📄 Project Documentation

**STORE MANAGER : KEEP TRACK OF INVENTORY**

1.Introduction:

**Project Title:**

*Inventory Management System – Store Manager*



**Frontend**: React.js

***Team id :***

*NM2025TMID29932-3*

***Team leader:***

* **DHARSHINI N**  *-202400152@sigc.edu* **Team Members:**
* **THULASIYA T** *-202400544@sigc.edu*
* **SARANYA G** -*-202400678@sigc.edu*

**Abstract:**

The **Inventory Management System – Store Manager** is a web-based application designed to help retail managers and store administrators effectively monitor and manage product stock. With real-time updates, intuitive UI, and efficient workflows, the system reduces manual effort and prevents errors related to inventory shortages or overstocking.

2.Project management:

**Purpose:**

The purpose of this project is to provide store managers with a **centralized system** to manage inventory, track sales, and monitor stock availability. By automating stock management, the system minimizes human error, saves time, and ensures business continuity.

**Objectives:**

* Provide a **user-friendly dashboard** with clear stock insights.
* Enable CRUD operations (*Create, Read, Update, Delete*) on products.
* Maintain stock levels and send **low-stock alerts**.
* Support **search, filter, and category-based organization** of items.
* Implement **user authentication** for secure access.
* (Future scope) Enable report generation and analytics for decisionmaking.

**Key Features:**

* **Product Management** → Add, update, delete products.
* **Stock Monitoring** → Alerts for low/inactive stock.
* **Category Management** → Organize products by category.
* **Search & Filter** → Quick access to product information.
* **Authentication** → Manager/Admin login functionality.
* **Reports (Future)** → Downloadable analytics and performance

reports.

1. System Architecture:

**Component Structure:**

* + **App.js** – Root component handling routes and global providers.
  + **Dashboard** – Displays stock summaries, sales statistics, and alerts.
  + **ProductList** – Product catalog with search, filter, and pagination.
  + **ProductForm** – Add/Edit product details.
  + **StockAlert** – Highlights items below reorder level.
  + **Reports (Future)** – Generate reports (PDF/Excel).
  + **Auth Components** – Login, Logout, Role-based access.
  + **UI Components** – Navbar, Sidebar, Buttons, Modals.

**State Management:**

* + **Global State:** Managed using **Context API / Redux Toolkit** for:
  + Authentication
  + Product data
  + Stock levels & notifications
  + **Local State:** Managed with **React Hooks (useState, useEffect)** for:
  + Form input handling
  + UI modals
  + Search/filter controls
  + **Routing (React Router v6):**
  + /login → Login page
  + /dashboard → Inventory overview
  + /products → Product listing
  + /add-product → Product creation form • /edit-product/:id → Product update form

1. Setup Instructions:

**Prerequisites:**

* + - Node.js (LTS ≥ 16.x)
    - npm or yarn
    - Git (for version control)
    - VS Code (recommended IDE)

**Installation:**

# Clone the repository git clone [repository-url]

#Navigate to project folder

cd client

# Install dependencies

npm install

#Configureenvironmentvariables

cp .env.example .env

**Extra Step for Windows Users:**

* 1. Open Windows PowerShell (Admin).
  2. Run:
  3. set-executionPolicy unrestricted
  4. Type Y and press Enter.

1. Folder Structure:

client/

├── public/ # Static assets

├── src/

│ ├── components/ # Reusable UI components (Navbar, Button, Modal, etc.)

│ ├── pages/ # Page-level components (Dashboard, Products, Login, etc.)

│ ├── assets/ # Images, icons

│ ├── hooks/ # Custom hooks (useAuth, useFetch)

│ ├── context/ # Context API for global state (AuthContext,

ProductContext)

│ ├── utils/ # Helper functions (API calls, formatters, validators)

│ ├── App.js # Main app with routes

│ └── index.js # React entry point

1. Running the Application:
   * he development serverStart t o npm start
   * Application runs at  [http://localhost:3000](http://localhost:3000/)
2. Component Documentation:
   * **Dashboard** → Displays key statistics: total products, low-stock warnings, sales trends (future).
   * **ProductList** → List of products with **search, filter, pagination, sorting**.
   * **ProductForm** → Form for **Add/Edit** product details.
   * **StockAlert** → Dedicated section showing products below reorder level.
   * **Navbar & Sidebar** → Provides intuitive navigation.
   * **Auth Pages** → Login, Logout, Session Handling.
3. State Management:

**Global state:**

* + Authentication status (user role: Manager/Admin).
  + Product inventory list (synchronized with backend).

