



# 최종 발표

# 나 자신

나 자신을 아시나요?

”

발표 시작 >



# 발표 순서

”

**01** 팀원 소개 및 R&R

**02** 서비스 소개

**03** 서비스 기획 및 디자인

**04** 유저 스토리 & 아키텍처

**05** 인증 및 인가

**06** 프론트엔드 개발 포인트

• • ○

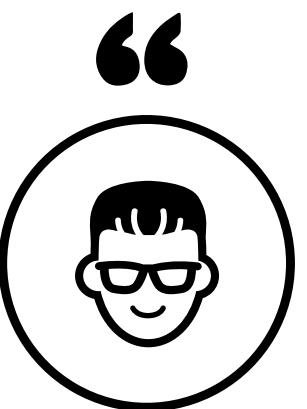
”

# 팀원 소개 및 R&R

01

# 01. 팀원 소개 및 R&R

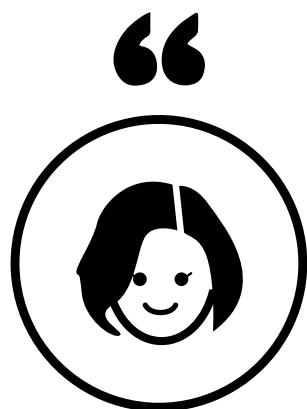
프론트 엔드 및 기획



**헨리**

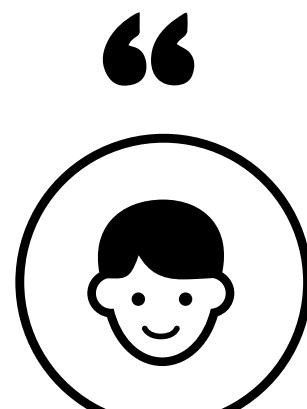
**팀장**

로그인, 우리들의 이야기 페이지  
프론트 인증/인가  
레이더 차트 그래프



**에바**

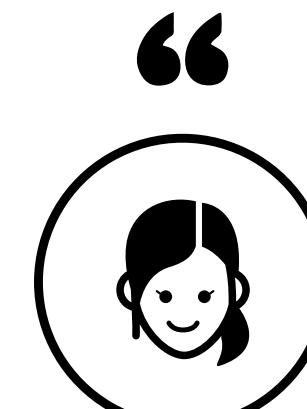
**백엔드 소통, 일정 관리**  
등록 페이지  
공통 컴포넌트



**앤디**

**문서 작업**

마이페이지  
마이 페이지 내 컴포넌트, 모달 포탈



**티나**

**디자인 소통**  
타인 등록하기 페이지  
공통 컴포넌트

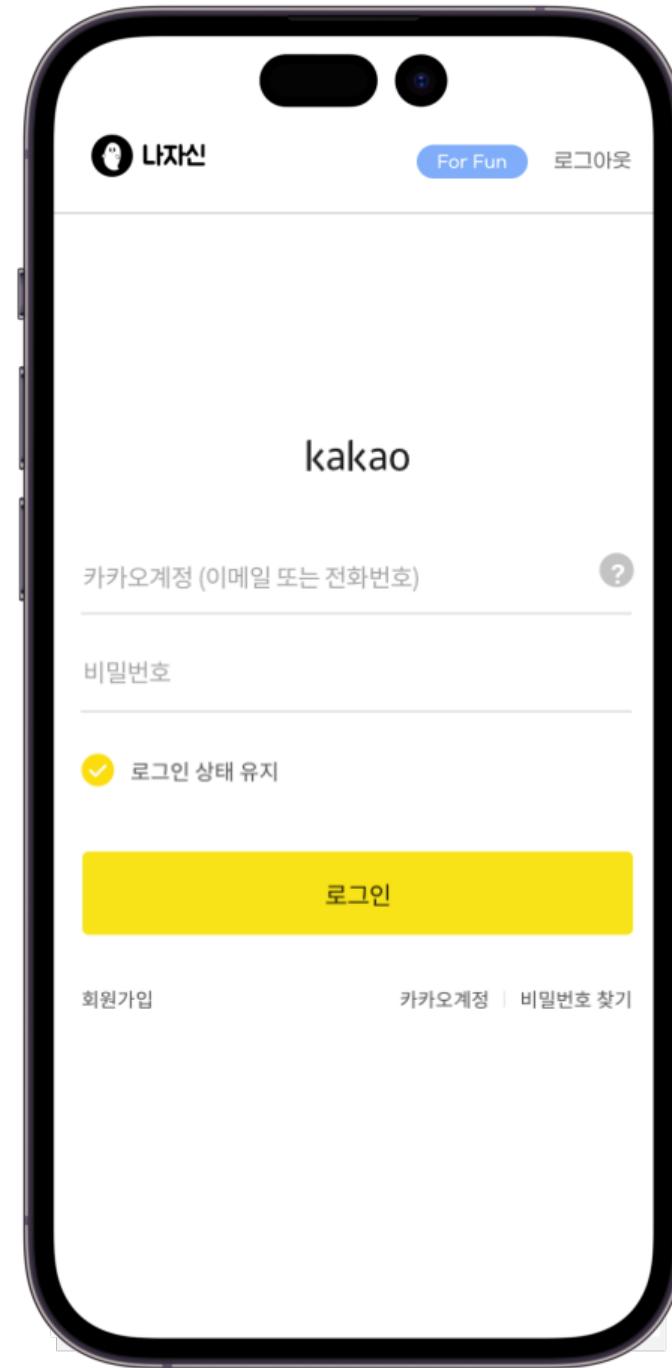
서비스 소개

02

”

