

INVENTORY MANAGEMENT

Store Manager: keep track of Inventory

Frontend Development with React.js



Introduction to the Project

Welcome to the documentation for the Inventory Management web application's frontend. This guide provides a comprehensive overview on inventory management, covering the project's purpose, architecture, development setup, and key features. Our goal is to ensure a seamless and efficient development experience.

1

TEAM ID :NM2025TMID39541

TEAM SIZE : 3

2

TEAM LEADER

MEENAKSHI

3

TEAM MEMBER

VICTORIYA J

4

TEAM MEMBER

NASRIN

VICT@KCS

INVENTORY MANAGEMENT

Project Overview & Goals

The Inventory Management application is designed to streamline stock operations, providing organizations with a powerful tool to track, manage, and update product inventories efficiently.

Purpose

- A comprehensive inventory management system with integrated e-commerce functionality for small businesses to manage products, track sales, and handle customer orders efficiently.

Key Features

- Product catalog with category filtering
- Inventory management with stock tracking
- Shopping cart functionality
- Sales record tracking
- Responsive design with modern UI
- Local storage persistence



Frontend Architecture: Component Structure

The frontend is built with a modular and scalable architecture, primarily using React.js. This structure facilitates maintainability and allows for independent development of features.



INVENTORY MANAGEMENT

Development Setup: Get Started

To begin contributing to the Inventory Management frontend, follow these simple setup instructions. Ensure you have the necessary prerequisites installed.

Prerequisites

- **Node.js:** Version 18+ is required.
- **npm** : Package managers for dependencies.
- **Git:** Version control system.
- **Editor:** Visual studio code

Interaction Flow: Users interact with individual pages, which in turn utilize reusable components. Data and state updates are managed through Context/Redux, ensuring a reactive and dynamic UI.

Installation Steps

Download Node.js LTS version from <https://nodejs.org/en/download/>

After installing Node.js, open Windows PowerShell As Administrator

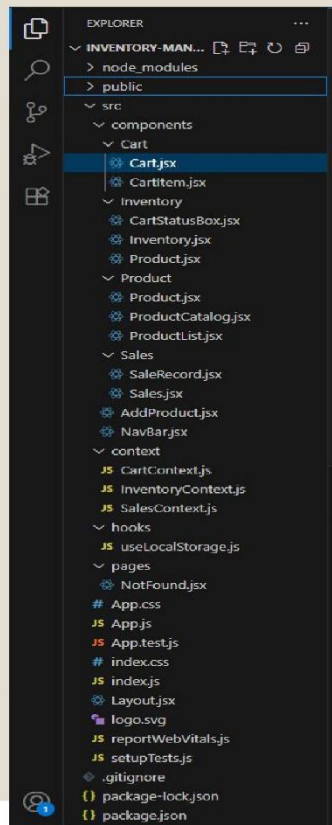
Type set-executionPolicy unrestricted and press enter type Y and press enter

Now right click on the downloaded zip folder and click on extract all

Now open VS code and open the code folder from the extracted folder

Type npm install in your terminal and press Enter and wait till all the dependencies gets download and then type npm start and press Enter

Folder Structure



INSTALLING THE APPLICATION

Install dependencies

npm install

RUNNING THE APPLICATION

Start frontend development server


npm start

Application runs on http://localhost:3000


INVENTORY MANAGEMENT

Core Features at a Glance


The application is built with a comprehensive set of features to address key inventory management needs. These features are accessible through a responsive and user-friendly interface.

**Dashboard Overview**


A centralized view of inventory metrics and summaries.

**Product Management**


Intuitive tools to add, edit, and delete product entries.

**Real-time Stock Updates**


Accurate and immediate updates on inventory levels.

**Product Categorization**

Organize products into logical categories for easy navigation.

**Search & Filter**

Robust functionality to quickly find specific products.

**Responsive UI**

Optimized user experience across all devices and screen sizes.

