

# 봉어탐정

“길 잃은 봉어빵을 찾아주는 새로운 방법”

× × × ×

알고 풀어- 강시원, 권지영, 김남희, 박치호, 이종민, 황일찬



# Table of Content

- 프로젝트 배경
- 프로젝트 개요
- 시연
- 프로젝트 후기

# 01. 프로젝트 배경



× × × ×

## '사라지는 봉어빵'…추억도 함께 사라집니다

이데일리 원문 | 기사전송 2025-01-18 09:15 최종수정 2025-01-18 09:21

사회

### 2030 '봉어빵 창업' 현실은... "하루 3만원 벌기도 벅차요"

동아일보 | 업데이트 2023-01-11 11:55 ▾

### '불법' 신고에 사라지는 봉어빵 노점…시민들은 "아쉬워"

윤수진 기자 jjin@imaeil.com

매일신문 입력 2025-01-09 16:26:50 수정 2025-01-09 21:42:14

### 길거리 봉어빵 대신 고급 봉어빵?...‘상생’ 사라진 거리노점

노점 단속 강화에 길거리 봉어빵 가게 사라져

5년 새 거리 노점 약 21% ↓

‘상생’ 목적으로 시작한 제도, 노점 없애는 방향으로 바뀌어

“하나의 문화로 인정하고 포용하는 정책 필요해”

등록 2025-01-06 오전 5:45:00  
수정 2025-01-06 오전 5:45:00

가 가



## “편의점으로 들어간 붕어” 붕어빵 노점이 사라졌다

김하나 기자 | 입력 2024.11.29 | 호수 626 | 댓글 0

표4 틈새 파고드는 편의점·카페	
업체	내용
GS25	2023년 길거리 붕어빵을 재현한 ‘꼬리까지 맛있는 붕어빵’ 출시
	2+1 프로모션 등으로 소비자 확보 나서
	올해 붕어빵 판매 매장 전국 5000개로 확대
	11월 붕어빵 매출액 전년 동기 대비 11.7% 신장
이디야 커피	9~11월 붕어빵 2종 판매량 28만3000개
	기존 미니 붕어빵에서 실제 노점 붕어빵 크기로 리뉴얼

[자료 | 각 사]



## 겨울 단골 아이템 ‘붕어빵’의 진화.. “신봉세권부터 밀키트까지”

입력 2024.12.23 10:09 수정 2024.12.23 10:18 | 최승근 기자 (csk3480@dailian.co.kr)

▶▶▶▶▶

## 02. 프로젝트 개요





# 붕어탐정



1  
붕탕오더

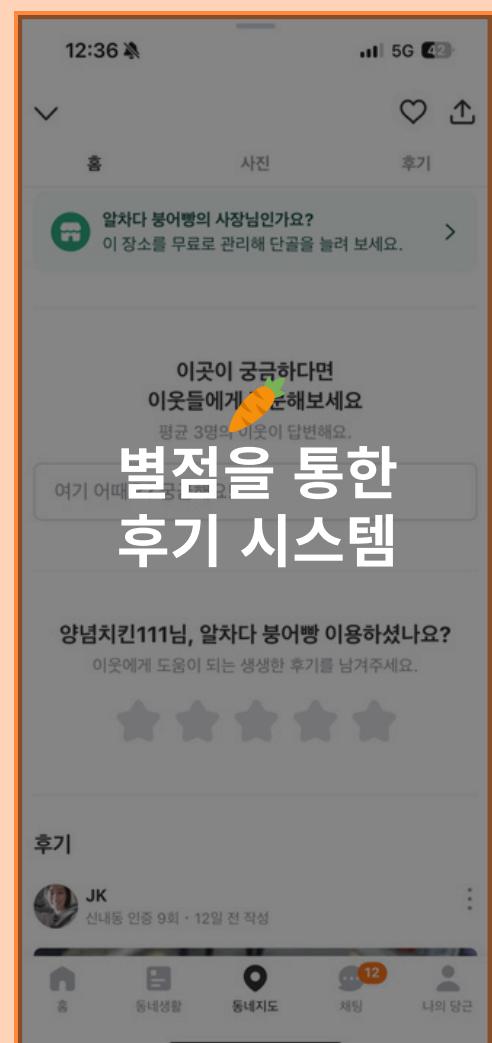
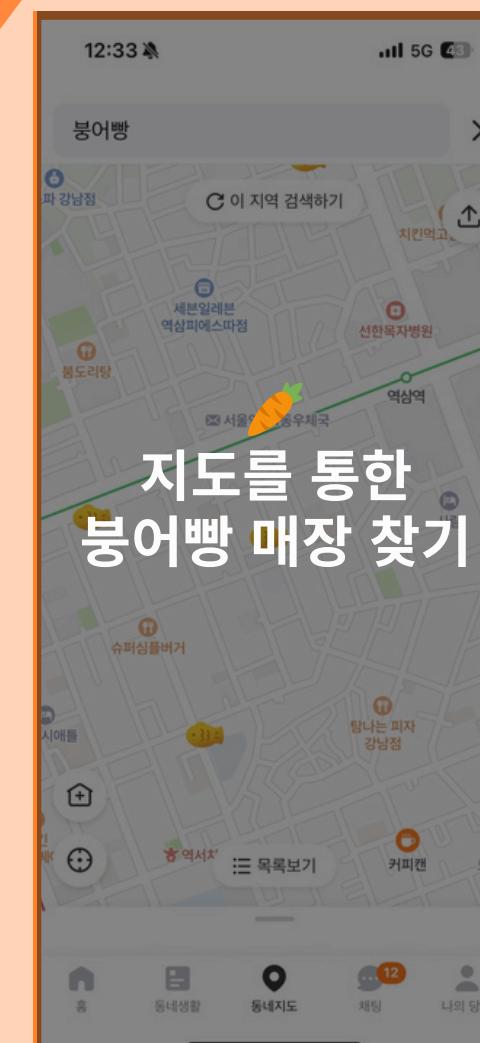
2

