

SHOP SMART

PROJECT DOCUMENTATION

Features

□ **User Features**

- Easy Registration & Login
- Browse Products by Category & Brand
- Add to Cart & Wishlist
- Apply Promo Codes & Discounts
- Secure Payments (UPI, Card, Wallet)
- Live Order Tracking
- View Order History & Reorder

□ **Vendor/Admin Features**

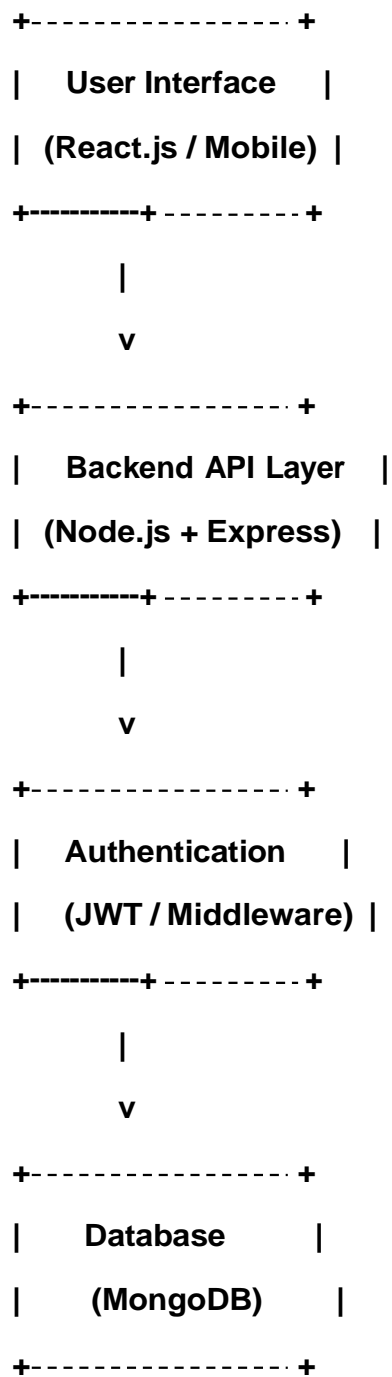
- Add/Edit/Delete Products
- Manage Stock & Pricing
- View Orders & Sales Reports
- Control Product Availability
- Manage User Queries

□ **Other Key Features**

- Responsive Design (Web & Mobile)
- Smart Search & Filters
- Personalized Product Recommendations
- Role-Based Access (User, Vendor, Admin)
- Notifications via Email/SMS

1. Architecture

System Architecture Diagram



Components

1. Frontend (React.js)

- Handles UI/UX for customers, vendors, and admins
- Interacts with backend via REST APIs
- Responsive design for web and mobile

2. Backend (Node.js + Express)

- REST API layer for handling user requests
- Routes for user, product, cart, order, and admin operations
- JWT-based authentication and authorization

3. Database (MongoDB)

- Stores users, products, carts, orders, and vendor data
- Enables flexible schema and fast querying

4. Authentication

- Secures user sessions using JWT
- Role-based access: user, vendor, admin

5. External Services (Optional)

- Payment Gateway (e.g., Razorpay/Stripe)
- Email/SMS Notification APIs
- Cloud Storage for product images (e.g., Cloudinary)

2. Setup Instructions

Prerequisites

- Node.js ≥ 18
- MongoDB installed and running (local or cloud - MongoDB Atlas)
- npm or yarn

Installation

```
git clone https://github.com/your-username/ShopSmart.git
```

```
cd ShopSmart
```

```
cd server
```

```
npm install
```

```
cd ../client
```

```
npm install
```

Environment Variables

Create a .env file in the /server folder with the following content:

PORT=5000

MONGO_URI=mongodb+srv://yourusername:<db_password>@cluster0.mongodb.net/shopsmart?retryWrites=true&w=majority&appName=Cluster0

JWT_SECRET=your_jwt_secret



Replace <db_password> with your actual MongoDB password.



