Project Documentation

1. Project Title

Project Title: Inventory and Sales Management System

Team ID: NM2025TMID38756

Team Leader: SUMESH S — mgc7sumesh24@gmail.com

Team Members:

• DIVAKAR S — mgc7divakar24@gmail.com
• SARAVANAN V — mgc7saravanan24@gmail.com
• VEERABATHRAN J — mgc7veera24@gmail.com

2. Project Overview

Purpose: A web-based application to manage products, inventory, sales, and cart operations for an online store.

Key Features:
• Add, list, and manage products
• Track sales and record transactions
• Maintain cart operations for user purchases
• Manage inventory stock levels
• Navigation bar for easy page access
• Persistent local storage for cart/session data

3. Architecture

• Frontend: React.js with Tailwind CSS
• State Management: React Context API
• Custom Hooks: for local storage persistence
• Backend (optional): Not included in this repo (can be added later)
• Database: Not included (currently frontend-only demo)

4. Setup Instructions

Prerequisites:
Node.js, Visual Studio Code, Git, npm/yarn

Installation:

```
# Clone the repository
git clone <repo-url>

# Move into client folder
cd client

# Install dependencies
npm install
```

```
# Start the development server
npm start
```

5. Folder Structure

```
client/
 ■■■ src/
 ■   ■■■ components/
 ■   ■   ■■■ Cart/
 ■   ■   ■   ■■■ Cart.jsx
 ■   ■   ■■■ Product/
 ■   ■   ■   ■■■ Product.jsx
 ■   ■   ■   ■■■ ProductItem.jsx
 ■   ■   ■   ■■■ AddProduct.jsx
 ■   ■   ■■■ Sales/
 ■   ■   ■   ■■■ Sales.jsx
 ■   ■   ■■■ NavBar.jsx
 ■   ■
 ■   ■■■ context/
 ■   ■   ■■■ CartContext.js
 ■   ■   ■■■ InventoryContext.js
 ■   ■   ■■■ SalesContext.js
 ■   ■
 ■   ■■■ hooks/
 ■   ■   ■■■ useLocalStorage.js
 ■   ■
 ■   ■■■ App.js
 ■   ■■■ index.js
 ■
 ■■■ public/
 ■■■ package.json
 ■■■ tailwind.config.js
 ■■■ README.md
```

6. Running the Application

```
cd client
npm start
```

Access it on: http://localhost:3000

7. State Management

• CartContext manages cart operations (add, remove, update)
• InventoryContext manages product stock and inventory updates
• SalesContext tracks sales records and totals

8. Custom Hook

• useLocalStorage.js: Handles persistent storage of state in localStorage.

9. User Interface Components

• NavBar.jsx: Main navigation header
• Cart.jsx: Displays selected products for checkout
• Product.jsx / ProductItem.jsx / AddProduct.jsx: Manage and display products
• Sales.jsx: View and record sales transactions

10. Testing

• Manual testing using Chrome DevTools
• Unit/component tests can be added using Jest and React Testing Library

11. Future Enhancements

• Connect to backend API with Express.js and MongoDB
• Add authentication and user roles
• Implement advanced reporting dashboards