
MVC pattern2 based
used car sale
responsive web site
- VARCHAR

# 概要

VARCHAR

VARCHAR

2 page

## 1 プロジェクト概要

- 企画, 役割, 開発環境, 日程, バージョン管理, ウェブページ構成

## 2 設計

- ERD, UserFlow, LogicProcess

## 3 機能

- 主機能紹介, API, Plugin

## 4 試演

## 5 終

# プロジェクト概要

VARCHAR

VARCHAR

3 page

- 企画背景
- 役割分担
- 開発環境
- 開発日程
- バージョン管理
- ウェブページ構成
- 反応型ウェブデザイン

# 企画背景とその概要

VARCHAR

VARCHAR

4 page

- **企画背景**
  - 最近半導体大乱と原価上昇によるオンライン中古車市場の活発性

- **企画目的**
  - 繊細な商品検索機能を持つウェブページ具現

# 役割分担

VARCHAR

VARCHAR

5 page

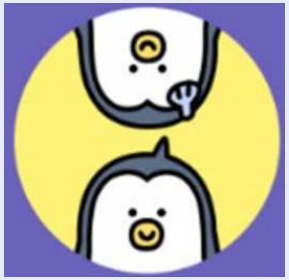
## 코알라 조 : 코딩 알려줘라

### 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

### 金宗賢

main : VIEW
sub : CTRL
role : Front 総括
c補助

### 이향준

메인 : CTRL
서브 : MODEL
주 담당 : 필터 검색 기능

### 황지민

메인 : CTRL
서브 : MODEL
주 담당 : 프로젝트 총괄

# 開発環境

VARCHAR

VARCHAR

6 page

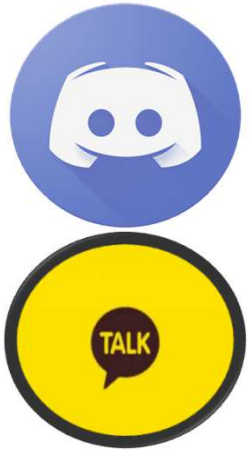
## 総合開発環境(IDE)


## DBMS


## Server


## 使用言語


## 疎通




## バージョン管理

# 開発日程

VARCHAR

VARCHAR

7 page

| 企画日程 | AUG | | | | | | | | | | | | | | | | | | | SEP | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 主題選定&役割分担 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| DB構築 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 変数, method設計 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| API (メール、SMS等) | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| UI　具現 / 配置 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 有効性検索 / テスト | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| PPT&発表準備 | | | | | | | | | | | | | | | | | | | | | | | | | | | |

# バージョン管理

VARCHAR

VARCHAR

8 page

## 8/12~8/16
**1.0.0 V**
- ウェブtemplate選定

## 8/18~8/21
**2.0.0V 主題変更**
- ウェブtemplate選定
- 検索機能 / カートUI具現
- 中古車データーcrawling

## 8/24
**2.0.1 V**
- 一次バグ修正
- 検索SQL文修正

## 8/27
**2.1.0 V**
- マップ API 追加
- 会員情報入力機能具現
- 検索機能改善

## 8/30
**2.2.0 V**
- 検索SQL文修正
- 掲示板コメント機能追加

## 8/31
**2.3.0V**
- SMS API 追加
- 範囲検索機能追加
- Model バージョンアップ

## 9/01
**2.4.0V**
- Email API 機能具現
- カート 削除機能追加

## 9/03
**2.4.1 V**
- 検索SQLソース改善

## 9/04
**2.4.2V**
- テスト
- Cバージョンアップ

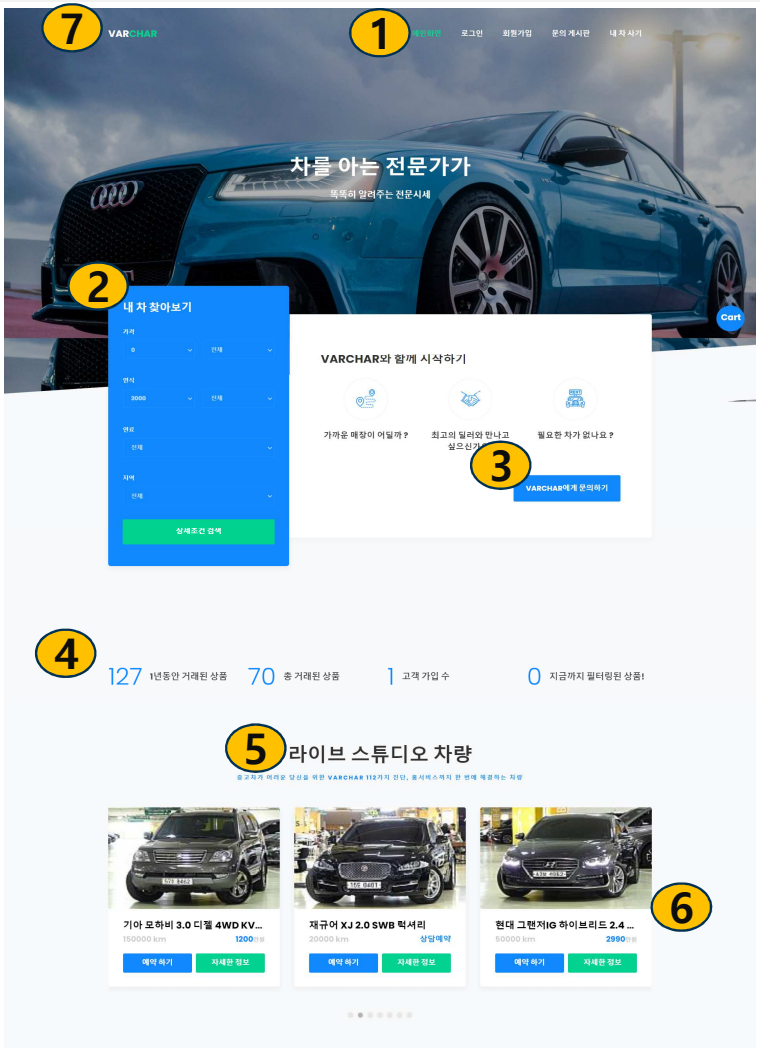
## 9/06
**2.4.3V**
- ソース確認,
- コメント管理

# ウェブページ構成[0] – Main

VARCHAR

VARCHAR

9 page



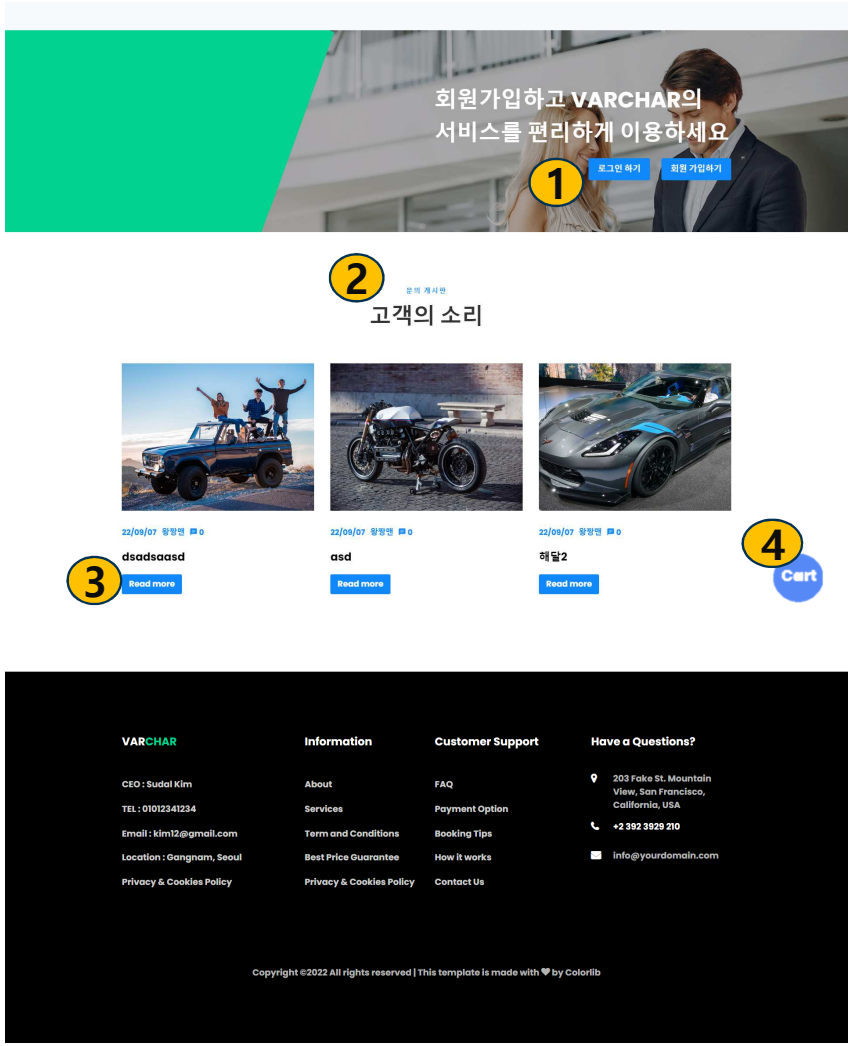

| DEVELOPE Description | |
|---|---|
| 1 | 各ページ移動 |
| 2 | 商品検索 |
| 3 | 問い合わせページ移動 |
| 4 | 各種統計 |
| 5 | ランダム商品推薦 |
| 6 | カートオープナー |
| 7 | メインページ移動 |

