<웹프로그래밍>

웹사이트 최종 보고서

- 룸메이트 매칭 사이트 -

과목명 : 웹프로그래밍 (CSE2015-01)
담당 교수 : 송양의 교수님
제출일 : 2020년 12월 12일

통계학과 2018****** 이윤정
통계학과 2018****** 0 0 0

목차

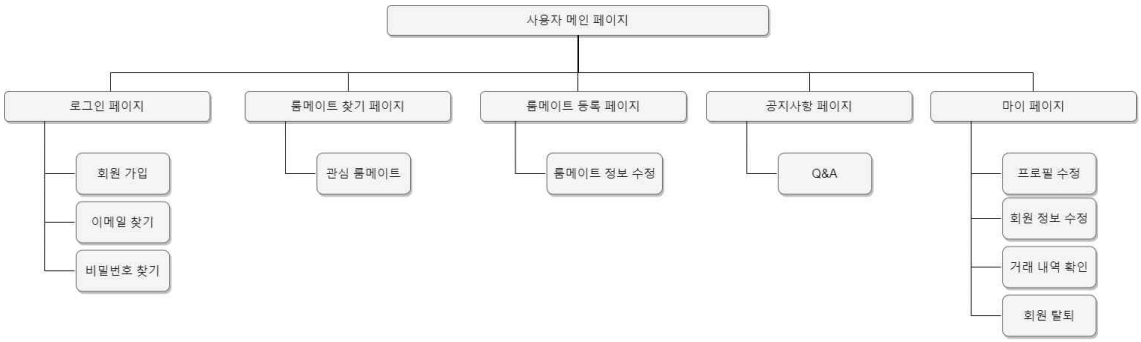
1. 개요 및 주제 소개
2. 사이트 맵
3. 전체 화면구성 및 상세 설명
4. DB
5. 자바 코드 설명
6. 외부 접속 주소

1. 개요 및 주제 소개

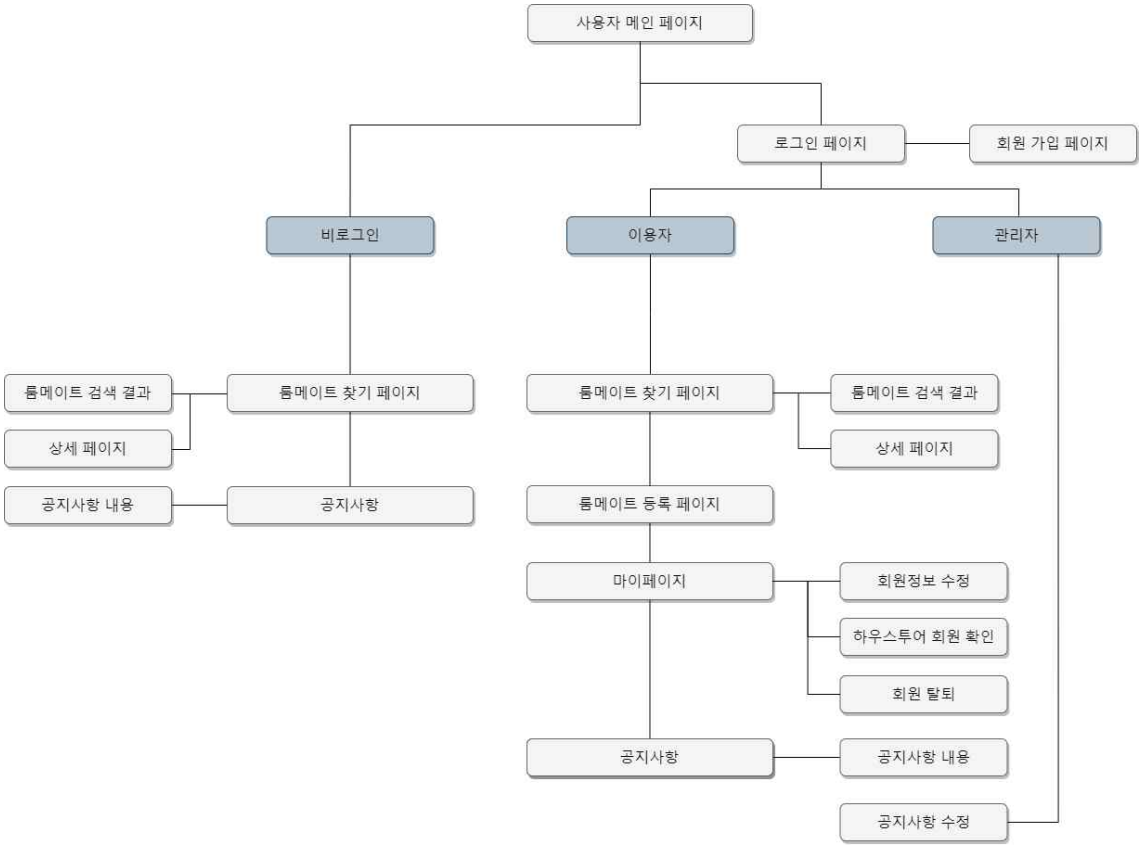
본 조가 진행할 팀 프로젝트 주제는 ‘룸메이트 매칭 사이트‘이다.
최근 1인 가구의 증가로 쉐어하우스 등 공유 주택이 많아지면서, 함께 동거를 할 ‘름메이트’ 구인 글 역시 증가하였다. 룸메이트는 주거 영역부터 서로의 생활 일부분까지 공유하기 때문에 위치나 금액 뿐만 아니라, 성향이나 생활 패턴 등 내적 요인까지 고려를 하는 것이 중요하다. 하지만, 기존에 존재하는 룸메이트 구인 사이트는 부동산 어플의 일부분이거나 위치나 금액만 고려하기 때문에 서로에게 맞는 룸메이트를 찾기 어려운 실정이다. 그러므로 본 조는 위치와 금액부터 성향, 생활 패턴까지 맞는 룸메이트를 찾을 수 있는 사이트를 기획하였다.

해당 웹사이트의 이용자는 실거주 상태로 함께 거주할 룸메이트를 찾는 ‘거주자’와 룸메이트와 거주 공간을 구하는 ‘입주 예정자’로 분류할 수 있다. 룸메이트 매칭 프로세스는 다음과 같다. 거주자와 입주 예정자 모두 웹사이트에 회원 가입 후 프로필 설정 과정을 거친다. 거주자는 프로필 설정 이후 ‘룸메이트 등록’ 페이지를 통해 현재 거주 중인 주거 공간에 대한 글을 등록한다. 입주 예정자는 ‘룸메이트 찾기’ 페이지에서 금액/입주 예정일 등의 정보를 입력하고, 웹사이트는 해당 선택지를 기반으로 필터링된 거주자의 프로필과 룸메이트 등록글을 노출한다. 입주 예정자는 노출된 정보를 기반으로 거주자와 컨택을 진행하며, 이때 거주자와 입주 예정자는 서로의 프로필을 확인할 수 있다. 스토리 보드에서는 자세하게 다루지는 않았지만, 컨택을 하는 방식은 다음과 같다. 입주자가 룸메이트 찾기 기능을 통하여 매물을 확인한 후 집을 직접 보고싶다고 신청(이하, 하우스 투어 신청)을 하게 되면 거주자가 이를 확인하고 연락을 취하는 방식으로 진행된다. 거주자가 컨택이 완료된 후 거주자와 입주 예정자 모두 동의하는 경우 매칭이 성사된다.
해당 사이트명은 ‘DRD’로 결정하였고 이는 ‘다 룸메 됨’의 뜻을 가지고 있다.

2. 사이트맵



[스토리 보드에 사용된 사이트맵]

