

M Zone

Metaverse Zone

MZ Team

한승은 김혜린
윤지영 노지의
박가영

목차

01
기획의도

02
USECASE DIAGRAM

03
구조도 및 ERD

04
업무분담

05
세부기능

06
개발환경 및 작업 일정 표

기획의도

Meta
초월 / 가공
= 구현 **Universe**
현실세계 또는 우주

Meta Verse

현실같이 구현한 가상 세계
현실과 온라인의 연결공간

서비스

직간접적 소통을 통한
사회적 교류공간

새로운 친구를
만날 수 있는 가상공간

미니게임을 통한
놀이 공간

현실 친구를
가상에서 만나는 재미

유저들의
실시간 동시접속

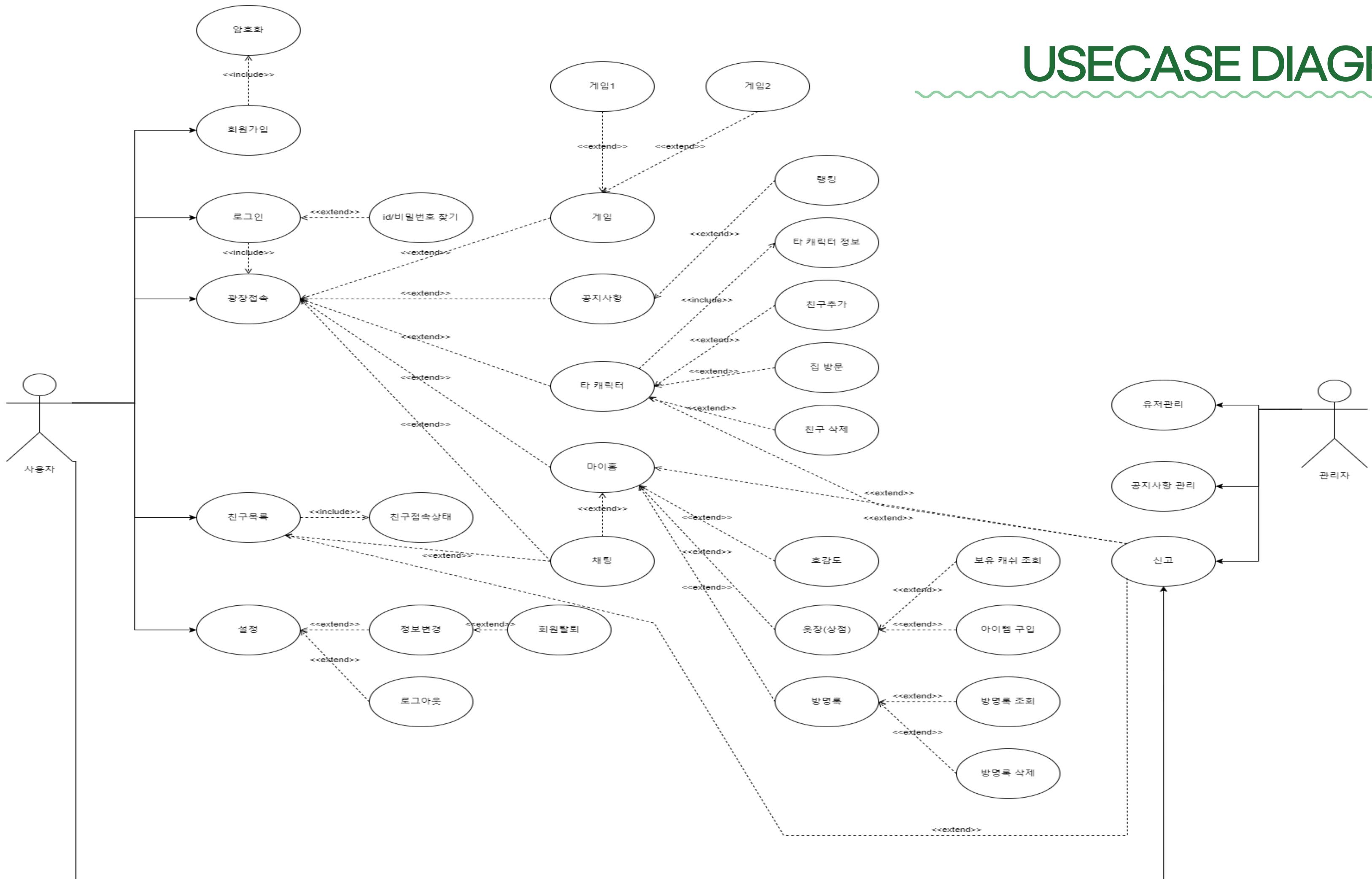
개발

API 연동 경험

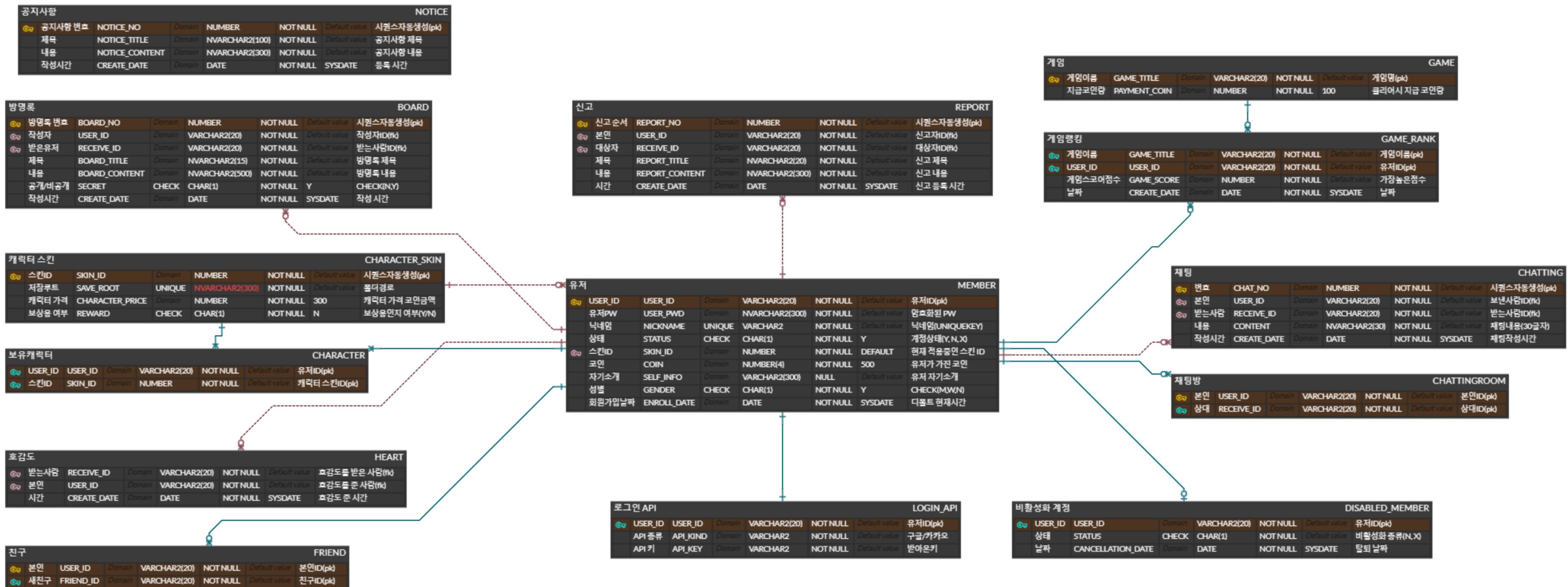
미니게임
제작 & 구현

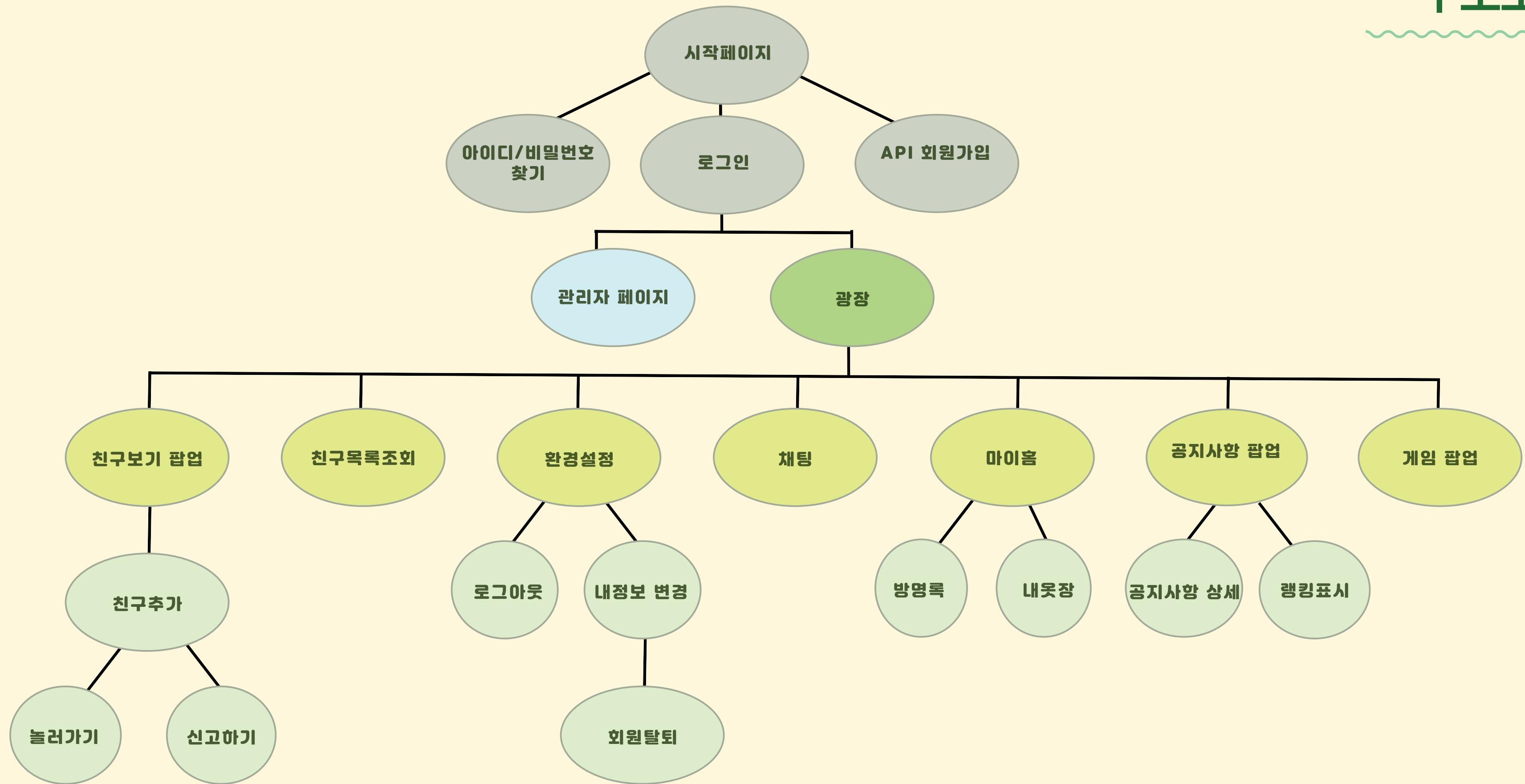
웹소켓을 활용한
채팅 가능

USECASE DIAGRAM

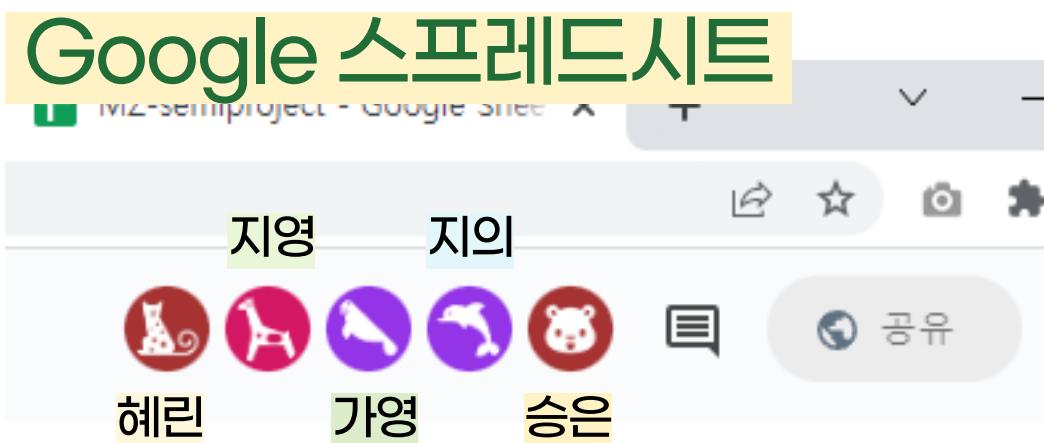
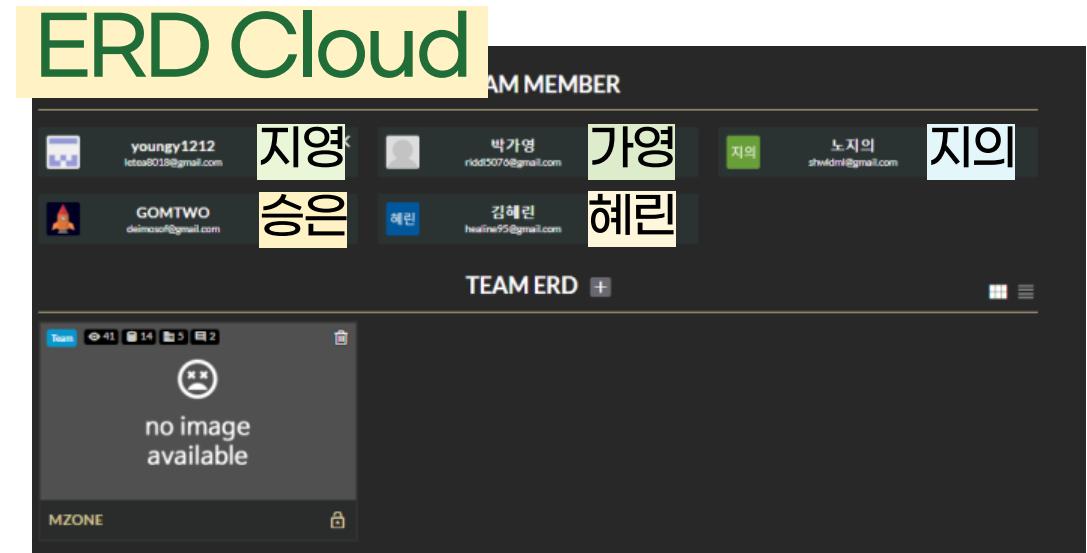


ERD





협업툴



이름 ↑	수정일 ↑
MZone 기획 보고서 - 최종판	2023. 4. 20. 윤지영
MZone - 윤지영	2023. 4. 20. 윤지영
MZone - 한승은	2023. 4. 20. .
MZone - 박가영	2023. 4. 20. 박가영
MZone - 김혜린	2023. 4. 20. 김혜린
MZone - 노지의	2023. 4. 20. 노지의

클릭

Github

ERD Cloud

Google Sheets

미리캔버스

JSP - module 지정

```
<script type="module" src="<%=contextPath%>/resource/js/squareinit.js"></script>
<script type="module" src="<%=contextPath%>/resource/js/alert.js"></script>
```

JavaScript - export

```
export function getContextPath() {
  let hostIndex = location.href.indexOf(location.host) + location.host.length;
  let contextPath = location.href.substring(
    hostIndex,
    location.href.indexOf("/", hostIndex + 1)
  );
  return contextPath;
}
```

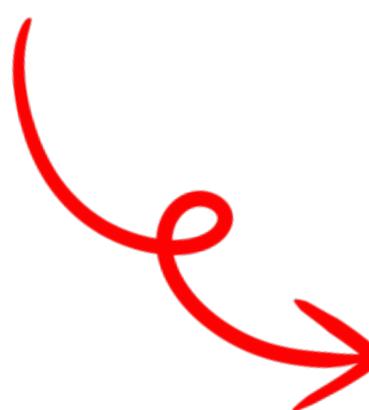


import 와 export

- 함수/변수들을 export 하여 여러 js파일에서 공용으로 사용
- 공용 코드는 common.js와 같은 식으로 파일을 분리시켜 사용함으로서 코드의 중복을 최소화시킴
- 서로의 코드를 공유함으로서 로직 구현 시간을 단축 시킴

JavaScript - import

```
import { getContextPath, getSessionStorage } from './common.js';
import { modalstopfn } from './squareCanvas.js';
import { openChatRoom } from './chat/chatFront.js';
import * as Alert from "./alert.js";
import { homeOpenAlert } from "./homeAlert.js";
```



업무분담

김혜린

회원가입

로그인 / API

아이디 저장
자동로그인

내 정보 변경

닉네임 변경
비밀번호 변경
성별체크
자기소개 작성

회원탈퇴

윤지영

광장 접속

유저 멀티 접속 기능
접속자 위치 연동
광장 이벤트 연동
맵 오브젝트 블락
(유저의 이동 가능한
공간/불가능한 공간)

사용자 목록

친구 목록 조회
로그아웃

미니게임 제작 & 기능구현

한승은

채팅

전체 채팅
1:1 채팅

관리자 계정

유저 관리
공지사항 관리
캐릭터 스킨 관리
기타 데이터 관리

박가영

정보 조회

친구 추가
친구 마이홈 방문
친구 삭제
친구 신고
호감도 표시

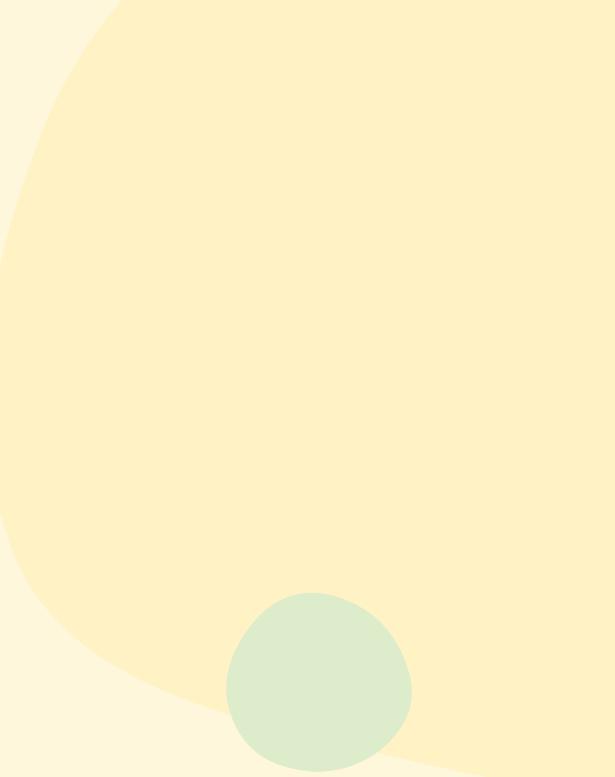
공지사항

호감도 랭킹 조회
공지사항 조회

노지의

마이홈

마이룸 유저 표시
호감도
닉네임
방명록
조회
작성
수정
삭제
옷장 / 상점
스킨 교체
아이템 구입



김혜린



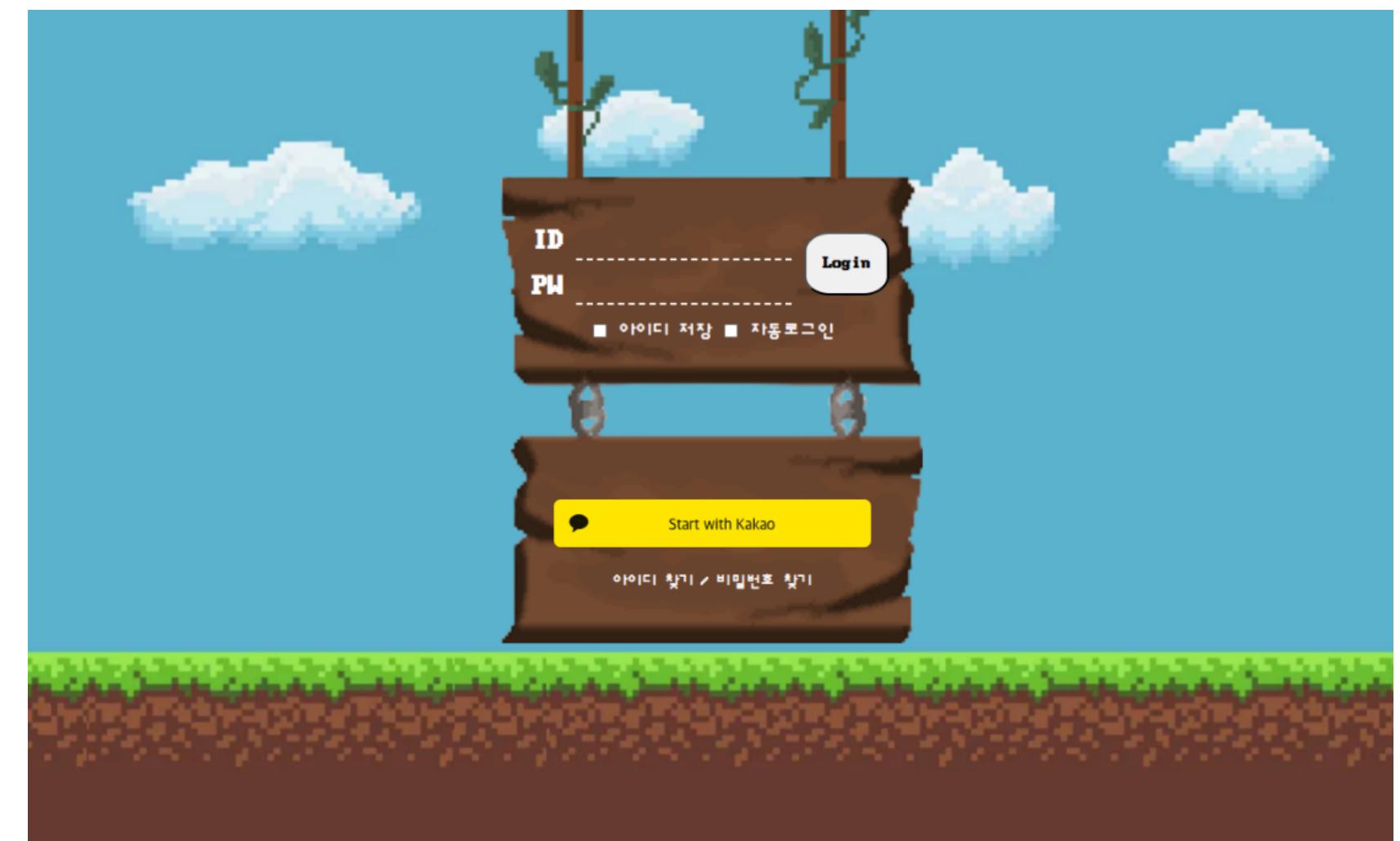
Main



메인 로고 제작 / 로고 버튼 클릭 이벤트



로그인 패널 SlideDown



카카오 API



[main]



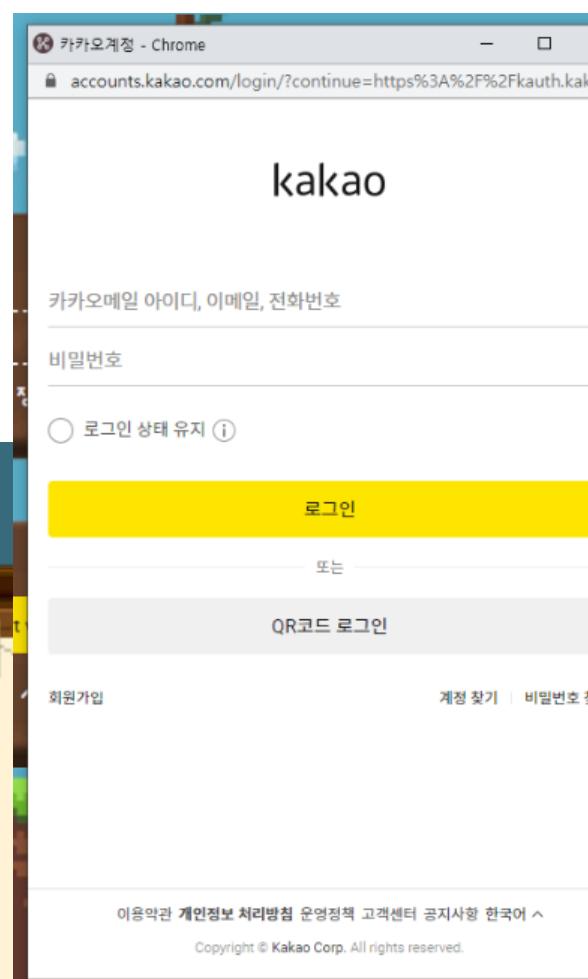
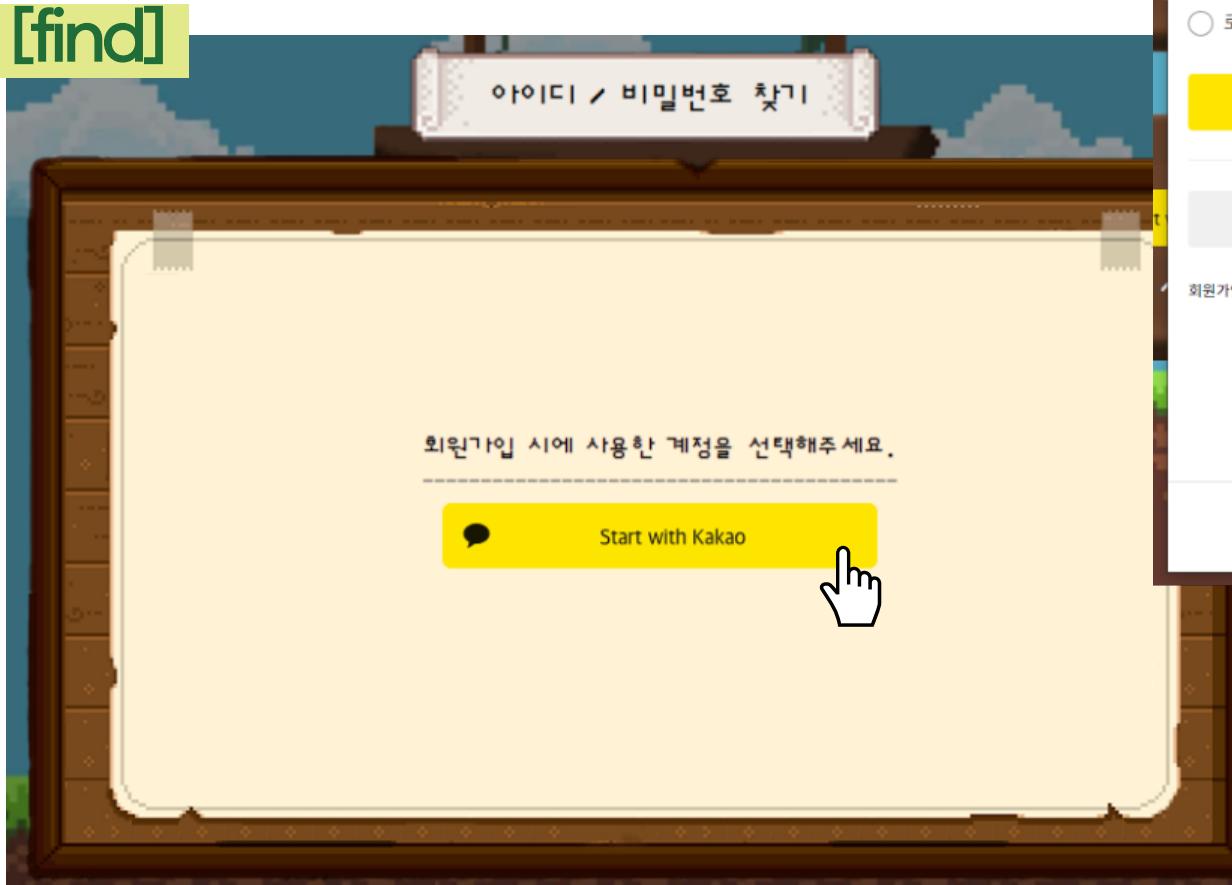
- 카카오 계정을 본인 인증 수단으로 사용