# ウェブページ構成[1] – Main

VARCHAR

VARCHAR

10 page

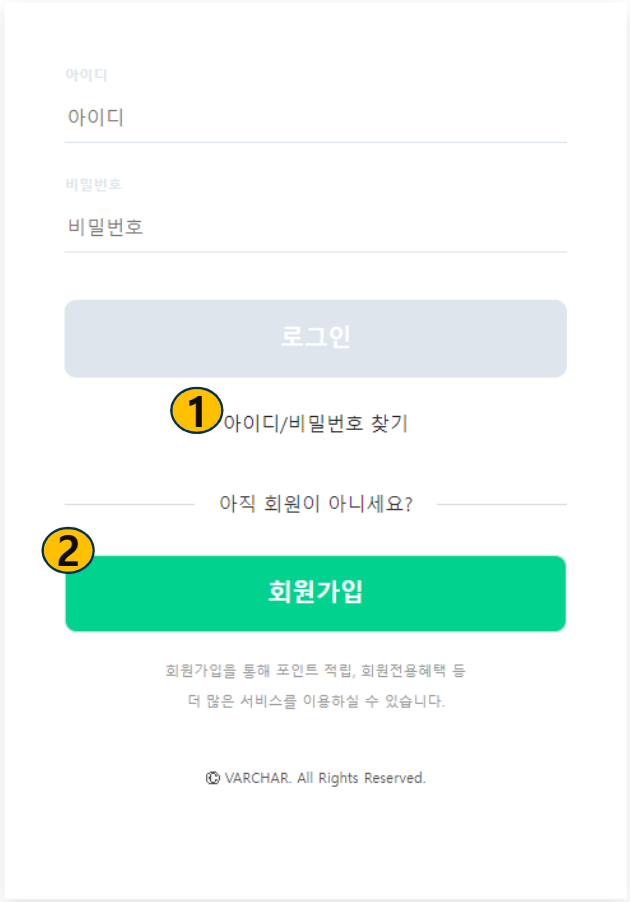



| | DEVELOPE Description |
|---|---|
| 1 | Login、会員登録ボタン |
| 2 | 最近作成された三つのポスト |
| 3 | お問い合わせページ移動 |
| 4 | カートオープナー |
| 5 | |

# ウェブページ構成[2] – Login, ID/PW検索

VARCHAR

VARCHAR | 11 page





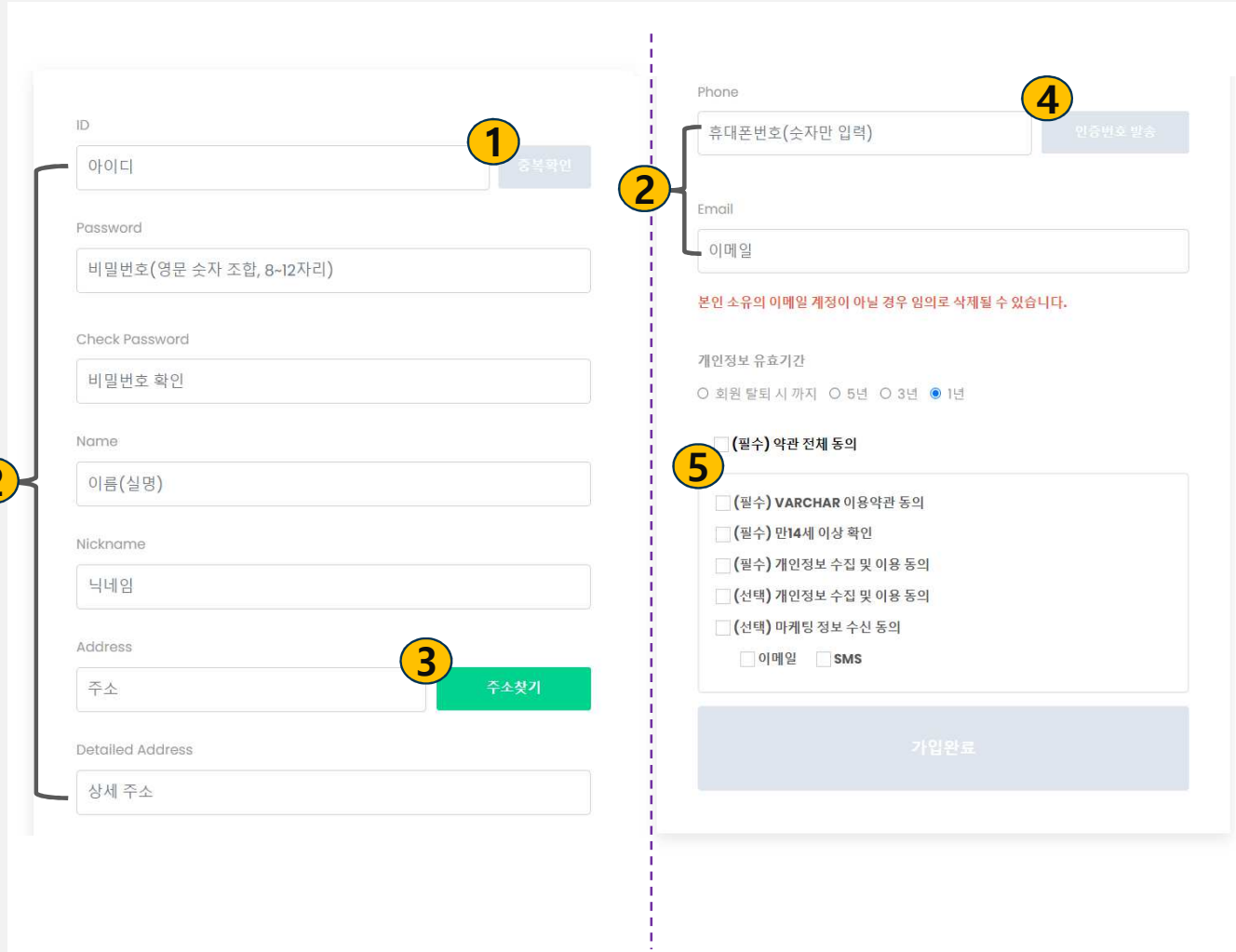
| DEVELOPE Description | |
|---|---|
| 1 | ID、暗証番号検索ページ移動 |
| 2 | 会員登録ページ移動 |
| 3 | 検索されたIDをメール発送(AJAX、メールAPI) |
| 4 | ログインページ移動 |
| 5 | |

