

CONCERT BOOKING MANAGEMENT SYSTEM

A MINI-PROJECT REPORT

Submitted by

M. AISHWARYA 230701016

ATHIENA RACHEL 230701046

In partial fulfilment of the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE

RAJALAKSHMI ENGINEERING COLLEGE (AUTONOMOUS)

THANDALAM

CHENNAI-602105

2023- 24

BONAFIDE CERTIFICATE

Certified that this project report “**CONCERT BOOKING MANAGEMENT SYSTEM**” is the Bonafide work of “**M. AISHWARYA (230701016), ATHIENA RACHEL(230701046),**”

who carried out the project work under my supervision.

Submitted for the Practical Examination held on _____

SIGNATURE

Dr.R.SABITHA
Professor and II Year Academic Head
Computer Science and Engineering,
Rajalakshmi Engineering College
(Autonomous),
Thandalam, Chennai - 602 105

SIGNATURE

Ms.V.JANANEE
Assistant Professor (SG),
Computer Science and Engineering,
Rajalakshmi Engineering College,
(Autonomous),
Thandalam, Chennai - 602 105

INTERNAL EXAMINER

EXTERNAL EXAMINER

ABSTRACT

Concert Booking Management System is a simple console application written in C that uses linked lists to manage concert bookings. The system allows users to book tickets, cancel bookings, and view booking records with user details. Each concert is identified by a unique concert ID, and users can book tickets by entering the concert ID along with their personal details.

The system ensures that users can only book one ticket for a concert at a time. When a user cancels their booking, the ticket becomes available for others. The system also provides a way to display the list of all current bookings, showing which concerts have been booked and by which users. Through basic operations, such as booking tickets, cancelling bookings, and viewing records, the system efficiently tracks concert attendance and user participation.

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

2.2.1 SQL

2.2.2 JAVA

3. REQUIREMENTS AND ANALYSIS

3.1 REQUIREMENT SPECIFICATION

3.2 HARDWARE AND SOFTWARE REQUIREMENTS

3.3 ARCHITECTURE DIAGRAM

4. PROGRAM CODE

5. RESULTS AND DISCUSSION

6. CONCLUSION

7. REFERENCES

CHAPTER 1

1. INTRODUCTION

In this Concert Booking Management System, users can perform basic operations like booking tickets for concerts and viewing booked ticket records along with user details.

Here a new user can sign-up to the interface by entering the necessary details and then with those details they can login to the interface.

If a user is already existing they can just login with the details.

After login, users can select the desired concert and book tickets accordingly.

If tickets for a particular concert is sold out the interface will show an error ,notifying the user immediately, else it will generate a bill.

This system offers a secure, user-friendly way to manage concert ticket booking.

SYSTEM SPECIFICATIONS

HARDWARE SPECIFICATIONS:

- **PROCESSOR** : Intel i5
- **MEMORY SIZE** : 4GB(Minimum)
- **HARD DISK** : 500 GB of free space

SOFTWARE SPECIFICATIONS:

- **PROGRAMMING LANGUAGE** : Java, MySQL
- **FRONT-END** : Java
- **BACK-END** : MySQL
- **OPERATING SYSTEM** : Windows 10

PROGRAM:

//Main.java

```
package application;
import javafx.application.Application;
import javafx.scene.effect.DropShadow;
import javafx.scene.paint.Color;
import javafx.scene.text.Font;
import javafx.scene.text.FontWeight;
import javafx.geometry.Insets;
import javafx.geometry.Pos;
import javafx.scene.Scene;
import javafx.scene.control.Alert;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.PasswordField;
import javafx.scene.control.TextField;
import javafx.scene.layout.BorderPane;
import javafx.scene.layout.VBox;
import javafx.scene.paint.Color;
import javafx.scene.paint.LinearGradient;
import javafx.scene.paint.Stop;
import javafx.scene.text.Font;
import javafx.stage.Stage;
import javafx.scene.layout.Background;
import javafx.scene.layout.BackgroundFill;
import javafx.scene.layout.CornerRadii;
import javafx.scene.control.*;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;

import javafx.scene.layout.*;
import javafx.scene.text.Text;
import javafx.scene.control.ComboBox;
import javafx.scene.control.DatePicker;
import java.time.LocalDate;
import java.util.Arrays;
import java.util.ArrayList;
import java.util.List;
import javafx.scene.layout.StackPane;
import java.io.File;
import java.sql.ResultSet;

import java.sql.Connection;
import java.sql.SQLException;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.Statement;
```

```

public class Main extends Application {
    public static String Loginn,emaill;
    public static long mobilee;
    private Stage primaryStage; // Store the primary stage for scene switching

    @Override
    public void start(Stage primaryStage) {
        this.primaryStage = primaryStage; // Initialize the primary stage

        // Start with the home scene
        showHomeScene();

        // Set the stage title
        primaryStage.setTitle("Concert Ticket Management System");
        primaryStage.show();
    }

    private void showHomeScene() {
        BorderPane root = new BorderPane();

        // Create a VBox for buttons
        VBox buttonLayout = new VBox(20); // 20 is the spacing between buttons
        buttonLayout.setAlignment(Pos.CENTER); // Center the buttons in the VBox
        buttonLayout.setStyle("-fx-padding: 20;"); // Add padding around the buttons

        // Create buttons
        Button loginButton = new Button("User Login");
        Button signUpButton = new Button("Sign Up");
        Button concertsButton = new Button("View Concerts");
        concertsButton.setVisible(false);
        // Style buttons
        styleButton(loginButton);
        styleButton(signUpButton);
        styleButton(concertsButton);

        // Set button actions
        loginButton.setOnAction(e -> showLoginScene());
        signUpButton.setOnAction(e -> showSignUpScene());
        concertsButton.setOnAction(e -> showConcertsScene());

        // Add buttons to the VBox
        buttonLayout.getChildren().addAll(loginButton, signUpButton, concertsButton);

        // Place the VBox at the center of the root layout
        root.setCenter(buttonLayout);
    }

