

Advanced Prompting: Problem Decomposition Prompting

Automated Expense Tracker Development

Task:

Guide an LLM in designing a simple program that automatically tracks and categorizes personal expenses. Begin by using Program-of-Thoughts Prompting to break down the project into coding tasks such as inputting expense data, categorizing expenses, etc. Each step involves writing specific code segments, which are then executed to build the complete program.

Main prompt:

Act as a software engineer. Design an outline for an automated expense tracker that helps users to track and categorize their personal expenses more efficiently. Please write easily understandable code. Breakdown the project into coding tasks and next use them to develop a complete automated expense tracker simple program.

Main respond:

Designing an outline for an automated expense tracker involves several key components to handle user inputs, categorize expenses, and provide useful feedback to the user. I'll break down this project into manageable coding tasks and then integrate them into a simple yet complete program.

Project Breakdown into Coding Tasks

1. **User Input Handling**

- Task: Create a function to capture user inputs for expense entries. Inputs might include the amount spent, the date of the transaction, and a brief description.

2. ****Expense Categorization****

- Task: Implement a basic categorization system that assigns each expense to a category based on keywords in the description.

3. ****Data Storage****

- Task: Design a way to store and retrieve expense entries, ideally in a structured format like a list or a database.

4. ****Reporting and Analysis****

- Task: Develop functions to generate reports, such as total expenses by category or spending over time.

5. ****User Interface****

- Task: While a full graphical user interface (GUI) is complex, we can simulate a simple text-based interface in the console for input and output.

6. ****Integration and Testing****

- Task: Ensure all components work together seamlessly and perform tests to validate functionality.

Development of an Automated Expense Tracker Program

Given the scope and the request for simplicity, let's focus on a basic but complete implementation that covers these tasks, using Python for its readability and ease of use.

```

python

# Simplified Automated Expense Tracker Program

# Global list to store expenses

expenses = []

# Task 1: User Input Handling

def input_expense():

    amount = float(input("Amount spent: "))

    date = input("Date (YYYY-MM-DD): ")

    description = input("Description: ")

    expense = {'amount': amount, 'date': date, 'description': description, 'category':
categorize_expense(description)}

    expenses.append(expense)

    print("Expense added successfully.")

# Task 2: Expense Categorization

def categorize_expense(description):

    categories = {'Food': ['coffee', 'restaurant', 'groceries'],

                  'Transport': ['uber', 'bus', 'subway', 'gas'],

                  'Utilities': ['electricity', 'internet', 'water']}

    for category, keywords in categories.items():

        if any(keyword in description.lower() for keyword in keywords):

```

```
        return category

    return 'Other'
```

Task 3: Data Storage

Using a global list `expenses` as the storage method here.

Task 4: Reporting and Analysis

```
def generate_report():

    category_totals = {}

    for expense in expenses:

        category = expense['category']

        category_totals[category] = category_totals.get(category, 0) + expense['amount']

    for category, total in category_totals.items():

        print(f"{category}: ${total:.2f}")
```

Task 5: User Interface

```
def main_menu():

    while True:

        print("\nAutomated Expense Tracker")

        print("1. Add an Expense")

        print("2. Generate Report")

        print("3. Exit")

        choice = input("Choose an option: ")
```

```
if choice == '1':  
    input_expense()  
  
elif choice == '2':  
    generate_report()  
  
elif choice == '3':  
    print("Exiting.")  
    break  
  
else:  
    print("Invalid option, please try again.")
```

Task 6: Integration and Testing

```
if __name__ == "__main__":  
    main_menu()  
'''
```

This program integrates the outlined tasks into a simple automated expense tracker. Users can add expenses, which are then automatically categorized and stored. They can generate reports to see their spending by category. While basic, this program demonstrates the core functionality of an automated expense tracker and can be expanded with more sophisticated categorization logic, a graphical user interface, or integration with external data sources for more comprehensive tracking and analysis.