Project Documentation

STORE MANAGER:keep track of inventory.

1. Introduction

• Project Title: Store Manager: keep track of inventory .

TEAM ID: NM2025TMID33787

TEAM LEADER: NOOR FAYISHA H. - fayishafazila25@gmail.om

TEAM MEMBER: NASEEHA S. - naseehaa2006@gmail.com

TEAM MEMBER:RAHIMA SAJEENA. - rahimasajeena28@gmail.com

TEAM MEMBER:RAj

The purpose of the Store Manager pro

ject is to make inventory management easier and

more efficient. In many small shops and stores, products are managed manually in notebooks or registers. This often leads to mistakes like missing items, wrong stock counts, or difficulty in updating product details.

FEATURES:

Store Manager: Keep Track of Inventory .

Here are some key features and a description of an inventory management system:

Inventory Management: Inventory Management helps maintain healthy stock levels in a store and acquire them in time.

Stock Updates: Stock will automatically update on sale of products, and it can be updated on adding new stock.

Cart: Products can be added to cart for a particular sale and quantity can be added to each product.

Checkout at Cart: Upon checkout, cart is cleared, inventory is updated, and a sale record is made.

Adding New Products to Inventory: New products can be added to the inventory by

providing product name, image URL, price, stock, tags.

Alert View for Depleting Stock: Depleting stocks are shown in red background, and alert requirements

Search Functionality for Products: Products in inventory and product catalog can be

searched.

Sale Records: All sale records are stored with sale value, products and datetime.

Architecture – Store Manager

The architecture of the Store Manager pro

ject follows a client–server model. It mainly

consists of three layers:

1. Frontend :

Developed using React.js.

Provides the user interface where shopkeepers can add, update, delete, and view products.

Ensures a simple and user-friendly interaction.

2. Backend :

Implemented using Node.js with npm packages.

Handles the logic of the application and runs the project commands.

Acts as a bridge between the frontend and the database.

3. Database:

MongoDB is used for storing product details like Product ID, Name, Price, and

Quantity.Provides permanent storage and easy retrieval of data.

Project setup and configuration

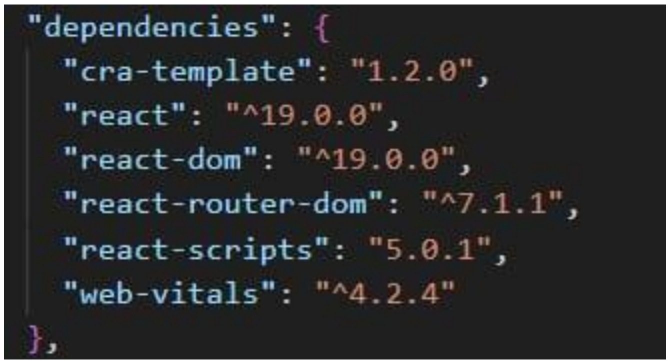
Project Setup

Step 1: Initialize a new React application.

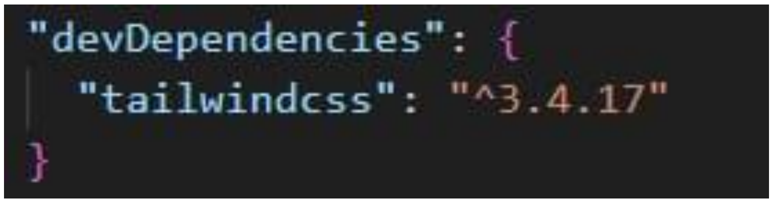
Use create-react-app or Vite for project

Step 2: Install dependencies.

Add tailwindcss, react, react-dom, and other necessary Step 3: Configure TailwindCSS.



```
"dependencies": {
  "cra-template": "1.2.0",
  "react": "^19.0.0",
  "react-dom": "^19.0.0",
  "react-router-dom": "^7.1.1",
  "react-scripts": "5.0.1",
  "web-vitals": "^4.2.4"
},
```



```
"devDependencies": {
  "tailwindcss": "^3.4.17"
}
```

Set up the tailwind.config.js file.

Add the required styles in index.css.

Step 4: Create a basic folder structure.

