NATIONAL INSTITUTE OF TECHNOLOGY DELHI

DBMS PROJECT

RAIL-NEXUS: Railway reservation system

Ajay Kumawat(241210011)

Aryan Srivastava(241210025)

Deepak Kumar(241210034)

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to **Dr. Shelly Sachdeva**, our course instructor, for giving us the opportunity to work on this project, *Rail-Nexus: Railway Reservation System*. Her constant guidance, encouragement and support throughout the course helped us understand DBMS concepts in a much deeper and practical way.

We are also thankful to our Teaching Assistants for their help and timely suggestions. Their inputs made the project more structured and easier to complete.

Working on this project has been a valuable experience for our entire group. Through Rail-Nexus, we learned how real reservation systems work, how large amounts of data are managed, and how different database components connect with each other. The project helped us improve our SQL skills, understand ER modelling, and gain hands-on experience with relational databases.

We would also like to thank our institute and faculty members for providing the resources and environment that made this project possible. Completing Rail-Nexus has been an enriching learning experience for all of us.

CONTENTS

CASE STUDY

1. A passenger starts the process by logging into the Rail Nexus system and selecting the source, destination, date of journey, and preferred class of travel. The system fetches available train options from the database and displays details like timings, seat availability, and fare. Once the passenger chooses a train, the booking request is verified against real-time data, and the seat is locked temporarily until the payment is completed.
2. Seat allocation is managed dynamically. The system checks whether a confirmed seat is available; if not, it assigns RAC (Reservation Against Cancellation) or waitlist status. The database updates instantly to prevent double-booking and reflects the passenger's current booking status.
3. After seat selection, the passenger is directed to the payment gateway. Rail Nexus supports multiple payment methods including credit/debit cards, net banking, UPI, and digital wallets. Each transaction is securely logged in the database, ensuring transparency and traceability of payments. In case of cancellation, refund records are also updated with proper linkage to the original booking.
4. Train schedules are centrally managed in the database, containing details of arrivals, departures, halts, and platform assignments. Administrators can modify these records whenever changes occur, and passengers booking tickets are always shown the latest, updated schedule.
5. Station details such as codes, names, and locations are stored systematically, enabling accurate mapping of routes. This ensures that booking queries and ticket generation can handle multi-stop and long-distance journeys without conflicts.
6. Passenger accounts in the system maintain personal information, travel history, booking records, and cancellation details. This not only improves user experience by providing quick access to past data but also helps administrators verify information for audits and reports.
7. The database is designed with relational integrity using primary keys, foreign keys, and constraints, which ensures that records are accurate, unique, and

consistent. This structure minimizes redundancy and improves efficiency in managing large volumes of railway data.

8. Administrators can generate detailed reports on ticket sales, passenger flow, peak seasons, and revenue statistics. These insights support better decision-making, resource allocation, and service improvements, making Rail Nexus a reliable and scalable system for railway management.
9. Rail Nexus also incorporates basic security mechanisms, such as authentication and role-based access, to ensure that sensitive user information and administrative data remain protected.
10. Backup and recovery mechanisms help maintain system reliability. In the event of a server failure or unexpected shutdown, essential booking and payment data can be restored without impacting users.
11. The system is designed to scale effectively, ensuring smooth performance even during peak booking hours. Proper indexing and optimized queries help manage thousands of concurrent requests efficiently.

INTRODUCTION

Rail Nexus is a database management system project designed to streamline the operations of railway reservation and management. The system focuses on organizing and handling large volumes of data related to trains, passengers, stations, ticket bookings, schedules, and payments in a structured and efficient manner. By using concepts of relational databases, the project ensures accurate storage, quick retrieval, and easy management of information while minimizing redundancy and maintaining data integrity.

This project also emphasizes user convenience by providing a reliable and scalable backend for tasks such as booking tickets, checking train availability, managing seat allocations, and tracking journey details. Rail Nexus not only demonstrates the practical application of DBMS concepts like entity–relationship modelling, normalization, and SQL queries, but also reflects how such systems form the backbone of real-world railway networks. It highlights the importance of databases in creating robust, secure, and user-friendly solutions for large-scale transportation services.

SYSTEM REQUIREMENTS

Software Requirements:

Bootstrap Studio for designing the front-end.

- HTML, CSS and JS for developing the front-end.
- MySQL as a back-end query language.
- PHP shall be used as a scripting language.
- Web Browser (Chrome/Firefox, etc.) and XAMPP as a web host on local server.

Hardware Requirements:

i) Operating System: Windows 7/8/8.1/10.

ii) Memory (RAM): 1 GB or above.

iii) Hard Disk Space: at least 200 MB.

