**Film booking system**

**19/20 MSc Mobile and Client Side User Experience**

**Project Proposal Form for Assignment**

**Student Name**

**Project Area**

Movie Tickets Searching and Booking

**User’s Chosen Goal and Task Scenario (User Requirements)**

Group of 4 , 2 adults and 2students, are looking for a 3D action movie in Derby on the night of October 30. The film must cost less than 20 pounds,and has a student discount, with seats in the middle. Ideally, there are bars and restaurants nearby.

**Specific End Result – for the User (System Output)**

Tickets are £18 for adults and £12 for students. The seats are 12 seats in 7 rows, 13 seats in 7 rows, 14 seats in 7 rows and 15 seats in 7 rows. Shops and bars are within a mile.

**Brief List of Tasks / Activities Needed to Achieve the User’s Current Goal (System Functions)**

Go to the cinema and check out the brochure. In each brochure, looking for action movies, browsing movie times and profiles, and draw up a short list of eligible movies. Find all the criteria for the best match. Call the cinema or go to the cinema ticket office to book tickets.

**Brief Indication of Current Usability Problems (Problem Definition)**

It takes a long time to go to the cinema, it takes a long time to choose seats over the phone or in person, it is easy to miss the latest movie, and it takes too long to wait for the next one...

**contents**

[1.Introduction 3](#_Toc10411)

[1.1.Project Area 3](#_Toc22243)

[1.2.User’s Chosen Goal and Task Scenario (User Requirements) 3](#_Toc8159)

[1.3.Specific End Result – for the User (System Output) 3](#_Toc4977)

[1.4.Brief List of Tasks / Activities Needed to Achieve the User’s Current Goal (System Functions) 3](#_Toc12507)

[1.5.Brief Indication of Current Usability Problems (Problem Definition) 3](#_Toc1324)

[2.System Interface Design 4](#_Toc957)

[3. System Implement 8](#_Toc20972)

[3.2. Data Model 10](#_Toc5223)

[3.2.1Users 10](#_Toc26064)

[3.2.2.Movies 11](#_Toc1845)

[3.3.Model related processing 14](#_Toc403)

[3.3.1.User correlation 14](#_Toc22660)

[3.3.2.Movies correlation 17](#_Toc7299)

[3.4. Interactive processing 20](#_Toc25920)

[3.4.1.Login correlation 20](#_Toc16202)

[3.4.2.Order correlation 23](#_Toc32021)

[3.5.Interface 25](#_Toc26091)

[3.5.1.Homepage Interface 25](#_Toc29432)

[3.5.2.Purchase Interface 29](#_Toc10902)

**1.Introduction**

With the development of society and the progress of science and technology, people's living standards are constantly improving. It is an indispensable part of everyone's life to go to the cinema to watch the films of interest in their spare time. Traditional movie viewing needs to go to the cinema for film selection and consultation, which is time-consuming. If there is no favorite movie, it often goes for nothing to reduce the experience of movie viewing. With the development of science and technology and the continuous extension of the Internet, it is the general trend to preview, select and order online movies. This system creates a simple ticket booking model for cinemas, helps cinemas and customers to solve the urgent problems, and improves the service quality and new experience of watching movies.

**1.1.Project Area**

Movie Tickets Searching and Booking

**1.2.User’s Chosen Goal and Task Scenario (User Requirements)**

Group of 4 , 2 adults and 2students, are looking for a 3D action movie in Derby on the night of October 30. The film must cost less than 20 pounds,and has a student discount, with seats in the middle. Ideally, there are bars and restaurants nearby.

**1.3.Specific End Result – for the User (System Output)**

Tickets are £18 for adults and £12 for students. The seats are 12 seats in 7 rows, 13 seats in 7 rows, 14 seats in 7 rows and 15 seats in 7 rows. Shops and bars are within a mile.

**1.4.Brief List of Tasks / Activities Needed to Achieve the User’s Current Goal (System Functions)**

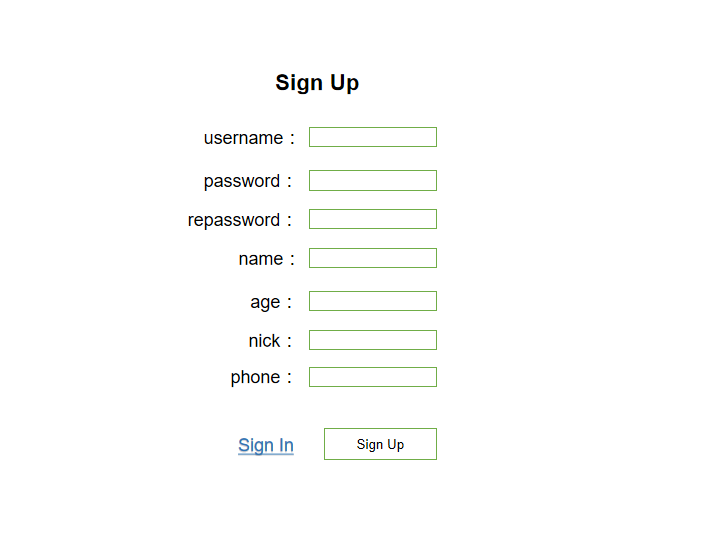
Go to the cinema and check out the brochure. In each brochure, looking for action movies, browsing movie times and profiles, and draw up a short list of eligible movies. Find all the criteria for the best match. Call the cinema or go to the cinema ticket office to book tickets.

**1.5.Brief Indication of Current Usability Problems (Problem Definition)**

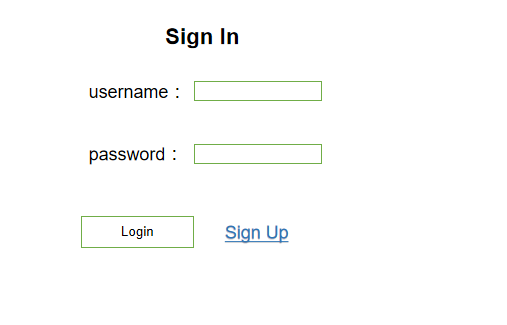
It takes a long time to go to the cinema, it takes a long time to choose seats over the phone or in person, it is easy to miss the latest movie, and it takes too long to wait for the next one...

**2.System Interface Design**

Now I will show the design effect of all interfaces involved in the system, and roughly identify relevant functions.

