

# 영화관 프로젝트 포트폴리오

최 성 윤

## 동기

한국정보교육원의 프로젝트 과정에서 영화관 시스템에 대한 호기심이 생겨 조원들과의 협의를 통해 영화관 홈페이지를 구현하자고 제안하였습니다.

프로젝트 구조는 과정 마지막에 배웠던 Spring Legacy project로 만들어서 사용하자는 의견으로 통합하여 진행하게 되었습니다.

## 프로젝트 기간

기간: 10월 21일~ 10월 30일 (주말 제외 총8일)

맡은 역할:

맡은 기능인 예매기능을 빠르게 끝낸 뒤 부 조장으로서 미흡한 조원들 코드 도와주는 것을 목표로 하였습니다.

10월 21일 ~ 10월 26일 (주말 제외 4일)

프로젝트 시작 당일 DB table, Mapper, Mybatis-config, Spring-MVC 설정 후 조원들에게 배부하였습니다.

10월 26일 날 영화 예매기능 완성하였습니다.

10월 27일~ 10월 29일

다른 팀원이 맡은 로그인, 회원가입, 영화 세부조회 기능을 도와주었습니다. 로그인, 세부조회, 회원가입 기능을 완성하였습니다.

10월 30일 게시판을 완성하여 기능들을 합친 뒤 프로젝트를 마무리 하였습니다.

## SPEC 및 사용 기술

Spec:

CPU: i7-2700

RAM: 16GB

HDD: 1TB

SSD: 250GB

OS: Window 10

WAS: Apache tomcat 8.5

DataBase: Oracle SQL Developer

사용 기술:

[front]-html, CSS

[back]-spring Legacy Project, java, spring, mybatis, EL, JSTL , JavaScript

## 기능 구현

혼자 구현한 기능:

영화 예매기능 전부,  
게시판을 제외한 DB table전부,  
영화 회원가입 기능 중 약관동의 페이지, css를 제외한 전부,  
영화 찾기 기능 중 jsp파일에 있는 html css를 제외한 전부를 구현하였습니다.

팀원들이 구현한 기능:

메인 페이지, 회원가입 약관 JSP, 고객센터, 예매확인,  
예매삭제, 문의 게시판, 자주 묻는 질문, header.css, tail.css, 기존에 있던 기능 css를  
재 수정하였습니다.

## 메인 페이지

C G X

HOME

영화

예매

고객 지원

[로그인](#) | [회원 가입](#)



## 예매 및 고객지원 페이지

예매와 고객지원 기능은 로그인 을 해야지만 사용이 가능하도록 설정하였습니다.

'로그인이 필요한 기능입니다.' 페이지가 뜨고 2초 후 로그인 페이지로 이동합니다.

C G X				C G X			
HOME 영화 예매 고객 지원				HOME 영화 예매 고객 지원			
<div>로그인이 필요한 기능입니다.</div> <div>이 기능은 로그인이 필요한 기능입니다. 로그인 하신 후 다시 시도해주세요. 잠시 후 로그인 화면으로 이동합니다.</div>				<div>회원 로그인</div> <div>아이디 <input type="text" value="jycobh"/></div> <div>암호 <input type="password" value="123456"/></div> <div><input type="button" value="로그인"/> <input type="button" value="취소"/></div>			

# 회원 가입

왼쪽에서 오른쪽순서로 회원가입 페이지가 진행됩니다.

회원 등록 시 아이디는 4~8자 제한이 있습니다.

회원등록을 마치면 DB customer table에 데이터가 등록됩니다.

CGX 이용약관

HOME 영화 예매 고객 지원

CGX 이용약관

회원 등록

회원 등록

회원가입

CUSTOMER_UID	CUSTOMER_ID	CUSTOMER_PW	CUSTOMER_NAME	CUSTOMER_RES	CUSTOMER_TEL
1	5 bbbb	1234	최성윤	9505311	01023451245



## 로그인 실패

아이디가 존재하지 않거나 비밀번호가 틀릴 경우 로그인 이 되지 않습니다.

C G X

로그인

HOME영화예매고객 지원

로그인 실패!

로그인에 실패하였습니다. 아이디 또는 암호가 맞지 않습니다.  
잠시후 다시 로그인 화면으로 돌아갑니다.

## 영화 목록 페이지

영화 예매목록과 DB table을 공유합니다.

좋아하는 영화, 찾고싶은 영화를CGX에서 찾아드립니다!


영화검색 :

장르

- ☐ 드라마
- ☒ 액션
- ☒ 범죄
- ☐ 전체선택


관람등급

- ☐ 12세 관람가
- ☐ 15세 관람가
- ☐ 청소년 관람불가
- ☐ 전체선택



테넷

장르 - 액션, 12세 이용가



소리도 없이





장르 - 범죄, 18세 이용가

	MOVIERANK	TITLE	LOCA	GENRE	TTIME	GRADE
1	1	삼진그룹 영어토익반	3관	드라마	12:30	12
2	2	테넷	1관	액션	10:30	12
3	3	베이비티스	2관	드라마	11:30	15
4	4	소리도 없이	4관	범죄	13:30	18

## 영화 예매 목록 페이지

로그인 후 예매 버튼을 누르면 나오는 페이지 입니다.  
사진을 클릭 시 상영관 좌석페이지로 이동합니다.

yychh 님 로그아웃

HOME	영화	예매	고객 지원		
순위	영화	상영관	장르	시간	나이제한
1위 삼진그룹 영어 토익반		3관	드라마	12:30	12세 이상
2위 테넷		1관	액션	12:30	12세 이상
3위 베이비티스		2관	드라마	11:30	15세 이상
4위 소리도 없이		4관	범죄	13:30	18세 이상


	MOVIERANK	TITLE	LOCA	GENRE	TTIME	GRADE
1	1	삼진그룹 영어토익반	3관	드라마	12:30	12
2	2	테넷	1관	액션	10:30	12
3	3	베이비티스	2관	드라마	11:30	15
4	4	소리도 없이	4관	범죄	13:30	18

## 좌석 예매

사회적 거리두기 를 고려하여 짝수 좌석은 x 표시로 변경하였습니다.

예매 취소 기능에 좌석 번호가 겹치지 않고 삭제하고 싶다는 조장의 말에 좌석 번호를 상영관에 따라 다르게 설정하였습니다.

원하시는 좌석 한 자리를 선택해 주세요.

 안내 - 사회적 거리두기 2단계로 인하여 영화관 좌석이 한자리 씩 떨어져 있습니다. 이용에 불편을 드려 죄송합니다.

4관 스크린




41 X 43 X 45 X 47 X 49 X



취소

원하시는 좌석 한 자리를 선택해 주세요.

 안내 - 사회적 거리두기 2단계로 인하여 영화관 좌석이 한자리 씩 떨어져 있습니다. 이용에 불편을 드려 죄송합니다.

2관 스크린



21 X 23 X 25 X 27 X 29 X



취소

## 예매 하기


좌석을 클릭 시 해당 영화의 정보를 보여주고 클릭 한 좌석번호를 넘겨줍니다.  
결제하기를 누르면 결제 후 메인 페이지로 넘어갑니다.  
결제를 선택 시 예매 정보가 DB ticket table에 저장됩니다.

	HOME	영화	예매	고객 지원
<b>예매 정보</b>				
회원 아이디:	<input type="text" value="bbbb"/>			
영화 순위:	<input type="text" value="4"/>			
영화 제목:	<input type="text" value="소리도 없이"/>			
상영관:	<input type="text" value="4관"/>			
장르:	<input type="text" value="범죄"/>			
시작시간:	<input type="text" value="13:30"/>			
관람 연령:	<input type="text" value="18"/>	세 이상		
결제 금액:	<input type="text" value="10000"/>			
좌석 번호 :	<input type="text" value="49"/>			
결제하시겠습니까?				
<input type="button" value="결제"/> <input type="button" value="취소"/>				

## 예매 후 좌석

예매 후 4관 좌석 49번 좌석은 x 처리 되었습니다.