```



```

        // Create the scene with the root layout
        Scene scene = new Scene(root, 800, 600);

scene.getStylesheets().add(getClass().getResource("application.css").toExternalForm());

        // Set the scene on the primary stage
        primaryStage.setScene(scene);
    }

    private void showLoginScene() {

        BorderPane root = new BorderPane();

        // Set a gradient background to resemble concert lighting
        root.setBackground(new Background(new BackgroundFill(
            new LinearGradient(0, 0, 1, 1, true, null,
                new Stop(0, Color.DARKBLUE),
                new Stop(1, Color.MEDIUMPURPLE)),
            CornerRadii.EMPTY, Insets.EMPTY)));

        // Create a rectangular VBox for the login form
        VBox layout = new VBox(15);
        layout.setPadding(new Insets(20));
        layout.setAlignment(Pos.CENTER);
        layout.setPrefWidth(350); // Set a fixed width for rectangular shape
        layout.setPrefHeight(400); // Set a fixed height to give it a rectangular shape

        // Set a background color for the login box
        layout.setBackground(new Background(new BackgroundFill(
            Color.rgb(30, 30, 30, 0.9), // Dark, semi-transparent box
            new CornerRadii(10), Insets.EMPTY)));
        layout.setStyle("-fx-border-color: #FF5733; -fx-border-width: 2px;"); // Border for
definition

        // Title label with vibrant color and larger font size
        Label title = new Label("Concert Ticket Login");
        title.setFont(Font.font("Arial", 26));
        title.setTextFill(Color.web("#FFDD44")); // Bright yellow color for clear
visibility

        // Username and password fields with labels
        Label usernameLabel = new Label("Username:");
        usernameLabel.setFont(Font.font("Arial", 20)); // Increase font size for labels
        usernameLabel.setTextFill(Color.web("#FFDD44"));
        TextField usernameField = new TextField();
        usernameField.setPrefWidth(250);
        usernameField.setStyle("-fx-background-color: #222; -fx-text-fill: white; -fx-
border-color: #4CAF50;");

        Label passwordLabel = new Label("Password:");
        passwordLabel.setFont(Font.font("Arial", 20)); // Increase font size for labels
        passwordLabel.setTextFill(Color.web("#FFDD44"));
        PasswordField passwordField = new PasswordField();
        passwordField.setPrefWidth(250);
        passwordField.setStyle("-fx-background-color: #222; -fx-text-fill: white; -fx-
border-color: #4CAF50;");

        // Login button with vibrant color and hover effect
        Button loginButton = new Button("Login");
        loginButton.setStyle("-fx-background-color: #FF5733; -fx-text-fill: white; -fx-font-

```

```

size: 14px; " +
    "-fx-padding: 10px 20px; -fx-border-radius: 5px; -fx-background-radius: 5px;");

    loginButton.setOnMouseEntered(e -> loginButton.setStyle("-fx-background-color:
#FF8C00; -fx-text-fill: white;"));
    loginButton.setOnMouseExited(e -> loginButton.setStyle("-fx-background-color:
#FF5733; -fx-text-fill: white;"));

    // Back to Home button
    Button backButton = new Button("Back to Home");
    backButton.setStyle("-fx-background-color: #4CAF50; -fx-text-fill: white; -fx-font-
size: 14px; " +
        "-fx-padding: 10px 20px; -fx-border-radius: 5px; -fx-background-radius: 5px;");

    backButton.setOnMouseEntered(e -> backButton.setStyle("-fx-background-color:
#81C784; -fx-text-fill: white;"));
    backButton.setOnMouseExited(e -> backButton.setStyle("-fx-background-color: #4CAF50;
-fx-text-fill: white;"));

    backButton.setOnAction(e -> showHomeScene());

    // Adding all components to the layout
    layout.getChildren().addAll(title, usernameLabel, usernameField, passwordLabel,
passwordField, loginButton, backButton);
    root.setCenter(layout);

    Scene scene = new Scene(root, 800, 600);
    primaryStage.setScene(scene);

    loginButton.setOnAction(e -> {

        String url1 = "jdbc:mysql://localhost:3306/concert_booking"; // Replace with
your DB URL
        String dbUsername1 = "root"; // Your DB username
        String dbPassword1 = "ramco"; // Your DB password

        String un=usernameField.getText();
        String pa=passwordField.getText();
        String sql="select count(*) as cou from users where username='" + un + "' and
password='" + pa + "'";

        Integer val=null;
        System.out.println("Login method called with username: " + sql); // Debugging
line

        try (Connection conn1 = DriverManager.getConnection(url1, dbUsername1,
dbPassword1);
            Statement st = conn1.createStatement())
        {
            ResultSet rs1 = st.executeQuery(sql);
            while (rs1.next())
            {
                val = rs1.getInt(1);
            }

            rs1.close();

```

```

        rs1 = st.executeQuery("select email,mobile from users where username='"
+ un + "'");
        while (rs1.next())
        {
            email= rs1.getString(1);
            mobile= rs1.getLong(2);
        }

        rs1.close();

        // Check if the user exists and the password matches
        if (val==1) {
            System.out.println("Login successful!"); // Debugging line
            Alert alert = new Alert(Alert.AlertType.INFORMATION);
            alert.setTitle("Login Successful");
            alert.setContentText("Welcome, " + un + "!");
            alert.showAndWait();
            loginn=un;
            showConcertsScene();
            // You can proceed with the next scene or functionality after login
        } else {
            System.out.println("Invalid username or password."); // Debugging
line
            Alert alert = new Alert(Alert.AlertType.ERROR);
            alert.setTitle("Login Failed");
            alert.setContentText("Invalid username or password. Please try
again.");

            usernameField.clear();
            passwordField.clear();

            alert.showAndWait();
        }

        } catch (SQLException ex) {
            // Handle database exceptions
            System.out.println("SQLException: " + ex.getMessage()); // Debugging
line
            Alert alert = new Alert(Alert.AlertType.ERROR);
            alert.setTitle("Database Error");
            alert.setContentText("An error occurred while connecting to the
database.");

            alert.showAndWait();
            ex.printStackTrace();
        }

    });

}

```

```

private void login(String username, String password) {

```

```

    /*
    * Integer val=null; System.out.println("Login method called with username: " +
    * username); // Debugging line
    *
    * String url = "jdbc:mysql://localhost:3306/concert_booking"; // Replace with
    * your DB URL String dbUsername = "root"; // Your DB username String