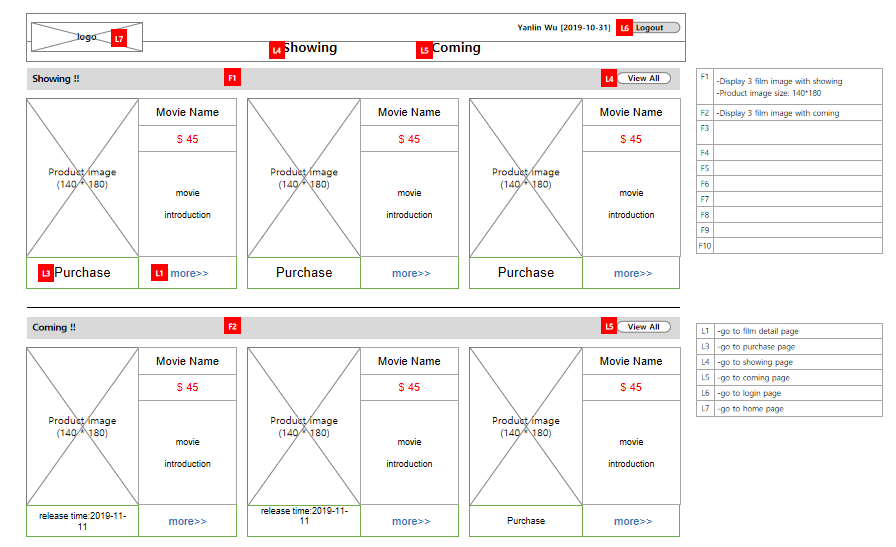
****

SignUp Page

****

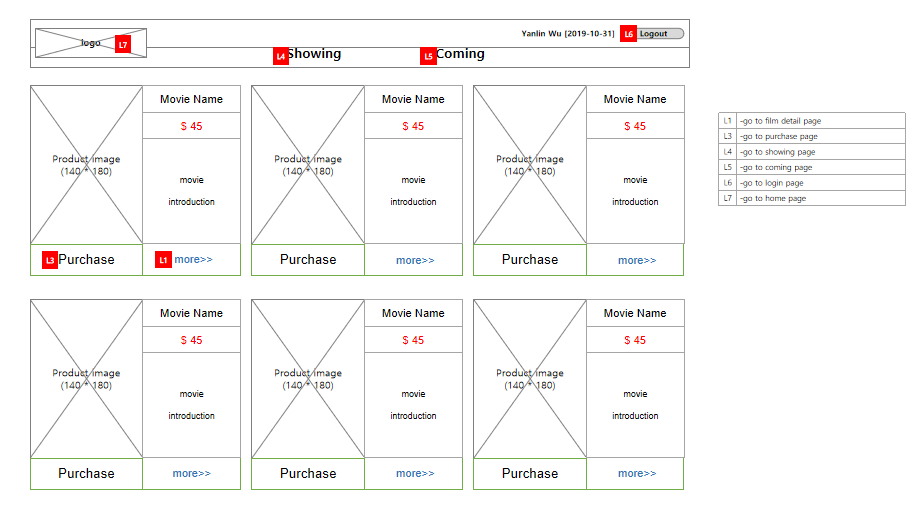
Login Page

To use the movie ticket ordering system, you need to register an account first, and then log in to the system to browse.

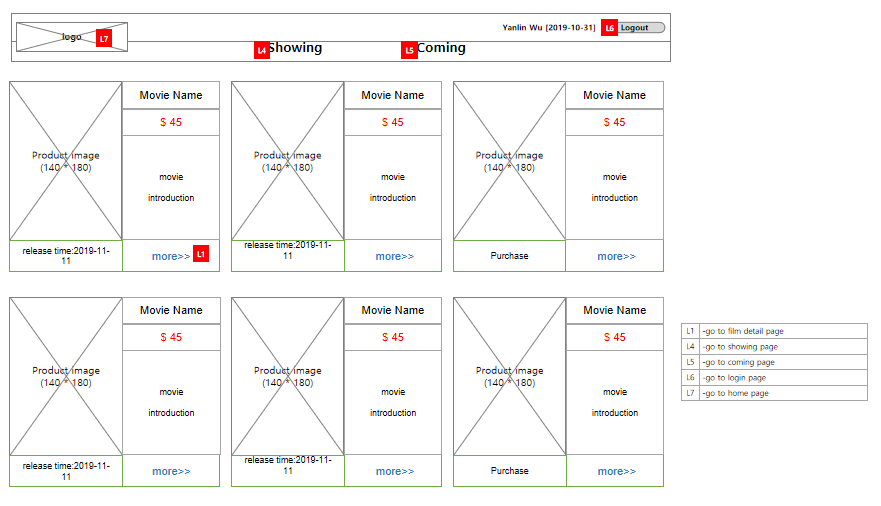


Homepage

The Homepage interface displays some basic information, including the nickname and current date of the user who logs in incorrectly. In addition, there are three currently released movies and the upcoming movies. Every movie has some simple movie information and price.

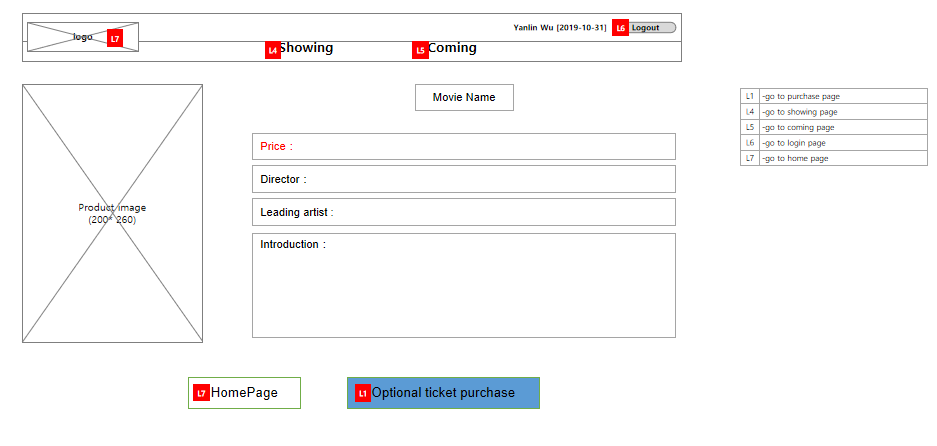


Showing Page



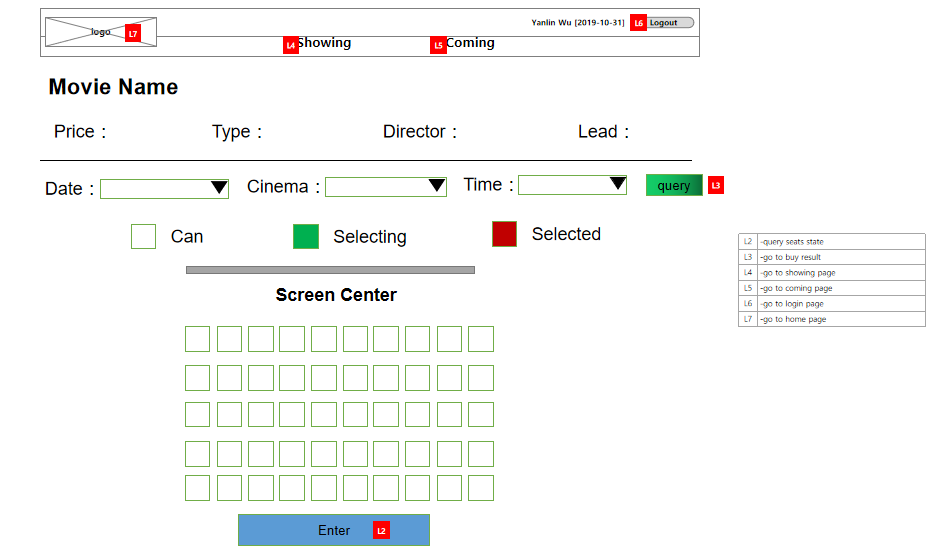
Coming Page

These two screens are classified to show all the films that are on and about to be released.



