**JAVA SWING BASED -BOOK MY TICKET-SQL**

**CONNECTIVITY USING JDBC**

*A*

*Report*

*Submitted in partial fulfilment of the*

*Requirements for the award of the Degree of*

***BACHELOR OF TECHNOLOGY***

***IN***

**INFORMATION TECHNOLOGY**

By

**R.THIRUPATHI<1602-20-737-031>**

**Under the Guidance of**

**B. Leelavathy**



**Department of Information Technology**

**Vasavi College of Engineering (Autonomous)**

**(Affiliated to Osmania University)**

**Ibrahimbagh, Hyderabad-31**

**2021-2022**

# BONAFIDE CERTIFICATE

This to Certify that the project report titled

**“BOOK MY TICKET”** project work of Mr.R.THIRUPATHI bearing Roll.no:1602-20-737-031 who carried out this project under my supervision in the IV semester for the academic year 2021-2022.

*Signature*   *Signature*   *external examiner internal examiner*

**ABSTRACT:**

In this project we book ticket using online movie Ticket reservation system.

We enter into Web page by logging with User Name and Password. Then we select the Movie and later in which Theatre movie is running. Later choose Show Timings and enter no of tickets you want .Finally it displays the details of the procedure and print the form to show at respective ticket counter to get ticket.

**Introduction: REQUIREMENT ANALYSIS**

**REQUIREMENT ANALYSIS:**

**List of Tables**:

Movie

Cinema hall

Booking

Clinet

**List of Attributes with their Domain Types:**

**Movie**

Movie\_id

Title

Duration

Language

release\_date

**Cinema\_hall:**

Cinemahall\_id

total\_seats

Name

Location

**Booking:**

Booking\_id

Status

**Client:**

Client**\_**name

Client\_id

Email

Phone

Date : day-date **THROUGH THE PROJECT:**

This project helps to store data in a efficient way and it can be achieved through various sql commands and we can also store this for any future use and also we can save our data in a many different areas so we cannot lost all the data at once. The quality and product details are must in now a days because quality matters every where, these project stores details and feedback and testing details in database so that whenever it is necessary to know how and when a product can be used.

**ARCHITECTURE AND TECHNOLOGY USED:**

**SOFTWARE USED:**

Java Eclipse, Oracle 11g Database, Java SE version 8,SQL Plus.

**Java SWING:**

**Swing** is a GUI widget toolkit for Java. It is part of Oracle's Java Foundation Classes (JFC) – an API for providing a graphical user interface (GUI) for Java programs.

Swing was developed to provide a more sophisticated set of GUI components than the earlier AWT. Swing provides a look and feel that emulates the look and feel of several platforms, and also supports a pluggable look and feel that allows applications to have a look and feel unrelated to the underlying platform. It has more powerful and flexible components than AWT. In addition to familiar components such as buttons, check boxes and labels, Swing provides several advanced components such as tabbed panel, scroll panes, trees, tables, and lists.

**SQL:**

Structure Query Language (SQL) is a database query language used for storing and managing data in Relational DBMS. SQL was the first commercial language introduced for E.F Codd's **Relational** model of database. Today almost all RDBMS (MySql, Oracle, Infomix, Sybase, MS Access) use **SQL** as the standard database query language. SQL is used to perform all types of data operations in RDBMS.

**Java-SQL Connectivity using JDBC:**

**Java Database Connectivity** (**JDBC**) is an application programming interface (API) for the programming language Java, which defines how a client may access a database. It is a Java-based data access technology used for Java database connectivity. It is part of the Java Standard Edition platform, from Oracle Corporation. It provides methods to query and update data in a database and is oriented towards relational databases.

The connection to the database can be performed using Java programming (JDBC API) as:

try {

Class.*forName*("oracle.jdbc.driver.OracleDriver");

}

catch (Exception e)

{

System.*err*.println("Unable to find and load driver");

System.*exit*(1);

}

public void connectToDB()

{

try

{

connection =

DriverManager.*getConnection*("jdbc:oracle:thin:@localhost:1522:ORCL","mydbms","mydbms"); statement = connection.createStatement();

}

catch (SQLException connectException)

{

System.*out*.println(connectException.getMessage());

System.*out*.println(connectException.getSQLState());

System.*out*.println(connectException.getErrorCode());

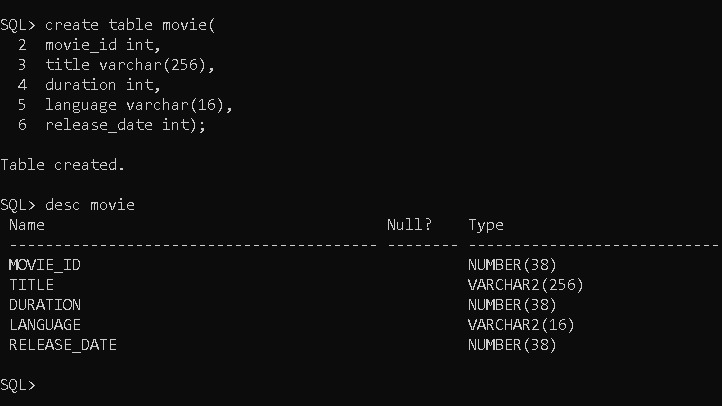
System.*exit*(1);

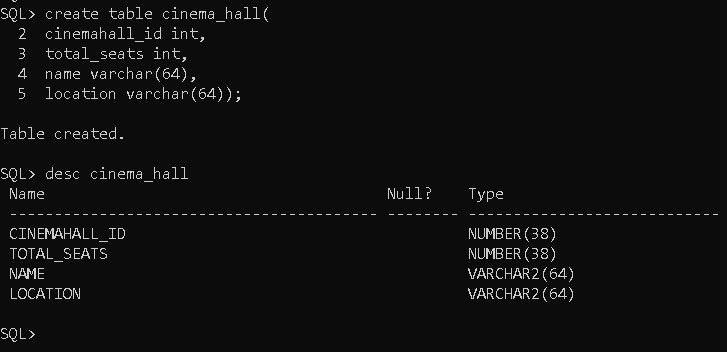
}

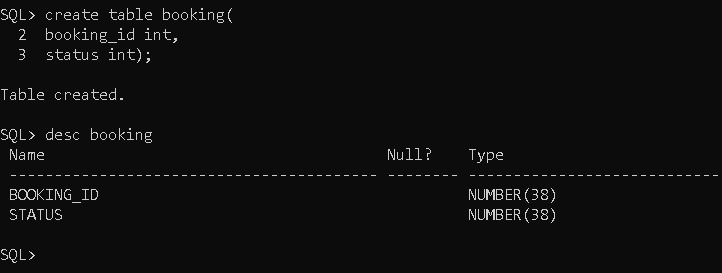
}

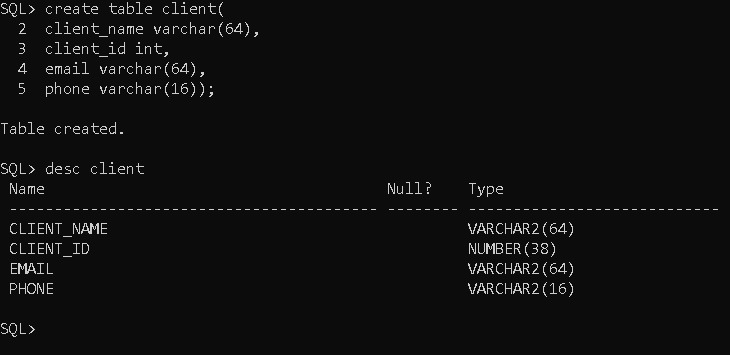
Thus, the connection from Java to Oracle database is performed and therefore, can be used for updating tables in the database directly.

