Personal Expense Tracker

The code provided is a Python-based **Personal Expense Tracker** program that allows users to manage their monthly budgets and expenses. It includes several functionalities, such as setting a budget, adding expenses, viewing expenses, tracking expenses for a particular month, and downloading the data in CSV format. The user interacts with the program through a menu-based system, and their choices dictate which actions the program performs.

Code Breakdown

1. display_menu() Function

This function is responsible for displaying the interactive menu to the user. It provides them with the following options:

- **1. Set Monthly Budget**: Allows the user to define a budget for a particular month.
- **2. Add Expense**: Enables the user to add a new expense (e.g., category, amount, date).
- **3. View Expenses for a Specific Month**: Lets the user view all recorded expenses for a specific month and year.
- **4. Track Expenses for a Specific Month**: Displays the total expenses for a specified month and checks it against the budget (if one is set).
- **5. Save and Download Expenses**: Saves the expenses and budgets to a CSV file, which can be downloaded.
- **6. Clear All Records**: Clears all the expenses and budget data stored in the program.
- **7. Exit**: Exits the program.

2. main() Function

The main() function is the entry point for running the program and handles the core logic. It enters an infinite loop where the menu is displayed, and the user is prompted to choose an option. Based on the input, the program performs various tasks by calling the corresponding functions.

Here's how the main() function works:

- **Loading Existing Data**: The load_expenses() function is called at the beginning of the program to load any previously saved expenses and budget data from a CSV file.
- **Menu Display**: Inside the loop, the display_menu() function is called to display the menu options to the user.
- **User Input**: The user is asked to choose an option by entering a number. The program then uses a series of if-elif conditions to perform the corresponding action.
    - **Option 1: Set Budget**: Calls set_budget(), which prompts the user to define a budget for a particular month and year.
    - **Option 2: Add Expense**: Calls add_expense() to allow the user to input an expense for a specified month.

    - **Option 3: View Expenses**: Calls view_expenses() to display the expenses for a particular month.
    - **Option 4: Track Expenses**: Calls track_expenses() to calculate the total expenses for a given month and compares it against the budget.
    - **Option 5: Save and Download**: Calls save_expenses() to save the data to a CSV file and download_csv() to provide a downloadable link (in Jupyter Notebook environments).
    - **Option 6: Clear Records**: Calls reset_records() to clear all the expenses and budget data.
    - **Option 7: Exit**: Calls save_expenses() to save data before exiting the program.
- **Error Handling**: If the user enters an invalid option, the program displays an error message and prompts them to try again.

3. Menu Options and Their Functions

Here is a brief overview of each menu option and the related functions:

- **Option 1: Set Monthly Budget**
    - This calls set_budget(), which prompts the user for a year, month, and budget amount. It stores the budget data in the budgets dictionary, keyed by the year and month.
- **Option 2: Add Expense**
    - This option calls add_expense(), where the user can input the date, category, amount, and description of the expense. The function stores each expense in the expenses dictionary, categorized by year and month.
- **Option 3: View Expenses for a Specific Month**
    - The view_expenses() function is triggered, which prompts the user to input a specific year and month. It then retrieves and displays all the expenses recorded for that month.
- **Option 4: Track Expenses for a Specific Month**
    - The track_expenses() function is invoked, which calculates the total expenses for a given month and checks whether they exceed the budget (if one is set). It displays the total amount spent, remaining budget, and alerts the user if the budget has been exceeded.
- **Option 5: Save and Download Expenses**
    - This option calls save_expenses() to write all expenses and budget information to a CSV file, and download_csv() to provide a download link for the file in a Jupyter Notebook environment.
- **Option 6: Clear All Records**

    - The reset_records() function is used to clear all the stored expenses and budget data. This is useful if the user wants to reset the program and start fresh.

- **Option 7: Exit**
    - Before exiting the program, the save_expenses() function is called to ensure that any data entered is saved to a file.

Summary

The code is designed to help users manage their personal finances by allowing them to track their monthly budgets and expenses. It provides the functionality to:

- Set budgets for specific months.
- Record detailed expenses with categories and descriptions.
- Track expenses against the budget.
- Save and download the recorded data in CSV format.

The program runs interactively, allowing the user to choose different options from the menu. The logic ensures that users can view their financial data and keep it up-to-date. It also provides a mechanism to reset or clear records when needed. The interactive nature of the program makes it user-friendly, and the ability to save and download data ensures that users don't lose their financial information between sessions.