지금 만족하고 있어요

최기환

풀스택 개발자

소프트웨어 엔지니어 최기환 이라고 합니다. 컴퓨터만 있다면 세상의 다양한 문제를 해결하는데 기여할 수 있는, 무에서 유를 창조하는것 같은 직업의 매력에 이끌려 개발자의 길을 걷게 되었습니다.

자기소개

기술 블로그 | 공부 기록 블로그 | 기술 블로그(구) | 포트폴리오

안녕하세요 빠르게 성장하는 개발자 최기환 이라고 합니다. 돈의 추월차선이라는 책을 읽으며 웹 개발에 대해 알게 되었고 생활 코딩의 강의를 들은 후 무에서 유를 창조하는 소프트웨어 개발의 매력에 빠져 개발자의 길을 걷게 되었습니다.

소프트웨어 개발을 통해 문제를 해결하는 것을 좋아합니다. 다양한 사이드 프로젝트를 진행했지만 유저 도입에 실패해 버려지는 코드를 작성하는것이 싫어 프리랜서 개발을 시작했습니다. 프리랜서 개발을 시작하며 내가 좋아하는것은 개발로 누군가의 문제를 해결하는 것이구나 라는걸 깨달았습니다.

성장하기 위해 노력합니다. bookSailor라는 독서 스터디를 운영하고 있으며 최근에 "프로그래머의 뇌"라는 책을 주제로 독서 스터디를 시작했습니다. 꾸준하게 노력해 성장하는 방식도 좋지만, 같은 시간 노력해도 효율적으로 노력하고 싶었습니다. 그렇기에 "프로그래머의 뇌라는 책"을 읽기로 결정 했고, 이 책을 얻은 정보가 복리가 되어 성장을 가속화 시켜줄거라 생각하고 있습니다.

고민하는것을 좋아합니다. 최근에는 리액트에서의 클린코드에 대해 고민해 보았습니다. 그 안에서도 `props`의 드릴링이 무조건 나쁜가, 컴포넌트 인터페이스의 설계 방법 등이 있었습니다. 스스로 깊은 고민을 해보.. 결과 리액트의 컴포넌트는 순수 함수라는 점에서 `props`의 명시적인 전달은 데이터의 추적과 관리에 도움이

되지만 `props`의 전달이 강한 의존 관계를 만들어 문제가 되는 상황이 있음을 깨달은 경험이 있습니다. 이 당시의 다양한 고민의 흔적을 블로그에 기록해 두었습니다

프로젝트

Blooming

DnD
2024.07. ~ 진행 중

IT 연합 동아리 DnD에서 진행한 프로젝트 입니다. 초보 식집사들을 위한 식물 키우기 가이드와 물, 비료 주기 알림 기능을 제공합니다. 현재 웹 뷰 형태로 앱 스토어, 플레이 스토어에 배포를 준비하고 있습니다. 플레이 스토어의 경우 현재 비공개 테스트를 진행중입니다.

배포 | 문서 | 깃허브 | RN 깃허브 | 스토리북

스택: React, Tanstack Query, TypeScript, Vite, TailwindCSS, Framer Motion, MSW, PWA, ReactNative, Expo

멤버: 디자이너(2) / 백엔드(2) / 프론트엔드(2)

기여도: 80%

성과: 최종 발표회 1등

ReactNative 기반의 웹 뷰 앱 개발

- PWAbuilder를 활용한 안드로이드 앱 빌드에서 푸쉬 알림 기능이 동작하지 않는 문제가 있었습니다. RN, Expo 기반의 웹뷰 앱으로 개발하고, 로그인 후 메인페이지 접속시 웹 뷰에서 ReactNative로 `window.ReactNativeWebView.postMessage` 메서드를 사용해 `accessToken`을 보내도록 했습니다. ReactNative 에서는 액세스 토큰을 받아, 푸쉬 알림 허용 다이얼로그를 띄우고 디바이스 토큰을 생성합니다. 생성된 디바이스 토큰과 액세스 토큰을 사용해 유저 디바이스 토큰 등록 API를 호출하도록 했습니다. 웹 환경에서는 `ReactNativeWebView` 객체의 반환 값이 `undefined` 라면 웹 환경의 푸쉬 알림을 등록하도록 구현했습니다. 이 경험을 통해 웹뷰 환경과 네이티브 환경이 데이터를 주고 받는 방법에 대해 학습할 수 있었고, 네이티브와 웹뷰 환경에서의 데이터 통신 로직 설계에 대한 경험을 할 수 있었습니다.
- PWAbuilder를 활용 안드로이드 앱 배포 과정에 리소스 낭비가 심함을 느꼈습니다. Google Cloud Developer API와 Expo, EAS 기반의 배포 환경을 구성했으며 간단하게 `eas build`, `eas submit` CLI 를 사용해 번들을 업로드하고 출시 버전을 업데이트 할 수 있도록 했습니다.

