**Expense Tracker Application Documentation**

**Overview**

The Expense Tracker is a simple application built using Python and the Tkinter library. It allows users to track their daily expenses by entering details such as amount, category, and description. The application provides features like adding, editing, and deleting expenses, as well as displaying the total expenses.

**Program Structure**

The Expense Tracker application consists of the following components:

**1. Main Application Class (ExpenseTrackerApp):**

* This class represents the main application window.
* It contains methods to initialize the GUI, handle user inputs, display expenses, and manage data persistence.

**Key methods:**

* \_\_init\_\_(self, root): Initializes the main application window, creates input fields, display frames, and initializes expenses list.
* add\_expense(self): Adds a new expense to the list and updates the display.
* populate\_expenses\_display(self): Populates the display frame with expense details.
* save\_expenses(self): Saves the expenses list to a JSON file.
* load\_expenses(self): Loads expenses from the JSON file.

**2. Edit Expense Window Class (EditExpenseWindow):**

* This class represents a pop-up window for editing expense details.
* It contains methods to create input fields for editing and saving edited expense data.

**Key methods:**

* \_\_init\_\_(self, root, expense\_data, callback): Initializes the edit expense window with input fields.
* save\_edited\_expense(self): Validates and saves edited expense data.

**3. Main Function (main):**

* This function is the entry point of the application.
* It creates the main application window and starts the event loop.

**Functionality**

**Adding Expense:**

Users can input expense details such as amount, category, and description. On clicking the "Add Expense" button, the entered expense is added to the list and displayed in the GUI.

**Editing Expense:**

Users can edit an existing expense by clicking the "Edit" button next to the expense entry. This opens a pop-up window with fields for editing. After editing, the changes are saved and reflected in the main window.

**Deleting Expense:**

Users can delete an existing expense by clicking the "Delete" button next to the expense entry. A confirmation dialog is displayed before deletion.

**Displaying Total Expense:**

The application calculates and displays the total expenses at the bottom of the main window.

**Data Persistence:**

Expense data is stored in a JSON file (expenses.json). It is loaded when the application starts and saved whenever there is a change in the expense list.

**Usage**

* Run the expence.py script.
* Enter the expense details (amount, category, description) in the respective fields.
* Click the "Add Expense" button to add the expense.
* Optionally, click the "Edit" button next to an expense to edit it.
* Optionally, click the "Delete" button next to an expense to delete it.

**Dependencies**

1. Python 3.x
2. Tkinter (standard library)
3. JSON (standard library)
4. Pandas: For data manipulation and analysis.
5. Matplotlib: For creating visualizations and charts.
6. Seaborn: For statistical data visualization.
7. Plotly: For interactive graphing and dashboards.
8. SQLite: For local storage of expense data.
9. Requests: For making HTTP requests.
10. Beautiful Soup: For web scraping.