

FUEL PLEASE

Planning

API

Design

*Detail
of
Function*

Technology

Evaluation

Planning

계획

**Team
&
Project**

**Division
Of
Work**

**Project
Intention**

Tools

Team & Project

Why?

주유나 충전할 장소를 찾으며 생각

Team Name

주유해주유~

Project Name

FuelPlease

Why?

기억하기 쉽고
팀의 목적을 분명히 할

Project Intention

서울특별시에 존재하는 모든 주유소와 LPG, 전
기차 충전소의 정보와 위치를 쉽게 확인하기 위함

개인 차량과 전기차 사용이 많아짐에 따라 빠른
위치확인의 필요성 느낌

주유소나 충전소에 대한 이용자들의 의견을 통해
더 좋은 서비스를 원하는 이용자의 의견 정리 필요





FuelPlease 총괄

로그인, 회원가입 기능 구현

즐겨찾기 서비스 기능 구현

카카오 주소검색 API 응용

주유소, 전기차충전소 API 응용

(주)석유동향 Opinet API 응용

김지원

카카오 오픈 목업 제작

BootStrap ver.5 응용

메인페이지 구현

로그인, 회원가입, 게시판 디자인 구현

Header, Footer 디자인 구현

시그니처 로고 제작

-제원예대 23학번 김정아

최지혁

정보게시판 서비스 구현

카카오 키워드 검색 API 응용

LPG충전소 API 응용

(주)석유동향 Opinet API 응용

최저가 검색 기능 구현

DB 제작

문창주

카카오 오픈 목업 제작

Header 메뉴 기능 구현

로그아웃, 회원탈퇴 기능 구현

가계부 서비스 기능 구현

가계부 페이지 디자인 구현

DB 제작

개발 구분	세부 항목	5월												6월				
		16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	1
분석설계	아이템구상																	
	화면 목업제작																	
	DB설계																	
	API찾기																	
디자인	홈 페이지																	
	로그인 페이지																	
	회원가입 페이지																	
	게시판 페이지																	
	검색 페이지																	
	마이 페이지																	
	자계부 페이지																	
개발	DB제작																	
	사용자 관리																	
	게시판 관리																	
	API 저장작업																	
	API 사용작업																	
	자계부 관리																	
	카카오 지도 작업																	
테스트	통합 테스트																	
자료작성	보고서 및 PPT																	

Tools

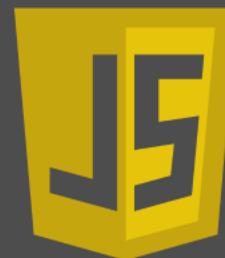


MyBatis

HTML

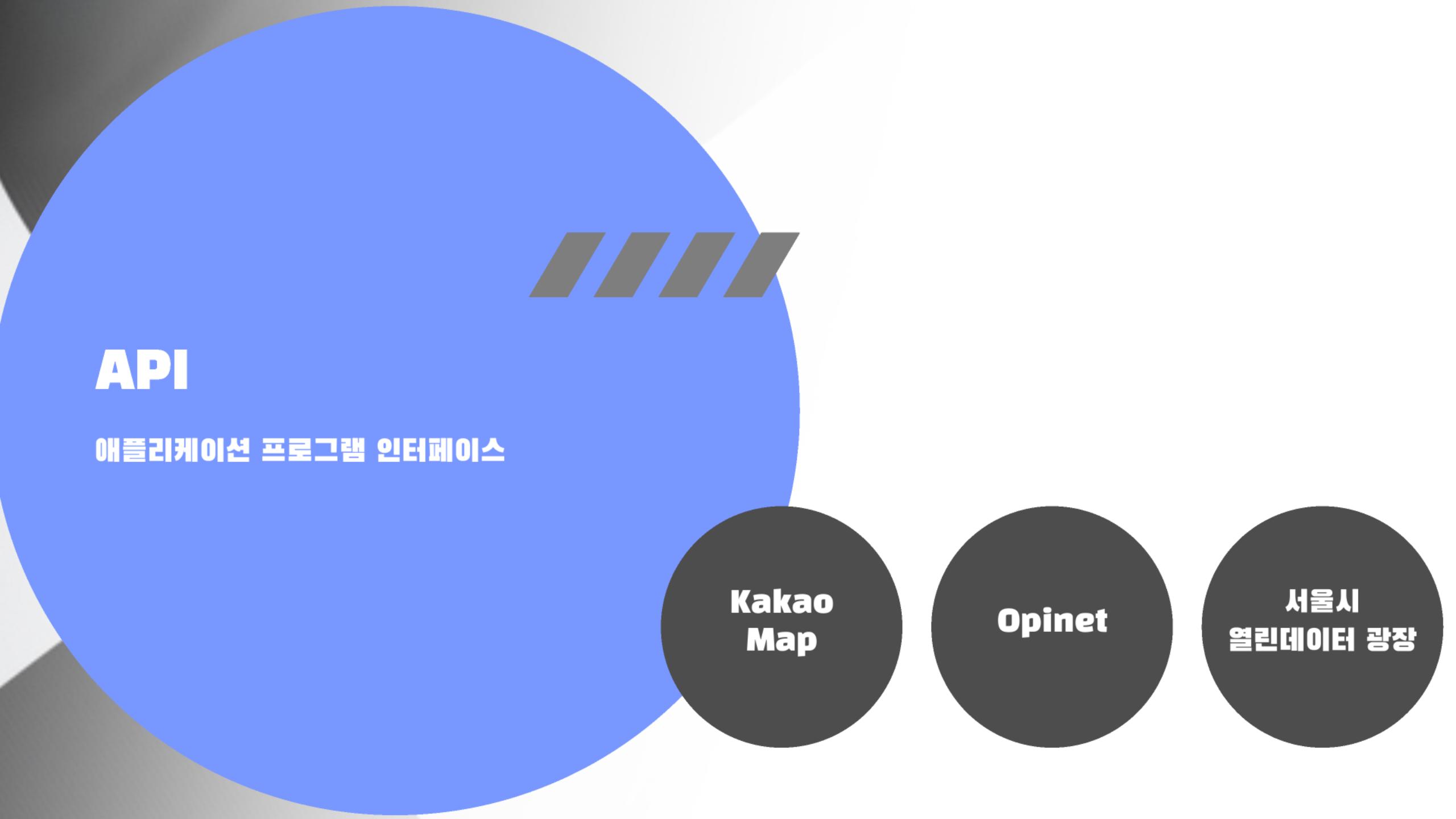


JS



CSS





API

애플리케이션 프로그램 인터페이스

Kakao
Map

Opinet

서울시
열린데이터 광장

Kakao Map



**카카오 Map Api 의
주소검색, 키워드 검색 사용**

<https://apis.map.kakao.com/web/sample/addr2coord/>

Opinet

(주)석유동향- 오피넷의 평균유가 정보, 최저가 주유소 API 사용

<https://www.opinet.co.kr/user/custapi/custApilInfo.do>

■ 지역별 최저가 주유소(TOP20)	
기본정보	http://www.opinet.co.kr/api/lowTop10.do - 전국 또는 지역별 최저가 주유소 TOP20
요청 파라미터	
Element	Description
code	필수 공사에서 부여한 키(key) 정보
out	필수 정보 형식을 정의 한다(xml/json)
prodcd	필수 제품구분(화유B027, 청유D047, 고급화유B034, 살내화유C004, 저온차유K015)
area	선택 지역구분 (미입력시 전국, 시도코드(2자리)-해당시도, 기준, 시군코드(4자리)-해당시군 기준)
cnt	선택 최저가순 결과 갯수 (1 ~ 20 사이) 숫자 입력, 미입력시 기본값 10개
변수값	
Element	Description
UNI_ID	주유소코드
PRICE	판매가격
POLL_DIV_CD	상표ISKE-SK에너지, GSC-GS칼텍스, HDO현대오일뱅크, SOLS-OIL, RTO-자영밀집, RTX-고속도로밀집, ENO-농협밀집, ETC-자기상표, E10- E1, SKG-SK기스)
OS_NM	상호
VAN_ADR	지번주소
NEW_ADR	도로명주소
GIS_X_COOR	GIS X좌표(KATEC)
GIS_Y_COOR	GIS Y좌표(KATEC)
사용 예시	
REQUEST	http://www.opinet.co.kr/api/lowTop10.do?out=xml&code=XXXXXX&prodcd=B027&area=0101&cnt=2
RESPONSE	<?xml version="1.0" encoding="UTF-8"?> <lowTop10> <code>UN-A001000</code> <prodcd>B027</prodcd> <area>0101</area> <cnt>2</cnt> <list> <item> <UNI_ID>UN-A001000</UNI_ID> <PRICE>1330</PRICE> <POLL_DIV_CD>RTX</POLL_DIV_CD> <OS_NM>RTX</OS_NM> <VAN_ADR>서울특별시 강남구 테헤란로 118 4층</VAN_ADR> <NEW_ADR>서울특별시 강남구 테헤란로 118 4층</NEW_ADR> <GIS_X_COOR>281595.200000</GIS_X_COOR> <GIS_Y_COOR>319897.000000</GIS_Y_COOR> </item> <item> <UNI_ID>UN-A001000</UNI_ID> <PRICE>1330</PRICE>

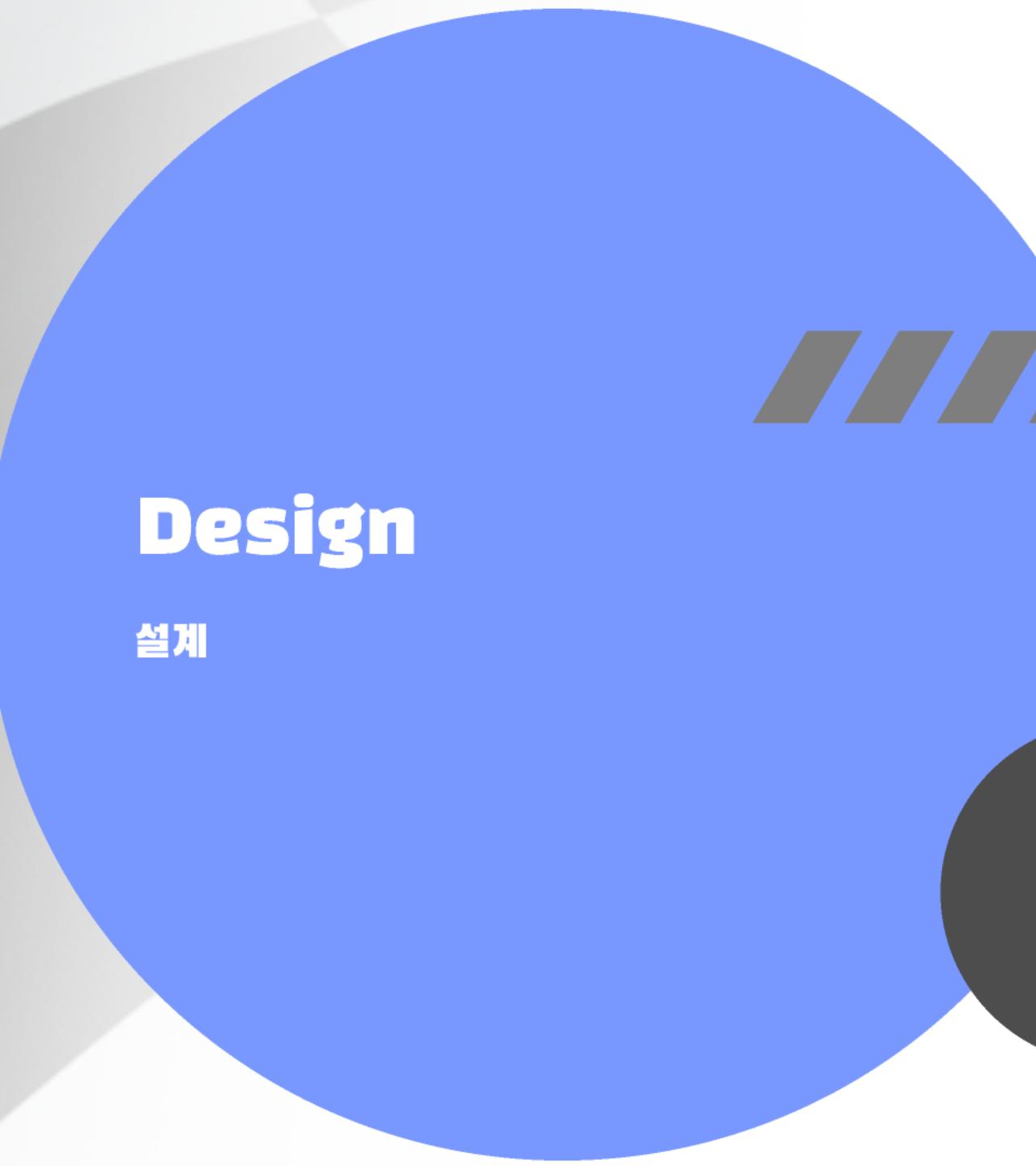
서울시 열린데이터 광장

The screenshot shows a search result for "서울시 전기차 충전소 정보" (Seoul City Electric Vehicle Charging Station Information). The results are filtered by "환경" (Environment). The first result is a table titled "데이터 정보" (Data Information) containing the following details:

공개일자	2023/05/08	최신수정일자	2023/05/08
경신주기	월간	분류	환경
원본시스템		제작국자	서울특별시
제공기관	서울특별시	제공부서	기후환경부 친환경차정책과
담당자	안정화 (02-2133-9777)		
원본형태	File	제작자/작권자	없음
라이선스		저작권자표시(CC BY) 이용이나 변경 및 2차적 저작물의 작성을 포함한 자유이용을 허락합니다.	
관련태그	전기차, 충전소, 충전소 목록, 친환경, 기후환경		

