****

**CS23333 – Object Oriented Programming using Java Project Report**

**BUS TICKET RESERVATION SYSTEM**

**A MINI PROJECT REPORT**

**Submitted by**

**SIVA SRUTHY VGP 231001205**

**VENISHA M 231001240**

In partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

IN

INFORMATION TECHNOLOGY

RAJALAKSHMI ENGINEERING COLLEGE (AUTONOMOUS)

THANDALAM

CHENNAI-602105

**2024-2025**

**BONAFIDE CERTIFICATE**

Certified that this project report **“BUS TICKET RESERVATION SYSTEM”** is the Bonafide

Work of **“SIVA SRUTHY VGP (231001205), VENISHA M (231001240)”**

who carried out the project work under my supervision.

**Submitted for the Practical Examination held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**SIGNATURE SIGNATURE**

**Dr.P.Valarmathie Mr.Narayana K.E**

**Professor and Head Assistant Professor,**

**Information Technology, Information Technology,**

**Rajalakshmi Engineering College Rajalakshmi Engineering College**

**(Autonomous), (Autonomous),**

**Thandalam, Chennai - 602 105. Thandalam, Chennai - 602 105.**

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**TABLE OF CONTENTS**

**1.INTRODUCTION**

1.1 INTRODUCTION

1.2 OBJECTIVES

1.3 MODULES

**2.SURVEY OF TECHNOLOGIES**

2.1 SOFTWARE DESCRIPTION

2.2 LANGUAGES

**3.REQUIREMENTS AND ANALYSIS**

3.1 REQUIREMENT SPECIFICATION

3.2 HARDWARE AND SOFTWARE REQUIREMENTS

3.3 ARCHITECTURE DIAGRAM

3.4 ER DIAGRAM

**4.PROGRAM CODE**

**5.RESULTS AND DISCUSSION**

**6.CONCLUSION**

**7.REFERENCES**

**ABSTRACT**

The Bus Ticket Reservation System aims to provide an efficient platform for reserving bus tickets. This system allows users to view bus details, check seat availability, and confirm bookings. The system also prevents overbooking by verifying seat availability before confirming a reservation. It stores bus information such as bus numbers, capacity, and whether the bus is air-conditioned. The system is developed using Java programming and MySQL as the database. It is designed to handle multiple buses and reservations, ensuring users can make bookings based on real-time availability.

**1.INTRODUCTION**

* 1. **INTRODUCTION**

The Bus Ticket Reservation System is a software application designed to manage bus ticket bookings efficiently. This system simplifies the process of booking and managing bus tickets, ensuring users can reserve seats on buses for specific dates. The system takes into account bus details such as bus numbers, capacities, and whether the buses are air-conditioned. It also allows passengers to book tickets, ensuring no double booking occurs by checking the availability of seats on specific buses and dates. By automating the booking process, the system enhances the overall experience for both the users and administrators.

The system is implemented using Java and MySQL for data management. The core functionalities include bus information management, passenger booking, and availability checks, which allow the system to simulate real-time reservations.

* 1. **OBJECTIVES**
* **Efficient Booking Process**: To provide a simple and user-friendly interface for passengers to book bus tickets with ease.
* **Availability Management**: To ensure no double bookings are made by checking the availability of seats for each bus on a specific date.
* **Bus Information Management**: To manage and display information about buses, including bus numbers, seat capacity, and AC/non-AC options.
* **Database Integration**: To utilize MySQL for storing and retrieving bus and booking data, ensuring a scalable and reliable solution.
* **Interactive Frontend**: To provide a dynamic and responsive user interface using HTML, CSS, and JavaScript.
* **Error Handling**: To ensure that invalid inputs, such as incorrect dates or bus numbers, are handled gracefully.
* **Real-time Reservation Updates**: To keep the system updated with real-time booking information and seat availability.
  1. **MODULES**

1. **Bus Module**:This module handles the management of bus information, such as bus number, AC availability, and total capacity.It includes methods to add new buses, display bus details, and modify bus information.
2. **Booking Module**:The booking module is responsible for managing passenger bookings, including storing the passenger name, bus number, and date of travel.It includes a function to check seat availability before confirming a booking.
3. **Frontend Module**:This module, built with HTML, CSS, and JavaScript, handles user interaction.It provides features such as displaying bus details, accepting booking inputs, and displaying confirmation or error messages.
4. **Database Module**:The database module connects to MySQL using JDBC and stores all relevant data, such as bus details and booking information.It retrieves and updates data to ensure real-time accuracy of bus availability.

