

I'm Frontend Developer



PortFolio 사이트 만들기

☞ 소개글	Next.js 14를 활용한 포트폴리오 웹 사이트입니다.
🕒 진행기간	2024. 04 - 2024. 05
☰ 사용 기술	Framer Motion Next.js React Tailwind CSS TypeScript

🔗 Link

프론트엔드 개발자 김정수입니다.

3년차 프론트엔드 개발자 김정수 포트폴리오입니다.

☞ <http://www.jungsookim-portfolio.com>



프로젝트 소개



8:05

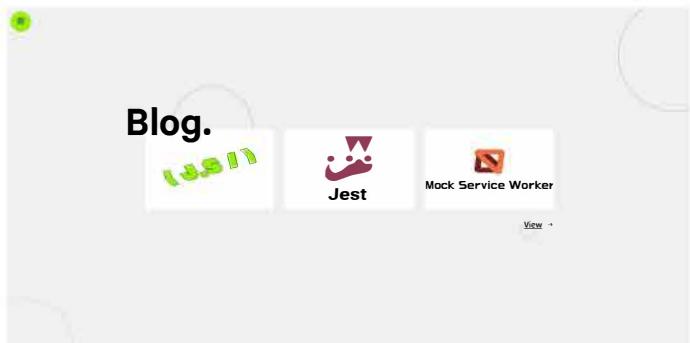
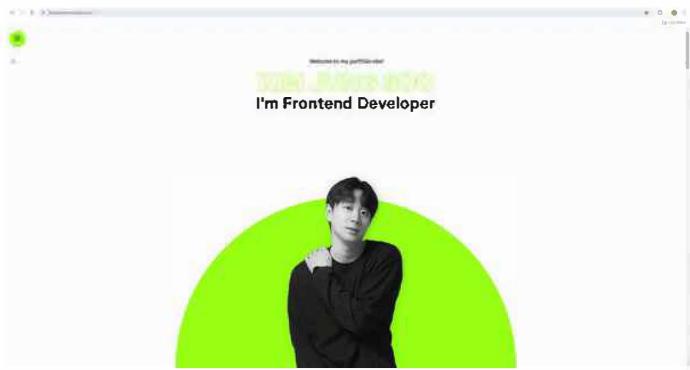
LTE 47%

Welcome to my portfolio site!

KIM JUNG SOO I'm Frontend Developer



jungsookim-portfolio.com



- **Next.js**와 **Tailwind CSS**를 깊게 공부하기 위하여 만든 저의 포트폴리오 사이트입니다.
- Tailwind CSS를 더 깊게 공부해보기 위해 반응형으로 제작하였습니다.
- Interactive UI를 공부해보고자 **Framer motion**을 사용하여 간단한 인터렉션을 구현했습니다.
- Vercel에 배포를 하고 가비아에서 도메인을 구매해 도메인 연동을 하였습니다.

사용 기술

- **Next.js**: 포트폴리오 사이트를 만들고 구글 검색 순위에 높이 올리고자 SEO에 유리한 프레임워크가 필요했고 그에 따라 Next.js를 채택하였습니다.
- **Tailwind CSS**: CSS의 라이브러리인 Tailwind CSS를 공부해보고자 도입하였습니다.
- **Framer Motion**: Interactive 한 웹 UI를 공부해보고자 Framer motion을 도입 하였습니다.

트러블 슈팅

SNS 배포 링크 공유 시 이미지가 출력되지 않는 이슈

문제 배경

프로젝트를 완료하고 SNS에 나의 포트폴리오 링크를 공유했을 때 내가 설정한 메타 이미지가 노출되지 않는 현상이 발생 했습니다.

해결 방법

```
export const metadata: Metadata = {
  ...,
  openGraph: {
    title: '프론트엔드 개발자 김정수입니다.',
    description: '3년차 프론트엔드 개발자 김정수 포트폴리오입니다.',
    url: 'https://www.jungsookim-portfolio.com',
    siteName: "Jungsoo's PortFolio",
    images: [
      {
        url: '/images/opengraph.png', //이 부분이 문제
        alt: '김정수 프로필 사진',
        width: 800,
        height: 600
      }
    ]
  }
};
```

이미지의 url를 만들 때 상대경로를 설정해서 출력이 되지 않았습니다. [공식문서](#)에 따르면 openGraph의 이미지는 절대경로로 변경 후 해결하였습니다!

결과

이 글 링크에서도 볼 수 있듯이 openGraph의 이미지가 정상적으로 출력되었습니다.

그 뿐만 아니라 이번 시간을 통해 **Meta Data에 대한 기본적인 정보를 공부할 수 있던 시간이 되었고, SEO에 유리한 Meta 데이터를 설정할 수 있게 되었습니다.**

Tailwind CSS에서 class 충돌으로 CSS가 안먹히는 이슈

문제 배경

Tailwind CSS를 처음 사용하면서 같은 CSS 속성을 변경하는 클래스가 두 개 이상일 때 제대로 동작하지 않고 먼저 있던 클래스가 반영되는 현상이 있었습니다. 빌드가 되면 Tailwind에서 정의된 CSS 순서대로 빌드가 되는데, 그렇게되면 cascading 방식에 따라 의도하지 않게 css가 결정되기 때문이었습니다.

해결 방법

```
import { ClassValue, clsx } from 'clsx';
import { twMerge } from 'tailwind-merge';

export const cn = (...inputs: ClassValue[]) => {
  return twMerge(clsx(inputs));
};
```

`tailwind-merge` 라이브러리를 활용하여 해결했습니다. `tailwind-merge` 라이브러리는 tailwindCSS의 class 충돌을 해결해주는 라이브러리입니다.

이와 더불어 `clsx` 라이브러리와 결합하여 사용하게 되면 클래스의 충돌 없이 tailwind의 클래스를 조건 부로 결합이 가능합니다. (shadcn에서 영감 받았습니다.)

결과

- tailwind 사용 시 발생할 수 있는 클래스 우선순위 문제에 대해 직접 겪으면서 tailwindcss를 더 주의 하여 사용할 수 있었습니다.
- class를 바로 사용하게되면 코드가 줄바꿈 없이 무자비(?)하게 길어지는 이슈를 커스텀 함수를 통해 해결할 수 있었습니다.

