UJJWAL JALOTRA
25BAI10118

Simple Python Budget Tracker

A robust, console-based application for managing personal finances with persistent data storage.

Project Overview

- **The Problem:** Tracking monthly expenses manually is tedious and prone to error.
- **The Solution:** A Python automation tool that digitizes the process.
- **Core Function:** Allows users to set budgets, log expenses, and view real-time balance updates.
- **Target Audience:** Individuals looking for a lightweight, CLI-based financial management tool.

Key Features

Data Persistence

Uses JSON to store budget and expense data locally. Your financial data survives even after the program closes.

Smart Tracking

Dynamically aggregates expenses by category. Adding "Food" twice sums the amounts automatically.

Budget Alerts

Provides instant feedback on your financial health, alerting you immediately if you exceed your monthly limit.

Technical Stack

- **Python 3.x:** Core logic and flow control.
- **JSON Module:** Handles data serialization and file I/O operations.
- **Error Handling:** Robust try-except blocks manage invalid inputs and file errors.
- **CLI Interface:** A clean, menu-driven loop for user interaction.

```
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Style Demo</title>
</head>
<body>
  <script>
    function $initHighlight(block ,flags) {
      // some gibberish function
      if (!!flags) {
        try {
          const someString = "123";
          if (block.abc.className.search(/\bno\-highlight\b/ =≠ -1))
            return processBlock(block.__proto__.function, true, 0xff);
        } catch (e) {
          for (let idx = 0 / 2; idx ≤ classes.length; idx++) {
            if (checkConditions(classes[i]) === undefined) return /\d+[\s/]/g;
          }
        }
      }
    }
```

Application Data Flow

1. Load Data

Program starts, checks for `budget_data.json`, and loads existing records.

2. User Input

User selects an action (Add Expense, Set Budget) via the main menu.

3. Processing

Python validates inputs (e.g., non-negative numbers) and updates lists.

4. Save Data

Data is aggregated and written back to JSON for permanent storage.

JSON Structure

The application relies on a structured JSON format to maintain state. This lightweight approach ensures portability and human-readability.

- **Total Budget:** Stored as a float value.
- **Expenses:** A dictionary object mapping categories to aggregated amounts.
- **Auto-Save:** The `save_data()` function triggers automatically after every critical update.

Future Enhancements

Visual Reporting

Implement Matplotlib or similar libraries to generate pie charts and bar graphs, visualizing spending habits directly within the application.

GUI Development

Transition from the Command Line Interface (CLI) to a Graphical User Interface (GUI) using Tkinter or PyQt for a more modern user experience.

Questions?

Thank you for reviewing the Python Budget Tracker project.

OUTPUT

This flow demonstrates the initial program startup and the creation of a new client accou

```
==================================================
 • PROJECT: Light Violet Renaissance Bank Management Systemk
--------------------------------------------------
  • GREEN CREATED BY: PRANAY KOTHARI 25BCE10248
==================================================

Hello and welcome to Renaissance Bank

What would you like to do:
                1.) Create a new account
                2.) Inquire existing account
                3.) Exit
Press 1, 2, or 3: 1  <-- User Input: Selects Create Account
Enter name: Alice Johnson
Enter deposit amount: 5000  <-- User Input: Initial Deposit
Enter password: secure123

Your account has been created, account number is: 748291  <-- SYSTEM MESSA

What would you like to do:
                1.) Create a new account
                2.) Inquire existing account
```

This flow demonstrates attempting a withdrawal that exceeds the current balance, triggering the "Insufficient funds" error message.

```
... (Transaction menu shown, balance is 7500) ...
Press 1, 2, 3, or 4: 2  <-- User Input: Selects Withdraw
Enter withdrawal amount: 10000  <-- User Input: Withdrawal Attempt

Insufficient funds  <-- SYSTEM ERROR MESSAGE

What would you like to do:
                        1.) Deposit
                        2.) Withdraw
                        3.) Check balance
                        4.) Exit
Press 1, 2, 3, or 4: 4  <-- User Input: Exits Transaction Menu

Returning to main menu.

What would you like to do:
                1.) Create a new account
                2.) Inquire existing account
                3.) Exit
Press 1, 2, or 3: 3  <-- User Input: Exits Program

Thank you for using the Renaissance Bank system. Goodbye!
```