

CS530

**DATABASE
MANAGEMENT
SYSTEM**

Semester Project

**BUS TICKET RESERVATION
SYSTEM**

**Shashank Krishna
Naik**

USN: 01JST18CS127

Sparsh Venkappa Rao

USN: 01JST18CS139

Sanjay HB

USN: 01JST18CS120

Anant

Rayabhattachanavar

USN: 01JST18CS012

Instructor:

Dr. Manimala S

Professor

JSS Science and
Technology University
Mysore

Professor

JSS Science and
Technology University
Mysore



December 26, 2020

CONTENTS:

- 1. Abstract**
- 2. Introduction**
- 3. ER Diagram**
- 4. Schema**
- 5. Set representation**
- 6. DDL and DML Statements**
- 7. Interface View**
- 8. Conclusion**
- 9. Reference**

1 Abstract:

In this project we have ~~can~~ created a bus ticket reservation system using MySQL for storing and managing data using HTML, JavaScript, CSS, Php, Ajax for the website design. Our aim of creating this project was to get in-depth knowledge of how the real-time websites are designed and are integrated with the databases for better user experience.

2 Introduction:

With the effective and efficient mode of transportation, one could travel thousands of miles with hours and days and communicate across the globe within split of seconds. So, we've created a website for reservation of bus ticket for convenience of the customer.

Activities:

1. The passenger can register/login.
2. Can reserve bus tickets.
3. Can view ticket details.
4. Can cancel the reserved ticket.

Processes for the Passenger:

1. Register on the website:

- a. On the register page the passenger enters his personal details which is stored in the passenger table in the database.
- b. Then the passenger can login using these credentials.

2. Reserve bus ticket:

- a. The passenger enters the travel details and gets bus schedules.
- b. He selects the convenient bus schedule and receives the seat availability for the same.
- c. Available seat is selected and reserved by the passenger.
- d. The ticket details are shown upon reservation.

3. Ticket details:

- a. The passenger can get the reserved ticket details using his/her ticket id.

4. Cancel ticket:

- a. The passenger can cancel his/her reserved ticket using id before the departure date.

Back-end processes:

1. When the passenger enters the registration details it is validated in the website and transferred to the backend where a connection to the Database is made and the details are inserted in the passenger table.
2. When the passenger logs in using the credentials, it is validated with the data present in the passenger table and then redirected to the reservation page.
3. The travel details are validated and based on the particular route, it is checked in the bus schedule table and the total fare is calculated by aggregating the point-to-point fares in the path present in route table, which is displayed to the passenger.
4. Bases on the schedule selected by the passenger the seat availability is

calculated by checking the tickets reserved in the ticket table for the particular bus schedule and displayed to the user.

5. When the user selects a ticket and reserves it then the details are inserted in the ticket table and the ticket id is generated, and along with the details displayed to the user.
6. If the user cancels a ticket then the ticket id is validated and the details removed from the ticket table.

Candidate Keys

The choice of candidate keys was made by finding attribute closure of all subsets of the attributes of the relation followed by identification of Super Keys.(Set of attributes whose attribute close contains all attributes of the relation). Next we select the super keys which do not have a proper subset which is a super key in itself.

These keys qualify as our candidate keys. Following are our candidate keys :

1. Passenger
 - Email
 - Phone no
 - { Name,Phone no }
 2. Ticket
 - Tid
 - { Email,Bsid }
 3. Bus Schedule
 - Bsid
 4. Conductor
 - Cid
 5. Bus has route
 - {Rid,Bsid}
 6. Route
 - Rid
 7. Bus
 - regNo
 8. Cancel ticket Details
 - Ctid
 - {Email,Bsid,Seat No}
- Primary Keys have been underlined

Data Types

- INTEGER -age,tid,seatno,bsid,cid,phoneno
- TIME – Departure time,Arrival time,cancelled time
- The rest of the attributes are of type varchar

Functional Dependencies

1. PASSENGER

- Category->age
- Email-> password,Name,Phone no,Gender,age,category

2. Ticket

- {Origin, Destination} -> Fare
- Tid -> Bsid, Email
- Bsid -> Seat no, Origin, Destination, Departure, Arrival time, Date

3. Bus Schedule

- Bsid -> Path, Cid, Regno

4. Conductor

- Cid -> Phoneno, name

5. Bus has route

- {Rid, Bsid} -> Departure, Arrivaltime

6. Route

- Rid -> Origin, Destination
- {Origin, Destination} -> Fare

7. Bus

- regNo -> Maxseats, Model

8. Cancel ticket Details

- Ctid -> Email, Cancelledtime, Bsid
- {Bsid, Email} -> Origin, Destination, SeatNo, Reservation Date

Triggers

❖ Category Triggers

- This trigger checks if the input age range and based on that range specifies the category of the passenger.
- If the age > 60 then it allocates the category as 'senior citizen'
- If it is between 18 and 60 then it allocates the category as major
- And if it is below 18 then minor is allotted.

❖ Cancellation Trigger

- This trigger is useful when a passenger cancels a ticket. It stores the details about the cancellation in a separate table for further analysis.

