

Personal Finance Tracker:

Project Overview and Objectives

The **Personal Finance Tracker** is a comprehensive, menu-driven Python application designed to help users efficiently manage their personal expenses. It allows users to record daily expenses, categorize spending, monitor monthly budgets, and generate detailed financial reports.

Objectives

- Provide an easy-to-use interface for tracking expenses
 - Store financial data persistently using file operations
 - Generate meaningful insights through reports and statistics
 - Apply modular programming principles
 - Demonstrate error handling and data validation
-

Features

- Add, edit, and delete expenses with full validation
 - Categorize expenses (Food, Transport, Entertainment, Bills, etc.)
 - Save data to JSON file with automatic backups
 - Load data on startup with error recovery
 - Generate monthly expense reports
 - View category-wise spending breakdown
 - Set and track monthly budgets
 - Export data to CSV for external analysis
 - User-friendly menu interface
-

Technologies Used

- **Programming Language:** Python 3
 - **Data Storage:** JSON, CSV
 - **Libraries:** json, csv, datetime, os, shutil
-

Setup and Installation Instructions

Prerequisites

- Python 3.8 or higher installed

- Command line / terminal access

Steps to Run the Application

```
cd week4-finance-tracker
```

```
python run.py
```

Project Directory Structure

```
week4-finance-tracker/  
|— finance_tracker/  
|   |— __init__.py  
|   |— expense_manager.py  
|   |— file_handler.py  
|   |— report_generator.py  
|   |— budget_manager.py  
|   |— statistics.py  
|   └— utils.py  
|  
|— data/  
|   |— expenses.json  
|   |— expenses_backup.json  
|  
|— run.py  
|— README.md
```

Code Structure Explanation

run.py

- Entry point of the application
- Displays the main menu
- Handles user interaction and navigation

expense_manager.py

- Add, edit, delete, and search expenses
- Input validation and category enforcement

file_handler.py

- Handles JSON read/write operations
- Automatic backup creation
- CSV export functionality
- Robust error handling for missing or corrupt files

report_generator.py

- Generates monthly expense summaries
- Displays totals and trends

budget_manager.py

- Set and update monthly budgets
- Compare expenses against budget limits

statistics.py

- Provides insights like highest spending category
- Calculates averages and totals

utils.py

- Common helper functions (date validation, formatting, menus)

• Technical Requirements – How They Were Met

• 1. File Operations (JSON & CSV)

- ✓ Expenses are saved to `expenses.json`
- ✓ Automatic backup stored as `expenses_backup.json`
- ✓ CSV export implemented for external analysis

• 2. Modular Code Structure

- ✓ Separate modules for expenses, reports, budgets, and files
- ✓ Clean separation of concerns
- ✓ Easy to maintain and extend

• 3. Error Handling

- ✓ Handles missing files on startup
- ✓ Recovers from corrupted JSON using backup
- ✓ Input validation for dates, amounts, and categories

• 4. Data Persistence

- ✓ All data is preserved between program runs
- ✓ Backup created before every save operation

• 5. Menu-Driven Interface

- ✓ Clear and user-friendly menu
- ✓ Input prompts with validation

• 6. Reporting and Analysis

- ✓ Monthly reports
- ✓ Category-wise breakdown
- ✓ Spending statistics
- **7. Budget Tracking**
- ✓ Monthly budget setting
- ✓ Alerts when budget is exceeded
- **8. Export Functionality**
- ✓ CSV export for Excel or Google Sheets
- **9. Code Readability**
- ✓ Meaningful function names
- ✓ Inline comments and documentation

Sample Main Menu

```
=====
MAIN MENU
=====

1. Add New Expense
2. View All Expenses
3. Search Expenses
4. Generate Monthly Report
5. View Category Breakdown
6. Set/Update Budget
7. Export Data to CSV
8. View Statistics
9. Backup/Restore Data
0. Exit
=====
```

Future Enhancements

- GUI version using Tkinter or PyQt
- Data visualization with charts
- Cloud storage integration
- User authentication

