RAJALAKSHMI
ENGINEERING COLLEGE
An AUTONOMOUS Institution
Affiliated to ANNA UNIVERSITY, Chennai

BUS RESERVATION SYSTEM BY USING JAVA AND SQL

A PROJECT REPORT
Submitted by

SAI SANJAY SV (231501142)
SAILESH RANGARAJ (231501141)

CS23333 - OBJECT ORIENTED PROGRAMMING USING JAVA

Department of Artificial Intelligence and Machine Learning

Rajalakshmi Engineering College, Thandalam.

BONAFIDE CERTIFICATE

This is to certify that the Mini project work titled “**BUS RESERVATION SYSTEM BY USING JAVA AND SQL** ” done by , SAI SANJAY SV (231501142), SAILESH RANGARAJ (231501141) is a record of Bonafide work carried out by him/her under my supervision as a part of MINI PROJECT for the subject CS23333 - OBJECT ORIENTED PROGRAMMING USING JAVA by Department of Artificial Intelligence and Machine Learning.

SIGNATURE	SIGNATURE
Dr.Sekar K M.E., Ph.D,	Mr. Devendar Rao.
HEAD OF THE DEPARTMENT	FACULTY IN CHARGE
Department of Artificial Intelligence	Department of Artificial
Intelligence and Machine Learning,	and Machine Learning,
Rajalakshmi Engineering College,	Rajalakshmi EngineerinCollege,
Thandalam,	Thandalam,
Chennai-602 105.	Chennai- 602 105.

Submitted for the project viva-voce examination held on____________________.

INTERNAL EXAMINER EXTERNAL **EXAMINER**

TABLE OF CONTENTS:

CHAPTER 1
ABSTRACT

The Bus Reservation System is a web-based application designed to streamline the process of booking and managing bus reservations. This project aims to enhance the user experience by providing a user-friendly interface, efficient database management, and robust functionalities for viewing routes, making reservations, and canceling bookings. By leveraging PHP and MySQL, the system ensures a seamless interaction between the frontend and backend components, facilitating real-time data processing and management. This report details the architecture, implementation, and outcomes of the project, showcasing its potential to improve transportation services.

CHAPTER 2
INTRODUCTION

In today's rapidly advancing world, efficient transportation systems play a critical role in ensuring seamless mobility and connectivity. As urbanization and population growth continue to surge, the demand for reliable, accessible, and user-friendly transportation solutions has become more pressing. One area that has seen significant challenges is bus travel, where passengers often struggle with issues such as limited access to accurate information, difficulty in booking tickets, and a lack of streamlined processes for managing reservations. These challenges can lead to inconvenience for travelers, inefficiencies for bus operators, and missed opportunities for optimizing operations.

To address these issues, the Bus Reservation System (BRS) project has been developed. This system seeks to revolutionize the bus travel experience by providing an intuitive, accessible, and efficient platform for both travelers and bus operators. With the integration of modern web technologies, the Bus Reservation System aims to simplify the reservation process, enhance operational efficiency, and offer a more user-friendly experience for passengers. The system offers a range of features designed to address common pain points, including real-time access to route information, online booking of tickets, seat reservations, and easy management of bookings.

Project Overview:

The Bus Reservation System is a web-based application designed to facilitate the booking and management of bus tickets for passengers and optimize the day-to-day operations of bus operators. The platform serves as an intermediary between travelers and bus operators, providing users with a comprehensive set of tools to view available routes, check schedules, reserve seats, and manage their bookings in real-time. For bus operators, the system offers an efficient backend to manage bus routes, schedules, and passenger reservations, reducing manual work and minimizing errors.

Key Features and Benefits:

1. **Route and Schedule Information**
   One of the primary challenges in traditional bus travel is the lack of easily accessible, up-to-date information about available routes, schedules, and bus stops. The Bus Reservation System provides passengers with a detailed list of bus routes, along with up-to-date schedule information. Users can easily search for and view available routes between different destinations, along with departure and arrival times. This feature not only helps passengers plan their journeys but also improves transparency and trust in the system.

2. **Seat Reservation and Ticket Booking**
   With the system, passengers can reserve their seats in advance, reducing the risk of overcrowding and ensuring a guaranteed spot on the bus. Users can select their preferred bus, choose from available seats, and make payments online to confirm their bookings. This feature eliminates the need for physical ticket counters, reducing long queues and enabling passengers to book tickets from the comfort of their homes or on the go.

3. **User Account Management**
   The platform allows passengers to create and manage user accounts, where they can track their booking history, view upcoming trips, and make changes to their reservations if necessary. This personalized experience enhances convenience and ensures that passengers have full control over their travel plans. Users can also receive notifications regarding booking confirmations, cancellations, and updates related to their trips.

4. **Real-time Updates and Notifications**
   For both passengers and bus operators, real-time updates are crucial. The system

provides users with notifications regarding bus delays, cancellations, or other changes to schedules. This ensures that passengers are always informed of any potential issues before they head to the bus terminal, minimizing frustration and improving the overall experience.

5. **Payment Integration**
   The system integrates secure online payment options, enabling passengers to pay for their tickets using a variety of methods, including credit/debit cards, e-wallets, or other online payment systems. This feature not only streamlines the payment process but also adds a layer of security, ensuring that financial transactions are handled safely and efficiently.

6. **Backend Management for Bus Operators**
   From the perspective of bus operators, the Bus Reservation System offers an easy-to-use backend interface that simplifies route management, ticket sales, and passenger data. Operators can add, update, or delete bus routes, adjust schedules, manage seat availability, and track reservations in real-time. The system also generates reports and analytics to help operators optimize their fleet utilization and overall operations.

7. **Scalability and Flexibility**
   As demand for bus services grows, the system is designed to be scalable, allowing for easy expansion to include more routes, buses, and even additional features. The flexible architecture ensures that the system can be customized to suit the unique needs of different bus operators, whether they are managing a small fleet or a large transportation network.

Technological Aspects:

The Bus Reservation System is built using modern web development technologies to ensure a smooth, responsive, and secure experience for users. Some key technologies utilized in the development of the system include:

- **Frontend Development**:
  The user interface (UI) is designed to be intuitive and responsive, ensuring that it works seamlessly across a range of devices, from desktops to mobile phones. Technologies such as HTML, CSS, JavaScript, and popular front-end frameworks like React or Angular are used to create a dynamic and user-friendly interface.

- **Backend Development**:
  The backend of the system is responsible for processing user requests, managing databases, and handling business logic. Technologies such as Node.js, Python, or PHP can be used for server-side development, depending on the specific requirements. A relational database (such as MySQL or PostgreSQL) or a NoSQL database (such as MongoDB) can be used to store and manage user data, booking information, routes, and schedules.

- **Payment Gateway Integration**:
  A secure payment gateway, such as PayPal, Stripe, or Razorpay, is integrated to handle online transactions, providing passengers with a safe and reliable way to make payments.

- **Real-time Updates**:
  Real-time features, such as bus tracking, schedule updates, and notifications, are enabled using technologies like WebSockets or Push Notifications. This ensures that both passengers and operators have access to the latest information at all times.

- **Security**:
  Security is a top priority in the development of the Bus Reservation System. The platform uses HTTPS for secure communication, encryption methods for user data protection, and authentication protocols to prevent unauthorized access to sensitive information.

Purpose and Objectives:

The purpose of this project is to enhance the travel experience for passengers while also improving operational efficiency for bus operators. Specifically, the objectives of the Bus Reservation System include:

1. **Improving Accessibility**: Providing an easy-to-use platform where passengers can quickly access route information, check schedules, and make bookings without needing to visit a physical ticket counter.
2. **Streamlining the Booking Process**: Simplifying the reservation process by allowing passengers to book tickets, choose seats, and make payments online.
3. **Reducing Operational Costs**: Helping bus operators reduce the need for manual ticketing, thereby reducing operational costs, minimizing errors, and ensuring more efficient resource management.
4. **Enhancing Customer Satisfaction**: Offering real-time updates and personalized experiences to improve customer satisfaction and loyalty.
5. **Promoting Scalability**: Designing a system that can easily adapt to the growing demands of both passengers and bus operators.

