버전	1.0
작성일	2021-10-16
소속	JHTA
작성자	윤인용

- 목차 -

프로젝트 개요

- 컴퓨터 부품 온라인 거래 웹사이트
- 상품의 목록에 상품을 추가하고, 업데이트하고, 삭제 할 수 있게 서비스를 구성(CRUD)
- Model 2 MVC 설계 모델로 개발
- 사용자 친화적인 웹 애플리케이션 개발을 목표로 함
  (SNS연동, 고객센터, 커뮤니티 등)
- 관리자 입장에서도 관리하기 편한 쇼핑몰

프로젝트 개요

주제 선정 의도

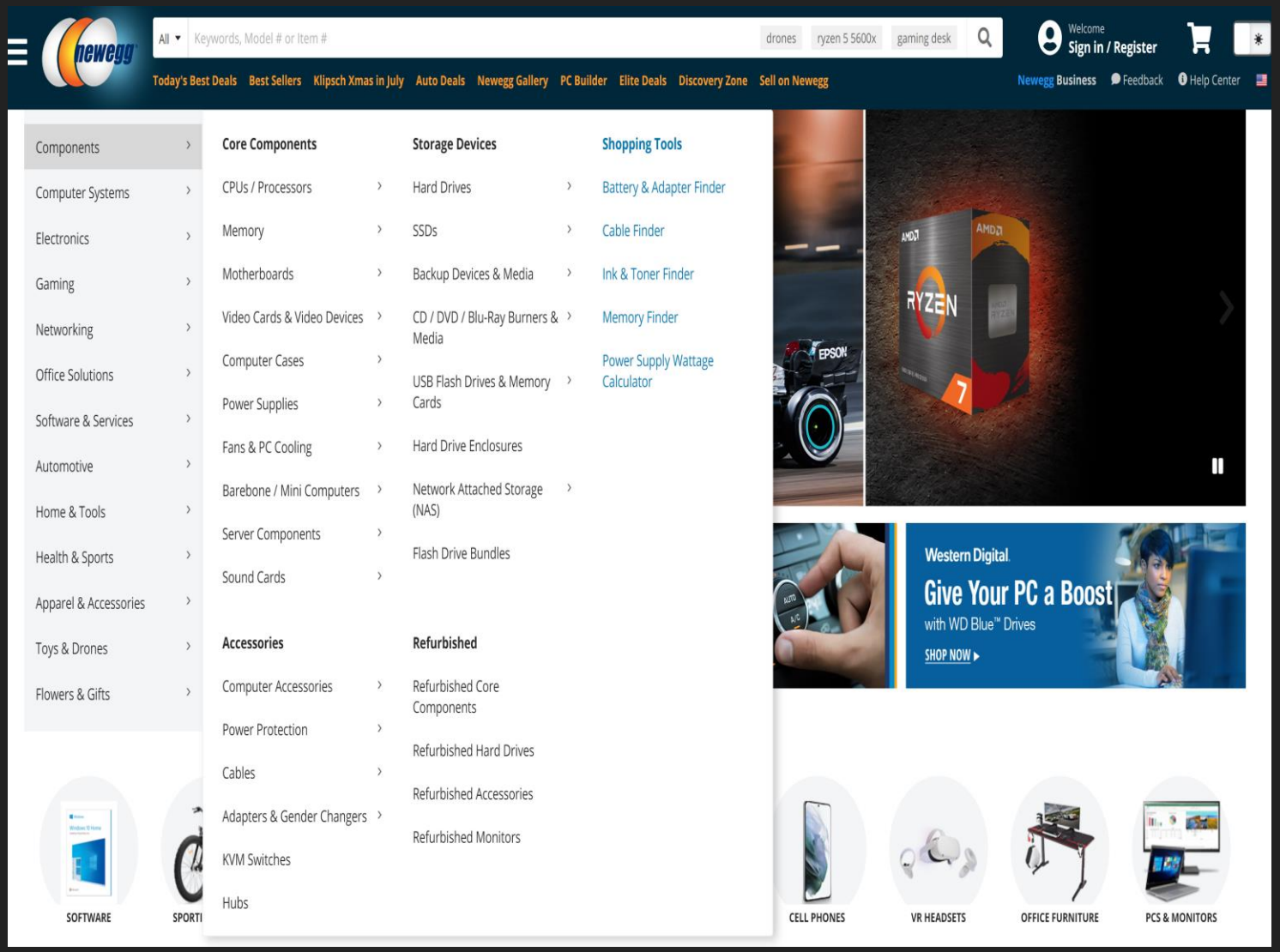
- 수강 과정동안 공부했던 것들에 대해 정확히 이해하고, 활용해 보는 것
- 저번 세미 프로젝트 때, 제대로 이해를 하지 못하고 넘어간 것들이 많았고, 다른 프로그램, 기술을 사용해보고 싶었지만 부족한 시간으로 인해 사용해보지 못했던 아쉬움
- 방대한양의 DB를 다뤄볼 필요가 있다고 생각함

사용 언어/툴/프레임워크

- 사용 언어
  - Java
  - JavaScript(jsp,jQuery)
- 프레임워크
  - Spring
  - Bootstrap
- 버전 관리 시스템
  - Git
- 빌드 툴
  - Maven
- DBMS
  - MySql