**서울시 열린데이터 광장의
주유소 정보, 전기차 충전소 정보, LPG 충전소
정보 API 사용**

<https://data.seoul.go.kr/>



Design

설계

DB 설계

UI 설계

기능 설계

DB 설계

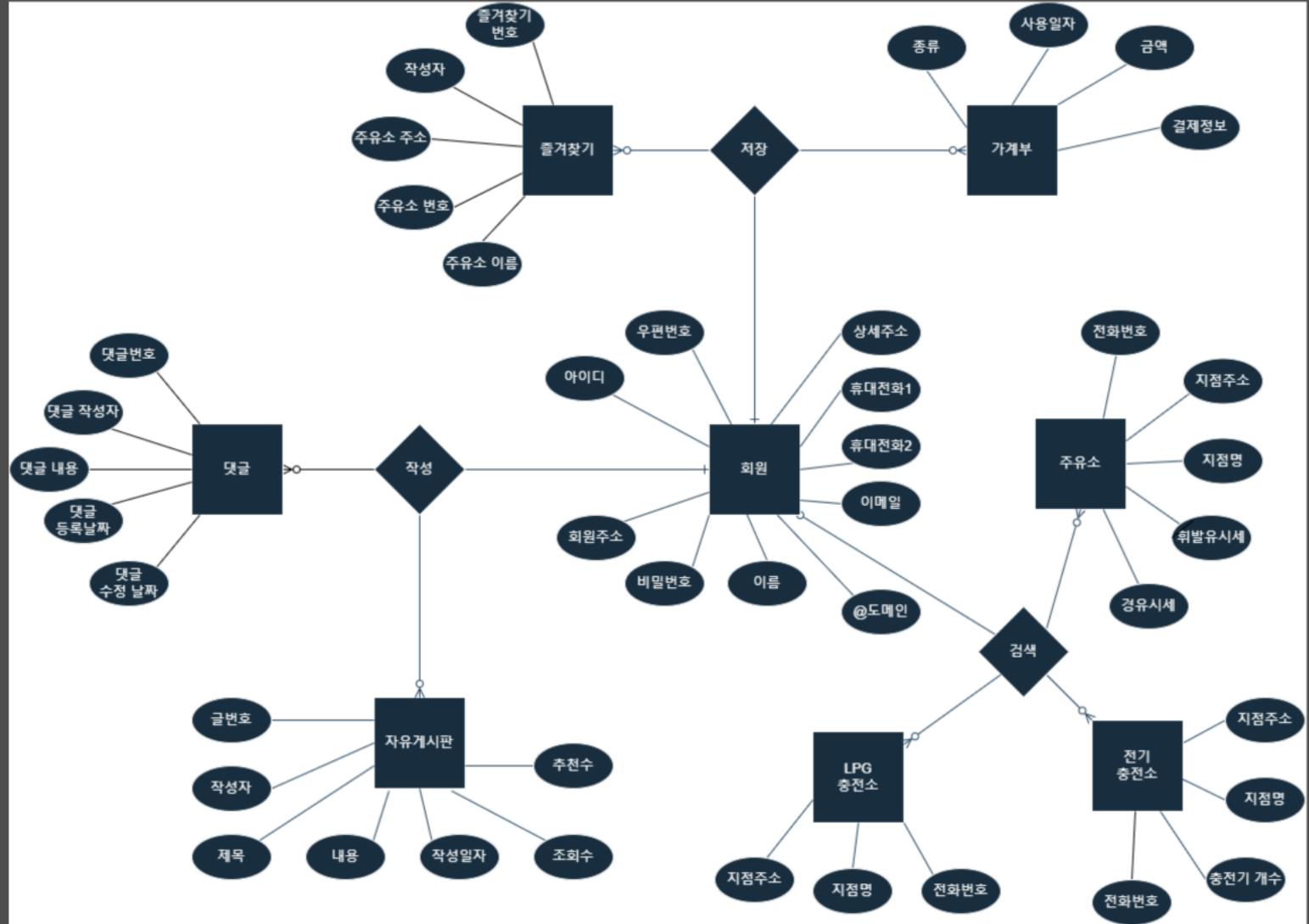
Modeling

Conceptual
Modeling

Logical Modeling

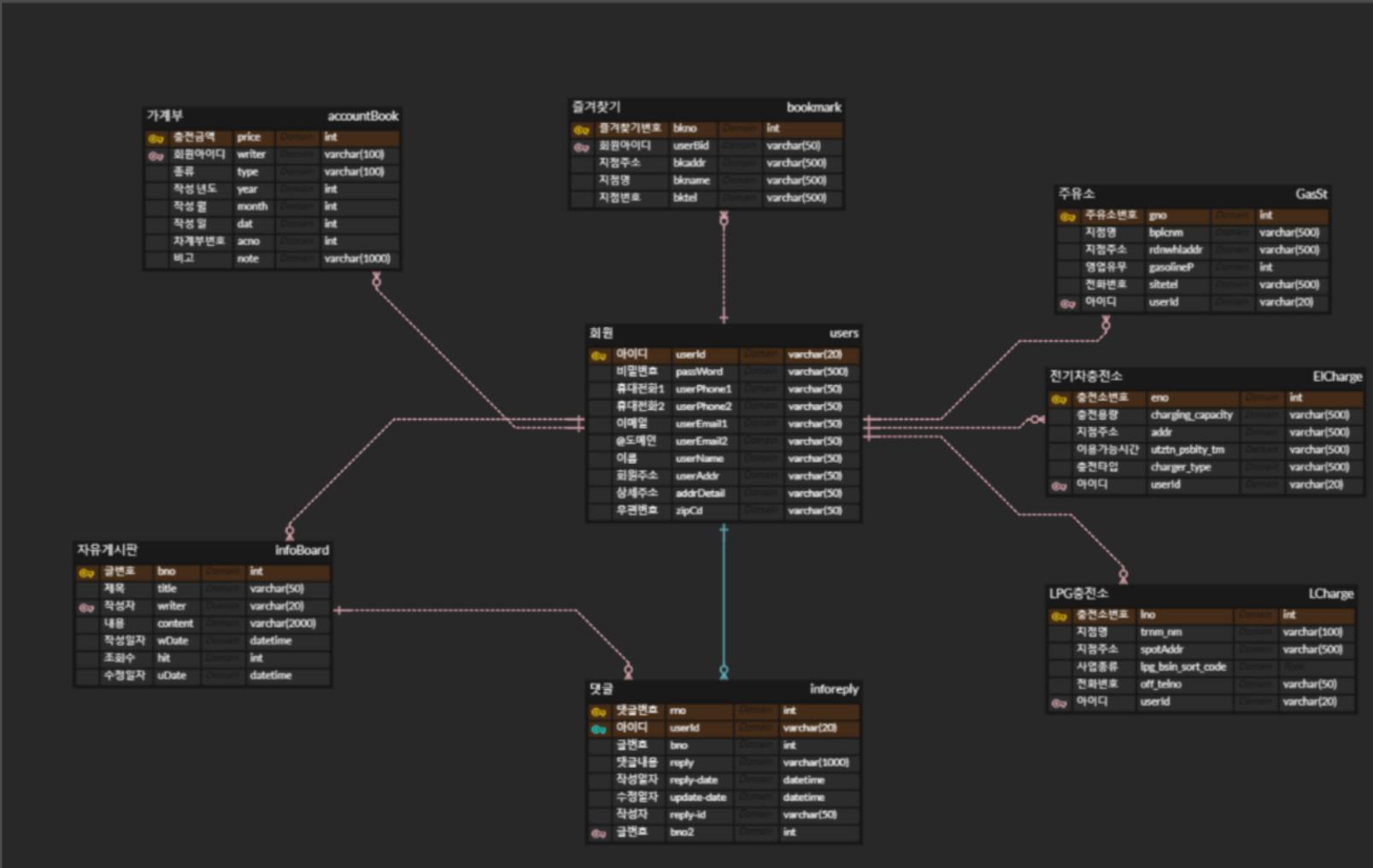
Physical Modeling

Conceptual - ERD



NEXT

Logical - ERD



NEXT

Physical Desgin

Users

InfoBoard

InfoReply

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
rno	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>					<input checked="" type="checkbox"/>		
bno	INT		<input type="checkbox"/>							NULL
reply	VARCHAR(1000)		<input type="checkbox"/>							NULL
reply_id	VARCHAR(50)		<input type="checkbox"/>							NULL
reply_pw	VARCHAR(50)		<input type="checkbox"/>							NULL
reply_date	DATETIME		<input type="checkbox"/>							CURRENT_TIMESTAMP
update_date	DATETIME		<input type="checkbox"/>							NULL

GasolinelInfo

Eleclnfo

LPGInfo

Bookmark

CarAccount

UI 설계

Home
page

Mock Up

Login page
Join page

MyPage
Car Account Book
BookMark

InfoBoard page
Reply page

Keyword Search
Search Place

Mapview

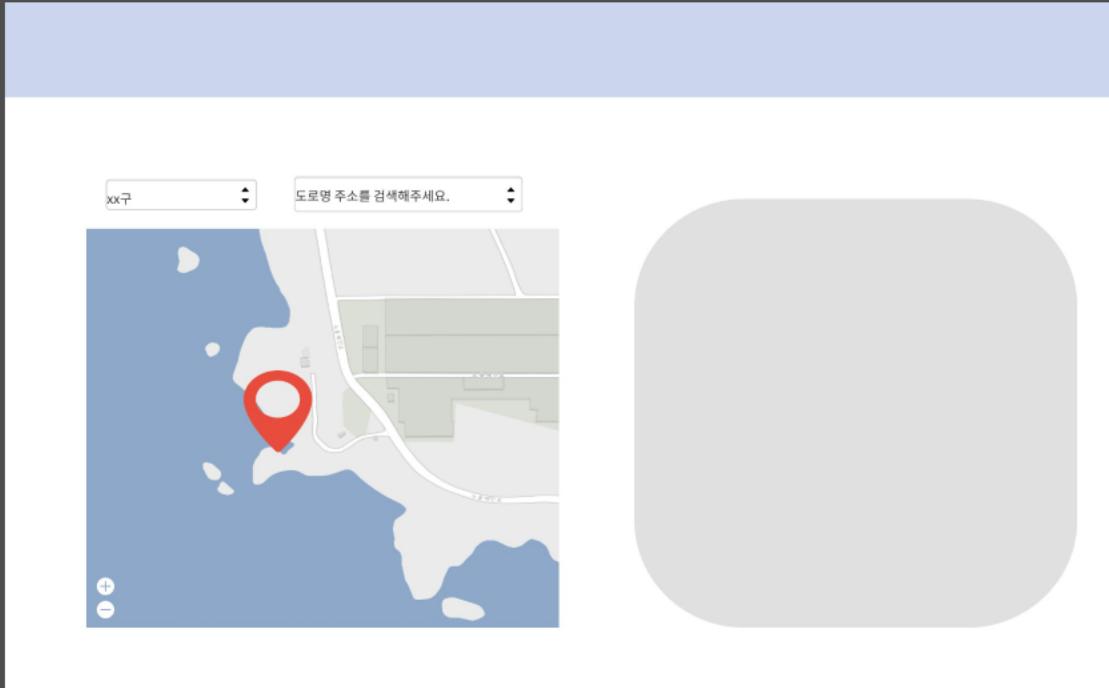
Mock up

Home Page



Map Page

Map Page



Board Page

Board Page

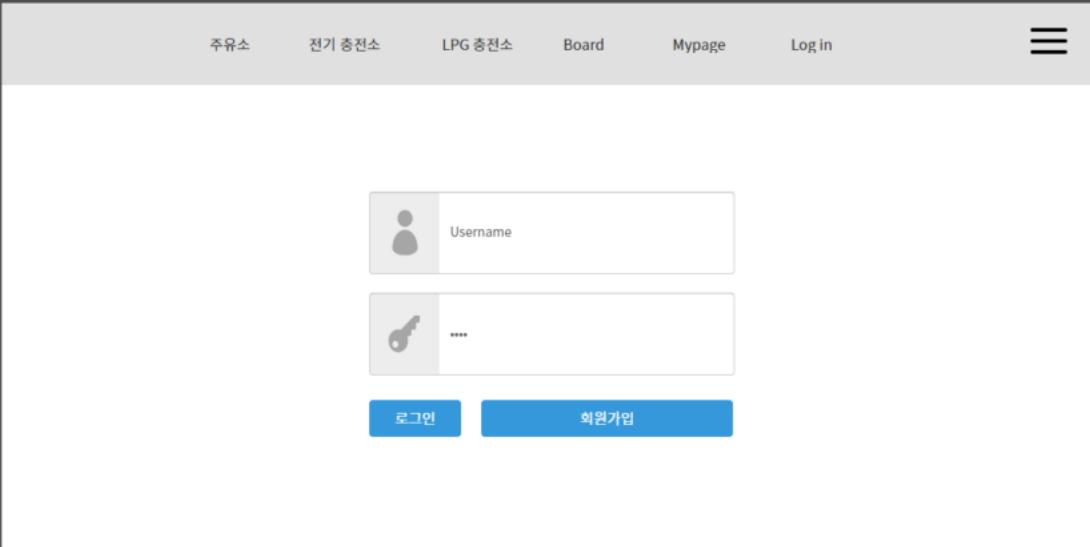
The screenshot shows a web-based application interface. At the top, there is a navigation bar with links: 주유소 (Gas Station), 전기 충전소 (Electric Charging Station), LPG 충전소 (LPG Charging Station), Board, Mypage, and Log in. To the right of the navigation bar is a three-line menu icon. Below the navigation bar is a search bar containing the placeholder text "내용을 입력해주세요" (Please enter content) and a blue "검색" (Search) button. The main content area features a table with three columns: No., Name, and Value. There are two rows of data in the table:

No.	Name	Value
1		
2		

At the bottom of the table area is a blue "글쓰기" (Write Article) button.

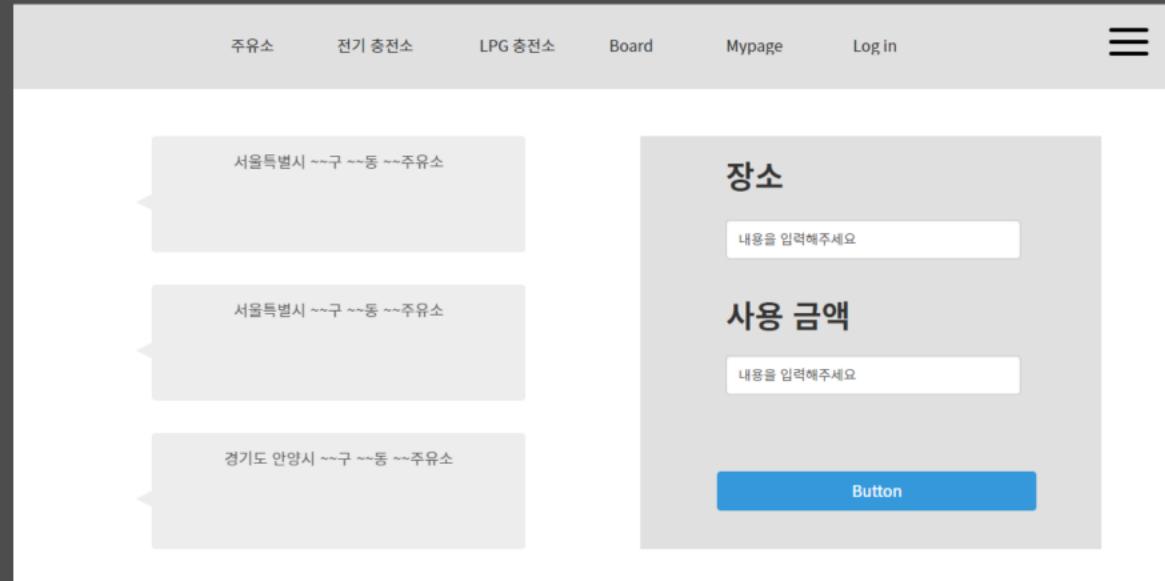
Login Page

Login Page



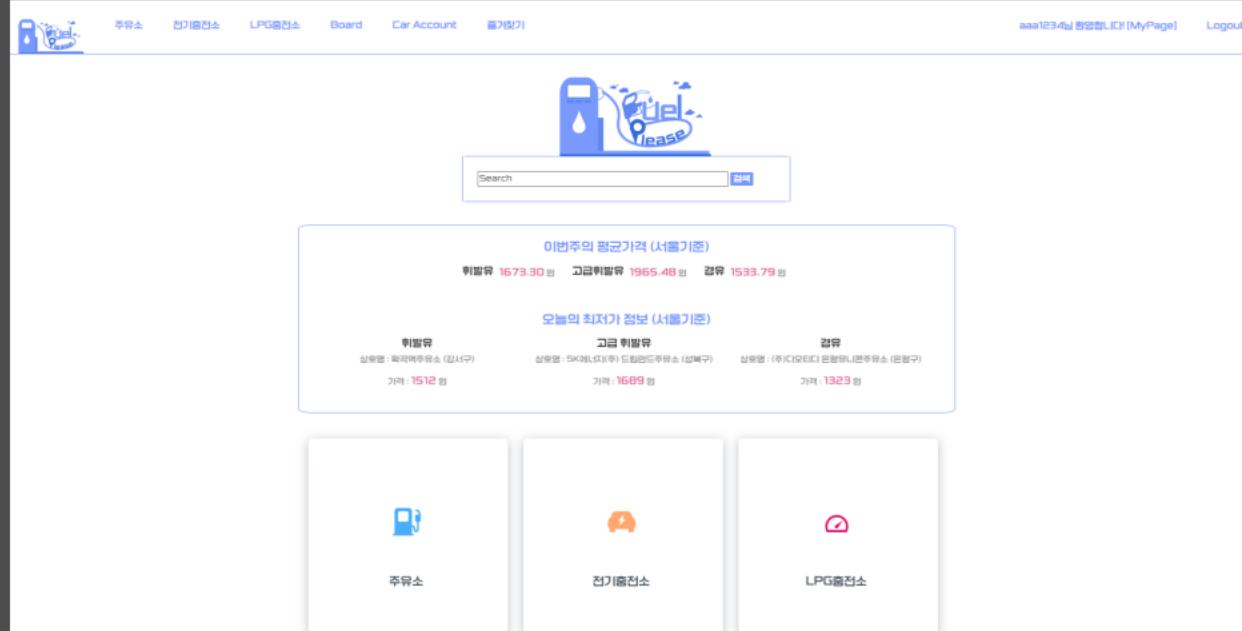
*Car
Account
Book*

Car Account Book



Join Page

Home Page



Fuel Please 의 기능을 한번에 확인가능한 페이지

Login Page

Join Page

회원가입

아이디
아이디를 (영문포함 4~16자 이상) 아이디생성

비밀번호
비밀번호 (영 대/소문자, 숫자 조합 8~16자 이상)

비밀번호 확인
비밀번호를 확인해주세요.

