1. 팀 소개

1.1 팀명 : 집동산

1.2 팀원 및 역할

빅데이터 전문가 양성과정 B반의 2조

김한얼(조장, 시연) earlthinking@gmail.com : 서버 개발 및 일정 관리

송재환(발표) : 다음 지도 적용 및 웹 개발

김지승 angguri@naver.com : 알고리즘 개발 및 DB 설계

박정훈 : 알고리즘 개발 및 다음 지도 적용

1. 프로젝트 소개

2.1 프로젝트 명

당신의 집

부동산 매물 거래시, 부동산 매물과 사용자 기호의 매칭율을 제안

2.2 프로젝트 개요

집동산 서비스는 사용자가 원하는 위치정보(주소) 주변의 환경적인 요소(대중교통, 편의점, 대형마트, 식당, 공원/체육시설, 병원)의 유무, 사용자의 가중치, 거리를 조합하여 선택한 부동산의 환경요소의 가치를 평가해줌으로써 의사결정을 지원한다.

2.3 프로젝트 목적

부동산 매물 선택시, 편의 시설에 대한 안내가

직관적이지 않음. 부동산 결정에 도움이 되는 정보를 제공함으로써, 사용자가 비슷한 조건의 매물을 직관적으로 비교할 수 있도록 도움주기 위함.

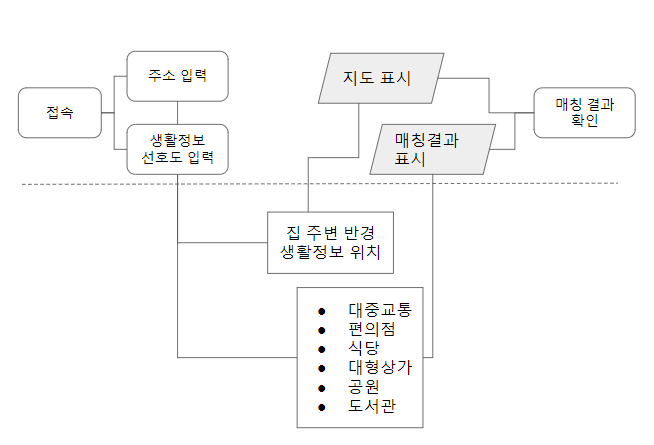
1. 서비스 소개

3.1 개요

부동산 시장은 공인중개사를 통해서만 거래할 수 있었던 대표적인 비대칭적 정보를 중개하여 수익을 올리는 상품이다. 직방, 다방과 같은 서비스는 웹과 모바일을 활용하여 소비자들이 다양한 부동산 매물 정보와 판매자에게 직접 접근할 수 있게 되어 부동산 거래에 필요한 시간적, 금전적 비용을 줄여줘 많은 호응을 얻었다. 그러나 소비자들은 부동산 거래 시 부동산 자체만의 조건을 따지는 것이 아닌 부동산 주변의 환경적인 요소를 함께 고려하는 과정까지 지원해주는 서비스는 없는 것이 현실이다.