후기

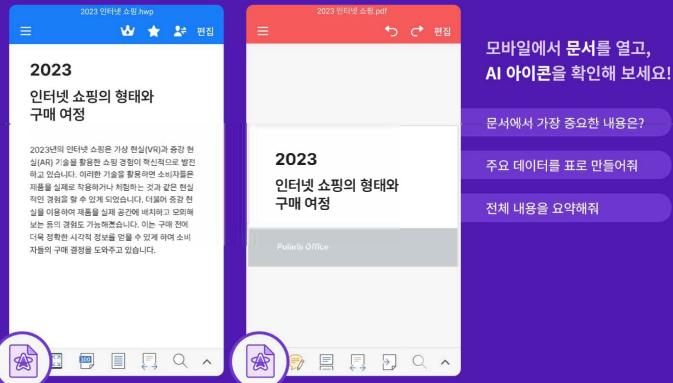
인터렉티브 웹 개발을 처음 짹먹해본 듯 합니다. 이번 기회로 Framer motion 라이브러리를 처음 사용해 보았는데, 사실 비슷한 동작들을 사용하기만 해서 잘 쓴 것 같지는 않습니다. ㅠㅠ 이 사이트는 제 포트폴리오 사이트이긴 하지만 인터렉티브 웹 개발을 위해 **이것저것 실험해볼 예정입니다..**

하나씩 동작을 추가해보면서 인터렉티브한 개발을 고도화시키는게 최종 목표입니다.

또한 Tailwind CSS에 대한 장단점도 느낄 수 있었던 프로젝트였던 것 같습니다. 제가 생각한 장점은 Tailwind CSS를 통해 일일이 CSS를 작성할 필요 없이 **미리 정의된 CSS를 클래스를 통해 사용**하면 되기 때문에 편하고 무엇보다 **스타일링면에서는 통일성을 가질 수 있어 협업에 굉장한 이점을 누릴 수 있다**고 느꼈습니다. 하지만 반대로 **tailwind에서 직접 정의한 CSS를 외워야 한다는 부분과 마크업에 class 가 비대해진다**는 단점도 있는 것 같습니다.

ASK Doc

길고 복잡한 문서, 주요 내용만 보고 싶다면?
문서와 채팅하며 필요한 정보만 간편하게!



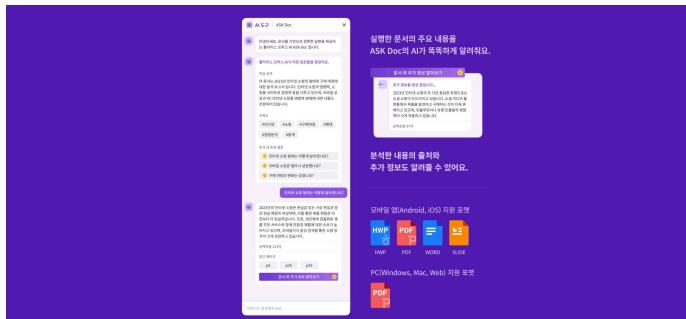
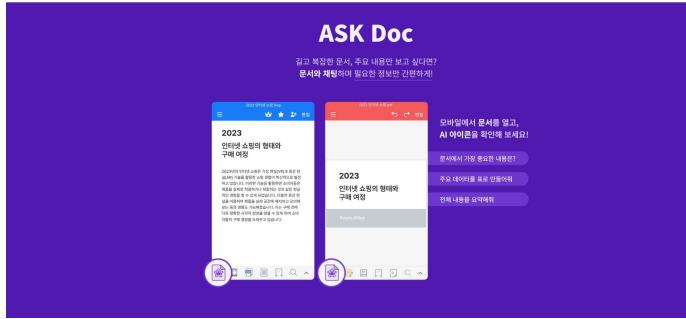
AskDoc Frontend 개발

소개글	폴라리스 드라이브 사용자를 위한 PDF AI 문서 분석 프로그램
진행기간	2023.10 - 2023.12
사용 기술	GA4 MSW React Redux Styled Component Typescript Webpack

🔗 How to Check

- [폴라리스오피스 공식 홈페이지](#)에 접속 후 로그인
- 드라이브 페이지에서 1000자 이상의 pdf 파일 열기
- AskDoc으로 문서 분석하기 버튼 클릭
- 우측 패널에 열려진 웹뷰 확인

프로젝트 소개



하는 경우에 유용하다. 또한, 끌뷰는 이미지 용량을 줄여주는 압축프로그램으로, 웹사이트의 로딩속도에 영향을 미치는데 도움을 준다. 티스워드는 티스토리 사용자를 위한 무료 웹사이트로, 다양한 기능을 제공하며 블로거들에게 유용하다. 미리캔버스는 블로그 썸네일 작성률을 도와주는 웹사이트이고, 픽사베이는 상업적으로 이용이 가능한 무료이미지를 제공해주는 사이트이다.

키워드

#Onetab #끌뷰 #티스워드
#미리캔버스 #픽사베이

문서 내 추천 질문

Onetab의 주요 기능은 무엇인가?
끌뷰를 사용하는 이점은 무엇인가?
티스워드의 주요 기능은 무엇인가?

끌뷰를 사용하는 이점은 무엇인가?

내용을 생성 중입니다...

Stop

대화 1회당 4크레딧이 차감됩니다.



- 폴라리스오피스 드라이브에 저장되어져 있는 PDF를 AI가 분석하여 채팅 형식으로 내용을 유추할 수 있는 "Ask Doc" 프로젝트입니다.
- native에서 이식이 용이하도록 반응형 UI와 웹뷰 형태로 구현하였습니다.
- 일 5000+명 이상의 사용자가 사용하는 애플리케이션입니다.

내가 기여한 부분 (70+%)

문서 분석 플로우 개발

- Native Webview와 통신하기 위한 Bridge 인터페이스 설계
- Funnel로 문서 분석에 필요한 API들을 한번에 관리
- 개발시간 단축을 위한 MSW Mocking
- GA4를 활용한 문서 분석 플로우 및 이탈율 데이터 추적
- styled component를 활용하여 사용자 친화적인 프로그레스바 스타일링

Dialog Layer 개발

- Redux toolkit과 useModal hook을 이용한 팝업 레이어 개발 및 효율적인 팝업 관리
- Custom 에러 처리를 통한 효율적인 에러 관리

사용 기술

- **React:** Webview 내부에서 호출되어야 하고, SEO에 감춰져서 보여주어야 하는 프로젝트였기에 Pure React를 선택하였으며, 프로젝트 크기가 크지 않기 때문에 CRA로 구성하였습니다.
- **Styled Component:** 프로젝트 크기가 크지 않기에 styled component의 사용 시 문제가 될 수 있는 번들 크기에는 크게 무리가 없었고, 코로케이션을 통한 스타일 관리가 적합하다고 생각하여 선택하였습니다.
- **MSW:** 촉박한 개발기간으로 인해 조금 더 백엔드와 분리되고 시간적 측면에서 효율적으로 개발하기 위하여 MSW를 사용하였습니다.
- **Redux Toolkit:** 프로젝트 규모에 맞지 않는 복잡한 Redux 스토어 구성을 조금 더 빠르고 효율적으로 사용하고자 도입하였습니다.
- **Google Analytics4:** 프로젝트의 문서 분석 흐름 중 사용자의 이탈율과 유입율을 로깅하기 위해 사용하였습니다.

