

Full Stack Development with MERN

Project Documentation

1. Introduction

- Project Title: ShopSmart: Your Digital Grocery Store Experience

Team Members:

Team Size: 3

Team Leader:

- **Padma Gadi** – Full Stack Development (Backend APIs, Database Integration, Project Integration & Deployment) & Frontend Support & Testing

Team Members:

- **Sai Joshika Ayyankala** – Documentation & Presentation Support
- **Jagarapu Hyma** – Frontend Development (UI Components & Design)

2. Project Overview

- **Purpose:**

The purpose of this project is to build a complete full-stack grocery e-commerce web application where users can browse products, add items to cart, place orders, and track payments.

The system also includes an admin dashboard to manage users, products, categories, and orders.

The main goal is to demonstrate MERN stack development skills in a real-world application.

- **Features:**

User Features:

- User Registration & Login
- Browse Products
- Filter by Category
- Add to Cart
- Place Orders
- View Order History

- Provide Feedback

  **Admin Features:**

- Admin Login
- Add Category
- Add Products
- Update/Delete Products
- View Users
- Manage Orders
- View Payments
- View Feedback

3. Architecture

• **Frontend: (Client – Angular/React)**

- Developed using Angular (as per your project)
- Uses component-based architecture
- Routing handled using Angular Router
- HTTP requests handled using Axios/HttpClient
- Bootstrap used for styling

Frontend communicates with backend APIs using REST APIs.

• **Backend: (Server – Node.js + Express.js)**

- Built using Node.js and Express.js
- REST API based architecture
- JWT Authentication implemented
- Password hashing using bcrypt
- Middleware used for authentication and authorization

• **Database: (MongoDB)**

MongoDB used as NoSQL database.

Collections:

- Users
- Category
- Product
- AddToCart
- Orders
- Payments

- Feedback

Each schema is defined using Mongoose.

4. Setup Instructions

• Prerequisites:

- Node.js (v16 or above)
- MongoDB (Local or Atlas)
- Git
- VS Code

Installation:

Step 1: Clone Repository

```
git clone https://github.com/PadmaGadi17/shopsmart-grocery-webapp.git
```

Step 2: Install Backend Dependencies

```
cd server
```

```
npm install
```

Step 3: Install Frontend Dependencies

```
cd client
```

```
npm install
```

Environment Variables:

Inside server:

Create .env file:

```
MONGO_URI=mongodb://127.0.0.1:27017/grocery-webapp
```

```
JWT_SECRET=mysecretkey
```

5. Folder Structure

• **Client:**

```
client/
|
|   └── src/
|       ├── components/
|       ├── admin_components/
|       ├── assets/
|       ├── services/
|       └── app.module.ts
```

• **Server:**

```
server/
|
|   └── src/
|       ├── db/
|       ├── models/
|       ├── routes/
|       └── app.js
```

6. Running the Application:

Frontend:

- cd client
- npx ng serve

Frontend runs on: <http://localhost:4200>

Backend:

- cd server
- npm start

Server runs on: <http://localhost:5100>

7. API Documentation

- **Authentication:**

Method	Endpoint	Description
POST	/register	Register new user
POST	/login	Login user

- **Products:**

Method	Endpoint
GET	/products
GET	/products/:id
POST	/add-products
PUT	/products/:id
DELETE	/products/:id

- **Cart:**

Method	Endpoint
POST	/add-to-cart
GET	/cart/:id
DELETE	/remove-from-cart/:id

- **Orders:**

Method	Endpoint
POST	/orders
GET	/orders
PUT	/orders/:id
GET	/my-orders/:id

8. Authentication

- Passwords hashed using bcrypt
- JWT tokens used for authentication
- Admin and user roles handled
- Token stored in frontend after login
- Middleware verifies token before protected routes

9. User Interface

- The UI includes:

- Home Page



- Login Page



The image features a man sitting at a desk, working on a laptop. A speech bubble above him contains a user interface mockup showing a profile picture and the text "WELCOME TO E-COMMERCE APP". To his left is a potted plant, and to his right is a trash bin.

Login

Email

Password

Sign In

[Don't have an account? click here](#)

- Registration Page



The image shows the same man from the previous login screen, now interacting with a laptop. A speech bubble above him displays a user interface with a profile picture and the text "SIGN UP". A potted plant is to his left, and a trash bin is to his right.

Registration

First Name

Last Name

User Name

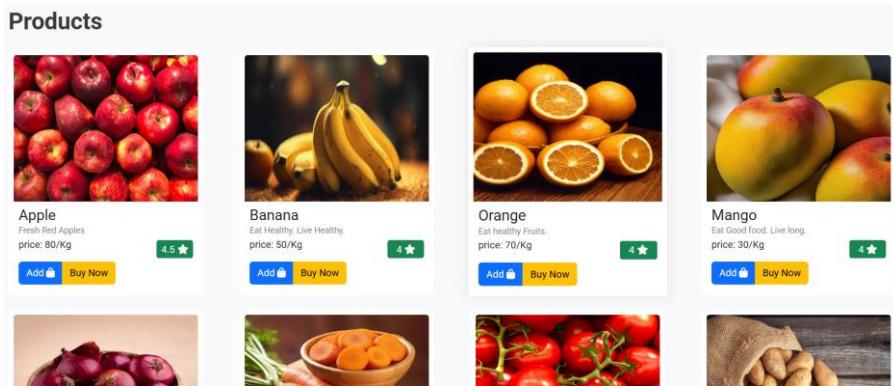
Email

Password

Sign Up

[Already have an account? click here](#)

- Product Listing Page



- Cart Page

My Cart

	Apple Price: 80 /-	Buy this product
	Banana Price: 50 /-	Buy this product
	Potato Price: 25 /-	Buy this product
DairyMilk Silk		

- Admin Dashboard

Grocery

- [Dashboard](#)
- [Users](#)
- [Products](#)
- [Add Products](#)
- [Add Category](#)
- [Orders](#)
- [Feedback](#)

Dashboard

Users Total users: 7 View Users	Orders Total orders: 4 View Orders	Add Products Click to add new products Add Products
--	---	--

Add Category
Click to add new category
[Add Category](#)

- Order Management Panel

Product ID:	6992bd93fea1cb59a9c37f44
Quantity:	10
Total price:	700
Payment Method:	cash-on-delivery
Address:	Gajuwaka
Created At:	Feb 17, 2026
Status:	Pending
Update status	
<hr/>	
Order ID:	69955ef7814d13ea7899eaa6
Fullname:	Lalitha gadi

10. Testing

- Manual testing performed for all APIs

Verified:

- Registration
- Login
- Add Product
- Order Placement
- Payment Creation

Tested using browser and Postman

11. Screenshots or Demo

- [Demo](#) [Video](#) [Link:](#)

(<https://drive.google.com/file/d/1j2SASFUITXHUQEf2uakJCDA0ro-eKSEi/view?usp=sharing>)

Screenshots included in documentation.

12. Known Issues

- No payment gateway integration

- No email notification system
- UI responsiveness can be improved
- Basic admin authentication logic

13. Future Enhancements

- Razorpay/Stripe Payment Integration
- Product image upload via Cloudinary
- Role-based access control
- Order tracking with real-time updates
- Admin analytics dashboard
- Deployment on AWS or Render