Store manager keep track of inventory – Frontend Documentation

# 1. Project Title

Store manager keep track of inventory – Frontend Module  
  
Team ID: NM2025TMID40772  
Team Leader: DIVYASRI.T & divyasri22006@gmail.com  
Team Members:4  
- DHARSHINI.M & [dharshinimahendran06@gmail.com](mailto:dharshinimahendran06@gmail.com)

-FOUSIYA R.S & [rsfousiya@gmail.com](mailto:rsfousiya@gmail.com)

-DIVYA.G & [nalinidivya05@gmail.com](mailto:nalinidivya05@gmail.com)

# 2. Project Overview

Purpose:  
The Inventory Management System frontend provides an easy-to-use interface to manage products, track inventory, process sales, and manage cart operations.  
  
Frontend Features:  
- Product listing, adding, and updating  
- Sales tracking and history  
- Cart management (add/remove items)  
- Responsive navigation bar and layout  
- Real-time inventory state management using React Context API  
- Clean UI built with Tailwind CSS

# 3. Architecture (Frontend)

- Framework: React.js  
- Styling: Tailwind CSS  
- State Management: React Context API (Inventory, Sales, Cart)  
- Custom Hooks: useLocalStorage.js to persist data

# 4. Setup Instructions

Prerequisites:  
- Node.js (latest LTS)  
- npm (comes with Node.js)  
- Code editor (VS Code)  
  
Installation Steps:  
# Clone the repository  
git clone <repository-url>  
cd inventory-management-frontend  
  
# Install dependencies  
npm install  
  
# Start development server  
npm start  
  
Access:  
Open http://localhost:3000 in browser

# 5. Folder Structure (Frontend Only)

inventory-management-frontend/  
 ├── package.json  
 ├── tailwind.config.js  
 ├── src/  
 │ ├── index.js  
 │ ├── App.js  
 │ ├── Layout.jsx  
 │ ├── context/  
 │ │ ├── InventoryContext.js  
 │ │ ├── SalesContext.js  
 │ │ └── CartContext.js  
 │ ├── components/  
 │ │ ├── NavBar.jsx  
 │ │ ├── AddProduct.jsx  
 │ │ ├── Inventory/  
 │ │ │ ├── Inventory.jsx  
 │ │ │ └── Product.jsx  
 │ │ ├── Product/  
 │ │ │ ├── Product.jsx  
 │ │ │ ├── ProductCatalog.jsx  
 │ │ │ └── ProductList.jsx  
 │ │ ├── Sales/  
 │ │ │ ├── Sales.jsx  
 │ │ │ └── SaleRecord.jsx  
 │ │ └── Cart/  
 │ │ ├── Cart.jsx  
 │ │ └── CartItem.jsx  
 │ ├── pages/  
 │ │ └── NotFound.jsx  
 │ └── hooks/  
 │ └── useLocalStorage.js  
 └── public/  
 ├── index.html  
 ├── logo192.png  
 ├── logo512.png  
 ├── favicon.ico  
 ├── manifest.json  
 ├── robots.txt  
 ├── cartImage.png  
 └── inventoryLogo.png

# 6. Running the Application

- Start frontend:  
npm start  
  
- Default port: 3000  
- Open browser → http://localhost:3000

# 7. UI Components

- NavBar.jsx – Main navigation bar  
- Layout.jsx – Base layout wrapper  
- AddProduct.jsx – Add new product form  
- Inventory.jsx – Inventory list view  
- Product.jsx – Individual product card  
- ProductCatalog.jsx – Product catalog display  
- ProductList.jsx – Filtered product list  
- Sales.jsx – Sales dashboard  
- SaleRecord.jsx – Single sale record display  
- Cart.jsx – Shopping cart page  
- CartItem.jsx – Individual cart item

# 8. Context API

- InventoryContext.js – Stores inventory state and operations  
- SalesContext.js – Tracks sales and revenue data  
- CartContext.js – Manages cart state (add/remove/clear)

