

# E-Commerce Platform Documentation

## 1. Project Overview

This project is a **Full-Stack E-Commerce Application** built using the MERN stack (MongoDB, Express.js, React.js, Node.js).

It provides user authentication, product browsing, cart management, checkout with Stripe, order management, and an admin dashboard for analytics.

## Main Features

- User registration and login (JWT authentication)
- Product catalog with search, filter, and pagination
- Add to cart & wishlist functionality
- Order history and order tracking
- Admin dashboard to manage products, view analytics, and track sales
- Stock-based charting for admin

## 2. Technology Stack

### Frontend

- React.js (Vite)
- Redux Toolkit (State Management)
- Tailwind CSS (Styling)
- Axios (API Calls)
- React Router DOM (Routing)
- Chart.js / Recharts (Admin Analytics)

### Backend

- Node.js
- Express.js
- MongoDB + Mongoose

- JSON Web Token (JWT) for authentication
- bcrypt.js for password hashing
- Multer / Cloudinary (for image upload)
- Stripe API (for simulated payment)

### **3. User Roles & Responsibilities**

#### **3.1 Customer**

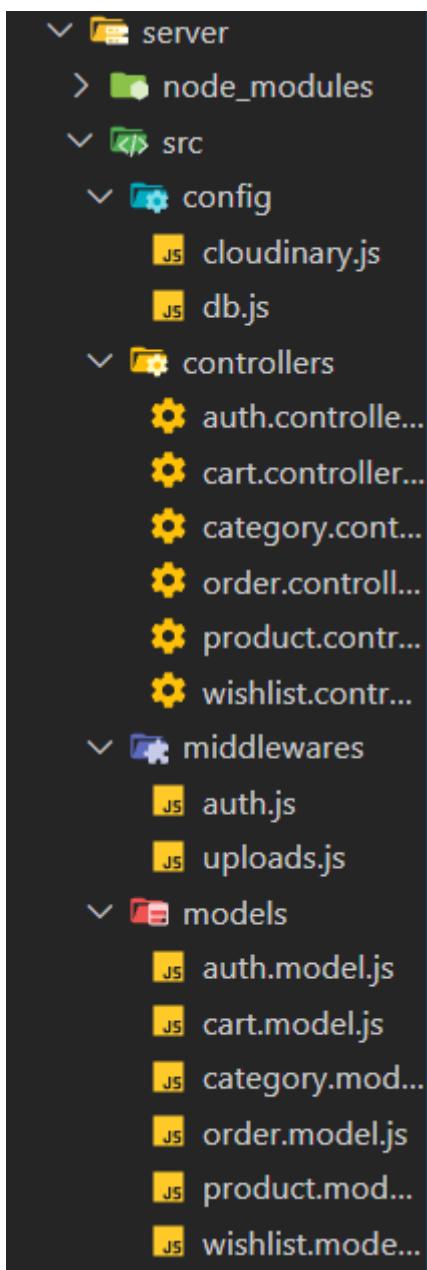
- Browse products with search & filters
- Add to cart or wishlist
- Checkout & place orders
- View order history
- Update profile details

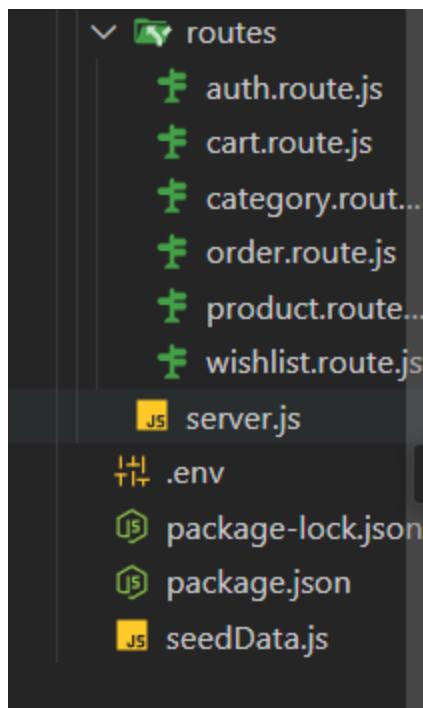
#### **3.2 Admin**

- Manage products (add, update, delete)
- Track total sales & stock analytics
- Manage orders (update status, cancel, refund)
- Access admin dashboard charts

