RAILWAY RESERVATION SYSTEM
USING
DATABASE MANAGEMENT SYSTEM

by

DIVAKAR K G - 71382202032

PROJECT BASED LEARNING

report submitted in partial fulfillment of the
requirements for the Degree of
Bachelor of Engineering in
Computer Science and Engineering

SECOND YEAR

Sri Ramakrishna Institute of Technology

Coimbatore – 641010

MAY 2024

APPROVAL AND DECLARATION

This project report titled "Railway Reservation System Using Database Management System" was prepared and submitted by **Divakar K G (71382202032)** and has been found satisfactory in terms of scope, quality and presentation as partial fulfillment of the requirement for the **Bachelor of Engineering (Computer Science and Engineering)** in Sri Ramakrishna Institute of Technology, Coimbatore (SRIT).

Checked and Approved by

_____

Ms.K.Saranya

Assistant Professor

Project Supervisor

Department of Computer Science and Engineering
Sri Ramakrishna Institute of Technology
Coimbatore -10

May 2024

TABLE OF CONTENTS

LIST OF FIGURES

.

LIST OF ABBREVIATIONS

SQL	Structured Query Language
DBMS	Database Management System
ER	Entity Relationship
API	Application Programming Interface
RRS	Railway Reservation System

CHAPTER 1

AIM

The aim of the Railway Reservation System using DBMS is to create a comprehensive and efficient platform for managing the booking, reservation, and cancellation of train tickets. This system aims to streamline the process of ticket reservation, reduce manual errors, provide accurate information to passengers, and optimize resource utilization for railway authorities.

1. **Automated Ticket Booking:** Develop an intuitive interface for searching schedules, checking availability, and booking tickets seamlessly.
2. **Real-Time Information:** Integrate APIs for up-to-date schedules, availability, and fares to empower informed decisions.
3. **User Management:** Securely handle user authentication, registration, profile updates, and booking history.
4. **Seat Reservation:** Enable seat selection based on preferences, availability, and coach allocation.
5. **Cancellation and Refund:** Implement easy cancellation and refund processes adhering to predefined criteria.
6. **Payment Gateway Integration:** Integrate secure payment gateways for reliable online transactions.
7. **Reporting and Analytics:** Generate insightful reports on sales, revenue,demographics and routes for informed decisions.
8. **Admin Dashboard:** Provide administrators with a centralized tool for configuration, monitoring, issue resolution, and administrative tasks.
9. **Scalability and Performance:** Design for scalability and optimal performance to handle peak loads effectively.
10. **Data Security and Privacy:** Ensure robust security measures to protect sensitive data and comply with regulations.

CHAPTER 2

PROBLEM STATEMENT EXPLANATION

2.1 Requirements

Operating System	:	Windows 7 & above
Database Used	:	Oracle 11g Express Edition
Simulator Tool	:	SQL Command Line 8.0
Language	:	Structured Query Language

2.2 Problem Statement Explanation

The existing railway reservation system lacks efficiency and user-friendliness, resulting in inconvenience for passengers and inefficiency for railway authorities. Passengers face challenges in booking tickets, checking seat availability, and managing their bookings, while railway authorities struggle with manual processes and limited data management capabilities. Additionally, there is a need for better integration of payment processing and ticketing systems to streamline transactions and improve revenue management.

Solution:

To address these challenges, we propose the development of a comprehensive Railway Reservation System using a Database Management System (DBMS). The system will offer the following features and functionalities:

Improved User Experience: Passengers will enjoy a seamless booking experience with access to real-time information and secure online payments.

Operational Efficiency: Railway authorities will benefit from streamlined processes, automated ticketing, and better data management capabilities.

Overall, the proposed Railway Reservation System will modernize ticketing processes, improve passenger satisfaction, and contribute to the efficient operation of the railway transportation system.