Personal Finance Tracker Code:

```
import json
def search_expenses(expenses):
    for e in results:
        print(f'{e["date"]} | {e["category"]} | ₹{e["amount"]} | {e["description"]}')

def monthly_report(expenses):
    print("\n--- MONTHLY REPORT ---")
    month = input("Enter month (YYYY-MM): ")
    total = 0
    for e in expenses:
        if e["date"].startswith(month):
            total += e["amount"]
    print(f"Total expenses for {month}: ₹{total}")

def category_breakdown(expenses):
    print("\n--- CATEGORY BREAKDOWN ---")
    category_totals = defaultdict(float)
    for e in expenses:
        category_totals[e["category"]] += e["amount"]
    for cat, total in category_totals.items():
        print(f"{cat}: ₹{total}")

def set_budget():
    print("\n--- SET / UPDATE BUDGET ---")
    category = input("Enter category: ")
    amount = float(input("Enter monthly budget amount: "))
    budget = load_budget()
    budget[category] = amount
    save_budget(budget)
    print("Budget updated successfully!")

def export_csv(expenses):
    print("\n--- EXPORT TO CSV ---")

def add_expense(expenses):
    description = input("Enter description: ")
    date = input("Enter date (YYYY-MM-DD) [leave blank for today]: ")
    if not date:
        date = datetime.today().strftime("%Y-%m-%d")
    expense = {
        "amount": amount,
        "category": category,
        "description": description,
        "date": date
    }
    expenses.append(expense)
    save_data(expenses)
    print("Expense added successfully!")

def view_expenses(expenses):
    print("\n--- ALL EXPENSES ---")
    if not expenses:
        print("No expenses found.")
        return
    for i, e in enumerate(expenses, 1):
        print(f'{i}. {e["date"]} | {e["category"]} | ₹{e["amount"]} | {e["description"]}')

def search_expenses(expenses):
    print("\n--- SEARCH EXPENSES ---")
    keyword = input("Enter category or description keyword: ").lower()
    results = [e for e in expenses if keyword in e["category"].lower()
               or keyword in e["description"].lower()]
    if not results:
        print("No matching expenses found.")
        return
    for e in results:
        print(f'{e["date"]} | {e["category"]} | ₹{e["amount"]} | {e["description"]}')
```

```
File Edit Selection View Go Run Terminal Help
importjson.py X
C:\Users> Santhosh Kumar > importjson.py > ...
70 def search_expenses(expenses):
71     for e in results:
72         print(f'{e["date"]} | {e["category"]} | ₹{e["amount"]} | {e["description"]}')
73
74
75 def monthly_report(expenses):
76     print("\n--- MONTHLY REPORT ---")
77     month = input("Enter month (YYYY-MM): ")
78
79     total = 0
80     for e in expenses:
81         if e["date"].startswith(month):
82             total += e["amount"]
83
84     print(f"Total expenses for {month}: ₹{total}")
85
86
87 def category_breakdown(expenses):
88     print("\n--- CATEGORY BREAKDOWN ---")
89     category_totals = defaultdict(float)
90
91     for e in expenses:
92         category_totals[e["category"]] += e["amount"]
93
94     for cat, total in category_totals.items():
95         print(f"{cat}: ₹{total}")
96
97
98 def set_budget():
99     print("\n--- SET / UPDATE BUDGET ---")
100     category = input("Enter category: ")
101     amount = float(input("Enter monthly budget amount: "))
102
103     budget = load_budget()
104     budget[category] = amount
105     save_budget(budget)
106
107     print("Budget updated successfully!")
108
109
110 def export_csv(expenses):
111     print("\n--- EXPORT TO CSV ---")
```

```
File Edit Selection View Go Run Terminal Help
importjson.py X
C:\Users> Santhosh Kumar > importjson.py > view_statistics
120 def export_csv(expenses):
121     writer = csv.DictWriter(
122         f, fieldnames=["date", "category", "amount", "description"]
123     )
124     writer.writeheader()
125     writer.writerows(expenses)
126
127     print("Data exported to expenses.csv")
128
129
130 def view_statistics(expenses):
131     print("\n--- STATISTICS ---")
132     if not expenses:
133         print("No data available.")
134         return
135
136     total_spent = sum(e["amount"] for e in expenses)
137     avg_spent = total_spent / len(expenses)
138
139     print(f"Total spent: ₹{total_spent}")
140     print(f"Average expense: ₹{avg_spent:.2f}")
141
142
143 def backup_restore(expenses):
144     print("\n--- BACKUP / RESTORE ---")
145     print("1. Backup Data")
146     print("2. Restore Data")
147
148     choice = input("Choose option: ")
149
150     if choice == "1":
151         with open(BACKUP_FILE, "w") as f:
152             json.dump(expenses, f, indent=4)
153             print("Backup completed successfully!")
154
155     elif choice == "2":
156         if os.path.exists(BACKUP_FILE):
157             with open(BACKUP_FILE, "r") as f:
158                 restored = json.load(f)
159                 save_data(restored)
160                 print("Data restored successfully!")
161         else:
```

```
File Edit Selection View Go Run Terminal Help
import json, py X
C:\Users> Santhosh Kumar > import json.py > view_statistics
164 print("No backup file found.")
165
166
167 ----- Main Menu -----
168 def main():
169     expenses = load_data()
170
171     while True:
172         print("\n" + "-" * 60)
173         print("      PERSONAL FINANCE TRACKER")
174         print("-" * 60)
175         print("\n=====")
176         print("      MAIN MENU")
177         print("=====")
178         print("1. Add New Expense")
179         print("2. View All Expenses")
180         print("3. Search Expenses")
181         print("4. Generate Monthly Report")
182         print("5. View Category Breakdown")
183         print("6. Set/Update Budget")
184         print("7. Export Data to CSV")
185         print("8. View Statistics")
186         print("9. Backup/Restore Data")
187         print("0. Exit")
188         print("=====")
189
190         choice = input("\nEnter your choice (0-9): ")
191
192         if choice == "1":
193             add_expense(expenses)
194         elif choice == "2":
195             view_expenses(expenses)
196         elif choice == "3":
197             search_expenses(expenses)
198         elif choice == "4":
199             monthly_report(expenses)
200         elif choice == "5":
201             category_breakdown(expenses)
202         elif choice == "6":
203             set_budget()
204         elif choice == "7":
205             export_csv(expenses)
```

```
File Edit Selection View Go Run Terminal Help
import json, py X
C:\Users> Santhosh Kumar > import json.py > view_statistics
168 def main():
169     print("1. Add New Expense")
170     print("2. View All Expenses")
171     print("3. Search Expenses")
172     print("4. Generate Monthly Report")
173     print("5. View Category Breakdown")
174     print("6. Set/Update Budget")
175     print("7. Export Data to CSV")
176     print("8. View Statistics")
177     print("9. Backup/Restore Data")
178     print("0. Exit")
179     print("=====")
180
181     choice = input("\nEnter your choice (0-9): ")
182
183     if choice == "1":
184         add_expense(expenses)
185     elif choice == "2":
186         view_expenses(expenses)
187     elif choice == "3":
188         search_expenses(expenses)
189     elif choice == "4":
190         monthly_report(expenses)
191     elif choice == "5":
192         category_breakdown(expenses)
193     elif choice == "6":
194         set_budget()
195     elif choice == "7":
196         export_csv(expenses)
197     elif choice == "8":
198         view_statistics(expenses)
199     elif choice == "9":
200         backup_restore(expenses)
201     elif choice == "0":
202         print("\n" + "-" * 60)
203         print("Thank you for using Personal Finance Tracker!")
204         print("-" * 60)
205         break
206     else:
207         print("Invalid choice. Please try again.")
208
209
210
211
212
213
214
215
216
217
218
```