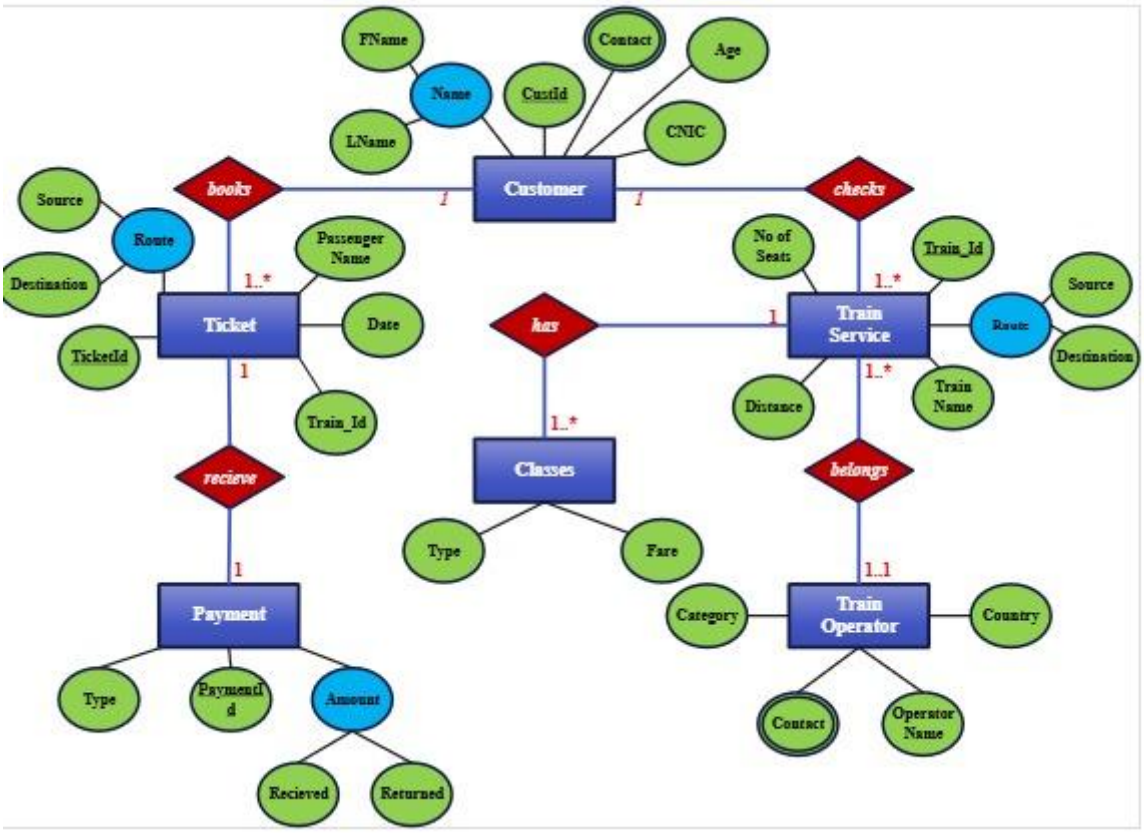
iv) Processor: Intel i5 processor.

ASSUMPTIONS

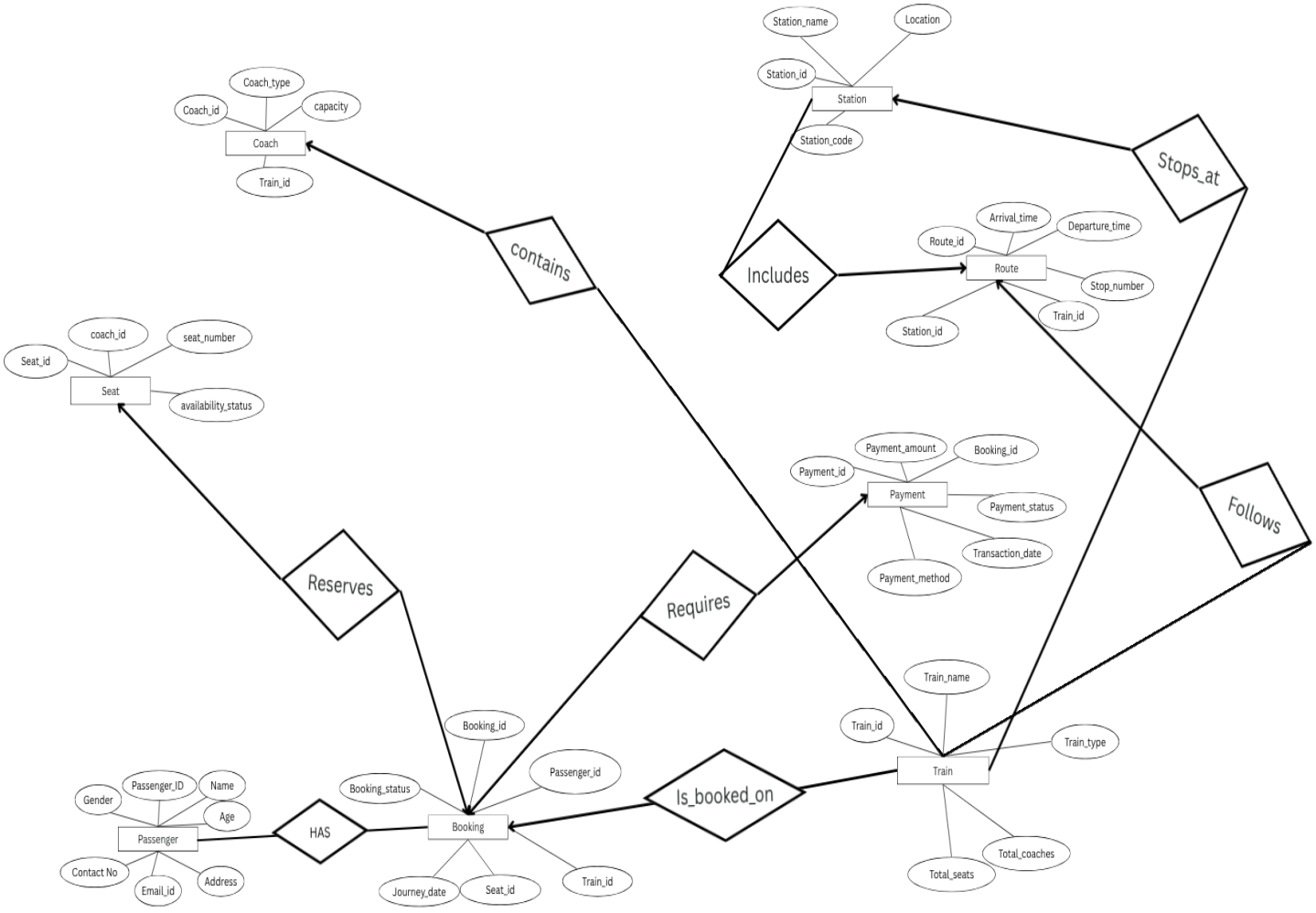
We have considered the following assumptions while preparing our project to maintain clarity and reduce possible errors:

1. Each passenger has a **unique Passenger ID**, which is used to identify bookings, payments, and travel history.
2. A passenger is assumed to have **only one active contact number and email** registered in the system at a time.
3. Each train has a **fixed set of coaches and seats**, and the capacity does not change during the journey.
4. A seat can be assigned to **only one passenger** for a particular train journey and date.
5. Train routes are assumed to be **predefined** and do not change during the booking process.
6. A single booking can include **only one passenger** (for simplicity of schema). Multi-passenger booking is treated as separate bookings.
7. Payment for a ticket must be completed within the allowed time window; otherwise, the seat is **released automatically**.
8. RAC and Waitlist positions follow a **first-come-first-served** approach.
9. Every station has a **unique station code** for mapping and identification.
10. Cancellation refunds are assumed to follow a **standard fixed rule**, and partial refunds or special cases are not handled.

ER Diagram (Previous)

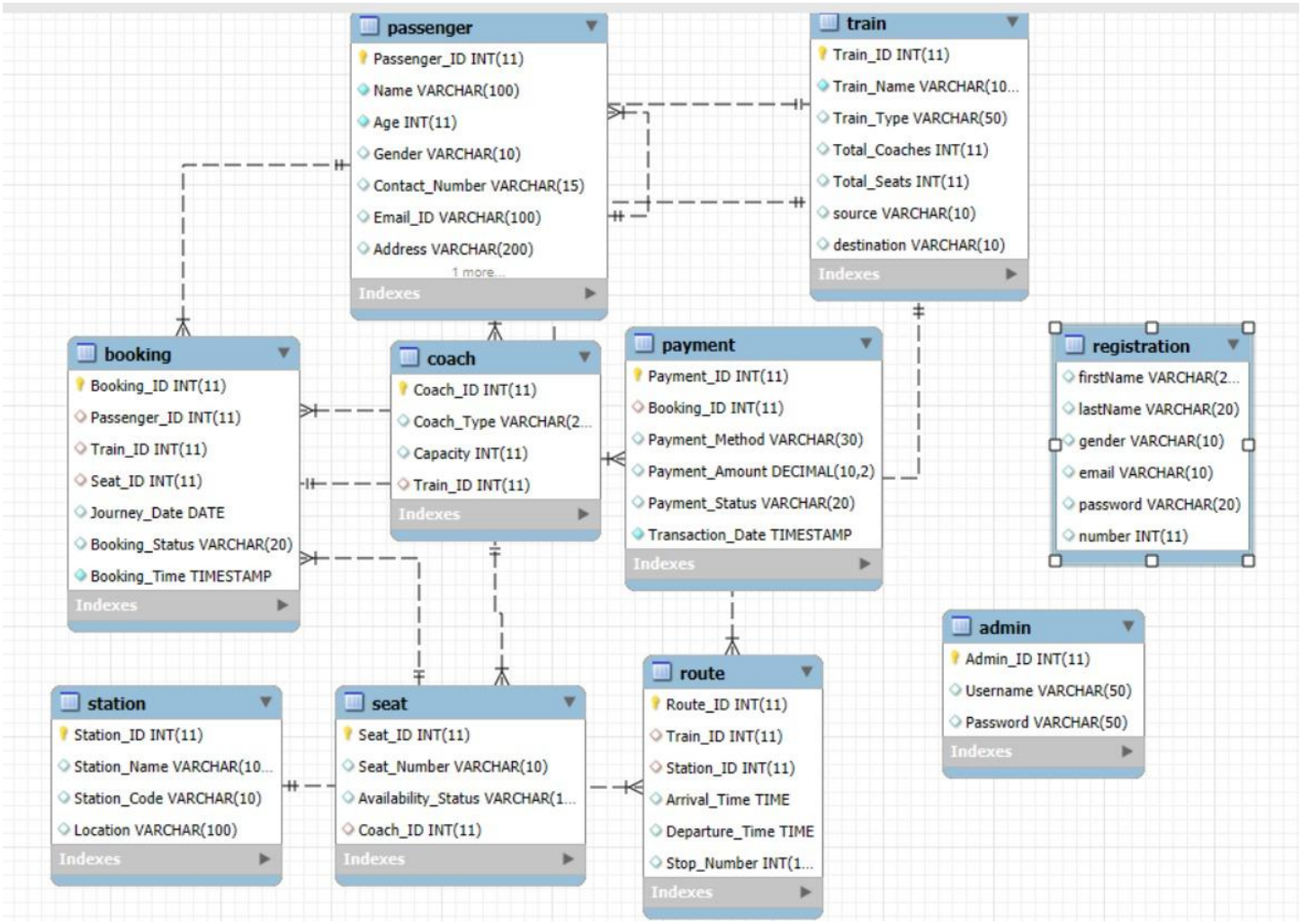


ER Diagram (Updated)