```

dbPassword

```
* = "ramco"; // Your DB password String query =
* "SELECT * FROM userlogin WHERE username = ? AND password = ?"; // SQL query
* for validation
*
* try (Connection conn = DriverManager.getConnection(url, dbUsername,
* dbPassword); Statement st = conn.createStatement()) { ResultSet rs =
* st.executeQuery(query); while (rs.next()) { val = rs.getInt(0); }
*
* rs.close();
*
* // Check if the user exists and the password matches if (val==1) {
* System.out.println("Login successful!"); // Debugging line Alert alert = new
* Alert(Alert.AlertType.INFORMATION); alert.setTitle("Login Successful");
* alert.setContentText("Welcome, " + username + "!"); alert.showAndWait(); //
* You can proceed with the next scene or functionality after login } else {
* System.out.println("Invalid username or password."); // Debugging line Alert
* alert = new Alert(Alert.AlertType.ERROR); alert.setTitle("Login Failed");
* alert.setContentText("Invalid username or password. Please try again.");
* alert.showAndWait(); }
*
* } catch (SQLException ex) { // Handle database exceptions
* System.out.println("SQLException: " + ex.getMessage()); // Debugging line
* Alert alert = new Alert(Alert.AlertType.ERROR);
* alert.setTitle("Database Error");
* alert.setContentText("An error occurred while connecting to the database.");
* alert.showAndWait(); ex.printStackTrace(); }
*/
    }

private void stylePasswordField(PasswordField passwordField) {
    passwordField.setStyle(
        "-fx-background-color: #f0f0f0; " + // Light gray background
        "-fx-border-color: #ccc; " + // Light border
        "-fx-border-radius: 5px; " +
        "-fx-padding: 10px; " + // Padding inside the field
        "-fx-font-size: 14px; " + // Font size
        "-fx-pref-width: 300px;"); // Preferred width
}

private void styleLoginButton(Button button) {
    button.setStyle(
        "-fx-background-color: #4CAF50; " + // Green background
        "-fx-text-fill: white; " +
        "-fx-font-size: 16px; " + // Larger font size
        "-fx-padding: 10px 20px; " +
        "-fx-border-radius: 5px; " +
        "-fx-background-radius: 5px; " +
        "-fx-effect: dropshadow(gaussian, rgba(0,0,0,0.5), 5, 0.0, 0, 1);");

    // Add hover effects
    button.setOnMouseEntered(e -> button.setStyle(
        "-fx-background-color: #45a049; " + // Darker green on hover
        "-fx-text-fill: white; " +
        "-fx-font-size: 16px; " +
        "-fx-padding: 10px 20px; " +
        "-fx-border-radius: 5px; " +
        "-fx-background-radius: 5px; " +
        "-fx-effect: dropshadow(gaussian, rgba(0,0,0,0.5), 5, 0.0, 0, 1);"));

    button.setOnMouseExited(e -> styleLoginButton(button)); // Reset to original style
}
```

```

}

private void showSignUpScene() {
    BorderPane root = new BorderPane();

    // Set a gradient background for the signup scene
    root.setBackground(new Background(new BackgroundFill(
        new LinearGradient(0, 0, 1, 1, true, null,
            new Stop(0, Color.DARKBLUE),
            new Stop(1, Color.MEDIUMPURPLE)),
        CornerRadii.EMPTY, Insets.EMPTY)));

    // Create a rectangular VBox for the sign-up form
    VBox layout = new VBox(15);
    layout.setPadding(new Insets(20));
    layout.setAlignment(Pos.CENTER);
    layout.setPrefWidth(350); // Set a fixed width for rectangular shape
    layout.setPrefHeight(450); // Set a fixed height for rectangular shape

    // Set a background color for the signup box
    layout.setBackground(new Background(new BackgroundFill(
        Color.rgb(30, 30, 30, 0.9), // Dark, semi-transparent box
        new CornerRadii(10), Insets.EMPTY)));
    layout.setStyle("-fx-border-color: #FF5733; -fx-border-width: 2px;"); // Border for
definition

    // Title label with vibrant color and larger font size
    Label title = new Label("Sign Up");
    title.setFont(Font.font("Arial", 26));
    title.setTextFill(Color.web("#FFDD44")); // Bright yellow color

    // Username, email, and password fields with labels
    Label usernameLabel = new Label("Username:");
    usernameLabel.setFont(Font.font("Arial", 20));
    usernameLabel.setTextFill(Color.web("#FFDD44"));
    TextField usernameField = new TextField();
    usernameField.setPrefWidth(250);
    usernameField.setStyle("-fx-background-color: #222; -fx-text-fill: white; -fx-
border-color: #4CAF50;");

    Label emailLabel = new Label("Email:");
    emailLabel.setFont(Font.font("Arial", 20));
    emailLabel.setTextFill(Color.web("#FFDD44"));
    TextField emailField = new TextField();
    emailField.setPrefWidth(250);
    emailField.setStyle("-fx-background-color: #222; -fx-text-fill: white; -fx-border-
color: #4CAF50;");

    Label passwordLabel = new Label("Password:");
    passwordLabel.setFont(Font.font("Arial", 20));
    passwordLabel.setTextFill(Color.web("#FFDD44"));
    PasswordField passwordField = new PasswordField();
    passwordField.setPrefWidth(250);
    passwordField.setStyle("-fx-background-color: #222; -fx-text-fill: white; -fx-
border-color: #4CAF50;");

    Label mobileLabel = new Label("Mobile:");
    mobileLabel.setFont(Font.font("Arial", 20));
    mobileLabel.setTextFill(Color.web("#FFDD44"));

```

```

        TextField mobileField = new TextField();
        mobileField.setPrefWidth(250);
        mobileField.setStyle("-fx-background-color: #222; -fx-text-fill: white; -fx-border-
color: #4CAF50;");

        // Sign-up button with vibrant color and hover effect
        Button signUpButton = new Button("Sign Up");
        signUpButton.setStyle("-fx-background-color: #FF5733; -fx-text-fill: white; -fx-
font-size: 14px; " +
            "-fx-padding: 10px 20px; -fx-border-radius: 5px; -fx-background-radius: 5px;");

        signUpButton.setOnMouseEntered(e -> signUpButton.setStyle("-fx-background-color:
#FF8C00; -fx-text-fill: white;"));
        signUpButton.setOnMouseExited(e -> signUpButton.setStyle("-fx-background-color:
#FF5733; -fx-text-fill: white;"));

        signUpButton.setOnAction(e -> {
            // Get values from the input fields
            String username = usernameField.getText();
            String email = emailField.getText();
            String password = passwordField.getText();
            long mob=Long.parseLong(mobileField.getText());
            // Call the signUp method
            signUp(username, email, password,mob);

            // Show an alert after signing up
            Alert alert = new Alert(Alert.AlertType.INFORMATION);
            alert.setTitle("Sign Up");
            alert.setContentText("Signing up with username: " + username + " and email: " +
email);
            alert.showAndWait();
        });

        // Back button to return to the home scene
        Button backButton = new Button("Back to Home");
        backButton.setStyle("-fx-background-color: #4CAF50; -fx-text-fill: white; -fx-font-
size: 14px; " +
            "-fx-padding: 10px 20px; -fx-border-radius: 5px; -fx-background-radius: 5px;");

        backButton.setOnMouseEntered(e -> backButton.setStyle("-fx-background-color:
#81C784; -fx-text-fill: white;"));
        backButton.setOnMouseExited(e -> backButton.setStyle("-fx-background-color: #4CAF50;
-fx-text-fill: white;"));

        backButton.setOnAction(e -> showHomeScene());

        // Adding all components to the layout
        layout.getChildren().addAll(title, usernameLabel, usernameField, passwordLabel,
passwordField,emailLabel, emailField,mobileLabel,mobileField, signUpButton, backButton);
        root.setCenter(layout);

        Scene scene = new Scene(root, 800, 600);
        primaryStage.setScene(scene);
    }
    private void signUp(String username, String email, String password,long mobile) {

        String url = "jdbc:mysql://localhost:3306/concert_booking"; // Replace with your DB
URL
        String dbUsername = "root"; // Your DB username
        String dbPassword = "ramco"; // Your DB password
    }

