

MOVIE BOOKING SYSTEM

A MINI PROJECT REPORT

SUBMITTED BY

DHANVIN P R

230701071

DHARMARAJ

230701074

In partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

IN

**COMPUTER SCIENCE RAJALAKSHMI ENGINEERING
COLLEGE (AUTONOMOUS)**

THANDALAM

CHENNAI-602105

2024 – 25

**RAJALAKSHMI ENGINEERING COLLEGE
CHENNAI**

BONAFIDE CERTIFICATE

Certified that this project report called “**MOVIE BOOKING SYSTEM**” is the Bonafide work of “ **DHARMARAJ T (230701074) DHANVIN P R (230701071)** ” who carried out the project work under my supervision.

Submitted for the Practical Examination held on _____

SIGNATURE

MRS. DIVYA M
ASSISTANT PROFESSOR
COMPUTER SCIENCE AND
ENGINEERING , RAJALAKSHMI
ENGINEERING COLLEGE ,
THANDALAM, CHENNAI – 602105

SIGNATURE

MR. RAGHU
ASSISTANT PROFESSOR
COMPUTER SCIENCE
AND ENGINEERING ,
RAJALAKSHMI
ENGINEERING COLLEGE ,
THANDALAM, CHENNAI
– 602105

INTERNAL EXAMINER

EXTERNAL EXAMINER

ABSTRACT:

D2D Movies is a Java-based mini-project designed to provide an efficient and user-friendly platform for movie ticket booking. This system, developed using Java Swing (JFrame) for the frontend and MySQL for the backend, is specifically tailored for small-scale cinemas. The application offers distinct functionalities for users and administrators, ensuring an organized and seamless ticket booking experience.

The system begins with a secure login interface, allowing users and administrators to access their respective functionalities. Users can browse a curated list of available movies, with detailed information such as movie titles, genres, durations, show timings, ticket prices, and the number of available seats. Upon selecting a movie, users can choose the desired number of seats for booking. The system calculates the total cost based on the selected seats and generates a confirmation of the booking. While the seat availability updates occur after each transaction, the system operates in a structured sequential manner rather than in real-time concurrency.

For administrators, **D2D Movies** provides essential tools for managing the movie database. Administrators can add new movies, update existing details such as show timings, ticket prices, and seat availability, or delete outdated records. These functionalities help maintain the integrity and accuracy of the database, ensuring a smooth operational workflow.

The backend is powered by MySQL, where all data, including user credentials, movie details, seat availability, and booking records, is securely stored. Data interaction between the Java application and MySQL is handled using JDBC (Java Database Connectivity), ensuring a stable and efficient connection for retrieving and updating information.

In summary, **D2D Movies** is a practical and efficient solution for small-scale cinemas looking to digitize their ticket booking process. The system focuses on ease of use, structured functionality, and reliable database management, offering an enhanced experience for both users and administrators.

TABLE OF CONTENTS :

Chapter 1

1 INTRODUCTION

1.1 INTRODUCTION -----	6
1.2 OBJECTIVES -----	7

Chapter 2

2 LANGUAGES

2.1 SQL-----	8
2.2 JAVA-----	8

Chapter 3

3 DIAGRAMS

3.1 ARCHITECTURE DIAGRAM -----	10
3.2 ER DIAGRAM -----	10

Chapter 4

4 PROGRAM CODE

4.1 PROGRAM CODE-----	11
-----------------------	----

Chapter 5

5 RESULTS

5.1 RESULTS-----	27
------------------	----

Chapter 6

6 CONCLUSION

6.1 CONCLUSION-----	29
---------------------	----

Chapter 7

7 REFERENCES

7.1 REFERENCES-----	30
---------------------	----

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

D2D Movies is a Java-based mini-project developed to provide a structured and efficient platform for booking movie tickets. With the rise of digital platforms, cinema operations are increasingly transitioning from manual systems to automated solutions to enhance user convenience and streamline management tasks. This project aims to simulate such a system by combining the versatility of Java Swing for the user interface and the reliability of MySQL for database management.

The application is designed to cater to two primary roles: **users** and **administrators**. Users can log in to explore available movies, view details such as ticket prices and seat availability, and book tickets with ease. The system calculates the total ticket cost based on the number of seats selected and provides a confirmation upon successful booking. On the other hand, administrators have access to tools for managing the database, allowing them to add, update, or delete movie details, ensuring that the platform remains up-to-date and functional.

