College Student Budget & Savings Advisor

NOOLU HARI NARAYANA THUSHAR KAPIL

Registration Number: 25MEI10032
Course: Programming/Problem Solving

Introduction

- College students often struggle with managing money effectively.
- Daily expenses include food, travel, recharge, shopping, and more.
- Lack of financial awareness often leads to overspending.
- This project helps students track expenses and understand savings.

Problem Statement

Category	Icon	Description
Limited Funds	$	Students typically receive limited pocket money or income and need to make it last.
No Tracking	?	There is no systematic method for tracking small, daily expenses effectively.
Unknowns	–	Difficult to know where money goes, how much is spent monthly, and potential savings.

Project Objective

- **Build a Python Program:** Create a robust tool to track daily expenses.
- **Provide Summaries:** Offer category-wise and monthly spending reports.
- **Compare Data:** Analyze income versus total spending to gauge financial health.
- **Recommend Savings:** Give actionable advice to encourage financial awareness.

Key Features

- **Input & Tracking:** Easily input income and all major expense categories to calculate left-over funds.
- **Summary & Advice:** Generates an immediate budget summary and provides financial health feedback.
- **Savings Advisor:** Applies the 20% savings rule to recommend a target savings amount.

- **Goal Planner:** Calculates the time needed to reach a specific savings goal based on current habits.

System Design: Program Flow

Start → Input (Income & Expenses) → Process (Calculate Total & Leftover) →
Advise (Health & Savings Target) → Goal (Completion Time)

Code Snippet 1: Input & Expense Entry

```
def college_budget_advisor():
    """Main function for budget tracking."""
    print("\n--- College Student Budget & Savings Advisor ---\n")

    # 1. Monthly Income Input
    while True:
        try:
            income = float(input("Enter your total monthly income: $"))
            if income < 0:
                print("Income cannot be negative.")
                continue
            break
        except ValueError:
            print("Invalid input. Please enter a numeric value.")

    # 2. Expense Categories Input
    print("\nEnter your estimated monthly expenses:")

    def get_expense(category):
        while True:
            try:
                amount = float(input(f" {category}: $"))
                if amount < 0:
                    print("Expense cannot be negative.")
                    continue
                return amount
            except ValueError:
                print("Invalid input.")

    rent = get_expense("Rent / Housing")
    food = get_expense("Food / Groceries")
    transportation = get_expense("Transportation")
    utilities = get_expense("Utilities / Phone / Internet")
    entertainment = get_expense("Entertainment")
    others = get_expense("Others")
```

Listing 1: Part 1: Defining the main function and handling income/expense inputs

Code Snippet 2: Calculation & Summary

```
    # 3. Calculation
    total_expenses = rent + food + transportation + utilities + entertainment + others
    leftover = income - total_expenses

    # 4. Summary
    print("\n--- Monthly Budget Summary ---")
    print(f"Total Income:        ${income:.2f}")
    print(f"Total Expenses:      ${total_expenses:.2f}")

    # Display the breakdown
    print("\n Expense Breakdown:")
    print(f" Housing:           ${rent:.2f}")
    print(f" Food:              ${food:.2f}")
    print(f" Transportation:    ${transportation:.2f}")
    print(f" Utilities:         ${utilities:.2f}")
    print(f" Entertainment:     ${entertainment:.2f}")
    print(f" Others:            ${others:.2f}")
    print("-" * 28)
    print(f"Money Left Over:     ${leftover:.2f}")
```

Listing 2: Part 2: Calculating total expenses and displaying the summary

Code Snippet 3: Logic & Goals

```
    # 5. Financial Advice
    print("\n--- Financial Advice ---")

    if leftover > 0:
        print("\nSuccess! You are spending less than you earn.")
        recommended_savings = income * 0.20
        print(f"- Target monthly savings (20%): ${recommended_savings:.2f}")

        if leftover >= recommended_savings:
            print("- Excellent! You meet the savings target.")
        else:
            print("- Warning: You are below the 20% savings target.")

    elif leftover == 0:
        print("\nWarning: You are breaking even.")
    else:
        print(f"\nWarning: You are spending ${abs(leftover):.2f} MORE than you earn!")

    # 6. Simple Goal Planner
    print("\n--- Savings Goal Planner ---")

    goal = input("Enter a savings goal: ")
    while True:
        try:
            goal_amount = float(input(f"Goal amount for '{goal}' ($): "))
            if goal_amount < 0:
                print("Goal amount cannot be negative.")
                continue
            break
        except ValueError:
            print("Invalid input. Please enter a numeric value.")

    if leftover > 0:
        months_needed = goal_amount / leftover
        print(f"\nAt ${leftover:.2f}/mo, you reach '{goal}' in {months_needed:.1f} months
            .")
    else:
        print("You need positive savings to calculate a timeline.")

    print("\n--- End of Report ---\n")

# Run the program (Note: full function definition required for actual execution)
```

```
if __name__ == "__main__":
    pass
```

Listing 3: Part 3: Providing financial advice and setting savings goals

Results & Conclusion

Results

- Students can effectively track all daily spending.
- Category-wise insights help identify areas of overspending instantly.
- Savings advice provides actionable financial guidance.

Conclusion

- Project solves a real-world problem for students.
- Simple, effective, and practical implementation.
- Reinforces key programming concepts and logical thinking.

Contact Information

NOOLU HARI NARAYANA THUSHAR KAPIL
25MEI10032