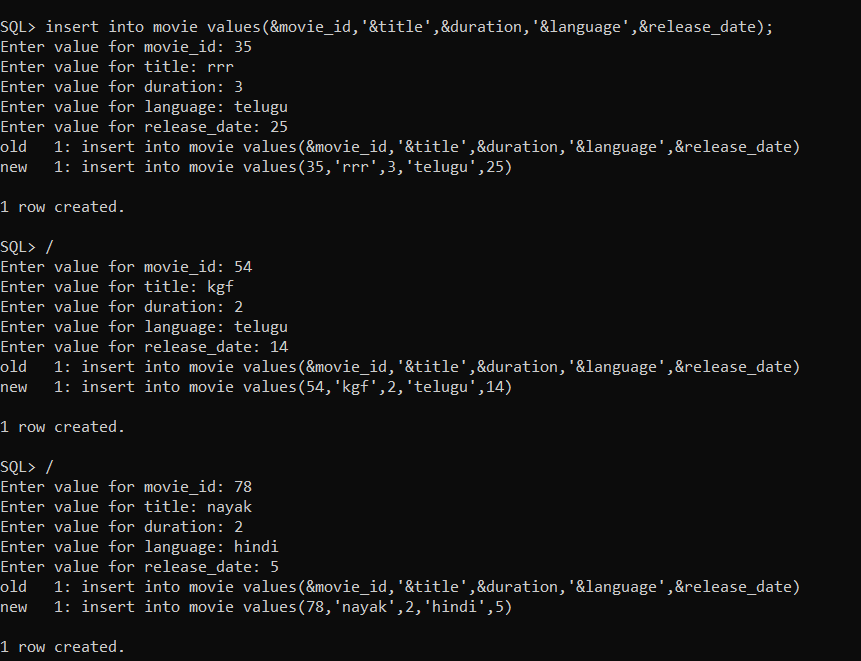
**Table Created in SQL for above mentioned purpose is as:**