DB 내 계정(KEY) 존재 유무와 사용처에 따라 다르게 기능

[main] 회원가입 or 카카오 계정으로 로그인

[find] 아이디찾기 / 비밀번호 재설정

[find]



카카오 API 계정 로그인

```
// 카카오로그인
function kakaoLogin(page) {
  Kakao.Auth.login({
    success: function (response) {
      Kakao.API.request({
        url: '/v2/user/me',
        success: function (response) {
          //console.log(response.id);
        }
      });
    },
    fail: function (error) {
      console.log(error);
    }
  });
  checkKakao(response.id, page);
}

// page: main(계정로그인, 회원가입) // page: find(아이디, 패스워드 찾기)
```

```
// 카카오 계정으로 존재하는 회원인지 확인하는 함수
let checkKakao = function(key, page){
  if(page == "main"){
    // 쿠키에 키 저장 => 회원가입
  }
  if(page == "main"){
    // 메인 => 광장으로 이동시키기
  }
  if(page == "find"){
    // 비밀번호 재설정 모달 열기
  }
}
```

카카오 버튼 - [main]

DB에 API(key) 존재



[main] → [square]

카카오계정으로 로그인 처리

session에 loginUser 정보 저장



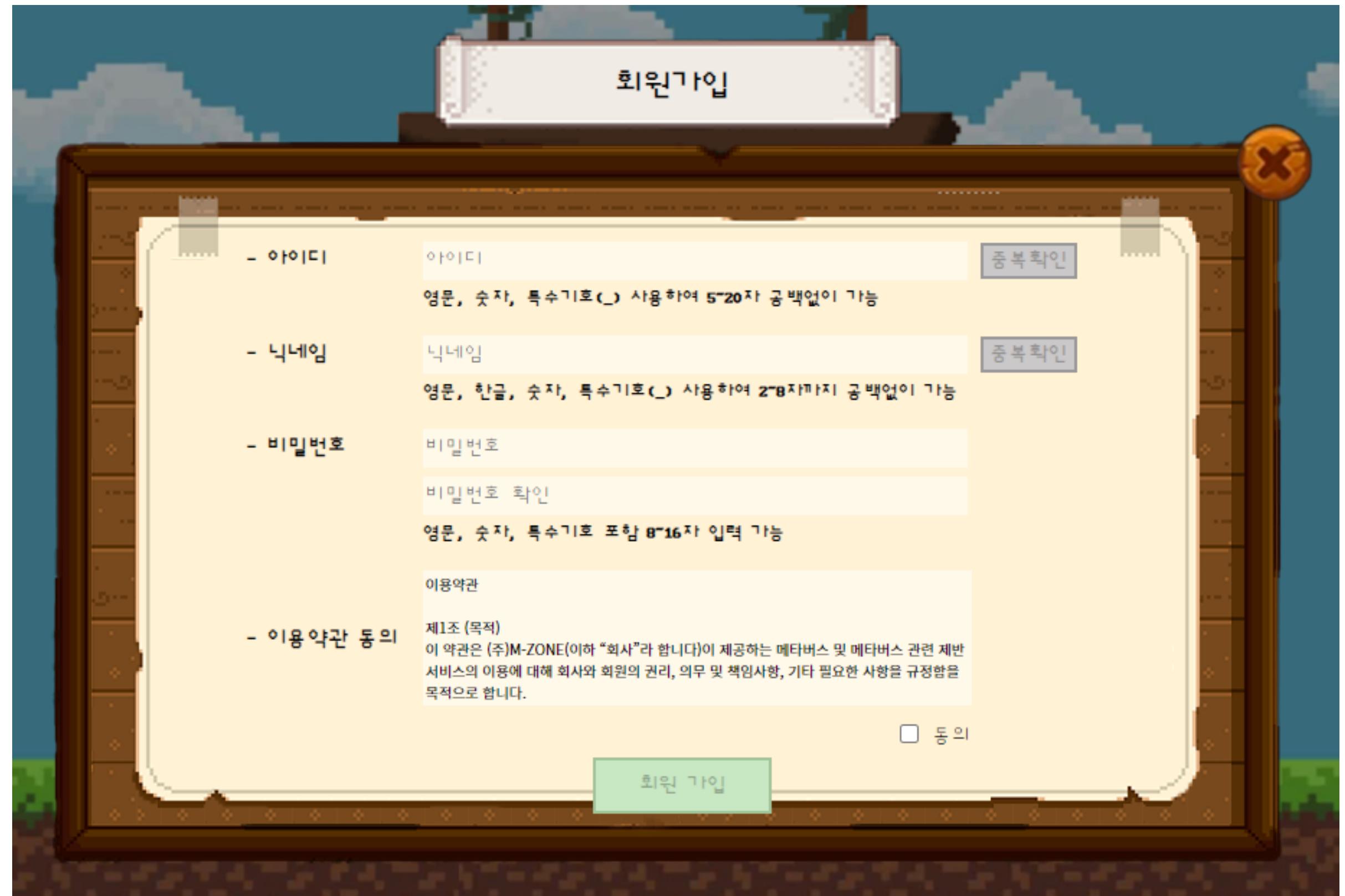
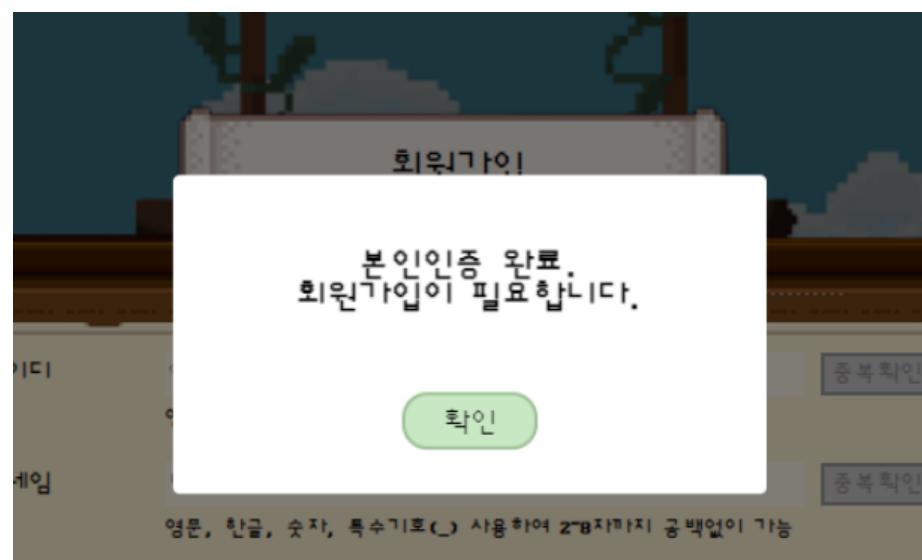
카카오 버튼 - [main]

DB에 API(key) 미존재

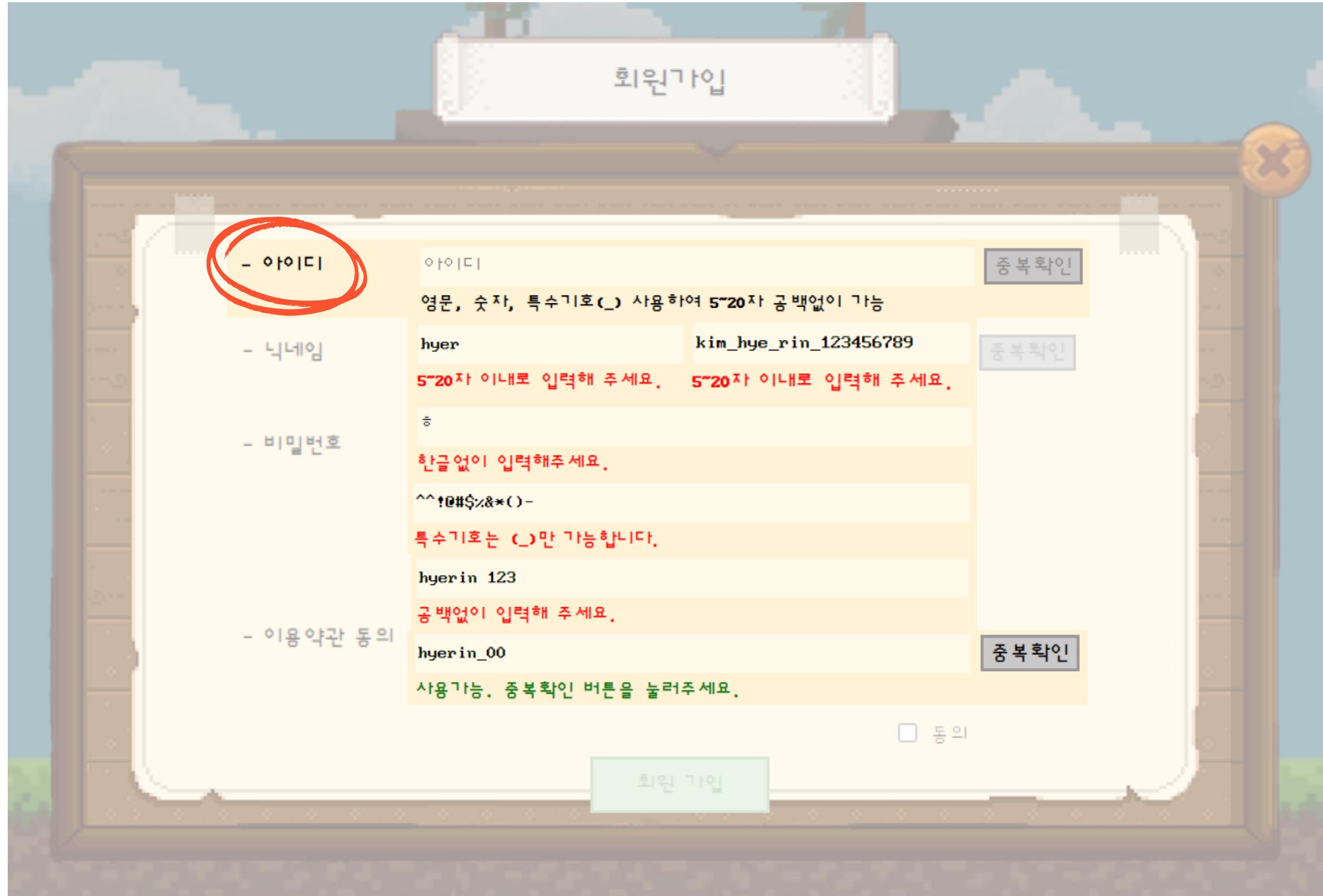


alert / 회원가입 모달 열림

+ 계정 본인인증과 동시에 cookie에 사용자의 key와 API의 종류를 저장



회원가입



- 유효성 검사

(정규식으로 정합성 체크 후 하단 text 표기)

아이디

- 영문, 숫자, 특수기호(_) 사용 가능
- 5~20 자 제한
- 공백 불가

닉네임

- 한글, 영문, 숫자, 특수기호(_) 사용 가능
- 2~8 자 제한
- 공백 불가

비밀번호

- 영문, 숫자, 특수문자 필수 포함
- 8~16 자 제한
- 비밀번호 확인 값과 일치

회원가입



The screenshot shows a registration form with the following fields:

- 아이디**: Input field containing "혜". Error message: "닉네임은 혜린킹왕짱" (Nickname cannot be Hyelin King King). Confirmation button: "중복 확인".
- 닉네임**: Input field containing "혜". Error message: "2~8자 이내로 입력해 주세요." (Please enter 2~8 characters). Confirmation button: "중복 확인".
- 비밀번호**: Input field containing "혜린_최고암". Error message: "한글을 올바르게 입력해 주세요." (Please enter Hangeul correctly). Confirmation button: "중복 확인".
- 이용 약관 동의**: Input field containing "혜린_짱짱". Error message: "특수기호는 _만 가능합니다." (Only underscores are allowed). Confirmation button: "중복 확인".

At the bottom, there is a checkbox labeled "동의" (Agree) and a green "회원 가입" (Join Member) button.

- 유효성 검사

(정규식으로 정합성 체크 후 하단 text 표기)

아이디

- 영문, 숫자, 특수기호(_) 사용 가능
- 5~20 자 제한
- 공백 불가

닉네임

- 한글, 영문, 숫자, 특수기호(_) 사용 가능
- 2~8 자 제한
- 공백 불가

비밀번호

- 영문, 숫자, 특수문자 필수 포함
- 8~16 자 제한
- 비밀번호 확인 값과 일치

회원가입



회원가입

- 아이디 아이디 중복 확인

영문, 숫자, 특수기호(_) 사용하여 5~20자 공백없이 가능

..... = hyerin
비밀번호 확인 비밀번호 확인
8~16자 이내로 입력해 주세요. 8~16자 이내로 입력해 주세요.

- 닉네임
비밀번호 확인

- 비밀번호
비밀번호 확인

영문, 숫자, 특수기호 포함 8~16자 입력 가능

..... = hyerin00 = hyerin^^
비밀번호 확인 비밀번호 확인
특수기호를 넣어 입력해 주세요. 숫자를 넣어 입력해 주세요.

..... = 1234567! = hyerinhyerin
비밀번호 확인 or 12345678 / or !@#\$%^&*

영문자를 넣어 입력해 주세요. 영문, 숫자, 특수기호를 혼합하여 입력해 주세요.

..... = hyerin_0 = hyerin_0
비밀번호 확인 -> or = hyerin_0
사용 가능. 비밀번호 확인란에 동일하게 입력해 주세요. 입력한 비밀번호가 일치하지 않습니다. 일치

- 유효성 검사

(정규식으로 정합성 체크 후 하단 text 표기)

아이디

- 영문, 숫자, 특수기호(_) 사용 가능
- 5~20 자 제한
- 공백 불가

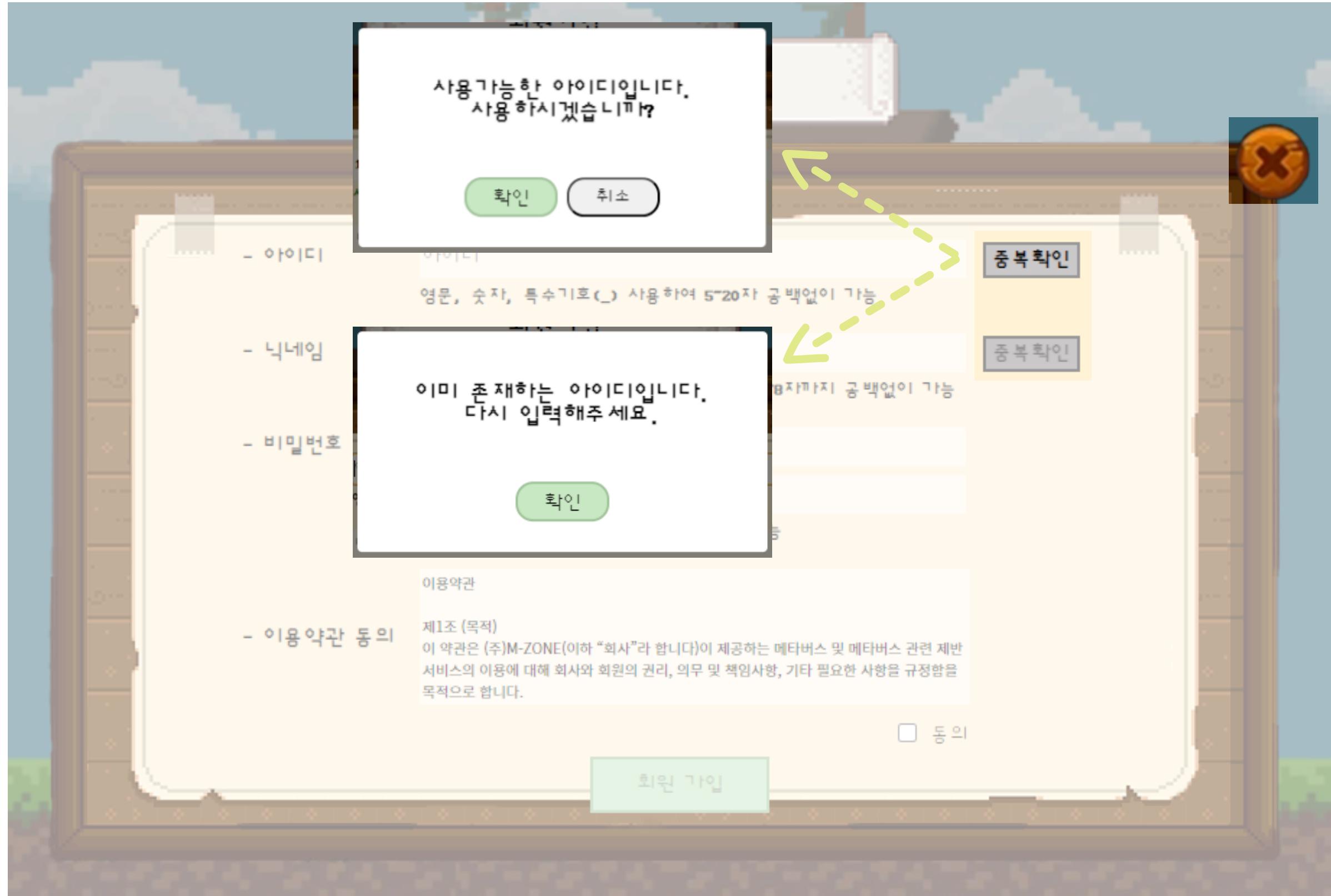
닉네임

- 한글, 영문, 숫자, 특수기호(_) 사용 가능
- 2~8 자 제한
- 공백 불가

비밀번호

- 영문, 숫자, 특수문자 필수 포함
- 8~16 자 제한
- 비밀번호 확인 값과 일치

회원가입



- 중복확인 버튼 (아이디 / 닉네임)

사용자 입력 keyup에 따라

- 유효성 검사 조건 만족 시 [중복확인 버튼] 활성화
- 유효성 검사 불만족 시 [중복확인 버튼] 비활성화

활성화 된 [중복확인 버튼] 클릭 시 DB에서 중복된 아이디/닉네임 유무 확인 후 confirm or alert 처리

중복X, 사용 가능 confirm 창

- 확인 버튼
 - 회원가입 버튼 활성화 조건 중 [중복버튼 조건] 만족
- 취소 버튼
 - 해당 text란으로 focus / 이전 입력흔적 초기화

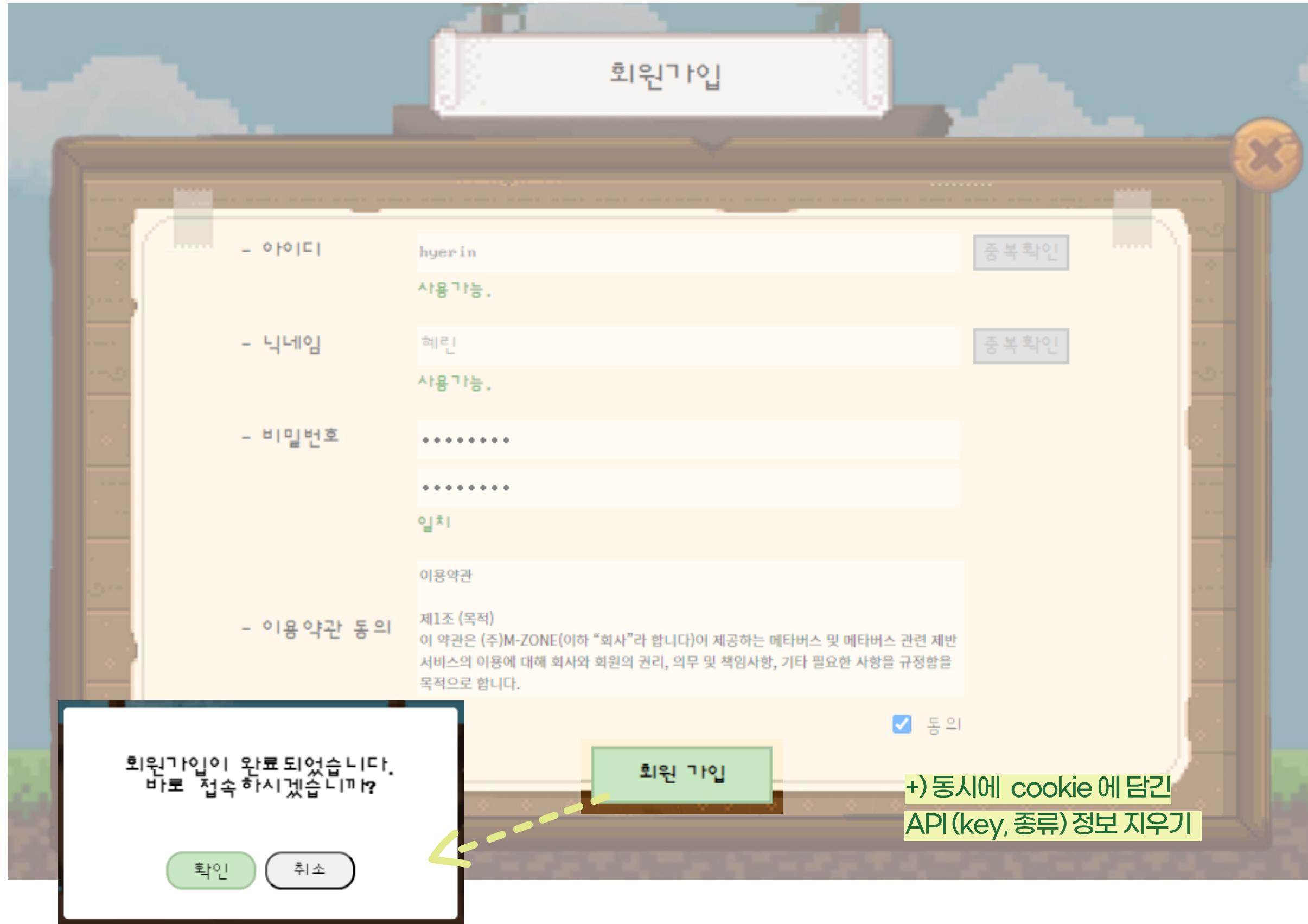
중복O, 사용 불가 alert 창

- 확인 버튼
 - 해당 text란으로 focus / 이전 입력흔적 초기화

- X 버튼 (모달 창 닫기)

display (block / none) 특성 상 모달 안의 입력 흔적, 버튼 활성화 조건들이 그대로 남아있어 따로 초기화 처리
입력 중에 모달을 닫고 다시 계정 인증 후 회원가입 모달이 열렸을 때 모달 내 모든 요소들 리셋 (초기화)

회원가입



- 회원가입 버튼

회원가입 모달 내 모든 조건 만족 시 [회원가입 버튼] 활성화

- 아이디 유효성 검사 조건 충족
- 아이디 중복확인 체크
- 닉네임 유효성 검사 조건 충족
- 닉네임 중복확인 체크
- 비밀번호 / 비밀번호 확인 유효성 검사 충족
- 비밀번호, 비밀번호 확인 일치
- 이용약관 동의 체크