성능 최적화

기존 Lighthouse 성능 지표가 68점이 나오던 문제가 있었습니다. 이에 대한 원인을 파악했고 번들 사이즈 크기, 폰트 파일 사이즈 크기, 이미지 로딩 등의 문제가 있음을 발견했습니다. 이를 다음과 같은 방식을 통해 해결했습니다.

- 애플 로그인에 필요한 외부 스크립트와 같은 스크립트에 `async` 태그를 사용해 스크립트의 로드와 실행이 렌더링을 블로킹 하지 않도록 했습니다. 애플 로그인 스크립트의 경우 언제 실행되어도 상관 없다고 생각했기 때문에 `defer` 가 아닌 `async` 를 사용했습니다.
- svg는 프로젝트 자바스크립트 번들에 포함되며, 이미지의 로딩은 LCP에 영향을 주기 때문에 최적화가 필요함을 느꼈습니다. `Vite Image Optimizer` 플러그인을 사용해 svg와 이미지 최적화를 진행해 1.4MB 에서 0.99MB 로 약 60% 사이즈를 줄일 수 있었습니다.
- 압축률이 큰 `woff2` 형식의 폰트를 사용하고, 서브셋 폰트를 사용해 폰트 사이즈를 줄였습니다. 이에 더해 폰트 로딩이 렌더링을 막지 않도록 font-diplay 옵션에 swap을 적용했습니다.
- React.lazy를 사용해 코드 스플리팅을 적용했습니다. 초기 렌더링에 필요한 컴포넌트만 자바스크립트 번들에 포함되도록 하였습니다. 또한 다음 페이지로의 이동 시간 단축을 위해, 다음 페이지로 이동할 가능성이 높은 페이지에 필요한 데이터를 ReactQuery의 prefetch를 통해 미리 받아와 캐싱하도록 했습니다.
- HTML, CSS, JavaScript 파일에 GZIP 압축을 적용했습니다.
- Vite의 롤업 빌드시 메뉴얼 청킹을 적용했습니다. 이 과정에서 무조건 적인 청킹은 네트워크 요청 빈도를 늘리며 오히려 성능 저하를 가져온다는걸 깨달아, 기능 단위의 청킹을 적용했습니다. 예를들어 zod, react-hook-form과 같은 식물 등록 페이지에는 필요하지만 초기 렌더링 되는 페이지에는 필요하지 않은 라이브러리이기에 'form'을 단위로 함께 청킹 되도록해 식물 등록 페이지를 렌더링할 때 모두 불러올 수 있도록 했습니다.
- 데이터 요청 후 콘텐츠를 그리게 되는데, 데이터를 요청하는 동안 콘텐츠를 그릴 수 없어 LCP 성능이 떨어지는 문제가 있었습니다. 연결 설정 시간의 대부분은 데이터 교환이 아닌 기다리는데 사용 된다고 한다는것을 알게 되어. link 태그의 preconnect를 사용해 미리 서버 도메인에 연결할 수 있도록 했습니다.

이러한 작업들을 통해 폰트 사이즈 700KB -> 200KB, JS 번들 사이즈 398KB 에서 285KB로 줄일 수 있었으며, Lighthouse 성능 지표를 68점에서 100점으로 만들어낼 수 있었습니다. 이 과정을 통해 웹 성능 최적화 방법에 대해 깊이 이해할 수 있었으며, 블로그 게시글에 정리함으로써 레퍼런스로 활용할 수 있도록 했습니다.

디자인 시스템 구현

- 빠른 MVP 배포를 위해 디자인 시스템을 빠르게 구현할 필요가 있었습니다. 이전 인프런의 기술 블로그를 참고해 [illegible]하기로 결정했습니다.

MUI, ShandCN, MantineUI 등의 다양한 라이브러리를 조사한 후 커스터마이징의 간편성, TailwindCSS 와의 호환성 등을 고려해 ShandCN 라이브러리를 사용했습니다.
- 구현한 디자인 시스템의 QA 진행을 원활히 하기 위해, vercel을 사용해 storybook을 배포했습니다. 모바일 사이즈 전용 앱이기에 앱 사이즈의 레이아웃을 기본 레이아웃으로 적용해 레이아웃을 디자이너가 스토리북에 대해 모르더라도 확인하고 이용할 수 있도록 했습니다.

