CSN-351
DATABASE MANAGEMENT SYSTEM
Bus Booking Management System Project

Group Members:

1.	Balne Niteesha	18114017
2.	Diksha	18114019
3.	Karan Singh	18114035
4.	Kavya Barnwal	18114039
5.	Khushi	18114040

.

Abstract

Traveling is a large growing business across all countries. Bus reservation system deals with maintenance of records of details of each passenger. It also includes maintenance of information like schedule and details of each bus.

We observed the working of the Bus reservation system and after going through it, we got to know that there are many operations, which they have to do manually. It takes a lot of time and

causes many errors while data entry. Due to this, sometimes a lot of problems occur and they are facing many disputes with customers. To solve the above problem, and further maintain records of passenger details, seat availability, bus availability and other things, we are offering this proposal of a computerized reservation system.

By using this software, we can reserve tickets from any part of the world, through telephone lines, via the internet. Customers can check availability of buses and reserve selective seats. The project provides and checks all sorts of constraints so that the user does give only useful data and thus validation is done in an effective way.

Introduction

The focus of the project is to computerize traveling companies to manage data, details of customers, details of various buses so as to ease companies' tasks and shifting the ticket booking process to an online platform for the easier access of customers. It reduces the possibility of errors or any discrepancy in any kind of data. It replaces all the paperwork.

This bus booking management system has three modules.:

- First module helps the customer to login or register into the system.
- Second module helps to inquire about the availability of seats in a particular bus at a particular date. It helps him to reserve a ticket or cancel a reserved ticket.
- There is a third module which allows the admin of the system to add seats to various buses and to view customers databases.

Features of the system

The system is very simple in design and to implement. The system requires very low system resources and the system will work in almost all configurations. It has got following features:

✓Availability of seats can be enquired very easily.

✓Passengers can also cancel their tickets easily.

✓Minimum time needed for the various processing

✓Better Service

✓Ensures data accuracy.

✓Records are efficiently maintained by DBMS.

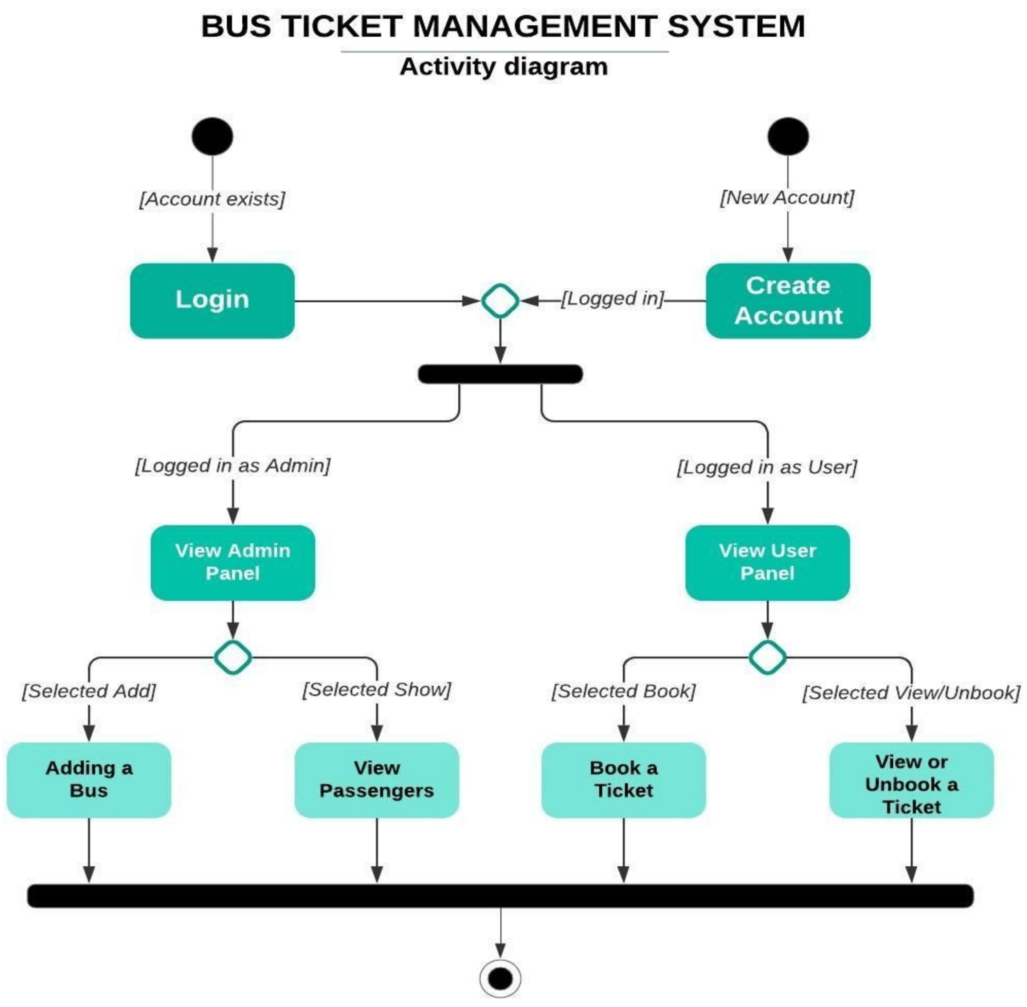
✓DBMS also provides security for the information.

