반응형 웹/앱
가구 쇼핑몰 사이트

조원 : 정성윤, 우수정, 임세훈, 김수훈

ILOKE

ILOKE

CONTENTS

01 기획

02 벤치마킹

03 마인드 맵

04 GUI

05 작업 툴

06 요구사항 정의서

07 시스템 구조도

08 Use Case

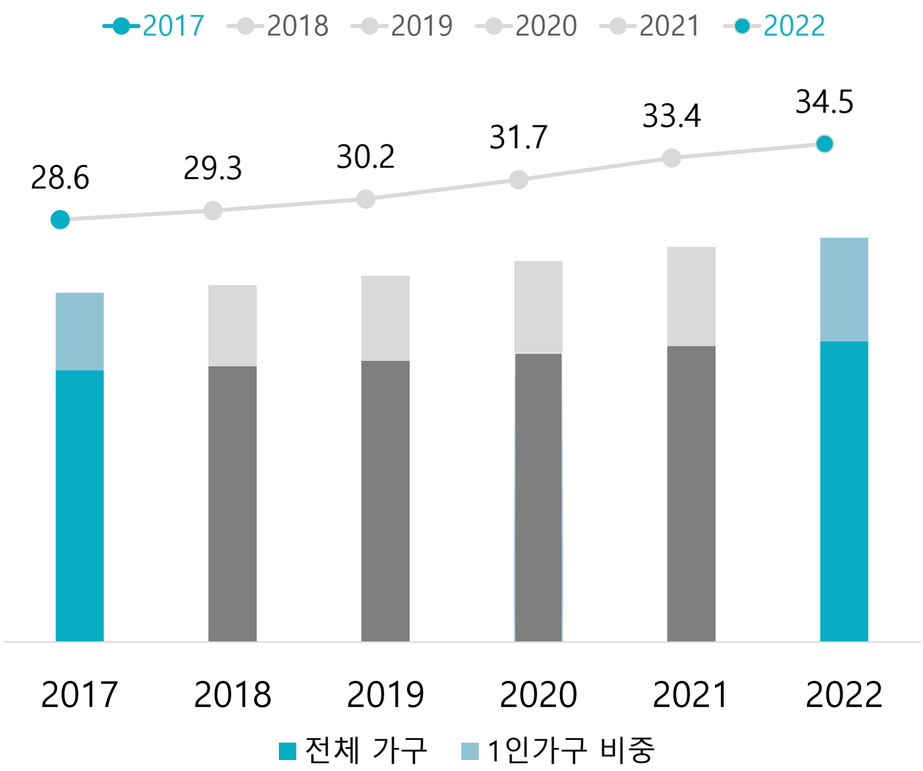
09 ERD

10 역할 분담

11 시간 계획서

기획

1인 가구 추이



단위: 가구, % / 자료 = 통계청 (2023.12.12) https://www.chosun.com/

1인 가구가 증가하는 시대에 1인 가구를 위한 상품에 대한 접근성이 중요

기획

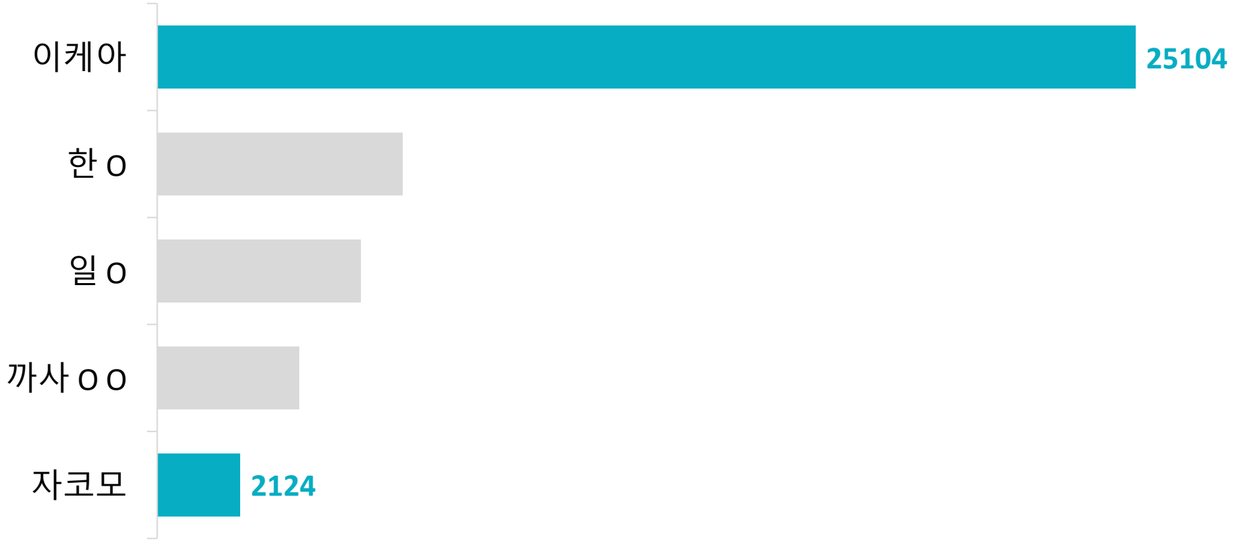
２０３０ 남녀 가격 대비 만족도 조사

출처 : 데일리팝 (2016.12.09)

1인 가구 비율이 높고 혼자 사용하기 가성비 좋은 제품 찾기 때문

기획

랭키파이 브랜드 가구 순위 차트



출처 : 랭키파이 (2024.04.24)

키워드 검색량과 구글 트렌드 점수를 이용해 랭키파이에서 산출한

트렌드 지수 **이케아 1위**

기획

고객 인지도 1위 '자코모', 베스트 소파 브랜드 대상

오한준 기자 | 승인 2024.04.18 17:10 | 댓글 0

사진=자코모 제공

소파 전문 브랜드 자코모가 '2024 대한민국 대표 브랜드 대상'에서 소파브랜드 부문 대상을 수상하였다. 자코모는 모든 문항에서 압도적인 1위를 달리며 명실상부 국내 대표 소파 브랜드임을 입증했다.

출처 : 데일리경제 (2024.04.18) http://www.kdpress.co.kr/

2 0 2 4 대한민국 대표 베스트 소파 브랜드 대상

고객 인지도 자코모 1위

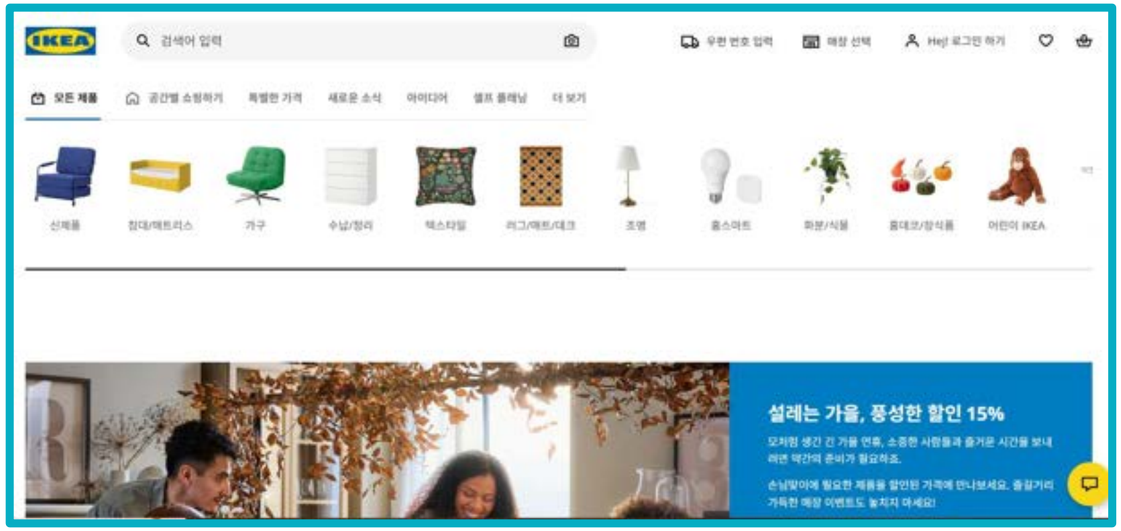
기획

자코모

브랜딩을 위한 큰 사진과 첫 카테고리에 브랜드 나열

출처 : 자코모 홈페이지 https://jakomo.co.kr/main/index.php

이케아



상품 위주 나열이며 브랜드 관련 내용은 좌측 상단 로고 배치가 전부

출처 : 이케아 홈페이지 https://www.ikea.com/kr/ko/

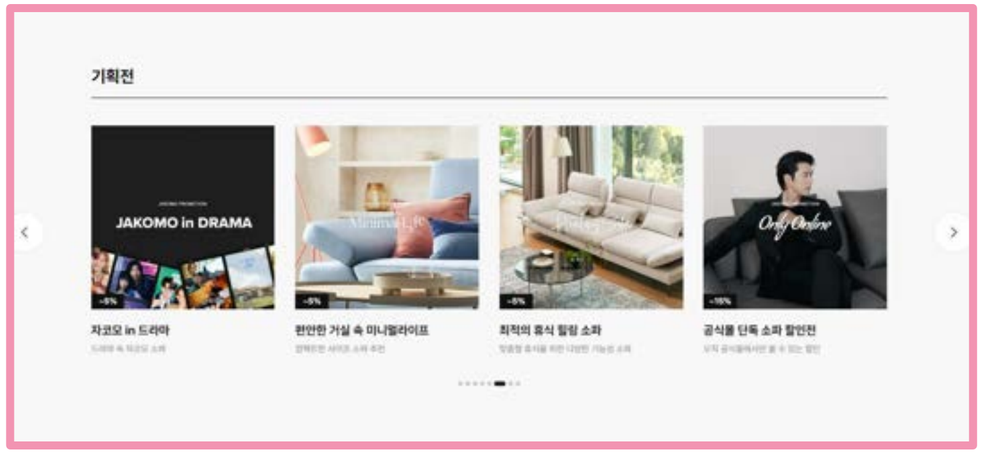
브랜드 친화적 메인 화면 레이아웃 구성을 가진 자코모

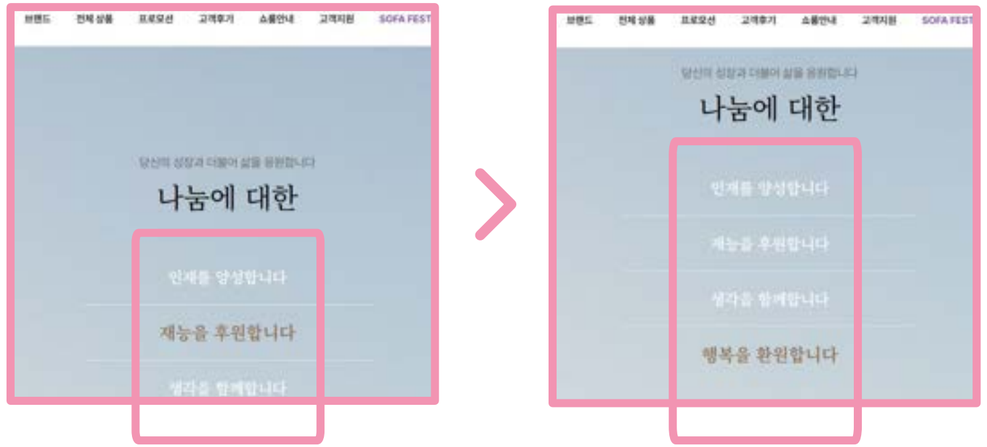
벤치마킹

JAKOMO 장점

출처 : 자코모 홈페이지 https://jakomo.co.kr/main/index.php

세련된 디자인으로 품격 높은 브랜드 이미지 표현

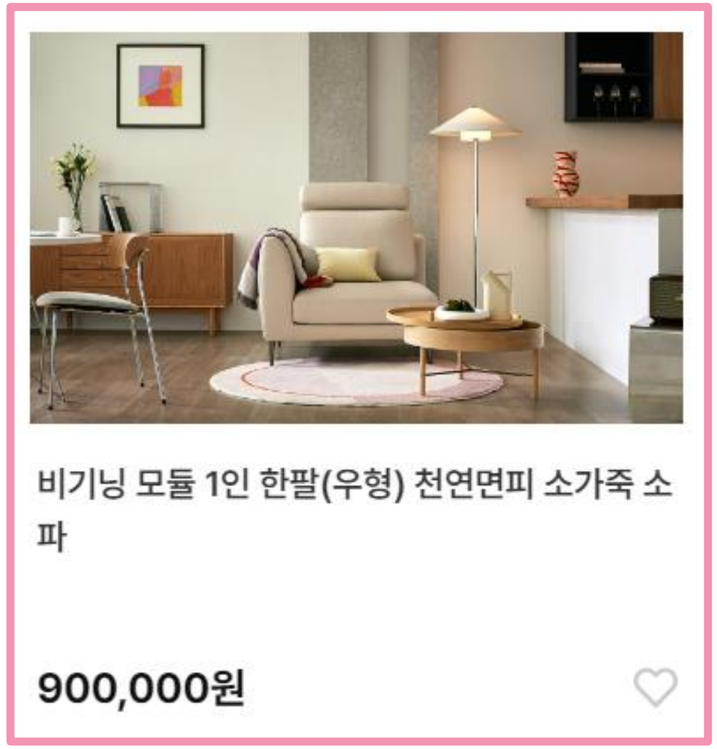




기획전과 이벤트가 다양하며, 시각적으로 화려

벤치마킹

JAKOMO 단점



출처 : 자코모 홈페이지 https://jakomo.co.kr/main/index.php

제품 상세 정보 부족

상품 비용이 높아 구매에 부담

벤치마킹

장점

더 낮은 새로운 가격

SEGERÖN 세게륀

야외커피테이블, 화이트/베이지, 73x73 cm

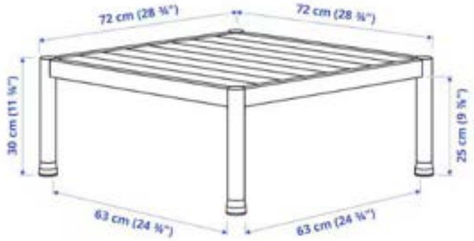
₩119,000

기존가: ₩139,000

(0)

치수

높이: 30 cm
길이: 73 cm
폭: 73 cm



SEGERÖN 세게륀

야외커피테이블

제품 번호

905.111.72

폭: 44 cm
높이: 7 cm
길이: 73 cm
무게: 10.69 kg
패키지: 1

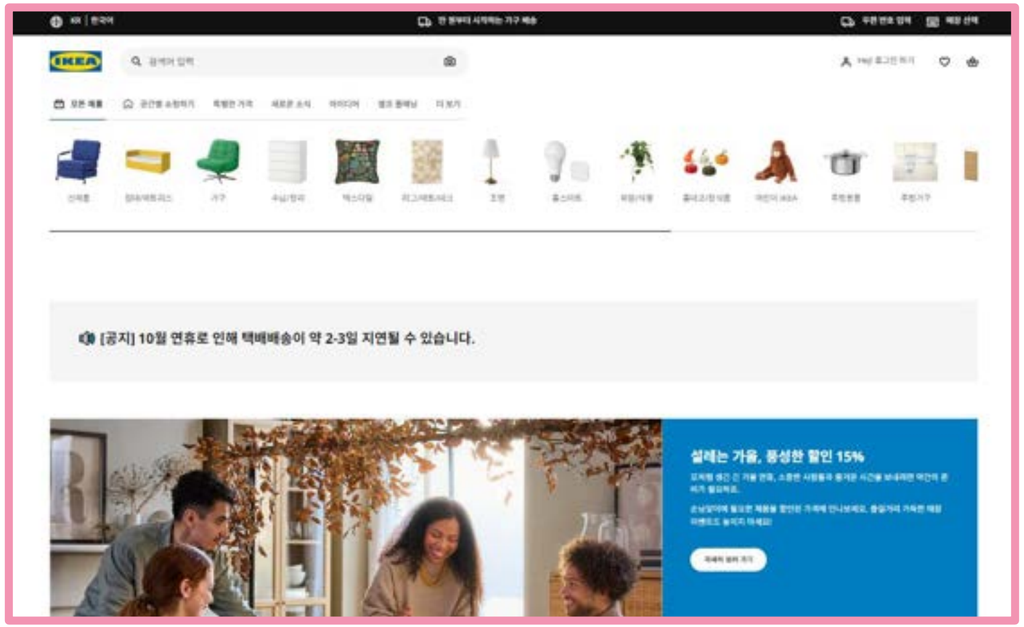
출처 : 이케아 홈페이지 https://www.ikea.com/kr/ko/

정보 제공 풍부
이동, 재배치가 간단한 조립식 가구
위주

벤치마킹

IKEA 단점

출처 : 이케아 홈페이지 https://www.ikea.com/kr/ko/



이벤트 다소 부족
시각적으로 화려하지 않아 사이트 자체의 매력은 없음
기업 브랜딩이 약함

벤치마킹 - 결론

JAKOMO, IKEA의 UI/UX 디자인, 상품 정보 제공 방식, 마케팅 및 프로모션을 바탕으로 벤치마킹

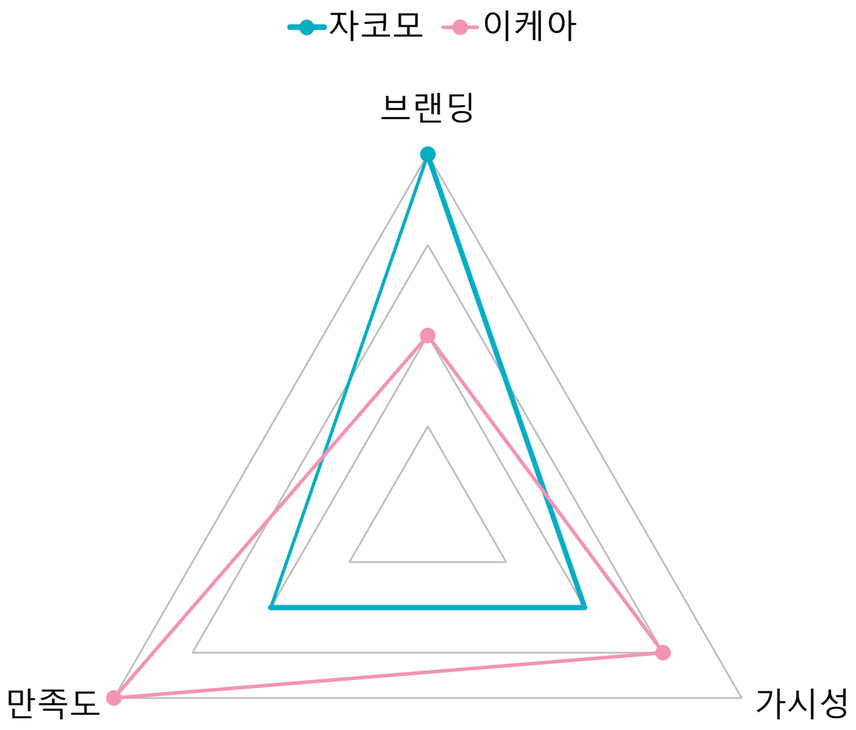
0 1. **브랜딩**
업체와 제품의 이미지 및 설명이 얼마나 매력적인지 평가

0 2. **가시성**
레이아웃과 네비게이션이 직관적인지 평가

0 3. **만족도**
판매중인 상품이 편리한지 평가

벤치마킹 - 결론

벤치마킹 웹 사이트 분석 그래프

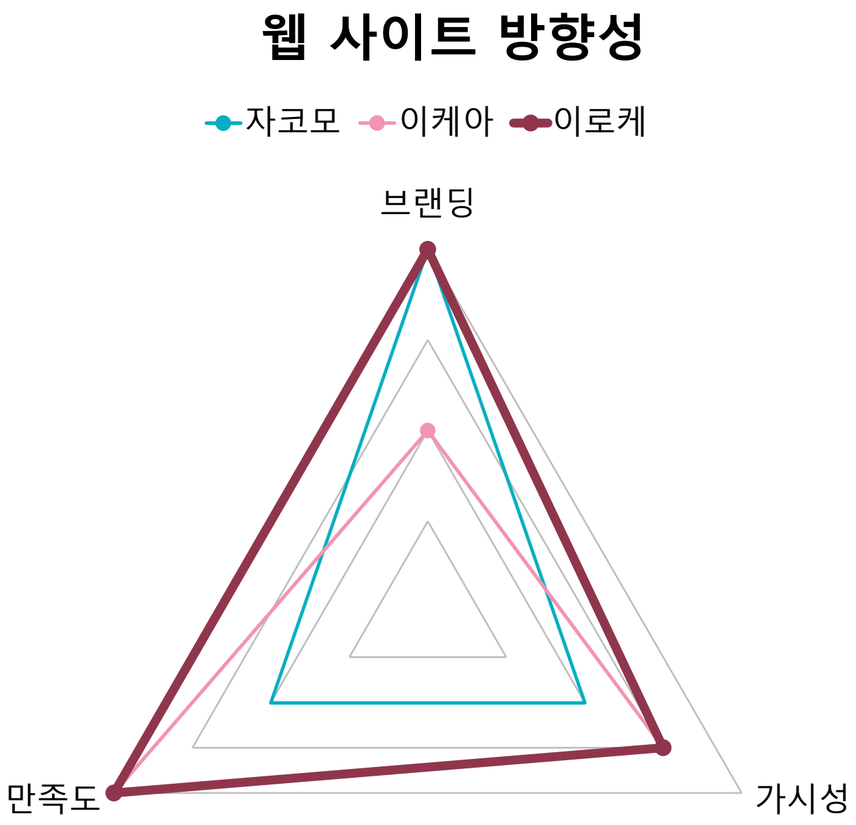


자코모의 세련된 디자인과 브랜딩

+

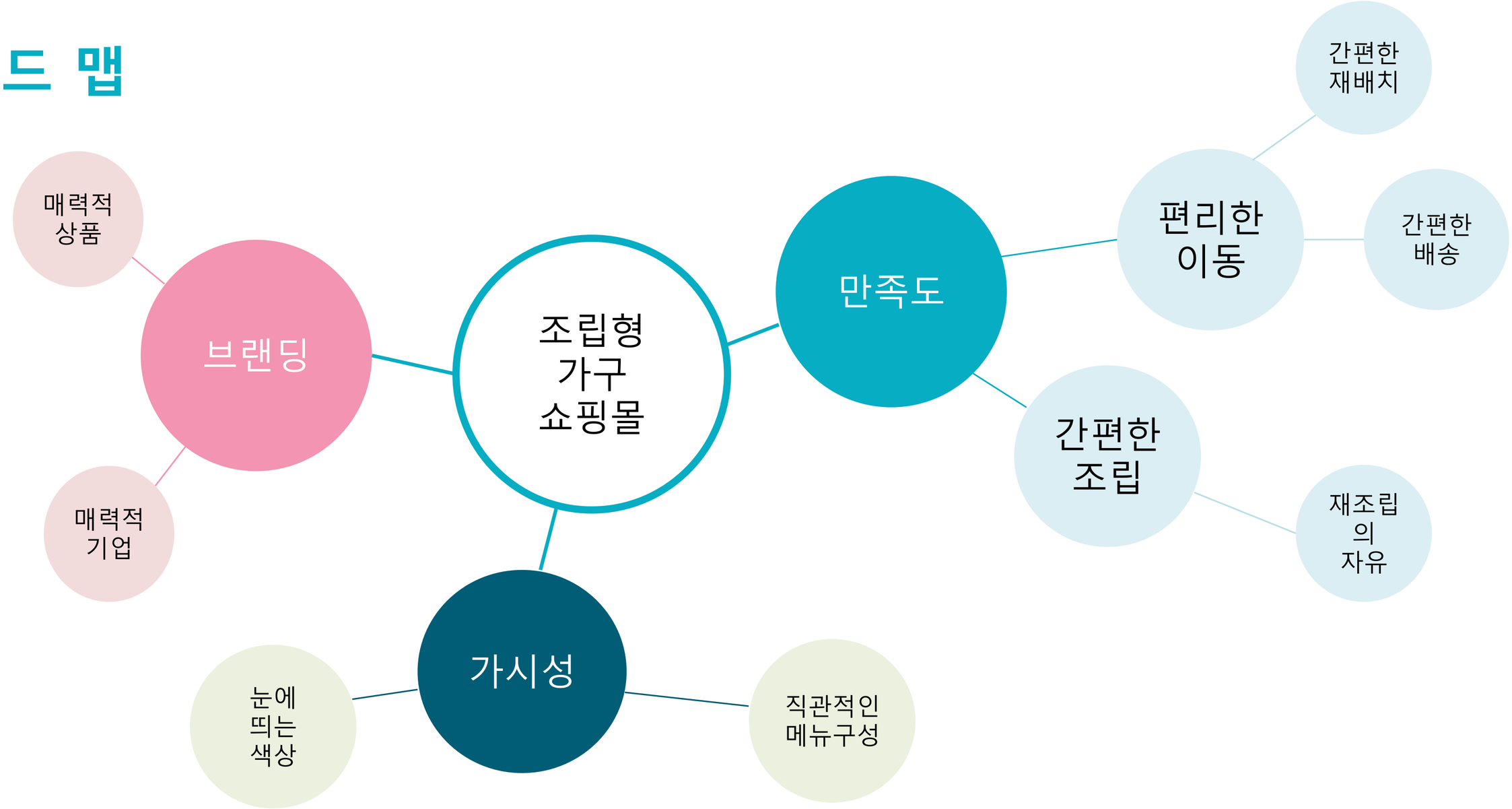
이케아의 편리하고 풍부한 상품 정보 제공 방식

벤치마킹 - 결론



세련된 디자인과 브랜딩,
편리하고 풍부한 상품 정보 제공 방식을 가진
1인 가구를 위한 가구 쇼핑몰 사이트

마인드 맵


조립형
가구
쇼핑몰
브랜딩
매력적
상품
매력적
기업
만족도
편리한
이동
간편한
재배치
간편한
배송
간편한
조립
재조립
의
자유
가시성
눈에
띄는
색상
직관적인
메뉴구성

GUI – 색상선정

#8F364D	#F294B2
#005D75	#06ADC3

메인 포인트 색상
분홍 #F294B2
초록 #06ADC3

서브 색상
진분홍 #8F364D
진초록 #005D75

주요 타깃 연령층인 2030의 젊고 활기찬 이미지를 반영하여
보색 대비를 통해 밝고 에너지 있는 느낌을 강조,
균형을 위해 가볍거나 부담스럽지 않은 부드러운 분홍-민트 색조 사용

GUI – 폰트선정

한글 - Noto Sans Kr	영어 – Roboto
가나다라마바사아자차카타파하	AbCdEfGhIjKlMnOpQrStUvWxYz
가나다라마바사아자차카타파하	AbCdEfGhIjKlMnOpQrStUvWxYz
가나다라마바사아자차카타파하	AbCdEfGhIjKlMnOpQrStUvWxYz
가나다라마바사아자차카타파하	AbCdEfGhIjKlMnOpQrStUvWxYz
가나다라마바사아자차카타파하	AbCdEfGhIjKlMnOpQrStUvWxYz

저작권 문제가 없어 무료로 사용이 가능하며
깔끔한 디자인에 기존 타 사이트 에서도 많이 사용하는 폰트이기에 사용자 친숙

작업툴

IDE

VSCode Ver_1.93.1

OS

Windows 10 Ver_22H2

Windows 11 Ver_23H2

런타임 환경

Node.JS
Ver_22.12.0

DBMS

MariaDB Ver_11.5.2

OA

PowerPoint Office16

Excel Office16

디자인

FontAwsome Ver_6.6.0

그림판 Ver_11.2406.42.0

작업툴

웹 클라이언트

React

SCSS

Swiper Ver_11.2.2

HTML5

CSS3

JS

JQUERY Ver_3.7.1

협업도구

카카오톡

Git Ver_2.46.0.windows.1

Github

SourceTree Ver_3.4.19

WinMerge Ver_2.16.42.1

요구사항 정의서 - 사용자

우선순위	분류	소분류	기능	중요도	전제조건
1	장바구니	상품 목록	장바구니 담기 버튼을 누르면 해당 상품 및 옵션정보, 추가금, 수량 등이 저장되며, 저장된 정보를 장바구니로 가져와 상품목록을 조회.	상	홈페이지 우측 상단 ↓ 장바구니 아이콘 ---------- 상품리스트 ↓ 장바구니 아이콘
		선택 상품 삭제	각 상품목록의 체크박스를 선택 후 선택 상품 삭제버튼을 누르면 해당 상품이 삭제.		
		쇼핑 계속하기	클릭 시 전체상품으로 연결.		
		합계 표시 구역	선택된 상품 금액이 실시간으로 반영.		
		상품 주문	주문 페이지로 연결.		
2	상품주문	주문 상세 내역	장바구니에서 보여줬던 상품 정보들을 조회.	상	장바구니 ↓ 상품주문 버튼
		장바구니 가기	장바구니로 연결.		
		주문자 정보	주문자 정보는 고정된 회원정보로 사용.		
		배송 정보	휴대폰 번호, 받으실 분, 받으실 주소 등을 필수적으로 입력하고 유효성 검사를 진행.		
		결제 정보	사용자가 보유한 잔여 마일리지를 표시. 사용자가 사용할 마일리지 양을 입력 시 실시간으로 업데이트. 잔여 마일리지와 결제금액을 변동하여 업데이트. 가지고 있는 양 이상의 마일리지를 사용하려고 할 시 잔여 마일리지만큼만 사용.		
		결제수단 선택	결제 방법을 신용카드, 휴대폰 중에 체크박스로 선택.		
		최종 결제 금액	마일리지 입력 값, 할인율에 따라 값 변동.		
		동의란	체크박스에 체크해야 결제 가능.		
		결제하기	필수입력사항 유효성 검사(이름-한글만/ 주소-한글,영어,숫자만/ 이메일/ 휴대폰번호) 후 주문완료 페이지로 연결. 데이터를 추가. 그 후 주문완료 페이지로 이동.		
3	주문완료	주문 요약	결제수단, 주문번호, 받으실 분 이름, 주소, 휴대폰번호, 결제금액 등을 화면에 표시.	중	상품주문 ↓ 결제하기 버튼
		확인 버튼	클릭 시 메인페이지로 연결.		

요구사항 정의서 - 사용자

우선순위	분류	소분류	기능	중요도	전제조건
1	회원가입 정보입력	아이디	영어 대/소문자+숫자 조합 최소 4자 이상. 12자 이하가능. 조건 일치시, 불일치시 각각 메시지 출력. 중복확인기능 및 메시지 출력. 입력하지 않고 회원가입 누를 시 아이디가 유효하지 않다는 메시지창 팝업.	상	회원가입 안내 ↓ 쇼핑몰 회원가입 버튼 ↓ 필수약관동의 후 다음단계 버튼
		비밀번호	영문 대/소문자, 숫자, 특수문자 중 2가지 이상 조합, 10자 이상. 조건 일치시, 불일치시 각각 메시지 출력. 입력하지 않고 회원가입 누를 시 비밀번호가 유효하지 않다는 메시지창 팝업.		
		비밀번호 확인	위 비밀번호 조건과 동일. 위 비밀번호와 일치, 불일치 한다는 메시지 출력. 불일치시 회원가입 버튼 누르면 비밀번호가 일치하지 않습니다 메시지창 팝업.		
		이름	한글 2자~6자 입력가능. 입력하지 않고 회원가입 누를 시 이름을 입력해주세요 메시지창 팝업.		
		닉네임	선택사항이지만 입력 시 중복체크 및 메시지 출력. 한글, 영문자, 숫자 가능. 특수문자 불가능. 최소 2자 이상 10자 이하. 조건 일치시, 불일치시 각각 메시지 출력.		
		이메일	로컬파트(영문자, 숫자, ._%+- 만 가능) @ 도메인(영문자와 숫자, .- 만 가능) . 최상위도메인(영문자만 허용, 최소 2글자 이상). 도메인 선택 시 로컬파트 뒤에 자동으로 입력. 조건 일치시, 불일치시 각각 메시지 출력.		
		휴대폰 번호	번호형식은 xxx-xxxx-xxxx. 미입력 또는 조건 불일치 시 회원가입 버튼 누를 시 휴대전화 번호 형식이 유효하지 않습니다 메시지창 팝업.		
		주소	한글, 숫자, 영문자, 공백, 특수문자 '-'만 허용, 최소 8자 이상.		
		생년월일	선택사항. 미입력도 통과.		
		가입완료	회원 가입 완료.		
2	회원가입 완료	홈으로, 로그인 버튼	회원가입 완료 페이지 (회원이름)님의 회원가입을 축하합니다 라는 메시지가 출력. 홈으로 버튼 누를 시 메인페이지로, 로그인 버튼 누를시 로그인 페이지로 이동.	하	회원가입 정보입력 ↓ 회원가입 버튼

요구사항 정의서 - 사용자

우선순위	분류	소분류	기능	중요도	전제조건
1	회원가입 안내	쇼핑몰 회원가입	회원가입 페이지로 이동.	하	메인페이지 우측 상단 회원가입
2	회원가입 약관동의	이용 약관 동의	약관 전체동의 체크시 전체동의 체크 및 개별 체크 .	하	회원가입 안내 ↓ 쇼핑몰 회원가입 버튼
		이전	회원가입 진입 전 회원가입 안내화면으로 이동.		
		다음	회원가입 정보입력 페이지로 이동(필수항목 체크시에만 가능. 필수항목 미 체크시 필수 약관에 모두 동의해주세요 라는 메시지 출력.		
3	아이디 찾기	이름, 이메일 입력	도메인 선택시 @와 도메인이름이 자동으로 이메일 아이디 뒤에 붙음.	중	홈페이지 접속 ↓ 헤더 우측 로그인 ↓ 회원 로그인 폼 하단 아이디 찾기 비밀번호 찾기 버튼
		아이디 찾기 버튼	데이터베이스에 등록된 아이디와 이메일 일치 시 찾기 완료. 불일치시 찾기 불가.		
		아이디 찾기 완료	찾고자 하는 회원 아이디 출력.		
		비밀번호 찾기, 로그인하기	비밀번호 찾기 페이지, 로그인 페이지로 이동.		
4	비밀번호 찾기	이름, 아이디 입력	이름과 아이디 입력. 유효성 검사는 하지 않음.	중	
		다음	이름과 아이디가 일치 시 비밀번호 재설정 가능. 불일치시 회원정보가 일치하지 않습니다 라는 메시지 출력.		
		아이디 찾기	아이디 찾기 페이지.		
		비밀번호 재설정 페이지	영문 대/소문자, 숫자, 특수문자 중 2가지 이상 조합, 10자 이상 재설정. 조건 불충족 조건에 맞게 입력하라는 메시지 출력. 조건은 충족하지만 위 아래 비밀번호 불일치시 비밀번호가 일치하지 않습니다 메시지 출력		
		새 비밀번호 입력	유효성 조건에 맞고 두 비밀번호가 일치해야 함		
		확인	위 조건들을 충족시키면 비밀번호 변경완료 메시지와 함께 비밀번호 변경완료 화면 이동.		
		비밀번호 변경 완료 페이지	비밀번호가 성공적으로 변경되었다는 메시지와 로그인 버튼 누를 시 로그인 페이지로 이동.		

요구사항 정의서 - 사용자

우선순위	분류	소분류	기능	중요도	전제조건
1	로그인	로그인 페이지	로그인 화면	상	홈페이지 접속 ↓ 헤더 우측 상단 로그인
		로그인 버튼	아이디, 비밀번호 미입력시 '입력란을 작성하세요' 메시지. 아이디, 비밀번호 불일치시 '아이디 또는 비밀번호를 잘못입력하셨습니다.' 출력. 아이디와 비밀번호 일치시 정상 로그인 되면 메인 화면으로 이동		
2	마이페이지	진행 중인 주문	배송준비, 배송중, 배송완료, 구매확정 / 취소, 반품 몇 건씩 인지 표시	상	홈페이지 접속 ↓ 헤더 우측 상단 사람 아이콘 클릭
		최근 주문 정보	배송준비, 배송중 상태에서는 주문취소 가능, 배송완료 상태에서는 구매확정, 반품완료만 가능구매확정인 상태에서는 후기 등록 가능. 구매 확정될 시 구매 총 금액의 10%가 마일리지로 적립됨.		
		마일리지	현재 보유 중인 마일리지 표시. 마일리지 점수 클릭 시 사용 및 적립 내역이 담긴 표가 출력.(적립 날짜, 적립내용, 유효날짜, 변동사항 확인 가능) 돌아가기 버튼 누를 시 마일리지 내역 표는 사라지고 기존 주문 정보 확인 내역들 출력		
		회원 정보 수정	회원 정보 수정 전 비밀번호 인증비밀번호 일치 시 회원 정보 수정 페이지 진입. 아이디 제외한 개인정보 수정 가능. (유효성은 회원가입과 동일)		
		회원 탈퇴	회원 탈퇴 안내문과 함께 비밀번호 입력창 출력. 비밀번호 일치시 '정말로 탈퇴하시겠습니까' 라는 메시지와 함께 확인을 누르면 회원 탈퇴 처리. 탈퇴회원 정보는 데이터베이스에는 남아있지만 로그인 허용하지 않음.		
3	메인화면	메인 배너 영상 섹션3	제품 등을 슬라이드 형식으로 보여주고 각 배너 클릭 시 해당 페이지로 이동.	중	홈페이지 접속
5	푸터	대표번호 링크	회사정보, 고객센터 전화번호와 운영시간, 이메일 표시 및 SNS링크 바로가기	하	홈페이지 접속
6	네비게이션	메인, 서브메뉴	메뉴에 마우스 올리면 메뉴명 색상 변경, 하단에 하위 서브메뉴 활성화	중	홈페이지 접속

요구사항 정의서 - 사용자

우선순위	분류	소분류	기능		중요도	전제조건
1	상품 리스트	상품 전체보기	등록 된 총 상품 개수 출력		상	네비게이션 ↓ 전체상품 ↓ 전체보기
			등록 된 상품의 이름, 가격 데이터 출력			
			할인율이 있다면 원가에 자동 계산되어 최종금액으로 출력			
2	상품 상세	상품 정보 조회	등록 된 상품의 이름, 가격 데이터 출력		상	상품 리스트 클릭
			상품 최종 선택 후 장바구니에 구매 할 상품 데이터 전달			
			할인율이 있다면 원가에 자동 계산되어 최종금액으로 출력			
		옵션선택	옵션 선택 후 추가 시 유료 옵션일 경우 옵션가격이 포함되어 최종금액으로 출력			
		슬라이드	선택한 이미지 확인 가능. 이미지 목록 슬라이드 기능			
3	게시물 리스트	전체후기	작성 및 수정 가능	페이징 처리 기간 조회 기간을 클릭하여 해당 날짜를 박스에 표시 조회기간 버튼으로 바로 기간 설정 마지막 날짜가 시작하는 날짜보다 앞에 있으면 문구 표시 글쓰기 버튼을 누르면 상품문의 글쓰기 페이지로 이동 답변이 필요한 게시판에 답변 유무 조회 조회할 때마다 조회수 값 +1	상	네비게이션 ↓ 고객후기 - 전체후기 고객센터 – 공지사항 고객센터 - 상품문의 고객센터 - FAQ
		포토후기	작성(파일 업로드) 및 수정 가능			
		공지사항	읽기 가능			
		상품문의	전체 글 조회 및 작성, 수정 가능 (답변처리 유무 확인 가능)			
		FAQ	읽기 가능			
		1:1 문의	본인 작성 글 조회 및 작성, 수정 가능 (답변처리 유무 확인 가능)			
4	게시물 상세		각 상세 페이지에 맞는 (수정, 삭제, 목록으로) 기능 활성화		중	게시물 리스트 클릭

요구사항 정의서 - 관리자

우선순위	분류	소분류	기능	중요도	전제조건
1	관리자 로그인	로그인	관리자 페이지는 접근 권한이 관리자인 아이디로만 로그인 가능	상	관리자 페이지 접속
2	상품등록	등록	새로운 상품의 이름, 가격, 사진, 재고, 설정 가능한 옵션/상세정보 키워드 선택, 상품 사용 여부, 가격/할인 데이터 등록	상	관리자 페이지에서 상품등록 페이지 접속
3	상품목록	수정	등록 된 상품 정보 수정 후 저장	상	관리자 페이지에서 상품목록 페이지 접속
		조회	등록 된 전체 상품 리스트 조회 및 각 상품 상세 조회		
4	옵션관리	등록	상품 등록 시 필요한 옵션 / 옵션값 신규 등록	상	관리자 페이지에서 옵션 관리 페이지 접속
		수정	등록 된 옵션/옵션값 수정		
		삭제	등록 된 옵션/옵션값 삭제		
5	키워드 관리	등록	상품 검색 시 필요한 상품 상세정보 키워드 / 키워드 값 신규 등록	중	관리자 페이지에서 상세정보 키워드 페이지 접속
		수정	등록 된 상품 상세정보 키워드 / 키워드 값 수정		
		삭제	등록 된 상품 상세정보 키워드 / 키워드 값 삭제		

요구사항 정의서 - 관리자

우선순위	분류	소분류	기능	중요도	전제조건
1	주문관리	시작일	시작일을 기점으로 그 이후의 데이터만 포함하여 조회할 수 있는 필터	상	관리자 페이지에서 주문 조회 페이지 접속
		종료일	종료일을 기점으로 그 이전의 데이터만 포함하여 조회할 수 있는 필터		
		주문 상태	"전체", "입금대기", "입금완료", "배송준비", "배송시작", "배송완료" 의 상태별로 나누어 조건에 맞는 데이터 조회		
		결제 방법	"전체", "휴대폰", "신용카드" 의 상태별로 나누어 조건에 맞는 데이터 조회		
		조회	"전체", "구매확정", "구매미확정" 의 상태별로 나누어 조건에 맞는 데이터 조회		
		초기화	필터에 맞는 조건들을 선택한 후, 조회버튼을 클릭하면 필터에 맞는 데이터들을 조회		
2	환불관리	시작일	시작일을 기점으로 그 이후의 데이터만 포함하여 조회할 수 있는 필터	상	관리자 페이지에서 환불 관리 페이지 접속
		종료일	종료일을 기점으로 그 이전의 데이터만 포함하여 조회할 수 있는 필터		
		상태	"전체", "취소요청", "반품요청", "교환요청", "취소완료", "반품완료", "교환완료" 의 상태별로 나누어 조건에 맞는 데이터 조회		
		조회	필터에 맞는 조건들을 선택한 후, 조회버튼을 클릭하면 필터에 맞는 데이터들을 조회하는 기능		
		초기화	모든 필터와 조건들을 초기화		
3	정산관리	일별 조회	각 일별에 해당하는 상품 데이터의 총 매출액을 조회	중	관리자 페이지에서 정산 관리 페이지 접속
		주별 조회	각 주별에 해당하는 상품 데이터의 총 매출액을 조회		
		월별 조회	각 월별에 해당하는 상품 데이터의 총 매출액을 조회		
		상품별 조회	각 상품 데이터의 총 매출액을 조회		

요구사항 정의서 - 관리자

우선순위	분류	소분류	기능	중요도	전제조건
4	게시판 답변 관리	시작일	시작일을 기점으로 그 이후의 데이터만 포함하여 조회할 수 있는 필터	상	관리자페이지에서 답변 관리 페이지 접속
		종료일	종료일을 기점으로 그 이전의 데이터만 포함하여 조회할 수 있는 필터		
		게시판 구분	조회할 게시판 종류를 구분하는 필터		
		제목	제목을 검색할 수 있는 필터		
		작성자	작성자를 검색할 수 있는 필터		
		내용	내용 클릭 시 상세 내용 확인 및 코멘트 달수 있는 항목 표시		
		조회	필터에 맞는 조건들을 선택한 후, 조회버튼 클릭 시 필터에 맞는 데이터들을 조회		
		삭제	게시글 삭제 처리		
5	게시판 등록 관리	시작일	시작일을 기점으로 그 이후의 데이터만 포함하여 조회할 수 있는 필터	중	관리자페이지에서 등록 관리 페이지 접속
		종료일	종료일을 기점으로 그 이전의 데이터만 포함하여 조회할 수 있는 필터		
		게시판 구분	조회할 게시판 종류를 구분하는 필터		
		제목	제목을 검색할 수 있는 필터		
		작성자	작성자를 검색할 수 있는 필터		
		내용	내용 클릭 시 상세 내용 확인 및 코멘트 달수 있는 항목 표시		
		조회	필터에 맞는 조건들을 선택한 후, 조회버튼 클릭 시 필터에 맞는 데이터들을 조회		
		글쓰기	필터에 맞는 게시판에 글 작성		
		수정	선택한 글 제목 및 내용 수정		
		삭제	게시글 삭제 처리		

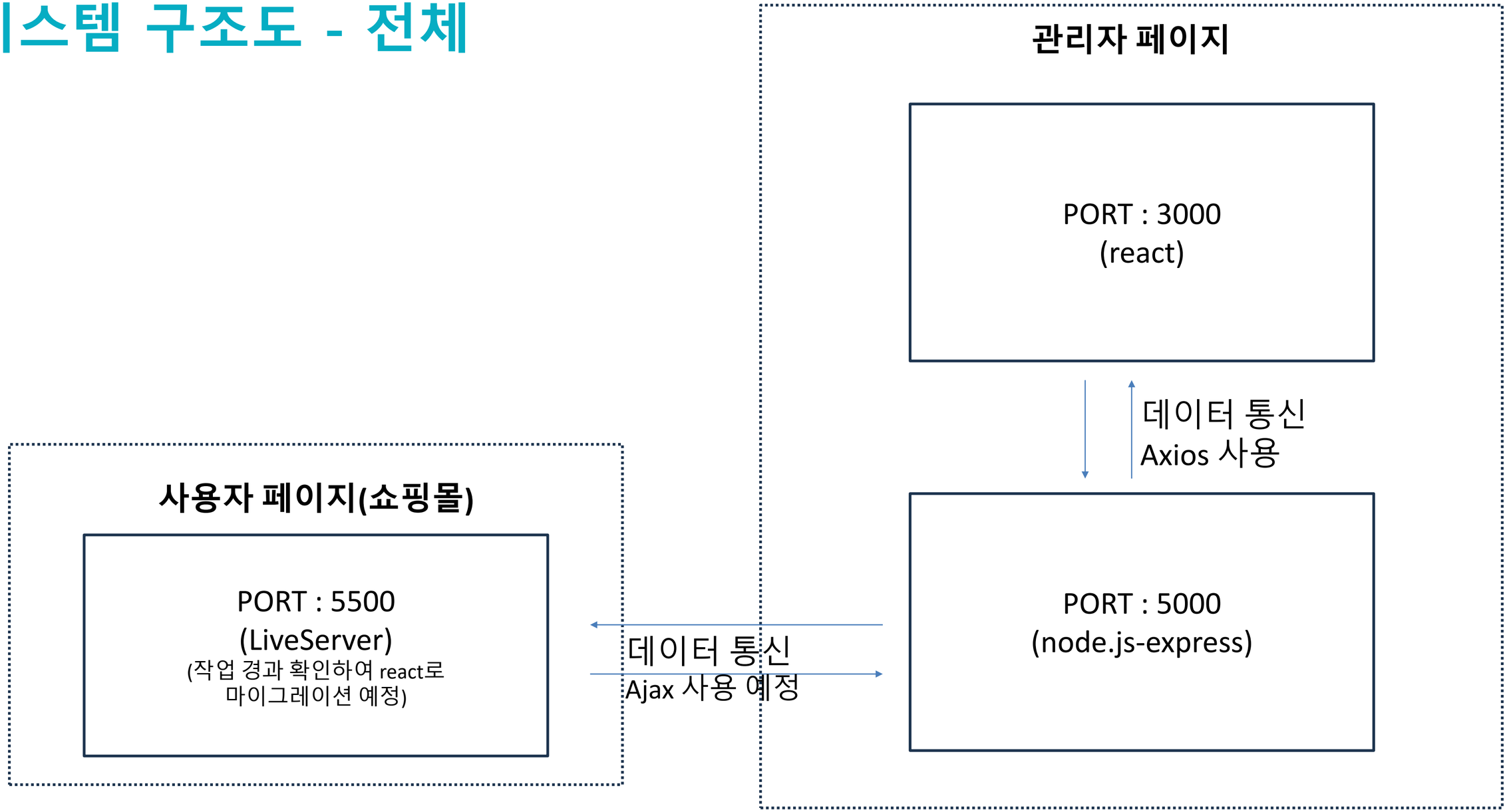
요구사항 정의서 - 관리자

우선순위	분류	소분류	기능	중요도	전제조건
4	회원 관리	등록	회원등록 버튼 클릭시 회원 신규 등록	상	관리자페이지에서 회원 정보 관리 페이지 접속
		내용	회원 정보 클릭시 상세 정보 확인		
		수정	회원 상세 정보 폼에서 수정 버튼 클릭 시 회원 상세 정보 수정		
		삭제	회원 상세 정보 폼에서 삭제 버튼 클릭 시 회원 상세 정보 삭제		
		마일리지 내역 조회	회원 정보 클릭 시 회원의 사용되거나 적립된 마일리지 내역 출력		
		회원 id 조회	회원 아이디를 검색하여 회원 정보를 조회		
		회원 이름 조회	회원 이름을 검색하여 회원 정보를 조회		
		회원 구분 조회	회원 구분별로 검색하여 회원 정보를 조회		
		회원 등급 조회	회원 등급별로 검색하여 회원 정보를 조회		
		가입 시작일 조회	가입 기간으로 검색하여 회원 정보를 조회		
		검색 초기화	초기화 버튼을 클릭 시 검색 필터 초기화 및 회원 데이터 목록 재정렬		

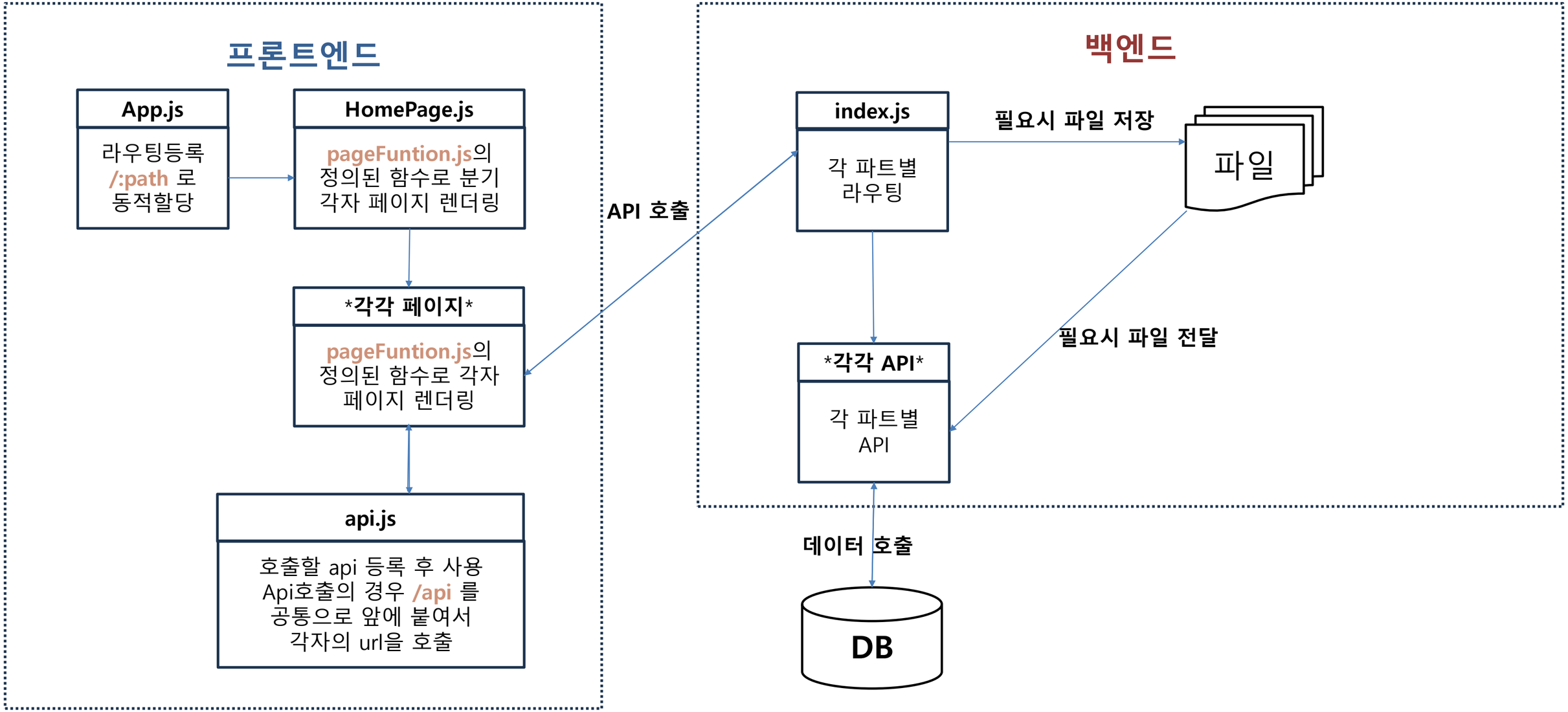
요구사항 정의서 – API

우선순위	분류	소분류	기능	중요도	전제조건
1	지도 api	매장 위치 확인	메인의 고객센터-> 지점으로 이동	중	홈페이지 고객센터 ↓ 지점 ---------- 결제 진행 ↓ 지점 선택 버튼
		매장 위치 검색	검색창 또는 우측 목록과 페이지네이션을 통해 원하는 지점을 선택, 정보와 위치를 확인할 수 있음..		
		배달 지점 선택	장바구니->결제 진행 페이지에서 지점 선택 버튼을 눌러 모달창을 통해 배송받을 지점을 선택 가능		
2	스마트 에디터 api (Quill 스마트에디터)	주문 상세 내역	장바구니에서 보여줬던 상품 정보들을 조회.	중	장바구니 ↓ 상품주문 버튼
		장바구니 가기	장바구니로 연결.		
		주문자 정보	주문자 정보는 고정된 회원정보로 사용.		
		배송 정보	휴대폰 번호, 받으실 분, 받으실 주소 등을 필수적으로 입력하고 유효성 검사를 진행.		
		결제 정보	사용자가 보유한 잔여 마일리지를 표시. 사용자가 사용할 마일리지 양을 입력 시 실시간으로 업데이트. 잔여 마일리지와 결제금액을 변동하여 업데이트. 가지고 있는 양 이상의 마일리지를 사용하려고 할 시 잔여 마일리지만큼만 사용.		
		결제수단 선택	결제 방법을 신용카드, 휴대폰 중에 체크박스로 선택.		
		최종 결제 금액	마일리지 입력 값, 할인율에 따라 값 변동.		
		동의란	체크박스에 체크해야 결제 가능.		
		결제하기	필수입력사항 유효성 검사(이름-한글만/ 주소-한글,영어,숫자만/ 이메일/ 휴대폰번호) 후 주문완료 페이지로 연결. 데이터를 추가. 그 후 주문완료 페이지로 이동.		
3	(카카오) 연동 로그인 api	카카오로그인	클릭 시 필수 동의항목 체크 후 회원가입	상	로그인 ↓ 카카오 로그인 버튼
		회원 아이콘 클릭	클릭 시 가입한 카카오 계정 이름으로 알림창.		