Conclusion:

The Bus Reservation System is an innovative solution designed to address the challenges faced by travelers and bus operators in the transportation industry. By combining modern web technologies with user-centered design, this system provides a seamless, efficient, and reliable platform for booking and managing bus travel. The system not only enhances convenience for passengers but also streamlines backend operations for bus operators, ultimately improving the overall efficiency of bus services. With its user-friendly interface, real-time updates, and secure payment integration, the Bus Reservation System represents a significant step forward in the modernization of public transportation.

CHAPTER 3
RELATED WORK

Limitations of Existing Bus Reservation Systems:

While there are numerous bus reservation systems currently available in the market, many of these systems face significant challenges that undermine their effectiveness and user satisfaction. These issues often arise due to outdated technology, inefficient design, and inadequate customer service features. Common limitations include:

1. **Complex User Interfaces (UI)**
   Many existing reservation systems have overly complicated or non-intuitive interfaces that make it difficult for users to navigate. This complexity can lead to user frustration, especially for individuals who may not be tech-savvy. A convoluted interface can hinder the process of booking tickets, viewing schedules, or making changes to reservations. As a result, users may abandon the system or prefer to use alternative methods, such as booking tickets through physical counters.
2. **Lack of Real-time Updates**
   Real-time information is critical for effective bus travel planning. However, numerous bus reservation systems fail to provide live updates regarding bus schedules, delays, or cancellations. This gap in functionality can lead to confusion for passengers, especially when changes occur at the last minute. Without real-time tracking, passengers may arrive at the bus terminal only to find that their bus has been delayed or canceled, resulting in frustration and wasted time.
3. **Inadequate Customer Support Features**
   Many existing systems lack robust customer support functionalities, which are essential for resolving issues that users encounter during the booking process. Whether it's troubleshooting payment problems, addressing concerns regarding reservations, or handling cancellations, effective customer support is a key factor in ensuring user satisfaction. Without immediate assistance, users may face challenges in resolving issues in a timely manner, leading to dissatisfaction with the service.
4. **Limited Feedback Mechanisms**
   Another limitation often observed in existing systems is the absence of effective feedback mechanisms. User feedback is an essential tool for improving any system, as it provides direct insights into user needs, preferences, and pain

points. Many bus reservation systems lack features that allow users to easily rate their experience, report problems, or suggest improvements. This lack of feedback can result in a static system that does not evolve or improve over time based on user needs.

Importance of User-Centered Design:

User-centered design (UCD) is a design approach that prioritizes the needs, preferences, and behaviors of users throughout the development process. Numerous studies emphasize the importance of UCD in the creation of reservation systems, as it directly contributes to higher levels of customer satisfaction and usability. UCD focuses on the following key principles:

1. **Usability and Simplicity**
   The core goal of user-centered design is to ensure that the system is easy to use and accessible to a wide range of users. Research indicates that systems that are designed with user needs in mind significantly improve the likelihood of adoption and continued use. Features such as simple navigation, clear instructions, and easily accessible options contribute to a positive user experience, making it more likely that users will choose to interact with the system over other alternatives.
2. **Personalization and Flexibility**
   UCD also emphasizes creating systems that are flexible and customizable to meet individual user preferences. By offering personalized experiences, such as storing frequent routes, offering tailored recommendations, or providing options for seat preferences, systems can engage users on a deeper level. Personalization also enhances user loyalty and satisfaction, as it makes the system feel more relevant and intuitive to the individual.
3. **Incorporation of User Feedback**
   The continuous incorporation of user feedback is a cornerstone of user-centered design. Research suggests that systems which integrate user input—whether through surveys, direct feedback forms, or user testing—are far more likely to be successful in the long term. This approach allows developers to identify pain points early in the design process and make iterative improvements. Additionally, when users feel that their feedback is valued, they are more likely to have a positive perception of the system, leading to increased trust and engagement.
4. **Improved Customer Satisfaction**
   Studies consistently show that systems developed with a focus on user experience (UX) result in higher levels of customer satisfaction. This is

particularly relevant in the context of bus reservation systems, where ease of use, timely information, and customer support play critical roles in shaping the passenger's overall experience. Systems that prioritize these aspects create a more pleasant and seamless experience for users, increasing the likelihood of repeat usage and positive word-of-mouth.

Our Approach: Building on User-Centered Design Principles:

Our Bus Reservation System (BRS) project takes into account the limitations of existing systems and aims to address these shortcomings by building a platform that prioritizes **usability, responsiveness**, and **customer-centric features**. In doing so, we leverage the insights gained from studies on user-centered design and aim to build a system that not only meets but exceeds the expectations of modern travelers. Below are the key aspects that differentiate our approach:

1. **Intuitive User Interface**
   One of the primary goals of our project is to ensure that the system has an intuitive, user-friendly interface. We have designed the platform with simplicity in mind, ensuring that users can easily navigate through the different stages of booking, viewing routes, and managing their reservations. The design follows a logical flow, with clear instructions and minimal clutter. Features like a search bar for route selection, clear buttons for booking, and visible confirmation steps are implemented to minimize confusion and enhance usability.
2. **Real-Time Updates and Notifications**
   Unlike many existing systems, our platform integrates real-time updates on bus schedules, delays, cancellations, and seat availability. We use technologies such as WebSockets and push notifications to deliver instant updates to users. This ensures that passengers are kept informed at all times, minimizing the chances of arriving at a terminal only to find unexpected changes in their travel plans. Real-time bus tracking features further enhance this experience by allowing passengers to track their bus in real-time.
3. **Robust Customer Support Features**
   Understanding the critical role that customer support plays in enhancing user satisfaction, we have implemented a robust support system within our platform. This includes options for users to contact support agents through live chat, email, or a helpdesk portal. Furthermore, users can quickly find answers to frequently asked questions (FAQs), troubleshoot common issues, and resolve problems such as payment issues or incorrect reservations without delays.
4. **User Feedback Mechanisms**

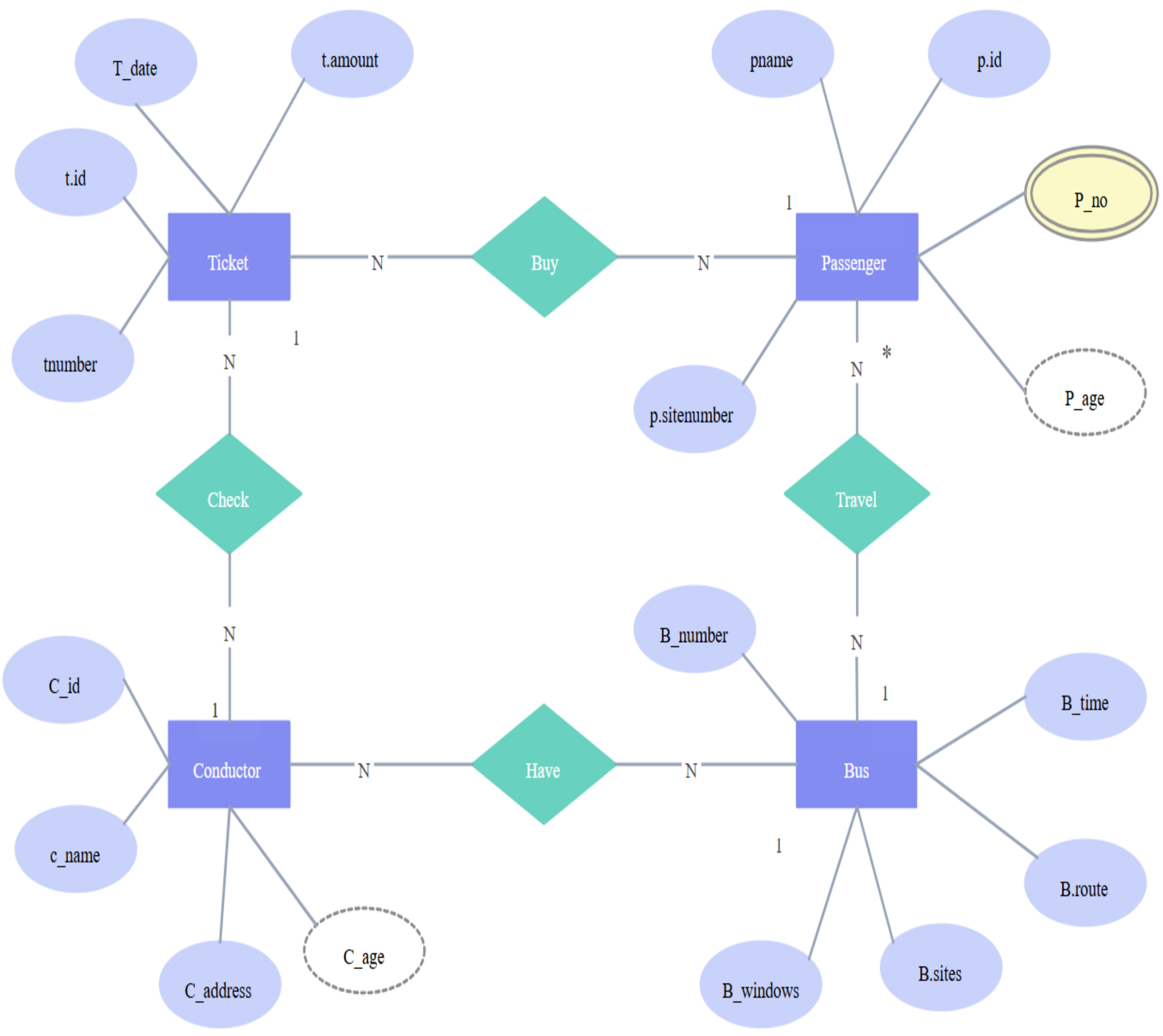
To ensure continuous improvement, we have integrated feedback mechanisms throughout the system. After each booking, users are prompted to rate their experience, report issues, and provide suggestions for improvement. This feedback is invaluable for refining the system over time and ensuring that it meets the evolving needs of passengers. Moreover, users who provide feedback are often incentivized with discounts or loyalty points, making the feedback process more engaging.