✓Any person across the world, with required setup can access this service.

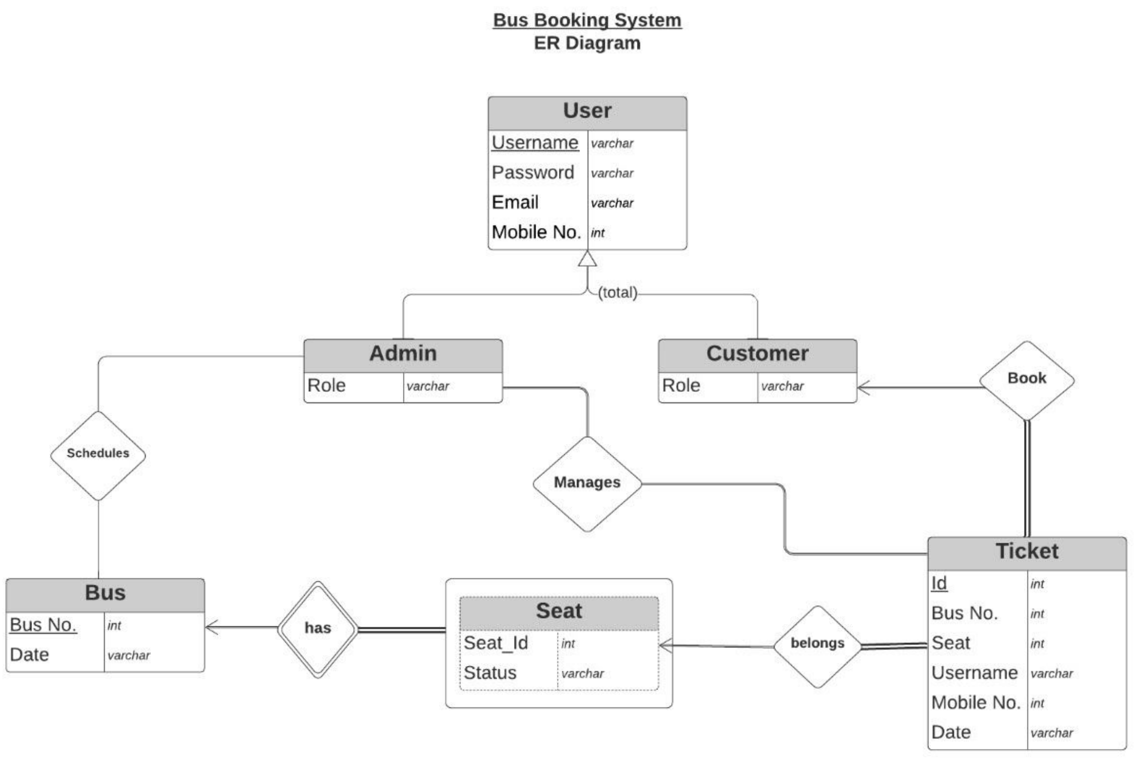
✓This would help the corporation prepare and organize its schedules more efficiently on the basis of traffic demand.

- **Hardware Requirements:**
    - PC with Pentium IV processor. (optional)
    - 512 MB RAM or above.
    - 40 GB Hard Disk or above.
    - Java Virtual Machine
- **Software Requirements:**
    - Operating system : Windows (optional)
    - Front end : Java Runtime
    - Integrated Development Environment(IDE) : Netbeans
    - Server to host the database on machine : XAMPP
    - JDBC driver : MySQL Connector

