

메인페이지, 방 찾기, 상품상세페이지

신화원

— 목차

01. 프로젝트 개요

02. 시스템 구성

03. 각 파트 소개

04. 프로젝트 소감

01. 프로젝트 개요

- **목 적** : MVC2 등을 적용하여 부동산 거래 웹사이트 생성
(회원가입, 부동산 매물 정보, 매물 등록 및 관리, 마이페이지, 게시판 생성)
- **기 간** : 2021.03.09. ~ 04.15.(4주간)
- **기대효과**
 - ▲ 프로그래밍 언어 숙달
 - ▲ MVC2 구조 습득
 - ▲ 정보 추천 알고리즘 방식 이해
- **결과** : 부동산 거래 홈페이지 구축 완료

01. 프로젝트 개요

프로젝트 개발 기간 (WBS)

SPRING 의 MVC2 패턴을 적용한 부동산 홈페이지 구현

사전 준비

03.09~03.10

팀 구성 및 프로젝트
아이디어 공유

기획

03.15~03.16

주제 선정 및 구체화

요구 분석

03.17~03.18

요구사항 정의서 및
테이블 흐름도 작성

설계

03.19~03.22

각 페이지 초안 및 DB
테이블/변수명 선정

구현 시작

03.23~04.06

기본기능 중심 코드 구현
및 DB 연동

구현 완료

04.07~04.09

디자인 적용(부트스트
랩, CSS)
상세 주석 수정

통합 및 테스트

04.12~04.14

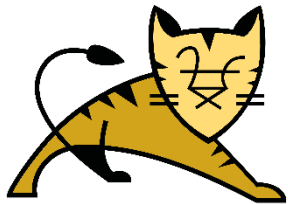
각 파트 통합 및 오류 개선

마무리

04.14~04.15

발표자료 작성 및 제출

02. 시스템 구성 - 사용 프로그램



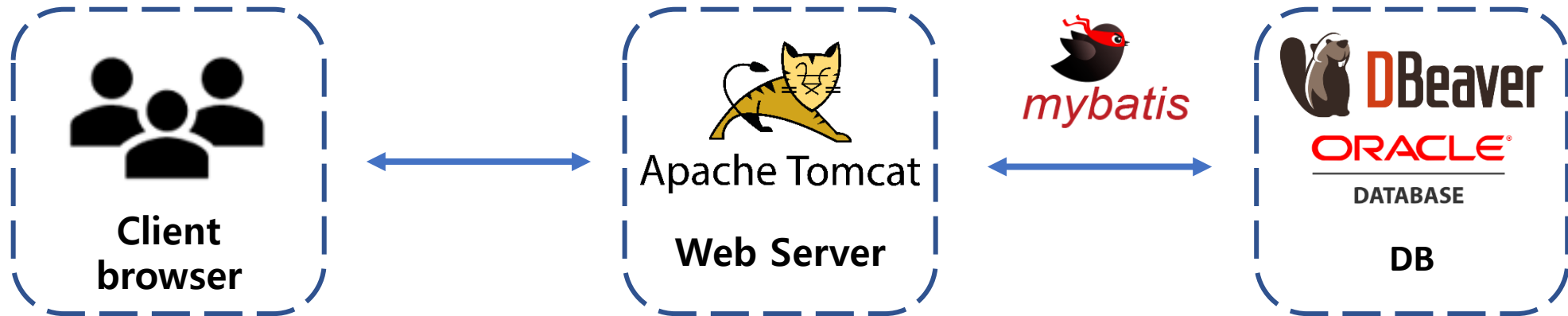
Apache Tomcat



02. 시스템 구성 - 시스템 구성도

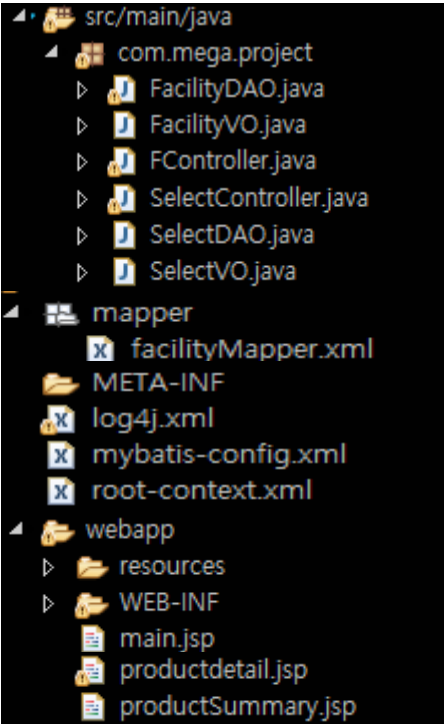


Development Tool

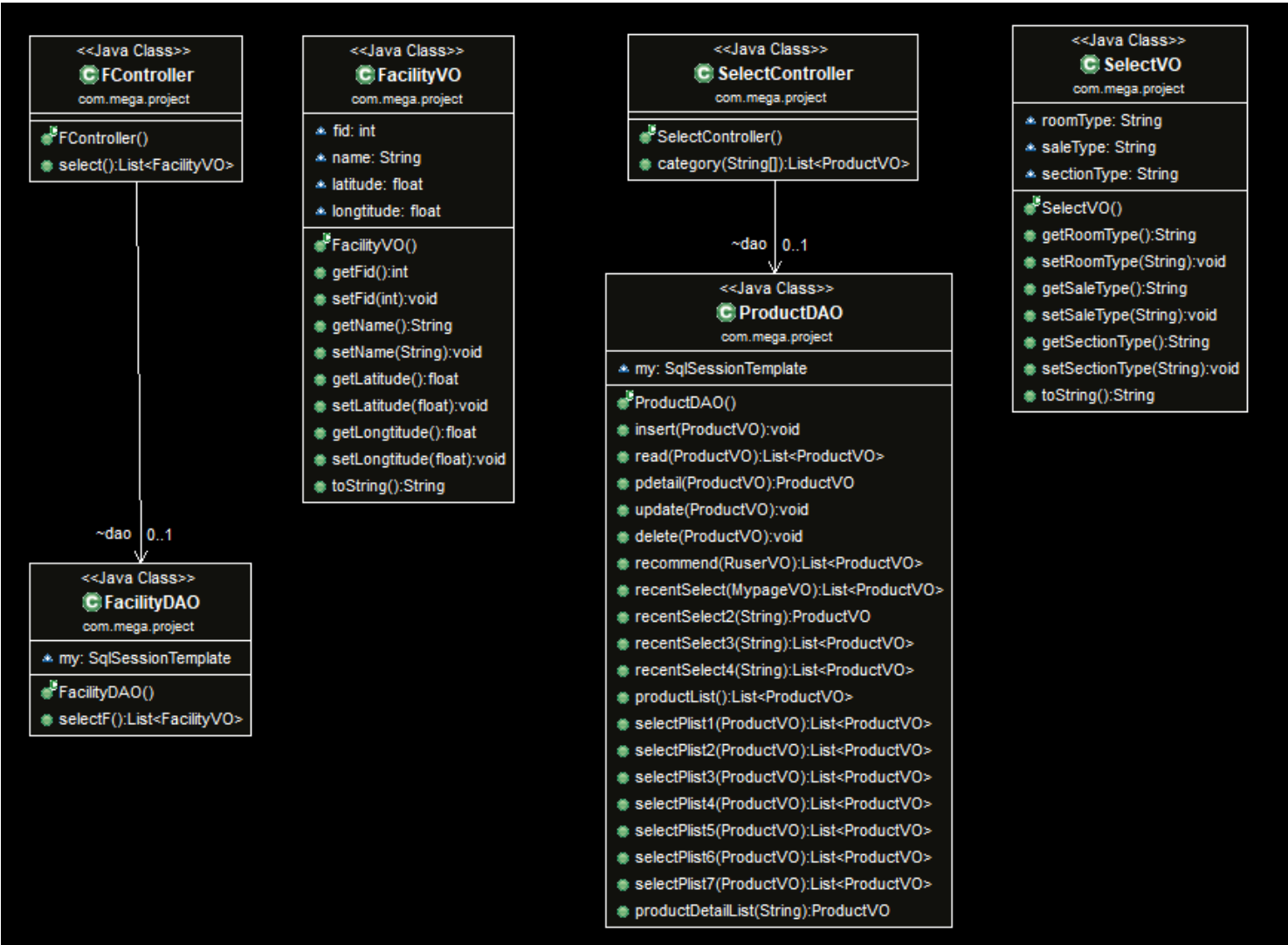


2. 시스템 구성도 - 프로젝트 트리, UML

프로젝트 트리

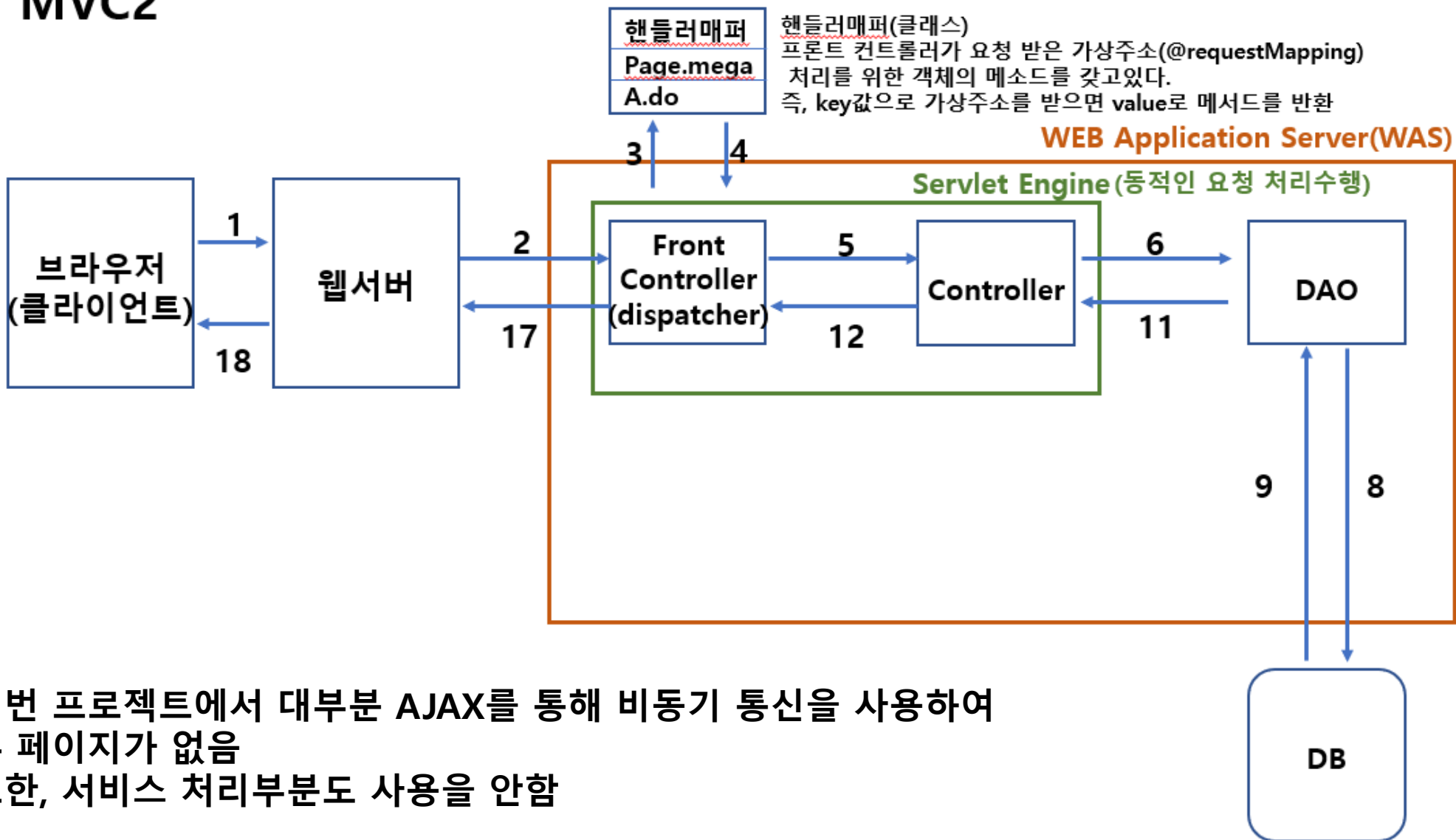


UML



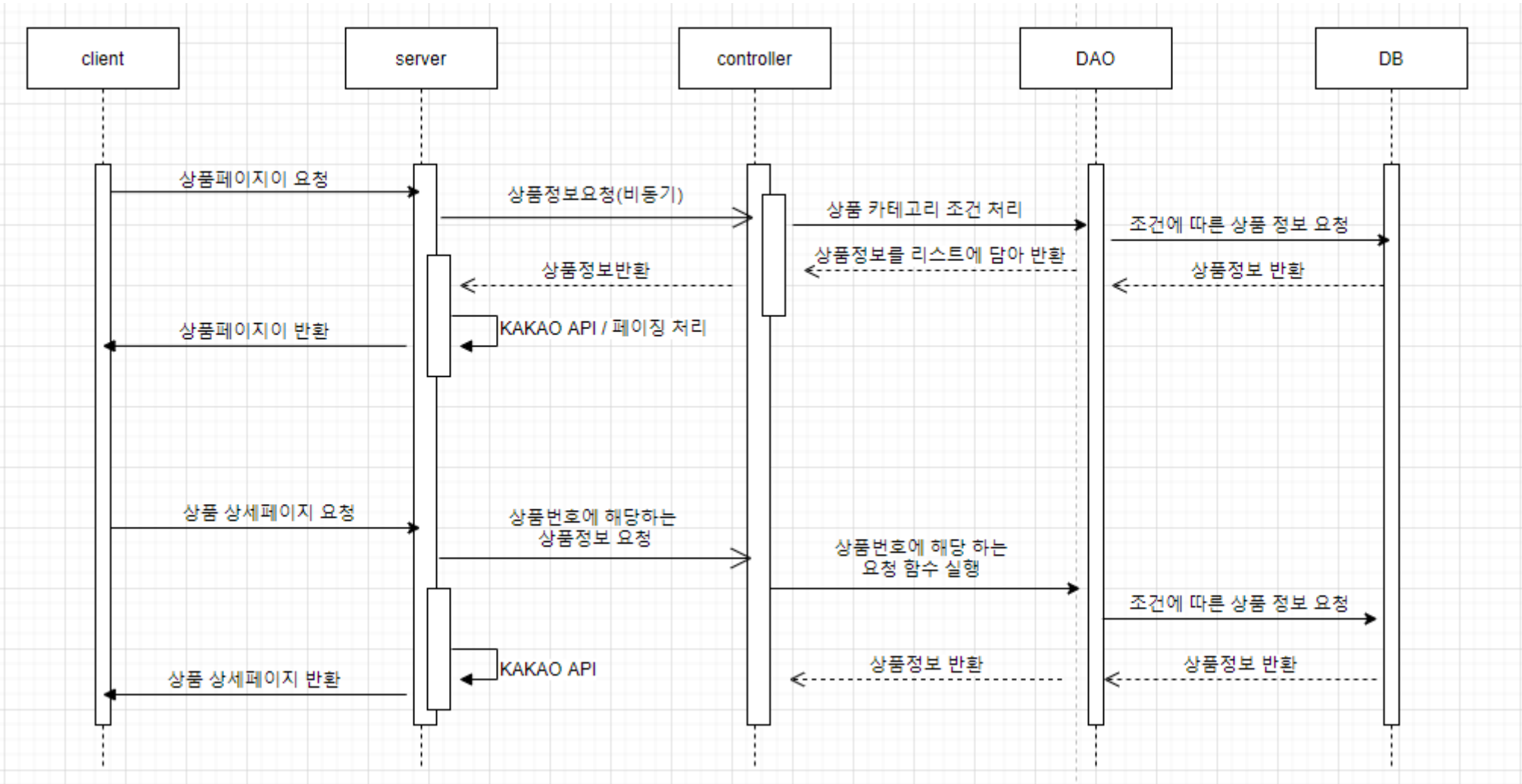
2. 시스템 구성도 - Spring MVC2 모델 처리과정

MVC2

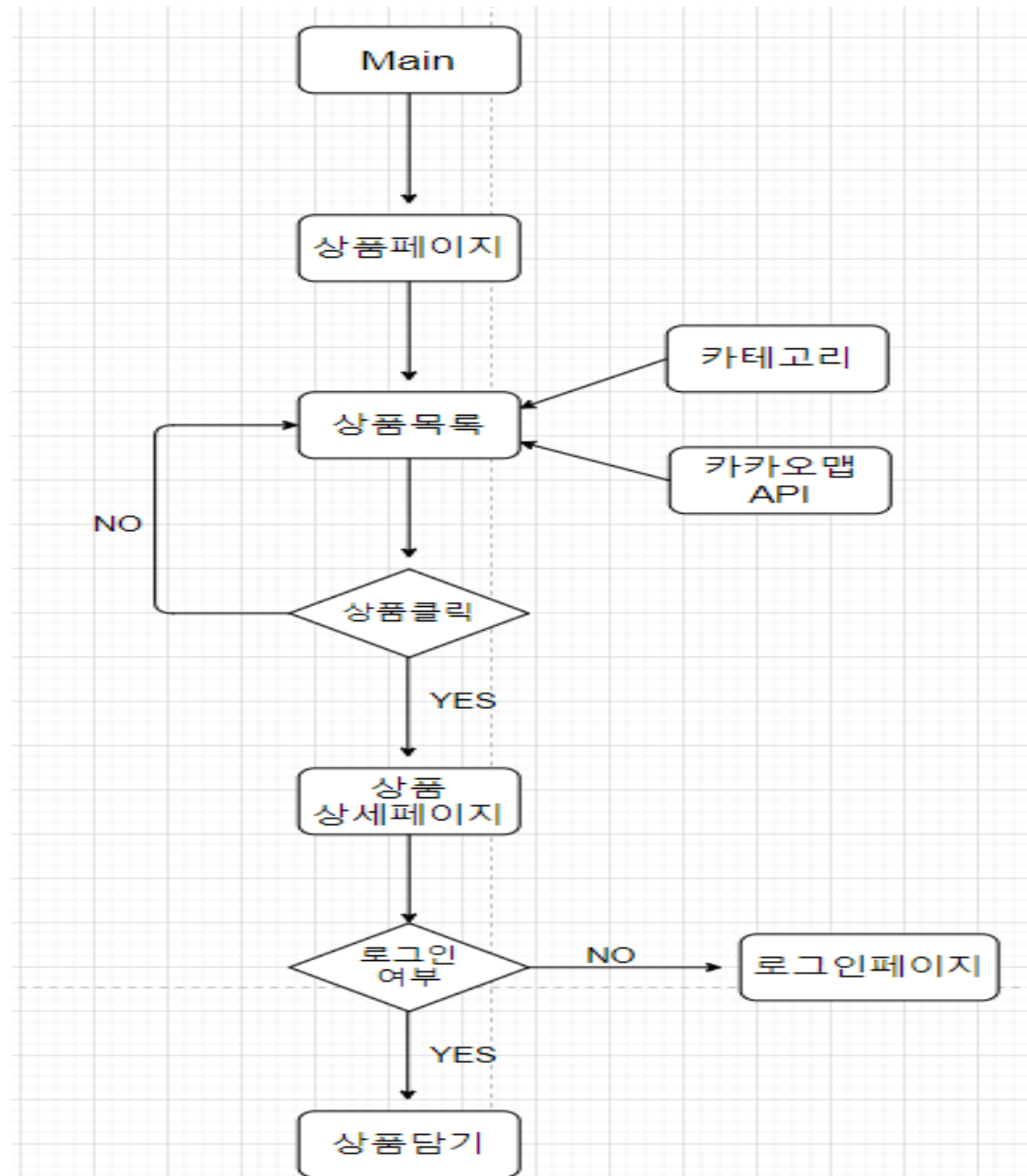


이번 프로젝트에서 대부분 AJAX를 통해 비동기 통신을 사용하여
뷰 페이지가 없음
또한, 서비스 처리부분도 사용을 안함

2. 시스템 구성도 – Sequence diagram



2. 시스템 구성도 - 개발과정 순서도



2. 시스템 구성도 - 테이블 항목

Name: FACILITY

Tablespace: SYSTEM

Comment:

Table Type: N/A

IOT Type:

IOT Name:

☐ Temporary ☐ Secondary ☐ Nested

컬럼명	#	Type	Type Mod	Not Null
ABC NAME	1	VARCHAR2(100)		[X]
123 LATITUDE	2	FLOAT		[X]
123 LONGITUDE	3	FLOAT		[X]
123 FID	4	NUMBER(38,0)		[X]