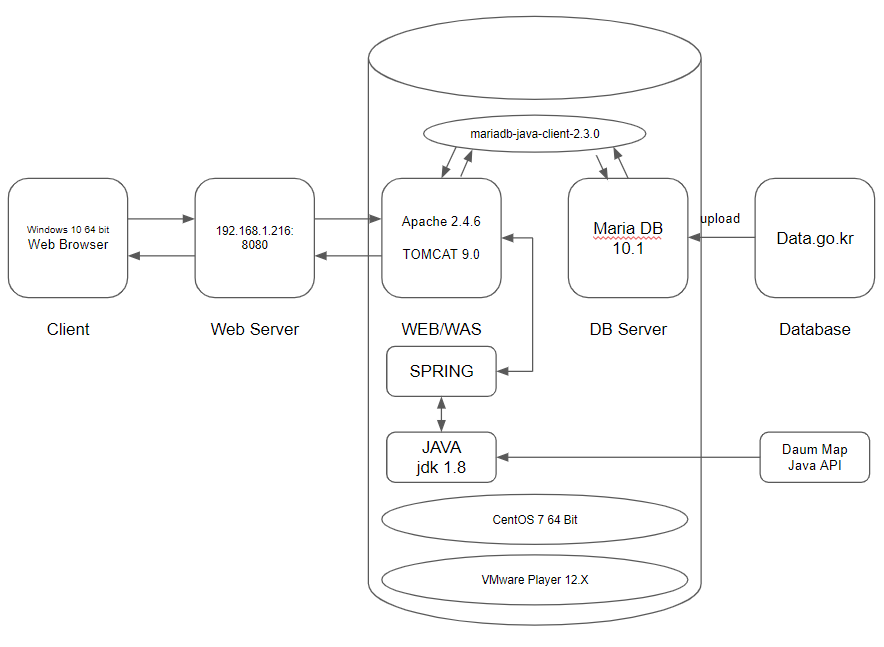
3.2 서비스 흐름

사용자는 ‘당신의 집’에 접속하여 주소와 생활정보 별 선호도를 입력하여 결과를 산출한다. 사용자는 생활요소의 위치를 지도의 반경별로 파악할 수 있으며 위치에 따른 선호도가 반영된 매칭결과를 점수로 확인할 수 있다.



1. 개발 내용

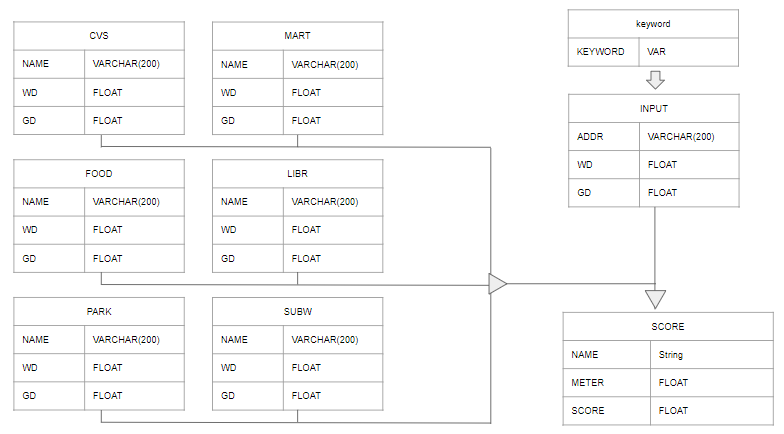
4.1 구성



4.2 소프트웨어 스택

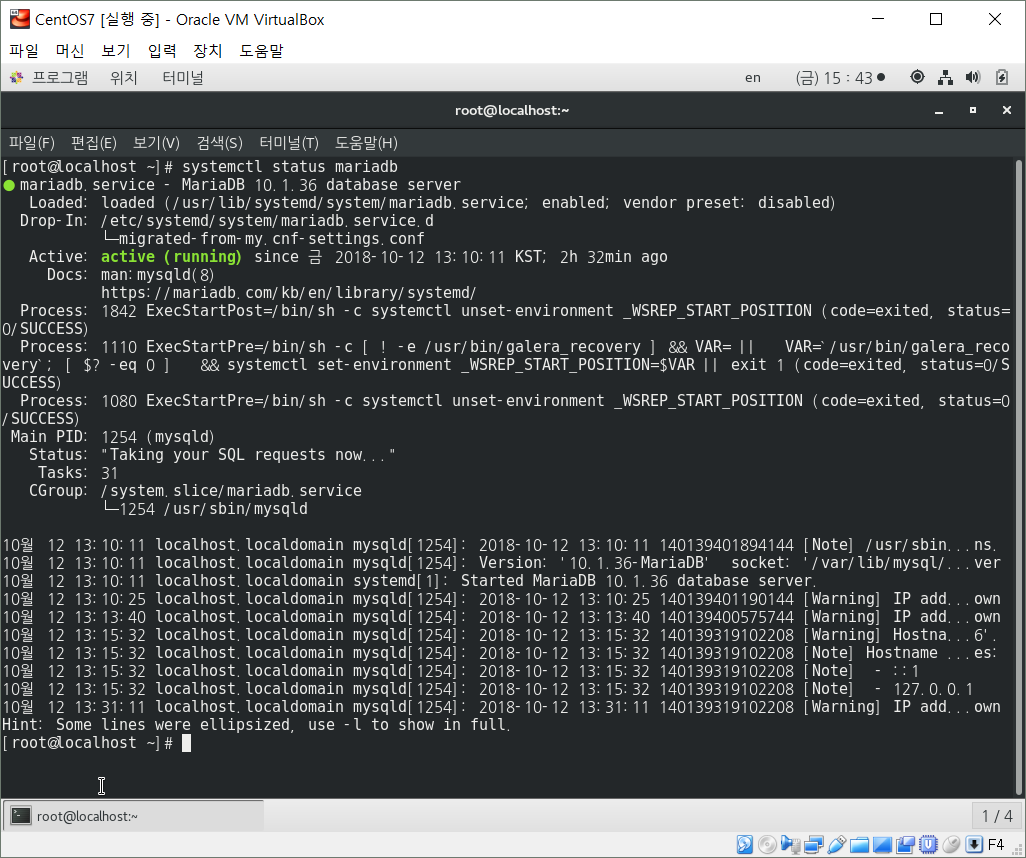
|  |  |  |
| --- | --- | --- |
| 구분 | 이름 | 버전 |
| DBMS | MariaDB | 10.1 |
| Language | Java | 8 |
| WAS | Apache Tomcat | 9.0.12 |
| IDE | Spring Tool Suite | 3.9.5 |
| Database Management | Heidi SQL | 9.4 |
| OS | CentOS | 7 |
| Virtual Machine | VMWare | 12.1 |