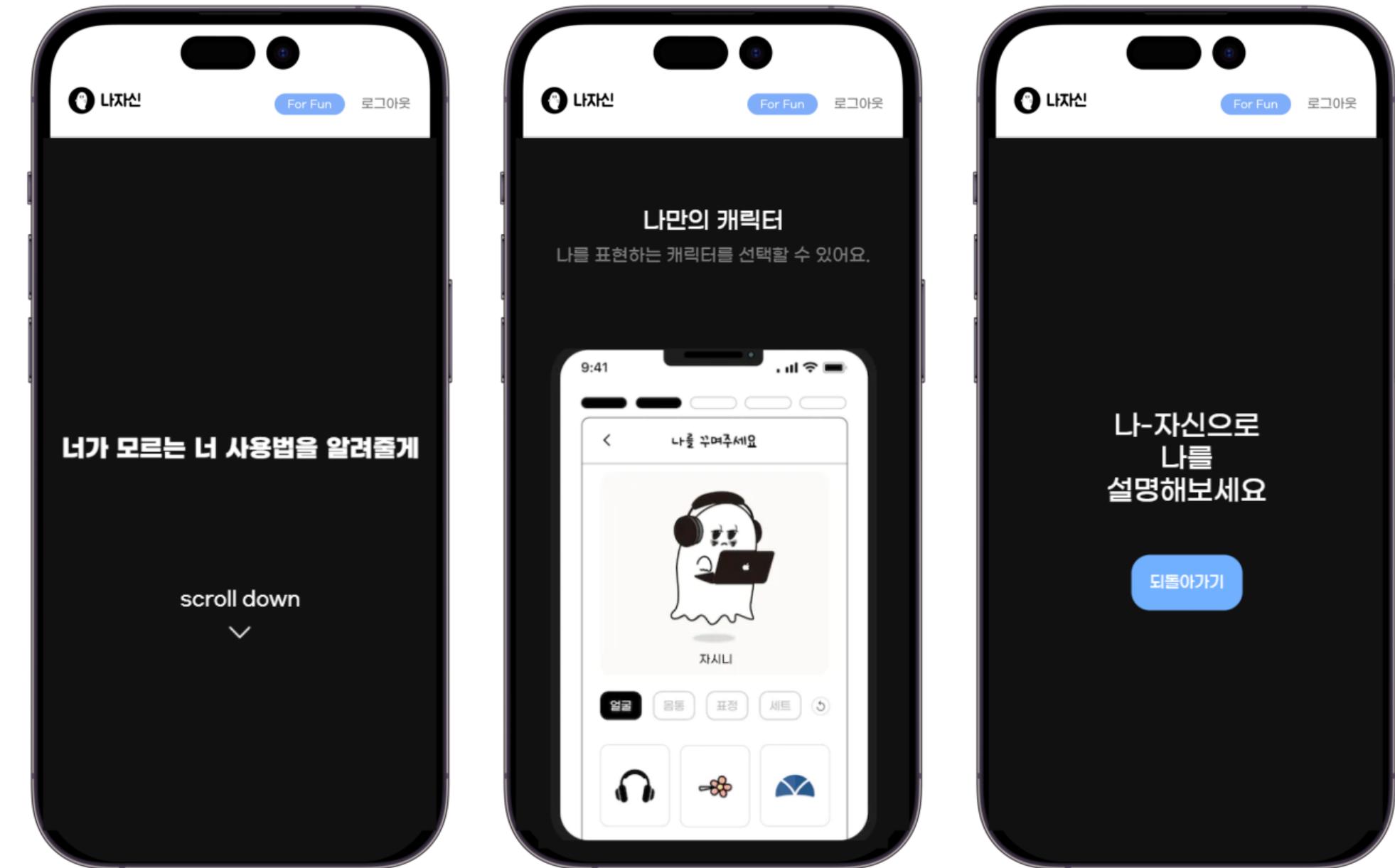
# 로그인 페이지

구글, 카카오 소셜 계정으로  
간편하게 로그인 가능



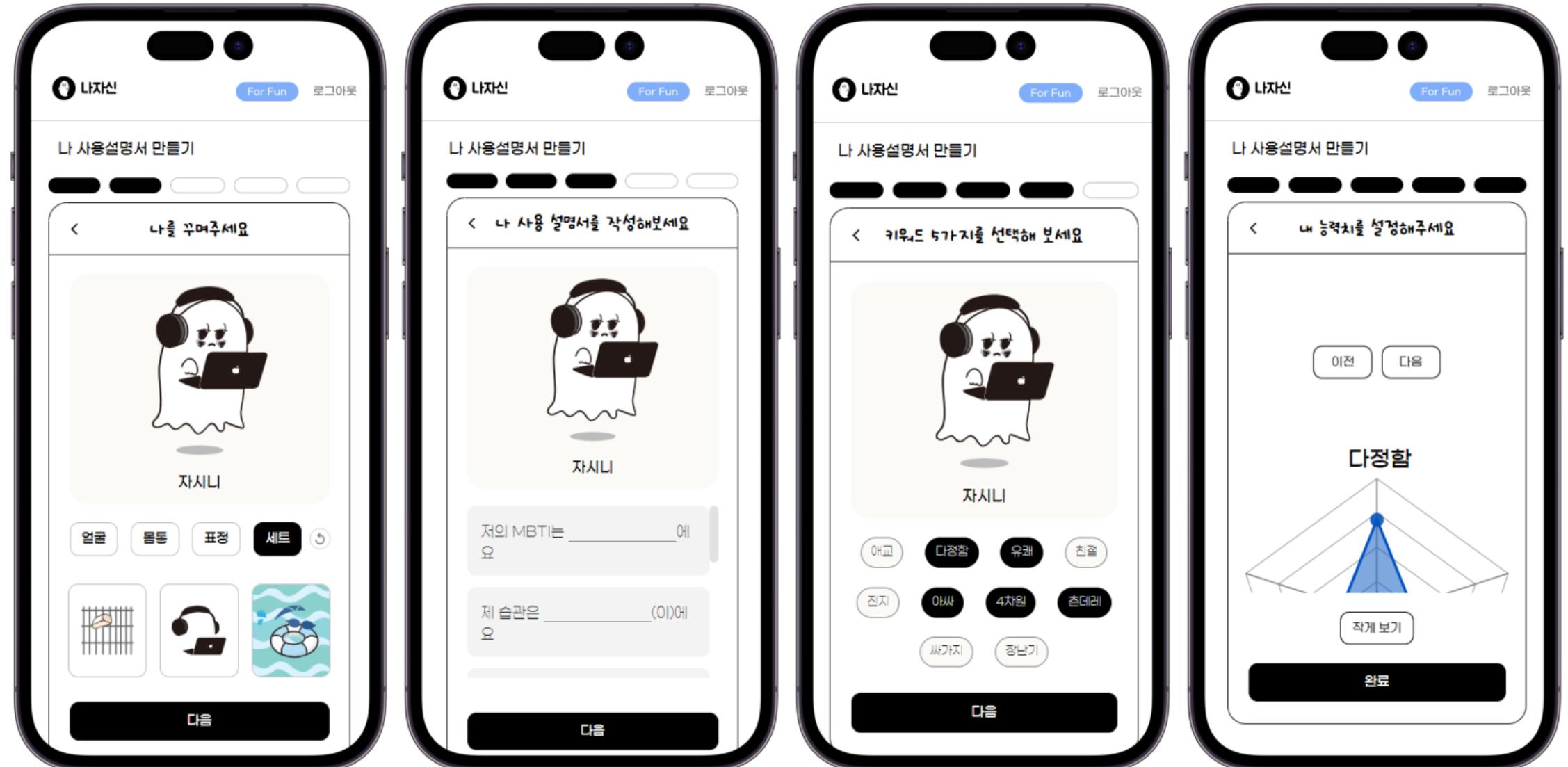
# 우리들의 이야기

서비스에 대한 설명과 함께  
로그인을 유도



# 등록하기

나를 대표하는 캐릭터,  
문장, 능력치 등을 설정하여  
**프로필을 등록**



# 타인 설명서 작성 페이지

공유받은 링크에서  
사용 설명서의 주인에 대한 **의견 작성**



# 마이페이지

다른 사람들에게 나의 설명서를  
받아볼 수 있는 **링크를 공유**

내 설명서와 다른 사람들이 작성해준  
**설명서를 비교** 가능



서비스 기획  
및 디자인

”  
03



# 나 자신의 시작?

우리는 유저 확보 경험을 하고 싶다!

앞선 2번의 프로젝트, 1번의 개인 프로젝트로 코딩 실력은 올렸고,  
특별히 새로운 기술을 적용할 수 없다면..  
우리에게 가장 필요한 것은 **USER!!!**

+@ 애니메이션 .. 테스팅 ..



# 나 자신의 시작?



## 바이럴이 될 수 밖에 없었던 서비스들!

아무런 요청, 부탁, 언급이 없었음에도 스스로 인스타그램 스토리에 받은 내용 or 프로필 화면+친구추가 링크를 올리는 것이 바이럴!