이름
이름을 입력하세요.

휴대폰번호
010 번호를 입력해주세요.

이메일
이메일 @naver.com

인증번호 6자리를 입력하세요.

주소
우편번호 주소찾기

기본주소

설세주소

회원 서비스를 활용하기 위한 회원가입 페이지

MyPage

Login Page

Login

아이디	아이디
비밀번호	비밀번호
<input type="button" value="LOGIN"/>	<input type="button" value="JOIN OUR SERVICE"/>

회원 서비스를 사용하기 위한 로그인 페이지

*Car
Account
Book*

MyPage

MEMBER INFO

*ID	ws1340cc
*이름	황우신
*비밀번호	
*비밀번호확인	ws1340cc
*E-mail	@naver.com
이메일 인증	
인증번호 입력란	
인증번호 6자리를 입력하세요.	
인증 확인	
*휴대폰	010 38855154
*우편번호	14039 주소찾기
*주소	경기 안양시 동안구 만양천동로 101-101
*상세주소	
수정 회원탈퇴	

회원 정보를 수정 가능한 페이지

BookMark

Car Account Book

Fuel Please 차계부

번호	일자	주유타입	가격	비고	
2	2023/6/7	고급휘발유	90000	안양 비산동에서 페리리 주유	<button>삭제</button>
1	2023/6/6	고급휘발유	150000	강남에서 포르쉐 911 주유	<button>삭제</button>
내가 사용한 총 금액		240000			

[작성하기](#)

1

회원이 사용한 유류비, 세차비 등 가계부를
기록하며 확인 할수있는 페이지

BookMark

나의 즐겨찾는 장소



장소 정보

서울특별시 관악구 남부순환로163길 14지하 1층 친일로 모른쪽 면길

DC콤보

24시간 이용가능

즐겨찾기 삭제

서울특별시 강남구 고덕로 39 (임시점)

서울특별시 강서구 화곡로 1013번지 14호

서울특별시 관악구 남부순환로163길 14지하 1층 친일로 모른쪽 면길

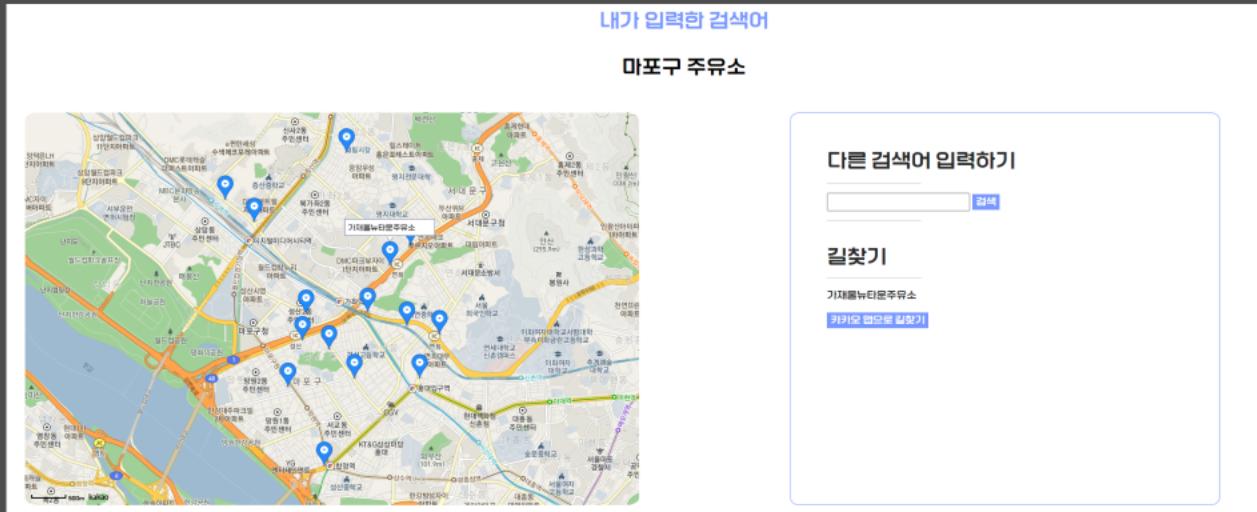
서울특별시 강북구 도봉로 124 (미아점)

서울특별시 강동구 전화대로 101997(전화로, 전화역광점주자점) 지하1층 A37번
기점

회원의 즐겨찾기 목록을 확인할수있는 페이지

*Find
GasStation*

Keyword Search



키워드를 통해 지도에서 위치를 찾는 페이지

*Find EV
Charging
Station*

Find GasStation

주유소 찾기

김남구 도곡로 208 (도곡동) 검색



검색한 주유소 정보

주유소 이름
SK에너지(주) 긴밀래주유소

주유소 주소
서울특별시 강남구 도곡로 208 (도곡동)

주유소 번호(TEL)
34620018

[경로찾기 추가]

주소 검색으로 주유소를 찾는 페이지

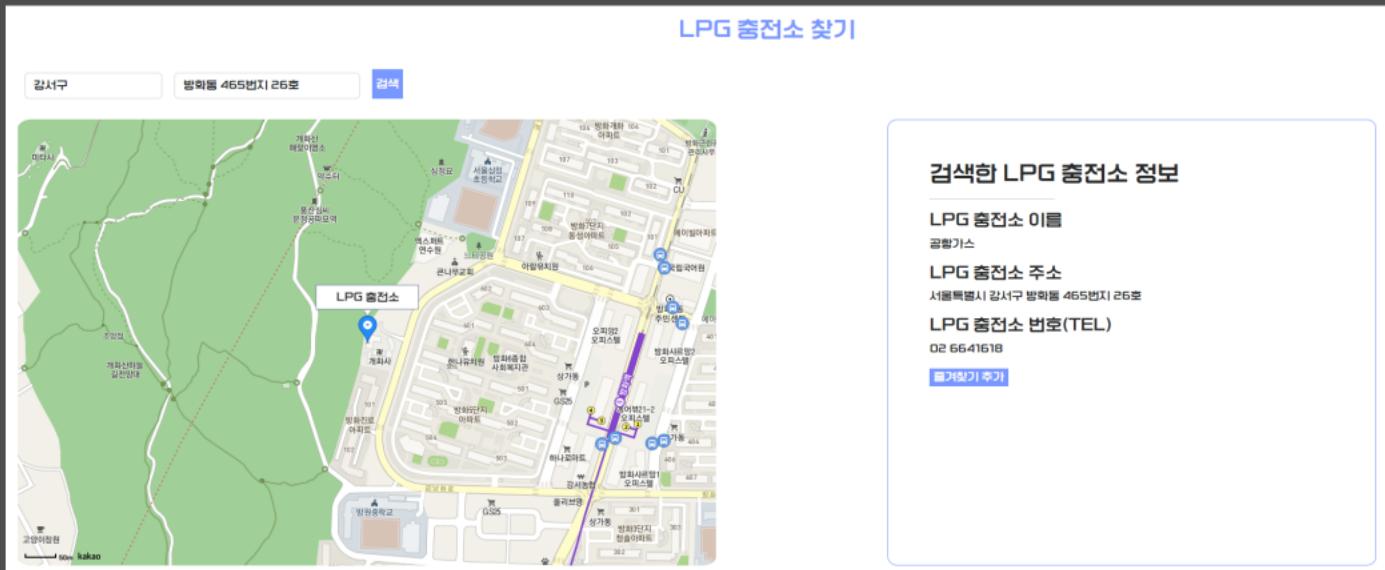
Find EV Charging Station

*Find LPG
Charging
Station*



주소 검색을 통해 전기차 충전소를 찾는 페이지

Find LPG Charging Station



주소 찾기를 통해 LPG 충전소를 찾는 페이지

기능 설계

MapView

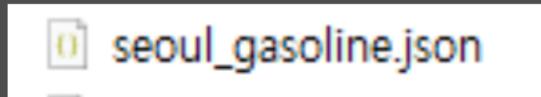
Opinet

Others

BookMark

**Car
Account**

MapView

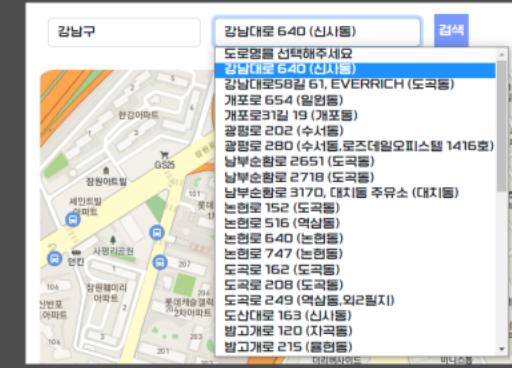
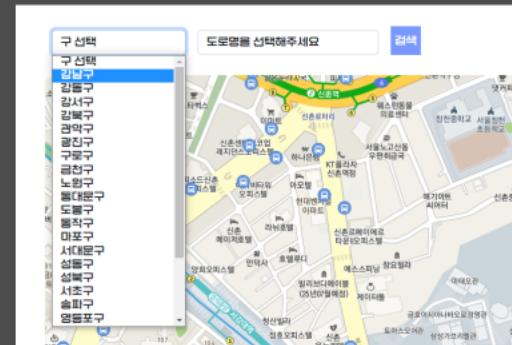


gno	sitetel	rdnwhladdr	bplcmn
4	000204177111	서울특별시 송파구 송파대로 397 (식동)	에이지디현대오일뱅크(주)직영 점실클럽주유소
5	02 4527723	서울특별시 강진구 진포대로 584 (능동)	에이지디현대오일뱅크(주)직영 능동주유소
6	0222172017	서울특별시 강진구 진포대로 258 (장암동)	삼명주유소
7	02 3755156	서울특별시 은평구 증산로 441 (신사동)	주식회사 오밀도컴 다회주유소
8	0233920473	서울특별시 강남구 풍납로 1772 (상계동)	(주)소모 수학산셀프주유소
9	8685758	서울특별시 구로구 구로로 94 (구로동)	SK구로주유소
10	02 8965145	서울특별시 강남구 서초대로 118 (서초동)	지에스칼텍스(주) 헬신주유소
11	34113271	서울특별시 강남구 남부순환로 3170, 대치동 주유소 (대치동)	지에스칼텍스(주) 헬신주유소
12	02 5444075	서울특별시 강남구 논현로 640 (논현동)	유한책임회사 풀하석유
13	02 22170009	서울특별시 성북구 아리랑로 4길 17 (동선동5가)	풀성석유
14	02 4981981	서울특별시 강진구 풍납로 266 (군자동)	(주)장수주유소
15	000222020388	서울특별시 송파구 오금로 18길 6 (송파동)	힐체석유
16	02 26332716	서울특별시 영등포구 선유로 260 (용마동47)	제2한강주유소
17	02 5151055	서울특별시 강남구 헬신로 11 (서초동)	거제서우

```
try {
    ObjectMapper objectMapper = new ObjectMapper();
    JsonNode rootnode = objectMapper.readTree(new File("C:\\Woosin\\mid_project_fuel\\FuelPlease\\seoul_gasoline.json"));

    JsonNode dataNode = rootNode.get("DATA");

    if (dataNode.isArray()) {
        for (JsonNode node : dataNode) {
            GasolineVO spot = objectMapper.convertValue(node, GasolineVO.class);
            storeList.add(spot);
        }
        filteredList = storeList.stream().filter(spot -> spot.getTrdStatenm().equals("영업/정상"))
            .collect(Collectors.toList());
    }
} catch (Exception e) {
    e.printStackTrace();
}
```



검색한 주유소 정보

주유소 이름

대교주유소

주유소 주소

서울특별시 강남구 김남대로 640 (신사동)

주유소 번호(TEL)

02 5125521

JSON 파일을 DB에 저장.

DB에 저장한 값을 화면에서 옵션선택을 통해 불러옴.

조합된 주소값을 카카오 지도검색에 전달.

지도에 마커 표시 후, 해당 주소를 가지고있는 주유소의 정보 표시.

BookMark

검색한 주유소 정보

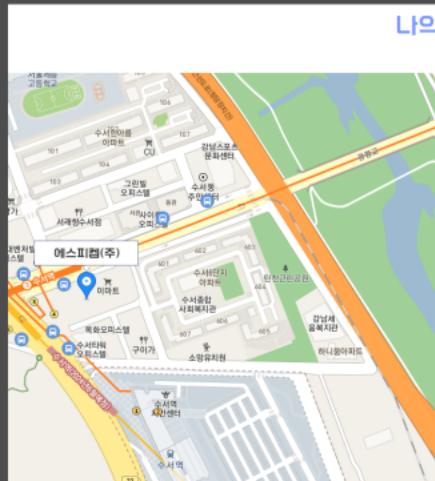
주유소 이름
에스피캠(주)

주유소 주소
서울특별시 강남구 광평로 280 (수서동,로즈데일오피스텔 1416호)

주유소 번호(TEL)
02 4596952

즐겨찾기 추가

나의 즐겨찾는 장소



장소 정보

서울특별시 강남구 광평로 280 (수서동,로즈데일오피스텔 1416호)
02 4596952
에스피캠(주)

즐겨찾기 목록

- 서울특별시 강남구 광평로 163길 14(서아름진길) 모쁜복 단원
- 서울특별시 강북구 도봉로 124 (미아동)
- 서울특별시 강동구 청호대로 710(천호동, 천호역글당주차장) 지하1층 A37번
기쁨
- 서울특별시 구로구 고척동
- 서울특별시 강남구 광평로 280 (수서동,로즈데일오피스텔 1416호)

```
INFO : jdbc.sqltiming - select bkaddr, bktel, bkname from bookmark where bkaddr = '서울특별시 강남구 광평로 280 (수서동,로즈데일오피스텔 1416호)' and bkuser_id = 'ws1340cc'
(executed in 1 msec)
INFO : jdbc.resultsettable -
|-----|-----|-----|
|bkaddr|bktel |bkname|
|-----|-----|-----|
서울특별시 강남구 광평로 280 (수서동,로즈데일오피스텔 1416호) |02 4596952 |에스피캠(주) |
```

bkno	bkuser_id	bkaddr	bktel	bkname
4	aaa1234	서울특별시 강동구 고덕로 168 (명일동)	0234281739	명일주유소
5	aaa1234	서울특별시 강동구 청호대로 지하997(천호동, ...	DC자데모+AC3상+DC콤보	24시간 이용가능
6	aaa1234	서울특별시 강동구 둔촌동 67번지	02 4886805	가스만서비스
7	aaa1234	서울특별시 강남구 강남대로 640 (신사동)	02 5125521	대교주유소
8	aaa1234	서울특별시 강남구 강구정로 321(서울특별시 ...	DC콤보	24시간 이용가능
24	ws1340cc	서울특별시 강동구 고덕로 39 (암사동)	02 4413227	(주)삼표에너지 고덕주유소
29	ws1340cc	서울특별시 강서구 화곡동 1013번지 14호	0226051527	돌서가스
30	ws1340cc	서울특별시 관악구 남부순환로163길 14(자하 1...	DC콤보	24시간 이용가능
31	ws1340cc	서울특별시 강북구 도봉로 124 (미아동)	02 9809551	주식회사 지에스이엔알 미아주유소
32	ws1340cc	서울특별시 강동구 청호대로 지하997(천호동, ...	DC자데모+AC3상+DC콤보	24시간 이용가능
33	ws1340cc	서울특별시 구로구 고척동	0220600923	할동에너지
35	ws1340cc	서울특별시 강남구 광평로 280 (수서동,로즈데...	02 4596952	에스피캠(주)

MapView에서 확인한 정보를 즐겨찾기 추가 버튼 클릭으로 즐겨찾기 DB 저장.
해당 기능은 회원 전용.
회원의 아이디별로 불러와서 타 이용자의 즐겨찾기는 볼 수 없음.
즐겨찾기 페이지에서 클릭시 해당 장소의 정보를 확인 가능.

