1. 자바 기본 환경에서 스프링 구성

1. eclipse-jee-2021-03 download & install & start (java15 포함되어 있음)

```
https://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/2021-03/R/eclipse-jee-2021-03-R-win32-x86_64.zip&mirror_id=1275
```

2. 이클립스에서 1_JavaBasicSpring 프로젝트를 만들어서 다음과 같이 작성 후 실행

```
package my.test1;
import my.bean.Hello;
public class HelloMain {
	public static void
main(String[] args) {
		Hello h=new Hello();

System.out.println(h.getMsg());
	}
}
```

```
package my.bean;
public class Hello {
	private String msg="hello";

	public String getMsg() {
		return msg;
	}

	public void setMsg(String
msg) {
		this.msg = msg;
	}
}
```

3. spring_basic_lib 다운 & 압축해제

```
https://drive.google.com/drive/u/1/folders/1MgBUpvWI5GTYWRKIoaGynHdjxTsLyMfK
```

4. 1_JavaBasicSpring 프로젝트에서 Build Path> Add External Archives...를 하여 이 jar들 연결

5. 이 프로젝트에 다음과 같이 xml 작성

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
	"http://www.springframework.org/dtd/spring-beans-2.0.dtd">

<beans>
  <bean id="h" class="my.bean.Hello"></bean>
</beans>
```

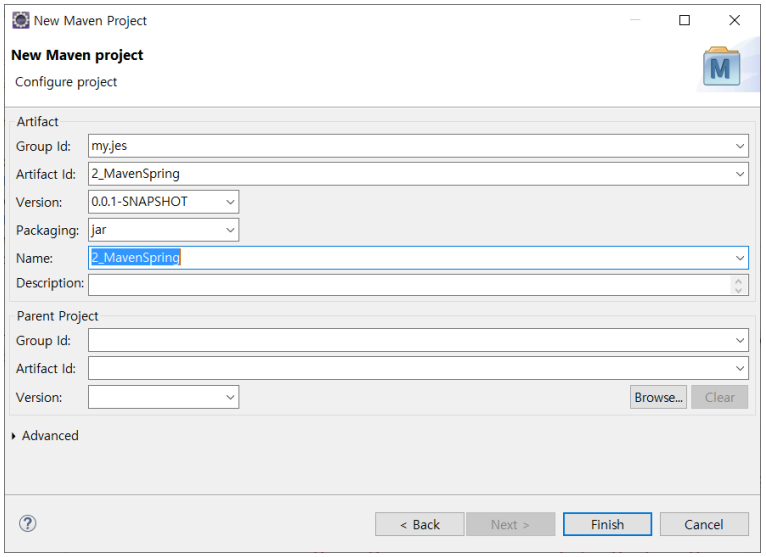
6. HelloMain.java를 다음과 같이 수정 후 실행

```
package my.test1;
import org.springframework.beans.factory.BeanFactory;
import
org.springframework.context.support.FileSystemXmlApplicationContext;

import my.bean.Hello;
public class HelloMain {
        public static void main(String[] args) {
                BeanFactory factory=new
FileSystemXmlApplicationContext("beans.xml");
                Hello h=(Hello) factory.getBean("h");
                System.out.println(h.getMsg());
        }
}
```

7. Circle, Rectangle, Triangle을 Shape 상속 후 getBean으로 설명
   ⇒ chap02

2. maven project로 스프링 구성

1. eclipse>File>New>Other>Maven>Maven Project ⇒ Create a simple project를 체크 ⇒ 다음과 같이 입력후 Finish



2. 1번 Hello 예제를 복사해 오면 import 부분에서 에러가 난다. pom.xml에 다음을 추가한 뒤 에러가 사라지는 것을 확인한다.

```
<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>my.jes</groupId>
  <artifactId>2_MavenSpring</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>2_MavenSpring</name>

  <dependencies>
          <!--
https://mvnrepository.com/artifact/org.springframework/spring-beans -->
                  <dependency>
                     <groupId>org.springframework</groupId>
                     <artifactId>spring-beans</artifactId>
                     <version>5.3.9</version>
                  </dependency>

                  <!--
https://mvnrepository.com/artifact/org.springframework/spring-context -->
                  <dependency>
                     <groupId>org.springframework</groupId>
                     <artifactId>spring-context</artifactId>
                     <version>5.3.9</version>
                  </dependency>


  </dependencies>


</project>
```

3. beans.xml을 이 프로젝트 바로 아래로 복사해온 뒤 HelloMain 실행

3. 이클립스에 스프링 플러그인 구성

1. 이클립스 마켓플레이스에는 sts4(스프링부트용 플러그인)만 지워되므로 이클립스>install new software에서 다음 플러그인을 버전에 맞게 추가, 설치

```
http://dist.springsource.com/snapshot/STS/nightly-distributions.html
```

2. File>New>Others>Spring>Spring legacy project>Simple Spring Utility Project를 선택하고 3_SimpleSpringPlugin이라고 프로젝트이름을 짓는다

(테스트 코드들은 전부 삭제한다)

3. HelloMain.java, Hello.java, beans.xml을 복사해 넣고 HelloMain을 실행 ⇒ 잘됨

(**추가 실습)

4. 프로젝트에서 자동으로 생성된 코드인 ExampleService.java와 Service.java를 beans.xml에 등록해 보자

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
	xmlns:p="http://www.springframework.org/schema/p"
	xmlns:aop="http://www.springframework.org/schema/aop"
	xmlns:context="http://www.springframework.org/schema/context"
	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-3.0.xsd
http://www.springframework.org/schema/aop
http://www.springframework.org/schema/aop/spring-aop-3.0.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-3.0.xsd">


  <context:component-scan
base-package="com.jes.test1"></context:component-scan>
  <bean id="h" class="com.jes.test1.Hello"></bean>

</beans>
```

5. HelloMain.java를 다음과 같이 수정 후 실행

```
package com.jes.test1;

import org.springframework.beans.factory.BeanFactory;
import
org.springframework.context.support.FileSystemXmlApplicationContext;

public class HelloMain {

	public static void main(String[] args) throws Exception {
		BeanFactory factory=new
FileSystemXmlApplicationContext("beans.xml");
		Service h=(Service) factory.getBean("exampleService");
//이름은 반드시 소문자로 시작해야 함
		System.out.println(h.getMessage());
	}
```

```
}
```

4. 기본 서블릿 구성

1. tomcat9.0.52 download
2. File>New>Dynamic Web Project ⇒ tomcat 9.0.52 설정 ⇒ web module version을 반드시 2.5로 선택 (web.xml이 생기도록)
3. web.controller.MainServlet을 /main 이라는 alias로 설정
4. Run As>Run on Server 로 실행
5. mybatis lib를 다운로드 한뒤 WEB-INF 밑에 저장한다
6.
7. webapp/index.jsp를 작성한다

```
<%@ page language="java" contentType="text/html; charset=EUC-KR"
    pageEncoding="EUC-KR"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="EUC-KR">
<title>Insert title here</title>
</head>
<body>
index.jsp
</body>
</html>
```

8. MainServlet에서 ServletDispatcherRequest를 통해 forward한다

5. Spring입힌 서블릿 구성 (수동 스프링MVC)

1. https://drive.google.com/drive/u/1/folders/1MgBUpvWI5GTYWRKIoaGynHdjxTsLyMfK 에서 spring_all_lib.zip을 다운 받아 위 프로젝트의 WEB-INF/lib 폴더를 만들고 그 밑에 풀어 놓는다
2. web.xml을 다음과 같이 수정한다

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" version="2.5">
<servlet>
   <servlet-name>app</servlet-name>

<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
 </servlet>
 <servlet-mapping>
   <servlet-name>app</servlet-name>
  <url-pattern>*.do</url-pattern>
 </servlet-mapping>
</web-app>
```

3. web.xml과 같은 위치에 app-servlet.xml을 다음과 같이 작성한다

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:context="http://www.springframework.org/schema/context"
      xmlns:mvc="http://www.springframework.org/schema/mvc"
      xmlns:aop="http://www.springframework.org/schema/aop"
      xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-4.0.xsd
                  http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-4.0.xsd
                  http://www.springframework.org/schema/mvc
http://www.springframework.org/schema/mvc/spring-mvc-4.0.xsd
                  http://www.springframework.org/schema/aop
http://www.springframework.org/schema/aop/spring-aop-4.0.xsd ">


    <bean id="main" class="web.controller.MainServlet"></bean>

    <bean id="urlMapping"
class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
          <property name="mappings">
                <props>
                      <prop key="/index.do">main</prop>
                </props>
          </property>
    </bean>
</beans>
```

4. MainServlet.java를 다음과 같이 수정한다

```
package web.controller;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.web.servlet.ModelAndView;
import org.springframework.web.servlet.mvc.Controller;

public class MainServlet implements Controller {


	@Override
	public ModelAndView handleRequest(HttpServletRequest request,
HttpServletResponse response) throws Exception {

		return new ModelAndView("index.jsp");
	}

}
```

5. http://localhost:8080/4_Servlet/index.do 로 요청 시 index.jsp가 잘 보이는지 확인

6. SpringMVC 구성 (자동 스프링 MVC)

1. File>New>Other...>Spring>Spring Legacy Project>Spring MVC Project 하여 Run As > Run on Server 하면 http://localhost:8080/web/ 로 home.jsp의 결과가 나옴

7. 서블릿으로 게시판 작성

1. File>New>Dynamic Web Project ⇒ 6_Servlet_Board 를 서블릿 모듈 2.5로 작성
2. 부트스트랩 free code download

```
https://startbootstrap.com/theme/landing-page
```

3. 내 프로젝트의 webapp 밑에 풀어놓음
4. index.html의 20행 부근을 다음과 같이 수정

```
<!-- Navigation-->
        <nav class="navbar navbar-light bg-light static-top">
            <div class="container">
                <a class="navbar-brand" href="#!" >Start Bootstrap</a>
                        <a class="navbar-brand" href="#"
onclick='window.open("main?sign=boardList", "_blank",
"toolbar=yes,scrollbars=yes,resizable=yes,top=100,left=500,width=800,height=600
");'>게시판</a>
                <a class="btn btn-primary" href="#signup">Sign Up</a>
            </div>
        </nav>
```

5. MainServlet의 alias를 main으로 지정하며 다음과 같이 작성

```
package web.controller;

import java.io.IOException;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * Servlet implementation class MainServlet
```

```
 */
public class MainServlet extends HttpServlet {

	protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		myService(request, response);
	}


	protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		myService(request, response);
	}

	protected void myService(HttpServletRequest request,
HttpServletResponse response) throws ServletException, IOException {
		request.setCharacterEncoding("utf-8");
		String sign=request.getParameter("sign");
		if(sign==null) {
			RequestDispatcher
disp=request.getRequestDispatcher("jsp/error.jsp");
			request.setAttribute("msg", "invalid request sign");
			disp.forward(request, response);
		}else {
			if(sign.equals("boardList")) {
				RequestDispatcher
disp=request.getRequestDispatcher("jsp/boardList.jsp");
				request.setAttribute("msg", "ok");
				disp.forward(request, response);
			}
		}
	}
}
```

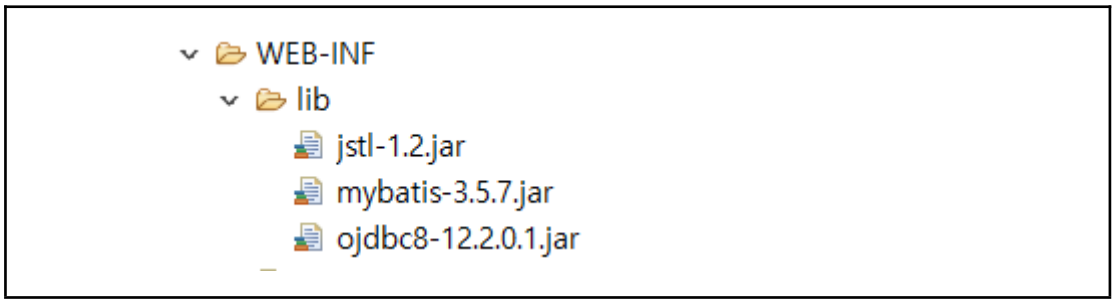
6. webapp/jsp 폴더를 만들고 error.jsp와 boardList.jsp를 간단히 만든다 ⇒ 여기까지 확인

7. DB에 다음과 같이 Board라는 테이블을 만든다

```
CREATE SEQUENCE  board_seq
START WITH 1
INCREMENT BY 1;



CREATE TABLE BOARD
(
  ARTICLENO NUMBER(10) NOT NULL
, PARENTNO NUMBER(10) DEFAULT 0 NOT NULL
, TITLE VARCHAR2(500) NOT NULL
, CONTENT VARCHAR2(4000)
, IMAGEFILENAME VARCHAR2(100)
```

```
, WRITEDATE DATE DEFAULT current_date NOT NULL
, ID VARCHAR2(20)
, CONSTRAINT BOARD_PK PRIMARY KEY
  (
    ARTICLENO
  )
  ENABLE
);



CREATE OR REPLACE TRIGGER BOARD_TRIGGER
BEFORE INSERT ON BOARD
REFERENCING NEW AS NEW
FOR EACH ROW
BEGIN
SELECT BOARD_SEQ.NEXTVAL INTO :NEW.ARTICLENO FROM dual;
END;

ALTER TRIGGER board_trigger ENABLE;


Insert into BOARD
(ARTICLENO,PARENTNO,TITLE,CONTENT,IMAGEFILENAME,WRITEDATE,ID)
values (1,0,'test1','test1',null,to_date('21/04/07','RR/MM/DD'),'a');
Insert into BOARD
(ARTICLENO,PARENTNO,TITLE,CONTENT,IMAGEFILENAME,WRITEDATE,ID)
values (2,1,'test1','테스트 성공했나요?',null,to_date('21/04/07','RR/MM/DD'),'a');


SELECT * FROM BOARD;

select level,
articleNO,
parentNO,
LPAD(' ', 4*(LEVEL-1)) || title title,
content,
writeDate,
imageFileName,
id
from board
start with parentNO=0
connect by prior articleNO=parentNO
order siblings by articleNO desc;


drop table board cascade CONSTRAINTS;
```

8. maven repository에서 다음 3개의 라이브러리를 다운 받아 WEB-INF/lib 폴더에 넣는다



9. webapp/mybatis 폴더를 만들고 MyBatisConfig.xml을 다음과 같이 작성한다

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE configuration
  PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
        <typeAliases>
                <typeAlias type="web.vo.BoardVO" alias="boardVO"/>
        </typeAliases>

  <environments default="development">
    <environment id="development">
      <transactionManager type="JDBC"/>
      <dataSource type="POOLED">
        <property name="driver" value="oracle.jdbc.driver.OracleDriver"/>
        <property name="url" value="jdbc:oracle:thin:@localhost:1521:XE"/>
        <property name="username" value="JES"/>
        <property name="password" value="1234"/>
      </dataSource>
    </environment>
  </environments>
  <mappers>
    <mapper resource="web/dao/board.xml"/>
  </mappers>
</configuration>
```

10. web.vo.BoardVO.java 작성

```
package web.vo;

import java.util.Date;

public class BoardVO {
        private int level,articleNO,parentNO;
        private String title,content,imageFileName,id;
        Date writeDate;
```

