



Store Manager: Keep Track of Inventory

Introduction

Project Title: Store Manager: Keep Track of Inventory

Team ID : NM2025TMID40071

Team Leader : Karthik Krishnan G S

NM Id: 32C13D770FC06BD94004F02EEBBA3DA0

Email:karthikkrishnanbca2024@gmail.com

Team members:

Name : Mohammed Shuhail M

NM Id: 46016A565E3D0D68F5C4853889466075

Email:mohammedshuhailbca2024@gmail.com

Name: Arun Kumar M

NM Id: F464978966789FC9CD04E593EAE4A1F0

Email:arunbca24@gmail.com

Name: Rishikesavan R

NM Id: C82DC177A9AC5D2882446122FDF4A139

Email:r.rishikesavanbca2024@gmail.com

Name: Praveen Kumar D

NM Id: 59EDD1719AAD7B5B42F409235BD1F58B

Email:praveenkumarpkmsd16@gmail.com

Project Overview:

Purpose:

The objective of this project is to design and develop a comprehensive **Inventory Management System** that enables efficient tracking, updating, and management of stock in a store. The system aims to simplify day-to-day inventory operations by providing real-time stock updates, automated alerts for low stock, and streamlined sales management through cart and checkout features.

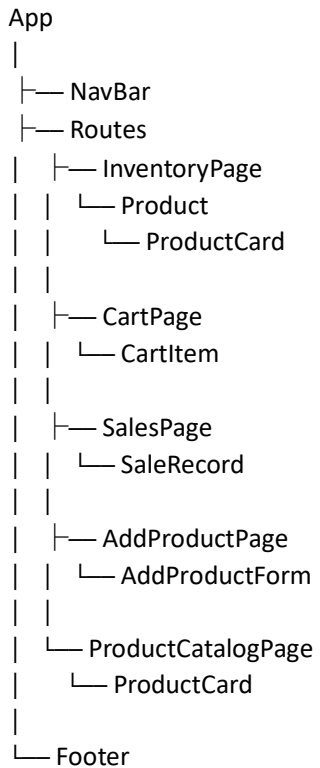
Features:

- Maintain accurate inventory records with seamless updates on sales and new stock additions.
- Facilitate quick and efficient sales through a cart and checkout mechanism while ensuring inventory consistency.
- Add and manage new products with essential details like name, image, price, stock, and tags.
- Provide alerts for depleting stock levels to support timely replenishment.
- Enable easy search and retrieval of products from the inventory and product catalog.
- Keep a record of all sales with details such as sale value, products sold, and date/time for accountability and reporting.

This project ultimately seeks to enhance **efficiency, accuracy, and visibility** in inventory and sales management, reducing manual effort and minimizing the risk of stockouts or overstocking.

Architecture

Component Structure:



State Management:

- Using **React Context API** to manage global states:
 - ❖ **products:** Array of current inventory items.
 - ❖ **cartItems:** Items added to cart.
 - ❖ **salesHistory:** List of sales records.
- **Methods exposed:**
 - ❖ `addProduct(productObject)`
 - ❖ `removeProduct(productId)`
 - ❖ `addToCart(productId)`
 - ❖ `removeFromCart(productId)`
 - ❖ `recordSale(cartItems)`

Routing:

- Implemented using **react-router-dom**:
 - ❖ **/inventory** – Inventory management page.
 - ❖ **/cart** – Shopping cart view.
 - ❖ **/sales** – Sales history page.
 - ❖ **/add-product** – Add new product page.
 - ❖ **/catalog** – Product catalog for customers.

Setup Instructions

Prerequisites:

- Node.js (\geq v14)
- npm (comes with Node.js)

Installation Steps:

1) Clone the Repository

```
>>git clone https://github.com/coderkarthikkrishnan/store-manager.git
```

```
>>cd Storemanager
```

2) Install Dependencies

Before running the app, install all required packages:

```
>>npm install
```

3) Required Packages

Make sure the following packages are installed:

```
>>npm install react-router-dom react-toastify
```

(If already installed, you can skip this step.)

