Database system Lab

SRS Document

Submitted to

Prof. Awais Awan

Submitted by

M Mudassir(23-CS-90)

Uzair Khalid(23-CS-10)

Aneeqa Imran(23-CS-32)

Amna Naseer(23-CS-150)

Department Of Computer Science

University Of Engineering And Technology, Taxila

Table of Contents

1 Introduction:

1.1 Purpose

The purpose of this document is to define the software requirements for the **Train Reservation System**, which aims to provide an efficient and user-friendly platform for booking train tickets. The system will allow users to register, search for trains, book tickets, and manage reservations. Administrators will have control over train schedules, availability, and user management. The system will be developed using **C# with .NET Framework** for the application and **MS SQL Server** for database management.

1.2 Scope

The Train Reservation System will facilitate seamless ticket booking by maintaining real-time data on train schedules, available seats, and user reservations. The system will provide role-based access, allowing customers to book tickets and administrators to manage train operations. Data integrity and security measures will be implemented to ensure secure transactions and prevent unauthorized access. Future extensions may include payment gateway integration for online payments.

1.3 Intended Audience and Usage

- **End Users (Customers)**: Individuals booking train tickets online.
- **Administrators**: Responsible for managing train schedules and monitoring reservations.
- **Developers & Testers**: Software engineers maintaining and enhancing the system.
- **Project Managers**: Overseeing the project development and ensuring quality compliance.

1.4 Technologies Used

- **Frontend & Backend**: C# with .NET Framework
- **Database**: MS SQL Server
- **Development Environment**: Visual Studio
- **Security Measures**: Hashing for password storage, database constraints for integrity

2. Functional Requirements:

2.1 User Management

- Users can register and log in with unique credentials.
- User roles include **Customer** and **Admin** with different permissions.
- Users can update their profile information and change passwords securely.

2.2 Train Management (Admin Only)

- Admins can add, update, and remove train details.
- The system maintains details like **train name, source, destination, departure time, arrival time, and available seats**.
- Admins can monitor reservations and track seat availability.

2.3 Reservation Management

- Users can search for trains based on **source, destination, and date**.
- Users can book, cancel, and view their reservations.
- The system automatically updates available seats upon booking or cancellation.
- Reservation status options include **Confirmed** and **Cancelled**.

2.4 Payment Processing (Future Scope)

- Future updates will include payment gateway integration for online transactions.
- Secure payment processing will be implemented using encryption and validation techniques.

3. Database Schema:

3.1 Tables

The database consists of the following tables:

1. **Users** (Manages user information)

- **userID** (Primary Key) - Unique identifier for users
- **fullName**, **emailAddress (Unique)**, **passwordHash**, **contactNumber**, **userRole (Customer/Admin)**

2. **Trains** (Stores train details)
   - **trainID** (Primary Key) - Unique identifier for trains
   - **trainName**, **sourceStation**, **destinationStation**, **departureDateTime**, **arrivalDateTime**, **totalAvailableSeats**
3. **Reservations** (Stores reservation details)
   - **reservationID** (Primary Key) - Unique booking ID
   - **userID** (Foreign Key) → References Users(userID)
   - **trainID** (Foreign Key) → References Trains(trainID)
   - **journeyDate**, **numberOfSeatsBooked**, **reservationStatus (Confirmed/Canceled)**

3.2 Relationship

1 **Users → Reservations (One-to-Many)**

- **One User** can make **multiple Reservations**.
- **userID** is the **Primary Key** in the **Users** table and acts as a **Foreign Key** in the Reservations table.

2 **Trains → Reservations (One-to-Many)**

- **One Train** can have **multiple Reservations**.
- **trainID** is the **Primary Key** in the **Trains** table and acts as a **Foreign Key** in the Reservations table.

3.2 Normalization

The database schema is designed following the **Third Normal Form (3NF)** to ensure minimal redundancy and maximum data integrity:

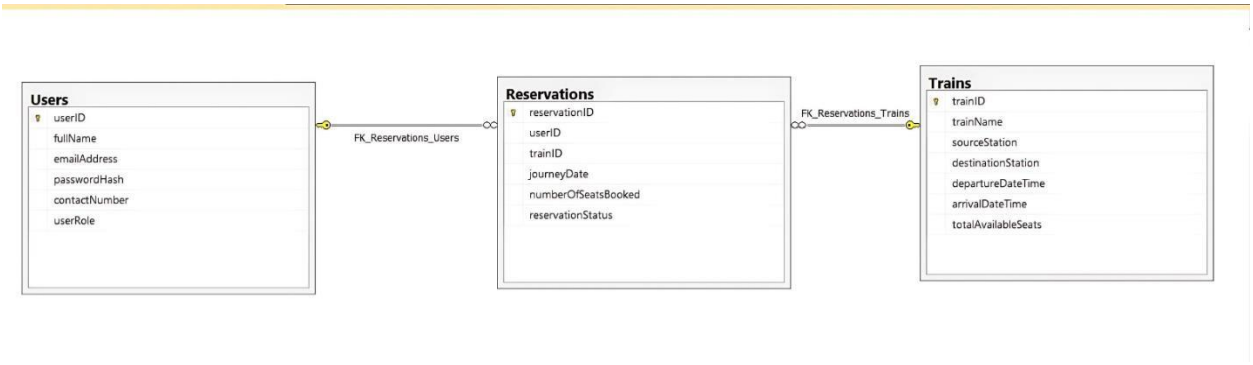
1. **1NF (First Normal Form)**: Ensures atomicity by storing only single values in each field.
2. **2NF (Second Normal Form)**: Eliminates partial dependencies by ensuring all non-key attributes fully depend on the primary key.
3. **3NF (Third Normal Form)**: Removes transitive dependencies so that non-key attributes are not dependent on other non-key attributes.