# ウェブページ構成[3] – 会員登録

VARCHAR

VARCHAR

12 page

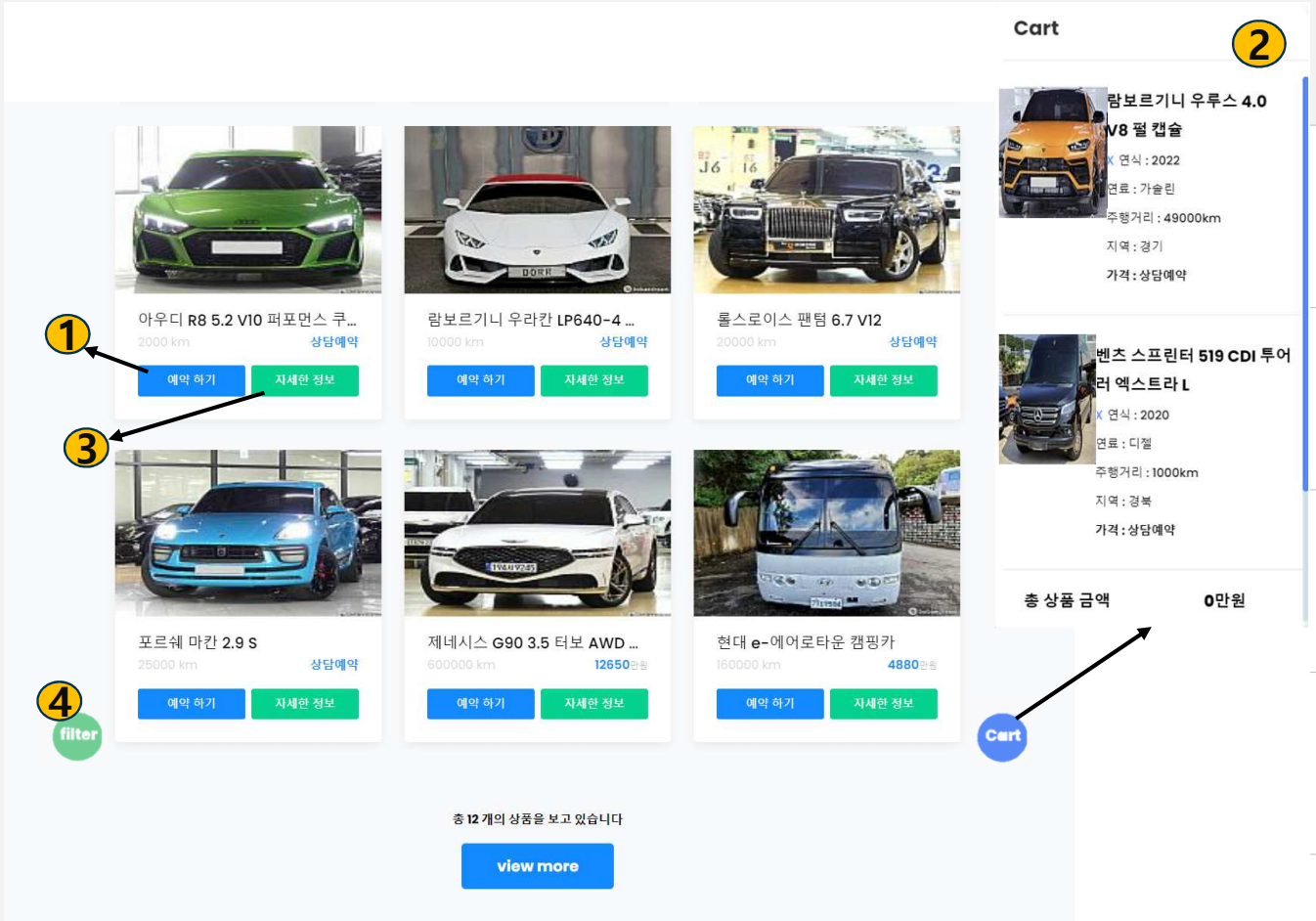



| DESIGN Description | |
|---|---|
| 1 | ID修復検査 |
| 2 | 会員情報入力 |
| 3 | 住所検索（カカオマップ API 使用） |
| 4 | 携帯認証番号発送（SMS API） |
| 5 | 約款同意ボタン<br>（必須要素同意後ボタンクリック可能） |

# ウェブページ構成[4] – 商品一覧

VARCHAR

VARCHAR | 13 page




| | DESIGN Description |
|---|---|
| 1 | 予約ボタン（カート入り） |
| 2 | カートオープン（総額計算可能） |
| 3 | 該当商品詳細ページ移動 |
| 4 | 商品検索（OPEN/CLOSE可能） |
| 5 | |

# ウェブページ構成[5] – 商品検索

VARCHAR

VARCHAR

14 page


| | DESIGN Description |
|---|---|
| 1 | 商品検索 |
| 2 | 検索ボタン |
| 3 | 検索条件初期化 |
| 4 | |
| 5 | |

# ウェブページ構成[6] – 商品詳細

VARCHAR

VARCHAR | 15 page

랜드로버 레인지로버 4.4 P530 LWB 오토바이오그래피 7인승 상담 예약

73000KM

1 찜하기

주행거리 73000km | 가격 상담예약 | 연식 2023 | 지역 경기 | 연료 가솔린

2 주요사항 딜러의 한마디

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

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

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 정상
- 프론트휀더 정상
- 도어 정상
- 사이드실패널 정상
- 트렁크리드 정상
- 루프패널 정상
- 쿼터패널 정상

- 프론트패널 정상
- 크로스멤버 정상
- 인사이드패널 정상
- 사이드멤버 정상
- 휠하우스 정상
- 패키지트레이 정상
- 대쉬패널 정상
- 플로어패널 정상
- 필러패널 정상

차를 잘 몰라서 불안하신가요?
걱정하지 마세요! 차를 아는 전문가가 직접 확인하고 진단한 차량입니다.

오토벨 상세 진단서의 정보와 실제 차량 상태가 상이할 경우, 최대 150만원까지 보상해 드립니다.
(구매 확정 후 3개월 이내, 주행거리 5,000km 이내 차량에 한함)

3 이 차량을 VARCHAR 가 보증합니다!

4 뒤로가기 메인으로가기

## DESIGN Description

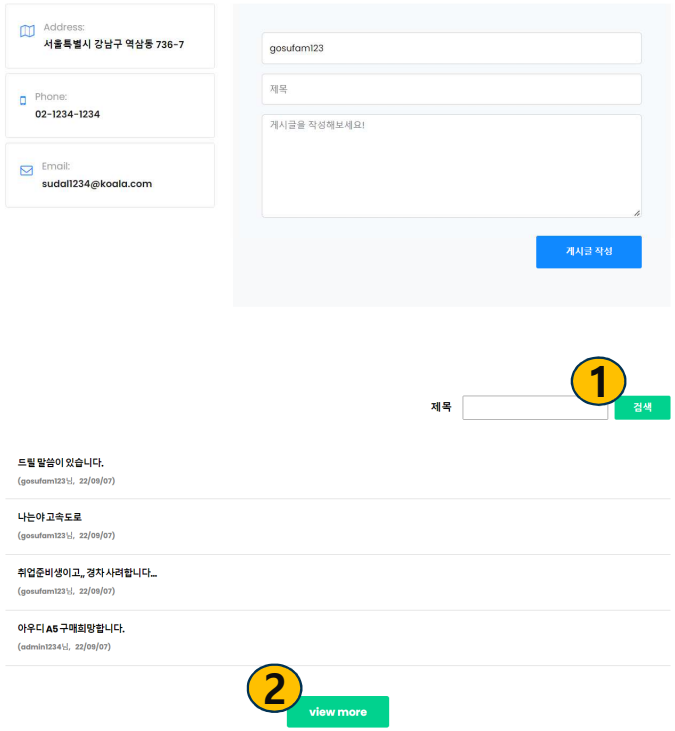
| | |
|---|---|
| 1 | カート入り |
| 2 | 詳細 |
| 3 | VARCHAR LOGO |
| 4 | 直前のページ移動 |
| 5 | |

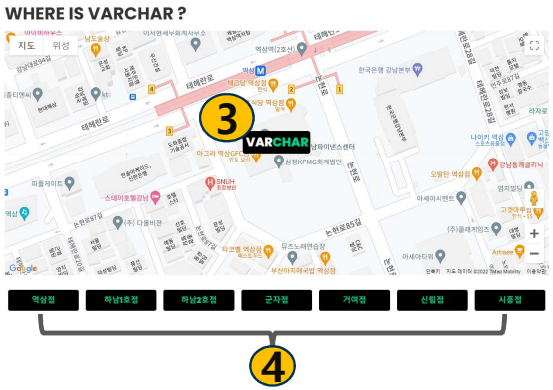
# ウェブページ構成[6] – お問い合わせ

VARCHAR

VARCHAR

16 page



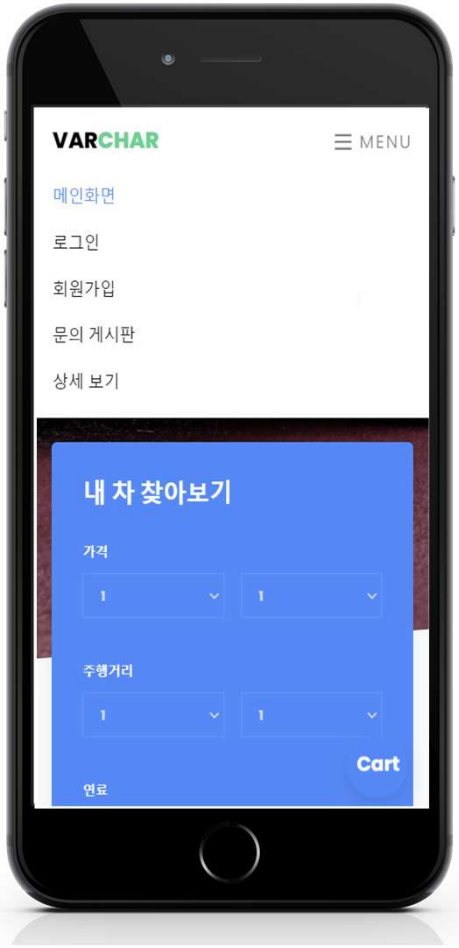

| | DESIGN Description |
|---|---|
| 1 | 検索ボタン |
| 2 | もっと見る |
| 3 | VARCHAR LOGO |
| 4 | 各支店位置情報(GOOGLE MAP API) |
| 5 | |