While the system does not operate in real-time concurrency, it provides an efficient and sequential workflow that ensures accurate updates to the database after every transaction. Integration with MySQL via JDBC ensures smooth data handling, making the application a reliable and practical solution for small-scale cinema operations.

In summary, **D2D Movies** combines simplicity, functionality, and scalability to provide an intuitive movie ticket booking system, serving as an ideal project for learning and implementing concepts in database management, Java programming, and application development.

1.2 OBJECTIVES

1. Simplify movie ticket booking for users.
2. Ensure efficient database management using MySQL.
3. Provide role-based functionality for users and administrators.
4. Enable seamless integration between the Java frontend and MySQL backend via JDBC.
5. Calculate ticket prices dynamically and update seat availability.
6. Serve as a practical learning platform for Java, Swing, and database integration.
7. Enhance administrative efficiency through database management tools.
8. Offer a scalable and adaptable solution for future enhancements.

CHAPTER 2

LANGUAGES

2.1 MYSQL

Role:

MySQL serves as the **Database Management System (DBMS)** for **D2D Movies**, storing and managing data such as user credentials, movie details, seat availability, and booking records.

Advantages:

1. **Efficient Data Management:** Handles large datasets effectively.
2. **Reliability:** Ensures stable and secure data storage.
3. **Easy Integration:** Works seamlessly with Java via JDBC.
4. **High Performance:** Fast data retrieval and updates.
5. **Cost-Effective:** Open-source and free.
6. **Data Integrity:** Maintains consistency and accuracy.

Usage:

1. Stores user credentials and movie records.
2. Tracks seat availability and updates after bookings.
3. Allows admins to manage movie records.
4. Executes queries for data retrieval and updates.

2.2 JAVA

Role:

Java is the primary programming language used for developing the **D2D Movies** application. It is responsible for creating the user interface, handling business logic, managing user interactions, and connecting to the MySQL database via JDBC.

Advantages:

1. **Platform Independence:** Write once, run anywhere (cross-platform compatibility).

2. **Object-Oriented:** Promotes modular, reusable, and maintainable code.
3. **Rich API:** Provides a wide range of libraries and tools for UI development (Swing), database connectivity (JDBC), and more.
4. **Security:** Strong security features, making it ideal for handling sensitive data.
5. **Scalability:** Easily scalable to accommodate additional features or larger user bases.

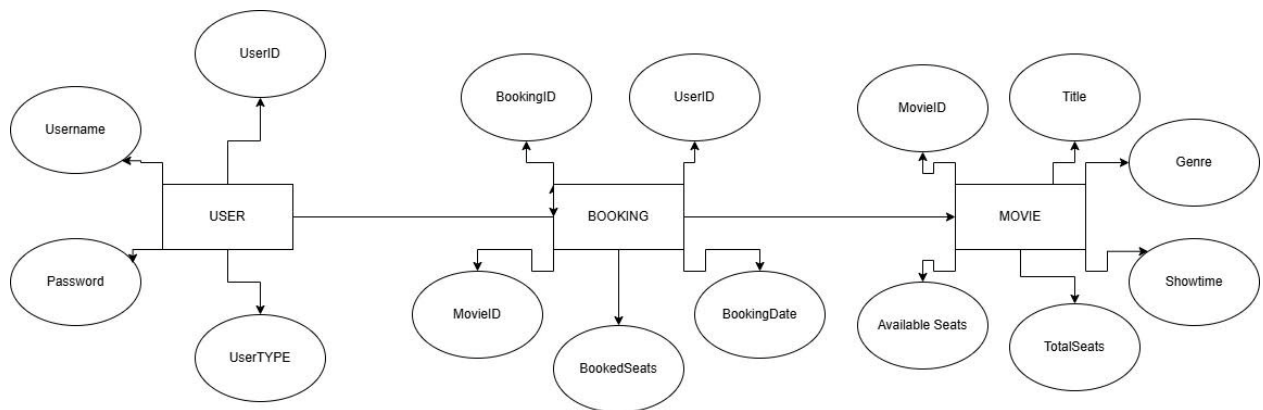
Usage::

1. **User Interface (UI):** Developed using Java Swing (JFrame) for a graphical interface.
2. **Business Logic:** Manages seat bookings, ticket calculations, and database operations.
3. **Database Connectivity:** Uses JDBC to interact with MySQL for storing and retrieving data.
4. **Role-Based Access:** Implements login and access control for users and administrators.