```
	public int getLevel() {
		return level;
	}
	public void setLevel(int level) {
		this.level = level;
	}
	public int getArticleNO() {
		return articleNO;
	}
	public void setArticleNO(int articleNO) {
		this.articleNO = articleNO;
	}
	public int getParentNO() {
		return parentNO;
	}
	public void setParentNO(int parentNO) {
		this.parentNO = parentNO;
	}
	public String getTitle() {
		return title;
	}
	public void setTitle(String title) {
		this.title = title;
	}
	public String getContent() {
		return content;
	}
	public void setContent(String content) {
		this.content = content;
	}
	public String getImageFileName() {
		return imageFileName;
	}
	public void setImageFileName(String imageFileName) {
		this.imageFileName = imageFileName;
	}
	public String getId() {
		return id;
	}
	public void setId(String id) {
		this.id = id;
	}

	public Date getWriteDate() {
		return writeDate;
	}
	public void setWriteDate(Date writeDate) {
		this.writeDate = writeDate;
	}
	@Override
	public String toString() {
		return "ArticleVO [level=" + level + ", articleNO=" + articleNO + ",
parentNO=" + parentNO + ", title=" + title
				+ ", content=" + content + ", imageFileName=" +
```

```
imageFileName + ", id=" + id + ", writeDate="
                                        + writeDate + "]";
        }

}
```

11. web.dao.BoardDAO.java 작성

```
package web.dao;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileWriter;
import java.util.List;
import java.util.Map;

import org.apache.ibatis.io.Resources;
import org.apache.ibatis.session.SqlSession;
import org.apache.ibatis.session.SqlSessionFactory;
import org.apache.ibatis.session.SqlSessionFactoryBuilder;

import web.vo.BoardVO;


public class BoardDAO {


        SqlSession sqlSession;

        public BoardDAO(BufferedReader br) {
                try {

                        SqlSessionFactory factory=new
SqlSessionFactoryBuilder().build(br);
                        sqlSession=factory.openSession();
                        System.out.println(sqlSession);
                }catch(Exception e) {
                        e.printStackTrace();
                }
        }

        public List<BoardVO> boardList() {
                List<BoardVO>
list=sqlSession.selectList("mapper.board.boardList");
                System.out.println(list.size());
                return list;
        }

}
```

12. DAO와 같은 폴더에 board.xml 작성

```
<?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="mapper.board">

        <select id="boardList" resultType="boardVO">
                <![CDATA[
                        select level,
                                articleNO,
                                parentNO,
                                LPAD(' ', 4*(LEVEL-1)) || title title,
                                content,
                                writeDate,
                                imageFileName,
                                id
                        from board
                        start with parentNO=0
                        connect by prior articleNO=parentNO
                        order siblings by articleNO desc

                ]]>
        </select>

        <insert id="insert" parameterType="java.util.Map">
                <![CDATA[
                        insert into board(title,content,imageFileName,id,parentNO)
values
                        (#{title},#{content},#{imageFileName},#{id},#{parentNO} )

                ]]>
        </insert>

        <select id="selectBoard" resultType="boardVO" parameterType="int">
                <![CDATA[
                        select * from board
                        where articleNO=#{articleNO}
                ]]>
        </select>


</mapper>
```

13. web.service.BoardService.java 작성

```
package web.service;
```

```
import java.io.BufferedReader;
import java.io.File;
import java.util.List;

import web.dao.BoardDAO;
import web.vo.BoardVO;

public class BoardService {
	BoardDAO boardDAO;
	public BoardService(BufferedReader br) {
		boardDAO=new BoardDAO(br);
	}

	public List<BoardVO> boardList(){
		return boardDAO.boardList();
	}
}
```

14. MainServlet을 다음과 같이 수정

```
package web.controller;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.List;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import web.service.BoardService;
import web.vo.BoardVO;

/**
 * Servlet implementation class MainServlet
 */
public class MainServlet extends HttpServlet {

	BoardService boardService;

	public void init() {

		InputStream in =
getServletContext().getResourceAsStream("mybatis/MyBatisConfig.xml");
		BufferedReader br = new BufferedReader(new
```

```
InputStreamReader(in));



                boardService = new BoardService(br);

        }

        protected void doGet(HttpServletRequest request, HttpServletResponse
response)
                        throws ServletException, IOException {
                myService(request, response);
        }

        protected void doPost(HttpServletRequest request, HttpServletResponse
response)
                        throws ServletException, IOException {
                myService(request, response);
        }

        protected void myService(HttpServletRequest request,
HttpServletResponse response)
                        throws ServletException, IOException {
                request.setCharacterEncoding("utf-8");
                String sign = request.getParameter("sign");
                if (sign == null) {
                        RequestDispatcher disp =
request.getRequestDispatcher("jsp/error.jsp");
                        request.setAttribute("msg", "invalid request sign");
                        disp.forward(request, response);
                } else {
                        if (sign.equals("boardList")) {
                                List<BoardVO> list = boardService.boardList();
                                //System.out.println(list.size());
                                RequestDispatcher disp =
request.getRequestDispatcher("jsp/boardList.jsp");

                                request.setAttribute("articlesList", list);
                                disp.forward(request, response);
                        }
                }

        }

}
```

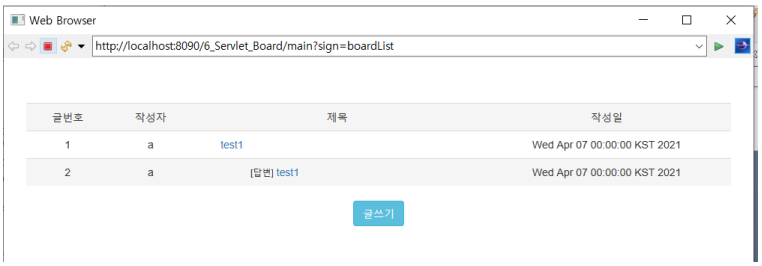
15. boardList.jsp를 다음과 같이 수정

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"
```

```
   isELIgnored="false" %>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<c:set var="contextPath"  value="${pageContext.request.contextPath}"  />
<%
  request.setCharacterEncoding("UTF-8");
%>
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script
>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/jquery-cookie/1.4.1/jquery.cookie.min.js"
></script>
 <style>
   .cls1 {text-decoration:none;}
   .cls2{text-align:center; font-size:30px;}
  </style>
  <meta charset="UTF-8">
  <title>글목록창</title>
</head>
<script>
          function fn_articleForm(articleForm){
                    var login=$.cookie('logined');
            if(login){
              location.href=articleForm;
            }else{
              alert("로그인 후 글쓰기가 가능합니다.")
              window.close();
            }
          }
</script>
<body>
<br><br><br>
<div class="container">
<table align="center" border="0"  width="80%"  class="table table-striped">
  <tr height="10" align="center"  bgcolor="lightgray">
    <td width="10%">글번호</td>
    <td >작성자</td>
    <td >제목</td>
    <td >작성일</td>
  </tr>
<c:choose>
  <c:when test="${articlesList ==null }" >
    <tr  height="10">
      <td colspan="4">
```

```
        <p align="center">
          <b><span style="font-size:9pt;">등록된 글이 없습니다.</span></b>
        </p>
      </td>
    </tr>
  </c:when>
  <c:when test="${articlesList !=null }" >
    <c:forEach  var="article" items="${articlesList }" varStatus="articleNum" >
     <tr align="center">
          <td width="5%">${articleNum.count}</td>
          <td width="10%">${article.id }</td>
          <td align='left'  width="35%">
            <span style="padding-right:30px"></span>
             <c:choose>
               <c:when test='${article.level > 1 }'>
                 <c:forEach begin="1" end="${article.level }" step="1">
                     <span style="padding-left:20px"></span>
                 </c:forEach>
                 <span style="font-size:12px;">[답변]</span>
                <a class='cls1'
href="../viewArticle?articleNO=${article.articleNO}">${article.title}</a>
                 </c:when>
                 <c:otherwise>
                   <a class='cls1'
href="../viewArticle?articleNO=${article.articleNO}">${article.title }</a>
                 </c:otherwise>
               </c:choose>
            </td>
            <td  width="30%">${article.writeDate}</td>
          </tr>
    </c:forEach>
     </c:when>
    </c:choose>
</table>
</div>
<!-- <a  class="cls1"  href="#"><p class="cls2">글쓰기</p></a> -->
<center>
<a    href="javascript:fn_articleForm('../boardWriteForm')"><p class="btn
btn-info">글쓰기</p></a>
</center>
</body>
</html>
```

16. 결과 확인



8. 스프링으로 게시판 작성

1. File>New>Other…>Spring>Spring Legacy Project>Spring MVC Project
2. pom.xml에 다음 4개의 lib를 설정한다

```
<!-- https://mvnrepository.com/artifact/org.mybatis/mybatis -->
<dependency>
    <groupId>org.mybatis</groupId>
    <artifactId>mybatis</artifactId>
    <version>3.5.7</version>
</dependency>

<!-- https://mvnrepository.com/artifact/org.mybatis/mybatis-spring -->
<dependency>
    <groupId>org.mybatis</groupId>
    <artifactId>mybatis-spring</artifactId>
    <version>2.0.6</version>
</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-jdbc -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-jdbc</artifactId>
    <version>5.3.9</version>
</dependency>


<!-- https://mvnrepository.com/artifact/com.oracle.database.jdbc/ojdbc8 -->
<dependency>
    <groupId>com.oracle.database.jdbc</groupId>
    <artifactId>ojdbc8</artifactId>
    <version>21.1.0.0</version>
</dependency>
```

3. web.xml을 다음과 같이 수정한다

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://Java.sun.com/xml/ns/javaee
https://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">

        <!-- The definition of the Root Spring Container shared by all Servlets and
```

```
Filters -->
	<context-param>
		<param-name>contextConfigLocation</param-name>
		<param-value>
			/WEB-INF/spring/root-context.xml
			/WEB-INF/spring/spring-mybatis.xml
		</param-value>
	</context-param>

	<!-- Creates the Spring Container shared by all Servlets and Filters -->
	<listener>

<listener-class>org.springframework.web.context.ContextLoaderListener</listener-
class>
	</listener>

	<!-- Processes application requests -->
	<servlet>
		<servlet-name>appServlet</servlet-name>

<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
		<init-param>
			<param-name>contextConfigLocation</param-name>

<param-value>/WEB-INF/spring/appServlet/servlet-context.xml</param-value>
		</init-param>
		<load-on-startup>1</load-on-startup>
	</servlet>

	<servlet-mapping>
		<servlet-name>appServlet</servlet-name>
		<url-pattern>/</url-pattern>
	</servlet-mapping>

</web-app>
```

4. WEB-INF/spring 폴더에 spring-mybatis.xml을 만든다

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/mvc"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xmlns:beans="http://www.springframework.org/schema/beans"
	xmlns:context="http://www.springframework.org/schema/context"
	xsi:schemaLocation="http://www.springframework.org/schema/mvc
https://www.springframework.org/schema/mvc/spring-mvc.xsd
		http://www.springframework.org/schema/beans
https://www.springframework.org/schema/beans/spring-beans.xsd
		http://www.springframework.org/schema/context
https://www.springframework.org/schema/context/spring-context.xsd">

	<beans:bean id="dataSource"
```

```
class="org.apache.ibatis.datasource.pooled.PooledDataSource">
		<beans:property name="driver"
value="oracle.jdbc.driver.OracleDriver"/>
		<beans:property name="url"
value="jdbc:oracle:thin:@localhost:1521:XE"></beans:property>
		<beans:property name="username"
value="JES"></beans:property>
		<beans:property name="password"
value="1234"></beans:property>

	</beans:bean>

	<beans:bean id="sqlSessionFactory"
class="org.mybatis.spring.SqlSessionFactoryBean">
		<beans:property name="dataSource"
ref="dataSource"></beans:property>
		<beans:property name="configLocation"
value="classpath:mybatis/MyBatisConfig.xml"></beans:property>
		<beans:property name="mapperLocations"
value="classpath:mybatis/mappers/*.xml"></beans:property>
	</beans:bean>

	<beans:bean id="sqlSession"
class="org.mybatis.spring.SqlSessionTemplate">
		<beans:constructor-arg index="0"
ref="sqlSessionFactory"></beans:constructor-arg>

	</beans:bean>
</beans:beans>
```

5. src/main/resources 밑에 mybatis라는 package를 생성하고 MyBatisConfig.xml을 다음과 같이 작성해 놓는다

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE configuration
  PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
	<typeAliases>
		<typeAlias type="my.jes.web.vo.BoardVO" alias="boardVO"/>
	</typeAliases>

</configuration>
```

6. src/main/resources/mybatis 밑에 mappers라는 package를 생성하고 board.xml을 다음과 같이 작성한다

```
<?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="mapper.board">

        <select id="boardList" resultType="boardVO">
                <![CDATA[
                        select level,
                                articleNO,
                                parentNO,
                                LPAD(' ', 4*(LEVEL-1)) || title title,
                                content,
                                writeDate,
                                imageFileName,
                                id
                        from board
                        start with parentNO=0
                        connect by prior articleNO=parentNO
                        order siblings by articleNO desc

                ]]>
        </select>

        <insert id="insert" parameterType="java.util.Map">
                <![CDATA[
                        insert into board(title,content,imageFileName,id,parentNO)
values
                        (#{title},#{content},#{imageFileName},#{id},#{parentNO} )

                ]]>
        </insert>

        <select id="selectArticle" resultType="boardVO" parameterType="int">
                <![CDATA[
                        select * from board
                        where articleNO=#{articleNO}
                ]]>
        </select>


</mapper>
```

7. my.jes.web.controller.BoardController.java를 작성한다

```
package my.jes.web.controller;

import java.text.DateFormat;
```

```
import java.util.Date;
import java.util.List;
import java.util.Locale;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;

import my.jes.web.service.BoardService;
import my.jes.web.vo.BoardVO;

/**
 * Handles requests for the application home page.
 */
@Controller
public class BoardController {

	@Autowired
	BoardService boardService;

	@RequestMapping(value = "boardList", method = RequestMethod.GET)
	public String boardList(Locale locale, Model model) {
		List<BoardVO> list=boardService.boardList();
		model.addAttribute("articlesList", list);
		return "boardList";
	}

}
```

8. my.jes.web.service.BoardService.java 를 작성한다

```
package my.jes.web.service;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import my.jes.web.dao.BoardDAO;
import my.jes.web.vo.BoardVO;

@Service
public class BoardService {
	@Autowired
	BoardDAO boardDAO;
```

```
	public List<BoardVO> boardList(){
		return boardDAO.boardList();
	}
}
```

9. my.jes.web.dao.BoardDAO.java를 작성한다

```
package my.jes.web.dao;

import java.io.BufferedReader;
import java.io.FileWriter;
import java.util.List;
import java.util.Map;

import org.apache.ibatis.session.SqlSession;
import org.apache.ibatis.session.SqlSessionFactory;
import org.apache.ibatis.session.SqlSessionFactoryBuilder;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;

import my.jes.web.vo.BoardVO;

@Repository
public class BoardDAO {

	@Autowired
	SqlSession sqlSession;



	public List<BoardVO> boardList() {
		List<BoardVO>
list=sqlSession.selectList("mapper.board.boardList");
		System.out.println(list.size());
		return list;
	}

	public void boardWrite(Map<String, Object> articleMap) {
		sqlSession.insert("mapper.board.insert", articleMap);

	}


	public BoardVO viewArticle(int articleNO){
		return sqlSession.selectOne("mapper.board.selectArticle",
articleNO);
	}
```

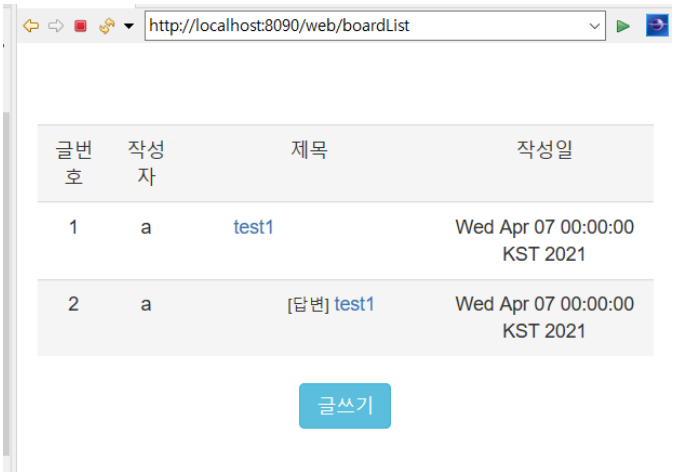
```
}
```

10. my.jes.web.vo.BoardVO.java를 작성한다

