**DATABASE SYSTEMS**

**MINI PROJECT**

### **Railway-Reservation-System**

**Team Members-**

**1) 221CS162 - Vasa Taraka Ravindra**

**2) 221CS110 - Anumola Yoga Anil Kumar**

**3) 221CS163 - Vikrant Kumar**

**4) 221CS126 - Gupta Harsh Hemanth**

**Content Page**

**1) Introduction - 3**

**2) Literature Survey - 3**

**3) Objectives - 5**

**4) Features - 5**

**5) Requirement Specifications - 6**

**6) System Design - 6**

**● ER diagram (Entity-Relationship) - 6**

● **Database schema design. - 7**

**● Normalization process. - 7**

**● Tables - 8**

**7) Implementation - 13**

**8) References - 13**

**1.Introduction:**

In today's fast-paced world, where travel is an integral part of our lives, booking train tickets should be a straightforward and stress-free experience. Imagine a booking system that simplifies the entire process, allowing users to effortlessly search for trains, choose their preferred seats, and complete transactions swiftly and securely. This introduction aims to lay the groundwork for the development of such a user-friendly booking system, promising to revolutionize the way people plan their train journeys.

However, amidst the quest for simplicity and efficiency, there lurks a common challenge: the possibility of multiple users trying to book tickets for the same train simultaneously. This scenario can lead to confusion and chaos, potentially resulting in overbooking or double bookings for the same seats. To mitigate this risk and ensure a fair and seamless booking experience for all users, the implementation of a robust concurrency control mechanism becomes essential. This mechanism acts as a guardian, overseeing the booking process and ensuring that each user has a fair chance to secure their tickets without encountering conflicts or discrepancies.

Therefore, creating an easy-to-use booking system, we must not only prioritize user convenience and security but also pay careful attention to the intricate details of concurrency control. By striking a balance between efficiency and fairness, our goal is to deliver a booking platform that not only meets but exceeds the expectations of passengers, making their travel experiences smoother, more enjoyable, and ultimately, more memorable.

**2. Literature Survey**

Web-based railway reservation systems serve as digital platforms that allow users to conveniently book train tickets from the comfort of their homes or on the go. These systems offer a range of features and functionalities aimed at enhancing the booking experience for users.

**Convenience and Accessibility :**

Web-based railway reservation systems have revolutionized ticket booking by offering users the convenience of booking tickets anytime, anywhere. Studies by authors like Smith et al. (2019) have emphasized how these platforms eliminate the need for physical visits to ticket counters, allowing users to book tickets from the comfort of their homes or while traveling.

**24/7 Availability:** reservation systems offer users the convenience of booking tickets at any time of the day or night, without being restricted by the operating hours of traditional ticket counters. This accessibility ensures that users can make their travel arrangements whenever it's most convenient for them, whether they're planning in advance or making last-minute bookings

**Location Independence:** One of the key advantages of online booking platforms is their ability to cater to users regardless of their geographic location. Users can access the reservation system from anywhere with an internet connection, whether they're at home, at work, or on the go. This eliminates the need for users to physically visit ticket counters, saving them time and effort .

**Multi-Device Compatibility:** Web-based reservation systems are designed to be compatible with a wide range of devices, including desktop computers, laptops, smartphones, and tablets. This flexibility allows users to access the platform using their preferred device, further enhancing convenience and accessibility. Additionally, responsive design ensures that the user interface adapts seamlessly to different screen sizes and resolutions

**Allocation of seat done by Indian Railways**

Even by using Web-based reservation, users or passengers can’t choose their wished seat in train, But they can prioritize their seat type and coach number in that specific train. It is because to ensure the safety and stability of a train, the software is designed in such a way that it distributes the load evenly in a train.

For instance, imagine a train having sleeper coaches from s1 to s10 each with 72 seats, when a passenger books a ticket, the software will assign a seat in the middle coach like s5, a middle seat numbered between 30-40 and preferably lower berth. The Railways fills the lower berths before the upper ones to achieve a low center of gravity. This ensures that each coach has an equal load distribution, which is crucial for stability of train.