위 조건 1개라도 불만족 시 [회원가입 버튼] 비활성화

활성화 된 [회원가입 버튼] 클릭 시 confirm 창

- 확인 버튼
 - 바로 로그인 처리 [main] -> [square]
- 취소 버튼
 - 회원가입 모달 닫히고 로그인 창

DB 처리 및 session에 loginUser 정보 저장

- MEMBER 테이블 INSERT
 - 비밀번호 암호화 처리
- CHARACTER 테이블 INSERT
- LOGIN_API 테이블 INSERT
 - API 키 암호화 처리

아이디 / 비밀번호 찾기

[find]



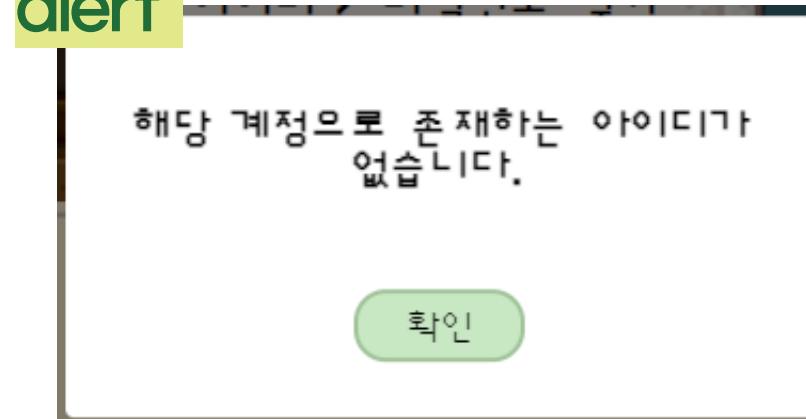
DB에 API(key) 존재



비밀번호 재설정 모달 열림



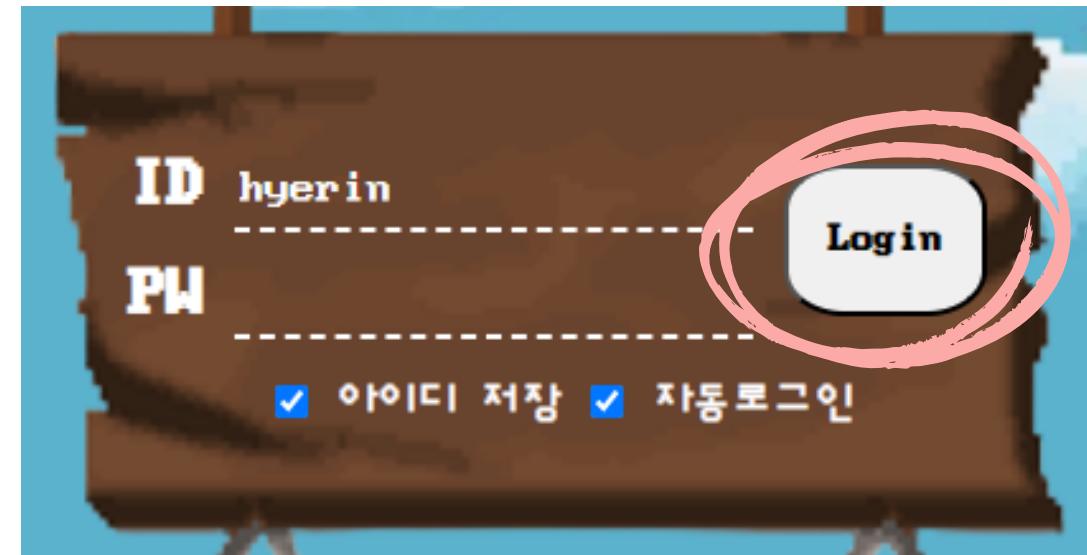
alert



- 비밀번호 재설정

- key값으로 DB에서 조회한 유저 아이디 상단에 표시
- 비밀번호 유효성 만족 시 (회원가입과 동일)
[비밀번호 재설정 버튼] 활성화
- 활성화된 [비밀번호 재설정 버튼] 클릭 시 바뀐 비밀번호로
로그인 할 수 있도록 alert과 함께 모든 모달이 닫힘
- DB에 새로운 비밀번호 암호화 후 UPDATE

기본 로그인



keyup 이벤트 (enter 키)로
Login 버튼 클릭과 같은 일
처리 (편리한 사용)

- 아이디 저장

체크 후 로그인 시 다음 로그인부터 아이디 표시

체크 해제 후 로그인 시 아이디 미표시

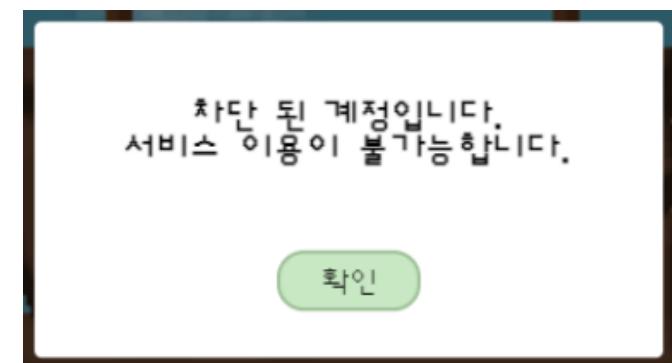
(localStorage 사용)

- 자동 로그인

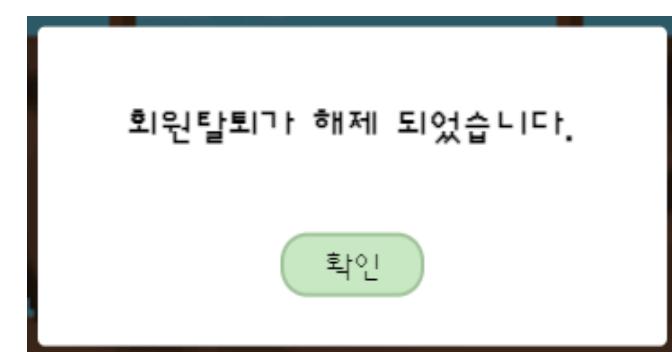
체크 후 로그인 시 웹을 닫고 새로 열거나, 서버를 껐다가
켰을 경우에 storage에 저장된 아이디값으로 자동 로그인
처리 (localStorage 사용)

사용자가 서비스 중에 로그아웃을 하면 storage 내의
정보가 사라지고 자동로그인 취소처리 (체크해제)

- 일반 계정 (STATUS = Y)
아이디 / 패스워드 입력 후 DB와 일치 시 로그인 처리
아이디 / 패스워드 불일치 시 alert 처리
session에 loginUser 정보 저장



- 차단 계정 (STATUS = X)
입력한 아이디가 차단된 계정일 경우
차단된 계정으로 사용이 불가함을 알리는 alert 처리



- 탈퇴 계정 (STATUS = N)
탈퇴한 계정이 탈퇴일로부터 15일 이내에 재로그인 했을
경우 계정 복구 처리 (STATUS = Y로 UPDATE) 및
탈퇴 해제 alert / 로그인 처리
session에 loginUser 정보 저장

(계정 종류에 따른 처리는 카카오계정으로 로그인도 동일)

- 관리자 계정 (admin)
관리자 계정의 아이디 / 패스워드 일치 시
관리자페이지로 이동

내 정보 변경



[square]



keyup 이벤트 (enter 키)로
확인 버튼 클릭과 같은 일 처리
(편리한 사용)

비밀번호 불일치

alert

비밀번호가 일치하지 않습니다.
다시 확인해주 세요.

확인

비밀번호 일치



- DB에서 실시간 유저 패스워드 값과 비교

DB 정보가 UPDATE 되는 모든 부분이 모달로 되어있어
실제 DB에는 UPDATE 처리되었으나 보여지는 화면 속
에서는 새로고침이 일어나지 않아 MEMBER 정보 데이터
를 사용하는 대부분의 경우에 즉시 DB에서 조회하여 사용

내 정보 변경



성별 여(W) 남(M) 비공개(N)

자기소개

성별

- 실시간 DB에서의 유저 정보 표시

처음 모달이 열렸을 때

입력 중에 모달을 닫고 다시 열었을 때 (+ 모달 내용 리셋)

정보 수정 후 모달을 닫고 다시 열었을 때

- 특이사항

성별 표시에 radio 사용

radio의 선택자로 checked 표시하는데 어려움이 있었음 DB와 연동을 하더라도 실제 마지막으로 사용자가 체크한 값이 모달 위에 남아있기 때문에 모달을 닫고 열었을 때 DB와 불일치하는 경우가 생김

- 해결방안

radio의 label에 클릭함수를 넣어서 간단히 적용

```
if(m.gender == "M"){
    $("label[for='M']").click();
} else if(m.gender == "W"){
    $("label[for='W']").click();
} else{
    $("label[for='N']").click();
}
```

내 정보 변경



내 정보 변경

- 아이디 hyerin

- 닉네임 변경 혜린
영문, 한글, 숫자, 특수기호(_) 사용하여 2~8자까지 공백없이 가능

- 비밀번호 변경 비밀번호
비밀번호 확인
영문, 숫자, 특수기호 포함 8~16자 입력 가능

- 성별 ♂ 여 ♂ 남 ♂ 비공개
저는 로그인, 회원 가입, 정보 변경, 탈퇴를 막았어요~!
발표 준비를 위해 PPT 제작을 하고 있어요. 하하.
저는 로그인, 회원 가입, 정보 변경, 탈퇴를 막았어요~!
발표 준

- 자기소개

100자 초과.

정보수정

회원가입 버튼과는 다르게
활성화 상태가 기본값

100자 이내로 작성해 주세요.

정보수정

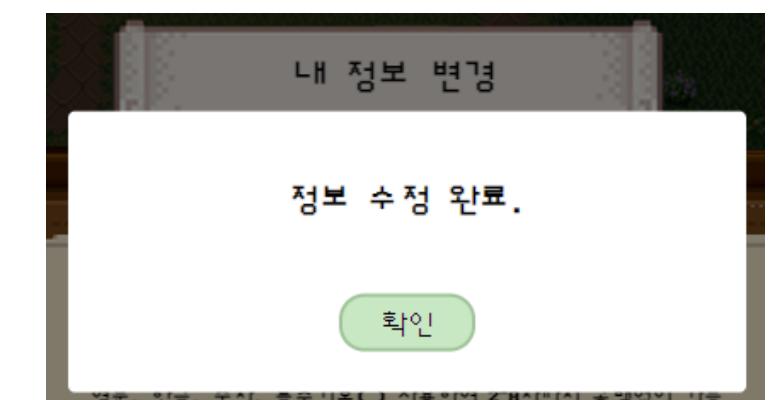
회원탈퇴

- [정보수정 버튼]

닉네임 / 비밀번호 유효성 조건 / 자기소개 란 글자 수 조건 불만족 시 비활성화
닉네임 중복확인 버튼 활성화 조건은 회원가입과 동일

활성화 된 [정보수정 버튼] 클릭 시 정보 수정을 알리는 alert / 변경된 데이터 DB에 UPDATE 처리

광장이나 마이룸 등에서 실시간으로 표시되는 캐릭터 하단에 닉네임 표시에 바로 업데이트 된 닉네임을 표시 할 수 있도록 sessionStorage 에도 닉네임 정보 업데이트



회원 탈퇴

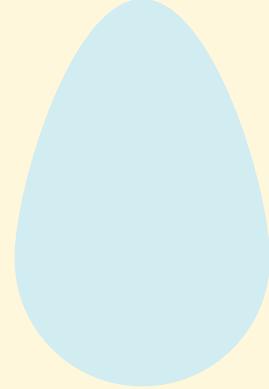


- 회원 탈퇴 처리

DB

- MEMBER TABLE UPDATE (STATUS = N)
- DISABLED_MEMBER TABLE INSERT
- BOARD TABLE 유저 정보 DELETE
- CHATTING TABLE 유저 정보 DELETE
- HEART TABLE 유저 정보 DELETE
- FRIEND TABLE 유저 정보 DELETE

Session 로그인 유저 정보 지우기



윤지영



광장 - 캐릭터



- 캐릭터 외관 표시

유저 계정 정보에 따라 해당 유저가 설정해둔 캐릭터 표시
캐릭터 하단 : 닉네임 표시

- 특이사항

비동기로 유저 session정보 변경시 새로고침이 되지 않아
닉네임이 바로 변경되지 않는 오류 발생

해결 : 비동기 통신 결과를 바로 fillText로 변경해주는 걸로 해결!

광장 - 이동



- 캐릭터 이동

키보드 방향키 입력에 따른 캐릭터 이동 가능

- 구현

canvas, keyboard 이벤트를 사용하여 위치 값을 업데이트

-> requestAnimationFrame()을 통해 위치 값 변경이 움직이는 것처럼 보이도록 반복 호출

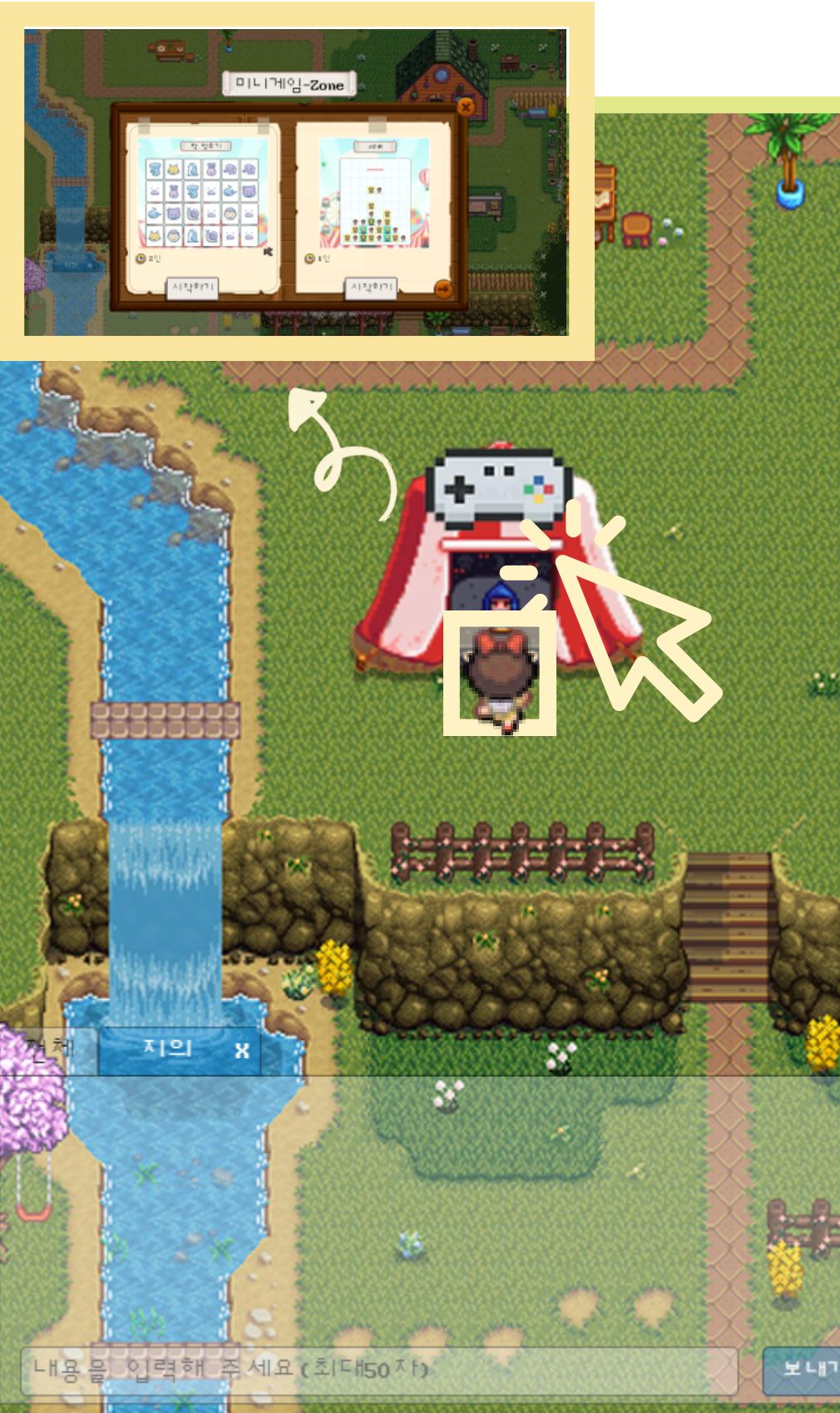
광장 - 이동제한

- 이동구역 제한

마이홈, 게시판, 캔버스
범위 밖 등 움직일 수
없는 지역 제한



광장 - 상호작용



- 상호작용

- 캐릭터가 맵 상의 상호작용 가능
- 사용자 캐릭터의 위치가 일정구역
도달 했을 시 이벤트 오픈
 - 캐릭터 이동 외에 마우스 클릭으로
도 상호작용 가능
 - 메뉴 이용중일 경우 방향키 입력에
따른 캐릭터이동을 막기

광장 - 캐릭터 클릭



- 타인캐릭터 클릭

타인캐릭터 클릭시 해당 유저 정보 오픈

- 특이사항

square.js 와 userInfo.js 서로의 함수와 변수를 import, export 하여 사용해 코드의 생산성을 높임.

마찬가지로 home.js에서도 userInfo.js를 import하여 사용.

home.js -> userInfo.js -> square.js까지 오게 되면서 충돌 오류 발생.

해결 : js를 타고 넘어와도 문제없게 바로 호출하는 이벤트들을 function으로 정리하여 실행용 squareinit.js를 새로 작성하여 실행시키는 구조로 변경.

광장 - 최초접속



중복접속



- 최초진입

로그인 후 최초 진입일 경우 스타트 지점에 캐릭터 표시
접속중인 유저가 중복 접속을 시도할 경우 로그아웃 되도록 처리

- 특이사항

자체 제작한 alert창을 사용하다보니, 페이지가 넘어가면서
띄워놓은 alert창을 확인하기도 전에 사라지는 오류 발생.

해결 1 : alert 확인버튼에 페이지 이동을 걸 경우, alert 확인버튼을
누르기 전까지는 로그인되어 있다고 판별되어 지속적인 중복접속오류
발생.

해결 2 : localStorage를 통해 중복접속으로 main으로 이동했을
경우 main에서 alert메세지를 띠워주는 것으로 해결!

광장 - 동시접속



- 동시접속

현재 접속중인 유저들의 캐릭터 움직임 실시간 표시 가능
각 사용자의 캐릭터 움직임을 실시간으로 볼 수 있도록 동작

- 구현

WebSocket API를 통해 구현

- 특이사항

keyboard 이벤트에서 socket.send로 움직임을 전달.
화면 랜더링 속도와 socket.send의 타이밍 차이로 움직임이 버벅이면서 표현.



너무 많은 메세지를 send하게 되어, 다중 접속이 많은 경우 소켓이 끊어지는 오류 발생. 이럴 경우 모든 socket 이벤트까지 종료.

해결 : socket.send를 화면 랜더링 속도와 맞추어 좌표 값 update쪽으로 이동

해결 : 소켓이 끊어지면 선언되어있던 socket 이벤트를 객체에 넣어 재호출 하는 방법으로 해결!

광장 - 이벤트 호출



게임 이벤트 중
캐릭터위치



- 이벤트 호출 후 위치

이벤트 호출후 광장에 재진입 했을때 캐릭터 표현



공지사항 이벤트
중 캐릭터위치

- 특이사항

광장에 훔으로 화면이동을 할 경우, 광장에서 캐릭터가 지워져하는데 캐릭터가 계속 남아있는 오류 발생.

→ 원인 : 페이지 이동이 일어나는 그 찰나에 생기는 키보드 이벤트 때 문에 소켓에 이미지가 전송되어 일어나는 결과

→ 해결 : 이벤트를 정지하는 변수를 통해 페이지 이동 이후의 모든 이벤트를 정지시켜는 것으로 해결!

광장 - 캐릭터표시



- 타인캐릭터 표시

종류에 따른 사용자 캐릭터 표시 위치 처리 필요

- 기존에 접속중인 타유저들의 캐릭터 표시 (최초접속)
- 현재 새로 접속한 유저 캐릭터를 기존 접속중이던 유저들 화면에 실시간으로 표시 (접속중인 유저)

- 특이사항

걷는 모션이 한 캐릭터뿐만 아니라 모든 캐릭터에 걸리는 오류

해결 : 필터링을 통해 소켓에서 받은 user이미지만 움직일 수 있게
로직 수정

광장 - 종료



- 종료

로그아웃등으로 사용자가 페이지 사용을 중지했을 경우
해당 캐릭터를 타 사용자들의 화면에서 제거 필요

- 특이사항

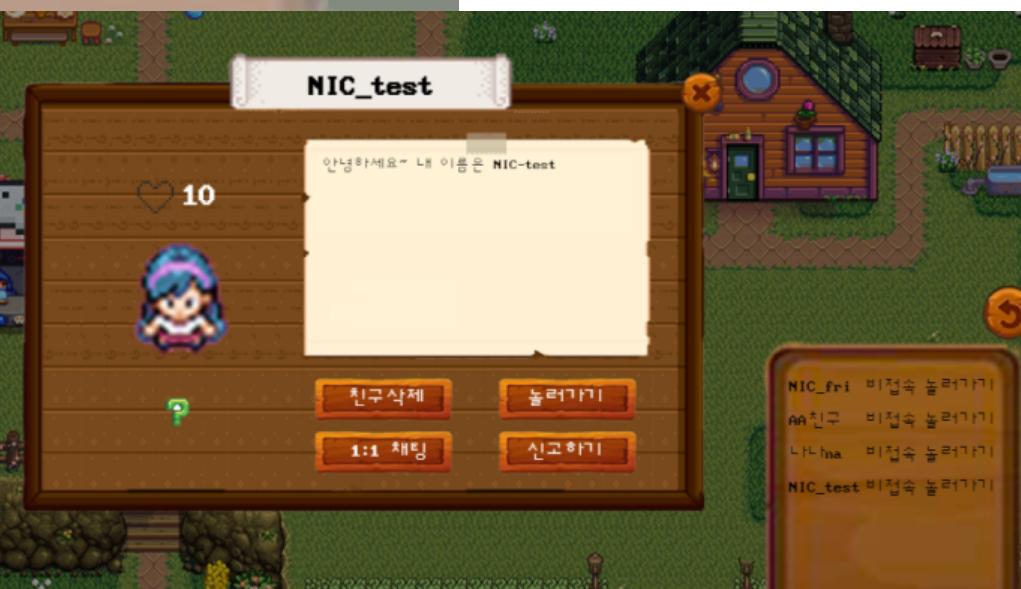
@OnClose를 사용해 나가는 유저를 체크하니, 기존에 있던 유저가
움직어서 다시 랜더링이 되지 않으면, 삭제되지 않는 현상 발견

해결 : JS 측에도 종료이벤트를 추가하여, 종료와 동시에 이미지를 삭제할 수 있도록 메세지 전송

설정 - 친구목록



친구들 : 접속/비접속 표시
접속중일 경우 맨위로



놀러가기 클릭
해당 유저정보 오픈

- 친구목록

모든 화면에서 상시 표시되는 친구 목록 버튼

- 현재 친구 추가되어있는 유저 목록+현재 접속상태 표시
- 놀러가기 클릭시 해당 친구 유저의 캐릭터 상세보기 팝업

- 특이사항

비동기로 서버 데이터를 받아와서 HTML생성 과 동시에 이벤트 부여 할 경우 해당 HTML을 찾을 수 없는 오류 발생

- 원인 : HTML이 만들어지기 전에 이벤트를 부여하여 없는 HTML 이벤트를 부여한 오류

해결 1: 상위 부모요소를 선택하여 자식에게 이벤트를 내려주는 방법
하지만 결국 userId 등 date값을 프린트단에 저장해야한다는 점이 보안상의 의도와 맞지않음.

해결 2 : createElement("td"); 를 반복문으로 생성하여 생성한 td 태그에 이벤트를 부여하는 방식으로 해결 : 코드 수는 좀 더 길어졌지만, userid를 직접적으로 노출하지 않을 수 있게됨

설정 - 설정버튼



광장의
설정버튼
↑

- 상시표시

모든 화면에서 상시 표시

- 클릭 시 내정보변경, 로그아웃 버튼 팝업

마이룸의
설정버튼



- 특이사항

광장에서 작동하는 버튼 JS가 마이홈에서는 실행되지 않는 오류 발생

- 원인: 버튼 JS에서 사용하는 import 해온 광장JS와 홈JSP간의 오류발생.
ex) 광장의 공지사항에 부과된 이벤트는 home에서는 공지사항이 없으니 발생할 수 없는 이벤트. 그렇다고 이부분만 export 해서 빼어내기에는 그에 딸리는 함수들이 유기적으로 연결되어있음.

