Original Sketch 1: Personal Budgeting and Expense Tracker

Description:

The Personal Budgeting and Expense Tracker app is designed to help users manage their finances more effectively. The app allows users to set budgeting goals, track their spending habits, and receive insights on how to save money more efficiently.

Use Case:

This app enables users to log their daily expenses, categorize them, and monitor where their money is being spent. Additionally, the app offers tips on reducing unnecessary expenses. Managing finances adhering to a budget.

Challenges:

The primary challenge will be ensuring that the app remains simple enough for everyday use while providing powerful insights into spending habits. Another challenge is securely

integrating the app with users' bank accounts for automatic expense tracking.

Users and Stakeholders:

• **Primary Users:** Individuals looking to improve their financial management skills.

• **Secondary Stakeholders:** Financial institutions and developers of budgeting tools.

Task/Problem:

The app addresses the issue of financial disorganization by offering a clear and easy method to track spending, set budgets, and save money.

Data Used:

The app utilizes data from bank transactions, manually entered expenses, and budgeting goals to generate detailed reports and insights on users' financial health.

Results:

Users will gain a better understanding of their spending patterns and develop a more disciplined approach to saving money. The app presents these results through clear charts, progress bars, and spending summaries.

Feedback – Chris Harding

I like #1 best but you need to think about how spending habits work. I would advise against a connection with a bank account and rather just enter the value from it by hand. Or at least leave that as a maybe for later if everything else works. You could have several categories (maybe user defined?) and each could have a limit. you will also need to enter a date for each expense. Finally, it'd be not too difficult to visualize spending over time including some sort of warning when you're over budget. Nice work.

Reworked Project Sketch

Description: The Personal Budgeting and Expense Tracker app is designed to help users manage their finances effectively. It enables users to set budgeting goals, log expenses manually, categorize spending, and receive insights on how to save money efficiently.

Use Case: This app allows users to track their daily expenses by entering them manually, categorizing each expense (user-defined categories), and monitoring their spending habits. Users can set budget limits for each category and receive alerts when they approach or exceed those limits.

Users and Stakeholders:

- **Primary Users:** Individuals looking to improve their financial management skills.
- **Secondary Stakeholders:** Financial educators, budgeting tool developers, and potential sponsors interested in financial literacy.

Task/Problem: The app addresses financial disorganization by providing a clear and straightforward method for users to log their expenses, set budget limits, and save money.

Data Used: The app utilizes manually entered expense data, user-defined categories, and budgeting goals to generate detailed reports and insights on users' financial health. Each expense entry will require a date for better tracking.

Results: Users will gain a clearer understanding of their spending patterns and develop a more disciplined approach to saving money. The app will visualize spending over time through charts and graphs, displaying spending summaries and providing warnings when users exceed their budget limits.

User Categories examples:

Food,

Housing,

Transportation,

Entertainment,

Health and Fitness,

Personal Care,

Education,

Savings and Investments

Others/Miscellaneous

PROJECT SPECS

General Description of the Project

The Personal Budgeting and Expense Tracker app is a user-friendly tool designed to help people manage their money. It allows users to manually enter their daily expenses, categorize them (like food, entertainment, or bills), and set spending limits for each category. You can also add the date for each expense. The app provides easy-to-read charts and summaries to help you understand your spending habits. This app makes budgeting easy and supports you in reaching your financial goals.

Technical Details

Flask for building web applications.

One for displaying chart (Bar Chart, Pictogram or Pie Chart??) of spending habits.

Vignette of tasks users will have to complete

1. Logging an Expense

User is trying to keep track of her spending to save for a vacation. They would open the personal budgeting and expense tracker app and navigate to the section for logging an expense. They add the amount, category, and date of expense. They enter the amount for lunch, select "Food" from the user-defined categories, and input the date. After clicking “Save,” the app confirms expense has been recorded.

Input – Amount, Category and Date

Output – Recorded data from users input and Summary displayed

Modules Involved – Flask, Storage of data, Summary of data recorded

2. Setting a Budget Goal

User wants to stick to a budget this month. They open the app and go to the section for budget goal. Here, they can see the current spending limits for each category. They decide to set a

budget for “Entertainment” category. They type in the amount and click “Set Budget.” The app shows a confirmation message and updates their budget overview.

Inputs – Entertainment, Amount

Outputs - Budget goal saved to the database and Updated budget overview displayed

Modules Involved – Flask, Storage of data, Summary of data recorded

3. Viewing Spending Insights

A user is interested in their spending habits. They click on the "Insights" tab in the app. The app generates a summary of their spending for the month and visualizes it with charts. They can see that they have spent the most on “Shopping” and are nearing their budget limit.

Input – User's expense data for the week or month

Output – Visual Charts and Summary displayed

Modules Involved – Flask, Storage of data, Visualization

Data Flow Overview

1. **User Interface (UI)**:
    - **Function**: Collects user inputs (expenses, budgets) and displays outputs (insights).
    - **Interaction**: User input (text fields for expenses/budgets), output (visuals, summaries).
    - **Data Types**: Strings (for categories and values), dates.
2. **Flask Web Framework**:
    - **Function**: Manages HTTP requests and responses.
    - **Interaction**: Receives inputs from the UI, sends data to the business logic.
    - **Data Types**: JSON.
3. **Business Logic Layer**:
    - **Function**: Processes data, validates inputs, calculates insights.