FACILITY
123 FID
ABC NAME
123 LATITUDE
123 LONGITUDE

컬럼명	번호	DataType	제약조건	내용
FID	1	VARCHAR(20)	Primary key Not Null	편의시설 넘버
NAME	2	VARCHAR(30)	Not Null	편의시설 이름
LATITUDE	3	VARCHAR(10)	Not Null	편의시설 위도
LONGTITUDE	4	VARCHAR(15)	Not Null	편의시설 경도

```
<!-- json으로 변환을 위한 java라이브러리 -->
<dependency>
  <groupId>com.fasterxml.jackson.core</groupId>
  <artifactId>jackson-databind</artifactId>
  <version>2.9.0</version>
</dependency>
<!-- XML문서의 파싱 및 조작을 위한 라이브러리 -->
<dependency>
  <groupId>xerces</groupId>
  <artifactId>xercesImpl</artifactId>
  <version>2.9.1</version>
</dependency>
<!-- 오라클 라이브러리 -->
<dependency>
  <groupId>com.oracle</groupId>
  <artifactId>ojdbc6</artifactId>
  <version>11.2.0.3</version>
</dependency>

<!-- JDBC 3,4단계인 sql문 객체화 와 db로 전달하는 역할을 하는게 마이바티스라이브러리 -->
<dependency>
  <groupId>org.mybatis</groupId>
  <artifactId>mybatis</artifactId>
  <version>3.4.0</version>
</dependency>

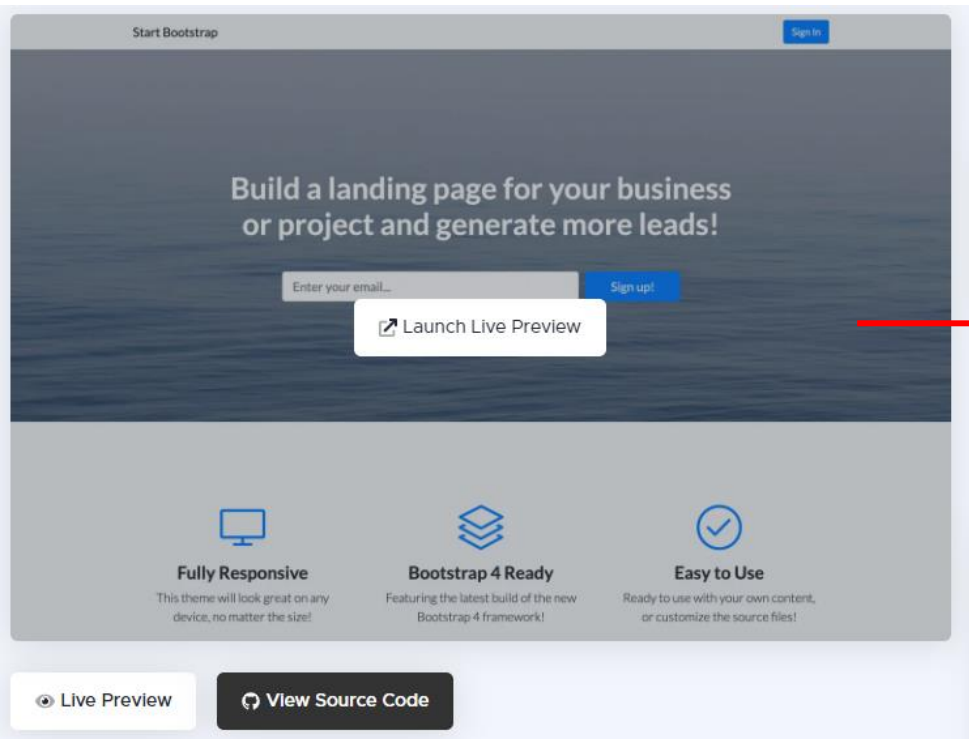
<!-- JDBC 1,2단계인 commons는 connection을 담당하는 것 dbcp는 말 그대로 db커넥션 풀이라고해서 톰캣에서 db로 커넥션 관리해줌 (dbcp라이브러리) -->
<dependency>
  <groupId>commons-dbcp</groupId>
  <artifactId>commons-dbcp</artifactId>
  <version>1.4</version>
</dependency>

<!-- 스프링은 마이바티스와 같은 외부라이브러리 사용을 위해 확장기능 추가 -->
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-jdbc</artifactId>
  <version>5.0.1.RELEASE</version>
</dependency>

<!-- 스프링과 마이바티스를 연결해주는 연결 라이브러리 -->
<dependency>
  <groupId>org.mybatis</groupId>
  <artifactId>mybatis-spring</artifactId>
  <version>1.3.2</version>
</dependency>
```

메인페이지

메인화면