5. **Secure and Scalable Database Structure**
   We have implemented a robust backend infrastructure that supports the system's dynamic operations, ensuring that data integrity and security are never compromised. The database is designed to handle large volumes of data, including user profiles, booking histories, routes, schedules, and payment transactions. The system uses encryption and secure authentication protocols to ensure that sensitive information, such as personal and payment details, is protected at all times.
6. **Personalization Options**
   Our platform offers personalized features such as saving frequent routes, remembering seat preferences, and recommending buses based on the user's past travel history. This personalization helps make the experience smoother and more tailored to individual needs, contributing to greater user satisfaction and engagement.

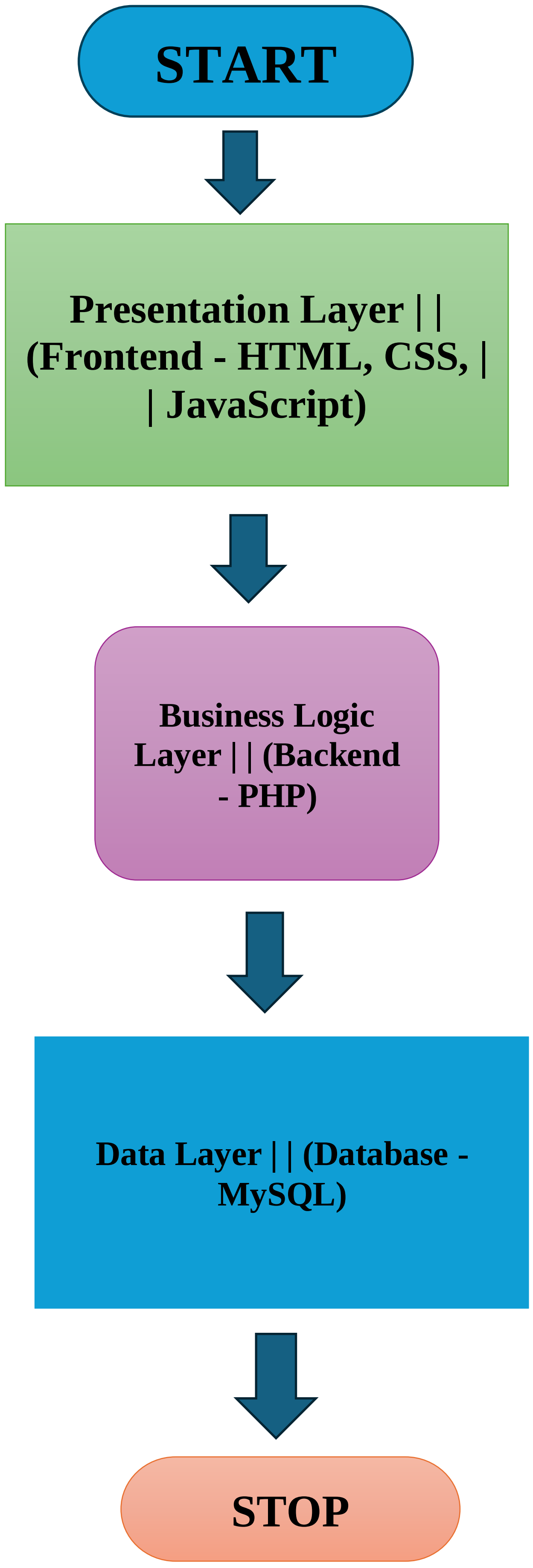
Summary:

In summary, while many existing bus reservation systems struggle with limitations like complex interfaces, lack of real-time updates, and poor customer support, our project seeks to overcome these challenges by adopting a **user-centered design** approach. By focusing on **usability, responsiveness, real-time updates**, and **robust customer support**, we aim to create a bus reservation system that offers a superior user experience. Furthermore, by implementing a secure and scalable database architecture, our system ensures that passenger data remains protected while providing the flexibility to grow with future demands. Ultimately, the goal is to deliver a system that not only simplifies the bus booking process but also enhances overall customer satisfaction, setting a new standard for modern transportation services.

CHAPTER 4
ER DIAGRAM



MODEL ARCHITECTURE:


START
Presentation Layer | | (Frontend - HTML, CSS, | | JavaScript)
Business Logic Layer | | (Backend - PHP)
Data Layer | | (Database - MySQL)
STOP

Flowchart Description:

1. Presentation Layer (Frontend):

- The user interacts with the Presentation Layer (the Frontend), which is the part of the system that they can see and use directly. It is responsible for:
- Displaying available routes, bus schedules, and seat availability.
- Allowing users to make reservations, cancel bookings, and view their booking history.
- Communicating with the Business Logic Layer for user inputs (e.g., booking a ticket, entering payment details).

2. Business Logic Layer (Backend):

- Once a user interacts with the Presentation Layer, the Business Logic Layer processes the user request. This layer is where most of the application logic happens:
- It handles the user requests like searching for routes, making reservations, and processing cancellations.
- It processes data and calls the Data Layer (Database) to retrieve, insert, or update information based on user inputs.
- The Business Logic Layer uses PHP scripts to perform the necessary operations.

3. Data Layer (Database):

- The Data Layer (Database) stores all the critical data needed for the system, including:
- User details, routes, bus schedules, seat availability, and reservation information.
- It is managed by a MySQL database, which is queried by the Business Logic Layer to store and retrieve data securely

Key Flow Steps:

1. User Interaction:

- The user interacts with the Presentation Layer, such as by viewing routes, searching for buses, or selecting a seat.

2. Business Logic Processing:

- The Business Logic Layer processes the user's actions, such as querying for available seats or making a reservation.

3. Database Query/Interaction:

1) The Business Logic Layer sends requests to the Data Layer (MySQL database) to store or retrieve data. For instance:

   a) Searching available routes, buses, or seats.

   b) Storing new reservations, updating seat availability.

   c) Retrieving or canceling existing bookings.

4. Response Back to User:

- After the Data Layer processes the request (e.g., a new reservation is stored or a canceled booking is updated), the Business Logic Layer processes the response and sends the appropriate data or confirmation back to the Presentation Layer.
- The Presentation Layer then updates the user interface, showing the results to the user, such as displaying confirmed bookings, updated seat availability, or any errors.