```
package my.jes.web.vo;

import java.util.Date;

public class BoardVO {
	private int level,articleNO,parentNO;
	private String title,content,imageFileName,id;
	Date writeDate;



	public int getLevel() {
		return level;
	}
	public void setLevel(int level) {
		this.level = level;
	}
	public int getArticleNO() {
		return articleNO;
	}
	public void setArticleNO(int articleNO) {
		this.articleNO = articleNO;
	}
	public int getParentNO() {
		return parentNO;
	}
	public void setParentNO(int parentNO) {
		this.parentNO = parentNO;
	}
	public String getTitle() {
		return title;
	}
	public void setTitle(String title) {
		this.title = title;
	}
	public String getContent() {
		return content;
	}
	public void setContent(String content) {
		this.content = content;
	}
	public String getImageFileName() {
		return imageFileName;
	}
	public void setImageFileName(String imageFileName) {
```

```
                this.imageFileName = imageFileName;
        }
        public String getId() {
                return id;
        }
        public void setId(String id) {
                this.id = id;
        }

        public Date getWriteDate() {
                return writeDate;
        }
        public void setWriteDate(Date writeDate) {
                this.writeDate = writeDate;
        }
        @Override
        public String toString() {
                return "ArticleVO [level=" + level + ", articleNO=" + articleNO + ",
parentNO=" + parentNO + ", title=" + title
                                + ", content=" + content + ", imageFileName=" +
imageFileName + ", id=" + id + ", writeDate="
                                + writeDate + "]";
        }

}
```

11. 수행결과



9. SpringBoot로 웹 프로젝트 만들기

1. STS에서 New Spring Starter Project 에서 dependency에 H2, JDBC, Oracle, MySQL, MyBatis, Spring Web, Spring Boot DevTools 를 선택함
2. application.properties 파일에 다음 작성

```
server.port=7777
```

3. HomeController.java를 다음과 같이 작성

```
package com.example.web.controller;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;

@Controller
public class HomeController {

	@ResponseBody
	@RequestMapping("home")
	public String home() {
		return "여기는 home()입니다";
	}

}
```

4. web browser로 http://localhost:7777/home 요청

5. 이클립스 마켓플레이스에서 Eclipse Web Developer Tools 설치

6. https://startbootstrap.com/template/shop-homepage 에서 샘플 웹 리소스를 다운받아 src/main/resources/static에 저장

7. src/main/resources/static에 index.html을 다음과 같이 수정

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no" />
    <meta name="description" content="" />
    <meta name="author" content="" />
    <title>Shop Homepage - Start Bootstrap Template</title>
    <!-- Favicon-->
    <link rel="icon" type="image/x-icon" href="assets/favicon.ico" />
```

```
        <!-- Bootstrap icons-->
        <link
href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.5.0/font/bootstrap-icons.css"
rel="stylesheet" />
        <!-- Core theme CSS (includes Bootstrap)-->
        <link href="css/styles.css" rel="stylesheet" />
    </head>
    <body>
        <!-- Navigation-->
        <nav class="navbar navbar-expand-lg navbar-light bg-light">
            <div class="container px-4 px-lg-5">
                <a class="navbar-brand" href="#!">Start Bootstrap</a>
                <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
data-bs-target="#navbarSupportedContent"
aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation"><span class="navbar-toggler-icon"></span></button>
                <div class="collapse navbar-collapse" id="navbarSupportedContent">
                    <ul class="navbar-nav me-auto mb-2 mb-lg-0 ms-lg-4">
                        <li class="nav-item"><a class="nav-link active" aria-current="page"
href="#!">Home</a></li>
                        <li class="nav-item"><a class="nav-link" href="#!"
onclick='window.open("html/memberInsert.html", "_blank",
"toolbar=yes,scrollbars=yes,resizable=yes,top=500,left=500,width=400,height=400
");'>About</a></li>
                        <li class="nav-item dropdown">
                            <a class="nav-link dropdown-toggle" id="navbarDropdown"
href="#" role="button" data-bs-toggle="dropdown"
aria-expanded="false">Shop</a>
                            <ul class="dropdown-menu" aria-labelledby="navbarDropdown">
                                <li><a class="dropdown-item" href="#!">All Products</a></li>
                                <li><hr class="dropdown-divider" /></li>
                                <li><a class="dropdown-item" href="#!">Popular
Items</a></li>
                                <li><a class="dropdown-item" href="#!">New Arrivals</a></li>
                            </ul>
                        </li>
                    </ul>
                    <form class="d-flex">
                        <button class="btn btn-outline-dark" type="submit">
                            <i class="bi-cart-fill me-1"></i>
                            Cart
                            <span class="badge bg-dark text-white ms-1
rounded-pill">0</span>
                        </button>
                    </form>
                </div>
            </div>
        </nav>
        <!-- Header-->
        <header class="bg-dark py-5">
            <div class="container px-4 px-lg-5 my-5">
                <img src="img/title.gif" width="500">
            </div>
        </header>
```

```
        <!-- Section-->
        <section class="py-5">
            <div class="container px-4 px-lg-5 mt-5">
                <div class="row gx-4 gx-lg-5 row-cols-2 row-cols-md-3 row-cols-xl-4
justify-content-center">
                    <div class="col mb-5">
                        <div class="card h-100">
                            <!-- Product image-->
                            <img class="card-img-top" src="img/americano.jpg" alt="..." />
                            <!-- Product details-->
                            <div class="card-body p-4">
                                <div class="text-center">
                                    <!-- Product name-->
                                    <h5 class="fw-bolder">Fancy Product</h5>
                                    <!-- Product price-->
                                    $40.00 - $80.00
                                </div>
                            </div>
                            <!-- Product actions-->
                            <div class="card-footer p-4 pt-0 border-top-0 bg-transparent">
                                <div class="text-center"><a class="btn btn-outline-dark
mt-auto" href="#">View options</a></div>
                            </div>
                        </div>
                    </div>
                    <div class="col mb-5">
                        <div class="card h-100">
                            <!-- Sale badge-->
                            <div class="badge bg-dark text-white position-absolute"
style="top: 0.5rem; right: 0.5rem">Sale</div>
                            <!-- Product image-->
                            <img class="card-img-top" src="img/latte.jpg" alt="..." />
                            <!-- Product details-->
                            <div class="card-body p-4">
                                <div class="text-center">
                                    <!-- Product name-->
                                    <h5 class="fw-bolder">Special Item</h5>
                                    <!-- Product reviews-->
                                    <div class="d-flex justify-content-center small text-warning
mb-2">
                                        <div class="bi-star-fill"></div>
                                        <div class="bi-star-fill"></div>
                                        <div class="bi-star-fill"></div>
                                        <div class="bi-star-fill"></div>
                                        <div class="bi-star-fill"></div>
                                    </div>
                                    <!-- Product price-->
                                    <span class="text-muted
text-decoration-line-through">$20.00</span>
                                    $18.00
                                </div>
                            </div>
                            <!-- Product actions-->
                            <div class="card-footer p-4 pt-0 border-top-0 bg-transparent">
```

```
                        <div class="text-center"><a class="btn btn-outline-dark
mt-auto" href="#">Add to cart</a></div>
                    </div>
                </div>
            </div>
            <div class="col mb-5">
                <div class="card h-100">
                    <!-- Sale badge-->
                    <div class="badge bg-dark text-white position-absolute"
style="top: 0.5rem; right: 0.5rem">Sale</div>
                    <!-- Product image-->
                    <img class="card-img-top" src="img/moca.jpg" alt="..." />
                    <!-- Product details-->
                    <div class="card-body p-4">
                        <div class="text-center">
                            <!-- Product name-->
                            <h5 class="fw-bolder">Sale Item</h5>
                            <!-- Product price-->
                            <span class="text-muted
text-decoration-line-through">$50.00</span>
                            $25.00
                        </div>
                    </div>
                    <!-- Product actions-->
                    <div class="card-footer p-4 pt-0 border-top-0 bg-transparent">
                        <div class="text-center"><a class="btn btn-outline-dark
mt-auto" href="#">Add to cart</a></div>
                    </div>
                </div>
            </div>
            <div class="col mb-5">
                <div class="card h-100">
                    <!-- Product image-->
                    <img class="card-img-top" src="img/sugar.jpg" alt="..."
height="160"/>
                    <!-- Product details-->
                    <div class="card-body p-4">
                        <div class="text-center">
                            <!-- Product name-->
                            <h5 class="fw-bolder">Popular Item</h5>
                            <!-- Product reviews-->
                            <div class="d-flex justify-content-center small text-warning
mb-2">
                                <div class="bi-star-fill"></div>
                                <div class="bi-star-fill"></div>
                                <div class="bi-star-fill"></div>
                                <div class="bi-star-fill"></div>
                                <div class="bi-star-fill"></div>
                            </div>
                            <!-- Product price-->
                            $40.00
                        </div>
                    </div>
                    <!-- Product actions-->
```

```
                    <div class="card-footer p-4 pt-0 border-top-0 bg-transparent">
                        <div class="text-center"><a class="btn btn-outline-dark
mt-auto" href="#">Add to cart</a></div>
                    </div>
                </div>
            </div>
            <div class="col mb-5">
                <div class="card h-100">
                    <!-- Sale badge-->
                    <div class="badge bg-dark text-white position-absolute"
style="top: 0.5rem; right: 0.5rem">Sale</div>
                    <!-- Product image-->
                    <img class="card-img-top"
src="https://dummyimage.com/450x300/dee2e6/6c757d.jpg" alt="..." />
                    <!-- Product details-->
                    <div class="card-body p-4">
                        <div class="text-center">
                            <!-- Product name-->
                            <h5 class="fw-bolder">Sale Item</h5>
                            <!-- Product price-->
                            <span class="text-muted
text-decoration-line-through">$50.00</span>
                            $25.00
                        </div>
                    </div>
                    <!-- Product actions-->
                    <div class="card-footer p-4 pt-0 border-top-0 bg-transparent">
                        <div class="text-center"><a class="btn btn-outline-dark
mt-auto" href="#">Add to cart</a></div>
                    </div>
                </div>
            </div>
            <div class="col mb-5">
                <div class="card h-100">
                    <!-- Product image-->
                    <img class="card-img-top"
src="https://dummyimage.com/450x300/dee2e6/6c757d.jpg" alt="..." />
                    <!-- Product details-->
                    <div class="card-body p-4">
                        <div class="text-center">
                            <!-- Product name-->
                            <h5 class="fw-bolder">Fancy Product</h5>
                            <!-- Product price-->
                            $120.00 - $280.00
                        </div>
                    </div>
                    <!-- Product actions-->
                    <div class="card-footer p-4 pt-0 border-top-0 bg-transparent">
                        <div class="text-center"><a class="btn btn-outline-dark
mt-auto" href="#">View options</a></div>
                    </div>
                </div>
            </div>
            <div class="col mb-5">
```

```
                    <div class="card h-100">
                        <!-- Sale badge-->
                        <div class="badge bg-dark text-white position-absolute"
style="top: 0.5rem; right: 0.5rem">Sale</div>
                        <!-- Product image-->
                        <img class="card-img-top"
src="https://dummyimage.com/450x300/dee2e6/6c757d.jpg" alt="..." />
                        <!-- Product details-->
                        <div class="card-body p-4">
                            <div class="text-center">
                                <!-- Product name-->
                                <h5 class="fw-bolder">Special Item</h5>
                                <!-- Product reviews-->
                                <div class="d-flex justify-content-center small text-warning
mb-2">
                                    <div class="bi-star-fill"></div>
                                    <div class="bi-star-fill"></div>
                                    <div class="bi-star-fill"></div>
                                    <div class="bi-star-fill"></div>
                                    <div class="bi-star-fill"></div>
                                </div>
                                <!-- Product price-->
                                <span class="text-muted
text-decoration-line-through">$20.00</span>
                                $18.00
                            </div>
                        </div>
                        <!-- Product actions-->
                        <div class="card-footer p-4 pt-0 border-top-0 bg-transparent">
                            <div class="text-center"><a class="btn btn-outline-dark
mt-auto" href="#">Add to cart</a></div>
                        </div>
                    </div>
                </div>
                <div class="col mb-5">
                    <div class="card h-100">
                        <!-- Product image-->
                        <img class="card-img-top"
src="https://dummyimage.com/450x300/dee2e6/6c757d.jpg" alt="..." />
                        <!-- Product details-->
                        <div class="card-body p-4">
                            <div class="text-center">
                                <!-- Product name-->
                                <h5 class="fw-bolder">Popular Item</h5>
                                <!-- Product reviews-->
                                <div class="d-flex justify-content-center small text-warning
mb-2">
                                    <div class="bi-star-fill"></div>
                                    <div class="bi-star-fill"></div>
                                    <div class="bi-star-fill"></div>
                                    <div class="bi-star-fill"></div>
                                    <div class="bi-star-fill"></div>
                                </div>
                                <!-- Product price-->
```

```
                                $40.00
                            </div>
                        </div>
                        <!-- Product actions-->
                        <div class="card-footer p-4 pt-0 border-top-0 bg-transparent">
                            <div class="text-center"><a class="btn btn-outline-dark
mt-auto" href="#">Add to cart</a></div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </section>
    <!-- Footer-->
    <footer class="py-5 bg-dark">
        <div class="container"><p class="m-0 text-center text-white">Copyright
&copy; Your Website 2021</p></div>
    </footer>
    <!-- Bootstrap core JS-->
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.0/dist/js/bootstrap.bundle.min.js"><
/script>
    <!-- Core theme JS-->
    <script src="js/scripts.js"></script>
  </body>
</html>
```

8. img 폴더에 적당한 이미지들을 넣고 화면 꾸밈
9. src/main/resources/static 밑에 html 폴더를 만들고 memberInsert.html을 다음과 같이 작성

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
  <form action="../memberInsert">
        ID<input name="id"><br>
        PW<input name="pw" type="password"><br>
        Name<input name="name"><br>
        <input type="submit" value="회원가입">
  </form>
</body>
</html>
```

10. com.example.web.controller.MemberController.java를 다음과 같이 작성

```
package com.example.web.controller;

import javax.servlet.http.HttpServletRequest;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.example.web.service.MemberService;
import com.example.web.vo.Member;

@RestController
public class MemberController {

	@Autowired
	MemberService memberService;


	@RequestMapping("memberInsert")
	public String memberInsert(HttpServletRequest request) {
		String id=request.getParameter("id");
		String pw=request.getParameter("pw");
		String name=request.getParameter("name");

		Member m=new Member(id, pw, name);
		System.out.println(m);

		memberService.memberInsert(m);

		return "회원 가입 완료 <button onclick='window.close()'
>닫기</button>";
	}

}
```

11. com.example.web.service.MemberService.java를 다음과 같이 작성

```
package com.example.web.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.example.web.dao.MemberDAO;
import com.example.web.vo.Member;

@Service
public class MemberService {

	@Autowired
```

```
	MemberDAO memberDAO;


	public void memberInsert(Member m) {
		memberDAO.memberInsert(m);

	}

}
```

12. com.example.web.dao.MemberDAO.java를 다음과 같이 작성

```
package com.example.web.dao;

import org.apache.ibatis.annotations.Mapper;
import org.springframework.dao.DataAccessException;
import org.springframework.stereotype.Repository;

import com.example.web.vo.Member;