<https://startbootstrap.com/theme/landing-page>
무료 부트스트랩 템플릿을 다운받아서 사용

로그인 파트에서 세션을 잡아준 경우
`Session.getAttribut("세션잡아준ID")`를
활용하여 차이를 둠

내 집은 신촌에 있나방

로그인 유무에 따른 메뉴 바

어떤 방을 찾으세요?

지역 또는 단지명을 입력하세요. [방찾기](#)

[방찾기](#) [마이페이지](#) [고객센터](#) [회원정보수정](#) [회원정보검색](#) [로그인](#) [회원가입](#)

[방찾기](#) [마이페이지](#) [방내놓기](#) [고객센터](#) [회원정보수정](#) [회원정보검색](#) [회원탈퇴](#) [로그아웃](#) [apple님 환영합니다.](#)

```
<!-- 세션처리 -->
<!-- 로그인인 안 된 경우 -->
<% if(session.getAttribute("userid") == null){ %>
<a class="btn btn-primary btn-lg" href="productSummary.jsp">방찾기</a>
<a class="btn btn-primary btn-lg" href="ruser_login.jsp">마이페이지</a>
<a class="btn btn-primary btn-lg" href="qnamain.jsp">고객센터</a>
<a class="btn btn-primary btn-lg" href="ruser_update.jsp">회원정보수정</a>
<a class="btn btn-primary btn-lg" href="ruser_select.jsp">회원정보검색</a>
<a class="btn btn-primary btn-lg" href="ruser_login.jsp">로그인</a>
<a class="btn btn-primary btn-lg" href="signup.jsp">회원가입</a>
<!-- 로그인 된 경우 -->
<% }else{ %>
<a class="btn btn-primary btn-lg" href="productSummary.jsp">방찾기</a>
<a class="btn btn-primary btn-lg" href="mypage_main.jsp">마이페이지</a>
<a class="btn btn-primary btn-lg" href="PMain.jsp">방내놓기</a>
<a class="btn btn-primary btn-lg" href="qnamain.jsp">고객센터</a>
<a class="btn btn-primary btn-lg" href="ruser_update.jsp">회원정보수정</a>
<a class="btn btn-primary btn-lg" href="ruser_select.jsp">회원정보검색</a>
<a class="btn btn-primary btn-lg" href="ruser_delete.jsp">회원탈퇴</a>
<a class="btn btn-primary btn-lg" href="ruser_logout.jsp">로그아웃</a>
<div class="btn btn-primary btn-lg">${userid}님 환영합니다.</div>
<% } %>
```


어떤 방을 찾으세요?

