**Department of Software Engineering**

**BAHRIA UNIVERSITY KARACHI CAMPUS** 

**Discovering Knowledge**

**COURSE: CSL 220**

**DATABASE MANAGEMENT SYSTEM**

**PROJECT REPORT**

**CLASS: BSE – 4C (SPRING - 2024)**

**Project Title**

**“ Transport management systems”**

**Group Members**

|  |  |
| --- | --- |
| **Student Name** | **Enrollment#** |
| Ayesha | 02-131222-090 |
| Hafsa Shahid | 02-131222-088 |
| Arifa Naseem | 02-131222-097 |

**Submitted to:**

Course Instructor: Engr. Bushra Fazal

Lab Instructor: Engr. Noor us Sabah

**Abstract**

The transport management system (TMS) offers a centralized platform for efficiently managing various aspects of transportation operations. This system addresses significant challenges such as manual scheduling, booking inefficiencies, and passenger management issues. By leveraging a relational database model using SQL Server and MySQL, the TMS enhances operational accuracy and efficiency. Key features of the system include user management, bus information storage, scheduling, ticketing, and revenue calculation. The primary objective of this project is to improve overall user satisfaction and operational performance for transportation companies, providing a scalable and adaptable solution to meet the evolving needs of the industry.

TABLE OF Contents

[1. Introduction 4](#_Toc168743085)

[2. Problem Statement 4](#_Toc168743086)

[3. Proposed Solution 4](#_Toc168743087)

[3.1. Features of the project 4](#_Toc168743088)

[3.2. Methodology 4](#_Toc168743089)

[3.3. Technologies 5](#_Toc168743090)

[3.4. Block Diagram 5](#_Toc168743091)

[3.5. Technologies to be used 5](#_Toc168743092)

[4. Software Design Description 5](#_Toc168743093)

[4.1. Design Overview 5](#_Toc168743094)

[4.2. Work Flow Diagram 6](#_Toc168743095)

[4.3. Use Case Diagram 7](#_Toc168743096)

[4.4. Sequence Diagram 8](#_Toc168743097)

[4.5. ER Diagram 9](#_Toc168743098)

[4.6. Technologies 9](#_Toc168743099)

[4.7. Block Diagram 10](#_Toc168743100)

[4.8. Technologies to be used 10](#_Toc168743101)

[4.9. Screen Images 10](#_Toc168743102)

[5. Project Scope 13](#_Toc168743103)

[6. Module Distribution 13](#_Toc168743104)

[7. Code 13](#_Toc168743105)

[8. Conclusion 17](#_Toc168743106)

[9. References 17](#_Toc168743107)

# Introduction

Transport management systems play a crucial role in efficiently managing various aspects of transportation operations. This database represents a comprehensive system designed to handle user management, administrative tasks, bus information, scheduling, ticketing, and passenger details.

# Problem Statement

The transportation sector often faces challenges in organizing schedules, managing bookings, and ensuring passenger satisfaction. Manual processes can be cumbersome and error-prone, leading to inefficiencies and customer dissatisfaction.

# Proposed Solution

The proposed solution is a robust transport management system that automates key tasks involved in managing transportation services. It streamlines operations, enhances accuracy, and improves the overall passenger experience

## Features of the project

* **User Management**:

Allows the registration and management of both administrators and non-administrative users.

* **Bus Information**:

Stores details about buses, including registration numbers, agency information, seating arrangements, and amenities.

* **Scheduling**:

Facilitates the scheduling of bus routes, driver assignments, and fare calculation.

* **Ticketing**:

Manages ticket bookings, including dates, seat reservations, and passenger information.

* **Passenger Details**:

Records passenger information for each journey, ensuring accurate tracking and reporting.

* **Revenue Calculation**:

total revenue generated by each agency based on fare collection.

