**STORE MANAGER: KEEP TRACK OF INVENTORY**

**TEAM ID:** NM2025TMID40223

**TEAM DETAILS**

**Team ID:** NM2025TMID40223

**Project Title:** Store Manager: Keep Track Of Inventory

**Team Leader:** Sriram D

Mail I'd : [sriramaasif@gmail.com](mailto:sriramaasif@gmail.com)

NM ID: B8ED8F79F450F60AE1347C826B6733E5

**Team Member:** Sanjay B

Mail I'd: [baskarsanjay4@gmail.com](mailto:baskarsanjay4@gmail.com)

NM ID: 1103F13BD54D73486734078BA9532B97

**Team Member:** Dinesh Kumar M

Mail I'd:[emrsdineshkumar@gmail.com](mailto:emrsdineshkumar@gmail.com)

NM ID: 2AAAAE5069FDF5D98DC4BB54B5801243

**Team Member:** Guna sekaran D

Mail I'd:[gunasekarandurai1234@gmail.com](mailto:gunasekarandurai1234@gmail.com)

NM ID: BC8D48A55A49A5EA9E44C584DE944BA9

**Team Member:** Durga Devi V

Mail I'd: [durgadevi091206@gmail.com](mailto:durgadevi091206@gmail.com)

NM ID: B3D3676288CC532B6C04BE704698ED7D

---

**Objective**

To develop a system that enables the store manager to efficiently track, monitor, and manage inventory by maintaining accurate stock records, updating stock levels in real time upon sales or purchases, preventing stockouts or overstocking, and ensuring smooth day-to-day operations of the store.

---

**Platform**

Web-based Application (accessible from browser across devices)

Mobile Application (Android/iOS for on-the-go management)

Desktop Application (optional, for offline store operations)

---

**Technologies Used**

Frontend (User Interface):

React.js (dynamic and responsive UI)

HTML5, CSS3, JavaScript

TailwindCSS / Bootstrap

**Backend (Server-side):**

Node.js with Express.js (lightweight and fast)

OR Django / Flask (Python stack)

RESTful APIs / GraphQL

**Database (Inventory Data Storage):**

MySQL/PostgreSQL (relational)

OR MongoDB (NoSQL option)

**Authentication & Security:**

JWT / OAuth2

HTTPS & Data Encryption

**Deployment / Hosting:**

AWS, Google Cloud, or Azure

Docker for containerization

Git + GitHub/GitLab for version control

**Optional Add-ons:**

Barcode/QR Code Scanning

Inventory Alerts (Email/SMS/Push)

Analytics & Reporting (Chart.js, Recharts, Power BI)

---

**Implementation Steps**

**1. Requirement Analysis**

Identify store needs (stock monitoring, sales tracking, alerts).

Define user roles (Manager, Staff, Admin).

**2. System Design**

Database schema: Products, Categories, Sales, Purchases, Users.

UI: Dashboard, Add/Update stock, Cart/Checkout.

APIs for product management, sales, reporting.

**3. Development**

Frontend: React.js with tables, search, filters.

Backend: APIs in Node.js/Express or Django.

Database: MySQL/PostgreSQL setup.

**4. Integration**

Connect frontend and backend with APIs.

Enable barcode/QR scanning.

Real-time stock updates.

**5. Testing**

Unit testing for stock, checkout, sales.

Integration testing (UI → API → DB flow).

User acceptance testing.

**6. Deployment**

Deploy backend on cloud (AWS/GCP).

Host frontend on Netlify/Vercel.

Docker for scalability.

**7. Maintenance**

Regular updates, bug fixes, new features.

