권동우 | Frontend Developer

Email : hyeonggyugwon3@gmail.com
Phone : 010-8786-9260
GitHub : @kwondongwoo0424

실패가 즐거운 개발자

안녕하세요, 실패를 통해 더욱 성장할 수 있다고 믿는 **신입 개발자 권동우**입니다.
프론트엔드 기술에 흥미를 갖고 활발히 활동하고 있으며,
애자일과 MVP 방식으로 개발하는 것을 즐깁니다.
서비스 **기획 / 디자인 / 개발 / 배포 / 운영** 경험이 있습니다.

보다 나은 개발자가 되기 위해 **끊임없이 도전**하며,
실패 속에서 배움을 얻고 성장하고 있습니다.
프로젝트를 진행할때 **프로덕트 오너십**을 가지고 지속적으로 학습하고 UX를 개선합니다.
"나를 죽이지 못하는 것은 나를 더 강하게 만든다" 는 니체의 말처럼,
실패를 **성장의 발판**으로 삼아 더 나은 개발자로 나아가고 있습니다.

기술

React, TypeScript, JavaScript, Next.js, HTML
styled-components, Tailwind CSS, SCSS, CSS
TanStack Query, Zustand, Storybook
Github, Figma, Vercel, Github Pages
Java, Spring Boot, Express.js

학력

대구소프트웨어마이스터고등학교

2024.03. ~ 2027.02 (졸업예정)

이수 전공 과목

파이썬 프로그래밍 / 웹 프로그래밍 기초 / 데이터베이스 프로그래밍 / 자료구조 / 인공지능 기초 / 소프트웨어 공학 / 자바 프로그래밍 / 웹 프로그래밍 / 네트워크 / 웹 프로그래밍 실무 / 자바 프로그래밍 실무

프로젝트 목차

대표 프로젝트

Sync Desktop

팀 프로젝트
총원 7명

비개발자와 개발자의 협업을 위한 AI 기반 프로젝트 관리 SaaS

React | TypeScript | Electron | styled-components | TanStack Query | Zustand

데스크탑 앱 개발 / UI 디자인 / 디자인 시스템 구축 / 렌딩 페이지 개발 / SSE 기반 실시간 단어 해석 기능 개발

사이드 프로젝트

DIA

팀 프로젝트
총원 4명

대구소프트웨어마이스터고 성적 가산출 서비스

React | TypeScript | styled-components | React Context API

성적 계산 페이지 개발 / 성적 산출 알고리즘 모듈화

ColorVerse

개인 프로젝트
총원 1명

색 조합을 추천해주는 웹 서비스

React | styled-components | Express.js | OpenAI API

Express.js 서버 구축 / OpenAI API 통합 및 프롬프트 엔지니어링

Flick

팀 프로젝트
총원 5명

학교 축제 부스 운영을 위한 전자 화폐형 실시간 거래•정산 플랫폼

Next.js | Tailwind CSS | TypeScript | pnpm

어드민 페이지 실시간 부스 랭킹 시스템 개발

외부 프로젝트 기여 경험

Legacy web

"게임으로 배우는 우리나라 문화와 역사" 웹·앱 기반 학습 서비스

React | TypeScript | styled-components

OAuth 버튼 컴포넌트 리팩터링 (PR #75) / 보안 취약점 발견 및 이슈 제기 (Issue #76)

Starthub web

AI 기반 맞춤형 창업 지원 정보 제공 서비스

React | TypeScript | styled-components

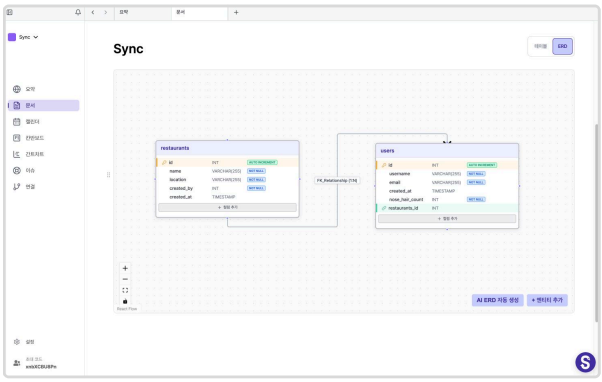
낙관적 업데이트로 반응성 개선 (PR #199)