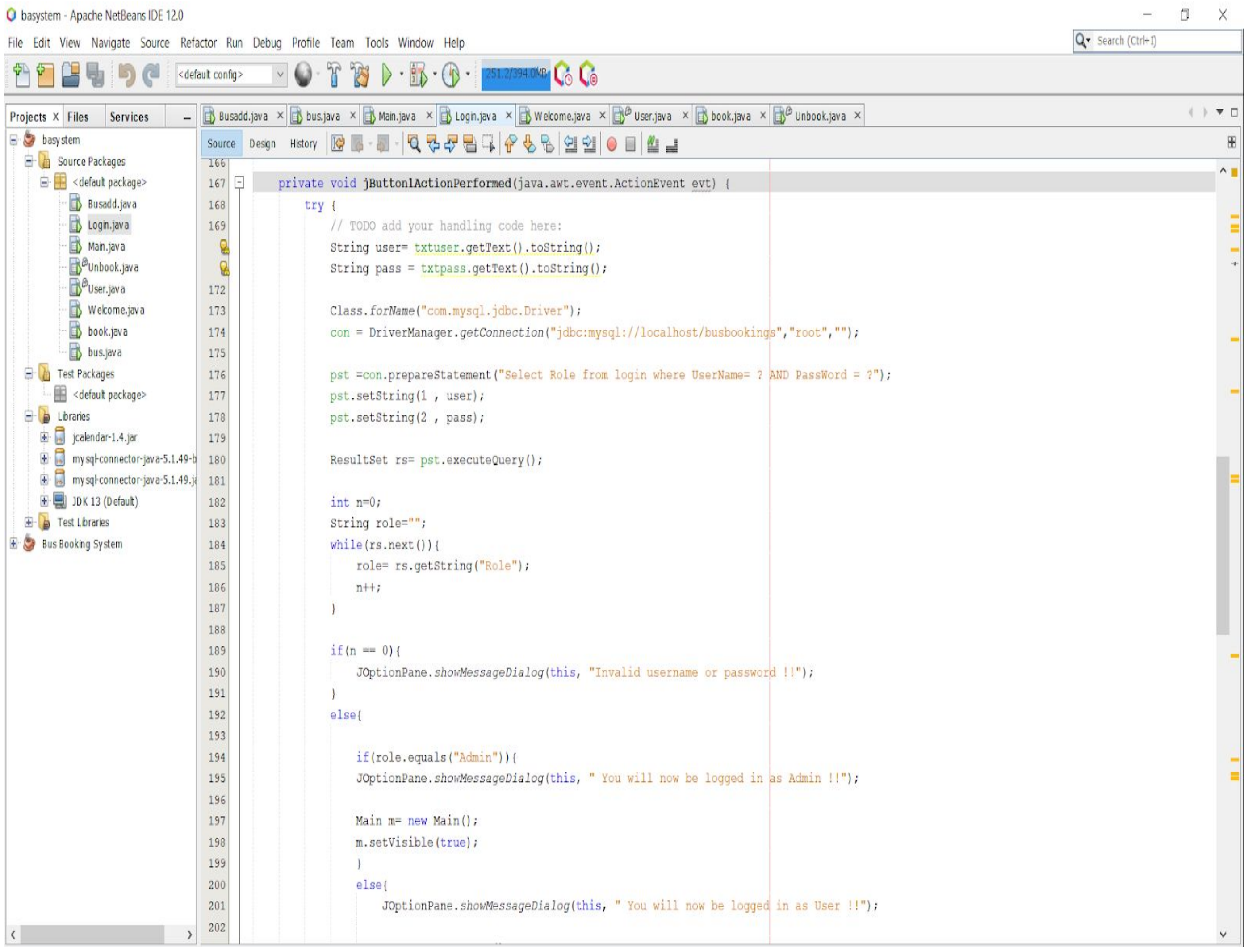
Activity Diagram:



ER Diagram:


Bus Booking System
ER Diagram
User
Username varchar
Password varchar
Email varchar
Mobile No. int
(total)
Admin
Role varchar
Customer
Role varchar
Book
Schedules
Manages
Ticket
Id int
Bus No. int
Seat int
Username varchar
Mobile No. int
Date varchar
Bus
Bus No. int
Date varchar
has
Seat
Seat_Id int
Status varchar
belongs