**Local State:**

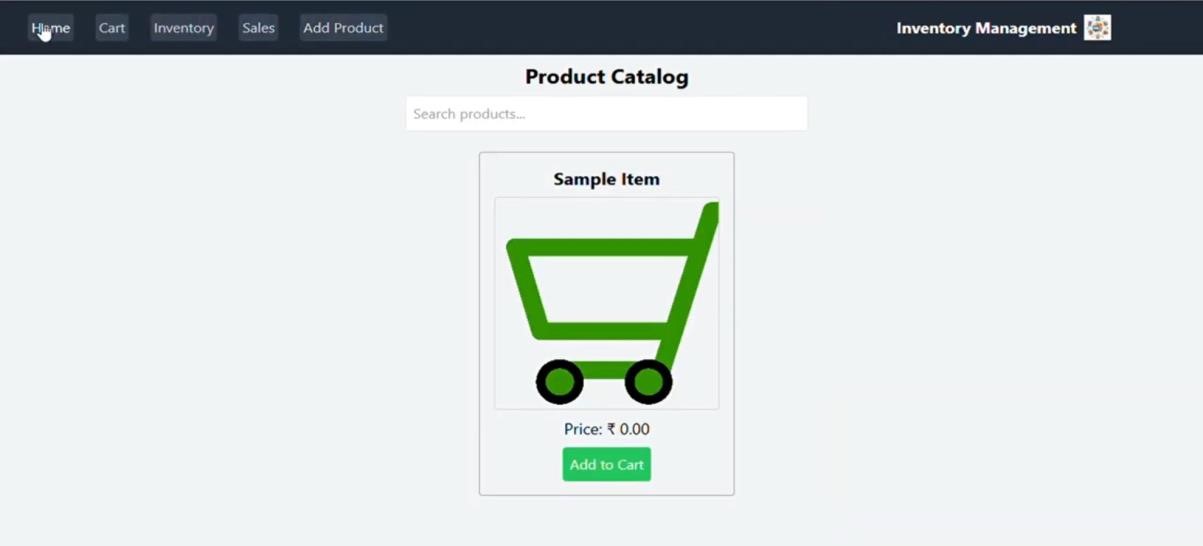
* + Form inputs (controlled components).
  + Modal visibility (e.g., add product modal).
  + Search and filter queries.

1. User Interface:

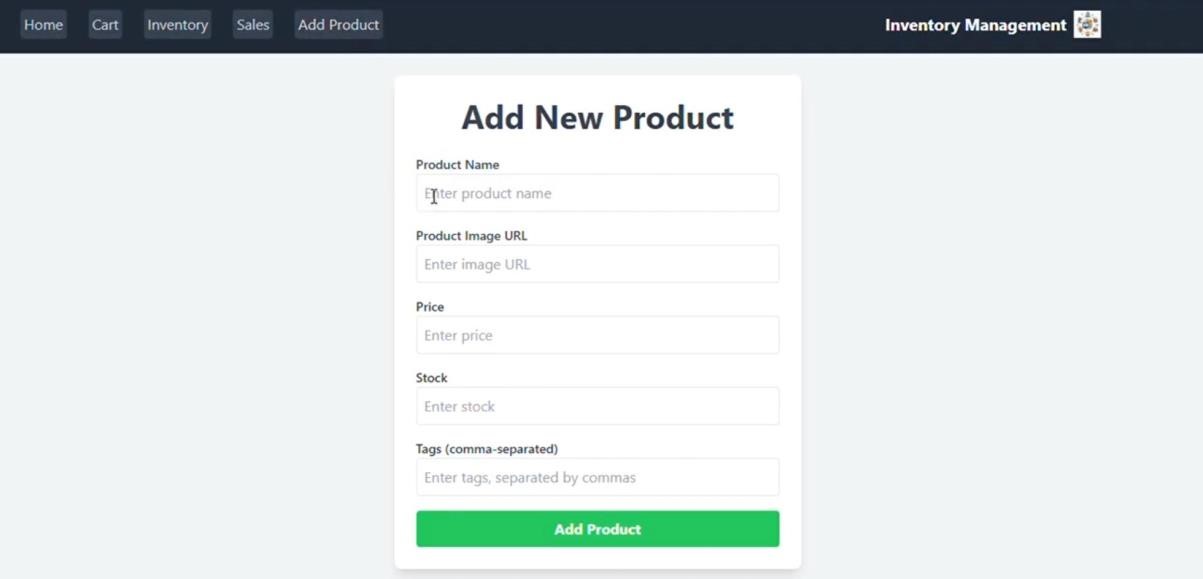
Suggested UI Screens (to be added as screenshots):