대표 프로젝트

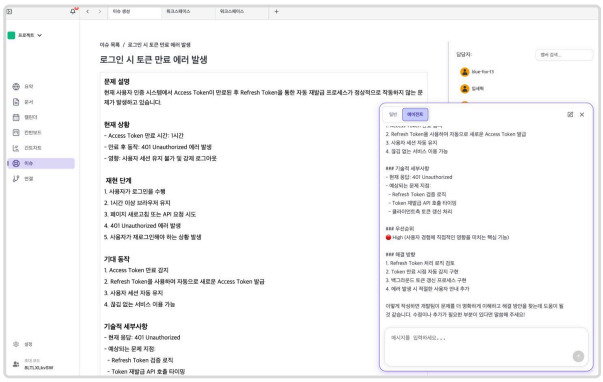
Sync Desktop

2025.03. ~ 2025.10

비개발자와 개발자의 협업을 위한 AI 기반 프로젝트 관리 SaaS



문서 페이지 - 기능명세서 기반 자동 ERD 생성



DeepSync - 이슈 내용 자동 수정

개요

디자이너, 기획자 같은 비개발자와 개발자 간의 기술 격차로 인해 발생하는 의사소통 문제와 협업 비효율을 해결하기 위한 AI 기반 프로젝트 관리 데스크톱 앱입니다.
지식 격차를 AI가 보완하고, 팀원 간 실시간 협업 환경을 제공하는 것이 핵심 목표입니다.

핵심 문제

- 비개발자는 기술 지식 부족으로 요구사항 표현이 어려움
- 개발자는 비즈니스 맥락을 놓쳐 커뮤니케이션 비용 발생
- 협업 과정에서 잦은 오해, 시간 낭비, 반복 작업 발생

해결 아이디어

AI로 비개발자의 기술 이해도를 높이고,
실시간 협업 기능으로 개발자와의 커뮤니케이션을 최소 비용으로 만든다.

주요 기능

- 비개발자를 위한 드래그 기반 개발 단어 해석
- 프로젝트 컨텍스트를 이해하는 AI 챗봇으로 맞춤형 도움말 제공
- 실시간 협업이 가능한 마크다운 문서 페이지 및 기능명세서 기반 AI 자동 ERD 생성

팀원

- 총원 7명
프론트엔드 4명(본인포함), 백엔드 3명, 디자인 2명(본인포함)
본인 역할: Frontend Developer & UI/UX Designer

주요 사용 기술

React

상태 변화가 많은 화면을 빠르게 구성하기 위해 선택

TypeScript

협업과 유지보수에 필요한 타입 안정성을 확보하기 위해 사용

Electron

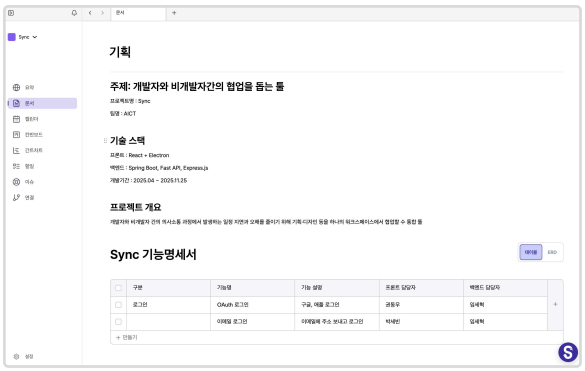
- 기존에 익숙한 웹 기술을 활용해 데스크탑 앱을 개발하기 위함
- macOS · Windows 등 다양한 환경에서 동일한 UX를 제공하기 위한 멀티 플랫폼 지원
- 브라우저 환경과 유사한 개발 흐름을 유지하며 개발 속도와 생산성을 확보할 수 있음

styled-components

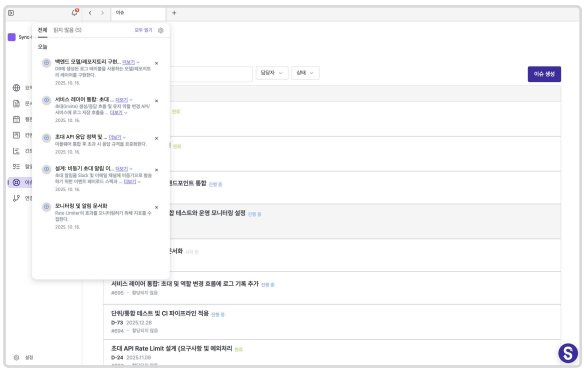
컴포넌트 단위로 스타일을 일관성 있게 관리하기 위해 사용

주요 활동

UI / UX 디자인



문서 페이지



이슈 페이지

Figma의 **Auto Layout** 기능을 활용하여 사용자 경험을 고려한 화면을 설계

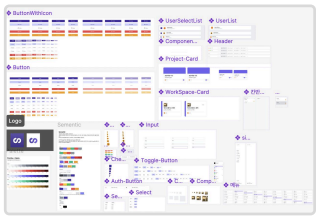
sds 개발

디자인 시스템 구축 (sync-design-system)

