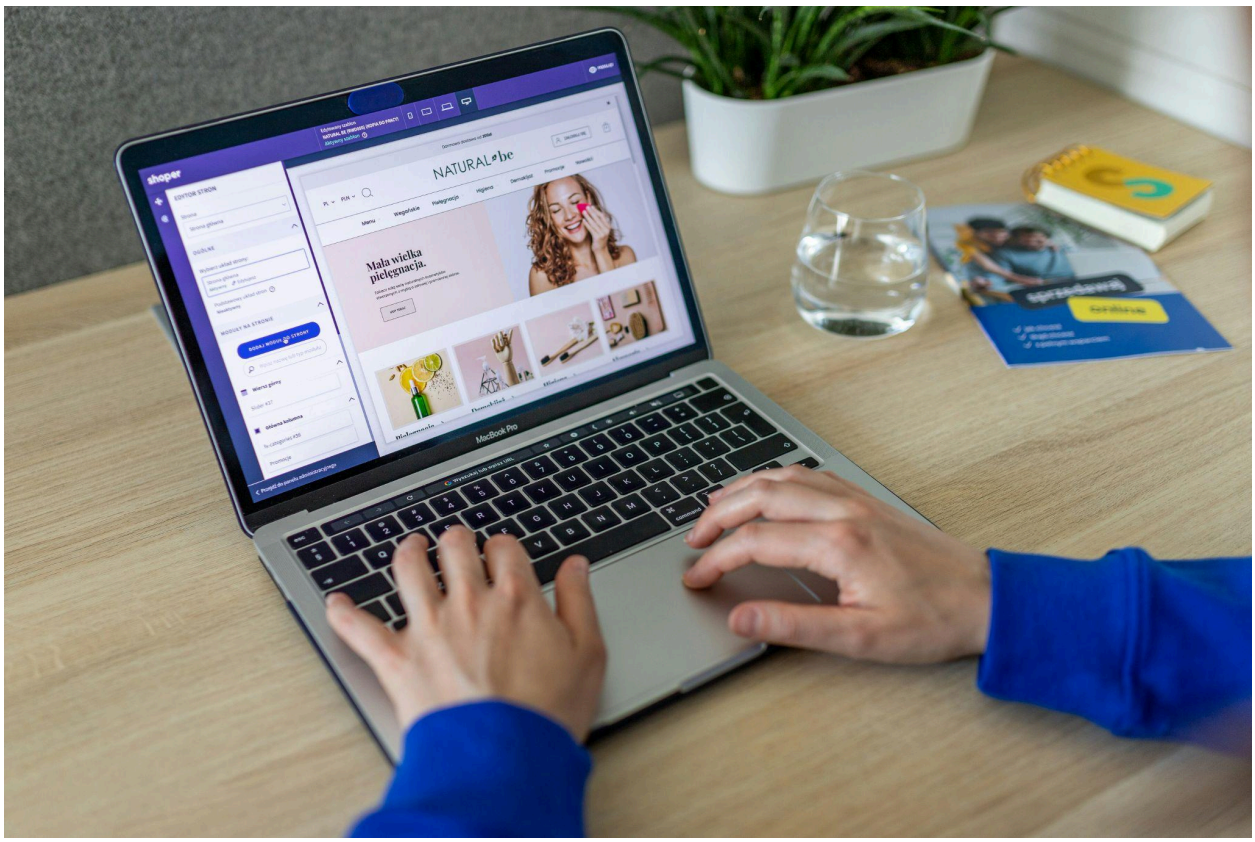
DB Theory

PROJECT REPORT

ECOMMERCE MULTIMART



==========

RAYYAN- UR - REHMAN **(23K-0634)**

MUHAMMAD ABDULLAH KHAN **(23K-0607)**

ABDUL AHAD MUNAF **(23K-0590)**

1. Introduction

E-Commerce MultiMart is a modern, full-stack web application designed to emulate the core functionalities of an advanced online shopping platform. The project demonstrates a professionally structured architecture integrating **frontend technologies (React/Vite)**, **Node.js/Express backend**, **authentication**, **cart & checkout workflow**, and **role-based dashboards**.

