강민재

프론트엔드 개발자

minijae011030@gmail.com

안녕하세요.
빠르게 성장하는 개발자 강민재입니다.

자기소개

문제를 근본적으로 해결하는 개발자입니다.
개발 과정에서 단순히 동작하는 화면을 만드는 데서 멈추지 않고, 반복되는 불편이나 비효율을 근본적으로 해결하는 데 집중해왔습니다. React Native 프로젝트에서는 Alert 로직이 각 컴포넌트마다 중복 작성되어 유지보수가 어려웠는데 이를 공통 유틸로 모듈화하여 Alert 관련 코드량을 약 70% 줄였습니다. 그 결과 코드 가독성과 유지보수성이 크게 개선되었고, 실제 팀원들의 개발 속도 역시 향상되었습니다.
품질과 성능을 고려한 코드를 작성합니다.
코드의 완성도는 '동작'에서 끝나는 것이 아니라 유지보수성, 성능, 일관성까지 포함된다고 생각합니다. HealthKit 데이터를 활용한 기능 개발 과정에서는 불필요한 중복 요청으로 서버와 클라이언트 모두에서 성능 저하가 발생했는데, 데이터 요청 방식을 개선해 중복 전송량을 80% 줄였습니다. 이를 통해 메모리 사용량을 낮추고 응답 속도를 개선하면서 사용자 경험을 안정적으로 보장할 수 있었습니다.
협업 속에서 가치를 만들어냅니다.
프론트엔드 2명, 백엔드 3명으로 진행된 프로젝트에서는 API 반영 여부를 확인할 수 없어 작업 지연이 반복되는 문제가 있었습니다. 저는 노션 기반 협업 보드를 도입해 API 개발 현황과 반영 여부를 모두가 투명하게 공유하도록 개선했습니다. 그 결과 팀 전체의 소통 비용이 줄어들고, 막힘 없는 개발 플로우가 가능해졌습니다. 이러한 경험은 좋은 코드는 협업 구조 속에서 더욱 빛난다는 것을 체감하게 했습니다.

기술 스택

HTML/CSS, JavaScript, TypeScript, React, React Native, Next.js, Vue.js, Figma, Git, GitHub, storybook

---

대외활동

KB IT's Your Life 해커톤 / 장려상(4등)

KB 국민은행

2025

멀티캠퍼스와 KB국민은행이 주관한 KB IT’s Your Life 부트캠프 해커톤에 참여하여 팀 프로젝트를 수행했습니다.
프론트엔드 개발을 담당하며 서비스 UI 구현과 사용자 경험 개선을 주도했고, 백엔드·디자인 파트와 협업해 최종 결과물을 완성했습니다.
제한된 시간 안에 핵심 기능을 구현하고 완성도를 높인 점을 인정받아 장려상(4등)을 수상했습니다.

SFM 외식서비스 앱 기획 및 개발 경진대회 / 우수상(1등)

세종대학교 LINK 3.0

2022

세종대학교 LINK 3.0에서 주관한 외식서비스 앱 기획 및 개발 경진대회에 참가하여 배공(배달을 공유하자) 서비스를 기획·개발했습니다.
배달비를 절약할 수 있도록 이웃과 함께 주문을 공유하는 기능을 설계하고, 실제 동작 가능한 프로토타입까지 구현하여 실현 가능성을 입증했습니다.
사용자 편의성과 서비스 차별성을 인정받아 우수상(1등)을 수상했습니다.

포트폴리오

링크

프로젝트 포트폴리오

프로젝트

MOS | 스터디 팀 모집·생성부터 일정 관리까지 통합된 기능을 제공하는 웹 플랫폼

개인

2025. 03. ~ 2025. 09.

프로젝트 개요

스터디 모집과 관리 과정을 돕는 웹 서비스로, 사용자가 스터디를 개설하고 참여하며, 진행 상황을 효율적으로 관리할 수 있도록 지원하는 플랫폼입니다.