## Methodology

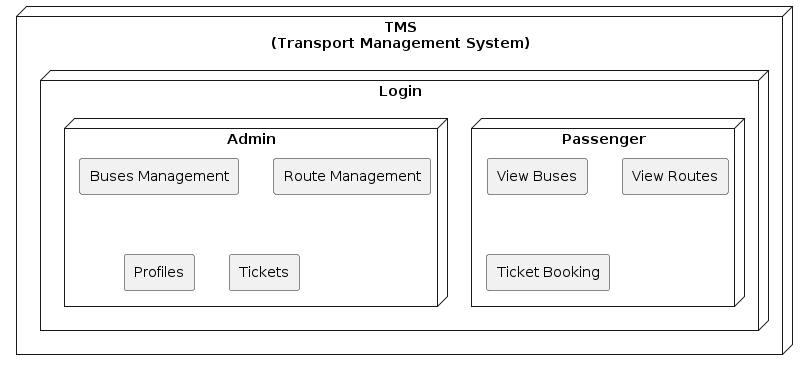
The system utilizes a relational database model to organize and manage data efficiently. Structured Query Language (SQL) is employed to perform database operations, including data insertion, retrieval, updating, and deletion. Triggers and procedures are implemented to automate certain tasks and enhance system functionality.

## Technologies

**Database Management System**: SQL Server, MySQL

**Additional Tools**: SQL Server Management Studio, Visual Studio

## Block Diagram



## Technologies to be used

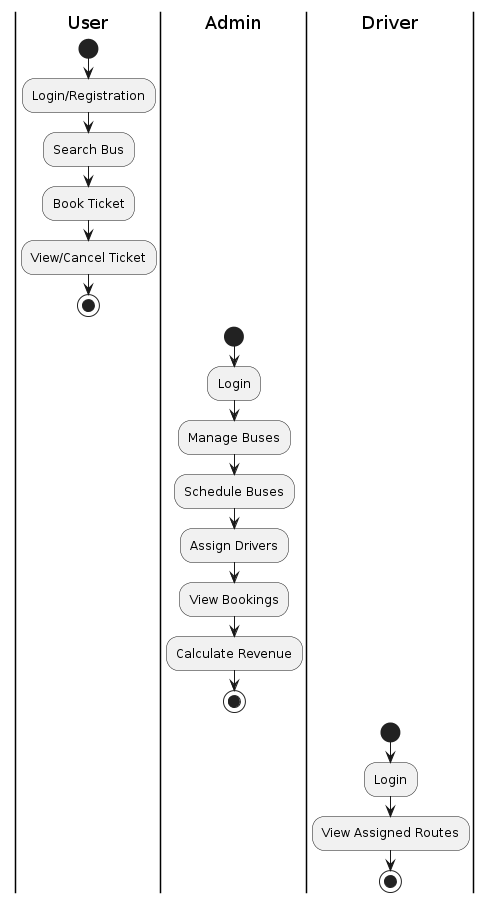
**Programming Language**: SQL, C#

# Software Design Description

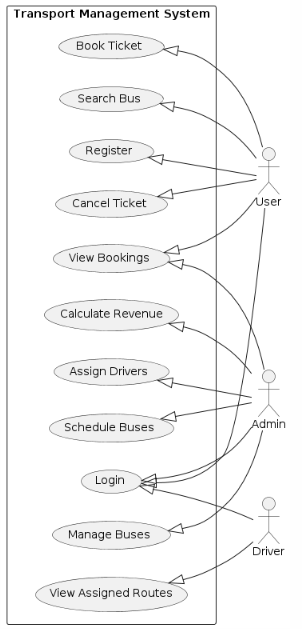
## Design Overview

The system is designed to manage transportation operations by providing a centralized platform for user management, bus information, scheduling, ticketing, and revenue calculation.