This project aims to provide a highly scalable, modular, and user-friendly digital commerce experience. It also serves as a practical demonstration of full-stack development concepts, user flow design, and backend API engineering—suitable for academic, professional, and enterprise-level implementation.

==========

2. Target Audience

E-Commerce MultiMart is intended for the following groups:

1. **Consumers (End Users)**
   Individuals who want to browse, search, and purchase products online with a smooth checkout experience.

2. **Sellers / Vendors**
   Small to medium-scale merchants who need their own online platform to list products, manage inventory, track orders, and analyze sales.

3. **Administrators**
   System-level users responsible for managing platform content, monitoring user activity, handling payments, and maintaining platform integrity.

4. **Developers / Researchers**
   Computer science students, software engineers, and researchers interested in studying full-stack e-commerce workflows, authentication patterns, and production-grade architectures.

==========

3. Project Scope

- Full user authentication (sign-up, login, protected routes).
- Product browsing, filtering, and detailed product pages.
- Cart and checkout system including payment.
- User-side order tracking.
- Role-based features for admin and sellers.
- Frontend user interface built with modern standards (React + Vite).
- Backend RESTful APIs using Node.js and Express.
- Centralized database handling with MongoDB.
- Responsive UI suitable for mobile and desktop.
  .

==========

4. Functional Requirements

4.1 User Module

- Users can register with email and password.
- Secure login with JWT-based authentication.
- View product categories and individual product details.
- Add products to cart, update quantities, remove items.
- Perform checkout and place an order.

4.2 Admin Module

- Create,update,delete product listings.

- Create,update,delete category.
- View sales analytics and revenue charts.

4.3 Cart & Checkout

- Add/remove items from cart.
- Address selection for delivery.

==========

5. Non-Functional Requirements

5.1 Performance

- API responses should be optimized to maintain low latency.
- Product listings and category pages should load within acceptable time

5.2 Security

- JWT authentication for all protected routes.
- Password hashing using industry-standard algorithms.
- Secure validation for input fields and payloads.
- Prevention of unauthorized access through role-based access control.

5.3 Scalability

- Microservice-ready backend architecture.
- Modular code structure allowing addition of new features without major refactoring.
- Database schema designed for horizontal scaling.