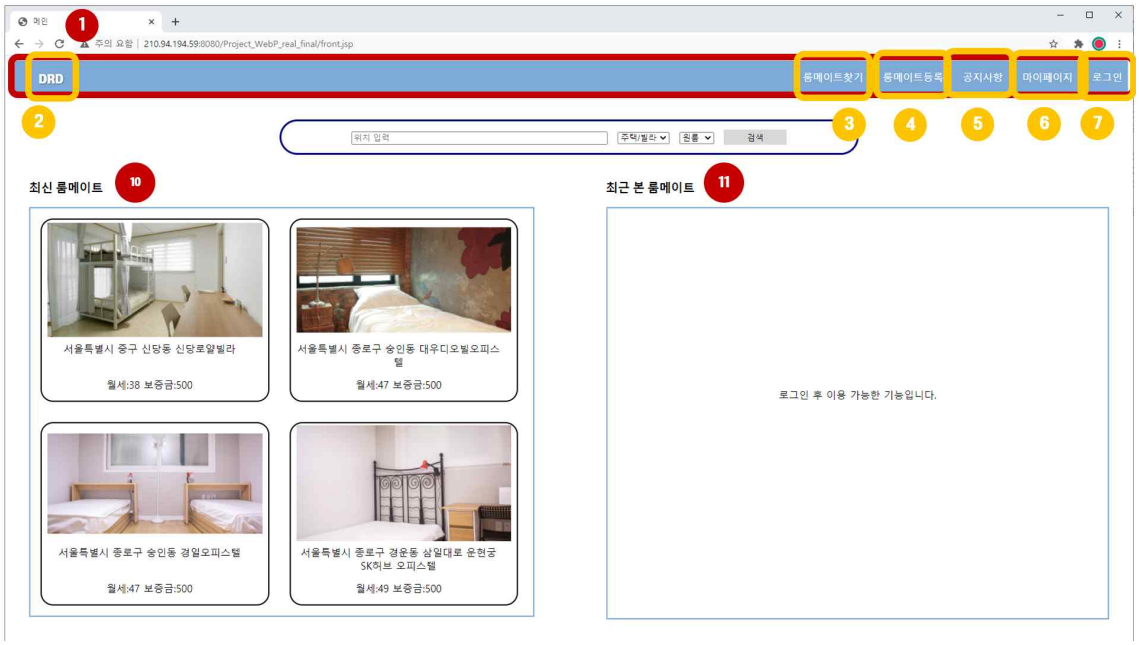


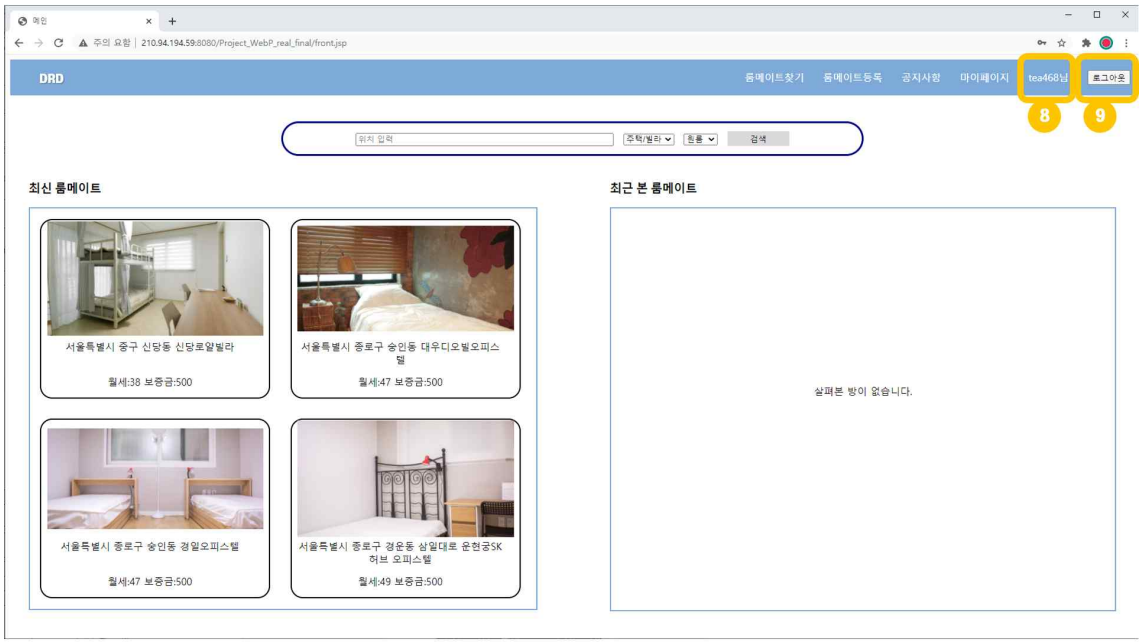
[최종 보고서에 사용된 사이트맵]

최종적으로 기존에 계획했던 사이트 맵 중 로그인페이지와 회원가입 페이지, 룸메이트 찾기 페이지, 룸메이트 등록 페이지, 공지사항 페이지, 마이페이지, 프로필 수정 페이지, 회원 정보 수정 페이지, 회원 탈퇴 페이지를 구현하였다. 그리고 실 거주자를 같이 구하는 사이트의 특성상 1명의 가입자는 오로지 룸메이트를 찾거나, 1개의 집만 등록할 수 있게 하였기 때문에 스토리 보드에서의 거래 내역 확인 페이지를 자신의 집에 컨택이 들어온 사람을 확인할 수 있는 하우스 투어 회원 확인 페이지로 특성을 바꾸어 구현하였다. 또한, 비로그인과 로그인한 이용자를 구분하여 접근할 수 있는 페이지에 차이를 두었다. 비로그인의 경우 룸메이트 찾기 관련페이지와 공지사항, 공지사항 내용 확인 페이지까지만 접근 가능하고, 이용자의 경우 공지사항 수정 페이지를 제외한 모든 페이지에 접근 할 수 있다. 관리자는 이용자가 접근 가능한 페이지에 추가로 공지사항 수정 페이지를 접근하여 공지를 수정하거나 삭제할 수 있다.

3. 전체 화면구성 및 상세 설명

Ⅰ. 사용자 메인 페이지

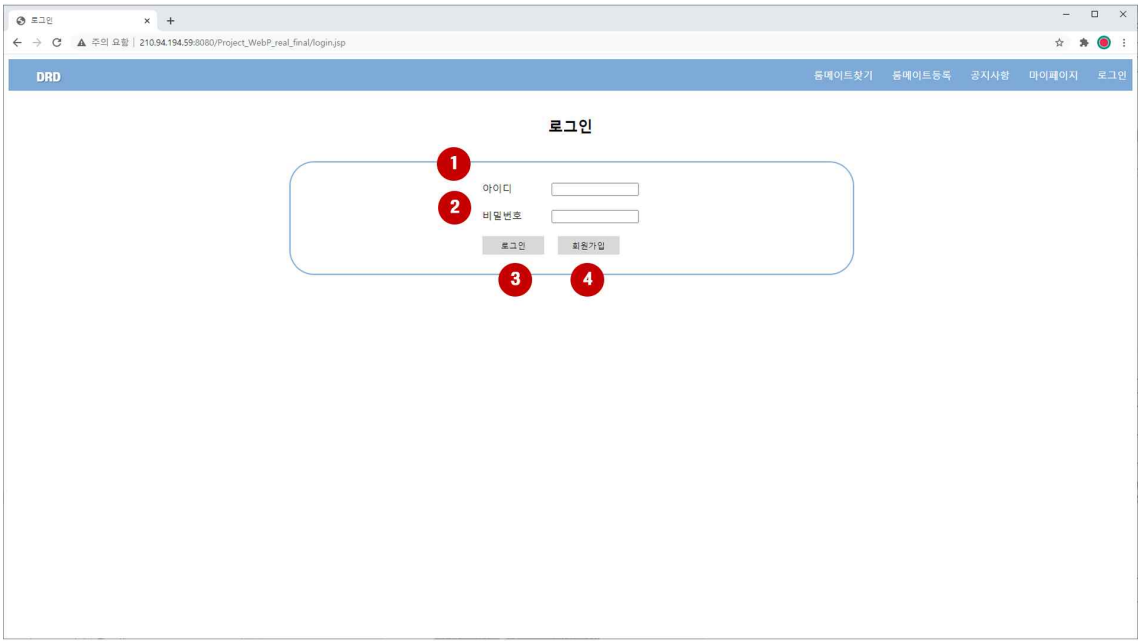




front.jsp	
번호	**설명**
1	홈페이지 배너 - session을 사용하여 로그인 유무의 배너 디자인에 차이를 두었다. 로고, 룸메이트찾기, 룸메이트 등록, 공지사항, 마이페이지는 동일하지만 로그인 버튼이 없어지고 아이디와 로그아웃 버튼이 나타난다.
2	메인페이지 이동

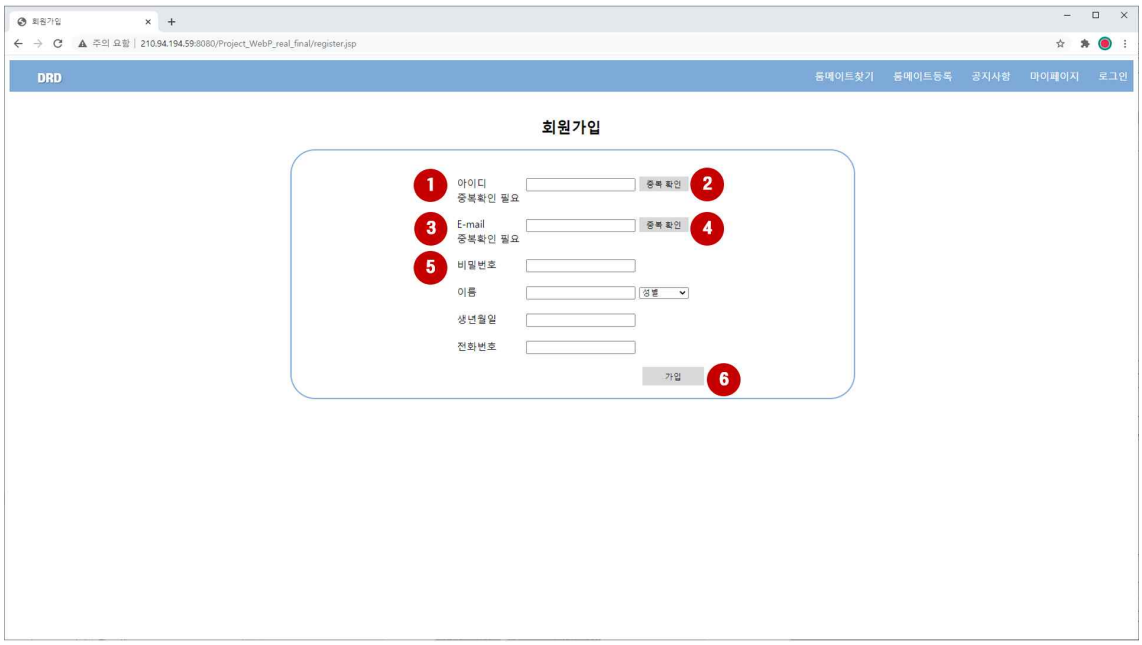
	- a 태그와 image 태그를 사용하여 로고를 누르면 front.jsp로 이동한다.
3	룸메이트 찾기 페이지 이동 - a 태그를 이용하여 find.jsp로 이동한다.
4	로그인이 된 경우에는 룸메이트 등록 페이지로, 로그인이 되어있지 않으면 로그인 페이지로 이동 - 로그인 전 session에서는 login.jsp로 이동하고 로그인 후에는 enrollment.jsp로 이동한다.
5	공지사항 이동 - a 태그를 이용하여 notice_list.jsp로 이동한다.
6	로그인이 된 경우에는 마이페이지로, 로그인이 되어있지 않으면 로그인 페이지로 이동 - 로그인 전 session에서는 login.jsp로 이동하고 로그인 후에는 mypage.jsp로 이동한다.
7	로그인 페이지 이동 - a 태그를 이용하여 login.jsp로 이동한다.
8	아이디 - session.getAtribute("id")를 이용하여 로그인 한 회원의 아이디를 보여준다
9	로그아웃 - logout.jsp로 이동하여 session을 초기화 한 후 front.jsp(메인페이지)로 이동한다.
10	최신 룸메이트 - 최근에 올라온 거주지 4개를 띄워준다. - 기존 스토리보드에는 4개보다 많은 개수를 띄우려고 계획하였으나, 구현 과정에서 4개가 적절하다고 판단하여 4개를 띄워준다.
11	최근 본 룸메이트 - 최근에 본 거주지 1개를 띄워준다. - 기존 스토리보드에는 여러개를 띄워주나, 구현 과정에서 어려움을 겪어 1개만 띄우기로 구현하였다.
특이사항	로그인 유무에 따라 접근할 수 있는 페이지를 다르게 설정하였다.

