



김명재 이력서

지원 직무 : Frontend Developer

 1994.07.30, ENFP

 총 경력: 6년

- 기획부터 개발, 배포까지 직접 주도해온 **고객 중심 프론트엔드 개발자**입니다.
- 금융·핀테크 분야에서 **문제의 본질을 정의하고, 가장 쉽게 해결하며 '고객의 불안 해결'**에 집중해왔습니다.




인사

본질을 추구하는 개발자 김명재입니다.

제 휴대폰 바탕화면에는 아래 3가지 원칙이 저장되어 있습니다.

1. **고객 중심**
2. **본질 추구**
3. **가장 쉬운 해결책**

지금 나의 고객이 누구인지 먼저 파악을 하고 고객이 갖고 있는 문제점과 불편함의 본질을 고민합니다. 어쩌면 개발이 아닌 다른 방법이 가장 빠르지 않을까 항상 의심하며 마침내 가장 쉬운 방법으로 해결합니다.

 [포트폴리오](#)

스택

Next.js · TypeScript · Zustand · Tailwind · Capacitor

Django · Docker · AWS

분야	기술/도구
Frontend	React, Next.js, TypeScript, Zustand, Tailwind, React Query
Mobile	Capacitor(Android/iOS), OAuth, In-App Purchase, PWA
Backend/Infra	Django, DRF, Docker, AWS(EC2/RDS/S3), GitLab CI/CD
기획/디자인	Figma, Builder.io, Adobe XD, Photoshop
데이터/마케팅	GA4, GTM, SEO, AEO, 광고 데이터 분석
Database	PostgreSQL, SQLite

경력

(주) 알케미랩 | 서비스개발실 실장

2021.04 ~ 현재 | 서비스 기획 및 프론트엔드 개발 총괄

분야: 가상자산·핀테크 / 웹·앱 하이브리드 서비스 개발

핵심 성과 요약

기간	프로젝트명	주요 키워드
2024~25	zolbo-APP	Next.js + Capacitor · 앱 통합 · 고객 피드백 기반 개선
2023~24	EntityX	Django · 비동기 구조개선 · 상태관리 도입 · FE 최적화
2022~23	Selfrecipe	TradingView 시각화 · 환경분리 · 자동배포
2022	Ragnar	Django · WebSocket 설계 · 실시간 통신 · 문서화
2021	hompape	브랜드 리뉴얼 · 조직 정체성 확립
2021	zolbo-WEB	Django · 도메인 학습 · 금융결산 · Chart.js 시각화

1 업비트 로보어드바이저 “졸보(zolbo)” 하이브리드 앱 개발

2024~2025



키워드: Next.js · Capacitor · PWA · Zustand · React Query · WebSocket · OAuth · In-App Purchase

[프로젝트 개요 및 목적]

- 개인이 API를 등록해 자산 운용을 자동화할 수 있는 하이브리드 앱 개발 프로젝트.
- 웹과 앱(PWA·Android·iOS)을 하나의 코드베이스로 관리하여 유지보수 효율성과 일관된 UX 확보가 목표.
- 사용자 접근성과 신뢰성을 높여 초보 투자자의 진입 장벽을 낮추기 위해 시작된 프로젝트입니다.
- FE 총괄 리드로, **Next.js + Capacitor 기반의 하이브리드 앱(PWA·Android·iOS)** 설계 및 구현
- Binance 기반 MVP에서 **Upbit API 연동형 자동 운용 서비스**로 전환
- 웹·앱 통합 구조 설계, 상태관리·인증·빌드 파이프라인까지 전체 FE 구조 총괄

[팀 구성]

- 프론트엔드 2명, 백엔드 2명, 디자이너 1명, 기획자 1명
- 김명재: FE Lead로 전반적인 구조 설계 및 핵심 기능 구현 담당

[담당업무]

- **아키텍처 설계:** Next.js(App Router) 기반 SSR/CSR 통합 구조 설계
 - `apps/web` 분리 구조로 PWA·AOS·iOS 빌드 통합 지원
- **API 정규화:** Upbit·Zolbo·Point 도메인별 응답 스키마 통합
- **상태 관리:** Zustand 기반 로그인·토큰 관리 및 persist 스토리지 설계
- **데이터 관리:** React Query 기반 데이터 캐싱 및 병렬 요청 처리
 - WebSocket 기반 실시간 자산·주문 데이터 스트림 구현
- **빌드·배포:** AWS Amplify + GitLab CI/CD 자동 빌드·배포 파이프라인 구축
- **UX 설계:** 로그인→인증→레퍼럴→계정등록→운용 설정 BottomSheet 플로우 구성
- **인증 구조:** Google OAuth + 휴대폰 인증 3단계 플로우 BottomSheet로 통합
- **인앱결제:** Google Play 결제 연동 후 ZLB 포인트 자동 지급 로직 구현

[문제 해결 과정]

- 기존에는 flutter 를 사용한 이중 개발로 App 개발이 진전이 보이지 않았음 → Next.js + Capcitor 을 활용한 단일 코드 기반으로 해결
- 초기에는 로그인/인증 과정이 복잡해 이탈률과 진입장벽이 높았음 → BottomSheet 기반 단계별 인증 UI로 개선
- 실시간 자산 데이터가 페이지 새로고침 시 끊김 → WebSocket 구독 구조를 통합해 실시간 갱신 구현

- 배포 시간이 길고 환경이 불안정 → Amplify + GitLab CI/CD로 환경 자동화

[성과]

- 하이브리드 단일 코드베이스 확립으로 **App 런칭 성공** 및 코드 유지보수 효율 **50% 개선**
- 인증/레퍼럴/포인트 플로우 자동화로 **가입 전환율 35% 향상**
- 실시간 운용 ON/OFF 시스템 도입으로 **월간 유지율 25→40% 향상**
- WebSocket 기반 실시간 자산 갱신으로 **새로고침 빈도 70% 감소**
- Amplify + CI/CD 자동배포로 **배포시간 80% 단축**
- 코드 정규화 및 Cursor.ai 도입으로 **신규 기능 도입 속도 2배 향**
- 실시간 로그 분석 기반 **투자전환 데이터 정확도 향상**

기술은 문제를 해결하지만, 고객은 '불안'을 해결받고 싶어 한다.

