**Navatar Remote Hospital Consultation System - Complete Design Document**

**1. Introduction**

The Navatar system allows remote hospital consultations using robotic units (Navatars). This document provides a detailed explanation of the system architecture, including backend database schema, admin/user roles, booking logic, and mobile app interaction.

**2. Database Design**

**2.1 company**

CREATE TABLE company (

company\_id SERIAL PRIMARY KEY,

company\_name VARCHAR(100) NOT NULL,

pincode VARCHAR(10),

country\_code VARCHAR(50)

);

**2.2 navatar**

CREATE TABLE navatar (

navatar\_id SERIAL PRIMARY KEY,

company\_id INTEGER NOT NULL,

navatar\_name VARCHAR(100) NOT NULL,

location VARCHAR(100),

FOREIGN KEY (company\_id) REFERENCES company(company\_id) ON DELETE CASCADE

);

**2.3 users**

Defines users such as doctors, admins, and assistants.

CREATE TABLE users (

user\_id SERIAL PRIMARY KEY,

user\_name VARCHAR(100) NOT NULL,

email VARCHAR(100) NOT NULL,

mobileno VARCHAR(15),

company\_id INTEGER NOT NULL,

role VARCHAR(50), -- e.g., doctor, nurse, admin

FOREIGN KEY (company\_id) REFERENCES company(company\_id) ON DELETE CASCADE

);

**2.4 navatar\_admin**

Stores temporary navatar admins during active bookings.

CREATE TABLE navatar\_admin (

admin\_name VARCHAR(100) NOT NULL,

admin\_email VARCHAR(100) PRIMARY KEY,

navatar\_id INTEGER NOT NULL

);

**2.5 meeting**

Stores booking and session details.

CREATE TABLE meeting(

id SERIAL PRIMARY KEY,

user\_id INTEGER NOT NULL,

date DATE NOT NULL,

start\_time TIME NOT NULL,

duration INTEGER NOT NULL,

booking\_status VARCHAR(20) NOT NULL, -- e.g., confirmed, cancelled

meeting\_status VARCHAR(20) NOT NULL, -- e.g., completed, pending

booking\_reference VARCHAR(50) NOT NULL,

booking\_amount DECIMAL(10, 2),

partner\_id INTEGER NOT NULL, -- P.A or doctor ID

payment\_id SERIAL,

payment\_amount DECIMAL(10,2),

payment\_date\_time TIMESTAMP,

FOREIGN KEY (user\_id) REFERENCES users(user\_id) ON DELETE CASCADE

);

**3. User Role Flow**

**Super Admin**

* Adds companies
* Adds navatars to companies
* Assigns at least one hospital admin per company

**Hospital Admin**

* Adds users (doctors, nurses, PAs)
* Assigns navatars to floors/rooms

**Doctor/PA**

* Books available navatars for sessions
* Controls navatar from mobile app during session

**4. Booking and Usage Flow**

**4.1 Booking Logic**

1. User (doctor/PA) selects:
   * Floor/location
   * Date, time, and duration
2. System:
   * Filters navatars by company and location
   * Checks meeting table for conflict using SQL:

SELECT \* FROM navatar

WHERE company\_id = ?

AND location = ?

AND navatar\_id NOT IN (

SELECT navatar\_id FROM meeting

WHERE date = ?

AND start\_time < (requested\_start + duration)

AND (start\_time + interval 'duration minutes') > requested\_start

);

1. If available, a navatar is booked and meeting row created
2. Payment processed (e.g., via Razorpay)

**5. Mobile App Flow (After Booking)**

**Booking Confirmed**

* Doctor or PA logs into the mobile app
* Sees their confirmed sessions

**During Session**

* App UI provides:
  + Live video stream from Navatar
  + Buttons: Move Forward, Left, Right, Stop

**Backend Communication**

* App sends navigation commands via:
  + MQTT or WebSocket
  + Django backend receives and forwards to robot

**Navatar Admin**

* During usage, the current user becomes navatar admin (recorded in navatar\_admin table)
* Released after session ends

**6. Frontend Features**

**Super Admin Dashboard**

* Add company form
* Add navatar form (with location and company)
* View and manage navatars

**Hospital Admin Dashboard**

* Add user form
* Assign roles (dropdown)
* View users by company

**Doctor/PA Interface**

* Book navatar page (form with location/date/time)
* View upcoming sessions

**7. Backend Components (Django)**

**Models**

* Company
* Navatar
* User
* Meeting
* NavatarAdmin

**Views**

* Add company
* Add user
* Add navatar
* Book navatar
* Confirm payment

**API (After Booking)**

* /api/move/ (POST):

{

"booking\_id": 10,

"command": "forward"

}

* /api/status/ – optional, to poll navatar health/live stream

**8. Suggested Enhancements**

* Auto-generate user IDs (e.g., DOC001)
* Expiry/timeout for navatar control
* Push notifications for booking reminders
* Admin-level analytics dashboard

**9. Flowchart and Diagrams**

(Diagrams will be provided visually if needed in PNG. Example: Booking decision tree, mobile app connection.)

Let me know if you'd like this exported as a PDF, or if you want visual flowcharts and a mobile UI wireframe added!