**Dynamic Pricing of tickets**

Dynamic Pricing is a strategy adopted by businesses to set flexible prices for their product depending on the current market demands. The prices are based on few factors like pricing of competitors, time-based pricing etc. Indian Railways came out with the policy of Dynamic Pricing in 2016. Under this policy, the fares of the railway tickets are to increase by 10 percent after every 10 per cent of berths or seats being sold. This helps the business organization to reduce losses by selling at a low price in a situation of low demand. This pricing method has been adopted with the sole objective of increasing occupancy factors along with maximizing revenues. The point is that the three layers of pricing apply to the same class of travel and the same facilities being provided.

The first layer is aimed at achieving high occupancy by charging only 40-50% of the original fares of the tickets. But a minimum occupancy factor is kept in mind to make the services viable. In the second layer after the 10% cap, the regular fares come into play (it is reached slowly and steadily by increasing 10 per cent fares after each 10 per cent occupancy). The third layer is used when the regular fare is also exhausted. Prices are no longer fixed at this stage and the organization is free to charge any price.

**3. Objectives:**

Accurate Booking and Reservation: Implement validation checks to ensure that user inputs are accurate and adhere to the required format. Develop error handling mechanisms to notify users of any issues during the booking process and provide guidance on how to resolve them.

Implement a good concurrency control mechanism to ensure seamless and fair handling of ticket bookings when two or more users attempt to reserve tickets for the same journey concurrently, preventing overbooking and maintaining data integrity while optimizing user experience.

**4. Features:**

**User Authentication and Authorization**

Provide secure login with role-based access control that allows only authorised users to access system features and do the tasks they are permitted to do, improving system security and user privacy.

**Concurrency Control**

Implement concurrency control mechanisms to manage simultaneous access to and updates of booking-related data by multiple users without conflicts or data corruption, enabling the system to operate smoothly and reliably.

**Data Encryption and Security**

Implement data encryption techniques to protect sensitive booking data stored in the database and enforce access controls to ensure data privacy and security.

**Efficient Search and Filtering :** Develop advanced search and filtering capabilities to allow users to quickly find and compare available options based on their specific travel requirements.

**Booking Management Dashboard:** Provide users with access to a booking management dashboard where they can view and manage their bookings, including upcoming trips, past bookings, and booking history. Allow users to easily retrieve booking details, modify reservations, or initiate cancellation requests from the dashboard.

**5. Requirement Specification**

**Back End:**

**MySQL**: MySQL is a popular open-source relational database management system (RDBMS) that is used to store and manage structured data. You'll use MySQL to create and manage the database schema for storing user information, booking details, train schedules, and other relevant data.

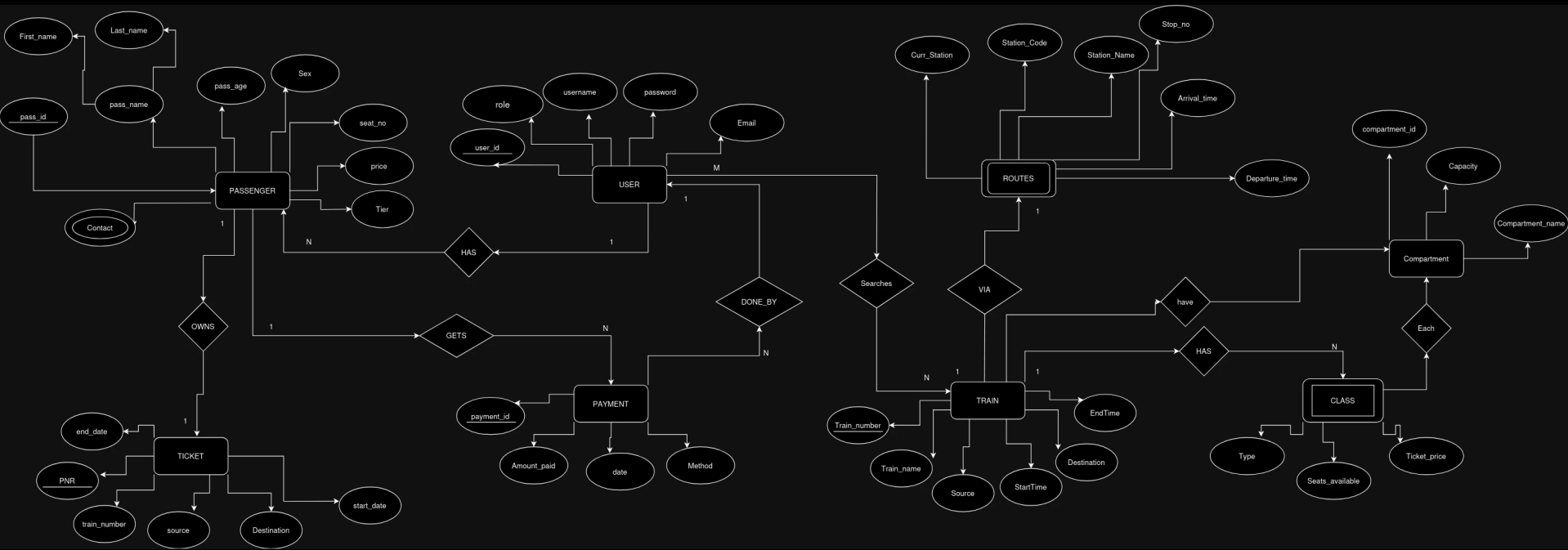
**Node.js**: Node.js is a server-side JavaScript runtime environment that allows you to build scalable and efficient server applications. we'll be using Node.js to handle server-side logic, manage database connections, and implement RESTful APIs for communication between the frontend and backend.

