Supply Chain PO Automation

One-page summary derived from repository code and setup docs.

What it is

Supply Chain PO Automation is a multi-agent purchase order system with a Next.js frontend and a FastAPI backend.
It analyzes inventory and forecasts, selects suppliers, plans container usage, creates draft POs, and records an audit trail for human review.

Who it's for

Primary user: supply chain planners and approvers managing replenishment and purchase-order review.

How to run

1. Create root .env and frontend/.env.local with Supabase, OpenRouter, and backend URL values.
2. Apply supabase/schema.sql and seed data with data/seed_data.py.
3. Start the backend from backend/ with uvicorn main:app --reload.
4. Start the frontend from frontend/ with npm install, then npm run dev, then open http://localhost:3000.

What it does

- Shows system health and live operational counts from Supabase.
- Runs a 4-step PO pipeline for selected or auto-detected SKUs.
- Calculates net demand from inventory, forecasts, safety stock, and MOQ rules.
- Scores suppliers, chooses recommended vendors, and estimates container plans.
- Creates draft purchase orders with line items, totals, and AI-written notes.
- Supports human approval/rejection and a full decision log with rationale and confidence.

How it works

- Next.js frontend calls the FastAPI backend via NEXT_PUBLIC_BACKEND_URL; the dashboard also reads directly from Supabase.
- FastAPI backend exposes health, pipeline run, approval, PO listing, and log endpoints.
- LangGraph runs DemandAnalyst -> SupplierSelector -> ContainerOptimizer -> POCompiler, with early exits on errors or empty results.
- A Python MCP client launches Node-based MCP servers over stdio JSON-RPC for ERP, supplier, logistics, and PO operations.
- Supabase stores operational data, purchase orders, line items, and decision logs; OpenRouter powers LLM rationales and summaries.