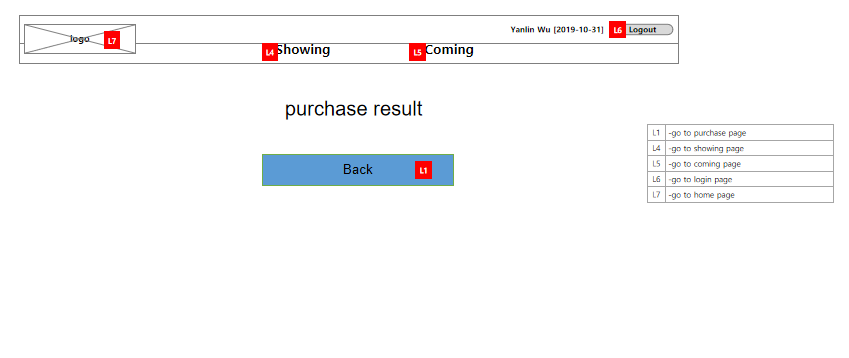
Details Page

The movie details page displays a large amount of information about the movie, where users can learn about the introduction of the movie and then make a decision.



Purchase Page

After selecting a movie, you can choose the date, theater, viewing time and favorite seats. White stands for free, red stands for sold, and green stands for the location you choose.



Purchase Failure

If you have a good balance in your account, congratulations on buying the movie ticket you want.

1. **System Implement**

The system is a web project that uses Java language, MySQL database, JSP interface and Servlet Technology to programming. After learning the related programming technology and ideas, I divided the system into three layers. The first layer is the model layer corresponding to the database, the second layer is the data processing layer for processing the database, and the last is the data presentation layer. There are a lot of system related codes. The following will give an example of the whole framework related code, and then explain the core part of the code.

* 1. **Database design and connection implementation**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Key** | **Field description** |
| username | nvarchar(45) | PRI | username |
| psw | nvarchar(45) |  | password |
| name | nvarchar(45) |  | True name |
| nick | nvarchar(45) |  | User nickname |
| age | INT(11) |  | User age |
| phone | nvarchar(45) |  | User phone |
| money | nvarchar(45) |  | User balance |

Users Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Key** | **Field description** |
| id | INT(11) | PRI | Unique number |
| city | nvarchar(45) |  | Cinema City |
| name | nvarchar(255) |  | The name of the cinema and the cinema |

Cinemas Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Key** | **Field description** |
| id | INT(11) | PRI | Unique number |
| name | nvarchar(45) |  | Movie name |
| price | FLOAT |  | Movie price |
| director | nvarchar(45) |  | Film director |
| artists | nvarchar(45) |  | leading artist |
| leading | nvarchar(45) |  | Film starring |
| intro | nvarchar(255) |  | Movie introduction |
| img | nvarchar(45) |  | Movie pictures |
| state | nvarchar(45) |  | Movie status |
| dates | nvarchar(45) |  | Movie release time |
| type | nvarchar(45) |  | Movie genre |

Movies Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Key** | **Field description** |
| id | INT(11) | PRI | Unique number |
| cinemaId | INT(11) |  | Cinema No |
| movieId | INT(11) |  | Movie No |
| dates | nvarchar(45) |  | Movie date |
| times | nvarchar(45) |  | Movie Time |
| seats | nvarchar(255) |  | Seat status |

Cinemastate Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Key** | **Field description** |
| id | INT(11) | PRI | Unique number |
| cinemaId | INT(11) |  | Cinema No |
| movieId | INT(11) |  | Movie No |
| dates | nvarchar(45) |  | Movie date |
| times | nvarchar(45) |  | Movie Time |
| username | nvarchar(45) |  | User number |
| orderTime | nvarchar(45) |  | Ticket purchase time |
| seats | nvarchar(255) |  | Seats ordered |
| prices | FLOAT |  | Amount spent |

Orderinfo Table

The core code of database connection is as follows:

**import** java.sql.\*;

**public** **class** ConnDB {

**private** String driver = "com.mysql.jdbc.Driver";

**private** String url = "jdbc:mysql://127.0.0.1:3306/movies";

**private** String username = "root";

**private** String password = "root";

**private** Connection ct = **null**;

**public** Connection getConn() {

**try** {

Class.*forName*(driver);

ct = DriverManager.*getConnection*(url, username, password);

} **catch** (Exception e) {

// **TODO**: handle exception

e.printStackTrace();

}

**return** ct;

}

}

* 1. **Data Model**

Now we need to create the data model corresponding to the database table in the code to facilitate the access between the data. The following shows the data model implementation of users table and movies table.

**3.2.1Users**

**public** **class** Users {

**private** String username ;

**private** String psw ;

**private** String name ;

**private** String nick ;

**private** **int** age ;

**private** String phone ;

**private** **float** money ;

**public** String getUsername() {

**return** username;

}

**public** **void** setUsername(String username) {

**this**.username = username;

}

**public** String getPsw() {

**return** psw;

}

**public** **void** setPsw(String psw) {

**this**.psw = psw;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getNick() {

**return** nick;

}

**public** **void** setNick(String nick) {

**this**.nick = nick;

}

**public** **int** getAge() {

**return** age;

}

**public** **void** setAge(**int** age) {

**this**.age = age;

}

**public** String getPhone() {

**return** phone;

}

**public** **void** setPhone(String phone) {

**this**.phone = phone;

}

**public** **float** getMoney() {

**return** money;

}

**public** **void** setMoney(**float** money) {

**this**.money = money;

}

**public** Users(){}

**public** Users(String username, String psw, String name, String nick, **int** age, String phone, **float** money) {

**super**();

**this**.username = username;

**this**.psw = psw;

**this**.name = name;

**this**.nick = nick;

**this**.age = age;

**this**.phone = phone;

**this**.money = money;

}

}

**3.2.2.Movies**

Movie tables are used to store basic information about movies, including introduction, price, and type.