# 나 자신의 시작?



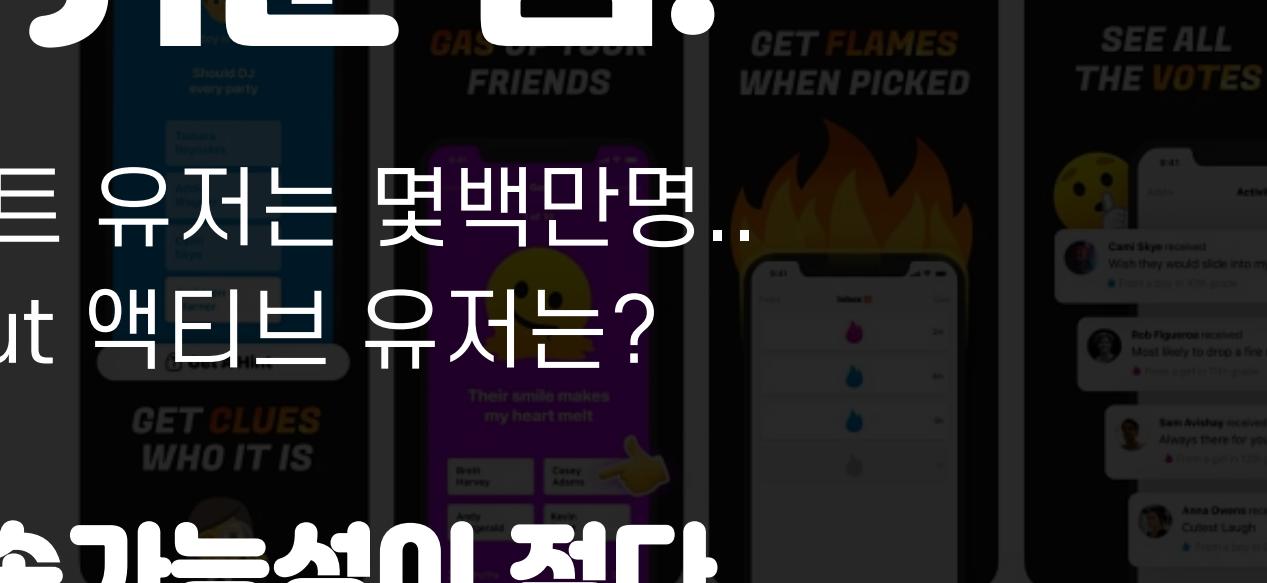
# 아쉬운 점!

게스트 유저는 몇백만명..  
But 액티브 유저는?

지속가능성이 적다

바이럴이 될 수 밖에 없었던 서비스들!

아무런 요청, 부탁, 언급이 없었음에도 스스로 인스타그램 스토리에  
받은 내용 or 프로필 화면+친구추가 링크를 올리는 것이 바이럴!



# 액티브 유저를 확보할 수 밖에 없는 서비스?

The screenshot shows the Specter app's main landing page. At the top, there are three large cards for '지원자' (Contributor), '기업' (Company), and '작성자' (Writer). Below these are two smaller cards for 'GET FLAMES WHEN PICKED' and 'SEE ALL THE VOTES'. At the bottom, there's a summary section with icons for '경력 인증 기반 객관적 평판' (Certified professional reviews based on experience), '인당 평균 3.8개 평판' (Average 3.8 reviews per person), and '2일 이내에 조회 가능' (Reviews available within 2 days). A blue button at the bottom right says '서비스소개서 신청' (Request service introduction). The overall theme is dark with purple and blue accents.

Specter

서비스 소개 이용 가이드

로그인 회원가입 도입문의

스펙터, 평판조회를 혁신하다

지원자

기업

작성자

지원한 기업에서 평판등록을 요청받았어요

우리 기업 지원자의 평판을 조회하고 싶어요

함께 일한 동료에게 평판작성을 요청받았어요

평판 등록하기

평판 조회하기

평판 작성하기

\*스펙터를 통한 평판조회 프로세스는 관련 법안을 모두 준수합니다.

경력 인증 기반 객관적 평판

인당 평균 3.8개 평판

2일 이내에 조회 가능

서비스소개서 신청

인재검증의 최신 트렌드

선진 채용문화를 가진 3,300개 이상 기업들이 스펙터와 함께하고 있습니다

지속적으로 사용하게 만드는 컨텐츠가 있어야 한다!

## 밖에 없었던 서비스들!

급이 없었음에도 스스로 인스타그램 스토리에 하면+친구추가 링크를 올리는 것이 바이럴!

# 나 자신의 시작?

## 의외로 전통 깊은 SNS 문화

싸이월드 시절부터 ~ 네이버 블로그인 현재까지  
이어져온 100문 100답!

젊은이들, 특히 SNS를 하는 사람으면  
메타인지에 흥미가 있다!

### [일상/생각] 나도 할래, 블로그 백문백답 (5)

백문백답 마지막!! 마지막이라 그런지 질문이 좀 더 심오해서 하기싫어졌…지만 열심히 해보도록 하죠 ㅎㅎ 81. 죽을 때 남기고 싶은 유언은... 100문 100답을 마무리하면서 느낀 점 생각보다 너무 재미있었고 또 생각보다 조회수가 많이 나와서 부끄러워용 >< #일상 #생각 #백문백답 #블로그...

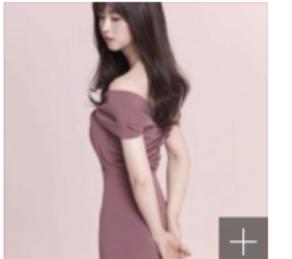
최점니 | Jimn2y's Diary | 2023. 1. 9.



### [허니와 소확행] 2022 백문백답으로 자존감 키우기 나를 아는 것이 인생의 힘!!!!

나를 위한 시간 "백문백답" 1. 이름 수지 2. 생년월일 11/23 3. MBTI ENTJ 4. 이상형 내향적, 차분한 사람 다정하고 낭만적인사람 자기 전문분야가... 백문백답 쓰는 이유 오랜만에 쉬는 날, 다시 한번 나를 되짚어보기 위해 99. 다 작성하고 할 일 국진이빵 마저 먹기 100. 100이 나에게 주는 의미 난...

Honey | 이미지|브랜딩|퍼스널컬러 허니의 블로그입니다 | 2022. 10. 1.



### 블로그 백문백답 - 나도 도전 !

안녕하세요 빈이媽이에요:) 잠도 안오고 고적고적 심심해서 백문백답을 가져왔어요 :) 첨부파일 블로그 100문 100답 양식 공유.txt 어언 10년도 더 된 옛날옛적(?) 싸이월드를 열심히 했던 그때 백문백답을 했었는데 ㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋ 오글거리긴 하지만.. 그때를...

빈이媽의일상 | 일상을 공유하자:)♥ | 2023. 8. 9.



+ 일촌맞기 + 만되기

TODAY 1215 | TOTAL 20201215

탱구의 100문 100답 2020ver.

DIARY

- What Do I Call You
- Playlist
- To the moon
- 들불 (Wildfire)
- Galaxy
- Happy

100문 100답 시작!

태연 2020.12.15 18:00 스크랩 8,939

61.하얀 마스크 vs 검정 마스크 9☆.....▶ 하얀마스크

62.스트레스 해소법 9☆.....▶ 껌을씹는다, 걷기, 화장품쇼핑

63.배워보고 싶은 것 9☆.....▶ 제빵, 영상편집

64.해보고 싶은 것 9☆.....▶ 남의 눈치보지 않고 당당하게 걷기

65.공포영화 vs 로맨스영화 9☆.....▶ 공포영화

66.콜라보 해보고 싶은 아티스트는 9☆.....▶ 태민, 태용, 태일

67.지금 핸드폰 기종은 9☆.....▶ 11프로 아이폰

68.맥시멈라이프 vs 미니멀라이프 9☆.....▶ 미니멀하고싶은 맥시멈라이프

69.좋아하는 날씨 9☆.....▶ 태풍 오기 전

70.하루 중 TV 보는 시간 9☆.....▶ 4시간

71.가고싶은 여행지 9☆.....▶ 전주

72.나의 단골집 9☆.....▶ 집게 앞 편의점, 뺑집, 피어싱가게

73.좋아하는 음료수 9☆.....▶ 매실음료,

74.오늘 차 안에서 먹은 간식은 9☆.....▶ 젤리!