Example:

css

src/

??? components/

??? context/

??? hooks/

??? pages/

https://react.dev/learn/installation

https://react-bootstrap-v4.netlify.app/getting-started/introduction/

https://axios-http.com/docs/intro

https://reactrouter.com/en/main/start/tutorial

Inventory: Add hover effects and responsive search bar.

Cart: Highlight selected products and display alerts on low stock.

Sales: Use a clean table or card layout for sale records.

Step 3: Ensure consistency.

Apply a theme or consistent color palette.

Milestone 6 - Testing and Debugging

Step 1: Test individual components.

Ensure correct rendering and functionality.

Step 2: Test context and reducers.

Verify state updates and interactions with localStorage.

Step 3: Debug UI/UX issues.

Check for responsiveness and usability.

PRE-REQUISITES

Here are the key prerequisites for developing a frontend application using React.js:

Node.js and npm:

Node.js is a powerful JavaScript runtime environment that allows you to run JavaScript code on the local environment. It provides a scalable and efficient platform for building network applications.

Install Node.js and npm on your development machine, as they are required to run JavaScript on the server side.

Download: https://nodejs.org/en/download/

Installations: https://nodejs.org/en/download/package-manager/

React.js:

React.js is a popular JavaScript library for building user interfaces. It enables developers to create interactive and reusable UI components, making it easier to build dynamic and responsive web applications.

Install React.js, a JavaScript library for building user interfaces.

Create a new React app:

```
npx create-react-app my-react-app
```

Replace my-react-app with your preferred project name.

Navigate to the project directory:

```
cd my-react-app
```

Running the React App:

With the React app created, you can now start the development server and see your React application in action.

Start the development server:

```
npm start
```

This command launches the development server, and you can access your React app at http://localhost:3000 in your web browser.

HTML, CSS, and JavaScript: Basic knowledge of HTML for creating the structure of your app, CSS for styling, and JavaScript for client-side interactivity is essential.

Version Control: Use Git for version control, enabling collaboration and tracking changes throughout the development process. Platforms like GitHub or Bitbucket can host your repository.

• Git: Download and installation instructions can be found at: https://git-scm.com/downloads

Development Environment: Choose a code editor or Integrated Development Environment (IDE) that suits your preferences, such as Visual Studio Code, Sublime Text, or WebStorm.

• Visual Studio Code: Download from https://code.visualstudio.com/download

• Sublime Text: Download from https://www.sublimetext.com/download

• WebStorm: Download from https://www.jetbrains.com/webstorm/download

To get the Application project from drive:

Follow below steps:

Install Dependencies:

• Navigate into the cloned repository directory and install libraries:

```
cd store
```

```
npm install
```

Start the Development Server:

• To start the development server, execute the following command:

```
npm start
```

Access the App:

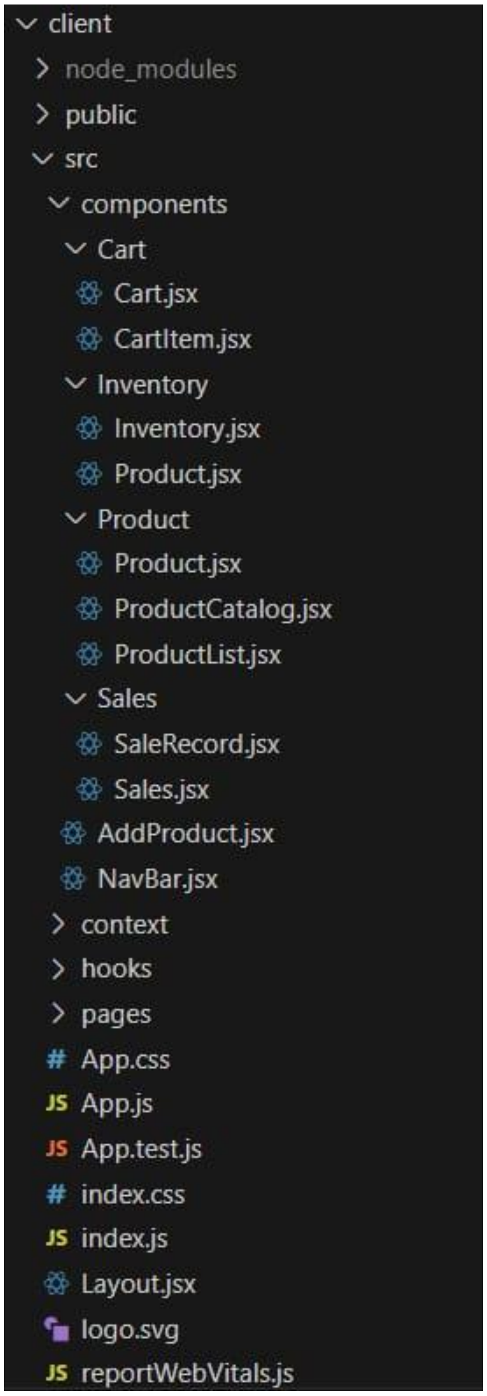
• Open your web browser and navigate to http://localhost:3000