Make sure your IP is whitelisted in MongoDB Atlas or use 0.0.0.0/0 for open access (development only).

Running the Application

1. Start the backend server:

```
cd server
```

```
npm start
```

Runs at <http://localhost:5000>

2. Start the frontend:

```
cd ../client
```

```
npm run dev
```

Runs at <http://localhost:5173>

3. Folder Structure

ShopSmart/

|

| — client/ # Frontend (React.js)

| | — public/ # Static files

| | — src/

| | — assets/ # Images, icons

| | — components/ # Reusable UI components

| | — pages/ # Route-based page components

| | — context/ # React context (e.g., Auth, Cart)

| | — services/ # API calls

```

|   └─ App.jsx      # Main app component
|   └─ main.jsx     # React DOM entry point
|
|   └─ server/      # Backend (Node.js + Express)
|   └─ config/      # DB connection, JWT config
|   └─ controllers/ # Request handlers
|   └─ models/      # Mongoose models (User, Product, Order)
|   └─ routes/      # API route files
|   └─ middleware/  # Auth, error handling
|   └─ utils/       # Helper functions
|   └─ .env         # Environment variables
|   └─ server.js    # Entry point for backend
|
|   └─ .gitignore   # Files/folders to ignore in Git
|   └─ README.md    # Project documentation
|   └─ package.json # Root config (optional if monorepo)

```

Application

Frontend:

- cd client
- npm start

Backend:

- cd server
- npm start

7. API Documentation

□ Authentication & Users

Method	Endpoint	Description	Access
POST	/api/auth/register	Register a new user	Public
POST	/api/auth/login	Login and receive JWT token	Public

GET	/api/users/me	Get current user info	Private
------------	----------------------	------------------------------	----------------

□ Products

Method	Endpoint	Description	Access
GET	/api/products	Get all products	Public
GET	/api/products/:id	Get product by ID	Public
POST	/api/products	Create new product	Admin/Vendor
PUT	/api/products/:id	Update product details	Admin/Vendor
DELETE	/api/products/:id	Delete a product	Admin/Vendor

□ Cart

Method	Endpoint	Description	Access
GET	/api/cart	Get user cart	Private
POST	/api/cart	Add item to cart	Private
PUT	/api/cart/:id	Update item quantity	Private
DELETE	/api/cart/:id	Remove item from cart	Private

□ Orders

Method	Endpoint	Description	Access
POST	/api/orders	Place a new order	Private
GET	/api/orders	Get current user's orders	Private
GET	/api/orders/all	Get all orders (admin only)	Admin
PUT	/api/orders/:id	Update order status (e.g. shipped, delivered)	Admin

