

문고 한 벌로 가!

한 벌의 코드로 모바일 웹/모바일 앱/PC 웹 서비스 확장 개발기

윤영제

NAVER 플레이스서비스개발

CONTENTS

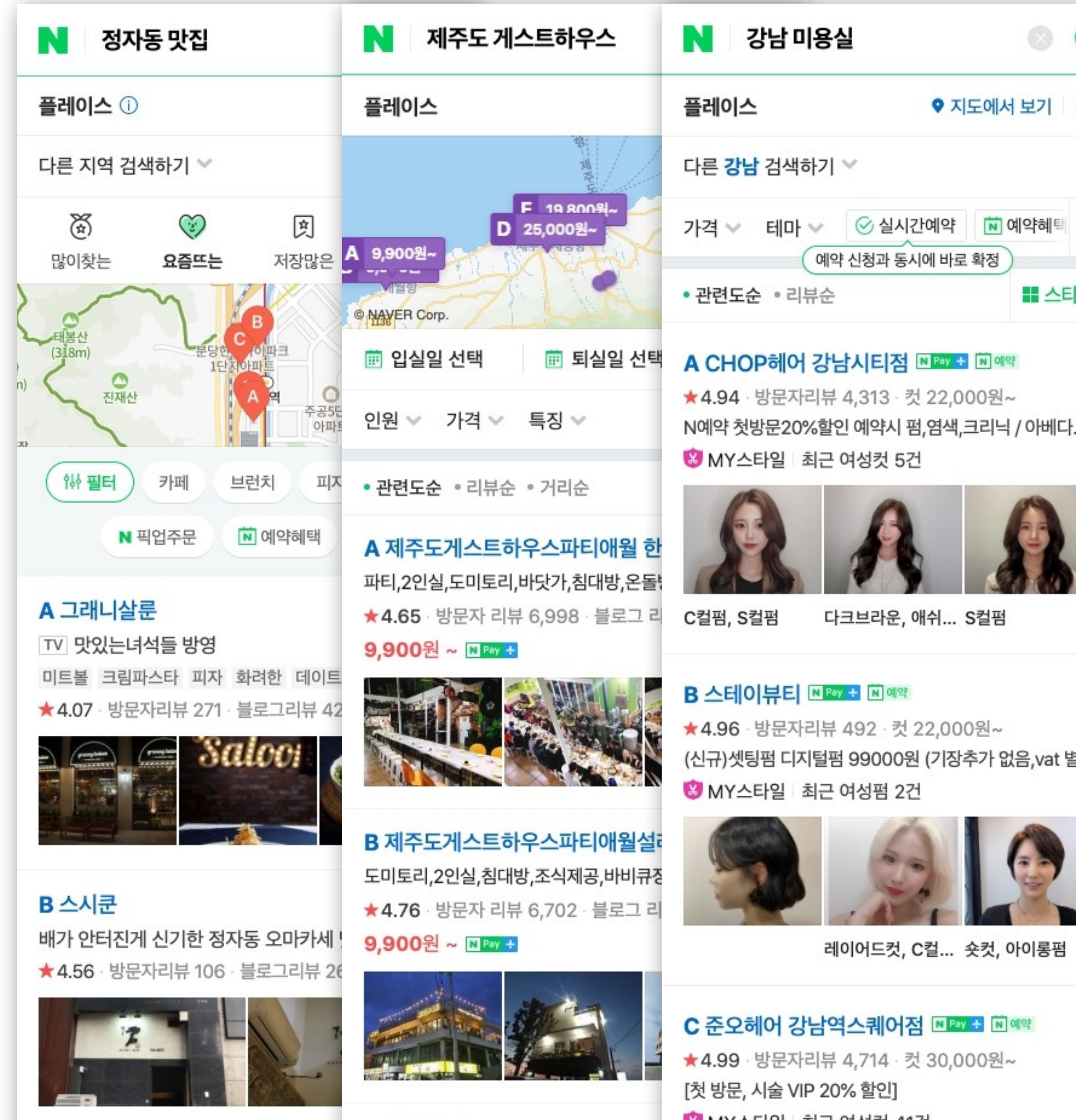
1. 왜 코드 한 벌을 여러 서비스에 넣었을까?
2. 지도앱의 장소 검색 결과를 Native에서 Webview로 변경하기
3. PC지도 웹(Angular)의 장소 검색 결과를 <iframe>(React)으로 집어넣기
4. 한 벌의 코드로 여러 서비스 제공은 어떻게 할까?
5. 성능 개선
6. 앞으로 해야할 일들

1. 왜 코드 한 벌을 여러 서비스에 넣었을까?

1.1 플레이스와 지도

플레이스랑 지도랑 다른 서비스였어?

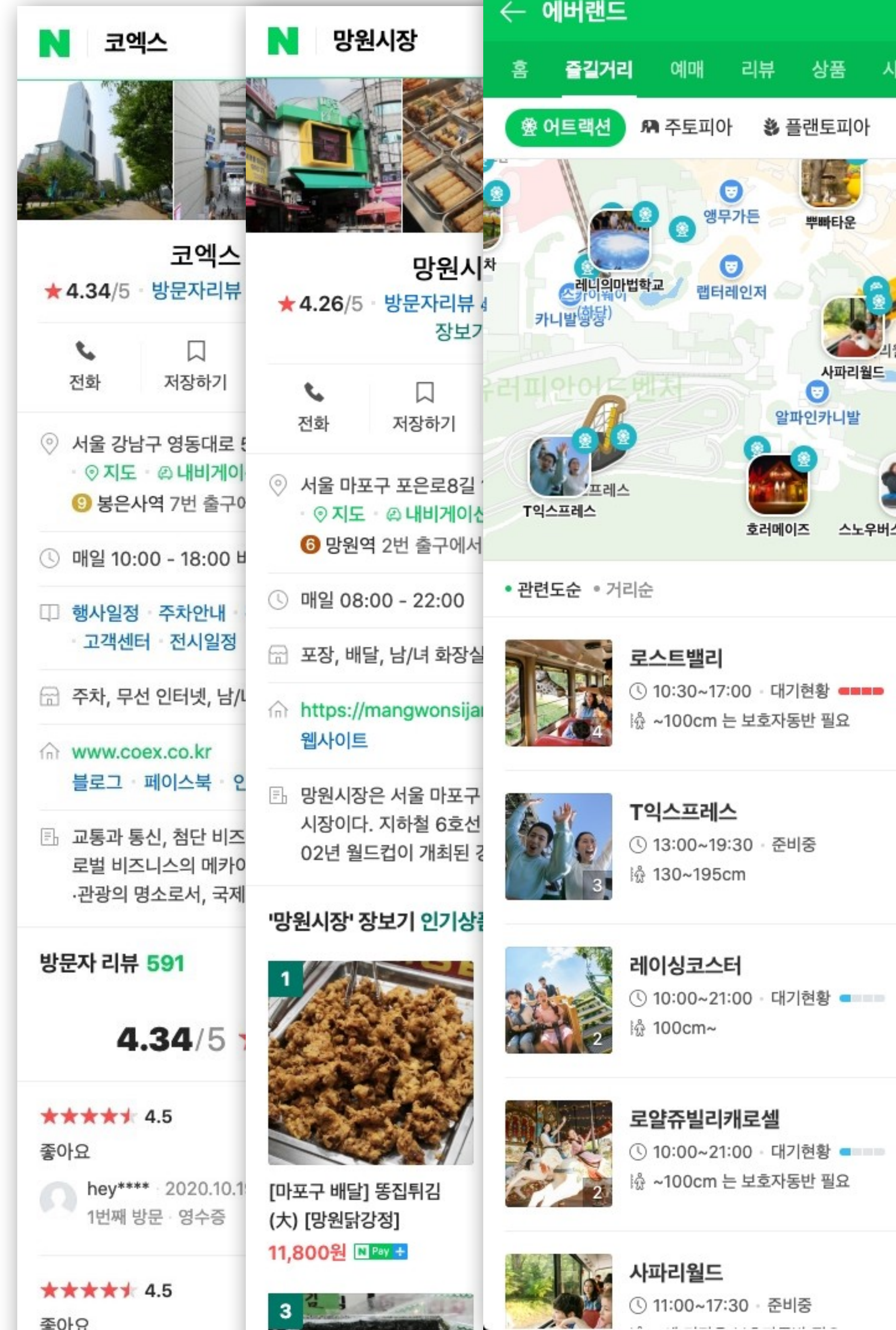
- 플레이스와 지도는 별개의 서비스
- 검색에서 업종별로 특화 화면을 잘 보여주면 어떨까
- 👉 플레이스 서비스의 시작
- 식당, 숙박, 미용실, 명소...



1.2 플레이스/지도 통합의 배경

플레이스 서비스의 성장

- NAVER 통합 검색의 15% 결과를 담당 (모바일 기준)
- 업종별 특화 세분화 & 늘어나는(늘어 날) 특화 업종



1.2 플레이스/지도 통합의 배경

N

동해 해수욕장

플레이스

분당구 정자3동



• 관련도순 • 리뷰순 • 거리순

망상해수욕장

해수욕장, 해변

오토캠핑 해변카페 카라반 깨끗한

176km 블로그리뷰 1,241

어달해수욕장

해수욕장, 해변

해변카페 드라이브 대게빵 깨끗한

178km 블로그리뷰 244

추암해수욕장

해수욕장, 해변

해변 바다열차 일출 깨끗한 경치

181km 블로그리뷰 1,041

한섬해수욕장

해수욕장, 해변

해변 섬 일출 아담한 촬영지

178km 블로그리뷰 155

N

우리은행

플레이스

분당구 정자3동



• 관련도순 • 리뷰순 • 거리순

우리은행에서 제공하는 실시간 대기현황

지점은 우리은행 앱(APP)을 통해 모바일

능합니다. (제공시간: 평일 09시~16시)

우리은행365 은행

690m | 경기 성남시 분당구 정자동

길찾기 저장 공유

우리은행 분당정자지점 은행

1.3km | 경기 성남시 분당구

평일 09:00~16:00

대기현황 대출상당 0명

전화 길찾기 저장

우리은행 정자역지점 은행

1.4km | 경기 성남시 분당구

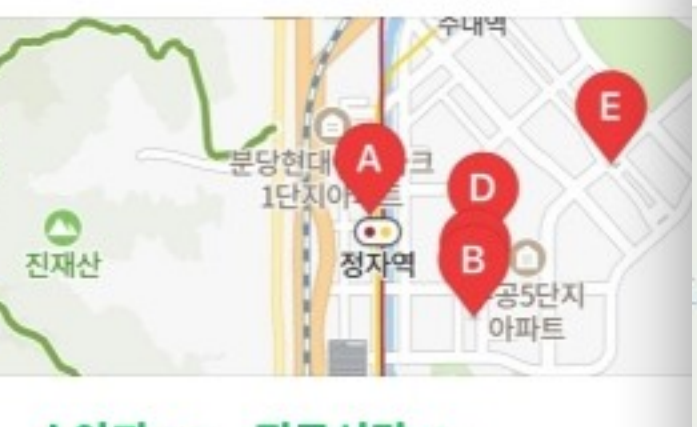
평일 09:00~16:00

N

소아청소년과

플레이스

분당구 정자3동



소아과 진료시간

전체	야간진료
일요일진료	공휴일진료

• 관련도순 • 거리순

A 21세기연세의원 내과

평일 08:30~18:00 13:00~14:00 점심시

08:30~14:00 | 일요일 휴무 | 공휴일 휴무

1.3km 블로그.카페 4 방문자리뷰 6

B 늘해랑소아청소년과의원

소아청소년과

평일 09:00~18:00 | 수요일

09:00~13:00 | 토요일 09:00~13:00

460m 블로그.카페 11 방문자리뷰

C 황혜영소아과의원 소아청소년과

평일 09:00~18:00 점심시간:1-2시 | 화

N

주유소

플레이스

분당구 정자3동



휘발유, 경유 반경2km

전체

GS칼텍스

현대오일

• 가격순 • 거리순

최저 휘 1

현대오일뱅크 제이제이네트웍스 주유소

1.7km 휘 최저1,328 경 최저1,11

S-OIL 래디우스랩 구도일주유소

1.8km 휘 1,358 경 1,158

현대오일뱅크 직영 수내동주유소

1.1km 휘 1,439 경 1,259

GS칼텍스 소모에너지엔테크놀러지 주유소

N

레이어드컷

플레이스

분당구 정자3동

여성 레이어드컷

• 관련도순 • 리뷰순 • 최신순

머리하면? 인영 실장!

시술 34건 리뷰 31

머리하면? 인영 실장!

시술 34건 리뷰 31

N

약국

플레이스

성남시 분당구 서현동

내위치 변경



마스크 보유 현황

실시간 마스크 재고는 업데이트 상황에 따라 실제와 다를 수 있습니다.

A 맘스케어약국

526m | 분당구 서현동 124-5

평일 09:30~21:00 · 주말 09:30~21:00

마스크 수량 100+ 5분전

전화 길찾기 내비게이션

B 서현종로약국

1.2km | 분당구 서현동 23-12

평일 09:30~21:00 · 주말 09:30~21:00

마스크 수량 30~100 5분전

전화 길찾기 내비게이션

C 동암약국

2.6km | 분당구 서현동 225-3

1.2 플레이스/지도 통합의 배경

NAVER 검색 결과와 지도 검색 결과가 왜 달라요?



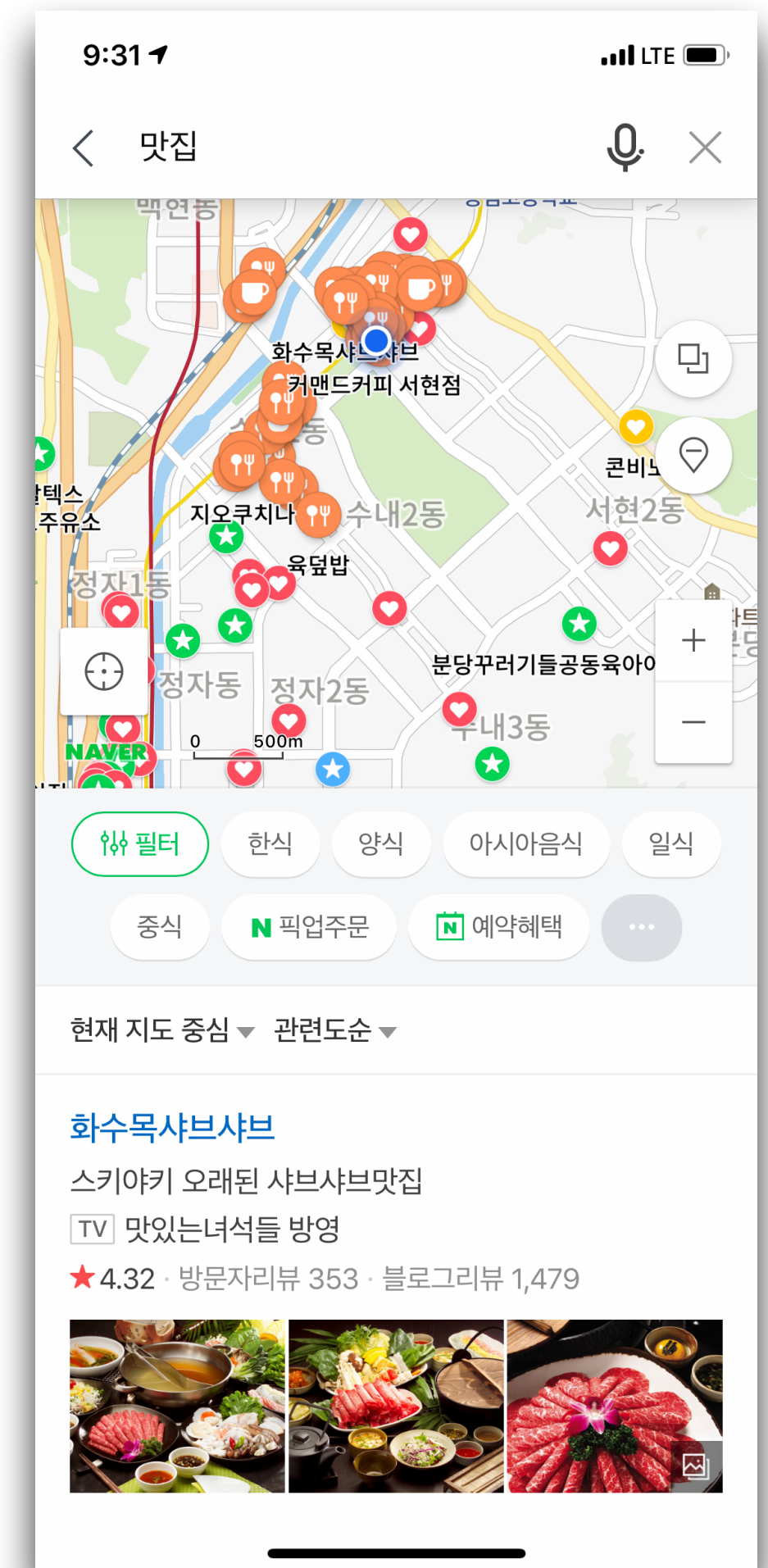
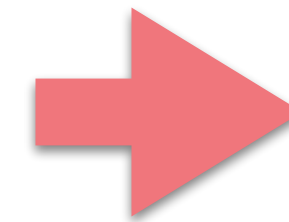
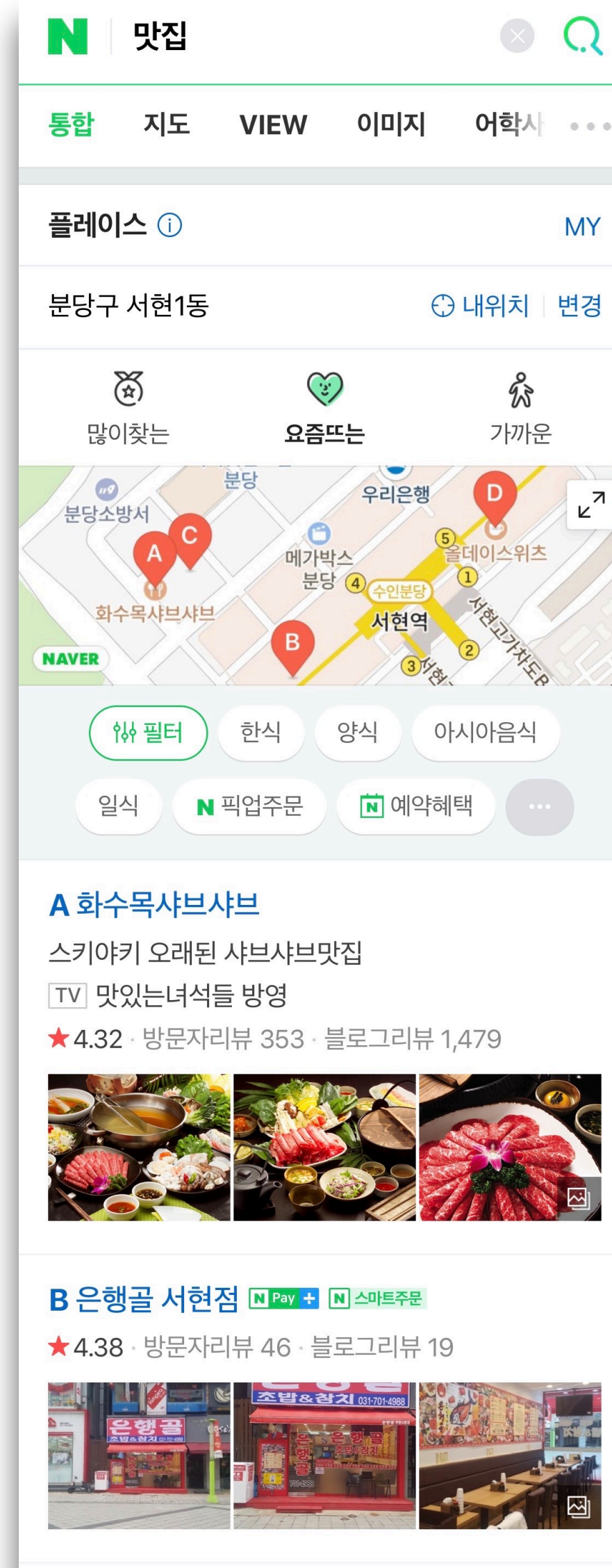
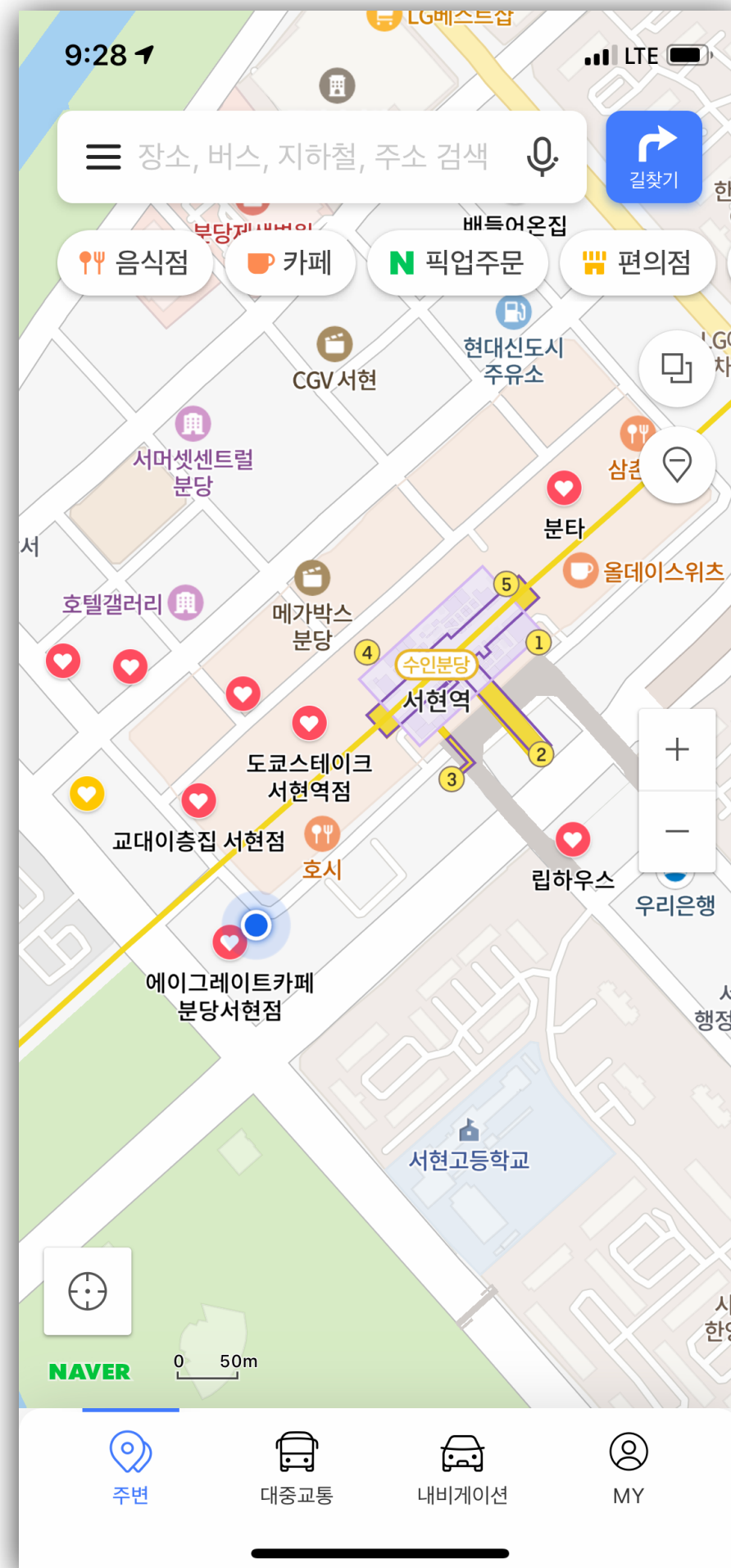
출처: 영화 <닥터 스트레인지> 2016