**public** **class** Movies {

**private** **int** id ;

**private** String name ;

**private** **float** price ;

**private** String director ;

**private** String artists ;

**private** String leading ;

**private** String intro ;

**private** String img ;

**private** String state ;

**private** String dates ;

**private** String type ;

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** **float** getPrice() {

**return** price;

}

**public** **void** setPrice(**float** price) {

**this**.price = price;

}

**public** String getDirector() {

**return** director;

}

**public** **void** setDirector(String director) {

**this**.director = director;

}

**public** String getArtists() {

**return** artists;

}

**public** **void** setArtists(String artists) {

**this**.artists = artists;

}

**public** String getLeading() {

**return** leading;

}

**public** **void** setLeading(String leading) {

**this**.leading = leading;

}

**public** String getIntro() {

**return** intro;

}

**public** **void** setIntro(String intro) {

**this**.intro = intro;

}

**public** String getImg() {

**return** img;

}

**public** **void** setImg(String img) {

**this**.img = img;

}

**public** String getState() {

**return** state;

}

**public** **void** setState(String state) {

**this**.state = state;

}

**public** String getDates() {

**return** dates;

}

**public** **void** setDates(String dates) {

**this**.dates = dates;

}

**public** String getType() {

**return** type;

}

**public** **void** setType(String type) {

**this**.type = type;

}

**public** Movies( String name, **float** price, String director, String artists, String leading, String intro,

String img, String state, String dates, String type) {

**this**.name = name;

**this**.price = price;

**this**.director = director;

**this**.artists = artists;

**this**.leading = leading;

**this**.intro = intro;

**this**.img = img;

**this**.state = state;

**this**.dates = dates;

**this**.type = type;

}

**public** Movies(){}

}

**3.3.Model related processing**

Next, we will create operation classes corresponding to the data model, which implement data access and organization operations on related databases. Now, we will show the associated operation classes of users table and movies table, which are usersbo.java and moviesbo.java respectively

**3.3.1.User correlation**

**import** java.sql.Connection;

**import** java.sql.PreparedStatement;

**import** java.sql.ResultSet;

**public** **class** UsersBO {

**private** Connection ct = **null**;

**private** ResultSet rs = **null**;

**private** PreparedStatement ps = **null**;

**public** String updateUser(String username, **float** newMoney) {

String b="false";

**try**{

ConnDB cd = **new** ConnDB();

ct = cd.getConn();

ps = ct.prepareStatement("update users set money='"+

newMoney+"' where username='"+username+"'");

**int** a = ps.executeUpdate();

b="true";

}**catch** (Exception e) {

e.printStackTrace();

} **finally** {

**this**.close();

}

**return** b;

}

**public** Users getUsers(String u) {

Users bean = **new** Users();

**try** {

ConnDB cd = **new** ConnDB();

ct = cd.getConn();

ps = ct.prepareStatement("select \* from users where userName='"

+ u + "' limit 0, 1");

rs = ps.executeQuery();

**if** (rs.next()) {

bean.setUsername(rs.getString(1));

bean.setPsw(rs.getString(2));

bean.setName(rs.getString(3));

bean.setNick(rs.getString(4));

bean.setAge(rs.getInt(5));

bean.setPhone(rs.getString(6));

bean.setMoney(rs.getFloat(7));

}

} **catch** (Exception e) {

e.printStackTrace();

} **finally** {

**this**.close();

}

**return** bean;

}

**public** **boolean** checkUser(String u, String p) {

**boolean** b=**false**;

**try** {

ConnDB cd = **new** ConnDB();

ct = cd.getConn();

ps = ct.prepareStatement("select psw from users where username='"

+ u + "' limit 0, 1");

rs = ps.executeQuery();

**if** (rs.next()) {

String dbPasswd = rs.getString(1);

**if** (dbPasswd.equals(p)) {

b=**true**;

}

}

} **catch** (Exception e) {

e.printStackTrace();

} **finally** {

**this**.close();

}

**return** b;

}

//true:false:exist

**public** String addUser(String username, String psw,

String name, String nick, **int** age,

String phone) {

String b="false";

**if**(isExistUsers(username))**return** "exist";

**try**{

ConnDB cd = **new** ConnDB();

ct = cd.getConn();

ps = ct.prepareStatement("insert into users "

+ "(username, psw, name, nick, age, "

+ "phone, money) values(?,?,?,?,?,?,?)");

ps.setString(1, username);

ps.setString(2, psw);

ps.setString(3, name);

ps.setString(4, nick);

ps.setInt(5, age);

ps.setString(6, phone);

ps.setFloat(7, 100.0f);

**int** a = ps.executeUpdate();

b="true";

}**catch** (Exception e) {

e.printStackTrace();

} **finally** {

**this**.close();

}

**return** b;

}

**private** **boolean** isExistUsers(String username) {

**boolean** b=**false**;

**try** {

ConnDB cd = **new** ConnDB();

ct = cd.getConn();

ps = ct.prepareStatement("select psw from users where username='"

+ username + "' limit 0, 1");

rs = ps.executeQuery();

**if** (rs.next()) {

b=**true**;

}

} **catch** (Exception e) {

e.printStackTrace();

} **finally** {

**this**.close();

}

**return** b;

}

**public** **void** close() {

**try** {

**if** (rs != **null**) {

rs.close();

rs = **null**;

}

**if** (ps != **null**) {

ps.close();

ps = **null**;

}

**if** (ct != **null**) {

ct.close();

ct = **null**;

}

} **catch** (Exception ex) {

ex.printStackTrace();

}

}

}

**3.3.2.Movies correlation**

**import** java.sql.Connection;

**import** java.sql.PreparedStatement;

**import** java.sql.ResultSet;

**import** java.util.ArrayList;

**import** java.util.HashMap;

**import** java.util.Map;

**public** **class** MoviesBO {

**private** Connection ct = **null**;

**private** ResultSet rs = **null**;

**private** PreparedStatement ps = **null**;

**public** Map<String,ArrayList<Movies>> getHomePage() {

Map<String,ArrayList<Movies>> map=**new** HashMap<>();

**try** {

ConnDB cd = **new** ConnDB();

ct = cd.getConn();

ArrayList<Movies> al\_showing=**new** ArrayList<Movies>();

String showing\_sql="select \* from movie where "

+ "state='showing' order by id desc limit 0,3";

ps=ct.prepareStatement(showing\_sql);

rs=ps.executeQuery();

**while**(rs.next()){

Movies gBean=**new** Movies();

gBean.setId(rs.getInt(1));

gBean.setName(rs.getString(2));

gBean.setPrice(rs.getFloat(3));

gBean.setDirector(rs.getString(4));

gBean.setArtists(rs.getString(5));

gBean.setLeading(rs.getString(6));

gBean.setIntro(rs.getString(7));

gBean.setImg(rs.getString(8));

gBean.setState(rs.getString(9));

gBean.setDates(rs.getString(10));

gBean.setType(rs.getString(11));

al\_showing.add(gBean);

}

map.put("showing", al\_showing);

//coming

ArrayList<Movies> al\_coming=**new** ArrayList<Movies>();

showing\_sql="select \* from movie where "

+ "state='coming' order by id desc limit 0,3";

ps=ct.prepareStatement(showing\_sql);

rs=ps.executeQuery();

**while**(rs.next()){

Movies gBean=**new** Movies();

gBean.setId(rs.getInt(1));

gBean.setName(rs.getString(2));

gBean.setPrice(rs.getFloat(3));

gBean.setDirector(rs.getString(4));

gBean.setArtists(rs.getString(5));

gBean.setLeading(rs.getString(6));

gBean.setIntro(rs.getString(7));

gBean.setImg(rs.getString(8));

gBean.setState(rs.getString(9));

gBean.setDates(rs.getString(10));

gBean.setType(rs.getString(11));

al\_coming.add(gBean);

}

map.put("coming", al\_coming);

} **catch** (Exception e) {

// **TODO**: handle exception

e.printStackTrace();

