CS530

DATABASE
MANAGEMENT
SYSTEM

ONLINE TICKECT BOOKING SYSTEM

Nagabasavanna S
01JST19CS085

Mohammed Raqim Bidchol
01JST19CS084

Instructor:
Dr. VIJAYALAXMI
Associate Professor
JSS Science and Technology University
Mysore

Divakara N
Assistant Professor
JSS Science and Technology University
Mysore

january 17, 2021

CONTENTS:

1 Abstract:

In this project we have 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.

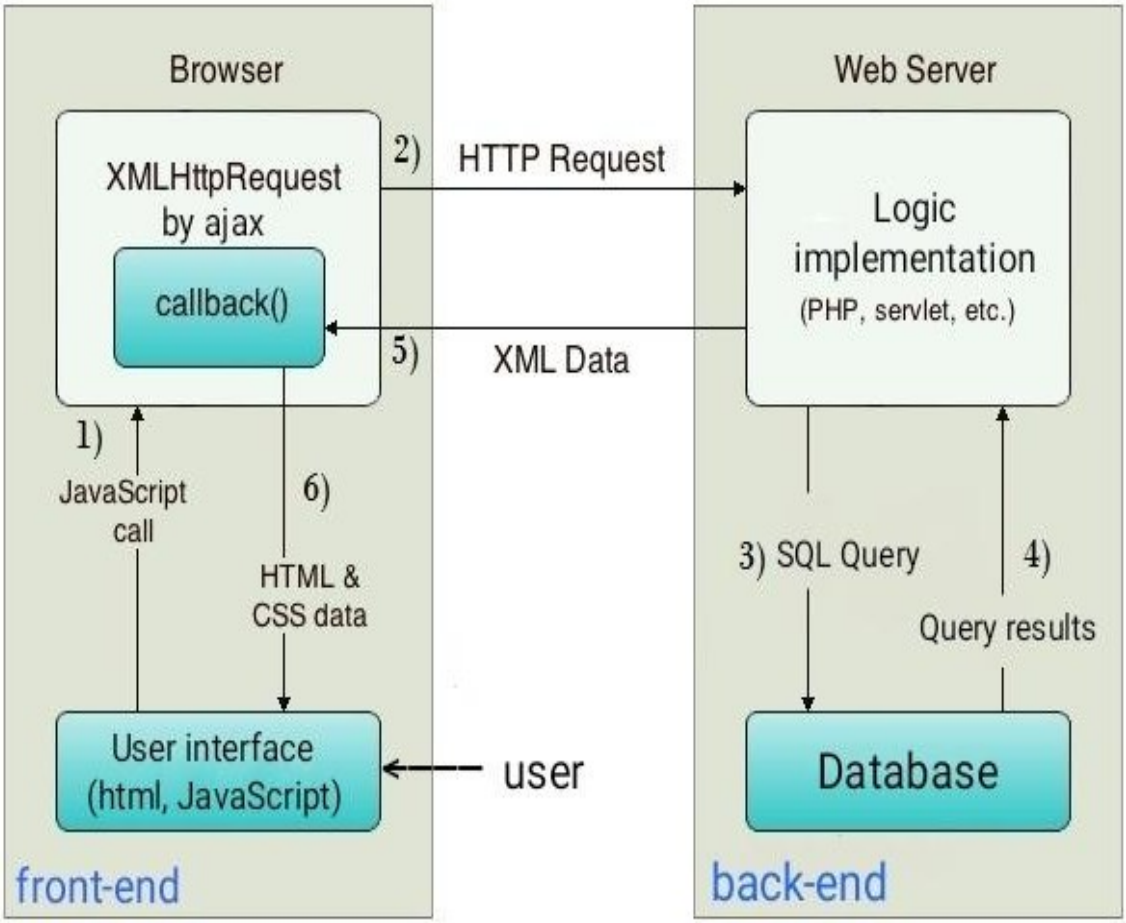
Back-end processes:

1. When the passenger enters the registration details it is validated in the website and transferred to the back-end 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