원하시는 좌석 한 자리를 선택해 주세요.

 안내 - 사회적 거리두기 2단계로 인하여 영화관 좌석이 한자리 씩 떨어져 있습니다. 이용에 불편을 드려 죄송합니다.

4관 스크린



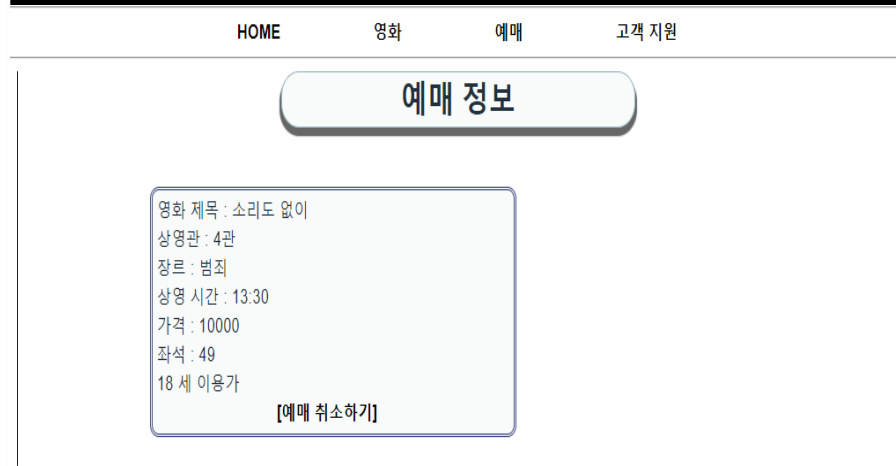
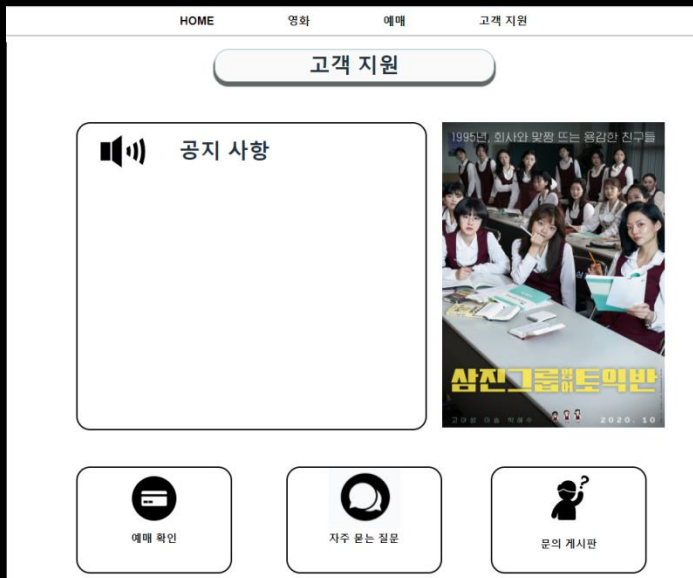
41 X 43 X 45 X 47 X X X



취소

## 고객 지원 페이지

예매 후 해당 고객의 예매를 확인하기 위한 기능이 있습니다.  
예매 확인을 누를 시 예매 정보가 뜨며, 예매 정보는 DB ticket table 데이터에서 가져온 값입니다.



## DB 목록

회원 가입을 위해 만든 데이터베이스 테이블입니다. 고유번호 UID 는 개발과정에서 식별하기 위해 기본 키로 설정 하였으나 사용하지 않았습니다.

아이디, 주민등록번호, 전화번호는 중복불가 이므로 not null unique를 추가하여 만들었습니다.

❖	COLUMN_NAME	❖	DATA_TYPE	❖	NULLABLE	DATA_DEFAULT	❖	COLUMN_ID	❖	COMMENTS
1	CUSTOMER_UID		NUMBER(6,0)		No	(null)		1		(null)
2	CUSTOMER_ID		VARCHAR2(100 BYTE)		No	(null)		2		(null)
3	CUSTOMER_PW		VARCHAR2(100 BYTE)		Yes	(null)		3		(null)
4	CUSTOMER_NAME		VARCHAR2(100 BYTE)		Yes	(null)		4		(null)
5	CUSTOMER_RES		VARCHAR2(100 BYTE)		No	(null)		5		(null)
6	CUSTOMER_TEL		VARCHAR2(20 BYTE)		No	(null)		6		(null)

```
/*고객 테이블 생성*/
create table customer(
    customer_UID number(6) not null,
    customer_id varchar2(100)not null UNIQUE,
    customer_pw varchar2(100),
    customer_name varchar2(100),
    customer_res varchar2(100)not null UNIQUE,
    customer_tel varchar2(20)not null UNIQUE,
    CONSTRAINT customer_customer_UID_pk PRIMARY key(customer_UID)
);
```



## DB 목록

예매할 영화목록과 영화 상세조회 공용으로 만든 테이블입니다.

Insert into로 데이터 값을 넣은 상태로 사용하였습니다.

Values값을 sequence를 생성하여 movieRank 값을 넣었습니다.(영화순위)  
(영화순위, 제목, 상영관, 장르, 시간, 나이제한)

/*영화 목록 생성*/		⚡ COLUMN_NAME	⚡ DATA_TYPE	⚡ NULLABLE	DATA_DEFAULT	⚡ COLUMN_ID	⚡ COMMENTS
create table reservation(		1 MOVIERANK	NUMBER(10,0)	No	(null)	1	(null)
movieRank number(10) PRIMARY key,		2 TITLE	VARCHAR2(30 BYTE)	Yes	(null)	2	(null)
title varchar2(30),		3 LOCA	VARCHAR2(30 BYTE)	Yes	(null)	3	(null)
loca varchar2(30),		4 GENRE	VARCHAR2(30 BYTE)	Yes	(null)	4	(null)
genre varchar2(30),		5 TTIME	VARCHAR2(30 BYTE)	Yes	(null)	5	(null)
Ttime varchar2(30),		6 GRADE	VARCHAR2(30 BYTE)	Yes	(null)	6	(null)
grade varchar2(30)							
);							
create SEQUENCE seq_movierank		⚡ MOVIERANK	⚡ TITLE	⚡ LOCA	⚡ GENRE	⚡ TTIME	⚡ GRADE
start with 1		1	1 삼진그룹 영어토익반	3관	드라마	12:30	12
increment by 1;		2	2 테넷	1관	액션	10:30	12
drop SEQUENCE seq_movierank;		3	3 베이비티스	2관	드라마	11:30	15
insert into reservation(movieRank,title,loca,genre,Ttime,grade)		4	4 소리도 없이	4관	범죄	13:30	18
values (SEQ_MOVIERANK.nextval, '테넷', '1관', '액션', '10:30', '12');							
insert into reservation(movieRank,title,loca,genre,Ttime,grade)							
values (SEQ_MOVIERANK.nextval, '베이비티스', '2관', '드라마', '11:30', '15');							
insert into reservation(movieRank,title,loca,genre,Ttime,grade)							
values (SEQ_MOVIERANK.nextval, '삼진그룹 영어토익반', '3관', '드라마', '12:30', '12');							
insert into reservation(movieRank,title,loca,genre,Ttime,grade)							
values (SEQ_MOVIERANK.nextval, '소리도 없이', '4관', '범죄', '13:30', '18');							

## DB 목록

영화 상영관이 1관부터 4관까지 있으므로 테이블을 4개 생성하여 좌석을 만들었습니다. 코로나로 인해 사회적 거리두기를 고려하여 좌석을 생성시 2씩 증가하도록 Sequence를 설정하였습니다.

```
/*1,2,3,4 상영관 좌석 테이블*/
create table seatNo1(SeatNum1 NUMBER(19));
create table seatNo2(SeatNum2 NUMBER(19));
create table seatNo3(SeatNum3 NUMBER(19));
create table seatNo4(SeatNum4 NUMBER(19));

create SEQUENCE seq_movieloca
start with 1
increment by 2;
drop SEQUENCE seq_movieloca;
insert into seatNo(moiveLocation,SeatNum)
values ('1관',seq_movieloca.nextval);
insert into seatNo(moiveLocation,SeatNum)
values ('2관',seq_movieloca.nextval);
insert into seatNo(moiveLocation,SeatNum)
values ('3관',seq_movieloca.nextval);
insert into seatNo(moiveLocation,SeatNum)
values ('4관',seq_movieloca.nextval);

insert into seatNo1(SeatNum)
values (seq_movieloca.nextval);
insert into seatNo2(SeatNum)
values (seq_movieloca.nextval);
insert into seatNo3(SeatNum)
values (seq_movieloca.nextval);
insert into seatNo4(SeatNum)
values (seq_movieloca.nextval);
```

	SEATNUM1
1	11
2	13
3	15
4	17
5	19

	SEATNUM2
1	21
2	23
3	25
4	27
5	29

	SEATNUM3
1	31
2	33
3	35
4	37
5	39

	SEATNUM4
1	41
2	43
3	45
4	47
5	49

## DB 목록

예매 정보를 입력 받을 테이블입니다.

