김지훈

Frontend Engineer

2001년생

wlgns6921@gmail.com

010-5664-5041

Github

개발 블로그

포트폴리오 사이트

배움과 **개선**을 멈추지않는 **책임감** 있는 개발자 김지훈 입니다.

Summary

기획 단계부터 **개발, 배포, 운영**까지 서비스를 완성해본 경험을 갖춘 프론트엔드 개발자입니다.

실제 사용자 환경을 고려하여 **UI를 개선**하고 **문제를 해결하는 과정**에 가치를 두며,

협업과 **지속적인 학습**을 통해 더 나은 제품을 만들고 배우는 것을 목표로 성장하고 있습니다.

Skills

Languages	TypeScript, JavaScript
Frameworks	React
Styling & CSS	Tailwind CSS, CSS(Responsive Design)
Deployment	Nginx, Vercel, Docker, Jenkins, AWS
Tools	Git, Figma, Notion, Slack

Education

2025.02

성결대학교 정보통신공학과 학사 졸업 — 2019.03 ~ 2025.02

학점 3.71 / 4.5

Awards

프로그래밍 설계 경진대회 우수 — 2020.06

웹서버 프로그래밍 교내 경진대회 최우수 — 2023.06

Project 1

2025.03

펫토리(PETORY) - 반려동물 일정 및 공유 돌봄 웹

친구/가족/돌보미와 함께 반려동물의 일정을 공유 및 관리할 수 있는 웹서비스

개발 환경 | 2025.03 - 2025.12 / 2명 (BE 1, FE 1)

포지션 | Frontend

기술 스택 | React, JavaScript, CSS(Responsive Design)

개발 내용

1 React 구조 설계 및 컴포넌트 분리

- 페이지 / 레이아웃 / 공통 컴포넌트를 분리한 구조로 확장성과 유지보수성 확보
- UI 컴포넌트와 서비스 로직 분리로 기능 추가 시 영향 최소화
- 기능 단위 설계로 재사용성 향상 및 중복 코드 감소

2 React Query 기반 성능 최적화

- 쿼리 키, 캐싱, 무효화 기준을 정리해 중복 API 호출 제거
- API 호출 70~80% 감소, 재방문 로딩 35~40ms 수준 단축
- 화면 전환·재방문 흐름에 맞게 서버 상태 갱신 정책을 정립

3 Axios 인터셉터 + 인증 흐름 기반 전역 에러/접근 제어 구조화

- Axios 인터셉터로 네트워크/서버 에러를 단일 지점에서 처리
- 401/403 감지 시 보호 라우팅 및 로그인 리다이렉션 구현
- 중복 예외 처리 제거로 에러 UX 일관성 및 유지보수 효율 향상

4 배포 구조 개선 및 실서비스 운영 관점 최적화

- EC2 단일 인스턴스 기반 배포 한계를 인식하고 프론트엔드 Vercel 이전
- 배포·운영 부담 감소, HTTPS/도메인 관리 단순화로 서비스 안정성 향상

5 UI / UX 설계 및 실서비스 운영 관점 개선

- 모바일 브라우저별 주소창 차이로 발생하는 뷰포트 레이아웃 깨짐 이슈 해결
- 모바일 반응형 레이아웃 및 크로스 브라우징 대응
- Modal 기반 멀티 스텝 폼 설계로 사용성 개선
- 전역 테마관리로 인한 다크모드 등 환경 대응

Github | https://github.com/wlgns5041/Petroy-FrontEnd

배포 URL | https://www.petory.site

OverView | 펫토리 소개

Project 2

2025.12

개인 포트폴리오 웹사이트

나 자신과 프로젝트 경험을 소개하고 개발 역량을 보여주는 웹 기반 포트폴리오

개발 환경 | 2025.12 - 2026.01 / 1명 (개인 프로젝트)

포지션 | Frontend

기술 스택 | React, TypeScript, Tailwind CSS

개발 내용

1 TypeScript 기반 컴포넌트 구조 및 타입 안전성 확보

- 섹션·프로젝트·모달 등 주요 도메인을 타입으로 정의해 데이터 구조를 명확히 관리
- props·상태를 타입 계약으로 고정해 기능 확장 시 사이드 이펙트 최소화
- 개발 단계에서 타입 오류를 사전 차단해 런타임 버그 발생 가능성 감소

2 인터랙션 중심 React 렌더링 성능 최적화

- 스크롤·모달·전환 인터랙션에서 React state 남용을 피하고 렌더 흐름 설계
- 애니메이션·스크롤 연동 로직을 외부 값 중심으로 처리해 렌더 비용 최소화
- 인터랙션이 많은 화면에서도 UI 응답성과 프레임 안정성 유지

3 코드 스플리팅 및 Lazy Loading 기반 초기 로딩 최적화

- 상세 모달 및 PDF가 초기 번들에 포함되어 메인 스레드를 점유하던 구조를 개선
- React.lazy + Suspense를 활용하여 사용자 인터랙션 시점에만 로드
- TBT 30ms → 20ms (약 33% 감소)로 초기 메인 스레드 차단 시간 개선

4 실서비스 운영을 고려한 UI 구조 및 사용자 흐름 안정화

- 모바일·브라우저 환경별 동작 차이를 고려해 레이아웃 기준 정립
- 비동기 처리·로딩·실패 상황에서 대체 UX 흐름을 함께 설계해 사용자 혼란 최소화
- Framer Motion을 활용한 UX 일관성 유지

Github | https://github.com/wlgns5041/portfolio

배포 URL | https://jihoon.cloud