



개발자
FROG

TOYSPACE

1.



제품 리스트 페이지

TOY SPACE

PRODUCTS

ABOUT

Q 검색...



SHOP

ALL PRODUCTS

TOY

Filters

☐ 디즈니(9)

☐ 스타워즈(4)

☐ 왕좌의 게임(4)


☐ 포켓몬스터(8)

☐ 마블(4)

검색


제품수 (29)

Name: A-Z




디즈니
미키마우스
15000.0원

ADD TO CART




디즈니
모아나
9900.0원

ADD TO CART




스타워즈
요다
5900.0원

ADD TO CART




스타워즈
레이아공주
9900.0원

ADD TO CART




포켓몬스터
피카츄
8900.0원

ADD TO CART




포켓몬스터
고부기
8900.0원

ADD TO CART



포켓몬스터
이상해씨
8900.0원

ADD TO CART



마블
아이언맨
8900.0원

ADD TO CART

PREV

1





2

3

4

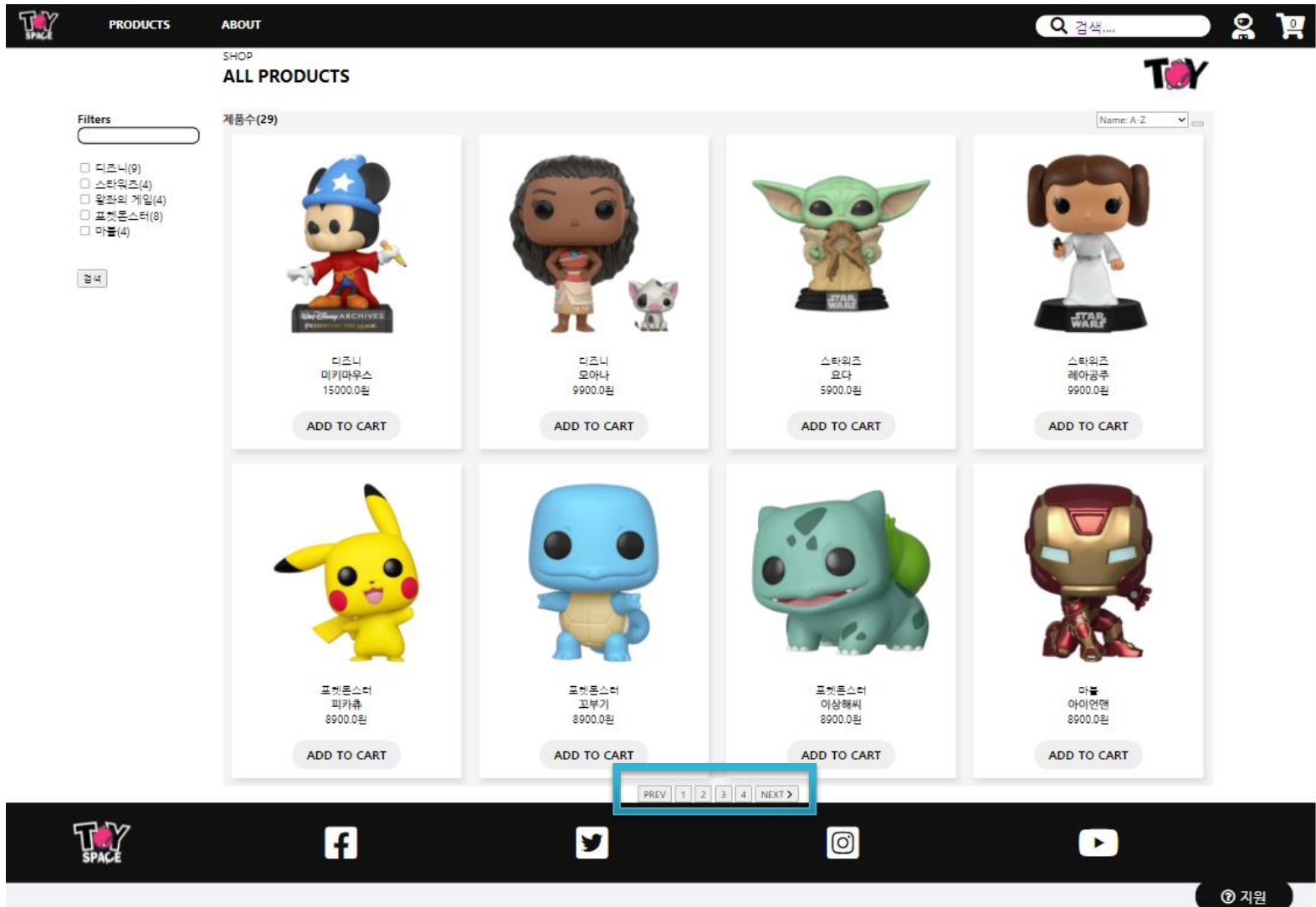
NEXT

TOY SPACE



② 지원

페이징 처리



제품 카테고리

PRODUCTS

ABOUT

DISNEY

STARWARS

왕좌의 게임

포켓몬

MARVEL

전체 상품

Filters

☐ 디즈니(9)

☐ 스타워즈(0)


☐ 왕좌의 게임(0)

☐ 포켓몬스터(0)

☐ 마블(0)


검색

제품수(9)




디즈니
라퐁젤
9900.0원


ADD TO CART



디즈니
몰란
13000.0원

ADD TO CART





관리자 계정 관리



관리자 페이지

주문관리

상품등록

상품관리

매출현황

태그등록

채팅관

:관리자 계정 관리:

관리자 회원 목록

	아이디	비밀번호	NICKNAME	이름	LEVLE	수정 / 삭제
<input type="checkbox"/>	admin	*****	마스터	마스터	0	수정 삭제
<input type="checkbox"/>	admin1	*****	관리자	관리자	1	수정 삭제
<input type="checkbox"/>	admin2	*****	부관리자	부관리자	2	수정 삭제
<input type="checkbox"/>	test	****	2	2	2	수정 삭제

선택 삭제

관리자 계정 생성

아이디	비밀번호	NICKNAME	이름	LEVEL	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	추가

계정 선택삭제, 수정/삭제

:관리자 계정 관리:

관리자 회원 목록					
<input type="checkbox"/>	아이디	비밀번호	NICKNAME	이름	LEVLE
<input type="checkbox"/>	admin	*****	마스터	마스터	0
<input type="checkbox"/>	admin1	*****	관리자	관리자	1
<input type="checkbox"/>	admin2	*****	부관리자	부관리자	2
<input type="checkbox"/>	test	****	2	2	2

선택 삭제

수정 / 삭제
수정 | 삭제
수정 | 삭제
수정 | 삭제

➤ 선택삭제

- 1개 이상의 체크박스를 선택 후 선택 삭제 버튼을 클릭 시 선택된 다수의 해당 계정이 삭제

➤ 수정/삭제

- 수정 : 해당 라인의 변경하고 싶은 정보를 타이핑 한 후 클릭 시 해당 정보 수정
- 삭제 : 클릭 시 해당 라인의 계정 삭제

관리자 계정 관리

관리자 계정 생성

아이디	비밀번호	NICKNAME	이름	LEVEL	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="추가"/>

- 관리자 계정 추가
 - 추가 하고자 하는 계정의 정보 입력 후 추가 버튼을 클릭 하면 계정이 생성 된다 해당 모든 정보가 입력되어야 한다.

주소 검색 API



이메일

배송지 주소

이름

성

전화번호

- -

06234

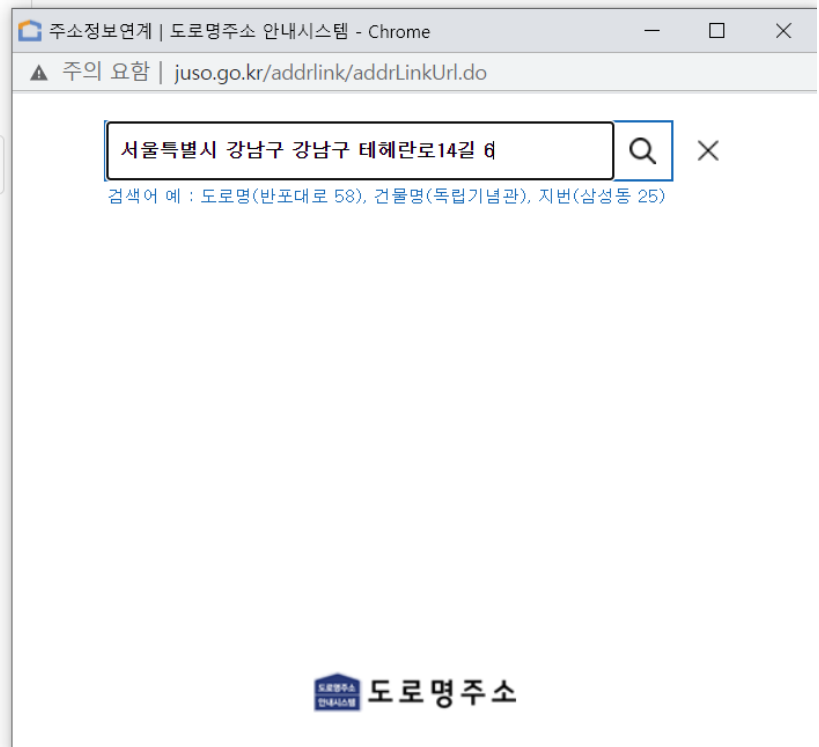
주소검색

서울특별시 강남구 테헤란로14길 6

kh정보교육원

(역삼동)