Reference

Newegg.com



-장점-

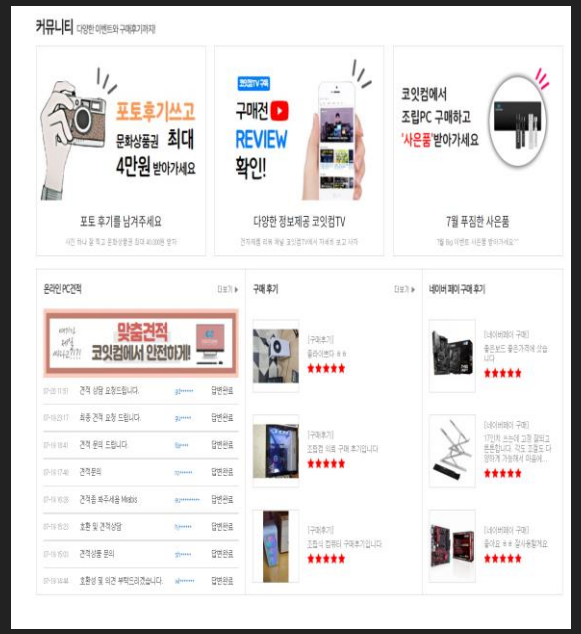
- 방대한 컴퓨터 관련 상품 목록

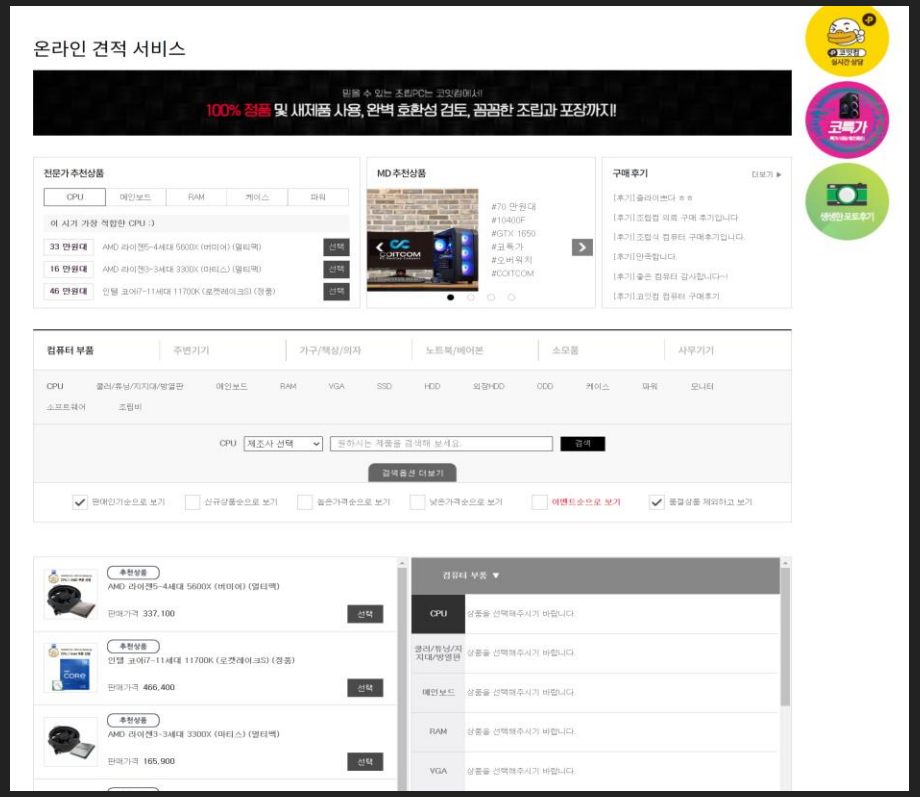
-단점-

- 불편한 고객센터
- 낮은 신뢰도의 검색 기능

Reference

코잇컴





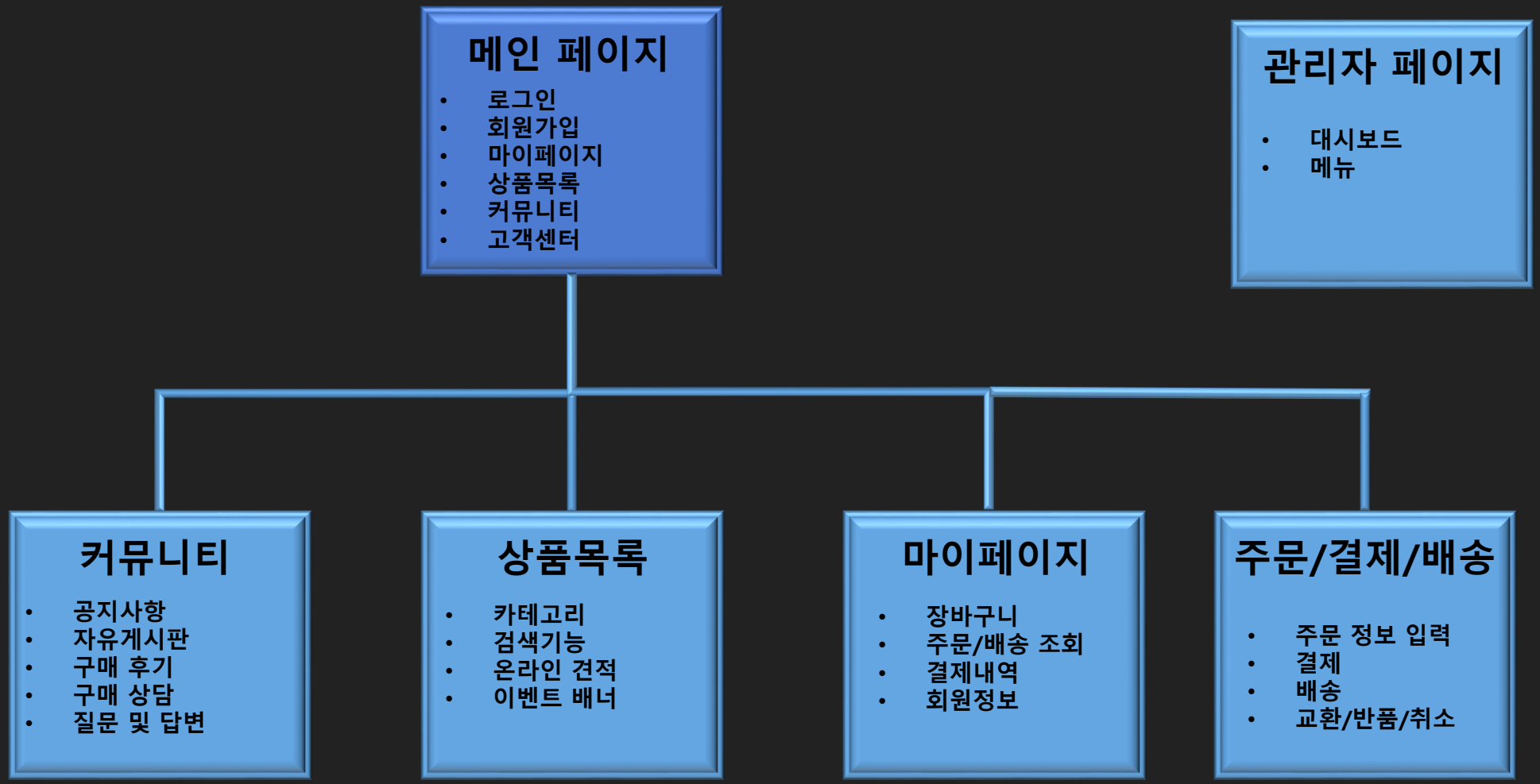
-장점-

- 온라인 견적 기능의 호환성 체크
- 다양한 커뮤니티

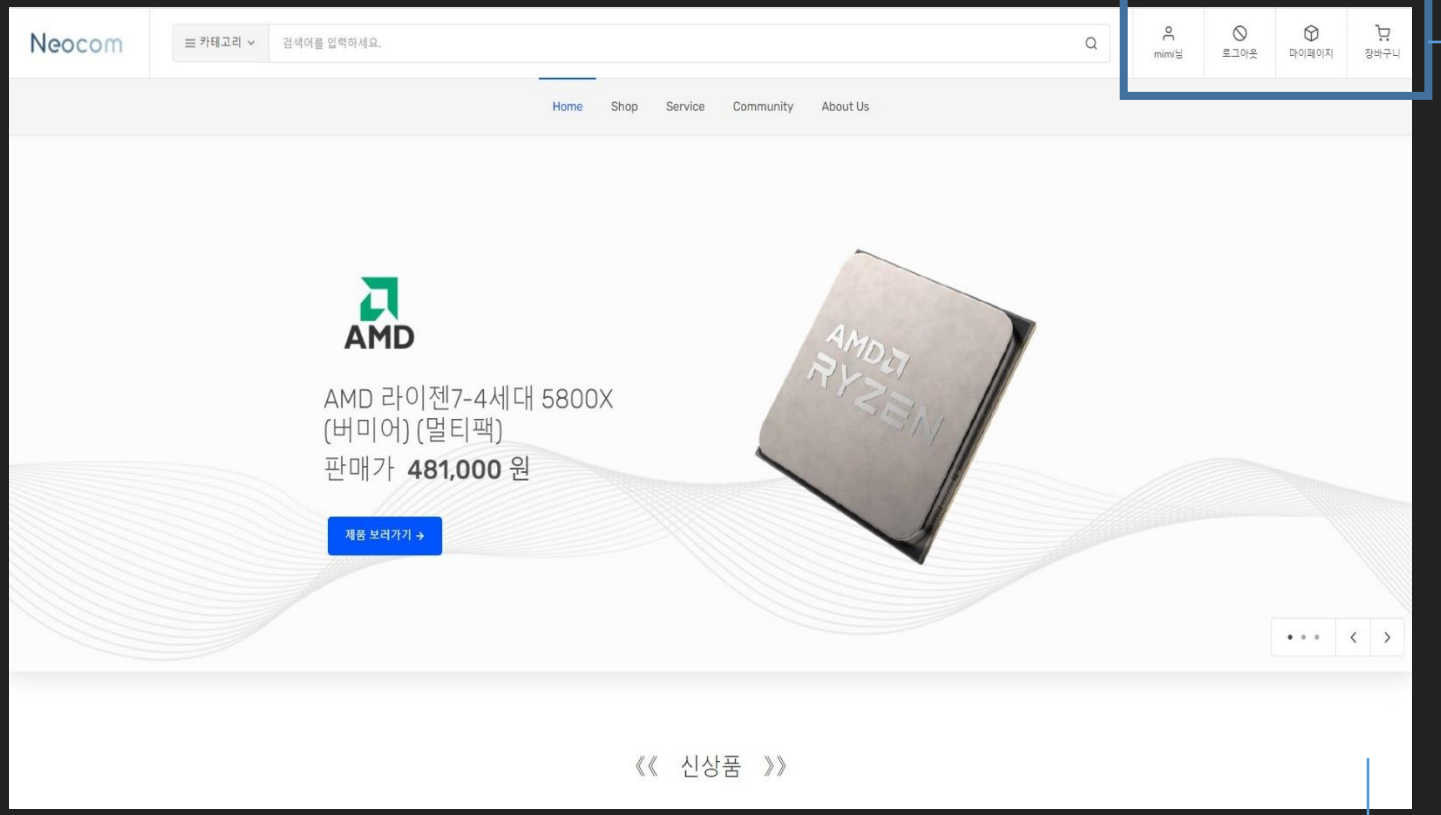
-단점-

- 복잡한 인터페이스
- 너무 많은 광고 배너

기본 인터페이스 구상도 (전체 구조)



메인페이지



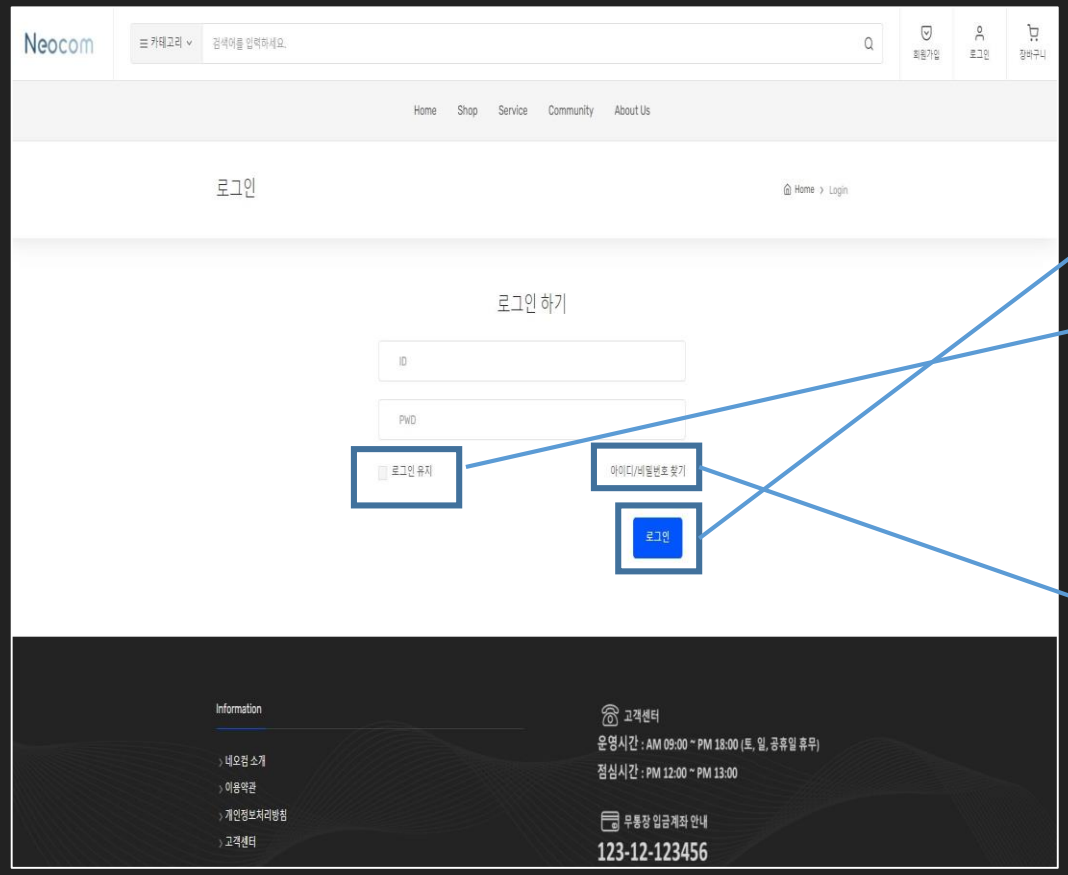
Spring Security로
로그인 기능 구현

```
<div class="toolbar-inner">
    <sec:authorize access="isAnonymous()">
        <div class="toolbar-item">
            <a href="${pageContext.request.contextPath }/account/join1">
                <div><i class="icon-pocket"></i><span class="text-label">회원가입</span></div>
            </a>
        </div>
        <div class="toolbar-item">
            <a href="${pageContext.request.contextPath }/account/login">
                <div><i class="icon-user"></i><span class="text-label">로그인</span></div>
            </a>
        </div>
    </sec:authorize>

    <sec:authorize access="isAuthenticated()">
        <div class="toolbar-item">
            <a>
                <div><i class="icon-user"></i><span class="text-label"><sec:authentication property="principal.memberVo.id"/>님</span></div>
            </a>
        </div>
        <div class="toolbar-item">
            <a href="${pageContext.request.contextPath }/account/logout">
                <div><i class="icon-slash"></i><span class="text-label">로그아웃</span></div>
            </a>
        </div>
        <div class="toolbar-item">
            <a href="${pageContext.request.contextPath }/account/mypage_order">
                <div><i class="icon-box"></i><span class="text-label">마이페이지</span></div>
            </a>
        </div>
    </sec:authorize>
```