Normalization

1st Normal Form

It was evident that in the bus table the conductor details was a composite attributes and a separate table has been devised which contains the conductors details that too the values are not composite.

TEST FOR LOSSLESS JOIN DECOMPOSITION

Since CONDUCTOR and BUS SCHEDULE have one attribute in common (Cid) which is also the primary key of CONDUCTOR table and the union of the

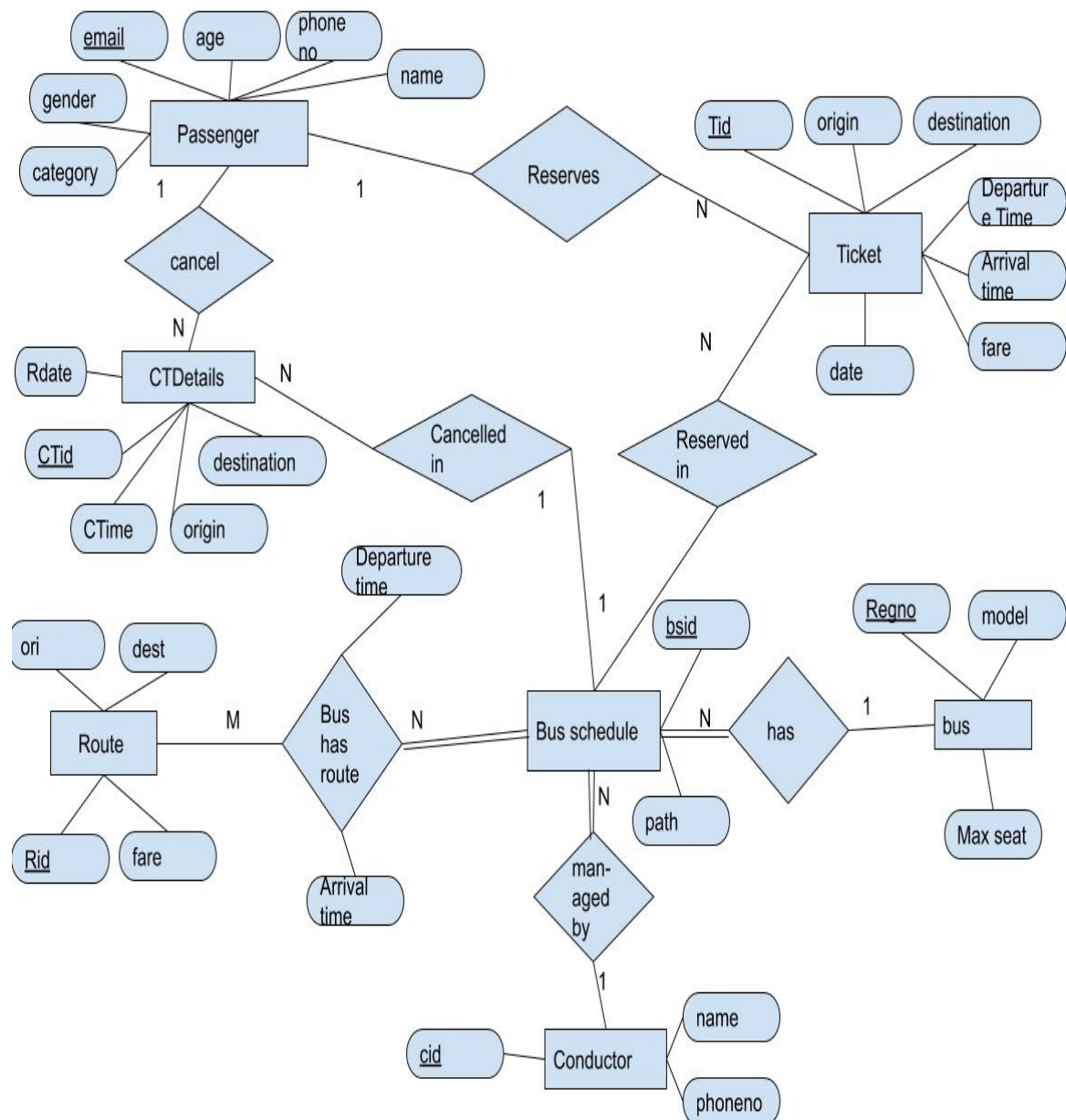
2nd Normal Form

By looking at the functional dependencies in this data model and the choice of candidate keys, it can be concluded that none of the non-prime attributes have partial dependency on any candidate key. All the tables in this data model are in 2nd Normal Form.