Opinet

```
String url = "https://www.opinet.co.kr/api/avgLastWeek.do?prodcd=B027&code="+ApiKey+"&sido=01&out=xml";
```



```
for (temp = 0; temp < nList.getLength(); temp++) {  
  
    Log.info("휘발유 평균가격: " + getTagValue("PRICE", eElement));  
  
}  
return getTagValue("PRICE", eElement);
```



```
<RESULT>  
  <01L>  
    <UNI_ID>A0001005</UNI_ID>  
    <PRICE>1513</PRICE>  
    <POLL_DIV_CD>SOL</POLL_DIV_CD>  
    <OS_NM>현대주유소</OS_NM>  
    <VAN_ADDR>서울 양천구 신월동 167-1</VAN_ADDR>  
    <NEW_ADDR>서울 양천구 남부순환로 372 (신월동)</NEW_ADDR>  
    <GIS_X_COOR>296738.35070</GIS_X_COOR>  
    <GIS_Y_COOR>548590.85610</GIS_Y_COOR>  
  </01L>  
  <01L>  
    <UNI_ID>A0000978</UNI_ID>  
    <PRICE>1513</PRICE>  
    <POLL_DIV_CD>GSC</POLL_DIV_CD>  
    <OS_NM>풀라트(주)서호주유소</OS_NM>  
    <VAN_ADDR>서울 양천구 신월동 52-1</VAN_ADDR>  
    <NEW_ADDR>서울 양천구 남부순환로 317</NEW_ADDR>  
    <GIS_X_COOR>296517.01008</GIS_X_COOR>  
    <GIS_Y_COOR>549138.09694</GIS_Y_COOR>  
  </01L>  
  <01L>  
    <UNI_ID>A0000981</UNI_ID>  
    <PRICE>1519</PRICE>  
    <POLL_DIV_CD>SKE</POLL_DIV_CD>  
    <OS_NM>가로공원주유소</OS_NM>  
    <VAN_ADDR>서울 양천구 신월동 87-15</VAN_ADDR>  
    <NEW_ADDR>서울 양천구 가로공원로 165 (신월동)</NEW_ADDR>  
    <GIS_X_COOR>297172.00000</GIS_X_COOR>  
    <GIS_Y_COOR>548950.00000</GIS_Y_COOR>  
  </01L>  
</RESULT>
```

인증키를 통해 실시간으로 오픈 API 링크를 통해
XML 데이터를 받기.
태그에서 필요한 값을 추출함.
평균 유가 값과 최저가 주유소의 정보를 표시.
보안을 위해 Properties에 키 저장.

Car Account

나만의 차계부

번호	일자	주유타입	가격	비고	삭제
5	2023/6/2	LPG	40000	개봉동에서 다마스 충전	<button>삭제</button>
4	2023/6/9	전기	64000	고척동에서 타이칸 충전	<button>삭제</button>
3	2023/6/8	휘발유	90000	강남구에서 머스탱GT 주유	<button>삭제</button>
2	2023/4/9	고급휘발유	200000	비산 1동 주유소에서 페라리에 주유	<button>삭제</button>
내가 사용한 총 금액			394000		

충전소 LPG충전소 Board Car

삭제 하시겠습니까?

확인 취소

ws1340cc

나만의 차계부

번호	일자	주유타입	가격	비고	삭제
5	2023/6/2	LPG	40000	개봉동에서 다마스 충전	<button>삭제</button>
4	2023/6/9	전기	64000	고척동에서 타이칸 충전	<button>삭제</button>
3	2023/6/8	휘발유	90000	강남구에서 머스탱GT 주유	<button>삭제</button>
2	2023/4/9	고급휘발유	200000	비산 1동 주유소에서 페라리에 주유	<button>삭제</button>
내가 사용한 총 금액			394000		

작성하기

Fuel Please 차계부등록

연도
2023

월
4

일
9

주유타입 고급휘발유

가격
200000

비고
비산 1동 주유소에서 페라리에 주유

등록하기 목록

**주유타입과 금액을 작성하여 기록할 수 있는 가계부형식의 기록 기능.
하단에 전체 합계표시.
회원의 기록만 표시하며, 이용자 본인 외의 정보는 확인불가.**

Others

Login

아이디
ws1340cc

비밀번호

LOGIN JOIN OUR SERVICE

자유게시판

작성자
ws1340cc

제목
김주!!신모류주유소

내용
구로구 신모류 주유소 서비스 좋습니다!!

등록 목록

수정하기

번호
72

작성자
ws1340cc

제목
수점) 마포구주유소

내용
이자지! 짤oret 글을 적었네요
마포구에 있는 주유소입니다!

등록 목록

자유게시판

번호	제목	작성자	등록일	수정일
74	김주!!신모류주유소 [0]	ws1340cc	2023년 06월 08일 10시 31분	
73	이제 상용화 해도 되겠죠? [2]	ws1340cc	2023년 06월 07일 09시 43분	
72	수점) 마포구주유소 [0]	ws1340cc	2023년 06월 06일 18시 33분	2023년 06월 08일 10시 33분

**회원가입과 로그인을 통해 회원만 작성할 수 있는
정보게시판과 차계부, 즐겨찾기 서비스를 사용.
단, 장소 검색이나 키워드 통합 검색은 비회원도 사용이 가능.**

검색한 주유소 정보

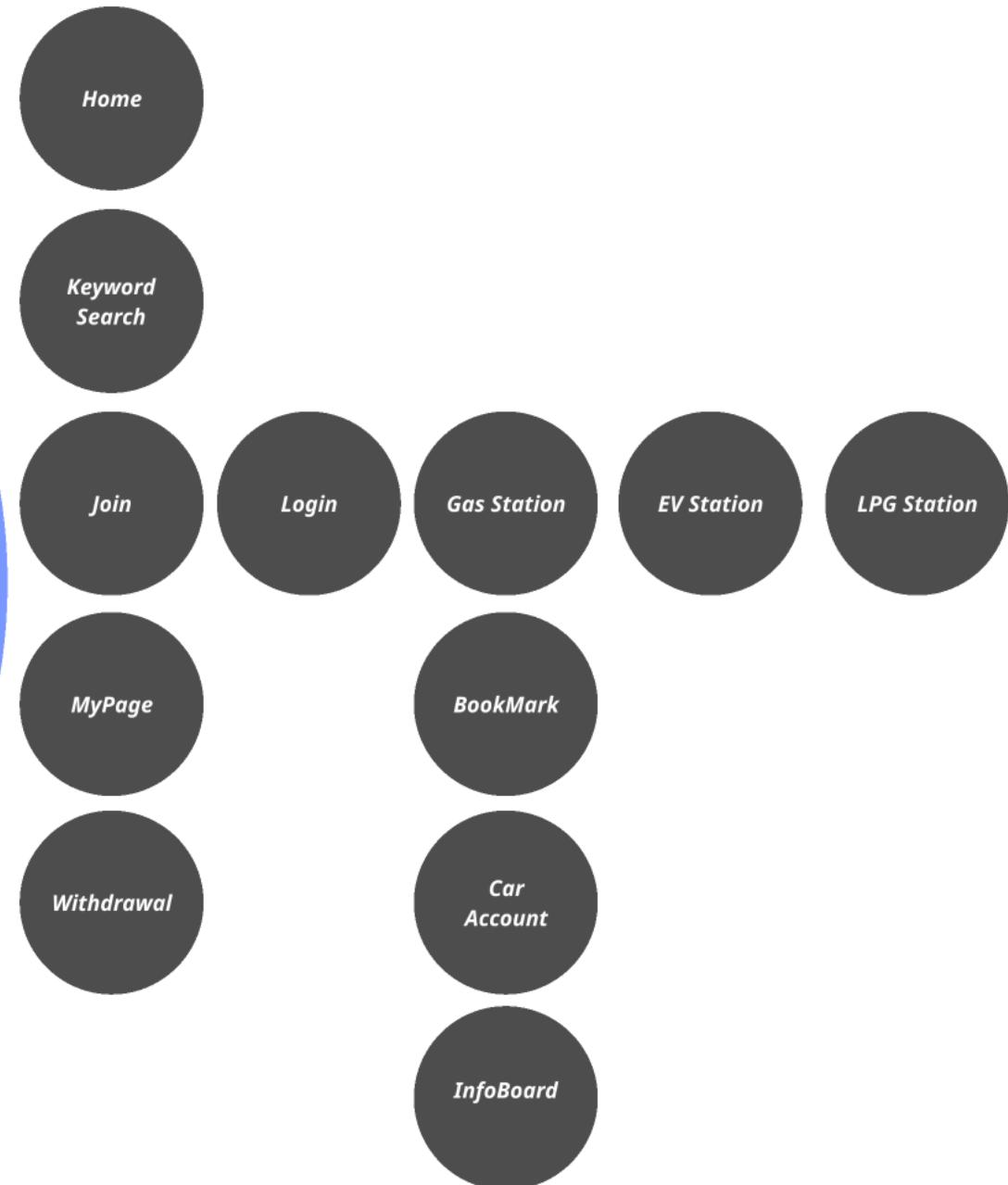
주유소 이름
서울석유(주)양재주유소

주유소 주소
서울특별시 서초구 바우뫼로 178 (양재동)

주유소 번호(TEL)
025771621

Detail of function

기능 설명

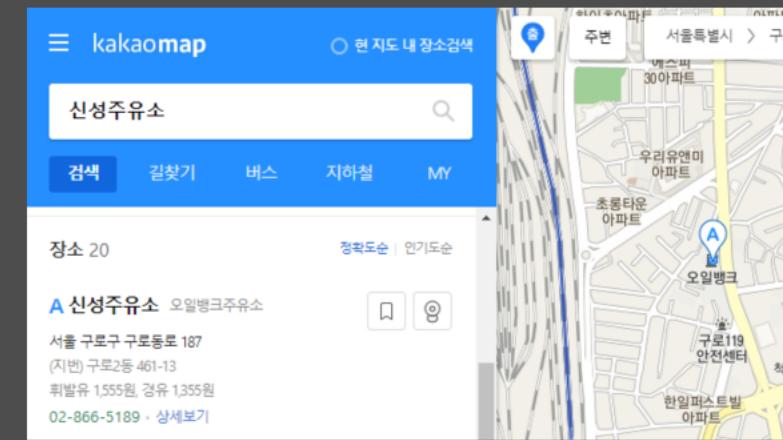
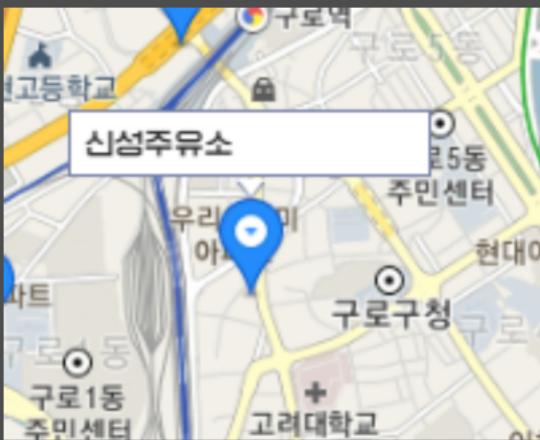
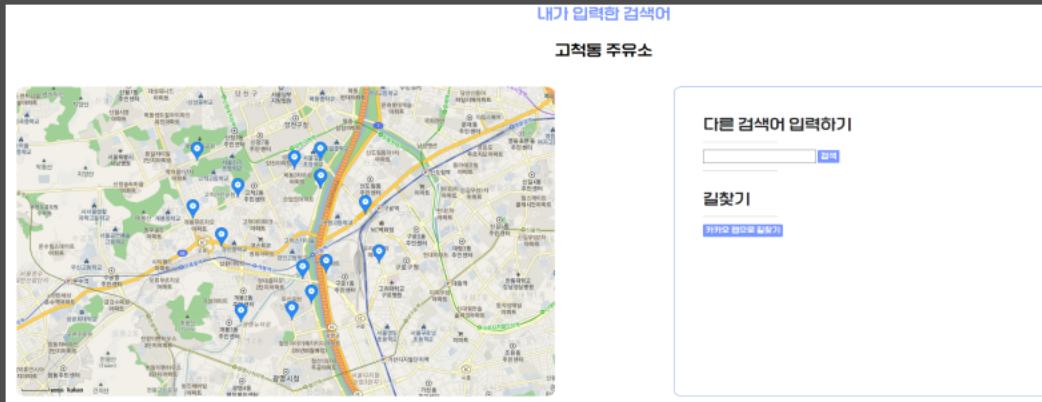
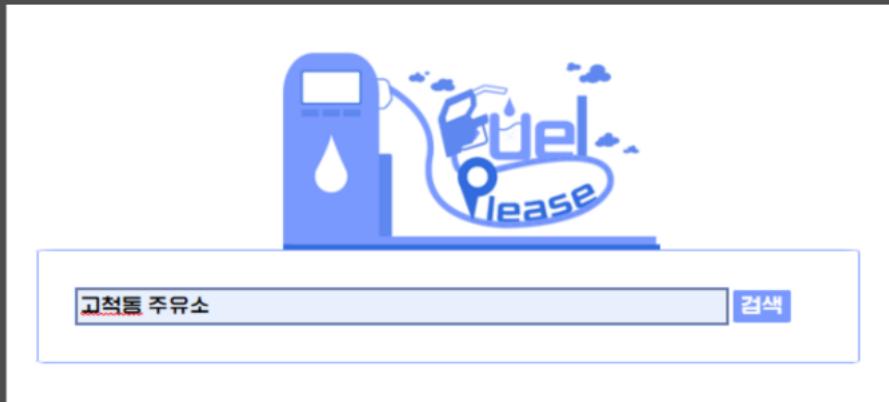


Home Page

The screenshot shows the homepage of a fuel price information website. At the top, there is a navigation bar with icons for fuel pumps and Korean text: "주유소" (Gas Station), "전기충전소" (EV Charger), "LPG충전소" (LPG Charger), and "Board". On the right side of the navigation bar are "Login" and "회원가입" (Member Registration) buttons. Below the navigation bar is a large logo featuring a blue fuel pump icon with the Korean word "연유" (Fuel) and the English word "Please". A search bar with the placeholder "Search" and a "검색" (Search) button is positioned below the logo. The main content area is enclosed in a light blue border. It contains two sections: "이번주의 평균가격 (서울기준)" (Average Price of the Week (Seoul Standard)) and "오늘의 최저가 정보 (서울기준)" (Today's Lowest Price Information (Seoul Standard)). The first section displays average prices: 휘발유 1673.30 원, 고급휘발유 1965.48 원, and 경유 1533.79 원. The second section displays today's lowest prices: 휘발유 1512 원, 고급 휘발유 1689 원, and 경유 1323 원. Below this information are three large white boxes with rounded corners, each containing a fuel pump icon and text: "주유소" (Gas Station), "전기충전소" (EV Charger), and "LPG충전소" (LPG Charger).

홈페이지에서 평균 유가 정보와 지역 최저가 주유소 정보를 확인할 수 있다.
각종 페이지로 이동할 수 있는 메뉴를 제공한다.

Keyword Search



홈페이지의 검색창을 통해 키워드를 작성하면 키워드에 해당되는 모든 장소가 지도에 표시된다.