결제



주소 검색 API

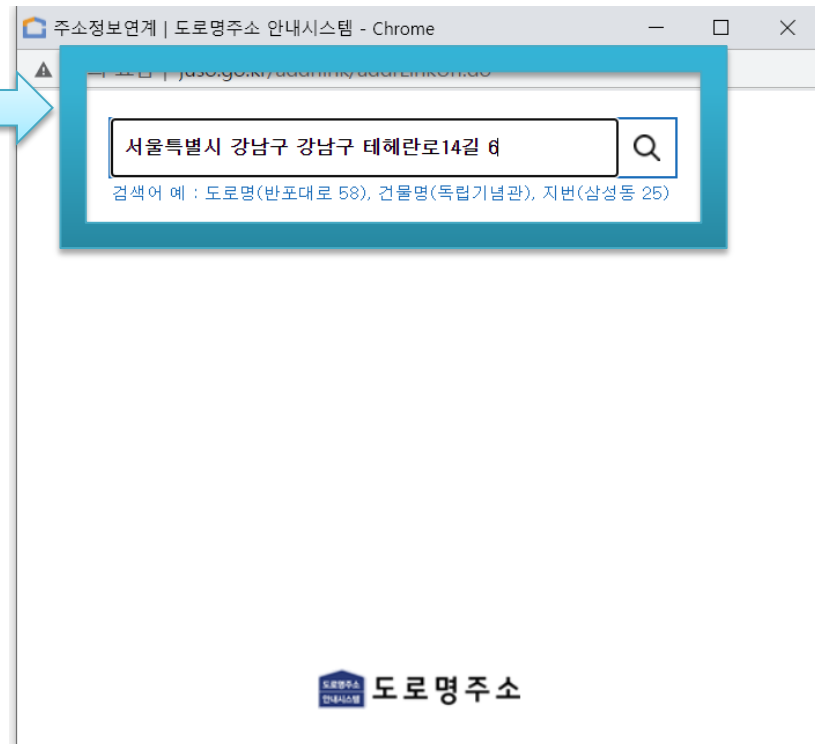
06234

주소검색

서울특별시 강남구 테헤란로14길 6

kh정보교육원

(역삼동)



- 주소검색
 - 버튼을 클릭 하면 검색 팝업 창이 생성 되며 검색된 데이터가 연결된 텍스트 박스로 이동 된다

지도 API

위치



TOY SPACE

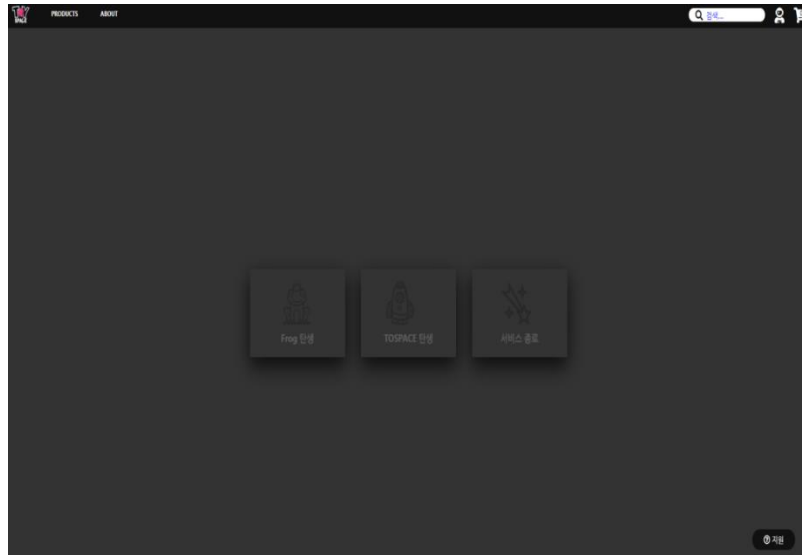
- API 를 통한 원하는 위치 구현
- 원하는 좌표에 회사 소개



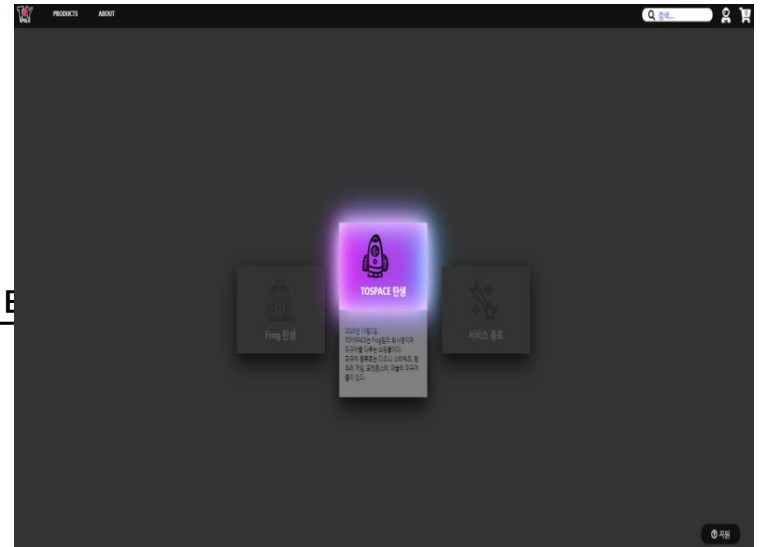
TOYSPACE

2.

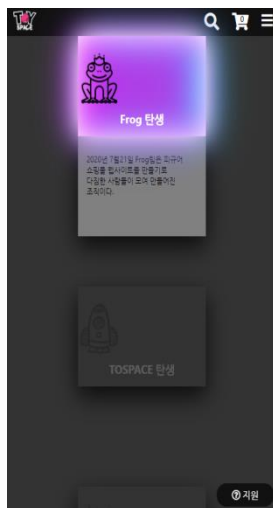
회사 소개



호버 이벤트



반응형



```
@media (max-width: 992px){  
  .container  
  {  
    width: 100%;  
    flex-direction: column;  
    align-items: center;  
  }  
  .container .card  
  {  
    width: 400px;  
  }  
}
```

채용



호버 이벤트



지원

지원

반응형



```
@media (max-width: 992px){  
  .container  
  {  
    width: 100%;  
    flex-direction: column;  
    align-items: center;  
  }  
  .container .card  
  {  
    width: 400px;  
  }  
}
```

ASSISTANCE



TOYSPACE OPEN !!

반품 정책

TOYSPACE 구매를 반품하고 싶다면 저희가 도와드리겠습니다. 주문을 받은 후 5 일 이내에 TOYSPACE 상품을 반품 할 수 있으며 환불에서 배송비를 공제합니다. 원래의 상자과 포장에 있는 항목을 반환하십시오.

실시간 채팅 문의를 통해 반품 관련 글을 적으신 후 10일 이내에 상품을 반송해야 합니다. 글을 적은 후 10일이 지난 상품을 보내면 환불에서 20% 재입고 수수료가 공제됩니다. 환불은 원래 결제 방법으로 상품을 수령 한 후 영업일 기준 2일 이내에 처리됩니다.

다음은 위의 반품 정책에 대한 예외입니다.

- 상품에 대한 우발적 파손에 대하여 책임을 지지 않습니다.
- 상품의 리셀 제거시



TOYSPACE OPEN !!

교환 정책

당사 사이트에서 구매 한 모든 상품은 타사 운송 업체에서 배송됩니다. 결과적으로, 그러한 상품에 대한 소유권 및 손실 또는 손상 위험은 당사가 운송 업체에 배송하는 즉시 귀하에게 이전됩니다. 모든 누락, 손상 및 잘못된 배송 또는 항목은 수령 후 15일 이내에 보고해야 하며 이메일 제출시 사진 문서를 포함해야 합니다. 이러한 항목은 지원 담당자가 검토하며 해당되는 경우 손상된 제품에 대해서만 교체가 발행되며 해당 제품의 포장을 덮거나 포함하지 않습니다. 포장 오류는 손상된 품목의 일부로 간주되지 않습니다. 모든 교체는 고객 지원의 재량에 따릅니다. 다음 교체 제한 사항에 유의하십시오.

우리는 사소한 제조 변형을 통제 할 수 없으며 사소한 페인트 결함이나 그림 위치를 포함하되 이에 국한되지 않는 이러한 변형을 기반으로 항목을 교체 할 수 없습니다. 포장의 출장은 손상된 그림의 일부로 간주되지 않습니다.

제품 포장

우리는 내부의 수집품을 보호하기 위해 포장을 설계하며, 바로 이러한 이유로 상자를 밀봉하지 않습니다. 당시의 Pop! 피규어 및 기타 제품을 상자에서 꺼내서 즐기십시오! 상자에 그림을 보관하도록 선택하더라도 포장을 교체 할 수 없습니다.

거부되거나 배송되지 않는 패키지

구매자 오류 또는 배송 거부로 인해 반품 패키지에는 구매가격의 최대 20% 까지 재입고 수수료가 부과됩니다. 고객은 UPS에 연락하여 대체 배송을 준비 할 수 있습니다. TOYSPACE는 이 대체 배송과 관련된 수수료에 대해 책임을 지지 않습니다.