📁 Admin/Vendor Management

Method	Endpoint	Description	Access
GET	/api/users	Get all users	Admin

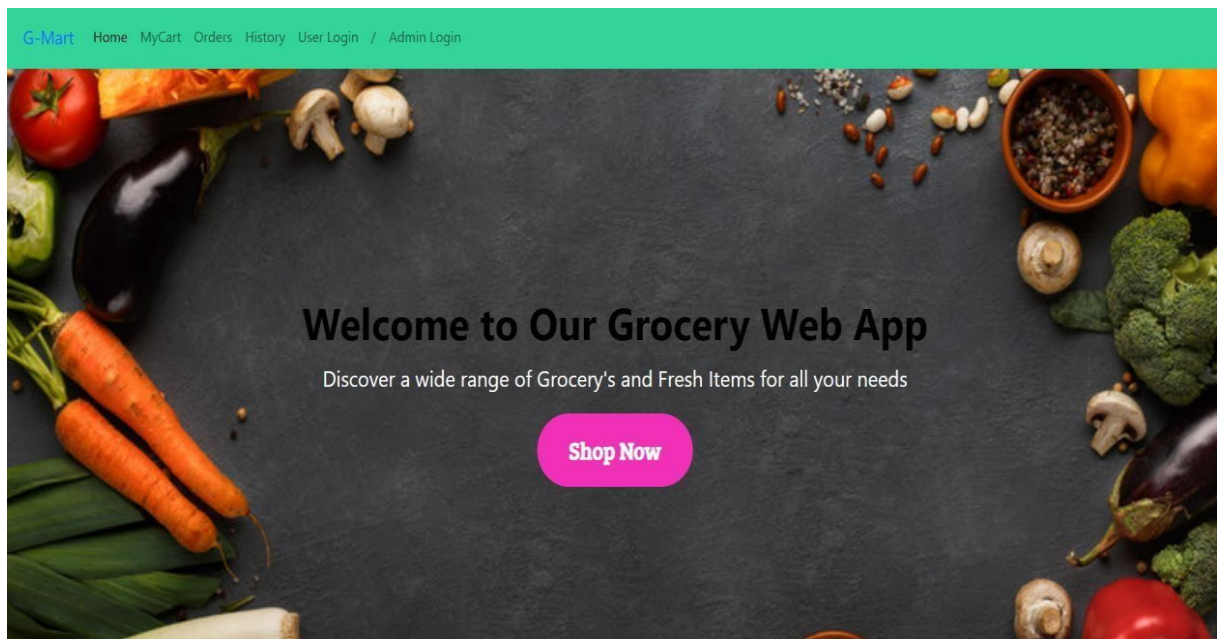
DELETE	/api/users/:id	Delete a user	Admin
GET	/api/dashboard/stats	Get overall statistics	Admin

8. Authentication

1. User Registers/Login
2. Server verifies credentials
3. If valid, server returns a JWT token
4. Token is stored in client (localStorage/cookies)
5. Protected routes require the token in headers

9. User Interface

Home Page with flight search



Login form

Login

Email

Password

Login

Don't have an account? [Sign Up](#)

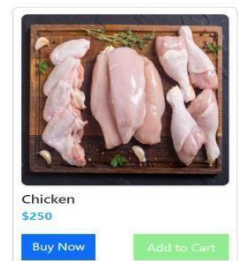
Items Page:-

Search By Product Name

Filter By Category

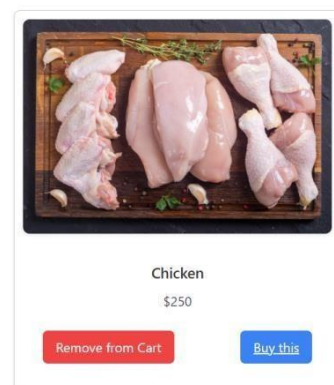
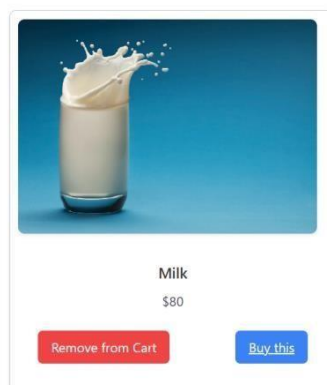
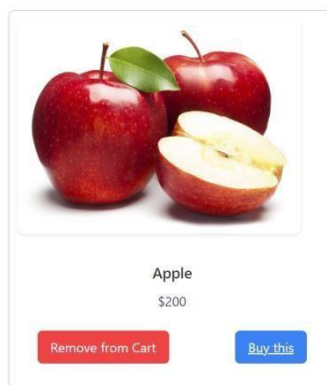
all

Products



My Cart:-

My Cart



My Orders Page:-

[G-Mart](#) [Home](#) [MyCart](#) [Orders](#) [History](#) [Logout](#)

My Orders

Order ID: 6614b085c30b51d3c700f20a

Name: syed arshad

Phone: 9505221870

Date: 2024-04-09T03:05:41.563Z

Price: 400

Status: Pending

Payment Method: credit

My History Page:-

[G-Mart](#) [Home](#) [MyCart](#) [Orders](#) [History](#) [Logout](#)