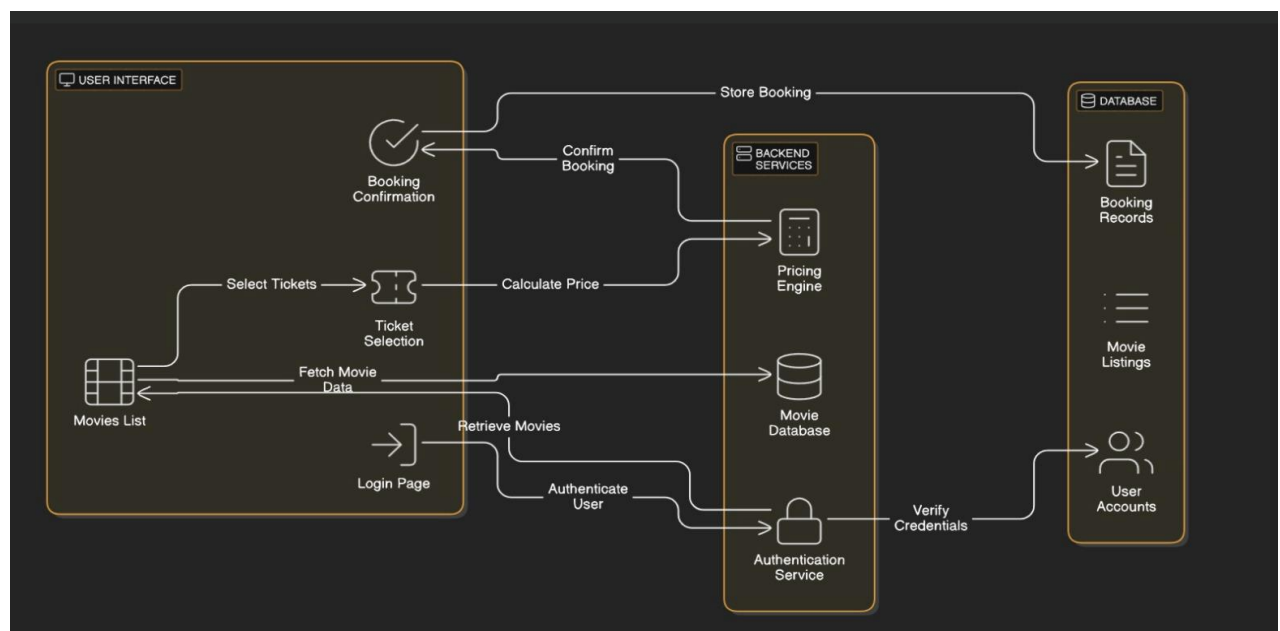
CHAPTER 3

DIAGRAMS

3.1 ER DIAGRAM:



3.2 ARCHITECTURE DIAGRAM :



4.1 LOGIN PAGE

```
package database;

import javax.swing.*;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

public class LoginPage extends javax.swing.JFrame {

    private javax.swing.JButton loginButton;

    private javax.swing.JButton registerButton;

    private javax.swing.JLabel passwordLabel;

    private javax.swing.JPasswordField passwordField;

    private javax.swing.JLabel usernameLabel;

    private javax.swing.JTextField usernameField;

    private javax.swing.ButtonGroup userTypeGroup;

    private javax.swing.JLabel userTypeLabel;

    private javax.swing.JLabel actionLabel;

    private javax.swing.JPasswordField jPasswordField;

    public LoginPage() {

        initComponents();

    }

    @SuppressWarnings("unchecked")

    // <editor-fold defaultstate="collapsed" desc="Generated Code">

    private void initComponents() {
```

```

jComboBox1 = new javax.swing.JComboBox<>();

jLabel1 = new javax.swing.JLabel();

jTextField1 = new javax.swing.JTextField();

jLabel2 = new javax.swing.JLabel();

REGISTER = new javax.swing.JButton();

LOGIN = new javax.swing.JButton();

jLabel3 = new javax.swing.JLabel();

jPasswordField1 = new javax.swing.JPasswordField();

jButton1 = new javax.swing.JButton();

jLabel4 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

getContentPane().setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());

jComboBox1.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] {
"User", "Admin", " ", " " }));

jComboBox1.addActionListener(new java.awt.event.ActionListener() {

    public void actionPerformed(java.awt.event.ActionEvent evt) {

        jComboBox1ActionPerformed(evt);

    }

});

getContentPane().add(jComboBox1, new
org.netbeans.lib.awtextra.AbsoluteConstraints(76, 90, -1, -1));

jLabel1.setFont(new java.awt.Font("Segoe UI", 1, 12)); // NOI18N

jLabel1.setForeground(new java.awt.Color(255, 255, 255));

jLabel1.setText("User Name");

getContentPane().add(jLabel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(76,
124, 72, -1));