**Front End:**

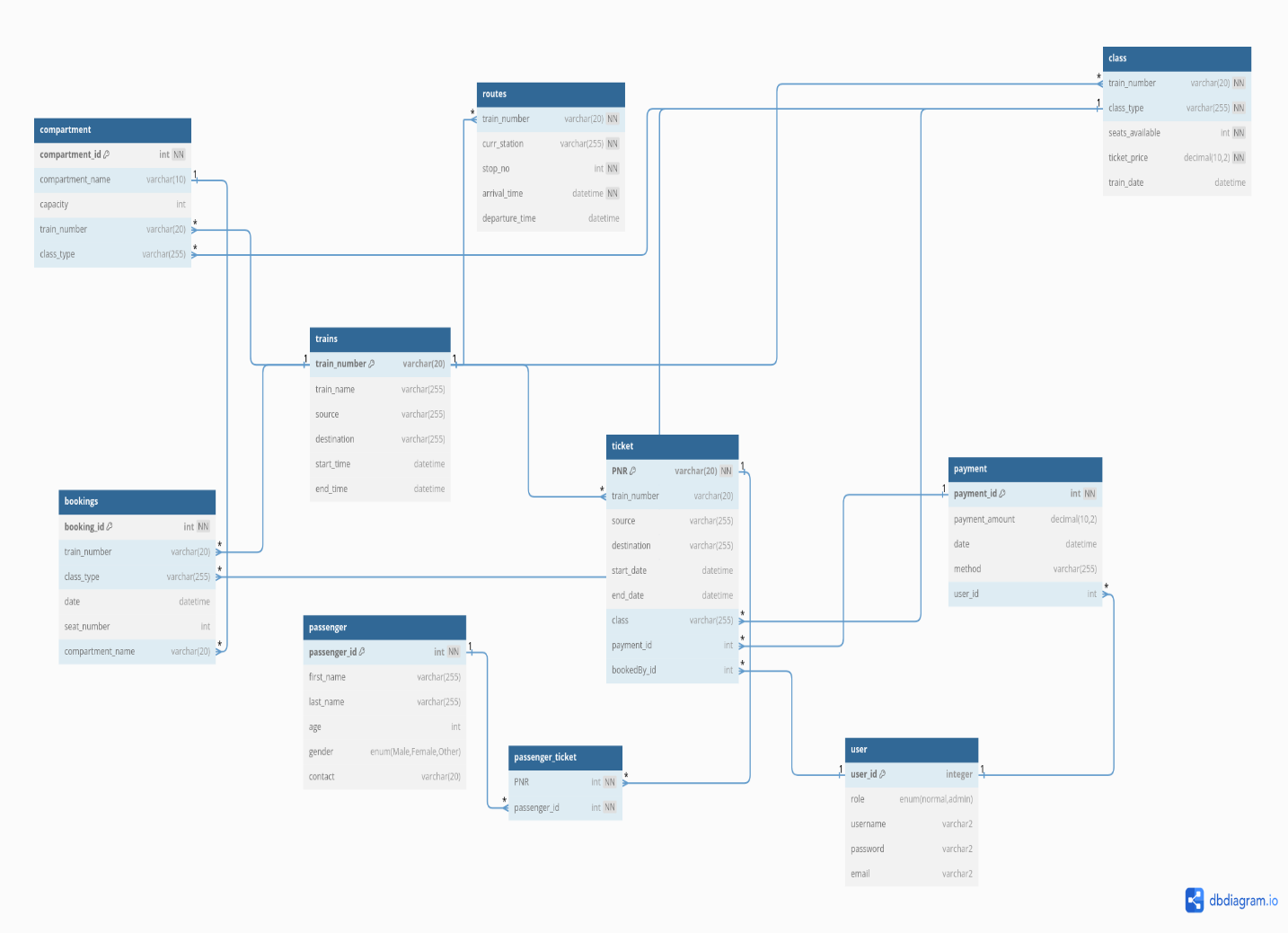
**React.js**: React is a JavaScript library for building user interfaces. We will’be using React to develop the frontend of your web-based reservation system, including designing responsive and interactive user interfaces, managing application state, and handling user interactions.

