**<인터넷 컴퓨팅 프로젝트 – 홈페이지 구현하기 by jsp>**

**컴퓨터SW 이재우**

**1) 홈페이지 : SonFanPage (축구 선수 손흥민의 팬 페이지)**

**\* 구현한 기능**

① 회원가입

② 로그인 / 로그아웃

③ 게시판

- 글 작성(등록)

- 글 삭제

- 글 수정

- 글 목록 보기

- 글 검색

④ 마이페이지 (계정 관리)

- 회원정보 변경

- 비밀번호 변경

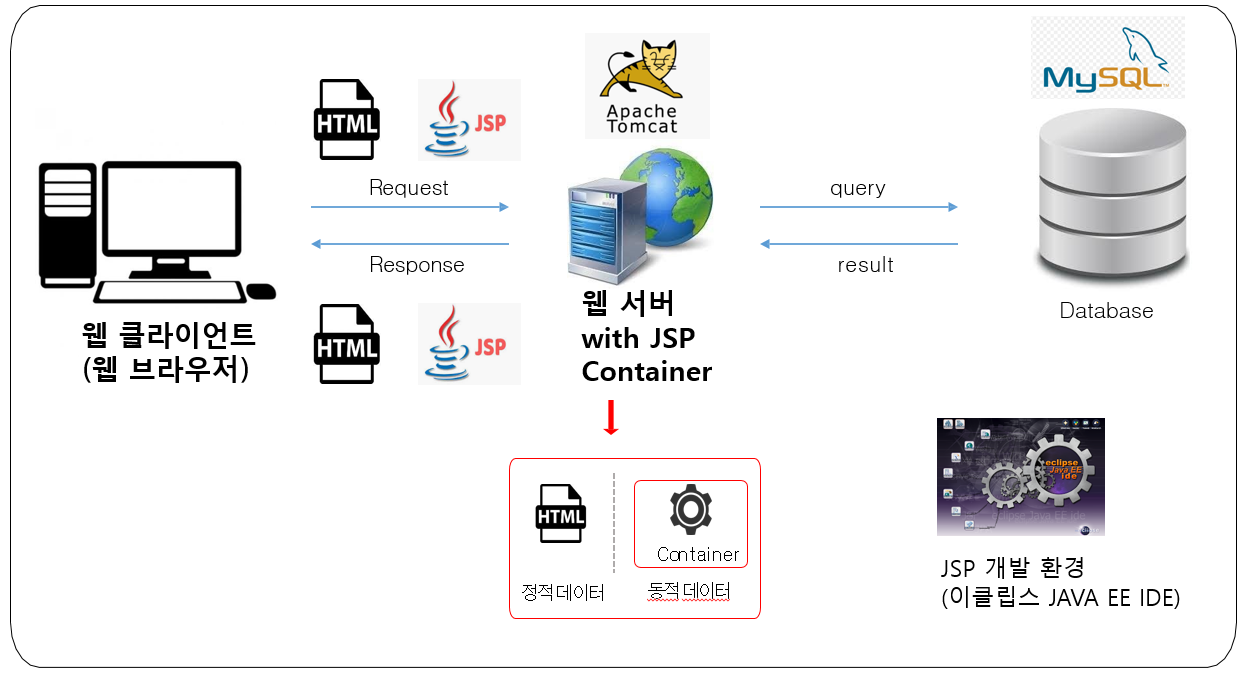
- 회원 탈퇴

**\* 외부 접속 Domain Name or ip:port**

**: [호스팅도메인입력]**

**2) 구축한 웹 서버 환경에 대한 그림 및 간단한 설명**

**\* 전체적인 구성 및 간단한 설명**



① 클라이언트가 어떤 동작을 함으로써 웹 서버에 필요한 정보를 요청한다.

② JSP 컨테이너가 JSP 파일을 읽는다.

③ JSP 컨테이너가 변환 작업을 통해 Servlet 파일(.java)을 생성한다.

④ .java 파일은 다시 .class 파일로 컴파일 된다.

⑤ 실행을 통해 HTML 파일을 생성하여 JSP 컨테이너에게 전달한다.

⑥ JSP는 HTTP 프로토콜을 통해 HTML 페이지를 클라이언트에게 전달한다.

⑦ 데이터베이스와 서버는 sql문(Query)을 통해 소통을 하는데, 이는 코드 내 DAO,

DTO파일과 관련이 있다.

# DAO (Data Access Object)

: DAO란 데이터 베이스에 접근해서 데이터 추가, 삭제, 수정 등의 작업을 하는 클래스

: 유지 보수 및 코드의 모듈화를 위해 클래스를 따로 만들어 사용한다.

# DTO (Data Transfer Object) or VO (Value Object)

: DAO 클래스를 이용하여 데이터 베이스에서 데이터를 관리할 때 데이터를 일반적인 변수에저장하여 사용할 수 있지만 해당 데이터의 클래스를 만들어 사용한다. 즉, 모든 변수 저장값들의 저장소라고 보면 된다.

**\* 개발 환경 정보**

- 개발 언어 : JSP

- 개발 IDE : Eclipse JAVA EE IDE

- 서버 : Apache Tomcat v9.0 Server

- 데이터베이스 : MySQL -> MariaDB (웹 호스팅 과정에서 바꾸게 됨, 커넥션 풀 설정 값만

바꾸면 되기 때문에 별 다른 어려운 과정이 없음)

**\* JSP (Java Server Page)**

- JSP는 Java 언어를 이용하여 HTML 페이지를 만들기 위한 방법 중 하나로, HTML 코드에 JAVA

코드를 넣어 동적 웹페이지를 생성하는 웹어플리케이션 도구이다.

- JSP는 사용자가 코드를 작성한 다음 컴파일 과정을 거치지 않고 서버에서 실행시킬 수 있다.

- JSP는 웹 서버에 의해 서블릿으로 변환된 다음 실행된다.

- 자바 서블릿(Java Servlets)은 서버에서 실행되는 자바 프로그램으로 서블릿이 사용자에 의해

요청 되면 서버에서는 HTTP request에 응답하기 위해 웹 서버에서 서블릿 객체를 생성하고

이를 통해 사용자의 요청을 처리한다.