붕템샵

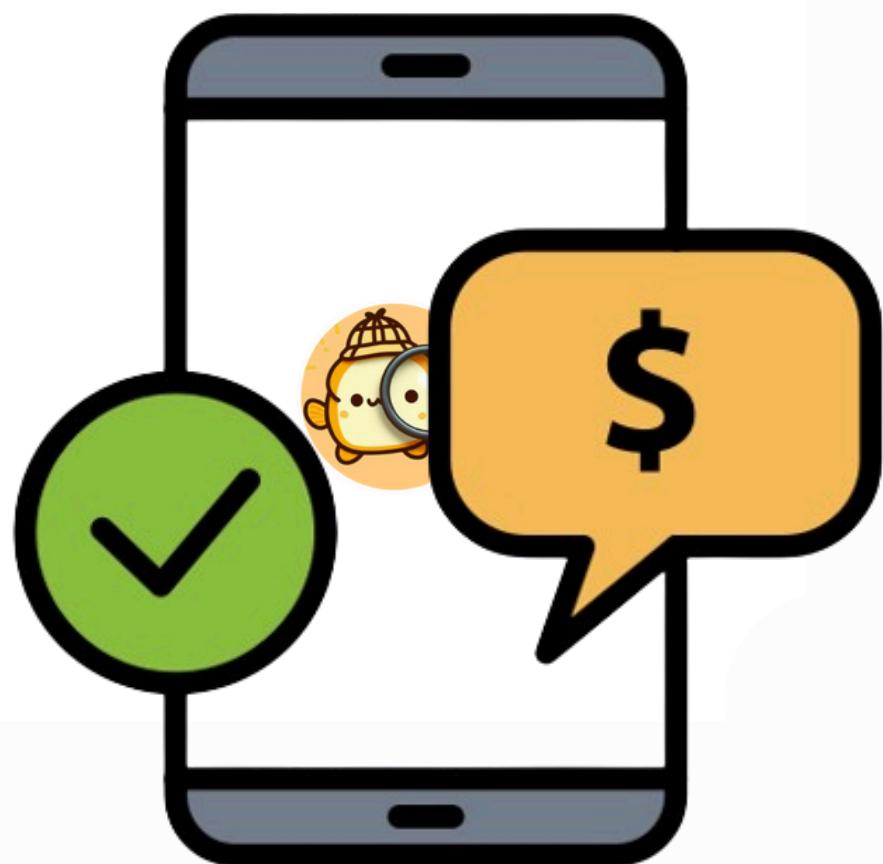
3

커뮤니티

# 붕어탐정



# 봉탐오더



앱내 결제 + 사전 주문 가능





2

# 붕템샵

붕어빵을 사랑하는 사람들을 위한 특별한 공간

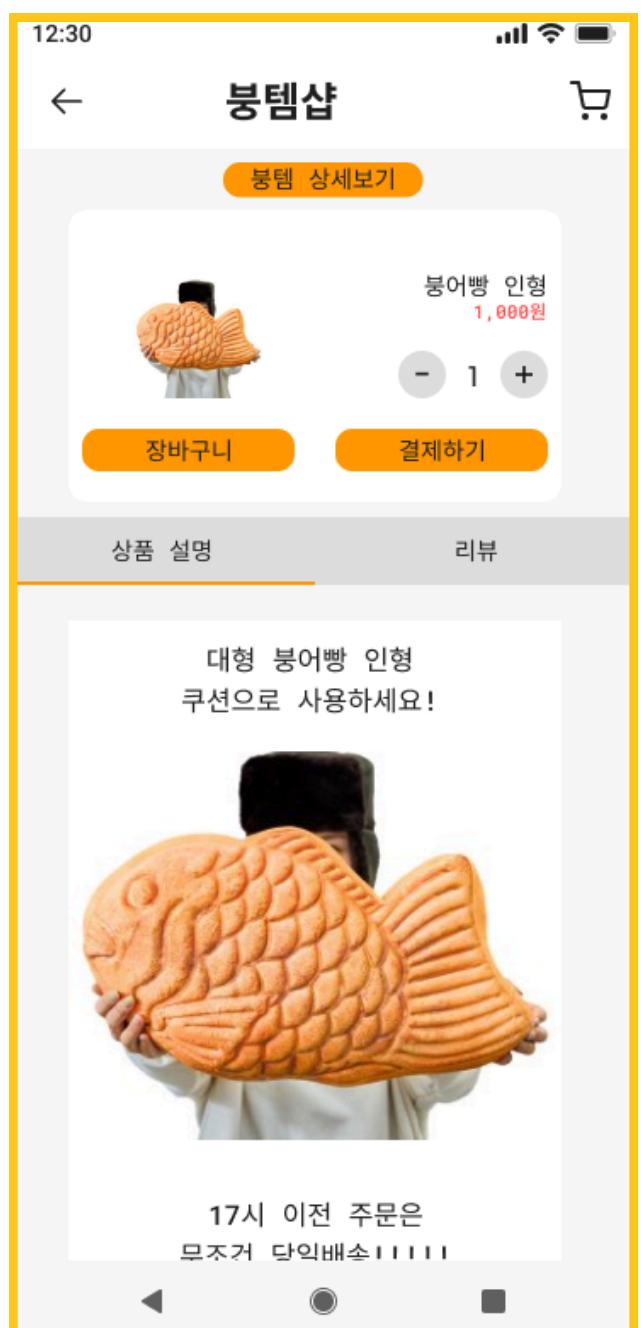
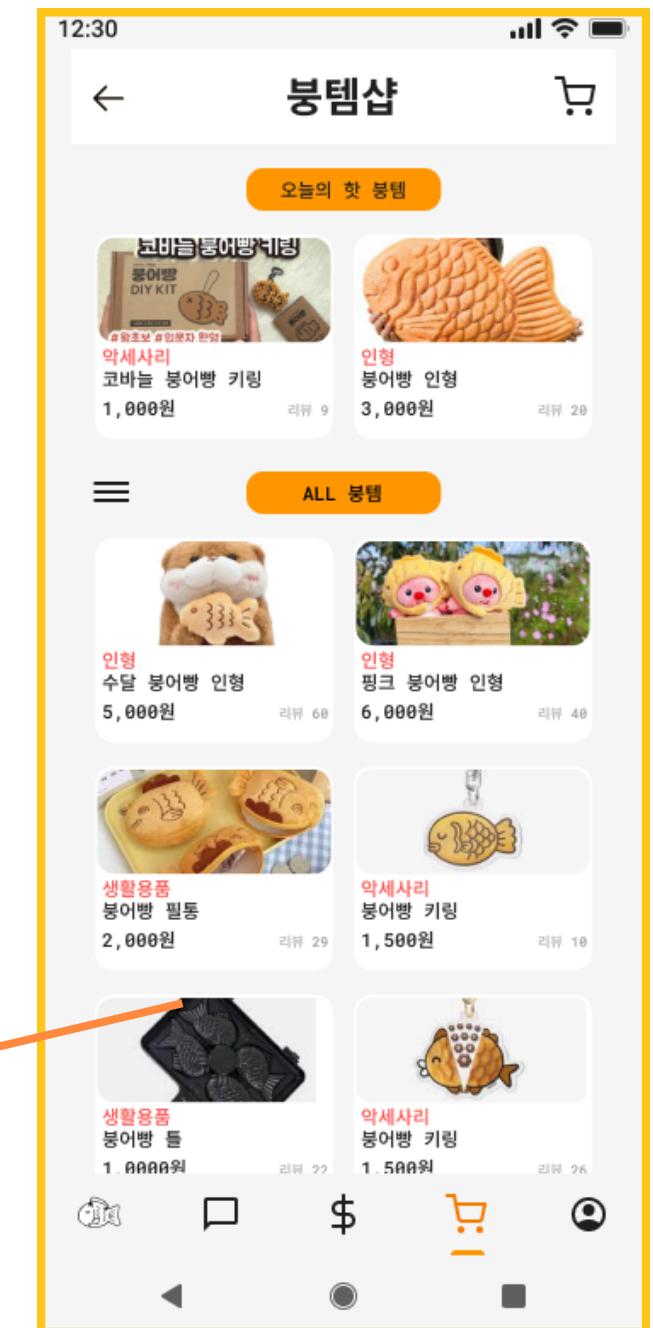
다양한 굿즈