4.3 데이터 구조

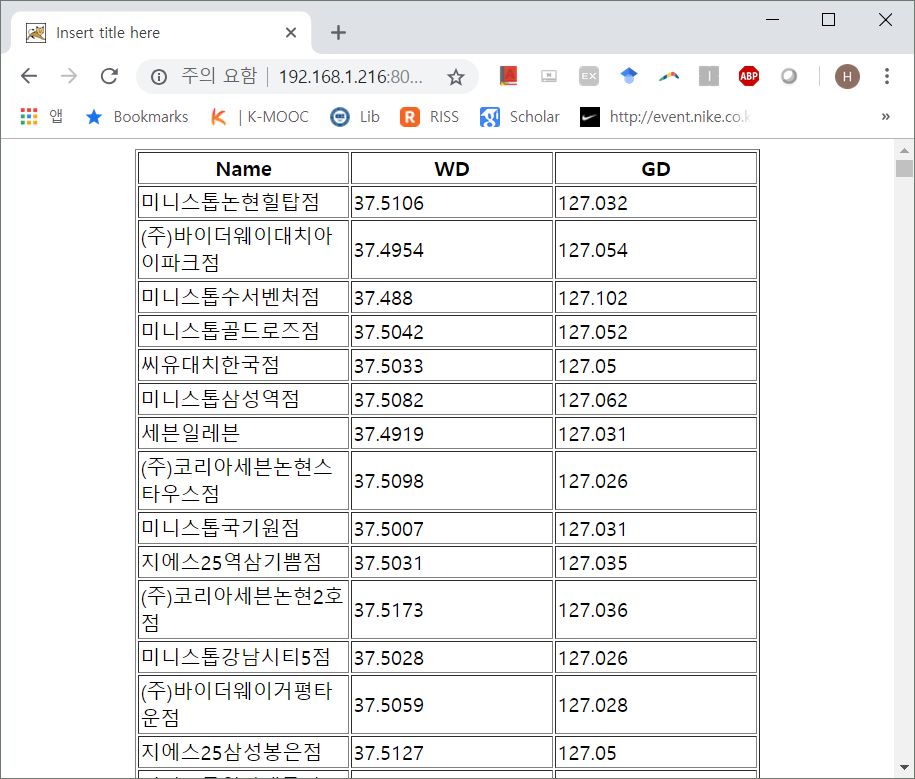
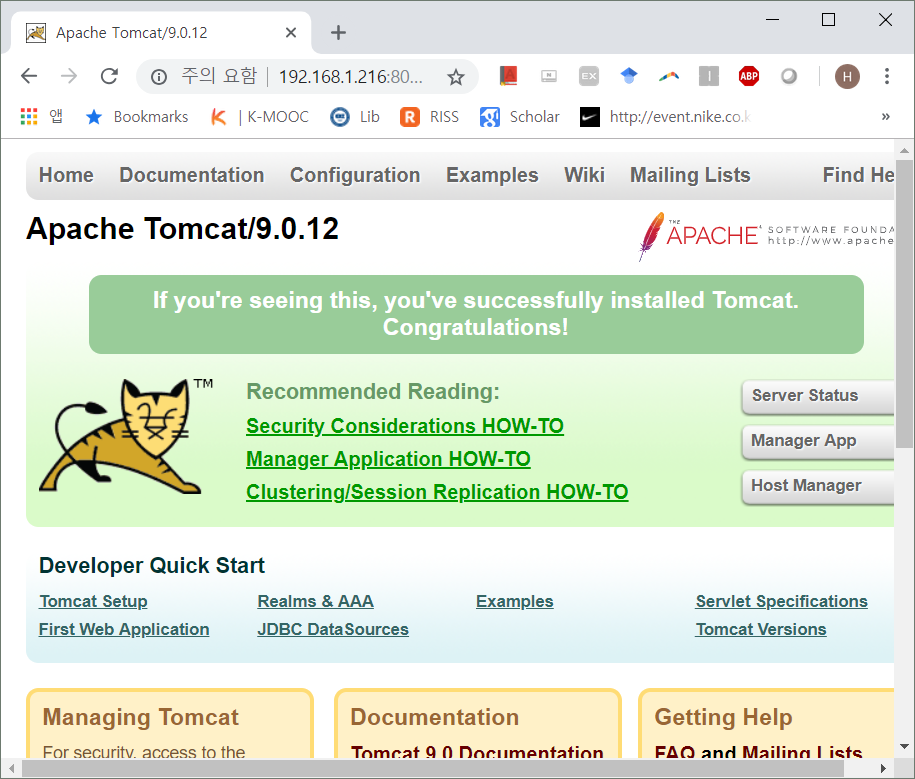