```

```

jTextField1.addActionListener(new java.awt.event.ActionListener() {

    public void actionPerformed(java.awt.event.ActionEvent evt) {

        jTextField1ActionPerformed(evt);

    }

});

getContentPane().add(jTextField1, new
org.netbeans.lib.awtextra.AbsoluteConstraints(214, 118, 71, -1));

jLabel2.setFont(new java.awt.Font("Segoe UI", 1, 12)); // NOI18N
jLabel2.setForeground(new java.awt.Color(255, 255, 255));
jLabel2.setText("Password");

getContentPane().add(jLabel2, new org.netbeans.lib.awtextra.AbsoluteConstraints(76,
152, 64, -1));

REGISTER.setBackground(new java.awt.Color(153, 153, 255));

REGISTER.setText("REGISTER");

REGISTER.addActionListener(new java.awt.event.ActionListener() {

    public void actionPerformed(java.awt.event.ActionEvent evt) {

        REGISTERActionPerformed(evt);

    }

});

getContentPane().add(REGISTER, new
org.netbeans.lib.awtextra.AbsoluteConstraints(76, 201, -1, -1));

LOGIN.setBackground(new java.awt.Color(153, 153, 255));

LOGIN.setText("LOGIN");

LOGIN.addActionListener(new java.awt.event.ActionListener() {

    public void actionPerformed(java.awt.event.ActionEvent evt) {

        LOGINActionPerformed(evt);

    }

}

```

```

    });

    getContentPane().add(LOGIN, new org.netbeans.lib.awtextra.AbsoluteConstraints(214,
201, -1, -1));

    jLabel3.setFont(new java.awt.Font("Algerian", 1, 24)); // NOI18N

    jLabel3.setForeground(new java.awt.Color(255, 255, 255));

    jLabel3.setText("D2D MOVIES");

    getContentPane().add(jLabel3, new org.netbeans.lib.awtextra.AbsoluteConstraints(130,
33, -1, -1));

    getContentPane().add(jPasswordField1, new
org.netbeans.lib.awtextra.AbsoluteConstraints(214, 152, 71, -1));

    jButton1.setBackground(new java.awt.Color(51, 255, 255));

    jButton1.setForeground(new java.awt.Color(51, 0, 0));

    jButton1.setText("Next Page");

    jButton1.addActionListener(new java.awt.event.ActionListener() {

        public void actionPerformed(java.awt.event.ActionEvent evt) {

            jButton1ActionPerformed(evt);

        }

    });

    getContentPane().add(jButton1, new org.netbeans.lib.awtextra.AbsoluteConstraints(147,
250, -1, -1));

    jLabel4.setIcon(new javax.swing.ImageIcon(getClass().getResource("/Ve series online,
ve películas online.jpg"))); // NOI18N

    getContentPane().add(jLabel4, new org.netbeans.lib.awtextra.AbsoluteConstraints(0, 0,
400, 300));

    pack();

    setLocationRelativeTo(null);

} // </editor-fold>

```

```

private void jComboBox1ActionPerformed(java.awt.event.ActionEvent evt) {

    JOptionPane.showMessageDialog(this, "Registration feature coming soon!");

}

private void jTextField1ActionPerformed(java.awt.event.ActionEvent evt) {

}

private void LOGINActionPerformed(java.awt.event.ActionEvent evt) {

    String username = jTextField1.getText(); // Username field

    String password = new String(jPasswordField1.getPassword()); // Correct password field

    if (username.isEmpty() || password.isEmpty()) {

        JOptionPane.showMessageDialog(this, "Please enter both username and password.");

        return;

    }

    String userType = (String) jComboBox1.getSelectedItem(); // Get the selected user type

    if (userType == null || userType.trim().isEmpty()) {

        JOptionPane.showMessageDialog(this, "Please select 'User' or 'Admin'.");

        return;

    }

    try (Connection conn = DatabaseConnection.getConnection()) {

        String query = "SELECT * FROM user WHERE username = ? AND password = ?";

        PreparedStatement pst = conn.prepareStatement(query);

        pst.setString(1, username);

        pst.setString(2, password);

        ResultSet rs = pst.executeQuery();

        if (rs.next()) {