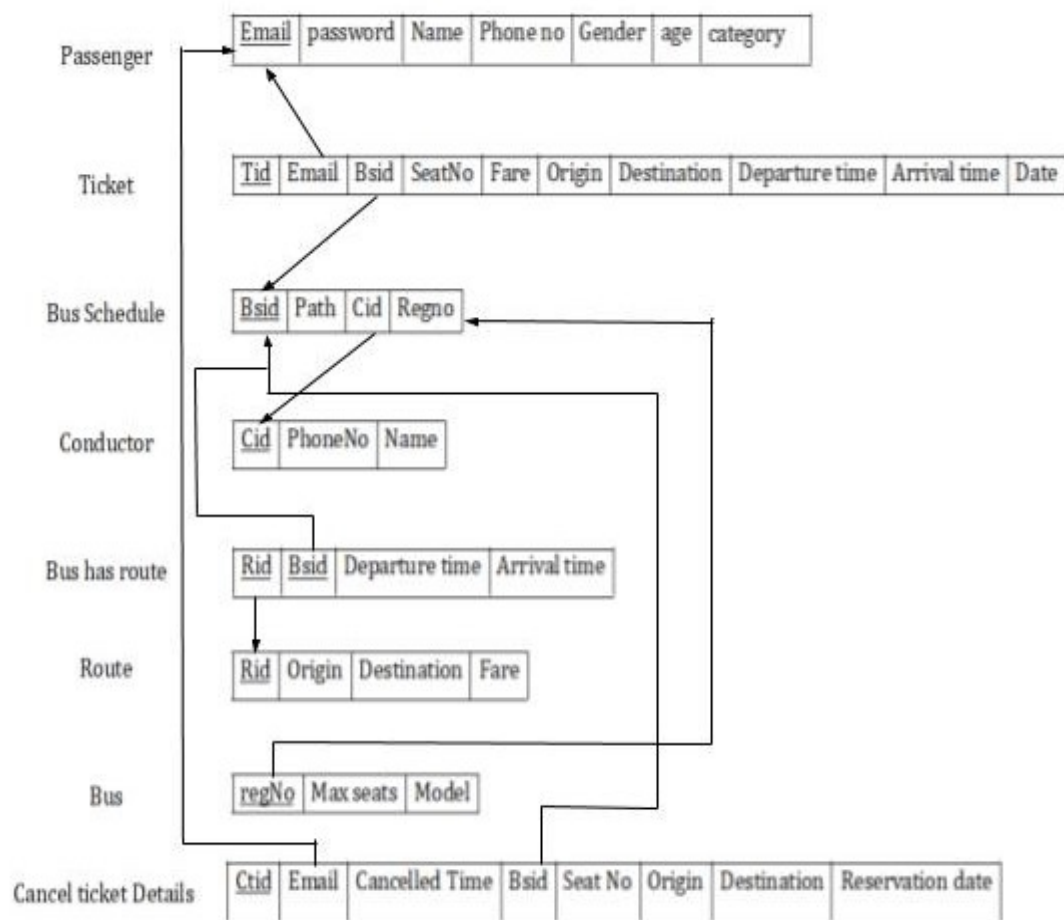
3rd Normal Form

The functional dependencies and the candidate keys clearly indicate that there is no functional dependency between two non-prime attributes in any of the tables. In other words, there is no transitive dependency in any of the tables, Hence all the tables are in 3rd Normal Form.

