



MVC 패턴2 기반 반응형 중고차 웹 사이트 - VARCHAR

코알라

황지민, 이향준, 김종현, 김수연, 임환욱, 이준선



1

프로젝트 개요

- 기획 및 개요, 역할 분담, 개발환경, 일정, 버전관리, 웹 페이지 구성

2

설계

- ERD, UserFlow, LogicProcess

3

기능

- 주요기능(코드 설명), API, Plugin

4

시연

5

마무리



- 기획 배경
- 역할 분담
- 개발 환경
- 개발 일정
- 버전 관리
- 웹 페이지 구성
- 반응형 웹

← 기획 배경 및 개요

VARCHAR

VARCHAR

4 page



• 기획 배경

- 최근 반도체 대란과 원가 상승으로 온라인 중고차 시장이 활발

• 기획 목적

- 섬세한 검색 기능을 갖춘 사이트 구현

• 기대 효과

- 의사소통 능력 향상
- MVC2 패턴 개념
- MVC2 패턴 기반의 검색 기능, 게시판 기능 구현 능력
- 다양한 API 구현



코알라 조 : 코딩 알려줘라



김수연

메인 : MODEL
서브 : VIEW
주 담당 : 문자 API
이메일 API
주소 API



임환욱

메인 : MODEL
서브 : CTRL
주 담당 : 필터 검색 기능
ppt 제작, 발표



이준선

메인 : VIEW
서브 : MODEL
주 담당 : 장바구니 기능 구현
ppt 제작, 발표



김종현

메인 : VIEW
서브 : CTRL
주 담당 : Front 총괄
모든 C 작업 참여



이향준

메인 : CTRL
서브 : MODEL
주 담당 : 필터 검색 기능



황지민

메인 : CTRL
서브 : MODEL
주 담당 : 프로젝트 총괄



통합개발환경(IDE)



사용 언어



DBMS

ORACLE

소통 및 협업



Server



버전관리





개발 일정

VARCHAR



7 page





버전 관리

VARCHAR

VARCHAR

8 page



8/12~8/16

1.0.0V

- 웹 템플릿 설정

8/24

2.0.1V

- 1자 버그 수정
- 필터 검색 기능 SQL 문 수정

8/30

2.2.0V

- 필터 검색 기능 SQL 문 전체 수정
- 문의 메시판 답글 기능 추가

9/01

2.4.0V

- Email API 기능 구현
- 장바구니 삭제 기능 추가

9/04

2.4.2V

- 각종 버그 수정
- 컨트롤러 버전 업

8/18~8/21

2.0.0V 주제 변경 이슈

- 웹 템플릿 설정
- 필터 검색, 장바구니 모달창 구현
- 차량 목록 크롤링 구현

8/27

2.1.0V

- 지도 API 추가
- 회원가입, 로그인 유효성 검사
- 필터 검색 기능 추가

8/31

2.3.0V

- 문자 API 추가
- 필터 검색 범위 슬라이더 추가
- Model 버전 업

9/03

2.4.1V

- 필터 검색 코드 수정

9/06

2.4.3V

- 코드 확인,
- 주석 관리

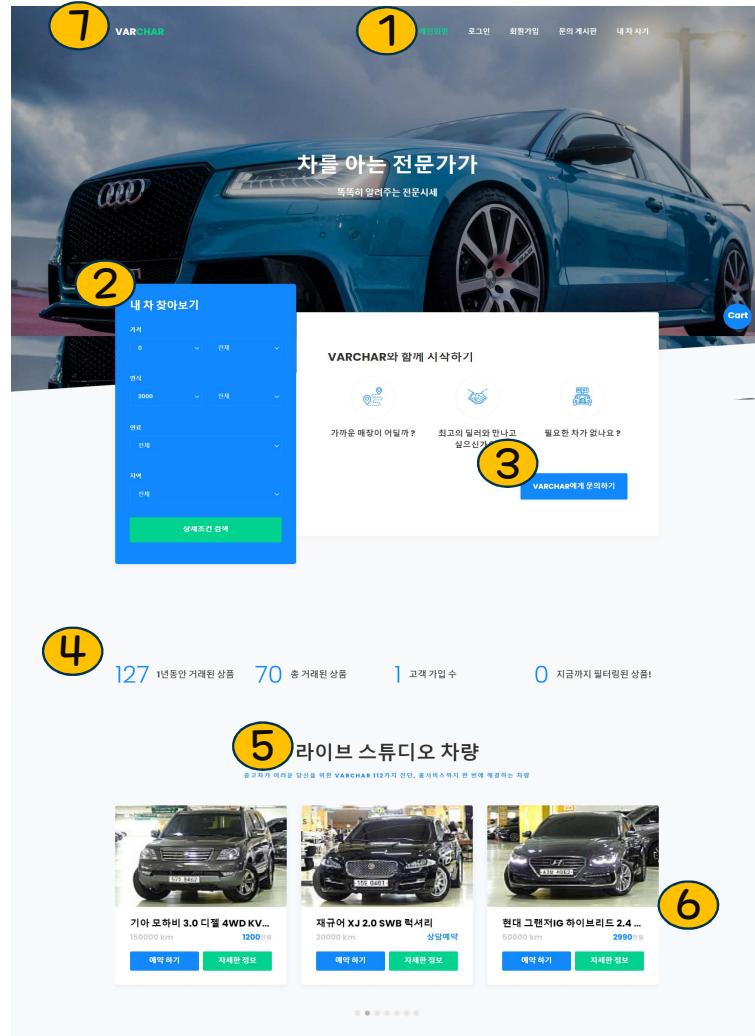


웹 페이지 구성[0] – Main

VARCHAR

VARCHAR

9 page

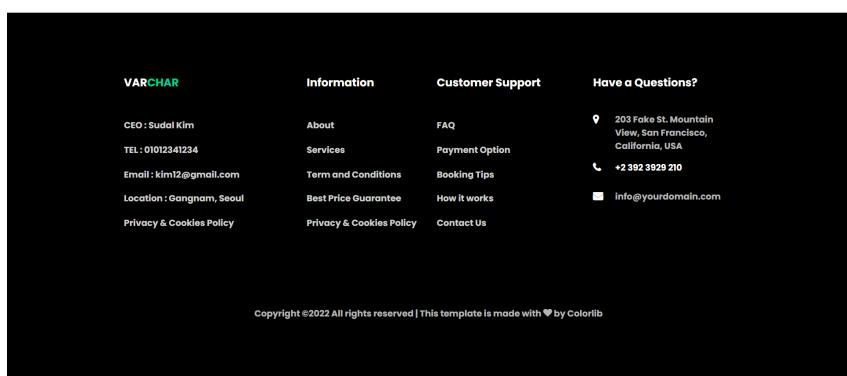
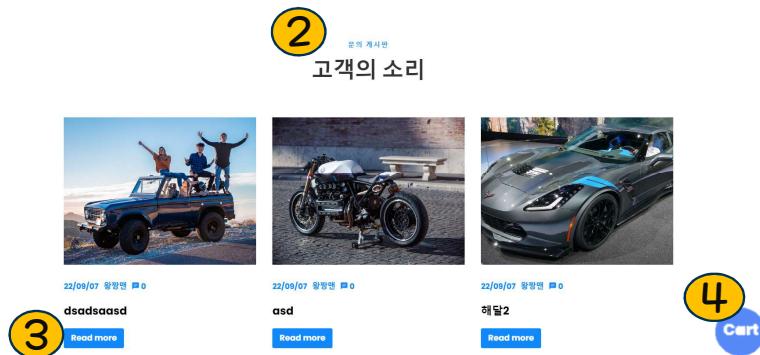
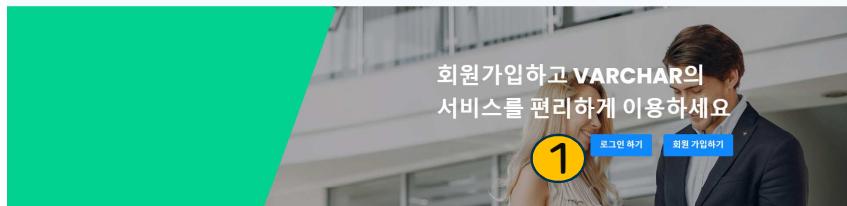


DESIGN Description

- 1** 과 페이지 이동 메뉴바
- 2** 상품 검색 품
- 3** 문의 게시판 이동 버튼
- 4** 각종 통계 표현
- 5** VARCHAR가 보유한 차량 랭킹 슬라이더
- 6** 짐목록 확인 모달창
- 7** 사이트명 버튼

DEVELOP Description

- 1** 로그인 시 마이페이지, 로그아웃 버튼 생성
- 2** 지역, 가격, 연식 검색 등 필터 검색
- 3** 문의 게시판 페이지로 이동
- 4** 지금까지 판매차량, 회원 수, 보유차량 대수 출력
- 5** 보유 차량 중에서 랭덤하게 출력
- 6** 모든 페이지 우측 하단 고정
클릭 시 소멸, 우측하단에 짐 목록 모달창 생성
- 7** 메인화면으로 이동



DESIGN Description

1 로그인, 회원가입 버튼

2 최근 작성된 문의 글 3개 출력

3 문의 게시판 버튼

4 짐목록 모달창

5

DEVELOP Description

1 로그인, 회원가입 페이지 이동

2 최근 작성된 문의 글 3개 출력

3 문의게시판으로 이동

4 회원관련 페이지를 제외한 모든 페이지 활성화

5



웹 페이지 구성[2] – Login, ID/PW찾기

VARCHAR

VARCHAR

11 page



아이디
아이디

비밀번호
비밀번호

로그인

1 아이디/비밀번호 찾기

아직 회원이 아니세요?

2 회원가입

회원가입을 통해 포인트 적립, 회원전용혜택 등
더 많은 서비스를 이용하실 수 있습니다.

© VARCHAR. All Rights Reserved.

아이디 찾기 비밀번호 찾기

이름
이름(실명)

이메일
이메일 주소

3 아이디 복송

4 **로그인**

회원가입 시 등록한 이름, 이메일을 입력해주세요!

아직 회원이 아니세요?

회원가입

회원가입을 통해 포인트 적립, 회원전용혜택 등
더 많은 서비스를 이용하실 수 있습니다.

© VARCHAR. All Rights Reserved.