마커를 클릭하면 상호명을 표시하고 카카오맵으로 길찾기 버튼을 누르면 카카오맵 화면이 띄워지며 상세정보를 볼 수 있다.

Join Page

회원가입

아이디
아이디를 (영문포함 4~12자 이상)

비밀번호
비밀번호 (영 대/소문자, 숫자 조합 8~16자 이상)

비밀번호 확인
비밀번호를 확인해주세요.

이름
이름을 입력하세요.

휴대폰번호
010

이메일
이메일 @naver.com

인증번호 6자리를 입력하세요.

주소
우편번호
기본주소
상세주소

localhost 내용:
인증번호가 전송되었습니다. 확인후 입력란에 정확히 입력하세요.

아이디
abc123123 확인

사용 가능한 아이디입니다.

비밀번호

사용 가능합니다.

비밀번호 확인

비밀번호가 일치합니다.

이름
홍길동

휴대폰번호
011 12345678

이메일
ws1340cc @naver.com

인증번호 6자리를 입력하세요.

주소
우편번호
기본주소
상세주소

비밀번호가 일치합니다!

이름
홍길동

휴대폰번호
011 12345678

이메일
ws1340cc @naver.com

인증번호가 일치합니다!

주소
우편번호
기본주소
상세주소

Datum Postcode Service - Chrome
① about:blank
거구장
도로명 전체 지역명 전체
04108 도로명 서울 마포구 백범로 23 (거구장)
자·번 서울 마포구 신수동 63-14
내·외 지번주소 8번 [보기]
53054 도로명 경남 통영시 통영2길 22-3 (거구장갈비)
자·번 경남 통영시 통영2길 147-33
50049 도로명 경남 합천군 합천읍 합천남서로 1074 (거구장)
자·번 경남 합천군 합천읍 이온리 302-12
1 / 1

주소
04108

서울 마포구 백범로 23

상세주소

회원 서비스를 이용하기 위해 회원가입을 진행한다.
아이디 중복확인과 이메일 인증이 필수이며 작성 양식에 따라 정보를 생성해야한다.
카카오 우편번호 찾기 API를 통해 주소를 입력할 수 있다.

Login Page

The screenshot shows a web-based login interface. At the top left is a logo featuring a blue fuel pump and a white car. To its right are menu items: 주유소 (Gas Station), 전기차 충전소 (EV Charging Station), LPG 충전소 (LPG Charging Station), and Board. On the far right are Login and 회원가입 (Join) buttons. The main area is titled "Login". It contains two input fields: "아이디" (ID) with placeholder "아이디" and "비밀번호" (Password) with placeholder "비밀번호". Below these are two buttons: "LOGIN" and "JOIN OUR SERVICE".

This is a zoomed-in view of the "Login" form. The "아이디" field is filled with "abc123123". The "비밀번호" field contains several dots as a placeholder. The "LOGIN" and "JOIN OUR SERVICE" buttons are visible at the bottom.

회원가입을 진행한 사용자는 로그인을 통해 여러 서비스를 활용할 수 있다.

사용 가능 : 주유소, 전기차 충전소, LPG 충전소 즐겨찾기 서비스.
차계부, 정보공유게시판 게시글 작성 및 댓글 작성.

Find GasStation Page

The screenshot shows a map of a city area with various landmarks and roads. A search interface is overlaid, featuring a search bar at the top left and a large blue button labeled '검색' (Search) at the top right. Below the search bar, there are three dropdown menus: '구 선택' (District selection), '도로명을 선택해주세요' (Please select a road name), and another '검색' button. To the right of the map, a sidebar titled '주유소 찾기' (Find GasStation) contains a section titled '검색한 주유소 정보' (Information about the searched gas station). This section includes fields for '주유소 이름' (Gas station name), '주유소 주소' (Gas station address), '주유소 번호(TEL)' (Gas station phone number), and a '즐겨찾기 추가' (Add to favorites) button. A blue box highlights the '주유소 주소' field.

주유소 찾기 페이지에서 사용자는 주소를 선택하여 해당 주유소의 위치와 주유소 정보를 볼 수 있다.

또한 회원은 즐겨찾기 추가 버튼을 눌러 본인의 즐겨찾기 페이지 공간에 저장할 수 있다.

This screenshot shows a detailed map of a specific area, likely a university campus or industrial zone, with buildings labeled such as '도곡파크', '도곡스타빌', 'KAIST 도곡캠퍼스', '오일뱅크', '이스타빌 오피스텔', '메이저아파트', and '신선설농장'. A blue callout box labeled '주유소' (Gas Station) points to a location on the map. To the right of the map is a sidebar titled '검색한 주유소 정보' (Information about the searched gas station), which displays the same fields as the previous screenshot: '주유소 이름' (Gas station name), '주유소 주소' (Gas station address), '주유소 번호(TEL)' (Gas station phone number), and a '즐겨찾기 추가' (Add to favorites) button. A blue box highlights the '주유소 주소' field.

Find EV Charge Staion Page

전기차 충전소 찾기

구 선택 도로명을 선택해주세요 검색

검색한 전기차 충전소 정보

전기차 충전소 주소
-
충전소 이용 시간
-
전기차 충전 타입
-
전기차 충전 용량
-
즐겨찾기 추가

광진구

도로명을 선택해주세요
도로명을 선택해주세요
자양번영로 45노상금명주자장 유료 42번
구의강변로 11점문 앞 유료 48번

도로명 216

검색한 전기차 충전소 정보

전기차 충전소 주소
서울특별시 광진구 능동로 216(서울특별시 광진구 능나루로 495 전기차충전소)
충전소 이용 시간
24시간 이용 가능
전기차 충전 타입
DC콤보
전기차 충전 용량
200kW

전기차 충전소

검색한 전기차 충전소 정보

전기차 충전소 주소
서울특별시 광진구 능동로 216(서울특별시 광진구 능나루로 495 전기차충전소)
충전소 이용 시간
24시간 이용 가능
전기차 충전 타입
DC콤보
전기차 충전 용량
200kW

전기차 충전소 찾기 페이지에서 사용자는 주소를 선택하여 해당 충전소의 위치와 충전소 정보를 볼 수 있다.

또한 회원은 즐겨찾기 추가 버튼을 눌러 본인의 즐겨찾기 페이지 공간에 저장할 수 있다.

Find LPG Staion Page

LPG 충전소 찾기

구 선택 도로명을 선택해주세요 검색

검색한 LPG 충전소 정보

- LPG 충전소 이름
- LPG 충전소 주소
- LPG 충전소 번호(TEL)
- 즐겨찾기 추가

강동구

도로명을 선택해주세요

- 도로명을 선택해주세요
- 둔촌동 67번지 1호
- 명일동 312번지 1호
- 상일동 20번지 1호**
- 상내동 453번지 2호
- 임사동 511번지 3호
- 천호동 325번지 8호
- 천호동 450번지 3호

검색

도로명을 선택해주세요

둔촌동 67번지 1호

명일동 312번지 1호

상일동 20번지 1호

상내동 453번지 2호

임사동 511번지 3호

천호동 325번지 8호

천호동 450번지 3호

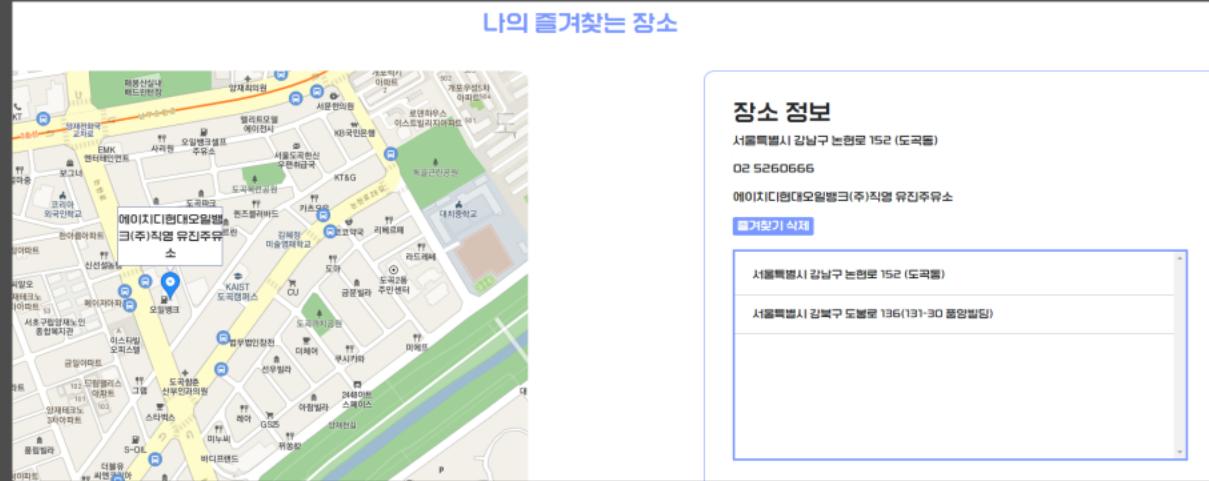
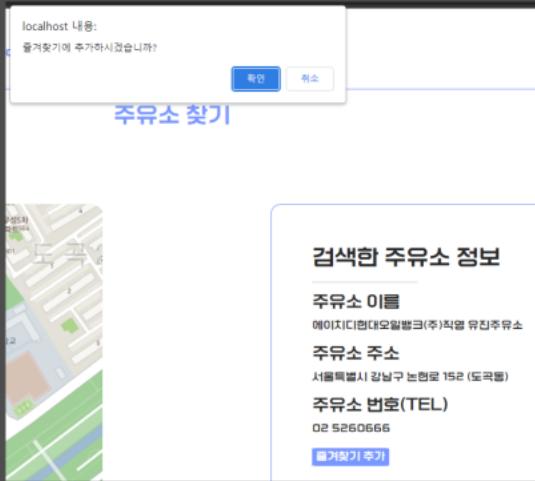
검색한 LPG 충전소 정보

- LPG 충전소 이름
- LPG 충전소 주소
- 서울특별시 강동구 상일동 20번지 1호
- LPG 충전소 번호(TEL)
- 02 4291800
- 즐겨찾기 추가

LPG 충전소 찾기 페이지에서 사용자는 주소를 선택하여 해당 충전소의 위치와 충전소 정보를 볼 수 있다.

또한 회원은 즐겨찾기 추가 버튼을 눌러 본인의 즐겨찾기 페이지 공간에 저장할 수 있다.

BookMark Page



앞서 주유소 찾기 페이지에서 즐겨찾기를 추가한 회원은 즐겨찾기 페이지에서 자신이 추가한 장소를 다시 확인할 수 있다.
또한 삭제버튼을 통해 즐겨찾기 목록에서 해당 장소를 삭제할 수 있다.

Car Account Page

차계부 등록

연도 2023	월 6	일 1
주유타입 고급휘발유		
가격 200000		
비고 강남구 논현동에서 주유함.		
<button>등록하기</button>	<button>목록</button>	

나만의 차계부

번호	일자	주유타입	가격	비고
11	2023/5/31	전기	50000 원	마포구 토정로에서 충전함.
10	2023/6/1	고급휘발유	2000000 원	강남구 논현동에서 주유함.
내가 주유한 총 금액				250,000 원
나의 평균 주유비용				125,000 원
<button>작성하기</button> 1				

차계부

localhost 내용: 삭제 하시겠습니까?		<button>확인</button> <button>취소</button>
abc123123		
가격	비고	
50000 원	마포구 토정로에서 충전함.	<button>삭제</button>
200000 원	강남구 논현동에서 주유함.	<button>삭제</button>
250,000 원		
125,000 원		
<button>작성하기</button> 1		

차계부 페이지에서는 회원의 주유 기록을 작성할 수 있고 지출 합계와 평균 주유비용을 확인할 수 있다. 또한, 삭제를 통해 기록을 정정할 수 있다.

InfoBoard Page

자유게시판

작성자 abc123123
제목 김남구 논현동 주유소
내용 김남구 논현동 주유소 생각보다 잘끔하네요!

등록 목록

자유게시판

번호	제목	작성자	등록일	수정일
76	김남구 논현동 주유소 [0]	abc123123	2023년 06월 08일 18시 04분	
75	주기요청합니다! [0]	abc123123	2023년 06월 08일 17시 36분	
74	김주민(화재주유소) [0]	ws1340cc	2023년 06월 08일 10시 31분	
73	이제 상품화 해도 되겠죠? [2]	ws1340cc	2023년 06월 07일 09시 43분	
72	수입 대포구주유소 [0]	ws1340cc	2023년 06월 06일 18시 33분	2023년 06월 08일 10시 39분
71	테스트 [0]	ws1340cc	2023년 06월 06일 18시 33분	
70	테스트 [0]	ws1340cc	2023년 06월 06일 18시 33분	
69	테스트 [0]	ws1340cc	2023년 06월 06일 18시 33분	
68	테스트 [0]	ws1340cc	2023년 06월 06일 18시 33분	
67	테스트 [0]	ws1340cc	2023년 06월 06일 18시 33분	

1 2 3 4 5 다음
등록하기

localhost 내용:
변경 페이지로 이동합니다.

확인 취소

DATE
2023년 06월 08일 18시 04분

번호
76

작성자
abc123123

제목
김남구 논현동 주유소

내용
김남구 녺현동 주유소 생각보다 잘끔하네요!

등록 수정 삭제

abc123123

하하하!

등록하기

abc123123

비방 및 욕설 등록 시 처벌 받을 수 있습니다.

등록하기

abc123123 방금 전 삭제
하하하!

ws1340cc 방금 전 삭제
그렁군요!!

등록 수정 삭제

내용
김남구 논현동 주유소 생각보다 깨끗해용!

등록 수정 삭제

게시판에서 회원은 게시글을 작성하며 정보를 공유할 수 있다.
또한, 댓글을 작성할 수 있다. 모든 게시글과 댓글은 회원 본인이 작성한 것만 수정과 삭제가 가능하다.
비회원은 게시글을 볼 수 있으나 작성은 할 수 없다.

MyPage

MEMBER INFO

*ID abc123123	*이름 홍길동	*비밀번호
*비밀번호확인 ws1340cc	*E-mail @naver.com	이메일 인증 인증번호 입력란 인증 확인
*휴대폰 011 3401515		인증번호 6자리를 입력하세요.
*우편번호 04108		이메일 인증 인증번호 입력란 인증 확인
*주소 서울 마포구 백범로 23		인증번호 6자리를 입력하세요.
*상세주소 3층		인증 확인
<input type="button" value="수정"/> <input type="button" value="회원탈퇴"/>		

MEMBER INFO

localhost 내용:
정보 수정을 진행합니다.

*ID abc123123	*이름 홍길동	*비밀번호 ***** 사용 가능합니다.
*비밀번호확인 ws1340cc	*E-mail @naver.com	비밀번호가 일치합니다. 이메일 인증 인증번호 입력란 인증 확인
*휴대폰 011 3401515		인증번호 6자리를 입력하세요.
*우편번호 06062		이메일 인증 인증번호 입력란 인증 확인
*주소 서울 강남구 도산대로 402-2		인증번호 6자리를 입력하세요.
*상세주소 101-101		인증 확인
<input type="button" value="수정"/> <input type="button" value="회원탈퇴"/>		

마이 페이지에서는 회원의 정보를 확인할 수 있다.
회원의 정보 수정을 진행 할 수 있다.

Withdrawal from Membership

회원탈퇴

비밀번호
 비밀번호

탈퇴

마이 페이지에서 회원 탈퇴 버튼을 클릭하면
회원 탈퇴 페이지로 이동하고
비밀번호를 정확히 일치 하지 않으면 탈퇴하지 못한다.
회원의 비밀번호를 올바르게 입력할 경우
확인 알림이 나타나고 확인을 누르면 탈퇴가 된다.

회원탈퇴

비밀번호
 비밀번호
비밀번호 불일치

탈퇴

localhost 내용:
정말 탈퇴하시겠습니까?