CHAPTER 5
IMPLEMENTATION

The implementation of the Bus Reservation System involves a carefully designed architecture that integrates a robust database, a user-friendly frontend, and a functional backend to provide an intuitive booking experience for passengers and seamless operation management for bus operators. Below is a detailed breakdown of the system's implementation, covering each key component.

1. Database Design:

The foundation of the Bus Reservation System is the MySQL database, which stores and manages all the critical data related to buses, routes, and reservations. The database schema consists of three main tables: Buses, Routes, and Reservations. Each table has specific columns to store relevant information and relationships between entities.

Buses Table:

This table stores information about the buses available in the fleet.

- **Columns:**
  - **BusID**: Unique identifier for each bus.
  - **BusNumber**: The registration number or bus identifier.
  - **BusType**: Type of the bus (e.g., Standard, Luxury).
  - **TotalSeats**: Total seating capacity of the bus.
  - **AvailableSeats**: Number of available seats for booking on the bus.
- **Purpose**: This table tracks details about each bus in the system, including the number of seats and the bus type. The available seats column is updated as passengers book or cancel reservations.

Routes Table:

This table contains information about the routes operated by the buses.

- **Columns:**
  - **RouteID**: Unique identifier for each route.
  - **StartLocation**: The starting location of the route.
  - **EndLocation**: The destination of the route.
  - **DepartureTime**: Scheduled departure time of the bus.
  - **ArrivalTime**: Scheduled arrival time at the destination.
  - **BusID**: A foreign key referencing the **Buses** table to indicate which bus is operating the route.
- **Purpose**: This table holds details about the routes available in the system, linking each route to a

specific bus and providing information such as departure and arrival times.

Reservations Table:

This table stores data related to passenger reservations, allowing the system to track bookings.

- **Columns:**
    - **ReservationID**: Unique identifier for each reservation.
    - **PassengerName**: Name of the passenger making the booking.
    - **PassengerContact**: Contact details (phone number or email) of the passenger.
    - **RouteID**: A foreign key referencing the **Routes** table to indicate which route the reservation is associated with.
    - **SeatsBooked**: Number of seats booked by the passenger.
    - **ReservationDate**: The date when the reservation was made.
- **Purpose**: This table keeps track of all reservations made by passengers, storing their details and the corresponding route and seat information.

ER Diagram for Database Schema:

A simple **Entity Relationship Diagram (ERD)** for the database schema would look like this:

- **Buses Table** → Connected to **Routes Table** by BusID (1-to-many relationship).
- **Routes Table** → Connected to **Reservations Table** by RouteID (1-to-many relationship).

2. Frontend Development:

The frontend of the Bus Reservation System is designed with a focus on simplicity and usability. The goal is to ensure that users can easily interact with the system without confusion. The frontend is built using HTML, CSS, and JavaScript.

Key Components of Frontend:

1. **Home Page (index.html)**
    - This page is the starting point for users and provides the following options:
        - View Available Routes: A list of available bus routes with departure and arrival information.
        - Make a Reservation: A form where users can select a route, choose the number of seats, and enter their details for booking.
        - Cancel Booking: A feature for users to cancel their existing bookings by providing reservation details.
    - The home page acts as a dashboard for users to easily navigate to other parts of the system.
2. **CSS Styling (styles.css)**
    - The frontend has a visually appealing layout, with well-structured components that enhance the user experience. **CSS** ensures that the pages are responsive and easy to navigate, regardless of

the device used (desktop, tablet, or mobile). For instance, the layout includes:

  - Clean, readable fonts.
  - Organized sections for booking routes, selecting seats, and managing reservations.
  - Clear buttons and call-to-action elements to guide the user.

3. **Interactive Elements with JavaScript**
   - **JavaScript** is used to handle dynamic user interactions on the frontend. For example:
     - When a user selects a route, JavaScript dynamically updates the available seat count.
     - It also validates user inputs on the reservation form to ensure all required fields are filled correctly before submitting.

3. Backend Development:

The backend of the system is developed using PHP, which handles user requests, processes interactions with the database, and manages the application logic.

Key Backend Components:

1. **Database Connection (db.php):**

- This script establishes a connection to the **MySQL** database using PHP's **PDO** (PHP Data Objects) extension. It ensures that all database operations (e.g., fetching routes, updating bookings) are performed securely and efficiently.
- Example of a database connection in PHP:

Php:

```
try {
    $pdo = new PDO("mysql:host=localhost;dbname=bus_reservation_system", "username", "password");
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
} catch (PDOException $e) {
    die("Could not connect to the database: " . $e->getMessage());
}
```

2. **View Routes (view_routes.php)**

- This PHP script queries the **Routes** and **Buses** tables to fetch available routes and their corresponding bus details (e.g., bus type, availability). The results are then displayed on the frontend for the user to view.
- Example of viewing available routes:

Php:

```
$stmt = $pdo->query("SELECT R.RouteID, R.StartLocation, R.EndLocation, R.DepartureTime,
                     R.ArrivalTime, B.BusNumber, B.BusType
                     FROM Routes R JOIN Buses B ON R.BusID = B.BusID");
$routes = $stmt->fetchAll(PDO::FETCH_ASSOC);
```

3. **Reserve Seats (reserve.php):**
   - When a user selects a route and books a seat, this PHP script processes the reservation. It first checks the available seats on the selected bus, updates the **Buses** table to reduce the available seats, and then inserts the reservation details into the **Reservations** table.
   - Example of booking a ticket:

Php:

```
$pdo->beginTransaction();
// Check available seats
$stmt = $pdo->prepare("SELECT AvailableSeats FROM Buses WHERE BusID = ?");
$stmt->execute([$busID]);
$availableSeats = $stmt->fetchColumn();
if ($availableSeats >= $seatsRequested) {
    // Update available seats
    $stmt = $pdo->prepare("UPDATE Buses SET AvailableSeats = AvailableSeats - ? WHERE BusID = ?");
    $stmt->execute([$seatsRequested, $busID]);
    // Insert reservation
    $stmt = $pdo->prepare("INSERT INTO Reservations (PassengerName, PassengerContact, RouteID, SeatsBooked, ReservationDate)
                           VALUES (?, ?, ?, ?, CURDATE())");
    $stmt->execute([$passengerName, $passengerContact, $routeID, $seatsRequested]);

    $pdo->commit();
} else {
    // Handle not enough seats
    echo "Not enough available seats.";
}
```

4. **Cancel Reservation (cancel_reservation.php)**
   - This script allows users to cancel their reservations. It checks the reservation in the **Reservations** table and updates the **Buses** table to increase the available seats.
   - Example of canceling a reservation:

Php:

```
$stmt = $pdo->prepare("SELECT SeatsBooked, RouteID FROM Reservations WHERE ReservationID
= ?");
$stmt->execute([$reservationID]);
$reservation = $stmt->fetch(PDO::FETCH_ASSOC);

if ($reservation) {
   // Update available seats
   $stmt = $pdo->prepare("UPDATE Buses SET AvailableSeats = AvailableSeats + ? WHERE BusID
= (SELECT BusID FROM Routes WHERE RouteID = ?)");
   $stmt->execute([$reservation['SeatsBooked'], $reservation['RouteID']]);

   // Delete reservation
   $stmt = $pdo->prepare("DELETE FROM Reservations WHERE ReservationID = ?");
   $stmt->execute([$reservationID]);
} else {
   echo "Reservation not found.";
}
```

4. Sample Queries:

Here are some sample SQL queries used in the project:

To View All Buses:

```
SELECT * FROM Buses;
```

To View All Routes Along with Bus Details:

```
SELECT R.RouteID, R.StartLocation, R.EndLocation, R.DepartureTime,
       R.ArrivalTime, B.BusNumber, B.BusType
FROM Routes R
JOIN Buses B ON R.BusID = B.BusID;
```

To Book a Ticket:

```
START TRANSACTION;
SELECT AvailableSeats FROM Buses WHERE BusID = ?;
UPDATE Buses SET AvailableSeats = AvailableSeats - ? WHERE BusID = ?;
INSERT INTO Reservations (PassengerName, PassengerContact, Route)
```

CHAPTER 6
RESULTS AND DISCUSSIONS

Upon implementation of the Bus Reservation System:

1. **User-Friendly Interface**: Easy navigation for all users.
2. **Real-Time Seat Availability**: Instant updates for booking and cancellations.
3. **Multiple Payment Options**: Supports cards, wallets, and net banking.
4. **Booking History**: Access past bookings and trip details.
5. **Notifications**: Real-time updates via email/SMS for bookings and changes.
6. **Mobile & Desktop Compatible**: Fully responsive across devices.
7. **Advanced Search Filters**: Quickly find routes based on time, price, or availability.
8. **Flexible Cancellation & Refunds**: Clear policies based on ticket type.
9. **Admin Dashboard**: Easy route and seat management for admins.
10. **Enhanced Security**: Secure payment and data protection.
11. **24/7 Customer Support**: Accessible help via chat, phone, and FAQs.
12. **Group Booking**: Reserve multiple seats with ease.
13. **Feedback System**: Users can rate and review their experience.
14. **Seasonal Offers**: Special discounts and loyalty rewards.