MAPPING FROM ER MODEL TO R MODEL

Relational Schema



DATA DICTIONARY

1 admin

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
Admin_ID	int(11)		No		auto_increment			
Username	varchar(50)		Yes	NULL				
Password	varchar(50)		Yes	NULL				

2 booking

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
Booking_ID	int(11)		No		auto_increment			
Passenger_ID	int(11)		Yes	NULL		-> passenger.Passenger_ID ON UPDATE RESTRICT ON DELETE CASCADE		
Train_ID	int(11)		Yes	NULL		-> train.Train_ID ON UPDATE RESTRICT ON DELETE CASCADE		
Seat_ID	int(11)		Yes	NULL		-> seat.Seat_ID ON UPDATE RESTRICT ON DELETE SET_NULL		
Journey_Date	date		Yes	NULL				
Booking_Status	varchar(20)		Yes	NULL				
Booking_Time	timestamp		No	current_timestamp()				

3 coach

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
Coach_ID	int(11)		No		auto_increment			
Coach_Type	varchar(20)		Yes	NULL				
Capacity	int(11)		Yes	NULL				
Train_ID	int(11)		Yes	NULL		-> train.Train_ID ON UPDATE RESTRICT ON DELETE CASCADE		

4 passenger

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
Passenger_ID	int(11)		No		auto_increment			
Name	varchar(100)		No					
Age	int(11)		No					
Gender	varchar(10)		Yes	NULL				
Contact_Number	varchar(15)		Yes	NULL				
Email_ID	varchar(100)		Yes	NULL				
Address	varchar(200)		Yes	NULL				

5 payment

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
Payment_ID	int(11)		No		auto_increment			
Booking_ID	int(11)		Yes	NULL		-> booking.Booking_ID ON UPDATE RESTRICT ON DELETE CASCADE		
Payment_Method	varchar(30)		Yes	NULL				
Payment_Amount	decimal(10,2)		Yes	NULL				
Payment_Status	varchar(20)		Yes	NULL				
Transaction_Date	timestamp		No	current_timestamp()				

6 route

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
Route_ID	int(11)		No		auto_increment			
Train_ID	int(11)		Yes	NULL		-> train.Train_ID ON UPDATE RESTRICT ON DELETE CASCADE		
Station_ID	int(11)		Yes	NULL		-> station.Station_ID ON UPDATE RESTRICT ON DELETE CASCADE		
Arrival_Time	time		Yes	NULL				
Departure_Time	time		Yes	NULL				
Stop_Number	int(11)		Yes	NULL				

7 seat

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
Seat_ID	int(11)		No		auto_increment			
Seat_Number	varchar(10)		Yes	NULL				
Availability_Status	varchar(15)		Yes	NULL				
Coach_ID	int(11)		Yes	NULL		-> coach.Coach_ID ON UPDATE RESTRICT ON DELETE CASCADE		

8 station

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
Station_ID	int(11)		No		auto_increment			
Station_Name	varchar(100)		Yes	NULL				
Station_Code	varchar(10)		Yes	NULL				
Location	varchar(100)		Yes	NULL				

9 train

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
Train_ID	int(11)		No		auto_increment			
Train_Name	varchar(100)		No					
Train_Type	varchar(50)		Yes	NULL				
Total_Coaches	int(11)		Yes	NULL				
Total_Seats	int(11)		Yes	NULL				
source	varchar(10)		Yes	NULL				
destination	varchar(10)		Yes	NULL				

SQL Queries

```
create database railway_reservation;

use railway_reservation;

CREATE TABLE Admin(
    Admin_ID INT AUTO_INCREMENT PRIMARY KEY,
    Username VARCHAR(50),
    Password VARCHAR(50)
);


CREATE TABLE Passenger (
    Passenger_ID INT AUTO_INCREMENT PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    Age INT NOT NULL,
    Gender VARCHAR(10),
    Contact_Number VARCHAR(15),
    Email_ID VARCHAR(100),
    Address VARCHAR(200)
);


CREATE TABLE Train (
    Train_ID INT AUTO_INCREMENT PRIMARY KEY,
    Train_Name VARCHAR(100) NOT NULL,
    Train_Type VARCHAR(50),
    Total_Coaches INT,
```

```
    Total_Seats INT
);
alter table Train
add column source varchar(10);
alter table Train
add column destination varchar(10);

CREATE TABLE Coach (
    Coach_ID INT AUTO_INCREMENT PRIMARY KEY,
    Coach_Type VARCHAR(20),
    Capacity INT,
    Train_ID INT,
    FOREIGN KEY (Train_ID) REFERENCES Train(Train_ID)
        ON DELETE CASCADE
);

CREATE TABLE Seat (
    Seat_ID INT AUTO_INCREMENT PRIMARY KEY,
    Seat_Number VARCHAR(10),
    Availability_Status VARCHAR(15), -- Available / Booked
    Coach_ID INT,
    FOREIGN KEY (Coach_ID) REFERENCES Coach(Coach_ID)
        ON DELETE CASCADE
);
```

```
CREATE TABLE Station (
    Station_ID INT AUTO_INCREMENT PRIMARY KEY,
    Station_Name VARCHAR(100),
    Station_Code VARCHAR(10),
    Location VARCHAR(100)
);

CREATE TABLE Route (
    Route_ID INT AUTO_INCREMENT PRIMARY KEY,
    Train_ID INT,
    Station_ID INT,
    Arrival_Time TIME,
    Departure_Time TIME,
    Stop_Number INT,
    FOREIGN KEY (Train_ID) REFERENCES Train(Train_ID)
        ON DELETE CASCADE,
    FOREIGN KEY (Station_ID) REFERENCES Station(Station_ID)
        ON DELETE CASCADE
);

CREATE TABLE Booking (
    Booking_ID INT AUTO_INCREMENT PRIMARY KEY,
    Passenger_ID INT,
```

```
    Train_ID INT,
    Seat_ID INT,
    Journey_Date DATE,
    Booking_Status VARCHAR(20),   -- Confirmed / Pending / Cancelled
    Booking_Time TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (Passenger_ID) REFERENCES Passenger(Passenger_ID)
        ON DELETE CASCADE,
    FOREIGN KEY (Train_ID) REFERENCES Train(Train_ID)
        ON DELETE CASCADE,
    FOREIGN KEY (Seat_ID) REFERENCES Seat(Seat_ID)
        ON DELETE SET NULL
);

CREATE TABLE Payment (
    Payment_ID INT AUTO_INCREMENT PRIMARY KEY,
    Booking_ID INT,
    Payment_Method VARCHAR(30),  -- UPI / Card / NetBanking / Wallet
    Payment_Amount DECIMAL(10,2),
    Payment_Status VARCHAR(20),  -- Paid / Failed / Pending
    Transaction_Date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (Booking_ID) REFERENCES Booking(Booking_ID)
        ON DELETE CASCADE
);
INSERT INTO Station (Station_Name, Station_Code, Location) VALUES
```