생생한 후기

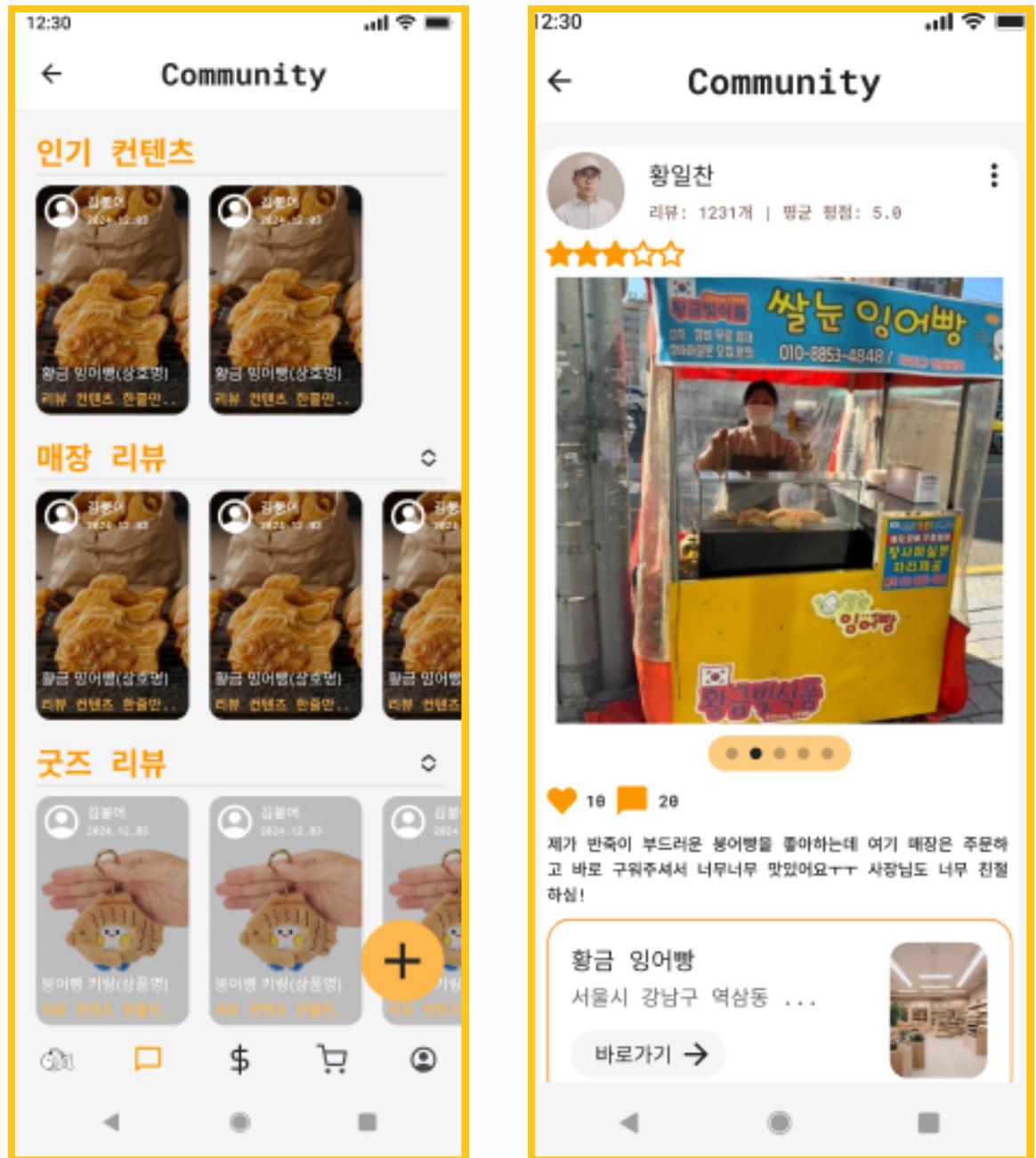
인앱 구매



붕어빵 밀키트



X X X X



3

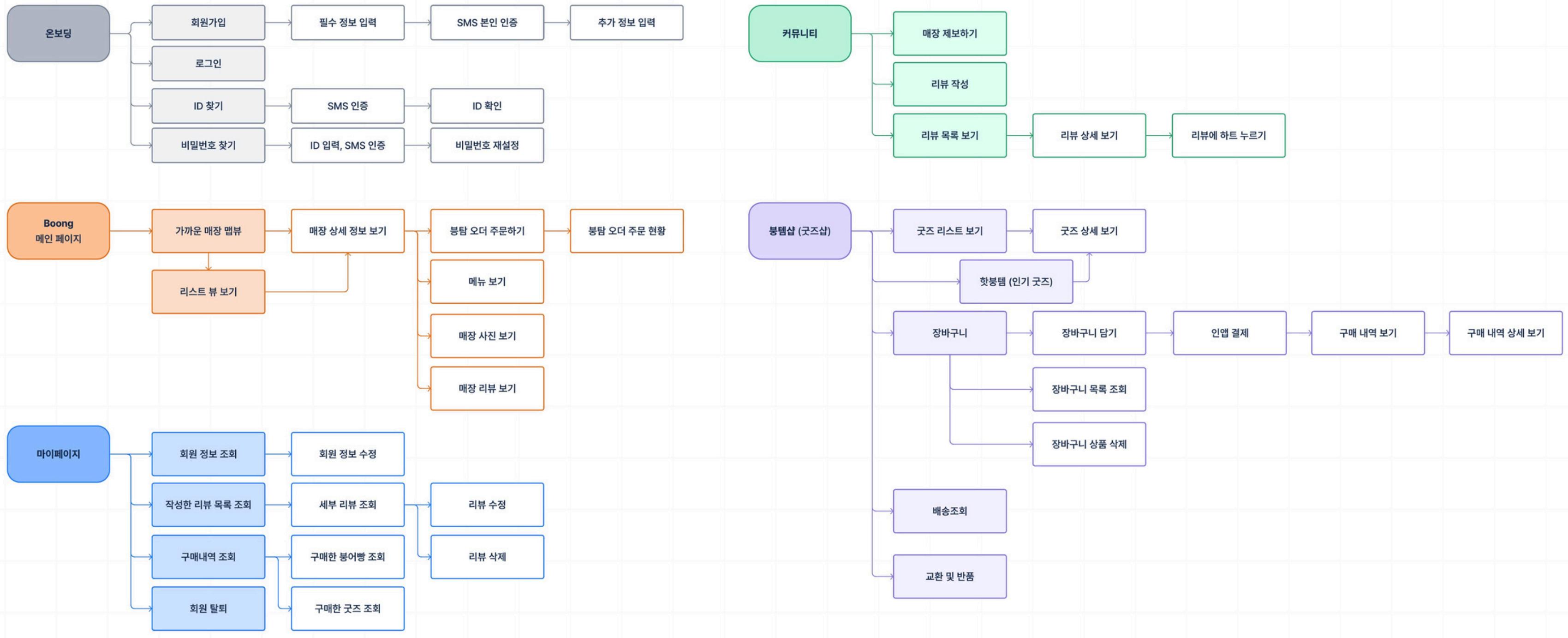


# 커뮤니티 기능 강화

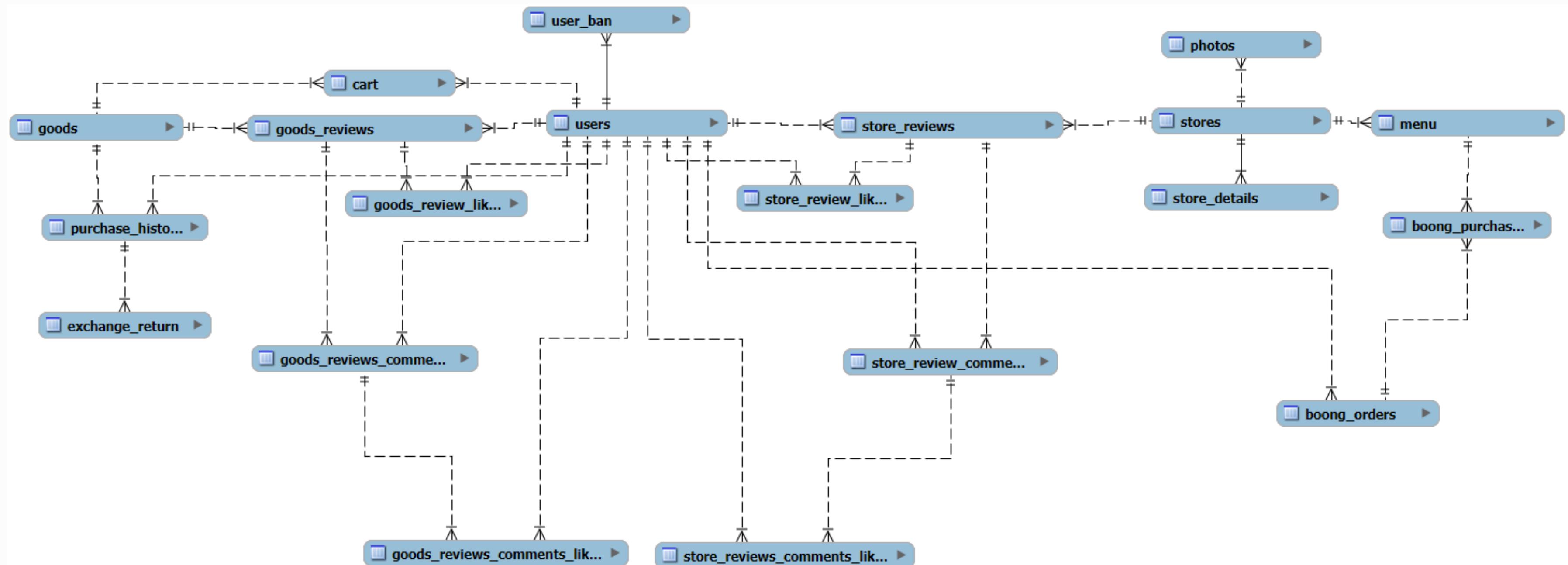




# IA



# DB 테이블 구조





# 붕어탐정 프로젝트에 사용된 협업툴

## 협업툴

소통



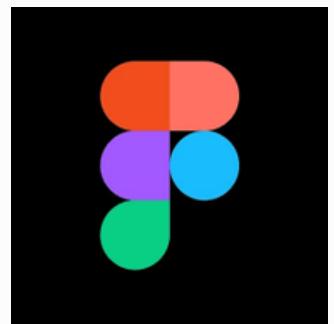
문서 및 일정관리



코드 관리



## 기획 / 디자인



피그마



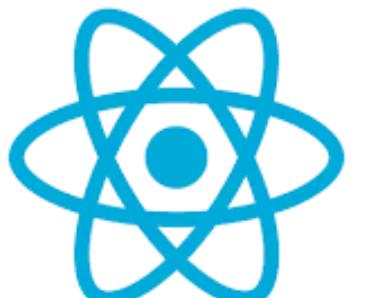
갈릴레이  
AI



× × × ×

# 붕어탐정 프로젝트에 사용된 개발툴

## 프런트



React Native



Google Maps

## 백엔드

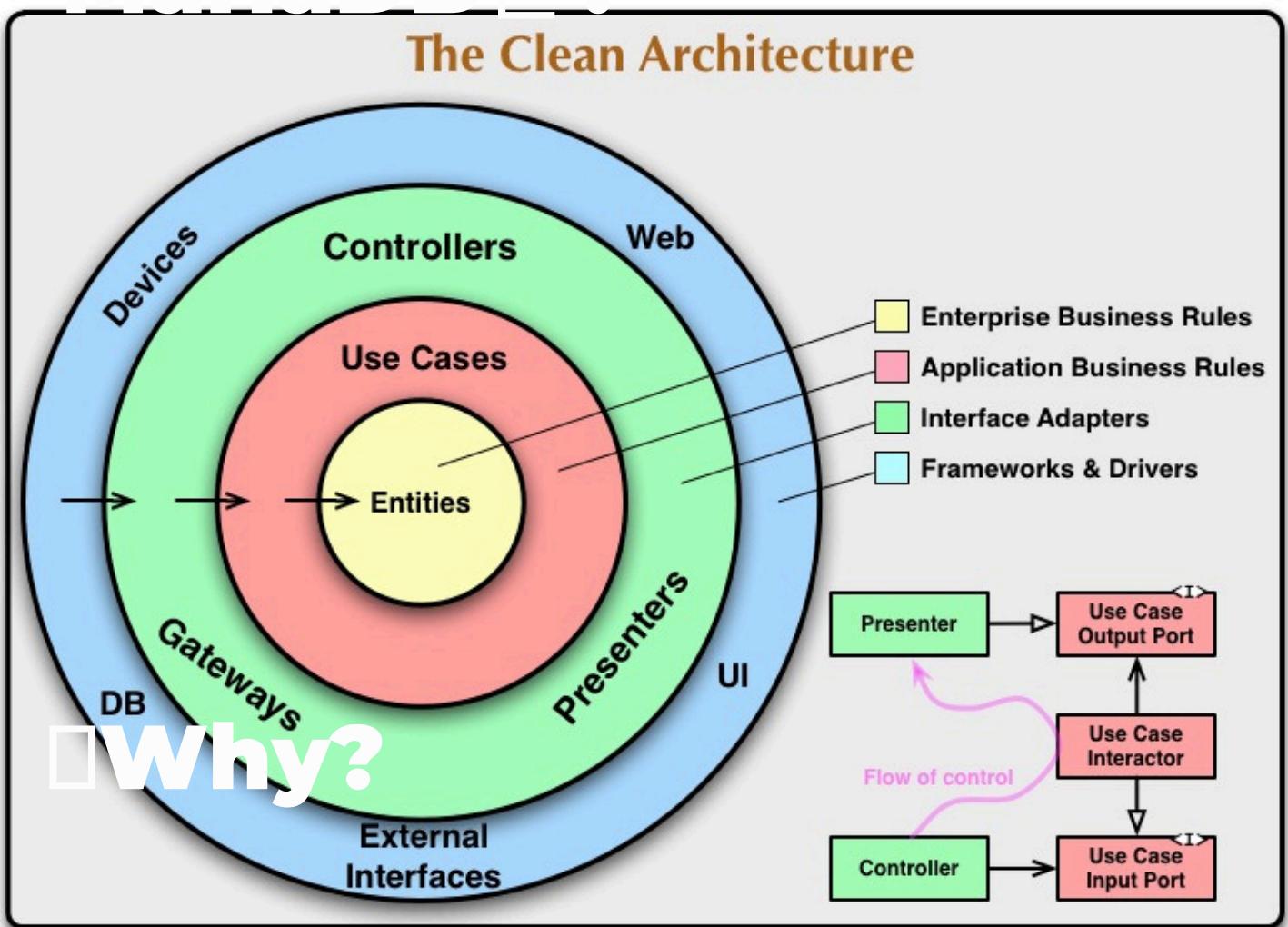


Amazon EC2



amazon  
RDS

# 주요 코드 - 클린아키텍처



```
// service
export const login = async (loginInfo) =>
  await client.post(makeEndPoint('auth/'), loginInfo)
```

```
// Repository
export async function login(id, password) {
  try {
    const loginInfo = { id, password }
    const response = await service.login(loginInfo)
    if (response.code == 200) {
      await AsyncStorage.setItem('token', JSON.stringify(response.token))
    }
    return response
  } catch (error) {
    console.log(error)
  }
}
```

```
// usecase
export const loginUseCase = async (id, password) => {
  return await repository.login(id, password)
}
```



