# Hotel Room Booking System Backend Documentation

## Overview

This backend system facilitates hotel room booking management through APIs and automated WhatsApp communication. It is designed for scalability, security, and maintainability, using SQLite for the database and Node.js with Express.js for API development. Key features include user and admin management, booking functionalities, WhatsApp reminders, and feedback collection.

## Architecture

### Technologies Used:

- Node.js: Backend runtime environment.  
- Express.js: Web framework for building APIs.  
- SQLite: Lightweight database solution.  
- bcrypt: For password hashing.  
- JWT (jsonwebtoken): Secure authentication and authorization.  
- Axios: HTTP client for external API calls.  
- node-schedule/cron: Task scheduling.  
- dotenv: Environment variable management.

### Key Files:

1. admin.js  
2. Bot.js  
3. index.js

## Functional Modules

### 1. admin.js

Purpose: Handles admin user creation and login.

#### Functions

Hash Passwords: Uses bcrypt to securely hash passwords.

Example:  
  
bcrypt.hash(plainPassword, saltRounds, (err, hashedPassword) => {  
  
 db.run('INSERT INTO admins (username, password) VALUES (?, ?)', [username, hashedPassword]);  
  
});

### 2. Bot.js

Purpose: Integrates WhatsApp Cloud API for automated communication.

#### Key Functionalities

Send Text Messages:  
  
async function sendWhatsAppTextMessage(to, text) {  
  
 const response = await axios.post(`${WHATSAPP\_API\_URL}`, {  
  
 messaging\_product: "whatsapp",  
  
 recipient\_type: "individual",  
  
 to: to,  
  
 type: "text",  
  
 text: { body: text }  
  
 });  
  
}

Webhook Endpoints: Handles incoming WhatsApp messages.

### 3. index.js

Purpose: Main API logic for users, bookings, rooms, and admin functionality.

#### Endpoints

Admin Endpoints:  
  
1. Login: POST /admin/login  
  
2. View Users: GET /admin/users  
  
3. Manage Rooms: POST, PATCH, DELETE on /admin/rooms  
  
4. View Stats: GET /admin/stats

Booking Endpoints:  
  
1. Create Booking: POST /api/bookings  
  
2. Modify Booking: PATCH /api/bookings/:id  
  
3. Cancel Booking: DELETE /api/bookings/:id  
  
4. Check Availability: POST /api/rooms/availability

Feedback Endpoints: POST /api/feedback

## Deployment Instructions

1. Install dependencies: npm install  
  
2. Set environment variables in .env file:  
  
 - WHATSAPP\_API\_URL  
  
 - WHATSAPP\_ACCESS\_TOKEN  
  
 - JWT\_SECRET  
  
 - HOTEL\_NAME  
  
3. Run the server: node index.js  
  
4. Database will auto-create necessary tables on startup.