CHAPTER 3

E-R DIAGRAM

3.1 E-R Diagram of the Proposed System

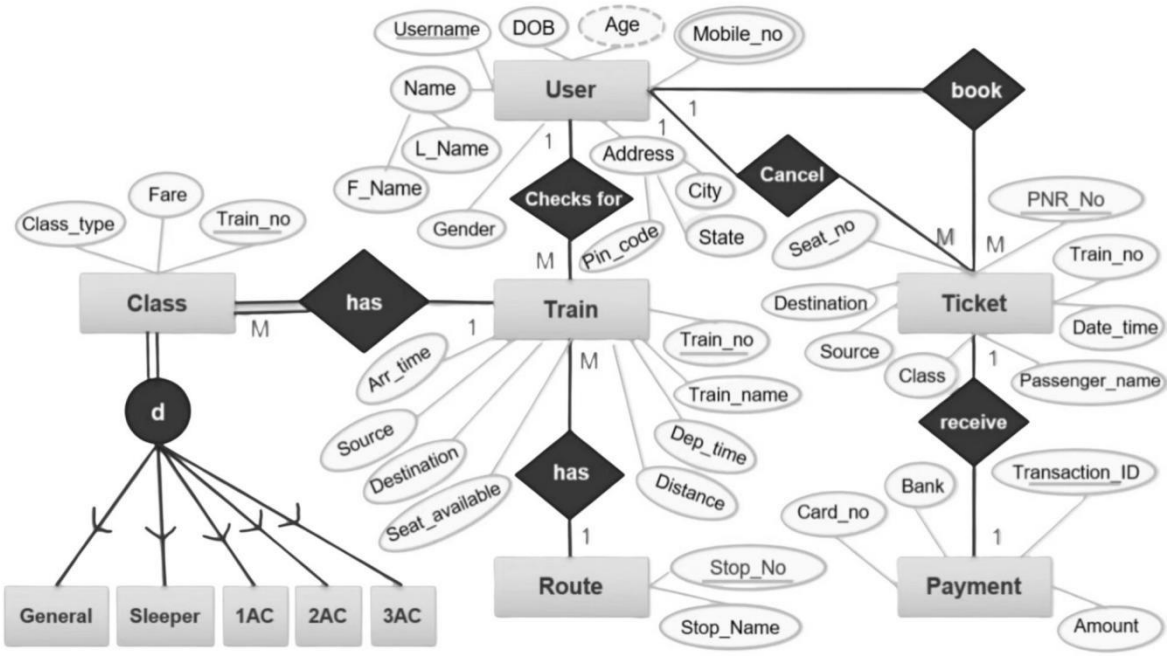

Figure 3.1 E-R Diagram

Figure 3.1 Show the Entity Relationship Diagram of Railway Reservation System

3.2 Explanation of the Components

Entities-relationships is diagrammatic representation and the backbone of the railway reservation system, allowing users to interact with train information, book tickets, and make payments seamlessly. They facilitate efficient data management and retrieval, enabling users to plan and manage their journeys effectively.

ENTITIES

User: This entity represents individuals who use the railway reservation system. It includes attributes such as Username, Date of Birth (DOB), Age, Mobile Number, First Name, Last Name, Gender, and Address (City, State, Pincode).

Train: This entity represents the trains available in the system. It includes attributes such as Train Number (Train_No), Train Name, Departure Time (Dep_Time), Distance, Arrival Time (Arr_Time), Source, Destination, and Seat Availability.

Route: This entity represents the route taken by each train. It includes attributes such as Stop Number (Stop_No) and Stop Name.

Class: This entity represents the different classes available on each train. It includes attributes such as Fare, Class Type, and Train Number.

Ticket: This entity represents the tickets booked by users. It includes attributes such as Ticket Number, Booking Date, and Status (whether booked or cancelled).

Payment: This entity represents the payment details associated with a ticket. Attributes may include Bank Name, Card Number, Transaction ID, and Amount.

RELATIONSHIP

One-to-Many Relationship between User and Train_Info:

Each user can have multiple interactions with different train information. For example, a user might check the availability of multiple trains for a particular route or date. This relationship allows users to access various train details such as schedules, availability, and routes.

One-to-Many Relationship between Train and Route:

Each train can have multiple stops along its route. For instance, a train traveling from City A to City B might make stops at intermediate cities along the way. This relationship captures the sequence of stops for each train, enabling users to view the complete route itinerary.

One-to-Many Relationship between Train and Class:

Each train can offer multiple classes of service, such as economy, business, or first class. This relationship allows for flexibility in pricing and accommodation options for passengers. Each class may have its own fare structure and amenities, providing users with choices based on their preferences and budget.

One-to-Many Relationship between User and Ticket:

Each user can book multiple tickets for different journeys or dates. This relationship reflects the booking history of users within the system. Users can manage their bookings, including viewing, modifying, or canceling tickets as needed.

One-to-One Relationship between Ticket and Payment:

Each ticket transaction is associated with a single payment. This relationship ensures that each ticket is successfully paid for, either through online transactions, credit card payments, or other payment methods. It also facilitates tracking and reconciliation of payments with booked tickets.

Overall, the ER diagram facilitates a comprehensive understanding of how users interact with trains, book tickets, and make payments within the railway reservation system. It provides a structured representation of the system's data model, enabling efficient data management and retrieval for seamless booking experiences.

