**Project Title: "Smart Railway Ticket Booking System"**

**Overview:**

Develop a smart railway ticket booking system where:

- Users enter the number of seats & passenger details (No manual seat selection).
- The system must intelligently allocate seats in the same row (where possible).
- Pending seats in partially filled rows should be filled first before opening a new row.
- A complex analytics dashboard should aggregate booking data in multiple ways.

---

Key Features

**Backend (Node.js, Express, MongoDB, WebSocket's, Redis for concurrency handling)**

**Train & Compartment Structure**

- Each train consists of multiple compartments (A1, A2, B1, etc.).
- Each compartment has multiple rows of 6 seats per row.
- **Data Model:**
    - Train Details (Train Number, Name, Route)
    - Compartment Details (Total Seats, Available Seats, Rows & Seats Status)

**Smart Seat Allocation Algorithm (IRCTC Style)**

1. Pending Seat Handling
    - If a row has pending seats, new passengers should fill the gaps first.
    - Example:
        - User 1 books 5 seats → 1 seat is pending.
        - User 2 books 6 seats → They get the next empty row.
        - User 3 books 1 seat → Assigned to the row where 1 seat was pending.
2. Multiple Seat Booking (Group Allocation)
    - If a user books multiple seats, they must be assigned seats in the same row.
    - If not enough seats are available in one row, move to the next completely empty row.

**Concurrency & Real-Time Updates**

- Use MongoDB transactions and Redis locking to prevent double booking.
- Use WebSockets to update seat availability in real time.

**Booking Flow**

- User enters:
    - Number of seats required
    - Passenger details (Name, Age, Gender, etc.)
- System auto-allocates seats based on pending seat logic.
- Booking is confirmed with a PNR Number.

**MongoDB Optimization**

- Indexes for fast seat searches.
- Aggregation pipelines for analytics.

---

## Advanced Analytics Dashboard (MongoDB Aggregations)

**1. Booking Time Trends**

- Find peak booking hours of the day.
- Find trains that get booked first.
- Aggregation Query: Group bookings by train & count per hour.

**2. Group vs. Solo Travelers**

- Find how many users book in groups (3+ seats) vs. solo travelers.
- Aggregation Query: Group by number_of_seats and count.

**3. Booking Lead Time Analysis**

- Find out how early users book their tickets before travel.
- Aggregation Query: Compute the difference between booking_date and travel_date.

---

## Frontend (React, Redux, TailwindCSS, WebSockets)

**Train Search & Booking Form**

- Users can search for trains by Source, Destination, Date.
- Enter number of seats & passenger details (No manual seat selection).

**Live Seat Allocation Status**

- Show total available seats per compartment.
- Real-time updates as seats get booked (via WebSockets).

**Booking Confirmation**

- Users receive a PNR Number after successful booking.

---

**Submission Guidelines**

- GitHub Repo with clean code & README.
- API Documentation (swagger, …etc).

---

Good luck!