1.3 어떻게 통합할까?

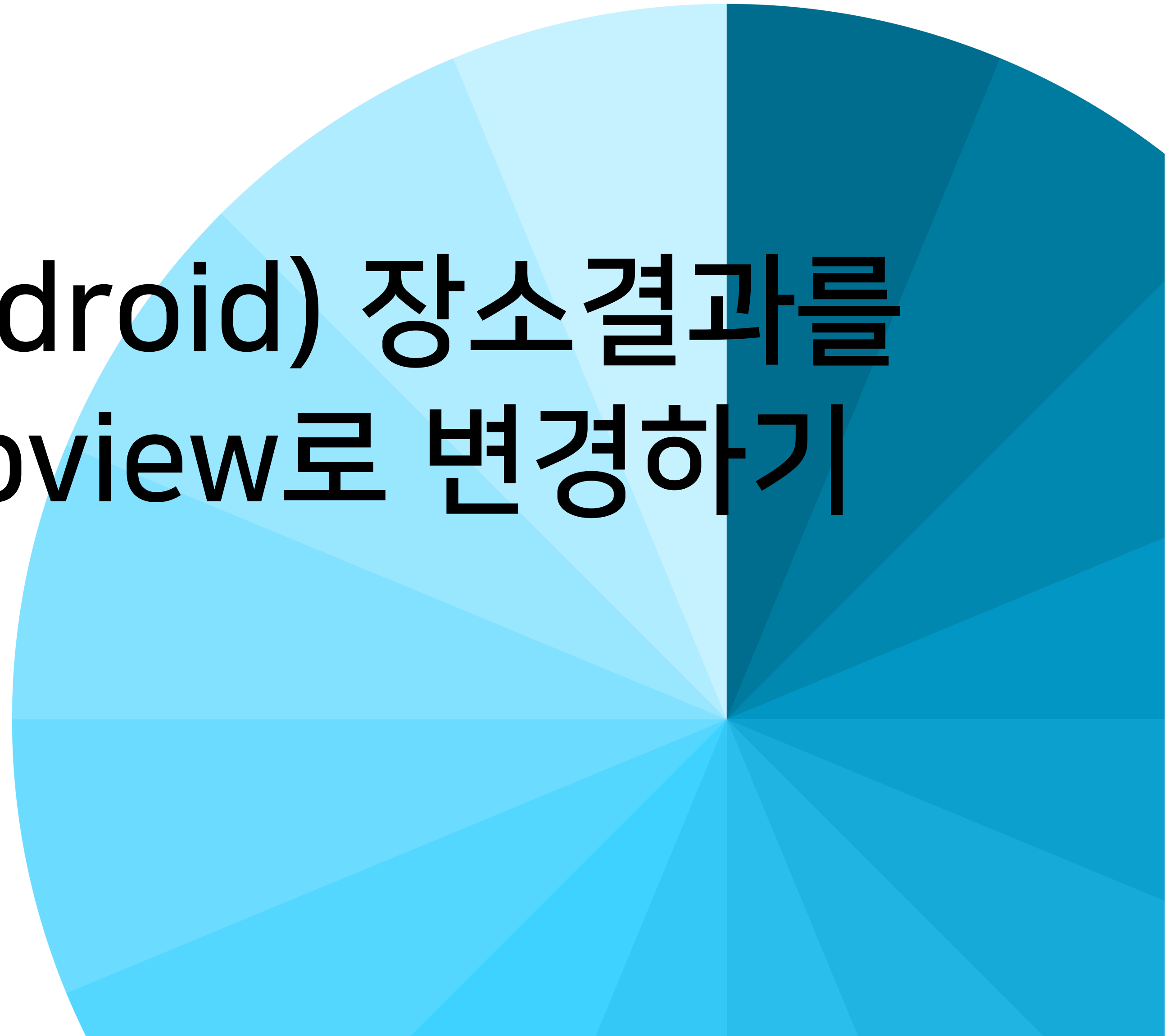
가장 빠르고 효율적인 방법이 뭐야?

- 중복 개발은 피했으면 합니다. 🙏
- 코드 재사용할 수 있도록 기존 화면을 그.대.로. 집어 넣으면 어떨까요?
- 지도앱에 Webview 형태로 플레이스 서비스를 넣으면 되지 않을까요?

1.3 어떻게 통합할까?



2. 지도앱(iOS/Android) 장소결과를 Native에서 Webview로 변경하기



2.1 어느 영역을 Webview로 전환하지?

설계 조건

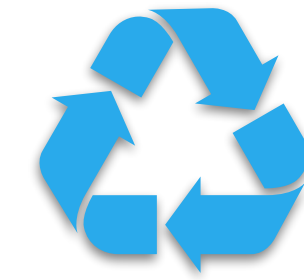
- 사용자는 어색함을 느끼지 않아야 한다.
- 플레이스는 가급적 독립적인 영역으로 고립시켜야 한다.
- Native와 Webview의 장점을 잘 살릴 수 있어야 한다.



2.2 Webview 설계

장소 검색 결과는 플레이스 Webview가 담당한다.

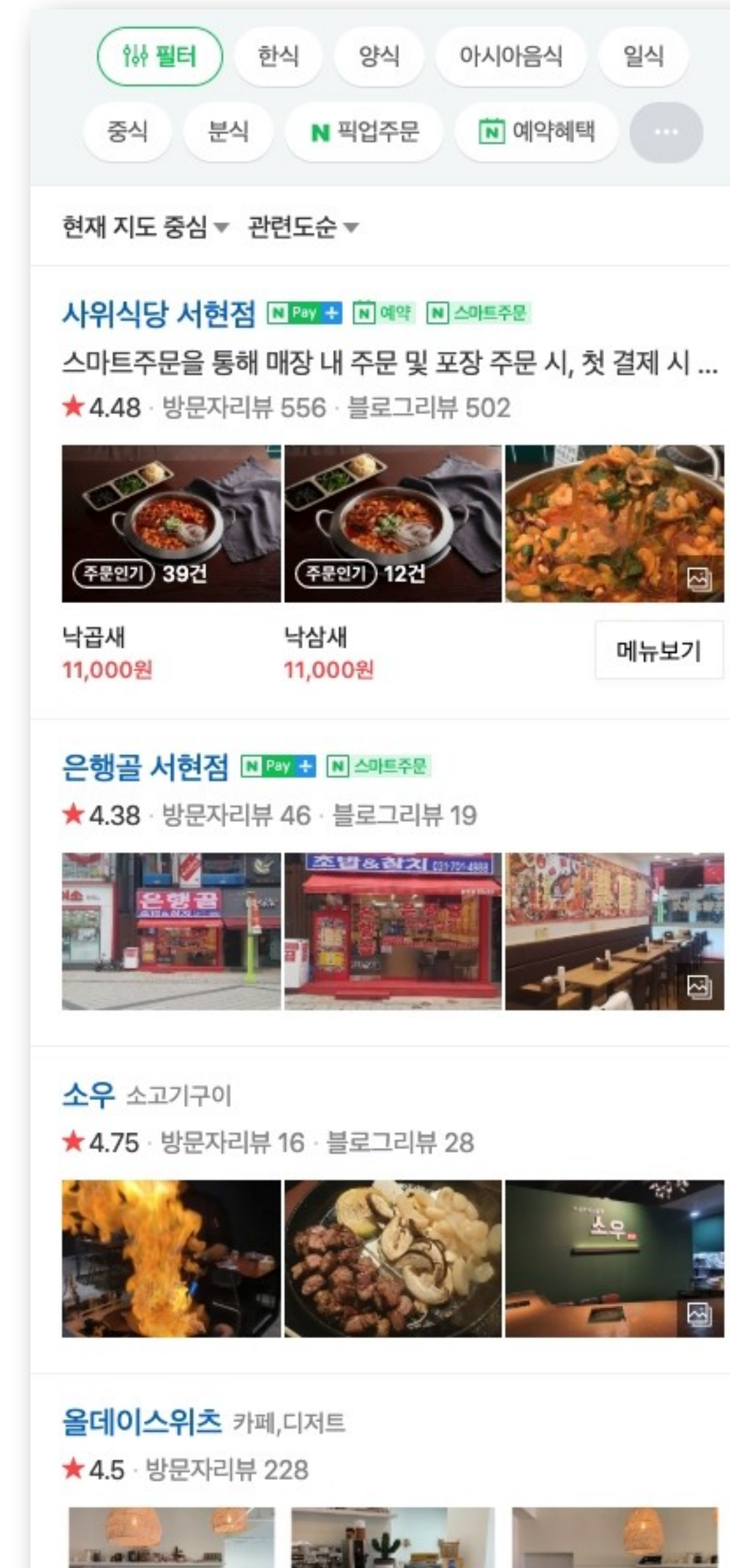
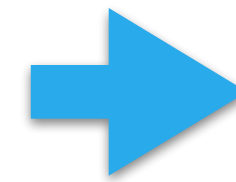
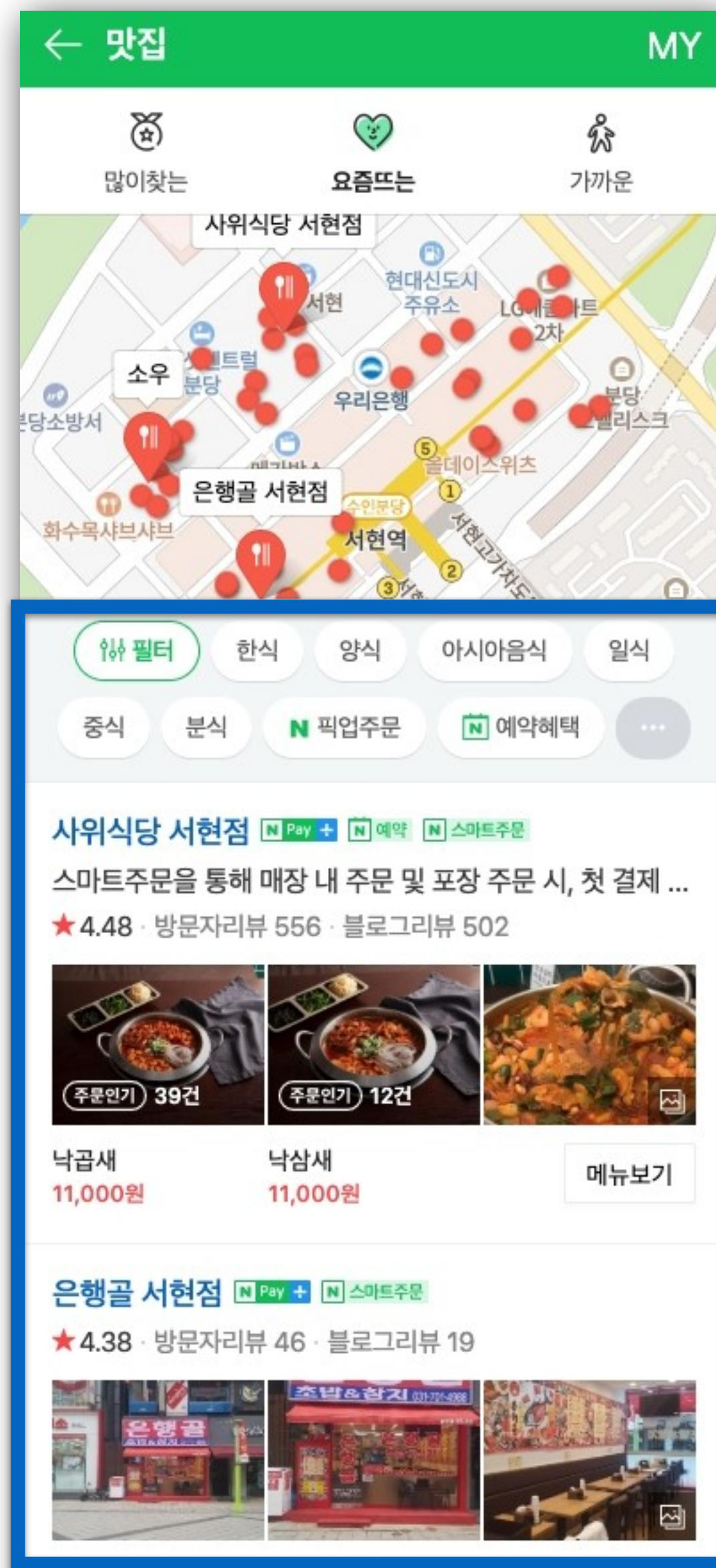
- 플레이스 목록 페이지 & 업체 페이지에서 가져오자!



- 지도앱 특화 기능들은 분기 처리로 대응한다. 