}**finally**{

**this**.close();

}

**return** map;

}

**public** ArrayList<Movies> getMovieByState(String state) {

ArrayList<Movies> al=**new** ArrayList<Movies>();

**try** {

ConnDB cd = **new** ConnDB();

ct = cd.getConn();

String sql="select \* from movie where "

+ "state='"+state+"' order by id desc";

ps=ct.prepareStatement(sql);

rs=ps.executeQuery();

**while**(rs.next()){

Movies gBean=**new** Movies();

gBean.setId(rs.getInt(1));

gBean.setName(rs.getString(2));

gBean.setPrice(rs.getFloat(3));

gBean.setDirector(rs.getString(4));

gBean.setArtists(rs.getString(5));

gBean.setLeading(rs.getString(6));

gBean.setIntro(rs.getString(7));

gBean.setImg(rs.getString(8));

gBean.setState(rs.getString(9));

gBean.setDates(rs.getString(10));

gBean.setType(rs.getString(11));

al.add(gBean);

}

} **catch** (Exception e) {

// **TODO**: handle exception

e.printStackTrace();

}**finally**{

**this**.close();

}

**return** al;

}

**public** Movies getMovies(String id) {

Movies gBean=**new** Movies();

**try** {

ConnDB cd = **new** ConnDB();

ct = cd.getConn();

String showing\_sql="select \* from movie where id='"+id+"'";

ps=ct.prepareStatement(showing\_sql);

rs=ps.executeQuery();

**if**(rs.next()){

gBean.setId(rs.getInt(1));

gBean.setName(rs.getString(2));

gBean.setPrice(rs.getFloat(3));

gBean.setDirector(rs.getString(4));

gBean.setArtists(rs.getString(5));

gBean.setLeading(rs.getString(6));

gBean.setIntro(rs.getString(7));

gBean.setImg(rs.getString(8));

gBean.setState(rs.getString(9));

gBean.setDates(rs.getString(10));

gBean.setType(rs.getString(11));

}

} **catch** (Exception e) {

// **TODO**: handle exception

e.printStackTrace();

}**finally**{

**this**.close();

}

**return** gBean;

}

**public** **void** close() {

**try** {

**if** (rs != **null**) {

rs.close();

rs = **null**;

}

**if** (ps != **null**) {

ps.close();

ps = **null**;

}

**if** (ct != **null**) {

ct.close();

ct = **null**;

}

} **catch** (Exception ex) {

ex.printStackTrace();

}

}

}

**3.4. Interactive processing**

This part mainly realizes the analysis, processing and data return of the interface request, which is the bridge between the front-end interface and the back-end logic.

**3.4.1.Login correlation**

**import** java.io.IOException;

**import** java.io.PrintWriter;

**import** java.text.SimpleDateFormat;

**import** java.util.Date;

**import** javax.servlet.ServletException;

**import** javax.servlet.annotation.WebServlet;

**import** javax.servlet.http.HttpServlet;

**import** javax.servlet.http.HttpServletRequest;

**import** javax.servlet.http.HttpServletResponse;

**import** com.movie.model.Users;

**import** com.movie.model.UsersBO;

@WebServlet("/LoginCL")

**public** **class** LoginCL **extends** HttpServlet {

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

**protected** **void** doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

response.setContentType("text/html");

response.setCharacterEncoding("utf-8");

PrintWriter out=response.getWriter();

String type=request.getParameter("type");

UsersBO ubo=**new** UsersBO();

String u=request.getParameter("username");

String p=request.getParameter("password");

**if**(type.equals("login")){

//login

**if**(ubo.checkUser(u, p)){

//save to session

Users ub=ubo.getUsers(u);

//request.getSession().setAttribute("userInfo", ub);

request.getSession().setAttribute("username", ub.getUsername());

request.getSession().setAttribute("nick", ub.getNick());

//now time

SimpleDateFormat format = **new** SimpleDateFormat("yyyy-MM-dd");

String strDate = format.format(**new** Date());

request.getSession().setAttribute("date", strDate);

request.getRequestDispatcher("homepage.jsp").forward(request, response);

}**else**{

request.setAttribute("loginError", "The user does not exist !");

request.getRequestDispatcher("login.jsp").forward(request, response);

}

}**else**{

//sign up

String repassword=request.getParameter("repassword");

String name=request.getParameter("name");

String age=request.getParameter("age");

String nick=request.getParameter("nick");

String phone=request.getParameter("phone");

**if**(u.equals("")||p.equals("")||repassword.equals("")||

name.equals("")||age.equals("")||nick.equals("")||phone.equals("")){

request.setAttribute("signRet", "All inputs cannot be empty !");

request.getRequestDispatcher("signUp.jsp").forward(request, response);

**return** ;

}

**if**(!p.equals(repassword)){

request.setAttribute("signRet", "Two password entries are inconsistent !");

request.getRequestDispatcher("signUp.jsp").forward(request, response);

**return** ;

}

String ret = ubo.addUser(u, p, name, nick, Integer.*parseInt*(age), phone);

**if**(ret.equals("true"))ret="add success!";

**else** **if**(ret.equals("false"))ret="add fail!";

**else** ret=" !";

request.setAttribute("signRet", ret);

request.getRequestDispatcher("signUp.jsp").forward(request, response);

}

}

**protected** **void** doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

doGet(request, response);

}

}

**3.4.2.Order correlation**

@WebServlet("/OrderCL")

**public** **class** OrderCL **extends** HttpServlet {

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

**protected** **void** doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

response.setContentType("text/html");

response.setCharacterEncoding("utf-8");

PrintWriter out=response.getWriter();

String type=request.getParameter("type");

**if**(type.equals("buy")){

String username=(String)request.getSession().getAttribute("username");

String date=request.getParameter("date");

String cinema=request.getParameter("cinemaId");

String time=request.getParameter("times");

String buy=request.getParameter("buy");

String movieId=request.getParameter("movieId");

String price=request.getParameter("price");

String[] buyArr=buy.split(",");

**float** money=(buyArr.length-1)\*Float.*parseFloat*(price);

UsersBO ubo=**new** UsersBO();

Users user=ubo.getUsers(username);

**if**(user.getMoney()<money){

//error

request.setAttribute("buyError", "You don't have enough money !");

request.setAttribute("id", movieId);

request.getRequestDispatcher("buyRet.jsp").forward(request, response);

}**else**{

**float** newMoney=user.getMoney()-money;

ubo.updateUser(username, newMoney);

//now time

SimpleDateFormat format = **new** SimpleDateFormat("yyyy-MM-dd mm:ss");

String orderTime = format.format(**new** Date());

OrderInfoBO oib=**new** OrderInfoBO();

buy=buy.substring(2);

oib.addOrder(Integer.*parseInt*(cinema), Integer.*parseInt*(movieId), date,

time, username, orderTime, buy, money);

CinemaState cState=oib.getOrderState(cinema, movieId, date, time);

//seat

**if**(cState==**null**){

//0:no;1:Ordered

StringBuffer seats=**new** StringBuffer("00000000000000000000000000000000000000000000000000");

**for**(**int** i=1;i<buyArr.length;i++){

**int** bTempInt=Integer.*parseInt*(buyArr[i]);

seats.replace(bTempInt-1, bTempInt,"1");