**SYSTEM DESIGN**

**1.ER DIAGRAM**



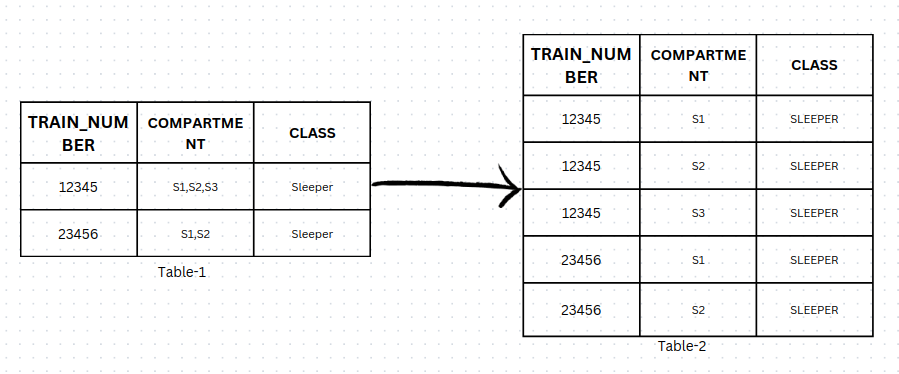
**2.DATABASE SCHEMA DESIGN**

****

**Normalization Process :**

**1NF :**

First Normal Form (1NF) ensures that each attribute in a table contains atomic values, meaning they cannot be further divided. In our database, we achieved 1NF by decomposing a column with multiple values into separate rows, resulting in two normalized tables where each record represents a single entity, facilitating efficient data management and retrieval.

Here’s an example of how it’s done

Each train contain multiple compartments, compartment table can constructed as shown in Table-2 above

All other tables i.e., payment, user, passenger, bookings, .... are in 1st Normal Form

**2NF**

Second Normal Form (2NF) builds upon 1NF by ensuring that non-key attributes are fully functionally dependent on the entire primary key, rather than just part of it. To achieve 2NF, we decompose tables with composite primary keys into separate tables, each with a single primary key, thereby eliminating partial dependencies and reducing data redundancy. This process results in a more organized database structure, enhancing data integrity and simplifying query operations .

For example, consider some tables in the above schema

**Routes:**

All attributes (curr\_station, stop\_no, arrival\_time, departure\_time) are functionally dependent on the composite primary key (train\_number, curr\_station).

**Compartment:**

Compartment\_id uniquely identifies each compartment record, attributes like compartment\_name, capacity, train\_number, and class\_type are functionally dependent on the entire primary key, compartment\_id. There are no partial dependencies present.