```

```

        if ("Admin".equals(userType)) {

            JOptionPane.showMessageDialog(this, "Admin Login Successful!");

        } else {

            JOptionPane.showMessageDialog(this, "User Login Successful!");

        }

    } else {

        JOptionPane.showMessageDialog(this, "Invalid credentials. Please try again.");

    }

} catch (Exception e) {

    e.printStackTrace();

    JOptionPane.showMessageDialog(this, "Error: " + e.getMessage());

}

}

private void REGISTERActionPerformed(java.awt.event.ActionEvent evt) {

    String userType = (String) jComboBox1.getSelectedItem(); // Get user type from
combo box

    String username = jTextField1.getText(); // Get username from text field

    char[] passwordCharArray = jPasswordField1.getPassword(); // Get password from
JPasswordField

    String password = new String(passwordCharArray); // Convert password to String

    if (username.isEmpty() || password.isEmpty()) {

        JOptionPane.showMessageDialog(this, "Please enter both username and password.");

        return; // Stop further processing if either field is empty

    }

    if ("User".equals(userType)) {

        try (Connection conn = DatabaseConnection.getConnection()) {

```



```

String query = "INSERT INTO user (username, password) VALUES (?, ?)";

PreparedStatement pst = conn.prepareStatement(query);

pst.setString(1, username); // Set username in the query

pst.setString(2, password); // Set password in the query

int rowsInserted = pst.executeUpdate();

if (rowsInserted > 0) {

    JOptionPane.showMessageDialog(this, "Registration Successful!");

    this.setVisible(false); // Hide the current window

    new Movies().setVisible(true); // Show the movies page

}

} catch (Exception e) {

    e.printStackTrace();

    JOptionPane.showMessageDialog(this, "Error: " + e.getMessage()); // Display error
message

}

} else {

    JOptionPane.showMessageDialog(this, "Please select 'User' to register.");

}

}

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

System.out.println("Button clicked!");

this.setVisible(false);

Movies moviesFrame = new Movies();

moviesFrame.setVisible(true);

}

```

```

public static void main(String args[]) {

    try {

        for (javax.swing.UIManager.LookAndFeelInfo info :
            javax.swing.UIManager.getInstalledLookAndFeels()) {

            if ("Nimbus".equals(info.getName())) {

                javax.swing.UIManager.setLookAndFeel(info.getClassName());

                break;

            }

        }

    } catch (ClassNotFoundException ex) {

        java.util.logging.Logger.getLogger(LoginPage.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

    } catch (InstantiationException ex) {

        java.util.logging.Logger.getLogger(LoginPage.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

    } catch (IllegalAccessException ex) {

        java.util.logging.Logger.getLogger(LoginPage.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

    } catch (javax.swing.UnsupportedLookAndFeelException ex) {

        java.util.logging.Logger.getLogger(LoginPage.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

    }

    java.awt.EventQueue.invokeLater(new Runnable() {

        public void run() {

            new LoginPage().setVisible(true);

        }

    });

```

```

        });
    }

    private javax.swing.JButton LOGIN;

    private javax.swing.JButton REGISTER;

    private javax.swing.JButton jButton1;

    private javax.swing.JComboBox<String> jComboBox1;

    private javax.swing.JLabel jLabel1;

    private javax.swing.JLabel jLabel2;

    private javax.swing.JLabel jLabel3;

    private javax.swing.JLabel jLabel4;

    private javax.swing.JPasswordField jPasswordField1;

    private javax.swing.JTextField jTextField1;
}

```

MOVIES PAGE:

```

package database;

public class Movies extends javax.swing.JFrame {

    public Movies() {

        System.out.println("Movies JFrame is being constructed...");

        initComponents();

    }

    private void initComponents() {