* + Login Page
  + Dashboard Overview
  + Product List with search/filter
  + Add/Edit Product Form
  + Stock Alerts & Notifications

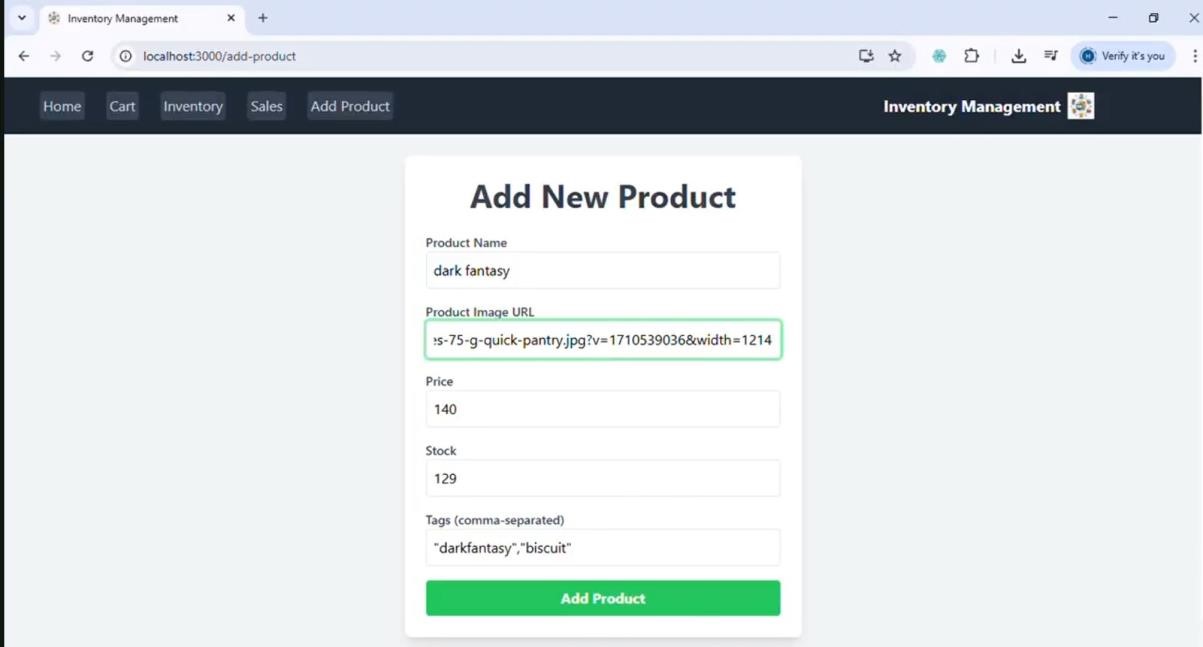