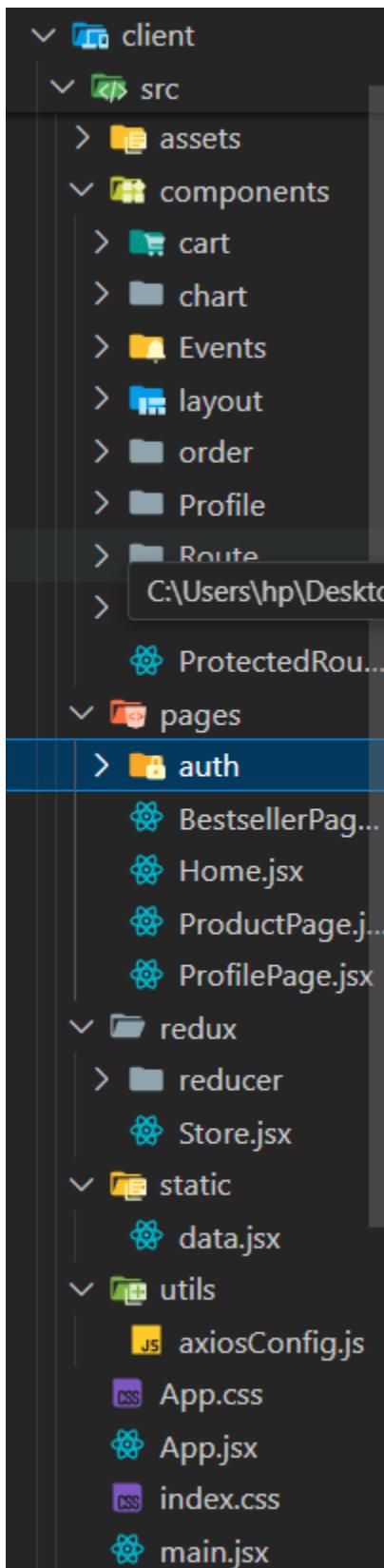
#### 4. Folder Structure (MVC)

##### Backend (Express + MVC)





## Frontend (React + Redux Toolkit)



## 5. Feature-wise Component & API Flow

Feature	Frontend Component	Backend API Endpoint(s)
User Auth	Login.jsx, Register.jsx, Profile.jsx	POST /api/auth/register, POST /api/auth/login, GET /api/auth/user, PUT /api/auth/user/address, PATCH /api/auth/user/profile-image
Product Listing	Home.jsx, ProductCard.jsx, ProductDetails.jsx	GET /api/products, GET /api/products/:id
Product Management (Admin)	AdminProduct.jsx, ProductForm.jsx	POST /api/products, PUT /api/products/:id, DELETE /api/products/:id
Cart Management	Cart.jsx	POST /api/cart, GET /api/cart, PUT /api/cart/:productId, DELETE /api/cart/:productId
Wishlist Management	Wishlist.jsx	POST /api/wishlist, GET /api/wishlist, DELETE /api/wishlist/:productId
Category Management (Admin)	AdminCategory.jsx, CategoryForm.jsx	GET /api/category, POST /api/category, PUT /api/category/:id, DELETE /api/category/:id
Orders	Checkout.jsx, Orders.jsx, OrderDetails.jsx	POST /api/orders, GET /api/orders/my-orders, GET /api/orders/:orderId, PATCH /api/orders/:orderId/status
Analytics (Admin)	AdminDashboard.jsx, AdminChart.jsx	GET /api/products (stock data), GET /api/orders (sales data)

## 6. Component Descriptions

- Navbar.jsx – Navigation bar with search, cart, wishlist, and authentication buttons.
- ProductCard.jsx – Displays product info, stock status, and an add-to-cart button.
- ProductDetails.jsx – Detailed product page with buy/add-to-wishlist functionality.
- AdminProduct.jsx – Admin view to manage product list.
- ProductForm.jsx – Admin form to add or edit products.

- Cart.jsx – Shows cart items with quantity updates and removal options.
  - Wishlist.jsx – Displays saved wishlist items with remove option.
  - AdminCategory.jsx – Admin view to manage product categories.
  - Checkout.jsx – Manages order placement and payment integration (Stripe).
  - Orders.jsx – Displays order history for the logged-in user.
  - OrderDetails.jsx – Shows detailed info for a single order.
  - AdminDashboard.jsx – Summary of key stats for the admin.
  - AdminChart.jsx – Graphical representation of sales and stock data.
- 

## 7. API Documentation

### Authentication

- POST /api/auth/register → Register new user (*with optional profile image upload*)
  - POST /api/auth/login → Login user
  - GET /api/auth/user → Get logged-in user details (*Protected*)
  - PUT /api/auth/user/address → Update user address (*Protected*)
  - PATCH /api/auth/user/profile-image → Update profile image (*Protected*)
- 

### Products

- GET /api/products → Get all products
  - GET /api/products/:id → Get single product by ID
  - POST /api/products → Add product (*Admin only, with image upload*)
  - PUT /api/products/:id → Update product (*Admin only, with image upload*)
  - DELETE /api/products/:id → Delete product (*Admin only*)
-

## Cart

- POST /api/cart → Add item to cart
  - GET /api/cart → Get user cart
  - PUT /api/cart/:productId → Update cart item quantity
  - DELETE /api/cart/:productId → Remove cart item
- 

## Wishlist

- POST /api/wishlist → Add item to wishlist
  - GET /api/wishlist → Get user wishlist
  - DELETE /api/wishlist/:productId → Remove wishlist item
- 

## Category

- GET /api/category → Get all categories
  - POST /api/category → Create category (*Admin only*)
  - PUT /api/category/:id → Update category (*Admin only*)
  - DELETE /api/category/:id → Delete category (*Admin only*)
- 

## Orders

- POST /api/orders → Place new order
- GET /api/orders/my-orders → Get orders of logged-in user
- GET /api/orders/:orderId → Get single order details
- PATCH /api/orders/:orderId/status → Update order status (*Admin only*)

## 8. Screenshots

### 8.1 Registration

The screenshot shows a registration form titled "Register Account". It includes fields for Name, Email, Password, Phone, Street, City, State, ZIP, Role (set to Customer), and Profile Image (with a "Choose File" button). A "Register" button is at the bottom, and a link to "Login here" is below it.