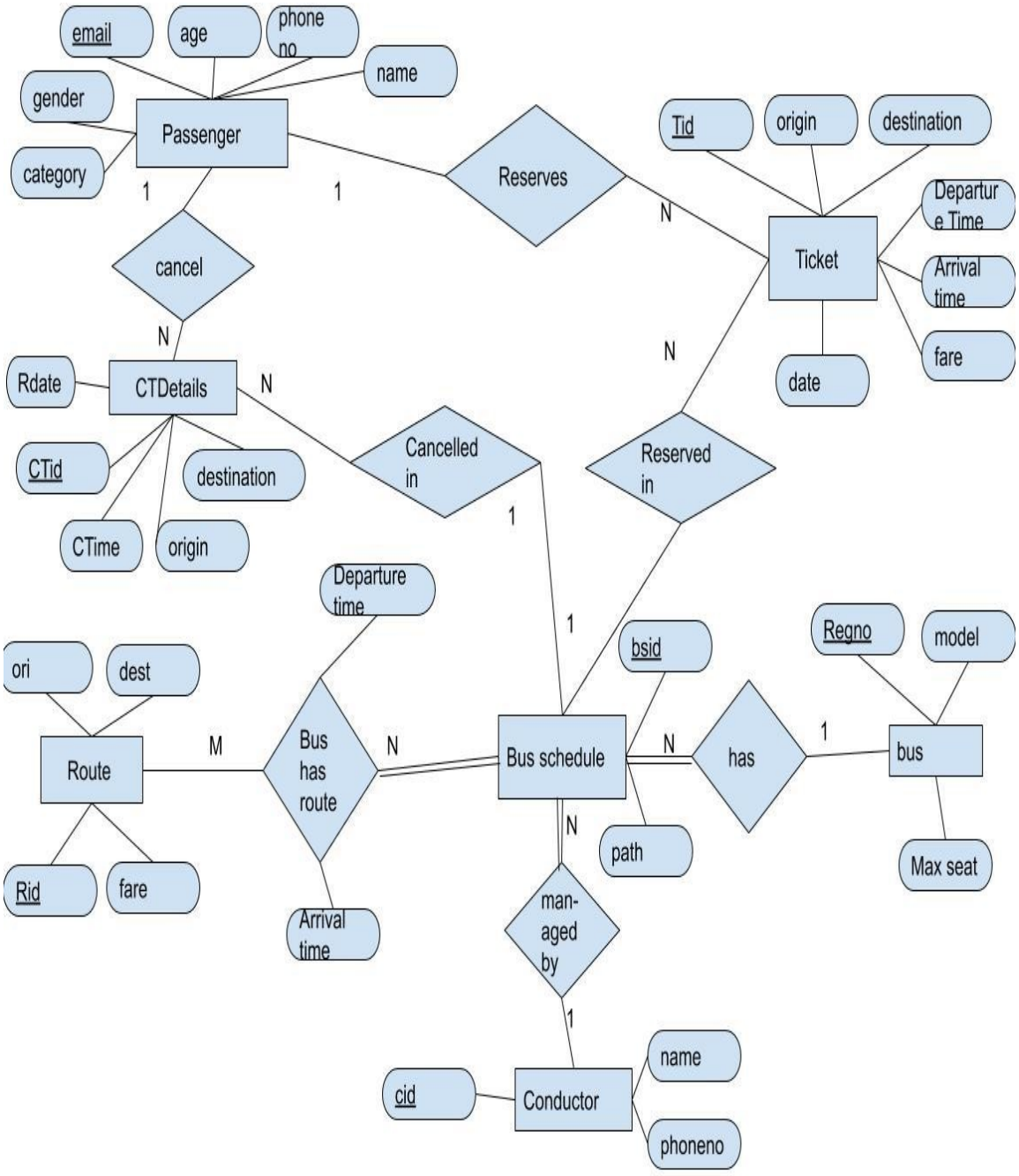
calculated by checking the tickets reserved in the ticket table for the particular bus schedule and displayed to the user.

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.