}

oib.addCinemaState(Integer.*parseInt*(cinema),

Integer.*parseInt*(movieId), date, time,seats.toString());

}**else**{

StringBuffer seats=**new** StringBuffer(cState.getSeats());

**for**(**int** i=1;i<buyArr.length;i++){

**int** bTempInt=Integer.*parseInt*(buyArr[i]);

seats.replace(bTempInt-1, bTempInt,"1");

}

oib.updateCinemaState(cState.getId(),seats.toString());

}

request.setAttribute("buyError", "Purchase success !");

request.setAttribute("id", movieId);

request.getRequestDispatcher("buyRet.jsp").forward(request, response);

}

}**else** **if**(type.equals("toBuyPage")){

CinemasBO cbo=**new** CinemasBO();

ArrayList<Cinemas> al\_cine= cbo.getCinemas();

request.setAttribute("id", request.getParameter("id"));

request.setAttribute("times", 10+"");

request.setAttribute("cinemaId", al\_cine.get(0).getId()+"");

request.setAttribute("nowDate", request.getSession().getAttribute("date"));

request.getRequestDispatcher("buy.jsp").forward(request, response);

}**else** **if**(type.equals("reBuyPage")){

request.setAttribute("id", request.getParameter("movieId"));

request.setAttribute("times", request.getParameter("times"));

request.setAttribute("cinemaId", request.getParameter("cinemaId"));

request.setAttribute("nowDate", request.getParameter("date"));

request.getRequestDispatcher("buy.jsp").forward(request, response);

}

}

**protected** **void** doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

// **TODO** Auto-generated method stub

doGet(request, response);

}

}

**3.5.Interface**

This part is the implementation of the specific interface, using JSP, CSS, JS and other technologies to complete the layout and implementation of the interface. The following will show the relevant codes of the main interface and the final implementation effect.

**3.5.1.Homepage Interface**

<%@page import=*"com.movie.model.Movies"*%>

<%@page import=*"com.movie.model.MoviesBO"*%>

<%@ page language=*"java"* import=*"java.util.\*"* pageEncoding=*"utf-8"*%>

<%

String path = request.getContextPath();

String basePath = request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";

MoviesBO mbo=**new** MoviesBO();

Map<String,ArrayList<Movies>> map=mbo.getHomePage();

ArrayList<Movies> al\_showing=map.get("showing");

ArrayList<Movies> al\_coming=map.get("coming");

%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>

<head>

<base href=*"*<%=basePath%>*"*>

<title>My JSP 'shopping2.jsp' starting page</title>

<meta http-equiv=*"pragma"* content=*"no-cache"*>

<meta http-equiv=*"cache-control"* content=*"no-cache"*>

<meta http-equiv=*"expires"* content=*"0"*>

<meta http-equiv=*"keywords"* content=*"keyword1,keyword2,keyword3"*>

<meta http-equiv=*"description"* content=*"This is my page"*>

<link href=*"css/myCss.css"* rel=*"stylesheet"* type=*"text/css"* />

</head>

<body background=*"img/bg.jpg"*>

<table width=*"80%"* border=*"0"* align=*"center"*>

<tr>

<td colspan=*"3"* align=*"center"*><jsp:include flush=*"true"* page=*"head.jsp"*></jsp:include></td>

</tr>

<tr>

<td colspan=*"3"*><hr /></td>

</tr>

<tr bgcolor=*"#E1E1E1"*>

<td width=*"33%"* height=*"30"*>Showing ! !</td>

<td width=*"33%"* height=*"30"*>&nbsp;</td>

<td width=*"33%"* height=*"30"* align=*"right"*>

<input name=*"button"* type=*"submit"* class=*"btnHead"* value=*"View All"*

onclick = "window.location.href='cate.jsp?state=showing'"

/></td>

</tr>

<tr align=*"center"*>

<%

**for**(**int** i=0;i<al\_showing.size();i++){

Movies tMs=al\_showing.get(i);

%>

<td width=*"33%"* height=*"220"*>

<table width=*"100%"* border=*"0"* align=*"center"*>

<tr>

<td width=*"60%"* height=*"180"* rowspan=*"3"* align=*"center"* valign=*"middle"*><img src=*"img/*<%=tMs.getImg() %>*"* width=*"140"* height=*"180"* /></td>

<td width=*"40%"* height=*"30"* align=*"center"*><h5><%=tMs.getName() %></h5></td>

</tr>

<tr>

<td width=*"40%"* height=*"30"* align=*"center"* class=*"redText"*>$<%=tMs.getPrice() %></td>

</tr>

<tr>

<td width=*"40%"* height=*"120"* >

<p class=*"comment\_inner"*><%=tMs.getIntro() %></p>

</td>

</tr>

<tr>

<td width=*"60%"* height=*"40"* align=*"center"*>

<form action=*"OrderCL"* method=*"post"*>

<input type=*"hidden"* name=*"type"* value=*"toBuyPage"* />

<input type=*"hidden"* name=*"id"* value=*"*<%=tMs.getId() %>*"* />

<input name=*"button3"* type=*"submit"* class=*"purchaseBtn"* value=*"Purchase"* />

</form>

</td>

<td width=*"40%"* height=*"40"* align=*"center"*>

<input name=*"button2"* type=*"button"* class=*"btnHead"* value=*"more&gt;&gt;"*

onclick = *"window.location.href='info.jsp?id=*<%=tMs.getId() %>*'"*/>

</td>

</tr>

</table>

</td>

<%} %>

</tr>

<tr>

<td colspan=*"3"*><hr /></td>

</tr>

<tr bgcolor=*"#E1E1E1"*>

<td width=*"33%"* height=*"30"*>Coming Soon ! !</td>

<td width=*"33%"* height=*"30"*>&nbsp;</td>

<td width=*"33%"* height=*"30"* align=*"right"*>

<input name=*"button4"* type=*"submit"* class=*"btnHead"* value=*"View All"*

onclick = "window.location.href='cate.jsp?state=coming'"/></td>

</tr>

<tr align=*"center"*>

<%

**for**(**int** i=0;i<al\_coming.size();i++){

Movies tMs=al\_coming.get(i);

%>

<td width=*"33%"* height=*"220"*>

<table width=*"100%"* border=*"0"* align=*"center"*>

<tr>

<td width=*"60%"* height=*"180"* rowspan=*"3"* align=*"center"* valign=*"middle"*><img src=*"img/*<%=tMs.getImg() %>*"* width=*"140"* height=*"180"* /></td>

<td width=*"40%"* height=*"30"* align=*"center"*><h5><%=tMs.getName() %></h5></td>

</tr>

<tr>

<td width=*"40%"* height=*"30"* align=*"center"* class=*"redText"*>$<%=tMs.getPrice() %></td>

</tr>

<tr>

<td width=*"40%"* height=*"120"* >

<p class=*"comment\_inner"*><%=tMs.getIntro() %></p>

</td>

</tr>

<tr>

<td width=*"60%"* height=*"40"* align=*"center"*>

<input name=*"button3"* type=*"button"* class=*"purchaseBtn"* value=*"release time:*<%=tMs.getDates() %>*"* /></td>