지역 또는 단지명을 입력하세요.

방찾기



아파트 / 145000
매매 / 84.978 / 편세권
서대문구에 위치한 아파트입니다.



아파트 / 147800
매매 / 84.98 / 편세권
서대문구에 위치한 아파트입니다.



아파트 / 132500
매매 / 84.968 / 편세권
서대문구에 위치한 아파트입니다.

```
@RequestMapping("checkBasket")
@ResponseBody //메서드의 return값을 HTTP reponse의 body에 담는 역할
public List<ProductVO> checkBasket(@RequestParam(value="userid1") String userid) {
    // main 페이지 userid1 값을 가져옴
    System.out.println("유저아이디: "+userid);
    RuserVO bag = dao.checkBasket(userid);
    // Ruser checkbasket 메서드의 userid에 해당함

    List<ProductVO> list = dao2.recommend(bag);
    // ProductVO
    // RuserVO, R
    <select id="checkBasket" parameterType="String" resultType = "ruserVO">
    select usertype, userroom, userinterest from RUSER where userid = #{userid}
    </select>
    System.out.p
    return list;
    // 방 이미지가 담긴 리스트를 반환값으로 넘겨줌
}
```

```
public RuserVO checkBasket(String id) {
    return my.selectOne("ruser.checkBasket", id);
}
```

```
<select id="checkBasket" parameterType="String" resultType = "ruserVO">
select usertype, userroom, userinterest from RUSER where userid = #{userid}
</select>
```

```
//세션처리된 아이디에 해당하는 추천상품
$(function() {
    userid = "${userid}" //세션처리된 id를 userid변수에 담는다.
    //AJAX를 통해 checkBasket이라는 가상주소로 넘어간다.
    if(userid != null & userid != ""){ underdefined방지
        console.log(userid)
        $.ajax({
            async:false,
            url: "checkBasket",
            data: {
                userid1: userid,
            },
            success: function(result) {
                if(result.length == 1){
                    $('#room1').html("<a href = 'productdetail.jsp?pnum="+result[0].pnum+"'">img src = 'resources/rimg/
                    $('#room1').append('<h4>'+result[0].ptype+ " / "+result[0].price+"</h4>")
                    $('#room1').append('<h5>'+result[0].ctype+ " / "+result[0].rsize+ " / "+result[0].placetype+"</h5>")
                    $('#room1').append('<h5>'+result[0].pinfo+"</h5><br>")
                    $('#room2').html("")
                    $('#room3').html("")
                }
            }
        })
    }
}
```

세션 처리된 아이디를 checkBasket으로 넘겨주
넘겨준 아이디의 관심목록 해당하는 추천 정보를
출력

방찾기 페이지

방찾기

버튼 클릭

productsummary.jsp

내 집은 신촌에 있나방

- 방찾기
- 마이페이지
- 방내놓기
- 고객센터
- 회원정보수정
- 회원탈퇴
- 로그아웃
- apple님 환영합니다.

방종류 거래유형 땡세권



전세 아파트
60000(만원)
역세권



월세 아파트
10000/90(만원)
역세권



월세 아파트
68000/90(만원)
숲세권



월세 아파트
5000/83(만원)
물세권



전세 아파트
90000(만원)
역세권



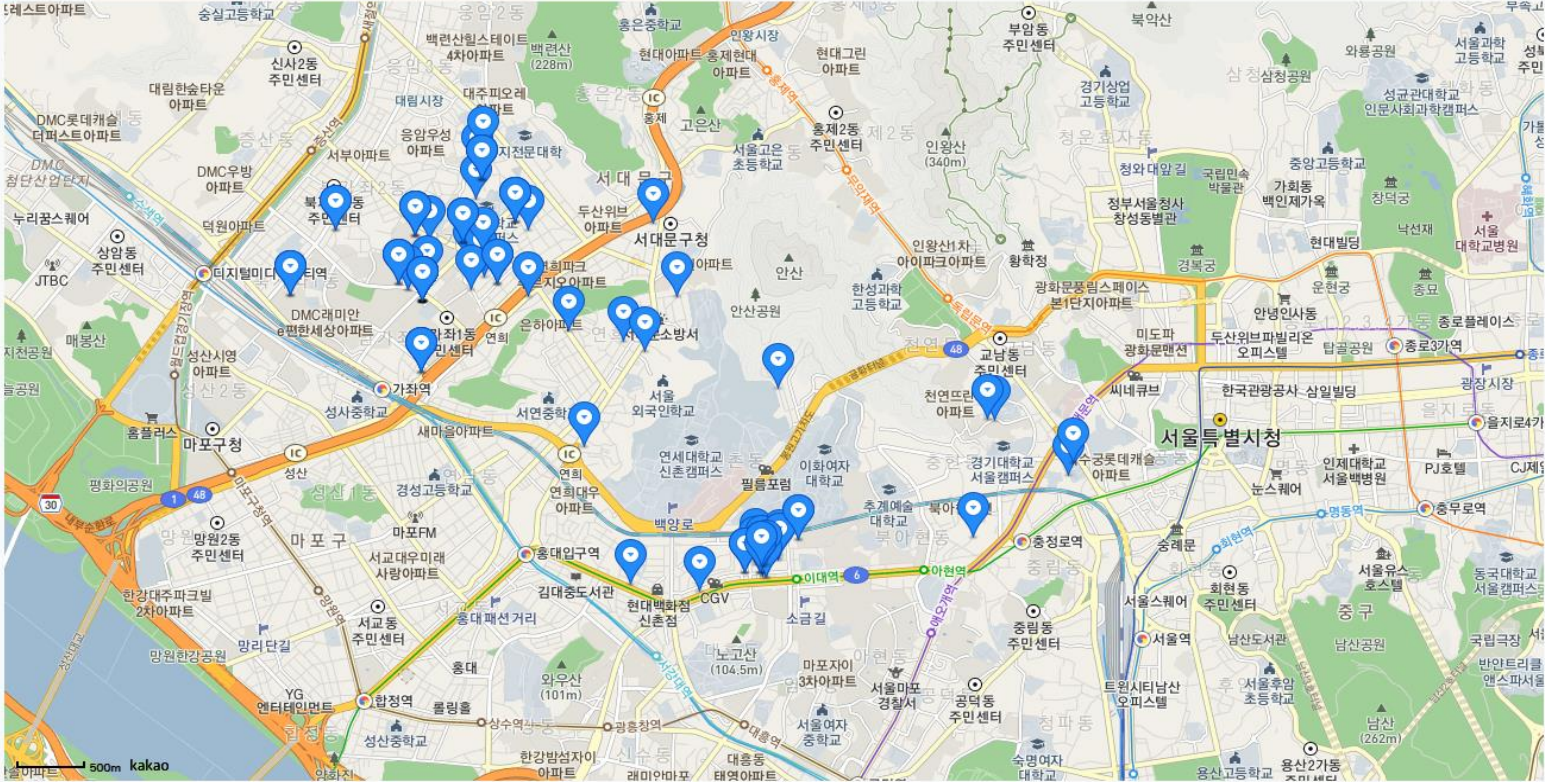
월세 아파트
70000/80(만원)
편세권



월세 아파트
30000/150(만원)
팍세권



월세 아파트
2000/70(만원)
슬세권



방찾기(AJAX활용하여 지도와 상품목록에 필요한 데이터 요청)

```
$(function() {  
  /*  
  AJAX 활용  
  1. 등록된 상품목록들을 지도에 찍어주는 KaKao API  
  2. 등록된 상품목록들을 가져와 화면에 찍어주기  
  3. 상품목록들의 페이지징 처리  
  4. 셀렉트 태그를 이용해 카테고리별로 구분된 상품목록 가져오기 + 지도에 찍어주기  
  
  상품컨트롤러에 접근하여 전체상품 정보 가져오기  
  */  
  
  //카테고리별 입력값 저장할 배열  
  //기본 화면 : 카테고리 선택된게 없기 때문에 none, none, none이 들어가있음  
  var arr1 = [ "none", "none", "none"];  
  //디폴트 화면(기본 시작화면 모든 상품정보를 보여주도록 설정)  
  //셀렉트를 통해 상품을 결정하지 않은 경우 ~> 기본화면  
  if(arr1[0] == "none" & arr1[1] == "none" & arr1[2] == "none"){  
    $.ajax({  
      url: "getProduct",  
      type: "POST",  
      data: {  
        arr: arr1 //카테고리 내역이 담긴 배열  
      },  
      success: function(result) {  
        //DB에서 받아온 주소 목록들을 담아둘 배열  
        var address = []  
  
        //방사진을 담을 배열  
        var rooming = []  
        var rooming2 = []  
        var rooming3 = []  
  
        //DB에서 리턴한 리스트를 JSON으로 변경해주고 주소, 상품의 특징을 담아준다  
        //JSON은 현재 배열상태  
        //받아온 모든 정보를 for문을 통해 하나씩 담아준다.  
        for (var i = 0; i < result.length; i++) {  
          address[i] = result[i].address //주소  
          feature1[i] = result[i].ctype//특징1(거래타입)  
          feature2[i] = result[i].price//특징2(가격)  
        }  
      }  
    });  
  }  
});
```

result에 전체 상품에 대한 정보
Json데이터로 들어있다.