75.좋아하는 과자 9☆.....▶ 꼬부기칩, 로투스

76.가장 최근 연락을 주고 받은 사람은 9☆.....▶ 엄마, 놀토피디님

77.존경하는 선생님 9☆.....▶ 이수만 선생님?

78.신곡 발표 후 가장 먼저 연락 온 멤버 9☆.....▶ 최민호

79.그 멤버가 남긴 멘트 9☆.....▶ 느낌이 오..너무 좋아 몸이 베베 괴일것 같아

80.생일 선물 중 가장 기억에 남는거 9☆.....▶ 팬들이랑 한 생파

총  
프로필  
다이어리  
사진첩  
게시판  
동영상  
방명록



# 나 자신의 시작?

당근마켓 김준비 개발자님의 메타인지 특강 中



그래 이거야!

## 나 사용법

- 성공에 대한 나의 생각
- 나에게 동기부여가 되는 것
- 나의 커뮤니케이션 스타일
- 혹시 내가 불편하게 할지도 모르는 것들
- 나의 신뢰를 얻거나 잃게 만드는 것들
- 나의 강점
- 나의 성장 방향
- 살면서 가장 성취감을 느꼈던 순간은?

### 가까운 거래경험팀의 "나 사용법"

사람은 누구나 보여지는 면과 보여주고 싶은 면이 있어요.

우리는 팀으로써 업무를 하다보면 자연스럽게 서로의 보여지는 면에 대해서는 파악하게 돼요.  
그리고 해당 동료를 내가 관찰한 보여지는 면으로 어떤 사람인지 판단해버려요.

반면에 내가 보여주고 싶은 면은 내가 직접 표현하지 않으면 동료가 알아차려주기가 쉽지 않아요.

생각보다 많은 불화는 바로 이 부분에서 발생해요.

내가 보여주고 싶은 면과 상대가 피악한 나의 보여지는 면이 불일치할 때, 서로 오해가 쌓이고 감정적인 충돌까지도 이어지곤 해요.

그래서 우리는 우리가 어떤 사람인지 스스로 소개할 필요가 있어요.

이름, 나이, 직군, 경력, 취미 등을 소개하는 형식적인 자기소개가 아닌, 동료로써의 나는 어떤 사람인지를 스스로 소개해요.

나는 어떤 부분에 예민(무던)한 사람인지, 커리어나 업무에 어떤 욕심이 있는지, 자신 있는 역량과 자신 없는 역량은 어떤 것인지, 내가 팀을 위해 어떤 부분을 채워줄 수 있고, 팀이 나를 위해 어떤 부분을 채워줬으면 좋겠는지 등에 대해 공유해요.

### 나 사용 설명서

아래 내용을 간단히 작성해 주세요! 팀원들에게 내가 일에 대해 어떤 가치관을 가지고 있는지를 공유하고 나와 협업할 때 어떤 부분을 신경써주면 좋을지 알리기 위함이에요.

- ▶ 성공에 대한 나의 생각
- ▶ 나를 동기부여하게 만드는 것
- ▶ 커뮤니케이션 방식
- ▶ 혹시 내가 불편하게 할지도 모르는 것들
- ▶ 나의 신뢰를 얻거나 잃게 만드는 것들
- ▶ 나의 강점
- ▶ 나의 성장 방향
- ▶ 일과 관련해 가장 성취감을 느꼈던 경험

메타인지가 뛰어난 사람들과 함께 일을 하면, 팀의 협업 능력과 퍼포먼스가 눈에 띄게 상승해요!



오지랖이 넓고, 관심 받는 것을 좋아하는 요즘 MZ의 니즈를 충족할 수 있는

## 나사용 설명서

### Just For Fun

모든 것의 시작은 재미여야하니까!

쉽게 공유하고, 쉽게 의견을 받을 수 있는 기능!

캐릭터 꾸미기, 나사용설명서 작성,  
다른 사람에게 나에 대해서 적어달라고 공유하자

### For Dev

개발자 블 시대! 개발자 취업에 관심이 있는 사람이라면,  
취준생들이 만들었다는 본 서비스에 관심을 가질 것!

- GitHub readme에 나사용 설명서 및 5 Stats Radder 그래프를 넣을 수 있도록 한다.
- 동료들에게 부탁해 나의 업무 스타일, 커뮤니케이션 스타일 등을 평가해 달라고 할 수 있는 서비스를 제공한다!
- 추후 평가 받기 기능 등을 추가해서 지속적으로 사용 가능하도록 한다!  
지속적으로 평가를 업데이트 해서 난 어떤 개발자인지 메타인지한다.
- 랜딩 페이지에서 DevFolio를 홍보한다.



오지랖이 넓고, 관심 받는 것을 좋아하는 요즘 MZ의 니즈를 충족할 수 있는  
**나사용 설명서**

## Just For Fun

모든 것의 시작은 재미여야하니까!

쉽게 공유하고, 쉽게 의견을 받을 수 있는 기능!

캐릭터 꾸미기, 나사용설명서 작성,  
다른 사람에게 나에 대해서 적어달라고 공유하자

## For Dev

개발자 븐 시대! 개발자 취업에 관심이 있는 사람이라면,  
취준생들이 만들었다는 본 서비스에 관심을 가질 것!

- GitHub readme에 나사용설명서 및 5 Stats Radder 그래프를 넣을 수 있도록 한다.  
**이후 추가 develop!**
- 동료들에게 부탁해 나의 업무 스타일, 커뮤니케이션 스타일 등을 평가해 달라고 할 수 있는 서비스를 제공한다!
- 추후 평가 받기 기능 등을 추가해서 지속적으로 사용 가능하도록 한다!  
지속적으로 평가를 업데이트 해서 난 어떤 개발자인지 메타인지한다.
- 랜딩 페이지에서 DevFolio를 홍보한다.



# 디자인 협업

figma를 적극 활용한 소통!

The screenshot shows a Figma workspace with a grid of mobile application screens at the top. Below this, several components are highlighted with callouts:

- 내가 하는 나 설명서**: A component showing a user profile and a large input field.
- 내적나사 1 (캐릭터)**: A character creation screen with a grid of options.
- 내적나사 2 (질문)**: A screen asking questions about the user's personality.
- 내적나사 3 (설정)**: A screen for setting personality traits.
- Frame 1140**: A sequence of five cards showing a flow from entering a nickname to selecting personality traits.