<td width=*"40%"* height=*"40"* align=*"center"*>

<input name=*"button2"* type=*"button"* class=*"btnHead"* value=*"more&gt;&gt;"*

onclick = *"window.location.href='info.jsp?id=*<%=tMs.getId() %>*'"*/>

</td>

</tr>

</table>

</td>

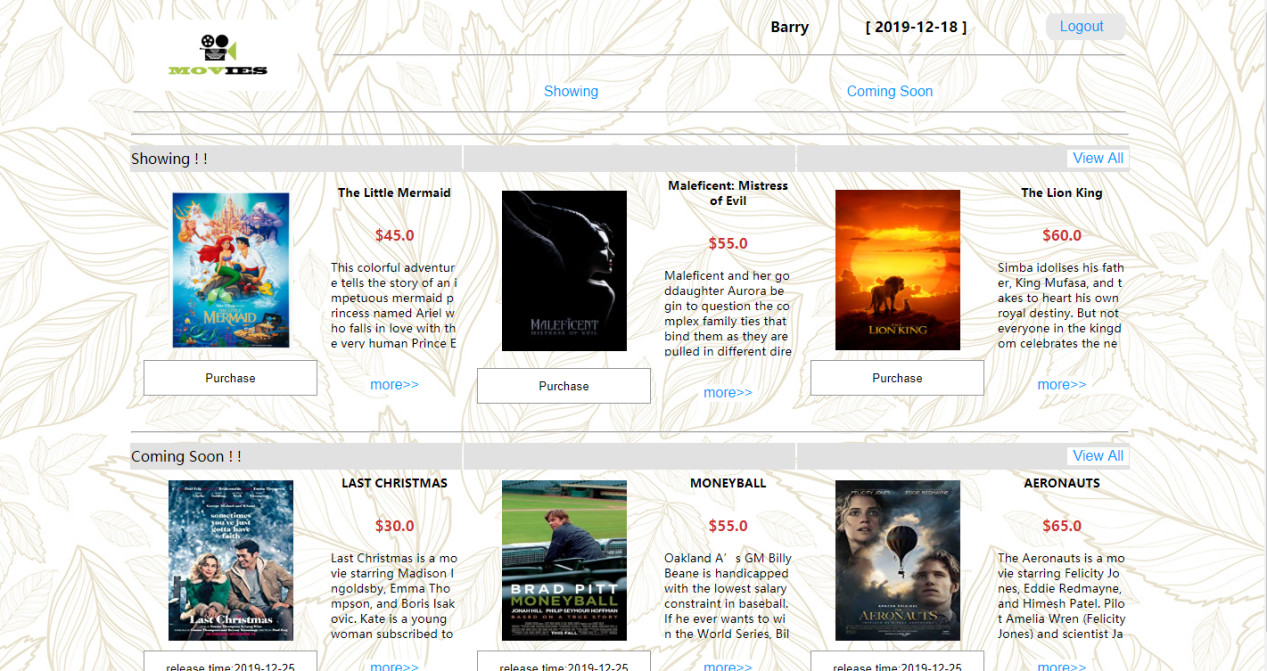
<%} %>

</tr>

</table>

</body>

</html>



Homepage

**3.5.2.Purchase Interface**

<%@page import=*"com.movie.model.CinemaState"*%>

<%@page import=*"com.movie.model.Movies"*%>

<%@page import=*"com.movie.model.MoviesBO"*%>

<%@page import=*"com.movie.model.OrderInfoBO"*%>

<%@page import=*"com.movie.model.CinemasBO"*%>

<%@page import=*"com.movie.model.Cinemas"*%>

<%@ page language=*"java"* import=*"java.util.\*"* pageEncoding=*"utf-8"*%>

<%

String path = request.getContextPath();

String basePath = request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";

String id=(String)request.getAttribute("id");

String times=(String)request.getAttribute("times");

String cinemaId=(String)request.getAttribute("cinemaId");

String calDate=(String)request.getAttribute("nowDate");

MoviesBO mbo=**new** MoviesBO();

Movies movie=mbo.getMovies(id);

CinemasBO cbo=**new** CinemasBO();

ArrayList<Cinemas> al\_cine= cbo.getCinemas();

OrderInfoBO obo=**new** OrderInfoBO();

CinemaState cState=obo.getOrderState(cinemaId, id, calDate, times);

String seatState="00000000000000000000000000000000000000000000000000";

**if**(cState!=**null**)seatState=cState.getSeats();

**char**[] arrState=seatState.toCharArray();

%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>

<head>

<base href=*"*<%=basePath%>*"*>

<title>My JSP 'shopping2.jsp' starting page</title>

<meta http-equiv=*"pragma"* content=*"no-cache"*>

<meta http-equiv=*"cache-control"* content=*"no-cache"*>

<meta http-equiv=*"expires"* content=*"0"*>

<meta http-equiv=*"keywords"* content=*"keyword1,keyword2,keyword3"*>

<meta http-equiv=*"description"* content=*"This is my page"*>

<link href=*"css/myCss.css"* rel=*"stylesheet"* type=*"text/css"* />

<script type=*"text/javascript"*>

**var** buyMap = **new** Map();

**function** fun(obj) {

**var** nowImg= obj.getAttribute("src");

**if**(nowImg=="img/null.jpg"){

obj.src=src="img/select.jpg";

buyMap.set(obj.id,obj.id);

}

**else** **if**(nowImg=="img/used.jpg"){

alert("The seat has been sold !");

}**else**{

obj.src=src="img/null.jpg";

buyMap.**delete**(obj.id);

}

**var** temp="0";

buyMap.forEach(**function**(value, key) {

temp+=","+value;

});

document.getElementById("buy").value=temp;

console.log(temp);

}

**function** fBuy(){

document.getElementById("type").value="buy";

document.form1.submit();

}

**function** fQuery(){

document.getElementById("type").value="reBuyPage";

document.form1.submit();

}

</script>

</head>

<body background=*"img/bg.jpg"*>

<form action=*"OrderCL"* method=*"post"* name=*"form1"*>

<table width=*"80%"* border=*"0"* align=*"center"*>

<tr>

<td align=*"center"*><jsp:include flush=*"true"* page=*"head.jsp"*></jsp:include></td>

</tr>

<tr>

<td height=*"50"*><h2><%=movie.getName() %></h2></td>

</tr>

<tr>

<td>

<table width=*"100%"* border=*"0"*>

<tr>

<td width=*"25%"* height=*"30"* align=*"left"*>Price：$ <%=movie.getPrice() %></td>

<td width=*"25%"* height=*"30"* align=*"left"*>Type：<%=movie.getType() %></td>

<td width=*"25%"* height=*"30"* align=*"left"*>Director：<%=movie.getDirector() %></td>

<td width=*"25%"* height=*"30"* align=*"left"*>Lead：<%=movie.getLeading() %></td>

</tr>

<tr>

<td colspan=*"4"*><hr /></td>

</tr>

<tr>

<td width=*"100%"* height=*"30"* colspan=*"4"* align=*"center"*>

Date:

