**웹플랫폼 개발**

**최종 결과 제출**

멀티미디어공학과

2012112851 장규영

1. **index/enrollment\_check.jsp**

삽입 / 수정 / 삭제가 진행되는 화면

(1) 회원가입 첫 페이지





(2) 필수 입력 사항을 입력하지 않았을 때





(3) 로그인 후에는 enrollment\_check 페이지가 개인정보 수정 페이지로 바뀌는 모습,

가입 버튼도 수정 완료 버튼으로 바뀌는 모습





(4) 필수 사항을 입력하지 않고 수정 완료를 눌렀을 때

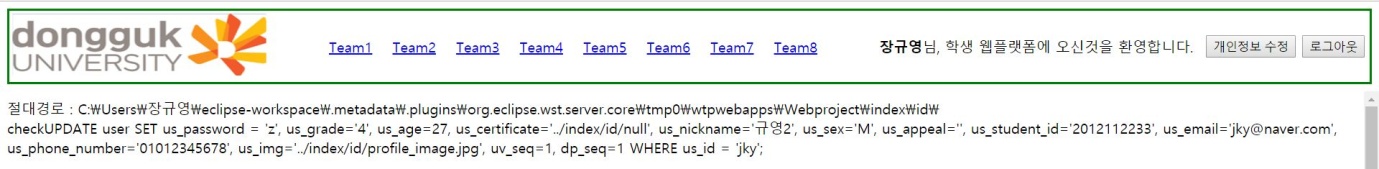




(5) 회원가입 삽입



(6) 회원가입 수정



(7) 회원가입 삭제



쿼리문1. enrollment.jsp에서 입력한 회원가입 아이디가 중복되는지 확인하는 쿼리문

SELECT us\_id FROM user WHERE us\_id='enrollment.jsp에서 입력한 아이디'"

쿼리문2. 현재 로그인한 아이디가 없을 때 회원가입 정보를 삽입

INSERT INTO user(us\_id, us\_pw, us\_name, us\_grade, us\_age, us\_certificate, us\_nickname, us\_sex, us\_appeal, us\_student\_id, us\_email, us\_phone\_number, us\_img, uv\_seq, dp\_seq) VALUES('아이디', '비밀번호', '이름', 학년, 나이, '재학증명서주소','닉네임', '성별', '어필멘트', '학번', '이메일', '핸드폰번호', '이미지 주소', 대학교번호, 전공번호);

쿼리문3. 현재 로그인한 아이디가 있을 때, 회원 정보를 수정

UPDATE user SET us\_password = '패스워드', us\_grade='학년', us\_age=나이, us\_certificate='재학증명서 주소', us\_nickname='별명', us\_sex='성별', us\_appeal='어필멘트', us\_student\_id='학번', us\_email='이메일', us\_phone\_number='핸드폰번호', us\_img='이미지 주소', uv\_seq=학교번호, dp\_seq=전공번호 WHERE us\_id = '유저 아이디';

쿼리문4. 대학교 정보 조회

SELECT uv\_seq, uv\_name FROM university;

쿼리문5. 전공 정보 조회

SELECT dp\_seq, dp\_name FROM department

회원 탈퇴는 index/hidden.jsp에서 처리

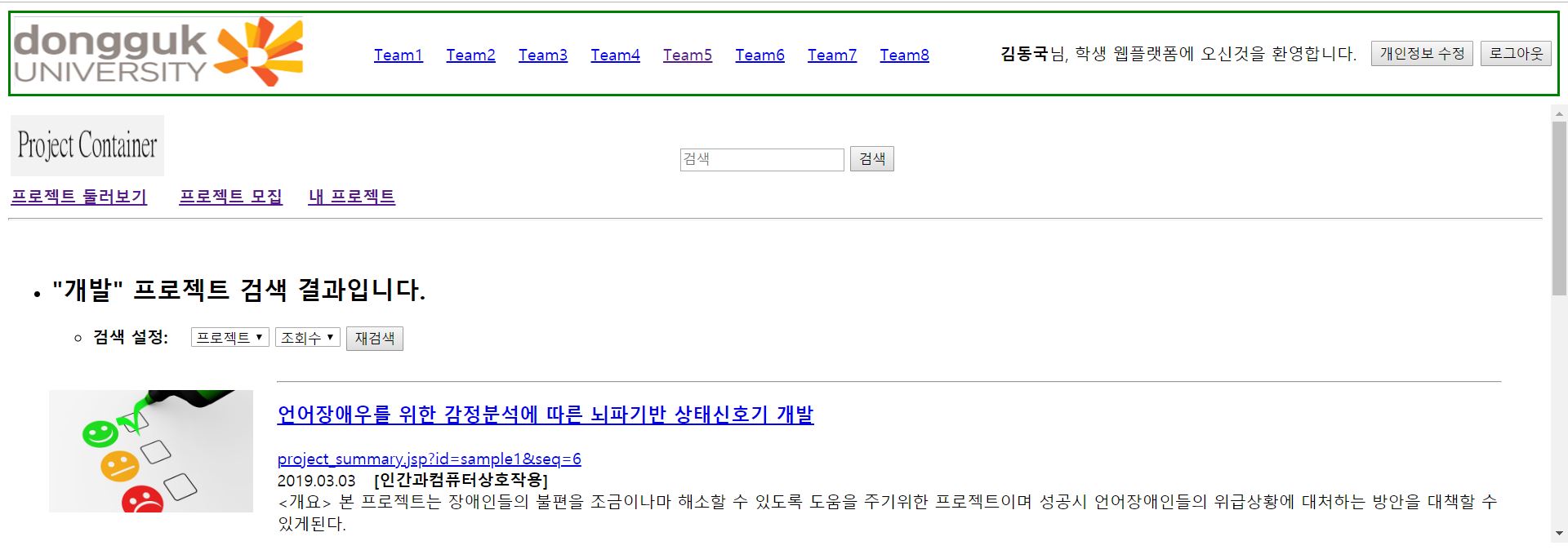
쿼리문1.

