📘 Expense Tracker Application – Project Report

A screenshot of a computer

AI-generated content may be incorrect.

# 1. Introduction

Managing personal expenses is an essential part of financial discipline. Many people struggle to track their daily spending, which can lead to poor financial planning.

This project, **Expense Tracker Application**, is designed to help users record and view their expenses in a simple and user-friendly way. The app allows the user to enter **amount, category, date, and description** of each expense, which is stored in a database and displayed in the interface.

# 2. Objectives

* To design a lightweight and simple desktop/mobile application for managing expenses.
* To provide an interactive **Graphical User Interface (GUI)** for data entry and display.
* To store expense records in a **database** for future reference.
* To demonstrate how **Python** can be used for both frontend and backend development.

# 3. Technologies Used

1. **Python**
   * Programming language used to build the application.
   * Handles logic, user input, and database communication.
2. **Kivy Framework**
   * Open-source Python framework for developing **cross-platform GUI applications**.
   * Provides widgets such as BoxLayout, TextInput, Button, Label, and RecycleView.
   * Used for designing the application interface.
3. **SQLite Database**
   * Lightweight, embedded database.
   * Used to store expense details (amount, category, date, description).
   * Integrated with Python using the sqlite3 module.

# 4. System Design

4.1 Architecture

* **Frontend (UI):** Kivy
* **Backend (Logic):** Python
* **Database (Storage):** SQLite

4.2 Workflow

1. User enters expense details in the app (amount, category, date, description).
2. On clicking **Add Expense**, the data is validated and stored in the SQLite database.
3. The **Expense List** is refreshed to show the newly added entry.
4. Users can view all stored expenses in the list format.

# 5. Features

* Add an expense with **Amount, Category, Date, and Description**.
* Automatically stores expenses in a local SQLite database.
* Displays all stored expenses in a scrollable list (RecycleView).
* Simple and user-friendly interface.
* Cross-platform (can run on Windows, Linux, macOS, and Android).

# 6. Advantages

* Lightweight and fast.
* Works offline (no internet required).
* Easy to use and extendable.
* Demonstrates integration of **GUI + Database** in Python.

# 7. Limitations

* Currently only supports adding and viewing expenses.
* No delete or update option.
* No advanced reporting (monthly/yearly summaries).

# 8. Future Enhancements

* Add **Delete/Update** functionality for expenses.
* Provide **search and filter** options.
* Generate **monthly reports** and visualizations using charts.
* Export expenses to **Excel or PDF**.
* Deploy on **mobile (Android/iOS)**.

# 9. Conclusion

The **Expense Tracker Application** is a practical project that demonstrates how Python can be used for building real-world applications. By integrating **Kivy (for UI)** and **SQLite (for database)**, the project provides both frontend and backend functionality within a single language.

This application can serve as a foundation for larger financial management systems and can be enhanced with advanced features in the future.

# 10. References

* Python Official Documentation – [https://docs.python.org](https://docs.python.org/)
* Kivy Documentation – <https://kivy.org/doc/stable/>
* SQLite Documentation – <https://www.sqlite.org/docs.html>