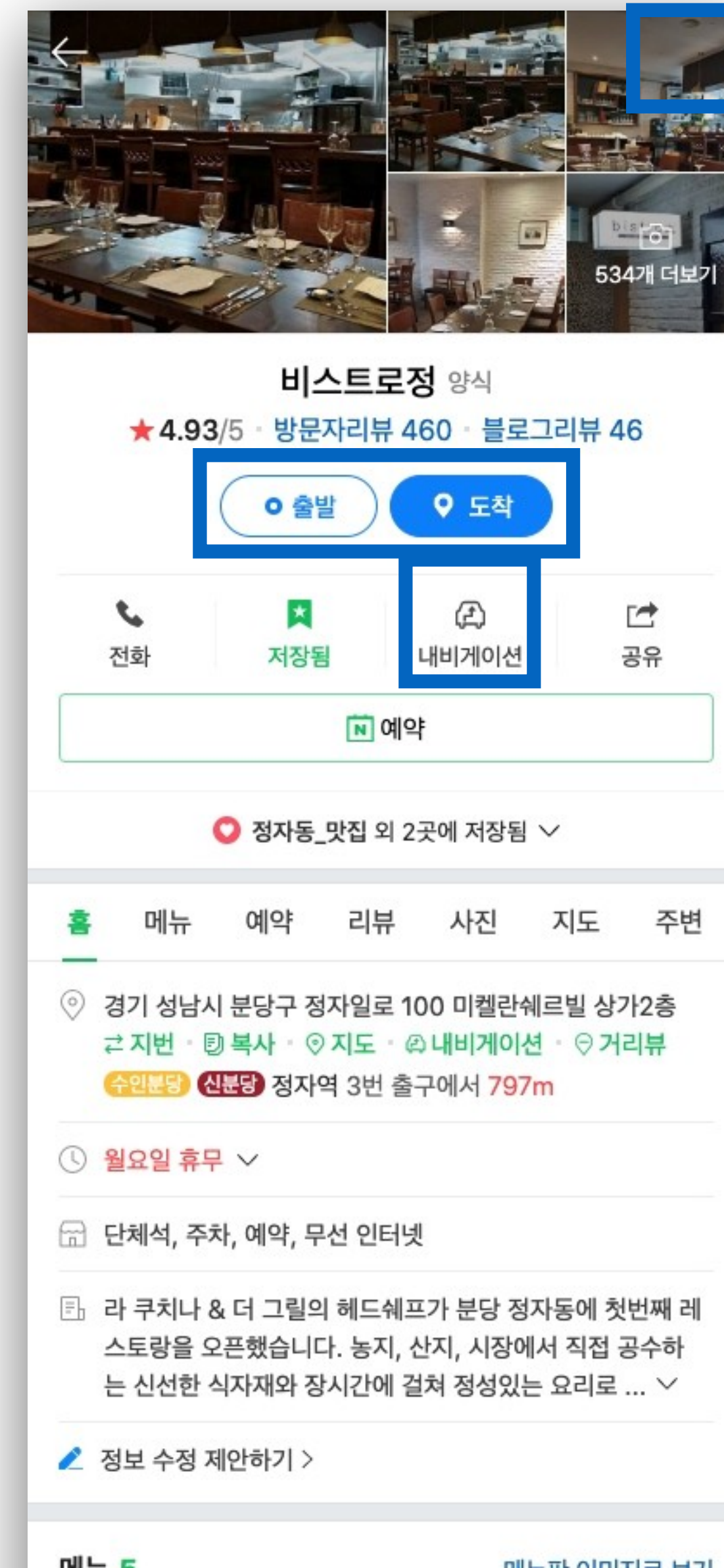
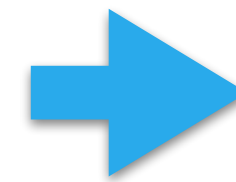
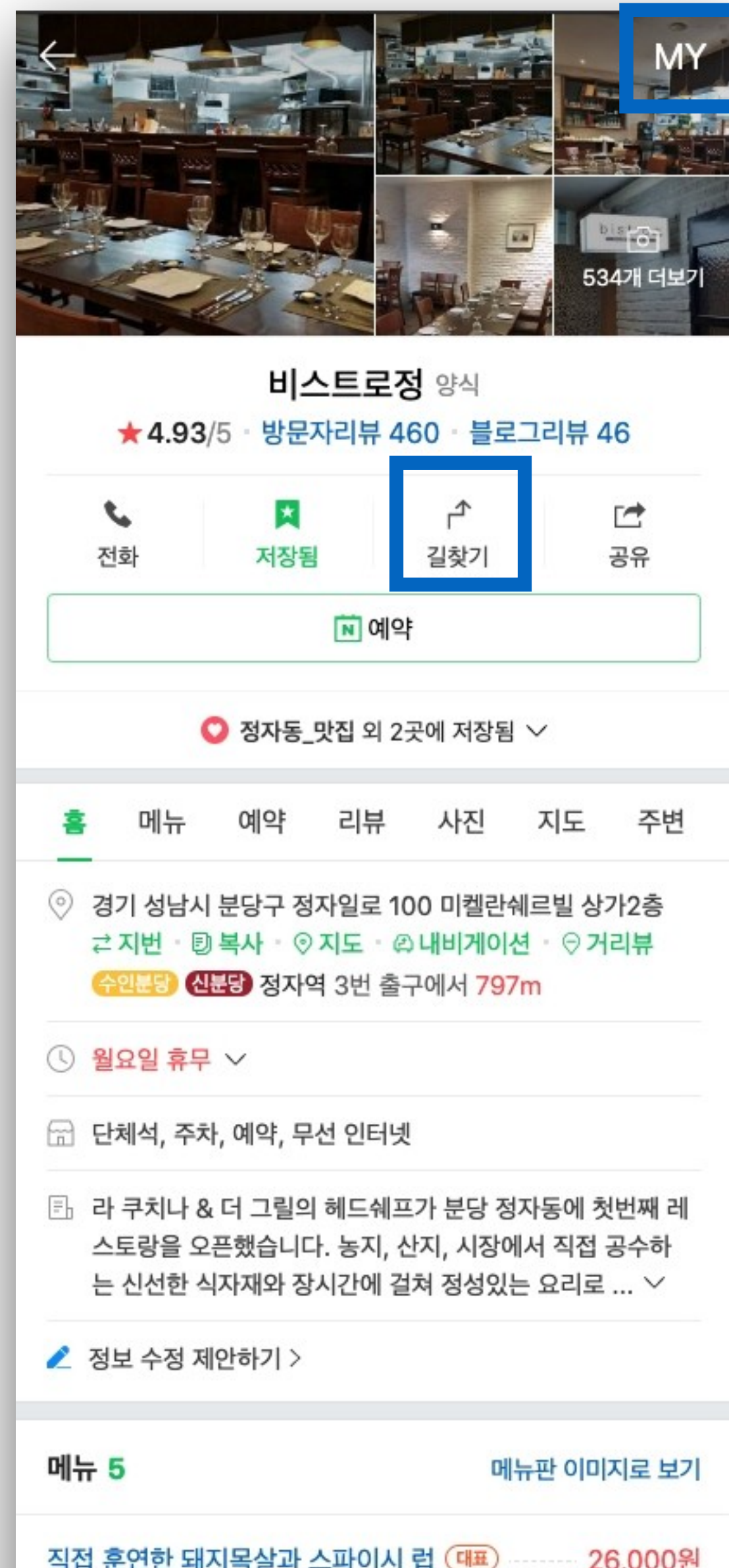
2.2 Webview 설계

목록

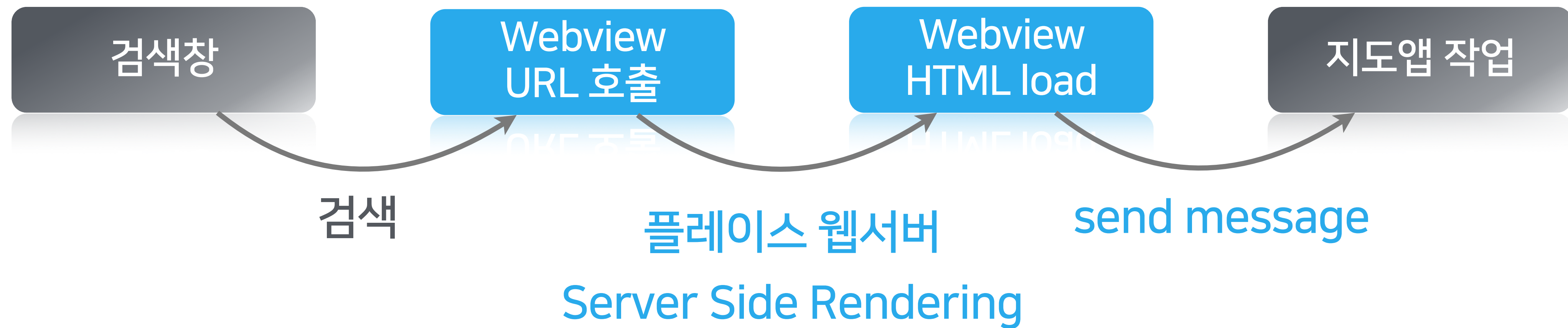


2.2 Webview 설계

업체 페이지



2.2 지도앱 + 플레이스 검색 Flow



2.3 Webview -> Native 통신

목록 결과를 지도에 마커로 그려줘야 해요

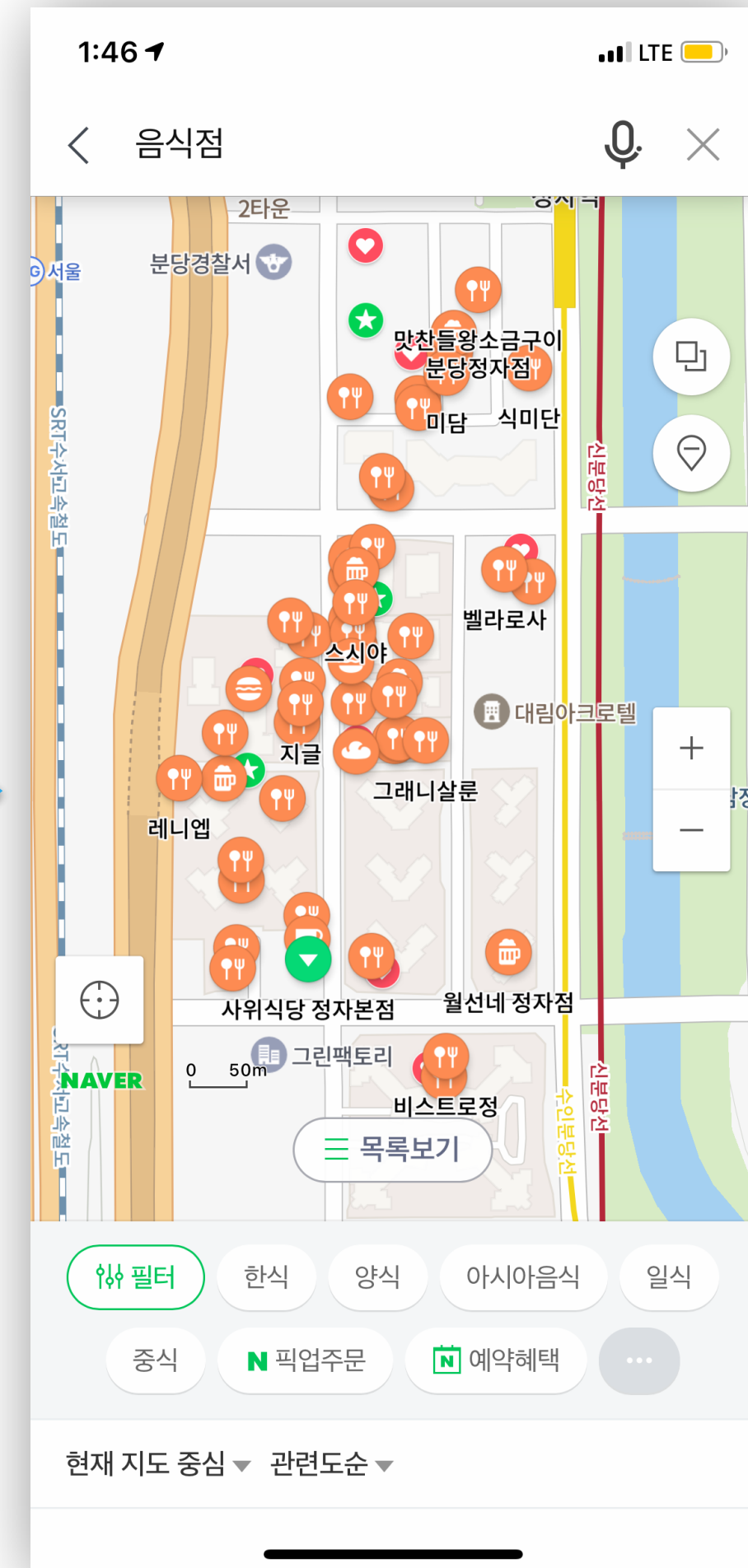
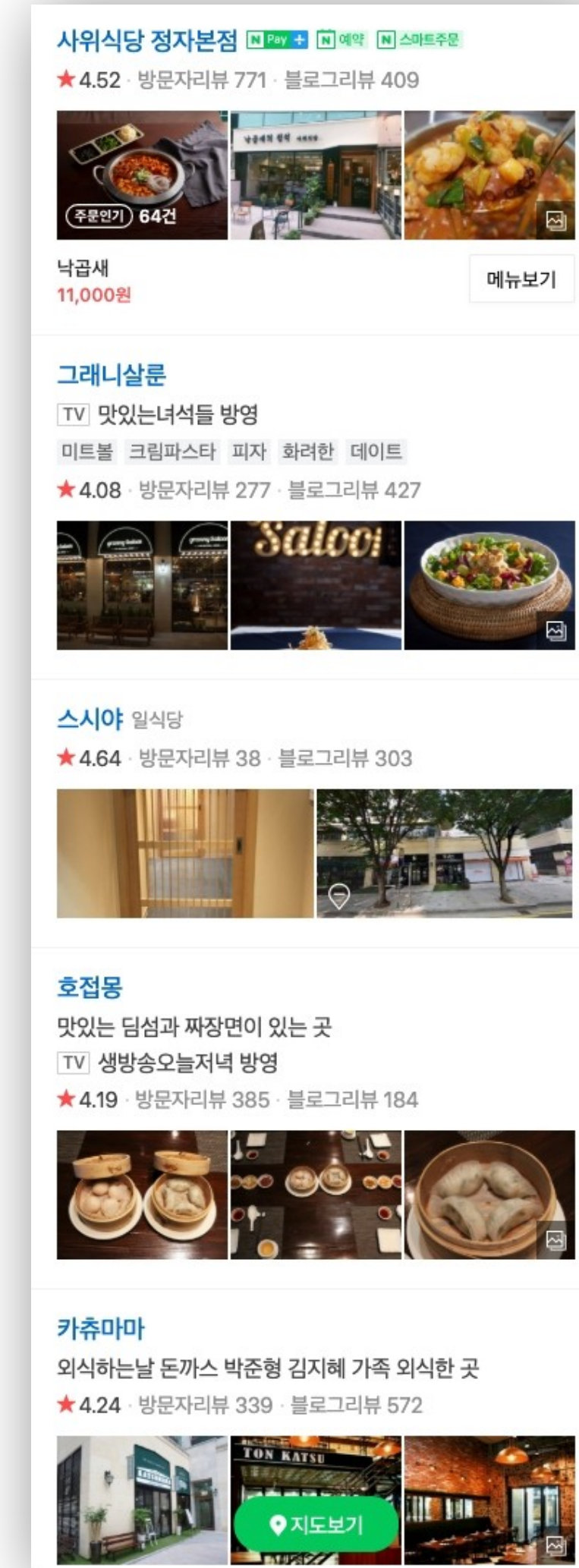
- Webview -> Native 어떻게 통신할 것인가?

1) (aOS) App scheme 사용 (inapp://)

- 주소 url 변경하면 side-effect 발생하는 경우가 있음
- 보이지 않는 `<iframe>` 만들어서 App scheme 호출

2) (iOS) webkit.messageHandler 사용

- 스크롤 하면서 잦은 app scheme 호출시 성능 이슈 발생
- render에 영향주지 않는 `postMessage()` 사용



2.3 Native -> Webview 호출

Native에서 Webview 호출 방식

- 기본: url 호출

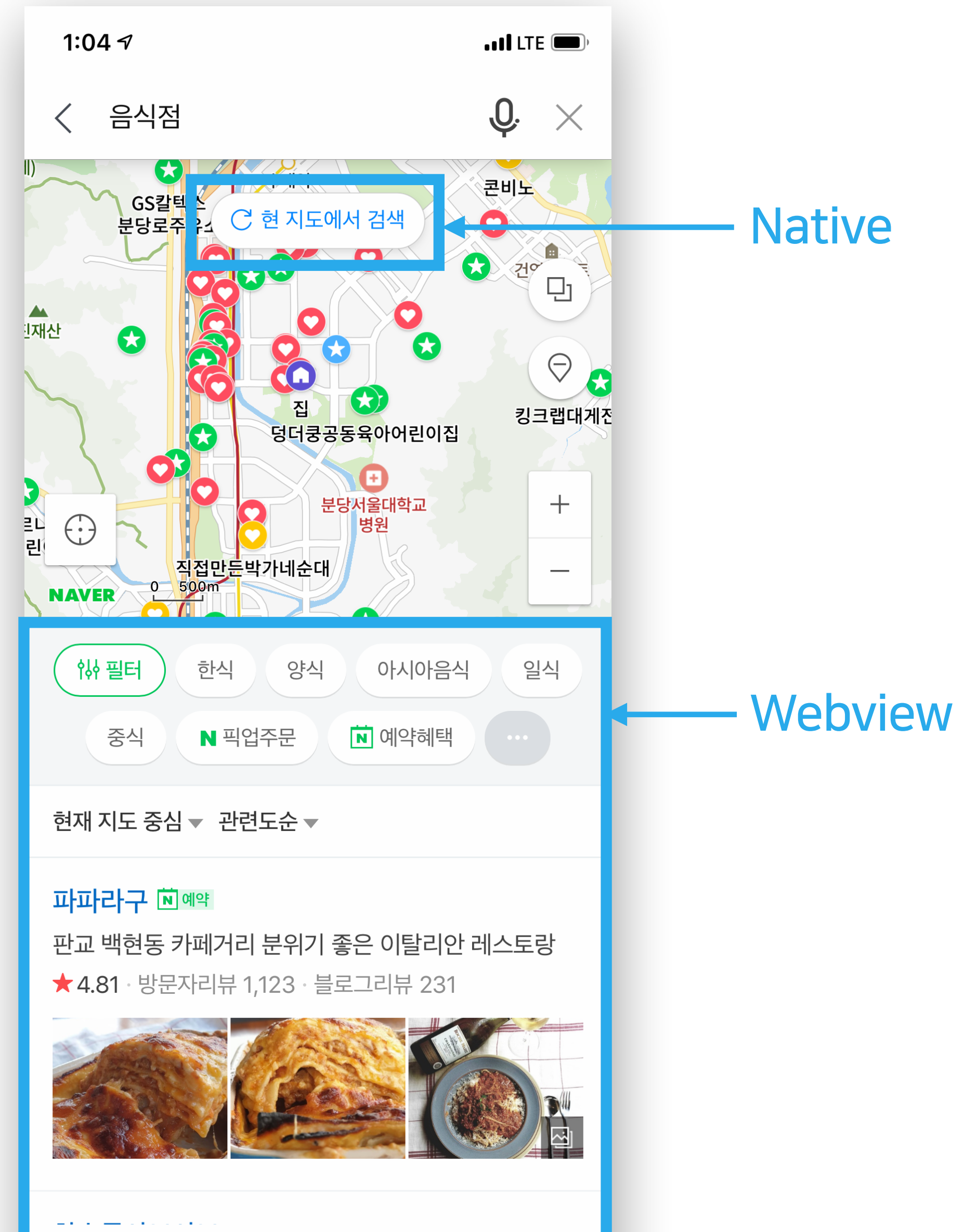
문제) '현 지도에서 검색' 터치 시 깜빡여요 🙄

원인) Native -> Webview - url 변경 호출

- 서버로 요청이 넘어가서 SSR(server side rendering)된 결과를 그린다.
- 우리는 SPA(Single Page App)인데? 🤔
- 보는 화면에서 목록만 업데이트 시켜 주세요.

해결) Webview에서 Global 함수를 제공하자

- SSR이 아닌 CSR(client side rendering)으로 업데이트



2.3 Webview와 Native의 event 처리

Webview 밑에 깔려 있는 지도판을 touch 할 수 있어야 해요.

- Webview의 이벤트를 어떻게 지도에게 전달하지?



