

팀번호: 01

챗봇과 데이터 시각화를 이용한  
중고거래 서비스 Hidden 100

version  
v1.0

【P-실무프로젝트】

챗봇과 데이터 시각화를 이용한 중고거래 서비스

# 설 계 서

2019.12.2

가천대학교 컴퓨터공학과

- 팀장 ■ 한혜경(201736058)
- 팀원 ■ 이우석(201433860)
- 팀원 ■ 이다은(201632230)
- 팀원 ■ 서지연(201637615)

팀번호: 01

챗봇과 데이터 시각화를 이용한  
중고거래 서비스 Hidden 100version  
v1.0

【P-실무프로젝트】

## 문서 정보

본 문서는 챗봇과 데이터 시각화를 이용한 중고거래 서비스 개발을 위한 요구규격 및 설계서입니다.

버전	0.8
작성일	2019-12-02
상태	<input type="checkbox"/> 완료 <input checked="" type="checkbox"/> 진행 중 <input type="checkbox"/> 초안

버전	변경한 사람	변경한 날짜	버전업 변경(또는 추가)내용
0.1	이병문 교수님	11/27	설계서 기본 템플릿 제공
0.2	이우석	11/28	프로젝트 개요 및 환경구성 작성
0.3	이우석	11/28	서비스기능 정의 작성
0.4	한혜경	11/29	프로세스(기능) 설계 작성
0.4	이다은	11/30	데이터베이스 설계 작성
0.5	한혜경	12/1	완료된 부분 버전 통합
0.6	이다은	12/2	데이터베이스 설계 일부 수정
0.7	이다은	12/2	REST API 서버/클라이언트 구조 정의
0.8	한혜경	12/2	REST API 상세 설계
0.9	서지연	12/2	UI디자인
1.0	한혜경	12/2	설계서 작성 완료

# 목 차

<b>1. 프로젝트 개요</b>	
1.1 개발목표와 범위	4
1.2 개발일정/산출물	5
1.3 개발조직/역할	5
<b>2. 서비스기능 정의</b>	
2.1 서비스 개요	7
2.2 유스케이스 모델	9
2.3 요구기능 정의(사용자)	11
2.4 요구기능 정의(관리자)	14
<b>3. 프로세스(기능) 설계</b>	
3.1 시스템 아키텍쳐	17
3.2 사용자기능 설계(Sequence Diagram)	18
3.3 관리자기능 설계(Sequence Diagram)	41
<b>4. 화면(UI)설계</b>	
4.1 스토리보드 설계	58
4.2 초기화면(로그인 전화면)설계	59
4.3 사용자UI 설계	60
4.4 관리자UI 설계	97
<b>5. REST API 인터페이스 설계</b>	
5.1 서버/클라이언트 구조	114
5.2 인터페이스 정의	115
5.3 상세 REST API 설계	117
<b>6. 데이터베이스 설계</b>	
6.1 ERD	152
6.2 논리적설계(테이블명세서)	153
6.3 물리적설계(SQL스크립트)	157
<b>7. 환경구성</b>	
7.1 개발환경 및 운영환경	164
7.2 소스디렉터리 구조	165

## 1. 프로젝트 개요

### 1.1. 개발목표와 범위

#### ■ 개발목표

본 설계서는 PC환경에서 중고거래를 위한 쇼핑몰 웹 서비스를 개발하기 위한 설계서이다. 주요 사용자는 중고물을 거래하고 싶어하는 개인들이며, 챗봇(카카오톡 서비스)을 통해서 접속하고 사용하기 편리하도록 고안되는 것을 전제로 한다. 쇼핑몰 웹 서비스를 효과적으로 관리하기 위해 관리기능도 구성하였으며 최대한 편리성과 수익성을 추구할 수 있도록 개발한다. 관리자는 웹기반 시각화된 UI를 통해 쇼핑몰을 관리할 수 있도록 한다.

아래 그림1-1은 본 서비스의 전체적인 구성을 나타내며, 중고 물품의 구매와 판매를 희망하는 사용자가 본 서비스를 사용한다.

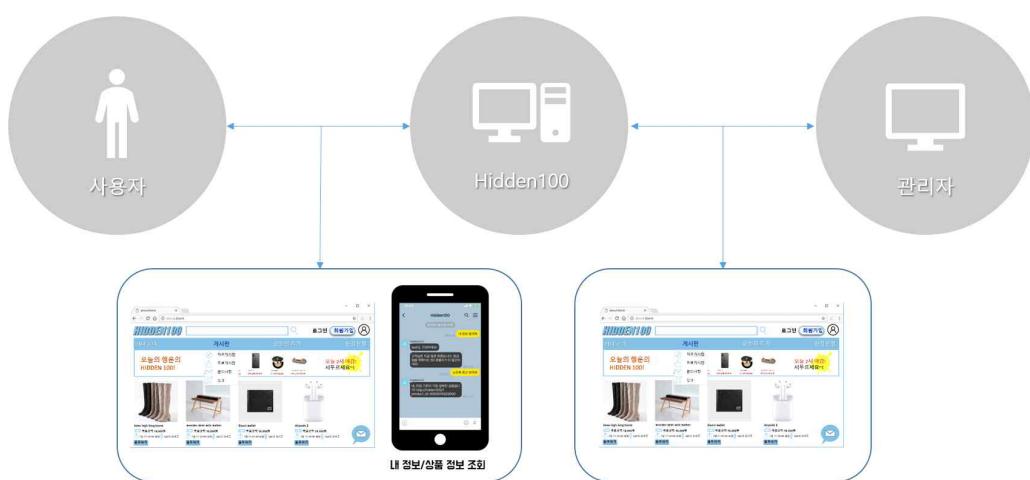


그림 1.1 Hidden100 웹서비스 설계범위

## ■ 개발필요성

현재 가장 보편적인 중고 거래 사이트들은 회원 관리와 거래 방식의 구조적 문제로 인하여 사기 행위를 하여도 별다른 제재를 가할 수단이 없다. 오픈마켓과 비슷한 형식으로 쇼핑몰이 제재하고 관리하는 방식도 중고거래의 소량상품, 다수의 판매자 특성 때문에 비효율적이다. 다수의 판매자와 구매자를 유도해야 하는 중고거래에서는 오픈마켓처럼 판매자 등록 절차가 있는 경우 불리하다. 그리고 기존 방식은 구매자와 판매자 간에 거래 과정에서 판매자가 제시한 금액과 구매자가 원하는 가격에 차이가 있어 이를 조정하는데 시간, 감정 등이 소요 되는 경우가 많다. 따라서 이러한 기존 방식을 개선할 필요가 있다.

대부분의 중고거래 쇼핑몰은 이 해결책으로 한 종류의 상품에 집중하여 판매와 구매를 모두 대행하는 경우가 많으나 수동으로 해결하기에는 한계가 있다. 또 앞서 말한 방법으로는 구매가와 판매가에서 차익을 낸다 하더라도 구매 심리를 이용하기 위해서는 한계가 있고, 인력이 소모되기 때문에 쇼핑몰에서는 인건비를 감수하고 큰 이득을 보기 어렵다. 따라서 관리자가 하나하나 확인하지 않아도 중고거래가 가능하며 거래사기를 방지하기 위해 자체 재화를 사용하지만, 자체 재화의 필요성을 사용자가 납득 가능한 서비스가 필요하다.

본 서비스는 그러한 과정을 경매, 투자, 추첨 방식을 혼합하여 거래 방식 단순화, 구매자의 흥미 유발, 판매자의 수익 최대화를 통해 판매자와 구매자 모두 만족하는 결과를 만들어낸다. 판매자는 예상 가격보다 높은 값을 받을 수 있으며, 구매자는 경품추첨과 비슷한 방식으로 운이 좋다면 아주 적은 금액에 좋은 중고 물건을 구매할 수 있다. 현재 중고 거래 방식에서는 거래가 성사되기까지 판매자와 구매자 모두 중고 거래 사이트에 오래 머무를 수밖에 없다. 본 서비스만의 독특한 거래 방식과 챗봇 서비스는 구매자와 판매자 모두를 모니터 앞에서 해방시키며 언제 어디에서나 거래 과정에 참여 할 수 있는 동적인 서비스를 제공한다.

## 1.2. 개발일정/산출물

본 설계서는 아래 그림 1.3과 같이 개발일정에 따라 진행하며, 각 단계별로 산출물을 개발한다.

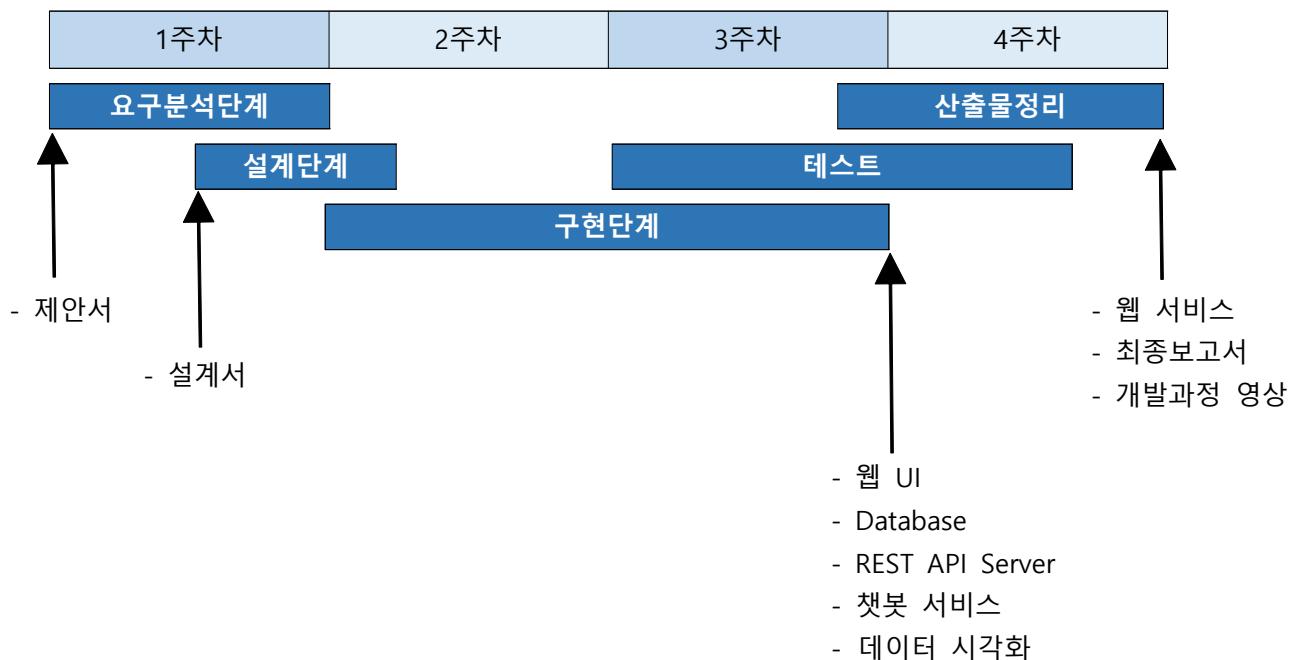


그림 1.2 개발일정 및 산출물

## 1.3. 개발조직/역할

역할	이름	역할(개발기능)	비고
팀장	한혜경	사용자 인증, 챗봇 사용자 기능 관리자 인증, 수익관리	
팀원	이우석	사용자 게시판, 챗봇 관리자 기능 회원 관리	
팀원	이다은	중고거래(판매자), 사용자 설정 데이터 시각화	
팀원	서지연	중고거래(구매자), 관리자 설정 중고거래(관리자)	

## 2. 서비스기능 정의

### 2.1. 서비스 개요

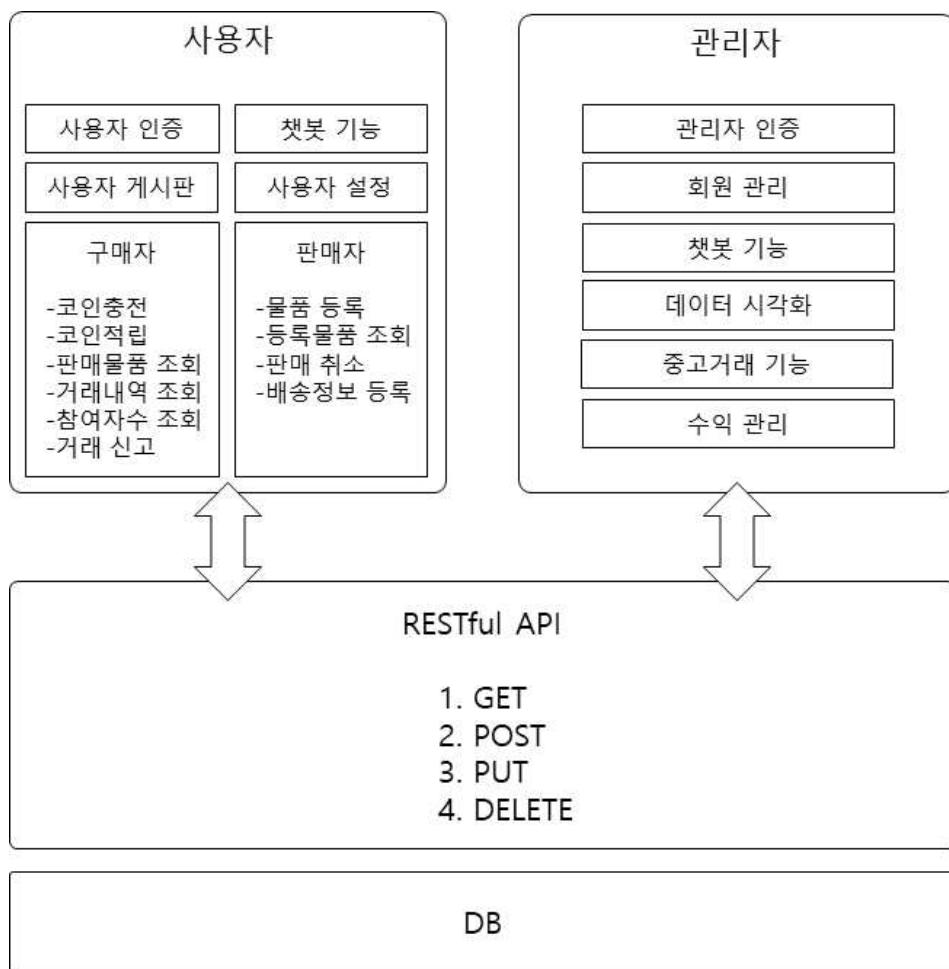


그림 2.1 Hidden100 서비스 모델 구성도

- ① Hidden 100은 중고거래를 지원하는 웹쇼핑몰이다. 사용자는 중고거래 판매자와 구매자가 될 수 있고 관리자는 거래 수수료로 이득을 얻는다. 판매자가 선정한 가격만큼 구매자들이 소액 적립을 통해 도달하면 추첨하여 당첨자에게 중고 물품을 전달하는 방식이다. 중고거래에 자체 재화를 사용하여 사기를 방지할 수 있고 사용자에게 코인을 충전할 수 있는 기능을 지원한다. 관리자는 수익을 조회하고 회원을 관리할 수 있다. 사용자와 관리자는 RESTful API를 이용해서 서버에 저장된 데이터를 확인하고 거래에 참여하거나 정보를 수정·가입하는 등의 처리를 Web UI를 통해 수행할 수 있다.
- ② 사용자는 판매물을 조회할 수 있다. Hidden100의 비밀 소액적립형 추첨경매 서비

스로 현재 적립금을 확인할 수 없고 참여자만 확인할 수 있다. 그러나 자신이 등록한 상품의 경우 적립금과 참여자 수를 모두 확인할 수 있다. 또 사용자가 상품을 조회할 때 챗봇을 통해 편하게 조회할 수 있으며, 상품 검색이 가능하다. 일부 조회를 제외한 대부분의 기능은 회원가입과 로그인으로 사용자 인증을 마친 다음에 이용 가능하다.

- ③ 관리자가 상품을 조회하는 경우 관리자는 상품 등록에 대한 모든 정보를 확인할 수 있다. 현재 적립금과 가격 등록한 사용자까지 정보를 파악할 수 있으며 쇼핑몰 안에서 진행중인 전체 중고거래에 대한 확인이 가능하다. 관리자는 누적 매출과 수수료를 포함한 거래 정보에 대해 확인 가능하다. 또 챗봇 로그에서 사용자가 검색한 결과에 대한 합산을 조회하여 쇼핑몰의 거래 흐름과 사용자의 관심에 대해 확인할 수 있다.
- ④ 사용자는 코인을 충전하여 상품에 적립하는 방식으로 거래에 참여한다. 따라서 코인의 현재 적립액은 관리자와 판매자만 확인할 수 있다. 마감시간까지 적립금이 판매자가 설정한 목표금액에 도달하면 적립한 사람 중에 당첨자를 뽑아 중고 물건을 보낸다. 적립금이 높을수록 당첨 확률이 높아지며 배송은 판매자가 하기 때문에 배송정보를 입력하는 기능을 지원한다. 구매자가 배송수령을 확인하면 거래 대금이 판매자에게 전달되며 배송으로 중고거래 사기를 할 경우 구매자는 거래신고를 할 수 있다.
- ⑤ 로그인한 사용자는 게시판의 게시글을 조회할 수 있다. 게시글을 통해 사용자들은 커뮤니티와 후기를 공유할 수 있고 신고와 문의도 게시글을 통해 처리한다. 컴퓨터로 완전히 처리할 수 없는 거래사기 같은 경우 구매자가 신고를 하면 관리자가 확인하여 구매자에게 제재를 가하거나 판매 적립금을 보내지 않고 환불하는 방식으로 사기 피해를 최소화할 수 있다.
- ⑥ 관리자는 회원의 정보를 조회하고 회원 정보를 수정할 수 있다. 회원 계정을 정지상태로 만들면 사용자는 로그인이 불가능하며, 회원을 탈퇴시키거나 회원 정보를 수정하는 기능을 사용할 수 있다. 관리자도 일반 회원과 마찬가지로 회원가입과 로그인으로 관리자인증을 마친 다음에 관리자 기능을 이용할 수 있다.

## 2.2. 유스케이스 모델

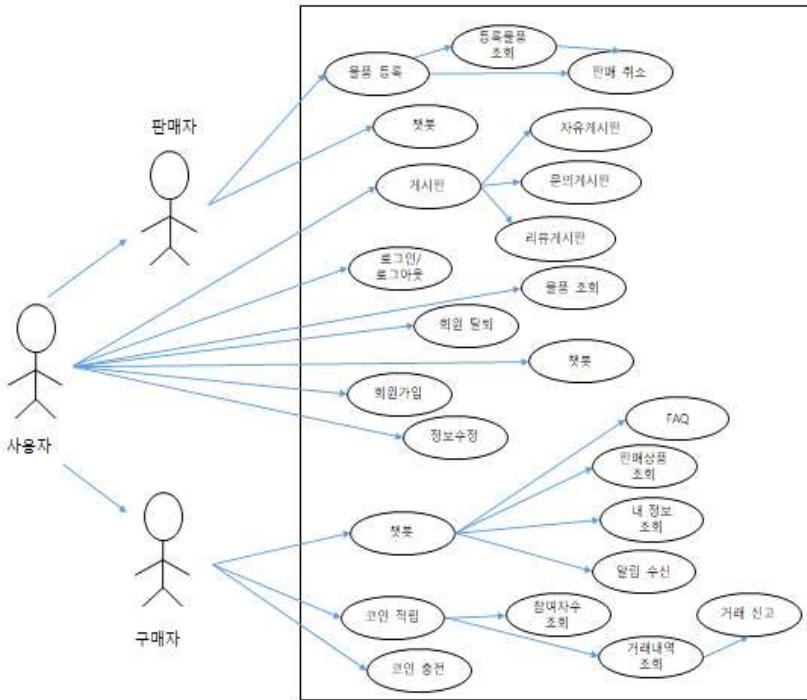


그림 2.2 사용자용 유스케이스 디아어그램

사용자는 공통적으로 게시판, 로그인, 로그아웃, 회원 탈퇴, 챗봇, 회원 가입, 정보 수정 페이지에 접근할 수 있다. 사용자는 구매자와 판매자로 나뉘게 된다. 하지만 구매자는 동시에 판매자가 될 수 있다. 판매자의 경우에도 마찬가지이다. 구매자는 챗봇을 통해 FAQ, 판매 상품 조회, 내 정보 조회, 알림 수신을 할 수 있다.

구매자는 구매하고자 하는 상품에 코인을 적립하고 나면 그 거래에 참여하고 있는 참여자 수 등 거래에 관련 된 정보 조회 페이지에 접근할 수 있다. 구매자는 자신이 코인을 얼마나 사용했는지 등을 조회 할 수 있는 거래 내역 조회 페이지에 접근 할 수 있다. 거래 내역 조회 페이지를 통해 거래 신고 페이지에 접근 할 수 있다.

판매자는 물품 등록 페이지에 접근 할 수 있다. 물품을 등록하고 나면 자신이 등록한 물품의 목록을 조회할 수 있는 페이지에 접근 할 수 있다. 등록한 물품을 판매 중지 할 수 있는 페이지로 접근이 가능 하다.

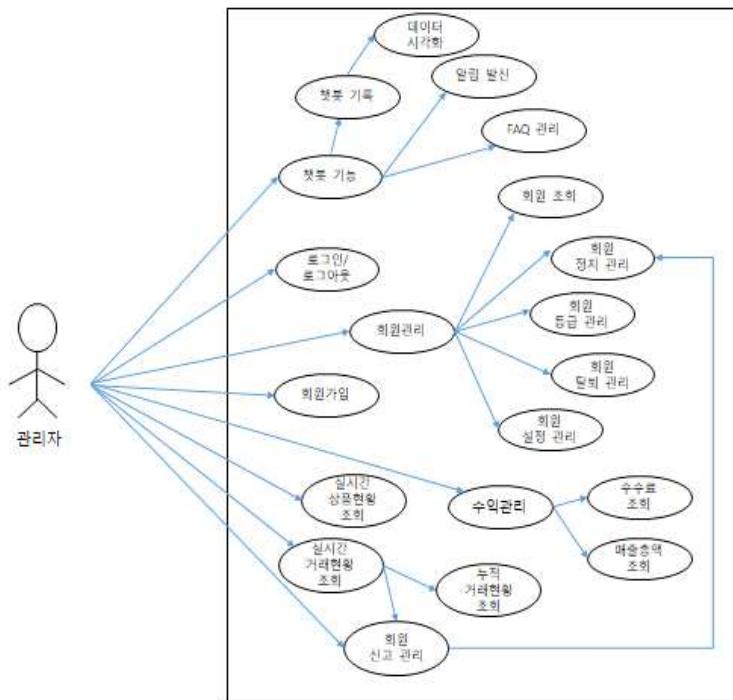


그림 2.3 관리자용 유스케이스 다이어그램

관리자는 로그인, 로그아웃, 회원가입 페이지에 접근 할 수 있다. 회원 관리 페이지를 통해 회원조회, 회원 정지 관리, 회원 탈퇴 관리, 회원 설정 관리 페이지로 이동 할 수 있다. 챗봇의 로그를 조회, 조회된 데이터를 시각화 해주는 페이지에 접근할 수 있다. 챗봇을 통해 사용자에게 알람을 발신하거나 챗봇에 새로운 정보를 등록할 수 있는 페이지에 접근할 수 있다. 관리자는 수익관리 페이지에 접근할 수 있으며, 수수료 수익 조회, 매출 총액 조회 페이지에 접근 할 수 있다.

### 2.3. 요구기능 정의 (사용자)

번호	주요기능명	상세기능명	개발담당
1)	사용자인증	회원가입	한혜경
2)		로그인	한혜경
3)		로그아웃	한혜경
4)	챗봇 기능	알림 수신	이다은
5)		내 정보 조회	이다은
6)		판매상품 조회	이다은
7)		FAQ	이다은
8)	사용자게시판	게시글 등록	이우석
9)		게시글 조회	이우석
10)		게시글 수정	이우석
11)		게시글 삭제	이우석
12)	중고거래(판매자)	물품등록	한혜경
13)		등록 물품 조회	서지연
14)		판매취소	한혜경
15)		배송정보 등록	이다은
16)	중고거래(구매자)	코인충전	이다은
17)		코인적립	이다은
18)		판매물품조회	서지연
19)		거래내역조회	서지연
20)		참여자수조회	서지연
21)		거래신고	서지연
22)	사용자설정	정보수정	이우석
23)		회원탈퇴	이우석

#### 1) 회원가입

- 본 서비스를 이용하기 위해 사용자 정보를 등록한다.
- 품을 통해 사용자로부터 아이디, 비밀번호, 이름, 주소를 받는다. 받은 정보를 DB에 저장한다.

#### 2) 로그인

- 사용자로부터 아이디와 비밀번호를 품을 통해 받아 DB에 저장되어 있는 데이터와 비교한다.
- 비교 후 로그인 성공 화면과 실패 화면을 경우에 따라 보여준다.

#### 3) 로그아웃

- 회원 정보, 구매 정보, 판매 정보 등을 DB에 저장하고 로그아웃한다.

## 4) 챗봇 알림 수신

- 당첨 여부, 판매 취소, 환불 등 사용자와 관계 된 정보를 알림받는다.

## 5) 챗봇 내 정보 조회

- 사용자의 등급, 적립한 금액, 물건, 보유 포인트 등을 챗봇에 물어본다.
- 챗봇API가 입력된 문자열을 처리 후 해당 조건에 맞는 쿼리로 DB를 조회한다.
- 조회 된 결과를 사용자에게 보내준다.

## 6) 챗봇 판매 상품 조회

- 사용자가 구매하고자 하는 물건을 챗봇에 물어본다.
- 챗봇API가 입력된 문자열을 처리 후 해당 조건에 맞는 쿼리로 DB를 조회한다.
- 조회 된 결과를 사용자에게 보내준다.

## 7) FAQ

- 사용자가 챗봇에 질문을 한다.
- 챗봇API가 입력된 문자열을 처리 한 후 질문 내용이 FAQ에 등록되어 있는 질문이면 그에 맞는 쿼리로 DB를 조회한다.
- 조회된 결과를 사용자에게 보내준다.

## 8) 게시글 등록

- 게시판에 글을 작성한다..
- 게시글을 DB에 저장한다.

## 9) 게시글 조회

- 사용자가 작성한 글을 조회한다.
- 다른 사용자가 작성한 글을 조회한다.
- 품에 검색어를 입력하여 검색어를 포함한 쿼리문으로 DB를 조회 후 조건에 부합하는 게시글을 조회한다.

## 10) 게시글 수정

- 사용자가 작성한 글을 수정한다.
- 수정된 내용을 DB에 저장한다.

## 11) 게시글 삭제

- 사용자가 작성한 글을 삭제한다.
- DB에서도 삭제한다.

## 12) 판매자 물품 등록

- 물건의 종류, 상품 명, 목표 금액을 품을 통해 입력한다.
- 입력 된 내용을 DB에 저장한다.

## 13) 등록 물품 조회

- 자신이 등록한 물건을 DB에서 조회한다.

## 14) 판매 취소

- 자신이 등록한 물품의 판매를 취소한다.

- DB에 저장되어있는 등록 물품을 삭제한다.

#### 15) 배송정보 등록

- 구매자에게 물건을 배송한 후 이를 증명하기 위해 품에 배송장 정보를 입력한다.
- 입력된 정보를 DB에 저장 후 운송 업체 DB와 비교한다.

#### 16) 코인 충전

- 쇼핑몰 자체 전자 화폐를 충전한다.
- 변경된 전자화폐 값으로 DB에 저장된 값을 수정한다.

#### 17) 코인 적립

- 구매자가 구매하고자 하는 물건에 코인을 적립한다.
- 물건에 대한 적립금이 바뀌면 그 바뀐 값을 DB에 저장한다.

#### 18) 판매 물품 조회

- 구매자가 구매하고자 하는 물건의 조건을 품에 입력한다.
- 입력된 조건에 맞는 쿼리로 DB에서 판매 등록된 물건을 조회한다.
- 조회 결과를 구매자에게 보여준다.

#### 19) 거래내역 조회

- 구매자가 지금까지 구매한 물건들을 DB에서 조회한다.
- 조회된 결과값을 보여준다.

#### 20) 참여자 수 조회

- 구매자가 현재 참여하고 있는 판매건에 몇 명이 참여하고 있는지 DB에서 조회한다.
- 조회된 결과값을 보여준다.

#### 21) 거래 신고

- 자신이 구매한 물건이 제대로 배송되지 않았거나 본래 거래한 물건이 아닐 경우 신고 게시판에 글을 남긴다.
- 신고글을 DB에 저장한다.

#### 22) 정보 수정

- 사용자의 정보를 품을 통해 수정한다.
- 수정된 값으로 DB에 저장되어있는 값을 수정한다.

#### 23) 회원 탈퇴

- 회원 탈퇴 요청을 한다.
- 사용자의 정보를 DB에서 지운다.

## 2.4. 요구기능 정의 (관리자)

번호	주요기능명	상세기능명	개발담당
1)	관리자인증	회원가입	한혜경
2)		로그인	한혜경
3)		로그아웃	한혜경
4)	회원 관리	회원 조회	서지연
5)		회원등급관리	서지연
6)		회원설정관리	서지연
7)		회원정지관리	서지연
8)		회원탈퇴관리	서지연
9)	챗봇기능	알림 발신	이다은
10)		FAQ 관리	이다은
11)	시각화	챗봇 로그	이다은
12)	중고거래기능	실시간 상품등록 현황조회	이우석
13)		실시간 거래 현황 조회	이우석
14)		누적 거래 현황 조회	이우석
15)		회원신고관리	이우석
16)	수익관리	수수료조회	한혜경
17)		매출총액조회	한혜경

### 1) 회원가입

- 신입 관리자를 등록한다.
- 폼을 통해 관리자 정보를 입력받는다.
- 입력된 정보를 DB에 저장한다.

### 2) 로그인

- 아이디와 비밀번호를 폼에 입력한다.
- 입력받은 아이디와 비밀번호를 DB에 있는 정보와 비교 한다.
- 일치하는 경우 로그인 된 상태의 메인 페이지를, 불일치 하는 경우 알림 메시지를 띠어준다.

### 3) 로그아웃

- 로그인 해있는 동안 변경된 상태, 정보, 값을 DB에 저장하고 로그아웃한다.

### 4) 회원 조회

- 관리자는 회원 가입할 때 입력한 DB에 저장되어 있는 모든 회원들의 정보를 조회할 수 있다.

### 5) 회원등급관리

- 관리자는 DB에 저장되어 있는 회원의 등급을 조회한다.
- 회원의 등급을 수정하고 다시 DB에 저장 할 수 있다.

## 6) 회원 설정 관리

- 관리자는 DB에 저장되어 있는 회원들의 정보를 조회할 수 있다.
- 관리자는 DB에 저장되어 있는 회원들의 정보를 수정하고 DB에 다시 저장 할 수 있다.

## 7) 회원 정지 관리

- 관리자는 특정 회원을 쿼리의 조건을 통하여 DB로부터 조회할 수 있다.
- 관리자는 회원의 계정을 정지 시킬 수 있다.

## 8) 회원 탈퇴 관리

- 관리자는 회원 탈퇴 신청한 회원의 정보를 DB에서 삭제한다.

## 9) 알림 발신

- 관리자는 챗봇을 통해 사용자에게 판매 완료, 구매 완료 등의 상태를 전달 할 수 있다.

## 10) FAQ 관리

- 관리자는 FAQ를 DB에 추가하거나 삭제할 수 있다.
- 사용자가 챗봇에 질문을 한다.
- 챗봇은 문자열 처리 후 조건에 해당되는 FAQ를 쿼리를 통해 DB에서 조회할 수 있다.
- 결과를 사용자에게 알려준다.

## 11) 시각화

- 사용자가 챗봇에 입력한 질문 내용을 텍스트 시각화한다.

## 12) 실시간 상품 등록 현황 조회

- 관리자는 현재 서비스에 상품이 얼마나 등록되어 있는지, 어떤 상품들이 등록 되어 있는지 등을 여러 조건의 쿼리로 DB로부터 조회 할 수 있다.
- 조회 된 데이터를 그래프 등 시각처리 할 수 있다.

## 13) 실시간 거래 현황 조회

- 관리자는 현재 거래가 성사된 상품들을 여러 조건의 쿼리로 DB로부터 조회 할 수 있다.
- 데이터를 시각화한다.

## 14) 누적 거래 현황 조회

- 관리자는 현재까지 거래가 성사되어 배송까지 완료 된 상품들을 여러 조건의 쿼리로 DB로부터 조회 할 수 있다.
- 데이터를 시각화한다.

## 15) 회원 신고 관리

- 사용자가 회원 신고 게시판의 입력 폼에 신고 내용을 입력한다.
- 입력한 데이터를 DB에 저장한다.

- 관리자는 DB에 저장되어 있는 데이터를 조회한다.
- 데이터를 바탕으로 회원의 정보를 DB로부터 조회한다.
- 회원의 계정 상태를 수정하여 DB에 저장한다.
- 신고 내용에 따라 조치를 취한 내용을 DB에 저장한다.
- 사용자는 신고에 대한 조치를 게시판의 댓글 형태로 DB에서 조회할 수 있다.

#### 16) 수수료 조회

- 관리자는 수수료로 얼마의 수익을 냈는지 DB로부터 조회 할 수 있다.

#### 17) 매출총액관리

- 관리자는 매출과 순이익이 얼마인지를 DB로부터 조회 할 수 있다.
- 조회한 데이터를 시각화하여 보기 좋게 한다.

### 3. 프로세스(기능) 설계

#### 3.1. 시스템 아키텍쳐

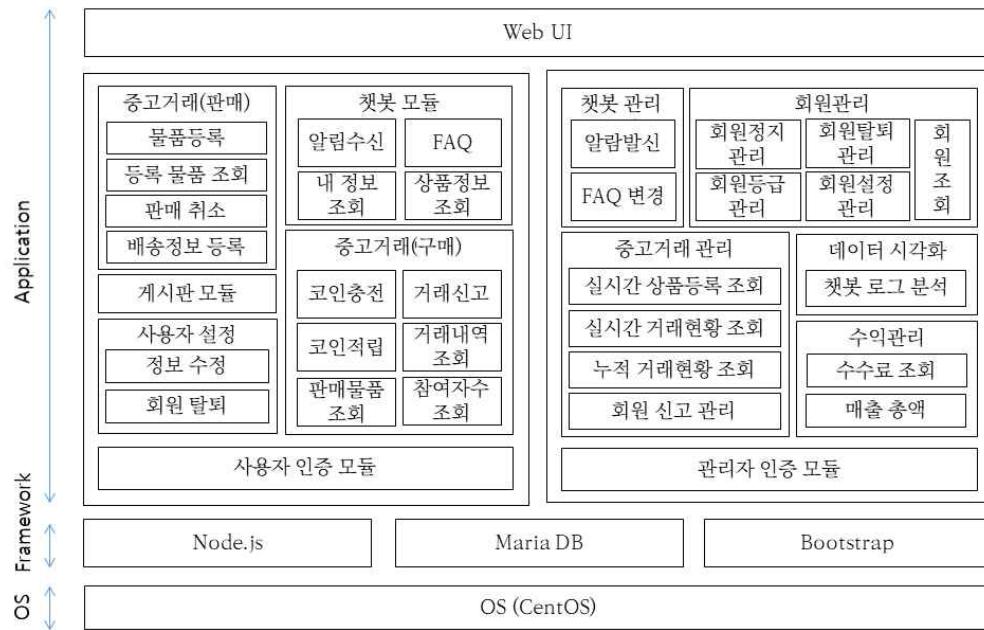


그림 3.1 시스템 아키텍쳐

소액적립형 경매 서비스 Hidden100은 그림 3.1과 같이 시스템을 사용자 기능과 관리자 기능으로 구분할 수 있다. 웹서버는 CentOS 기반으로 NPM(Node Package Manager), Node.js, MariaDB, Bootstrap 그리고 외부모듈을 이용하여 구현한다. NPM 을 이용하여 외부모듈을 관리한다. 사용자와 관리자는 Web UI를 이용하여 서비스를 사용할 수 있다.

사용자 기능과 관리자 기능은 각각 인증을 마친 후에 사용할 수 있지만, 물품 조회 기능과 회원가입은 사용자 인증이 없어도 이용 가능하다. 사용자 기능으로 사용자 인증, 중고거래 구매와 판매, 사용자 설정, 게시판, 챗봇 기능을 지원한다. 관리자 기능으로 관리자 인증, 중고거래 관리, 회원관리, 수익관리, 데이터 시각화, 챗봇 관리 기능을 제공한다. Hidden100의 서비스 특성으로 사용자는 응모 가격 등을 확인할 수 없으나 관리자는 모두 확인할 수 있기 때문에 별도의 기능으로 모듈화한다.

### 3.2. 사용자기능 설계 (Sequence Diagram)

#### 1) 사용자 인증 - 회원가입

사용자는 사용자 인증 후 Hidden100 서비스를 이용할 수 있기 때문에 회원가입이 필요하다. 회원가입 기능을 구현하기 위해 아래와 같이 설계한다.