## Work Flow Diagram

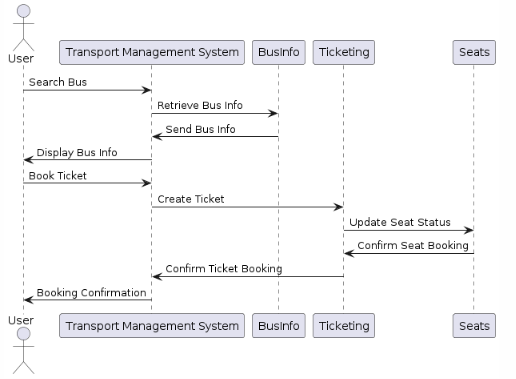


## Use Case Diagram

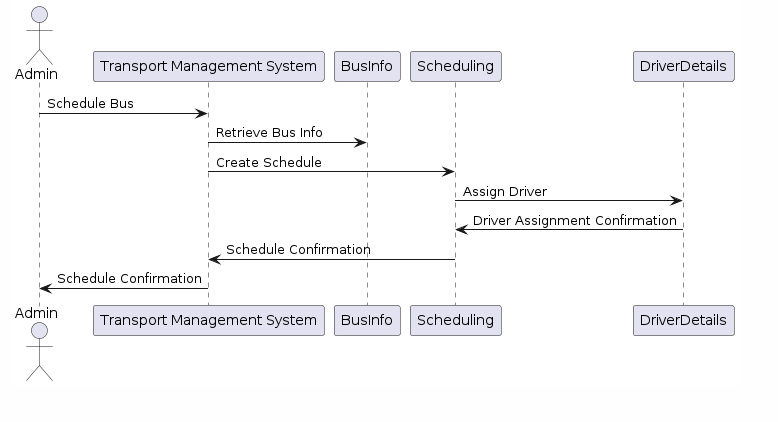


## Sequence Diagram

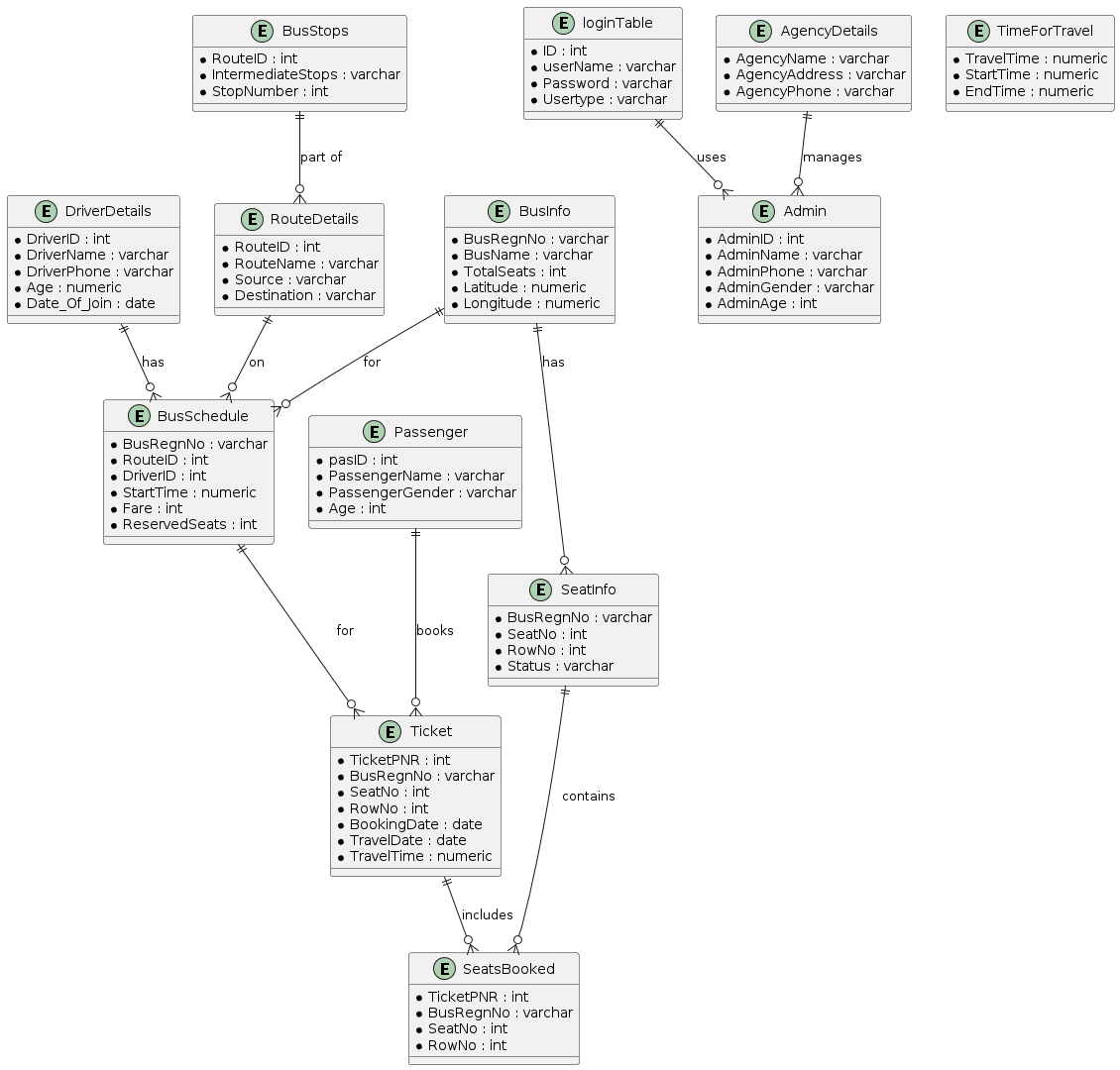
**For Booking a Ticket:**



**For Scheduling a Bus:**



## ER Diagram

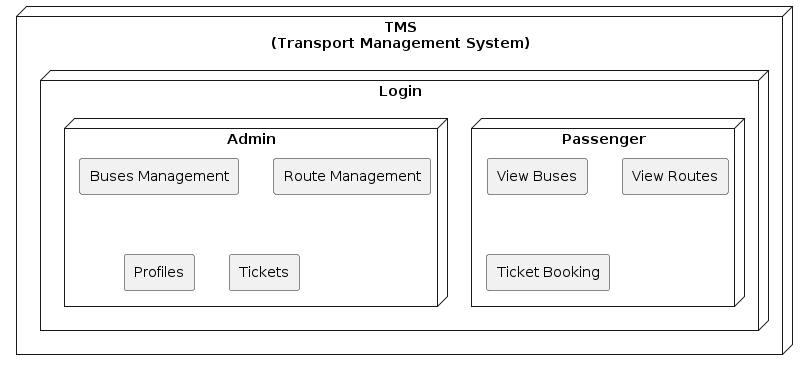


## Technologies

**Database Management System**: SQL Server, MySQL

**Additional Tools**: SQL Server Management Studio, Visual Studio

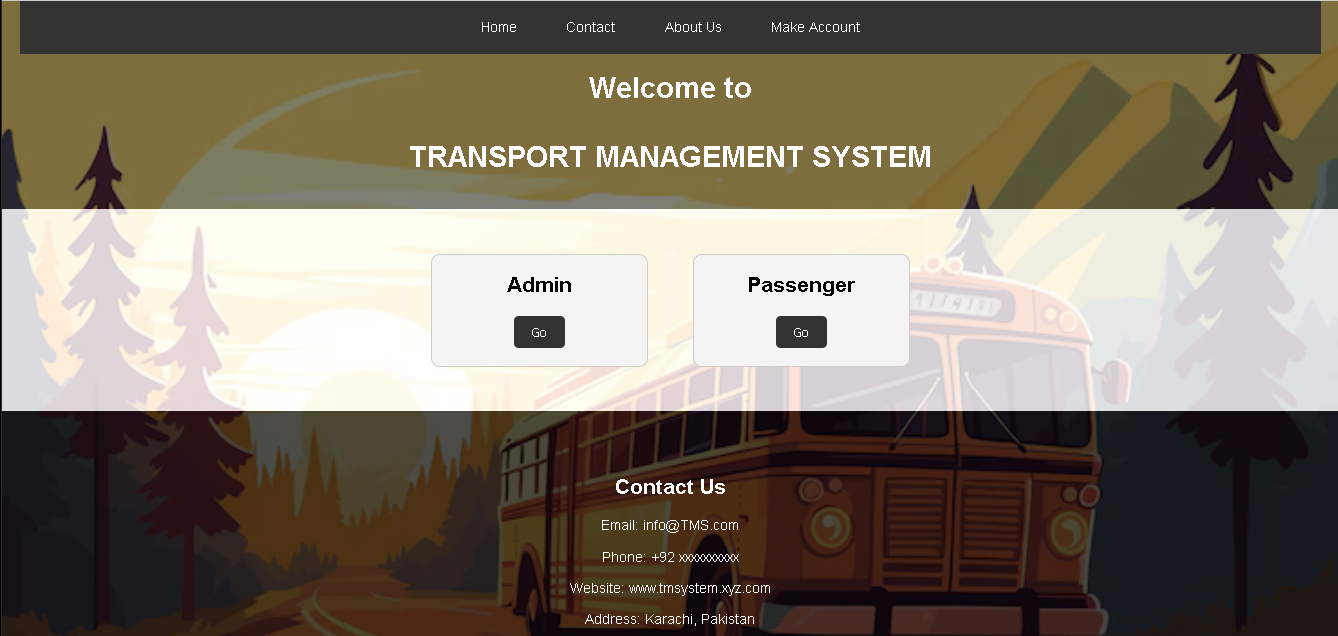