Personal Finance Tracker output :

```
PS C:\Users\Santhosh Kumar> python finance_tracker.py
PS C:\Users\Santhosh Kumar> & 'c:\Users\Santhosh Kumar\AppData\Local\Programs\Python\Python314\python.exe' 'c:\Users\Santhosh Kumar\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '53711' '-' 'c:\Users\Santhosh Kumar\import_json.py'

=====
PERSONAL FINANCE TRACKER
=====

MAIN MENU
=====
1. Add New Expense
2. View All Expenses
3. Search Expenses
4. Generate Monthly Report
5. View Category Breakdown
6. Set/Update Budget
7. Export Data to CSV
8. View Statistics
9. Backup/Restore Data
0. Exit
=====

Enter your choice (0-9): 2

--- ALL EXPENSES ---
No expenses found.

=====
PERSONAL FINANCE TRACKER
=====

MAIN MENU
=====
1. Add New Expense
2. View All Expenses
3. Search Expenses
4. Generate Monthly Report
5. View Category Breakdown
6. Set/Update Budget
7. Export Data to CSV
8. View Statistics
9. Backup/Restore Data
0. Exit
```

```
PS C:\Users\Santhosh Kumar> & 'c:\Users\Santhosh Kumar\AppData\Local\Programs\Python\Python314\python.exe' 'c:\Users\Santhosh Kumar\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '53711' '-' 'c:\Users\Santhosh Kumar\import_json.py'
0. Exit

Enter your choice (0-9): 9

--- BACKUP / RESTORE ---
1. Backup Data
2. Restore Data
Choose option: 1
Backup completed successfully!

=====
PERSONAL FINANCE TRACKER
=====

MAIN MENU
=====
1. Add New Expense
2. View All Expenses
3. Search Expenses
4. Generate Monthly Report
5. View Category Breakdown
6. Set/Update Budget
7. Export Data to CSV
8. View Statistics
9. Backup/Restore Data
0. Exit
=====

Enter your choice (0-9): 5

--- CATEGORY BREAKDOWN ---

=====
PERSONAL FINANCE TRACKER
=====

MAIN MENU
=====
1. Add New Expense
2. View All Expenses
3. Search Expenses
```