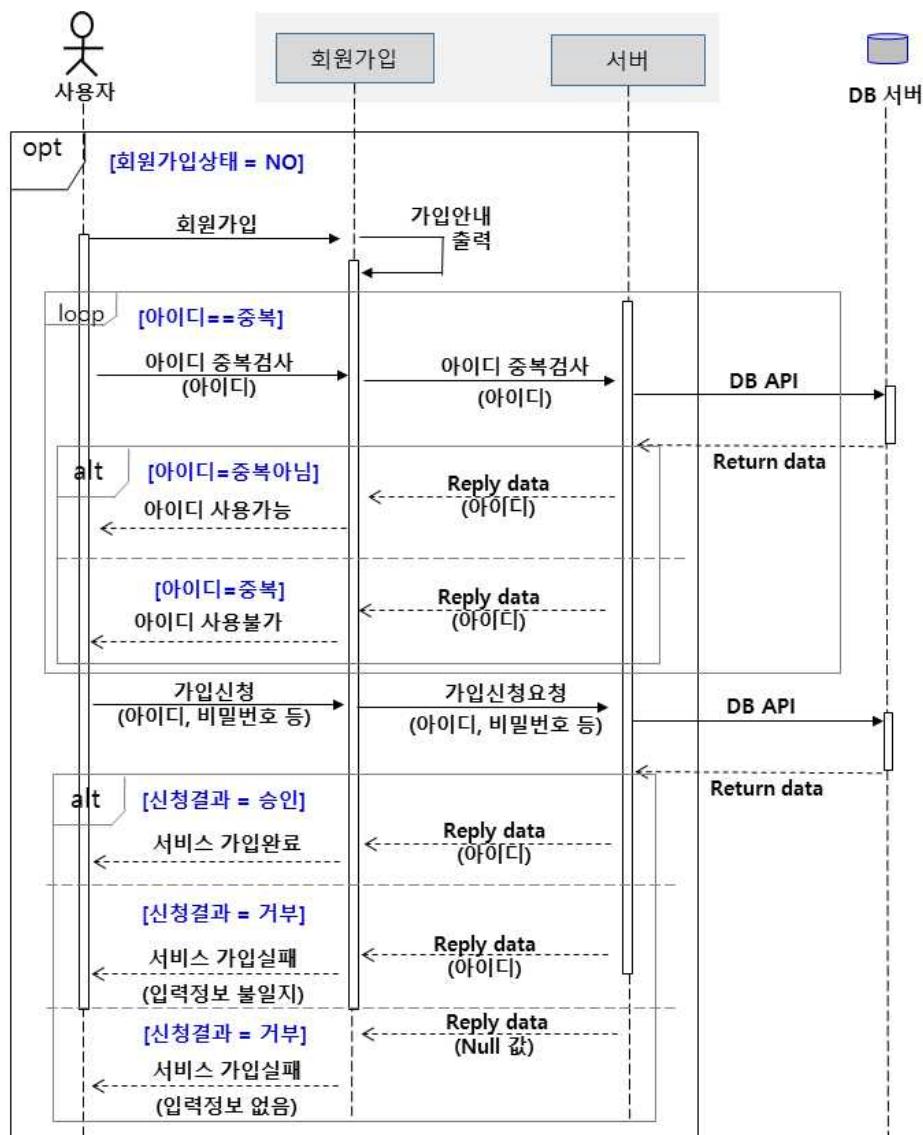


그림 3.2 회원가입의 Sequence Diagram

그림 3.2를 보면 사용자가 회원가입 하지 않은 경우에 회원가입을 진행한다. 사용자는 아이디를 중복검사한다. 아이디의 경우 DB에서 키로 사용되기 때문에 중복을 허용하지 않으며 중복이 아닌 경우에만 다음 단계로 넘어갈 수 있다. 아이디가 중복

이 아닐 때까지 검사를 반복한다. 사용자는 중복검사를 마친 후에 회원가입 정보를 입력하여 회원가입을 진행할 수 있다. 만약 입력한 값에 오류가 있으면 회원가입신청이 거부된다. 입력 정보가 없는 경우에도 회원가입 신청이 거부된다.

## 2) 사용자 인증 - 로그인기능

Hidden100의 사용자 기능을 이용하기 위해서는 아이디와 패스워드를 이용하여 로그인 기능이 필요하다. 로그인을 구현하기 위해 아래와 같이 설계한다.

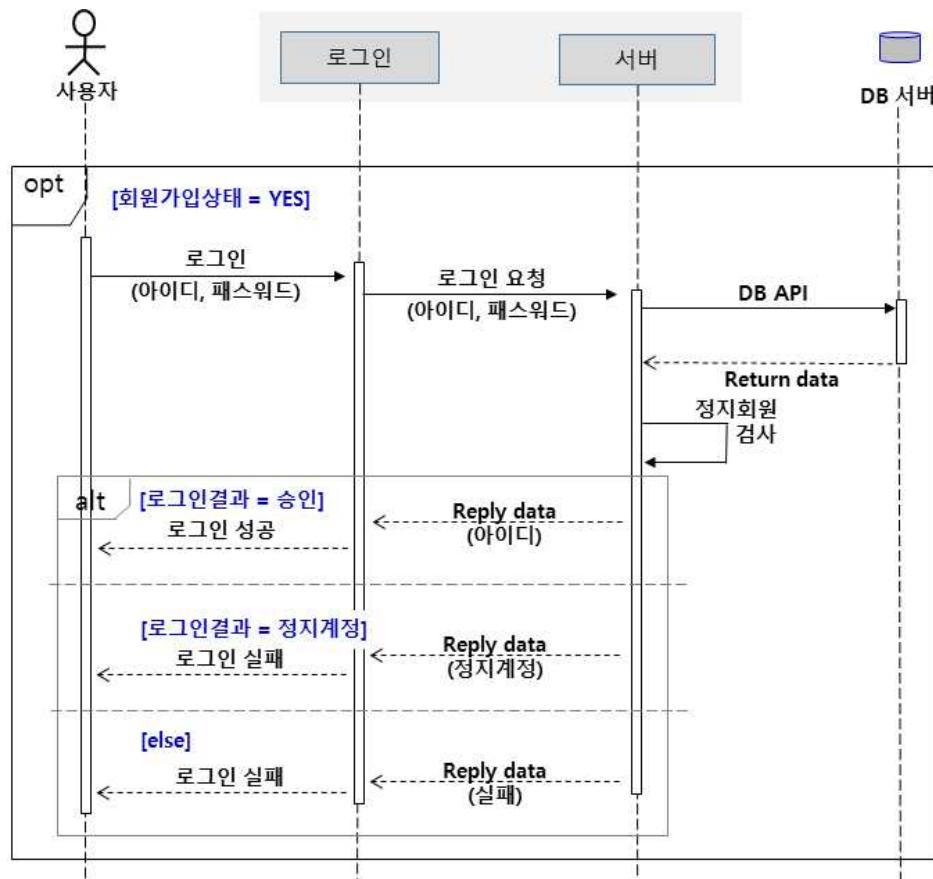


그림 3.3 로그인의 Sequence Diagram

그림 3.3를 보면 회원가입한 경우에 로그인으로 사용자 인증이 가능하다. 사용자는 아이디와 패스워드를 입력하고 로그인 버튼을 눌러 로그인을 요청한다. 서버에서는 로그인 요청을 받아 DB에서 아이디를 키로 회원정보를 받아와 입력받은 정보와 비교한다. 아이디와 패스워드가 DB에 저장된 회원정보와 일치하면 로그인에 성공한다. 로그인하고 나서 다른 Hidden100 사용자 서비스를 이용할 수 있다. 패스워드가 다르거나 정지계정인 경우 로그인에 실패한다. 로그인에 실패한 경우 실패 메시지와 실패 사유를 안내한다.

### 3) 사용자 인증 - 로그아웃 기능

사용자가 서비스 이용을 종료하거나 다른 계정으로 전환이 필요한 순간에 로그아웃 기능을 수행한다. 로그아웃으로 사용자는 세션에 저장된 로그인 상태를 종료할 수 있다.

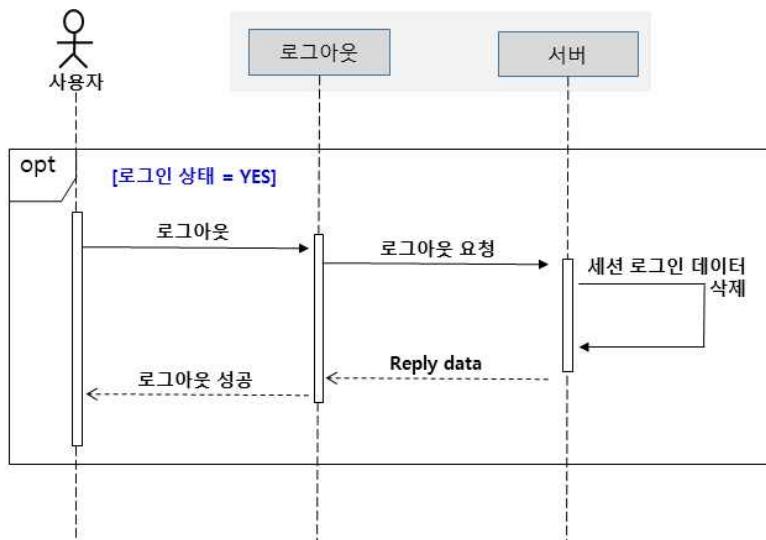


그림 3.4 로그아웃의 Sequence Diagram

그림 3.4를 보면 사용자는 로그아웃을 수행하기 위해 로그아웃 버튼을 누른다. 로그아웃 기능은 서버로 가서 현재 세션에 저장된 사용자의 로그인 데이터를 삭제한다. 서버에서 로그아웃을 처리한 후에 사용자에게 로그아웃된 화면으로 전송하여 보여준다.

#### 4) 사용자 설정 - 정보수정 기능

사용자가 본인의 정보를 변경하기 위해 정보수정 기능을 사용한다. 사용자는 로그인한 다음 회원 본인의 데이터를 수정할 수 있으며 아래와 같이 설계한다.

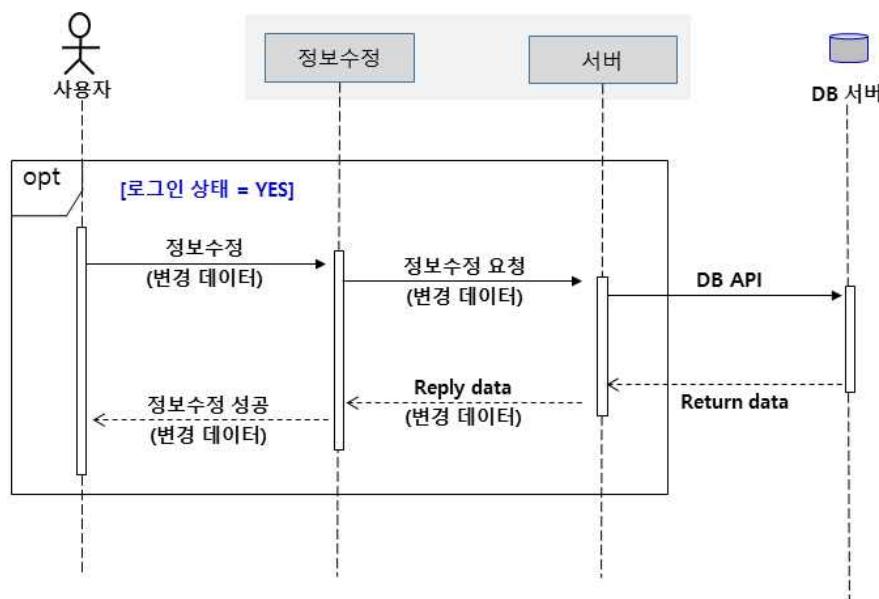


그림 3.5 정보수정의 Sequence Diagram

그림 3.5를 보면 사용자는 웹화면을 통해 자신의 정보를 수정할 수 있다. 변경하고 싶은 정보를 수정하고 정보수정 버튼을 누르면 입력한 데이터가 서버로 넘어가 DB에 저장된다. 변경하면 사용자의 화면에도 변경 데이터로 출력한다.

## 5) 사용자 설정 - 회원탈퇴 기능

사용자가 자신의 계정을 삭제하기 위해 회원탈퇴 기능을 사용한다. 사용자는 로그인 한 다음 회원탈퇴를 진행할 수 있으며 다음과 같이 설계한다.

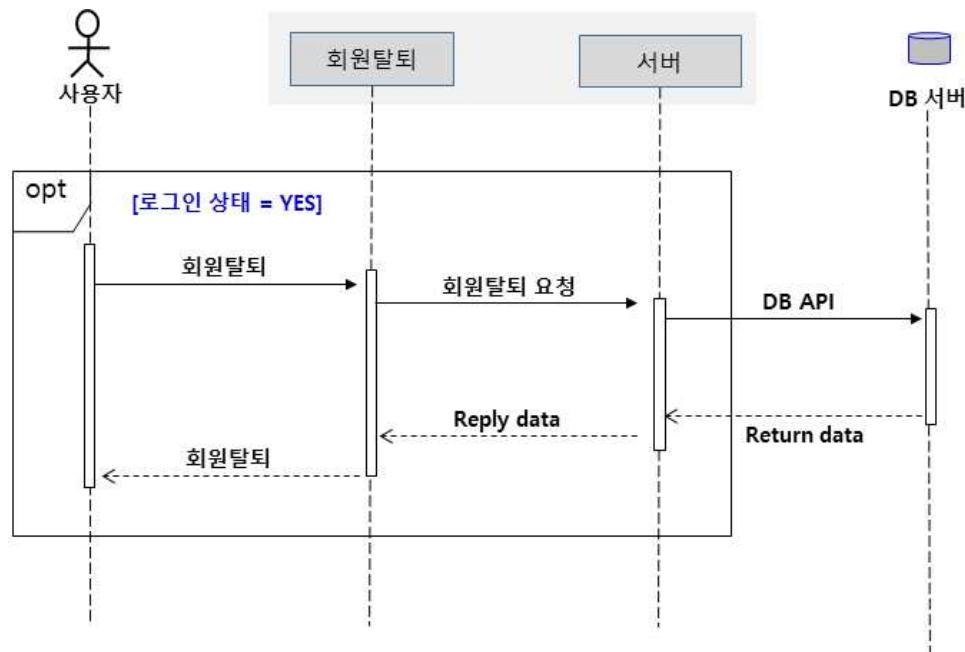


그림 3.6 회원탈퇴의 Sequence Diagram

그림 3.6을 보면 사용자는 로그인한 상태에서 회원탈퇴를 진행한다. 회원탈퇴 버튼을 누르면 회원탈퇴 요청이 서버로 올라가고 DB 서버에서 사용자를 삭제한다. 탈퇴된 회원은 서비스를 사용할 수 없고 탈퇴된 결과를 사용자에게 보낸다.

## 6) 중고거래(판매) - 물품등록

중고 물건을 판매하고 싶은 사용자는 물품등록을 이용해 판매할 수 있다. Hidden100에서는 중고거래 물품 등록을 위해 다음과 같이 설계한다.

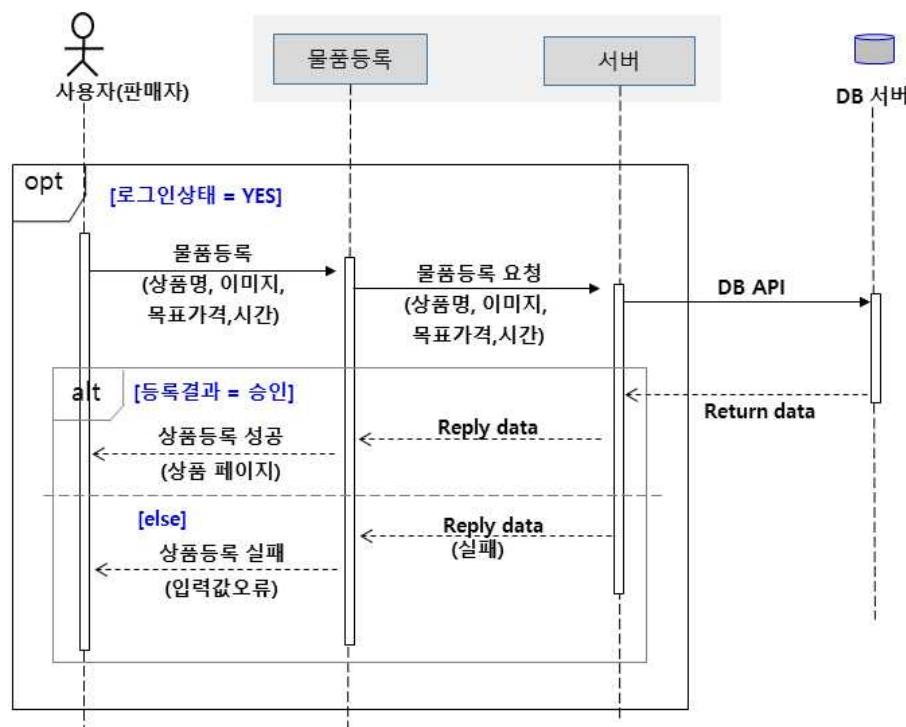


그림 3.7 물품등록의 Sequence Diagram

Hidden100은 사용자가 판매자와 구매자의 역할을 수행하는 중고거래 사이트이며 사용자가 로그인만 하면 상품을 판매할 수 있다. 그림 3.7을 보면 사용자는 로그인한 상태에서 물품등록을 할 수 있다. 물품등록을 위해 판매할 물품의 상품명, 이미지, 목표가격, 시간 등을 설정한다. 사용자가 Web UI를 통해 입력한 정보는 서버로 전달하고 DB에 저장된다. 입력 값이 비어있거나 오류가 있는 경우에는 상품등록이 거절된다. 물품 등록에 성공하면 중고 물품은 판매자가 설정한 시간 동안 적립을 시작한다. 저장 결과에 따라 사용자에게 등록결과를 안내한다.

## 7) 중고거래(판매) - 물품조회

Hidden100은 적립금을 비밀로 진행하는 소액적립형 경매 서비스이기 때문에 일반적으로는 상품의 적립금을 확인할 수 없으나 자신이 등록한 상품은 적립금을 확인할 수 있다. 따라서 판매자의 물품조회를 다음과 같이 설계한다.

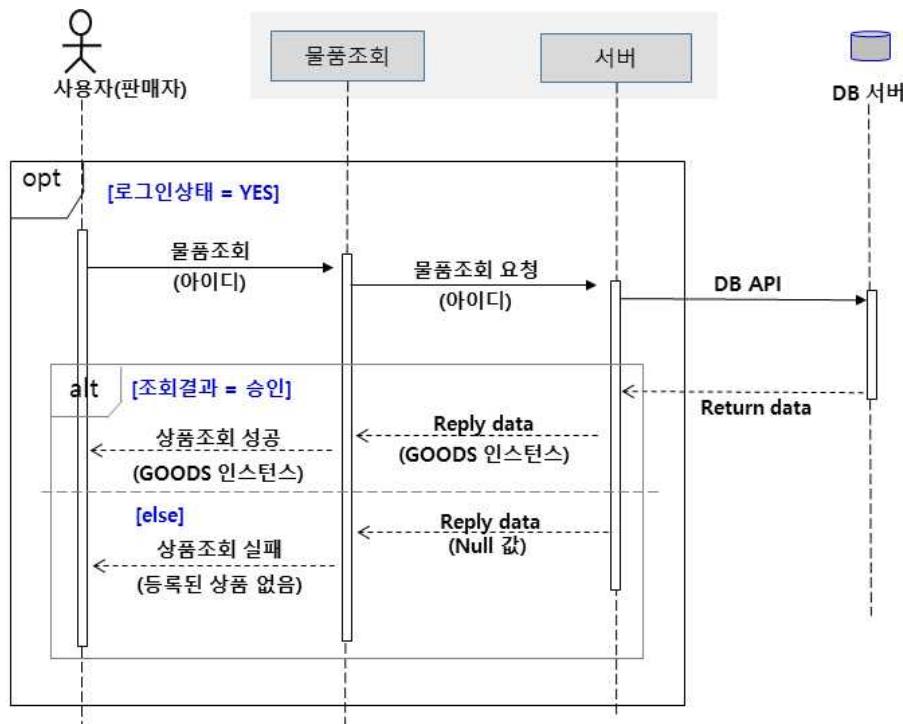


그림 3.8 물품조회의 Sequence Diagram

판매자의 물품 조회는 로그인 상태에서 진행된다. 그림 3.8을 보면 사용자가 페이지에 들어가 물품조회가 발생하면 서버로 물품조회 요청이 전송된다. DB에서 사용자 아이디를 가지고 검색해 상품이 있으면 해당하는 상품에 대한 정보를 사용자에게 전달한다. DB에 저장된 정보는 웹 화면에서 사용자가 열람하기 편하게 시각화되어 보여준다. 만약 사용자가 등록한 상품이 없어 상품 조회에 실패한 경우 등록된 상품이 없다는 안내를 사용자에게 보낸다.

### 8) 중고거래(판매) - 판매 취소

판매자가 부득이한 사정으로 중고 물품의 판매를 취소할 수 있다. 판매 취소를 구현하기 위해 다음과 같이 설계한다.

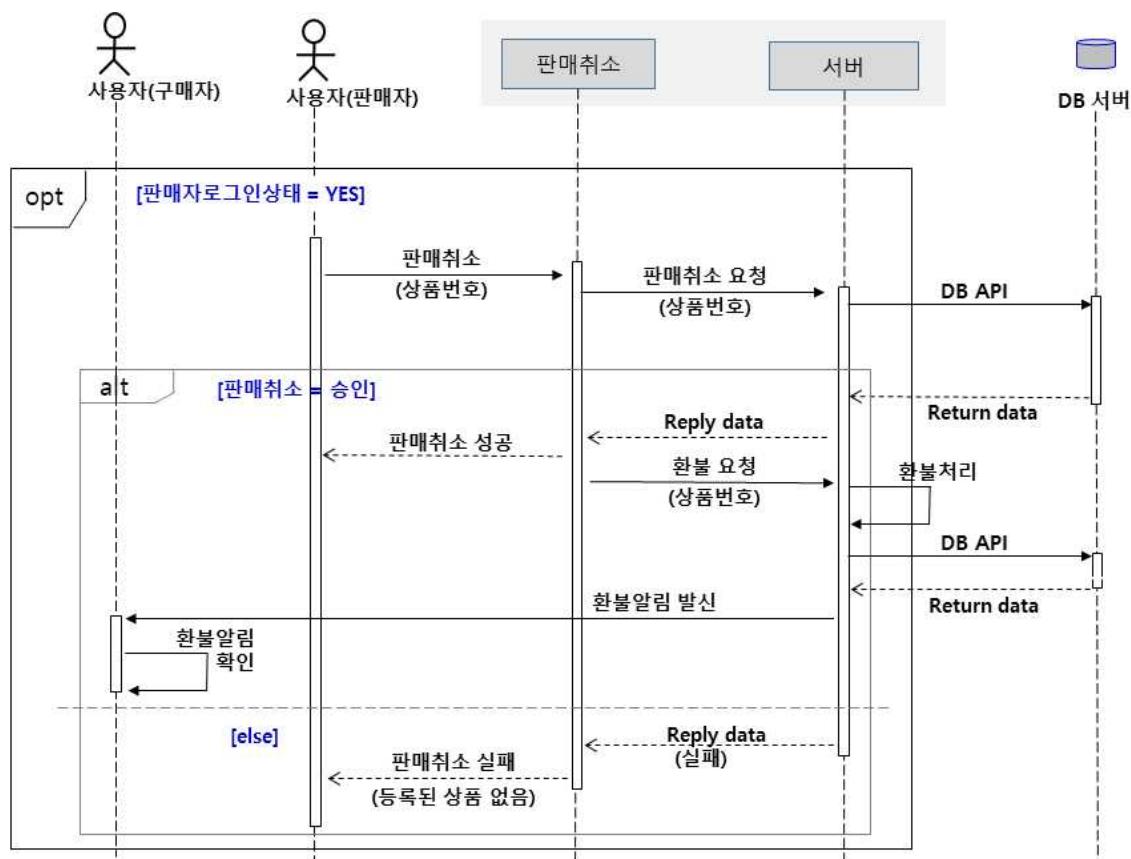


그림 3.9 물품조회의 Sequence Diagram

사용자(판매자)는 로그인한 상태에서 자신의 상품을 판매취소 요청할 수 있다. 판매자가 판매취소 요청을 보낼 때 상품 번호를 포함하여 서버에 전달한다. 서버는 DB에서 상품을 조회하고 삭제한다. DB에서 상품을 찾지 못한 경우에는 판매취소 실패를 판매자에게 보낸다. 이미 취소한 상품에 적립한 사람이 있는 경우에 환불처리를 수행하고 환불에 대해 사용자에게 푸시알림을 보내 안내한다.

### 9) 중고거래(판매) - 배송정보 등록

중고거래 적립이 끝난 후에 당첨자에게 물건을 보내고 사용자는 배송정보에 대해 입력한다. 배송정보 등록을 구현하기 위해 다음과 같이 설계한다.

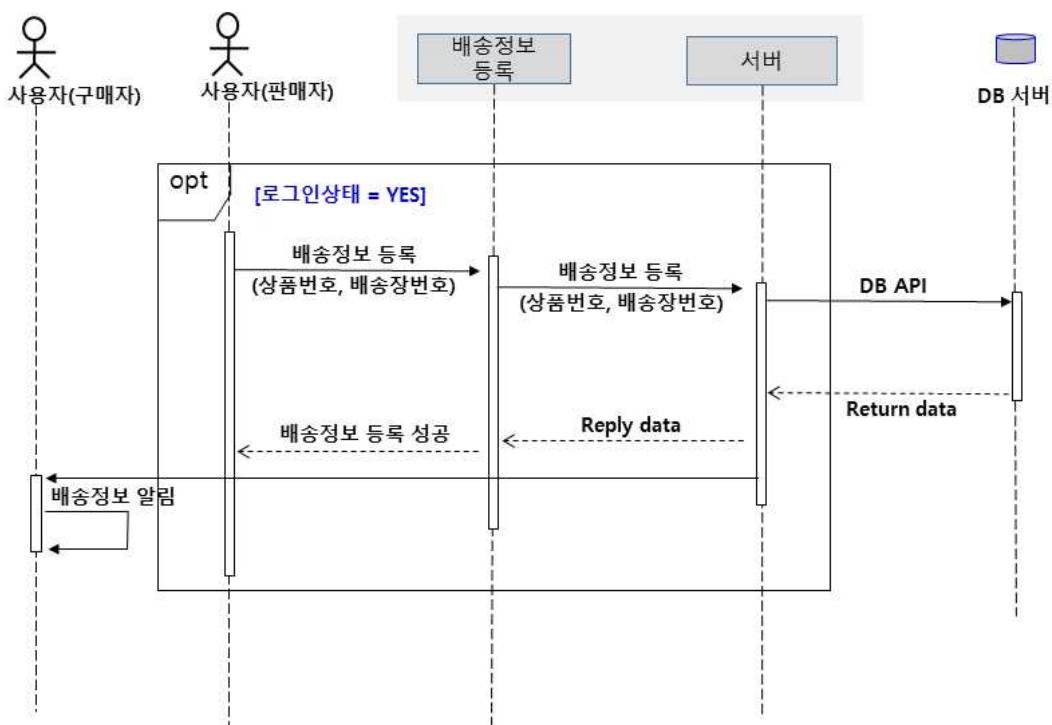


그림 3.10 배송정보 입력의 Sequence Diagram

그림 3.10을 보면 사용자(판매자)는 로그인한 상태에서 자신의 상품에 배송정보를 입력하여 사용자(구매자)에게 전달한다. 판매자가 입력한 정보는 서버로 넘어가고 서버에서는 DB를 조회해서 당첨자를 찾아 배송 정보를 알림으로 전달한다.

### 10) 중고거래(구매) - 코인충전

사용자는 서비스를 이용하기 위해 쇼핑몰 재화인 코인(coin)을 충전해야 한다. 코인 충전을 위해 아래와 같이 기능을 설계한다.

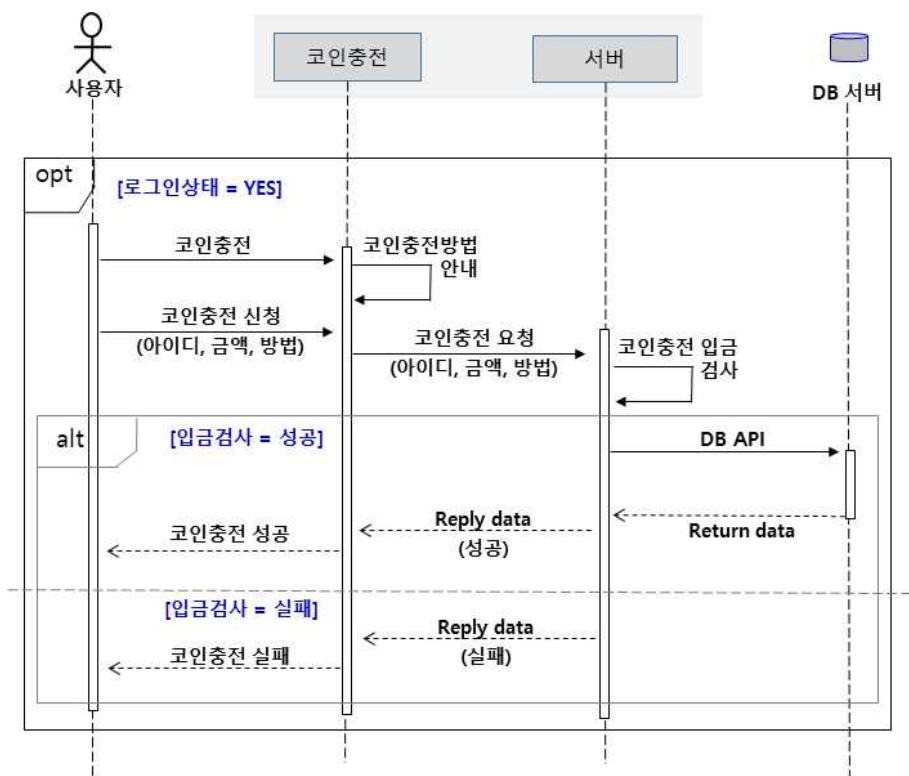


그림 3.11 코인충전의 Sequence Diagram

사용자는 웹화면으로 코인 충전에 대한 안내를 받는다. 사용자가 코인충전을 선택하여 아이디, 금액이 서버로 넘어간다. 서버에서는 코인충전 방식에 맞게 입금을 검사하고 결과에 따라 사용자가 코인충전에 성공한다. 입금되지 않았을 경우에는 코인이 충전되지 않고 결과를 사용자에게 안내한다. 사용자는 코인을 이용해 Hidden100의 서비스를 이용한다.

### 11) 중고거래(구매) - 코인적립

사용자는 Hidden100 서비스는 상품에 코인을 적립하여 추첨하는 랜덤경매의 형태이다. 코인적립을 위해 아래와 같이 기능을 설계한다.

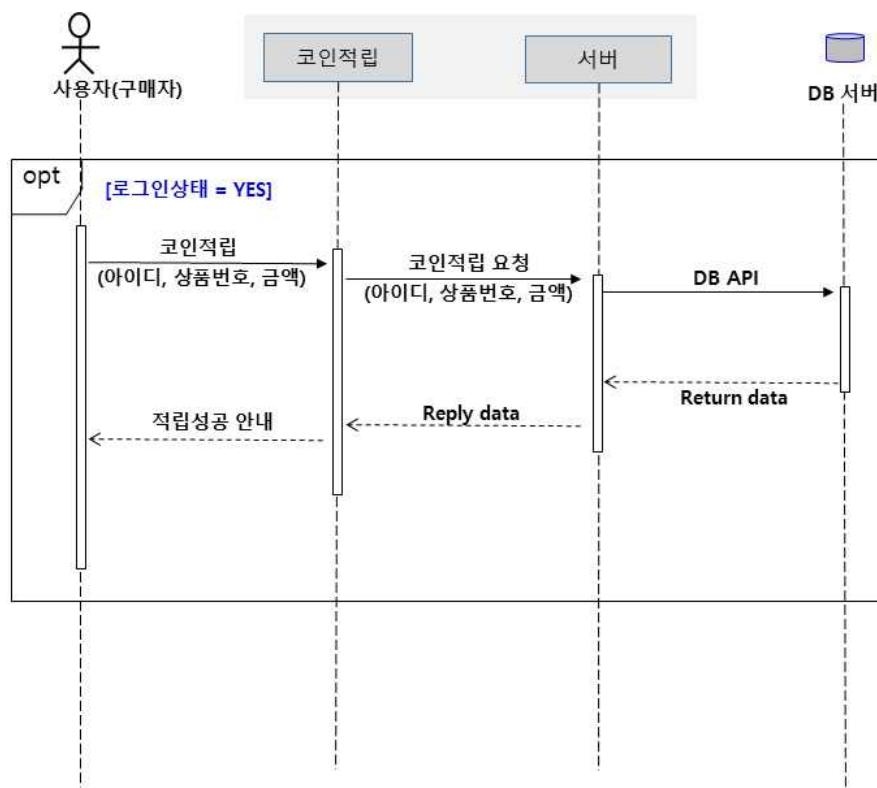


그림 3.12 코인적립의 Sequence Diagram

그림 3.12을 보면 사용자가 코인을 적립하면 적립내역이 DB에 저장된다. 사용자는 Web UI를 통해 코인을 적립한다. 상품에 응모버튼을 눌러 금액을 선택하고 아이디, 상품번호 정보를 서버에 같이 전달하여 코인을 적립한다. 적립한 참여자는 DB에 저장되고 이후 추첨을 통해 중고 상품을 받을 수 있다.

## 12) 중고거래(구매) - 판매물품 조회

판매물품 조회는 로그인 없이도 가능한 기능으로 사용자는 물품을 검색하거나 조회할 수 있다. 판매물품 조회 기능은 아래와 같이 설계한다.

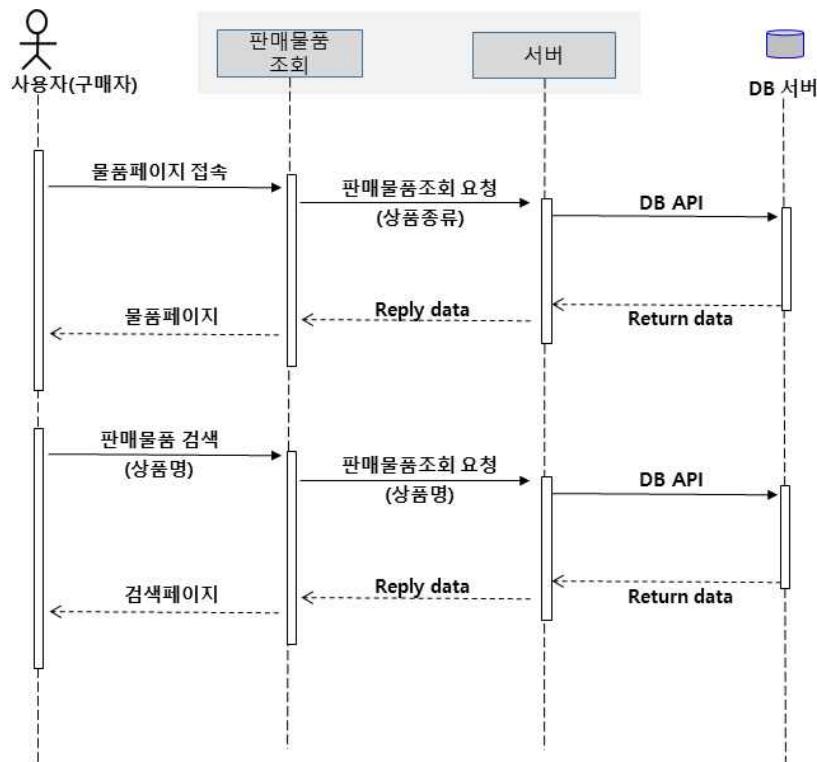


그림 3.13 판매물품 조회의 Sequence Diagram

그림 3.13과 같이 판매물품 조회는 웹페이지에 접속하거나 특정 버튼을 눌렀을 경우 상품에 대한 조회 기능을 수행한다. 상품 종류에 대해 요청하면 해당하는 상품의 정보를 가져와 웹페이지에 출력한다. 혹은 상품 이름, 키워드로 검색하면 해당하는 상품을 DB에 검색하여 결과를 웹페이지에 출력하여 사용자가 상품을 조회한다. 검색과 조회는 사용자 인증이 없어도 사용 가능하다.

### 13) 중고거래(구매) - 거래내역조회

사용자는 자신이 참여한 거래내역을 조회한다. 거래내역 조회 기능은 아래와 같이 설계한다.

