Project SuperMaya: MVP Handover Document

To: Pratham Mangla
From: Yash Gunjal (yash830gunjal@gmail.com)
Date: August 6, 2025
Subject: Delivery of the AI Backend Engineer MVP Prototype: Project SuperMaya

---

1. Executive Summary

Project SuperMaya is a powerful, multi-modal, and agentic AI platform designed to power next-generation digital experiences. This MVP successfully delivers the core backend architecture and a functional frontend workspace, demonstrating a robust foundation for conversational AI, real-time data analysis, and dynamic visualization.

The platform's key innovation lies in its **agentic architecture**, where a central "MetaAgent" intelligently classifies user intent and routes tasks to specialized sub-agents. This allows the system to handle diverse queries, from general knowledge and image analysis to real-time financial data fetching.

Crucially, the system goes beyond simple text responses by dynamically generating interactive data visualizations (charts, graphs) on-the-fly using a universal rendering engine, showcasing a truly intelligent and scalable approach to data-driven conversation. This MVP successfully meets and, in key areas of intelligence and scalability, exceeds the core requirements of the initial project scope.

2. Live Demonstration & Guide

A comprehensive walkthrough of the project, including its features, architecture, and the challenges overcome, is available in the following Loom video:

- **Loom Video Link:** Video Link

To experience the platform directly, please follow these steps:

1. **Run the Backend:** Navigate to the SuperMaya directory and run uvicorn app.main:app --reload.
2. **Run the Frontend:** Navigate to the supermaya-ui directory and run npm run dev.
3. **Access the UI:** Open http://localhost:5173 in your browser.

Testing the Core Features:

- **User Authentication:** Use the UI to **Register** a new account, then **Login**. All subsequent interactions are secure and tied to your user profile.
- **General Knowledge & Rich Content:** Ask a question like:

tell me about neural network and its diagram
The system will provide a text answer, a link to a valid diagram image, and reference URLs.

- **Real-Time Financial Data & Charting:** Ask a financial query like:

give me a chart for today's sensex
The system will route the query to the FinancialAgent, fetch live data from Yahoo Finance, and render a responsive line chart in the UI.

- **Generative Data Visualization:** Ask a general knowledge question that requires a different type of chart:

i want a pie chart of the wealth distribution of india
The TextAgent will generate the statistical data and a Vega-Lite specification for a pie chart, which the UI will render dynamically.

- **Multi-Modal Vision Analysis:** Click the paperclip icon (📎), upload an image of a menu, and ask:

ocr this menu and list the items and prices
The VisionAgent will analyze the image, perform OCR, and return a structured list of the menu items.

3. Technical Architecture

The project is a full-stack application designed for scalability and intelligence.

Backend (Python / FastAPI)

- **Framework:** Built on FastAPI for high-performance, asynchronous API services.
- **Database:** Uses SQLAlchemy with SQLite for robust data persistence, including user accounts, custom prompts, and full chat history.
- **Core Logic (The "Brain"):** The system is built around a MetaAgent that performs intent classification.
    - **General Agent (llama3-70b-8192 via Groq):** Handles text queries, finds images/links, and dynamically generates Vega-Lite data visualization specs.
    - **Financial Agent (yfinance library):** A specialized tool that extracts stock market symbols, fetches real-time data, and formats it for visualization.
    - **Vision Agent (gemini-1.5-flash):** Handles multi-modal inputs, capable of general image analysis and specific tasks like OCR.
- **Security:** Implemented JWT-based authentication for all user-facing endpoints.
- **Data Integrity:** Leverages Pydantic and the instructor library to ensure all AI model outputs are returned as clean, validated, and structured JSON.

Frontend (React / Vite)

- **Framework:** A modern, fast UI built with React and Vite.
- **Communication:** Uses axios with a robust interceptor to manage JWT authentication seamlessly.
- **Universal Visualization Engine:** The UI features a single, powerful VegaChart component. Instead of being hardcoded to render a few chart types, it can render **any** valid Vega-Lite specification provided by the backend. This makes the system infinitely extensible without frontend code changes.

- **Component-Based & Interactive:** Responses are rendered as rich components, including clickable tags for follow-up queries and feedback buttons.

4. Key Challenges & Solutions

The development process involved overcoming several real-world AI engineering challenges:

- **Challenge:** AI models often return unstructured or unreliable text.
    - **Solution:** Implemented Pydantic models and the instructor library to enforce strict, validated JSON schemas on all AI outputs, ensuring data integrity across the application.
- **Challenge:** Different queries require vastly different tools and data sources.
    - **Solution:** Designed a MetaAgent with an intent-classification router. This allows the system to delegate tasks to the appropriate specialized agent (e.g., routing a "Sensex" query to the FinancialAgent while a "pie chart" query goes to the more creative TextAgent).
- **Challenge:** External financial data APIs have restrictive free tiers and can be unreliable.
    - **Solution:** After hitting API limits, the system was re-architected to use the industry-standard yfinance library, providing a more robust and stable source of real-time market data.
- **Challenge:** Hardcoding the UI for every possible chart type is unscalable and limits the AI's potential.
    - **Solution:** Implemented a universal visualization engine using **Vega-Lite**. The backend AI now generates a complete visualization specification, decoupling it from the frontend and allowing it to dynamically create any chart type it can conceive of, from line and pie charts to complex scatter plots, without requiring any changes to the UI code.

5. Compliance with Project Scope

This MVP delivers a powerful foundation that covers the most critical aspects of the problem statement.

- ✅ **Design scalable, modular API services:** Achieved with FastAPI and agentic architecture.
- ✅ **Implement multi-model AI workflows:** Achieved with Groq (Llama3) and Google (Gemini).
- ✅ **Process user uploads (text and images):** Fully implemented.
- ✅ **Manage chat histories and contextual logs:** Fully implemented via the database.
- ✅ **Handle image and file uploads:** Fully implemented.
- ✅ **Robust logging and error handling:** Implemented in all agentic and API layers.
- ✅ **Computer vision and text-image processing:** Achieved with the VisionAgent.

- ✅ **Advanced Prompt Engineering:** Demonstrated in the strict, schema-enforcing prompts for all agents.
- ◑ **Integrate with external APIs:** Foundationally achieved with the FinancialAgent. The architecture is now ready to easily add new agents for Google Calendar, weather, etc.
- ◑ **Context tracking and user session management:** Full context is stored in the database. JWT provides secure session management.
- ❌ **Admin Dashboards & Customization UI:** Deferred as a post-MVP feature. The backend, however, already supports storing user-specific system prompts.

6. Future Roadmap

The current architecture is primed for rapid expansion. The next logical steps include:

- **Expanding the Agent Toolbox:** Integrate new agents for Google Calendar (proactive meeting briefs), weather, and email.
- **Building a User Dashboard:** Create a settings page where users can update their custom AI prompt (system_prompt) and manage their integrations.
- **Implementing Feedback Loops:** Develop an internal tool that analyzes user feedback scores (feedback_score in the database) to identify areas where the AI's responses can be improved.
- **Vector Database for Long-Term Memory:** Integrate a vector store like ChromaDB to give the AI a long-term, semantic memory of past conversations, enabling deeper personalization.

Thank you for the opportunity to build this prototype. I am confident that Project SuperMaya provides a powerful and scalable foundation for the next generation of AI-driven applications.