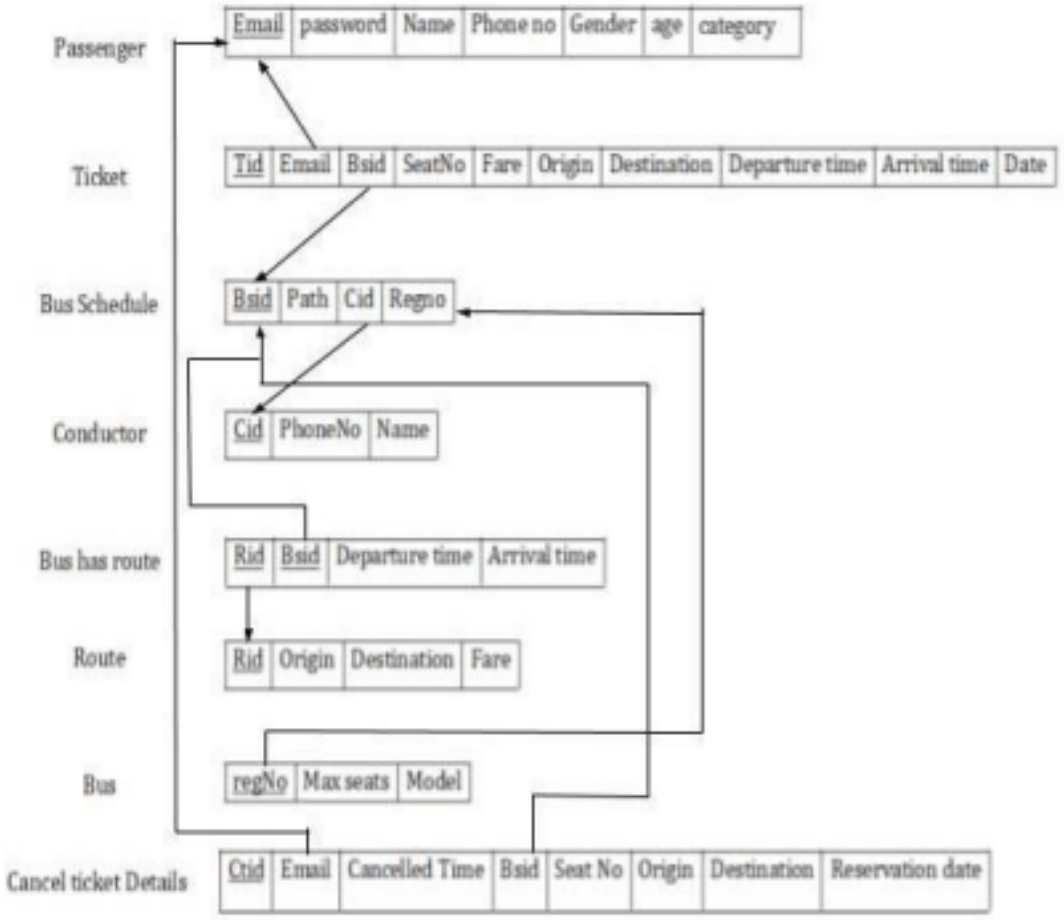
Interaction between front-end and back-end:



3 ER Diagram:

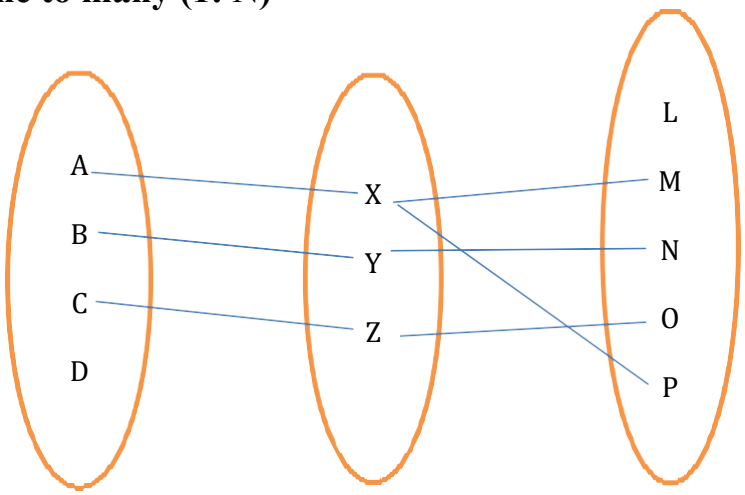

email
age
phone no
name
gender
Passenger
category
1
1
Reserves
Tid
origin
destination
N
Ticket
Departure Time
Arrival time
fare
date
cancel
N
Rdate
CTDetails
N
CTid
destination
CTime
origin
Cancelled in
1
N
Reserved in
Departure time
1
bsid
Regno
model
ori
dest
Route
M
Bus has route
N
Bus schedule
N
has
1
bus
path
Max seat
N
Rid
fare
Arrival time
man-aged by
1
cid
Conductor
name
phoneno

4 Schema:



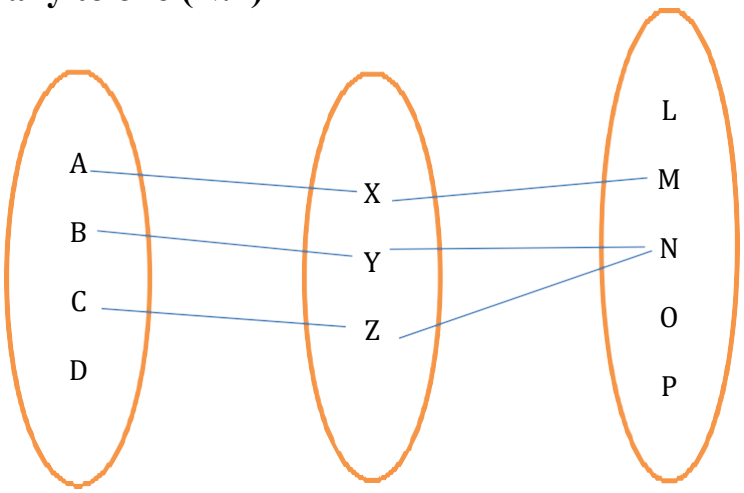
5 Set representation

a.one to many (1: N)



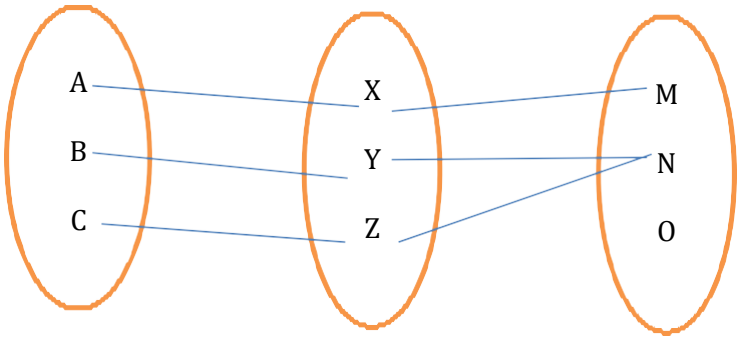
Passenger Reserves Ticket
Passenger Cancels Ticket

b.many to one (N:1)



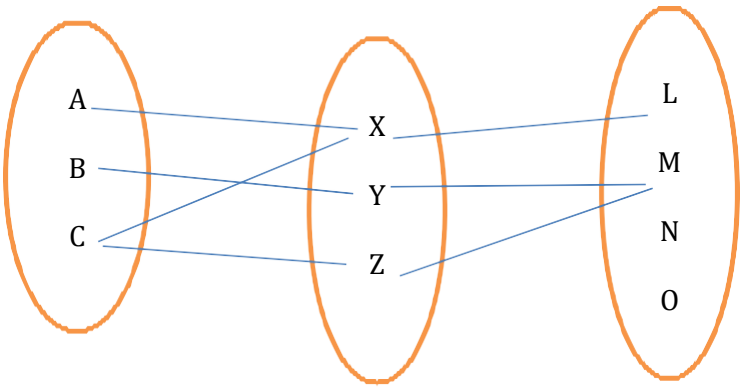
Ticket — Reserved in — Bus schedule
Ticket — Cancelled in — Bus schedule

c.many to one (N:1) with total participation



Bus schedule — has — Bus
Bus schedule — managed by — Conductor

d.many to many (N:M) with total participation



Bus schedule — has — Route

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');