**Displays the name of the system**



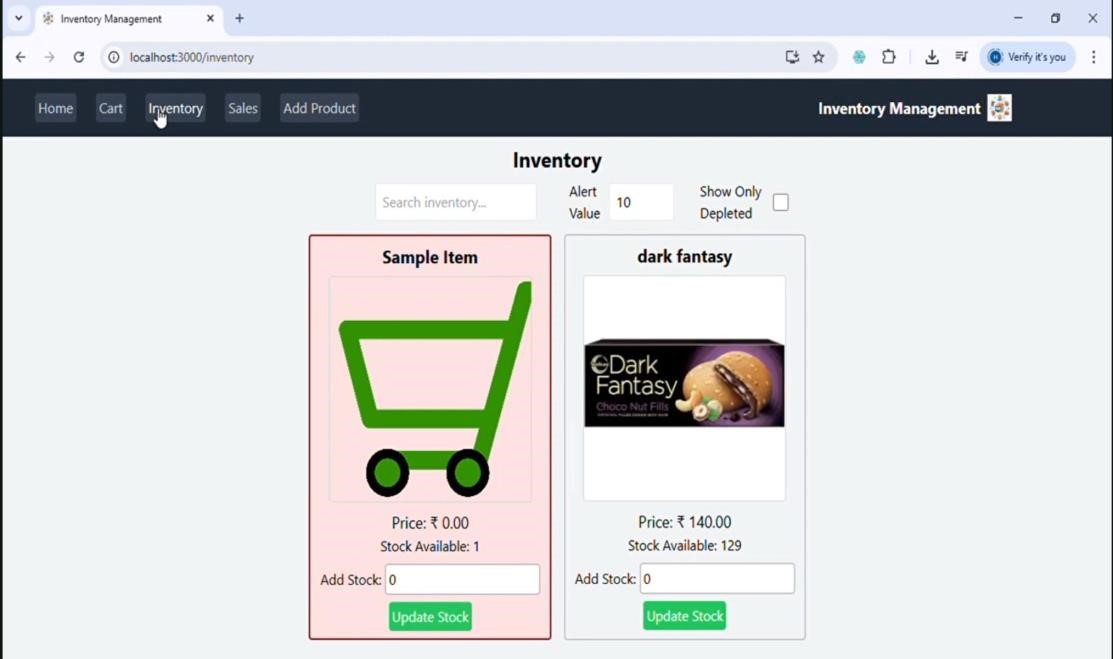
**Name of the product you are adding**



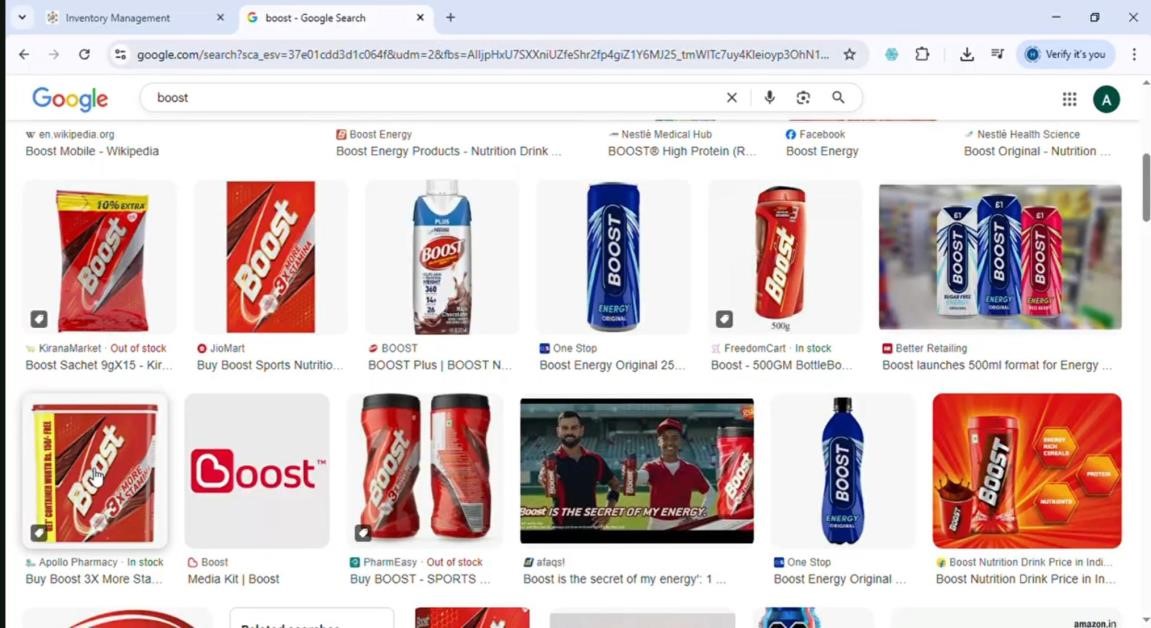
**Copying the image address link and paste**



