TRAIN TICKET BOOKING SYSTEM

A MINI-PROJECT BY:

Akshay kumar 230701021

Daniel Leve Manickam 230701060

in partial fulfillment of the award of the degree

OF

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

An Autonomous Institute

CHENNAI

NOVEMBER 2024

BONAFIDE CERTIFICATE

Certified that this project **"TRAIN TICKET BOOKING SYSTEM"** is the bonafide work of **"AKSHAY KUMAR S , DANIEL LEVE MANICKAM D A"** who carried out the project work under my supervision.

Submitted for the practical examination held on _____

SIGNATURE

DR.B.DEEPA,
Assistant Professor(SG), Computer Science and Engineering, Rajalakshmi Engineering College (Autonomous),
Thandalam,Chennai-602105 .

INTERNAL EXAMINER EXTERNAL EXAMINER

ABSTRACT

The railway ticket booking system is a software application built using the Java Swing framework, designed to streamline the train reservation process. This system enables users to search for available trains, view detailed information, and book tickets according to their travel requirements. The application is integrated with a MySQL database, which stores essential backend data, such as train details, seat availability, and user reservations.

The project leverages JDBC (Java Database Connectivity) to establish a connection between the Java application and the MySQL database. This integration enables the system to retrieve real-time train schedules and manage booking transactions, ensuring that all reservation data is updated and securely stored. Users can create accounts, browse train schedules, and check seat availability before making bookings, with confirmations and ticket details accessible through the system.

This project not only demonstrates the use of Java Swing for creating an interactive GUI but also tests the practical application of JDBC for database connectivity, providing a smooth and efficient ticketing experience. The system is designed for user independence, allowing them to manage reservations and retrieve data as needed, with all interactions securely recorded in the MySQL backend.

TABLE OF CONTENTS

INTRODUCTION

1.1 PROJECT OVERVIEW

The Train Ticket Booking System is a desktop application developed using Java Swing for the graphical user interface (GUI) and MySQL as the backend database. The application helps users search for available trains between two stations, book tickets, cancel tickets, and view a list of booked tickets. It is designed to provide an efficient and straightforward way to handle railway ticket reservations.

1.2 OBJECTIVE

The objective of this project is to create a user-friendly and efficient train ticket booking system that allows passengers to:

- Search for available trains between any two stations.
- Book tickets by selecting the train, specifying the number of seats, and providing personal details.
- Cancel bookings if needed and update available seats accordingly.
- View a list of all previously booked tickets.

1.3 SCOPE OF THE APPLICATION

This system offers the following functionalities:

- Searching for available trains based on source and destination.
- Book tickets for a specific train by selecting the number of seats.
- Cancel tickets by providing a valid ticket number.
- Display a list of all booked tickets with details like ticket number, passenger name, and train ID.
- Interaction with a MySQL database to store and retrieve information related to trains, passengers, and bookings.

1.4 APPLICATION FEATURES

- **Search Available Trains**: Allows users to search for trains based on source and destination.
- **Book Tickets**: Users can book tickets by selecting a train, entering passenger details, and confirming the booking.
- **Cancel Tickets**: Users can cancel tickets by providing the ticket number.
- **View Booked Tickets**: Displays a list of all booked tickets.

SYSTEM SPECIFICATIONS

2.1 HARDWARE SPECIFICATIONS:

PROCESSOR : Intel i5

MEMORY SIZE : 4GB(Minimum) HARD DISK :