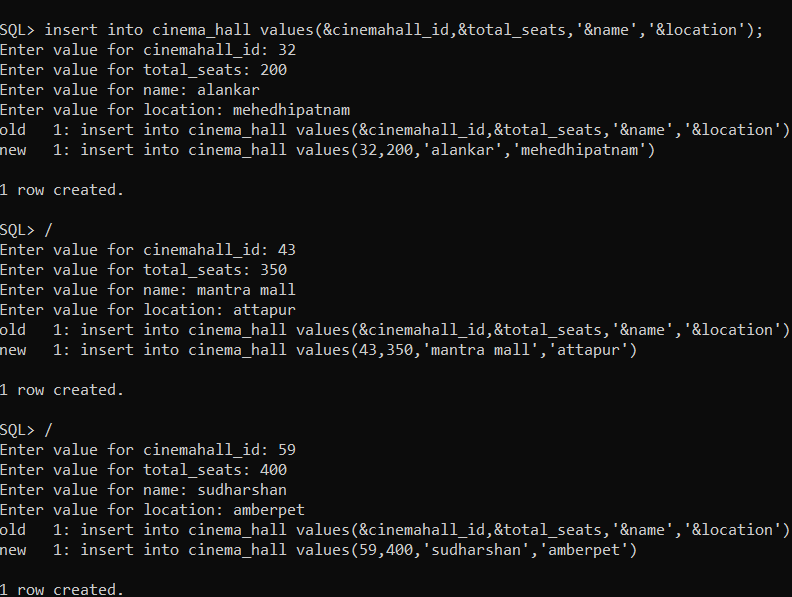


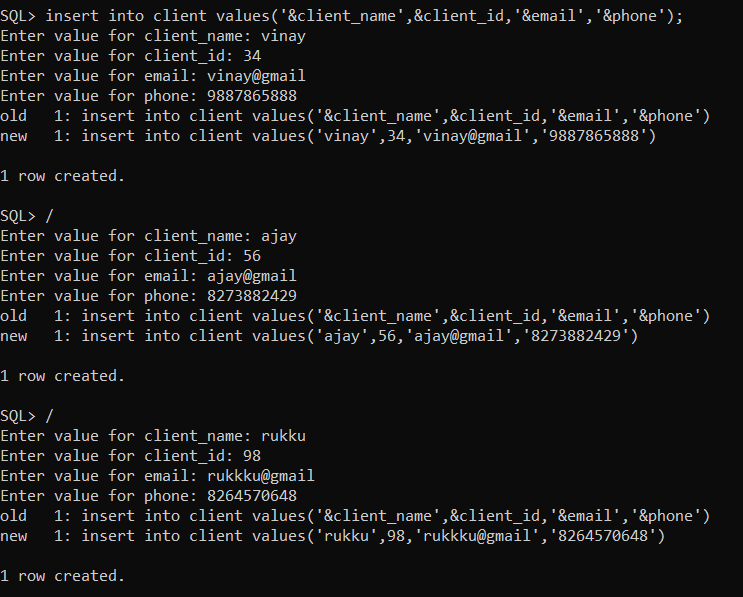


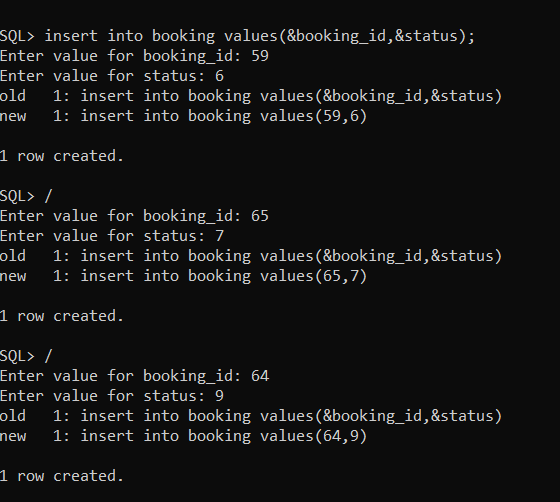




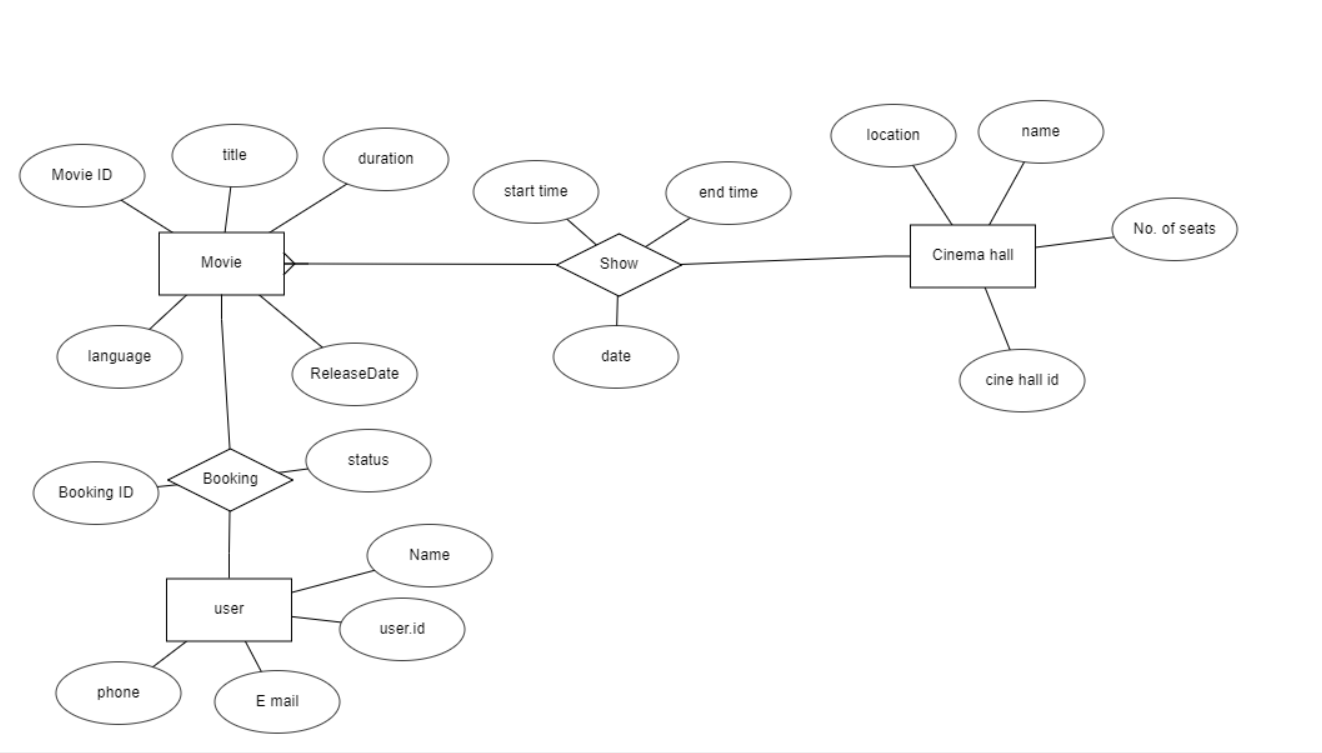




-



**ER DIAGRAM :**



**Implementation:**

**Program:**

package cinematicketbookingsystem;

import javax.swing.JOptionpane

public class LoginBooking extends javax.swing.JFrame {

/\*\*

\* Creates new form NewJFrame

\*/

public LoginBooking() {

initComponents();

setSize(500,300);

setLocationRelativeTo(null);

//setLocation(100,100);

setResizable(false);

setTitle("Login");

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents

private void initComponents() {

jLabel1 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

jButton1 = new javax.swing.JButton();

jTextField1 = new javax.swing.JTextField();

jButton2 = new javax.swing.JButton();

jLabel3 = new javax.swing.JLabel();

jPasswordField1 = new javax.swing.JPasswordField();

jLabel4 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

jLabel1.setFont(new java.awt.Font("Tahoma", 1, 14)); // NOI18N

jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);

jLabel1.setText("Username");

jLabel2.setFont(new java.awt.Font("Tahoma", 1, 14)); // NOI18N

jLabel2.setText("Password");

jButton1.setFont(new java.awt.Font("Tahoma", 1, 14)); // NOI18N

jButton1.setForeground(new java.awt.Color(255, 0, 0));

jButton1.setText("Login");

jButton1.setBorder(null);

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

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

jButton1ActionPerformed(evt);

}

});

jTextField1.setFont(new java.awt.Font("Tahoma", 1, 14)); // NOI18N

jTextField1.setBorder(null);

jButton2.setFont(new java.awt.Font("Tahoma", 1, 14)); // NOI18N

jButton2.setForeground(new java.awt.Color(255, 0, 0));

jButton2.setText("Reset");

jButton2.setBorder(null);

jButton2.addActionListener(new java.awt.event.ActionListener() {

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

jButton2ActionPerformed(evt);

}

});

jLabel3.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

jLabel3.setText("Admin Login");

jLabel4.setIcon(new javax.swing.ImageIcon("C:\\Users\\Ravikumar\\Desktop\\Ticket\\Dolby.gif")); // NOI18N

jLabel4.setText("jLabel4");

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(116, 116, 116)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(39, 39, 39)

.addComponent(jButton1, javax.swing.GroupLayout.PREFERRED\_SIZE, 54, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(27, 27, 27)

.addComponent(jButton2, javax.swing.GroupLayout.PREFERRED\_SIZE, 61, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(jLabel2)

.addComponent(jLabel1))

.addGap(51, 51, 51)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(jTextField1)

.addComponent(jPasswordField1, javax.swing.GroupLayout.PREFERRED\_SIZE, 104, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addGroup(layout.createSequentialGroup()

.addGap(54, 54, 54)

.addComponent(jLabel3)))

.addContainerGap(134, Short.MAX\_VALUE))

.addComponent(jLabel4, javax.swing.GroupLayout.PREFERRED\_SIZE, 0, Short.MAX\_VALUE)

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jLabel4, javax.swing.GroupLayout.PREFERRED\_SIZE, 79, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jLabel3)

.addGap(26, 26, 26)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel1)

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(24, 24, 24)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel2)

.addComponent(jPasswordField1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(27, 27, 27)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jButton1, javax.swing.GroupLayout.PREFERRED\_SIZE, 30, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jButton2, javax.swing.GroupLayout.PREFERRED\_SIZE, 30, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addContainerGap(38, Short.MAX\_VALUE))

);

pack();

}// </editor-fold>//GEN-END:initComponents

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jButton1ActionPerformed

// TODO add your handling code here:

String username = "admin";

String password = "123";

if(username.equals("admin")&& password.equals("123"))

{

JOptionPane.showMessageDialog(null, "Login success");

booking b1 = new booking();

b1.setVisible(true);

this.setVisible(false);

}

else

{

JOptionPane.showMessageDialog(null, "Login Failed");

}

}//GEN-LAST:event\_jButton1ActionPerformed

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jButton2ActionPerformed

// TODO add your handling code here:

jTextField1.setText(null);

jPasswordField1.setText(null);

}//GEN-LAST:event\_jButton2ActionPerformed

**/\*\***

**\* @param args the command line arguments**

**\*/**

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

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(LoginBooking.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

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

} catch (IllegalAccessException ex) {

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

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

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

}

//</editor-fold>

//</editor-fold>

/\* Create and display the form \*/

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

public void run() {

new LoginBooking().setVisible(true);

}

});

}

// Variables declaration - do not modify//GEN-BEGIN:variables

private javax.swing.JButton jButton1;

private javax.swing.JButton jButton2;

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;

// End of variables declaration//GEN-END:variables

}

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package cinematicketbookingsystem;

/\*\*

\*

\* @author Ravikumar

\*/

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

import javax.swing.ImageIcon;

//import java.util.ArrayList;

import java.io.\*;

public class booking extends JFrame implements ActionListener

{

JTextField seat, totalcost;

Choice type, block, time; // Choices for the GUI

JLabel typelbl, blocklbl, timelbl, lblbl, mblbl, rblbl, inputseatlbl, totallbl, title;

JButton submitbut, showbut, paybut;

String selectedtime, total, selectedseat, selectedblock, tickettype;

int timechoice, convertedseat1, convertedseat2, totalprice;

int totaltickets = 0, currenti;

Seat leftside1[][] = new Seat [4] [4];

Seat midside1[][] = new Seat [4] [4];

Seat rightside1[][] = new Seat [4] [4]; // Each one of these creates an array for each block

Seat leftside3[][] = new Seat [4] [4];

Seat midside3[][] = new Seat [4] [4];

Seat rightside3[][] = new Seat [4] [4];

Seat leftside5[][] = new Seat [4] [4];

Seat midside5[][] = new Seat [4] [4];

Seat rightside5[][] = new Seat [4] [4];

Seat leftside7[][] = new Seat [4] [4];

Seat midside7[][] = new Seat [4] [4];

Seat rightside7[][] = new Seat [4] [4];

Seat leftside9[][] = new Seat [4] [4];

Seat midside9[][] = new Seat [4] [4];

Seat rightside9[][] = new Seat [4] [4];

Ticket tickets[] = new Ticket[30]; //Array for storing the tickets booked

public booking()

{

setSize(800,500);

setLocation(200,200);

setResizable(false);

setTitle("Ticket Booking");

setLayout(new FlowLayout());