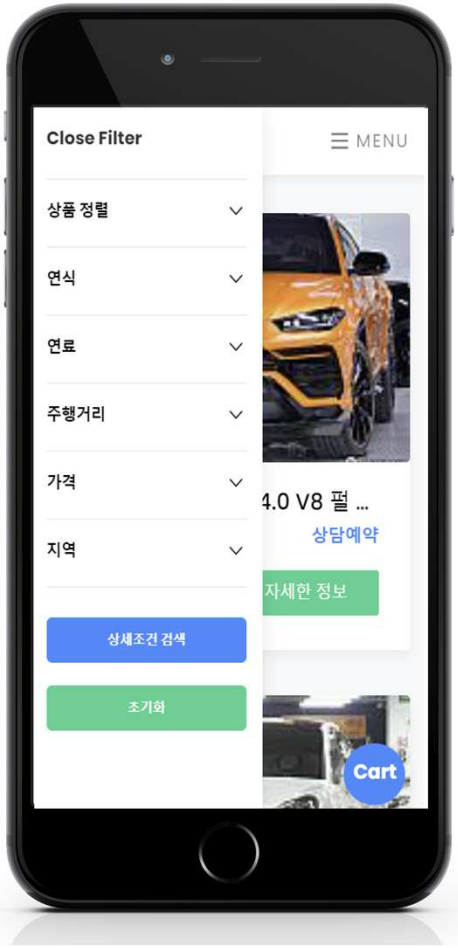
# RESPONSIVE WEB PUBLISHING

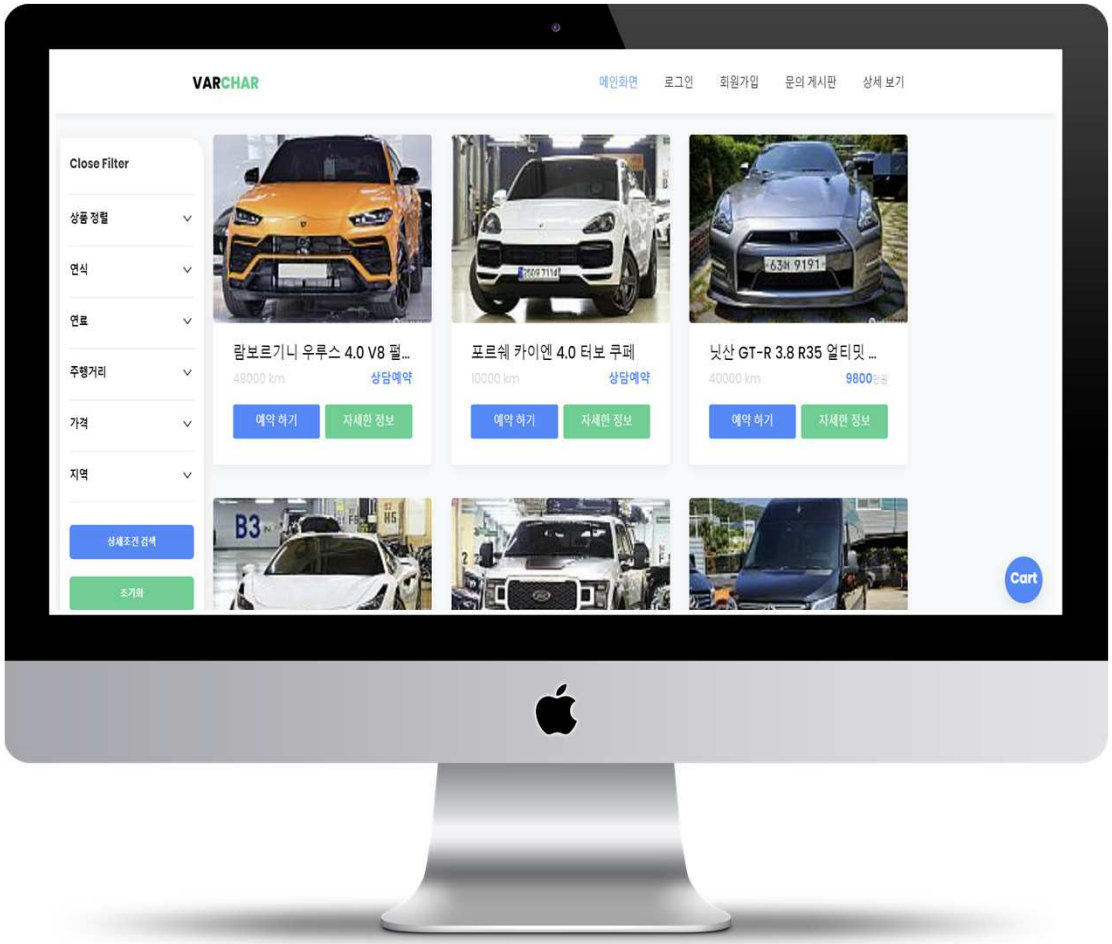
VARCHAR

VARCHAR

17 page



- ERD
- User Flow
- Logic Process

# ERD

VARCHAR

VARCHAR

19 page

| 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 |

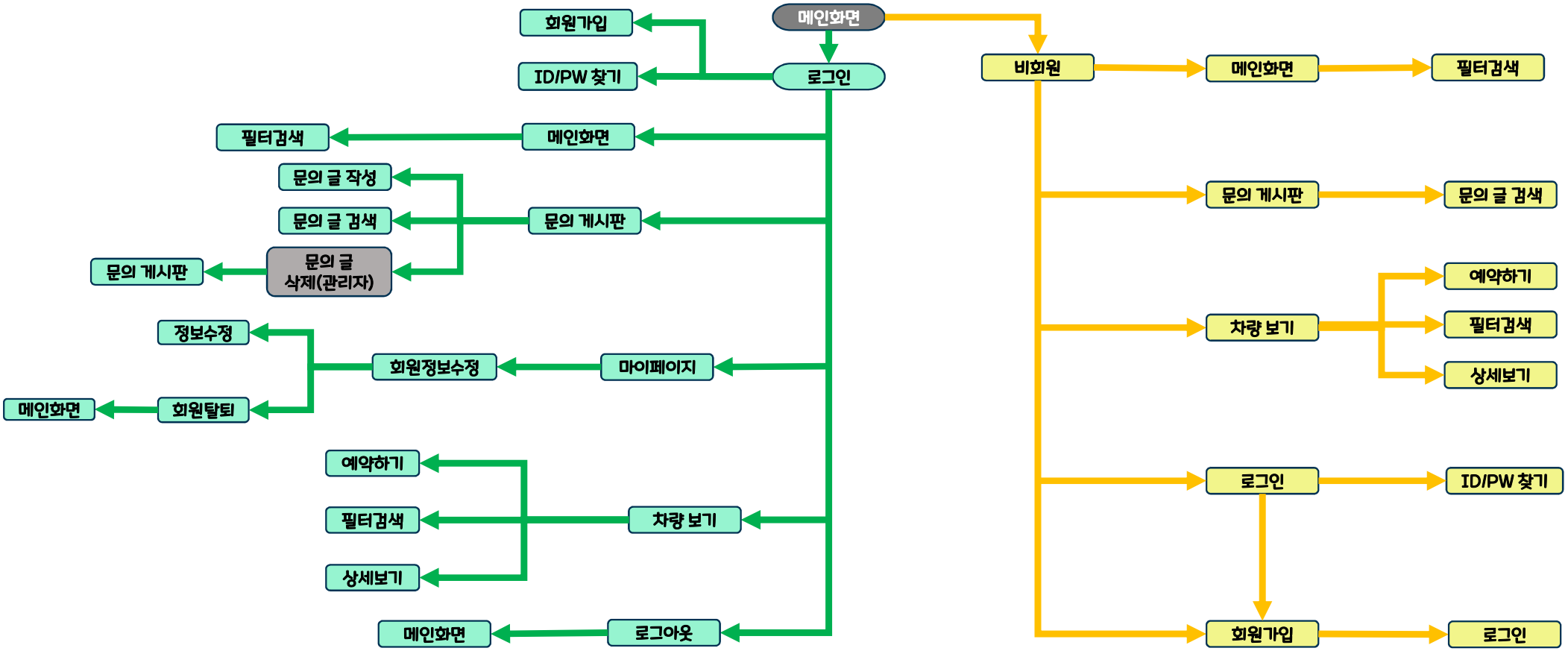
| CAR | | | |
|---|---|---|---|
| 차량고유번호 | CNUM | INT | PK |
| 차량 품명 | CTITLE | VARCHAR(300) | NOT NULL |
| 차량 정보 | CSUBTILE | VARCHAR(300) | NOT NULL |
| 차량 연식 | CYEAR | INT | NOT NULL |
| 차량 연료 | CFUEL | CARCHAR(20) | NOT NULL |
| 주행 거리 | CKM | INT | NOT NULL |
| 차량 가격 | CPRICE | INT | NOT NULL |
|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