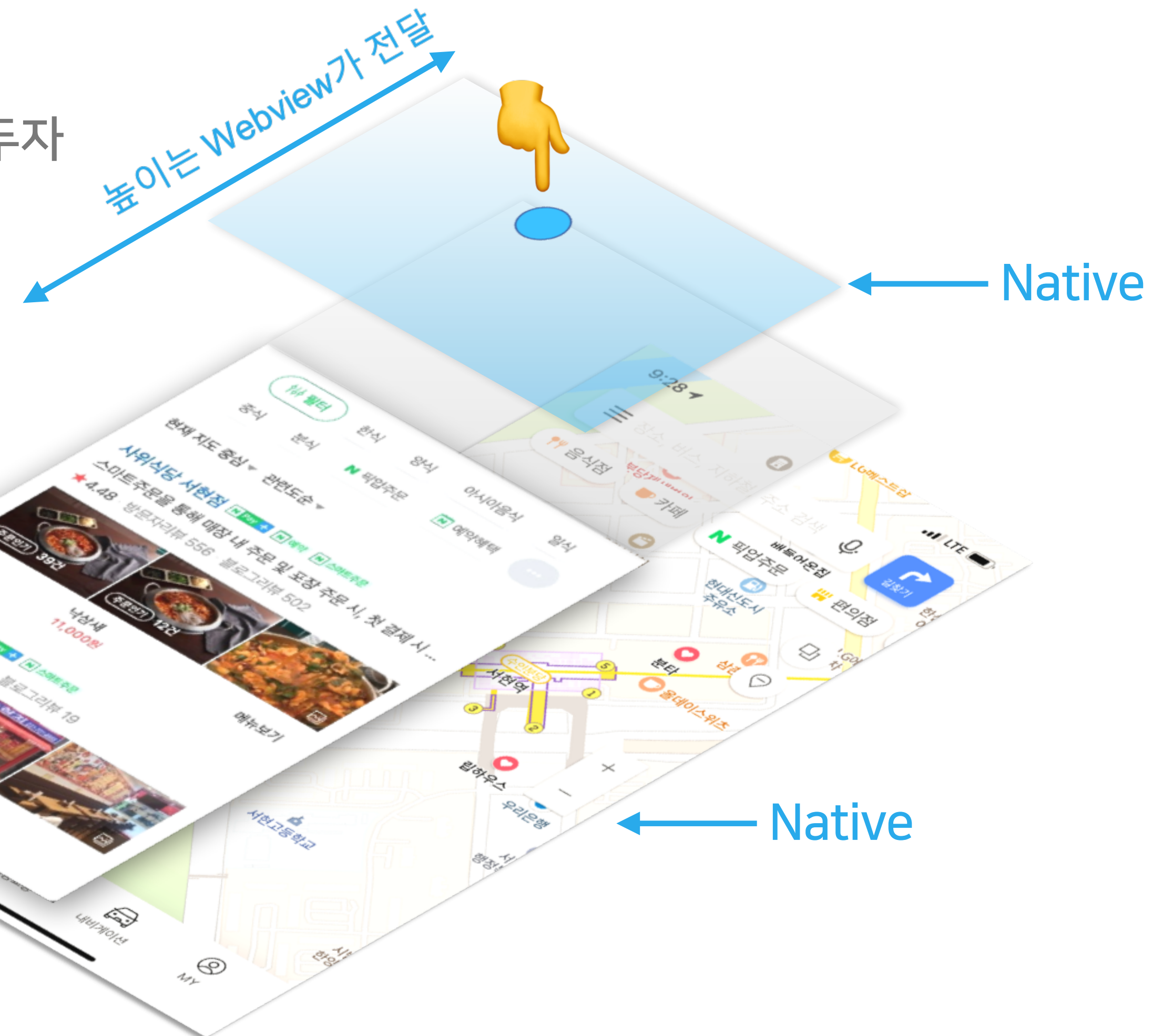
2.3 Webview와 Native의 event 처리

Webview 밑에 깔려 있는 지도판을 touch 할 수 있어야 해요.

Trick!

- Webview 위에 touch 이벤트를 받을 수 있는 가상 layer를 하나 더 두자
- Layer 사이즈는 Webview에서 전달하는 이벤트 기반으로 동작
- Webview 스크롤 될때마다 높이값 Native에 전달

Webview →



Q. 왜 Webview 영역 전체를 스크롤 하지 않았나요?

A. Webview 영역을 위로 올려 최상단이 되었을 때
내부 system scroll로 전환되는 사용성이 좋지 않음

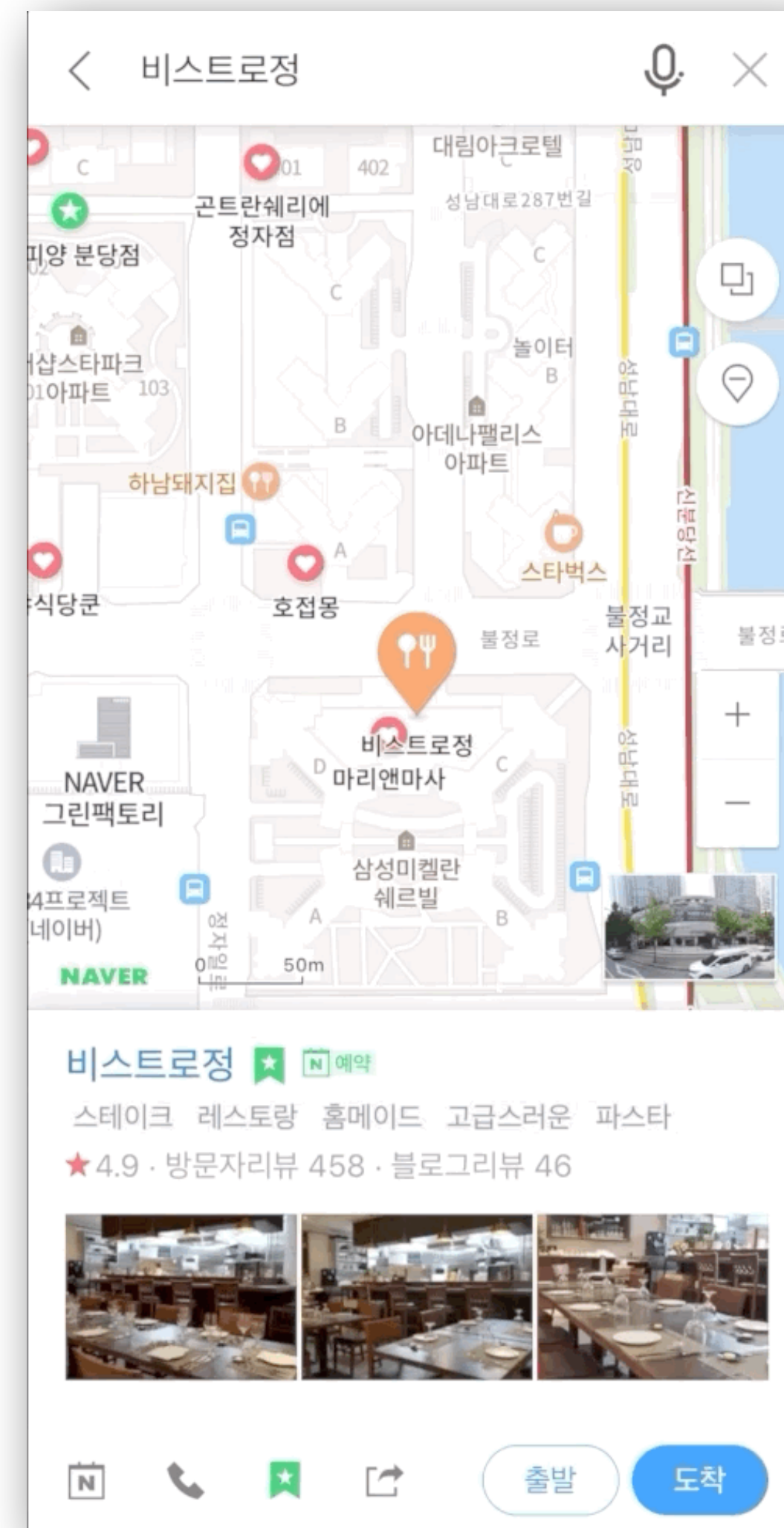
2.4 Native 와 webview의 결합

Native 카드와 Webview를 묶어서
자연스럽게 보이도록 하자

- Pull to refresh destroy

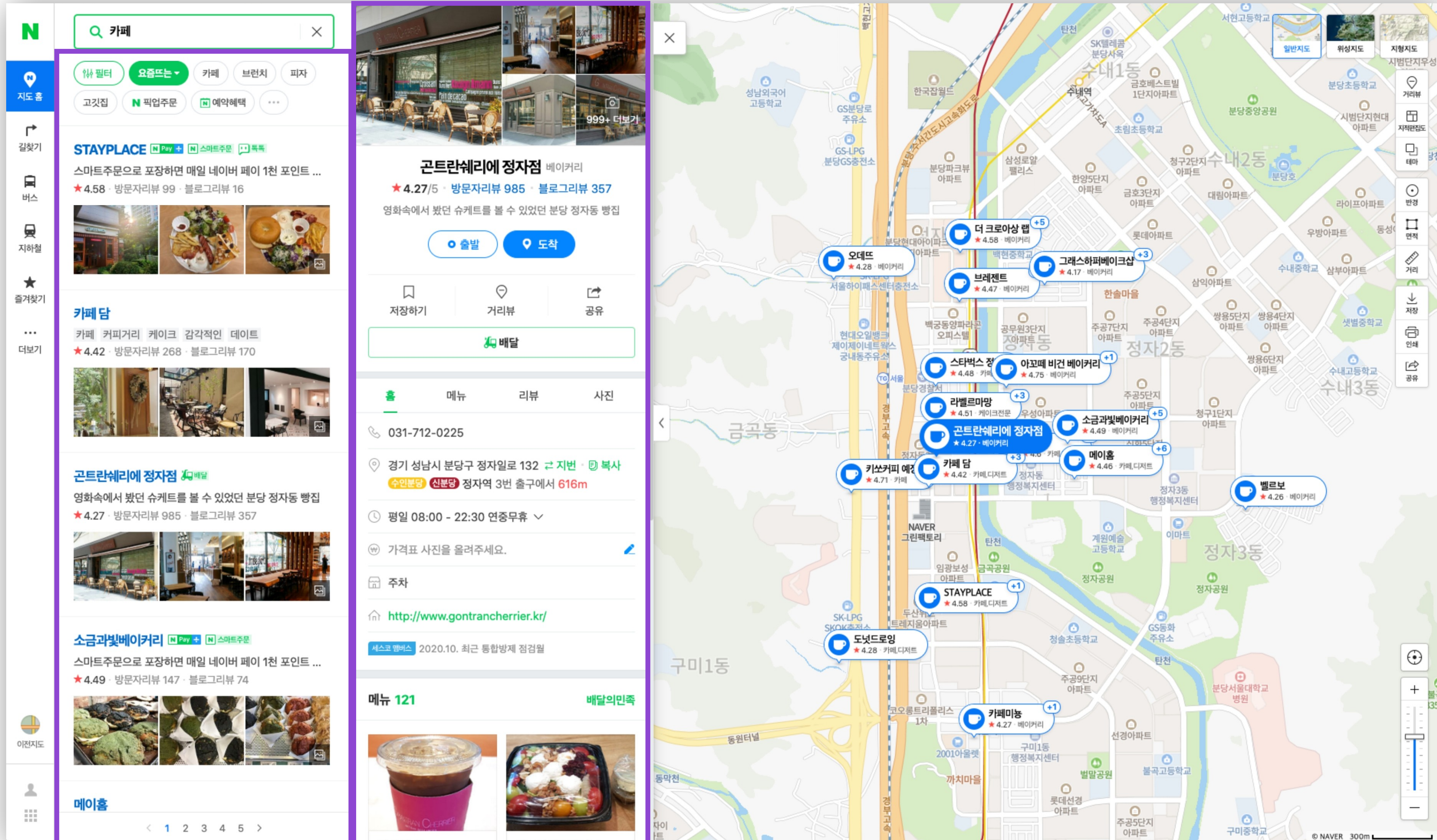
업체 카드
Native

업체 페이지
Webview



3. PC지도 서비스(Angular)에 플레이스 서비스(React) <iframe>으로 집어넣기

3.1 어떻게 통합할 것인가?



3.1 어떻게 통합할 것인가?

1안) 플레이스는 API만 제공 & 화면은 지도웹에서

2안) 플레이스 화면을 재사용 & 지도웹은 마커 🤞

2-1안) Angular에 React Component 포함

2-2안) HTML import(link rel) -> deprecated

2-3안) `<iframe>`으로 포함 🤞

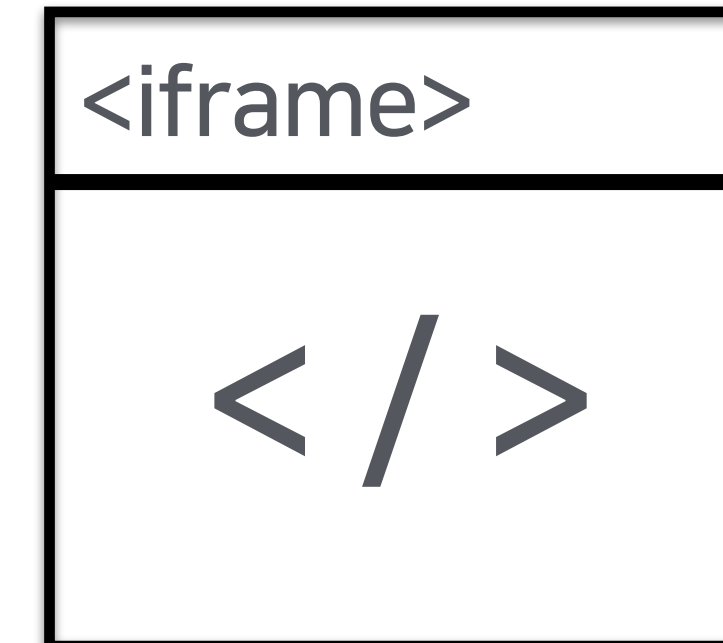
3.2 <iframe> 관참을까?!

단점)

- url 문제
- debugging 어려움
- 보안 조심

장점)

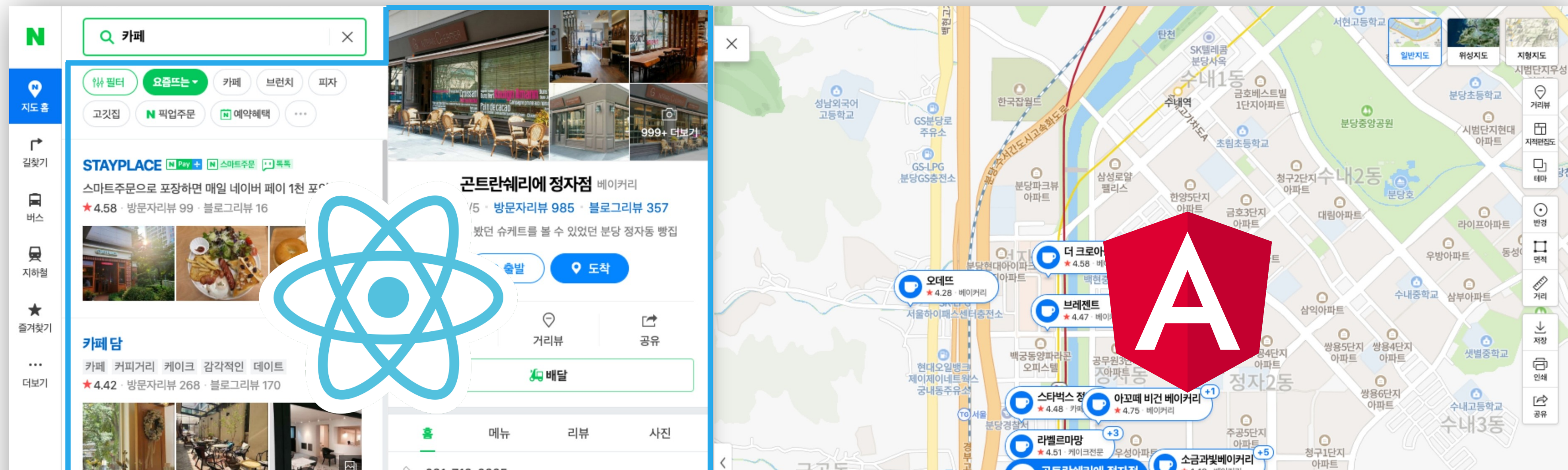
- 두 영역을 분리하여 고립시키기에는 오히려 좋다.
- cross domain components 조합



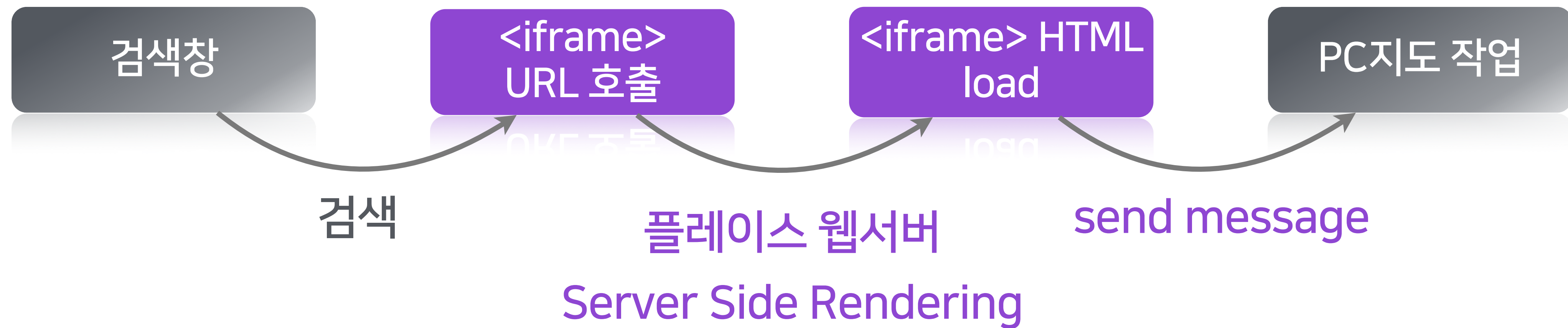
3.3 Angular + React

지도 서비스(Angular)와 플레이스 서비스(React)의 공존

- <iframe> 사용으로 서비스 간 고립
- library/framework가 달라서 발생하는 문제는 없음
- 서비스 간에 통신을 어떻게 할지가 더 중요



