Store Manager – Inventory Management

1. Introduction

Project Title: Store Manager – Inventory Management Frontend

Team Members: [Tharun kumar.L , Mohammed Usman , Vasantha Ruban.K , Vignesh.R ]

2. Project Overview

Purpose:

\* To enable users to track and manage inventory efficiently through a simple React-based frontend.

Features:

\*Real-time inventory tracking

\*Intuitive user interface

\*Easy setup and deployment

3. Architecture

Component Structure:

\*Main dashboard component for inventory overview

\*Form components for adding/updating items

\*List/display components for inventory items

State Management:

Uses React built-in state (useState, useEffect) and can be extended with Context API.

Routing:

Utilizes React Router for page navigation between dashboard and item management views.

4. Setup Instructions

Prerequisites:

\*Node.js LTS version (download from https://nodejs.org/en/download/)

\*PowerShell (Windows) or Terminal (macOS/Linux)

Installation:

\*Install Node.js using the correct installer for your operating system.

\*On Windows, open PowerShell as Administrator and run

Text:

\*set-executionPolicy unrestricted

\*Confirm with 'Y'.

\*Download the project code folder from the provided link.

\*Extract the ZIP archive to your preferred location.

\*Open Visual Studio Code and open the extracted code folder.

Run the following to install dependencies:

\*Text

\*npm install

Start the development server:

\*Text

\*npm start

5. Folder Structure

Client:

\*/public – static files

\*/src – source components, logic, styling

\*/components – UI components

\*/pages – page-level views

other files:

package.json, .gitignore, README.md, tailwind.config.js

Utilities:

[Describe any helper functions, classes, or custom hooks used]

6. Running the Application

Open a terminal in the project root.

Run:

\*Text

\*npm start

\*Access the application in your browser at http://localhost:3000.

7. Component Documentation

Key Components:

\*Dashboard: Displays inventory summary

\*InventoryForm: Add/edit item details

\*ItemList: Shows all inventory items

\*Reusable Components: [Describe any reusable forms, buttons, or modals]

8. State Management

\*Global State: Currently managed at the App level. Can be extended for complex flows.

\*Local State: Used within Dashboard, forms, and lists for user actions and item updates.

9. User Interface

[Provide screenshots of dashboard, item form, and item list—add images or demo links if available]

10. Styling

\*CSS Frameworks/Libraries: Tailwind CSS for rapid UI development.

\*Theming: [Describe custom themes or design tokens, if any]

11. Testing

\*Testing Strategy: [Example: Use Jest for unit tests, React Testing Library for integration]

\*Code Coverage: [List tools used for measuring test coverage]

12. Screenshots or Demo

[Insert screenshots or a demo video link]

13. Known Issues

[Document any setup hurdles, npm vulnerabilities, potential browser compatibility problems]

14. Future Enhancements

\*Add role-based access

\*Implement advanced analytics and export options

\*Enhance UI with animations

\*Improve mobile responsiveness

This format matches the professional structure of React project documentation while retaining all key details from the original instructions.