3 ER Diagram:

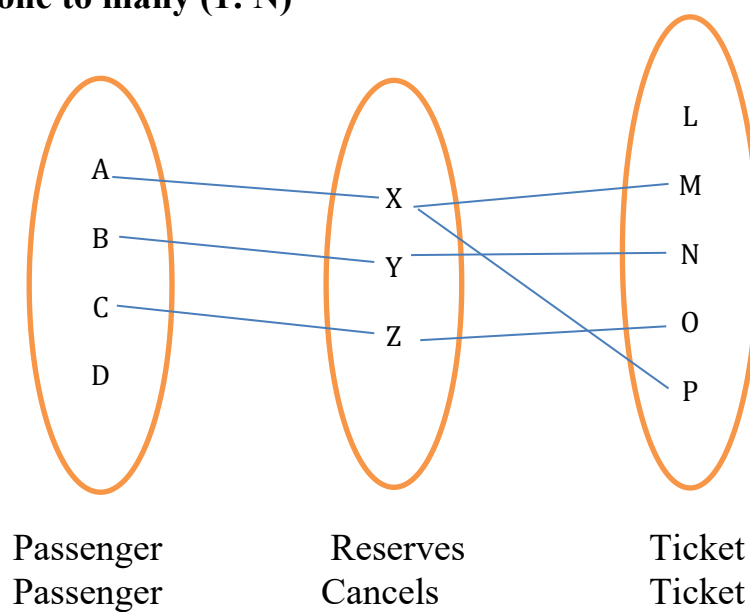


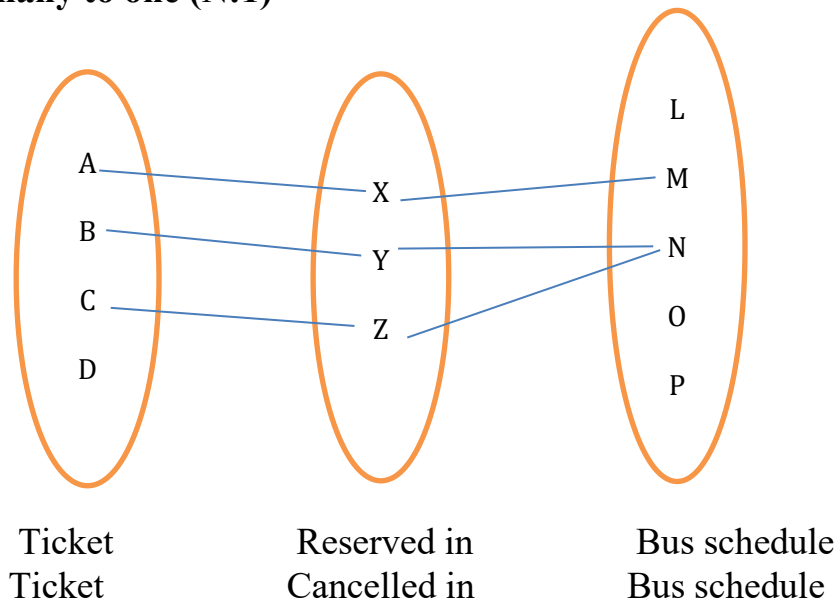
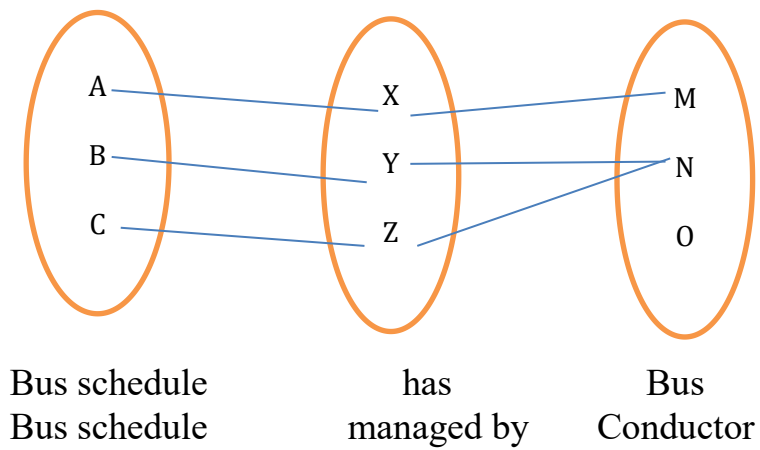
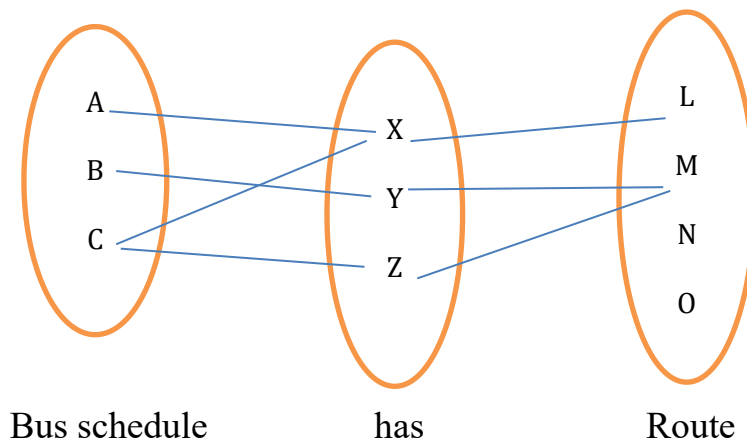
4 Schema:



5 Set representation

a. one to many (1: N)



b. many to one (N:1)**c. many to one (N:1) with total participation****d. many to many (N:M) with total participation**