4.3 기능 별 산출 결과

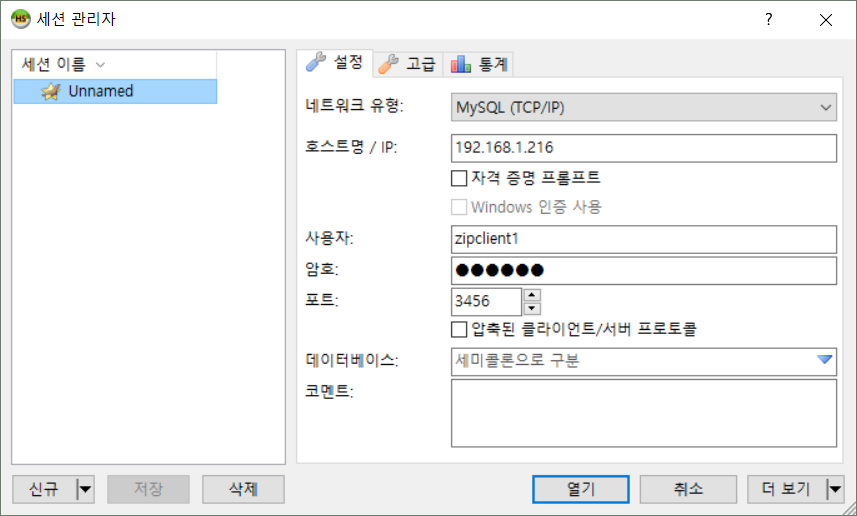
4.3.1 MariaDB

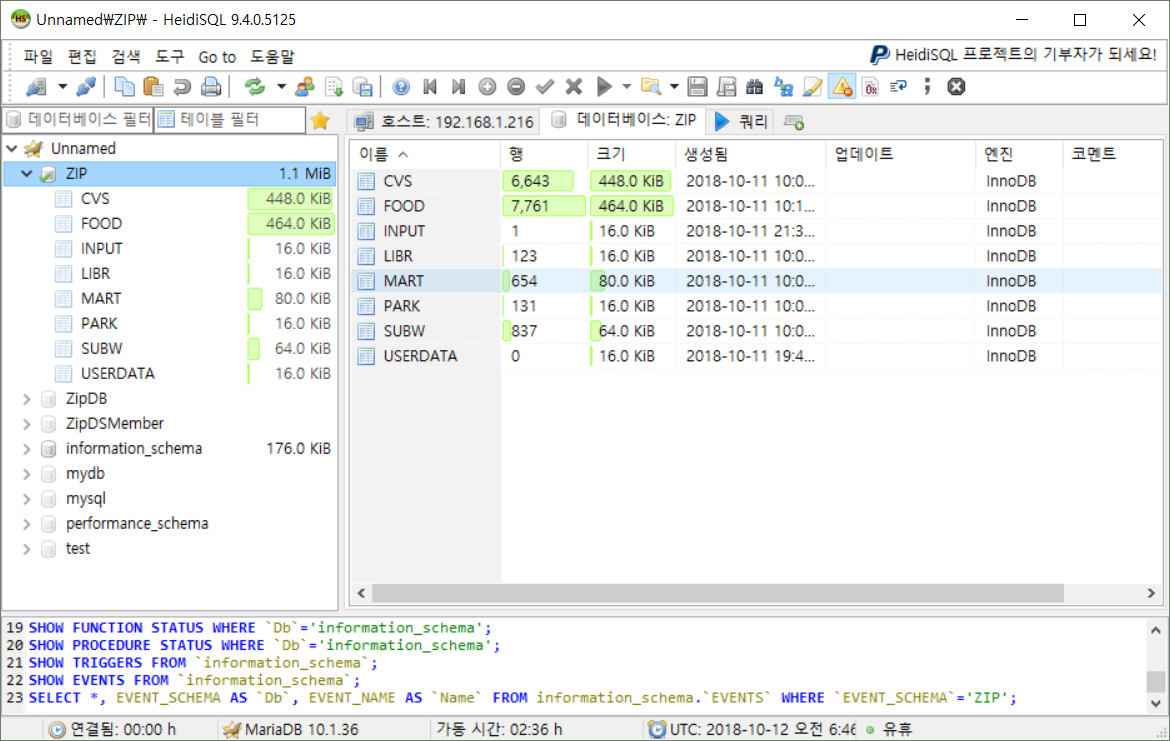


4.3.2 Apache Tomcat과 MariaDB 연결 확인



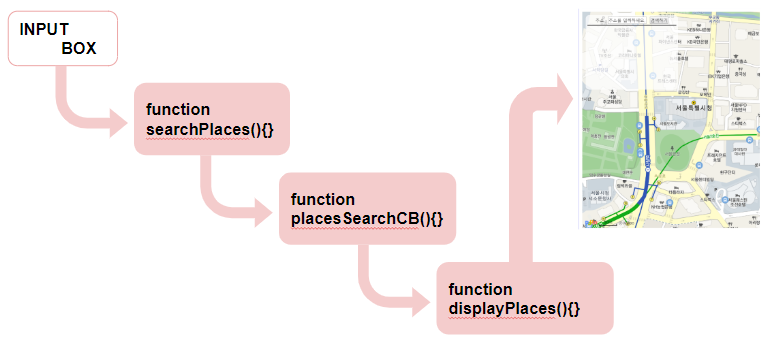
4.3.3 Heidi SQL에서 DB Server 접속 성공





4.4 알고리즘 설명

4.4.1 다음 지도 API 적용



4.4.2.1. function searchPlaces : 키워드 검색을 요청하는 함수입니다.

4.4.2.2. function placesSearchCB : 장소검색이 완료됐을 때 호출되는 콜백함수 입니다.(if문을 써서 검색결과가 적합 여부 판별)

4.4.2.3. function displayPlaces : 검색 결과 목록과 마커를 표출하는 함수입니다.

4.4.2.4. function getListItem : 검색결과 항목을 Element로 반환하는 함수입니다.

4.4.2.5. function addMarker : 마커를 생성하고 지도 위에 마커를 표시하는 함수입니다.

4.4.2.6. function removeMarker : 지도 위에 표시되고 있는 마커를 모두 제거합니다.

4.4.2.7. function displayPagination : 검색결과 목록 하단에 페이지번호를 표시는 함수입니다.

4.4.2.8. function displayInfowindow : 검색결과 목록 또는 마커를 클릭했을 때 호출되는 함수입니다. 인포윈도우에 장소명을 표시합니다.