# User Flow

VARCHAR

VARCHAR

20 page

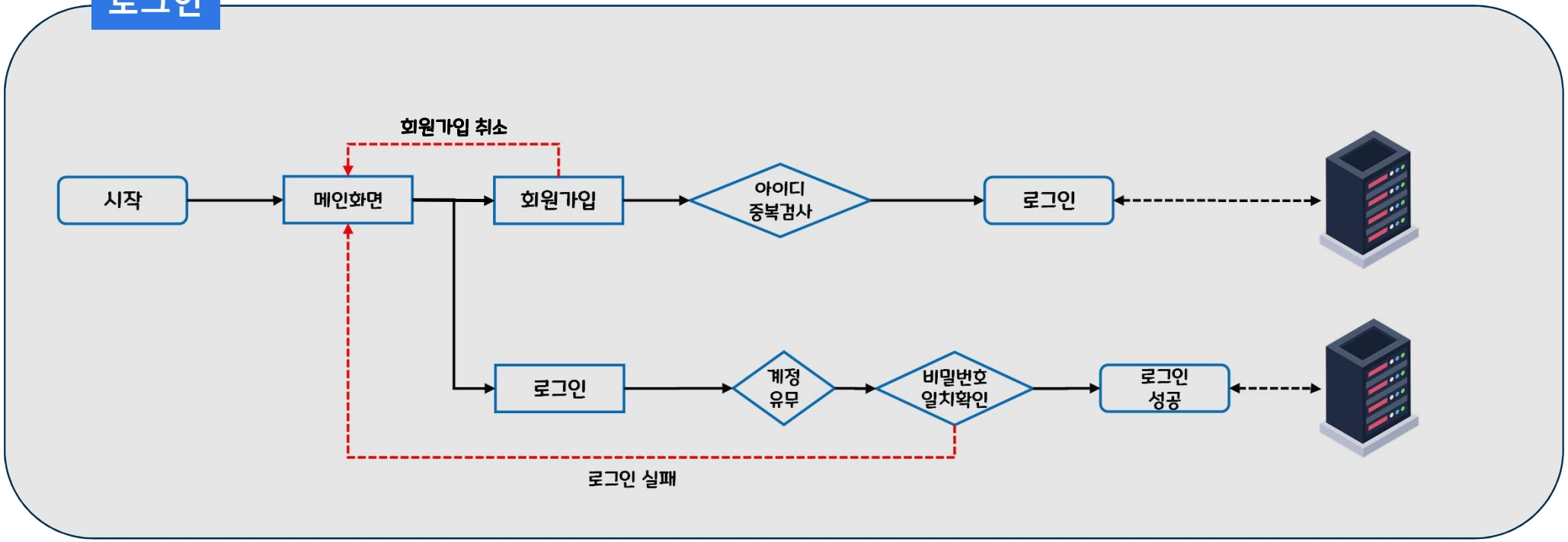


# Logic Process[0] - Login

VARCHAR



## 로그인

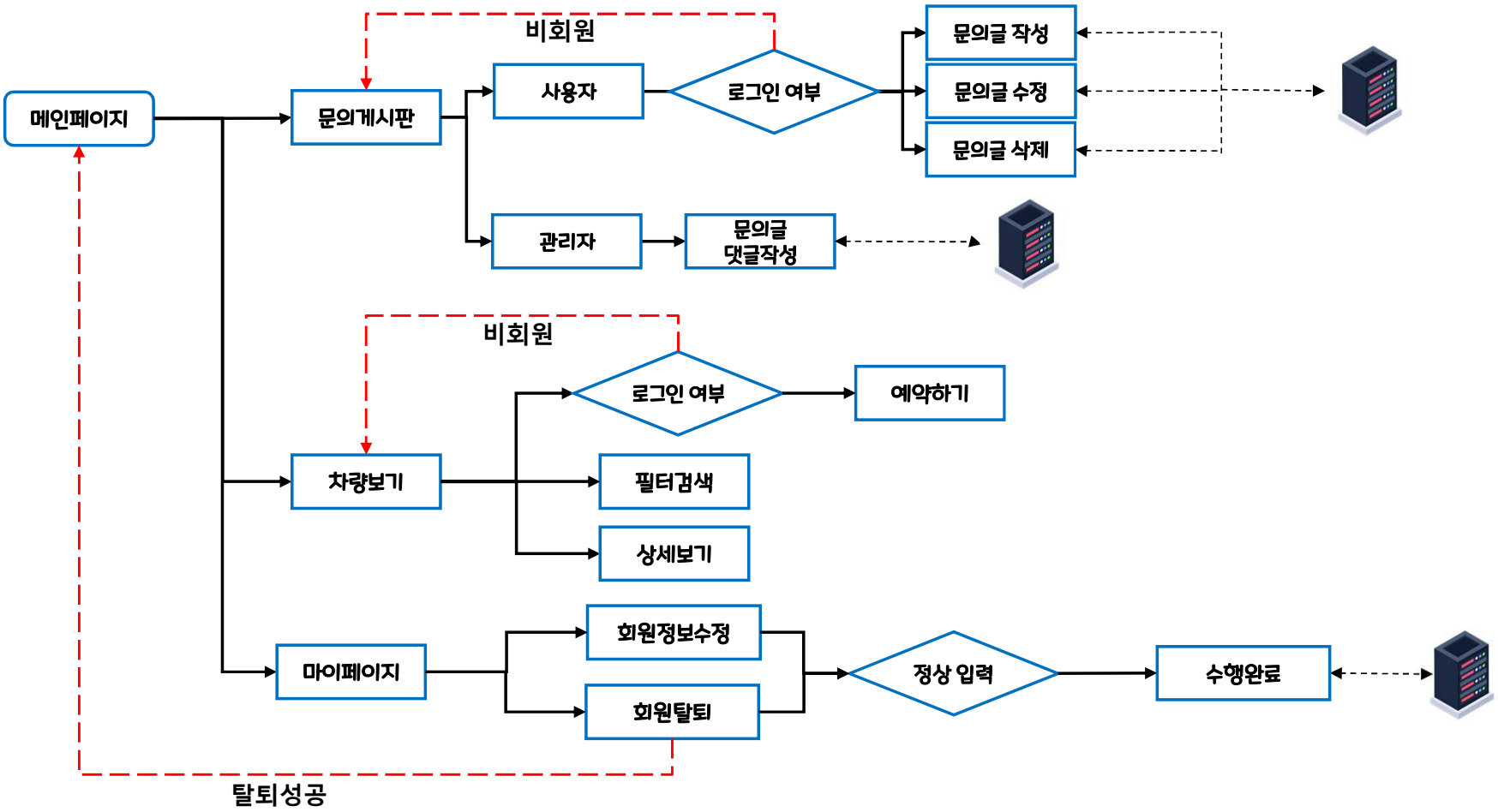


# Logic Process[1]

VARCHAR

VARCHAR

22 page



- 主機能(ソース解説)
- API, Plugin

# 主機能 – 商品検索[0]

VARCHAR



## 検索：SearchVO.java

```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;
```

※ CLIENTの検索情報基盤

- Checkboxは未選択 / 多重選択可能の為Array利用

# 主機能 – 商品検索[1]

VARCHAR



**価格検索：CarSearchAction.java**

```
// 가격
//검색할 때 입력한 값이 null이 아닌경우 = 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());
```

