PROJECT SPEC

Description

My idea is to build a budget tracking app to help users have better control over spending.

The functionalities would include:

1) Create categories and add spendings/purchases to it.
2) Set a budget limiter and the app would remind the users if they're going above budget.

Revision

Backend – I would be using CSV files for the backend part of the project as it simpler and user friendly.

GUI – The project would work ideally on a Tkinter GUI.

Vignette

1) **Create Categories and Add Spendings/Purchases**

Emily, a college student, logs into her expense tracker app and clicks on the "Categories" tab. She sees existing categories and selects "Add Category" to create one for her coffee expenses. After entering "Coffee," she saves it, and it appears in her list. She then navigates to "Add Spending," selects "Coffee," inputs her spending amount, and adds a note. Upon saving, she receives confirmation, and her spending updates in her financial summary.

Technical Details:

Revision

Change in use of flask to CSV - Simplified data storage using CSV files instead of a database like SQLite. Data is read from and written to the CSV file directly, offering easier file-based operations for the project timeline.

2) Set a Budget Limiter and Receive Reminders

Emily goes to the "Budget" section to set a monthly limit for her coffee expenses. She inputs $100 for "Coffee" and clicks "Set Budget." As she adds purchases, the app tracks her spending. When she nears her limit, a notification pops up: "You are nearing your budget limit for Coffee. Current spending: $90. Remaining: $10." This helps her control her spending.

Technical Flow(Blocks)

1) **User Interface (UI)**

Function: Handles user input and displays output.
User Actions: Inputs for categories, spendings, and budgets; outputs for confirmations and notifications.
Data Types: Strings and numbers.

Revision

2) **Data flow**

User Input (via Tkinter UI)

- User inputs expense data (category, date, amount, description) or budget limit.
- User actions trigger the writing or reading of data from the appropriate CSV file.

Write to CSV (Add Expense or Set Budget)

- For a new expense, the app uses Python's csv.writer() to append the data (category, date, amount, description) as a new row in the **expenses.csv** file.

- For a budget limit, the app writes the category and budget amount to the **budget.csv** file.

Read from CSV (Display Data)

- When displaying expenses or categories, the app uses csv.reader() to read from the **expenses.csv** or **categories.csv** files.
- The data is processed and displayed in the UI as a list of expenses or available categories.

Budget Check (Alert if Budget Exceeded)

- The app reads the **budget.csv** file and compares the budget limit for a category with the total spending in that category.
- If the sum of expenses in the **expenses.csv** file exceeds the budget, the app displays an alert.

Revised Data Flow

1. Creating a Category:

Input: User enters a new category name.

Flow:

- **UI (Tkinter):** User inputs the category name and budget amount.
- **Controller:** The app processes the input and checks if the category already exists in the CSV.
- **CSV (expenses.csv):** The app adds a new row for the category with the budget specified.
- **UI (Tkinter):** Displays confirmation that the category has been successfully created.

2. Adding a Spending Entry:

Input: User selects a category and enters the amount, and description for the expense.

Flow:

- **UI (Tkinter):** User selects the category and inputs the expense details.
- **Controller:** The app processes and validates the data.
- **CSV (expenses.csv):** Appends the expense as a new row under the respective category, including amount and description.
- **UI (Tkinter):** Updates the summary and displays the added expense.

3. Notification System (Budget Alerts):

Function: The app monitors the spending for each category and compares it against the set budget after each entry.

Flow:

- **CSV (expenses.csv):** After each new spending entry is added, the app calculates the total expenses for the selected category by reading the CSV file.
- **Controller:** Compares the total spending in the category to the set budget.
- **UI (Tkinter):** If the total spending exceeds the budget, the app sends an alert to notify the user.

Ideas, updates for next few weeks –

UI Refinement - Improve the UI by adding a table-like display for expenses.

Data Filtering – Filter option to categorize expenses based on amount(lower, greater).

Proposed Features -

Expense Breakdown - Implement a visual chart (pie chart, etc) to represent category-wise expense breakdown.

Summary and Reporting - Implement a reporting feature that generates summary reports of a users budget and expenses.

Link to Repository –

https://github.com/sandeep711/HCI584-Project