@Mapper
@Repository
public interface MemberDAO {

	public void memberInsert(Member m) throws DataAccessException;

}
```

13. com.example.web.vo.Member.java를 다음과 같이 작성

```
package com.example.web.vo;

public class Member {

	private String id,pw,name;

	public Member() {
		super();
		// TODO Auto-generated constructor stub
	}

	public Member(String id, String pw) {
		super();
		setId(id);
		setPw(pw);
	}
```

```
	public Member(String id, String pw, String name) {
		this(id,pw);
		setName(name);
	}

	public String getId() {
		return id;
	}

	public void setId(String id) {
		this.id = id;
	}

	public String getPw() {
		return pw;
	}

	public void setPw(String pw) {
		this.pw = pw;
	}

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}

	@Override
	public String toString() {
		return "Member [id=" + id + ", pw=" + pw + ", name=" + name + "]";
	}

}
```

14. application.properties에 컨넥션, 마이바티스 관련 설정을 추가

```
server.port=9999

spring.datasource.url=jdbc:oracle:thin:@localhost:1521:XE
spring.datasource.username=human
spring.datasource.password=1234
spring.datasource.driver-class-name=oracle.jdbc.driver.OracleDriver

mybatis.config-location=classpath:mybatis-config.xml

mybatis.type-aliases-package=com.example.web.vo
```

15. src/main/resources 밑에 mybatis-config.xml을 생성

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration
  PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-config.dtd">

  <configuration>
        <mappers>
              <mapper resource="mybatis/mappers/member.xml" />
        </mappers>

  </configuration>
```

16. src/main/resources밑에 mybatis/mappers 라는 서브 디렉토리를 차례로 만들고 여기에 member.xml을 작성해 놓는다 (스프링부트에서는 이름 규칙을 철저히 지켜야 한다)

```
<?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="com.example.web.dao.MemberDAO">

        <insert id="memberInsert" parameterType="member">
              insert into Member(id,pw,name) values( #{id}, #{pw}, #{name} )

        </insert>

</mapper>
```

17. 수행하여 회원 가입이 되는지 본다
18. index.html에서 로그인 영역을 다음과 같이 지정한다

```
 <div id='loginDiv'>
              ID<input size='5'  id="login_id">
              PW<input size='5' id="login_pw" type="password">
              <button id="loginBtn">로그인</button>
</div>
```

19. index.html에 jQuery CDN과 my.js link를 다음과 같이 추가한다

```
  <script src="https://code.jquery.com/jquery-3.6.0.min.js"
integrity="sha256-/xUj+3OJU5yExlq6GSYGSHk7tPXikynS7ogEvDej/m4="
crossorigin="anonymous"></script>
```

```
<script src="js/my.js"></script>
```

20. js/my.js 작성

```
$(document).ready(function(){

	$("#loginBtn").click(function(){
		const id=$("#login_id").val();
		const pw=$("#login_pw").val();

		alert(id+":"+pw);
		$.post('login', {id,pw},
			function(data){
				console.log(data);
			});
	});
});
```

21. MemberController를 수정

```
package com.example.web.controller;

import javax.servlet.http.HttpServletRequest;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.example.web.service.MemberService;
import com.example.web.vo.Member;

@RestController
public class MemberController {

	@Autowired
	MemberService memberService;

	@PostMapping("login")
	public String login(@ModelAttribute Member m) {
		System.out.println(m);
		String name=memberService.login(m);
		return name+ "님 환영합니다";
	}
```

```
	@PostMapping("memberInsert")
	public String memberInsert(@ModelAttribute Member m) {
		System.out.println(m);
		memberService.memberInsert(m);
		return "회원 가입 완료 <button onclick='window.close()'
>닫기</button>";
	}

}
```

22. MemberService 수정

```
package com.example.web.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.example.web.dao.MemberDAO;
import com.example.web.vo.Member;

@Service
public class MemberService {

	@Autowired
	MemberDAO memberDAO;

	public void memberInsert(Member m) {
		memberDAO.memberInsert(m);
	}

	public String login(Member m) {
		return memberDAO.login(m);
	}

}
```

23. MemberDAO 수정

```
package com.example.web.dao;

import org.apache.ibatis.annotations.Mapper;
import org.springframework.dao.DataAccessException;
import org.springframework.stereotype.Repository;

import com.example.web.vo.Member;

@Mapper
@Repository
```

```
public interface MemberDAO {

	public void memberInsert(Member m) throws DataAccessException;

	public String login(Member m)  throws DataAccessException;

}
```

24. member.xml 수정

```
<?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="com.example.web.dao.MemberDAO">

	<insert id="memberInsert" parameterType="member">
		insert into Member(id,pw,name) values( #{id}, #{pw}, #{name} )
	</insert>

	<select id="login" parameterType="member"  resultType="String">
		select name from Member where id=#{id} and pw=#{pw}
	</select>

</mapper>
```

25. 로그인 확인
26. 오류 처리를 위해서 MemberService 수정

```
package com.example.web.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.example.web.dao.MemberDAO;
import com.example.web.vo.Member;

@Service
public class MemberService {

	@Autowired
	MemberDAO memberDAO;

	public void memberInsert(Member m)  throws Exception{
		memberDAO.memberInsert(m);
	}

	public String login(Member m)  throws Exception {
```

```
                return memberDAO.login(m);
        }

}
```

27. pom.xml에 json lib 추가

```
<dependency>
                <groupId>com.googlecode.json-simple</groupId>
                <artifactId>json-simple</artifactId>
                <version>1.1.1</version>
</dependency>
```

28. MemberController에서 JSON으로 응답하기 위해 수정

```
package com.example.web.controller;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;

import org.json.simple.JSONObject;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.example.web.service.MemberService;
import com.example.web.vo.Member;

@RestController
public class MemberController {

        @Autowired
        MemberService memberService;

        @PostMapping("login")
        public String login(@ModelAttribute Member m, HttpSession session) {
                JSONObject json=new JSONObject();
                try {
                        System.out.println(m);
                        String name=memberService.login(m);

                        if(name!=null) {//login ok
                                m.setName(name);
                                session.setAttribute("m", m);
                                json.put("name", name);
                        }else {
                                json.put("errMsg",  "로그인 오류");
```

```
                        }
                }catch(Exception e) {
                        json.put("errMsg", "로그인 오류");
                }

                return json.toJSONString();
        }


        @PostMapping("memberInsert")
        public String memberInsert(@ModelAttribute Member m) {
                System.out.println(m);
                try {
                        memberService.memberInsert(m);
                        return "회원 가입 완료 <button onclick='window.close()'
>닫기</button>";
                } catch (Exception e) {
                        e.printStackTrace();
                        return "회원가입실패: ID를 확인하세요";
                }

        }

}
```

29. my.js에서 쿠키 저장

```
$(document).ready(function(){

        $("#loginBtn").click(function(){
                const id=$("#login_id").val();
                const pw=$("#login_pw").val();

                alert(id+":"+pw);
                $.post('login', {id,pw},
                        function(data){
                                data=JSON.parse(data);
                                console.log(data);
                                if(data.errMsg){
                                        alert(data.errMsg);
                                }else{
                                        $("#loginDiv").html(data.name+"님
환영합니다");
                                        $.cookie("logined_id",id);
                                }
                        });
        });
});
```

30. index.html에서 jquery cookie cdn 추가

```
 <script src="https://code.jquery.com/jquery-3.6.0.min.js"
integrity="sha256-/xUj+3OJU5yExlq6GSYGSHk7tPXikynS7ogEvDej/m4="
crossorigin="anonymous"></script>
      <script
src="https://cdnjs.cloudflare.com/ajax/libs/jquery-cookie/1.4.1/jquery.cookie.min.js"
integrity="sha512-3j3VU6WC5rPQB4Ld1jnLV7Kd5xr+cq9avvhwqzbH/taCRNURoe
EpoPBK9pDyeukwSxwRPJ8fDgvYXd6SkaZ2TA==" crossorigin="anonymous"
referrerpolicy="no-referrer"></script>
      <script src="js/my.js"></script>
```

31. my.js에 로그아웃 버튼 추가

```
$("#loginBtn").click(function(){
                const id=$("#login_id").val();
                const pw=$("#login_pw").val();

                //alert(id+":"+pw);
                $.post('login', {id,pw},
                        function(data){
                                data=JSON.parse(data);
                                console.log(data);
                                if(data.errMsg){
                                        alert(data.errMsg);
                                }else{
                                        $("#loginDiv").html(data.name+"님
환영합니다<button id='logoutBtn'>로그아웃</button>");
                                        $.cookie("logined_id",id);
                                }
                        });
        });
```

32. my.js에서 로그아웃 처리

```
$(document).on("click", "#logoutBtn", function(event) { //로그아웃 처리

                $.post("logout",
                        {

                        },
                        function(){
                                $.removeCookie("logined_id");
                                location.reload();

                        }
                );//end post()
        });//end 로그아웃 처리
```

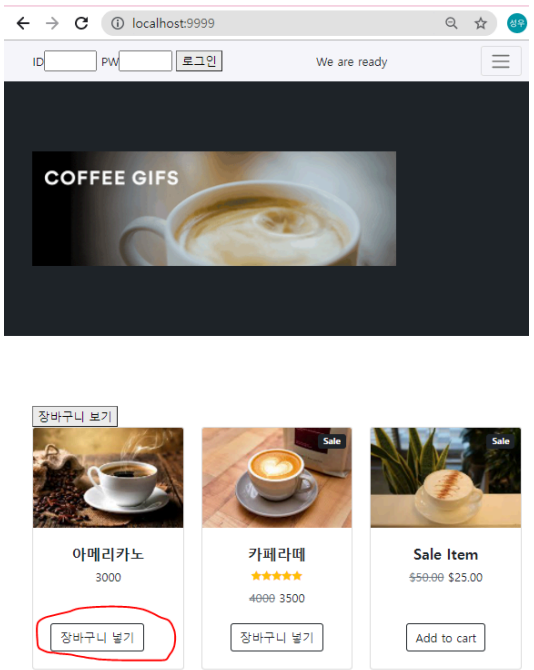
33. MemberController.java에서 로그아웃 처리

```
@PostMapping("logout")
	public String logout(HttpSession session) {
		session.invalidate();
		return "logout ok";
	}
```

34. 로그인 되어 있는 상태에서도 '새로고침'을 하면 로그인 폼이 보이는 문제 해결하기. my.js에서 다음을 추가 (logined_id라는 쿠키가 있으면 환영 메세지가 계속 보이게)

```
$(document).ready(function(){

  const logined_id=$.cookie("logined_id");
	if(logined_id){
		$("#loginDiv").html(logined_id+"님 환영합니다<button id='logoutBtn'>로그아웃</button>");
	}

...
```

35. 화면에 물품이 다음과 같이 리스팅 되어지게 index.html을 적당히 꾸민다




```
<div class="card h-100">
                        <!-- Product image-->
                        <img class="card-img-top"
src="img/americano.jpg" alt="..." />
                        <!-- Product details-->
                        <div class="card-body
p-4">
                            <div
class="text-center">
                                <!-- Product name-->
                                <input type="hidden"
id="product"  value="아메리카노" >
                                <h5
class="fw-bolder" >아메리카노</h5>
                                <!-- Product price-->
                                3000
                            </div>
                        </div>
                        <!-- Product actions-->
                        <div class="card-footer
p-4 pt-0 border-top-0 bg-transparent">
                            <div class="text-center
btn btn-outline-dark mt-auto" id="basketDiv"
>장바구니 넣기</div>
                        </div>
                    </div>
```

36. 첫번째 장바구니 넣기 버튼을 누르면 아메리카노가 장바구니 넣기 되도록 my.js에 다음과 같이 추가한다

```
function basketInsert(productName){
	//alert(productName);
	$.post('basketInsert', {productName},
			function(data){
				data=JSON.parse(data);
				alert(data.msg);
	});
}

$(document).ready(function(){


	$("#basketDiv").click(function(){
		basketInsert($("#product").val());
	});
```

37. com.example.web.controller.BasketController.java를 다음과 같이 작성한다

```
package com.example.web.controller;

import java.util.List;

import javax.servlet.http.HttpSession;

import org.json.simple.JSONObject;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RestController;

import com.example.web.service.BasketService;
import com.example.web.vo.Basket;
import com.example.web.vo.Member;

@RestController
public class BasketController {

	@Autowired
	BasketService basketService;

	@PostMapping("/basketInsert")
	public String basketInsert(@ModelAttribute Basket b, HttpSession session)
{
		JSONObject json=new JSONObject();
		Member m=(Member) session.getAttribute("m");
		if(m!=null) {
			b.setMemberId(m.getId());
```

```
                        System.out.println(b);
                        try {
                                basketService.basketInsert(b);
                                json.put("msg", "장바구니 넣기 완료");
                        } catch (Exception e) {
                                e.printStackTrace();
                                json.put("msg", "장바구니 넣기 실패");
                        }
                }else {
                        json.put("msg", "로그인 먼저 하세요");
                }
                return json.toJSONString();
        }

}
```

38. com.example.web.vo.Basket.java를 다음과 같이 작성한다

```
package com.example.web.vo;

public class Basket {

        private String memberId, productName;
        private int quantity;

        public Basket() {
                super();
                // TODO Auto-generated constructor stub
        }

        public Basket(String memberId, String productName, int quantity) {
                super();
                setMemberId(memberId);
                setProductName(productName);
                setQuantity(quantity);
        }

        public String getMemberId() {
                return memberId;
        }

        public void setMemberId(String memberId) {
                this.memberId = memberId;
        }

        public String getProductName() {
                return productName;
        }

        public void setProductName(String productName) {
                this.productName = productName;
```

```
	}

	public int getQuantity() {
		return quantity;
	}

	public void setQuantity(int quantity) {
		this.quantity = quantity;
	}

	@Override
	public String toString() {
		return "Basket [memberId=" + memberId + ", productName=" +
productName + ", quantity=" + quantity + "]";
	}
}
```

39. com.example.web.service.BasketService.java를 다음과 같이 작성한다

```
package com.example.web.service;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.dao.DataAccessException;
import org.springframework.stereotype.Service;
import com.example.web.dao.BasketDAO;
import com.example.web.vo.Basket;

@Service
public class BasketService {

	@Autowired
	BasketDAO basketDAO;

	public void basketInsert(Basket b) throws Exception{
		basketDAO.basketInsert(b);
	}
}
```

40. com.example.web.dao.BasketDAO.java를 interface로 작성한다

```
package com.example.web.dao;

import java.util.List;
import org.apache.ibatis.annotations.Mapper;
import org.springframework.dao.DataAccessException;
import org.springframework.stereotype.Repository;
import com.example.web.vo.Basket;
```

```
@Mapper
@Repository
public interface BasketDAO {

	public void basketInsert(Basket b) throws DataAccessException;
}
```

41. src/main/resources/mybatis/mappers에 basket.xml을 추가한다

```
<?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="com.example.web.dao.BasketDAO">

	<insert id="basketInsert" parameterType="basket">
		insert into Basket(memberId, productName, quantity)
		values( #{memberId}, #{productName}, 1 )
	</insert>

</mapper>
```

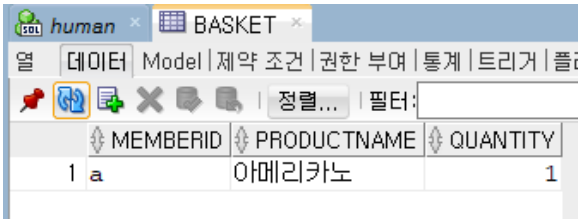
42. mybatis-config.xml에 basket.xml 등록

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration
 PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
 "http://mybatis.org/dtd/mybatis-3-config.dtd">