4.4.2 SQL 적용

서버 명 zipdongsan

ZIP 아래

<이름> <위도> <경도>

VARCHAR(100) FLOAT FLOAT

CVS 편의점 CU한양대점 37.XXXX 127.XXXXX

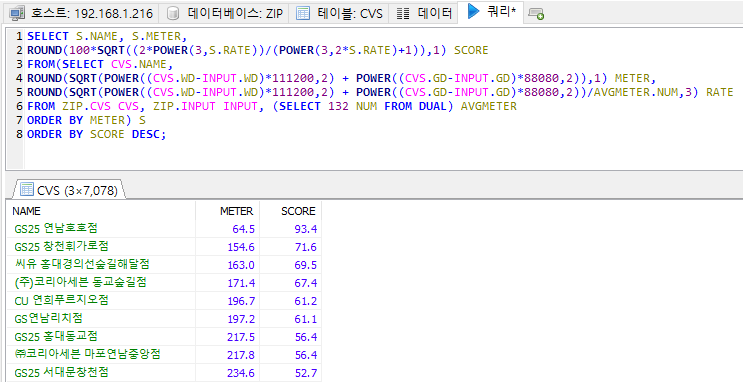
FOOD 식당

LIBR 도서관

MART 마트

PARK 공원

SUBW 지하철



편의점 데이터 인풋데이터 편의점 평균거리

<CVS> <INPUT> <AVGMETER> <- DB자료

| | |

ㄴ--------------------ㅜ-----------------------|

ㅣ

<S> ( 편의점 이름, 인풋데이터와 편의점 사이의 거리, 거리비)

|

출력 ( 편의점 이름, 인풋데이터와 편의점 사이의 거리, 점수)

이렇게 인풋값과 점수가 높은 순으로 정렬해서 상위 3개만 출력함

* 점수 산정 절차

1. 입력받은 주소와 데이터 베이스에 입력되어 있는 좌표들 사이의 거리를 구함
2. 6대 항목 별 서울시 평균 거리를 구함
3. 거리를 평균거리로 나눠서 거리 비를 구함
4. 거리 비`
5. 위도차(m) = (데이터 위도 - 인풋 위도)\*111200

경도차(m) = (데이터 경도 - 인풋 경도)\*88080

* 111200, 88080은 서울시 평균 위치를 기준으로 위도 경도를 m단위로 바꿔주는 상수

거리 = sqrt(위도차^2+경도차^2)

2) 평균거리 = sqrt( S / N \*)

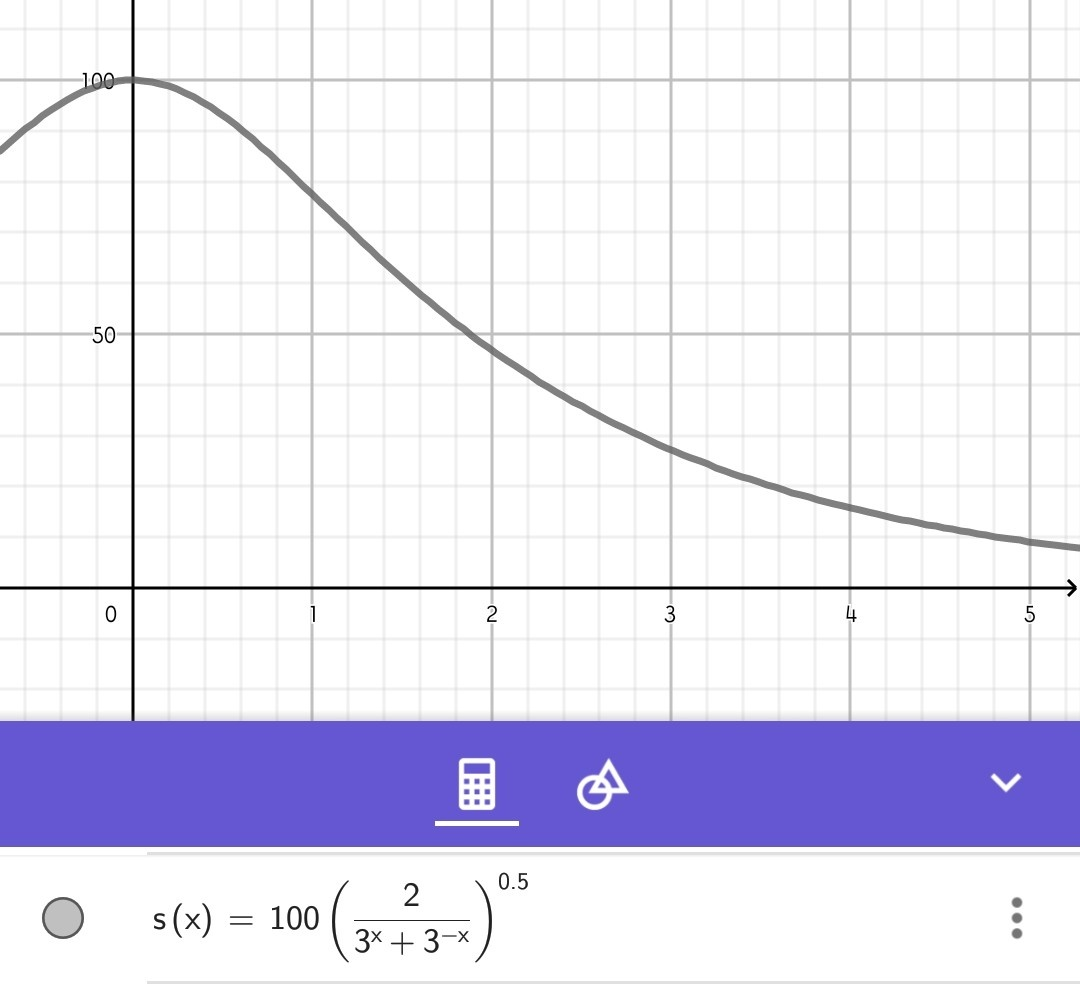
S=서울의 전체 면적(약 600km^2)

