Hotel Concierge AI Chatbot 🏨

A next-gen AI-powered hotel chatbot and admin dashboard built for the IIT Hackathon 2025 (Challenge: AIChieftain). It handles room bookings, service requests, FAQs, and sentiment analysis using cutting-edge LLM tech.

🚀 Features

- Voice and text-based chatbot
- Booking system with availability check
- Room service request (with SMS confirmation)
- AI-generated replies via Google Gemini
- Sentiment analysis and dashboard summary
- Admin panel for staff to manage requests/bookings

📄 Tech Stack

- **Frontend**: React (Chatbot & Admin Dashboard)
- **Backend**: FastAPI
- **LLM**: Gemini 1.5 Flash (via LangChain)
- **Database**: Supabase
- **Email**: SendGrid
- **SMS**: Twilio

🌐 Live Demo

- **Chatbot**: https://chief-train-hr5t.vercel.app
- **Admin Panel**: https://chief-train-hr5t.vercel.app/admin

🔧 Setup Instructions

1. **Clone the repo**
2. **Add `.env` file** with required credentials in backend:

```
SUPABASE_URL=...
SUPABASE_KEY=...
OPENAI_API_KEY=...
SENDGRID_API_KEY=...
TWILIO_ACCOUNT_SID=...
```

TWILIO_AUTH_TOKEN=...
TWILIO_PHONE_NUMBER=...

3. **Start backend (FastAPI)**:

```
cd backend
python -m venv venv
venv\Scripts\activate
pip install -r requirements.txt
uvicorn app.main:app --reload
```

4. **Start frontend (React)**:

```
cd frontend
npm install
npm run dev
```

🔹 Sample Users

- **Room Guest**: use chatbot for queries/booking/service
- **Admin**: login with password `hotel@123` to manage backend

📏 System Architecture

🚀 Overview

This architecture powers a hotel chatbot and admin system using FastAPI backend, React frontend, and Gemini LLM integration.

🔹 Components

1. Frontend (React)

- Chatbot UI (`Chatbot.jsx`)
- Admin Dashboard (`AdminDashboard.jsx`)
- Handles all user input, displays responses, connects to backend APIs

2. Backend (FastAPI)

- Endpoints: `/chat`, `/booking`, `/request`, `/faq`
- LLM: Gemini 1.5 Flash via LangChain
- Sentiment Analysis & AI summary
- Email (SendGrid) & SMS (Twilio) integration

3. Database (Supabase)

- Stores `chats`, `bookings`, and `room_requests`
- Used by both chatbot and dashboard

4. External Services

- **Gemini**: LLM to respond to chat and classify sentiment
- **Twilio**: Sends SMS for room requests/resolution
- **SendGrid**: Sends booking confirmation/rejection emails

🔄 Data Flow

1. Guest interacts via chatbot UI
2. Frontend sends request to FastAPI backend
3. Backend processes input via Gemini and database
4. Response sent back to frontend UI and stored in Supabase
5. Admin dashboard retrieves & displays data

🔖 API Documentation

Base URL: [`https://chieftrain.onrender.com`](https://chieftrain.onrender.com)

📡 `/chat/`

- **POST**: `{ user_id, message }`
- **Returns**: `{ response: "..." }`
- Saves chat + sentiment in DB

📡 `/chat/sentiment-summary-ai`

- **GET**
- **Returns**: `{ top_positive, top_negative, top_neutral }`

📆 `/booking/`

- **POST**: `{ user_id, name, room_type, check_in, check_out }`
- **Returns**: `{ message: "submitted!" }`
- **GET**: Fetch all bookings
- **PATCH**: `/booking/{id}/status?status=confirmed|rejected`
- **DELETE**: `/booking/{id}`
- **GET**: `/booking/availability/?check_in=YYYY-MM-DD&check_out=YYYY-MM-DD`
- **Returns**: availability by room type

🛎️ `/request/`

- **POST**: `{ room_number, phone_number, request }`
- Sends SMS and logs request
- **GET**: Fetch all requests
- **PATCH**: `/request/{id}/resolve`
- **DELETE**: `/request/{id}`

📖 `/faq/`

- **GET**: Returns hardcoded FAQs (3 sample Q&A)