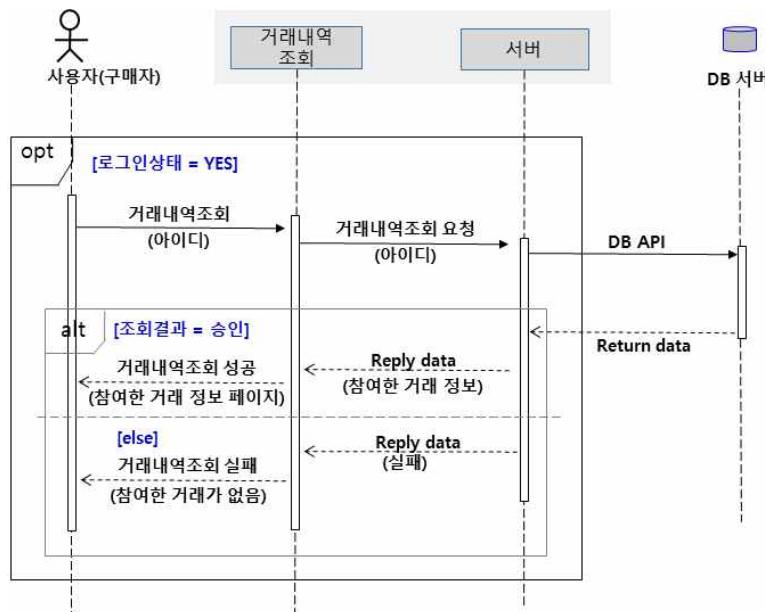


그림 3.14 판매물품 조회의 Sequence Diagram

그림 3.14과 같이 사용자가 로그인한 상태에서 거래내역 조회를 요청하면 서버로 사용자의 아이디를 전달하고, DB에서 해당 아이디가 참여한 거래 내역을 가져온다. 받아온 정보는 시각화 기능을 이용하여 웹페이지 형태로 사용자에게 전달되고 참여한 거래 정보가 없으면 실패로 처리하여 조회할 정보가 없다고 안내한다.

#### 14) 중고거래(구매) - 참여자 수 조회

사용자는 거래 상품의 총적립금액은 알 수 없지만, 참여자 수는 확인할 수 있다. 참여자 수 조회 기능을 구현하기 위해 다음과 같이 설계한다.

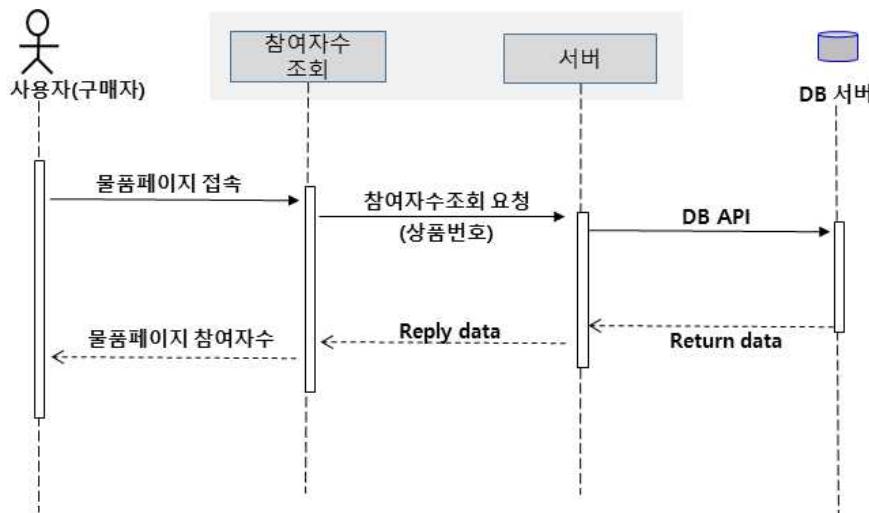


그림 3.15 참여자수조회의 Sequence Diagram

그림 3.15와 같이 사용자가 물품 페이지에 접속하면 참여자수에 대한 정보도 같이 제공된다. DB에 담긴 정보를 종합하여 웹페이지에 함께 출력한다. 요청이 들어오면 서버는 DB에서 검색한 정보를 처리하여 사용자에게 보낸다.

### 15) 중고거래(구매) - 거래신고

사용자는 중고거래 도중 사기가 의심되는 경우 거래신고를 할 수 있어야 한다. 거래신고 기능을 구현하기 위해 아래와 같이 설계한다.

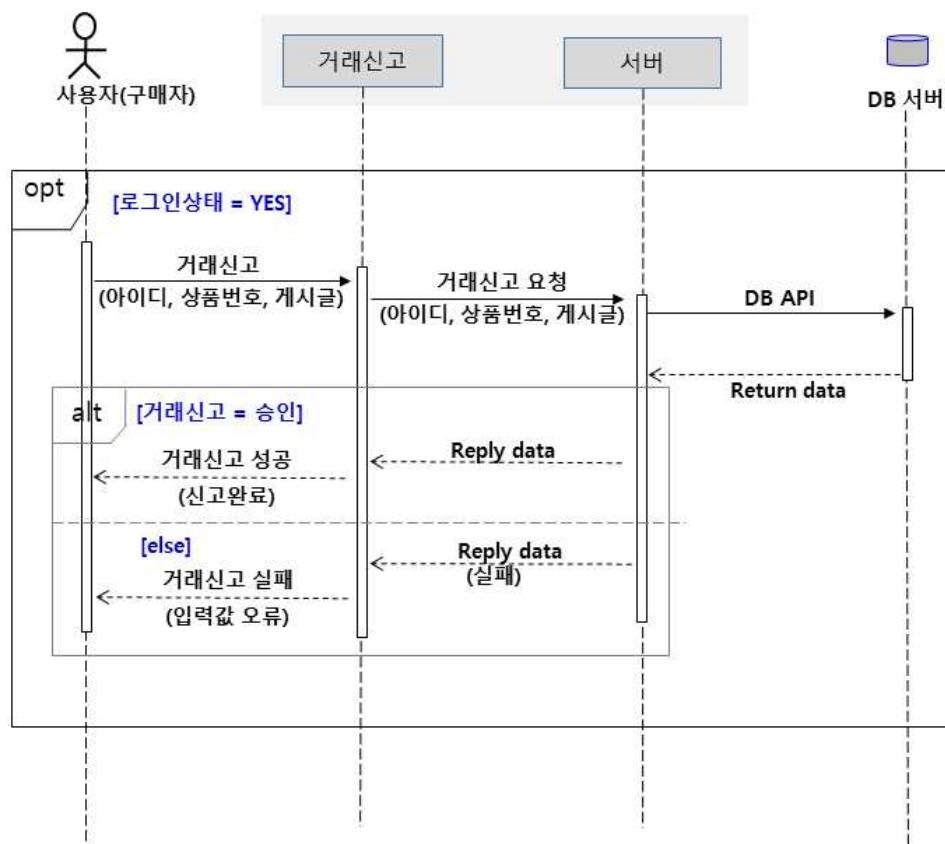


그림 3.16 거래신고의 Sequence Diagram

그림 3.16과 같이 사용자는 거래신고하기 위해 로그인을 해야 한다. 사용자가 신고할 상품에 대해 상품 번호와 신고 내용을 작성하여 서버에 전달한다. 서버는 해당 내용을 DB에 저장하고 결과에 따라 신고 결과를 안내한다. 신고 결과에 따른 보상이나 처리는 관리자 영역에서 수행한다. 사용자는 위의 과정을 통해 거래사기 신고접수를 할 수 있다.

## 16) 게시판 모듈 - 게시글 등록

사용자는 게시글을 작성할 수 있어야 한다. 게시글 등록 기능을 구현하기 위해 아래와 같이 설계한다.

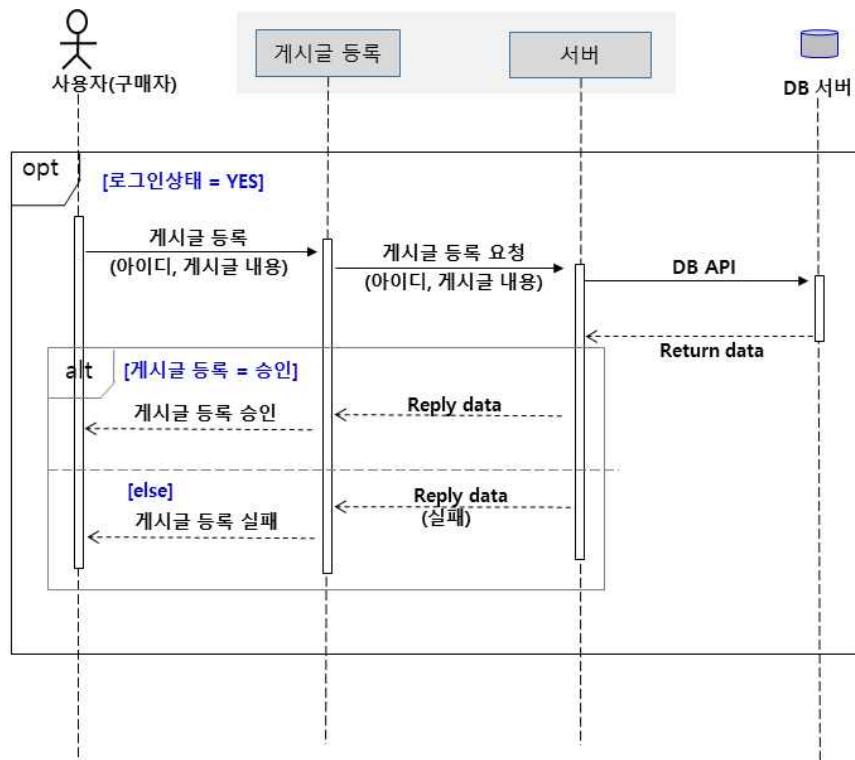


그림 3.17 게시글 등록의 Sequence Diagram

그림 3.17과 같이 사용자는 게시글을 등록하기 위해서 사용자는 로그인해야 한다. 로그인한 사용자가 게시글 내용을 작성하여 등록버튼을 누르면 서버로 정보를 전송하여 DB에 저장한다. 내용이 비어있거나 오류가 있는 경우에는 게시글 등록 실패로 연결하며 게시글 등록에 성공한 경우 성공 메시지를 전달한다.

### 17) 게시판 모듈 - 게시글 수정

사용자는 작성한 게시글의 내용을 수정할 수 있다. 게시글 수정 기능을 구현하기 위해 아래와 같이 설계한다.

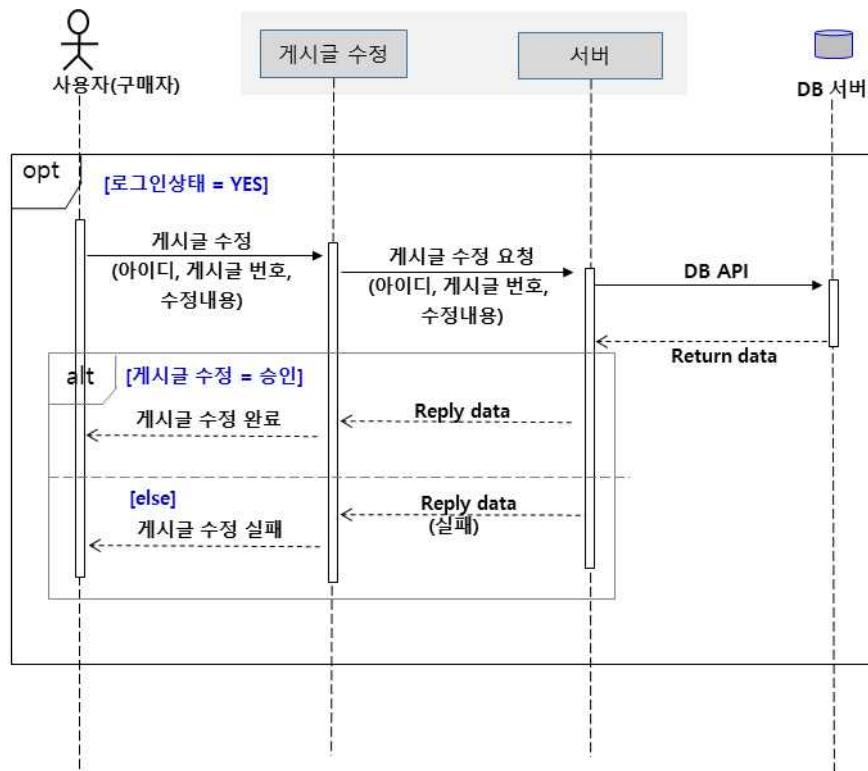


그림 3.18 게시글 수정의 Sequence Diagram

그림 3.18과 같이 사용자는 게시글을 수정하기 위해서 사용자는 로그인해야 한다. 로그인한 사용자가 자신이 작성한 게시글의 수정 버튼을 누르면 게시글을 수정할 수 있다. 새로운 게시글 내용을 작성하여 게시글을 저장하면 변경된 게시글 내용과 게시글 번호, 아이디가 서버로 전달된다. 서버는 해당하는 게시글의 내용을 DB에서 변경하고 사용자에게 수정 결과를 안내한다.

## 18) 게시판 모듈 - 게시글 삭제

사용자는 작성한 게시글의 내용을 삭제할 수 있다. 게시글 삭제 기능을 구현하기 위해 아래와 같이 설계한다.

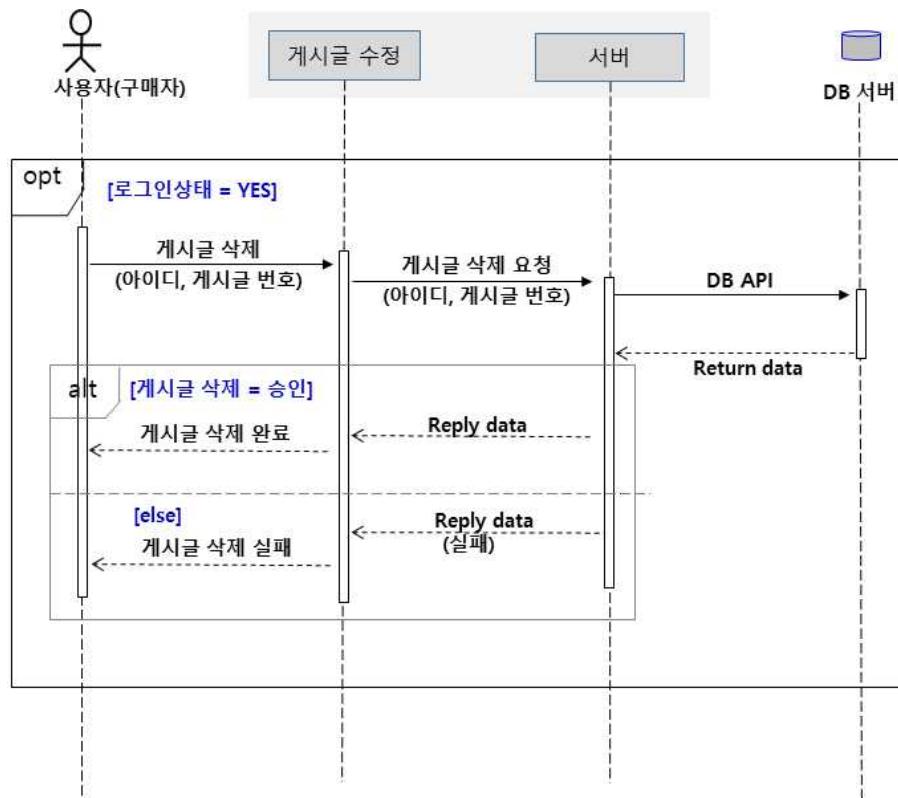


그림 3.19 게시글 삭제의 Sequence Diagram

그림 3.19과 같이 사용자는 자신의 게시글을 삭제하기 위해서 로그인해야 한다. 사용자가 자신의 게시물에 삭제 버튼을 누르면 아이디와 게시글 번호를 전달하여 삭제를 요청한다. DB에서 해당 게시글 데이터를 삭제하고 서버로 결과를 반환하며, 서버는 사용자에게 반환 결과를 안내한다.

### 19) 게시판 모듈 - 게시글 조회

사용자는 작성한 게시글의 내용을 조회할 수 있다. 게시글 조회 기능을 구현하기 위해 아래와 같이 설계한다.

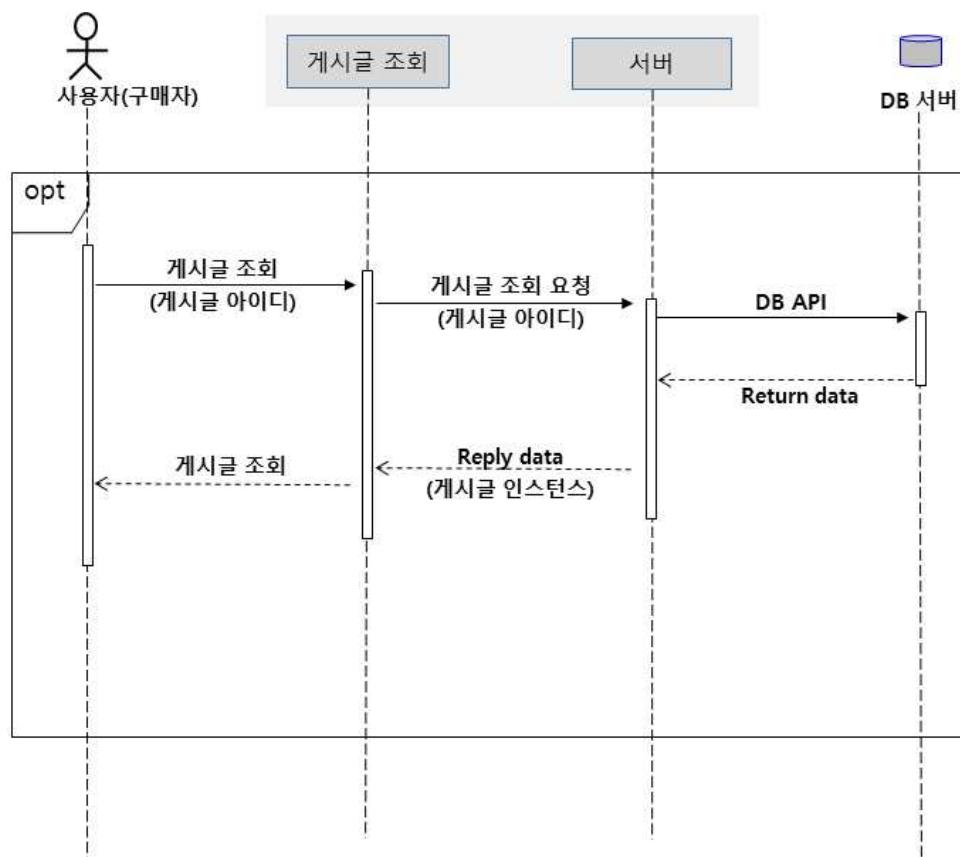


그림 3.20 게시글 조회의 Sequence Diagram

그림 3.20과 같이 사용자는 로그인 상태에서 다른 사용자의 게시글을 조회할 수 있다. 웹페이지에서 게시글을 조회하면 해당 게시글에 대한 아이디로 게시글 정보를 서버에 요청하고 서버는 DB에서 해당 게시글의 내용을 받아 전달한다. 사용자는 전달된 데이터를 웹페이지 형태로 조회한다.

## 20) 챗봇기능 - 내 정보 조회

사용자는 챗봇에서 자신의 정보를 조회할 수 있다. 내 정보 조회 기능을 구현하기 위해 아래와 같이 설계한다.

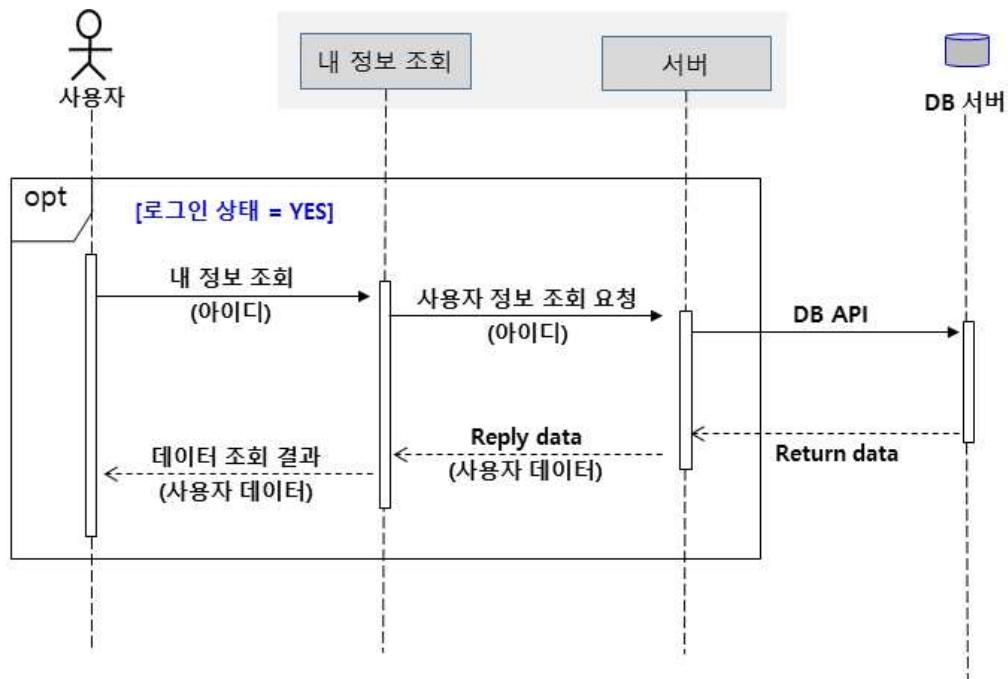


그림 3.21 내 정보 조회의 Sequence Diagram

그림 3.21과 같이 사용자는 로그인 상태에서 자신의 정보를 조회할 수 있다. 챗봇 UI로 내 정보 조회를 요구하면 서버로 사용자 정보 조회 요청을 보낸다. 서버에서 해당 사용자의 정보를 DB에서 조회해 돌려주면 사용자는 챗봇 UI를 통해 자신의 정보를 조회한다.

## 21) 챗봇기능 - 판매상품 조회

사용자는 챗봇에서 간편하게 대화하는 방식으로 판매상품을 조회할 수 있다. 판매 상품 조회 기능을 구현하기 위해 아래와 같이 설계한다.

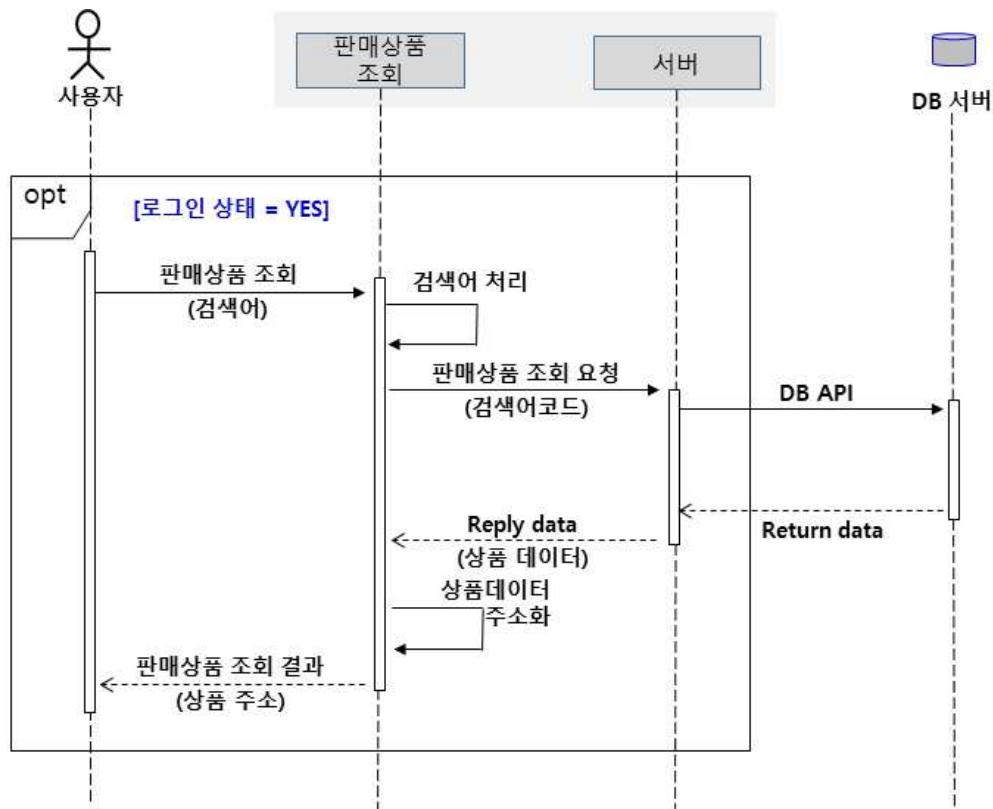


그림 3.22 내 정보 조회의 Sequence Diagram

그림 3.22과 같이 사용자는 로그인한 다음 챗봇으로 판매상품을 조회할 수 있다. 사용자가 원하는 상품의 키워드 등을 입력하면 해당 문자열을 처리하여 검색어 코드로 변경한다. 해당 검색어 코드에 가장 적합한 상품을 DB에서 조회해서 해당 상품의 주소를 사용자에게 전달하여 조회 서비스를 제공한다.

## 22) 챗봇기능 - FAQ

사용자는 챗봇에서 간편하게 대화하는 방식으로 FAQ를 이용할 수 있다. FAQ 기능을 구현하기 위해 아래와 같이 설계한다.

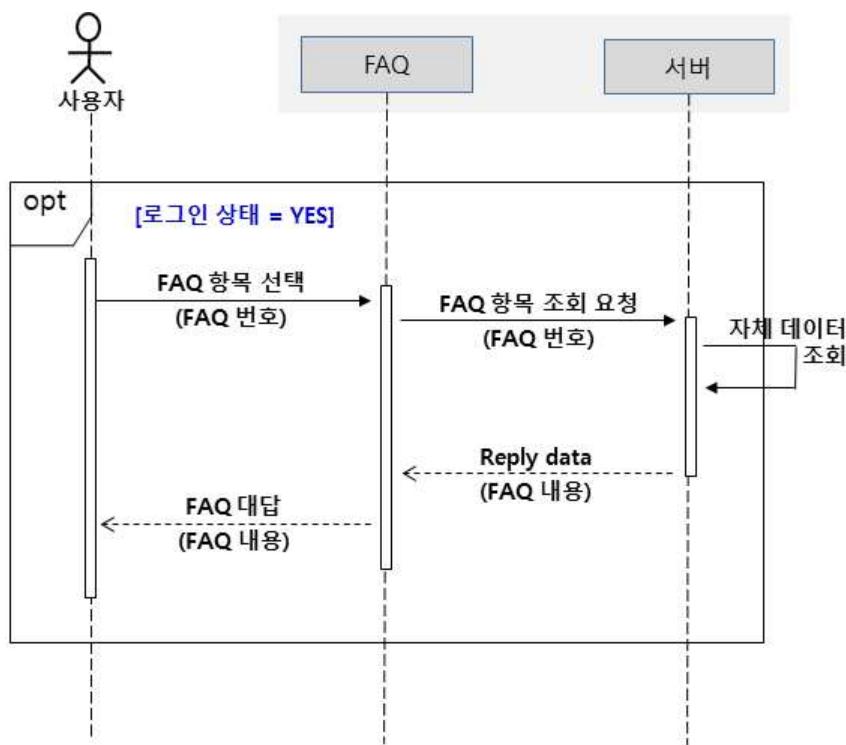


그림 3.23 FAQ 기능의 Sequence Diagram

그림 3.23과 같이 사용자가 로그인한 다음 챗봇 UI에서 FAQ 항목을 선택한다. 선택된 FAQ 번호는 서버로 넘어가서 데이터를 조회하고 해당하는 결과를 다시 반환한다. 데이터를 챗봇 UI로 처리하여 사용자는 간편하게 FAQ 서비스를 이용한다.

### 3.3. 관리자기능 설계 (Sequence Diagram)

#### 1) 관리자 인증 - 관리자 회원가입

관리자는 인증키를 이용하여 회원가입을 진행할 수 있다. 관리자의 회원가입 기능을 구현하기 위해 아래와 같이 설계한다.

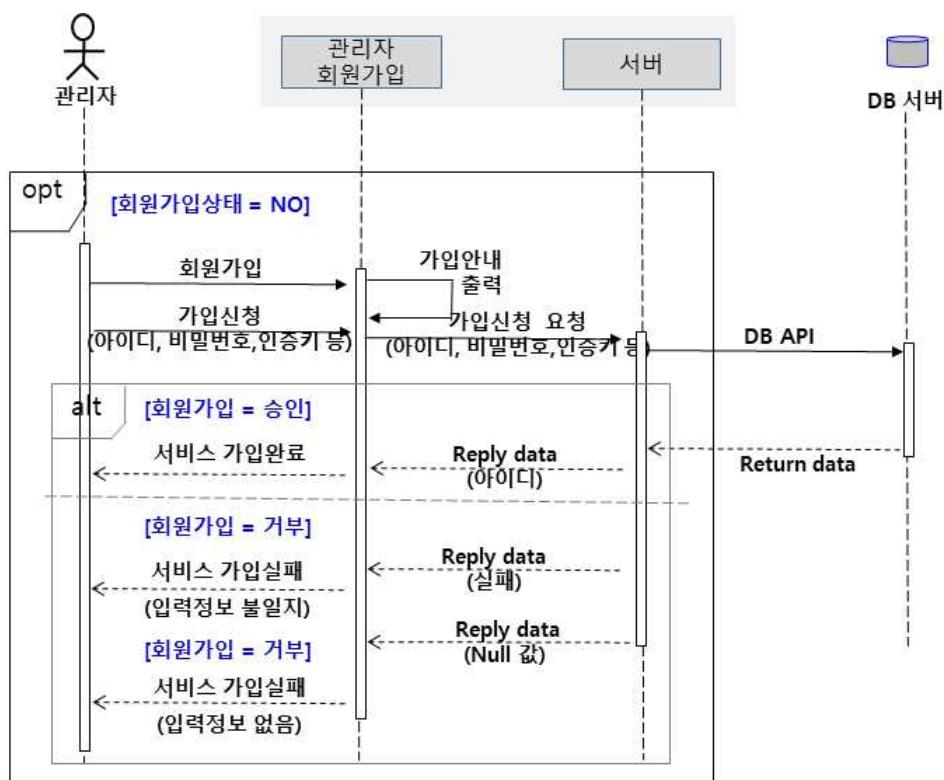


그림 3.24 회원가입의 Sequence Diagram

그림 3.24를 보면 관리자가 회원가입을 신청하면 가입정보를 입력할 수 있는 폐이지로 연결한다. 기존에 등록된 인증키와 관리자가 입력한 인증키가 일치하는 경우에 회원가입에 성공하며 정보가 일치하지 않거나 비어있을 경우 회원가입에 거부된다. 회원가입에 성공한 관리자는 관리자 인증을 통해 관리자 기능을 수행할 수 있다. 가입 결과에 대한 메시지를 관리자에게 전달한다.

## 2) 관리자 인증 - 관리자 로그인

관리자는 회원가입 이후 로그인을 통해 관리자 인증을 하고 관리자 기능을 사용할 수 있다. 관리자의 로그인 기능을 구현하기 위해 아래와 같이 설계한다.

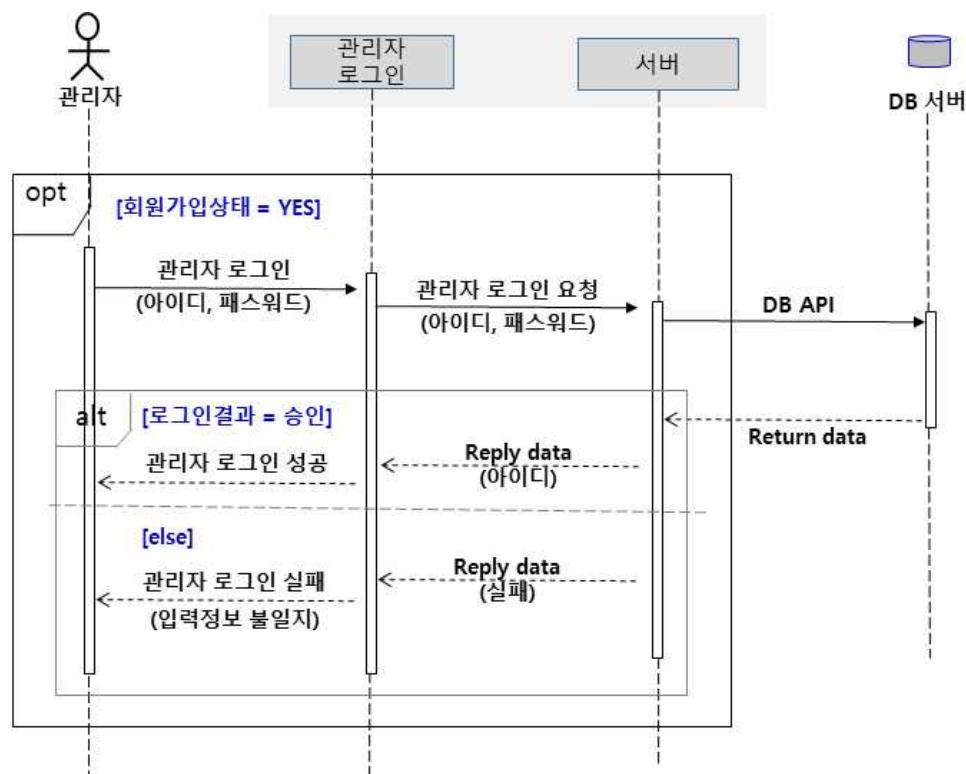


그림 3.25 관리자 로그인의 Sequence Diagram

그림 3.25를 보면 관리자가 아이디와 패스워드를 입력하여 로그인할 수 있다. 서버에서는 로그인 요청을 받아 DB에서 아이디를 키로 관리자 정보를 받아와 입력받은 정보와 비교한다. 아이디와 패스워드가 DB에 저장된 관리자 정보와 일치하면 로그인에 성공한다. 로그인하고 나서 다른 Hidden100 사용자 서비스를 이용할 수 있다. 패스워드가 다르거나 정지계정인 경우 로그인에 실패한다. 로그인에 실패한 경우 실패 메시지와 실패 사유를 안내한다.

### 3) 관리자 인증 - 관리자 로그아웃

관리자는 관리자 기능의 사용을 종료할 때 로그아웃으로 관리자 로그인 데이터를 삭제한다. 관리자의 로그아웃 기능을 구현하기 위해 아래와 같이 설계한다.

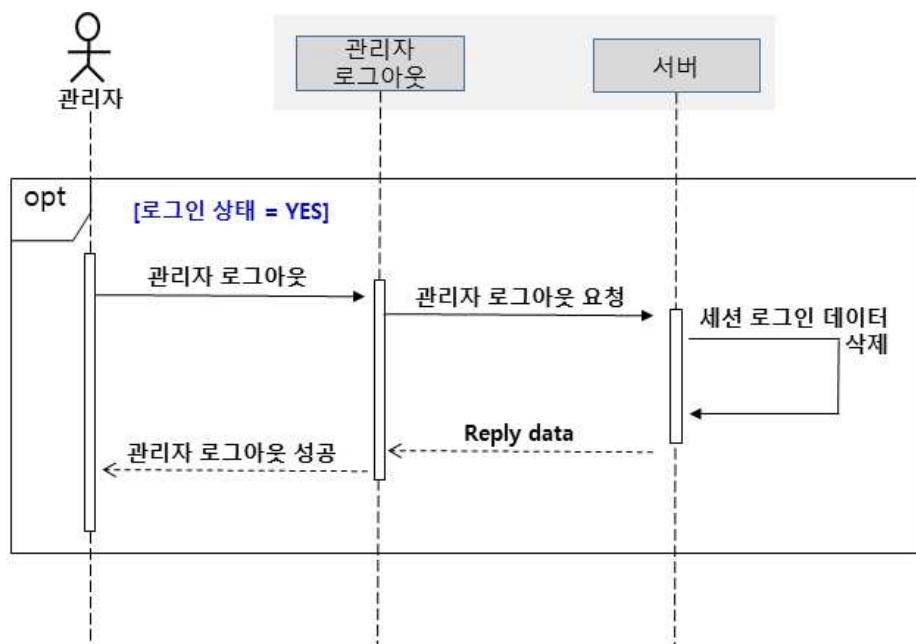


그림 3.26 관리자 로그아웃의 Sequence Diagram

그림 3.26를 보면 관리자는 로그아웃을 수행하기 위해 로그아웃 버튼을 누른다. 로그아웃 기능은 서버로 가서 현재 세션에 저장된 관리자의 로그인 데이터를 삭제한다. 서버에서 로그아웃을 처리한 후에 사용자에게 로그아웃된 화면으로 전송하여 보여준다.

#### 4) 회원관리 - 회원조회

관리자는 관리자 인증을 마친 후에 회원관리 기능을 사용할 수 있다. 관리자의 회원조회 기능을 구현하기 위해 아래와 같이 설계한다.