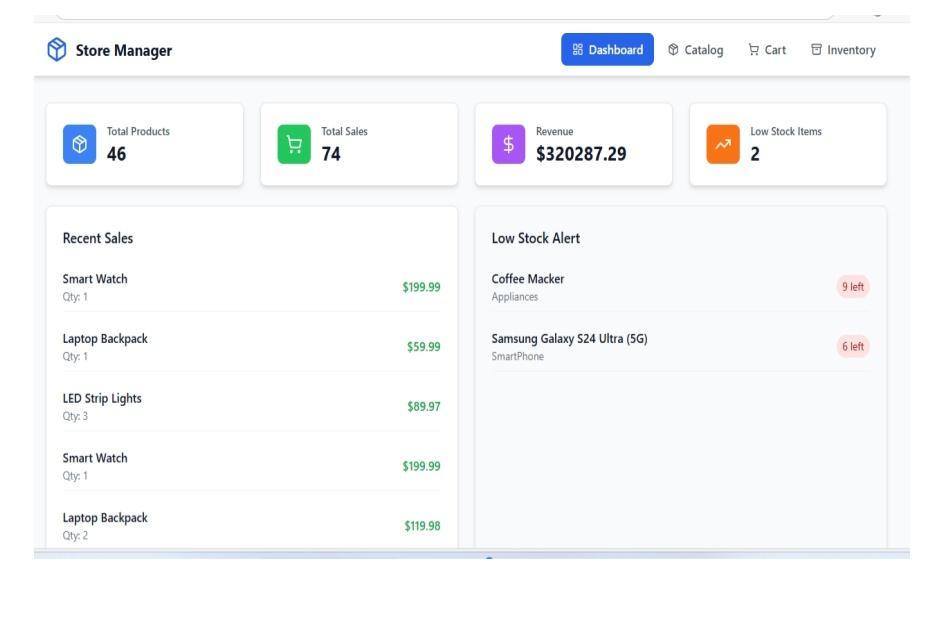
Backup & recovery for database.

Continuous monitoring.

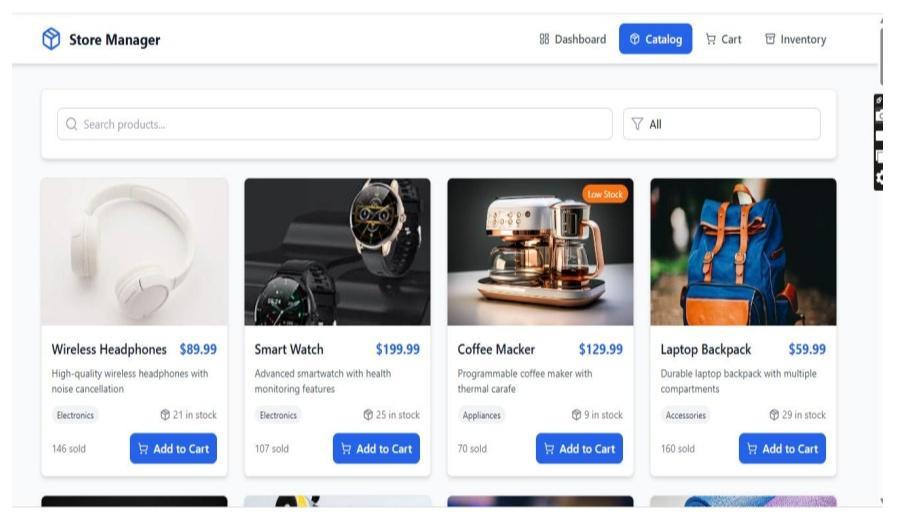
---

**Output & Result**

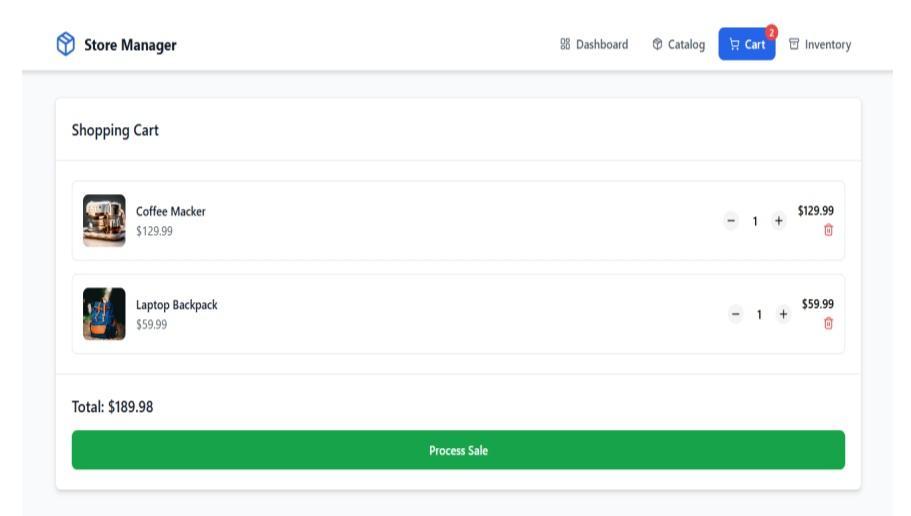
**Dashboard**



**Catalog**



**Cart**



**Inventory**