**2.SURVEY OF TECHNOLOGIES**

**2.1 SOFTWARE DESCRIPTION**

The Bus Ticket Reservation System uses Java for backend development and MySQL for database management, ensuring efficient handling of data and operations. JDBC facilitates seamless interaction between Java and MySQL. A lightweight frontend, built using Java Swing, provides basic user interaction.

**2.2LANGUAGES:**

1. Java  
   Java is the core programming language used for backend development. It ensures efficient logic implementation, object-oriented structure, and smooth database interactions through JDBC.
2. SQL  
   SQL, supported by MySQL, manages data storage, retrieval, and updating, forming the relational database backbone for bus and booking information.
3. JDBC  
   JDBC bridges the Java application with the MySQL database, executing queries and maintaining data integrity.

**3.REQUIREMENTS AND ANALYSIS**

**3.1 REQUIREMENT SPECIFICATION**

**1) FUNCTIONAL REQUIREMENTS:**

- User registration and login system.

- Database for storing scheme information.

- Search and filter functionality.

- Eligibility checking tool.

- Application guidance and steps.

- Notification system.

**2)NON-FUNCTIONAL REQUIREMENTS:**

- Scalability to handle a growing number of users and schemes.

- Security measures to protect user data.

- High availability and reliability.

**3.2 HARDWARE AND SOFTWARE REQUIREMENTS**

**1)HARDWARE:**

- Server with sufficient CPU and RAM to handle multiple requests.

- Storage for database and backups.