Code Snippets:



```java
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        try {
            // TODO add your handling code here:
            String user= txtuser.getText().toString();
            String pass = txtpass.getText().toString();

            Class.forName("com.mysql.jdbc.Driver");
            con = DriverManager.getConnection("jdbc:mysql://localhost/busbookings","root","");

            pst =con.prepareStatement("Select Role from login where UserName= ? AND PassWord = ?");
            pst.setString(1 , user);
            pst.setString(2 , pass);

            ResultSet rs= pst.executeQuery();

            int n=0;
            String role="";
            while(rs.next()){
                role= rs.getString("Role");
                n++;
            }

            if(n == 0){
                JOptionPane.showMessageDialog(this, "Invalid username or password !!");
            }
            else{

                if(role.equals("Admin")){
                JOptionPane.showMessageDialog(this, " You will now be logged in as Admin !!");

                Main m= new Main();
                m.setVisible(true);
                }
                else{
                    JOptionPane.showMessageDialog(this, " You will now be logged in as User !!");

```

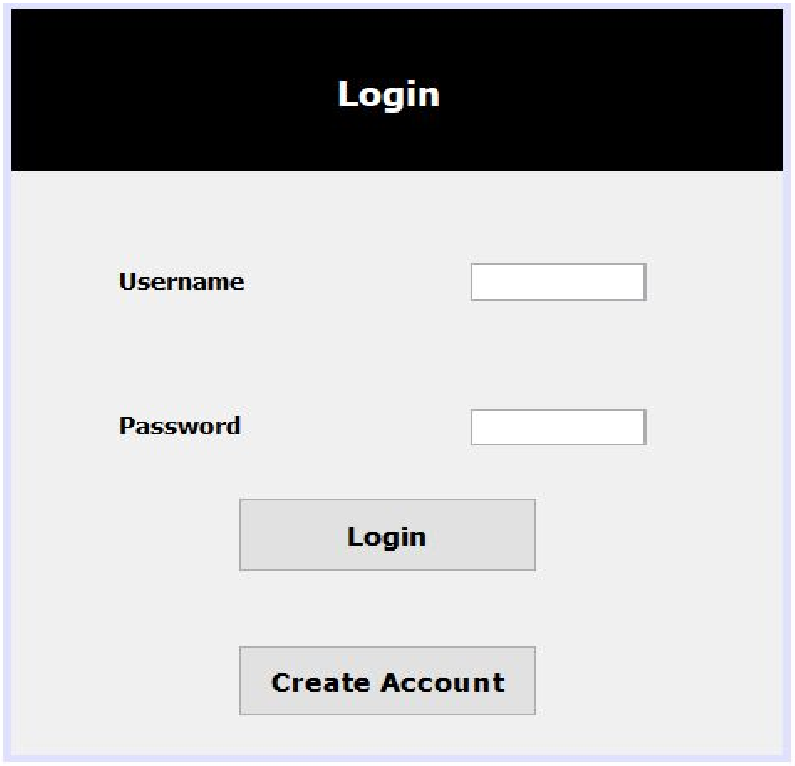
System:

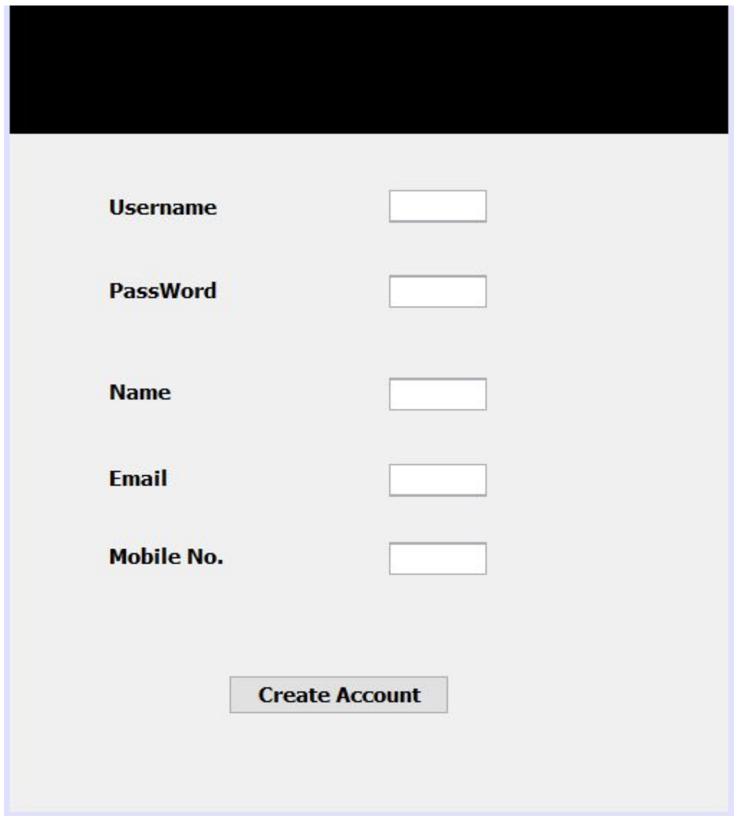
Login and create account windows:

You can either login by giving your credentials or create a new account . If the username or password is wrong or such an entry does not exist in the database , a window pops up showing the message

“Invalid username or password!” . Also , if a username is already taken while creating a new account , a window pops up with the message:

“Username already taken!” .