CHAPTER 7
CONCLUSION AND FUTURE WORKS

Conclusion:

The Bus Reservation System effectively modernizes the traditional ticket booking process by offering a streamlined, user-friendly interface that simplifies reservations, payments, and ticket management. Its integration of advanced web technologies and robust database management ensures both efficiency and reliability, improving the overall user experience while addressing key challenges such as booking errors and inefficiencies.

Future Uses:

Future enhancements could include mobile application integration for on-the-go bookings, real-time tracking of buses, and personalized notifications. Additionally, advanced analytics for bus operators could optimize route planning, occupancy forecasting, and dynamic pricing, further improving service quality and operational efficiency.

CHAPTER 8

APPENDIX

CODE:

1. CREATING A DATABASE USING SQL:

Step 1: Create the Database

```
CREATE DATABASE BusReservationSystem;
USE BusReservationSystem;
```

Step 2: Create Tables

1. Buses Table:

```
CREATE TABLE Buses (
BusID INT PRIMARY KEY AUTO_INCREMENT,
BusNumber VARCHAR(20) NOT NULL,
BusType VARCHAR(20),
TotalSeats INT,
AvailableSeats INT
);
```

2. Routes Table:

```
CREATE TABLE Routes (
RouteID INT PRIMARY KEY AUTO_INCREMENT,
StartLocation VARCHAR(50) NOT NULL,
EndLocation VARCHAR(50) NOT NULL,
DepartureTime TIME,
ArrivalTime TIME,
BusID INT,
FOREIGN KEY (BusID) REFERENCES Buses(BusID)
);
```

3. Reservations Table:

```
CREATE TABLE Reservations (
ReservationID INT PRIMARY KEY AUTO_INCREMENT,
PassengerName VARCHAR(50) NOT NULL,
PassengerContact VARCHAR(15),
RouteID INT,
SeatsBooked INT,
ReservationDate DATE,
FOREIGN KEY (RouteID) REFERENCES Routes(RouteID)
);
```

Step 3: Insert Sample Data

1. Insert Buses:

```
INSERT INTO Buses (BusNumber, BusType, TotalSeats, AvailableSeats)
VALUES (&#39;BUS123&#39;, &#39;AC&#39;, 40, 40),
(&#39;BUS456&#39;, &#39;Non-AC&#39;, 50, 50);
```

2. Insert Routes:

```
INSERT INTO Routes (StartLocation, EndLocation, DepartureTime, ArrivalTime,
BusID)
VALUES (&#39;City A&#39;, &#39;City B&#39;, &#39;08:00:00&#39;, &#39;12:00:00&#39;, 1),
(&#39;City C&#39;, &#39;City D&#39;, &#39;14:00:00&#39;, &#39;18:00:00&#39;, 2);
```

Step 4: Reservation Queries

1. Book a Ticket: Reduce AvailableSeats in Buses table and add a new reservation record.

```
START TRANSACTION;
-- Check seat availability
SELECT AvailableSeats FROM Buses WHERE BusID = 1;
-- Assuming 3 seats are being booked
UPDATE Buses SET AvailableSeats = AvailableSeats - 3 WHERE BusID = 1;
-- Add reservation record
INSERT INTO Reservations (PassengerName, PassengerContact, RouteID,
SeatsBooked, ReservationDate)
VALUES (&#39;John Doe&#39;, &#39;1234567890&#39;, 1, 3, CURDATE());
COMMIT;
```

2. View All Reservations:

```
SELECT R.ReservationID, R.PassengerName, R.SeatsBooked, R.ReservationDate,
Ro.StartLocation, Ro.EndLocation
FROM Reservations R
JOIN Routes Ro ON R.RouteID = Ro.RouteID;
```

3. Cancel a Reservation: Free up seats in Buses and delete the reservation record.

```
START TRANSACTION;
```

```
-- Find the number of seats booked and BusID from RouteID
SELECT SeatsBooked, RouteID INTO @SeatsBooked, @RouteID
FROM Reservations
WHERE ReservationID = 1;
-- Find the BusID associated with the Route

SELECT BusID INTO @BusID
FROM Routes
WHERE RouteID = @RouteID;
-- Update the AvailableSeats in Buses table
UPDATE Buses SET AvailableSeats = AvailableSeats + @SeatsBooked WHERE BusID
= @BusID;
-- Delete the reservation
DELETE FROM Reservations WHERE ReservationID = 1;
COMMIT;
```

Step 5: Sample Queries for Bus and Route Info

1. List All Buses:

```
SELECT * FROM Buses;
```

2. List All Routes:

```
SELECT R.RouteID, R.StartLocation, R.EndLocation, R.DepartureTime,
R.ArrivalTime, B.BusNumber, B.BusType
FROM Routes R
JOIN Buses B ON R.BusID = B.BusID;
```

2. DATABASE CONNECTION (DB.PHP):

Create a db.php file to connect PHP to the MySQL database:

```
php
<?php
$servername = "localhost";
$username = "root";
$password = "";
$database = "BusReservationSystem";

$conn = new mysqli($servername, $username, $password, $database);

if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
?>
```

3. Frontend HTML and CSS:

index.html - Home Page to View and Book Routes:

```
html
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Bus Reservation System</title>
   <link rel="stylesheet" href="styles.css">
</head>
<body>
   <h1>Welcome to the Bus Reservation System</h1>

   <!-- View Routes Section -->
   <section>
      <h2>Available Routes</h2>
      <form action="view_routes.php" method="GET">
         <button type="submit">View Routes</button>
      </form>
   </section>

   <!-- Reserve a Seat Section -->
   <section>
      <h2>Make a Reservation</h2>
      <form action="reserve.php" method="POST">
         <label for="route_id">Route ID:</label>
         <input type="number" id="route_id" name="route_id" required>

         <label for="passenger_name">Name:</label>
         <input type="text" id="passenger_name" name="passenger_name" required>

         <label for="contact">Contact:</label>
         <input type="text" id="contact" name="contact" required>

         <label for="seats">Seats to Book:</label>
         <input type="number" id="seats" name="seats" required>

         <button type="submit">Reserve</button>
      </form>
   </section>

   <!-- Cancel Reservation Section -->
   <section>
      <h2>Cancel Reservation</h2>
      <form action="cancel_reservation.php" method="POST">
         <label for="reservation_id">Reservation ID:</label>
         <input type="number" id="reservation_id" name="reservation_id" required>

         <button type="submit">Cancel Reservation</button>
      </form>
   </section>
```

```
</body>
</html>
```
styles.css - Basic Styling

```
css
Copy code
body {
   font-family: Arial, sans-serif;
   text-align: center;
}

section {
   margin: 20px;
}

input, button {
   padding: 8px;
   margin: 10px 0;
}
```