DESIGN Description

- 1 **아이디/비밀번호 찾기 버튼**
- 2 **회원가입 버튼**
- 3 **아이디발송 버튼**
- 4 **로그인 버튼**
- 5

DEVELOP Description

- 1 **아이디/비밀번호 찾기 페이지로 이동**
- 2 **회원가입 페이지로 이동**
- 3 **Email API 사용**
- 4 **로그인 화면으로 이동**
- 5



웹 페이지 구성[3] – 회원가입

VARCHAR

VARCHAR

12 page



1 중복확인

2

3 주소찾기

4 인증번호 발송

5

DESIGN Description

1 아이디 중복 검사 버튼

2 사용자 정보 입력란

3 주소 찾기 버튼

4 휴대폰 인증번호 발송 버튼

5 약관동의 버튼

DEVELOP Description

1 Ajax 비동기처리 방식으로 구현

2 정규표현식으로 유효성검사

3 카카오맵 API 사용

4 문자 API 사용

5 필수약관 체크시 가입완료 버튼 활성화



웹 페이지 구성[4] – 차량보기

VARCHAR

VARCHAR

13 page



아우디 R8 5.2 V10 퍼포먼스 쿠페
2000 km
상담예약



람보르기니 우라칸 LP640-4 ...
10000 km
상담예약



롤스로이스 팬텀 6.7 V12
20000 km
상담예약



포르쉐 마칸 2.9 S
25000 km
상담예약



제네시스 G90 3.5 터보 AWD ...
600000 km
12650만원
상담예약



현대 e-에어로타운 캠핑카
160000 km
4880만원
상담예약



총 12 개의 상품을 보고 있습니다

view more

아우디 R8 5.2 V10 퍼포먼스 쿠페
2000 km
예약 하기 **자세한 정보**

람보르기니 우라칸 LP640-4 ...
10000 km
예약 하기 **자세한 정보**

롤스로이스 팬텀 6.7 V12
20000 km
예약 하기 **자세한 정보**

포르쉐 마칸 2.9 S
25000 km
예약 하기 **자세한 정보**

제네시스 G90 3.5 터보 AWD ...
600000 km
12650만원
예약 하기 **자세한 정보**

현대 e-에어로타운 캠핑카
160000 km
4880만원
예약 하기 **자세한 정보**

총 상품 금액 **0만원**

Cart

Cart (2)

<p>람보르기니 우루스 4.0 V8 펠 캡슐 연식 : 2022 연료 : 가솔린 주행거리 : 49000km 지역 : 경기 가격: 상담예약</p>
<p>벤츠 스프린터 519 CDI 투어 리 엑스트라 L 연식 : 2020 연료 : 디젤 주행거리 : 1000km 지역 : 경북 가격: 상담예약</p>

DESIGN Description

1 예약하기 버튼

2 모달창 활성화

3 해당 차량 상세 정보 버튼

4 필터검색 모달창

5

DEVELOP Description

1 짐목록에 저장

2 짐한 차량 출력, 총 상품금액 출력 및 짐한 상품 삭제

3 상품 정보 페이지로 이동

4 차량목록 페이지에서 필터검색 좌측 하단 고정
클릭 시 소멸, 좌측하단에 필터검색 모달창 생성

5



웹 페이지 구성[5] – 차량보기.필터검색

VARCHAR

VARCHAR

14 page

Close Filter

상품 정렬

최신순 ▾

연식

2017 ~ 2022

연료

전체 LPG 휘발유
경유 CNG 전기

주행거리

최소: 1000 km 최대: 700000 km

가격

2000 ~ 6000

지역

전체 서울 경기
인천 강원 충북
충남 대전 세종
경북 경남 대구
부산 울산 전북
전남 광주 제주

상세조건 검색

초기화

A large black arrow originates from the green circular 'filter' button at the bottom left of the filter panel. It points upwards and to the right, passing over the '주행거리' section, and ends at the blue rectangular '상세조건 검색' button.

① filter

② 상세조건 검색

③ 초기화

DESIGN Description

1 필터검색 모달창

2 상세조건 검색 버튼

3 초기화 버튼

4

5

DEVELOP Description

1 모달창 활성화

2 검색 후, 검색조건 유지

3 모든 값 초기 상태

4

5



랜드로버 레인지로버 4.4 P530 LWB 오토바이오그래피 7인승 상담 예약

73000km



주행거리
73000km



가격
상당예약
2023

1

찜하기

2

주요사항
딜러의 한마디

차량 주요 특징: [정식/무주행 신차/리어 액슬 스티어링/전동 사이드스텝]

VARCHAR가 인증하는 차량! 10년 이상 경력의 전문 딜러가 당신의 소중한 차를 꼼꼼히 챙겨합니다!

위 [랜드로버 레인지로버 4.4 P530 LWB 오토바이오그래피 7인승] 차량에 대한 딜러의 의견도 확인하세요!

<랜드로버 레인지로버 4.4 P530 LWB 오토바이오그래피 7인승> 차량의 진단 평가 결과



- 후드
- 프론트휀더
- 도어
- 사이드실제널
- 트렁크리드
- 루프휀델
- 워터휀델

정상
정상
정상
정상
정상
정상
정상



- 프론트휀델
- 크로스멤버
- 인사이드휀델
- 사이드휀버
- 힐하우스
- 패킹지트레이
- 대수휀델
- 플로어휀델
- 밸류휀델

정상
정상
정상
정상
정상
정상
정상
정상
정상

차를 잘 몰라서 불안하신가요?
걱정하지 마세요! 차를 아는 전문가가 직접 확인하고 진단한 차량입니다.
오늘별 상세 진단서의 정보와 실제 차량 상태가 상이할 경우, 최대 150만원까지 보상해 드립니다.
(구매 확정 후 3개월 이내, 주행거리 6,000km 이내 차량에 한함)

3

이 차량을 VARCHAR 가 보증합니다

4

뒤로가기
메인으로가기

DESIGN Description

1 찜하기 버튼

2 주요사항 / 딜러의 한마디 버튼

3 VARCHAR 로고 버튼

4 뒤로가기 / 메인으로가기 버튼

5

DEVELOP Description

1 해당 상품 찜목록에 추가

2 차량 정보 또는 딜러의 멘트 출력

3 메인 화면으로 이동

4 전 페이지로 이동 또는 메인 화면으로 이동

5

Address:
서울특별시 강남구 역삼동 736-7

Phone:
02-1234-1234

Email:
sudall234@koala.com

제목: gosufam123
내용: 게시글을 작성해보세요!

게시글 작성

제목 ①

검색

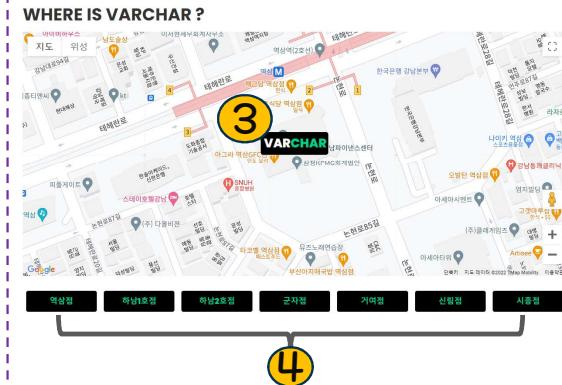
드릴 말씀이 있습니다.
(gosufam123님, 22/09/07)

나는 아고속도로
(gosufam123님, 22/09/07)

취업준비생이고, 경차 사려합니다~
(gosufam123님, 22/09/07)

아우디 A5 구매희망합니다.
(admin1234님, 22/09/07)

② view more

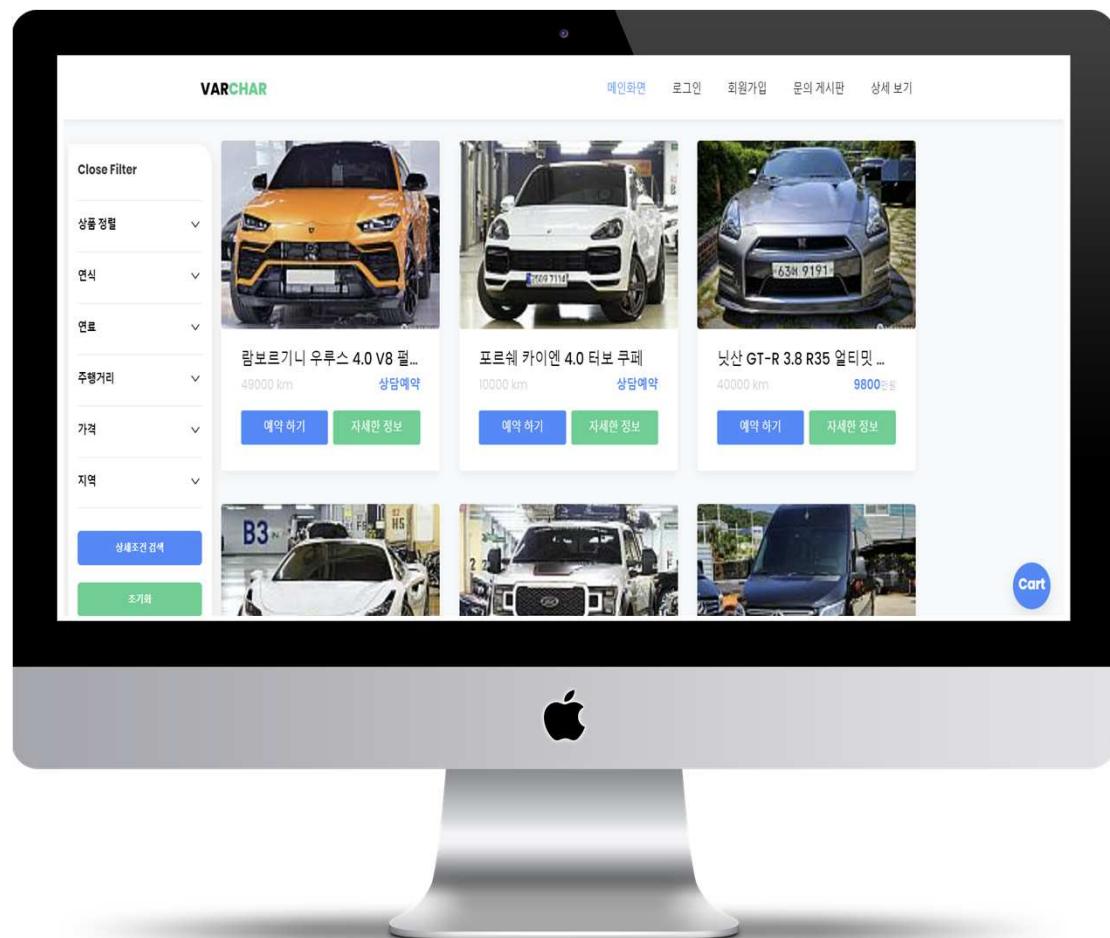
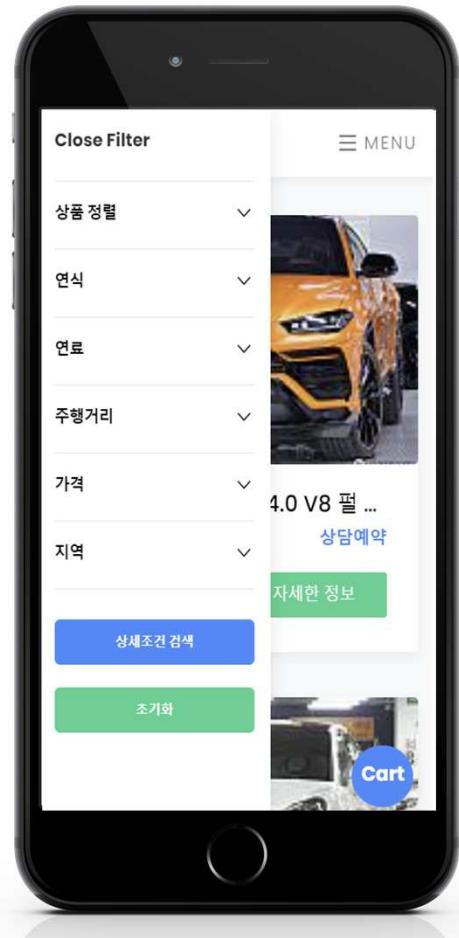
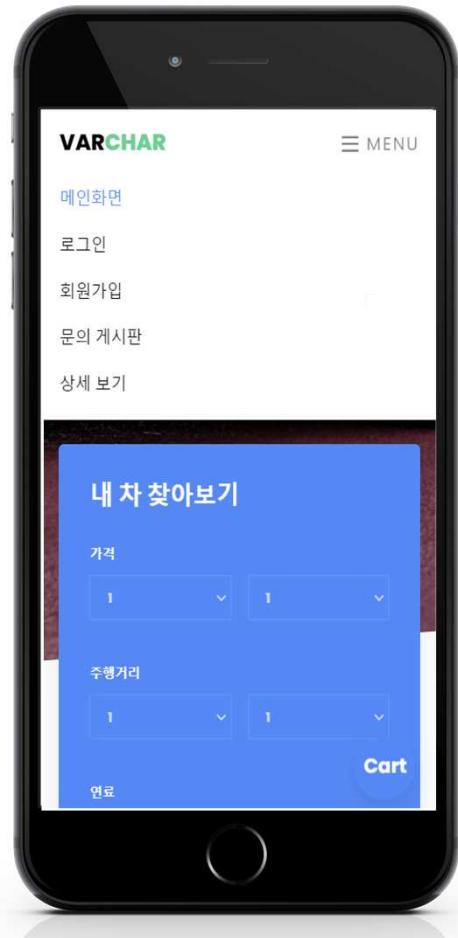