**2)SOFTWARE:**

- Operating System: Linux/Windows

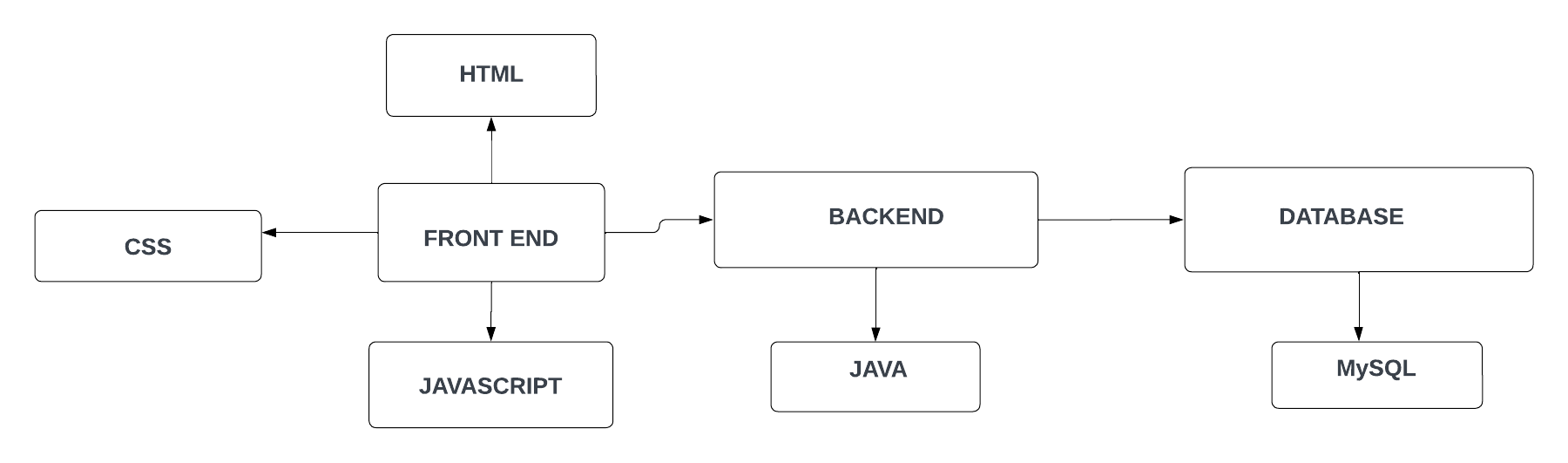
- Database: MySQL/PostgreSQL

- Backend: Python with Flask/Django

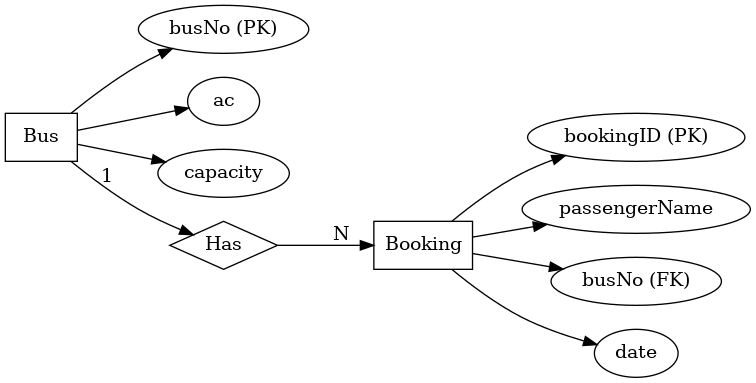
- Frontend: Java Swing

- Version Control: Git

**3.3 ARCHITECTURE DIAGRAM**

****

**3.4 ER DIAGRAM**

****

**4.PROGRAM CODE**

**JAVA CODE (JDBC CONNECTION):**

package BusRes;

import java.sql.\*;

import java.util.ArrayList;

import java.util.List;

public class DatabaseConnection {

    private static final String URL = "jdbc:mysql://localhost:3306/bus\_reservation";

    private static final String USER = "root";

    private static final String PASSWORD = "Venisha@123";

    public static Connection getConnection() {

        Connection connection = null;

        try {

            Class.forName("com.mysql.cj.jdbc.Driver");

            connection = DriverManager.getConnection(URL, USER, PASSWORD);

            if (connection != null) {

                System.out.println("Connection to the database established successfully!");

            }

        } catch (ClassNotFoundException e) {

            System.out.println("MySQL JDBC Driver not found!");

            e.printStackTrace();

        } catch (SQLException e) {

            System.out.println("Connection failed! Check the console for details.");

            e.printStackTrace();

        }

        return connection;

    }

    public static void closeConnection(Connection connection) {

        try {

            if (connection != null) {

                connection.close();

                System.out.println("Connection closed successfully!");

            }

        } catch (SQLException e) {

            System.out.println("Error closing connection.");

            e.printStackTrace();

        }

    }

    // Insert initial bus data for testing

    public static void insertInitialBusData() {

        Connection connection = getConnection();

        if (connection != null) {

            String insertBusQuery = "INSERT INTO bus (id, ac, capacity) VALUES (?, ?, ?)";

            try (PreparedStatement preparedStatement = connection.prepareStatement(insertBusQuery)) {

                // Insert Bus 1 with capacity 2 for testing

                preparedStatement.setInt(1, 1);

                preparedStatement.setBoolean(2, true);  // AC bus

                preparedStatement.setInt(3, 2);         // Capacity 2

                preparedStatement.executeUpdate();

                // Insert other buses (you can add more if needed)

                preparedStatement.setInt(1, 2);

                preparedStatement.setBoolean(2, false);  // Non-AC bus

                preparedStatement.setInt(3, 50);         // Capacity 50

                preparedStatement.executeUpdate();

                preparedStatement.setInt(1, 3);

                preparedStatement.setBoolean(2, true);  // AC bus

                preparedStatement.setInt(3, 48);         // Capacity 48

                preparedStatement.executeUpdate();

                System.out.println("Initial bus data inserted successfully.");

            } catch (SQLException e) {

                System.out.println("Error inserting bus data.");

                e.printStackTrace();

            } finally {

                closeConnection(connection);

            }

        }

    }

    ///

    public static List<Bus> getBusDetails() {

        List<Bus> buses = new ArrayList<>();

        try (Connection connection = getConnection()) {

            String query = "SELECT \* FROM bus";

            Statement statement = connection.createStatement();

            ResultSet resultSet = statement.executeQuery(query);

            while (resultSet.next()) {

                int id = resultSet.getInt("id");

                boolean ac = resultSet.getBoolean("ac");

                int capacity = resultSet.getInt("capacity");

                buses.add(new Bus(id, ac, capacity));