UI와 컴포넌트가 일관되지 않아 유지보수가 어렵고
재사용이 힘들었습니다.

- 재사용 가능한 15개 공통 컴포넌트 추출
- **npm 패키지**(@sync-design-system)로 **배포**하여 팀 내 공유
- **Storybook** 기반 **컴포넌트 문서화** 및 **사용 가이드 제공**

→ 체계적인 컴포넌트 설계로 **개발 효율 향상** 및 **UI 일관성 확보**



디자인 시스템 구축 (Figma 활용)

npmjs에 배포된 sds

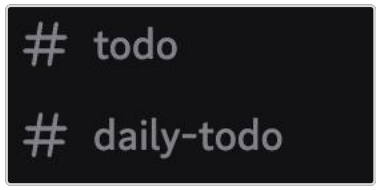
애자일

애자일 개발론 적용

7개월 팀 프로젝트에서 초기 2개월간 명확한 마일스톤 없이
진행하다 주요 기능 지연 및 팀원 간 진척 파악 어려움 발생했습니다.

- 1주 **스프린트 체계 도입**: 매주 목요일 데모 및 회고,
금요일 스프린트 계획
- Notion로 백로그 관리, 스프린트 보드, 회고록 문서화
- 디스코드로 데일리 리마인더, 블로킹 이슈 즉시 공유

→ 장기 프로젝트에서 일정 예측 가능성과 팀 생산성을
높이는 경험 습득



daily-todo 작성 (디스코드 활용)

1주일 마다 스프린트 회고 작성

주요 활동

UX 개선

React Query 캐싱으로 이슈 페이지 로딩 시간 단축

사용자가 이슈 목록 ↔ 상세 페이지를 이동할 때마다 API 재호출
→ 평균 2초 로딩 소요, 같은 데이터를 반복 요청해 서버 부하 증가

- **React Query** 도입으로 이슈 **데이터** 자동 **캐싱**
- staleTime 2분 설정: 신선한 데이터는 재요청 없이 즉시 표시 (이슈 목록은 자주 변경되는 데이터이므로 staleTime를 짧게 설정함)

→ 캐싱 전략으로 **성능**과 **사용자 경험**을 동시에 **개선**

낙관적 업데이트로 할 일 체크 반응성 개선

할 일 완료 체크 시 서버 응답 대기로 인한 0.5~1초 지연 발생
→ 사용자 경험 저하 발생

React Query **낙관적 업데이트 적용**으로 즉시 UI 반영

- **체감 응답속도** 평균 **800ms → 0ms**로 **단축**
- 네트워크 실패 시 자동 롤백 + 에러 토스트로 사용자 알림

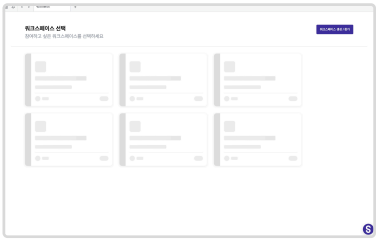
낙관적 업데이트 적용한 할 일 목록

불필요한 스켈레톤 UI 제거

워크스페이스 카드에 스켈레톤 UI 적용했으나
로딩 시 찰나에 나타났다 사라지는 플래시 현상 발생

- Chrome DevTools로 API 응답 시간 측정 결과 평균 180ms
- 스켈레톤 UI가 표시되자마자 사라져 시각적 노이즈로 작용

→ 스켈레톤 UI 제거, 데이터 도착 즉시 콘텐츠를 렌더링하여
깜빡임 없이 **깔끔한 로딩 경험 제공**

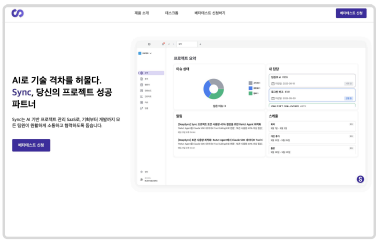

스켈레톤 UI가 적용된
워크스페이스 페이지

Sync introduce

베타테스트 랜딩 페이지(sync introduce) 개발

- GitHub Pages + 커스텀 도메인으로 베타테스트 신청 페이지 구축
- GitHub Releases 기반 배포로 Windows/macOS 설치 파일 제공

