Problem Statement

You have to build an **Agentic AI System** that transforms a traditional **search-and-click pharmacy** into an **autonomous ecosystem.** The system should behave like an **expert pharmacist**, understand natural voice/text conversations, predict medicine refill needs, enforce safety and prescription rules, and autonomously execute backend tasks such as inventory updates and procurement—with minimal human intervention.

Core Functional Requirements

Your system will be judged on its ability to handle these **Agentic Capabilities**:

- **Conversational Ordering:** A natural interface (text-based and voice-based) that can extract medicine names, dosages, and quantities from messy, human like dialogue.
- **Safety & Policy Enforcement:** The system must use the provided **Medicine Excel Sheet** as its "Source of Truth." It must autonomously decide if an order can proceed based on stock levels and "Prescription Required" flags.
- **Predictive Intelligence:** The system must be "Proactive." Based on the **Mock Customer History**, it should identify which users are running out of medicine and initiate a refill conversation or alert
- **Real-world Action (Tool Use):** The system shouldn't just "talk." It must execute actions such as updating your mock database, triggering webhooks (mock webhooks or n8n/Zapier), or sending order confirmations via channels like email, WhatsApp

Data & Environment

A. Data Assets

1. **Medicine Master Data (Excel/CSV):** Contains a list of medicines, current stock levels, unit types, and "Prescription Required" flags.
2. **Consumer Order History (Excel/CSV):** Contains historical data for a set of patients, including what they bought, the date of purchase, and the dosage frequency.

B. Mock Backend

- **No CMS provided:** You must build a mock API/Backend (FastAPI, Node.js, Supabase, etc.) to host the provided data.
- Your agents must interact with this backend to read inventory, check user history, and write new orders.

C. Observability (Mandatory)

Since this is an agentic system, **traceability is non-negotiable.**

- You must integrate an observability platform (e.g., **Langfuse**, LangSmith).
- The judges must be able to see the **Chain of Thought (CoT)**: How Agent A talked to Agent B, and why Agent B approved/rejected a request.
- *Submissions without a live Trace Log link will be disqualified.*

D. Minimal UI

- A simple UI is enough but can be enhanced to show the features
- It must include a **Chat-based interface** and Voice capability is a high-value priority
- A small "Admin View" showing the Mock Inventory levels and any "Proactive Refill" alerts the system has generated.

E. Workflow Automation

- Show a real-world When an order is finalized, your agent must trigger a warehouse fulfillment request via a mock **Webhook or API**. (Optional Bonus: Automate confirmation emails or WhatsApp messages).

E. Submission Checklist

- **GitHub Repo:** Clean code and a README for your Mock APIs.
- **Observability Public Link:** Access to view your agentic trace logs.