매진

보관 된 품목에 대한 교체 재고를 항상 사용할 수 있는 것은 아니므로 매진되었거나 오래된 보관 된 품목에 대한 교체 부품이 있을 것이라고 보장 할 수 없습니다. 허용되는 교체품이 발견 된 경우 교체 부품을 보내 드리기 위해 배송비가 부과 될 수 있습니다.

주문에 대한 도움이 필요하면 ? 지원 버튼을 눌러 실시간 채팅 문의로 TOYSPACE에 문의하십시오.



TOYSPACE OPEN !!

취소 정책

모든 판매는 최종적이며 주문이 제출 된 후에는 취소 할 수 없습니다.

주문을 제출하기 전에 구매를 검토하여 형구 / 배송, 주문 수량이 올바른지 확인하십시오. 주문이 확인 된 후에는 수량을 업데이트하거나 항목을 추가 또는 제거하거나 배송지 주소를 업데이트 할 수 없습니다.

수량 제한

판과 수집가에게 독점 또는 비 독점 품목을 구매할 수있는 최상의 기회를 제공하기 위해 또는 기타 이유로 당사는 수량을 제한 할 권리를 보유합니다.

품목에 대한 한도를 초과하는 것으로 확인 된 주문은 주문 후 취소 될 수 있습니다. 주문이 취소되면 당사에서 미결제 요금이 표시 될 수 있으며 영업일 기준 3 ~ 7 일 이내에 취소되지만 이 기간은 금융 기관에 따라 다를 수 있습니다.

여러 이메일 주소를 사용하거나, 배송 주소를 조작하거나, 동일한 제품을 여러 번 과도하게 주문하거나, 다른 방법으로 계층 계층을 우회하려는 경우 모든 주문이 취소되거나 향후 구매가 차단 될 수 있습니다.

대량 구매

TOYSPACE는 대량 구매에 적용되는 주문 및 구매 조건을 제한, 제한, 취소 또는 변경할 권리를 보유합니다. '대량 구매'라는 용어는 재판매 및 / 또는 동일한 품목을 6 개 이상 구매하기위한 구매로 정의됩니다.

대량 구매시 배송비가 적용될 수 있습니다.

주문에 대한 도움이 필요하면 실시간 채팅문의를 통해 TOYSPACE에 문의하십시오.

디자인



TOYSPACE 홈페이지
회원가입이 완료 되었습니다.



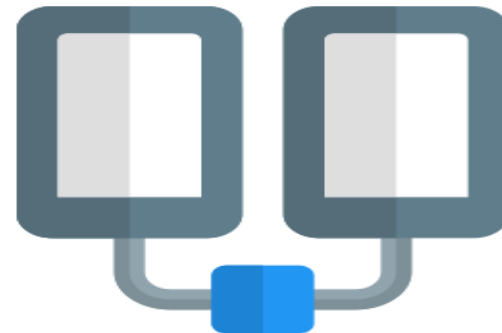
TOYSPACE 홈페이지
간편로그인으로 회원가입이 완료 되었습니다.

추가 정보를 입력하시겠습니까?

확인

취소

🔗 지원



이메일이 존재합니다.
연동하시겠습니까?

아이디 :

연동

회원가입 취소



TOYSPACE

3.

상품관리

상품관리 매출현황 태그등록 채팅관리 관리자

상품관리

상품 조회:

선택 삭제

상품명	상품가격	재고	수정
라쿤젤	9900.0원	201	<input type="button" value="상품 수정하기"/>
파이리	10000.0원	100	<input type="button" value="상품 수정하기"/>
물란	13000.0원	500	<input type="button" value="상품 수정하기"/>

Window.open()

```

</thead>
<tbody>
<button type="submit" onclick="return confirmRemoving();" >선택 삭제</button>
<%if(productsList!=null&&productsList.size()!=0){
    for(Product p : productsList){ %>
<tr>
<td><input type="checkbox" value="<%=p.getProductid()%>" name="productIds-removed"/></td>
<td><%=p.getProductid() %></td>
<td><%=p.getCategoryName() %></td>
<td><%=p.getProductname() %></td>
<td><%=p.getProductPrice() %>원</td>
<td><%=p.getProductStock() %></td>
<td>
<input
type="button"
onclick="window.open('<%=request.getContextPath() %>/admin/editProduct?productId=<%=p.getProductid()%>'
value="상품 수정하기"
/>

```

ProductId 값
(unique) 을 파라
미터로 함께 전송

상품수정

☆☆☆☆☆

상품ID

21

상품명

라쿤젤

상품 카테고리

디즈니

상품 가격

9900.0

상품 개수

201

태그 번호

태그명

7

왕자

18

남자

41

디즈니

42

공주

43

마법

44

여자

46

로봇몬스터

상품 태그 목록

41

42

43

44

상품 태그 번호

41

42

43

44

상업용*에*제품*이*가*있*다*.*

가*장*가*가*장*10*

상품 설명

제조사

(주)아이스페이스

제조국

프랑스

권장 연령

3

상품을 등록시 상품정보가 올바르게 입력되었는지 확인하십시오.

상품 등록에 실패하는 사유를 알립니다.

상품 등록에 실패하는 사유를 알립니다.

상품이 등록되었는지 확인하십시오.

상품 등록에 실패하는 사유를 알립니다.

주의사항

수정

취소

회원관리



```
<div class="search-search-container">
회원조회:
<select class="searchType">
  <option value="userId" <%=type!=null&&type.equals("userId")?"selected":"" %>>아이디</>
  <option value="userName" <%=type!=null&&type.equals("userName")?"selected":"" %>>회원명</>
</select>
<div class="search-userId">
  <form action="<%=request.getContextPath() %>/admin/memberSearch">
    <input type="hidden" name="searchType" value="user_id">
    <input type="text" name="searchKeyword" placeholder="아이디검색" size="25"
      value="<%=key!=null&&type!=null&&type.equals("user_id")?key:""%>">
    <button type="submit">검색</button>
  </form>
</div>
<div class="search-userName">
  <form action="<%=request.getContextPath() %>/admin/memberSearch">
    <input type="hidden" name="searchType" value="user_name">
    <input type="text" name="searchKeyword" placeholder="이름검색" size="25"
      value="<%=key!=null&&type!=null&&type.equals("user_name")?key:""%>">
    <button type="submit">검색</button>
  </form>
</div>
</div>
```

```
<script>
$(function(){
    let userId=$("#.search-userId");
    let userName=$("#.search-userName");

    $(".searchType").change(e => {
        userId.css("display", "none");
        userName.css("display", "none");
        let v=$(e.target).val();
        $(".search-"+v).css("display", "inline-block");
    });
    $(".searchType").change();
});
</script>
```

태그관리

Form 태그 이용하여 데이터
전송

태그관리

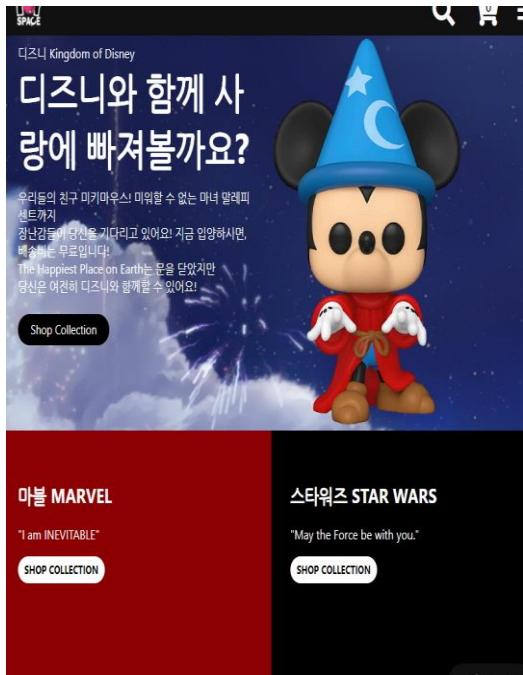
태그 조회:

태그No.	태그명	삭제하기	수정하기
7	<input type="text" value="올자"/>	<input type="button" value="삭제"/>	<input type="button" value="수정"/>
12	<input type="text" value="나쁜"/>	<input type="button" value="삭제"/>	<input type="button" value="수정"/>

태그 조회:

```
<div class="search-tagName">
  <form action="<%=request.getContextPath() %>/admin/searchTag">
    <input
      type="text"
      name="keyword"
      placeholder="태그명을 입력해주세요"
      size="25"
    />
    <button type="submit">검색</button>
  </form>
</div>
</div>
<hr />
<form action="<%=request.getContextPath() %>/admin/insertTag" method="post"
<input type="text" class="tag-add" name="newTag" required />
<button type="submit">태그등록</button>
</form>
<hr />
```

반응형 디자인



@media screen & flex-wrap 이용

리뷰

리뷰 남기기



리뷰

리뷰 남기기

"google,1602398828919"님 (google,1602398828919) 2020. 10. 14

3433

```

}
@media screen and (max-width: 768px) {
  div.main-container {
    flex-flow: column wrap;
    margin: 0;
  }
  div.text-container {
    max-width: 100%;
  }
  div.main2 {
    flex-direction: column;
    margin: 0;
    max-width: 100%;
  }
}
@media screen and (max-width: 768px) {
  div.main3-pic-text {
    flex-wrap: wrap;
  }
}
    
```

