**🏨 Simplified MERN Hotel Management System (HMS)**

Think of the project in **modules** instead of a huge document.

**🔹 Core Modules**

1. **Authentication & Roles**
   * Users: Admin, Manager, Receptionist, Housekeeping, Maintenance, Guest.
   * Role-based access (RBAC) using JWT + middleware.
2. **Room Management**
   * Add/edit/delete rooms.
   * Room status: available, occupied, cleaning, maintenance.
   * Pricing per room type.
3. **Reservations**
   * Guests can book rooms (online form).
   * Receptionist can also make bookings manually.
   * Automated check-in/check-out flow.
4. **Billing & Invoicing**
   * Auto-generate bills based on stay duration + extra services.
   * Export invoice as PDF/email.
5. **Housekeeping & Maintenance**
   * Assign cleaning tasks.
   * Staff updates room status (cleaned, needs maintenance).
   * Track maintenance requests.
6. **Reports & Analytics** (optional for MVP)
   * Occupancy rate.
   * Revenue reports.
   * Guest feedback analysis.

**⚙️ Tech Breakdown (MERN Stack)**

**Backend (Node.js + Express + MongoDB)**

* **Models (MongoDB Collections)**:
  + User (name, role, email, password hash).
  + Room (type, price, status).
  + Reservation (guest, room, check-in/out dates).
  + Invoice (amount, services, reservation ID).
  + Maintenance (room, issue, status).
* **APIs (Express)**:
  + /auth/register, /auth/login → JWT Auth.
  + /rooms → CRUD operations.
  + /reservations → create, update, cancel bookings.
  + /billing → generate invoices.
  + /maintenance → create/resolve requests.

**Frontend (React)**

* **Admin Dashboard** (manage users, reports).
* **Receptionist Panel** (reservations, check-in/out).
* **Housekeeping/Maintenance Panel** (update status).
* **Guest View** (book rooms, check status, give feedback).

Use libraries like:

* **React Router** → navigation.
* **Axios/Fetch** → API calls.
* **Redux/Context API** → state management.
* **TailwindCSS / Material-UI** → UI design.

**🏗️ Step-by-Step Guide to Build It**

**✅ Step 1: Setup**

* Install Node.js, MongoDB, React.
* Create project folders:

backend/

frontend/

**✅ Step 2: Backend (Express + MongoDB)**

1. Initialize backend:

mkdir backend && cd backend

npm init -y

npm install express mongoose cors bcryptjs jsonwebtoken

1. Create models (User, Room, Reservation, Invoice).
2. Build routes (Auth, Rooms, Reservations).
3. Add JWT authentication middleware.
4. Test APIs in **Postman**.

**✅ Step 3: Frontend (React)**

1. Initialize frontend:

npx create-react-app frontend

cd frontend

npm install axios react-router-dom

1. Setup pages:
   * /login
   * /dashboard (role-based view).
   * /rooms, /reservations, /billing.
2. Connect React to backend with Axios.
3. Implement **role-based navigation** (Admin sees full dashboard, Receptionist sees bookings only, etc.).

**✅ Step 4: Integration**

* Connect booking system with room status updates.
* Auto-generate invoices after check-out.
* Allow housekeeping staff to update room cleaning status.

**✅ Step 5: Extras**

* Add charts for reports (using **Chart.js/Recharts**).
* Add **email notifications** (Nodemailer).
* Deploy backend (Heroku/Render) + frontend (Netlify/Vercel).

**🎯 MVP (Minimum Viable Product)**

If you’re short on time, first focus on:

* **Login system (Admin, Receptionist, Guest).**
* **Room management (CRUD + availability).**
* **Reservation system (book → check-in/out).**
* **Billing (basic invoice).**

Once this works → then add housekeeping, maintenance, analytics.

👉 Would you like me to create a **project folder structure with sample code (models, routes, frontend components)** so you can directly start coding from there?