A sidebar on the right contains a message from a user named 신예진:

**신예진** 7 days ago  
이 페이지는 등록이기 때문에 해당 박스가 input입니다. 회색으로 동일하게 가기보다는 다른 디자인이 필요해요. focus도 들어가야합니다~ 폰트 스타일도 부탁드립니다!

Another message from 신예진:

**신예진** 7 days ago  
내적나사, 타적나사 처럼 등록인 다른페이지 다 동일한 수정사항입니다.

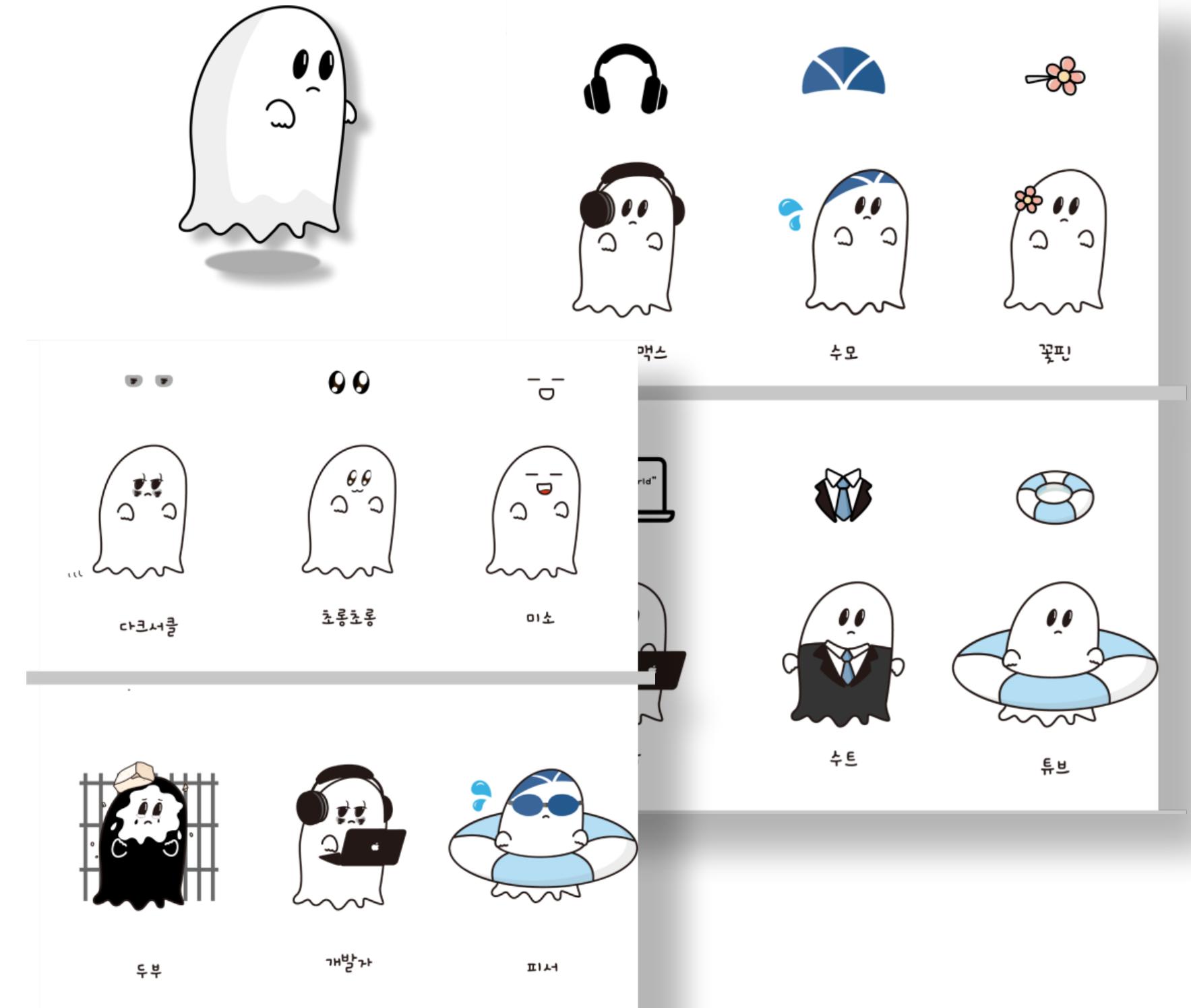
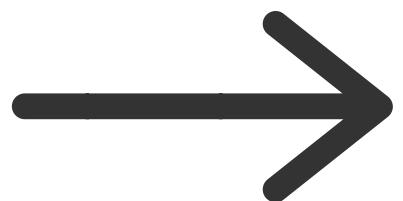
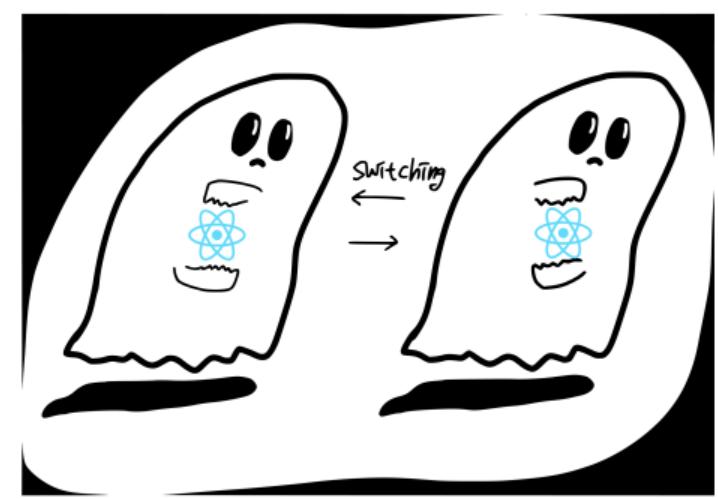
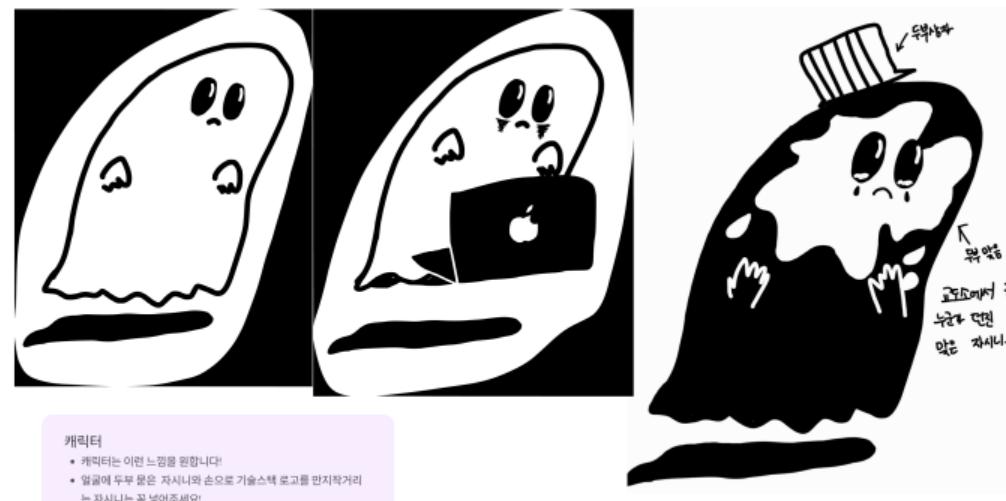
At the bottom, there is a "Reply" button and an upward arrow icon.