**\* 홈페이지를 만들기 위한 웹 프로그래밍(JSP)에 사용한 언어**

- CSS 와 JavaScript는 부가적인 요소를 구성(디자인 등)해주기 위해 조금 넣어줌

① Java : Java 웹 어플리케이션 구현을 위한 실질적 코드

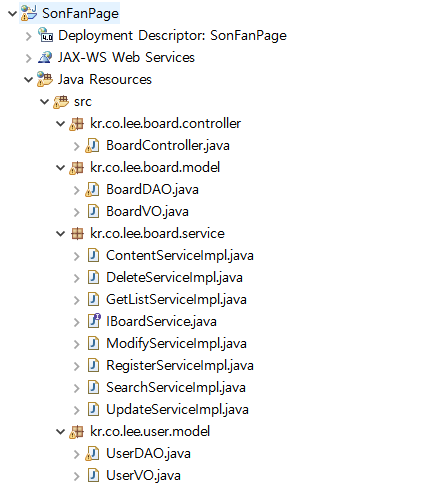
② HTML : 웹 어플리케이션 구현을 위한 틀

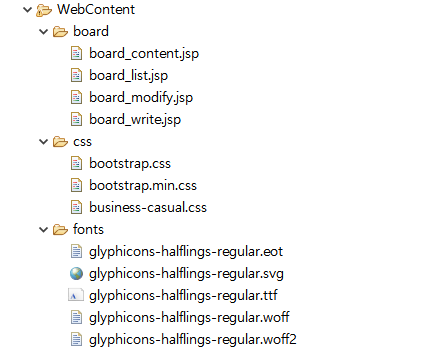
③ CSS : 웹 어플리케이션의 레이아웃 및 스타일을 지정하는 언어

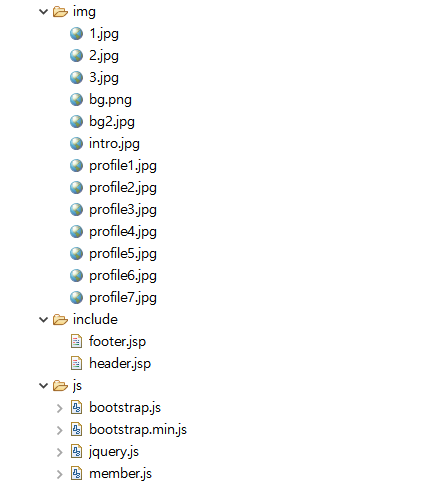
④ JavaScript : 클라이언트의 기능/동작을 구현하기 위한 언어

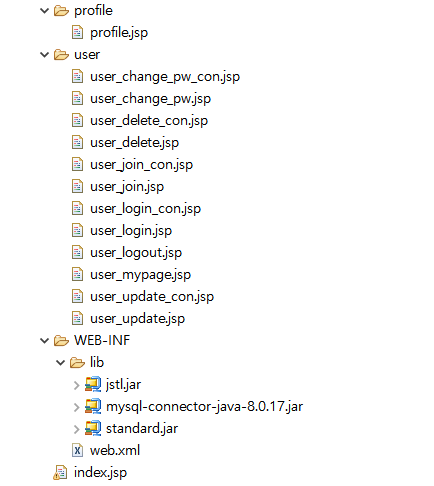
**3) 웹 페이지 작성에 대한 설명 및 코드**

**\* 전체적인 생성 파일 및 사용 라이브러리**

****

****

****

****

**\* 중요 일부 코드**

- 게시판은 모든 작업에 대해 Controller를 하나로 통합 -> BoardController.java

- 사용자 계정에 관한 작업은 각 작업에 대해 \*\_con.jsp 와 같이 control 페이지 따로 생성

- 코드에 대한 설명은 모두 코드 내 주석으로 처리 ( // or <!-- --> )

- 첨부 코드가 너무 많은 것 같지만 모두 꼭 필요하다 생각하여 첨부하였습니다.

**# BoardController.java**

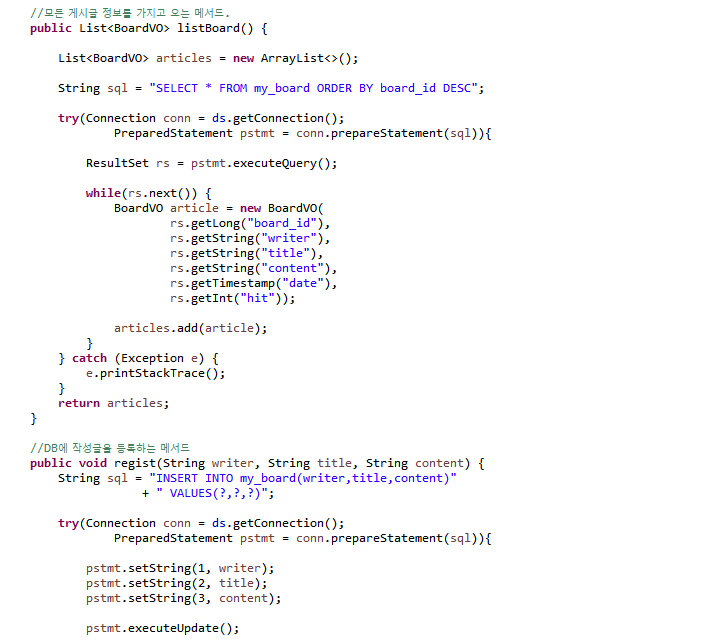




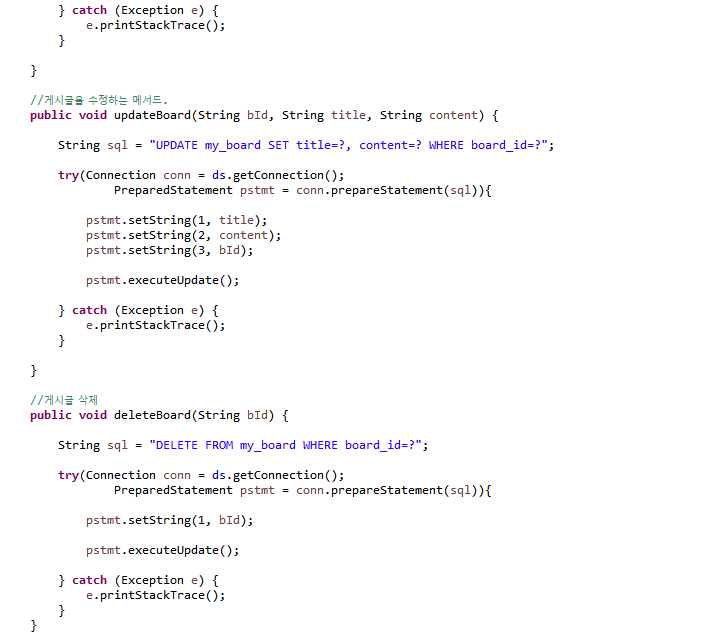


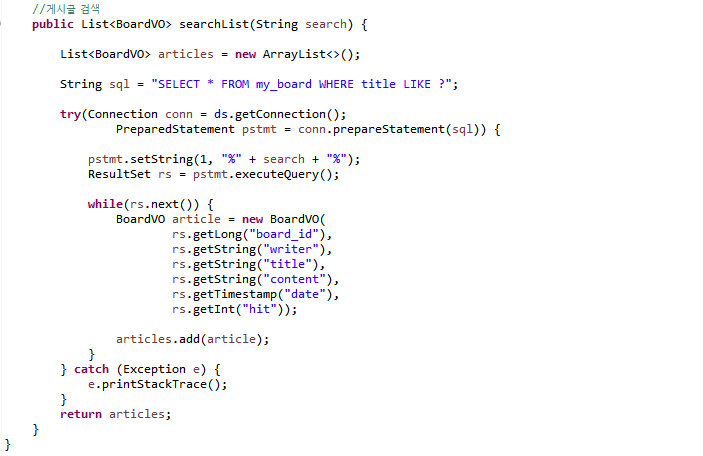
**# BoardDAO.java : 데이터 베이스에 접근해서 데이터 추가, 삭제, 수정 등의 작업을 하는 클래스, 여기서는 게시판에 관한 작업이 이루어지는 클래스**