3.4 PC지도 + 플레이스 검색 Flow



3.5 서비스간의 통신

단순한 통신 흐름 지향

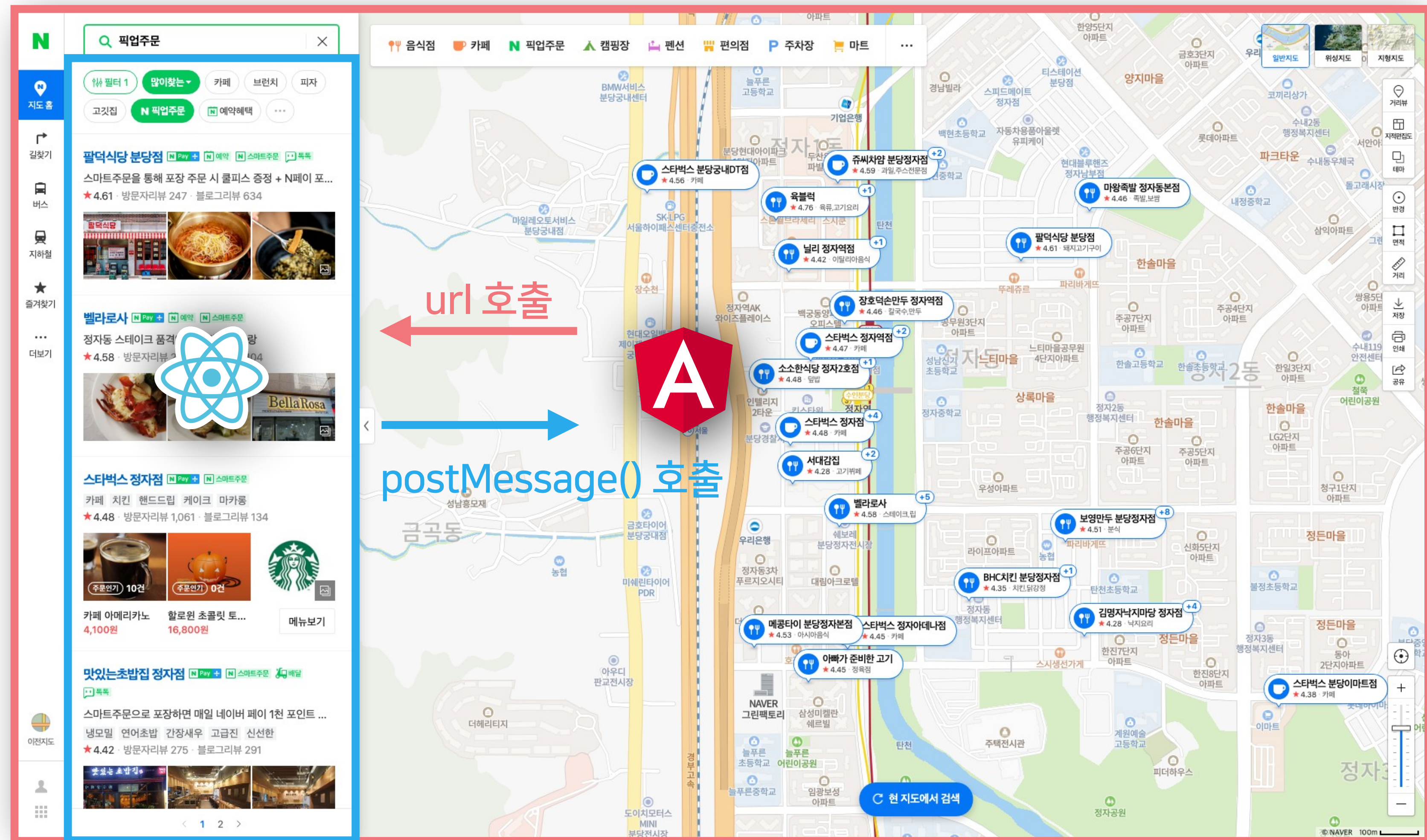
지도 -> 플레이스

- `<iframe>` URL 변경

플레이스 -> 지도

- `postMessage()` 사용

`postMessage({ action, data })`

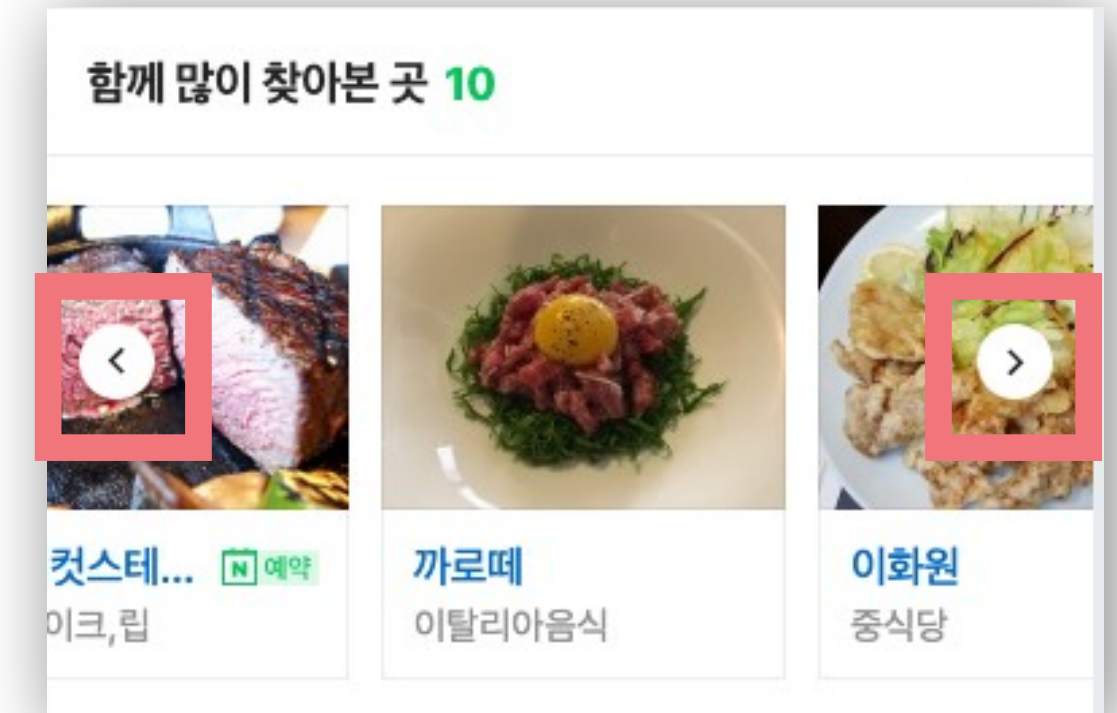
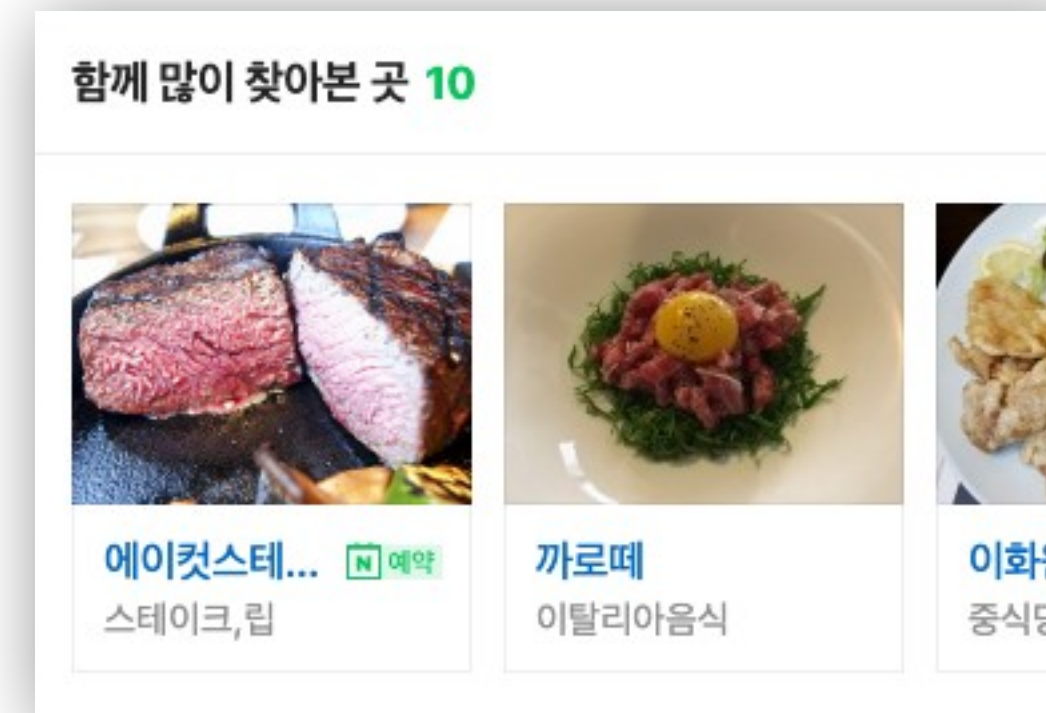


3.6 PC서비스에서 모바일과 달라지는 점

1) 가로 스크롤

가로 system scroll

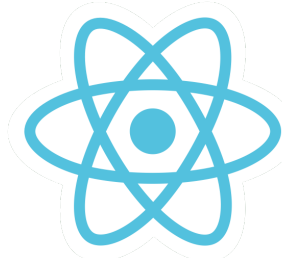
- PC에서 가로 스크롤은 익숙하지 않음
- IE11 스크롤 커스텀 노출 어려움
- > @egjs/flicking + 좌우 버튼 추가



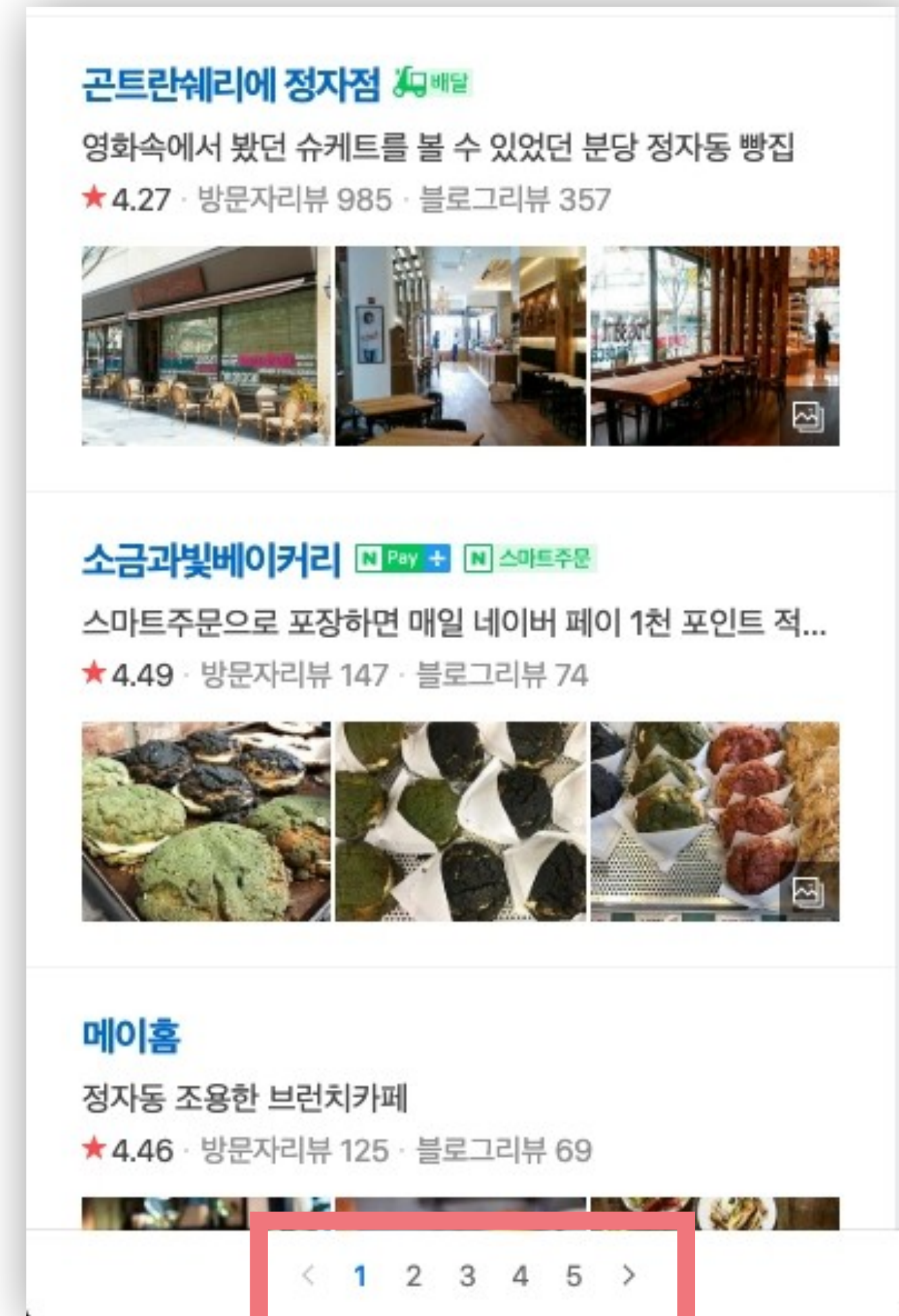
3.6 PC서비스에서 모바일과 달라지는 점

2) 무한 스크롤 관련처리

- 모바일: 무한 스크롤
- PC: 페이징



Wrapper Component 변경으로 대응



3.6 PC서비스에서 모바일과 달라지는 점

3) PC만 지원하는 이벤트 처리

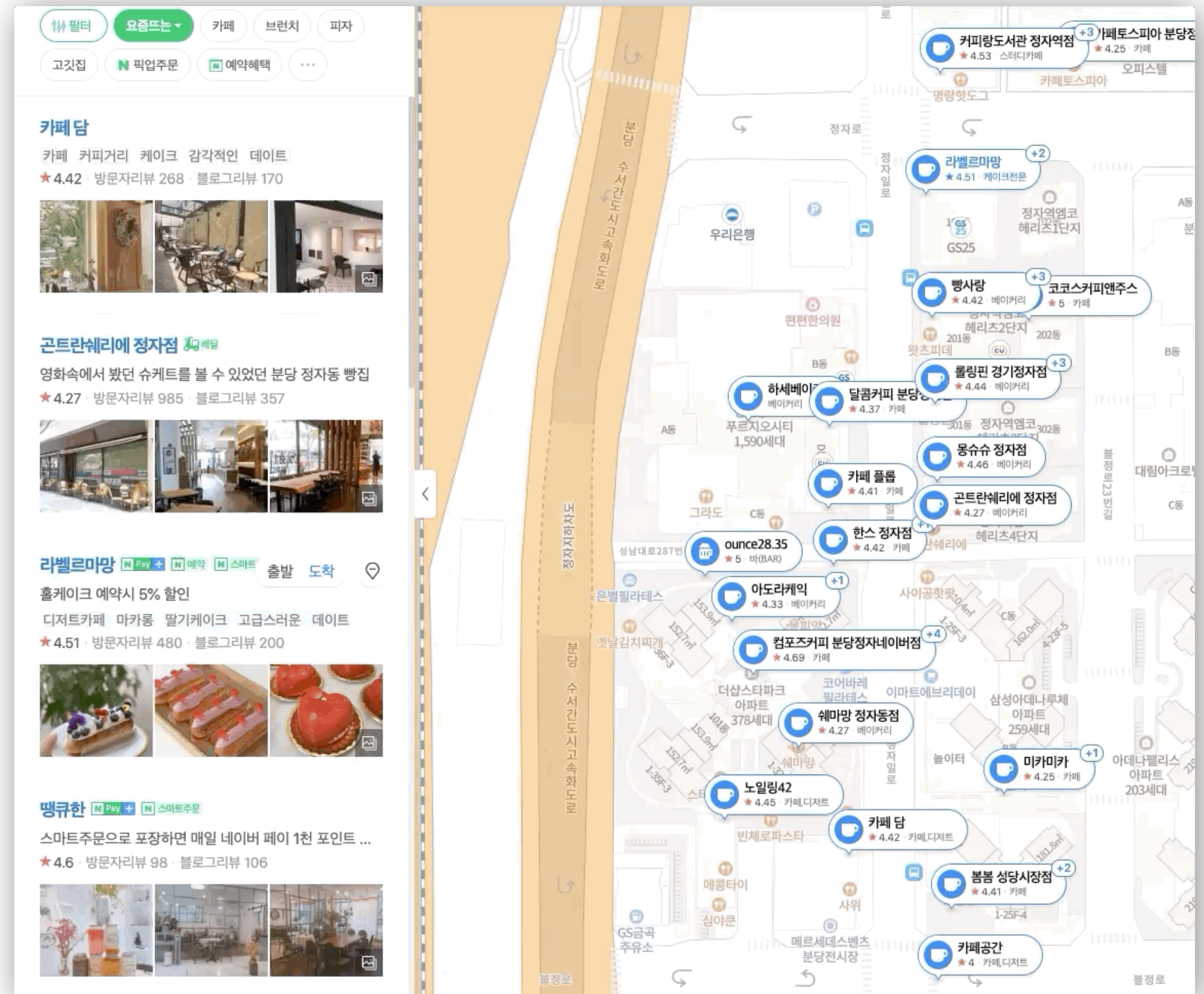
mouseenter 이벤트

mouseleave 이벤트

3) PC 미지원하는 이벤트 처리

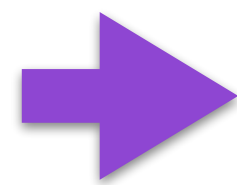
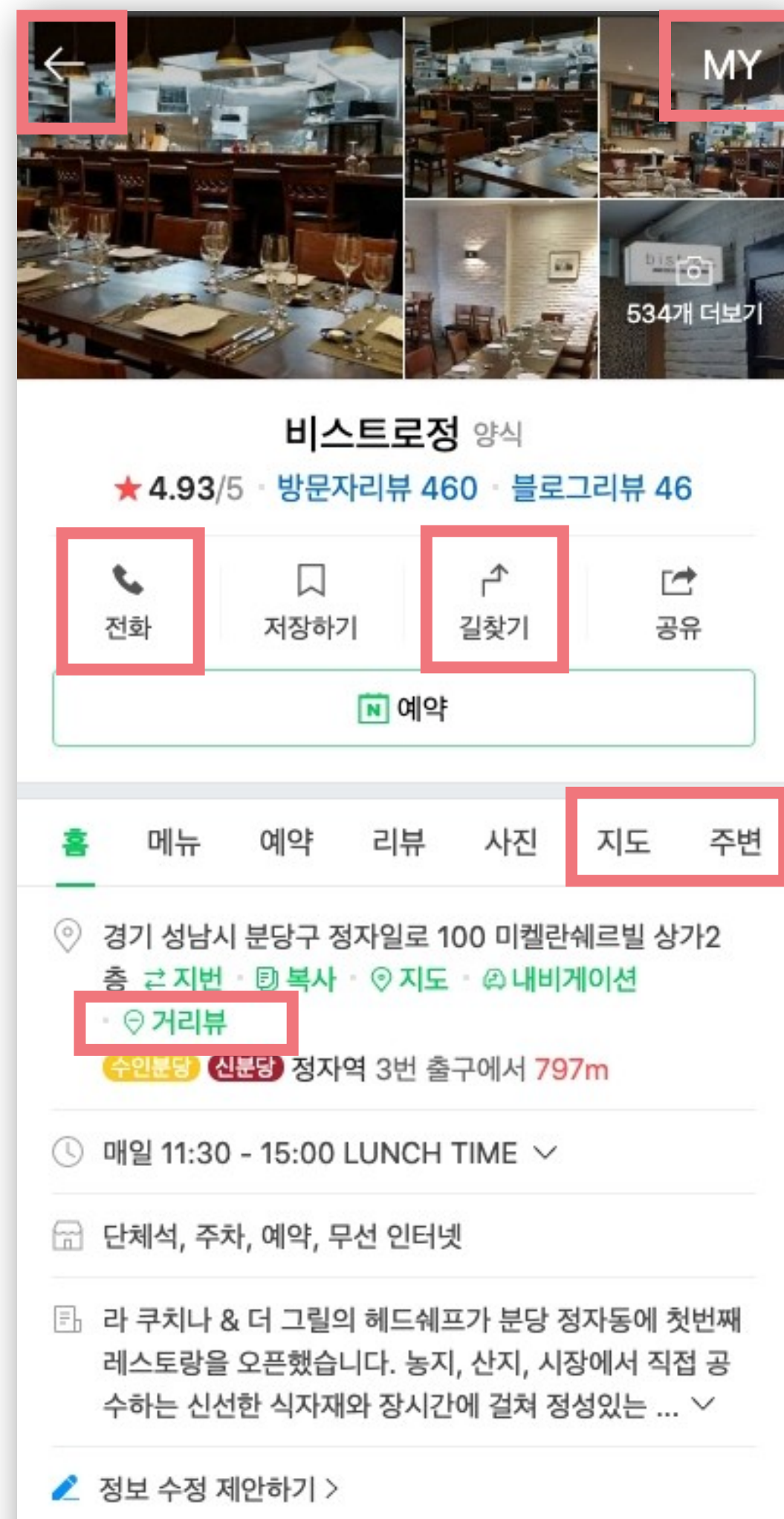
- touchevent시 사라지는 tooltip

