Project Proposal

Project Description

The project aims to create a command-line interface for managing personal expenses and budgets. The user can add an expense, view expenses, generate reports, set budgets for different categories, and view budgets for each category. The project will use a simple data structure. Problem to be Solved: Managing personal expenses and budgets can be a tedious and time-consuming task. It is easy to lose track of spending and exceed budget limits. This project aims to provide an easy-to-use tool to help users manage their expenses and budgets. Technical Objectives: Create a data structure to store expense and budget data Implement functions to add expenses, view expenses, generate reports, set budgets, and view budgets Implement user authentication to ensure only authorized users can access the system Implement error handling to ensure the system does not crash due to invalid inputs or missing files How the Project Will be Done: The project will be implemented in Python. The user interface will be a command-line interface. The project will use several functions to add expenses, view expenses, generate reports, set budgets, and view budgets. The system will use user authentication to ensure only authorized users can access the system. The system will also implement error handling to prevent the system from crashing due to invalid inputs or missing files. The project will be tested to ensure it meets the functional requirements and is user-friendly.

IPO:

Input: User input for adding a new expense: date of expense, amount, currency, category, and description. User input for generating reports: type of report (monthly, category-wise, or total), month (for monthly report), and category (for category-wise report). User input for setting a budget: category and budget amount.

Process: The program maintains a list of expenses, categories, budgets, and exchange rates. When a user adds a new expense, the program prompts the user for input, converts the amount to the user's main currency, and saves the expense data to the list. When a user generates a report, the program prompts the user for input, filters the expenses based on the user's input, and calculates the total expenses. When a user sets a budget, the program prompts the user for input and updates the budget dictionary.

Output: The program outputs a success message when an expense is added. The program outputs a list of expenses when the user chooses to view expenses. The program outputs the total expenses for a month or a category, or the total expenses overall when the user generates a report. The program outputs a success message when a budget is set. The program outputs a list of budgets when the user chooses to view budgets.

Methodology

The expense tracker application will be developed using the Python programming language. The project will involve the following steps:

Data Structure

The data structure will include lists to store expenses and categories, a dictionary to store exchange rates, and a dictionary to store budgets. Develop the add_expense() function - This function will allow users to input their expenses, convert them to their main currency and categorize them. Develop the view_expenses() function - This function will allow users to view their expense history. Develop the generate_reports() function - This function will allow users to generate reports based on specific criteria such as monthly, category-wise or total expenses. Develop the set_budget() function - This function will allow users to set budgets for different categories. Test the application - The application will be tested thoroughly to ensure it works as expected.

Schedule of Activities

The following is a timetable of deliverables for the expense tracker application:

Week 1: Define the data structure and develop the add_expense() function.

Week 2: Develop the view_expenses() function and the generate_reports() function.

Week 3: Develop the set_budget() function and any additional functions that the group might add

Week 4: Test the application and make any necessary adjustments.