****

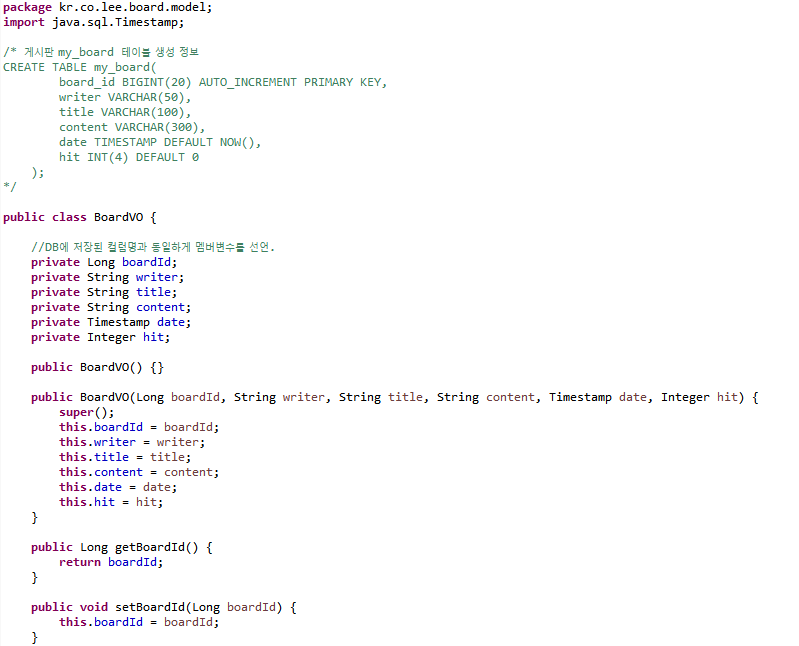
****

****

****

****

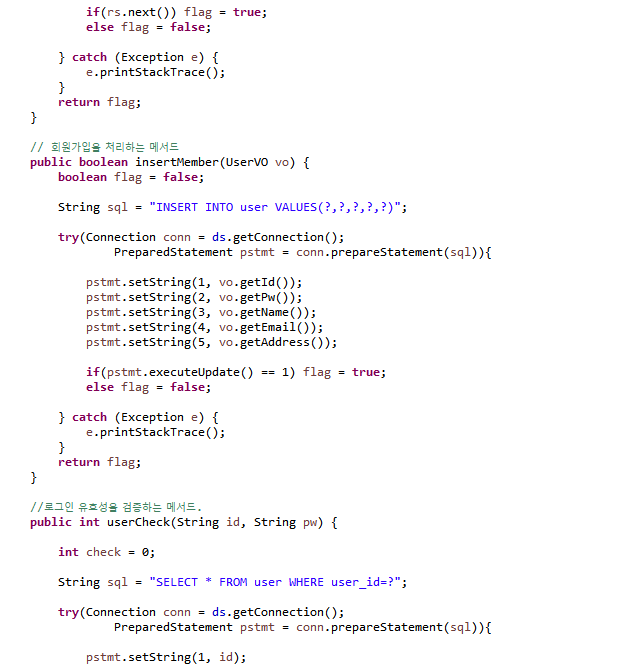
**# BoardVO.java : 데이터 베이스에서 데이터를 관리할 때 이용하는 데이터 값들의 저장소 (getter, setter), 여기서는 게시판에 관한 데이터 저장**

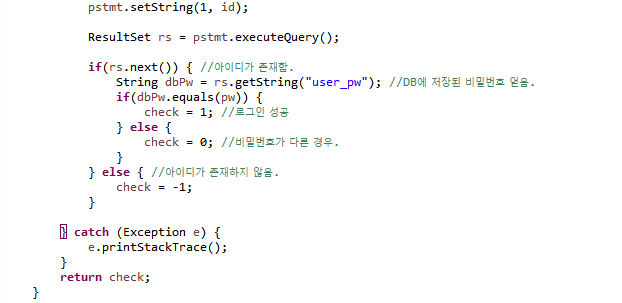
****

****

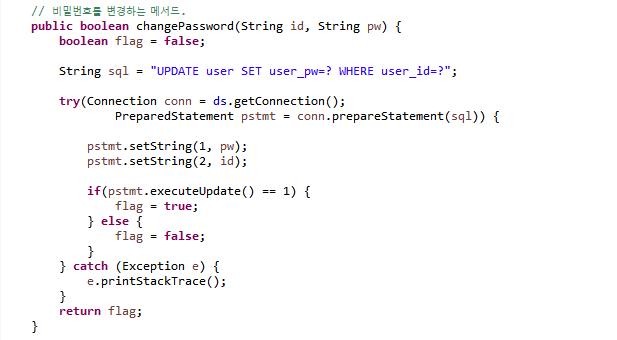
**# UserDAO.java : 데이터 베이스에 접근해서 데이터 추가, 삭제, 수정 등의 작업을 하는 클래스, 여기서는 사용자 계정에 관한 작업이 이루어지는 클래스**