6 DDL and DML Statements:

```
--
-- Database: `busrs`
--

CREATE DATABASE busrs;

USE busrs;

--
-- Table structure for table `passenger`
--

CREATE TABLE `passenger` (
  `Email` varchar(30) PRIMARY KEY NOT NULL,
  `password` varchar(36) NOT NULL,
  `Name` varchar(30) DEFAULT NULL,
  `Phone_No` bigint(10) UNIQUE NOT NULL,
  `Gender` varchar(10) DEFAULT NULL,
  `Age` int(11) NOT NULL,
  `category` varchar(20) DEFAULT NULL
);

--
-- Inserting data for table `passenger`
--
INSERT INTO `passenger` (`Email`, `password`, `Name`, `Phone_No`, `Gender`, `Age`,
`category`) VALUES
('sha@', 'dab5b23703d114558022b4b4c995379b', 'Krishna M Naik', 9538321578, 'male',
2, 'mnor'),
('sha@gmail.com', 'eb9279982226a42afdf2860dbdc29b45', 'Krishna M Naik',
9538321570, 'male', 60, 'senior citizen'),
('sha@gml.com', '86c22dd86a547dc8c4c46d5470ac6cda', 'Krishna M Naik', 9538321569,
'male', 4, 'minor'),
('shashankkrishnanaik@gmail.com', '11eba10d3544ac6d881143c0ecb59852', 'Krishna M
Naik', 9538321572, 'male', 19, 'major');

-----

CREATE TABLE `conductor` (
  `cid` int(11) PRIMARY KEY NOT NULL,
  `name` varchar(30) NOT NULL,
  `phoneNo` bigint(10) UNIQUE NOT NULL) ;

--
-- Inserting data for table `conductor`
--

INSERT INTO `conductor` (`cid`, `name`, `phoneNo`) VALUES
(5610, 'Johnny', 6545734567),
```

```
(5611, 'Robert', 8769845634),
(5612, 'Kevin', 9867954634),
(5613, 'Denzel', 9967843567),
(5614, 'Russell', 6574563458),
(5615, 'Angelina', 8746376983),
(5616, 'Amitabh', 8799543667),
(5617, 'Aamir', 6875467646),
(5618, 'Shah Rukh', 7665576566),
(5619, 'Akshay Kumar', 8887656656),
(5620, 'Hrithik', 7733872648),
(5621, 'Kumar', 8487266499),
(5622, 'Salman', 7498993766),
(5623, 'Ranveer', 7756352478);
```

```
-- -----
```

```
--
```

```
-- Table structure for table `bus`
```

```
--
```

```
CREATE TABLE `bus` (
  `regNo` varchar(20) PRIMARY KEY NOT NULL,
  `maxSeats` int(11) NOT NULL,
  `model` varchar(20) NOT NULL);
```

```
--
```

```
-- Inserting data for table `bus`
```

```
--
```

```
INSERT INTO `bus` (`regNo`, `maxSeats`, `model`) VALUES
('KA-O2 A-0985', 54, 'Tata Lpo 1623'),
('KA-O3 A-2379', 54, 'Tata Lp 909'),
('KA-O4 A-7827', 54, 'Tata Lpo 1623'),
('KA-O4 F-6536', 54, 'Tata Lp 909'),
('KA-O4 H-5623', 54, 'Tata Lpo 1623'),
('KA-O4 S-8332', 56, 'Volvo B7R'),
('KA-O5 F-9855', 56, 'Volvo B7R'),
('KA-O5 S-9429', 56, 'Volvo 9400XL'),
('KA-O7 H-0986', 56, 'Volvo B7R'),
('KA-O7 H-9886', 56, 'Volvo 9400XL'),
('KA-O7 S-2483', 54, 'Tata Lpo 1613');
```

```
--
```

```
-- Table structure for table `route`
```

```
--
```

```
CREATE TABLE `route` (
  `rid` int(11) PRIMARY KEY NOT NULL,
  `origin` varchar(20) NOT NULL,
  `destination` varchar(20) NOT NULL,
  `fare` int(11) NOT NULL
) ;
```

--

-- Inserting data for table `route`

--

INSERT INTO `route` (`rid`, `origin`, `destination`, `fare`) VALUES

(1, 'Karwar', 'Gokarna', 65),
 (2, 'Gokarna', 'Bhatkal', 88),
 (3, 'Bhatkal', 'Udupi', 89),
 (4, 'Udupi', 'Mangalore', 55),
 (5, 'Bhatkal', 'Sirsi', 149),
 (6, 'Sagar', 'Sirsi', 73),
 (7, 'Bhatkal', 'Sagar', 108),
 (8, 'Sagar', 'Shivamogga', 79),
 (9, 'Udupi', 'Shivamogga', 147),
 (10, 'Shivamogga', 'Mangalore', 196),
 (11, 'Mangalore', 'Shivamogga', 200),
 (12, 'Shivamogga', 'Udupi', 149),
 (13, 'Shivamogga', 'Sagar', 76),
 (14, 'Sagar', 'Bhatkal', 105),
 (15, 'Sirsi', 'Sagar', 70),
 (16, 'Sirsi', 'Bhatkal', 142),
 (17, 'Mangalore', 'Udupi', 55),
 (18, 'Udupi', 'Bhatkal', 90),
 (19, 'Bhatkal', 'Gokarna', 84),
 (20, 'Gokarna', 'Karwar', 64);

--Table structure for table `busschedule`

CREATE TABLE `busschedule` (

 `bsid` int(11) PRIMARY KEY NOT NULL,

 `path` varchar(40) NOT NULL,

 `cid` int(11) NOT NULL,

 `regNo` varchar(20) NOT NULL,

FOREIGN KEY (`cid`) REFERENCES `conductor` (`cid`) ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (`regNo`) REFERENCES `bus` (`regNo`) ON DELETE CASCADE ON UPDATE CASCADE);