('New Delhi', 'NDLS', 'Delhi'),

('Mumbai Central', 'MMCT', 'Mumbai'),

('Howrah Junction', 'HWH', 'Kolkata'),

('Chennai Central', 'MAS', 'Chennai'),

('Bengaluru City Junction', 'SBC', 'Bengaluru'),

('Hyderabad Deccan', 'HYB', 'Hyderabad'),

('Kanpur Central', 'CNB', 'Kanpur'),

('Lucknow Charbagh', 'LKO', 'Lucknow'),

('Patna Junction', 'PNBE', 'Patna'),

('Jaipur Junction', 'JP', 'Jaipur'),

('Ahmedabad Junction', 'ADI', 'Ahmedabad'),

('Pune Junction', 'PUNE', 'Pune'),

('Varanasi Junction', 'BSB', 'Varanasi'),

('Gorakhpur Junction', 'GKP', 'Gorakhpur'),

('Bhopal Junction', 'BPL', 'Bhopal'),

('Nagpur Junction', 'NGP', 'Nagpur'),

('Surat', 'ST', 'Surat'),

('Coimbatore Junction', 'CBE', 'Coimbatore'),

('Visakhapatnam Junction', 'VSKP', 'Visakhapatnam'),

('Thiruvananthapuram Central', 'TVC', 'Kerala');

INSERT INTO Train (Train_Name, Train_Type, Total_Coaches, Total_Seats, Source, Destination) VALUES

('Vande Bharat Express', 'Superfast', 16, 960, 'New Delhi', 'Varanasi'),

('Rajdhani Express', 'Rajdhani', 20, 1200, 'New Delhi', 'Mumbai Central'),

('Shatabdi Express', 'Shatabdi', 18, 1080, 'Howrah Junction', 'New Jalpaiguri'),

('Tejas Express', 'Superfast', 16, 900, 'Mumbai Central', 'Ahmedabad Junction'),

('Garib Rath Express', 'Express', 18, 1080, 'Lucknow', 'New Delhi'),

('Humsafar Express', 'Superfast', 20, 1180, 'Agartala', 'Bengaluru'),

('Jan Shatabdi Express', 'Express', 16, 960, 'Chennai Central', 'Mangaluru'),

('Duronto Express', 'Superfast', 19, 1150, 'Sealdah', 'Puri'),

('Sampark Kranti Express', 'Superfast', 20, 1250, 'Mumbai Central', 'Chandigarh'),

('Intercity Express', 'Express', 14, 840, 'Chennai Central', 'Coimbatore'),

('Howrah Mail', 'Express', 20, 1200, 'Mumbai CST', 'Howrah Junction'),

('Chennai Express', 'Superfast', 18, 1100, 'Mumbai CST', 'Chennai Central'),

('Karnataka Express', 'Express', 22, 1300, 'New Delhi', 'Bengaluru'),

('Jhelum Express', 'Express', 20, 1200, 'Pune', 'Jammu Tawi'),

('Gatimaan Express', 'Superfast', 12, 720, 'New Delhi', 'Jhansi'),

('Double Decker Express', 'Express', 14, 850, 'Jaipur', 'Delhi Sarai Rohilla'),

('Antyodaya Express', 'Express', 22, 1300, 'Chennai Central', 'Tirunelveli'),

('Dehradun Express', 'Express', 18, 1000, 'Mumbai CST', 'Dehradun'),

('Lucknow Mail', 'Superfast', 20, 1200, 'Lucknow', 'New Delhi'),

('Gorakhpur Express', 'Express', 17, 1020, 'Gorakhpur', 'Mumbai CST');

select*from train;

select* from passenger;

select*from booking;

select *from payment;