ImageIcon imgtitle = new ImageIcon("C:\\Users\\Ravikumar\\Documents\\NetBeansProjects\\CinemaTicketBookingSystem\\img\\title1.gif");

title = new JLabel(imgtitle);

typelbl = new JLabel("Ticket Type: ");

typelbl.setFont(new Font("Verdana", Font.BOLD, 22));

type = new Choice();

type.add("Adult");

type.add("Child");

type.add("Senior Citizen");

blocklbl = new JLabel("Block:");

blocklbl.setFont(new Font("Verdana", Font.BOLD, 22));

block = new Choice();

block.add("Left Block");

block.add("Middle Block");

block.add("Right Block");

timelbl = new JLabel("Time: ");

timelbl.setFont(new Font("Verdana", Font.BOLD, 22));

time = new Choice();

time.add("13:00");

time.add("15:00");

time.add("17:00");

time.add("19:00");

time.add("21:00");

paybut = new JButton("Pay");

paybut.setBackground(Color.white);

ImageIcon lbimg = new ImageIcon("C:\\Users\\Ravikumar\\Documents\\NetBeansProjects\\CinemaTicketBookingSystem\\img\\lblogo.gif");

lblbl = new JLabel (lbimg);

ImageIcon mbimg = new ImageIcon("C:\\Users\\Ravikumar\\Documents\\NetBeansProjects\\CinemaTicketBookingSystem\\img\\mblogo1.gif");

mblbl = new JLabel (mbimg);

ImageIcon rbimg = new ImageIcon("C:\\Users\\Ravikumar\\Documents\\NetBeansProjects\\CinemaTicketBookingSystem\\img\\rblogo.gif");

rblbl = new JLabel (rbimg);

inputseatlbl = new JLabel ("Enter Seat Num: ");

inputseatlbl.setFont(new Font("Verdana", Font.BOLD, 22));

seat = new JTextField(2);

seat.setFont(new Font("Verdana", Font.BOLD, 20));

seat.setHorizontalAlignment(seat.CENTER);

totallbl = new JLabel ("Total Price: ");

totallbl.setFont(new Font("Verdana", Font.BOLD, 22));

totalcost = new JTextField(7);

totalcost.setFont(new Font("Verdana", Font.BOLD, 22));

totalcost.setHorizontalAlignment(totalcost.CENTER);

totalcost.setEditable(false);

submitbut = new JButton("Submit");

submitbut.setBackground(Color.white);

showbut = new JButton("Show");

showbut.setBackground(Color.white);

/\* The following delcarations

initialise the variables for

the positions of the blocks \*/

int xleft = 0;

int yleft = 0;

int xmid = 0;

int ymid = 0;

int xright = 0;

int yright = 0;

xleft = 50;

yleft = 310; //Initial positions

int totalleftside1 = 0; //'Total' is the total and is used to label the seats

for (int i = 0; i < leftside1.length; i++)

{

for (int j = 0; j < 4; j++)

{

leftside1[i][j]=new Seat(totalleftside1, 0, xleft, yleft);

xleft += 50; //Moving the next seat along the x position by 50

totalleftside1 ++; //Adding one to the total for the next seat

}

xleft = 50; //Moving back to the left for the next row

yleft += 30; //Moving down for the next row

}

xmid = 300;

ymid = 310; //Initial positions

int totalmidside1 = 0; //'Total' is the total and is used to label the seats

for (int i = 0; i < midside1.length; i++)

{

for (int j = 0; j < 4; j++)

{

midside1[i][j]=new Seat(totalmidside1, 0, xmid, ymid);

xmid += 50; //Moving the next seat along the x position by 50

totalmidside1 ++; //Adding one to the total for the next seat

}

xmid = 300; //Moving back to the left for the next row

ymid += 30; //Moving down for the next row

}

xright = 550;

yright = 310; //Initial positions

int totalrightside1 = 0; //'Total' is the total and is used to label the seats

for (int i = 0; i < rightside1.length; i++)

{

for (int j = 0; j < 4; j++)

{

rightside1[i][j]=new Seat(totalrightside1, 0, xright, yright);

xright += 50; //Moving the next seat along the x position by 50

totalrightside1 ++; //Adding one to the total for the next seat

}

xright = 550; //Moving back to the left for the next row

yright += 30; //Moving down for the next row

}

/\* The next lines of code perform the same processes as the ones before such as positions for each block \*/

xleft = 50;

yleft = 310;

int totalleftside3 = 0;

for (int i = 0; i < leftside3.length; i++)

{

for (int j = 0; j < 4; j++)

{

leftside3[i][j]=new Seat(totalleftside3, 0, xleft, yleft);

xleft += 50;

totalleftside3 ++;

}

xleft = 50;

yleft += 30;

}

xmid = 300;

ymid = 310;

int totalmidside3 = 0;

for (int i = 0; i < midside3.length; i++)

{

for (int j = 0; j < 4; j++)

{

midside3[i][j]=new Seat(totalmidside3, 0, xmid, ymid);

xmid += 50;

totalmidside3 ++;

}

xmid = 300;

ymid += 30;

}

xright = 550;

yright = 310;

int totalrightside3 = 0;

for (int i = 0; i < rightside3.length; i++)

{

for (int j = 0; j < 4; j++)

{

rightside3[i][j]=new Seat(totalrightside3, 0, xright, yright);

xright += 50;

totalrightside3 ++;

}

xright = 550;

yright += 30;

}

xleft = 50;

yleft = 310;

int totalleftside5 = 0;

for (int i = 0; i < leftside5.length; i++)

{

for (int j = 0; j < 4; j++)

{

leftside5[i][j]=new Seat(totalleftside5, 0, xleft, yleft);

xleft += 50;

totalleftside5 ++;

}

xleft = 50;

yleft += 30;

}

xmid = 300;

ymid = 310;

int totalmidside5 = 0;

for (int i = 0; i < midside5.length; i++)

{

for (int j = 0; j < 4; j++)

{

midside5[i][j]=new Seat(totalmidside5, 0, xmid, ymid);

xmid += 50;

totalmidside5 ++;

}

xmid = 300;

ymid += 30;

}

xright = 550;

yright = 310;

int totalrightside5 = 0;

for (int i = 0; i < rightside5.length; i++)

{

for (int j = 0; j < 4; j++)

{

rightside5[i][j]=new Seat(totalrightside5, 0, xright, yright);

xright += 50;

totalrightside5 ++;

}

xright = 550;

yright += 30;

}

xleft = 50;

yleft = 310;

int totalleftside7 = 0;

for (int i = 0; i < leftside7.length; i++)

{

for (int j = 0; j < 4; j++)

{

leftside7[i][j]=new Seat(totalleftside7, 0, xleft, yleft);

xleft += 50;

totalleftside7 ++;

}

xleft = 50;

yleft += 30;

}

xmid = 300;

ymid = 310;

int totalmidside7 = 0;

for (int i = 0; i < midside7.length; i++)

{

for (int j = 0; j < 4; j++)

{

midside7[i][j]=new Seat(totalmidside7, 0, xmid, ymid);

xmid += 50;

totalmidside7 ++;

}

xmid = 300;

ymid += 30;

}

xright = 550;

yright = 310;

int totalrightside7 = 0;

for (int i = 0; i < rightside7.length; i++)

{

for (int j = 0; j < 4; j++)

{

rightside7[i][j]=new Seat(totalrightside7, 0, xright, yright);

xright += 50;

totalrightside7 ++;

}

xright = 550;

yright += 30;

}

xleft = 50;

yleft = 310;

int totalleftside9 = 0;

for (int i = 0; i < leftside9.length; i++)

{

for (int j = 0; j < 4; j++)

{

leftside9[i][j]=new Seat(totalleftside9, 0, xleft, yleft);

xleft += 50;

totalleftside9 ++;

}

xleft = 50;

yleft += 30;

}

xmid = 300;

ymid = 310;

int totalmidside9 = 0;

for (int i = 0; i < midside9.length; i++)

{

for (int j = 0; j < 4; j++)

{

midside9[i][j]=new Seat(totalmidside9, 0, xmid, ymid);

xmid += 50;

totalmidside9 ++;

}

xmid = 300;

ymid += 30;