추출된 데이터 길이만
반복문을 돌려 데이터
하나씩 출력

```
@Autowired  
ProductDAO dao;  
  
Selected Controller  
  
@RequestMapping("getProduct")  
//JSP(브라우저)에서 AJAX를 통해 보낸 배열을 컨트롤러에서 받는 방법  
//파라미터로 @RequestParam(value = "보낸 데이터 키값(아이디값)") String[] arr)  
// AJAX로 보낸 데이터(배열) 받는 데이터가 배열(List)  
@ResponseBody  
public List<ProductVO> category(@RequestParam(value = "arr[]") String[] arr) {  
  //AJAX를 통해 받아온 배열형태의 데이터를 각 타입의 문자열 변수에 대입  
  String roomType = arr[0];  
  String saleType = arr[1];  
  String sectionType = arr[2];  
  
  //받아온 배열의 내용을 풀어서 VO에 담아주기  
  ProductVO bag = new ProductVO();  
  bag.pctype = roomType;  
  bag.ctype = saleType;  
  bag.placetype = sectionType;  
  
  // 상품을 담은 리스트 초기화 => List는 인터페이스라 new키워드 못씀 -> ArrayList()대체  
  //상품테이블로 보내줌  
  List<ProductVO> list = new ArrayList<ProductVO>();  
  
  // 전체상품 정보를 출력  
  if(roomType.equals("none") & saleType.equals("none") & sectionType.equals("none")) {  
    list = dao.productList();  
    return list;  
  }  
}
```

초기 화면 모두 none이라는 값을 갖고있다.

받아온 정보를 ProductVO타입 변수에 저장

전체 상품정보를 담아줄 리스트 생성

```
// 전체 상품 목록 가져오기  
public List<ProductVO> productList() {  
  return my.selectList("product.productList");  
}
```

```
<!-- 전체상품 정보를 출력 -->  
<select id="productList" resultType="productVO">  
  select * from "PRODUCT"  
</select>
```

상품테이블에 있는 모든 로우값 가져오기

방찾기(카카오맵 API)

JavaScript API를 불러오기

```
<script type="text/javascript" src="//dapi.kakao.com/v2/maps/sdk.js?appkey=
```

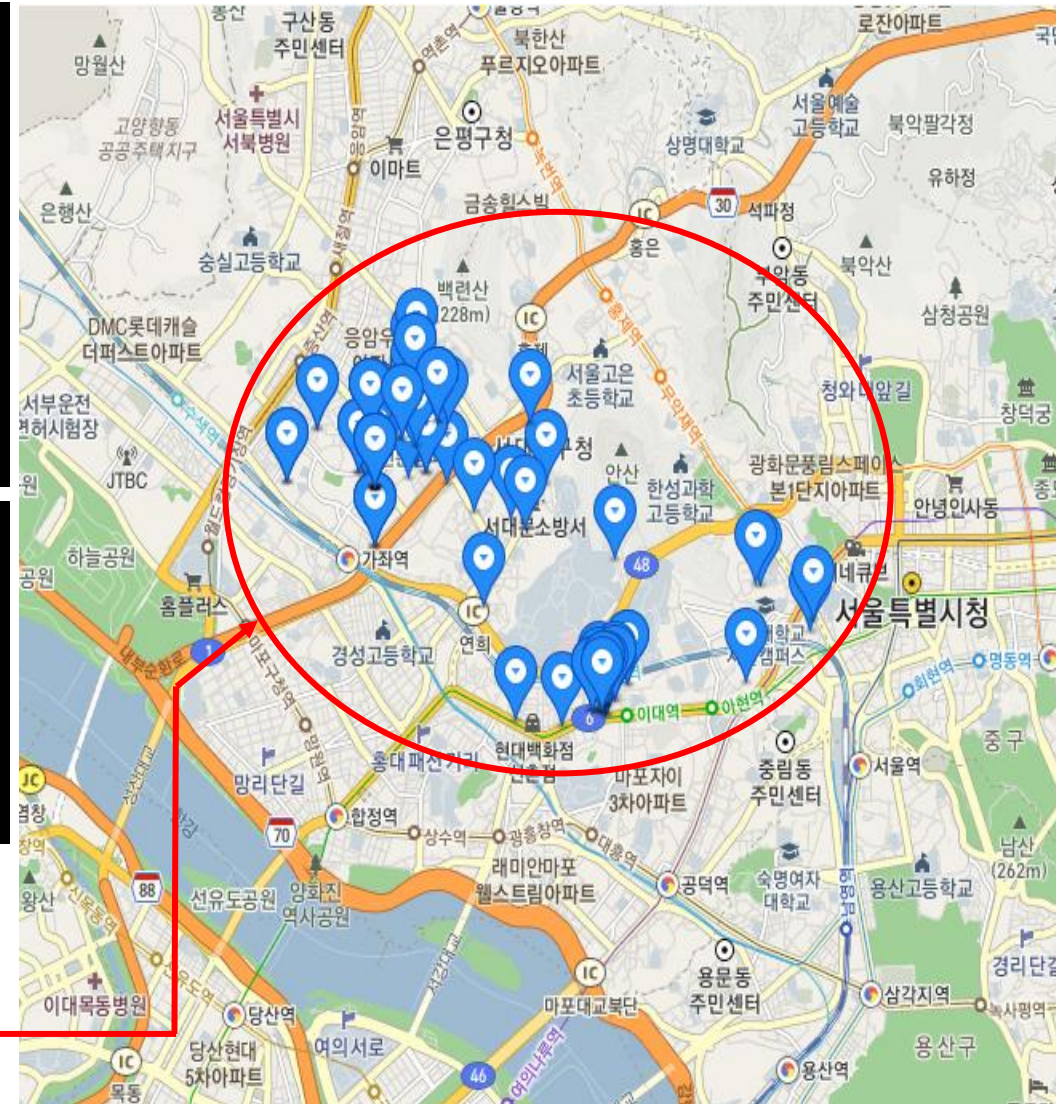
```
&libraries=services,clusterer,drawing"></script>
```

```
/*
KAKAO 지도 API 활용
위에서 얻어온 주소값(도로명/지명)들을 위도/경도로 바꿔준 후 -> 위도/경도를 Kakao서버로 넘겨 지도에 찍어준다.
*/
//geocoder : 주소를 입력하면 위도와 경도로 바꿔주는 객체
var geocoder = new kakao.maps.services.Geocoder();
//Body부분에서 'map'이라는 아이디를 갖는 태그를 가져옴
var mapContainer = document.getElementById('map');
var mapOption = {
  center : new kakao.maps.LatLng(37.56203952196803, 126.93773526913148), //지도의 기준점을 설정
  level : 7
};
var map = new kakao.maps.Map(mapContainer, mapOption); //지도의 요소들을 담은 map이라는 변수 생성
```

```
//배열로 받아온 주소개수 만큼 반복문 실행 => 그 개수만큼 지도에 표시해야하기 때문에
for (let i = 0; i < address.length; i++) {
  //addressSearch : 입력한 주소를 위도/경도로 변경해주는 메서드
  geocoder.addressSearch(address[i], function(result, status) {
    //정상적으로 검색이 완료됐으면
    if (status === kakao.maps.services.Status.OK) {
      //coords라는 변수에 위도와 경도를 찍어준다 ~> y좌표가 위도/x좌표가 경도
      //(https://developers.kakao.com/docs/latest/ko/local/dev-guide#address-coord : RoadAddress부분확인하기)
      var coords = new kakao.maps.LatLng(result[0].y, result[0].x);
      Coords : 변경된 위도/경도를 담고있
```

```
// 결과값으로 받은 위치를 마커로 표시합니다
var marker = new kakao.maps.Marker({
  map : map,
  position : coords,
  clickable: true
});
```