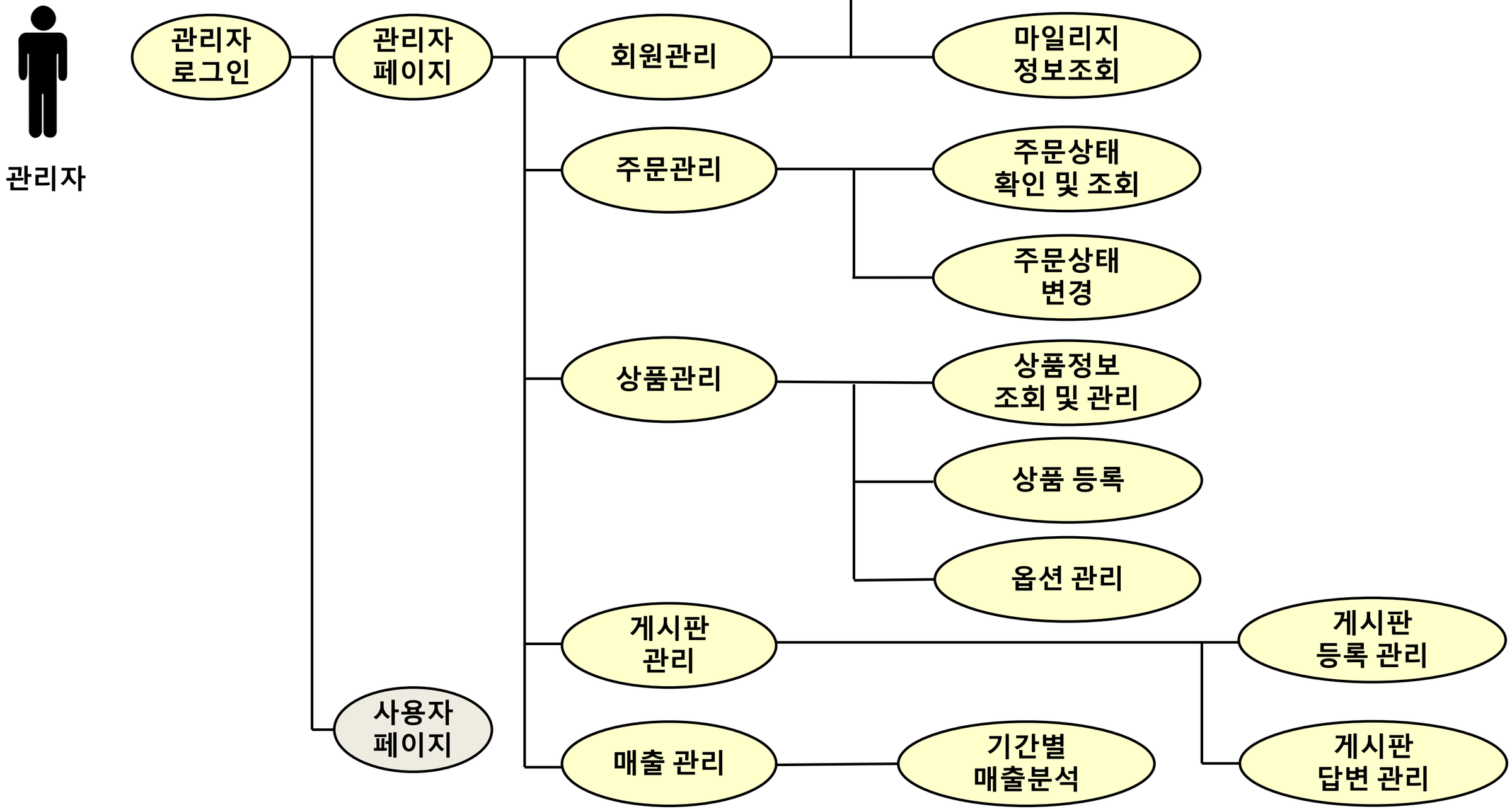
시스템 구조도 - 전체


관리자 페이지
PORT : 3000
(react)
데이터 통신
Axios 사용
PORT : 5000
(node.js-express)
사용자 페이지(쇼핑몰)
PORT : 5500
(LiveServer)
(작업 경과 확인하여 react로
마이그레이션 예정)
데이터 통신
Ajax 사용 예정

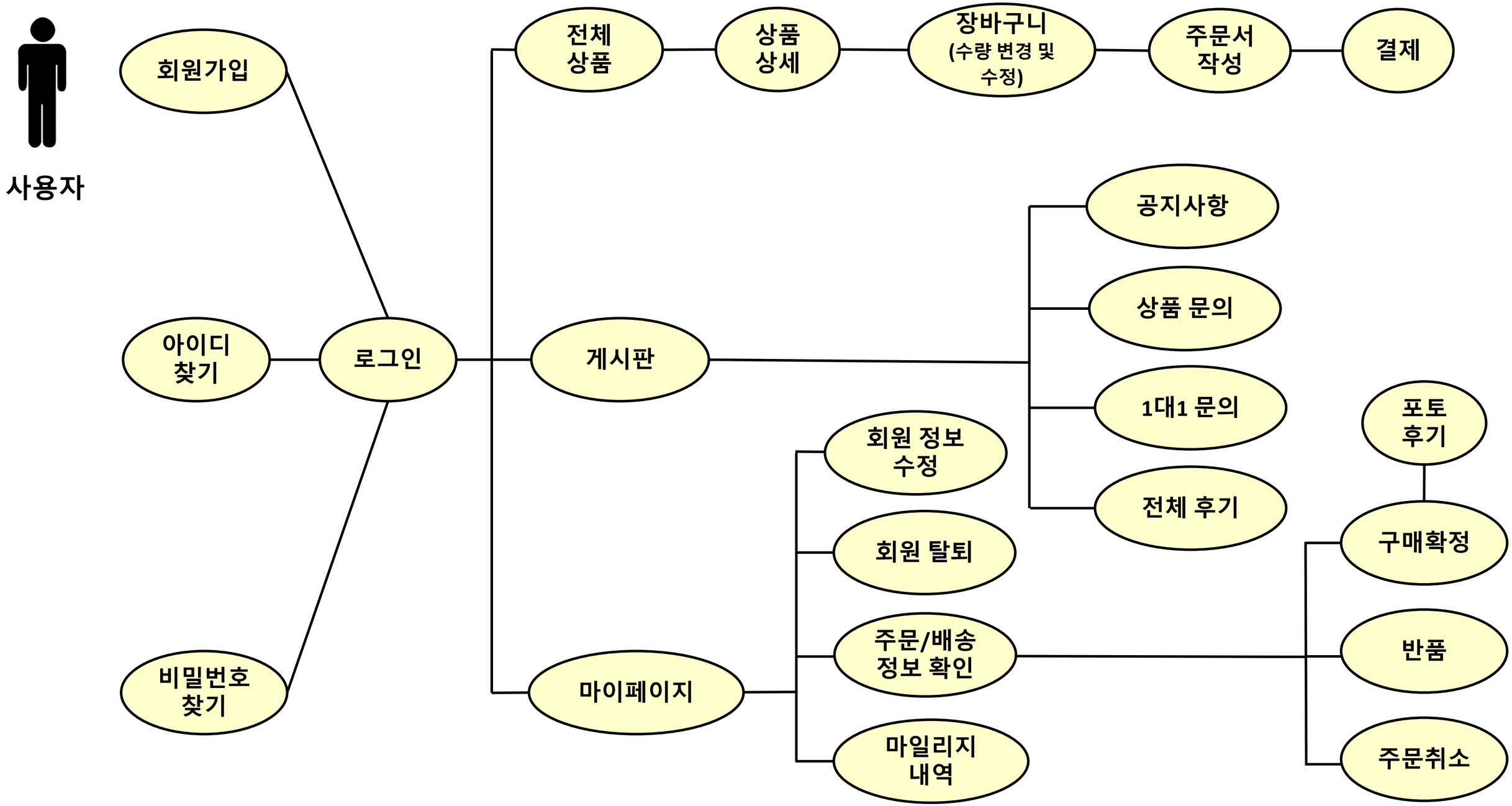
시스템 구조도 - 관리자 시스템



Use Case - 관리자


관리자
관리자 로그인
관리자 페이지
사용자 페이지
회원관리
회원정보 조회 및 관리
마일리지 정보조회
주문관리
주문상태 확인 및 조회
주문상태 변경
상품관리
상품정보 조회 및 관리
상품 등록
옵션 관리
게시판 관리
게시판 등록 관리
게시판 답변 관리
매출 관리
기간별 매출분석

Use Case - 사용자


사용자
회원가입
아이디 찾기
비밀번호 찾기
로그인
전체 상품
상품 상세
장바구니 (수량 변경 및 수정)
주문서 작성
결제
게시판
공지사항
상품 문의
1대1 문의
전체 후기
마이페이지
회원 정보 수정
회원 탈퇴
주문/배송 정보 확인
마일리지 내역
구매확정
포토 후기
반품
주문취소

ERD - 전체

주문 orders
주문번호 order_id INT
회원ID member_id INT
주문일자 order_date DATETIME
결제수단 pay_method ENUM('휴대폰','신용카드')
마일리지 순서 used_mileage INT
구매확정여부 is_confirmed CHAR(1)
요청사유 req_reasons VARCHAR(255)
요청거절사유 refuse_reasons VARCHAR(255)
작성자 reg_id VARCHAR(50)
작성일시 reg_date DATETIME
수정자 upt_id VARCHAR(50)
수정일시 upt_date DATETIME
상품별 주문 order_products
주문상품Id order_product_id INT
주문번호 order_id INT
상품 번호 product_id INT
상품 수량 quantity INT
상품 단가 unit_price INT
작성자 reg_id VARCHAR(50)
작성일시 reg_date DATETIME
수정자 upt_id VARCHAR(50)
수정일시 upt_date DATETIME
배송 deliveries
배송 번호 delivery_id INT
주문번호 order_id INT
회원ID member_id INT
주문 상태 order_status ENUM('배송준비','배송시작','배송완료','취소요청','반품요청','교환요청','취소완료','반품완료','교환완료')
배송준비날짜 prep_date DATETIME
배송시작날짜 start_date DATETIME
배송완료날짜 done_date DATETIME
취소요청날짜 cancel_req_date DATETIME
반품요청날짜 return_req_date DATETIME
교환요청날짜 exchange_req_date DATETIME
취소완료날짜 cancel_done_date DATETIME
반품완료날짜 return_done_date DATETIME
교환완료날짜 exchange_done_date DATETIME
작성자 reg_id VARCHAR(50)
작성일시 reg_date DATETIME
수정자 upt_id VARCHAR(50)
수정일시 upt_date DATETIME
상품 products
상품Id id VARCHAR(14)
상품명 name VARCHAR(100)
가격 price INT
할인율 discount INT
상세이미지 detail_exist BOOLEAN
등록재고수량 quan INT
사용여부 status BOOLEAN
혜택 benefit TEXT
등록자 reg_id VARCHAR(60)
등록일시 reg_date DATETIME
수정자 upt_id VARCHAR(60)
수정일시 upt_date DATETIME
상품 디테일 product_detail
하위값고유코드 code INT
코드 main_code INT
상품Id id VARCHAR(14)
순서 turn INT(2)
옵션키워드구분 type CHAR(1)
등록자 reg_id VARCHAR(60)
등록일시 reg_date DATETIME
수정자 upt_id VARCHAR(60)
수정일시 upt_date DATETIME
옵션키워드하위값 opKey_value
하위값고유코드 code INT
코드 main_code INT
구분 type BOOLEAN
값 명칭 name VARCHAR(100)
값 가격 price INT
사용여부 status TINYINT(1)
등록자 reg_id VARCHAR(60)
등록일시 reg_date DATETIME
수정자 upt_id VARCHAR(60)
수정일시 upt_date DATETIME
옵션키워드항목 option_keyword
코드 code INT
구분 type BOOLEAN
명칭 name VARCHAR(100)
필수선택여부 req BOOLEAN
사용여부 status BOOLEAN
등록자 reg_id VARCHAR(60)
등록일시 reg_date DATETIME
수정자 upt_id VARCHAR(60)
수정일시 upt_date DATETIME
cart cart
cart_id cart_id INT
product_id product_id INT
member_id member_id VARCHAR(50)
quantity quantity INT
reg_id reg_id VARCHAR(50)
reg_date reg_date DATETIME
upt_id upt_id VARCHAR(50)
upt_date upt_date DATETIME
파일관리 file_manage
고유Id id VARCHAR(14)
순서 turn INT(2)
요청 url api_url VARCHAR(50)
원본 파일명 org_name VARCHAR(100)
변경 파일명 new_name VARCHAR(255)
확장자명 ext VARCHAR(10)
저장 경로 path VARCHAR(255)
파일 크기 size BIGINT
설명 dsct TEXT
등록자 reg_id VARCHAR(60)
등록일자 reg_date DATETIME
수정자 upt_id VARCHAR(60)
수정일자 upt_date DATETIME
mem_info mem_info
회원Id id VARCHAR(50)
비밀번호 pw VARCHAR(255)
회원명 name VARCHAR(100)
닉네임 nick VARCHAR(50)
생년월일 birth_date DATE
휴대전화 phone_num VARCHAR(20)
이메일 email VARCHAR(100)
주소 addr VARCHAR(255)
접근권한 role ENUM('관리자','회원','비회원')
회원구분 type ENUM('일반','블랙리스트','휴면계정')
회원등급 level ENUM('브론즈','실버','골드','VIP')
가입일시 join_date DATETIME
최근접속일 last_date TIMESTAMP
잔여마일리지 m_remain INT
작성자 reg_id VARCHAR(60)
작성일시 reg_date DATETIME
수정자 upt_id VARCHAR(60)
수정일시 upt_date DATETIME
마일리지 mileage
순서 turn INT
회원Id id VARCHAR(50)
적립날짜 earn_date DATETIME
적립내용 description TEXT
유효날짜 valid_date DATE
변동내역 change_val INT
작성자 reg_id VARCHAR(60)
작성일시 reg_date DATETIME
수정자 upt_id VARCHAR(60)
수정일시 upt_date DATETIME
게시판 board
고유Id id2 VARCHAR(14)
분류코드 code CHAR(2)
제목 title VARCHAR(100)
내용 contents TEXT
첨부이미지코드 img_no VARCHAR(14)
상품ID products_id VARCHAR(14)
별점 score INT(1)
분류 division TINYINT
조회수 read_no INT(3)
등록자 reg_id VARCHAR(60)
등록일시 reg_date DATETIME
수정자 upt_id VARCHAR(60)
수정일시 upt_date DATETIME
답변 answer
고유Id id VARCHAR(14)
순번 turn INT(3)
코멘트 comment text
delete_yn delete_yn TINYINT(1)
등록자 reg_id VARCHAR(60)
등록일시 reg_date DATETIME
수정자 upt_id VARCHAR(60)
수정일시 upt_date DATETIME
정성윤
우수정
임세훈
김수훈

ERD – 회원 관리

mem_info			mem_info
회원id	id	Domain	VARCHAR(50)
비밀번호	pw	Domain	VARCHAR(255)
회원명	name	Domain	VARCHAR(100)
닉네임	nick	Domain	VARCHAR(50)
생년월일	birth_date	Domain	DATE
휴대전화	phone_num	Domain	VARCHAR(20)
이메일	email	Domain	VARCHAR(100)
주소	addr	Domain	VARCHAR(255)
접근권한	role	Domain	ENUM('관리자','회원','비회원')
회원구분	type	Domain	ENUM('일반','블랙리스트','휴면계정')
회원등급	level	Domain	ENUM('브론즈','실버','골드','VIP')
가입일시	join_date	Domain	DATETIME
최근접속일	last_date	Domain	TIMESTAMP
잔여마일리지	m_remain	Domain	INT
작성자	reg_id	Domain	VARCHAR(60)
작성일시	reg_date	Domain	DATETIME
수정자	upt_id	Domain	VARCHAR(60)
수정일시	upt_date	Domain	DATETIME

회원 상세 정보를 담은 테이블

ERD – 회원 관리

마일리지			mileage
순서	turn	Domain	INT
회원id	id	Domain	VARCHAR(50)
적립날짜	earn_date	Domain	DATETIME
적립내용	description	Domain	TEXT
유효날짜	valid_date	Domain	DATE
변동내역	change_val	Domain	INT
작성자	reg_id	Domain	VARCHAR(60)
작성일시	reg_date	Domain	DATETIME
수정자	upt_id	Domain	VARCHAR(60)
수정일시	upt_date	Domain	DATETIME

회원 ID를 참조하여
회원 마일리지 정보를 담은 테이블

ERD – 주문/배송

주문			orders
주문번호	order_id	Domain	INT
회원ID	member_id	Domain	INT
주문일자	order_date	Domain	DATETIME
결제수단	pay_method	Domain	ENUM('휴대폰','신용카드')
마일리지 순서	used_mileage	Domain	INT
구매확정여부	is_confirmed	Domain	CHAR(1)
요청사유	req_reasons	Domain	VARCHAR(255)
요청거절사유	refuse_reasons	Domain	VARCHAR(255)
작성자	reg_id	Domain	VARCHAR(50)
작성일시	reg_date	Domain	DATETIME
수정자	upt_id	Domain	VARCHAR(50)
수정일시	upt_date	Domain	DATETIME

회원ID를 참조하여
주문 정보를 담은 테이블

상품별 주문			order_products
주문상품id	order_product_id	Domain	INT
주문번호	order_id	Domain	INT
상품 번호	product_id	Domain	INT
상품 수량	quantity	Domain	INT
상품 단가	unit_price	Domain	INT
작성자	reg_id	Domain	VARCHAR(50)
작성일시	reg_date	Domain	DATETIME
수정자	upt_id	Domain	VARCHAR(50)
수정일시	upt_date	Domain	DATETIME

주문번호와 상품번호를 참조하여
주문별 상품 정보를 담은 테이블

ERD – 주문/배송

장바구니			cart
장바구니ID	cart_id	Domain	INT
상품ID	product_id	Domain	INT
회원ID	member_id	Domain	VARCHAR(50)
갯수	quantity	Domain	INT
작성자	reg_id	Domain	VARCHAR(50)
작성일시	reg_date	Domain	DATETIME
수정자	upt_id	Domain	VARCHAR(50)
수정일시	upt_date	Domain	DATETIME

회원 ID와 상품ID를 참조하여
장바구니 정보를 담은 테이블

ERD – 주문/배송

배송			deliveries
배송 번호	delivery_id	Domain	INT
주문번호	order_id	Domain	INT
회원ID	member_id	Domain	INT
주문 상태	order_status	Domain	ENUM('배송준비','배송시작','배송완료','취소요청','반품요청','교환요청','취소완료','반품완료','교환완료')
배송준비날짜	prep_date	Domain	DATETIME
배송시작날짜	start_date	Domain	DATETIME
배송완료날짜	done_date	Domain	DATETIME
취소요청날짜	cancel_req_date	Domain	DATETIME
반품요청날짜	return_req_date	Domain	DATETIME
교환요청날짜	exchange_req_date	Domain	DATETIME
취소완료날짜	cancel_done_date	Domain	DATETIME
반품완료날짜	return_done_date	Domain	DATETIME
교환완료날짜	exchange_done_date	Domain	DATETIME
작성자	reg_id	Domain	VARCHAR(50)
작성일시	reg_date	Domain	DATETIME
수정자	upt_id	Domain	VARCHAR(50)
수정일시	upt_date	Domain	DATETIME

상품별 주문 테이블의 주문번호
회원 정보 테이블의 회원 ID를 참조하여
배송 정보를 담은 테이블

ERD – 상품/옵션

상품		products	
상품id	id	Domain	VARCHAR(14)
상품명	name	Domain	VARCHAR(100)
가격	price	Domain	INT
할인율	discount	Domain	INT
상세이미지	detail_exist	Domain	BOOLEAN
등록재고수량	quan	Domain	INT
사용여부	status	Domain	BOOLEAN
혜택	benefit	Domain	TEXT
등록자	reg_id	Domain	VARCHAR(60)
등록일시	reg_date	Domain	DATETIME
수정자	upt_id	Domain	VARCHAR(60)
수정일시	upt_date	Domain	DATETIME

상품 ID 포함
상품 정보를 담은 테이블

상품 디테일		product_detail	
하위값고유코드	code	Domain	INT
코드	main_code	Domain	INT
상품id	id	Domain	VARCHAR(14)
순서	turn	Domain	INT(2)
옵션키워드구분	type	Domain	CHAR(1)
등록자	reg_id	Domain	VARCHAR(60)
등록일시	reg_date	Domain	DATETIME
수정자	upt_id	Domain	VARCHAR(60)
수정일시	upt_date	Domain	DATETIME

상품 ID를 참조하여
상품 상세 정보를 담은 테이블

ERD – 상품/옵션

옵션키워드항목			option_keyword
코드	code	Domain	INT
구분	type	Domain	BOOLEAN
명칭	name	Domain	VARCHAR(100)
필수선택여부	req	Domain	BOOLEAN
사용여부	status	Domain	BOOLEAN
등록자	reg_id	Domain	VARCHAR(60)
등록일시	reg_date	Domain	DATETIME
수정자	upt_id	Domain	VARCHAR(60)
수정일시	upt_date	Domain	DATETIME

상품에 관한 옵션 항목정보를
담고 있는 테이블

옵션키워드하위값			opKey_value
하위값고유코드	code	Domain	INT
코드	main_code	Domain	INT
구분	type	Domain	BOOLEAN
값 명칭	name	Domain	VARCHAR(100)
값 가격	price	Domain	INT
사용여부	status	Domain	TINYINT(1)
등록자	reg_id	Domain	VARCHAR(60)
등록일시	reg_date	Domain	DATETIME
수정자	upt_id	Domain	VARCHAR(60)
수정일시	upt_date	Domain	DATETIME

옵션 키워드 항목을 참조하여
옵션의 하위옵션 정보 테이블

ERD – 게시판

게시판			board
고유id	id2	Domain	VARCHAR(14)
분류코드	code	Domain	CHAR(2)
제목	title	Domain	VARCHAR(100)
내용	contents	Domain	TEXT
첨부이미지코드	img_no	Domain	VARCHAR(14)
상품ID	products_id	Domain	VARCHAR(14)
별점	score	Domain	INT(1)
분류	division	Domain	TINYINT
조회수	read_no	Domain	INT(3)
등록자	reg_id	Domain	VARCHAR(60)
등록일시	reg_date	Domain	DATETIME
수정자	upt_id	Domain	VARCHAR(60)
수정일시	upt_date	Domain	DATETIME

게시판 고유 ID를 포함한
게시판 정보를 담은 테이블

답변			answer
고유id	id	Domain	VARCHAR(14)
순번	turn	Domain	INT(3)
코멘트	comment	Domain	text
delete_yn	delete_yn	Domain	TINYINT(1)
등록자	reg_id	Domain	VARCHAR(60)
등록일시	reg_date	Domain	DATETIME
수정자	upt_id	Domain	VARCHAR(60)
수정일시	upt_date	Domain	DATETIME

게시판 고유 ID를 참조하는
게시판 답변 테이블

ERD – 게시판

파일관리			file_manage
고유id	id	Domain	VARCHAR(14)
순서	turn	Domain	int(2)
요청 url	api_url	Domain	VARCHAR(50)
원본 파일명	org_name	Domain	VARCHAR(100)
변경 파일명	new_name	Domain	VARCHAR(255)
확장자명	ext	Domain	VARCHAR(10)
저장 경로	path	Domain	VARCHAR(255)
파일 크기	size	Domain	BIGINT
설명	dsct	Domain	TEXT
등록자	reg_id	Domain	VARCHAR(60)
등록일자	reg_date	Domain	DATETIME
수정자	upt_id	Domain	VARCHAR(60)
수정일자	upt_date	Domain	DATETIME

게시판에 업로드 되는
파일관리 테이블

역할분담

게시판

정성윤

- 관리자
  공지, FAQ 글 작성, 전체후기, 포토후기 글 삭제 전환, 상품문의, 1:1 문의 답변작성
- 사용자
  글 읽기, 전체후기, 포토후기, 상품문의, 1:1문의 글 작성 및 삭제 및 읽기와 답변 여부 확인

주문 및 배송, API

임세훈

- 관리자
  주문, 환불, 정산조회 및 취소, 환불 요청에 대한 승인, 거절 기능
  상태, 날짜(일, 주, 월별) 결제수단, 상품별 조회 기능
- 사용자
  장바구니페이지, 주문페이지, 주문완료페이지, 마이페이지
  주문데이터 동적 이동 및 옵션값과 옵션추가금 반영
  취소, 반품요청, 구매확정, 후기등록 등의 기능, 마일리지 변동량 확인가능
  (카카오)지도 api, 현재 위치기반 매장 표시 및 선택

역할분담

상품관리

우수정

- 관리자
  상품 이름, 이미지, 사용 옵션/키워드 설정 등록, 재고 등록, 상품 가격, 할인율 자동계산
  후 최종 가격 반영. 등록된 상품 목록 조회
- 사용자
  등록된 상품 목록 수, 상품 정보 조회, 옵션값 추가금, 정보 반영 후 장바구니로 전달

회원관리

김수훈

- 관리자
  회원 등록, 회원정보 조회(회원 ID, 이름, 구분, 등급, 가입날짜 별), 수정, 삭제 및 마일리지 내역 조회
- 사용자
  회원가입, 로그인, 로그아웃, 마일리지 상세내역 조회, 회원정보 수정,
  아이디 및 비밀번호 찾기, 회원 탈퇴기능

시간계획서01

		11월													12월									
우선순위	기능	13	14	15	18	19	20	21	22	25	26	27	28	29	2	3	4	5	6	9	10	11	12	13
1	주제선정																						테스트	완료
2	파트분배																							
3	요구사항																							
4	사이트분석																							
5	Use Case																							
6	ERD																							
7	공지사항																							
7	FAQ																							
8	전체후기																							
9	포토후기																							
10	상품문의																							
11	1대1문의																							
12	상품 조회																							
13	상품 등록																							
14	상품 수정																							
15	메인 페이지																							
16	상품 리스트																							
17	상품 상세																							

시간계획서02

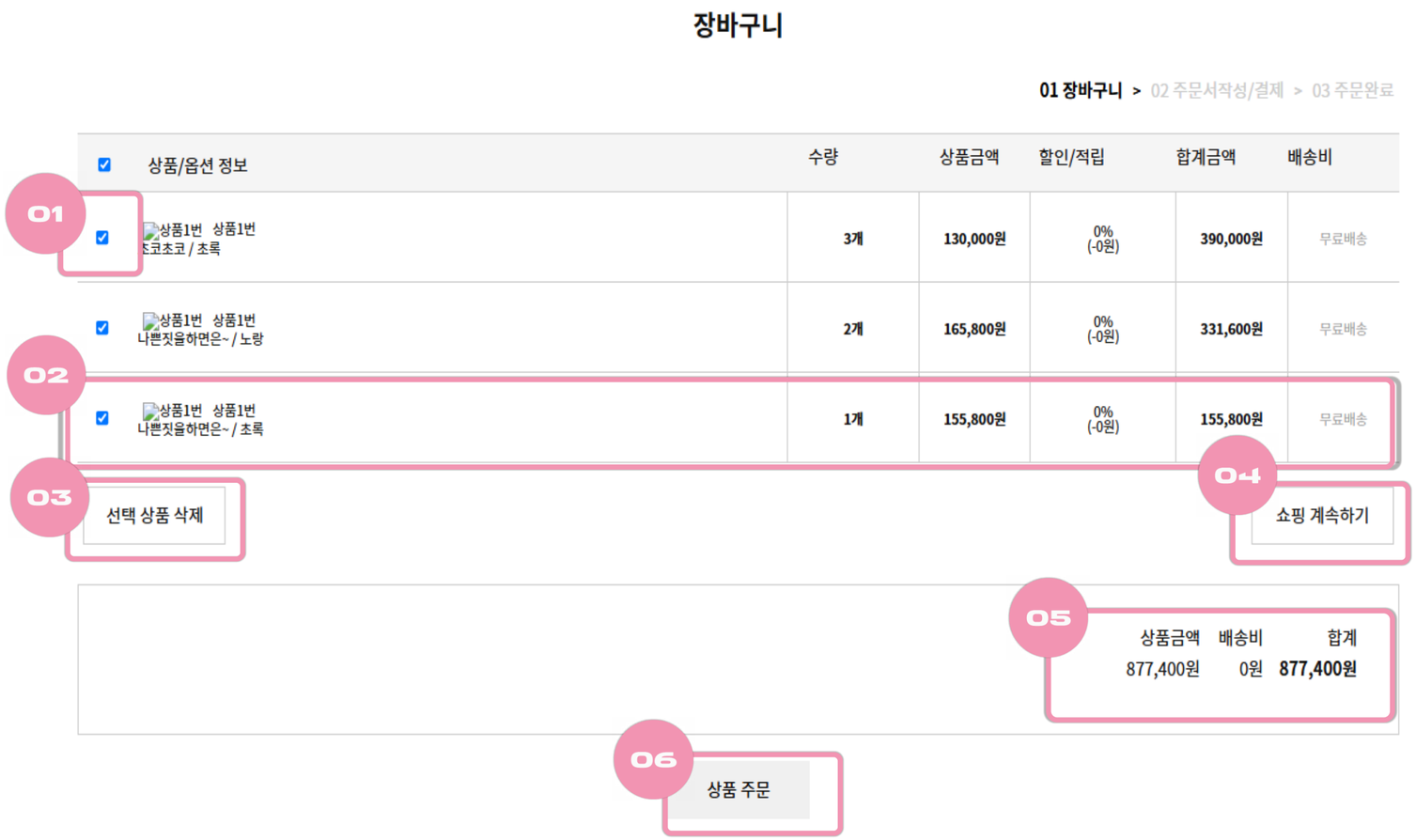
		11월													12월									
우선순위	기능	13	14	15	18	19	20	21	22	25	26	27	28	29	2	3	4	5	6	9	10	11	12	13
18	상품 옵션관리																						테스트	완료
19	상품 키워드 관리																							
20	주문 조회																							
21	환불 조회																							
22	환불 수정																							
23	정산 관리																							
24	마이페이지																							
25	결제																							
26	환불																							
27	장바구니																							
28	회원 관리																							
29	로그인/로그아웃																							
30	ID/PW 찾기																							
31	회원 가입																							
32	회원 정보 수정																							
33	회원 탈퇴																							

시간계획서03

		1월											
우선순위	기능	2	3	4	5	6	7	8	9	10	11	12	13
2	지도api											테스트	마무리
	지점설정 및 검색												
3	스마트 에디터 api												
	에디터 내 추가 기능												
1	카카오 로그인 구현												
	사용자 정보 관리 및 인증												

반응형 웹/앱
가구 쇼핑몰 시연 화면

01장바구니



Description List

01 체크박스를 통해 상품선택 가능

02 데이터를 통해 불러온 장바구니 상품정보

03 선택한 상품 삭제 기능

04 전체상품 페이지로 이동

05 각 금액을 자동으로 합산해서 표시함

06 주문으로 이동

01장바구니

01

```
try {
  // 옵션 데이터를 파싱 (JSON 문자열인지 확인)
  const parsedOptions = typeof cart.options === 'string'
    ? JSON.parse(cart.options) // JSON 문자열일 경우 파싱
    : cart.options; // 객체일 경우 그대로 사용

  // 옵션 값을 보기 좋은 형식으로 표시
  if (parsedOptions && typeof parsedOptions === 'object') {
    return Object.values(parsedOptions).join(' / '); // 값만 표시
  } else {
    return "옵션: 없음";
  }
} catch (error) {
  console.error("옵션 처리 오류:", error);
  return "옵션 데이터를 표시할 수 없습니다.";
}
})()}
```

02

```
const insertCartQuery = `
INSERT INTO cart (member_id, product_id, quantity, options, reg_id, option_unit_price)
VALUES (?, ?, ?, ?, ?,?)
`;
```

Description List

01 옵션값을 장바구니에 데이터마다 표시

02 장바구니 테이블에 데이터를 추가하는 쿼리

0 2 주문

주문서작성/결제

주문상세내역

01 장바구니 > 02 주문서작성/결제 > 03 주문완료

상품/옵션 정보	수량	상품금액	할인/적립	합계금액
상품1번 초코초코 / 초록	3개	100,000원	0%	390,000원
상품1번 나쁜짓을하면은~ / 노랑	2개	100,000원	0%	331,600원
상품1번 나쁜짓을하면은~ / 초록	1개	100,000원	0%	155,800원

01

02

장바구니 가기

3 종류의 총 상품금액 877,400원 + 배송비 0원

03

합계 877,400원

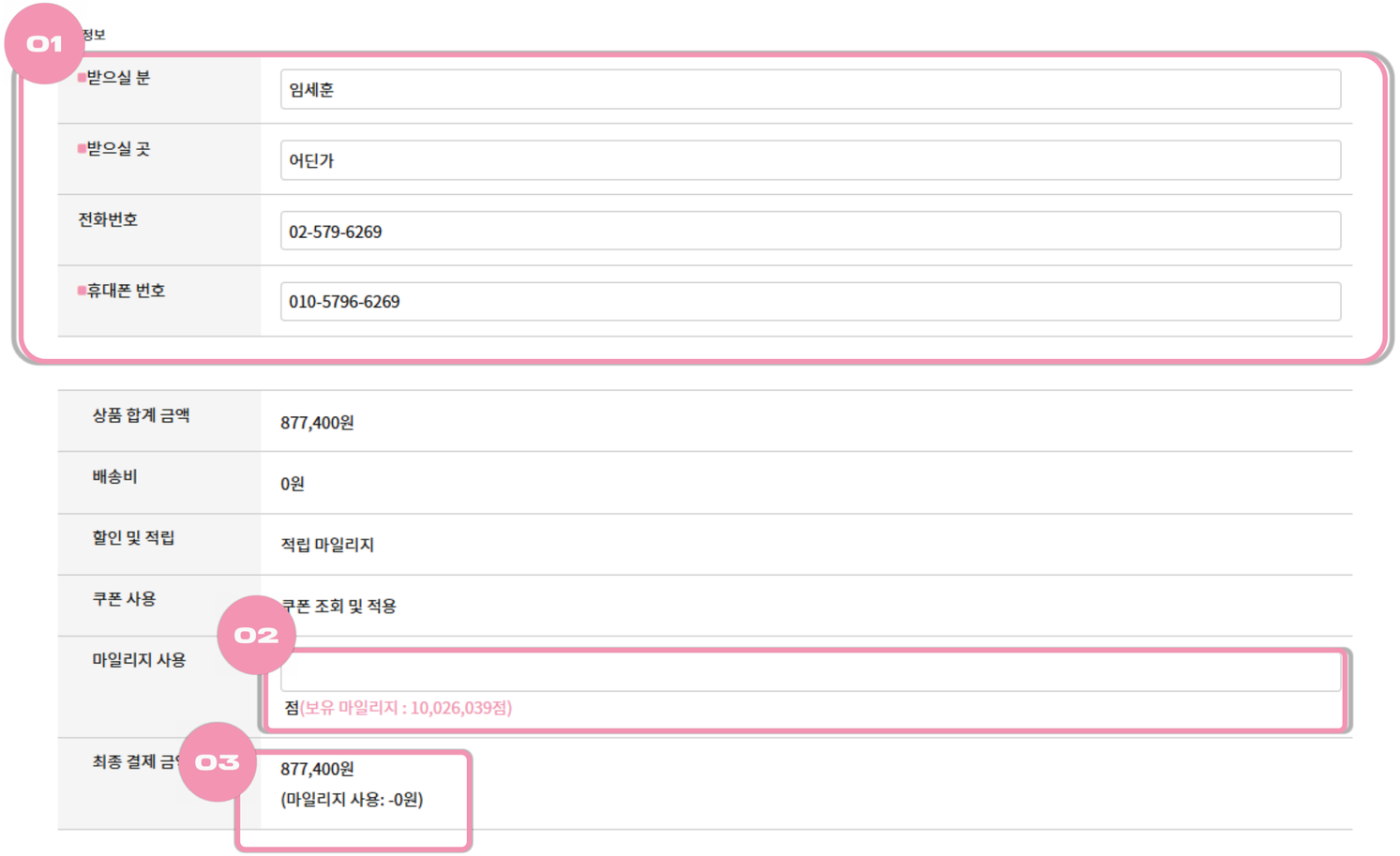
Description List

01 데이터를 통해 받아온 주문정보

02 장바구니로 이동

03 주문의 총금액을 자동합산

0 2 주문



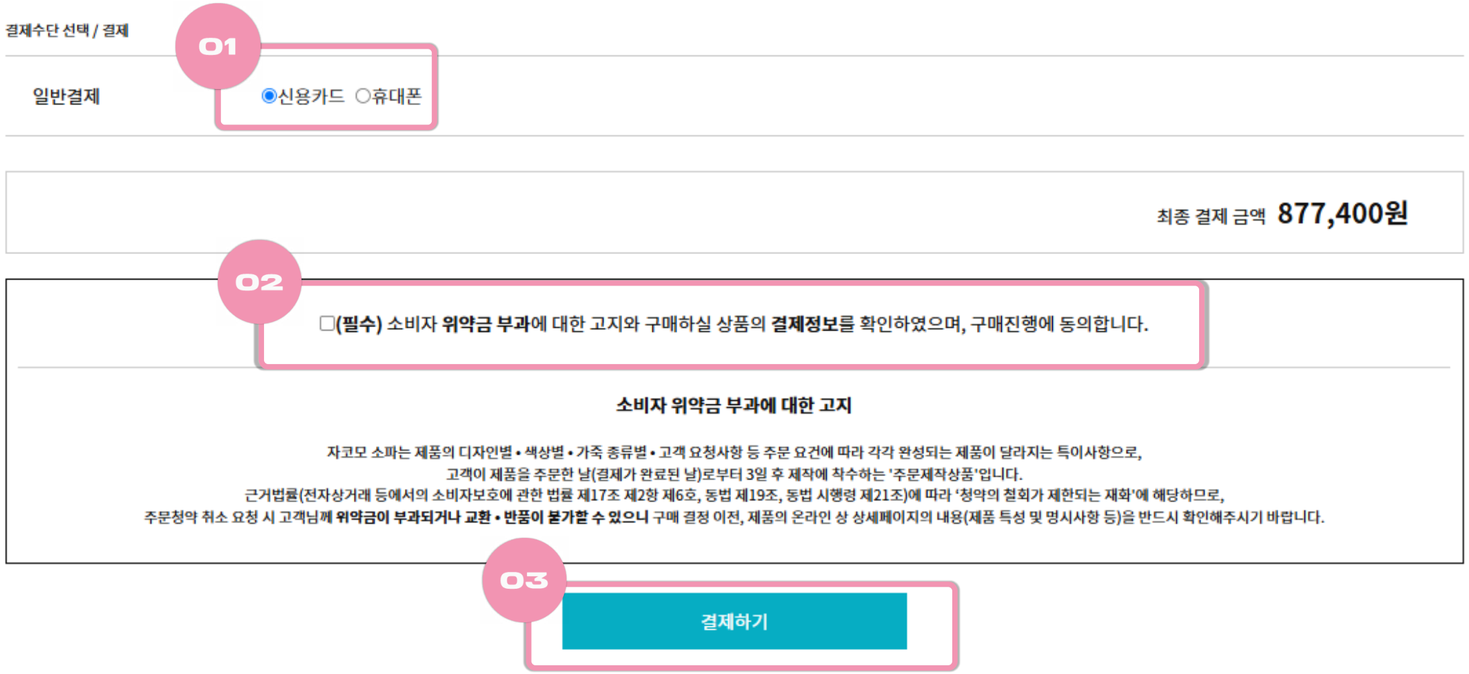
Description List

01 배송정보를 입력하면 DB에 저장

02 보유 마일리지를 조회하며 보유한 양 이하로 사용가능

03 사용한 마일리지 양만큼 총 결제금액이 감소됨

0 2 주문



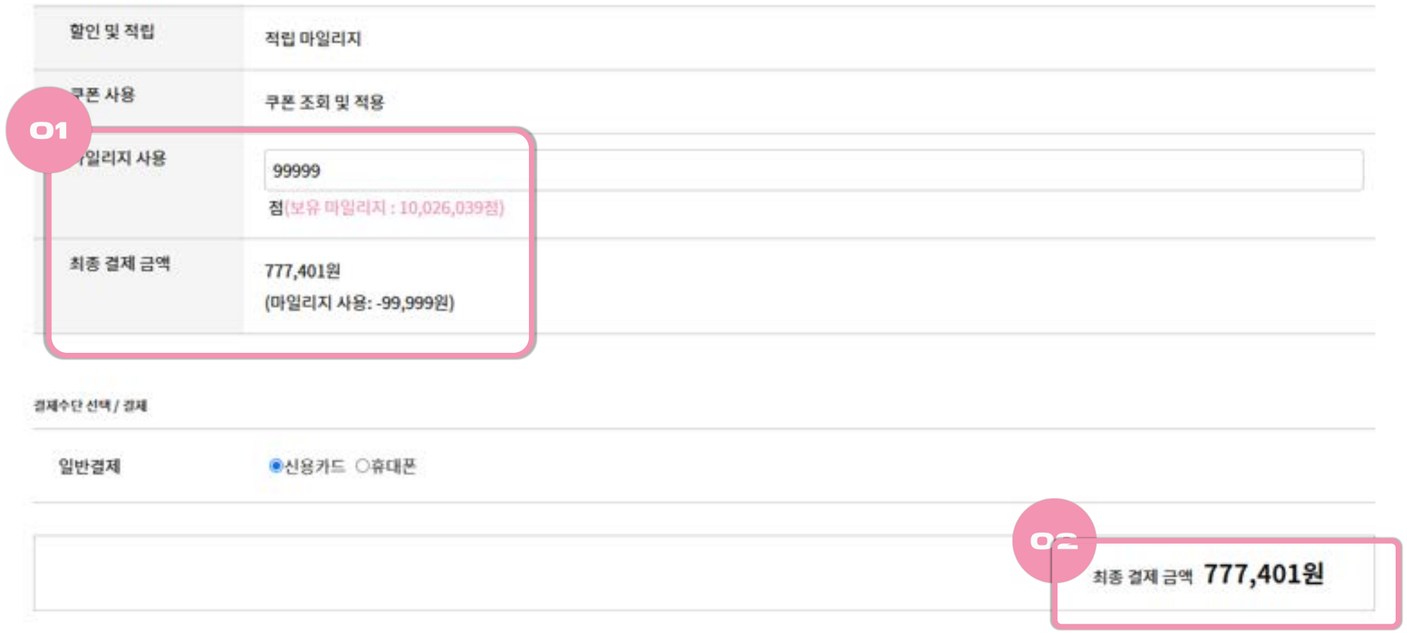
Description List

01 결제수단 신용카드,휴대폰 선택가능

02 약관동의 유효성

03 주문완료로 이동

0 2 주문



Description List

01 사용된 마일리지 양 표시

02 사용한 양만큼 결제금액 감소

0 2 주문

01

```
const Order = () => {
  const [formData, setFormData] = useState({
    ordererName: "임세훈",
    ordererPhone: "01057966269",
    ordererMobile: "01057966269",
    ordererEmail: "sehoon@naver.com",
    recipientName: "임세훈",
    recipientAddress: "어딘가",
    recipientPhone: "02-579-6269",
    recipientMobile: "010-5796-6269",
    recipientComment: "",
    mileage: "",
    deposit: "",
    agree: false,
    items: [],
    total_price: 0,
    pay_method: "신용카드",
  });


  const [errors, setErrors] = useState({}); // 유효성 에러 상태 추가

  const userId = sessionStorage.getItem('userId'); //  로그인한 사용자 ID 가져오기
```

Description List

01 주문페이지의 각 STATE와 기본값 설정

0 2 주문

01

```
<li>
  <input
    type="radio"
    name="pay_method"
    value="신용카드"
    checked={formData.pay_method === '신용카드'}
    onChange={(e) => setFormData({ ...formData, pay_method: e.target.value })}
  />
  <label>신용카드</label>
</li>
<li>
  <input
    type="radio"
    name="pay_method"
    value="휴대폰"
    checked={formData.pay_method === '휴대폰'}
    onChange={(e) => setFormData({ ...formData, pay_method: e.target.value })}
  />
  <label>휴대폰</label>
</li>
```

02

```
<tr>
  <th>전화번호</th>
  <td>
    <input
      type="text"
      name="recipientPhone"
      value={formData.recipientPhone}
      onChange={handleInputChange}
      className={styles.input220}
    />
  </td>
</tr>
```

Description List

01 결제수단 선택 및 체크박스 관련코드

02 배송정보입력 관련코드

0 2 주문

01

```
export const rFetchOrderInfo = (orderId, userId) => API.get(`/order/${orderId}`, {
    headers: { 'user-id': userId }
  });
```

02

```
//  useEffect로 주문 정보 가져오기
useEffect(() => {
  const fetchOrderInfo = async () => {
    try {
      const { data } = await rFetchOrderInfo(orderId, userId); //  orderId로 요청

      setOrderInfo(data || {});

    } catch (error) {
      setError("주문 정보를 불러오는 중 오류가 발생했습니다.");
    } finally {
      setIsLoading(false);
    }
  };

  if (orderId) {
    fetchOrderInfo();
  }
}, [orderId, userId]);
```

Description List

01 주문정보를 요청하는 서버에 요청하는 코드

02 주문정보를 받아오는 코드

0 3 주문완료

Description List

01 데이터에 저장된 주문정보를 동적으로 불러옴

02 확인버튼 메인으로 이동

0 3 주문완료

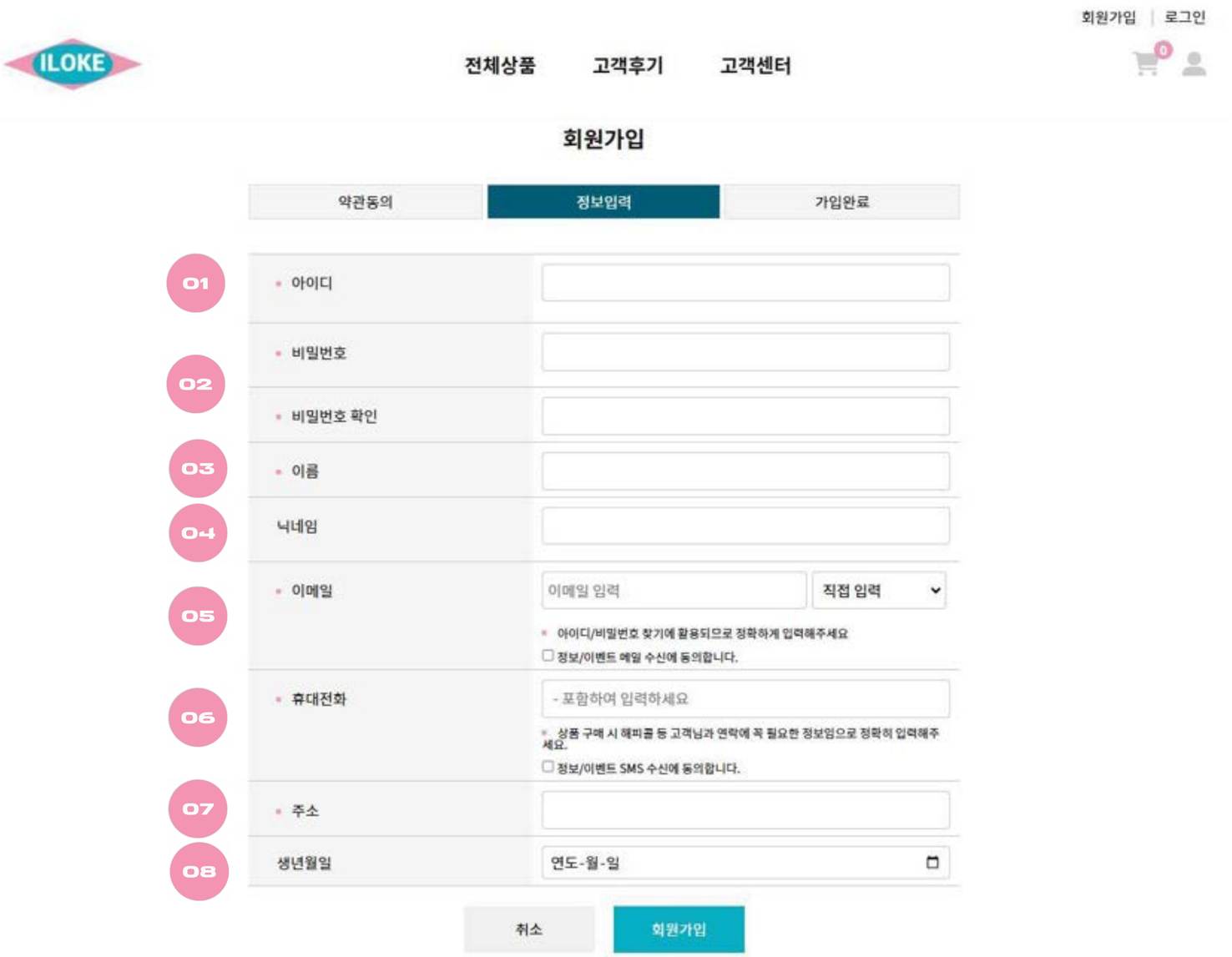
01

```
            <tr>
              <th>주문번호</th>
              <td>{orderInfo?.order_id}</td>
            </tr>
            <tr>
              <th>주문일자</th>
              <td>{orderInfo?.order_date}</td>
            </tr>
            <tr>
              <th>주문상품</th>
              <td>{orderInfo?.product_summary}</td>
            </tr>
            <tr>
              <th>주문자명</th>
              <td>{orderInfo?.orderer_name}</td>
            </tr>
            <tr>
              <th>배송정보</th>
              <td>
              <strong>수령자:</strong> {orderInfo?.receiver_name}
  <br />
  <strong>주소:</strong> {orderInfo?.receiver_address}
  <br />
  <strong>핸드폰 번호:</strong> {orderInfo?.receiver_phone}
</td>
            </tr>
```

Description List

01 데이터에 저장된 각종 주문정보를 동적으로 받아오는 코드

0 4 . 회원가입 정보입력



Description List

01 아이디

02 비밀번호 및 비밀번호 확인

03 이름

04 닉네임(선택사항)

05 이메일

06 휴대전화

07 주소

08 생년월일(선택사항)

0 4 . 회원가입 정보입력

Description List

01 회원가입 버튼

02 메인 페이지로 이동

03 아이디 미 입력 시 또는 조건에 맞지 않게 입력 시 alert 출

0 4 . 회원가입 정보입력

01

02

Description List

01 아이디 유효성 검사
(숫자, 영문자포함 최소 4자 이상 12자 이하)
충족 못할 시 메시지 출력

02 데이터베이스내 데이터와 중복 확인. 메시지 출력

0 4 . 회원가입 정보입력

01

회원가입

약관동의	정보입력	가입완료

▪ 아이디	ksh1234 사용 가능한 아이디입니다.
▪ 비밀번호	
▪ 비밀번호 확인	
▪ 이름	
닉네임	
▪ 이메일	이메일 입력 / 직접 입력 ▪ 아이디/비밀번호 찾기에 활용되므로 정확하게 입력해주세요 ☐ 정보/이벤트 메일 수신에 동의합니다.
▪ 휴대전화	- 포함하여 입력하세요

Description List

01 아이디 유효성 및 중복확인 통과 시 메시지 출력

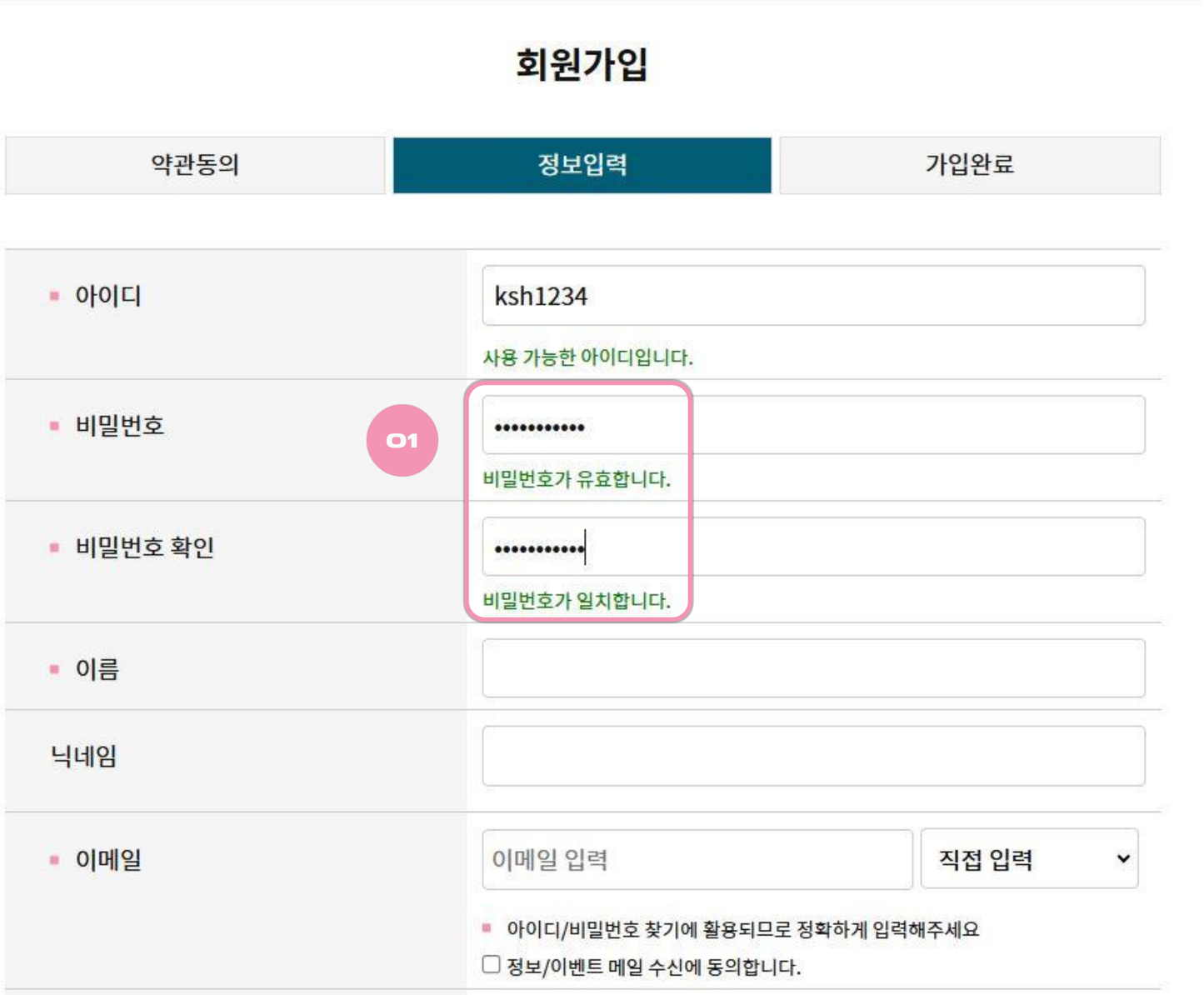
0 4 . 회원가입 정보입력

Description List

01 비밀번호 유효성 검사
(영문, 숫자, 특수문자 중 2가지 이상 조합 + 10자 이상)
충족 못할 시 메시지 출력

02 유효성 충족과 비밀번호 일치 둘 중 하나라도 충족 못할 시
통과 불가 및 alert 메시지 출력

0 4 . 회원가입 정보입력



Description List

01 유효성 및 비밀번호 일치 시 확인메시지 출력

0 4 . 회원가입 정보입력

01

localhost:5500 내용:

이름이 유효하지 않습니다.