--------------------------------------------------------
--
-- Table structure for table `conductor`
--

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'),
```

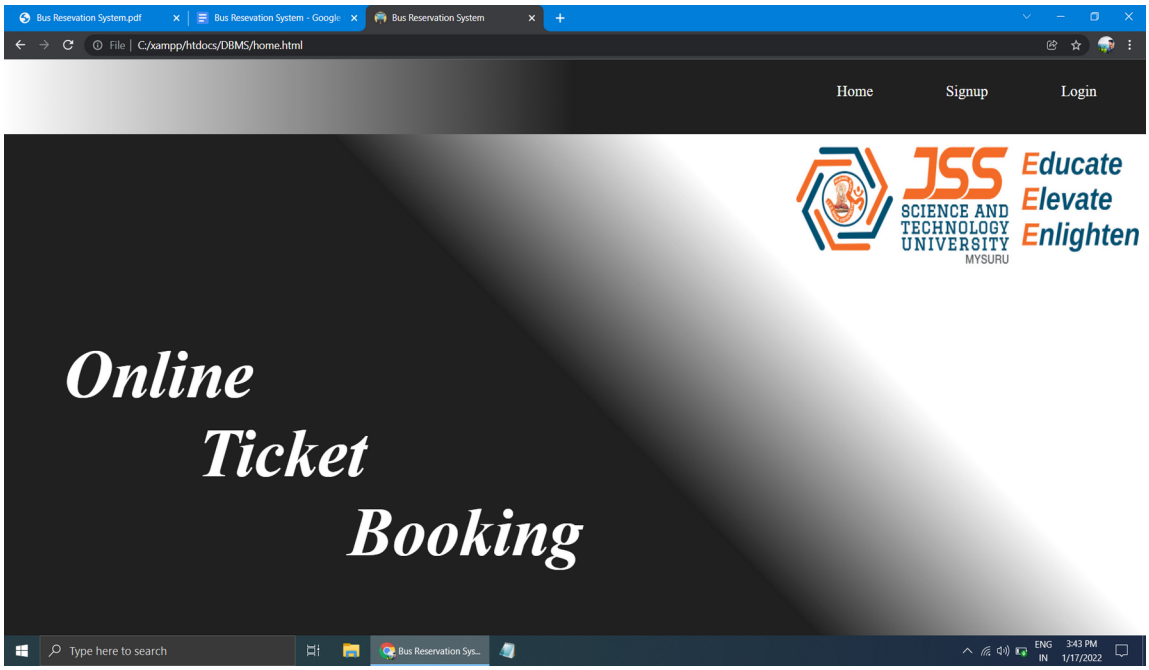
```
(5, 'Mangalore-Udupi-Bhatkal-Gokarna-Karwar', 5615, 'KA-O4 S-8332'),
(6, 'Mangalore-Shivamogga-Sagar', 5617, 'KA-O5 S-9429'),
(7, 'Karwar-Gokarna-Bhatkal-Sirsi', 5618, 'KA-O7 H-0986'),
(8, 'Sagar-Sirsi', 5620, 'KA-O7 S-2483'),
(9, 'Udupi-Shivamogga-Sagar-Sirsi', 5621, 'KA-O4 S-8332'),
(10, 'Sirsi-Bhatkal', 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
```

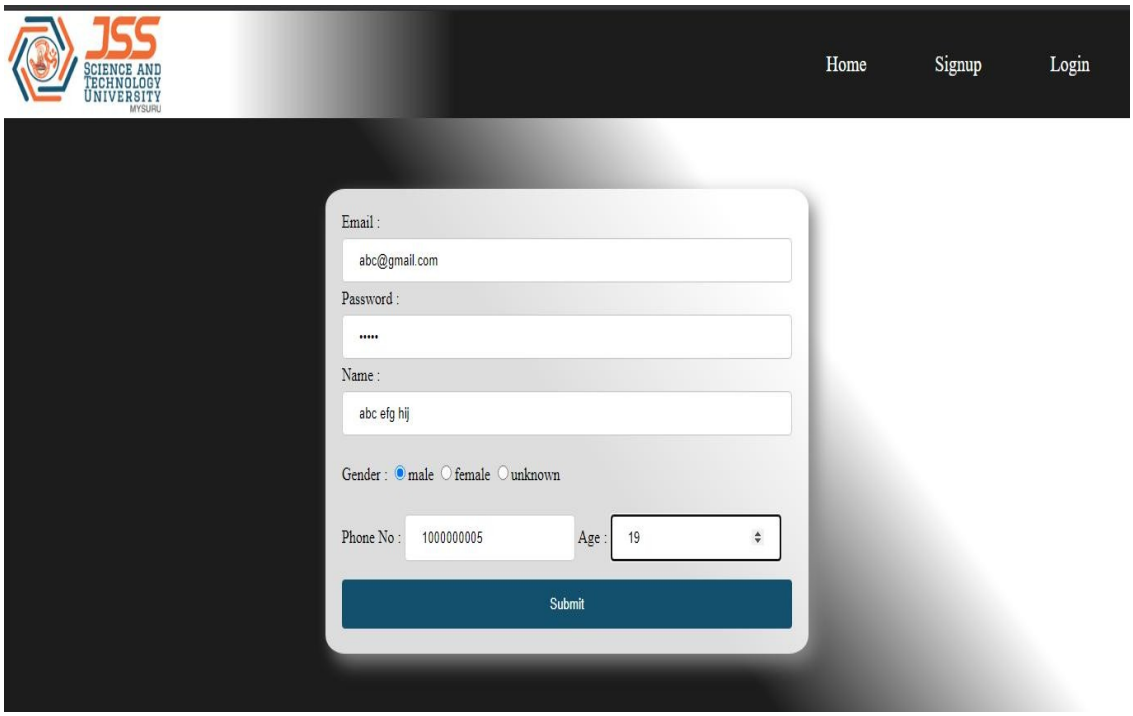
```
CASCADE ON UPDATE CASCADE
);
--
-- Inserting data for table `cancelledt`
--
INSERT INTO `cancelledt` (`cancelledtid`, `email`, `cancelledtime`, `bsid`, `seatNo`,
`origin`, `destination`, `reserveddate`) VALUES
(15, 'shashankkrishnanaik@gmail.com', '2020-12-15 11:28:28', 1, 1, 'KARWAR',
'BHATKAL',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