-- Inserting data for table `busschedule`

--

INSERT INTO `busschedule` (`bsid`, `path`, `cid`, `regNo`) VALUES

(1, 'Karwar-Gokarna-Bhatkal-Udupi-Mangalore', 5610, 'KA-O2 A-0985'),
 (2, 'Karwar-Gokarna-Bhatkal-Udupi-Mangalore', 5611, 'KA-O3 A-2379'),
 (3, 'Karwar-Bhatkal', 5613, 'KA-O4 A-7827'),
 (4, 'Bhatkal-Sagar-Shivamogga-Mangalore', 5616, 'KA-O4 H-5623'),
 (5, 'Mangalore-Udupi-Bhatkal-Gokarna-Karwar', 5615, 'KA-O4 S-8332'),
 (6, 'Mangalore-Shivamogga-Sagar', 5617, 'KA-O5 S-9429'),
 (7, 'Karwar-Gokarna-Bhatkal-Sirsi', 5618, 'KA-O7 H-0986'),

```
(8, 'Sagar-Sirsi', 5620, 'KA-O7 S-2483'),
(9, 'Udupi-Shivamogga-Sagar-Sirsi', 5621, 'KA-O4 S-8332'),
(10, 'Sirsi-Bhatkal', 5614, 'KA-O7 H-9886');
```

```
-----
```

```
--
```

```
-- Table structure for table `bshasroute`
```

```
--
```

```
CREATE TABLE `bshasroute` (
  `bsid` int(11) NOT NULL,
  `rid` int(11) NOT NULL,
  `departureTime` time NOT NULL,
  `arrivalTime` time NOT NULL,
  FOREIGN KEY (`bsid`) REFERENCES `busschedule` (`bsid`) ON DELETE
  CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (`rid`) REFERENCES `route` (`rid`) ON DELETE CASCADE ON
  UPDATE CASCADE);
```

```
--
```

```
-- Inserting data for table `bshasroute`
```

```
--
```

```
INSERT INTO `bshasroute` (`bsid`, `rid`, `departureTime`, `arrivalTime`) VALUES
(1, 1, '06:00:00', '07:00:00'),
(1, 2, '07:05:00', '08:30:00'),
(1, 3, '08:35:00', '10:00:00'),
(1, 4, '10:05:00', '11:00:00'),
(2, 1, '15:00:00', '16:00:00'),
(2, 2, '16:05:00', '17:30:00'),
(2, 3, '17:35:00', '19:00:00'),
(2, 4, '19:05:00', '20:00:00'),
(3, 1, '09:00:00', '10:00:00'),
(3, 2, '10:00:00', '11:30:00'),
(4, 7, '06:00:00', '07:00:00'),
(4, 8, '07:05:00', '08:00:00'),
(4, 10, '08:05:00', '09:00:00'),
(5, 17, '13:00:00', '14:00:00'),
(5, 18, '14:05:00', '15:00:00'),
(5, 19, '15:05:00', '16:00:00'),
(5, 20, '16:05:00', '17:00:00'),
(6, 11, '07:00:00', '08:00:00'),
(6, 13, '08:05:00', '09:00:00'),
(7, 1, '09:00:00', '10:00:00'),
(7, 2, '10:05:00', '11:00:00'),
(7, 5, '11:05:00', '14:00:00'),
(8, 6, '15:00:00', '17:00:00'),
(9, 9, '05:00:00', '07:00:00'),
(9, 13, '07:05:00', '08:00:00'),
(9, 15, '08:05:00', '08:30:00'),
(10, 16, '09:00:00', '12:30:00');
```

```

-----
-- --
-- Table structure for table `ticket`
--

CREATE TABLE `ticket` (
  `tid` int(11) PRIMARY KEY AUTO_INCREMENT NOT NULL,
  `email` varchar(30) NOT NULL,
  `bsid` int(11) NOT NULL,
  `seatNo` int(11) NOT NULL,
  `fare` int(11) NOT NULL,
  `origin` varchar(20) NOT NULL,
  `destination` varchar(20) NOT NULL,
  `departureTime` time NOT NULL,
  `arrivalTime` time NOT NULL,
  `date` date NOT NULL,
  AUTO_INCREMENT = 16,
  FOREIGN KEY (`email`) REFERENCES `passenger` (`
Email`) ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (`bsid`) REFERENCES `busschedule` (`bsid`) ON DELETE
CASCADE ON UPDATE CASCADE);

--
-- Inserting data for table `ticket`
--

INSERT INTO `ticket` (`tid`, `email`, `bsid`, `seatNo`, `fare`, `origin`, `destination`,
`departureTime`, `arrivalTime`, `date`) VALUES
(13, 'shashankkrishnanaik@gmail.com', 1, 1, 153, 'KARWAR', 'BHATKAL', '06:00:00',
'08:30:00', '2020-12-14'),
(14, 'shashankkrishnanaik@gmail.com', 1, 2, 153, 'KARWAR', 'BHATKAL', '06:00:00',
'08:30:00', '2020-12-15');

--
-- Table structure for table `cancelledt`
--

CREATE TABLE `cancelledt` (
  `cancelledtid` int(11) NOT NULL,
  `email` varchar(30) NOT NULL,
  `cancelledtime` datetime NOT NULL,
  `bsid` int(11) NOT NULL,
  `seatNo` int(11) NOT NULL,
  `origin` varchar(20) NOT NULL,
  `destination` varchar(20) NOT NULL,
  `reserveddate` date NOT NULL, FOREIGN KEY (`email`) REFERENCES `passenger`
(`Email`) ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (`bsid`) REFERENCES `busschedule` (`bsid`) ON DELETE
CASCADE ON UPDATE CASCADE
);
--