→ 제품 검증을 위한 **효율적인 베타 배포 전략 수립** 및 실행 경험



Sync introduce

주요 활동

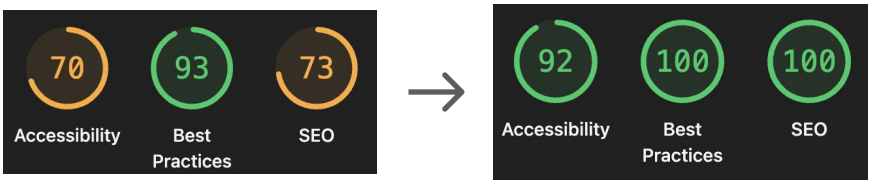
SEO 최적화

SEO 최적화를 통해 자연 검색 유입 개선

Sync Desktop의 검색 노출 개선 및 자연 유입 증대를 위해 랜딩 페이지(Sync-introduce)의 SEO 최적화를 진행

- sitemap.xml, robots.txt 설정 및 메타태그 최적화로 **검색엔진 크롤링 효율화**
- 커스텀 도메인 적용 및 **Lighthouse SEO** 점수 **100점** 달성
- 기술적 SEO 구현을 통한 유기적 사용자 유입 경로 확보

→ 검색엔진 크롤링 메커니즘 이해 및 기술적 구현 방법 습득



Light-House 측정 결과

트러블 슈팅

SSE 스트리밍 응답의 과도한 리렌더링 최적화

문제 상황

SSE를 활용해 사용자가 텍스트를 드래그하면 실시간으로 AI가 단어를 설명해주는 기능을 구현하던 중, 응답이 화면에 표시되는 동안 화면이 끊기고 버벅거리는 문제가 발생했습니다.

원인

매 청크 수신마다 setState를 호출하여 과도한 리렌더링이 발생함

- n개 청크 수신 시 → n회 리렌더링 발생
- 각 렌더링마다 Virtual DOM 생성 및 Diffing 과정이 수행됨

→ 성능 저하

```
onChunk: (chunk) => {
  // 현재 처리 중인 단어가 맞는지 확인
  if (currentWordRef.current !== word) {
    return; // 다른 단어의 응답이면 무시
  }

  // 매 청크마다 즉시 상태 업데이트
  setDefinition(prev => prev + chunk);
},
```

기존 코드

해결 과정

이 문제를 해결하기 위해 Throttling과 Debouncing의 두 가지 방식을 고민하였습니다.

1. Throttling
   - 일정 간격마다 렌더링을 시도하기 때문에 타이핑 효과가 일정하고 부드러움
   - 그러나 새로운 청크가 도착하지 않는 구간에서도 주기적으로 렌더링이 발생하여, 불필요한 렌더링이 발생됨

2. Debouncing
   - 청크가 연속적으로 수신되는 동안에는 렌더링을 지연
   - 일정 시간동안 새로운 청크가 수신되지 않을 때 한 번만 렌더링 실행

Throttling은 타이핑 애니메이션이 자연스럽다는 장점이 있었지만, 불필요한 렌더링이 발생해 성능적인 문제가 남아 있었음.
반면 Debouncing은 타이핑 효과는 약간 덜 자연스러울 수 있으나, 렌더링 횟수를 크게 줄여 UI 끊김 문제를 가장 효과적으로 해소할 수 있었음.
이러한 이유로 최종적으로 Debouncing 방식을 선택했습니다.