확인

브랜드

고객센터

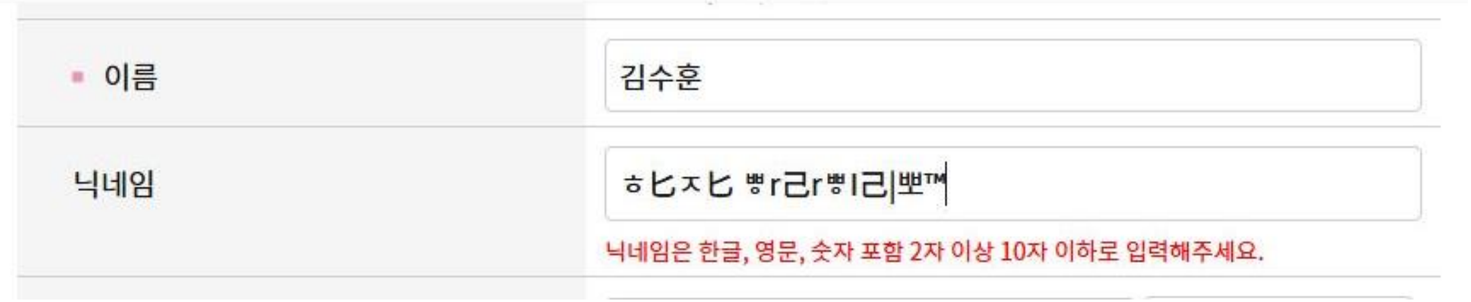
비밀번호	•••••••••••• 비밀번호가 유효합니다.
비밀번호 확인	•••••••••••• 비밀번호가 일치합니다.
이름	@@@

Description List

01 이름 유효성 검사
(한글만 최소 2자 이상 6자 이하. 공백 불가)
충족 못할 시 alert 메시지 출력

0 4 . 회원가입 정보입력

01



02

03

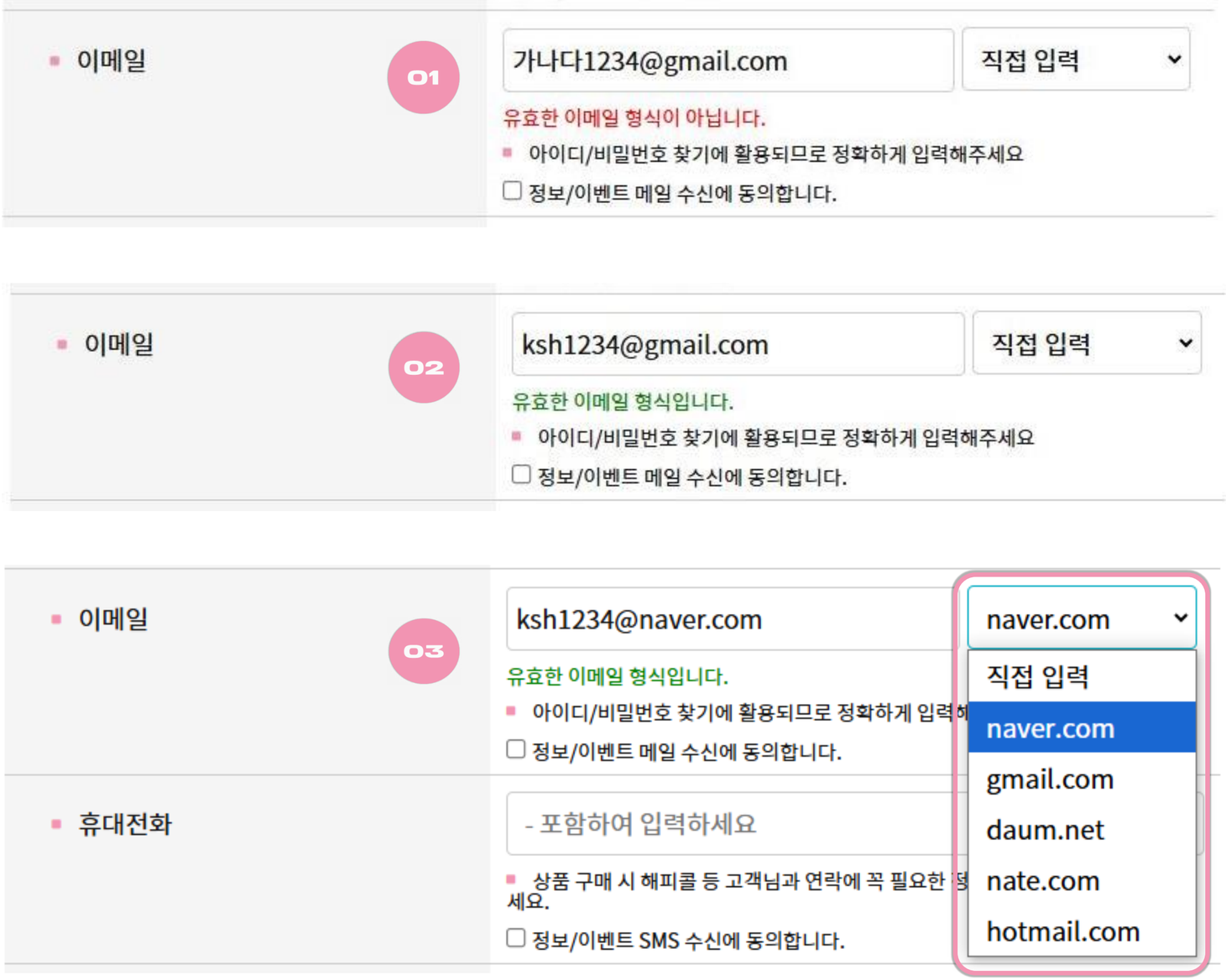
Description List

01 닉네임 유효성 검사
(한글, 영문, 숫자 포함 2글자 이상 10자 이하. 특수문자 불
충족 못할 시 메시지 출력

02 데이터베이스내 데이터와 중복 확인. 메시지 출력

03 유효성 충족 및 중복 없을 시 사용 가능 메시지 출력

0 4 . 회원가입 정보입력



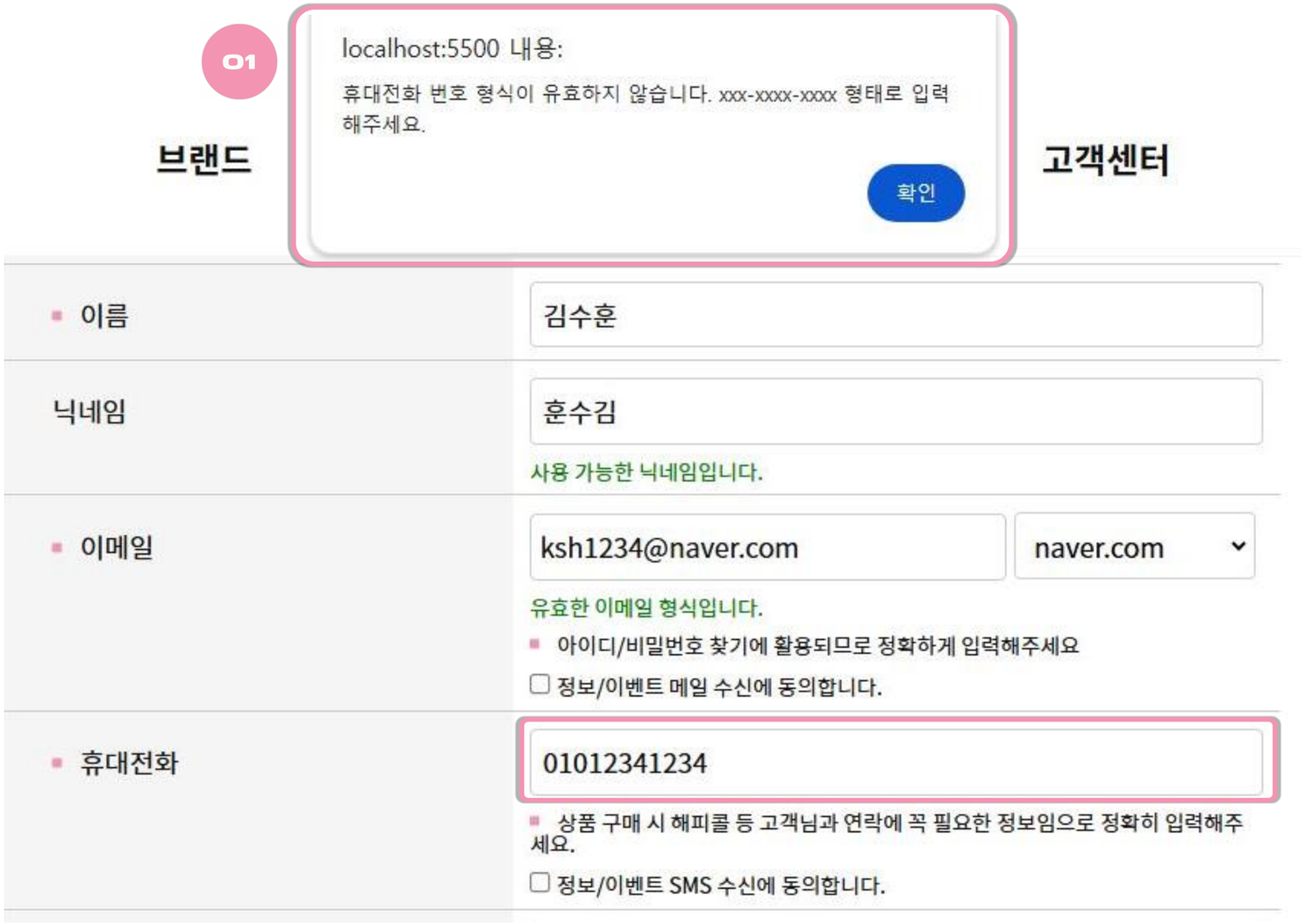
Description List

01 이메일 유효성 검사
로컬파트(영문자, 숫자, ._%+- 만 가능)
@ 도메인(영문자와 숫자, .- 만 가능) .
최상위도메인(영문자만 허용, 최소 2글자 이상)
충족 못할 시 메시지 출력

02 유효성 충족 시 메시지 출력

03 Select의 도메인 선택 시 자동으로 이메일 아이디 뒤에 붙

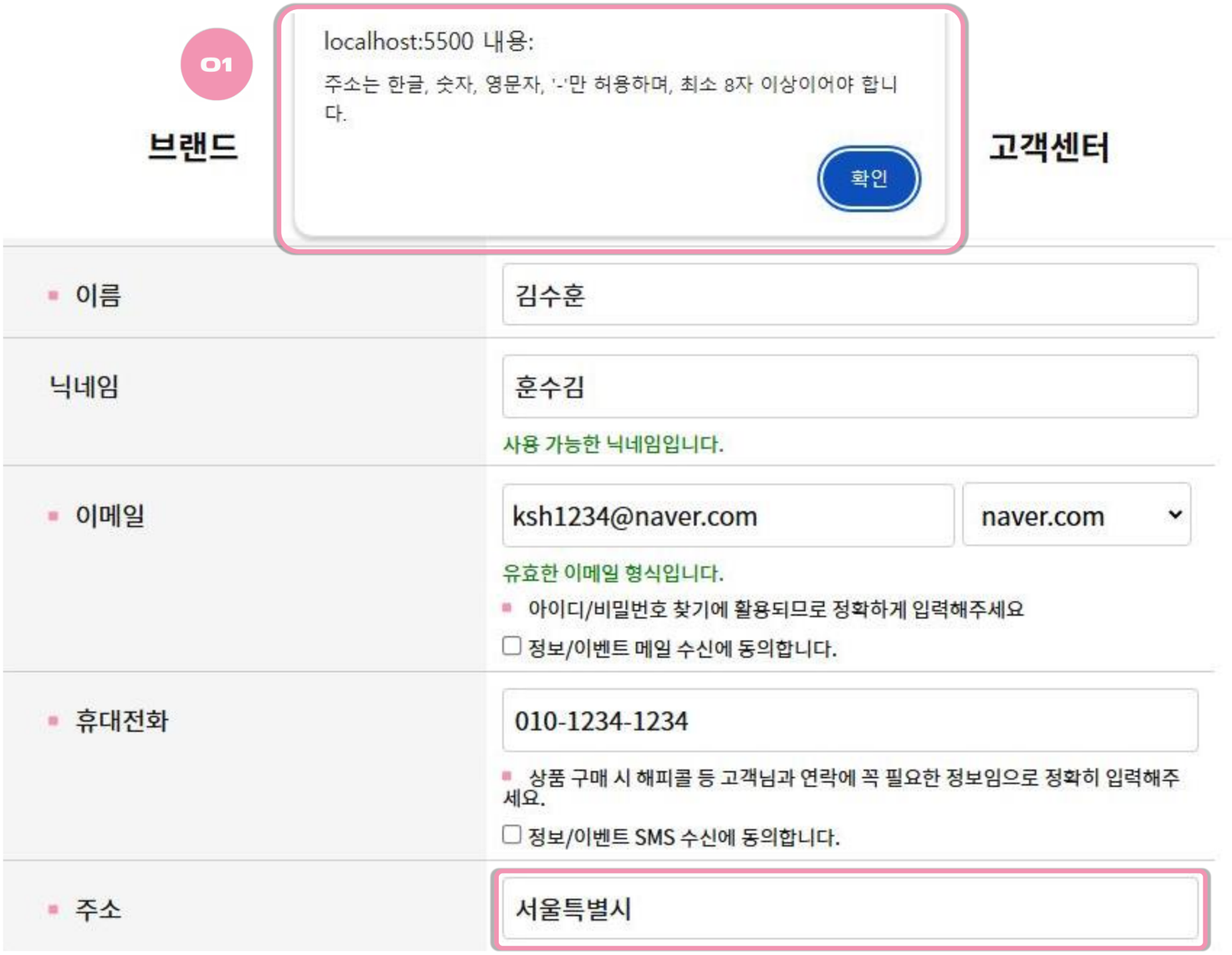
0 4 . 회원가입 정보입력



Description List

01 휴대전화 유효성 검사
(숫자만 허용. xxx-xxxx-xxxx의 형태만 허용)
충족 못할 시 alert 메시지 출력

0 4 . 회원가입 정보입력



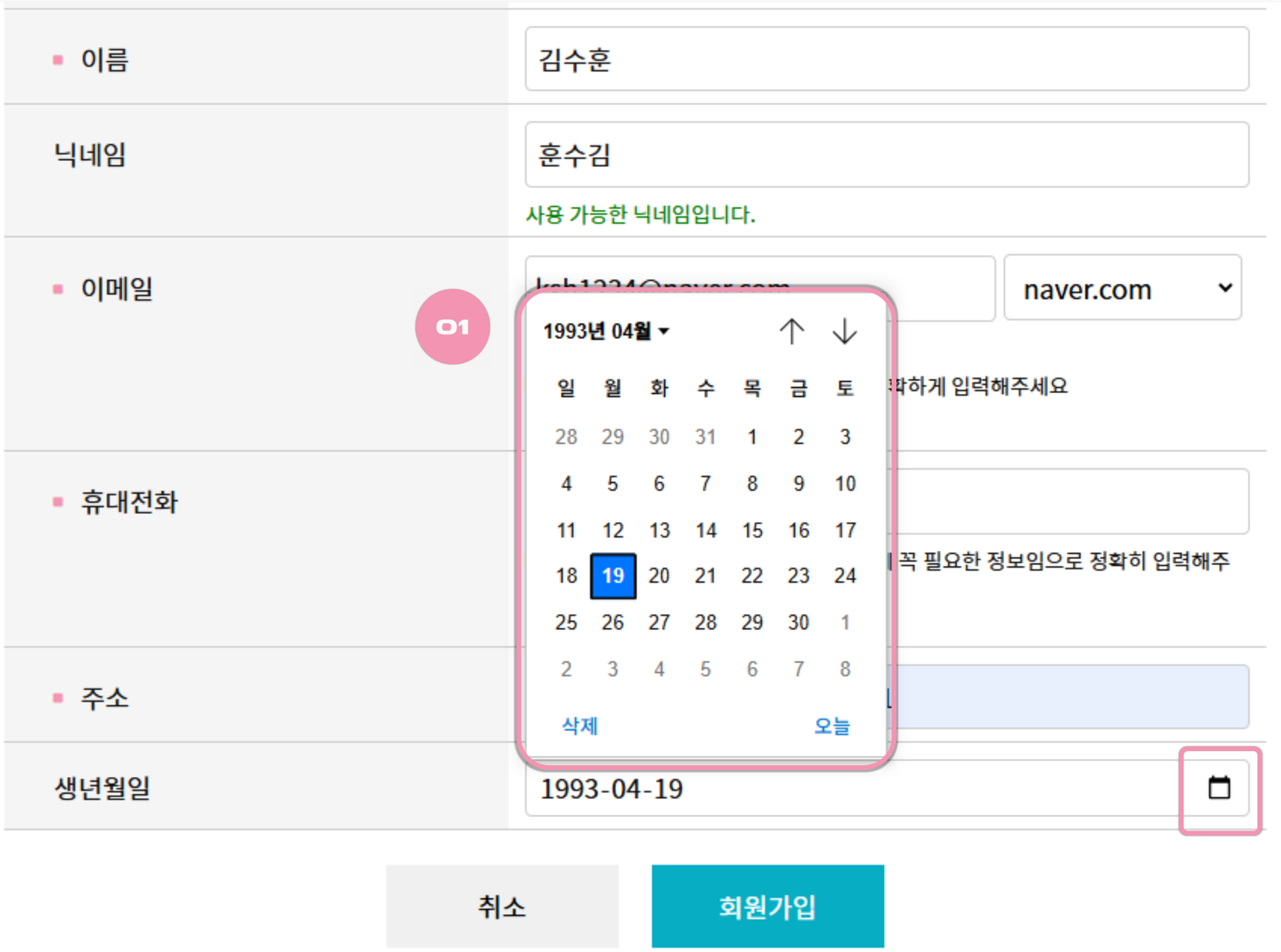
Description List

01 주소 유효성 검사

(한글, 숫자, 영문자, 공백, 특수문자 '-'만 허용, 최소 8자 이

충족 못할 시 alert 메시지 출력

0 4 . 회원가입 정보입력

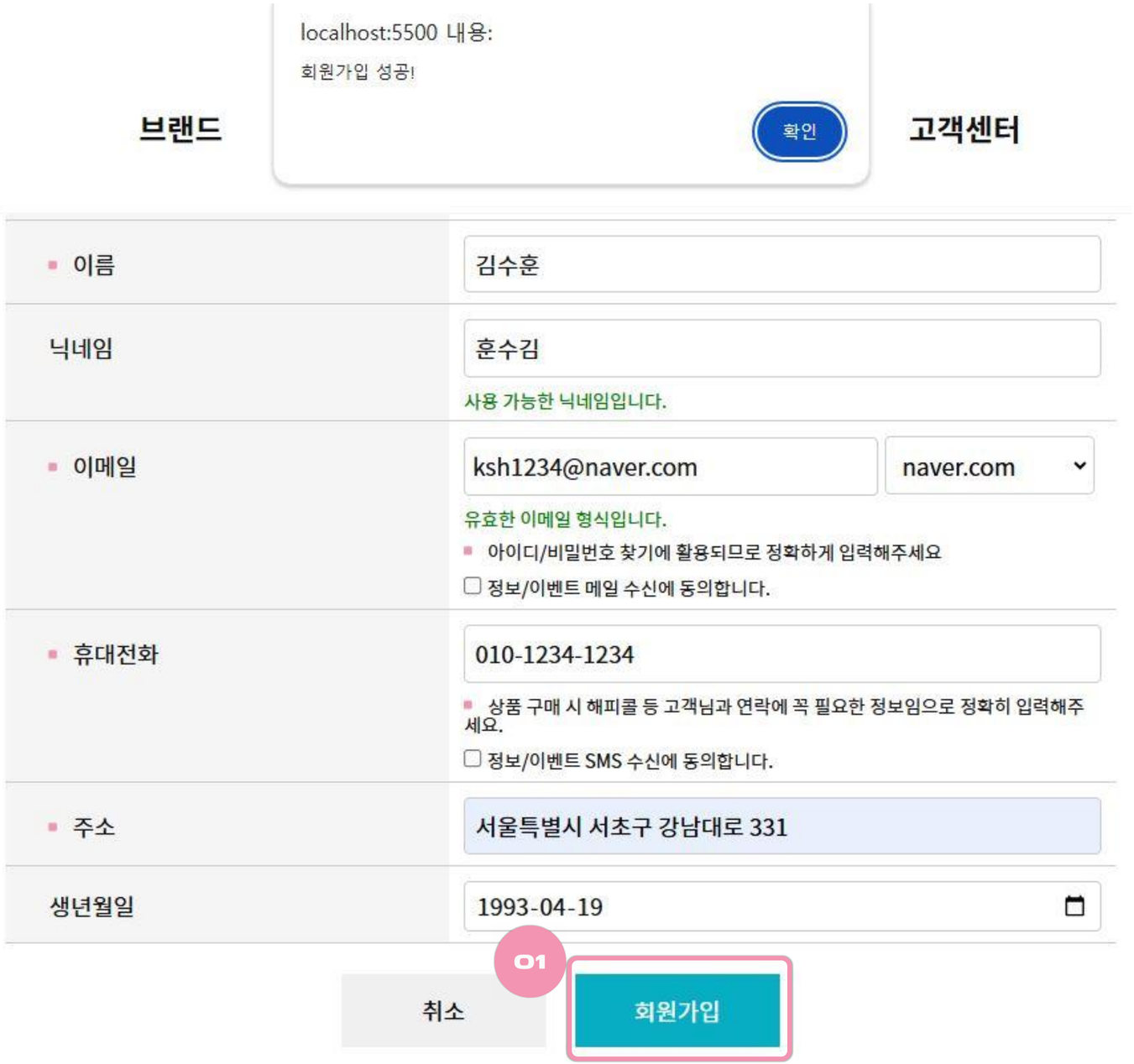


Description List

01 생년월일은 선택사항

달력모양 클릭 시 달력 출력 및 생년월일 선택

0 4 . 회원가입 정보입력



Description List

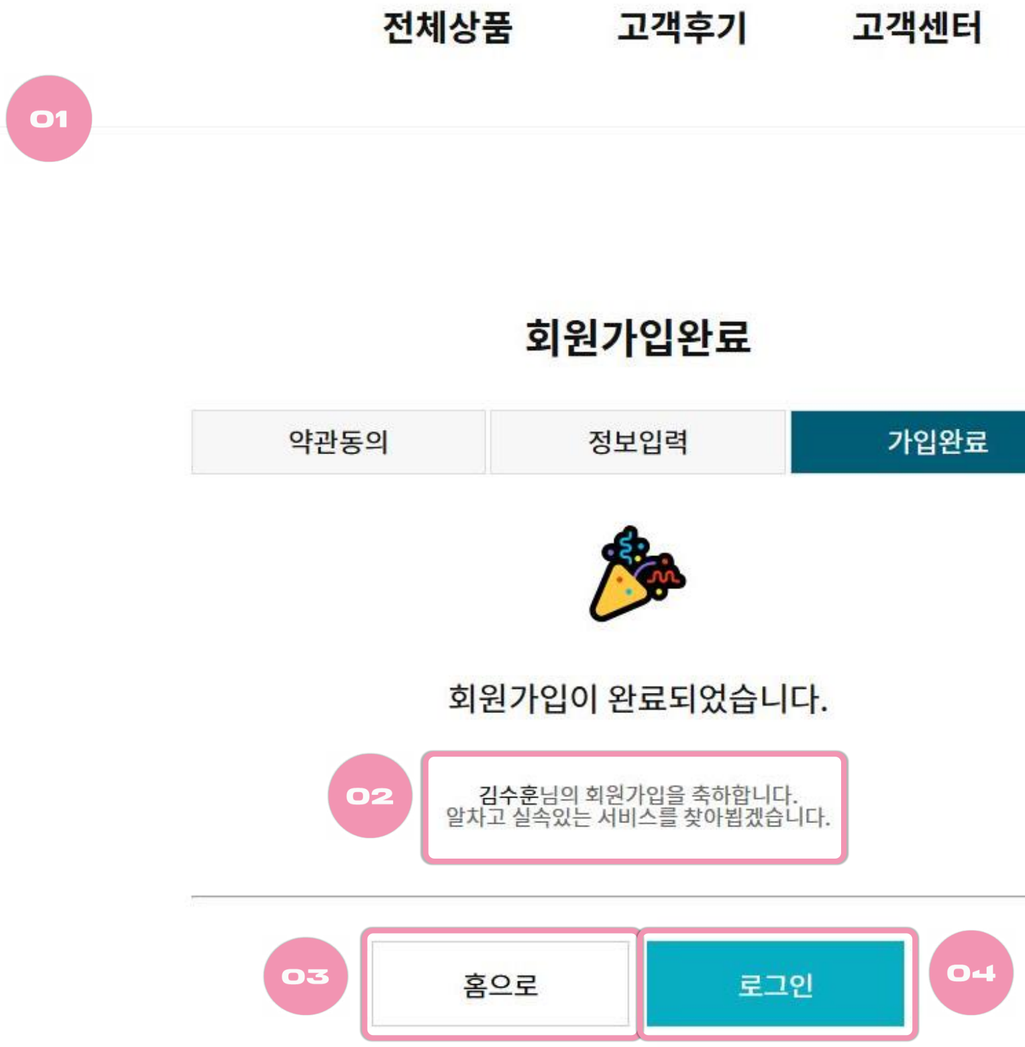
01 모든 정보를 조건에 맞게 입력 후 회원가입 버튼 클릭 시 회원가입 성공이라는 alert 메시지 출력

02 데이터베이스에 입력한 회원정보 저장

* id varchar(50)	* pw varchar(255)	* name varchar(100)	nick varchar(50)
ksh1234	tngns1234@@	김수훈	훈수김

birth_date date	phone_num varchar(20)	email varchar(100)	addr varchar(255)
1993-04-19	010-1234-1234	ksh1234@naver.com	서울특별시 서초구 강남

0 5. 회원가입 완료



Description List

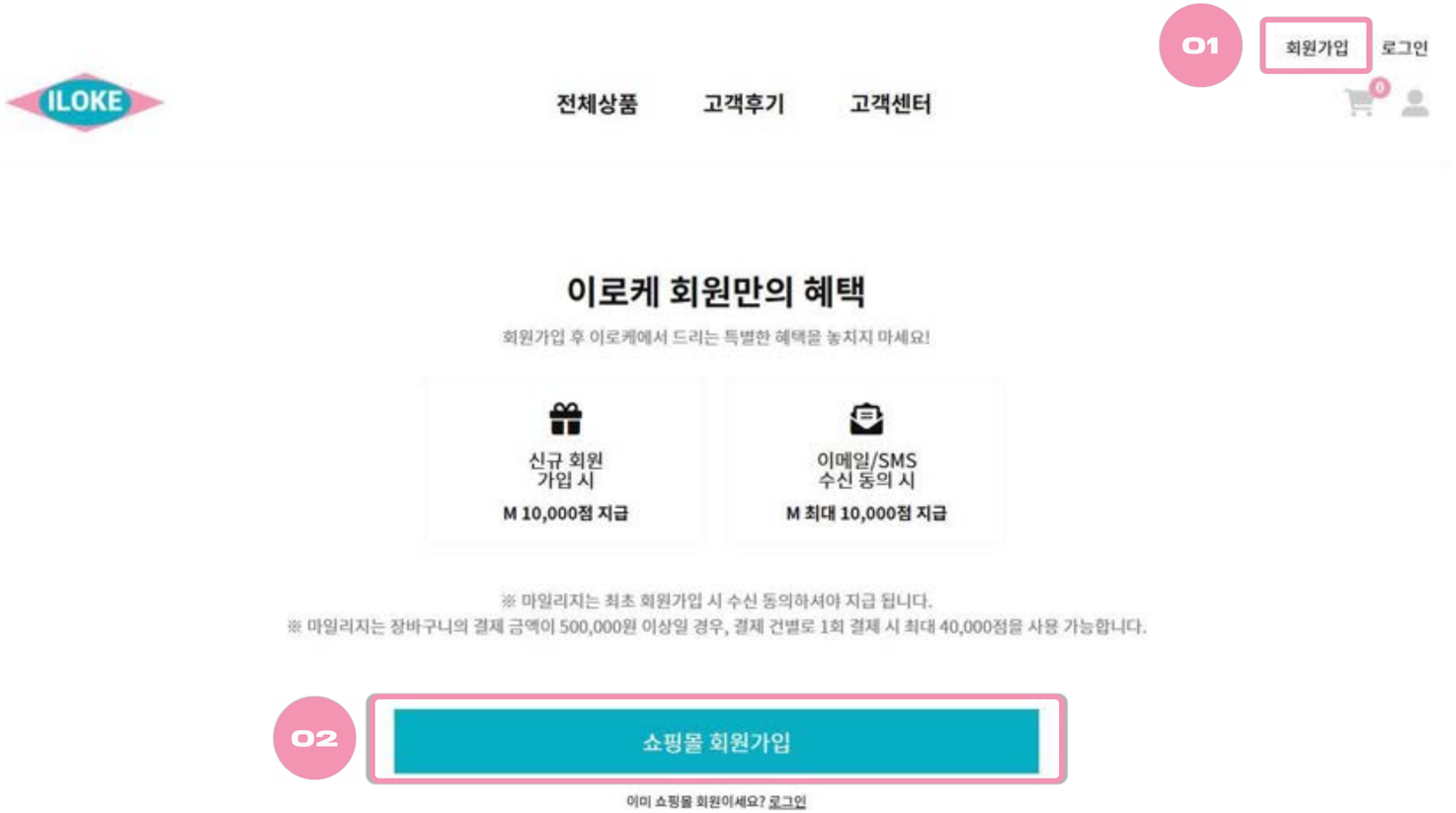
01 회원가입 완료 페이지

02 가입한 회원 이름과 함께 축하 메시지 출력

03 메인 페이지로 이동

04 로그인 페이지로 이동

06. 회원가입 안내 및 혜택



Description

01 회원가입 클릭 시 회원가입 안내 페이지로 이동

02 쇼핑몰 회원가입 버튼(클릭 시 약관동의 페이지로 이동)

07. 회원가입 이용약관

회원가입

01

약관동의	정보입력	가입완료

☐ 이로케의 모든 약관을 확인하고 전체 동의함 (전체동의는 선택항목도 포함됩니다)

☐ (필수) 이용약관 동의

제1조(목적)

이 약관은 (주)자코모 회사(전자거래 사업자)가 운영하는 자코모 사이버 몰(이하 "몰"이라 한다)에서 제공하는 인터넷 관련 서비스(이하 "서비스"라 한다)를 이용함에 있어 사이버몰과 이용자의 권리·의무 및 책임사항을 규정함을 목적으로 합니다.
※ 「PC통신등을 이용하는 전자거래에 대해서도 그 성질에 반하지 않는한 이 약관을 준용합니다」

제2조(정의)

① "몰"이란 (주)자코모 회사가 재화 또는 용역을 이용자에게 제공하기 위하여 컴퓨터등 정보통신설비를 이용하여 재화 또는 용역을 거래할 수 있도록 설정한 가상의 영업장을 말하며, 아울러 사이버몰을 운영하는 사업자의 의미로도 사용합니다.
② "이용자"란 "몰"에 접속하여 이 약관에 따라 "몰"이 제공하는 서비스를 받는 회원 및 비회원을 말합니다.
③ '회원'이라 함은 "몰"에 개인정보를 제공하여 회원등록을 한 자로서, "몰"의 정보를 지속적으로 제공받으며, "몰"이 제공하는 서비스를 계속적으로 이용할 수 있는 자를 말합니다.
④ '비회원'이라 함은 회원에 가입하지 않고 "몰"이 제공하는 서비스를 이용하는 자를 말합니다.

02

☐ (필수) 개인정보 수집 및 이용 동의

개인정보 수집 및 이용은 회원가입, 고객상담, 본인여부 확인, 주문/결제 및 배송 서비스 이용에 있다.
각 이용 목적과 그 항목, 기간은 아래로 정의된다.

- 수집 및 이용 목적 : 이용자 식별 및 본인여부 확인
- 수집 항목 : 이름, 아이디, 비밀번호
- 보유 및 이용기간 : 회원탈퇴 후 5일까지

- 수집 및 이용 목적 : 민원 등 고객 고충처리
- 수집 항목 : 이메일, 휴대폰번호
- 보유 및 이용기간 : 회원 탈퇴 후 5일까지

- 수집 및 이용 목적: 주문/결제 시 상품 배송
- 수집 항목: 구매자 정보, 상품 구매/취소/반품/교환/환불 정보, 수령인 정보
- 보유 및 이용기간: 회원 탈퇴 후 5일까지

- 수집 및 이용 목적 : 만 14세 미만 아동 확인

☐ (선택) 마케팅 정보 수신 및 활용 동의

(주)자코모에서 제공하는 이벤트 및 혜택 등 다양한 정보를 문자, 이메일 등으로 받아보실 수 있습니다.
마케팅 정보 수신 및 활용 동의 여부와 관계없이 회원가입 및 서비스를 이용하실 수 있습니다.
또한 서비스의 중요 안내 사항 및 주문/배송에 대한 정보는 마케팅 정보 수신 동의 여부와 관계없이 발송됩니다.

1. 수집 및 이용 목적
- 인구 통계학적 특성에 따른 서비스 제공 및 광고 게재
- 접속 이용 및 방문 빈도 파악, 서비스 이용에 대한 통계 등 업무와 관련된 각종 통계 자료의 작성 및 서비스 개발

Description List

01 전체 동의 체크박스 클릭 시 전체 약관 동의 체크

02 개별 체크도 가능

0 7. 회원가입 이용약관

☐ (필수) 개인정보 수집 및 이용 동의

개인정보 수집 및 이용은 회원가입, 고객상담, 본인여부 확인, 주문/결제 및 배송 서비스 이용에 있다.
각 이용 목적과 그 항목, 기간은 아래로 정의된다.

- 수집 및 이용 목적 : 이용자 식별 및 본인여부 확인
- 수집 항목 : 이름, 아이디, 비밀번호
- 보유 및 이용기간 : 회원탈퇴 후 5일까지

- 수집 및 이용 목적 : 민원 등 고객 고충처리
- 수집 항목 : 이메일, 휴대폰번호
- 보유 및 이용기간 : 회원 탈퇴 후 5일까지

- 수집 및 이용 목적: 주문/결제 시 상품 배송
- 수집 항목: 구매자 정보, 상품 구매/취소/반품/교환/환불 정보, 수령인 정보
- 보유 및 이용기간: 회원 탈퇴 후 5일까지

- 수집 및 이용 목적 : 만 14세 미만 아동 확인

☑ (선택) 마케팅 정보 수신 및 활용 동의

(주)자코모에서 제공하는 이벤트 및 혜택 등 다양한 정보를 문자, 이메일 등으로 받아보실 수 있습니다.
마케팅 정보 수신 및 활용 동의 여부와 관계없이 회원가입 및 서비스를 이용하실 수 있습니다.
또한 서비스의 중요 안내 사항 및 주문/배송에 대한 정보는 마케팅 정보 수신 동의 여부와 관계없이 발송됩니다.

1. 수집 및 이용 목적
- 인구 통계학적 특성에 따른 서비스 제공 및 광고 게재
- 접속, 이용 및 방문 빈도 파악, 서비스 이용에 대한 통계 등 업무와 관련된 각종 통계 자료의 작성 및 서비스 개발
- 신규 서비스 개발과 이벤트 행사에 따른 정보 전달 및 맞춤 서비스 제공
- 제품 안내, 홍보 및 이벤트 등의 마케팅, 광고성 정보 전달

2. 수집 및 이용 항목
- 회원 정보(이름, 휴대폰번호, 이메일, 성별, 생년월일, 가입일시)
- 구매 및 배송 서비스 이용 정보(장바구니 상품, 구매일시, 주소, 받으시는 곳)
- 수집 및 이용 항목은 마케팅 및 분석, 프로모션의 목적에 따라 달라질 수 있으며 수집 시점에 동의 여부를 별도 확인함

3. 보유 및 이용기간

01

❗ (필수) 약관에 모두 동의해주세요.

02

03

이전단계 | 다음단계

Description List

01 필수 사항 미 체크 시 메시지 출력

02 회원가입 안내 페이지로 이동

03 회원가입 정보입력으로 이동

08. 회원가입 핵심 코드

01

```
// 회원가입 처리 라우터
router.post('/join', async (req, res) => {
    const { id, pw, name, nick, birth_date, phone_num, email, addr } = req.body;

    // 필수 항목이 누락된 경우 클라이언트에 에러 응답 반환
    if (!id || !pw || !name || !email || !phone_num || !addr) {
        return res.status(400).json({ success: false, message: '필수 항목이 누락되었습니다.' });
    }

    try {
        // 회원 정보를 mem_info 테이블에 삽입하기 위한 SQL 쿼리
        const sql = `
            INSERT INTO mem_info (id, pw, name, nick, birth_date, phone_num, email, addr, reg_id)
            VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)
        `;
        // SQL 쿼리에 바인딩될 데이터 배열
        const data = [
            id,
            pw,
            name,
            nick || null,
            birth_date || null,
            phone_num,
            email,
            addr,
            id,
        ];

        const [result] = await conn.execute(sql, data);

        res.json({ success: true, message: '회원가입이 완료되었습니다.' });
    } catch (error) {
        console.error('회원가입 오류:', error);
        res.status(500).json({ success: false, message: '회원가입 중 오류가 발생했습니다.' });
    }
});

module.exports = router;
```

Description List

01 회원가입 요청 처리 및 데이터베이스에 회원 정보를 저장
회원가입 처리 라우터

08. 회원가입 핵심 코드

01

```
const handleSubmit = async (e) => {
  e.preventDefault();

// 아이디 유효성 검사 실패 시 제출 방지
if (!idValid) {
  alert("아이디가 유효하지 않습니다. 조건을 충족하는 아이디를 입력해주세요.");
  return;
}

// 아이디 중복 확인 로직
if (idDuplicate) {
  alert("이미 사용 중인 아이디입니다. 다른 아이디를 입력해주세요.");
  return;
}

// 비밀번호 확인 로직
if (formData.pw !== formData.pw2) {
  alert("비밀번호가 일치하지 않습니다.");
  return;
}

// 비밀번호 유효성 검사
if (!pwValid) {
  alert("비밀번호가 유효하지 않습니다. 조건을 충족하는 비
  return;
}

// 이름 유효성 검사
if (!nameValid) {
  alert("이름이 유효하지 않습니다.");
  return;
}
```

```
// 서버로 보낼 데이터 준비 (pw2 제외)
const dataToSend = {
id: formData.id,
pw: formData.pw,
name: formData.name,
nick: formData.nick,
email: formData.email,
phone_num: formData.phone_num,
addr: formData.addr,
birth_date: formData.birth_date,
};

try {
  const response = await memJoin(dataToSend);
  if (response.data.success) {
    alert("회원가입 성공!");
    navigate(`/joinFin?name=${encodeURIComponent(formData.name)}`);

  } else {
    alert("회원가입 실패: " + response.data.message);
  }
} catch (error) {
  alert("회원가입 중 오류가 발생했습니다.");
  }
};
```

Description List

01 각 유효성 검사 및 중복 확인을 종합하여 검증 완료 되었을
데이터를 서버로 전송하는 함수

08. 회원가입 핵심 코드

01

```
const express = require("express");
const router = express.Router();
const conn = require("../../utils/dbUtil");

// 아이디 중복 확인 라우터
router.post("/", async (req, res) => {
  const { id } = req.body;

  if (!id) {
    return res.status(400).json({ message: "아이디를 입력해주세요." });
  }

  try {
    const [results] = await conn.query
    ("SELECT COUNT(*) AS count FROM mem_info WHERE id = ?", [id]);
    const isDuplicate = results[0].count > 0;

    res.json({ isDuplicate });
  } catch (error) {
    res.status(500).json({ message: "서버 오류 발생" });
  }
});

module.exports = router;
```

02

```
app.use('/mall/checkId', checkIdController);          // 아이디 중복확인
```

03

```
export const checkIdDup = (id) => API.post('./checkId', { id });
```

04

```
// 아이디 중복 확인 함수
const checkIdDuplicate = async (id) => {
  if (!id) {
    setIdMsg("");
    setIdDuplicate(false);
    return;
  }

  // 유효성 검사 실패 시 중복 확인 생략
  if (!validId(id)) {
    return;
  }

  try {
    const response = await checkIdDup(id);
    if (response.data.isDuplicate) {
      setIdMsg("이미 사용 중인 아이디입니다.");
      setIdDuplicate(true);
    } else {
      setIdMsg("사용 가능한 아이디입니다.");
      setIdDuplicate(false);
    }
  } catch (error) {
    setIdMsg("아이디 중복 확인 중 오류가 발생했습니다.");
    setIdDuplicate(true);
  }
};
```

Description List

01 아이디 중복 확인 요청 라우터

02 아이디 중복 확인 요청을 처리하는 라우터 등록

03 아이디 중복확인을 위한 API 호출

04 아이디 중복 확인 함수

0 9 . 아이디 찾기

Description List

01 아이디 찾기 버튼
이름과 이메일 주소가 가입된 정보와 일치 시 찾기 가능

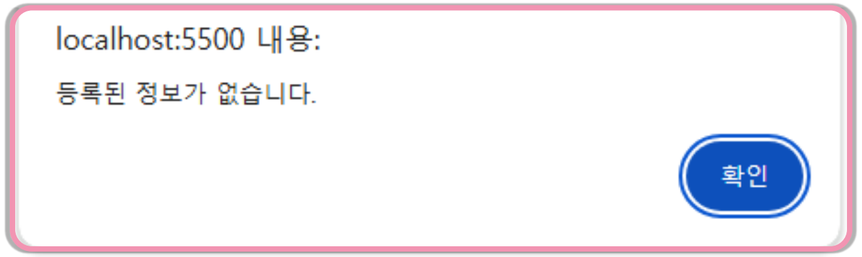
02 회원 가입 정보 입력의 이메일과 같은 형식으로
Select의 도메인 선택 시 자동으로 도메인이 붙음

03 비밀번호 찾기 페이지로 이동

04 로그인 페이지로 이동

0 9 . 아이디 찾기

01

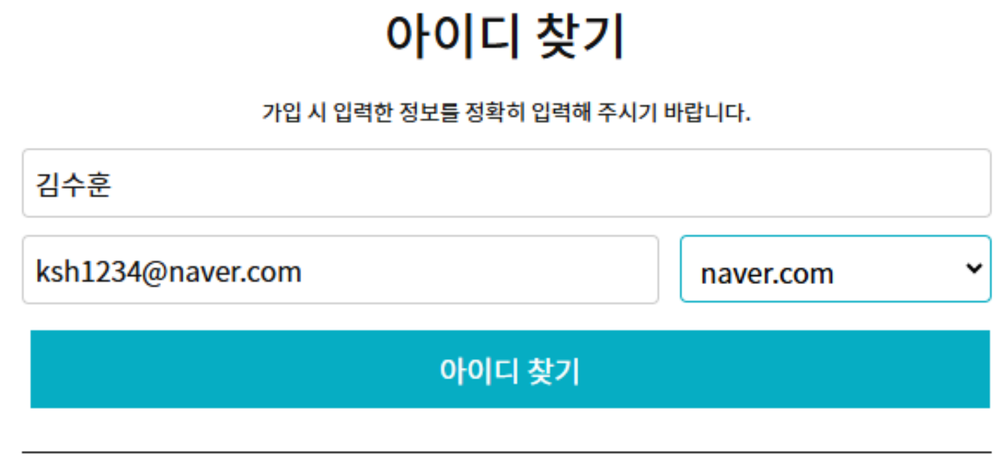


* id varchar(50)	* pw varchar(255)	* name varchar(100)	nick varchar(50)	birth_date date	phone_num varchar(20)	email varchar(100)
ksh1234	tngns1234@@	김수훈	훈수김	1993-04-19	010-1234-1234	ksh1234@naver.com

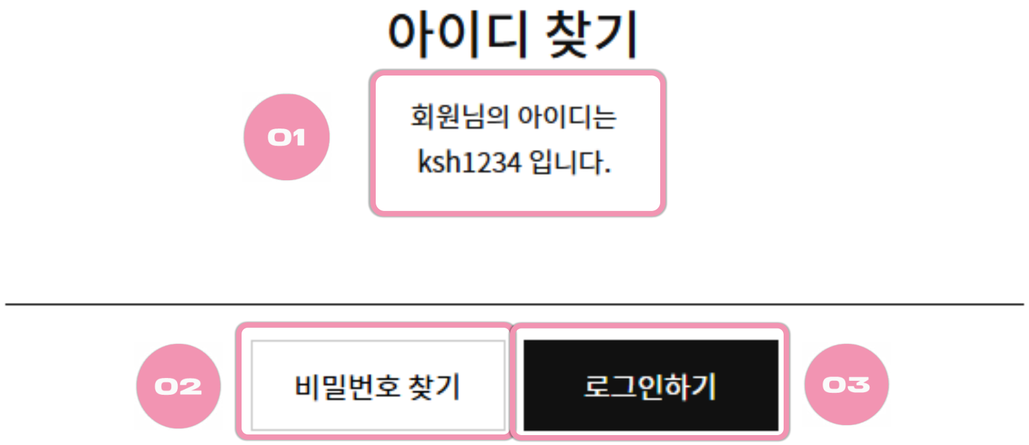
Description List

01 이름과 이메일 주소가 가입된 정보와 일치 하지 않을 시 아이디 찾기를 눌렀을 때 alert 메시지 출력

0 9 . 아이디 찾기



* id varchar(50)	* pw varchar(255)	* name varchar(100)	nick varchar(50)	birth_date date	phone_num varchar(20)	email varchar(100)
ksh1234	tngns1234@@	김수훈	훈수김	1993-04-19	010-1234-1234	ksh1234@naver.com



Description List

01 이름과 이메일이 가입 정보와 일치 시 해당하는 아이디가 출력됨

02 비밀번호 찾기 페이지로 이동

03 로그인 페이지로 이동

10. 아이디 찾기 핵심코드

01

```
router.post('/', async (req, res) => {
  const { name, email } = req.body; // 회원으로부터 이름과 이메일을 추출

  try {
    // 입력받은 이름과 이메일을 기반으로 데이터베이스에서 회원 정보를 조회
    const sql = 'SELECT id FROM mem_info WHERE name = ? AND email = ?';
    const [rows] = await conn.execute(sql, [name, email]);

    // 조회된 결과가 없는 경우
    if (rows.length === 0) {
      return res.status(404).json({ success: false, message: '등록된 정보가 없습니다.' });
    }
    // 조회 성공 시 아이디와 이름을 클라이언트에 응답
    res.json({ success: true, id: rows[0].id, name: rows[0].name });
  } catch (error) {
    console.error('아이디 찾기 오류:', error);
    res.status(500).json({ success: false, message: '서버 오류가 발생했습니다.' });
  }
});

module.exports = router;
```

02

```
const FindId = () => {
  const [name, setName] = useState(''); // 사용자 이름 상태
  const [email, setEmail] = useState(''); // 사용자 이메일 상태
  const navigate = useNavigate();

  // 아이디 찾기 처리 함수
  const handleFindId = async () => {
    try {
      const response = await findId(name, email); // 서버로 아이디 찾기 요청 전송
      navigate(`/findId2/${response.data.id}`); // find_id2.js로 이동하며 찾은 아이디 전달
    } catch (error) {
      alert(error.response?.data?.message || '서버 오류');
    }
  };
```

Description List

01 사용자 이름과 이메일을 기반으로 등록된 아이디를 반환하
아이디 찾기 라우터

02 아이디 찾기 요청 처리 함수

10. 아이디 찾기 핵심코드

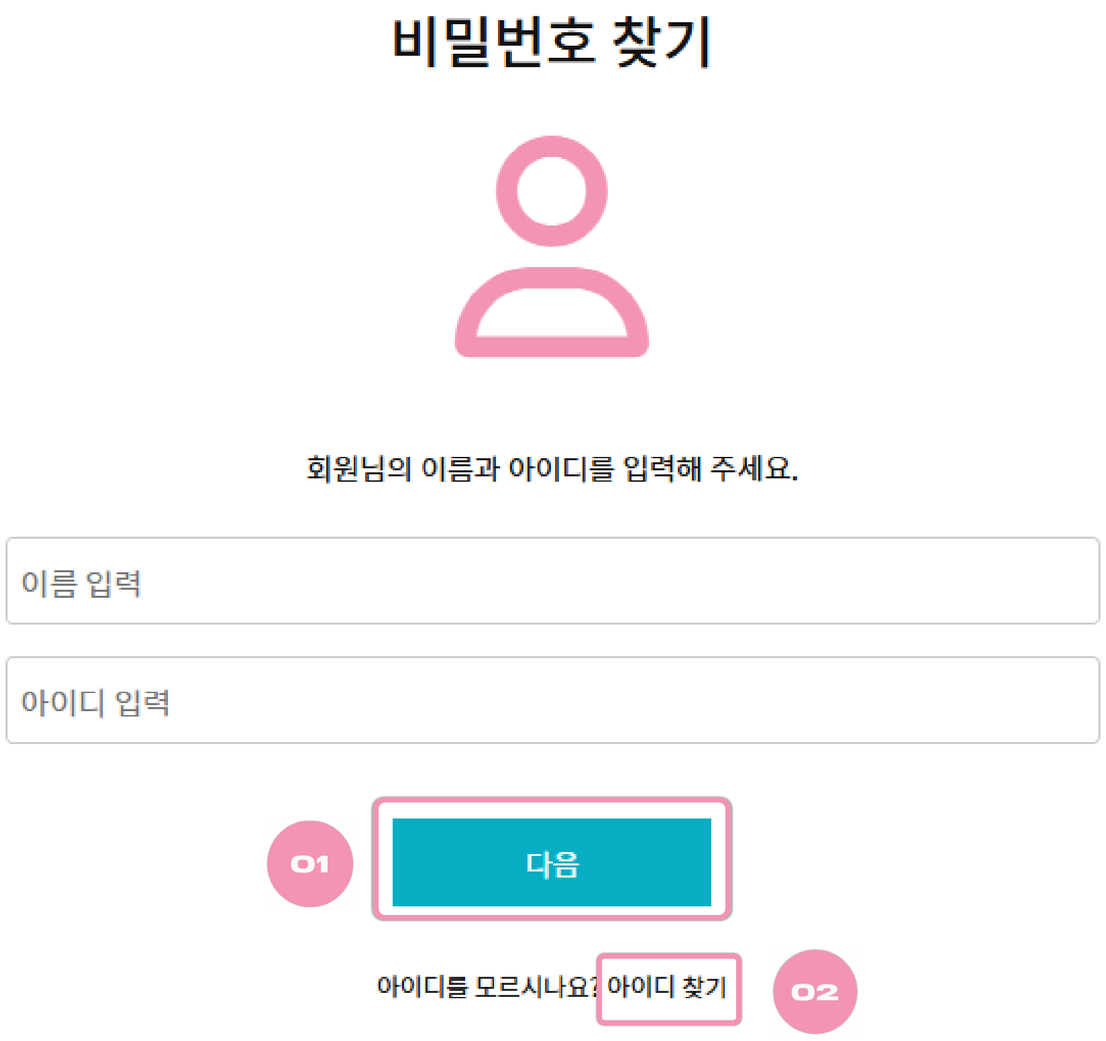
01

```
<select
  className={styles.emailDomain}
  onChange={(e) => setEmail((prevEmail) => `${prevEmail.split('@')[0]}@${e.target.value}`)}
>
  <option value="">직접입력</option>
  <option value="naver.com">naver.com</option>
  <option value="daum.net">daum.net</option>
  <option value="nate.com">nate.com</option>
  <option value="hotmail.com">hotmail.com</option>
  <option value="gmail.com">gmail.com</option>
</select>
```

Description List

01 이메일 도메인 선택 시 기존 입력된 로컬 파트와 결합하여
자동으로 이메일 주소를 완성

11. 비밀번호 찾기



Description List

01 이름과 아이디가 가입 정보와 일치 시 비밀번호 재설정 페이지로 이동

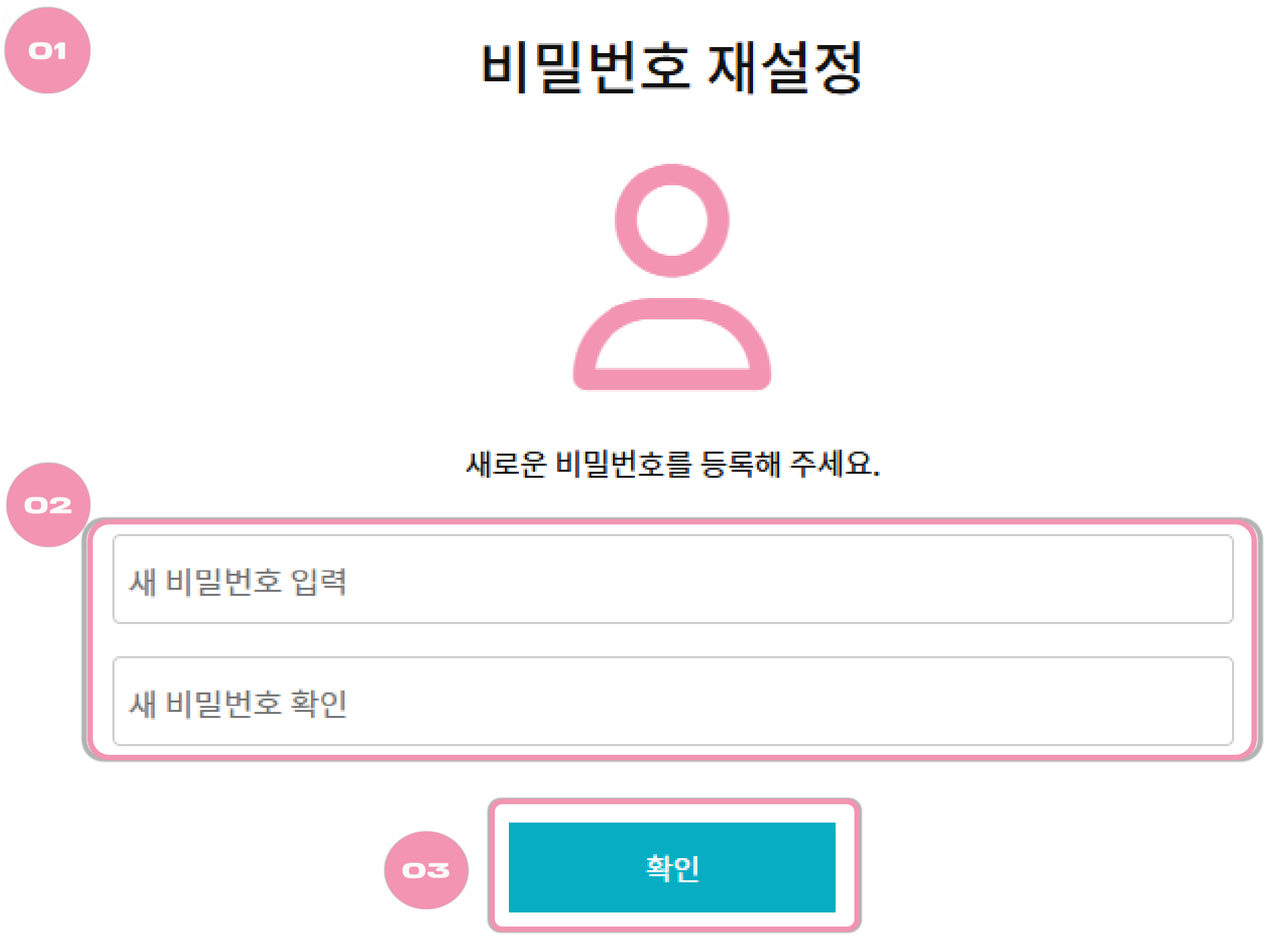
02 아이디 찾기 페이지로 이동

11. 비밀번호 찾기

Description List

01 이름과 아이디가 가입 정보와 일치 하지 않을 시 메시지 출력 및 비밀번호 재설정 페이지로 이동 불가

11. 비밀번호 찾기



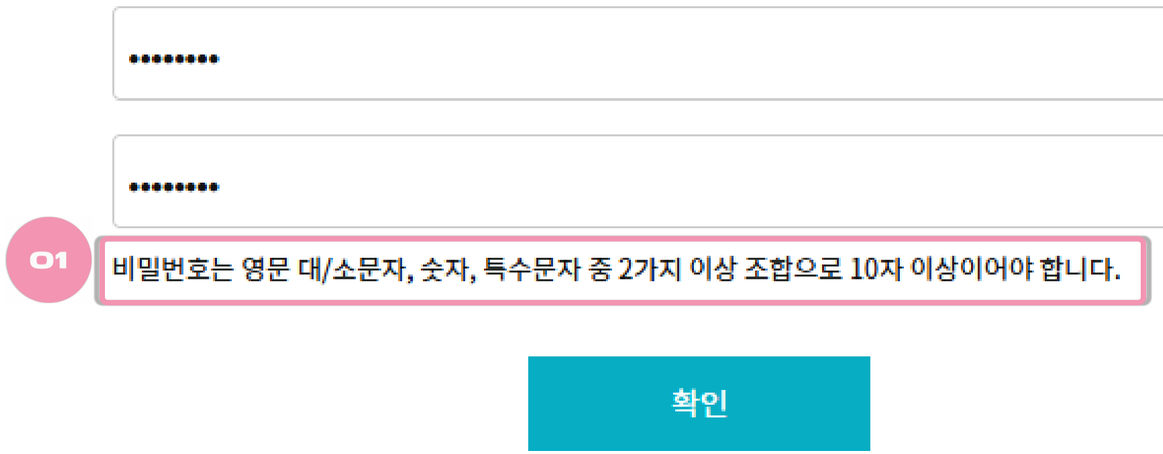
Description List

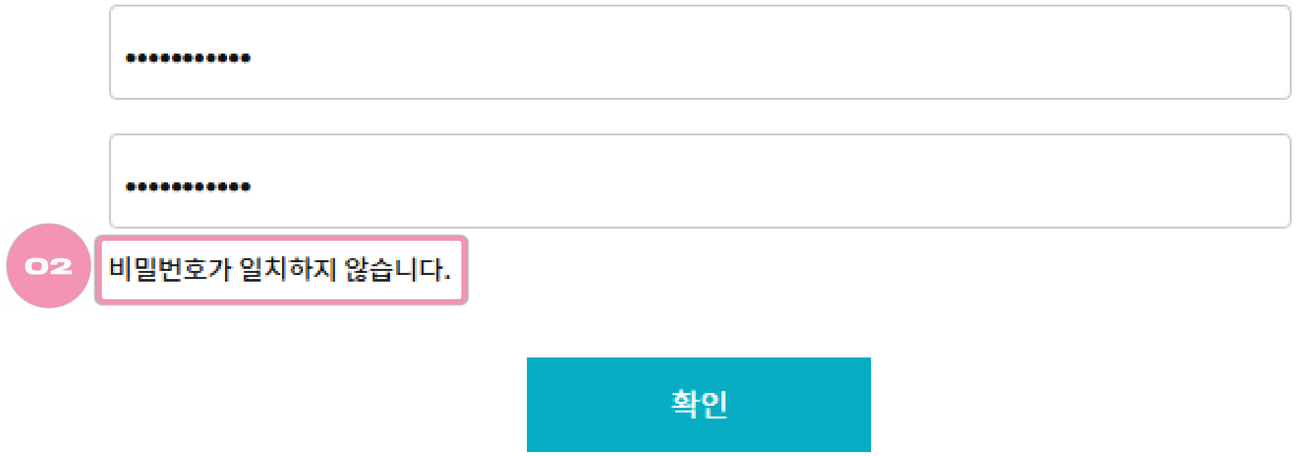
01 이름과 아이디가 가입 정보와 일치 시 비밀번호 재설정 페이지로 이동

02 새 비밀번호 입력 및 확인
두 비밀번호가 일치해야 하며 유효성은 기존 회원가입과
(영문, 숫자, 특수문자 중 2가지 이상 조합 + 10자 이상)

03 확인 버튼
유효성 통과 및 비밀번호 일치 해야함

11. 비밀번호 찾기





Description List

01 새 비밀번호가 유효성에 충족하지 못할 시 메시지 출력 (영문, 숫자, 특수문자 중 2가지 이상 조합 + 10자 이상)

02 새 비밀번호가 유효성은 충족해도 일치하지 않을 시 메시지 출력

11. 비밀번호 찾기

01

비밀번호 변경 완료

비밀번호가 성공적으로 변경되었습니다.