# 주요 코드 - ApiClient

```
class Client {
    async get(endpoint, requestBody) {
        return await fetchData(false, 'GET', endpoint, requestBody)
    }

    async post(endpoint, requestBody) {
        return await fetchData(false, 'POST', endpoint, requestBody)
    }

    async put(endpoint, requestBody) {
        return await fetchData(false, 'PUT', endpoint, requestBody)
    }

    async patch(endpoint, requestBody) {
        return await fetchData(false, 'PATCH', endpoint, requestBody)
    }

    async delete(endpoint, requestBody) {
        return await fetchData(false, 'DELETE', endpoint, requestBody)
    }
}
```

```
async function fetchData(requireToken, method, endpoint, requestBody) {
    try {
        const loadedToken = await AsyncStorage.getItem('token')
        const token = loadedToken ? loadedToken.replace(/"/g, '') : null;
        console.log('token : ', token);
        const options = {
            method,
            headers: {
                'Content-Type': 'application/json',
                'authorization': token,
            },
        }
        console.log('headers : ', options.headers)

        if (requestBody) {
            options.body = JSON.stringify(requestBody)
        }
        const resp = await fetch(BASE_URL + endpoint, options)

        if (!resp.ok) {
            throw new Error(`Client error! status: ${resp.status}`)
        }

        const data = await resp.json()
        return data
    } catch (error) {
        return { code: extractStatusCode(error) }
    }
}
```



# 주요 코드 - ApiClient

```
async function fetchData(requireToken, method, endpoint, requestBody) {  
  
  try {  
    const loadedToken = await AsyncStorage.getItem('token')  
    const token = loadedToken ? loadedToken.replace(/"/g, '') : null;  
    console.log('token : ', token);  
    const options = {  
      method,  
      headers: {  
        'Content-Type': 'application/json',  
        'authorization': token,  
      },  
    }  
    console.log('headers : ', options.headers)  
  
    if (requestBody) {  
      options.body = JSON.stringify(requestBody)  
    }  
    const resp = await fetch(BASE_URL + endpoint, options)  
  
    if (!resp.ok) {  
      throw new Error(`Client error! status: ${resp.status}`)  
    }  
  
    const data = await resp.json()  
    return data  
  } catch (error) {  
    return { code: extractStatusCode(error) }  
  }  
}
```



```
const response = await writeStoreReview(storeinfo);
```



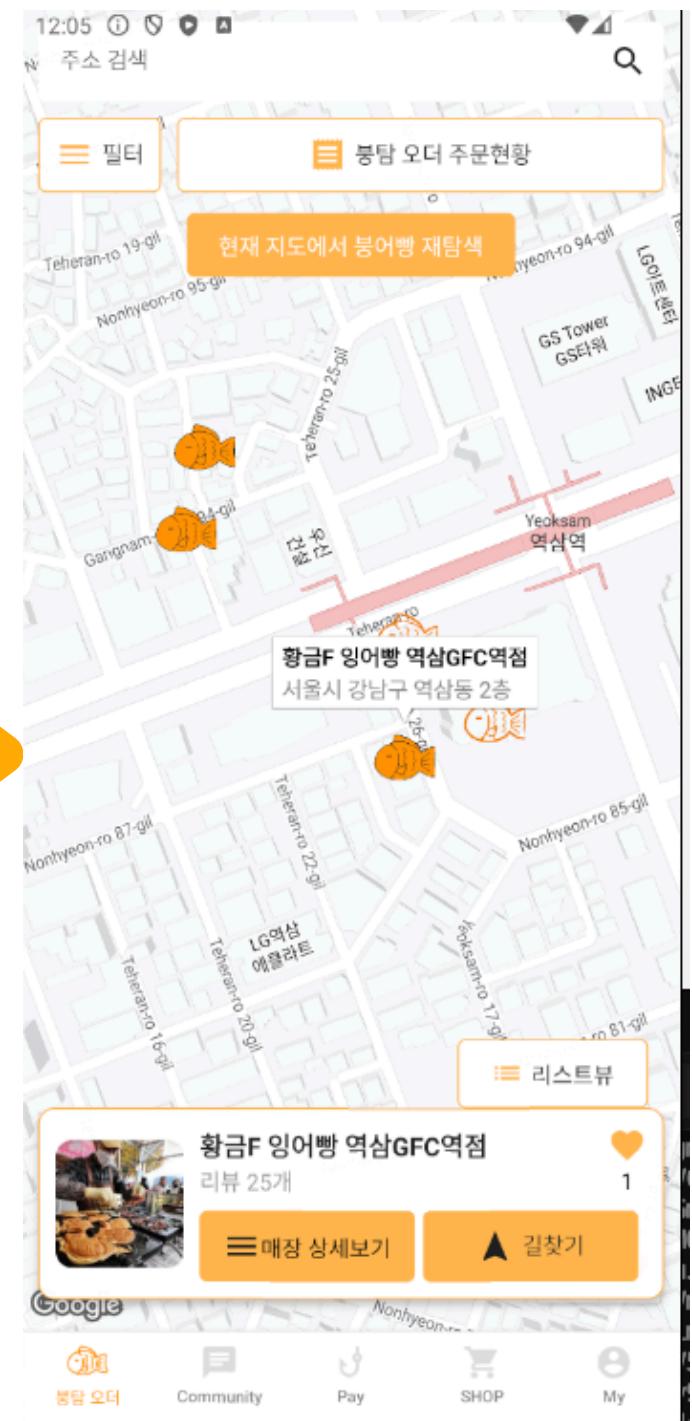
# 주요 코드 - Boong Map Api

```
export const fetchNearbyStores = async (lat, lng, radius = 5, count = 5, sortOrder = 'ORDER BY distance ASC') => {
  try {
    // 반경 필터링(5km) 후 매장 정보 등 가져옴
    const query = `
      SELECT
        s.store_id,
        s.store_name,
        s.address,
        s.latitude,
        s.longitude,
        s.thumbnail_url,
        s.heart_count,
        s.is_order_online,
        (6371 * ACOS(
          COS(RADIANS(?)) * COS(RADIANS(s.latitude)) * COS(RADIANS(s.longitude) - RADIANS(?)) +
          SIN(RADIANS(?)) * SIN(RADIANS(s.latitude))
        )) AS distance,
        COUNT(r.store_review_id) AS review_count
      FROM stores s
      LEFT JOIN store_reviews r ON s.store_id = r.store_id
      GROUP BY s.store_id
      HAVING distance <= ?
      ${sortOrder}
      LIMIT ?;
    `;
  