}

xright = 550;

yright = 310;

int totalrightside9 = 0;

for (int i = 0; i < rightside9.length; i++)

{

for (int j = 0; j < 4; j++)

{

rightside9[i][j]=new Seat(totalrightside9, 0, xright, yright);

xright += 50;

totalrightside9 ++;

}

xright = 550;

yright += 30;

}

/\* Adding the labels and text boxes to the screen \*/

getContentPane().add(title);

getContentPane().add(typelbl);

getContentPane().add(type);

getContentPane().add(blocklbl);

getContentPane().add(block);

getContentPane().add(timelbl);

getContentPane().add(time);

getContentPane().add(showbut);

getContentPane().add(inputseatlbl);

getContentPane().add(seat);

getContentPane().add(submitbut);

getContentPane().add(totallbl);

getContentPane().add(totalcost);

getContentPane().add(paybut);

getContentPane().add(lblbl);

getContentPane().add(mblbl);

getContentPane().add(rblbl);

setVisible(true);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

Color mycolor = new Color(255,255,255);

getContentPane().setBackground(mycolor);

submitbut.addActionListener(this);

showbut.addActionListener(this); //Action listener for each individual button

paybut.addActionListener(this);

}

public void paint(Graphics graf)

{

super.paint(graf);

switch(timechoice)

{

case 1:

for (int i = 0; i < leftside1.length; i++) //Looping throw the leftside at 1 block - gives initial size

{

for (int j = 0; j < 4; j++) //Looping through each row fully

{

leftside1[i][j].display(graf); //Displaying the left side @ 1

midside1[i][j].display(graf); //Displaying the mid side @ 1

rightside1[i][j].display(graf); //Displaying the right side @ 1

}

}

break;

case 3:

for (int i = 0; i < leftside3.length; i++) //Looping through the leftside at 3 block - gives initial size

{

for (int j = 0; j < 4; j++) //Looping through each row fully

{

leftside3[i][j].display(graf); //Displaying the left side @ 3

midside3[i][j].display(graf); //Displaying the mid side @ 3

rightside3[i][j].display(graf); //Displaying the right side @ 3

}

}

break;

case 5:

for (int i = 0; i < leftside5.length; i++) //Looping through the leftside at 5 block - gives initial size

{

for (int j = 0; j < 4; j++) //Looping through each row fully

{

leftside5[i][j].display(graf); //Displaying the left side @ 5

midside5[i][j].display(graf); //Displaying the mid side @ 5

rightside5[i][j].display(graf); //Displaying the right side @ 5

}

}

break;

case 7:

for (int i = 0; i < leftside7.length; i++) //Looping through the leftside at 7 block - gives initial size

{

for (int j = 0; j < 4; j++) //Looping through each row fully

{

leftside7[i][j].display(graf); //Displaying the left side @ 7

midside7[i][j].display(graf); //Displaying the mid side @ 7

rightside7[i][j].display(graf); //Displaying the right side @ 7

}

}

break;

case 9:

for (int i = 0; i < leftside9.length; i++) //Looping through the leftside at 7 block - gives initial size

{

for (int j = 0; j < 4; j++) //Looping through each row fully

{

leftside9[i][j].display(graf); //Displaying the left side @ 7

midside9[i][j].display(graf); //Displaying the mid side @ 7

rightside9[i][j].display(graf); //Displaying the right side @ 7

}

}

break;

default:

/\* Used as what appears when the program starts \*/

for (int i = 0; i < leftside1.length; i++) //Looping through the leftside at 1 block - gives initial size

{

for (int j = 0; j < 4; j++) //Looping through each row fully

{

leftside1[i][j].display(graf); //Displaying the left side @ 1

midside1[i][j].display(graf); //Displaying the mid side @ 1

rightside1[i][j].display(graf); //Displaying the right side @ 1

}

}

break;

}

}

public void actionPerformed(ActionEvent ev)

{

selectedblock = block.getSelectedItem(); //The values neeeded to book seats and check for availability taken from text boxes

selectedtime = time.getSelectedItem();

tickettype = type.getSelectedItem();

if (ev.getSource() == submitbut)

{

selectedseat = seat.getText();

int selectseat = Integer.parseInt(selectedseat); //Getting the seat number and converting it to an integer

/\* The following set of if statements checks for seat number and converts it

to an X, Y position which can be used for booking \*/

if (selectseat == 0)

{convertedseat1 = 0;

convertedseat2 = 0;}

if (selectseat == 1)

{convertedseat1 = 0;

convertedseat2 = 1;}

if (selectseat == 2)

{convertedseat1 = 0;

convertedseat2 = 2;}

if (selectseat == 3)

{convertedseat1 = 0;

convertedseat2 = 3;}

if (selectseat == 4)

{convertedseat1 = 1;

convertedseat2 = 0;}

if (selectseat == 5)

{convertedseat1 = 1;

convertedseat2 = 1;}

if (selectseat == 6)

{convertedseat1 = 1;

convertedseat2 = 2;}

if (selectseat == 7)

{convertedseat1 = 1;

convertedseat2 = 3;}

if (selectseat == 8)

{convertedseat1 = 2;

convertedseat2 = 0;}

if (selectseat == 9)

{convertedseat1 =2;

convertedseat2 = 1;}

if (selectseat == 10)

{convertedseat1 = 2;

convertedseat2 = 2;}

if (selectseat == 11)

{convertedseat1 = 2;

convertedseat2 = 3;}

if (selectseat == 12)

{convertedseat1 = 3;

convertedseat2 = 0;}

if (selectseat == 13)

{convertedseat1 = 3;

convertedseat2 = 1;}

if (selectseat == 14)

{convertedseat1 = 3;

convertedseat2 = 2;}

if (selectseat == 15)

{convertedseat1 = 3;

convertedseat2 = 3;}

if (selectedtime == "13:00") //If they are trying to book a ticket a 1 o'clock

{

timechoice = 1;

repaint();

if (selectedblock == "Left Block") // If they are trying to book a seat in the left block

{

int iftaken = leftside1[convertedseat1][convertedseat2].isTaken(); //Uses the istaken() method to see if it is taken

if (iftaken == 1)

{

JOptionPane.showMessageDialog(null, "This seat is taken and cannot be booked"); //Seat is taken and outputs an error message

}

else if (iftaken == 0)

{

leftside1[convertedseat1][convertedseat2].setSeat(); //Sets the seat to booked

repaint(); //Re-displays it

tickets[totaltickets] = new Ticket(tickettype, selectseat, selectedblock, selectedtime); //Creates a new ticket objects in the tickets array

int tempprice = tickets[totaltickets].getSeatPrice(); //Gets the seat price using the getSeatPrice() method

totalprice = totalprice + tempprice;

totalcost.setText("INR "+ Integer.toString(totalprice)); //Updates and adds to the text box

totaltickets += 1; //Updates the total amount of tickets booked

seat.setText("");

}

}

if (selectedblock == "Middle Block") // If they are trying to book a seat in the middle block

{

int iftaken = midside1[convertedseat1][convertedseat2].isTaken(); //Uses the istaken() method to see if it is taken

if (iftaken == 1)

{

JOptionPane.showMessageDialog(null, "This seat is taken and cannot be booked"); //Seat is taken and outputs an error message

}

else if (iftaken == 0)

{

midside1[convertedseat1][convertedseat2].setSeat(); //Sets the seat to booked

repaint();

tickets[totaltickets] = new Ticket(tickettype, selectseat, selectedblock, selectedtime); //Creates a new ticket objects in the tickets array

int tempprice = tickets[totaltickets].getSeatPrice(); //Gets the seat price using the getSeatPrice() method

totalprice = totalprice + tempprice;

totalcost.setText("INR "+ Integer.toString(totalprice)); //Updates and adds to the text box

totaltickets += 1; //Updates the total amount of tickets booked

seat.setText("");