• You should see the application's homepage, indicating that the installation and setup were successful.

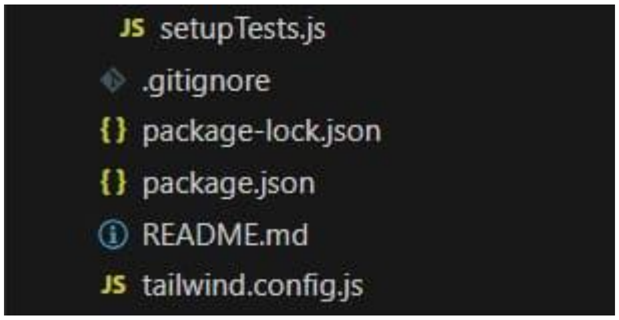
You have successfully installed and set up the application on your local machine. You can now proceed with further customization, development, and testing as needed

Project structure,:

The image is of the folder structure which shows all the files and folders that have been used in project



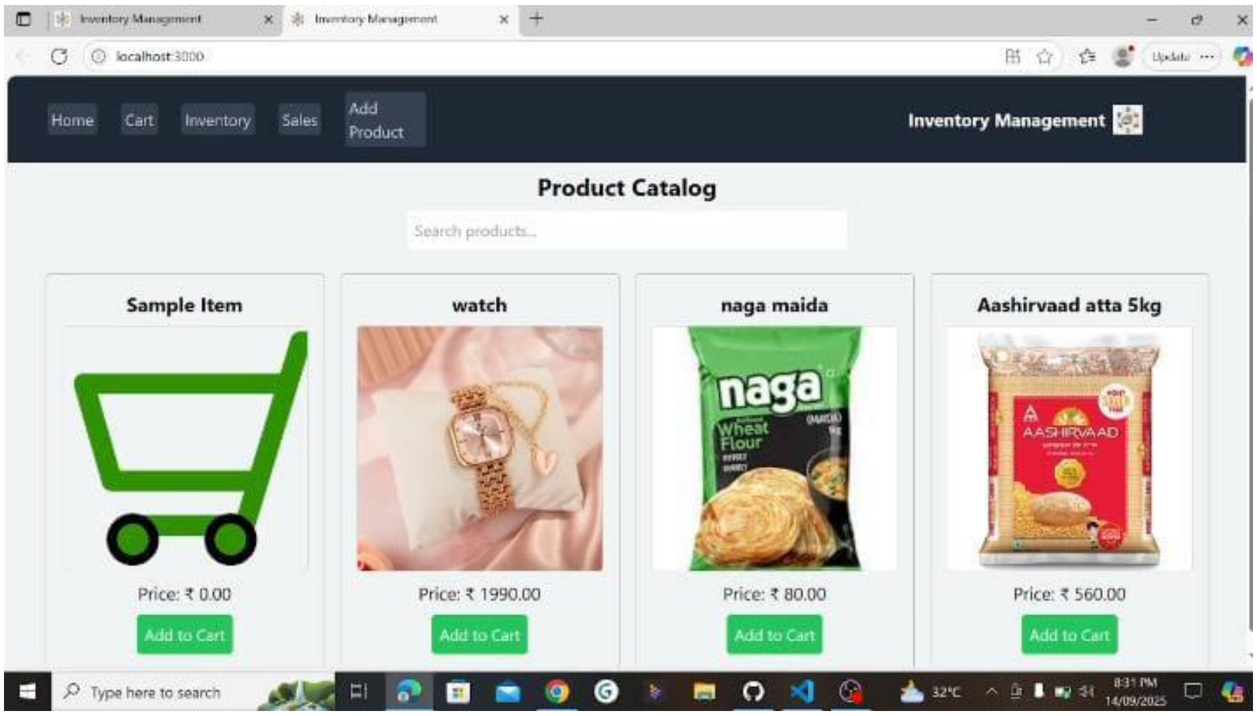
```
client
  node_modules
  public
  src
    components
      Cart
        Cart.jsx
        CartItem.jsx
      Inventory
        Inventory.jsx
        Product.jsx
      Product
        Product.jsx
        ProductCatalog.jsx
        ProductList.jsx
      Sales
        SaleRecord.jsx
        Sales.jsx
      AddProduct.jsx
      NavBar.jsx
    context
    hooks
    pages
    App.css
    App.js
    App.test.js
    index.css
    index.js
    Layout.jsx
    logo.svg
    reportWebVitals.js
```



```
    setupTests.js
  .gitignore
  package-lock.json
  package.json
  README.md
  tailwind.config.js
```