해결 방법

Debouncing + useRef 기반 버퍼링 패턴 적용

1. useRef를 중간 버퍼로 활용 → 리렌더링 없이 데이터 누적
2. 50ms Debouncing 적용 → 연속 청크 도착 시 렌더링 지연
3. 이전 타이머 취소 → 마지막 청크 기준으로 한 번만 setState 호출

→ 청크는 실시간으로 버퍼에 쌓이지만, 50ms 동안 추가 청크가 없을 때만 화면 갱신되도록 최적화

```
onChunk: (chunk) => {
  // 현재 처리 중인 단어가 맞는지 확인
  if (currentWordRef.current !== word) {
    return; // 다른 단어의 응답이면 무시
  }

  definitionRef.current += chunk;

  if (updateTimerRef.current) {
    clearTimeout(updateTimerRef.current);
  }

  updateTimerRef.current = setTimeout(() => {
    // 다시 한번 현재 단어 확인
    if (currentWordRef.current === word) {
      setDefinition(definitionRef.current);
    }
  }, 50);
},
```

Debouncing 방식을 적용한 코드

결과 및 배운점

- setState 호출 횟수 대폭 감소로 UI 끊김 문제 해결
- Throttling, Debouncing 같은 렌더링 최적화 기법 실전 적용 경험
- React 렌더링 메커니즘에 대한 깊이 있는 이해

느낀점 & 참고 자료

느낀점

7개월 동안 팀 프로젝트를 진행하며 팀워크는 저절로 만들어지지 않는다는 것을 깨달았습니다. 서로의 경험이 달라 의견 충돌도 있었고, 의도를 충분히 이해하지 못해 어려움을 겪기도 했습니다.

하지만 대화를 통해 작은 오해도 금방 풀 수 있었고, 문제 해결보다 중요한 것은 서로를 이해하려는 태도라는 것을 배웠습니다.

이번 프로젝트를 통해 기능을 만드는 것뿐 아니라, 팀이 함께 성장하는 과정의 가치를 느낄 수 있었습니다.

기타 활동

FIX 2025 대한민국 ICT 융합 엑스포 부스 운영	2025.10.22~2025.10.25
2025 ITCE 프로젝트 3등 수상	2025.10.24

참고 자료

Sync-Desktop 소스코드	: https://github.com/AICT-SYNC/sync-desktop
Sync Design System 소스코드	: https://github.com/AICT-SYNC/sync-design-system
Sync Design System NPM	: https://npmjs.com/package/sync-design-system
Sync Introduce 소스코드	: https://github.com/AICT-SYNC/sync-introduce
sync-saas 홈페이지(서비스 종료)	: https://sync-saas.com

사이드 프로젝트

DIA

2025.08.09 ~ 2025.09.09

대구소프트웨어마이스터고 성적 가산출 서비스

주요 업무

- 성적 계산 페이지
- 성적 산출 알고리즘 모듈화

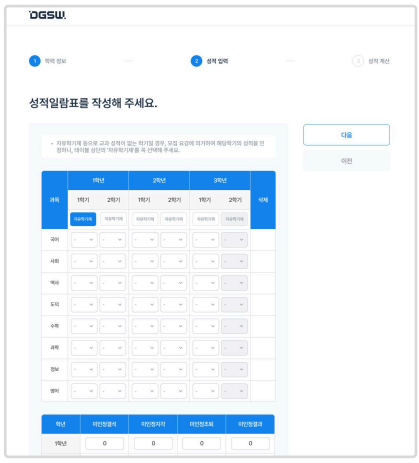
주요 기술

React | TypeScript | styled-components

참고 자료

- 프론트 소스 코드
: https://github.com/EntryCNS/DIA

- 웹 페이지
: https://trydgsw.com



DIA 졸업 예정자 성적 입력 페이지

개요

대구소프트웨어마이스터고 입학 1차 전형의 성적 반영 방식을 기반으로,
학생이 교과 성적을 입력하면 총점을 산출해주는 간단하고 직관적인 웹 서비스입니다.

팀원

- 총원 4명
웹 프론트엔드 4명(본인포함)

