Budget Tracker Tool

Project #1

By Shehabeldin Abdelaal

Course: ENGM4620-Python for Engineers

The Budget Tracker Tool

The Budget Tracker Tool is user-friendly software that simplifies the process of tracking personal finances. At its core, the programme provides individuals with a disciplined strategy to tracking, categorising, and analysing their expenses over time. By offering a comprehensive overview of expenditure trends, the Budget Tracker Tool enables users to make more informed financial decisions, resulting in improved budget management and financial wellness.

Project Concept

The Budget Tracker Tool was designed to address a prevalent issue: many people find it difficult to track their expenses, which often leads to financial mistakes or mismanagement. This innovative tool makes it easier to record daily spending by allowing users to categorise them into predefined or custom categories, allowing for a more in-depth analysis of their financial habits over time. Its design prioritises user-friendliness, with simple prompts and inquiries to collect information. This ensures that the tool is accessible and simple to use for everyone, regardless of technical expertise, making financial management a realistic objective for a wide range of people.

Benefits and Uses

The Budget Tracker Tool greatly helps users by creating financial discipline and promoting frequent expense recording and analysis, which can lead to more thoughtful spending and possible savings development. By categorising expenses, users may easily detect spending trends and areas of excess spending, such as dining out, shopping, or entertainment, providing insights on how to change spending habits to better line with financial goals. Furthermore, the tool's budget evaluation feature is extremely useful for creating and sticking to budgets within specific timeframes, assisting users in saving for large purchases, reducing debt, or simply managing daily expenses more effectively, providing a clear and structured approach to achieving financial goals.

Applications

The Budget Tracker Tool has a wide range of applications. It can be used as an instructional tool to teach financial literacy to people of all ages, helping them understand the value of budgeting and financial planning. Furthermore, its simplicity and efficiency make it a helpful tool for financial advisors who work with clients to change their spending patterns.

Development Journey

The Budget Tracker Tool was developed through a series of experiments, tweaks, and enhancements, demonstrating a dynamic development process. Throughout the process, a variety of obstacles were faced that required innovative problem solving and iterative improvements to the tool's functionality and user experience.

Initial Challenges

One of the first challenges was developing a system capable of appropriately categorising expenses without requiring considerable user input. It was intended to reduce the work required to report a cost while still ensuring that the classification was logical and intuitive. This resulted in the introduction of a categorization tool that offers categories based on the expense description, with users able to confirm or alter these choices. Balancing automation with user control was critical to the tool's efficiency and flexibility.

User Interaction and Error Handling

Enhancing user interaction was another priority. The development team realised early on that user input could vary greatly, potentially causing system faults or misinterpretations. To overcome this, effective error-handling techniques were implemented. For example, when users enter dates or prices, the system now checks for formatting and validity, presenting them with clear instructions if any changes are required. This not only increased the tool's reliability but also made it easier to use.

Refinements

The Budget Tracker Tool was refined in large part after testing the program. After acting as a user, testing sessions found that some users will be unsure about how to begin using the tool or how to complete specific actions. In response, a more extensive guide and interactive prompts for the tool was created. These changes will dramatically improve the user experience by making the tool's features more accessible and clearer.

Continuous Improvement

The Budget Tracker Tool's development journey was one of constant improvement. Each difficulty presented a greater understanding of user needs and preferences, driving the goal of iteratively improving the tool's functionality and convenience of use. The development process, which included refining the user interface, enhancing error handling, and expanding the tool's features, demonstrated a dedication to create a valuable and user-centered financial management tool.

Decision Making Process

The decision-making process for building the Budget Tracker Tool was important to the project's success, driving the selection of features, design choices, and general direction. This phase was distinguished by strategic thought, experimentation, and an emphasis on user-centered design concepts.

Functionality Selection

One of the initial decisions made concerned the tool's main functionalities. It was planned to find a compromise between providing sophisticated budget tracking features and keeping things simple for the user. This resulted in the addition of functionality including spending categorization, budget review across certain time periods, and error handling for user inputs. Choosing which features to include entailed taking into account the hypothetical user feedback, assessing the complexity of execution, and weighing the possible influence on the user experience.

Designing for Usability

Usability was a top focus, affecting many decisions about the user interface and interaction design. Initially, a command-line interface was chosen, with a focus on functionality and development speed. Recognising the necessity for intuitive use, significant work was put into providing clear prompts, useful error messages, and simple instructions. This technique was adopted to minimise the entry barrier, allowing consumers without technical expertise to use the program efficiently.

Approach to Error Handling

Another area of concentration was error management, which involved determining the best way to deal with faulty or unexpected user inputs. Early versions of the program featured poor error checking, which could lead to misunderstanding or crashes. The choice to add comprehensive error handling—such as validating date formats and assuring numeric inputs for prices—was motivated by the need to make the tool more robust and user-friendly. This included creating bespoke validation routines and incorporating user feedback loops to help users fix their mistakes.