Project implementation and Execution:

Product catalog:


Product Catalog

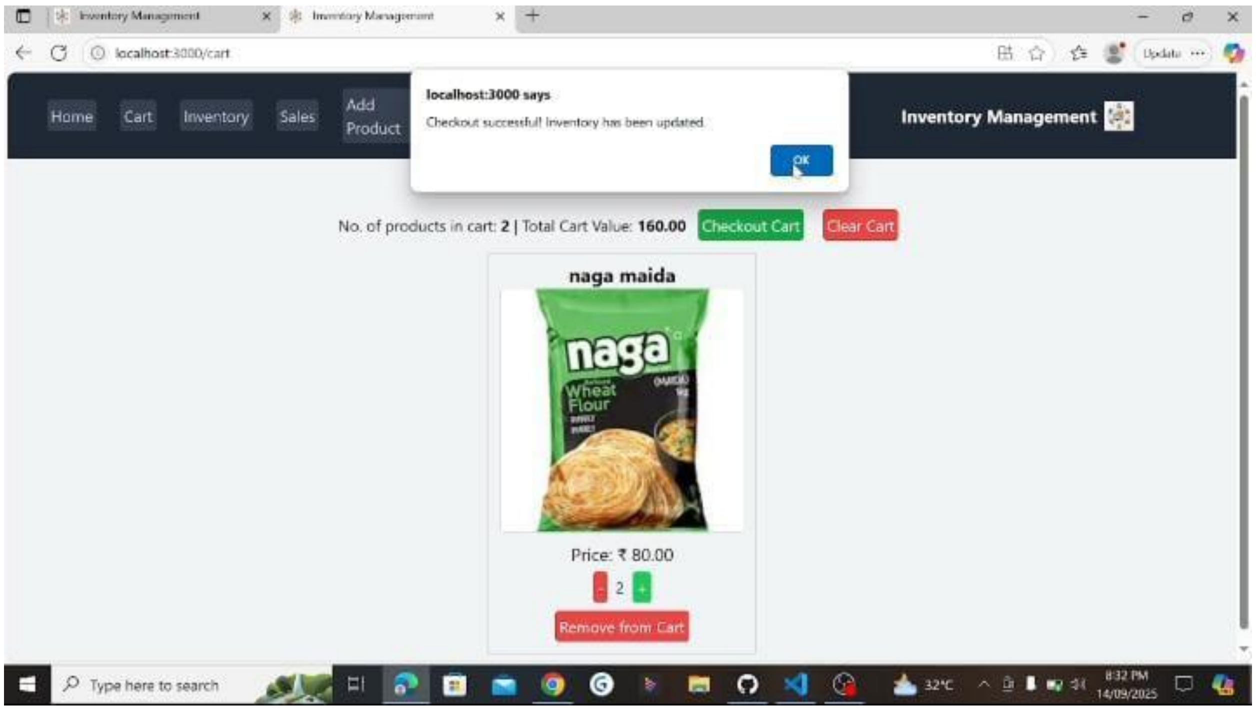
Sample Item
Price: ₹ 0.00

watch
Price: ₹ 1990.00

naga maida
Price: ₹ 80.00

Aashirvaad atta 5kg
Price: ₹ 560.00

Cart:

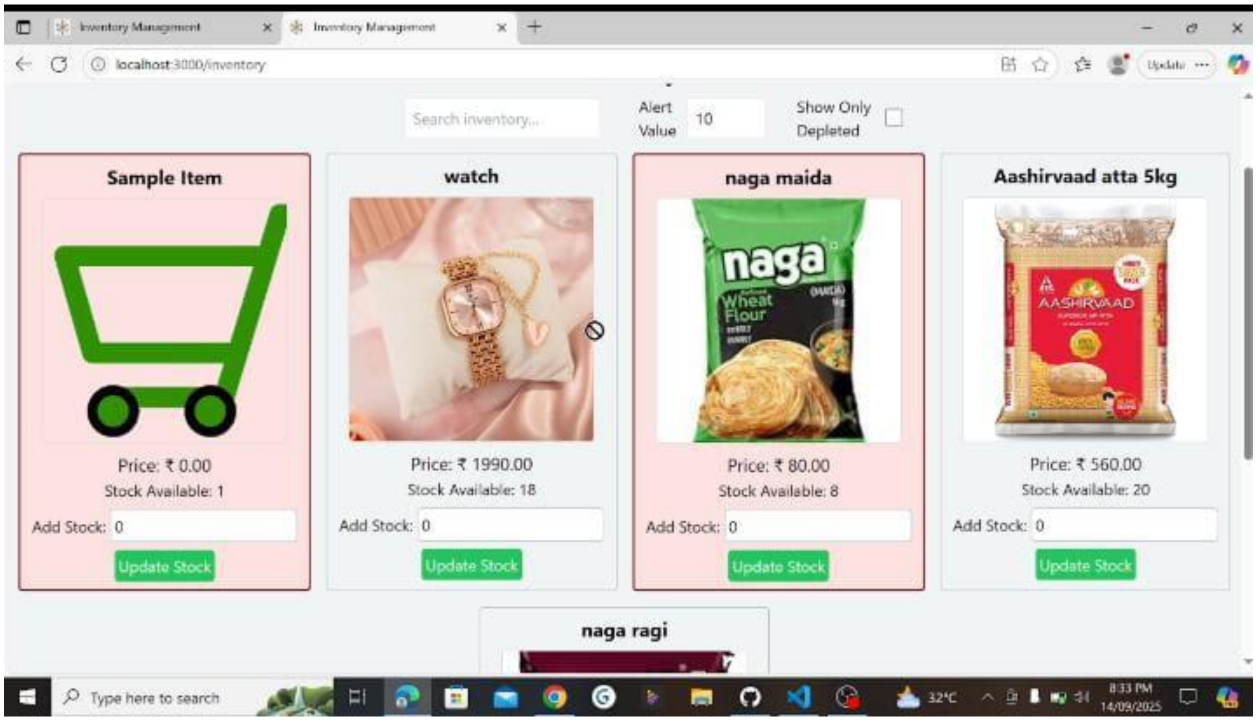

localhost:3000 says

Checkout successful! Inventory has been updated.