-> scroll event로 처리

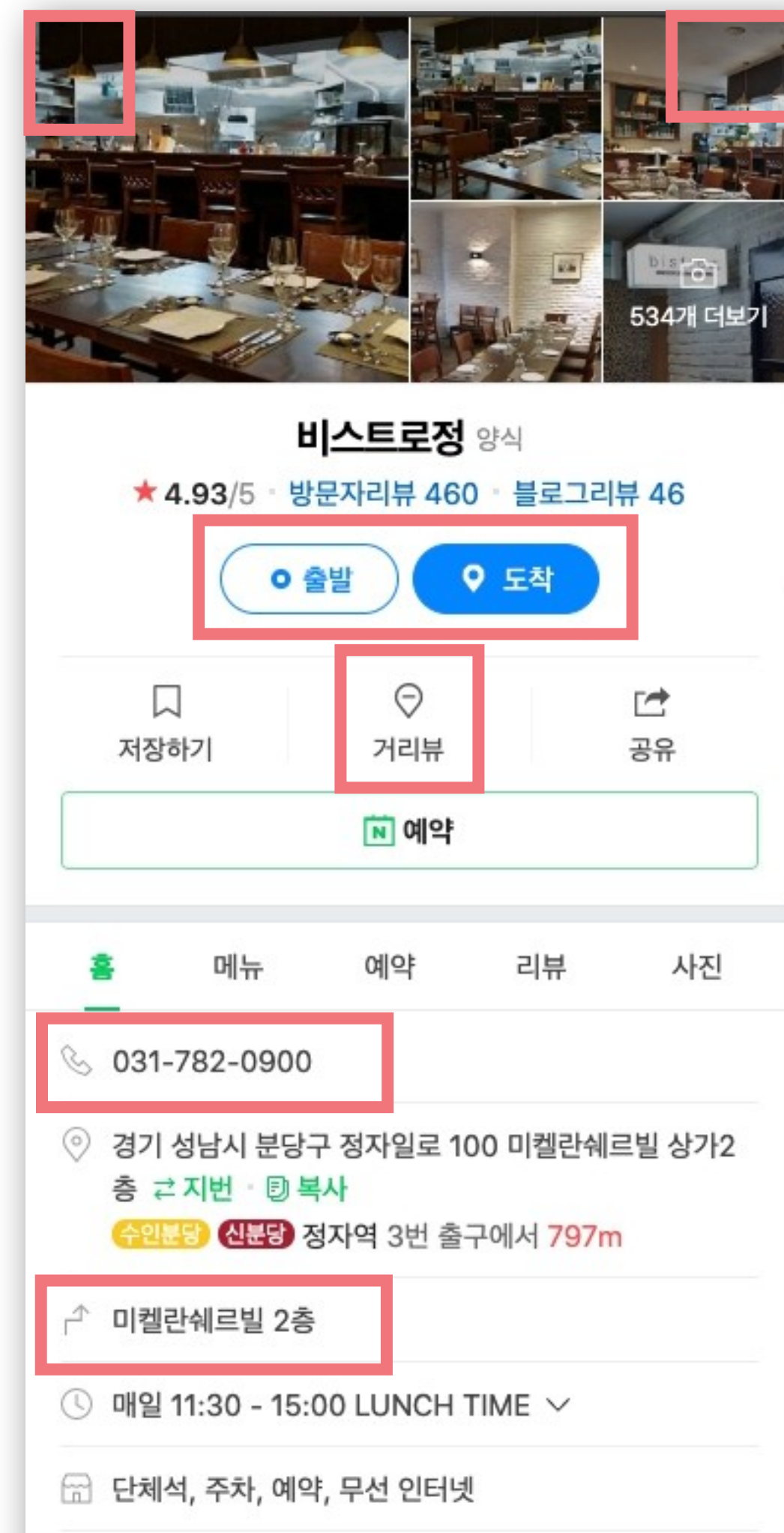


3.6 PC서비스에서 모바일과 달라지는 점

4) 사용성 때문에 변경되는 부분



동작 환경과
사용성에 맞춰 변경



3.6 PC서비스에서 모바일과 달라지는 점

4) 사용성 때문에 변경되는 부분

넓은 화면을 활용 - 포토뷰어 / 거리뷰

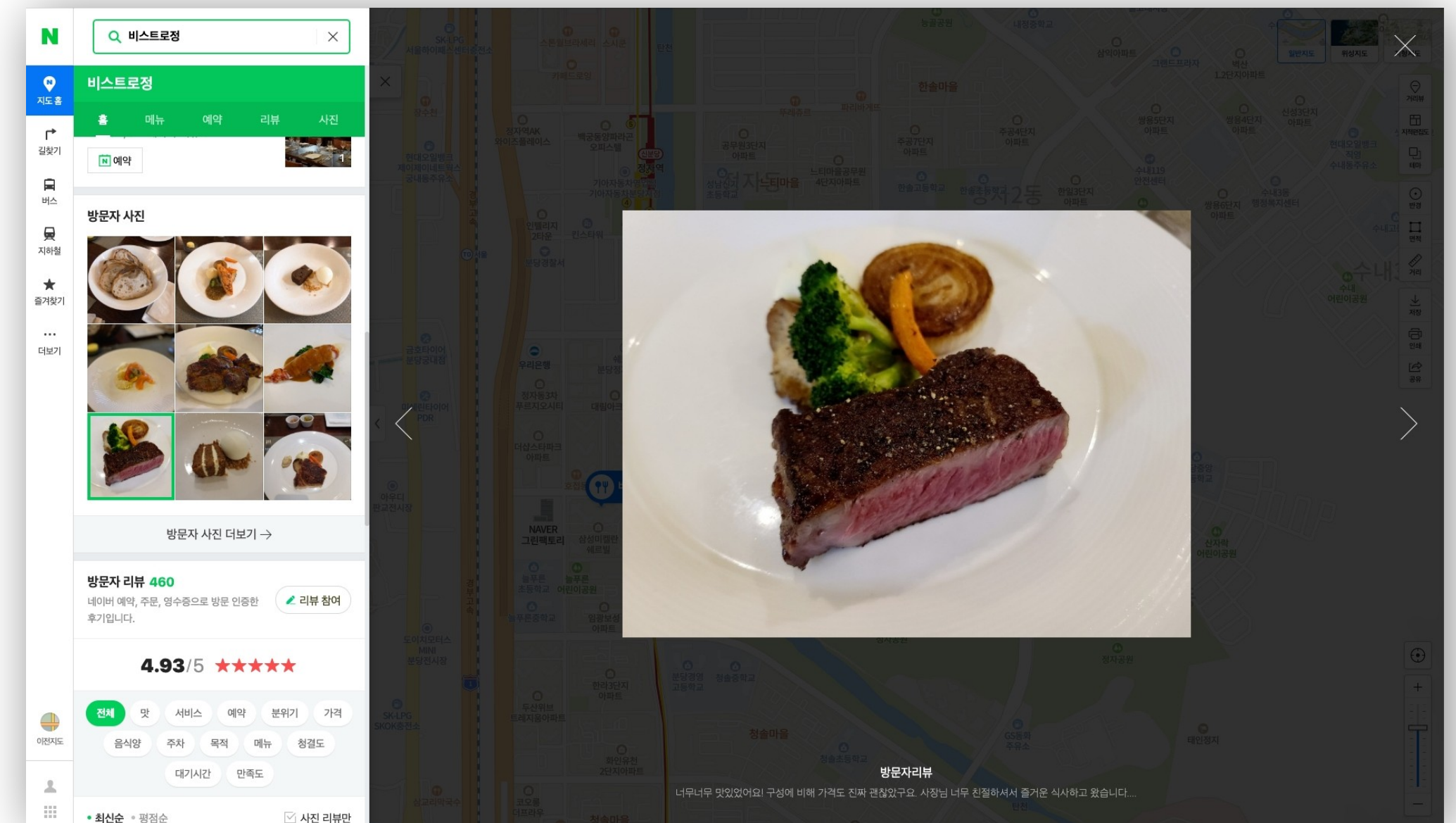
외부 페이지 이동

<iframe> 내에서 외부 페이지가 열려요. => 새 창 으로

로그인 필요한 경우

<iframe> 🖱️ 본 창에 로그인 처리 요청

🖱️ 본 창 & <iframe> 새로고침(로그인 쿠키 반영)



3.7 Internet Explorer 11 대응

PC 브라우저 대응은 어렵지 않습니다.

 IE 대응이 귀찮고 짜증나고 어렵습니다. 🙄🙄

3.7 Internet Explorer 11 대응

Build target 변경

모바일

target: ES6(ES2015) + polyfill 대응

IE11

target: ES5 + polyfill 대응

+ 사용하고 있는 library 들 target 확인 필요

react-spring

- ES5로 빌드된 main file 사용 필요. module file은 ES2015 기준으로 빌드 되어 있음.

svg-loader

- target: ES5로 변경
- webpack babel-loader에 plugin 추가로 대응

```
options: {  
  plugins: [  
    '@babel/plugin-transform-arrow-functions',  
    '@babel/plugin-transform-block-scoping',  
    '@babel/plugin-transform-destructuring'  
  ],  
}
```

3.7 Internet Explorer 11 대응

IE11 Polyfill 대응

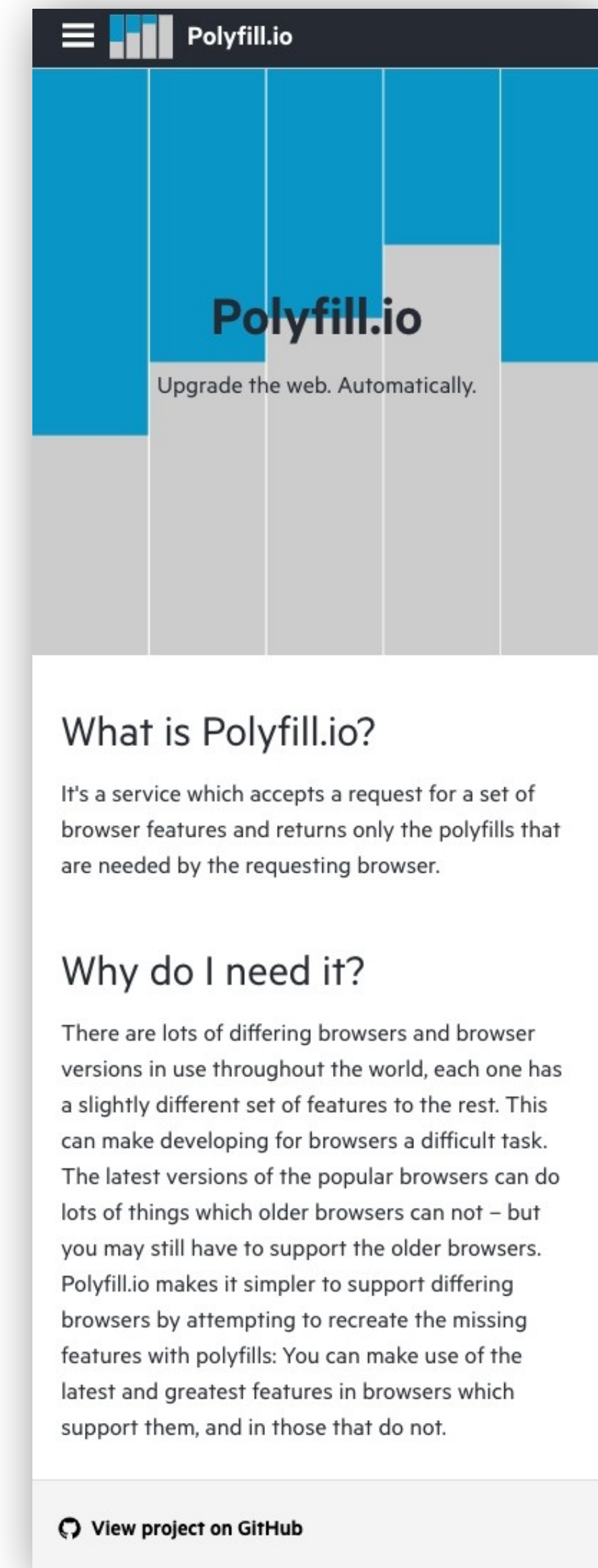
- polyfill.io 에서 IE11 용 polyfill 다운로드
 - features: es2015, es2016, es2017
 - flags: gated
- 웹 서버에서 **user-agent가 IE11인 경우** 응답 HTML에 IE11용 polyfill 포함

대안1) polyfill-library

- realtime user-agent 기반 대응은 좋으나 성능 평가 불확실함
- 서비스가 모든 user-agent 대응할 필요는 없음

대안2) react-app-polyfill

- ES2016+ 미지원 문법 발견함. 미지원 범위를 찾아 대응하기 번거로움



3.7 Internet Explorer 11 대응

IE11 미지원 CSS 변경

display: -webkit-line-clamp;

- 여러줄 말줄임 관련 스타일 지원 안함
- '...' 없이 max-height만 설정

position: sticky;

- position: fixed;로 변경 대응

SVG without width & height

- 두 값 중 하나만 설정 하는 경우 영역 크기를 제대로 잡지 못함
- width & height 항상 설정

object-fit: cover;

- 이미지 비율대로 늘리고 줄이기
- background-size: cover;로 변경



Desktop				
Chrome	Firefox	IE	Edge	Safari
14*	68*	No	17	5*

Mobile / Tablet			
Android Chrome	Android Firefox	Android	iOS Safari
85*	79*	2.3*	5.0-5.1*

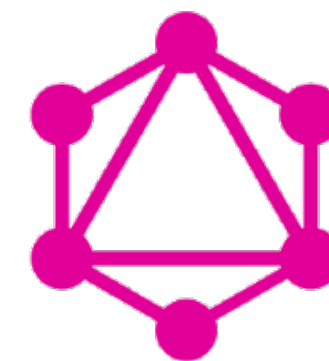
출처: <https://css-tricks.com/almanac/properties/l/line-clamp/>

4. 한 벌의 코드로 여러 서비스 제공은 어떻게 할까?

4.1 플레이스 서비스 개발 스택 (참고)

JavaScript Web Fullstack

- TypeScript
- Node.js with Koa
- React with hooks
- GraphQL with Apollo
- kubernetes with NCC



GraphQL



kubernetes

4.2 어떻게 분기처리 할 것인가

지속 가능한 코드 = 유지보수하기 쉬운 코드

- 서비스 환경 체크하는 if / else 분기는 되도록 적게
-  component 단위를 작은 기능에 집중해서 작게 만든다.

서비스 운영이 쉬운 환경 구성

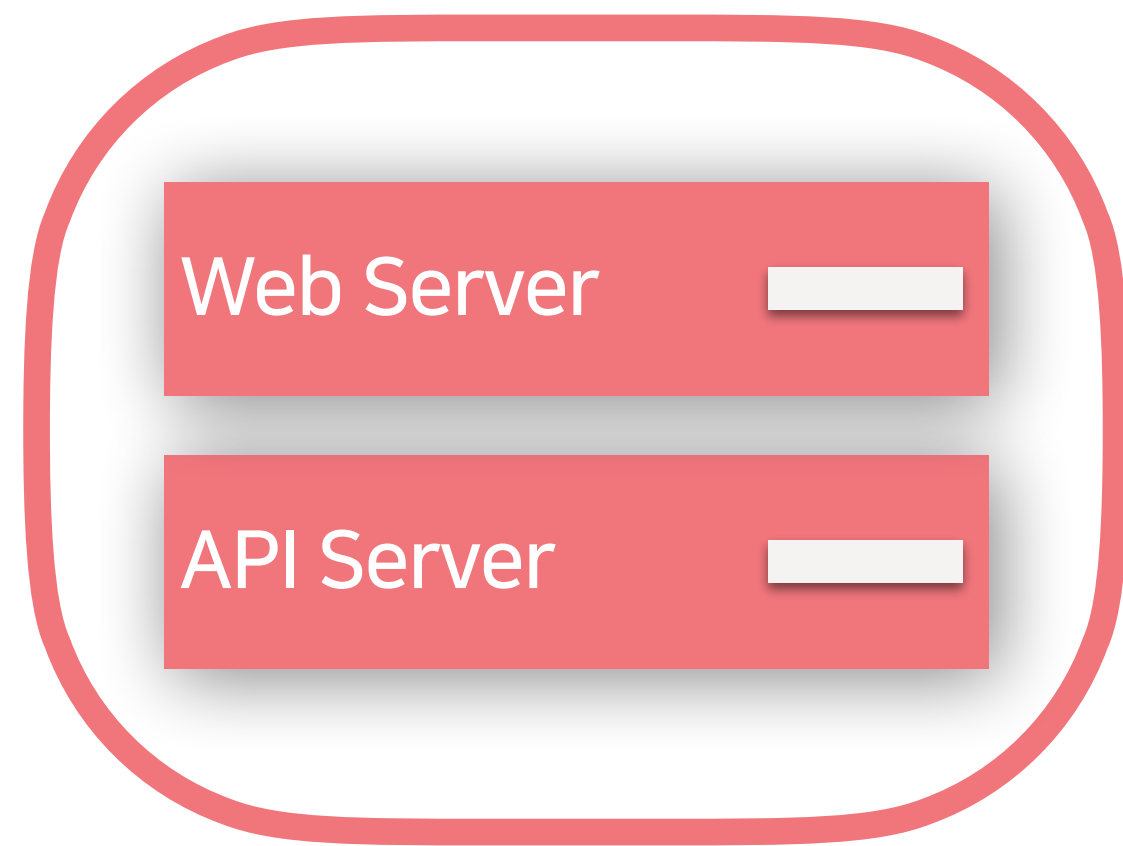
- 코드는 재사용
- domain & server & build 는 분리

👉 배포 시 배포 환경에만 집중할 수 있다!