Index.jsp

```
<section class="container padding-top-4x padding-bottom-3x mb-2">
    <h2 class="h3 pb-3 text-center">《《    신상품    》》</h2>
    <div class="row">
        <c:forEach var="vo" items="${list }" end="7">

            <div class="col-lg-3 col-md-4 col-sm-6" style="width:auto;height:auto;">
                <div class="product-card mb-30">
                    <a class="product-thumb"
                        href="${pageContext.request.contextPath}/shop/product_detail?n=${vo.product_id}&m=${vo.category_id}">

                        <img   src="<c:url value='/upload/product_img/${vo.uploadPath}/${vo.img_name_save}' />"
                        alt="<c:url value='/upload/product_img/${vo.uploadPath}/${vo.img_name_save}' />" />
                    </a>
                    <div class="product-card-body">
                        <div class="product-category">
                            <a href="#">${vo.brand }</a>
                        </div>
                        <h3 class="product-title">
                            ${vo.product_name }

                        </h3>
                        <h4 class="product-price">
                            <fmt:formatNumber pattern="###,###,###"
                                value="${vo.selling_price }" />
                            원
                        </h4>
                    </div>
                    <div class="product-button-group">
                        <a id="wish" class="product-button btn-wish" href="#" onclick="insertWish(${vo.product_id})">
                            <c:choose>
                                <c:when test="${status == 0 }">
                                    <i class="icon-heart" style="color:red;"></i>
                                </c:when>
                                <c:otherwise>
                                    <i class="icon-heart"></i>
                                </c:otherwise>
                            </c:choose>
                            <span>관심상품</span>
```

로그인페이지



LoginController.java

```java
@Controller
public class LoginController {

    @Autowired
    private MemberService service;

    @RequestMapping(value = "/account/login", method = RequestMethod.GET)
    public String loginForm(Model model, HttpSession session, boolean error) {
        if (error == true) {
            String errMsg = "아이디 또는 비밀번호가 틀렸습니다.";
            model.addAttribute("errMsg", errMsg);
        }
        return "frontend/account/login";
    }

    @GetMapping("/user")
    public @ResponseBody String user(@AuthenticationPrincipal CustomUserDetails principal) {
        System.out.println("Principal : " + principal);
        System.out.println("OAuth2 : " + principal.getUsername());
        // iterator 순차 출력 해보기
        java.util.Iterator<? extends GrantedAuthority> iter = principal.getAuthorities().iterator();
        while (iter.hasNext()) {
            GrantedAuthority auth = iter.next();
            System.out.println(auth.getAuthority());
        }

        return "유저 페이지입니다.";
    }
    @RequestMapping(value = "/account/login", method = RequestMethod.POST)

    public String login(Authentication authentication, MemberVo vo, Model model, HttpServletRequest req,
            HttpServletResponse response) throws IOException {
        System.out.println("로그인 컨트롤러 vo : " + vo);
        System.out.println("로그인 인증결과 : " + authentication.getPrincipal() + ", " + authentication.getCredentials() + ", "
                + authentication.getAuthorities());
        PrintWriter out = response.getWriter();
        out.println("<script>alert('');</script>");
        return "redirect:/";
    }

    @RequestMapping(value = "/account/denied", method = RequestMethod.POST)
    public String logout() {
        return "frontend/account/denied";
    }
}
```