```

```

-- Inserting data for table `cancelledt`
--
INSERT INTO `cancelledt` (`cancelledtid`, `email`, `cancelledtime`, `bsid`, `seatNo`,
`origin`, `destination`, `reserveddate`) VALUES
(15, 'shashankkrishnanaik@gmail.com', '2020-12-15 11:28:28', 1, 1, 'KARWAR',
'BHATKAL', '2020-12-16');

-----
--
-- Triggers `ticket`
--
DELIMITER $$
CREATE TRIGGER `ctd`
BEFORE DELETE ON `ticket`
FOR EACH ROW
insert into cancelledt values (old.tid,
old.email,now(),old.bsid,old.seatNo,old.origin,old.destination,old.date)
$$
DELIMITER ;
--
-- Triggers `passenger`
--
DELIMITER $$
CREATE TRIGGER `cat` BEFORE INSERT ON `passenger` FOR EACH ROW BEGIN
if (new.age>=60) THEN
        set new.category='senior citizen';
    elseif (new.age>=18) THEN
        set new.category='major';
    else set new.category='minor';
END if;
END
$$
DELIMITER ;
-----
--
-- Procedures
--
DELIMITER $$
CREATE PROCEDURE `selectticket` ()
BEGIN
select * from ticket;
END$$
DELIMITER ;

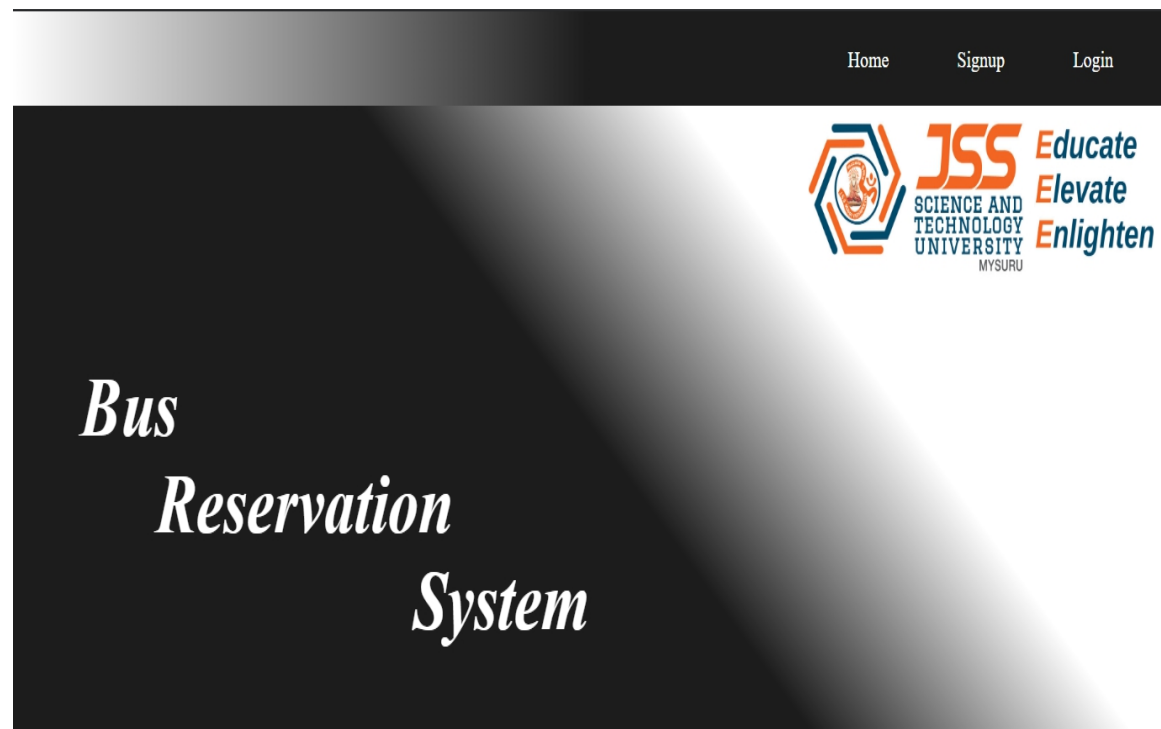
```

7 Interface View:

Link of Project files:

<https://github.com/sparshvrao/Bus-Reservation>


Home page:



Sign-up page:

The screenshot shows the sign-up page of the Bus Reservation System. At the top, there is a dark navigation bar with the JSS Science and Technology University logo on the left and links for 'Home', 'Signup', and 'Login' on the right. The main content area has a dark background with a diagonal light gradient. In the center, there is a white sign-up form with a light gray border. The form contains the following fields: 'Email' with the value 'abc@gmail.com', 'Password' with masked characters '*****', 'Name' with the value 'abc efghij', 'Gender' with radio buttons for 'male' (selected), 'female', and 'unknown', 'Phone No' with the value '1000000005', and 'Age' with a dropdown menu showing '19'. A blue 'Submit' button is located at the bottom of the form.