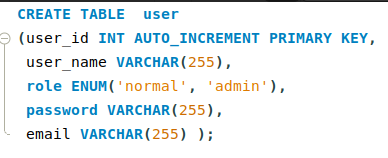
**TABLES**

**1. Users**

Attributes -

* **User\_id** (Primary key): Identifies user uniquely
* **User\_name**
* **Role**: normal user or Admin
* **Password**
* **Email**

Creating table (SQL query)

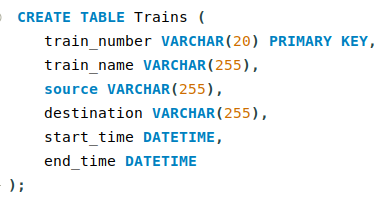


**2. Trains**

Attributes -

* **Train\_number** (Primary key): Number that uniquely identifies train
* **Train\_name**: Name of the train
* **Source**: Station from where train travel starts
* **Destination**: Station from where train travel ends
* **Start\_time**: Time when train travel starts
* **End\_time**: Time when train travel ends

Creating table (SQL query)

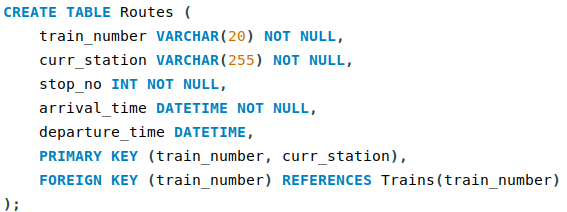


**3. Routes :**

Attributes -

* **Train\_number** (Foreign key) refers to Trains. train number
* **Curr\_station**: current located station
* **Stop\_no**: nth stop of train
* **Arrival\_time**
* **Departure\_time**

Creating table (SQL query)

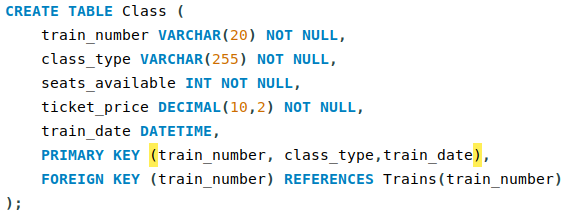


**4. Class :**

Attributes -

* **train\_number** (Foreign key)
* **class\_type**: (Sleeper, 3-Tier, 2-Tier, 1-Tier)
* **seats\_available**: Number of seats available in that class on a particular date
* **ticket\_price**: Price of each ticket in that class
* **train\_date**: Travel date of train

Creating table (SQL query)

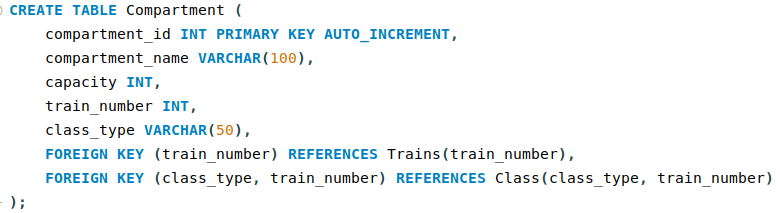


**5. Compartment:**

Attributes -

* **compartment\_id** (Primary key)
* **compartment\_name**
* **capacity**
* **train\_number** (Foreign key)
* **class\_type** (Foreign key)

Creating table (SQL query)

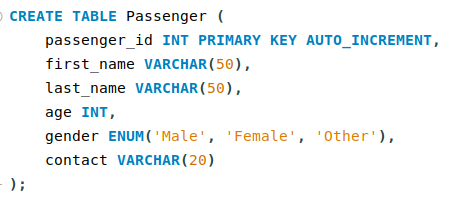


**6. Passenger:**

Attributes -

* **passenger\_id** (Primary key)
* **first\_name**
* **last\_name**
* **age**
* **gender** (Enum: 'Male', 'Female', 'Other')
* **contact**

Creating table (SQL query)