상품상

이미지 슬라이드

```
<div class="mySlides fade">
  <div class="numbertext">2 / 2</div>
  <% if(p.getProductImageFilePaths().get(1)!=null){ %>
    &#10094;</a>
  <a class="next" onclick="plusSlides(1)">&#10095;</a>
</div>
<br />
<div style="text-align: center">
  <span class="dot" onclick="currentSlide(1)"></span>
  <span class="dot" onclick="currentSlide(2)"></span>
</div>
```

```
// 이미지 슬라이드
let slideIndex = 1;
showSlides(slideIndex);

function plusSlides(n) {
  showSlides((slideIndex += n));
}

function currentSlide(n) {
  showSlides((slideIndex = n));
}

function showSlides(n) {
  let i;
  let slides = document.getElementsByClassName("mySlides");
  let dots = document.getElementsByClassName("dot");
  if (n > slides.length) {
    slideIndex = 1;
  }
  if (n < 1) {
    slideIndex = slides.length;
  }
  for (i = 0; i < slides.length; i++) {
    slides[i].style.display = "none";
  }
  for (i = 0; i < dots.length; i++) {
    dots[i].className = dots[i].className.replace(" active", "");
  }
  slides[slideIndex - 1].style.display = "block";
  dots[slideIndex - 1].className += " active";
}
```



상품정보

- 상품번호:21
- 상품명:라퐁젤
- 상품:라퐁젤:라퐁젤
- 사용연월:3세 이상
- 제조자:(주)트이스페이스
- 제조국:프랑스
- 사이즈:약 10.8cm
- 취급 시 주의사항
 - ⦿ 표창을 반드시 보호자가 개봉하십시오.
 - 계를 잘못 이용하여는 사용하지 마십시오.
 - 만 3 세 미만의 어린이가 사용하기에는 부적합합니다.
 - 건조기 혹은 뜨거운 바람에 말리지 마십시오.
 - 물에 젖을 경우 그늘에서 말리십시오.

리뷰

리뷰 남기기

코드리뷰

상품,태그,회원 같은 데이터들을 수정/삭제할때 일괄적으로 체크박스를 사용하여 전체삭제/선택삭제로 구현했으면 좀 더 실용적으로 보였을 것

이미지 슬라이더, 추후에 검색해보니 Swiper, bxslider, slick 같은 라이브러리를 사용한다면 좀 더 여러방면으로 기능추가 하여 사용하지 않았을까 함.
(물론, 스크립트,jQuery 기본을 다지는게 우선이긴 함)

댓글 수정/삭제 대댓글 구현을 마감에 임박해 하다보니 결국 댓글 쓰기만 구현함

반응형디자인, 때려맞추기 식으로 이 코드 저코드 쓰면서 불필요한 코드도 들어간 듯한 느낌이 많이 들었다. 속성값을 제대로 알고 불필요한 코드를 제거할 필요가 있다.



TOYSPACE

4.

회원가입(유3)

```
$("#submitbtn").on("click",function(){
    var pw = $("#userPassword").val();
    var name = $("#userName").val();
    var nickname = $("#userNickname").val();
    var email = $("#userEmail").val();

    var pwregex = /^[A-Za-z\d]{8,12}$/;
    var nameregex = /[가-힣]{2,}/;
    var nicknameregex = /[가-힣]{2,}/;
    var emailregex = /.+@[a-z](\.[a-z]){1,2}$/;

    var pwregex = pwregex.exec(pw);
    if(pwregex == null){
        alert("비밀번호양식을 다시 확인해주세요");
        retrun;
    }
    var nameregex = nameregex.exec(name);
    if(nameregex == null){
        alert("이름양식을 다시 확인해주세요");
        retrun;
    }
    var nicknameregex = nicknameregex.exec(nickname);
    if(nicknameregex == null){
        alert("별명양식을 다시 확인해주세요");
        retrun;
    }
    var emailregex = emailregex.exec(email);
    if(emailregex == null){
        alert("이메일양식을 다시 확인해주세요");
        retrun;
    }
}
```



SIGN UP

아이디

비밀번호

비밀번호확인

이름

별명

이메일

가입하기

계속하면 TOYSPACE에 동의하는 것으로 간주합니다. [이용약관](#) 그리고 [개인정보정책](#)
이미 계정이 있으신가요? [로그인](#)


```

//id pw name 닉네임 이메일
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    // TODO Auto-generated method stub
    String userId=request.getParameter("userId");
    String userPassword =request.getParameter("userPassword");
    String userName =request.getParameter("userName");
    String userNickname =request.getParameter("userNickname");
    String userEmail = request.getParameter("userEmail");

    String loc = "";
    String msg = "";

    Member m = new Member(0, userId,userEmail,userPassword,"X", userName, userNickname, 0, null, null, null, null, null,

    int result = new MemberService().insertMember(m);
    if(result!=1) {
        msg = "회원가입에 실패하였습니다.";
        loc =request.getContextPath();
    } else {
        msg = "회원가입에 성공하였습니다!";
        loc = request.getContextPath()+"/member/signUpCelebrate";
    }
    request.getRequestDispatcher("/msg?loc="+loc+"&msg="+msg).forward(request, response);
}

```

로그인



LOG IN

아이디를 입력해주세요.

비밀번호를 입력해주세요.

비밀번호를 잊으셨나요?

로그인

ToySpace는 처음이신가요? [지금 가입하기](#)

```
*/
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    // TODO Auto-generated method stub
    String userId= request.getParameter("id");
    String userPw= request.getParameter("pw");

    Member member = new MemberService().loadMembers(userId,userPw);

    String msg;
    String loc;
    String path;
    System.out.println("실행중");
    if(member!=null) {
        request.getSession().setAttribute("signedInMember", member);
        loc=request.getContextPath();
        msg="로그인 성공";
    } else {
        msg="로그인 실패";
        loc=request.getContextPath()+"/login.do";
    }
    request.setAttribute("msg", msg);
    request.setAttribute("loc", loc);
    path="/views/common/msg.jsp";
    request.getRequestDispatcher(path).forward(request, response);
}
```

이메일인증



TOYSPACE 계정 찾기

비밀번호 재설정이 필요하신가요? 이메일주소를 입력해주세요.

비밀번호찾기

TOYSPACE는 처음이신가요? [가입하기](#)

```
//받는 사람의 정보
String toName = request.getParameter("username");
String toEmail = request.getParameter("email");

//보내는 사람의 정보
String fromName = "관리자";
String fromEmail = "01088049128a@gmail.com";

try {
    Properties props = new Properties();
    props.put("mail.smtp.user", fromEmail);
    props.put("mail.smtp.host", "smtp.googlemail.com");
    props.put("mail.smtp.port", "465");
    props.put("mail.smtp.starttls.enable", "true");
    props.put("mail.smtp.auth", "true");
    props.put("mail.smtp.debug", "true");
    props.put("mail.smtp.socketFactory.port", "465");
    props.put("mail.smtp.socketFactory.class", "javax.net.ssl.SSLSocketFactory");
    props.put("mail.smtp.socketFactory.fallback", "false");

    // 메일 인증
    Authenticator myauth=new MyAuthentication();
    Session sess=Session.getInstance(props, myauth);

    InetAddress addr = new InetAddress();
    addr.setPersonal(fromName, "UTF-8");
    addr.setAddress(fromEmail);

    Message msg = new MimeMessage(sess);
    msg.setFrom(addr);
```



TOYSPACE

5.