DESIGN Description

- 1 검색 버튼
- 2 문의 글 더 보기 버튼
- 3 VARCHAR 로고 버튼
- 4 지점 위치 조회 버튼
- 5

DEVELOP Description

- 1 제목으로 문의 글 조회
- 2 4개 초과 시 더 보기 버튼 활성화
- 3 해당 지점 상세 주소 출력
- 4 해당 지점의 위치로 마커 이동
- 5





- ERD
- User Flow
- Logic Process

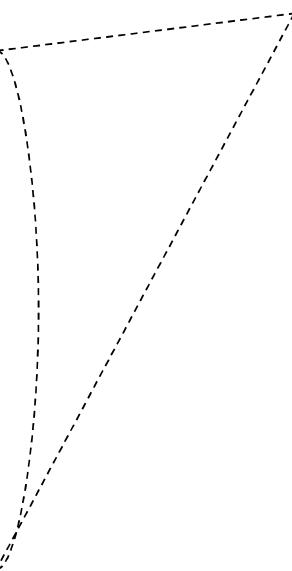


CMEMBER

아이디	MID	VARCHAR(20)	PK
비밀번호	MPW	VARCHAR(20)	NOT NULL
사용자 이름	MNAME	VARCHAR(10)	NOT NULL
사용자 별명	MNICKNAME	VARCHAR(20)	NOT NULL
사용자 주소	MADD	VARCHAR(200)	NOT NULL
휴대폰 번호	MPHONE	VARCHAR(20)	NOT NULL
사용자 이메일	MEMAIL	VARCHAR(100)	NOT NULL
사용자 권한	MROLE	VARCHAR(20)	NOT NULL

CREPLY

답글 고유번호	RID	INT	PK
사용자 아이디	MID	VARCHAR(10)	NOT NULL
글 고유번호	BNUM	INT	NOT NULL
답글 내용	CMSG	VARCHAR(500)	NOT NULL
제약조건	CONSTRAINT CBOARD_CREPLY FOREIGN KEY(BNUM) REFERENCES CBOARD(BNUM) ON DELETE CASCADE		



CBOARD

글 고유번호	BNUM	INT	PK
사용자 아이디	MID	VARCHAR(20)	NOT NULL
사용자 별명	MNICKNAME	VARCHAR(20)	NOT NULL
글 제목	BTITLE	VARCHAR(50)	NOT NULL
글 내용	BCONTENT	VARCHAR(500)	NOT NULL
글 개수	BCNT	INT	DEFAULT 0
글 날짜	BDATE	VARCHAR(20)	NOT NULL

CAR

차량고유번호	CNUM	INT	PK
차량 품명	CTITLE	VARCHAR(300)	NOT NULL
차량 정보	CSUBTITLE	VARCHAR(300)	NOT NULL
차량 연식	CYEAR	INT	NOT NULL
차량 연료	CFUEL	CARCHAR(20)	NOT NULL
주행 거리	CKM	INT	NOT NULL
차량 가격	CPRICE	INT	NOT NULL
파는 지역	CCITY	VARCHAR(20)	NOT NULL
차량 사진	CIMG	VARCHAR(500)	NOT NULL

