

Spring Web MVC

MVC : Model, View, Controller의 약자입니다.

현재 가장 많이 사용하는 개발 패턴으로 서비스를 위한 각 부분을 분리해 만들고 이를 통해 개발 및 유지 보수의 효율성을 높이는 것이 목적입니다.

- Model : 데이터를 관리하는 부분
- View : 눈에 보이는 부분을 구성하는 부분
- Controller : 요청에 따른 코드 흐름을 제어하는 부분

(1)	SpringConfigClass { 환경설정클래스 -ServletAppContext 환경Root 빈 정의 -RootAppContext }	: AbstractAnnotationConfigDispatcherServletInitializer 상속
(2)	@Configuration ServletAppContext { @ComponentScan("kr.co.korea.controller") @ComponentScan("kr.co.korea.dao") @ComponentScan("kr.co.korea.service") @PropertySource("/WEB-INF/properties/db.properties") @Bean -db접속,쿼리접속,쿼리실행 }	: WebMvcConfigurer 인터페이스 상속
(3)	@Configuration RootAppContext { @Bean("loginUserBean") @SessionScope }	

(1) @Bean : 메서드를 통해 반환하는 객체를 Bean으로 등록

ex) [kr.co.korea.config]패키지의 ServletAppContext 클래스의 DB접속정보, 쿼리문, mapper

(2) @Component : 개발자가 만든 클래스의 객체를 생성하여 Bean으로 등록

ex) [kr.co.korea.beans]패키지의 DataBean 클래스의 객체를 생성하여 등록할때

(3) @Controller : Component의 일종으로 사용자 요청에 따라 자동으로 호출되는 메서드를 가지고 있는 Bean을 등록. jsp로 리턴 ex) [kr.co.korea.controller]패키지의 HomeController클래스, TestController클래스

(4) @RestController : Component의 일종으로 사용자 요청에 따라 자동으로 호출되는 메서드를 가지고 있는 Bean을 등록. Restful API 서버 구성 시 사용. 전달 글자 그대로 리턴

(5) @ControllerAdvice : 예외가 발생했을 때 사용할 Global Exception Handler로 사용할 Bean을 등록

ex) [kr.co.korea.exception]패키지의 GlobalException 클래스 만들때...

(6) @Repository : dao에서 @Repository로 클래스를 빈 정의 해두고 service의 @Service로 정의한 Bean에서 @Autowired 로 주입받아 사용. 이 Bean은 데이터베이스와 관련된 작업을 구현. @Component로 정의한 Bean과 차이가 없음.

ex) [kr.co.korea.dao]패키지 TopMenuDao클래스 만들때

(7) @Service : service에서 빈 정의 해두고 Controller에서 @Autowired 이용하여 주입받아 사용.
@Component로 정의한 Bean과 차이는 없음. ex) [kr.co.korea.service]패키지의 UserService클래스,
TopMenuService클래스 만들때...

(8) @Configuration : 설정클래스임을 지정

ex) [kr.co.korea.config]패키지의 ServletApplicationContext.java에서 설정 클래스임을 지정

(9) @EnableWebMvc : 설정클래스에서 Spring MVC환경을 구성할 수 있음

ex) [kr.co.korea.config]패키지의 ServletApplicationContext.java

(10) @ComponentScan("패키지") : 스캔할 패키지 지정

ex) [kr.co.korea.config]패키지의 ServletApplicationContext.java 에서

(11) @PropertySource(), @PropertySources() : properties파일을 주입받기

ex) [kr.co.korea.config]패키지의 ServletApplicationContext.java

(12) @Value : properties파일의 값을 주입받기

ex) [kr.co.korea.config]패키지의 ServletApplicationContext.java

1. Java방법으로 스프링 설정

(1) project.sql

```
create sequence user_seq
start with 0
increment by 1
minvalue 0;

create sequence content_seq
start with 0
increment by 1
minvalue 0;

create table board_info_table(
    board_info_idx number constraint BOARD_INFO_PK primary key,
    board_info_name varchar2(500) not null
);

create table user_table(
    user_idx number constraint USER_PK primary key,
    user_name varchar2(50) not null,
    user_id varchar2(100) not null,
    user_pw varchar2(100) not null
);

create table content_table(
    content_idx number constraint CONTENT_PK primary key,
    content_subject varchar2(500) not null,
    content_text long not null,
    content_file varchar2(500),
    content_writer_idx number not null
        constraint CONTENT_FK1 references user_table(user_idx),
    content_board_idx number not null
        constraint CONTENT_FK2 references board_info_table(board_info_idx),
    content_date date not null
);
```

```

insert into board_info_table(board_info_idx, board_info_name) values (1, '1자유게시판');
insert into board_info_table(board_info_idx, board_info_name) values (2, '2게시판');
insert into board_info_table(board_info_idx, board_info_name) values (3, '3게시판');
insert into board_info_table(board_info_idx, board_info_name) values (4, '4게시판');

commit;

```

게시판 정보 테이블(board_info_table)

필드명(논리)	필드명	타입	NULL	PK	FK	UQ
게시판 번호	board_info_idx	number	X	O	X	O
게시판 이름	board_info_name	varchar2(500)	X	X	X	X

사용자 정보 테이블(user_table)

필드명(논리)	필드명	타입	NULL	PK	FK	UQ
사용자 번호	user_idx	number	X	O	X	O
사용자 이름	user_name	varchar2(50)	X	X	X	X
사용자 아이디	user_id	varchar2(100)	X	X	X	O
사용자 비밀번호	user_pw	varchar2(100)	X	X	X	X

게시글 테이블(content_table)

필드명(논리)	필드명	타입	NULL	PK	FK	UQ
게시글 인덱스	content_idx	number	X	O	X	O
게시글 제목	content_subject	varchar2(500)	X	X	X	X
게시글 내용	content_text	long	X	X	X	X
첨부파일	content_file	varchar2(500)	O	X	X	X
게시글 작성자 인덱스	content_writer_idx	number	X	X	user_table(user_idx)	X
게시판 인덱스	content_board_idx	number	X	X	board_info_table(board_info_idx)	X
작성날짜	content_date	date	X	X	X	X

(2) pom.xml

```

<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
    https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>minProjectJava</groupId>
  <artifactId>minProjectJava</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>war</packaging>
  <build>
    <sourceDirectory>src</sourceDirectory>
    <plugins>
      <plugin>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>3.8.1</version>
        <configuration>
          <source>1.8</source>
          <target>1.8</target>
        </configuration>
      </plugin>
      <plugin>
        <artifactId>maven-war-plugin</artifactId>
        <version>3.2.3</version>
      </plugin>
    </plugins>
  </build>
</project>

