**Project Documentation**

**Store Manager:Keep track of Inventory**

1. Introduction

Project Title:Store Manager:Keep Track of Inventory

Team ID:NM2025TMID35195

Team Leader: Name:JEEVA PRIYAN S& Mail ID:sethujeevakarur33@gamil.com

Team Members:

Name:NEDUNCHEZHIYAN T

Mail ID: [chezhiyan395@gmail.com](mailto:chezhiyan395@gmail.com)

Name:ELAVAZHAGAN.G

Mail ID: [elavazhaganamp2007@gmail.com](mailto:elavazhaganamp2007@gmail.com)

Name:MURUGAN.M

Mail ID: [muruganmurugesan2007@gmail.com](mailto:muruganmurugesan2007@gmail.com)

2. Project Overview

PURPOSE:

* To record and monitor all product details such as quantity, price, and availability in the store.
* To automatically update inventory whenever sales or purchases occur.
* To alert the manager when stock is running low.
* To reduce manual errors and save time in record-keeping.
* To generate reports (sales, profit, stock status) that help the manager analyze business performance.

**Features:**

* Project Posting and Bidding: Clients can post new projects, and freelancers can replace composition
* Secure Chat System: Real-time communication between clients and freelancers with privacy and security.
* Feedback and Review System: Both clients and freelancers can leave feedback and reviews after project completion.
* Admin Control Panel: Provides administrators with tools to manage users, monitor activities, and oversee platform operations.
* Inventory Tracking (Store Manager Module): Admins can also manage and track store inventory, ensuring resources and transactions are properly maintained.

**3. Architecture**

**Frontend:**

* Built using React.js for dynamic UI rendering.
* Bootstrap and Material UI are used for responsive and modern design components.
* Provides interfaces for project posting, bidding, chat system, reviews, and inventory management.

**Backend:**

Node.js with Express.js handles server-side logic and API endpoints.

Manages authentication, project workflows, chat communication, and inventory operations.

**Database:**

MongoDB is used to store and manage data such as:

* + - User details (clients, freelancers, admin)
    - Project postings and bids
    - Chat messages
    - Feedback and reviews
    - Inventory records (products, stock levels, pricing, and tags) for store manager module

**4.Setup Instructions**

Prerequisites:

- Node.js (v16+)

-npm or yarn package manager

-Git hub&react.js

**Installation steps:**

# Clone repository git clone

https://github.com/your-username/store-manager.git

# Move into project folder

cd store-manager# Install dependencies

npm install

# Start development server

npm start

**5.Folder Structure**

store-manager/

│── public/

│── src/

│ ├── assets/ # Images, icons

│ ├── components/ # Reusable components (Navbar, SearchBar, ProductCard)

│ ├── context/ # InventoryContext + reducer

│ ├── pages/ # Page-level components (Dashboard, AddProduct, EditProduct, ProductList)

│ ├── utils/ # Helper functions (formatting, validations, custom hooks)

│ ├── App.js # Main app with routes

│ ├── index.js # Entry point

│── package.json

**6.Running the Application**

# Run the frontend

npm start

**7.API Documentation**

**User:**

* Register User  
  POST /api/user/register  
  *Registers a new user (store manager / staff).*
* Login User  
  POST /api/user/login  
  *Authenticates a user and returns a JWT token.*

**2. Product & Inventory APIs**

* Add New Product  
  POST /api/products/create  
  *Adds a new product with details like name, price, stock, category, and image URL****.***
* Get All Products  
  GET /api/products  
  *Fetches the list of all products in inventory.*
* Get Single Product by ID  
  GET /api/products/:id  
  *Fetches details of a single product.*
* Update Product  
  PUT /api/products/:id  
  *Updates product details (price, stock, image, etc.).*
* Delete Product  
  DELETE /api/products/:id  
  *Removes a product from inventory.*

**3. Stock Management APIs**

* UpdateStock  
  PUT /api/stock/update/:id  
  *Updates stock quantity (increase/decrease) for a product.*
* Low Stock Alerts  
  GET /api/stock/low  
  *Returns products with stock below threshold.*

4. Applications (Optional - Staff Requests)

* Apply for Store Staff Role  
  POST /api/apply  
  *Handles staff/store assistant applications.*

5. Chat APIs (For Manager & Staff Communication)

* Send Message  
  POST /api/chat/send  
  *Sends a message between store manager and staff.*
* Get Chat by User ID  
  GET /api/chat/:userId  
  *Fetches chat messages with a specific user.*

**8. Authentication**

**JWT Authentication**

All private routes (Products, Stock, Chat) are protected using JWT tokens.