<pre> create table ticket(   res_id varchar2(30),     res_moive number(37) ,     res_title VARCHAR2(100),     res_loca VARCHAR2(100),     res_genre varchar2(100),     res_time VARCHAR2(100),     res_age NUMBER(37),     res_price NUMBER(37),     res_seat NUMBER(37) ); </pre>	⚡ COLUMN_NAME	⚡ DATA_TYPE	⚡ NULLABLE	DATA_DEFAULT	⚡ COLUMN_ID	⚡ COMMENTS
	1 RES_ID	VARCHAR2(30 BYTE)	Yes	(null)	1 (null)	
	2 RES_MOIVE	NUMBER(37,0)	Yes	(null)	2 (null)	
	3 RES_TITLE	VARCHAR2(100 BYTE)	Yes	(null)	3 (null)	
	4 RES_LOCA	VARCHAR2(100 BYTE)	Yes	(null)	4 (null)	
	5 RES_GENRE	VARCHAR2(100 BYTE)	Yes	(null)	5 (null)	
	6 RES_TIME	VARCHAR2(100 BYTE)	Yes	(null)	6 (null)	
	7 RES_AGE	NUMBER(37,0)	Yes	(null)	7 (null)	
	8 RES_PRICE	NUMBER(37,0)	Yes	(null)	8 (null)	
	9 RES_SEAT	NUMBER(37,0)	Yes	(null)	9 (null)	

# CONFIG.XML

DB와 연결하기 위해 설정한 xml입니다.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE configuration
3 PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
4 "http://mybatis.org/dtd/mybatis-3-config.dtd">
5
6 <configuration>
7   <properties resource="config/mybatis/db.properties"/>
8   <typeAliases>
9     <typeAlias type="kr.kedu.movie.spms.vo.MemberVO" alias="reservation"/>
10    <typeAlias type="kr.kedu.movie.spms.vo.SeatVO1" alias="seatlocal1"/>
11    <typeAlias type="kr.kedu.movie.spms.vo.SeatVO2" alias="seatlocal2"/>
12    <typeAlias type="kr.kedu.movie.spms.vo.SeatVO3" alias="seatlocal3"/>
13    <typeAlias type="kr.kedu.movie.spms.vo.SeatVO4" alias="seatlocal4"/>
14    <typeAlias type="kr.kedu.movie.spms.vo.customerReservationVO" alias="customRes"/>
15    <typeAlias type="kr.kedu.movie.spms.vo.CustomerVO" alias="custom"/>
16    <typeAlias type="kr.kedu.movie.spms.vo.MemberVO" alias="movie"/>
17  </typeAliases>
18
19  <environments default="development">
20    <environment id="development">
21      <transactionManager type="JDBC"/>
22      <dataSource type="POOLED">
23        <property name="driver" value="${oracle.driver}"/>
24        <property name="url" value="${oracle.url}"/>
25        <property name="username" value="${oracle.username}"/>
26        <property name="password" value="${oracle.password}"/>
27      </dataSource>
28    </environment>
29  </environments>
30  <mappers>
31    <mapper resource="config/sqlmap/Sql-mapper.xml"/>
32    <mapper resource="config/sqlmap/cus_mapper.xml"/>
33  </mappers>
34 </configuration>
```

- config
  - mybatis
    - db.properties
    - mybatis-config.xml
  - spring
    - spring-mvc.xml
  - sqlmap
    - cus\_mapper.xml
    - Sql-mapper.xml

```
1 oracle.driver=oracle.jdbc.driver.OracleDriver
2 oracle.url=jdbc:oracle:thin:@localhost:1521:xe
3 oracle.username=hr
4 oracle.password=hr
```

# POM.XML

Springframework, mybatis, ajax, fileupload, maven를 사용하기 위해 만든 xml입니다.

```
40 <dependency>
41   <groupId>org.springframework</groupId>
42   <artifactId>spring-jdbc</artifactId>
43   <version>5.0.2.RELEASE</version>
44 </dependency>
45 <dependency>
46   <groupId>org.springframework</groupId>
47   <artifactId>spring-webmvc</artifactId>
48   <version>${org.springframework-version}</version>
49 </dependency>
50
51 <dependency>
52   <groupId>org.mybatis</groupId>
53   <artifactId>mybatis-spring</artifactId>
54   <version>2.0.5</version>
55 </dependency>
56 <dependency>
57   <groupId>org.mybatis</groupId>
58   <artifactId>mybatis</artifactId>
59   <version>3.5.6</version>
60 </dependency>
61
62 <dependency>
63   <groupId>com.fasterxml.jackson.core</groupId>
64   <artifactId>jackson-core</artifactId>
65   <version>2.12.0-rc1</version>
66 </dependency>
67 <dependency>
68   <groupId>com.fasterxml.jackson.core</groupId>
69   <artifactId>jackson-databind</artifactId>
70   <version>2.12.0-rc1</version>
71 </dependency>
72 <dependency>
73   <groupId>com.fasterxml.jackson.core</groupId>
74   <artifactId>jackson-annotations</artifactId>
75   <version>2.12.0-rc1</version>
76 </dependency>
77
78 <!-- 커넥션 풀 관리 추가 -->
79 <dependency>
80   <groupId>org.apache.commons</groupId>
81   <artifactId>commons-dbcp2</artifactId>
82   <version>2.8.0</version>
83 </dependency>
84
85 <dependency>
86   <groupId>commons-fileupload</groupId>
87   <artifactId>commons-fileupload</artifactId>
88   <version>1.3.1</version>
89 </dependency>
```

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
4   <!-- spring legacy setting -->
5   <modelVersion>4.0.0</modelVersion>
6   <groupId>kr.kedu</groupId>
7   <artifactId>movie</artifactId>
8   <name>Movie</name>
9   <packaging>war</packaging>
10  <version>1.0.0-BUILD-SNAPSHOT</version>
11  <repositories>
12    <repository>
13      <id>oracle</id>
14      <name>ORACLE JDBC Repository</name>
15      <url>http://maven.jahia.org/maven2</url>
16    </repository>
17  </repositories>
18
19  <properties>
20    <java-version>1.8</java-version>
21    <org.springframework-version>5.0.2.RELEASE</org.springframework-version>
22    <org.aspectj-version>1.6.10</org.aspectj-version>
23    <org.slf4j-version>1.6.0</org.slf4j-version>
24  </properties>
25  <dependencies>
26    <!-- Spring -->
27    <dependency>
28      <groupId>org.springframework</groupId>
29      <artifactId>spring-context</artifactId>
30      <version>${org.springframework-version}</version>
31      <exclusions>
32        <!-- Exclude Commons Logging in favor of SLF4j -->
33        <exclusion>
34          <groupId>commons-logging</groupId>
35          <artifactId>commons-logging</artifactId>
36        </exclusion>
37      </exclusions>
38    </dependency>
39
40    <dependency>
41      <groupId>org.springframework</groupId>
42      <artifactId>spring-jdbc</artifactId>
43      <version>5.0.2.RELEASE</version>
44    </dependency>
45    <dependency>
46      <groupId>org.springframework</groupId>
47      <artifactId>spring-webmvc</artifactId>
48      <version>${org.springframework-version}</version>
49    </dependency>
50  </dependencies>
```