```

```
jLabel3 = new javax.swing.JLabel();

jLabel1 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

jLabel4 = new javax.swing.JLabel();

jLabel5 = new javax.swing.JLabel();

jLabel6 = new javax.swing.JLabel();

jLabel7 = new javax.swing.JLabel();

jLabel8 = new javax.swing.JLabel();

jButton1 = new javax.swing.JButton();

jButton2 = new javax.swing.JButton();

jButton3 = new javax.swing.JButton();

jButton4 = new javax.swing.JButton();

jButton5 = new javax.swing.JButton();

jButton6 = new javax.swing.JButton();

jLabel9 = new javax.swing.JLabel();

jLabel3.setText("jLabel3");


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

getContentPane().setLayout(new
org.netbeans.lib.awtextra.AbsoluteLayout());

jLabel1.setBackground(new java.awt.Color(153, 255, 102));

jLabel1.setFont(new java.awt.Font("Segoe UI", 1, 12)); // NOI18N

jLabel1.setForeground(new java.awt.Color(255, 255, 255));

jLabel1.setText(" TITANIC");
```

```
        getContentPane().add(jLabel1, new
org.netbeans.lib.awtextra.AbsoluteConstraints(303, 162, 53, -1));

        jLabel2.setFont(new java.awt.Font("Segoe UI", 1, 12)); // NOI18N
        jLabel2.setForeground(new java.awt.Color(255, 255, 255));
        jLabel2.setText(" VETTAIYAN ");

        getContentPane().add(jLabel2, new
org.netbeans.lib.awtextra.AbsoluteConstraints(79, 78, -1, -1));

        jLabel4.setFont(new java.awt.Font("Segoe UI", 1, 12)); // NOI18N
        jLabel4.setForeground(new java.awt.Color(255, 255, 255));
        jLabel4.setText(" AMARAN ");

        getContentPane().add(jLabel4, new
org.netbeans.lib.awtextra.AbsoluteConstraints(81, 162, -1, -1));

        jLabel5.setFont(new java.awt.Font("Segoe UI", 1, 12)); // NOI18N
        jLabel5.setForeground(new java.awt.Color(255, 255, 255));
        jLabel5.setText(" G.O.A.T");

        getContentPane().add(jLabel5, new
org.netbeans.lib.awtextra.AbsoluteConstraints(300, 78, 63, -1));

        jLabel6.setFont(new java.awt.Font("Segoe UI", 1, 12));
        jLabel6.setForeground(new java.awt.Color(255, 255, 255));
        jLabel6.setText(" KAITHI");

        getContentPane().add(jLabel6, new
org.netbeans.lib.awtextra.AbsoluteConstraints(86, 239, 53, -1));

        jLabel7.setFont(new java.awt.Font("Segoe UI", 1, 12));
        jLabel7.setForeground(new java.awt.Color(255, 255, 255));
```

```

jLabel7.setText(" THE DARK KNIGHT ");

getContentPane().add(jLabel7, new
org.netbeans.lib.awtextra.AbsoluteConstraints(281, 239, -1, -1));

jLabel8.setFont(new java.awt.Font("Algerian", 1, 36)); // NOI18N
jLabel8.setForeground(new java.awt.Color(255, 255, 255));
jLabel8.setText("MOVIES");

getContentPane().add(jLabel8, new
org.netbeans.lib.awtextra.AbsoluteConstraints(150, 30, -1, -1));

jButton1.setBackground(new java.awt.Color(153, 153, 255));
jButton1.setText("Book Now");
jButton1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton1ActionPerformed(evt);
    }
});

getContentPane().add(jButton1, new
org.netbeans.lib.awtextra.AbsoluteConstraints(70, 106, -1, -1));

jButton2.setBackground(new java.awt.Color(153, 153, 255));
jButton2.setText("Book Now");
jButton2.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton2ActionPerformed(evt);
    }
});