백엔드 개발과의 의존성 제거

- 프로젝트를 진행하며 백엔드의 개발에 프론트엔드 개발이 의존된 관계를 피하고자 했습니다. MSW를 사용하고, 노션을 기반으로 API문서를 작성하도록 했습니다. 스웨거 문서가 아닌 노션을 참고해 응답을 모킹해 백엔드 개발 의존성을 성공적으로 제거할 수 있었습니다.

아이콘 스토리북 생성 자동화

- 피그마 디자인에 사용되는 아이콘들이 정리되어있지 않았습니다. 따라서 개발하며 하나하나 export 하는 방식으로 svg 아이콘을 사용했습니다. 프로젝트를 진행하던 중 중복되는 아이콘들이 추가되고 어떤 아이콘들이 있는지 확인하기 어렵다는 문제가 있었습니다. Vite의 import.meta.glob 메서드를 `assets/icon` 폴더 내에 존재하는 모든 모듈을 불러오고 이 스토리북 컴포넌트에 이 모듈들의 타입(svg, tsx)에 따라 렌더링 되게끔 했습니다. 그 결과 아이콘의 스토리북 생성을 자동화하고, 손쉽게 아이콘을 추가, 제거, 수정할 수 있는 환경을 구축했습니다. - 블로그 게시글

CI/CD 자동화

- husky를 사용한 자동화에서 모든 파일에 prettier, eslint 명령어가 실행되는 문제가 있었습니다. lint- stage 라이브러리 도입을 통해 staging된 파일들만 prettier, eslint 명령어 실행되도록 구현해 해결했습니다.
- PR 생성시 리뷰어 할당, 테스트 결과 확인등 자동화할 수 있는 부분들을 메뉴얼하게 진행함에 리소스가 낭비됨을 느꼈습니다. Github Actions을 사용해 테스트 결과, 커버리지 리포트, 빌드 결과 등을 댓글로 생성해 한 눈에 확인할 수 있게 했으며, 리뷰어 할당을 자동화해 적극적인 PR 리뷰 문화를 만들어 낼 수 있었습니다.

포트폴리오 웹사이트 개발

개인
2024.04. ~ 진행 중

진행하는 프로젝트들을 정리하고 회고하기 위한 목적으로 개발한 개인 웹사이트 입니다.

깃허브 | 배포

스택: Next.js, Tanstack Query, TypeScript, TailwindCSS, Zustand, Framer Motion, PostgreSQL, Prisma

에러 핸들링

- 프로젝트를 정리하기 위해 게시글을 작성하다 에러가 발생하며 공들여 작성한 게시글이 모두 삭제되는 현상을 발견했습니다. 우선 예기치 못한 에러가 발생하더라도 최악의 사태를 막기 위해서 실시간으로 저장할 필요가 있다고 생각했습니다. 게시글의 내용이 변경될 때 내용을 DB에 저장하도록 했습니다. 또한 저장을 위해 잦은 요청이 발생하는 것을 막기 위해 debounce를 적용했습니다. 그리고 ErrorBoundary를 통해 에러가 발생하더라도 작성중인 내용을 저장 버튼을 통해 클립보드에 저장할 수 있도록 했습니다. 이 경험을 통해 예기치 못한 에러가 사용자 경험에 미치는 영향에 대해 깊게 고민하며 에러 관리를 통해 사용자 경험을 개선하는 방법에 대해 알 수 있었습니다.
- 여러개의 이미지 업로드시 요청 본문의 크기가 커지며 에러가 났습니다. 이터러블을 원하는 사이즈로 쪼갤 수 있는 chunk 함수를 구현해 특정 파일 갯수 단위로 요청을 나눈 후 보내도록 하여 해결했습니다.
- 사용자의 입장에서 웹을 사용하며 다양한 에러들을 만나게 되었습니다. 그러다 "만약 내가 사용자가 아니라 모르는 사용자가 앱을 사용하면서 오류를 만나면 그 오류를 어떻게 개선할 수 있을까?" 에 대한 고민을 하게 되었습니다. 이를 해결하기 위해 Sentry를 사용해 에러를 모니터링 할 수 있도록 했습니다. 재현하기 힘들거나, 고객의 입장에서 알기 힘든 에러의 발생 경로 등에 대한 에러 데이터를 저장해 데이터로부터 오류를 찾고 해결할 수 있는 환경을 구성했습니다.
- Zustand의 상태가 페이지 이동시에도 초기화 되지 않아 에러가 발생했습니다. 전역 상태 관리의 사이드 이펙트를 최소화 하고 싶다는 생각이 들었습니다. ContextAPI를 사용해 DI 패턴을 구현해 전역 상태가 특정 Provider 컴포넌트의 생명 주기와 함께 하도록 했습니다. 이를 통해 전역 상태를 안전하게 초기화 할 수 있는 환경을 구성했습니다.