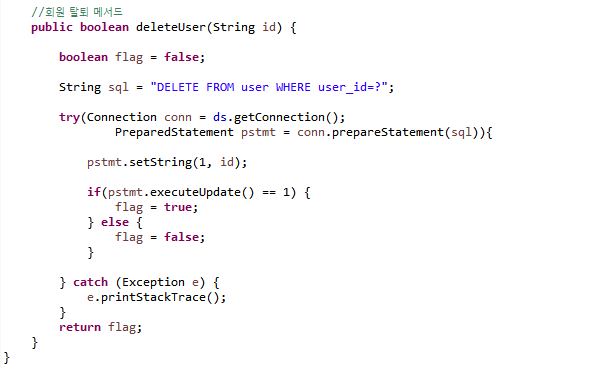
****

****

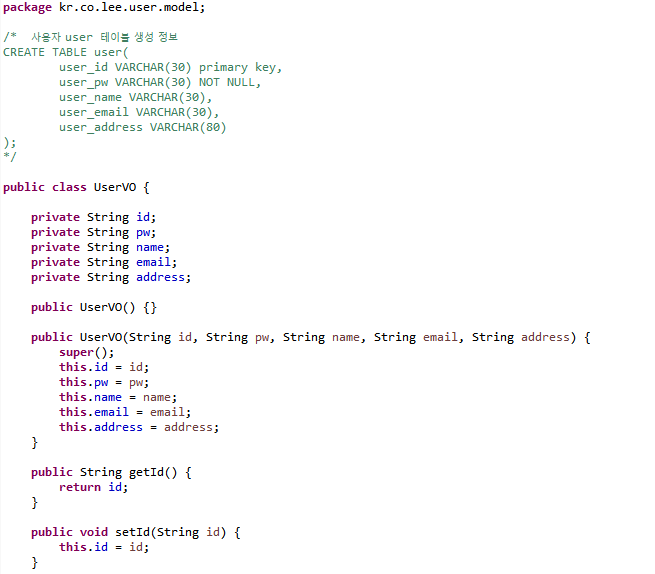
****

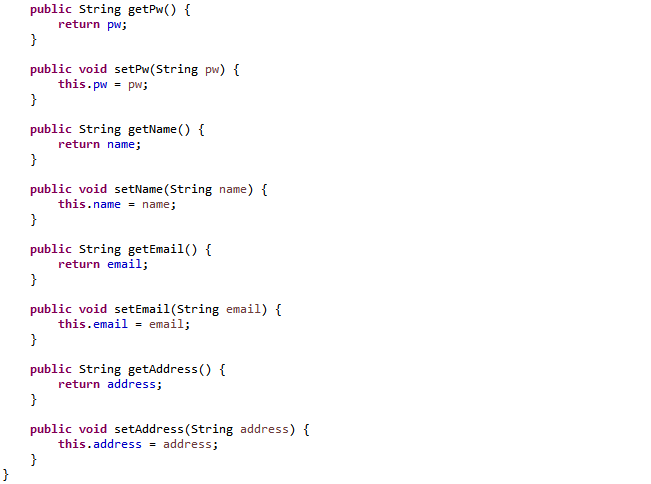
****

****

****

**# UserVO.java : 데이터 베이스에서 데이터를 관리할 때 이용하는 데이터 값들의 저장소 (getter, setter), 여기서는 사용자 계정에 관한 데이터 저장**

****

****

# 인터페이스 IBoardService.java 와 이를 확장(execute 메서드) 사용하여 각 기능을 담당하는 클래스들(DAO와 연계) –> 단순히 getParameter로 데이터 받아와서 동일하게 DAO 접근을 하는 코드이므로 주석 생략

# 인터페이스 - IBoardService.java : execute( ) 추상 메서드 정의 및 각 클래스 실행

# 확장 클래스 : getParameter( ) 로 데이터를 받아와 DAO getInstance 실행, 즉 데이터 베이스에 접근하여 데이터 저장 및 처리

- ContentServiceImpl.java : 게시판 조회수 처리 & 글 보기

- DeleteServiceImpl.java : 게시판 글 삭제 처리

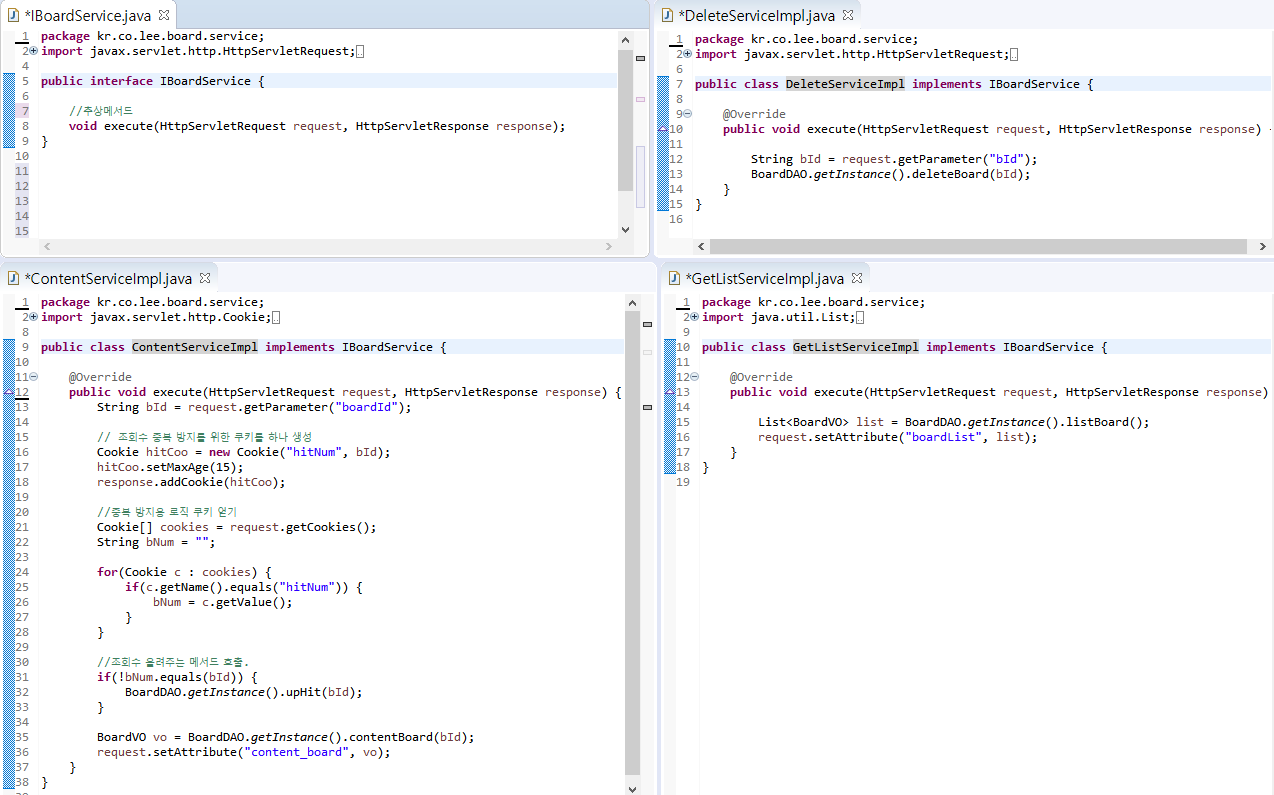
- GetListServiceImpl.java : 게시판 모든 글(목록) 보기 처리