확인 **취소**

회원탈퇴

비밀번호

탈퇴



*Log in
&
Log out*

*Join
&
Withdrawal*

MyPage

InfoBoard

Bookmark

CarAccount

Mapview

FuelAvg

Technology

Login & Log out

```
@GetMapping("/userLogin") // 로그인 페이지 이동  
public void loginPage() {
```

```
}
```

```
@PostMapping("/userLogin")// 로그인 요청
```

```
public void login(String userId, String userPw, Model model) {  
    Log.info("사용자 로그인 요청!");
```

```
    model.addAttribute("user", sv.userLogin(userId, userPw));// user에 정보 담아 보냄
```

```
@Override  
public String userLogin(String id, String pw) {  
    String dbPw = mp.userLogin(id);  
    if(dbPw != null) {  
        if(encoder.matches(pw, dbPw)) {  
            return id;  
        }  
    }  
    return null;  
}
```

```
// 접속란이 끝나면 마지막에 [마니아] Submit  
// 혹은 url은 //user/userLogin → post로 간다.
```

```
document.getElementById('loginBtn').onclick = function () {  
    const id = document.getElementById('userId').value;  
    const pw = document.getElementById('userPw').value;  
  
    if (id === '') {  
        alert('아이디를 입력해주세요');  
        return;  
    }  
    if (pw === '') {  
        alert('비밀번호를 입력해주세요');  
        return;  
    }  
  
    document.loginForm.submit();  
}
```

```
/[엔터키]  
document.addEventListener("keyup", function (event) {  
    const id = document.getElementById('userId').value;  
    const pw = document.getElementById('userPw').value;
```

```
    if (event.keyCode === 13) {  
        if (id === '') {  
            alert('아이디를 입력해주세요');  
            return;  
        }  
        if (pw === '') {  
            alert('비밀번호를 입력해주세요');  
            return;  
        }  
  
        document.loginForm.submit();  
    }  
});  
document.getElementById('joinBtn').onclick = () => {  
    location.href = '${pageContext.request.contextPath}/user/userJoin';  
}
```

```
@Override  
public void postHandle(HttpServletRequest request, HttpServletResponse response, Object handler,  
    ModelAndView modelAndView) throws Exception {  
  
    Log.info("로그인 인터셉터가 활작!");  
    Log.info("요청 방식" + request.getMethod());  
  
    if(request.getMethod().equals("POST")) {  
        ModelMap map = modelAndView.getModelMap(); //모델 객체 꺼내기  
        String id = (String) map.get("user"); //모델 내에 user라는 이름의 데이터 꺼내기  
        log.info("인터셉터 내부에서 user 확인: " + vo.toString());  
        if(id != null) { //로그인 성공  
            log.info("로그인 성공 로직이 활작합니다.");  
            //로그인 성공한 회원에게 세션 데이터를 생성해서 로그인 유지를 하게 해 줌.  
            HttpSession session = request.getSession();  
            session.setMaxInactiveInterval(60 * 60);  
            session.setAttribute("login", id);  
            Log.info((String) session.getAttribute("login"));  
            Log.info(id);  
            response.sendRedirect(request.getContextPath() + "/");  
        } else { //null == null -> 로그인 실패  
            modelAndView.addObject("msg", "loginFail");  
        }  
    }  
}
```

다음 코드는 로그인 로그아웃에 대한 코드이다.
유저는 회원가입을 할 때 회원의 비밀번호를 아이디로 조회를 한다. 조회를 한 다음 그것이 회원가입때 작성한 비밀번호와 같으면 로그인 함수를 실행한다.
로그인을 성공적으로 수행하면 세션에 로그인정보를 저장해두며 이는 회원서비스를 사용할 수 있게한다.

```
//로그인  
String userLogin(String id);  
  
//회원가입  
void userJoin(UserVO vo);  
  
//아이디 중복확인  
int idCheck(String id);  
  
//회원정보 얻어오기  
UserVO getInfo(@Param("userId") String id, @Param("paging") PageVO vo);  
  
//회원정보 수정  
void updateUser(UserVO vo);  
  
//회원 탈퇴  
void deleteUser(@Param("userId") String id, @Param("userPw") String userPw);
```

```
<select id="userLogin" resultType="string">  
    select user_pw from users  
    where user_id = #{userId}  
</select>
```

```
// 로그아웃  
// DB 작업 없으므로 따로 service나 mapper 작업 필요 없음  
@GetMapping("userLogout")  
public ModelAndView logout(HttpServletRequest session) {  
    Object object = session.getAttribute("login");  
  
    // 세션에 로그인 정보가 있다면  
    if(object != null) {  
        // "login" 세션 삭제  
        session.removeAttribute("login");  
        // 세션 정보 초기화  
        session.invalidate();  
    }  
  
    return new ModelAndView("redirect:/");  
}
```

로그아웃은 세션을 삭제하고 초기화 시키면 된다!

Join

Withdrawal

```

<insert id="userJoin">
    insert into users
    (user_id, user_pw, user_name, user_phone1, user_phone2,
    user_email1, user_email2, user_addr, addr_detail, zip_cd)
    values
    (#{userId},#{userPw},#{userName},#{userPhone1},#{userPhone2},
    #{userEmail1},#{userEmail2},#{userAddr},#{addrDetail},#{zipCd})
</insert>

<select id="idCheck" resultType = "int">
    select count(*) from users where user_id = #{userId}
</select>

@GetMapping("/userJoin") // 회원 가입 페이지 이동
public void join() {
}

@PostMapping("/userJoin") // 가입 후 홍으로 이동
public String join(UserVO vo, RedirectAttributes ra) {
    sv.userJoin(vo);
    ra.addFlashAttribute("msg", "joinSuccess"); // msg에 메세지 담아 보내서 이것으로 판단할것임
    return "redirect:/";
}

@PostMapping("/idCheck") // 아이디중복체크
@ResponseBody
public int idCheck(@RequestBody String id) {
    log.info("사용자가 입력한 아이디: " + id);
    if (sv.idCheck(id) == 1) { // 중복이면 1반환
        return 1;
    } else
        return 0;
}

@GetMapping("/mailCheck") // 이메일 인증
@ResponseBody
public String mailCheck(String email) {
    log.info("이메일 인증 요청 들어온: " + email);
    return mailsv.joinEmail(email);
}

//난수 발생
public int makeRandomNumber() {
    //난수의 범위: 111111 ~ 999999
    Random r = new Random();

    int checkNum = r.nextInt(888888)+111111;
    Log.info("인증번호: " + checkNum);
    return checkNum;
}

```

```

//회원가입
void userJoin(UserVO vo);

//아이디 중복확인
int idCheck(String id);

```

```

//비동기 요청 보내기
fetch('${pageContext.request.contextPath}/user/idCheck', {
    method: 'post',
    headers: {
        'Content-type': 'text/plain'
    },
    body: userId
})
.then(res => res.text()) //요청 완료 후 응답 정보에서 텍스트만 빼기
.then(data => { //텍스트만 뺀 Promise 객체로부터 data 전달받음.
    if (data === '1') {
        msg.textContent = '중복된 아이디입니다.';
    } else {
        document.getElementById('userId').setAttribute('readonly', true);
        //document.getElementById('idCheckBtn').setAttribute('disabled', true);
        msg.textContent = '사용 가능한 아이디입니다.';
    }
}); //아이디 중복확인 끝

```

```

//회원 가입 시 사용할 이메일 양식
public String joinEmail(String email) {
    authNum = makeRandomNumber();

    String setFrom = "ws1340cc@naver.com"; //email-config에 설정한 발신을 이메일 주소
    String toMail = email; // 수신받을 이메일(가입하고자 하는 사람의 이메일)
    String title = "회원 가입 인증 이메일.";
    String content = "홈페이지를 방문해 주셔서 감사합니다." +
        "<br><br>" +
        "인증 번호는 <strong>" + authNum + "</strong> 입니다." +
        "<br>" +
        "해당 인증 번호를 인증란에 기입해 주세요."; //이메일에 삽입할 내용 +
    mailSend(setFrom, toMail, title, content);
    return Integer.toString(authNum); //정수를 문자열로 리턴
}

```

```

@Override
public void userJoin(UserVO vo) {
    log.info("암호화 하기 전 비번: " + vo.getUserPw());
    //비밀번호를 암호화 해서 XOR 암호화 해서 다시 저장하기.
    String securePw = encoder.encode(vo.getUserPw());
    log.info("암호화 후 비번: " + securePw);
    vo.setUserPw(securePw);
    mp.userJoin(vo);
}

```

```

//비밀번호 형식 검사 스크립트/
var pw = document.getElementById("userPw");
pw.onkeyup = function () {
    var regex = /^[A-Za-z0-9+]{8,16}$/;
    if (regex.test(document.getElementById("userPw").value)) {
        document.getElementById("userPw").style.borderColor = "green";
        document.getElementById("msgPw").innerHTML = "사용 가능합니다";
        pwFlag = true;
    } else {
        document.getElementById("userPw").style.borderColor = "red";
        document.getElementById("msgPw").innerHTML = "올바른 비밀번호를 입력하세요";
        pwFlag = false;
    }
};

//비밀번호 확인검사/
var pwConfirm = document.getElementById("pwConfirm");
pwConfirm.onkeyup = function () {
    var regex = /^[A-Za-z0-9+]{8,16}$/;
    if (document.getElementById("pwConfirm").value === document.getElementById("userPw").value) {
        document.getElementById("pwConfirm").style.borderColor = "green";
        document.getElementById("msgPw-c").innerHTML = "비밀번호가 일치합니다";
    } else {
        document.getElementById("pwConfirm").style.borderColor = "red";
        document.getElementById("msgPw-c").innerHTML = "비밀번호 확인란을 확인하세요";
    }
};

```

회원가입은 먼저 사용자가 입력한 아이디를 비동기방식으로 전달한 후, 아이디체크함수를 통해 사용자의 아이디가 사용 가능한지 아닌지 판단하여 리턴값을 전달하고, 전달받은 리턴값을 통하여 화면단에 알림을 띠운다. 비밀번호는 정규표현식에 따라 검사를 진행하고 알맞은 비밀번호는 암호화를 통해 DB에 저장이 된다. 또한 이메일을 작성하면 난수를 생성하여 양식과 함께 사용자가 작성한 이메일로 전송이되며, 이는 인증용으로 사용된다.

Withdrawal

```
const msg = document.getElementById('msgId');
document.getElementById('delBtn').onclick = function () {
    const pw = document.getElementById("userPw").value;
    if (pw === '') {
        alert("비밀번호를 입력해주세요")
        return;
    }

    if (confirm('정말 탈퇴하시겠습니까?')) {
        fetch(`${pageContext.request.contextPath}/user/userDelete`, {
            method: 'post',
            headers: {
                'Content-type': 'text/plain'
            },
            body: pw
        })
        .then(res => res.text()) // 요청 완료 후 응답 정보에서 텍스트만 빼기
        .then(data => { // 텍스트만 뺀 Promise 객체로부터 data 전달받음.
            if (data === '1') {
                alert("탈퇴가 완료되었습니다.");
                location.href = `${pageContext.request.contextPath}/`;
            } else {
                console.log(data);
                console.log(pw);

                document.getElementById("userPw").value = '';
                msg.textContent = '비밀번호 불일치';
            }
        });
    }
} else return;
```

```
@GetMapping("/userDelete")
public void userDelete() {
}

@PostMapping("/userDelete")
@ResponseBody
public int userDelete(HttpServletRequest session, @RequestBody String userPw) {
    String id = (String)session.getAttribute("login");
    log.info("id: " + id);
    log.info("pw: " + userPw);
    int result = sv.deleteUser(id, userPw);
    if(result == 1) { // 아이디,비번 일치하고 회원삭제 하면 1 리턴.
        Object object = session.getAttribute("login");
        // 세션에 로그인 정보가 있다면
        if(object != null) {
            // "login" 세션 삭제
            session.removeAttribute("login");
            // 세션 정보 초기화
            session.invalidate();
            return 1;
        }
        return -2;
    }
    else return 0;
}
```

```
//회원 탈퇴
void deleteUser(@Param("userId")String id, @Param("userPw")String userPw);

@Override
public int deleteUser(String id, String userPw) {
    log.info("사용자 세션 아이디: " + id);
    String dbPw = mp.userLogin(id);
    log.info("DB저장 비번:" + dbPw);
    log.info("결과: {}",encoder.matches(userPw, dbPw));
    if(dbPw != null) {
        if(encoder.matches(userPw, dbPw)) {
            mp.deleteUser(id,dbPw);
            return 1;
        }
        return 0;
    }
    return -2;
}
```

회원탈퇴는 사용자가 입력한 값이 기존의 암호화된 비밀번호가 일치하는지 **encoder.matches**를 통해 확인한 후 리턴값을 넘겨 비동기방식으로 확인후 탈퇴를 진행한다. 일치하지 않다면 '비밀번호 불일치' 가 메세지로 출력이 된다.

MyPage

```
<!-- 마이페이지 -->
<select id="getInfo" resultMap="userMap">
    select
        user_name, user_phone1, user_phone2, user_pw,
        user_email1, user_email2,
        user_addr, addr_detail, zip_cd
    from users
    where user_id = #{userId}
</select>

<!-- 회원 정보 수정 -->
<update id="updateUser">
    update users
    set user_pw = #{userPw},
        user_phone1 = #{userPhone1},
        user_phone2 = #{userPhone2},
        user_email1 = #{userEmail1},
        user_email2 = #{userEmail2},
        user_addr = #{userAddr},
        addr_detail = #{addrDetail},
        zip_cd = #{zipCd}
    where user_id = #{userId}
</update>
```

```
// 마이페이지 이름 요청
@GetMapping("/userMypage")
public void userMypage(HttpServletRequest session, Model model, PageVO vo) {
    String id = (String) session.getAttribute("login");
    vo.setLoginId(id);
    PageCreator pc = new PageCreator(vo, bsv.getTotal(vo));
    model.addAttribute("userInfo", sv.getInfo(id, vo));
    model.addAttribute("pc", pc);
}
```

```
// 회원정보 수정
@PostMapping("/updateUser")
public String updateUser (UserVO vo) {
    sv.updateUser(vo);
    return "redirect:/user/userMypage";
}
```

```
@Override
public UserVO getInfo(String id, PageVO vo) {
    return mp.getInfo(id, vo);
}

@Override
public void updateUser(UserVO vo) {
    String updatePw = encoder.encode(vo.getUserPw());
    vo.setUserPw(updatePw);
    mp.updateUser(vo);
}
```

```
// 이메일 변경했는데 인증 안한 경우
let email1 = document.getElementById('userEmail1').value;
let email2 = document.getElementById('userEmail2').value;
let userEmail1 = '${userInfo.userEmail1}';
let userEmail2 = '${userInfo.userEmail2}';

// console.log(userEmail1);
// console.log(email1);

if (email1 !== userEmail1 || email2 !== userEmail2) {
    if (!document.getElementById('mail-auth-btn').disabled) [
        alert('이메일 인증을 해주세요');
        return;
    ]
}

if (confirm('정보 수정을 진행합니다.')) {
    document.updateForm.submit();
} else return;
```

```
//회원정보 얻어오기
UserVO getInfo(@Param("userId") String id, @Param("paging") PageVO vo);

//회원정보 수정
void updateUser(UserVO vo);
```

로그인한 회원만 마이페이지를 사용하도록 구현했다.

로그인된 사용자의 세션데이터를 가져와서 getInfo함수를 통해 호출하고 컨트롤러의 model객체를 통해 화면에 전달해준다.

회원 정보 수정시에는 비밀번호 입력과 확인절차가 필수이다.

회원의 아이디는 변경불가하다.

이메일을 변경했다면 이메일 인증절차를 다시 진행하도록 하였다.