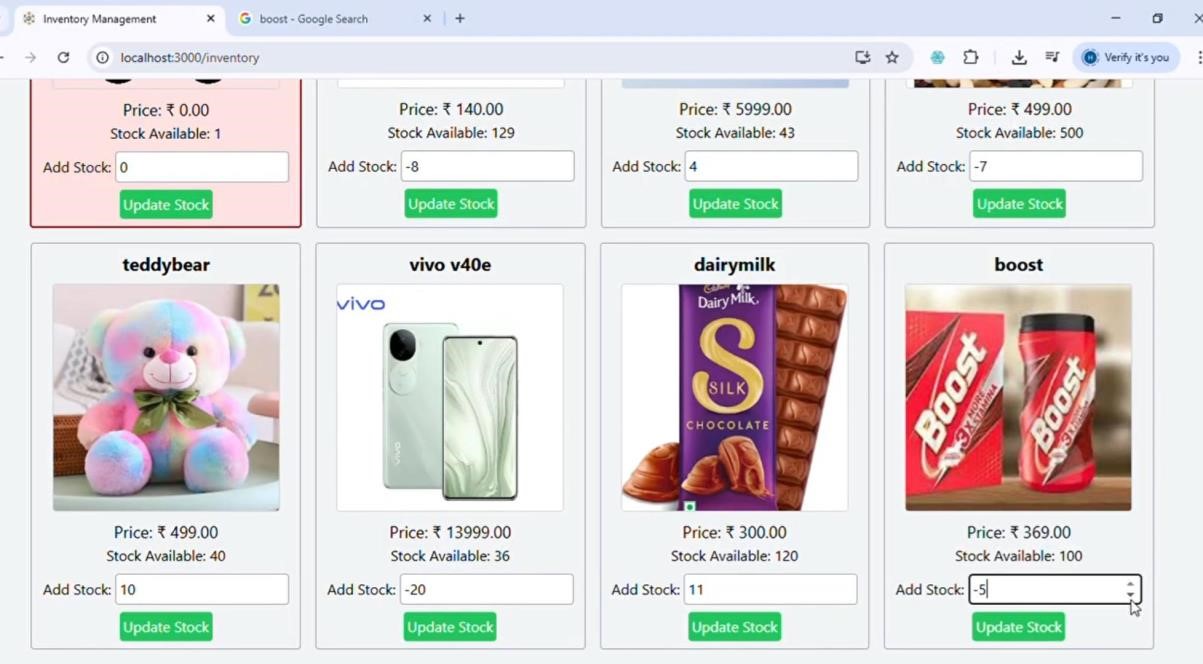
“**Inventory**” **page of the system shows two products in the inventory list.**

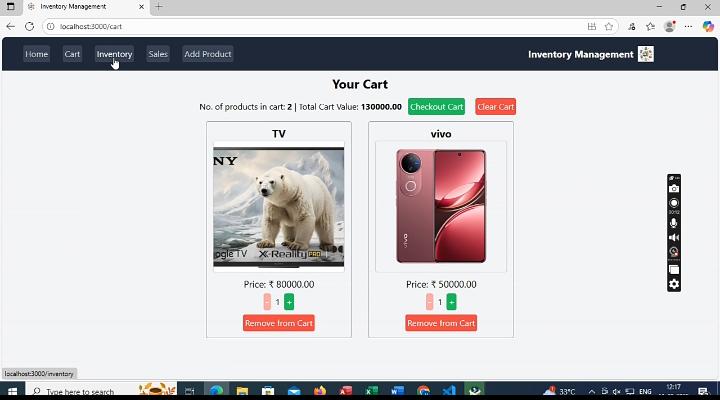


**Searching the image of address**



**Each product is displayed in a card format, showing:**



**Remove the cart:**

1. Styling:
   * **CSS Frameworks**: Tailwind CSS (preferred) / Material UI
   * **Responsive Design**: Mobile-first approach using Flex/Grid.
   * **Theming**: Option for Light/Dark mode (future scope).
   * **Animations**: Smooth UI transitions with Framer Motion.
2. Testing:

**Strategy:**

* + **Unit Tests** → React Testing Library + Jest (for components).
  + **Integration Tests** → Product flow: *Add → Edit → Delete*.
  + **E2E Tests (Optional)** → Cypress for full application flow.

1. Deployment:
   * **Local Development** → Runs on localhost:3000.
   * **Production Build** → npm run build
   * **Hosting Options** → Netlify, Vercel, GitHub Pages, or AWS Amplify.
2. Known Issues:
   * Pagination may not scale well for **very large datasets**.
   * No offline support (requires internet connection).
   * Reports & analytics not yet implemented.
3. Future Enhancements:
   * Barcode Scanner integration (faster product entry).
   * Export reports (Excel/PDF).
   * Advanced analytics (top-selling products, trends).
   * Multi-user roles: Manager, Staff, Admin.
   * Push Notifications for low stock alerts.
   * Mobile-friendly Progressive Web App (PWA).