- 해결 : 해당 이벤트를 if()광장에만 존재하는 것이 있다면으로 이벤트 부여를 선택적 부여

설정 - 로그아웃



- 로그아웃

로그아웃 가능한 버튼

- 로그아웃 버튼 클릭시 로그아웃하시겠습니까? 확인 알림창 팝업
- 세션에서 로그인 정보 삭제, 소켓상 해당 유저가 접속 해제 되었다는 정보를 전송해주기

- 특이사항

이외에 최초로직에는

추가되지 않았던, 탈퇴유저, 정지유저, 창 닫기 체크 추가함

미니게임 - 게임모달



- 구현기능

광장에서 미니게임 이벤트 발생시
미니게임 모달 오픈

게임시작 버튼 클릭시
해당 게임페이지로 넘어감

미니게임 - 짹 맞추기

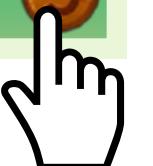


게임 입장과 동시에
카드 무작위 셋팅

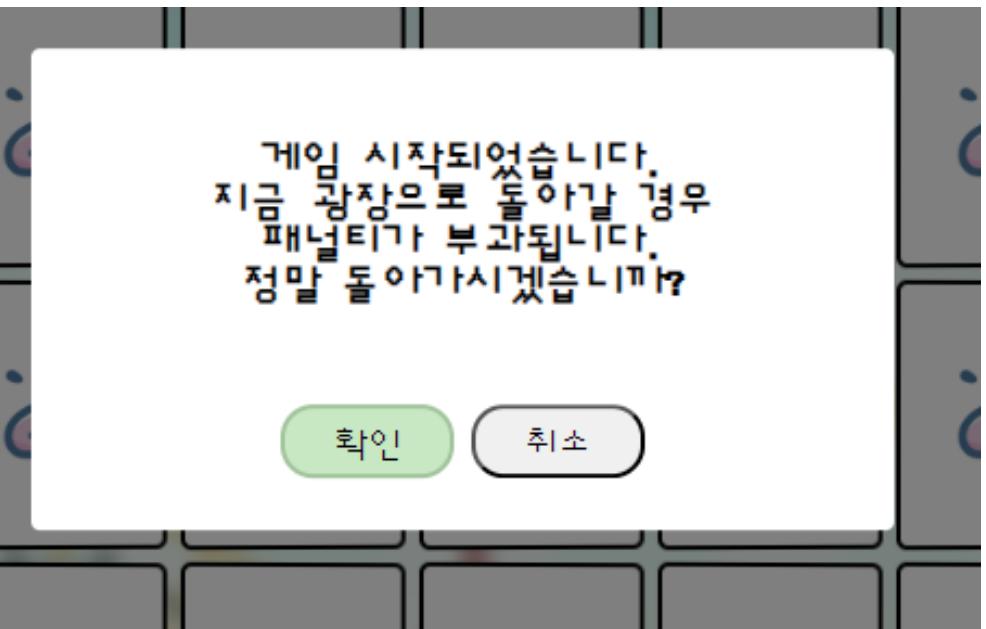
상대방이 들어오기 전까지는
모든 이벤트 발생하지 않음



미니게임 - 2인접속

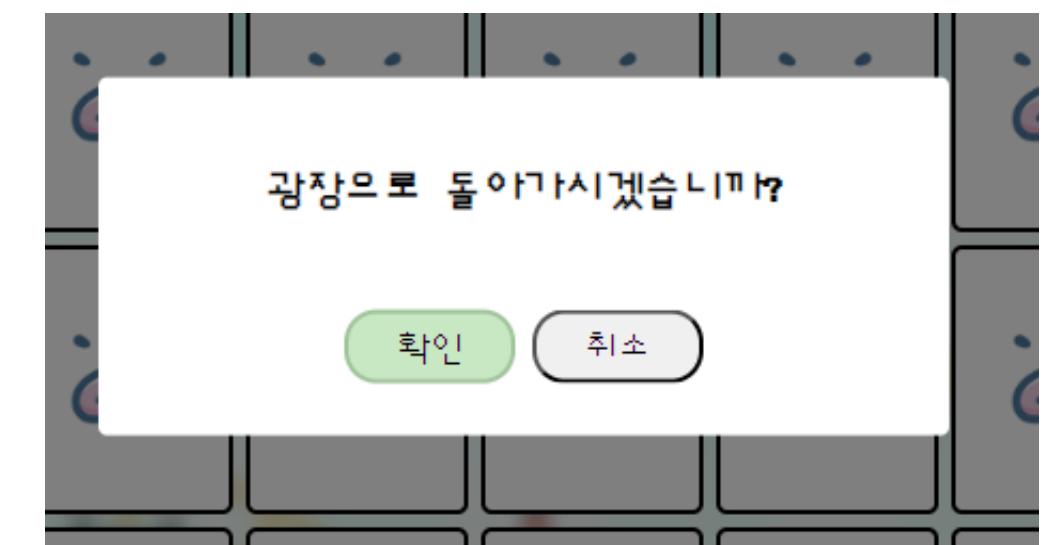


상대방이 있을 때



게임패널티 : db 골드 차감

상대방이 없을 때



미니게임 - 2인접속



- 2인 게임

2인 이상 들어올 수 없도록 제한
소켓을 통한 2인 게임 구현

- 2P 랜더링

상대방 유저가 들어오면, 해당 유저정보 표시
들어온 유저도 존재하는 유저 정보 표시

미니게임 - 게임플레이



모두 입장 후 최초 1회 패턴 보여주기

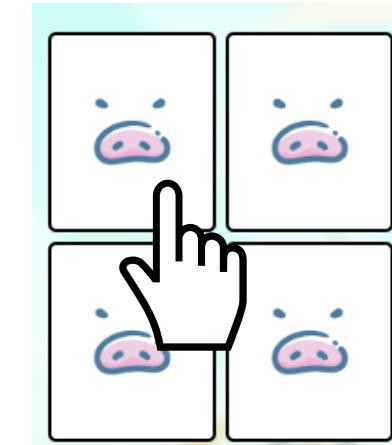


보여준 후 다시 감추기



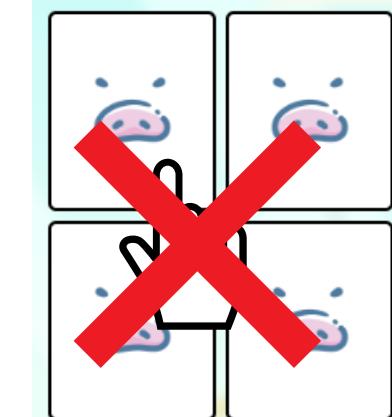
미니게임 - 게임플레이

1p(나의 턴)



클릭으로 카드 뒤집기 가능

2p(상대방의 턴)



클릭 사용 불가능

실시간으로 화면에 표시

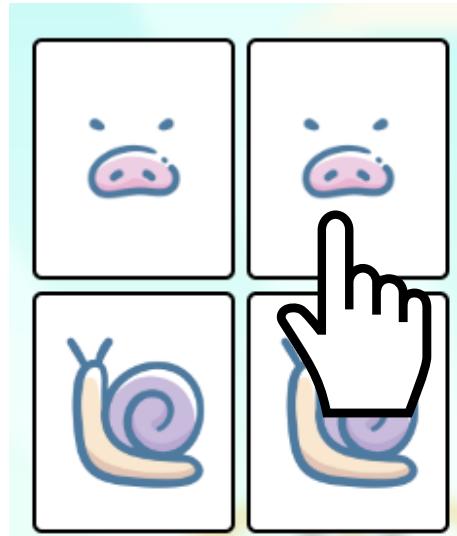
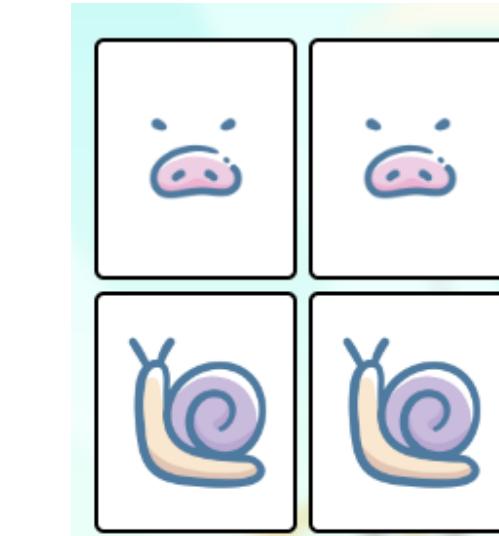


미니게임 - 게임플레이

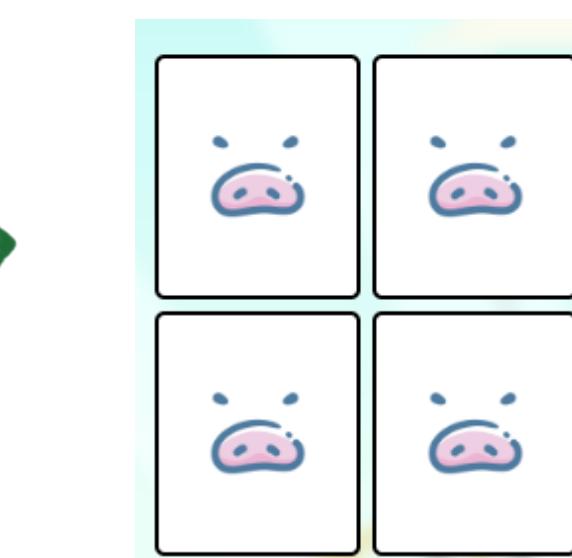
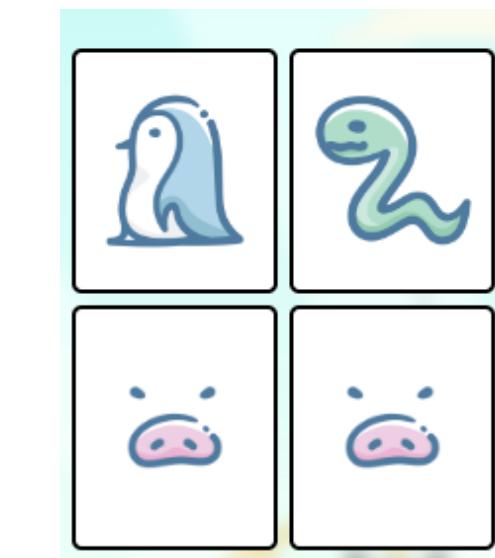
카드를 뒤집었을 때



첫번째 카드와
두번째 카드가 같음



첫번째 카드와
두번째 카드가 다름



다시 돌아가고
상대방에게 턴 넘김

계속 선택할
수 있음

미니게임 - 게임플레이

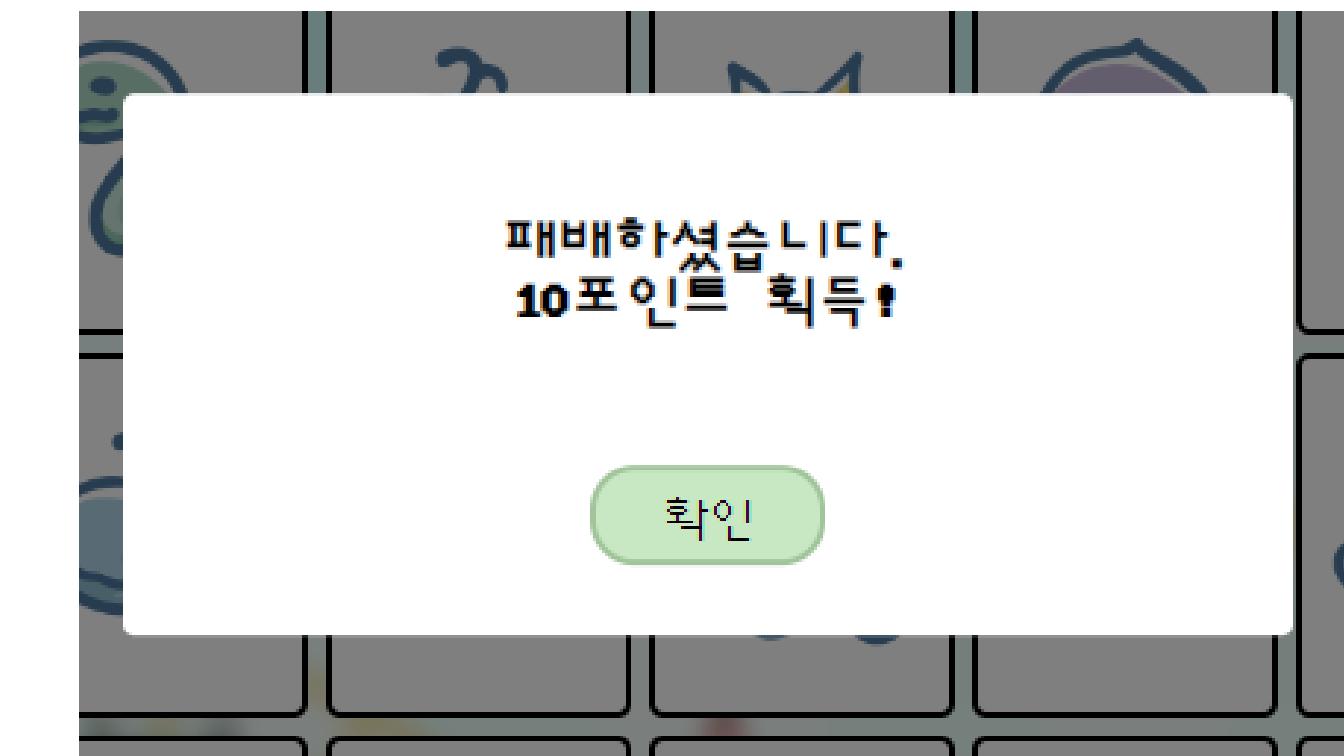
결과안내

카드가 전부 맞춰지면 게임 종료



승리하셨습니다.
50포인트 획득!

확인



패배하셨습니다.
10포인트 획득!

확인

박가영



User information(유저 정보창)



아무 정보도 변경하지 않은 기본 정보창



loginUser라는 이름으로 세션에 저장된
로그인한 본인의 id를 가져와 띠운
내 정보창

```
/* 내 정보 가져오기 */
export function getMyInfo() {
    countHeart(getSessionStorage('loginUser'));
    //console.log('나 클릭함', getSessionStorage('loginUser'));
    $.ajax({
        url: getContextPath() + "/userInfo",
        data: { userId: getSessionStorage('loginUser') },
```

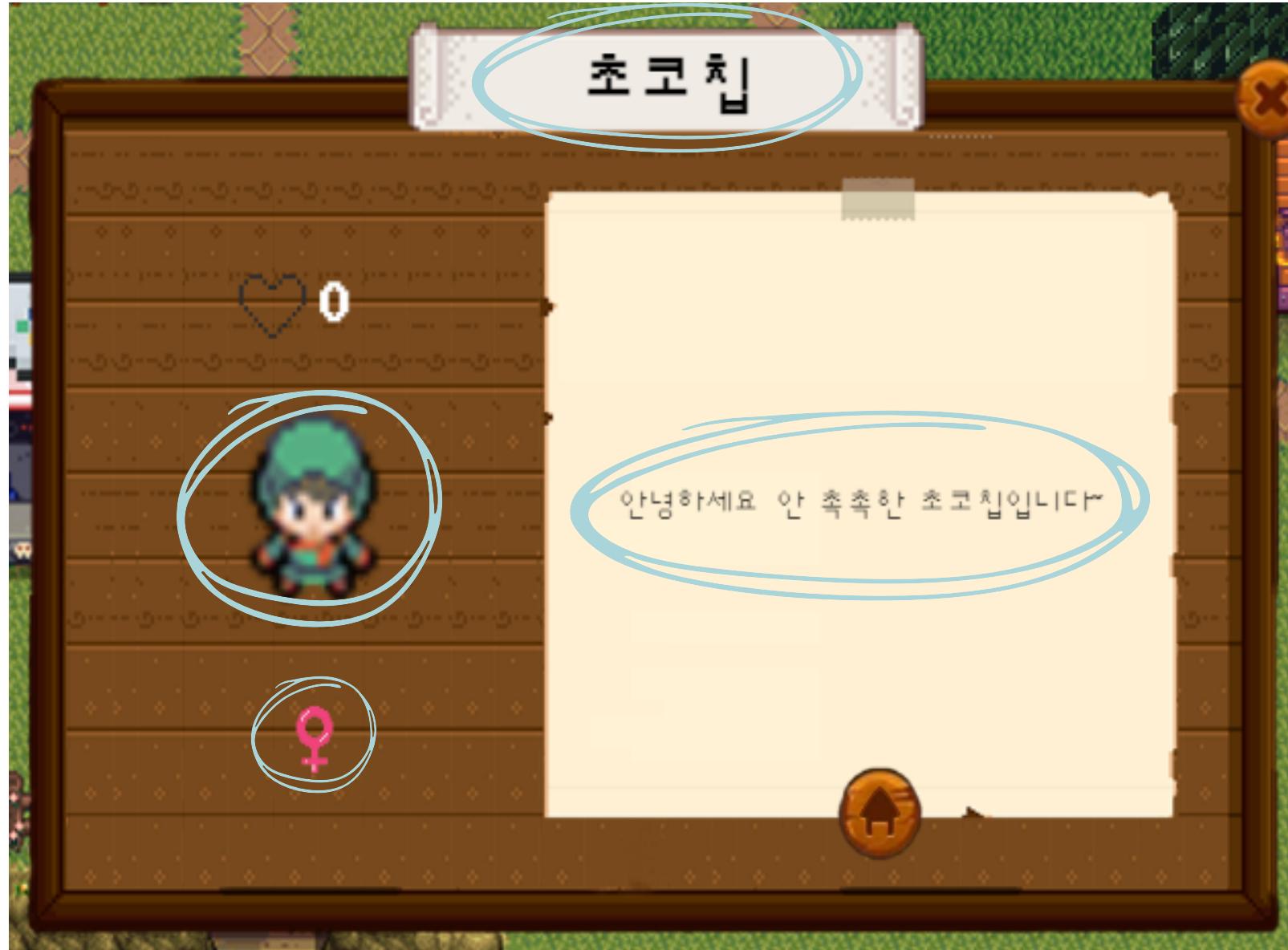
- 마이홈 가기

clickedUserId라는 이름으로 세션에 저장된
클릭한 유저의 id를 가져와 띠운
다른 유저의 정보창

```
/* 다른 유저 정보 가져오기 */
export function getUserInfo() {
    selectHeart();
    selectFriend();
    countHeart(sessionStorage.clickedUserId);
    //console.log('클릭한 유저', sessionStorage.clickedUserId);
    infoModalOpenOverlay();
    $.ajax({
        url: getContextPath() + "/userInfo",
        data: { userId: sessionStorage.clickedUserId },
```

- 친구추가/삭제
- 유저의 마이룸 놀러가기
- 1:1 채팅
- 신고하기
- 호감도 주기

User information(유저 정보창)



유저가 변경한 정보(닉네임, 캐릭터 스킨, 성별, 자기소개)를
db에서 조회하여 정보 변경 후 정보창을 열었을 때 새로고침을
하지 않아도 변경된 정보로 정보창 내용이 꾸며짐

User information(유저 정보창)



호감도

- 월요일부터 현재까지 해당 유저가 받은 누적 호감도 개수 표시
- 월요일마다 호감도 개수 리셋
- 동일 유저에게 일주일에 한 번만 호감도를 줄 수 있음



일주일 기간 동안 호감도를 줬던 대상인지 아닌지를 DB의 HEART 테이블을 조회

-> 해당 유저에게 호감도를 준 이력이 없거나 호감도 준 걸 취소하면 빈하트, 호감도를 처음 주거나 준 이력이 있으면 빨간 하트로 이미지 변경
(클릭 이벤트 가능)

-> 내 정보창에서는 받은 호감도 개수가 0일 때는 빈하트, 받은 호감도가 1개 이상일 때는 빨간 하트로 이미지 변경 (클릭 이벤트 불가능)

User information(유저 정보창)



- 친구 추가 버튼 클릭 시
-> 친구 추가 alert 활성화
-> 추가 클릭
-> 친구 목록에 친구 추가
- 친구 삭제 버튼 클릭 시
-> 친구 삭제 alert 활성화
-> 삭제 클릭
-> 친구 목록에서 친구 삭제



- 내 정보창에서 집 아이콘 클릭 시
-> 나의 마이홈으로 이동
- 다른 유저 정보창에서 놀러가기 클릭 시
-> 해당 유저의 마이홈으로 이동

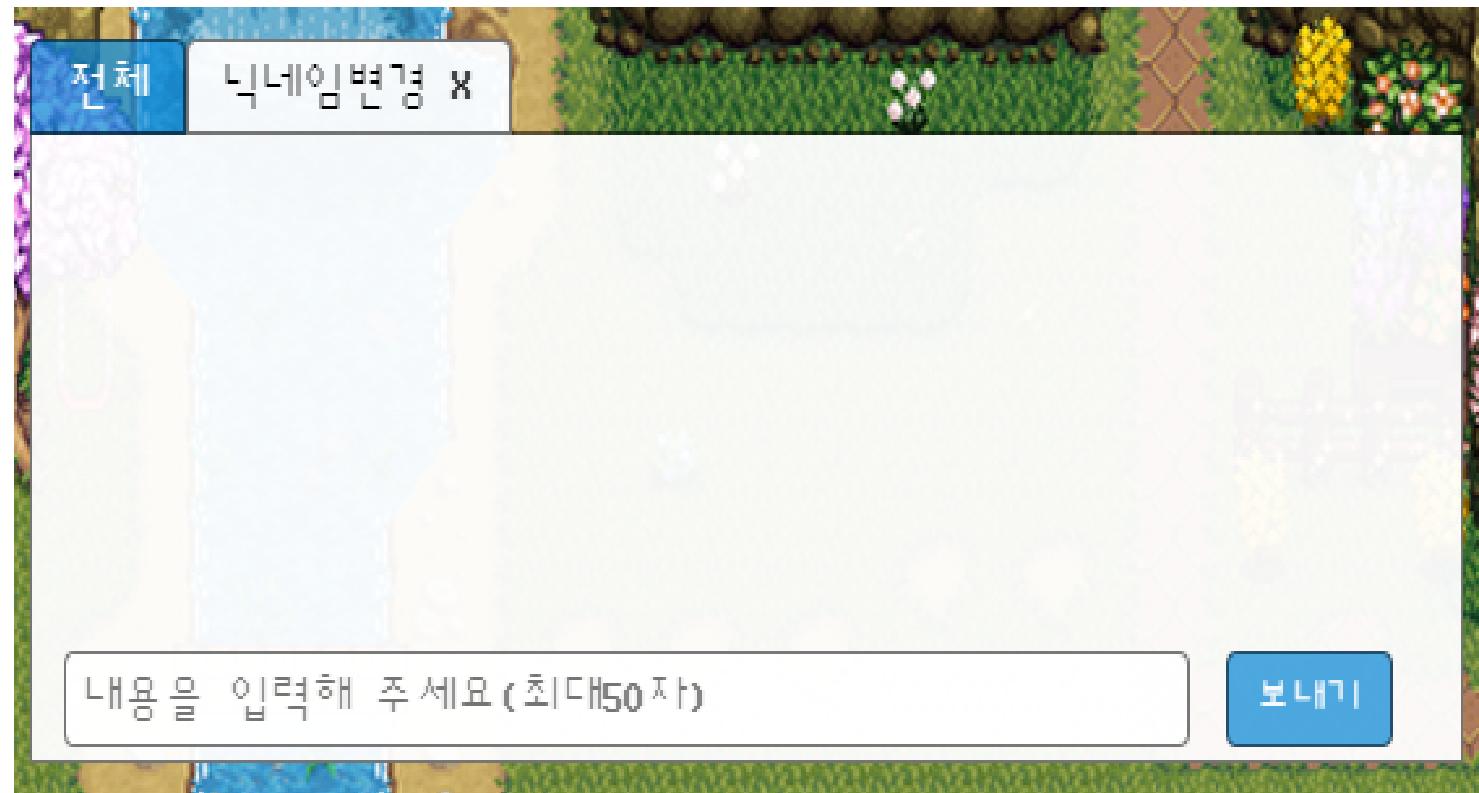
친구 목록에 친구가 추가됨



친구 목록에 친구가 삭제됨



User information(유저 정보창)



- 해당 유저와 1:1 채팅창이 열려있지 않다면 본인과 해당 캐릭터 유저간의 1:1 채팅창 추가되며 해당 채팅창으로 포커스 이동
- 기존에 채팅창이 열려있었다면 해당 채팅창으로 포커스 이동

- 신고 제목과 내용 미작성 시 신고 불가
- 신고 완료 시
-> 신고자의 ID, 신고 대상자의 ID, 신고 제목, 신고 내용이 DB에 저장
- 신고 취소 시
-> 등록한 신고를 취소하는 것이 아닌 현재 신고창 자체가 닫힘
- 신고 제목 20글자 제한
- 신고 내용 300글자 제한

☞ 사용자가 몇글자를 입력했는지 알 수 있게 실시간으로 글자 수를 세줌

☞ 모달창은 입력하던 상태로 창을 닫게 되면 다시 열었을 때 입력하던 흔적이 그대로 남아있어 입력 중 창을 닫으면 이전 입력 흔적을 초기화 처리

Notice Board(공지사항 게시판)



- 일정기간(7일) 동안의 누적 호감도 랭킹 상위 3명을 조회해 공지사항 상단에 표시
 - 매 주 월요일마다 랭킹 변경

상위 랭킹 3명에게는 보상용 특별 캐릭터 지급

- 랭킹에 있는 캐릭터 클릭 시 해당 유저의 정보창이 열림



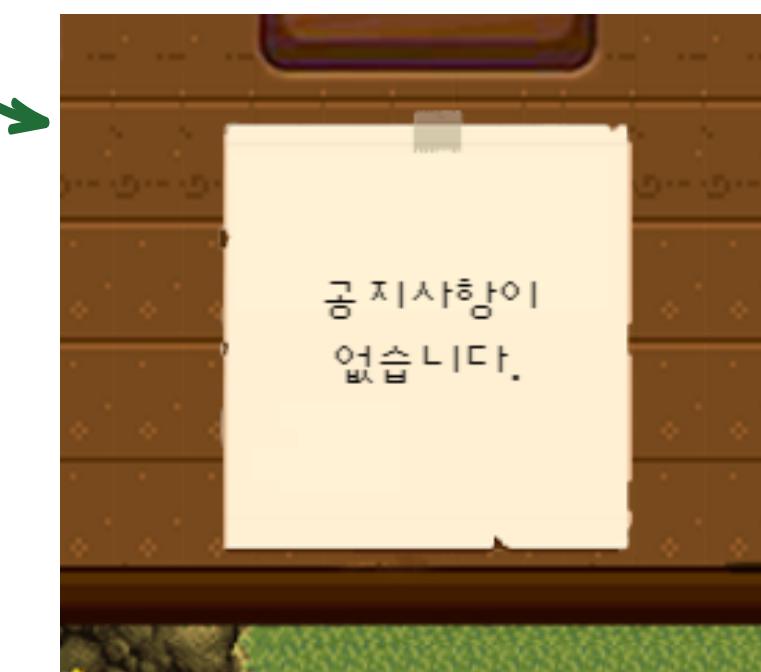
Notice Board(공지사항 게시판)