DELETE FROM user WHERE us\_id = '현재 로그인한 아이디';

2. **5/search.jsp**

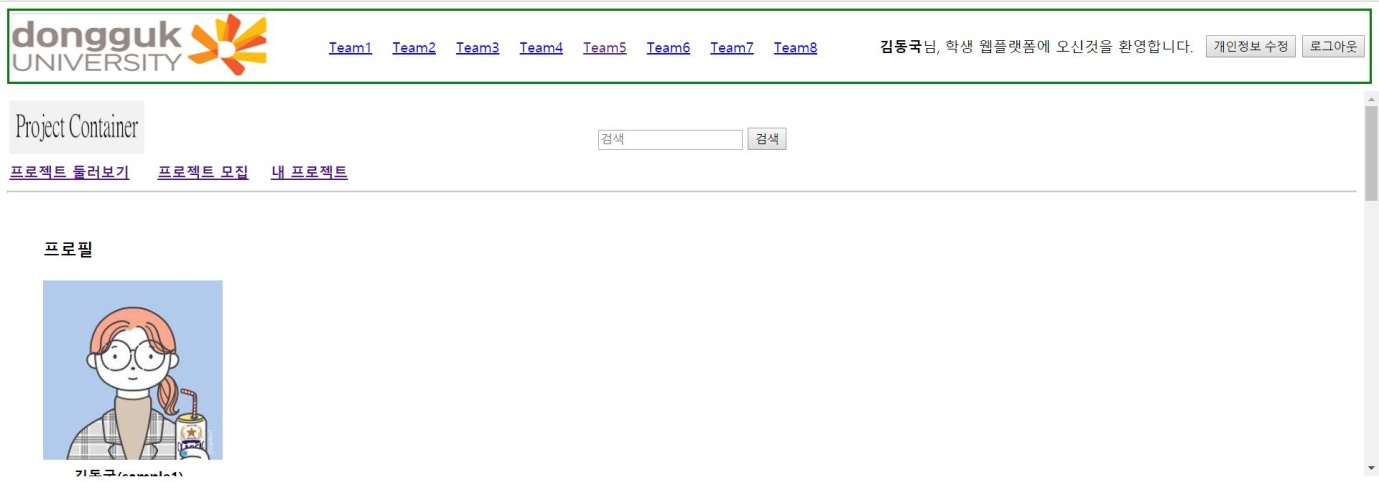
조회

(1) “개발”을 검색했을 때, 프로젝트 중에서 조회수가 높은 순서대로 출력





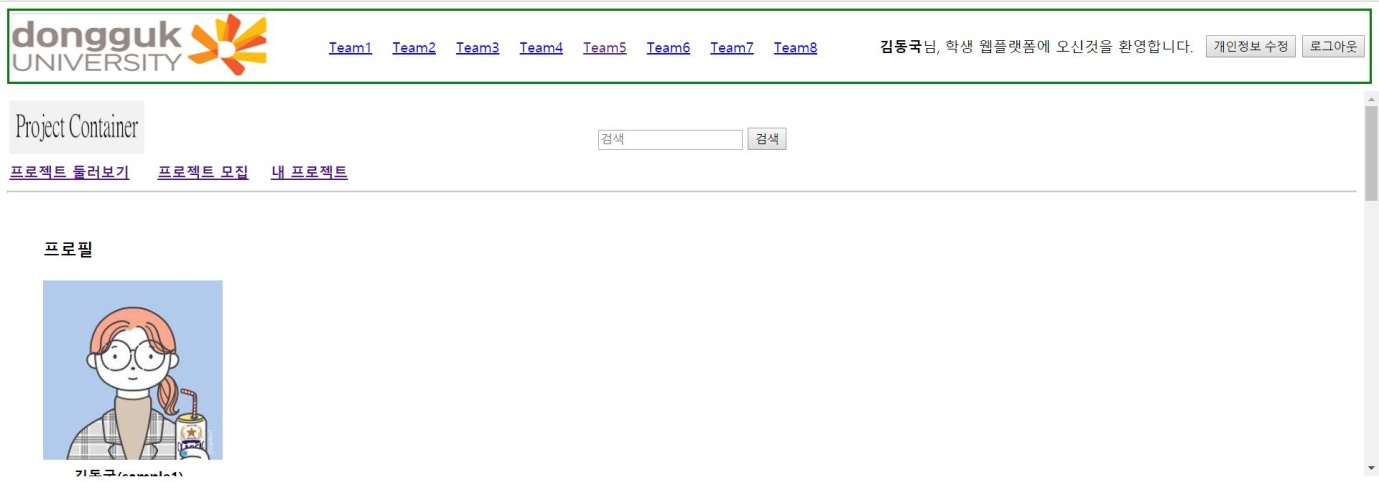
(2) “개발” 검색 결과를 최신순 기준으로 재검색했을 때 모습

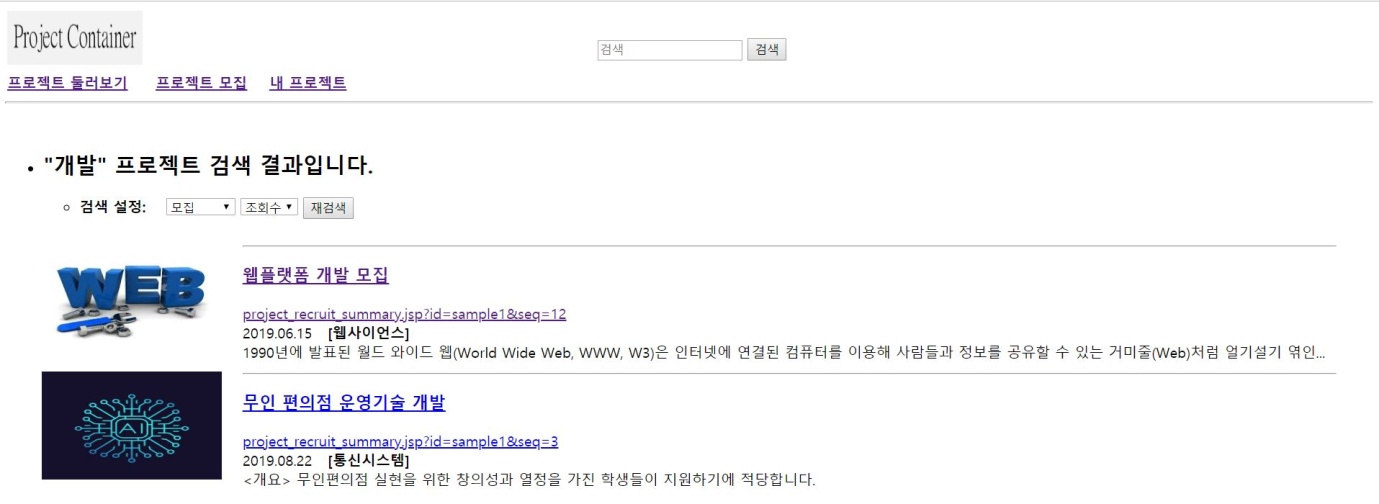






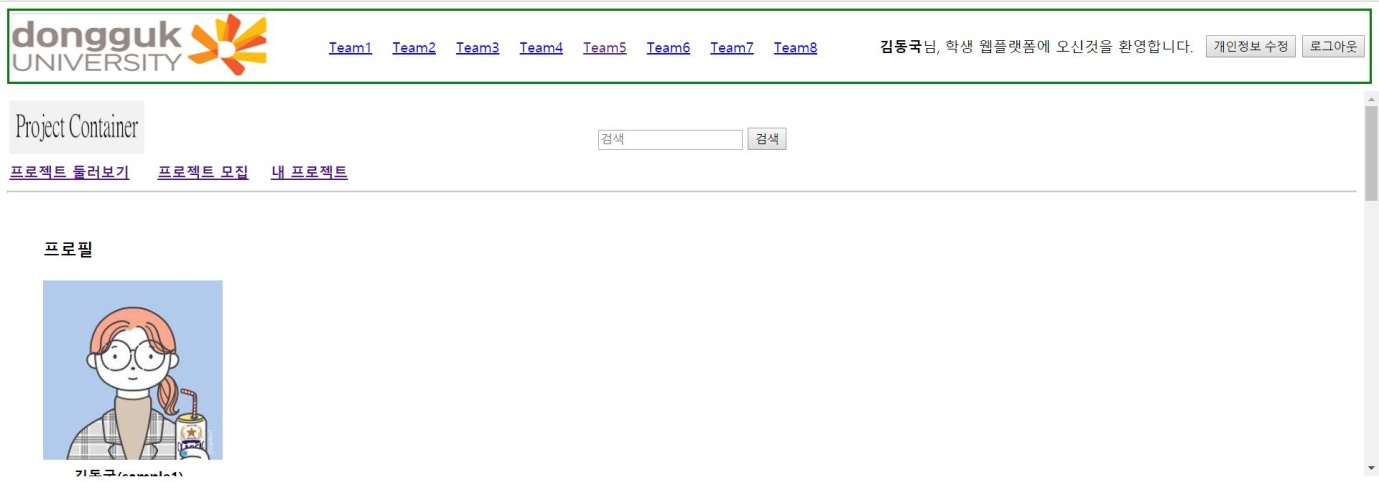
(3) “개발”이 포함된 모집 프로젝트에 대해 재검색한 모습







(4) “안녕”이 포함된 프로젝트가 없는 경우







쿼리문1. 검색어가 포함된 프로젝트를 조회수가 높은 순으로 보여줌

SELECT pj\_seq, pj\_img, pj\_name,ct\_name, pj\_start, pj\_text FROM (SELECT ct\_seq, pj\_seq, pj\_name, pj\_img, pj\_text, pj\_start, pj\_hit FROM project WHERE pj\_name LIKE '%검색어%') a JOIN category b ON a.ct\_seq = b.ct\_seq ORDER BY pj\_hit DESC;