InfoBoardPage

Reply

```
@Controller
@RequestMapping("/infoboard")
@SLf4J
public class InfoBoardController {

    @Autowired
    private InfoBoardService service;

    //글 목록 화면
    @GetMapping("/boardList")
    public void freeList(PageVO vo, Model model) {
        PageCreator pc = new PageCreator(vo, service.getTotal(vo));
        Log.info(pc.toString()); //로그창에 log 써기
        model.addAttribute("infoBoardList", service.getList(vo));
        model.addAttribute("pc", pc);
    }

    //글쓰기 페이지 열어주는 메서드
    @GetMapping("/regist")
    public String regist() {
        return "infoboard/boardRegist";
    }

    //글 등록 처리
    @PostMapping("/regist")
    public String regist(InfoBoardVO vo) {
        service.regist(vo);
        return "redirect:/infoboard/boardList";
    }

    //글 상세 보기 처리
    @GetMapping("/content/{bno}")
    public String content(@PathVariable int bno, @ModelAttribute("p") PageVO vo,
                          Model model) {
        model.addAttribute("article", service.getContent(bno));
        return "infoboard/boardDetail";
    }

    //글 수정 처리
    @PostMapping("/modify")
    public String modify(@ModelAttribute("article") InfoBoardVO vo) {
        return "infoboard/boardModify";
    }

    //글 수정 처리
    @PostMapping("/update")
    public String update(InfoBoardVO vo) {
        service.update(vo);
        return "redirect:/infoboard/content/" + vo.getBno();
    }

    //글 삭제 처리
    @PostMapping("/delete")
    public String delete(int bno) {
        service.delete(bno);
        return "redirect:/infoboard/boardList";
    }

    <mapper namespace="com.spring.fuelplease.infoboard.mapper.IInfoBoardMapper">
        <sql id="search">
            <if test="condition == 'title'">
                WHERE title LIKE CONCAT('%', #{keyword}, '%')
            </if>
            <if test="condition == 'writer'">
                WHERE writer LIKE CONCAT('%', #{keyword}, '%')
            </if>
            <if test="condition == 'content'">
                WHERE content LIKE CONCAT('%', #{keyword}, '%')
            </if>
            <if test="condition == 'titleContent'">
                WHERE title LIKE CONCAT('%', #{keyword}, '%')
                OR content LIKE CONCAT('%', #{keyword}, '%')
            </if>
        </sql>

        <sql id="myPage">
            <if test="loginId != null">
                WHERE writer = #{loginId}
            </if>
        </sql>

        <select id="getList" resultType="board">
            SELECT
                i.*,
                (SELECT count(*) FROM inforeply
                 WHERE bno = i.bno
                ) AS reply_cnt
            FROM infoboard i
            <include refid="search" />
            ORDER BY bno DESC
            LIMIT #[pageStart], #{pageSize}
        </select>

        <insert id="regist">
            INSERT INTO infoboard
            (title, writer, content)
            VALUES(#{title}, #{writer}, #{content})
        </insert>

        <select id="getTotal" resultType="int">
            SELECT COUNT(*)
            FROM infoboard
            <include refid="search" />
            <include refid="myPage" />
        </select>

        <select id="getContent" resultType="board">
            SELECT * FROM infoboard
            WHERE bno = #{bno}
        </select>

        <update id="update">
            UPDATE infoboard
            SET title = #{title}, content = #{content}, u_date = current_timestamp
            WHERE bno = #{bno}
        </update>

        <delete id="delete">
            DELETE FROM infoboard
            WHERE bno = #{bno}
        </delete>
    </mapper>
```

```
<script>
window.onload = function() {
    document.getElementById('pagination').addEventListener('click', e => {
        e.preventDefault();
        if(!e.target.matches('a')) {
            return;
        }

        const value = e.target.dataset.pagenum;
        document.pageForm.pageNum.value = value;
        document.pageForm.submit();
    })
}

document.getElementById('title').onclick = () => {
    const id = '${loginId}';
    if(id === '') {
        alert('로그인 후 이용 가능합니다. 로그인 페이지로 이동합니다.');
    }
}

//window.onload end
</script>
```

모든 사람이 글은 볼 수 있으나 글 등록이나 수정, 삭제는 로그인한 회원만 사용이 가능하다.

게시판의 첫화면인 **boardList**는 게시글의 리스트를 불러온다. 글 상세보기는 글의 번호를 URL경로에 남겨 **boardDetail** 페이지로 전달해준다.

전달받은 글번호에 따른 상세 내용을 **Detail**화면에서 볼 수 있게 해준다.

Reply

```
@Service
public class ReplyService implements IReplyService {
    @Autowired
    private IReplyMapper mapper;

    @Override
    public void replyRegist(ReplyVO vo) {
        mapper.replyRegist(vo);
    }

    @Override
    public List<ReplyVO> getList(int bno, int pageNum) {
        PageVO vo = new PageVO();
        vo.setPageNum(pageNum);
        vo.setCpage(3);
        Map<String, Object> data = new HashMap<>();
        data.put("paging", vo);
        data.put("bno", bno);
        return mapper.getList(data);
    }

    @Override
    public int getTotal(int bno) {
        return mapper.getTotal(bno);
    }

    @Override
    public void update(ReplyVO vo) {
        mapper.update(vo);
    }

    @Override
    public void delete(int rno) {
        mapper.delete(rno);
    }
}
```

```
@Autowired
private IReplyService service;

// 댓글 등록
@PostMapping("/regist")
public String replyRegist(@RequestBody ReplyVO vo) {
    service.replyRegist(vo);
    return "regSuccess";
}

// 댓글 목록 요청(페이징 포함)
@GetMapping("/getList/{bno}/{pageNum}")
public Map<String, Object> getList(@PathVariable int bno, @PathVariable int pageNum) {
    /*
     1. getList 메서드가 글 번호, 페이지 번호를 경로에서 빼옵니다.
     2. @PathVariable 인터페이스에 복수의 값을 전달하기 위해 Map을 끌어와 @Param을 끌고 지 선택.
     3. ReplyMapper.xml에 SQL문을 페이징 쿼리로 작성.
     4. 플레이언트 측으로 DB에서 조회한 댓글 목록을 보낼 때,
     페이징을 위한 댓글의 총 개수도 함께 보내줘야 합니다.
     복수개의 값을 리턴하기 위해서 리턴 타입을 Map을 끌고, VO형식으로 끌고 지정해야합니다.
     댓글 목록 리스트와 전체 댓글 개수를 함께 전달할 예정.
    */
    List<ReplyVO> list = service.getList(bno, pageNum);
    int total = service.getTotal(bno);

    Map<String, Object> map = new HashMap<>();
    map.put("list", list); // 댓글 목록
    map.put("total", total); // 게시글에 달려있는 댓글의 총 개수

    return map;
}

// 댓글 삭제 요청
@DeleteMapping("/{rno}")
public String delete(@PathVariable int rno) {
    log.info("댓글 삭제 요청 들어옴!");
    service.delete(rno);
    return "deleteSuccess";
}
```

글 상세보기 페이지에서는 사용자들이 남긴 댓글을 무한 페이징으로 볼 수 있도록 구현하였다. 사용자의 세션에 있는 로그인 정보와 댓글을 남긴 아이디 정보와 비교하여, 본인이 작성한 글만 삭제가 가능하도록 구현하였다.

```
<script>
// 상세보기 및 수정 버튼 권한 검사 로직
const $form = document.modifyForm;

document.getElementById('btn-modify').onclick = () => {
    if (confirm('댓글 페이지로 이동합니다.')) {
        const id = '${login}';
        if (document.getElementById('writer').value === id) {
            $form.submit();
            // alert('${login}');
        } else {
            alert('본인이 작성한 글만 수정할 수 있습니다.');
        }
    } else {
        return;
    }
} // 댓글 버튼 로직

// 더보기 버튼 로직
document.getElementById('moreList').onclick = () => {
    getList(++page, false);
}

// 댓글 등록하기 버튼 로직
document.getElementById('replyRegist').onclick = () => {
    const bno = '${article.bno}';
    const reply = document.getElementById('reply').value;
    const replyId = document.getElementById('replyId').value;

    if (reply === '') {
        alert('댓글을 등록이 불가능합니다.');
        return;
    }

    if (confirm('댓글을 등록하시겠습니까?')) {
        fetch(`${pageContext.request.contextPath}/reply/regist`, {
            method: 'post',
            headers: {
                'Content-Type': 'application/json'
            },
            body: JSON.stringify({
                'bno': bno,
                'reply': reply,
                'replyId': replyId
            })
        }).then(res => res.text())
        .then(data => [
            document.getElementById('reply').value = '',
            // document.getElementById('replyId').value = '';
            getList(1, true);
        ])
    }
} // 댓글 등록하기 로직
```

BookMark

```
@GetMapping("/userBookmark")
public void userBookmark() {
}

@PostMapping("/userBookmark")
@ResponseBody
public List<String> userBookmark(HttpServletRequest session) {
    String id = (String)session.getAttribute("login");
    return sv.userBookmark(id);
}

@PostMapping("/showBookmark")
@ResponseBody
public BookMarkVO showBookmark(HttpServletRequest session, @RequestBody String bkaddr) {
    Log.info(bkaddr);
    String id = (String)session.getAttribute("login");
    Log.info(id);
    return sv.showBookmark(bkaddr, id);
}

@PostMapping("/deleteBookmark")
@ResponseBody
public void deleteBookmark(HttpServletRequest session, @RequestBody String bkaddr) {
    String id = (String)session.getAttribute("login");
    sv.deleteBookmark(id, bkaddr);
}
```

```
<select id="userBookmark" resultType="string">
    select bkaddr from bookmark
    where bkuser_id = #{id}
</select>

<select id="showBookmark" resultType="bookmark">
    select bkaddr, bktel, bkname from bookmark
    where bkaddr = #{bkaddr} and bkuser_id = #{userId}
</select>

<delete id="deleteBookmark">
    delete from bookmark
    where bkaddr = #{bkaddr} and bkuser_id = #{userId}
</delete>
```

```
@Override
public List<String> userBookmark(String id) {
    return mp.userBookmark(id);
}

@Override
public BookMarkVO showBookmark(String id, String bkaddr) {
    Log.info(bkaddr);
    return mp.showBookmark(bkaddr, id);
}

@Override
public void deleteBookmark(String id, String bkaddr) {
    mp.deleteBookmark(id, bkaddr);
}
```

```
fetch('${pageContext.request.contextPath}/user/showBookmark', {
    method: 'post',
    headers: {
        'Content-type': 'application/json'
    },
    body: e.target.textContent
})
.then(res => res.json())
.then(data => {
    console.log(data);
    document.getElementById('addr').textContent = data.bkaddr;
    document.getElementById('tel').textContent = data.bktel;
    document.getElementById('name').textContent = data.bkname;
    name = data.bkname;
})
```

```
window.onload = function () {
    let selectBook = document.getElementById('bookbox');

    fetch('${pageContext.request.contextPath}/user/userBookmark', {
        method: 'post'
    })
    .then(res => res.json())
    .then(data => {
        console.log(data);
        for (var i = 0; i < data.length; i++) {
            var opt = document.createElement('a')
            var hr = document.createElement('hr')
            opt.textContent = data[i];
            opt.style.display = 'block';
            opt.setAttribute('value', data[i]);
            selectBook.appendChild(opt);
            selectBook.appendChild(hr);
        }
    })
}
```

```
document.getElementById('searchBtn').onclick = function () {
    let selectBook = document.getElementById('bookbox');
    let addr = document.getElementById('addr').textContent;
    if (addr === '') {
        alert('삭제할 목록을 선택해주세요')
    } else {
        if (confirm('삭제하시겠습니까?')) {
            fetch('${pageContext.request.contextPath}/user/deleteBookmark', {
                method: 'post',
                headers: {
                    'Content-type': 'text/plain'
                },
                body: addr
            })
            .then(res => res.text())
            .then(data => {
                location.href = '${pageContext.request.contextPath}/user/userBookmark';
            })
        }
    }
}
```

회원은 즐겨찾기 서비스를 mapView의 페이지들에서 즐겨찾기를 추가한다.

그럼, Bookmark페이지에서 자신이 추가한 즐겨찾기목록을 볼 수 있다.

먼저 화면이 onload되면 회원의 아이디정보로 추가된 주소값들을 비동기 방식을 통해 불러오고,

그 목록들 중 하나를 선택하면 지도에 검색이 된다.

또한, 삭제를 누를경우 회원의 아이디정보에 남겨있는 주소가 삭제가 된다.

Car Account

```

<!-- 등록하기 -->
<insert id="regist">
    INSERT INTO caraccount
    (writer, year, month, day, type, price, note)
    VALUES
    (#writer, #{year}, #{month}, #{day}, #{type}, #{price}, #{note})
</insert>

<!-- 목록 불러오기 -->
<select id="getList" resultType="caraccount">
    SELECT * FROM caraccount
    WHERE writer = #{loginId}
    ORDER BY acno DESC
    LIMIT #{pageStart}, #{perPage}
</select>

<!-- 총 등록수 구하기 -->
<select id="getTotal" resultType="int">
    SELECT COUNT(*) FROM caraccount
    WHERE writer = #{loginId}
</select>

<!-- 삭제하기 -->
<delete id="delete">
    DELETE FROM caraccount
    WHERE acno = #{acno}
</delete>

<!-- 가격 총합 -->
<select id="getPriceTotal" resultType="string">
    select sum(price) from caraccount
    where writer = #{writer};
</select>

<!-- 가격 평균 -->
<select id="getPriceAvg" resultType="String">
    select TRUNCATE(avg(price),0) from caraccount
    where writer = #{writer};
</select>

```

게시판과 비슷한 구조이다. 다만, 리스트를 불러올 때 로그인한 회원의 아이디를 기반으로 목록을 볼 수 있다.
회원은 차계부를 등록하고, 등록된 가격의 합계와 평균을 확인 할 수 있다.

```

// 차계부 페이지 이름 요청
@GetMapping("/accountList")
public void accountList(HttpServletRequest session, PageVO vo, Model model) {
    String id = (String) session.getAttribute("login");
    vo.setLoginId(id);
    PageCreator apc = new PageCreator(vo, acsv.getTotal(vo));
    model.addAttribute("accountlist", acsv.getList(vo));
    model.addAttribute("apc", apc);

    String total = acsv.getPriceTotal(id);
    log.info("total" + total);
    String avg = acsv.getPriceAvg(id);

    if(total != null) {
        model.addAttribute("total", acsv.getPriceTotal(id));
        model.addAttribute("avg", acsv.getPriceAvg(id));
    }
}

// 등록 페이지 열기
@GetMapping("/regist")
public String regist() {
    return "/caraccount/accountRegist";
}

// 차계부 등록
@PostMapping("/regist")
public String regist(HttpServletRequest session, CarAccountVO vo) {
    String id = (String) session.getAttribute("login");
    vo.setWriter(id);
    sv.regist(vo);
    return "redirect:/caraccount/accountList";
}

// 삭제 처리
@PostMapping("/delete")
@ResponseBody
public String delete(@RequestBody int acno) {
    sv.delete(acno);
    return "deleteSuccess";
}

```

```

<!-- 사용금액 합계 사용 부분 -->
<c:if test="${total != null}">
    <tr>
        <th colspan="3">내가 주유한 총 금액</th>
        <th colspan="3" style="display: none;">${total}</th>
        <th colspan="3" id="realTotal"></th>
    </tr>
</c:if>
<c:if test="${total == null}">
    <tr>
        <td colspan="3">
    </tr>
</c:if>

<c:if test="${avg != null}">
    <tr>
        <th colspan="3">나의 평균 주유비용</th>
        <th colspan="3" style="display: none;">${avg}</th>
        <th colspan="3" id="realAvg"></th>
    </tr>
</c:if>
<c:if test="${avg == null}">
    <tr>
        <td colspan="3">
    </tr>
</c:if>

