THIRD WEEK INTERNSHIP- PROJECT

**Aim:**

Task is the creation of a "Expense Tracker".

**Project Overview:**

The Expense Tracker project is designed to reinforce your understanding of Python programming concepts and enhance your skills in building practical applications. In this project, you will be developing an Expense Tracker application that allows users to manage their expenses efficiently. This real-world application will involve handling data, user input, and implementing key functionalities

**Project Objectives:**

To accurately outline the scope of work required for a project, it is crucial to first identify its objectives. Pinpointing what the project hopes to accomplish will assist in determining its inclusions and limitations.

User Input and Data Management: Develop a system that allows users to input their daily expenses.

Data Storage: Implement a mechanism to store and manage the entered expense data. Expense

Categories: Categorize expenses into different categories for better organization.

Data Analysis: Provide users with insights into their spending patterns, such as monthly summaries and category-wise expenditure.

User-Friendly Interface: Create a user-friendly interface for a seamless user experience. Error

Handling: Implement error handling to ensure the application can handle unexpected inputs gracefully.

Documentation: Document your code effectively to demonstrate clarity and understanding.

**Requirements and Features:**

User Input: Allow users to input their daily expenses, including the amount spent and a brief description.

Data Storage: Use appropriate data structures or file handling techniques to store and retrieve expense data.

Expense Categories: Implement the ability for users to categorize their expenses (e.g., food, transportation, entertainment).

Data Analysis: Provide users with the option to view summaries of their monthly expenses and category-wise expenditure.

User Interface: Create a simple and intuitive user interface to interact with the Expense Tracker.

Error Handling: Include error handling mechanisms to address potential issues during user interaction.

Documentation: Provide clear documentation for your code, explaining the logic behind key functions and overall program structure.

Program:

import json

from datetime import datetime

# Function to get user input for daily expenses

def get\_expense\_input():

amount = float(input("Enter the amount spent: "))

description = input("Enter a brief description: ")

category = input("Enter the category (e.g., food, transportation, entertainment): ")

timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")

return {'amount': amount, 'description': description, 'category': category, 'timestamp': timestamp}

# Function to save expenses to a JSON file

def save\_expenses(expenses):

with open('expenses.json', 'w') as f:

json.dump(expenses, f)

# Function to load expenses from a JSON file

def load\_expenses():

try:

with open('expenses.json', 'r') as f:

expenses = json.load(f)

except FileNotFoundError:

expenses = []

return expenses

# Function to display monthly summaries

def monthly\_summary(expenses):

# Calculate total spending for each month

monthly\_totals = {}

for expense in expenses:

month\_year = expense['timestamp'][:7] # Extract YYYY-MM from timestamp

if month\_year in monthly\_totals:

monthly\_totals[month\_year] += expense['amount']

else:

monthly\_totals[month\_year] = expense['amount']

# Display monthly summaries

for month\_year, total in monthly\_totals.items():

print(f"Total expenses for {month\_year}: ${total:.2f}")

# Main function

def main():

expenses = load\_expenses()

while True:

print("\n1. Add Expense\n2. View Monthly Summary\n3. Exit")

choice = input("Enter your choice: ")

if choice == '1':

expense = get\_expense\_input()

expenses.append(expense)

save\_expenses(expenses)

print("Expense added successfully!")

elif choice == '2':

monthly\_summary(expenses)

elif choice == '3':

print("Exiting...")

break

else:

print("Invalid choice. Please try again.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Output:**

1. Add Expense

2. View Monthly Summary

3. Exit

Enter your choice: 1

Enter the amount spent: 25.50

Enter a brief description: Lunch

Enter the category (e.g., food, transportation, entertainment): Food

Expense added successfully!

1. Add Expense

2. View Monthly Summary

3. Exit

Enter your choice: 1

Enter the amount spent: 50.25

Enter a brief description: Movie tickets

Enter the category (e.g., food, transportation, entertainment): Entertainment

Expense added successfully!

1. Add Expense

2. View Monthly Summary

3. Exit

Enter your choice: 2

Total expenses for 2024-06: $75.75

1. Add Expense

2. View Monthly Summary

3. Exit

Enter your choice: 3

Exiting...

This output demonstrates adding expenses, viewing a monthly summary, and exiting the program. You can continue testing and adding more features as needed for your project.

Result:

Source code and output for the Expense Tracker using the python programming language .

Top of Form

Bottom of Form