Middleware checks for Authorization: Bearer <token> before allowing access.

**9.User Interface**

* Simple and responsive UI.
* Screens include:
* Home page with summary
* Cart page
* Sales
* Add product form

**10.Testing**

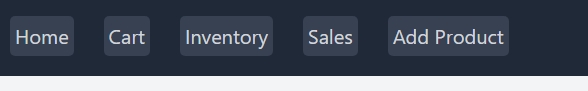
**Manual Testing**

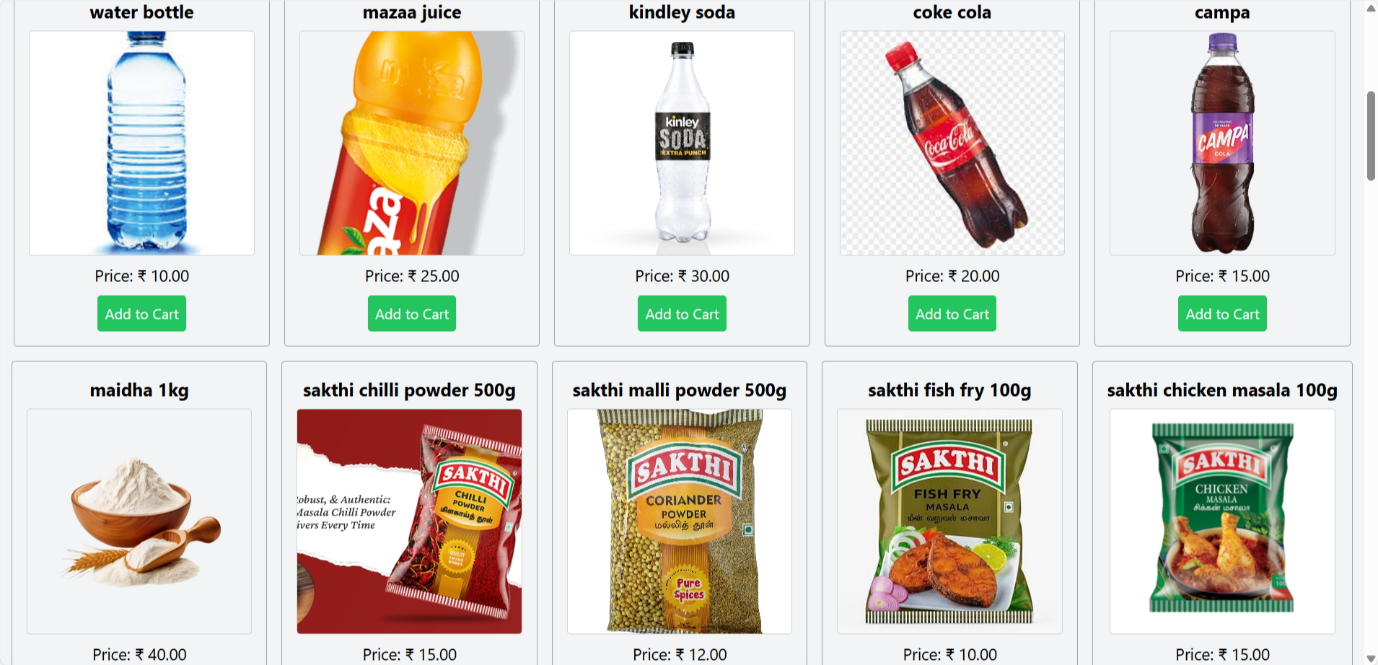
* Unit Tests – For reducers and utility functions.
* Component Tests – Using React Testing Library.
* Integration Tests – To ensure pages load and data flows correctly.

**Tools**

* Tested reducers for add, update, delete actions.
* UI component snapshot testing**.**

**11.Screenshots or demo**

****

****

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**12.Known Issues**

* Search filter does not work with special characters.
* No authentication system (anyone can access inventory).

**13.Future Enhancements**

→ Add authentication (Admin login).

→Export inventory data to Excel/PDF.

→Low-stock notifications.

→Dark mode / custom themes.

→Barcode scanner integration.