- 공지사항 리스트(공지사항 제목만) 최대 4개 표시
- 포스트잇 클릭 시 공지사항 상세화면 모달창 뜸
(클릭 이벤트 가능)

• 공지사항 제목의 길이가 정해진 길이보다
길어지면 말줄임표 처리
• 세번째 리스트까진 공지사항 제목 표시
네번째 리스트만 '더보기'로 텍스트로 변경

- 등록된 공지사항이 없을 경우
공지사항이 없다는 포스트잇 표시 (클릭 이벤트 불가능)



Notice Board(공지사항 게시판)

- 관리자가 작성한 공지사항의 제목만 표시 (최근 공지사항부터 나열)

제목이 지정한 길이보다 길어지면 말줄임표 처리

```
let listCount;
let noticeLimit = 6;
let pageLimit = 4;
let globalCurrentPage = 1;
let noticeList = [];
let maxPage;
let startPages;
let endPages;
```

- 한 페이지당 공지사항 리스트 여섯개 표시
- 페이지바 개수 네개 표시

1	2	3	4	>
---	---	---	---	---



- 공지사항 제목 클릭 시 해당 공지사항 상세내용 표시
- 공지사항 창을 닫았다면 항상 최근 공지사항이 보이게 구현

공지사항 내용이 길 경우 스크롤 처리

노지의



My Room

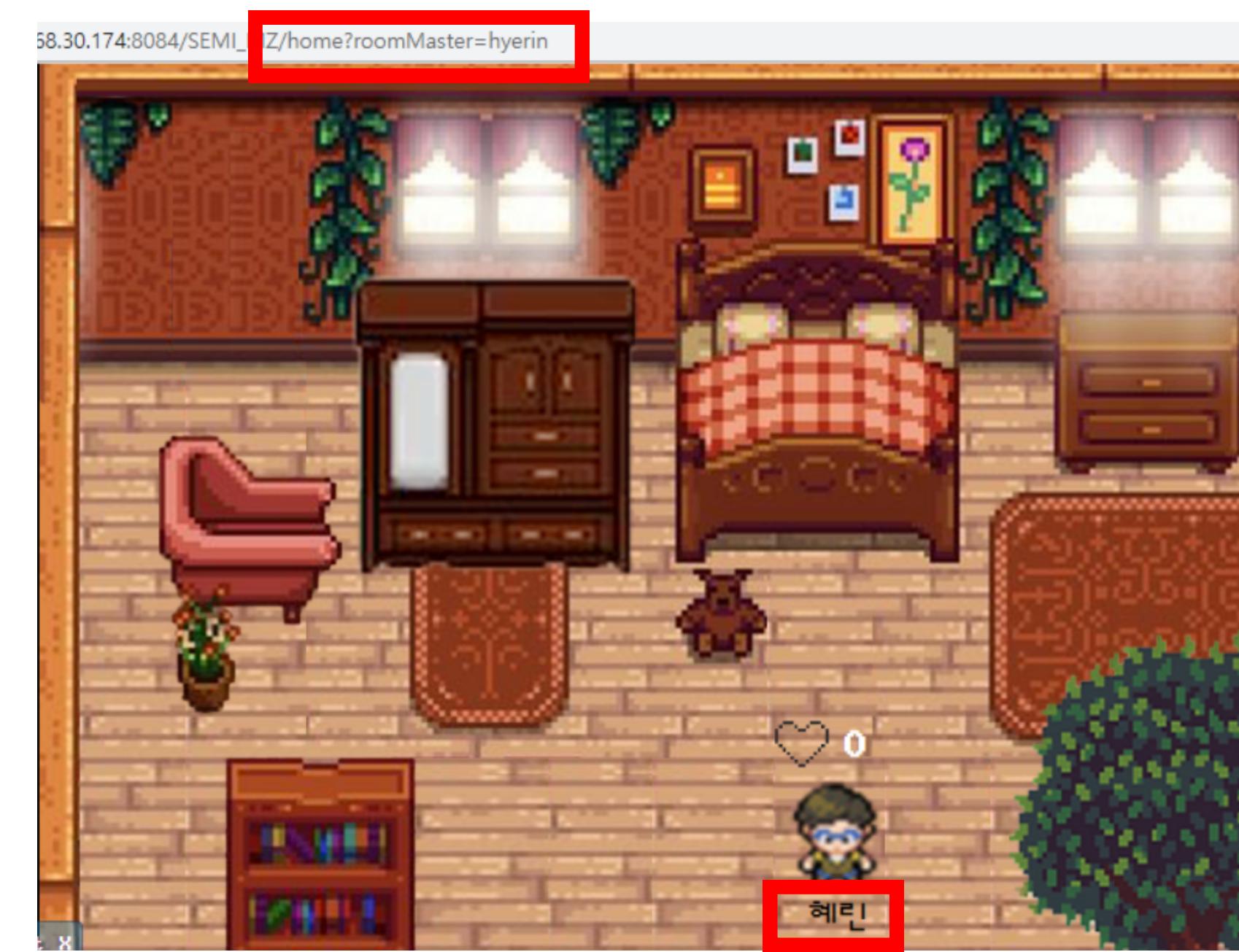
마이룸 진입시 로그인유저 , 접속 친구 유저 정보 받아오기

```
const gohome = () => {  
    location.href = getContextPath() + '/home?roomMaster=' + sessionStorage.clickedUserId;  
    location.href = path + '/home';  
};
```

[로그인 유저의 마이룸]



[접속유저(친구)룸 방문]



My Room

공통

- 스킨
 - 마이홈 중앙에 해당 마이홈 주인 캐릭터 표시
 - 해당 캐릭터의 누적 호감도 표시
 - DB의 HEART 테이블의 하트를 받은 유저의 호감도 총개수 조회해옴

```
.myroom-board-title{  
    width: 120px;  
    text-overflow: ellipsis; /* 말줄임표(...) 표시 */  
    overflow: hidden; /* 글자 자름 */  
    white-space: nowrap; /* 줄바꿈 방지 */  
}
```

<두곳 다 '지의'의 마이룸>



[로그인 유저 본인의 마이룸]

- 호감도
 - 누적호감도 조회만 가능(클릭이벤트 X)
→ 유저가 받은 호감도가 없을 시
빈하트로 표시
 - 받은 호감도가 1개 이상일 경우
채워진 하트로 표시



[타유저가 방문한 마이룸]

- 중앙 스킨
 - 방문한 집주인 유저 닉네임 표시
 - 중앙의 집주인 캐릭터 클릭 시
캐릭터 상세정보창 보여짐
(친구추가 / 1:1채팅 / 친구신고)
- 방명록
 - 타유저가 쓴 비밀글은 제목노출 X
→ "비밀글"로 표시
- 호감도
 - 하트 클릭 → 호감도 주기 / 빼기 가능

방명록(리스트 조회)

하나의 함수에서 receiveID 값에 따라

보여지는 화면구성이 다르게 구현

(receiveID : 방명록을 받은 유저의 아이디)

- 마이룸 방문시 → receiveID == 현재 로그인한 아이디
- 친구방 방문시 → receiveID == 방주인 아이디

```
function selectboardList(receiveID) {
    $(".board-list").show();
    $.ajax({
        url: path + "/selectBoardList",
        dataType: "json",
        data: { receive: receiveID },
        success: function(list) {
            },
    });
}
```

```
if (roomMasterId == "") {
    selectboardList(loginUserId);
} else {
    selectboardList(roomMasterId);
    // 글쓰기버튼 표시
    $("#writing-btn").css("display", "block");
}
```

<두곳 다 '지의'의 마이룸>

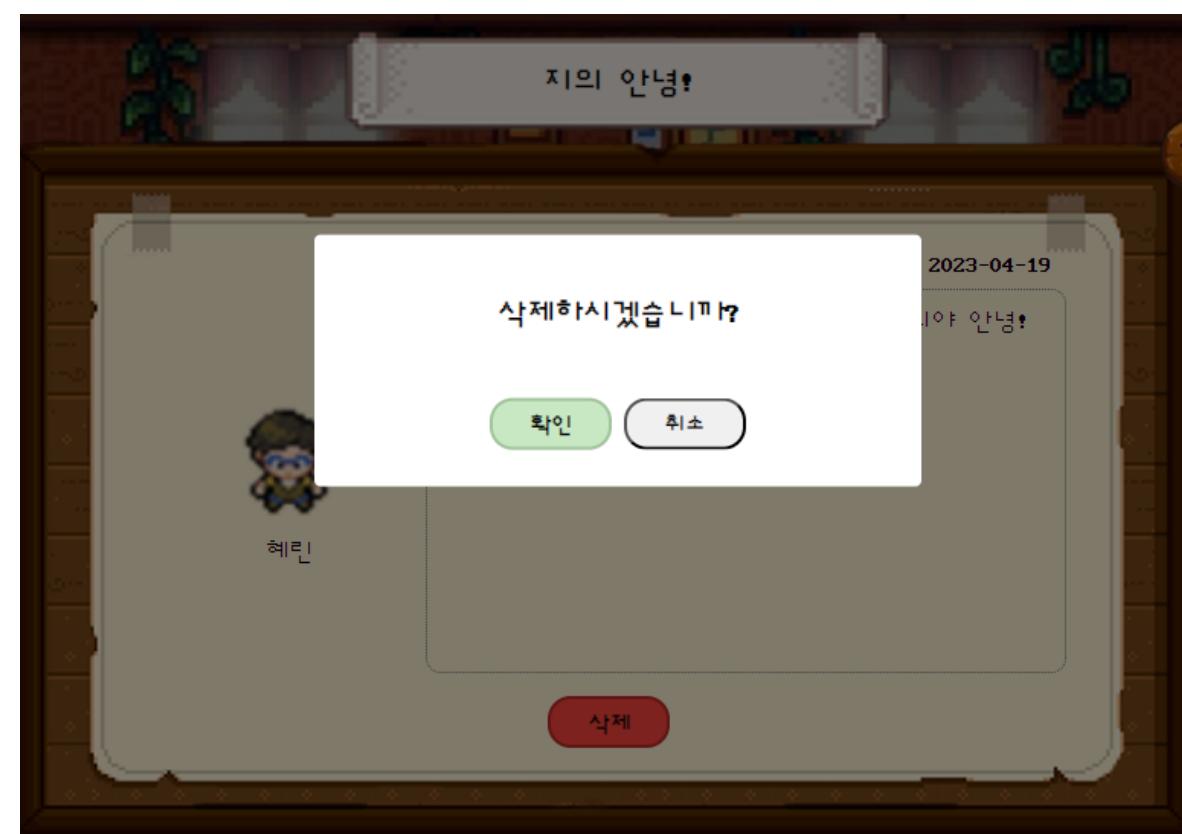


방명록(상세 조회) - [로그인유저:지의]

<'지의'본인의 마이룸>



- 작성자 캐릭터 스킨 + 닉네임
- 방명록 제목 / 내용 / 작성일
- 삭제 버튼



작성자 캐릭터 스킨 클릭 시
해당 캐릭터 상세정보창 팝업

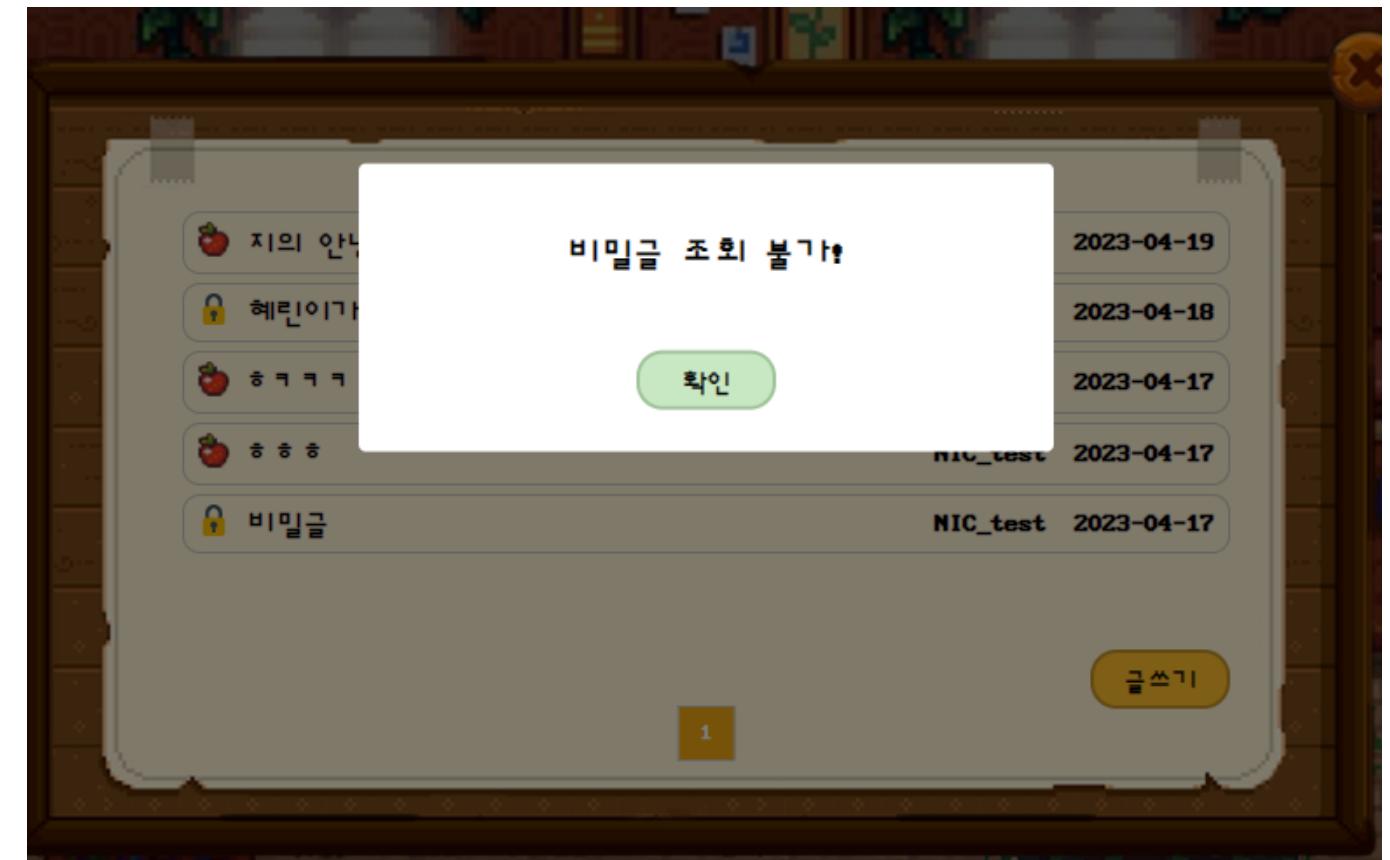
- 친구추가 / 삭제
- 친구마이룸 놀러가기
- 1:1 채팅
- 신고하기
- 호감도 주기

상세조회시 각각의 board-no 값
display : none 으로 처리 후
삭제기능 구현시 board-no 값만 넘겨줘
해당 게시글 삭제 처리

```
<div class="board-no" style="display: none;">121</div>
```

방명록(리스트 조회) - [로그인 유저:혜린]

<'지의'본인의 마이룸>



[본인 작성한 비밀글]

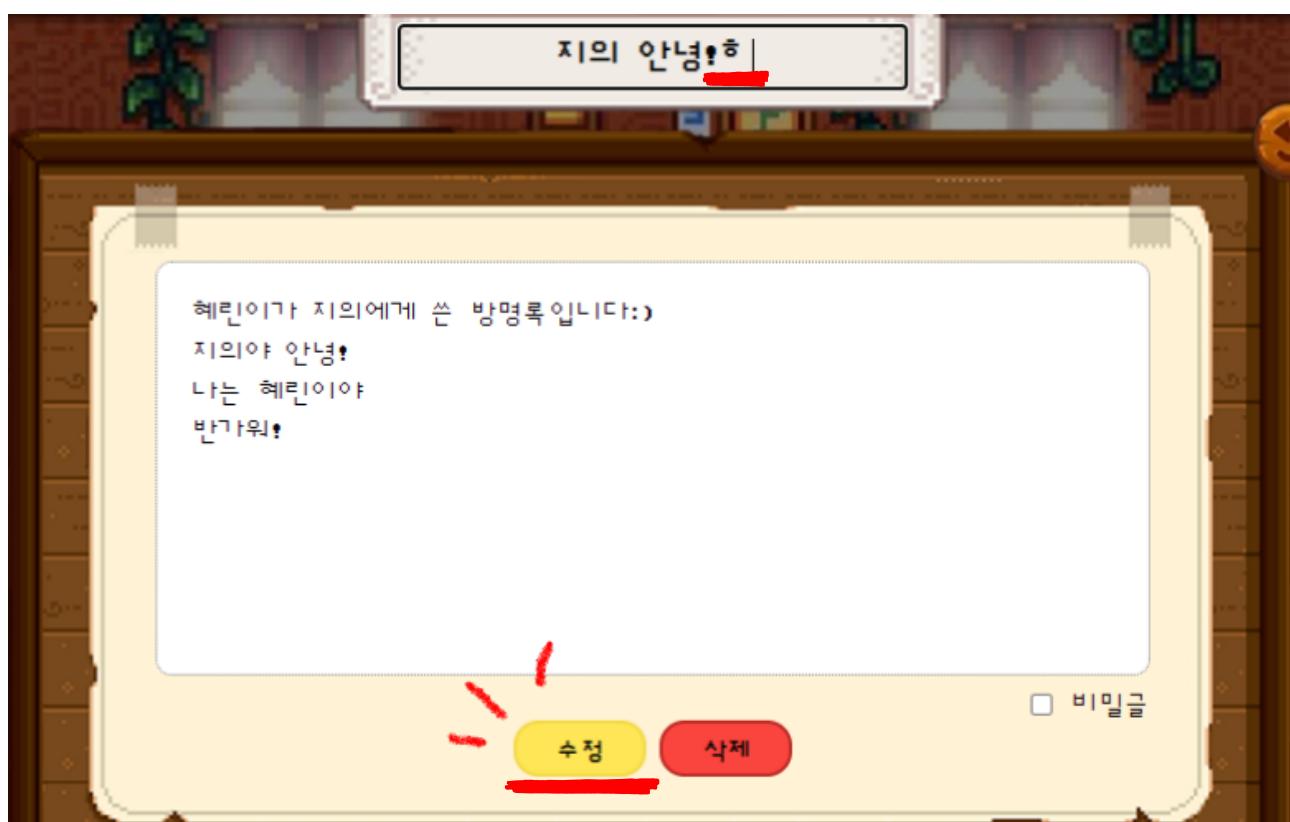
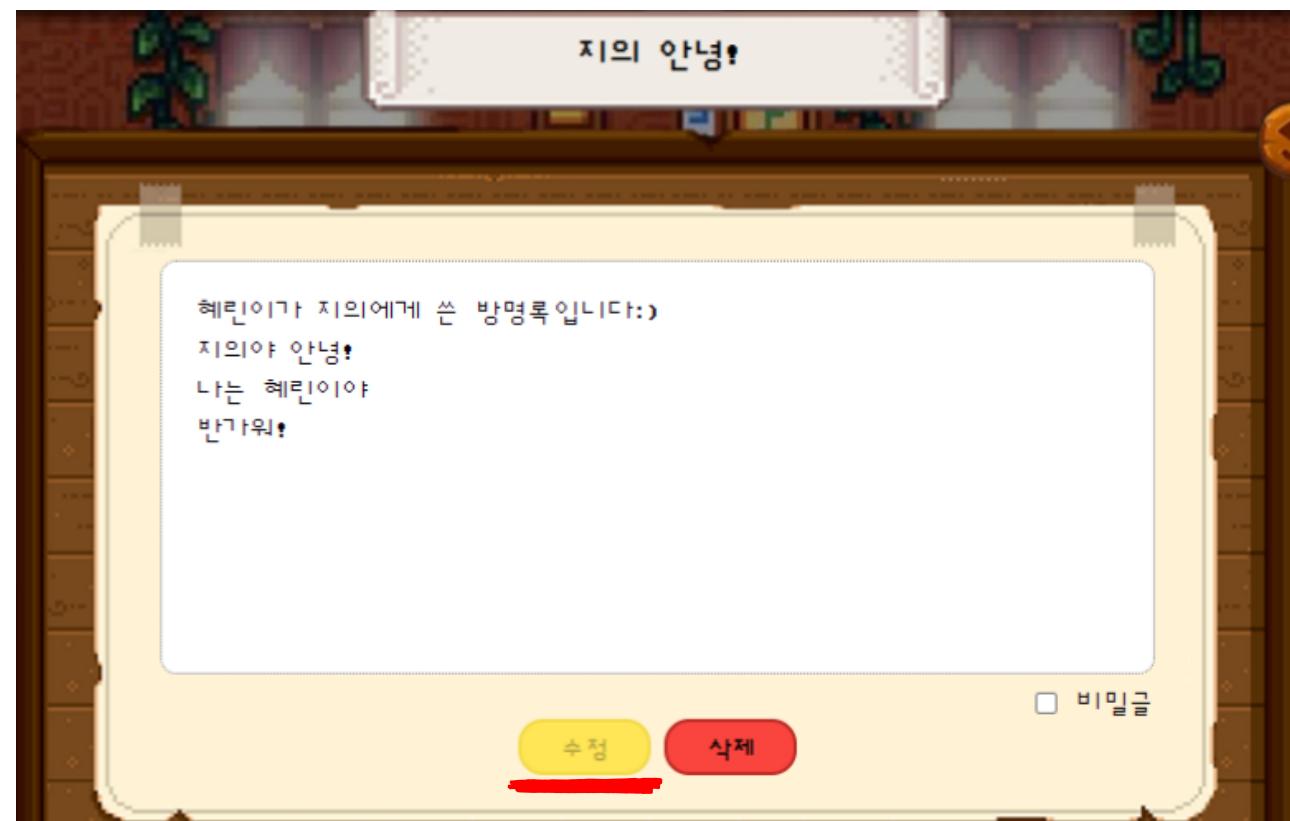
- 자물쇠 아이콘 표시

[타유저가 작성한 비밀글]

- 제목표시 x → "비밀글"
- 해당 게시글 조회불가 → 클릭시 alert 팝업

방명록(상세 조회) - [로그인 유저:혜린]

<'지의'의 마이룸>



[본인('혜린')이 작성한 방명록]