프로젝트 저장소: 프로젝트 깃허브 링크

인원 구성: FE 3명, BE 2명

기술 스택: Typescript, Next.js, Zustand, Tailwind

주요 업무

- Atomic Design과 FSD 기반의 프로젝트 폴더 구조 설계
- React Hook Form 기반의 멀티스텝 스터디 생성 폼 구현
- 프론트엔드-백엔드 간 API 명세 조율 및 협업 커뮤니케이션 중심 역할 수행
- 토큰 기반 인증 흐름 및 HTTP-only 쿠키 기반 oAuth 로그인 로직 구현

트러블 슈팅

1. 협업 효율을 고려한 폴더 구조와 상태 관리 설계

3인의 프론트엔드 개발자가 동시에 작업하기 위한 구조적 기반이 필요했습니다. 초기에는 디렉터리 혼잡과 관심사 분리가 부족해 유지보수가 어렵다는 문제점이 있었습니다.
>> Atomic Design과 FSD를 접목한 폴더 구조를 설계하여 관심사 분리를 명확히 했고, 각 기능 단위별로 컴포넌트, 상태, API, 스타일 파일을 구조화하여 협업 효율을 높였습니다.

2. 서로 다른 역할 간의 이해를 돕는 커뮤니케이션 브릿지 역할 수행

백엔드는 프론트엔드 개발 경험이 부족하고, 프론트엔드는 백엔드 시스템 구조에 익숙하지 않아 의견 차이가 자주 발생했습니다. 서로의 관점을 정확히 이해하고 반영하는 커뮤니케이션이 필요했습니다.
>> 양쪽 팀의 기술 배경과 의도를 파악하고, 용어와 요구사항을 서로의 언어로 풀어 설명하는 중간 다리 역할을 수행했습니다. 실시간 회의와 문서 정리를 통해 오해를 줄였고, 기능 구현에 있어 공통된 이해를 형성해 원활한 협업을 이끌었습니다.

3. 입력 데이터 유실을 방지한 멀티스텝 폼 구조 개선

스터디 생성 과정에서 입력 단계가 많아 실수로 새로고침하거나 페이지를 이탈하면 모든 입력값이 초기화되는 문제가 발생했습니다.
>> 스터디 만들기 폼은 여러 단계로 구성되어 있어 사용자가 입력 도중 실수로 새로고침하거나 페이지를 이탈하면 입력값이 모두 초기화되는 문제가 있었습니다. 이로 인해 사용자 경험이 크게 저하되었고, 다시 처음부터 입력해야 하는 불편함이 발생했습니다. 이를 해결하기 위해 localStorage를 활용하여 각 단계별 입력값을 실시간으로 저장하고, 컴포넌트 마운트 시 기존 값을 복원하는 로직을 구현했습니다. 또한 페이지가 완전히 이동되었을 때만 저장된 값을 초기화하도록 처리하여, 의도하지 않은 데이터 삭제를 방지했습니다.

4. 로그인 만료 및 상태 초기화 문제 해결을 통한 UX 개선

페이지를 새로고침하거나 브라우저를 닫았다 열 경우, 로그인 상태가 초기화되어 인증된 사용자도 보호된 페이지에 접근할 수 없거나, 로그인 유지 여부에 혼란이 생기는 문제가 발생했습니다.

>> Next.js App Router 환경에서 인증이 필요한 (auth) 디렉토리 접근 시 서버에서 쿠키 기반 토큰을 검사하고, 토큰이 없으면 클라이언트를 홈으로 리다이렉트하도록 구성했습니다. 또한 Zustand와 localStorage를 연동하여 로그인 성공/실패 여부 및 토큰 만료 여부를 저장하고, 클라이언트 단에서는 이 정보를 기반으로 토스트를 출력하여 사용자에게 명확하게 상태를 전달하도록 구현했습니다. 이를 통해 새로고침, 브라우저 종료/재접속 등 다양한 상황에서도 로그인 흐름이 끊기지 않도록 UX를 개선했습니다.

Archive | 웹 블로그 개발 프로젝트

개인

2024. 07. ~ 2025. 04.

프로젝트 개요

개인 블로그 서비스 개발 프로젝트로, 글 작성·조회·검색 기능을 갖춘 웹 애플리케이션입니다. 단순 CRUD를 넘어 SEO 최적화, 보안 강화, 서버 운영까지 실제 서비스 운영 환경을 고려해 설계한 것이 특징입니다.

