# Finance Project

Project Summary

The project combines three functionalities into a single application: a transaction tracker for daily transactions, a budget manager for managing financial budgets and tracking spending against set budgets and a basic arithmetic calculator.

System Architecture

1. Software Modules
   * Transaction Tracker: Manages and records daily financial transactions. It supports adding, viewing, and searching transactions and can categorize expenditures and income.
   * Budget Manager: Allows users to set, update, and view budgets. It calculates spending relative to the budgets and provides financial summaries.
   * Calculator Program: Provides arithmetic operations such as addition, subtraction, multiplication, division, power, and square root.
2. Hardware Components
   * Processor: Any modern processor capable of running Python.
   * Memory: Minimum of 1GB RAM, though more is recommended for larger data sets.
   * Storage: Sufficient disk space to store application files and user data
3. User Interface
   * Command-Line Interface (CLI): Text-based interface providing menu options to interact with different modules (Transaction Tracker, Budget Manager, Calculator).

Hardware, Software, and System Requirements

* + Hardware: Any standard PC or laptop with basic specifications.
  + Software: Python 3.12 environment.
  + System Requirements:
    1. Python 3.12 installed
    2. Basic text editor or IDE for code editing

Software Design

A diagram of a business process

Description automatically generated

Class Diagram

A diagram of a computer

Description automatically generated with medium confidence

Class Specifications

* Application
  + Attributes:
    - The main menu does not have attributes since it is a function managing the flow of user interaction.
  + Methods:
    - mainMenu() is a public method that provides a command-line interface for users to interact with the application. It manages user choices and calls appropriate methods from TransactionTracker, BudgetManager, and Calculator.
* TransactionTracker
  + Attributes:
    - expenses and incomes are lists that hold dictionaries with transaction details. These attributes are private (denoted by -), meaning they are intended to be accessed only within the class.
  + Methods:
    - addExpense() and addIncome() are public methods (denoted by +) used to add new entries to the lists. They take various parameters to capture the details of transactions.
    - view() is a public method that prints out the summary of all transactions.
    - search() is a public method used to find and display transactions that match a given search term
* Management
  + Attributes:
    - budgetList is a private list that holds dictionaries with budget details.
  + Methods:
    - setBudget() is a public method used to add or update the budget for a specific category. It takes the category, budget amount, and an optional description as parameters
    - viewSummary() is a public method that provides a summary of budgets compared to the expenses recorded in the TransactionTracker. This method takes a TransactionTracker instance as a parameter to access the expense data.
* Calculator
  + Attributes:
    - The Calculator class does not have attributes as it performs calculations based on the inputs provided
  + Methods:
    - Each method is public (denoted by +), allowing users to perform basic arithmetic and other mathematical operations.
    - Add, subtract, multiply, and divide perform basic arithmetic operations.
    - Power calculates the power of a number.
    - Square computes the square root of a number.

Interaction Diagrams

1. Transaction Tracking Interaction Sequence

A diagram of a person's process

Description automatically generated

* Adding an Expense or Income:
  + The **user** inputs transaction data.
  + The **application** calls the addExpense() or addIncome() method of TransactionTracker, passing the required parameters.
  + The **TransactionTracker** stores the new expense or income data in its internal expense or income list.
* Viewing Transactions:
  + The **user** requests to view all transactions.
  + The **application** invokes the view() method of TransactionTracker.
  + The **TransactionTracker** retrieves and returns all stored expenses and incomes to the **application**, which then displays the list to the **user**.
* Searching Transactions:
  + The **user** requests a search, providing a search term.
  + The **application** calls the search() method of **TransactionTracker**  with the search term.
  + The **TransactionTracker**  returns matching transactions, which the **application** displays to the **user.**

1. Budget Management Interaction Sequence

A diagram of a person

Description automatically generated

* User: The user selects the option to manage budgets from the main menu.
* Application: The application presents options like setting budgets or viewing current budget status.
* User: The user sets a budget for different expense categories.
* Application: The application forwards this data to the BudgetManagement module for processing.
* BudgetManagement: It updates and calculates the budget based on the user’s actions.
* BudgetManagement: The BudgetManagement returns the updated data (remaining budget, spending limits, etc.)
* Application: The final budget information is shown to the user in the app interface

1. Simple Calculator Interaction Sequence

A diagram of a person with a diagram

Description automatically generated

* User: The user selects the option to use the calculator from the main menu.
* Application: The application receives the user’s selection and initiates the Calculator module.
* Calculator: The Calculator module displays the calculator interface to the user.
* User: The user enters a calculation (e.g., addition, subtraction)
* Calculator: The Calculator module performs the calculation and returns the result to the user interface.
* Calculator: The Calculator module displays the result of the calculation to the user.

Design Considerations

1. Error Handling: Proper input validation to prevent invalid or malicious data entry.
2. User Experience: Intuitive CLI menu structure for ease of navigation.
3. Data Persistence: Optional file handling for saving data (not currently implemented)

User interface Design

1. Main Menu: Provides options to navigate to Transaction Tracker, Budget Manager, or Calculator.
2. Transaction Tracker Menu: Allows adding, viewing, and searching transactions.
3. Budget Manager Menu: Facilitates setting budgets and viewing budget summaries.
4. Calculator Menu: Provides options for various arithmetic operations.

Glossary of Terms

1. CLI: Command-Line Interface, a text-based interface for interacting with the application.
2. Transaction: An entry recording an expense or income in the financial tracker.
3. Budget: A financial plan that estimates income and expenditures over a period.
4. Data Persistence: Saving data to a file or database for future retrieval.

Reference

<https://github.com/Kngk0/financeproject.git>

“Python Tutorial.” Python Tutorial, www.w3schools.com/python/default.asp. Accessed 8 May 2024.