```

```

        <configuration>
            <warSourceDirectory>WebContent</warSourceDirectory>
        </configuration>
    </plugin>
</plugins>
</build>
<!-- 라이브러리 버전관리 -->
<properties>
    <javax.servlet-version>4.0.1</javax.servlet-version>
    <javax.servlet.jsp-version>2.3.3</javax.servlet.jsp-version>
    <javax.servlet-jstl-version>1.2</javax.servlet-jstl-version>
    <org.springframework-version>5.2.8.RELEASE</org.springframework-version>
</properties>

<!-- 라이브러리 셋팅 -->
<dependencies>
    <!-- https://mvnrepository.com/artifact/javax.servlet/javax.servlet-api -->
    <dependency>
        <groupId>javax.servlet</groupId>
        <artifactId>javax.servlet-api</artifactId>
        <version>${javax.servlet-version}</version>
        <scope>provided</scope>
    </dependency>
    <!-- https://mvnrepository.com/artifact/javax.servlet.jsp/javax.servlet.jsp-api -->
    <dependency>
        <groupId>javax.servlet.jsp</groupId>
        <artifactId>javax.servlet.jsp-api</artifactId>
        <version>${javax.servlet.jsp-version}</version>
        <scope>provided</scope>
    </dependency>
    <!-- https://mvnrepository.com/artifact/javax.servlet/jstl -->
    <dependency>
        <groupId>javax.servlet</groupId>
        <artifactId>jstl</artifactId>
        <version>${javax.servlet-jstl-version}</version>
    </dependency>
    <!-- https://mvnrepository.com/artifact/org.springframework/spring-webmvc -->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-webmvc</artifactId>
        <version>${org.springframework-version}</version>
    </dependency>

    <dependency>
        <groupId>javax.validation</groupId>
        <artifactId>validation-api</artifactId>
        <version>2.0.1.Final</version>
    </dependency>

    <dependency>
        <groupId>org.hibernate.validator</groupId>
        <artifactId>hibernate-validator</artifactId>
        <version>6.1.2.Final</version>
    </dependency>
    <!-- https://mvnrepository.com/artifact/org.springframework/spring-jdbc -->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-jdbc</artifactId>
        <version>5.2.8.RELEASE</version>
    </dependency>
    <!-- https://mvnrepository.com/artifact/org.apache.commons/commons-dbcp2 -->
    <dependency>
        <groupId>org.apache.commons</groupId>
        <artifactId>commons-dbcp2</artifactId>
        <version>2.7.0</version>
    </dependency>

    <!-- https://mvnrepository.com/artifact/org.mybatis/mybatis-spring -->
    <dependency>
        <groupId>org.mybatis</groupId>
        <artifactId>mybatis-spring</artifactId>
        <version>2.0.5</version>
    </dependency>
    <!-- https://mvnrepository.com/artifact/org.mybatis/mybatis -->
    <dependency>

```

```
        <groupId>org.mybatis</groupId>  
        <artifactId>mybatis</artifactId>  
        <version>3.5.5</version>  
    </dependency>  
  
</dependencies>  
  
</project>
```

1. mybatis 개념

(1) mapper작성:쿼리문

[kr.co.korea.mapper] UserMapper.java 쿼리문 작성

쿼리문작성 (java방식)	<pre>public interface UserMapper { @Select("select user_idx, user_name " + "from user_table " + "where user_id=#{user_id} and user_pw=#{user_pw}") UserBean getLoginUserInfo(UserBean tempLoginUserBean); }</pre>
쿼리문작성 (xml방식)	<pre><mapper namespace="user"> <select id="getLoginUserInfo" parameterType="kr.co.korea.beans.UserBean" resultType="kr.co.korea.beans.UserBean"> <![CDATA[select user_idx, user_name from user_table where user_id=#{user_id} and user_pw=#{user_pw}]]> </select> </mapper></pre>

(2) dao작성:mapper와 직접적으로 연결해주는 메소드정의

[kr.co.korea.dao] UserDao.java

Dao (java방식)	<pre>@Repository public class UserDao { @Autowired private UserMapper userMapper; public UserBean getLoginUserInfo(UserBean tempLoginUserBean) { return userMapper.getLoginUserInfo(tempLoginUserBean); } }</pre>
Dao (xml방식)	<pre>@Repository public class UserDao { @Autowired private SqlSessionTemplate sqlSessionTemplate; public UserBean getLoginUserInfo(UserBean tempLoginUserBean) { return sqlSessionTemplate.selectOne("user.getLoginUserInfo", tempLoginUserBean); } }</pre>

(3) service작성:dao의 메소드 가져와서 새로운 변형을 위한 메소드정의

[kr.co.korea.service] UserService.java

Service	<pre>@Service public class UserService { @Autowired private UserDao userDao; //UserDao의 getLoginUserInfo() 호출하도록 정의 public void getLoginUserInfo(UserBean tempLoginUserBean) { UserBean tempLoginUserBean2 = userDao.getLoginUserInfo(tempLoginUserBean); if(tempLoginUserBean2 != null) { //db에서 select된게 있으면 loginUserBean.setUser_idx(tempLoginUserBean2.getUser_idx()); loginUserBean.setUser_name(tempLoginUserBean2.getUser_name()); loginUserBean.setUserLogin(true); } } }</pre>
---------	---

(4) controller작성:service의 메소드 가져와서 jsp파일(views)로 이동하는 메소드 정의

[kr.co.korea.controller] UserController.java

Controller	<pre> @Controller @RequestMapping("/user") public class UserController { </pre>
	<pre> //userService 객체로 주입 @Autowired private UserService userService; @PostMapping("/login_pro") public String login_pro(@Valid @ModelAttribute("tempLoginUserBean") UserBean tempLoginUserBean, BindingResult result) { if(result.hasErrors()) { return "user/login"; } userService.getLoginUserInfo(tempLoginUserBean); if(loginUserBean.isUserLogin() == true) { return "user/login_success"; } else { return "user/login_fail"; } } } } </pre>

(1) java 방식 설정

1) db.properties

[WebContent-WEB-INF-properties폴더]

```

db.classname=oracle.jdbc.OracleDriver
db.url=jdbc:oracle:thin:@localhost:1521:xe
db.username=system
db.password=123456

```

2) BoardMapper.java 인터페이스

[kr.co.korea.mapper]

```

package kr.co.korea.mapper;

public interface BoardMapper {

}

```

3) ServletAppContext.java

```

//1)
@PropertySource("/WEB-INF/properties/db.properties")
public class ServletAppContext implements WebMvcConfigurer {

//2)
    @Value("${db.classname}")
    private String db_classname;

    @Value("${db.url}")
    private String db_url;

    @Value("${db.username}")
    private String db_username;

    @Value("${db.password}")
    private String db_password;

//3) 데이터베이스 접속 정보 관리 (org.apache.commons.dbcp2.BasicDataSource)
    @Bean
    public BasicDataSource dataSource() {
        BasicDataSource source = new BasicDataSource();
        source.setDriverClassName(db_classname);
        source.setUrl(db_url);
    }
}

```

```

        source.setUsername(db_username);
        source.setPassword(db_password);
        // System.out.println("db connect");
        return source;
    }

//4) 쿼리문과 접속 관리하는 객체
@Bean
public SqlSessionFactory factory(BasicDataSource source) throws Exception {
    SqlSessionFactoryBean factoryBean = new SqlSessionFactoryBean();
    factoryBean.setDataSource(source);
    SqlSessionFactory factory = factoryBean.getObject();
    // System.out.println("sql");
    return factory;
}

//5) 쿼리문 실행을 위한 객체 (BoardMapper생성할것)
@Bean
public MapperFactoryBean<BoardMapper> getBoardMapper(SqlSessionFactory factory) throws
Exception {
    MapperFactoryBean<BoardMapper> factoryBean = new
MapperFactoryBean<BoardMapper>(BoardMapper.class);
    factoryBean.setSqlSessionFactory(factory);
    return factoryBean;
}

```

(2) xml 방식 설정

1) db.properties

[WebContent-WEB-INF-properties폴더]

```

db.classname=oracle.jdbc.OracleDriver
db.url=jdbc:oracle:thin:@localhost:1521:xe
db.username=system
db.password=123456

```

2) board_mapper.xml

[WebContent-WEB-INF-mapper폴더]

```

<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="board">

</mapper>

```

3) servlet-context.xml

[WebContent-WEB-INF-config폴더]

```

<!--1) properties파일의 내용을 사용할 수 있도록 Bean를 정의 -->
<beans:bean class='org.springframework.beans.factory.config.PropertyPlaceholderConfigurer'>
    <beans:property name="location">
        <beans:value>/WEB-INF/properties/db.properties</beans:value>
    </beans:property>
</beans:bean>

<!--2) -->
<beans:bean class='org.apache.commons.dbcp2.BasicDataSource' id='basic_data_source'>
    <beans:property name='driverClassName' value='${db.classname}'/>
    <beans:property name='url' value='${db.url}'/>
    <beans:property name='username' value='${db.username}'/>
    <beans:property name='password' value='${db.password}'/>
</beans:bean>

<!--3) -->
<beans:bean class='org.mybatis.spring.SqlSessionFactoryBean' id='sqlSession'>