첫 진입화면 수정 버튼 비활성화



제목 / 내용 / 비밀글 중
하나라도 내용 변경 시
수정 버튼 활성화

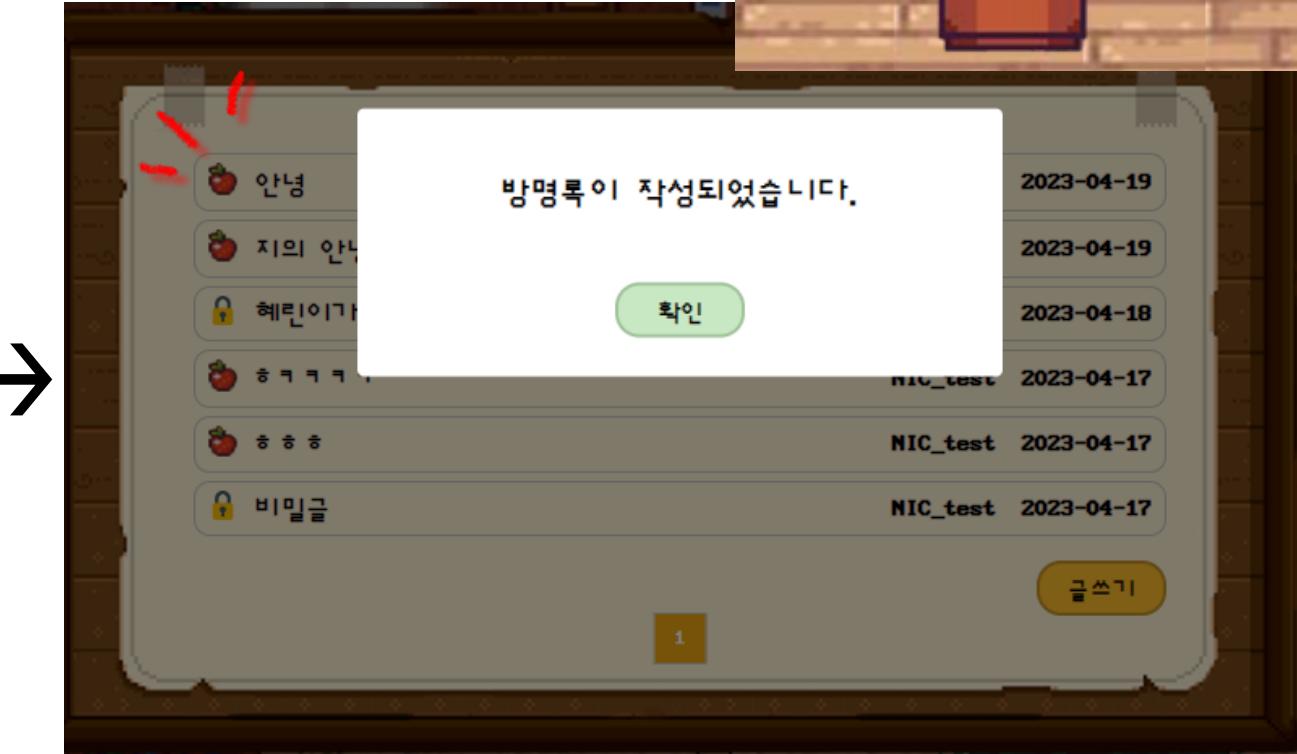
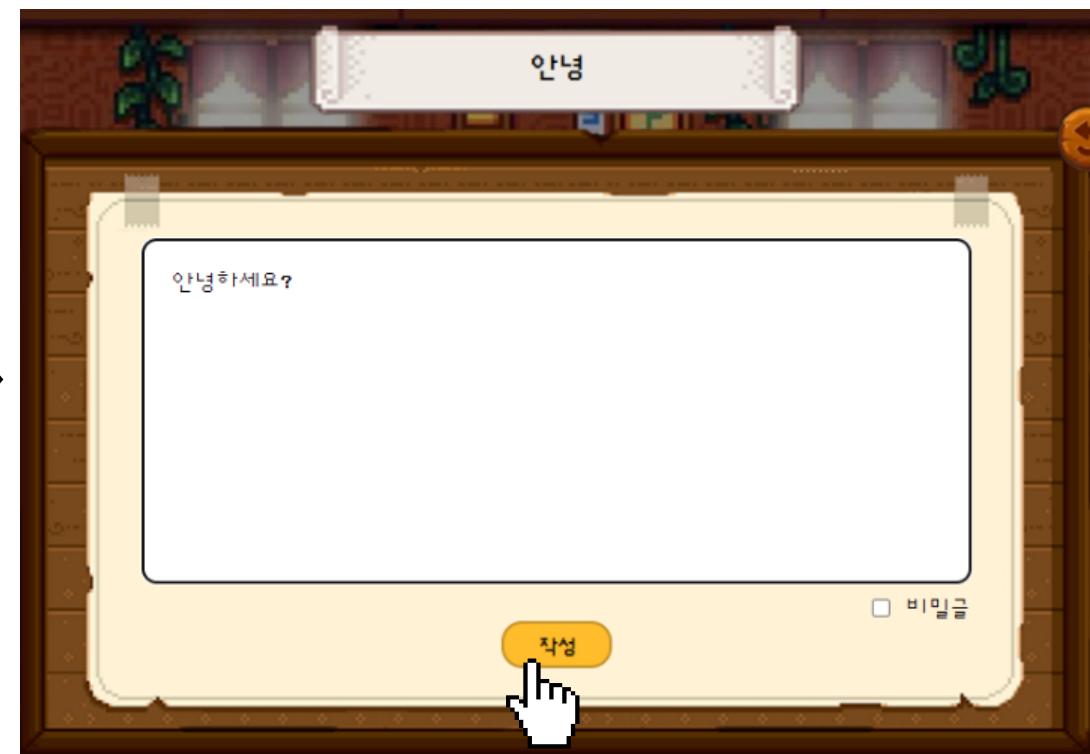
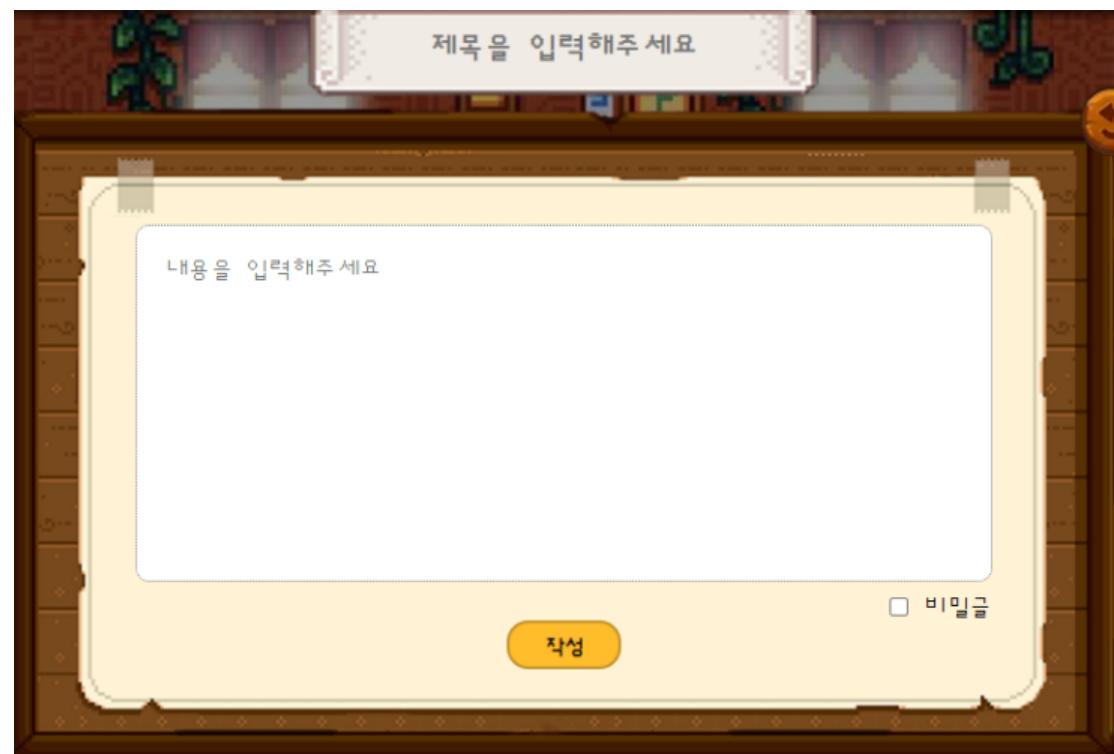
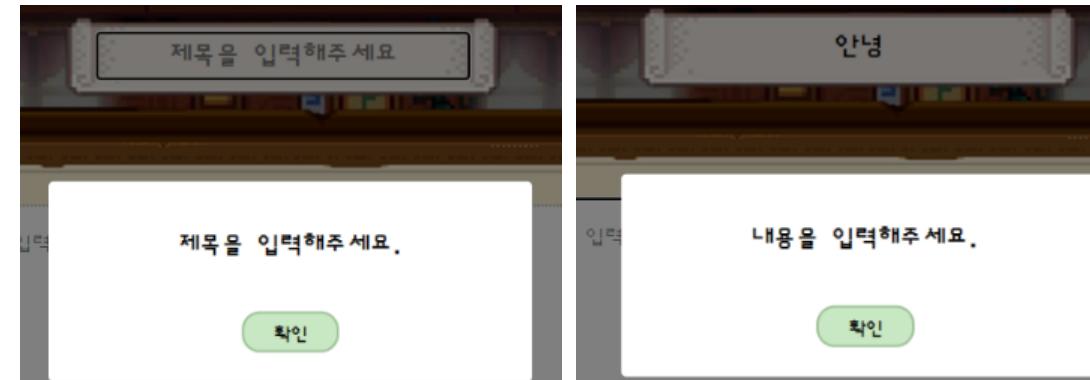
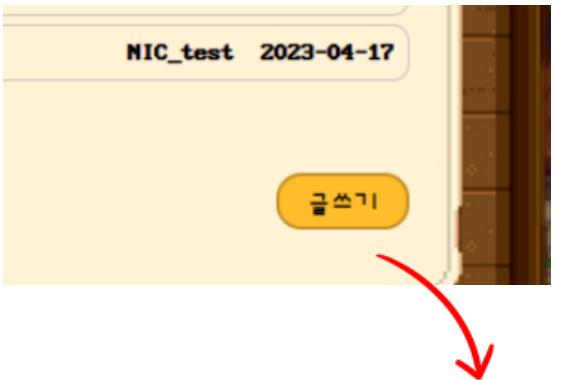
[타유저가 작성한 방명록]

- 읽기만 가능
- 수정 / 삭제 버튼 비활성화



방명록(작성) - [로그인 유저:혜린]

<'지의'의 마이룸>



글쓰기 버튼 클릭 → 방명록 작성 모달 열림

- 글자수 제한 (maxlength)
 - 제목 : 15글자
 - 내용 : 500글자
- 제목, 내용 필수값으로 각각의 입력칸 비어 있을 시 작성 불가하도록 alert창 생성

작성 완료시

1. 작성완료 alert창
2. 방명록 리스트 조회 모달 보여짐
 - a. 작성한 방명록 최상단으로 업로드
3. 마이룸 첫페이지 나무 아이콘
 - a. 작성한 방명록 최상단으로 업로드

옷장 / 상점

내용



- 1 본인의 마이홈일 경우에만 이용 가능
- 2 화면 진입시 옷장 모달 표시
- 3 현재 착용중인 캐릭터 스킨 좌측 미리보기에 표시
+ 착용 버튼 비활성화
- 4 우측 상단에 현재 보유중인 골드(코인) 량 표시
- 5 우측에 캐릭터 스킨 아이템 리스트 표시
- 6 한 화면에 스킨 개수 12개 이상 넘어갈 시
새로고침 없이 다음 페이지 바로 생성되도록 페이징 설정
- 7 우측 상단 옷장 나가기 버튼 표시

옷장 - 보유중인 캐릭터를 확인 및 캐릭터 스킨변경 가능한 메뉴

내용



- 스키ń박스 스키ń 클릭

→ 왼쪽 대표 스키ń에 이미지 적용 + 착용버튼 활성화



1. 착용버튼 활성화

→ 로그인 유저의 스키ń과 변경하고자 하는 스키ń값 조건문으로 비교

2. 미리보기 스키ń 변경

→ 변경하고자하는 스키ń의 src 값을 미리보기 스키ń의 src에 넣어주기

1 우측에 현재 보유중인 캐릭터 스키ń 리스트 표시

2 우측 캐릭터 스키ń 아이템 클릭시
좌측 캐릭터 미리보기 이미지를
그때그때 클릭한 캐릭터 스키ń 이미지로 변경

3 좌측 캐릭터 스키ń 미리보기
- 현재 착용중인 스키ń일 경우 착용 버튼 비활성화
- 현재 착용중인 스키ń이 아닐 경우 착용 버튼 활성화

4 착용 버튼 클릭시 하단 버튼 비활성화로 상태 변경
+ DB MEMBER-테이블 → SKIN_ID 업데이트

5 DB CHARACTER_SKIN 테이블
SKIN_ID == SAVE_ROOT 파일명 끝자리
동일하게 설정

DB CHARACTER_SKIN 테이블

SKIN_ID	SAVE_ROOT
0	/resource/img/user/skin0
1	/resource/img/user/skin1
2	/resource/img/user/skin2
3	/resource/img/user/skin3
4	/resource/img/user/skin4
5	/resource/img/user/skin5
6	/resource/img/user/skin6
7	/resource/img/user/skin7
8	/resource/img/user/skin8
9	/resource/img/user/skin9
10	/resource/img/user/skin10
11	/resource/img/user/skin11

상점 - 캐릭터를 구입 가능한 메뉴

내용



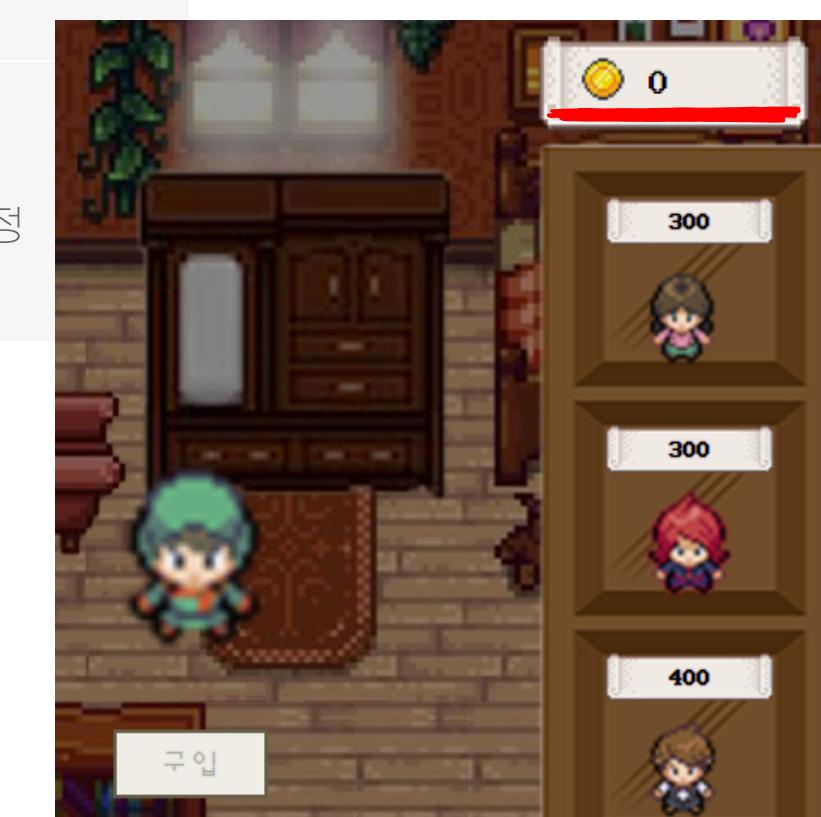
1 우측에 현재 보유중인 캐릭터 스킨 제외하고 표시

2 각 스킨별 구입 가격 표시

3 좌측 캐릭터 스킨 미리보기 하단에 구입 버튼 표시

4 보유 코인 없을 시 구입버튼 비활성화

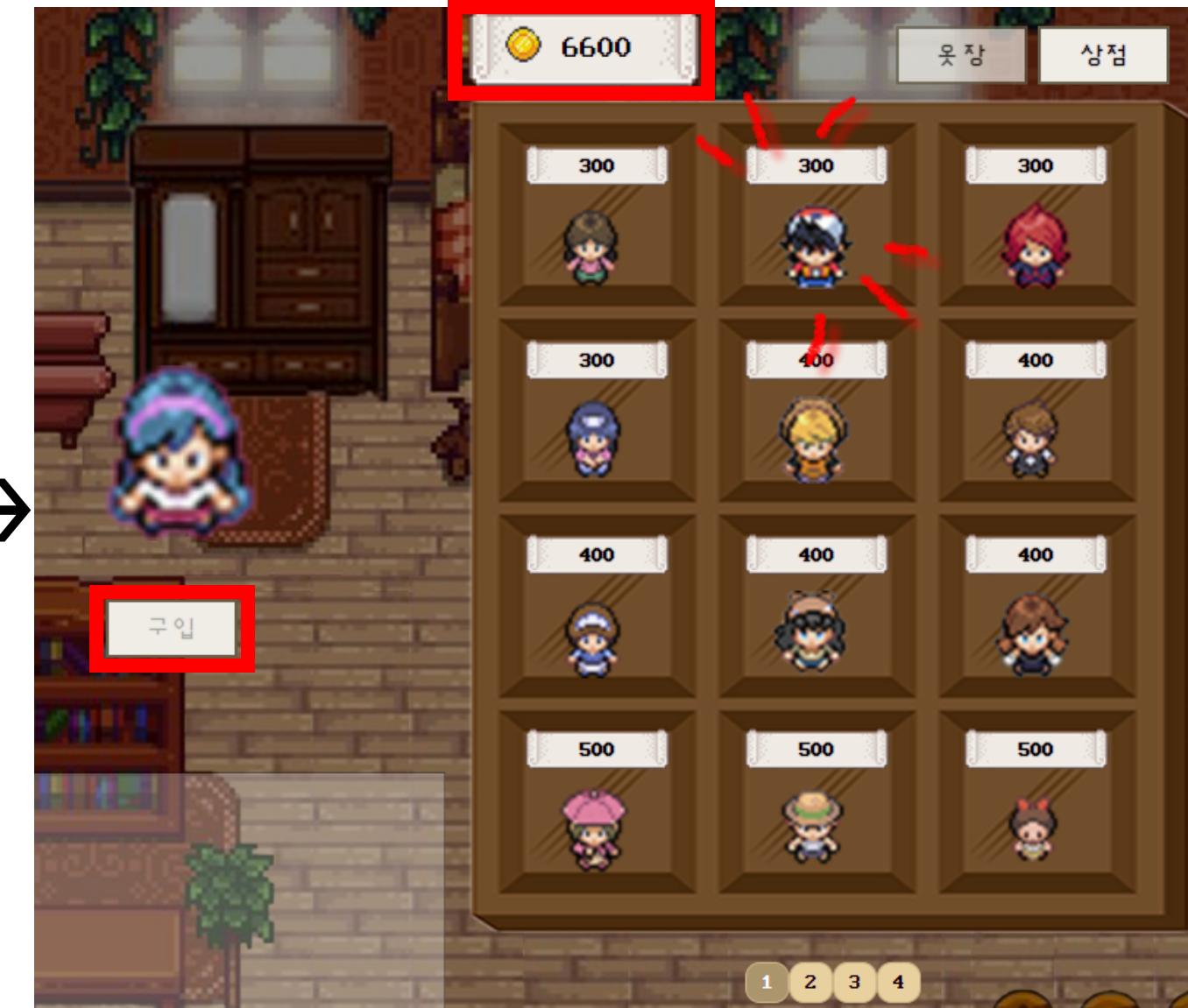
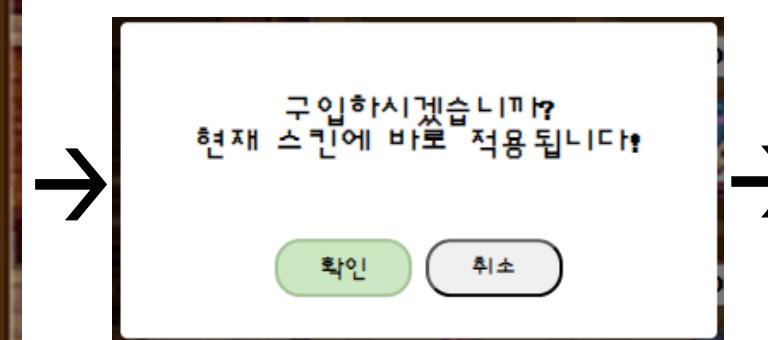
5 한 화면에 스킨 개수 12개 표시
스킨 개수에 따라 페이징 적용되도록 설정



상점 - [구입]



로그인 유저의
현재 스킨



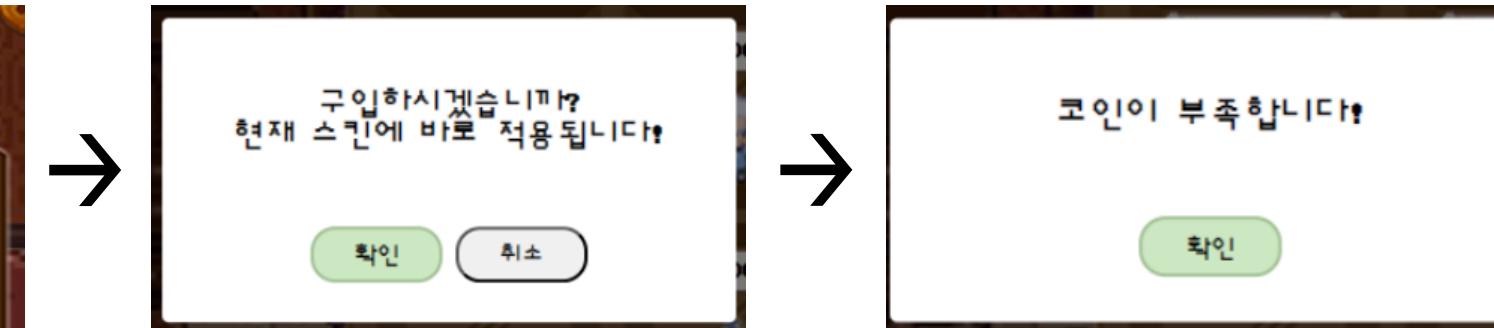
1. 구입하고자 하는 스킨 클릭
 - a. 왼쪽 미리보기 스킨에 이미지 적용
 - b. 구입 버튼 활성화
2. 구입버튼 클릭
 - o MEMBER 테이블에 COIN UPDATE
 - o CHARACTER 테이블에 구입스킨 INSERT

구입하기 버튼 클릭시
구입확인 알림창 발생

1. 현재 스킨으로 바로 적용
DB MEMBER-테이블 → SKIN_ID 업데이트
2. 구입 버튼 비활성화
3. 구입한 코인만큼 차감하여 보유 코인 표시
4. 구입한 스킨 제외
→ 스키니스트 함수 실행

상점 - [구입]

<보유 코인보다 더 큰 가격의 스킨 구입시>



<코드리뷰>

```
public int insertMySkin(String userId, int skinId) {  
    Connection conn = getConnection();  
    int result1 = new SkinDao().updateCoin(conn, userId, skinId);  
    int result2 = 0;  
    if(result1 > 0) {  
        result2 = new SkinDao().insertMySkin(conn, userId, skinId);  
        if(result2 > 0) {  
            commit(conn);  
        } else {  
            rollback(conn);  
        }  
    } else {  
        rollback(conn);  
    }  
    if(result1*result2 == 0) {  
        return -1;  
    }  
    return result1 * result2;  
}
```

```
<!-- 구입시 코인 차감 -->  
<entry key="updateCoin">  
    UPDATE MEMBER  
        SET COIN = COIN - ( SELECT CHARACTER_PRICE  
            FROM CHARACTER_SKIN  
            WHERE SKIN_ID = ? )  
  
        WHERE USER_ID = ?  
        AND COIN >= ( SELECT CHARACTER_PRICE  
            FROM CHARACTER_SKIN  
            WHERE SKIN_ID = ? )  
</entry>
```

```
<!-- 구입한 스키н insert -->  
<entry key="insertMySkin">  
    INSERT INTO CHARACTER(USER_ID, SKIN_ID)  
    VALUES (?, ?)  
</entry>
```

로그인 유저 ID 와 구입하고자 하는 SKIN_ID 값을 넘겨줌

- COIN UPDATE 가 실행되어야
- CHARACTER 테이블에 구입스킨 INSERT 가능

→ 하나의 로직에서 두가지 동시 실행되도록 구현

MEMBER 테이블에 COIN UPDATE 시
조건에 현재 내가 보유하고 있는 코인이
구매하고자 하는 코인보다 크거나 같을 경우에만
UPDATE 실행됨

비동기 페이징 처리



페이지 처리할 변수 전역변수로 선언

```
let listCount;
let skinLimit = 12;
let pageLimit = 5;
let globalCurrentPage = 1;
let skinList = [];
let maxPage;
let startPage;
let endPage;
```

- 현재 총 스킨 갯수
- 한 페이지에 나타낼 스킨 수
- 페이지 하단에 보여질 페이징바의 페이지 최대 갯수(페이지 목록을 몇개단위로 출력할건지)
- 현재 페이지(사용자가 요청한 페이지) → 비동기 요청으로 첫페이지 1로 고정
- 표시하려는 스킨 리스트
- 가장 마지막 페이지가 몇번 페이지인지(총 페이지 수)
- 페이지 하단에 보여질 페이징바의 시작 수
- 페이지 하단에 보여질 페이징바의 끝 수

상점 스킨 리스트 보여주는 함수(옷장 스킨 리스트도 동일하게 진행)

```
function selectSkinList(){
  $.ajax({
    url : path + "/skinList.my",
    success : function(list){
      listCount = list.length; → 총 스킨개수
      skinList = [];
      DB로부터 얻어온 스킨리스트 배열에 담기
      for (let i = 0; i < list.length; i++) {
        skinList.push({
          skinId : list[i].skinId,
          saveRoot : list[i].saveRoot,
          price : list[i].price,
          reward : list[i].reward
        });
      }
      //글 목록 표시 호출 (테이블 생성)
      displayData(1, skinLimit);
      //페이지 표시 호출
      paging(listCount, skinLimit, pageLimit, 1);
    }
  })
}
```

글 목록 표시 함수

```
function displayData(currentPage, skinLimit) {
  let str = "";

  //Number로 변환하지 않으면 아래에서 +를 할 경우 스트링 결합이 되어버림..
  currentPage = Number(currentPage);
  skinLimit = Number(skinLimit);

  let maxpnum = (currentPage - 1) * skinLimit + skinLimit;
  if (maxpnum > listCount) {
    maxpnum = listCount;
  }

  if($(".store-btn").css("opacity") == 1){
    for (let i = (currentPage - 1) * skinLimit; i < maxpnum; i++) {
      str += "<div class='closet-item'>" +
        "<div class='closet-skin-id' id='skin"+i+"'" + "style='display: none;'>" + skinList[i].skinId + "</div>" +
        "<div class='closet-price' id='"+skinList[i].price+"'" + skinList[i].price + "</div>" +
        "<div class='closet-skin'>" +
        "<img src='."+ skinList[i].saveRoot +"/fs.png' id='"+skinList[i].skinId+"'" +
        "</div>" +
        "</div>";
    }
    $(".store-skins").html(str); → 상점
  }else if($(".dress-btn").css("opacity") == 1){
    for (let i = (currentPage - 1) * skinLimit; i < maxpnum; i++) {
      str += "<div class='closet-item'>" +
        "<div class='closet-skin-id' id='myskin"+i+"'" + "style='display: none;'>" + skinList[i].skinId + "</div>" +
        "<div class='closet-skin'>" +
        "<img src='."+ skinList[i].saveRoot +"/fs.png'" +
        "</div>" +
        "</div>";
    }
    $(".closet-skins").html(str); → 옷장
  }
}
```