## Block Diagram

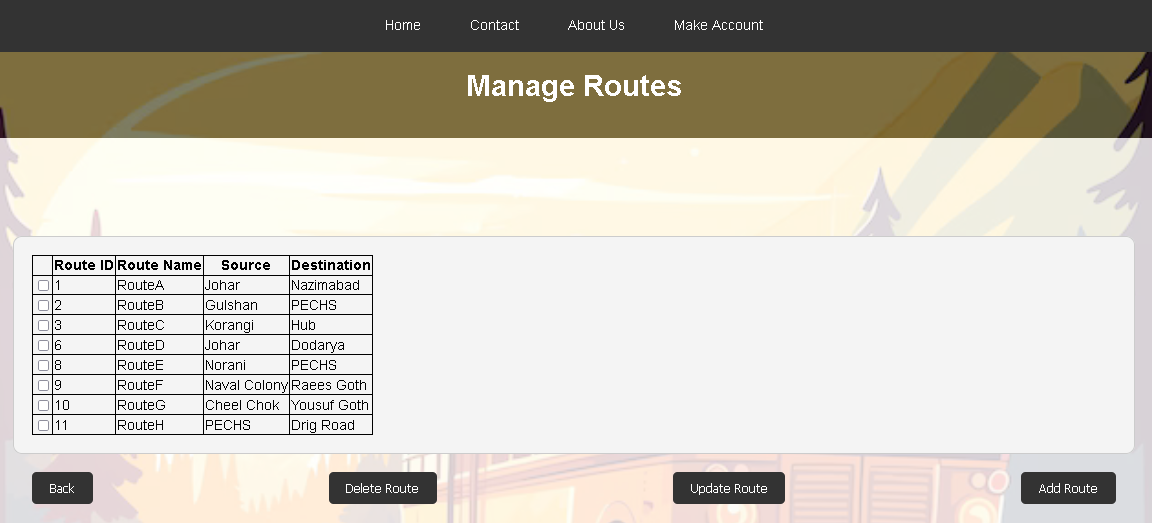


## Technologies to be used

**Programming Language**: SQL, C#

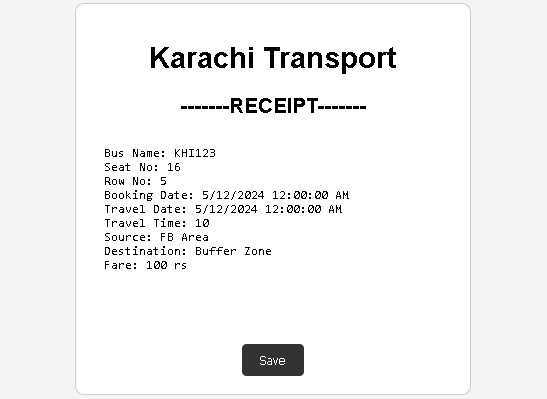
## Screen Images





|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
|  |  |



|  |  |
| --- | --- |
|  |  |
|  |

# Project Scope

The project aims to address the challenges faced in managing transportation services by providing a comprehensive and user-friendly solution. It is designed to scale and adapt to the evolving needs of transportation companies, including bus agencies, travel operators, and passengers.

# Module Distribution

**Member 1 (Arifa)**: Logical Module - Tickets bookings, buses, and routes planning.

**Member 2 (Ayesha)**: Scheduling Module - Implement features for creating and managing bus schedules; Database layer - database design and creation.