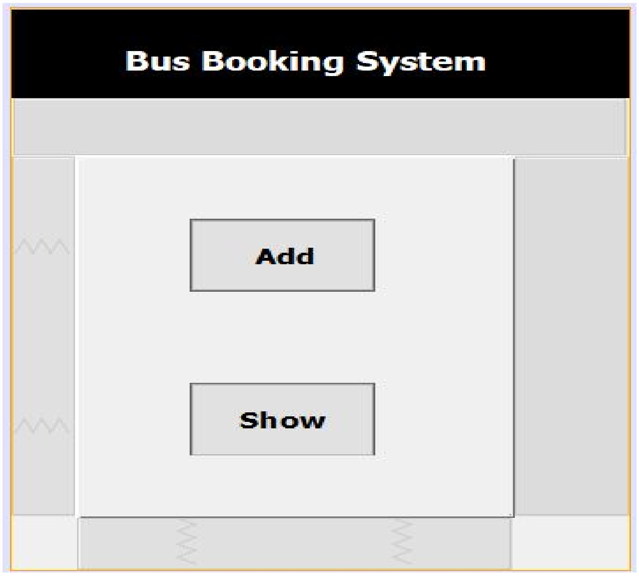


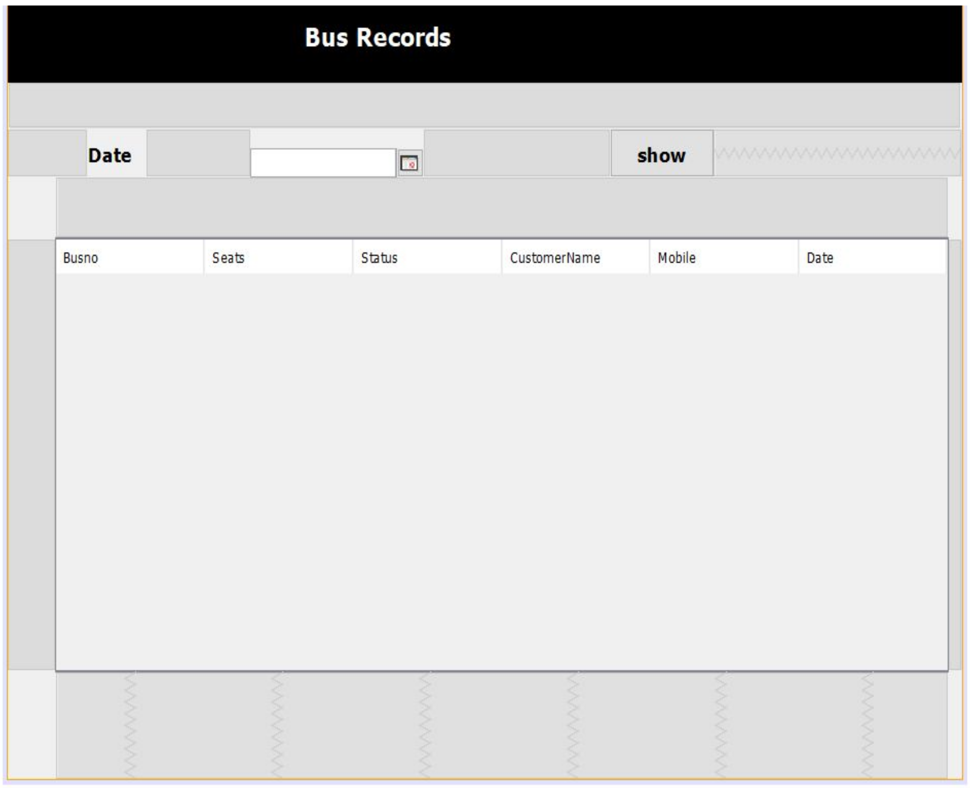


Admin View:

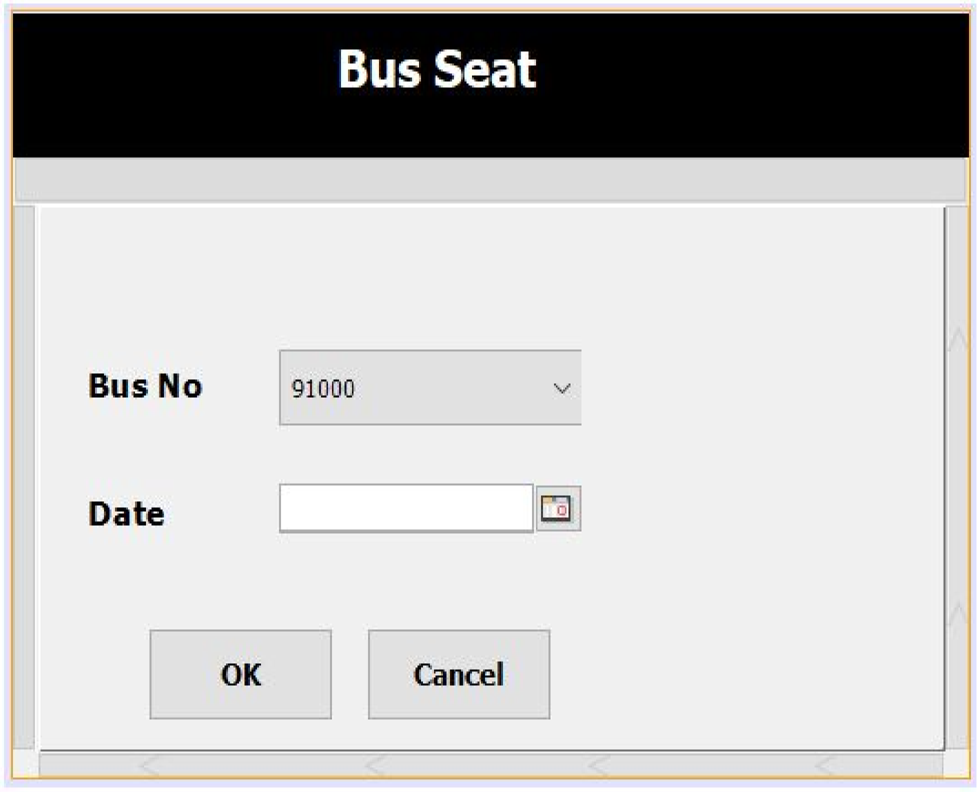
An admin manages the whole system , so he can add buses available on a particular date and the status of their seats in the database , with the help of the “Add” button , after which the Add Seat window opens .

The admin can also view all the bookings of any bus on any particular day , with the help of the “Show” button , after which , the Bus Records window opens up.





Busno	Seats	Status	CustomerName	Mobile	Date



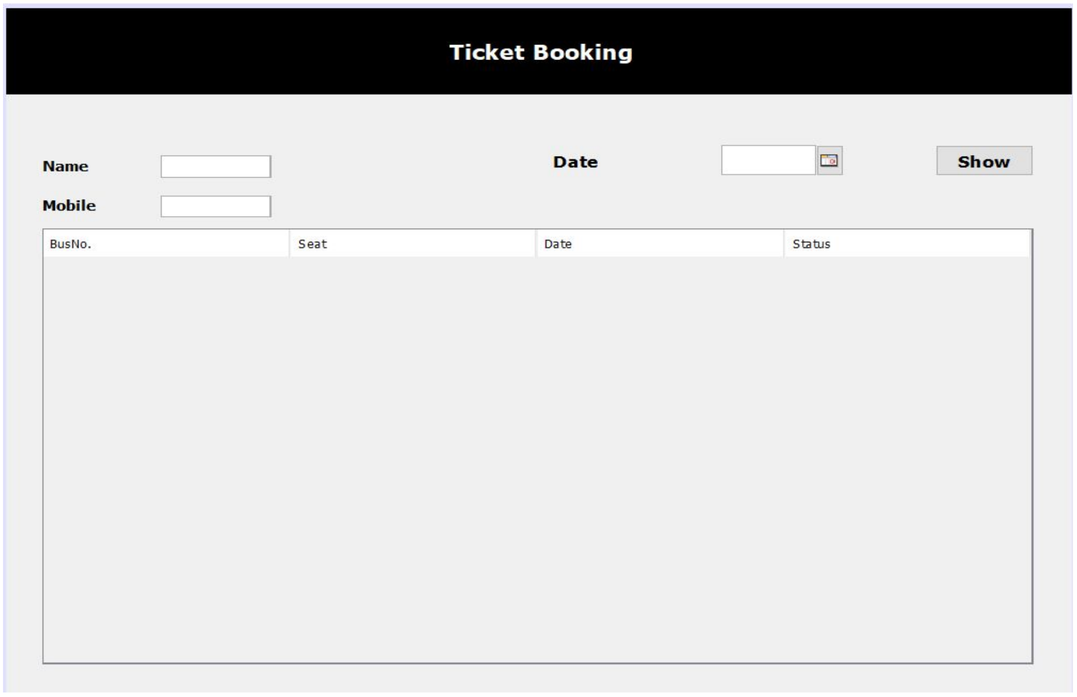
Bus No 91000

Date

User View:

A user can view all buses and seats available on the particular day , and also book them for himself or any relative (name and mobile No. required) , with the help of Ticket Booking window .