 <configuration>
	<mappers>
		<mapper resource="mybatis/mappers/member.xml" />
		<mapper resource="mybatis/mappers/basket.xml" />
	</mappers>
 </configuration>
```

43. DB에 Basket 테이블을 생성한다

```
CREATE TABLE "HUMAN"."BASKET"
 (	"MEMBERID" VARCHAR2(20 BYTE),
	"PRODUCTNAME" VARCHAR2(20 BYTE),
	"QUANTITY" NUMBER(*,0)
 )
```

44. 여기까지 확인



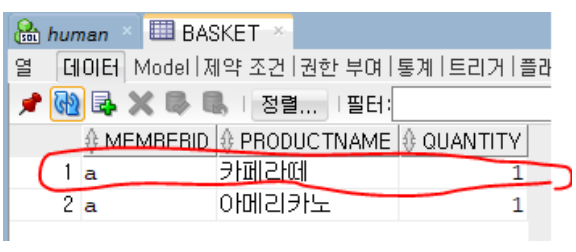
45. '카페라떼' 처리를 위해 index.html에 다음을 추가한다

```
<!-- Product name-->
                                <h5 class="fw-bolder"  id="productName2">카페라떼</h5>
                                <!-- Product reviews-->
                                <div class="d-flex justify-content-center small text-warning
mb-2">
                                    <div class="bi-star-fill"></div>
                                    <div class="bi-star-fill"></div>
                                    <div class="bi-star-fill"></div>
                                    <div class="bi-star-fill"></div>
                                    <div class="bi-star-fill"></div>
                                </div>
                                <!-- Product price-->
                                <span class="text-muted
text-decoration-line-through">4000</span>
                                3500
                            </div>
                        </div>
                        <!-- Product actions-->
                        <div class="card-footer p-4 pt-0 border-top-0 bg-transparent">
                            <div class="text-center"><div class="btn btn-outline-dark
mt-auto" id="basketDiv2">장바구니 넣기</div></div>
                        </div>
```

46. my.js에 basketDiv2 처리를 추가한다(h5태그 안에 있는 컨텐트를 얻으려면 text()를 써야한다)

```
$("#basketDiv2").click(function(){
                basketInsert($("#productName2").text());
        });
```

47. '카페라떼'항목이 잘 추가되는지 확인



48. 장바구니 항목 중복 처리를 해보자. BasketService.java를 다음과 같이 수정한다

```
	public void basketInsert(Basket b) throws Exception{
		Integer quantity=basketDAO.basketSelect(b);
		System.out.println(quantity);
		if(quantity==null) {
			b.setQuantity(1);
			basketDAO.basketInsert(b);
		}else {
			b.setQuantity(quantity.intValue()+1);
			basketDAO.basketUpdate(b);
		}
	}
```

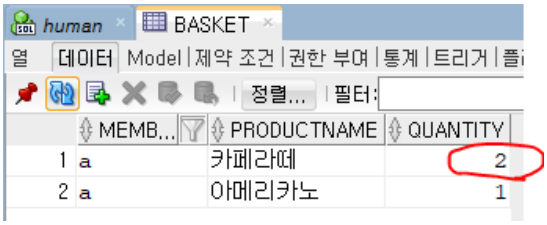
49. BasketDAO도 수정한다

```
public interface BasketDAO {

	public void basketInsert(Basket b) throws DataAccessException;

	public Integer basketSelect(Basket b) throws DataAccessException;

	public void basketUpdate(Basket b)  throws DataAccessException;
```

50. basket.xml에 insert quantity를 수정하고, select와 update를 추가한다

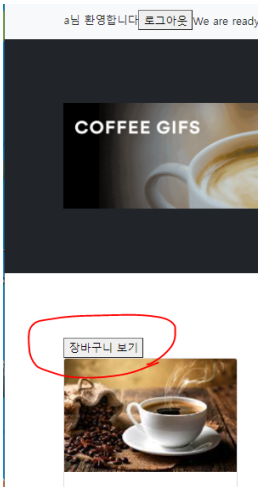
```
	<insert id="basketInsert" parameterType="basket">
		insert into Basket(memberId, productName, quantity)
		values( #{memberId}, #{productName}, #{quantity} )
	</insert>

	<select id="basketSelect" parameterType="basket"  resultType="Integer">
		select quantity from Basket where memberId=#{memberId} and
productName=#{productName}
	</select>

	<update id="basketUpdate"  parameterType="basket" >
		update Basket set quantity=#{quantity}
		where memberId=#{memberId} and productName=#{productName}
	</update>
```

51. 중복항목에 대해서 수량이 증가되는지 확인



52. index.html에서 적당한 곳에 장바구니 보기 버튼을 놓는다

```
<button id="basketViewBtn">장바구니 보기</button>
```



53. 장바구니 보기가 처리되도록 my.js에 다음을 추가한다. basketView 요청이 서버로 Get 방식으로 전송되고 처리 결과가 새창에 보이게 된다

```
$("#basketViewBtn").click(function(){
		window.open("basketView", "_blank",
"toolbar=yes,scrollbars=yes,resizable=yes,top=500,left=500,width=400,height=400
");
	});
```

54. BasketController에 다음 추가 (리턴 메세지에 주목하라)

```
@GetMapping("/basketView")
	public String basketView(HttpSession session) {
		Member m=(Member) session.getAttribute("m");
		if(m==null) {
			return "로그인 해주세요";
		}else {
			Basket b=new Basket();
			b.setMemberId(m.getId());
			List<Basket> list=basketService.basketView(b);
			session.setAttribute("basketList",list);

			System.out.println(list.size());
```

```
                String resultMsg="<html><head> <script
src=\"https://code.jquery.com/jquery-3.6.0.min.js\"></script><script
src='js/my.js'></script></head><body><h1>장바구니 목록</h1><br><table
border=1><tr><th>품명</th><th>수량</th></tr>";
                for(Basket basket:list) {
                        resultMsg
+="<tr><td>"+basket.getProductName()+"</td><td>"+basket.getQuantity()+"</td><
/tr>";
                }
                resultMsg+="</table><br><br><button
id='ordersBtn'>주문하기</button><button
onclick='window.close()'>닫기</button></body></html>";
                return resultMsg ;
        }
    }
```

55. BasketService에 다음 추가

```
public List<Basket> basketView(Basket b) {
        return basketDAO.basketView(b);
    }
```

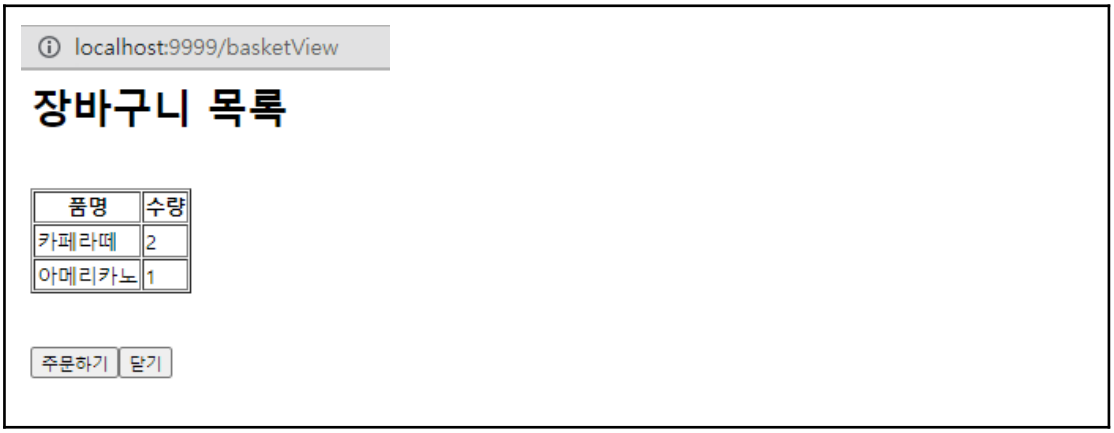
56. BasketDAO에 다음 추가

```
public List<Basket> basketView(Basket b)  throws DataAccessException;
```

57. basket.xml에 다음 추가

```
    <select id="basketView" parameterType="basket" resultType="basket">
        select * from Basket where memberId=#{memberId}
    </select>
```

58. '장바구니 보기' 확인



59. 주문하기 버튼을 누르면 처리되도록 my.js에 다음 추가 (주문하기 버튼이 동적 태그이므로 on메소드를 사용해서 이벤트를 처리해야 한다!!!)

```
$(document).on('click', '#ordersBtn', function(){
		$.post('orders',{}, function(data){
			alert(data);
			window.close();
		});
	});
```

60. BasketController에 다음 추가

```
@PostMapping("/orders")
	public String orders(HttpSession session) {
		Member m=(Member)session.getAttribute("m");
		if(m==null) {
			return "로그인 해주세요";
		}else {
			List<Basket>
list=(List<Basket>)session.getAttribute("basketList");
			if(list==null) {
				return "장바구니 내역이 없습니다";
			}else {
				try {
					basketService.orders(list,m.getId());
					return "주문 완료";
				} catch (Exception e) {
					e.printStackTrace();
					return "주문 처리 오류";
				}
			}
		}
	}
```

61. BasketService에 다음 추가 (장바구니에 있는 내역을 주문 테이블로 옮기고 장바구니는 비워야하는데 이것이 한번에 처리되어야 할 일이라 트랜잭션 처리를 해야함)

```
package com.example.web.service;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.dao.DataAccessException;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import com.example.web.dao.BasketDAO;
```

```
import com.example.web.dao.OrdersDAO;
import com.example.web.vo.Basket;
import com.example.web.vo.Orders;


@Service
public class BasketService {

	@Autowired
	BasketDAO basketDAO;

	@Autowired
	OrdersDAO ordersDAO;

	public void basketInsert(Basket b) throws Exception{
		Integer quantity=basketDAO.basketSelect(b);
		System.out.println(quantity);
		if(quantity==null) {
			b.setQuantity(1);
			basketDAO.basketInsert(b);
		}else {
			b.setQuantity(quantity.intValue()+1);
			basketDAO.basketUpdate(b);
		}

	}

	public List<Basket> basketView(Basket b) {
		return basketDAO.basketView(b);
	}

	@Transactional
	public void orders(List<Basket> list, String id)  throws Exception{
		// 장바구니에 있는 내역을 orders 테이블에 insert
		Orders o=new Orders();
		o.setMemberId(id);
		String contents="";
		for(Basket basket:list) {
			contents +=
basket.getProductName()+":"+basket.getQuantity()+"\t";
		}
		o.setContents(contents);
		Integer count=ordersDAO.ordersCount();
		o.setOrderNo( count.intValue()+1 );
		ordersDAO.ordersInsert(o);
		// basket 테이블에 있는 내역을 delete
		basketDAO.basketDelete(id);
	}

}
```

62. OrdersDAO를 다음과 같이 작성

```
package com.example.web.dao;

import org.apache.ibatis.annotations.Mapper;
import org.springframework.dao.DataAccessException;
import org.springframework.stereotype.Repository;
import com.example.web.vo.Orders;

@Mapper
@Repository
public interface OrdersDAO {
	public void ordersInsert(Orders o) throws DataAccessException;
	public Integer ordersCount() throws DataAccessException;
}
```

63. orders.xml을 다음과 같이 작성

```
<?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="com.example.web.dao.OrdersDAO">

	<insert id="ordersInsert" parameterType="orders">
		insert into Orders(memberId, contents, orderNo)
		values ( #{memberId},#{contents},#{orderNo}  )
	</insert>

	<select id="ordersCount"  resultType="Integer">
		select count(*) from Orders
	</select>

</mapper>
```

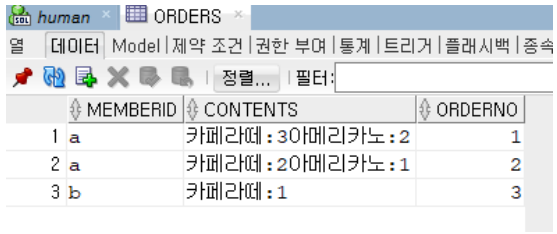
64. mybatis-config.xml에 orders.xml 등록

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration
  PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-config.dtd">

  <configuration>
	<mappers>
		<mapper resource="mybatis/mappers/member.xml" />
		<mapper resource="mybatis/mappers/basket.xml" />
		<mapper resource="mybatis/mappers/orders.xml" />
	</mappers>

  </configuration>
```

65. DB에 Orders 테이블을 생성한다

```
CREATE TABLE "HUMAN"."ORDERS"
 (      "MEMBERID" VARCHAR2(20 BYTE),
        "CONTENTS" VARCHAR2(500 BYTE),
        "ORDERNO" NUMBER(*,0)
 )
```

66. 주문 처리 결과를 확인한다



10. 리액트로 영화 사이트 만들기

1. cmd에서 npx create-react-app movie_app_2022 라고 명령한다
2. VS Code를 통해 이 폴더를 연다
3. package.json을 다음과 같이 수정한다

```
{
  "name": "movie_app_2022",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "@testing-library/jest-dom": "^5.14.1",
    "@testing-library/react": "^11.2.7",
    "@testing-library/user-event": "^12.8.3",
    "axios": "^0.21.4",
    "react": "^17.0.2",
    "react-dom": "^17.0.2",
    "react-router-dom": "^5.3.0",
    "react-scripts": "4.0.3",
    "web-vitals": "^1.1.2"
  },
  "scripts": {
    "start": "react-scripts start",
```

```
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },



  "eslintConfig": {
    "extends": [
      "react-app",
      "react-app/jest"
    ]
  },
  "browserslist": {
    "production": [
      ">0.2%",
      "not dead",
      "not op_mini all"
    ],
    "development": [
      "last 1 chrome version",
      "last 1 firefox version",
      "last 1 safari version"
    ]
  }
}
```

4. npm i 를 명령하여 모듈을 설치한다
5. index.html을 다음과 같이 수정한다

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <title>React App</title>
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootst
rap.min.css"
integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJl
SAwiGgFAW/dAiS6JXm" crossorigin="anonymous">
```

```
    <script
src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMV
NA/GpGFF93hXpG5KkN" crossorigin="anonymous"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd
/popper.min.js"
integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7
fakFPskvXusvfa0b4Q" crossorigin="anonymous"></script>
<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstra
p.min.js"
integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1Mqu
VdAyjUar5+76PVCmYl" crossorigin="anonymous"></script>

  </head>
  <body>
    <div id="root"></div>
  </body>
</html>
```

6. index.js를 다음과 같이 수정한다

```
import React from 'react';
import ReactDOM from 'react-dom';
import RouterApp from './RouterApp';


ReactDOM.render( <RouterApp />, document.getElementById('root')
);
```

7. RouterApp.js를 다음과 같이 작성한다

```
import React from "react";
import {HashRouter, Route} from 'react-router-dom';
import About from './routes/About';
import Home from "./routes/Home";
import Navigation from "./components/Navigation";
import Detail from "./routes/Detail";
```

```
import Board from "./components/Board";
import BoardDetail from "./components/BoardDetail";
import BoardWrite from "./components/BoardWrite";

class RouterApp extends React.Component{
    render(){
        return (
            <HashRouter>
                <Navigation />
                <Route path="/" exact={true}  component={Home}
/>
                <Route path="/about" component={About} />
                <Route path="/movie-detail" component={Detail}
/>
                <Route path="/board" component={Board} />
                <Route path="/board-detail"
component={BoardDetail} />
                <Route path="/board-write"
component={BoardWrite} />
            </HashRouter>
        );
    }
}

export default RouterApp;
```

8. Navigation.js를 다음과 같이 작성

```
import axios from 'axios';
import React, {useState} from 'react';
import {Link} from 'react-router-dom';
import './Navigation.css';




async function logout(){
   const returnData=await
axios.post('http://localhost:9999/logout');
   sessionStorage.removeItem("logined_name");
   sessionStorage.removeItem("JSESSIONID");
```