No. of products in cart: **2** | Total Cart Value: **160.00**

naga maida
Price: ₹ 80.00

Inventory:



Search inventory... Alert Value 10 Show Only Depleted

Sample Item
Price: ₹ 0.00
Stock Available: 1
Add Stock: 0
Update Stock

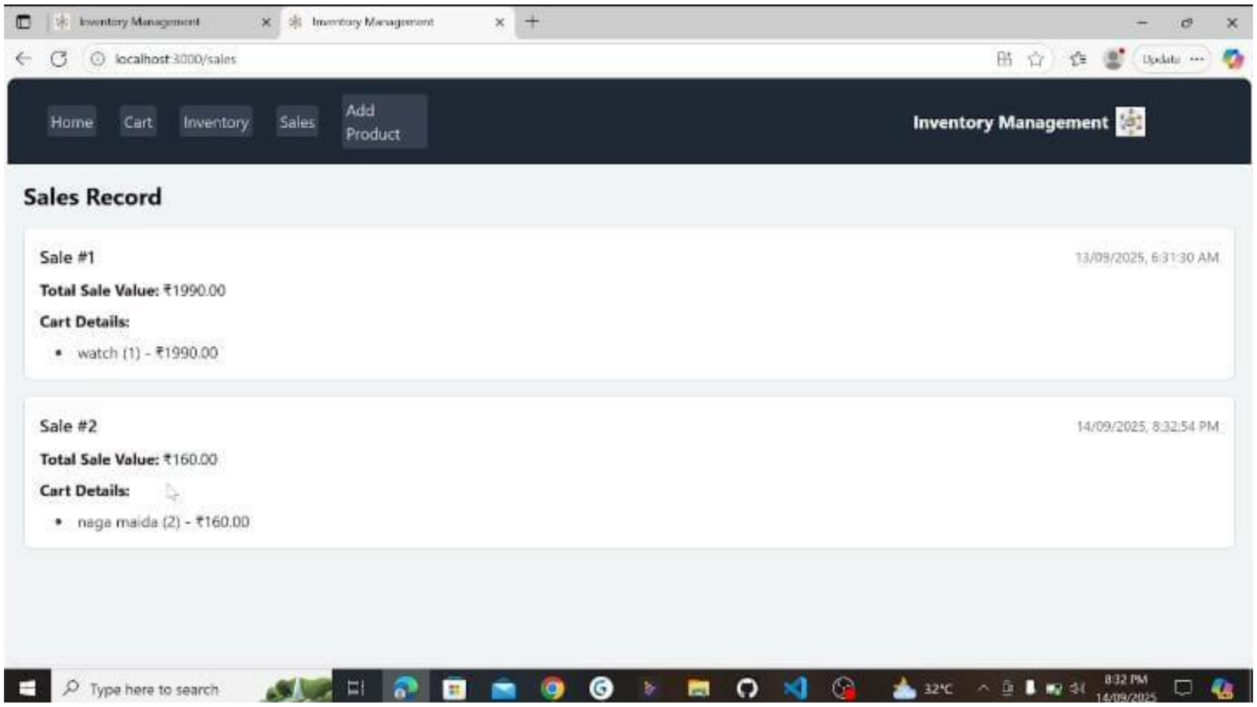
watch
Price: ₹ 1990.00
Stock Available: 18
Add Stock: 0
Update Stock

naga maida
Price: ₹ 80.00
Stock Available: 8
Add Stock: 0
Update Stock

Aashirvaad atta 5kg
Price: ₹ 560.00
Stock Available: 20
Add Stock: 0
Update Stock

naga ragi

Sales records :