배운점

React Context API를 사용해 전역 상태 관리

학생 유형 선택 → 성적 입력 → 결과 확인까지 3단계 폼 개발 중

- 5~6개 컴포넌트를 거쳐 props 전달 (5단계 depth)
- 새 데이터 필드 추가 시 6개 파일 모두 수정 필요

→ **React Context API**로 **전역 상태 관리 중앙화**

- 실시간 동기화: 여러 컴포넌트에서 동일한 상태 공유
- **Props Drilling 제거**: 5단계 전달 → 직접 접근으로 개선

→ 체계적인 상태 관리 설계 및 유지보수 용이한 아키텍처 구축

Redux, Zustand 대신 React Context API 선택 이유

- 프로젝트 요구사항이 “여러 페이지 간 폼 데이터 공유” 정도로 단순하여,
Redux의 action/reducer 구조나 Zustand의 selector 기반 구조까지 필요하지 않았음.
- 상태 변경 빈도가 낮아 Redux 또는 Zustand의 미세한 상태 분리 기능이 필요하지 않았고,
Context API만으로도 충분한 성능을 확보할 수 있었음.

```
Before (Props Drilling)

[App] - grades, setGrades, freeSem, attendance... (12개 props)
  ↓
[Router] - (12개 props 그대로 전달)
  ↓
[StudentWritePage] - (12개 props 그대로 전달)
  ↓
[Body] props만 전달, 사용 안함 - (12개 props 그대로 전달)
  ↓
[WriteGrade] - grades, setGrades 사용
  ↓
[WriteGradeListItem] - grades 사용
```

→

```
After (Context API)

[App]
  └ [ScoreProvider] 전역 상태
        ├ [StudentWritePage] → useScore()
        ├ [WriteGrade] → useScore()
        └ [WriteGradeListItem] → useScore()
```

사이드 프로젝트

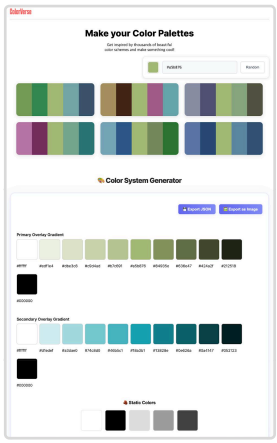
ColorVerse

색 조합을 추천해주는 웹 서비스

2025.06.21 ~ 2025.06.25

주요 기술

React | styled-components | Express.js | OpenAI API



홈 화면

참고 자료

- 소스 코드
: https://github.com/kwondongwoo0424/ColorVerse

- 노션 문서
: https://buly.kr/GvoBKj0

개요

디자이너와 개발자가 일관된 색상 시스템을 빠르게 구축하고,
AI의 도움을 받아 조화로운 색상 조합을 찾을 수 있도록 도와주는 웹 서비스입니다.

팀원

- 총원 1명

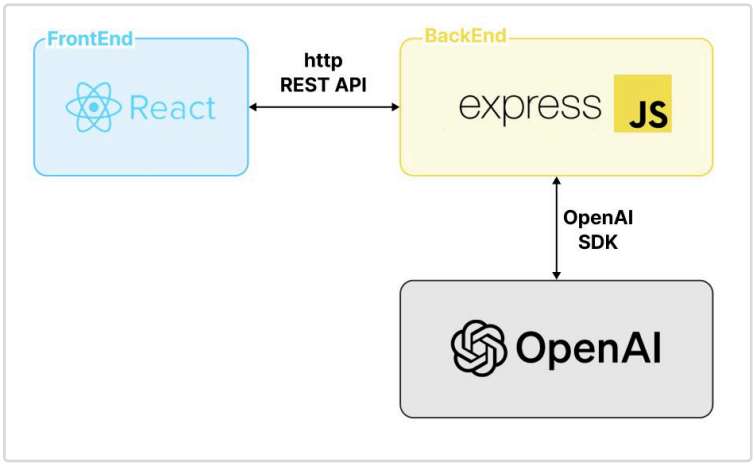
배운점

Express.js를 활용한 RESTful API 서버 구축

- /api/chat POST 엔드포인트로 OpenAI API 연동
- CORS 설정으로 프론트엔드 - 백엔드 간 통신 구현

OpenAI API 통합 및 프롬프트 엔지니어링

- OpenAI 공식 Node.js SDK(openai)로 GPT-3.5-turbo 모델 연동
- API 키 인증 → 요청 구성 → 응답 처리까지 전체 호출 흐름 구현



사이드 프로젝트

Flick

2025.05.05 ~ 2025.05.20

학교 축제 부스 운영을 위한 전자 화폐형 실시간 거래•정산 플랫폼

주요 업무

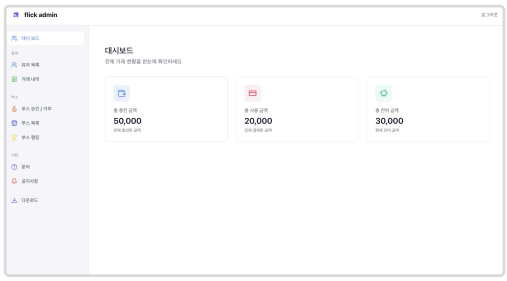
- 어드민 페이지 개발

주요 기술

Next.js | tailwind CSS | TypeScript

참고 자료

- 프론트 소스 코드
: https://github.com/EntryCNS/flick-web



Flick 어드민 대시보드 페이지

개요

Flick은 교내 축제 부스의 결제를 위한 포인트 기반 거래 서비스입니다.
작년에 선배들이 개발한 v1은 QR 코드 결제 방식에서 몇 가지 불편함이 있었고,
이를 개선하고 사용자 경험을 향상시키기 위해 Flick v2를 새롭게 개발하였습니다.