```

```
getContentPane().add(jButton2, new  
org.netbeans.lib.awtextra.AbsoluteConstraints(288, 106, -1, -1));
```

```
jButton3.setBackground(new java.awt.Color(153, 153, 255));
```

```
jButton3.setText("Book Now");
```

```
getContentPane().add(jButton3, new  
org.netbeans.lib.awtextra.AbsoluteConstraints(70, 184, -1, -1));
```

```
jButton4.setBackground(new java.awt.Color(153, 153, 255));
```

```
jButton4.setText("Book Now");
```

```
jButton4.addActionListener(new java.awt.event.ActionListener() {
```

```
    public void actionPerformed(java.awt.event.ActionEvent evt) {
```

```
        jButton4ActionPerformed(evt);
```

```
    }
```

```
});
```

```
getContentPane().add(jButton4, new  
org.netbeans.lib.awtextra.AbsoluteConstraints(288, 184, -1, -1));
```

```
jButton5.setBackground(new java.awt.Color(153, 153, 255));
```

```
jButton5.setText("Book Now");
```

```
getContentPane().add(jButton5, new  
org.netbeans.lib.awtextra.AbsoluteConstraints(70, 261, -1, -1));
```

```
jButton6.setBackground(new java.awt.Color(153, 153, 255));
```

```
jButton6.setText("Book Now");
```

```
        getContentPane().add(jButton6, new  
org.netbeans.lib.awtextra.AbsoluteConstraints(293, 261, -1, -1));
```

```
        jLabel9.setIcon(new  
javax.swing.ImageIcon(getClass().getResource("/Screenshot 2024-11-12  
084859.png"))); // NOI18N
```

```
        getContentPane().add(jLabel9, new  
org.netbeans.lib.awtextra.AbsoluteConstraints(0, 0, 441, 310));
```

```
    pack();
```

```
    setLocationRelativeTo(null);
```

```
}
```

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
```

```
}
```

```
private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
```

```
}
```

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
```

```
Vettaiyan vettaiyanFrame = new Vettaiyan();
```

```
vettaiyanFrame.setVisible(true);
```

```
this.dispose();
```

```
}
```

```
public static void main(String args[]) {
```

```
    try {
```



```

        for (javax.swing.UIManager.LookAndFeelInfo info :
             javax.swing.UIManager.getInstalledLookAndFeels()) {

            if ("Nimbus".equals(info.getName())) {

                javax.swing.UIManager.setLookAndFeel(info.getClassName());

                break;

            }

        }

    } catch (ClassNotFoundException ex) {

        java.util.logging.Logger.getLogger(Movies.class.getName()).log(java.util.logging.
            Level.SEVERE, null, ex);

    } catch (InstantiationException ex) {

        java.util.logging.Logger.getLogger(Movies.class.getName()).log(java.util.logging.
            Level.SEVERE, null, ex);

    } catch (IllegalAccessException ex) {

        java.util.logging.Logger.getLogger(Movies.class.getName()).log(java.util.logging.
            Level.SEVERE, null, ex);

    } catch (javax.swing.UnsupportedLookAndFeelException ex) {

        java.util.logging.Logger.getLogger(Movies.class.getName()).log(java.util.logging.
            Level.SEVERE, null, ex);

    }

    java.awt.EventQueue.invokeLater(new Runnable() {

        public void run() {

            new Movies().setVisible(true);

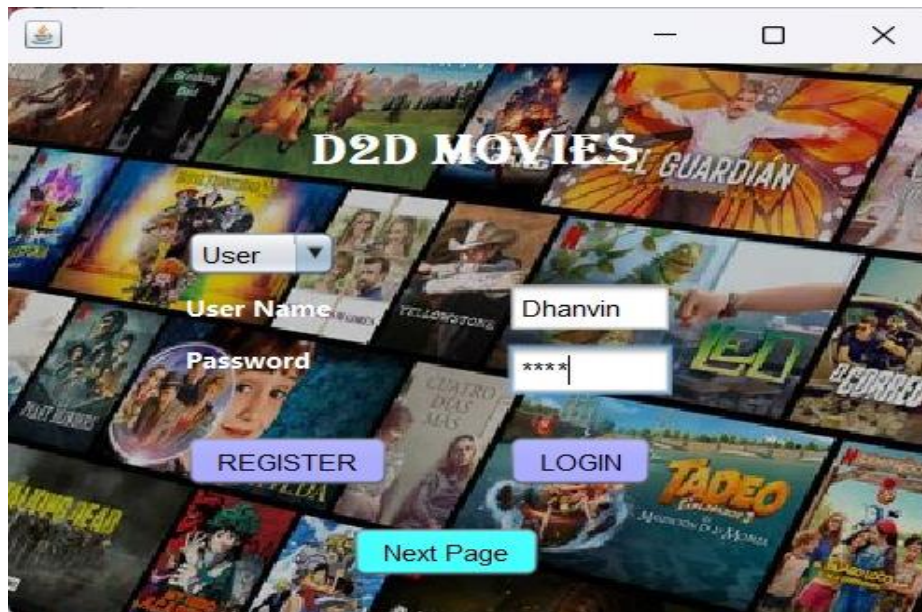
```

```
        }  
    });  
}  
  
private javax.swing.JButton jButton1;  
private javax.swing.JButton jButton2;  
private javax.swing.JButton jButton3;  
private javax.swing.JButton jButton4;  
private javax.swing.JButton jButton5;  
private javax.swing.JButton jButton6;  
private javax.swing.JLabel jLabel1;  
private javax.swing.JLabel jLabel2;  
private javax.swing.JLabel jLabel3;  
private javax.swing.JLabel jLabel4;  
private javax.swing.JLabel jLabel5;  
private javax.swing.JLabel jLabel6;  
private javax.swing.JLabel jLabel7;  
private javax.swing.JLabel jLabel8;  
private javax.swing.JLabel jLabel9;  
  
// End of variables declaration  
}
```