# WEB.XML

Dispatcher 기능을 하며 웹 페이지에 encoding하는 역할을 하는 xml입니다.  
URL주소가 .do로 끝나야 실행이 됩니다.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd" id="WebApp_ID" version="3.1">
3   <display-name>Movie</display-name>
4
5
6   <servlet>
7     <servlet-name>dispatcher</servlet-name>
8     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
9     <init-param>
10       <param-name>contextConfigLocation</param-name>
11       <param-value>
12         classpath:config/spring/spring-mvc.xml
13       </param-value>
14     </init-param>
15     <load-on-startup>1</load-on-startup>
16   </servlet>
17
18   <servlet-mapping>
19     <servlet-name>dispatcher</servlet-name>
20     <url-pattern>*.do</url-pattern>
21   </servlet-mapping>
22   <servlet-mapping>
23     <servlet-name>dispatcher</servlet-name>
24     <url-pattern>*.js$</url-pattern>
25   </servlet-mapping>
26
27   <error-page>
28     <error-code>500</error-code>
29     <location>/WEB-INF/view/error/500error.jsp</location>
30   </error-page>
31
32   <filter>
33     <filter-name>encodingFilter</filter-name>
34     <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
35     <init-param>
36       <param-name>encoding</param-name>
37       <param-value>UTF-8</param-value>
38     </init-param>
39   </filter>
40
41   <filter-mapping>
42     <filter-name>encodingFilter</filter-name>
43     <url-pattern>*</url-pattern>
44   </filter-mapping>
45
46   <welcome-file-list><!-- 기본 홈페이지 목록 -->
47     <welcome-file>Mainpage.jsp</welcome-file>
48   </welcome-file-list>
49 </web-app>
```

# MAPPER.XML

DB와 연동하여 자바 class에 있는 VO에 데이터 값을 넣기 위해 만든 Mapper.xml 입니다.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE mapper
3 PUBLIC "-//mybatis.org/DTD Mapper 3.0//EN"
4 "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
50 <mapper namespace="movie.spm.dao.MemberDAO">
60 <resultMap type="custom" id="memberResultMap0">
7 <id column="customer_UID" property="UID"/>
8 <id column="customer_id" property="id"/>
9 <id column="customer_name" property="name"/>
10 <id column="customer_pw" property="password"/>
11 <id column="customer_res" property="residentNum"/>
12 <id column="customer_tel" property="phoneNum"/>
13 </resultMap>
140 <insert id="insert" parameterType="custom">
15 insert into customer
16 (customer_UID,customer_id,customer_name,customer_pw,customer_res,customer_tel)
17 values(seq_members_UID.nextval,#{id},#{name},#{password},#{residentNum},#{phoneNum})
18 </insert>
190 <select id="exist" resultMap="memberResultMap0">
20 select customer_id,customer_pw from customer
21 where customer_id=#{id} and customer_pw=#{password}
22 </select>
230 <select id="selectlist" resultMap="memberResultMap0">
24
25 </select>
26
27 </mapper>
```

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE mapper
3 PUBLIC "-//mybatis.org/DTD Mapper 3.0//EN"
4 "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
50 <mapper namespace="movie.spm.dao.MemberDAO">
60 <resultMap type="reservation" id="memberResultMap">
7 <id column="movieRank" property="movieRank" />
8 <id column="title" property="title" />
9 <id column="loca" property="loca" />
10 <id column="genre" property="genre" />
11 <id column="Ttime" property="totime" />
12 <id column="grade" property="grade" />
13 </resultMap>
140 <resultMap type="seatlocal" id="seatResultMap1">
15 <id column="SeatNum1" property="seatNumber1" />
16 </resultMap>
17
180 <resultMap type="seatloca2" id="seatResultMap2">
19 <id column="SeatNum2" property="seatNumber2" />
20 </resultMap>
21
220 <resultMap type="seatloca3" id="seatResultMap3">
23 <id column="SeatNum3" property="seatNumber3" />
24 </resultMap>
25
260 <resultMap type="seatloca4" id="seatResultMap4">
27 <id column="SeatNum4" property="seatNumber4" />
28 </resultMap>
29
300 <resultMap type="customRes" id="customReslutMap">
31 <id column="res_id" property="resid" />
32 <id column="res_moive" property="movieRank2" />
33 <id column="res_title" property="title2" />
34 <id column="res_loca" property="loca2" />
35 <id column="res_genre" property="genre2" />
36 <id column="res_time" property="totime2" />
37 <id column="res_age" property="grade2" />
38 <id column="res_price" property="price" />
39 <id column="res_seat" property="seatNo" />
40 </resultMap>
410 <resultMap type="movie" id="memberResultMap2">
42
43 <id column="MOVIERANK" property="MovieRank" />
44 <id column="TITLE" property="title" />
45 <id column="LOCA" property="location" />
46 <id column="GENRE" property="genre" />
47 <id column="TTIME" property="time" />
48 <id column="GRADE" property="grade" />
49
50 </resultMap>
```

# SPRING-MVC.XML

Spring-mvc 구조에서 해당 경로에 있는 파일들을 알아서 찾아 실행하게 하는 xml 입니다. 그 외에 DB와 연동하는 SqlSession, transaction처리, 파일 업로드 용량제한이 있습니다.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4       xmlns:mvc="http://www.springframework.org/schema/mvc"
5       xmlns:p="http://www.springframework.org/schema/p"
6       xmlns:c="http://www.springframework.org/schema/c"
7       xmlns:tx="http://www.springframework.org/schema/tx"
8       xmlns:context="http://www.springframework.org/schema/context"
9       xsi:schemaLocation="http://www.springframework.org/schema/beans
10      http://www.springframework.org/schema/beans/spring-beans.xsd
11      http://www.springframework.org/schema/context/spring-context.xsd
12      http://www.springframework.org/schema/tx/spring-tx.xsd
13      http://www.springframework.org/schema/mvc/spring-mvc.xsd"
14      >
15     <context:component-scan base-package="kr.kedu.movie.spm">
16     </context:component-scan>
17     <context:property-placeholder
18     location="classpath:config/mybatis/db.properties"/>
19     <bean id="dataSource" class="org.apache.commons.dbcp2.BasicDataSource"
20     destroy-method="close"
21     p:driverClassName="${oracle.driver}"
22     p:url="${oracle.url}"
23     p:username="${oracle.username}"
24     p:password="${oracle.password}" />
25     <bean id="sqlSessionFactory"
26     class="org.mybatis.spring.SqlSessionFactoryBean"
27     <property name="dataSource" ref="dataSource" />
28     <property name="configLocation"
29     value="classpath:config/mybatis/mybatis-config.xml"/>
30     <!-- typeAliasesPackage는 자동으로 spms.vo 클래스는
31     클래스들을 클래스패스 경로로 찾아서 읽어옵니다. -->
32     </bean>
33     <!-- sqlSessionTemplate는 스프링에서 제공하는
34     sqlSession 객체 -->
35     <bean id="sqlSessionTemplate"
36     class="org.mybatis.spring.SqlSessionTemplate">
37     <constructor-arg ref="sqlSessionFactory" />
38     </bean>
39     <!-- 트랜잭션 처리 -->
40     <bean id="transactionManager"
41     class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
42     <property name="dataSource" ref="dataSource" />
43     </bean>
44     <tx:annotation-driven transaction-manager="transactionManager" />
45     <!-- 스프링에서 제공하는 인코딩 @어노테이션을 통해서 메시지를 주고 받는데
46     인코딩 방식을 지정 -->
47     <!-- 스프링에서 제공하는 인코딩 @어노테이션을 통해서 메시지를 주고 받는데
48     인코딩 방식을 지정 -->
49     </beans>
```

```
59 <!-- @ResponseBody로 String 처리할때 한글처리 -->
60 <mvc:message-converters>
61     <bean class="org.springframework.http.converter.StringHttpMessageConverter">
62     <property name="supportedMediaTypes">
63     <list>
64     <value>text/html; charset=UTF-8</value>
65     </list>
66     </property>
67     </bean>
68     <bean class="org.springframework.http.converter.json.MappingJackson2HttpMessageConverter">
69     <property name="supportedMediaTypes">
70     <list>
71     <value>text/html; charset=UTF-8</value>
72     <value>application/json; charset=UTF-8</value>
73     </list>
74     </property>
75     </bean>
76 </mvc:message-converters>
77 </mvc:annotation-driven>
78
79 <!-- 최대 업로드 파일 사이즈 : 10MB -->
80 <bean id="multipartResolver"
81     class="org.springframework.web.multipart.commons.CommonsMultipartResolver">
82     <property name="maxUploadSize" value="10485760" />
83     </bean>
84 <mvc:default-servlet-handler />
85
86 <!-- Spring mvc 사용하는데 필수적인 설정 -->
87 <mvc:view-resolvers>
88     <mvc:jsp prefix="/WEB-INF/view/" suffix=".jsp" />
89 </mvc:view-resolvers>
90 </beans>
```