옷장/상점 버튼 클릭 시

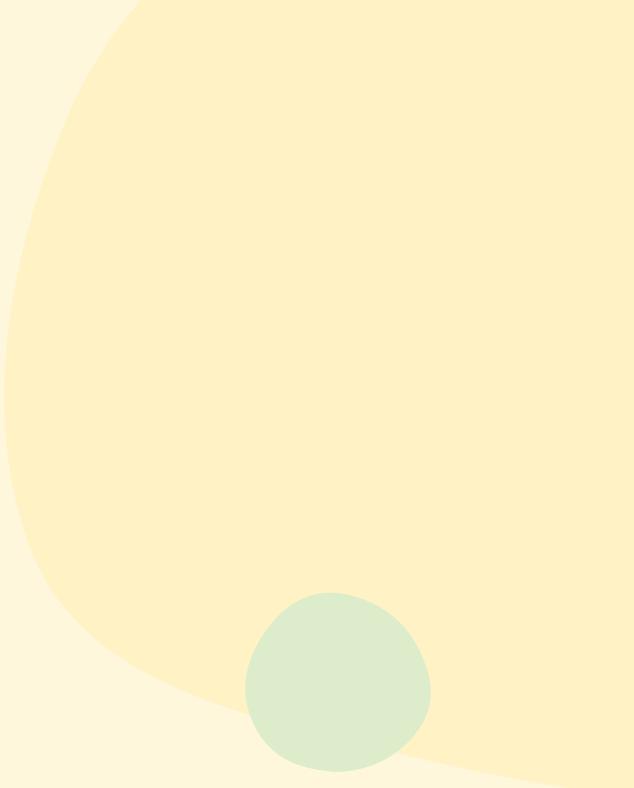
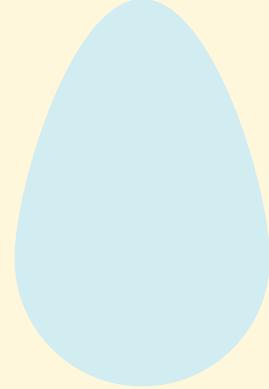
별도의 새로고침 없이 비동기 통신을 사용하여 데이터를 구성하였음

각각의 스키니 리스트를 얻어오는 과정에서

페이징 처리 역시 JSP파일 구성단계에서 데이터를 불러오지 않고

각 페이지 버튼 클릭 시

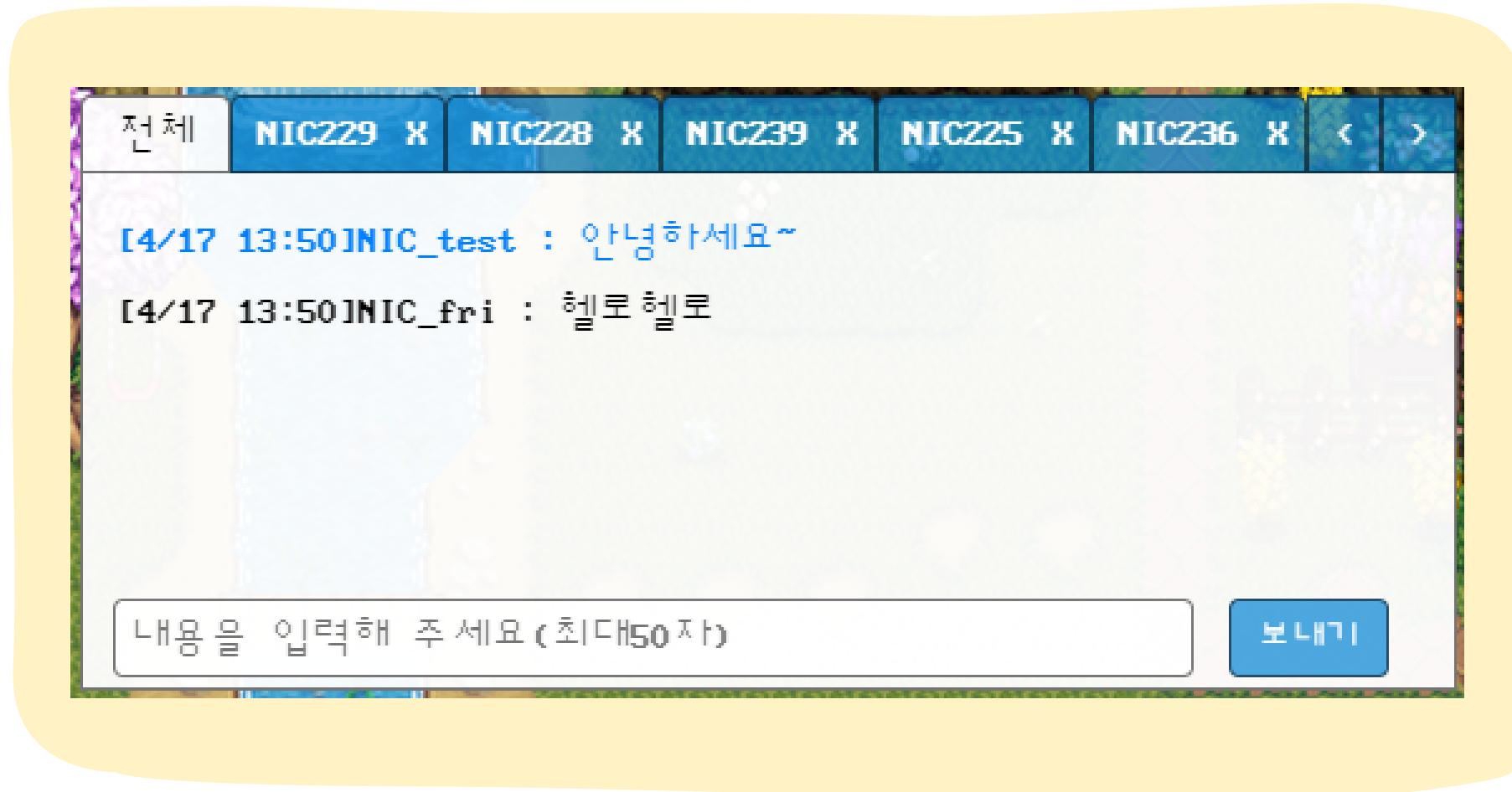
클릭한 버튼에 해당하는 데이터 목록을 불러오는 방식으로 별도의 페이지 새로고침 없이 처리



한승은



Chat_소개



채팅

- 사용자간의 소통을 위한 채팅 화면
- 여러 화면에서 공통적으로 사용
- 전체 채팅, 1:1 채팅으로 구성
- 각 채팅방 탭을 클릭하면 해당 방의 채팅내용 확인 가능

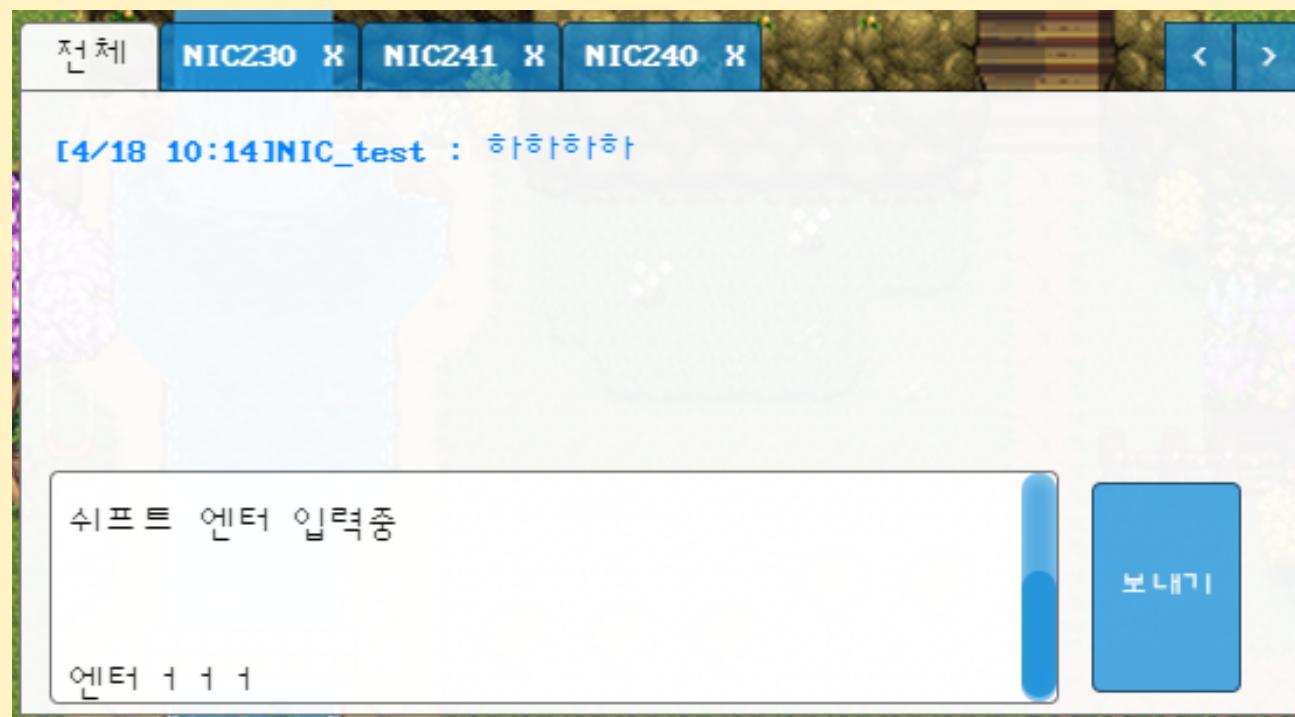
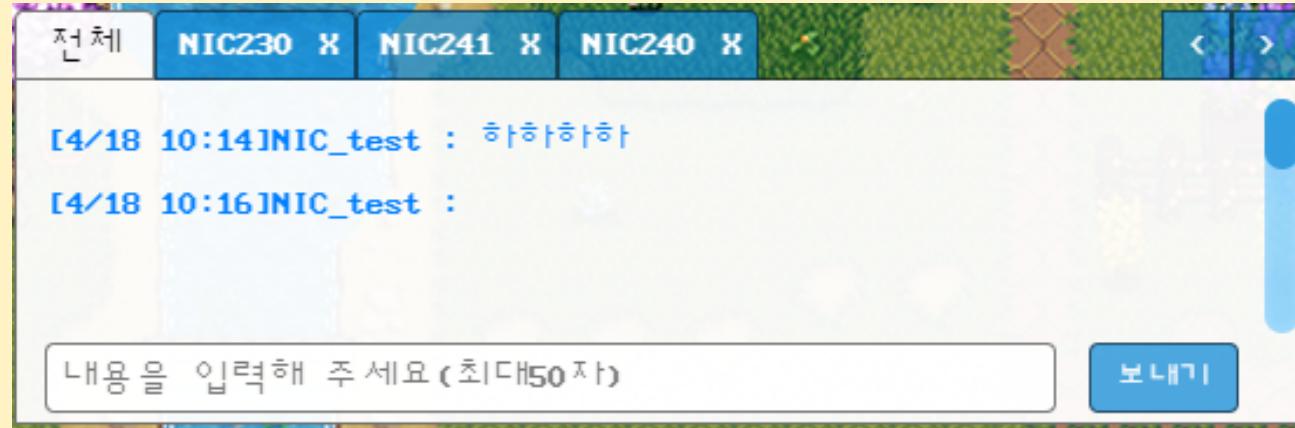
Chat_스타일



채팅창 스타일

#	종류	내용
1	배경색 활성화 비활성화	<ul style="list-style-type: none">- CSS파일 상 ':root'로 변수를 설정하여 채팅창 전체 색상 관리- 채팅창 외부 클릭 시 배경색 비활성화 색상으로 변경- 채팅 내부 클릭 시 배경색 활성화 색상으로 변경- 채팅창 외부에서 enter키를 입력 시 채팅입력칸 focus되며 배경색도 활성화 색상으로 변경
2	채팅방	<ul style="list-style-type: none">- 1:1 채팅방의 텍스트가 길 경우 말줄임표 처리- 사용자 채팅방 탭 클릭 시 클릭된 탭, 클릭되지 않은 탭 각 종류별 너비 변경처리- 선택되어있는 채팅방 배경색 변경 처리- 만약 현재 표시해야하는 채팅방이 5개 미만일경우 빈 채팅방 칸은 보이지 않도록 처리
3	화살표	<ul style="list-style-type: none">- 1:1 채팅방개수가 5개 이하일 경우 좌우 화살표 숨김 처리- 좌우 화살표 클릭 시 이전/다음 채팅방 리스트 5개씩 불러와 표시- 현재 표시되는 채팅방이 리스트의 시작점이라면 이전 화살표 클릭 시 동작하지 않도록 처리- 현재 표시되는 채팅방이 리스트의 마지막이라면 다음 화살표 클릭 시 동작하지 않도록 처리

Chat_스타일



채팅창 스타일

#	종류	내용
4	높이 변경	<ul style="list-style-type: none">- 마우스 드래그로 채팅창 높이 조절 가능- 드래그 가능한 위치에 마우스 오버시 마우스 커서 모양 변경
5	채팅 내용칸	<ul style="list-style-type: none">- 사용자가 조절한 크기보다 표시할 내용이 많을경우 스크롤바 생성- 본인의 채팅일 경우 채팅 텍스트 별도의 색상처리- 타인의 채팅일 경우 기본 텍스트 색상 black
6	채팅 입력칸	<ul style="list-style-type: none">- shift + enter 키를 입력했을 경우 줄바꿈 표시되며 입력칸 높이가 늘어나도록 처리- 최대 4줄까지 표시되며 내용이 더 길어질 경우 높이는 더이상 늘어나지 않고 스크롤바 생성- enter키 단독으로 눌렸거나 보내기 버튼 클릭 시 작성한 채팅 내용 비워지며 다시 한줄만 표시- 채팅 입력칸이 active 상태가 아닐때 enter키 입력이 발생한다면 채팅 입력칸으로 자동 focus되도록 처리 -> 다른 입력화면이 떠있을 경우엔 동작하지 않도록 처리

Chat_기능

#	종류	내용
1	채팅방	공통 <ul style="list-style-type: none">- 페이지 이동시 기존에 선택중이던 채팅방 그대로 선택되어 보이도록 처리- default 채팅방으로 기본생성되며 삭제불가한 채팅방
		전체 채팅방 <ul style="list-style-type: none">- 전체 채팅의 채팅내용은 DB상 저장되지 않으며 접속중에 발생한 채팅 내용들만 확인 가능- 현재 접속중인 전체 유저들간 채팅 가능한 채팅방
		<ul style="list-style-type: none">- 사용자간 자유롭게 추가/제거 가능한 채팅방- 사용자 정보창의 1:1채팅 클릭 시<ul style="list-style-type: none">->기존에 생성되어있지 않은 채팅방이라면 새로 생성되며 해당 채팅방 탭 자동선택->기존에 생성되어있었던 채팅방이라면 해당 채팅방 탭 자동 선택
		1:1 채팅방 <ul style="list-style-type: none">- 상대 유저의 닉네임으로 탭 생성 되며 1:1채팅 내용은 DB상 저장되어 재로그인시에도 최근 7일 내의 채팅 내역 조회 가능- 채팅방이 열려있지 않은 상대에게서 채팅이 수신될 경우 자동으로 채팅방 탭 추가(추가만 될 뿐 자동으로 열리지는 않게끔 처리)- 채팅내용은 DB와 세션스토리지 상 저장되며 DB를 조회하여 내역을 불러올 경우 한번에 50개씩 채팅 내역 불러와도록 처리- 스크롤이 최상단에 닿았을경우 현재 표시되고있는 채팅내역보다 이전의 채팅내역이 있다면 한번에 50개씩 불러올 수 있도록 처리- 새로 채팅이 불러와질 때 이전에 보고 있었던 채팅 내역에 스크롤 위치되어있도록 처리

Chat_기능

#

종류

내용

#	종류	내용
2	채팅	<ul style="list-style-type: none">- 연결된 소켓을 통하여 사용자가 입력한 채팅 내용을 전송할 수 있는 기능- 소켓 연결시 HttpSession상 저장되어있는 로그인 유저 객체를 별도로 UserProperties 저장해두고 1:1채팅 시 전송할 상대를 찾을 때 연결된 유저들의 구분 용도로 사용- 현재 선택되어있는 탭의 상대에게로 채팅 전송(현재 접속중인지 여부 관계 없이)- 채팅 내역의 스크롤을 위로 옮겨 확인중이었더라도 채팅 전송시 스크롤 최 하단으로 위치되도록 처리- 채팅 전송시 선택된 채팅방 종류에 따라 DB상 내역저장, 소켓으로 채팅전송 되도록 처리- 채팅 입력칸이 선택되어 있는 상태에서 enter키가 눌리거나 보내기 버튼 클릭 시 채팅 전송되도록 처리- 채팅입력칸의 입력글자가 없을경우 전송버튼을 눌러도 전송되지 않도록 처리
3	채팅 내역	<ul style="list-style-type: none">- 연결된 소켓을 통하여 상대방이 보낸 채팅 수신기능- 현재 열려있는 탭의 상대가 보낸 채팅일경우 채팅내역화면 하단에 신규 채팅 추가 및 세션스토리지 상 해당 채팅 추가- 현재 열려있지 않은 탭의 상대가 보낸 채팅일경우 세션스토리지 상 해당 상대와의 채팅내역이 있을경우에만 내역 추가하여 저장- 채팅내역 화면 상 신규 채팅 추가 시<ul style="list-style-type: none">-> 채팅 스크롤이 하단에 있을 경우 스크롤 그대로 하단 고정-> 채팅 스크롤이 위로 옮겨져 있을 경우 신규 채팅 추가되어도 스크롤 위치 변동 없도록 처리- 1:1 채팅방의 경우 최근 7일간의 채팅내역 확인 가능- "[전송시간]유저닉네임 : 채팅내용" 형태로 채팅 아이템 표시- 가장 최근에 보내진 채팅 내역이 하단에 표시되도록 정렬

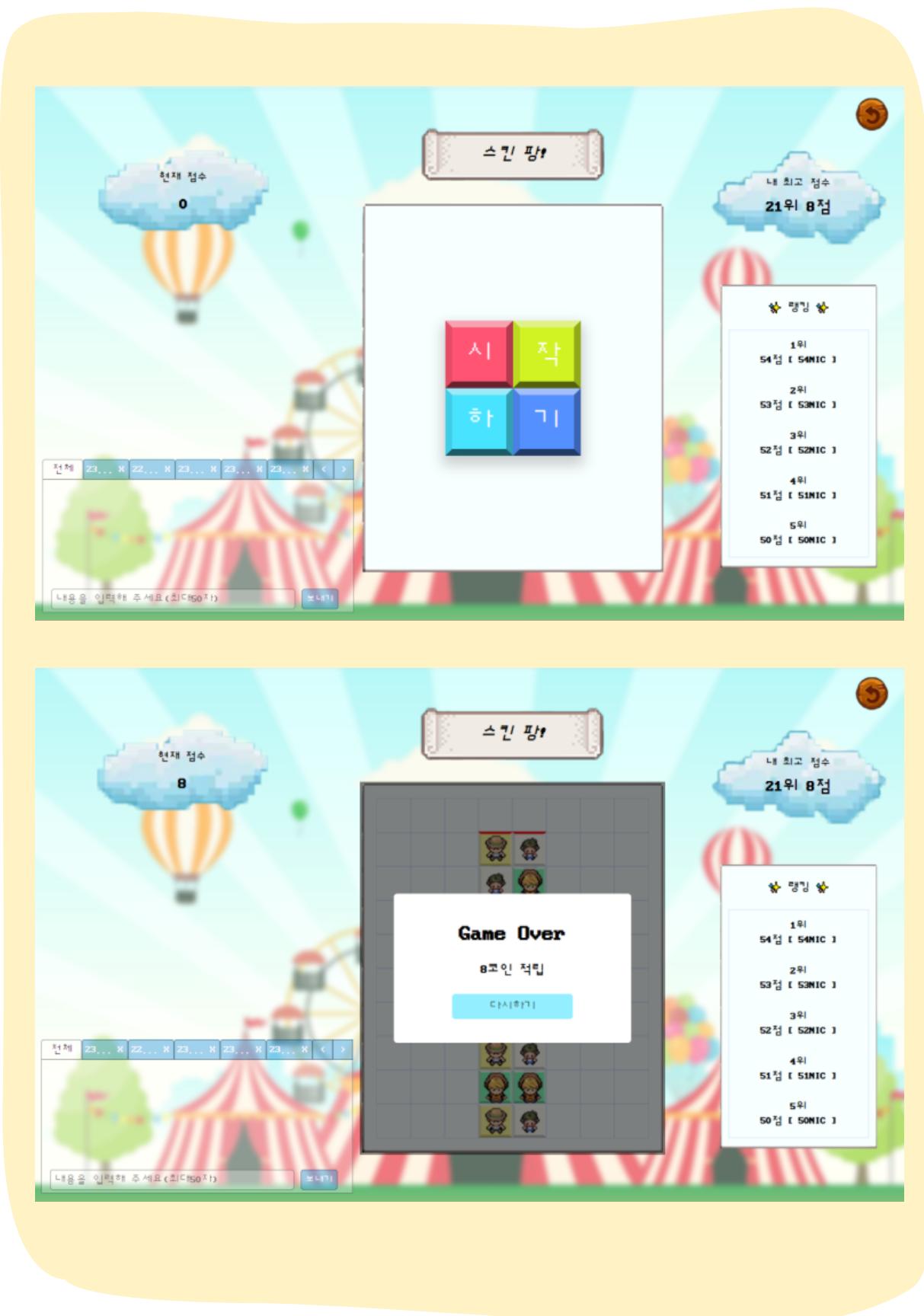
Game_ 스킨팡 소개



SkinPang

- 캐릭터 스킨을 구입하기 위한 코인을
얻을 수 있는 미니게임
- 플레이 점수에 따라 코인 지급

Game_스킨팡



SkinPang

#	종류	내용
1	SkinPang	<ul style="list-style-type: none"> - 위에서부터 캐릭터 블럭 두개가 일정 시간마다 하강하도록 처리 - 좌/우/아래로 블럭을 자유롭게 움직이며 최대한 많은 블럭을 합치는 것을 목표로 하는 게임 - 동일 캐릭터 블럭 3개가 모이면 다음단계의 블럭 1개로 합쳐지며 점수 1점 추가 - 하강하던 블럭이 쌓여있는 블럭을 만나거나 제일 아래에 도달했을 경우 위치 고정되며 더 이상 움직이지 않도록 처리 - 움직일 수 있는 두개의 블럭 중 하나가 위치 고정되었을 경우 남은 하나는 여전히 움직이고, 하강할 수 있도록 처리 - 상단의 붉은색 선까지 블럭이 차올랐을 경우 게임 오버 처리 - 게임 오버 시 현재 누적된 점수에 따라 코인이 지급되며 현재 점수를 과거 점수와 비교하여 더 높을 경우 랭킹에 반영
2	키 입력	<ul style="list-style-type: none"> - 스페이스바 : 움직이는 중의 블럭을 제일 아래로 바로 하강시킴 - 좌/우 방향키 : 움직이는 중의 블럭 위치를 좌/우 한칸씩 이동시킴 - 아래 방향키 : 움직이는 중의 블럭 위치를 아래로 한칸 이동시킴 - 위 방향키 : 두개의 블럭이 모두 움직이는 중일 때만 동작하며 양 블럭의 위치를 서로 바꾸게 됨
3	점수/랭킹	<ul style="list-style-type: none"> - 현재점수 : 게임 플레이 중일 경우 현재 점수 실시간 표시 - 내 최고 점수 : 플레이 점수 기록 중 가장 높은 점수가 표시되며 전체 유저 플레이 기록 중 몇위인지 랭킹 표시 - 랭킹 : 점수 기록 중 1~10위까지의 유저 닉네임, 점수, 랭킹 표시

Admin_소개



관리자 페이지

- 사이트 내부 관리를 위한 관리자 페이지
- admin 계정으로 접속 가능
- 사용자 , 캐릭터, 공지사항, 그외 데이터 삭제 등 관리 가능
- 화면 사이즈 조절 가능

Admin_메뉴_1



기본메뉴

#	종류	내용
1	사이드 바	<ul style="list-style-type: none">- 기본으로 좌측에 떠있도록 처리- 사이드바의 메뉴 클릭시 해당 메뉴 화면으로 페이지 이동처리
2	로그아웃 버튼	<ul style="list-style-type: none">- 기본으로 우측상단에 떠있게 처리- 클릭시 로그아웃되며 자동로그인 정보 삭제 처리- 로그아웃 후 로그인 화면으로 화면이동
3	Dashboard	<ul style="list-style-type: none">- 어드민페이지 접속시 기본적으로 표시되는 메뉴- 현재 사용자 수, 공지 수, 캐릭터 수 확인 가능- 각 메뉴 클릭 시 해당 관리 화면으로 이동

Admin_메뉴_2

The screenshot shows the Admin menu 2 interface with the following sections:

- Admin:** Shows 0개 (0 items) and a red "일괄삭제" (Batch Delete) button. A note says "※7일이상 지난 채팅내역을 삭제합니다" (Delete chat logs older than 7 days).
- 비밀번호 변경:** Shows 30개 (30 items) and a red "일괄삭제" (Batch Delete) button. A note says "※현재 랭킹 집계기간 이전의 호감도내역을 삭제합니다" (Delete history records before the current ranking accumulation period).
- 채팅내역 삭제:** A table showing 7 days of chat log counts:

#	아이디	닉네임	신고건	대상건	탈퇴일
50	t307	NIC307	0	0	23/03/30 15:11
49	t279	NIC279	0	0	23/03/30 15:11
48	t321	NIC321	0	0	23/03/30 15:11
47	t293	NIC293	0	0	23/03/30 15:11
46	t255	NIC255	0	0	23/03/30 15:11
45	t313	NIC313	0	0	23/03/30 15:11
44	t333	NIC333	0	0	23/03/30 15:11
43	t331	NIC331	0	0	23/03/30 15:11
- 호감도 내역 삭제:** A table showing 15 days of history record counts:

#	아이디	닉네임	신고건	대상건	탈퇴일
50	t307	NIC307	0	0	23/03/30 15:11
49	t279	NIC279	0	0	23/03/30 15:11
48	t321	NIC321	0	0	23/03/30 15:11
47	t293	NIC293	0	0	23/03/30 15:11
46	t255	NIC255	0	0	23/03/30 15:11
45	t313	NIC313	0	0	23/03/30 15:11
44	t333	NIC333	0	0	23/03/30 15:11
43	t331	NIC331	0	0	23/03/30 15:11
- 탈퇴계정 삭제:** Shows a red "일괄삭제" (Batch Delete) button. A note says "※탈퇴 후 15일이 지난 계정들을 삭제합니다" (Delete accounts 15 days after retirement).