와이어 프레임 전달 후 구체화



서비스의 정체성을 위한 컨셉 대표 캐릭터 디자인

# 캐릭터 디자인





# 디자인 시스템

## Color

전체적으로 다이어리 느낌이  
날 수 있는 모노톤,  
강조를 위한 **포인트 컬러** 사용



## 반응형

어떤 기기에서도 사용이 원활  
하도록, PC, Tablet, Mobile  
반응형 UI로 구현

PC: 1200px 이상

tablet: 768px 이상 ~ 1199px 이하

mobile: 375px 이상 ~ 767px 이하

375px 미만 사이즈의 디자인은 고려하지 않음



## Font

가독성 좋고 귀여운 Gmarket  
Sans 폰트 사용  
강조할 텍스트는 HS유자체 사용  
크기와 두께에 따른 위계 부여

H1 Headline	Regular 24	HS유자체
H2 Headline	Bold 20	Gmarket Sans
H3 Headline	Medium 20	Gmarket Sans
H4 Headline	Bold 16	Gmarket Sans
Text1	Light 22	Gmarket Sans
Text2	Light 20	Gmarket Sans
Text3	Medium 20	Gmarket Sans
Text4	Medium 18	Gmarket Sans
Text5	Medium 16	Gmarket Sans
Text6	Medium 14	Gmarket Sans
Text7	Light 14	Gmarket Sans
Button 1	Medium 17	Gmarket Sans
Button 2	Medium 16	Gmarket Sans
Button 3	Light 16	Gmarket Sans
Button 4	Regular 14	HS유자체
Button 5	Bold 14	Gmarket Sans
Button 6	Medium 14	Gmarket Sans
Button 7	Regular 12	HS유자체
Button 8	Light 12	Gmarket Sans

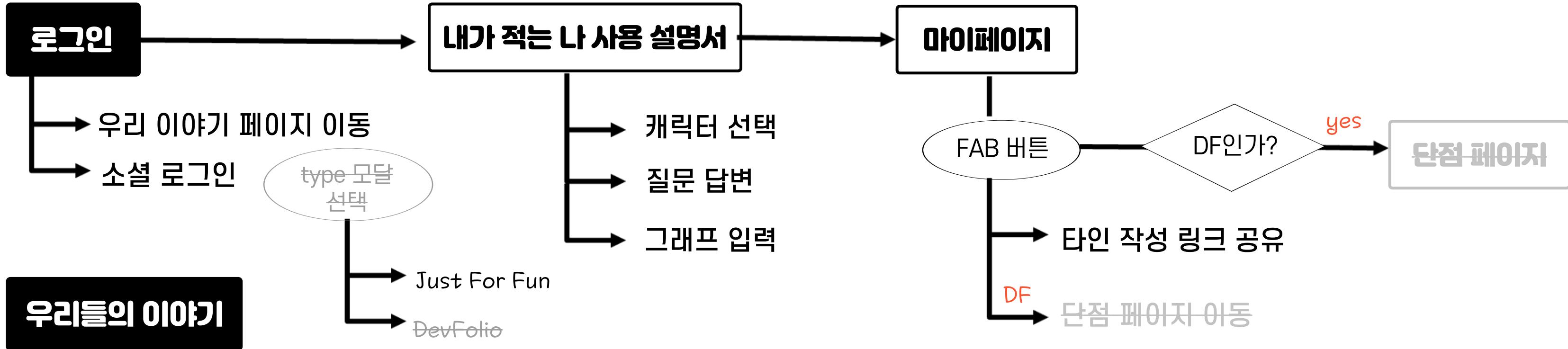
• • •

유저 스토리  
& 아키텍처

04

”

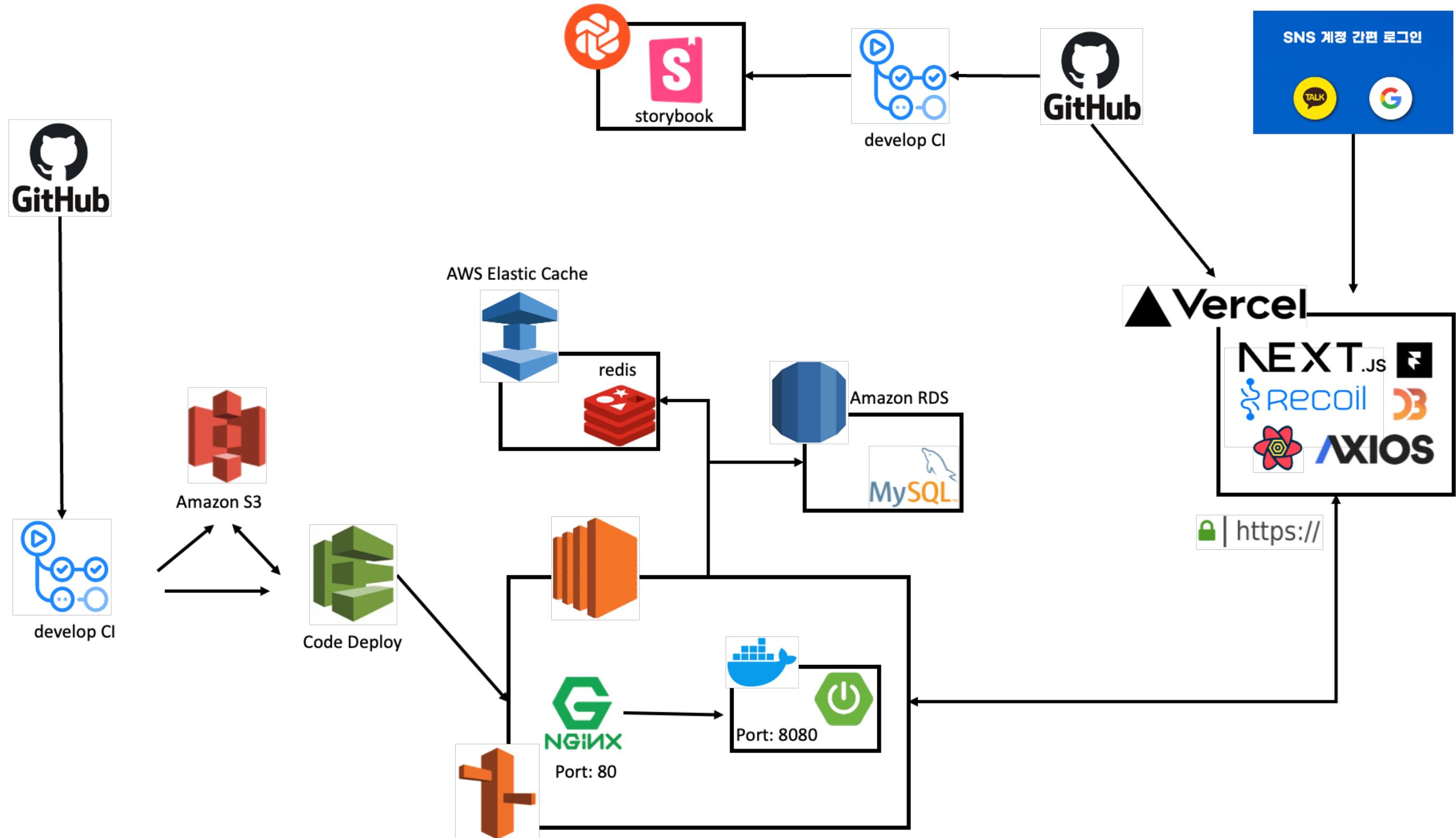
# • • ○ 유저 플로우



## 타인이 적는 나 설명 페이지

- 문답 입력
- 나 사용 설명서
- 그래프 수정
- 나도 해보기

# • • ○ 서비스 아키텍처



• • ○

인증 및 인가

”  
05

# 소셜 로그인 인증 플로우

유저 버튼 클릭 (프론트에서 api 호출)



백엔드에서 인가 코드 핸들링 & query string으로 토큰 응답



프론트 middleware에서 토큰 저장 및 페이지 라우팅



Authorization header에 엑세스 토큰 주입

각 요청마다 axios interceptor를 사용



로그아웃 시 토큰 제거 및 페이지 라우팅

• • ○

”

프로트 엔드  
개발 포인트

06

# Compound 디자인 패턴

## 01 장점

의존성을 분리한 독립적인 컴포넌트를 만들 수 있다.  
잦은 변경에 대응 가능하다.

