**Project Documentation**

**Store Management Inventory**

**1. INTRODUCTION:**

• Store Management Inventory

• Team ID : NM2025TMID39466

• Team Leader : Raja.M & megarajaraja10@gmail.com

• Team Members

– Rajesh.G & rajesh08032007@gmail.Com

– Neethiarasan.M & neethiarasan86@gmail.com

– Mohamed Uvaise.S & mohameduvaise462@gmail.com

**2. Project Overview:**

• **Purpose**:

The main purpose of stack management inventory is to track, organize, and

optimize stock or resources stored in a warehouse, store, or system.

• **Features**:

\* Accurate tracking

\* Prevention of stockouts & overstocking

\* Better space utilization in storage areas

\* Improved decision-making

\* Cost reduction

\* Faster accessibility

**3. Architecture:**

**Frontend**: React.js with Bootstrap and Material UI for building an

intuitive and responsive user interface for managing store inventory and operations.

**Backend**: Node.js and Express.js handle server logic, API endpoints, and business rules

and business rules related to inventory tracking, order processing, and stock management.

**Database**: MongoDB stores product details, stock levels, supplier

information, sales records, and customer data to support efficient store management

**4. Setup Instructions:**

• Prerequisites:

\* Node.js

\* MongoDB

\* Git

\* React.js

\* Express.js

\* Mongoose

\* Visual Studio Code

**• Installation Steps:**

bash

# Clone the repository

git clone <repo-url>

# Install client dependencies

cd client

npm install

# Install server dependencies

cd ../server

npm install

**• Folder Structure**

Store-Management/

|-- client/ # React frontend

| |-- components/

| |-- pages/

|-- server/ # Node.js backend

| |-- routes/

| |-- models/

| |-- controllers/

**• Running the Application**

Frontend:

bash

cd client

npm start

Backend:

bash

cd server

npm start

• **Access**: Visit <http://localhost:3000>

• **API Documentation**

Users:

`POST /api/user/register` — Register a user

`POST /api/user/login` — User login

• **Products / Inventory:**

`POST /api/products/create` — Add new product

`GET /api/products/:id` — Get product details

• **Stock Management:**

`POST /api/stock/update` — Update stock levels

• **Orders:**

`POST /api/orders/create` — Create an order

`GET /api/orders/:id` — Get order details

• **Notifications:**

`POST /api/notifications/send` — Send notification

`GET /api/notifications/:userId` — Get user notifications

**8. Authentication**

\* JWT-based authentication for secure user login

\* Middleware to protect private routes like inventory management and order processing

**9. User Interface**

\* Landing Page

\* Store Manager Dashboard

\* Admin Panel

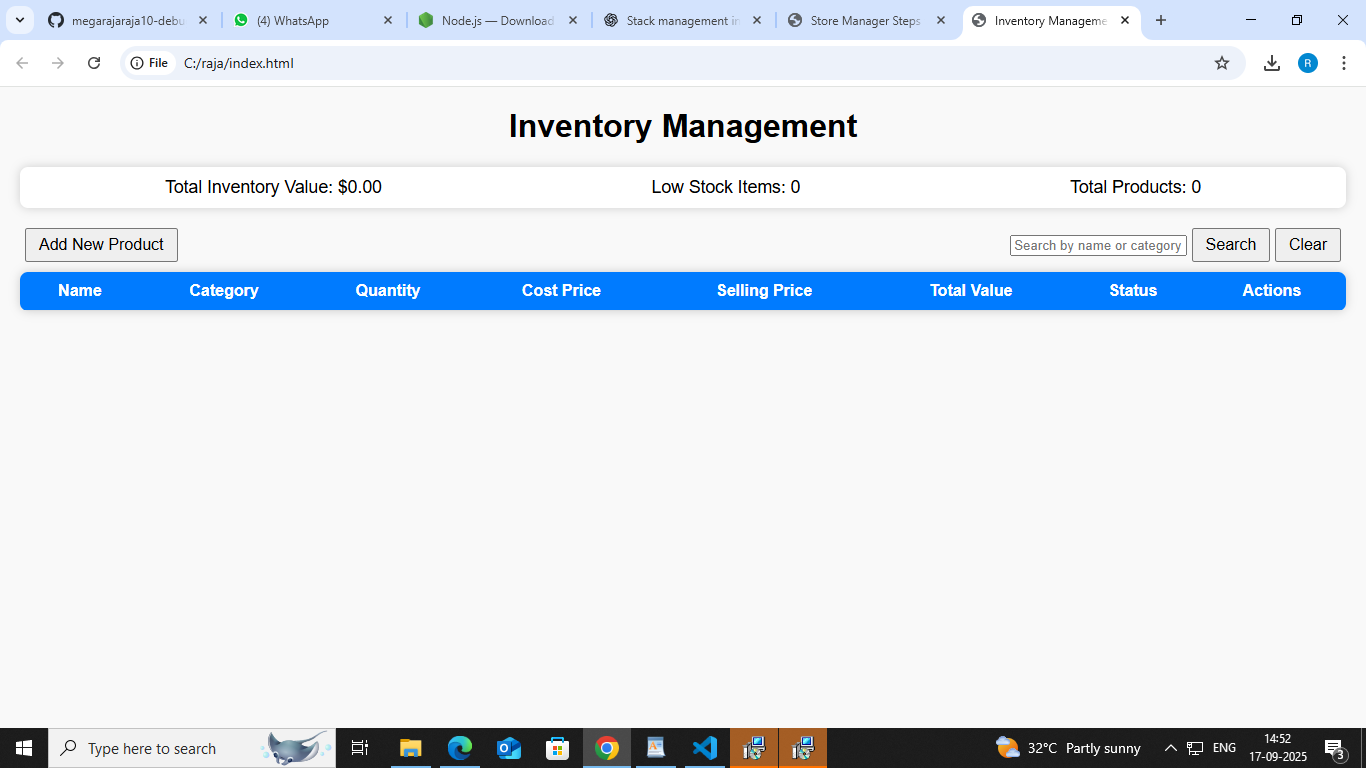
\* Product Details Page

**10. Testing**

\* Manual testing during development phases

\* Tools: Postman, Chrome Dev Tools

**11. Screenshots or Demo**

****

**12. Known Issues**

\* Occasional delays in real-time stock updates

\* Limited support for bulk import/export of inventory data

\* Minor UI glitches on mobile devices

\* Some edge cases in order processing under high load

**13. Future Enhancements**

\* Implement real-time notifications for low stock alerts

\* Add bulk import/export functionality for products and orders

\* Develop mobile-friendly responsive design

\* Integrate advanced analytics and reporting dashboards

\* Support multi-warehouse inventory management