쿼리문2. 검색어가 포함된 프로젝트를 나중에 등록된 순으로 보여줌

SELECT pj\_seq, pj\_img, pj\_name,ct\_name, pj\_start, pj\_text FROM (SELECT ct\_seq, pj\_seq, pj\_name, pj\_img, pj\_text, pj\_start, pj\_hit FROM project WHERE pj\_name LIKE '%검색어%') a JOIN category b ON a.ct\_seq = b.ct\_seq ORDER BY pj\_seq DESC;

쿼리문3. 검색어가 포함된 모집정보를 조회수가 높은 순으로 보여줌

SELECT rp\_img, rp\_seq, rp\_name,rp\_start,ct\_name,rp\_text FROM (SELECT ct\_seq, rp\_seq, rp\_name, rp\_img, rp\_text, rp\_start, rp\_hit FROM recruit\_project WHERE rp\_name LIKE '%검색어%') a JOIN category b ON a.ct\_seq = b.ct\_seq ORDER BY rp\_hit DESC;

쿼리문4. 검색어가 포함된 모집정보를 나중에 등록된 순으로 보여줌

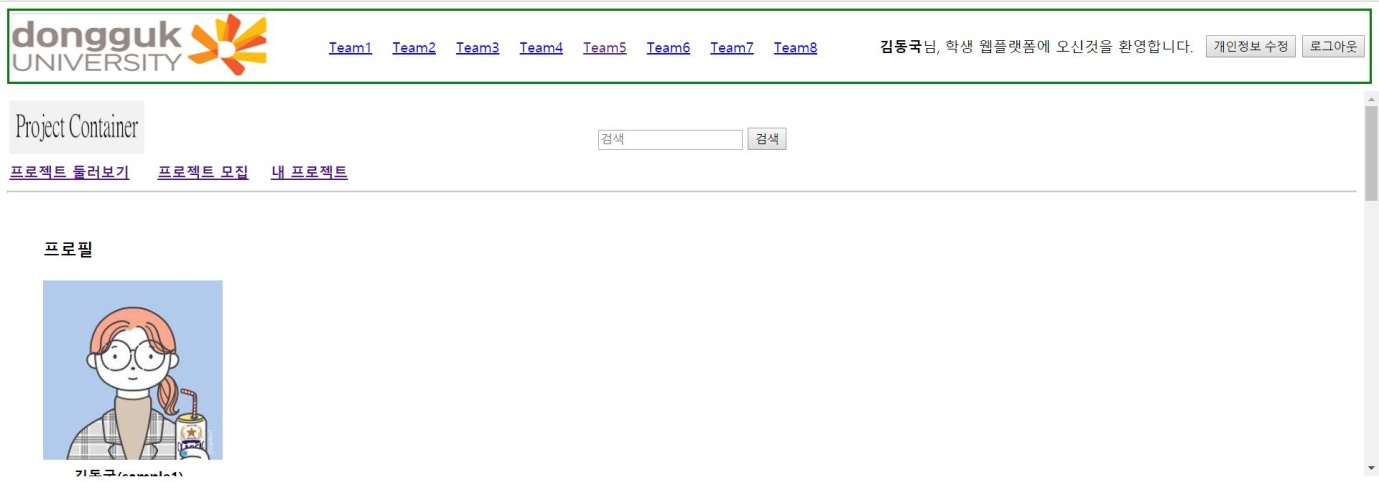
SELECT rp\_img, rp\_seq, rp\_name,rp\_start,ct\_name,rp\_text FROM (SELECT ct\_seq, rp\_seq, rp\_name, rp\_img, rp\_text, rp\_start, rp\_hit FROM recruit\_project WHERE rp\_name LIKE '%검색어%') a JOIN category b ON a.ct\_seq = b.ct\_seq ORDER BY rp\_seq DESC;