## 영화예매 목록.JSP

Login을 하지 않았다면 Login.jsp로 넘기게 설정하였습니다.

```
1 <%@page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
4 <!DOCTYPE html>
5 <html>
6   <link rel="stylesheet" href="${pageContext.request.contextPath}/css/Theater.css">
7   <link rel="stylesheet" href="${pageContext.request.contextPath}/css/register.css?after">
8 </head>
9 <meta charset="UTF-8">
10 <title>Insert title here</title>
11 </head>
12 <body>
13 <c:if test="${ticket.id!=null}">
14   <jsp:include page="/Header.jsp"></jsp:include>
15 <div id="aside">
16 <table>
17   <tr>
18     <th>순위</th>
19     <th>영화</th>
20     <th>상영권</th>
21     <th>장르</th>
22     <th>시간</th>
23     <th>나이제한</th>
24   </tr>
25   <tr><!-- 첫번째 줄 시작 -->
26     <td><h1>1위<br>상전그루 영여 트릭백 </h1></td>
27     <td><a href="/movie/member/reservation/seat1.do">
28       </a></td>
29     <td><h1>3관</h1></td>
30     <td><h1>드라마</h1></td>
31     <td><h1>12:30</h1></td>
32     <td><h1>12세 이상</h1></td>
33   </tr><!-- 첫번째 줄 끝 -->
34   <tr><!-- 두번째 줄 시작 -->
35     <td><h1>2위<br>태넷</h1></td>
36     <td><a href="/movie/member/reservation/seat2.do">
37       </a></td>
38     <td><h1>1관</h1></td>
39     <td><h1>역전</h1></td>
40     <td><h1>12:30</h1></td>
41     <td><h1>12세 이상</h1></td>
42   </tr><!-- 두번째 줄 끝 -->
43   <tr><!-- 세번째 줄 시작 -->
44     <td><h1>3위<br>베이비티스</h1></td>
45     <td><a href="/movie/member/reservation/seat3.do">
46       </a></td>
47     <td><h1>2관</h1></td>
48     <td><h1>드라마</h1></td>
49     <td><h1>11:30</h1></td>
50     <td><h1>15세 이상</h1></td>
51   </tr><!-- 세번째 줄 끝 -->
52   <tr><!-- 네번째 줄 시작 -->
53     <td><h1>4위<br>소리도 없이</h1></td>
54     <td><a href="/movie/member/reservation/seat4.do">
55       </a></td>
56     <td><h1>4관</h1></td>
57     <td><h1>범죄</h1></td>
58     <td><h1>13:30</h1></td>
59     <td><h1>18세 이상</h1></td>
60   </tr><!-- 네번째 줄 끝 -->
61 </table></div>
62 <jsp:include page="/Tail.jsp"></jsp:include>
63 </c:if>
64 <c:if test="${ticket.id==null}">
65   <meta http-equiv="Refresh" content="2;url=../auth/login.do">
66   <jsp:include page="/Header.jsp"></jsp:include>
67 <div id="aside">
68 <div id="board"><br>
69   <h2>로그인이 필요한 기능입니다.</h2><br>
70   <p>이 기능은 로그인이 필요한 기능입니다. 로그인 하신 후 다시 시도해주세요. <br>
71   잠시 후 로그인 화면으로 이동합니다.</p>
72 <br><br>
73 </div>
74 </div>
75 <jsp:include page="/Tail.jsp"></jsp:include>
76 </c:if>
77 </body>
78 </html>
```

## 좌석.JSP

예매한 좌석이 있으면 x처리 된 좌석으로 설정하기 위해 jstl을 사용하였습니다.

```
1 <%@page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <%@page import="kr.kedu.movie.spms.vo.SeatVO3"%>
4 <%@page import="kr.kedu.movie.spms.vo.customerReservationVO"%>
5 <%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
6 <!DOCTYPE html>
7 <html>
8 <link rel="stylesheet" href="${pageContext.request.contextPath}/css/ScreenHeader.css">
9 <head>
10 <meta charset="UTF-8">
11 <title>3관 좌석</title>
12 </head>
13 <body>
14 <jsp:include page="/Header.jsp"></jsp:include>
15 <div id="aside">
16 <h3>원하시는 좌석 한 자리를 선택해 주세요.</h3>
17 <div id="navi">
18 
19 <h4>안내 - 사회적 거리두기 2단계로 인하여 영화관 좌석이 한자리 씩 떨어져 있습니다. 이슬에 불편을 드려 죄송합니다.</h4>
20 </div>
21 <div id="screen">3관 스크린</div>
22 <form action="complete1.do" class="ticket1">
23
24 
25 
26
27
28
29 <div id="button">
30 <:if test="${seats31.seatNumber3!=ticket31.seatNo}">
31 <input type='submit' name='seat31' value='31' onclick="location.href='../reservation/complete1.do'">
32 </c:if>
33 <:if test="${seats31.seatNumber3==ticket31.seatNo}">
34 <input type='button' value='X'>
35 </c:if>
36
37 <input type='button' value='X'>
```

# 예매.JSP

좌석 선택이 1개만 되도록 설정되어 param으로 선택 좌석을 불러옵니다.