각 위도/경도에 해당하는 좌표를 찍어줌

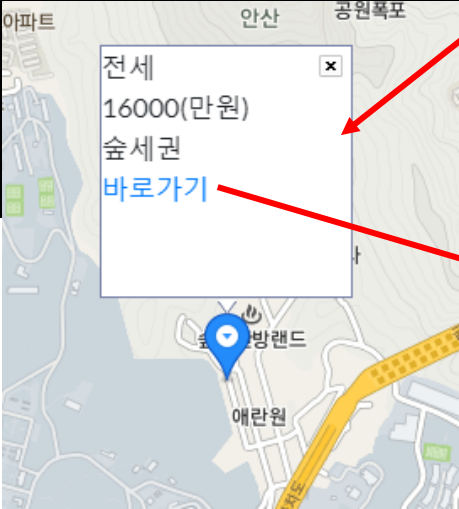


방찾기(카카오맵 API) - 인포윈도우 생성

```
// 마커를 클릭했을 때 마커 위에 표시할 인포윈도우를 생성합니다
// 지도에서 표시된 상품의 마커를 클릭시 -> 상세 정보를 보여주도록 설정
// 태그와 string 사이는 +로 연결
var iwContent = '<div style="width:120px; height:150px">'
+feature1[i]+'<br>'+feature2[i]+'(만원)+'<br>'+feature3[i]+'
"<br><a href =productdetail.jsp?pnun="+pnun[i]+">바로가기</a></div>', // 인포윈도우에 표출될 내용으로 HTML 문자열이나 document element가 가능합니다
iwRemoveable = true; // removeable 속성을 ture 로 설정하면 인포윈도우를 닫을 수 있는 x버튼이 표시됩니다

// 인포윈도우를 생성합니다(https://apis.map.kakao.com/web/sample/removableCustomOverlay/)
var infowindow = new kakao.maps.InfoWindow({
  content : iwContent,
  removable : iwRemoveable
});

// 마커에 클릭이벤트를 등록합니다
// 클릭시 인포윈도우를 띄어줌
kakao.maps.event.addListener(marker, 'click', function() {
  // 마커 위에 인포윈도우를 표시합니다
  infowindow.open(map, marker);
});
```



바로가기 버튼을 누르게 되면
상품상세 페이지로 전환


방찾기(상품목록 생성 & 페이징 처리)

```
/*
상품의 목록생성
상품목록 페이지처리
*/
var totalData = result.length; //전체 데이터
var dataPerPage = 8; //한 페이지에 표현할 데이터
var pageCount = 4; // 한 화면에 나타낼 페이지 수


paging(totalData, dataPerPage, pageCount, 1); // 맨 처음 초기화면 생성(페이징 초기화)
function paging(totalData, dataPerPage, pageCount, currentPage) {
    var totalPages = Math.ceil(totalData / dataPerPage); //전체 페이지
    var pageGroup = Math.ceil(currentPage / pageCount); //페이지 그룹
    var selectedPage = currentPage //선택된 페이지
    var last = pageGroup * pageCount;
    if (last > totalPages) // 마지막 페이지 그룹 넘버가 전체페이지를 넘는 경우
        last = totalPages;
    var next = last + 1; //페이지 그룹에서 마지막 숫자 + 1
    var first = last - (pageCount - 1); //페이지 그룹에서 보여지는 첫번째 숫자
    var prev = first - 1; //이전페이지는 현재 페이지 그룹에서 첫번째 숫자 - 1

    var html = ""; //페이징정보를 표시하기 위한 변수
    var list = ""; //상품목록들을 표시하기 위한 변수
    페이징, 상품 목록을 찍어주기 위
    변수 설정
}
```


```
//페이징 바 설정
if (prev > 0)
    html += "<a href='#' id='prev' style='font-size: 28px; font-weight: bold'></a>";
if(first < 0 | first == 0){
    for (var i = 1; i <= last; i++) {
        html += "<a style='font-size: 28px;font-weight: bold' href='#' id='"+ i + "'"> i + "</a>";
    }
}else{
    for (var i = first; i <= last; i++) {
        html += "<a style='font-size: 28px;font-weight: bold' href='#' id='"+ i + "'"> i + "</a>";
    }
}
if (last < totalPages)
    html += "<a href='#' id='next' style='font-size: 20px;font-weight: bold'>></a>";
```




전세 아파트
60000(만원)
역세권




전세 아파트
90000(만원)
역세권




월세 아파트
10000/90(만원)
역세권



월세 아파트
70000/80(만원)
편세권




월세 아파트
68000/90(만원)
숲세권



월세 아파트
30000/150(만원)
편세권



월세 아파트
5000/83(만원)
숲세권



월세 아파트
2000/70(만원)
스세권

1234 >

페이징 바 설정

//list에 상품들을 가져와 넣어줘야함=> 안그러면 그냥 페이지만 표시됨
 //페이지당 위에서 전역변수로 설정한 dataPerPage만큼 상품을 출력해줘야해서 설정한 부분(특정 패턴을 따라)

```
for (var i = (selectedPage - 1) * dataPerPage; i < selectedPage * dataPerPage; i++) {
  if(result[i] != null){ //데이터가 있을 때만 화면에 출력
    list += "<div class = 'product' style = 'margin: 0px 0px 0px 60px'"
    list += "<div class = 'pp'"
    //a태그를 통해서 사진을 누르게되면 상품 상세페이지인 productdetail.jsp로 pnum에 pn[i]의 값을 넣어서 보내주게된다
    //productdetail.jsp에서는 pnum을 받아서 사용할 수 있다.
    list += "<a href = 'productdetail.jsp?pnum="+pnum[i]+"'"><img src = 'resources/rimg/'
      +roomimg[i]+" style = 'width: 200px; height: 100px'"></a>"
    list += "<div>" + result[i].ctype+ " + result[i].ptype+ "</div>"
    list += "<div>" + result[i].price+ "(만원)"</div>"
    list += "<div>" + result[i].placetype+ "</div>"
    list += "</div>"
    list += "</div>"
  }
  else{
    list += "<div style = 'height : 69px'"></div>"
  }
}
```

상품 목록 생성


//초기화면을 위해 넣는 부분 끝

//.html()을 사용해서 body부분에 있는 div에 출력해줌 ~> append대신 사용


```
$('#list').html(list)
$("#paging").html(html):
```

상품 및 페이징 목록 생성


```
//페이징 번호클릭시 수행
$("#paging a").click(function() {
  var list = ""; // 상품목록 초기화해주기 : 주석 처리시 맨 처음 화면의 상품리스트에 쌓이게됨(중복됨)
  // 초기화를 해줘야지 클릭할 때마다 상품초기화 되고 다시 쌓아주는 식
  //이전페이지, 다음페이지 처리를 위한 설정
  var $item = $(this); //$(this) : 이벤트(현재 클릭이벤트)가 발생한 요소들의 정보들
  // 현재 페이지를 눌렀을때 <a href='#' id='+'>'+</a> 이러한 태그의 요소들의 정보인 a(태그) #주소 i(id정보) 가 출력됨
  // 2번째 페이지를 누를 경우 결과로 a#2가 나온다.
  var $id = $item.attr("id"); //attr : 요소 및 속성 가져오기 ~> 위에서 말한 a태그 #주소 i(id정보)중 id에 해당하는 것만 출력
  var selectedPage = $item.text();
  if ($id == 'next')
    selectedPage = next; //만약 next를 아이디로하는 a태그를 클릭(다음 페이지로 넘기기) selectedPage에 next번호가 들어간다
  if ($id == 'prev')
    selectedPage = prev;
  paging(totalData, dataPerPage, pageCount, selectedPage); // 페이징 함수 처리
})//클릭함수달기
```




전세 아파트
60000(만원)
역세권




월세 아파트
10000/90(만원)
역세권




월세 아파트
68000/90(만원)
숙세권




월세 아파트
5000/83(만원)
숙세권




전세 아파트
90000(만원)
역세권




월세 아파트
70000/80(만원)
편세권




월세 아파트
30000/150(만원)
편세권




월세 아파트
2000/70(만원)
스세권




월세 아파트
1000/110(만원)
스세권




매매 단독/다가구
16850(만원)
숙세권




전세 단독/다가구
7000(만원)
역세권




전세 단독/다가구
8000(만원)
편세권




매매 단독/다가구
170000(만원)
편세권



매매 단독/다가구
270000(만원)
스세권



전세 단독/다가구
9000(만원)
학세권



전세 단독/다가구
13000(만원)
숙세권

1 234 >
 1 234 >

변경

```
"<a href='#' id='prev' style='font-size: 28px; font-weight: bold'"></a>"
"<a style='font-size: 28px;font-weight: bold' href='#' id='+'>'+</a>"
"<a href='#' id='next' style='font-size: 20px;font-weight: bold'"></a>"
```


방찾기(카테고리 설정에 따른 상품&지도 변화)