<input name=*"date"* type=*"date"* value=*"*<%=calDate %>*"* class=*"select150"*/>

Cinema:

<select name=*cinemaId* size=*"1"* class=*"select300"* id=*"select"* value=*"*<%=cinemaId %>*"*>

<%

**for**(**int** i=0;i<al\_cine.size();i++)

{

Cinemas cines= al\_cine.get(i);

%>

<option value=*"*<%=cines.getId() %>*"* <%=cinemaId.equals(cines.getId()+"")?"selected":"" %>><%=cines.getCity() %> <%=cines.getName() %></option>

<%} %>

</select>

Time:

<select name=*"times"* size=*"1"* class=*"select150"* >

<option value=*"10"* <%=times.equals("10")?"selected":"" %>>10:00</option>

<option value=*"14"* <%=times.equals("14")?"selected":"" %>>14:00</option>

<option value=*"16"* <%=times.equals("16")?"selected":"" %>>16:00</option>

<option value=*"18"* <%=times.equals("18")?"selected":"" %>>18:00</option>

<option value=*"20"* <%=times.equals("20")?"selected":"" %>>20:00</option>

<option value=*"22"* <%=times.equals("22")?"selected":"" %>>22:00</option>

</select>

<input type=*"button"* name=*"querySeats"* value=*"querySeats"* onclick="fQuery();"/>

</td>

</tr>

</table>

</td>

</tr>

<tr>

<td><table width=*"100%"* border=*"0"* align=*"center"*>

<tr>

<td width=*"16%"* height=*"50"* align=*"right"*><img src=*"img/null.jpg"* width=*"30"* height=*"30"* /></td>

<td width=*"16%"* height=*"50"* align=*"left"*>Can</td>

<td width=*"16%"* height=*"50"* align=*"right"*><img src=*"img/select.jpg"* width=*"30"* height=*"30"* /></td>

<td width=*"16%"* height=*"50"* align=*"left"*>Selecting</td>

<td width=*"16%"* height=*"50"* align=*"right"*><img src=*"img/used.jpg"* width=*"30"* height=*"30"* /></td>

<td width=*"16%"* height=*"50"* align=*"left"*>Selected</td>

</tr>

</table></td>

</tr>

<tr>

<td height=*"30"* align=*"center"*><img src=*"img/screen.jpg"* width=*"417"* height=*"16"* /></td>

</tr>

<tr>

<td height=*"30"* align=*"center"*><h4>Screen Center</h4></td>

</tr>

<tr>

<td><table width=*"80%"* border=*"0"* align=*"center"*>

<tr>

<%

**for**(**int** i=0;i<50;i++){

**if**(i%10==0){

%>

</tr><tr>

<%} %>

<td width=*"10%"* height=*"50"* align=*"center"* valign=*"middle"*>

<img src=*"img/*<%=arrState[i]=='1'?"used":"null" %>*.jpg"* width=*"30"* height=*"30"* onclick="fun(this)" id=*"*<%=i+1 %>*"*/>

</td>

<%} %>

</tr>

</table></td>

</tr>

<tr><input name=*"buy"* id=*"buy"* type=*"hidden"* value=*""* />

<input name=*"type"* id=*"type"* type=*"hidden"* value=*"buy"* />

<input name=*"movieId"* type=*"hidden"* value=*"*<%=movie.getId() %>*"* />

<input name=*"price"* type=*"hidden"* value=*"*<%=movie.getPrice() %>*"* />

</tr>

<tr>

<td height=*"100"* align=*"center"* valign=*"bottom"*>

<input type=*"button"* class=*"blueBtn100"* value=*"Enter"* onclick="fBuy();"/></td>

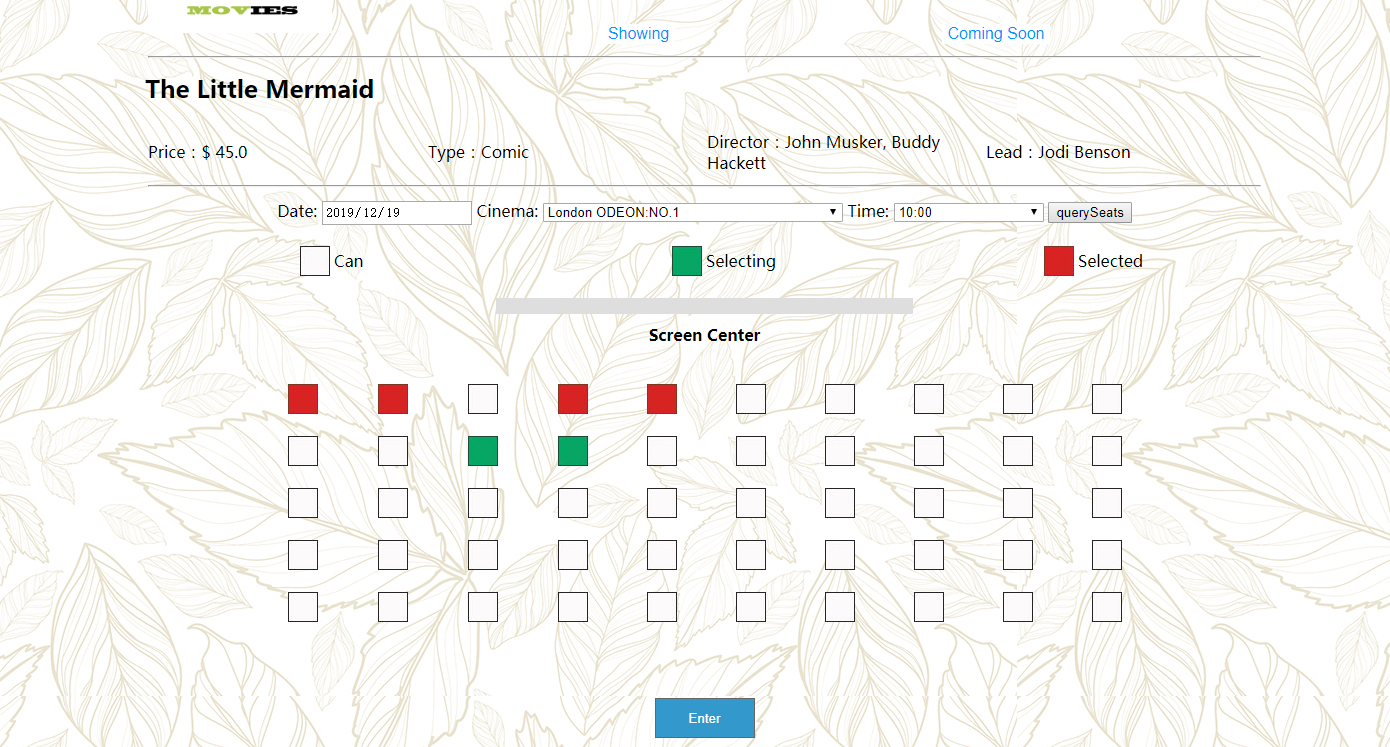
</tr>

</table>

</form>

</body>

</html>



Purchase Interface