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <%@page import="kr.kedu.movie.spms.vo.SeatV03"%>
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
5 <!DOCTYPE html>
6 <html>
7 <link rel="stylesheet" href="${pageContext.request.contextPath}/css/payment.css">
8 <head>
9 <meta charset="UTF-8">
10 <title>결제 페이지</title>
11 </head>
12 <body>
13 <jsp:include page="/Header.jsp"></jsp:include>
14 <div id="aside">
15 <div id="movieInfo">
16 <form action="completel.do" method="post">
17 <h1>예매 정보</h1>
18 <h5>회원 아이디:<input type='text' name = 'id' value='${ticket.id}' readonly></h5>
19 <c:forEach var="reservation" items="${reservations}">
20 <h5>영화 순위:<input type='text' name = 'rank' value='${reservation.movieRank}' readonly></h5>
21 <h5>영화 제목: <input type='text' name = 'movietitle' value='${reservation.title}' readonly></h5>
22 <h5>상영관: <input type='text' name = 'movieloca' value='${reservation.loca}' readonly></h5>
23 <h5>장르: <input type='text' name = 'moviegenre' value='${reservation.genre}' readonly></h5>
24 <h5>시작시간:<input type='text' name = 'restime' value= '${reservation.totime}' readonly></h5>
25 <h5>관람 연령: <input type='text' name='age' value='${reservation.grade}' readonly>세 이상</h5>
26 <h5>결제 금액: <input type='text' name='price' value='10000' readonly></h5>
27 <h5>좌석 번호 :<input type='text' name='seatNumber3' value='${param.seat31}${param.seat33}${param.seat35}${param.seat37}${param.seat39}' readonly></h5>
28
29 </c:forEach>
30
31 <h5>결제하시겠습니까?</h5>
32 <input type="submit" value ="결제" onclick="location.href='http://localhost:8080/movie/'">
33 <input type="button" value="취소" onclick="location.href='http://localhost:8080/movie/'">
34 </form>
35 </div>
36 </div>
37 <div>
38 
39 </div>
40 <jsp:include page="/Tail.jsp"></jsp:include>
41 </body>
42 </html>
```

# 영화 찾기.JSP

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
4 <!DOCTYPE html>
5
6 <html>
7 <link rel="stylesheet" href="${pageContext.request.contextPath}/css/moviesearch.css">
8 <script>
9 function toggle(val) {
10   var hobby = document.querySelectorAll(".a");
11
12   for(var i = 0; i<hobby.length; i++){
13     hobby[i].checked = val
14   }
15   form.all.checked=val;
16   /* 체크박스개제.checked = true; 체크
17     체크박스개제.checked = false; 체크 해제
18   */
19 }
20
21 function toggle1(val) {
22   var hobby1 = document.querySelectorAll(".b");
23
24   for(var i = 0; i<hobby1.length; i++){
25     hobby1[i].checked = val
26   }
27   form.all.checked=val;
28   /* 체크박스개제.checked = true; 체크
29     체크박스개제.checked = false; 체크 해제
30   */
31 }
32 function selectAll() {
33   toggle(true);
34 }
35 function selectAll1() {
36   toggle1(true);
37 }
38 function deSelectAll() {
39   toggle(false);
40 }
41 function deSelectAll1() {
42   toggle1(false);
43 }
44 </script>
45 <head>
46 <title>Insert title here</title>
47 </head>
48 <body>
49 <jsp:include page="/Header.jsp"></jsp:include>
50 <div id="aside">
51 <div class="contents">
52   <div class="finder">
```

# 영화 찾기.JSP

# 영화 찾기.JSP

제목을 치거나 체크박스를 선택 후 영화를 찾으면 해당영화가 나옵니다.

```
105 <input type="button" id="all-grade" name="all-grade" value="전체선택" onclick="selectAll()">
106 <input type="button" id="all-grade" name="all-grade" value="선택해제" onclick="deSelectAll()">
107 </li>
108 </ul>
109 </td>
110 </tr>
111 </tbody>
112 </table>
113 <div class="submit">
114 <button type="submit" class="inner" id="submit">
115 <span>검색</span>
116 </button>
117 <button type="button" onclick="location.href='${pageContext.request.contextPath}'" >
118 <span>취소</span>
119 </button>
120 </div>
121 </form>
122 </div>
123 </div>
124 <c:forEach items="${movie}" var="result">
125 <c:if test='${(param.genre0==result.genre or (param.grade0==result.grade or param.grade1==result.grade)) or
126 (param.genre1==result.genre or param.grade1==result.grade) or (param.genre2==result.genre or param.grade2==result.grade) or (param.search==result.title)
127 }'>
128 <div id="result">
129 <c:choose>
130 <c:when test='${result.movieRank==1}'>
131 <img id="photo" src='${pageContext.request.contextPath}/photo/samjin.jpg'>
132 </c:when>
133 <c:when test='${result.movieRank==2}'>
134 <img id="photo" src='${pageContext.request.contextPath}/photo/telnet.jpg'>
135 </c:when>
136 <c:when test='${result.movieRank==3}'>
137 <img id="photo" src='${pageContext.request.contextPath}/photo/baby.jpg'>
138 </c:when>
139 <c:when test='${result.movieRank==4}'>
140 <img id="photo" src='${pageContext.request.contextPath}/photo/sound.jpg'>
141 </c:when>
142 </c:choose>
143 <b>${result.title}</b><br>장르 -
144 ${result.genre},
145 ${result.grade}세 이상가
146
147 </div>
148 </c:if>
149 </c:forEach>
150 </div>
151 <div id = "footer">
152 SPMS &copy; 2020
153 </div>
154 </body>
155 </html>
```

## 회원가입.JSP

약관 동의 후 뜨는 jsp입니다. javaScript로 비어있는 값이 있거나 아이디가 4자 이상 8자 이하가 아니면 alert창이 뜹니다.

```
9 function idcheck(){
10     var mForm = document.mForm;
11     var name = mForm.name.value;
12     var id = mForm.id.value;
13     var pass = mForm.password.value;
14     var res = mForm.residentNum.value;
15     var phone = mForm.phoneNum.value;
16
17     if (!id) {
18         alert("아이디를 입력해주세요.");
19         mForm.id.focus();
20         return false;
21     } else if (id.length < 4 || id.length > 8) {
22         mForm.id.focus();
23         alert("아이디는 4자 이상 8자 이하여야합니다.");
24         return false;
25     }
26     if (!pass) {
27         alert("비밀번호를 입력해주세요.");
28         mForm.pass.focus();
29         return false;
30     }
31     if (!name) {
32         alert("이름을 입력해주세요.");
33         mForm.name.focus();
34         return false;
35     }
36     if (!res) {
37         alert("주민등록번호를 입력해주세요.");
38         mForm.res.focus();
39         return false;
40     }
41     if (!phone) {
42         alert("핸드폰 번호를 입력해주세요. (-제외)");
43         mForm.phone.focus();
44         return false;
45     }
46 }
47 </script>
48 <link href="form.css" rel="stylesheet" type="text/css">
49 <meta charset="UTF-8">
50 <title>회원가입 화면</title>
51 </head>
52 <body>
53     <jsp:include page="/Header.jsp"></jsp:include>
54 <div id="aside"><div id="board">
55     <h1>회원 등록</h1>
56 <form name="mForm" action="add.do" method="post" onsubmit="idcheck()">
57     ID: <input type="text" size="10" maxlength="15" name="id"><br><br>
58     PW: <input type="password" size="10" maxlength="15" name="password"><br><br>
59     NAME: <input type="text" size="10" maxlength="10" name="name"><br><br>
60     RES(-제외): <input type="password" size="10" maxlength="13" name="residentNum"><br><br>
61     PHN: <input type="text" size="10" maxlength="15" name="phoneNum"><br><br>
```

## 로그인.JSP

로그인 에 성공하면 post방식의 컨트롤러가 실행되어 login controller에서 아이디 비밀번호를 판독합니다. 로그인 실패 시 오른쪽에 있는 jsp가 실행됩니다.

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <link rel="stylesheet" href="${pageContext.request.contextPath}/css/register.css?after">
6 <head>
7 <meta charset="UTF-8">
8 <title>로그인</title>
9 </head>
10 <body>
11     <jsp:include page="/Header.jsp"></jsp:include>
12 <div id="aside">
13 <div id="board"><br>
14 <h2>회원 로그인</h2><br>
15 <form action = "login.do" method = "post">
16 아이디: <input type="text" name="id" required><br>
17 암호: <input type="password" name="password" required><br><br>
18 <input type="submit" value = "로그인" >
19 <input type="button" value = "취소" onclick="location='../'">
20 <br>
21 <br><br><br>
22 </form></div>
23 </div>
24 <jsp:include page="/Tail.jsp"></jsp:include>
25 </body>
26 </html>
```

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <link rel="stylesheet" href="${pageContext.request.contextPath}/css/register.css?after">
6 <head>
7 <meta charset="UTF-8">
8 <meta http-equiv="Refresh" content="2;url=login.do">
9 <title>로그인 실패</title>
10 </head>
11 <body>
12     <jsp:include page="/Header.jsp"></jsp:include>
13 <div id="aside">
14 <div id="board"><br>
15 <h2>로그인 실패!</h2><br>
16
17 <p>로그인에 실패하였습니다. 아이디 또는 암호가 맞지 않습니다. <br>
18 잠시후 다시 로그인 화면으로 돌아갑니다.</p>
19 <br><br>
20 </div>
21 </div>
22 <jsp:include page="/Tail.jsp"></jsp:include>
23
24 </body>
25 </html>
```