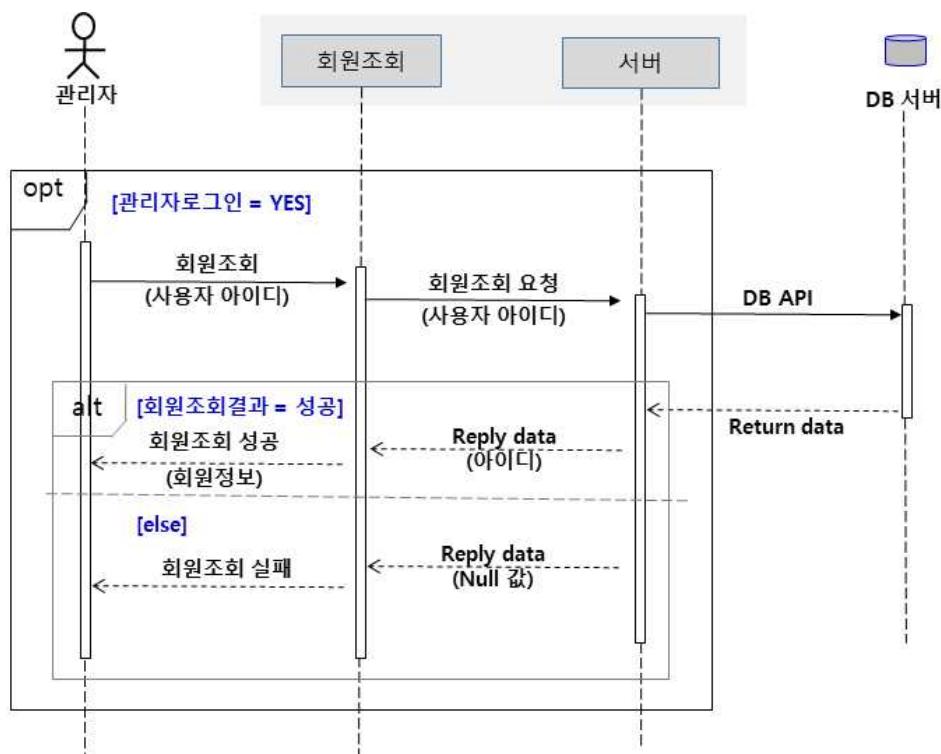


그림 3.27 회원조회의 Sequence Diagram

그림 3.27를 보면 관리자는 관리자 인증을 수행한 후에 사용자의 아이디로 회원 정보를 조회할 수 있다. 관리자가 사용자 아이디를 입력하면 사용자 아이디를 서버로 전달해 DB를 조회할 수 있도록 한다. DB에 조회에 성공하면 관리자에게 해당 사용자의 정보를 제공하고 DB에서 조회에 실패했을 경우 실패 메시지로 안내한다.

## 5) 회원관리 - 회원등급 관리

관리자는 회원의 정보를 관리할 수 있으며, 회원 등급을 변경할 수 있다. 관리자의 회원등급 관리 기능을 구현하기 위해 아래와 같이 설계한다.

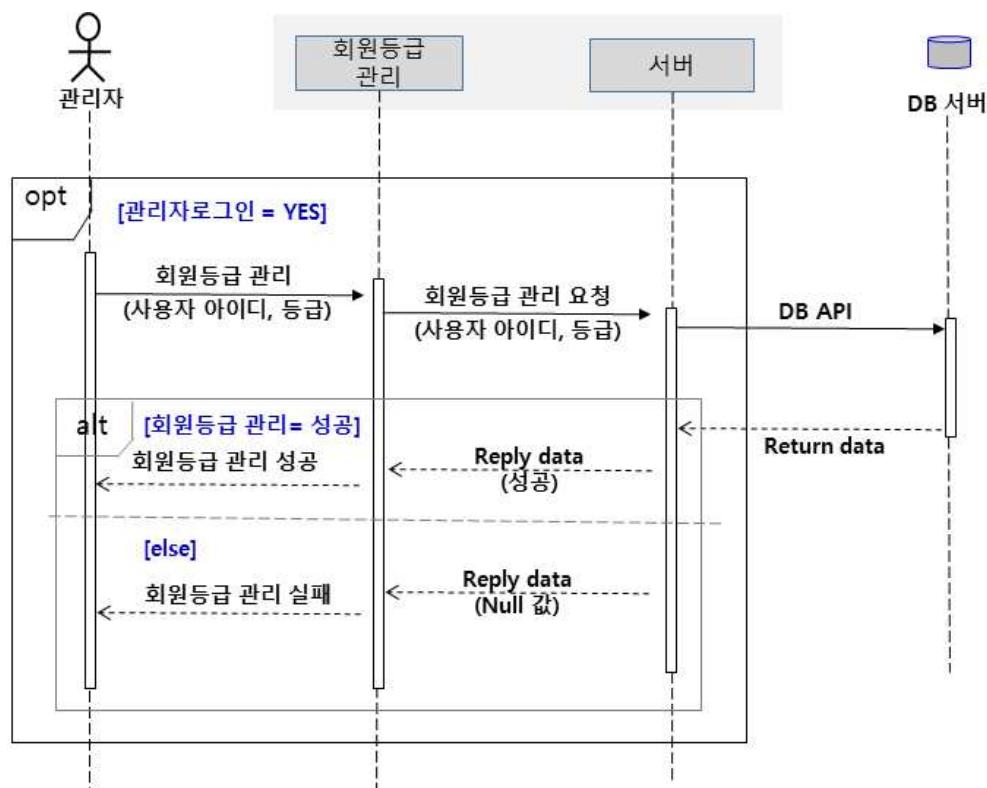


그림 3.28 회원등급 관리의 Sequence Diagram

그림 3.28를 보면 관리자는 관리자 인증을 성공한 후에 사용자의 등급을 변경할 수 있다. 변경하고자 하는 사용자의 아이디와 변경할 등급을 입력하면 해당 내용을 서버에 전달하고 서버에서 DB에 접근해 해당 사용자의 정보를 변경한다. 변경에 성공한 경우 관리자에게 성공 안내 메시지를 전달하고 실패한 경우 실패 안내 메시지를 전달한다.

## 6) 회원관리 - 회원설정 관리

관리자는 회원의 정보를 관리할 수 있으며, 회원설정을 변경할 수 있다. 관리자의 회원설정 관리 기능을 구현하기 위해 아래와 같이 설계한다.

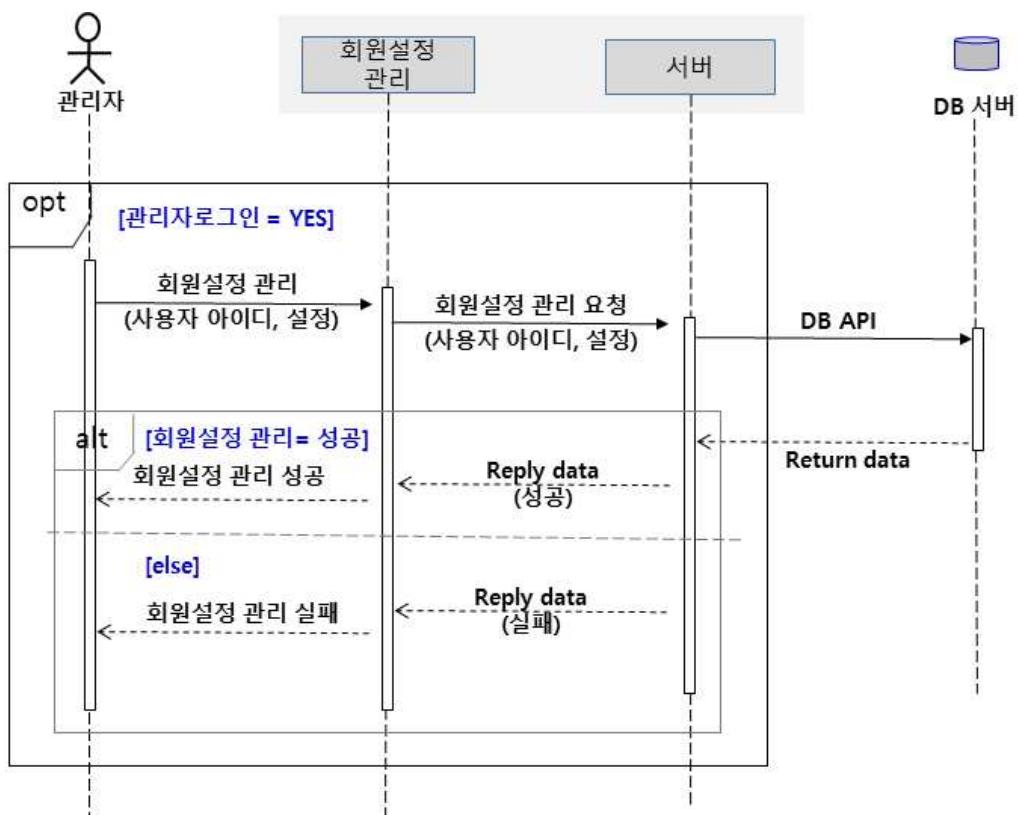


그림 3.29 회원설정 관리의 Sequence Diagram

그림 3.29를 보면 관리자는 관리자 인증을 성공한 후에 사용자의 설정을 변경할 수 있다. 변경하고자 하는 사용자의 아이디와 변경할 설정을 입력하면 해당 내용을 서버에 전달하고 서버에서 DB에 접근해 해당 사용자의 정보를 변경한다. 웹으로 사용자의 정보 데이터를 변경하며, 변경에 성공한 경우 관리자에게 성공 안내 메시지를 전달하고 실패한 경우 실패 안내 메시지를 전달한다.

## 7) 회원관리 - 회원정지 관리

관리자는 회원을 정지할 수 있으며 정지한 사용자는 로그인할 수 없다. 관리자의 회원정지 관리 기능을 구현하기 위해 아래와 같이 설계한다.

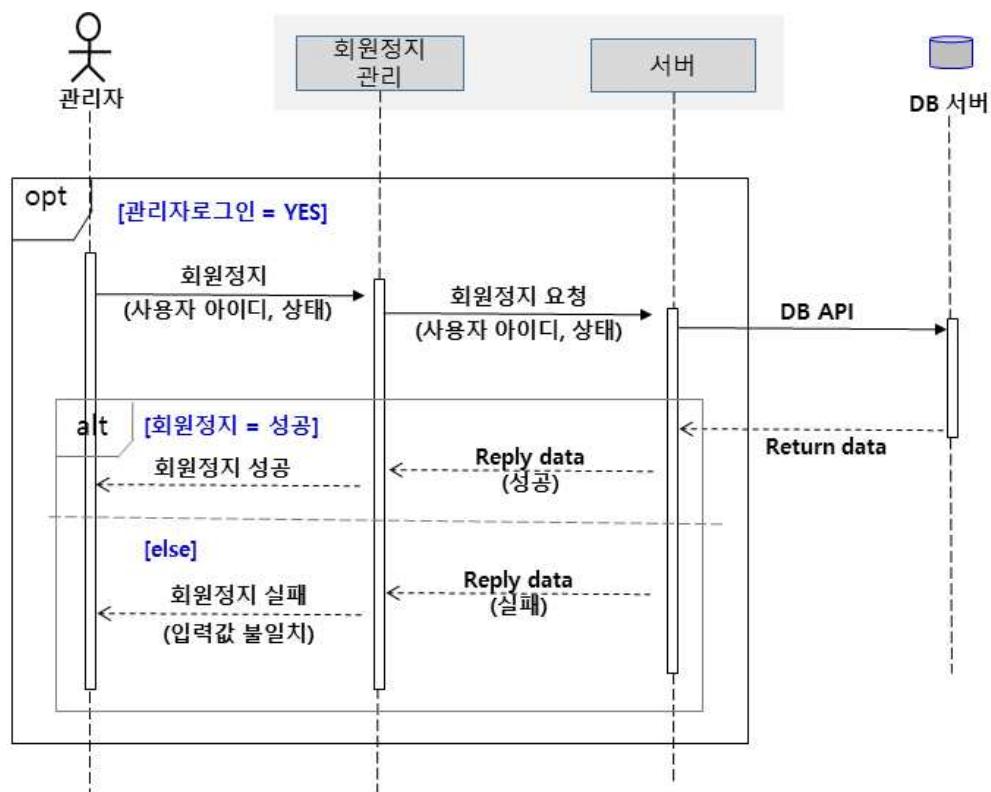


그림 3.30 회원정지 관리의 Sequence Diagram

그림 3.30를 보면 관리자는 관리자 인증을 성공한 후에 사용자를 정지시킬 수 있다. 정지 할 사용자의 아이디를 입력하면 해당 내용을 서버에 전달하고 서버에서 DB에 접근해 해당 사용자를 정지회원으로 변경한다. 변경에 성공한 경우 관리자에게 성공 안내 메시지를 전달하고 실패한 경우 해당하는 실패 안내 메시지를 전달한다.

### 8) 회원관리 - 회원탈퇴 관리

관리자는 회원을 탈퇴시킬 수 있으며 해당 사용자의 정보는 삭제된다. 관리자의 회원탈퇴 관리 기능을 구현하기 위해 아래와 같이 설계한다.

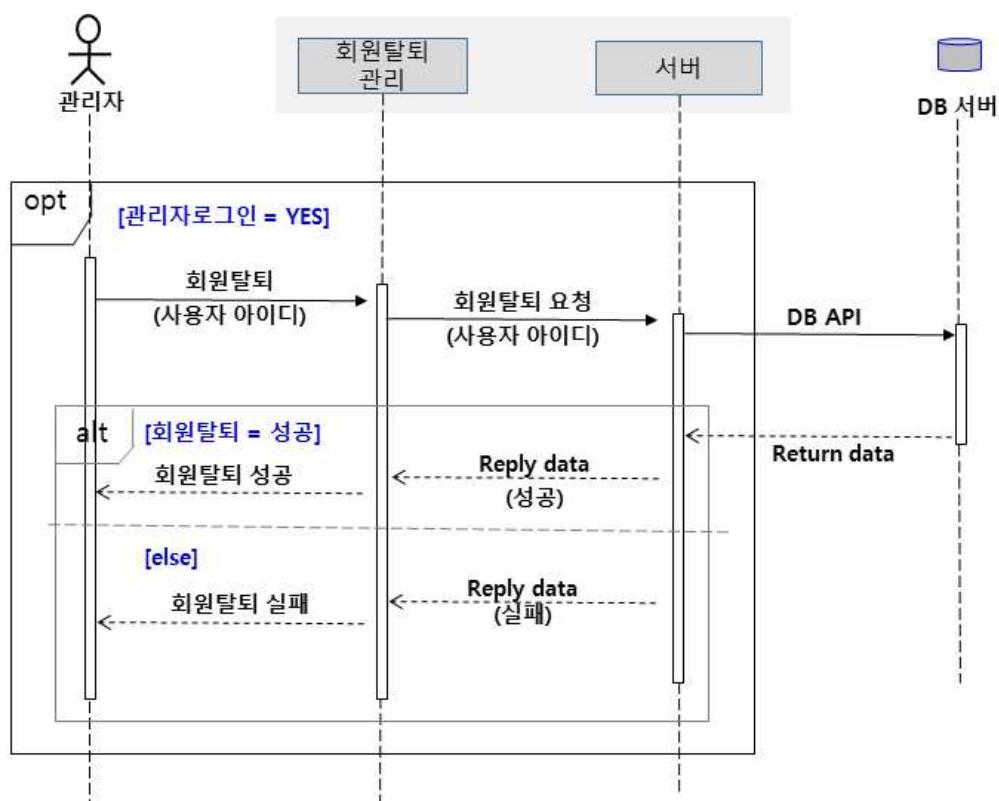


그림 3.31 회원탈퇴 관리의 Sequence Diagram

그림 3.31를 보면 관리자는 관리자 인증을 성공한 후에 사용자를 탈퇴시킬 수 있다. 탈퇴 시킬 사용자의 아이디를 입력하면 해당 내용을 서버에 전달하고 서버에서 DB에 접근해 해당 사용자를 탈퇴 처리한다. 탈퇴에 성공한 경우 관리자에게 성공 안내 메시지를 전달하고 실패 한 경우 해당하는 실패 안내 메시지를 전달한다.

### 9) 챗봇기능 - 알림 발신

관리자는 사용자에게 알림을 보낼 수 있으며 알림 발신 기능을 구현하기 위해 아래와 같이 설계한다.

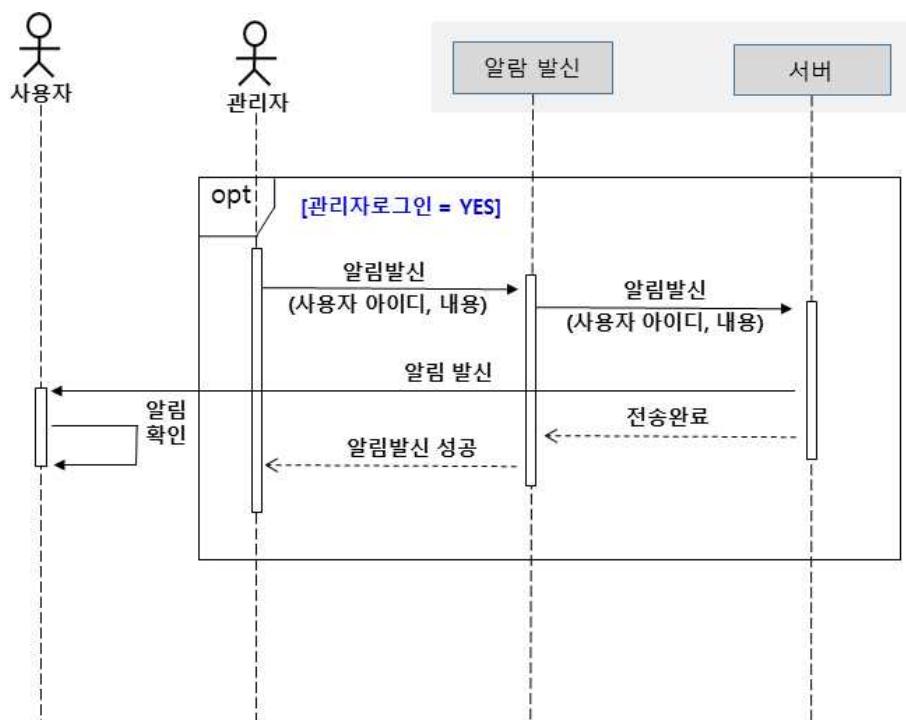


그림 3.32 알림 발신의 Sequence Diagram

그림 3.32를 보면 관리자는 관리자 인증을 성공한 후에 사용자에게 알림을 보낼 수 있다. 알림을 받을 사용자의 아이디와 보낼 내용을 작성하여 입력하면 서버로 해당 내용이 전송된다. 서버에서는 해당 사용자에게 내용을 알림으로 보내고 전송 완료 메시지를 다시 관리자에게 보낸다. 관리자는 사용자 아이디를 이용하여 사용자에게 알림을 발신할 수 있다.

## 10) 챗봇기능 - FAQ 관리

관리자는 사용자가 챗봇으로 확인하는 FAQ 내용을 관리할 수 있다. FAQ 관리 기능을 구현하기 위해 아래와 같이 설계한다.

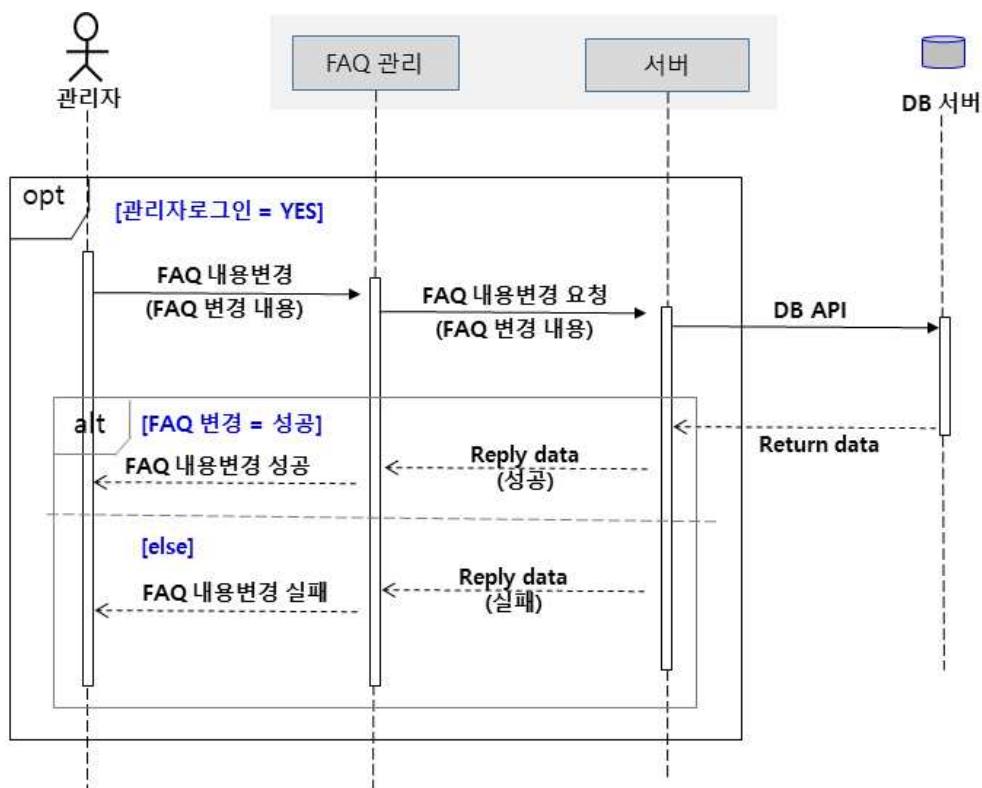


그림 3.33 FAQ 관리의 Sequence Diagram

그림 3.33를 보면 관리자는 관리자 인증을 성공한 후에 FAQ를 편집할 수 있다. 관리자가 변경 내용에 대해 서버로 보내면 서버에서는 DB에 해당 내용을 저장한다. 저장에 성공한 경우 관리자에게 내용 변경 성공 메시지를 전달하고 실패한 경우 실패 메시지를 전달한다. 관리자는 Web UI를 통해 챗봇에 대한 지식이 없어도 간편하게 FAQ를 관리할 수 있다.

### 11) 시각화 - 챗봇로그

관리자는 사용자가 챗봇으로 입력한 내용에 대한 분석자료를 확인할 수 있다. 챗봇로그 기능을 구현하기 위해 아래와 같이 설계한다.

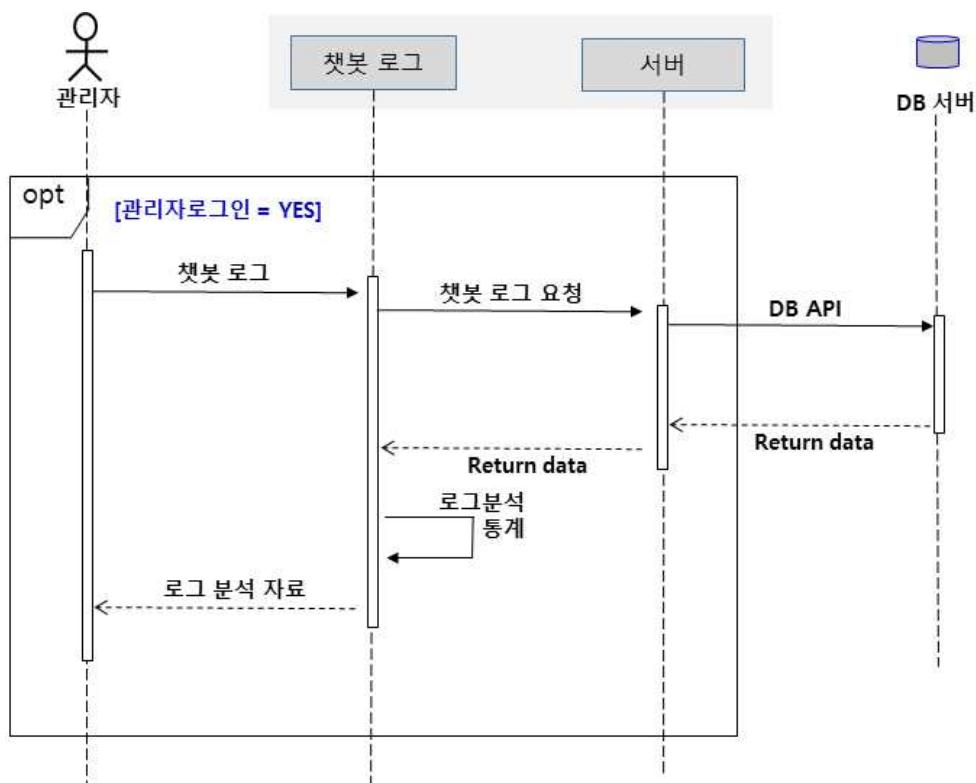


그림 3.34 챗봇로그의 Sequence Diagram

그림 3.34를 보면 관리자는 관리자 인증을 성공한 후에 챗봇 로그를 확인할 수 있다. 관리자가 챗봇로그 조회를 요청하면 서버에서는 챗봇을 이용한 사용자의 로그 데이터를 분석한다. 검색한 내용에 대해 통계를 내서 사용자가 주로 검색하고 사용하는 기능에 대해 시각화하여 Web UI로 제공한다. 관리자는 시각화된 자료로 쉽게 사용자가 챗봇의 어떤 기능을 주로 이용하고 어떤 내용을 검색하는지 등을 확인할 수 있다.

### 11) 중고거래 기능 - 실시간 상품등록 조회

관리자는 실시간으로 상품에 대한 모든 정보를 조회할 수 있다. 실시간 상품등록 조회 기능을 구현하기 위해 아래와 같이 설계한다.

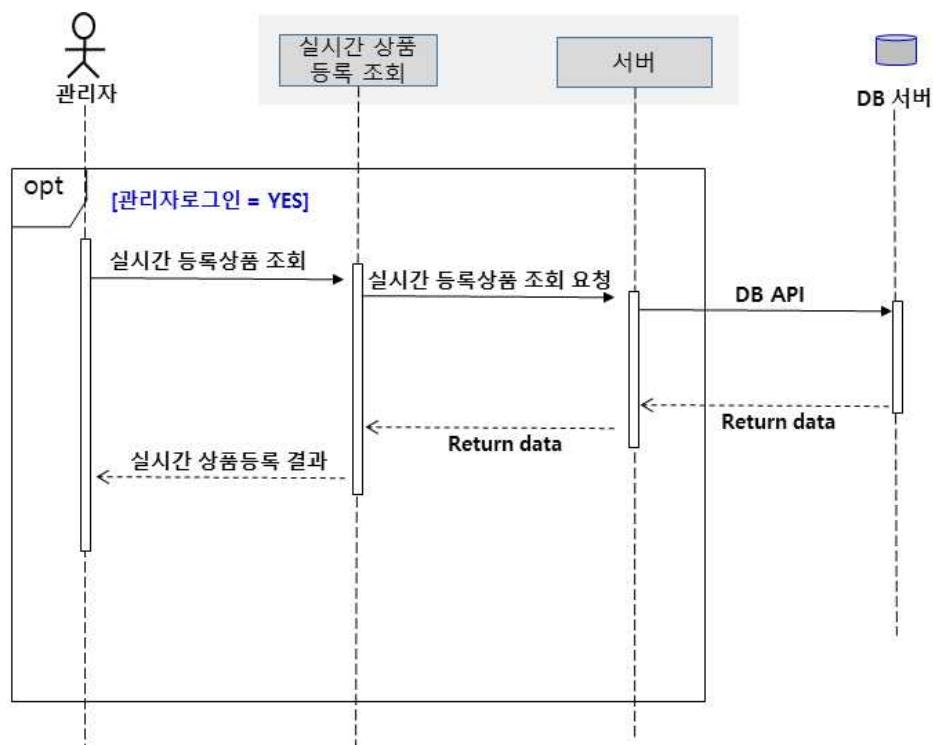


그림 3.35 실시간 상품등록 조회의 Sequence Diagram

그림 3.35를 보면 관리자는 관리자 인증을 성공한 후에 실시간으로 등록상품을 조회할 수 있다. 관리자가 조회 페이지에 들어가면 서버로 조회 요청이 들어가고 DB에서 데이터가 전달된다. 이때 관리자는 사용자와 달리 총 적립금을 포함하여 거래에 대한 모든 정보를 확인할 수 있다.

## 12) 중고거래 기능 - 실시간 거래현황 조회

관리자는 실시간으로 거래현황에 대한 정보를 조회할 수 있다. 실시간 거래현황 조회 기능을 구현하기 위해 아래와 같이 설계한다.

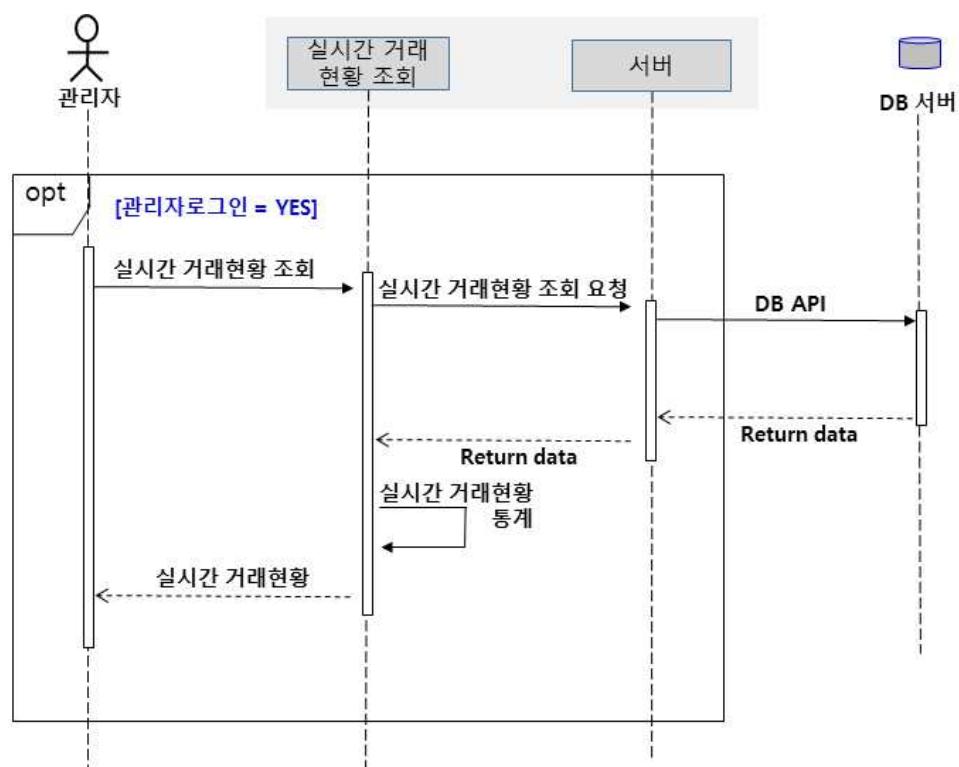


그림 3.36 실시간 거래현황 조회의 Sequence Diagram

그림 3.36를 보면 관리자는 관리자 인증을 성공한 후에 실시간으로 거래현황을 조회한다. 조회 요청이 서버로 들어가면 DB에서 해당하는 데이터를 받아오고 해당 데이터는 관리자가 파악하기 편한 형태로 가공하여 Web UI를 통해 전달한다. 관리자는 실시간으로 거래현황을 파악할 수 있다.

### 13) 중고거래 기능 - 누적 거래현황 조회

관리자는 누적 거래현황에 대한 정보를 조회할 수 있다. 누적 거래현황 조회 기능을 구현하기 위해 아래와 같이 설계한다.

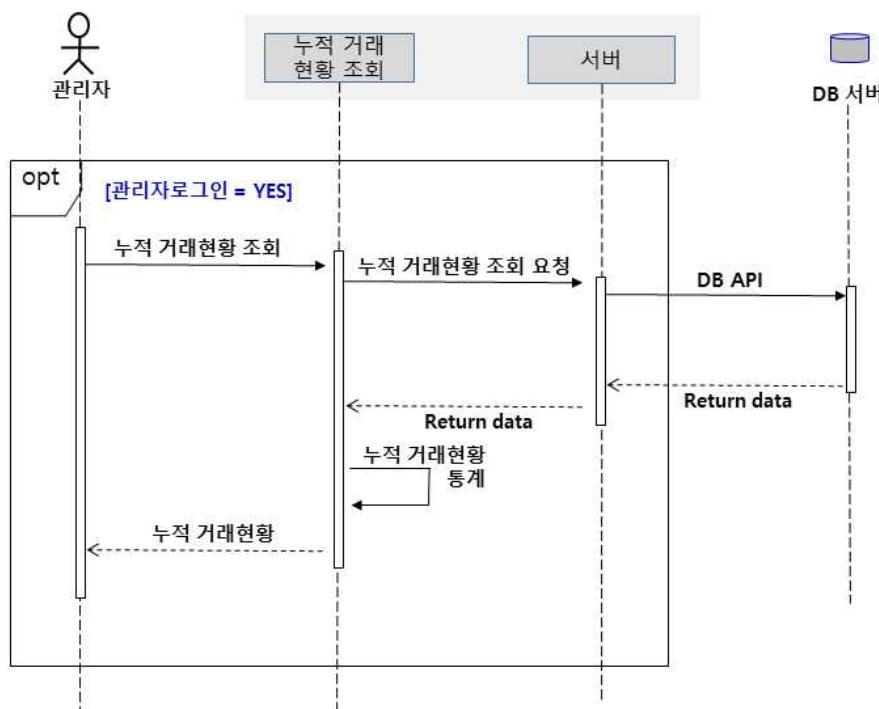


그림 3.37 누적 거래현황 조회의 Sequence Diagram

그림 3.37를 보면 관리자는 관리자 인증을 성공한 후에 누적 거래현황을 조회한다. 조회 요청이 서버로 들어가면 DB에서 해당하는 데이터를 받아오고 해당 데이터는 관리자가 파악하기 편한 형태로 가공하여 Web UI를 통해 전달한다. 많은 양의 누적 거래 데이터를 통계, 분석하여 관리자에게 시각화된 데이터로 제공한다. 관리자는 Web UI를 이용해 누적 거래현황을 파악할 수 있다.

#### 14) 중고거래 기능 - 회원신고 관리

관리자는 회원신고에 대한 정보를 조회할 수 있다. 사용자가 작성한 신고 게시글을 확인하여 관리자의 판단에 따라 처리할 수 있다. 회원신고 관리 기능을 구현하기 위해 아래와 같이 설계한다.

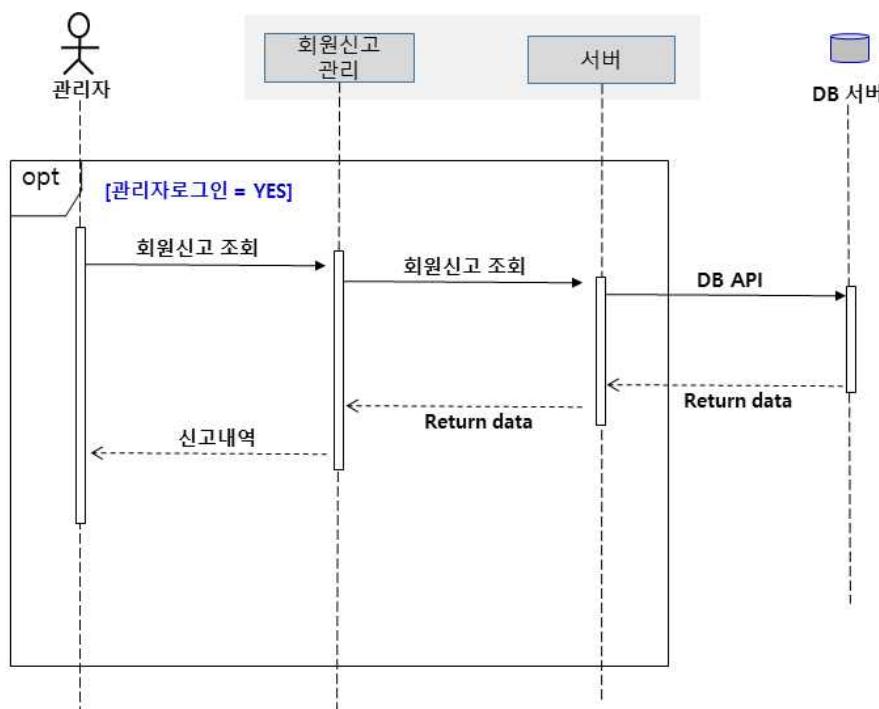


그림 3.38 회원신고 관리의 Sequence Diagram

그림 3.38를 보면 관리자는 관리자 인증을 성공한 후에 회원신고 관리 기능을 사용한다. 회원신고 내역을 조회하고 해당 데이터를 서버에서 DB에 접근해 출력해준다. 관리자는 신고 내역을 확인하고 회원정지, 회원등급 관리 등의 제재를 가할 수 있다.

### 15) 수익관리 - 수수료 조회

관리자는 중고거래로 발생한 수수료에 대해 파악할 수 있다. 수수료 조회 기능을 구현하기 위해 아래와 같이 설계한다.

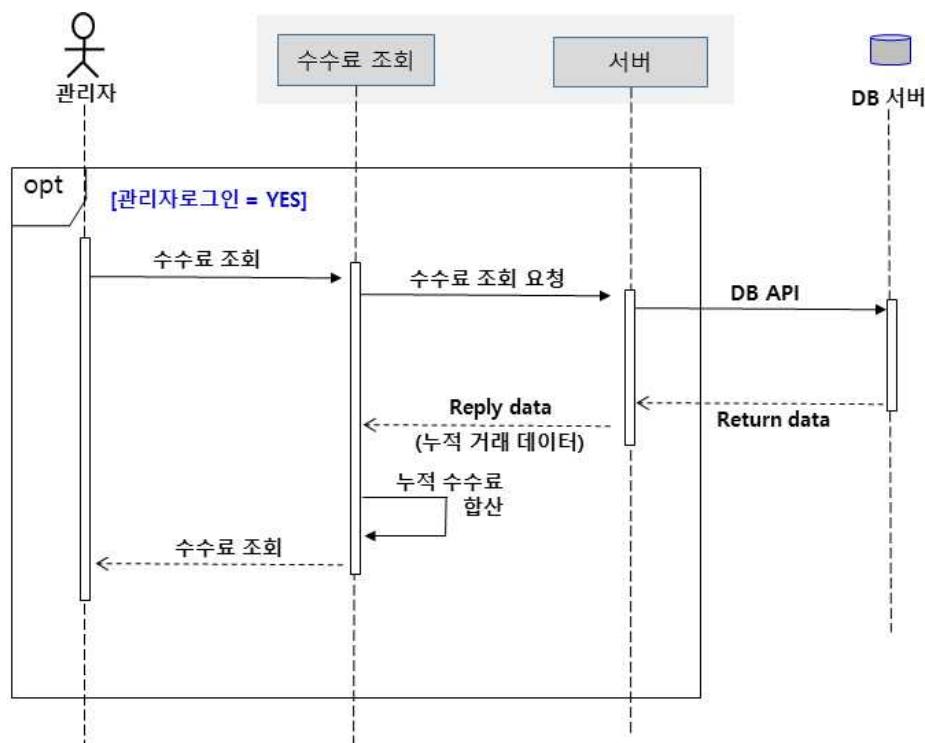


그림 3.39 수수료 조회의 Sequence Diagram

그림 3.39를 보면 관리자는 관리자 인증을 성공한 후에 수수료 조회 기능을 사용한다. 관리자가 수수료 조회를 요청하면 서버에서 수수료에 대한 정보를 DB에서 받아오고 합산하여 누적 수수료를 관리자에게 전달한다. 관리자는 Web UI를 통해 시각화된 수수료 정보를 확인할 수 있다.