}

}

if (selectedblock == "Right Block") // If they are trying to book a seat in the right block

{

int iftaken = rightside1[convertedseat1][convertedseat2].isTaken(); //Uses the istaken() method to see if it is taken

if (iftaken == 1)

{

JOptionPane.showMessageDialog(null, "This seat is taken and cannot be booked"); //Seat is taken and outputs an error message

}

else if (iftaken == 0)

{

rightside1[convertedseat1][convertedseat2].setSeat(); //Sets the seat to booked

repaint();

tickets[totaltickets] = new Ticket(tickettype, selectseat, selectedblock, selectedtime); //Creates a new ticket objects in the tickets array

int tempprice = tickets[totaltickets].getSeatPrice(); //Gets the seat price using the getSeatPrice() method

totalprice = totalprice + tempprice;

totalcost.setText("INR "+ Integer.toString(totalprice)); //Updates and adds to the text box

totaltickets += 1; //Updates the total amount of tickets booked

seat.setText("");

}

}

}

/\* Code is replicated for each of the times the user wants to book\*/

if (selectedtime == "15:00")

{

timechoice = 3;

repaint();

if (selectedblock == "Left Block")

{

int iftaken = leftside3[convertedseat1][convertedseat2].isTaken();

if (iftaken == 1)

{

JOptionPane.showMessageDialog(null, "This seat is taken and cannot be booked");

}

else if (iftaken == 0)

{

leftside3[convertedseat1][convertedseat2].setSeat();

repaint();

tickets[totaltickets] = new Ticket(tickettype, selectseat, selectedblock, selectedtime);

int tempprice = tickets[totaltickets].getSeatPrice();

totalprice = totalprice + tempprice;

totalcost.setText("INR "+ Integer.toString(totalprice));

totaltickets += 1;

seat.setText("");

}

}

if (selectedblock == "Middle Block")

{

int iftaken = midside3[convertedseat1][convertedseat2].isTaken();

if (iftaken == 1)

{

JOptionPane.showMessageDialog(null, "This seat is taken and cannot be booked");

}

else if (iftaken == 0)

{

midside3[convertedseat1][convertedseat2].setSeat();

repaint();

tickets[totaltickets] = new Ticket(tickettype, selectseat, selectedblock, selectedtime);

int tempprice = tickets[totaltickets].getSeatPrice();

totalprice = totalprice + tempprice;

totalcost.setText("INR "+ Integer.toString(totalprice));

totaltickets += 1;

seat.setText("");

}

}

if (selectedblock == "Right Block")

{

int iftaken = rightside3[convertedseat1][convertedseat2].isTaken();

if (iftaken == 1)

{

JOptionPane.showMessageDialog(null, "This seat is taken and cannot be booked");

}

else if (iftaken == 0)

{

rightside3[convertedseat1][convertedseat2].setSeat();

repaint();

tickets[totaltickets] = new Ticket(tickettype, selectseat, selectedblock, selectedtime);

int tempprice = tickets[totaltickets].getSeatPrice();

totalprice = totalprice + tempprice;

totalcost.setText("INR "+ Integer.toString(totalprice));

totaltickets += 1;

seat.setText("");

}

}

}

if (selectedtime == "17:00")

{

timechoice = 5;

repaint();

if (selectedblock == "Left Block")

{

int iftaken = leftside5[convertedseat1][convertedseat2].isTaken();

if (iftaken == 1)

{

JOptionPane.showMessageDialog(null, "This seat is taken and cannot be booked");

}

else if (iftaken == 0)

{

leftside5[convertedseat1][convertedseat2].setSeat();

repaint();

tickets[totaltickets] = new Ticket(tickettype, selectseat, selectedblock, selectedtime);

int tempprice = tickets[totaltickets].getSeatPrice();

totalprice = totalprice + tempprice;

totalcost.setText("INR "+ Integer.toString(totalprice));

totaltickets += 1;

seat.setText("");

}

}

if (selectedblock == "Middle Block")

{

int iftaken = midside5[convertedseat1][convertedseat2].isTaken();

if (iftaken == 1)

{

JOptionPane.showMessageDialog(null, "This seat is taken and cannot be booked");

}

else if (iftaken == 0)

{

midside5[convertedseat1][convertedseat2].setSeat();

repaint();

tickets[totaltickets] = new Ticket(tickettype, selectseat, selectedblock, selectedtime);

int tempprice = tickets[totaltickets].getSeatPrice();

totalprice = totalprice + tempprice;

totalcost.setText("INR "+ Integer.toString(totalprice));

totaltickets += 1;

seat.setText("");

}

}

if (selectedblock == "Right Block")

{

int iftaken = rightside5[convertedseat1][convertedseat2].isTaken();

if (iftaken == 1)

{

JOptionPane.showMessageDialog(null, "This seat is taken and cannot be booked");

}

else if (iftaken == 0)

{

rightside5[convertedseat1][convertedseat2].setSeat();

repaint();

tickets[totaltickets] = new Ticket(tickettype, selectseat, selectedblock, selectedtime);

int tempprice = tickets[totaltickets].getSeatPrice();

totalprice = totalprice + tempprice;

totalcost.setText("INR "+ Integer.toString(totalprice));

totaltickets += 1;

seat.setText("");

}

}

}

if (selectedtime == "19:00")

{

timechoice = 7;

repaint();

if (selectedblock == "Left Block")

{

int iftaken = leftside7[convertedseat1][convertedseat2].isTaken();

if (iftaken == 1)

{

JOptionPane.showMessageDialog(null, "This seat is taken and cannot be booked");

}

else if (iftaken == 0)

{

leftside7[convertedseat1][convertedseat2].setSeat();

repaint();

tickets[totaltickets] = new Ticket(tickettype, selectseat, selectedblock, selectedtime);

int tempprice = tickets[totaltickets].getSeatPrice();

totalprice = totalprice + tempprice;

totalcost.setText("INR "+ Integer.toString(totalprice));

totaltickets += 1;

seat.setText("");

}

}

if (selectedblock == "Middle Block")

{

int iftaken = midside7[convertedseat1][convertedseat2].isTaken();

if (iftaken == 1)

{

JOptionPane.showMessageDialog(null, "This seat is taken and cannot be booked");

}

else if (iftaken == 0)

{

midside7[convertedseat1][convertedseat2].setSeat();

repaint();

tickets[totaltickets] = new Ticket(tickettype, selectseat, selectedblock, selectedtime);

int tempprice = tickets[totaltickets].getSeatPrice();

totalprice = totalprice + tempprice;

totalcost.setText("INR "+ Integer.toString(totalprice));

totaltickets += 1;

seat.setText("");

}

}

if (selectedblock == "Right Block")

{

int iftaken = rightside7[convertedseat1][convertedseat2].isTaken();

if (iftaken == 1)

{

JOptionPane.showMessageDialog(null, "This seat is taken and cannot be booked");

}

else if (iftaken == 0)

{

rightside7[convertedseat1][convertedseat2].setSeat();

repaint();

tickets[totaltickets] = new Ticket(tickettype, selectseat, selectedblock, selectedtime);

int tempprice = tickets[totaltickets].getSeatPrice();

totalprice = totalprice + tempprice;

totalcost.setText("INR "+ Integer.toString(totalprice));

totaltickets += 1;

seat.setText("");

}

}

}

if (selectedtime == "21:00")

{

timechoice = 9;

repaint();

if (selectedblock == "Left Block")

{

int iftaken = leftside9[convertedseat1][convertedseat2].isTaken();

if (iftaken == 1)

{

JOptionPane.showMessageDialog(null, "This seat is taken and cannot be booked");

}

else if (iftaken == 0)

{

leftside9[convertedseat1][convertedseat2].setSeat();

repaint();

tickets[totaltickets] = new Ticket(tickettype, selectseat, selectedblock, selectedtime);

int tempprice = tickets[totaltickets].getSeatPrice();

totalprice = totalprice + tempprice;

totalcost.setText("INR "+ Integer.toString(totalprice));

totaltickets += 1;

seat.setText("");