## DTO(MEMBER VO)

영화목록에 대한 vo입니다.

```
1 package kr.kedu.movie.spms.vo;
2
3 import kr.kedu.movie.spms.vo.MemberVO;
4
5 public class MemberVO {
6     private int movieRank;
7     private String title;
8     private String loca;
9     private String genre;
10    private String totime;
11    private String grade;
12    private String location;
13    private String time;
14
15    public int getmovieRank() {
16        return movieRank;
17    }
18    public MemberVO setmovieRank(int movieRank) {
19        this.movieRank = movieRank;
20        return this;
21    }
```

## SEAT VO 1,2,3,4

좌석 VO 입니다.

```
1 package kr.kedu.movie.spms.vo;
2
3 public class SeatV01 {
4     private int seatNumber1;
5     public int getSeatNumber1() {
6         return seatNumber1;
7     }
8     public SeatV01 setSeatNumber1(int seatNumber1) {
9         this.seatNumber1 = seatNumber1;
10        return this;
11    }
12 }
13
14 package kr.kedu.movie.spms.vo;
15
16 public class SeatV02 {
17     private int seatNumber2;
18     public int getSeatNumber2() {
19         return seatNumber2;
20     }
21     public SeatV02 setSeatNumber2(int seatNumber2) {
22         this.seatNumber2 = seatNumber2;
23         return this;
24     }
25 }
26 }
```

```
1 package kr.kedu.movie.spms.vo;
2
3 public class SeatV03 {
4     private int seatNumber3;
5     public int getSeatNumber3() {
6         return seatNumber3;
7     }
8     public SeatV03 setSeatNumber3(int seatNumber3) {
9         this.seatNumber3 = seatNumber3;
10        return this;
11    }
12 }
13
14 package kr.kedu.movie.spms.vo;
15
16 public class SeatV04 {
17     private int seatNumber4;
18     public int getSeatNumber4() {
19         return seatNumber4;
20     }
21     public SeatV04 setSeatNumber4(int seatNumber4) {
22         this.seatNumber4 = seatNumber4;
23         return this;
24     }
25 }
26 }
```

# CUSTOMER VO

회원 VO 입니다.

```
1 package kr.kedu.movie.spms.vo;
2
3 import kr.kedu.movie.spms.vo.CustomerVO;
4
5 public class CustomerVO {
6     private int uid;
7     private String id;
8     private String name;
9     private String residentNum;
10    private String phoneNum;
11    private String password;
12
13    public String getName() {
14        return name;
15    }
16    public int getUid() {
17        return uid;
18    }
19    public CustomerVO setUid(int uid) {
20        this.uid = uid;
21        return this;
22    }
23    public CustomerVO setName(String name) {
24        this.name = name;
25        return this;
26    }
27    public String getId() {
28        return id;
29    }
30    public CustomerVO setId(String id) {
31        this.id = id;
32        return this;
33    }
34    public String getResidentNum() {
35        return residentNum;
36    }
37    public CustomerVO setResidentNum(String residentNum) {
38        this.residentNum = residentNum;
39        return this;
40    }
41    public String getPhoneNum() {
42        return phoneNum;
43    }
44    public CustomerVO setPhoneNum(String phoneNum) {
45        this.phoneNum = phoneNum;
46        return this;
47    }
48    public String getPassword() {
49        return password;
50    }
51    public CustomerVO setPassword(String password) {
52        this.password = password;
53        return this;
54    }
55 }
```

## CUSTOMER RESERVATION VO

예매 후 데이터 값을 저장 할 VO입니다.

```
1 package kr.kedu.movie.spms.vo;
2
3 public class customerReservationVO {
4     private String resid;
5     private int movieRank2;
6     private String title2;
7     private String loca2;
8     private String genre2;
9     private String totime2;
10    private int grade2;
11    private int price;
12    private int seatNo;
13
14
15    public int getMovieRank2() {
16        return movieRank2;
17    }
18    public customerReservationVO setMovieRank2(int movieRank2) {
19        this.movieRank2 = movieRank2;
20        return this;
21    }
```

## DAO

Mapper에서 sql문을 사용한 값을 dao에서 sqlSessionFactory 를 사용하여 Sql문을 실행시킬 준비를 하였습니다.

Repository annotation을 사용하여 memberDAO를 controller에서 호출 시 값을 전달하는 역할을 합니다.

Insert, select, selectOne, delete 기능이 있습니다.

```
@Repository("memberDAO")
public class MemberDAO {

    @Autowired
    SqlSessionFactory sqlSessionFactory;

    public void setSqlSessionFactory(SqlSessionFactory sqlSessionFactory) {
        this.sqlSessionFactory = sqlSessionFactory;
    }
}
```

# CONTROLLER

예약목록 controller 입니다. 예매 목록.jsp로 전달합니다.

예매목록.jsp -> 예약목록 controller -> dao -> mapper -> vo 로 진행됩니다.

```
1 package kr.kedu.movie.spms.controls;
2 import org.springframework.beans.factory.annotation.Autowired;
3 import org.springframework.stereotype.Controller;
4 import org.springframework.web.bind.annotation.RequestMapping;
5 import org.springframework.web.bind.annotation.RequestMethod;
6 import org.springframework.web.servlet.ModelAndView;
7 import kr.kedu.movie.spms.dao.MemberDAO;
8 import kr.kedu.movie.spms.service.MemberService;
9
10 @Controller
11 @RequestMapping("member")
12 public class MemberMovieReservationController implements SpmsController {
13
14     MemberDAO memberDAO;
15     @Autowired
16     MemberService service;
17
18
19     @Autowired
20     public MemberMovieReservationController setMemberDAO(MemberDAO memberDAO) {
21         this.memberDAO = memberDAO;
22         return this;
23     }
24
25     @RequestMapping(value="/reservation.do", method=RequestMethod.GET)
26     public ModelAndView execute() throws Exception {
27         System.out.println("성공");
28         ModelAndView mav =new ModelAndView("ticket/theaterForm");
29         return mav;
30     }
31
32     @Override
33     public void setService(MemberService service) throws Exception {
34         this.service = service;
35     }
36
37 }
38
39
40
41
```

# CONTROLLER

좌석 controller 입니다. 예매 목록.jsp에서 예매하고 싶은 영화 사진을 클릭 시 넘겨받는 페이지입니다. ticketSelectloca는 dao에서 만든 이미 결제한 좌석 값들을 호출한 것이고 seatOne은 dao에서 지정된 좌석번호들을 호출한 것입니다.

좌석.JSP 는 4개로 seat1.do, seat2.do, seat3.do, seat4.do 4개 controller로 찾아갑니다. ->dao ->mapper -> vo

```
@Controller
@RequestMapping("/member/reservation")
public class ReservationSeatController {
    MemberVO memberVO;
    MemberDAO memberDAO;

    @Autowired
    public ReservationSeatController setMemberDAO(MemberDAO memberDAO) {
        this.memberDAO = memberDAO;
        return this;
    }

    @RequestMapping(value="/seat1.do",method =RequestMethod.GET)
    public ModelAndView execute1() throws Exception {
        ModelAndView mav = new ModelAndView("ticket/Seat");
        mav.addObject("ticket31",memberDAO.ticketSelectloca31(1));
        mav.addObject("ticket33",memberDAO.ticketSelectloca33(3));
        mav.addObject("ticket35",memberDAO.ticketSelectloca35(5));
        mav.addObject("ticket37",memberDAO.ticketSelectloca37(7));
        mav.addObject("ticket39",memberDAO.ticketSelectloca39(9));
        mav.addObject("seats31",memberDAO.seatOne31(1));
        mav.addObject("seats33",memberDAO.seatOne33(3));
        mav.addObject("seats35",memberDAO.seatOne35(5));
        mav.addObject("seats37",memberDAO.seatOne37(7));
        mav.addObject("seats39",memberDAO.seatOne39(9));

        return mav;
    }
}
```

# CONTROLLER

예매 controller입니다. 좌석.jsp에서 예매한 영화목록과 좌석 데이터 값을 받아 get방식으로 저장하고 post 방식에서 입력 받을 값들을 전부 customerReservationVO 객체로 받습니다.

