Software Requirement Specification (SRS) Document

Financial Portfolio

1. Brief Description of Project

The Financial Portfolio Management System is a comprehensive console-based C project that helps individuals manage personal finances efficiently. It tracks assets, liabilities, income (active and passive), and expenses over time, factoring in inflation and deflation for precise growth analysis. The system estimates liability payoff timelines with interest calculations and provides tailored investment insights across stocks, real estate, and private equity sectors. It also aims on helping the user realise his Income Tax returns of the fiscal year.

2. Purpose / Goal

The system provides a clear, real-time view of financial health, empowering users to make informed decisions, boost savings, repay debt faster, and optimize investments and guide them in filing their income tax returns

3. Usefulness / Benefit

→ For Users:

Wealth Growth: By investing in assets that appreciate over time.

Income Generation: Through dividends, rent, interest, etc.

Risk Management: Diversifying investments to minimize loss.

Financial Planning: Helps with retirement, buying a house, paying off debt, etc

Taxation : Help the user save money in terms of tax relief, benefits and exemptions

→For Financial Advisors:

Helps provide personalized suggestions to clients.

Allows easy export and sharing of financial summaries.

4. Hardware / Software Involved

→Hardware Requirements:

- Standard PC/Laptop
- Minimum 2GB RAM
- 500MB disk space

→Software Requirements:

- Programming Language: C
- Compiler: GCC / Turbo C / Code::Blocks
- Operating System: Windows/Linux/MacOS
- Database: PostgreSQL / SQLite

5. Detailed Feature List

→User Module:

Register/Login

Add/Edit/Delete Assets and Liabilities

Enter Income (Active/Passive) and Expenses

Set Financial Goals

Track Financial Trends (adjusted for inflation/deflation)

Export Reports

→Financial Advisors Module:

Track Investment Portfolios (Stocks, Real Estate, Private Equity)

Give suggestions to user

6. Test / Demonstration Plan

→Unit Testing:

Test financial functions like ROI calculation, payoff period estimation, etc.

→Integration Testing:

Test interaction between modules

→System Testing:

End-to-end test of user journey, from input to report generation.

→User Acceptance Testing:

Verify performance and usefulness with target users.

7. Expected Interaction Interface and Sample Use Cases

→Interaction Interface:

Console based UI with Simple text-based input for user interaction

Financial summary screens displayed

→Sample Use Cases:

a) Track Net Worth:

User logs in → Adds current assets and liabilities → Views net worth chart over time.

b) Estimate Debt Payoff:

User enters a liability → Inputs interest rate and monthly payments → System shows estimated payoff timeline.

c) Analyze Investment Options:

User inputs surplus income → Sets risk preference → System suggests suitable investment categories.

d) Monitor Income/Expense Ratio:

User logs income and expenses → Views systemized data → Identifies savings trends.