}

}

if (selectedblock == "Middle Block")

{

int iftaken = midside9[convertedseat1][convertedseat2].isTaken();

if (iftaken == 1)

{

JOptionPane.showMessageDialog(null, "This seat is taken and cannot be booked");

}

else if (iftaken == 0)

{

midside9[convertedseat1][convertedseat2].setSeat();

repaint();

tickets[totaltickets] = new Ticket(tickettype, selectseat, selectedblock, selectedtime);

int tempprice = tickets[totaltickets].getSeatPrice();

totalprice = totalprice + tempprice;

totalcost.setText("INR "+ Integer.toString(totalprice));

totaltickets += 1;

seat.setText("");

}

}

if (selectedblock == "Right Block")

{

int iftaken = rightside9[convertedseat1][convertedseat2].isTaken();

if (iftaken == 1)

{

JOptionPane.showMessageDialog(null, "This seat is taken and cannot be booked");

}

else if (iftaken == 0)

{

rightside9[convertedseat1][convertedseat2].setSeat();

repaint();

tickets[totaltickets] = new Ticket(tickettype, selectseat, selectedblock, selectedtime);

int tempprice = tickets[totaltickets].getSeatPrice();

totalprice = totalprice + tempprice;

totalcost.setText("INR"+ Integer.toString(totalprice));

totaltickets += 1;

seat.setText("");

}

}

}

}

if (ev.getSource() == showbut)

{

/\* If the user has clicked the show button it sets a variable and then repaints the desired time \*/

if (selectedtime == "13:00")

{

timechoice = 1;

repaint();

}

if (selectedtime == "15:00")

{

timechoice = 3;

repaint();

}

if (selectedtime == "17:00")

{

timechoice = 5;

repaint();

}

if (selectedtime == "19:00")

{

timechoice = 7;

repaint();

}

if(selectedtime == "21:00")

{

timechoice = 9;

repaint();

}

}

if (ev.getSource() == paybut) //Pay button to generate tickets

{

try{

FileWriter stream= new FileWriter("tickets.txt"); //Opening the tickets.txt file

BufferedWriter output = new BufferedWriter(stream);

for (int i = 0; i < totaltickets; i++) //Looping through the number of tickets booked

{

/\* Getting the variables needed from methods related to the object \*/

int ticketnum = tickets[i].getSeatNum();

int ticketprice = tickets[i].getSeatPrice();

String tickettype = tickets[i].getType();

String tickettime = tickets[i].getTime();

String ticketblock = tickets[i].getBlock();

String newline = System.getProperty("line.separator");

output.write(newline);

output.write("Ticket Details");

output.write(newline);

output.write(newline);

output.write("Ticket Number: " + ticketnum); //Writing ticketnum

output.write(newline);

output.write("Ticket Price: " + ticketprice);

output.write(newline);

output.write("Ticket Type: " + tickettype);

output.write(newline);

output.write("Ticket Time: " + tickettime);

output.write(newline);

output.write("Ticket Block: " + ticketblock);

output.write(newline);

output.write("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

output.write(newline);

}

output.close();

}catch (Exception e){//Catch exception if any

System.err.println("Error: " + e.getMessage());}

JOptionPane.showMessageDialog(null, "Success!! Ticket(s) are booked."); //Success message

for(int i=0; i < tickets.length; i++)

{

tickets[i] = null; //Setting the tickets array values to null so new tickets can be booked

}

totaltickets = 0; //Resets totaltickets to 0

totalprice = 0; //Resets totalprice to 0

totalcost.setText(""); //Sets totalcost to an empty text field

seat.setText(""); //Wipes seat number text field

}

}

public static void main(String args[])

{

new booking();

}

}

class Seat

{

private final int boxheight = 30; //Height of the drawn text box

private final int boxwidth = 50; //Width of the drawn text box

private int seatnumber;

private int seattaken;

private int x;

private int y;

public Seat(int number, int taken, int xstart, int ystart)

{

seatnumber = number; //Setting the seatnumber

seattaken = taken; //Setting whether it is taken

x = xstart; //Display position

y = ystart;

}

public void display(Graphics graf)

{

int xdraw, ydraw;

switch (seattaken)

{

case 0:

xdraw = x + boxwidth;

ydraw = y + boxheight;

graf.drawRect(x,y,boxwidth,boxheight);

graf.drawString(Integer.toString(seatnumber),x+20,y+boxheight\*3/4); //Painting the seat with the seat number printed inside

break;

case 1:

xdraw = x + boxwidth;

ydraw = y + boxheight;

graf.drawRect(x,y,boxwidth,boxheight);

String msg = "N/A";

graf.drawString(msg,x+20,y+boxheight\*3/4); //Painting the seat with N/A printed inside

break;

default:

break;

}

}

public int isTaken()

{

return seattaken;

}

public void setSeat()

{

seattaken = 1;

}

}

class Ticket

{

private int price;

private String tickettype;

private int seatnum;

private String timeticket;

private String blockticket;

public Ticket (String type, int seatnumber, String block, String time)

{

seatnum = seatnumber; //Setting tickets seat number

tickettype = type; //Setting ticket type

blockticket = block; //Setting block

timeticket = time; //Setting time

if (blockticket == "Left Block")

{

if (type == "Adult")

{

price = 500; //Setting adult price

}

if (type == "Senior Citizen")

{

price = 300; //Setting Senior Citizen price

}

if (type == "Child")

{

price = 200; //Setting child price

}

}

if (blockticket == "Middle Block")

{

if (type == "Adult")

{

price = 1000; //Setting adult price

}

if (type == "Senior Citizen")

{

price = 3000; //Setting Senior Citizen price

}

if (type == "Child")

{

price = 500; //Setting child price

}

}

if (blockticket == "Right Block")

{

if (type == "Adult")

{

price = 500; //Setting adult price

}

if (type == "Senior Citizen")

{

price = 300; //Setting Senior Citizen price

}

if (type == "Child")

{

price = 200; //Setting child price

}

}

}

/\* The following get methods are used to print tickets \*/

public int getSeatPrice()

{

return price;

}

public String getTime()

{

return timeticket;

}

public String getBlock()

{

return blockticket;

}

public String getType()

{

return tickettype;

}

public int getSeatNum()

{

return seatnum;