```



```
        <beans:property name="dataSource" ref='basic_data_source'/>
        <beans:property name="mapperLocations" value= '/WEB-INF/mapper/*.xml'/>
    </beans:bean>

<!--4) -->
    <beans:bean class= 'org.mybatis.spring.SqlSessionTemplate' id= 'sqlSessionTemplate'>
        <beans:constructor-arg index= '0' ref= 'sqlSession'/>
    </beans:bean>
```

1. 회원가입

(1) 회원가입-저장처리

The screenshot shows a web browser window with the URL 'http://k-hrd.kr/'. The page has a dark header with navigation links: 'KMOVE1', '자유게시판', '유머게시판', '정치게시판', '스포츠게시판', '로그인', '회원가입', '정보수정', and '로그아웃'. The '회원가입' link is highlighted with a pink box. Below the header is a registration form, also highlighted with a pink box. The form contains the following fields and buttons:

- 이름 (Name): Text input field.
- 아이디 (ID): Text input field with a '중복확인' (Check Duplicate) button next to it.
- 비밀번호 (Password): Text input field.
- 비밀번호 확인 (Confirm Password): Text input field with a '회원가입' (Sign Up) button next to it.

1)-1 UserMapper.java -(mapper에서 쿼리문 만들기) -(Java방식)
[kr.co.korea.mapper]

```
package kr.co.korea.mapper;

import org.apache.ibatis.annotations.Insert;
import org.apache.ibatis.annotations.Select;

import kr.co.korea.beans.UserBean;

public interface UserMapper {

    @Select("select user_name "+
            "from user_table "+
            "where user_id= #{user_id}")
    String checkUserIdExist(String user_id);

    //(1) 회원가입-저장처리(insert into)
    @Insert("insert into user_table (user_idx, user_name, user_id, user_pw) " +
            "values (user_seq.nextval, #{user_name}, #{user_id}, #{user_pw})")
    void addUserInfo(UserBean joinUserBean);

}
```

1)-2 user_mapper.xml -(mapper에서 쿼리문 만들기) -(xml방식)
[WebContent-WEB-INF-mapper폴더]

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
    "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="user">
    <select id="checkUserIdExist" parameterType="java.lang.String" resultType="java.lang.String">
        <![CDATA[
            select user_name
            from user_table
            where user_id = #{user_id}
        ]]>
    </select>

    <!--(1) mapper -->
    <insert id="addUserInfo" parameterType="kr.co.softcampus.beans.UserBean">
        <![CDATA[
            insert into user_table (user_idx, user_name, user_id, user_pw)
            values (user_seq.nextval, #{user_name}, #{user_id}, #{user_pw})
        ]]>
    </insert>
</mapper>
```

```

    </insert>
  }
</mapper>
]]>

```

2)-1 UserDao.java -(dao에서 mapper의 addUserInfo() 정의해두기) -(Java방식)

[kr.co.korea.dao]

```

@Repository
public class UserDao {

    @Autowired
    private UserMapper userMapper;
    public String checkUserIdExist(String user_id) {
        return userMapper.checkUserIdExist(user_id);
    }

    //(2) mapper의 addUserInfo 호출하도록 addUserInfo()메소드 정의
    public void addUserInfo(UserBean joinUserBean) {
        userMapper.addUserInfo(joinUserBean); //mapper의 addUserInfo호출
    }
}

```

2)-2 UserDao.java -(dao에서 mapper.xml파일내의 “user.addUserInfo” 삽입명령 호출하도록 addUserInfo()메소드 정의해두기) -(xml방식)

[kr.co.korea.dao]

```

package kr.co.korea.dao;

import org.mybatis.spring.SqlSessionTemplate;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;

import kr.co.korea.beans.UserBean;
@Repository
public class UserDao {

    @Autowired
    private SqlSessionTemplate sqlSessionTemplate;

    public String checkUserIdExist(String user_id) {
        return sqlSessionTemplate.selectOne("user.checkUserIdExist", user_id);
    }

    //(2) mapper의 “user.addUserInfo” 삽입명령을 호출하도록 addUserInfo()메소드 정의
    public void addUserInfo(UserBean joinUserBean) {
        sqlSessionTemplate.insert("user.addUserInfo", joinUserBean);
    }
}

```

3) UserService.java -(service에서 dao의 addUserInfo() 정의해두기) -(java방식과 xml방식 동일)

[kr.co.korea.service]

```

@Service
public class UserService {

    @Autowired
    private UserDao userDao;
    public boolean checkuserIdExist(String user_id) {
        String user_name=userDao.checkUserIdExist(user_id);
        if(user_name==null) {
            return true;
        } else {
            return false;
        }
    }
}

```

```

// (3) UserDao의 addUserInfo() 호출하도록 정의
public void addUserInfo(UserBean joinUserBean) {
    userDao.addUserInfo(joinUserBean); // UserDao의 addUserInfo() 호출
}
}

```

4) UserController.java -(service의 addUserInfor() 호출하기) -(java방식과 xml방식 동일)
[kr.co.korea.controller]

```

@Controller
@RequestMapping("/user")
public class UserController {

// (4) userService 객체로 주입
@Autowired
private UserService userService;

@GetMapping("/login")
public String login() {
    return "user/login";
}

// 가입창부터 가입완료까지 -----
@GetMapping("/join")
public String join(@ModelAttribute("joinUserBean") UserBean joinUserBean) {
    return "user/join"; // 가입창 페이지
}

@PostMapping("/join_pro") // 가입창 페이지 -> "회원가입" 버튼 클릭시 처리부분
public String join_pro(@Valid @ModelAttribute("joinUserBean") UserBean joinUserBean, BindingResult
result) {
    if(result.hasErrors()) {
        return "user/join";
    }

// (4) 객체를 통해 service의 addUserInfo()메소드 호출
    userService.addUserInfo(joinUserBean);

    return "user/join_success"; // 가입완료 메시지 페이지
}

// -----

@GetMapping("/modify")
public String modify() {
    return "user/modify";
}

@GetMapping("/logout")
public String logout() {
    return "user/logout";
}

@InitBinder
public void initBinder(WebDataBinder binder) {
    UserValidator validator1 = new UserValidator();
    binder.addValidators(validator1);
}
}

```

5) join_success.jsp -(가입완료 메시지) -(java방식과 xml방식 동일)

```

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<c:set var="root" value="${pageContext.request.contextPath }"/>
<script>