CHAPTER 4

DATABASE CREATION WITH QUERIES

4.1 Queries With Explanation

Creating Database:

```
CREATE DATABASE RAILWAYRESERVATION;
```

Using the Database:

```
USE  RAILWAYRESERVATION;
```

Queries For Creating Tables:

```
CREATE TABLE Users (
        UserID INT GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
        UserName VARCHAR(15) UNIQUE NOT NULL,
        Password VARCHAR(15) NOT NULL,
        Email VARCHAR(50) UNIQUE NOT NULL,
        FirstName VARCHAR(15),
        LastName VARCHAR(15),
        DateOfBirth VARCHAR(10),
        Age INT,
        MobileNo VARCHAR(10),
       DateTimeOfBooking VARCHAR(30) NOT NULL
 );
CREATE TABLE Trains_info (
       TrainID INT PRIMARY KEY,
```

```
    TrainNumber VARCHAR(15) UNIQUE NOT NULL,
    TrainName VARCHAR(50) NOT NULL,
    DepartureTime VARCHAR(10),
    Distance DECIMAL(10, 2),
    ArrivalTime VARCHAR(10),
   Source VARCHAR(50),
   Destination VARCHAR(50),
   SeatAvailability INT,
   DateTimeOfBooking VARCHAR(30) NOT NULL
);
CREATE TABLE TrainRoutes (
    RouteID INT PRIMARY KEY,
    TrainID INT,
    StopNo INT,
    StopName VARCHAR(50),
    FOREIGN KEY (TrainID) REFERENCES Trains_info(TrainID),
    DateTimeOfBooking VARCHAR(30) NOT NULL
  );
CREATE TABLE TrainClasses (
    ClassID INT PRIMARY KEY,
    TrainID INT,
    Fare DECIMAL(10, 2) NOT NULL,
```

```
    ClassType VARCHAR(10) CHECK (ClassType IN ('General', 'Sleeper', '1AC', '2AC',
    '3AC')) NOT NULL,

    TrainNumber VARCHAR(15) NOT NULL,

    FOREIGN KEY (TrainID) REFERENCES Trains_info(TrainID),

    DateTimeOfBooking VARCHAR(30) NOT NULL

  );

CREATE TABLE Tickets (

    TicketID INT PRIMARY KEY,

    UserID INT,

    PNRNo VARCHAR(10) UNIQUE NOT NULL,

    SeatNo VARCHAR(10) NOT NULL,

   Destination VARCHAR(50) NOT NULL,

   Source VARCHAR(50) NOT NULL,

   Class VARCHAR(10) CHECK (Class IN ('General', 'Sleeper', '1AC', '2AC', '3AC'))
   NOT NULL,

   PassengerName VARCHAR(50) NOT NULL,

   DateTimeOfBooking VARCHAR(30) NOT NULL,

   TrainNumber VARCHAR(15) NOT NULL,

   Status VARCHAR(10) DEFAULT 'Booked' CHECK (Status IN ('Booked','Cancelled')),

   FOREIGN KEY (UserID) REFERENCES Users(UserID)

);

CREATE TABLE Payment (

    PaymentID INT PRIMARY KEY,
```

```
    TicketID INT UNIQUE,
    Bank VARCHAR(50),
    CardNo VARCHAR(16),
    TransactionID VARCHAR(50),
    Amount DECIMAL(10, 2),
    FOREIGN KEY (TicketID) REFERENCES Tickets(TicketID),
    DateTimeOfBooking VARCHAR(30) NOT NULL
  );
CREATE TABLE Trains_info (
  TrainID INT PRIMARY KEY,
  TrainNumber VARCHAR(15) UNIQUE NOT NULL,
  TrainName VARCHAR(50) NOT NULL,
  DepartureTime VARCHAR(10),
  Distance DECIMAL(10, 2),
  ArrivalTime VARCHAR(10);
  Source VARCHAR(15),
  Destination VARCHAR(15),