```

```

String sql="select count(*) as cou from users where username='" + username + "'";
Integer val=null;
System.out.println("Sign-up method called with username: " + sql); // Debugging line

try (Connection conn1 = DriverManager.getConnection(url, dbUsername, dbPassword);
     Statement st = conn1.createStatement())
{
    ResultSet rs1 = st.executeQuery(sql);
    while (rs1.next())
    {
        val = rs1.getInt(1);
    }

    rs1.close();
} catch (SQLException ex) {
    // Handle database exceptions
    System.out.println("SQLException: " + ex.getMessage()); // Debugging line
    Alert alert = new Alert(Alert.AlertType.ERROR);
    alert.setTitle("Database Error");
    alert.setContentText("An error occurred while connecting to the database.");
    alert.showAndWait();
    ex.printStackTrace();
}

```

```

String query = "INSERT INTO users (username, email, password,mobile) VALUES (?, ?, ?,?)"; // SQL query for insertion

```

```

try (Connection conn = DriverManager.getConnection(url, dbUsername, dbPassword);
     PreparedStatement pstmt = conn.prepareStatement(query)) {

```

```

    // Set the parameters for the query
    pstmt.setString(1, username);
    pstmt.setString(2, email);
    pstmt.setString(3, password);
    pstmt.setString(4, String.valueOf(mobile));
    // Execute the insert query

```

```

    // Show success or failure message based on query result

```

```

    if (val==0) {
        pstmt.executeUpdate();
        System.out.println("User registered successfully!"); // Debugging line
        Alert alert = new Alert(Alert.AlertType.INFORMATION);
        alert.setTitle("Sign Up Successful");
        alert.setContentText("User registered successfully!");
        alert.showAndWait();
    } else {

```

```

        System.out.println("Already user exists.Please select some other user
name"); // Debugging line
        Alert alert = new Alert(Alert.AlertType.ERROR);
        alert.setTitle("Sign Up Failed");
        alert.setContentText("Already user exists.Please select some other user
name");

```

```

        alert.showAndWait();
    }
    conn.close();
} catch (SQLException ex) {
    // Handle database exceptions
    System.out.println("SQLException: " + ex.getMessage()); // Debugging line
    Alert alert = new Alert(Alert.AlertType.ERROR);
    alert.setTitle("Database Error");
    alert.setContentText("An error occurred while connecting to the database.");
    alert.showAndWait();
    ex.printStackTrace();
}
}

```

```

private void bookTicket(String concertName) {
    // Example: Show an alert dialog when the user tries to book a ticket
    Alert alert = new Alert(Alert.AlertType.INFORMATION);
    alert.setTitle("Ticket Booking");
    alert.setHeaderText(null);
    alert.setContentText("You have booked a ticket for: " + concertName);
    alert.showAndWait();
}

```

```

private void showConcertsScene() {
    BorderPane root = new BorderPane();

    // Set a gradient background
    root.setBackground(new Background(new BackgroundFill(
        new LinearGradient(0, 0, 1, 1, true, null,
            new Stop(0, Color.DARKSLATEBLUE),
            new Stop(1, Color.DEEPPIK)),
        CornerRadii.EMPTY, Insets.EMPTY)));

    // Create an HBox for concert layout to arrange them horizontally
    HBox concertLayout = new HBox(30); // Horizontal spacing between concert items
    concertLayout.setAlignment(Pos.CENTER); // Center the HBox layout
    concertLayout.setPadding(new Insets(20));

    // Updated concert data
    String[] concertNames = {
        "Rhythms of Chennai: Arijit Singh Live",
        "Soul Beats: Neha Kakkar in Concert",
        "Symphony of Sounds: A. R. Rahman",
        "Bollywood Beats: Badshah's Live Performance"
    };
    String[] concertDates = {
        "Date: Dec 15, 2024",
        "Date: Jan 20, 2025",
        "Date: Feb 10, 2025",
        "Date: Mar 5, 2025"
    };
    String[] concertVenues = {
        "Nehru Indoor Stadium, Chennai",
        "Phoenix Marketcity, Chennai",
        "YMCA Grounds, Chennai",
        "VGP Universal Kingdom, Chennai"
    };
    String[] imagePaths = {

```

"https://media.insider.in/image/upload/c_crop,g_custom/v1676965539/cniesjacjimsjfeuzqc.png"


```

, // Arijit Singh
    "https://encrypted-
tbn0.gstatic.com/images?q=tbn:AND9GcR09PvLnPFtNYCydzBIAVAVJy1BxdL9yoZm8Q&s", // Neha Kakkar
    "https://i.ytimg.com/vi/n-acwP5pwlo/maxresdefault.jpg", // A. R. Rahman
    "https://encrypted-
tbn0.gstatic.com/images?q=tbn:AND9GcSACYuZH6tYSUXkMb8qaJwJtLTRFL0iRGSUqw&s" // Badshah
    };
    String[] locationLinks = {

        "https://www.google.com/maps/place/Jawaharlal+Nehru+Stadium/@13.0857373,80.2691727,17z/data=!3m1!4b1!4m6!3m5!1s0x3a5265f6e6a909ab:0x5a6046dfc9f0d784!8m2!3d13.0857373!4d80.2717476!16zL20vMdc4YzV5?entry=tту&_ep=EgoyMDI0MTAyOS4wIKXMDSoASAFQAw%3D%3D", // Arijit Singh

        "https://www.google.com/maps/place/Phoenix+Marketcity/@12.9929399,80.2152932,17z/data=!3m1!4b1!4m6!3m5!1s0x3a526763b48e60eb:0xdb3a29009036c251!8m2!3d12.9929399!4d80.2178681!16s%2Fg%2F1q54w6krf?entry=tту&_ep=EgoyMDI0MTAyOS4wIKXMDSoASAFQAw%3D%3D", // Neha Kakkar

        "https://www.google.com/maps/place/YMCA+Ground/@13.0243033,80.2340264,17z/data=!3m1!4b1!4m6!3m5!1s0x3a5267ae8bbeedcf:0x59928b97b499c64b!8m2!3d13.0243033!4d80.2366013!16s%2Fg%2F1hcbjbjzy?entry=tту&_ep=EgoyMDI0MTAyOS4wIKXMDSoASAFQAw%3D%3D", // A. R. Rahman

        "https://www.google.com/maps/place/VGP+Universal+Kingdom/@12.914221,80.2479491,17z/data=!3m1!4b1!4m6!3m5!1s0x3a525ce7cfa58535:0x96c3e0481b851d2f!8m2!3d12.914221!4d80.250524!16s%2Fm%2F05q6np0?entry=tту&_ep=EgoyMDI0MTAyOS4wIKXMDSoASAFQAw%3D%3D" // Badshah
    };

    String url1 = "jdbc:mysql://localhost:3306/concert_booking"; // Replace with your DB
    URL
    String dbUsername1 = "root"; // Your DB username
    String dbPassword1 = "ramco"; // Your DB password
    String sql="select * from concertmaster ";

    try (Connection conn3 = DriverManager.getConnection(url1, dbUsername1, dbPassword1);
        Statement st = conn3.createStatement())
    {
        ResultSet rs1 = st.executeQuery(sql);
        while (rs1.next())
        {

            // Image view for concert poster
            ImageView posterView = new ImageView();
            posterView.setFitWidth(200);
            posterView.setFitHeight(150);
            posterView.setPreserveRatio(true);

            try {
                Image posterImage = new Image(rs1.getString(6));
                posterView.setImage(posterImage);
            } catch (Exception e) {
                System.out.println("Error loading image: " + e.getMessage());
            }

            // Label for concert details
            Label concertTitle = new Label(rs1.getString(2));
            concertTitle.setFont(Font.font("Arial", FontWeight.BOLD, 16));
            concertTitle.setTextFill(Color.WHITE);

            Label concertDate = new Label(rs1.getString(3));