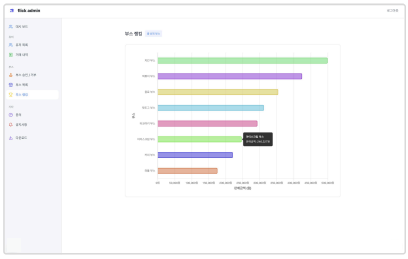
팀원

- 총원 5명
웹 프론트엔드 2명(본인포함), 백엔드 2명, 풀스택 1명

웹 개발

실시간 부스 랭킹 시스템 개발

- Chart.js와 WebSocket을 활용한 실시간 매출 순위 시각화 시스템 구축
- WebSocket 재연결 로직으로 안정적인 실시간 데이터 통신 구현



유저 앱 화면

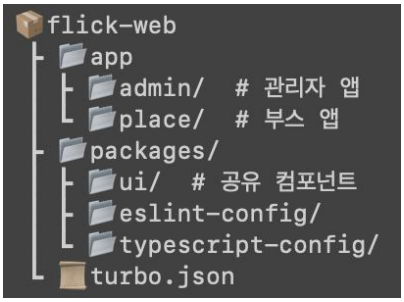
배운점

프로젝트의 코드베이스 통합 및 관리 효율화

Turborepo와 pnpm Workspace 기반의 **모노레포 아키텍처**로 관리자 대시보드와 부스 운영자용 두 개의 Next.js 앱을 통합 관리하며, 공통 코드를 재사용해 개발 효율을 높였습니다.

- @repo/ui: 공통 컴포넌트 중앙화로 코드 중복 제거
- pnpm 하드링크로 node_modules 용량 절감
- workspace:* 프로토콜로 패키지 간 의존성 자동 관리

→ 멀티 프로젝트 환경에서 생산성과 일관성을 동시에 확보한 경험



flick-web 프로젝트 구조

외부 프로젝트 기여 경험

Legacy (legacy-web)

개요 및 프로젝트 소개

Legacy-web은 “게임으로 배우는 우리나라 문화와 역사”를 목표로 개발된 웹·앱 기반 학습 서비스입니다.
본 프로젝트는 총 7명이 참여하였으며, 웹 파트는 2인으로 구성되어 있습니다.
저는 외부 기여자로서 코드 품질 및 보안 측면에서 개선할 수 있는 부분을 발견하여 기여를 진행하였습니다.

기여 내용

카카오 로그인 버튼 컴포넌트 리팩터링

기존 문제점

- Google·Apple 로그인 버튼은 컴포넌트화되어 있었으나 카카오 로그인 버튼만 페이지 내부에 직접 작성되어 있었음
- OAuth 버튼 구조의 일관성이 깨지고, 유지보수성과 재사용성이 떨어지는 문제가 존재함

개선 사항

- 카카오 로그인 버튼을 별도 컴포넌트로 분리
- 기존 OAuth 버튼들과 동일한 구조로 컴포넌트 구조를 일원화

적용 결과

- OAuth 로그인 버튼 전반의 구조적 일관성 확보
- UI 컴포넌트의 재사용성 향상
- 유지보수 효율 및 개발자 경험(DX) 개선

카카오 API 키 하드코딩 보안 취약점 발견 및 이슈 제기

발견한 문제

- 카카오 REST_API_KEY 및 JS_KEY가 코드에 하드코딩된 상태로 공개 저장소에 노출됨
- 외부에서 API 키를 악용할 수 있는 보안 위험 존재

제안한 해결 방안

- 노출된 API 키 즉시 재발급 요청
- 환경 변수 기반의 키 관리 구조로 전환

처리 결과

- 제기한 이슈가 팀 내에서 즉시 처리되며 보안 위험 해소
- 환경 변수 기반 관리로 전환되며 보안 수준 및 유지보수 안정성 향상

기여 결과

이번 기여를 통해 OAuth 기반 서비스에서 API 키를 안전하게 관리하는 것이 얼마나 중요한지 깨달았으며,
보안 사고는 작은 실수에서도 발생할 수 있음을 실감했습니다. 또한 환경 변수 관리와 Git 히스토리 정리 등
보안 친화적인 개발 환경을 갖추는 방법을 익히며, 개발자로서 보안 책임을 더 깊이 인식하는 계기가 되었습니다.

