Clothing eCommerce Website with MERN Stack - Full Roadmap

□ PHASE 1: PLANNING & SETUP

1. Define Requirements

- User roles: Guest, Customer, Admin
- Core features:
  - Product listing (by category, search, filters)
  - Product details page
  - Cart system
  - Wishlist
  - Signup/login (JWT-based)
  - Admin dashboard (manage products/orders/users)
  - Checkout with payment (Stripe/Razorpay)
  - Order history and tracking

2. Tech Stack & Tools

- **Frontend:** React, Tailwind CSS, React Router, Redux/Context API
- **Backend:** Node.js, Express.js
- **Database:** MongoDB with Mongoose
- **Auth:** JWT + bcrypt
- **Payments:** Stripe or Razorpay
- **Deployment:** Vercel (frontend), Render/Railway (backend), MongoDB Atlas (DB)

3. Project Setup

- Set up GitHub repo with monorepo or two folders: `client/` and `server/`
- Create boilerplates for React and Node
  - `npx create-react-app client`
  - `npm init -y` for `server/`

□ PHASE 2: BACKEND DEVELOPMENT

1. Connect MongoDB Atlas

- Use Mongoose
- Create `.env` and add `MONGO_URI`, `JWT_SECRET`

2. User Authentication

- Register/Login routes
- Hash passwords using `bcrypt`
- Generate JWT on login
- Middleware: protect routes with `authMiddleware`

3. Product API

- Model: title, description, price, image, category, size, stock
- CRUD routes for products
- Upload product images (use Cloudinary or Multer)

4. Cart & Orders

- Cart: Add/remove/update items
- Order: create order, order model, order history
- Admin: update order status

□ PHASE 3: FRONTEND DEVELOPMENT

1. Basic UI Setup

- Tailwind CSS + Mobile responsive design
- Layout: Navbar, Sidebar, Footer

2. Pages to Build

- Home (banner, featured products)
- Shop (category filter, sort, search)
- Product Detail Page
- Cart Page
- Checkout Page
- Login/Register Page
- User Profile (update info, view orders)
- Admin Dashboard (CRUD products, view orders/users)

3. State Management

- Cart (Context API/Redux)
- User Auth (localStorage + Context/Redux)
- Orders (GET/POST via API)

4. API Integration

- Use `axios` for API calls
- Secure private routes (auth headers)

□ PHASE 4: PAYMENT & ORDER FLOW

1. Integrate Stripe/Razorpay

- Add checkout form
- Create backend route for payment intent
- Handle success/failure
- Save order details in MongoDB

□ PHASE 5: ADMIN PANEL

1. Admin Routes

- Create isAdmin middleware
- Allow admin to:
  - Add/edit/delete products
  - View/manage users
  - View/update orders

2. Admin Dashboard UI

- Use a separate layout (e.g. `/admin`)
- Tables for products, users, orders

☁ PHASE 6: DEPLOYMENT

1. Frontend (Vercel or Netlify)

- Push React app to GitHub
- Connect GitHub to Vercel

- Set up environment variables

2. Backend (Render or Railway)

- Push server code
- Add MongoDB URI, JWT secret
- Enable auto-deploy

3. MongoDB Atlas

- Free cloud database
- Set IP access and create user/password

□ PHASE 7: EXTRA FEATURES (OPTIONAL)

- Wishlist
- Ratings & reviews
- Coupons
- Email confirmation / password reset
- Dark mode
- PWA (Progressive Web App)

□ Project Folder Structure (Monorepo)

```
ecommerce-app/
│
├── client/               # React frontend
│   ├── src/
│   │   ├── components/
│   │   ├── pages/
│   │   ├── context/ or redux/
│   │   └── App.js
│
├── server/               # Node + Express backend
│   ├── controllers/
│   ├── models/
│   ├── routes/
│   ├── middleware/
│   └── index.js / app.js
│
├── .gitignore
└── README.md
```