Design Architecture of the Chatbot

FinBuddy is a LLM-powered financial assistant built using **Streamlit, LangChain**, and **OpenAI GPT**. The architecture is designed for simplicity, modularity, and session-aware personalization, making it suitable for both prototypes and future production deployment.

Design Goals

- Deliver personalized financial advice using natural language
- Maintain minimal latency with efficient local execution
- Retain user-specific financial context during conversations
- Enable extensibility for additional financial tools and logic

---

Architecture Overview

The system follows a layered architecture:

1. Frontend Layer (Streamlit UI)

The **Streamlit** app serves as the user interface. It includes form-based inputs to capture financial details, a chatbot input field, and a sidebar for profile summaries. Session state is managed through **st.session_state**.

2. Logic Layer (Persona and Calculators)

The Persona class stores user attributes like age, income, savings, and investment goals. formulaengine.py encapsulates financial logic which is used both in frontend calculations and within the LLM tools.

3. Agent Layer (LangChain Agent)

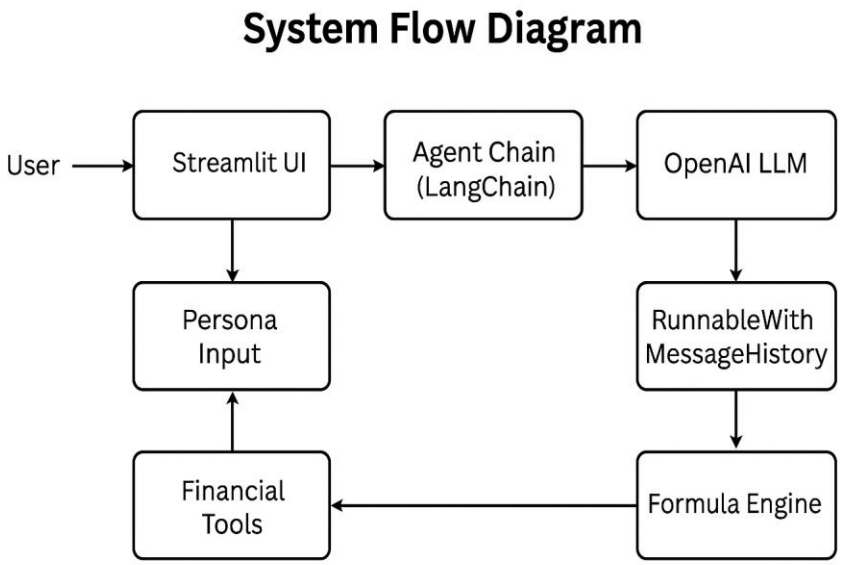
The **LangChain** agent is equipped with tools and uses OpenAI's GPT model to answer questions. The agent is wrapped with **RunnableWithMessageHistory** to maintain context over a session.

4. Memory Layer (Session Store)

The message history is stored using a simple dictionary indexed by **session_id.** This enables the app to track conversation flow and respond with continuity.

Design Trade-offs

- **Memory Design**: Using **RunnableWithMessageHistory** keeps the memory lightweight and session-scoped, but it lacks persistence unless integrated with an external backend.
- **Modularity**: Separating logic into distinct modules (Persona, Agent, Formulas) enhances readability, reuse, and future extensibility.
- **Simplicity vs. Scalability**: The current design is fast and easy to use locally, but horizontal scaling would require stateless APIs, user authentication, and a persistent chat history store



Tools Used

FinBuddy integrates several custom tools within the LangChain framework to enhance its ability to reason over financial data and provide personalized responses. These tools are lightweight, deterministic functions that the LLM can call when specific user queries require precise calculations or explanations.

1. Financial Calculation Tools

These tools are built in formulaengine.py and exposed to the agent for use during conversation:

- **Future Value Calculator – Computes the future worth of current savings or investments.**
- **Annuity Value Tools – Used to calculate recurring monthly savings effects (both present and future).**
- **Required Monthly Savings – Determines how much the user needs to save monthly to hit a target goal.**
- **Rule of 72 Tool – Estimates how long it will take to double an investment.**
- **NPER Tool – Calculates how many months or years are needed to reach a financial target.**

2. Formula Explainer Tool

This tool allows the agent to explain the logic or math behind its recommendations when users ask questions like *"How was this calculated?"*. It maps each financial function to a plain-English explanation and includes the formula used.

3. Side Hustle Search Tool (Tavily)

FinBuddy integrates with the Tavily Search API to suggest side hustles tailored to the user's financial gap. For example, if the user is ₹8,000 short in monthly savings, the tool dynamically queries for relevant income opportunities to bridge the gap.