POPULATED TABLE

←T→	Passenger_ID	Name	Age	Gender	Contact_Number	Email_ID	Address
Edit Copy Delete	1	deepak	19	Male	8299353112	241210034@nitdelhi.ac.in	nitdelhi
Edit Copy Delete	2	Aryan Srivastava	20	Male	8299353112	srivastavaaryan3111@gmail.com	nitdelhi
Edit Copy Delete	3	ROHIT	25	Male	8402184821	2412100102@nitdelhi.ac.in	nitdelhi
Edit Copy Delete	4	rahul	20	Male	1234567890	abc@gmail.com	xyz
Edit Copy Delete	5	Aarav Mehta	22	Male	9876543210	aarav.mehta@gmail.com	Delhi
Edit Copy Delete	6	Sanya Kapoor	20	Female	9812345678	sanya.kapoor@gmail.com	Mumbai
Edit Copy Delete	7	Rohan Verma	25	Male	9123456780	rohan.v@gmail.com	Chennai
Edit Copy Delete	8	Ishita Sharma	19	Female	9871203345	ishita.sharma@hotmail.com	Kolkata
Edit Copy Delete	9	Kunal Singh	23	Male	9988776655	kunalsingh@gmail.com	Hyderabad
Edit Copy Delete	10	Priya Nair	21	Female	9822456711	priya.nair@gmail.com	Kerala
Edit Copy Delete	11	Aditya Khanna	24	Male	8877665544	aditya.khanna@yahoo.com	Punjab
Edit Copy Delete	12	Shreya Mittal	20	Female	9123465788	shreya.mittal@gmail.com	Jaipur
Edit Copy Delete	13	Vikram Chauhan	26	Male	9988996655	vikramc@gmail.com	Ahmedabad
Edit Copy Delete	14	Neha Gupta	23	Female	9012345566	neha.gupta@gmail.com	Noida
Edit Copy Delete	15	Kabir Malhotra	22	Male	9765432109	kabir.malhotra@gmail.com	Gurgaon
Edit Copy Delete	16	Sakshi Jain	19	Female	9876543321	sakshi.jain@gmail.com	Bhopal
Edit Copy Delete	17	Arjun Reddy	24	Male	9123109876	arjun.reddy@gmail.com	Bangalore
Edit Copy Delete	18	Megha Rao	21	Female	9765201488	megha.rao@gmail.com	Chennai
Edit Copy Delete	19	Yash Thakur	22	Male	9988774411	yash.thakur@gmail.com	Lucknow
Edit Copy Delete	20	xyz	35	male	1234567890	abc@gmail.com	abc112

Edit Copy Delete	26	Vande Bharat Express	Superfast	16	960	New Delhi	Varanasi
Edit Copy Delete	27	Rajdhani Express	Rajdhani	20	1200	New Delhi	Mumbai Cen
Edit Copy Delete	28	Shatabdi Express	Shatabdi	18	1080	Howrah Jun	New Jalpai
Edit Copy Delete	29	Tejas Express	Superfast	16	900	Mumbai Cen	Ahmedabad
Edit Copy Delete	30	Garib Rath Express	Express	18	1080	Lucknow	New Delhi
Edit Copy Delete	31	Humsafar Express	Superfast	20	1180	Agartala	Bengaluru
Edit Copy Delete	32	Jan Shatabdi Express	Express	16	960	Chennai Ce	Mangaluru
Edit Copy Delete	33	Duronto Express	Superfast	19	1150	Sealdah	Puri
Edit Copy Delete	34	Sampark Kranti Express	Superfast	20	1250	Mumbai Cen	Chandigarh
Edit Copy Delete	35	Intercity Express	Express	14	840	Chennai Ce	Coimbatore
Edit Copy Delete	36	Howrah Mail	Express	20	1200	Mumbai CST	Howrah Jun
Edit Copy Delete	37	Chennai Express	Superfast	18	1100	Mumbai CST	Chennai Ce
Edit Copy Delete	38	Karnataka Express	Express	22	1300	New Delhi	Bengaluru
Edit Copy Delete	39	Jhelum Express	Express	20	1200	Pune	Jammu Tawi
Edit Copy Delete	40	Gatimaan Express	Superfast	12	720	New Delhi	Jhansi
Edit Copy Delete	41	Double Decker Express	Express	14	850	Jaipur	Delhi Sara
Edit Copy Delete	42	Antyodaya Express	Express	22	1300	Chennai Ce	Tirunelvel
Edit Copy Delete	43	Dehradun Express	Express	18	1000	Mumbai CST	Dehradun
Edit Copy Delete	44	Lucknow Mail	Superfast	20	1200	Lucknow	New Delhi

DB CONNECTIVITY

The database has been connected to the front-end user interface using **PHP** as the scripting language. PHP acts as the bridge between the user interface and the MySQL database, allowing the system to perform operations such as inserting data, retrieving information, updating records, and validating user inputs.

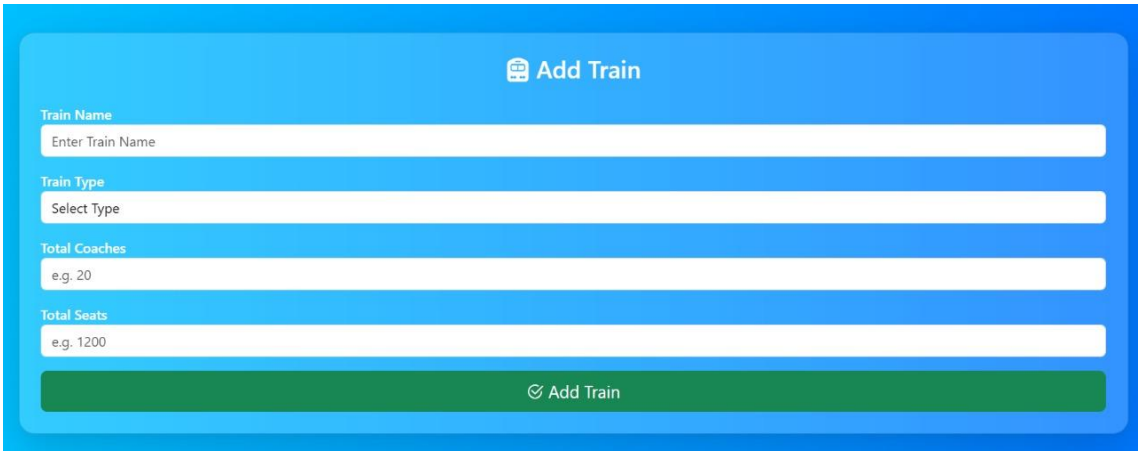
To establish the connection, the required database credentials such as hostname, username, password, and database name are provided within the PHP script. Once the connection is successfully created, the system can communicate with the database and execute SQL queries whenever needed.