```

```

concertDate.setFont(Font.font("Arial", FontWeight.NORMAL, 14));
concertDate.setTextFill(Color.LIGHTGRAY);

Label concertVenue = new Label("Venue: " + rs1.getString(4));
concertVenue.setFont(Font.font("Arial", FontWeight.NORMAL, 14));
concertVenue.setTextFill(Color.LIGHTGRAY);

// Hyperlink for venue location
Hyperlink venueLocationLink = new Hyperlink("View on Map");
venueLocationLink.setOnAction(e -> {
    web browser
    getHostServices().showDocument(locationLinks[0]); // Open in the default

});

// Button to book tickets
Button bookTicketButton = new Button("Book Ticket");
bookTicketButton.setStyle("-fx-background-color: #FF5733; -fx-text-fill:
white; -fx-font-weight: bold; -fx-border-radius: 5; -fx-padding: 10 15;");
int conid=rs1.getInt(1);
String conname=rs1.getString(2);
bookTicketButton.setOnAction(e -> showTicketOptions(conid,conname));

// Create a VBox for each concert entry
VBox concertEntry = new VBox(10, posterView, concertTitle, concertDate,
concertVenue, bookTicketButton);
concertEntry.setAlignment(Pos.CENTER);
concertEntry.setStyle("-fx-padding: 20; -fx-background-color: rgba(255, 255,
255, 0.1); -fx-border-radius: 10; -fx-background-radius: 10; -fx-effect:
dropshadow(gaussian, rgba(0,0,0,0.3), 10, 0, 0, 0);");

// Add concert entry to the HBox
concertLayout.getChildren().add(concertEntry);

// Add the concert layout to the center of the root layout
root.setCenter(concertLayout);

// Create a Back button to return to the home scene
Button backButton = new Button("Back to Home");
backButton.setStyle("-fx-background-color: #444; -fx-text-fill: white; -fx-font-
weight: bold; -fx-padding: 10 20;");
backButton.setOnAction(e -> showHomeScene());
root.setBottom(backButton);
BorderPane.setAlignment(backButton, Pos.CENTER);

// Create the scene with the root layout
Scene scene = new Scene(root, 1000, 600);
primaryStage.setScene(scene);

}

rs1.close();

} catch (SQLException ex) {
// Handle database exceptions
System.out.println("SQLException: " + ex.getMessage()); // Debugging line
Alert alert = new Alert(Alert.AlertType.ERROR);
alert.setTitle("Database Error");
alert.setContentText("An error occurred while connecting to the database.");
alert.showAndWait();
ex.printStackTrace();
}

```

```
    }  
}
```

```
private void showTicketOptions(int conid,String concertName) {  
  
    // Create a dialog to show ticket options  
    Dialog<ButtonType> dialog = new Dialog<>();  
    dialog.setTitle("Select Ticket Type for " + concertName);  
    dialog.setHeaderText("Choose your ticket type and enter your details:");  
  
    // Create a VBox for the dialog content  
    VBox dialogPane = new VBox(10);  
    dialogPane.setPadding(new Insets(20));  
  
    // Ticket options with prices  
    String[] ticketTypes = {"Gold - ₹3000", "Silver - ₹2000", "Bronze - ₹1000"};  
    ComboBox<String> ticketTypeDropdown = new ComboBox<>();  
    ticketTypeDropdown.getItems().addAll(ticketTypes);  
    ticketTypeDropdown.setPromptText("Select Ticket Type");  
  
    // Ticket quantity selection  
    Label quantityLabel = new Label("Number of Tickets:");  
    Spinner<Integer> ticketQuantitySpinner = new Spinner<>(1, 10, 1); // Minimum 1 ticket,  
    maximum 10 tickets  
  
    // User details fields  
    TextField connamee = new TextField();  
    connamee.setText(concertName);  
    connamee.setDisable(true);  
    TextField nameField = new TextField();  
    nameField.setPromptText("Enter your name");  
    nameField.setText(Loginn);  
    TextField emailField = new TextField();  
    emailField.setPromptText("Enter your email");  
    emailField.setText(emaill);  
    TextField phoneField = new TextField();  
    phoneField.setPromptText("Enter your phone number");  
    phoneField.setText(String.valueOf(mobilee));  
  
    // Add all elements to the dialog pane  
    dialogPane.getChildren().addAll(new Label("Concert Name"),connamee,  
        new Label("Ticket Types:"), ticketTypeDropdown,  
        quantityLabel, ticketQuantitySpinner,  
        new Label("Your Details:"),  
        new Label("Name:"), nameField,  
        new Label("Email:"), emailField,  
        new Label("Phone Number:"), phoneField  
    );  
  
    // Create buttons for the dialog  
    ButtonType bookButton = new ButtonType("Book Ticket", ButtonBar.ButtonData.OK_DONE);  
    dialog.getDialogPane().getButtonTypes().addAll(bookButton, ButtonType.CANCEL);  
    dialog.getDialogPane().setContent(dialogPane);  
  
    // Handle the booking when the book button is clicked  
    dialog.setResultConverter(dialogButton -> {  
        if (dialogButton == bookButton) {  
            String selectedTicket = ticketTypeDropdown.getValue();  
            int quantity = ticketQuantitySpinner.getValue();  
        }  
    });  
}
```

```

String name = nameField.getText();
String email = emailField.getText();
String phone = phoneField.getText();

// Validate email and phone number
if (selectedTicket != null && !name.isEmpty() && isValidEmail(email) &&
isValidPhone(phone)) {
    // Extract the price from the selected ticket type
    int ticketPrice = Integer.parseInt(selectedTicket.split("₹")[1].replace(",",""));
    int totalPrice = ticketPrice * quantity;

    // Show confirmation alert with total price and user details
    Alert confirmationAlert = new Alert(Alert.AlertType.INFORMATION);
    confirmationAlert.setTitle("Booking Details");
    confirmationAlert.setHeaderText("Booking Details are :");
    confirmationAlert.setContentText(
        "Ticket Type1: " + selectedTicket + "\n" +
        "Concert: " + concertName + "\n" +
        "Quantity: " + quantity + "\n" +
        "Total Price: ₹" + totalPrice + "\n\n" +
        "Booked By:\nName: " + name + "\nEmail: " + email + "\nPhone: " + phone
    );
    confirmationAlert.showAndWait();
} else {
    // Show warning if required fields are not filled or invalid
    Alert warningAlert = new Alert(Alert.AlertType.WARNING);
    warningAlert.setTitle("Invalid Information");
    warningAlert.setHeaderText("Please check your details.");
    warningAlert.setContentText("Ensure all fields are filled correctly:\n"
        + "- Name should not be empty\n"
        + "- Email should be in the correct format\n"
        + "- Phone number should contain only digits and
be 10 digits long.");
    warningAlert.showAndWait();
    System.exit(0);
}

int totaltickets = 0,bookedtickets = 0,availabletickets=0;

String url = "jdbc:mysql://localhost:3306/concert_booking"; // Replace with your
DB URL
String dbUsername = "root"; // Your DB username
String dbPassword = "ramco"; // Your DB password
String sql;

sql=" select totaltickets from concertmaster where cid=" + conid;

try (Connection conn1 = DriverManager.getConnection(url, dbUsername,
dbPassword);
    Statement st = conn1.createStatement())
{
    ResultSet rs1 = st.executeQuery(sql);
    while (rs1.next())
    {
        totaltickets = rs1.getInt(1);
    }
}