            }

        } catch (SQLException e) {

            e.printStackTrace();

        }

        return buses;

    }

    public static void insertBookingData(String passengerName, int busNo) {

        try (Connection connection = getConnection()) {

            String query = "INSERT INTO booking (passenger\_name, bus\_no, travel\_date) VALUES (?, ?, CURDATE())";

            PreparedStatement preparedStatement = connection.prepareStatement(query);

            preparedStatement.setString(1, passengerName);

            preparedStatement.setInt(2, busNo);

            preparedStatement.executeUpdate();

            System.out.println("Booking for passenger " + passengerName + " on bus " + busNo + " has been successfully added.");

        } catch (SQLException e) {

            e.printStackTrace();

        }

    }

    }

**JAVA SWING CODE:**

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.\*;

import java.util.ArrayList;

import java.util.List;

import BusRes.DatabaseConnection;

public class BusReservationSwingApp {

    private JFrame frame;

    private JTable busTable;

    private JTextField passengerNameField;

    private JButton bookButton;

    public BusReservationSwingApp() {

        frame = new JFrame("Bus Reservation System");

        frame.setSize(600, 400);

        frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        frame.setLayout(new BorderLayout());

        // Fetch bus details from the database

        List<Bus> buses = DatabaseConnection.getBusDetails();

        // Create a table model for displaying bus details

        DefaultTableModel model = new DefaultTableModel();

        model.addColumn("Bus ID");

        model.addColumn("AC");

        model.addColumn("Capacity");

        // Populate the table with bus data

        for (Bus bus : buses) {

            model.addRow(new Object[]{bus.getBusNo(), bus.getAc() ? "Yes" : "No", bus.getCapacity()});

        }

        // Create JTable

        busTable = new JTable(model);

        busTable.setSelectionMode(ListSelectionModel.SINGLE\_SELECTION);

        JScrollPane scrollPane = new JScrollPane(busTable);

        frame.add(scrollPane, BorderLayout.CENTER);

        // Passenger name input

        JPanel panel = new JPanel();

        panel.setLayout(new FlowLayout());

        panel.add(new JLabel("Enter Your Name:"));

        passengerNameField = new JTextField(15);

        panel.add(passengerNameField);

        // Booking button

        bookButton = new JButton("Book Ticket");

        panel.add(bookButton);

        frame.add(panel, BorderLayout.SOUTH);

        // Add ActionListener to the book button

        bookButton.addActionListener(new ActionListener() {

            @Override

            public void actionPerformed(ActionEvent e) {

                // Get passenger name and selected bus details

                String passengerName = passengerNameField.getText();

                int selectedRow = busTable.getSelectedRow();

                if (passengerName.isEmpty()) {

                    JOptionPane.showMessageDialog(frame, "Please enter your name.");

                    return;

                }

                if (selectedRow == -1) {

                    JOptionPane.showMessageDialog(frame, "Please select a bus.");

                    return;

                }

                // Get bus ID from the selected row

                int busId = (int) busTable.getValueAt(selectedRow, 0);

                // Insert booking data into the database

                DatabaseConnection.insertBookingData(passengerName, busId);

                // Show success message

                JOptionPane.showMessageDialog(frame, "Booking Successful for Bus ID: " + busId);

            }

        });

        frame.setVisible(true);