4. Backend PHP Scripts:

view_routes.php - Display Available Routes:

```
php
Copy code
<?php
include 'db.php';

$sql = "SELECT * FROM Routes";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
   echo "<h2>Available Routes</h2>";
   while ($row = $result->fetch_assoc()) {
      echo "Route ID: " . $row["RouteID"] . " - From " . $row["StartLocation"] . " to " . $row["EndLocation"]
.
         " - Departure: " . $row["DepartureTime"] . " - Arrival: " . $row["ArrivalTime"] . "<br>";
   }
} else {
   echo "No routes available.";
}

$conn->close();
?>
<a href="index.html">Back to Home</a>
```
reserve.php - Reserve a Seat

```
php
Copy code
<?php
include 'db.php';

$route_id = $_POST['route_id'];
$passenger_name = $_POST['passenger_name'];
$contact = $_POST['contact'];
$seats = $_POST['seats'];
```

```
// Check seat availability
$seat_check = $conn->query("SELECT AvailableSeats FROM Buses WHERE BusID = (SELECT BusID FROM Routes WHERE RouteID = $route_id)");
$available_seats = $seat_check->fetch_assoc()['AvailableSeats'];

if ($available_seats >= $seats) {
    // Reduce seat count in Buses table
    $conn->query("UPDATE Buses SET AvailableSeats = AvailableSeats - $seats WHERE BusID = (SELECT BusID FROM Routes WHERE RouteID = $route_id)");

    // Add reservation record
    $sql = "INSERT INTO Reservations (PassengerName, PassengerContact, RouteID, SeatsBooked, ReservationDate)
            VALUES ('$passenger_name', '$contact', $route_id, $seats, CURDATE())";
    if ($conn->query($sql) === TRUE) {
        echo "Reservation successful!";
    } else {
        echo "Error: " . $conn->error;
    }
} else {
    echo "Not enough available seats.";
}

$conn->close();
?>
<a href="index.html">Back to Home</a>
```

cancel_reservation.php - Cancel a Reservation

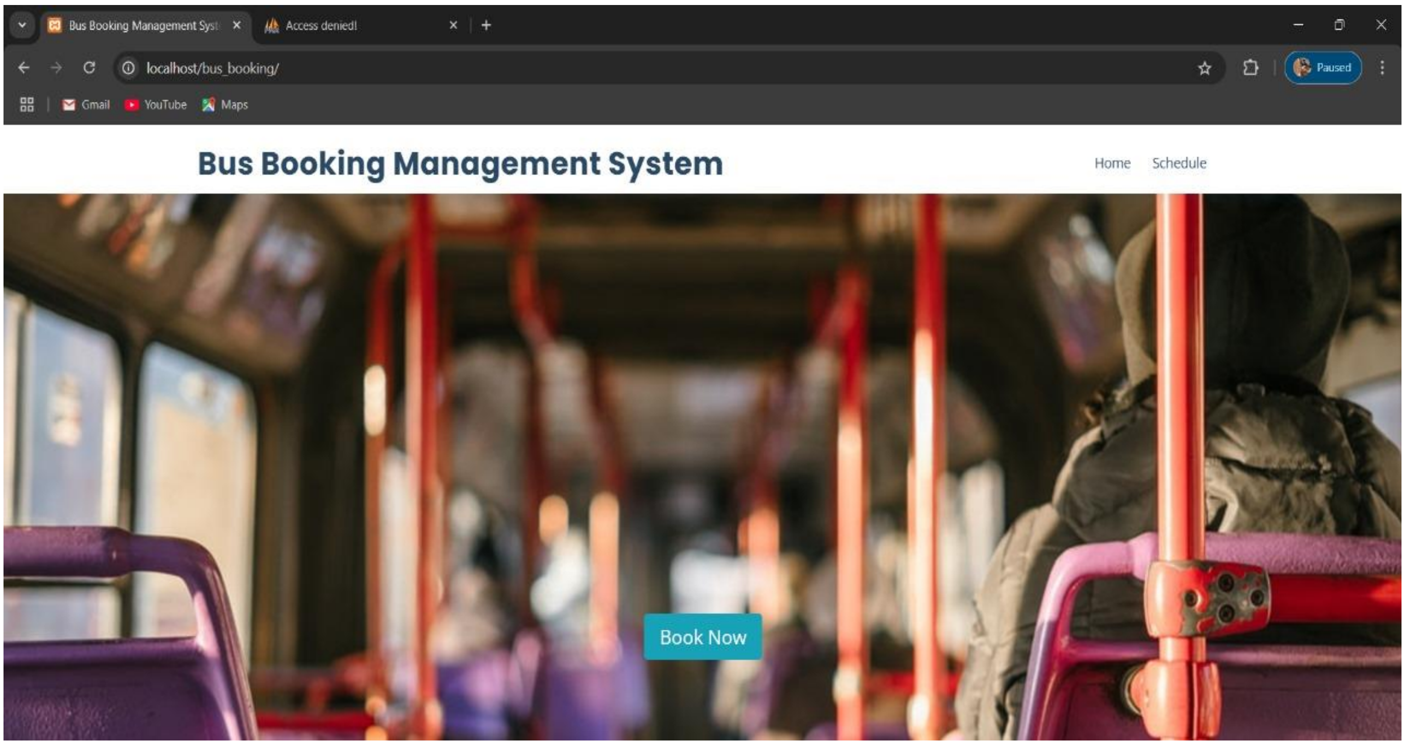
php
Copy code

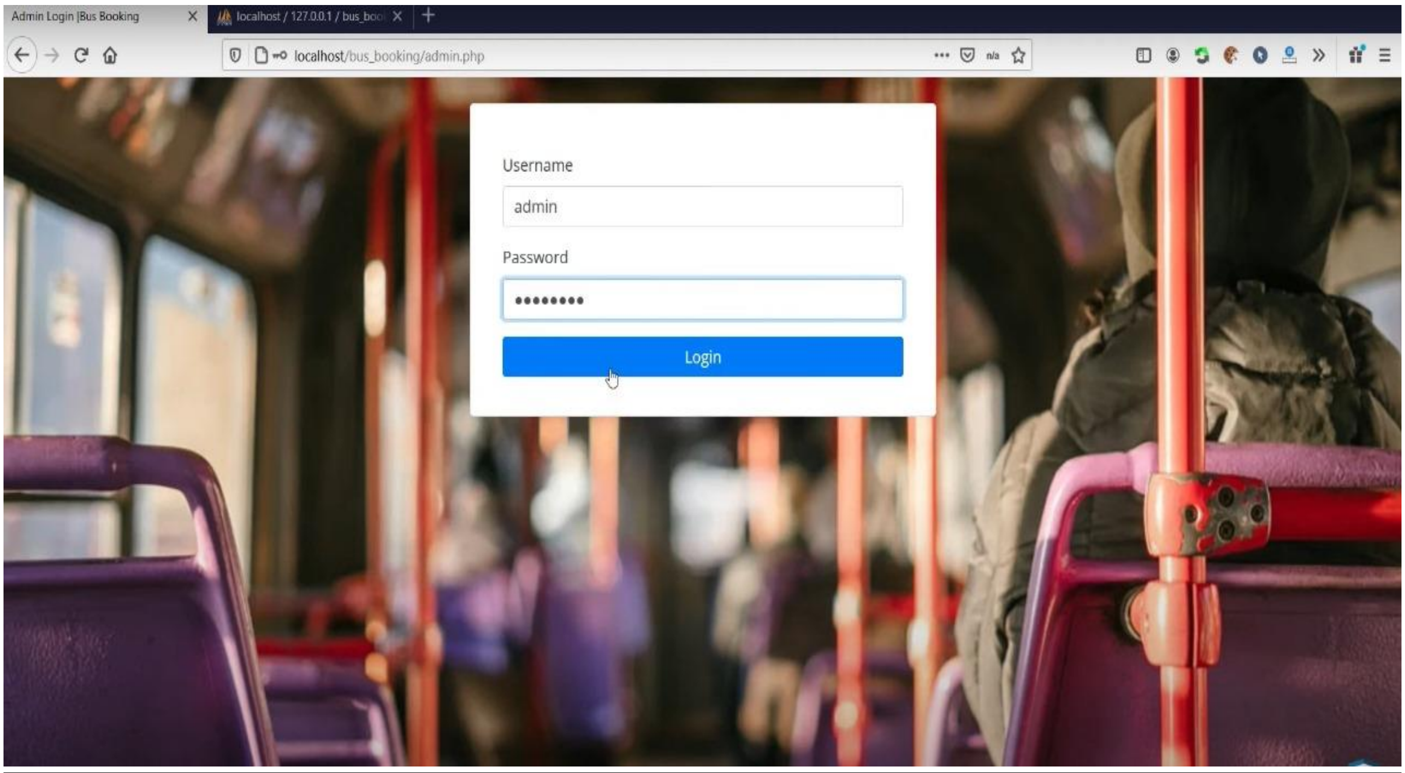
```
<?php
include 'db.php';

$reservation_id = $_POST['reservation_id'];

// Fetch route and seat details
$reservation = $conn->query("SELECT SeatsBooked, RouteID FROM Reservations WHERE ReservationID = $reservation_id");
$reservation_data = $reservation->fetch_assoc();

if ($reservation_data) {
    $seats_to_free = $reservation_data['SeatsBooked'];
    $route_id = $reservation_data['RouteID'];

    // Free seats in Buses table
    $conn->query("UPDATE Buses SET AvailableSeats = AvailableSeats + $seats_to_free WHERE BusID = (SELECT BusID FROM Routes WHERE RouteID = $route_id)");

    // Delete reservation record
    $conn->query("DELETE FROM Reservations WHERE ReservationID = $reservation_id");

    echo "Reservation canceled successfully.";
} else {
    echo "Reservation ID not found.";
}

$conn->close();
?>
<a href="index.html">Back to Home</a>
```

