**Supply Chain Inventory Management Agent**

**Problem Statement:**

Retailers and manufacturers must maintain optimal inventory levels to avoid stockouts or overstocking. Manual inventory monitoring and reordering leads to inefficiencies, delays, and capital waste—especially under unpredictable demand conditions.

**Goal:**

Build an AI-powered inventory management agent that:

* Monitors inventory and demand forecasts
* Simulates "what-if" scenarios and reordering decisions
* Automates restocking or alerts based on dynamic thresholds

**Value Proposition:**

* Reduces stockouts and over-ordering
* Improves supply chain responsiveness
* Automates reorder workflows
* Enhances operational efficiency without complex ERP setup

**Scope:**

* Simulated inventory and product catalog
* LLM agent that monitors stock levels and forecasts shortages
* Mock API calls to suppliers or alert systems
* Basic “reasoning” via prompt or LangChain workflows

**Future Important Options:**

* Real-time ERP integration
* Full warehouse tracking or shipping systems
* Multi-site inventory management

**Key Lean Metrics:**

* Inventory response time (time to reorder after threshold breach)
* Percentage stockouts avoided
* Inventory turnover rate improvement
* Number of false positive reorder alerts

**Sprint Plan with User Stories:**

**Sprint Duration:** 1 week **Team:** AI Engineer, Frontend/UI Engineer, Test Engineer, Backend/Data Engineer

**Sprint 1: Dataset & Inventory Logic Setup**

**Goal:** Create a small inventory simulation and define core logic

**User Stories**:

* Analyze data to simulate a small product catalog with price, quantity, and reorder threshold so that we can model basic inventory workflows.
* Define reorder point logic in Python using inventory formulas so we can calculate when items need restocking.
* The solution should predict outline scenarios (e.g., demand spike, supplier delay) to test agent decisions later.

**Deliverables:**

* Inventory dataset
* Python function for reorder point calculation
* Defined test cases for “what-if” inventory events

**Sprint 2: LLM Integration for Reorder Reasoning**

**Goal:** Use an LLM to identify stock shortages and suggest actions

**User Stories**:

* An AI Engineer should create prompts for checking inventory status and generating reorder suggestions in natural language.
* Backend/Data Engineer should wrap inventory logic inside LangChain so the agent can execute workflows like “check stock → order items”.
* Test Engineer should validate whether the model triggers reorders correctly under threshold conditions.

**Deliverables:**

* Prompt-based reasoning templates (e.g., “Item X will run out in Y days…”)
* LangChain pipeline for simulating supplier order placement
* Basic terminal-based output of reorder actions

**Sprint 3: Interface & Automation Layer**

**Goal:** Build a dashboard to visualize and control inventory agent behavior

**User Stories**:

* A user of the product would want to see real-time stock levels and reorder alerts on a dashboard so that the agentic actions can be understood and verified.
* Simulate supplier API responses so the agent can “place orders” or receive delivery timelines.
* Build simple visualizations (e.g., low stock charts) for intuitive insight.

**Deliverables:**

* Web dashboard with inventory tables and reorder indicators
* Simulated supplier/order API
* Notification/alert panel for critical stock levels

**Sprint 4: Scenario Testing & Optimization**

**Goal:** Run test cases and tune decision quality of the agent

**User Stories**:

* A test engineer should simulate 5–6 demand scenarios (e.g., high sale day, supply delay) to test if the agent adapts correctly.
* Assess reorder timing accuracy and false positives so we can fine-tune thresholds.
* The team leaders should collect feedback from internal users and adjust logic/presentation before MVP(Minimum Viable Product) rollout.

**Deliverables:**

* Scenario test logs and results
* Optimized reorder thresholds
* MVP walkthrough/demo package

**Continuous Improvement Plan:**

* Integrate with ERP APIs for live data feeds
* Use reinforcement learning for reorder policy optimization
* Incorporate seasonal demand modeling
* Add warehouse-to-shelf coordination logic in future phases