```
    window.location.reload();
}

function Navigation(){
    let [name, setName] = useState('');
    const logined_name=sessionStorage.getItem("logined_name");

    return (
        <div className="nav">
            {
            ( name ? name : name=logined_name) ?
                    (
                        <div> {name}님 환영합니다 <button
onClick={logout} >logout</button>
                        </div>

                    )
                    :
                    (
                        <div id="loginDiv">
                            ID <input size="5" id="id" />
                            PW <input size="5" type="password"
id="pw" />
                            <button onClick={  async ()=>{
                                    const
id_ele=document.getElementById("id");
                                    const
pw_ele=document.getElementById("pw");
                                    //console.log(id_ele.value,
pw_ele.value);
                                    const axios2 =
axios.create({
                                        withCredentials: true
                                    });
                                    const returnData=await
axios2.post('http://localhost:9999/login',null,
                                    { params : {id:id_ele.value,
pw: pw_ele.value} },
                                     );
                                    console.log(returnData);
                                    const
name=returnData.data.name;
```

```
                                        setName(name);

sessionStorage.setItem("logined_name",name);

                                }   }>login</button>
                            </div>
                        )
                }
                <Link to="/">Home</Link>
                <Link to="/board">게시판</Link>
                <Link to="/about" >About</Link>
            </div>
        );
}

export default Navigation;
```

9. Navigation.css를 다음과 같이 작성

```
.nav {
    z-index: 1;
    position: fixed;
    top: 50px;
    left: 10px;
    display: flex;
    flex-direction: column;
    background-color: white;
    padding: 10px 20px;
    box-shadow: 0 13px 27px -5px rgba(50, 50, 93, 0.25), 0 8px
16px -8px rgba(0, 0, 0, 0.3),
      0 -6px 16px -6px rgba(0, 0, 0, 0.025);
    border-radius: 5px;
  }

  @media screen and (max-width: 1090px) {
    .nav {
      left: initial;
      top: initial;
      bottom: 0px;
      width: 100%;
    }
```

```
  }

  .nav a {
    text-decoration: none;
    color: #0008fc;
    text-transform: uppercase;
    font-size: 12px;
    text-align: center;
    font-weight: 600;
  }

  .nav a:not(:last-child) {
    margin-bottom: 20px;
  }
```

10. Home.js를 다음과 같이 작성

```
import axios from "axios";
import React from "react";
import Movie from "../components/Movie";
import './Home.css';
import '../components/Movie.css';


class Home extends React.Component{
    state={
        isLoading: true,
        movies:[],
    }

    getMovies= async ()=>{
        const resultData= await
axios.get('https://yts-proxy.now.sh/list_movies.json?sort_by=rat
ing');
        //console.log(resultData.data.data.movies);
        this.setState({movies:resultData.data.data.movies,
isLoading:false});
    };

    componentDidMount(){
        this.getMovies();
```

```
    }

    render(){
        return (
            <div className="container">
                {this.state.isLoading ? 'Loading....' : (<div
className='movies'>{
                        this.state.movies.map( (item,index)=>{
return <Movie key={index} id={item.id} title={item.title}
year={item.year} summary={item.summary}
medium_cover_image={item.medium_cover_image}
genres={item.genres} /> } )
                    }</div>)}
            </div>
        );
    }
}

export default Home;
```

11. Home.css를 다음과 같이 작성

```
.container {
  height: 100%;
  display: flex;
  justify-content: center;
}

.loader {
  width: 100%;
  height: 100vh;
  display: flex;
  justify-content: center;
  align-items: center;
  font-weight: 300;
}
.movies {
  display: grid;
  grid-template-columns: repeat(2, minmax(400px, 1fr));
  grid-gap: 100px;
  padding: 50px;
```

```
  width: 80%;
  padding-top: 70px;
}

@media screen and (max-width: 1090px) {
  .movies {
    grid-template-columns: 1fr;
    width: 100%;
  }
}
```

12. Movie.js를 다음과 같이 작성

```
import React from 'react';
import './Movie.css';
import {Link} from 'react-router-dom';

function Movie( {id, title,year,summary, medium_cover_image,
genres} ){
    //console.log(genres);
    return (
        <div className="movie">
            <Link to={ { pathname:'/movie-detail' , state: {id,
year,title,summary,medium_cover_image,genres} } } >
                <img src={medium_cover_image} alt={title}
title={title}/>
                <div className="movie__data">
                    <h3 className="movie__title"  >{title}</h3>
                    <h5 className="movie__year">{year}</h5>
                    <ul className="movie__genres">
                    {genres.map((item,index)=>{ return <li
key={index}
                    className="movie__genre"> {item} </li>})}
</ul>
                    <p className="movie__summary">{summary}</p>
                </div>
            </Link>
        </div>

    );
}
```

```
export default Movie;
```

13. Movie.css를 다음과 같이 작성

```
.movies .movie {
    background-color: white;
    margin-bottom: 70px;
    font-weight: 300;
    padding: 20px;
    border-radius: 5px;
    color: #adaeb9;
    box-shadow: 0 13px 27px -5px rgba(50, 50, 93, 0.25), 0 8px
16px -8px rgba(0, 0, 0, 0.3),
      0 -6px 16px -6px rgba(0, 0, 0, 0.025);
  }

  .movies .movie a {
    display: grid;
    grid-template-columns: minmax(150px, 1fr) 2fr;
    grid-gap: 20px;
    text-decoration: none;
    color: inherit;
  }

  .movie img {
    position: relative;
    top: -50px;
    max-width: 150px;
    width: 100%;
    margin-right: 30px;
    box-shadow: 0 30px 60px -12px rgba(50, 50, 93, 0.25), 0 18px
36px -18px rgba(0, 0, 0, 0.3),
      0 -12px 36px -8px rgba(0, 0, 0, 0.025);
  }

  .movie .movie__title,
  .movie .movie__year {
    margin: 0;
    font-weight: 300;
  }
```

```
.movie .movie__title {
  margin-bottom: 5px;
  font-size: 24px;
  color: #2c2c2c;
}

.movie .movie__genres {
  list-style: none;
  padding: 0;
  margin: 0;
  display: flex;
  flex-wrap: wrap;
  margin: 5px 0px;
}

.movie__genres li,
.movie .movie__year {
  margin-right: 10px;
  font-size: 14px;
}

.movie__genre {
    background-color: lightgrey;
    color: black;
    border: lightgrey;
    border-radius: 20px;
}
```

14. Detail.js를 다음과 같이 작성

```
import axios from "axios";
import React from "react";
import './Detail.css';


class Detail extends React.Component{

  state={
      rating:0,
      runtime:0,
```

```
    };

    getMovieDetail= async (id)=>{
        const movieDetail=await
axios.get('https://yts-proxy.now.sh/movie_details.json?movie_id=
'+id);
        this.setState(
{rating:movieDetail.data.data.movie.rating, runtime:
movieDetail.data.data.movie.runtime} );
    };

    componentDidMount(){
        const {location, history}=this.props;
        if(location.state===undefined){
            history.push('/');
        }else{
            //console.log(location.state.id);
            this.getMovieDetail(location.state.id);
        }
    }

    render(){
        //console.log(this.props);
        const {location} =this.props;
        if(location.state){
            return(
                <div className="detail__container">
                    <img src={location.state.medium_cover_image}
alt={location.state.title} title={location.state.title}/>
                    <h4 className="movie__tit_detail"> {
location.state.title }  </h4>
                    <b className="movie__year_detail">Year :
{location.state.year}      |      rating :
{this.state.rating} point      |     
runtime : {this.state.runtime} min.</b>
                    <br/>
                    <ul className="movie__gen_detail">Genres :
  {location.state.genres.map((item,index) => { return <li
key={index}

className="movie__gen_detail">{item}</li>})}</ul>
                    <p
```

```
className="movie__sum_detail">{location.state.summary}</p>

                </div>
            );
        }else{
            return null;
        }

    }

}




export default Detail;
```

15. Detail.css를 다음과 같이 작성

```
.detail__container{
    box-shadow: 0 13px 27px -5px rgba(50, 50, 93, 0.25), 0 8px
16px -8px rgba(0, 0, 0, 0.3),
      0 -6px 16px -6px rgba(0, 0, 0, 0.025);
    border-radius: 5px;
    padding: 20px;
    background-color: white;
    margin: 0 auto;
    margin-top: 100px;
    width: 100%;
    max-width: 800px;
    font-weight: 300;

    text-align: center;
  }

  .movie__tit_detail{
      font-family: "Arial Black", sans-serif;
      font-size: 35px;
      font-weight: bold;
      background-color: #666666;
```

```
    color: #222222;
    text-shadow: 4px 2px 3px rgba(255,255,255,0.2);
    height: 45px;

}

.movie__gen_detail{
    margin-right: 20px;
    display: inline;
    font-size: 20px;


}

.movie__sum_detail {
    margin-top: 20px;
    font-size: 15px;
    color: gray;
}
```

16. About.js를 다음과 같이 작성

```
import React from "react";
import './About.css';

function About( props ){
    console.log(props);
    return (
    <div className="about__container">
        <span>사람은 삶의 목표가 행복이라고 여기지 않을 때에만
행복해진다. Men can only be happy when they do not assume that
the object of life is happiness.</span>
        <span> - George Orwell, 1984</span>
    </div>);
}

export default About;
```

17. About.css를 다음과 같이 작성

```
.about__container {
    box-shadow: 0 13px 27px -5px rgba(50, 50, 93, 0.25), 0 8px
16px -8px rgba(0, 0, 0, 0.3),
      0 -6px 16px -6px rgba(0, 0, 0, 0.025);
    padding: 20px;
    border-radius: 5px;
    background-color: white;
    margin: 0 auto;
    margin-top: 100px;
    width: 100%;
    max-width: 400px;
    font-weight: 300;
  }

  .about__container span:first-child {
    font-size: 20px;
  }
  .about__container span:last-child {
    display: block;
    margin-top: 10px;
  }
```

18. Board.js를 다음과 같이 작성

```
import axios from "axios";
import React from "react";
import './Board.css';
import {Link} from 'react-router-dom';


class Board extends React.Component{
    state={
        boardList:[],
    };

    getBoard=async ()=>{
        const boardList=await
axios.get('http://localhost:9999/boardList2');
        //console.log(boardList.data);
```

```
        this.setState({boardList:boardList.data});

    }

    componentDidMount(){
        this.getBoard();
    }


    render(  ){
        return (
            <div className="about__container">
                <table className='table table-striped'>
                    <thead className='thead-dark'>

<tr><th>글번호</th><th>글제목</th><th>글쓴이</th><th>작성일</th></tr
>
                    </thead>
                    <tbody>
                        {

this.state.boardList.map((item,index)=>{
                                const icon='↪️';
                                let iconResult='';
                                for(let i=1;i<item.level;i++){
                                    iconResult += icon;
                                }

                                return (
                                    <tr key={index}>

<td>{item.articleNo}</td>
                                        <td>
                                            <Link to={ {
pathname:'/board-detail' , state: {...item} } } >
                                                <font size="2"
title={item.content}>{iconResult}{item.title}</font>
                                            </Link>
                                        </td>
                                        <td>{item.id}</td>

<td>{item.writeDate}</td>
```

```
                                    </tr>
                                )


                            })
                        }

                    </tbody>
                </table>
                <div style={{textAlign: 'center' }}>
                    <Link to={ { pathname:'/board-write' ,
state:{articleNo:0}  } } >
                        <button className="btn
btn-info">새글쓰기</button>
                    </Link >  
                    <Link to={ { pathname:'/'  } } >
                        <button className="btn
btn-warning">홈으로</button>
                    </Link>
                </div>
            </div>
        );
    }
}

export default Board;
```

19. Board.css를 다음과 같이 작성

```
.about__container {
    box-shadow: 0 13px 27px -5px rgba(50, 50, 93, 0.25), 0 8px
16px -8px rgba(0, 0, 0, 0.3),
      0 -6px 16px -6px rgba(0, 0, 0, 0.025);
    padding: 20px;
    border-radius: 5px;
    background-color: white;
    margin: 0 auto;
    margin-top: 100px;
    width: 100%;
    max-width: 800px;
    font-weight: 300;
  }
```

```
  .about__container span:first-child {
    font-size: 20px;
  }
  .about__container span:last-child {
    display: block;
    margin-top: 10px;
  }
```

20. BoardDetail.js를 다음과 같이 작성

```
import React from "react";
import './Board.css';
import {Link} from 'react-router-dom';

class BoardDetail extends React.Component{
    componentDidMount(){
        const {location, history}=this.props;
        if(location.state===undefined){
            history.push('/board');
        }
    }

    render(){
        const {location} =this.props;
        console.log(location.state);
        if(location.state){
            return (

                <div className="about__container">

                    <div className="row">
                        <table className="table">
                            <thead>
                                <tr className="table-active">
                                    <th scope="col"
style={{width: '50%' }} > 글제목 : {location.state.title} <br/>
                                        글쓴이 :
{location.state.id}</th>
                                    <th scope="col"
style={{width: '50%' }} className="text-right"> 작성일 :
```

```
{location.state.writeDate} </th>
                                    </tr>
                                </thead>
                                <tbody>
                                    <tr><td
colSpan="2"><pre>{location.state.content}</pre></td></tr>
                                    <tr style={{textAlign: 'center'
}}>
                                        <td colSpan="2">
                                            <Link to={ {
pathname:'/board-write' , state:{
articleNo:location.state.articleNo} }  } >
                                                <button
className="btn btn-info">답글쓰기</button>
                                            </Link>
                                              

                                            <Link to={ {
pathname:'/board' , state: {} } } >
                                                <button
className="btn btn-warning">돌아가기</button>
                                            </Link>
                                        </td>
                                    </tr>
                                </tbody>
                            </table>
                        </div>
                    </div>
                );
            }else{
                return null;
            }

        }
}

export default BoardDetail;
```

21. BoardWrite.js를 다음과 같이 작성

```
import React from "react";
```

```
import './Board.css';
import {Link} from 'react-router-dom';
import axios from "axios";


class BoardWrite extends React.Component{

    boardWriteHandler= async ()=>{
        const title_ele=document.getElementById("title");
        const content_ele=document.getElementById("content");

        console.log(this.props.location.state.articleNo);
        const axios2 = axios.create({
            withCredentials: true
          });
        const returnData=await
axios2.post('http://localhost:9999/boardWrite',
        null,
        {params:{title:title_ele.value,
content:content_ele.value,
articleNO:this.props.location.state.articleNo}},
        );
        alert(returnData.data);
    };


    render(){
       //console.log(this.props);

        return (

            <div className="about__container">

                <div className="row">
                    <table className="table" style={{textAlign:
'center' }} >
                        <thead>
                            <tr className="table-active">
                                <th scope="col" style={{width:
'50%' }} > {this.props.location.state.articleNo>0 ?   '[ ' +
this.props.location.state.articleNo +' ]의 답글 :  ' : ''} 글제목
: <input  id='title' /> </th>
```

```
                            </tr>
                        </thead>
                        <tbody>
                            <tr><td colSpan="2"><textarea
rows="10" cols="50" id="content"></textarea></td></tr>
                            <tr >
                                <td colSpan="2">
                                    <button  className="btn
btn-info" onClick={ this.boardWriteHandler } >등록</button>
                                         
                                        <button  className="btn
btn-info">취소</button>
                                         

                                    <Link to={ {
pathname:'/board' , state: {} } } >
                                        <button  className="btn
btn-warning">글목록</button>
                                    </Link>
                                </td>
                            </tr>
                        </tbody>
                    </table>
                </div>
            </div>
        );