5.4 Reliability

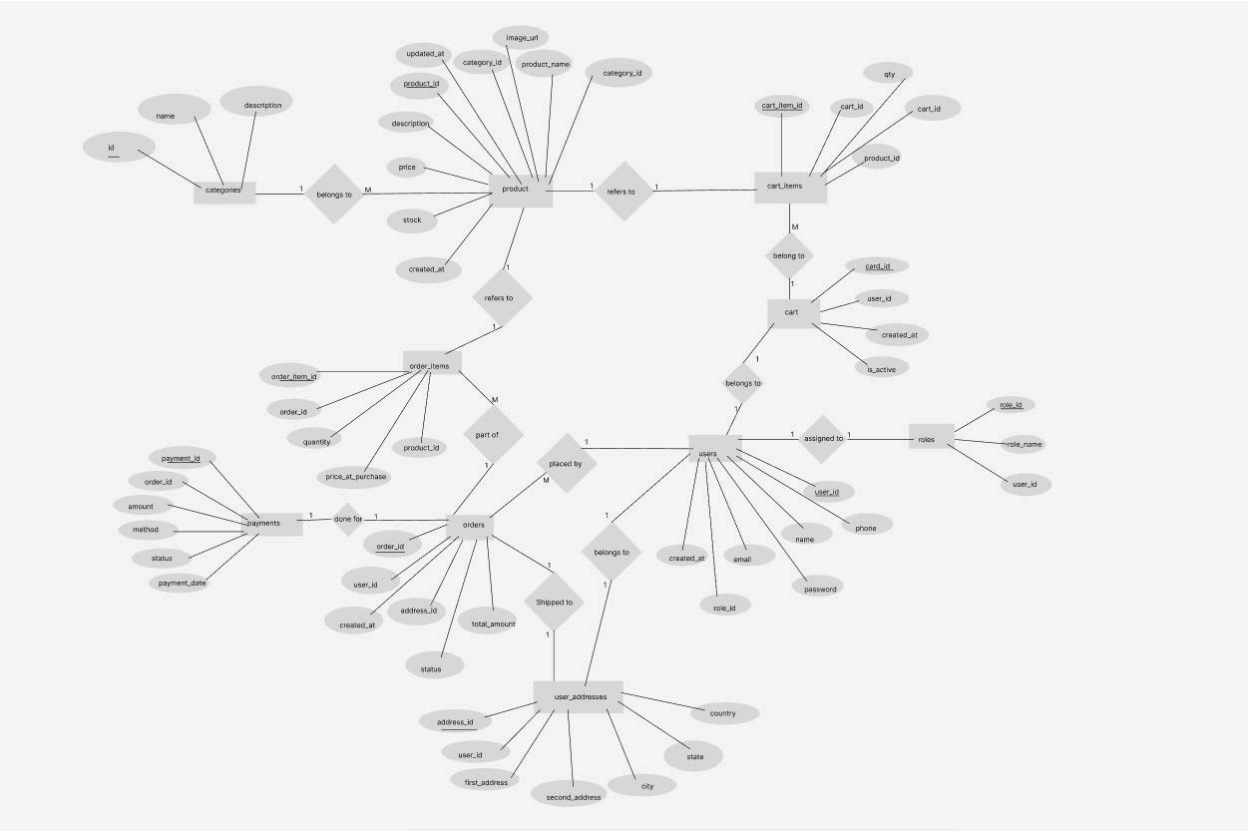
- Robust error handling and server-side validations.
- Consistent state management in frontend components.
- Maintaining data integrity across authentication, cart, and order modules.

5.5 Usability

- UI is designed to be intuitive, responsive, and user-friendly.
- Clear navigation flow for product browsing and checkout.
- Modern styling ensuring readability and accessibility.

==========

6.ERD Diagram



==========

7.Normalized Schema

CATEGORIES(category_id PK, name, description)

PRODUCTS(product_id PK, category_id FK, product_name, description, price, stock, image_url, created_at, updated_at)

USERS(user_id PK, role_id FK, name, email, phone, password, created_at)

ROLES(role_id PK, role_name)

CART(cart_id PK, user_id FK, created_at, is_active)

CART_ITEMS(cart_item_id PK, cart_id FK, product_id FK, quantity)

USER_ADDRESSES(address_id PK, user_id FK, first_address, second_address, city, state, country)

ORDERS(order_id PK, user_id FK, address_id FK, total_amount, status, created_at)

ORDER_ITEMS(order_item_id PK, order_id FK, product_id FK, quantity, price_at_purchase)

PAYMENTS(payment_id PK, order_id FK, amount, method, status, payment_date)

==========

8. Conclusion

E-Commerce MultiMart represents a well-structured and technically solid e-commerce platform, demonstrating proficiency in modern full-stack web development, REST API design, UI/UX principles, and secure authentication patterns. The architecture is scalable, modular, and expandable, making it ideal for academic submission, client demonstration, or future commercial development.

With additional enhancements such as integrating real payment gateways, SMS/email notifications, advanced analytics, and AI-powered recommendations, the platform can be transformed into a production-level e-commerce solution capable of serving thousands of concurrent users.

==========

9. References

1. Official React Documentation – https://react.dev
2. Node.js & Express Documentation – https://expressjs.com
3. JWT Authentication Standards – https://jwt.io
4. Vite Build Tool – https://vitejs.dev