FINANCIAL MANAGEMENT SYSTEM REPORT

Developed by – Aarush Rahul Patel

Project Overview

The **Personal Finance Manager** is a Python-based web application built using **Streamlit** and **SQLite**.  
It allows users to record, manage, and analyze their financial transactions in a simple interface — all offline, with data securely stored in a local database (finance.db).

This app provides features such as:

* Adding new transactions (credited or debited)
* Viewing all transactions with totals
* Updating or deleting existing records
* Searching transactions by date or date range
* Visualizing **weekly and monthly spending trends**
* Analyzing **purpose/category-wise spending**

The app helps users understand their income and expenses better, promoting effective financial planning and spending habits.

**2. Technologies Used**

| **Component** | **Technology** |
| --- | --- |
| Frontend | Streamlit |
| Backend | Python |
| Database | SQLite |
| Libraries | Pandas, Matplotlib, Streamlit |

**3. Key Features**

1. **Add Transaction:**  
   Users can input transaction details — ID, type, purpose, amount, and date — which are stored in the database.
2. **View Transactions:**  
   Displays all transactions in a clean, interactive table. It also shows the total credited, total debited, and current balance.
3. **Update/Delete Transactions:**  
   Users can modify or remove specific transactions using their transaction number.
4. **Search by Date:**  
   Allows filtering of transactions by a specific date or range to find relevant financial activities.
5. **Trends Visualization:**  
   Weekly and monthly spending trends are visualized through line graphs, helping users monitor their expenses over time.
6. **Purpose-wise Analysis:**  
   Generates a bar graph showing total spending per category/purpose (e.g., food, travel, bills).

**4. Working Mechanism**

* Data is stored persistently in an **SQLite** database (finance.db).
* Streamlit provides the user interface for interacting with the data.
* **Pandas** handles data manipulation, while **Matplotlib** generates visual insights.
* The user selects actions from the sidebar to perform CRUD (Create, Read, Update, Delete) operations or view trends.

**5. Expected Output**

The app outputs:

* Interactive dashboard with transaction records.
* Auto-calculated totals for credits, debits, and balance.
* Dynamic plots showing weekly/monthly spending patterns.
* Purpose-wise expense distribution chart.

Algorithm

1. **Start**
2. Connect to (or create) the SQLite database finance.db.
3. Create table expenses if it doesn’t exist.
4. Display sidebar menu with options:
   * Add Transaction
   * View Transactions
   * Search by Date
   * Update Transaction
   * Delete Transaction
   * View Trends
   * Purpose-wise Spending
5. **If user selects "Add Transaction":**
   * Take input for Transaction Number, ID, Type, Purpose, Amount, and Date.
   * Insert record into database.
   * Confirm addition to user.
6. **If "View Transactions":**
   * Retrieve all records from database.
   * Display as a data table.
   * Calculate and display total credited, total debited, and balance.
7. **If "Search by Date":**
   * Accept start date and optional end date.
   * Fetch and display transactions in that range.
8. **If "Update Transaction":**
   * Select a transaction to modify.
   * Choose which field to update.
   * Update in database and show success message.
9. **If "Delete Transaction":**
   * Select transaction number.
   * Delete it from the database and confirm.
10. **If "View Trends":**
    * Convert Date to week/month.
    * Group and plot spending trends using Matplotlib.
11. **If "Purpose-wise Spending":**
    * Group by Purpose.
    * Plot a bar chart showing spending by category.
12. **End**

Flowchart