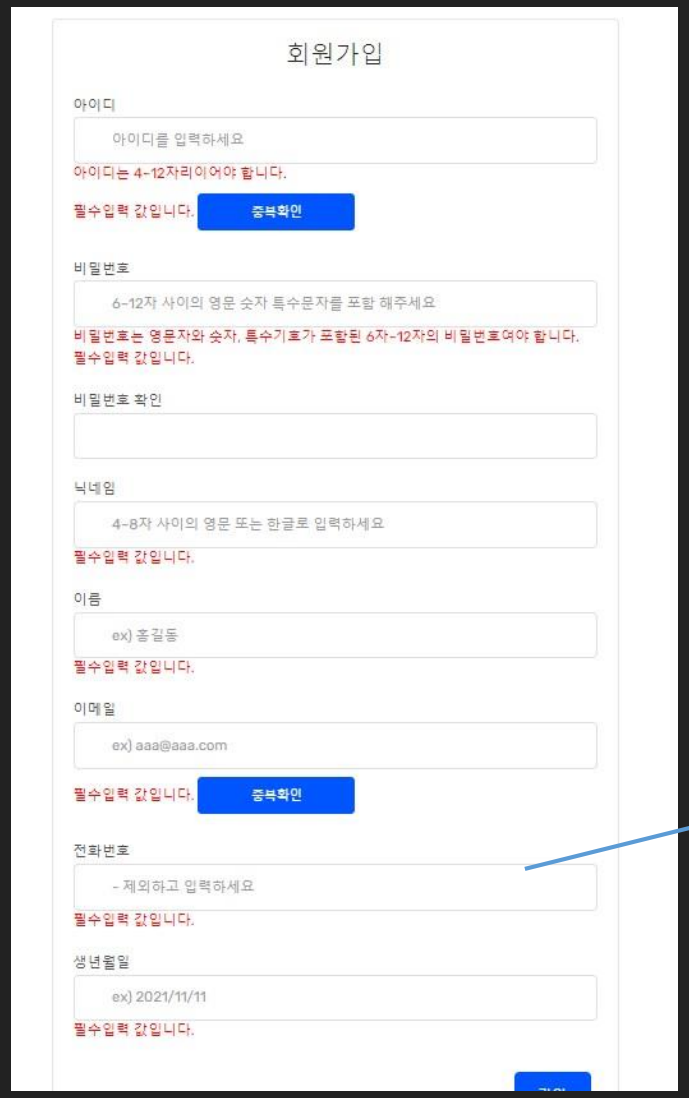
WebSecurityConfig

```java
@GetMapping("/account/findAccount")
public String findA() {
    return "frontend/account/findId_Pwd";
}

@RequestMapping(value = "/account/findId", method = RequestMethod.POST)
public String find_id(HttpServletResponse response, @RequestParam("email") String email, Model md)
        throws Exception {
    md.addAttribute("id", service.find_id(response, email));
    System.out.println("result:" + email);
    return "frontend/account/find_IdResult";

}

@RequestMapping(value = "/account/findPwd", method = RequestMethod.POST)
public String find_Pwd(HttpServletResponse response, @RequestParam("id") String id, Model md) throws Exception {
    md.addAttribute("pwd", service.find_pwd(response, id));
    System.out.println("result:" + id);
    return "frontend/account/find_PwdResult";

}
```

SpringBoot의 rememberMe기능으로 로그인유지 구현


```java
@GetMapping("/account/findAccount")
public String findA() {
    return "frontend/account/findId_Pwd";
}

@RequestMapping(value = "/account/findId", method = RequestMethod.POST)
public String find_id(HttpServletResponse response, @RequestParam("email") String email, Model md)
        throws Exception {
    md.addAttribute("id", service.find_id(response, email));
    System.out.println("result:" + email);
    return "frontend/account/find_IdResult";

}

@RequestMapping(value = "/account/findPwd", method = RequestMethod.POST)
public String find_Pwd(HttpServletResponse response, @RequestParam("id") String id, Model md) throws Exception {
    md.addAttribute("pwd", service.find_pwd(response, id));
    System.out.println("result:" + id);
    return "frontend/account/find_PwdResult";

}
```

회원가입



JoinController.java

```java
@Controller
public class Join2Controller {
    @Autowired
    private BCryptPasswordEncoder bCryptPasswordEncoder;
    @Autowired
    private MemberService service;

    @RequestMapping(value = "/account/join2", method = RequestMethod.GET)
    public String joinForm(Model model) {
        model.addAttribute("memberVo", new MemberVo());
        return "/frontend/account/join2";

    }

    @RequestMapping(value = "/account/join2", method = RequestMethod.POST)
    public String join(@Valid @ModelAttribute MemberVo memberVo, BindingResult result, Model model) {
        if (result.hasErrors()) {
            System.out.println("에러발생!!");
            return "/frontend/account/join2";
        }
        System.out.println("오예성공");
        memberVo.setPassword(bCryptPasswordEncoder.encode(memberVo.getPassword()));
        service.insert(memberVo);
        service.insert_role(memberVo.getMem_no());
        return "/frontend/account/join3";

    }

    @RequestMapping(value = "/idcheck.do", method = RequestMethod.POST)
    @ResponseBody
    public Map<Object, Object> idcheck(@RequestParam("id") String id) {
        int count = 0;
        Map<Object, Object> map = new HashMap<Object, Object>();

        count = service.idcheck(id);
        map.put("cnt", count);

        return map;
    }

    @RequestMapping(value = "/emailcheck.do", method = RequestMethod.POST)
    @ResponseBody
```

MemberVo

```java
@AllArgsConstructor
@NoArgsConstructor
@Data
@Entity
@Table(name = "member_Info")
public class MemberVo {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    // private int mem_id;
    private int mem_no;
    @NotBlank(message = "필수입력 값입니다.")
    private String nickname;
    @NotBlank(message = "필수입력 값입니다.")
    private String phone;
    @NotBlank(message = "필수입력 값입니다.")
    private String birth_date;
    private String reg_date;

    @NotBlank(message = "필수입력 값입니다.")
    private String name;
    @NotBlank(message = "필수입력 값입니다.")
    @Email(message = "이메일형식이 틀립니다.")
    private String email;
    @NotBlank(message = "필수입력 값입니다.")
    @Size(min = 4, max = 12, message = "아이디는 4~12자리이어야 합니다.")
    private String id;
    @NotBlank(message = "필수입력 값입니다.")
    @Pattern(regexp="(?=.*[0-9])(?=.*[a-z])(?=.*\\W)(?=\\S+$).{6,12}",
    message = "비밀번호는 영문자와 숫자, 특수기호가 포함된 6자~12자의 비밀번호여야 합니다.")
    private String password;

    @ManyToMany(cascade = CascadeType.ALL, fetch = FetchType.EAGER)
    @JoinTable(name = "member_Info_member_role", joinColumns = @JoinColumn(name =
    private Set<MemberRole> roles = new HashSet<>();
}
```

Hibernate사용하여 유효성 검사 진행

```java
@Controller
public class Join1Controller {
    @RequestMapping(value="/account/join1", method = RequestMethod.GET)
    public String join1Form() {
        return "frontend/account/join1";
    }

    @RequestMapping(value="/account/join1", method = RequestMethod.POST)
    public String join1(@RequestParam(value = "agree",defaultValue = "false")Boolean agree, Model
        if(!agree) {
            response.setContentType("text/html; charset=UTF-8");
            PrintWriter out=response.getWriter();
            out.println("<script>alert('약관동의체크는 필수사항입니다.');</script>");
            out.flush();
            return "/frontend/account/join1";
        }else {
            model.addAttribute("memberVo", new MemberVo());
            return "/frontend/account/join2";
        }
```