N=데이터 베이스에 있는 데이터 개수

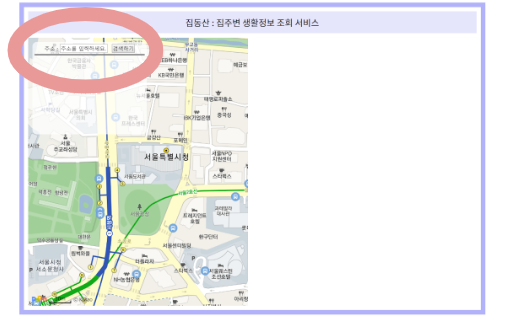
3) 거리 비 = 거리 / 평균거리

4) Score(x) = sqrt(2/(3^x + 3^(-x)) \* 100

x = 거리 비

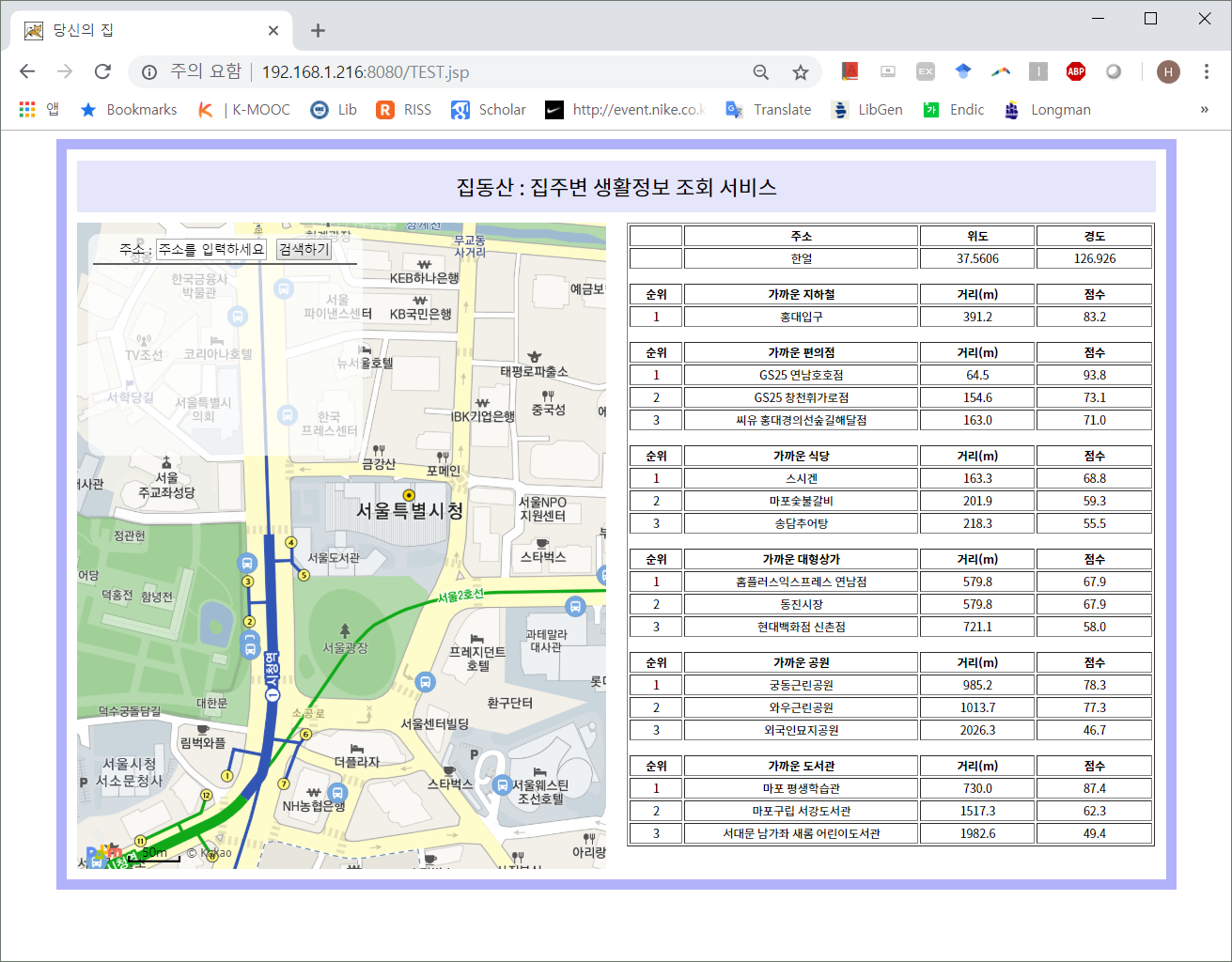


1. 결과
   1. 시연
      1. 검색





* + 1. 결과



1. 토의
   1. 한계점

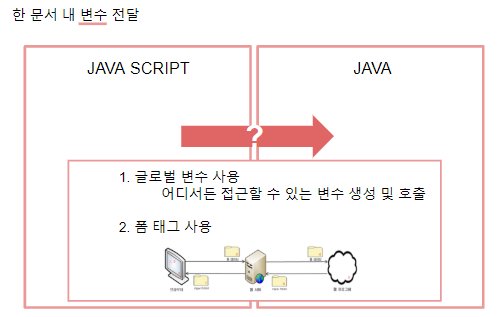
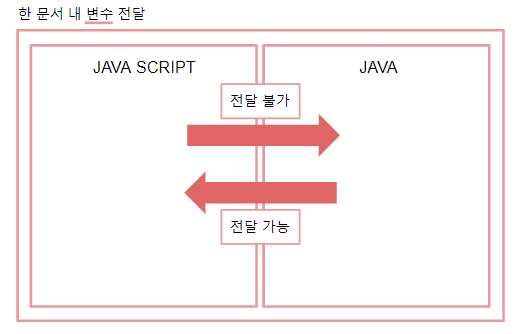
지도 API (정훈, 재환)

- Naver사이트의 api를 쓰기에는 초보자가 이해하기에 어려움.

- Google api를 쓰려고 했으나 네트워크의 응답시간이 너무 오래걸림

- Daum api가 인터페이스도 간편하고 코딩을 모르는 사람들도 쉽게 사용 할 수 있게 되어 있어서 daum api를 사용하게 되었음

* 1. 개선 계획



- JAVA SCRIPT와 JAVA 코드 간 변수 이동 구현, 구동 원활화

- 지역변수와 전역변수에 대한 이해 필요

- 사용자의 요소별 가중치 선택 기능 추가하여, 매칭 점수 신뢰도 확보

- 매칭 결과값 직관적으로 표현

- 사용된 데이터의 정확도 및 측정 요소 재검증

- 앱 형식으로 발전시켜, 사용자 접근성 및 이용 용이성 증가