500 GB of free space

2.2 SOFTWARE SPECIFICATIONS:

PROGRAMMING LANGUAGE : Java, MySQL

FRONT-END : Java BACK-END : MySQL

OPERATING SYSTEM : Windows 10

SAMPLE CODE

3.1 GUI Design and Layout

The GUI setup uses Java Swing and includes components for user input and actions:

```
// Setting up the JFrame
setTitle("Train Ticket Booking System");
setSize(400, 500);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setLayout(new GridLayout(0, 2));

// Input fields
sourceField = new JTextField();
destinationField = new JTextField();
trainIdField = new JTextField();
ticketsField = new JTextField();
nameField = new JTextField();
phoneField = new JTextField();
ticketNumberField = new JTextField();

// Buttons with ActionListeners for respective operations
JButton viewTrainsButton = new JButton("View Available
Trains");
```

```
viewTrainsButton.addActionListener(e ->
viewAvailableTrains());

JButton bookTicketButton = new JButton("Book Ticket");
bookTicketButton.addActionListener(e -> bookTicket());

JButton cancelTicketButton = new JButton("Cancel Ticket");
cancelTicketButton.addActionListener(e -> cancelTicket());

JButton viewBookedTicketsButton = new JButton("View Booked
Tickets");
viewBookedTicketsButton.addActionListener(e ->
viewBookedTickets());
```

This code establishes the GUI layout using `GridLayout` and adds text fields, labels, and buttons to interact with the application.

3.2 Ticket Booking Operation

The **bookTicket** method is responsible for handling ticket booking. It performs the following steps:

```
private void bookTicket() {

    int trainId = Integer.parseInt(trainIdField.getText());

    int ticketsToBook =
Integer.parseInt(ticketsField.getText());

    String name = nameField.getText();

    String phoneNumber = phoneField.getText();

    String selectQuery = "SELECT available_seats FROM Trains
WHERE train_id = ?";

    String updateQuery = "UPDATE Trains SET available_seats =
available_seats - ? WHERE train_id = ?";

    String insertPassengerQuery = "INSERT INTO Passengers
(name, phone_number, ticket_number, train_id) VALUES (?, ?, ?,
?)";

  String insertTicketNumberQuery = "INSERT INTO TicketNumbers
(ticket_number, train_id) VALUES (?, ?)";
```

```
    try (Connection connection =
DriverManager.getConnection(DB_URL, USER, PASS);

         PreparedStatement selectStatement =
connection.prepareStatement(selectQuery);

         PreparedStatement updateStatement =
connection.prepareStatement(updateQuery);

         PreparedStatement insertPassengerStatement =
connection.prepareStatement(insertPassengerQuery);

         PreparedStatement insertTicketNumberStatement =
connection.prepareStatement(insertTicketNumberQuery)) {



        // Checking available seats and performing booking if
seats are available

        // Generating random ticket number and inserting
passenger and ticket information into the database

    } catch (SQLException e) {

        e.printStackTrace();

    }

}
```

This function checks seat availability, generates a random ticket number, and updates the database with the passenger and ticket details.

3.3 Ticket Cancellation Operation

The **cancelTicket** method enables users to cancel tickets using a ticket number:

```
private void cancelTicket() {

    int ticketNumber =
Integer.parseInt(ticketNumberField.getText());

    String selectQuery = "SELECT train_id FROM Passengers
WHERE ticket_number = ?";

    String updateQuery = "DELETE FROM Passengers WHERE
```

```
ticket_number = ?";

    String updateSeatsQuery = "UPDATE Trains SET
available_seats = available_seats + 1 WHERE train_id = ?";


    try (Connection connection =
DriverManager.getConnection(DB_URL, USER, PASS);

         PreparedStatement selectStatement =
connection.prepareStatement(selectQuery);

         PreparedStatement updateStatement =
connection.prepareStatement(updateQuery);

         PreparedStatement updateSeatsStatement =
connection.prepareStatement(updateSeatsQuery)) {


        // Retrieving the associated train ID, deleting
passenger record, and updating available seats

    } catch (SQLException e) {

        e.printStackTrace();

    }

}
```

This method validates the ticket number, removes the associated passenger record from the **Passengers** table, and updates seat availability in the **Trains** table.