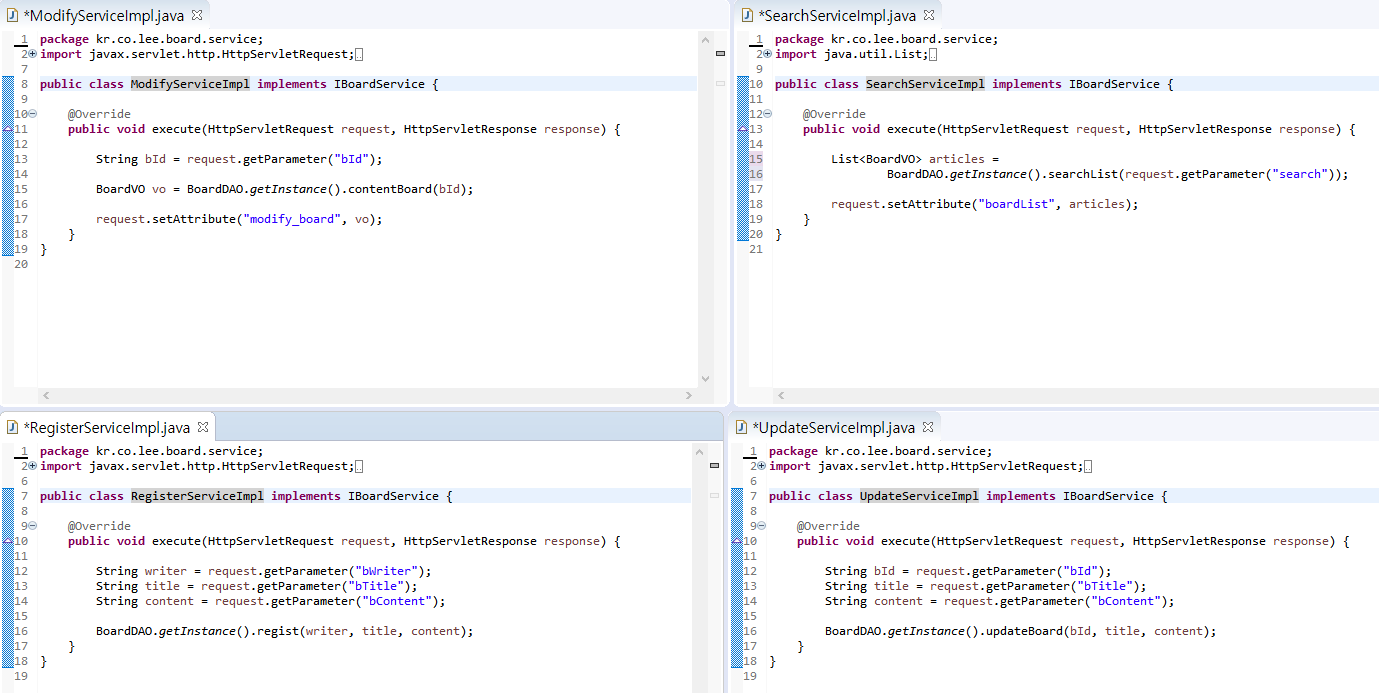
- ModifyServiceImpl.java : 게시판 글 보기 처리

- RegisterServiceImpl.java : 게시판 글 등록 처리

- SearchServiceImpl.java : 게시판 글 검색 처리

- UpdateServiceImpl.java : 게시판 글 수정 처리



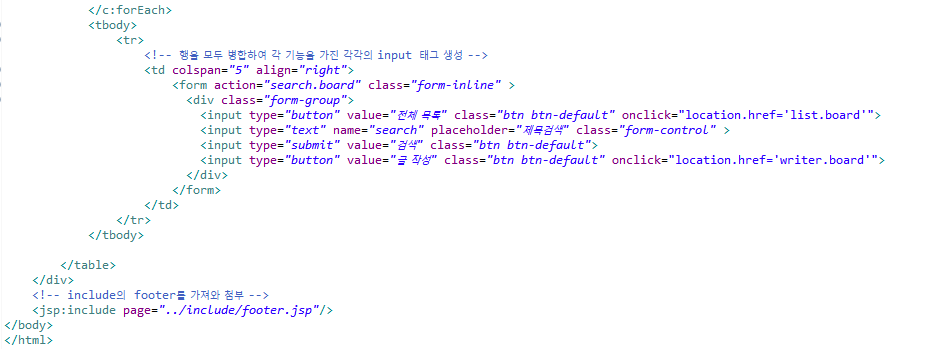


**# board\_content.jsp : 게시판 글 내용 보기 페이지 (주석은 <!-- > or // 로 표기)**



**# board\_list.jsp : 게시판 글 목록 보기 페이지 (주석은 <!-- > or // 로 표기)**





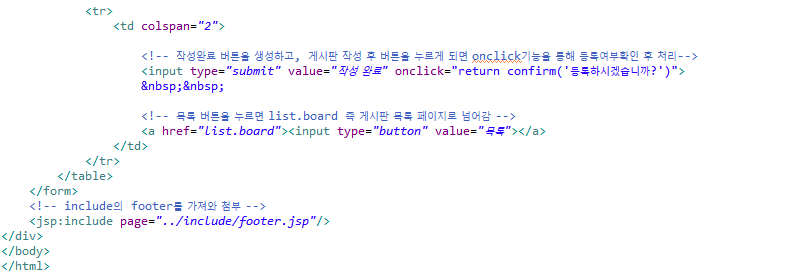
**# board\_modify.jsp : 게시판 글 수정 페이지 (주석은 <!-- > or // 로 표기)**





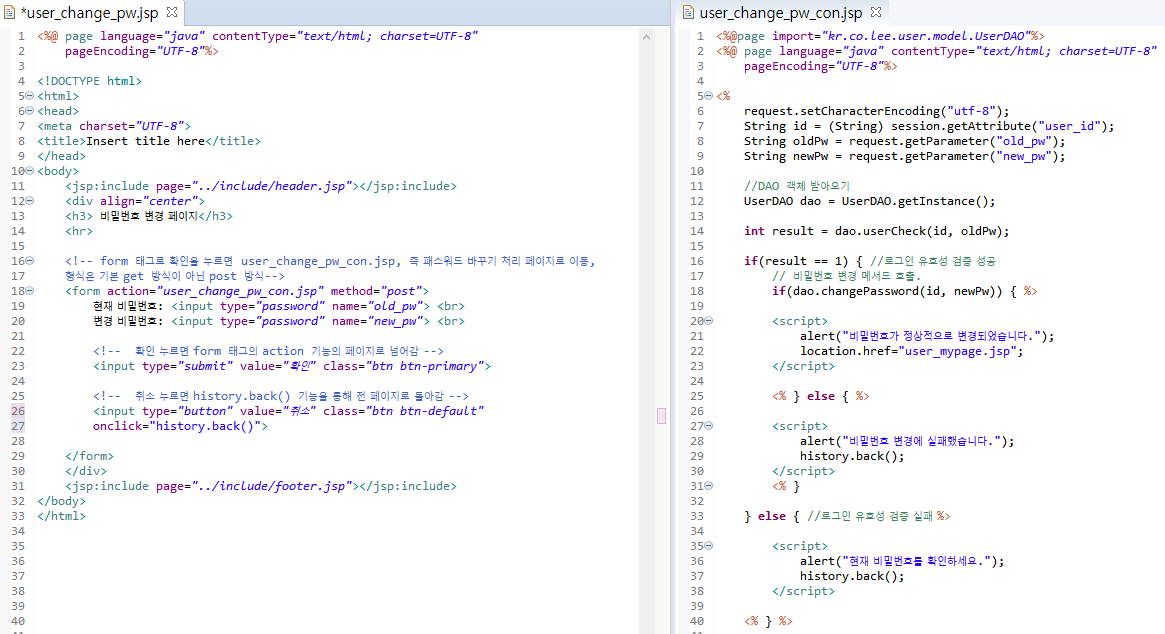
**# board\_write.jsp : 게시판 글 쓰기(등록) 페이지 (주석은 <!-- > or // 로 표기)**