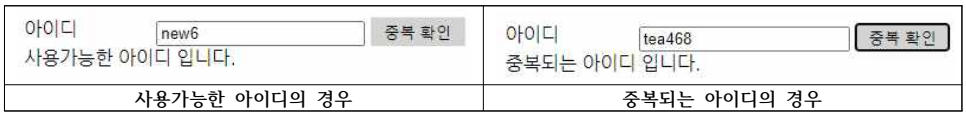
Ⅱ. 로그인 페이지

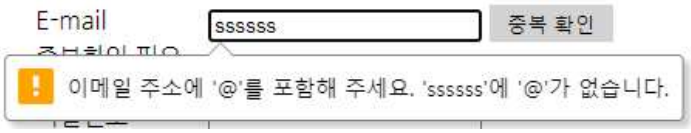


login.jsp	
번호	**설명**
1	아이디 입력 - input 태그를 이용하여 user_id에 텍스트 형식으로 받아온다
2	비밀번호 입력 - input 태그를 이용하여 user_pw에 password 형식으로 받아온다.
3	login_serv로 전달 - input 태그를 이용하여 저장된 user_id와 user_pw를 login_serv로 submit한다. 이때 db와 비교하여 로그인에 성공하면 front.jsp로 이동하고, 로그인에 실패하면 다음과 같은 팝업창이 뜨게 된다. - 확인 버튼을 누르면 다시 login.jsp(로그인 페이지)로 이동하게 된다.
4	회원가입 페이지로 이동 -button 태그의 onclick 기능을 이용하여 register.jsp(회원가입 페이지)로 이동한다.
특이사항	

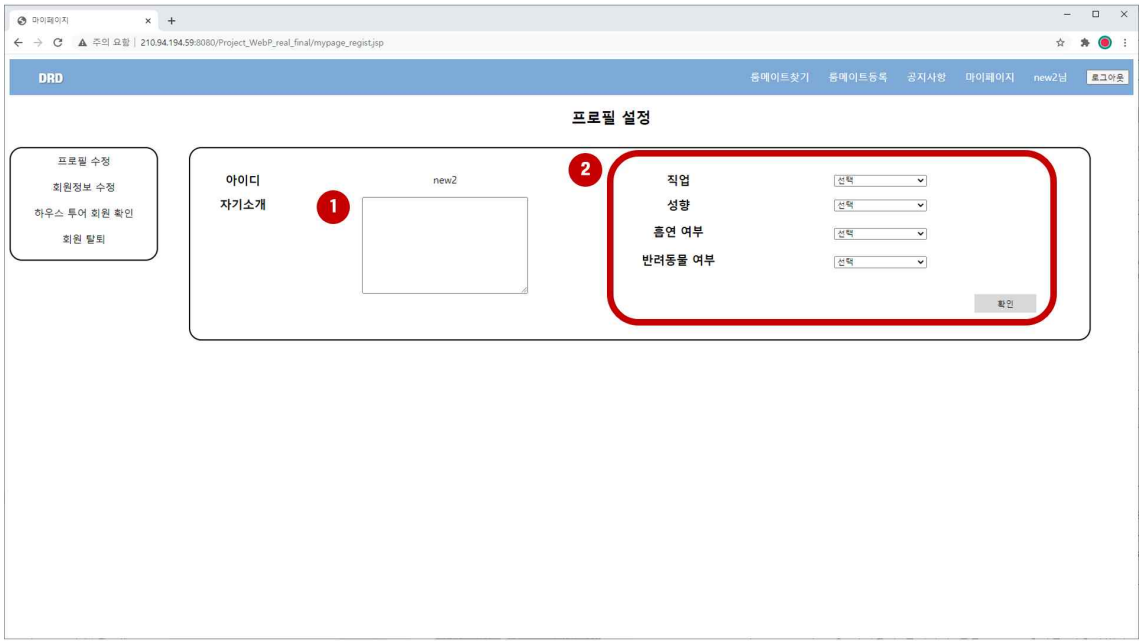
3. 회원 가입



register.jsp	
번호	**설명**
1	아이디 입력 - input 태그를 이용하여 id에 텍스트 형식으로 받아온다
2	아이디 중복확인 - register.js의 check1()함수를 사용하여 입력된 문자열의 길이에 따라 작동하는 방법이 다르다. • 4 미만인 경우 다음과 같은 팝업창이 뜬다. 이때 확인 버튼을 누르면 경고창이 닫힌다. - 1에서 입력받은 id값을 팝업창을 통하여 joinID.jsp로 전송한다. joinID.jsp에서는 전송받은 id값으로 joinService.idcheck 함수를 이용하여 중복 여부를 판단한다. 중복 여부에 따라 아이디 밑에 있던 '중복확인 필요' 문구가 변경된다.  **사용가능한 아이디의 경우** / **중복되는 아이디의 경우**
3	이메일 입력 - input 태그를 이용하여 e_mail에 이메일 형식으로 받아온다. - 이메일 type을 사용하기 때문에, 이메일 형식이 아니면 데이터가 전송되지 않고 경고 팝업이 뜬다.

	
4	이메일 중복확인 - register.js의 check2()함수를 사용하여 1에서 입력받은 email값을 joinE_mail.jsp로 전송한다. joinE_mail.jsp에서는 전송받은 email값으로 joinService.e_mail_check 함수를 이용하여 중복 여부를 판단한다. 중복 여부에 따라 이메일 밑에 있던 '중복확인 필요'문구가 변경된다. 사용가능한 이메일의 경우 / 중복되는 이메일의 경우
5	비밀번호 입력 - input 태그를 이용하여 passwd에 password 형식으로 받아온다.
6	register_serv로 전달 - 앞서 입력한 내용을 register_serv로 전달하고 회원 가입 완료 팝업이 나타난다. - 확인 버튼을 누르면 mypage_regist.jsp(초기 프로필 설정 페이지)로 이동한다.
특이사항	아이디와 비밀번호 중복 체크

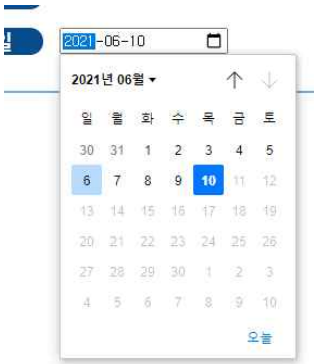
3. 초기 프로필 설정



mypage_regist.jsp	
번호	**설명**
1	자기소개 입력 - textarea 태그를 사용해서 간단한 인사말, 자기 소개에 대한 내용을 mypage_regist 서블릿으로 전달한다.
2	직업, 성향, 등 기타정보사항 입력 - select 태그를 사용해 직접 입력이 아닌 주어진 보기내에서 선택할 수 있는 option을 주어 정보입력에 부담감을 덜어줌. 정보를 다 선택한 후 확인 버튼을 누르면 mypage_regist서블릿으로 전달한다.
특이사항	해당 페이지는 mypage와 디자인이 같지만 회원가입직후에 뜨는 페이지로 mypage와 하는 역할이 다르다. mypage_regist를 통해 table에 새롭게 등록을 한 후 mypage로는 데이터 수정을 진행한다.

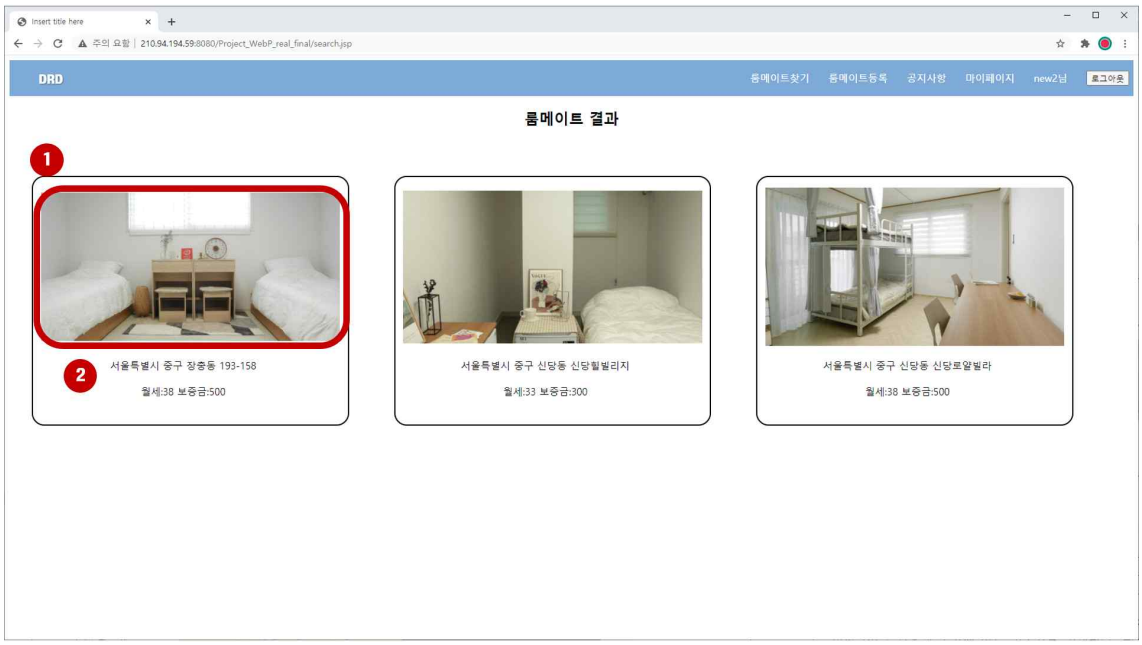
4. 룸메이트 찾기

find.jsp	
번호	**설명**
1	원하는 지역을 입력하여 검색 조건에 추가할 수 있다. -기존 스토리보드에는 시, 동 단위를 직접 설정한다고 계획했으나 구글 API를 사용하여 더 자세한 주소를 설정할 수 있게 되었다. - 구글 API는 전 세계 지역을 모두 검색할 수 있으나, componentRestrictions 옵션을 통해 한국 지역만 검색할 수 있도록 제한을 두었다.