3.3 Ticket Cancellation Operation

The **cancelTicket** method enables users to cancel tickets using a ticket number:

```
private void cancelTicket() {

    int ticketNumber =
Integer.parseInt(ticketNumberField.getText());
```

```
    String selectQuery = "SELECT train_id FROM Passengers
WHERE ticket_number = ?";

    String updateQuery = "DELETE FROM Passengers WHERE
ticket_number = ?";

    String updateSeatsQuery = "UPDATE Trains SET
available_seats = available_seats + 1 WHERE train_id = ?";


    try (Connection connection =
DriverManager.getConnection(DB_URL, USER, PASS);

         PreparedStatement selectStatement =
connection.prepareStatement(selectQuery);

         PreparedStatement updateStatement =
connection.prepareStatement(updateQuery);

         PreparedStatement updateSeatsStatement =
connection.prepareStatement(updateSeatsQuery)) {


        // Retrieving the associated train ID, deleting
passenger record, and updating available seats

    } catch (SQLException e) {

        e.printStackTrace();

    }

}
```

This method validates the ticket number, removes the associated passenger record from the **Passengers** table, and updates seat availability in the **Trains** table.

3.5 Database Queries and Interaction

This section includes the main SQL queries used in the application. The database has two primary tables:

- **Trains**: Stores train details, including seat availability.

- **Passengers**: Stores passenger information, ticket number, and associated train ID.

Database Schema

```
-- Trains Table
CREATE TABLE Trains (
    train_id INT PRIMARY KEY,
    train_name VARCHAR(100),
    source VARCHAR(50),
    destination VARCHAR(50),
    available_seats INT
);

-- Passengers Table
CREATE TABLE Passengers (
    passenger_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100),
    phone_number VARCHAR(15),
    ticket_number INT,
    train_id INT,
    FOREIGN KEY (train_id) REFERENCES Trains(train_id)
);
```

Key SQL Queries

1.View Available Trains

This query retrieves trains based on a user-specified source, destination, and available seats.
sql

```
SELECT * FROM Trains

WHERE source = ? AND destination = ? AND available_seats >= ?;
```

2.Book a Ticket

Booking a ticket involves three steps: checking seat availability, updating seat count, and inserting passenger details.

2.1Check Seat Availability

```
SELECT available_seats FROM Trains WHERE train_id = ?;
```

2.2Update Seat Count After Booking

```
UPDATE Trains

SET available_seats = available_seats - ?

WHERE train_id = ?;
```

2.3Insert Passenger Details

```
INSERT INTO Passengers (name, phone_number, ticket_number,
train_id)

VALUES (?, ?, ?, ?);
```

3.Cancel a Ticket

Canceling a ticket involves retrieving the train ID associated with the ticket, deleting the passenger record, and updating seat count.

3.1Retrieve Train ID for Ticket Number

```
SELECT train_id FROM Passengers WHERE ticket_number = ?;
```

3.2Delete Passenger Record

```
DELETE FROM Passengers WHERE ticket_number = ?;
```

3.3Update Seat Count After Cancellation

```
UPDATE Trains
SET available_seats = available_seats + 1
WHERE train_id = ?;
```

4.View Booked Tickets

This query retrieves all records from the **Passengers** table, showing ticket details for review.

```
SELECT * FROM Passengers;
```

FEATURE OVERVIEW

4.1 Booking

The **Booking Page** allows users to input their source and destination stations, select the train, and enter the number of tickets they want to book. It includes a "Book Ticket" button to finalize the booking.

4.2 Ticket Cancellation

The **Ticket Cancellation Page** provides a text field for the user to enter the ticket number and a "Cancel Ticket" button to delete the ticket and update available seats.

4.3 Available Trains Display

The **Available Trains Display** shows a list of trains that match the user's specified source and destination. It provides information such as train ID, departure time, arrival time, and available seats.

4.4 Booked Tickets Display

The **Booked Tickets Display** lists all the booked tickets with details such as ticket number, passenger name, and the train they are booked on. This page helps users track their reservations.

CONCLUSION

The Train Ticket Booking System is an effective tool for simplifying the railway ticket booking process. By integrating Java Swing for the user interface, JDBC connectivity for database interactions, and MySQL for the backend, the application offers a reliable and efficient system for managing train bookings. The project demonstrates how Java, JDBC connectivity, and SQL can be combined to create real-world applications with a focus on user-friendliness and functionality.

REFERENCES

1. **https://www.javatpoint.com/java-tutorial**

2. **MySQL JDBC Documentation**

3. **https://www.w3schools.com/sql/**

4. **MySQL 8.0 Reference Manual**