Frontend Development with React.js

Project Documentation

Introduction

Project Title: Store Manager: Keep Track of Inventory

Team Leader: K.S. Muthukaran

Team Members:
- G. Hariharan
- A. Kalaiarasan
- P. Muruganantham
- R. Monoji

Project Overview

Purpose: The purpose of the Store Manager project is to help businesses easily keep track of their inventory. It allows users to add, update, and delete products, check stock levels, and generate reports for better decision-making.

Features:
- Add, update, and delete products
- Track stock levels
- Search and filter products
- Generate reports

Architecture

Component Structure: The project consists of major components like ProductList, ProductForm, Dashboard, and Reports. These components interact through props and shared state.

State Management: The application uses Context API for managing global state such as inventory data and user interactions.

Routing: React Router is used to manage navigation between different pages like Dashboard, Products, and Reports.

Setup Instructions

Prerequisites:
- Node.js
- npm or yarn

Installation:
1. Clone the repository
2. Run npm install to install dependencies
3. Configure environment variables if required
4. Run npm start to start the development server

Folder Structure

Client:
- components/: Contains reusable React components
- pages/: Holds main pages like Dashboard, Products, Reports
- assets/: Images, icons, and styles
- context/: Global state management

Utilities: Custom hooks and helper functions are included for API calls and state handling.

Running the Application

To run the frontend server locally:
- cd client
- npm start

Component Documentation

Key Components:
- ProductList: Displays the list of products
- ProductForm: Allows adding and updating products
- Dashboard: Shows overall inventory statistics
- Reports: Generates inventory reports

Reusable Components: Includes components like Button, Modal, and InputField for consistent UI design.

State Management

Global State: Managed with Context API to store inventory and user data.

Local State: Handled within components using useState for UI-specific data like form inputs and modal visibility.

User Interface

The UI includes a clean dashboard, product listing with filters, and interactive forms for product management.

Styling

CSS Frameworks/Libraries: Tailwind CSS is used for styling with responsive design principles.

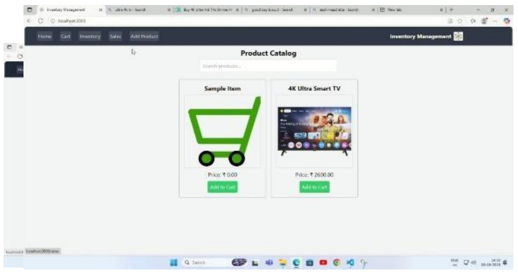
Theming: A consistent theme with custom colors and typography is applied across components.

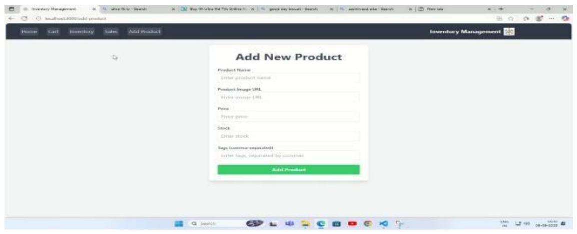
Testing

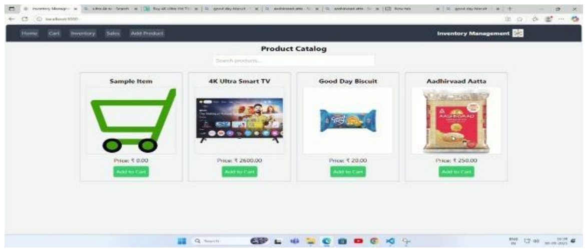
Testing Strategy: Jest and React Testing Library are used for unit and integration testing.

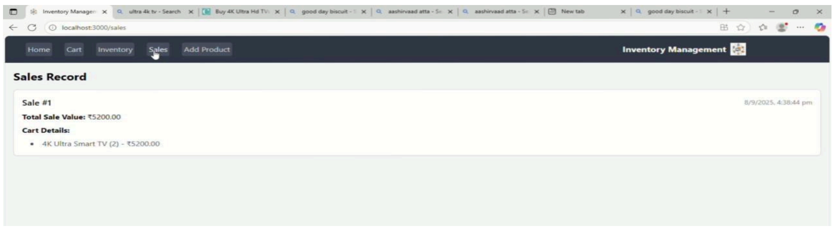
Code Coverage: Coverage reports ensure adequate testing of key features.

Screenshots or Demo









Sales Record

Sale #1 8/9/2025, 4:38:44 pm

Total Sale Value: ₹5200.00

Cart Details:

- 4K Ultra Smart TV (2) - ₹5200.00

Demo video link: https://drive.google.com/file/d/1GKmGyXQ0Huwce5f8z-nlyE1eOeASPhM3/view?usp=drivesdk

Known Issues

Currently no critical issues. Minor UI improvements may be required for better responsiveness.

Future Enhancements

Planned improvements:
- Add authentication and user roles
- Implement barcode scanning
- Export reports in multiple formats
- Integrate with backend APIs for real-time updates