He/She can also unbook any of his/her booking for a particular date , booked under any name (which needs to be provided as input) , in the View Your Bookings window .

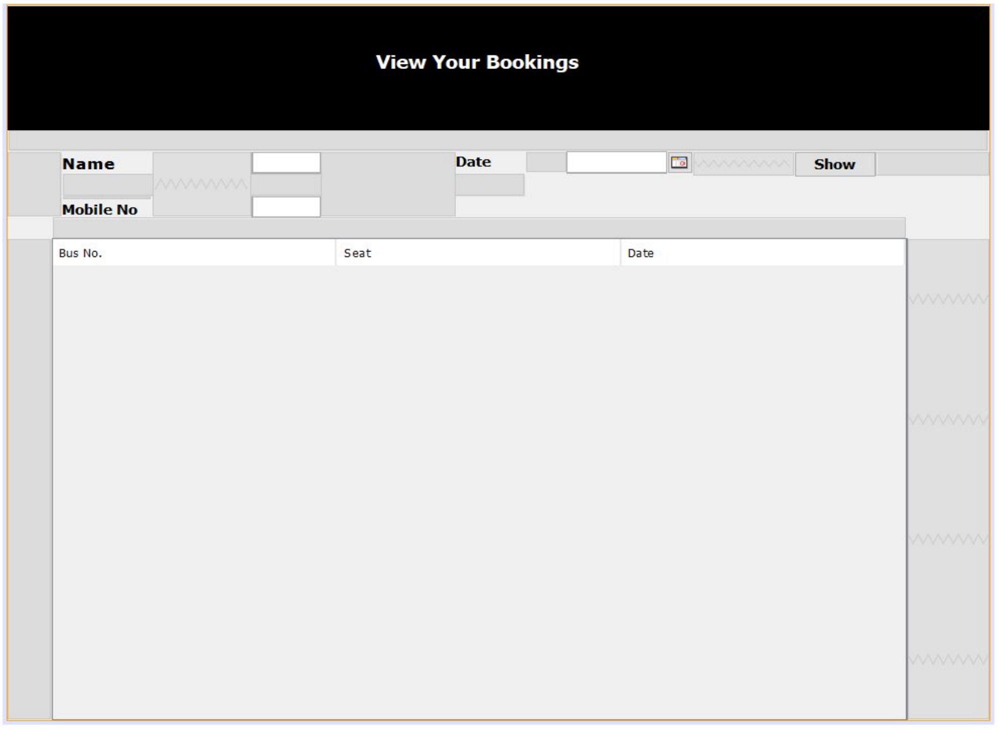


Ticket Booking

Name	
Mobile	
Date	

Show

BusNo.	Seat	Date	Status



View Your Bookings

Name	
Mobile No	
Date	

Show

Bus No.	Seat	Date

Queries Used:

1) Select Queries :

   a) SELECT Role

      FROM login

      WHERE UserName= ? AND PassWord = ?

   b) SELECT seat.Busno, seat.Seats, seat.Status, busbooked.UserName , busbooked.MobileNo, seat.Date

      FROM seat

      LEFT JOIN busbooked ON seat.BusNo=busbooked.BusNo AND seat.Seats = busbooked.Seat AND busbooked.Date = seat.Date

      WHERE seat.Date = date

   c) SELECT seat.Busno, seat.Seats,seat.Status,seat.Date

      FROM seat

      WHERE seat.status = status AND seat.Date = date

   d) SELECT busbooked.BusNo , busbooked.Seat , busbooked.Date

      FROM busbooked

      WHERE Date= date AND UserName = uname AND MobileNo = mno

2) Insert Queries :

   a) INSERT INTO seat(BusNo , Seats , Date, Status)

      VALUES(bno , sno , date , status)

   b) INSERT INTO login(UserName , PassWord , Name , Email , MobileNo , Role)

      VALUES(uname , pass , name , email , mno , role)

3) Update Queries:

   UPDATE seat

   SET Status = status

   WHERE BusNo = bno AND Seats = seats AND Date= date

4) Delete Queries :

   DELETE FROM busbooked

   WHERE BusNo = bno AND Seat = sno AND UserName= uname AND MobileNo = mno AND Date= date

Normalizations:

Normalization is the process of minimizing redundancy from a relation or set of relations. Redundancy in relation may cause insertion, deletion and updation anomalies. So, it helps to minimize the redundancy in relations. Normal forms are used to eliminate or reduce redundancy in database tables.