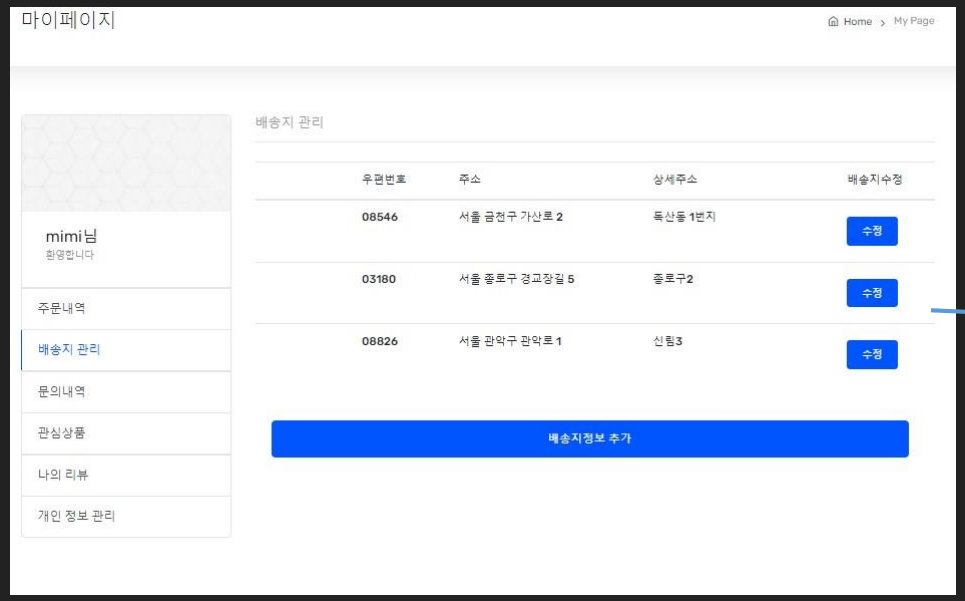
약관동의 체크 시에만 다음으로 이동하도록 구현

배송지 관리

mypage_plusDelivery.jsp

```jsp
<!-- 배송지 추가 -->
<div class="col-lg-9 col-md-8 order-md-2">
    <h6 class="text-muted text-lg text-uppercase">배송지 추가</h6>
    <hr class="margin-bottom-1x">

    <form action="${pageContext.request.contextPath }/account/delivery" method="post" name="adressForm">
    <div class="card-body">
        <div class="form-group input-group">

            <input class="form-control" type="text" id="sample2_postcode" name="zip_code" placeholder="우편번호">
            <input class="form-control" type="button" onclick="sample2_execDaumPostcode()" value="우편번호 찾기"><br>
        </div>
        <div class="form-group input-group">
            <input class="form-control" type="text" id="sample2_address" name="address" placeholder="주소"><br>
            <input class="form-control" type="text" id="sample2_detailAddress" name="adress_detail" placeholder="상세주소"><br>
            <input class="form-control" type="text" id="sample2_extraAddress" placeholder="참고항목"><br>
            <input class="btn btn-primary d-block" type="submit" value="배송지 등록">
        </div>
    </div>
    </form>

    <div id="layer" style="display:none;position:fixed;overflow:hidden;z-index:1;-webkit-overflow-scrolling:touch;">
        <img src="//t1.daumcdn.net/postcode/resource/images/close.png" id="btnCloseLayer" style="cursor:pointer;position:abso
    </div>
```

다음 주소 API를 사용하여 주소 등록 구현



mypage_delivery.jsp

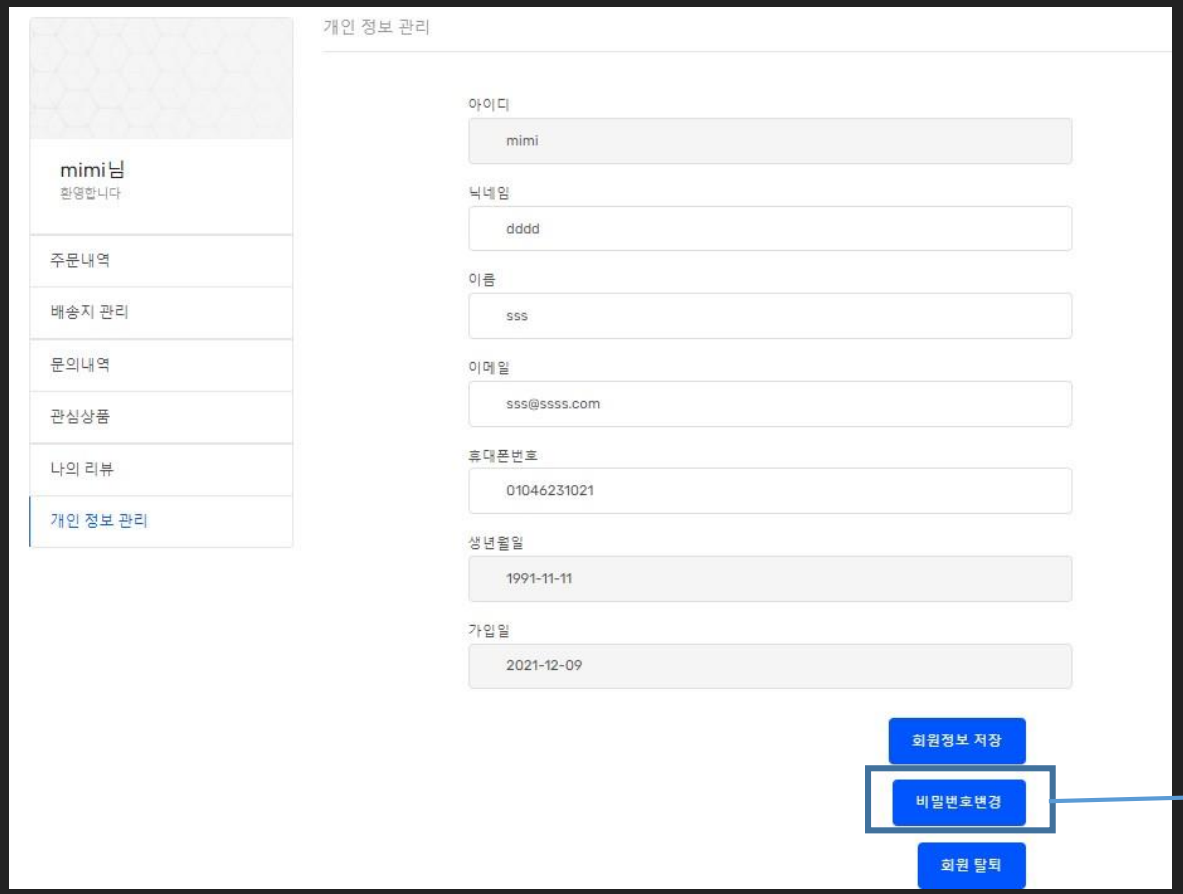
```jsp
<!-- 배송지 관리 -->
<div class="col-lg-9 col-md-8 order-md-2">
    <h6 class="text-muted text-lg text-uppercase">배송지 관리</h6>
    <hr class="margin-bottom-1x">

    <div class="table-responsive">
        <table class="table table-hover">
            <thead>
                <tr>
                    <th style="width:100px;"></th>
                    <th style="width:100px;">우편번호</th>
                    <th style="width:200px;">주소</th>
                    <th style="width:200px;">상세주소</th>
                    <th style="width:100px;">배송지수정</th>


                </tr>
            </thead>
            <tbody>
            <c:forEach var="vo" items="${list }">
                <tr>
                    <th></th>
                    <th>${vo.zip_code }</th>
                    <th>${vo.address }</th>
                    <th>${vo.adress_detail }</th>
                    <th><a href="${pageContext.request.contextPath }/pomgupdate?addr_no=${vo.addr_no}">
                    <input type="button" class="btn btn-primary btn-sm" value="수정">
                    </a></th>
                </tr>
            </c:forEach>
            </tbody>
        </table>
```