Functionalities

The Python-based Budget Tracker Tool allows users to easily track their costs and analyse their budgets over time. This tool is based on the Receipts class, which has all of the functionality required to add, categorise, and summarise expenses. Below is a discussion of its core capabilities and processes, with emphasis on clarity and simplicity of explanation.

Initializing the Tool

When you create an instance of the Receipts class, an empty list named costs is created. This list is intended to hold all expense entries, each of which is a dictionary providing information about the expense such as the date, description, price, and type.

Adding Expenses

The add_receipt method is a user-interactive process that requests information about an expense via input prompts. Users are requested to provide the date, description, and cost of the expense. The method handles errors to guarantee that the date is in the proper format (YYYY-MM-DD) and that the price is a valid number. If the input does not match the intended format or type, the user is notified with a pleasant error message, and the process terminates without adding the wrong data.

Categorizing Expenses

Once the categorize_expense method receives basic information about an expense, it determines its category using predefined keywords linked with each category. If the description corresponds to one of the keywords, the expense is automatically classified. If no matches are found, it is classified as 'random'. This way demonstrates the tool's capacity to automatically organise expenses, making it easier for users to manage their spending across multiple categories.

Evaluating Budget

Finally, the budget_evaluation method enables users to pick a timeframe and view their spending categorised and recorded for that period. Users enter a start and end date, and the method filters expenses that fall inside this timeframe, resulting in a more focused view of spending over specified periods. This functionality is useful for customers who want to monitor their spending habits over time and alter their budgets accordingly.

Test cases

The program was test will multiple scenarios to make sure that the functionality and concept is implemented as planned.

Adding a receipt:

```
[2]: # Example Usage and Test Scenario
receipts.add_receipt() # wrong format
receipts.add_receipt() # Dollarama snacks
receipts.add_receipt() # Soccer Match
receipts.add_receipt() # Parking fee
receipts.add_receipt() # wrong date
# Simulate other receipts as needed
```

```
Enter the date of the expense (YYYY-MM-DD):  Feb 20 year 2024
Please make sure this is the right date in the right format (YYYY-MM-DD). Try again.
Enter the date of the expense (YYYY-MM-DD):  2024-02-20
Describe the expense:  Bought some snacks from dollarama
Enter the price of the expense:  10
Receipt added successfully.
Enter the date of the expense (YYYY-MM-DD):  2024-02-24
Describe the expense:  Played some soccer with my friends in the BMO
Enter the price of the expense:  20
Receipt added successfully.
Enter the date of the expense (YYYY-MM-DD):  2024-03-04
Describe the expense:  paid a parking fee
Enter the price of the expense:  5
Expense 'paid a parking fee' does not match any category. Add as 'Random'? (y/n):  y
Receipt added successfully.
Enter the date of the expense (YYYY-MM-DD):  2024-30-09
Please make sure this is the right date in the right format (YYYY-MM-DD). Try again.
```

Evaluating the budget:

```
[5]: # Budget evaluation based on the start and end date
receipts.budget_evaluation()
```

```
Enter the start date of the budget period (YYYY-MM-DD):  2024-02-31
Enter the end date of the budget period (YYYY-MM-DD):  2024-03-19
Please make sure this is the right date in the right format (YYYY-MM-DD). Try again.
Enter the start date of the budget period (YYYY-MM-DD):  2024-02-20
Enter the end date of the budget period (YYYY-MM-DD):  2024-03-19
Expenses from 2024-02-20 to 2024-03-19:
Groceries: 10.0
Hangouts: 20.0
Random: 5.0
```

Conclusion

ChatGPT insights were especially useful during the creation of the Budget Estimator Tool, particularly in improving functionalities such as date handling with the **datetime** module, efficiently categorising expenses, and fixing issues in the README description. ChatGPT's advice was invaluable in overcoming initial challenges, particularly in guaranteeing correct date processing and developing a comprehensive system for expense categorization tailored to user inputs, which considerably improved the tool's functionality and user experience. Note that the contributions from ChatGPT were not directly copied but rather altered to meet the project's specific requirements, resulting in a custom solution tailored to improve user experience.

ChatGPT's support went beyond problem solving; it helped lay the groundwork for the tool's basic architecture and logic, particularly in areas that required thorough validation of user inputs and dynamic content management. This collaboration resulted in the development of a user-friendly interface as well as an effective classification mechanism, both of which were critical to the tools' success.

Furthermore, ChatGPT's markdown formatting advice helped improve the project's documentation, resulting in a more user-friendly and informative guide. It is critical to note that ChatGPT contributions were changed and updated to match the project's specific objectives, resulting in a tailored solution that improves user engagement.

Interestingly, the project did not require a **requirements.txt** file because it depended exclusively on Python's standard libraries, such as **datetime**, demonstrating the project's simplicity and ease of use. ChatGPT's role was essential, offering conceptual advice and a structural template that aided the development process, allowing for the smooth production of a functional tool without the need for additional libraries.