02

로그인

* id varchar(50)	* pw varchar(255)	* name varchar(100)
ksh1234	tngns1234@@	김수훈

* id varchar(50)	* pw varchar(255)	* name varchar(100)
ksh1234	tngns4321!!	김수훈

Description List

01 유효성 통과 및 새 비밀번호 일치 시 비밀번호 변경 완료

02 로그인 페이지로 이동

12. 비밀번호 찾기 핵심코드

01

```
// 비밀번호 재설정 라우트
router.post('/', async (req, res) => {
  const { userId, userPw } = req.body;

  // 비밀번호 유효성 검사 (숫자와 영문자를 모두 포함한 10글자 이상)
  const pwRegex = /^(?=.*[a-zA-Z])(?=.*[0-9]).{10,}$/;

  if (!pwRegex.test(userPw)) {
    return res.status(400).json({ success: false, message: '비밀번호는 숫자와 영문자를 모두 포함
  }

  try {
    // 비밀번호 업데이트 쿼리
    const sql = 'UPDATE mem_info SET pw = ? WHERE id = ?';
    const [result] = await conn.execute(sql, [userPw, userId]);

    // 업데이트된 행이 없으면 존재하지 않는 사용자 ID로 판단
    if (result.affectedRows === 0) {
      return res.status(404).json({ success: false, message: '존재하지 않는 아이디입니다.' });
    }

    res.json({ success: true, message: '비밀번호가 성공적으로 변경되었습니다.' });
  } catch (error) {
    console.error('비밀번호 변경 오류:', error);
    res.status(500).json({ success: false, message: '서버 오류가 발생했습니다.' });
  }
});

module.exports = router;
```

Description List

01 사용자가 입력한 새로운 비밀번호를 확인 후 해당 아이디
비밀번호를 재설정하는 API 라우터

12. 비밀번호 찾기 핵심코드

01

```
const FindPwReset = ({ userId }) => {
  //const { userId } = useParams(); // URL 파라미터에서 userId 가져오기
  const [userPw, setUserPw] = useState(''); // 새 비밀번호 입력 상태
  const [confirmPw, setConfirmPw] = useState(''); // 새 비밀번호 확인 상태
  const [errorMsg, setErrorMsg] = useState('');
  const navigate = useNavigate();

  const handleSubmit = async () => {
    // 비밀번호 유효성 검사 (영문 대/소문자, 숫자, 특수문자 중 2가지 이상 조합 10자 이상)
    const pwRegex = /^(?!.*\s)(?=.*[a-zA-Z])(?=.*\d|.*[!@#$%^&*])[a-zA-Z\d!@#$%^&*]{10,}$/; // 영문

    if (!pwRegex.test(userPw)) {
      setErrorMsg('비밀번호는 영문 대/소문자, 숫자, 특수문자 중 2가지 이상 조합으로 10자 이상이어야 합니다.');
      return;
    }

    if (userPw !== confirmPw) {
      setErrorMsg('비밀번호가 일치하지 않습니다.');
      return;
    }

    try {
      // 백엔드에 비밀번호 변경 요청
      const response = await findPwReset(userId, userPw); // `findPwReset`은 API 호출 함수
      if (response.data.success) {
        alert('비밀번호가 성공적으로 변경되었습니다.');
        navigate('/findPwComplete'); // 성공 시 완료 페이지로 이동
      } else {
        setErrorMsg(response.data.message || '비밀번호 변경에 실패했습니다.');
      }
    } catch (error) {
      console.error('비밀번호 변경 실패:', error.response?.data?.message || '서버 오류');
      setErrorMsg(error.response?.data?.message || '서버 오류');
    }
  };

  const fields = [
    { type: 'password', id: 'pw_input', placeholder: '새 비밀번호 입력', value: userPw,
      onChange: (e) => setUserPw(e.target.value), required: true },
    { type: 'password', id: 'pw_input_chk', placeholder: '새 비밀번호 확인', value: confirmPw,
      onChange: (e) => setConfirmPw(e.target.value), required: true },
  ];
```

Description List

01 새 비밀번호를 입력, 비밀번호 변경 요청을 보내는 비밀번호 재설정 화면 컴포넌트

13. 비밀번호 찾기 공통 컴포넌트화

01

```
import styles from './inputForm.module.css'

const InputForm = ({ fields, errorMessage, buttonText, onSubmit, className = '' }) => {
  return (
    <div className={`${styles.inputForm} ${className}`}>
      {fields.map((field, index) => (
        <div key={index} className={styles.inputField}>
          <input
            type={field.type}
            id={field.id}
            placeholder={field.placeholder}
            value={field.value}
            onChange={field.onChange}
            required={field.required || false}
          />
          {field.error && <p className={styles.fieldError}>{field.error}</p>}
        </div>
      ))}
      {errorMessage && <p className={styles.formError}>{errorMessage}</p>}
      <div className={styles.inputBtnDiv}>
        <button type="button" className={styles.submitButton} onClick={onSubmit}>
          {buttonText}
        </button>
      </div>
    </div>
  );
};

export default InputForm;
```

```
<section>
  <div className={styles.findPwContainer}>
    <div className={styles.findPwTitle}>비밀번호 찾기</div>

    <div className={styles.findMsgDiv}>
      <i id="mypage_i" className={`fa-regular fa-user ${st
      <p>회원님의 이름과 아이디를 입력해 주세요.</p>
    </div>

    <InputForm
      fields={fields}
      errorMessage={errorMessage}
      buttonText="다음"
      onSubmit={handleNextClick}
    />
```

```
<section>
  <div className={styles.resetContainer}>
    <div className={styles.resetTitle}>비밀번호 재설정</div>

    <div className={styles.resetPwMsg}>
      <i id="mypage_i" className={`fa-regular fa-user ${styl
      <p>새로운 비밀번호를 등록해 주세요.</p>
    </div>

    <InputForm
      fields={fields}
      errorMessage={errorMsg}
      buttonText="확인"
      onSubmit={handleSubmit}
    />
  </div>
```

Description List

01 비밀번호 찾기에서 공통으로 쓰이는 폼을 컴포넌트화

14. 로그인

Description List

01 로그인 페이지

14. 로그인

Description List

01 로그인 버튼

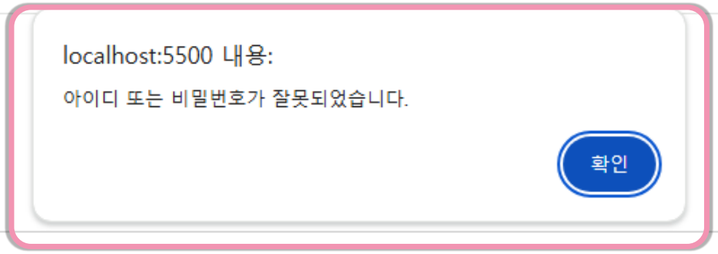
02 회원가입 페이지로 이동

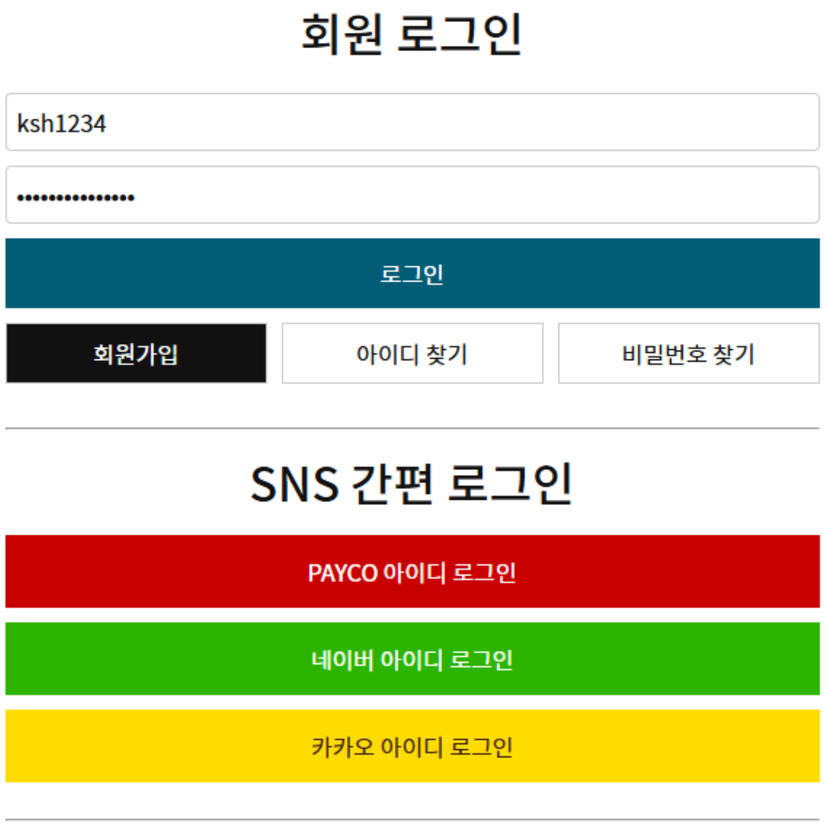
03 아이디 찾기 페이지로 이동

04 비밀번호 찾기 페이지로 이동

14. 로그인

01

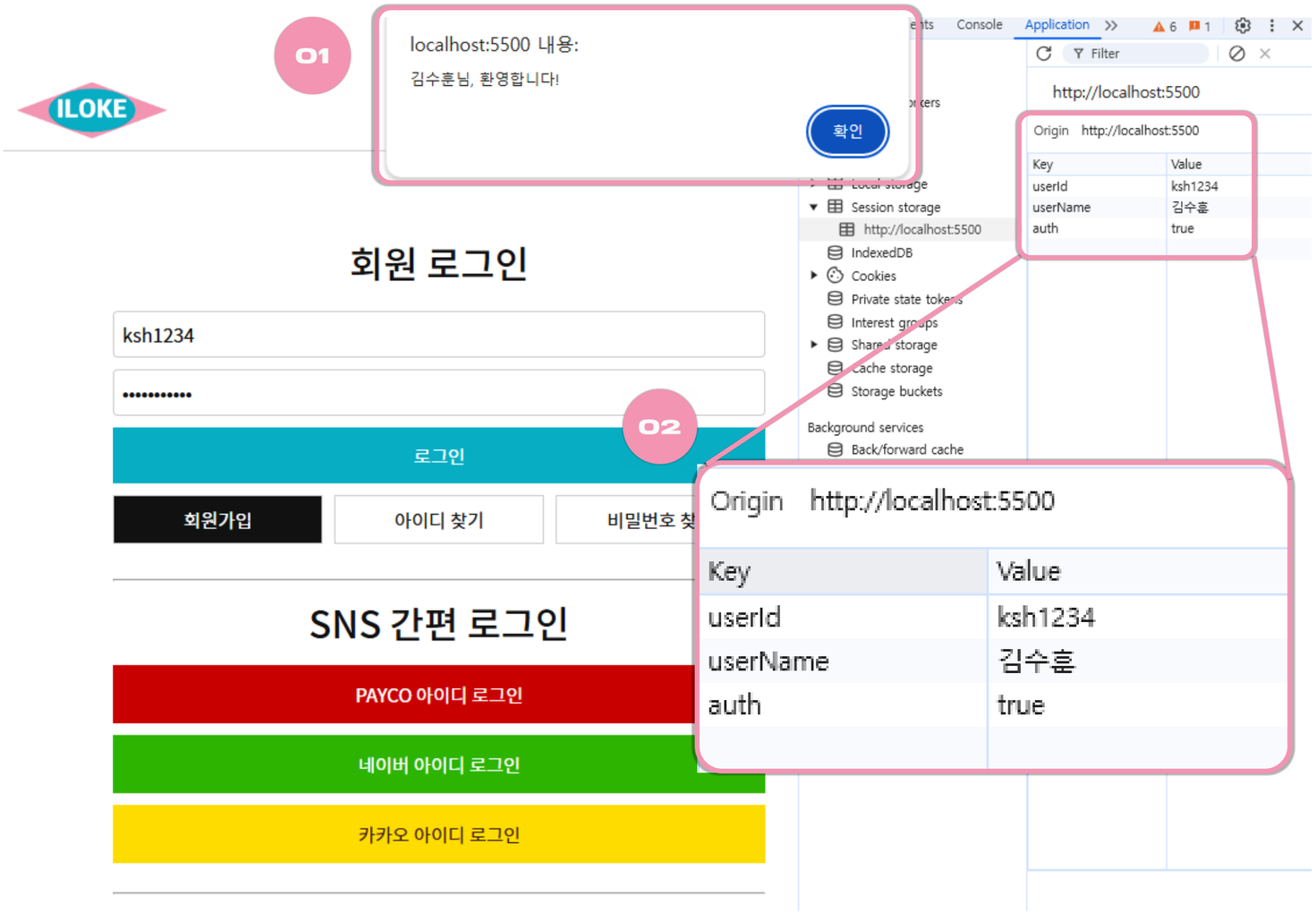




Description List

01 잘못된 아이디와 비밀번호 입력 시 alert 메시지 출력.

14. 로그인

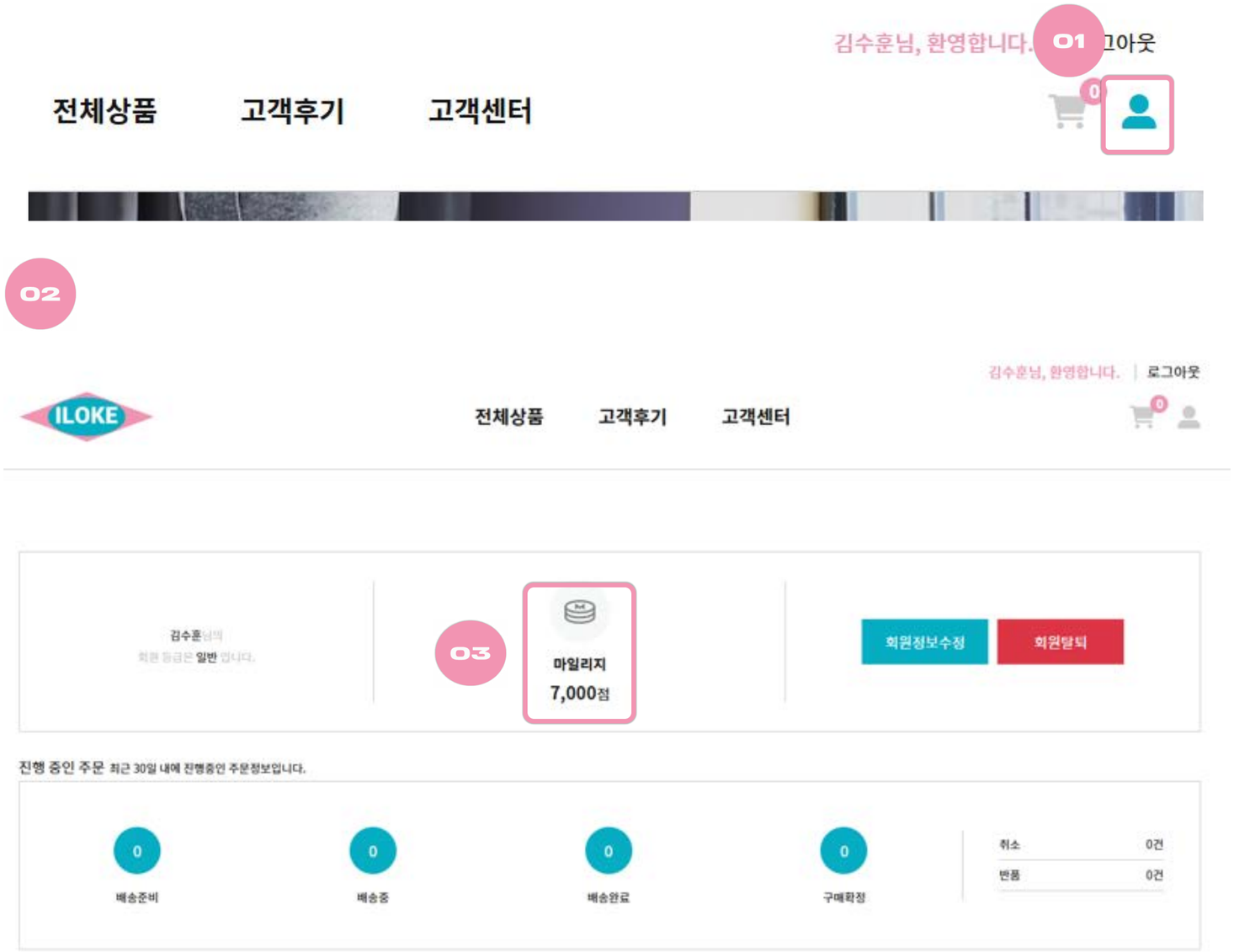


Description List

01 아이디와 비밀번호 일치 시 회원 이름과 함께 환영 메시지 출력 후 로그인 메인 페이지로 이동

02 로그인과 동시에 세션 스토리지에 회원 아이디, 회원이름이 저장

15. 회원 마일리지 조회



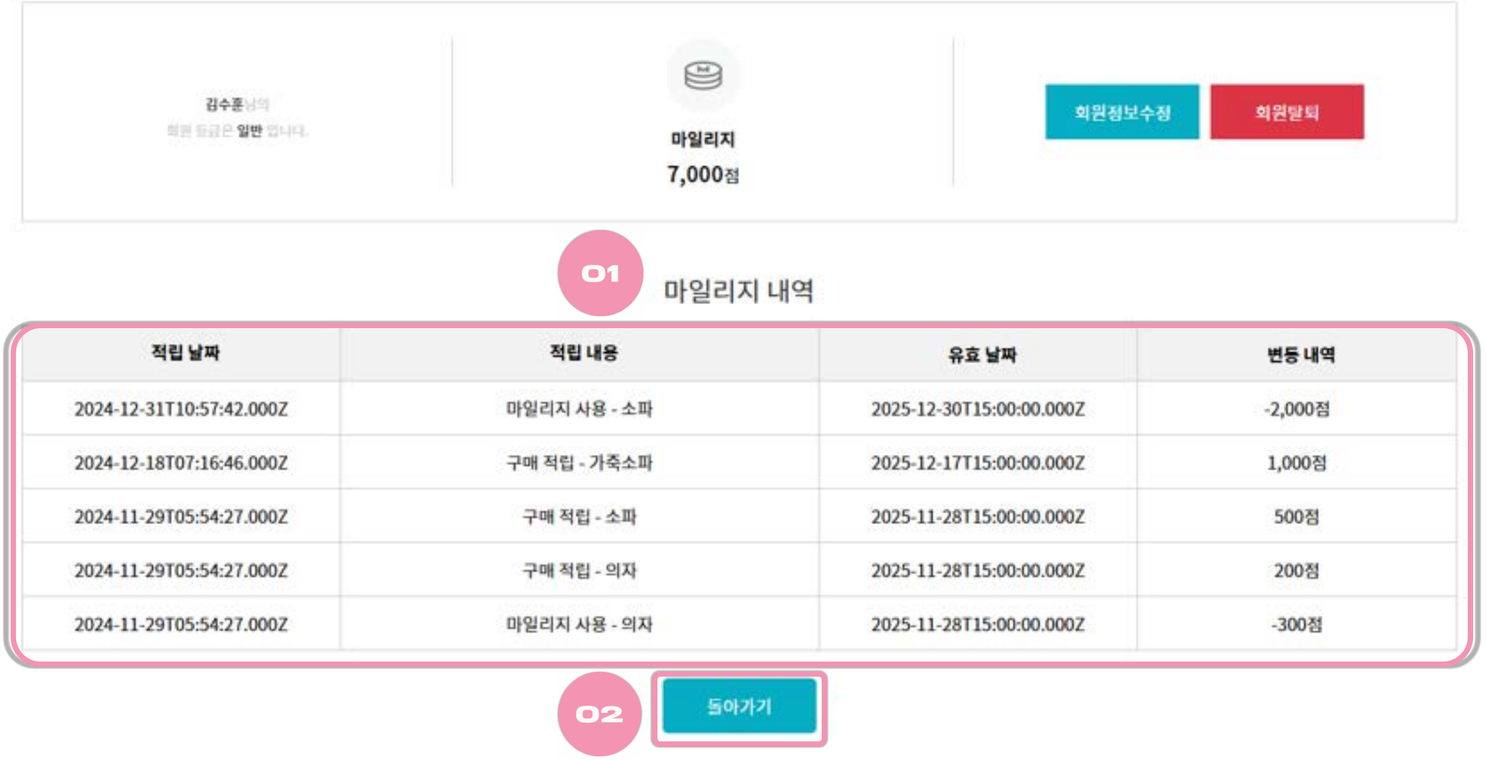
Description List

01 사람 아이콘 클릭 시 마이 페이지로 이동

02 마이 페이지

03 점수 클릭 시 마일리지 내역 출력

15. 회원 마일리지 조회



적립 날짜	적립 내용	유효 날짜	변동 내역
2024-12-31T10:57:42.000Z	마일리지 사용 - 소파	2025-12-30T15:00:00.000Z	-2,000점
2024-12-18T07:16:46.000Z	구매 적립 - 가죽소파	2025-12-17T15:00:00.000Z	1,000점
2024-11-29T05:54:27.000Z	구매 적립 - 소파	2025-11-28T15:00:00.000Z	500점
2024-11-29T05:54:27.000Z	구매 적립 - 의자	2025-11-28T15:00:00.000Z	200점
2024-11-29T05:54:27.000Z	마일리지 사용 - 의자	2025-11-28T15:00:00.000Z	-300점

Description List

01 회원 마일리지 내역 출력
(적립 날짜, 적립 내용, 유효 날짜, 변동 내역)

02 돌아가기 버튼
클릭 시 기존 배송 정보 출력

16. 회원 탈퇴

Description List

01 회원 탈퇴 안내문

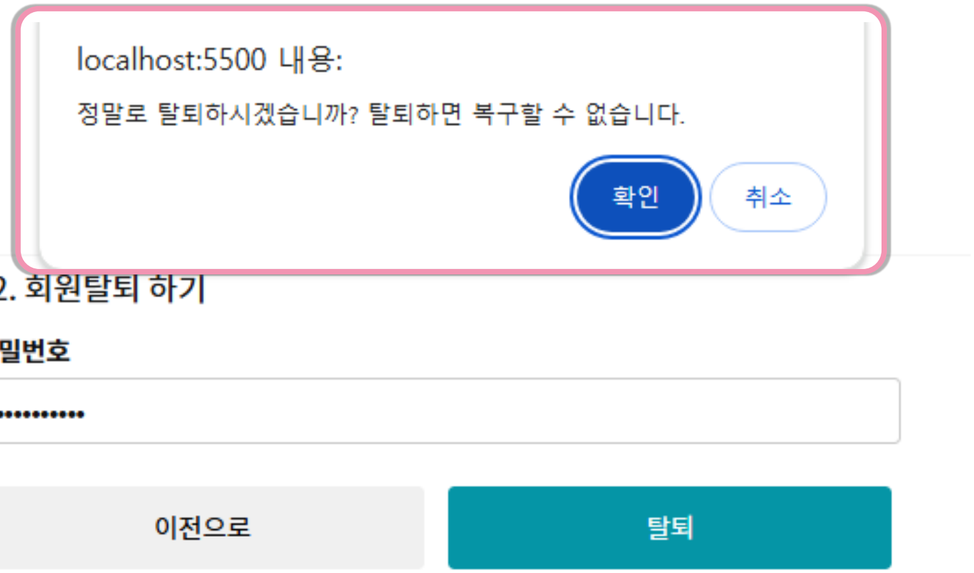
02 비밀번호 입력

03 이전으로 버튼(마이 페이지로 이동)

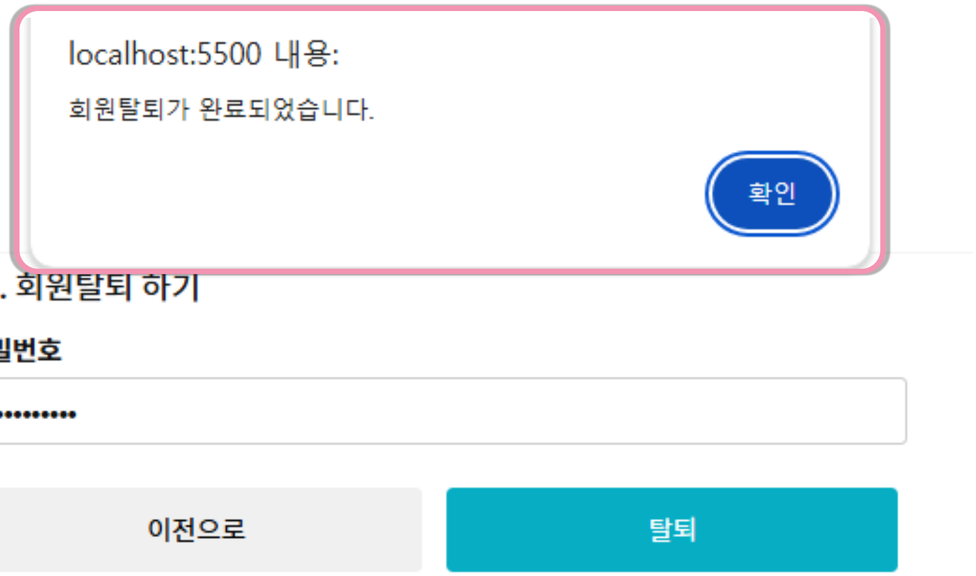
04 회원 탈퇴 버튼(비밀번호 일치 시 회원 탈퇴)

16. 회원 탈퇴

01



02



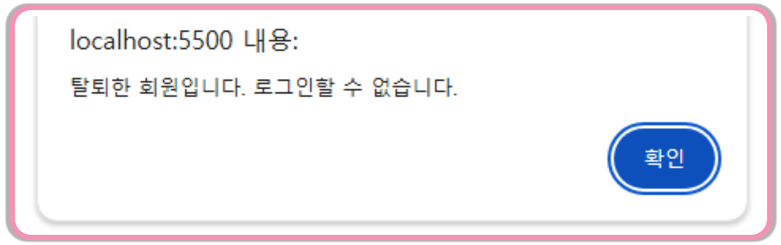
Description List

01 정말로 탈퇴 하실지 alert 메시지 출력

02 회원 탈퇴 완료
(확인 누를 시 자동 로그아웃 및 메인 페이지 이동)

16. 회원 탈퇴

01

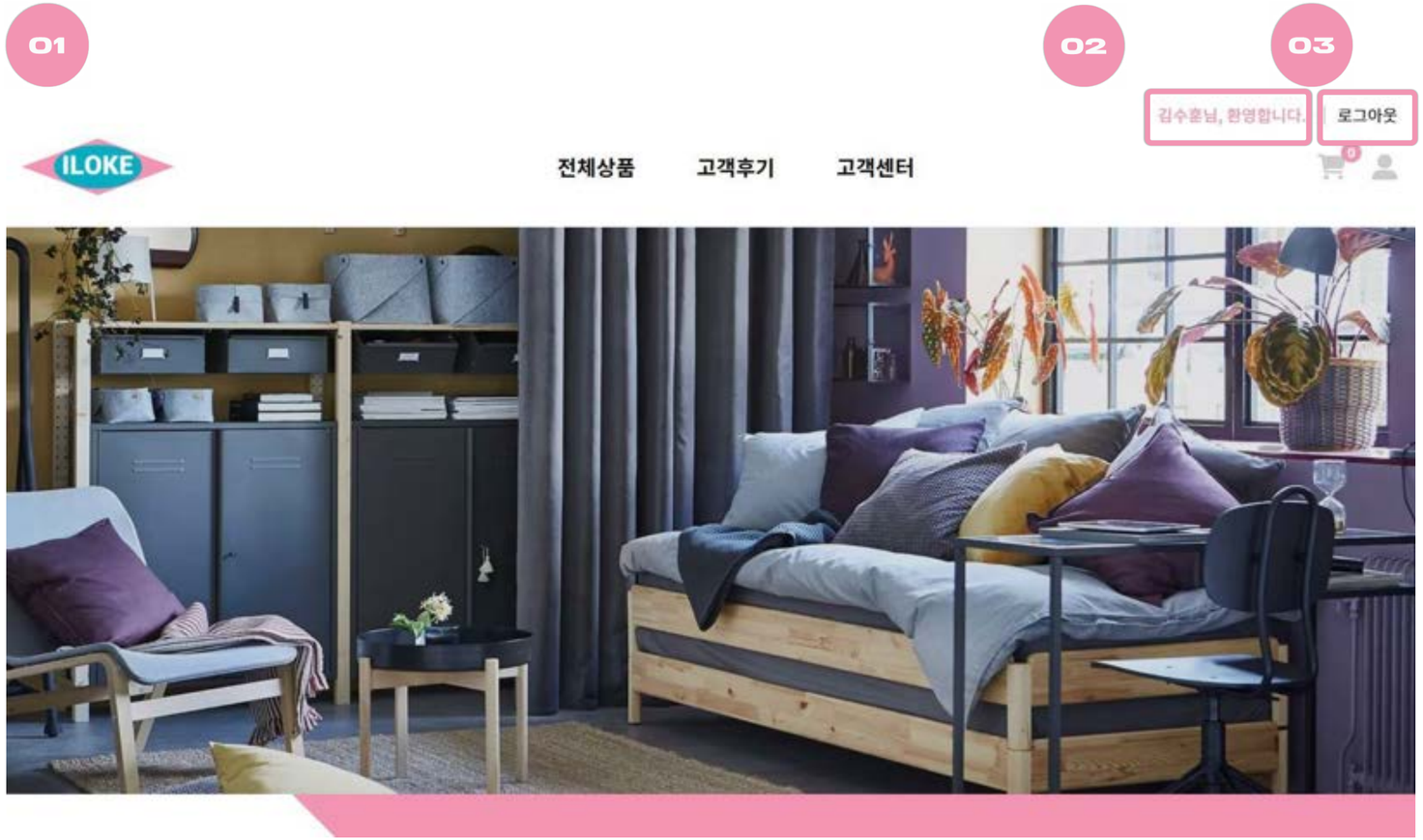


Description List

01 재 로그인 시 탈퇴 회원은 로그인 불가

02 수정된 회원 데이터
탈퇴 시 회원 -> 탈퇴회원으로 변경
로그인 불가

17. 로그인



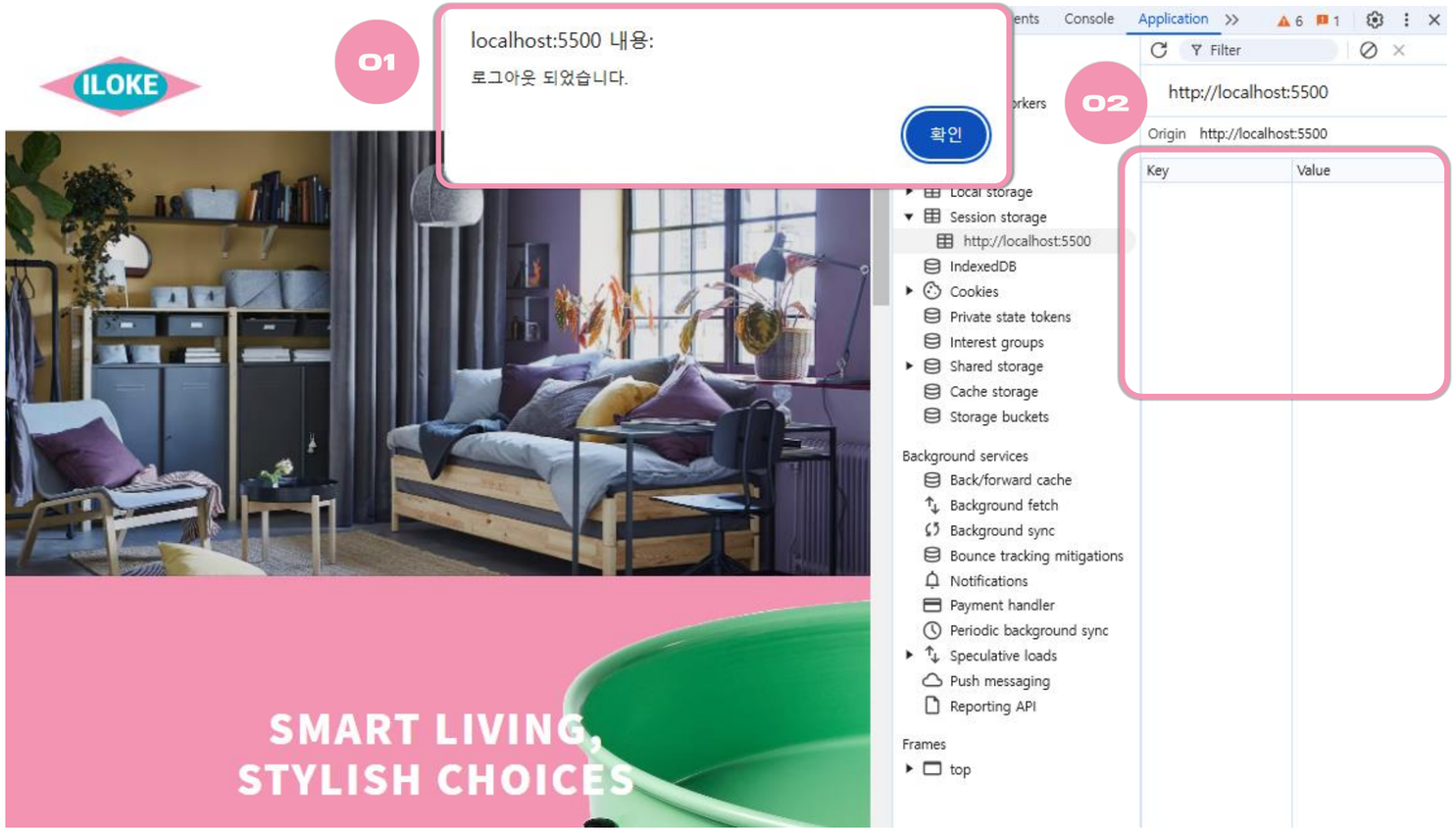
Description List

01 로그인 된 메인 페이지

02 회원 이름 출력과 환영 메시지

03 로그인 시 로그아웃 버튼 활성화

18. 로그아웃



Description List

01 로그아웃 클릭 시 로그아웃 되었다는 alert 메시지 출력

02 로그아웃과 동시에 세션 스토리지에 저장되어 있던 회원 이름과 회원 아이디 클리어

19. 로그인 핵심 코드

01

```
router.post('/', async (req, res) => {
    const { id, pw } = req.body;

    try {
      // 입력받은 아이디에 해당하는 회원 정보를 데이터베이스에서 조회
      const sql = 'SELECT * FROM mem_info WHERE id = ?';
      const [rows] = await conn.execute(sql, [id]);

       // 아이디가 존재하지 않을 경우 에러 응답 반환
      if (rows.length === 0) {
        return res.status(401).json({ success: false, message: '아이디가 존재하지 않습니다.' });
      }

      const user = rows[0]; // 조회된 회원 정보

      // 탈퇴한 회원인지 확인
      if (user.role === '탈퇴회원') {
        return res.status(403).json({ success: false, message: '탈퇴한 회원입니다. 로그인할 수 없습니다.' });
      }

      // 비밀번호 검증 (평문 비교인 경우)
      if (user.pw !== pw) {
        return res.status(401).json({ success: false, message: '비밀번호가 일치하지 않습니다.' });
      }

      // 로그인 성공 시 세션에 userId 저장
      req.session.userId = user.id;

      // console.log('세션에 저장된 userId:', req.session.userId);

      res.json({ success: true, message: '로그인 성공!', user: { id: user.id, name: user.name } });
    } catch (error) {
      console.error('로그인 오류:', error);
      res.status(500).json({ success: false, message: '서버 오류가 발생했습니다.' });
    }
  });

  module.exports = router;
```

Description List

01 사용자 로그인 처리 라우터

19. 로그인 핵심 코드

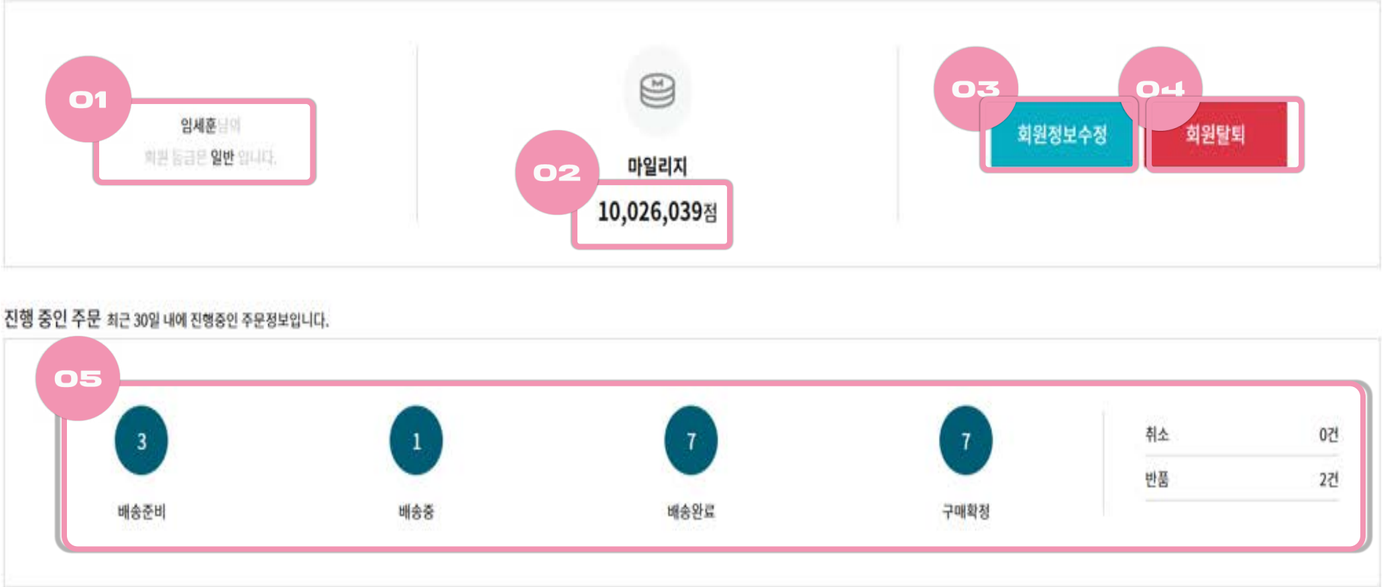
01

```
const Login = () => {

  const handleLogin = async (e) => {
    e.preventDefault();

    try {
      // 서버로 로그인 요청을 보냄 (memLogin 함수 호출)
      const response = await memLogin(id, pw);

      if (response.data.success) {
        // 로그인 성공 시 서버에서 반환된 사용자 이름을 추출
        const userName = response.data.user.name;

        // 세션 스토리지에 사용자 정보 저장
        sessionStorage.setItem('userId', id);
        sessionStorage.setItem('userName', userName);
        sessionStorage.setItem('auth', 'true');

        // storage 이벤트를 수동으로 발생시킴
        window.dispatchEvent(new Event('storage'));

        alert(`${userName}님, 환영합니다!`);

        navigate('/home'); // 홈 페이지로 이동
        //window.location.reload();
      } else {
        alert(response.data.message || '로그인에 실패했습니다.');
      }
    } catch (error) {
       if (error.response) {
      // 403 상태 코드 처리
      if (error.response.status === 403) {
        alert('탈퇴한 회원입니다. 로그인할 수 없습니다.');
      } else if (error.response.status === 401) {
        alert('아이디 또는 비밀번호가 잘못되었습니다.');
      } else {
        alert('서버 오류가 발생했습니다. 다시 시도해 주세요.');
      }
    } else {
      alert('네트워크 오류가 발생했습니다. 다시 시도해 주세요.');
    }
  }
};
```

Description List

01 사용자가 입력한 아이디와 비밀번호 인증 후 성공 시 세션 스토리지에 사용자 정보를 저장 및 로그인 하게하는 함수

２０. 마이페이지



Description List

01 회원에 대한 정보를 DB에서 불러옴

02 보유한 마일리지양을 DB에서 불러옴
금액 클릭시 마일리지 상세내역 조회가능

03 회원정보수정기능

04 회원탈퇴기능

05 주문을 주문상태에 따라 진행상황을 확인가능

２０. 마이페이지

임세훈님의
회원 등급은 일반 입니다.

01

마일리지
10,026,039점

회원정보수정 회원탈퇴

마일리지 내역

02

적립 날짜	적립 내용	유효 날짜	변동 내역
2024-12-30T06:46:48.000Z	주문 취소로 인한 마일리지 반환	2024-12-29T15:00:00.000Z	20,000점
2024-12-30T06:46:07.000Z	구매확정 마일리지 적립	2024-12-29T15:00:00.000Z	15,580점
2024-12-30T06:44:39.000Z	상품 구매 마일리지 사용	2024-12-29T15:00:00.000Z	-50,000점
2024-12-30T02:42:07.000Z	주문 취소로 인한 마일리지 반환	2024-12-29T15:00:00.000Z	3,000점
2024-12-30T02:41:23.000Z	구매확정 마일리지 적립	2024-12-29T15:00:00.000Z	30,400점
2024-12-30T02:39:49.000Z	상품 구매 마일리지 사용	2024-12-29T15:00:00.000Z	-20,000점
2024-12-30T01:26:39.000Z	주문 취소로 인한 마일리지 반환	2024-12-29T15:00:00.000Z	1,800점
2024-12-30T01:25:39.000Z	구매확정 마일리지 적립	2024-12-29T15:00:00.000Z	28,060점
2024-12-30T01:23:23.000Z	상품 구매 마일리지 사용	2024-12-29T15:00:00.000Z	-1,000점
2024-12-29T18:23:19.000Z	상품 구매 마일리지 사용	2024-12-29T15:00:00.000Z	-1,800점
2024-12-29T11:17:43.000Z	구매확정 마일리지 적립	2024-12-28T15:00:00.000Z	76,773점
2024-12-29T11:09:08.000Z	구매확정 마일리지 적립	2024-12-28T15:00:00.000Z	76,773점

03

돌아가기

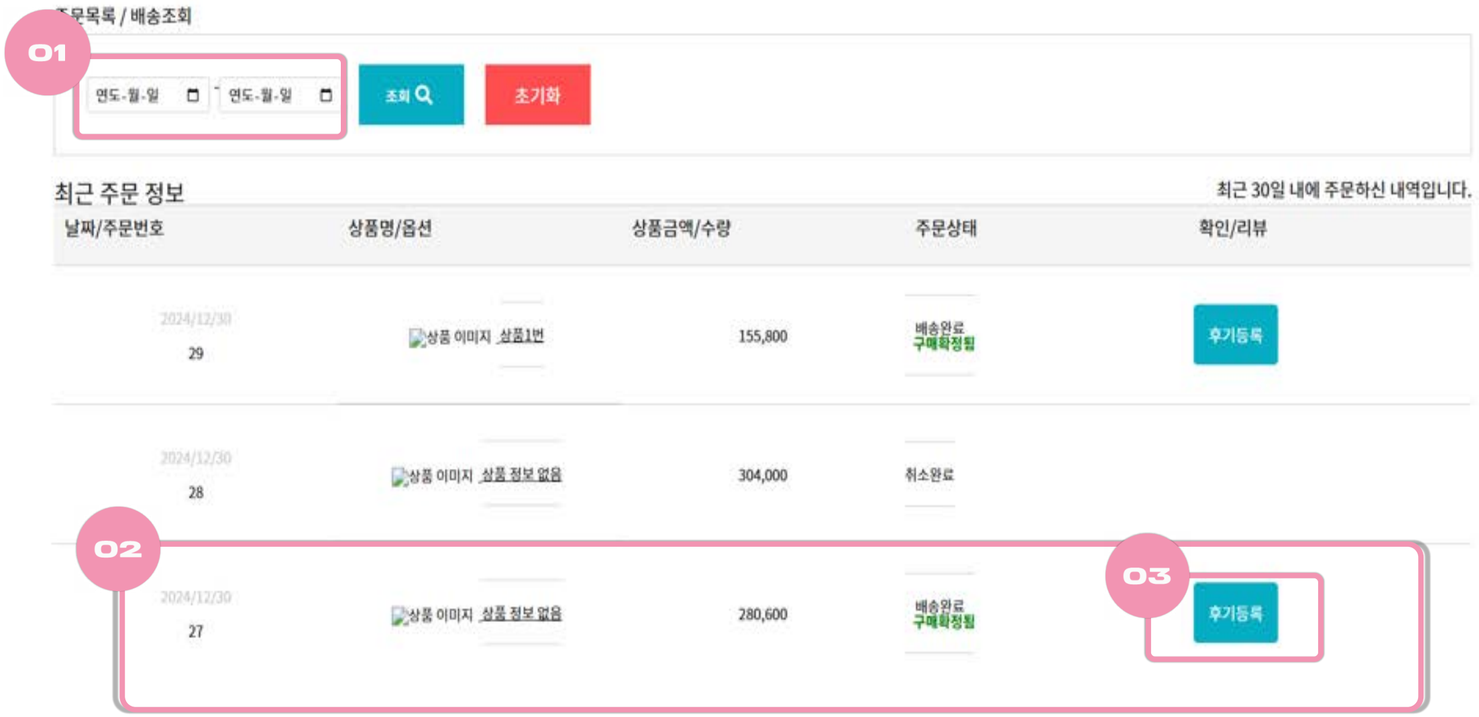
Description List

01 보유한 마일리지양을 DB에서 불러옴
금액 클릭시 마일리지 상세내역 조회가능

02 DB에서 불러온 마일리지 변동내역

03 돌아가기 버튼을 통해 기존 마이페이지로 이동

２０. 마이페이지



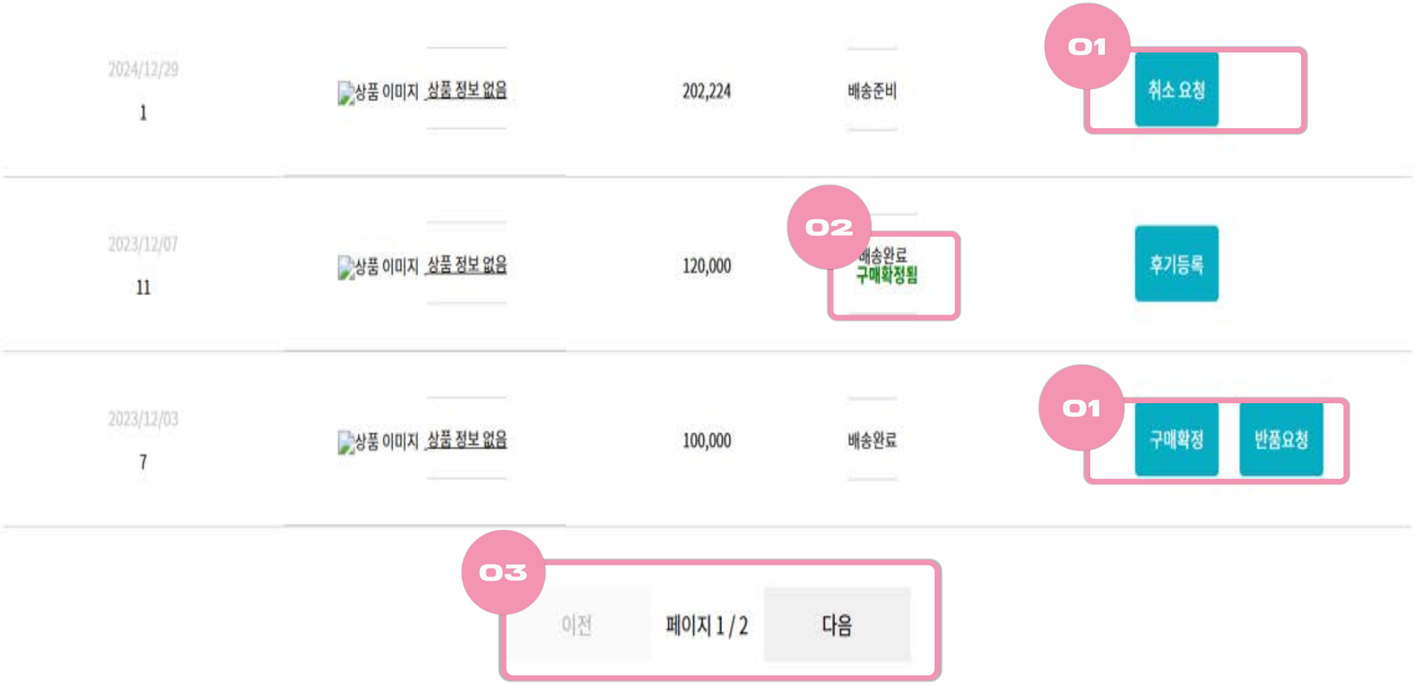
Description List

01 기간으로 주문조건을 검색할 수 있는 필터

02 주문을 데이터로 불러와 기간순으로 정렬

03 주문상태에 따른 버튼 생성
배송준비,배송중: 취소요청
배송완료(구매확정): 후기등록
배송완료(구매미확정): 구매확정,반품요청

２０. 마이페이지



Description List

01 주문상태 변경에 따른 버튼 생성

02 구매확정시 구매확정표시

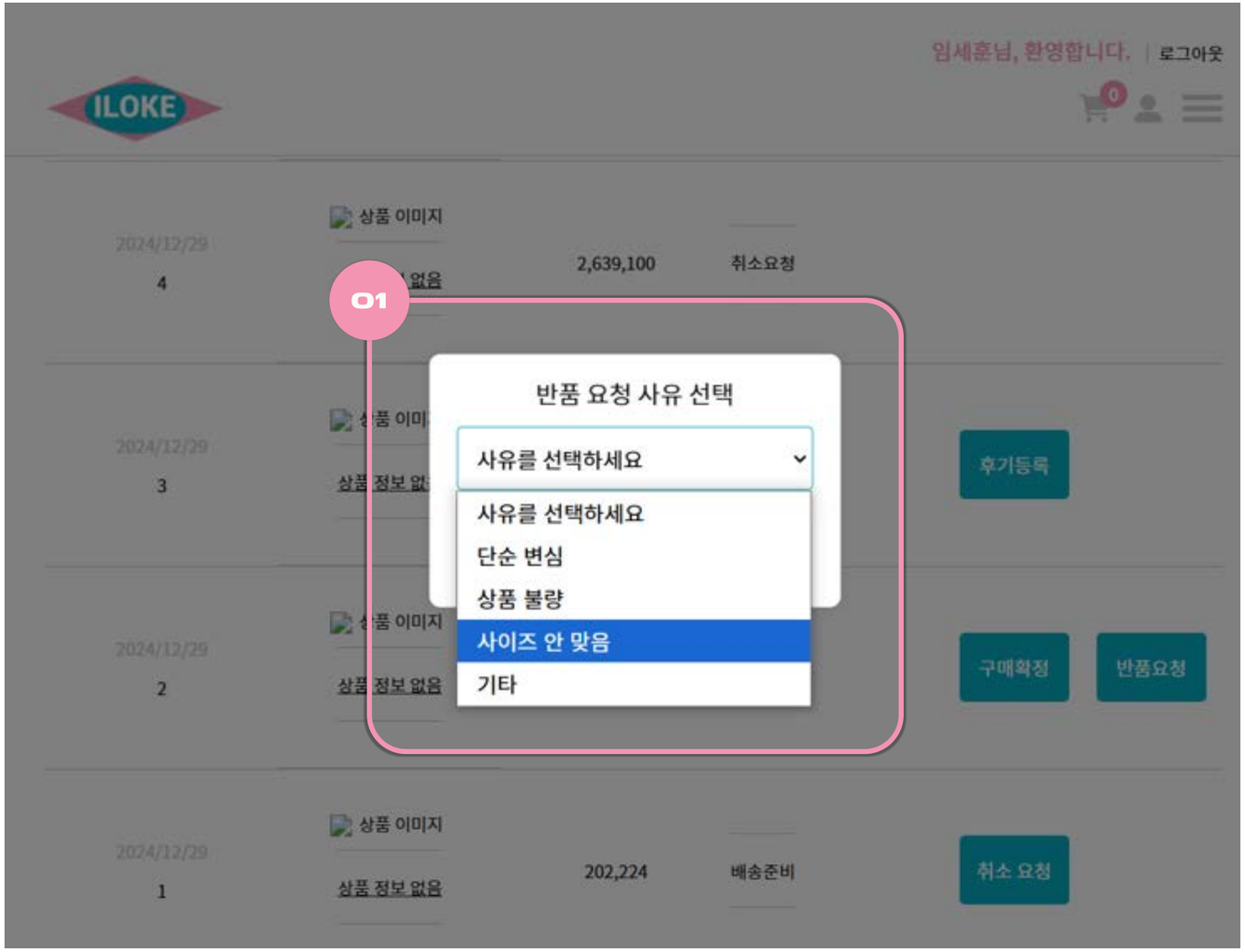
03 페이지네이션 기능

２０. 마이페이지

Description List

01 취소요청 모달창과 함께 사유선택
사유와 요청시기가 데이터로 등록됨

２０. 마이페이지

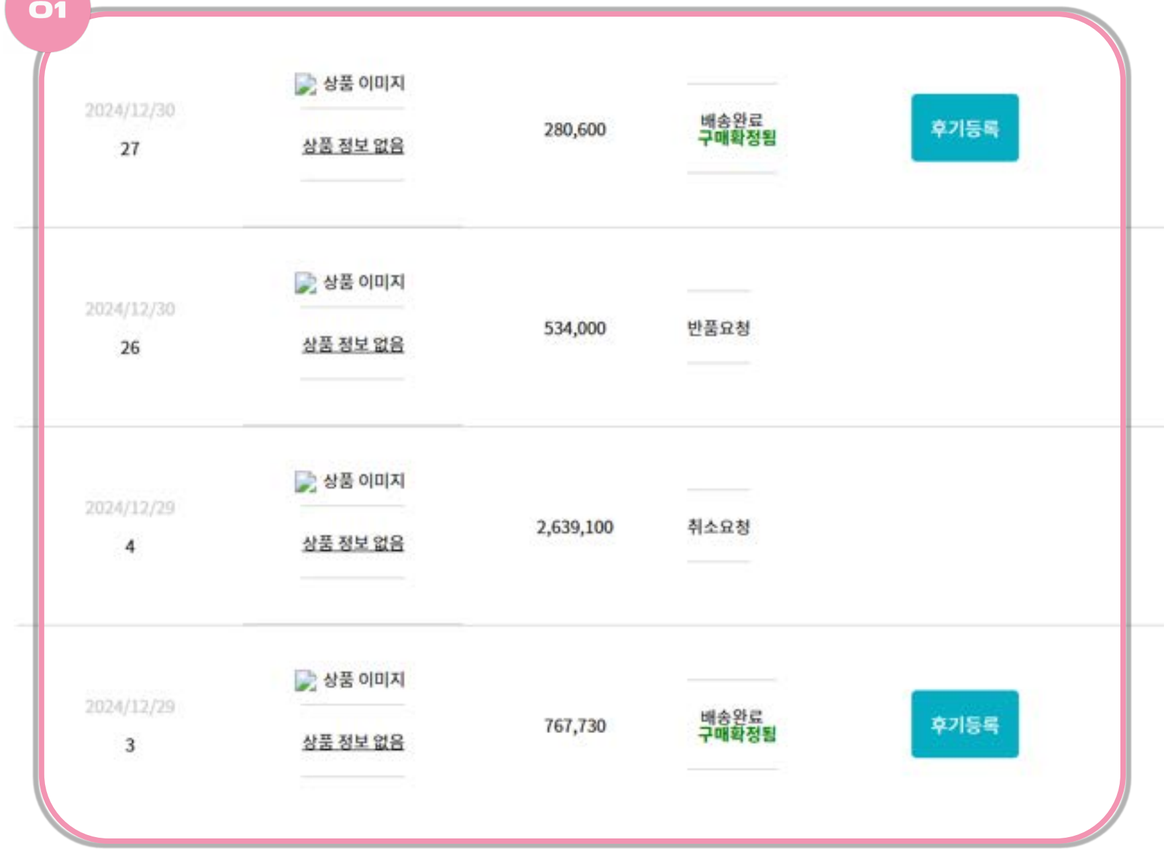


Description List

01 반품요청 모달창과 함께 사유선택
사유와 요청시기가 데이터로 등록됨

２０. 마이페이지

01



Description List

01 사유를 선택하고 확인버튼 클릭시
반품요청,취소요청 상태로 변경

２０. 마이페이지

Description List

01 요청사유와 모달창과 관련된 코드

02 페이지네이션 관련 코드

03 주문상태에 따른 숫자변경 코드

01

```
const [modalVisible, setModalVisible] = useState(false);
const [selectedOrder, setSelectedOrder] = useState(null);
const [reason, setReason] = useState('');
const [requestType, setRequestType] = useState(''); // 취소 or 반품 구분
```