    }

    public static void main(String[] args) {

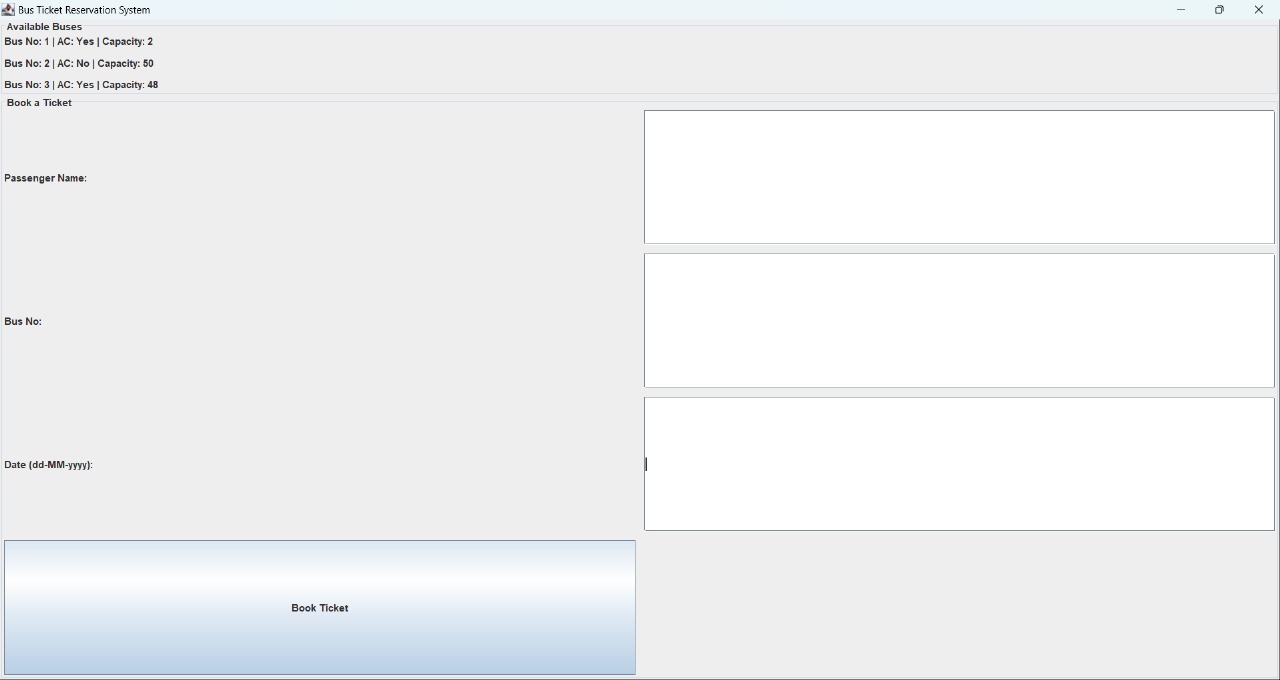
        new BusReservationSwingApp();