※ 未選択の場合、全体検索

- 受けたデーターが無いケースの処理必要

---

**販売地域検索：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);
```

- 多重選択が可能なため配列を利用
- コメント作成データーの受信確認

# 主機能 – 商品検索[2]



挿入されるSQL文：SearchDAO.java

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

※ main SQL文と挿入されるsub SQL文初期化

- 条件文によって文字列が与えられるsub SQL文
- sub SQL文を調合し完成されるmain SQL文

---

販売地域検索: 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()+")";
}
```

※ sub SQL文が完成されるロジック

- 文字列の調合に必要なStringBuilder
- Append関数で文字列追加
- sub SQL文に完成された文字列挿入

"AND Ccity IN ('서울', '경기')"

# 主機能 – 商品検索[3]

VARCHAR



★ ★ ★検索条件に基づき商品を照会するmain SQL文 ★ ★ ★ : 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”;

商品整列 : 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";
}
```

※ sub SQL分が調合せれ完成する main SQL文

- 一目の()
  - WHERE 1=1 は常に真を意味
  - 後に続く条件に該当する商品データ照会
- 二目の()
  - 一目の()によって 照会されたデーターを整列
- 三目の()
  - ROWNUMによって最終照会データーの範囲決定

# 主機能 – 商品検索[4].検索条件維持[0].checkbox,optionタグ

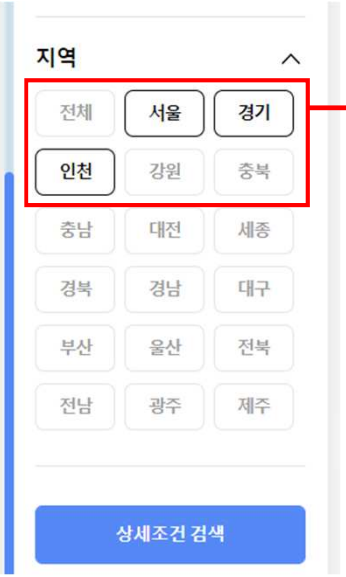
VARCHAR



**検索条件維持：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>
```




※ 実際 VARCHAR サイト検索　UI

- 検索後直前の検索条件維持

- ※ clientの便宜を図り検索後直前の検索条件維持
- JSTL coreタグの <c:if>タグ利用

# 主機能 – 商品検索[5].検索条件維持[1].範囲検索

VARCHAR



### 範囲検索条件維持: CarSearchAction.java

```
//주행거리
//        System.out.println("형변환 전 : "+request.getParameter("min-value"));
        if(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);
//            System.out.println("형변환 후 : "+km_max);
        }
```

8 7 9 12 11 10

filterSearch.jsp

```
<script id="Slider" kmin="${kmin}" kmax="${kmax}"
    src="js/rangeSlider.js"></script>
```

Clientからの検索条件をViewに伝達

### 範囲検索条件維持: filterSearch.jsp

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

1 6

### 範囲検索条件維持: rangeSlider.js

```
// Initialize slider:
$(document).ready(function () {
    var filterSlider = document.getElementById("Slider");
    //scriptTag로 kmin,kmax 값 전달 받기
let kmin = filterSlider.getAttribute("kmin");
let kmax = filterSlider.getAttribute("kmax");
//null값이 아니라 false로 해야함 => js로 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 hidden form 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 3 2 3 4 5

# 機能 – Listener Crawling.データー加工

VARCHAR

VARCHAR | 30 page

### 動的 crawling ロジック：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");
    }
}
```

### crawling データー選別：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
```

### crawling データー加工: 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";
    }
```

※ Listenerでサンプルデータの存在確認

- Listenerは最優先に作業遂行

- ターゲットサイトから持ち込むデータ選別
- データを適切で利用可能な形に加工

# 機能 – Encoding Filter



**xml 設定 : web.xml**

```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>
```

※ Encoding 設定によって全ページ適用可能

- Xmlファイルで UTF-8設定

**EncFilter.java**

```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");
    }

}
```

- 全ページにEncoding適用
- FilterConfigが web.xmlの設定値受信

# API – E-mail API[0]. ID検索[0]

VARCHAR



**E-mail 発送 ajax : findAccount.jsp**

```
//아이디 전송
document.getElementById("c          Id").onclick = () => {
    const name = $("입력한 이름").val();
    const email = $("#e          Id").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);
            }

        });
```

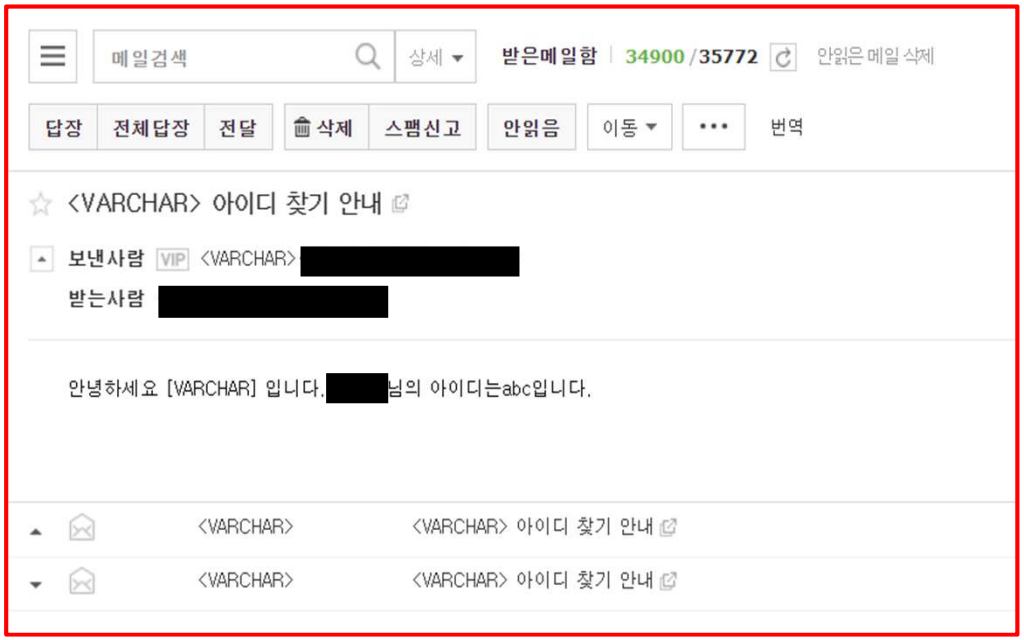
아이디 발송 버튼

입력한 이름

입력한 이메일

※ 発送ボタンクリック後 入力された名前、メール情報を POST式で Controller伝達

- 入力メールにID情報伝達




※ ロギングでデーター確認

# API – E-mail API[0].ID検索[1]

VARCHAR



### Controller : SendIdEmail.java

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) th

    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"));
5   System.out.println(mvo);
    String mid = edao.sendIdMail(mvo);   1

    //요청했던 곳으로 아이디를 보낼예정
    response.setContentType("text/html; charset=UTF-8");//
    response.getWriter().write(mid+"");
}
```

### 会員 ID 照会後セーブ : emailIdDAO.java

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


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

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

### Return ID : emailIdDAO.java

```
    // 메일 발송
    Transport.send( message );

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

### ID セーブ : 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);
    }
```

**※ 入力された会員情報を基盤にIDを検索し伝達する過程**

# API – E-mail API[0].ID検索[2]

VARCHAR



**成功 / 失敗の場合: 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);
            }

        });
```