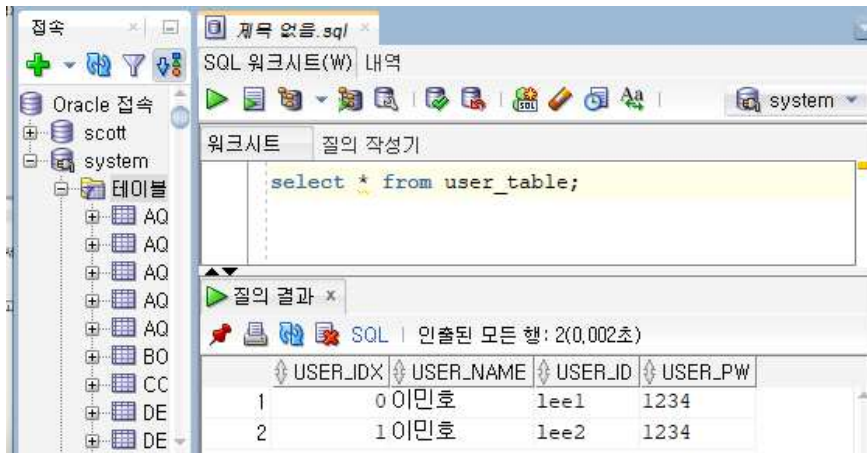
```

```

alert('가입이 완료되었습니다')
location.href = "${root}user/login"
</script>

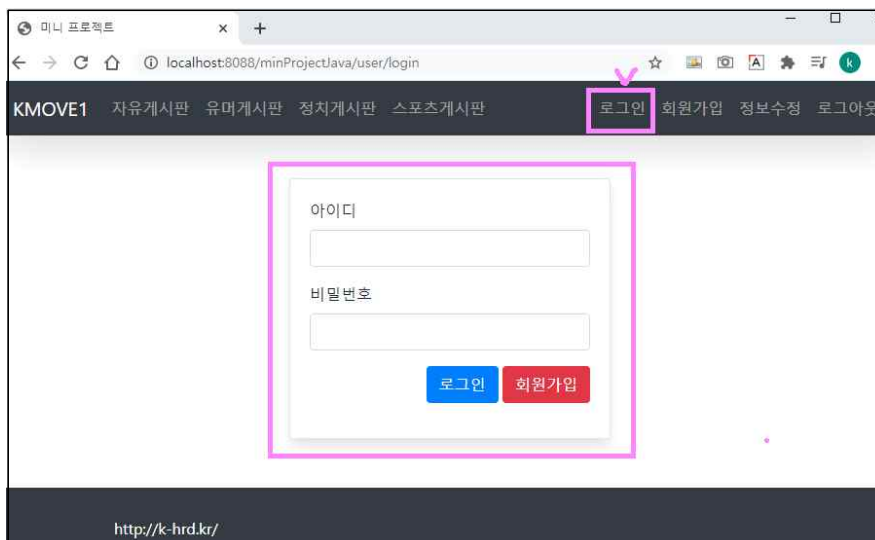
```

6) 저장된 결과 확인



2. 로그인

(1) 로그인 - 유효성 검사 (Java방식)



1) UserBean.java (XML방식 동일)

```

public class UserBean {
    private int user_idx;

    @Size(min=2, max=4)
    @Pattern(regexp = "[가-힣]*")
    private String user_name;

    @Size(min=4, max=20)
    @Pattern(regexp = "[a-zA-Z0-9]*")
    private String user_id;

    @Size(min=4, max=20)
    @Pattern(regexp = "[a-zA-Z0-9]*")
    private String user_pw;

    @Size(min=4, max=20)

```

<pre> @Pattern(regexp = "[a-zA-Z0-9]*") private String user_pw2; private boolean userIdExist; </pre>	
<pre> //로그인 되었는지 로그인 성공시 true저장 private boolean userLogin; </pre>	
<pre> public UserBean() { this.userIdExist = false; </pre>	
<pre> this.userLogin=false; } </pre>	
<pre> public int getUser_idx() { return user_idx; } public void setUser_idx(int user_idx) { this.user_idx = user_idx; } //이하생략 public boolean isUserLogin() { return userLogin; } public void setUserLogin(boolean userLogin) { this.userLogin = userLogin; } </pre>	

2)-1 RootApplicationContext.java -(UserBean 빈객체 만들어두기)

<pre> //빈객체 만들어두기 @Configuration public class RootApplicationContext { </pre>	<pre> //주입되는지 테스트하기 @Controller public class HomeController { </pre>
<pre> @Bean("loginUserBean") @SessionScope public UserBean loginUserBean() { return new UserBean(); } } </pre>	<pre> // @Resource(name="loginUserBean") // private UserBean loginUserBean; // @RequestMapping(value="/", //method=RequestMethod.GET) // public String home() { // System.out.println(loginUserBean); // return "redirect:main"; // } } </pre>

2)-2 root-context.xml -(UserBean 빈객체 만들어두기)

<pre> //빈객체 만들어두기 <?xml version="1.0" encoding="UTF-8"?> <beans xmlns="http://www.springframework.org/schema/beans" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd"> <bean class="kr.co.korea.beans.UserBean" id="loginUserBean" scope="session"/> </beans> </pre>	<pre> //주입되는지 테스트하기 @Controller public class HomeController { </pre>
	<pre> // @Resource(name="loginUserBean") // @Lazy // private UserBean loginUserBean; // @RequestMapping(value="/", //method=RequestMethod.GET) // public String home() { // System.out.println(loginUserBean); // return "redirect:main"; // } } </pre>

3) login.jsp -(form 태그라이브러리 활용) (XML방식 동일)

```

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>

<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form"%>

<c:set var='root' value='${pageContext.request.contextPath }/' />
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>미니 프로젝트</title>
<!-- Bootstrap CDN -->
<link rel="stylesheet"
    href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.0/css/bootstrap.min.css">
<script
    src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script
    src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.0/umd/popper.min.js"></script>
<script
    src="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.0/js/bootstrap.min.js"></script>
</head>
<body>

    <c:import url="/WEB-INF/views/include/top_menu.jsp" />

<div class="container" style="margin-top: 100px">
<div class="row">
    <div class="col-sm-3"></div>
    <div class="col-sm-6">
        <div class="card shadow">
            <div class="card-body">

                <div class="alert alert-danger">
                    <h3>로그인 실패</h3>
                    <p>아이디 비밀번호를 확인해주세요</p>
                </div>

                <form:form action="${root }user/login_pro" method='post'
                    modelAttribute="tempLoginUserBean">
                    <div class="form-group">
                        <form:label path="user_id">아이디</form:label>
                        <form:input path="user_id" class='form-control' />
                        <form:errors path='user_id' style='color:red' />
                    </div>
                    <div class="form-group">
                        <form:label path="user_pw">비밀번호</form:label>
                        <form:password path="user_pw" class='form-control' />
                        <form:errors path='user_pw' style='color:red' />
                    </div>
                    <div class="form-group text-right">
                        <form:button class='btn btn-primary'>로그인</form:button>
                        <a href="${root }user/join" class="btn btn-danger">회원가입</a>
                    </div>
                </form:form>
            </div>
        </div>
    </div>
</div>
<div class="col-sm-3"></div>
</div>
</div>

<c:import url="/WEB-INF/views/include/bottom_info.jsp" />
</body>
</html>

```

4) UserController.java -(XML방식 동일)

```

@Controller
@RequestMapping("/user")
public class UserController {

    //userService 객체로 주입
    @Autowired
    private UserService userService;
}

```

```

@GetMapping("/login")
public String login(@ModelAttribute("tempLoginUserBean") UserBean tempLoginUserBean) {
    return "user/login";
}

```

```

@PostMapping("/login_pro")
public String login_pro(@Valid @ModelAttribute("tempLoginUserBean") UserBean tempLoginUserBean,
BindingResult result) {
    if(result.hasErrors()) {
        return "user/login";
    }
    return "user/login_success";
}

```

```

@GetMapping("/join")
public String join(@ModelAttribute("joinUserBean") UserBean joinUserBean) {
    return "user/join";
}

@PostMapping("/join_pro")
public String join_pro(@Valid @ModelAttribute("joinUserBean") UserBean joinUserBean, BindingResult
result) {
    if(result.hasErrors()) {
        return "user/join";
    }