    }
}

export default BoardWrite;
```

22. 서버쪽 MemberController에서 다음을 추가

```
@RestController
@CrossOrigin(origins = "http://localhost:3000", allowCredentials = "true" )
public class MemberController {
```

23. BoardController를 다음과 같이 작성 (이전에 @Controller로 선언한 것을 바꾸지 않고 @ResponseBody를 추가하여 진행하였음)

```
package com.example.web.controller;

import java.util.List;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import org.json.simple.JSONArray;
import org.json.simple.JSONObject;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.ResponseBody;

import com.example.web.service.BoardService;
import com.example.web.vo.Board;
import com.example.web.vo.Member;

@Controller
@CrossOrigin(origins = "http://localhost:3000", allowCredentials = "true")
public class BoardController {

	@Autowired
	BoardService boardService;

	@PostMapping("/boardWrite")
	@ResponseBody
	public String boardWrite(@ModelAttribute Board b, HttpSession
session,HttpServletResponse response) {
		//System.out.println("새글쓰기 요청시 세션 ID:"+session.getId());

		Member m=(Member)session.getAttribute("m");
		if(m==null) {
			return "로그인부터 하세요";
		}else {
			b.setId(m.getId());
			if(b.getArticleNO()>0) {
				b.setParentNO(b.getArticleNO());
				b.setArticleNO(0);
			}
			System.out.println(b);
			try {
				boardService.boardWrite(b);
			} catch (Exception e) {
				e.printStackTrace();
				return "글등록 실패";
			}
```

```
                return "글이 등록되었습니다";
            }
        }

        @GetMapping("/boardList")
        public String boardList(Model model) {
            List<Board> list=boardService.boardList();
            model.addAttribute("articlesList", list);
            return "test";
        }

        @GetMapping("/boardList2")
        @ResponseBody
        public String boardList2() {
            List<Board> list=boardService.boardList();
            JSONArray arr=new JSONArray();
            for(Board b:list) {
                JSONObject o=new JSONObject();
                o.put("articleNo", b.getArticleNO());
                o.put("title", b.getTitle());
                o.put("content", b.getContent());
                o.put("id", b.getId());
                o.put("parentNo", b.getParentNO());
                o.put("writeDate", b.getWriteDate().toLocaleString());
                o.put("level", b.getLevel());
                arr.add(o);
            }
            System.out.println(arr);
            return arr.toJSONString();
        }
}
```

24. BoardService를 다음과 같이 작성

```
package com.example.web.service;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.example.web.dao.BoardDAO;
import com.example.web.vo.Board;

@Service
public class BoardService {

        @Autowired
```

```
	BoardDAO boardDAO;

	public List<Board> boardList() {
		return boardDAO.boardList();
	}

	public void boardWrite(Board b) throws Exception {
		boardDAO.boardWrite(b);

	}

}
```

25. BoardDAO를 다음과 같이 작성

```
package com.example.web.dao;

import java.util.List;

import org.apache.ibatis.annotations.Mapper;
import org.springframework.dao.DataAccessException;
import org.springframework.stereotype.Repository;

import com.example.web.vo.Board;

@Mapper
@Repository
public interface BoardDAO {

	List<Board> boardList() throws DataAccessException;

	void boardWrite(Board b) throws DataAccessException;

}
```

26. board.xml을 다음과 같이 작성

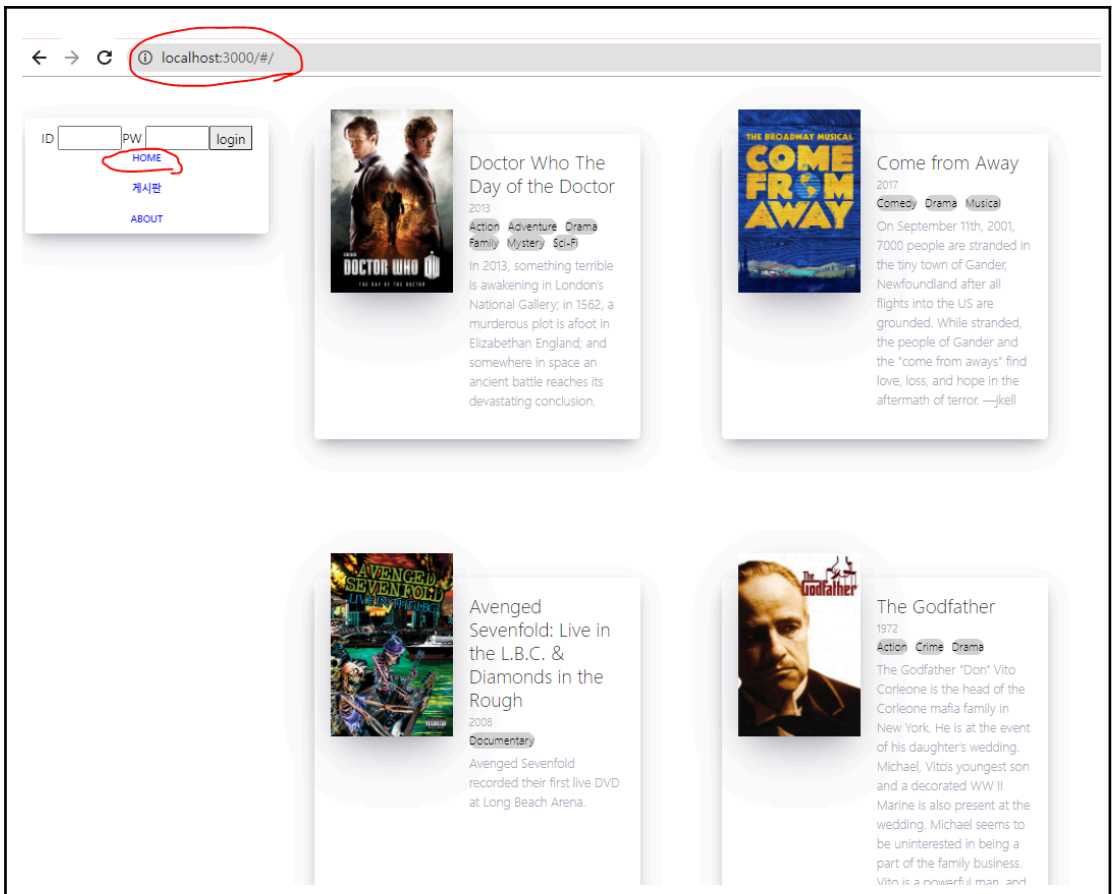
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper
	PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
	"http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="com.example.web.dao.BoardDAO">
	<insert id="boardWrite" parameterType="board">
		insert into board(title,content,id,writeDate,parentNO)
		values(#{title},#{content},#{id}, SYSDATE, #{parentNO})
	</insert>
```

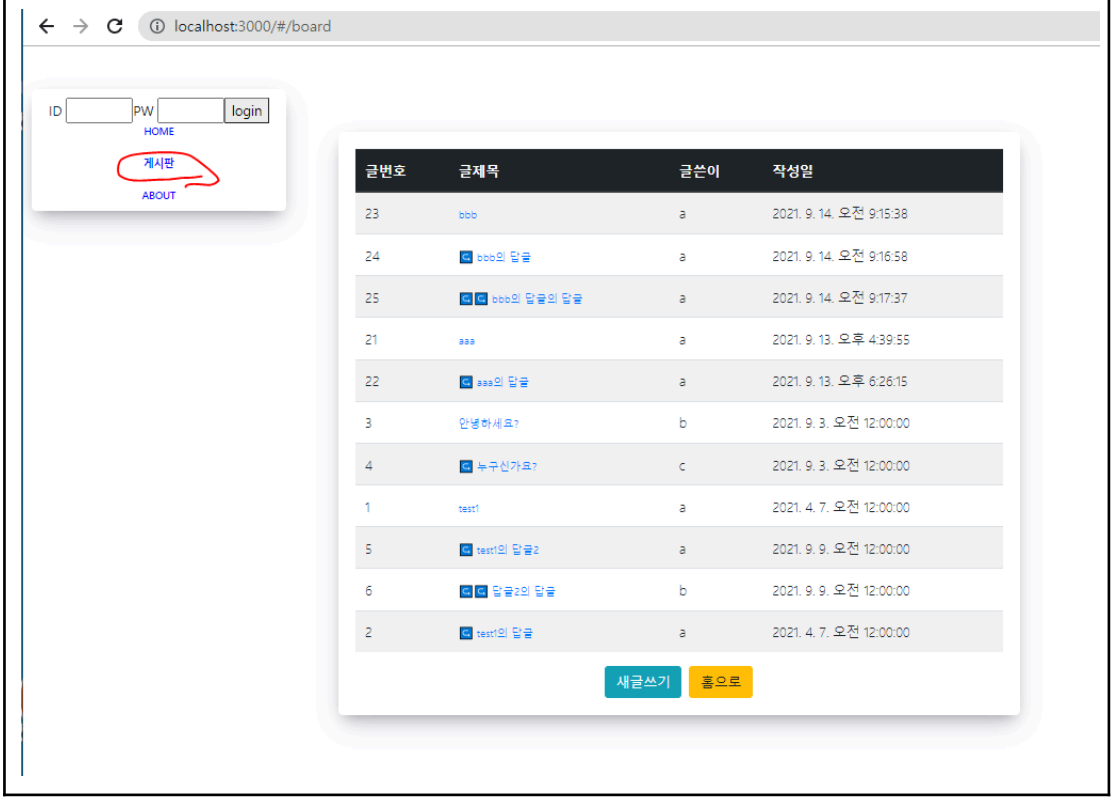
```
	<select id="boardList" resultType="board">
		<![CDATA[
			select level,
				articleNO,
				parentNO,
				LPAD(' ', 4*(LEVEL-1)) || title title,
				content,
				writeDate,
				imageFileName,
				id
			from board
			start with parentNO=0
			connect by prior articleNO=parentNO
			order siblings by articleNO desc
		]]>
	</select>

</mapper>
```

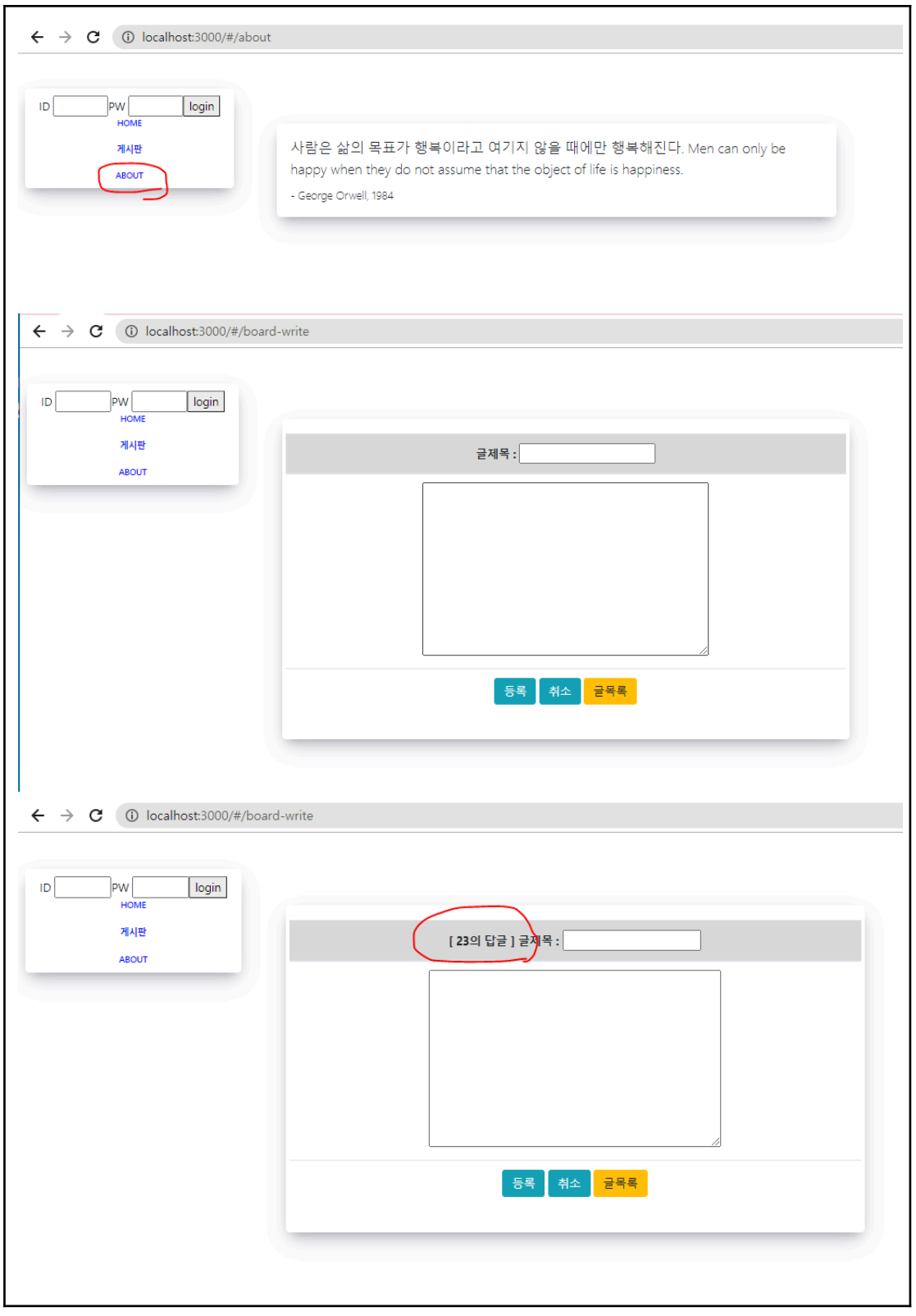
27. mybatis-config.xml에 다음을 추가

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration
  PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-config.dtd">

  <configuration>
	<mappers>
		<mapper resource="mybatis/mappers/member.xml" />
		<mapper resource="mybatis/mappers/basket.xml" />
		<mapper resource="mybatis/mappers/orders.xml" />
		<mapper resource="mybatis/mappers/board.xml" />
	</mappers>

  </configuration>
```

28. npm start하고 브라우저에서 확인한다


localhost:3000/#/
ID
PW
login
HOME
게시판
ABOUT
Doctor Who The Day of the Doctor
2013
Action Adventure Drama Family Mystery Sci-Fi
In 2013, something terrible is awakening in London's National Gallery; in 1562, a murderous plot is afoot in Elizabethan England; and somewhere in space an ancient battle reaches its devastating conclusion.
Come from Away
2017
Comedy Drama Musical
On September 11th, 2001, 7000 people are stranded in the tiny town of Gander, Newfoundland after all flights into the US are grounded. While stranded, the people of Gander and the "come from aways" find love, loss, and hope in the aftermath of terror. —jkell
Avenged Sevenfold: Live in the L.B.C. & Diamonds in the Rough
2008
Documentary
Avenged Sevenfold recorded their first live DVD at Long Beach Arena.
The Godfather
1972
Action Crime Drama
The Godfather "Don" Vito Corleone is the head of the Corleone mafia family in New York. He is at the event of his daughter's wedding. Michael, Vito's youngest son and a decorated WW II Marine is also present at the wedding. Michael seems to be uninterested in being a part of the family business.


localhost:3000/#/board
ID
PW
login
HOME
게시판
ABOUT
글번호 글제목 글쓴이 작성일
23 bbb a 2021. 9. 14. 오전 9:15:38
24 bbb의 답글 a 2021. 9. 14. 오전 9:16:58
25 bbb의 답글의 답글 a 2021. 9. 14. 오전 9:17:37
21 aaa a 2021. 9. 13. 오후 4:39:55
22 aaa의 답글 a 2021. 9. 13. 오후 6:26:15
3 안녕하세요? b 2021. 9. 3. 오전 12:00:00
4 누구신가요? c 2021. 9. 3. 오전 12:00:00
1 test1 a 2021. 4. 7. 오전 12:00:00
5 test1의 답글2 a 2021. 9. 9. 오전 12:00:00
6 답글2의 답글 b 2021. 9. 9. 오전 12:00:00
2 test1의 답글 a 2021. 4. 7. 오전 12:00:00
새글쓰기
홈으로



29. 글목록 paging 작업을 위해 board.xml에 다음 추가