2	월세의 범위 선택 - 좌우로 움직일 수 있는 바를 이용하여, 원하는 월세의 최대 범위를 조정할 수 있다. - 선택 범위의 게이지 바 색상이 기존 스토리보드에는 붉은 색으로 변경한다고 계획되었으나, 이는 전체적인 웹페이지의 테마색과 어울리도록 하기 위하여 청색으로 바꾸었다. - 사용자의 편의성을 위해 슬라이드 바를 사용하기 위해, input type=range로 입력 받았다. 이때, 월세는 최대 200만원까지 입력할 수 있도록 input의 max 옵션으로 설정하였고, 슬라이드 바 우측에 사용자가 입력한 월세 금액을 표시하였다.
3	성별 선택 - 기존 스토리 보드에서는 성별 선택 후 노란색 버튼으로 변경된다고 계획되었으나, 구현상 어려움이 존재하여 일반 select 형식을 사용하였다.
4	건물 유형 선택 - 기존 스토리 보드에서는 다중 선택이 가능하였으나, 데이터베이스 테이블에서 건물 유형을 단일로 지정하였고 이에 검색 기능 구현 시 너무 복잡해질 것을 우려하여 일반 radio 버튼 형식을 사용하였다.
5	건물 구조 선택 - 기존 스토리 보드에서는 다중 선택이 가능하였으나, 데이터베이스 테이블에서 건물 유형을 단일로 지정하였고 이에 검색 기능 구현 시 너무 복잡해질 것을 우려하여 일반 radio 버튼 형식을 사용하였다.
6	입주 예정일을 선택 - date형식을 이용하여 날짜를 지정할 수 있음 -다음 그림과 같이 등록일 기준 6개월 이내까지만 선택 가능하도록 구현  - 자바 스크립트를 이용하여 새로운 Date 생성자를 선언하였다. 이때, 해당 생성자에는 자동으로 당일 날짜가 입력된다. 하지만, Date 생성자로 불러온 날짜 값과, date type은 입력받는 양식이 다르므로 변환과정을 거친다. 변환과정을 거친 오늘 날짜는 초기값으로 설정함과 동시에 사용자가 오늘 이전의 날짜를 선택할 수 없도록 jquery의 attr 명령어를 통해 input의 min, value 옵션으로 설정하였다. 또한, 오늘 날짜로부터 180일을 더해 6개월 후의 날짜를 구한 다음 변환 과정을 거쳐 input의 max 옵션으로 설정하였다.

7	입력한 모든 조건들을 search.jsp로 전송 - form tag를 이용해 의 모든 조건들을 post방식으로 search.jsp에 전달한다. 전달 과정 중에 onsubmit option에서 return check()라는 함수를 실행해 지역입력창에서 구글api에서 제공해주는 지역명을 사용하게 되면 해당 경도와 위도를 받을 수 있으며 이를 이용해 기존 enroll table에 저장되어있는 위도 경도값을 이용하여 근처 2km이내에 있는 지역을 검색할 수 있다.
특이사항	구글 API를 사용한 지역명 입력, 경도 위도 계산, 구글 캘린더 사용하여 특정 기간 선택 구현

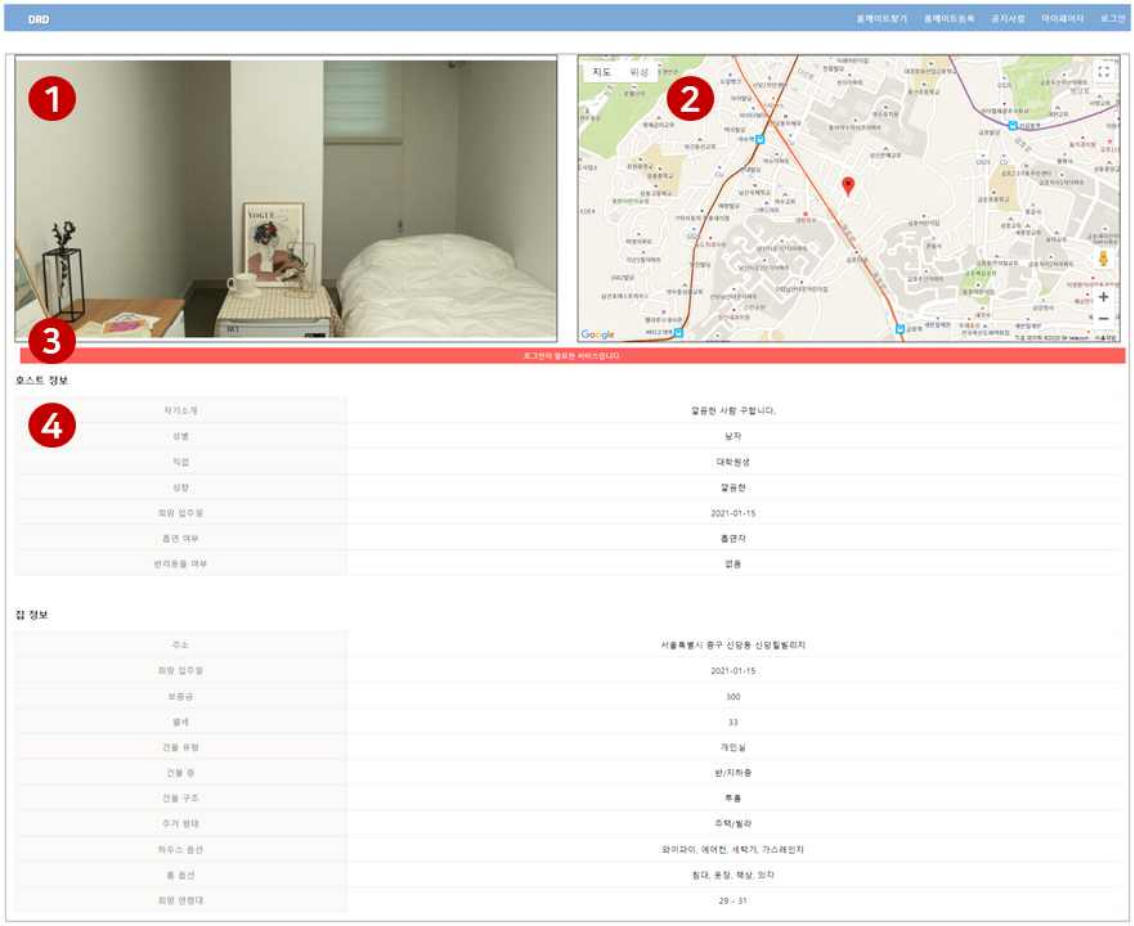
5. 룸메이트 결과



search.jsp	
번호	**설명**
-	전체적인 코드 설명 - search.jsp에서는 find.jsp와 front.jsp에서 넘어오는 지역명, 경도, 위도, 건물 유형을 받아 DB로 접속해 조건에 맞는 거주지를 검색한다. 현재 조건을 설정하면 위도 경도를 통한 가까운 건물(2km이내)에 건물 유형이 맞는 결과들만 화면에 보여준다. 결과에 따라 div창에 해당 거주지에 대한 정보를 주소, 월세, 보증금, 및 사진을 담아두었으며 동적으로 할당해 두었기 때문에 결과 개수에 따라 div창의 개수가 결정된다.
1	해당 룸의 디테일 페이지로 이동 - 해당 룸의 사진은 룸 등록시 업로드 폴더에서 업로드한 사용자의 id를 받아 폴더를 생성한 후 각 폴더에 image.jpg로 저장된다. 이렇게 하면 사용자들간의 image가 뒤섞일 일이 없다.

	- a태그와 img 태그를 이용하여 해당 룸의 detailpage.jsp로 이동하며 등록한 사용자의 id를 get방식으로 전송하여 해당 유저의 보다 상세한 정보를 확인할 수 있다.
2	주소와 금액 정보 나타냄 - db의 enroll table에서 받아온 region(지역), monthly(월세), deposit(보증금)을 보여준다.
특이사항	DB에서 결과를 동적으로 받아와 결과 화면의 div창의 개수를 자유롭게 조절 가능

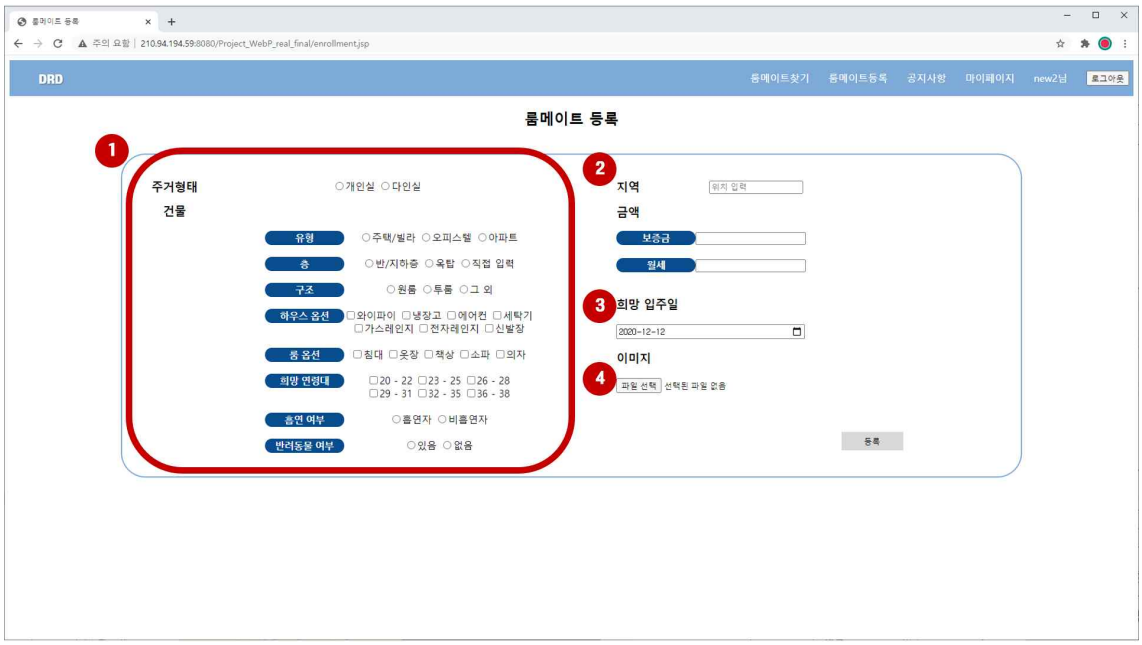
6. 룸메이트 상세 페이지



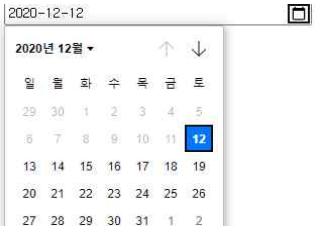
detailpage.jsp	
번호	**설명**
-	전체적인 코드 설명 - detailpage.jsp는 front.jsp나 search.jsp의 거주지 사진을 클릭해야 들어올 수 있다. 이미지 클릭과 동시에 해당 거주지의 등록자의 id를 get방식으로 전송받아 DB에서 해당 등록정보를 확인하여 정보를 보여준다. 구글api의 지도를 통해 근처 역세권이나 주요 시설등을 파악할 수 있어 보다 편리함을 제공한다. 집 둘러보기신청 버튼은 로그인이 되어있는 상태에서만 사용이 가능하며 본인의 집정보를 본다거나 로그인이안되있는 상태는 사용이 불가능하다.
1	해당 룸의 사진