Admin

#	종류	내용
1	Admin	<ul style="list-style-type: none">- 사이드 바에서 Admin 클릭 시 이동되는 화면- admin 계정의 비밀번호 변경 가능 메뉴- 기존비밀번호가 맞지 않거나 신규 비밀번호/비밀번호 확인의 내용이 서로 틀릴경우 화면 하단에 toast 메세지로 비밀번호 변경 실패메세지 표시- 비밀번호 변경 성공시 입력내용 지워지며 화면 하단에 toast메세지로 변경 성공 메세지 표시
2	비밀번호 변경	
3	채팅내역 삭제	<ul style="list-style-type: none">- 7일이 지난 채팅내역 개수 확인 및 일괄삭제할 수 있는 메뉴- 일괄삭제 클릭시 삭제 확인 메세지 표시 후 다시 확인 클릭이 있어야 DB내역 삭제처리 진행- 현재날짜부터 7일 이전의 채팅내역만 삭제 처리
4	호감도 내역 삭제	<ul style="list-style-type: none">- 호감도 랭킹에 집계되지 않는 (가장 최근 월요일로부터 -7일보다 이전의 내역들) 내역 개수 확인 및 일괄삭제할 수 있는 메뉴- 일괄삭제 클릭 시 확인메세지 표시 후 다시 확인클릭이 있어야 삭제처리 진행
5	탈퇴계정 삭제	<ul style="list-style-type: none">- 탈퇴후 15일이 지난 계정들 내역 확인 및 일괄삭제 가능한 메뉴- 탈퇴후 15일이 지난 계정 테이블 확인 가능- 각 계정클릭시 해당 유저 상세 화면으로 이동- 일괄삭제 클릭시 삭제 확인 메세지 표시 후 다시 확인 클릭이 있어야 DB내역 삭제처리 진행

Admin_메뉴_3

스킨 등록

#	대표사진	풀더명	가격(코인)	보상
39		skin39	3000	N
38		skin38	3000	N
37		skin37	3000	N
36		skin36	3000	N
35		skin35	3000	N
34		skin34	3000	N
33		skin33	1200	N
32		skin32	1200	N
31		skin31	1200	N
30		skin30	1000	N

1 2 3 4

캐릭터

#	종류	내용
1	캐릭터	- 사이드 바에서 캐릭터 클릭 시 이동되는 화면
2	공통	- 캐릭터 스킨 클릭시 해당 스킨관리화면으로 이동 - 캐릭터 스킨리스트 스킨등록일 기준 최신순으로 표시
3	Default 스킨	- 0번스킨으로 사용자 계정 생성시 기본으로 주어지는 스킨
4	보상용 스킨	- 호감도 보상으로 주어지는 스킨 리스트 - DB상 저장된 스킨 중 보상용으로 저장되어있는 내역들 표시
5	일반 스킨 리스트	- Default, 보상용이 아닌 일반 스킨 리스트 표시 - 스킨 이미지, 저장된 폴더명, 가격, 보상용 유무 확인 가능 - 페이징 바 처리하여 한 페이지당 10개씩 표시
6	스킨등록 버튼	- 신규 스킨 등록 가능한 페이지로 이동

캐릭터 - 스킨등록/수정

캐릭터 스킨 등록

가격(코인) : 보상용(Y/N) : 등록

※ 확장자가 'png'인 파일만 등록 가능합니다.(최대 10MB)

앞(fs)	앞(fd)	뒤(bs)	뒤(bd)

좌(ls)	좌(ld)	우(rs)	우(rd)

skin39 설정

가격(코인) : 보상용(Y/N) : 수정 삭제

※ 확장자가 'png'인 파일만 등록 가능합니다.(최대 10MB)

앞(fs)	앞(fd)	뒤(bs)	뒤(bd)

좌(ls)	좌(ld)	우(rs)	우(rd)

#	종류	내용
1	캐릭터 스킨등록	<ul style="list-style-type: none"> - 캐릭터 신규 스킨 등록 가능 - 가격/보상유무/이미지 등록 가능 - 가격 기본값 300, 보상 기본값 N - 이미지 파일 등록시 미리보기 표시되도록 처리 - 총 8개의 이미지 모두 첨부진행 해야하며 하나라도 파일이 첨부 되지 않았을 경우 등록 불가 및 toast메세지로 파일 등록 메세지 표시 - 이미지는 png 파일만 가능하며 다른 파일을 첨부하여 등록 진행시 toast 메세지로 확장자 제한 메세지 표시 - 스킨 등록시 서버 저장경로상 신규로 해당 스킨용 폴더 추가되며 생성된 폴더 내부에 실제 이미지 저장되도록 처리
2	캐릭터 스킨수정	<ul style="list-style-type: none"> - 등록되어있는 캐릭터 스킨 수정/삭제 가능한 메뉴 - 현재 수정중인 스킨 저장파일명 상단에 표시 - 스킨 삭제시 실제 저장되어있는 폴더와 이미지들도 삭제되도록 처리
3	뒤로가기 버튼 (공통)	<ul style="list-style-type: none"> - 이전에 보고있었던 캐릭터 페이지로 이동 - 만약 캐릭터 페이지 내에서 페이징바 2번을 선택해 보고있었다면 해당 화면으로 이동

Admin_메뉴_4

일반 계정

아이디 검색어 입력 후 엔터를 눌러주세요

정렬: 아이디 API: ALL Kakao Google

#	아이디	닉네임	코인	API	가입일	상태
245	test	NIC_test	644	카카오	23/04/14	Y
244	t99	NIC99	500	kakao	23/04/14	Y
243	t98	NIC98	500	google	23/04/14	Y
242	t97	NIC97	500	kakao	23/04/14	Y
241	t96	NIC96	500	google	23/04/14	Y
240	t95	NIC95	500	kakao	23/04/14	Y
239	t94	NIC94	500	google	23/04/14	Y
238	t93	NIC93	500	kakao	23/04/14	Y
237	t92	NIC92	500	google	23/04/14	Y
236	t91	NIC91	500	kakao	23/04/14	Y
235	t90	NIC90	500	google	23/04/14	Y
234	t9	NIC9	500	kakao	23/04/14	Y
233	t89	NIC89	500	kakao	23/04/14	Y
232	t88	NIC88	500	google	23/04/14	Y
231	t87	NIC87	500	kakao	23/04/14	Y
230	t86	NIC86	500	google	23/04/14	Y
229	t85	NIC85	500	kakao	23/04/14	Y
228	t84	NIC84	500	google	23/04/14	Y
227	t83	NIC83	500	kakao	23/04/14	Y
226	t82	NIC82	500	google	23/04/14	Y

1 2 3 4 5 6 7 8 9 10 >

사용자 - 일반계정

#	종류	내용
1	사용자 일반계정	- 사이드바 메뉴중 사용자 -> 일반계정 클릭 시 표시되는 화면
2	일반계정	- 현재 계정 상태가 Y(정상)인 계정리스트 확인 가능 - 각 계정 클릭 시 해당 계정 상세 화면으로 이동
3	정렬/필터	- 아이디/가입일로 정렬 기준 변경 가능(내림차순으로 정렬됨) - 계정이 가입한 API 종류에 따라 필터 적용 가능 -> 선택한 필터에 해당하는 데이터만 표시
4	검색	- 아이디/닉네임으로 계정 검색 가능 - 검색어와 일치하는 계정리스트 보여지며 검색어와 유사한 순으로 정렬 - 검색된 계정 클릭 시 해당 계정 상세화면으로 이동
5	페이지	- 현재 선택되어있는 정렬/필터기준으로 페이지 - 버튼 클릭 시 페이지 이동 없이 각 버튼에 해당하는 목록 표시 - 한 페이지당 목록 20개씩 표시

사용자 - 비활성화계정

비활성화 계정

아이디 ▼ 검색어 입력 후 엔터를 눌러주세요

정렬: 아이디 API: ALL Kakao Google
상태: ALL 탈퇴 차단

#	아이디	닉네임	코인	API	가입일	상태	비활성일
100	t342	NIC342	500	google	23/04/14	X	23/03/30
99	t341	NIC341	500	kakao	23/04/14	N	23/03/30
98	t340	NIC340	500	google	23/04/14	X	23/03/30
97	t339	NIC339	500	kakao	23/04/14	N	23/03/30
96	t338	NIC338	500	google	23/04/14	X	23/03/30
95	t337	NIC337	500	kakao	23/04/14	N	23/03/30
94	t336	NIC336	500	google	23/04/14	X	23/03/30
93	t335	NIC335	500	kakao	23/04/14	N	23/03/30
92	t334	NIC334	500	google	23/04/14	X	23/03/30
91	t333	NIC333	500	kakao	23/04/14	N	23/03/30
90	t332	NIC332	500	google	23/04/14	X	23/03/30
89	t331	NIC331	500	kakao	23/04/14	N	23/03/30
88	t330	NIC330	500	google	23/04/14	X	23/03/30
87	t329	NIC329	500	kakao	23/04/14	N	23/03/30
86	t328	NIC328	500	google	23/04/14	X	23/03/30
85	t327	NIC327	500	kakao	23/04/14	N	23/03/30
84	t326	NIC326	500	google	23/04/14	X	23/03/30
83	t325	NIC325	500	kakao	23/04/14	N	23/03/30
82	t324	NIC324	500	google	23/04/14	X	23/03/30
81	t323	NIC323	500	kakao	23/04/14	N	23/03/30

1 2 3 4 5

#	종류	내용
1	사용자 비활성화계정 화면	- 사이드바 메뉴중 사용자 -> 탈퇴/차단계정 클릭 시 표시되는 화면
2	비활성화계정	- 현재 계정 상태가 N(탈퇴)/X(차단)인 계정리스트 확인 가능 - 각 계정 클릭 시 해당 계정 상세 화면으로 이동
3	정렬/필터	- 아이디/가입일로 정렬 기준 변경 가능(내림차순으로 정렬됨) - 계정이 가입한 API 종류, 현재 계정 상태에 따라 필터 선택 가능 -> 선택한 필터에 해당하는 데이터만 표시
4	검색	- 아이디/닉네임으로 계정 검색 가능 - 이하 일반계정 내용과 동일
5	페이지	- 현재 선택되어있는 정렬/필터기준으로 페이지 - 이하 일반계정 내용과 동일

Admin_메뉴_4

유저 정보

적용 스킨 : 0 아이디 : test API : 카카오 상태 : Y

닉네임 : NIC_test	가입일 : 23/04/14 15:11	비활성일 :
코인 : 644	자기소개 :	
성별 : N		

보유중 스킨

0	40	41
---	----	----

사용자 - 유저정보

#	종류	내용
1	유저 정보	<ul style="list-style-type: none">- 사용자 계정 상세 화면- 해당 계정의 상세 정보 조회 가능- 현재 해당 계정이 대표스킨으로 적용중인 스킨아이디, 보유중인 스킨 미리보기/스킨아이디 확인 가능
2	수정	<ul style="list-style-type: none">- 코인, 자기소개 내용 수정 가능- 계정삭제, 차단/차단해제 가능- 계정 차단시 비활성일에 현재 시각 찍히며 상태값 X로 변경 및 친구목록/방명록/채팅방/채팅내역/호감도 테이블 상 해당 유저의 활동내역 일괄 삭제 진행
3	뒤로가기 버튼	<ul style="list-style-type: none">- 버튼 클릭시 이전 화면으로 뒤로가기 수행

Admin_메뉴_4

신고 목록

Q 대상자 ▾ 검색어 입력 후 엔터를 눌러주세요

#	신고번호	제목	신고자	대상자	신고일
80	83	신고제목	t64	t61	23/04/20
79	82	신고제목	t63	t63	23/04/20
78	81	신고제목	t62	t59	23/04/20
77	80	신고제목	t61	t61	23/04/20
76	79	신고제목	t60	t57	23/04/20
75	78	신고제목	t59	t59	23/04/20
74	77	신고제목	t58	t55	23/04/20
73	76	신고제목	t57	t57	23/04/20
72	75	신고제목	t56	t53	23/04/20
71	74	신고제목	t55	t55	23/04/20
70	73	신고제목	t54	t51	23/04/20
69	72	신고제목	t53	t53	23/04/20
68	71	신고제목	t52	t49	23/04/20
67	70	신고제목	t51	t51	23/04/20
66	69	신고제목	t50	t47	23/04/20
65	68	신고제목	t49	t49	23/04/20
64	67	신고제목	t48	t45	23/04/20
63	66	신고제목	t47	t47	23/04/20
62	65	신고제목	t46	t43	23/04/20
61	64	신고제목	t45	t45	23/04/20

1 2 3 4

사용자 - 신고 관리

#	종류	내용
1	신고 관리	<ul style="list-style-type: none"> - 사이드바 메뉴중 사용자 -> 신고 관리 클릭 시 표시되는 화면
2	신고 목록	<ul style="list-style-type: none"> - 사용자들의 신고 목록 확인 가능 - 각 목록 클릭 시 우측에 상세 내용 표시
3	신고 상세	<ul style="list-style-type: none"> - 신고 목록 클릭시 우측에 해당 신고 상세 내용 표시 - 신고자, 신고 대상자 계정 정보 확인 가능 - 유저 조회 클릭 시 해당 계정 상세 화면으로 이동 - 신고건 : 해당 계정의 신고한 수 - 대상건 : 해당 계정의 신고 당한 수 - 신고 제목/내용 수정 가능, 신고 삭제 가능
4	검색	<ul style="list-style-type: none"> - 신고자, 대상자, 신고제목으로 검색 가능 - 검색어 입력 후 엔터 입력시 내용 검색되며, 검색된 내용이 아래 표에 표시됨 - 검색어와 가까운 순/신고일 내림차순 정렬
5	페이지	<ul style="list-style-type: none"> - 신고번호/신고일 기준 내림차순 정렬 - 한 페이지 당 20개의 목록 표시

Admin_메뉴_5

공지사항

#	제목	작성일
5	다섯번째 공지사항 입니다.	23/04/20
4	네번째 공지사항 입니다.	23/04/20
3	게임 업데이트 사항	23/04/20
2	게임 업데이트 사항	23/04/20
1	첫번째 공지사항 입니다.	23/04/20

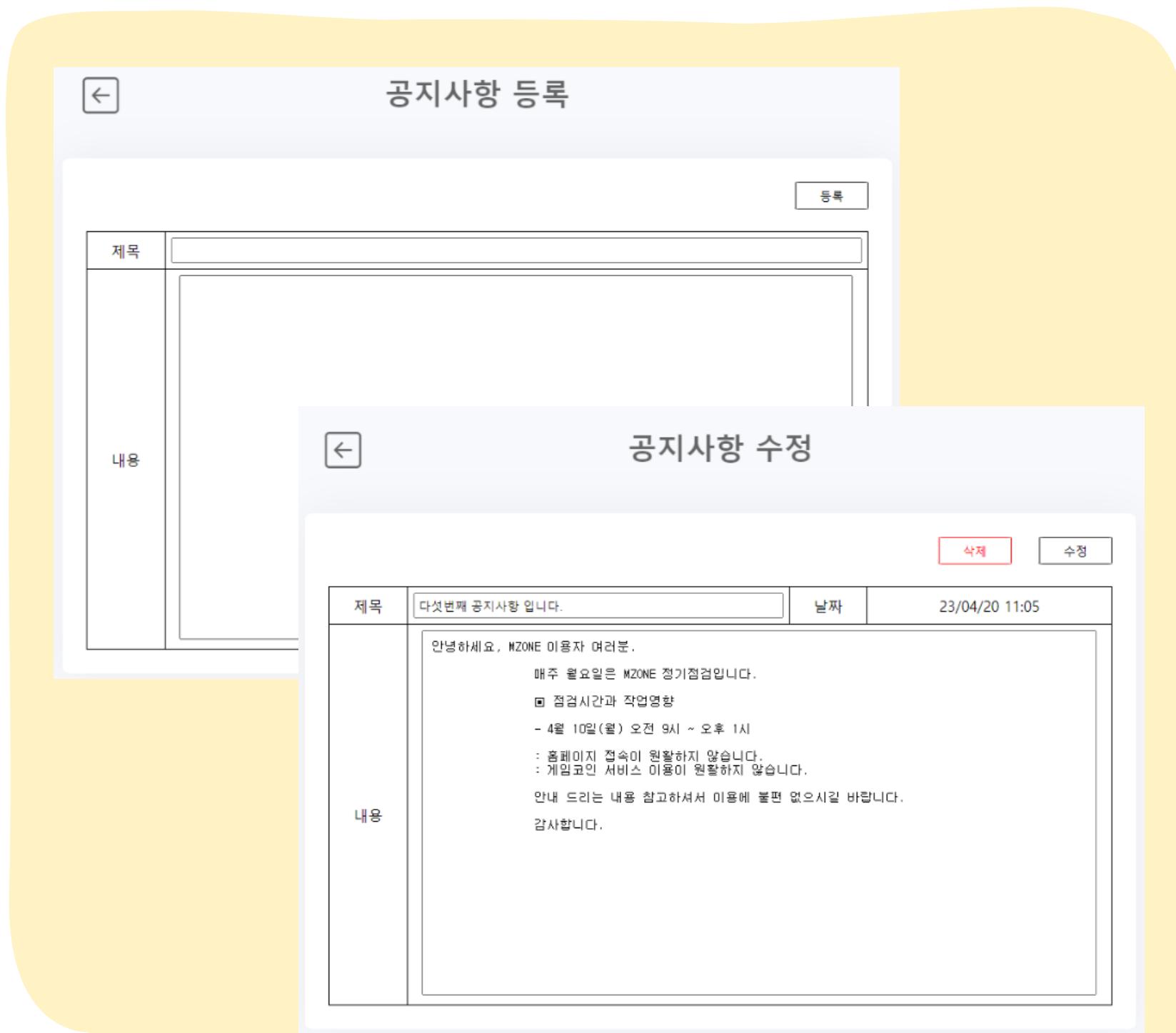
공지 등록

1

공지사항

#	종류	내용
1	공지사항	<ul style="list-style-type: none">- 사이드바 메뉴중 공지사항 클릭 시 표시되는 화면- 공지사항 리스트 표시- 각 공지사항 클릭시 공지사항 수정 페이지로 이동
2	공지 등록	<ul style="list-style-type: none">- 신규 공지사항을 등록할 수 있는 페이지로 이동

Admin_메뉴_5



공지사항 - 수정/등록

#	종류	내용
1	등록	<ul style="list-style-type: none">- 신규 공지를 등록 가능한 화면- 등록 버튼 클릭 시 빈칸이 있을 경우 toast메세지로 입력되지 않은 내용이 있다는 알림 발생- 제목/내용 모두 입력후 공지 등록 가능
2	수정	<ul style="list-style-type: none">- 공지사항 수정/삭제 가능한 화면- 제목/내용 수정 가능- 공지 삭제 가능

기타 담당 내용

#	종류	내용
1	화면 사이즈 고정	<ul style="list-style-type: none">- 원활한 사이트 이용을 위해 사이트 화면 사이즈 고정 로직 작성- 페이지 사이즈 컨트롤을 위해 사이트 접속 시 신규 페이지 팝업 및 사이즈 고정 코드 적용
2	호감도 랭킹 보상	<ul style="list-style-type: none">- 호감도 랭킹에 따른 스킨 보상 지급 로직 작성- 사용자 로그인 시 랭킹을 체크하여 보상용 스킨 지급 및 삭제 진행
3	DB 테이블 작성용 SQL 관리	<ul style="list-style-type: none">- 공통으로 구성되어야 하는 DB 테이블 작성용 SQL 파일 작성 및 관리

작업일정표



JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC



03 MAR

SUN	MON	TUE	WED	THU	FRI	SAT	MEMO
			1	2	3	4	
5	6	7	8	9	10	11	
12	13	14	15	16	17	18	
		프로젝트 기획 및 분석 회의		기획보고서 발표	UI 설계 회의		
19	20	UI 설계 회의	22	23	24	25	
				UI 설계 발표	데이터 베이스 설계		
26	27	28	29	30	31		
데이터 베이스 설계	DB 보고서 제출		프로젝트 구현				

YEARLY

MONTHLY

WEEKLY

DAILY

MOOD

HABIT

LINE

GRID

@heugimja

STICKERS

작업일정표



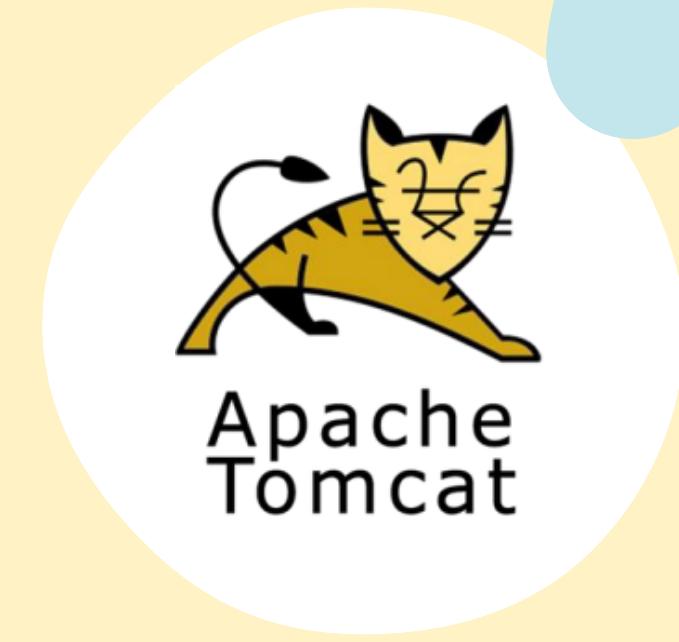
JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC



04 APR

SUN	MON	TUE	WED	THU	FRI	SAT	MEMO
						1	
2	3	4	5 프로젝트 구현	6	7	8	
9	10	11	12 프로젝트 구현	13	14	15	
16	17	18	19 프로젝트 통합테스트 & 디버깅 및 오류수정	20	21	22	
23/30	24 발표준비	25 프로젝트 발표	26	27	28	29	

개발환경



소감



소감



김혜린

아무래도 로그인, 회원가입 등 사용자의 정보와 관련된 코드를 짜다 보니 보안을 생각하게 되었고 그에 맞는 코드를 고려해보며 새롭게 알아가고 적용해볼 수 있는, 발전할 수 있는 시간이었습니다. 또 일의 우선순위를 세워 일을 진행해나가면서 기능의 중요도에 따라 일을 처리해 나가는 경험을 쌓을 수 있었습니다.

다양한 협업 툴을 사용하면서 효율적으로 협업하는 방법에 대해 배울 수 있었고 혼자 모든 것을 해결하려고 애쓰기보다 그때그때 상황이나 변동사항이 있을 때마다 팀원들과 상의, 의논하고 서로 의견을 제시해 도움을 주고 받으면서 협업의 즐거움을 느꼈고 많은 것을 배울 수 있는 시간이었습니다.

윤지영

저희 조 주제가 소통인 만큼 다양한 협업 툴을 사용하였는데, 이를 통해 다양한 의견을 나눌 수 있었습니다.

개발함에서도 지속적인 통합과, 소통으로 혼자 하는 개발이 아닌 여러 명에서 하나의 결과물을 완성한다고 느꼈습니다! 그리고 원래 설정했던 기능들이 많은데도, 이 모든 기능을 구현해서 기쁩니다.

이 프로젝트를 진행하면서 다양한 오류를 만나게 되었는데, 이러한 경험으로 다음에는 더 좋은 결과물을 만들어 낼 수 있을 것 같습니다.

박가영

구현하고자 하는 기능들을 다 비동기 방식으로 구현해야 했는데 비동기 방식을 배우고서 처음으로 혼자서 로직을 구현하고 실행시키려다 보니 헤매는 부분이 많았습니다.

하지만 조원들과 질문을 주고받으면서 이해의 폭이 넓어졌고 여러 정보들을 많이 습득하여 문제없이 기능들을 구현하게 됐습니다.

이번 프로젝트를 경험으로 조원들과의 지속적인 소통과 활발한 협업의 중요성을 배우게 됐고 무사히 프로젝트를 마칠 수 있어서 만족스럽습니다.

노지의

프로젝트를 처음 시작할 때 머릿속으로는 구현이 척척될 것 같았지만, 막상 코드를 짜보려고 하니 여러 기능들이 머리에서 정리가 안되고, 마음만 앞서가고 있는 상황에 부딪혀 초반에 어려움이 있었습니다.

하루에 수행해야 할 목표를 새워두고 해야 할 기능을 하나씩 처리하며 조원들에게 제가 현재 이해한 상태로 작업을 하는 건지 소통하며 의견을 주고 받아 어려움을 극복할 수 있었습니다.

팀 프로젝트를 하면서 조원들과 소통이 활발히 잘 이루어져 만족스러운 프로젝트가 나왔다고 생각합니다.

한승은

저희 조의 프로젝트 주제는 '소통'이었습니다. '소통'은 개발 주제이기도 했지만 이번 세미 프로젝트를 대하는 저희 조 자체의 주제이기도 했습니다. 이번 기회를 통해 여러 명의 팀원들과 협업하는 것을 최대한 경험해보고자 모두가 노력했고, 실제로도 지속적인 회의와 코드 공유를 통해 코드 중복을 방지하고 로직 구현 시간을 단축 시킬 수 있었습니다.

Github 사용과 그 외 협업 툴 사용에도 많이 숙달 될 수 있었던 값진 시간이었던 것 같습니다.

감사합니다