```

```

window.onload = function () {
    const total = '${total}';
    const avg = '${avg}';

    console.log(total.replace(/\B(=(\d{3})+(?!d))/g, ','));
    document.getElementById('realTotal').textContent = total.replace(/\B(=(\d{3})+(?!d))/g, ',') + ' 원';
    document.getElementById('realAvg').textContent = avg.replace(/\B(=(\d{3})+(?!d))/g, ',') + ' 원';

    document.getElementById('pagination').addEventListener('click', e => {
        e.preventDefault();
        if (!e.target.matches('a')) {
            return;
        }

        const value = e.target.dataset.pagenum;
        document.pageForm.pageNum.value = value;
        document.pageForm.submit();
    });
}

```

MapView

```
//주유소
@GetMapping("/gasolineMap")
public void findGasoline() {
    log.info("주유소 맵으로~");
}

//~~구의 도로명을 가져오기
@PostMapping("/gasolineMap")
@ResponseBody
public List<String> findGasoline(@RequestBody String keyword) {
    List<String> spot = gsv.findGasoline(keyword);
    log.info(spot);
    model.addAttribute("spot", spot);
    return spot;
}

@PostMapping("/addBookmarkGas")
@ResponseBody
public void addBookmarkGas(@RequestBody BookMarkVO vo, HttpSession session) {
    vo.setBkuserId((String)session.getAttribute("login"));
    gsv.addBookmarkGas(vo);
}

@PostMapping("/bookCheck")
@ResponseBody
public int bookCheck(@RequestBody String bkaddr, HttpSession session) {
    String id = (String)session.getAttribute("login");
    if(gsv.bookCheck(bkaddr, id)>=1)
        return 1;
    else
        return 0;
}

@PostMapping("/searchResult")
public String searchResult(String searchword) {
    return "redirect:/mapview/searchResult";
}
```

```
@Override
public List<String> findGasoline(String keyword) {
    log.info(keyword);
    return mp.findGasoline(keyword);
}

@Override
public GasolineVO getGasolineInfo(String loadId) {
    log.info(loadId);
    return mp.getGasolineInfo(loadId);
}

@Override
public void addBookmarkGas(BookMarkVO vo) {
    mp.addBookmarkGas(vo);
}

@Override
public int bookCheck(String bkaddr, String bkuserId) {
    if(mp.bookCheck(bkaddr, bkuserId)>=1)
        return 1;
    else
        return 0;
}
```

```
if (gasAddr === '') {
    alert('장소를 검색해주세요.')
} else {
    fetch(`${pageContext.request.contextPath}/mapview/bookCheck`, {
        method: 'post',
        headers: {
            'Content-type': 'text/plain'
        },
        body: gasAddr
    })
    .then(res => res.text())
    .then(data => {
        console.log(data);

        if (data === '1') {
            alert('이미 즐겨찾기에 추가되어있습니다.')
        } else {
            if (confirm('즐겨찾기에 추가하시겠습니까?')) {
                fetch(`${pageContext.request.contextPath}/mapview/addBookmarkGas`, {
                    method: 'post',
                    headers: {
                        'Content-type': 'application/json'
                    },
                    body: JSON.stringify({
                        'bkname' : gasName,
                        'bkaddr' : gasAddr,
                        'bktel' : gasNo
                    })
                })
                .then(res => res.json())
                .then(checkbook => {
                    console.log(checkbook);
                })
                alert('즐겨찾기에 추가되었습니다.')
            }
        }
    })
}
```

```
function getGasolineinfo() {
    const loadId = document.getElementById('selectLoad').value;
    console.log(loadId); //사용자가 선택한 도로명 주소 추출

    fetch(`${pageContext.request.contextPath}/mapview/gasolineMapInfo`, {
        method: 'post',
        headers: [
            'Content-type': 'text/plain'
        ],
        body: loadId
    })
    .then(res => res.json())
    .then(data => {
        console.log(data);

        document.getElementById('gasName').textContent = data.bplcnnm;
        document.getElementById('gasAddr').textContent = data.rdnwhladdr;
        document.getElementById('gasNo').textContent = data.sitetel;
    })
}
```

```
//키워드 넣기
document.getElementById('selectCounty').onclick = function () {
    document.getElementById('selectLoad').replaceChildren();
    var opt = document.createElement('option');
    opt.textContent = '도로명을 선택해주세요';
    document.getElementById('selectLoad').appendChild(opt);
}

document.getElementById('selectCounty').onchange = function () {
    let selectCounty = document.getElementById('selectCounty').value;
    console.log(selectCounty);

    fetch(`${pageContext.request.contextPath}/mapview/gasolineMap`, {
        method: 'post',
        headers: {
            'Content-type': 'text/plain'
        },
        body: selectCounty
    })
    .then(res => res.json())
    .then(data => {
        console.log(data);
        for (var i = 0; i < data.length - 1; i++) {
            var opt = document.createElement('option');
            opt.textContent = data[i];
            opt.setAttribute('value', data[i]);
            document.getElementById('selectLoad').appendChild(opt);
        }
    })
}
```

```
<div class="stationbar">
<select class="form-control input-s
    <option>구 선택</option>
    <option>강남구</option>
    <option>강동구</option>
    <option>강서구</option>
    <option>강북구</option>
    <option>관악구</option>
    <option>광진구</option>
    <option>구로구</option>
    <option>금천구</option>
    <option>노원구</option>
    <option>동대문구</option>
    <option>도봉구</option>
    <option>동작구</option>
    <option>마포구</option>
    <option>서대문구</option>
    <option>성동구</option>
    <option>성북구</option>
    <option>서초구</option>
    <option>송파구</option>
    <option>영등포구</option>
    <option>용산구</option>
    <option>양천구</option>
    <option>은평구</option>
    <option>종로구</option>
    <option>중구</option>
    <option>중랑구</option>
</select>
```

MapView의 주유소찾기이다. 위의 옵션을 선택하면 비동기 방식으로 DB에서 조회를 한 후 해당하는 구의 도로명 주소를 모두 불러와서 옵션을 생성한다. 선택된 두 옵션의 값을 합쳐 카카오 지도 api에 전달해주면 지도에 마커가 찍힌다.

또한, 즐겨찾기 추가 버튼을 눌렀을 때 해당 주소가 회원정보로 작성된 즐겨찾기가 있다면 리턴값을 통해 중복 추가를 방지한다.

FuelAvg

```
private String apiKey;  
  
// tag의 정보를 가져오는 함수  
public static String getTagValue(String tag, Element eElement) {  
  
    // 결과를 저장할 result 변수 선언  
    String result = "";  
    NodeList nlList = eElement.getElementsByTagName(tag).item(0).getChildNodes();  
    result = nlList.item(0).getTextContent();  
    return result;  
  
}  
  
// tag의 정보를 가져오는 함수  
public static String getTagValue(String tag, String childTag, Element eElement) {  
  
    // 결과를 저장할 result 변수 선언  
    String result = "";  
    NodeList nlList = eElement.getElementsByTagName(tag).item(0).getChildNodes();  
    for (int i = 0; i < eElement.getElementsByTagName(childTag).getLength(); i++) {  
        // result += nlList.item(i).getFirstChild().getTextContent() + " ";  
        result += nlList.item(i).getChildNodes().item(0).getTextContent() + " ";  
    }  
    return result;  
}
```

```
// 휘발유 최저가  
public List<String> getMinGasInfo() {  
  
    try {  
        // parsing할 때 API 키 포함해서  
        String url = "https://www.opinet.co.kr/api/lwTop10.do?out=xml&code="+ApiKey+"&prodcd=B027&area=01&cnt=1";  
  
        List<String> gasList = new ArrayList<>();  
  
        DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();  
        DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();  
        Document doc = dBuilder.parse(url);  
  
        // 제일 첫번째 태그  
        doc.getDocumentElement().normalize();  
  
        // 파싱할 tag  
        NodeList nList = doc.getElementsByTagName("OIL");  
  
        int temp = 0;  
        Node nNode = nList.item(temp);  
        Element eElement = (Element) nNode;  
        for (temp = 0; temp < nList.getLength(); temp++) {  
            log.info("최저가 휘발유 가격: " + getTagValue("PRICE", eElement));  
            log.info("최저가 주유소 이름: " + getTagValue("OS_NM", eElement));  
            log.info("최저가 주유소 주소(구만 일어남): " + getTagValue("VAN_ADDR", eElement).split(" ", 3)[1]); // 구 만 잘라서 사용  
            gasList.add(getTagValue("OS_NM", eElement));  
            gasList.add(getTagValue("PRICE", eElement));  
            gasList.add(getTagValue("VAN_ADDR", eElement).split(" ", 3)[1]);  
        }  
        return gasList;  
  
    } catch (Exception e) {  
        e.printStackTrace();  
        return null;  
    }  
}
```

```
<!-- 평균가격 뽑아내기 -->  
<div class="avgTotal" id="goodInfo">  
    <h4>이번주의 평균가격 (서울기준)</h4>  
  
<div class="gasavgInfo">  
    <h5>휘발유</h5>  
    <p><span id="price">${gas}</span> 원 &ampnbsp &ampnbsp &ampnbsp</p>  
  
    <h5>고급휘발유</h5>  
    <p><span id="price">${preGas}</span> 원 &ampnbsp &ampnbsp &ampnbsp</p>  
  
    <h5>경유</h5>  
    <p><span id="price">${dis}</span> 원 &ampnbsp &ampnbsp &ampnbsp</p>  
</div>
```

```
<h4>오늘의 최저가 정보 (서울기준)</h4>  
  
<div class="minInfo">  
    <div>  
        <h5>휘발유</h5>  
        <p>상호명 : ${minGas[0]} (${minGas[2]})</p>  
        <p>가격 : <span id="price">${minGas[1]}</span> 원</p>  
    </div>  
    <div>  
        <h5>고급 휘발유</h5>  
        <p>상호명 : ${minPreGas[0]} (${minPreGas[2]})</p>  
        <p>가격 : <span id="price">${minPreGas[1]}</span> 원</p>  
    </div>  
    <div>  
        <h5>경유</h5>  
        <p>상호명 : ${minDis[0]} (${minDis[2]})</p>  
        <p>가격 : <span id="price">${minDis[1]}</span> 원</p>  
    </div>  
</div>  
  
<div class="avgTotal" style="display: none;" id="badInfo">  
    <h4>가격 정보를 업데이트 중입니다.</h4>  
    <h4>문의는 아래 메일로 보내주시면 감사하겠습니다.</h4>  
    <h4>kim1234@naver.com</h4>  
</div>
```

```
// 휘발유  
public String getGasAvg() {  
  
    try {  
        // parsing할 때 API 키 포함해서  
        String url = "https://www.opinet.co.kr/api/avgLastWeek.do?prodcd=B027&code="+ApiKey+"&sido=01&out=xml";  
  
        DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();  
        DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();  
        Document doc = dBuilder.parse(url);  
  
        // 제일 첫번째 태그  
        doc.getDocumentElement().normalize();  
  
        // 파싱할 tag  
        NodeList nList = doc.getElementsByTagName("OIL");  
  
        int temp = 0;  
        Node nNode = nList.item(temp);  
        Element eElement = (Element) nNode;  
        for (temp = 0; temp < nList.getLength(); temp++) {  
  
            log.info("휘발유 평균가격: " + getTagValue("PRICE", eElement));  
        }  
        return getTagValue("PRICE", eElement);  
  
    } catch (Exception e) {  
        e.printStackTrace();  
        return null;  
    }  
}
```

(주)석유동향-오피넷의 api를 사용하여 최저가, 평균유가에 대해 작성하였다.
api사용시 XML로 전달되는 데이터를 **getTagValue**함수를 통해 정보를 불러오고
각 태그별 값들을 리스트에 담아 리턴을 시켰다.
이를 홈페이지 화면에 전달한다.

Evaluation Opinion

기본적인 웹 페이지의 원리에 대해 이해하였다.

이번 프로젝트를 진행하면서 API의 적절한 사용법에 대해 연구하였다.

API에 대해 이해도를 한층 높였다.

JSON 데이터와 XML 데이터를 활용하며 익숙해지는 시간이었다.

프로젝트를 기획하면서 실질적으로 사용자의 입장이 되었을 때, 정말 필요했던 기능들을 고민하며 한층 요구사항 분석에 대해 깊이 접근하는 방법을 고찰했다.

**무료 API 사용으로 기능적인 부분에 한계가 있었지만, 완벽히 잘 사용하였다.
목업을 기반으로한 페이지 디자인 구현이 원하는 스타일로 잘 구현되었다.**

Evaluation

황우신

API를 처음 사용해보면서 공공데이터의 형태와 우리가 만든 서비스가 유사페이지와의 다른점에 대해 고찰하고 보완할 점을 찾아 더 나은 서비스 방안을 생각하게 되었다.

이미 시중에 서비스 되고있는 페이지가 존재하다보니 그와 다른 점을 서비스 하기 노력했고, 자연스럽게 고객의 입장에서 더 나은 서비스로 어떤 점을 제공하면 좋을지 요구사항 수용에 대한 부분을 더 크게 생각하게 된 계기가 된 것 같다.

문창주

프로젝트를 진행하면서 **fullcalendar**라는 강력한 기능의 달력 api를 커스텀 하여 적용 시켜 볼 기회가 있었다.

전체적인 기능은 그대로 쓰되 간단하게 커스텀을 해보려 하였으나 구조적으로 너무 복잡하였고,

제한된 프로젝트 수행시간으로 인해 전체 코드를 분석 해 볼 시간이 부족하여 결국엔 사용하지 못했던 점이 너무 아쉬웠다.

이번 프로젝트를 통하여 api의 강력함과 유연성은 적재적소에 활용하면 정말 좋은 기능을 구현 할 수 있었다는 점을 알게 되었고, 부족했던 부분을 좀 더 개선시켜 다음 기회에는 꼭 성공시키고 싶다.

Evaluation

최지혁

중간 프로젝트를 통해 공공API를 활용하여 다양한 정보를 데이터베이스에 저장하고 가공하는 방법에 대해 많은 시행착오를 겪으며 숙지할 수 있었다. 이를 통해 다음 프로젝트에서는 더 수월하게 API를 활용할 수 있을 것 같다.

게시판을 만들며 여러 고려사항들을 보완하고 페이징처리 등의 로직을 더욱 효과적으로 사용할 수 있게 되었다.

또한, 팀원들과의 커뮤니케이션과 깃허브를 통한 협업 방법에 대해 많은 것을 배웠다. 서로 도움을 주고받으며 프로젝트를 진행하는 과정에서 깃 사용 및 기능 로직 구현에 대해 더욱 능숙해졌다. 이번 프로젝트에서는 아쉽게도 어려운 기능들을 많이 추가하지 못했다. 그러나 중간 프로젝트의 경험을 바탕으로 앞으로의 지속적인 발전을 위해 노력하고 앞으로는 우리가 배운 것들을 토대로 더욱 어려운 기능들을 구현해 나갈 수 있을 것 같다.

김지원

이번 프로젝트에서 각 페이지의 프론트엔드를 주로 만들었고 처음 사용하는 부트스트랩을 적용하는 법을 공부해가며 프론트 엔드를 구성하였다. 프론트엔드를 구성할 때 실제 사용자의 입장이 되어 어떤식으로 화면을 구성해야 사용자들이 편리하게 서비스를 이용할 수 있는지를 많이 생각했고 실제로 주유소 찾는 사이트이기 때문에 깔끔한것이 중요하다고 생각했다. 이 생각들을 토대로 목업들을 구성해갔으며 이 목업들에서 벗어나지 않을려고 노력을 많이 한 결과 목업대로 페이지 프론트엔드가 구현되었다. 부트스트랩을 적용하는 과정을 거치면서 템플릿을 끌고와서 페이지에 적용하는 과정을 배웠고 결과적으로 더 많은 부트스트랩 기능을 적용할 수 있었는데 그러지 못한거 같아서 아쉬웠다. 조원들과 깃 작업에 대해서 같이 공부해가며 협업 시 깃 사용에 대해 능숙해진것 같고 앞으로 이번 프로젝트에서 있던 과정들을 토대로 한 층 성장한 것 같다.