State Management & UI Features

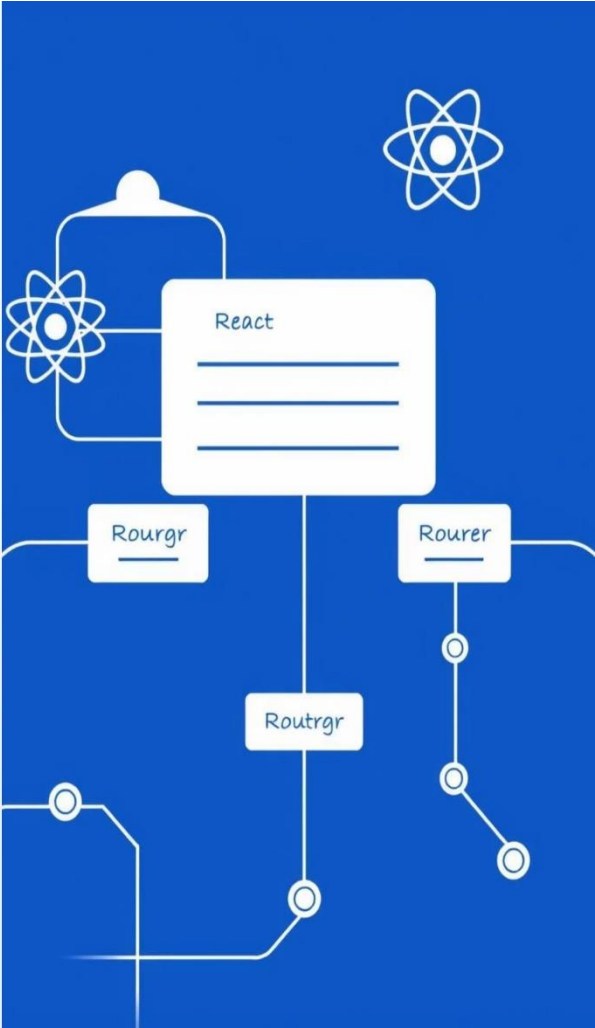
Effective state management and intuitive routing are crucial for a smooth user experience and efficient data flow within the application.

State Management

- **Global State:**
- **CartContext:** Manages shopping cart state
- **InventoryContext:** Handles product inventory state
- **SalesContext:** Manages sales records
- **Local State:** useState hook for component-level state management

UI Features:

- Responsive product grid layout
- Modern form designs with validation
- Real-time cart updates
- Sales dashboard with transaction history
- Inventory management interface



INVENTORY MANAGEMENT

Styling & Testing Strategy

Consistency in styling and rigorous testing are integral to the quality and user experience of our application.

Styling

CSS Framework: Tailwind CSS for utility-first styling

Custom Styling:
Custom gradient backgrounds
Hover effects and transitions
Responsive design patterns
Custom color scheme (pink/purple theme)

Theming: Consistent color palette and typography throughout

Testing Strategy

Manual testing of all user flows

Component functionality testing

Cross-browser compatibility testing

Tools: React Developer Tools for state debugging



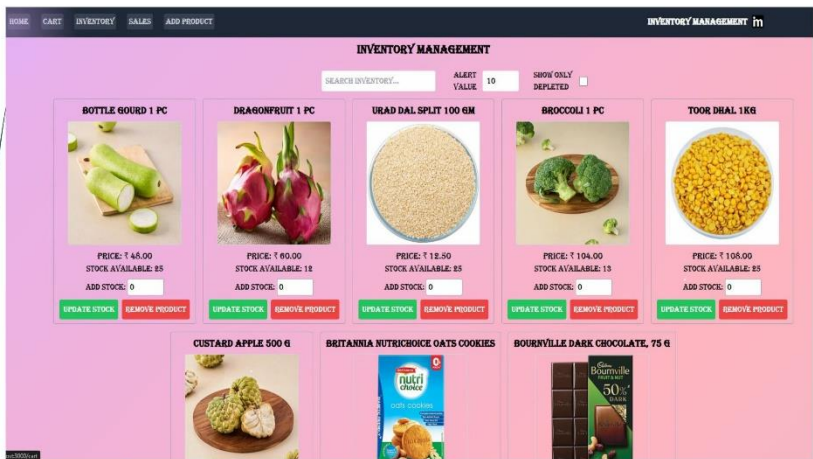
Key Components & Utilities

The application utilizes several key components and utility functions to ensure reusability, maintainability, and efficient development.