Register Account

Name

Email

Password

Phone

Street

City  State  ZIP

Role   No file chosen

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

### 8.2 Login

The screenshot shows a login form titled "Welcome Back". It has fields for Email (containing "sha@gmail.com") and Password (containing "....."). A "Login" button is at the bottom, and a link to "Sign Up" is below it.

Welcome Back

Login to your account

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

## 8.3 Home

The screenshot shows the homepage of an e-commerce website. At the top, there is a navigation bar with links for "Home", "Best Selling", and "Products". Below the navigation bar is a large banner with the text "We Have The Best For Everything" and a "Shop Now" button. The banner features a background image of a shopping cart filled with various products. Below the banner are four promotional boxes: "Free Shipping" (from all orders over 100\$), "Daily Surprise Offers" (Save up to 25% off), "Affordable Prices" (Get Factory direct price), and "Secure Payments" (100% protected payments). The overall theme is dark with yellow accents.

## 8.4 Products

The screenshot shows the products page of the e-commerce website. The page has a similar header and navigation bar as the homepage. Below the header, there are two search/filter input fields: "Min amount" and "Max amount". The main content area displays a grid of product cards. Each card includes a thumbnail image, the product name, a short description, star ratings, price, and the number of sales. The products shown are:

- Sample Product: A yellow car thumbnail. Rating: ★★★★☆. Price: \$ 455\$. Sold: 0.
- test shop: A blue circular thumbnail. Rating: ★★★★☆. Price: \$2000. Sold: 4.
- Gaming Headphone Asus: A black and blue headphones thumbnail. Rating: ★★★★☆. Price: \$239. Sold: 20.
- sum sung: A MacBook thumbnail. Rating: ★★★★☆. Price: \$1049. Sold: 35.
- Gents Shoes: A placeholder image thumbnail. Rating: ★★★★☆. Price: \$89. Sold: 49.
- A watch thumbnail (partially visible).
- An iPhone thumbnail (partially visible).

## 8.5 Cart

A screenshot of a web browser displaying a shopping cart interface. The cart contains two items:

- test**: test product, ₹8,000, quantity 4.
- Men's Watch 2023 Model**: Elegant and fashionable watch for men..., ₹79, quantity 1.

Below the cart summary is a "place order" button. At the bottom right of the page is a "Proceed to Checkout" button.

## 8.6 Wishlist

A screenshot of a web browser displaying a wishlist interface. The wishlist contains one item:

- Men's Watch 2023 Model**: Elegant and fashionable watch for men..., ₹100, ₹79.

At the bottom right of the page is a message: "You can move these items to your cart anytime."

## 8.7 Profile

The screenshot shows a profile page with a sidebar on the left containing links for Profile, Orders, Refunds, Inbox, Track Orders, Payment Methods, Address, and Logout. The main area displays a circular profile picture of a car, followed by form fields for Full Name (sha), Email Id (sha@gmail.com), Phone Number (7454524854), Zip Code (686673), State (Kerala), and Address (pezhakkappily). A yellow "Update" button is at the bottom.

Subscribe us for News

Enter Your email...

## 8.8 Order

The screenshot shows an order page with a sidebar on the left containing links for Profile, Orders, Refunds, Inbox, Track Orders, Payment Methods, Address, and Logout. The main area displays a table of orders:

ID	address	itemsQty	total	Status	Action
68970c2f90b25e432d473...	pezhakkappily, Ernakulam...	1	₹8000	processing	→
689729e33e7ae52aa91fc...	pezhakkappily, ernakulam...	1	₹79	delivered	→

At the bottom, there is a "Rows per page:" dropdown set to 100, and a message "1–2 of 2". Below the table is a "Snipping Tool" window showing the screenshot being copied to clipboard.

Subscribe us for News

Events and Offers

Screenshot copied to clipboard  
Automatically saved to screenshots folder.

Mark-up and share

## **9. How to Run the Project**

### **9.1 Backend**

```
cd server
```

```
npm install
```

```
npm run dev
```

### **9.2 Frontend**

```
cd frontend
```

```
npm install
```

```
npm run dev
```

### **9.2 MongoDB Connection**

- Use MongoDB Atlas or local MongoDB
- Update config/db.js with your connection string

## **10. Conclusion**

### **Outcomes:**

- Successfully built a role-based e-commerce application
- Integrated Stripe API for simulated payments
- Implemented analytics using Chart.js/Recharts

### **Challenges Faced:**

- Managing role-based routing & permissions
- Handling file uploads and storage (Cloudinary)