프로젝트 저장소: 프로젝트 깃허브 링크

인원 구성: FE 1명, BE 1명

기술 스택: Typescript, Next.js, Zustand, Tailwind

주요 업무

- React를 활용하여 개인 블로그 개발 프로젝트의 프론트엔드 전반을 담당
- React Query를 활용한 API 캐싱 및 서버 상태 관리 최적화
- HttpOnly Cookie를 이용한 보안 강화된 JWT 기반 인증 처리
- SEO 최적화를 위해 react-helmet을 활용한 메타데이터 동적 설정
- 기존 React SPA를 App Router 기반 Next.js 프로젝트로 마이그레이션
- pnpm + PM2 기반 서버 빌드 및 운영 환경 구성
- Nginx 리버스 프록시 설정을 통한 도메인 연결 및 서비스 안정화

트러블 슈팅

1. 서버 부담을 줄이기 위한 React Query의 활용으로 서버 부하 45% 감소

계속된 account 함수 호출로 서버 부하가 증가하였고, 요청 시간이 평균 150ms에서 200ms로 소요되었습니다. 이를 해결하기 위해 전역 상태 관리 방법을 고민하게 되었습니다.

이에 Recoil과 Redux를 고려했으나, 유저 정보 변경을 실시간으로 감지하고 처리하는 것이 어려웠고, 함수 호출 위치도 명확히 할 수 없어 해결이 어려웠습니다.

>> React Query의 캐싱 기능을 활용하여 중복 요청을 줄이고, 데이터 재사용을 통해 서버 부하를 80% 감소시켰습니다. 이를 통해 요청 응답 시간도 평균 50ms로 단축되었습니다.

유저 정보가 변경될 때나 캐시가 만료되었을 때 데이터를 갱신하여 최신 상태를 유지하였습니다.

2. CSR 한계 극복을 위한 메타데이터 설정으로 검색 노출 가능성 개선

CSR 방식으로 구축된 블로그에서 초기 HTML에 콘텐츠가 포함되지 않아, 검색 엔진이 페이지 내용을 크롤링하지 못하는 문제가 발생했습니다.
>> 이를 보완하기 위해 React 환경에서 사용할 수 있는 react-helmet을 활용하여 각 페이지마다 title, description, og 태그 등 메타데이터를 동적으로 설정했고, 그 결과 검색 노출 가능성이 개선되었습니다.

3. Atomic Design 및 FSD 도입으로 프로젝트 구조 정리 및 유지보수성 향상

프로젝트 구조가 기능 단위로 명확하게 나뉘지 않아 컴포넌트 재사용과 유지보수에 어려움이 있었습니다. 특히 공통 컴포넌트와 기능별 컴포넌트의 경계가 불분명해, 의존성이 높아지고 새로운 기능 추가 시 폴더 탐색 시간이 길어졌습니다.
>> 구조 개선을 위해 Atomic Design Pattern을 적용하여 공통 컴포넌트를 atoms, molecules, organisms 단위로 분리하였고, Feature-Sliced Design(FSD)을 참고해 도메인 단위로 기능을 분리했습니다.
기존에는 account 함수가 여러 페이지에서 중복 호출되었지만, 기능별 폴더로 모듈화한 후 한 곳에서 일괄 관리할 수 있도록 리팩토링했습니다. 이 과정에서 기존에 3군데의 폴더에 분산되어 있던 account 관련 로직을 한 곳의 디렉토리에서 관리되게 하여 폴더의 복잡성을 줄이고 유지보수성을 높였습니다.

4. Next.js 기반 SSR 마이그레이션 및 서버 배포 환경 구성

CSR 기반의 React 앱은 SEO와 초기 렌더링 속도 측면에서 한계를 보였고, 정적인 배포 방식은 유연한 서버 설정에 제약이 있었습니다.
>> App Router 기반의 Next.js로 마이그레이션하여 SSR과 동적 메타데이터 설정을 통해 SEO 문제를 개선하고, pnpm + PM2 + Nginx 기반 서버 실행 구조를 도입하여 안정적인 운영 환경을 구축했습니다.

Antiheimer | AI 치매 진단 애플리케이션

세종대학교 Capstone디자인
2024. 08. ~ 2024. 10.