    }

}

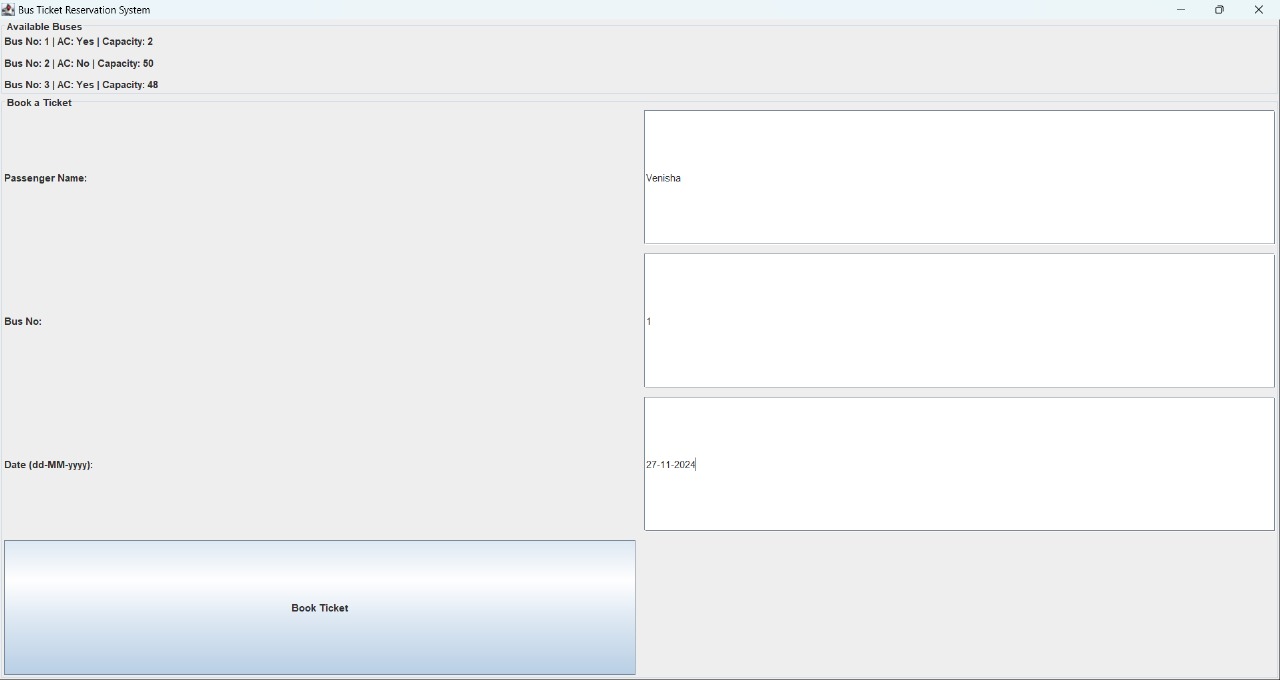
**5. RESULTS AND DISCUSSION:**

This page shows the available buses information.

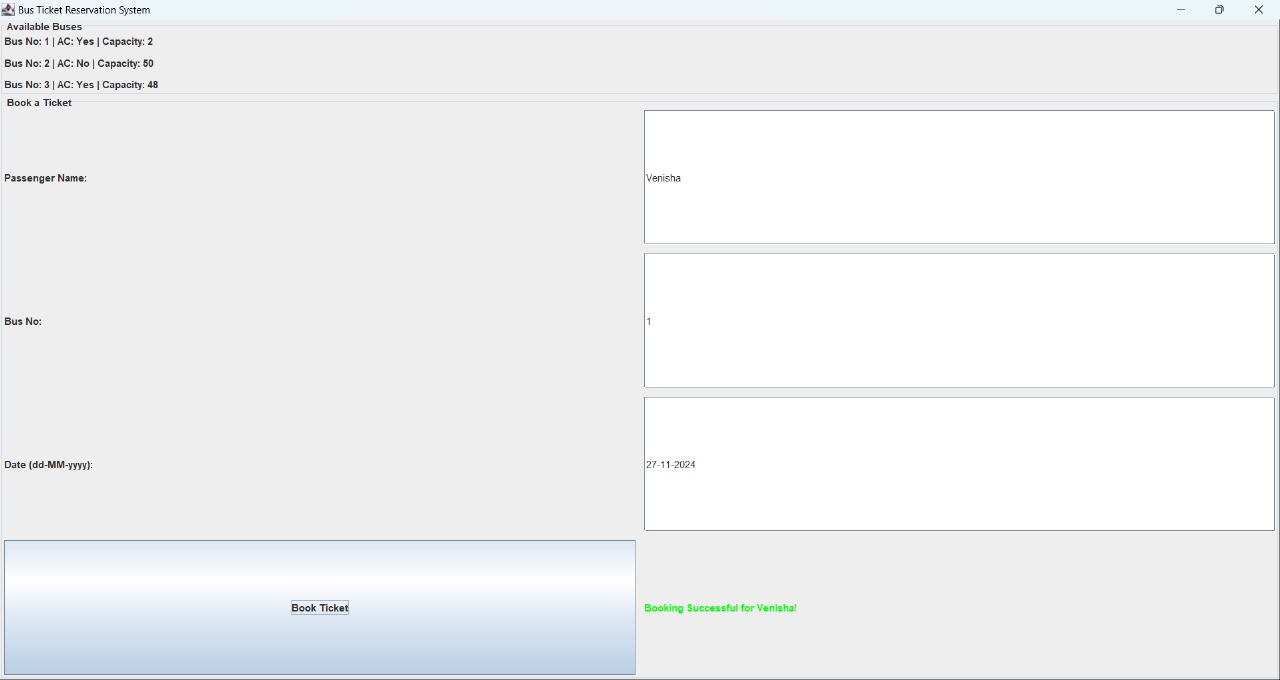


The user enters their information-name, bus selected, booking date.

