1. **Hospital Admin Interface:** Adds doctors, nurses, personal assistants, and manages bookings.
2. **Doctor/PA Mobile App:** Used to view bookings, receive reminders, and control the Navatar.
3. **Django Backend:** API provider and booking validation system.
4. **MQTT Broker:** Publishes control commands to Navatar.
5. **Raspberry Pi (Navatar):** Executes commands, streams camera feed, and displays video of the doctor.

**5. Modules & Functionality**

**5.1 Super Admin**

* Add companies
* Assign hospital admins
* Add navatars
* View all hospitals and users

**5.2 Hospital Admin**

* Add users: doctors, nurses, personal assistants
* View and manage hospital's navatars
* View booking records and usage

**5.3 Doctor/PA**

* Log in via mobile app
* Book a navatar
* Receive notification/reminder before booking
* Use app to control Navatar during session

**5.4 Raspberry Pi (Navatar Robot)**

* Connects to MQTT broker
* Listens for movement commands (forward, backward, left, right)
* Controls wheels using GPIO and motor driver
* Streams camera view over HTTP/WebRTC
* Displays doctor’s video via mounted iPad/tablet

**6. Database Design**

**company**

* company\_id (PK)
* company\_name
* pincode
* country

**users**

* user\_id (PK)
* user\_name
* email
* mobileno
* role (doctor, nurse, pa, admin)
* company\_id (FK)

**navatar**

* navatar\_id (PK)
* company\_id (FK)
* navatar\_name
* location

**meeting**

* meeting\_id (PK)
* user\_id (FK)
* navatar\_id (FK)
* date
* start\_time
* duration
* booking\_status (pending, confirmed, cancelled)
* meeting\_status (pending, completed)
* booking\_reference

**payment** (optional)

* payment\_id (PK)
* meeting\_id (FK)
* payment\_amount
* payment\_method
* payment\_status
* payment\_reference
* payment\_date\_time

**7. Frontend Design**

* Super Admin: dashboard with buttons for adding companies, admins, and navatars.
* Admin: dashboard with user management, view bookings, assign roles.
* Doctor/PA App:
  + Login screen
  + Booking list
  + Navatar control screen (4 direction buttons)
  + Live camera feed
  + Session countdown

**8. Mobile App Flow**

1. Login using email/password or OAuth
2. API call to /api/my-bookings/ to fetch upcoming bookings
3. In-app timer or push notification to alert 30 minutes before session
4. Launch navatar control screen if session time is valid
5. Send control commands to Django API
6. Load video feed from Pi camera

**9. Raspberry Pi Hardware & Logic**

* Raspberry Pi 3/4
* L298N motor driver
* DC motors
* Pi Camera or USB camera
* MQTT client (Python + paho-mqtt)
* Python GPIO control script
* mjpg-streamer or WebRTC for video

MQTT command receiver example:

client.subscribe("navatar/2/control")

if message == "forward": move\_forward()

**10. API Endpoints**

**/api/my-bookings/** (GET)  
Returns a list of confirmed bookings for the user.

**/api/control/** (POST)  
Receives command and booking ID, verifies session, sends command to MQTT.

**/api/book-navatar/** (POST)  
Creates a new booking if navatar is idle.

**11. Real-Time Communication**

* MQTT broker (e.g., Mosquitto)
* Raspberry Pi subscribes to navatar/<id>/control
* Django publishes command on valid booking time
* App sends POST request on each button press

**12. Session Flow**

1. Doctor books navatar
2. Reminder sent before session
3. App shows control screen
4. App sends command
5. Django publishes to MQTT
6. Pi receives command → robot moves
7. Camera feed streamed live
8. Session ends automatically or manually

**13. Testing**

* Manual booking validation tests
* Real-time motor tests with dummy commands
* Video streaming latency tests
* Booking session window checks

**14. Future Enhancements**

* Obstacle detection and path planning
* Integration with hospital records (EHR)
* Two-way voice communication
* Automatic docking/charging system
* Indoor location-based navigation

**15. Conclusion**

Navatar demonstrates the possibility of delivering healthcare consultations remotely using cost-effective robotics and real-time systems. By combining Django, MQTT, and Raspberry Pi, the platform achieves seamless booking, secure navigation, and effective virtual presence, transforming how doctors can reach patients in real-time without being physically present.

**Appendix**

* MQTT setup notes
* Wiring diagram for Raspberry Pi + L298N
* Sample JSON payloads for control and bookings
* Screenshot of control screen and booking confirmation

**End of Document**