- **Interaction**: Receives inputs from the Flask layer, accesses the database, and returns processed results.

- **Data Types**: Dictionaries (for expense and budget details), lists (for expenses), numerical data (for calculations).

4. **SQLite Database**:

   - **Function**: Stores user data (expenses, budgets) persistently.

   - **Interaction**: Receives data from the business logic (inserts/updates) and provides data for analysis.

   - **Data Types**: Tables (structured data, rows for records).

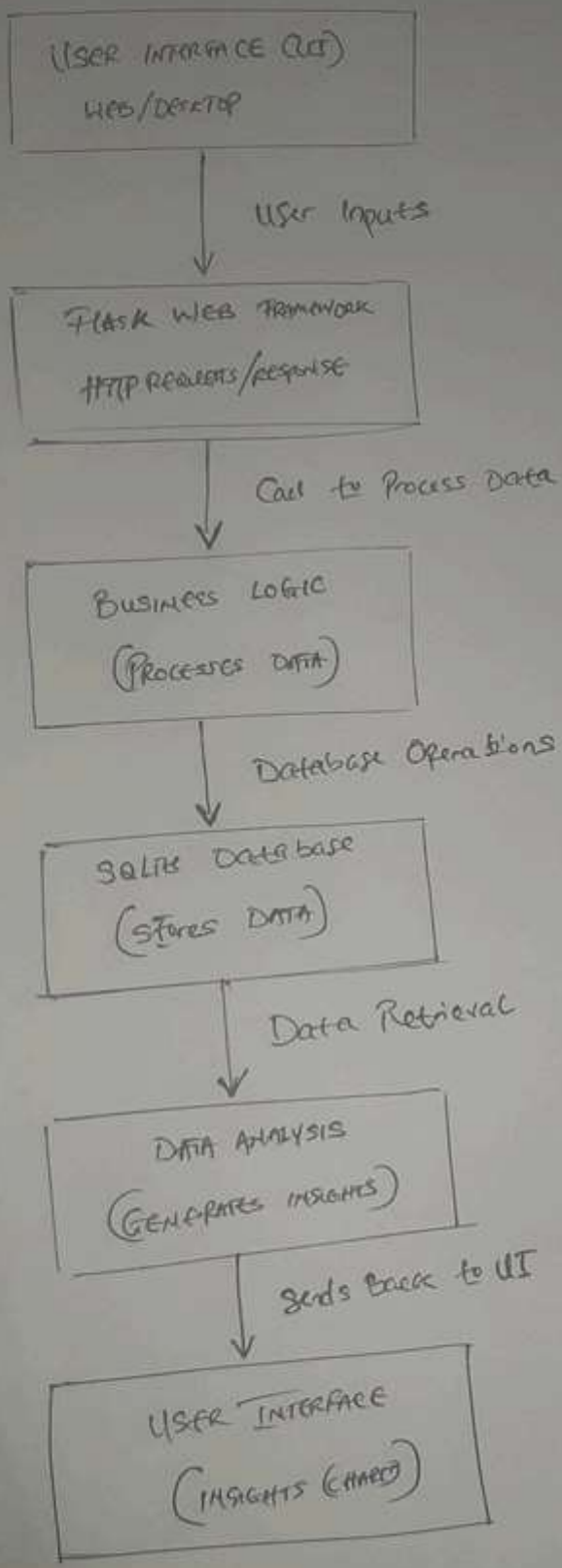
5. **Data Analysis Module**:

   - **Function**: Analyzes user data and generates insights/visualizations.

   - **Interaction**: Pulls data from the database and returns visual representations to the business logic.

   - **Data Types**: Images (for visualizations).

DATA FLOW DIAGRAM



Python Code for Class

```
class UserInterface:
        def collect_input(self):
        # Code to collect user inputs
        pass

        def display_output(self, insights):
        # Code to display results to the user
        pass

class FlaskApp:
        def handle_request(self, request):
        # Code to process incoming HTTP requests
        pass

class BusinessLogic:
        def validate_data(self, data):
        # Code to validate user inputs
        pass

        def calculate_insights(self, expenses):
        # Code to analyze expenses
        pass

class Database:
        def insert_expense(self, expense):
        # Code to insert expense into database
        pass

        def retrieve_expenses(self):
        # Code to retrieve expenses from database
        pass

class DataAnalysis:
        def generate_visuals(self, expenses):
        # Code to create charts and insights
        pass
```

Conclusion

Initially, I considered integrating automatic bank account syncing for expense tracking. However, based on feedback, I've adjusted to manual entry of data, simplifying the design and focusing on user-defined categories.

I believe that creating information charts (like bar chart or pie chart) would be challenging.

FeedBack - Chris Harding

Good job, some suggested changes:

- no need for a real SQL database, just use a excel of csv file and read/write/process that in a pandas dataframe.
- Don't worry about the UI for version 1, just write the functions.classes/methods you need and simulate user input via hard coding
- For version 2 you should look into Streamlit which I think is easier to work with than Flask. Or you could just use TkInter/ttk (see my tab demo!), there's really no need to make this a Webapp.
- Data validation: in your version 1 this can be omitted and for version 2 just make sure to do most of the on the front end or but choosing widgets that are fool proof.
- if you use a dataframe for you data and also add a data for each transaction, graphing expenses over time or in aggregate is not tricky.
- Watch attached video for more details

2024-09-24 14-25-34.mp4