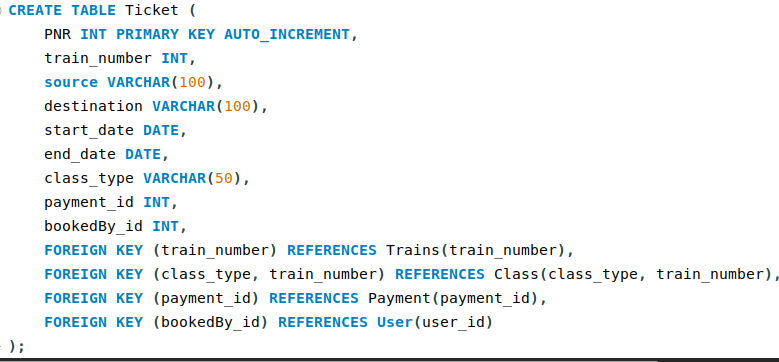


**7. Ticket:**

Attributes -

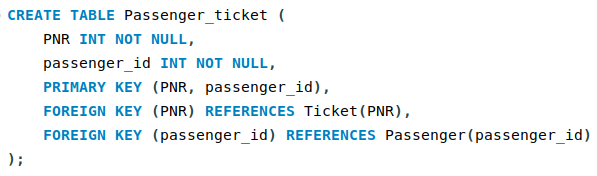
* **PNR (Primary key)**
* **train\_number (Foreign key)**
* **source**
* **destination**
* **start\_date**
* **end\_date**
* **class** (Foreign key)
* **payment\_id** (Foreign key)
* **bookedBy\_id** (Foreign key)

Creating table (SQL query)



**8. Passenger\_ticket**

Attributes -

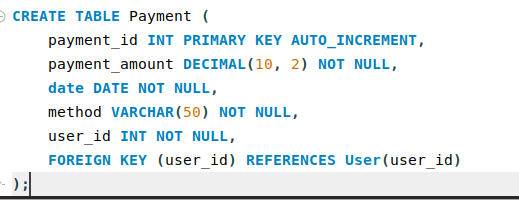
* **PNR** (Foreign key)
* **passenger\_id** (Foreign key)
* Creating table (SQL query) 

**9. Payment:**

Attributes -

* **payment\_id**(Primary key)
* **payment\_amount**
* **date**
* **method**
* **user\_id** (Foreign key)

Creating table (SQL query)

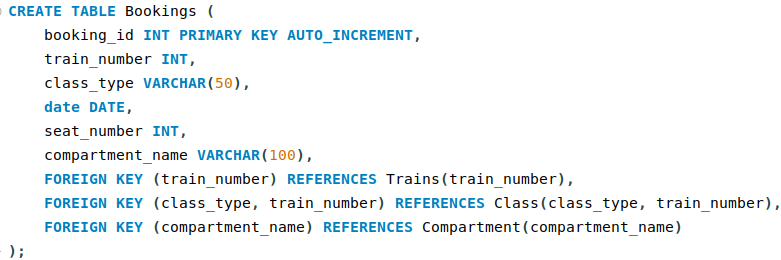


**10. Bookings**

Attributes -

* **booking\_id** (Primary key)
* **train\_number** (Foreign key)
* **class\_type** (Foreign key)
* **date**
* **seat\_number**
* **compartment\_name** (Foreign key)

Creating table (SQL query)



**IMPLEMENTATION**

**React.js**

Setting up react application using Vite front end tooling, and creating components like navigation bar, Search form for searching trains based on source, destination and date of travel and State management in application is done by using Recoil js library and designing interactive web-pages using tailwind css and also performing api calls to the Node server.

**Node.js and MySQL**

Initializing Node.js project for making API and using ‘mysql’ package , we can connect server to database and execute queries. Defining routes for handling CRUD (Create, Read, Update, Delete) operations will be done in the server.

**Creating authentication routes –** can be done in express using jwt (jsonwebtokens) and hashing passwords using bcryptjs package and all the logic of sign in and sign up implemented in these routes.

**Implementing SQL queries -**

By using ‘MySQL’ package for connecting database and server, we can create connection to MySQL database by providing information of host, username, password and database on which queries to be implemented. This package also provides error handling and transaction management.

**References:**

1. https://www.sciencedirect.com/science/article/pii/S1875389212015155?ref=pdf\_download&fr=RR-2&rr=85f1b7815e4a7f37
2. <https://www.news18.com/news/auto/why-irctc-does-not-allow-seat-selection-option-to-its-passengers-5369077.html>
3. <https://link.springer.com/article/10.1057/s41272-023-00450-w>
4. <https://www.altexsoft.com/blog/railway-reservation-system/>
5. https://www.fabhotels.com/blog/indian-railways/railway-reservation/