트러블 슈팅

파편화된 팝업

문제 배경

이번 프로젝트에서는 에러를 모두 팝업으로 노출하는 형태로 구성 되어졌습니다. 에러로 노출될 UI는 4 가지 정도가 되었지만 서버에서 에러 처리 로직을 변경하게되면서 모든 API에서 4가지의 에러를 전부 받을 수 있게 되었습니다. 처음에는 분석 흐름에 따라 한 컴포넌트에서 하나의 에러 팝업을 노출 시켰지만 에러코드에 따라 에러를 다르게 출력하게 되어 보다 효율적인 팝업 관리가 필요하였습니다.

해결 방법

- Redux toolkit을 활용하여 팝업을 중앙에서 배열로 관리하였습니다.
- 배열로 관리되는 팝업 컴포넌트들을 컨트롤할 수 있는 커스텀 hook을 만들어 의존성을 분리시켰습니다.
- API 에러 단계에서 조금 더 React스럽게 에러팝업을 관리할 수 있도록 useErrorHandler hook을 만들어 에러 처리의 책임을 분리 하였습니다.
- [해결 과정 링크](#)

결과

이 결과로 웹 애플리케이션에서 빼놓을 수 없는 팝업을 보다 효율적으로 관리하는 방법을 깨달았습니다.

팝업의 전역 관리를 통해 팝업을 사용하는 컴포넌트는 팝업에 대해 의존성을 가지지 않고, 오로지 컴포넌트의 일만 집중할 수 있었고, 코드 또한 많이 개선 되었습니다.

7일간의 해커톤

문제 배경

프로젝트가 내부 이슈로 인해서 홀딩이 되었다 재 개발이 들어가게 되었는데 출시일은 바뀌지 않아서 일주일 안에 개발을 해야 하는 상황이 생겼습니다.

백엔드 또한 같이 홀딩 후 작업이 들어간 상황이라 API 스펙이 정의되지 않았고 병렬적으로 개발해야하는 상황에 직면하게 되었습니다.

해결 방법

백엔드 담당자 분과 개발 방법에 대한 논의 후 Mock Service Worker 라이브러리를 도입하여 개발하였습니다. 실제 API를 호출하는 것 처럼 네트워크를 Mocking 하여, 필요한 여러 코드와 response 값을 정의 후 효율적으로 개발해 나갔습니다.

그리고 여유가 되는 시간에는 Mocking한 API에 대해 피드백하며 더 나은 API 설계에 함께 참여했습니다.

결과

이를 통해 개발기간을 2주 정도 단축할 수 있었고, 프로젝트 출시일에 맞춰서 성공적으로 출시할 수 있었습니다.

MSW의 더 깊은 사용법과 활용법을 익힐 수 있었습니다.

인포테인먼트에서도 뛴다아!!

차량용 'ASK DOC' 출시



차량용 폴라리스 드라이브

소개글	차량 소프트웨어에 주입되는 드라이브 페이지를 개발하였습니다,
진행기간	2023.09 - 2023.10
사용 기술	React React Query Styled Component TypeScript Vite Vitest

Link

Polaris Office

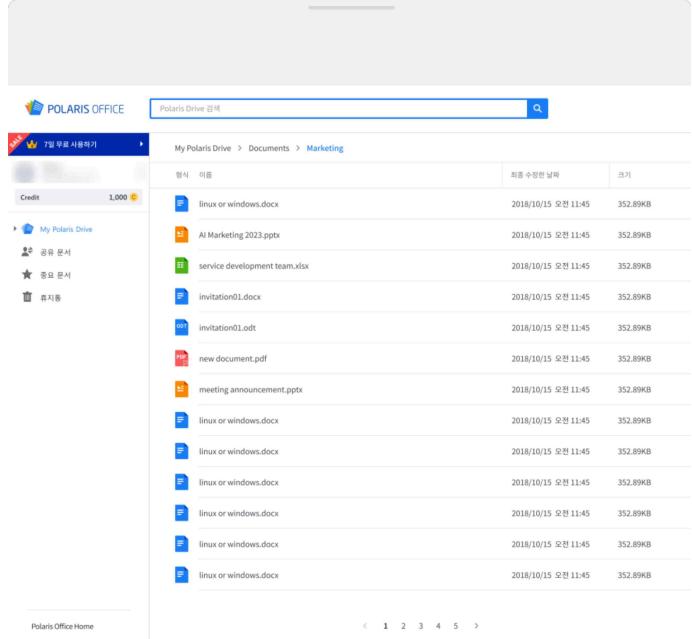
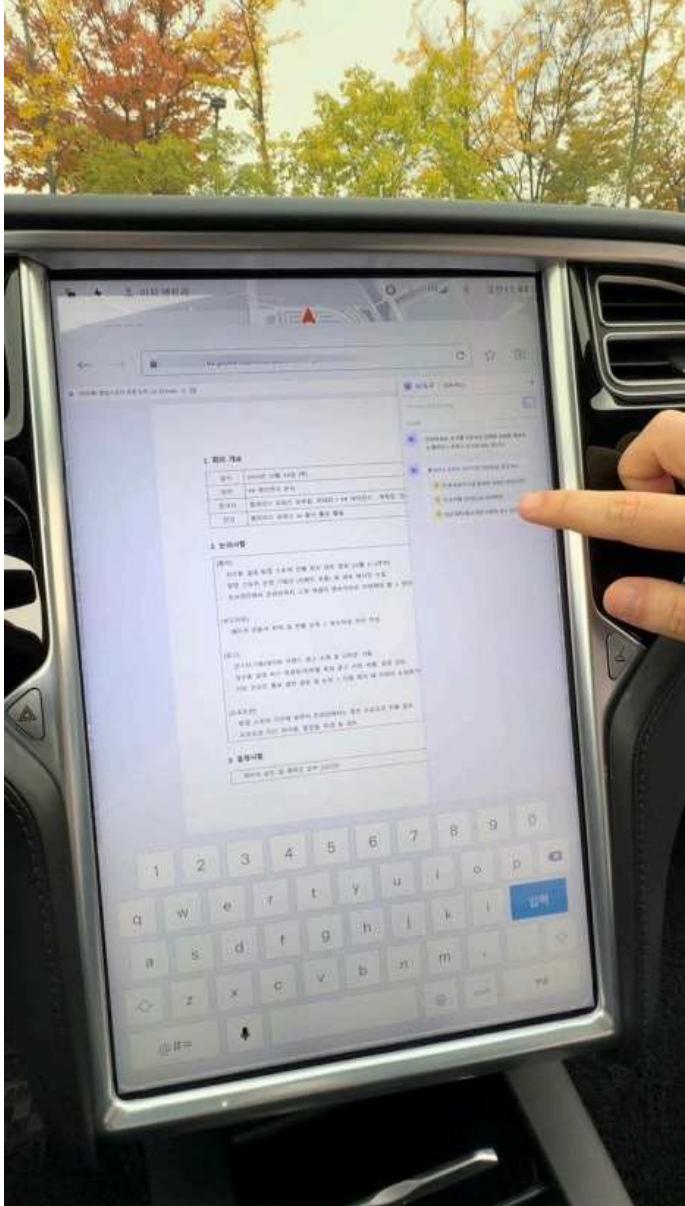
 <https://www.polarisoffice.com/ko/carMode>