02

```
const [totalOrders, setTotalOrders] = useState(0); //  총 주문 수 추가
const [currentPage, setCurrentPage] = useState(1); //  현재 페이지 추가
const ordersPerPage = 10; //  한 페이지당 10개의 주문
```

03

```
const [startDate, setStartDate] = useState("");
const [endDate, setEndDate] = useState("");
const [recentOrders, setRecentOrders] = useState([]);
const [orderCounts, setOrderCounts] = useState({
  배송준비: 0,
  배송중: 0,
  배송완료: 0,
 구매확정:0
});

const [returnCancelCounts, setReturnCancelCounts] = useState({
  취소: 0,
  반품: 0,
});
```

２０. 마이페이지

01

```
onst handleConfirmRequest = async () => {
if (!selectedOrder || !reason) {
  alert("사유를 선택해주세요.");
  return;
}

const requestData = {
  orderId: selectedOrder.id,
  reason: reason,
  requestType: requestType
};

try {
  const response = await uRequestCancelOrReturn(memberId, requestData); //  API 호출

  if (response.data.success) {
    alert(`${requestType} 요청이 완료되었습니다.`);
```

02

```
export const uRequestCancelOrReturn = (memberId, requestData) =>
  API.patch('/mypage/request',
    requestData,
    {
      headers: { 'user-id': memberId }
    }
  );
```

Description List

01 요청사유 전달 코드

02 백서버에 데이터를 요청하는 api

２０. 마이페이지

01

```
const [userInfo, setUserInfo] = useState({ name: '***', m_remain: 0, type: '일반회원등급' });
```

02

```
const cancelReasons = ['단순 변심', '상품 불량', '배송 지연', '기타'];
const returnReasons = ['단순 변심', '상품 불량', '사이즈 안 맞음', '기타'];
```

02

```
st memberId = sessionStorage.getItem('userId');

const [isLoading, setIsLoading] = useState(true); // 로딩 상태 추가

useEffect(() => {
  if (!memberId) {
    alert('로그인 정보가 없습니다. 다시 로그인 해주세요.');
    window.location.href = '/login';
  } else {
    setIsLoading(false);
  }
}, [memberId]);
```

Description List

01 회원정보받아오는 state 설정

02 취소사유와 반품사유 설정

03 Session값으로 사용자 정보를 받아와
비로그인시 접근 불가능하게 만듬

２０. 마이페이지

01

```
// 마일리지 내역 가져오기
const fetchMileageHistory = async () => {
  setLoading(true);  // 로딩 시작
  try {
    const response = await rFetchMileageHistory(memberId);
    setMileageHistory(response.data.mileageHistory);
    setShowMileageTable(true);

  } catch (error) {
    console.error(" 마일리지 내역 가져오기 오류:", error.message);
    alert("마일리지 정보를 가져오는 중 오류가 발생했습니다.");
  } finally {
    setLoading(false);  // 로딩 종료
  }
};
```

02

```
const [showMileageTable, setShowMileageTable] = useState(false);
const [mileageHistory, setMileageHistory] = useState([]);
```

Description List

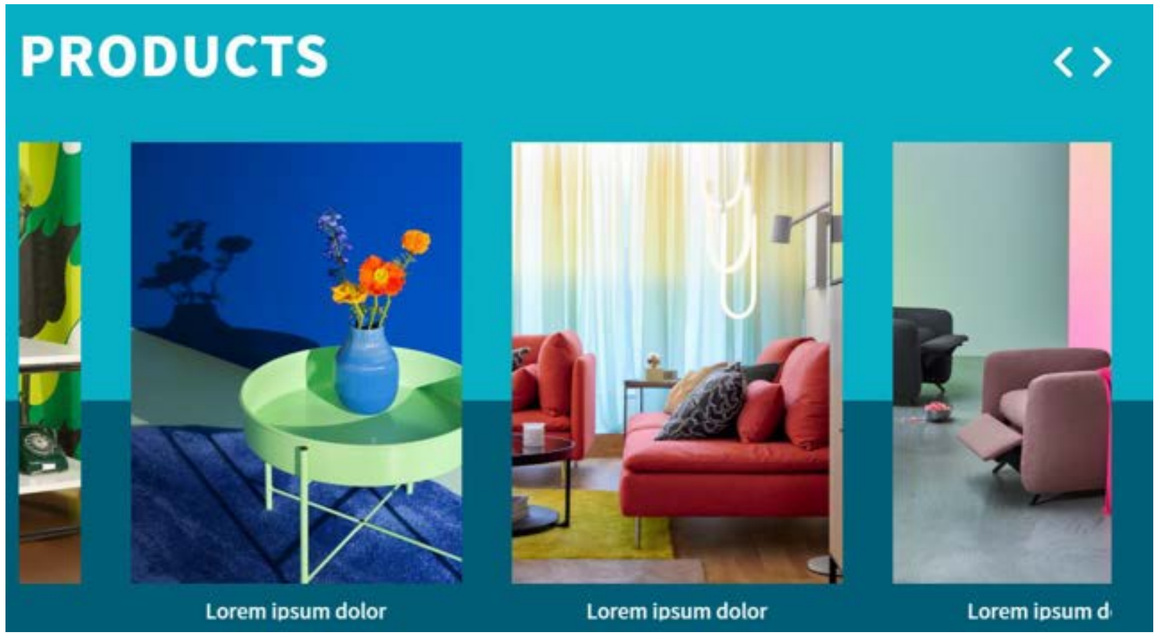
01 마일리지 내역 조회 함수

02 마일리지 내역 조회를 위한 state 설정

21. 쇼핑몰 메인화면 스와이퍼

01

02



01 메인배너

02 section3 슬라이드

21. 쇼핑몰 메인화면 스와이퍼 - 메인배너

01 html – js 형식 스와이퍼를 리액트 방식으로 변경

01

```
<Swiper
  tag='section'
  className='main-banner main-con'
  wrapperTag='div'
  loop= {true}
  autoPlay = {
      {delay: 4000 ,disableOnInteraction: false}
  }
  modules={[EffectFade]}
  effect="fade"
  speed = {1000}
  >
      <SwiperSlide className="banner-img01"></SwiperSlide>
      <SwiperSlide className="banner-img02"></SwiperSlide>
      <SwiperSlide className="banner-img03"></SwiperSlide>
      <SwiperSlide className="banner-img04"></SwiperSlide>
      <SwiperSlide className="banner-img05"></SwiperSlide>
</Swiper>
```

21. **쇼핑몰 메인화면 스와이퍼** - section3 슬라이드01

01

```
<Swiper
  tag='div'
  className='product-swipe'
  wrapperTag='ul'
  loop= {true}
  slidesPerView = {1}
  spaceBetween = {30}
  modules={{Navigation}}
  navigation={{
    //스와이퍼 외부에 컨트롤러 설정하기
    prevEl: prevRef.current,
    nextEl: nextRef.current,
  }}
  onBeforeInit={(swiper) => {
  // 초기화 전에 네비게이션 버튼을 swiper에 할당
  swiper.params.navigation.prevEl = prevRef.current;
  swiper.params.navigation.nextEl = nextRef.current;
  }}
  autoPlay = {
      { delay: 3000 ,
        disableOnInteraction: false}
  }
  speed = {1000}
  breakpoints={{
    1200: { slidesPerView: 3, spaceBetween : 60 },
    800: { slidesPerView: 2, spaceBetween : 30 }
  }}
  >
```

01 html – js 형식 스와이퍼를 리액트 방식으로 변경

21. **쇼핑몰 메인화면 스와이퍼** - section3 슬라이드02

02

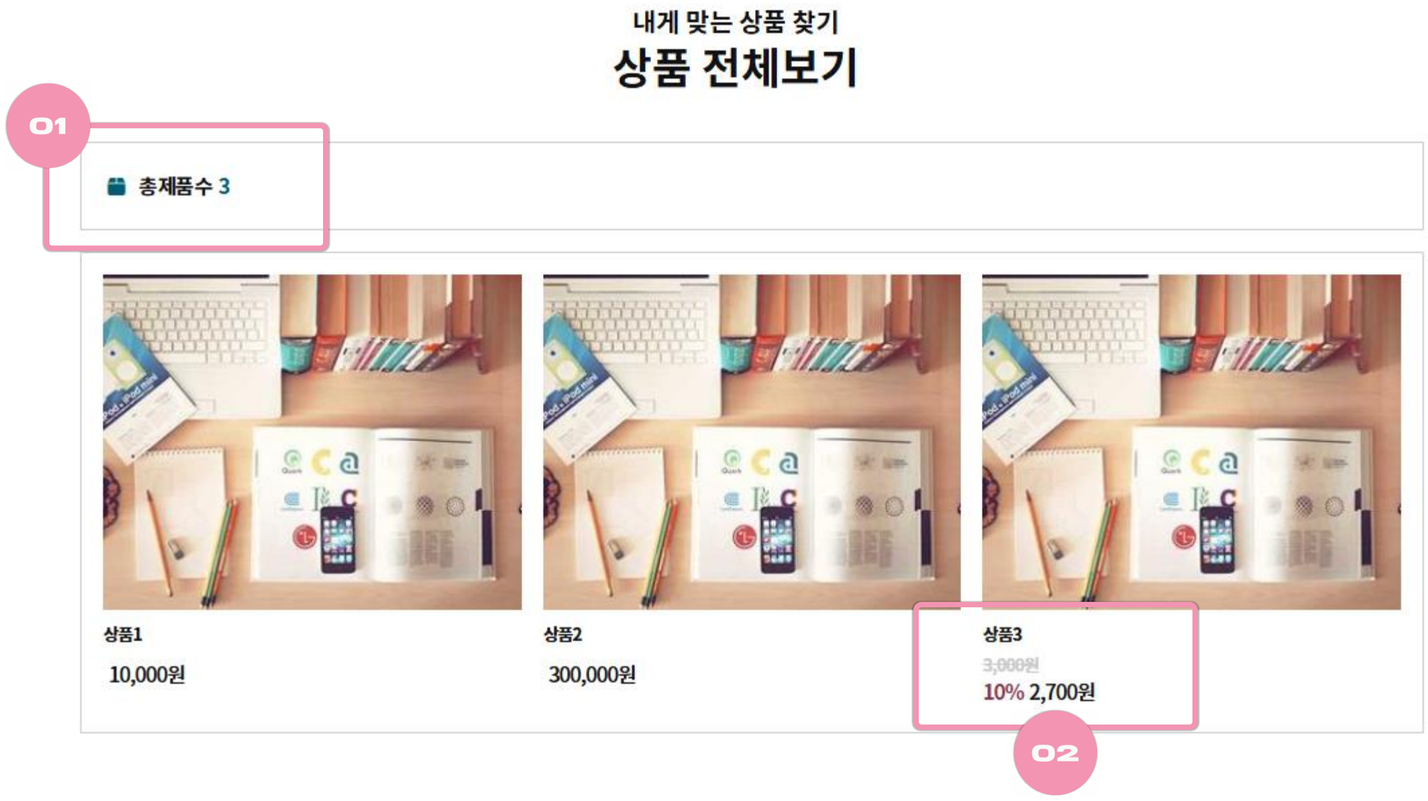
```
  >
    {[
    "main_product09.jpg",
    "main_product10.jpg",
    "main_product11.jpg",
    "main_product12.jpg",
    "main_product13.jpg",
    "main_product14.jpg",
    "main_product15.jpg",
    "main_product16.jpg",
  ].map((image, index) => (
    <SwiperSlide
      tag='li'
      key={index}>
      <Link to={'/'}>
        <span className="product-img">
          <img
            src={`./content/img/main/${image}`}
            alt={`Product ${index + 1}`}
          />
        </span>
        <span className="product-tit">Lorem ipsum dolor</span>
      </Link>
    </SwiperSlide>
  ))}
</Swiper>
```

02 이미지를 map 반복문 사용하여 스와이퍼에 적용

22. 쇼핑몰 상품 리스트

01 등록된 상품 개수

02 등록된 상품 정보



22. 쇼핑몰 상품 리스트

01

```
const [listVal, setListVal] = useState([]);//데이터 초기값
const [chkVal, setChkVal] = useState(null);//옵션 체크박스 초기값
let listLength = listVal.length;//리스트 개수
let listTxt = listLength;
```

02

```
// 데이터 연결
useEffect(()=>{
    if(!listLength || listLength == 0){
        listTxt = 0
    }
    goodsSelect()
    .then(res =>{
        // console.log(res.data);// data 확인 - 배열
        setListVal(res.data)
    })
    .catch(err => {
        console.error('상품리스트 db 프론트 불러오기 오류 ', err)
    })
},[listLength])
```

01 초기값 설정

02 등록된 상품 목록 들고오기

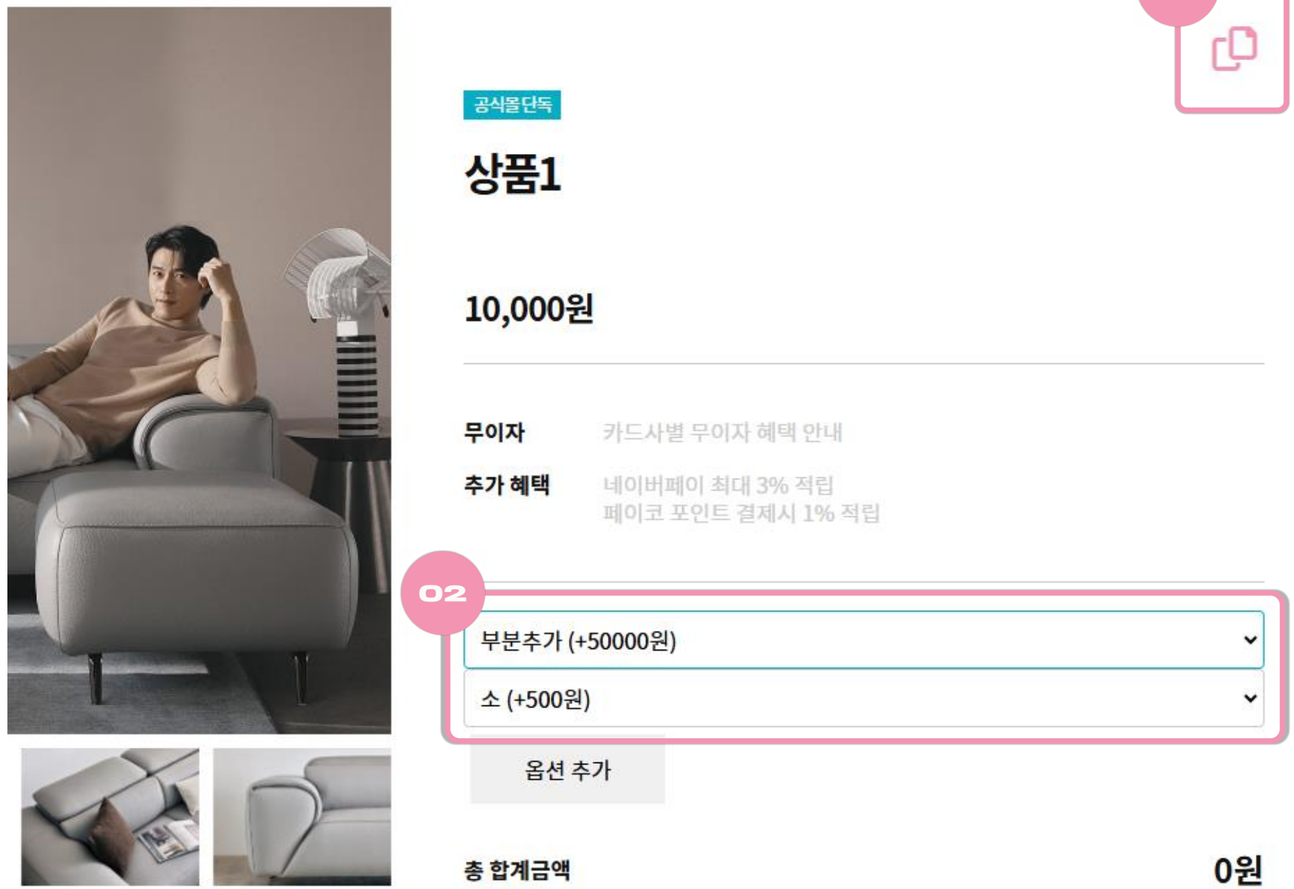
22. 쇼핑몰 상품 리스트

03 저장된 상품 정보 가져오기

03

```
<Link to={`/goodsDetail/${item.id}`} className={styles.product_item} key={idx}>
    <div className={styles.product_image_link}>
            <img src="https://picsum.photos/300/240" alt="protoImage"/>
            <p>{item.name}</p>
        <span className={styles.original_price}>{slash}</span>
        <div className={styles.price_container}>
            <div className={styles.price_details}>
                <b className={styles.discount_percent}>{discount}</b>
                <strong className={styles.discounted_price}>{newFnPrice.toLocaleString()}원</strong>
            </div>
        </div>
    </div>
</Link>
```

23. 쇼핑몰 상품 상세



01 현재 보고있는 상품 상세 페이지 링크 복사하기

02 선택된 옵션/키워드 목록, 가격 불러오기

23. 쇼핑몰 상품 상세

01

```
const [goodsInfo, setGoodsInfo] = useState("")//상품정보
const [thumbsSwiper, setThumbsSwiper] = useState(null);//슬라이드 썸네일

const [orPrice, setOrPrice] = useState('');//원가 문자열(ex 10,000)
const [fnPrice, setfnPrice] = useState(0)//숫자열로 변경된 최종가

const [opOne, setOpOne] = useState(null)//옵션1(1차 작업명 color-option)
const [opTwo, setOpTwo] = useState(null)//옵션2(1차 작업명 leather-option)
const [opThree, setOpThree] = useState(null)//옵션3(1차 작업명 stool-option)

const [opList, setopList] = useState({})//옵션리스트
const [loading, setLoading] = useState(true)
const [error, setError] = useState(null)


const [colorOption, setColorOption] = useState("none");
const [leatherOption, setLeatherOption] = useState("none");
const [stoolOption, setStoolOption] = useState("none");
const [optionList, setOptionList] = useState([]);
const [totalPrice, setTotalPrice] = useState(0);
const [isHeartActive, setIsHeartActive] = useState(false);
const [productId, setProductId] = useState(null) // ✅ 상품 ID 상태 추가
 const memberId = sessionStorage.getItem('userId'); // 사용자 ID를 세션에서 가져옴
let optionTitArr = [];//옵션명 배열 전체 출력
let optionTit = [];//옵션명 중복 제거 배열
let optionMainArr = [];//옵션명 배열 전체 출력
let optionMain = [];//옵션명 중복 제거 배열
let optionValCal = {};//옵션값
let optionValCalAll = [];//옵션값
let optionData = {};//옵션가격
let optionName = {};//옵션명
let optionNameArr = [];//옵션명
let opValArr = {};
```

01 초기값 설정

23. 쇼핑몰 상품 상세 - 링크 복사

02 현재 주소값을 가져와서 클립보드에 복사

02

```
<button onClick={handleClipboardCopy} className={styles.clipboard_btn}
    <i className="fa-regular fa-copy"></i>
</button>
```

```
const handleClipboardCopy = () => {
  navigator.clipboard.writeText(window.location.href);
  alert("URL 주소를 복사했습니다.");
};
```

23. **쇼핑몰 상품 상세** – 옵션/키워드 목록, 가격 불러오기

```
{goodsInfo.length && !goodsInfo.opkey_value_status ? (

        <div className={styles.product_options}>{/* 옵션 */}
            {
                optionMain.map((st, i) => {//옵션 메인코드
                    console.log('Rendering optionMain:', st);
                    const opNameIdx = optionTit[i]//옵션명
                    const optionValues = opValArr[st] || [];//이걸 나눠야해
                    const optionPrices = optionData[opNameIdx] || [];//옵션금액
                    const optionNames = optionName[opNameIdx] || [];//옵션값
                    // console.log(st)
                    // console.log(optionMainArr)

                    return (
                    <select
                    // className="color_option"
                    className={
                        i === 0 ? "color_option" :
                        i === 1 ? "leather_option" : "stool_option"
                    }
                    value={
                        i === 0 ? colorOption :
                        i === 1 ? leatherOption : stoolOption
                    }
                    onChange={
                        i === 0 ? handleColorChange :
                        i === 1 ? handleLeatherChange : handleStoolChange
                    } //옵션개수에 따라 핸들러 달라짐
                    key={i}>
                        <option value="none" data-chrg="0">= {opNameIdx} =</option>
                        {
                            optionValues.map((value, idx) => (
                                <option key={idx}
                                value={st.toString()+optionValues[idx].toString()} //option의 값
                                data-chrg={optionPrices[idx]} //해당 option의 가격
                                >
                                {optionNames[idx]} (+{optionPrices[idx]}원)
                            </option>
                        ))}
                    </select>)
                })
            }

        <button onClick={handleOptionSelect}>옵션 추가</button>
    </div>
```

03

03 옵션 셀렉트 순서/개수에 따라 다른 클래스명, Value값 , 핸들러가 주어져서 각 옵션명과 가격을 가져온다.

23. **쇼핑몰 상품 상세** – 옵션/키워드 목록, 가격 불러오기

04

```
<div className={styles.option_wrap}>
    {optionList.map((option, index) =>
        (<div className={styles.option_bg} key={`${option.key}-${index}`}>
            {/* {console.log(option)} */}
            <span className={styles.option_select}>
                {option.color} / {option.leather}
                 {/* / {option.stool} */}
            </span>
            <div className={styles.option_display}>
                <div className={styles.option_quantity}>
                    {option.quantity > 1 ? (
                        <>
                            <button onClick={() => decreaseQuantity(option.key)}>-</button>
                            <span>{option.quantity}</span>
                            <button onClick={() => increaseQuantity(option.key)}>+</button>
                        </>
                    ) : (
                        <>
                            <button onClick={() => removeOption(option.key)}>삭제</button>
                            <span>{option.quantity}</span>
                            <button onClick={() => increaseQuantity(option.key)}>+</button>
                        </>
                    )}
                </div>

                <div className={styles.option_amount}>
                    <strong>{(option.price * option.quantity).toLocaleString()}</strong>원
                </div>
            </div>
        </div>)
    )}
    <div className={styles.option_calc}>
        <dl className={styles.option_total}>
            <dt>총 합계금액</dt>
            {/* <dd>{fnPriceNum}원</dd> */}
            <dd>{totalPrice.toLocaleString()}원</dd>
        </dl>
    </div>
</div>
```

04

가져온 셀렉트 값의 옵션 가격을 원가와 더해서 최종 가격으로 출력.

23. **쇼핑몰 상품 상세** – 옵션/키워드 목록, 가격 불러오기

05 선택한 옵션목록 출력 > 주문 정보로 보내짐

05

```
let optionValCalMapped = {}; // 결과 객체//여기서 옵션명 받아와야함

if(optionValCalAll && optionNameArr && optionMainArr) {
    // 두 배열을 순회하여 객체에 "조합된 키": "옵션명" 형태로 저장
    optionValCalAll.forEach((optionValue, index) => {
        const optionName = optionNameArr[index];  // 해당 인덱스의 옵션명
        const optionMain = optionMainArr[index];  // 해당 인덱스의 메인 코드

        // 새로운 키 생성: mainCode + optionValue (예: 42 + 1 = 421)
        const combinedKey = optionMain * 10 + optionValue;

        // 새로운 키와 옵션명을 객체에 매핑
        optionValCalMapped[combinedKey] = optionName;
        // console.log(optionValue);//옵션값 코드 전체다
    });
}
```

24. 게시물 - 전체후기

Description List

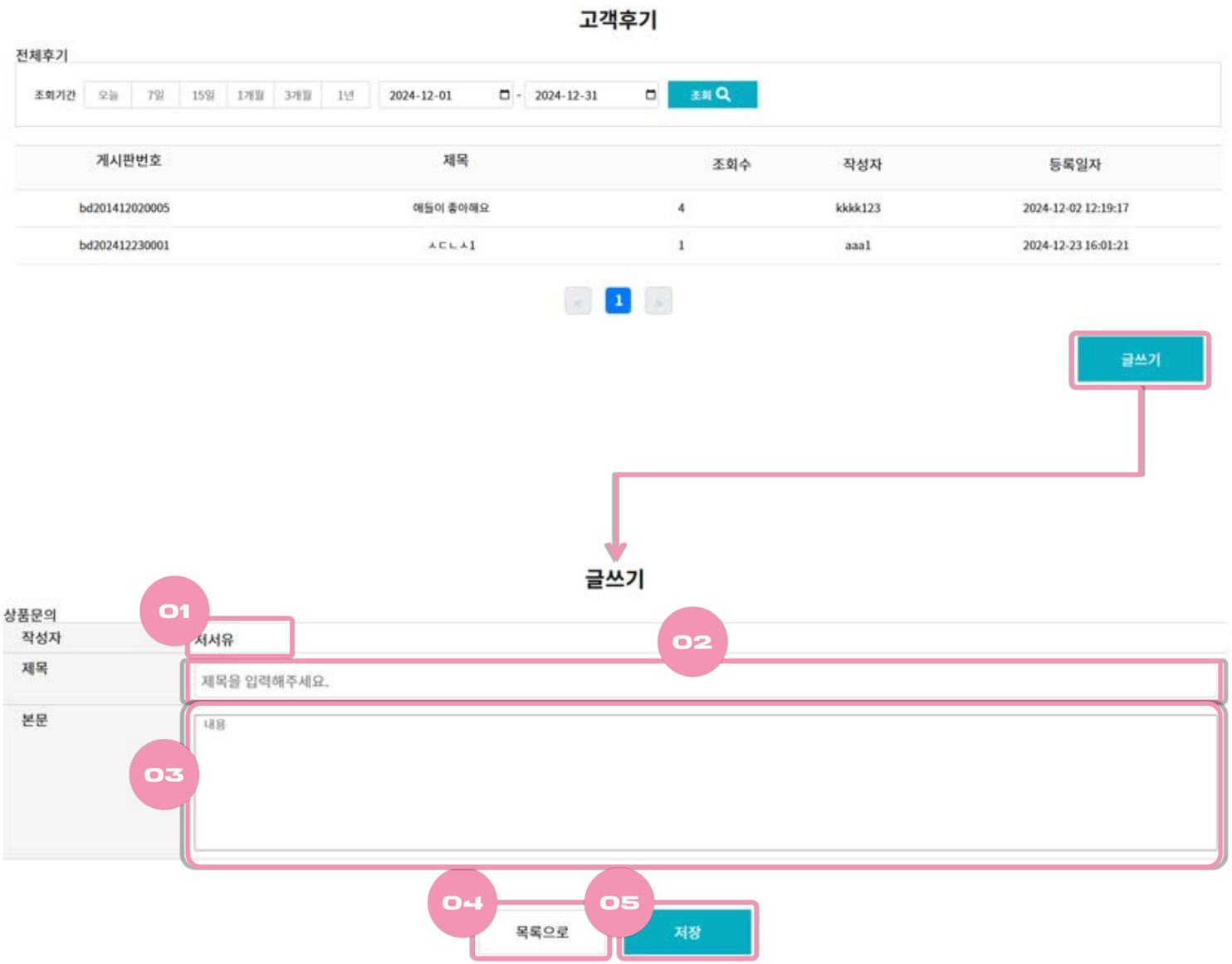
01 검색 필터
-조회기간

02 검색된 전체후기 리스트
- 게시판번호
- 제목
- 조회수
- 작성자
- 등록일자

03 페이징

04 글쓰기

24. 게시물 - 전체후기 - 글쓰기



Description List

01 작성자 : 로그인시 저장된 세션값 표시

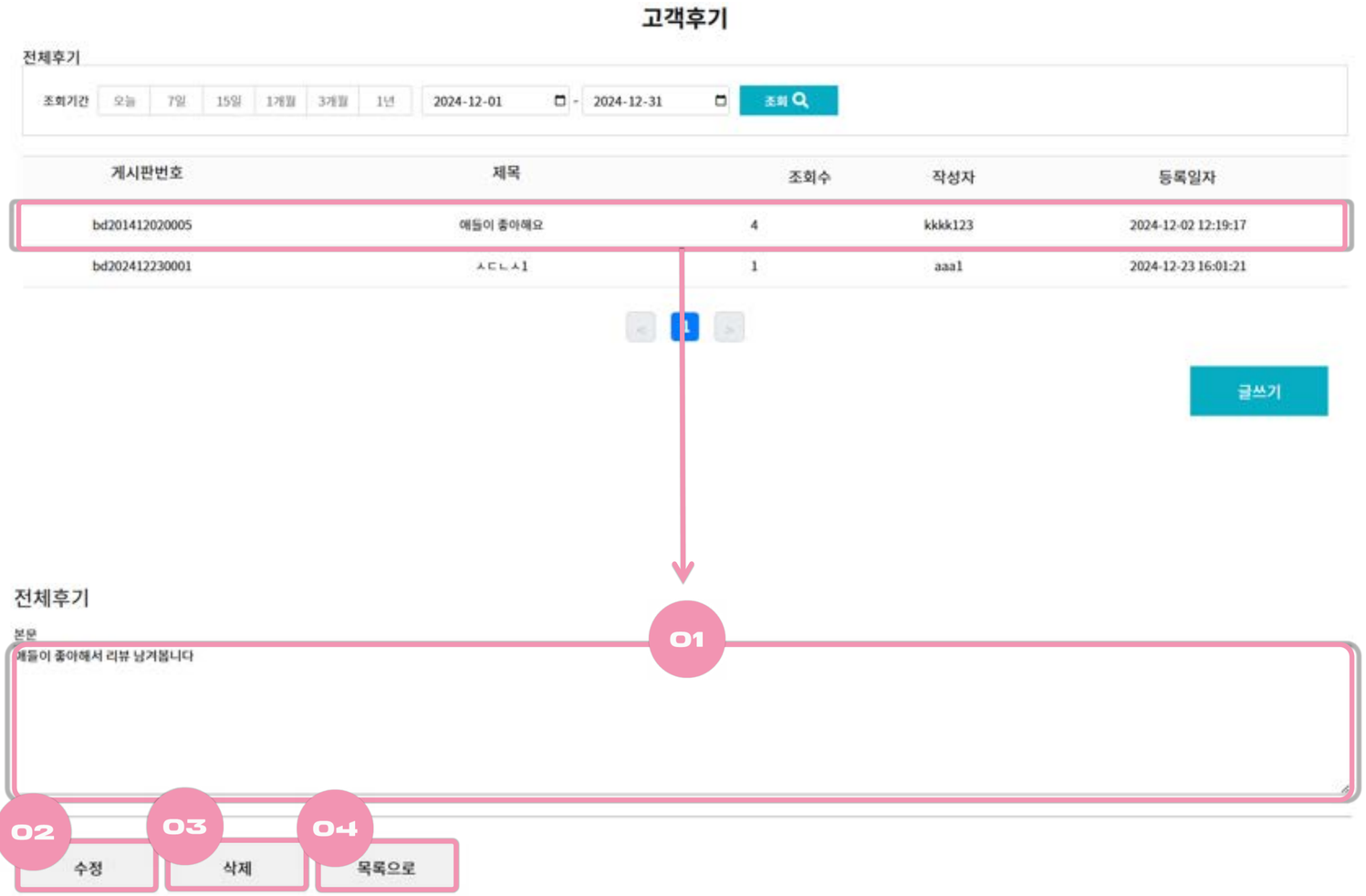
02 제목 : 제목 작성란

03 본문 : 내용 작성란

04 목록으로 : 리스트로 돌아가기

05 저장 : 작성한 내용 저장

24. 게시물 - 전체후기 - 상세보기



Description List

01 본문 : 작성된 내용

02 수정 : 게시글 내용 수정

03 삭제 : 게시글 삭제

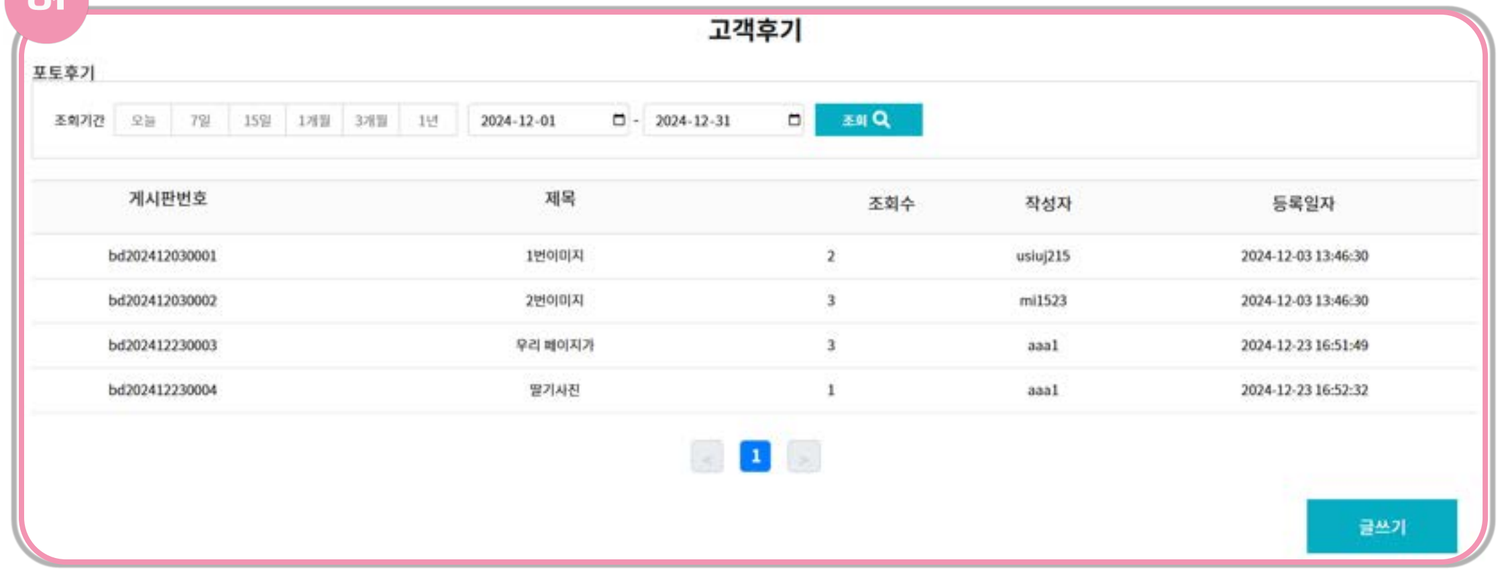
04 목록으로 : 리스트로 돌아가기

24. 게시물 - 포토후기

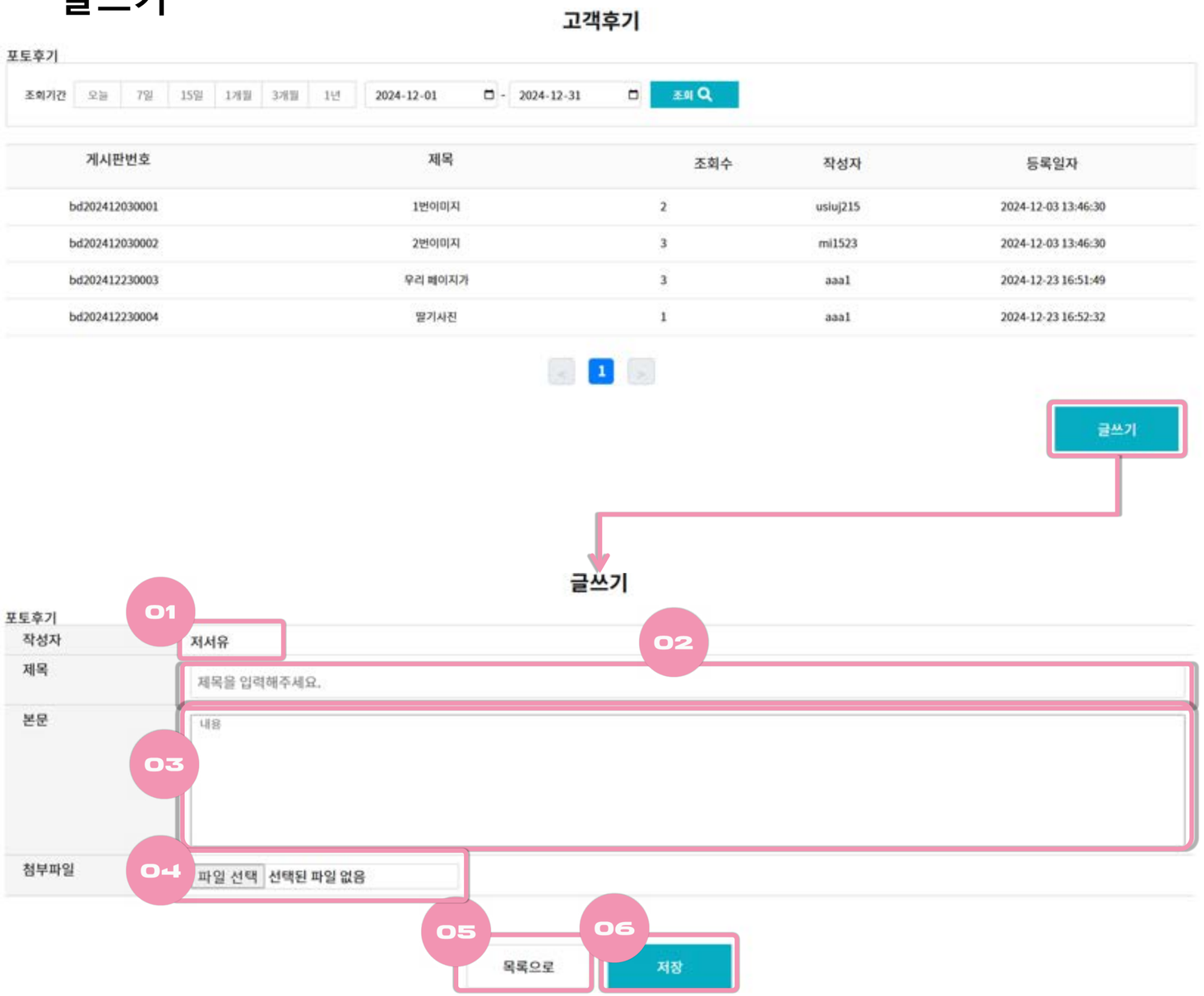
Description List

01 전체 후기와 동일

01



24. 게시물 - 포토후기 - 글쓰기



Description List

01 작성자 : 로그인시 저장된 세션값 표시

02 제목 : 제목 작성란

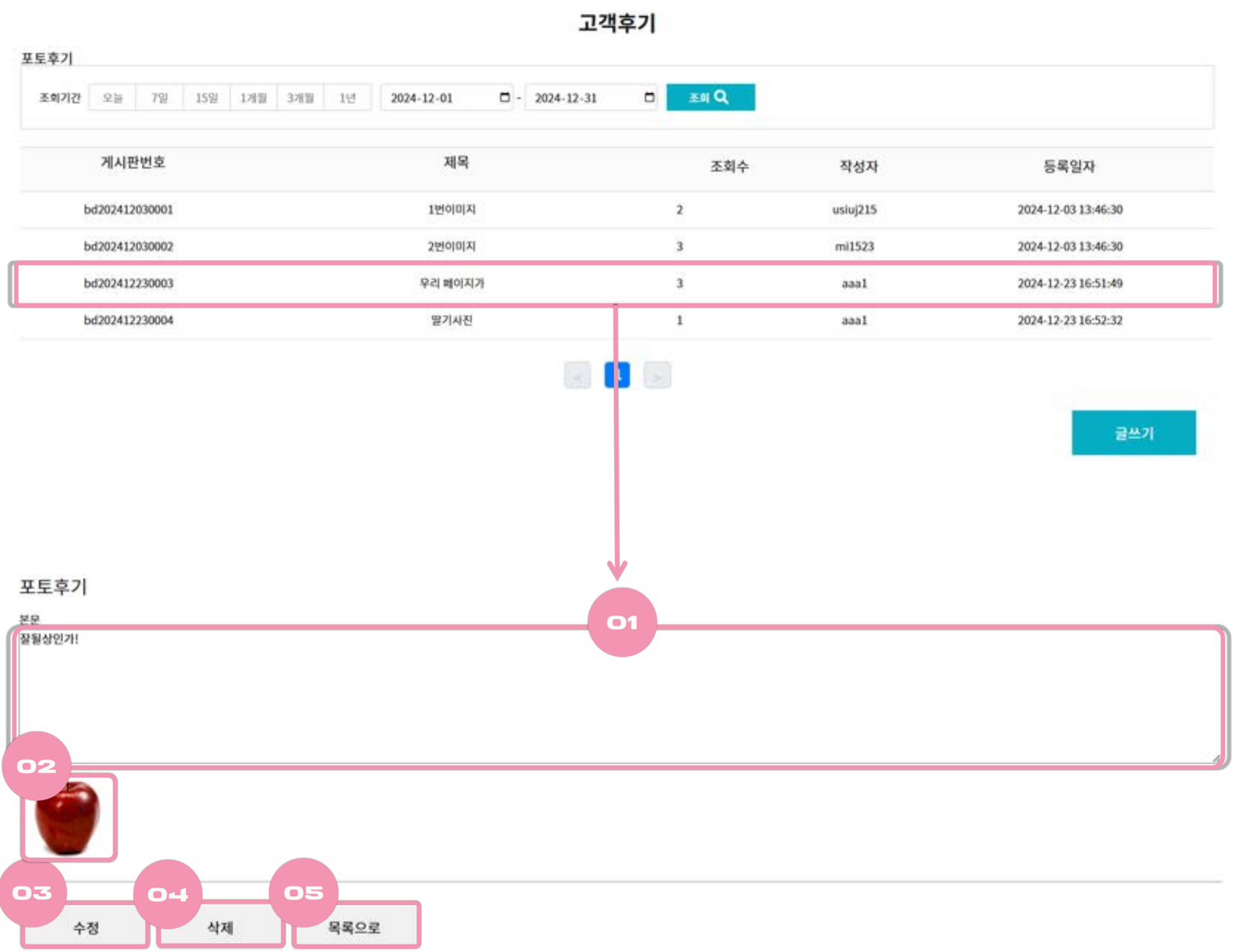
03 본문 : 내용 작성란

04 파일 선택 : 사진 첨부

05 목록으로 : 리스트로 돌아가기

06 저장 : 작성한 내용 저장

24. 게시물 - 포토후기 - 상세보기



Description List

01 본문 : 작성된 내용

02 사진 : 등록한 사진 표기

03 수정 : 게시글 내용 수정

04 삭제 : 게시글 삭제

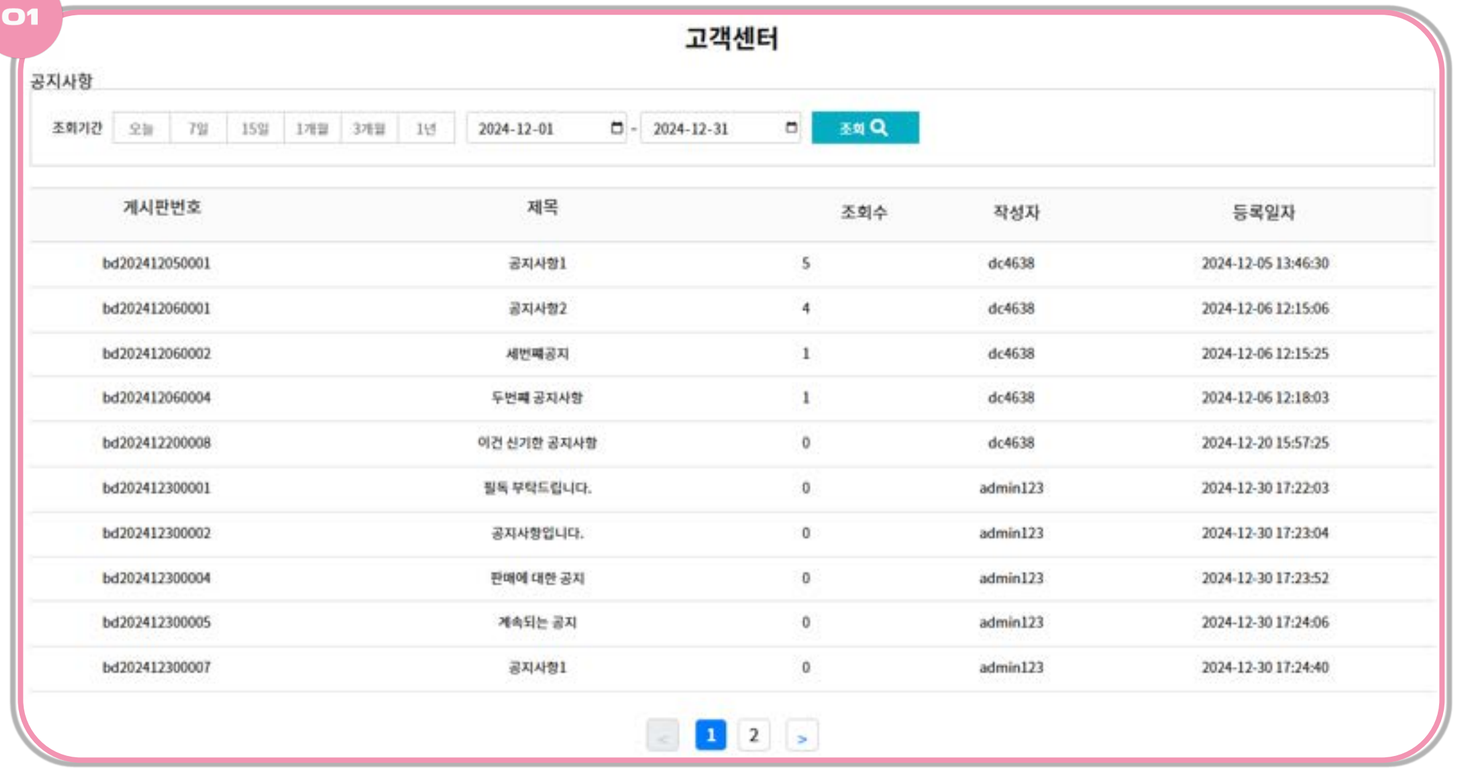
05 목록으로 : 리스트로 돌아가기

24. 게시물 - 공지사항

Description List

01 전체 후기와 동일

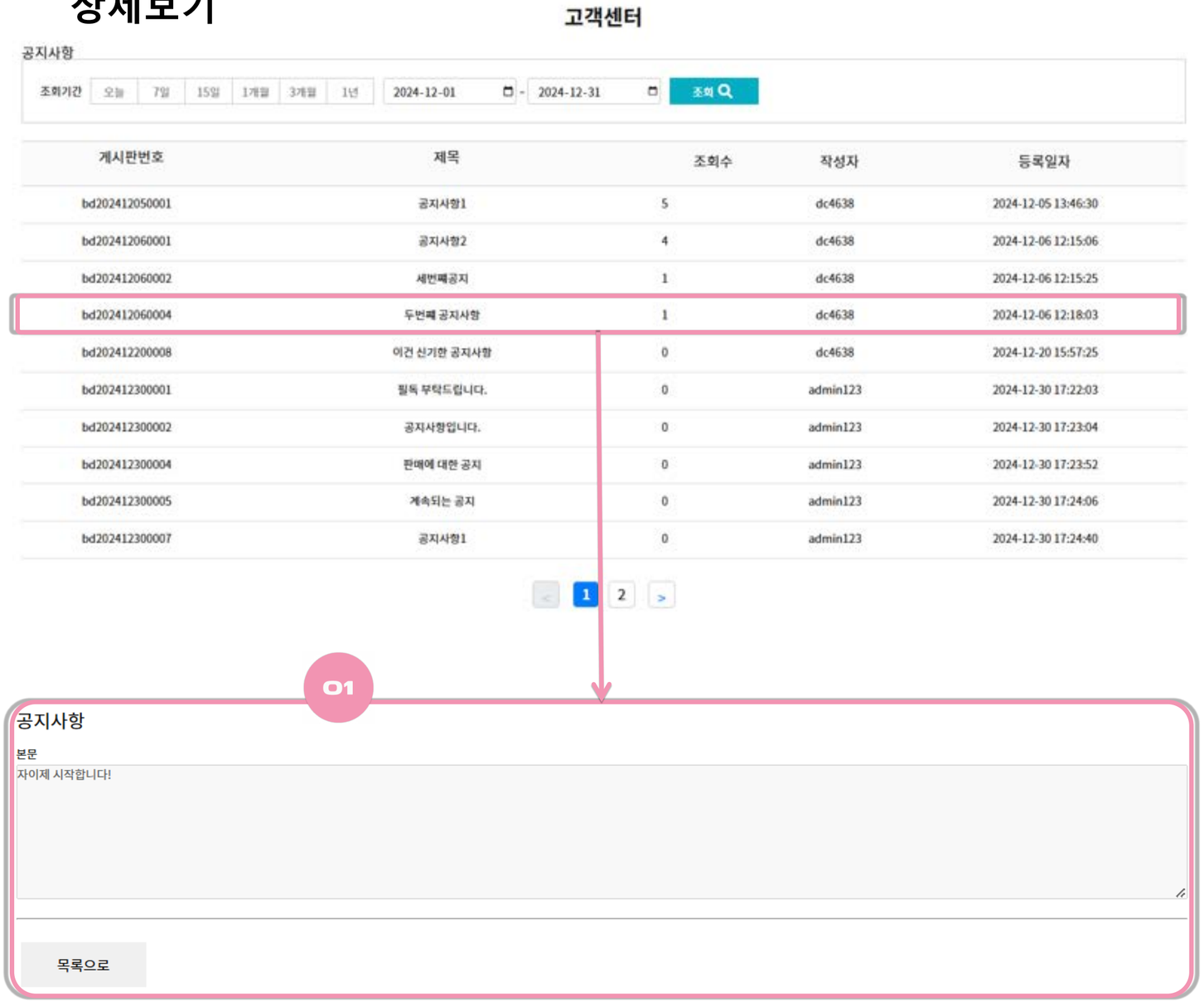
01



게시판번호	제목	조회수	작성자	등록일자
bd202412050001	공지사항1	5	dc4638	2024-12-05 13:46:30
bd202412060001	공지사항2	4	dc4638	2024-12-06 12:15:06
bd202412060002	세번째공지	1	dc4638	2024-12-06 12:15:25
bd202412060004	두번째 공지사항	1	dc4638	2024-12-06 12:18:03
bd202412200008	이건 신기한 공지사항	0	dc4638	2024-12-20 15:57:25
bd202412300001	필독 부탁드립니다.	0	admin123	2024-12-30 17:22:03
bd202412300002	공지사항입니다.	0	admin123	2024-12-30 17:23:04
bd202412300004	판매에 대한 공지	0	admin123	2024-12-30 17:23:52
bd202412300005	계속되는 공지	0	admin123	2024-12-30 17:24:06
bd202412300007	공지사항1	0	admin123	2024-12-30 17:24:40