Normal Forms:

First Normal Form (1NF)	A relation is in first normal form if every attribute in that relation is a single-valued attribute. If a relation contains a composite or multi-valued attribute, it violates first normal form.
Second Normal Form (2NF)	To be in second normal form, a relation must be in first normal form and relation must not contain any partial dependency. A relation is in 2NF if it has no Partial Dependency, i.e., no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table.
Third Normal Form (3NF)	A relation is in third normal form, if there is no transitive dependency for non-prime attributes as well as it is in second normal form. A relation is in 3NF if at least one of the following condition holds in every non-trivial functional dependency X -> Y - X is a super key. - Y is a prime attribute (each element of Y is part of some candidate key).
Boyce-Codd Normal Form (BCNF)	A relation R is in BCNF if R is in Third Normal Form and for every FD, LHS is super key. A relation is in BCNF iff in every non-trivial functional dependency X -> Y, X is a super key.

Relation tables:

- Table name - busbooked
    - Table contains the information of the tickets booked.
    - Attributes: Id, BusNo, Seat, Name, MobileNo, Date
    - Primary key - Id
- Table name - login
    - Table contains the details of Admin and Users.
    - Attributes: UserName, PassWord, Email, MobileNo, Role
    - Primary key - UserName
- Table name - seat
    - Table contains the information of the tickets booked.
    - Attributes: Id, BusNo, Seats, Date, Status
    - Primary key - Id

Illustration of attributes:

- busbooked.Id - attribute issued to a seat while booking. [auto_increment mode]
- seat.Id - attribute allotted to a seat, which is solely dependent on the sequence of buses added (by the Admin). [auto_increment mode]
- BusNo - unique identifier of a bus.
- Seat - unique identifier of a seat in a bus (can be same for different buses).
- Name - name of the passenger (not necessarily belonging to User).
- busbooked.MobileNo - contact number of the passenger (not necessarily belonging to User).
- Date - date when the bus runs.
- UserName - unique identifier of the User.
- PassWord - attribute used by the User for logging in.
- Email - mail address of the User.

- login.MobileNo - contact number of the User.
- Role - Admin or User as values
- Status - booked or unbooked as values.

Input constraints:

Name, Email and MobileNo are taken as input with single values. (And all other attributes are single valued by default.)

Proving 1NF, 2NF, 3NF and BCNF:

- Table *busbooked* satisfies the conditions of-
    - 1NF, Since:
        - No repeating groups (composite or multi-valued attribute)
        - Primary Key identified
    - 2NF, Since:
        - 1NF
        - No partial dependencies
            - Since all attributes individually represent ticket parameters.
    - 3NF, Since:
        - 2NF
        - No transitive dependency
            - Since only one dependency exists.
    - BCNF, Since:
        - 3NF
        - Every determinant is a candidate key.
            - Since only one dependency exists.
- Table *login* satisfies the conditions of-
    - 1NF, Since:
        - No repeating groups (composite or multi-valued attribute)
        - Primary Key identified
    - 2NF, Since:
        - 1NF
        - No partial dependencies
            - Since all attributes individually give the details of the User.

    - 3NF, Since:
        - 2NF
        - No transitive dependency
            - Since only one dependency exists.
    - BCNF, Since:
        - 3NF
        - Every determinant is a candidate key.
            - Since only one dependency exists.
- Table *seat* satisfies the conditions of-
    - 1NF, Since:
        - No repeating groups (composite or multi-valued attribute)
        - Primary Key identified
    - 2NF, Since:
        - 1NF
        - No partial dependencies
            - Since all attributes individually represent seat parameters
    - 3NF, Since:
        - 2NF
        - No transitive dependency
            - Since only one dependency exists.
    - BCNF, Since:
        - 3NF
        - Every determinant is a candidate key.
            - Since only one dependency exists.

Conclusion:

This project includes a user-friendly interface for Bus ticket management, with optimised queries and normalized databases. They make the system more efficient to store and organize data than spreadsheets; also allow a centralized facility such that the data can easily be accessed, modified and quickly shared among multiple users.