    //객체를 통해 service의 addUserInfo()메소드 호출
    userService.addUserInfo(joinUserBean);

    return "user/join_success";
}

@GetMapping("modify")
public String modify() {
    return "user/modify";
}

@GetMapping("logout")
public String logout() {
    return "user/logout";
}

@InitBinder
public void initBinder(WebDataBinder binder) {
    UserValidator validator1 = new UserValidator();
    binder.addValidators(validator1);
}
}

```

5) error_message.properties

Size.joinUserBean.user_name = 사용자 이름은 2~4글자여야 합니다.
 Size.joinUserBean.user_id = 사용자 아이디는 4~20글자여야 합니다.
 Size.joinUserBean.user_pw = 비밀번호는 4~20글자여야 합니다.
 Size.joinUserBean.user_pw2 = 비밀번호는 4~20글자여야 합니다.
 Pattern.joinUserBean.user_name = 사용자 이름은 한글만 허용합니다.
 Pattern.joinUserBean.user_id = 사용자 아이디는 영문자대소문자, 숫자만 허용합니다.
 Pattern.joinUserBean.user_pw = 비밀번호는 영문대소문자, 숫자만 허용합니다.
 Pattern.joinUserBean.user_pw2 = 비밀번호는 영문대소문자, 숫자만 허용합니다

NotEquals.joinUserBean.user_pw = 비밀번호가 일치하지 않습니다
 NotEquals.joinUserBean.user_pw2 = 비밀번호가 일치하지 않습니다

DontCheckUserIdExist.joinUserBean.user_id = 중복확인을 해주세요

Size.tempLoginUserBean.user_name = 사용자 이름은 2~4글자여야 합니다.
 Size.tempLoginUserBean.user_id = 사용자 아이디는 4~20글자여야 합니다.
 Pattern.tempLoginUserBean.user_id = 사용자 아이디는 영문자대소문자, 숫자만 허용합니다.
 Pattern.tempLoginUserBean.user_pw = 비밀번호는 영문대소문자, 숫자만 허용합니다.

6) Validator.java (UserBean 클래스를 joinUserBean, tempLoginUserBean 2개의 이름으로 사용하므로 validate메소드의 유효성 검사를 모두 하게 된다. joinUserBean일때만 validate메소드 호출하도록 조건지정)

```
package kr.co.korea.validator;

import org.springframework.validation.Errors;
import org.springframework.validation.Validator;

import kr.co.korea.beans.UserBean;

public class Validator implements Validator{

    @Override
    public boolean supports(Class<?> clazz) {
        // TODO Auto-generated method stub
        return UserBean.class.isAssignableFrom(clazz);
    }

    @Override
    public void validate(Object target, Errors errors) {
        // TODO Auto-generated method stub
        UserBean userBean = (UserBean)target;

        String beanName = errors.getObjectName();

        if(beanName.equals("joinUserBean")) {
            if(userBean.getUser_pw().equals(userBean.getUser_pw2()) == false) {
                errors.rejectValue("user_pw", "NotEquals");
                errors.rejectValue("user_pw2", "NotEquals");
            }

            if(userBean.isUserIdExist() == false) {
                errors.rejectValue("user_id", "DontCheckUserIdExist");
            }
        }
    }
}
```

7) login_success.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<c:set var="root" value="${pageContext.request.contextPath }"/>
<script>
    alert('로그인 되었습니다')
    location.href = "${root}main'
</script>
```

(2) 로그인 정보와 일치하는 sql쿼리 조회

select user_idx, user_name from user_table where user_id='choil' and user_pw='123456';

1)-1 UserMapper.java -(mapper에서 쿼리문 만들기) -(Java방식)

[kr.co.korea.mapper]패키지

```
package kr.co.korea.mapper;

import org.apache.ibatis.annotations.Insert;
import org.apache.ibatis.annotations.Select;
```

```

import kr.co.korea.beans.UserBean;

public interface UserMapper {

    @Select("select user_name " +
            "from user_table " +
            "where user_id= #{user_id}")
    String checkUserIdExist(String user_id);

    @Insert("insert into user_table (user_idx, user_name, user_id, user_pw) " +
            "values (user_seq.nextval, #{user_name}, #{user_id}, #{user_pw})")
    void addUserInfo(UserBean joinUserBean);

    // 로그인 (아이디와 패스워드 일치할 경우 idx, name 조회)
    @Select("select user_idx, user_name " +
            "from user_table " +
            "where user_id=#{user_id} and user_pw=#{user_pw}")
    UserBean getLoginUserInfo(UserBean tempLoginUserBean);

}

```

1)-2 user_mapper.xml -(mapper에서 쿼리문 만들기) -(xml방식)

[WebContent-WEB-INF-mapper]폴더

```

<?xml version='1.0' encoding='UTF-8?'>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="user">
    <select id="checkUserIdExist" parameterType="java.lang.String" resultType="java.lang.String">
        <![CDATA[
            select user_name
            from user_table
            where user_id = #{user_id}
        ]]>
    </select>

    <insert id="addUserInfo" parameterType="kr.co.korea.beans.UserBean">
        <![CDATA[
            insert into user_table (user_idx, user_name, user_id, user_pw)
            values (user_seq.nextval, #{user_name}, #{user_id}, #{user_pw})
        ]]>
    </insert>

    <!--(1) mapper 아이디,비번이 일치하는 idx,name 조회-->
    <select id="getLoginUserInfo" parameterType="kr.co.softcampus.beans.UserBean"
resultType="kr.co.softcampus.beans.UserBean">
        <![CDATA[
            select user_idx, user_name
            from user_table
            where user_id=#{user_id} and user_pw=#{user_pw}
        ]]>
    </select>

</mapper>

```

2)-1 UserDao.java -(dao에서 mapper의 getLoginUserInfo() 정의해두기) -(Java방식)

[kr.co.korea.dao]

```

@Repository
public class UserDao {

    @Autowired
    private UserMapper userMapper;
    public String checkUserIdExist(String user_id) {
        return userMapper.checkUserIdExist(user_id);
    }

    public void addUserInfo(UserBean joinUserBean) {
        userMapper.addUserInfo(joinUserBean); /
    }

    //mapper의 getLoginUserInfo 호출하도록 정의

```

```

    public UserBean getLoginUserInfo(UserBean tempLoginUserBean) {
        return userMapper.getLoginUserInfo(tempLoginUserBean); //mapper의 getLoginUserInfo호출
    }
}

```

2)-2 UserDao.java -(mapper의 'user.getLoginUserInfo' 조회명령 호출하도록 getLoginUserInfo() 정의해두기) -(xml방식)
[kr.co.korea.dao]

```

@Repository
public class UserDao {

    @Autowired
    private SqlSessionTemplate sqlSessionTemplate;

    public String checkUserIdExist(String user_id) {
        return sqlSessionTemplate.selectOne("user.checkUserIdExist", user_id);
    }

    public void addUserInfo(UserBean joinUserBean) {
        sqlSessionTemplate.insert("user.addUserInfo", joinUserBean);
    }

    public UserBean getLoginUserInfo(UserBean tempLoginUserBean) {
        return sqlSessionTemplate.selectOne("user.getLoginUserInfo", tempLoginUserBean);
    }
}

```

3)-1 UserService.java -(service에서 dao의 getLoginUserInfo() 정의해두기) -(Java방식)
[kr.co.korea.service]

```

@Service
public class UserService {

    @Autowired
    private UserDao userDao;

    @Resource(name = "loginUserBean")
    private UserBean loginUserBean;

    public boolean checkuserIdExist(String user_id) {
        String user_name=userDao.checkUserIdExist(user_id);
        if(user_name==null) {
            return true;
        } else {
            return false;
        }
    }

    public void addUserInfo(UserBean joinUserBean) {
        userDao.addUserInfo(joinUserBean);
    }

    //UserDao의 getLoginUserInfo() 호출하도록 정의
    public void getLoginUserInfo(UserBean tempLoginUserBean) {
        UserBean tempLoginUserBean2 = userDao.getLoginUserInfo(tempLoginUserBean); //호출
        if(tempLoginUserBean2 != null) { //db에서 select된게 있으면
            loginUserBean.setUser_idx(tempLoginUserBean2.getUser_idx());
            loginUserBean.setUser_name(tempLoginUserBean2.getUser_name());
            loginUserBean.setUserLogin(true);
        }
    }
}