< 1 2 >

24. 게시물 - 공지사항 - 상세보기



Description List

01 전체 후기와 동일

24. 게시물 - 상품문의

Description List

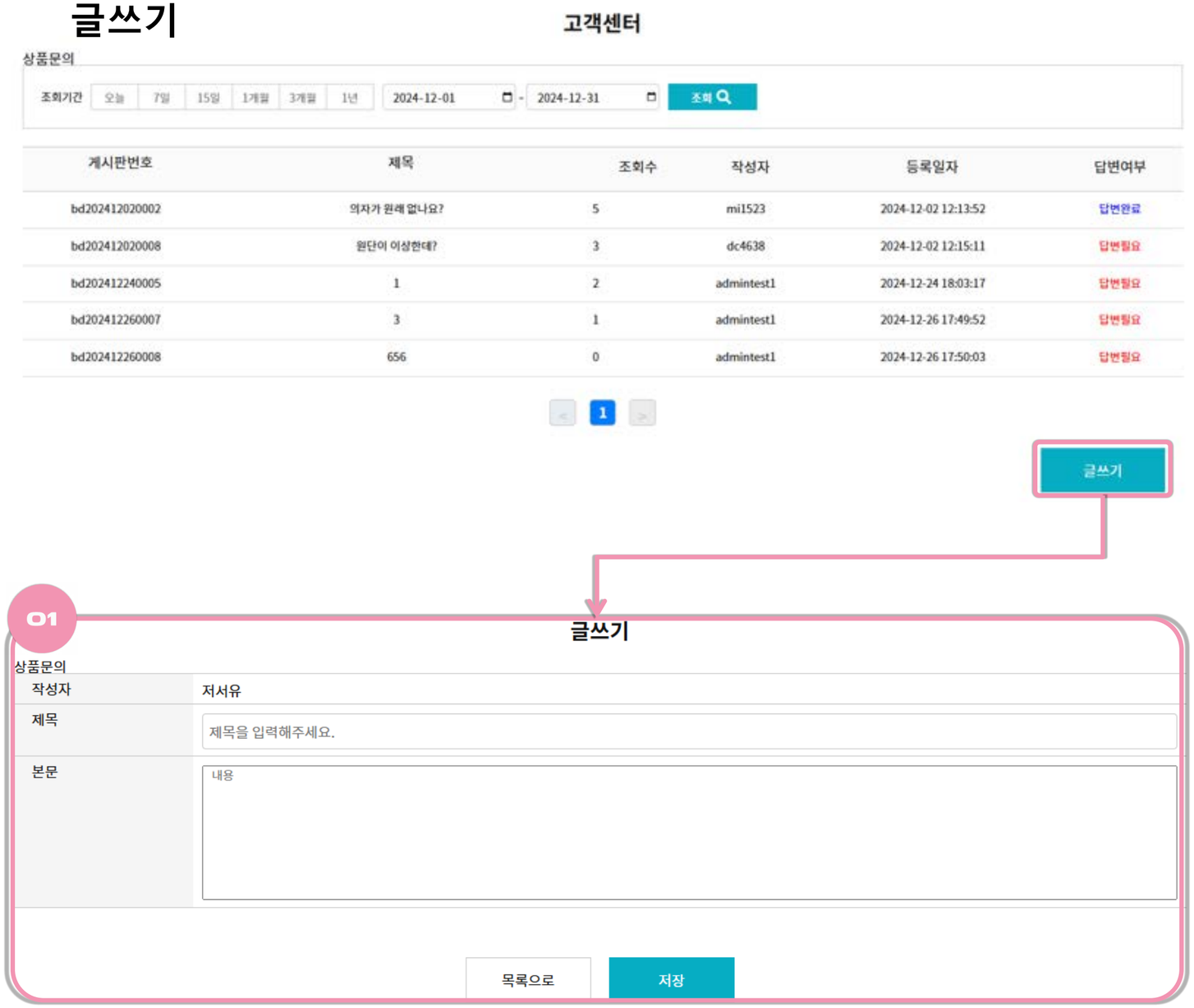
01 검색 필터
-조회기간

02 검색된 전체후기 리스트
- 게시판번호
- 제목
- 조회수
- 작성자
- 등록일자
- 답변여부

03 페이징

04 글쓰기

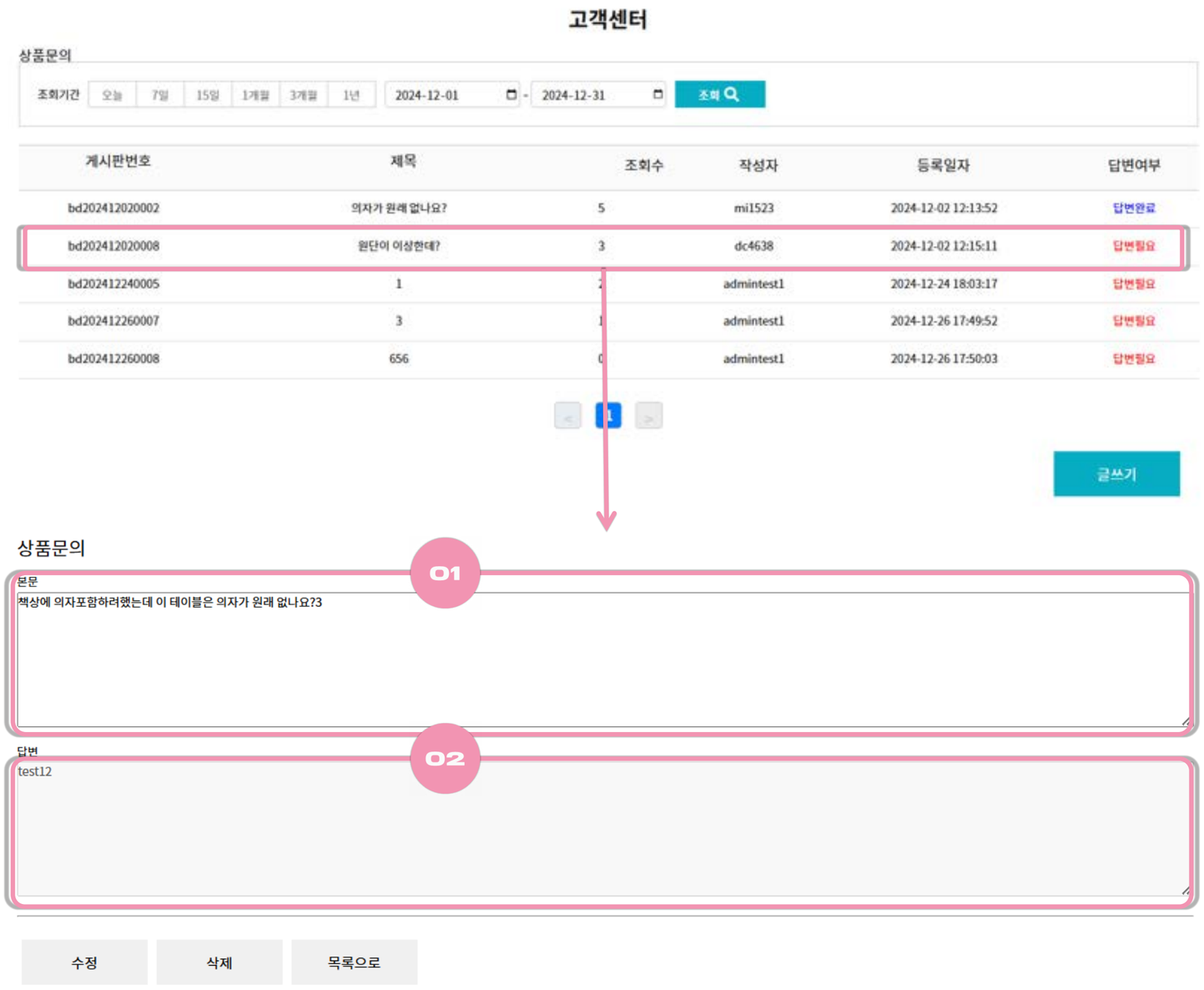
２４. 게시물 - 상품문의 - 글쓰기



Description List

01 전체 후기와 동일

24. 게시물 - 상품문의 - 상세보기



Description List

01 본문 내용

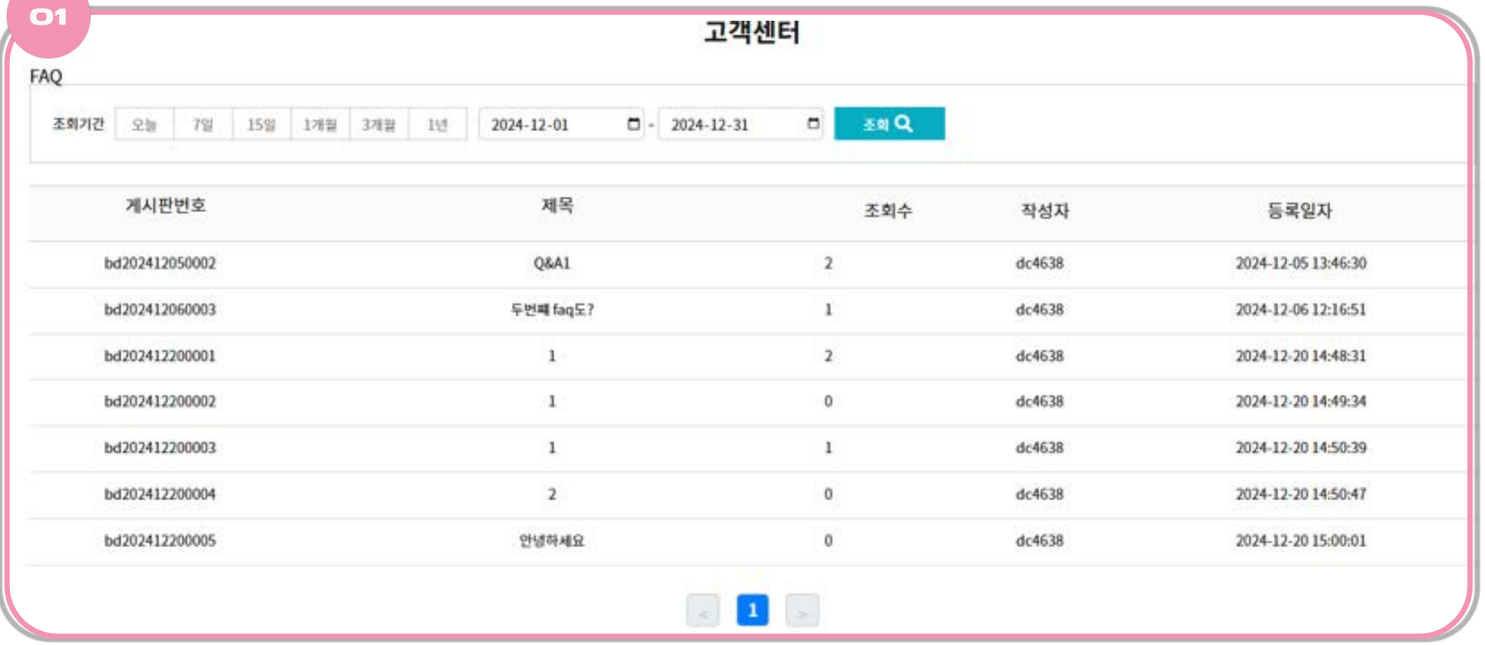
02 답변 내용

24. 게시물 - FAQ

Description List

01 전체 후기와 동일

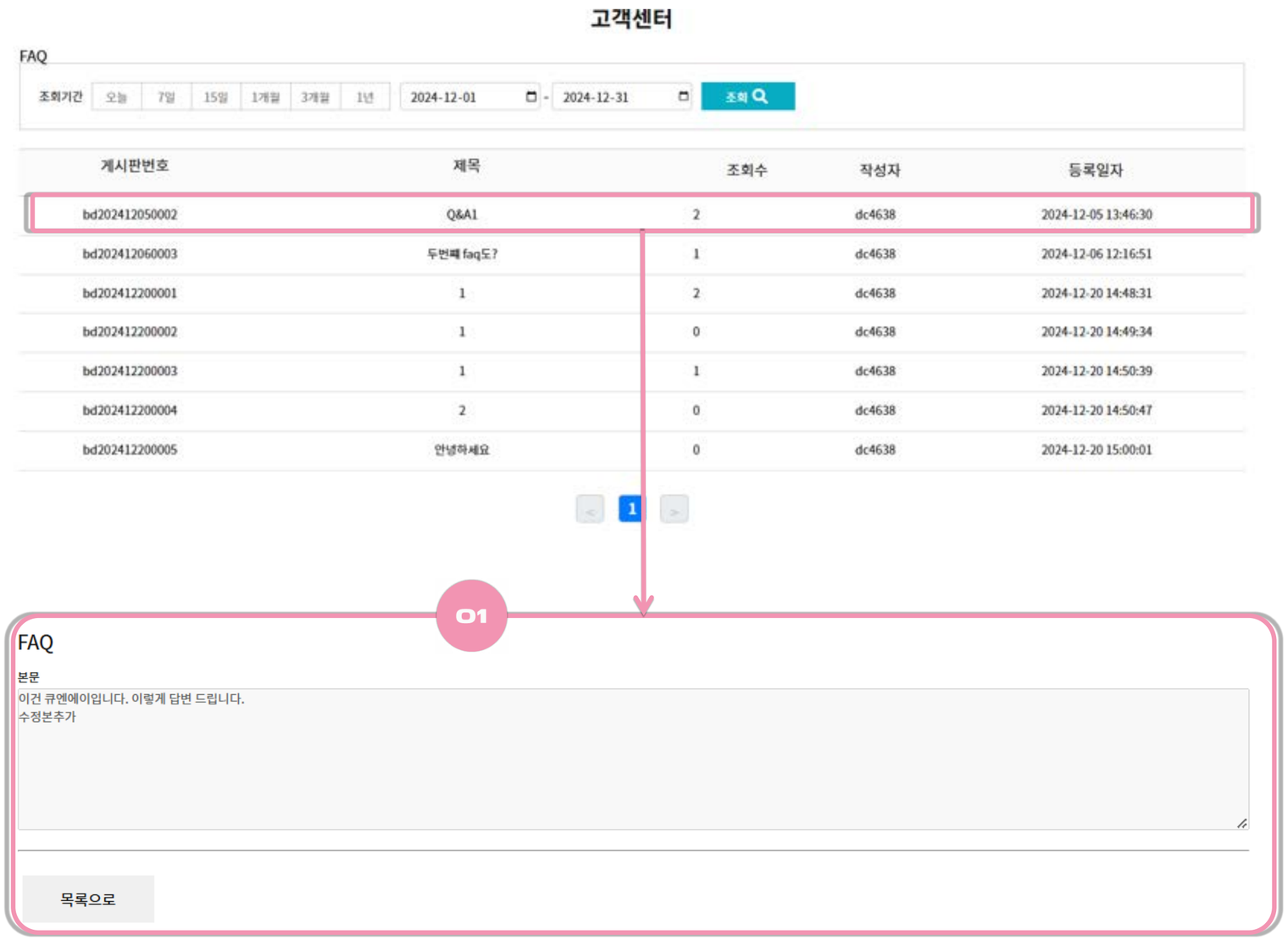
01



게시판번호	제목	조회수	작성자	등록일자
bd202412050002	Q&A1	2	dc4638	2024-12-05 13:46:30
bd202412060003	두번째 faq도?	1	dc4638	2024-12-06 12:16:51
bd202412200001	1	2	dc4638	2024-12-20 14:48:31
bd202412200002	1	0	dc4638	2024-12-20 14:49:34
bd202412200003	1	1	dc4638	2024-12-20 14:50:39
bd202412200004	2	0	dc4638	2024-12-20 14:50:47
bd202412200005	안녕하세요	0	dc4638	2024-12-20 15:00:01

< 1 >

２４. 게시물 - FAQ - 상세보기



Description List

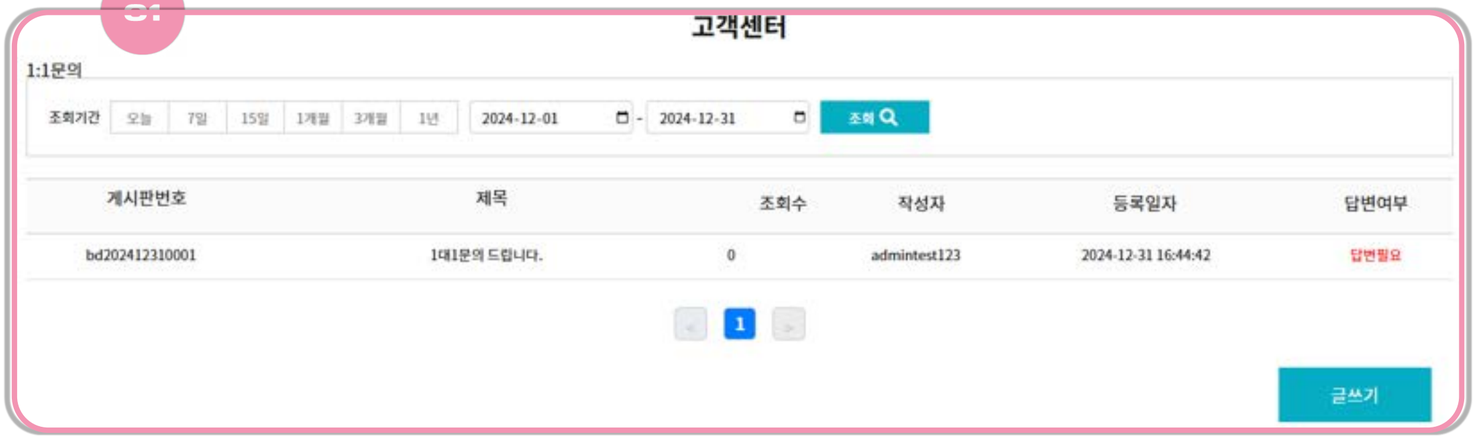
01 공지사항과 동일

2 4 . 게시물 - 1:1문의

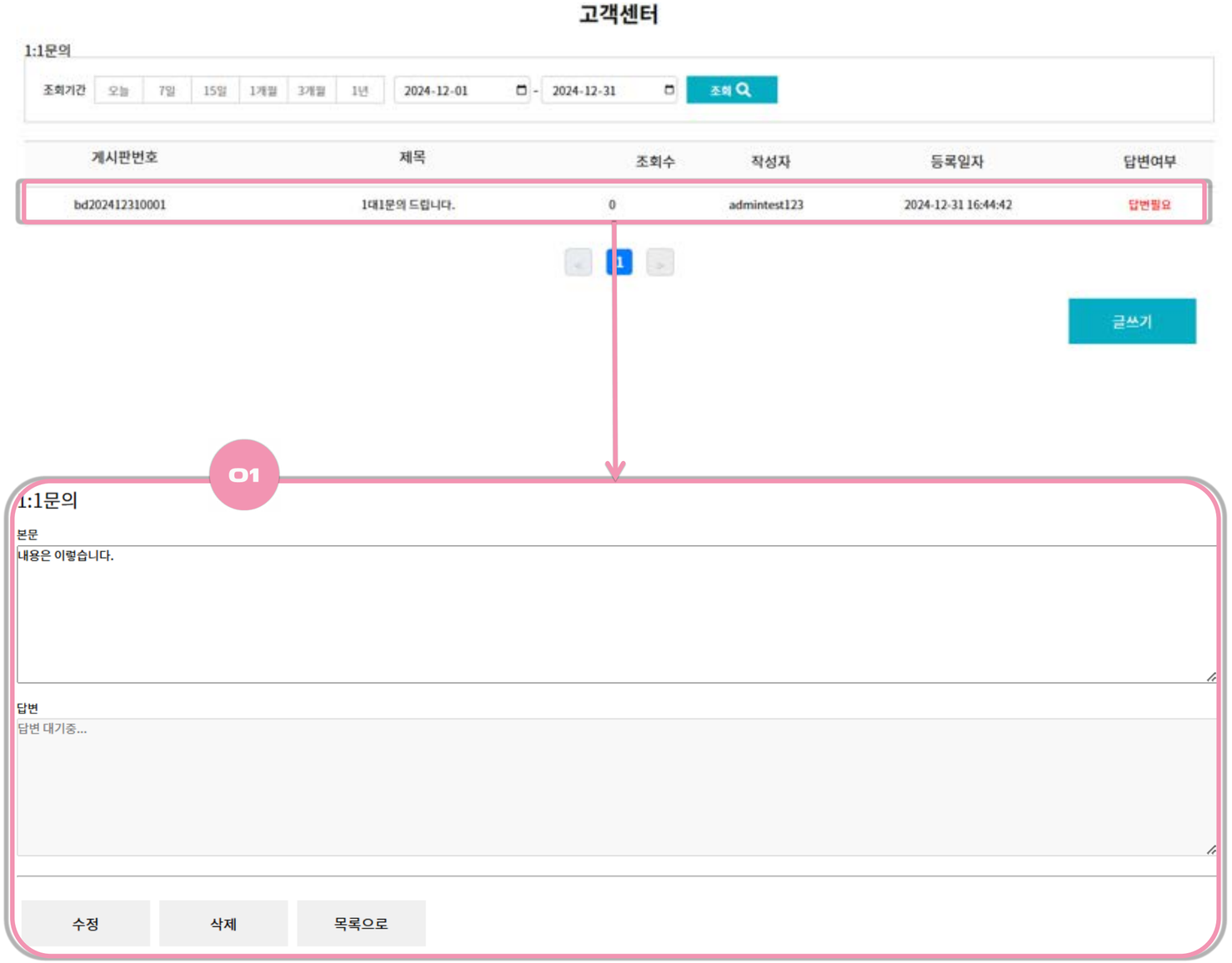
Description List

01 상품 문의와 동일

01



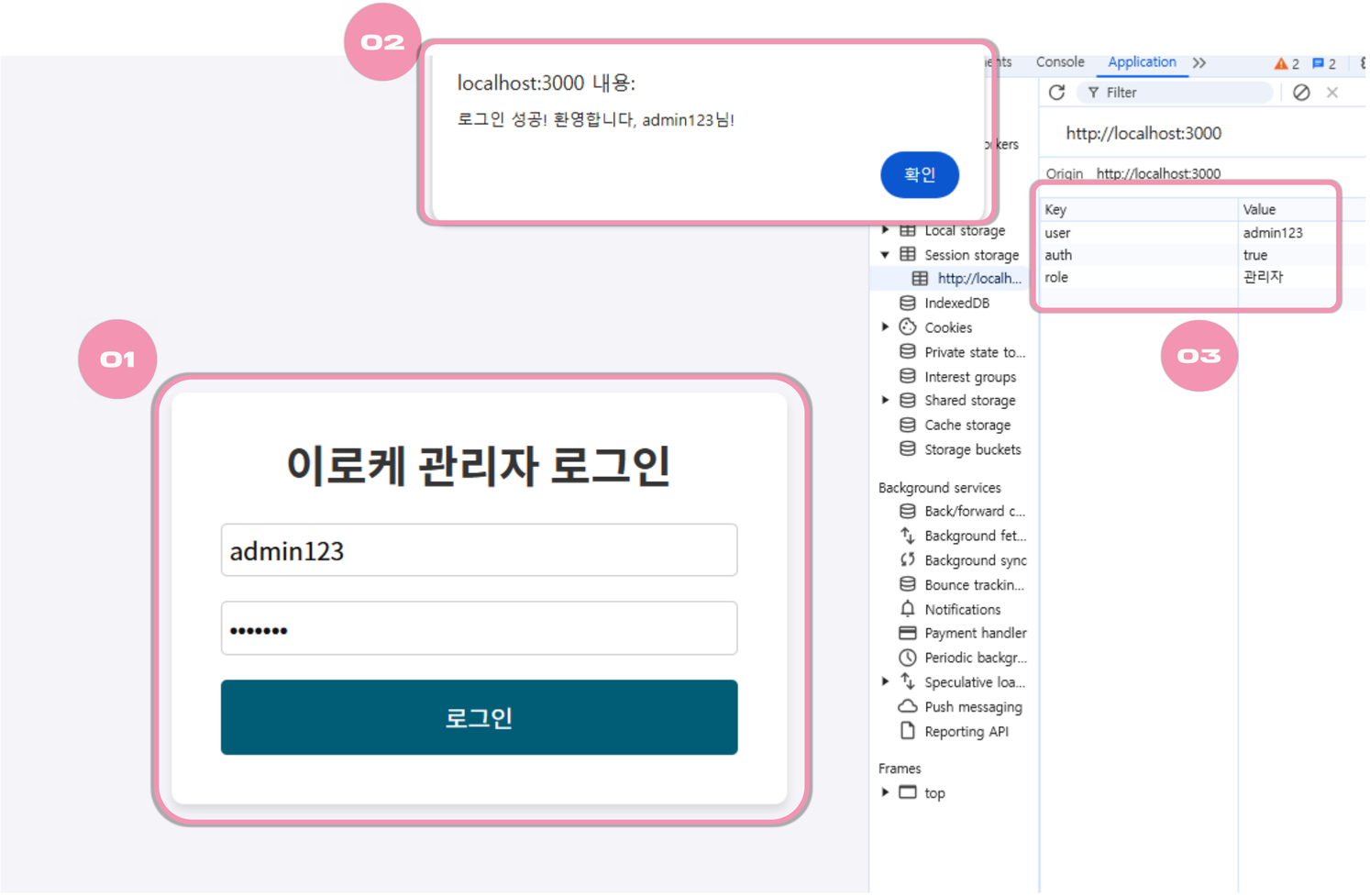
24. 게시물 - 1:1문의 - 상세보기



Description List

01 상품 문의와 동일

25. 관리자 페이지 로그인



Description List

01 관리자 아이디 및 비밀번호 입력

02 로그인 성공 시 alert 메시지 출력

03 로그인과 동시에 세션 스토리지에 정보 저장

26. 관리자 상품 등록

Description List

01 옵션/키워드 설정
- 옵션/키워드 선택
- 하위 옵션/키워드값 체크

02 가격 계산
- 상품 가격
- 할인율
- 최종 가격

26. 관리자 상품 등록 - 옵션/키워드 설정 01

01

```
const [option01, setOption01] = useState([]); //옵션 목록(배열)
const [option02, setOption02] = useState(""); //상위 옵션 선택
const [keyword01, setKeyword01] = useState([]); //키워드 목록(배열)
const [keyword02, setKeyword02] = useState(""); //상위 키워드 선택
const [opVal, setOpVal] = useState([]); // 하위 옵션 값
const [keyVal, setKeyVal] = useState([]); // 하위 키워드 값
const [selectedOption, setselectedOption] = useState([]);
const [selectedKeyword, setselectedKeyword] = useState([]);
```

02

```
//옵션 목록 불러오기
  useEffect(() => {
    selectOption(myData)
      .then((res) => {
        setOption01(res.data);
      })
      .catch((err) => {
        console.error("옵션 리스트 백엔드 연결 실패", err);
      });
  }, [myData]);
```

```
const optionClick = (e) => {
  setOption02(e.target.value); // 선택 옵션 저장
};
```

Description List

01 초기값 설정

02 등록 되어있는 옵션 목록 불러오기

- 셀렉트 된 옵션 저장(키워드는 함수명만 다르고 코드 동

26. 관리자 상품 등록 - 옵션/키워드 설정 02

03

```
//옵션값 불러오기
useEffect(() => {
  if (option02) {
    // 상위 옵션이 선택된 경우에만 실행
    // 옵션 리스트에서 선택된 option02의 code 값을 찾아서 전달
    const selectedOption = option01.find(function (opt) {
      // option02와 일치하는 name 찾기
      return opt.name === option02;
    });

    if (selectedOption) {
      const { code } = selectedOption; // 선택된 옵션의 code 값 추출
      selectOptionVal(code)
        .then((res) => {
          setOpVal(res.data); // 옵션값 저장
        })
        .catch((err) => {
          console.error("옵션값 불러오기 실패", err);
        });
    }else{
      setselectedOption([]);
    }
  }
}, [option02, option01]);
```

Description List

03 선택된 옵션/ 키워드의 등록된 하위 값들 불러오기

- 선택된 체크박스의 값이 배열에 저장
- 새로운 상위 옵션/키워드 선택 시 저장됐던 배열 리셋 (키워드는 함수명만 다르고 코드 동일)

26. 관리자 상품 등록 - 가격 계산

01

```
const [orPrice, setOrPrice] = useState(0); //원가
const [disV, setDisV] = useState(0); //할인율
const [fnPrice, setFnPrice] = useState(0); //최종가
```

02

```
function fnPriceChange() {
    //소수점 제거
    let newFnPrice = Math.round(orPrice - orPrice * (disV / 100));
    //1원단위 남지 않게 10으로 나눠서 소수점 올림 후 다시 10 곱하기
    newFnPrice = Math.round(newFnPrice / 10) * 10;

    if (orPrice && disV > 0) {
      //원가 할인율 둘 다 있을 시
      setFnPrice(newFnPrice); //계산된 가격
      if (newFnPrice < 0 || disV > 100) {
        //최종가가 마이너스거나 할인율이 100퍼 넘을 때
        alert("할인율을 다시 입력하세요.");
        return setDisV(0);
      } else if (disV == 100) {
        //할인율 100퍼 맞는지 체크
        let checConfirm = confirm("할인율 100%를 입력하신게 맞나요?");
        if (!checConfirm) {
          return setDisV(0);
        }
      }
    } else if (!disV || disV == 0) {
      //할인율 없거나 0일 시
      let newOrPrice = Math.ceil(orPrice / 10) * 10;
      if (orPrice % 10 <= 4) {
        newOrPrice = Math.round(orPrice / 10) * 10;
      }
      setFnPrice(newOrPrice); //원가 가격
    }
  }
```

Description List

01 초기값 설정

02 인풋에 원가 입력 시

- 할인율 없을 시 1원 단위 남지 않게 반올림 자동 계산
- 할인율 있을 시 할인율로 가격 할인 계산 후 1원 반올림

27. 관리자 상품 목록

상품2 상세정보

상품명	상품2
사진	파일 선택 선택된 파일 없음
상세이미지	파일 선택 선택된 파일 없음
등록 재고 수량	300000 개
옵션	옵션1 색상
옵션값	대,중,소
키워드	키워드2 배송
키워드값	배송가능
사용 여부	ON OFF
혜택	
상품 가격	300000 원
할인율	0 %
최종 가격	300000 원

01 설정한 옵션/키워드 가져오기

27. 관리자 상품 목록

01

```
const [proList, setProlist] = useState({}); //초기값
// 로딩 화면(처음에 값 바로 안불러와서 설정정)
const [loading, setLoading] = useState(true);
// const [imgShow, setImgShow] = useState([]);//이미지 들고오기
const [dtList, setDtList] = useState([]); //상품디테일 들고오기
const [opList, setOpList] = useState([]); //옵션목록 들고오기
const [keyList, setKeyList] = useState([]); //키워드목록 들고오기
const [keyValList, setKeyvalList] = useState([]); //키워드값 들고오기
const [mainCode, setmainCode] = useState([]); //선택옵션
const [dtcode, setdtcode] = useState([]); //선택옵션값
const [dtBack, setDtBack] = useState({ mainCode: [], dtCode: [] }); // 상품 상세 정보
const [option01, setOption01] = useState([]); //옵션 목록(배열)
const [option02, setOption02] = useState(""); //상위 옵션 선택
const [opVal, setOpVal] = useState([]); // 하위 옵션 값
const [selectedOptions, setSelectedOptions] = useState([]); // 선택된 하위 옵션들
```

02

```
//상품 디테일 들고오기
    selectDetail(main_code)
      .then((res) => {
        setDtList(res.data || []);
      })
      .catch((err) => {
        console.error("상품 상세 백엔드 연결 실패", err);
      });
```

01 초기값 설정

02 상품 – 옵션/키워드와 관련된 상품 디테일 테이블 정보 가져오기

27. 관리자 상품 목록

03

```
//옵션목록 들고오기
    selectOption(myData)
      .then((res) => {
        setOpList(res.data || []);
        // console.log(res.data)
      })
      .catch((err) => {
        console.error("옵션목록 백엔드 연결 실패", err);
      }, []);

    //옵션값 들고오기
    selectOptionVal(code)
      .then((res) => {
        // console.log('옵션값 API 응답:', res);  // 응답 확인
        if (res.data) {
          setOpVal(res.data); // 정상적으로 데이터가 오면 상태 업데이트
          // setOpvalList(res.data);  // 정상적으로 데이터가 오면 상태 업데이트
          // console.log(res.data)
        } else {
          console.error("옵션값 데이터가 비어 있습니다.");
        }
      })
      .catch((err) => {
        console.error("옵션값 API 호출 실패:", err); // 에러 로그
      });
```

03

옵션목록/옵션값 들고오기

(키워드도 함수명만 다르고 코드 동일)

28. 옵션 관리(1)

Description List

01 검색 필터
- 필수여부 : 전체, 필수O, 필수X
- 사용여부 : 전체, 사용, 미사용
- 옵션명

02 검색된 옵션 리스트
- 번호
- 옵션명
- 필수선택
- 사용여부

03 상위 리스트 행추가, 저장

04 페이징

28. 옵션 관리(2)

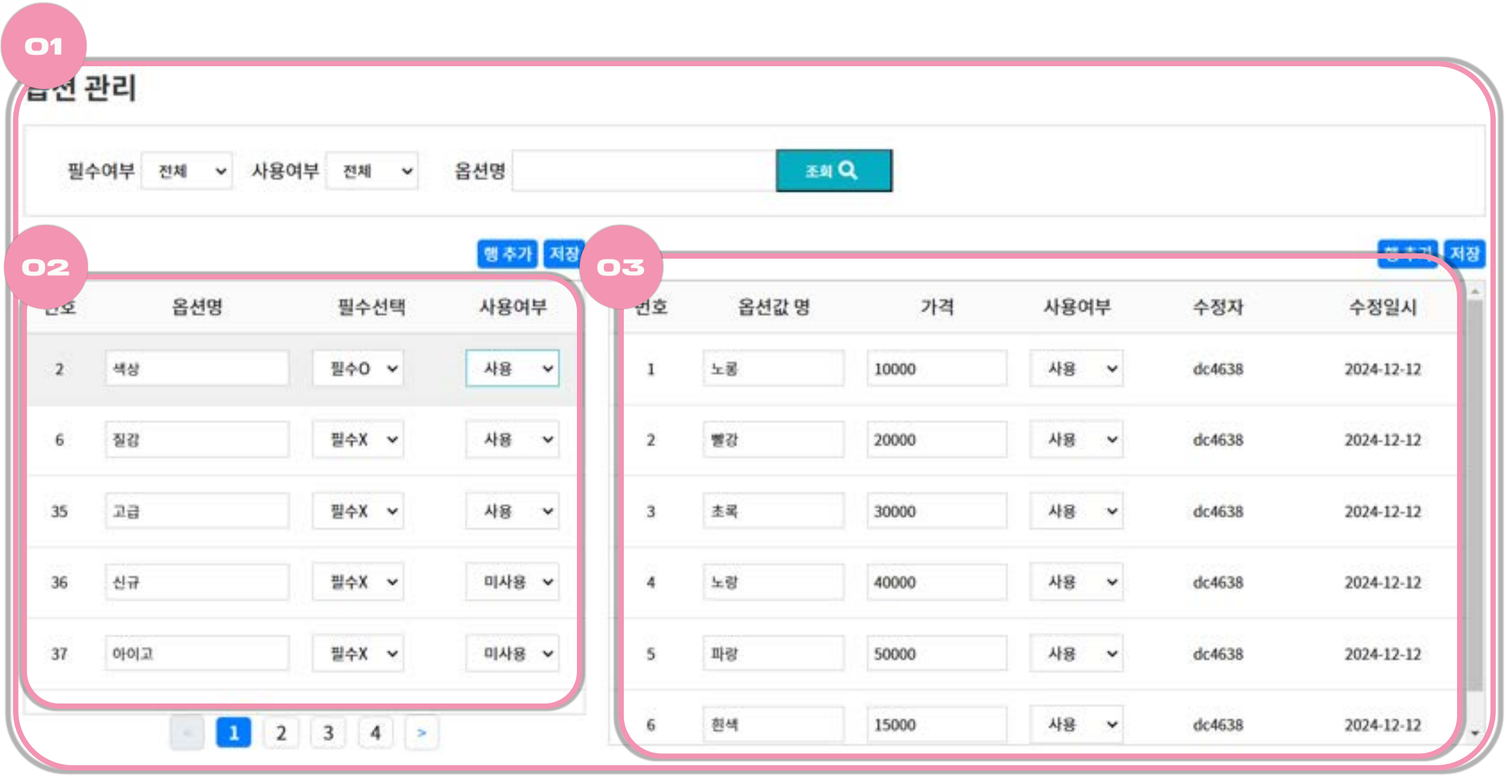
Description List

05 선택된 옵션 리스트 하위 값 (스크롤 처리)

- 번호
- 옵션값 명
- 가격
- 사용여부
- 수정자
- 수정일시

06 하위 리스트 행추가, 저장

28. 옵션 페이지 컴포넌트화(1)



Description List

01 전체 소스

02 옵션 관리 화면 컴포넌트화

03 옵션값 관리 화면 컴포넌트화

28. 옵션 페이지 컴포넌트화(2)

Description List

01 전체 소스

01

```
<div className="inquiry-section">
  <div className="filter-header">
    <h1>옵션 관리</h1>
  </div>
  <form name="myFrm">
    <div className="mypage-date-check">
      <div>
        <span className="mypage-period-text">필수여부</span>
        <select
          name="reqe"
          className="color-option"
          value={selectedReq}
          onChange={(e) => setSelectedReq(e.target.value)}
        >
          <option value="">전체</option>
          <option value="1">필수O</option>
          <option value="0">필수X</option>
        </select>
      </div>
      <div>
        <span className="mypage-period-text">사용여부</span>
        <select
          name="status"
          className="color-option"
          value={selectedStatus}
          onChange={(e) => setSelectedStatus(e.target.value)}
        >
          <option value="">전체</option>
          <option value="1">사용</option>
          <option value="0">미사용</option>
        </select>
      </div>
      <div>
        <span className="mypage-period-text">옵션명</span>
        <input type="text" name="name" />
      </div>
      <div className="mypage-date-search">
        <button className="mypage-date-search-btn" onClick={searchOptions}>
          조회
        </button>
      </div>
    </div>
  </form>

  <div className={styles.table_container}>
    <div className={styles.head_container}>
      <HeadContainer
        options={options}
        searchOptions={searchOptions}
        setOptionValues={setOptionValues}
        setOptions={setOptions}
        setSelectedHeadCode={setSelectedHeadCode}
      />
    </div>
    <div className={styles.detail_container}>
      <DetailContainer
        optionValues={optionValues}
        setOptionValues={setOptionValues}
        selectedHeadCode={selectedHeadCode}
      />
    </div>
  </div>
</div>
```

28. 옵션 페이지 컴포넌트화(3)

02

```
currentItems.map((item, index) => (
  <tr
    key={index}
    onClick={() => handleRowClick(item.code, index)}
    className={getRowClassName(index)}
  >
    <td>{item.code}</td>
    <td>
      <input
        type="text"
        value={item.name}
        onChange={(e) => handleReqChange(e, item, "name")}
        required
        placeholder="옵션 명"
      />
    </td>
    <td>
      <select
        value={item.req}
        onChange={(e) => handleReqChange(e, item, "req")}
      >
        <option value="0">필수X</option>
        <option value="1">필수O</option>
      </select>
    </td>
    <td>
      <select
        value={item.status}
        onChange={(e) => handleReqChange(e, item, "status")}
      >
        <option value="0">미사용</option>
        <option value="1">사용</option>
      </select>
    </td>
  </tr>
))
```

Description List

02 옵션 관리 화면 컴포넌트화

28. 옵션 페이지 컴포넌트화(4)

03

```
<tr key={index}>
                <td>{item.code}</td>
                <td>
                  <input
                    type="text"
                    value={item.name}
                    onChange={(e) => handleReqChange(e, item, "name")}
                    placeholder="옵션값 명"
                  />
                </td>
                <td>
                  <input
                    type="number"
                    value={item.opVal_price}
                    onChange={(e) => handleReqChange(e, item, "opVal_price")}
                    placeholder="가격"
                  />
                </td>
                <td>
                  <select
                    value={item.status}
                    onChange={(e) => handleReqChange(e, item, "status")}
                  >
                    <option value="0">미사용</option>
                    <option value="1">사용</option>
                  </select>
                </td>
                <td>{item.upt_id}</td>
                <td>{item.upt_date}</td>
              </tr>
```

Description List

03 옵션값 관리 화면 컴포넌트화

2 9 . 키워드 관리(1)

Description List

01 검색 필터
- 필수여부 : 전체, 필수O, 필수X
- 사용여부 : 전체, 사용, 미사용
- 키워드명

02 검색된 옵션 리스트
- 번호
- 키워드 명
- 필수선택
- 사용여부

03 상위 리스트 행추가, 저장

04 페이징

29. 키워드 관리(2)

Description List

05 선택된 옵션 리스트 하위 값 (스크롤 처리)

- 번호
- 키워드값 명
- 가격
- 사용여부
- 수정자
- 수정일시

06 하위 리스트 행추가, 저장

30. 주문 관리

Description List

01 검색 필터: 조회 기간
주문상태: 배송준비, 배송중, 배송완료
결제방법: 신용카드, 휴대폰
구매확정: 확정, 미확정

02 검색된 주문리스트
- 주문ID
- 회원ID
- 주문상태
- 결제방법
- 구매확정
- 총금액
- 날짜
- 상태변경

03 페이징처리

04 주문상태에 따른 변경버튼 생성
배송준비 상태일시 배송시작버튼 생성
배송중 상태일시 배송완료버튼 생성

30. 주문 관리

01

```
const [orders, setOrders] = useState([]);
const [totalOrders, setTotalOrders] = useState(0);
const [selectedStatus, setSelectedStatus] = useState("전체");
const [selectedPayMethod, setSelectedPayMethod] = useState("전체");
const [selectedConfirmation, setSelectedConfirmation] = useState("전체");
const [searchTerm, setSearchTerm] = useState("");
const [currentPage, setCurrentPage] = useState(1);
const [startDate, setStartDate] = useState("");
const [endDate, setEndDate] = useState("");
const [filters, setFilters] = useState(null); // 조회 버튼 눌렀을 때의 필터 저장
```

02

```
{order.status === "배송중" && (
  <button
    onClick={() =>
      handleStatusChange(order.order_id, "배송완료")
    }
  >
    배송완료
  </button>
)}
```

Description List

01 검색필터 초기값 설정

02 주문상태에 따른 변경버튼생성

30. 주문 관리

Description List

01 주문정보 출력

02 검색필터 설정

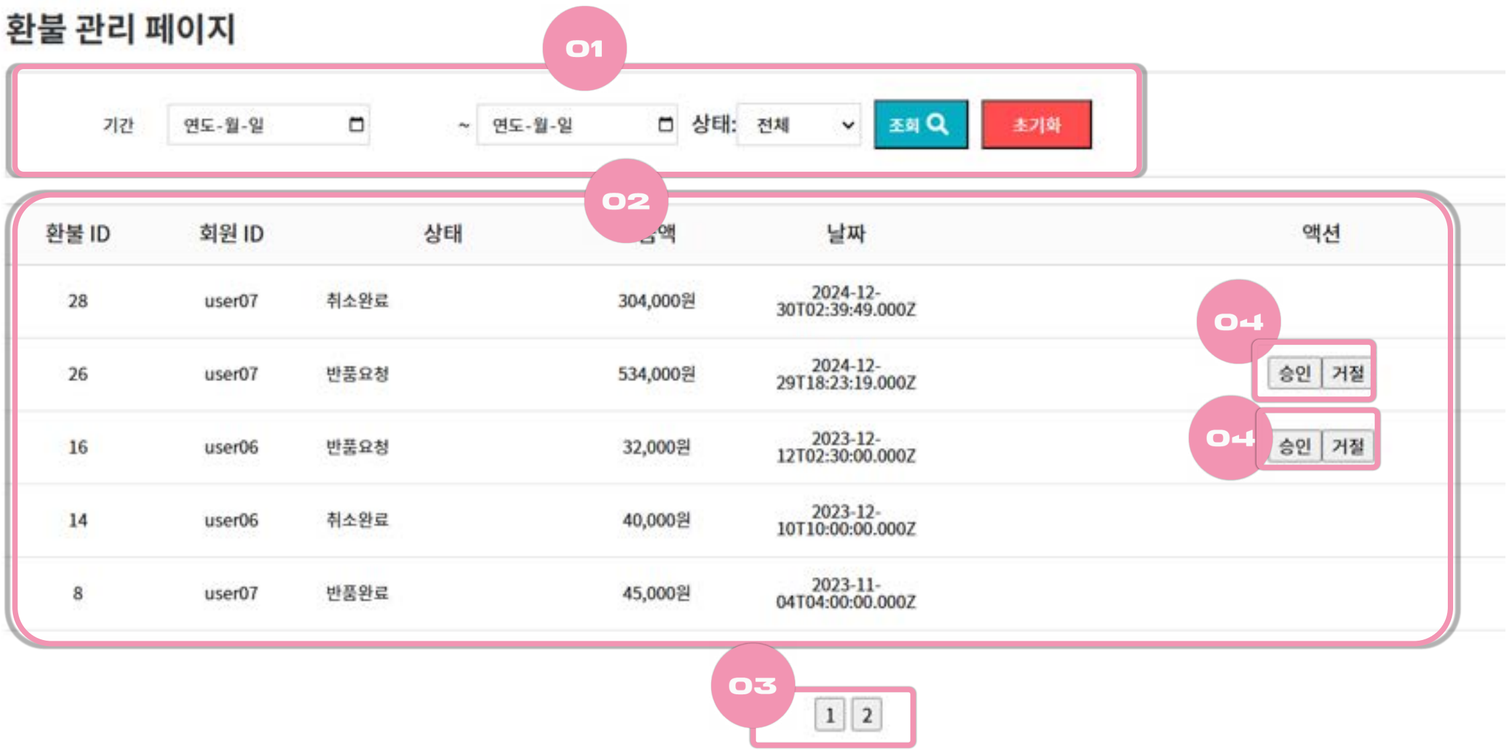
01

```
{orders.map((order) => (
  <tr key={order.order_id}>
    <td>{order.order_id || "N/A"}</td>
    <td>{order.member_id || "N/A"}</td>
    <td>{order.status || "N/A"}</td>
    <td>{order.payMethod || "N/A"}</td>
    <td>{order.isConfirmed || "N/A"}</td>
    <td>
      {order.total ? `${order.total.toLocaleString()}원` : "N/A"}
    </td>
    <td>{order.order_date || "N/A"}</td>
```

02

```
const statuses = ["전체", "배송준비", "배송중", "배송완료"];
const payMethods = ["전체", "휴대폰", "신용카드"];
const confirmations = ["전체", "구매확정", "구매미확정"];
```

31. 환불 관리



Description List

01 검색 필터: 조회 기간
주문상태:취소요청,취소완료,반품요청,빈품완료

02 검색된 주문리스트
- 환불ID(주문ID)
- 회원ID
- 주문상태
- 총금액
- 날짜
- 상태변경

03 페이징처리

04 주문상태에 따른 변경버튼 생성
승인 버튼 클릭시 해당 요청이 완료되며
저장된 데이터값이 변경됨

27. 환불 관리

01

02

Description List

01 거절버튼 클릭시 거절사유 모달창 뜨게됨

02 거절사유 입력후 확인버튼 클릭시
상태값이 거절됨으로 변경됨

31. 환불 관리

01

```
const [modalIsOpen, setModalIsOpen] = useState(false); // 모달 상태
const [selectedRefundId, setSelectedRefundId] = useState(null); // 선택한 환불 ID
const [rejectionReason, setRejectionReason] = useState(""); // 거절 사유
const [rejectedRefunds, setRejectedRefunds] = useState({}); // 거절된 환불 저장

const itemsPerPage = 5; // 한 페이지당 데이터 개수
const statuses = ["전체", "취소요청", "취소완료", "반품요청", "반품완료"];
const rejectionReasons = ["재고 부족", "상품 손상", "기타"];
```

02

```
const [selectedStatus, setSelectedStatus] = useState("전체");
const [currentPage, setCurrentPage] = useState(1); // 현재 페이지
const [startDate, setStartDate] = useState(""); // 시작일
const [endDate, setEndDate] = useState(""); // 종료일
const [filters, setFilters] = useState(null); // 필터 저장
```

Description List

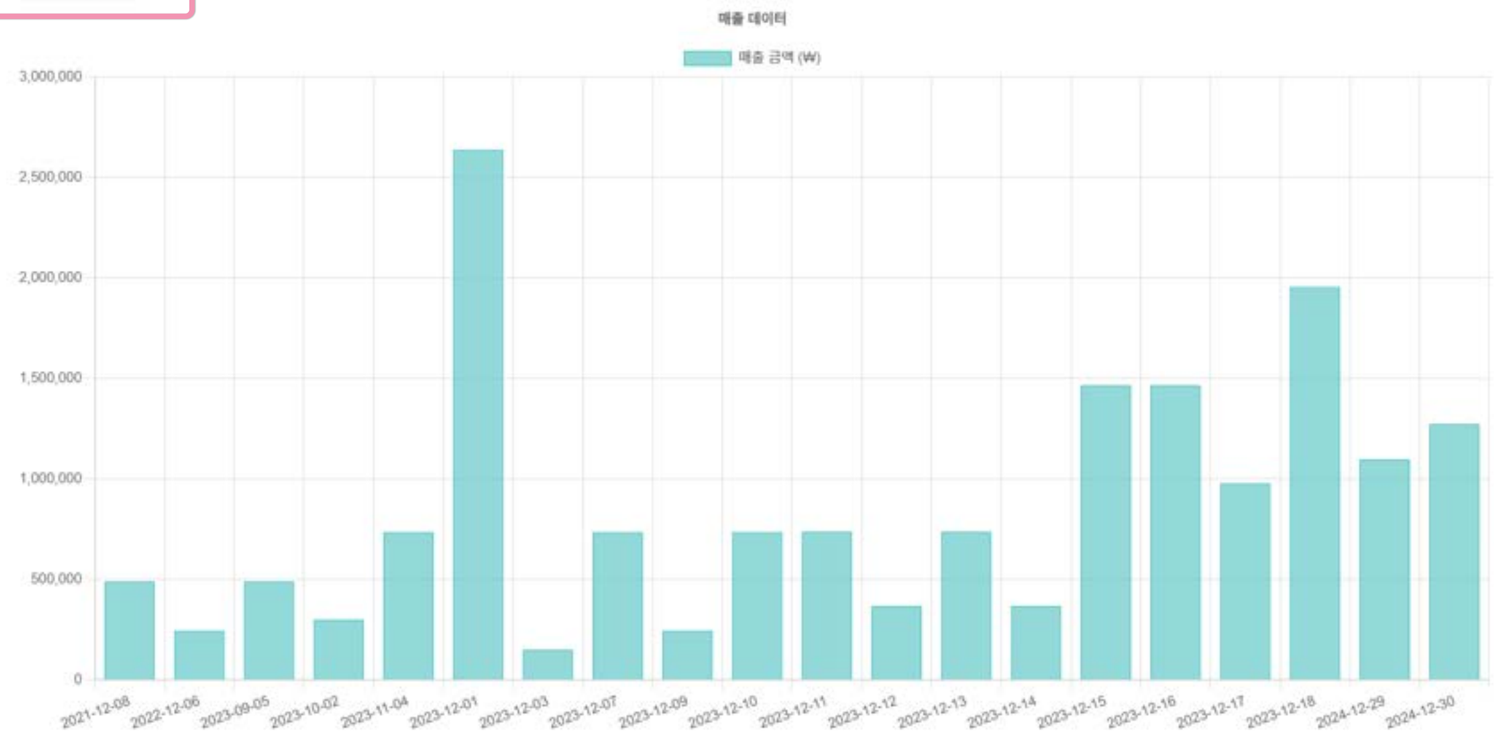
01 거절사유모달과 거절사유와 관련된 코드

02 필터에 대한 코드

32. 정산 관리

01

일별 매출



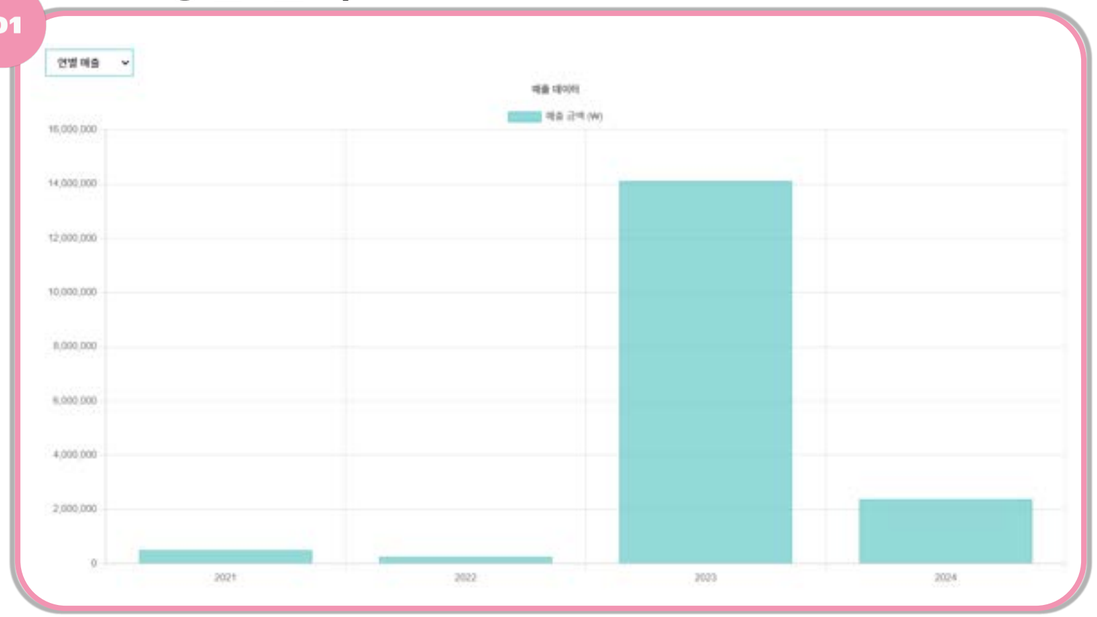
Description List

01 필터 : 일별,월별,연별,상품별

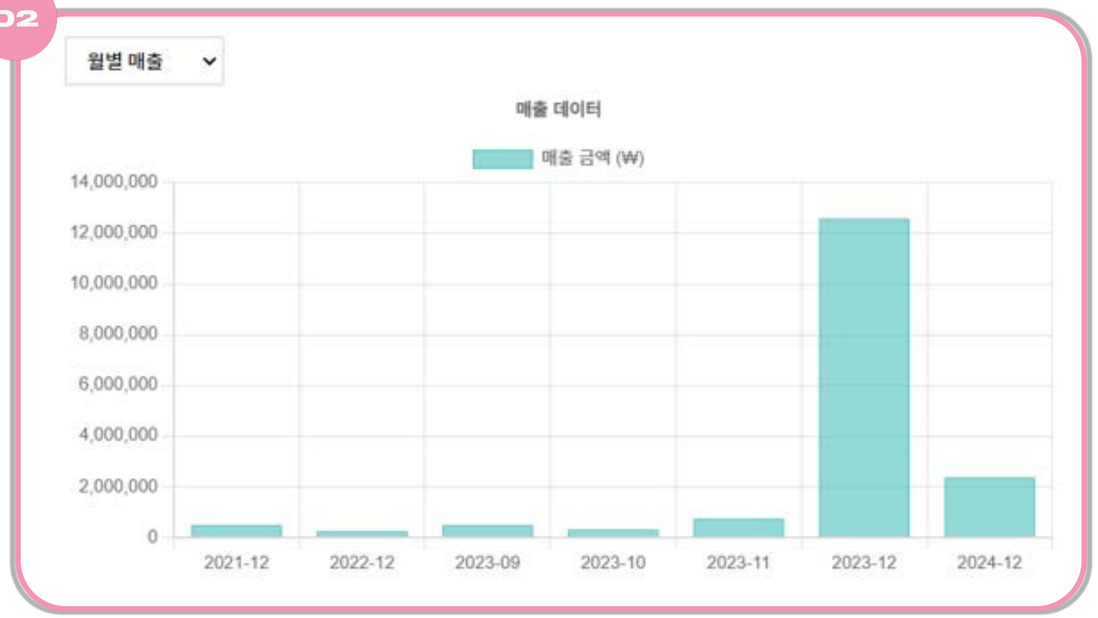
02 검색필터 설정

3 2. 정산 관리

01



02



Description List

01 월별 정산조회

02 연별 정산조회

32. 정산 관리

01

```
if (type === 'daily') {
    query = `
        SELECT
            DATE_FORMAT(o.order_date, '%Y-%m-%d') AS date,
            SUM(op.quantity * op.unit_price) AS sales
        FROM \`order\` o
        JOIN order_product op ON o.order_id = op.order_id
        GROUP BY DATE_FORMAT(o.order_date, '%Y-%m-%d')
        ORDER BY date ASC;
    `;
} else if (type === 'monthly') {
    query = `
        SELECT
            DATE_FORMAT(o.order_date, '%Y-%m') AS date,
            SUM(op.quantity * op.unit_price) AS sales
        FROM \`order\` o
        JOIN order_product op ON o.order_id = op.order_id
        GROUP BY DATE_FORMAT(o.order_date, '%Y-%m')
        ORDER BY date ASC;
    `;
```

