Project Report: Expense Tracker

---

1. Introduction

The **Expense Tracker** is a Python-based application designed to help users manage their daily expenses. By leveraging **Object-Oriented Programming (OOP)** principles, the program organizes expenses effectively and provides functionalities to add, remove, save, and display expenses, as well as calculate the total amount spent. The data is persisted in a text file to ensure expenses are not lost between sessions.

---

2. Objectives

- **Primary Objective**: To create a user-friendly expense management system that tracks and organizes financial data efficiently.
- **Secondary Objectives**:
    - Implement file handling to save and retrieve expenses.
    - Provide a menu-driven interface for easy user interaction.
    - Use OOP principles like inheritance and encapsulation for modularity and clarity.

---

3. Features

1. **Add Expense**: Allows users to input details about an expense, including its name, amount, and category.
2. **Remove Expense**: Enables users to delete an expense by its name.
3. **Display Expenses**: Shows a list of all recorded expenses.
4. **Calculate Total**: Calculates and displays the total of all expenses.
5. **Save Expenses**: Persists expense data to a text file (`expenses.txt`).
6. **Load Expenses**: Loads previously saved expense data from a file.

---

4. Design and Implementation

The Expense Tracker project is built using the **Object-Oriented Programming (OOP)** paradigm. It includes the following core classes:

4.1 Class: ExpenseBase

- **Purpose**: Base class representing an expense.
- **Attributes**:

- `name`: Name of the expense.
- `amount`: Amount spent on the expense.
- `category`: Category of the expense.

4.2 Class: Expense

- **Purpose**: Derived class that overrides `__str__` to format expense details.
- **Attributes**: Inherits all attributes from `ExpenseBase`.

4.3 Class: FileHandler

- **Purpose**: Handles file operations such as saving and loading expenses.
- **Methods**:
  - `save_expenses_to_file(filename)`: Saves expense data to a file.
  - `load_expenses_from_file(filename)`: Loads expense data from a file.
  - `display_expenses()`: Displays all expenses.
  - `calculate_total()`: Calculates and displays the total expense amount.

4.4 Class: ExpenseTracker

- **Purpose**: Main class for managing expenses, inheriting methods for file handling.
- **Attributes**:
  - `expenses`: A list that stores all `Expense` objects.
- **Methods**:
  - `add_expense(name, amount, category)`: Adds a new expense to the list.
  - `remove_expense(name)`: Removes an expense by its name.

---

5. Menu-Driven Interface

A user-friendly menu system provides access to the program's functionalities:

- Users can navigate through the options by entering a choice from 1 to 7.
- The menu allows for adding, removing, displaying, saving, loading expenses, and calculating totals.

---

6. Code Overview

Key Highlights

1. **Inheritance**: `ExpenseTracker` inherits from `FileHandler` to manage expenses and file operations seamlessly.
2. **Encapsulation**: Expense details are encapsulated within the `Expense` class.

3. **File Handling**: Uses Python's `open()` function to persist data between sessions.
4. **Dynamic Data Manipulation**:
    - Expenses are dynamically added or removed from the list.
    - Totals are recalculated based on current data.