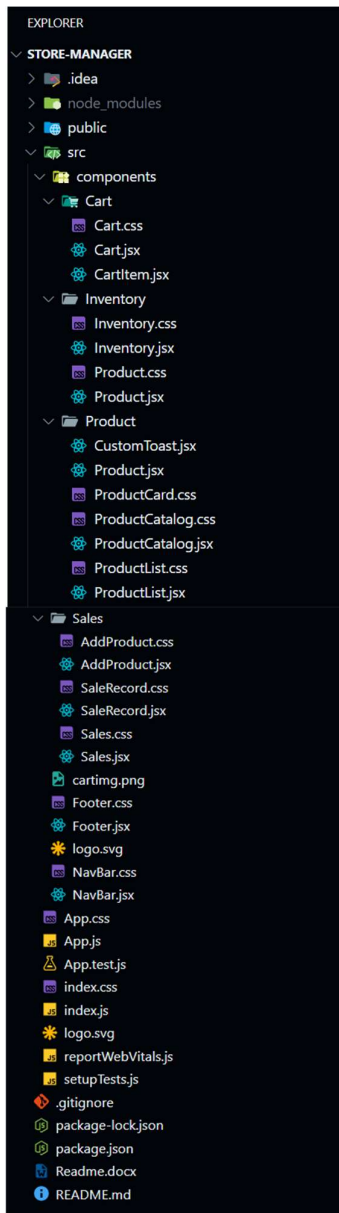
4) Start the Development Server

Run the app locally:

>>npm start

This will start the app at <http://localhost:3000/>

Project structure:



Component Documentation

Key Components

- **NavBar.jsx**
Provides site navigation links to Inventory, Cart, Sales, Add Product, and Product Catalog.
- **Inventory.jsx**
Displays list of inventory products using Product.jsx and ProductCard.jsx.
- **Cart.jsx**
Displays the user's shopping cart. Uses CartItem.jsx for individual cart items.
- **Sales.jsx**
Shows all recorded sales using SaleRecord.jsx.
- **AddProduct.jsx**
Form allowing new product addition to inventory.
- **ProductCatalog.jsx**
Displays catalog of products available to browse.

Reusable Components

- **ProductCard.jsx**
Displays product details (name, image, price, stock) with Add to Cart button.
- **CartItem.jsx**
Displays individual cart item info with options to increase quantity or remove.
- **CustomToast.jsx**
Provides custom toast notifications on actions like adding to cart or submitting form.

State Management

Global State

- Managed using React Context API:
Stores products, cartItems, and salesHistory.
- State Flow Example:
 - Adding a new product → updates products.
 - Adding an item to cart → updates cartItems.
 - Recording a sale → updates salesHistory.

Local State

- Used for forms in components such as AddProduct.jsx:

```
const [productName, setProductName] = useState("");  
const [productPrice, setProductPrice] = useState("");  
const [stockQuantity, setStockQuantity] = useState("");
```

User Interface

- ❖ Inventory Page
- ❖ Product catalog page
- ❖ cart Page
- ❖ Sales History
- ❖ Add Product Form

Styling

- **CSS Libraries/Frameworks:**
Pure CSS with modular CSS files for each component (e.g., ProductCard.css, Cart.css).
- **Theming:**
No external theming system; uses CSS variables for primary colors and fonts.

Testing

- ❖ Manual testing during milestones
- ❖ Tools: Postman, Chrome Dev Tools

Screenshots or Demo

Inventory Page :

inv track.

Inventory Catalog Sales Add Product

Search products

Name or tag...

Low-stock alert ≤ 3

Product Image	Product Name	Category	Price	Stock	Update	Remove
	Tissue box	Tissue, accessory	₹500	10	Update	Remove
	Milk Biscuit	food, biscuit, snack	₹10	30	Update	Remove
	Notebook A4	stationery	₹149	25	Update	Remove
	Bluetooth Speaker	electronics, audio, music	₹1299	15	Update	Remove
	Instant Coffee	beverage, coffee, food	₹299	40	Update	Remove
	Ballpoint Pen Set	stationery, writing	₹99	50	Update	Remove
	LED Desk Lamp	electronics, lighting, home				
	Pack of Green Tea	beverage, tea, food				
	Portable Water Bottle	accessory, travel, home				
	Yoga Mat	fitness, exercise, health				
	Ceramic Mug	home, kitchen, drinkware				
	Desk Organizer	home, office, stationery				

Cart Page Example:

inv track.