※ 条件文によって成功 / 失敗の場合のページ上コメント

# API – E-mail API[1].暗証番号検索[0]

VARCHAR

VARCHAR

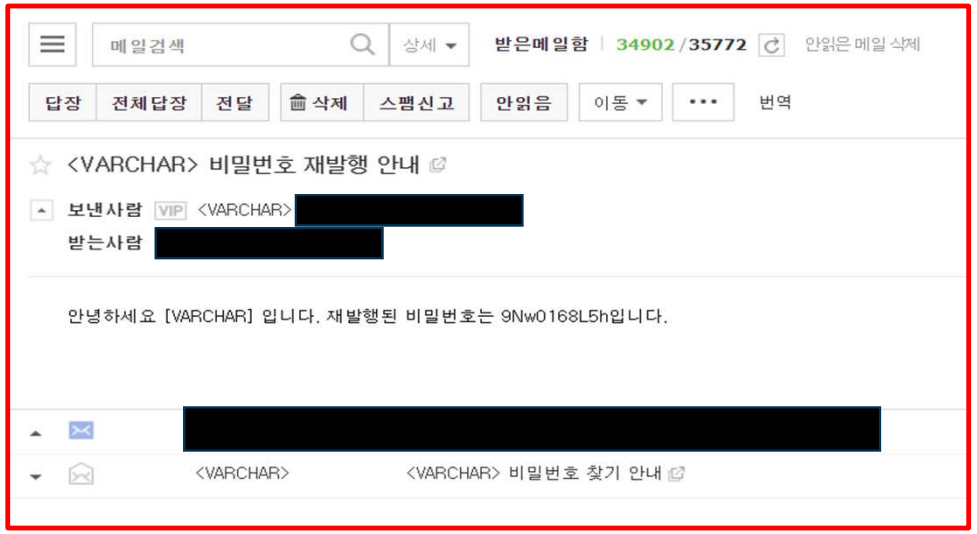
35 page

ランダム暗証番号発行ロジック: emailPwDAO.java

```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);
```

※ 現暗証番号を新たな暗証番号に変更

- Char、intタイプの値を組み合わせランダム暗証番号発行後メール送信




※ StringBufferを利用し文字列調合

# API – E-mail API[1].暗証番号検索[1]

VARCHAR



ランダム暗証番号発行後メール伝達: emailPwDAO.java

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

※ 現在の暗証番号を
UPDATE SQL文で変更

ランダム暗証番号発行: emailPwDAO.java

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

            // 메일 발송
            Transport.send( message );

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

※ 新たなランダム暗証番号を発行

# API – SMS API[0].認証番号

VARCHAR



## 携帯番号認証: signup.jsp

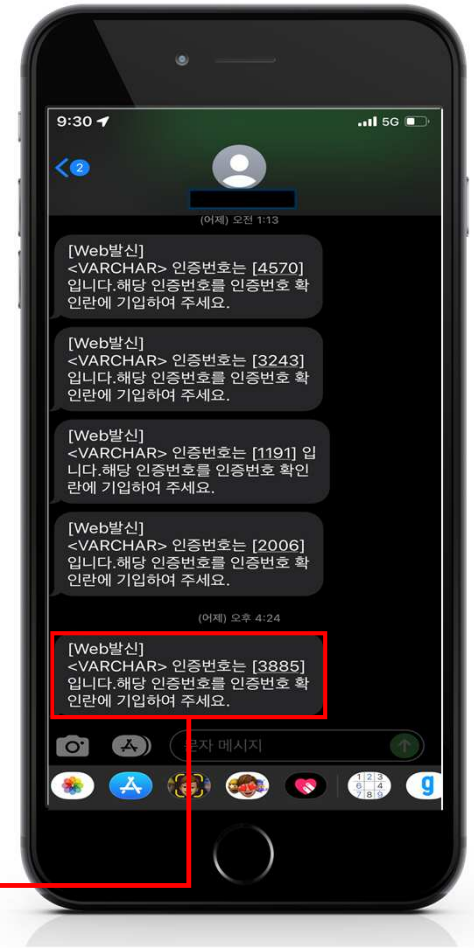
```
function startTimer(count, display) {
    //console.log("로그 : startTimer 시작");
   //문자 API - 해당 파일 160번 라인
    const phone = $("#phoneinput").val(); // 입력한 번호 값
    console.log("로그 : phone " + phone );
       $.ajax({
          type:'POST',
          url: '${pageContext.request.contextPath}/sendMSG',
          data: {phone:phone}, // 왼쪽은 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 시작되야 하기 때문에 문자발송 코드 이후에 붙인것)
      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 + ")";

       // 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);
```

## 認証タイマー : signup.jsp

```
// 버튼 눌르게 되면 타이머함수 실행
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 발송 / 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", "#eb5a46");
               certificateCheckBtn.disabled = false;
            }
         }
      });
   }
```

- ランダム認証番号を携帯に送りタイマーが起動
- Clientの入力番号と認証番号が一致するか確認
- 一致すると認証完了と同時にタイマーストップ

※ 実際携帯画面（SMS 受信状態）



# API – マップ API

VARCHAR



## 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>' +
            '<p>Tel: 02-1234-5678</p>' +
            '</div>',
             maxWidth: markerMaxWidth
      });
    google.maps.event.addListener(marker, 'click', function() {
        infowindow.open(map, marker);
    });
}
```

## VARCHAR 支店位置UI: 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=[redacted]&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>
  </div>
</section>
```

**WHERE IS VARCHAR ?**


### ※ VARCHAR 支店位置情報

- 下のVARCHAR 支店ボタンクリックで位置情報を確認可能
- Google Map API 利用

# API – 住所 API

VARCHAR



## 住所検索　API：signup.jsp

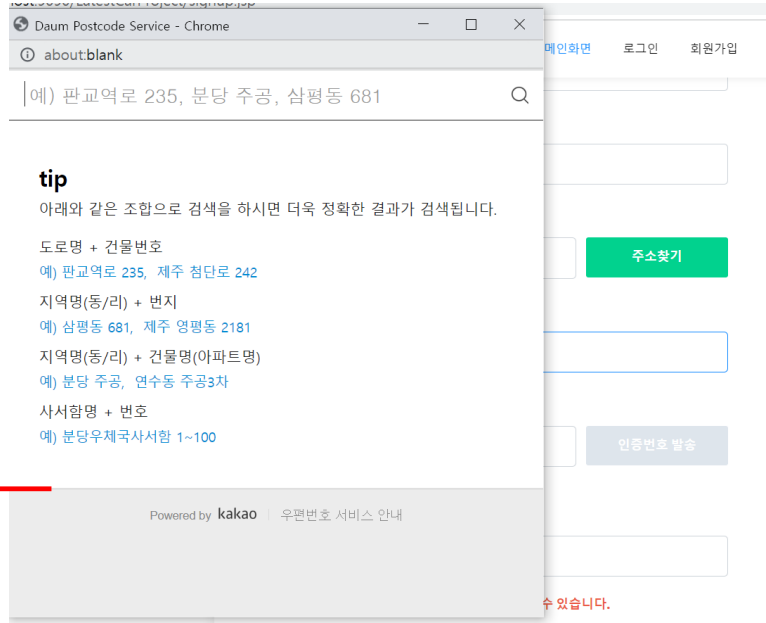
```
<!-- 카카오 주소 API -->
<script src="//t1.daumcdn.net/mapjsapi/bundle/postcode/prod/postcode.v2.js"></script>
```

```
// 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>
```