```

사용자 위치, 매장 위치 기반 거리 계산  
→ 반경 내 매장 필터링  
→ 매장별 리뷰 개수 계산  
→ 가까운 순/인기순으로 정렬

반환 개수는 제한 가능

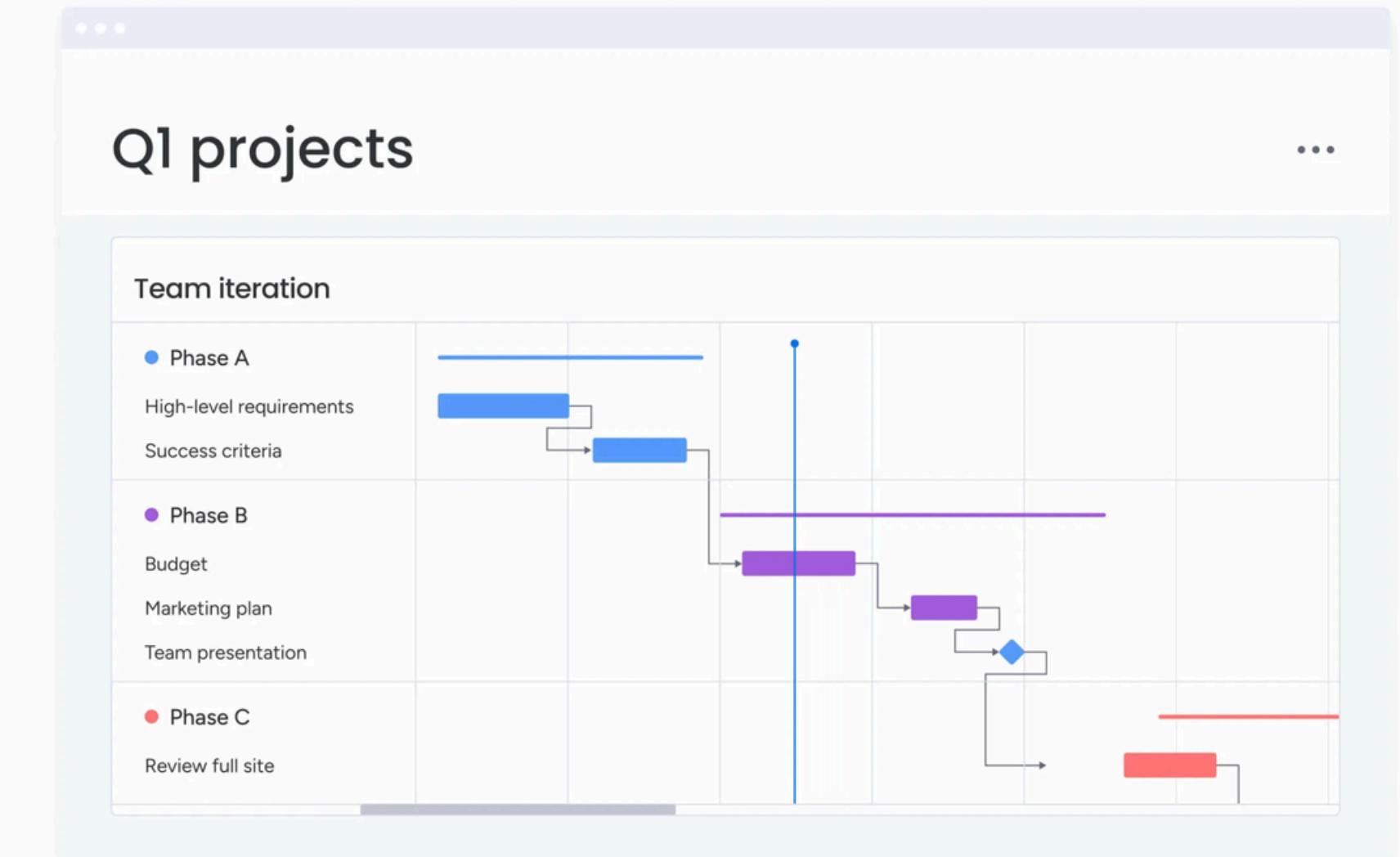
하버사인 공식 (Haversine formula)