Inventory Catalog Sales Add Product


Cart (5 items)

Image	Product	Price	Qty	Subtotal	Remove
	Portable Water Bottle	₹250	1	₹250	Remove
	Ceramic Mug	₹349	1	₹349	Remove
	Milk Biscuit	₹10	2	₹20	Remove
	Tissue box	₹500	1	₹500	Remove


Clear Cart

Total: ₹1119 Checkout

Sales History Example:




InventoryCatalogSalesAdd Product

0

Sales

Date/Time	Products	Total
9/3/2025, 9:42:34 PM	Ceramic Mug (x1), Bluetooth Speaker (x1), Notebook A4 (x1), Tissue box (x1), Milk Biscuit (x1)	₹2307



Quality products, curated for you. Shop smart, live better.

Quick Links


- Inventory
- Catalog
- Cart
- Sales

Contact


Email: gskarthikkishnan@gmail.com
Phone: +91 7305962714

© 2025 InvTrack. All rights reserved.

Add Product Form:



InventoryCatalogSalesAdd Product

0

Add Product

Name

Image URL

Price (₹)

Stock

Tags (comma separated)

Add

Reset



Quality products, curated for you. Shop smart, live better.

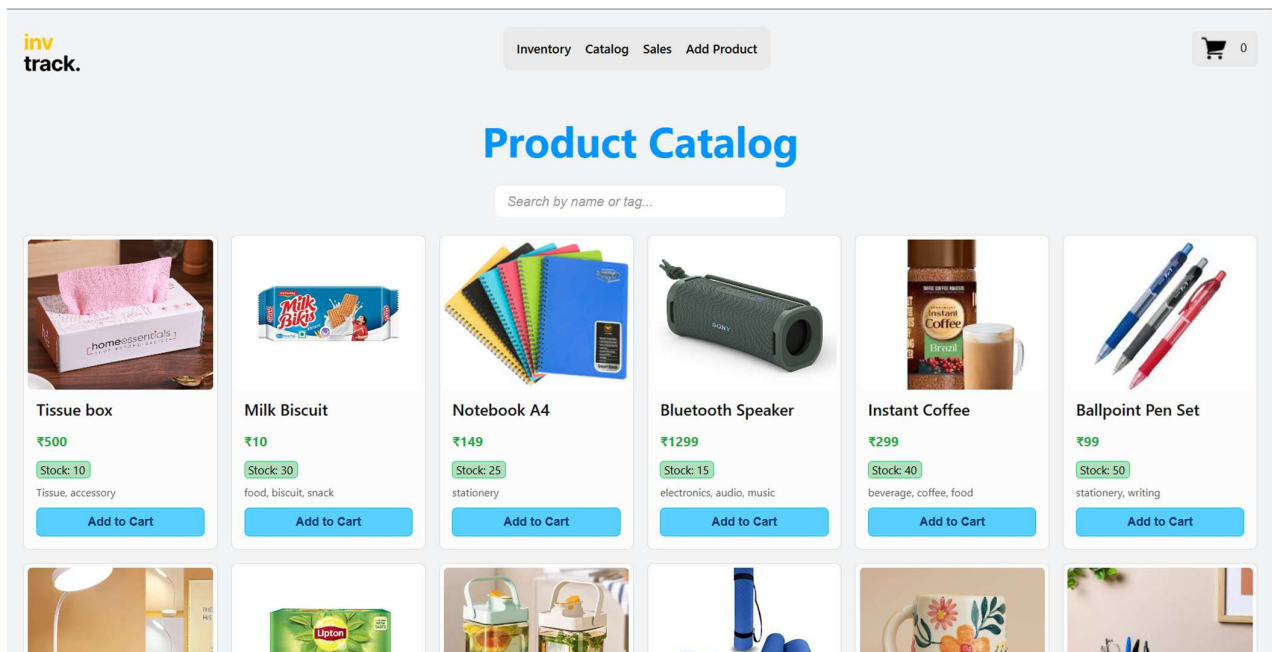
Quick Links

- Inventory
- Catalog
- Cart

Contact

Email: gskarthikkishnan@gmail.com
Phone: +91 7305962714

Product Catalog:



Known Issues

- ❖ Cart state resets on page refresh (needs localStorage persistence).
- ❖ No input validation on Add Product form.
- ❖ Missing error handling on API calls.
- ❖ No user authentication (admin vs customer).

Future Enhancements

- ✓ Persist cart state in localStorage or backend storage.
- ✓ Add input validation (Formik + Yup).
- ✓ Add role-based authentication.
- ✓ Implement proper error handling for API calls.