E-Commerce – Full Stack MERN Project Documentation

1. Introduction

Project Title: E-Commerce – Online Shopping Platform

Team Member:M.Sanjeevi

2. Project Overview

Purpose

The E-Commerce system is designed to provide a complete online shopping experience for customers, enabling them to browse products, add items to cart, place orders securely, and track purchase history. The platform eliminates traditional offline buying limitations and brings all essential shopping features into one centralized digital ecosystem.

The system aims to:

- Provide customers with a fast, responsive, and user-friendly shopping platform.
- Allow administrators to manage products, categories, users, and orders.
- Offer a scalable MERN-based architecture that ensures high performance.
- Automate product listing, inventory management, and order handling.

Features

- User Registration & Login (JWT Authentication)
- Product Listing, Filtering & Search
- Product Categories & Detailed View
- Add to Cart, Update Cart, Remove Items
- Checkout & Order Placement
- Admin Dashboard
- Product Management
- Order Management
- Secure REST APIs
- Responsive UI with Bootstrap

3. Architecture

Frontend (React.js)

- React with Hooks and Context API / Redux
- Axios for API communication
- React Router for navigation
- Protected Routes for secure access

Backend (Node.js + Express.js)

- RESTful API structure
- Authentication & authorization middleware
- Product, Cart, Order, User controllers
- Payment gateway integration support

Database (MongoDB + Mongoose)

- Collections: Users, Products, Categories, Orders, Cart Items
- Schema validation for data integrity
- Indexing for faster product search

4. Setup Instructions

Prerequisites

- Node.js (v16+)
- MongoDB / MongoDB Atlas
- Git

Installation Steps

1. Clone the repository.

2. Install dependencies in client & server folders: npm install

3. Create .env file in backend with:

- MONGO_DB="mongodb_connection_string"
- JWT_SECRET="your_secret_key

4. Start the backend: npm start

5. Start the frontend: npm start

5. Folder Structure

Client (React)

- /src/components – Reusable UI components
- /src/pages – Main screens
- /src/context – Global state management
- /src/services – API services
- /src/assets – Images, icons, banners

Server (Node.js)

- /routes – API endpoints
- /controllers – Business logic
- /models – Mongoose schemas
- /middleware – Auth & validation
- /config – DB connection & environment setup

6. Running the Application

Frontend: npm start

Backend: nodemon index.js

7. API Documentation

User APIs

- POST /api/user/register
- POST /api/user/login
- GET /api/user/profile

Product APIs

- GET /api/products

- GET /api/products/:id
- POST /api/products/create
- PUT /api/products/update/:id
- DELETE /api/products/:id

Cart APIs

- POST /api/cart/add
- GET /api/cart/user
- PUT /api/cart/update/:id
- DELETE /api/cart/remove/:id

Order APIs

- POST /api/orders/place
- GET /api/orders/user
- GET /api/admin/orders

Admin APIs

- GET /api/admin/users
- GET /api/admin/products
- GET /api/admin/orders

8. Authentication

- JWT-based authentication
- Access tokens stored in localStorage
- Role-based access (User, Admin)