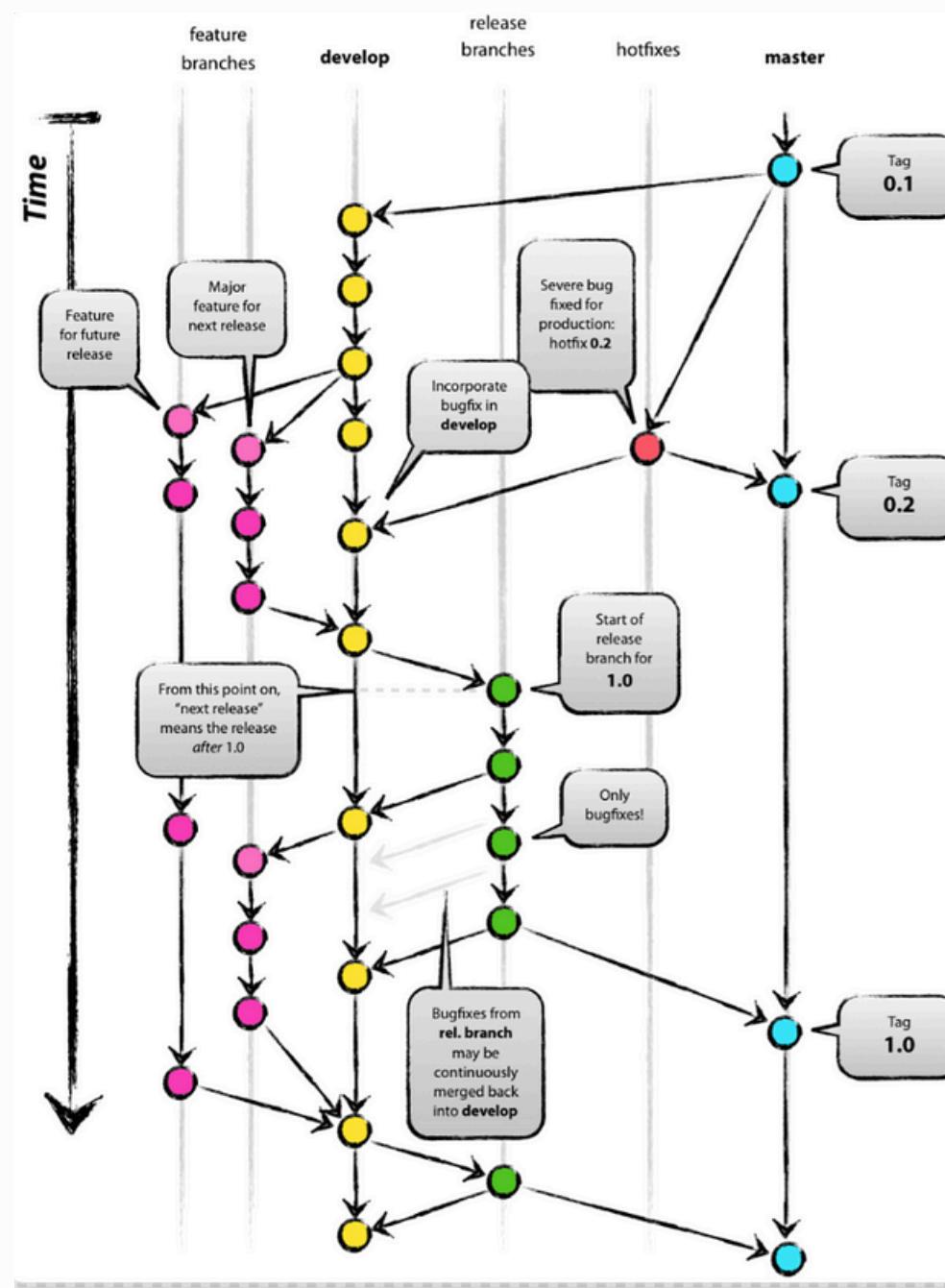


## 03. 앱 시연



# 04. 프로젝트 총평



😊 좋았던 점


[RN] 커뮤니티 API 명세서 작성하기	<input checked="" type="checkbox"/> KAN-43	
판매자 앱 와이어프레임 만들기	<input checked="" type="checkbox"/> KAN-28	
[RN] 커뮤니티 페이지 구현	<input checked="" type="checkbox"/> KAN-105	
매장 리뷰 상세보기 페이지 구현	<input checked="" type="checkbox"/> KAN-117	
구매내역 상세보기	<input checked="" type="checkbox"/> KAN-141	
[RN] 구글 지도 등록	<input checked="" type="checkbox"/> KAN-154	
+ 이슈 만들기		
[RN] 회원가입 api 연결	<input checked="" type="checkbox"/> KAN-109	
[RN] 메인화면(지도뷰) 제작	<input checked="" type="checkbox"/> KAN-108	
[RN] 신고하기 페이지	<input checked="" type="checkbox"/> KAN-118	
[RN] 신고하기 상세 페이지 -굿즈 해보기	<input checked="" type="checkbox"/> KAN-119	
[RN] 커뮤니티 마무리	<input checked="" type="checkbox"/> KAN-160	
구매내역 api	<input checked="" type="checkbox"/> KAN-140	

<input type="checkbox"/> ↗ develop/BT-138: [server] 커뮤니티 - 굿즈 및 매장 리뷰 상세 리스트 조회 api 추가	#32 by pokmonlove was merged last week • Review required	1
<input type="checkbox"/> ↗ Develop bt 135: [server] 리뷰 좋아요 (하트) 토플 기능 추가	#31 by pokmonlove was merged last week • Review required	1
<input type="checkbox"/> ↗ Develop/BT-132: [server] 커뮤니티 - 굿즈 및 매장 리뷰 보기 기능 추가	#30 by pokmonlove was merged 2 weeks ago • Approved	3
<input type="checkbox"/> ↗ Develop/BT-127: [server] 커뮤니티 - 매장 제보 기능 추가	#29 by pokmonlove was merged 2 weeks ago • Review required	2
<input type="checkbox"/> ↗ develop/BT-120: [server] 로그인 및 회원가입 api 수정	#28 by CCiwon was merged last week • Approved	2
<input type="checkbox"/> ↗ develop/BT-124: [server] 커뮤니티-매장 및 굿즈 리뷰 작성 API 추가	#27 by pokmonlove was merged 2 weeks ago • Review required	1
<input type="checkbox"/> ↗ develop/BT-114 : [server] 마이페이지 리뷰 조회, 수정, 삭제	#26 by pokmonlove was merged 2 weeks ago • Review required	2



😊 좋았던 점

Category	Test	연결완료	Name	End Point	Method	Params Type	LINK
AUTH	✓		회원가입 API	BASE_URL/auth/sign	POST	BODY	<a href="#">new 회원가입 API</a>
	✓		로그인 API	BASE_URL/auth	POST	BODY	<a href="#">new 로그인 API</a>
	✓		닉네임 중복 확인 API	BASE_URL/auth/check-user-id	POST	BODY	<a href="#">아이디 중복확인 API</a>
	✓		ID 찾기	BASE_URL/auth/find-ID	POST	BODY	<a href="#">ID 찾기 API</a>
	✓		PW 찾기	BASE_URL/auth/find-ID	POST	BODY	<a href="#">비밀번호 찾기 API</a>
VERIFICATION	✓		인증번호 전송 api	BASE_URL/verification/send-code	POST	BODY	<a href="#">SMS 인증 API 명세서</a>
	✓		인증번호 검증 api	BASE_URL/verification/verify-code	POST	BODY	<a href="#">SMS 인증 API 명세서</a>
ROUTE	Test	연결완료	Name	End Point	Method	Params Type	LINK
COMMUNITY	✓	✓	굿즈 리뷰 리스트 API	BASE_URL/community/gre	GET	QUERY	<a href="#">굿즈 리뷰 리스트 API</a>
COMMUNITY	✓		리뷰 좋아요 토클 API	BASE_URL/community/re/like	PATCH	BODY	<a href="#">리뷰 좋아요 토클 API</a>
COMMUNITY	✓		매장 리뷰 상세 리스트 API	BASE_URL/community/sr/detail	GET	QUERY	<a href="#">매장 리뷰 상세 리스트 API</a>
COMMUNITY	✓		굿즈 리뷰 상세 리스트 API	BASE_URL/community/gd/detail	GET	QUERY	<a href="#">굿즈 리뷰 상세 리스트 API</a>
SHOP	✓	✓	카테고리별 굿즈 리스트 API	BASE_URL/goods	GET	QUERY	<a href="#">굿즈 리스트 API</a>
SHOP	✓	✓	봉투샵 - 핫템샵	BASE_URL/goods/hot-items	GET	QUERY	<a href="#">봉투샵 리스트 API</a>
SHOP	✓		봉투샵 - 장바구니	BASE_URL/goods/cart	GET	QUERY	<a href="#">굿즈샵 장바구니 API</a>
SHOP	✓		봉투샵 - 결제	BASE_URL/goods/checkout	POST	BODY	<a href="#">굿즈샵 결제 API</a>
SHOP	✓		봉투샵 - 구매내역	BASE_URL/goods/purchase_history	GET	QUERY	<a href="#">굿즈샵 구매내역 API</a>
SHOP	✓		봉투샵 - 구매내역상세보기	BASE_URL/goods/purchase_history/{purchase_id}	GET	QUERY	<a href="#">굿즈샵 구매내역 상세보기 API</a>
SHOP	✓		봉투샵 - 굿즈상세보기	BASE_URL/goods/{goods_id}	GET	QUERY	<a href="#">굿즈 상세보기 API</a>
SHOP	✓		봉투샵 - 배송조회	BASE_URL/goods/post	GET	QUERY	<a href="#">배송조회 API</a>
SHOP	✓		봉투샵 - 교환	BASE_URL/goods/exchange	GET POST	BODY	<a href="#">교환 신청 페이지</a>
SHOP	✓		봉투샵 - 반품	BASE_URL/goods/return	GET POST	BODY	<a href="#">반품 신청 페이지</a>
USER	✓		마이페이지-회원정보	BASE_URL/user/info	GET PATCH	BODY	<a href="#">회원 정보 조회 및 수정 API</a>

API 명세서 표

### 매장 리뷰 리스트 API

소유자: 이종민 ...  
김남희(가) 최근 전에 마지막 업데이트. 본 사용자의 수 보기

- Endpoint: BASE\_URL/community/sre
- Method: GET
- Authorized Token: Required
- 커뮤니티탭 메인페이지 인기 컨텐츠, 매장 리뷰

**요청(Request)**

- 예시 : {{base}}community/sre?sort=popular&count=5
- sre : store review

이름	타입	
sort	QUERY	• popular : 인기순 • latest : 최신순(default)
count	QUERY	• 5(default value)