The code for establishing the connection is as follows:

```
<?php


$servername = "localhost";

$username = "root";

$password = "";

$database = "railway_reservation";


$conn = new mysqli($servername, $username, $password, $database);


if ($conn->connect_error) {

    die("Connection failed: " . $conn->connect_error);

}

?>
```

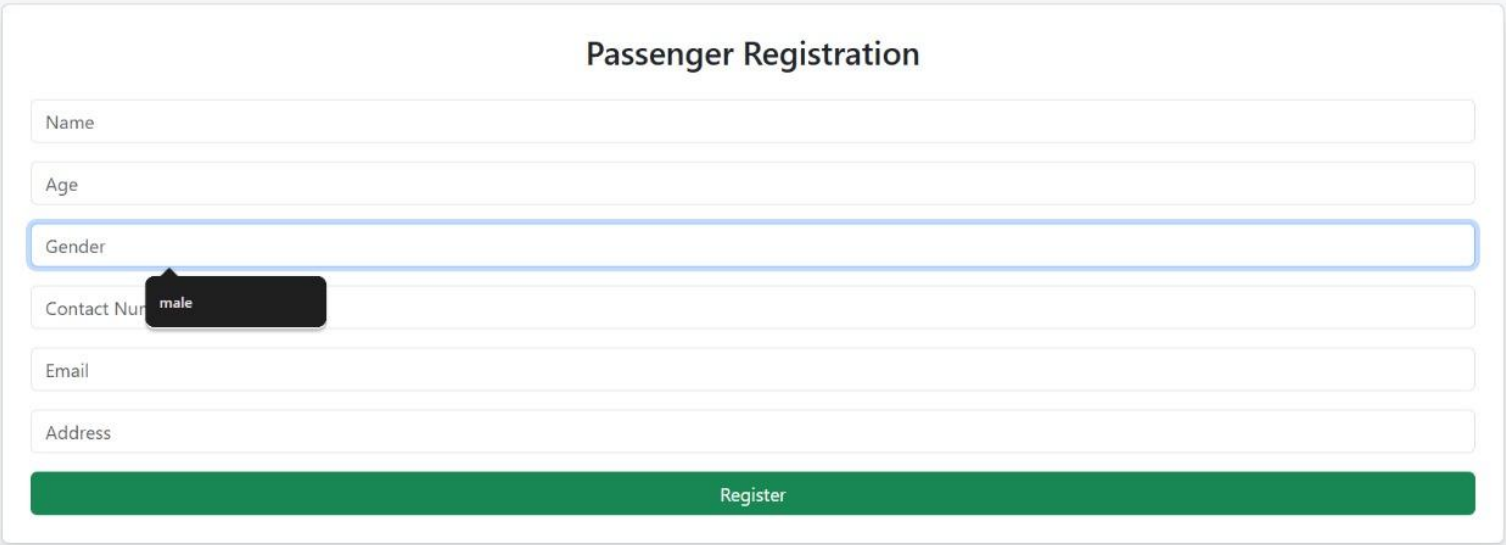
The above code establishes the connection between the backend (database) and the front-end (user interface). Once connected, PHP enables the system to perform different operations such as booking tickets, storing passenger details, validating login credentials, and displaying results dynamically.

FUNCTIONALITIES



Train List

ID	Train Name	Type	Coaches	Seats	Action
2	vande bharat	Express	25	350	Delete
3	Rajdhani Express	Superfast	25	200	Delete
4	Shatabdi Express	Superfast	30	300	Delete
6	Vande Bharat Express	Superfast	16	960	Delete
7	Rajdhani Express	Rajdhani	20	1200	Delete
8	Shatabdi Express	Shatabdi	18	1080	Delete
9	Duronto Express	Superfast	19	1150	Delete