```

```

    }

    rs1.close();

    sql="select cid,sum(nooftickets) from concerttransaction where cid=" +
conid + " group by cid";
    rs1 = st.executeQuery(sql);
    while (rs1.next())
    {
        bookedtickets = rs1.getInt(2);
    }

    rs1.close();

    availabletickets=totaltickets-bookedtickets;

} catch (SQLException ex) {
    // Handle database exceptions
    System.out.println("SQLException: " + ex.getMessage()); // Debugging
line
    Alert alert = new Alert(Alert.AlertType.ERROR);
    alert.setTitle("Database Error");
    alert.setContentText("An error occurred while connecting to the
database.");

    alert.showAndWait();
    ex.printStackTrace();
}

String query = "INSERT INTO concerttransaction (cid, bookedby,
tickettype,nooftickets) VALUES (?, ?, ?,?)"; // SQL query for insertion

try (Connection conn2 = DriverManager.getConnection(url, dbUsername,
dbPassword);
    PreparedStatement pstmt = conn2.prepareStatement(query)) {

    // Set the parameters for the query
    pstmt.setInt(1, conid);
    pstmt.setString(2, Loginn);
    pstmt.setString(3, selectedTicket);
    pstmt.setInt(4, quantity);

    int rowsAffected = 0;
    System.out.print("Tot"+totaltickets);
    System.out.print("Booked"+bookedtickets);
    System.out.print("Avai"+availabletickets);

    // Execute the insert query
    if(availabletickets>=quantity)
    {
        rowsAffected = pstmt.executeUpdate();
    }

    // Show success or failure message based on query result
    if (rowsAffected > 0) {
        System.out.println("Tickets Booked successfully!" + availabletickets);
// Debugging line
        Alert alert = new Alert(Alert.AlertType.INFORMATION);
        alert.setTitle("Booking Status");

```

```

        alert.setContentText("Tickets Booked successfully!");
        alert.showAndWait();
    } else {
        System.out.println("Tickets not available."); // Debugging line
        Alert alert = new Alert(Alert.AlertType.INFORMATION);
        alert.setTitle("Booking Status");
        alert.setContentText("Tickets not available. Please try for other
dates");
        alert.showAndWait();
    }
    conn2.close();
} catch (SQLException ex) {
    // Handle database exceptions
    System.out.println("SQLException: " + ex.getMessage()); // Debugging line
    Alert alert = new Alert(Alert.AlertType.ERROR);
    alert.setTitle("Database Error");
    alert.setContentText("An error occurred while connecting to the
database.");
    alert.showAndWait();
    ex.printStackTrace();
}

}
return null;
});

dialog.showAndWait();
}

```

```

private void storeTicketBooking(String concertName, String ticketType, int quantity, int
totalPrice, String name, String email, String phone) {
    String sql = "INSERT INTO tickets (concert_name, ticket_type, quantity, total_price,
name, email, phone) VALUES (?, ?, ?, ?, ?, ?, ?)";

```

```

    try (PreparedStatement stmt = connection.prepareStatement(sql)) {
        stmt.setString(1, concertName);
        stmt.setString(2, ticketType);
        stmt.setInt(3, quantity);
        stmt.setInt(4, totalPrice);
        stmt.setString(5, name);
        stmt.setString(6, email);
        stmt.setString(7, phone);

        int rowsAffected = stmt.executeUpdate();

        if (rowsAffected > 0) {
            System.out.println("Booking successful! Data saved in the database.");
        } else {
            System.out.println("Failed to save booking in the database.");
        }
    } catch (SQLException e) {
        System.out.println("Error saving booking: " + e.getMessage());
    }
}

```

```

// Email validation method
private boolean isValidEmail(String email) {
    // Basic email pattern for validation
    String emailPattern = "^[\\w-\\.]+@([\\w-]+\\.)+[\\w-]{2,4}$";
    return email.matches(emailPattern);
}

```

```

}

// Phone number validation method
private boolean isValidPhone(String phone) {
    // Check if phone has only digits and is 10 characters long
    return phone.matches("\\d{10}");
}


private void styleButton(Button button)
{
    button.setPrefWidth(150); // Set preferred width
    button.setStyle("-fx-font-size: 14; -fx-padding: 10;");
    button.setStyle(
        "-fx-background-color: #191970; " +
        "-fx-text-fill: white; " +
        "-fx-font-size: 14px; " +
        "-fx-padding: 10px 20px; " +
        "-fx-border-radius: 5px; " +
        "-fx-background-radius: 5px; " +
        "-fx-effect: dropshadow(gaussian, rgba(0,0,0,0.5), 5, 0.0, 0, 1);");

    button.setOnMouseEntered(e -> button.setStyle(
        "-fx-background-color: #005B99; " +
        "-fx-text-fill: white; " +
        "-fx-font-size: 14px; " +
        "-fx-padding: 10px 20px; " +
        "-fx-border-radius: 5px; " +
        "-fx-background-radius: 5px; " +
        "-fx-effect: dropshadow(gaussian, rgba(0,0,0,0.5), 5, 0.0, 0, 1);"));

    button.setOnMouseExited(e -> styleButton(button));

}

public static void main(String[] args)
{
    Launch(args);
}

}