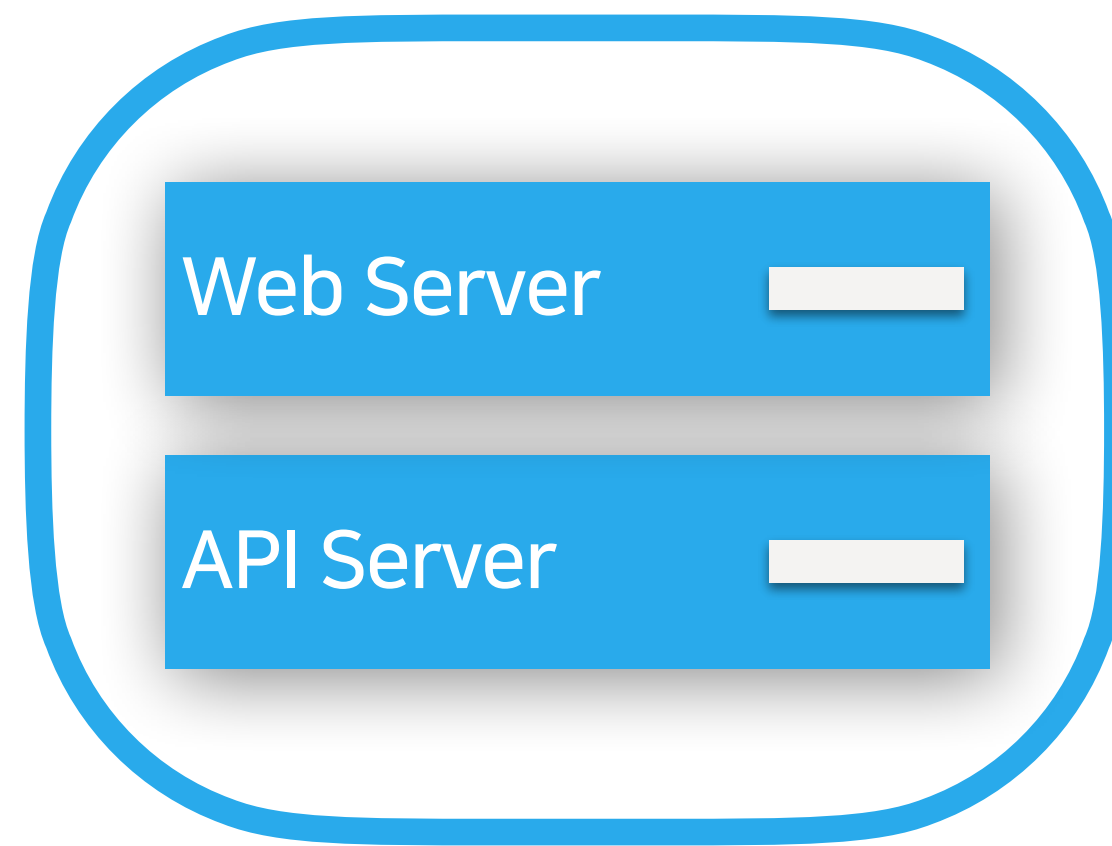
4.3 배포 환경 분리

독립된 별도 서버군으로 별도 도메인으로 대응

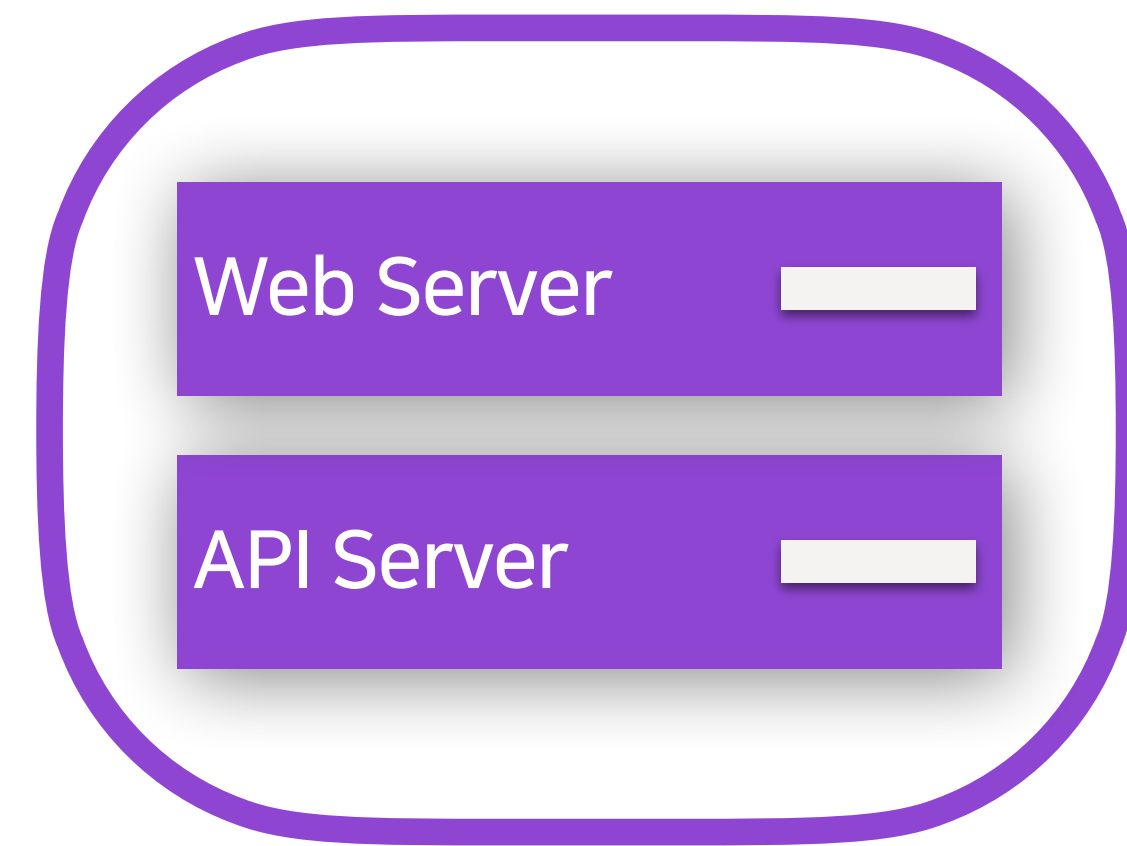
- 코드는 재사용하지만 서비스간 독립성 보장
- = webpack build를 별도로 한다.
- = 환경별로 entry 포인트를 따로 가져간다.



모바일 웹



지도앱

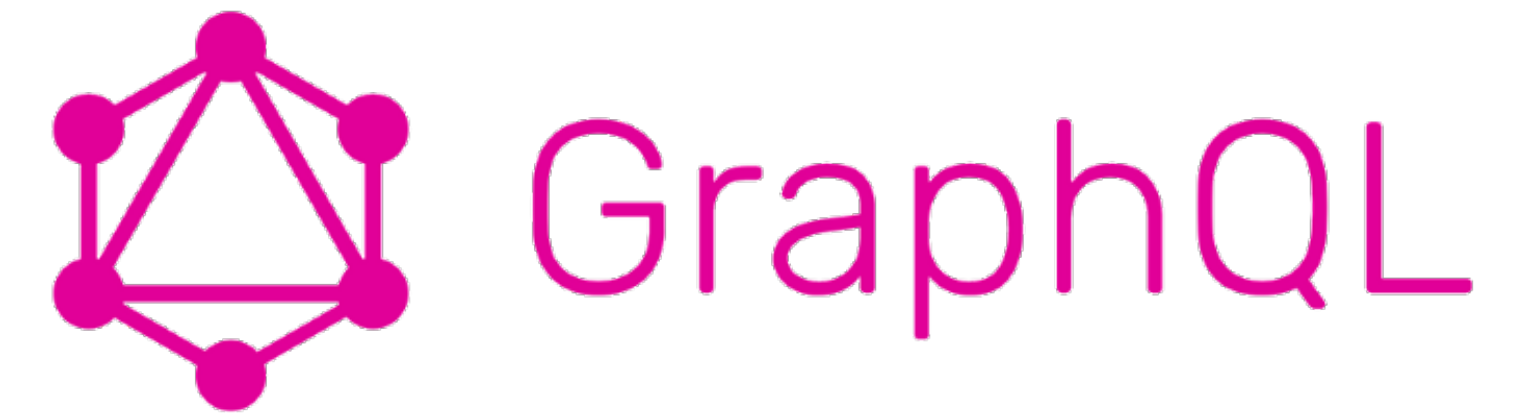


PC지도

4.3 API 레벨의 분기

GraphQL의 장점을 살리자

- GraphQL은 **versionless** 추구
 - 하위 호환성을 지키기 위해 field type 변경 X
 - 변경이 필요한 field는 신규로 추가하여 대응
- 화면별로 필요한 field를 정의하고 호출하는 것으로 하자
- GraphQL resolver는 client가 어떤 환경인지 몰라도 되도록 관리
- **분기의 권한은 client에게** 있다.



4.3 API 레벨의 분기

GraphQL의 장점을 살리자

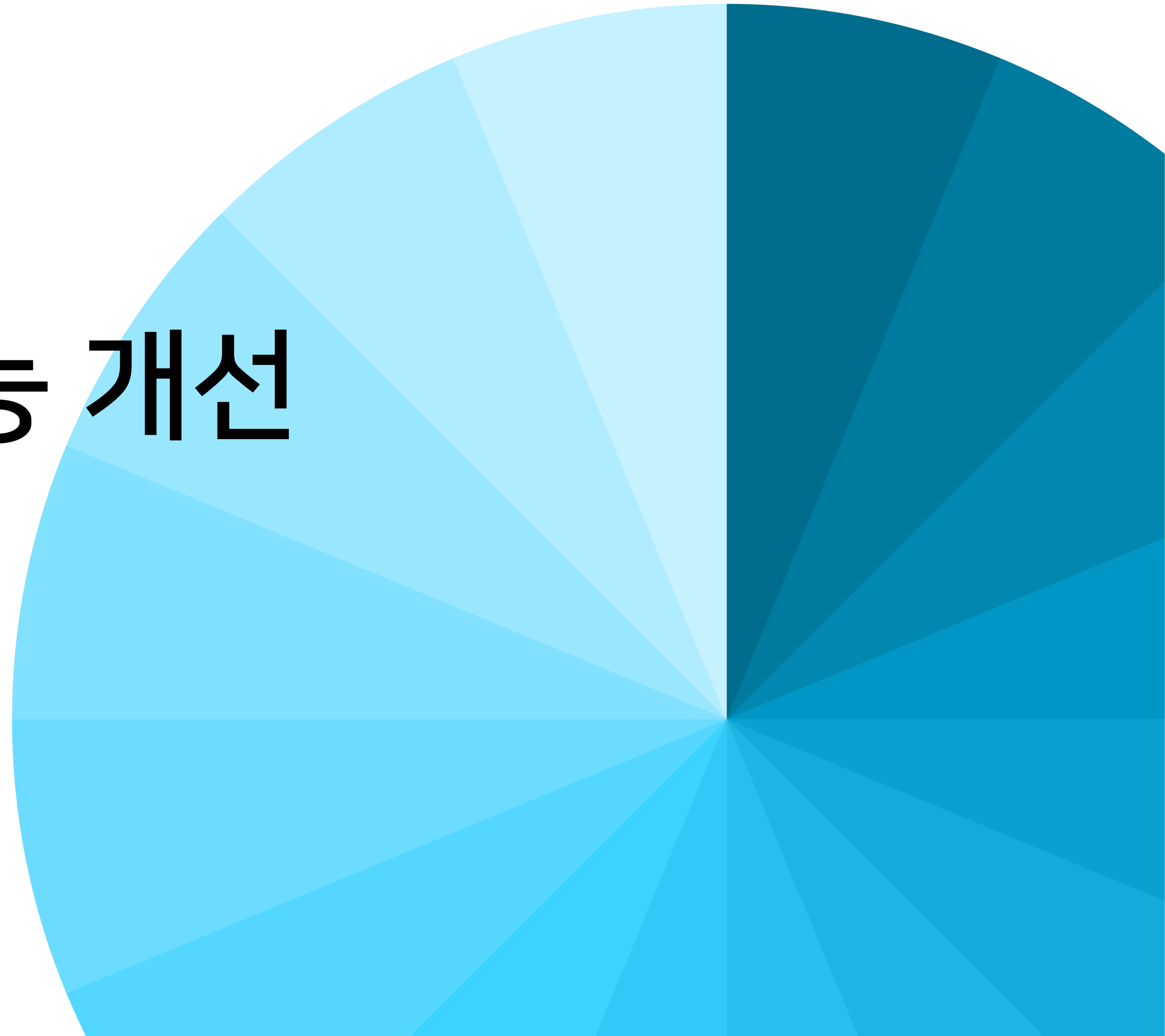
GraphQL query 레벨에서 device 별 분기 처리 진행

- @skip, @include 사용

```
type Business {  
  name: String!  
  mobileBookingUrl: String  
  pcBookingUrl: String  
}
```

```
query getBusiness($id: String!, $isPC: Boolean!) {  
  business(id: $id) {  
    name  
    mobileBookingUrl @skip(if: $isPC)  
    pcBookingUrl @include(if: $isPC)  
  }  
}
```

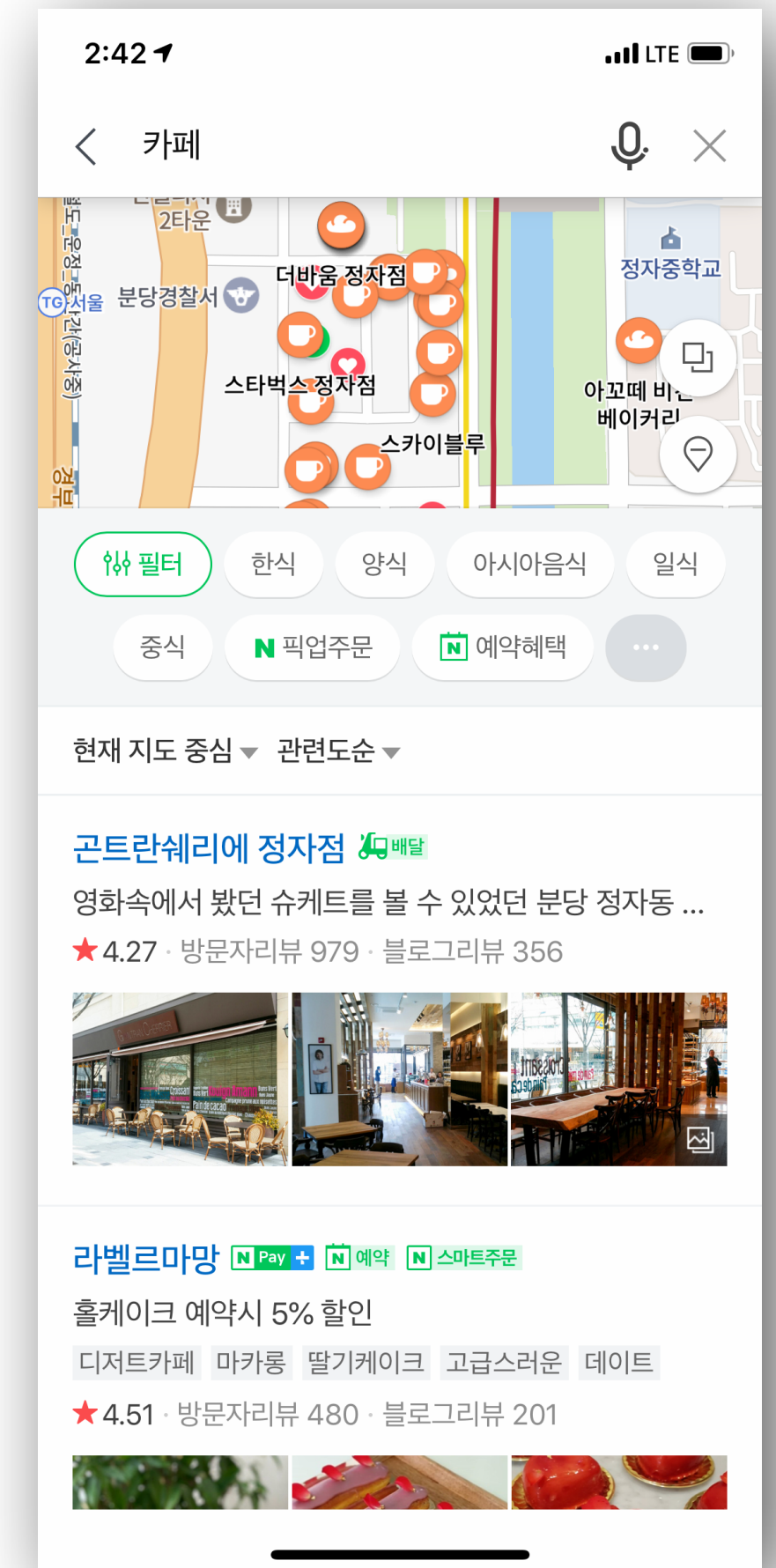
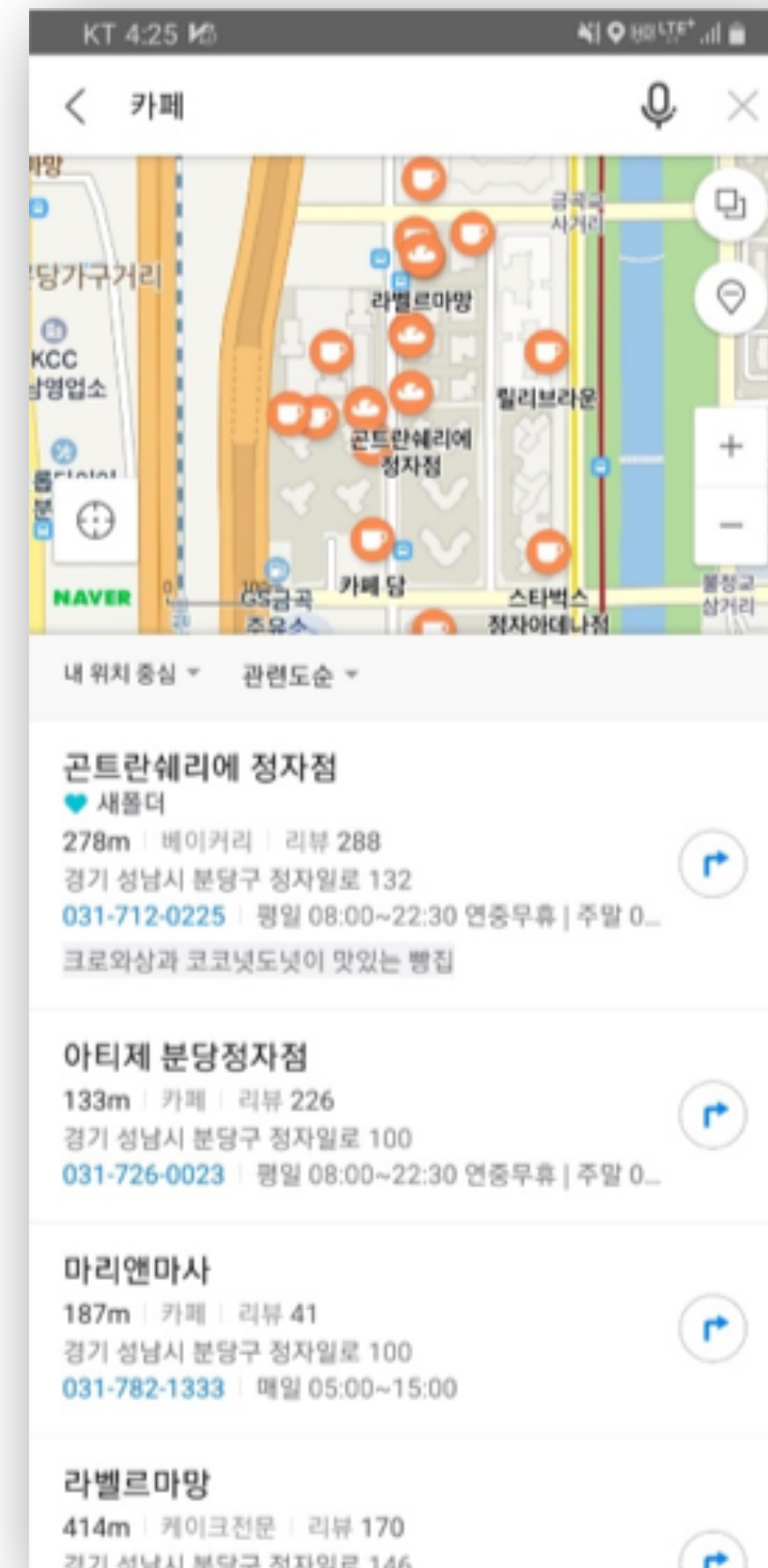
5.성능 개선



5.1 이미지 사이즈를 줄여야 한다.