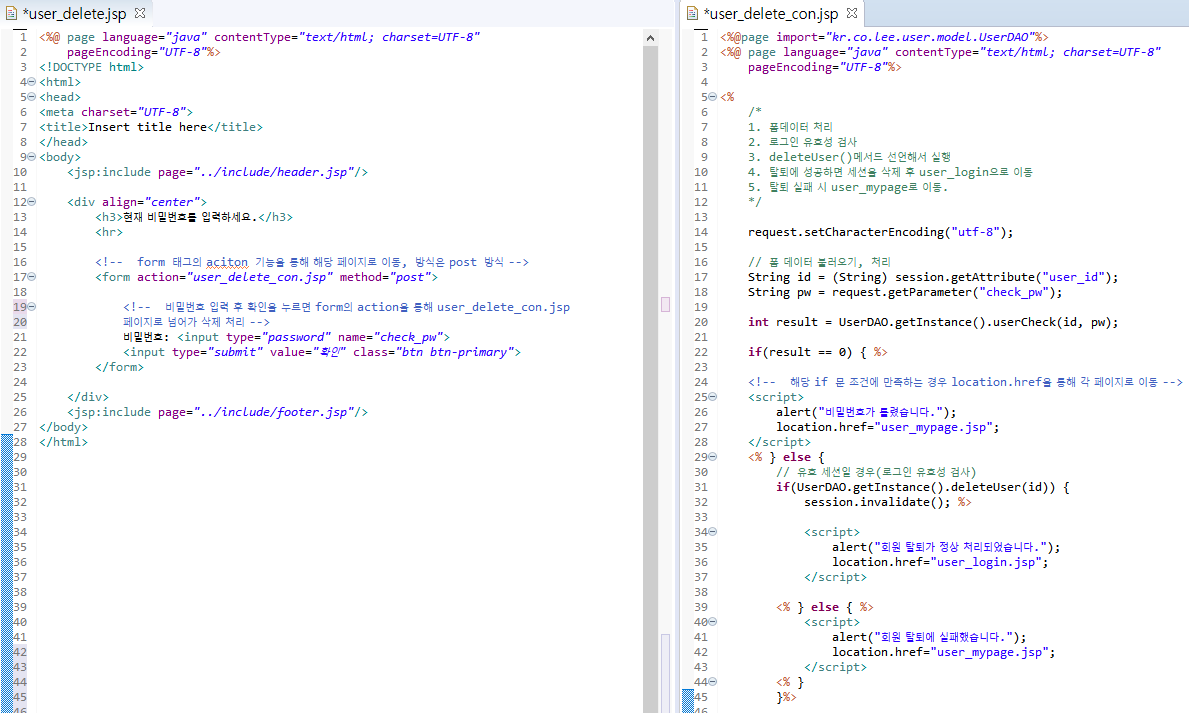


**# user\_change\_pw.jsp와 user\_change\_pw\_con.jsp**

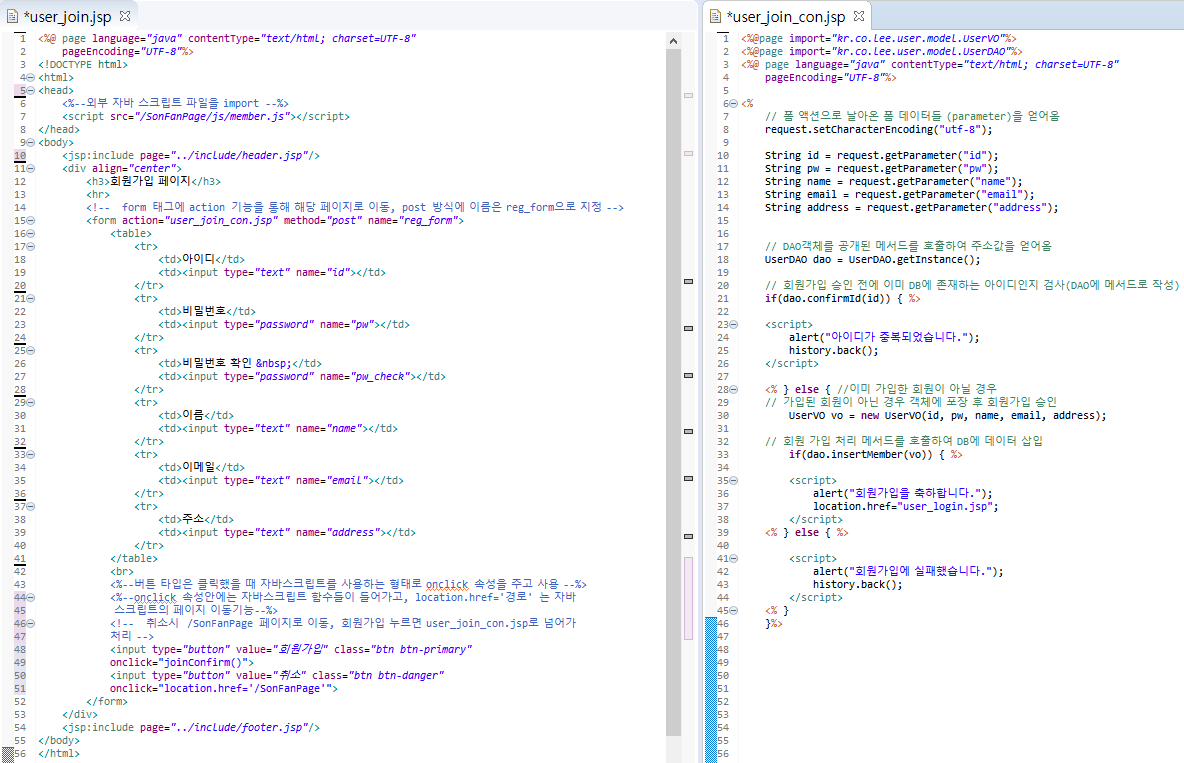
**: 비밀번호 변경 페이지와 컨트롤러(데이터 처리)**



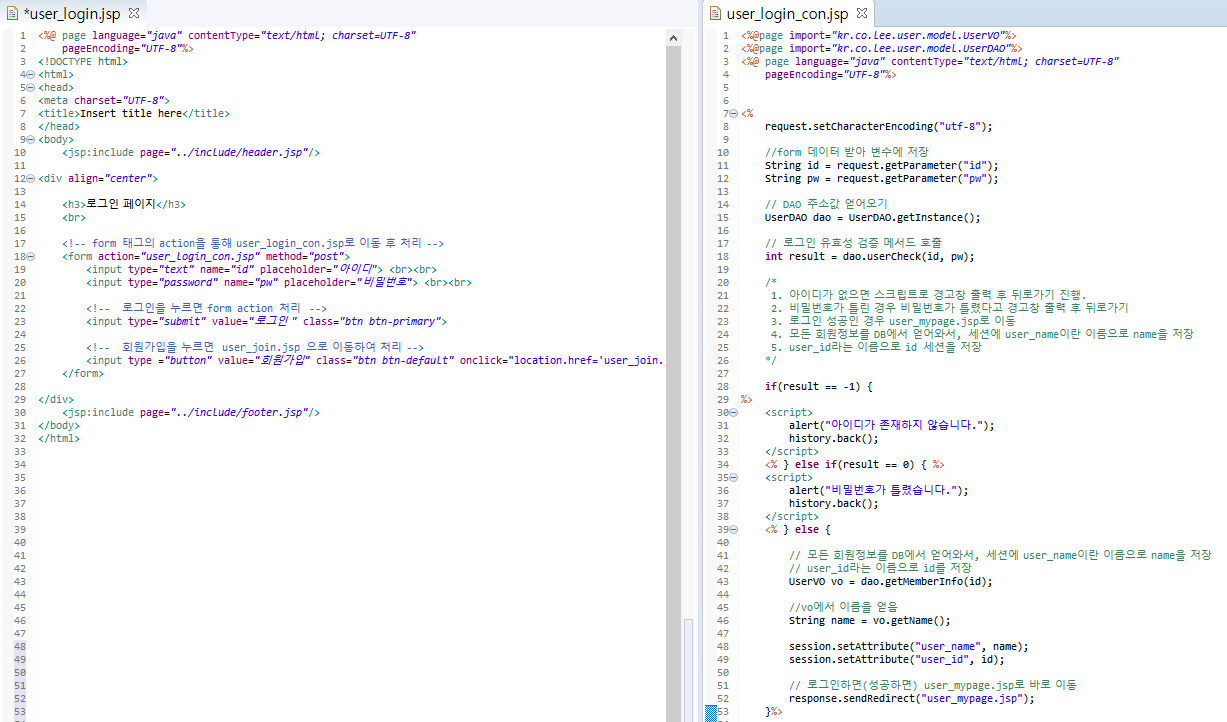
**# user\_delete.jsp와 user\_delete\_con.jsp : 