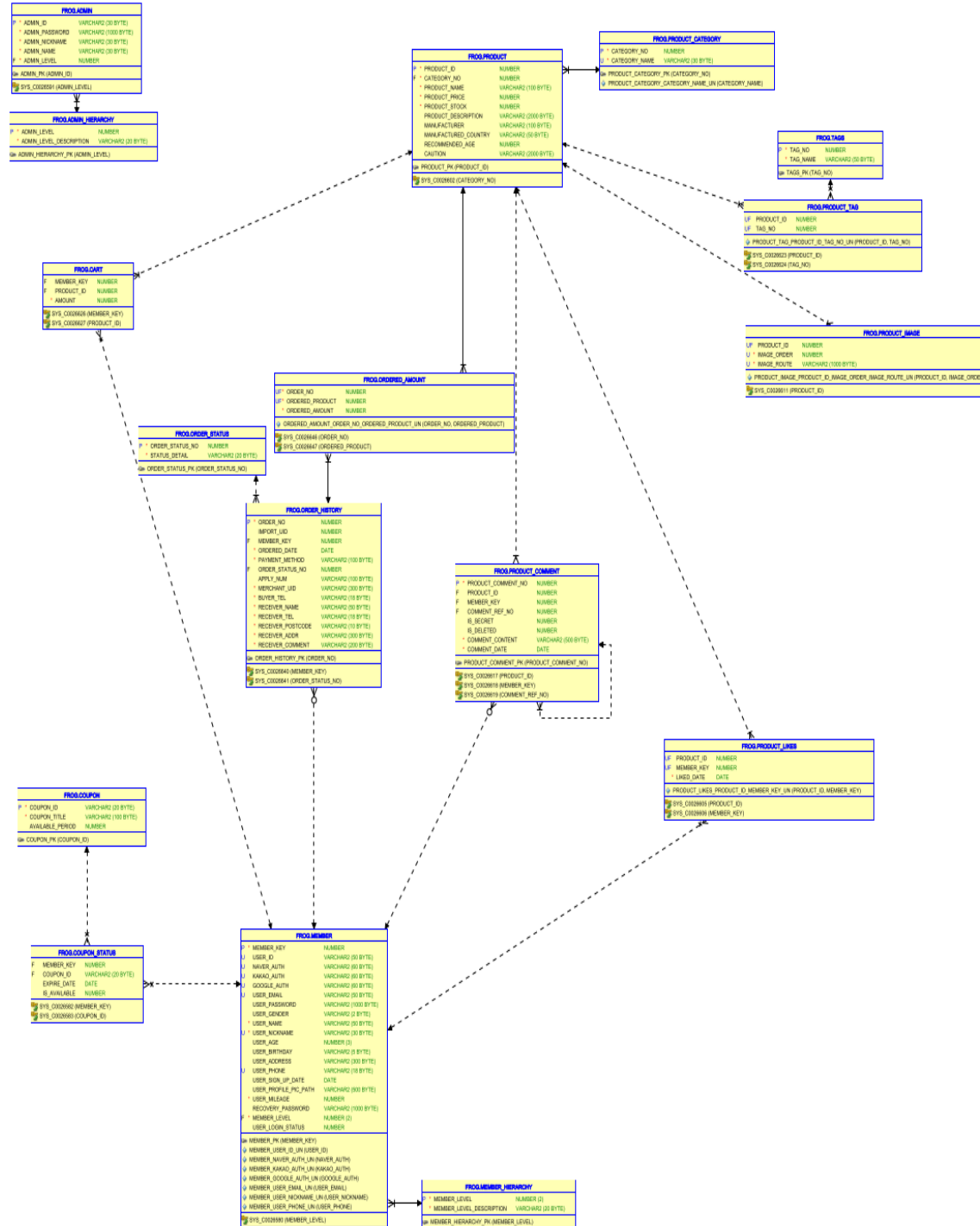
맡은 역할

로고 디자인, DB설계,
3대 간편 로그인/가입(구글/카카오/네이버),
장바구니, 결제(아임포트),
소비자용 상품 검색 기능(with 경록, 세민),
커스토머 용 헤더 푸터 구성,
규약 관리(css/js/java 공유 코드, 페이징 객체 등),
디버깅, 호환성 등

-완전 구현 실패 목록

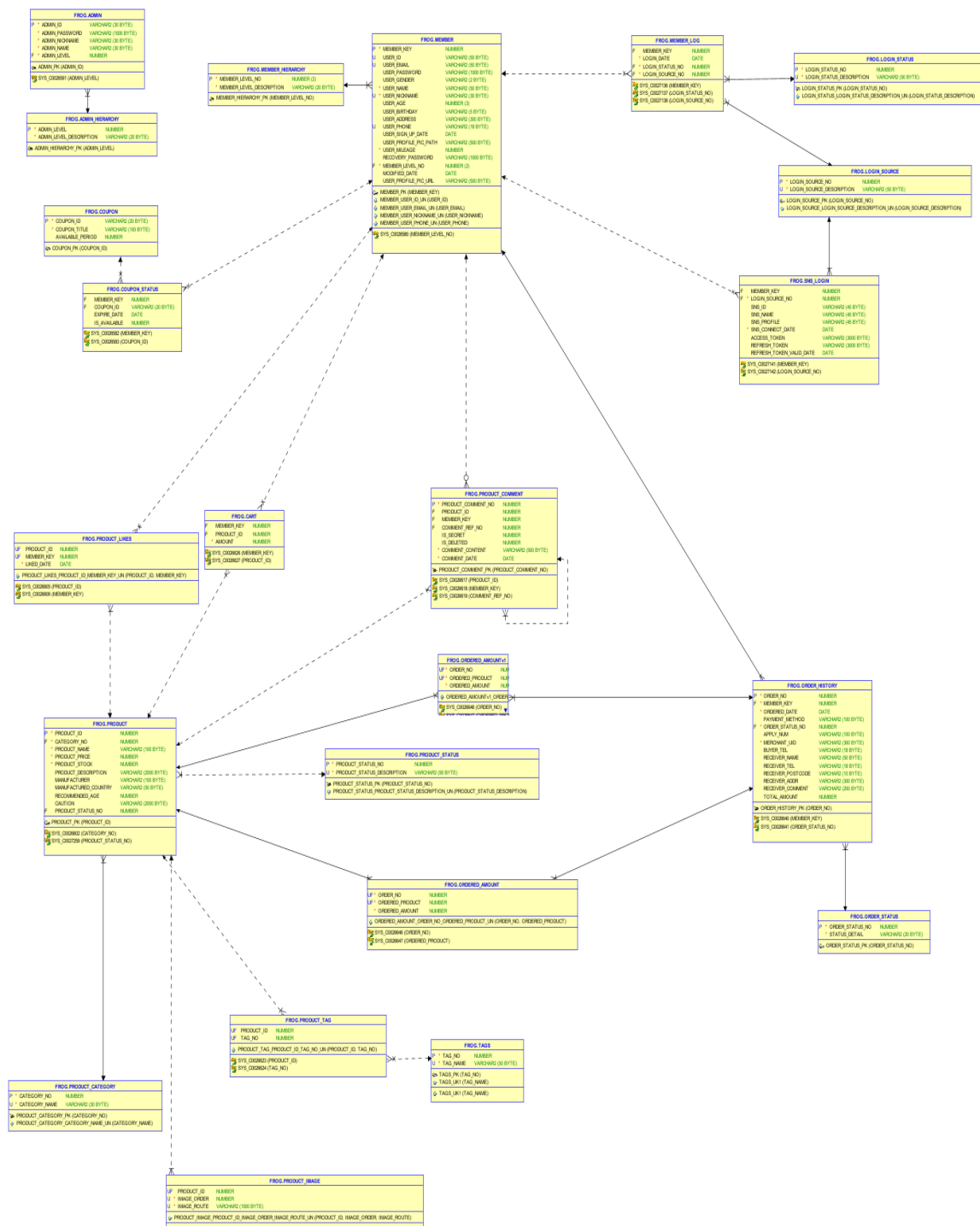
소비자용 결제 목록(관리자 용은 구현), 채팅

원안



11/11/2019

1. 단일 행으로 처리 될 수 있는 데이터들을 위한 테이블과 다중 행이어야 하는 테이블 들을 나눔
2. INSERT시 부적절한 데이터가 들어가는 것을 막기 위해 STATUS 등을 숫자 형으로 표현하고, 그것을 설명하는 별개의 테이블을 둬



```

// TODO Auto-generated constructor stub
}

public boolean insertProduct(Product insertion, List<Integer> tagsList) {
    boolean result = false;

    Connection conn = getConnection();
    int productId = dao.nextSeq(conn);
    if(productId == -1) {
        rollback(conn);
        close(conn);
        return result;
    }
    result = dao.insertProduct(conn, productId, insertion);
    if(!result) {
        rollback(conn);
        close(conn);
        return false;
    }

    result = dao.insertFilePathsIntoProduct(conn, productId, insertion.getPr
    if(!result) {
        rollback(conn);
        close(conn);
        return false;
    }

    result = dao.insertTagsIntoProduct(conn, productId, tagsList);
    if(result) commit(conn);
    else rollback(conn);
    close(conn);
    |
    return result;
}

```

```

}
//상품수정
public boolean updateProduct(Product p, String[] tagsArr) {
    Connection conn=getConnection();
    int productId = p.getProductId();
    boolean result=dao.updateProduct(conn,p);
    if(!result) {
        rollback(conn);
        System.out.println("상품 업데이트 실패!");
        close(conn);
        return result;
    }
    result = dao.removeProductTags(conn, productId);
    if(!result) {
        rollback(conn);
        System.out.println("상품의 태그들 삭제 실패!");
        close(conn);
        return result;
    }
    result = dao.updateProductTags(conn, productId,tagsArr);
    if(!result) {
        rollback(conn);
        System.out.println("상품의 태그들 업데이트 실패!");
        close(conn);
        return result;
    }
    commit(conn);
    close(conn);
    return result;
}

```

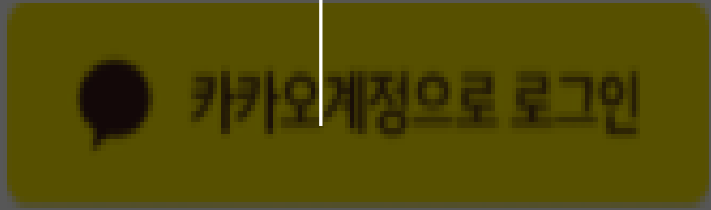
단점

- SQL문이 복잡해진다.
- 하나의 기능을 위해 여러 번의 INSERT/UPDATE/DELETE가 필요하다.

loyspace는 처음이신가요? [지금 가입하기](#)