프로젝트 소개

테슬라에서 만나볼 수 있는 폴라리스오피스 AI

어렸을 때, 공상과학 주제의 뉴스 기사나 만화, 영화들을 한 번쯤은 접해보셨을 텐데요. 차가 하늘을 날아...

[blog https://blog.naver.com/polarisoffice_kr/223246517390](https://blog.naver.com/polarisoffice_kr/223246517390)



파일명	최종 수정한 날짜	크기
linux or windows.docx	2018/10/15 오전 11:45	352.89KB
AI Marketing 2023.pptx	2018/10/15 오전 11:45	352.89KB
service development team.xlsx	2018/10/15 오전 11:45	352.89KB
invitation01.docx	2018/10/15 오전 11:45	352.89KB
invitation01.odt	2018/10/15 오전 11:45	352.89KB
new document.pdf	2018/10/15 오전 11:45	352.89KB
meeting announcement.pptx	2018/10/15 오전 11:45	352.89KB
linux or windows.docx	2018/10/15 오전 11:45	352.89KB
linux or windows.docx	2018/10/15 오전 11:45	352.89KB
linux or windows.docx	2018/10/15 오전 11:45	352.89KB
linux or windows.docx	2018/10/15 오전 11:45	352.89KB
linux or windows.docx	2018/10/15 오전 11:45	352.89KB
linux or windows.docx	2018/10/15 오전 11:45	352.89KB
linux or windows.docx	2018/10/15 오전 11:45	352.89KB



- AI라는 새로운 변화에 발맞춰 자동차 내 인포테인먼트(Infotainment)에서 문서를 열고, AI가 문서를 분석해 주는 새로운 기능입니다.
- 일반 컴퓨터나 디바이스보다 네트워크 성능이 좋지 않아 기존 드라이브 페이지를 경량화 하였습니다.
- 테슬라를 시작으로 타 브랜드 B2B 차량 소프트웨어 도입을 고려해 아토믹 디자인 패턴을 도입하여 개발 하였습니다.

내가 기여한 부분 (80+%)

드라이브 페이지 경량화

- 10.4 Mb → 1.4MB 번들 크기 86% 경량화
- React Query를 이용한 API 캐싱 처리

Atomic Design Pattern에 따른 컴포넌트 개발

- Atomic한 컴포넌트를 개발
 - List 컴포넌트, Table Cell 컴포넌트, Input 컴포넌트
 - 아토믹 디자인의 장점과 단점을 이해하였습니다.
- Compound Component Pattern을 도입하여 B2B시 유저가 원하는 방향대로 용이하게 설계할 수 있도록 처리

사용 기술

- **React**: Webview 내부에서 호출되어야 하고, SEO에 감춰져서 보여주어야 하는 프로젝트였기에 Pure React를 선택하였으며, 프로젝트 크기가 크지 않기 때문에 CRA로 구성하였습니다.
- **Styled Component**: 아토믹한 디자인을 구성하기에 고립된 스타일링을 제공할 수 있다고 판단하였고, 타 프로젝트와 동일한 스타일링 스택을 유지하여 유지보수가 용이하도록 하기 위해 선택하였습니다.
- **Vite**: 트렌드 번들러인 Vite를 공부해보고자 도입하게 되었습니다.
- **React Query**: 차량용 소프트웨어는 네트워크 속도가 매우 느리고, 주행 중에는 더욱 느려지는 현상이 있습니다. 그렇기에 네트워크 호출을 최적화하고자 React Query를 사용하였습니다.
- **Vitest**: B2B를 대응하는 만큼 안정성이 중요하다고 생각하여 unit 테스트를 진행하였습니다. 그러면 서도 Vite와 호환이 잘 되는 Vitest를 도입하여 사용 하였습니다.

트러블 슈팅

배포 인증 문제

문제 배경

위 프로젝트는 모노레포 내부에 프로젝트를 생성하지 않고 별도의 레포지토리로 관리하게 되었습니다. 그렇게 별도의 프로젝트로 배포 환경을 구성하였으나, 페이지 연결 시 **인증이 이어지지 않는 현상**이 발생했습니다.

원인은 쿠키 기반 인증방식을 사용 중이었지만 배포된 도메인이 서로 달라 인증정보를 가져오지 못하는 현상이 생겼습니다.

해결 방법

- 배포 파이프 라인을 변경하여 해결하였습니다. Github Action으로 CI/CD 파이프라인 구축 후, 현재 서비스 되고 있는 S3로 경로를 변경하여 해결하였습니다.

결과

인증 방식 및 쿠키가 도메인에 전달되는 방식을 습득할 수 있었습니다. Frontend쪽이나 Backend 쪽에서 각각 쿠키 공유 제한을 해제할 수 있는 방법도 공부할 수 있었고, 이로인해 생길 수 있는 보안적 문제도 습득할 수 있었습니다.