MyPageOtherController

```java
@RequestMapping(value = "/account/mypage_delivery")
public String frontendMyPageDelivery(Authentication auth, Model model) {

    CustomUserDetails cud = (CustomUserDetails) auth.getPrincipal();
    String id = cud.getUsername();
    System.out.println(id);
    List<AddressVo> list = adressService.addrList(id);
    model.addAttribute("list", list);
    System.out.println("리스트:" + list);
    return "frontend/account/mypage_delivery";
}

// 마이페이지 배송지추가
@GetMapping("/account/delivery")
public String PlusdeliveryForm(Model model) {
    return "frontend/account/mypage_plusDelivery";
}

// 배송지 추가,리스트 제출력
@PostMapping("/account/delivery")
public String Plusdelivery(AddressVo vo, Model model, Authentication auth) {
    System.out.println("auth:" + auth);
    System.out.println("우편추가시작");
    System.out.println("우편:" + vo.getZip_code() + "주소1:" + vo.getAddress() + "주소2:" + vo.getAdress_detail());

    CustomUserDetails cud = (CustomUserDetails) auth.getPrincipal();
    MemberVo mvo = cud.getMemberVo();
    String id = cud.getUsername();
    int mem_no = mvo.getMem_no();
    vo.setMem_no(mem_no);
    adressService.addrTest(vo);
    List<AddressVo> list = adressService.addrList(id);
    model.addAttribute("list", list);
    return "frontend/account/mypage_delivery";
}
```

CRUD 기능으로 구현

개인정보 관리



mypage_modify.jsp

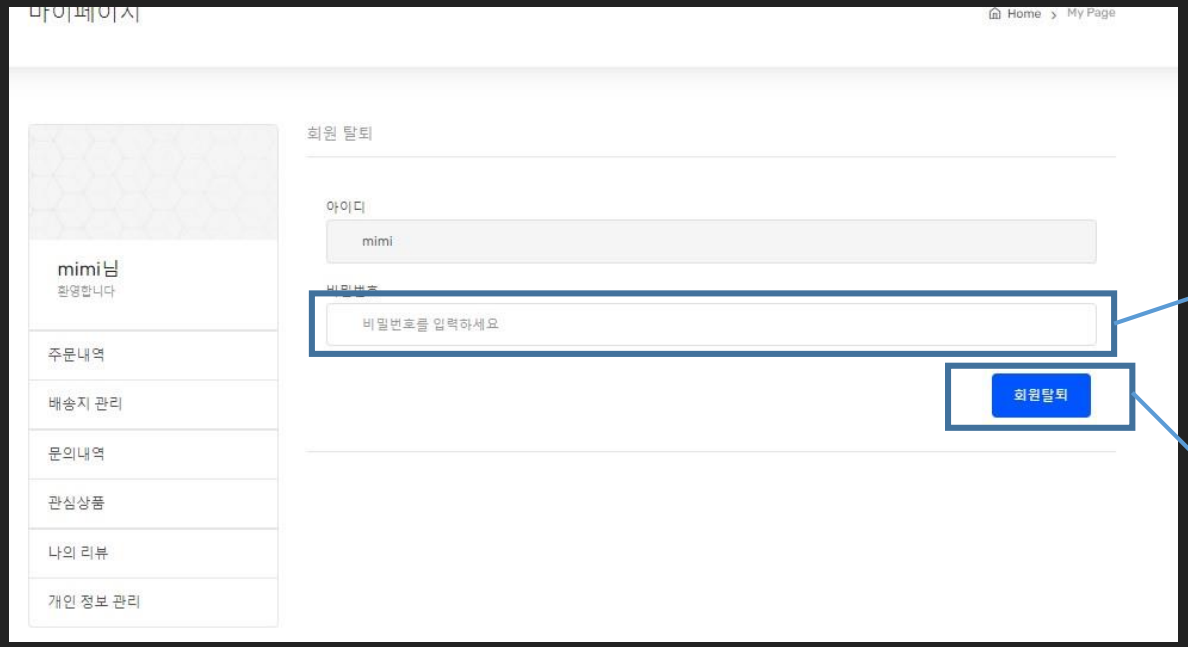
```jsp
<h6 class="text-muted text-lg text-uppercase">개인 정보 관리</h6>
<hr class="margin-bottom-1x">

<div class="card-body">
<form action="${pageContext.request.contextPath }/account/update" method="post">
<div class="form-group input-group">
    아이디
    <input style="cursor:default" class="form-control" type="text" name="id" readonly="readonly" value="${vo.id }">
</div>
<div class="form-group input-group">
    닉네임
    <input class="form-control" type="text" name="nickname" value="${vo.nickname }">
</div>
<div class="form-group input-group">
    이름
    <input class="form-control" type="text" name="name" value="${vo.name }">
</div>
<div class="form-group input-group">
    이메일
    <input class="form-control" type="text" name="email" value="${vo.email }">
</div>
<div class="form-group input-group">
    휴대폰번호
    <input class="form-control" type="text" name="phone" value="${vo.phone }">
</div>
<div class="form-group input-group">
    생년월일
    <input style="cursor:default" class="form-control" type="text" name="birthday" readonly="readonly" value="${vo.birth_date }">
</div>
<div class="form-group input-group">
    가입일
    <input style="cursor:default" class="form-control" type="text" name="regdate" readonly="readonly" value="${vo.reg_date }">
</div>


<div class="text-center text-sm-right">
    <button class="btn btn-primary margin-bottom-none" type="submit">회원정보 저장</button>
</div>
</form>
<form action="${pageContext.request.contextPath }/account/pwdmodify">
<div class="text-center text-sm-right">
    <input class="form-control" type="hidden" name="id" readonly="readonly" value="${vo.id }">
    <input class="form-control" type="hidden" name="password" readonly="readonly" value="${vo.password }">
    <button class="btn btn-primary margin-bottom-none" type="submit">비밀번호변경</button>
</div>
</form>
```