3. **5/my\_project.jsp**

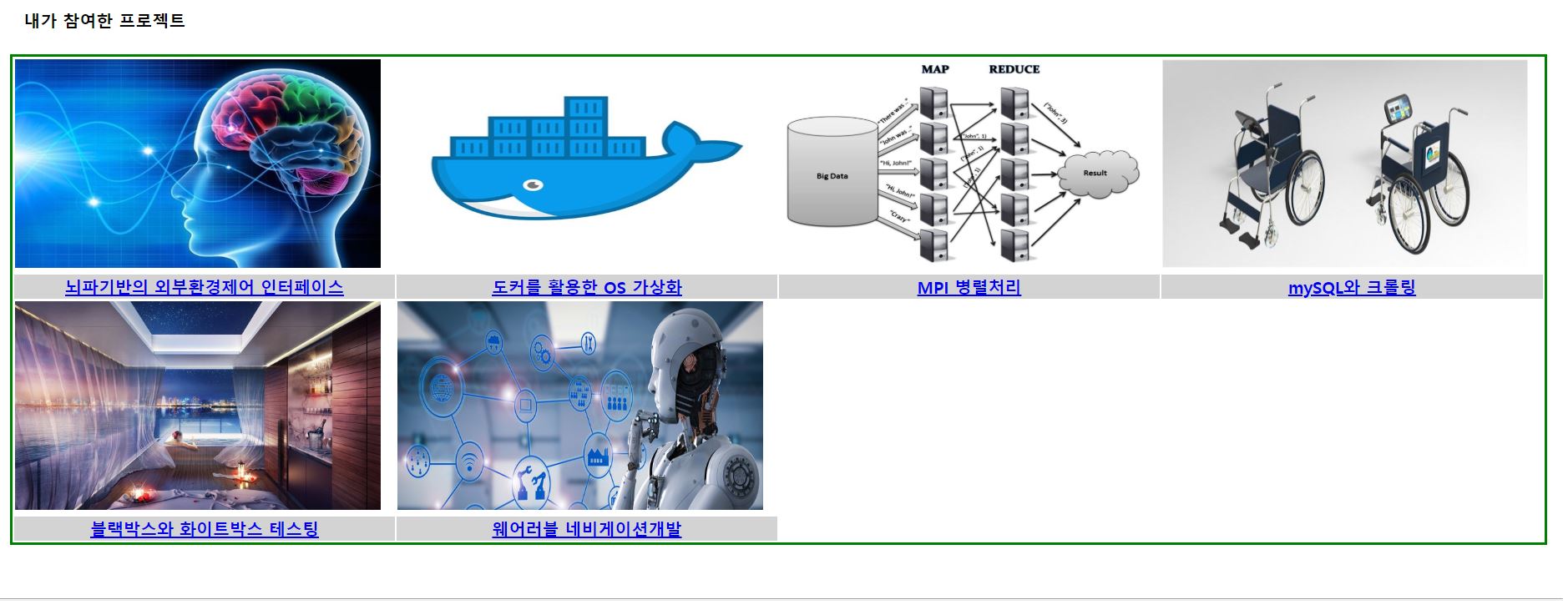
조회

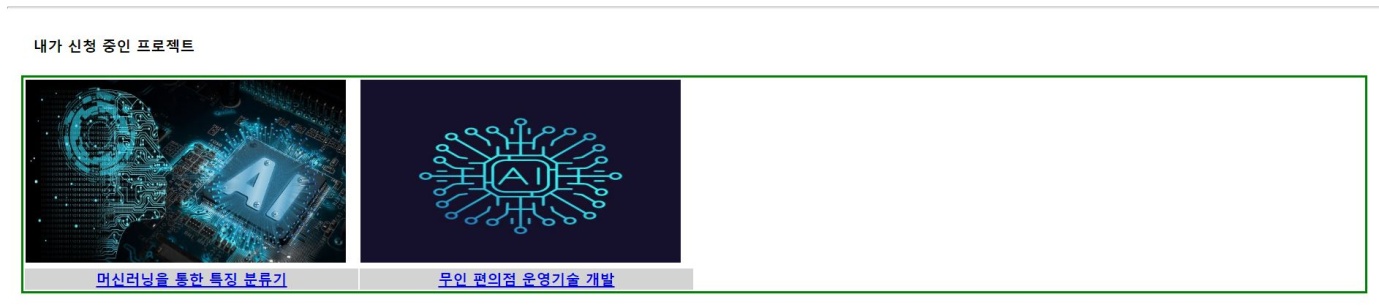
페이지 설명: 사용자의 간략한 프로필 정보를 보여주고, 참여한 프로젝트 목록과 참여 신청을 한 프로젝트 목록을 보여주는 페이지

(1) 마이 프로젝트 페이지가 프로필, 참여한 프로젝트, 내가 신청 중인 프로젝트 순으로 이루어진 모습











쿼리문1. 사용자의 프로필 정보 가져오기

SELECT us\_id, us\_name, us\_grade, us\_email, us\_img, dp\_name, us\_appeal, uv\_name FROM user a JOIN (select uv\_seq, uv\_name, dp\_seq, dp\_name from department c Join university d) b ON a.dp\_seq = b.dp\_seq AND a.uv\_seq = b.uv\_seq WHERE a.us\_id = ' 현재 로그인한 아이디';

쿼리문2. 사용자가 참여한 프로젝트 목록 가져오기

SELECT a.pj\_seq, pj\_name, pj\_img FROM project a JOIN project\_member b ON a.pj\_seq = b.pj\_seq WHERE us\_id = '현재 로그인한 아이디';