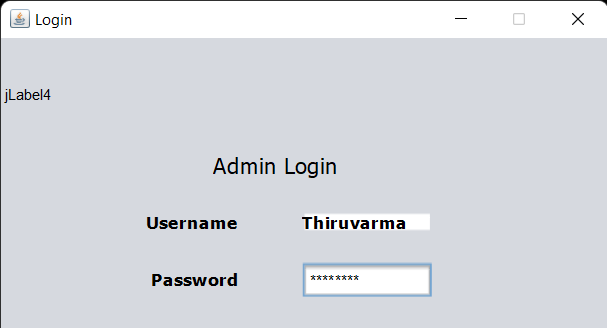
}

}

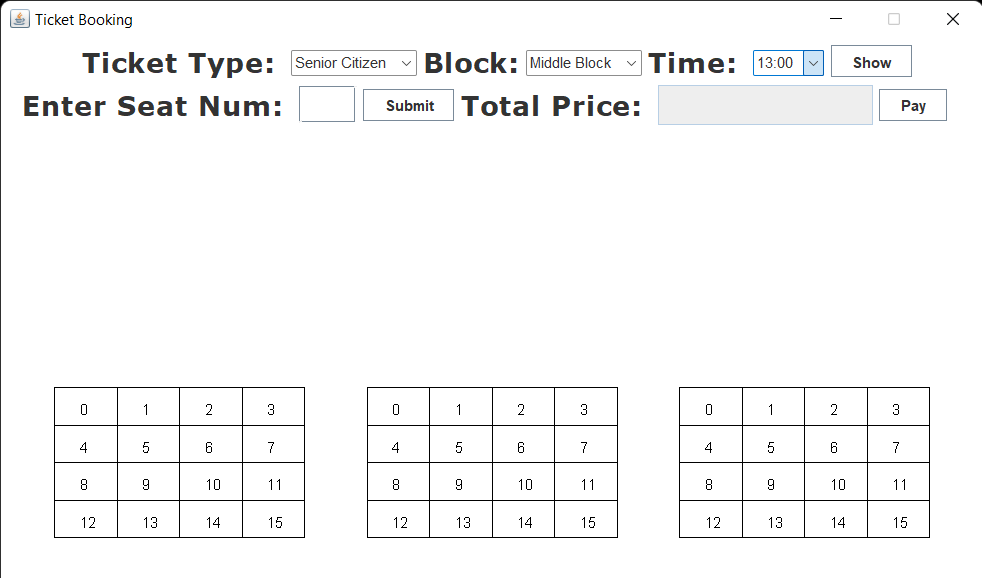
## GitHub links and folder structure

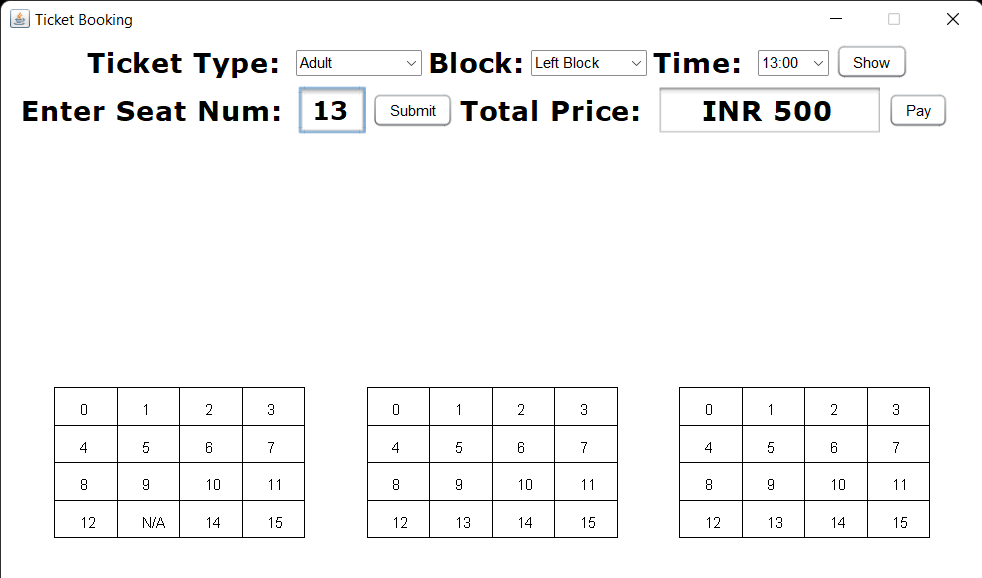
<https://github.com/Thirupathi2707/BOOK-MY-TICKET-.git>

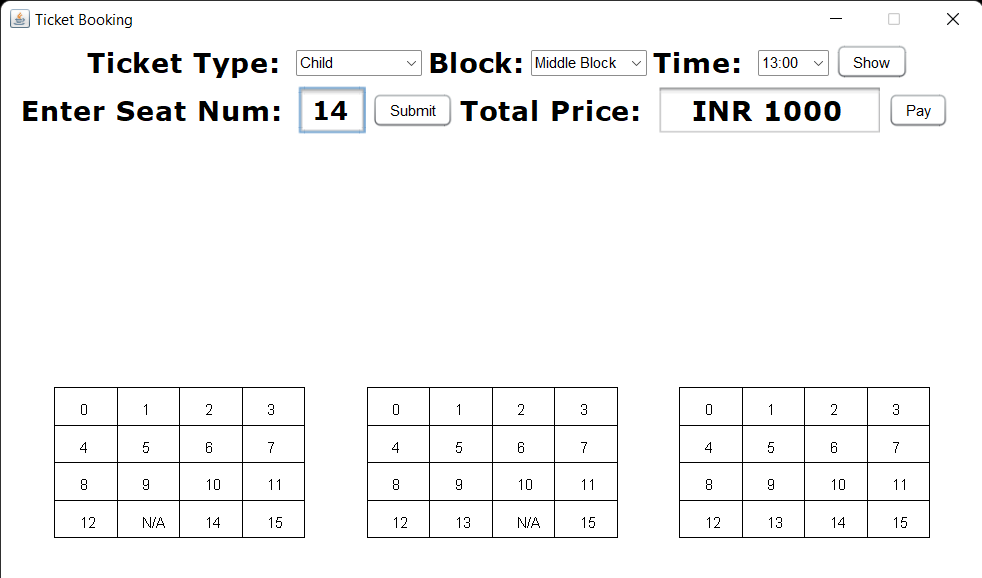
**Testing:**

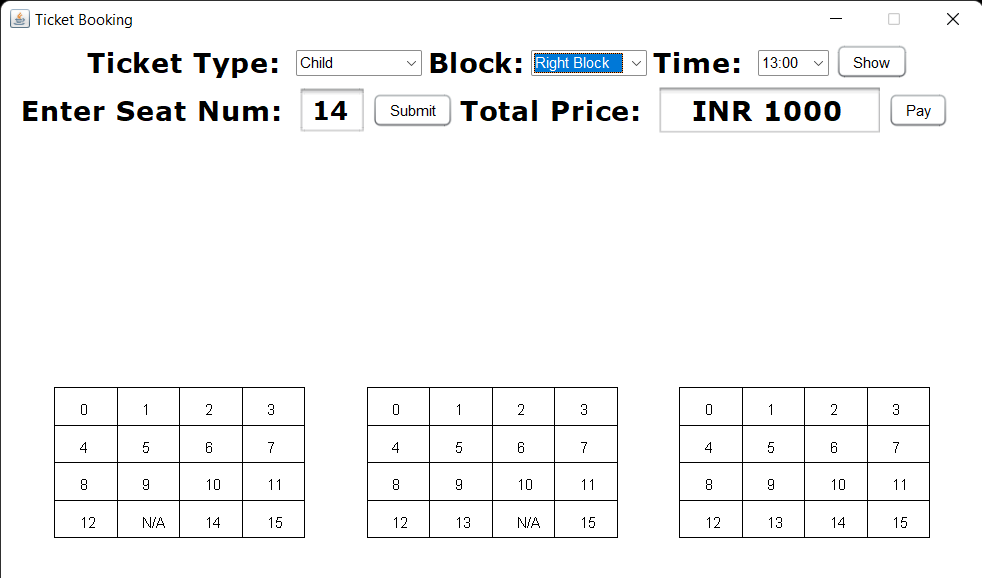


**Java GUI Testing:**

****







**The data entered in the above form is updated in the “Booking” table of the Oracle database 11g as:**

**Results:**

I successfully completed this MINI PROJECT “BOOK MY TICKET”.

## Discussion and Future work

While doing this project I got new ideas I understood how to work on projects. Now to further extend this project I want to create a android app by which I can control my project on my hand and connect to it.

**References:**

* [https://www.academia.edu/36893248/Ramakrishnan\_-](https://www.academia.edu/36893248/Ramakrishnan_-_Database_Management_Systems_3rd_Edition)

[\_Database\_Management\_Systems\_3rd\_Edition](https://www.academia.edu/36893248/Ramakrishnan_-_Database_Management_Systems_3rd_Edition) ▪ <https://docs.oracle.com/javase/7/docs/index.html>

* <https://www.javatpoint.com/dbms-tutorial>
* <http://www.sqlines.com/articles/java/sql_server_jdbc_connection>