네트워크 속도 문제

문제 배경

차량에서 직접 테스트를 진행할 때, 생각했던 것 보다 더욱 네트워크 속도가 나오지 않았습니다. 정차 중일 때의 속도는 보통 3G 정도의 속도를 보였었고, 주행 중일 때는, 속력에 비례하여 느린 3G에서 그 이하 정도의 퍼포먼스를 보였습니다. 문서 플랫폼 특성 상 용량이 나가는 파일들이 많아서 해결 방법이 필요 했습니다.

해결 방법

속도의 한계를 UI적으로 많이 극복시킨 프로젝트였던 것 같습니다. 위에서 설명드렸던 것처럼 API호출은 React Query를 통해서 API 캐싱을 하여 불필요한 API 호출을 줄였습니다. 그리고 기존 페이지에 있던 불필요한 광고나, 정적 파일 등을 제거하여 리소스를 줄였습니다.

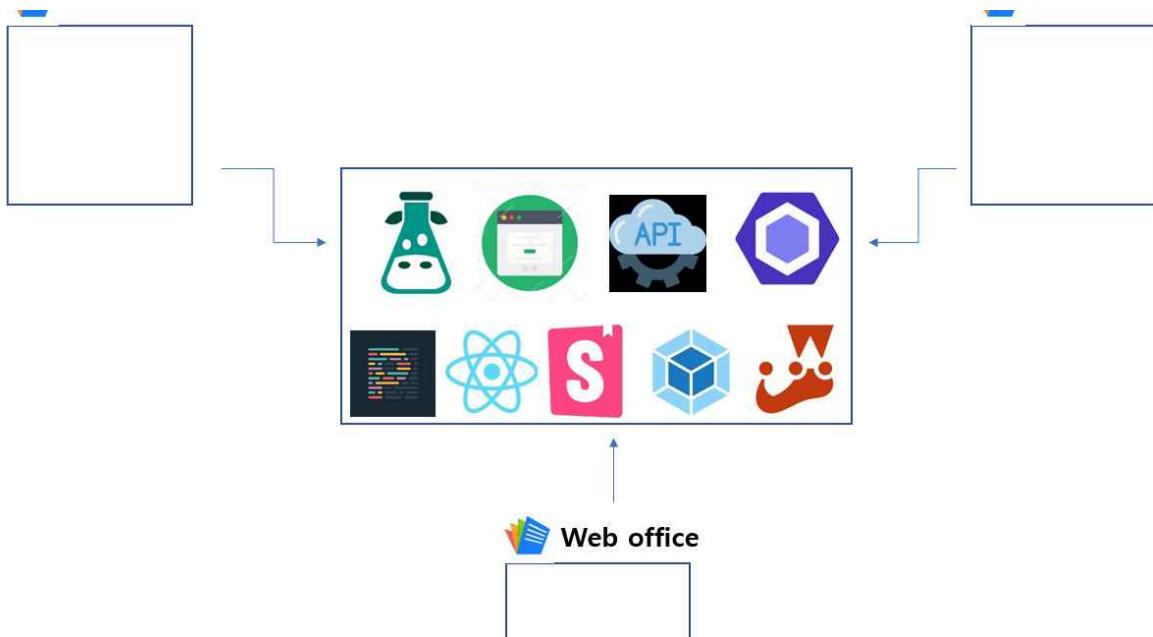
그럼에도 느려져 보이는 부분은 스켈레톤 UI를 도입하여 사용자가 기다리는 중에도 심리적으로 편할 수 있도록 하였습니다.

결과

- 스켈레톤 UI, 로딩 스피너 등의 로딩 화면에 대한 차이점을 알 수 있는 시간이 되었습니다.
- React Query의 캐싱 시간을 조절하여 좀 더 고도화 된 프로젝트를 만들 수 있었습니다.

후기

- 비즈니스적 문제를 UI적으로도 해결할 수 있다는 부분을 배울 수 있었습니다.
- Github Action의 파이프라인을 처음부터 구축해보면서 CI, CD의 개념을 다시 한 번 익힐 수 있었습니다.
- 아토믹 디자인패턴을 적용하면서 컴포넌트의 설계의 중요성을 깊게 깨달을 수 있었습니다.



플라리스오피스 플랫폼 레거시 개선

소개글	JSP와 jQuery를 기반으로 한 플랫폼을 Next.js로 마이그레이션 한 프로젝트입니다.
진행기간	2023. 03 - 2023.10
사용 기술	Jest Next.js React React Query Storybook Typescript Yarnberry i18n

프로젝트 소개

- JSP와 jQuery를 기반으로한 거대한 모놀리식 프로젝트를 Next.js와 React 기반의 프레임워크로 마이그레이션 한 프로젝트입니다.
- 20년이 넘는 시간 동안 코드가 쌓이고 여러 개발자들이 지나간 흔적들이 있는 레거시를 유지보수 가 쉽도록 점진적 마이그레이션을 진행하였습니다.
- 프로젝트 도메인을 분리하고, 디자인 시스템을 도입하며 Yarnberry Workspace 기반 모노레포로 새로이 구축하였습니다.

문제상황 인지

- JSP로 인해 Backend와 Frontend 양 쪽에 **비즈니스 로직 혼재**
- 대규모의 UI를 jQuery를 통한 control로 인한 **성능 저하 및 유지보수의 불편함** 발생
- 서로 다른 도메인이어도 같은 레포지토리 안에 있어 **비즈니스 로직의 결합도가 높음**
- JSP를 사용하고 있음에도 **웹 접근성이 최적화 되지 않고 있음**
- 서로 다른 script에서 변수를 사용할 때에 변수 타입 에러의 잦은 발생
- 더 이상 **미 사용되는 리소스들의 정리 필요성**

내가 기여한 부분 (30+%)

모노레포 환경 구축

- Yarnberry Workspace를 활용하여 모노레포 환경을 구축하였습니다.
- 컴포넌트 라이브러리에 Storybook을 사용하여 디자인팀과 새롭게 디자인 시스템을 구축했습니다.
- 잦은 타입 에러를 해결하기 위해 Typescript와 ESLint Config를 회의를 통해 규칙을 정하고 이를 바탕으로 프로젝트에 반영하였습니다.
- Jenkins를 통한 각 패키지 별로 빌드 파이프라인을 구축 하였습니다.
- [모노레포 환경 구축 관련 글](#)

공통 라이브러리 개발

- 유지보수가 용이한 디자인 시스템 구축을 위해 Dialog, Button, Input 등의 **공통 컴포넌트를 개발**하였습니다.
- 라이브러리용 컴포넌트 패키지는 **ESBuild를 통해 빌드** 시켜 빠른 빌드 최적화를 도왔습니다.
- **Storybook을 사용**하여 디자이너와 협업을 촉진 시켰습니다.
 - Figma Plugin을 통한 디자이너도 개발자의 결과물 확인 가능하도록 처리

플라리스오피스 소개페이지 개발



- 정적 랜딩 페이지를 **Next.js**를 이용해 개발하였습니다.