MyPageOtherController

```java
@RequestMapping(value = "/account/pwdmodify", method = RequestMethod.POST)
public String update_pw(@ModelAttribute MemberVo memberVo, String id, String old_pw, HttpSession session,
        HttpServletResponse response, RedirectAttributes rttr, Authentication authentication) throws Exception
    // String old_pw=bCryptPasswordEncoder.encode("old_pw");
    String opwd = memberService.selectpwd(id);

    System.out.println("old비번:" + old_pw);
    System.out.println("vo비번:" + opwd);
    // 기존비번 비교해야함

    if (bCryptPasswordEncoder.matches(old_pw, opwd)) {
        memberVo.setPassword(bCryptPasswordEncoder.encode(memberVo.getPassword()));

        MemberVo vo = new MemberVo(memberVo.getMem_no(), memberVo.getNickname(), memberVo.getEmail(),
                memberVo.getPhone(), memberVo.getBirth_date(), null, memberVo.getName(), memberVo.getId(),
                memberVo.getPassword(), memberVo.getRoles());
        session.setAttribute("member", memberService.updatePwd(vo));

        response.setContentType("text/html; charset=UTF-8");
        PrintWriter out = response.getWriter();
        out.println("<script>alert('비밀번호가 수정되었습니다.');</script>");
        out.flush();

        session.setAttribute("vo", memberService.selectid(id));

        return "frontend/account/mypage_modify";

    } else {
        response.setContentType("text/html; charset=UTF-8");
        PrintWriter out = response.getWriter();
        out.println("<script>");
        out.println("alert('기존 비밀번호가 틀립니다');");
        out.println("history.go(-1);");
        out.println("</script>");
        out.close();
        return null;
```

bCryptPasswordEncoder.matches를 VO에 비밀번호와 입력한비밀번호를 매칭 후 일치 여부 판단

회원 탈퇴



MyPageOtherController

```
@RequestMapping(value = "/account/memberDel", method = RequestMethod.POST)
public String memberDel(HttpSession session, MemberVo memberVo, Model model, String password,
        HttpServletRequest req) {
    // memberVo.setPassword(bCryptPasswordEncoder.encode(memberVo.getPassword()));
    // password=req.getParameter("password");
    String voPwd = bCryptPasswordEncoder.encode(memberVo.getPassword());
    System.out.println("voPwd: " + voPwd);
    boolean isMatches = bCryptPasswordEncoder.matches(password, voPwd);
    System.out.println("password: " + password);
    System.out.println("isMatches: " + isMatches);
    memberVo.getMem_no();
    if (isMatches == false) {
        return "frontend/account/mypage_memberDelete";
    } else {
        System.out.println(memberVo.getMem_no() + "+" + memberVo.getId());
        memberService.delete_role(memberVo.getMem_no());
        // memberService.memberDel(memberVo);
        session.invalidate();
        return "redirect:/";
    }
}
```

mypage_memberDelete.jsp

```
<div class="col-lg-9 col-md-8 order-md-2">
    <h6 class="text-muted text-lg text-uppercase">회원 탈퇴</h6>
    <hr class="margin-bottom-1x">

    <div class="card-body">
    <form action="${pageContext.request.contextPath }/account/memberDel" name="removefrm" method="post">
    <div class="form-group input-group">
        <input type="hidden" name="mem_no" value="${vo.mem_no }">
        아이디
        <input class="form-control" type="text" name="id" readonly="readonly" value="${vo.id }">
    </div>
    <div class="form-group input-group">
        비밀번호
        <input class="form-control" type="password" name="password" id="password"placeholder="비밀번호를 입력하세요">
    </div>
    <div class="text-center text-sm-right">
        <input class="btn btn-primary margin-bottom-none" onclick="removeCheck()" type="button" value="회원탈퇴">
    </div>
    </form>
</div>
<hr class="mt-2 mb-3">
</div>

div>
```

입력한 비밀번호와 VO에 담긴 비밀번호를 bCryptPasswordEncoder.matches로 매칭하여 true 이면 동의 얼라트창 출력후 진행 하도록 구현

테이블 구조

회원정보 테이블

번호	컬럼명	자료형	길이	NULL	의미	설명
1	Mem_no	Int			기본 키	멤버 테이블 기본키
2	Id	Varchar	40	Null	아이디	
3	Password	Varchar	255	Null	비밀번호	
4	Name	Varchar	20	Null	이름	
5	Nickname	Varchar	20	Null	닉네임	
6	Email	Varchar	255	Null	이메일	
7	Phone	Varchar	20	Null	전화번호	
8	Birth_date	Date		Null	생일	
9	Reg_date	date		null	가입일	

권한 테이블

번호	컬럼명	자료형	길이	NULL	의미	설명
1	Fno	Int			기본키	권한 테이블 기본키
2	roleName	Varchar	255	null	권한이름	유저,관리자 등을 지정하는 컬럼

주소 테이블

번호	컬럼명	자료형	길이	NULL	의미	설명
1	Addr_no	Int			기본 키	주소 테이블 기본키
2	Mem_no	Int			참조 키	멤버 테이블 참조 키
3	Zip_code	Varchar	30	not	우편번호	
4	Address	Varchar	255		주소	
5	Address_detail	Varchar	50	Not	상세 주소	
6	Address_name	Varchar	45	Not	주소 이름	

참조 테이블

번호	컬럼명	자료형	길이	NULL	의미	설명
1	Mem_no	Int			기본 키	멤버 테이블 참조 키
2	fno	int			기본 키	권한 테이블 참조 키