**Member 3 (Hafsa)**: View Module - User interface design, view data including user’s data, buses, and routes data.

# Code

|  |  |
| --- | --- |
| create database TMSdb;  use TMSdb;  CREATE TABLE BusSchedule (  BusRegnNo VARCHAR(15) NOT NULL,  RouteID INT,  DriverID INT,  StartTime NUMERIC(4,2),  Fare INTEGER CONSTRAINT fare\_chk CHECK (Fare > 20),  ReservedSeats INTEGER DEFAULT 0,  PRIMARY KEY (RouteID, DriverID, StartTime),  FOREIGN KEY (BusRegnNo) REFERENCES BusInfo(BusRegnNo),  FOREIGN KEY (RouteID) REFERENCES RouteDetails(RouteID),  FOREIGN KEY (DriverID) REFERENCES DriverDetails(DriverID),  CONSTRAINT Start\_chk CHECK (StartTime >= 6 AND StartTime <= 12)  );  CREATE TABLE Ticket (  TicketPNR INT IDENTITY(1,1) PRIMARY KEY NOT NULL,  BusRegnNo VARCHAR(15) FOREIGN KEY REFERENCES BusInfo(BusRegnNo),  SeatNo INT,  RowNo INT,  BookingDate DATE,  TravelDate DATE,  TravelTime NUMERIC(10) CONSTRAINT TravelTime\_con CHECK (TravelTime >= 6 AND TravelTime <= 12),  FOREIGN KEY (BusRegnNo, SeatNo, RowNo) REFERENCES SeatInfo(BusRegnNo, SeatNo, RowNo)  );  CREATE TABLE SeatsBooked (  TicketPNR INT,  BusRegnNo VARCHAR(15),  SeatNo INT,  RowNo INT,  FOREIGN KEY (TicketPNR) REFERENCES Ticket(TicketPNR),  FOREIGN KEY (BusRegnNo, SeatNo, RowNo) REFERENCES SeatInfo(BusRegnNo, SeatNo, RowNo)  );  ----------------------------------------------------CRUD QUERIES FOR TABLES--------------------------------------------------  BEGIN TRANSACTION;  DECLARE @BusRegnNo VARCHAR(15), @RouteID INT, @DriverID INT, @StartTime NUMERIC(4,2), @Fare INT, @ReservedSeats INT;  SET @BusRegnNo = 'KHI123';  SET @RouteID = 1;  SET @DriverID = 1;  SET @StartTime = 8.00;  SET @Fare = 500;  SET @ReservedSeats = 0;  INSERT INTO BusSchedule (BusRegnNo, RouteID, DriverID, StartTime, Fare, ReservedSeats)  VALUES (@BusRegnNo, @RouteID, @DriverID, @StartTime, @Fare, @ReservedSeats);  COMMIT TRANSACTION;  ---------------------------------------------------  DECLARE @RouteID INT, @DriverID INT, @StartTime NUMERIC(4,2);  SET @RouteID = 1;  SET @DriverID = 1;  SET @StartTime = 8.00;  SELECT \* FROM BusSchedule  WHERE RouteID = @RouteID AND DriverID = @DriverID AND StartTime = @StartTime;  -----------------------------------------------  BEGIN TRANSACTION;  DECLARE @BusRegnNo VARCHAR(15), @RouteID INT, @DriverID INT, @StartTime NUMERIC(4,2), @Fare INT, @ReservedSeats INT;  SET @BusRegnNo = 'KHI123';  SET @RouteID = 1;  SET @DriverID = 1;  SET @StartTime = 8.00;  SET @Fare = 600;  SET @ReservedSeats = 5;  UPDATE BusSchedule  SET Fare = @Fare, ReservedSeats = @ReservedSeats  WHERE RouteID = @RouteID AND DriverID = @DriverID AND StartTime = @StartTime;  COMMIT TRANSACTION;  -----------------------------------------------  BEGIN TRANSACTION;  DECLARE @RouteID INT, @DriverID INT, @StartTime NUMERIC(4,2);  SET @RouteID = 1;  SET @DriverID = 1;  SET @StartTime = 8.00;  DELETE FROM BusSchedule  WHERE RouteID = @RouteID AND DriverID = @DriverID AND StartTime = @StartTime;  COMMIT TRANSACTION;  BEGIN TRANSACTION;  DECLARE @TicketPNR INT, @BusRegnNo VARCHAR(15), @SeatNo INT, @RowNo INT;  SET @TicketPNR = 4;  SET @BusRegnNo = 'KHI123';  SET @SeatNo = 4;  SET @RowNo = 1;  INSERT INTO SeatsBooked (TicketPNR, BusRegnNo, SeatNo, RowNo)  VALUES (@TicketPNR, @BusRegnNo, @SeatNo, @RowNo);  COMMIT TRANSACTION;  ----------------------------------  DECLARE @TicketPNR INT;  SET @TicketPNR = 1;  SELECT \* FROM SeatsBooked  WHERE TicketPNR = @TicketPNR;  ----------------------------------  BEGIN TRANSACTION;  DECLARE @TicketPNR INT, @BusRegnNo VARCHAR(15), @SeatNo INT, @RowNo INT;  SET @TicketPNR = 1;  SET @BusRegnNo = 'KHI123';  SET @SeatNo = 5;  SET @RowNo = 1;  UPDATE SeatsBooked  SET BusRegnNo = @BusRegnNo, SeatNo = @SeatNo, RowNo = @RowNo  WHERE TicketPNR = @TicketPNR;  COMMIT TRANSACTION;  -----------------------------------  BEGIN TRANSACTION;  DECLARE @TicketPNR INT;  SET @TicketPNR = 1;  DELETE FROM SeatsBooked  WHERE TicketPNR = @TicketPNR;  COMMIT TRANSACTION;  ------------------------------------------------------------------TRIGGERS------------------------------------------------------------  ------Trigger 1: Enforce Referential Integrity for BusSchedule-------  CREATE TRIGGER trg\_BusSchedule\_Insert  ON BusSchedule  AFTER INSERT  AS  BEGIN  -- Check if the inserted RouteID exists in RouteDetails  IF NOT EXISTS (SELECT 1 FROM RouteDetails WHERE RouteID = (SELECT RouteID FROM inserted))  BEGIN  RAISERROR ('RouteID does not exist in RouteDetails.', 16, 1);  ROLLBACK TRANSACTION;  RETURN;  END  -- Check if the inserted DriverID exists in DriverDetails  IF NOT EXISTS (SELECT 1 FROM DriverDetails WHERE DriverID = (SELECT DriverID FROM inserted))  BEGIN  RAISERROR ('DriverID does not exist in DriverDetails.', 16, 1);  ROLLBACK TRANSACTION;  RETURN;  END  END; | ------------------Trigger 2: Log Changes in BusInfo------------------  -- Create a log table  CREATE TABLE BusInfoLog (  LogID INT IDENTITY(1,1) PRIMARY KEY,  BusRegnNo VARCHAR(15),  ChangeType VARCHAR(10),  ChangeTime DATETIME DEFAULT GETDATE(),  OldBusName VARCHAR(20),  NewBusName VARCHAR(20),  OldLatitude NUMERIC(17,10),  NewLatitude NUMERIC(17,10),  OldLongitude NUMERIC(17,10),  NewLongitude NUMERIC(17,10));  -- Create the trigger  CREATE TRIGGER trg\_BusInfo\_Update  ON BusInfo  AFTER UPDATE  AS  BEGIN  INSERT INTO BusInfoLog (BusRegnNo, ChangeType, OldBusName, NewBusName, OldLatitude, NewLatitude, OldLongitude, NewLongitude)  SELECT  i.BusRegnNo,  'UPDATE',  d.BusName, i.BusName,  d.Latitude, i.Latitude,    d.Longitude, i.Longitude  FROM  inserted i  JOIN  deleted d ON i.BusRegnNo = d.BusRegnNo;  END;  ------Trigger 3: Automatically Update Seat Status in SeatInfo--------  CREATE TRIGGER trg\_SeatsBooked\_Insert  ON SeatsBooked  AFTER INSERT  AS  BEGIN  UPDATE SeatInfo  SET Status = 'booked'  WHERE EXISTS (  SELECT 1  FROM inserted i  WHERE SeatInfo.BusRegnNo = i.BusRegnNo  AND SeatInfo.SeatNo = i.SeatNo  AND SeatInfo.RowNo = i.RowNo  );  END;  ----------Trigger 4: Ensure Age Constraint in Passenger--------------  CREATE TRIGGER trg\_Passenger\_Insert\_Update  ON Passenger  AFTER INSERT, UPDATE  AS  BEGIN  IF EXISTS (SELECT 1 FROM inserted WHERE Age < 6)  BEGIN  RAISERROR ('Passenger age must be greater than 5.', 16, 1);  ROLLBACK TRANSACTION;  RETURN;  END  END;  ----------------------------------CREATING VIEWS -------------------------  ----------View 1: Passenger Details with Their Booked Tickets-----------  CREATE VIEW PassengerTicketDetails AS  SELECT  p.pasID,  p.PassengerName,  p.PassengerGender,  p.Age,  t.TicketPNR,  t.BusRegnNo,  t.SeatNo,  t.RowNo,  t.BookingDate,  t.TravelDate,  t.TravelTime  