- Lighthouse**를 통한 측정값 평균 4배의 성능개선

- FCP: 3.4초 → 1초 **약 3.4배** 상승
- Speed Index : 5.8 → 2.0초 **약 2.5배** 상승
- LCP : 5.9 → 2.3초 **약 2배** 상승

- 보다 나은 웹 접근성을 고려하여 개발하였습니다.

- 코드 리뷰 시 시멘틱 태그도 잘 사용하도록 확인하는 문화 정착시켰습니다.
- ARIA 및 Next.js의 Meta태그 적극 활용하여 **SEO도 최적화** 하도록 하였습니다.

🛠️ 사용 기술

- React**: 리액트를 사용하게 된 이유는 위 글에 자세히 설명되어 있습니다.
- Next.js**: SEO의 최적화, 이미지 최적화, Metadata 간편생성과 서버 컴포넌트를 사용하기 위해 App Router를 채택하였습니다.
- Jest**: 가장 보편적인 테스팅 라이브러리이고, 모노레포 환경에서 편하게 구축할 수 있도록 Doc가 잘 되어져 있어서 Jest를 통한 테스팅 환경을 구축하였습니다.
- Yarnberry Workspace**: lerna, npm workspace 등의 모노레포 툴이 많았지만, Zero install과 PnP 기능을 활용하고, 원티드 프리온보딩 과정에서 yarnberry workspace를 학습한 경험이 있어서 채택하였습니다.
- Storybook**: 디자이너와 새로운 디자인시스템을 구축하기 위하여 채택하였습니다.
- ESBuild**: 공통 컴포넌트 패키지는 빌드된 패키지로 사용하기로 약속을 정하였습니다. 그러나 공통 패키지가 커질수록 빌드 시간도 점점 늘어나게 되어, 이 부분을 개선하고자 ESBuild를 도입하였습니다.
- i18n**: 16개 국어를 지원하기 위해서 i18n을 도입하였고, 언어만 담당하는 패키지를 별도로 만들어서 라이브러리 형태로 공통적으로 언어를 꺼내쓸 수 있도록 하였습니다.

트러블 슈팅

테스트 코드 도입 및 테스팅 환경 구축

문제상황

사내에서 프로젝트를 진행하며 테스트코드에 대한 필요성을 많이 느꼈습니다. 레거시 코드였기 때문에 테스트 코드 환경은 없었고 사내에 QA팀을 통해 릴리즈 전에 QA에서 오류를 잡고 처리하는 프로세스였습니다. QA과정에서 나온 오류들에서 40% 가량이 코드 수준에서 나올만한 가벼운 오류들이 많았습니다.

해결 방법

- 컴포넌트 단위의 Unit 테스트를 도입하였습니다.
- 테스트가 생소한 팀원들을 위해 프론트엔드 테스트코드 스터디를 주도하여 진행하였습니다.
- 코드리뷰 시 테스트코드 유무 확인을 통해 어느 정도의 테스트 코드의 강제성을 부여하였습니다.

결과

- マイグ레이션 후 QA에서 노출되는 Low레벨의 수가 많이 줄게 되었습니다.
- 테스트 작성에 용이한 컴포넌트를 작성하기 위해 SRP에 부합하는 컴포넌트를 설계하고 짜는 능력이 향상되었습니다.

후기

- 과거 모던 웹 개발의 장단점과 현대 웹 개발의 장단점을 크게 느낄 수 있는 시간이었습니다.
- Next.js의 App Router에 대해 실무적으로 사용할 수 있는 능력이 향상 되었고 React Server Component의 동작원리를 깊게 이해할 수 있었습니다.
- 디자인시스템을 구축하여 디자이너와 긴밀하게 협업할 수 있는 시간을 가졌습니다. 그 덕분에 조금 더 효율적으로 디자인 소통할 수 있는 방법을 깨달았습니다.
- 모놀리식 레포를 모노레포로 전환을 하면서 모노레포의 장단점을 학습할 수 있었고, 모노레포 환경을 직접 구축할 수 있습니다.

폴라리스 오피스 웹

한글, 시트, 슬라이드, 워드를 프로그램 설치 없이 웹 브라우저에서 무료로 사용하세요.



웹 오피스 편집모드

☞ 소개글	뷰어만 제공되던 폴라리스오피스 Web의 편집모드를 제공하였습니다.
▣ 진행기간	2023. 02 - 2023.04
☰ 사용 기술	IndexedDB Javascript jQuery