My History

Order ID: 6614a8ddc30b51d3c700f1b4

Name: syed arshad

Phone: 9505221870

Date: 2024-04-09T02:36:47.498Z

Price: 400

Status: Delivered

Payment Method: debit

Place Order Page:-

[G-Mart](#) [Home](#) [MyCart](#) [Orders](#) [History](#) [Logout](#)

localhost:3000/login

Order Details

First Name:

Enter your first name

Last Name:

Enter your last name

Phone:

Enter your phone number

Quantity:

Enter the quantity

Address:

Enter your address

Payment Method:

Cash on Delivery (COD)

Submit

Admin Dashboard Page:

Grocery Web App

DashboardUsersProductsAdd productOrdersLogout

Dashboard

Product Count

6 Products

View Products

User Count

1 Users

View Users

Order Count

2 Orders

View Orders

Add Product

Add

Users Page:-

Grocery Web App

DashboardUsersProductsAdd productOrdersLogout

Users

sl/no	Userid	User name	Email	Operation
1	6614a859c30b51d3c700f1a0	syed	syed@gmail.com	<div><div></div>view</div>

Add Product page:-

Grocery Web App

DashboardUsersProductsAdd productOrdersLogout

Add Product

Product Name

Enter product name

Rating

Enter product rating

Price

Enter product price

Image URL

Enter image URL

Category

Select Category

Count in Stock

Enter count in stock

Description

Enter product description

Add Product

Admin Orders Page:-

Grocery Web App

Dashboard

Users

Products

Add product

Orders

Logout

Orders

Order ID: 6614a8ddc30b51d3c700f1b4

Fullname: syed arshad

Phone: 9505221870

Product ID: 660e84641c659d0b2b10e135

Quantity: 2

Total price: 400

Payment Method: debit

Address: hyderabad

Created At: 2024-04-09T02:36:47.498Z

Status: Delivered

Delivered

Order ID: 6614b085c30b51d3c700f20a

Fullname: syed arshad

Phone: 9505221870

Product ID: 660e84641c659d0b2b10e135

Quantity: 2

Total price: 400

10. Testing

The ShopSmart application is tested using **Postman** for API endpoint validation and **manual UI testing** to ensure smooth user experience across registration, login, cart, and checkout flows. Automated testing can be added using **Jest** and **Super test** for backend coverage.

11. Demo

Link:

https://drive.google.com/file/d/1IASuHTlv2TI_arKlpXn0_FCMEFnnpB9EI/view?usp=drive_link

12. Known Issues

- ☐ **✗ No automated test coverage** – Testing is currently manual; missing unit and integration tests.
- ☐ **⚠ Limited error handling** – Some API responses lack detailed validation or error messages.
- ☐ **📦 No pagination or filtering in product APIs** – May affect performance with large datasets.
- ☐ **🔑 Token expiration handling not implemented** – JWT tokens do not auto-refresh, requiring manual re-login.

13. Future Enhancements

- ☐ **Add Refresh Tokens** – Implement secure token renewal to improve authentication flow.
- ☐ **Mobile App Version** – Build a native mobile app using React Native or Flutter.
- ☐ **Inventory Auto-Update** – Integrate real-time stock tracking for vendors.

- ❑ **AI-Powered Suggestions** – Use machine learning to improve product recommendations.
- ❑ **Chat Support System** – Enable live chat between customers and vendors.
- ❑ **Subscription/Auto-Order Feature** – Allow users to set recurring grocery orders.
- ❑ **Multi-language Support** – Expand user base by adding regional language options.
- ❑ **Invoice & Billing Module** – Generate downloadable order invoices for customers.
- ❑ **Analytics Dashboard** – Provide vendors/admins with real-time sales insights.

Github link: <https://github.com/jyothika19/shopsmart-your-digital-grocery-store-experience.git>