E-Com Application

Folder Structure

```
ecommerce-app/
│
├── backend/                          # Backend code
│   ├── config/                       # Configurations (e.g., database, JWT)
│   │   ├── db.js                     # MongoDB connection
│   │   ├── stripe.js                 # Stripe configuration
│   │   └── dotenv.js                 # Environment variable loader
│   │
│   ├── controllers/                  # Route handlers
│   │   ├── authController.js         # User authentication
│   │   ├── productController.js      # Product operations
│   │   ├── orderController.js        # Order management
│   │   ├── cartController.js         # Cart operations
│   │   └── paymentController.js      # Stripe payment logic
│   │
│   ├── middleware/                   # Middleware functions
│   │   ├── authMiddleware.js         # Protect routes with JWT
│   │   ├── errorMiddleware.js        # Custom error handling
│   │
│   ├── models/                       # Mongoose models
│   │   ├── User.js                   # User model
│   │   ├── Product.js                # Product model
│   │   ├── Order.js                  # Order model
│   │   ├── Cart.js                   # Cart model
│   │   └── Payment.js                # Payment model (Stripe webhook)
│   │
│   ├── routes/                 # API routes
│   │   ├── authRoutes.js       # User authentication routes
│   │   ├── productRoutes.js    # Product routes
│   │   ├── orderRoutes.js      # Order routes
│   │   ├── cartRoutes.js       # Cart routes
```

```
│   │   └── paymentRoutes.js   # Payment routes
│   │
│   ├── utils/              # Utility functions
│   │   ├── jwtToken.js       # Generate JWT tokens
│   │   └── errorHandler.js    # Error response formatting
│   │
│   ├── .env                # Environment variables
│   ├── server.js           # Main backend entry file
│   └── package.json          # Backend dependencies
│
├── frontend/               # Frontend code
│   ├── public/             # Public assets
│   │   ├── index.html        # Main HTML file
│   │
│   ├── src/                # Source code
│   │   ├── components/       # Reusable UI components
│   │   ├── pages/           # Application pages
│   │   ├── redux/           # State management (actions, reducers, store)
│   │   ├── services/         # API calls using axios
│   │   ├── App.js           # Main app component
│   │   ├── index.js         # Entry point
│   │   └── routes.js         # Route configurations
│   │
│   ├── package.json          # Frontend dependencies
│
└──
README.md               # Project documentation
```

Database

User DB

```
name: { type: String, required: true },
  email: { type: String, required: true, unique: true },
  password: { type: String, required: true },
```

```
  isAdmin: { type: Boolean, default: false },
  address: {
    street: String,
    city: String,
    postalCode: String,
    country: String,
  },
```

Product DB

```
  name: { type: String, required: true },
  description: { type: String, required: true },
  price: { type: Number, required: true },
  category: { type: String, required: true },
  countInStock: { type: Number, required: true },
  rating: { type: Number, default: 0 },
  numReviews: { type: Number, default: 0 },
  image: { type: String },
```

Cart DB

```
  user: { type: mongoose.Schema.Types.ObjectId, ref: 'User', re
  items: [
    {
      product: { type: mongoose.Schema.Types.ObjectId, ref: 'Pr
      quantity: { type: Number, required: true, min: 1 },
    },
  ],
  totalPrice: { type: Number, required: true }
```

Order DB

```
  user: { type: mongoose.Schema.Types.ObjectId, ref: 'User', re
  orderItems: [
```

```
    {
      product: { type: mongoose.Schema.Types.ObjectId, ref: 'Pr
      quantity: { type: Number, required: true },
    },
  ],
  shippingAddress: {
    street: String,
    city: String,
    postalCode: String,
    country: String,
  },
  paymentResult: {   // Store Stripe payment details inline
    stripeId: { type: String },      // PaymentIntent ID from St
    status: { type: String },        // Payment status (e.g., "s
    amount: { type: Number },        // Payment amount
    created: { type: Date },         // Payment date
  },
  totalPrice: { type: Number, required: true },
  isPaid: { type: Boolean, default: false },
  paidAt: { type: Date },
```