02

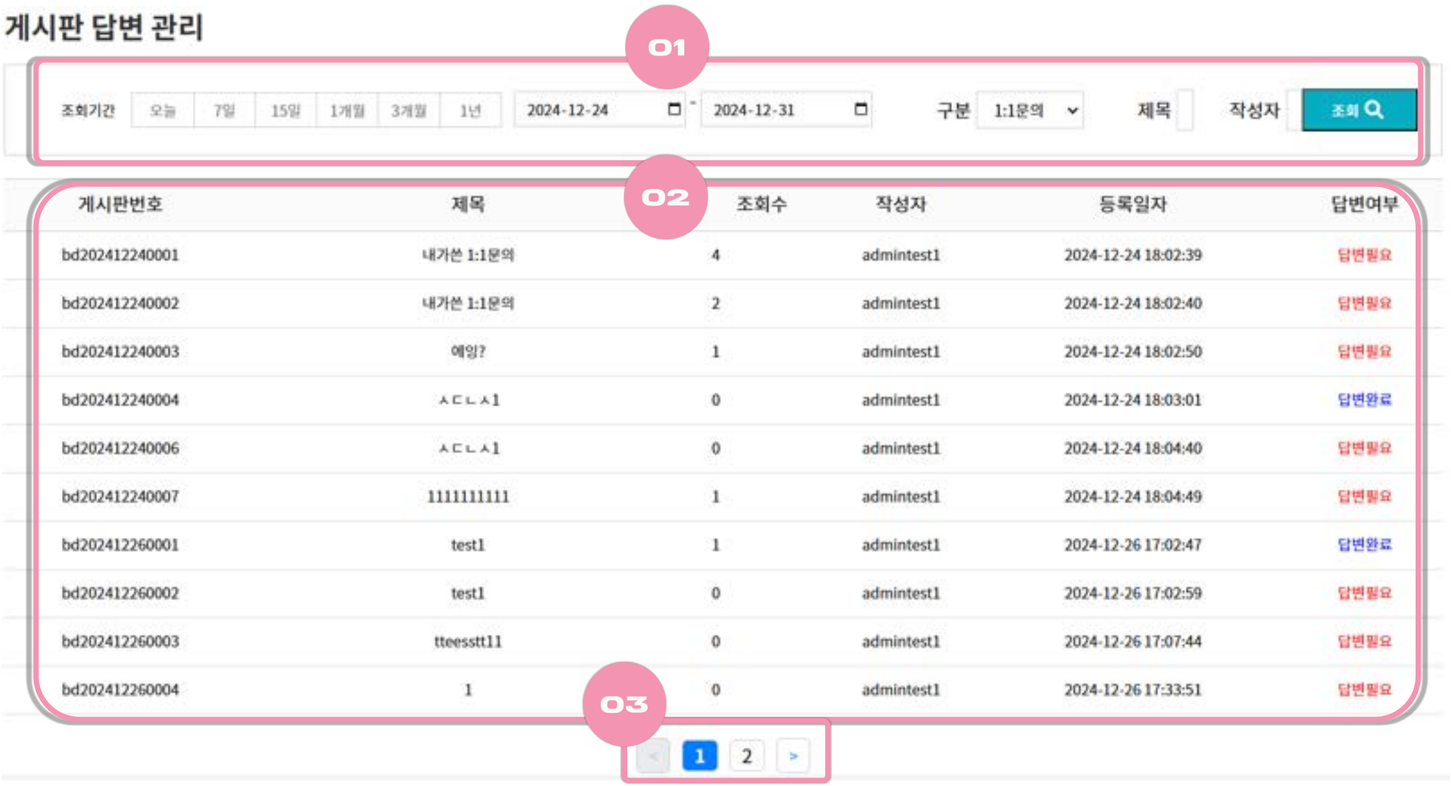
```
<div>
  <select value={type} onChange={(e) => setType(e.target.value)}>
    <option value="daily">일별 매출</option>
    <option value="monthly">월별 매출</option>
    <option value="yearly">연별 매출</option>
    <option value="product">상품별 매출</option>
  </select>
  <Bar data={chartData} options={chartOptions} />
</div>
```

Description List

01 기간별 필터에 대한 쿼리문

02 필터에 대한 코드

33. 게시판 답변 관리



게시판번호	제목	조회수	작성자	등록일자	답변여부
bd202412240001	내가쓴 1:1문의	4	admintest1	2024-12-24 18:02:39	답변필요
bd202412240002	내가쓴 1:1문의	2	admintest1	2024-12-24 18:02:40	답변필요
bd202412240003	예잉?	1	admintest1	2024-12-24 18:02:50	답변필요
bd202412240004	ㅅㄷㄴㅅ1	0	admintest1	2024-12-24 18:03:01	답변완료
bd202412240006	ㅅㄷㄴㅅ1	0	admintest1	2024-12-24 18:04:40	답변필요
bd202412240007	1111111111	1	admintest1	2024-12-24 18:04:49	답변필요
bd202412260001	test1	1	admintest1	2024-12-26 17:02:47	답변완료
bd202412260002	test1	0	admintest1	2024-12-26 17:02:59	답변필요
bd202412260003	tteesstt11	0	admintest1	2024-12-26 17:07:44	답변필요
bd202412260004	1	0	admintest1	2024-12-26 17:33:51	답변필요

Description List

01 검색 필터
- 조회 기간
- 구분 : 전체후기, 포토후기, 상품문의, 1:1문의

02 검색된 등록 게시판 리스트
- 게시판 번호
- 제목
- 조회수
- 작성자
- 등록일자
- 답변여부

03 페이징

34. 게시판 등록 관리

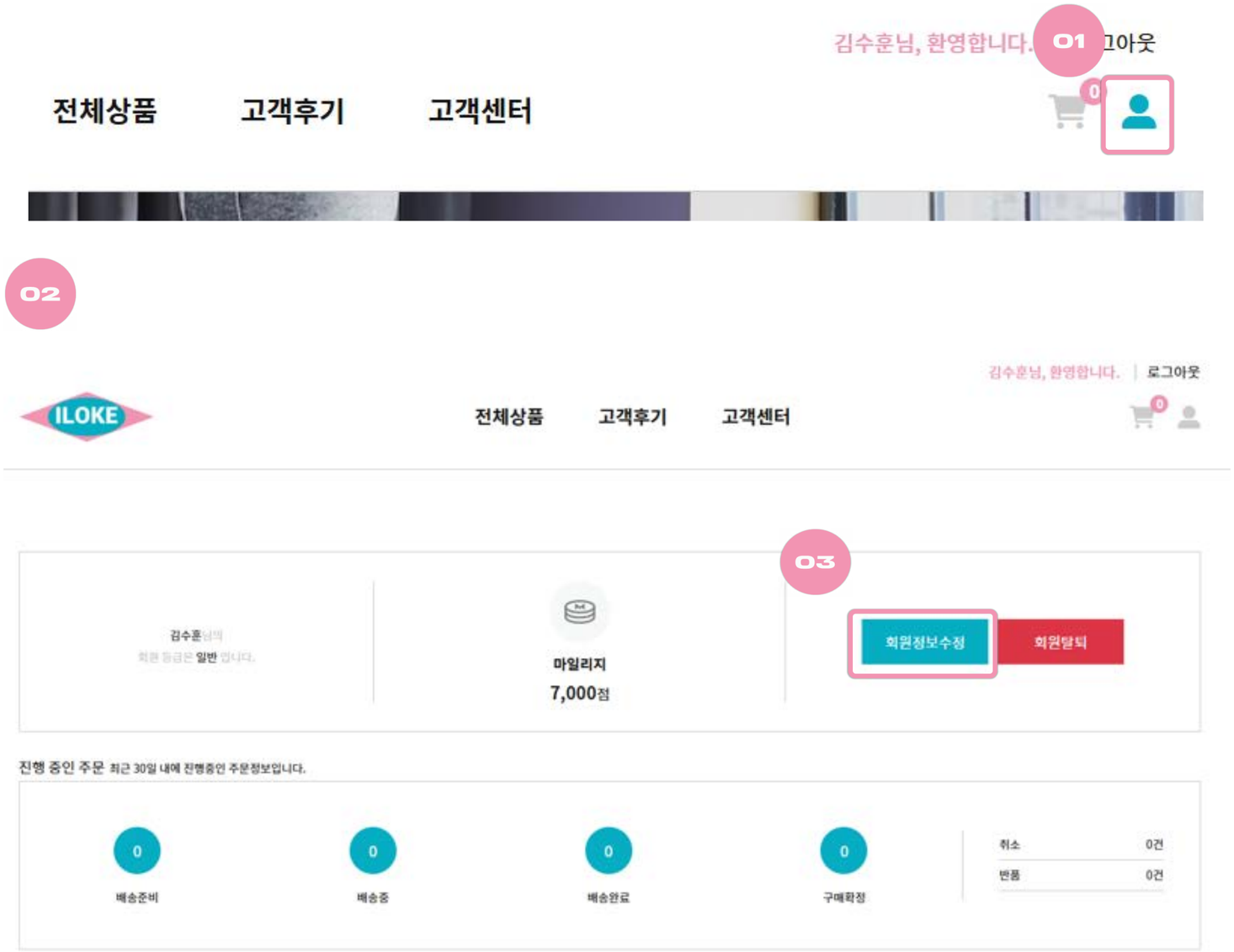
Description List

01 검색 필터
- 조회 기간
- 구분 : 공지사항, FAQ
- 제목
- 작성자

02 검색된 등록 게시판 리스트
- 게시판 번호
- 제목
- 조회수
- 작성자
- 등록일자
- 수정자
- 수정일자
- 삭제여부

03 신규 글 작성

35. 회원 정보 수정



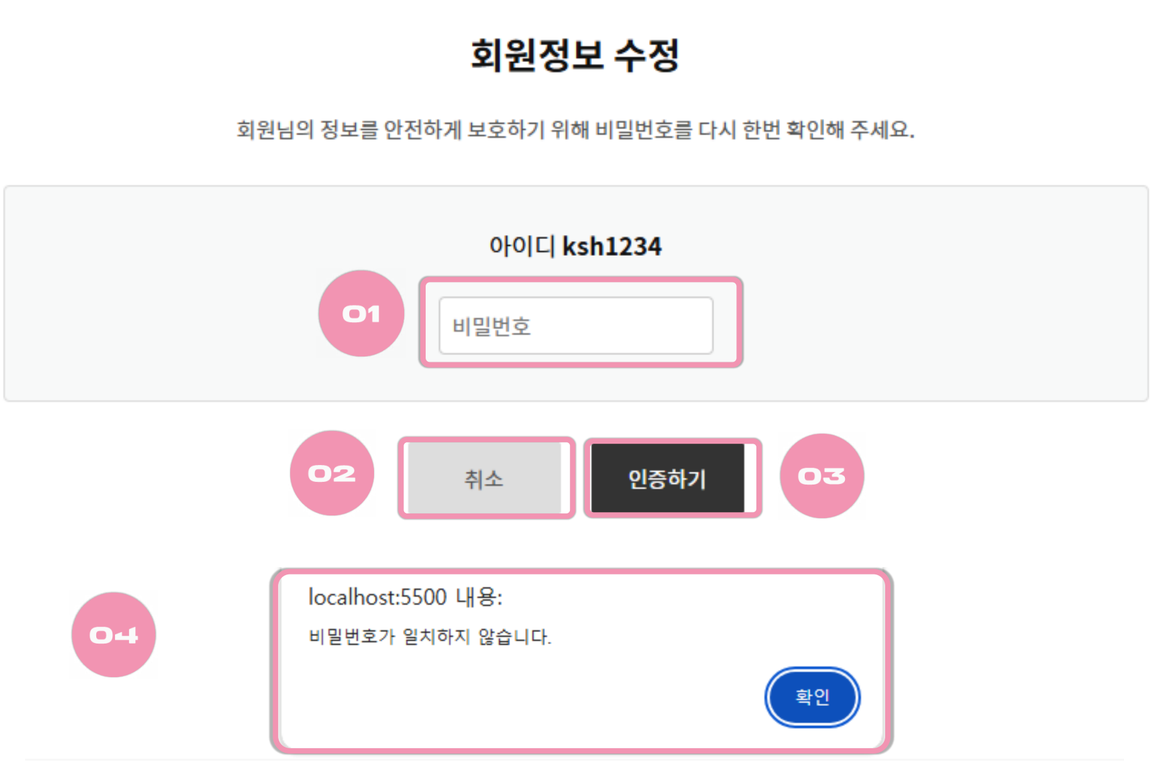
Description List

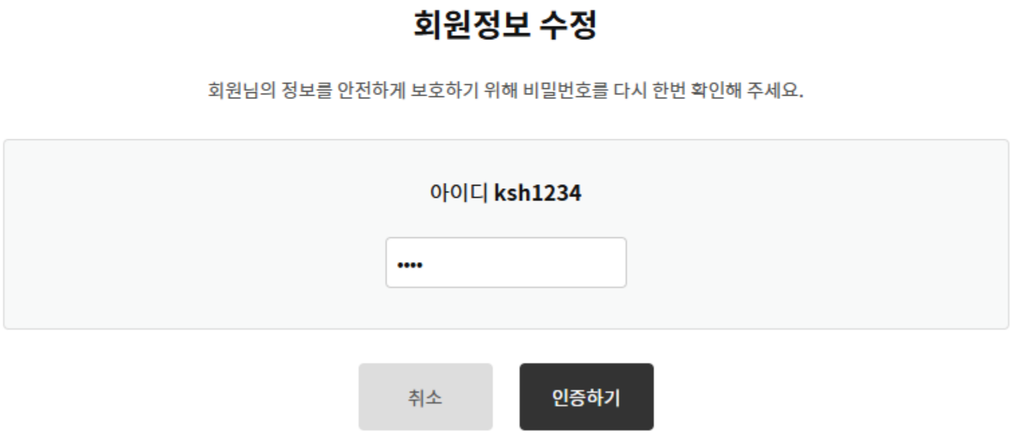
01 사람 아이콘 클릭 시 마이 페이지로 이동

02 마이 페이지

03 회원 정보 수정 버튼

35. 회원 정보 수정





Description List

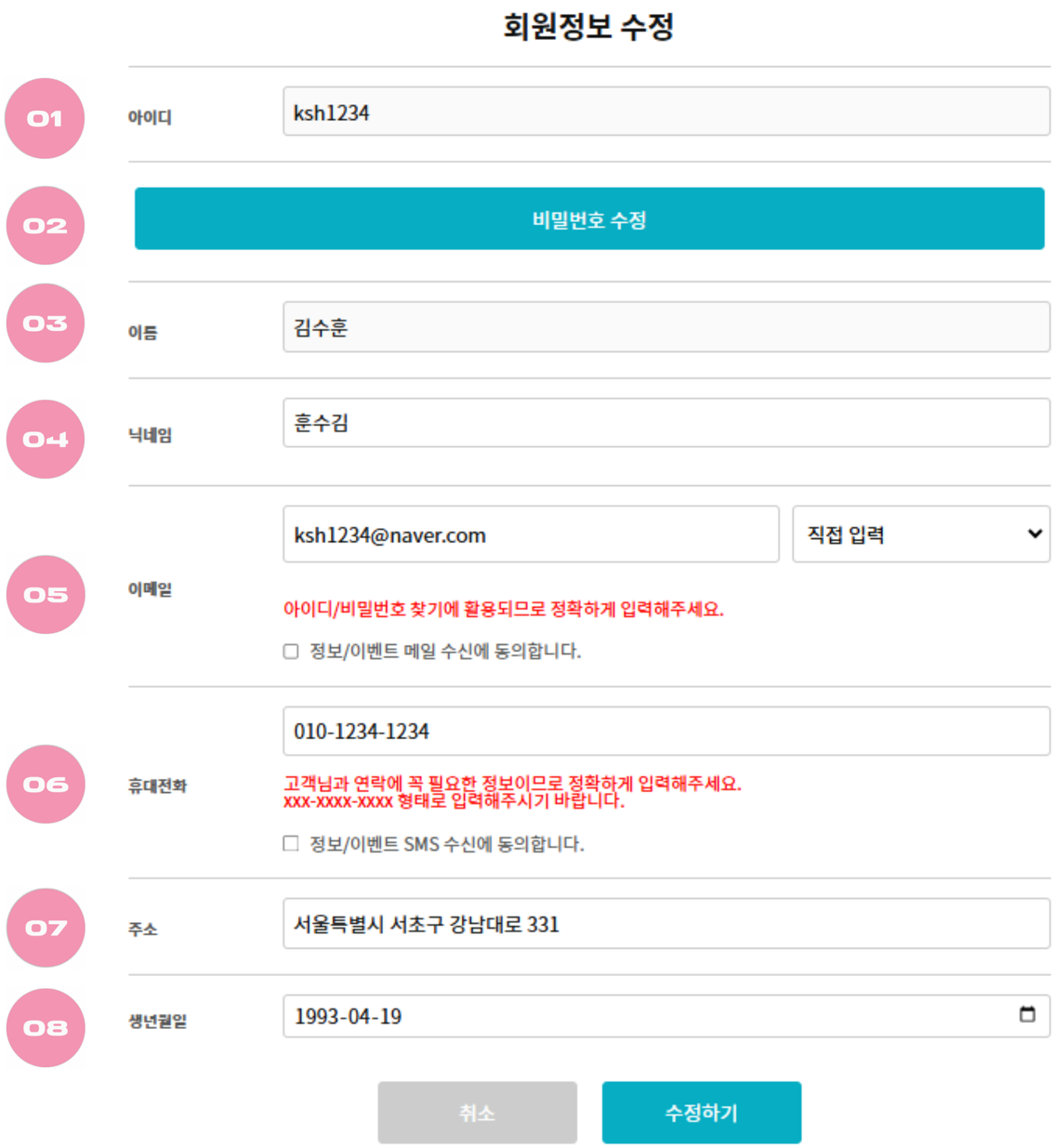
01 비밀번호 입력

02 취소 버튼(메인페이지로 이동)

03 인증하기 버튼
일치 시 회원 정보 수정 화면으로 이동

04 비밀번호 일치하지 않을 시 alert 메시지 출력

35. 회원 정보 수정



Description List

01 아이디(변경 불가)

02 비밀번호 수정(버튼 클릭 시 수정 가능)

03 이름(변경 불가)

04 닉네임(선택사항)

05 이메일

06 휴대전화

07 주소

08 생년월일(선택사항)

35. 회원 정보 수정

01

비밀번호 수정 취소

새 비밀번호

새 비밀번호 확인

02

새 비밀번호

비밀번호는 영문 대/소문자, 숫자, 특수문자 중 2가지 이상 조합으로 10자 이상이어야 합니다.

새 비밀번호 확인

비밀번호가 일치합니다.

03

새 비밀번호

사용 가능한 비밀번호입니다.

새 비밀번호 확인

비밀번호가 일치하지 않습니다.

04

새 비밀번호

사용 가능한 비밀번호입니다.

새 비밀번호 확인

비밀번호가 일치합니다.

Description List

01 비밀 번호 수정 창(취소 버튼 클릭 시 닫힘)

02 새 비밀번호가 일치하지만 유효성 충족하지 못할 시
메시지 출력
유효성은 기존 회원가입과 동일
(영문, 숫자, 특수문자 중 2가지 이상 조합 + 10자 이상)

03 유효성은 충족하지만 일치하지 않을 시
메시지 출력

04 유효성 충족 및 비밀번호 일치 시

35. 회원 정보 수정

새 비밀번호 확인 ••••••••••••

비밀번호가 일치합니다.

이름 김수훈

닉네임 01 흥민소니

중복되는 닉네임입니다.

* id varchar(50)	* pw varchar(255)	* name varchar(100)	nick varchar(50)
asdfff1134	tntntn1234	니냔뇨뇨	후루루루
bbb123	12341234	김루피는바보	강아지
ccc123	tngns0328@	손흥민	흥민소니

이름 김수훈

닉네임 02 훈수@@

닉네임은 한글, 영문, 숫자 포함 2자 이상 10자 이하이어야 합니다.

이름 김수훈

닉네임 03 훈수

사용가능한 닉네임입니다.

Description List

01 닉네임은 선택사항이지만 입력 시 중복 확인
중복 시 사용 불가 및 메시지 출력

02 유효성 충족하지 못할 시 메시지 출력
(한글, 영문, 숫자 포함 2글자 이상 10자 이하. 특수문자 불

03 중복 닉네임 없을 시 사용 가능 및 메시지 출력

35. 회원 정보 수정

01

02

ksh1234@naver.com

직접 입력

이메일

아이디/비밀번호 찾기에 활용되므로 정확하게 입력해주세요.

정보/이벤트 메일 수신에 동의합니다.

03

수훈1234@naver.com

직접 입력

이메일

이메일 형식이 올바르지 않습니다.

아이디/비밀번호 찾기에 활용되므로 정확하게 입력해주세요.

정보/이벤트 메일 수신에 동의합니다.

04

ksh1234@gmail.com

gmail.com

이메일

사용 가능한 이메일입니다.

아이디/비밀번호 찾기에 활용되므로 정확하게 입력해주세요.

정보/이벤트 메일 수신에 동의합니다.

Description List

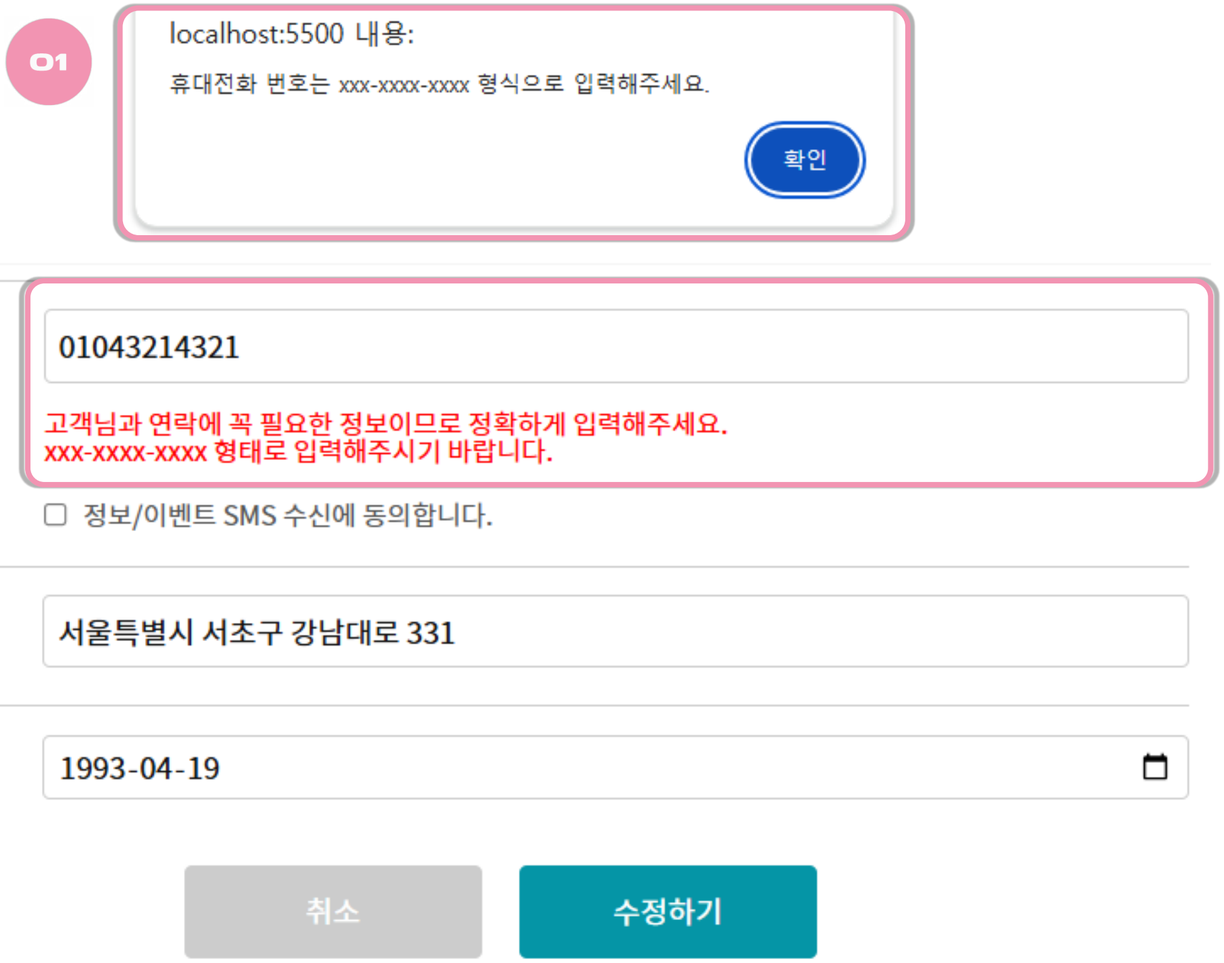
01 이메일 수정 (유효성 조건은 회원가입과 동일)
로컬파트(영문자, 숫자, ._%+- 만 가능)
@ 도메인(영문자와 숫자, .- 만 가능) .
최상위도메인(영문자만 허용, 최소 2글자 이상)
충족 못할 시 메시지 출력

02 Select의 도메인 선택 시 자동으로 이메일 아이디 뒤에 붙

03 유효성 충족하지 못할 시 메시지 출력

04 유효성 충족 시

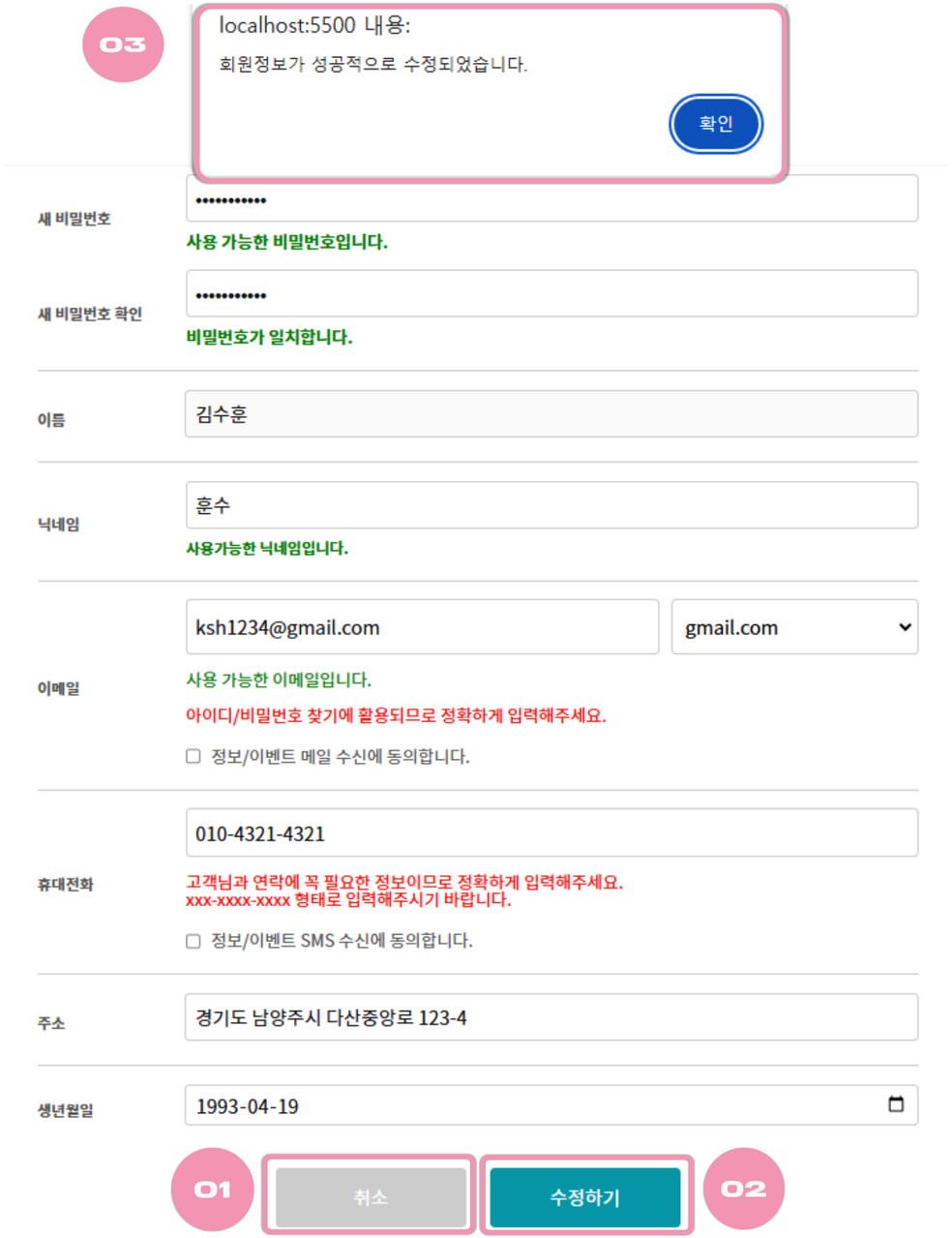
35. 회원 정보 수정



Description List

01 휴대전화 번호 수정 (유효성 조건은 회원가입과 동일)
(숫자만 허용. xxx-xxxx-xxxx의 형태만 허용)
충족 못할 시 alert 메시지 출력.

35. 회원 정보 수정



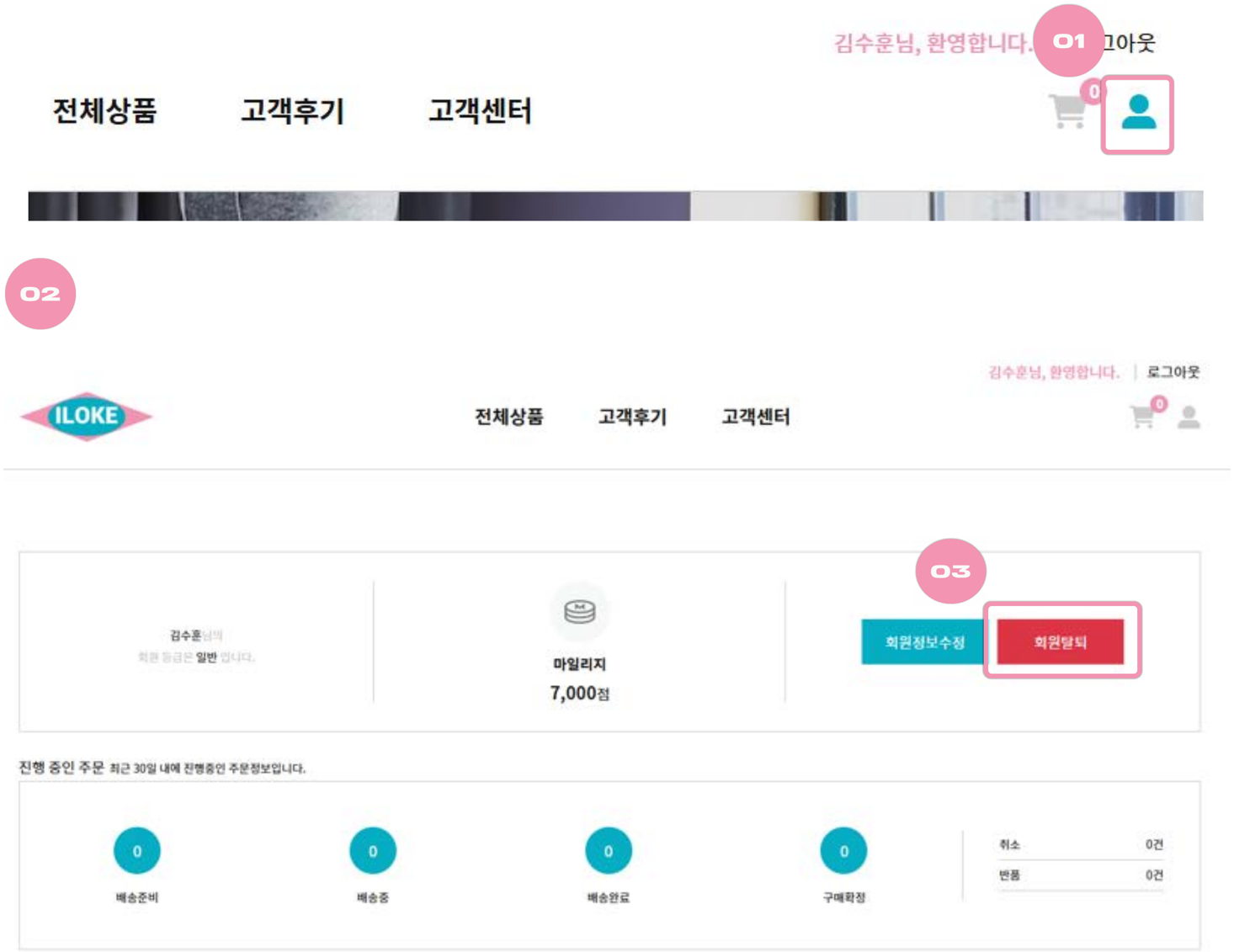
Description List

01 취소 버튼(메인 페이지로 이동)

02 수정하기 버튼

03 수정 할 정보 중복확인 및 유효성 통과 시 Alert 메시지와 함께 수정 완료 (데이터베이스에 수정된 데이터로 저장)

35. 회원 정보 수정



Description List

01 사람 아이콘 클릭 시 마이 페이지로 이동

02 마이 페이지

03 회원 탈퇴 버튼

36. 회원정보 수정 핵심코드

01

```
// 사용자 정보 불러오기
router.get('/:id', async (req, res) => {
  const userId = req.params.id; // URL 경로에서 사용자 ID를 추출
  const query = 'SELECT id, name, nick, email, phone_num, addr, DATE_FORMAT(birth_date, "%Y-%m-%d") AS birth_date FROM mem_info WHERE id = ?';

  try {
    // 데이터베이스에서 사용자 정보를 조회
    const [results] = await conn.query(query, [userId]);
    if (results.length > 0) {
      // 사용자 정보가 존재하면 성공 응답 반환
      res.json({ success: true, userData: results[0] });
    } else {
      res.json({ success: false, message: '사용자를 찾을 수 없습니다.' });
    }
  } catch (err) {
    console.error('사용자 정보 불러오기 오류:', err);
    res.status(500).json({ success: false, message: '서버 오류가 발생했습니다.' });
  }
});
```

Description List

01 회원정보 수정을 위해 사용자 정보를 조회하는 API 라우터

36. 회원정보 수정 핵심코드

01

```
// 세션 스토리지에서 아이디 가져오기
useEffect(() => {
  const storedUserId = sessionStorage.getItem("userId");
  if (storedUserId) {
    fetchUserData(storedUserId); // 사용자 정보 가져오기
  } else {
    alert("로그인이 필요합니다.");
    navigate("/login");
  }
}, [navigate]);

// 사용자 정보 불러오기
const fetchUserData = async (userId) => {
  try {
    const response = await fetchData(userId); // 서버에서 사용자 정보 요청
    if (response.data.success) {
      setFormData(response.data.userData);    // 사용자 정보를 상태에 저장
      setOgNick(response.data.userData.nick); // 기존 닉네임 저장
    } else {
      alert("사용자 정보를 불러오는데 실패했습니다.");
    }
  } catch (error) {
    alert("사용자 정보를 불러오는 중 오류가 발생했습니다.");
  }
};
```

Description List

01 로그인 상태를 확인 후 서버에서 사용자 정보를 가져와 저장하여 화면에 표시

36. 회원정보 수정 핵심코드

01

```
// 사용자 정보 수정
router.put('/update', async (req, res) => {
  // 클라이언트로부터 요청된 회원정보를 가져옴
  const { userId, pw, name, nick, email, phone_num, addr, birth_date } = req.body;
  // console.log(userId)

  try {
    let query;  // SQL 쿼리 변수
    let params; // 쿼리 매개변수 배열

    // 비밀번호가 제공된 경우 비밀번호도 업데이트
    if (pw) {
      query = `
        UPDATE mem_info
        SET pw = ?, name = ?, nick = ?, email = ?, phone_num = ?, addr = ?, birth_date = ?
        WHERE id = ?
      `;
      params = [pw, name, nick, email, phone_num, addr, birth_date, userId];
    } else {
      // 비밀번호가 제공되지 않은 경우 비밀번호를 제외하고 업데이트
      query = `
        UPDATE mem_info
        SET name = ?, nick = ?, email = ?, phone_num = ?, addr = ?, birth_date = ?
        WHERE id = ?
      `;
      params = [name, nick, email, phone_num, addr, birth_date, userId];
    }
    // 데이터베이스 업데이트 실행
    await conn.query(query, params);
    res.json({ success: true, message: '회원정보가 수정되었습니다.' });
  } catch (err) {
    console.error('회원정보 수정 오류:', err);
    res.status(500).json({ success: false, message: '회원정보 수정 중 오류가 발생했습니다.' });
  }
});

module.exports = router;
```

Description List

01 사용자 정보 수정 API 라우터

36. 회원정보 수정 핵심코드

01

```
// 폼 제출 시 정보 수정
const handleSubmit = async (e) => {
  e.preventDefault();

  if (showPwFields) {
    if (!checkPwValid(formData.pw)) {
      alert("비밀번호는 영문 대/소문자, 숫자, 특수문자 중 2가지 이상 조
      return;
    }

    if (formData.pw !== formData.pw2) {
      alert("비밀번호가 일치하지 않습니다.");
      return;
    }
  }
```

```
    const dataToSend = {
      userId: formData.id,
      pw: showPwFields ? formData.pw : undefined, // 비밀번호 필드가 보일 때만 새 비밀번호 전송
      name: formData.name,
      nick: formData.nick,
      email: formData.email,
      phone_num: formData.phone_num,
      addr: formData.addr,
      birth_date: formData.birth_date,
    };

    try {
      const response = await updateUser(dataToSend);
      if (response.data.success) {
        alert("회원정보가 성공적으로 수정되었습니다.");
        navigate("/");
      } else {
        alert("회원정보 수정 실패: " + response.data.message);
      }
    } catch (error) {
      alert("회원정보 수정 중 오류가 발생했습니다.");
    }
  };
```

Description List

01 수정된 회원 정보 제출 함수

36. 회원정보 수정 핵심코드

01

```
// 닉네임 중복 확인
router.post("/", async (req, res) => {
    const { nick } = req.body;  // 요청 본문에서 닉네임 추출

    try {
      // 데이터베이스에서 입력된 닉네임이 존재하는지 확인
      const [results] = await conn.query("SELECT COUNT(*) AS count FROM mem_info WHERE nick = ?", [nick]);

      // 닉네임이 존재하면 true, 없으면 false
      const isDuplicate = results[0].count > 0;
      res.json({ isDuplicate });  // 중복 여부를 클라이언트에 반환
    } catch (error) {
      res.status(500).json({ isDuplicate: false, message: "서버 오류 발생" });
    }
  });
```

02

```
// 닉네임 중복 확인 함수
const checkNickDuplicate = async (nick) => {
  // 현재 닉네임과 같으면 중복 체크를 하지 않음
  if (nick === ogNick) {
    setNickMsg("");
    setIsNickDuplicate(false);
    return;
  }

  if (!validNick(nick)) {
    setNickMsg("닉네임은 한글, 영문, 숫자 포함 2자 이상 10자
    setIsNickDuplicate(true);
    return;
  }
```

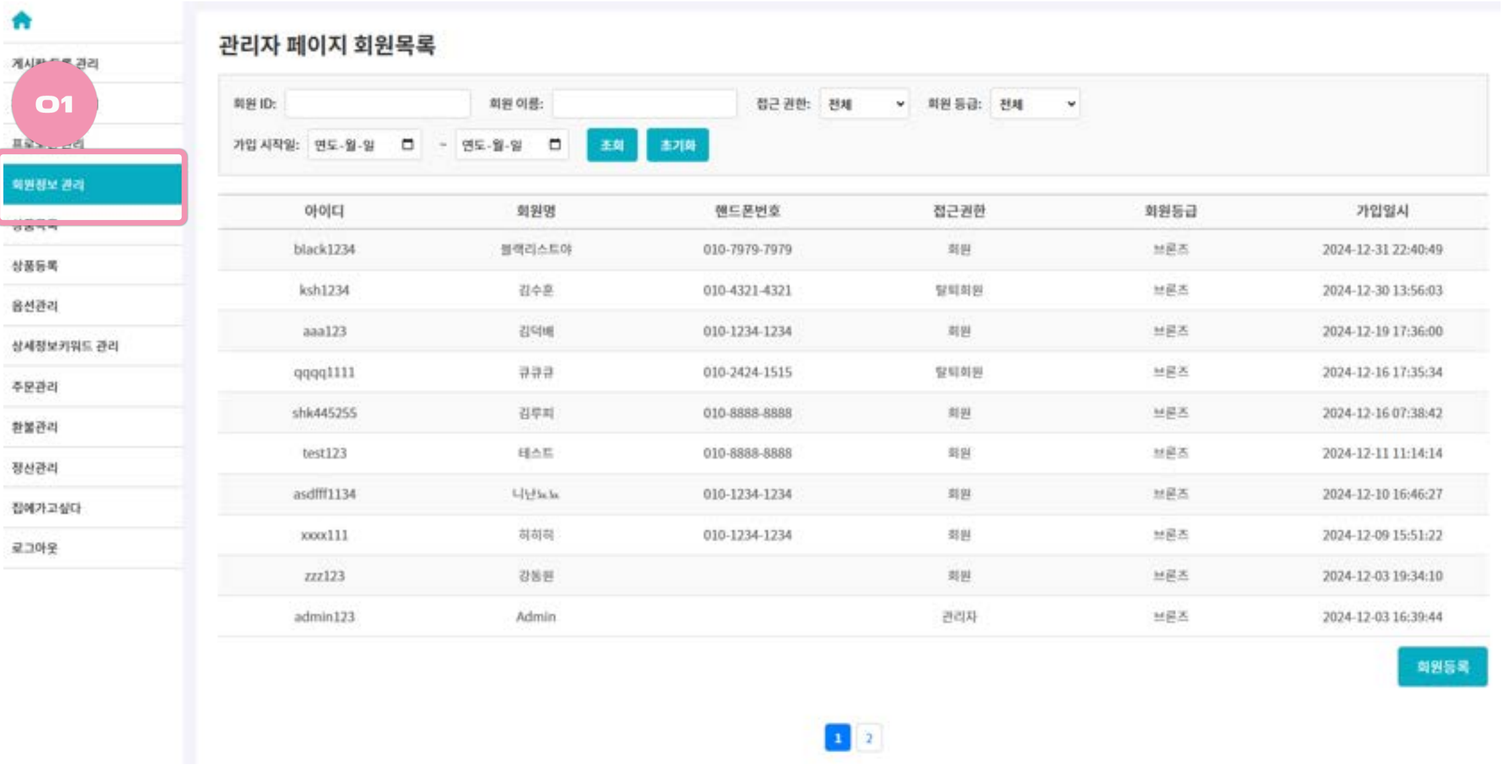
```
  try {
    const response = await checkNickDup(nick);
    if (response.data.isDuplicate) {
      setNickMsg("중복되는 닉네임입니다.");
      setIsNickDuplicate(true)
    } else {
      setNickMsg("사용가능한 닉네임입니다.");
      setIsNickDuplicate(false);
    }
  } catch (error) {
    setNickMsg("닉네임 중복 확인 중 오류가 발생했습니다.");
    setIsNickDuplicate(true);
  }
};
```

Description List

01 닉네임 중복 여부 확인 API 라우터

02 닉네임 중복 여부를 서버에 요청해 유효성을 검증하는 닉네임 중복 확인 함수

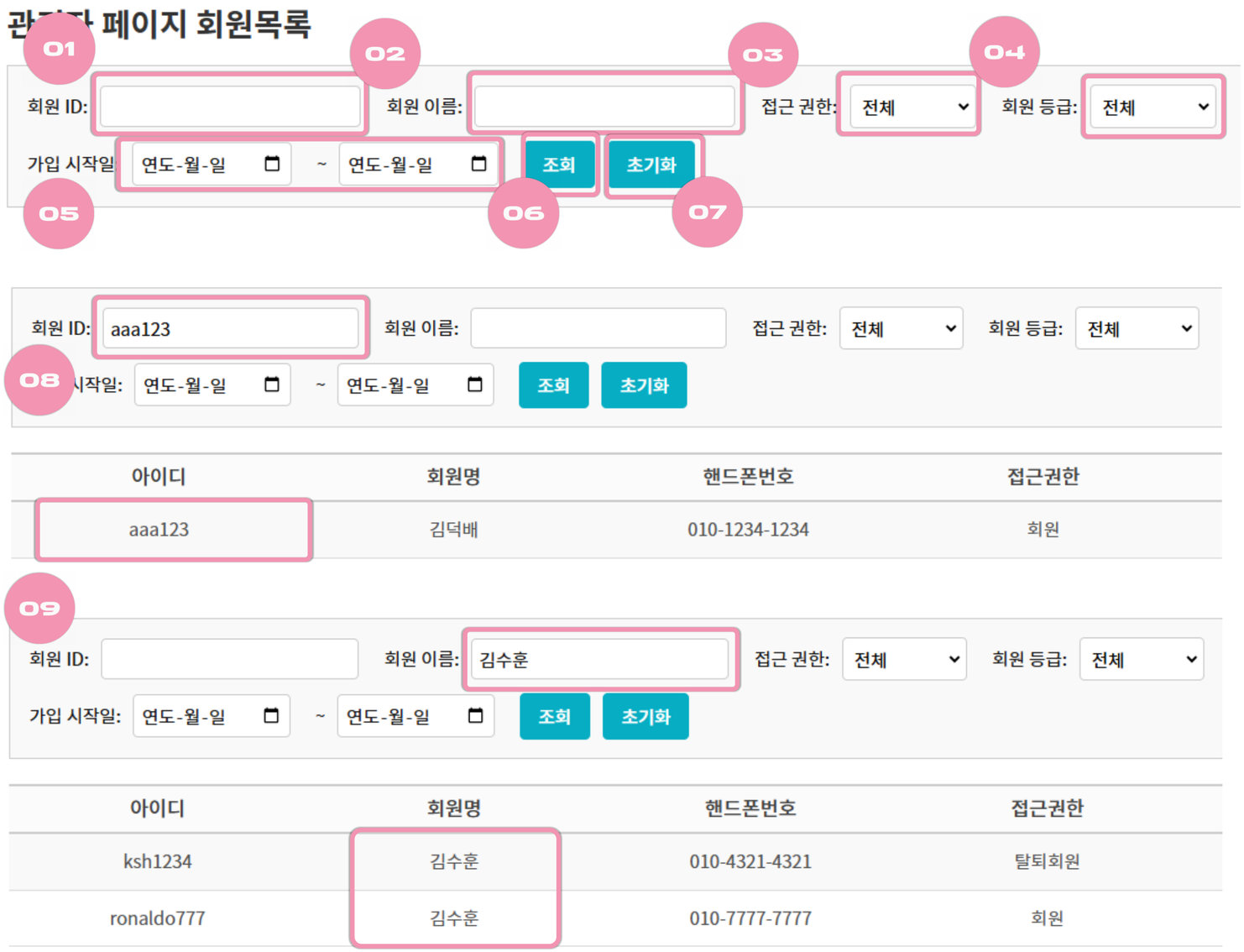
37. 관리자 페이지 회원정보 관리



Description List

01 관리자 페이지 회원 정보 관리

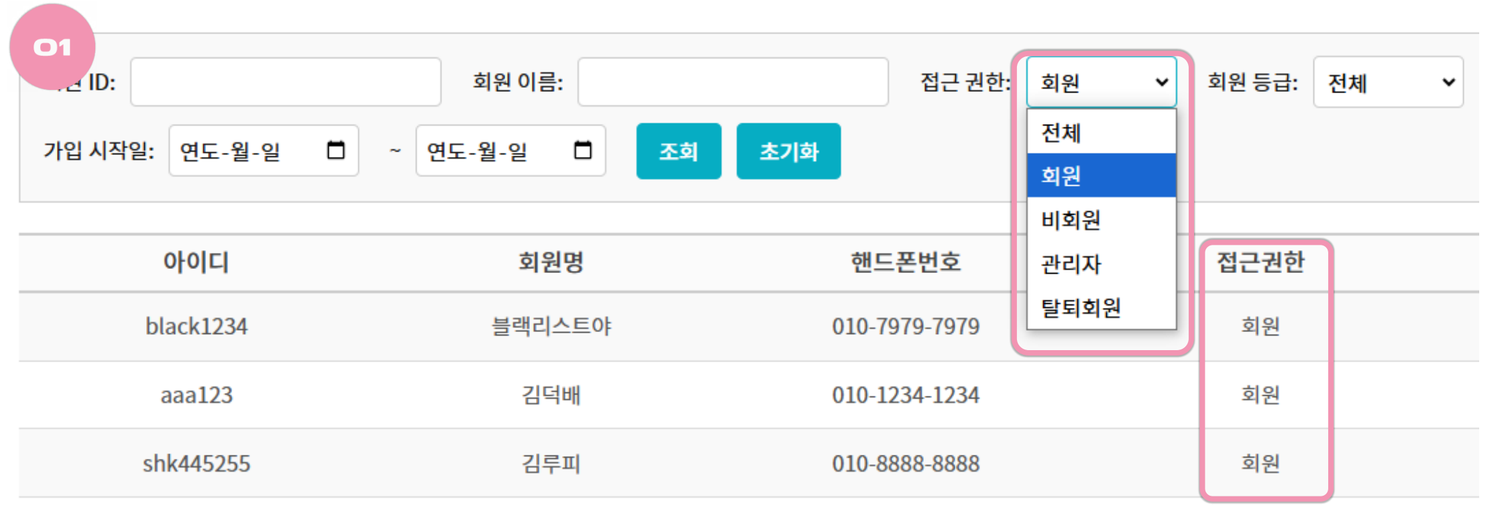
37. 관리자 페이지 회원정보 관리

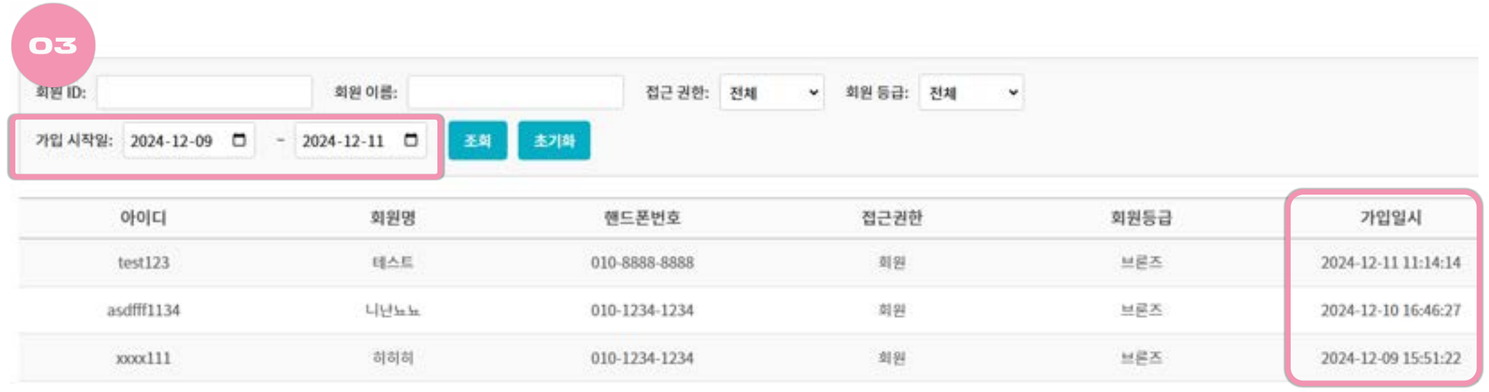


Description List

- 01 아이디로 조회
- 02 회원 이름으로 조회
- 03 접근 권한으로 조회(전체, 회원, 비회원, 탈퇴회원)
- 04 회원 등급으로 조회
- 05 가입 시작일로 조회
- 06 조회 버튼
- 07 초기화(회원 전체목록 보여줌)
- 08 회원 아이디로 조회 결과
- 09 회원 이름으로 조회 결과

37. 관리자 페이지 회원정보 관리





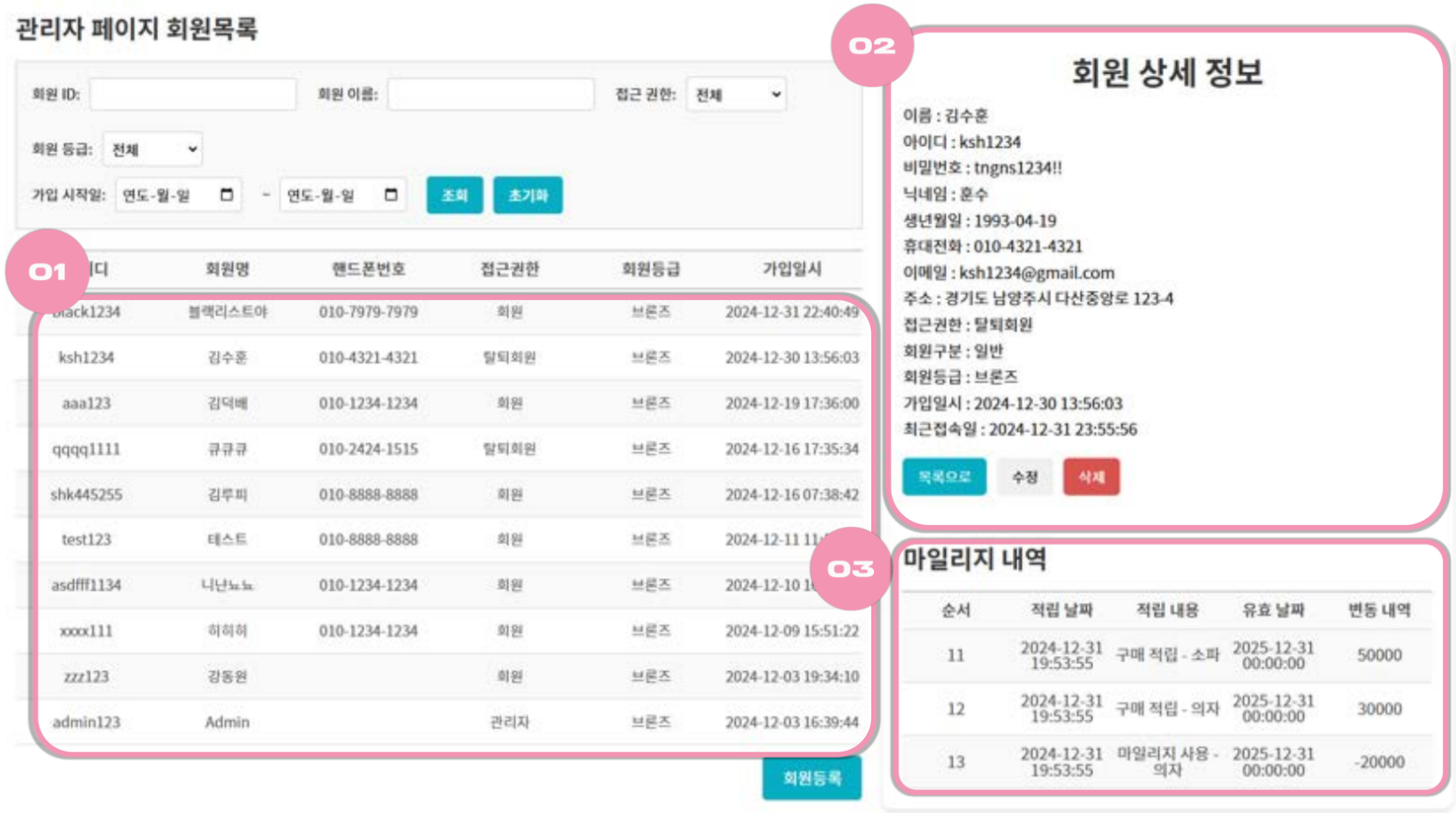
Description List

01 접근 권한으로 조회 결과

02 회원 등급으로 조회 결과

03 가입 일시로 조회 결과

37. 관리자 페이지 회원정보 관리



Description List

01 회원 목록

02 회원 상세 정보
(회원 목록에서 조회 하고자 하는 회원 클릭 시 출력)