```

//application.css

```
/* Setting background image for BorderPane */
.root {
    -fx-background-image: url("https://img.freepik.com/free-photo/back-view-excited-audience-with-arms-raised-cheering-front-stage-music-concert-copy-space_637285-538.jpg");
    -fx-background-size: cover; /* Makes the image cover the entire window */
    -fx-background-position: center; /* Centers the image */
}

.button {
    -fx-background-color: #5C6BC0; /* Background color */
    -fx-text-fill: white; /* Text color */
    -fx-font-size: 14px; /* Font size */
    -fx-padding: 10px 20px; /* Padding */
    -fx-border-radius: 5px; /* Rounded corners */
    -fx-background-radius: 5px; /* Rounded background */
    -fx-effect: dropshadow(gaussian, rgba(0,0,0,0.5), 5, 0.0, 0, 1); /* Shadow effect */
}

.button:hover {
    -fx-background-color: #3F51B5; /* Darker background on hover */
}

/* application.css */

/* Style for the login scene */

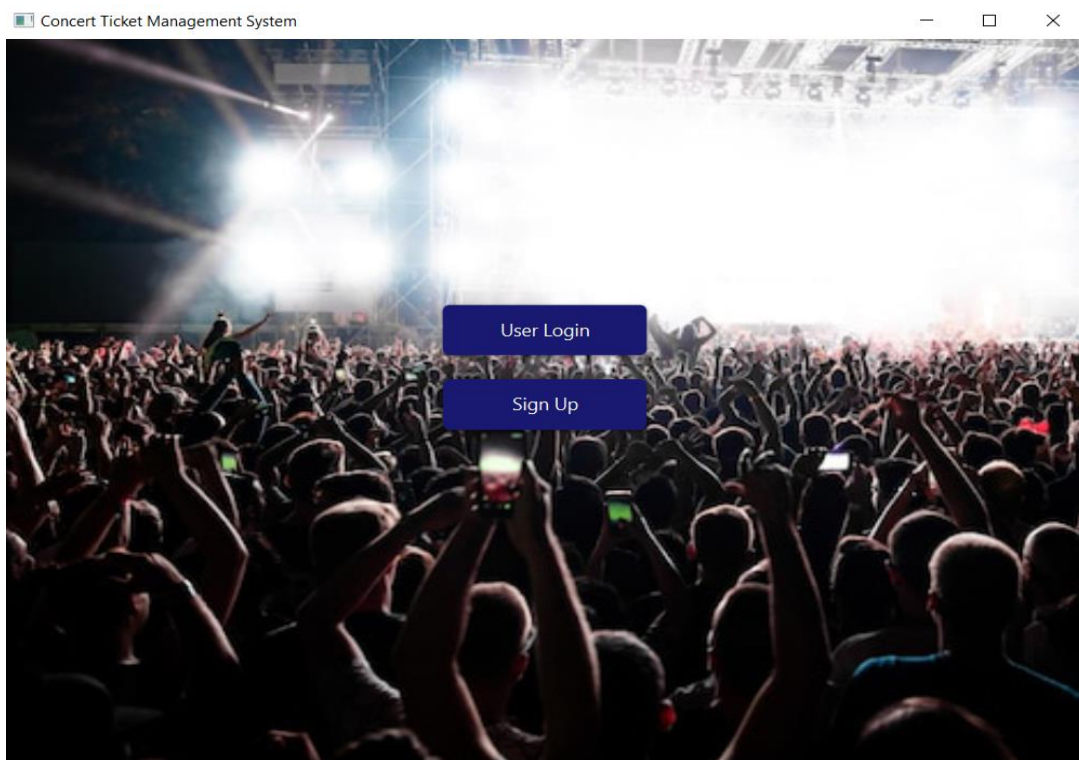
/* styles.css */
.login-background {
    -fx-background-image: url("https://cdn.prod.website-files.com/64ad227a3e66387fc5d89320/65cc59af2da41a018f861fae_concert-background-dd0syiox7rmi78l0.jpg");
    -fx-background-size: cover; /* Ensures the image covers the entire pane */
    -fx-background-position: center; /* Centers the image */
}

/* Optional: Style for the buttons */
.button {
    -fx-background-color: #4CAF50; /* Green background */
    -fx-text-fill: white; /* White text */
    -fx-font-size: 14px; /* Font size */
    -fx-padding: 10px; /* Padding inside the button */
}

/* Optional: Style for labels */
.label {
    -fx-font-size: 16px; /* Font size for labels */
    -fx-text-fill: #333; /* Dark text color */
}
```


SNAPSHOTS

HOME PAGE



SIGN UP PAGE

A screenshot of a web browser window titled "Concert Ticket Management System" showing the "Sign Up" page. The page has a dark blue background. At the top, the text "Sign Up" is displayed in a yellow font. Below it, the label "Username:" is followed by a text input field containing the name "Anushka". The label "Password:" is followed by a password input field represented by a series of dots. The label "Email:" is followed by a text input field containing "anu@gmail.com". The label "Mobile:" is followed by a text input field containing "9566170508". At the bottom of the form, there are two buttons: a red "Sign Up" button and a green "Back to Home" button. The browser window includes standard OS controls in the top right corner.

Concert Ticket Management System

Sign Up

Username:

Anushka

Message

.....

anu@gmail.com

Mobile:

9566170508

Sign Up

Back to Home

Sign Up Successful

Message

User registered successfully!

OK

Sign Up

Message

Signing up with username: Anushka and email: anu@gmail.com

OK

LOGIN PAGE

Concert Ticket Management System

Concert Ticket Login

Username:

Anushka

Password:

••••••

Login

Back to Home

Login Successful


Message

Welcome, Anushka!

