Frontend Development with React.js

Store Management Project Documentation

---

1. Introduction

Project Title: Store Management Web Application

Team Members:

Nasreen fathima N

Priyadharshini S

Chandrika k

Rakesh s

---

2. Project Overview

Purpose:

 To manage store inventory, display products, handle add/update/delete products, and allow simple user interaction in a clean frontend interface.

Features:

 • Add, update, delete store products

 • Display product list with name, price, quantity

 • Simple UI with responsive layout

---

3. Architecture

Component Structure:

 • Components: Header, ProductList, ProductForm, ProductItem

 • Interaction via props and local state

State Management:

 • React’s useState and useEffect

 • No external state library used

Routing:

 • Simple one-page app (no react-router)

---

4. Setup Instructions

Prerequisites:

 • Node.js installed

 • NPM installed

Installation:

 1. Clone repository:

  https://github.com/Deepak-1006S/store-management.git

 2. Navigate to project folder:

  cd store-management-app

 3. Install dependencies:

  npm install

 4. Start development server:

  npm start

---

5. Folder Structure

Client:

 • /src/components – Header, ProductList, ProductForm, ProductItem

 • /src/App.js – Main entry point

 • /public/index.html – HTML template

Utilities:

 • Helper functions in /src/utils

---

6. Running the Application

npm start

Runs the app in development mode. Open http://localhost:3000 to view in browser.

---

7. Component Documentation

Header: Displays project title.

ProductList: Shows all products as ProductItem components.

ProductForm: Form to add new products or edit existing ones.

ProductItem: Displays single product with name, price, quantity, edit/delete buttons.

---

8. State Management

Global State: None (Local component state only).

Local State:

 • products – Array of store product objects.

 • currentProduct – Object for editing product.

---

9. User Interface

📸 Screenshot Preview:

Features:

 • Product list display

 • Add product form

 • Edit/delete product actions

---

10. Styling

CSS used for responsive layout and simple design.

Flexbox used for grid layout of product cards.

---

11. Testing

Testing Strategy:

 Manual testing to verify add/update/delete functionality and UI responsiveness.

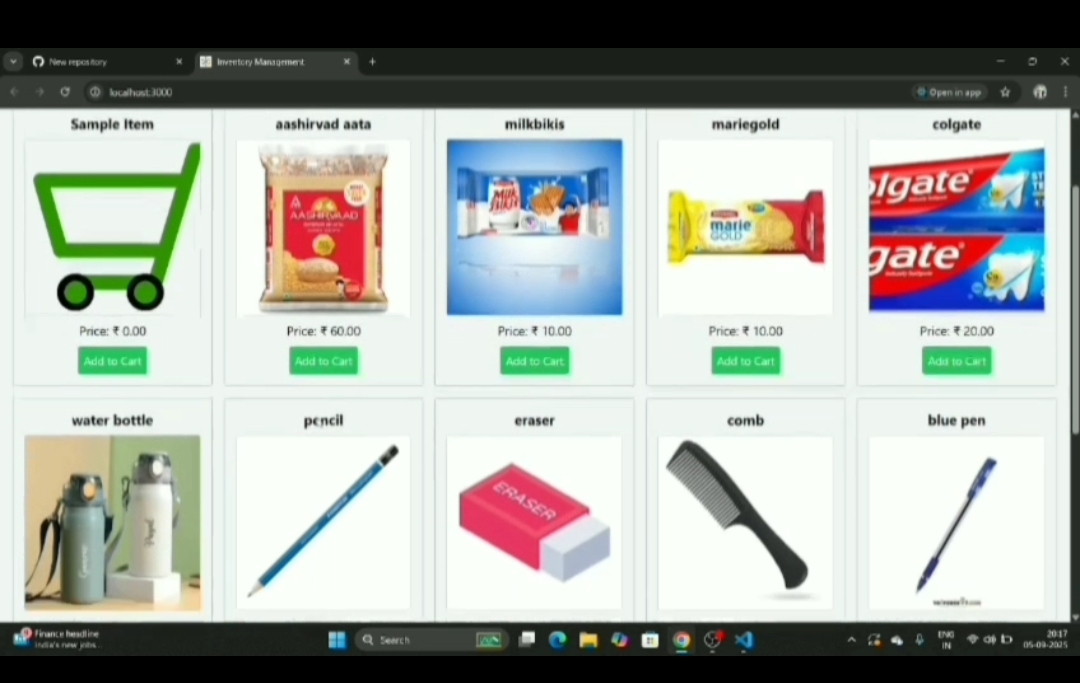
Code Coverage:

 No automated tests implemented.

---

12. Screenshots

📸 Screenshot:



---

13. Known Issues

Data not persistent (stored in local state only).

No pagination for product list.

---

14. Future Enhancements

Add backend support (Node.js + MongoDB) for persistent data.

Add product categories filter.

Implement dark mode.

Add unit tests using Jest.