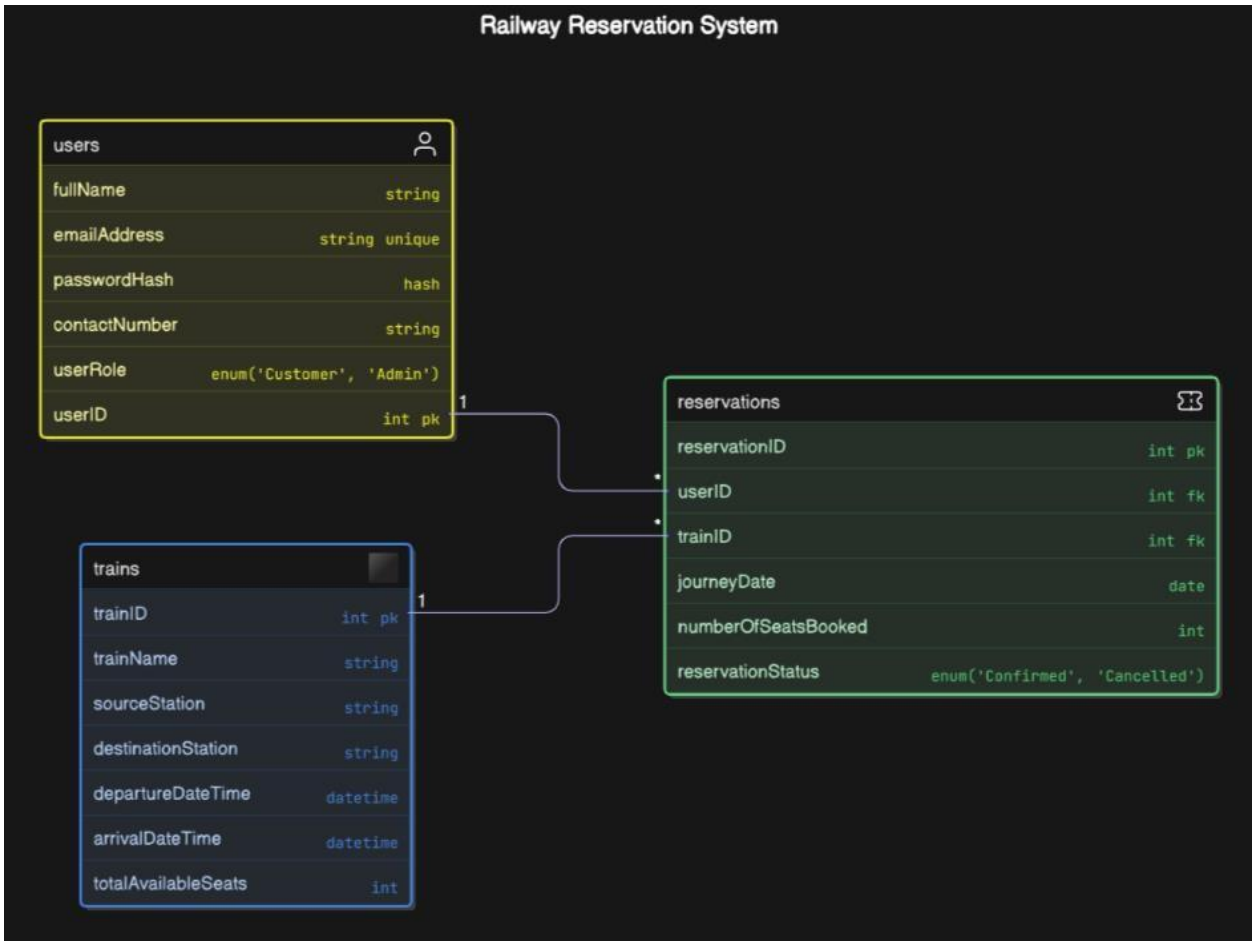
4. Data Dictionary:

Table Name	Cloumn Name	Data Type	Constrains	Description
Users	userID	INT	PRIMARY KEY	Unique identifier for users
Users	emailAddress	STRING	UNIQUE, NOT NULL	User's email (login credential)
Users	passwordHash	HASH	NOT NULL	Encrypted password for security
Trains	trainID	INT	PRIMARY KEY	Unique identifier for trains
Trains	trainName	STRING	PRIMARY KEY	Name of the train
Trains	totalAvailableSeats	INT	NOT NULL	Total number of seats available
Reservations	reservationID	INT	PRIMARY KEY	Unique booking ID
Reservations	userID	INT	FOREIGN KEY	References Users(userID)

Reservations	trainID	INT	FOREIGN KEY	References Trains(trainID)
Reservations	reservationStatus	ENUM	NOT NULL	Booking status (Confirmed/Canceled)

5. ER Diagram:





6. Non-Functional Requirements:

6.1 Scalability

- The system must support a growing number of users and train bookings without performance degradation.
- The database should be optimized for **efficient query execution**.

6.2 Security

- User passwords will be **hashed** before storage.
- The system will implement **role-based access control (RBAC)** for users and administrators.
- SQL queries will be **sanitized** to prevent SQL injection attacks.

6.3 Performance

- The database should ensure **fast query execution** for real-time seat availability.
- Indexing should be used on frequently queried fields such as userID, trainID, and reservationID.

7. Conclusion:

This document outlines the system's functional and database design specifications in detail. The **Train Reservation System** is designed to offer an efficient, secure, and user-friendly experience for both customers and administrators. The use of **C# with .NET and MS SQL Server** ensures a robust and scalable platform that adheres to best practices in database design and software development. Future enhancements, such as **payment processing**, can be integrated to provide a complete online booking solution.