## 02 단점

불필요한 컴포넌트가 많아질 수 있다.  
컴포넌트를 독립적으로 만드는 건 쉽지 않다.



```
<component>
  <component.sub1
  /><component.sub2
  /><component.sub3
  />...
</component>
```



# 풀더 구조



`/store` → 전역 상태 관리

`/types` → 타입 관리

`/components` → 컴포넌트 관리

`/hooks` → 상태 로직 관리

`/helpers` → 유тил 함수 관리

`/adaptors` → api interceptor, 웹 소켓 등 외부 데이터 연결 통로

`/services` → 데이터 받기 위한 api 호출, 웹 서비스 호출

# 브랜치 전략

## 01 Trunk-based development

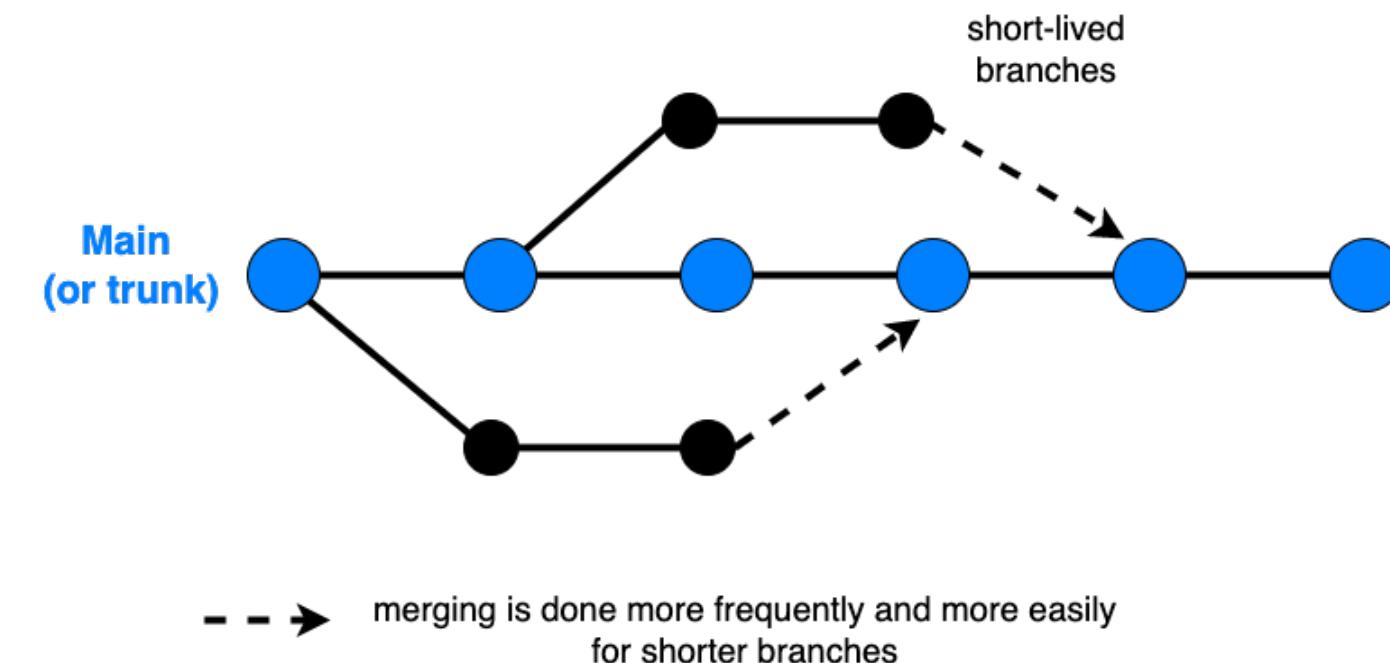
항상 배포 가능한 main 브랜치를 두고,  
짧은 수명을 가진 feature 브랜치를 운용하였다.

## 02 Squash merge

여러개의 커밋을 하나의 커밋으로 합친 후 merge한다.

Trunk-based development

StatusNeo





# 스토리북

## 01 스토리북을 활용한 컴포넌트 문서화

범용적으로 사용되는 공통 컴포넌트의 다양한 쓰임세를 스토리북으로 문서화

vercel (bot) deployed to Preview 3 weeks ago

View deployment

github-actions (bot) commented 3 weeks ago

storybook: <https://64d231ce4fe0c154eb235ed7-fwtffhdbv.chromatic.com/>

Andy-Shin99의 storybook 배포 #249

Summary

Jobs

storybook

Run details

Usage

Workflow file

storybook

succeeded yesterday in 2m 23s

- > Set up job
- > checkout repository
- > cache dependencies
- > dependency install
- > publish to chromatic
- > comment PR
- > Post cache dependencies
- > Post checkout repository
- > Complete job

## 02 chromatic과 github action을 활용한 스토리북 자동 배포

pr마다 스토리북도 자동으로 배포해 공유하기 때문에 타인이 만든 컴포넌트를 사용할 때 편리

- [ItemBox](#)
- [DESCRIPTION CARD](#)
- [FAB](#)
- [FOOTER](#)
- [GNB](#)
- [MODALLAYOUT](#)
- [CIRCLEBTN](#)
- [COMMONBTN](#)
- [INPUT](#)
- [CONFETTI](#)
- [COPYTOAST](#)
- [EDITBTN](#)
- [FORMBOX](#)
- [GHOSTBTN](#)
- [GHOSTCURSOR](#)
- [ITEMBTN](#)
- [KEYWORDBTN](#)
- [SHARED](#)
- [RADARCHART](#)
- [RESETBTN](#)
- [SIMPLELAYOUT](#)



# useFunnel custom hook

내부적으로 `next/router`에 의존하고 있어,  
`next/navigation`으로 마이그레이션이 필요한 `next 13` 버전에서는  
`@toss/use-funnel`의 기능을 사용할 수 없다.

custom해서 사용하자!

## 필요한 기능

1. Step에 따라 렌더링되는 기능
2. 뒤로가기 할 시, 이전 Step으로 가는 History 기능

[Feature]: @toss/use-funnel 의 NextJS 13 app directory 지원 #228

Open 1 task done CodePsy-2001 opened this issue on Apr 2 · 0 comments

CodePsy-2001 commented on Apr 2 · edited

**Package Scope**

Add to an existing package

Package name: `@toss/use-funnel`

**Overview**

내부적으로 `next/router`에 의존하고 있어, `next/navigation`으로 마이그레이션이 필요한 App Directory 버전에서는 `@toss/use-funnel`의 기능을 사용할 수 없습니다.

참고자료1: <https://stackoverflow.com/questions/74421327/nextrouter-was-not-mounted-next-js>  
참고자료2: <https://nextjs.org/docs/messages/next-router-not-mounted>

**Describe the solution you'd like**

`appdirectory` 경로에서 App Directory 호환 버전을 불러올 수 있도록 합니다.

16



# useFunnel custom hook

```
// 단계별로 컴포넌트를 렌더링해주는 Funnel 컴포넌트
function Funnel<T extends readonly string[]>({
  step,
  children,
}: IFunnelProps<T>) {
  // 유효한 자식 요소 필터링
  const validElement = Children.toArray(children).filter(isValidElement)
  // 현재 단계와 일치하는 Step 컴포넌트를 찾음
  const targetElement = validElement.find(
    (child) => (child.props as IStepProps<T>)?.name === step,
  )
  // 일치하는 Step 컴포넌트가 없으면 null 반환
  if (!targetElement) {
    return null
  }

  return <>{targetElement}</>
}

// 단순히 자식 요소들을 렌더링해주는 Step 컴포넌트
function Step<T extends readonly string[]>({ children }: IStepProps<T>)
{ return <>{children}</>
}
```

step에 따른 렌더링 가능

## Funnel 컴포넌트

단계별로 컴포넌트를 렌더링해주는 Funnel 컴포넌트

유효한 자식 요소 필터링해 현재 단계와 일치하는 Step 컴포넌트를 찾는다.