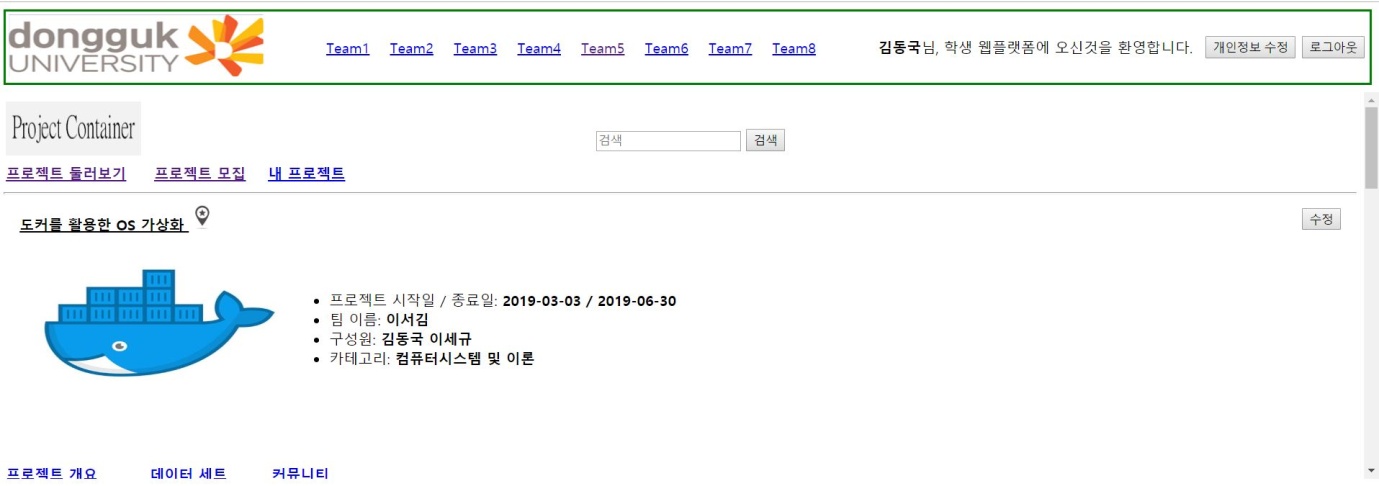
쿼리문3. 사용자가 지원한 모집 프로젝트 목록 가져오기

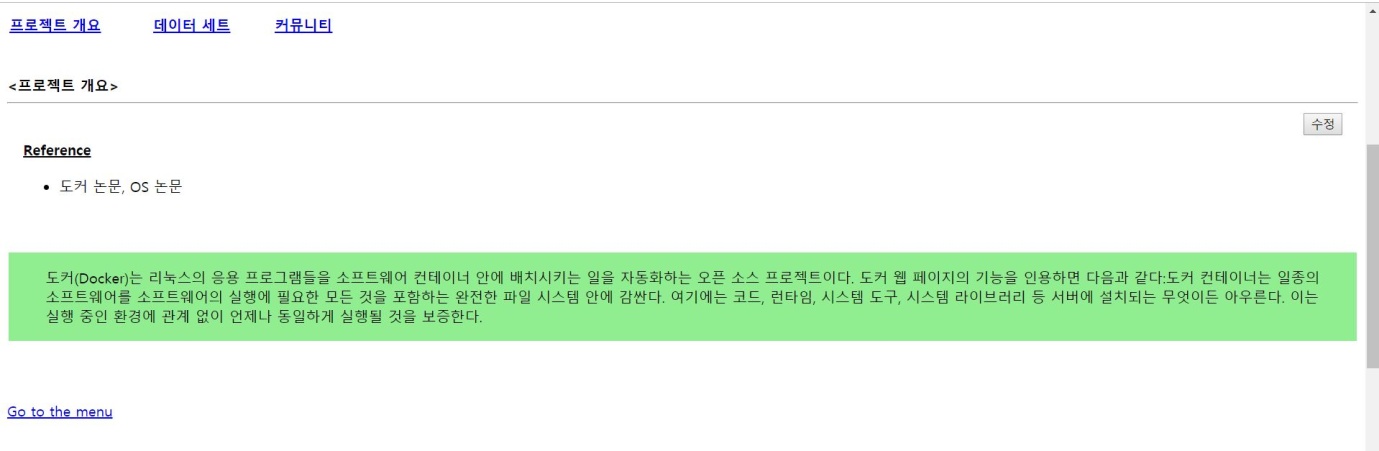
SELECT a.rp\_seq, rp\_name, rp\_img FROM recruit\_project a JOIN recruit\_project\_applier b ON a.rp\_seq = b.rp\_seq WHERE b.us\_id = '현재 로그인한 아이디';

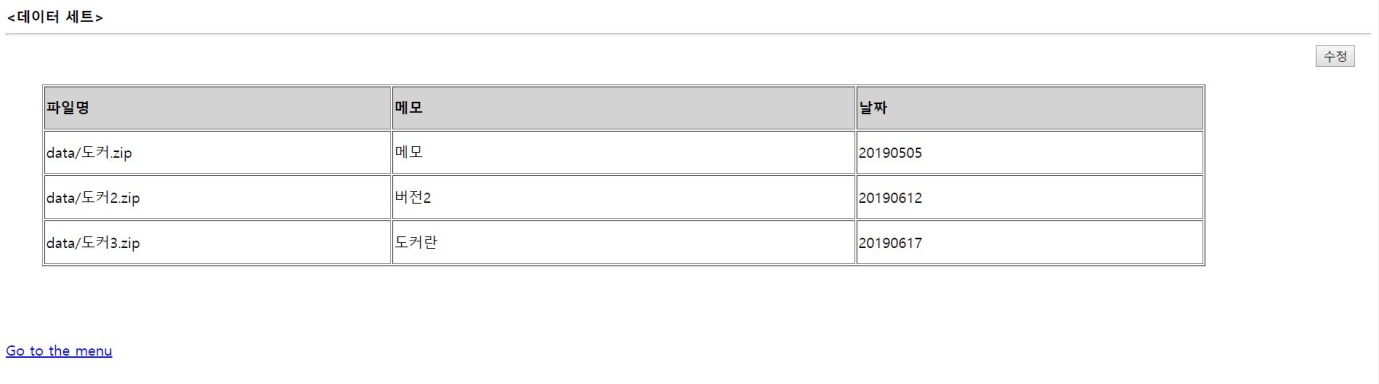
4. **5/project\_community\_modification.jsp**