아파트

월세

학세권



월세 아파트
10000/140(만원)
학세권



월세 아파트
30000/135(만원)
학세권

상품개수: 2

카테고리를 설정한 경우
콘솔을 통해 확인한 결과
위의 조건에 부합하는
상품은 2개

1



```
//방종류 셀렉트 태그에서 선택된 값 받아오기
//$(this).val() -> 타입 종류
$('#roomType').change(function() {
    var roomType1 = $(this).val() //roomType변수에 방종류 넣어줌
    arr1[0] = roomType1 //전역변수로 처리된 arr1배열에 넣어준다
    //서버로 데이터 넘겨주기
    $.ajax({
        type: "POST",
        url: "getProduct",
        data: {
            arr: arr1,
        },
        success: function(result) {
```

```
<select name="roomType" id = "roomType" style="width: 100px; height: 30px;">
    <option value="none" selected>방종류</option>
    <option value="단독/다가구">단독/다가구</option>
    <option value="아파트">아파트</option>
    <option value="오피스텔">오피스텔</option>
</select>
```

```
//거래유형 셀렉트 태그에서 선택된 값 받아오기
$('#saleType').change(function() {
    var saleType1 = $(this).val() //saleType변수에 거래유형 넣어줌
    arr1[1] = saleType1 //전역변수로 처리된 arr1배열에 넣어준다
    console.log(arr1)

    $.ajax({
        type: "POST",
        url: "getProduct",
        data: {
            "arr": arr1,
        },
        success: function(result) {
```

```
<select name="saleType" id = "saleType" style="width: 100px; height: 30px;">
    <option value="none" selected>거래유형</option>
    <option value="전세">전세</option>
    <option value="월세">월세</option>
    <option value="매매">매매</option>
</select>
```

```
//평세권 셀렉트 태그에서 선택된 값 받아오기
$('#sectionType').change(function() {
    var sectionType1 = $(this).val() //sectionType변수에 평세권 넣어줌
    arr1[2] = sectionType1 //전역변수로 처리된 arr1배열에 넣어준다
    console.log(arr1)
    $.ajax({
        type: "POST",
        url: "getProduct",
        data: {
            "arr": arr1
        },
        success: function(result) {
```

```
<select name="sectionType" id = "sectionType" style="width: 100px; height: 30px;">
    <option value="none" selected>평세권</option>
    <option value="역세권">역세권</option>
    <option value="편세권">편세권</option>
    <option value="학세권">학세권</option>
    <option value="술세권">술세권</option>
    <option value="스세권">스세권</option>
</select>
```

방찾기(카테고리 설정에 따른 상품정보 요청)

@Controller

```
public class SelectController {
```

@Autowired

```
ProductDAO dao;
```

@RequestMapping("getProduct")

//JSP(브라우저)에서 AJAX를 통해 보낸 배열을 컨트롤러에서 받는 방법

//파라미터로 @RequestParam(value = "보낸 데이터 키값(아이디값)[]") String[] arr)

// AJAX로 보낸 데이터(배열) 받는 데이터가 배열(List)

@ResponseBody

```
public List<ProductVO> category(@RequestParam(value = "arr[]") String[] arr) {
```

//AJAX를 통해 받아온 배열형태의 데이터를 각 타입의 문자열 변수에 대입

```
String roomType = arr[0];
```

```
String saleType = arr[1];
```

```
String sectionType = arr[2];
```

//받아온 배열의 내용을 풀어서 VO에 담아주기

```
ProductVO bag = new ProductVO();
```

```
bag.ptype = roomType;
```

```
bag.ctype = saleType;
```

```
bag.placetype = sectionType;
```

// 상품을 담은 리스트 초기화 => List는 인터페이스라 new키워드 못씀 -> ArrayList()대체

//상품데이터로 보내줌

```
List<ProductVO> list = new ArrayList<ProductVO>();
```

카테고리 선택여부에 따라 조건을 나누고 그에 해당하는 DAO의 메소드를 호출

방종류 거래유형 땡세권

1. 하나의 카테고리만 선택된 경우

```
// 선택된 방종류에 해당하는 정보를 출력
else if(saleType.equals("none") & sectionType.equals("none")) {
    list = dao.selectPlist1(bag);
    return list;
}
```

<!-- 선택된 방종류에 해당하는 정보를 출력 -->

```
<select id="productList1" parameterType="productVO" resultType="productVO">
    select * from "PRODUCT" where ptype = #{ptype}
</select>
```

방종류 거래유형

거래유형 땡세권

방종류 땡세권

<!-- 선택된 방종류와 거래유형에 해당하는 정보를 출력 -->

```
<select id="productList4" parameterType="productVO" resultType="productVO">
    select * from "PRODUCT" where ptype = #{ptype} and ctype = #{ctype}
</select>
```

2. 두개의 카테고리가 선택된 경우

```
// 선택된 방종류와 거래유형에 해당하는 정보를 출력
else if(sectionType.equals("none")) {
    list = dao.selectPlist4(bag);
    return list;
}
```

방종류 거래유형 땡세권

3. 세개의 카테고리가 선택된 경우

```
// 선택된 방종류와 거래유형 그리고 세권유형에 해당하는 정보를 출력
else {
    list = dao.selectPlist7(bag);
    return list;
}
```

<!-- 선택된 방종류와 거래유형 그리고 세권유형에 해당하는 정보를 출력 -->

```
<select id="productList7" parameterType="productVO" resultType="productVO">
    select * from "PRODUCT" where ptype = #{ptype} and ctype = #{ctype} and placetype = #{placetype}
</select>
```


상품 상세 페이지

상품 상세 페이지

localhost:8888/project/productdetail.jsp?pnun=59

DMC파크뷰자이1단지 전용면적

매매 132500

84.968㎡

상품담기

상품상세설명

편세권

주)래미안북덕방부동산중개법인

김미희 01082739244

거래현황: 협의중

해당층/건물층 10/20층

전용/공급면적

84.968㎡

방수/욕실수

3/2개

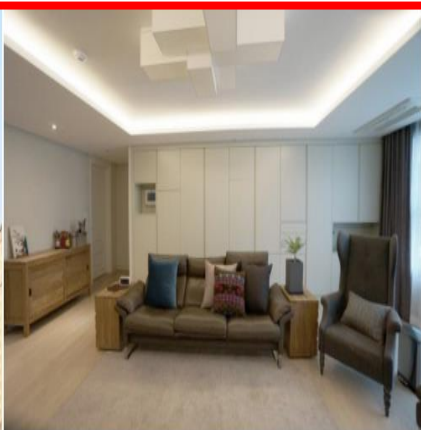
난방종류

개별난방

주차대수

총 1대

입주가가능일 즉시입주



```
var pnun = ${param.pnun} //상품페이지url을 통해 넘겨온 파라미터값
var pname; //전역변수 선언
var address; //전역변수 선언(주소)

$.ajax({
  //매물정보 받아올 AJAX
  /*
  수정할 부분
  상품컨트롤러로 상품번호를 보내 상품 받아오기
  */
  url: "productInfo", //매물번호를 통해서 상품정보를 가져온다
  data: {
    pnun : pnun,
  },
```

pnun으로 넘겨온 파라미터를
productInfo(가상주소)로 넘겨줌
-> 해당 상품 정보 가져오기

```
success: function(result) {
  console.log("결과 확인: ", result)
  pname = result.pname //매물명 : 리모델링된 원룸
  ptype = result.ptype //매물타입 : 아파트
  ctype = result.ctype //거래형태 : 매매
  price = result.price //가격 : 500/42만원

  //html body태그안에 있는 div태그에 출력된 정보들을 찍어주는 과정
  $(".pname").html(pname)
  $(".ctype").html(ctype)
  $(".price").html(price)
}
```

//상품상세페이지에서 상품 목록 가져오기(productDetail.jsp페이지에서 넘겨진 pnun(상품번호)을 통해 해당하는 상품의 정보를 출력)

@RequestMapping("productInfo")

@ResponseBody

public ProductVO productInfo(@RequestParam(value = "pnun") String pnun) { //AJAX에서 data를 보낼 때 String타입으로 보낸 경우 받아줄때

ProductVO list = dao.productDetailList(pnun);

System.out.println(list);

return list;

//상품상세페이지에 사용될 상품정보 출력

```
public ProductVO productDetailList(String pnun) {
  return my.selectOne("product.productDetailList", pnun);
}
```

<!-- 상품번호에 해당하는 상품정보를 출력(상품상세페이지에서 사용) -->

```
<select id="productDetailList" parameterType="String" resultType="productVO">
  select * from "PRODUCT" where pnun = #{pnun}
</select>
```

상품 상세 페이지(편의시설 & CCTV에 대한 위치정보 정보요청)



```
//편의시설 & cctv에 대한 위치정보를 담은 배열을 전역변수 선언
csLatitude = []
csLongitude = []

cctvLatitude = []
cctvLongitude = []

index = 0 //cs위치정보를 담은 배열 인덱스 카운트를 위한 변수 선언
index1 = 0 //cctv위치정보를 담은 배열 인덱스 카운트를 위한 변수 선언

$.ajax({
  url: "getFlocation",
  success: function(result) {
    console.log(result)
    //result에 편의시설에 대한 위치정보가 다 들어있음(편의점, cctv) ~> 따라서 분류(편의점, cctv)를 통해 나눠준 후
    for (let i = 0; i < result.length; i++) {
      if(result[i].name == "cs"){
        csLatitude[index] = result[i].latitude
        csLongitude[index] = result[i].longitude
        index++
      }
    }
  }
}) //편의시설 & cctv에 대한 위치정보를 가져올 AJAX 종료부분
```