FROM  Passenger p  JOIN  Ticket t ON p.pasID = t.TicketPNR;  ----------View 2: Bus Schedule with Route and Driver Details-----------  CREATE VIEW BusScheduleDetails AS  SELECT  bs.BusRegnNo,  bs.RouteID,  r.RouteName,  r.Source,  r.Destination,  bs.DriverID,  d.DriverName,  bs.StartTime,  bs.Fare,  bs.ReservedSeats  FROM  BusSchedule bs  JOIN  RouteDetails r ON bs.RouteID = r.RouteID  JOIN  DriverDetails d ON bs.DriverID = d.DriverID;  ----------View 3: Active Buses with Their Locations-----------  CREATE VIEW ActiveBuses AS  SELECT  b.BusRegnNo,  b.BusName,  b.Latitude,  b.Longitude,  b.AgencyName  FROM  BusInfo b;  ----------View 4: Agency Details with Their Buses-----------  CREATE VIEW AgencyBuses AS  SELECT  a.AgencyName,  a.AgencyAddress,  a.AgencyPhone,  b.BusRegnNo,  b.BusName  FROM  AgencyDetails a  JOIN  BusInfo b ON a.AgencyName = b.AgencyName;  ----------View 5: Seat Availability for a Specific Bus-----------  CREATE VIEW SeatAvailability AS  SELECT  s.BusRegnNo,  s.SeatNo,  s.RowNo,  s.Status  FROM  SeatInfo s;  **PRESENTATION LAYER CODE:**  using System.Web.UI;  using System.Web.Services;  namespace TMSystem1  { public partial class receipt : System.Web.UI.Page  {protected void Page\_Load(object sender, EventArgs e)  {if (!IsPostBack)  {if (Session["Receipt"] != null)  {string receiptDetails = Session["Receipt"].ToString();  if (Session["Source"] != null)  {receiptDetails += "\nSource: " + Session["Source"].ToString(); }  if (Session["Destination"] != null)  {receiptDetails += "\nDestination: " + Session["Destination"].ToString();}  receiptDetails += "\nFare: 100 rs";  ReceiptDetailsTextBox.Text = receiptDetails; }}}  [WebMethod]  public static void SaveImage(string imgData)  { try  { string base64 = imgData.Replace("data:image/png;base64,", "");  byte[] bytes = Convert.FromBase64String(base64);  string filePath = @"C:\Users\Administrator\Downloads\captured\_screen.png";  using (MemoryStream ms = new MemoryStream(bytes))  { using (Bitmap bitmap = new Bitmap(ms))  {bitmap.Save(filePath, ImageFormat.Png);}} }  catch (Exception ex)  { Console.WriteLine(ex.Message); throw new Exception("An error occurred while saving the receipt. Please try again.", ex);}}}} |

# Conclusion

The Transport Management System (TMS) developed in this project effectively addresses the challenges of manual transportation management by automating critical tasks such as scheduling, ticketing, and passenger management. Leveraging SQL Server and MySQL, the system enhances operational efficiency, accuracy, and user satisfaction. Key features like user management, bus information storage, and revenue calculation are seamlessly integrated, providing a scalable and adaptable solution for transportation companies. This project not only meets its initial objectives but also offers a robust foundation for future enhancements, significantly improving the overall management and user experience in the transportation sector.

# References

<https://ijrpr.com/uploads/V3ISSUE5/IJRPR4070.pdf>

<https://acropolium.com/blog/transportation-management-software-development-a-detailed-overview/>

**Drive Link:**

[**https://drive.google.com/drive/folders/1jYZQdTz1FF\_KcG9UIXG51k\_GG-RGyjIN?usp=sharing**](https://drive.google.com/drive/folders/1jYZQdTz1FF_KcG9UIXG51k_GG-RGyjIN?usp=sharing)