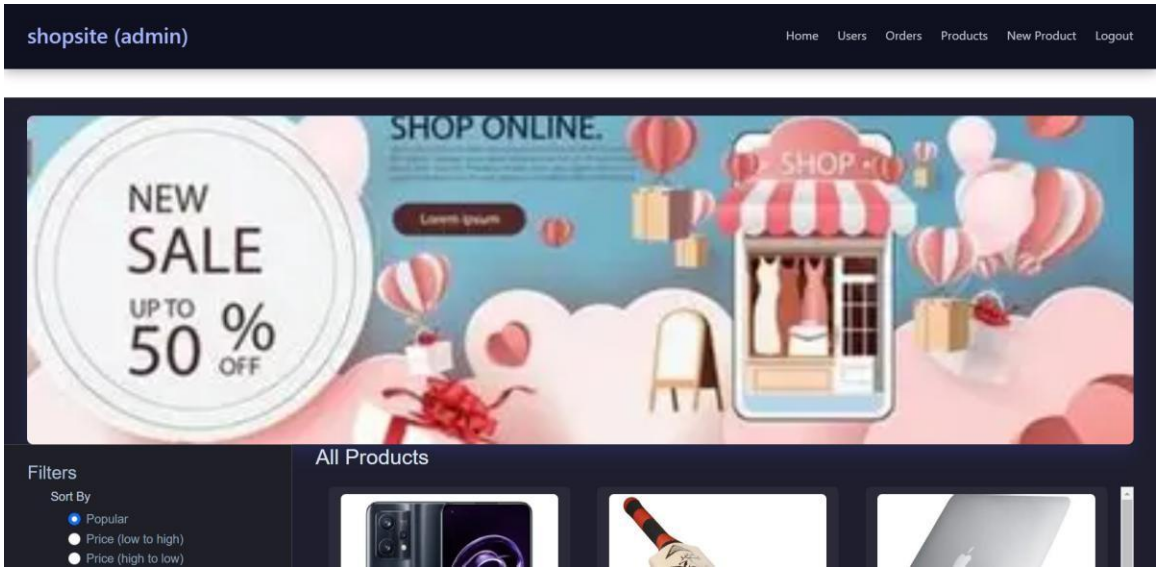
9. User Interface

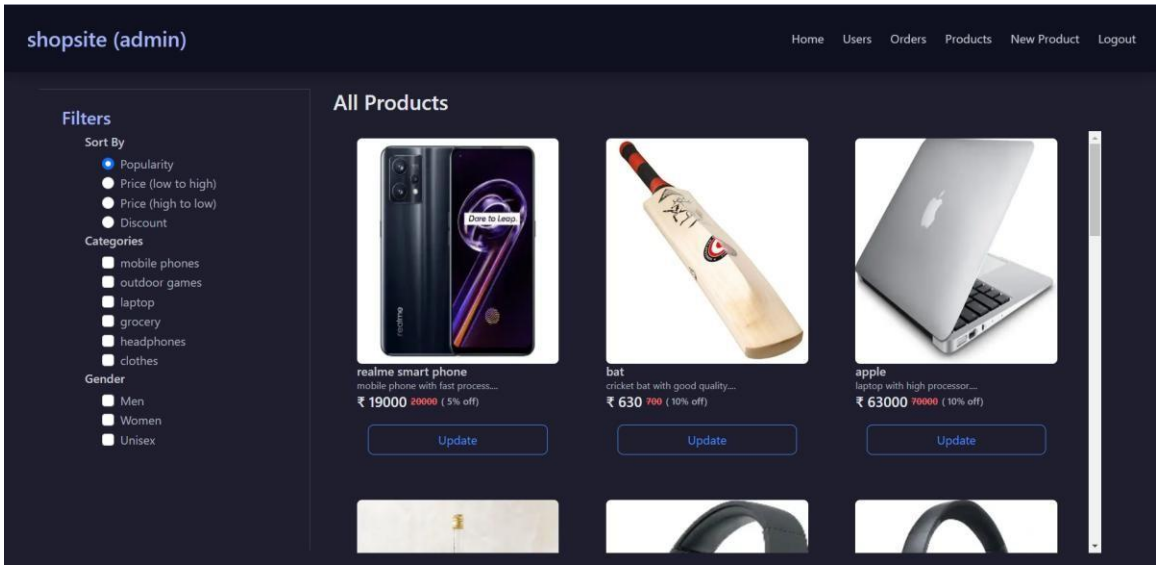
- Product grid layout
- Responsive navbar
- Product filters & categories
- Cart UI with quantity update
- Admin dashboard

10. Testing

- Token validation
- Browser compatibility testing

11. Screenshots / Demo





12. Known Issues

- Search delay on large data
- File upload depends on hosting
- Mobile UI improvements needed

13. Future Enhancements

- Payment gateway
- Wishlist system
- Live tracking
- Delivery partner module