CCTV및 편의시설 위치정보를 담은 배열

위에 생성한 배열에 하나씩 넣어 추가

컬럼명	#	Type	Type Mod	Not Null
ABC NAME	1	VARCHAR2(100)		[X]
123 LATITUDE	2	FLOAT		[X]
123 LONGITUDE	3	FLOAT		[X]
123 FID	4	NUMBER(38,0)		[X]

ABC NAME	123 LATITUDE	123 LONGITUDE	123 FID
cs	37.554104	126.93576	26
cs	37.5538	126.93631	27
cs	37.553486	126.93738	28
cs	37.553524	126.937614	29
cs	37.554955	126.94151	30
cctv	37.55338	126.937874	31
cctv	37.55292	126.93759	32
cctv	37.555065	126.93608	33
cctv	37.555103	126.937874	34
cctv	37.554626	126.93716	35

```
//편의 시설 테이블에 대한 컨트롤 해주는 컨트롤러
@Controller
public class FController {
  @Autowired
  FacilityDAO dao; //dao싱글톤 객체 생성

  @RequestMapping("getFlocation")
  @ResponseBody //뷰페이지로 넘어가지 않고 바로 실행된 AJAX로 결과값을 리턴
  public List<FacilityVO> select() {
    List<FacilityVO> fList = dao.selectF();
    return fList;
  }
}
```

//name에는 편의점/CCTV가 들어가서 각 위치정보를 리스트로 뽑아줌

<!-- 가데이터로 저장되어져 있는 편의시설, cctv에 대한 위도/경도 추출용 SQL -->
 <select id="select" resultType="fVO">
 select "NAME", "LATITUDE", "LONGTITUDE" from "FACILITY" <!-- L
 </select>

상품 상세 페이지(버튼이벤트에 따른 편의시설& 안전시설 위치정보 지도에 출력)



클릭



```
//편의시설 버튼을 누른 경우
if ($("#b2").click(function() {
    //마커에 이미지 업하기
    var imageSrc = 'https://www.urbanbrush.net/web/wp-content/uploads/edd/2018/02/web-20180209042218b641969.png', // 마커이미지의 주소입니다
    imageSize = new kakao.maps.Size(30, 30), // 마커이미지의 크기입니다
    imageOption = {
        offset : new kakao.maps.Point(27, 69)
    }; // 마커이미지의 옵션입니다. 마커의 좌표와 일치시킬 이미지 안에서의 좌표를 설정합니다
    var markerImage = new kakao.maps.MarkerImage(imageSrc, imageSize, imageOption)

    for (let i = 0; i < csLatitude.length; i++) { //편의시설 배열 길이
        // 마커가 표시될 위치입니다
        markerPosition = new kakao.maps.LatLng(csLatitude[i], csLongitude[i]); //편의시설 위도/경도 정보
        // 마커를 생성합니다
        var marker = new kakao.maps.Marker({
            position : markerPosition,
            image : markerImage
        }); // 마커이미지 설정
        // 마커가 지도 위에 표시되도록 설정합니다
        marker.setMap(map);
    }
})});
```

```
//안전시설 버튼을 누른 경우
if ($("#b3").click(function() {
    //마커에 이미지 업하기
    var imageSrc = 'https://www.urbanbrush.net/web/wp-content/uploads/edd/2018/02/web-20180221070608779012.png', // 마커이미지의 주소입니다
    imageSize = new kakao.maps.Size(30, 30), // 마커이미지의 크기입니다
    imageOption = {
        offset : new kakao.maps.Point(27, 69)
    }; // 마커이미지의 옵션입니다. 마커의 좌표와 일치시킬 이미지 안에서의 좌표를 설정합니다
    var markerImage = new kakao.maps.MarkerImage(
        imageSrc, imageSize, imageOption)

    for (let i = 0; i < cctvLatitude.length; i++) { //안전시설 배열 길이
        // 마커가 표시될 위치입니다
        markerPosition = new kakao.maps.LatLng(
            cctvLatitude[i], cctvLongitude[i]); //안전시설 위도/경도 정보
        // 마커를 생성합니다
        var marker = new kakao.maps.Marker({
            position : markerPosition,
            image : markerImage
        }); // 마커이미지 설정
        // 마커가 지도 위에 표시되도록 설정합니다
        marker.setMap(map);
    }
})});
```

상품 상세 페이지(최근방문페이지 + 장바구니 데이터전달)

```
// 비로그인 상태에서 장바구니 버튼을 눌렀을 때 로그인 페이지로 전환
$("#b4").click(function(){
    alert("로그인이 필요한 서비스입니다.")
    console.log("hid")
    location.href = "ruser_login.jsp"
});
```

localhost:8888 내용:
로그인이 필요한 서비스입니다.

확인

로그인

아이디

비밀번호

로그인

회원가입

장바구니 버튼

장바구니 버튼을 누른 경우

1. (기본적으로)상품페이지로 넘어오면 pnum과 세션처리된 아이디, like에 0을 마이페이지 컨트롤러로 보내준다

2. (상품담기버튼 누른 경우)pnum , 세션처리된 아이디, like에 1을 담아 보내준다

*/

var userid = '\${userid}' // \${userid}라고 하면 안넘어감..... -> userid는 문자열인데 param.userid로 안하는 이유? -> getAttribute이기 때문에

var liked = 0 -> 상품 담기 구분을 위한 값

//상품상세페이지로 넘어오면 자동적으로 최근방문페이지 카운트를 위해 AJAX를 이용하여 마이페이지 컨트롤러로 정보들을 넘겨준다

\$.ajax({

url: "my_insert",

data: {

pnum : pnum,

userid : userid,

liked : liked,

},

success: function() {

console.log("최근방문페이지")

}

}

상품상세페이지로 넘어오면

상품번호, 사용자 아이디, 상품 담기 유무판단 데이터를 넘겨줌

//장바구니 버튼을 누른 경우

//AJAX를 통해 해당 아이디와 상품번호, like=1 보내줌 -> 마이페이지 컨트롤러로 -> insert SQL수행

\$("#b1").click(function() {

alert("장바구니에 담겼습니다")

liked = 1

\$.ajax({

url: "basket",

data: {

pnum : pnum,

userid : userid,

liked : liked,

},

success: function() {

console.log("찜하기")

}

}

상품 담기 버튼을 누른 경우 상품 담기 구분 값을 1로 바꿔서

가상주소로 보내줌

상품담기

localhost:8888 내용:

장바구니에 담겼습니다

확인

04. 프로젝트 소감 – 신화원

지난 번 프로젝트가 MVC1패턴을 이용해 jsp사이에서만 놀았다면, 2차 프로젝트에서는 SPRING MVC2패턴을 사용하여 2번째 프로젝트를 진행했다. 처음에는 MVC2 패턴에 익숙해 지기 까지 너무 많은 시간이 걸렸다. 확실히 익숙해지면 편하다고 느껴지지만 그 단계로 가기 까지 힘들었다. 이번에 내가 맡은 역할은 메인 페이지 구성과 상품목록페이지 & 상품 상세 페이지 였다. 메인페이지 구성을 위해 처음으로 부트스트랩을 이용해서 틀을 잡는데 큰 도움을 받았다. 부트스트랩을 사용하며 어려운 점이 있었다면, 직접 만든 코드가 아닌 남이 작성한 코드다 보니 수정하는 것에 있어서 어려움이 있었다. 또한, 1차 프로젝트에는 데이터를 가져오거나 작업을 수행할 때 항상 다른 페이지로 넘어가서 해결을 했다면 이번에는 AJAX를 이용해 페이지 전환 없이도 해결할 수 있다는 걸 배웠고 대부분 AJAX를 통해 비동기통신을 했기 때문에 상당한 편리함을 느낄 수 있었다.

또한, 페이징처리 및 카카오맵 API를 활용했다. 페이징처리를 통해 많은 양의 데이터를 좀 더 깔끔하게 정렬할 수 있었다. 페이징 처리에 있어서는 오픈소스를 이용하여 수정을 통해 의도에 맞게 수정을 하여 사용했다. 또한 카카오맵 API를 활용하여 지도에 해당 데이터를 찍어줄 수 있어 시각적 표현에 있어서 큰 도움을 받았다. 또한 다양한 종류의 예제 및 도움말이 있어서 차근차근 읽어 가면서 프로젝트 방향에 맞게 활용할 수 있어서 좋았다.

이번 프로젝트를 통해 어려웠던&아쉬웠던 점에 대해 얘기를 하면, 처음으로 조장을 하다 보니 조원들 이끌 리더십이 부족했고 좀 더 MVC2에 대한 이해도가 있었더라면 조원들 오류를 더 잘 잡아줄 수 있었을 텐데 하는 아쉬움이 있었다. 또한, 원래 수행하려고 했던 기능 중 '검색기능'을 해결하지 못하고 끝난 게 조금 아쉽지만 다음 프로젝트에는 반드시 추가할 생각이다. 마지막으로, 적극적으로 팀원과의 의사소통을 못해서 아쉬웠다. 조장으로서 적극적으로 팀원과의 소통을 했어야 하는데 그러지 못한 점 반성하고 다음 프로젝트에는 노력을 할 생각이다. 그래도 전체적으로 배운 내용에 대해서 잘 적용했고 끝난 거 같아서 좋았다.