프로젝트 개요

사용자의 건강 데이터를 기반으로 AI 진단을 제공하는 **헬스케어 모바일 애플리케이션**입니다. Apple HealthKit 데이터를 수집·시각화하고, AI 모델의 예측 결과를 실시간으로 반영하여 개인 맞춤형 건강 관리 경험을 제공하는 것을 목표로 했습니다. React Native로 크로스플랫폼 환경을 구현하였습니다.
프로젝트 저장소: 프로젝트 깃허브 링크
인원 구성: FE 1명, BE 2명, AI 1명
기술 스택: HTML, CSS, JavaScript, ReactNative, Recoil, Figma

주요 업무

- React Native 기반 크로스플랫폼 모바일 애플리케이션 개발
- Apple HealthKit 데이터를 활용한 사용자 건강 데이터 시각화 기능 구현

- React Native Maps를 활용한 현재 위치 표시
- JWT를 활용한 사용자 인증 관리 및 보안 강화
- Figma를 이용한 UI/UX 디자인 및 사용자 경험 최적화
- REST API 통신을 통한 AI 모델 예측 결과 처리 및 UI 반영

트러블 슈팅

1. AIert 유틸 함수 모듈화로 중복 코드 70% 제거 및 유지보수성 향상

첫 번째 React Native 프로젝트에서는 AIert 관련 로직이 각 컴포넌트마다 중복되어 작성되었고, 이로 인해 전체 코드가 비효율적으로 길어졌으며 유지보수가 어려운 구조가 되었습니다.
>> 이 프로젝트에서는 AIert 로직을 ConfirmAIert, CancelAIert 유틸 함수로 모듈화하고, 반복되던 AIert 작성 코드를 평균 10줄 → 1줄로 줄이며, 총 50줄 이상의 중복 코드를 제거했습니다. 이를 통해 AIert 관련 코드의 양을 약 70% 줄였고, AIert 스타일 및 동작을 일괄적으로 관리할 수 있게 되어 유지보수성과 확장성이 크게 향상되었습니다.

2. 건강 데이터 요청 최적화로 중복 전송 80% 감소 및 메모리 효율 개선

AI 진단을 위해 약 10일치의 사용자 건강 데이터를 불러와야 했습니다. 하지만 앱을 처음 사용하는 사용자와 기존 사용자 모두에게 동일한 데이터를 요청하여, 기존 사용자에게는 중복된 건강 데이터가 서버에 전송되는 문제가 발생했습니다. 또한 앱을 실행할 때마다 4개의 건강 카테고리에서 10일치의 데이터를 불러오다보니 메모리 낭비 이슈도 발생하였습니다.
>> 이전 로직에서는 예를 들어 2일마다 앱에 접속할 경우, 매번 4개 카테고리 각각에서 10일치 데이터를 요청해 총 40개의 데이터를 중복 전송하게 되었습니다.
이를 해결하기 위해 서버에서 각 카테고리의 마지막 수집일을 받아와 그 이후의 데이터만 요청하는 방식으로 개선하였고, 2일마다 접속하는 유저의 경우 4개 카테고리 × 2일치 = 8개 데이터만 전송되도록 변경했습니다.
그 결과, 중복 전송 데이터가 기존 대비 80% 감소하였고, 클라이언트에서 처리해야 할 데이터 양도 함께 줄어들어 앱 실행 시 메모리 사용량과 응답 속도 모두 개선되었습니다.

3. 서버 인프라 구축 및 HTTPS 적용 경험으로 서비스 전체 흐름 이해도 향상

팀 내에 서버를 다룰 수 있는 인원이 없어, 프론트엔드 외에 서버 관리까지 맡게 되었습니다. 원활한 프로젝트 진행을 위해 배포 환경 구축 경험이 없던 상황에서 HTTPS 적용과 도메인 연결까지 일주일 내로 완료해야 했습니다.
>> 백엔드 서버(Spring Boot)와 AI 서버(Python 기반)를 직접 관리하며, 클라이언트와의 통신을 위한 인프라를 구축했습니다.
Ubuntu 환경에서 Nginx를 이용해 리버스 프록시 및 HTTPS 인증서를 적용했고, 도메인을 연결하여 외부에서도 접근 가능한 환경을 구성했습니다. 운영 중인 서버는 nohup으로 백그라운드 실행 후, 로그를 확인하며 모니터링하였습니다.
이 경험을 통해 인프라 환경 구성에 대한 감각을 키울 수 있었고, 프론트엔드뿐 아니라 서비스 전체 흐름을 고려한 개발이 가능해졌습니다.

자격증

토익

715점 | 한국TOEIC위원회

2024. 09.

교육

멀티캠퍼스/KB IT's Your Life

수료 | 사설 교육 | 프로그래밍

2025. 03. ~ 2025. 08.

세종대학교

졸업 | 대학교(학사) | 외식경영학과/컴퓨터공학과

2020. 03. ~ 2025. 02.