사진이 많은 건 보기 좋은데 느려졌어요. 🐌

- 빠르고 가벼운 화면 VS. 사용자에게 좋은 정보
- 이미지 최적화를 하자! 🛠️



5.1 이미지 사이즈를 줄여야 한다.

이미지 최적화

1) 노출 형상에 맞는 Thumbnail 생성

2) Thumbnail 최적화

- quality 줄이기 = **이미지 용량** ↓
- sharpen 추가하기 = **이미지 품질** ↑



750x500 jpeg quality: 95 & no sharpen (116KB)



jpeg quality: 60 & sharpen: 0.6 (25.1KB)

5.1 이미지 사이즈를 줄여야 한다.



윤영제

commented on 11 Feb

Author



quality 60 & sharpen 0.6 적용 시 기존 대비 사이즈

사진 5장 용량

- 기존: w750 5장 = 116KB * 5 = 580KB
- 개선: w560_sharpen 1장 + w278_sharpen 4장 = 62.3KB

약 89% 사이즈 절감 효과

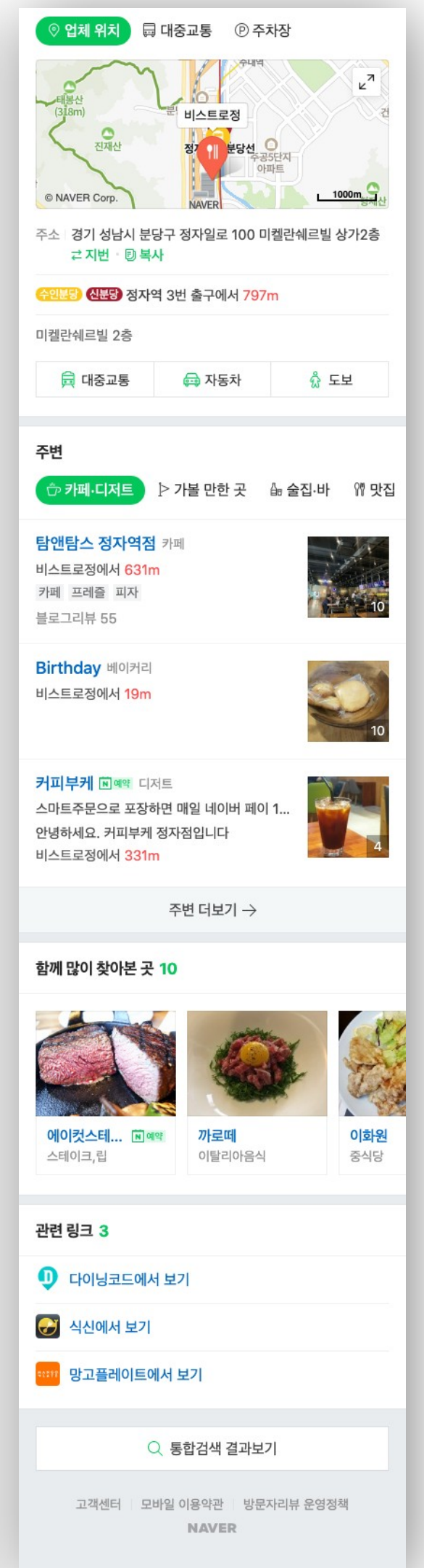
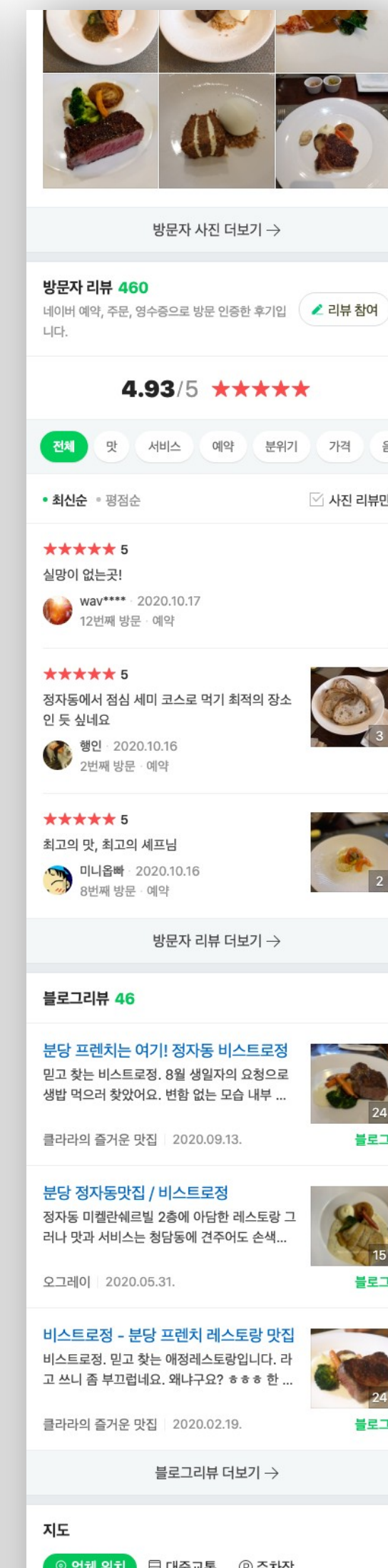
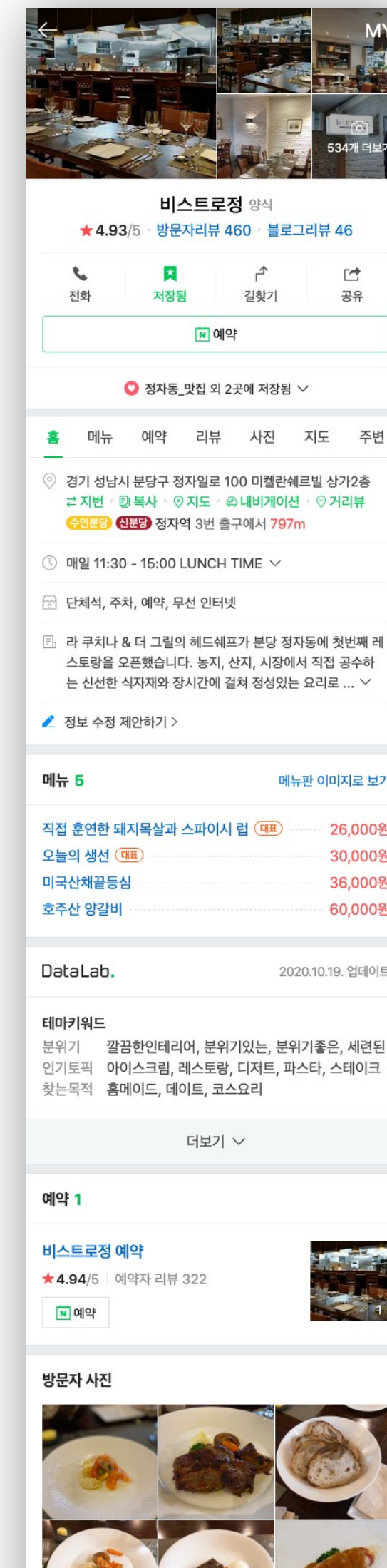
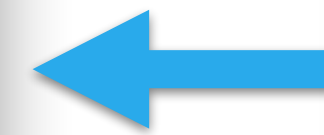


2

5.2 첫 로딩 사이즈를 줄여야 한다.

이미지 lazy loading & Component lazy loading

- 화면에 보이는 부분만 loading
- DOM 최적화 + 이미지 load 최적화
- react-lazyload 사용



5.2 첫 로딩 사이즈를 줄여야 한다.

Dynamic Loading

- code splitting - asset size 최적화
- `@loadable/component`
- react-router와 조합 -> path 별 resource 분리



**loadable
components**

5.2 로딩 속도를 빠르게

SSR(Server Side Rendering)인데 왜 화면이 늦게 보이지?

플레이스페이지 노출여부 제어는 지도에서 한다.

플레이스페이지 로드 완료는 message로 플레이스에서 지도로 전달

원인: message 전달 타이밍 이슈

- CSR(Client Side Rendering) 시점에 message 전달 -> SSR 이점 못살림
- SSR load 후 바로 로드 완료 message 전달로 수정

단점

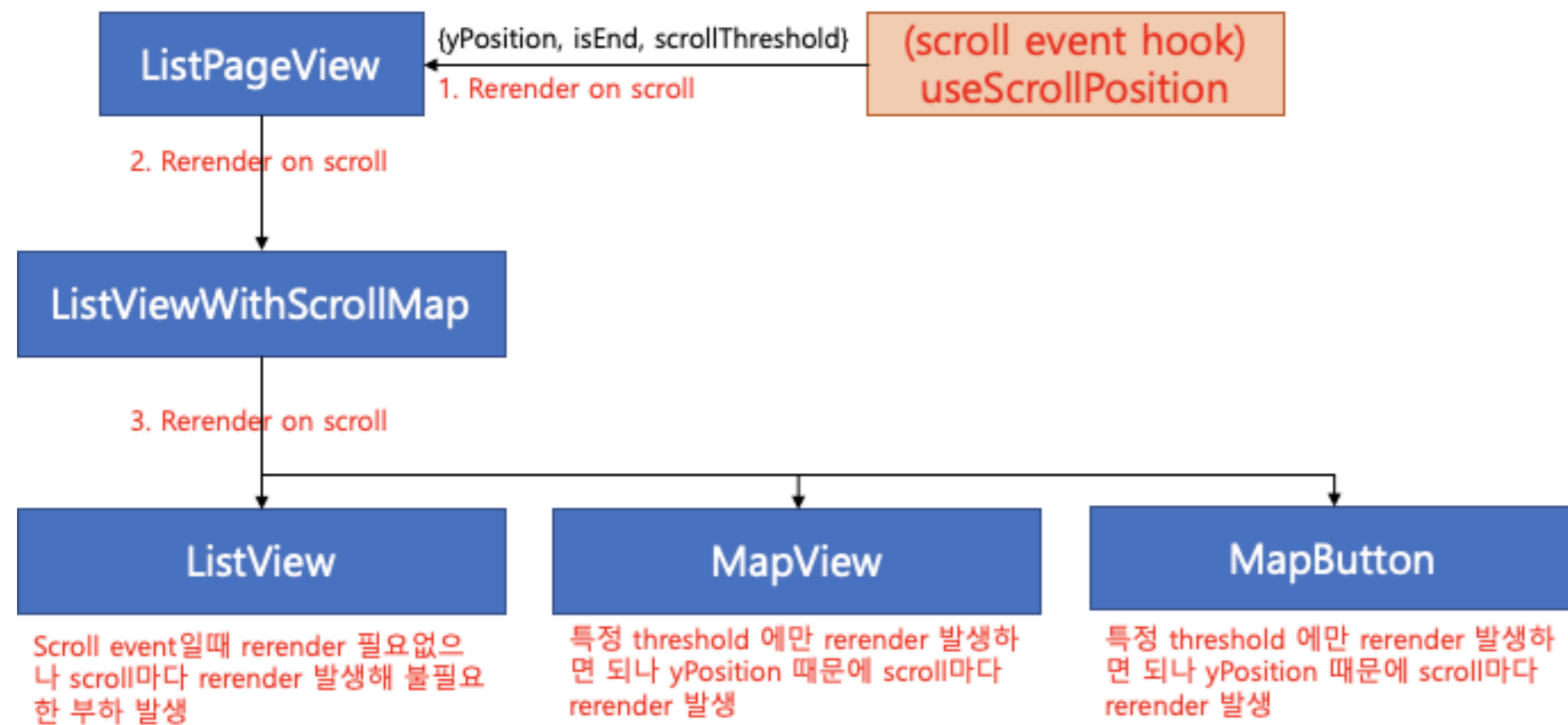
화면은 미리 보이지만 interaction은 CSR 후 가능.

SSR과 CSR 사이의 시간 < 사용자가 화면 인식 하는 시간

5.3 Rendering 최적화

스크롤 이벤트로 인한 React Component re-render 최적화

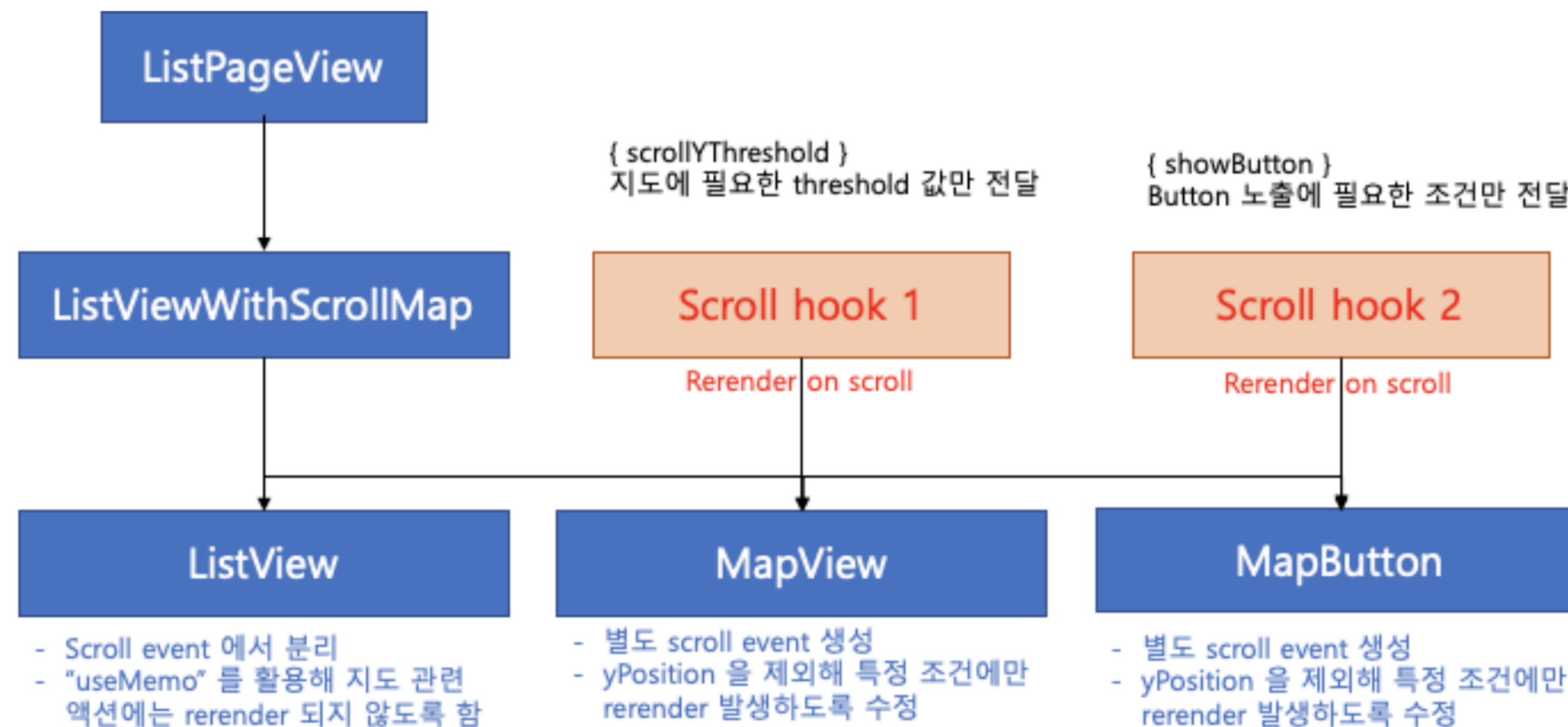
상위 Component에 걸쳐있는 scroll 이벤트로 인한 불필요한 re-render



5.3 Rendering 최적화

스크롤 이벤트로 인한 React Component re-render 최적화

component 별로 스크롤 이벤트 필요한 부분만 처리하는 hook 생성



5.4 스크롤 최적화

IntersectionObserver를 이용한 무한 스크롤 최적화

- 무한 스크롤 시 언제 추가 contents 로딩 할건지 확인

기존) scroll event + getBoundingClientRect() 사용

개선) `IntersectionObserver()` 사용 - `async non-blocking API`

장점

- frame 안정적. 저사양 폰에서 효과 좋음
- 코드 개선 효과

5.4 스크롤 최적화

IntersectionObserver를 이용한 무한 스크롤 최적화

The screenshot shows a GitHub Pull Request interface. At the top, the title is "[PR] Refactor infinite scroll using intersection observer #6234". Below the title, it says "Merged 김원준 merged 1 commit into develop from i5908 on 1 Jul". There are tabs for "Conversation 5", "Commits 1", "Checks 0", and "Files changed 1". A status bar shows "+33 -43" with a green bar. On the left, a comment by 김원준 from 3 Jun is visible. The "Issues" section lists "#5908". The "Details" section contains three bullet points, with the last two highlighted by red boxes. The right sidebar shows "Reviewers" (류효석, 윤영제) with green checkmarks, "Assignees" (김원준), "Labels" (공통), and "Projects".

[PR] Refactor infinite scroll using intersection observer
#6234

Merged 김원준 merged 1 commit into develop from i5908 on 1 Jul

Conversation 5 Commits 1 Checks 0 Files changed 1 +33 -43

김원준 commented on 3 Jun

Issues

- #5908

Details

- scroll 이벤트에서 해당 영역이 교차되었는지 판단하는게 아니라 IntersectionObserver 를 통해 판단합니다.
- 기존 scroll에 throttle이 걸려있어서 성능이 엄청 좋아지지는 않았으나 그래도 동일 스크롤시 frame이 좀 더 안정적으로 나타났습니다.
- 코드량이 줄어 좀 더 깔끔해졌습니다.

Reviewers

- 류효석 ✓
- 윤영제 ✓

Assignees

- 김원준

Labels

- 공통

Projects

6. 앞으로 해야 할 일

6. 앞으로 해야할 일

- 개발 후 확인해야 할 환경이 많다. -> 테스트 자동화 🤖
- Polyfill 관리 불편한데... -> 환경별 자동화 🤔
- 업종 추가/개선 📍
- 끊임 없는 Refactoring / 성능 개선 / 사용성 개선 ⚡



함께 하실 분들을 기다리고 있습니다. 🙋

dl_glance_dev_recruit@navercorp.com



Thank You 🤘



Q & A