## 16) 수익관리 - 매출총액 조회

관리자는 중고거래로 발생한 매출액 대해 파악할 수 있다. 매출총액 조회 기능을 구현하기 위해 아래와 같이 설계한다.

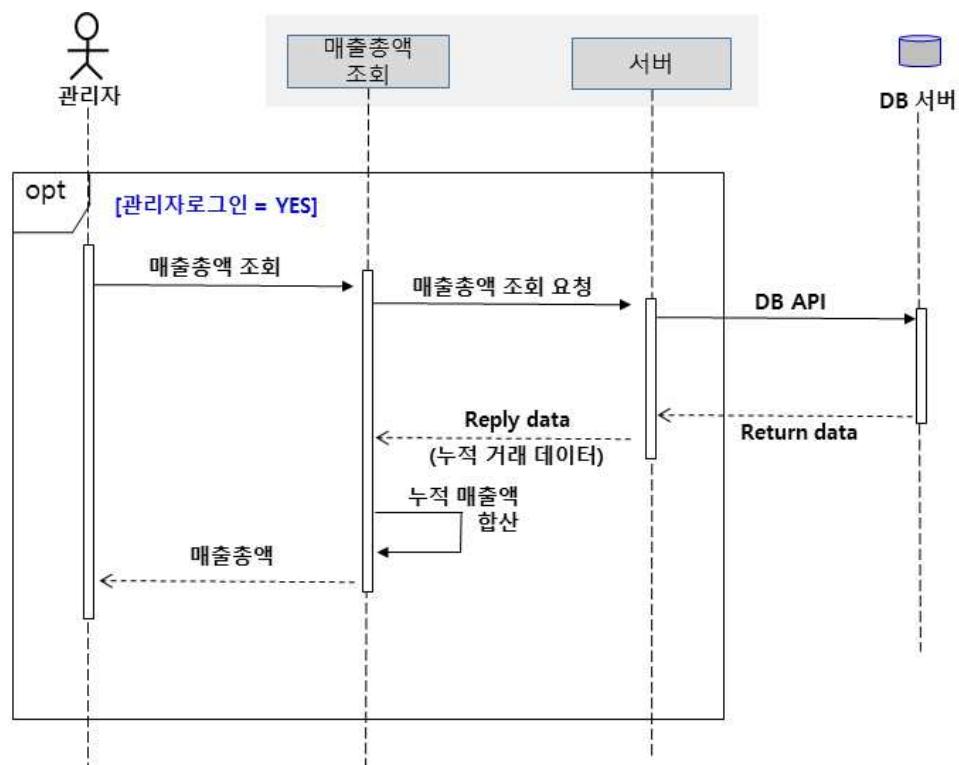


그림 3.40 매출총액 조회의 Sequence Diagram

그림 3.40를 보면 관리자는 관리자 인증을 성공한 후에 매출총액 조회 기능을 사용한다. 관리자가 매출총액 조회를 요청하면 서버에서 해당하는 정보를 DB에서 받아오고 합산하여 누적 매출액을 관리자에게 전달한다. 관리자는 Web UI를 통해 시각화된 매출 정보를 확인할 수 있다.

## 4. 화면(UI) 설계

### 4.1. 스토리보드(메뉴) 구성

#### ■ 사용자

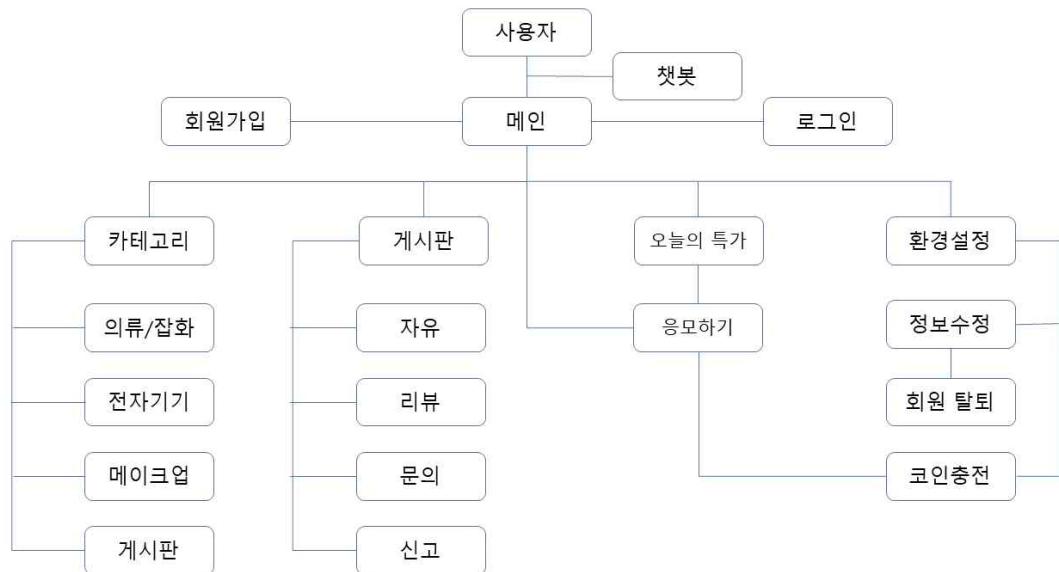


그림 4.1 사용자용 메뉴구조도

#### ■ 관리자

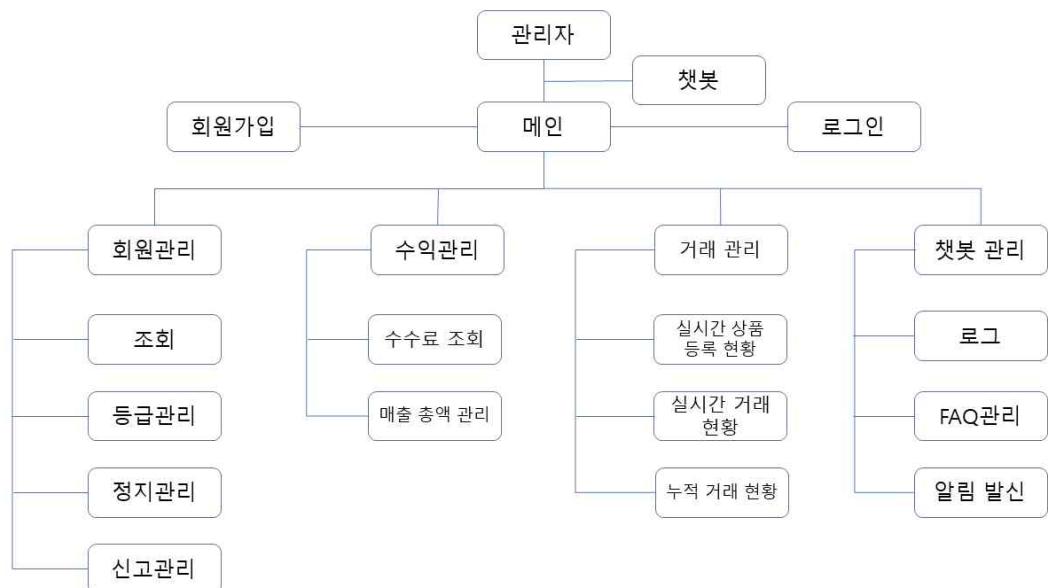
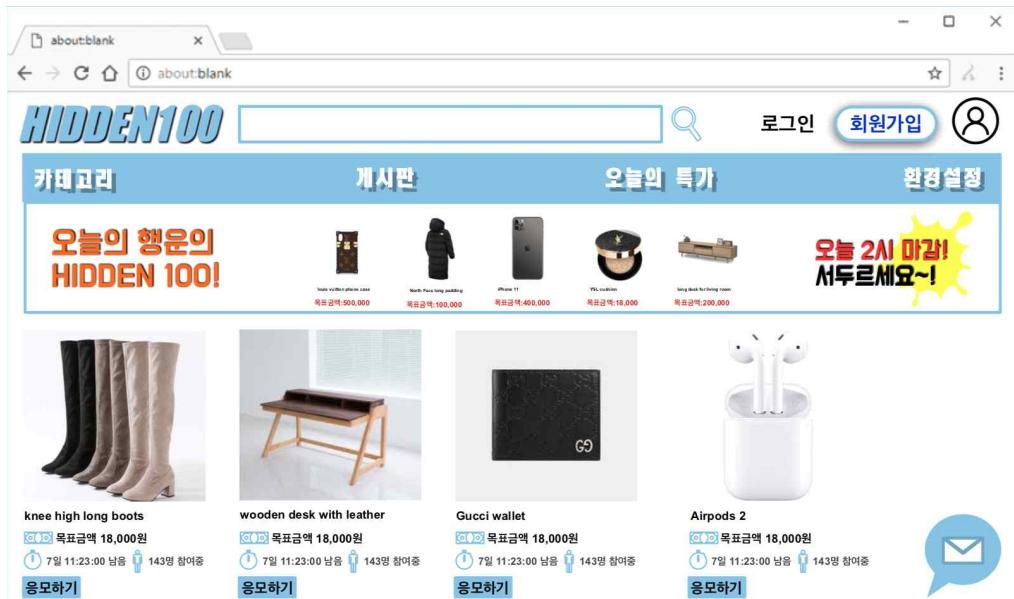


그림 4.2 관리자용 메뉴 구조도

## 4.2. 초기화면(로그인 전화면) 설계

### ■ 초기 화면(로그인 전)



- 웹에 접속하면 가장 먼저 보이는 메인 페이지이다.
- 네비게이션 바에서 카테고리, 게시판, 오늘의 특가, 환경설정 기능을 선택할 수 있다.
- 오른쪽 상단의 로그인, 회원가입을 통해 사용자 인증이 가능하다.
- 인기응모제품들이 메인 페이지 상단에 보여진다.
- 오른쪽 하단의 메시지 버튼을 클릭하여 챗봇 기능을 사용할 수 있다.
- 최상단의 네모박스에서 찾고자 하는 제품을 검색할 수 있다.
- 제품 하단의 돈 아이콘 옆의 목표금액은 응모 성사를 위한 목표금액이다. 만약 총 응모 금액이 목표금액을 넘지 못할 시에는 응모 성사가 이루어지지 않게 된다. 이는 실패한 응모로 간주되고 총 응모금액은 관리자가 소유한다.
- 돈 아이콘 하단의 시계 아이콘은 응모 가능한 남은 시간이다. 시간이 0이 되면 응모는 더 이상 불가능하다.
- 시계 아이콘 오른편의 사람 아이콘은 해당 상품에 응모한 총 참여자수이다.
- 사용자는 상품 최하단의 파란색 응모하기 버튼을 클릭하여 응모에 참여할 수 있다. ━

### 4.3. 사용자UI 설계

#### ■ 회원가입 초기 화면

The screenshot shows a web browser window for 'about:blank'. The main content area displays the 'Hidden 100' logo and a search bar. On the right side, there are buttons for '로그인' (Login) and '회원가입' (Member Registration), with '회원가입' being highlighted. Below this, a navigation menu includes '카테고리' (Category), '의류/잡화' (Clothing/General Goods), '게시판' (Board), '오늘의 특가' (Today's Specials), and '환경설정' (Environment Settings). The central part of the page features a title '3초만에 회원가입하기' (Join in 3 seconds) and a link '로그인 하러가기' (Go to login). It contains three input fields: 'ID' (아이디를 입력해주세요), 'PW' (비밀번호를 입력해주세요), and 'PW확인' (비밀번호를 다시 입력해주세요). A blue button labeled '완료' (Complete) is at the bottom left, and a blue speech bubble icon with an envelope symbol is at the bottom right.

- 회원가입을 누르면 나타나는 화면이다.
- 사용자는 이 화면에서 회원가입을 할 수 있다.

■ 회원 가입 화면(텍스트 창이 공란일 때)

The screenshot shows a web browser window with the URL 'about:blank'. The main content is a registration form titled '3초만에 회원가입하기' (Join in 3 seconds). The form includes fields for ID, PW, and PW확인 (PW confirmation), each containing placeholder text ('아이디를 입력해주세요', '\*\*\*\*\*', '\*\*\*\*\*'). A button labeled '완료' (Complete) is at the bottom. On the right side of the form, there is a blue speech bubble icon with an envelope symbol. At the top right of the page, there are links for '로그인' (Login) and '회원가입' (Registration), with '회원가입' being highlighted.

- 텍스트 창이 공란이면 경고 문구가 나타난다.

■ 회원 가입 화면(아이디 중복 확인)

The screenshot shows the same registration form as the previous one, but with a different outcome. The ID field now contains 'sonsin', which is highlighted in red. A message next to it reads '사용 불가능한 아이디입니다.' (This ID is unavailable). All other fields (PW and PW확인) are empty. The '완료' button is present at the bottom. The '회원가입' link at the top right is also highlighted.

- 아이디를 입력하고 아이디 중복 체크를 한다.
- 이미 사용 중인 아이디라면 ‘사용 불가능한 아이디입니다’ 경고 문구가 나타난다.

■ 회원 가입 화면(아이디 중복 확인)

**3초만에 회원가입하기**

ID: **soonsin** 아이디 중복 체크 사용 가능한 아이디입니다.

PW: 비밀번호를 입력해주세요

PW확인: 비밀번호를 다시 입력해주세요

**완료**

- 아이디를 입력하고 아이디 중복 체크를 한다.
- 미사용 중인 아이디라면 ‘사용 가능한 아이디입니다’ 문구가 나타난다.

■ 회원 가입 화면(입력한 비밀번호와 비밀번호 확인이 불일치할 때)

**3초만에 회원가입하기**

ID: **soonsin** 아이디 중복 체크 사용 가능한 아이디입니다.

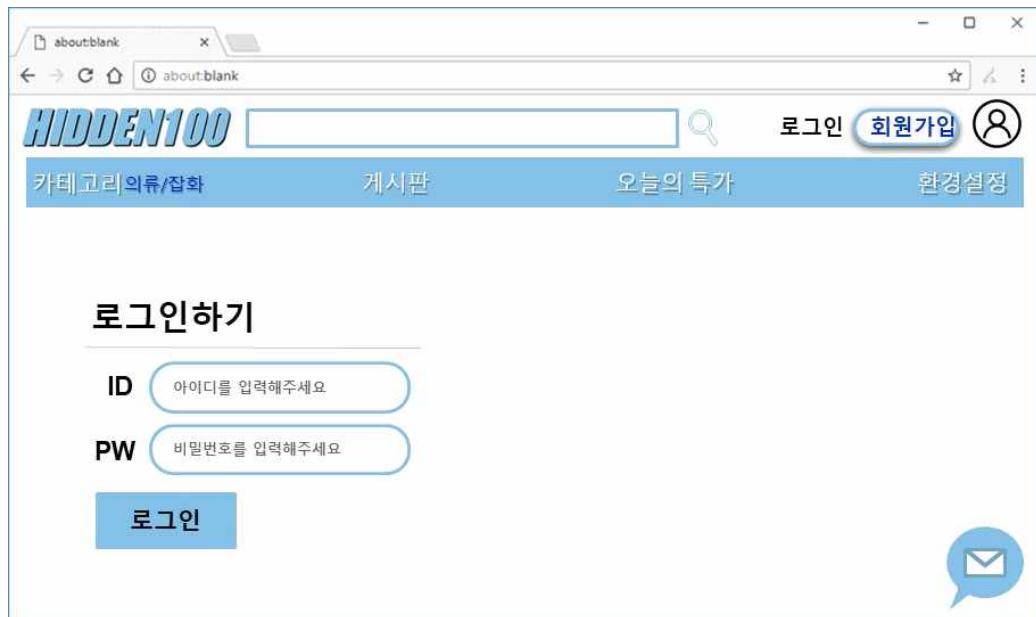
PW: \*\*\*\*\*

PW확인: \*\*\*\*\* 비밀번호가 일치하지 않습니다.

**완료**

- 비밀번호와 비밀번호 확인을 입력하고 완료 버튼을 클릭한다.
- 두 데이터가 불일치하면 ‘비밀번호가 일치하지 않습니다’ 경고 문구가 나타난다.

### ■ 로그인 초기 화면



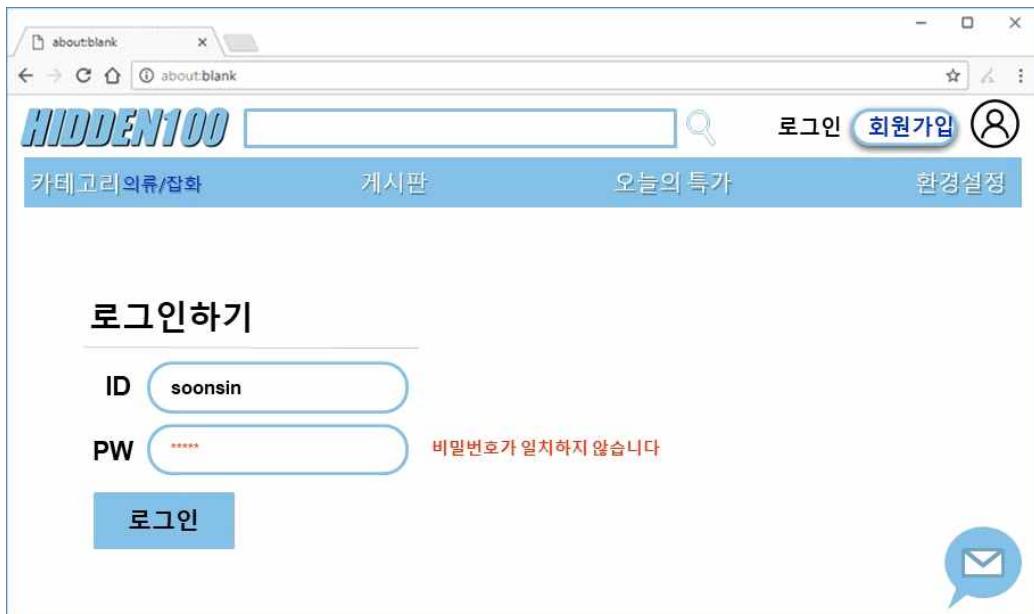
- 사용자가 로그인을 하는 화면이다.
- 아이디와 비밀번호를 입력해서 로그인한다.

### ■ 로그인 화면(텍스트 창이 공란일 때)



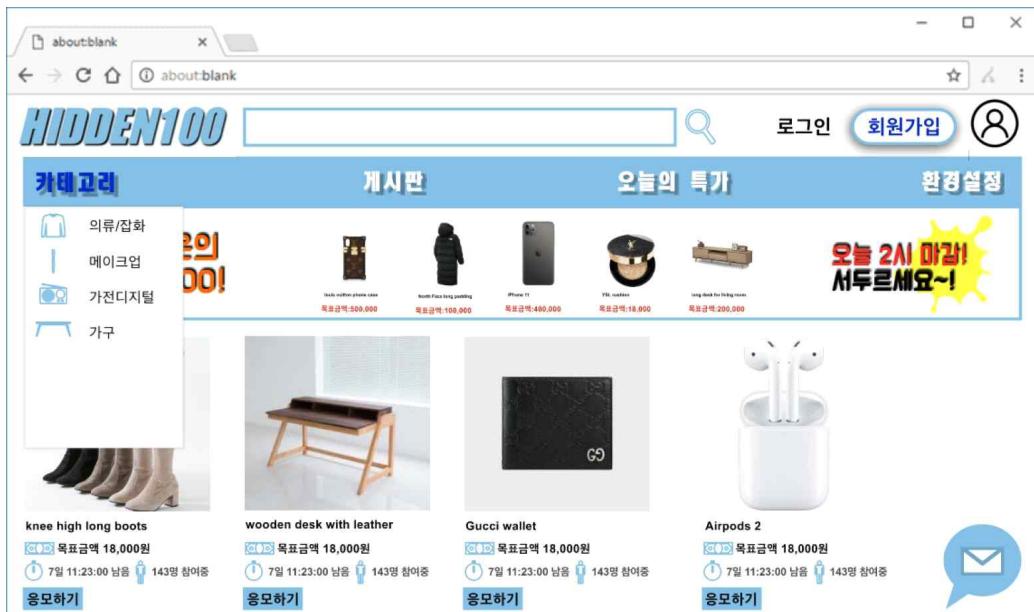
- 아이디 텍스트 창이 공란이면 ‘아이디가 입력되지 않았습니다’ 경고 문구가 나타난다.
- 비밀번호 텍스트 창이 공란이면 ‘비밀번호가 입력되지 않았습니다’ 경고 문구가 나타난다.

- 로그인 화면(텍스트 창의 데이터와 데이터베이스의 데이터가 불일치할 때)

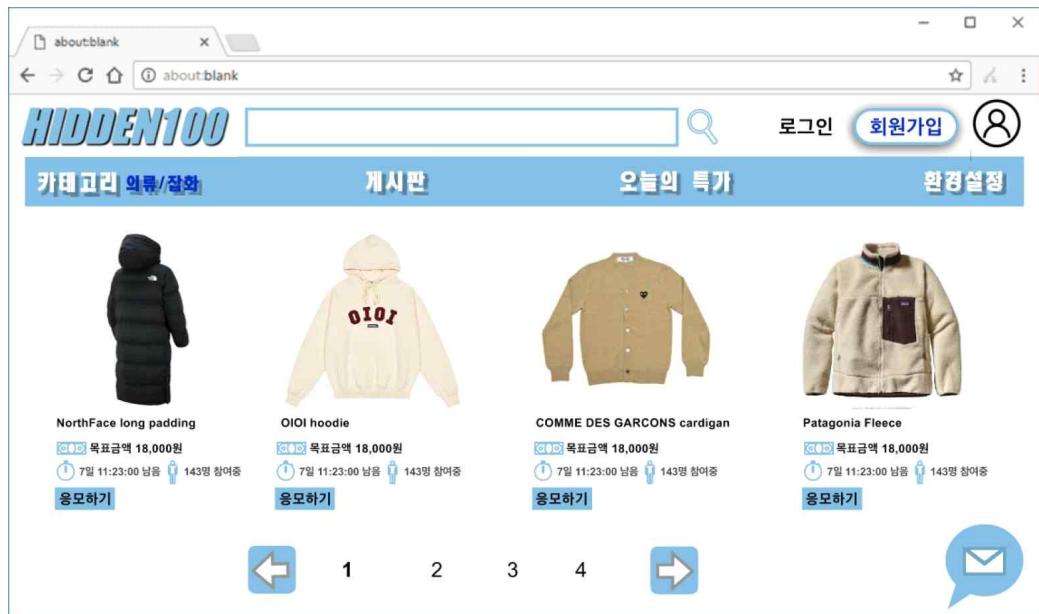


- 아이디 텍스트 창의 데이터와 데이터베이스의 데이터가 불일치하면 ‘아이디가 일치하지 않습니다’ 경고 문구가 나타난다.
- 비밀번호 텍스트 창의 데이터와 데이터베이스의 데이터가 불일치하면 ‘비밀번호가 일치하지 않습니다’ 경고 문구가 나타난다.

- 초기화면(네비게이션 바 마우스 호버링)



- 메인 페이지에서 네비게이션 바의 ‘카테고리’ 분류에 마우스를 호버링한 상태이다.
- 각 분류에 마우스를 호버링하면 폰트색이 변경되면서 드롭다운 메뉴가 나타난다.

**■ 의류/잡화 판매 페이지**

- ‘카테고리’에서 ‘의류/잡화’를 선택한 페이지이다.
- ‘카테고리’ 옆에 ‘의류/잡화’ 텍스트가 나타난다.
- 사용자는 해당 페이지에서 물건판매를 할 수 있다.
- 사용자는 해당 페이지에서 물건등록을 할 수 있다.
- 페이지네이션 기능을 이용하여 페이지 이동이 가능하다.

■ 상품 페이지



- 판매자가 등록한 상품의 이미지, 이름, 목표금액, 시간이 나타난다.
- 구매자가 응모할수록 응모 참여자수가 증가한다.
- 구매자가 응모하기 버튼을 클릭하면 상품에 응모할 수 있다.

■ 상품 페이지(로그인 상태가 아닐 때)



- 로그인 상태가 아닌 경우 응모하기 버튼을 클릭하면 '로그인이 필요한 서비스입니다' 경고 메시지가 나타난다.
- 사용자는 로그인 상태인 경우에만 상품 응모가 가능하다.

### ■ 응모 페이지



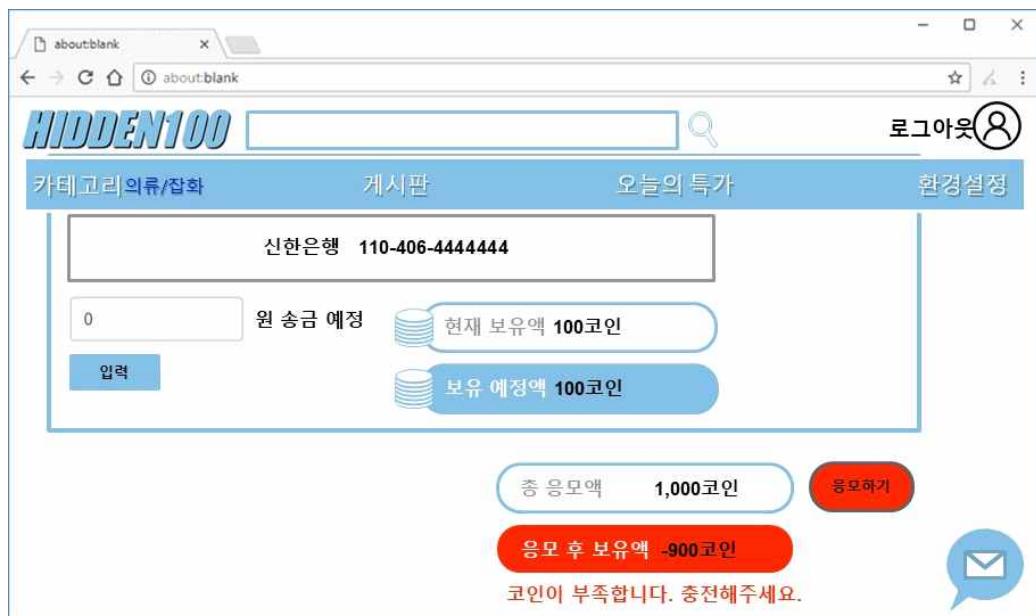
- 기본 상품응모횟수는 1번이다.
- 상품응모횟수를 변경하면 응모금액이 자동으로 업데이트되고 응모금액 부분에 변경된 응모금액이 나타난다.

### ■ 응모 페이지(응모금액 변경)



- 상품응모금액을 변경하면 총 응모액과 응모 후 보유액이 업데이트된다.
- 구매자는 노란색의 응모하기 버튼을 클릭하여 상품응모가 가능하다.

■ 응모 페이지(현재 보유액이 부족한 경우)



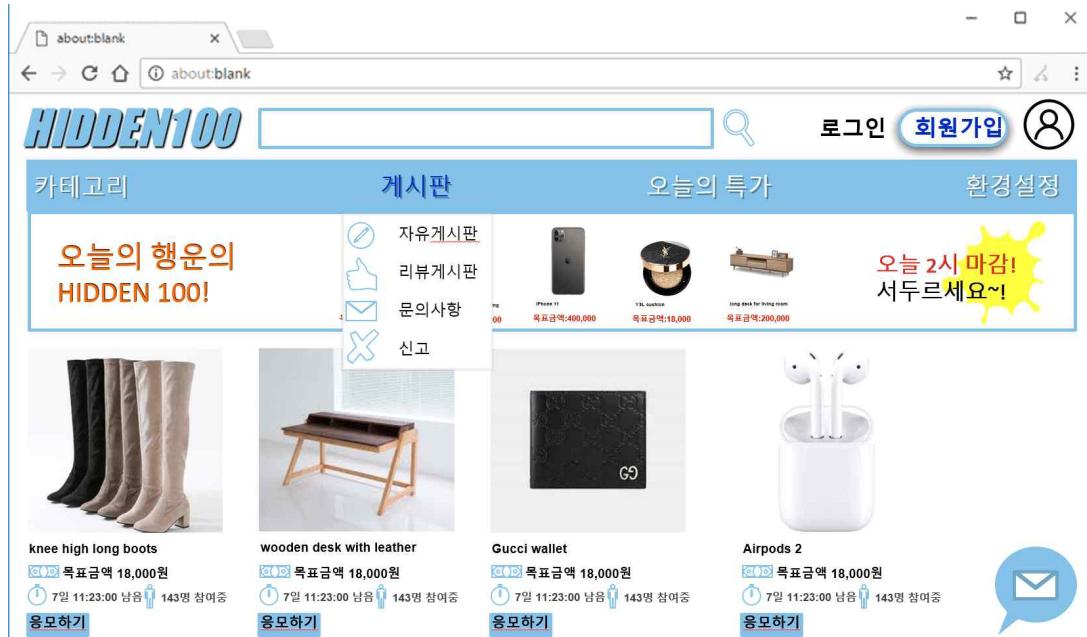
- 현재 보유액이 총 응모액보다 부족하면 응모 후 보유액이 음수로 변한다.
- 현재 보유액이 총 응모액보다 부족하면 ‘코인이 부족합니다. 충전해주세요’ 경고 문구가 나타난다.
- 사용자는 코인을 충전해서 현재 보유액을 증가시킬 수 있다.

■ 응모 페이지(응모에 성공한 경우)



- 현재 보유액이 총 응모액보다 높아 응모에 성공한 경우 ‘축하합니다. 응모에 성공하셨습니다!’ 문구가 화면 상단에 나타난다.
- 사용자는 화면 하단의 다른 응모 보러 가기 버튼을 클릭하여 등록되어 있는 다른 상품들을 볼 수 있다.

■ 초기화면(네비게이션 바 마우스 호버링)



- 메인 페이지에서 네비게이션 바의 ‘게시판’ 분류에 마우스를 호버링한 상태이다.
- 각 분류에 마우스를 호버링하면 폰트색이 변경되면서 드롭다운 메뉴가 나타난다.

## ■ 자유게시판 화면

번호	제목	작성자	작성일	조회수
1	나 당첨 좀 시켜주라	이우석	2019-11-30	0
2	노트북 급처합니다!!!!	한혜경	2019-10-20	40
3	후드티 팔아요	이다은	2019-10-30	129
4	아이폰 6s 삽니다~~~~.	서지연	2019-09-20	342

- ‘게시판’에서 ‘자유게시판’을 선택한 페이지이다.
- ‘게시판’ 옆에 ‘자유게시판’ 텍스트가 나타난다.
- 챗봇 링크의 위치가 화면 하단에서 중간으로 이동한다.
- 사용자는 글쓰기 버튼을 클릭해서 게시판에 글작성이 가능하다.
- 사용자는 텍스트 창에 단어를 입력해서 원하는 글을 검색할 수 있다.
- 페이지네이션 기능을 이용하여 페이지 이동이 가능하다.

■ 자유게시판 화면(로그인 상태가 아닌 경우 글쓰기 버튼을 눌렀을 때)

The screenshot shows the 'HIDDEN 100' website's free classifieds section. At the top right, there are '로그인' (Login) and '회원가입' (Sign Up) buttons. Below the header, there's a search bar and a '확인' (Check) button. The main content area displays a list of posts under the heading '오늘의 행운의 HIDDEN 100!' (Today's Fortune of HIDDEN 100!). The first post is '나 당첨 좀 시켜주라'. A large yellow speech bubble on the right says '오늘 2시 마감! 서두르세요~!' (Deadline at 2 PM today! Hurry up!). At the bottom, there's a navigation bar with page numbers 1, 2, 3, 4 and a '글쓰기' (Write) button.

번호	제목	작성자	작성일	조회수
1	나 당첨 좀 시켜주라	이우석	2019-11-30	0
2	노트북 급처합니다!!!!	한혜경	2019-10-20	40
3	후드티 팔아요	이다은	2019-10-30	129
4	아이폰 6s 삽니다~~~.	서지연	2019-09-20	342

- 사용자가 로그인 상태가 아닌 경우 글쓰기 버튼을 클릭하면 ‘로그인이 필요한 서비스입니다’ 경고 메시지가 나타난다.
- 사용자는 로그인을 한 상태에서만 글쓰기가 가능하다.

■ 자유게시판 화면(로그인 상태인 경우)

The screenshot shows the same website structure as above, but for logged-in users. The '로그인' button has been replaced by a '로그아웃' (Logout) button. The rest of the interface, including the classifieds list and the '글쓰기' button, remains identical.

번호	제목	작성자	작성일	조회수
1	나 당첨 좀 시켜주라	이우석	2019-11-30	0
2	노트북 급처합니다!!!!	한혜경	2019-10-20	40
3	후드티 팔아요	이다은	2019-10-30	129
4	아이폰 6s 삽니다~~~.	서지연	2019-09-20	342

- 사용자가 로그인 상태인 경우 글쓰기 버튼을 클릭하면 게시판 글작성이 가능하다.
- 화면 오른쪽 상단의 ‘로그인/회원가입’이 ‘로그아웃’으로 변경된다.

## ■ 글쓰기 화면



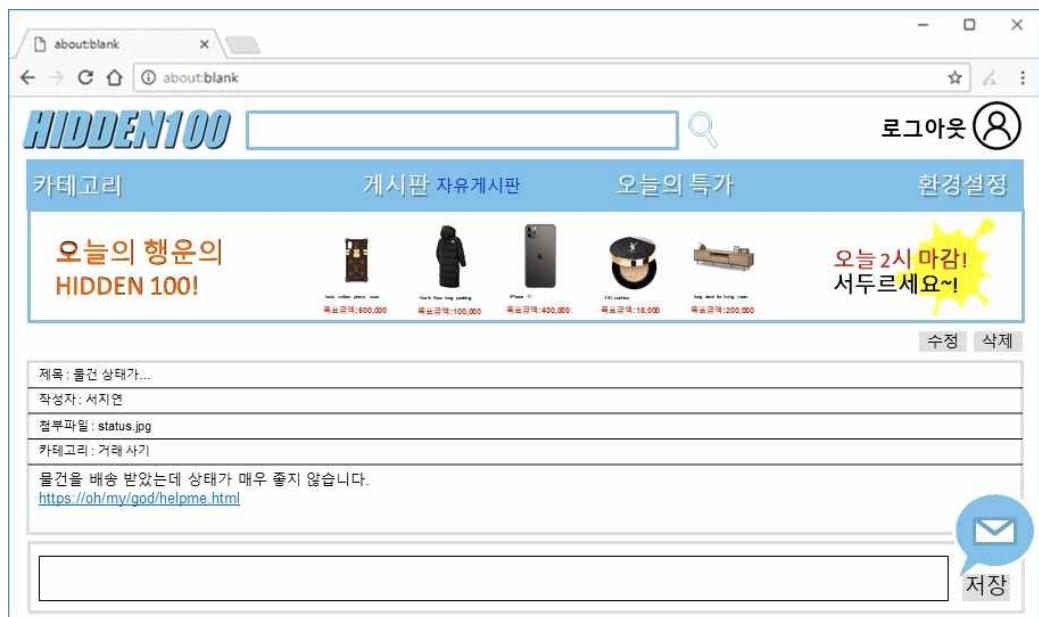
- ‘카테고리’에서 게시판 종류를 선택할 수 있다.
- 게시판은 자유게시판, 문의게시판, 신고게시판이 있다.
- 첨부파일은 ‘찾기’ 버튼을 클릭하여 추가할 수 있다.

## ■ 글쓰기 화면(내용이 공란인 상태에서 글작성을 시도했을 때)



- 내용이 빈 상태에서 ‘저장’ 버튼을 클릭하면 글작성이 거부되고 ‘내용을 입력해 주세요’ 경고 메시지가 나타난다.
- 경고 메시지의 ‘확인’ 버튼을 클릭하면 경고 메시지가 사라진다.
- 화면 하단의 ‘취소’ 버튼을 클릭하면 글작성이 취소되고 게시판 페이지로 돌아간다.