여기서 저장된 값은 DB ticket table 로 insert 됩니다.

```
1 package kr.kedu.movie.spms.controllers;
2
3 import java.util.Map;
4
5 @Controller
6 @RequestMapping("member")
7 public class SeatPaymentController implements SpmsController {
8     MemberVO memberVO;
9     MemberDAO memberDAO;
10
11     @Autowired
12     MemberService service;
13
14     @Autowired
15     public SeatPaymentController setMemberDAO(MemberDAO memberDAO) {
16         this.memberDAO = memberDAO;
17         return this;
18     }
19
20     @RequestMapping(value="/reservation/complete1.do", method=RequestMethod.GET)
21     public ModelAndView execute1() throws Exception {
22         ModelAndView mav = new ModelAndView("ticket/payment");
23         mav.addObject("reservations", memberDAO.selectList1());
24         mav.addObject("seats31", memberDAO.seatOne31());
25         mav.addObject("seats33", memberDAO.seatOne33());
26         mav.addObject("seats35", memberDAO.seatOne35());
27         mav.addObject("seats37", memberDAO.seatOne37());
28         mav.addObject("seats39", memberDAO.seatOne39());
29         return mav;
30     }
31
32     @RequestMapping(value="/reservation/complete1.do", method =RequestMethod.POST)
33     public ModelAndView execute11(@RequestParam("id") String id,
34         @RequestParam("rank")int rank,
35         @RequestParam("movietitle")String title,
36         @RequestParam("movieLoca")String loca,
37         @RequestParam("moviegenre")String genre,
38         @RequestParam("restime")String restime,
39         @RequestParam("age")int age,
40         @RequestParam("price")int price,
41         @RequestParam("seatNumber3")int seat) throws Exception {
42         System.out.println("예약완료");
43         ModelAndView mav = new ModelAndView("redirect:/");
44         customerReservationVO crvo=new CustomerReservationVO().setResid(id)
45             .setMovieRank2(rank).setTitle2(title)
46             .setLoca2(loca).setGenre2(genre)
47             .setTotime2(restime).setGrade2(age)
48             .setPrice(price).setSeatNo(seat);
49         memberDAO.insert01(crvo);
50         return mav;
51     }
52 }
```



# CONTROLLER

영화 세부목록을 찾기 위한 controller입니다.

영화찾기.JSP -> controller -> dao -> mapper -> vo 순으로 진행됩니다.

```
1 package kr.kedu.movie.spms.controls;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4
5
6
7
8
9
10
11
12
13 @Controller
14 @RequestMapping("/member")
15 public class MovieSearchController {
16
17     @Autowired
18     MemberDAO memberDAO;
19
20     @Autowired
21     public MovieSearchController setMemberDAO
22         (MemberDAO memberDAO) {
23         this.memberDAO = memberDAO;
24         return this;
25     }
26     @RequestMapping(value="/list.do", method=RequestMethod.GET)
27     public ModelAndView doGet() throws Exception {
28
29         ModelAndView mav = new ModelAndView("/member/moviesearch");
30
31
32         |
33         return mav;
34     }
35     @RequestMapping(value="/list.do", method=RequestMethod.POST)
36     public ModelAndView doPost() throws Exception {
37
38         ModelAndView mav = new ModelAndView("/member/moviesearch");
39
40         mav.addObject("movie",memberDAO.selectListTest());
41
42         return mav;
43     }
44 }
45
```

# CONTROLLER

로그인, 로그아웃 Controller입니다. Sql에 저장된 customer 아이디와 비밀번호가 일치해야 로그인 이 되도록 설정하였습니다. 로그아웃은 클릭 시 로그인 으로 경로를 설정하였습니다.

로그인.jsp -> 로그인 controller -> dao -> mapper -> vo 로 진행됩니다.

```
1 package kr.kedu.movie.spms.controls;
2 import javax.servlet.http.HttpSession;
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.web.bind.annotation.GetMapping;
5 import org.springframework.web.bind.annotation.PostMapping;
6 import org.springframework.web.servlet.ModelAndView;
7 import kr.kedu.movie.spms.dao.MemberDAO;
8 import kr.kedu.movie.spms.service.MemberService;
9 import kr.kedu.movie.spms.vo.CustomerVO;
10
11
12 @org.springframework.stereotype.Controller
13 public class LoginController implements SpmsController {
14     MemberDAO memberDAO;
15
16     @Autowired
17     MemberService service;
18
19     @Autowired
20     public LoginController setMemberDAO(MemberDAO memberDAO) {
21         this.memberDAO = memberDAO;
22         return this;
23     }
24
25     @GetMapping("/auth/login.do")
26     public ModelAndView doGet() throws Exception {
27         return new ModelAndView("/auth/LoginForm");
28     }
29
30     @PostMapping("/auth/login.do")
31     public ModelAndView doPost(CustomerVO customer, HttpSession request) throws Exception {
32         CustomerVO login = memberDAO.exist(customer.getId(), customer.getPassword());
33         if(login != null) {
34             ModelAndView mav = new ModelAndView("redirect:/");
35             request.setAttribute("ticket", login);
36             return mav;
37         } else {
38             return new ModelAndView("/auth/LoginFail");
39         }
40     }
41
42 }
43
44 @Override
45 public void setService(MemberService service) throws Exception {
46     this.service = service;
47 }
48 }
```

```
1 package kr.kedu.movie.spms.controls;
2
3 import java.util.Map;
4
5 @org.springframework.stereotype.Controller
6 public class LogoutController implements SpmsController {
7     MemberDAO memberDAO;
8
9     @Autowired
10    MemberService service;
11
12    public LogoutController setMemberDAO
13        (MemberDAO memberDAO) {
14        this.memberDAO = memberDAO;
15        return this;
16    }
17
18    @GetMapping("/auth/logout.do")
19    public ModelAndView doGet(HttpSession session) throws Exception {
20        session.invalidate();
21        return new ModelAndView("/auth/LoginForm");
22    }
23
24    public String execute(Map<String, Object> model) throws Exception {
25        return null;
26    }
27
28    @Override
29    public void setService(MemberService service) throws Exception {
30        // TODO Auto-generated method stub
31        this.service = service;
32    }
33
34 }
```

# CONTROLLER

회원가입 컨트롤러 입니다. 회원가입 텍스트 창에 입력 받은 데이터를  
dao -> mapper -> vo로 전달하여 줍니다.

```
@Controller
public class MemberAddController {
    MemberDAO memberDAO;

    @Autowired
    public MemberAddController setMemberDAO(MemberDAO memberDAO) {
        this.memberDAO = memberDAO;
        return this;
    }

    @GetMapping("/member/add.do")
    public ModelAndView doGet() throws Exception {
        return new ModelAndView("/member/MemberForm");
    }

    @PostMapping("/member/add.do")
    public ModelAndView doPost(@RequestParam("id") String id, @RequestParam("name") String name,
        @RequestParam("password") String password, @RequestParam("residentNum") String residentNum,
        @RequestParam("phoneNum") String phoneNum) throws Exception {

        ModelAndView mav = new ModelAndView("redirect:/");

        CustomerVO ticket = new CustomerVO().setId(id).setName(name).setPassword(password).setResidentNum(residentNum)
            .setPhoneNum(phoneNum);
        memberDAO.insert(ticket);

        return mav;
    }

    public String execute(Map<String, Object> model) throws Exception {
        if (model.get("member") == null) {
            return "/member/MemberForm.jsp";
        } else {
            CustomerVO member = (CustomerVO) model.get("member");
            int i = memberDAO.insert(member);
            return "redirect:list.do";
        }
    }
}
```

A decorative floral pattern in a dark gray color, featuring stylized leaves and flowers, is positioned at the top of the image.

이상입니다.

봐주셔서 감사합니다.