**응답(response)**

store\_review\_info

이름	타입	설명
store_review_id	number	
store_id	number	

### new 회원가입 API

소유자: 김남희 ...  
조금 전에 마지막 업데이트. 본 사용자의 수 보기

- Endpoint: BASE\_URL/auth/sign
- Method: POST
- Authorized Token: Not Required
- Request Body
  - (필수만 적은 것)

```

1 {
2   "phone": "01020314444",
3   "id": "nam111",
4   "password": "11111111"
5 }
```

store\_review\_info

이름	타입	설명	
phone	BODY	전화번호	NOT NULL
id	BODY	닉네임	NOT NULL
password	BODY	비밀번호	NOT NULL
address1	BODY	주소	NULLABLE
address2	BODY	상세 주소	NULLABLE
zipcode	BODY	도로명주소	NULLABLE // 1월 21일 추가
email	BODY	이메일	NULLABLE

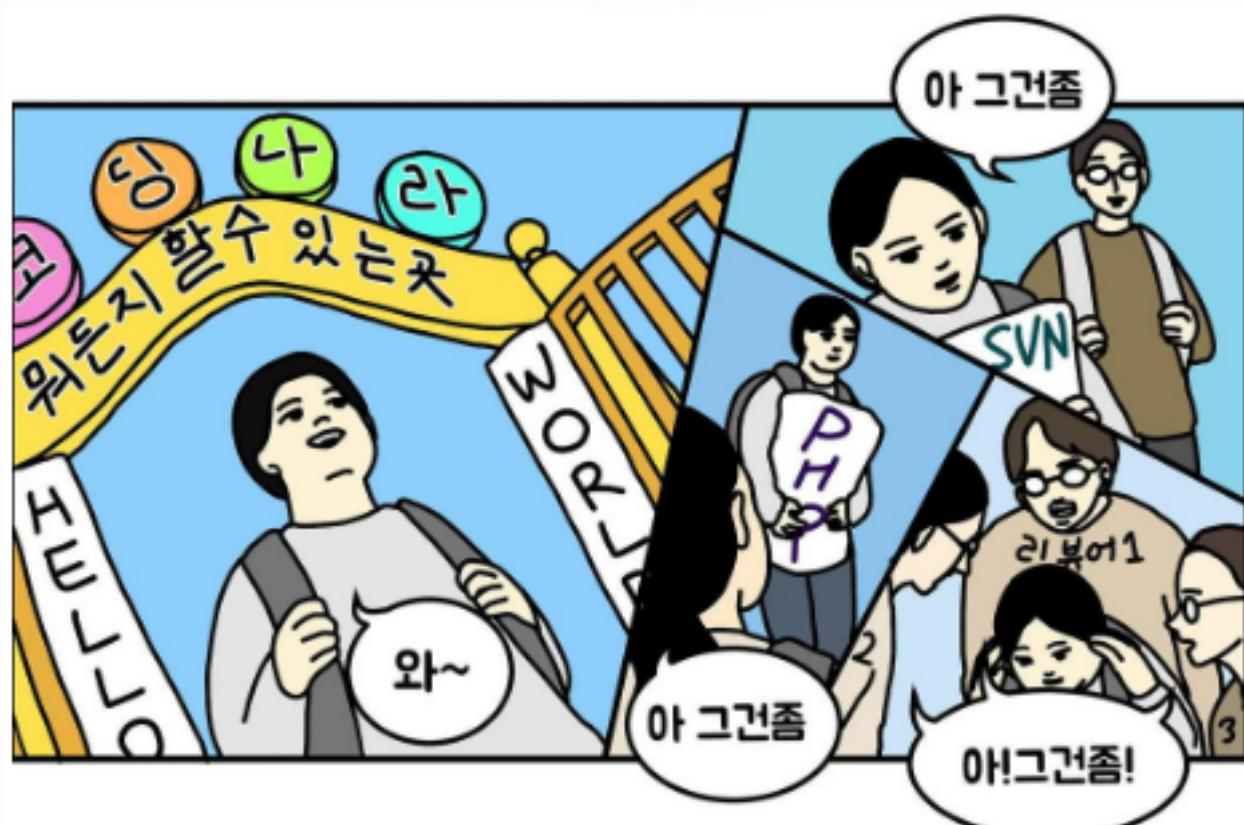
API 명세서

# 황일찬

# 프로젝트 하면서 느낀 점

## 좋았던 점

- 시야를 넓힐 수 있는 기회가 되었다.



- leeJongMin58 on Dec 16, 2024 Member ...  
앞으로 constants에 있는 color와 Typography를 사용하기 위해서는 이러한 StyleSheet가 모든 뷰에 필요한거죠??
- ilchan123 on Dec 16, 2024 Member Author ...  
넵.  
Typo같은 여러 형식이 지정된 글꼴의 경우.  
const styles = StyleSheet.create를 사용해야 [...] 스프레드 연산자로 적용해서 사용할 수 있습니다.
- leeJongMin58 on Dec 16, 2024 Member ...  
볼드 적용도 안되어 있네용
- ilchan123 on Dec 16, 2024 Member Author ...  
그러면 export로 따로 빼서 사용할까요  
bold를 하려고 했지만, 문자로 bold를 따로 지정할 경우 숫자랑 다르게, 별도의 설치 과정이 필요해서 숫자로 적용했습니다.  
bold 타입과 normal에 대한 수치가 정의되어 있어서 700이 bold라고 보시면 됩니다.  
아 저 부분은 700이 아니라 500이라고 되어있네요. 수정완료했습니다
- leeJongMin58 on Dec 16, 2024 Member ...  
LineHeight이 150이 아니라 150%예요 전체 :)
- ilchan123 on Dec 16, 2024 Member ...  
설치 하시면 됩니다 ㅎㅎ
- 이게 타입 에러라고 떠서 % 적용이 안됩니다. 이걸 전해드린다는 게 까먹었습니다.  
정확한  
[150%와 같은 퍼센트 표현은 React Native의 lineHeight에서 지원되지 않습니다.]입니다.
- leeJongMin58 on Dec 17, 2024 Member ...  
fontsize \* 1.5로 하시면 적용 됩니다  
예를 들어 이런식으로요  
fontFamily: 'Roboto Mono',  
fontSize: 57,  
fontWeight: '500',  
lineHeight: 57 \* 1.5

박치호

## 프로젝트 하면서 느낀 점



- 성취감

- 새로운 것에 대한 학습

- 팀워크



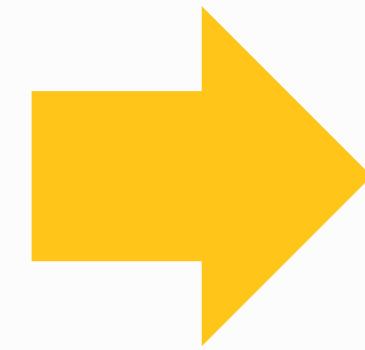
- 나태해진 나의 모습





- 하고자 하면 할 수 있고, 안 되는 건 없다는 것을 느낀 프로젝트!
- 모르는 게 많으면 자랑이 아니라, 팀에 민폐 그 자체다!
- 피할 수 없다면 즐겨야 하고, 즐길 수 없다면 미쳐야 한다!
- 취준생에게 주말은 사치 그 자체이다!
- 마지막까지 API 연결을 기다려준 팀원들에게 너무 미안하고, 너무 감사합니다. 😍





**Thank You**  
**For Watching**