  SeatAvailability INT
);
CREATE TABLE TrainRoutes (
  RouteID INT PRIMARY KEY,
  TrainID INT,
```

```
    StopNo INT,
    StopName VARCHAR(15),
    FOREIGN KEY (TrainID) REFERENCES Trains_info(TrainID)
);
CREATE TABLE TrainClasses (
    ClassID INT PRIMARY KEY,
    TrainID INT,
    Fare DECIMAL(10, 2) NOT NULL,
    ClassType VARCHAR(10) CHECK (ClassType IN ('General', 'Sleeper', '1AC', '2AC',
      '3AC')) NOT NULL,
    TrainNumber VARCHAR(15) NOT NULL,
    FOREIGN KEY (TrainID) REFERENCES Trains_info(TrainID)
);
CREATE TABLE Tickets (
    TicketID INT PRIMARY KEY,
    UserID INT,
    PNRNo VARCHAR(10) UNIQUE NOT NULL,
    SeatNo VARCHAR(10) NOT NULL,
    Destination VARCHAR(15) NOT NULL,
    Source VARCHAR(15) NOT NULL,
    Class VARCHAR(10) CHECK (Class IN ('General', 'Sleeper', '1AC', '2AC', '3AC')) NOT
    NULL,
    PassengerName VARCHAR(15) NOT NULL,
```

```
    DateTimeVARCHAR(10) NOT NULL,
    TrainNumber VARCHAR(15) NOT NULL,
    Status VARCHAR(10) DEFAULT 'Booked' CHECK (Status IN ('Booked', 'Cancelled')),
    FOREIGN KEY (UserID) REFERENCES Users(UserID)
);
CREATE TABLE Payment (
    PaymentID INT PRIMARY KEY,
    TicketID INT UNIQUE,
    Bank VARCHAR(15),
    CardNo VARCHAR(16),
    TransactionID VARCHAR(15),
    Amount DECIMAL(10, 2),
    FOREIGN KEY (TicketID) REFERENCES Tickets(TicketID)
);
-- Join Users table with Tickets table
SELECT *
FROM Users u
JOIN Tickets t ON u.UserID = t.UserID;
-- Join Users table with Payment table
SELECT *
FROM Users u
JOIN Payment p ON u.UserID = p.UserID_Payment;
```

```
-- Join Trains table with Tickets table
SELECT *
FROM Trains tr
JOIN Tickets t ON tr.TrainNumber = t.TrainNumber;

-- Join Trains table with TrainRoutes table
SELECT *
FROM Trains tr
JOIN TrainRoutes trr ON tr.TrainID = trr.TrainID;

-- Join Trains table with TrainClasses table
SELECT *
FROM Trains tr
JOIN TrainClasses tc ON tr.TrainID = tc.TrainID;

-- Join TrainRoutes table with Trains table
SELECT *
FROM TrainRoutes trr
JOIN Trains tr ON trr.TrainID = tr.TrainID;

-- Join TrainClasses table with Trains table
SELECT *
FROM TrainClasses tc
JOIN Trains tr ON tc.TrainID = tr.TrainID;
```

```
-- Join Tickets table with Users table

SELECT *

FROM Tickets t

JOIN Users u ON t.UserID = u.UserID;

-- Join Tickets table with Trains table

SELECT *

FROM Tickets t

JOIN Trains tr ON t.TrainNumber = tr.TrainNumber;

-- Join Tickets table with Payment table

SELECT *

FROM Tickets t

JOIN Payment p ON t.TicketID = p.TicketID;

-- Join Payment table with Users table & Payment table with Tickets table

SELECT u.*, p.Bank, p.CardNo, p.TransactionID, p.Amount

FROM Users u

JOIN Tickets t ON u.UserID = t.UserID

JOIN Payment p ON t.TicketID = p.TicketID;
```

