NAAN MUDHALVAN PROJECT

Store Manager-Keep Track Of Inventory

1. Introduction

Project Title: Keep Track

Team id:NM2025TMID46178

Team leader Name: S.Saranya

Mail id:saranyasi2007@gmail.com

Team Members:

1. C.Rajamani– rajamanichinathampi@gmail.com Demo video making

2. P.Ramya – ramyasaanthi@gmail.com- Code Developer 3.

3.A.Rishika – rishikabanu7@gmail.com - Document

4. A.Samyuktha - samyukthaa34@gmail.com- Document

2. Project Overview

- **Purpose**:
  To help store managers efficiently track stock levels, manage product information, monitor sales, and prevent shortages or overstocking.
- **Features**:
  - Add, update, delete products
  - Track stock quantity
  - Low-stock alerts
  - Search/filter products
  - Dashboard with sales & stock overview

3. Architecture

- **Component Structure**:
  - `App.js` → Root component
  - `Header` → Navigation bar
  - `ProductList` → Displays all products
  - `ProductForm` → Add/Edit product
  - `InventoryDashboard` → Shows stock summary & alerts
  - `Footer` → Footer details
- **State Management**:
  - Context API (or Redux if advanced) for product state
  - Local state for forms
- **Routing**:
  - `react-router-dom` for navigation between Dashboard, Products, Add Product, etc.

4. Setup Instructions

- **Prerequisites**:
  - Node.js (LTS version)
  - npm or yarn
- **Installation**:
- `git clone <repo-url>`
- `cd inventory-app`
- `npm install`
- `npm start`

5. Folder Structure

```
inventory-app/
 ├── public/
 ├── src/
 │   ├── components/
 │   │   ├── Header.js
 │   │   ├── Footer.js
 │   ├── pages/
 │   │   ├── ProductList.js
 │   │   ├── ProductForm.js
 │   │   ├── Dashboard.js
 │   ├── context/
 │   │   ├── ProductContext.js
 │   ├── assets/
 │   │   ├── images/
 │   ├── App.js
 │   └── index.js
 └── package.json
```

- **Utilities**:
  - `utils/validation.js` → for input validation
  - `utils/alerts.js` → for low-stock alerts

6. Running the Application

```
cd inventory-app
npm start
```

The app runs at `http://localhost:3000`

7. Component Documentation

- **Key Components**:
  - `ProductList`: Displays all inventory items
  - `ProductForm`: Form to add/edit product
  - `Dashboard`: Stock summary, low stock alerts
- **Reusable Components**:
  - `Button`, `Modal`, `InputField`

8. State Management

- **Global State**:
  - Product list stored in `ProductContext`
- **Local State**:
  - Form fields handled with `useState`

9. User Interface

- Product List page (Table with Name, Quantity, Price)

- Add Product page (Form)
- Dashboard page (Summary + Alerts)

10. Styling

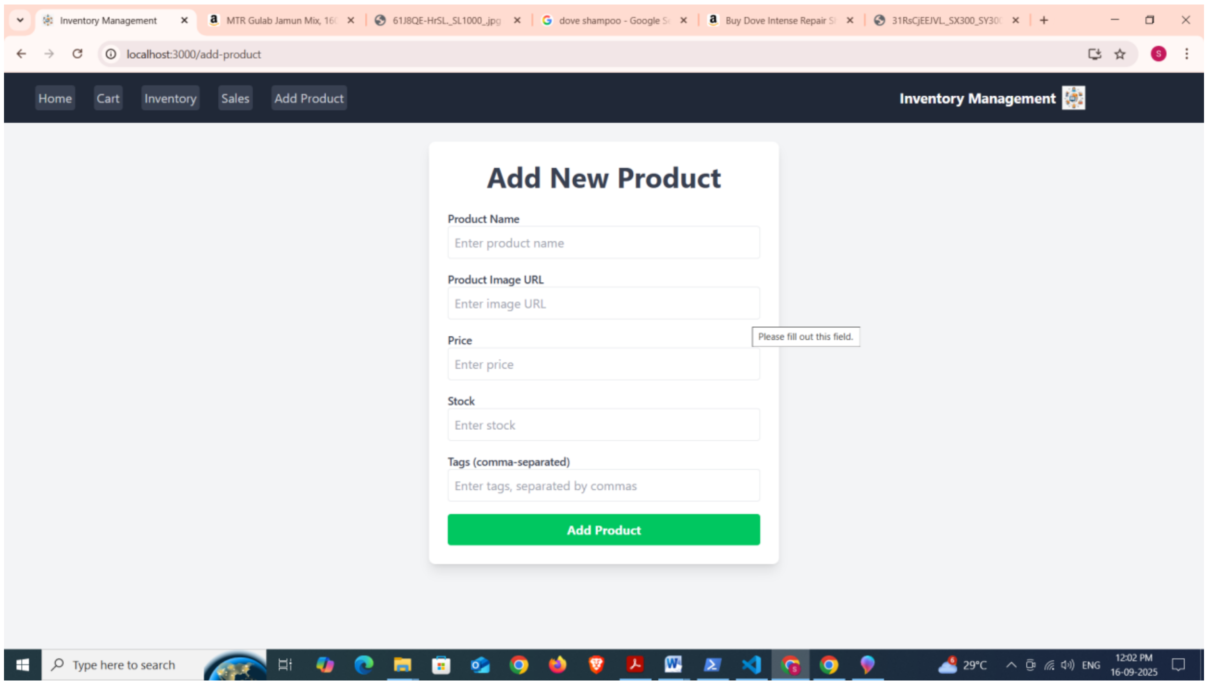
- **CSS Framework**: Tailwind CSS or Bootstrap
- **Theming**: Simple, professional store dashboard look