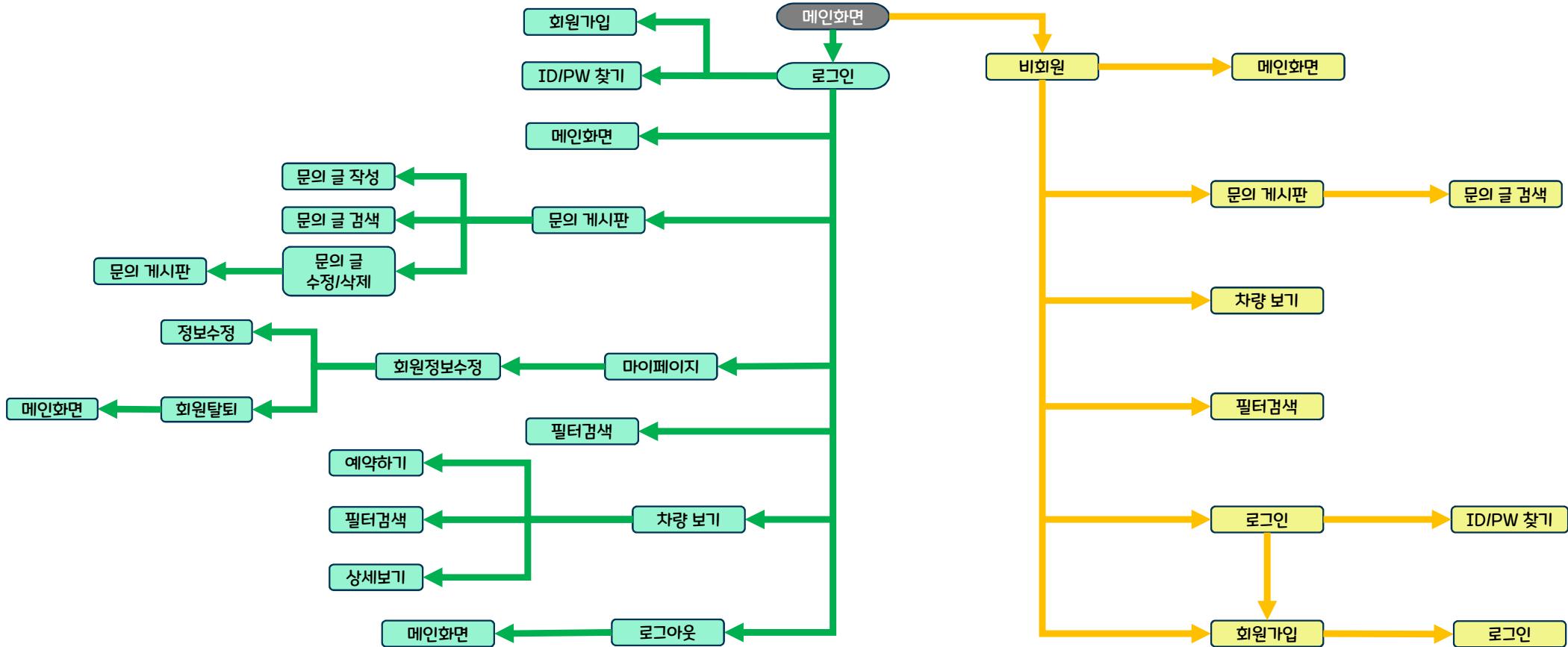


User Flow

VARCHAR

VARCHAR

20 page



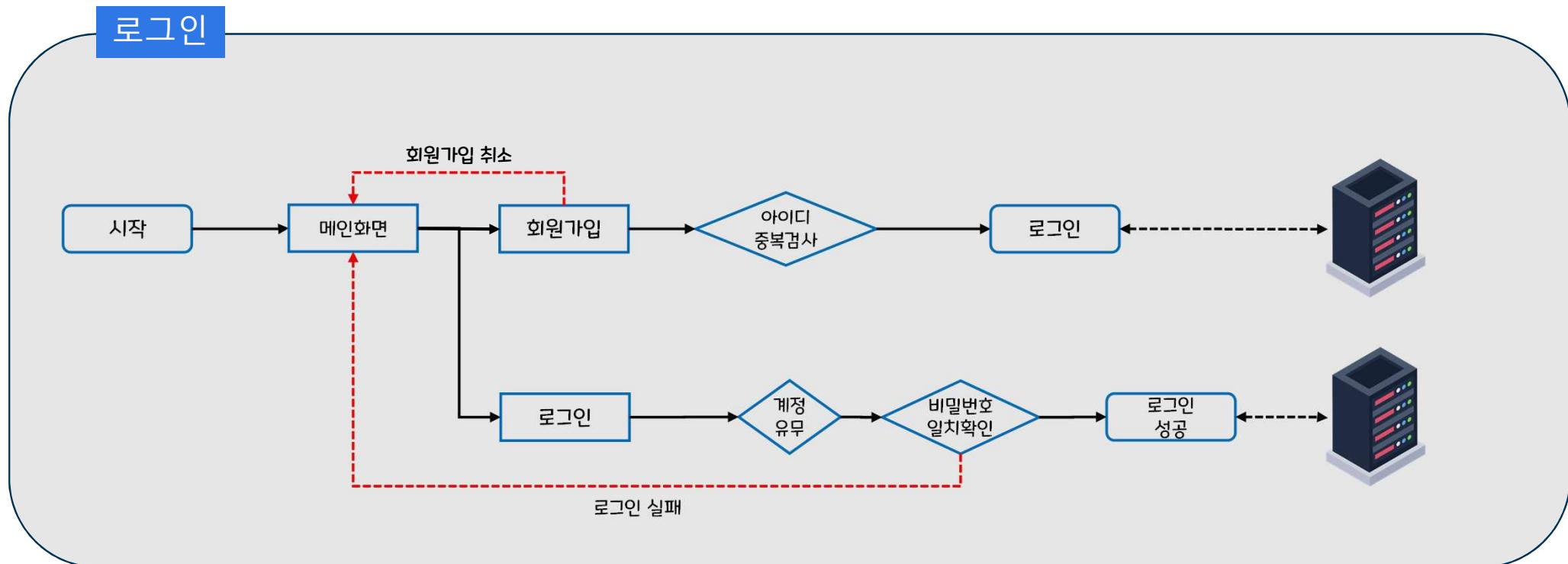


Logic Process[0] - Login

VARCHAR

VARCHAR

21 page



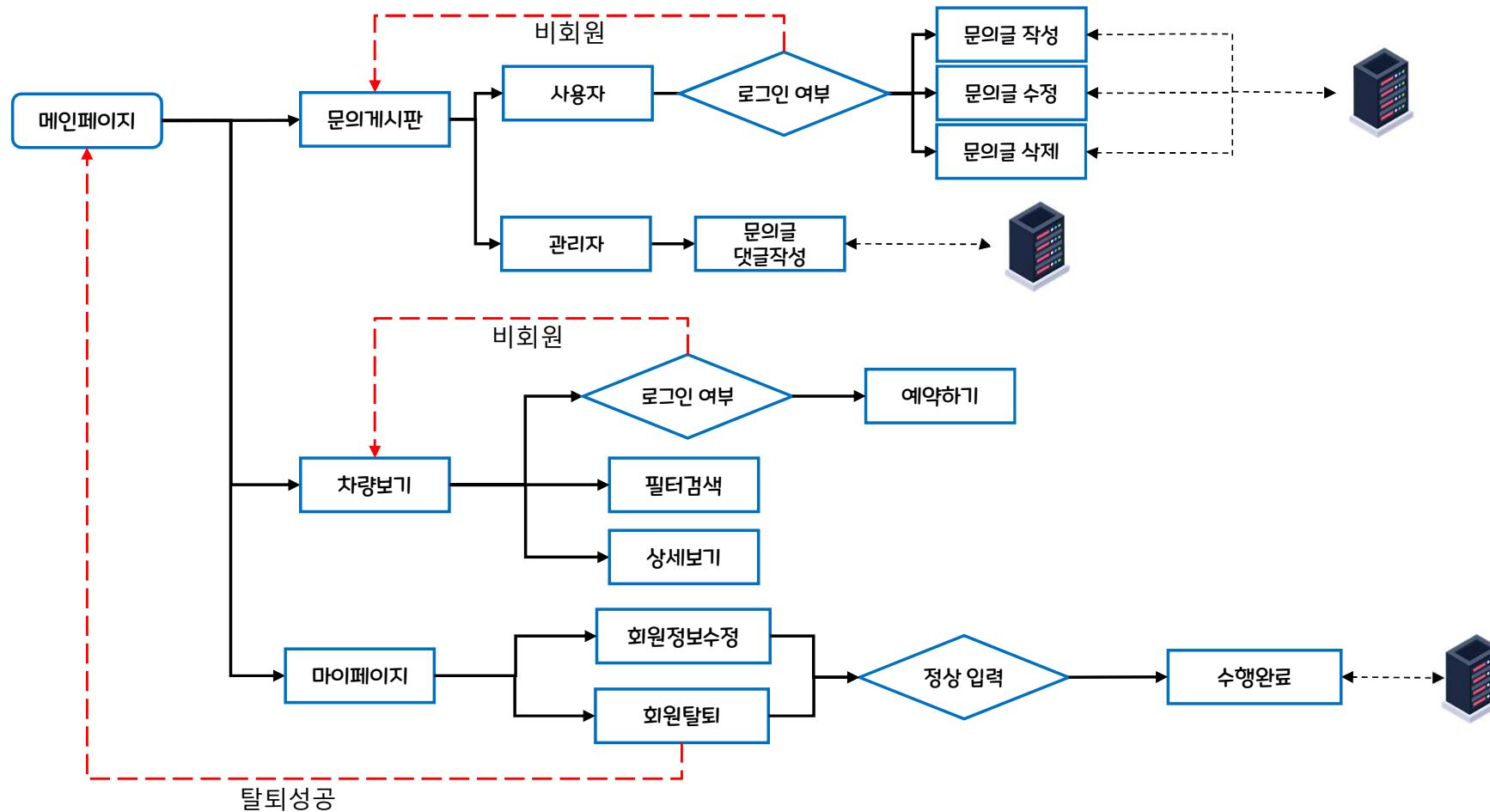


Logic Process[1]

VARCHAR

VARCHAR

22 page





- 주요기능(코드설명)
- API, Plugin



주요기능 – 필터검색[0]

VARCHAR

VARCHAR

24 page

필터 검색 : SearchVO.java

```
//input태그의 checkbox를 사용하여 중복 선택을 할 수 있으므로 문자열을 배열형태로 저장
private ArrayList<String> fuelList = new ArrayList<>();
//input태그의 checkbox를 사용하여 중복 선택을 할 수 있으므로 문자열을 배열형태로 저장
private ArrayList<String> cityList = new ArrayList<>();
// 차량 검색페이지로 처음 이동 시 차량 가격의 초기값을 설정할 변수 (기본생성자에 초기값 설정)
private int price_min;
private int price_max;
// 차량 검색페이지로 처음 이동 시 주행거리의 초기값을 설정할 변수 (기본생성자에 초기값 설정)
private int km_min;
private int km_max;
// 차량 검색페이지로 처음 이동 시 연식의 초기값을 설정할 변수 (기본생성자에 초기값 설정)
private int year_min;
private int year_max;
// 더 보기 초기값 설정할 변수
private int range1;
private int range2;
//정렬 데이터가 저장될 변수
private String checksort;
```

※ 사용자가 검색한 값들을 담아줄 변수 선언

- Checkbox속성은 단일, 다중선택이 가능함으로 배열 형태로 저장
- 최소/최대 값과 더 보기 관련 변수는 기본 생성자로 초기값을 설정

← ↻ ↺ 주요기능 – 필터검색[1]

VARCHAR

VARCHAR

25 page



가격검색 : CarSearchAction.java

```
// 가격  
//검색할 때 입력한 값이 null이 아닌 경우 = 범위를 위해 값을 입력하고 검색한 경우  
// => 첫 차량검색페이지를 실행한 경우 파라미터 값이 null임  
if(request.getParameter("pmin") != null && request.getParameter("pmax") != null) {  
    int price_min = Integer.parseInt(request.getParameter("pmin")); //가격의 최소 값  
    int price_max = Integer.parseInt(request.getParameter("pmax")); //가격의 최대 값  
    svo.setPrice_min(price_min);  
    svo.setPrice_max(price_max);  
    request.setAttribute("pmin", price_min);  
    request.setAttribute("pmax", price_max);  
    // System.out.println("로그-가격 : "+price_min+"~"+price_max);  
}  
// System.out.println("로그-가격확인 : "+svo.getPrice_min()+'/'+svo.getPrice_max());
```

판매지역검색 : CarSearchAction.java

```
//지역  
String city[] = request.getParameterValues("city"); //지역 다중 선택 시  
  
if(city != null){  
    for(int i=0;i<city.length;i++) { //선택된 필터 개수에 따라 반복처리하여 검색  
        cList.add(city[i]);  
        System.out.println("받아온 지역 파라미터 값 : " + city[i]);  
    }  
    svo.setCityList(cList); //SearchVO의 배열객체에 필터선택된 지역배열값 저장  
}  
request.setAttribute("cList", cList);  
System.out.println("로그 CarSearchAction fList : " + fList);  
System.out.println("로그 CarSearchAction cList : " + cList);  
  
ArrayList<CarVO> datas = sdao.selectAll(svo);
```

※ 전체 차량 출력은 사용자의 설정 값을 받지 않음

- 사용자의 값을 받지 않으면 0이 아닌 null값이 설정됨
- 조건문을 사용해 초기설정 값을 받도록 유도

※ 최소/최대 값과 같이 범위로 값이 설정되는 것이 아닌 선택 값은 선택하지 않을 시 해당 선택항목의 전체 범위로 설정

- 다중선택이 가능함으로 배열에 저장
- 사용자가 다중 선택을 했다면 받아온 배열의 길이 만큼 배열리스트에 저장
- 로그 처리로 데이터 값 확인하며 코드 작성

← ↴ ↵ 주요기능 – 필터검색[2]

VARCHAR

VARCHAR

26 page

삽입할 부분SQL문 : SearchDAO.java

```
String CfuelSql = "";  
String CcitySql = "";  
String CyearSql = "";  
String CkmSql = "";  
String CpriceSql = "";  
String Check="";  
String sql_selectAll = "";
```

※ 하나의 메인SQL문과 삽입해줄 서브SQL문 초기화

- 조건문의 의해 문자열이 재할당되는 서브SQL문
- 서브SQL문이 모여 완성될 메인SQL문

검색한 판매지역 : SearchDAO.java

```
if(svo.getCityList().size() > 0){ //지역의 필터 값을 저장한 배열객체의 길이가 1 이상일 때  
    StringBuilder ccitySb = new StringBuilder();  
    ArrayList<String> citydata = svo.getCityList();  
  
    for(int i = 0; i < svo.getCityList().size() ; i++){  
        ccitySb.append("\\" + citydata.get(i) + "\\");  
        if(i+1 < svo.getCityList().size())  
            ccitySb.append(",");  
    }  
    CcitySql = "AND CCITY IN ("+ccitySb.toString()+")";  
}
```

※ 서브SQL문 만들어지는 로직

- 문자열을 합쳐야 함으로 StringBuilder 객체 생성
- append함수로 문자열 조합
- 서브SQL문에 조합된 문자열 재할당

“AND CFUEL IN ('서울', '경기')”

← ↻ ⓘ 주요기능 – 필터검색[3]

VARCHAR

VARCHAR

27 page

★★★ 검색조건에 맞게 데이터를 조회해줄 메인SQL문 ★★★ : SearchDAO.java

```
sql_selectAll = "SELECT * FROM (SELECT B.*, ROWNUM AS R FROM (SELECT A.* FROM (SELECT * FROM CAR WHERE 1=1 "
    + CfuelSql + " " + CcitySql + " " + CyearSql + " " + CkmSql + " " + CpriceSql + ") A "+Check+") B) WHERE R BETWEEN "+svo.getRange1()+" AND "+svo.getRange2();
```

“SELECT A.* FROM (SELECT * FROM CAR WHERE 1=1 AND CCITY IN(‘서울’,‘경기’) AND CYEAR BETWEEN
2000 AND 2023 AND CKM BETWEEN 1 AND 100 AND CPRICE BETWEEN1 AND 100) A”;