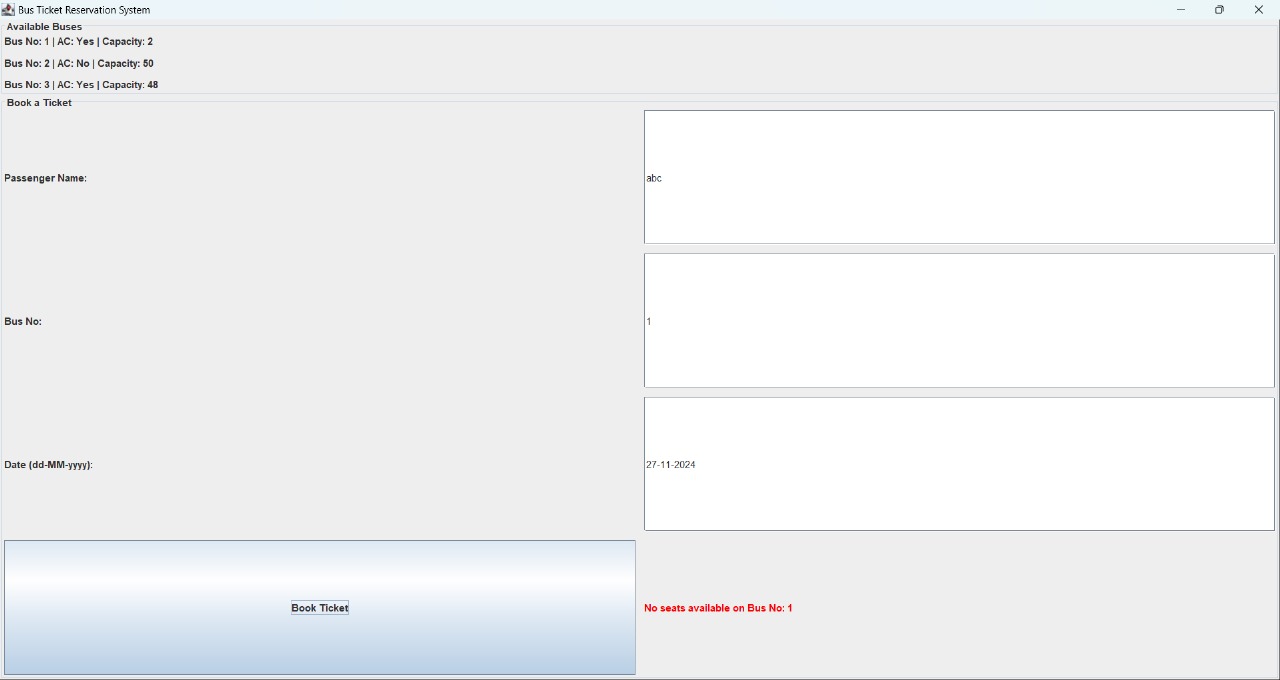
Then the user clicks the Book now button.



Now, if the Booking is confirmed, the message “Booking successful” is displayed.



If the bus is fully booked on the selected date, the message “No seats available” is displayed.



**6.CONCLUSION**

The **Bus Ticket Reservation System** successfully integrates **Java**, **MySQL**, **JDBC**, **Java Swing**, to provide a functional and user-friendly platform for booking bus tickets. The system allows users to check bus availability and make bookings, with all data stored in a MySQL database.

**Key Achievements:**

* Integration of **JDBC** for connecting Java with MySQL.
* Dynamic display of bus details using **JavaSwing**.
* Booking functionality with real-time availability checks.

**Future Improvements:**

* Implement **user authentication** and personalized features.
* Integrate **payment gateway** for online transactions.
* Optimize for **mobile compatibility** and add an **admin panel**.

This project provides a solid foundation and can be enhanced with additional features for better user experience and functionality.

**7.REFERENCES**

**Books:**

1. **"Database System Concepts" by Abraham Silberschatz, Henry F. Korth, and S. Sudarshan**
   * This book provides a solid foundation in database design, including concepts like ER models, normalization, and SQL, which are crucial for your project.
2. **"Head First SQL" by Lynn Beighley**
   * A beginner-friendly guide to mastering SQL for database management, which is vital for handling queries in your project.

**Websites:**

1. **W3Schools - SQL Tutorial**
   * Comprehensive tutorials for understanding and implementing SQL queries, including examples for creating and managing databases.
2. [**GeeksforGeeks - Bus Reservation System Project**](https://www.geeksforgeeks.org/)
   * A useful source for understanding the step-by-step implementation of reservation systems in different programming languages.

**Research Papers:**

1. **"Design and Implementation of Online Bus Ticket Reservation System" (IJERT)**
   * This paper provides insights into the technical and design aspects of implementing a bus reservation system.