignore file for your repository, read the official documentation on gitignore

Project completed!