Full Stack Development with MERN

Project Documentation

1. Introduction

Project Title: ShopMart: Your Digital Grocery Store Experience
Team ID: LTVIP2025TMID55453
Team Lead: Vanjarapu Suryavardhan
Team Members: V. Naga Venkata Pavan, V. Sai Viswanadh, Uppada Sagar

ShopMart is a full-stack e-commerce application built using the **MERN stack** (MongoDB, Express, React, Node.js) as part of the Smart Bridge initiative. The system is designed to provide a seamless shopping experience, enabling customers to browse, filter, and purchase grocery products, while also equipping administrators with powerful management tools. ShopMart blends responsive design, secure transactions, and real-time interactivity for an efficient, scalable marketplace.

2. Project Overview

Purpose:
The primary goal of ShopMart is to deliver a scalable, user-friendly grocery shopping solution for customers and robust management tools for administrators. The platform streamlines product discovery, simplifies order management, and supports modern business workflows for an evolving digital marketplace.

Features:

- Register new users and manage secure authentication
- Browse and filter groceries by categories, brands, price, or popularity
- Add items to cart, adjust quantities, and proceed to secure checkout
- Save delivery details, view order history, and track orders in real-time
- Admin dashboard to manage products, users, and orders
- Product management (add/update/remove, with categories, images, brands)
- Order status management and ability to update/cancel as required

3. Architecture

Frontend:

- Built with React.js for dynamic, responsive user interfaces
- State management through Redux Toolkit
- Customizable, modern UI using Tailwind CSS

User flows: Home, product listings, cart, checkout, profile, admin panel

Backend:

Node.js with Express.js for robust, scalable REST API development

API endpoints for products, users, authentication, cart, checkout, and admin tasks

Secure authentication (JWT), password hashing (bcrypt)

Database:

MongoDB Atlas as primary NoSQL database

Flexible schema for users, products, orders, and shopping carts

4. Setup Instructions

Prerequisites:

Node.js v16+

MongoDB Atlas account

Git

Code editor (VS Code recommended)

Installation Steps:

Clone the ShopMart repository from GitHub

Navigate to /client and /server folders

Install dependencies in each (npm install)

Update .env files with environment-specific variables

5. Folder Structure

Client (React Frontend):

src/assets – Static files

src/components – Shared UI components

src/config – App constants, API URLs

src/lib – API hooks, utilities

src/pages – Route-level components for user/admin src/store – Redux slices and state

App.jsx, main.jsx – Root components

Server (Node.js Backend):

controllers – Route logic

models – Database schemas (mongoose)

routes – API endpoints

helpers – Utility functions (including auth)

server.js – Application entrypoint

6. Running the Application

Frontend: In the /client directory, run

```
npm start
```

Backend: In the /server directory, run

```
npm start
```

7. API Documentation (Key Endpoints)

Endpoint	Method	Description
/api/auth/register	POST	Register a new user
/api/auth/login	POST	Log in and receive a JWT token
/api/products	GET	Get all products
/api/products/:id	GET	Get details of one product
/api/cart/add	POST	Add an item to user's cart
/api/cart/remove	DELETE	Remove item from cart
/api/checkout	POST	Process payment/order
/api/admin/users	GET	Admin: view all registered users

8. Authentication

JWT-based authentication: Users receive a token on login (stored in localStorage)

Auth-protected routes require Authorization: Bearer <token> header

Passwords saved securely as bcrypt hashes

Middleware enforces roles (admin/customer) and permissions throughout

9. User Interface Overview

Navbar – Dynamic for login/logout, cart preview

Homepage – Product grid, filters, offers

Search Bar – Keyword-based search

Cart – Add/remove/view/update cart items

Checkout – Enter shipping, review summary, pay securely

Admin Dashboard – User, order, and inventory management

10. Testing

Approaches:

Functional Testing: Manual user journey verification

API Testing: Automated with Postman collections

Performance Testing: Load scenarios using Apache JMeter

Accessibility & Speed: Audited using Lighthouse

11. Project Screenshots or Demo Links

[Demo Video/Screenshots]: https://drive.google.com/file/d/1JyzfGeiCo-fudjG-ay7qKcjt9d0YsTF3/view?usp=drive_link

12. Known Issues

Some UI components might not be fully responsive on legacy browsers

Voice/AI assistant (if any) may be limited on certain platforms

13. Future Enhancements

Native mobile application (React Native)

Live order tracking

Notifications via email/SMS

Coupon and promo code system

Advanced admin/seller roles

Sales analytics dashboard

AI-driven chatbot support