## ※ 会員登録の際、便利性向上

- ボタンクリック後稼働
- カカオマップ機能




## ※ 実際 VARCHAR 会員登録の時

# Plugin – AccordionPlugin

VARCHAR



**お問い合わせ AccordionPlugin 使用：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);
},
```

**※ ページ活用性向上の図りAccordionPlugin 適用**

**初期**

VARCHARにて파는 중고차

[삭제]

(admin123님, 22/09/07)

**クリック後**

VARCHARにて파는 중고차

[삭제]

(admin123님, 22/09/07)

최고입니다.

# Plugin – エーラ ページ



**xml 設定 : web.xml**

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

**※ 404エーラ発生後エーラ ページ移動**

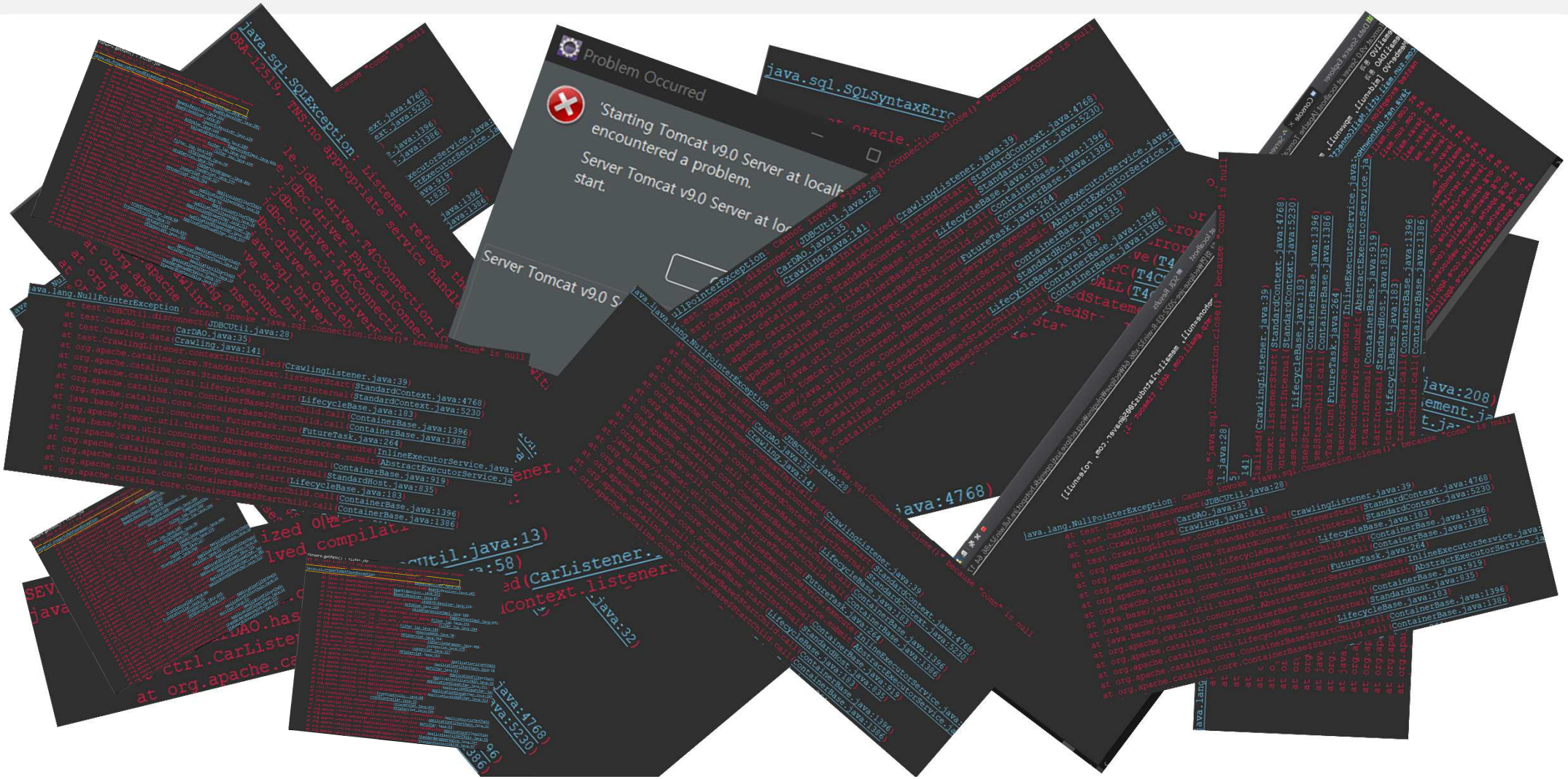
- デーラーページ

# エラー解決

VARCHAR

VARCHAR

42 page

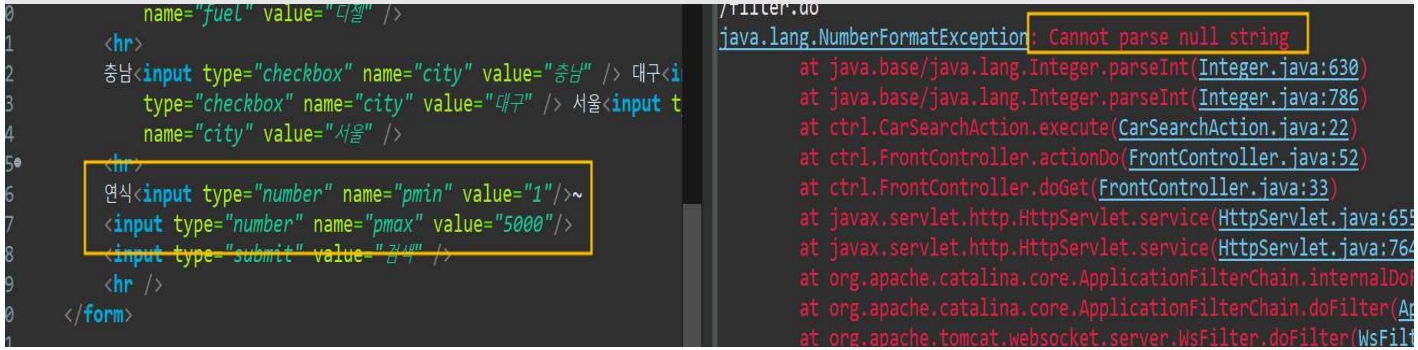


# エラー解決[0] – 商品検索

VARCHAR



**キャスチングエラー: (旧)filter.jsp**



```
        name="fuel" value="디젤" />
    <hr>
    충남<input type="checkbox" name="city" value="충남" /> 대구<i
        type="checkbox" name="city" value="대구" /> 서울<input t
        name="city" value="서울" />
    <hr>
    연식<input type="number" name="pmin" value="1"/>~
    <input type="number" name="pmax" value="5000"/>
    <input type="submit" value="검색" />
    <hr />
</form>
```

```
/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.internalDo
        at org.apache.catalina.core.ApplicationFilterChain.doFilter(Ap
        at org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilt
```

※ 最初ページ接続の際エーラ発生

**CarSearchAction.java**

```
//검색할 때 입력한 값이 null이 아닌경우 = 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());
```

- 伝達されるデーターの不在が原因
- 例外処理で解決

# エラー解決[1] – ID検索[0]

VARCHAR

VARCHAR

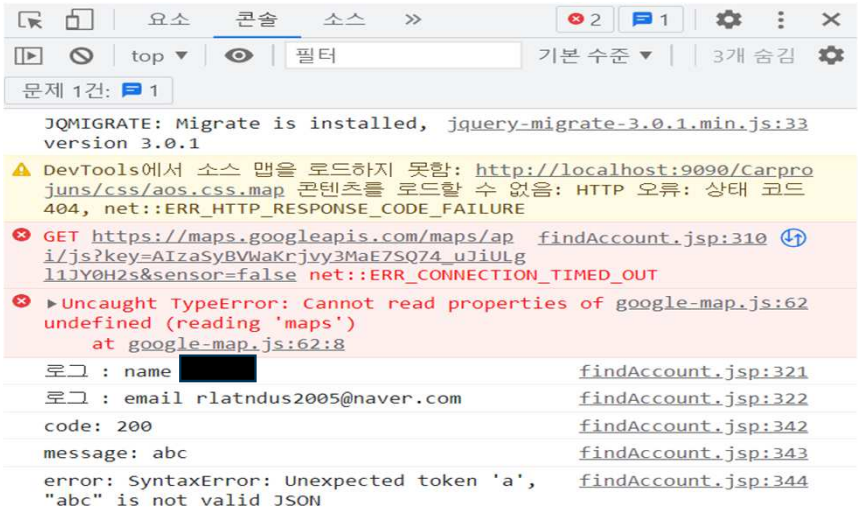
44 page

### 通信形式問題：sendIDEmail.java

```
protected void doPost(HttpServletRequest request, HttpServletResponse respons

    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);

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

### 通信形式問題：emailIdDAO.java

```
// MIME 타입 설정
MailcapCommandMap MailcapCmdMap = (MailcapCommandMap) C
MailcapCmdMap.addMailcap("text/html;; x-java-content-ha
MailcapCmdMap.addMailcap("text/xml;; x-java-content-han
MailcapCmdMap.addMailcap("text/plain;; x-java-content-h
MailcapCmdMap.addMailcap("multipart/*;; x-java-content-
MailcapCmdMap.addMailcap("message/rfc822;; x-java-conte
CommandMap.setDefaultCommandMap(MailcapCmdMap);
```




**※ 要請と応答の形の違いが原因**

- **要請はtext/html,　応答はx-json**

# エラー解決[2] – ID検索[1]

VARCHAR



```
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
```

```
50          mvo.setMemail(request.getParameter("email"));
51          System.out.println(mvo);
52          String mid = edao.sendIdMail(mvo);
53
54          //요청했던 곳으로 아이디를 보낼예정
55          response.setContentType("text/html; charset=UTF-8");//
56          response.getWriter().write(mid+"");
57      }
58
59
60
61  }
```

- どちらもtext/htmlで一致させ解決

# 試演

# 終

VARCHAR

VARCHAR

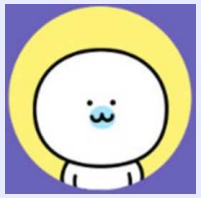
47 page

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 커져서 인지 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 보완 한다면 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

# Q & A

# 有難うございました！