Link

Polaris Office

Shared with Polaris Office

<https://polarisoffice.com/editor/po/hwp>

웹 한글

Polaris Office

Shared with Polaris Office

<https://polarisoffice.com/editor/po/sheet>

웹 시트

Polaris Office

Shared with Polaris Office

<https://www.polarisoffice.com/editor/po/word>

웹 워드

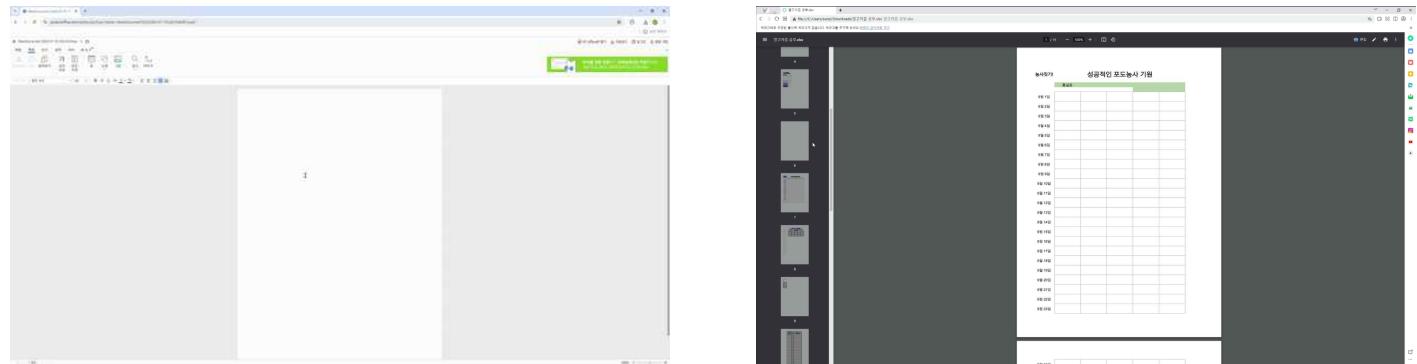
Polaris Office

Shared with Polaris Office

<https://www.polarisoffice.com/editor/po/slide>

웹 슬라이드

프로젝트 소개



- 기존에 웹 뷰어로만 볼 수 있던 웹 오피스를 jQuery를 이용하여 편집모드를 구현 하였습니다.
- 유저 레벨에 따른 메뉴 커스텀과 로컬 문서를 웹에서 열 수 있도록 구현 하였습니다.
- 엔진 SDK를 전달받아 웹에서 사용할 수 있도록 브릿지 인터페이스를 설계하였습니다.
- 네이버 웨일팀과 협업하여 웨일 브라우저에서 문서 오픈 시 웹 오피스로 열릴 수 있도록 Endpoint 를 설정하였습니다.
- 일 평균 35000+ 이상의 사용자가 사용하고 있습니다.

내가 기여한 부분 (50+%)

SDK 인터페이스 모듈 개발

- 엔진 SDK를 제어할 수 있는 인터페이스를 개발했습니다.
- 효과적인 모듈 제어를 위해 Class로 선언하여 User 레벨에 따른 동작 제어를 용이하게 처리
 - 비즈니스 로직을 Class로 집중시켜 요구사항이 변경되어도 유지보수가 용이하게 되었습니다.

로컬문서 열기 메뉴 탭 개발

- 재사용되는 UI를 컴포넌트 형태로 개발하였습니다.
 - **jQuery를 React스럽게 사용**하여 UI를 재사용할 수 있도록 처리했습니다.
 - 공유문서, 드라이브 문서, 중요 문서 등 동일한 UI지만 호출부만 다른 형태를 컴포넌트 형태로 추상화하고, Props를 통해 타입을 선택하도록 구현하여 재사용성을 높였습니다.

유저 레벨에 따른 Tools 커스텀 로직 개발

- 유저의 Level에 따라 다르게 사용할 수 있는 Tools 로직을 개발 하였습니다.
- **전략 디자인 패턴**을 활용하여 각기 다른 행동들을 추상화하여 효과적으로 동작들을 주입하도록 처리하였습니다.
 - 새로운 기능이 추가되고 그 기능들이 레벨별로 나뉘어져도 동작 클래스에만 추가하면 되기 때문에 유지보수가 용이해졌습니다.

작업 문서 보존 로직 개발

- IndexedDB를 이용하여 작업 중 문서를 문서 용량이 크더라도 손실되지 않도록 처리하였습니다.

🛠️ 사용 기술

- **jQuery**: 프로젝트 자체가 jQuery로 구현되어져 있기 때문에 프로젝트를 따라 jQuery를 사용 하였습니다.
- **IndexedDB**: 문서를 로컬 브라우저에 저장이 필요했는데, 문서 파일이 커지는 것을 고려하여 구조화된 데이터를 저장할 수 있는 IndexedDB를 사용하였습니다.

💡 트러블 슈팅

작업 중 문서가 소실되는 문제

문제 배경

로그인 후에 작업 중 문서로 돌아오게 되더라도 작업 중이던 문서가 그대로 있어야 했지만, 문서 파일이 커질 경우 문서의 내용이 소실되는 현상이 발생했습니다. 그 이유는 기존에 LocalStorage에 저장했지만 문서 용량이 저장 용량(약 5MB)을 넘어갈 경우 소실되는 현상이었습니다.

해결 방법

- 저장 로직을 Local Storage에서 IndexedDB로 변경하여 해결
 - 구조화된 데이터를 직접 저장할 수 있고, 저장 가능한 용량이 크다.
 - 웹 오피스 SDK가 ES6 이상을 지원하는 최신 브라우저만 지원하기에 IndexedDB 사용에도 문제가 없었다.
- 해결 과정 링크

결과

아무리 큰 용량이나 이미지가 많은 문서여도 전혀 문제 없이 저장이 가능했습니다.

Blob 파일을 컨트롤 할 때 기존에는 로컬스토리지에 저장하기 위해 Base64로 인코딩과 디코딩을 시도했어야 했지만 Blob 파일을 직접 저장할 수 있기에 코드 로직이 감소 하였습니다.

80개가 넘는 Tools 기능 UserLevel 별 관리 문제

문제 배경

메이저하게 사용되는 유저 Level은 6개가 있었고, 웹 오피스에 사용되는 Tools는 80개 정도가 되었습니다. Tools 메뉴는 배열로 관리되고 있었는데 처음에는 User의 Level에 따라 직접 저장하는 하드 코딩 형태로 되어져 있어서 유지보수가 복잡할 것이라고 예상되어 로직 변경이 필요했습니다.

해결 방법

```
//pseudo-code
interface UserWebOfficeLevelStrategy {
    displayUserLevel(): 'A' | 'B' | 'C' ...;
    getToolsByLevel(): 'draw' | 'insertImage' | 'useAI';
}

//유저 A Class
class UserAStrategy implements UserWebOfficeLevelStrategy {
    getUserLevel() {
        return 'A'
    }
    getTools() {
        return ['insertImage']
    }
}

//유저 B Class
```

```

class UserBStrategy implements UserWebOfficeLevelStrategy {
    getUserLevel() {
        return 'c'
    }
    getTools() {
        return ['insertImage', 'draw', 'useAI']
    }
}

// 유저 사용 함수
function displayAndPerformAction(strategy: UserWebOfficeLevelStrategy) {
    strategy.getUserLevel();
    strategy.getTools();
}

displayAndPerformAction(new UserAStrategy());

```

전략 디자인 패턴을 도입하여 웹 오피스가 열릴 때 동적으로 변경할 수 있도록 처리하였습니다.

위 의사코드처럼 유저 레벨과 각 Tools들을 정의하여 로그인한 유저의 레벨에 따라 동적으로 주입시킬 수 있도록 처리하였습니다.

결과

Tools의 기능이 바뀌거나 새로운 유저 레벨이 도입되더라도 패턴에서 정의한 결과 값만 수정해주면 되기 때문에 유지보수가 용이해졌습니다.

네이버 웨일이나 타 프로그램에 웹 오피스가 주입 되어도 독립적인 Tools들을 정의할 수 있었습니다.