<p>ProductTable</p> <p>Displays products with sorting and filtering. Props: products, onEdit, onDelete.</p>	<p>ProductForm</p> <p>Form for creating/editing products. Props: initialValues, onSubmit.</p>
<p>DashboardCard</p> <p>Shows inventory metrics. Props: title, value, icon.</p>	<p>Reusable UI</p> <p>Button, Input, Modal, Loader are used extensively.</p>
<p>Custom Hooks</p> <p>useFetch.js for simplified API calls.</p>	<p>Utility Functions</p> <p>formatDate.js for date handling and validators.js for form validation.</p>

INVENTORY MANAGEMENT

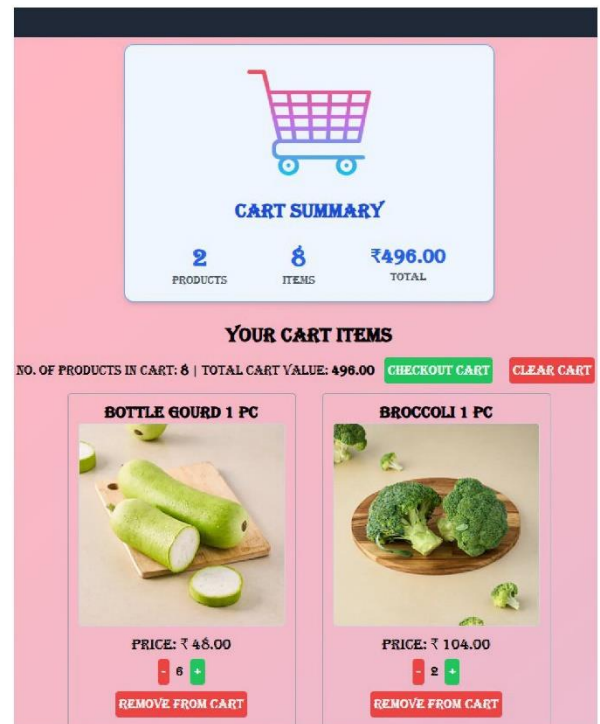
INVENTORY



TO ADD NEW PRODUCTS

A screenshot of the 'ADD NEW PRODUCT' form. It has a pink background and a dark blue header. The form contains several input fields: 'PRODUCT NAME', 'ENTERED PRODUCT NAME', 'PRODUCT IMAGE URL', 'ENTERED IMAGE URL', 'PRICE', 'ENTERED PRICE', 'STOCK', and 'ENTERED STOCK'. There is also a section for 'TABS (TABNA SEPARATED)' with a note 'ENTER TABS SEPARATED BY COMMAS'. At the bottom, there is a pink button labeled 'ADD PRODUCT'.

CART PAGE



Future Enhancements & Known Issues

 We continuously strive to improve the application. Here are some areas for future development and current known issues.

Known Issues

- Image loading issues for missing product images
- Limited form validation in some components
- No user authentication system
- Basic error handling implementation.

Future Enhancements

- Planned Features:**
- User authentication and authorization
 - Payment gateway integration
 - Order management system
 - Email notifications
 - Advanced reporting and analytics
 - Mobile app version
 - Multi-language support
 - Advanced search and filtering
 - product reviews and ratings
 - Inventory low stock alerts
 - Bulk operations for inventory management
 - Data export functionality (CSV, PDF)
 - Dark mode toggle
 - PWA capabilities for offline functionality

INVENTORY MANAGEMENT