■ 글쓰기 화면(내용이 입력되고 글작성을 시도했을 때)



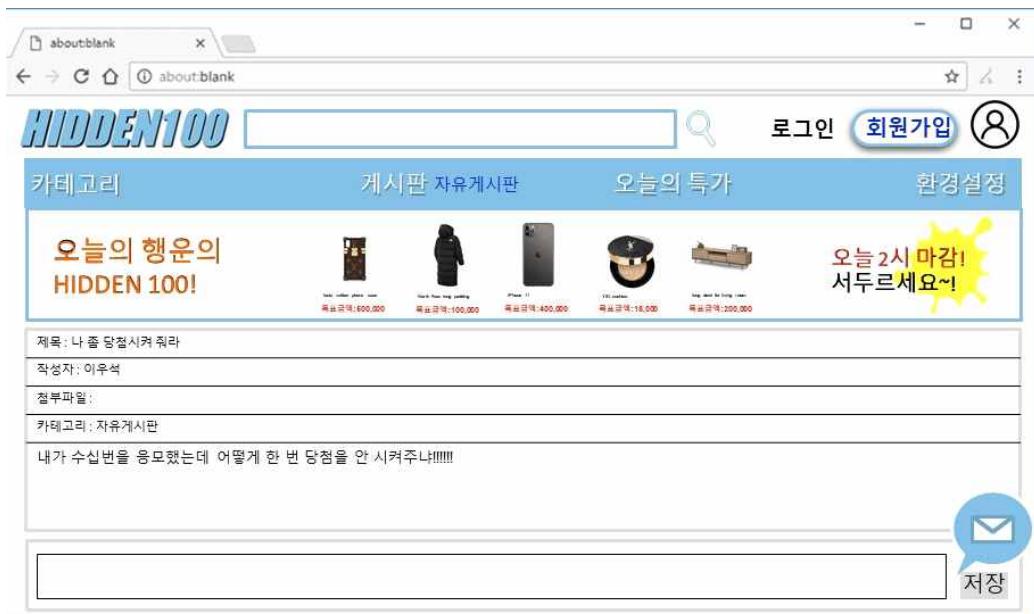
- 내용이 입력된 상태에서 '저장' 버튼을 클릭하면 글작성이 완료된다.
- 제목, 작성자, 카테고리, 내용에 데이터가 채워진다.
- 첨부파일을 추가하고 글작성을 하면 첨부파일이 나타나고 다운로드 가능하다.

### ■ 게시글 조회 화면(글 작성자인 경우)



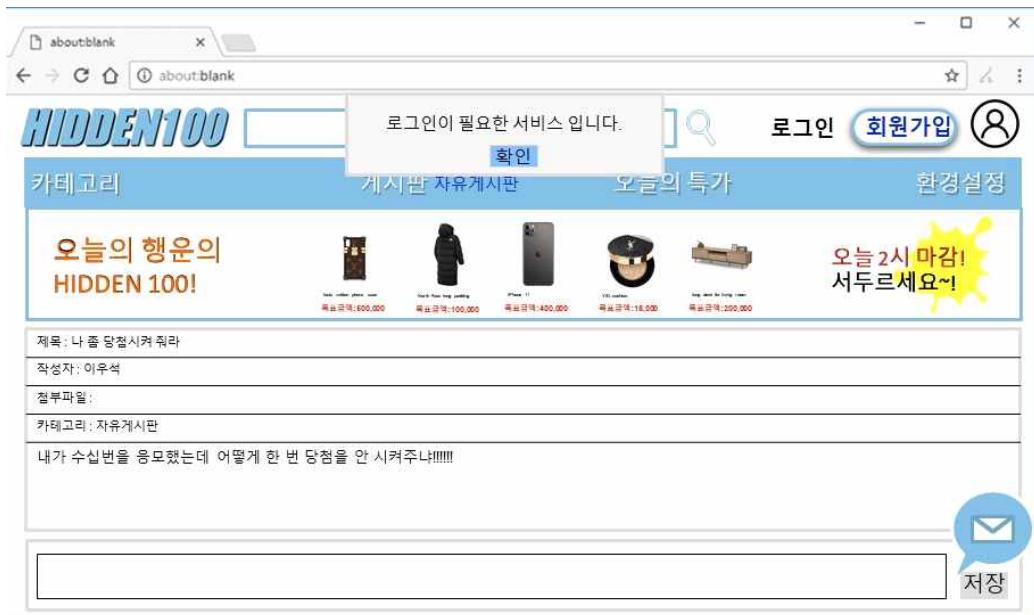
- 게시글 위편에 ‘수정’과 ‘삭제’ 버튼이 나타난다.
- 사용자는 댓글창에서 댓글을 입력할 수 있다.
- 챗봇 링크는 화면 하단에 위치한다.

### ■ 게시글 조회 화면(글 작성자가 아닌 경우)



- 게시글 위편에 ‘수정’과 ‘삭제’ 버튼이 나타나지 않는다.
- 사용자는 댓글창에서 댓글을 입력할 수 있다.
- 챗봇 링크는 화면 하단에 위치한다.

### ■ 댓글 입력 화면(로그인 상태가 아닌 경우)



- 댓글입력을 시도하면 ‘로그인이 필요한 서비스입니다’ 경고 메시지가 나타난다.
- 경고 메시지의 ‘확인’ 버튼을 클릭하면 로그인 화면으로 이동한다.
- 사용자는 로그인 상태인 경우에만 댓글작성이 가능하다.

### ■ 댓글 입력 화면(댓글 작성칸이 공란인 경우)



- 댓글 작성칸이 비어있는 경우 ‘저장’ 버튼을 클릭하면 ‘내용을 입력해 주세요’ 경고 메시지가 나타난다.
- 경고 메시지의 ‘확인’ 버튼을 클릭하면 현재 페이지에 머무른다.

### ■ 댓글 조회 화면(댓글 작성자인 경우)



- 사용자는 자신이 작성한 댓글에 대해서 수정 혹은 삭제가 가능하다.
- 사용자는 자신이 작성한 댓글 옆의 ‘수정’과 ‘삭제’ 버튼을 클릭하여 원하는 기능을 이용할 수 있다.

### ■ 댓글 조회 화면(댓글 작성자가 아닌 경우)



- 사용자는 자신이 작성한 댓글이 아니면 수정 혹은 삭제가 불가능하다.
- 사용자는 자신이 작성한 댓글이 아니면 ‘수정’과 ‘삭제’ 버튼을 볼 수 없다.

■ 게시물 검색 결과 화면(검색창이 공란인 경우)

번호	제목	작성자	작성일	조회수
1	나 당첨 좀 시켜주세요	이우석	2019-11-30	0
2	노트북 급처합니다!!!	한혜경	2019-10-20	40
3	후드티 팔아요	이다은	2019-10-30	129
4	아이폰 6s 삽니다~~~~~	서지연	2019-09-20	234

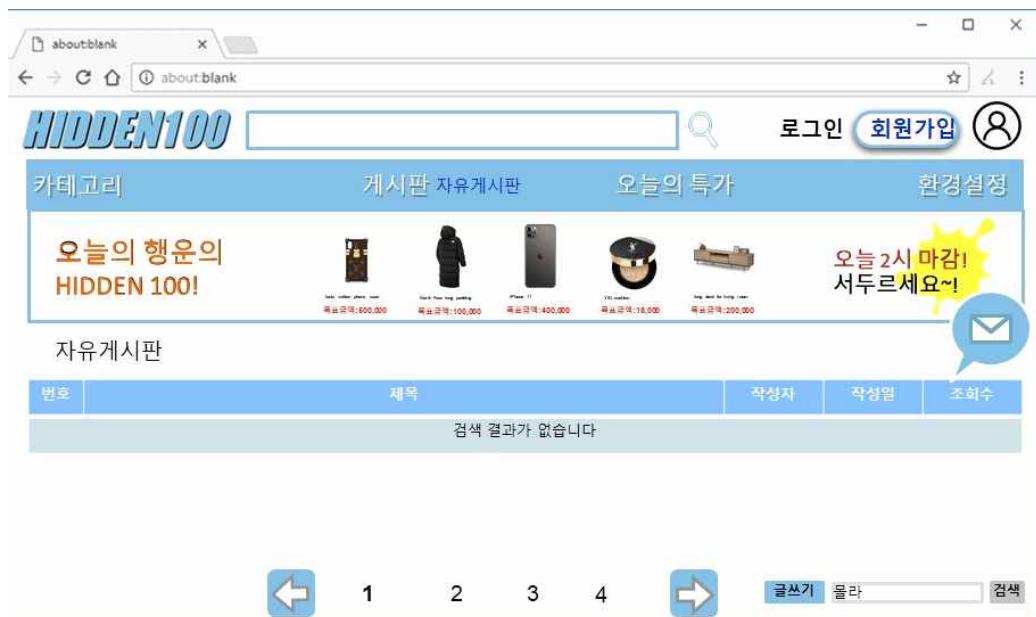
- 검색창이 비어있는 상태에서 ‘검색’ 버튼을 클릭하면 ‘한 글자 이상 입력해주세요’ 경고 메시지가 나타난다.
- 경고 메시지의 ‘확인’ 버튼을 클릭하면 현재 페이지에 머무른다.

■ 게시물 검색 결과 화면(일치하는 게시글이 존재하는 경우)

번호	제목	작성자	작성일	조회수
1	아이폰 6s 삽니다~~~~~	이우석	2019-11-30	0
2	아이폰 내놓습니다.	한혜경	2019-10-20	40
3	아이폰 중고로 살까 신상 살까	이다은	2019-10-30	129
4	요즘 아이폰 별로인 듯	서지연	2019-09-20	234

- 검색창에 입력한 단어와 일치하는 게시글이 존재하면 일치하는 게시글만 보여준다.
- 검색한 단어와 일치하는 단어에는 강조 처리해준다.

- 게시물 검색 결과 화면(일치하는 게시글이 존재하지 않는 경우)



- 검색창에 입력한 단어와 일치하는 게시글이 존재하지 않으면 '검색 결과가 없습니다'가 나타나고 아무 게시글도 보여지지 않는다.

## ■ 리뷰게시판

번호	제목	작성자	작성일	조회수
1	단돈 5000원에 아이폰11pro 득템ㅎㅎ	이우석	2019-11-30	0
2	홀길동 판매자님 감사합니다😊	한혜경	2019-10-20	40
3	10000원에 룽패딩 샀다:)	이다은	2019-10-30	129
4	드디어 첫 당첨!!!!!!신 이시어!!!!!!!	서지연	2019-09-20	324

- ‘게시판’에서 ‘리뷰게시판’을 선택한 페이지이다.
- ‘게시판’ 옆에 ‘리뷰게시판’ 텍스트가 나타난다.
- 화면 중앙의 리뷰게시판 옆에 ‘리뷰 남기면 포인트 지급!!!’ 문장이 보여진다.
- 챗봇 링크의 위치가 화면 하단에서 중간으로 이동한다.
- 사용자는 글쓰기 버튼을 클릭해서 게시판에 글작성이 가능하다.
- 사용자는 텍스트 창에 단어를 입력해서 원하는 글을 검색할 수 있다.
- 페이지네이션 기능을 이용하여 페이지 이동이 가능하다.

## ■ 문의게시판

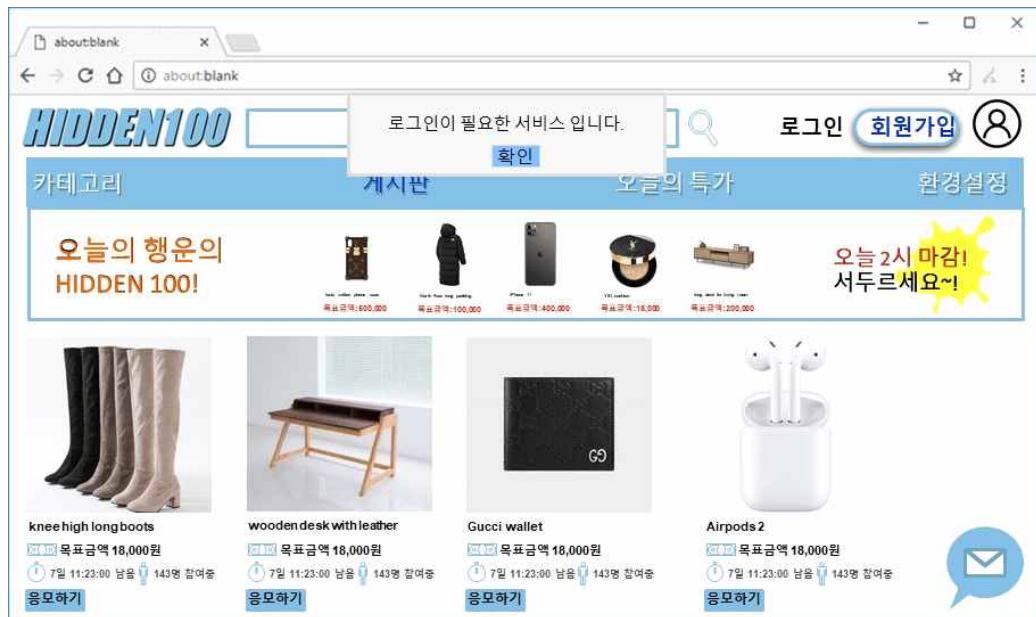
번호	제목	작성자	작성일	조회수
1	배송 일자 문의	이우석	2019-11-30	0
2	환불 되나요???	한혜경	2019-10-20	40
3	배송 조회 좀.....	이다은	2019-10-30	129
4	회원 탈퇴 방법??	서지연	2019-09-20	234

- ‘게시판’에서 ‘문의게시판’을 선택한 페이지이다.
- ‘게시판’ 옆에 ‘문의게시판’ 텍스트가 나타난다.
- 챗봇 링크의 위치가 화면 하단에서 중간으로 이동한다.
- 사용자는 글쓰기 버튼을 클릭해서 게시판에 글작성이 가능하다.
- 사용자는 텍스트 창에 단어를 입력해서 원하는 글을 검색할 수 있다.
- 페이지네이션 기능을 이용하여 페이지 이동이 가능하다.

## ■ 신고게시판

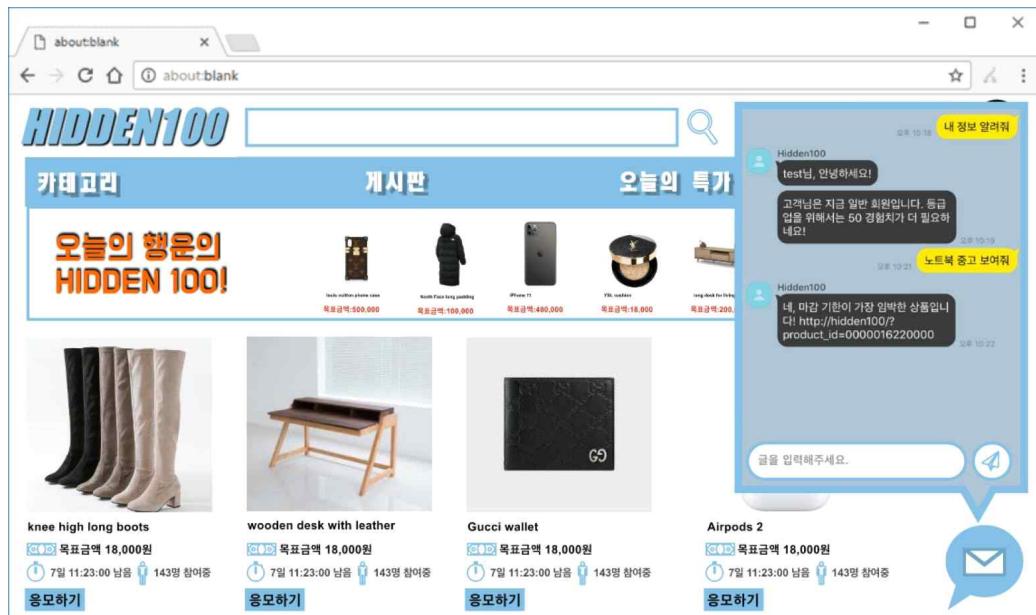
번호	제목	작성자	작성일	조회수
1	돈 떼임ㄷㄷ	이우석	2019-11-30	0
2	신고합니다.	한혜경	2019-10-20	40
3	물건이 안 와요	이다은	2019-10-30	129
4	물건 상태가.....	서지연	2019-09-20	234

- ‘게시판’에서 ‘신고게시판’을 선택한 페이지이다.
- ‘게시판’ 옆에 ‘거래 사기 신고’ 텍스트가 나타난다.
- 챗봇 링크의 위치가 화면 하단에서 중간으로 이동한다.
- 사용자는 글쓰기 버튼을 클릭해서 게시판에 글작성이 가능하다.
- 사용자는 텍스트 창에 단어를 입력해서 원하는 글을 검색할 수 있다.
- 페이지네이션 기능을 이용하여 페이지 이동이 가능하다.

**■ 챗봇 사용 화면(로그인 상태가 아닌 경우)**

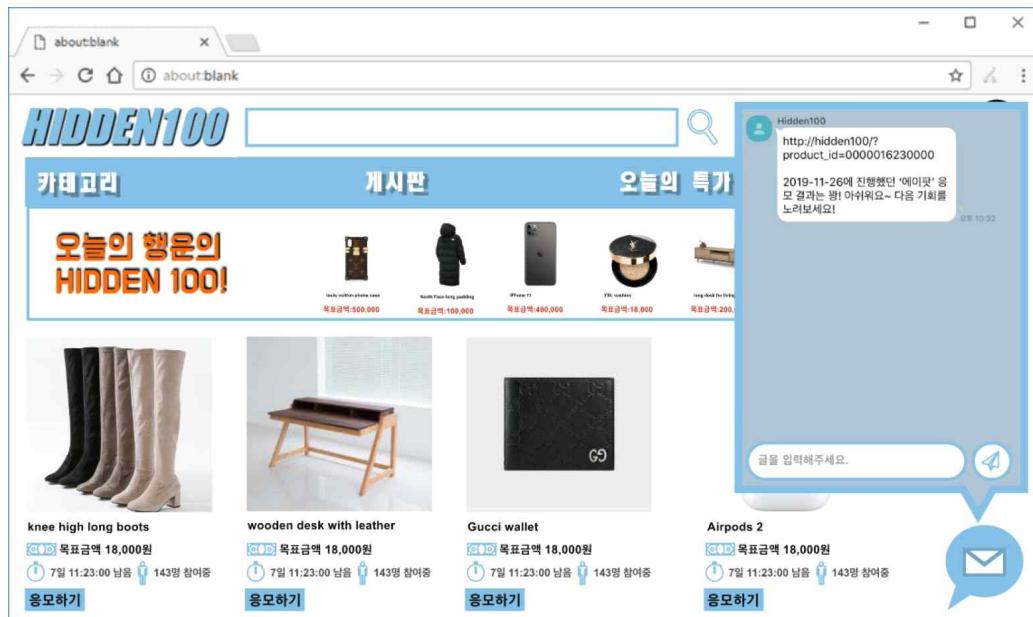
- 사용자가 로그인 상태가 아닌 경우 챗봇 서비스 이용을 시도하면 ‘로그인이 필요한 서비스입니다’ 경고 메시지가 나타난다.
- 사용자는 로그인 상태에서만 챗봇 서비스를 이용할 수 있다.

## ■ 챗봇 사용 화면(로그인 상태인 경우)



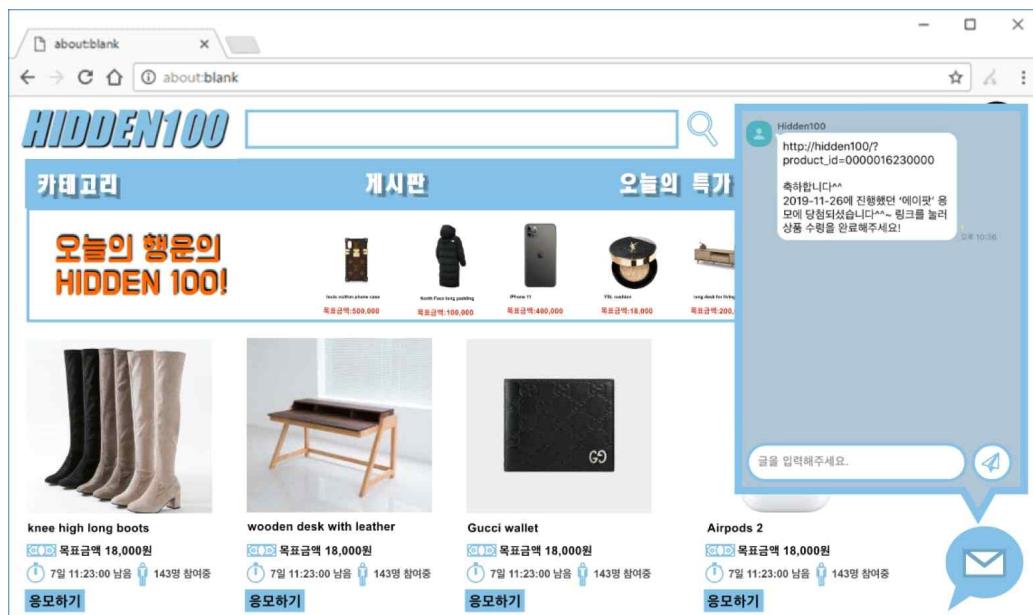
- 챗봇 서비스를 클릭하면 화면과 같은 UI가 나타난다.
- 텍스트 칸에 텍스트를 입력하고 메시지를 전송하면 메시지 피드가 업데이트 된다.
- 챗봇은 사용자와 소통이 가능하다.
- 챗봇은 사용자의 요청에 따른 적절한 응답을 메시지 피드에 보여준다.

## ■ 챗봇 사용 화면(미당첨인 경우)



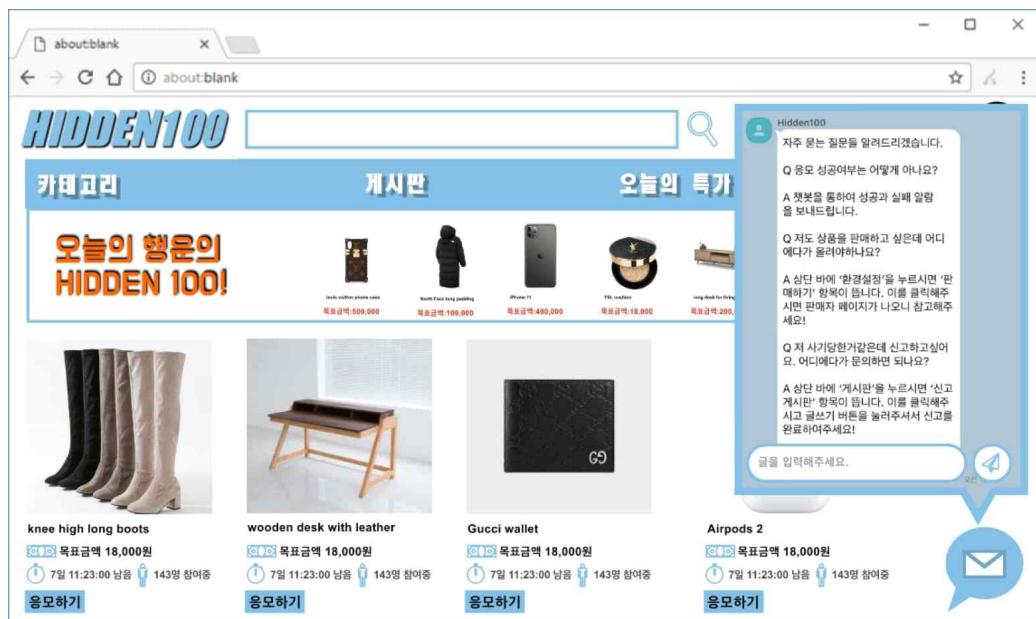
- 미당첨 응모결과가 챗봇 메시지 피드에 나타난다.
- 챗봇이 전송한 메시지를 통해 사용자는 어떤 상품에 응모한 결과인지 알 수 있다.

## ■ 챗봇 사용 화면(당첨인 경우)



- 당첨 응모결과가 챗봇 메시지 피드에 나타난다.
- 챗봇이 전송한 메시지를 통해 사용자는 어떤 상품에 응모한 결과인지 알 수 있다.

■ 챗봇 사용 화면(FAQ 조회 화면)



- 챗봇의 FAQ 기능을 통해 사용자는 질의응답을 할 수 있다.

■ 사용자 환경 설정 초기 화면(사용자 정보수정)



- ‘환경설정’을 선택한 페이지이다.
- ‘환경설정’ 텍스트의 색이 변경된다.
- 사용자는 해당 페이지에서 자신의 정보를 수정할 수 있다.
- 사용자는 내 정보, 코인, 거래내역, 배송 조회 등을 할 수 있다.

■ 사용자 환경 설정 화면(정보수정을 했을 때)

The screenshot shows a user profile settings page. At the top right, there is a success message: "수정 완료되었습니다." (Modification completed) with a "예" (Yes) button. The top navigation bar includes links for "로그아웃" (Logout), "환경설정" (Environment Settings), "오늘의 특가" (Today's Specials), "개시판" (Board), and "카테고리" (Category). Below the navigation bar, the main content area has a title "환경 설정" (Environment Settings) and a sub-section "개인 정보 변경" (Change personal information). A blue button labeled "수정 완료" (Modification completed) is highlighted. The form contains fields for "아이디" (ID), "비밀번호" (Password), "비밀번호 확인" (Password confirmation), "이름" (Name), "주소" (Address), and "전화번호" (Phone number). To the right of the form is a user profile picture and a "변경" (Change) button. At the bottom right of the content area is a "탈퇴하기" (Logout) link. On the far right, there are sections for "내 정보" (My information) with a "개인 정보 변경" (Change personal information) link, "코인" (Coin) with "보유 코인 조회" (Check held coins) and "코인 충전" (Coin recharge) links, "거래내역" (Transaction history) with "응모 내역" (Entry history) and "판매 내역" (Sales history) links, and a "배송" (Delivery) section with a mail icon.

- 사용자가 개인정보를 수정하고 '수정 완료' 버튼을 클릭하면 '수정 완료되었습니다' 메시지가 화면 상단에 나타난다.
- 사용자가 입력한 텍스트대로 데이터베이스의 회원 테이블도 업데이트 된다.

## ■ 사용자 환경 설정 화면(회원 탈퇴)

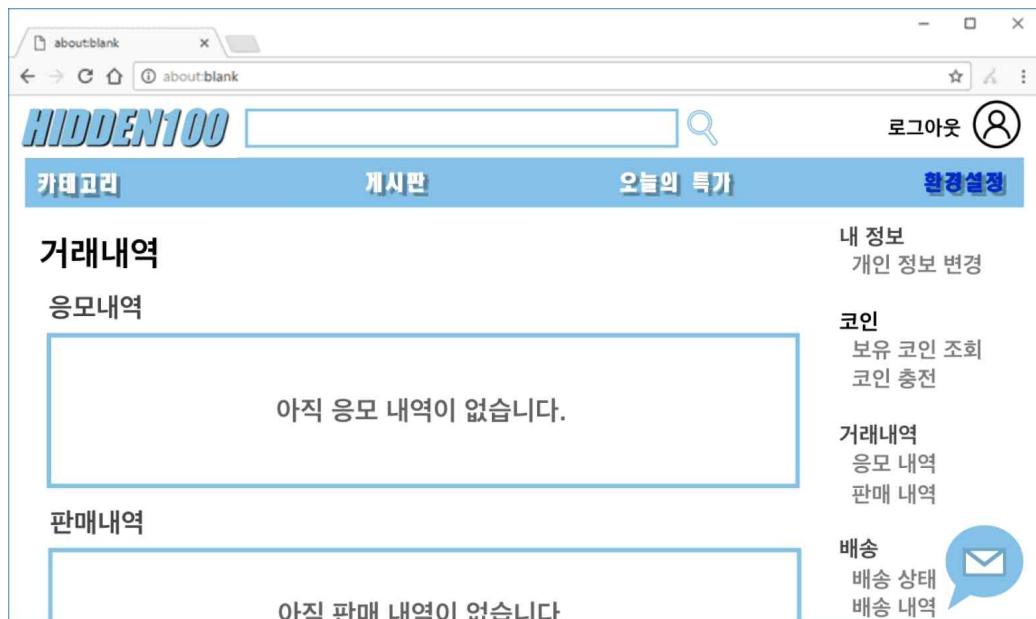


- 사용자가 화면 중앙 하단의 '탈퇴하기' 버튼을 누르면 '정말 탈퇴하시겠습니까?' 알림 메시지가 나타난다.
- 사용자가 '예'를 누르면 '탈퇴 완료되었습니다' 알림 메시지가 나타난다.
- 사용자가 '아니요'를 누르면 현재 페이지에 머무른다.

■ 사용자 환경 설정 화면(회원 탈퇴 되었을 때)



- 사용자가 회원 탈퇴를 하면 ‘탈퇴 완료되었습니다.’ 알림 메시지가 나타난다.
- 사용자가 ‘예’ 버튼을 클릭하면 웹 사이트의 메인 페이지로 돌아간다.

**■ 사용자 환경 설정 화면(거래내역이 없는 경우)**

- 사용자가 어떤 거래도 하지 않은 경우 응모내역과 판매내역에 각각 ‘아직 응모 내역이 없습니다’와 ‘아직 판매 내역이 없습니다’ 텍스트가 나타난다.

**■ 사용자 환경 설정 화면(응모내역이 있는 경우)**

The screenshot shows a web browser window for 'about:blank'. The main content area displays the 'HIDDEN100' website. At the top, there is a navigation bar with tabs: '카테고리', '게시판', '오늘의 특가', and '환경설정'. The '환경설정' tab is currently active. On the left, there is a sidebar with sections: '내 정보' (Personal Information) with a '개인 정보 변경' link; '코인' (Coins) with links to '보유 코인 조회' (Check Available Coins) and '코인 충전' (Top Up Coins); '거래내역' (Transaction History) with links to '응모 내역' (Bid History) and '판매 내역' (Sale History); and '배송' (Delivery) with links to '배송 상태' (Delivery Status) and '배송 내역' (Delivery History). The main content area is titled '거래내역' (Transaction History) and '응모내역' (Bid History). It lists two items:

- COMME DES GARCONS cardigan**: 목표금액 180,000원. Bidding status: 1,000원 (1), 7일 11:23:00시간 남음, 143명 참여중. Bid amount: 1,000원.
- Airpods**: 목표금액 100,000원. Bidding status: 100원 (50), 7일 11:23:00시간 남음, 1,143명 참여중. Bid amount: 5,000원.

- 사용자가 최소 하나의 응모라도 한 경우 응모내역에 상품응모리스트가 나타난다.

**■ 사용자 환경 설정 화면(판매내역이 있는 경우)**

The screenshot shows a web browser window for 'about:blank' with the title 'HIDDEN100'. The main content area displays two items for sale:

- McQueen sneakers**: Listed at 목표금액 300,000원. Status: 7일 11:23:00시간 남음, 653명 참여중. Current bid: 1,000원 by 653명.
- Airpods**: Listed at 목표금액 100,000원. Status: 2일 11:23:00시간 남음, 1,143명 참여중. Current bid: 100원 by 1,143명.

On the right side of the screen, there are several navigation links and icons:

- 내 정보**: 개인 정보 변경
- 코인**: 보유 코인 조회, 코인 충전
- 거래내역**: 응모 내역, 판매 내역
- 배송**: 배송 상태, 배송 내역

- 사용자가 최소 하나의 판매라도 한 경우 판매내역에 판매상품리스트가 나타난다.

## ■ 배송정보 조회 화면

The screenshot shows a web browser window for 'about:blank'. The main content area displays the 'HIDDEN100' website. At the top, there is a navigation bar with tabs for '카테고리', '게시판', '오늘의 특가', and '환경설정'. On the right side of the header, there are links for '로그아웃' (Logout), '내 정보' (My Information), and '개인 정보 변경' (Change Personal Information). Below the header, the word '배송' (Delivery) is prominently displayed. Under '배송 상태', there is a card for a 'Wooden chair' with a price of 300,000원 and a delivery status of '배송중' (In Progress). Another card for a 'Samsung Galaxy S10' with a price of 200,000원 is partially visible. To the right, there are sections for '코인' (Coin), '거래내역' (Transaction History), and a '배송' (Delivery) section containing links for '배송 상태' and '배송 내역'. A blue speech bubble icon is also present.

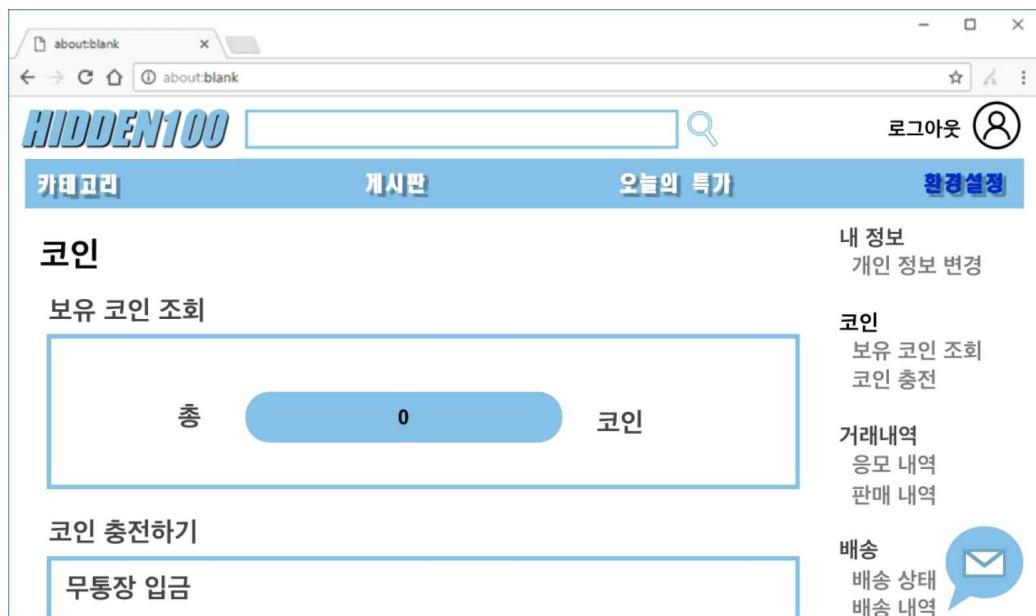
- 사용자는 거래 성사된 상품에 한하여 배송 상태를 조회할 수 있다.
- 사용자는 거래가 성사되고 배송까지 받은 상품에 한하여 배송 내역을 확인할 수 있다.

## ■ 배송 조회 화면(화면을 아래로 스크롤한 경우)



- 사용자가 보고 있는 화면을 아래로 스크롤하면 페이지 아래의 내용이 보여진다.
- 사용자는 필요한 정보를 보다 편리하게 볼 수 있다.

## ■ 코인 조회 화면



- 사용자는 자신이 보유하고 있는 코인을 조회할 수 있다.

## ■ 코인 충전 화면



- 사용자는 자신이 보유하고 있는 코인을 증가시키기 위해 코인을 충전할 수 있다.
  - 사용자는 충전할 만큼의 금액을 입력해서 코인을 충전할 수 있다.
- 코인 충전 화면(입력 텍스트 창에 금액을 입력했을 때)



- 사용자는 입력 텍스트 창에 충전할 만큼의 금액을 입력하여 코인을 충전할 수 있다.
- 사용자가 입력한 금액에 따라 보유 예정액의 숫자가 자동으로 업데이트 된다.

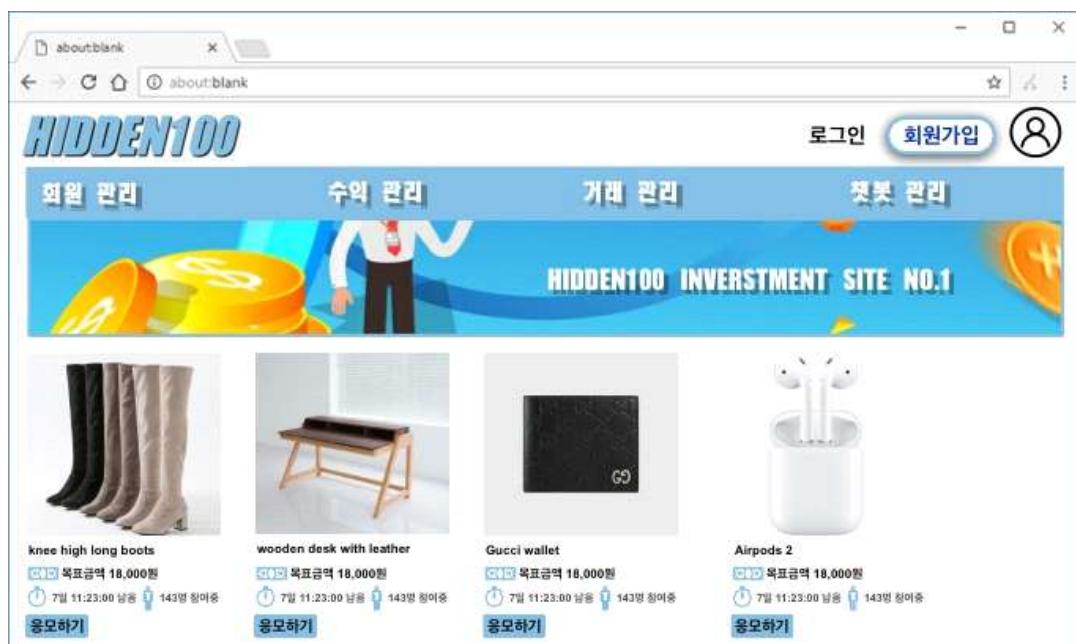
## ■ 코인을 조회 화면(코인 충전을 완료한 경우)



- 사용자가 코인 충전을 완료하면 자신이 보유한 코인양이 증가된다.
- 사용자는 보유한 코인에 따라 등록된 상품에 응모할 수 있다.