11. Testing

- **Strategy**:
  - Unit test: Add Product function
  - Integration test: Adding & Displaying in ProductList
- **Tools**: Jest + React Testing Library

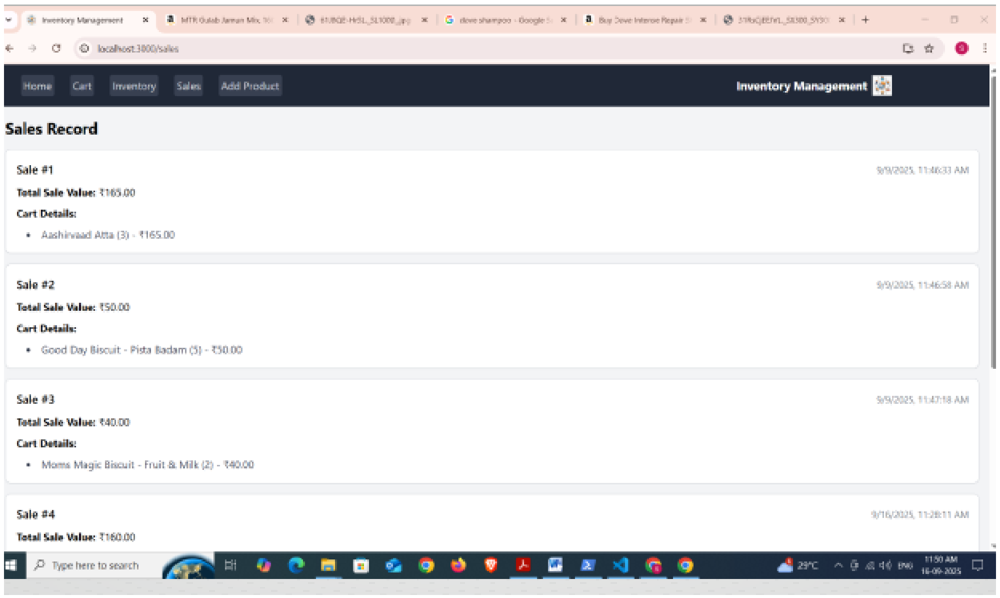
12. Screenshots / Demo

(Insert screenshots of UI: Dashboard, Product List, Add Product form)



Add New Product

Product Name
Enter product name
Product Image URL
Enter image URL
Price
Please fill out this field.
Enter price
Stock
Enter stock
Tags (comma-separated)
Enter tags, separated by commas



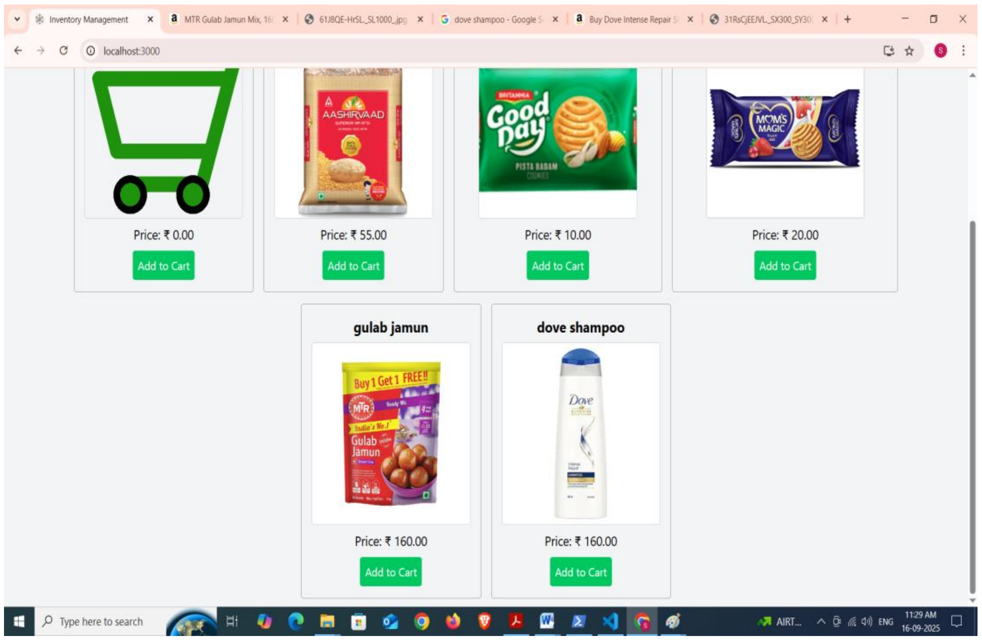
Sales Record

Sale #1 9/9/2025, 11:46:33 AM
Total Sale Value: ₹165.00
Cart Details:
- Aashirvaad Atta (3) - ₹165.00

Sale #2 9/9/2025, 11:46:58 AM
Total Sale Value: ₹50.00
Cart Details:
- Good Day Biscuit - Pista Badam (5) - ₹50.00

Sale #3 9/9/2025, 11:47:18 AM
Total Sale Value: ₹40.00
Cart Details:
- Moms Magic Biscuit - Fruit & Milk (2) - ₹40.00

Sale #4 9/16/2025, 11:28:11 AM
Total Sale Value: ₹160.00



13. Known Issues

- Product data resets on page refresh (if no backend used)
- Limited reporting features

14. Future Enhancements

- Add authentication (login for store manager)
- Connect with backend + database (MongoDB / Firebase)
- Export stock reports to Excel/PDF
- Barcode scanning integration

3. Future Enhancements

Add shopping list generation from recipe ingredients
Introduce push notifications for new recipes
Offline mode for saved recipes