```

```
}
```

3)-2 UserService.java -(service에서 dao의 getLoginUserInfo() 정의해두기) -(xml방식)
[kr.co.korea.service]

```
@Service
public class UserService {

    @Autowired
    private UserDao userDao;

    @Resource(name = "loginUserBean")
    @Lazy
    private UserBean loginUserBean;

    public boolean checkuserIdExist(String user_id) {
        String user_name=userDao.checkUserIdExist(user_id);
        if(user_name==null) {
            return true;
        } else {
            return false;
        }
    }

    public void addUserInfo(UserBean joinUserBean) {
        userDao.addUserInfo(joinUserBean);    //UserDao의 addUserInfo()호출
    }

    //UserDao의 getLoginUserInfo() 호출하도록 정의
    public void getLoginUserInfo(UserBean tempLoginUserBean) {

        UserBean tempLoginUserBean2 = userDao.getLoginUserInfo(tempLoginUserBean); //호출

        if(tempLoginUserBean2 != null) { //db에서 select된게 있으면
            loginUserBean.setUser_idx(tempLoginUserBean2.getUser_idx());
            loginUserBean.setUser_name(tempLoginUserBean2.getUser_name());
            loginUserBean.setUserLogin(true);
        }
    }
}
```

4) UserController.java -(service의 getLoginUserInfo() 호출하기) -java방식과 xml방식 동일
[kr.co.korea.controller]

```
@Controller
@RequestMapping("/user")
public class UserController {

    @Autowired
    private UserService userService;

    @Resource(name = "loginUserBean")
    private UserBean loginUserBean;

    @GetMapping("/login")
    public String login(@ModelAttribute("tempLoginUserBean") UserBean tempLoginUserBean) {
        return "user/login";
    }

    @PostMapping("/login_pro")
    public String login_pro(@Valid @ModelAttribute("tempLoginUserBean") UserBean tempLoginUserBean,
        BindingResult result) {

        if(result.hasErrors()) {
            return "user/login";
        }
    }
}
```

```

        userService.getLoginUserInfo(tempLoginUserBean); //db를 통해 조회된 idx.name select하기
        if(loginUserBean.isUserLogin() == true) {
            return "user/login_success";
        } else {
            return "user/login_fail";
        }
    }

    @GetMapping("/join")
    public String join(@ModelAttribute("joinUserBean") UserBean joinUserBean) {
        return "user/join";
    }

    @PostMapping("/join_pro")
    public String join_pro(@Valid @ModelAttribute("joinUserBean") UserBean joinUserBean, BindingResult
result) {
        if(result.hasErrors()) {
            return "user/join";
        }

        //객체를 통해 service의 addUserInfo()메소드 호출
        userService.addUserInfo(joinUserBean);

        return "user/join_success";
    }

    @GetMapping("/modify")
    public String modify() {
        return "user/modify";
    }

    @GetMapping("/logout")
    public String logout() {
        return "user/logout";
    }

    @InitBinder
    public void initBinder(WebDataBinder binder) {
        UserValidator validator1 = new UserValidator();
        binder.addValidators(validator1);
    }
}

```

5) login_fail.jsp -(가입완료 메시지) -java방식과 xml방식 동일
실패시 로그인창으로 다시 이동

```

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<c:set var="root" value="${pageContext.request.contextPath}"/>
<script>
    alert('로그인에 실패하였습니다')
    location.href = "${root}user/login'
</script>

```

(3) 로그인 실패와 성공 표현

1) UserController.java

```

@RequestMapping("/user")
public class UserController {

    @Autowired
    private UserService userService;
}

```

```

@Resource(name = "loginUserBean")
private UserBean loginUserBean;

@GetMapping("/login")
public String login(@ModelAttribute("tempLoginUserBean") UserBean tempLoginUserBean,
    @RequestParam(value = "logcheck", defaultValue = "true") boolean logcheck,
    Model model) {

    model.addAttribute("logcheck", logcheck); //logcheck

    return "user/login";
}

```

//이하 생략

2) login.jsp

```

<body>

    <c:import url="/WEB-INF/views/include/top_menu.jsp" />

    <div class="container" style="margin-top: 100px">
    <div class="row">
        <div class="col-sm-3"></div>
        <div class="col-sm-6">
            <div class="card shadow">
                <div class="card-body">
                    <c:if test="${logcheck == false}">
                        <div class="alert alert-danger">
                            <h3>로그인 실패</h3>
                            <p>아이디 비밀번호를 확인해주세요</p>
                        </div>
                    </c:if>
                    <form:form action="${root }user/login_pro" method='post'
                        modelAttribute="tempLoginUserBean">
                        <div class="form-group">
                            <form:label path="user_id">아이디</form:label>
                            <form:input path="user_id" class='form-control' />
                            <form:errors path='user_id' style='color:red' />
                        </div>
                        <div class="form-group">
                            <form:label path="user_pw">비밀번호</form:label>
                            <form:password path="user_pw" class='form-control' />
                            <form:errors path='user_pw' style='color:red' />
                        </div>
                        <div class="form-group text-right">
                            <form:button class="btn btn-primary">로그인</form:button>
                            <a href="${root }user/join" class="btn btn-danger">회원가입</a>
                        </div>
                    </form:form>
                </div>
            </div>
        </div>
    </div>

```

3) login_fail.jsp

```

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<c:set var="root" value="${pageContext.request.contextPath}"/>
<script>
    alert('로그인에 실패하였습니다')
    location.href = "${root}user/login?logcheck=false'
</script>

```

3. 로그아웃

(1) 로그인된 경우 로그아웃 글자만 표현

1) ServletAppContext.java

```

@Resource(name = "loginUserBean")
private UserBean loginUserBean;

```

```

@Override
public void addInterceptors(InterceptorRegistry registry) {
    // TODO Auto-generated method stub
    WebMvcConfigurer.super.addInterceptors(registry);
    TopMenuInterceptor topMenuInterceptor=new TopMenuInterceptor(TopMenuService, loginUserBean);

    InterceptorRegistration reg1=registry.addInterceptor(topMenuInterceptor);
    reg1.addPathPatterns("/**");
}

```

2)-1 TopMenuInterceptor.java -(java방식)

```

public class TopMenuInterceptor implements HandlerInterceptor{

    private TopMenuService topMenuService;
    private UserBean loginUserBean;

    public TopMenuInterceptor(TopMenuService topMenuService, UserBean loginUserBean) {
        this.topMenuService=topMenuService;
        this.loginUserBean=loginUserBean;
        // TODO Auto-generated constructor stub
    }

    @Override
    public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object
handler)
        throws Exception {
        // TODO Auto-generated method stub
        List<BoardInfoBean> topMenuList=topMenuService.getTopMenuList();
        request.setAttribute("topMenuList", topMenuList);
        request.setAttribute("loginUserBean", loginUserBean);

        return true;
    }
}

```

2)-2 TopMenuInterceptor.java -(xml방식)

```

public class TopMenuInterceptor implements HandlerInterceptor{

    @Autowired
    private TopMenuService topMenuService;

    @Resource(name="loginUserBean")
    @Lazy
    private UserBean loginUserBean;

    @Override
    public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object
handler)
        throws Exception {
        // TODO Auto-generated method stub
        List<BoardInfoBean> topMenuList = topMenuService.getTopMenuList();
        request.setAttribute("topMenuList", topMenuList);
        request.setAttribute("loginUserBean", loginUserBean);
        return true;
    }
}

```

3) top_menu.jsp -java방식과 xml방식 동일

```

<nav class="navbar navbar-expand-md bg-dark navbar-dark fixed-top shadow-lg">
    <a class="navbar-brand" href="{root }main">KMOVE1</a>
    <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navMenu">
        <span class="navbar-toggler-icon"></span>
    </button>
    <ul class="navbar-nav ml-auto">

