Smart Financial Coach – Design Documentation

1. Problem Statement

Many individuals struggle to understand their spending habits. Manually tracking expenses is time-consuming and lacks personalized insight. This project solves that by providing an AI-driven dashboard that automatically processes transactions and highlights financial patterns.

2. Solution Overview

Smart Financial Coach is a web-based tool that:

- Lets users upload or input financial transactions.
- Categorizes and stores them in a database.
- Uses AI models to generate insights about spending and budgeting.
- Displays visual analytics and summaries through an interactive frontend.

3. System Architecture

- **Frontend:** React + Vite for fast UI and visualization.
- **Backend:** FastAPI for APIs and data handling.
- **Database:** PostgreSQL for persistent storage.
- **Containerization:** Docker + docker-compose to run the full system easily.

4. Technical Stack

Component	Technology	Purpose
Frontend	React, Vite	User interface and charts
Backend	FastAPI (Python)	API server and data processing
Database	PostgreSQL	Transaction data storage
AI / ML	OpenAI / custom Python model	Generating spending insights
Deployment	Docker, docker-compose	Containerization and orchestration

5. Design Choices

- **FastAPI** was chosen for speed, async support, and easy integration with Python ML code.
- **React + Vite** provides a lightweight, modern frontend build.
- **PostgreSQL** offers relational structure and reliability for financial data.
- **Docker** ensures consistent setup across environments.

6. Key Features

- Upload and store transactions (CSV or manual entry).
- AI-generated insights from transaction data.
- Interactive dashboard with spending visualizations.
- Persistent and isolated environment using Docker.

7. Challenges & Learnings

- Debugging backend API routes and ensuring smooth communication with the frontend.
- Managing environment variables between containers.
- Understanding Docker networks and dependencies.
- Building a consistent user interface for financial data visualization.

8. Future Enhancements

- Add authentication and user accounts.
- Integrate external APIs (Plaid, Stripe) for live transaction imports.
- Expand AI to forecast spending or savings trends.
- Deploy to a cloud service (AWS/GCP/Azure) with CI/CD pipeline.