03 회원 마일리지 내역

37. 관리자 페이지 회원정보 관리

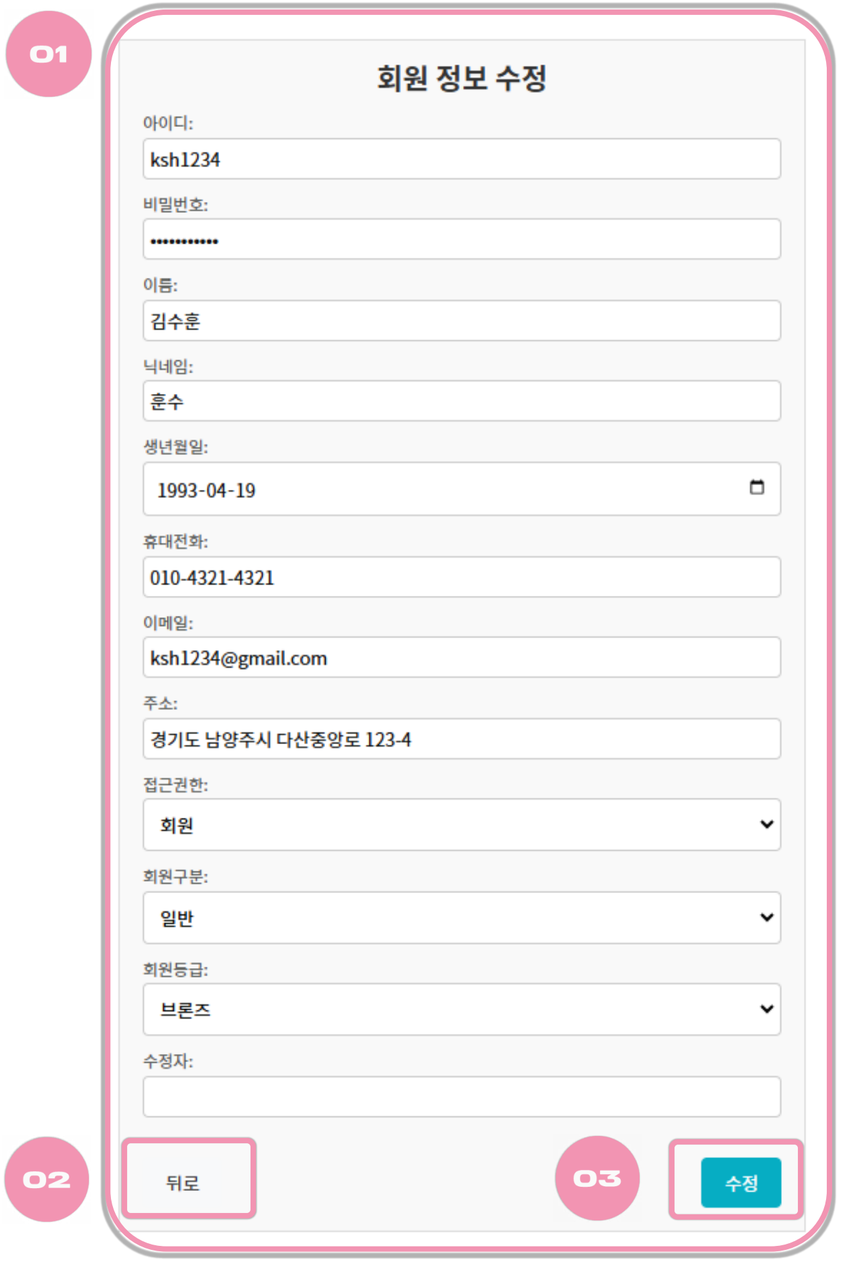
Description List

01 목록으로
(클릭 시 상세정보 창 사라지고 회원 목록만 보여짐)

02 수정
(클릭 시 회원 정보 수정 창 출력)

03 삭제
(클릭 시 회원 정보 삭제)

37. 관리자 페이지 회원정보 관리



Description List

01 회원 정보 수정 폼

02 뒤로
(회원 정보 수정 폼 사라지고 회원 목록만 출력)

03 수정

37. 관리자 페이지 회원정보 관리

01

회원 정보 수정

아이디:
ksh1234

비밀번호:
••••••••••••

이름:
김수훈

닉네임:
훈수

생년월일:
1993-04-19

휴대전화:
010-4321-4321

이메일:
ksh1234@gmail.com

주소:
경기도 남양주시 다산중앙로 123-4

접근권한:
회원

회원구분:
일반

회원등급:
브론즈

수정자:

뒤로 수정

02

회원 정보 수정

아이디:
ksh1234

비밀번호:
••••••••••••

이름:
김수훈

닉네임:
훈수

생년월일:
1993-04-19

휴대전화:
010-1234-1234

이메일:
ksh1234@naver.com

주소:
경기도 남양주시 다산중앙로 777

접근권한:
관리자

회원구분:
일반

회원등급:
브론즈

수정자:

뒤로 수정

Description List

01 수정 전

02 수정 후

03 수정 후 회원 상세 정보

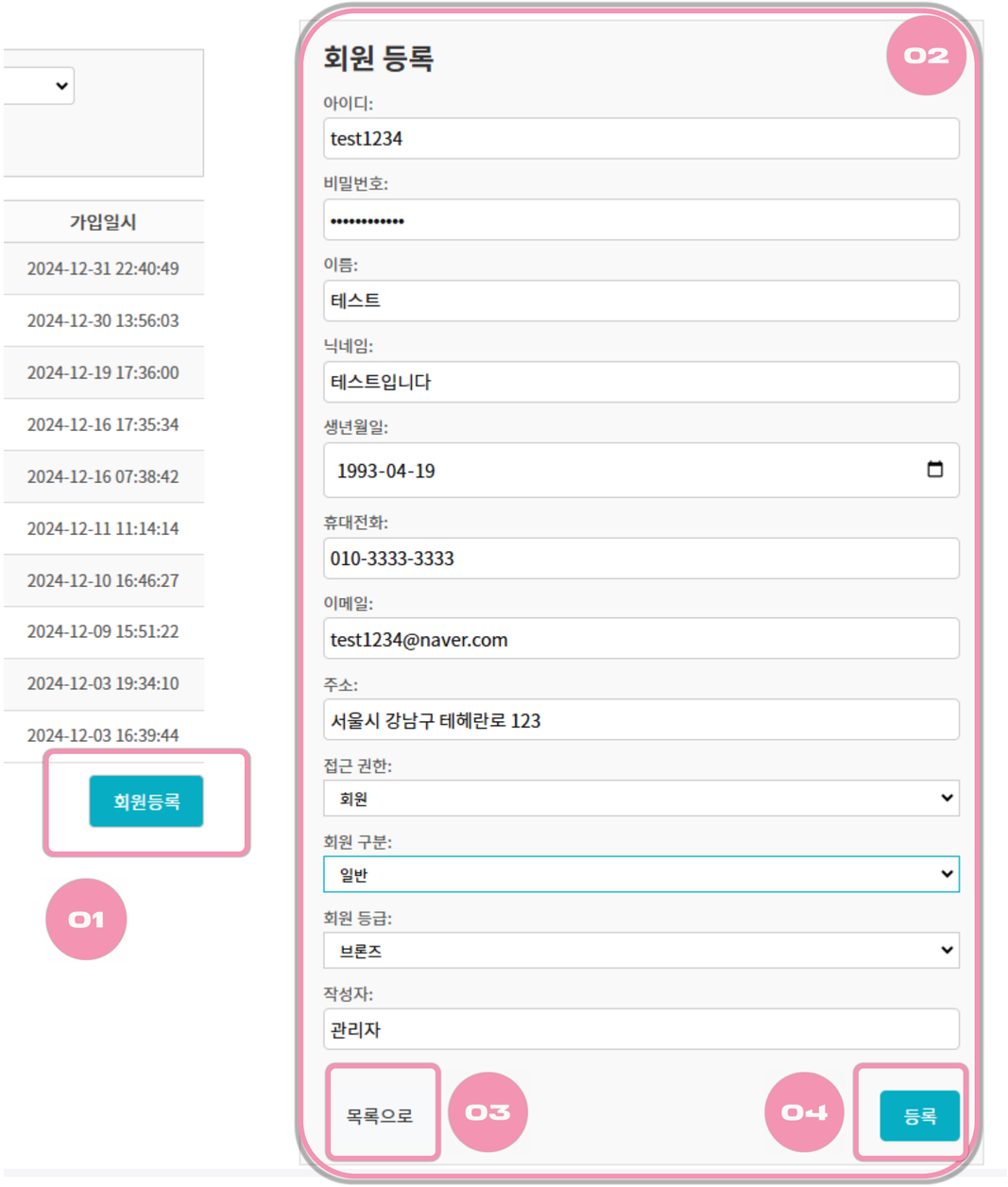
03

회원 상세 정보

이름 : 김수훈
아이디 : ksh1234
비밀번호 : tngns1234!!
닉네임 : 훈수
생년월일 : 1993-04-19
휴대전화 : 010-1234-1234
이메일 : ksh1234@naver.com
주소 : 경기도 남양주시 다산중앙로 777
접근권한 : 관리자
회원구분 : 일반
회원등급 : 브론즈
가입일시 : 2024-12-30 13:56:03
최근접속일 : 2025-01-01 00:32:29

목록으로 수정 삭제

37. 관리자 페이지 회원정보 관리



Description List

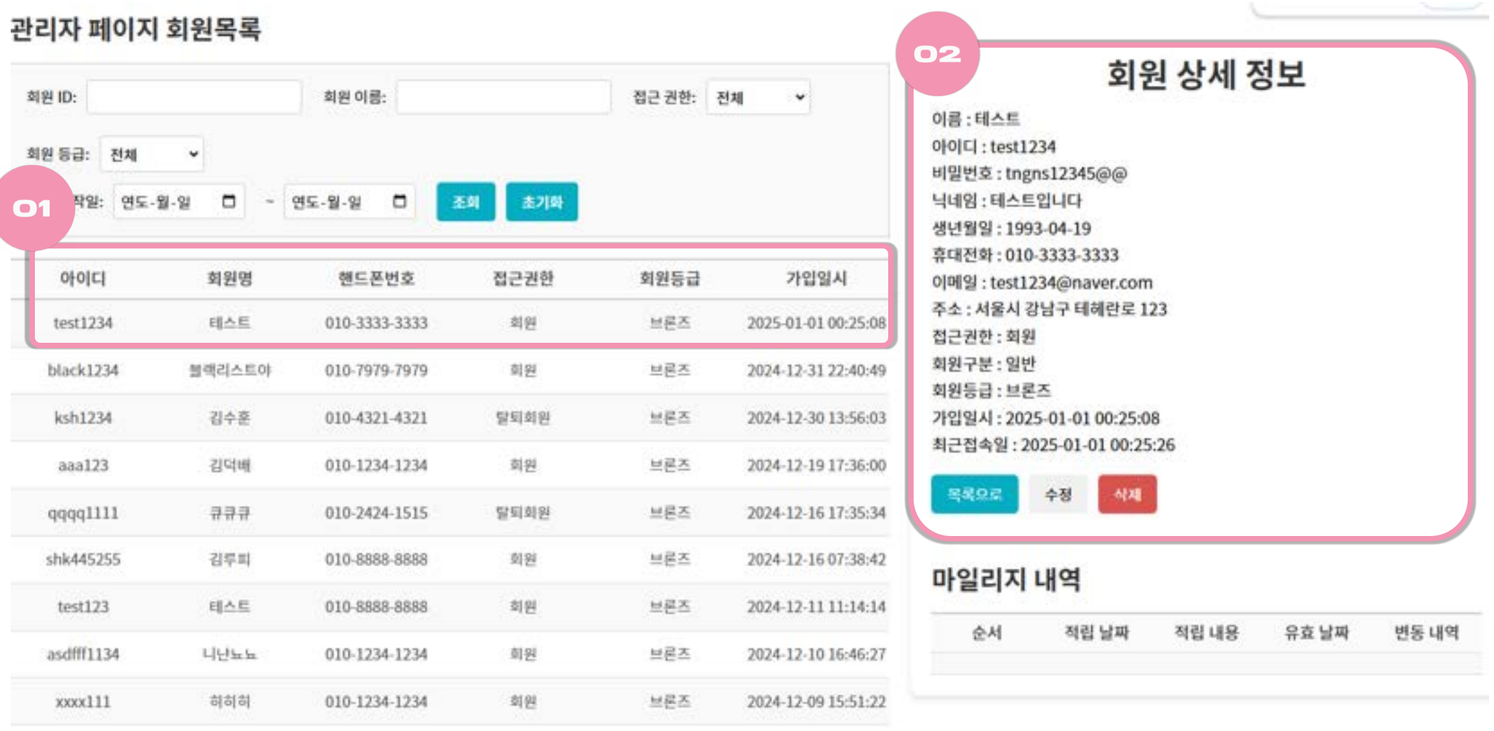
01 회원 등록
클릭 시 오른쪽에 회원 등록 폼 출력

02 회원 등록 폼

03 목록으로
클릭 시 회원 등록 폼 사라지고 회원 목록만 출력

04 등록
입력한 정보가 등록됨

37. 관리자 페이지 회원정보 관리



Description List

01 회원 목록에 새로 생성 된 회원 추가

02 회원 등록으로 생성된 회원 상세 정보 조회

37. 관리자 페이지 회원정보 관리

02

localhost:3000 내용:

삭제되었습니다.

확인

접근권한	회원등급	가입일시
회원	브론즈	2025-01-01 00:25:08
회원	브론즈	2024-12-31 22:40:49
관리자	브론즈	2024-12-30 13:56:03
회원	브론즈	2024-12-19 17:36:00
탈퇴회원	브론즈	2024-12-16 17:35:34
회원	브론즈	2024-12-16 07:38:42

회원 상세 정보

이름 : 테스트
아이디 : test1234
비밀번호 : tngns12345@@
닉네임 : 테스트입니다
생년월일 : 1993-04-19
휴대전화 : 010-3333-3333
이메일 : test1234@naver.com
주소 : 서울시 강남구 테헤란로 123
접근권한 : 회원
회원구분 : 일반
회원등급 : 브론즈
가입일시 : 2025-01-01 00:25:08
최근접속일 : 2025-01-01 01:46:58

목록으로 수정 삭제 01

관리자 페이지 회원목록

회원 ID: 회원 이름:

회원 등급: 전체

03 연도-월-일 ~ 연도-월-일 조회 초기화

아이디	회원명	핸드폰번호	접근권한
test1234	테스트	010-3333-3333	회원
black1234	블랙리스트야	010-7979-7979	회원
ksh1234	김수훈	010-4321-4321	탈퇴회원

관리자 페이지 회원목록

회원 ID: 회원 이름: 접근 권한:

04 시작일: 연도-월-일 ~ 연도-월-일 조회 초기화

아이디	회원명	핸드폰번호	접근권한
black1234	블랙리스트야	010-7979-7979	회원
ksh1234	김수훈	010-1234-1234	관리자

Description List

01 삭제(회원 정보 삭제)

02 삭제 버튼 클릭 시 삭제되었다는 alert 출력

03 삭제 전 회원 목록

04 삭제 후 갱신된 회원 목록

37. 관리자 페이지 회원정보 관리 핵심코드

01

```
const [selectedId, setSelectedId] = useState(null); // 선택된 회원 ID
const [showJoin, setShowJoin] = useState(false); // 회원등록 화면 표시 여부
const [showModify, setShowModify] = useState(false); // 회원 수정 화면 표시 여부
```

02

```
import MemberDetail from "./MemberDetail"; // MemberDetail 컴포넌트 임포트
import MemberJoin from "./MemberJoin"; // MemberJoin 컴포넌트 임포트
import MemberModify from "./MemberModify"; // MemberModify 컴포넌트 임포트

{/* 선택된 ID가 있고 수정 화면이 아닐 경우, 회원 상세 정보 출력 */}
{selectedId && !showModify && (
  <MemberDetail
    id={selectedId}
    fetchMembers={fetchMembers}
    setSelectedId={setSelectedId}
    setShowModify={setShowModify}
  />
)}

{/* 회원 수정 화면 출력 */}
{showModify && (
  <MemberModify
    id={selectedId}
    setShowModify={setShowModify}
    fetchMembers={fetchMembers}
  />
)}

{/* 회원 등록 화면 출력 */}
{showJoin && (
  <MemberJoin fetchMembers={fetchMembers} setShowJoin={setShowJoin} />
)}
```

Description List

01 동적인 UI 상태 관리

02 회원 등록, 상세, 수정을 하나의 컴포넌트 내에서 관리 및 세가지 주요 기능을 컴포넌트화

38. 라우팅 동적 할당(1)

01

```
import { BrowserRouter as Router, Routes, Route } from 'react-router-dom';
import MainPage from './pages/MainPage';

function App() {
    return (
        <Router>
            <Routes>
                <Route path="/*" element={<MainPage />} />
            </Routes>
        </Router>
    );
}

export default App;
```

Description List

01 라우팅 분기

38. 라우팅 동적 할당(2)

02

```
import { useContent } from '../hooks/pageFunction';
import Footer from './Footer';
import Header from './Header';

function MainPage(props) {
    const content = useContent(location.pathname); // useContent 훅 호출

    const isMainPage = location.pathname === '/'; // 메인 페이지 확인

    return (
        <div>
            <Header />
            <div id="cont" className={!isMainPage ? 'content' : ''}>
                {content} {/* 동적으로 렌더링될 컴포넌트 */}
            </div>
            <Footer />
        </div>
    );
}

export default MainPage;
```

Description List

02 가변적 화면 분배

38. 라우팅 동적 할당(3)

03

```
const [content, setContent] = useState(<Home />);
    const location = useLocation(); // 현재 경로 가져오기

    useEffect(() => {
        const path = location.pathname;

        if (!path || path === "/" || path === "/home") {
            setContent(<Home />);
        } else if (path === "/goods/myPage") {
            setContent(<MyPage />);
        }
```

Description List

03 페이지 분기 처리

39. 페이징 처리

01

```
// 현재 페이지 데이터 슬라이싱
const indexOfLastItem = currentPage * itemsPerPage;
const indexOfFirstItem = indexOfLastItem - itemsPerPage;
const currentItems = boards.slice(indexOfFirstItem, indexOfLastItem);
```

02

```
// 페이지 변경 처리
const handlePageChange = (page) => {
  if (page >= 1 && page <= totalPages) {
    setCurrentPage(page);
  }
};
```

03

```
<PaginationControls
  currentPage={currentPage}
  totalPages={totalPages}
  onPageChange={handlePageChange}
/>
```

Description List

01 초기값 설정

02 페이징 변경 함수

03 페이징 출력

페이징 출력 화면

카카오 로그인

Description List

01 로그인 클릭

02 로그인 페이지 접속

03 카카오 로그인 클릭

카카오 로그인

01

```
<div className={styles.snsLoginSection}>
  <div className={styles.snsLoginTitle}>SNS 간편 로그인</div>
  <KakaoLoginButton />
</div>
```

02

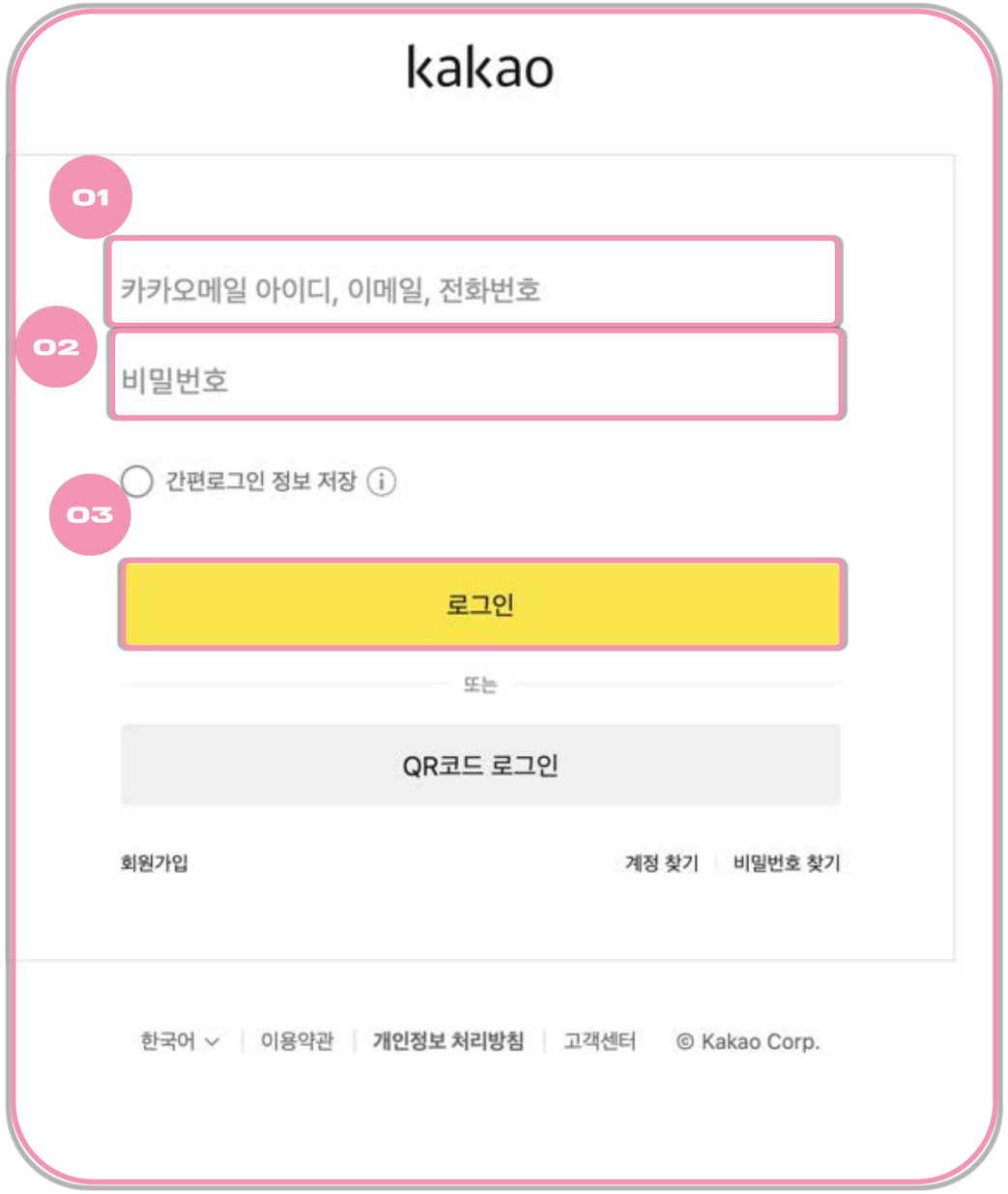
```
useEffect(() => {
  const urlParams = new URLSearchParams(window.location.search);
  const authCode = urlParams.get('code');

  if (authCode) {
    console.log('인증 코드:', authCode);
    getAccessToken(authCode);
  } else {
    console.error('인증 코드가 없습니다.');
  }
}, []);

const getAccessToken = async (authCode) => {
  const tokenUrl = "https://kauth.kakao.com/oauth/token";
  const data = new URLSearchParams({
    grant_type: "authorization_code",
    client_id: REST_API_KEY,
    redirect_uri: REDIRECT_URI,
    code: authCode,
  });

  console.log("토큰 요청 데이터:", {
    grant_type: "authorization_code",
    client_id: REST_API_KEY,
    redirect_uri: REDIRECT_URI,
    code: authCode,
  });
```

Description List

01 KakaoLoginButton 컴포넌트

02 KakaoLoginButton 처리

카카오 로그인



Description List

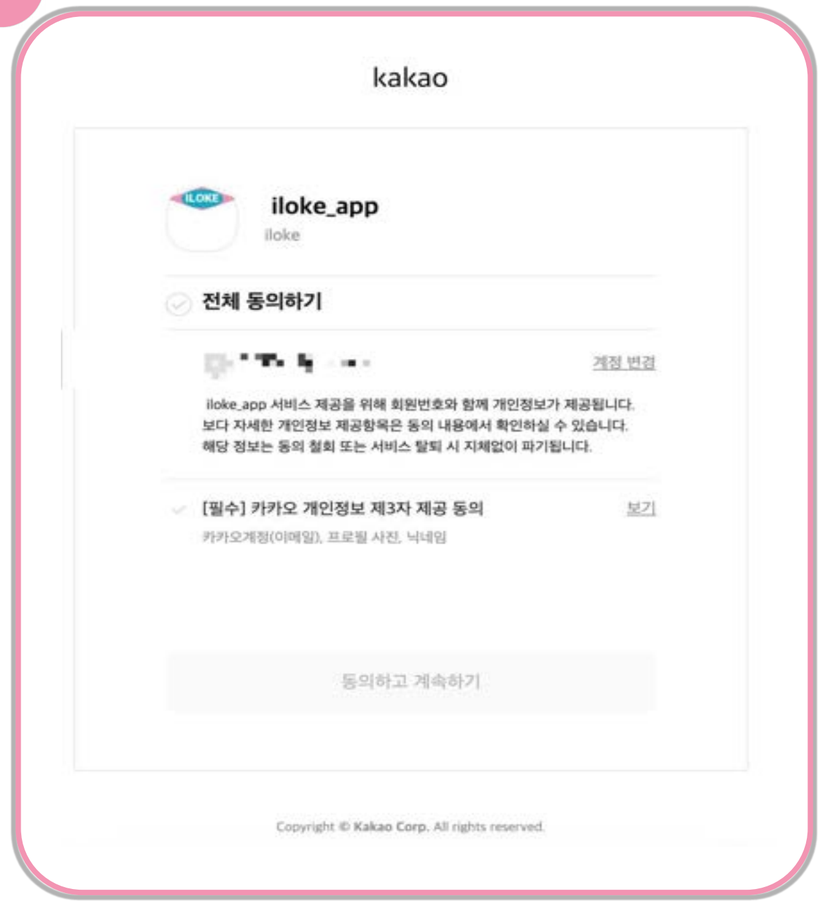
01 카카오 아이디,이메일,전화번호 입력

02 비밀번호 입력

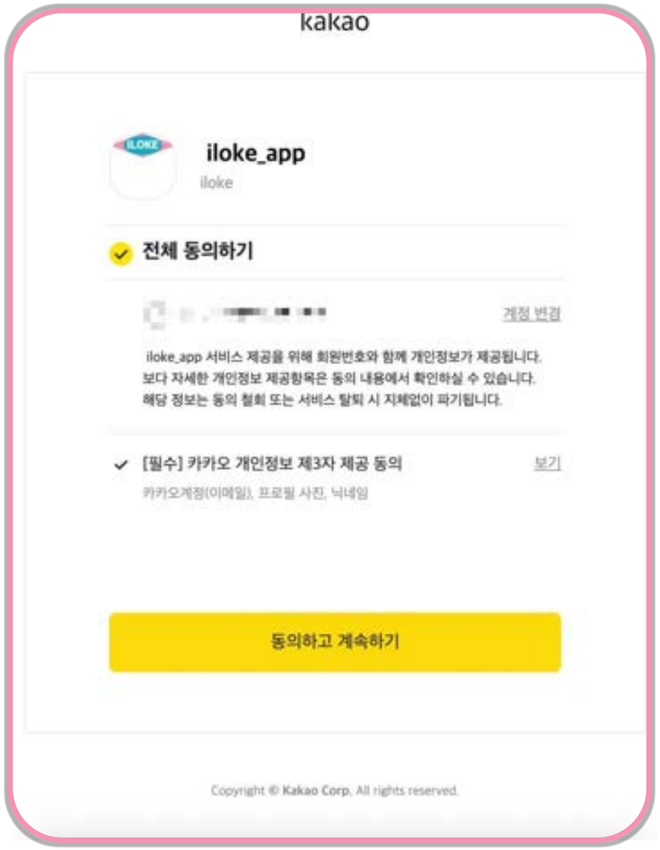
03 로그인 클릭

카카오 로그인

01



02



Description List

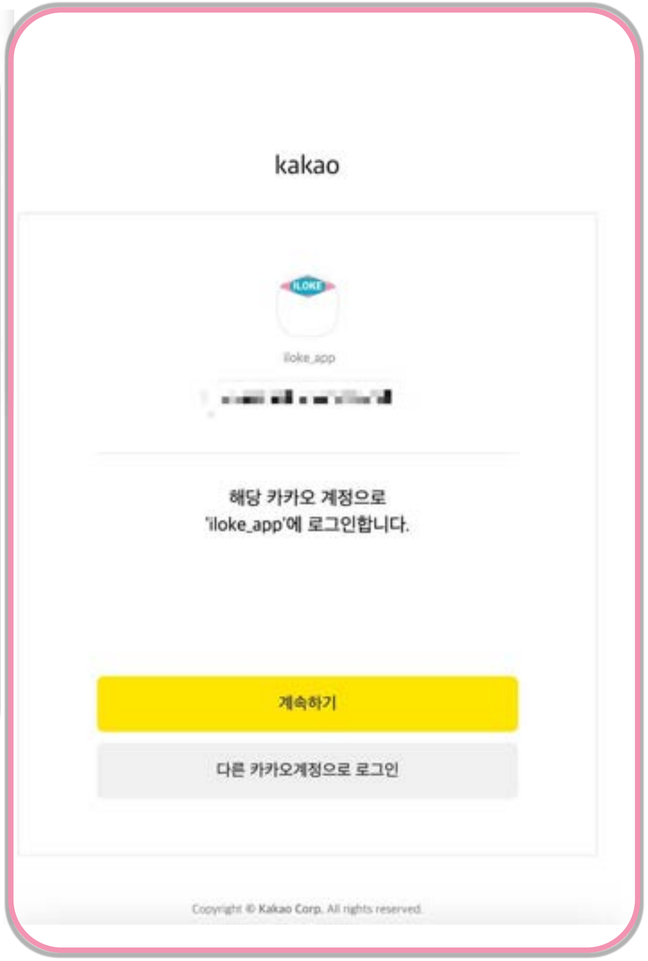
01 가입되지 않은 회원일 경우:
필수 항목 동의

02 동의하고 계속하기 버튼 클릭

카카오 로그인

01

02



Description List

01 가입된 회원일 경우 : 2단계 인증 (필요시)

02 인증 후 계속하기 버튼 클릭

카카오 로그인

01

Description List

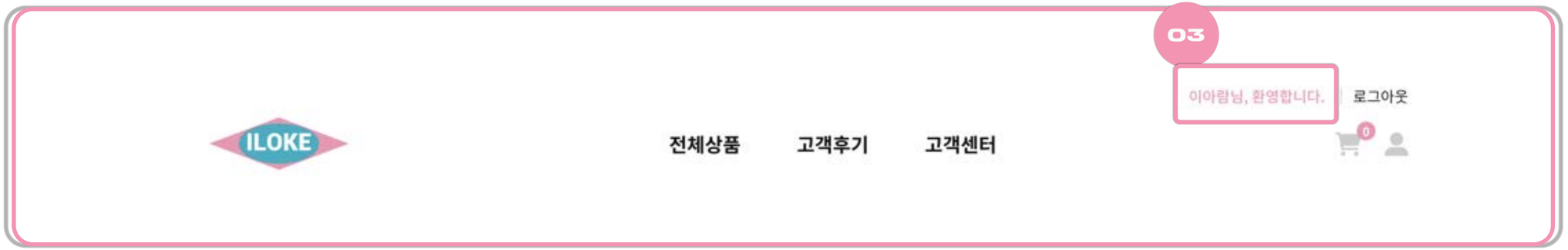
01 카카오 계정 이름으로 alert창 표시

02 카카오 계정 이름으로 로그인

03 '사용자님, 환영합니다.' 문구 표시

02

03



카카오 로그인

01

```
{isLoggedIn ? (
  <span id="welcomeMessage">{username}님, 환영합니다.</span>
) : (
```

02

```
try {
  const response = await fetch(tokenUrl, {
    method: "POST",
    headers: {
      "Content-Type": "application/x-www-form-urlencoded",
    },
    body: data,
  });

  if (response.ok) {
    const result = await response.json();
    console.log("액세스 토큰:", result.access_token);
    getUserInfo(result.access_token);
  } else {
    console.error("토큰 요청 실패:", response.statusText);
    const errorResponse = await response.json();
    console.error("오류 응답:", errorResponse);
  }
} catch (error) {
  console.error("토큰 요청 중 오류 발생:", error);
}
};
```

Description List

01 'Username님, 환영합니다.'

02 토큰 요청 처리

카카오 로그인

01

Description List

01 1:1문의 클릭

02 카카오 계정으로 작성자 이름 표시

02

bd202501090002	재입고 문의할게요	1	user07	2025-01-09 12:13:58	답변필요
bd202501100001	상품2 언제나오나요	2	이아람	2025-01-10 14:35:28	답변필요
bd202501100004	상품22222	1	이아람	2025-01-10 14:39:38	답변필요

카카오 로그인

01

```
{item.reg_id === sessionStorage.getItem('userId') && sessionStorage.getItem('userType') === 'kakao'
  ? sessionStorage.getItem('nickname') || '알 수 없는 사용자' // 내가 작성한 글이라면 카카오 닉네임 표시
  : item.reg_id // 다른 사용자가 작성한 글은 원래 등록된 ID 표시
}
```

02

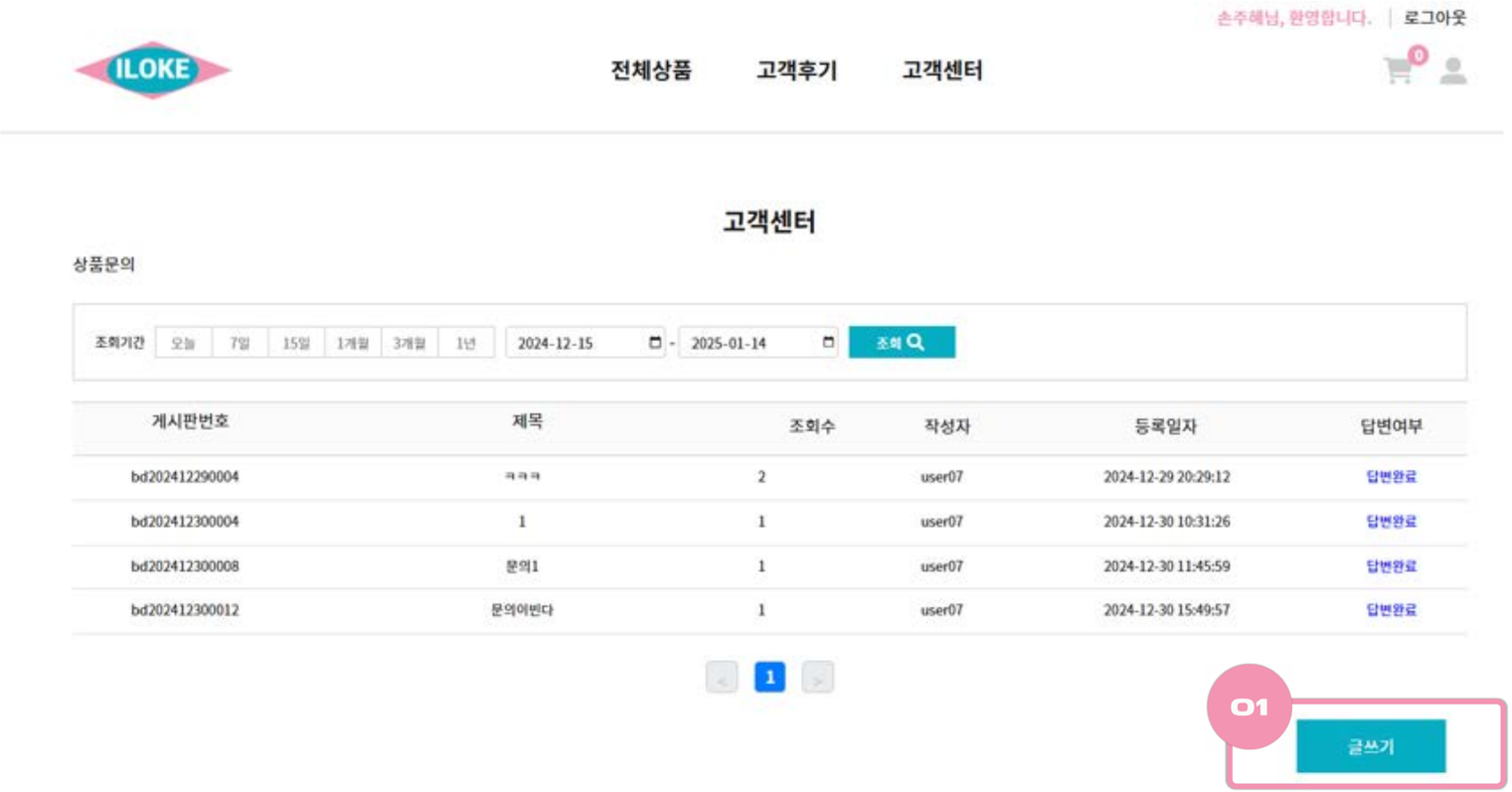
```
const getUserInfo = async (accessToken) => {
  try {
    const response = await fetch("https://kapi.kakao.com/v2/user/me", {
      method: "GET",
      headers: {
        Authorization: `Bearer ${accessToken}`,
      },
    });

    if (response.ok) {
      const userInfo = await response.json();
      console.log("사용자 정보:", userInfo);
      // 로그인 상태를 세션에 저장
      sessionStorage.setItem('userId', userInfo.id); // kakaoid->userid
      sessionStorage.setItem('userType', 'kakao'); // 사용자 유형 저장
      sessionStorage.setItem('email', userInfo.kakao_account?.email || null);
      sessionStorage.setItem('nickname', userInfo.properties?.nickname || 'unknown');
      alert(`환영합니다, ${userInfo.properties?.nickname || '사용자'}님!`);
      // 필요한 경우 리디렉션
      window.location.href = '/home';
    } else {
      console.error("사용자 정보 요청 실패:", response.statusText);
    }
  } catch (error) {
    console.error("사용자 정보 요청 중 오류 발생:", error);
  }
};
```

Description List

01 카카오 로그인 후 작성한 닉네임 표시

02 getUserInfo 함수 사용하여 사용자 닉네임 가져오기

Quill 스마트에디터



Description List

01 글쓰기 버튼 클릭

Quill 스마트에디터

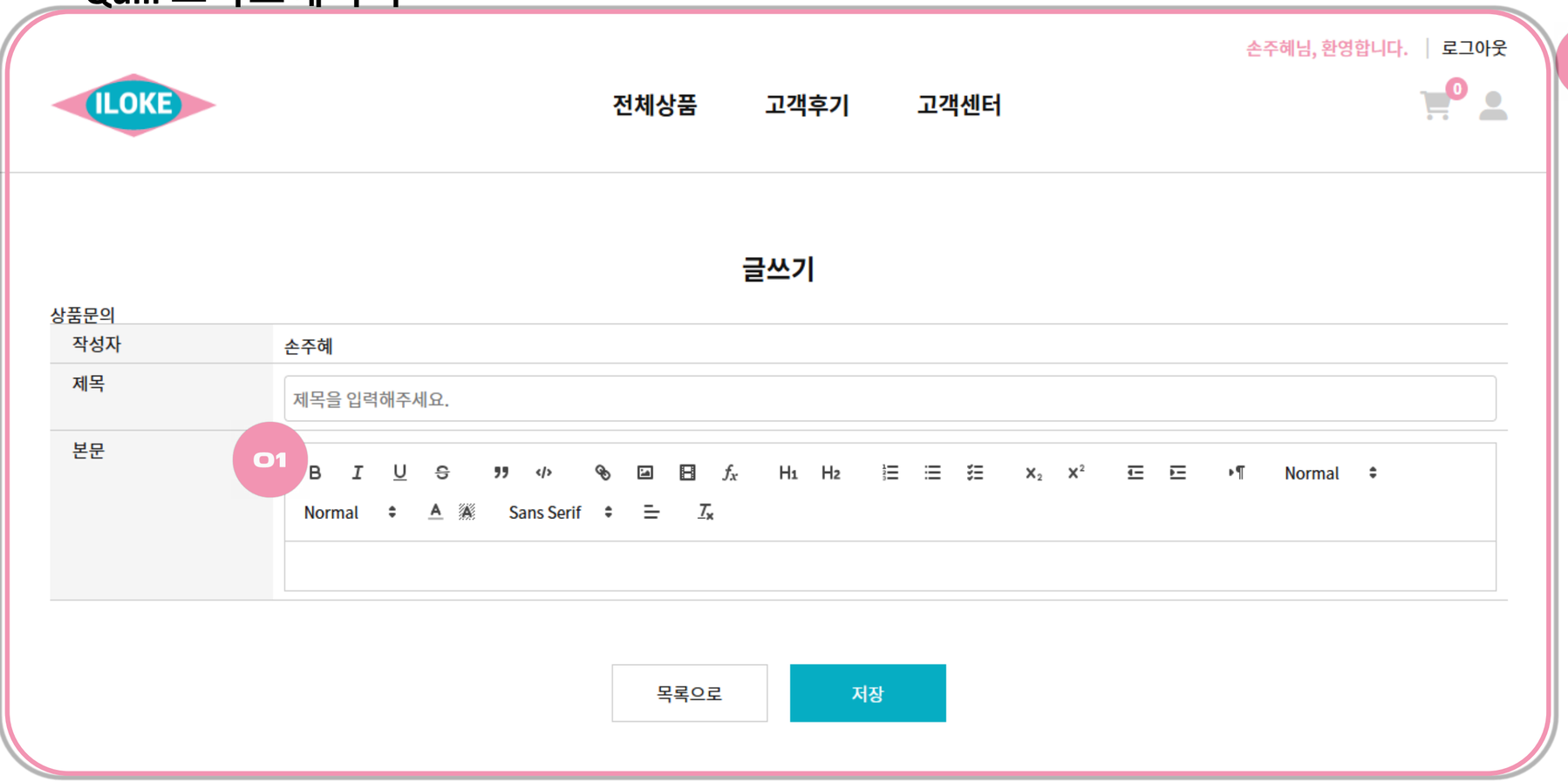
01

```
<PaginationControls
    currentPage={currentPage}
    totalPages={totalPages}
    onPageChange={handlePageChange}
 />

 <div>
    <button type="button" className={styles.write_btn} onClick={() => {
        navigate(`/notice/itemWrite`, {
            state: { code: selectedCode, previousPage: '/notice/one_inquery'}
        });
    }}>글쓰기</button>
 </div>
```

Description List

01 글쓰기 버튼 클릭

Quill 스마트에디터



Description List

01 글쓰기 스마트 에디터

Quill 스마트에디터

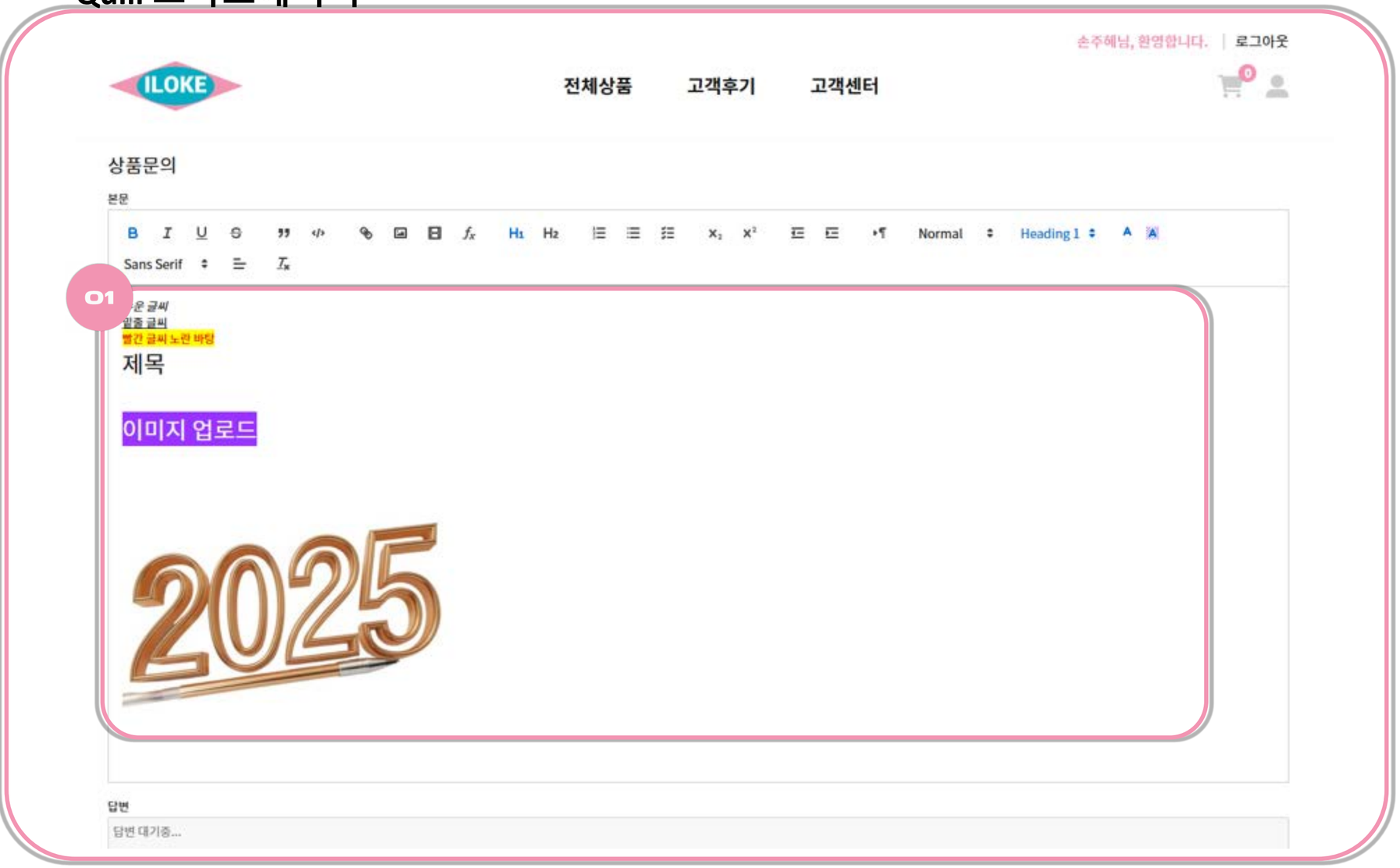
01

```
<tr>
    <th>본문</th>
    <td>
        <div ref={quillRef} id="editor" className={styles.textfield}></div> {/* quillRef를 추가 */}
    </td>
</tr>
{code === 'pr' ? (
    <tr>
        <th>첨부파일</th>
        <td>
            <input className={styles.file_upload} type="file" name="upfile" onChange={handleFileChange} />
        </td>
    </tr>
) : null}
```

Description List

01 스마트 에디터 코드 장착

Quill 스마트에디터



Description List

01 스마트 에디터로 글쓰기

Quill 스마트에디터

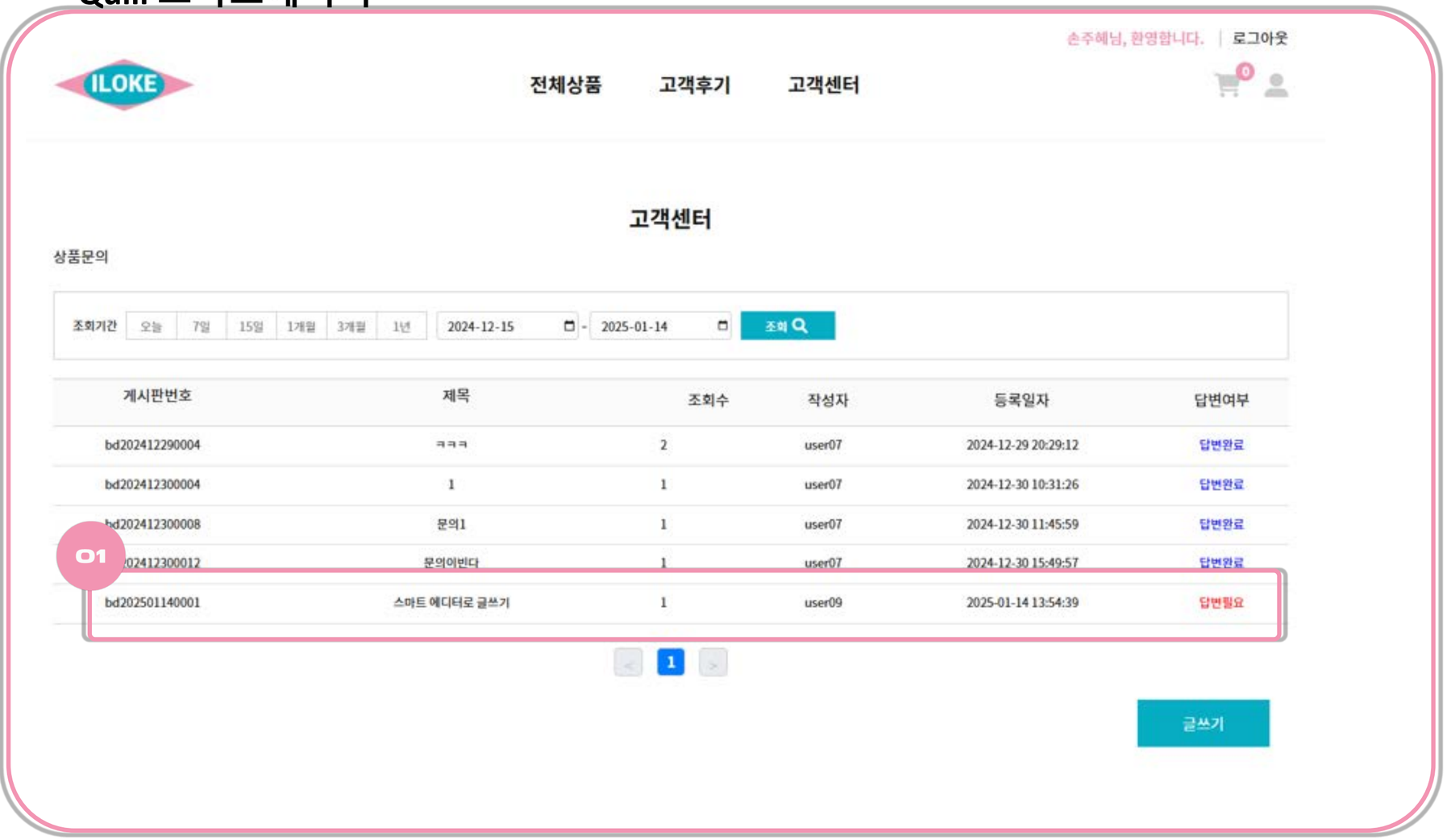
01

```
const quillRef = useRef(null);

useEffect(() => {
    const toolbarOptions = [
        ['bold', 'italic', 'underline', 'strike'],
        ['blockquote', 'code-block'],
        ['link', 'image', 'video', 'formula'],
        [{ 'header': 1 }, { 'header': 2 }],
        [{ 'list': 'ordered'}, { 'list': 'bullet' }, { 'list': 'check' }],
        [{ 'script': 'sub'}, { 'script': 'super' }],
        [{ 'indent': '-1'}, { 'indent': '+1' }],
        [{ 'direction': 'rtl' }],
        [{ 'size': ['small', false, 'large', 'huge'] }],
        [{ 'header': [1, 2, 3, 4, 5, 6, false] }],
        [{ 'color': [] }, { 'background': [] }],
        [{ 'font': [] }],
        [{ 'align': [] }],
        ['clean']
    ];

    if (quillRef.current) {
        const quill = new Quill(quillRef.current, {
            theme: 'snow',
            modules: {
                toolbar: toolbarOptions
            }
        });

        quill.on('text-change', () => {
            const html = quill.root.innerHTML; // Quill 에디터의 HTML 콘텐츠를 추출
            setContents(html);
        });
    }
}, []);
```

Description List

01 스마트 에디터로 글쓰기

Quill 스마트에디터



Description List

01 스마트 에디터로 쓴 글 저장

Quill 스마트에디터

01

```
const [content, setContent] = useState(contents || '');
  const [answerValue, setAnswerValue] = useState(comment || '');
  const [imageSrc, setImageSrc] = useState(null);
  const quillRef = useRef(null);

  const userId = sessionStorage.getItem('userId');

  useEffect(() => {
    if (code !== 'rv') {
      imageLoad();
    }

    if (quillRef.current) {
      const quill = new Quill(quillRef.current, {
        theme: 'snow',
        modules: {
          toolbar: [
            ['bold', 'italic', 'underline', 'strike'],
            ['blockquote', 'code-block'],
            ['link', 'image', 'video', 'formula'],
            [{ 'header': 1 }, { 'header': 2 }],
            [{ 'list': 'ordered'}, { 'list': 'bullet' }, { 'list': 'check' }],
            [{ 'script': 'sub'}, { 'script': 'super' }],
            [{ 'indent': '-1'}, { 'indent': '+1' }],
            [{ 'direction': 'rtl' }],
            [{ 'size': ['small', false, 'large', 'huge'] }],
            [{ 'header': [1, 2, 3, 4, 5, 6, false] }],
            [{ 'color': [] }, { 'background': [] }],
            [{ 'font': [] }],
            [{ 'align': [] }],
            ['clean']
          ]
        }
      });

      quill.on('text-change', () => {
        const html = quill.root.innerHTML;
        setContent(html);
      });

      quill.root.innerHTML = contents;
    }
  }, [code, contents]);
```

Description List

01 스마트 에디터로 쓴 글 저장 및 글 수정하기에도 코드 추가

주문 관리

Description List

01 고객센터 하위 메뉴 지점 선택

주문 관리

Description List

01 원하는 지점명을 검색할 수 있는 검색창

02 카카오 api를 통해 연동한 지도

03 기본적으로 데이터베이스에 저장되어있는 지점들을 나열

04 지점들의 목록을 페이지네이션을 통해 나열

주문 관리

Description List

01 검색어 입력 후 검색

02 해당 검색어를 포함하는 지점들이 목록에 표시됨

주문 관리

01

```
useEffect(() => {
  const script = document.createElement('script');
  script.src =
    '//dapi.kakao.com/v2/maps/sdk.js?appkey=e207779030ab91ab36823aa2027266a6&libraries=services&autoload=false';
  script.async = true;
  document.head.appendChild(script);
```

02

```
// 검색 필터 추가
if (search) {
  whereClause += ' AND (name LIKE ? OR address LIKE ?)';
  queryParams.push(`%${search}%`, `%${search}%`);
}
```

03

```
const dataQuery = `
  SELECT id, name, address, phone, hours, image, lat, lng
  FROM showrooms
  ${whereClause}

`;
```

Description List

01 카카오 api지도 연동

02 검색 필터 관련 코드

03 데이테베이스상의 지점 정보 가져오기

주문 관리

Description List

01 해당 지점을 선택하면 저장된 위치로 이동

02 각 위치값이 저장된 곳에 마커로 표시

03 마커 선택시 해당지점으로 이동 및 정보 표시
뒤로가기 버튼을 통해 원래 창으로 이동

주문 관리

02

```
/* 모달 창 */}
{showModal && (
    <div className={styles.modal_overlay}>
      <div className={styles.modal_content}>
        <button
          className={styles.close_button}
          onClick={() => setShowModal(false)} // 모달 닫기
        >
          닫기
        </button>
        <ShowRoomInfo
        onSelectShowroom={handleSelectShowroom}
        mode="modal" // 모달에서는 modal 모드로 설정
         />
      </div>
    </div>
  )}
```

Description List

01 주소 선택 버튼을 통해 각 지점을 찾아 해당 주소로 주문 진행가능 (직접 입력도 가능)

02 모달 상태일때는 예,아니오 버튼이 뜨도록 설정

주문 관리

Description List

01 버튼 클릭시 모달창 출력

01

주문 관리

Description List

01 모달창에서는 지점 클릭시
예,아니오 버튼을 통해 해당 지점으로 배송
할것인지 여부를 선택가능

주문 관리

Description List

01 찾은 지점에서 예 버튼을 클릭해 지점의 주소로 등록된 화면