#### 4.4. 관리자UI 설계

##### ■ 관리자 기본 페이지



- 관리자도 사용자와 같이 웹페이지에 접속해서 관리자 인증 기능을 수행한다.
- 회원가입과 로그인으로 관리자 인증한다.
- 기본 페이지는 일반 사용자와 관리자가 동일하다.
- 관리자 회원가입은 다른 데이터베이스에 저장된다.

## ■ 관리자 로그인



- 관리자 기능을 이용하기 위해서는 관리자 로그인이 필요하다.
- 로그인 페이지에서 관리자 아이디와 비밀번호를 입력하면 관리자 로그인이 가능하다.
- '로그인'버튼을 선택하여 데이터를 전달한다.

■ 관리자 로그인(아이디 비밀번호 입력)



- 관리자 아이디와 비밀번호를 입력하고 로그인 버튼을 통해 관리자 인증할 수 있다.
- 관리자 인증에 성공하면 관리자 기능을 이용할 수 있다.
- 관리자 인증을 포함한 관리자 기능은 Web UI로 진행된다.

## ■ 관리자 회원가입

관리자 로그인이 필요합니다

회원가입하기

ID 아이디를 입력해주세요

PW 비밀번호를 입력해주세요

PW확인 비밀번호를 다시 입력해주세요

완료

로그인 하러가기 →

- 관리자도 사용자와 같이 웹페이지에서 회원가입을 진행할 수 있다.
- 관리자가 아이디와 패스워드 정보를 입력하여 '완료'버튼을 누르면 회원가입을 신청한다.
- 입력 정보에 오류가 없으면 회원가입 성공으로 넘어간다.

**■ 관리자 회원가입(데이터 입력 완료)**

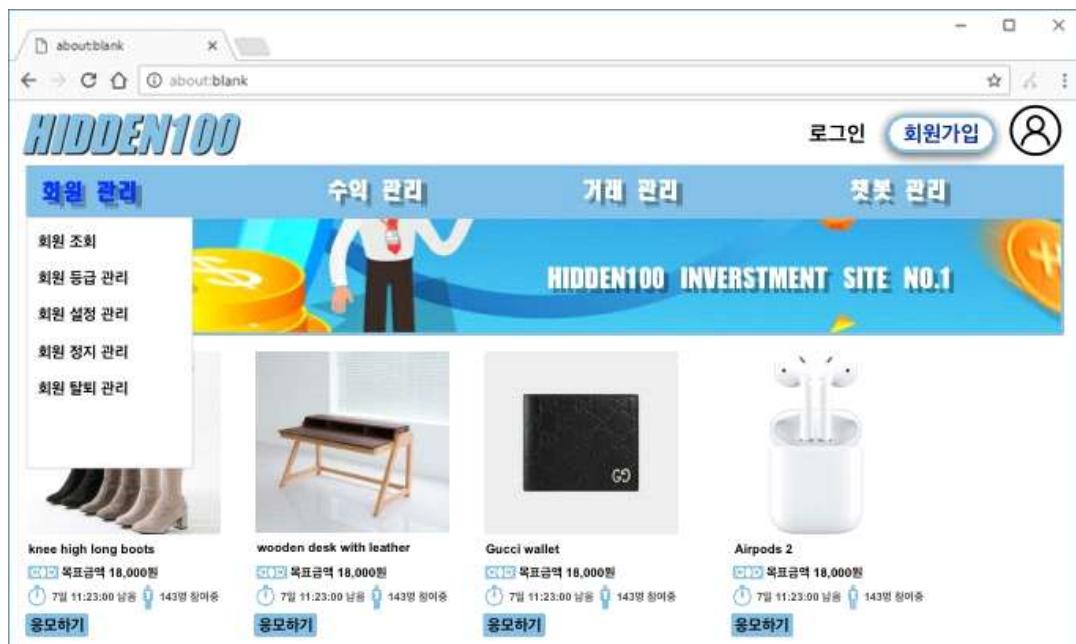
The screenshot shows two browser windows. The left window displays the main site interface with tabs for '회원 관리', '수익 관리', '기획 관리', and '챗봇 관리'. A message at the top says '회원가입이 완료되었습니다.' (Member registration completed). The right window shows a login page with tabs for '로그인' and '회원가입'. Below these tabs is a search bar and a user icon. The main area contains a heading '회원가입하기' (Register), a link '로그인 하러가기 →', and three input fields: 'ID' (soonsin), 'PW' (\*\*\*\*\*), and 'PW확인' (\*\*\*\*\*). A blue button labeled '완료' (Complete) is at the bottom. A status message next to the ID field says '아이디 중복 체크' (Duplicate ID check) and '사용 가능한 아이디입니다.' (Valid user ID).

- 관리자도 사용자와 같이 웹페이지에 접속해서 관리자 인증 기능을 수행한다.
- 회원가입과 로그인으로 관리자 인증한다.

## ■ 관리자 로그아웃



- 관리자도 사용자와 같이 로그아웃으로 세션에 저장된 관리자 인증 정보를 삭제하고 로그아웃할 수 있다.
- 로그아웃하면 기존의 관리자 기능은 다시 인증해야 이용 가능하다.

**■ 관리자 페이지 회원 관리 드롭다운**

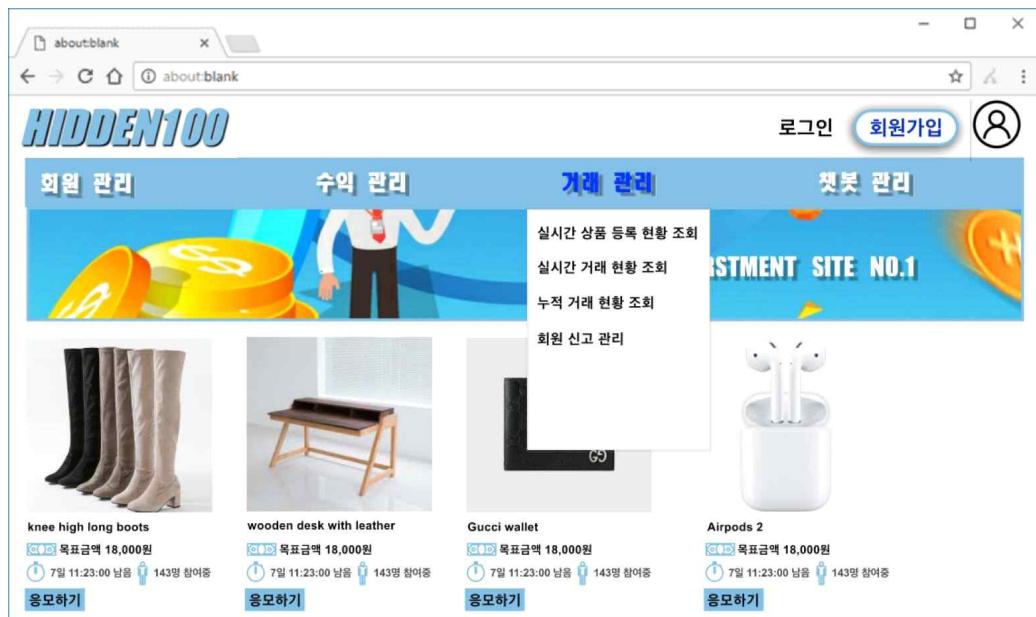
- 관리자 로그인에 성공하면 상단바에서 관리자 기능을 선택할 수 있다.
- 드롭다운으로 등장한 버튼을 눌러 관리자는 회원관리 기능을 사용한다.
- 관리자는 모든 일반 회원에 대해 ‘회원조회’, ‘회원등급관리’, ‘회원설정 관리’, ‘회원정지 관리’, ‘회원탈퇴 관리’가 가능하다.

## ■ 회원등급 관리

번호	아이디	비밀번호	회원명	주소	전화번호	등급
1	sister01	1321	이다온	경기도 성남시 구미동 영진로 12 112-302	010-2333-4322	5
2	wowow123	wow12345	이우석	경기도 성남시 서현동 연구로 15 123-301	010-3433-6564	3
3	haehae435	hae!98766	한혜경	서울 특별시 마포동 연남로 153 503-201	010-2323-6554	4
4	yo1yo1	YOyoyo040321	서지연	서울 특별시 강남구 강남로 143-203	010-4343-7654	3

- 관리자는 회원의 등급을 웹페이지에서 변경할 수 있다.
- 회원 정보를 조회하고 등급을 변경한다.
- 등급은 경험치에 따라 자동으로 변경되나 신고가 들어오는 경우 등급을 하락시킬 수 있다.

## ■ 거래관리화면(호버링)



- ‘거래 관리’부분이 강조 되고 드롭 다운 메뉴가 나타난다.

## ■ 거래관리 화면(실시간 상품 등록현황)

The screenshot shows a web browser window with the title 'about:blank'. The main content area displays a table titled '실시간 상품 등록 현황 조회' (Real-time Product Registration Status Inquiry) with 259 entries. The table columns include 번호 (Number), 물품 번호 (Product Number), 물품명 (Product Name), 응모 가격 (Bid Price), 판매자 (Seller), 총 응모 인원(명) (Total Bidder Count), 총 응모 금액 (Total Bid Amount), and 수수료 (Commission). The data shows four items listed.

번호	물품 번호	물품명	응모 가격 (원)	판매자	총 응모 인원(명)	총 응모 금 액	수수료
1	M120002	Bose Speaker	100,000	이다은	103	103,000	1.5
2	M120003	Gucci Wallet	180,000	이우석	60	60,000	-
3	M120004	iPhone	320,000	한혜경	11	11,000	-
4	M120005	Wooden Desk	60,000	서지연	32	32,000	-

## ■ 거래관리 화면(실시간 상품 거래 현황)

The screenshot shows a web browser window with the title 'about:blank'. The main content area displays a table titled '실시간 상품 거래 현황 조회' (Real-time Product Transaction Status Inquiry) with 169 entries. The table columns are identical to the previous table. The data shows four items listed.

번호	물품 번호	물품명	응모 가격 (원)	판매자	총 응모 인원(명)	총 응모 금 액	수수료
1	M120002	Bose Speaker	100,000	이다은	103	103,000	1.5
2	M120003	Gucci Wallet	180,000	이우석	230	230,000	1.5
3	M120004	iPhone	320,000	한혜경	521	521,000	2
4	M120005	Wooden Desk	60,000	서지연	61	61,000	1.5

■ 거래관리 화면(누적 거래 현황 조회)

번호	물품 번호	물품명	총 가격 (원)	판매자	총 인원(명)	총 금액	수수료
1	M120002	Bose Speaker	100,000	이다은	103	103,000	1.5
2	M120003	Gucci Wallet	180,000	이우석	230	230,000	1.5
3	M120004	iPhone	320,000	한혜경	521	521,000	2
4	M120005	Wooden Desk	60,000	서지연	61	61,000	1.5

■ 챗봇 관리 화면(FAQ관리)

- 챗봇 관리
  - 알림 발신
  - FAQ 관리
  - 챗봇 로그

- 가장 많이 질문했던 질문들을 키워드화하여 단어로 표현한 후 이를 이미지화 시켜서 보여준다.

## ■ 챗봇 관리 화면(알림발신)

**Hidden 100**

로그인 회원가입

회원 관리 수익 관리 거래 관리 챗봇 관리

챗봇 관리

알림 발신

FAQ 관리

챗봇 로그

추가하기

번호	아이디	회원명	받은 메세지	챗봇 메세지
1	sister01	이다은	-	실패입니다! 다음번에 도전해주세요
2	wowow123	이우석	-	성공입니다! 상품을 놀려 상품을 수령해주세요
3	haehae435	한혜경	내 등급 지금 뭐야?	회원님은 현재 등급 1이십니다. 더 많은 응모 부탁드릴게요~
4	yo1yo1	서지연	신고하고 싶은데 어떻게 하나요?	신고는 상단 바의 게시판에서 신고 게시판에 문의해주시면 가능합니다.
5	workit543	김수경	노트북 30만원대 검색해줘.	<a href="http://hidden100/item=11022344">http://hidden100/item=11022344</a> 마감에 임박한 제품입니다! 서둘러 확인해주세요~

## ■ 챗봇 관리 화면(로그)

**Hidden 100**

로그인 회원가입

회원 관리 수익 관리 거래 관리 챗봇 관리

챗봇 관리

챗봇 로그

추가하기

번호	받은 메세지	챗봇 메세지
1	판매하기 어떻게해?	상단 바의 환경설정 탭 누르시면 하단에 판매하기 버튼 눌러주시면 가능합니다.
2	당첨되었는지 어떻게 알아?	메세지로 알림 보내드릴꺼예요! 마감시간까지 조금만 기다려주세요~
3	내 등급 지금 뭐야?	회원님은 현재 등급 (#grade)이십니다. 더 많은 응모 부탁드릴게요~
4	신고하고 싶은데 어떻게 하나요?	신고는 상단 바의 게시판에서 신고 게시판에 문의해주시면 가능합니다.
5	노트북 30만원대 검색해줘.	<a href="http://hidden100/item=11022344">http://hidden100/item=11022344</a> 마감에 임박한 제품입니다! 서둘러 확인해주세요~

- 챗봇으로 받은 메시지들과 보낸 메시지들의 기록들이 보여지게 된다.
- 오른쪽 상단의 추가하기 버튼을 누름으로써 챗봇으로 메시지를 보낼 수 있다.

## ■ 회원정지 관리

번호	아이디	비밀번호	회원명	주소	전화번호	등급
1	sister01	1321	이다온	경기도 성남시 구미동 영진로 12 112-302	010-2333-4322	5
2	wowow123	wow12345	이우석	경기도 성남시 서현동 연구로 15 123-301	010-3433-6564	3
3	haehae435	hae!98766	한혜경	서울 특별시 마포동 연남로 153 503-201	010-2323-6554	4
4	yo1yo1	YOyoyo040321	서지연	서울 특별시 강남구 강남로 143-203	010-4343-7654	3

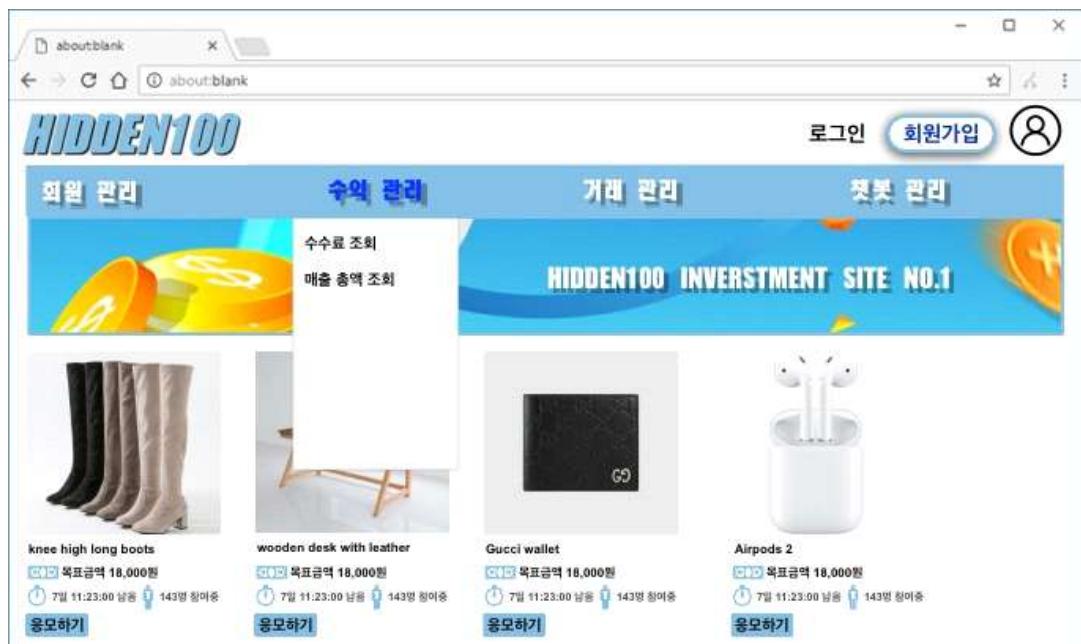
- 관리자는 회원 계정을 정지시킬 수 있다.
- 정지된 사용자는 관리자가 정지를 풀어주기 전까지 로그인할 수 없다.
- 거래신고나 사기, 게시판 악용등이 있을 경우 관리자 판단에 따라 사용자를 정지한다.

## ■ 회원탈퇴 관리

The screenshot shows a web browser window for 'about:blank' with the title 'HIDDEN100'. The main navigation menu includes '회원 관리', '수익 관리', '거래 관리', and '챗봇 관리'. On the right side, there is a sidebar with links: '회원 관리', '회원 조회', '회원 등급 관리', '회원 설정 관리', '회원 정지 관리', '회원 탈퇴 관리', and '회원 신고 관리'. The central content area is titled '회원관리' and '회원 탈퇴 관리'. It displays a table with four rows of member information, each with a '탈퇴' (Delete) button.

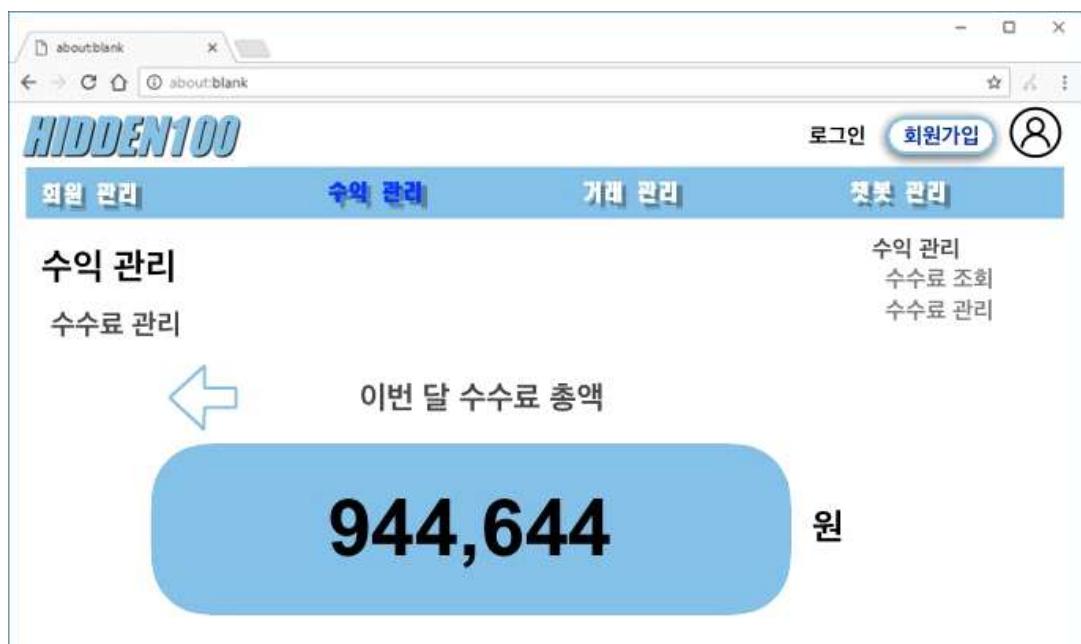
아이디	비밀번호	회원명	주소	전화번호	
sister01	1321	0 나은	경기도 성남시 구미 동 영진로 12 112-302	010-2333-4322	<b>탈퇴</b>
wowow123	wow12345	이우석	경기도 성남시 서현 동 연구로 15 123-301	010-3433-6564	<b>탈퇴</b>
haehae435	hael98766	한혜경	서울 특별시 마포동 연남로 153 503-201	010-2323-6554	<b>탈퇴</b>
yo1yo1	YOyoyo040321	서지연	서울 특별시 강남구 강남로 143-203	010-4343-7654	<b>탈퇴</b>

- 관리자는 회원 계정을 탈퇴시킬 수 있다.
- 탈퇴된 계정은 서버에서 삭제된다.

**■ 관리자 메뉴 수익관리 드롭다운**

- 수익관리에 마우스를 올리면 드롭다운으로 메뉴가 표시된다.
- 관리자는 ‘수수료 조회’와 ‘매출총액 조회’가 가능하다.

## ■ 수수료 조회



- 쇼핑몰은 중고거래가 성사된 경우와 코인을 적립할 때 수수료를 받는다.
- 관리자는 순이익인 수수료를 쉽게 확인할 수 있다.

**■ 매출총액 조회**

- 거래가 성사된 경우 총 적립액을 기록한다.
- 관리자는 쉽게 총 매출액을 확인할 수 있다.

## 5. REST API 인터페이스 설계

### 5.1. 서버/클라이언트 구조

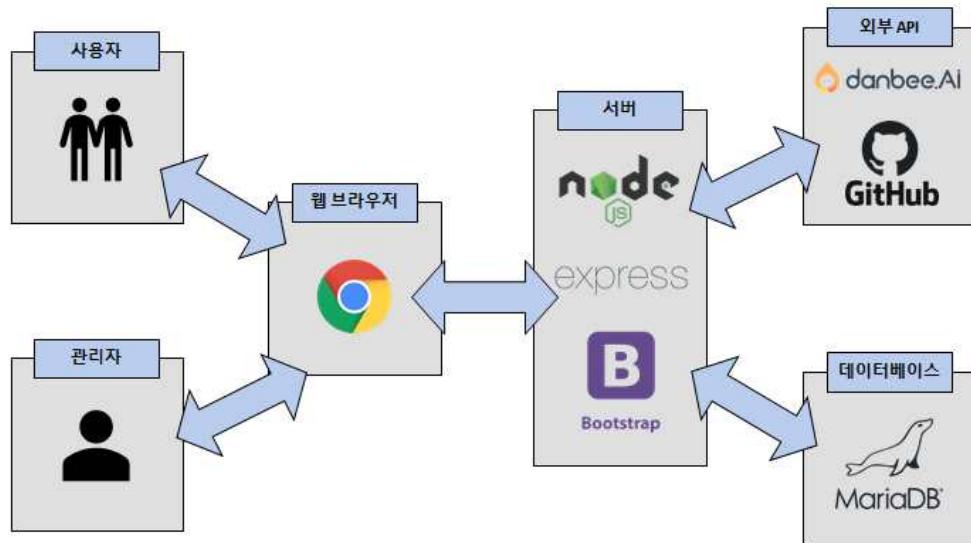


그림 5.1 서비스 인터페이스 구조

본 설계는 그림 5.1과 같이 클라이언트(사용자(판매자, 구매자), 관리자)와 서버(웹 서버)가 웹 브라우저인 구글 크롬을 통해 필요한 정보를 송수신하는 인터페이스 전송 구조이다. 즉, 클라이언트가 서버에게 정보를 요청하거나 응답하는 방식으로 REST API를 이용하여 설계한다.

Hidden 100의 메인 서버는 Node.js, Express, Bootstrap 등 내부에 여러 모듈로 구성되어 있으며, 데이터베이스는 MariaDB를 사용한다. 그리고 외부 API들을 통해 사용자에게 챗봇과 데이터 시각화 서비스를 제공한다.

클라이언트가 서버에게 요청하는 메시지 타입은 크게 4가지인 조회(GET), 생성(POST), 수정(PUT), 삭제(DELETE)로 구분하고, 각 요청별로 알맞은 REST API 방식으로 메시지를 설계한다. 클라이언트와 서버의 통신을 원활히 설계하기 위해 다음과 같이 각 연동별로 구분하여 REST API를 정의한다.

- 웹 서비스 - 서버와의 연동
- 데이터베이스 - 서버와의 연동

이어서 각 연동별로 REST API를 정의한 후 상세설계를 한다.

## 5.2. REST API 정의

Method	URI	Description
GET	/seller/goods/<userid>	자신이 등록한 상품의 정보 요청 (ex. 상품명, 참여자 수, 적립금 등)
	/user/goods/all	판매물품 조회 (ex. 상품명, 적립 마감 시간 등)
	/user/goods/<option>	판매물품 검색 (ex. 상품명, 적립 마감 시간 등)
	/user/goods/participant/<goodid>	참여자수 조회 (ex. 참여자 수)
	/user/board/<boardname>	게시글 조회 (ex. 게시글 제목, 내용 등)
	/user/chatbot/myinfo/<userid>	내 정보 조회 (ex. 이름, 등급, 경험치 등)
	/user/chatbot/goods/<str>	판매상품 조회 (ex. 상품 URL 등)
	/admin/user/<userid>	회원정보 조회 (ex. 아이디, 비밀번호, 이름, 주소 등)
	/admin/chatbot	챗봇로그 요청 (ex. 챗봇 검색 빈도 결과 등)
	/admin/goods	실시간 등록상품 조회 (ex. 상품명, 참여자 수, 적립금 등)
	/admin/goods/<goodid>	등록상품 조회 요청 (ex. 상품명, 참여자 수, 적립금 등)
	/admin/active	실시간 거래현황 조회 (ex. 실시간 적립금, 예상 수수료 등)
	/admin/total	누적 거래현황 조회 (ex. 누적 적립금, 누적 수수료 등)
	/admin/report	신고내역 조회 요청 (ex. 신고글 내용, 신고자 명 등)
	/admin/profit	수수료 내역 요청 (ex. 수수료 내역 통계 등)
	/admin/sales	전체 매출액 요청 (ex. 거래 결과 적립금 내역 통계 등)

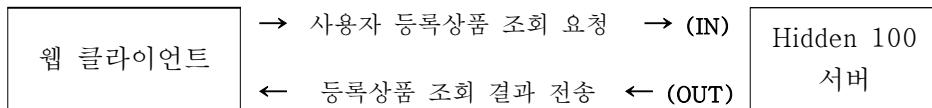
Method	URI	Description
POST	/user/singin	사용자 회원가입
	/admin/singin	관리자 회원가입
	/seller/register	판매자 물품등록
	/seller/register/shipment	판매자 배송정보 등록
	/user/participant	사용자 코인적립 참여
	/user/report	거래 사기 신고
	/user/board	게시글 작성
	/admin/faq	FAQ 작성

Method	URI	Description
PUT	/user/<userid>/info	사용자 본인 정보수정
	/user/<userid>/coin	사용자 보유 코인 정보수정
	/user/<boardid>/board	사용자 게시글 수정
	/admin/<userid>/level	회원등급 정보수정
	/admin/<userid>/info	회원설정 정보수정
	/admin/<userid>/status	회원 정지상태 수정

Method	URI	Description
DELETE	/user/<userid>/delete	사용자 회원탈퇴
	/user/board/<boardid>/delete	사용자 게시글 삭제
	/seller/goods/<gooid>/delete	판매자 물품삭제
	/admin/user/<userid>/delete	관리자 회원삭제

### 5.3. 상세 REST API 설계

GET /seller/goods/<userid>



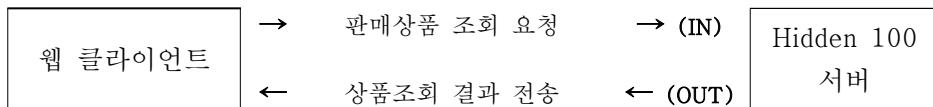
#### Parameter

속성	전송방향		Type	Description
	IN	OUT		
userid	O		String	요청한 사용자 아이디
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지
goods_info		O	Array	상품 정보

## 실제 전송내용 (JSON 의 경우라면)

```
전송방향 : IN
{
    "userid" : "hiddens"
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
    "goods_info" :
    [
        [
            "name" : "아이폰8 64GB 스페이스 그레이" ,
            "type" : "휴대폰",
            "price" : 2000,
            "time_year" : 2019 ,
            "time_month" : 12 ,
            "time_day" : 25 ,
            "time_hour" : 0 ,
            "time_minute" : 0 ,
            "coin" : 500,
            "participation" : 4
        ],
    ]
}
```

GET /user/goods/all



## Parameter

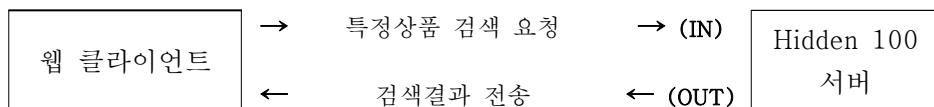
속성	전송 방향		Type	Description
	IN	OUT		
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지
goods_info		O	Array	상품 정보

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
    "goods_info" :
    [
        [
            {
                "name" : "아이폰8 64GB 스페이스 그레이" ,
                "type" : "휴대폰",
                "price" : 2000,
                "time_year" : 2019 ,
                "time_month" : 12 ,
                "time_day" : 25 ,
                "time_hour" : 0 ,
                "time_minute" : 0 ,
                "coin" : 500,
                "participation" : 4
            },
            ...
        ]
    }
}
  
```

GET /user/goods/&lt;option&gt;



## Parameter

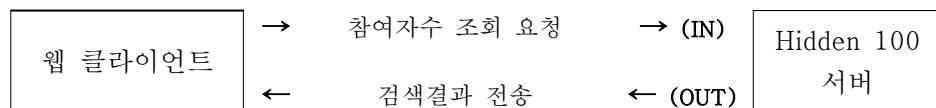
속성	전송 방향		Type	Description
	IN	OUT		
option	O		String	요청한 상품 검색어
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지
goods_info		O	Array	상품 정보

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "option" : "아이폰"
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
    "goods_info" :
    [
        [
            {
                "name" : "아이폰8 64GB 스페이스 그레이" ,
                "type" : "휴대폰",
                "price" : 2000,
                "time_year" : 2019 ,
                "time_month" : 12 ,
                "time_day" : 25 ,
                "time_hour" : 0 ,
                "time_minute" : 0 ,
                "coin" : 500,
                "participation" : 4
            },
        ]
    }
}
  
```

GET /user/goods/participant/<gooid>



### Parameter

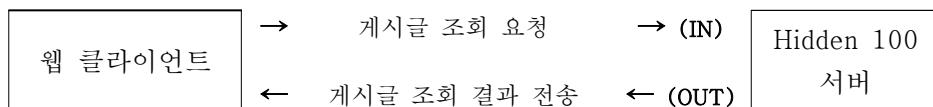
속성	전송방향		Type	Description
	IN	OUT		
gooid	O		int	요청한 상품고유번호
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지
goo_part_info		O	Array	상품 참여자수 정보

### 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "gooid" : 35832742
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
    "goo_part_info" : 178
}
  
```

GET /user/board/&lt;boardname&gt;



## Parameter

속성	전송 방향		Type	Description
	IN	OUT		
boardname	O		String	요청한 게시글 검색어
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지
board_info		O	Array	게시글 정보

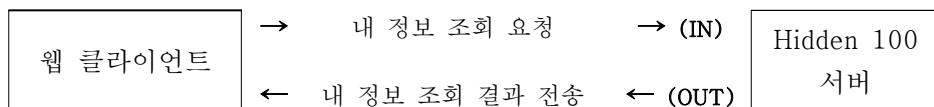
## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "boardname" : "에어팟"
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
    "board_info" :
    [
        [
            [
                "num" : 6,
                "mem_id" : "hiddens",
                "subject" : "에어팟 프로 어떤가요" ,
                "content" : "가격대비 성능 궁금합니다",
                "regist_day" : "2019-11-30",
                "hit" : 156 ,
                "file_name" : "" ,
                "file_type" : "" ,
                "file_copied" : ""
            ],
        ]
    }
}

```

GET /user/chatbot/myinfo/&lt;userid&gt;

**Parameter**

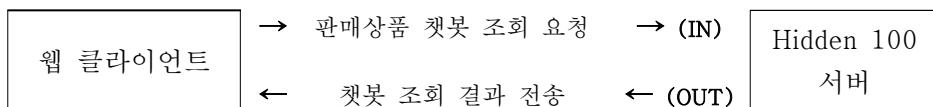
속성	전송 방향		Type	Description
	IN	OUT		
userid	O		String	요청한 사용자 아이디
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지
my_info		O	Array	개인정보

**실제 전송내용 (JSON 의 경우라면)**

```

전송방향 : IN
{
    "userid" : "hiddens"
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
    "my_info" :
    [
        {
            "id" : "hiddens" ,
            "name" : "김용보" ,
            "address" : "서울특별시 석촌로 38" ,
            "regist_day" : "2019-12-01" ,
            "coin" : 3700 ,
            "level" : "1" ,
            "point" : 58 ,
            "status" : "활성"
        }
    ]
}
  
```

GET /user/chatbot/goods/&lt;str&gt;



## Parameter

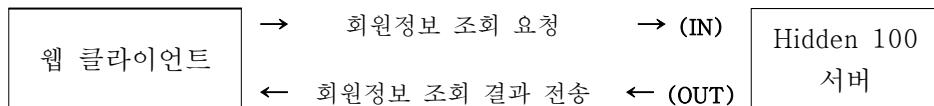
속성	전송 방향		Type	Description
	IN	OUT		
str	O		String	요청한 상품 검색어
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지
goods_info		O	Array	상품 정보

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "str" : "아이폰"
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
    "goods_info" :
    [
        [
            [
                "name" : "아이폰8 64GB 스페이스 그레이" ,
                "type" : "휴대폰",
                "price" : 2000,
                "time_year" : 2019 ,
                "time_month" : 12 ,
                "time_day" : 25 ,
                "time_hour" : 0 ,
                "time_minute" : 0 ,
                "coin" : 500,
                "participation" : 4
            ],
            ...
        ]
    }
}
  
```

GET /admin/user/&lt;userid&gt;



## Parameter

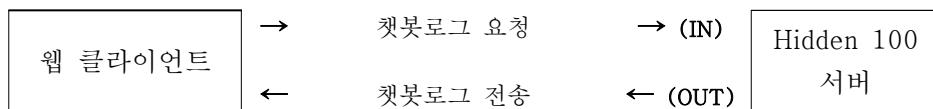
속성	전송방향		Type	Description
	IN	OUT		
userid	O		String	요청한 사용자아이디
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지
user_info		O	Array	회원 정보

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "userid" : "hiddens"
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
    "user_info" :
    [
        {
            "id" : "hiddens" ,
            "pass" : "hidden1234" ,
            "name" : "김용모" ,
            "address" : "서울특별시 석촌로 38" ,
            "regist_day" : "2019-12-01" ,
            "coin" : 3700 ,
            "level" : "1" ,
            "point" : 58 ,
            "status" : "활성"
        }
    ]
}
  
```

GET /admin/chatbot



## Parameter

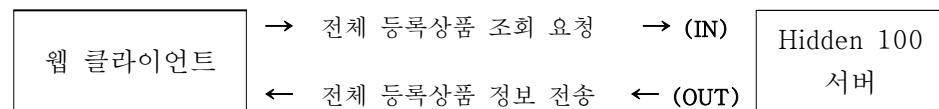
속성	전송 방향		Type	Description
	IN	OUT		
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지
chat_log_info		O	Array	챗봇로그 정보

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
    "chat_log_info" :
    [
        [
            {
                "mem_id" : "hiddens" ,
                "mem_name" : "김옹보" ,
                "content" : "내 정보 조회해줘" ,
                "regist_day" : "2019-12-01"
            },
        ]
    ]
}
  
```

GET /admin/goods



## Parameter

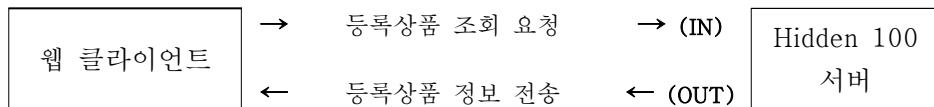
속성	전송 방향		Type	Description
	IN	OUT		
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지
goods_info		O	Array	센서 정보

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
    "goods_info" :
    [
        [
            {
                "name" : "아이폰8 64GB 스페이스 그레이" ,
                "type" : "휴대폰",
                "price" : 2000,
                "time_year" : 2019 ,
                "time_month" : 12 ,
                "time_day" : 25 ,
                "time_hour" : 0 ,
                "time_minute" : 0 ,
                "coin" : 500,
                "participation" : 4
            },
            ...
        ]
    }
}
  
```

GET /admin/goods/&lt;gooid&gt;



## Parameter

속성	전송 방향		Type	Description
	IN	OUT		
gooid	O		int	요청한 상품 고유 번호
content_type		O	String	application/json
result_code		O	int	처리 결과 코드 (200: 성공, 600: 오류 코드)
result_req		O	String	처리 결과 오류 메세지
goods_info		O	Array	상품 정보

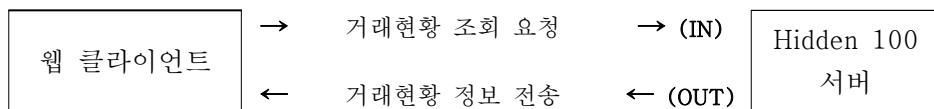
## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "gooid" : 35832742
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
    "goods_info" :
    [
        {
            "name" : "아이폰8 64GB 스페이스 그레이" ,
            "type" : "휴대폰",
            "price" : 2000,
            "time_year" : 2019 ,
            "time_month" : 12 ,
            "time_day" : 25 ,
            "time_hour" : 0 ,
            "time_minute" : 0 ,
            "coin" : 500,
            "participation" : 4
        }
    ]
}

```

GET /admin/active



## Parameter

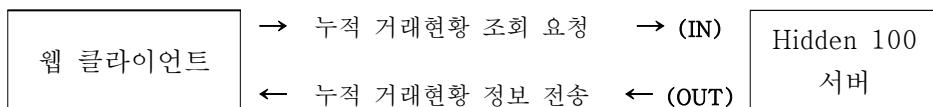
속성	전송 방향		Type	Description
	IN	OUT		
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지
deal_active_info		O	Array	실시간 거래현황 정보

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
    "deal_active_info" :
    [
        [
            {
                "num" : 1 ,
                "mem_id" : "hiddens" ,
                "goo_id" : 39283784 ,
                "goo_name" : "에어팟2" ,
                "price" : 80000 ,
                "coin" : 9000 ,
                "regist_day" : "2019-11-30"
            },
            ...
        ]
    }
}
  