```

```

<c:choose>
  <c:when test='${loginUserBean.userLogin == true }> //userLogin값이 true이면 로그인상태
    <li class="nav-item">
      <a href="${root }user/modify" class="nav-link">정보수정</a>
    </li>
    <li class="nav-item">
      <a href="${root }user/logout" class="nav-link">로그아웃</a>
    </li>
  </c:when>
  <c:otherwise> //userLogin값이 false 이면 로그아웃상태
    <li class="nav-item">
      <a href="${root }user/login" class="nav-link">로그인</a>
    </li>
    <li class="nav-item">
      <a href="${root }user/join" class="nav-link">회원가입</a>
    </li>
  </c:otherwise>
</c:choose>
</ul>
</div>
</nav>

```

4) UserController.java -java방식과 xml방식 동일

```

@GetMapping("logout")
public String logout() {

    loginUserBean.setUserLogin(false); //userLogin변수에 false값을 세트

    return "user/logout";
}

```

(2) 로그인된 경우에만 정보 확인 가능 (url에 주소 입력할 경우 로그인창으로 이동)

1)-1 CheckLoginInterceptor.java -(java방식)

```

public class CheckLoginInterceptor implements HandlerInterceptor{

    private UserBean loginUserBean;

    public CheckLoginInterceptor(UserBean loginUserBean) {
        // TODO Auto-generated constructor stub
        this.loginUserBean = loginUserBean;
    }

    @Override
    public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler)
        throws Exception {
        // TODO Auto-generated method stub
        if(loginUserBean.isUserLogin() == false) {
            String contextPath = request.getContextPath();
            response.sendRedirect(contextPath + "/user/not_login");
            return false;
        }
        return true;
    }
}

```

1)-2 CheckLoginInterceptor.java -(xml방식)

```

public class CheckLoginInterceptor implements HandlerInterceptor{

    // private UserBean loginUserBean;

    // public CheckLoginInterceptor(UserBean loginUserBean) {
    //     // TODO Auto-generated constructor stub
    //     this.loginUserBean = loginUserBean;
    // }

    @Resource(name="loginUserBean")

```



```

@Lazy
private UserBean loginUserBean;

@Override
public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object
handler)
    throws Exception {
    // TODO Auto-generated method stub
    if(loginUserBean.isUserLogin() == false) {
        String contextPath = request.getContextPath();
        response.sendRedirect(contextPath + "/user/not_login");
        return false;
    }
    return true;
}

```

2)-1 ServletAppContext.java -(java방식)

```

@Override
public void addInterceptors(InterceptorRegistry registry) {
    // TODO Auto-generated method stub
    WebMvcConfigurer.super.addInterceptors(registry);
    TopMenuInterceptor topMenuInterceptor = new TopMenuInterceptor(TopMenuService,
loginUserBean);

    InterceptorRegistration reg1 = registry.addInterceptor(topMenuInterceptor);
    reg1.addPathPatterns("/**");

    CheckLoginInterceptor checkLoginInterceptor = new CheckLoginInterceptor(loginUserBean);
    InterceptorRegistration reg2 = registry.addInterceptor(checkLoginInterceptor);
    reg2.addPathPatterns("/user/modify", "/user/logout", "/board/*");
    reg2.excludePathPatterns("/board/main");
}

```

2)-2 servlet-context.xml -(xml방식)

```

<interceptors>
    <interceptor>
        <mapping path="/**"/>
        <beans:bean class='kr.co.korea.interceptor.TopMenuInterceptor'/>
    </interceptor>

    <interceptor>
        <mapping path="/user/modify"/>
        <mapping path="/user/logout"/>
        <mapping path="/board/*"/>
        <exclude-mapping path="/board/main"/>
        <beans:bean class='kr.co.korea.interceptor.CheckLoginInterceptor'/>
    </interceptor>
</interceptors>

```

3) UserController.java

```

@GetMapping("/not_login")
public String not_login() {
    return "user/not_login";
}

```

4) not_login.jsp

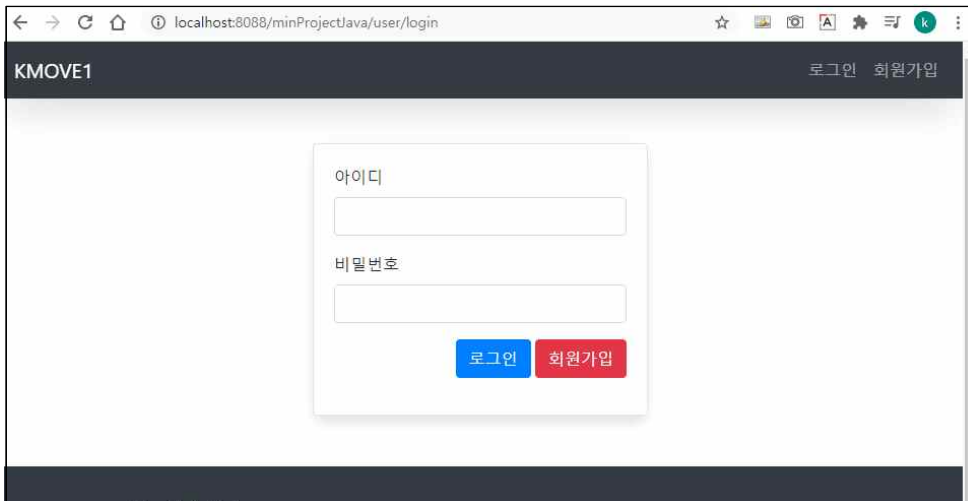
```

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<c:set var="root" value="${pageContext.request.contextPath }"/>
<script>
    alert('로그인 해주세요')

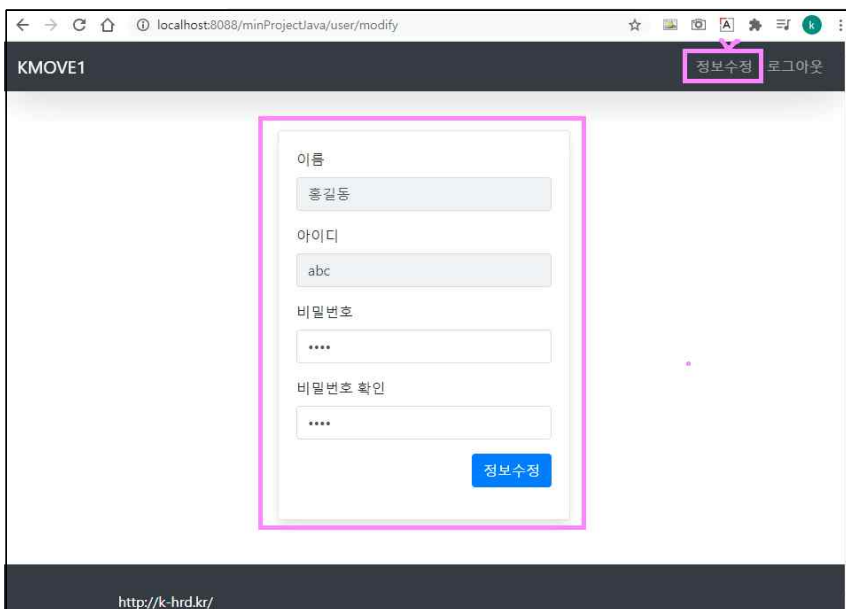
```

```
location.href='${root}user/login'  
</script>
```

5) 실습시 :<http://localhost:8088/minProjectJava/user/modify> 입력시
user/login으로 이동하도록



4. 정보수정



(1) 로그인 정보에 대한 - 정보수정

1) modify.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"  
    pageEncoding="UTF-8"%>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
```

