ShopSmart: Grocery Store Web App

Team ID: LTVIP2025TMID56355

Team Member: Manjusha Bezawada

Solution Requirements

1. Functional Requirements

These are the core features your app must support.

User Interaction:

- User should **view products** with image, name, price, and model.
- User must be able to **register** with name, mobile number, place, and password.
- **Login/Logout** functionality must be available.
- Users can **add products to cart**.
- Users can **buy products directly** via a “Buy Now” button.
- Users can **view the cart** with total price calculation.
- Users must be able to **place an order**.
- Users should be able to **view all their past orders**.

Authentication:

- User authentication must be secure using basic password protection.
- No access to orders/cart without login.

2. Non-Functional Requirements

These define how the system behaves under certain conditions.

Performance:

- The app should **load products quickly** on the homepage.
- Orders and cart actions should be processed in **real-time** or with minimal delay.

User Interface:

- Clean, **colourful, and elegant UI** using ReactJS + Bootstrap.
- Intuitive navigation to Home, Cart, Orders, and Login/Signup.

Data Storage:

MongoDB must be used to **store user data**, cart items, and order records

Customer Journey Mapping

This **Customer Journey Map** outlines the four key stages a user goes through while using the Grocery Store Web App: Signup, Browsing, Cart, and Orders.

🔄 **Stages and Insights:**

1. **Signup**
    - The user registers using a mobile number, name, place, and password.
    - **Goal:** Quick onboarding to get users into the system easily.
    - **Opportunity:** Keep the registration form minimal for faster sign-up.
2. **Browsing**
    - Users explore available grocery items and either add them to the cart or use the 'Buy Now' option.
    - **Goal:** Offer a seamless browsing experience with clear visuals.
    - **Opportunity:** Organize products clearly and enable quick decisions.
3. **Cart**
    - Users review selected items, see the total price, and can place their order.
    - **Goal:** Transparent pricing and easy checkout.
    - **Opportunity:** Clear call-to-action buttons improve the purchase flow.
4. **Orders**
    - Users access their order history and check the status of previous purchases.
    - **Goal:** Provide clarity and confidence in the shopping process.
    - **Opportunity:** Enhance user trust by showing confirmed orders neatly.

This journey map helps in identifying not just user actions, but also design and development decisions that support a better experience.

Signup	Browsing	Cart	Orders
			
Registers using mobile number, name, place an password 	Looks through avaiilale grocery items, adds to cr or buys now 	Vlews selected items in cart with total price, places order 	Sees all ordered grocery items in the orders page 
Opportunities	Optimize product form for simplicity and speed	Provide clear call-to-action buttons	Enable easy order tracking and status

Technology Stack

Frontend:

- **ReactJS** – For building the user interface as a single-page application (SPA)
- **HTML5 & CSS3** – For structuring and styling components
- **Bootstrap** – For responsive and elegant UI components
- **React Router** – For client-side routing and navigation

Backend:

- **Node.js** – JavaScript runtime environment for server-side logic
- **Express.js** – Web framework for handling routes and API requests

Database:

- **MongoDB** – NoSQL database to store user data, cart items, and orders
- **Mongoose** – ODM library to interact with MongoDB easily from Node.js

Authentication:

- **Custom Auth Logic** – Users register/login using mobile number, name, place, and password
- **Session or State Passing via React Router** – User identity is passed between pages

Tools & Deployment (optional):

- **Git & GitHub** – For version control and repository management
- **MongoDB Community Server** – For local development and testing