이주영

Frontend Engineer · 프론트엔드 개발자 | React · TypeScript · Next.js

Email juyung0903@naver.com | GitHub github.com/phnml1 | Portfolio phnml1.github.io

Summary

React·TypeScript 기반 내부 운영 웹과 PDA 현장 화면을 개발했고, Next.js 프로젝트에서 REST API·서버 상태 관리·실시간 댓글 동기화를 다뤘습니다. 주문 변환·검수 화면, 스캐너 입력·오프라인 작업 보존처럼 화면 상태와 API 응답이 자주 바뀌는 흐름을 구현했습니다.

Skills

- Frontend: HTML5 · CSS3 · JavaScript · TypeScript · React · Next.js · Tailwind CSS · React Native
- State & Data: TanStack Query · Zustand · IndexedDB · Local Persistence
- Realtime & Integration: REST API · WebSocket(STOMP) · Socket.io · Android WebView · CustomEvent
- Quality & Performance: Git · Jest · Playwright · Lighthouse · React Profiler · GitHub Actions
- Additional Experience: Kotlin · Spring Boot API 연동 · MySQL · PostgreSQL · Object Storage

Experience

MJ Corporation (vandro.ai) | 계약직 | 2026.05 - 2026.08

React · TypeScript 중심 Frontend 개발 / Kotlin · Spring Boot 기반 백엔드 연동

- Excel/VBA 주문 처리의 웹 전환과 PDA/WebView 기반 WMS 구축에서 React·TypeScript 프론트엔드 개발을 담당하고 Kotlin·Spring Boot API를 연동했습니다.
- 늘품마루 전략기획팀과 PDA 현장 작업자의 업무 흐름을 확인해 주문 변환·검수·산출물, 스캐너 입력·오프라인 복구 요구사항을 화면에 반영했습니다.

Projects

NP WMS Picking

PDA 작업자의 로케이션 이동, 상품 확인, 수량 피킹, 부분·부족 처리, 분류 흐름을 지원하는 WMS 연동 물류 피킹 시스템

회사 프로젝트 | 2026.07 - 2026.08 | React · TypeScript · Vite · IndexedDB · Android WebView · Spring Boot API 연동

역할: PDA Frontend 개발 및 Android WebView 연동

- 실제 PDA 작업자의 피킹·스캔 순서를 확인해 640×480급 환경의 UI/UX를 조정하고, 한 화면에 한 단계와 핵심 액션이 보이도록 설계했습니다.
- PDA 바코드 입력이 키 입력·붙여넣기·Android IME·WebView 이벤트로 들어와도 하나의 제출 흐름에서 처리하도록 통합했습니다. 물리 스캐너 입력은 Android에서 받아 WebView로 전달했습니다.
- 네트워크가 불안정한 현장의 작업 유실·중복 전송에 대비해 작업 기록과 전송 대기 명령을 IndexedDB에 보존했습니다. 재연결 시 명령을 순서대로 재전송하고, 중복 전송과 version 충돌을 구분해 처리했습니다.
- 부분·부족 피킹에서 완료 수량과 잔여 작업을 분리했습니다. PDA 작업 상태와 WMS의 stateVersion·assignmentVersion이 일치하지 않는 경우를 감지해 일반 완료 흐름과 분리했습니다.

NP-OIS

Excel/VBA 중심 주문 가공 업무를 웹 기반 업로드·변환·검수·산출물 생성 흐름으로 전환한 내부 운영 시스템

회사 프로젝트 | 2026.05 - 2026.07 | React · TypeScript · TanStack Query · Kotlin · Spring Boot API 연동 · MySQL · Object Storage

역할: React 업무 화면 구현 및 주문 변환·검수·산출물 API 연동

- 늘품마루 전략기획팀과 실무 미팅을 진행해 주문 파일별 양식 차이로 VBA를 반복 실행하던 업무와 실제 주문 데이터의 활용 방식을 확인했습니다. 이를 바탕으로 업로드, 양식 감지·미리보기, 표준 주문 변환, 검수, 산출물 생성 화면을 구현했습니다.
- 주문 변환이 진행 중인 동안에만 queue/run/progress 상태를 polling하고 완료 후 중단하도록 구성했습니다. 완료·실패 전이에 맞춰 관련 query를 무효화하고, 성공한 실행은 별도 새로고침 없이 검수 화면에 반영되도록 했습니다.
- 원본 주문 행과 변환된 표준 주문 항목을 함께 확인하는 검수 화면을 구현하고, 누락된 기준정보 수정, 수동 매핑, 재검증, 검수 재개 흐름을 처리했습니다.