Summary:

1. **index.html** - Main front end with forms to view routes, book, and cancel reservations.
2. **db.php** - Connects to the MySQL database.
3. **view_routes.php** - Lists all routes.
4. **reserve.php** - Handles seat booking and reservation.
5. **cancel_reservation.php** - Allows for reservation cancellation.

OUTPUT:



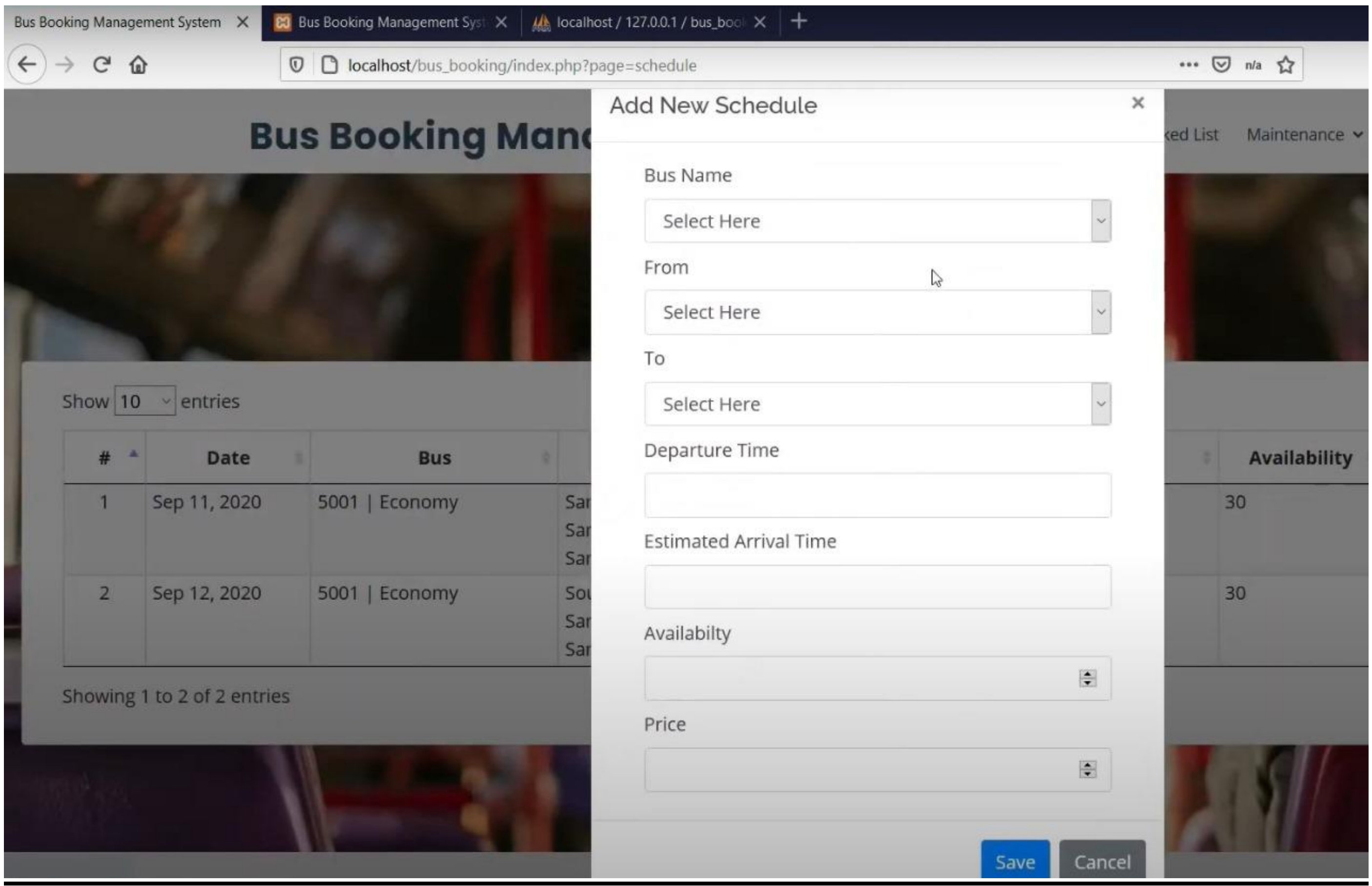

Admin Login |Bus Booking
localhost / 127.0.0.1 / bus_boo
localhost/bus_booking/admin.php
Login

Username

admin

Password

••••••••


Bus Booking Management System | Bus Booking Management Sys | localhost / 127.0.0.1 / bus_boo

localhost/bus_booking/index.php?page=schedule

Bus Booking Manc | ked List | Maintenance

Add New Schedule

Bus Name: Select Here

From: Select Here

To: Select Here

Departure Time:

Estimated Arrival Time:

Availabilty:

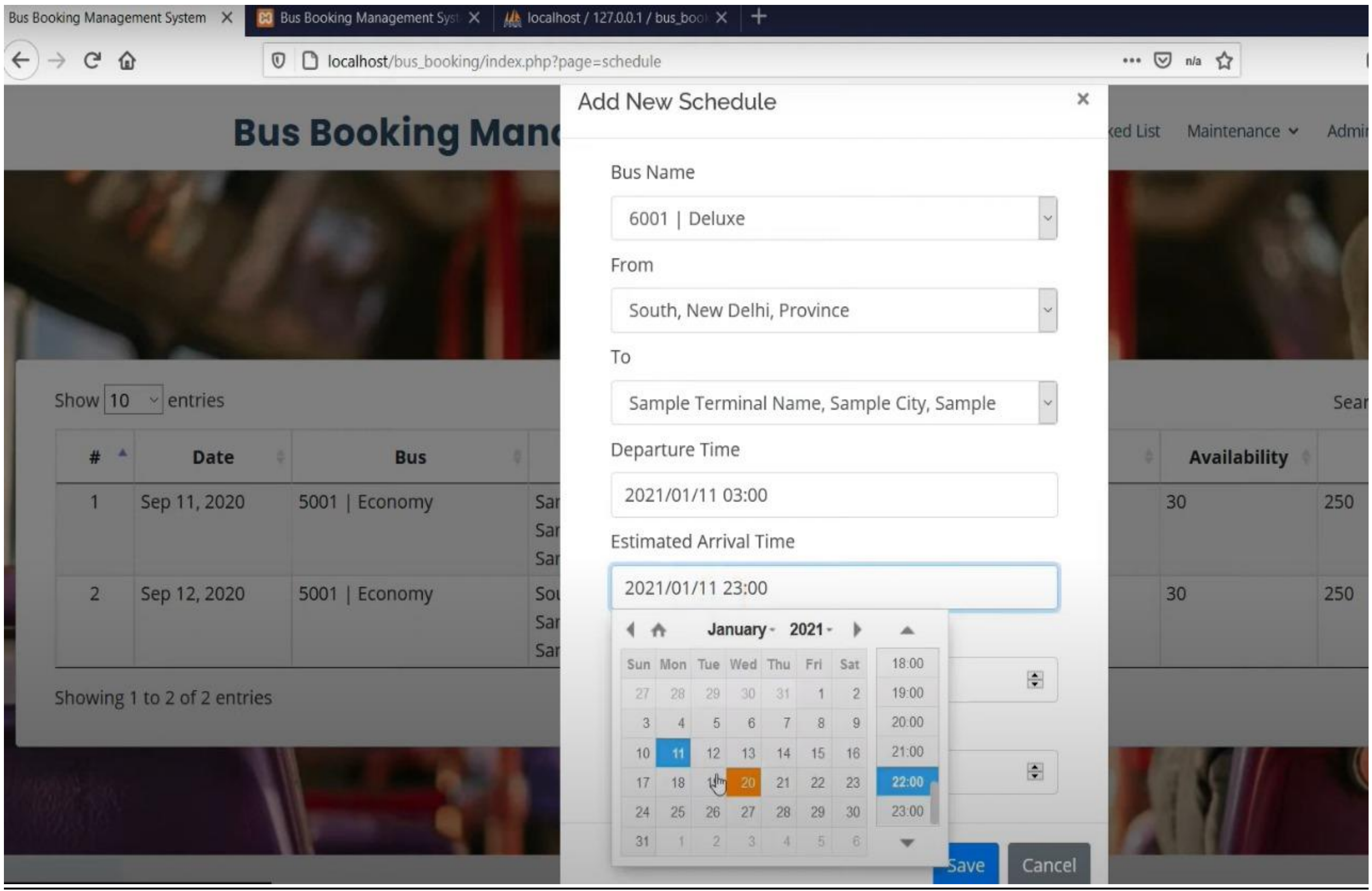
Price:

Save | Cancel

Show 10 entries

#	Date	Bus		Availability
1	Sep 11, 2020	5001 | Economy	Sar Sar Sar	30
2	Sep 12, 2020	5001 | Economy	Sou Sar Sar	30

Showing 1 to 2 of 2 entries


Bus Booking Management System | Bus Booking Management Sys | localhost / 127.0.0.1 / bus_boo

localhost/bus_booking/index.php?page=schedule

Bus Booking Manc | ked List | Maintenance | Admi

Add New Schedule

Bus Name: 6001 | Deluxe

From: South, New Delhi, Province

To: Sample Terminal Name, Sample City, Sample

Departure Time: 2021/01/11 03:00

Estimated Arrival Time: 2021/01/11 23:00

January 2021

Sun	Mon	Tue	Wed	Thu	Fri	Sat
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

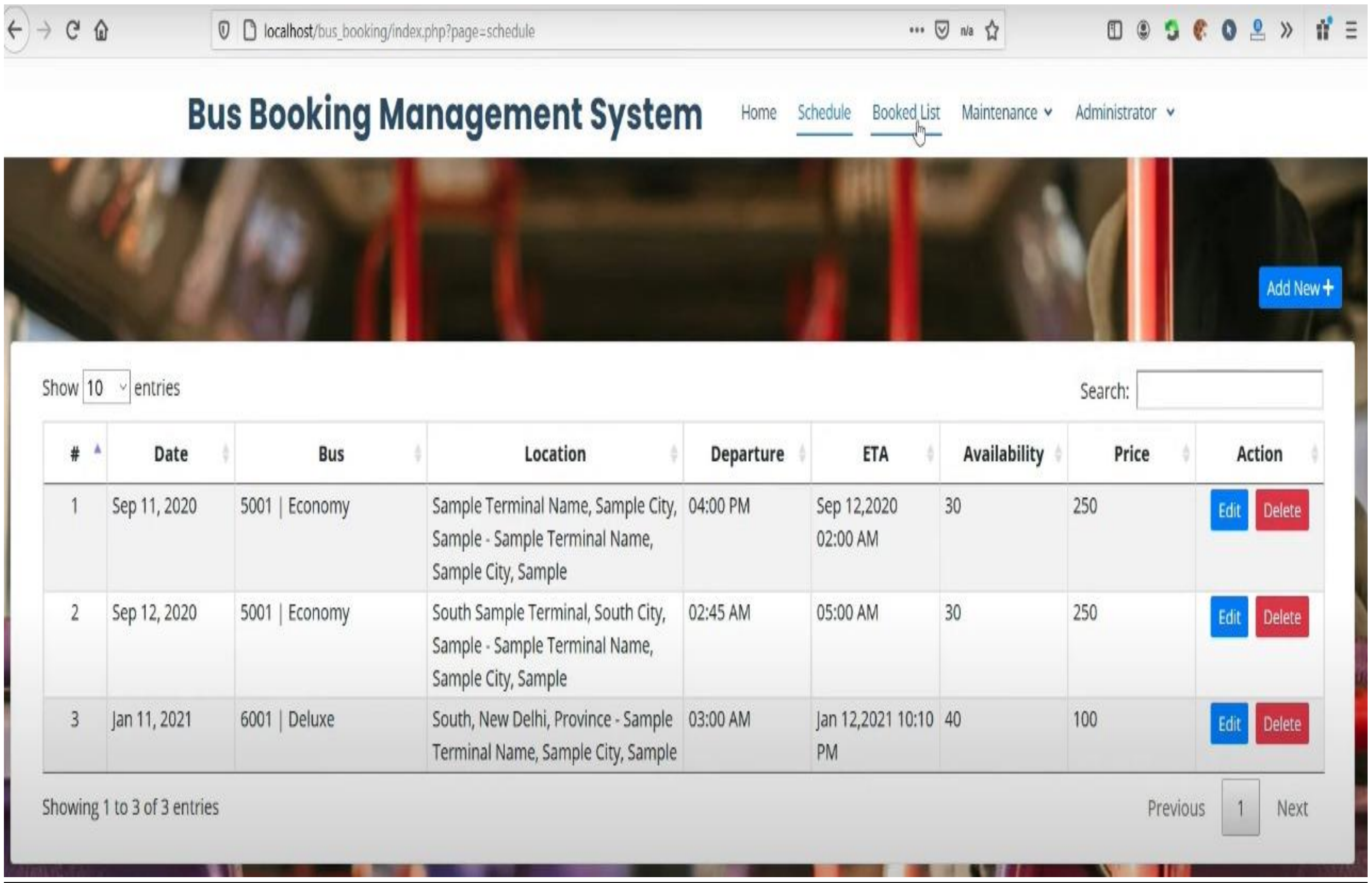
18:00
19:00
20:00
21:00
22:00
23:00

Save | Cancel

Show 10 entries | Sear

#	Date	Bus		Availability	
1	Sep 11, 2020	5001 | Economy	Sar Sar Sar	30	250
2	Sep 12, 2020	5001 | Economy	Sou Sar Sar	30	250

Showing 1 to 2 of 2 entries



#	Date	Bus	Location	Departure	ETA	Availability	Price	Action
1	Sep 11, 2020	5001 | Economy	Sample Terminal Name, Sample City, Sample - Sample Terminal Name, Sample City, Sample	04:00 PM	Sep 12,2020 02:00 AM	30	250	Edit Delete
2	Sep 12, 2020	5001 | Economy	South Sample Terminal, South City, Sample - Sample Terminal Name, Sample City, Sample	02:45 AM	05:00 AM	30	250	Edit Delete
3	Jan 11, 2021	6001 | Deluxe	South, New Delhi, Province - Sample Terminal Name, Sample City, Sample	03:00 AM	Jan 12,2021 10:10 PM	40	100	Edit Delete

CHAPTER 9

REFERENCE

1) https://projectsgeek.com/2017/04/bus-ticket-reservation-system-project.html

2) https://www.perplexity.ai/search/generate-me-a-refference-link-x2VMiJnrQlqERabfy3aiSg

3) https://chatgpt.com/

4) https://github.com/topics/bus-reservation?o=desc&s=stars

5) https://www.blackbox.ai/