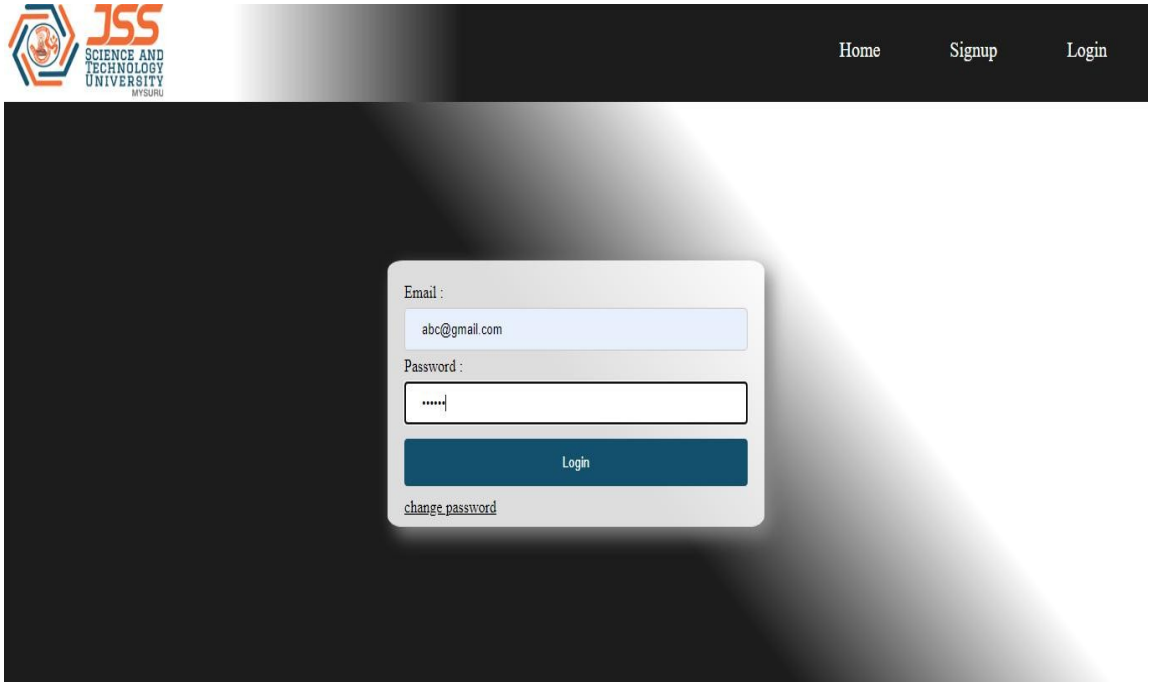
Home page:



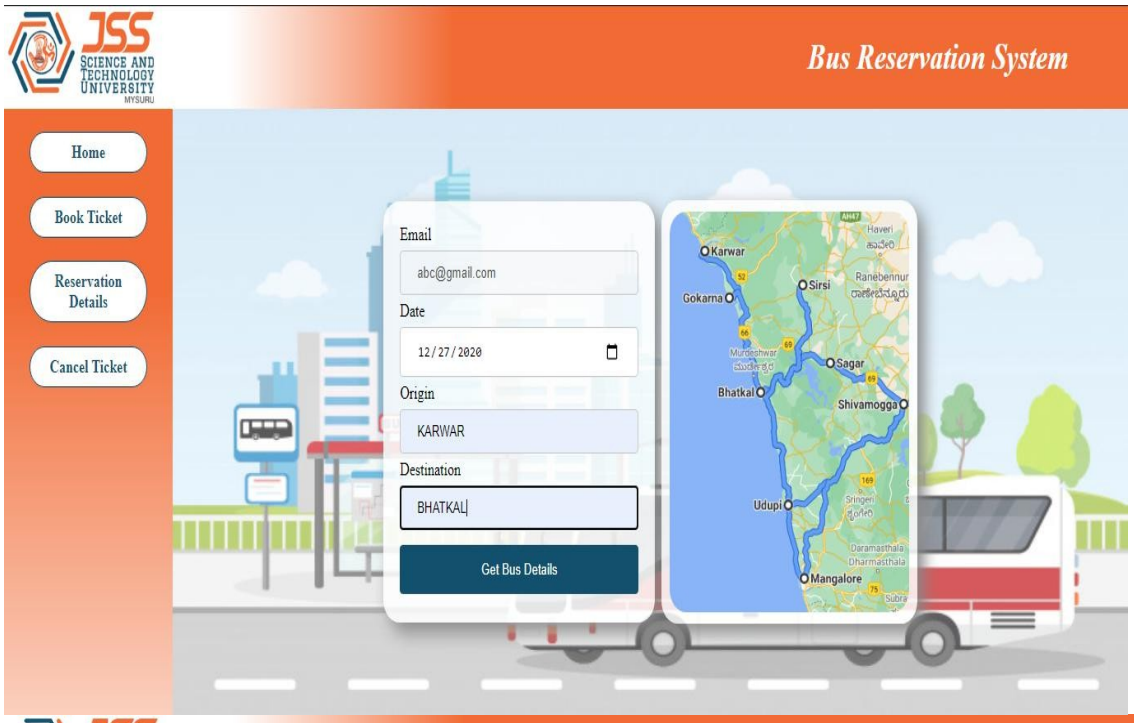
Sign-up page:

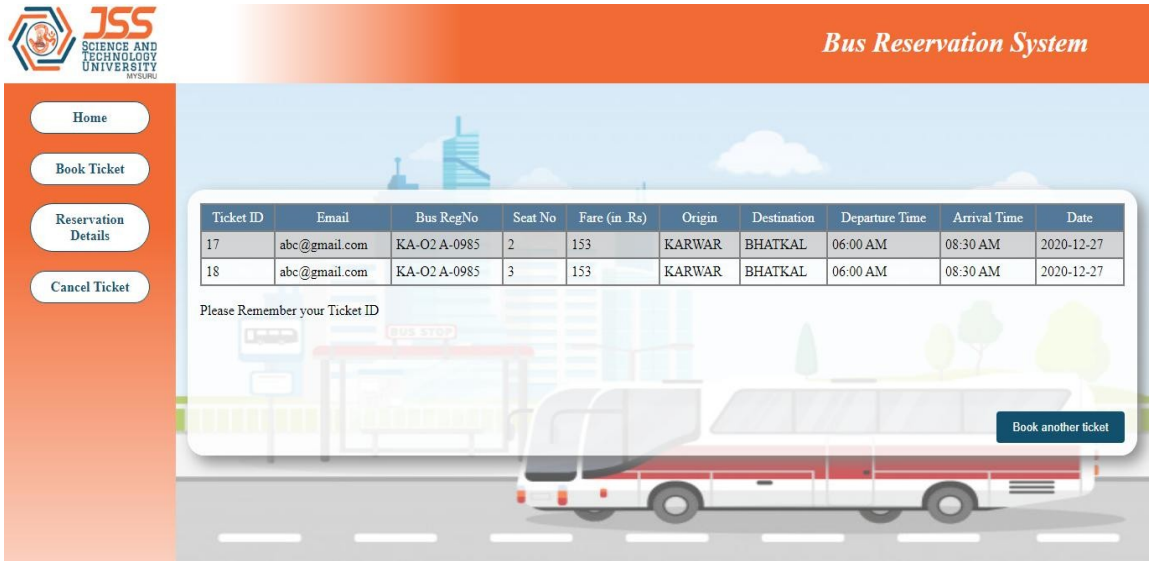


Log -In:



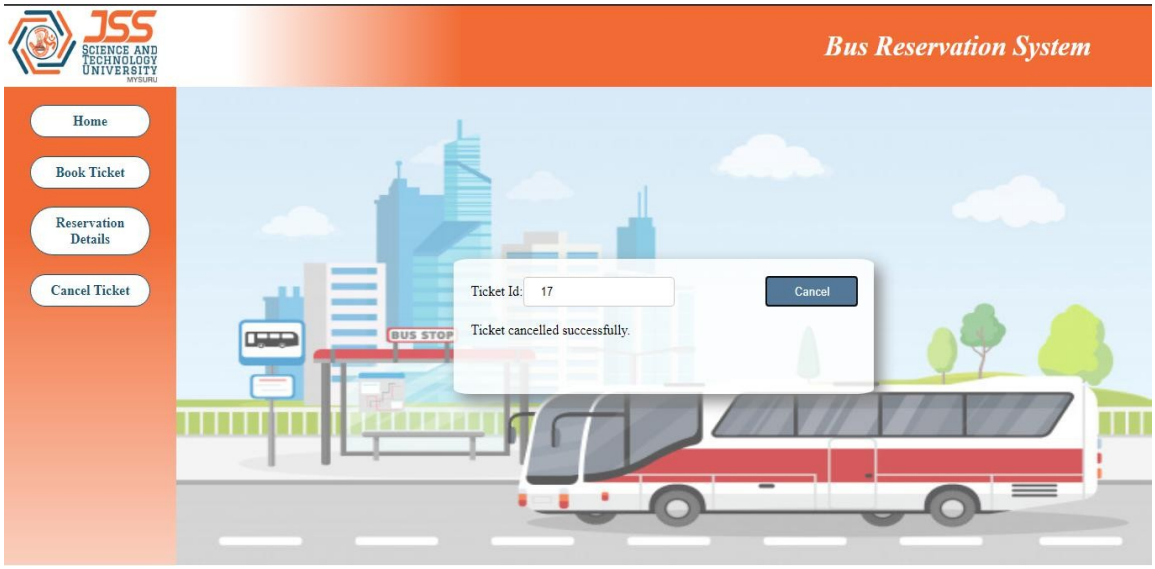
Bus Reservation:





Reservation Details:

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/