참고 자료

Pull Request: https://github.com/TeamDetail/legacy-web/pull/75
Issue: https://github.com/TeamDetail/legacy-web/issues/76
Repository: https://github.com/TeamDetail/legacy-web

외부 프로젝트 기여 경험

Starthub (starthub-web)

개요 및 프로젝트 소개

StartHub는 AI 기반으로 맞춤형 창업 지원 정보를 제공하는 서비스로,
웹 애플리케이션은 React와 TypeScript를 기반으로 개발되어 있습니다.
본 프로젝트는 퍼블릭 GitHub 레포지토리에서 운영되고 있습니다.
저는 서비스를 직접 이용하던 중 사용자 경험을 저해하는 문제를 발견하여 외부 기여자로 참여하였습니다.

기여 내용

공고 상세 페이지 '좋아요' 기능의 낙관적 업데이트 적용

기존 문제점

- 공고 상세 페이지에서 좋아요/취소 클릭 시 서버 응답 전까지 UI가 반응하지 않음
- 클릭이 정상 처리됐는지 확인하기 어려워 UX가 저하됨
- 네트워크 환경이 나쁠수록 지연이 더 심각하게 체감됨

개선 사항

- React Query useMutation 활용
- 낙관적 업데이트 적용하여 서버 응답 전 UI 상태 즉시 변경
- 요청 실패 시 이전 상태로 롤백 처리해 안정성 확보

적용 결과

- 클릭 즉시 UI가 반응 → 사용자 경험 크게 개선됨
- 네트워크 상태가 불안정한 환경에서도 자연스러운 인터랙션 제공
- 성공·실패 케이스가 모두 고려된 견고한 좋아요 기능 완성

기여 결과

이번 개선을 통해 공고 상세 페이지의 사용자 경험을 실질적으로 향상시켰으며, 클릭 이후 즉각적인 피드백을 제공함으로써 서비스의 반응성을 크게 개선할 수 있었습니다. 또한 사용자의 관점에서 문제를 직접 식별하고 해결한 사례로, 서비스 품질 향상에 기여했다는 점에서 의미 있는 경험이었습니다.

참고 자료

Pull Request: https://github.com/JinInSaDaeCheonMyeong/starthub-web/pull/199
Repository: https://github.com/JinInSaDaeCheonMyeong/starthub-web

자격증 및 기타

제 23회 소프트웨어 역량 검정(TOPCIT) 정기평가 527점(3수준) 2025.05.24

수상

2025 ITCE 프로젝트 3등 수상 2025.10.24

2025 교내 교과 우수상 (운동) 2025.7.22

활동

2025학년도 SW마이스터고 연합 해커톤 참가 2025.11.05 ~ 2025.11.07

FIX 2025 대한민국 ICT 융합 엑스포 부스 운영 2025.10.22~2025.10.25

2025 교내 소프트웨어 해커톤 참가 2025.07.15 ~ 2025.07.16

2025 교내 나르샤 프로젝트 참가 2025.07.15 ~ 2025.07.16
- Sync Desktop

2025 교내 글로벌 1:1 원어민 화상영어 참가 2025.05.13~2025.12.23(예정)

교내 도서부 부장 2024.12~

2025학년도 신입생 2차 면접 봉사 참여 2024.10.25

2024 교내 나르샤 프로젝트 참가 2024.08.20~2024.12.26

2024 교내 소프트웨어 해커톤 참가 2024.07.17 ~ 2024.07.18

2024 교내 글로벌 1:1 원어민 화상영어 참가 2024.05.8~2025.01.08

교내 전공 동아리 연합 코딩테스트 참가 2024.04.17

2024 스마틴앱 챌린지 참가 2024.04.02

교내 전공 동아리 CNS 입부 2024.03.14

끝까지 읽어주셔서 감사합니다.
아직 많이 부족하지만, 매일 한 걸음씩 성장하는 개발자 권동우가 되겠습니다.