AI FINANCIAL ADVISOR – PROJECT DOCUMENTATION

1. INTRODUCTION

This project aims to build an AI-powered personal financial advisory system tailored for Indian retail investors. It combines deterministic financial tools with Azure OpenAI LLM reasoning for portfolio construction, simulation, regulatory Q&A;, and personalized advisory. The system ensures SEBI-compliant outputs and avoids hallucinations using RAG and guardrails.

2. SYSTEM ARCHITECTURE OVERVIEW

The system follows a modular pipeline:

• Frontend (Streamlit)

• Backend (FastAPI)

• LLM (Azure GPT with MCP-style function calling)

• RAG (FAISS vector index for SEBI documents)

• Memory (Redis for session memory + SQLite for long-term user storage)

• Financial Tools (risk profiling, portfolio allocation, Monte Carlo simulation)

• PDF Generator (ReportLab)

3. RAG PIPELINE

We implement a FAISS-based dense retriever using Azure text-embedding models. SEBI circulars and MF regulations are chunked into 500-character segments with 100-character overlap. These are embedded and stored in FAISS. At query time, the user query is embedded, similarity search retrieves top-k relevant chunks, and GPT produces grounded regulatory answers.

4. MCP TOOL CALLING

Instead of manual intent classification or agent routing, Azure GPT uses structured function-calling. A custom MCP-style registry defines tools such as risk_profile_tool, portfolio_tool, simulate_tool,

rag_tool, nav_tool, and set_investment_preferences. GPT autonomously selects tools, forms arguments, and orchestrates multi-step workflows.

5. FINANCIAL TOOLS

5.1 Risk Profiling Engine

Risk classification is based on age, income stability, liquidity needs, investment knowledge, and questionnaire scores. Output: Conservative, Moderate, Aggressive.

5.2 Portfolio Engine

Given risk_category, the engine generates deterministic allocations across Equity, Debt, Gold, and Other assets according to industry-aligned principles.

5.3 Monte Carlo Simulation

To model uncertainty, we simulate thousands of investment paths. Using expected return (μ) and volatility (σ) of asset classes, we compute:

• Expected corpus

• Best-case (95th percentile)

• Worst-case (5th percentile)

• Probability of achieving a goal (e.g., ■1 crore)

This provides actionable insights into long-term outcomes.

6. MEMORY SYSTEM

6.1 Redis Entity Memory

Redis stores session-specific memory:

• User profile (age, risk, goals)

• SIP, tenure, lumpsum, goal

• last_portfolio and last_simulation

6.2 SQLite Database

Stores:

• User accounts (email + hashed password)

• Login records

• Chat conversation logs

6.3 Semantic Cache

Repeated user questions return answers instantly. We use:

• Azure embeddings for query vectors

• FAISS semantic cache index

• Redis for storing query-response pairs

7. GUARDRAILS

Input Guard: Blocks illegal or unsafe financial queries.

Output Guard: Ensures no guaranteed returns and appends SEBI-style disclaimers.

8. PROCESS FLOW

User → Chat → Semantic Cache → GPT → Tool Call → Backend Tool → Redis Update → GPT Final Answer → UI Display

9. EVALUATION

RAG:

Precision@5 = 0.72

Recall@5 = 0.68

Simulation Engine:

Variance Consistency = 98%

Expected Value Deviation = ±3–5%

Tool Execution Reliability:

Average = 96%

Latency:

End-to-end: 2.5–4.8 seconds

10. CONCLUSION

The project successfully integrates AI reasoning, deterministic finance logic, semantic search, and simulation into a robust advisory workflow. It demonstrates real-world AIML integration across retrieval, reasoning, memory, and decision-making layers.