```

GET /admin/total



## Parameter

속성	전송 방향		Type	Description
	IN	OUT		
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지
deal_total_info		O	Array	누적 거래현황 정보

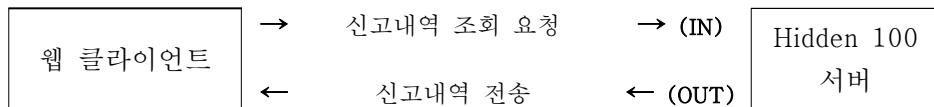
## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
}

전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
    "deal_total_info" :
    [
        [
            {
                "num" : 1 ,
                "goo_id" : 39284291 ,
                "goo_name" : "갤럭시 노트 5" ,
                "price" : 80000 ,
                "coin_total" : "84000" ,
                "mem_num" : 142 ,
                "start_day" : "2019-11-29" ,
                "fin_day" : "2019-12-1"
            },
            ...
        ]
    }
}
  
```

GET /admin/report



## Parameter

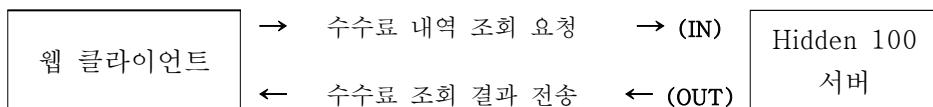
속성	전송 방향		Type	Description
	IN	OUT		
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지
report_info		O	Array	신고내역 정보

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
    "report_info" :
    [
        [
            {
                "num" : 1 ,
                "mem_id" : "user5" ,
                "mem_name" : "박신고" ,
                "subject" : "물건 배송이 안 와요",
                "content" : "운송장번호 입력한 지 10일이 됐는데 못 받았습니다",
                "regist_day" : "2019-11-28" ,
                "hit" : 6
            },
            ...
        ]
    }
}
  
```

GET /admin/profit



## Parameter

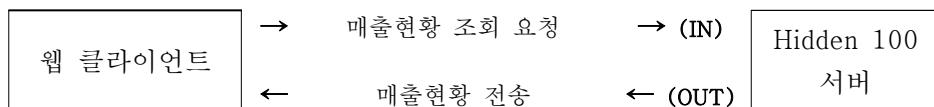
속성	전송 방향		Type	Description
	IN	OUT		
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지
profit_info		O	Array	수수료 정보

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
    "profit_info" :
    [
        "level1" : "10%" ,
        "level2" : "5%" ,
        "total" : 89000
    ]
}
  
```

GET /admin/sales

**Parameter**

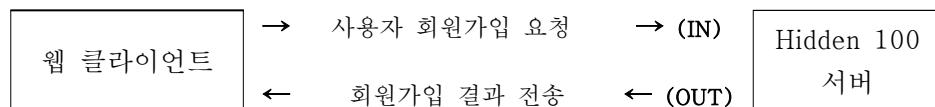
속성	전송방향		Type	Description
	IN	OUT		
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지
sales_info		O	Array	전체 매출액 정보

**실제 전송내용 (JSON 의 경우라면)**

```

전송방향 : IN
{
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
    "sales_info" :
    [
        "profit" : 89000 ,
        "deal" : 106000 ,
        "total" : 195000
    ]
}
  
```

POST /user/signin



## Parameter

속성	전송방향		Type	Description
	IN	OUT		
user	O		Array	회원가입할 사용자 정보
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지

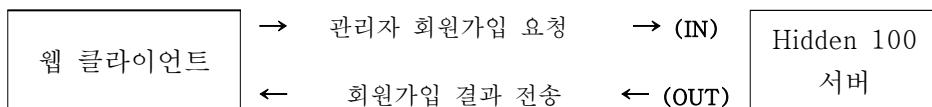
## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "user" :
    [
        {
            "name" : "홍길동",
            "id" : "user333",
            "pass" : "scanf23"
        }
    ]
}

전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
}
  
```

POST /admin/signin



## Parameter

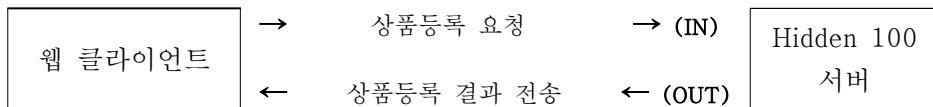
속성	전송방향		Type	Description
	IN	OUT		
admin	O		Array	회원가입할 관리자 정보
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "admin" :
    [
        "name" : "김길동" ,
        "id" : "admin123",
        "pass" : "printf123"
    ]
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
}
  
```

POST /seller/register

**Parameter**

속성	전송 방향		Type	Description
	IN	OUT		
userid	O			등록하는 판매자 아이디
good_info	O		Array	등록할 상품정보
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지

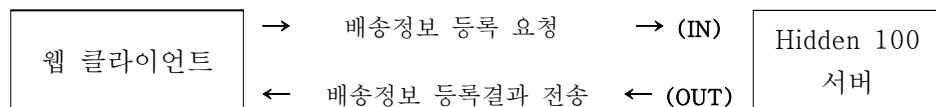
**실제 전송내용 (JSON 의 경우라면)**

```

전송방향 : IN
{
    "userid" : "sns_member",
    "good_info" :
    [
        {
            "name" : "아이폰8 64GB 스페이스 그레이" ,
            "type" : "휴대폰",
            "price" : 2000,
            "time_year" : 2019 ,
            "time_month" : 12 ,
            "time_day" : 25 ,
            "time_hour" : 0 ,
            "time_minute" : 0 ,
            "participation" : 4
        }
    ]
}

전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
}
  
```

POST /seller/register/shipment



## Parameter

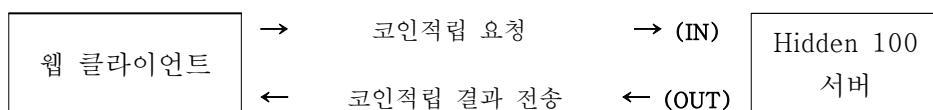
속성	전송 방향		Type	Description
	IN	OUT		
goodid	O		int	등록할 상품 아이디
shipment_num	O		String	등록한 배송번호
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "goodid" : 145,
    "shipment_num" : "36005017"
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : ""
}
  
```

POST /user/participant



## Parameter

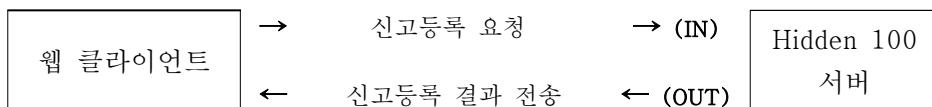
속성	전송방향		Type	Description
	IN	OUT		
userid	O		String	요청한 사용자아이디
coin	O		int	적립한 코인
goodname	O		String	적립한 상품명
goodid	O		int	적립한 상품 아이디
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "userid" : "user123",
    "coin" : 100,
    "goodname" : "아이폰8 64GB",
    "goodid" : 142
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
}
  
```

POST /user/report



## Parameter

속성	전송 방향		Type	Description
	IN	OUT		
report	O		Array	사용자가 작성한 게시글
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지

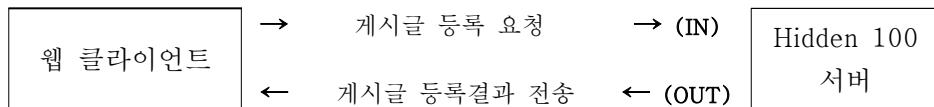
## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "report" : [
        "userid": "user123",
        "subject": "아이폰 거래사기 신고",
        "contents": "벽돌이 왔어요 상품번호:000123",
        "filename": "",
        "filetype": "",
        "filecopied": ""
    ]
}

전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : ""
}
  
```

POST /user/board



## Parameter

속성	전송 방향		Type	Description
	IN	OUT		
board	O		Array	사용자가 작성한 게시글
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지

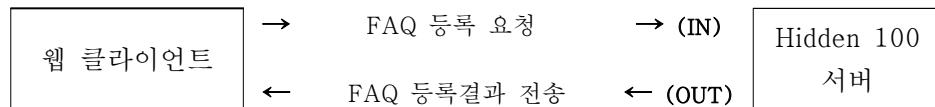
## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "board" : [
        "userid": "user123",
        "subject": "아이폰 당첨후기",
        "contents": "당첨이 되었는데 상품을 받으니까~~~~ 감사합니다.",
        "filename": "",
        "filetype": "",
        "filecopied": ""
    ]
}

전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : ""
}
  
```

POST /admin/faq



## Parameter

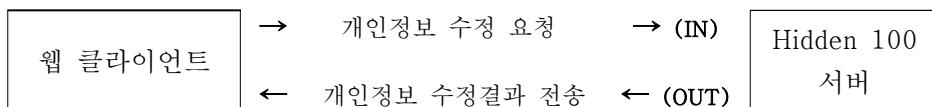
속성	전송 방향		Type	Description
	IN	OUT		
faq	O			등록할 FAQ
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
  "faq" :
  [
    "Q" : "Hidden100 레벨은 무슨 영향이 있나요?",
    "A" : "레벨이 오르면 코인충전할 때 수수료가 감소합니다!"
  ]
}
전송방향 : OUT
{
  "content_type" : "json" ,
  "result_code" : 200 ,
  "result_req" : ""
}
  
```

PUT /user/&lt;userid&gt;/info



## Parameter

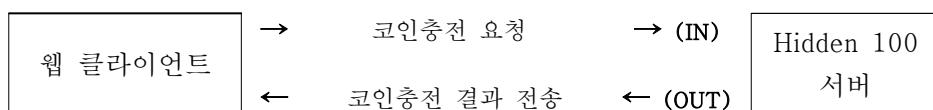
속성	전송 방향		Type	Description
	IN	OUT		
userid	O		String	요청한 사용자아이디
user_info	O		Array	요청한 사용자 변경 정보
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "userid" : "user123"
    "user_info" :
    [
        {
            "userid" : "user123",
            "address" : "경기도 성남시 수정구 태평로 20길 103호"
        }
    ]
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : "" ,
}
  
```

PUT /user/&lt;userid&gt;/coin



## Parameter

속성	전송 방향		Type	Description
	IN	OUT		
userid	O		String	요청한 사용자아이디
coin	O		int	충전한 코인 금액
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지

## 실제 전송내용 (JSON 의 경우라면)

```

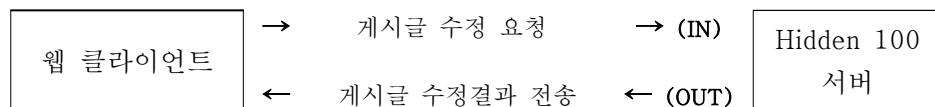
전송방향 : IN
{
    "userid" : "user123",
    "coin" : 4500,
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : ""
}
  
```

팀번호: 01

챗봇과 데이터 시각화를 이용한  
중고거래 서비스 Hidden 100version  
v1.0

【P-실무프로젝트】

PUT /user/&lt;boardid&gt;/board



## Parameter

속성	전송 방향		Type	Description
	IN	OUT		
boardid	O		int	요청한 사용자아이디
board	O		Array	요청한 사용자아이디
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지

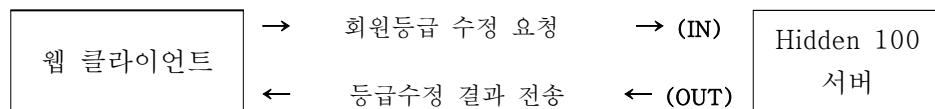
## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "boardid" : 12,
    "board" : [
        "subject": "아이폰 당첨후기",
        "contents": "당첨이 되었는데 상품을 받으니까~~~~ 감사합니다."
    ]
}

전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : ""
}
  
```

PUT /admin/&lt;userid&gt;/level



## Parameter

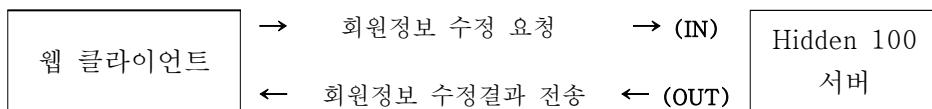
속성	전송방향		Type	Description
	IN	OUT		
userid	O		String	요청한 사용자아이디
level	O		String	변경할 사용자 레벨
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "userid" : "sns_member",
    "level" : "2"
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : ""
}
  
```

PUT /admin/&lt;userid&gt;/info



## Parameter

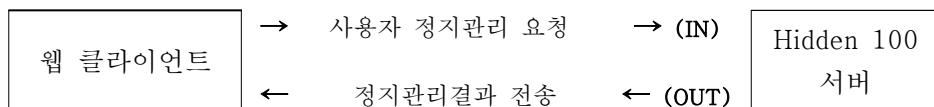
속성	전송방향		Type	Description
	IN	OUT		
userid	O		String	요청한 사용자아이디
user_info	O		Array	변경할 사용자 정보
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "userid" : "user123",
    "user_info" :
    [
        {
            "id" : "user123" ,
            "name" : "홍길동" ,
            "coin" : 300 ,
            "level" : "1" ,
            "point" : 20
        }
    ]
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : ""
}
  
```

PUT /admin/&lt;userid&gt;/status



## Parameter

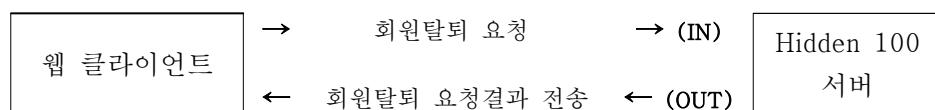
속성	전송 방향		Type	Description
	IN	OUT		
userid	O		String	변경할 사용자 아이디
status	O		String	변경할 사용자 상태
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "userid" : "userid123",
    "status" : "정지"
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : ""
}
  
```

DELETE /user/&lt;userid&gt;/delete



## Parameter

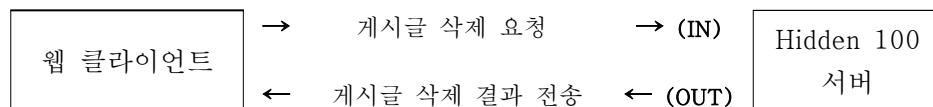
속성	전송방향		Type	Description
	IN	OUT		
userid	O		String	요청한 사용자아이디
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "userid" : "userid123"
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : ""
}
  
```

DELETE /user/board/&lt;boardid&gt;/delete



## Parameter

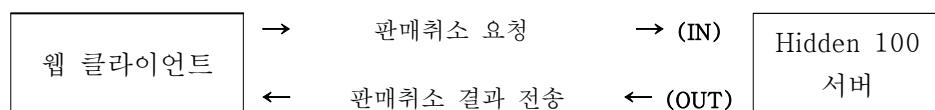
속성	전송 방향		Type	Description
	IN	OUT		
boardid	O		int	요청한 사용자아이디
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지

## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "boardid" : 12
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : ""
}
  
```

DELETE /seller/goods/&lt;goodid&gt;/delete



## Parameter

속성	전송 방향		Type	Description
	IN	OUT		
goodid	O		int	취소할 상품 아이디
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지

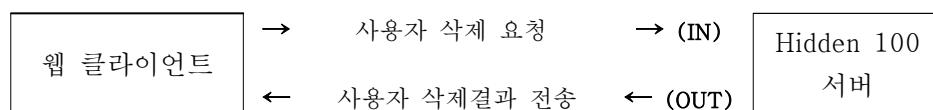
## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "goodid" : 12
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : ""
}

```

DELETE /admin/user/&lt;userid&gt;/delete



## Parameter

속성	전송 방향		Type	Description
	IN	OUT		
userid	O		String	삭제할 사용자 아이디
content_type		O	String	application/json
result_code		O	int	처리결과 코드(200: 성공, 600: 오류코드)
result_req		O	String	처리결과 오류메세지

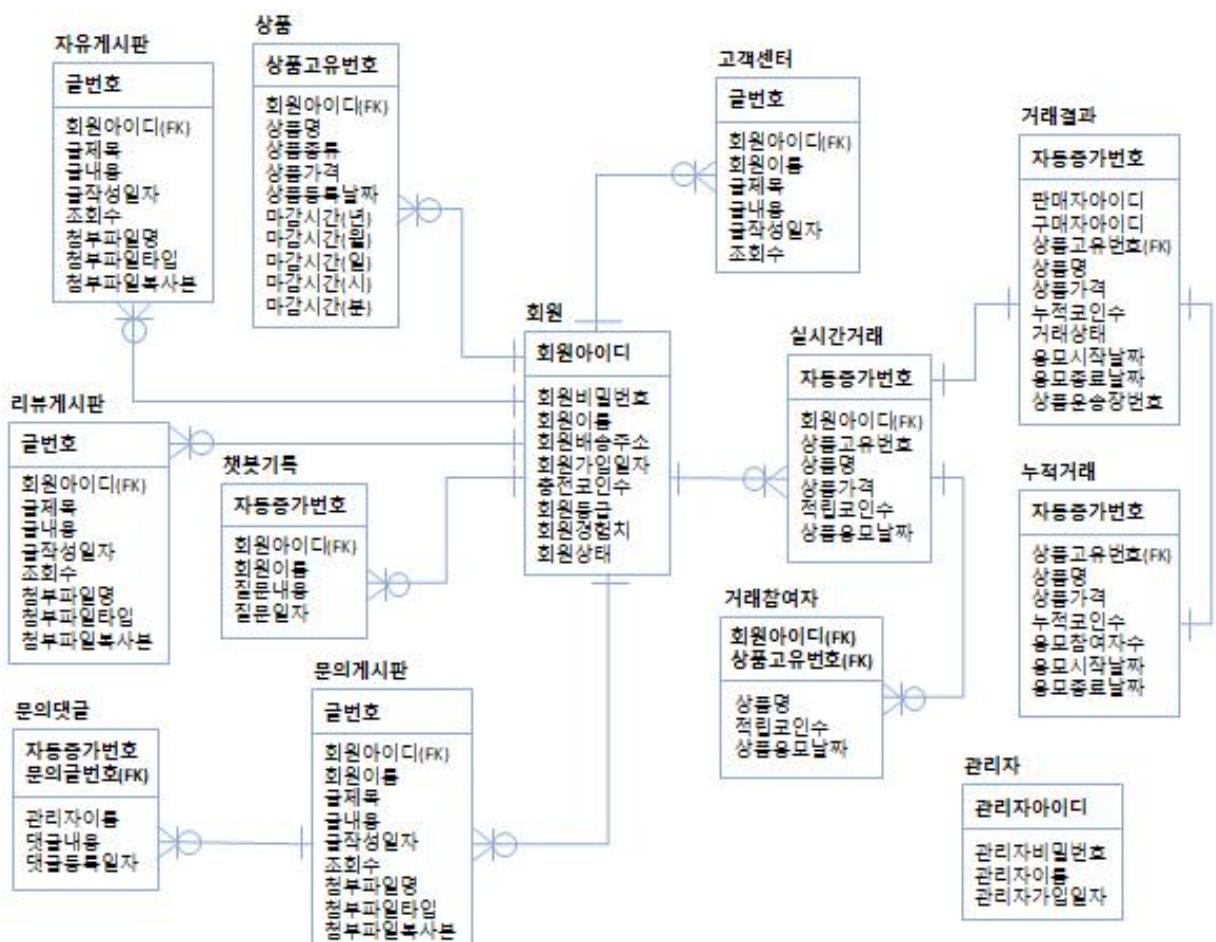
## 실제 전송내용 (JSON 의 경우라면)

```

전송방향 : IN
{
    "userid" : "userid123"
}
전송방향 : OUT
{
    "content_type" : "json" ,
    "result_code" : 200 ,
    "result_req" : ""
}
  
```

## 6. 데이터베이스 설계

### 6.1. ERD



## 6.2. 논리적 DB설계 (테이블명세서)

테이블명(회원) :			T1_MEMBER			
NO	Attribute	Data Type	NN	PK	FK	Description
1	ID	VARCHAR(15)	O	O		회원아이디
2	PASS	VARCHAR(20)	O			회원비밀번호
3	NAME	VARCHAR(15)	O			회원이름
4	ADDRESS	VARCHAR(100)	O			회원배송주소
5	REGIST_DAY	TIMESTAMP	O			회원가입일자
6	COIN	INTEGER	O			충전코인수
7	LEVEL	VARCHAR(10)	O			회원등급
8	POINT	INTEGER	O			회원경험치
9	STATUS	VARCHAR(10)	O			회원상태

테이블명(관리자) :			T1_ADMIN			
NO	Attribute	Data Type	NN	PK	FK	Description
1	ADMIN_ID	VARCHAR(15)	O	O		관리자아이디
2	ADMIN_PASS	VARCHAR(20)	O			관리자비밀번호
3	ADMIN_NAME	VARCHAR(15)	O			관리자이름
4	REGIST_DAY	TIMESTAMP	O			관리자가입일자

테이블명(상품) :			T1_GOODS			
NO	Attribute	Data Type	NN	PK	FK	Description
1	ID	INTEGER	O	O		상품고유번호
2	MEM_ID	VARCHAR(15)	O		O	회원아이디
3	NAME	VARCHAR(50)	O			상품명
4	TYPE	VARCHAR(50)	O			상품종류
5	PRICE	INTEGER	O			상품가격
6	REGIST_DAY	TIMESTAMP	O			상품등록날짜
7	TIME_YEAR	INTEGER	O			마감시간(년)
8	TIME_MONTH	INTEGER	O			마감시간(월)
9	TIME_DAY	INTEGER	O			마감시간(일)
10	TIME_HOUR	INTEGER	O			마감시간(시)
11	TIME_MINUTE	INTEGER	O			마감시간(분)

테이블명(실시간거래) :			T1 DEAL_ACTIVE			
NO	Attribute	Data Type	NN	PK	FK	Description
1	NUM	INTEGER	O	O		자동증가번호
2	MEM_ID	VARCHAR(15)	O		O	회원아이디
3	GOO_ID	INTEGER	O			상품고유번호
4	GOO_NAME	VARCHAR(50)	O			상품명
5	PRICE	INTEGER	O			상품가격
6	COIN	INTEGER	O			적립코인수
7	REGIST_DAY	TIMESTAMP	O			상품응모날짜

테이블명(누적거래) :			T1 DEAL TOTAL			
NO	Attribute	Data Type	NN	PK	FK	Description
1	NUM	INTEGER	O	O		자동증가번호
2	GOO_ID	INTEGER	O		O	상품고유번호
3	GOO_NAME	VARCHAR(50)	O			상품명
4	PRICE	INTEGER	O			상품가격
5	COIN_TOTAL	INTEGER	O			누적코인수
6	MEM_NUM	INTEGER	O			응모참여자수
7	START_DAY	TIMESTAMP	O			응모시작날짜
8	FIN_DAY	TIMESTAMP	O			응모종료날짜

테이블명(거래참여자) :			T1 DEAL PARTICIPATION			
NO	Attribute	Data Type	NN	PK	FK	Description
1	MEM_ID	VARCHAR(15)	O	O	O	회원아이디
2	GOO_ID	INTEGER	O	O	O	상품고유번호
3	GOO_NAME	VARCHAR(50)	O			상품명
4	COIN	INTEGER	O			적립코인수
5	REGIST_DAY	TIMESTAMP	O			상품응모날짜

테이블명(거래결과) :			T1 DEAL RESULT			
NO	Attribute	Data Type	NN	PK	FK	Description
1	NUM	INTEGER	O	O		자동증가번호
2	SELL_MEM_ID	VARCHAR(15)	O			판매자아이디
3	BUY_MEM_ID	VARCHAR(15)	O			구매자아이디
4	GOO_ID	INTEGER	O		O	상품고유번호
5	GOO_NAME	VARCHAR(50)	O			상품명
6	PRICE	INTEGER	O			상품가격
7	COIN_TOTAL	INTEGER	O			누적코인수
8	STATUS	VARCHAR(10)	O			거래상태
9	START_DAY	TIMESTAMP	O			응모시작날짜
10	FIN_DAY	TIMESTAMP	O			응모종료날짜
11	DELIV_NUM	INTEGER				상품운송장번호

테이블명(챗봇기록) :			T1 CHAT LOG			
NO	Attribute	Data Type	NN	PK	FK	Description
1	NUM	INTEGER	O	O		자동증가번호
2	MEM_ID	VARCHAR(15)	O		O	회원아이디
2	MEM_NAME	VARCHAR(15)	O			회원이름
3	CONTENT	TEXT	O			질문내용
4	REGIST_DAY	TIMESTAMP	O			질문일자

테이블명(자유게시판) :			T1_BOARD			
NO	Attribute	Data Type	NN	PK	FK	Description
1	NUM	INTEGER	O	O		글번호
2	MEM_ID	VARCHAR(15)	O		O	회원아이디
3	SUBJECT	VARCHAR(200)	O			글제목
4	CONTENT	TEXT	O			글내용
5	REGIST_DAY	TIMESTAMP	O			글작성일자
6	HIT	INTEGER	O			조회수
7	FILE_NAME	VARCHAR(40)				첨부파일명
8	FILE_TYPE	VARCHAR(40)				첨부파일타입
9	FILE_COPIED	VARCHAR(40)				첨부파일복사본

테이블명(문의게시판) :			T1_QUESTION			
NO	Attribute	Data Type	NN	PK	FK	Description
1	Q_NUM	INTEGER	O	O		글번호
2	MEM_ID	VARCHAR(15)	O		O	회원아이디
3	MEM_NAME	VARCHAR(20)	O			회원이름
4	SUBJECT	VARCHAR(200)	O			글제목
5	CONTENT	TEXT	O			글내용
6	REGIST_DAY	TIMESTAMP	O			글작성일자
7	HIT	INTEGER	O			조회수
8	FILE_NAME	VARCHAR(40)				첨부파일명
9	FILE_TYPE	VARCHAR(40)				첨부파일타입
10	FILE_COPIED	VARCHAR(40)				첨부파일복사본

테이블명(문의댓글) :			T1_Q_COMMENT			
NO	Attribute	Data Type	NN	PK	FK	Description
1	NUM	INTEGER	O	O		자동증가번호
2	QUEST_NUM	INTEGER	O		O	문의글번호
3	ADMIN_NAME	VARCHAR(20)	O			관리자이름
4	CONTENT	TEXT	O			댓글내용
5	REGIST_DAY	TIMESTAMP	O			댓글등록일자

테이블명(리뷰게시판) :			T1 REVIEW			
NO	Attribute	Data Type	NN	PK	FK	Description
1	NUM	INTEGER	O	O		글번호
2	MEM_ID	VARCHAR(15)	O		O	회원아이디
3	SUBJECT	VARCHAR(200)	O			글제목
4	CONTENT	TEXT	O			글내용
5	REGIST_DAY	TIMESTAMP	O			글작성일자
6	HIT	INTEGER	O			조회수
7	FILE_NAME	VARCHAR(40)				첨부파일명
8	FILE_TYPE	VARCHAR(40)				첨부파일타입
9	FILE_COPIED	VARCHAR(40)				첨부파일복사본

테이블명(고객센터) :			T1_C_SERVICE			
NO	Attribute	Data Type	NN	PK	FK	Description
1	NUM	INTEGER	O	O		글번호
2	MEM_ID	VARCHAR(15)	O		O	회원아이디
3	MEM_NAME	VARCHAR(20)	O			회원이름
4	SUBJECT	VARCHAR(200)	O			글제목
5	CONTENT	TEXT	O			글내용
6	REGIST_DAY	TIMESTAMP	O			글작성일자
7	HIT	INTEGER	O			조회수

### 6.3. 물리적 DB설계 (SQL스크립트)

테이블명	T1_MEMBER
<pre>CREATE TABLE t1_member (     id varchar(15) NOT NULL,     pass varchar(20) NOT NULL,     name varchar(15) NOT NULL,     address varchar(100) NOT NULL,     regist_day timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,     coin integer NOT NULL,     level varchar(10) NOT NULL,     point integer NOT NULL,     status varchar(10) NOT NULL,     PRIMARY KEY (id) );</pre>	

테이블명	T1_ADMIN
<pre>CREATE TABLE t1_admin (     admin_id varchar(15) NOT NULL UNIQUE,     admin_pass varchar(20) NOT NULL,     admin_name varchar(15) NOT NULL,     regist_day timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,     PRIMARY KEY (admin_id) );</pre>	

테이블명	T1_GOODS
------	----------

```

CREATE TABLE t1_goods
(
    id integer NOT NULL UNIQUE,
    mem_id varchar(15) NOT NULL,
    name varchar(50) NOT NULL,
    type varchar(50) NOT NULL,
    price integer NOT NULL,
    regist_day timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
    time_year integer NOT NULL,
    time_month integer NOT NULL,
    time_day integer NOT NULL,
    time_hour integer NOT NULL,
    time_minute integer NOT NULL,
    PRIMARY KEY (id),
    FOREIGN KEY (mem_id)
    REFERENCES member (id)
);

```

테이블명	T1 DEAL ACTIVE
------	----------------

```

CREATE TABLE t1_deal_active
(
    num integer NOT NULL AUTO_INCREMENT,
    mem_id varchar(15) NOT NULL,
    goo_id integer NOT NULL UNIQUE,
    goo_name varchar(50) NOT NULL,
    price integer NOT NULL,
    coin integer NOT NULL,
    regist_day timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (num),
    FOREIGN KEY (mem_id)
    REFERENCES member (id)
);

```

테이블명	T1 DEAL TOTAL
------	---------------

```
CREATE TABLE t1_deal_total
(
    num integer NOT NULL AUTO_INCREMENT,
    goo_id integer NOT NULL UNIQUE,
    goo_name varchar(50) NOT NULL,
    price integer NOT NULL,
    coin_total integer NOT NULL,
    mem_num integer NOT NULL,
    start_day timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
    fin_day timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (num),
    FOREIGN KEY (goo_id)
    REFERENCES deal_result (goo_id)
);
```

테이블명	T1 DEAL PARTICIPATION
------	-----------------------

```
CREATE TABLE t1_deal_participation
(
    mem_id varchar(15) NOT NULL,
    goo_id integer NOT NULL UNIQUE,
    goo_name varchar(50) NOT NULL,
    coin integer NOT NULL,
    regist_day timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (mem_id, goo_id),
    FOREIGN KEY (mem_id, goo_id)
    REFERENCES deal_active (mem_id, goo_id)
);
```

테이블명	T1 DEAL RESULT
------	----------------

```

CREATE TABLE t1_deal_result
(
    num integer NOT NULL AUTO_INCREMENT,
    sell_mem_id varchar(15) NOT NULL,
    buy_mem_id varchar(15) NOT NULL,
    goo_id integer NOT NULL UNIQUE,
    goo_name varchar(50) NOT NULL,
    price integer NOT NULL,
    coin_total integer NOT NULL,
    status varchar(10) NOT NULL,
    start_day timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
    fin_day timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
    deliv_num integer,
    PRIMARY KEY (num),
    FOREIGN KEY (goo_id)
    REFERENCES deal_active (goo_id)
);

```

테이블명	T1 CHAT LOG
------	-------------

```

CREATE TABLE t1_chat_log
(
    num integer NOT NULL AUTO_INCREMENT,
    mem_id varchar(15) NOT NULL,
    mem_name varchar(15) NOT NULL,
    content text NOT NULL,
    regist_day timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (num),
    FOREIGN KEY (mem_id)
    REFERENCES member (id)
);

```

테이블명 | T1\_BOARD

```
CREATE TABLE t1_board
(
    num integer NOT NULL AUTO_INCREMENT,
    mem_id varchar(15) NOT NULL,
    subject varchar(200) NOT NULL,
    content text NOT NULL,
    regist_day timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
    hit integer NOT NULL,
    file_name varchar(40),
    file_type varchar(40),
    file_copied varchar(40),
    PRIMARY KEY (num),
    FOREIGN KEY (mem_id)
    REFERENCES member (id)
);
```

테이블명 | T1\_QUESTION

```
CREATE TABLE t1_question
(
    q_num integer NOT NULL AUTO_INCREMENT,
    mem_id varchar(15) NOT NULL,
    mem_name varchar(20) NOT NULL,
    subject varchar(200) NOT NULL,
    content text NOT NULL,
    regist_day timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
    hit integer NOT NULL,
    file_name varchar(40),
    file_type varchar(40),
    file_copied varchar(40),
    PRIMARY KEY (q_num),
    FOREIGN KEY (mem_id)
    REFERENCES member (id)
);
```

테이블명 | T1\_Q\_COMMENT

```
CREATE TABLE t1_q_comment
(
    num integer NOT NULL AUTO_INCREMENT,
    quest_num integer NOT NULL UNIQUE,
    admin_name varchar(20) NOT NULL,
    content text NOT NULL,
    regist_day timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (num),
    FOREIGN KEY (quest_num)
    REFERENCES question (q_num)
);
```

테이블명 | T1 REVIEW

```
CREATE TABLE t1_review
(
    num integer NOT NULL AUTO_INCREMENT,
    mem_id varchar(15) NOT NULL,
    subject varchar(200) NOT NULL,
    content text NOT NULL,
    regist_day timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
    hit integer NOT NULL,
    file_name varchar(40),
    file_type varchar(40),
    file_copied varchar(40),
    PRIMARY KEY (num),
    FOREIGN KEY (mem_id)
    REFERENCES member (id)
);
```

테이블명 | T1\_C\_SERVICE

```
CREATE TABLE t1_c_service
(
    num integer NOT NULL AUTO_INCREMENT,
    mem_id varchar(15) NOT NULL,
    mem_name varchar(20) NOT NULL,
    subject varchar(200) NOT NULL,
    content text NOT NULL,
    regist_day timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
    hit integer NOT NULL,
    PRIMARY KEY (num),
    FOREIGN KEY (mem_id)
        REFERENCES member (id)
);
```

## 7. 환경구성

### 7.1. 개발환경 및 운영환경

#### ■ 서버

구분	개발환경	규격	비고
개발툴	편집툴	ATOM	
	개발IDE	VSCODE, BRAKET	
	프레임워크	Node.js	
	Express Router Jade Mysql Body-parser Http .		
서버	TCP Port주소		
	IP주소		
	OS	CentosOS	
	DB	MariaDB	
서버 하드웨어	CPU	Intel Core2Duo	
	RAM	4G RAM	
	DISK	500G DISK (SATAII)	
	Network	100Mbps	

#### ■ 클라이언트

구분	개발환경	규격	비고
클라이언트	OS	Window/Linux/macOs	

## 7.2. 소스디렉터리 구조

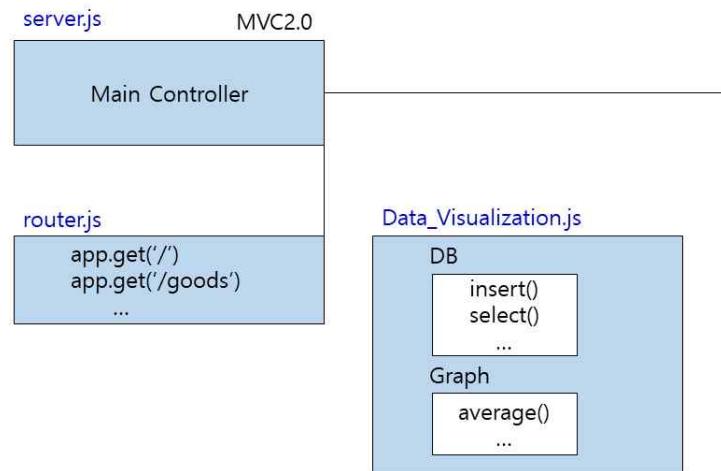


그림 7.1 디렉토리 구조

그림 7.1과 같이 MVC 구조로 개발 디렉토리 구조를 정의하며, 그림 7.2와 같이 세부 디렉터리와 파일명으로 모듈화를 정의한다.

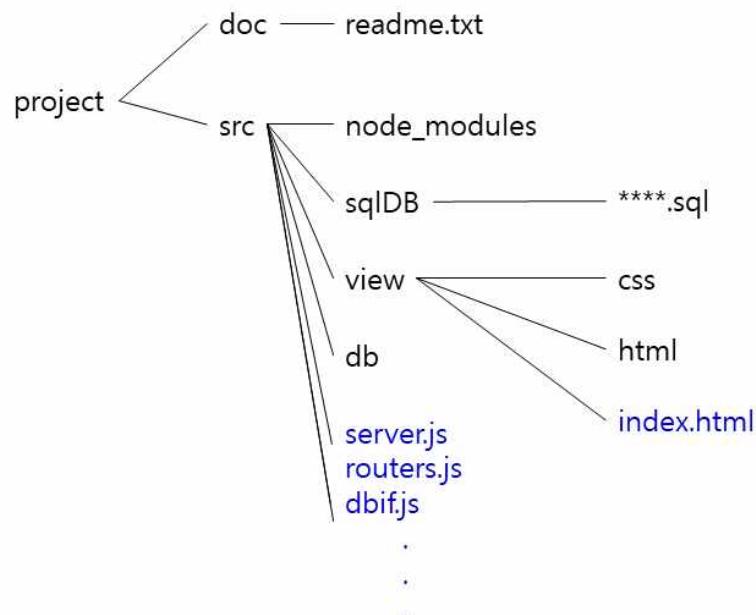


그림 7.2 디렉터리 및 파일환경