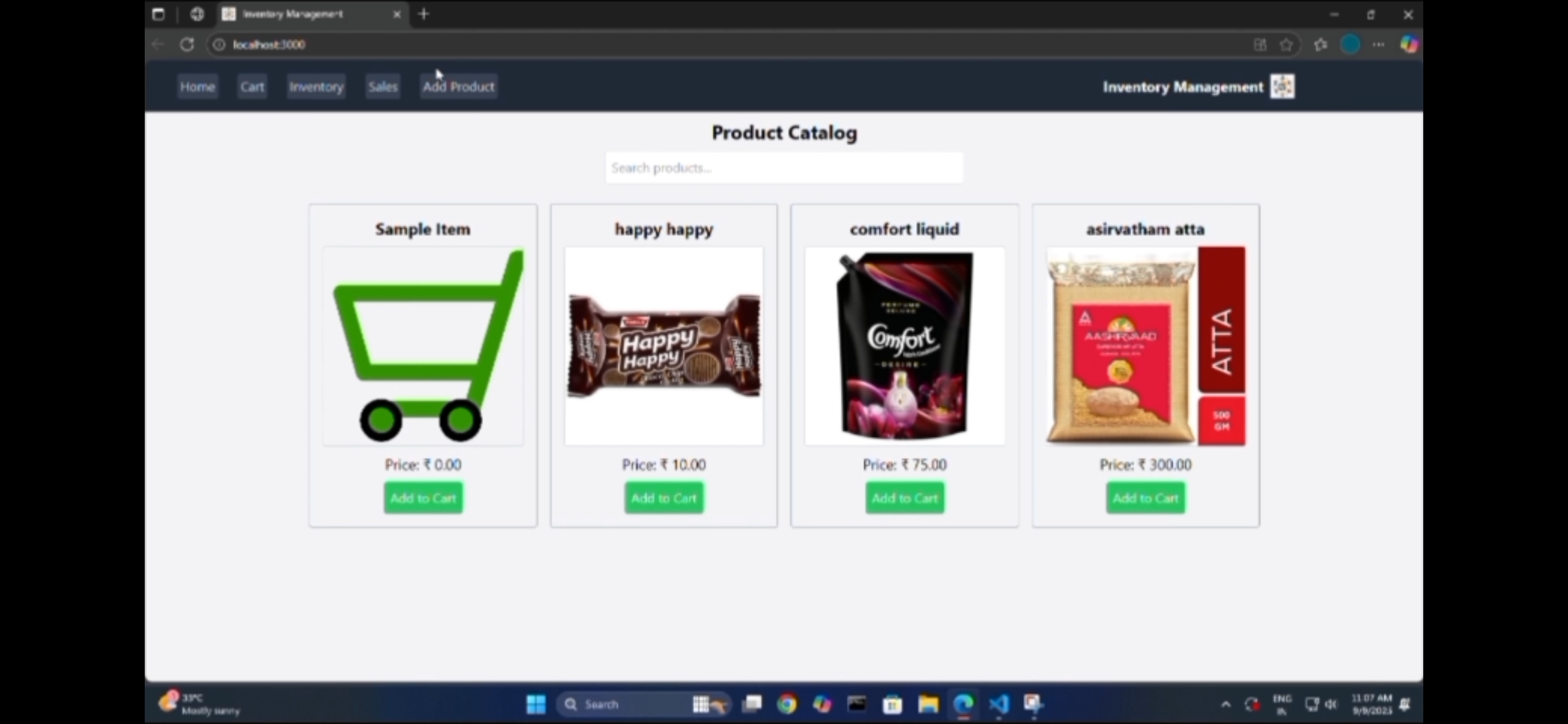
# 9. Hooks

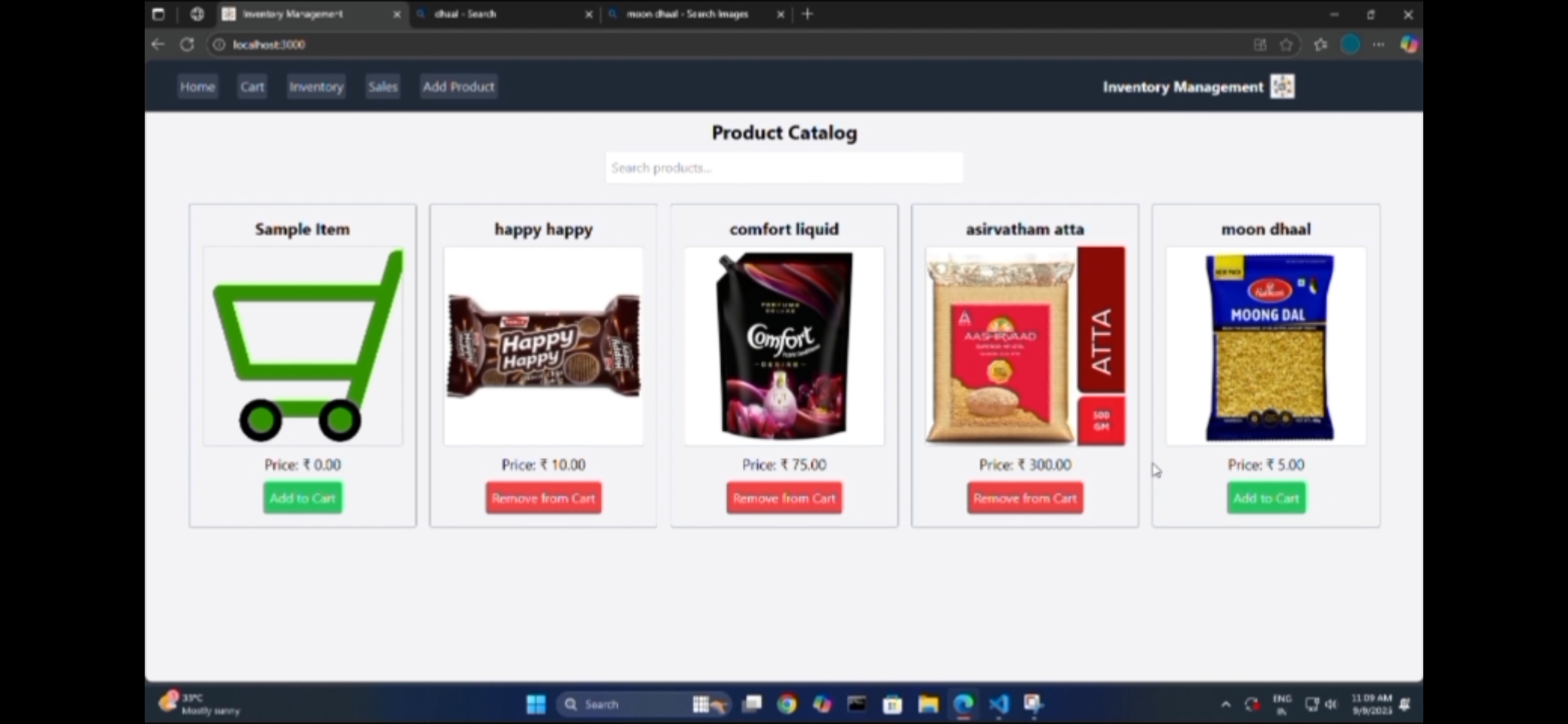
- useLocalStorage.js – Custom hook to persist state in browser local storage