	- upload폴더의 거주자 id폴더에 있는 image.jpg를 보여준다. 현재는 1장밖에 등록이 안되있지만 추후 사진의 개수에 따라 여러장의 화면을 띄울 수 있는 UI를 계획중이다.
2	해당 룸의 위치를 구글맵에 띄워줌 - 전달받은 id값을 통해 DB에 연결하여 enroll table에서 lat, lng 값을 받은 후 initMap()함수에 넣어주면 해당 위치의 지도를 google.maps.Map을 통해 받아와 화면에 표시해줌.
3	집 둘러보기 신청 내가 등록한 집입니다. 로그인이 필요한 서비스입니다. - 해당 기능은 로그인을 한 후 사용이 가능하며 로그인이 안되어있거나 내가 등록한 거주지일 경우 위 그림과 같이 나오면서 클릭이 되지 않는다. 집 둘러보기를 신청하면 사용자의 id는 matching table에 거주지 등록자의 id와 같이 저장이 되며 거주지 등록자가 하우스 투어 회원 확인을 하게 되면 신청자의 정보를 확인할 수 있다.
4	호스트 정보와 집 정보 - 전달받은 id값을 통해 DB에 연결, profile table과 enroll table에서 거주자와 룸에 관련된 정보를 가져와 나타낸다.
특이사항	구글 API를 통한 거주지의 지도 표시, 집 둘러보기 신청

7. 룸메이트 등록



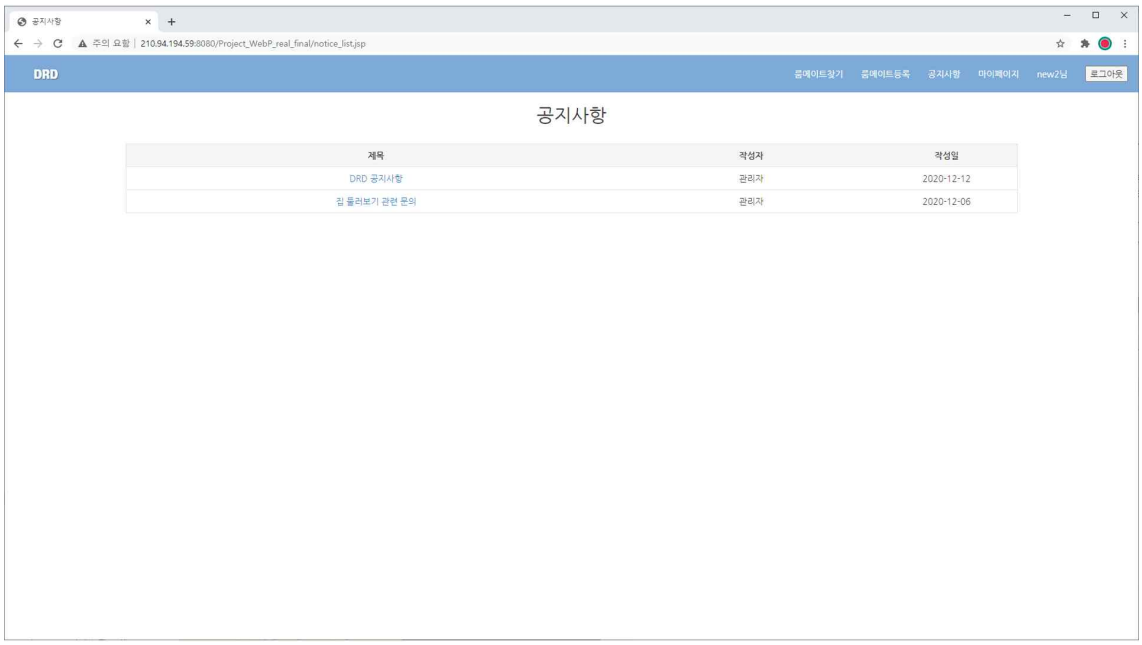
enrollment.jsp	
번호	**설명**
-	전체적인 코드 설명 거주지 등록에는 많인 정보가 필요하며 거주지 유형, 층, 구조 등 하나의 선택 옵션은 Radio button을 사용, 다중 선택 옵션은 check box tag를 사용해 중복 선택을 허용하였

	으며 지역 입력의 경우 구글 api를 통해 정확한 위치를 식별할 수 있도록 도움을 주었다.  날짜선택도 마찬가지로 현재를 기준으로 그보다 이전 날짜는 말이 안되기 때문에 선택을 못하도록 하였으며 희망 입주일의 기한 역시 6개월로 제한하였다. input tag의 file type을 사용해 파일을 첨부받을 수 있게 만들었으며 enctype="multipart/form-data"옵션을 통해 파일이 전송될 수 있게 설정했다.
1	거주지와 관련한 정보 입력 - 주거 형태, 건물유형, 층, 구조, 흡연여부, 반려동물 여부는 모두 radio버튼을 사용하여 하나만 받아오게 하였고, 하우스옵션과 룸옵션, 그리고 희망 연령대는 체크박스를 이용하여 중복 설정이 가능하도록 하였다.
2	지역 선택 - 구글 API를 이용하여 등록하고자 하는 주소를 검색하여 선택하고 위도와 경도를 받아 각각 lat, lng 변수에 넣은 후 enrollment_serv에 전송한다.
3	희망 입주일을 선택 - date형식을 이용하여 날짜를 지정할 수 있음 -다음 그림과 같이 등록일 기준 6개월 이내까지만 선택 가능하도록 구현
4	이미지 선택 - input 태그를 이용한 파일선택 버튼을 눌러 이미지 파일을 업로드한다. 이때 파일 선택창에서 이미지 확장자만 가진 파일만 뜨게 accept="image/*"을 써서 제한을 걸어둠.

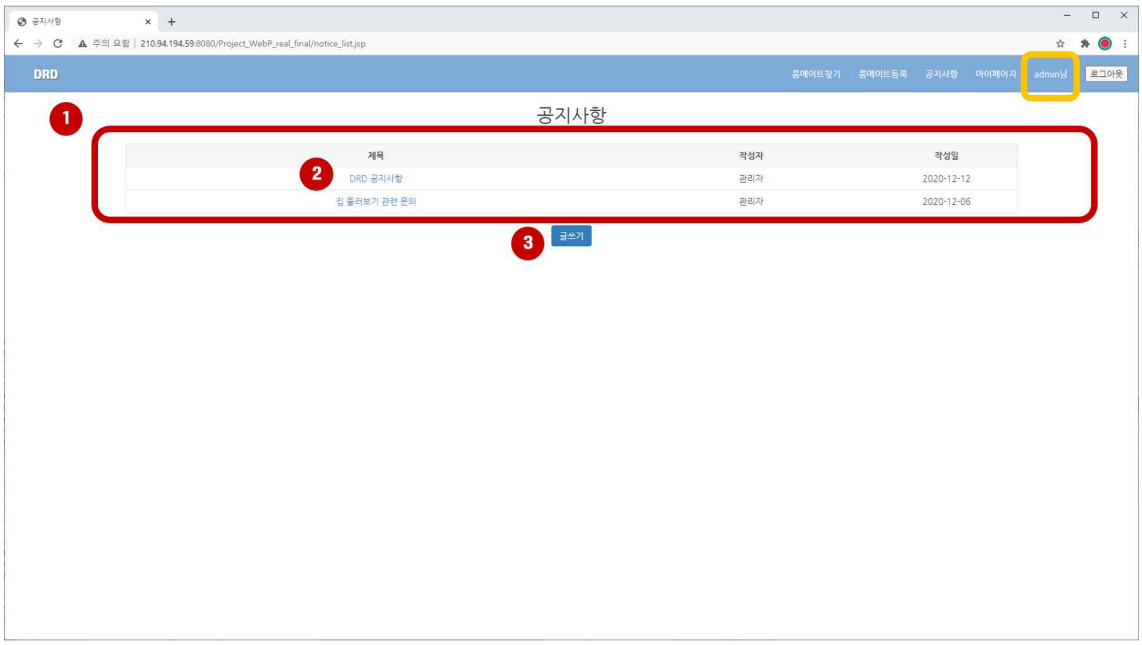
특이사항	구글 API를 통한 검색어 입력, 달력의 날짜 제한

8. 공지사항

(1) 관리자가 아닐 때

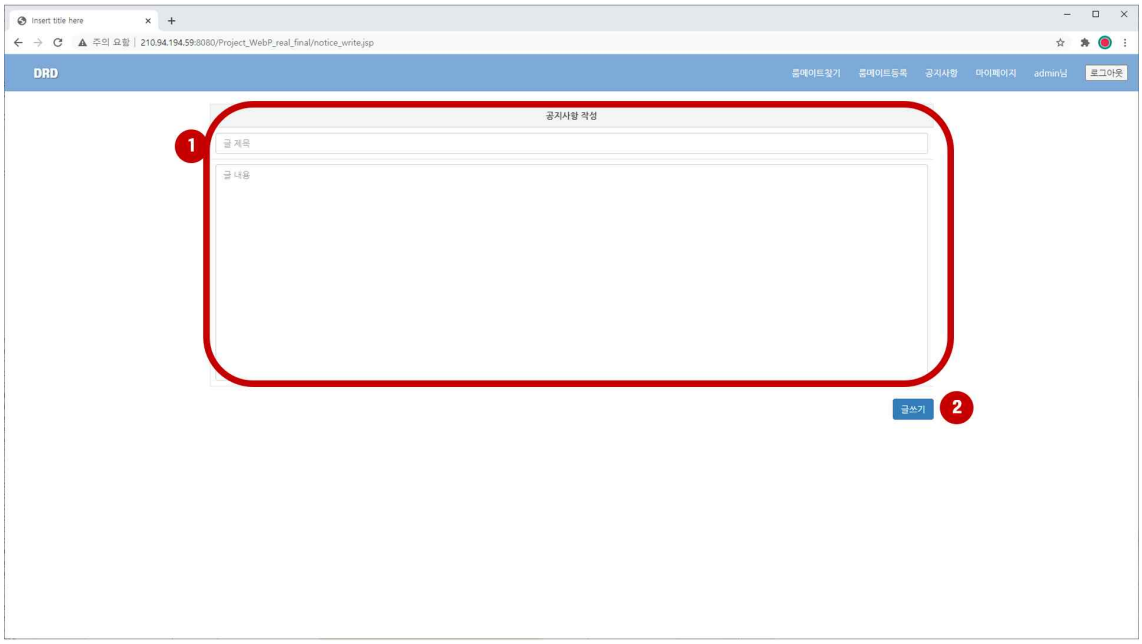


(2) 관리자가 로그인 했을 때



notice_list.jsp	
번호	**설명**
1	공지사항 목록 - sql을 사용하여 notice table에서 id를 기준으로 오름차순 정렬을 하여 title(제목)과 date(작성 날짜)를 가져와 목록을 보여준다.
2	공지사항 내용 확인 - a태그를 이용하여 제목을 누르면 notice_read.jsp로 이동하여 공지사항 내용을 확인할 수 있다.
3	공지 작성 - 공지 작성 버튼은 관리자(admin)에게만 보여진다. - a태그를 이용하여 글쓰기 버튼을 누르면 notice_write.jsp로 이동한다. - bootstrap.css를 이용하여 버튼 모양을 구현하였다.
특이사항	bootstrap.css 사용하여 기존보다 화려한 게시판을 만드는데 성공했다