Add Train Route

Select Train

Select Train

Select Station

Select Station

Arrival Time

--:--

Departure Time

--:--

Stop Number

Add Route

Add Coach

Coach Type

AC

Capacity

Select Train

Select Train

Add Coach

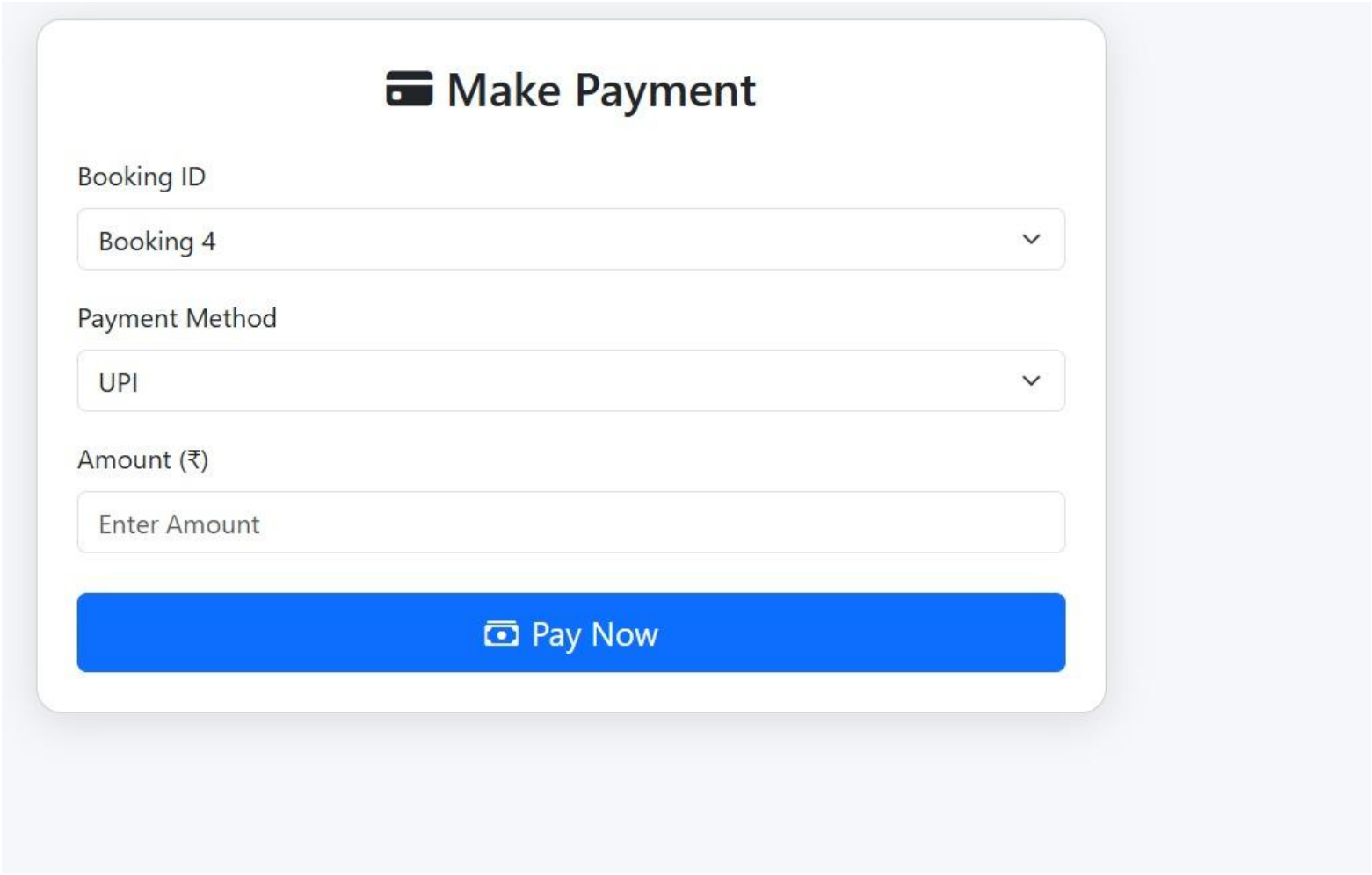
Add Seat

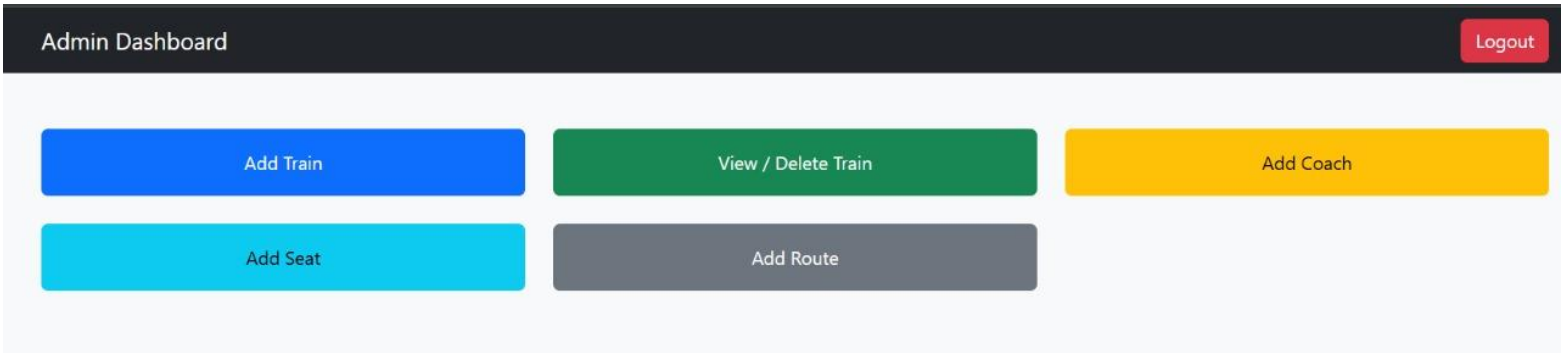
Seat Number

Select Coach

Select Coach

Add Seat


Make Payment
Booking ID
Booking 4
Payment Method
UPI
Amount (₹)
Enter Amount
Pay Now


Admin Dashboard
Logout
Add Train
View / Delete Train
Add Coach
Add Seat
Add Route

13. Bibliography

- Database System Concepts – Silberschatz
- MySQL & PHP Documentation
- Indian Railways Official Data