정렬 선택 로직 : SearchDAO.java

```
if(svo.getChecksort() != null) { //값이 참이라면
    String element = "";
    if(svo.getChecksort().equals("최신순")) {
        element = "ROWNUM";
    }
    if(svo.getChecksort().equals("제목순정렬")) {
        element = "CTITLE";
    }
    if(svo.getChecksort().equals("가격순정렬")) {
        element = "CPRICE";
    }
    if(svo.getChecksort().equals("주행거리순")) {
        element = "CKM";
    }
    System.out.println("로그 element값 : "+element);
    Check = "ORDER BY "+element+" ASC";
}
```

※ 서브SQL문들이 조합되는 메인SQL문

• 첫 번째 괄호

- WHERE 1=1 은 참을 의미하는 무의미한 쿼리
- 조건문에 의해 조합된 SQL문에 의한 차량 목록을 출력

• 두 번째 괄호

- 첫 번째 괄호에 의해 출력될 데이터들을 정렬할 SQL문 설정

• 세 번째 괄호

- ROWNUM에 의해 웹 페이지에서 출력될 데이터 개수 지정



주요기능 – 필터검색[4].검색조건 유지[0].checkbox,option태그

VARCHAR

VARCHAR

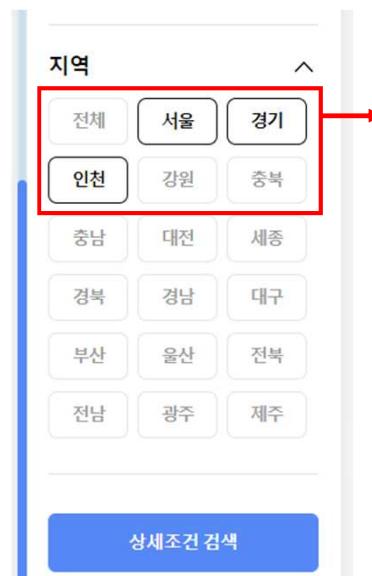
28 page



검색조건 유지 : filterSearch.jsp

```
<li><span> <input type="checkbox" value="전체"
    name="city">
    <c:if test="${cList.contains('전체')}">checked='checked'</c:if> />
    <label class="region">전체</label>
</span></li>
<li><span> <input type="checkbox" value="서울"
    name="city">
    <c:if test="${cList.contains('서울')}">checked='checked'</c:if> />
    <label class="region">서울</label>
</span></li>
<li><span> <input type="checkbox" value="경기"
    name="city">
    <c:if test="${cList.contains('경기')}">checked='checked'</c:if> />
    <label class="region">경기</label>
</span></li>

<option <c:if test="${ymin == 2018}">selected='selected'</c:if>>2018</option>
<option <c:if test="${ymin == 2019}">selected='selected'</c:if>>2019</option>
<option <c:if test="${ymin == 2020}">selected='selected'</c:if>>2020</option>
<option <c:if test="${ymin == 2021}">selected='selected'</c:if>>2021</option>
<option <c:if test="${ymin == 2022}">selected='selected'</c:if>>2022</option>
```



※ 실제 VARCHAR 사이트 필터검색 모달 창

- 검색조건이 유지된 캡쳐 화면

※ 사용자의 편의성을 위해 필터검색 후 검색조건 값 유지

- JSTL core태그를 선언해 <c:if>태그 사용

범위 슬라이더 검색 조건 유지 : CarSearchAction.java

```
//주행거리
// 8 request.getParameter("min-value") != null & request.getParameter("max-value") != null {
int km_min= integer.parseInt(request.getParameter("min-value")); //주행거리 최소 값
int km_max= integer.parseInt(request.getParameter("max-value")); //주행거리 최대 값
svo.setKm_min(km_min);
svo.setKm_max(km_max);
request.setAttribute("kmin", km_min);
request.setAttribute("kmax", km_max);
System.out.println("주행거리 : "+km_min+"~"+km_max); <script id="Slider" kmin='${kmin}' kmax='${kmax}' src="js/rangeSlider.js"></script>
// System.out.println("형변환 후 : "+km_max);
}
```

사용자로부터 받은 값이 웹에서 보여질 요소의 값에 재할당 되는 과정

범위 슬라이더 검색 조건 유지 : filterSearch.jsp

```
<div>
    <div class="col-sm-12">
        <div id="slider-range"></div>
    </div>
    <div class="slider-labels">
        <div>
            <strong>최소 : </strong> <span id="slider-range-value1"></span>&ampnbsp&ampnbspkm
        </div>
        <div>
            <strong>최대 : </strong> <span id="slider-range-value2"></span>&ampnbsp&ampnbspkm
        </div>
        <div class="col-sm-12">
            <input type="hidden" name="min-value" value="" /> <input
                type="hidden" name="max-value" value="" />
        </div>
    </div>
</div>
```

범위 슬라이더 검색 조건 유지 : rangeSlider.js

```
// Initialize slider:
$(document).ready(function () {
    var filterSlider = document.getElementById("Slider");
    // 스크립트태그로 기간 조건 바기
    let kmin = filterSlider.getAttribute("kmin");
    let kmax = filterSlider.getAttribute("kmax");
    //null값이 아니라 false로 해야한 --> is로 가져올 때 데이터가 없으면 null이 아니라 false
    if(kmin==false & kmax==false){ //처음 페이지 진입 시 초기 값 설정
        kmin=1000;
        kmax=700000;
        console.log("rangeSlider.js에서 kmin,kmax 값 reset");
    }
    console.log(kmin);
    console.log(kmax);
    $( ".noUi-handle" ).on("click", function () {
        $(this).width(50);
    });
    var rangeSlider = document.getElementById("slider-range");
    var moneyFormat = wNumb({
        decimals: 0,
        //thousand: ",",
        prefix: ""
        // prefix: "$"
    });
    noUiSlider.create(rangeSlider, {
        start: [kmin, kmax],
        step: 1000,
        range: {
            min: [1000],
            max: [700000],
        },
        format: moneyFormat,
        connect: true,
    });

    // Set visual min and max values and also update value from inputs
    rangeSlider.noUiSlider.on("update", function (values, handle) {
        document.getElementById("slider-range-value1").innerHTML = values[0];
        document.getElementById("slider-range-value2").innerHTML = values[1];
        document.querySelector("input[name='min-value']").value = moneyFormat.from(values[0]);
        document.querySelector("input[name='max-value']").value = moneyFormat.from(values[1]);
    });
});
```

1

6

11

13

2

3

4

5



기능 – 리스너 크롤링 데이터 가공

VARCHAR

VARCHAR

30 page



동적 크롤링 작업 로직 : CrawlingListener.java

```
public void contextInitialized(ServletContextEvent sce) {
    ServletContext sc = sce.getServletContext();
    CarDAO cDAO = new CarDAO();
    System.out.println("TestListener : contextInitialized()에서 실행 중 : 톰캣 시작이 감지됨");
    System.out.println("크롤링 시작");
    if (!cDAO.hasSample(null)) { //DB에 데이터가 없다면
        Crawling.data(); //크롤링 메서드 실행
    }
    System.out.println("크롤링 작업 완료");
}
```

※ 리스너로 샘플 데이터의 존재 유무 확인

- 리스너는 최우선적으로 작업을 수행

- 타겟 사이트에서 가지고 올 샘플 데이터 선택
- 샘플 데이터를 가공해서 원하는 데이터 저장

크롤링 데이터 선택 : Crawling.java

```
String ctitle1 = "p.tit.ellipsis > a"; // 차량명
String csubtitle1 = "p.stxt.ellipsis > a"; // 특징
String cyear1 = "div.mode-cell.year > span.text"; // 연식
String cfuel1 = "div.mode-cell.fuel > span.text"; //엔진
String ckm1 = "div.mode-cell.km > span.text"; //주행거리
String cprice1 = "div.mode-cell.price"; //가격
String ccity1 = "ul.content-list > li:nth-child(1) > span.text"; // 지역
String cimg1 = "div.list-inner > div.mode-cell.thumb > a.img.w132 > img"; // 이미지
```

크롤링 데이터 가공 : Crawling.java

```
while (ctitle3.hasNext()) {
    //자동차 명
    String ctitle4 = ctitle3.next().text();
    System.out.println("차 : " + ctitle4);
    //자동차특징
    String csubtitle4 = csubtitle3.next().text();
    System.out.println("설명 : " + csubtitle4);

    //연식 : 데이터 가공
    String cyear4 = cyear3.next().text();
    cyear4 = cyear4.substring(cyear4.length()-5, cyear4.length()-3); //연식 xx/xx 중 '/' 앞부분의 2자리수만 추출
    cyear4 = "20".concat(cyear4); //20xx 년도로 저장하기 위해 concat으로 문자열 합침
    System.out.println("연식 : " + Integer.parseInt(cyear4));

    //엔진 : 데이터 가공
    String cfuel4 = cfuel3.next().text();
    if(cfuel4.indexOf("+") > 0) { // 문자열 '+'의 인덱스가 0보다 클 경우
        cfuel4 = cfuel4.substring(0,cfuel4.indexOf("+")); //문자열 처음부터 '+' 전까지 잘라 저장
    }
    System.out.println("엔진 : " + cfuel4);

    //주행거리 : 데이터 가공
    String ckm4 = ckm3.next().text();
    if(ckm4.equals("미등록")) {
        ckm4 = "0";
    }
}
```



기능 – 인코딩 필터

VARCHAR

VARCHAR

31 page

xml 설정 값 : web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app>
    <context-param>
        <param-name>encoding</param-name>
        <param-value>UTF-8</param-value>
    </context-param>
```

※ 인코딩 설정을 모든 페이지에 함으로 반복적인 코드를 줄이기 용이

- Xml파일에서 UTF-8로 인코딩 값 설정

EncFilter.java

```
@WebFilter({"*.do", "*.jsp"})
public class EncFilter extends HttpFilter implements Filter {

    private String encoding;

    public EncFilter() {
        super();
    }

    public void destroy() {
    }

    public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain) throws IOException, ServletException {
        request.setCharacterEncoding(this.encoding);
        response.setCharacterEncoding(this.encoding);
        //System.out.println("=====페이지 이동===== ");
        chain.doFilter(request, response);
    }

    public void init(FilterConfig fConfig) throws ServletException {
        this.encoding=fConfig.getServletContext().getInitParameter("encoding");
        System.out.println("필터 생성");
    }
}
```

- 모든 페이지 인코딩 설정 적용

- FilterConfig 객체로 web.xml에서 설정 값 전달받음



API – E-mail API[0].아이디 찾기[0]

VARCHAR

VARCHAR

32 page



E-mail 발송 ajax : findAccount.jsp