SEO

- 포트폴리오 웹사이트 이기 때문에 SEO가 중요하다고 생각했습니다. robot.txt, sitemap.xml, 적절한 메타태그 사용해 SEO를 개선할 뿐만 아니라, 동적으로 SiteMap과 jsonLD를 생성하도록 했습니다.

태성 환경 연구소 모바일 앱 개발 - 부산 연합 프로젝트 경진대회

태성환경연구소
2023.08. ~ 2024.04.

태성 환경 연구소는 악취 [illegible] 탐지 결과를 모니터링 하고 악취 탐지기를 원격으로 [illegible]. 부산 연합 프로젝트

경진대회에서 이 앱을 통해 2등이라는 성과를 기록했고, 태성 환경 연구소의 요청으로 앱 개발 외주를 진행했습니다.

문서 | 시연영상

멤버: 디자인(1) / 개발(1)

성과: 부산 연합 프로젝트 경진대회 2등, 200만원 외주 계약

앱 개발

깃허브

스택: Flutter, Riverpod, Freezed, Firebase

- 소켓 통신을 사용해 실시간 모니터링 UI를 개발했습니다. Riverpod을 사용해 소켓 통신의 연결을 관리하도록 해, 많은 소켓 통신을 모듈 단위로 관리할 수 있도록 했습니다.
- Freezed Anotation을 사용해 간편하게 Data Object Model을 생성하도록 했습니다.
- 도메인 지식이 부족해 앱 기획이 어려웠으며 이미 구축되어 있는 악취 모니터링 서비스의 인프라에 팀에서 개발한 서버 및 앱이 추가되어야 하는 상황이었습니다. 회사 홈페이지의 카탈로그를 보며 악취 탐지기 관련 도메인 지식의 이해도를 높이고, 회사 담당자와 지속적으로 소통하며 인프라가 어떻게 구성되어 있는지에 대한 지식을 쌓았습니다. 그 결과 성공적으로 앱, 서버 개발에 성공했고 부산 연합 프로젝트 경진대회에서 2등이라는 성과를 만들어 냈습니다. 여기서 끝나지 않고 앱, 서버 개발을 완성해 사용 설명서와 함께 납품하는데 성공했습니다.

서버 개발

깃허브

스택: Nest.js, Prisma, Firebase, Azure

- 데이터 베이스의 변경을 감지해 소켓통신 이벤트를 발생시키는 방식으로 실시간성을 유지해야 했습니다. 이를 구현하기 위한 방식으로 CDC와 풀링이 있다는걸 파악했습니다. CDC 방식은 개발 리소스가 필요하며, 마감기한까지 적용하기 어렵다고 판단해 풀링 방식을 적용해 실시간 데이터베이스 감지 기능을 개발했습니다.
- Firebase를 사용해 푸쉬 알림 기능을 구현했습니다. 사용자의 기기별 알림 설정에 따라 푸쉬 알림을 보내도록 구현했으며, Chron을 활용한 작업 스케줄링을 통해 일정 시간 간격으로 새로운 알림 데이터를 파악하고 보낼 수 있도록 구현했습니다.

CodeforChanges - 구글 솔루션 챌린지

구글
2024.01. ~ 2024.02.

구글에서 주최하는 개발을 통해 17가지의 SDGs중(인류 보편적 문제, 경제 문제, 환경 문제) 하나를 해결하는 솔루션을 제안하는 공모전입니다. 쓰레기의 사진을 찍으면 해당 쓰레기가 재활용 가능한지, 어떻게 분리 배출 해야하는지에 대한 정보를 제공하는 서비스를 개발했습니다.