일치하는 Step 컴포넌트가 없으면 null 반환한다.

## Step 컴포넌트

단순히 자식 요소들을 렌더링해주는 Step 컴포넌트이다.



# useFunnel custom hook

```
export const useFunnel = <T extends readonly string[]>(
  steps: T,
  defaultStep: T[number],
) => {
  const [step, setStep] = useState(defaultStep)

  // Funnel 컴포넌트와 함께 Step 컴포넌트를 반환하는 객체를 생성
  const FunnelElement = useMemo(
    () =>
      troble shooting!
      Object.assign(
        (props: Omit<IFunnelProps<T>, 'step'>) => (
          <Funnel step={step} {...props} />
        ),
        { Step: (props: IStepProps<T>) => <Step<T> {...props} /> },
      ),
    [step],
  )

  const pathname = usePathname()
  const searchParams = useSearchParams().get('step')
  const router = useRouter()
```

```
useEffect(() => {
  if (step === defaultStep) {
    return
  }
  if (searchParams) setStep(searchParams)
}, [defaultStep, searchParams])

useEffect(() => {
  router.push(`/${pathname}?step=${step}`)
}, [pathname, router, step])

useEffect(
  () => () => {
    setStep(defaultStep)
  },
  [defaultStep],
)

const goNext = () => {
  const idx = steps.indexOf(step)
  if (idx === steps.length - 1) return
  setStep(steps[idx + 1])
}

const goPrev = () => {
  const idx = steps.indexOf(step)
  if (idx === 0) return
  setStep(steps[idx - 1])
}

// FunnelElement와 현재 단계를 설정할 수 있는 setter를 튜플로 반환
return { Funnel: FunnelElement, step, setStep, goNext, goPrev } as
const
```

history 가능

trouble shooting!

# useFunnel custom hook

문제 상황 : Infinite Depth.

## Unhandled Runtime Error

Error: Maximum update depth exceeded. This can happen when a component repeatedly calls setState inside componentWillUpdate or componentDidUpdate. React limits the number of nested updates to prevent infinite loops.

## Source

```
hooks/useBreakpoint.hooks.ts (15:4) ↗  
@ setMounted  
  
13 | const value = useMediaQuery(settings);  
14 | useEffect(() => {  
15 |   setMounted(true)  
|   ^  
16 | }, []);  
17 |  
18 | return mounted ? value : false
```

Show collapsed frames

```
// Funnel 컴포넌트와 함께 Step 컴포넌트를 반환하는 객체를 생성  
const FunnelElement = useMemo(  
  () =>  
    Object.assign(  
      (props: Omit<IFunnelProps<T>, 'step'>) => (  
        <Funnel step={step} {...props} />  
      ),  
      { Step: (props: IStepProps<T>) => <Step<T> {...props} /> },  
      [step],  
    )
```

useMemo를 사용해 step이 변경되었을 때만 객체를 새로 생성!



step 상태가 변경될 때마다 객체가 새로 생성되면서 Funnel을 사용중인 모든 컴포넌트들이 리렌더링된다?



# react hook form

## 비제어 컴포넌트, 데이터 한곳에서 관리

제어 컴포넌트 방식으로 input value를 관리하게 되면 렌더링 이슈가 발생!

비제어 컴포넌트 방식으로 관리하게 되면 동기화를 위해 props drilling이 필연적!



비제어 컴포넌트 방식으로 렌더링 이슈 해결,  
데이터는 계속 동기화되며 분산되지 않아  
데이터를 한 곳에서 관리 가능

그외 간편한 invalid 등

for문을 돌려도 register key를 지정해 invalid를 각각 할 수 있다.

```
const { handleSubmit, register, formState } = useForm<IFormData>()

<MyDescriptionCard
  key={data.id}
  question={data.question}
  defaultValue={data.answer}
  register={register(`answers.${data.id}`, validationRules)}
  isValid={formState && formState.isSubmitted
    ? !!formState.errors.answers?.[dataId]
    : undefined
  }
/>
```



# Framer Motion

리액트 전용 애니메이션 라이브러리



```
const { scrollYProgress } = useScroll({
  target: targetRef,
  offset: ['end end', 'end start'],
})

const y = useTransform(scrollYProgress, [0, 0.5], [0, -75])
const opacity = useTransform(scrollYProgress, [0, 0.3], [1, 0])
```

간단한 애니메이션을 적용하기 쉽고, 특히 **스크롤 애니메이션 툴**이 잘 되어 있다.  
**서비스 소개 페이지**에서 스크롤 중심의 애니메이션을 구현하기 위해 적용하였다.

# D3.js

## 차트 그래프 라이브러리



```
svg
  .selectAll("area")
  .data([dataValues])
  .enter("")
  .append("")
  .attr("class", "")
  .style("stroke", "")
  .on("mouseover", () => {})
```

레이더 차트에 커스터마이징(조건에 따라 그래프 보여주고, 이벤트로 데이터 추출)이 많이 필요하다.  
구현 시 svg 조작이 필요하여 **svg 관련 다양한 핸들러를 제공해주기 때문에 선택하였다.**



# 상태관리

## 01 Tanstack Query

서버 상태 관리를 위해 Tanstack Query의 **dehydrate**를 사용하여 서버 컴포넌트에서 가져온 데이터를 **prop drilling** 없이 클라이언트로 주입

## 02 Recoil

컴포넌트의 관심사를 나누다 보면  
자연스럽게 컴포넌트 구조가 복잡성 높아져서  
**recoil**을 사용하여 불필요한 **prop drilling** 방지

```
export default async function Page() {
  const queryClient = getQueryClient()
  await queryClient.prefetchQuery(['key'], () => getRadarData())

  const dehydratedState = dehydrate(queryClient)

  return (
    <QueryHydrate state={dehydratedState}>
      // ...
    </QueryHydrate>
  )
}
```

서버 컴포넌트에서의 **dehydrate** 예시

# Mocking

Next.js api를 mock api로 사용

```
● ● ●  
svg  
  .selectAll("area")  
  .data([dataValues])  
  .enter("")  
  .append("")  
  .attr("class", "")  
  .style("stroke", "")  
  .on("mouseover", () => {})
```

msw를 사용해서 백엔드 api 개발 전까지 mock 데이터를 사용하려 했으나 **app router 지원하지 않는 문제**가 생겼다.  
해결책으로 **next.js api를 mock api로 대체**하여 백엔드 api 개발 완료 전에 프론트 개발이 밀리지 않도록 하였다.



# React Portal

포탈 사용을 간편하게 해주는 라이브러리

```
● ● ○  
return mounted ? (  
  createPortal(modal, document.getElementById('modal-root') as HTMLElement)  
) : (  
  <></>  
)
```

모달과 로딩 컴포넌트에 리액트 포탈을 적용하여

부모 컴포넌트의 DOM 계층 구조 바깥에 있는 DOM 노드에 자식을 렌더링 할 수 있다.

모달의 경우 포탈로 컴포넌트 구조 내에서 완전 분리를 통해 얻을 수 있는 장점이 있다.

● ● ○

시연

”  
07

지금까지 나자신 팀의  
발표였습니다 !

”

발표에 대한 질의응답은 언제나 환영입니다 :)

발표 끝 !