```
<select id="boardListPerPage"  parameterType="int" resultType="board">
                <![CDATA[
SELECT * FROM
```

```
(
    SELECT T1.*,
    ROWNUM AS SEQ,
    FLOOR((ROWNUM-1)/6+1) AS PAGE,              -- 한 페이지에 6행씩 보여주는
경우
    COUNT(*) OVER() AS TTCNT FROM               -- 테이블에 저장된 총
라인(행) 수 -
    (
      select LEVEL, articleNO,parentNO, LPAD(' ', 4*(LEVEL-1)) || title
title,content,writeDate,imageFileName,id  from board
      START WITH parentNO=0
      CONNECT BY PRIOR articleNO=parentNO
      order siblings by articleNO desc
    ) T1
)
WHERE PAGE = #{pageNo}
                        ]]>
        </select>
```

30. BoardDAO에 다음 메소드 추가

```
        List<Board> boardListPerPage(int pageNo) throws DataAccessException;
```

31. BoardService에 다음 메소드 추가 (boardList()를 overloading했음)

```
@Service
public class BoardService {

        @Autowired
        BoardDAO boardDAO;

        public List<Board> boardList() {
                        return boardDAO.boardList();
        }

        public List<Board> boardList(int pageNo) {
                        return boardDAO.boardListPerPage(pageNo);
        }

        public void boardWrite(Board b) throws Exception {
                boardDAO.boardWrite(b);

        }
```

32. BoardController에서 request parameter로 페이지번호를 얻는다

```
@GetMapping("/boardList2")
```

```
	@ResponseBody
	public String boardList2(HttpServletRequest request) {
		String page=request.getParameter("page");
		int pageNo=Integer.parseInt(page);
		//System.out.println(pageNo);

		List<Board> list=boardService.boardList(pageNo);
		JSONArray arr=new JSONArray();
		for(Board b:list) {
			JSONObject o=new JSONObject();
			o.put("articleNo", b.getArticleNO());
			o.put("title", b.getTitle());
			o.put("content", b.getContent());
			o.put("id", b.getId());
			o.put("parentNo", b.getParentNO());
			o.put("writeDate", b.getWriteDate().toLocaleString());
			o.put("level", b.getLevel());
			arr.add(o);
		}
		//System.out.println(arr);
		return arr.toJSONString();
	}
```

33. Board.js를 다음과 같이 수정 (서버로 페이지번호가 넘어 가도록 처리)

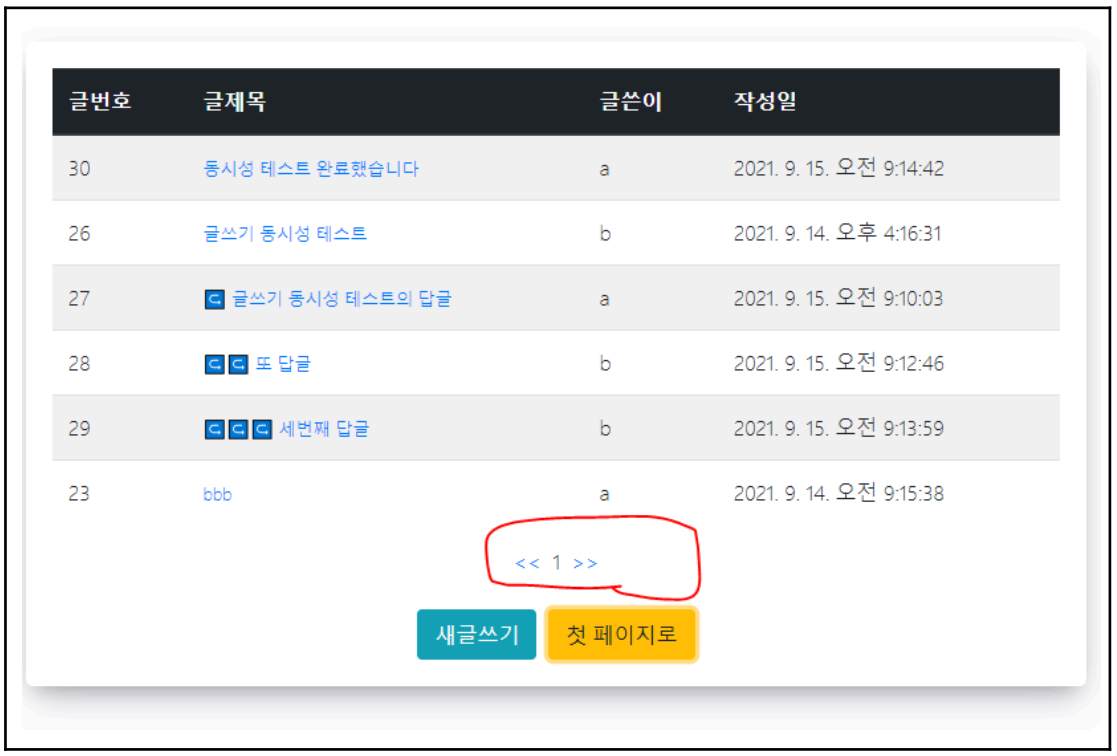
```
import axios from "axios";
import React from "react";
import './Board.css';
import {Link} from 'react-router-dom';


class Board extends React.Component{
    state={
        page:1,
        boardList:[],
    };

    getBoard=async ()=>{
        const boardList=await
axios.get('http://localhost:9999/boardList2?page='+this.state.pa
ge);
        //console.log(boardList.data);
        this.setState({boardList:boardList.data});

    }
```

```
    componentDidMount(){
        this.getBoard();
    }

    nextPage= async ()=>{
        if(this.state.boardList.length===0){
            return alert("마지막 페이지입니다");
        }
        await this.setState({page: this.state.page+1});
        this.getBoard();
    };

    prevPage= async ()=>{
        if(this.state.page<=1){
            return alert('첫 페이지입니다');
        }
        await this.setState({page: this.state.page-1});
        this.getBoard();
    };

    firstPage= async ()=>{
        await this.setState({page: 1});
        this.getBoard();
    };


    render(  ){
        return (
            <div className="about__container">
                <table className='table table-striped'>
                    <thead className='thead-dark'>

<tr><th>글번호</th><th>글제목</th><th>글쓴이</th><th>작성일</th></tr
>
                    </thead>
                    <tbody>
                        {

this.state.boardList.map((item,index)=>{
                                const icon='↪️';
                                let iconResult='';
                                for(let i=1;i<item.level;i++){
```

```
                                    iconResult += icon;
                                }

                                return (
                                    <tr key={index}>

<td>{item.articleNo}</td>
                                        <td>
                                            <Link to={ {
pathname:'/board-detail' , state: {...item} } } >
                                                <font size="2"
title={item.content}>{iconResult}{item.title}</font>
                                            </Link>
                                        </td>
                                        <td>{item.id}</td>

<td>{item.writeDate}</td>
                                    </tr>
                                )

                            })
                        }

                    </tbody>
                </table>

                <div style={{textAlign: 'center' }}>
                    <Link  to='/board'  onClick={this.prevPage}
>{'<<'}</Link>  
                    {this.state.page}
                      <Link  to='/board'
onClick={this.nextPage} >{'>>'}</Link>
                    <br/><br/>
                    <Link to={ { pathname:'/board-write' ,
state:{articleNo:0}  } } >
                        <button className="btn
btn-info">새글쓰기</button>
                    </Link >  
                    <Link to='/board'  onClick={this.firstPage}
>
                        <button className="btn btn-warning">첫
페이지로</button>
```

```
                    </Link>
                </div>
            </div>
        );
    }
}

export default Board;
```

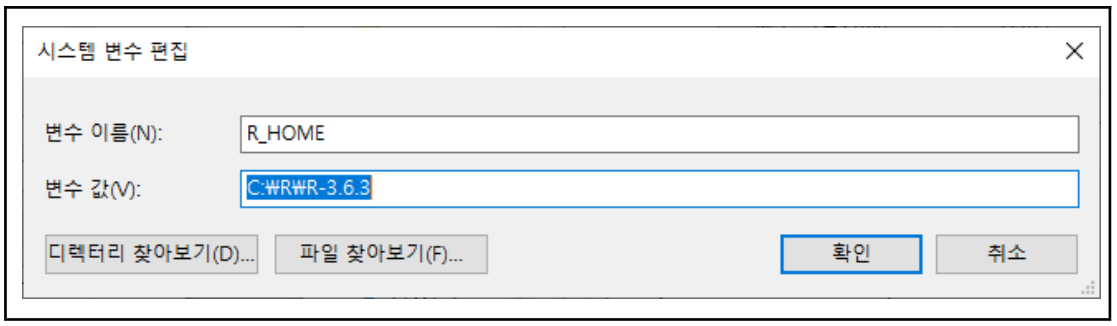
34. 페이징 확인



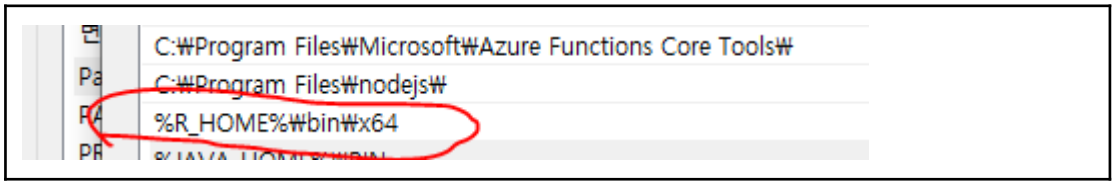
11. R 설치

1. https://cran.seoul.go.kr/ 에서 R 3.6.3을 내려 받아, 띄어쓰기가 없는 경로 (예 C:\R\R-3.6.3)에 설치한다.

2. R을 환경변수에 추가한다



3. Path에 다음과 같이 추가한다



4. https://drive.google.com/drive/u/1/folders/1JhRe6e0CsTD4XJ_1uYqBt5Q7LRdk7q7F 에서 cloudWord_library.zip을 다운 받아 C:\R\R-3.6.3\library에 풀어 놓는다.

5. c:\\temp\\wordCloud.R을 다음과 같이 작성한다 (마킹 된 부분을 필히 확인하라~!)

```
Sys.setenv(JAVA_HOME='C:\\Program Files\\Java\\jdk-11.0.12')
library(RColorBrewer)
palete<-brewer.pal(9,"Set1")
setwd("c:\\Temp")
# install.packages("wordcloud")
library(wordcloud)
data1<-readLines("data.txt",encoding='UTF-8')
# install.packages('hash')
# install.packages('tau')
# install.packages('Sejong')
# install.packages('RSQLite')
# install.packages('devtools')
# install.packages('rJava')
#
install.packages("https://cran.r-project.org/src/contrib/Archive/KoNLP/KoNLP_0.80.2.tar.gz",repos=NULL, type="source", INSTALL_opts=c('--no-lock'))
library(KoNLP)
data2<-sapply(data1,extractNoun,USE.NAMES = F)
data3<-unlist(data2)
data3<-Filter(function(x){nchar(x)>=2},data3)
data3<-gsub("\\d+","",data3)
data3<-gsub("\\(","",data3)
data3<-gsub("\\)","",data3)
data3<-gsub("\\;","",data3)
data3<-gsub("the","",data3)
data3<-gsub("and","",data3)
data3<-gsub("of","",data3)
data3<-gsub("to","",data3)
data3<-gsub("in","",data3)
```

```
data3<-gsub("or","",data3)
data3<-gsub("for","",data3)
data3<-gsub("by","",data3)
data3<-gsub("which","",data3)
write(unlist(data3),"data_cloud.txt")
data4<-read.table("data_cloud.txt")
wordcount<-table(data4)
wordcount
jpeg(filename = "cloud.jpg",width = 1000, height = 1000)
wordcloud(names(wordcount),freq = wordcount,scale=c(5,1),rot.per = 0.1,min.freq
= 1,random.order = F, color=T, colors=palete)
dev.off()
```

6. src/main/webapp 밑에 cloudImg라는 폴더를 만들어 놓는다

7. BoardController에 다음 추가

```
@GetMapping("/boardAnaly")
	@ResponseBody
	public String boardAnaly() {
		try {
			//System.out.println(1);
			boardService.boardAnaly();
			//System.out.println(6);
			return "http://localhost:9999/cloudImg/wordCloud.jpg";
		} catch (Exception e) {
			return "";
		}
	}
```

8. BoardService에 다음 추가

```
public void boardAnaly() throws Exception {
		Runtime rt=Runtime.getRuntime();
```

```
                Process pc=null;
                try {
                        pc=rt.exec("rscript c:\\temp\\wordCloud.R");
                        //System.out.println(2);
                } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }finally {
                        try {
                                pc.getErrorStream().close();
                                pc.getInputStream().close();
                                pc.getOutputStream().close();

                                pc.waitFor();
                                //System.out.println(3);
                        } catch (InterruptedException e) {
                        }
                        pc.destroy();
                        //System.out.println(4);
                }
                // 1. 원본 File, 복사할 File 준비
                File file = new File("c:\\temp\\cloud.jpg");
                File newFile = new
File("C:\\0jes\\workspace2\\2_SpringBootMVC\\src\\main\\webapp\\cloudImg\\word
Cloud.jpg");
                // 2. 복사
                Files.copy(file.toPath(), newFile.toPath(),
StandardCopyOption.REPLACE_EXISTING);
                //System.out.println(5);
        }
```

9. Navigation.js에 [영화평 분석 보기] 추가

```
<Link to="/">Home</Link>
              <Link to="/board">영화평 게시판</Link>
              <Link to="/board-analy">영화평 분석 보기</Link>
              <Link to="/about" >About</Link>
```

10. RouterApp.js데 다음 추가

```
import BoardAnaly from "./components/BoardAnaly";

…

<Route path="/board-analy" component={BoardAnaly} />
```

11. BoardAnaly.js를 다음과 같이 작성한다

```
import React from "react";
import './BoardAnaly.css';
import axios from "axios";

class BoardAnaly extends React.Component{
    state={
        isLoading: true,
        img:'',
    }

    getBoardAnaly= async ()=>{
        const resultData= await
axios.get('http://localhost:9999/boardAnaly');
        console.log(resultData.data);
        this.setState({img:resultData.data, isLoading:false});
    };

    componentDidMount(){
        this.getBoardAnaly();
    }

    render(){
        return (
        <div className="about__container" style={{textAlign:
'center' }}>
            <h3 >영화평 분석 보기</h3>
            {this.state.isLoading ? 'Loading....' : (<div> <img
src={this.state.img}  /></div>)}

        </div>);
    }
}

export default BoardAnaly;
```

12. BoardAnaly.css를 다음과 같이 작성한다

```
.about__container {
    box-shadow: 0 13px 27px -5px rgba(50, 50, 93, 0.25), 0 8px
```

```
16px -8px rgba(0, 0, 0, 0.3),
      0 -6px 16px -6px rgba(0, 0, 0, 0.025);
    padding: 20px;
    border-radius: 5px;
    background-color: white;
    margin: 0 auto;
    margin-top: 100px;
    width: 1000px;
    /*max-width: 400px;*/
    font-weight: 300;
  }

  .about__container span:first-child {
    font-size: 20px;
  }
  .about__container span:last-child {
    display: block;
    margin-top: 10px;
  }
```

13. 확인