깃허브 | 데모 영상

멤버: FE(2) / BE(2)

인프라 관리

기여도: 100%

스택: Docker, Jenkins, GCE

- 메인 서버, AI서버를 GCE 인스턴스에 배포했습니다. 자동화를 적용하지 않아 새로운 버전의 서버 배포를 수동적으로 하는것에 리소스가 낭비됨을 느꼈습니다. Docker와 Docker Componse를 사용해 컨테이너화 한 후, Jenkins 와 Github Web Hooks를 사용해 CI/CD 자동화를 적용했습니다. 인프라 환경에서의 DX를 개선해 개발 리소스 낭비를 줄였으며, Jenkins와 Docker Hub를 이용한 배포 자동화 방법을 학습할 수 있었습니다.

앱 개발

기여도: 70%

스택: Flutter, GetX, Dio

- 게시글에 좋아요를 누르는 경우와 같이 즉각적인 반응이 중요한 경우 낙관적 업데이트를 적용해 UX를 향상시켰습니다.
- 게시글 업로드에 Infinite Scroll을 적용해 네트워크 리소스 낭비를 줄이고 사용자 경험을 개선했습니다.
- 쓰레기 사진을 찍은 후 결과의 응답을 받을 때 제대로된 값을 받아오지 못하는 문제가 있었습니다. 확인 결과 AI 서버의 부하를 줄이기 위해 RabitMQ를 적용해, 메인 서버에서의 응답과 AI 서버 객체 탐지 결과의 갱신 타이밍이 비동기적이기 때문에 발생했습니다. 이를 해결하기 위해 N초 단위의 요청을 C번 보내는 리트라이 메커니즘을 적용해 해결했습니다. 독립적인 두 서버가 메시지 큐로 연결되어 동작하는 경우 필요한 네트워크 전략에 대해 고민해 볼 수 있었습니다.

서버 개발

기여도: 40%

스택: Nest.js, Prisma, PostgreSQL

- 권한 확인 미들웨어를 [illegible] 댓글 [illegible]도록 했으며, AuthGuard를 통해 권한이 필요한 API 요청에 데코레이터를 사용하면 간단하게 권한 확인을

할 수 있도록 했습니다.
- 게시글 CRUD 엔드포인트를 개발했습니다. 단일 책임 원칙을 준수해 효과적으로 Service 레이어의 책임을 분리시켜 재사용 가능하도록 구현했습니다.

경력

크몽

프리랜서 | 프리랜서
2023.06. ~ 2023.12. (7개월)

고객이 처한 문제를 개발로 해결하며 다양한 웹서비스를 개발 및 배포 했습니다. 총 12건의 서비스를 판매했으며 9개의 5점만점 리뷰를 유지했습니다. 진행했던 주요 프로젝트 몇가지를 소개하겠습니다.

Whatshot (운영 중지)

관리자가 SNS 임베디드 코드를 저장해 다른사람에게 저장한 다양한 SNS(레딧, 인스타그램, 유튜브, 트위터등)의 게시글들을 공유하는 웹사이트입니다. AWS에 배포 되었습니다.

백엔드 | 프론트엔드

스택: React, BootStrap, Sass, React Query, AWS, Nest.js, PostgreSQL, Prisma, Pm2

- Tanstack Query를 useInfiniteQuery와 React Infinite Scroll 라이브러리를 사용해 Infinite Scroll을 구현했습니다. 이를 통해 네트워크 요청 빈도를 줄였으며 UI/UX를 개선했습니다.
- 랜덤 정렬의 인피니트 스크롤을 구현해야 했으나, Prisma에서는 랜덤 정렬을 사용할 수 없었습니다. Prisma의 rawQuery 메서드를 사용해 랜덤 정렬 기능을 구현했습니다.
- Https가 적용되지 않은 웹사이트의 보안 알림이 사용자 경험을 저하시킨다는 고객의 불만사항이 있었습니다. CloundFront, ELB를 사용해 Https를 적용했습니다.

우리동네 커뮤니티 (운영 중지)

Vanilla Javascript로 구현한 동네 커뮤니티 서비스입니다. 주민등록증 사진 인증을 통해 동네 인증 후 사용 가능하며, 본인의 동네에만 글을 게시하거나, 열람할 수 있습니다. Cafe24를 통해 배포 진행했습니다.

깃허브

스택: HTML, CSS, JavaSc | | 댓글 s, MySQL

- Socket.io를 사용해 실시간 채팅기능을 구현했습니다. 게시글마다 실시간 채팅창을 가질 수 있도록 구현했습니다.
- Naver Clova OCR API 사용해 주민등록증 사진을 기반으로 사용자 정보 추출 기능을 구현했습니다. 신분증 인증이 완료된 사용자만 게시글 등록, 수정이 가능하도록 구현했습니다.