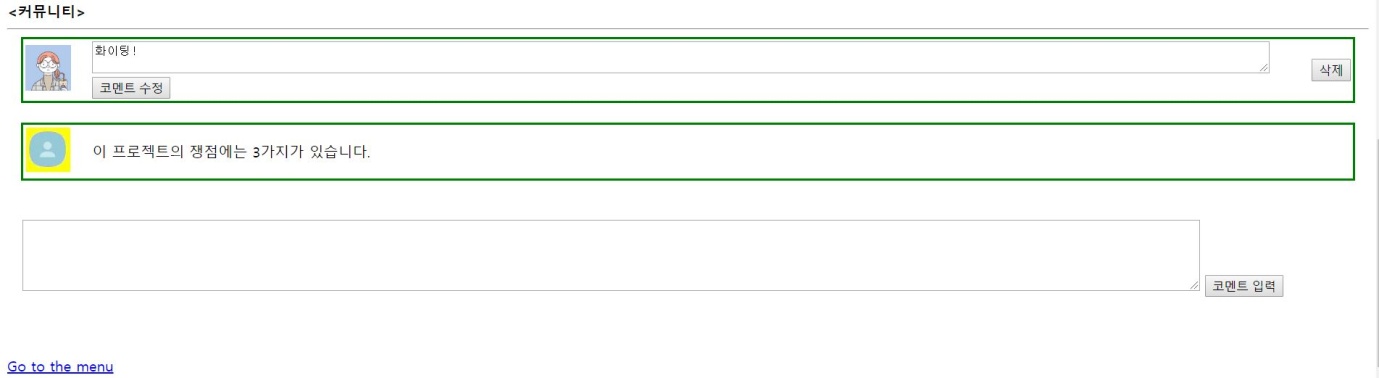
삽입 / 수정 / 삭제가 진행되는 화면

(1) 자신이 쓴 코멘트를 수정하는 창, 아래 텍스트 박스에서 새로운 코멘트를 입력할 수도 있음



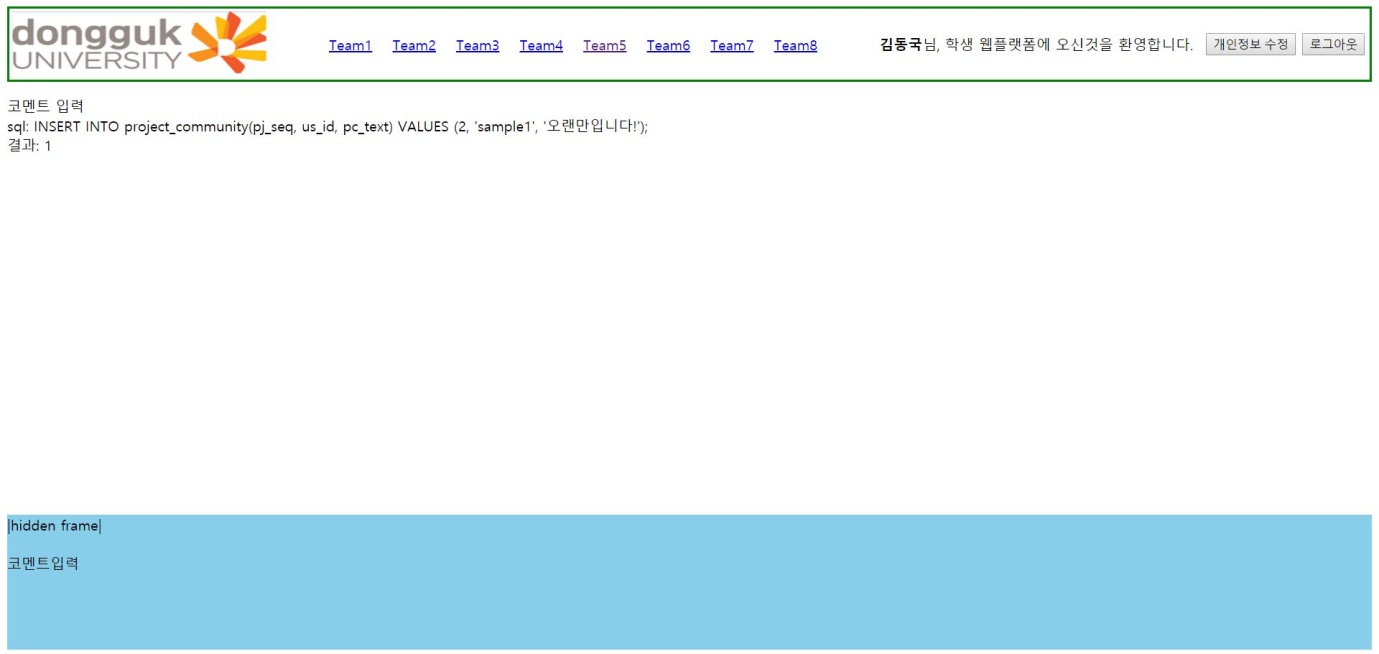




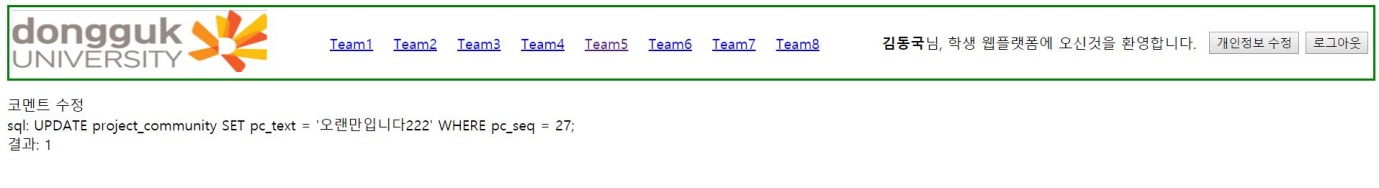




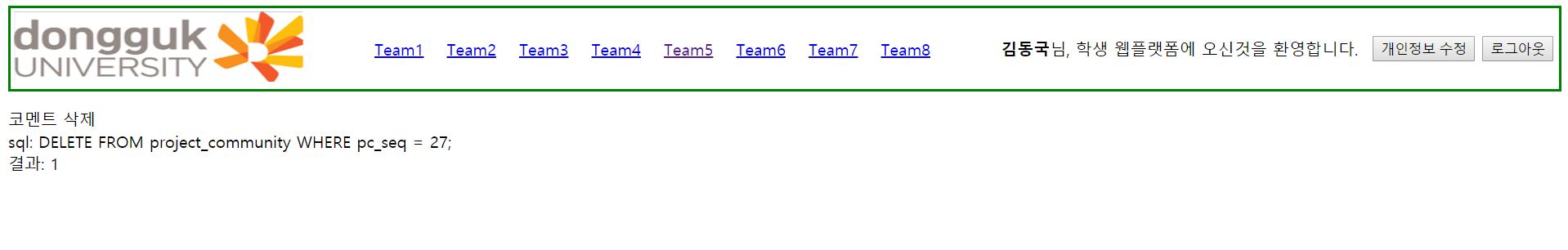
(2) 텍스트 박스에서 새로운 코멘트 삽입



(3) 기존에 작성했던 코멘트 수정



(4) 기존에 작성했던 코멘트 삭제



(조회)

쿼리문1. 프로젝트 상세 정보 가져오기

SELECT pj\_name, pj\_img, pj\_start, pj\_end, pj\_team, ct\_seq, pj\_ref, pj\_text FROM project WHERE pj\_seq=프로젝트번호;

쿼리문2. 현재 로그인한 사용자(id)가 현재 프로젝트에 참여한 멤버(id)인지 확인하여, 수정 버튼이 화면에 뜨게 함

SELECT us\_id FROM project a JOIN project\_member b ON a.pj\_seq = b.pj\_seq WHERE a.pj\_seq=프로젝트번호 AND b.us\_id = '현재 로그인한 아이디’;

쿼리문3. 프로젝트에 참여한 모든 참여자의 이름(name)을 불러옴

SELECT us\_name FROM project\_member a INNER JOIN user b ON a.us\_id = b.us\_id WHERE pj\_seq=현재프로젝트번호