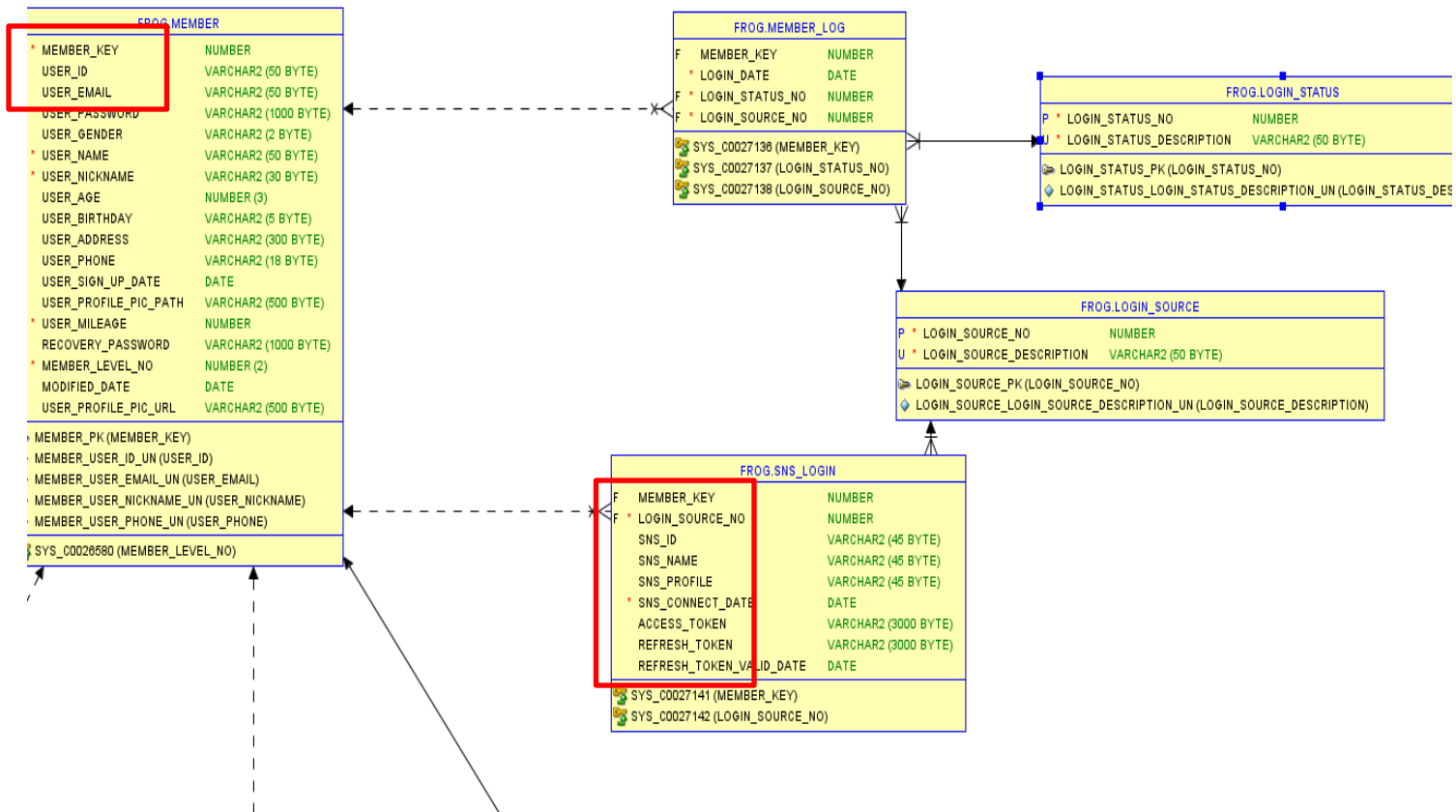


간편로그인



이제는 거의 필수가 되어버린
간편 로그인

간편로그인을 위한 DB 구조



간편 로그인 코드

- 구글을 로그 아웃을 구현해야함
- 네이버는 redirect된 곳에서 로그인 처리
- REST API

```
<script>
function onSignIn(googleUser) {
  let id_token = googleUser.getAuthResponse().id_token;
  let xhr = new XMLHttpRequest();
  xhr.open('POST', 'http://mightymosses.hopto.org:9090/project_frog_01/member/googleSignIn');
  xhr.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded');
  xhr.send('id_token=' + id_token + '&client_id=<%=clientId%>');
  /* 어디로 보낼지 설정하기 */
  /* 로그인 객체는 세션에 담아짐 */
  xhr.addEventListener('load', function(e){
    const result = JSON.parse(e.target.response);
    switch(result){
      case 3: alert("기존에 가입된 아이디를 발견하였습니다."); location.href = "<%=request.getContextPath()%>/member/mergeId"; break;
      /* 간편가입시 '추가 정보 기입하시겠습니까?' 등을 물어보는 서블릿으로 이동 */
      case 2: alert("기존에 가입된 이력이 없습니다. 간편 가입되었습니다."); location.href = "<%=request.getContextPath()%>/member/";
      case 1: alert("구글을 통해 간편로그인을 하셨습니다.");
        location.href = "<%=request.getContextPath()%>";
        break;
    }
  })
}

const naverLogin = new naver.LoginWithNaverId(
{
  clientId: "ZsYfto7388DFNUATk2ze",
  callbackUrl: "http://mightymosses.hopto.org:9090/project_frog_01/views/login/sns_login/redirect/naverRedirect.jsp",
  isPopup: false, /* 팝업을 통한 연동처리 여부 */
  loginButton: {color: "green", type: 3, height: 36} /* 로그인 버튼의 타입을 지정 */
});

/* 설정정보를 초기화하고 연동을 준비 */
naverLogin.init();

/* 카카오 인증 초기화 */
Kakao.init('9bf121caabaa1ff4de5c5d96dfa03136');
Kakao.Auth.createLoginButton({
  container: '.kakao-container',
  success: function (authObj) {
    Kakao.API.request({
      url: '/v2/user/me',
      success: function(res) {

$.ajax({
  url: '<%=request.getContextPath()%>/member/kakaoSignIn',
  data: {"id" : res.id, "access_token": Kakao.Auth.getAccessToken(), "birthday" : res.kakao_account.birthday, "email" : res.kakao_account.email, "nickname" : res.kakao_account.profile.nickname, "profile_image_url": res.kakao_account.profile.profile_image_url},
  type: 'POST',
  success : (data)=>{
    const result = JSON.parse(data);
    switch(result){
      case 3: alert("기존에 가입된 아이디를 발견하였습니다."); location.href = "<%=request.getContextPath()%>/member/mergeId"; break;
      /* 간편가입시 '추가 정보 기입하시겠습니까?' 등을 물어보는 서블릿으로 이동 */
      case 2: alert("기존에 가입된 이력이 없습니다. 간편 가입되었습니다."); location.href = "<%=request.getContextPath()%>"; break;
      case 1: alert("카카오를 통해 간편로그인을 하셨습니다.");
        location.href = "<%=request.getContextPath()%>";
        break;
    }
  }
});

```

★ **Important:** Do not use the Google IDs returned by `getId()` or the user's profile information to communicate the currently signed in user to your backend server. Instead, send ID tokens, which can be securely validated on the server.

Sign out a user

You can enable users to sign out of your app without signing out of Google by adding a sign-out button or link to your site. To create a sign-out link, attach a function that calls the `GoogleAuth.signOut()` method to the link's `onclick` event.

```
<a href="#" onclick="signOut();">Sign out</a>
<script>
  function signOut() {
    var auth2 = gapi.auth2.getAuthInstance();
    auth2.signOut().then(function () {
      console.log('User signed out.');
```

인증
확인은
서버
사이드에
서!



LOG IN

아이디를 입력해주세요.

비밀번호를 입력해주세요.

비밀번호를 잊으셨나요?

로그인

ToySpace는 처음이신가요? [지금 가입하기](#)



로그인



네이버 아이디로 로그인



카카오계정으로 로그인



Sign in - Google Accounts - Chrome



accounts.google.com/o/oauth2/auth/oauthchooseaccount?redirect_uri=storagerelay...



Sign in with Google

Choose an account

to continue to [mightymosses.hopto.org](#)



Byung-hyun Yoon

mightymosses@gmail.com



라쿤

ogj0423@gmail.com



Use another account

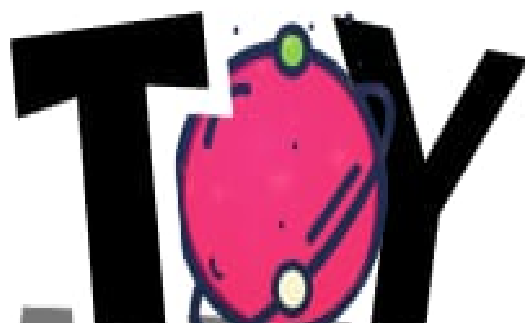
To continue, Google will share your name, email address, language preferences, and



mightymosses.hopto.org:9090 내용:

기존에 가입된 이력이 없습니다. 간편 가입되었습니다.

확인





TOYSPACE 홈페이지
간편로그인으로 회원가입이 완료 되었습니다

추가 정보를 입력하시겠습니까?

확인

취소

첫 가입
시

	MEMBER_KEY	USER_ID	USER_EMAIL	USER_PASSWORD	USER_GENDER	USER_NAME	USER_NICKNAME
1		11 google_1602408169434	blue604321@gmail.com	(null)	(null)	kim kim	google_1602408169434
2		10 google_1602399425938	01088049128a@gmail.com	(null)	(null)	진호 황	google_1602399425938
3		29 mighty	mightymosses@naver.com	1234	X	1234	asdf
4		68 qweqwe	qwe@qweqwe.qwe	qweqwe	X	qwe	qwe
5		69 강경록	kkkk@kk.com	1234	X	강경록	경록
6		76 semin	blue604321@google.com	semin	X	김세민	세밍키
7		79 qweqweqwe	qqq@qweqwe.qwe	qweqweqw	X	qwe	qq
8		81 google_1602825098560	mightymosses@gmail.com	(null)	(null)	Yoon Byung-hyun	google_1602825098560

NAVER

Sign in to ToySpace using Naver account.

angelstrap

Sign in

☒ Stay Signed in

Easier sign in

If this PC is used by multiple people, try it. X



Sign in with QR code



Facebook



line

Forgot your [Username](#) or [Password](#)? | [Sign up](#)

NAVER Copyright © NAVER Corp. All Rights Reserved.