```

<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>

<c:set var='root' value='${pageContext.request.contextPath }/'/>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>미니 프로젝트</title>
<!-- Bootstrap CDN -->
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.0/css/bootstrap.min.css">
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.0/umd/popper.min.js"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.0/js/bootstrap.min.js"></script>

</head>
<body>

<c:import url="/WEB-INF/views/include/top_menu.jsp"/>

<div class="container" style="margin-top:100px">
<div class="row">
    <div class="col-sm-3"></div>
    <div class="col-sm-6">
        <div class="card shadow">
            <div class="card-body">

                <form:form action= "${root }user/modify_pro' method= 'post'
modelAttribute= "modifyUserBean">
                    <div class="form-group">
                        <form:label path= "user_name">이름</form:label>
                        <form:input path= "user_name" class= 'form-control' readonly= "true"/>
                    </div>
                    <div class="form-group">
                        <form:label path= "user_id">아이디</form:label>
                        <form:input path= "user_id" class= 'form-control' readonly= "true"/>
                    </div>
                    <div class="form-group">
                        <form:label path= "user_pw">비밀번호</form:label>
                        <form:password path= "user_pw" class= 'form-control' />
                        <form:errors path= 'user_pw' style= 'color:red' />
                    </div>
                    <div class="form-group">
                        <form:label path= "user_pw2">비밀번호 확인</form:label>
                        <form:password path= "user_pw2" class= 'form-control' />
                        <form:errors path= 'user_pw2' style= 'color:red' />
                    </div>
                    <div class="form-group">
                        <div class="text-right">
                            <form:button class= 'btn btn-primary'>정보수정</form:button>
                        </div>
                    </div>
                </form:form>

            </div>
        </div>
    </div>
</div>
<div class="col-sm-3"></div>
</div>

<c:import url="/WEB-INF/views/include/bottom_info.jsp"/>

</body>
</html>

```

2) UserController.java

```

@GetMapping("/modify")
public String modify(@ModelAttribute("modifyUserBean") UserBean modifyUserBean) {

```

```
        return "user/modify";
    }
```

(3) db에서 로그인한 정보 가져오기

1)-1 UserMapper.java -(mapper에서 쿼리문 만들기) -(Java방식)

[kr.co.korea.mapper]

```
// 로그인한 사람의 정보 가져와서 "정보수정" 페이지에서 활용
@Select("select user_id, user_name " +
        "from user_table " +
        "where user_idx = #{user_idx}")
UserBean getModifyUserInfo(int user_idx);
```

1)-2 user_mapper.xml -(mapper에서 쿼리문 만들기) -(xml방식)

```
<!--(1) mapper 아이디와 일치하는 id.name 조회하여 '정보수정' 페이지에서 활용 -->
<select id="getModifyUserInfo" parameterType="java.lang.Integer"
resultType="kr.co.softcampus.beans.UserBean">
    <![CDATA[
        select user_id, user_name
        from user_table
        where user_idx = #{user_idx}
    ]]>
</select>
```

2)-1 UserDao.java -(dao에서 mapper의 getModifyUserInfo() 정의해두기) -(Java방식)

[kr.co.korea.dao]

```
//mapper의 getModifyUserInfo 호출하도록 정의
public UserBean getModifyUserInfo(int user_idx) {
    return userMapper.getModifyUserInfo(user_idx);
}
```

2)-2 UserDao.java -(dao에서 mapper.xml파일내의 "user.getModifyUserInfo"

조회하도록 getModifyUserInfo()메소드 정의해두기) -(xml방식)

```
public UserBean getModifyUserInfo(int user_idx) {
    return sqlSessionTemplate.selectOne("user.getModifyUserInfo", user_idx);
}
```

3) UserService.java -(service에서 dao의 getModifyUserInfo() 정의해두기) -(java방식과 xml방식 동일)

[kr.co.korea.service]

```
//UserDao의 getModifyUserInfo() 호출하도록 정의
public void getModifyUserInfo(UserBean modifyUserBean) {
    UserBean tempModifyUserBean = userDao.getModifyUserInfo(loginUserBean.getUser_idx());

    modifyUserBean.setUser_id(tempModifyUserBean.getUser_id());
    modifyUserBean.setUser_name(tempModifyUserBean.getUser_name());
    modifyUserBean.setUser_idx(loginUserBean.getUser_idx());
}
```

4) UserController.java

```

@GetMapping("/modify")
public String modify(@ModelAttribute("modifyUserBean") UserBean modifyUserBean) {

    userService.modifyUserInfo(modifyUserBean);

    return "user/modify";
}

@PostMapping("/modify_pro")
public String modify_pro(@Valid @ModelAttribute("modifyUserBean") UserBean modifyUserBean,
BindingResult result) {

    if (result.hasErrors()) {
        return "user/modify";
    }
    return "user/modify_success";
}

```

(4) 유효성 검사

1) error_message.properties

```

Size.modifyUserBean.user_pw = 비밀번호는 4 ~ 20글자여야 합니다
Size.modifyUserBean.user_pw2 = 비밀번호는 4 ~ 20글자여야 합니다
Pattern.modifyUserBean.user_pw = 비밀번호는 영문대소문자, 숫자만 허용합니다
Pattern.modifyUserBean.user_pw2 = 비밀번호는 영문대소문자, 숫자만 허용합니다
NotEquals.modifyUserBean.user_pw = 비밀번호가 일치하지 않습니다
NotEquals.modifyUserBean.user_pw2 = 비밀번호가 일치하지 않습니다

```

2) UserValidator.java

```

@Override
public void validate(Object target, Errors errors) {
    // TODO Auto-generated method stub
    UserBean userBean = (UserBean)target;

    String beanName = errors.getObjectNames();

    if(beanName.equals("joinUserBean") || beanName.equals("modifyUserBean")) {
        if(userBean.getUser_pw().equals(userBean.getUser_pw2()) == false) {
            errors.rejectValue("user_pw", "NotEquals");
            errors.rejectValue("user_pw2", "NotEquals");
        }
    }

    if(beanName.equals("joinUserBean")) {
        if(userBean.isUserIdExist() == false) {
            errors.rejectValue("user_id", "DontCheckUserIdExist");
        }
    }
}

```

(5) db에 정보수정

1)-1 UserMapper.java

```

// id 일치하는 것 찾아 비번 변경
@Update("update user_table " +
        "set user_pw = #{user_pw} " +
        "where user_idx = #{user_idx}")
void modifyUserInfo(UserBean modifyUserBean);

```

1)-2 user_mapper.xml

```

<update id="modifyUserInfo" parameterType="kr.co.softcampus.beans.UserBean">
    <![CDATA[
        update user_table
        set user_pw = #{user_pw}
    ]]>

```

```

        where user_idx = #{user_idx}
    ]]>
</update>

```

2)-1 UserDao.java

```

public void modifyUserInfo(UserBean modifyUserBean) {
    userMapper.modifyUserInfo(modifyUserBean);
}

```

2)-2 UserDao.java

```

public void modifyUserInfo(UserBean modifyUserBean) {
    sqlSessionTemplate.update("user.modifyUserInfo", modifyUserBean);
}

```

3) UserService.java

```

public void modifyUserInfo(UserBean modifyUserBean) {
    modifyUserBean.setUser_idx(loginUserBean.getUser_idx());
    userDao.modifyUserInfo(modifyUserBean);
}

```

4) UserController.java

```

@PostMapping("/modify_pro")
public String modify_pro(@Valid @ModelAttribute("modifyUserBean") UserBean modifyUserBean,
BindingResult result) {
    if (result.hasErrors()) {
        return "user/modify";
    }

    userService.modifyUserInfo(modifyUserBean);

    return "user/modify_success";
}

```

5) modify_success.jsp

```

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<c:set var='root' value= "${pageContext.request.contextPath}"/>
<script>
    alert('수정되었습니다')
    location.href = '${root}user/modify'
</script>

```