CHAPTER 5

RESULTS

5.1 RESULTS




LOGIN PAGE



MOVIES PAGE

VETTAIYAN
Ticket Rate :150 Rs



No. of Se...
5

Amount :
750

Book Now

	id	username	password
▶	1	dhanvin	123
	2	dhaanya	dhaanya
	3	RAGU	234
	4	iman	imran
★	NULL	NULL	NULL

VETTAIYAN
Ticket Rate :150 Rs

No. of Se...

Amount : 750

Book Now

Booking Confirmation

BOOKED SUCCESSFULLY

OK

6.1 CONCLUSION

In conclusion, the **D2D Movies** application successfully combines the power of Java and MySQL to create a comprehensive movie ticket booking system that is both user-friendly and efficient. By leveraging Java's capabilities for the frontend development through Swing (JFrame), and using MySQL for reliable data storage and management, the application provides a smooth and secure booking experience. It simplifies the process for users by offering easy navigation to browse movies, check seat availability, and book tickets. At the same time, it offers administrators powerful tools to manage and maintain movie records, seating arrangements, and pricing details, ensuring that the database is kept up-to-date and accurate.

The integration of Java and MySQL through JDBC ensures that all data transactions—such as ticket bookings, movie updates, and seat reservations—are handled efficiently, securely, and reliably. The use of role-based authentication allows for controlled access to the system, distinguishing between the functionalities available to users and administrators. This contributes to a seamless operation where users can focus on browsing and booking tickets, while administrators can manage the backend system with ease.

Furthermore, the **D2D Movies** system demonstrates the effectiveness of object-oriented programming principles, such as modularity, reusability, and maintainability, all of which are crucial for building scalable and adaptable software. The project also serves as a valuable learning experience for anyone interested in database management, Java programming, and application development, offering hands-on exposure to key concepts like JDBC, user interface design, and backend integration.

While currently designed for small-scale cinema operations, the **D2D Movies** system can easily be scaled up to accommodate more features or a larger user base, making it a flexible solution for businesses looking to digitize their movie ticket booking process. Overall, **D2D Movies** is a robust, efficient, and scalable application that not only meets the needs of cinema owners but also enhances the movie-going experience for users, providing them with a seamless and interactive platform for booking movie tickets online.

7.1 REFERENCES

1. **Java Documentation.** (n.d.). *Official Java documentation*. Retrieved from <https://docs.oracle.com/javase/>
Details on Java programming, including Swing and JDBC integration for building the **D2D Movies** app.
2. **MySQL Documentation.** (n.d.). *Official MySQL documentation*. Retrieved from <https://dev.mysql.com/doc/>
Information on MySQL features, database management, and integration with Java via JDBC.
3. *Head First Java* by Kathy Sierra and Bert Bates. (2005). A beginner-friendly guide to Java, covering object-oriented programming, Swing, and database connectivity.
4. *Java: The Complete Reference* by Herbert Schildt. (2021). A comprehensive guide to Java, including JDBC and integrating databases like MySQL.
5. **JDBC Tutorial.** (n.d.). *Official Java JDBC tutorial*. Retrieved from <https://docs.oracle.com/javase/tutorial/jdbc/>
A tutorial on using JDBC for database interactions in Java applications.
6. **Swing Tutorial.** (n.d.). *Java Swing tutorial*. Retrieved from <https://docs.oracle.com/javase/tutorial/uiswing/>
A guide for building graphical user interfaces (GUIs) with Java Swing.