회원탈퇴 페이지와 컨트롤러(데이터 작업 처리)**



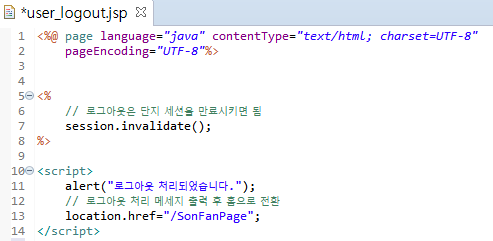
**# user\_join.jsp와 user\_join\_con.jsp : 회원가입 페이지와 컨트롤러(데이터 작업 처리)**



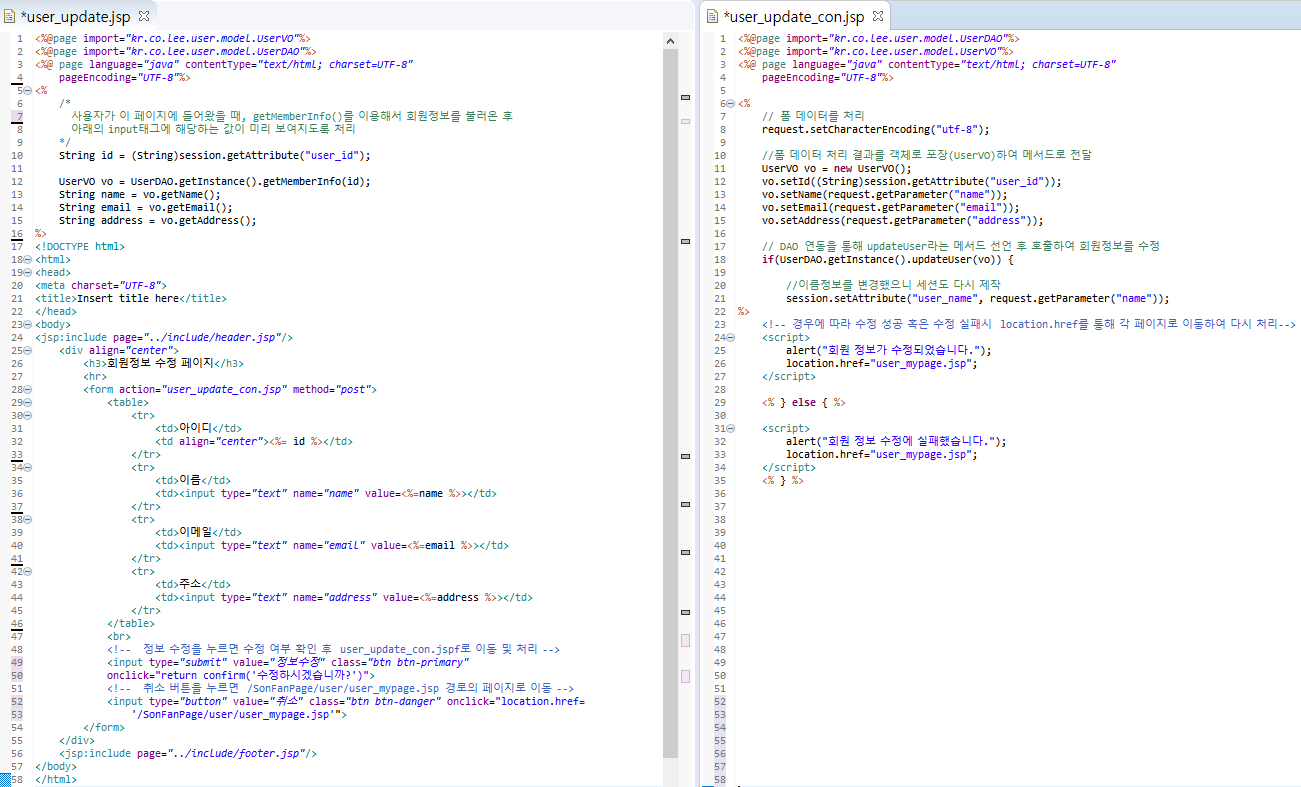
**# user\_login.jsp와 user\_login\_con.jsp : 로그인 페이지와 컨트롤러(데이터 작업 처리)**



