# Personal Expense Tracker

# Problem statement:

In today’s fast-paced world, individuals need to track and manage their expenses effectively. Your task is to build a personal expense tracker that allows users to log daily expenses, categorize them, and track spending against a monthly budget. The tracker should also be able to save and load expenses from a file for future reference.

# Objectives:

1. Design and implement a personal expense tracker that enables users to manage their expenses
2. Allow users to categorize expenses and set monthly budgets
3. Implement file-handling functionality to save and load expense data
4. Create an interactive, menu-driven interface for ease of use

# Steps to perform:

1. **Add an expense:**

* Create a function to prompt the user for expense details. Ensure you ask for:
  + The date of the expense in the format YYYY-MM-DD
  + The category of the expense, such as Food or Travel
  + The amount spent
  + A brief description of the expense
* Store the expense in a list as a dictionary, where each dictionary includes the date, category, amount, and description as key-value pairs

Example:

# {'date': '2024-09-18', 'category': 'Food', 'amount': 15.50, 'description': 'Lunch with friends'}

1. **View expenses:**

* Write a function to retrieve and display all stored expenses
  + Ensure the function loops through the list of expenses and displays the date, category, amount, and description for each entry
* Validate the data before displaying it
  + If any required details (date, category, amount, or description) are missing, skip the entry or notify the user that it’s incomplete

# Set and track the budget:

* Create a function that allows the user to input a monthly budget. Prompt the user to:
  + Enter the total amount they want to budget for the month
* Create another function that calculates the total expenses recorded so far
  + Compare the total with the user’s monthly budget
  + If the total expenses exceed the budget, display a warning (Example: You have exceeded your budget!)
  + If the expenses are within the budget, display the remaining balance (Example: You have 150 left for the month)

# Save and load expenses:

* Implement a function to save all expenses to a CSV file, with each row containing the date, category, amount, and description of each expense
* Create another function to load expenses from the CSV file. When the program starts, it should:
  + Read the saved data from the file
  + Load it back into the list of expenses so the user can see their previous expenses and continue from where they left off

# Create an interactive menu:

* Build a function to display a menu with the following options:
  + Add expense
  + View expenses
  + Track budget
  + Save expenses
  + Exit
* Allow the user to enter a number to choose an option
* Implement the following conditions:
  + If the user selects option 1, call the function to add an expense
  + If the user selects option 2, call the function to view expenses
  + If the user selects option 3, call the function to track the budget
  + If the user selects option 4, call the function to save expenses to the file
  + If the user selects option 5, save the expenses and exit the program