```
//아이디 전송
document.getElementById("ce아이디발송버튼Id").onclick = () => {
    const name = $("#입력한이름Id").val();
    const email = $("#입력한이메일Id").val();
    console.log("로그 : name " + name);
    console.log("로그 : email " + email );
    $.ajax({
        type:'POST',
        url: '${pageContext.request.contextPath}/sendIdEmail',
        data: {name:name ,email:email},
        success: function(mid){
            console.log("로그 mid : ["+mid+"]");
            if(mid != null){
                console.log("로그 mid1 : ["+mid+"]");
                $('#checkResult').text("아이디 전송 완료");
                $('#checkResult').css("color", "blue");
            }else{
                console.log("로그 mid2 : ["+mid+"]");
                $('#checkResult').text("아이디 전송 불가");
                $('#checkResult').css("color", "red");
            }
        },
        error: function(request, status, error){ // 서블릿에서 에러 발생 시
            console.log("code: "+request.status);
            console.log("message: "+request.responseText);
            console.log("error: "+error);
        }
    });
}
```

※ 버튼을 누르게 되면 입력한 이름, 이메일을
POST방식으로 Controller로 전달

- 입력한 이메일로 아이디 관련 정보 메일 발송

The screenshot shows a mail client interface with two received emails. The top email is from '보낸사람' (VIP) <VARCHAR> [REDACTED] and the recipient is '받는사람' [REDACTED]. The subject is '<VARCHAR> 아이디 찾기 안내'. The message body says: '안녕하세요 [VARCHAR] 입니다. [REDACTED] 님의 아이디는abc입니다.'

The bottom email is also from '보낸사람' <VARCHAR> [REDACTED] and the recipient is '받는사람' [REDACTED]. The subject is '<VARCHAR> 아이디 찾기 안내'. It has a reply arrow pointing to the first email.

※ 로깅처리로 전달하는 값 확인



API – E-mail API[0].아이디 찾기[1]

VARCHAR

VARCHAR

33 page



Controller : SendIdEmail.java

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) {
    emailVO evo = new emailVO();
    System.out.println("emailVO 통과");
    emailIdDAO edao = new emailIdDAO();
    System.out.println("emailDAO 통과");

    evo.setEmail(request.getParameter("email")); // 사용자의 email을 VO에 저장
    MemberVO mvo = new MemberVO();
    mvo.setMname(request.getParameter("name"));
    mvo.setMemail(request.getParameter("email"));
    System.out.println(mvo);
    String mid = edao.sendIdMail(mvo); 1
    // 요청했던 곳으로 아이디를 보낼 예정
    response.setContentType("text/html; charset=UTF-8");
    response.getWriter().write(mid+"");
}
```

5

아이디 값 반환 : emailIdDAO.java

```
// 메일 발송
Transport.send(message);

} catch (Exception e) {
    e.printStackTrace();
}
return mid; // 사용자 아이디 4
```

4

사용자 아이디 조회 후 저장 : emailIdDAO.java

```
public class emailIdDAO {
    MemberDAO mDAO = new MemberDAO();
    // public static void main(String args[]) { sendMail(); }

    public String sendIdMail(MemberVO mvo) {
        // 아이디가 차이나서 set
        mvo.setMid(mDAO.findId(mvo)), 2
        // 메일 인코딩
        final String bodyEncoding = "UTF-8"; // 컨텐츠 인코딩

        String subject = "<VARCHAR> 아이디 찾기 안내";
        String fromEmail = "dream82sy@naver.com";
        String fromUsername = "<VARCHAR>";
        String toEmail = mvo.getMemail(); // 콤마(,)로 여러개 나열
    }
}
```

2

아이디 저장 : emailIdDAO.java

```
String mid = mvo.getMid(); 3
try {
    // 메일 서버 인증 계정 설정
    Authenticator auth = new Authenticator() {
        protected PasswordAuthentication getPasswordAuthentication() {
            return new PasswordAuthentication(username, password);
        }
    }
}
```

※ 사용자가 입력한 데이터들을 기반으로 아이디를 찾아 전달하는 과정



API – E-mail API[0].아이디 찾기[2]

VARCHAR

VARCHAR

34 page



성공/실패 조건문 로직 : findAccount.jsp

```
//아이디 전송
document.getElementById("certificateBtnForId").onclick = () => {
    const name = $("#nameInput").val();
    const email = $("#emailInputForId").val();
    console.log("로그 : name " + name);
    console.log("로그 : email " + email );
    $.ajax({
        type:'POST',
        url: '${pageContext.request.contextPath}/sendIdEmail',
        data: {name:name ,email:email},
        success: function(mid){
            console.log("로그 mid : ["+mid+"]");
            if(mid != null){
                console.log("로그 mid1 : ["+mid+"]");
                $('#checkResult').text("아이디 전송 완료");
                $('#checkResult').css("color", "blue");
            }else{
                console.log("로그 mid2 : ["+mid+"]");
                $('#checkResult').text("아이디 전송 불가");
                $('#checkResult').css("color", "red");
            }
        },
        error: function(request, status, error){ // 서블릿에서 에러 발생 시
            console.log("code: "+request.status);
            console.log("message: "+request.responseText);
            console.log("error: "+error);
        }
    });
}
```

※ 조건문을 통해 사용자에게 실시간으로 성공/실패 멘트 출력

The screenshot shows a user interface for account recovery. It has two input fields: '아이디 찾기' (ID Search) and '비밀번호 찾기' (Password Recovery). Below these are fields for '이름' (Name) and '이메일' (Email), with the email field containing '[REDACTED]@naver.com'. A large blue button labeled '아이디 발송' (Send ID) is positioned to the right of the email field. At the bottom is a green button labeled '로그인' (Login).

아이디 찾기

비밀번호 찾기

이름

[REDACTED]

이메일

[REDACTED]@naver.com

아이디 발송

아이디 전송 완료

로그인

API – E-mail API[1].비밀번호 찾기[0]

랜덤비밀번호 생성 로직 : emailPwDAO.java

```
public String sendPwMail(MemberVO mvo) {
    // 랜덤으로 비밀번호 재발행
    StringBuffer temp = new StringBuffer();
    Random rnd = new Random();
    for (int i = 0; i < 10; i++) {
        int rIndex = rnd.nextInt(3);
        switch (rIndex) {
            case 0:
                // a-z
                temp.append((char) ((int) (rnd.nextInt(26)) + 97));
                break;
            case 1:
                // A-Z
                temp.append((char) ((int) (rnd.nextInt(26)) + 65));
                break;
            case 2:
                // 0-9
                temp.append((rnd.nextInt(10)));
                break;
        }
    }
    String randomPw = temp.toString();
    System.out.println("랜덤 비밀번호emailPwDAO : " + randomPw);
}
```

※ 사용자 비밀번호에 랜덤비밀번호 재할당

- Char타입 문자와 int타입 정수를 섞어 랜덤비밀번호 생성 후 메일 발송



※ StringBuffer 객체가 가지고 있는 데이터들을 문자열로 조합한 후 변수에 저장

랜덤비밀번호를 사용자비밀번호에 재할당 로직 : emailPwDAO.java

```
// 메일에 출력할 텍스트
StringBuffer sb = new StringBuffer();
sb.append("안녕하세요 [VARCHAR] 입니다.\n");
if(mDAO.findPw(mvo) != null) { // 등록된 회원의 때
    mvo.setMpw(randomPw); // 랜덤 비밀번호를 mvo의 mpw에 set해줌
    mDAO.updatePw(mvo); // db에 저장된 pw를 랜덤 비밀번호로 변경
    sb.append("새로행된 비밀번호는 " + randomPw + "입니다.");
} else {
    sb.append("죄송합니다. 입력하신 정보는 미가입된 계정입니다. 회원가입 부탁 드립니다.");
}
String html = sb.toString();
```

※ 사용자의 현재 비밀번호에
UPDATE SQL문을 통해 재할당

랜덤비밀번호 값 반환 : emailPwDAO.java

```
MailcapCmdMap.addMailcap("message/rfc822; x-java-content-handler=CommandMap");
CommandMap.setDefaultCommandMap(MailcapCmdMap);

// 메일 발송
Transport.send( message );

} catch ( Exception e ) {
    e.printStackTrace();
}
return randomPw; //재발행된 비밀번호
```

※ 랜덤비밀번호를 저장한 값을 반환

휴대폰번호 인증 : signup.jsp

```

function startTimer(count, display) {
    // ...
    // 문자 API - 해당 파일 160번 라인
    const phone = $("#phoneinput").val(); // 입력한 번호 값
    console.log("로그 : phone " + phone);

    $.ajax({
        type: 'POST',
        url: '${pageContext.request.contextPath}/sendMSG',
        data: {phone:phone}, // 원쪽은 변수명 같은 것 오른쪽 phone이 434번 라인 phone

        success: function(randomNumber){
            console.log("로그: ["+randomNumber+"]")
            if(randomNumber != null){
                $('#checkResult').text("인증번호 전송 완료");
                $('#checkResult').css("color", "blue");

                number = randomNumber;
            } else{
                console.log("로그: ["+randomNumber+"]")
                $('#checkResult').text("인증번호 전송 불가");
                $('#checkResult').css("color", "red");
            }
        },
        error: function(request, status, error){ // 서블릿에서 에러 발생 시
            console.log("code: "+request.status);
            console.log("message: "+request.responseText);
            console.log("error: "+error);
        }
    });
}

// 타이머 시작(문자가 보내지고 타이머가 시작되어 하기 때문에 문자발송 코드 이후에 불안)
console.log("로그 : 타이머 시작");
isRunning = true;
let minutes, seconds;
timer = setInterval(function () {
    minutes = parseInt(count / 60, 10);
    seconds = parseInt(count % 60, 10);
    minutes = minutes < 10 ? "0" + minutes : minutes;
    seconds = seconds < 10 ? "0" + seconds : seconds;
    display.textContent = "(" + minutes + ":" + seconds + ")";
}, 1000);

```

```

// 타이머 끝
if (--count < 0) {
    clearInterval(timer);
    display.textContent = "";
    isRunning = false;
    number = 0; // 인증번호 초기화
    certificateCheckBtn.disabled = true;
    certificateCheckBtn.style.backgroundColor = "rgb(222, 229, 236)";
    certificateCheckBtn.style.cursor = "no-drop";
    // console.log("로그 : 종료 " + isRunning);
}
}, 1000);

```

인증번호 타이머 : signup.jsp

```

// 버튼 눌르게 되면 타이머할수 설정
certificateBtn.onclick = sendAuthNum;

certificateCheckBtn.onclick = function sendCheck(){
    const checkNum = $("#certificateInput").val(); // 172라인

    $.ajax({
        type: 'POST',
        url: '${pageContext.request.contextPath}/sendCheck',
        data : {randomNumber:number, checkNum:checkNum},
        success: function(result){
            console.log("로그: ["+result+"]");
            if(result == 1){
                $('#checkResult').text("본인 인증 성공!");
                $('#checkResult').css("color", "#01d28e");

                $('#timer').remove(); //타이머 제거
                // 성공하면 인증번호 별송 / 확인 버튼 비활성화
                certificateCheckBtn.disabled = true;
                document.getElementById("phoneboxright").style.backgroundColor = "rgb(222, 229, 236)";
                document.getElementById("phoneboxright").disabled = true;
                document.getElementById("phoneboxright").style.cursor = "no-drop";
            } else{
                $('#checkResult').text("인증번호 불일치!");
                $('#checkResult').css("color", "#e05446");
                certificateCheckBtn.disabled = false;
            }
        }
    });
}

