بازار

Case Study Challenge | Engineering

Business Context and Case Overview

The Engineering Team at Bazaar Technologies plays a critical role in building scalable and reliable solutions that power our products. Our engineers work closely with customers, deploy directly to production, and contribute across multiple tech stacks.

We're now tackling the challenge of designing and scaling an **Inventory Tracking System**—starting from a single kiryana store and evolving into a multi-store, multi-supplier platform with built-in audit capabilities. This system must be designed for high performance, scalability, and reliability to support high-volume retail operations.

At Bazaar, we operate in a dynamic retail-tech landscape where robust systems are key to business success. Efficient inventory management is crucial for maintaining store profitability, streamlining supply chain operations, and delivering seamless customer experiences. As part of our Engineering function, you'll be expected to build solutions that integrate with the broader Bazaar technology ecosystem, while handling high transaction volumes securely and efficiently.

Key Challenges:

- Designing for real-time stock visibility to prevent stockouts and overstocking.
- Architecting a scalable system that accommodates multiple stores, suppliers, and inventory variations.
- Ensuring data integrity, security, and performance as the system scales to thousands of stores.

Candidates should consider system evolution from a single-store model to a large-scale distributed architecture. Key aspects include database design, API efficiency, asynchronous processing, caching strategies, and security measures to support high concurrency and transactional integrity.

Your Role in This Challenge

As part of the Engineering team, you have been asked to develop a backend service to track product inventory and stock movements for a single kiryana store, with future scalability to support multiple stores, suppliers, and audits.

<u>Key Factors to Consider:</u>

- Efficient data modeling to track product stock-in, sales, and manual removals.
- API design and data storage strategy (local storage transitioning to a relational database).
- Security, access control, and scalability considerations at different stages.

<u>Key Data Points & Assumptions</u>

Initially, you should focus on building an inventory tracking module for a single store. A product can be stocked in, sold, or removed manually.
Key factors to consider:

- Model product, stock movements, current quantity
- CLI or simple API is acceptable
- Store data locally (flat file or SQLite)

بازار

During Stage 2 of the expansion process, your service should support 500+ stores with a central product catalog and store-specific stock.

Key factors to consider:

- Add REST API endpoints for core actions
- Enable filtering/reporting by store, date range
- Introduce basic auth & request throttling
- Use a relational DB (e.g., PostgreSQL)

In the last stage (stage 3) of the expansion process, your service should support thousands of stores, concurrent operations, near real-time stock sync, and audit logs.

Key factors to consider:

- Make your system horizontally scalable
- Add support for asynchronous updates (event-driven)
- Consider read/write separation, caching, and API rate-limits
- Capture design decisions and trade-offs

Submission Guidelines:

Your solution should include:

1. Code (even if partial/stubbed)
2. README including:

- Design decisions
- Assumptions
- API design
- Evolution rationale (v1 → v3)

This case mirrors the real challenges our Engineering teams solve every day. We're excited to see how you think critically, leverage data, and bring creativity to drive sustainable growth at Bazaar.

Best of luck! 🚀