# Frontend Development with React.js

Project Documentation for Store Manager – Inventory Tracking System

## Introduction

Project Title: Store Manager – Inventory Tracking System

Team Members: [List team members and their roles]

## Project Overview

Purpose: The Store Manager app allows businesses to efficiently track and manage their inventory. It provides features to add, edit, delete, and monitor products, track stock levels, receive low-stock alerts, and maintain sales history.

Features:

• Add, edit, and delete products  
• Track stock levels and sales  
• Search and filter inventory  
• Receive low-stock alerts  
• Generate reports on sales and inventory trends  
• User authentication and role-based access (optional)

## Architecture

Component Structure: The application is divided into reusable React components including `ProductList`, `ProductForm`, `InventoryDashboard`, and `StockAlerts`. Components are organized into pages such as Dashboard, Inventory, Reports, and Settings.

State Management: Redux (or Context API) is used to manage the global state of inventory and sales data.

Routing: React Router is used to navigate between pages like Dashboard, Inventory, Reports, and Settings.

## Setup Instructions

Prerequisites: Node.js, npm/yarn installed.  
Installation:  
1. Clone the repository  
2. Run `npm install` to install dependencies  
3. Configure environment variables (if needed)  
4. Start the development server with `npm start`

## Folder Structure

Client: The React application is organized into folders:  
- `components/`: Contains reusable UI components (ProductCard, StockAlert, etc.)  
- `pages/`: Contains major pages (Dashboard, Inventory, Reports, Settings)  
- `assets/`: Images, icons, styles  
- `redux/` or `context/`: State management setup  
- `utils/`: Helper functions and custom hooks

## Running the Application

Run the following command to start the application locally:  
`npm start`

## Component Documentation

Key Components:  
- `ProductList`: Displays all products with stock levels  
- `ProductForm`: Add or edit product details  
- `InventoryDashboard`: Overview of stock and alerts  
- `StockAlerts`: Displays low-stock notifications  
- `Reports`: Shows sales and inventory reports

## State Management

Global State: Managed with Redux/Context API to keep track of inventory and sales.  
Local State: Component-level state is used for form handling and UI interactions.

## User Interface

The UI includes:  
- Product tables with pagination and search  
- Forms for adding/editing products  
- Low-stock alerts  
- Graphs and charts for reports

## Styling

CSS Frameworks/Libraries: Tailwind CSS or Material-UI is used for styling.  
Theming: Supports light/dark themes and custom branding.

## Testing

Testing Strategy: Jest and React Testing Library are used for unit and integration testing.  
Code Coverage: Coverage reports are generated to ensure reliability.

## Screenshots or Demo

Provide screenshots of the dashboard, product management pages, and reports.  
Optionally, include a demo link.

## Known Issues

Document any existing bugs, such as incomplete report filtering or limited mobile responsiveness.

## Future Enhancements

• Barcode scanning support  
• Export inventory data to CSV/Excel  
• Multi-store inventory management  
• Integration with third-party POS systems  
• Push notifications for low-stock alerts