OK

CONCERT PAGE

Concert Ticket Management System




Rhythms of Chennai: Arijit Singh Live

2024-12-15 18:00:00

Venue: Nehru Indoor Stadium, Chennai

Book Ticket



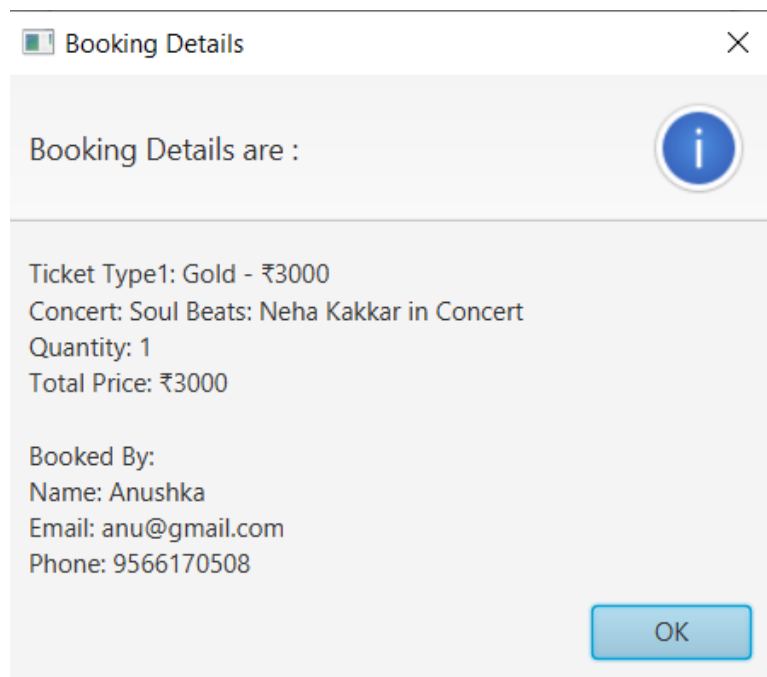
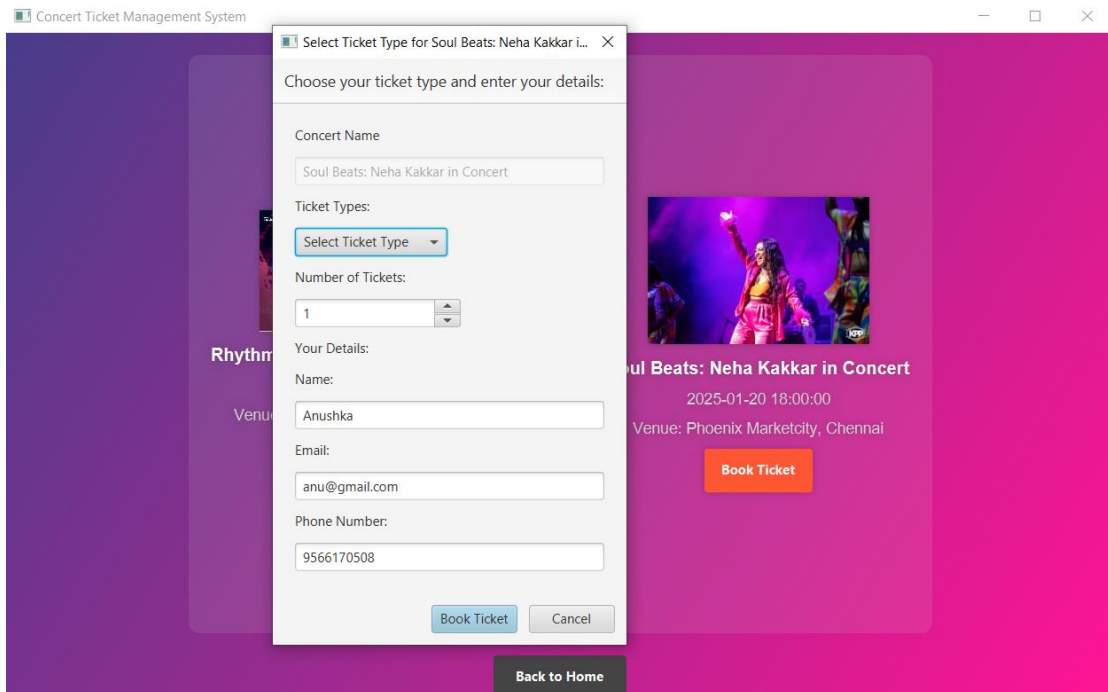
Soul Beats: Neha Kakkar in Concert

2025-01-20 18:00:00

Venue: Phoenix Marketcity, Chennai

Book Ticket

Back to Home



Select Ticket Type for Soul Beats: Neha Kakkar i... X

Choose your ticket type and enter your details:

Concert Name

Booking Status X

Message

Tickets Booked successfully!

OK

Name:

Anushka

Email:

anu@gmail.com

Phone Number:

9566170508

Book Ticket

Cancel

Concert Ticket Management System

Select Ticket Type for Rhythms of Chennai: Arijit... X

Choose your ticket type and enter your details:

Concert Name

Rhythms of Chennai: Arijit Singh Live

Ticket Types:

Silver - ₹2000

Number of Tickets:

3

Your Details:

Name:

Anushka

Email:

anu@gmail.com

Phone Number:

9566170508

Book Ticket

Cancel

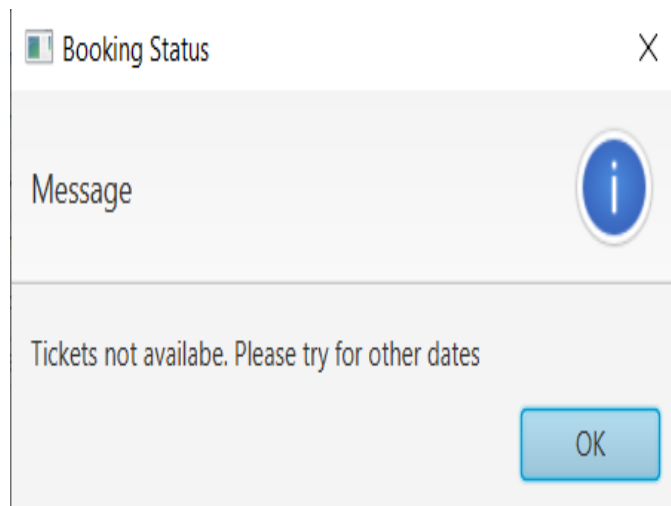
Back to Home

Soul Beats: Neha Kakkar in Concert

2025-01-20 18:00:00

Venue: Phoenix Marketcity, Chennai

Book Ticket



RESULTS AND DISCUSSION

The Concert Booking Management System provides a seamless experience for users to sign up, log in, browse concerts, and book tickets. Users can easily register with basic details like username, email, and password. After logging in, they can browse available concerts, select ticket types, and book tickets. The system checks for ticket availability, generates a bill if seats are available, and notifies users of sold-out concerts with an error message.

The system is user-friendly and ensures accurate ticket availability, but challenges like **concurrency management** and **scalability** should be addressed to handle high traffic and simultaneous bookings effectively.

CONCLUSION

The **Concert Booking Management System** efficiently manages concert reservations, offering a secure and user-friendly interface for users to book tickets. With real-time availability checks and error handling, the system ensures a smooth experience. Future improvements can focus on handling high-demand scenarios and enhancing user experience with personalized features.

REFERENCES

- 1** **MY SQL TUTORIAL**
[Learn MySQL Tutorial - javatpoint](#)
- 2** **JavaFX Tutorial**
[JavaFX Tutorial - javatpoint](#)