Personal Expense Tracker

Project Report

PERSONAL EXPENSE TRACKER

A Simple Command-Line Expense Manager

Submitted by:
Name: ANSHUMAN SINGH
Reg no.:25BAI10452

Course: CSE AI ML

---

1. Introduction

This project is a simple command-line application to track daily expenses. It helps users record their spending, view expenses by category, and understand where their money goes. The app stores data locally in a text file and works entirely offline.

Main Purpose:

- Track daily expenses easily
- See spending patterns
- Manage money better

---

2. Problem Statement

Many people don't know where their money goes each month. Existing expense apps are either too complicated or require internet and share your data online.

Problems:

- Hard to track daily expenses
- Existing apps are complex
- Privacy concerns with online apps
- Need smartphone for most apps

Solution: A simple, offline tool that runs on any computer with Python and keeps your data private on your own machine.

3. Functional Requirements

What the system can do:

1. **Add Expense** - Save expenses with date, category, amount, and description
2. **View All Expenses** - See all expenses in a table
3. **Search by Category** - Find expenses by type (Food, Transport, etc.)
4. **Calculate Total** - Show total money spent
5. **Category Summary** - Show spending breakdown by category with percentages
6. **Delete Last Expense** - Remove the most recent entry
7. **Auto Setup** - Creates data file automatically on first use

4. Non-functional Requirements

Usability:

- Simple menu with numbered options
- Easy to learn and use **Performance:**

- Fast response (under 1 second) **Reliability:**

- Data doesn't get lost
- Handles errors properly **Portability:**

- Works on Windows, Mac, and Linux
- Only needs Python 3

5. Design Decisions & Rationale

1. Text File Storage

- Why: Easy to use, no database needed, human-readable
- Benefit: Simple setup, works everywhere

2. Command-Line Interface

- Why: Works on all computers, fast and lightweight
- Benefit: Quick to use, no graphics needed

3. Fixed Categories

- Why: Prevents spelling mistakes, makes analysis easier
- Categories: Food, Transport, Bills, Shopping, Other

4. Single Python File

- Why: Easy to share and run
- Benefit: Beginners can understand easily

5. Delete Only Last Entry

- Why: Prevents accidental data loss
- Benefit: Simple and safe

---

6. Implementation Details

Technology Used:

- Language: Python 3
- Libraries: `os` (for file operations)
- Storage: Text file (expenses.txt) **Key Functions:**

1. `initialize_system()` - Creates file if needed
2. `add_expense()` - Adds new expense to file
3. `view_all_expenses()` - Shows all expenses
4. `search_by_category()` - Filters by category
5. `calculate_total()` - Adds up all expenses
6. `category_summary()` - Groups and calculates by category
7. `delete_last_expense()` - Removes last entry **Data Format:**

```
date|category|amount|description 22-11-
2024|Food|500.0|Lunch at restaurant
```

---

7. Testing Approach

Testing Method: Manual testing of all features **Test**

Cases:

1. **Add Expense** - Added 5 different expenses → ✓ Pass
2. **View Empty File** - Checked before adding data → ✓ Pass

3. **View All** - Saw all 5 expenses correctly → ✓ Pass
4. **Search Food** - Found only Food expenses → ✓ Pass
5. **Calculate Total** - Total matched manually → ✓ Pass
6. **Category Summary** - Percentages added to 100% → ✓ Pass
7. **Delete Last** - Last entry removed successfully → ✓ Pass
8. **Invalid Input** - Tested wrong category → ✓ Handled properly **Result:** All features

working as expected!

---

8. Challenges Faced

Challenge 1: File Formatting

- Problem: Keeping data aligned in tables
- Solution: Used Python string formatting with fixed widths

Challenge 2: Category Matching

- Problem: Users typing different cases (food vs Food)
- Solution: Made search case-insensitive with `.lower()`

Challenge 3: Empty File Handling

- Problem: Errors when file is empty Solution: Added checks before reading file

Challenge 4: Deleting Specific Entries

- Problem: Hard to delete middle entries safely
- Solution: Only allow deleting last entry for safety

---

9. Learnings & Key Takeaways

Technical Skills:

- File reading and writing in Python
- String manipulation and formatting
- Error handling with try-except
- Menu-driven program design

Soft Skills:

- Breaking big problems into small functions
- Writing clean, readable code
- Testing each feature thoroughly
- Documenting code properly **Key Takeaways:**

- Simple solutions often work best
- User experience matters even in command-line apps
- Testing is important for finding bugs
- Privacy can be achieved with local storage

---

10. Future Enhancements

Short-term improvements:

1. Edit existing expenses
2. Delete any expense (not just last one)
3. Search by date range
4. Monthly/yearly reports

Long-term improvements:

1. Add graphical interface (GUI)
2. Export to Excel/PDF
3. Set budget limits and alerts
4. Add charts and graphs
5. Multi-user support
6. Backup and restore feature
7. Recurring expense tracking
8. Mobile app version

---

11. References

Learning Resources:

- Python official tutorial
- Class room notes and lectures
- VITYARTHI ONLINE COURSE **Tools Used:**

- Python 3.x
- Text editor (VS Code)