**Project Documentation**

**Store Manager Keep Track of Inventory**

1. **Introduction**

**Team ID :** NM2025TMID33655

**Team Size :** 4

**Team Leader :** SRINATH.K(joshua37302@gmail.com)

**Team member :** Sarveswaran. R(sarveswaranSanjay@gmail.com)

**Team member :** Srithar(srithar.41656@gmail.com)

**Team member :** B.Tholkapiyan (nithisgowtham20@gmail.com)

2. Project Overview

Project Name:

Description:

Target Audience:

Key Features:

---

3. Architecture

Explain the overall architecture:

Frontend framework: React.js

State management: (Redux / Context API / Zustand, etc.)

API handling: (Axios / Fetch API)

Routing: (React Router)

Build tools: (Vite / Webpack / Create React App)

Include an architecture diagram if possible.

---

4. Setup Instructions

Step-by-step guide for installation:

# Clone the repository

git clone <repo-url>

# Navigate into project

cd project-folder

# Install dependencies

npm install

# Run development server

npm start

---

5. Store Manager: Keep Track of Inventory

(Special feature section — explain in detail)

Purpose of the Store Manager

Core functionalities (Add items, update stock, delete items, view reports)

Tech stack used

---

6. Folder Structure

Provide a standard folder structure:

project-root/

│── public/

│── src/

│ ├── assets/ # Images, fonts, icons

│ ├── components/ # Reusable components

│ ├── pages/ # Page-level components

│ ├── store/ # Redux/Context store

│ ├── hooks/ # Custom React hooks

│ ├── services/ # API calls

│ ├── styles/ # CSS/Tailwind/SCSS

│ ├── App.js

│ └── index.js

│── package.json

│── README.md

---

7. Running the Application

Explain commands for running:

Development: npm start

Build: npm run build

Test: npm test

---

8. Component Documentation

Document each major component:

Header.js → Navigation bar

InventoryList.js → Displays all items

ItemForm.js → Add/Edit items

Dashboard.js → Overview of stock

---

9. State Management

Explain chosen method:

Redux / Context API / Zustand

Store structure

Example reducer or context snippet

---

10. User Interface (UI)

UI design principles followed

Screenshots / wireframes

Responsiveness (mobile, tablet, desktop)

---

11. Styling

Framework (CSS, Tailwind, SCSS, MUI, Chakra)

Naming conventions (BEM, utility-first, etc.)

---

12. Testing

Testing framework: (Jest, React Testing Library, Cypress)

Example test case

test('renders inventory list', () => {

render(<InventoryList />);

expect(screen.getByText(/Item Name/i)).toBeInTheDocument();

});

---

13. Screenshots or Demo

https://drive.google.com/file/d/1\_FsN0RtteqM1vwA5cN7oRNLjf059krKk/view?usp=drivesdk

---

14. Known Issues

List bugs or limitations

---

15. Future Enhancements

Add authentication

Role-based access for store staff

Analytics & reporting dashboard

Offline support with PWA