**Project: Expense Tracker**

import json

import os

from datetime import datetime

from collections import defaultdict

DATA\_FILE = 'expenses.json'

def load\_data():

"""Load existing expense data from the JSON file."""

if not os.path.exists(DATA\_FILE):

return []

with open(DATA\_FILE, 'r') as file:

return json.load(file)

def save\_data(data):

"""Save expense data to the JSON file."""

with open(DATA\_FILE, 'w') as file:

json.dump(data, file, indent=4)

def get\_next\_id(data):

"""Get the next unique ID based on current data."""

if not data:

return 1

return max(exp['id'] for exp in data) + 1

def add\_expense(data):

"""Add a new expense to the tracker."""

try:

amount = float(input("Amount (₹): "))

category = input("Category (e.g., Food, Transport, Bills): ").strip().title()

note = input("Note (optional): ").strip()

date\_str = input("Date (YYYY-MM-DD) or leave blank for today: ").strip()

if date\_str:

try:

datetime.strptime(date\_str, "%Y-%m-%d")

date = date\_str

except ValueError:

print("❌ Invalid date format. Using today instead.")

date = datetime.now().strftime("%Y-%m-%d")

else:

date = datetime.now().strftime("%Y-%m-%d")

expense = {

"id": get\_next\_id(data),

"amount": amount,

"category": category,

"note": note,

"date": date

}

data.append(expense)

save\_data(data)

print("✅ Expense added successfully!")

except ValueError:

print("❌ Invalid input. Please enter a numeric amount.")

def view\_expenses(data):

"""Display all expenses in a tabular format."""

if not data:

print("No expenses found.")

return

print(f"\n{'ID':<4} {'Amount(₹)':<10} {'Category':<15} {'Date':<12} Note")

print("-" \* 60)

for exp in data:

print(f"{exp['id']:<4} {exp['amount']:<10.2f} {exp['category']:<15} {exp['date']:<12} {exp['note']}")

def delete\_expense(data):

"""Delete an expense entry by its ID."""

view\_expenses(data)

try:

exp\_id = int(input("\nEnter the ID of the expense to delete: "))

original\_length = len(data)

data = [exp for exp in data if exp['id'] != exp\_id]

if len(data) == original\_length:

print("❌ No expense found with that ID.")

else:

save\_data(data)

print("✅ Expense deleted.")

return data # Return updated data

except ValueError:

print("❌ Invalid ID format.")

return data

def show\_summary(data):

"""Display a summary of expenses by category and by month."""

if not data:

print("No data to summarize.")

return

category\_summary = defaultdict(float)

monthly\_summary = defaultdict(float)

total = 0

for exp in data:

category\_summary[exp['category']] += exp['amount']

month\_key = exp['date'][:7] # Format: YYYY-MM

monthly\_summary[month\_key] += exp['amount']

total += exp['amount']

print("\n📊 Expense Summary")

print(f"Total Spent: ₹{total:.2f}\n")

print("By Category:")

for category, amt in sorted(category\_summary.items()):

print(f" - {category}: ₹{amt:.2f}")

print("\nBy Month:")

for month, amt in sorted(monthly\_summary.items()):

year, mon = month.split('-')

print(f" - {datetime.strptime(mon, '%m').strftime('%B')} {year}: ₹{amt:.2f}")

def main():

"""Main menu loop for the CLI application."""

data = load\_data()

while True:

print("\n Personal Expense Tracker")

print("1. Add Expense")

print("2. View Expenses")

print("3. Delete Expense")

print("4. Show Summary")

print("5. Exit")

choice = input("Choose an option (1-5): ").strip()

if choice == '1':

add\_expense(data)

elif choice == '2':

view\_expenses(data)

elif choice == '3':

data = delete\_expense(data)

elif choice == '4':

show\_summary(data)

elif choice == '5':

print("👋 Exiting... Goodbye!")

break

else:

print("❌ Invalid choice. Please select a valid option.")

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

main()