```

- 인증번호를 랜덤하게 생성하고 사용자에게 문자 전송, 문자가 전송된 후에 타이머 시작
- 사용자가 입력한 값과 위에서 생성한 랜덤비밀번호가 동일한지 비교한 후 int타입 1 또는 0으로 반환하여 조건문 수행 후 멘트 출력

※ 실제 문자가 수신된 캡쳐 화면





지도 출력 로직 : google-map.js

```
var map;
var markerMaxWidth = 300;
function initMap() {
    // The location of Uluru
    var yeoksamplace = { lat: 37.4999269, lng: 127.0365526 };
    // The map, centered at Uluru
    map = new google.maps.Map(document.getElementById("map"), {
        zoom: 18,
        center: yeoksamplace
    });
    // The marker, positioned at Uluru
    var marker = new google.maps.Marker({
        position: yeoksamplace,
        map: map,
        title : "VARCHAR 역삼 본사",
        icon : {
            url : "images/varchargoogle.png",
            labelOrigin : new google.maps.Point(100,55)
        }
    });
    var infowindow = new google.maps.InfoWindow(
        {
            content: '<div>'+ '<h6>VARCHAR 역삼 본사</h6>' +
            '<p>주소: 서울특별시 강남구 역삼동 736-7</p>' +
            '</div>',
            maxWidth: markerMaxWidth
        });
    google.maps.event.addListener(marker, 'click', function() {
        infowindow.open(map, marker);
    });
}
```

VARCHAR 사무실 위치 구글 맵 API 사용 : board.jsp

```
<!-- 구글맵 시작점 -->
<h3 id="mapTitle">WHERE IS VARCHAR ?</h3>
<!-- The div element for the map -->
<div id="map">

<script src="https://maps.googleapis.com/maps/api/js?key=
&callback=initMap&v=weekly" defer></script>
</div>
```

`<div id="storeContainer">`
`<button type = "button" id = "CarStore1" class= "carStore">역삼점</button>`
`<button type = "button" id = "CarStore2" class= "carStore">하남1호점</button>`
`<button type = "button" id = "CarStore3" class= "carStore">하남2호점</button>`
`<button type = "button" id = "CarStore4" class= "carStore">군자점</button>`
`<button type = "button" id = "CarStore5" class= "carStore">거여점</button>`
`<button type = "button" id = "CarStore6" class= "carStore">신림점</button>`
`<button type = "button" id = "CarStore7" class= "carStore">시흥점</button>`
`</div>`
`</div>`
`</div>`
`</section>`

WHERE IS VARCHAR ?



※ VARCHAR 사무실 위치 지도로 표현

- 서울, 경기 곳곳에 있는 VARCHAR 지점 버튼을 누르면 손쉽게 확인 가능
- 구글맵 API 사용

주소 검색 API : signup.jsp

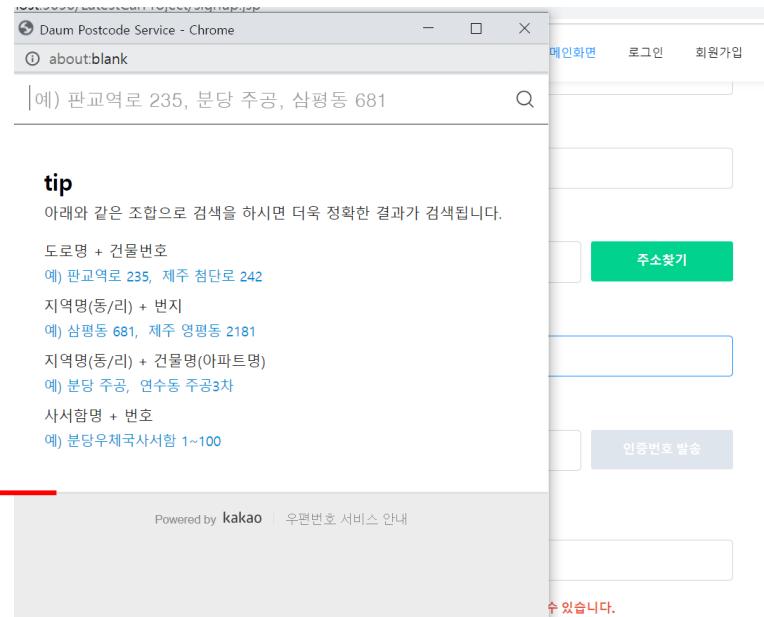
```
<!-- 카카오 주소 API -->
<script src="//t1.daumcdn.net/mapjsapi/bundle/postcode/prod/postcode.v2.js"></script>

// 카카오 주소 API
function kakaoaddress() {
    new daum.Postcode({
        oncomplete: function(data) {
            document.querySelector("#addressinput").value = data.address
        }
    }).open();
}

<div class="inputbox">
    <label class="text-black" for="fname">Address</label>
    <div id="addressbox">
        <input type="text" class="form-control" placeholder="주소"
               id="addressinput" name="madd" autocomplete="off" />
        <button id="searchAddressBtn" type="button" onclick="kakaoaddress()">주소찾기</button>
    </div>
    <div class="warnbox">
        <span>주소를 입력해주세요!</span>
    </div>
</div>
```

※ 회원가입시 주소 검색과 기입 편의성 향상

- 주소 찾기 버튼 클릭 시 주소 팝업 창 활성화
- 카카오 맵 API 사용



※ 실제 VARCHAR사이트 회원가입 시 주소찾기 팝업 창 활성화



Plugin – AccordionPlugin

VARCHAR

VARCHAR

40 page



문의 글 AccordionPlugin 사용 : boardMenu.js

```
_menuInit: function () {
    var _self = this;
    if (this.$selector.data(this.config.dataValue)) return false;
    $.each(this.detect.children, function () {
        var $this = $(this);
        _self.$selector.data(_self.config.dataValue, true);
        $this.hide();
    });
},
_initEvent: function () {
    $(document).on(
        "click.acc.ck",
        this.detect.clickTarget,
        $.proxy(this._activate, this)
    );
    this._hashCheck();
},
_activate: function (e) {
    var $this = $(e.target);
    $this
        .parent("li")
        .toggleClass(this.config.className)
        .siblings()
        .removeClass(this.config.className)
        .children(this.config.descendant.ul + ", " + this.config.descendant.div)
        .slideUp("fast");
    this._effect($this);
},
```

※ 페이지 활용성 향상 위해 아코디언 플러그인 사용

- 문의 글 제목을 누르면 플러그인 동작

VARCHAR 사이트 아코디언 동작 전

VARCHAR에서 파는 중고차

[삭제]

(admin123님, 22/09/07)



VARCHAR 사이트 아코디언 동작 후

VARCHAR에서 파는 중고차

[삭제]

(admin123님, 22/09/07)

최고입니다.

← Plugin – 에러 페이지

VARCHAR

VARCHAR

41 page



xml 설정 값 : web.xml

```
<error-page>
    <error-code>404</error-code>
    <location>/error/error.jsp</location>
</error-page>
```

※ 404에러 발생 시 에러페이지로 이동

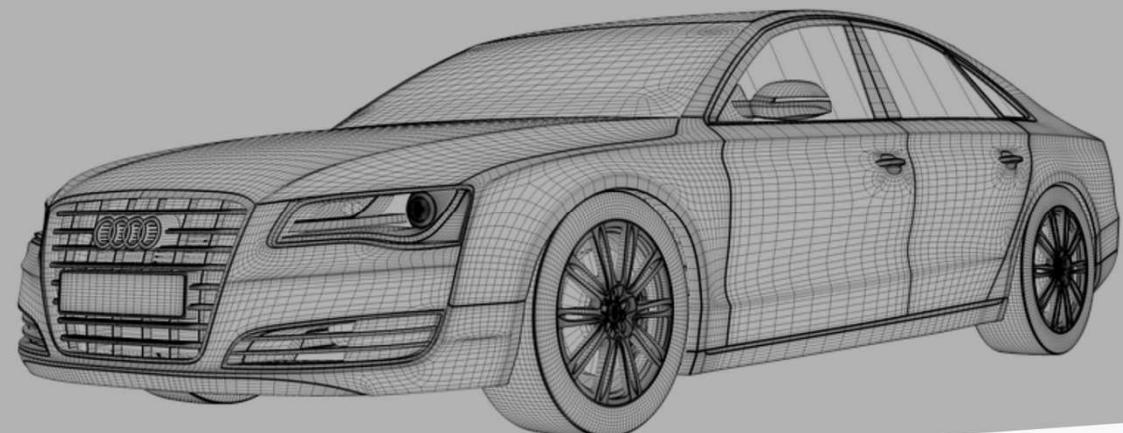
- 에러페이지 캡쳐 화면

VARCHAR

요청하신 페이지를 찾을 수 없습니다...

404Error...