쿼리문4. 현재 프로젝트의 카테고리 이름을 불러옴

SELECT ct\_name FROM project a INNER JOIN category b ON a.ct\_seq = b.ct\_seq WHERE pj\_seq=현재프로젝트번호

쿼리문5. 현재 프로젝트의 데이터 파일을 불러옴

SELECT pd\_file, pd\_note, pd\_date FROM project\_data WHERE pj\_seq=현재프로젝트번호

쿼리문6. 현재 프로젝트에 달린 댓글 정보를 가져옴

SELECT us\_img, pc\_text, a.us\_id, pc\_seq FROM project\_community a INNER JOIN user b ON a.us\_id = b.us\_id WHERE pj\_seq=현재프로젝트번호

(처리) -> 5/executeUpdate.jsp에서는 보여지는 화면 없이 코멘트 데이터를 삽입, 수정, 삭제만 수행

쿼리문1. 코멘트 내용을 삽입

INSERT INTO project\_community(pj\_seq, us\_id, pc\_text) VALUES (현재프로젝트번호", '현재 로그인한 아이디', '코멘트내용');

쿼리문2. 코멘트 내용을 수정

UPDATE project\_community SET pc\_text = '코멘트내용' WHERE pc\_seq = 현재프로젝트번호;

쿼리문3.

DELETE FROM project\_community WHERE pc\_seq = 현재프로젝트번호;