9. 공지사항 작성

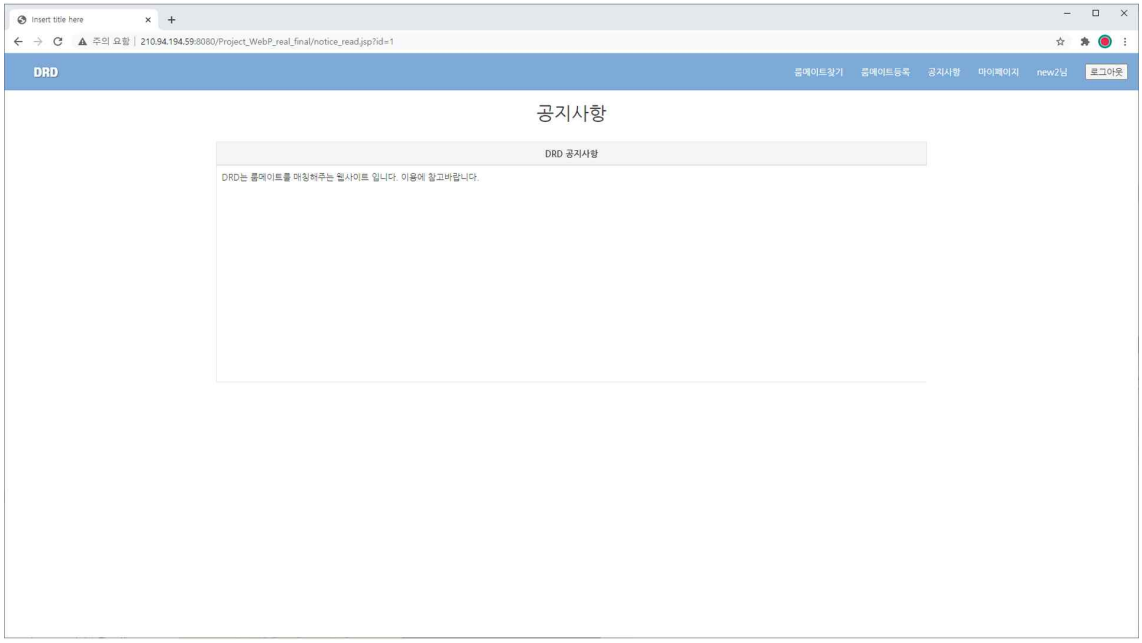


notice_write.jsp	
번호	**설명**
1	공지의 제목과 내용 입력 -input 태그와 textarea 태그를 이용하여 각각 title과 content로 입력받음
2	공지 작성 -input 태그를 이용하여 앞에서 받은 내용을 writeAction.jsp로 전송한다. -bootstrap.css를 사용하여 버튼 모양을 구현하였다.

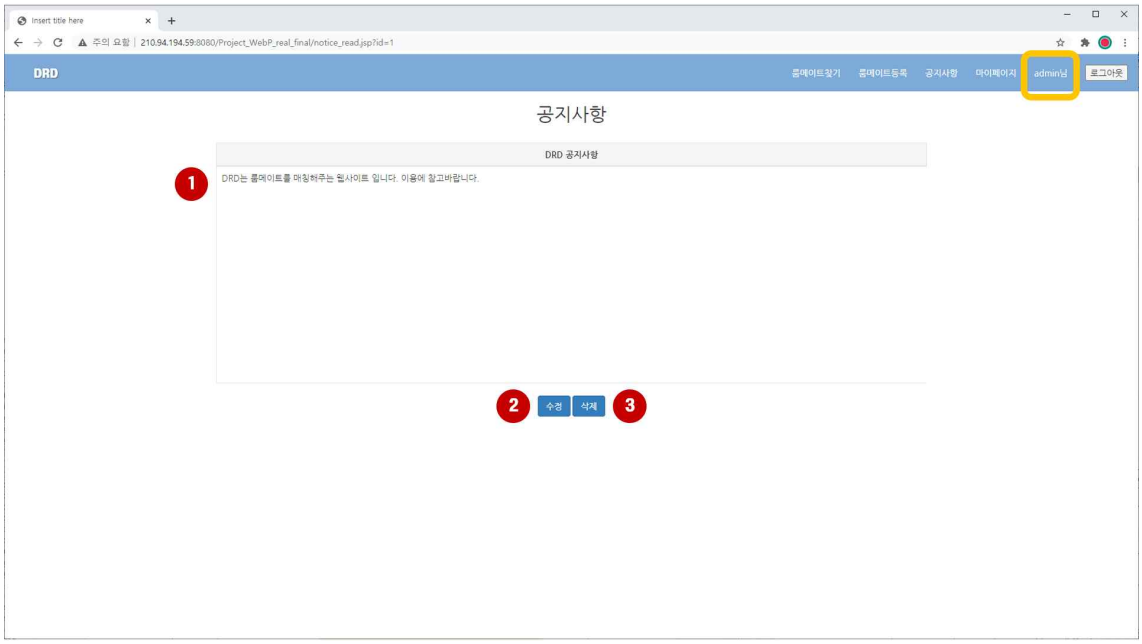
특이사항	bootstrap.css 사용

10. 공지사항 내용

(1) 관리자가 아닐 때



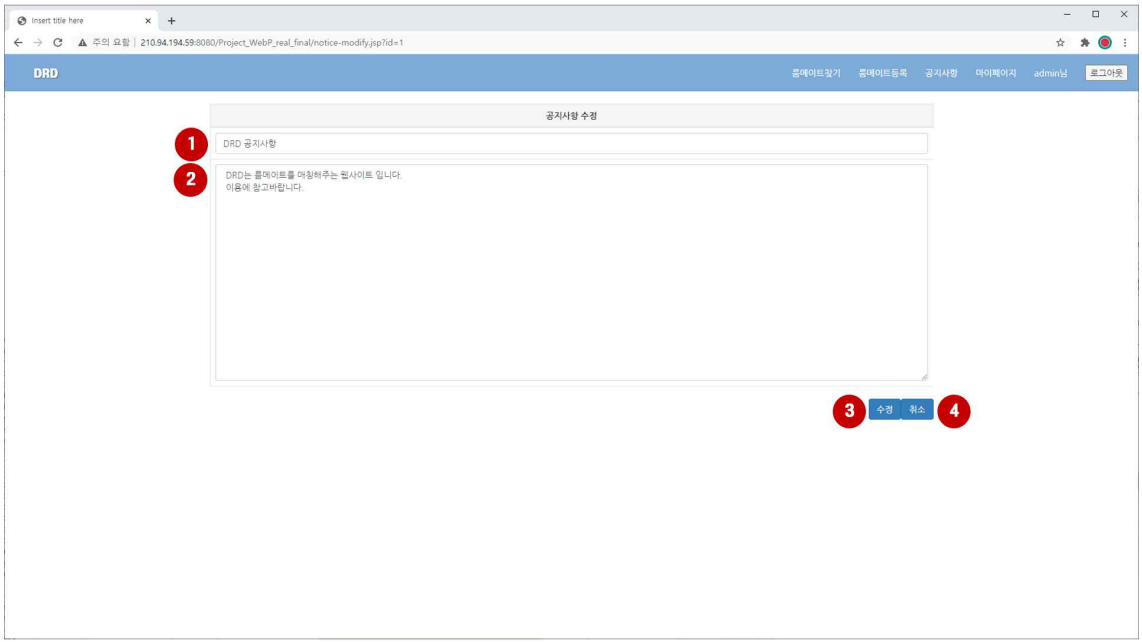
(2) 관리자가 로그인 했을 때



notice_list.jsp	
번호	설명

1	공지사항 내용 - sql을 사용하여 notice table에서 id를 기준으로 오름차순 정렬을 하여 title(제목)과 content(내용)을 가져와 목록을 보여준다.
2	공지사항을 수정할 수 있다. - 공지사항 수정 버튼은 관리자에게만 보인다. - a태그를 이용하여 notice-modify.jsp로 이동한다. - bootstrap.css를 이용하여 버튼 모양을 구현하였다.
3	공지 삭제 - 공지사항 삭제 버튼은 관리자에게만 보인다. - a태그를 이용하여 글쓰기 버튼을 누르면 notice-delete-db.jsp로 이동한다. - bootstrap.css를 이용하여 버튼 모양을 구현하였다.
특이사항	bootstrap.css 사용

11. 공지사항 수정



notice-modify.jsp	
번호	**설명**
1	제목 수정 - 원래 title(제목)을 notice 테이블에서 받아와 input 태그의 value값으로 두었다. - 그러므로 기존 제목이 입력 칸에 나타나고 수정하지 않는다면 원래 제목 그대로 유지된다.
2	내용 수정 - 원래 content(내용)을 notice 테이블에서 받아와 textarea 태그의 내용으로 두었다. - 그러므로 기존 내용이 텍스트칸에 나타나고 수정하지 않는다면 원래 내용 그대로 유지된다.
3	제목과 내용의 정보를 notice-modify-db.jsp로 전송