Log -In:



[Home](#)
[Signup](#)
[Login](#)

Email :

Password :

Login

[change password](#)

Bus Reservation:



[Home](#)
[Book Ticket](#)
[Reservation Details](#)
[Cancel Ticket](#)

Bus Reservation System

Email

Date

Origin

Destination

Get Bus Details





Home

Book Ticket

Reservation Details

Cancel Ticket

Email

abc@gmail.com

Date

12/27/2020

Origin

KARWAR

Destination

BHATKAL


Get Bus Details

Bus Schedule ID	Departure Time	Arrival Time	Fare (in Rs)
1	06:00 AM	08:30 AM	153
2	03:00 PM	05:30 PM	153
3	09:00 AM	11:30 AM	153
7	09:00 AM	11:00 AM	153

1	12	23	34	45
2	13	24	35	46
3	14	25	36	47
4	15	26	37	48
5	16	27	38	49
6	17	28	39	50
7	18	29	40	51
8	19	30	41	52
9	20	31	42	53
10	21	32	43	54
11	22	33	44	

Total amt :306

submit



Home

Book Ticket

Reservation Details


Cancel Ticket

Ticket ID	Email	Bus RegNo	Seat No	Fare (in Rs)	Origin	Destination	Departure Time	Arrival Time	Date
17	abc@gmail.com	KA-O2 A-0985	2	153	KARWAR	BHATKAL	06:00 AM	08:30 AM	2020-12-27
18	abc@gmail.com	KA-O2 A-0985	3	153	KARWAR	BHATKAL	06:00 AM	08:30 AM	2020-12-27

Please Remember your Ticket ID

Book another ticket

Reservation Details:



Home

Book Ticket

Reservation Details

Cancel Ticket

Tid: 17

Get

Ticket ID

17

Email

abc@gmail.com

Bus RegNo

KA-O2 A-0985

Seat No

2

Fare (in Rs)

153

Origin

KARWAR

Destination

BHATKAL

Departure Time

06:00 AM

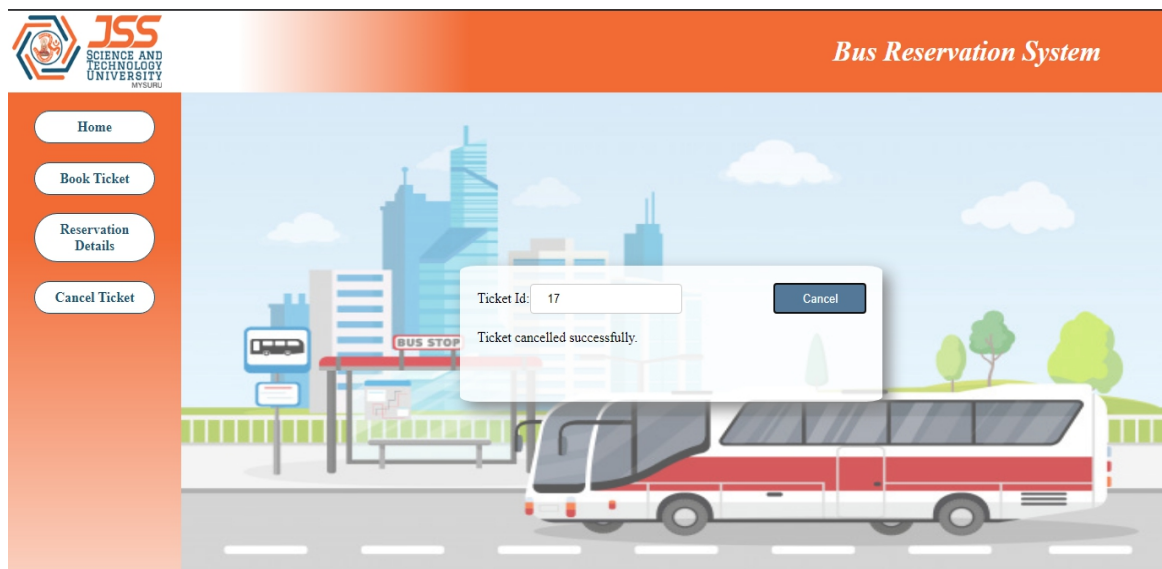
Arrival Time

08:30 AM

Date

2020-12-27

Cancel Ticket:



8 Conclusion:

What we learned?

How to implement the database in real world scenario.

We learned Database functionalities like Triggers, Stored Procedures and knowledge about nested queries.

Learned how to connect database to the back-end and data transfer between front-end and back-end.

Importance of concurrency protocols as real-time applications depends on that.

What problems did we faced and how we solved it?

Minimizing of redundant data from the tables: -

By eliminating repeating groups in individual tables and by creating a separate table for each set of related data.

Calculation of fares for the intermediate routes: -

We created a separate table containing a point-to-point fare and by using the sum function to obtain the specified route, the fare was calculated.

What are the functionalities that we implemented/added?

During cancellation, trigger is added: -

We added a trigger to store the details like timestamp and ticket date in a separate table so that further analysis can be done.

Validation of data: -

We had included basic validations in the front-end itself so that bandwidth is not wasted to send invalid data.

9 Reference:

a. Fundamentals of Database Systems

-Ramez Elmasri

-Shamkant B. Navathe

b. <https://www.w3schools.com/>

c. <https://stackoverflow.com/>

d. <https://www.geeksforgeeks.org/>