🛒 MERN Stack Project Assignment

E-Commerce Web Application

🎯 Project Objective

Build a **full-stack E-Commerce web application** using:

- **MongoDB**
- **Express.js**
- **React.js**
- **Node.js**

The system must support:

- User authentication
- Product management
- Shopping cart
- Order processing
- Admin dashboard
- Payment integration (mock or real)
- Role-based access

🧱 Project Architecture

Client (React)

↓

Express API (Node.js)

↓

MongoDB Database

👥 User Roles

1. **Customer**
2. **Admin**

🗄️ Database Design (MongoDB Models)

1️⃣ User Model

```
{
  name: String,
  email: { type: String, unique: true },
  password: String (hashed),
  role: { type: String, enum: ["customer", "admin"] },
  isVerified: Boolean,
  createdAt: Date,
  updatedAt: Date
}
```

2️⃣ Product Model

```
{
  title: String,
  description: String,
  price: Number,
  discountPrice: Number,
  category: String,
  stock: Number,
  images: [String],
  ratings: Number,
  numReviews: Number,
  createdBy: ObjectId (Admin),
  createdAt: Date
}
```

3️⃣ Cart Model

```
{
  user: ObjectId,
  items: [
    {
      product: ObjectId,
```

```
    quantity: Number
   }
 ],
 totalAmount: Number
}
```

4️⃣ Order Model

```
{
 user: ObjectId,
 items: [
  {
   product: ObjectId,
   quantity: Number,
   price: Number
  }
 ],
 shippingAddress: {
  address: String,
  city: String,
  pincode: String,
  country: String
 },
 paymentMethod: String,
 paymentStatus: String,
 orderStatus: String,
 totalAmount: Number,
 createdAt: Date
}
```

5️⃣ Review Model (Optional Advanced)

```
{
```

user: ObjectId,

product: ObjectId,

rating: Number,

comment: String,

createdAt: Date

}

🔐 **Authentication Requirements**

Students must implement:

- JWT authentication
- Password hashing (bcrypt)
- Email verification (token based)
- Password reset
- Role-based route protection

Middleware Required:

- authMiddleware
- adminMiddleware

🌐 **Backend API Specifications**

🔑 **Auth Routes**

POST /api/auth/register

POST /api/auth/login

GET /api/auth/verify/:token

POST /api/auth/forgot-password

POST /api/auth/reset-password

GET /api/auth/me

🛍️ **Product Routes**

GET /api/products

GET /api/products/:id

POST /api/products (Admin)

PUT /api/products/:id (Admin)

DELETE /api/products/:id (Admin)

Features Required:

- Pagination
- Search by name
- Filter by category
- Sort by price
- Stock validation

🛒 Cart Routes

GET /api/cart

POST /api/cart/add

PUT /api/cart/update

DELETE /api/cart/remove/:productId

DELETE /api/cart/clear

📦 Order Routes

POST /api/orders

GET /api/orders/my-orders

GET /api/orders/:id

GET /api/orders (Admin)

PUT /api/orders/:id/status (Admin)

💻 Frontend (ReactJS) Requirements

🔹 Pages to Implement

Public Pages

- Home Page
- Product Listing Page
- Product Details Page
- Login Page
- Register Page

Customer Pages

- Dashboard
- Cart Page
- Checkout Page
- Order History
- Profile Page

Admin Pages

- Admin Dashboard
- Product Management
- Order Management
- User Management

🧩 **React Technical Requirements**

Students must use:

- Functional Components
- React Hooks
- React Router
- Axios
- Context API or Redux
- Protected Routes
- Form Validation

💳 **Payment Integration**

Choose ONE:

- Razorpay (India friendly)
- Stripe
- Mock payment system

Must:

- Update payment status

- Update order status after payment

📦 Submission Requirements

Students must submit:

- GitHub Repository
- README.md with:
    - Setup instructions
    - API documentation
    - Environment variables
- Screenshots
- Deployed links (Optional Bonus)