End-to-End Supply Chain Analytics API using FastAPI, Supabase, and Python

1. Introduction

Modern supply chains generate massive volumes of transactional and operational data, such as supplier shipments, customer sales orders, and warehouse inventory snapshots. Turning this raw data into actionable insights is essential for improving efficiency, forecasting demand, and mitigating risks such as stockouts or delayed shipments.

However, traditional analytics approaches often require manual data processing, scattered spreadsheets, or heavy enterprise systems that are difficult to scale. To address these challenges, this project develops an automated, API-driven supply chain analytics microservice that streamlines the entire workflow.

The platform is built using FastAPI for lightweight yet high-performance REST APIs and integrates with Supabase (PostgreSQL backend) for persistent data storage. For prototyping or quick testing, the system can also run entirely in-memory, making it flexible for both enterprise-scale deployments and academic or proof-of-concept use cases.

Key Capabilities:

- Data Ingestion
  Seamlessly ingest structured data from suppliers, sales orders, and inventory systems via JSON payloads or CSV uploads.
- KPI Computation
  Automatically calculate key performance indicators (KPIs) such as daily demand trends, rolling 7-day/28-day demand, average supplier lead times, and stockout risks.
- Demand Forecasting
  Leverages time-series forecasting methods (Seasonal Moving Average) to generate SKU-level demand predictions over a user-defined horizon.
- API-First Design
  Exposes analytics results via REST endpoints for direct integration into dashboards, business intelligence tools, or downstream decision support systems.

With this design, the system acts as a scalable backend service for supply chain analytics, enabling real-time monitoring, forecasting, and optimization of complex supply chain operations.

2. Objectives

- Automate ingestion of supply chain data in JSON/CSV format.
- Compute rolling demand KPIs (7-day and 28-day windows).
- Estimate supplier lead times and detect potential stockouts.
- Provide short-term demand forecasting using Seasonal Moving Average (SMA).

- Make analytics accessible via REST API endpoints.

3. Methodology

This project follows a modular API-driven analytics workflow, combining data ingestion, KPI computation, and forecasting into a single, lightweight service.

3.1 Architecture

The system is designed around FastAPI for performance and modularity, with optional database integration for scalability.

- **FastAPI** → High-performance API framework for exposing ingestion, analytics, and forecast endpoints.
- **Supabase (PostgreSQL)** → Cloud-hosted database for persistent storage and production deployment (optional).
- **Pandas & NumPy** → Core libraries for data wrangling, KPI computation, and time-series forecasting.
- **dotenv** → Secure handling of environment variables (e.g., API keys, database credentials).

This architecture ensures flexibility: the service can run in in-memory mode for testing or connect to Supabase for persistent, real-world deployments.

3.2 Data Model (Logical Tables)

Data is structured into logical tables to mirror key entities in a supply chain:

- **suppliers** → Supplier details (ID, metadata).
- **skus** → Stock-keeping units (unique product identifiers).
- **supplier_shipments** → Shipment records (supplier, SKU, quantity, lead time, cost).
- **sales_orders** → Customer orders (SKU, date, quantity, price).
- **inventory_snapshots** → Time-based inventory levels (on-hand, in-transit).
- **kpi_daily** → Pre-computed KPIs for demand, lead time, and stockout risk.
- **forecasts** → Forecast results (SKU, horizon, predicted demand).

This schema supports both operational data capture and analytical outputs in a clean, normalized structure.

3.3 Analytics Workflow

1. **Data Ingestion (/ingest)**
    - Accepts data via JSON or CSV uploads.
    - Stores in Supabase (if configured) or in-memory tables for rapid testing.

2. **KPI Computation (/compute)**
    - Daily demand per SKU.
    - Rolling demand windows (7-day and 28-day).
    - Average supplier lead time (from shipment records).
    - Stockout risk flags (on-hand ≤ 0).
3. **Forecasting (/retrain)**
    - Seasonal Moving Average (SMA) model applied per SKU.
    - Generates 14-day ahead forecasts for demand planning.
4. **Data Access (/kpis, /forecast)**
    - Exposes KPIs and forecasts through REST endpoints.
    - Can be consumed by dashboards, BI tools, or decision support systems.

4. Results

Example KPIs

SKU	KPI Date	Demand 7D	Demand 28D	Avg Lead Time	Stockout
SKU-1	2025-07-09	260	260	9.5 days	False
SKU-2	2025-07-11	155	155	12 days	False

Example Forecast

SKU	Forecast Date	Horizon (Days)	Forecast Qty	Method
SKU-1	2025-07-12	1	130.0	SMA_7
SKU-1	2025-07-13	2	130.0	SMA_7
SKU-1	2025-07-14	3	130.0	SMA_7