CodeMate

GitHub PR 리뷰, AI 분석 요청, 실시간 댓글·알림, 통계 기능을 구현한 코드 리뷰 협업 플랫폼

개인 프로젝트 | 2026.02 - 현재 | Next.js · TypeScript · Tailwind CSS · TanStack Query · Prisma · PostgreSQL · Socket.io · Jest · Playwright

Live: code-mate-two.vercel.app | GitHub: github.com/phnml1/CodeMate

- Socket 이벤트로 받은 댓글 변경을 TanStack Query setQueryData로 캐시에 반영했습니다. 댓글 작성·수정·삭제 뒤에는 관련 query를 재조회하고, 연결 실패 시에는 polling fallback을 유지했습니다.
- 댓글 페이지의 Server Component가 자기 /api/repositories를 호출하던 경로를 DAL 직접 조회로 바꿔 서버 내부 HTTP 호출을 없앴습니다.
- 통계 카드·차트·최근 PR을 각각 Suspense 경계로 분리했습니다. 이 작업을 포함한 여러 Next.js 변경 전후를 동일한 로컬 조건에서 15회씩 비교했고, 대시보드 LCP p75는 4,948ms → 4,236ms였습니다.
- Webhook·댓글·알림·AI 응답·점수 계산 로직을 Jest로 검증하고, lint → type check → test → build → Playwright 순서의 GitHub Actions CI를 구성했습니다.
- 검색창·댓글 입력창과 아이콘 버튼에 화면 읽기 도구가 알아볼 수 있는 이름을 붙이고, 댓글·PR diff의 접기/펼치기 버튼에는 현재 상태를 표시했습니다. 검색·댓글 화면은 Playwright 테스트로 확인했습니다.

Fly:On

패러글라이딩 비행·관광 일정을 관리하는 React Native 기반 모바일 앱

팀 프로젝트 (FE 2 / BE 2 / Design 1) | 2025.06 - 2025.11 | React Native · TanStack Query · Zustand · SecureStore · Jest · React Profiler

역할: 일정 편집 성능 개선, 상태 구조 및 인증 상태 복원 로직 구현 | 출시 기록: 원스토어 Fly:On | GitHub: SQUAD-FLY-ON/FLY-ON

- EAS 배포와 원스토어 출시 전 과정을 맡았습니다.
- 해당 Drag&Drop interaction에서 상태 변경이 컴포넌트 트리 전체로 전파되는 병목을 React Profiler로 확인했습니다. React.memo와 컴포넌트 분리를 적용해 total commit duration 211.5ms → 72.9ms, 평균 commit duration 17.6ms → 6.6ms로 줄였습니다.
- Zustand persist와 SecureStore로 토큰·회원 정보를 분리하고 앱 초기화 시 저장된 토큰으로 인증 상태를 복원했습니다. 로그인·로그아웃·토큰 갱신·인증 초기화 흐름은 Jest로 검증했습니다.

러너스하이

GPS 기반 러닝 기록과 실시간 경쟁 기능을 구현한 React Native 모바일 앱

팀 프로젝트 (FE 1 / BE 2) | 2025.02 - 2025.07 | React Native · Expo · TypeScript · WebSocket(STOMP) · TanStack Query · Zustand

역할: 프론트엔드 개발 전담 | GitHub: SQUAD-RUNNERS-HIGH/frontend

- 지도 영역에 맞춰 러닝 코스를 조회하고, 코스 마커를 누르면 경로와 전체 코스가 보이도록 지도 시점을 조정했습니다.
- 솔로·코스·경쟁·크루 러닝마다 다른 화면과 위치 처리 로직을 모드별 훅으로 나누고, 공통 러닝 상태는 Zustand에서 관리했습니다.
- STOMP 연결에 heartbeat·재연결을 설정하고, 크루 러닝과 개인 러닝의 구독·위치 전송 경로를 분리했습니다.

Education

한국공학대학교 공학사 | 2019.03 - 2026.02

주전공: IT경영 · 복수전공: 소프트웨어공학과