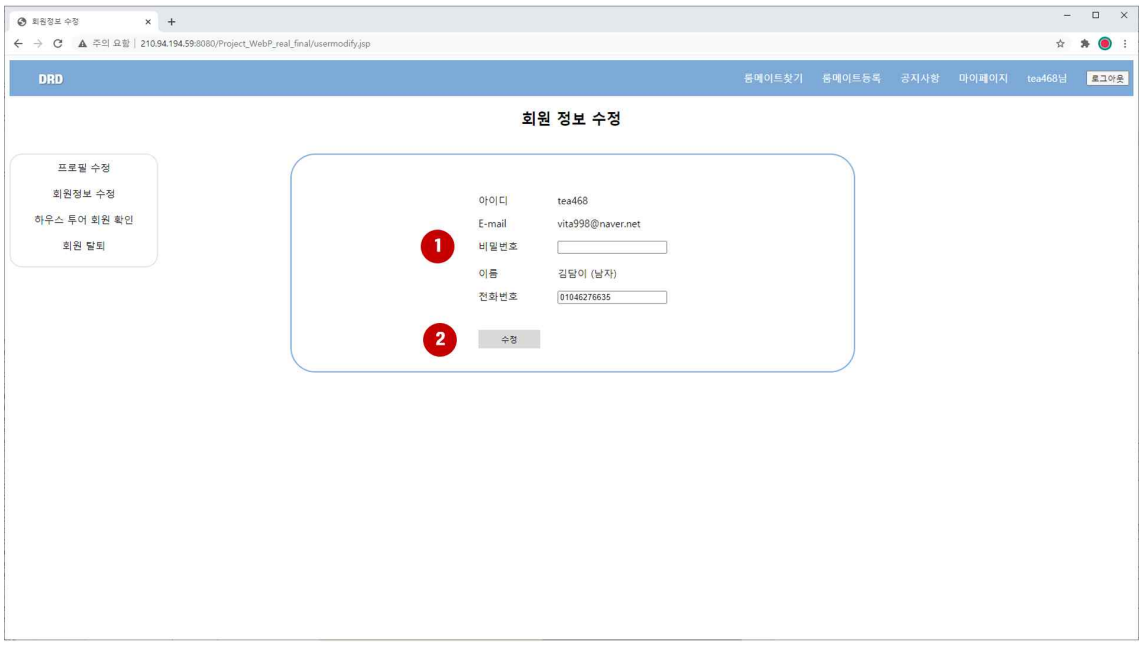
	- post방식 이용
4	수정 취소 - a 태그를 이용하여 수정을 멈추고 notice_list.jsp(공지사항)로 이동한다. - bootstrap.css를 사용하여 버튼 모양을 구현하였다.
특이사항	bootstrap.css 사용

12. 마이페이지

mypage.jsp	
번호	**설명**
-	전체적인 코드 설명 mypage_regist랑 같은 화면의 jsp파일이다. mypage_regist는 profile table에 내용을 등록하기 위해 존재한다면 해당 페이지는 profile table의 내용을 수정하기 위해서 존재한다.
1	프로필 수정 - a태그를 이용하여 mypage.jsp로 이동한다. 마이페이지 내의 메뉴를 이용한 후 다시 프로필 수정(마이페이지)로 이동할 때 이용한다.
2	회원정보 수정 - a태그를 이용하여 usermodify.jsp(회원정보수정 페이지)로 이동한다.
3	하우트 투어 회원 확인 - a태그를 이용하여 housetouruser.jsp(하우스투어회원확인페이지)로 이동한다.
4	회원 탈퇴 - a태그를 이용하여 withdraw.jsp(회원 탈퇴 페이지)로 이동한다.
5	아이디를 제외한 프로필을 변경 가능 - 기존에 저장되어있었던 프로필을 테이블에서 받아와 각 textarea와 select 형식의

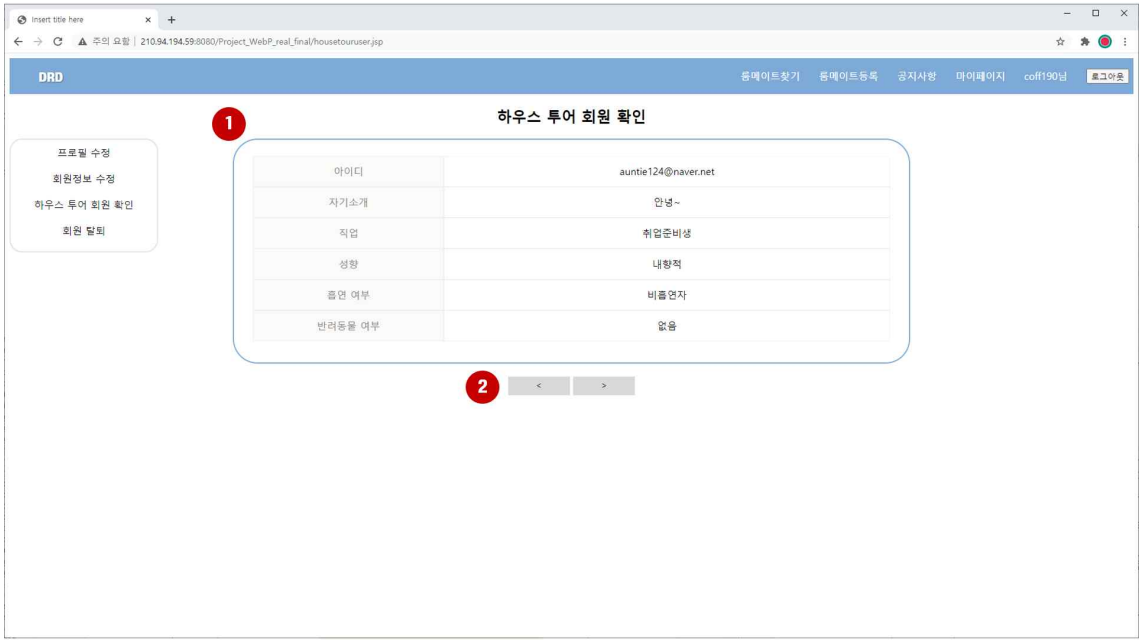
	value값으로 두었다. 만약 수정하지 않는다면 원래내용 그대로 유지된다. - input 태그를 이용하여 앞에서 수정한 내용을 mypage.java로 전달한다. - input태그의 히든 타입을 이용하여 id도 함께 전달하여 프로필 수정을 용이하게 한다.

13. 회원 정보 수정



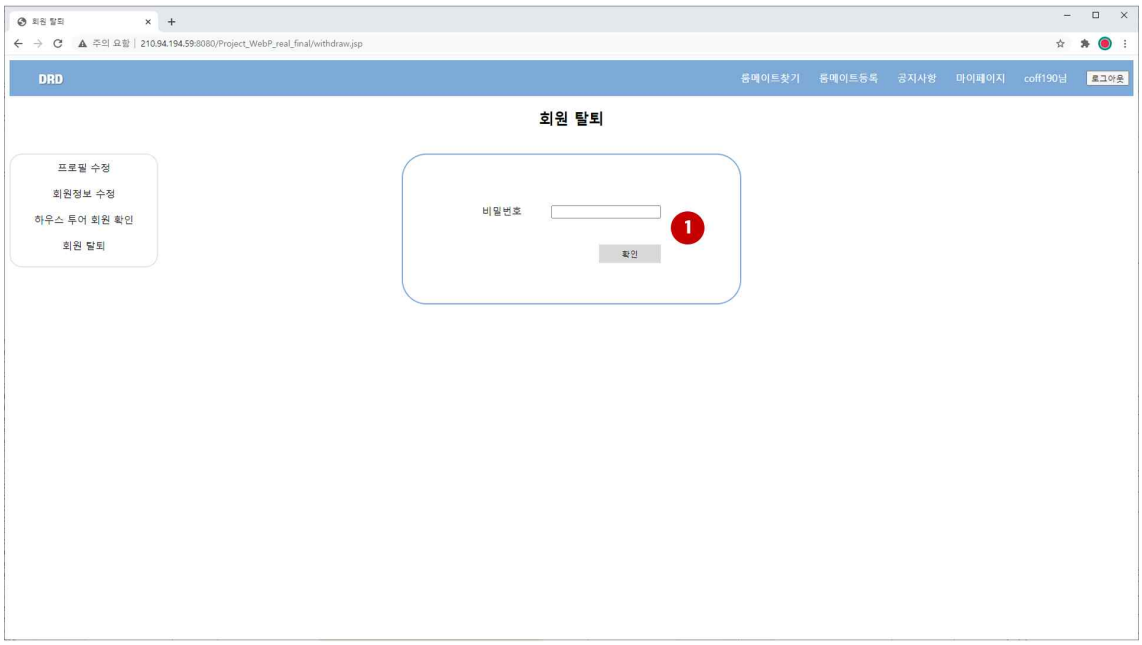
usermodify.jsp	
번호	**설명**
1	비밀번호와 전화번호 수정 가능 - 둘다 input태그를 사용하지만 비밀번호는 password 타입을 사용하여 값이 보이지 않고 전화번호는 text형식이기 때문에 입력한 그대로 보인다.
2	정보 수정 - input태그를 사용하여 user_modify.java로 앞에서 입력한 값을 전달
특이사항	회원정보 수정탭을 누르면 해당 사용자의 id가 session에서 제공되어 id를 기반으로 jsp 내부에서 sql query문에서 사용자의 정보를 검색한 후 아이디 e-mail, 이름 전화번호에 대한 정보를 미리 입력해 둔다.

14. 하우스 투어 회원 확인



housetouruser.jsp	
번호	**설명**
1	하우스 투어를 원하는 회원 확인 가능 - 룸메이트 찾기에서 원하는 거주지에 집 둘러보기 신청을 하게되면 다음과 같이 거주자의 하우스투어 회원 확인 페이지에 신청자의 프로필이 보여지게 된다.
2	여러 프로필을 슬라이드 형식으로 확인 가능 - PrevSlide()함수와 NextSlide()함수를 이용하여 각 프로필이 슬라이드 형식처럼 이전과 다음으로 넘어가게 된다.
-	query문을 이용한 div 동적할당 기능 - matching table에서 해당 데이터를 검색 후 나온 결과만큼 slide를 생성해서 결과를 나타낸다.
특이사항	PrevSlide()함수와 NextSlide() 함수 구현

15. 회원 탈퇴



withdraw.jsp	
번호	**설명**
1	비밀번호를 입력한다. - input 태그를 이용하여 받은 passwd를 withdraw_serv로 전송한다.
특이사항	

4. DB

본 프로젝트에서 테이블은 총 5개를 사용했다.