팀 스파르타

APM | 플러스 프론트엔드팀 | 재직 중
2024.09. ~ 재직 중 (3개월)

주니어 개발자 대상의 부트캠프 항해 플러스 프론트엔드 3기 매니저로 재직하고 있습니다.
교육생들이 겪는 다양한 문제를 해결하고 교육생들의 이탈을 최소화 하는것을 목표로 합니다.

상황에 따른 유동적인 대처

교육 시작 전 1주일 정도의 기간동안 가장 많은 이탈이 발생합니다. 이탈을 최소화 하기 위해 이탈 위험군에게 개인적인 연락을 통해 정해진 질문 리스트를 전달해야 한다는 업무를 받았습니다. 첫 날 해당 업무를 진행했고 마주한 문제는 다음과 같았습니다.

1. 수강생의 개인 휴대폰으로 통화를 걸어 질문을 전달하니 대부분의 수강생이 부담스러워 했습니다.
2. 통화는 사적인 영역입니다. 사적인 공간에 갑작스럽게 모르는 번호로 전화가와 질문에 대한 응답을 요청하니 불쾌감을 전하는 수강생도 있었습니다.

해당 응답을 듣고 질문 리스트의 변경과 접근 방식에 변화가 필요하다는 생각이 들었습니다. 이 질문을 하는 이유는 수강생의 이탈을 막기 위함이고, 이탈을 막기 위해서는 교육 과정과 운영진에 대한 신뢰도가 가장 중요하다는 생각이 들었습니다. 이에 신뢰도를 올리기 위해서 어떤일을 해야하는지에 대해 고민하고 다음과 같이 접근했습니다.

1. 이탈률 감소를 목표로 두는게 아닌 교육생과의 교감 및 커리큘럼 신뢰도 상승을 위해 연락을 시도해야 한다. 이를 위한 질문 리스트는 유동적으로 변경한다.
2. 수강생의 성장을 위해 매니저가 세밀하게 신경쓰고 있다는 인상을 심어줘야한다. 그를 통해 교육 과정에 대한 신뢰도 상승을 얻어내야 한다.

이렇게 내가 지금 어떤 문제를 해결하고 있는지를 파악한 후 업무에 다시 들어갔으며,
결과적으로 교육 시작까지 0%의 이탈률을 유지할 수 있었습니다.

교육

동의대학교

대학교(학사) | 정보통신공학과
2017.03. ~ 2024.03. | 졸업

기술 스택

Next.js, Flutter, React, tailwind-css, zustand, react-query, NestJS, PostgreSQL, supabase, riverpod

---

포트폴리오

URL

그동안 받은 상장

한이음 자율주행 배달로봇 동작 시연 영상

첨부파일

한이음_자율주행_배달로봇_논문.pdf

대외활동

bookSailor

기타

현재 리딩하고 있는 북 스터디 입니다. "귀에 쏙쏙 들어오는 함수형 프로그래밍", "프로그래머의 뇌"를 주제로 성공적으로 완료 했으며 계속해서 운영하며 개발 관련 도서 50가지를 읽어내는 것을 목표로 하고 있습니다.

깃허브

항해 플러스 프론트엔드 1기

팀 스파르타

주니어 프론트엔드 개발자 대상의 프론트엔드 심화 과정을 수료했습니다. 가장 우수한 수료생에게 주어지는 블랙뱃지를 받으며 수료했습니다.

교육 내용

- V8 엔진의 히든 클래스의 동작을 살펴보고 이를 사용한 최적화 방법에 대해 학습했습니다.
- React의 diff 알고리즘, React Hook의 동장 방식 등을 직접 구현하며 리액트에 대한 깊은 이해도를 얻었습니다.
- 유닛, 통합, E2E 테스트와 TDD 방법론에 대해 학습했습니다. 이를 프로젝트에 실제로 적용하며 테스트 코드를 언제, 어떻게 작성해야 하는지에 대한 고민을 할 수 있었습니다.
- React Fiber에 관한 주제로 발표를 진행했습니다. 다양한 자료를 찾아보며 비동기적 방식인 React Fiber가 동기적 방식인 Stack이라 불리던 렌더링 방식의 어떤 문제점을 해결하려 했는지와 어떻게 동작하는 지에 대해 깊이 이해할 수 있었습니다.

외국어

영어

일상 회화 가능