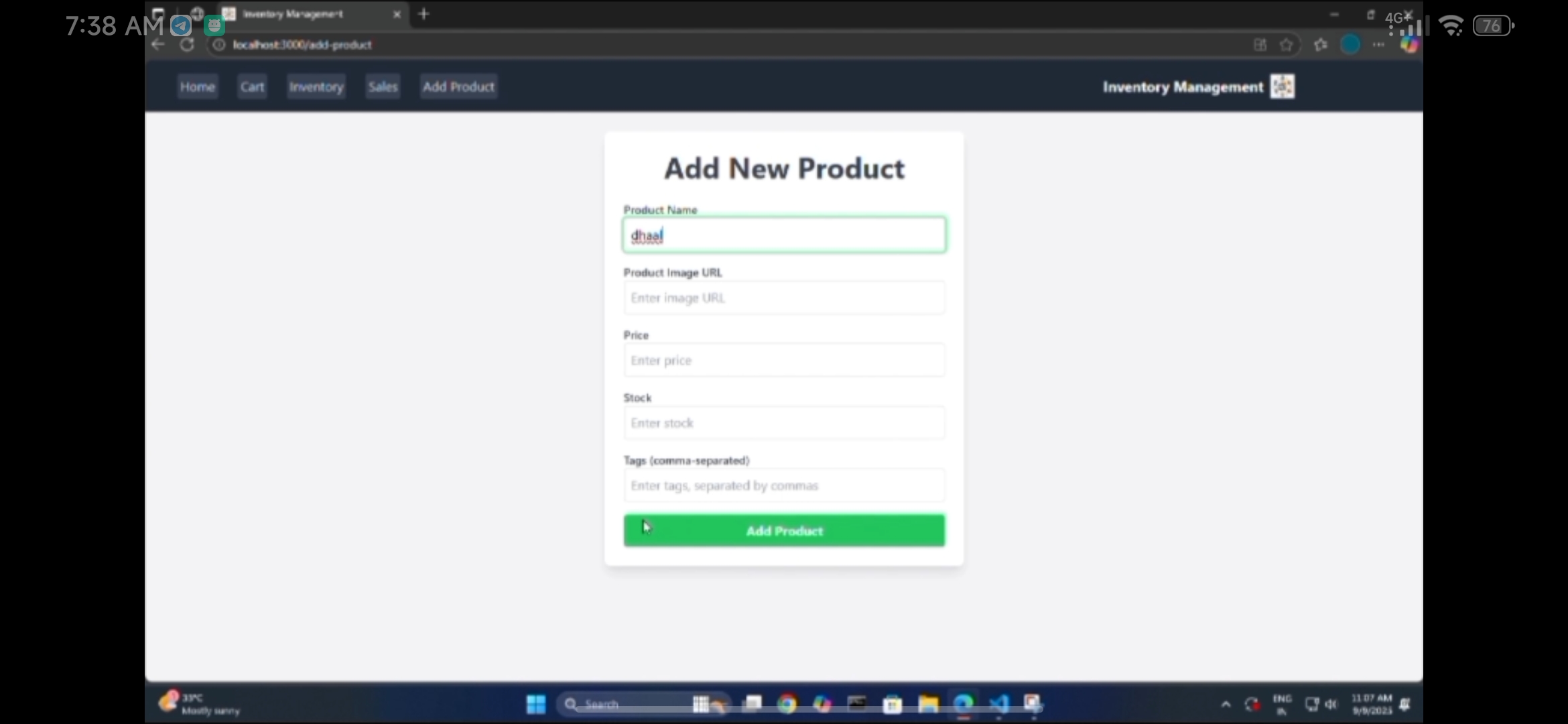
# 10. User Interface (Frontend Pages)

- Product management screen  
- Sales dashboard  
- Cart page with item list  
- Responsive navigation bar  
- NotFound.jsx for unmatched routes

# 11. Screenshots







# 12. Testing (Frontend)

- Manual testing using browser dev tools  
- Checked state updates across contexts  
- Verified UI responsiveness

# 13. Future Enhancements

- Add authentication & user roles  
- Integrate backend APIs for persistent storage  
- Generate reports and analytics  
- Add search and filter options for inventory