<table>
<tr><th colspan="2">Mysql Table</th></tr>
<tr><th>번호</th><th>설명</th></tr>
<tr><td>1</td><td>- user table

Field	Type	Null	Key	Default	Extra
id	varchar(15)	NO	PRI	NULL	
e_mail	varchar(30)	NO		NULL	
passwd	varchar(20)	NO		NULL	
name	varchar(20)	NO		NULL	
sex	varchar(10)	NO		NULL	
birth	varchar(15)	NO		NULL	
phone	varchar(15)	YES		NULL	

총 7개의 column을 사용했으며 기존 웹 페이지에서 이미 중복 검사를 진행한 후 가이을 진행하지만 혹시 모를 오류에 대비해 id column은 primary key로 지정해 중복 id의 가입을 방지했다. 데이터는 아무것도 안들어오면 안되기에 not null option을 부여, 회원가입 시 기본적으로 적는 데이터들을 저장하는 공간이다.</td></tr>
<tr><td>2</td><td>- profile table

Field	Type	Null	Key	Default	Extra
e_mail	varchar(30)	NO	PRI	NULL	
nickname	varchar(15)	YES		NULL	
introduce	varchar(100)	YES		NULL	
job	varchar(10)	YES		NULL	
tendency	varchar(10)	YES		NULL	
date	varchar(15)	YES		NULL	
smoking	varchar(10)	YES		NULL	
pet	varchar(10)	YES		NULL	

총 8개의 column을 사용했으며 user의 e_mail을 primary key로 사용하였다. nickname의 경우 게시판 기능이 활성화 될 경우를 대비해 만들어두었지만 아직 게시판 기능은 공지사항밖에 없기 때문에 id를 저장하는 공간으로 바뀌었다. 그 외 자료들은 룸메이트를 구할 때 보여주는 profile 자료로 회원가입 직후 작성하는 mypage_regist에서 작성하는 데이터를 토대로 자료를 등록하며 그 외 마이페이지 버튼을 누르면 mypage.jsp에서 수정하는 기능을 추가했다. 자료형은 모두 varchar로 지정해서 데이터를 불러오거나 저장할 때 불편함을 제거했다.</td></tr>
<tr><td>3</td><td>- enroll table

Field	Type	Null	Key	Default	Extra
id	varchar(30)	NO		NULL	
form	varchar(10)	NO		NULL	
type	varchar(10)	NO		NULL	
floor	varchar(10)	NO		NULL	
structure	varchar(10)	NO		NULL	
houseoption	varchar(50)	YES		NULL	
roomoption	varchar(25)	YES		NULL	
age	varchar(60)	YES		NULL	
smoking	varchar(10)	NO		NULL	
pet	varchar(10)	NO		NULL	
region	varchar(100)	YES		NULL	
deposit	varchar(20)	NO		NULL	
monthly	varchar(20)	NO		NULL	
date	varchar(25)	NO		NULL	
lat	varchar(15)	YES		NULL	
ing	varchar(15)	YES		NULL	
idx	int	NO	PRI	NULL	auto_increment

enroll의 경우 본 프로젝트에서 가장 많은 column을 사용했다. 해당 데이터는 detailpage.jsp에서 사용되는데 방을 구하는 사람들이 거주자의 정보를 확인할 때 여기있</td></tr>
</table>

<table>
<tr><td></td><td>는 모든 정보가 활용된다. idx는 최신글을 구별하기 위해 등록될때마다 자동등록되게 했으며 이를 통해 front 페이지에서 가장 최신글4개를 화면에 보여준다. lat, lng는 위도 경도를 나타내는데 이를 이용해 가장 가까운 거리에 있는 거주지또한 검색이 가능하다. 각 자료형 역시 varchar로 두었으며 이 역시 데이터 이동 시 편리함을 위해서 해두었다.</td></tr>
<tr><td>4</td><td>- matching table
<table>
<tr><th>Field</th><th>Type</th><th>Null</th><th>Key</th><th>Default</th><th>Extra</th></tr>
<tr><td>owner_id</td><td>varchar(15)</td><td>NO</td><td>PRI</td><td>NULL</td><td></td></tr>
<tr><td>mate_id</td><td>varchar(15)</td><td>NO</td><td>PRI</td><td>NULL</td><td></td></tr>
<tr><td>mate_email</td><td>varchar(30)</td><td>NO</td><td></td><td>NULL</td><td></td></tr>
</table>
matching table은 방을 구하는 사람이 거주자가 등록해놓은 사진과 정보를 보고 마음에 들 경우 “집을 둘러봅니다” 라는 기능을 사용했을 경우 이 table로 거주자의 id와 방을 구하는 유저의 id 및 e_mail이 저장된다. 해당 데이터는 거주자가 다음에 하우스 투어 리스트를 열어 볼 경우 이 table의 자료를 사용하며 방을 구하는 사람의 id와 e_mail을 통해 profile을 살펴본다. 이 table은 primary key가 2개가 한쌍인 구조이다. 한명의 owner에 여러명의 방을 구하는 사람들이 존재하기 때문에 2개의 id가 하나의 primary키를 이루는 구조 만들었다.</td></tr>
<tr><td>5</td><td>- notice table
<table>
<tr><th>Field</th><th>Type</th><th>Null</th><th>Key</th><th>Default</th><th>Extra</th></tr>
<tr><td>title</td><td>varchar(40)</td><td>YES</td><td></td><td>NULL</td><td></td></tr>
<tr><td>content</td><td>varchar(100)</td><td>YES</td><td></td><td>NULL</td><td></td></tr>
<tr><td>date</td><td>varchar(20)</td><td>YES</td><td></td><td>NULL</td><td></td></tr>
</table>
공지사항을 쓰기 위한 게시판의 내용을 저장하는 table이다. 수업시간에 배운 구조를 기본적으로 활용했으며 게시물의 index기능을 사용하지는 않았다. 후에 자유게시판이나 기타 커뮤니티를 위한 활동을 장려할 수 있는 기능을 제작시 게시판 기능을 좀 더 추가할 계획이다.</td></tr>
</table>

5. 자바 코드 설명

본 프로젝트에서 백엔드 부분을 처리하기 위해 여러 가지 방법을 썼으며 그 중 사용한 자바(서블릿 및 클래스파일)코드는 아래와 같다.

(1) 기능서블릿

서블릿 기능을 사용해서 회원가입 및 profile 등록, 수정 그리고 거주지에 대한 상세정보 확인 기능등을 작성하였다. 총 9개의 서블릿 파일이 존재하며 이들이 사용한 기능은 table insert, modify, delete 기능이다.

<table>
<tr><th colspan="2">Servlet & java File</th></tr>
<tr><th>번호</th><th>설명</th></tr>
<tr><td>1</td><td>- detail_serv.java
detailpage.jsp에서 집 둘러보기 신청을 누를 경우 post형식으로 detail_serv에 데이터를 전송한다. 전송받은 데이터를 이용해 sql query문 “insert into matching values를 통해 matching table에 데이터를 저장한 후 신청 결과에 따라 PrintWriter를 이용해</td></tr>
</table>

	alert를 이용해 경고창을 통해 성공 여부를 전달해준다.
2	- enrollment_serv.java enrollment 서블릿의 경우 enrollment.jsp에서 다른 form tag와는 다르게 enctype="multipart/form-data" 라는 옵션을 통해 파일을 전송받을 수 있게 설정해 두었다. 거주지 등록의 경우 집 image가 필요하기 때문에 이러한 방법을 생각했으며 프로젝트 내부에 upload위치를 만들고 다른 사용자들의 이미지와 겹침을 방지하기 위해 사용자의 id로 되어있는 이미지저장폴더를 생성 후 image.jpg로 이미지를 저장받는다. 이후 역시 "insert into enroll (colums) values (data)를 통해 데이터를 전송해 저장하였으며 경고창을 통해 성공 여부를 전송하였다.
3	- login_serv.java lgoin.jsp에서 넘어온 데이터들을 이용해 user테이블에서 아이디와 비밀번호가 매칭되는 row가 있는지 확인 후 로그인 성공 여부를 전달한다. 로그인 성공 시 Session을 이용해 id정보를 session에 저장한다.
4	- mypage_regist, mypage.java mypage_regist와 mypage는 거의 유사한 서블릿이다. regist는 처음 회원가입시 사용자의 profile을 입력하기 위한 query문이 실행되는 서블릿이며 mypage는 그 후 저장되어 있는 profile을 불러오며 profile 수정 시 update query문을 통해 profile table에 저장되어있는 유저 정보를 수정하는 역할을 한다.
5	- register_serv.java register.jsp에서 넘겨받은 데이터를 통해 user 테이블에 정보를 저장하는 역할을 한다. user table에서 id는 primary key이므로 중복이 불가능하며 또한 register.jsp에서 별도의 중복 확인 절차가 있기 때문에 이 서블릿에서는 단순 sql query문을 실행하며 경고창을 통해 성공여부를 전달한다.
6	- user_modify.java user_modify 서블릿은 user테이블의 정보를 수정하기 위한 테이블이며 id와 email의 경우 변경하면 많은 문제가초래 되기 때문에 사실상 전화번호밖에 수정을 못하게 해두었다. sql query문 "update user set column = data" 를 사용했으며 마찬가지로 경고창으로 결과를 보여주었다.
7	- withdraw_serv.java 회원 탈퇴를 위해 생성한 서블릿이다. 2개의 sql_query문을 사용해서 탈퇴를 진행하는데 첫 번째 query문은 id와 passwd의 확인, 2번째는 단계적으로 모든 테이블에서 탈퇴하는 id의 정보를 제거하는 작업을 진행한다. 첫 번째 query문에서의 검색 결과를 Result.next()로 확인하여 결과를 경고창으로 전송한다.
6	- JoinDao, JoinService.java 해당 코드는 id와 e_mail의 중복 여부를 확인하기 위해 만든 코드이다. 중복 여부를 확인하기 위한 가장 간단한 방법은 servlet에 아이디를 전송한 후 중복되는 id가 있는지 검사하는 방법이 있지만 이럴 경우 기존에 작성해두었던 데이터들이 새로고침 되면서 다 날라간다는 단점이 존재한다. 해서 새로운 창을 열어 그 창으로 데이터를 전송한 후 중복

	확인 결과를 기존 창으로 새로고침 없이 불러오는 방법을 찾아 적용하기 위해 만든 클래스이다.
7	- Setup_sql.java 서블릿이 늘어나면서 DB에 접속하기 위한 코드 10~12줄 정도가 지속적으로 반복된다. 해당 반복을 피하기 위해 이 클래스를 생성해 객체가 생성될 때마다 DB에 접속하는 코드를 만들어 DB접속을 위한 코드를 1줄로 줄였다.

6. 외부접속주소

http://210.94.194.59:8080/Project_WebP_real_final/front.jsp