메인화면 이동



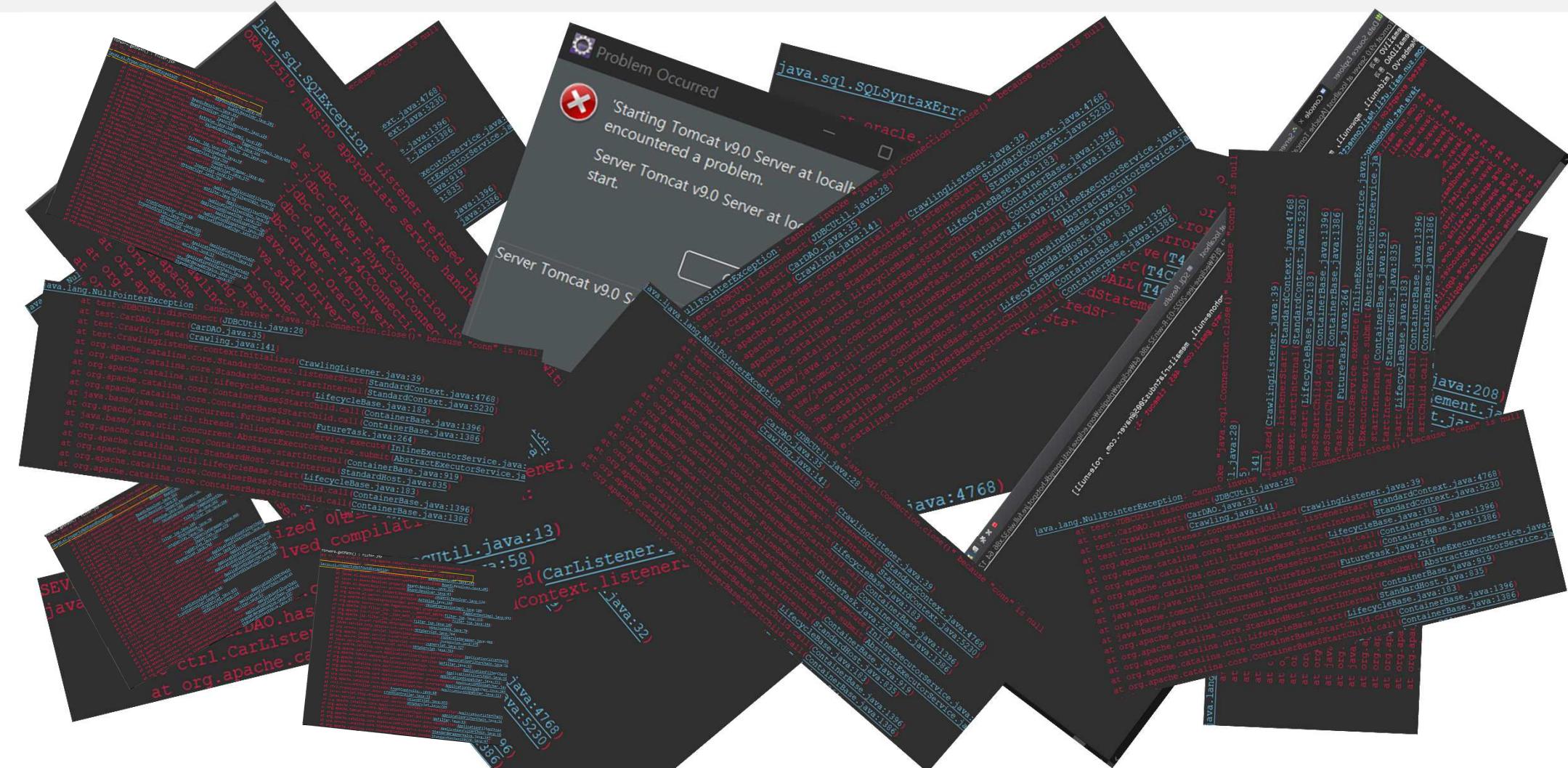


오류 해결

VARCHAR

VARCHAR

42 page



형 변환 오류 : (구)filter.jsp

```

0      name="fuel" value="디젤" />
1  <hr>
2  충남<input type="checkbox" name="city" value="충남" /> 대구<i
3      type="checkbox" name="city" value="대구" /> 서울<input t
4      name="city" value="서울" />
5  <hr>
6  연식<input type="number" name="pmin" value="1"/>~
7  <input type="number" name="pmax" value="5000"/>
8  <input type="submit" value="검색" />
9  <hr />
0  </form>
1

```

filter.do
java.lang.NumberFormatException: Cannot parse null string
at java.base/java.lang.Integer.parseInt(Integer.java:630)
at java.base/java.lang.Integer.parseInt(Integer.java:786)
at ctrl.CarSearchAction.execute(CarSearchAction.java:22)
at ctrl.FrontController.actionDo(FrontController.java:52)
at ctrl.FrontController doGet(FrontController.java:33)
at javax.servlet.http.HttpServlet.service(HttpServlet.java:655)
at javax.servlet.http.HttpServlet.service(HttpServlet.java:764)
at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:225)
at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:215)
at org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)

※ 필터검색 페이지 진입 시 전달 받는
파라미터 값은 null이라 형 변환 오류

CarSearchAction.java

```

//검색할 때 입력한 값이 null이 아닌 경우 = 범위를 위해 값을 입력하고 검색한 경우
if(request.getParameter("pmin") != null && request.getParameter("pmax") != null) {
    price_min = Integer.parseInt(request.getParameter("pmin")); //가격의 최소 값
    price_max = Integer.parseInt(request.getParameter("pmax")); //가격의 최대 값
    svo.setPrice_min(price_min);
    svo.setPrice_max(price_max);
    request.setAttribute("pmin", price_min); //el식으로 사용하기 위해 속성명으로 저장
    request.setAttribute("pmax", price_max);
    System.out.println("가격 : "+price_min+"~"+price_max);
}
System.out.println("가격확인 : "+svo.getPrice_min()+"/"+svo.getPrice_max());

```

- 조건문을 통해 VO에서 초기화해준 값을
받을 수 있게 함



오류 해결 [1] – 아이디 찾기[0]

VARCHAR

VARCHAR

44 page

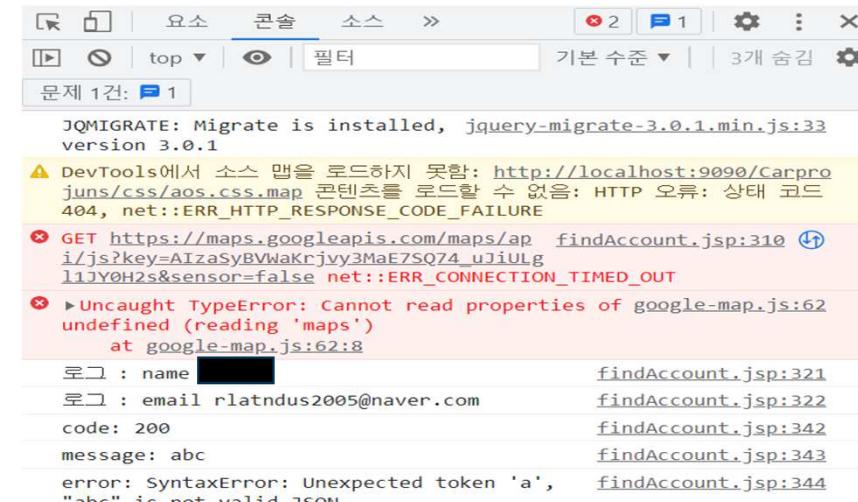


방식 문제 : sendIDEmail.java

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    emailVO evo = new emailVO();
    System.out.println("emailVO 통과");
    emailIdDAO edao = new emailIdDAO();
    System.out.println("emailDAO 통과");

    evo.setEmail(request.getParameter("email")); // 사용자의 email을 빼고 저장
    MemberVO mvo = new MemberVO();
    mvo.setMname(request.getParameter("name"));
    mvo.setMemail(request.getParameter("email"));
    System.out.println(mvo);
    String mid = edao.sendIdMail(mvo);

    // 요청했던 곳으로 아이디를 보낸 예정
    response.setContentType("application/x-json; charset=UTF-8"); // 이부분이 문제
    response.getWriter().write(mid);
}
```



방식 문제 : emailIdDAO.java

```
// MIME 타입 설정
MailcapCommandMap MailcapCmdMap = (MailcapCommandMap) CommandMap.getDefaultCommandMap();
MailcapCmdMap.addMailcap("text/html;; x-java-content-handler"); // 이부분이 문제
MailcapCmdMap.addMailcap("text/xml;; x-java-content-handler");
MailcapCmdMap.addMailcap("text/plain;; x-java-content-handler");
MailcapCmdMap.addMailcap("multipart/*;; x-java-content-handler");
MailcapCmdMap.addMailcap("message/rfc822;; x-java-content-handler");
CommandMap.setDefaultCommandMap(MailcapCmdMap);
```

※ 요청과 응답의 방식이 달라서 발생한 오류

- 데이터를 응답하는 방식이 text/html 방식,
다시 데이터를 응답하는 방식이 x-json 방식



오류 해결 [2] – 아이디 찾기[1]

VARCHAR

VARCHAR

45 page



```
99
100 // MIME 타입 설정
101 MailcapCommandMap MailcapCmdMap = (MailcapCommandMap) CommandMap.getDefaultCommandMap();
102 MailcapCmdMap.addMailcap("text/html; x-java-content-handler=com.sun.mail.handlers.text_html");
103 MailcapCmdMap.addMailcap("text/xml; x-java-content-handler=com.sun.mail.handlers.text_xml");
104 MailcapCmdMap.addMailcap("text/plain; x-java-content-handler=com.sun.mail.handlers.text_plain");
105 MailcapCmdMap.addMailcap("multipart/*; x-java-content-handler=com.sun.mail.handlers.multipart_mixed");
106 MailcapCmdMap.addMailcap("message/rfc822; x-java-content-handler=com.sun.mail.handlers.message_rfc822");
107 CommandMap.setDefaultCommandMap(MailcapCmdMap);
108
109 // 메일 발송
110 Transport.send( message );
111
112 } catch ( Exception e ) {
113     e.printStackTrace();
114 }
115 return mid; //사용자 아이디
116 }
117
118
119
```

```
49 mvo.setNickname(request.getParameter("name"));
50 mvo.setEmail(request.getParameter("email"));
51 System.out.println(mvo);
52 String mid = edao.sendIdMail(mvo);
53
54 //요청했던 곳으로 아이디를 보낼 예정
55 response.setContentType("text/html; charset=UTF-8");//  

56 response.getWriter().write(mid+"");
57 }
58
59
60
61 }
62 }
```

- 주고 받는 방식을 같게 설정하여 오류 해결



시연

VARCHAR

VARCHAR

46 page



시.연.



마무리

VARCHAR

VARCHAR

47 page



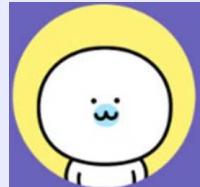
이전에 했던 프로젝트보다 구현하는 기능들과 데이터가 많다 보니 자신이 담당하지 않은 파트에 대해서는 이해하기가 어려웠는데 이를 통해 팀원들 과의 소통의 중요성을 느끼게 되는 계기가 되었습니다.

김수연



프로젝트를 진행하면서 초반 설계 단계가 정말 중요하다는 것을 느꼈습니다. 또한 프로젝트 규모가 전보다 커지게 되면서 역할 분담으로 인한 효율성을 알게 되었습니다.

이향준



View는 프로젝트의 첫인상이고 데이터 전송의 시작과 끝을 담당하기 때문에 다소 책임감과 부담감을 느꼈습니다. 하지만 다른 파트의 조원들이 일찍 내부 설계를 탄탄하게 완성해준 덕분에 무사히 마무리할 수 있었으며, 프로젝트를 통해 고객의 편의성, 유효성 검사, 데이터 전달 등을 총괄적으로 담당하며 프론트 엔드의 매력을 듬뿍 느꼈습니다.

김종현



프로젝트의 규모가 커져서인지 처음에 부담감을 많이 느꼈었는데, 팀원들끼리 서로가 잘 의지하며 풀어 나아갔습니다. 이번 프로젝트도 마찬가지로 어떠한 작업에 있어서 팀원들 과의 의사소통이 중요하다는 걸 많이 깨닫게 되는 계기가 되었습니다.

임환욱



이번 프로젝트를 하면서 스스로한테 부족한점이 많이 보였지만 이런 점을 더 보완한다면 더욱더 완벽 해질수 있을 거란 생각이 들었습니다.

이준선



저희 조만의 특장점이자 구현에 있어 높은 난이도를 가졌던 "필터 검색" 을 이해하고 습득하는 과정에서 자바 로직을 이전 프로젝트에 비해 다양하게 활용했기 때문에 백엔드 개발자에 대한 흥미가 생겼고, 자바 로직을 분석하는 능력 또한 이전에 비해 많이 발전 되었습니다.

황지민



마무리

VARCHAR

VARCHAR

48 page



Q & A





들어 주셔서 감사합니다.
이상입니다~!

하
양
게
불
태
웠
어