후기

- IndexedDB에 대해서 깊게 알 수 있었습니다.
 - IndexedDB를 직접 열고 환경을 구성 할 수 있습니다.
- Blob 자료구조에 대해 알 수 있었고, Blob을 Base64로 변환할 때 생기는 문제를 파악할 수 있었습니다.
- 전략 디자인 패턴을 공부하고 실무적으로 활용할 수 있었습니다.
- jQuery를 사용하더라도 React처럼 웹 컴포넌트 형태의 패턴을 사용하여 개발할 수 있었습니다.
 - 그로 인해 생기는 이점과 컴포넌트 개발 방법이 어떠한 문제를 해결하기 위해 등장했는지 깨달을 수 있었습니다.



[해, 커리어] 해커톤

소개글	원티드에서 주관한 채용 연계형 해커톤입니다.
진행기간	2021. 11
사용 기술	

소개

- COVID-19 시대 대학생을 위한 프로그램 제작 해커톤입니다.
- 개발자가 아닌 기획자로 참여하여 프로젝트의 전체적인 기획을 맡았습니다.
- 작성한 기획서

결과

wanted

안녕하세요. [해, 커리어] 브론즈 배지 수상 팀으로 선정되었습니다.

[해, 커리어]에 많은 관심 보내주시고, 참여해 주셔서 진심으로 감사드립니다! 부여된 브론즈 배지는 12/6(월)부터 원티드 내 이력서 프로필에서 노출 예정입니다. 자세한 내용은 아래 본문 확인 부탁드립니다.

김정수님의 행복한 커리어 여정을 응원하겠습니다.

- 동상을 수상하여 브론즈 배지를 수상하였습니다.

회고

개발자가 아닌 기획자로 참여하여 어려운 부분이 정말 많았습니다. 기획이란 사용자의 불편한 점을 찾고 더 나은 방향과 쉬운 UI,UX를 통해서 비즈니스적 가치를 실현해야 하는 직무지만 개발자의 입장에서 생각하는 것이 너무도 많았습니다.

그래도 시장분석, 문제정의, 포지셔닝 맵 작성, 타겟 분석등을 직접 해보면서 기획에 대한 시각을 넓힐 수 있었고, 개발자로서 기획의 의도가 무엇인지 더 넓은 범위로 파악할 수 있게 되었습니다.

Walking Together



Walking Together

☞ 소개글	삼육대학교 학생들의 봉사활동을 위한 '함께 걷기' 서비스
▣ 진행기간	2021. 01 - 2021. 06
☰ 사용 기술	Javascript React Redux SCSS

🔗 Link

[Walking_Together/client at master · sunpl13/Walking_Together](#)

Contribute to sunpl13/Walking_Together development by creating an account on GitHub.

**sunpl13/
Walking_Together**



https://github.com/sunpl13/Walking_Together/tree/master/client

3 Contributors 0 Issues 2 Stars 1 Fork



프로젝트 소개

The figure displays four screenshots of the 'Walking Together' mobile application interface, arranged in a 2x2 grid.

- Top Left (Home Screen):** Shows the title 'Walking Together' with a walking couple icon. Below it is a list of recent activities:
 - # 공지사항 (Notice) - 2021-06-02
 - 공지사항 (Notice)
 - 사회봉사단 (Community Service Team)
 - 공지사항 (Notice) - 2021-05-31
 - 사회봉사단 (Community Service Team)A blue button labeled '더보기' (View more) is at the bottom right.
- Top Right (Engagement Screen):** Shows a group of people cheering with a yellow circle icon labeled '1'. Below are six user profiles:
 - 2 김현 (Kim Hyun) - 2015100885, 7914m
 - 3 김수 (Kim Soo) - 2015100899, 44403m
 - 4 김정 (Kim Jung) - 2018100956, 3000m
 - 5 김우 (Kim Woo) - 007236, 0m
 - 6 이희 (Lee Hee) - 1234567891, 0mA blue button labeled '종료' (End) is at the bottom right.
- Bottom Left (Fitness Screen):** Shows a grey circle icon labeled '피드' (Feed). Below is a list of recent feed items:
 - 2021-05-31 종료 40014m 파트너 김동봄
 - 2021-06-01 종료 4389m 파트너 김동봄A blue button labeled '종료' (End) is at the bottom right.
- Bottom Right (Map Screen):** Shows a map of a city area with a green route line and a blue circle icon labeled '활동' (Activity). A blue button labeled '종료' (End) is at the bottom right.

개발 동기

- COVID-19로 인해 코로나 이전(2019년) 대비 2020년 자원봉사율이 52% 감소됨
- 둔화된 봉사활동에 어떻게 활기를 줄 수 있는지 고민에서 시작

세부 기능

1. 이동 거리, 이동 경로 등의 정보를 수집하고, 수집된 정보에 따른 경로를 출력

- Geolocation을 사용하여 위치정보 수집
- Kakao Map API를 사용하여 경로 출력

2. 별도의 관리자 모드에서는 학생정보, 봉사활동 정보 및 파트너 정보를 간편하게 관리 가능

- 관리자 모드는 관리자 특성상 데스크탑 PC형태의 적응형으로 제작
- 공지사항 입력 가능
- 학생 정보 및 파트너 정보 Excel로 별도로 출력 가능
- 학생의 봉사활동 인증서 출력 가능

[file.pdf](#)

사용자 인증서

3. Walking Together를 통해 삼육대학교 학생들은 직접 봉사처를 알아보는 시간적 투자와 활동 시간을 할애하는 투자를 상대적으로 줄일 수 있어서 심적 부담감 완화

4. 반응형 웹앱으로서 모바일에 특화 된 웹앱(Web App)

- PWA화 완료하여 모바일에서 더 부드럽게 동작
- PlayStore에 출시

사용 패키지 및 라이브러리

- Redux/React-redux
 - 데이터 전역 관리를 위한 리덕스 패키지
- SASS
 - 컴포넌트 스타일을 설정하는 패키지
- Axios
 - 서버 통신용 패키지
- dotenv

- 환경변수 설정을 위한 패키지
 - lodash
 - 디바운싱 구현을 위한 패키지
 - React-Quill
 - 게시판 입력용 커스텀 툴
 - Pdfmake
 - 사용자 인증서 발급을 위한 패키지
-

관련활동

- 삼육대학교 신문 Walking Together 관련기사 발췌
 - 삼육대 신문 제 419호

[삼육대신문 419호\(발췌본\).pdf](#)

- 베타 테스터 모집
 - [재학생 대상으로 베타테스터 모집](#)
 - [모집 결과](#)
-

회고 및 느낀점

| [김정수 Walking Together 회고록](#)