가입된
회원이
다른
간편인증
시

ns_login/redirect/naverRedirect.jsp#access_token=AAAAOmdn_DZzGWRb31ozPs1Mq-w8lrUDA

화상 [

mightymosses.hopto.org:9090 내용:

기존에 가입된 아이디를 발견하였습니다

확인

이메일이 존재합니다.
연동하시겠습니까?

아이디 : **google_1602825098*****

연동

회원가입 취소

mightymosses.hopto.org:9090 내용:

SNS 정보를 google_1602825098560에 삽입 완료!

확인



화상



[4]

mightymosses.hopto.org:9090 내용:

구글을 통해 간편로그인을 하셨습니다.

확인



```

* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
*/
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    // TODO Auto-generated method stub

    gsonFlag의 용도: 응답페이지에 json로 플래그를 보내줌
    1: 이미 가입된 사람인 경우
    2: 가입되어있지만 연동이 안되어있는 경우로 판단되는 경우
    3: 가입되어있지 않은 경우
    int gsonFlag = -1;

    MemberService ms = new MemberService();
    HttpSession session = request.getSession();
    Gson gson = new Gson();

    String idTokenString = request.getParameter("id_token");
    String clientId = request.getParameter("client_id");

    GoogleIdTokenVerifier verifier = new GoogleIdTokenVerifier.Builder(new NetHttpTransport(),
        .setAudience(Collections.singletonList(clientId))
        .build());
    GoogleIdToken idToken=null;
    Payload payload =null;
    try {
        idToken = verifier.verify(idTokenString);
    } catch (GeneralSecurityException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    if (idToken!=null) {
        payload = idToken.getPayload();
        정보 받아오기!

        String id = payload.getSubject();

        //이 구글 유니크 아이디가 벌써 등록되어있는지 체크
        로그인 소스 1은 구글 2는 네이버 3은 카카오
        int loginSource =1;
        Member m = ms.checkMemberThroughSNSId(loginSource, id);

        m이 존재하면 그 아이디로 멤버를 불러와 세션 어트리뷰트에 담는다.
        서비스 통해 로그인 -> 서버로 가서 이미 있는 아이디인지 확인하고
        있으면 멤버 불러 세션에 담아줌 (여차피 로그인 상태는 이 세션에 따라다니니)
        if(m!=null) {
            session.setAttribute("signedInMember", m);

            장바구니 불러오기
            new CartService().loadSavedCart(session, m.getMemberKey());

            gsonFlag=1;

```

```

int loginSource =1;
Member m = ms.checkMemberThroughSNSId(loginSource, id);

m이 존재하면 그 아이디로 멤버를 불러와 세션 어트리뷰트에 담는다.
서비스 통해 로그인 -> 서버로 가서 이미 있는 아이디인지 확인하고
있으면 멤버 불러 세션에 담아줌 (여차피 로그인 상태는 이 세션에 따라다니니)
if(m!=null) {
    session.setAttribute("signedInMember", m);

    장바구니 불러오기
    new CartService().loadSavedCart(session, m.getMemberKey());

    gsonFlag=1;
    gson.toJson(gsonFlag, response.getWriter());
    return;
}

구글 페이로드에서 더 많은 정보 받아오기
만약 해당 이메일로 이미 회원정보가 있으면 패스워드를 확인한 뒤 그걸로 가인
(이메일은 유니크이므로 중복값을 없애기 위함이기도 함)
-만약 패스워드 틀려버렸으면 다시 찾아야함
String email = payload.getEmail();
String pictureURL = (String)payload.get("picture");
String familyName = (String)payload.get("family_name");
String givenName = (String)payload.get("given_name");
String name = familyName+" "+givenName;

SNSLogin sns= new SNSLogin();
sns.setLoginSourceNo(loginSource);
sns.setAccessToken(idTokenString);
sns.setSnsId(id);
sns.setSnsName(name);

Member emailM=ms.checkMemberThroughSNSEmail(email);
if(emailM!=null) {
    session.setAttribute("email-found-new-member", emailM);

    session.setAttribute("SNS-for-email-found-new-member", sns);
    gsonFlag=3;
    gson.toJson(gsonFlag, response.getWriter());
    return;
}

멤버 객체 만들기
Member newM= new Member();
newM.setUserEmail(email);
newM.setUsername(name);
newM.setUserProfilePicUrl(pictureURL);

멤버 정보 생성!
Member signInMember = ms.signInThroughSNS(newM, sns);

session.setAttribute("signedInMember", signInMember);

장바구니 불러오기

```

구글 rest api 객체 사용

상황별 분기 코드 삽입

```

정보: 서비스 [Catalina](를) 시작합니다.
10월 16, 2020 2:19:57 오후 org.apache.catalina.core.StandardEngine startInternal
정보: Starting Servlet Engine: Apache Tomcat/8.5.57
10월 16, 2020 2:19:57 오후 org.apache.jasper.servlet.TldScanner scanJars
정보: 적어도 하나의 JAR가 TLD들을 찾기 위해 스캔되었으나 아무 것도 찾지 못했습니다.
10월 16, 2020 2:19:59 오후 org.apache.jasper.servlet.TldScanner scanJars
정보: 적어도 하나의 JAR가 TLD들을 찾기 위해 스캔되었으나 아무 것도 찾지 못했습니다.
10월 16, 2020 2:19:59 오후 org.apache.coyote.AbstractProtocol start
정보: 프로토콜 핸들러 ["http-nio-9090"]을(를) 시작합니다.
10월 16, 2020 2:19:59 오후 org.apache.catalina.startup.Catalina start
정보: Server startup in 1937 ms

```

REST API
라이브러
리가 편하
긴 하지만..

- 서버 반응 속도도 느려짐...

```

google-api-client-servlet-1.30.2.jar
google-api-client-xml-1.30.2.jar
google-http-client-1.30.1.jar
google-http-client-appengine-1.30.1.jar
google-http-client-gson-1.30.1.jar
google-http-client-jackson2-1.30.1.jar
google-http-client-jackson2-1.30.2.jar
google-http-client-protobuf-1.30.1.jar
google-http-client-xml-1.30.1.jar
google-oauth-client-1.30.1.jar
google-oauth-client-appengine-1.30.1.jar
google-oauth-client-servlet-1.30.1.jar
grpc-context-1.19.0.jar
gson-2.8.6.jar
guava-26.0-android.jar
hamcrest-core-1.3.jar
httpclient-4.5.9.jar
httpcore-4.4.11.jar
import jar

```




장바구니
-아이콘에
애니메이션 추가

체크 아웃을 클릭하면 이용 약관에 동의 하고 모든 판매가 최종적임을 이해합니다. 적용 가능한 모든 할인 또는 쿠폰은 결제 시 반영됩니다.

총 금액

75000 원

상품

개수

가격



디즈니
미키마우스



-

5

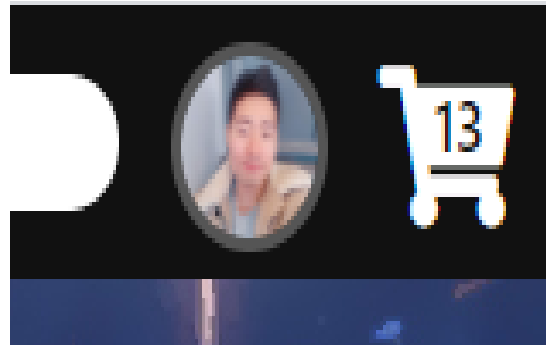
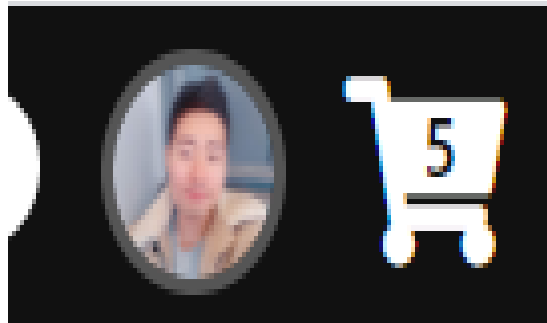
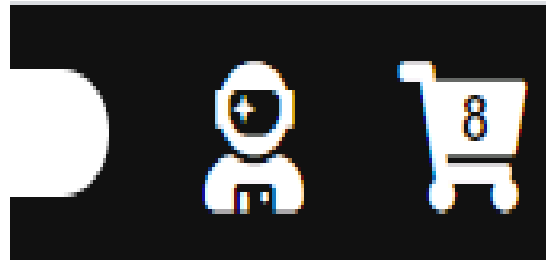
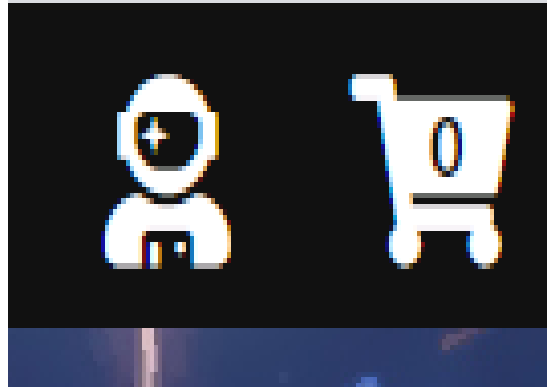
+