CHAPTER 5

OUTPUT SCREENSHOT

5.1 Queries Execution

```
SQL> CREATE TABLE Users (
  2         UserID INT GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
  3         UserName VARCHAR(15) UNIQUE NOT NULL,
  4          Password VARCHAR(15) NOT NULL,
  5          Email VARCHAR(15) UNIQUE NOT NULL,
  6          FirstName VARCHAR(15),
  7          LastName VARCHAR(15),
  8         DateOfBirth VARCHAR(10),
  9          Age INT,
 10         MobileNo NUMBER(10)
 11         );

Table created.
```

Figure 5.1 Creation of User's Table

Figure 5.1 shows the structure of the user's table for railway reservation system.

```
SQL> CREATE TABLE Trains_info (
  2      TrainID INT PRIMARY KEY,
  3      TrainNumber VARCHAR(15) UNIQUE NOT NULL,
  4      TrainName VARCHAR(50) NOT NULL,
  5      DepartureTime TIMESTAMP,
  6      Distance DECIMAL(10, 2),
  7      ArrivalTime TIMESTAMP,
  8      Source VARCHAR(50),
  9      Destination VARCHAR(50),
 10      SeatAvailability INT
 11  );

Table created.
```

Figure 5.2 Creation of Train's Info Table

Figure 5.2 Shows the structure of the Train's Info table for Railway Reservation System.

```
SQL> CREATE TABLE TrainRoutes (
  2      RouteID INT PRIMARY KEY,
  3      TrainID INT,
  4      StopNo INT,
  5      StopName VARCHAR(50),
  6      FOREIGN KEY (TrainID) REFERENCES Trains_info(TrainID)
  7  );

Table created.
```

Figure 5.3 Creation of Train Routes Table

Figure 5.3 Shows the structure of the Train Routes table for Railway Reservation System.

```
SQL> CREATE TABLE TrainClasses (
  2      ClassID INT PRIMARY KEY,
  3      TrainID INT,
  4      Fare DECIMAL(10, 2) NOT NULL,
  5      ClassType VARCHAR(10) CHECK (ClassType IN ('General', 'Sleeper', '1
AC', '2AC', '3AC'))
  6  NOT NULL,
  7      TrainNumber VARCHAR(15) NOT NULL,
  8      FOREIGN KEY (TrainID) REFERENCES Trains_info(TrainID)
  9  );

Table created.
```

Figure 5.4 Creation of Train's Classes Table

Figure 5.4 Shows the structure of the Train's Classes table for Railway Reservation System.

```
SQL> CREATE TABLE Tickets (
  2      TicketID INT PRIMARY KEY,
  3      UserID INT,
  4      PNRNo VARCHAR(20) UNIQUE NOT NULL,
  5      SeatNo VARCHAR(10) NOT NULL,
  6      Destination VARCHAR(50) NOT NULL,
  7      Source VARCHAR(50) NOT NULL,
  8      Class VARCHAR(10) CHECK (Class IN ('General', 'Sleeper', '1AC', '2AC', '3AC')) NOT NULL,
  9      PassengerName VARCHAR(50) NOT NULL,
 10      DateTime TIMESTAMP NOT NULL,
 11      TrainNumber VARCHAR(15) NOT NULL,
 12      Status VARCHAR(10) DEFAULT 'Booked' CHECK (Status IN ('Booked', 'Cancelled')),
 13      FOREIGN KEY (UserID) REFERENCES Users(UserID)
 14  );

Table created.
```

Figure 5.5 Creation of Train's Tickets Table

Figure 5.5 Shows the structure of the Train's Tickets table for Railway Reservation System.

```
SQL> CREATE TABLE Payment (
  2      PaymentID INT PRIMARY KEY,
  3      TicketID INT UNIQUE,
  4      Bank VARCHAR(50),
  5      CardNo VARCHAR(16),
  6      TransactionID VARCHAR(50),
  7      Amount DECIMAL(10, 2),
  8      FOREIGN KEY (TicketID) REFERENCES Tickets(TicketID)
  9  );

Table created.
```

Figure 5.6 Creation of Payment Table

Figure 5.6 Shows the structure of the Payment table for Railway Reservation System.

CHAPTER 6

CONCLUSION

The Railway Reservation System presented offers a comprehensive solution for efficient management of train bookings, user information, and payment processing. Key features include user management, ticket booking, cancellation, train information maintenance, and payment processing. Integration with existing ticketing systems and robust reporting capabilities enhance operational efficiency and decision-making. The system streamlines user experience, ensures secure transactions, and optimizes revenue generation for railway authorities.

Overall, it represents a modernized approach to railway reservation, facilitating seamless travel experiences for passengers while empowering railway operators with data-driven insights for improved service delivery.

REFERENCES

1. Abraham Silberschatz- Henry F. Korth, S. Sudharshan, “Database System Concepts”, Tata McGraw Hill, 7th Edition, 2019.
2. Ramez Elmasri, Shamkant B. Navathe, “Fundamentals of Database Systems”, Pearson Education, 7 th Edition, 2015.
3. C.J. Date, A.Kannan, S.Swamynathan, “An Introduction to Database Systems”, Pearson
4. Education, 8th Edition, 2006.
5. Raghu Ramakrishnan, “Database Management Systems”, McGraw-Hill College Publications, 4th Edition, 2015.
6. G.K.Gupta, "Database Management Systems”, Tata McGraw Hill, 1st Edition, 2018.
7. Atul Kahate, “Introduction to Database Management Systems”, Pearson Education, 1st Edition, 2004.
8. Ivan Bayross, “SQL, PL/SQL the Programming Language of Oracle”, BPB Publications, 2010.

RUBRICS

SNO	DESCRIPTION	MARKS AWARDED	MARKS OBTAINED
1	Abstract & Problem Statement	10	
2	Schema Definition & E-R Diagram	10	
3	Implementation	20	
4	Demo & Viva	10	
TOTAL		50	