**Target Architecture (Python + Streamlit + OpenAI + API)**

* **Frontend:** Streamlit app
  + Pages: Dashboard, Query, Suggestions, Admin.
  + Components: text input (NL query), multi-select filters, results grid, charts, export.
* **Backend API:** FastAPI
  + Endpoints (minimal set):
    - POST /query – run NL→SQL and return results.
    - POST /restock-suggestions – compute reorder quantities + notes.
    - GET /schema – expose safe schema (for prompting).
    - GET /filters – plants, SKUs, vendors for UI.
  + Middleware: Auth (JWT/OIDC), rate limiting, tenancy, request tracing.
* **Data:** Read-only role to warehouse/DB (e.g., Postgres, Snowflake, BigQuery).
  + Core tables: inventory\_snapshots, skus, plants, vendors, lead\_times, moq\_rules, open\_pos, inbound\_asn, sales\_forecast.
* **Intelligence (OpenAI):**
  + Model 1 (NL→SQL): system prompt grounded with **schema JSON** + **policy guardrails**.
  + Model 2 (Explainers): convert numeric outputs into concise, auditable rationale (“Order 120 units due to 10-day lead time and 12/day demand.”)
* **Security/Compliance:** RLS by tenant/plant; SQL sanitizer (block DDL/DML; only SELECT), column masking for cost/PII, query timeouts.
* **Observability:** Structured logs, prompt/response traces (w/ redaction), query catalog (approved SQL), drift and failure metrics.