75000 원

세션 종료시, 로그인한 멤버라면 장바구니 저장 - 세션 리스너

```
30 public void sessionCreated(HttpSessionEvent e) {  
31     // TODO Auto-generated method stub  
32 }  
33  
34 /**  
35  * @see HttpSessionListener#sessionDestroyed(HttpSessionEvent)  
36  */  
37 public void sessionDestroyed(HttpSessionEvent e) {  
38     HttpSession session = e.getSession();  
39     if(session.getAttribute("signedInMember")!=null && session.getAttribute("cart")!=null) {  
40         @SuppressWarnings("unchecked")  
41         TreeMap<Integer, Integer> cartValues=(TreeMap<Integer, Integer>)session.getAttribute("cart");  
42         Member m = (Member)session.getAttribute("signedInMember");  
43         int memberKey = m.getMemberKey();  
44         boolean result = new CartService().updateMemberCartValues(memberKey, cartValues);  
45  
46         if(result) System.out.println(memberKey+"님의 장바구니가 업데이트 되었습니다.");  
47         else System.out.println(memberKey+"님의 장바구니 업데이트에 실패했습니다.");  
48     }  
49 }  
50  
51 }  
52 }
```

기존 장바구니 로드- > 로그인 전의 내용과



AJAX 꼼수 -> AJAX가 리턴하 는 값은 플래그

```
203     $('#summary-amount').html(totalAmount);
204 }
205
206 function removeItem(e, productId){
207     $.ajax({
208         url: "<%=contextPath%>/cart/removeItem",
209         type: "POST",
210         data: {
211             "productId" : productId
212         },
213         success: (data) =>{
214             successRoutine(data);
215             $("#summary-qty").html(data);
216             const targetCont = $(e.target).parent().parent().parent();
217             targetCont.prev().remove();
218             targetCont.next().remove();
219             targetCont.remove();
220
221             let totalAmount = 0;
222             $(".sub-amount").each((i,v)>=>{
223                 totalAmount+=Number($(v).html());
224             })
225             $("#summary-amount").html(totalAmount);
226         },
227         fail : error =>{
228             console.log(error);
229         }
230     })
231 }
```

결제

```
/**
 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
 */
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    // TODO Auto-generated method stub
    // 나중에 카드의 작업을 옮기는 과정을 해야합니다.
    // 프로덕트 서비스로 아임포트에 결제 과정 전 처리 → 디비/결제 담기에 결제 준비중으로 담기
    HttpSession session = request.getSession();
    Member m = (Member)session.getAttribute("signedInMember");
    TreeMap<Integer, Integer> cartValues = (TreeMap<Integer, Integer>)session.getAttribute("cart");
    TreeMap<Integer, Product> productInfo = new ProductService().loadSelectedProductsWithMainPic(cartValues);
    int totalAmount=0;
    for(int productId : productInfo.keySet()) {
        Product p= productInfo.get(productId);
        totalAmount+=p.getProductPrice() * cartValues.get(productId);
    }

    String merchant_uid = new OrderHistoryService().createPaymentLog(m.getMemberKey(), cartValues, totalAmount);

    String path = "";

    if(merchant_uid==null) {
        path="/msg?loc="+request.getContextPath()+"/cart.do&msg=결제 정보 생성에 실패했습니다.";
    } else {
        path = "/views/cart/payment.jsp";
        request.setAttribute("merchant_uid", merchant_uid);
        request.setAttribute("productInfo", productInfo);
    }

    // 지금은 카트가 완성되지 않았기 때문에 데모로 그냥 넘어가도록 함
    request.getRequestDispatcher(path).forward(request, response);
}
/**
```

결제 모듈 소환

```
requestPay();
return false;
}
function requestPay(){
    IMP.request_pay({ // param
        pg: "html5_inicis",
        pay_method: "card",
        merchant_uid: '<%=merchant_uid%>',
        name: '<%=productInfo.get(firstProduct).getProductName()%> 외 <%=productInfo.size()-1%>',
        amount: Number('<%=totalAmount%>'),
        buyer_email: $("#email").val(),
        buyer_name: $("#lastName").val()+" "+$("#firstName").val(),
        buyer_tel: $("#phone-first").val()+"-"+$("#phone-second").val()+"-"+$("#phone-third").val(),
        buyer_addr: $("#roadAddrPart1").val()+" "+$("#addrDetail").val()+" "+$("#roadAddrPart2").val(),
        buyer_postcode: $("#zipNo").val()
    });

    }, function (rsp) { // callback
        if (rsp.success) {
            console.log("성공");

            $.ajax({
                url: '<%=contextPath%>/order/process',
                type: "POST",
                data:{
                    "imp_uid" : rsp.imp_uid,
                    "merchant_uid" : rsp.merchant_uid,
                    "member_key" : <% if(m!=null){%><%=m.getMemberKey()%><%}%>,
                }
            }).done(function (data) {
                /* 서버 결제 api성공시 로직! */
                const result = JSON.parse(data);
                switch(result){
                    case 0 : alert("결제 위변조가 발생하였습니다. 서버관리자에게 보고됩니다."); location.href = "<%=contextPath%>/order/process";
                    case 1 : location.href = "<%=contextPath%>/order/complete"; break;
                }
            })
        } else {
            console.log("실패");
            $.ajax({
                url: '<%=contextPath%>/order/moveFromPayment',
                type: "POST",
                data:{
                    "merchant_uid" : '<%=merchant_uid%>'
                },
                success: (data)⇒{
                    <%System.out.println("성공");%>
                },
                fail: error⇒{
                    <%System.out.println("실패");%>
                }
            })
        }
    })
}
```

결제 서버사이드 인증 필수!
-> 자바스크립트는 노출되어 있어
결제금액을 위변조할 수 있기 때문

- 아임포트 라이브러리 사용
- -> 메이븐 때문에 골치앳음

```
e.printStackTrace();
}

IamportResponse<Payment> payRes = null;

try {
    payRes = client.paymentByImpUid(imp_uid);
} catch (IamportResponseException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
} catch (IOException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}

Payment payment = payRes.getResponse();
//////////테스트 코드//////////
/*
 * System.out.println(imp_uid);
System.out.println(payment.getImpUid());
System.out.println(merchant_uid);
System.out.println(payment.getMerchantUid());
System.out.println(amount);
System.out.println(payment.getAmount().doubleValue());
*/
//
if(imp_uid.equals(payment.getImpUid()) && merchant_uid.equals(payment.getMerchantUid()) &&
    amount == payment.getAmount().intValue()) {

서버저장 로직 실행!
System.out.println(m.getMemberKey()+"님의 결제가 정상적으로 완료되었습니다!");
OrderHistory oh = new OrderHistory();

oh.setPaymentMethod(payment.getPayMethod());
oh.setApplyNum(payment.getApplyNum());
oh.setBuyerTel(payment.getBuyerTel());
oh.setReceiverName(payment.getBuyerName());
oh.setReceiverPostcode(payment.getBuyerPostcode());
oh.setReceiverAddr(payment.getBuyerAddr());
```


SHOP

ALL PRODUCTS

Filters

- ☐ 디즈니(9)
- ☐ 스타워즈(4)
- ☐ 왕좌의 게임(4)
- ☐ 포켓몬스터(8)
- ☐ 마블(4)

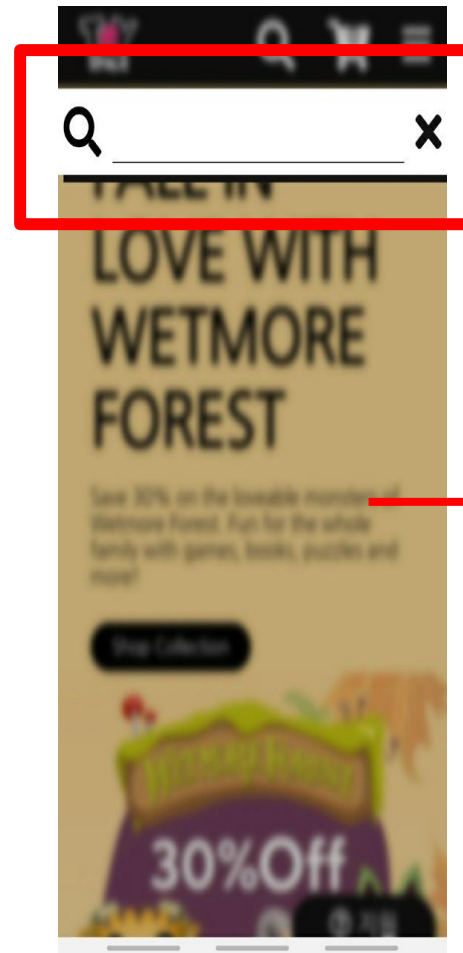
검색

제품수(29)

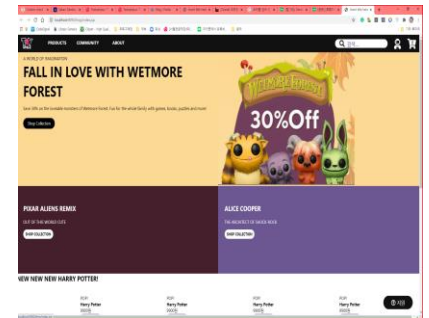


코드
->경록
세민

->숫자
표시 및
다중 검색
구현(병현)



블러 처리로
포커스
아웃.
사용자
경험
향상



반응형