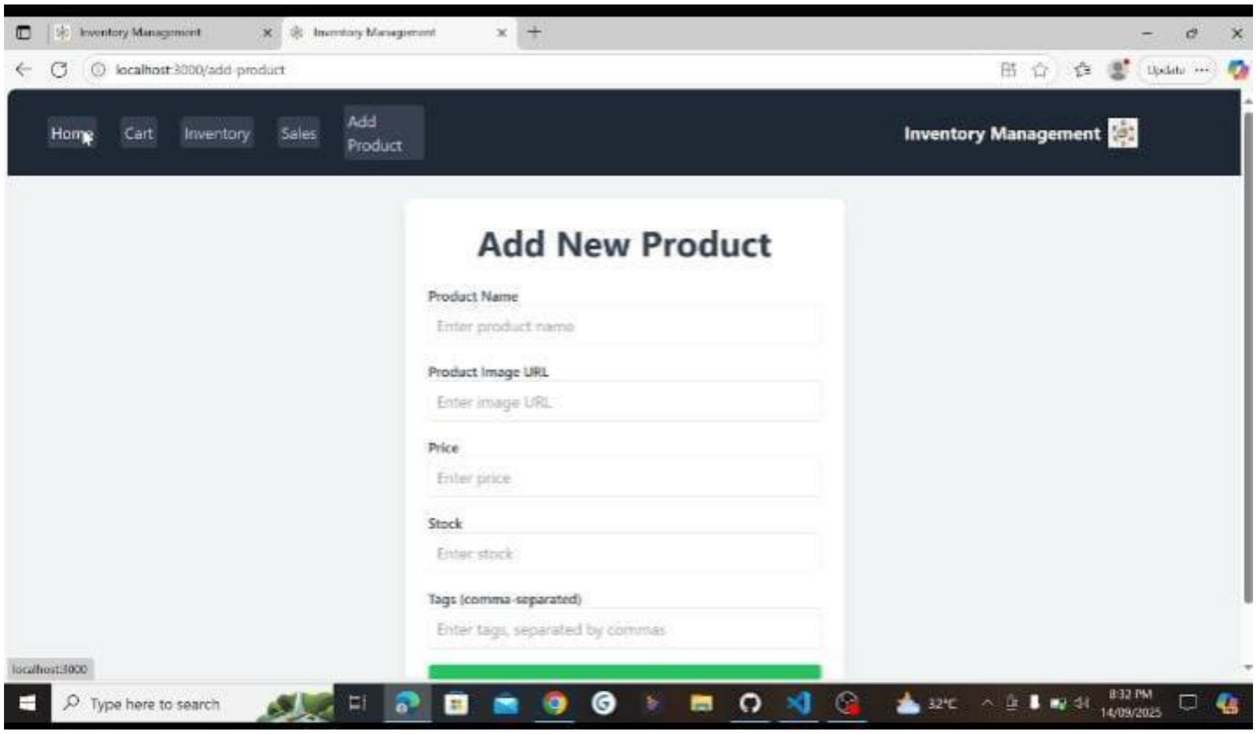
Home Cart Inventory Sales Add Product — Inventory Management

Sales Record

Sale #1 — 13/09/2025, 6:31:30 AM
Total Sale Value: ₹1990.00
Cart Details:
- watch (1) - ₹1990.00

Sale #2 — 14/09/2025, 8:32:54 PM
Total Sale Value: ₹160.00
Cart Details:
- naga maida (2) - ₹160.00

Add product :



API Documentation

Add Product API → Adds a new product.

Update Stock API → Updates stock when items are sold.

Get Products API → Displays product catalog.

Sales History API → Shows past transactions.

(Currently simulated with React state, can be extended to real backend in future.)

Authentication

Present project works without authentication.

Future upgrade:

Admin (Manager): Full access.

Cashier/User: Limited access.

Can be implemented using JWT or Firebase Authentication.

User Interface (UI)

Built with React and CSS.

Simple and responsive design.

Navigation bar for quick access to Home, Add Product, Cart, Sales History.

Easy-to-use forms and buttons for smooth interaction.

Testing

Unit Testing: Verified individual components.

Integration Testing: Ensured features work together correctly.

Functional Testing: Checked main workflows (Add Product → Cart → Checkout → Sales History).

User Acceptance Testing: Tested in http://localhost:3000 to ensure usability

Demo link:

https://drive.google.com/file/d/1IX2ezYZmRrwPybg7zVN8xdj48dFjtDXy/view?usp=drivesdk

Future Enhancements:

Connect to a real backend and database (Node.js + MySQL/MongoDB).

Implement user authentication with roles (Admin, Cashier).

Generate automated reports and analytics.

Deploy as a cloud-based web application for real-time usage.

Conclusions:

The Store Manager project successfully demonstrates how a small retail store can digitalize its inventory and sales process. It reduces manual errors, improves accuracy, and makes product management and billing faster and more efficient.

This project proves that even a simple system built using React.js can handle real-world store operations like adding products, managing stock, checkout, and sales history.

Overall, the project meets its objectives of being user-friendly, efficient, and reliable, and it lays a strong foundation for future improvements like authentication, database integration, and deployment.