**# user\_logout.jsp : 로그아웃 페이지**

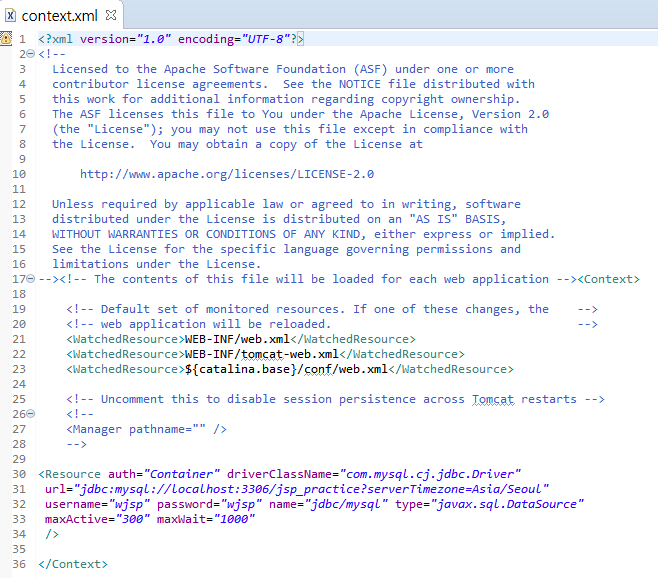


**# user\_update.jsp와 user\_update\_con.jsp : 회원 정보 수정 페이지와 컨트롤러(데이터 작업 처리)**

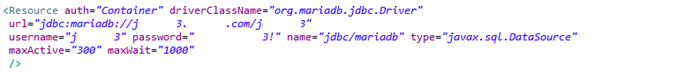


**# 커넥션 풀 설정** : 데이터 베이스로의 연결을 매번 하지 않고 context.xml 파일에 설정을 해두어 필요시 함수 호출과 같이 호출만하여 데이터 베이스로 연결할 수 있도록 한다. 아래의 캡쳐 사진과 같이 맨 마지막 부분에 자신이 사용하는, 연결한 데이터 베이스의 정보(Resource)를 형식에 맞게 적어두면 된다.

\* 원래 내 컴퓨터 localhost 환경에서



\* 웹 호스팅 환경



**4) 프로젝트 진행에 있어서의 개인적인 소감과 어려웠던 점 및 해결과정**

**\* 개인적인 소감**

: 사실 학교에서 하는 공부만으로는 개인적인 잘못이긴 하지만, 시험 공부에만 맞추다 보니

3학년 2학기가 되었음에도 불구하고 스스로 할 줄 아는 것도 하나 없고 앞으로의 취업이

걱정되어 뭐라도 해보기 위해 학원을 다니게 되었습니다. 학원에서 JSP 과목을 수강했는데

마침 인터넷 컴퓨팅 수업에서 홈페이지를 만드는 프로젝트를 제출하게 되어서 공부한 것을

활용하여 많이 어렵지만 그래도 스스로 만들어 보기 위해 배웠던 것을 바탕으로 최대한

살려 코드를 하나하나 작성하여 홈페이지를 만들었습니다. 어떻게 보면 HTML만을 활용하여

틀만 있는 홈페이지를 만들었다면 훨씬 수월하게 금방 만들 수 있었을 것입니다.

그러나 스스로 한번 해보겠다는 뭔가 남겨 보겠다는 생각을 가지고 JSP를 활용하여 홈페이지를

만들어봤습니다. 코드를 작성하면서 많은 공부가 되었고 개인적으로 무엇인가를 만들었다는

성취감과 함께 뿌듯함도 느꼈습니다. 물론 홈페이지가 부족한 부분이 많고 어설픈 부분이

많지만 솔직히 개인적으로 프로젝트 점수와 관계없이 스스로 공부했다는 것에 의미를 두고

싶습니다. 이번 프로젝트를 통해 다른 사람한테는 단지 제출하기 위한 프로젝트였을 수 있지만,

저한테는 굉장히 의미 있는 많은 것을 배우게 해준 프로젝트였습니다. 프로젝트 제출 이후에도

조금 더 보고 수정해서 더 나은 홈페이지를 만들어 볼 것입니다. 마지막으로 추가하자면, 웹

호스팅 과정에서 DB 연결하는데 있어 어려움이 있었는데 계속 무엇이 잘못되었는지 보면

스스로 문제를 해결할 수 있어서 아주 큰 공부가 되었고 뿌듯했습니다.

**\* 어려웠던 점 및 해결과정**

: 솔직히 하나 하나 다 어려웠습니다. 그러나 이해하고 보면 낯설었을 뿐 그렇게 생각한 만큼 어렵지 않았다는 것을 깨닫게 되었습니다. 그러나 코드를 다 작성하고 홈페이지를 실행시켜 보니 게시판 글 번호 부분에서 문제가 생겼습니다. 다른 기능은 정상적으로 작동하는데, 게시판의 글을 새로 작성하거나 삭제를 하면 글 번호에 대한 처리는 이루어지지 않는 것입니다. 예를 들어 글이 3개 등록되어 있어 각 글은 밑에서부터 차례대로 1, 2, 3번의 글 번호를 갖는데, 이 리스트에서

2번 글을 지우면 2번 글이 삭제되면서 게시글 목록에 3번의 글 번호가 자동으로 2번으로 바뀔 줄 알았는데 바뀌지 않고 그대로 3번으로 남아있는 것이었습니다. 또한 게시글을 다 지우고 홈페이지를 닫고 다시 실행시켜 게시글 목록에 들어가서 게시글을 추가하면 1번에서부터 시작하지 않고 삭제 전의 마지막 번호부터 나열되는 것이었습니다. 그래서 찾아보고 생각해보니 데이터 베이스에 글 번호를 저장하고 그 번호를 그대로 받아오면서 처리가 되어 이러한 문제가 생기는 것이었습니다. 그런데 다시 생각해보니 데이터 베이스 상에서 글 번호는 primary key로 auto\_increment가 설정되어 자동으로 바뀌어야 하는 게 아닌가? 라는 의문이 들었습니다. 그래서 혹시나 쉽게 생각해서 mysql 상에서 쿼리 문으로 ALTER TABLE my\_board AUTO\_INCREMENT = 1; 을 작성하여 실행해봤는데, 역시나 정상적으로 작동하지 않았습니다. 그래서 두번 째로 생각한 것이 이클립스 내 코드에서 count 변수를 통해 게시글 번호를 처리하는 방법이었습니다. 게시글이 하나도 없으면 혹은 다 삭제되어 없어지면 글 번호의 count 값을 0으로 주고 …. 하면 되겠지라고 일단 생각해보았습니다. 그런데 어렵게 생각해서 그런지 모르겠지만 게시글이 삭제될 때 마다 자동으로 count 값을 변경하는 코드를 생각해내지 못하였습니다. 코드 내에서 수정을 하면 수정한 값이 또 데이터 베이스에 잘 들어갈지도 헷갈렸습니다. 쉽게 말해서 데이터 베이스와 개발 환경을 왔다갔다 하는 데이터 값을 어디서 다뤄야 할지 잘 몰랐던 것이었습니다. 또한 코드 내에서 바꾼다 해도 자동 값 변경의 count 변수를 다루는 법을 잘 모르는 것이었습니다. 그래서 서칭을 해봐도 제가 생각한 코드 내 count 변수를 다루는 것이 맞는 것 같았지만 해결을 하지 못하였습니다. 코드를 다 작성한 후에 수정하려니 또 머리가 너무 복잡해져서 이 부분이 너무 어려웠습니다. 그래서 프로젝트를 제출하고 시험이 끝난 이후에라도 반드시 이 부분을 수정해보려 합니다.