진정한 서비스 완성은 **기술 x 고객 이해의 교차점**에 있다.

2 가상자산 거래소 "EntityX" 개발 리드

2023.10~2024.06



키워드: WebSocket · TradingView · 비동기 통신 · 실시간 거래

[요약]

- OKX 기반 거래소에서 Bybit로 전환하며 **실시간 대시보드 및 주문 기능**을 개발
- Django + Vanilla JS + WebSocket + TradingView 기반 실시간 트레이딩 UI 구축

[담당업무]

- WebSocket 기반 실시간 데이터 스트림 및 차트 반영 구조 설계
- 주문 생성·취소·체결·포지션 업데이트 UI 및 비동기 API 연동
- Django 백엔드와 실시간 데이터 동기화 파이프라인 구성
- TradingView 차트 위젯 커스터마이징 및 포지션별 손익 표시
- OKX → Bybit 전환 시 API 구조 차이에 따른 FE 마이그레이션 수행
- 데이터 업데이트 주기 분리로 렌더링 부하 최소화

[성과]

- OKX/Bybit API 연동 구조 확립, 실시간 거래 안정화
- 마이그레이션 소요시간 **3주 → 1주로 단축(⅓)**
- 실시간 차트 렌더링 지연률 **40% 감소**
- 주문 응답 시간 **600ms → 280ms로 단축**, 거래 안정성 확보
- 포지션/성과 탭 도입으로 **이탈률 50% 감소**

효율적인 코드가 곧 사용자 경험의 속도다.

3 펀드마켓 솔루션 "화라쥘" 개발 리드

2023.02~2023.07



키워드: Django · Tailwind · 화이트라벨 · 멀티테넌트 구조

[요약]

- 퍼스널 펀드마켓 **화이트라벨 솔루션 '화라폴'** 총괄 개발
- 누구나 포트폴리오 기반 펀드 운용 및 투자자 모집이 가능한 구조 설계

[담당업무]

- Django 기반 멀티테넌트 구조 설계 (운용사별 계정/보수/계약 관리)
- 기본보수·성과보수·회원관리 자동화
- 브랜드 테마 커스터마이징(로고·컬러·모드) 기능 구현
- 퍼블렛·지브릭 등 파트너사 API 연동 및 표준화
- 법적 정산 및 보수 프로세스 자동화 설계
- 운용사 온보딩 대시보드 및 관리자 시스템 구축

[성과]

- MC9·Inus·Lia Holdings 등 **파트너사 화이트라벨 계약 체결 및 상용화**
- 점조직형 펀드 모델 정산 자동화로 **운영 리소스 30% 절감**
- 브랜드 커스터마이징 기능으로 **온보딩 속도 2배 향상**
- 화이트라벨의 본질은 기술보다 **프로세스와 신뢰 설계**에 있다는 인사이트 확보

4 알케미랩 홈페이지 리뉴얼 프로젝트 리드

2022.11



키워드: Django · Tailwind · 브랜드 아이덴티티 · 조직문화

[요약]

- 브랜드 리뉴얼 총괄: 미션·비전·전략·가치 재정의 및 웹 반영
- AWS EC2 + Docker 기반 배포 환경 구성

[담당업무]


- 25개 기업 벤치마킹을 통한 알케미랩 핵심가치 구조화
- 내부 워크숍·인터뷰 통해 '핀테크 자본주의를 만드는 사람' 키워드 도출
- 브랜드 언어·UI 가이드·스토리 섹션 통합 설계
- 디자인팀·개발팀 간 협업으로 메시지 일관성 확보

[성과]

- 구성원 공감도 90% 이상, 페이지 조회수 2.3배 상승
- 외부 투자 피칭용 홈페이지 활용
- "누구나 편안하게 투자할 수 있는 금융생태계"라는 조직 정체성 명문화

5 자문형 크립토 서비스 "SelfRecipe" 개발

| 2022.05~2022.10

 **키워드:** TradingView · 실시간 시각화 · WebSocket · Docker

[요약]

- Binance API 기반 실시간 시세·차트 연동
- TradingView 도입으로 신뢰성 높은 실시간 데이터 시각화

[담당업무]

- TradingView 차트 커스터마이징 및 위젯 구조 설계
- 실시간 자문 포트폴리오 데이터 시각화
- 차트 반응형 업데이트 및 사용자 테마 구성 기능 구현
- 환경 분리 및 GitLab CI/CD 자동배포 파이프라인 구축

[성과]

- 실시간 시각화로 사용자 리텐션 **30% 증가**
- 체류시간 **2배(4→9분)**, 차트 렌더링 지연 **40% 감소**
- “데이터는 보여지는 순간 신뢰가 된다”는 FE 문화 정립

6 커스텀 트레이드 웹 “Ragnar” 개발 총괄

| 2022.02~2022.04

 **키워드:** Django · WebSocket · Docker · 실시간 거래

[요약]

- Deribit API 기반 실시간 포지션·주문 관리 시스템 구축
- EC2 + Docker 기반 서버 배포 자동화

[담당업무]

- WebSocket 실시간 체결/포지션 업데이트 구현
- 주문/거래내역 데이터 시각화 및 반응형 UI 구성
- 환경 변수 관리 및 보안체계 구축

[성과]

- 거래 데이터 처리속도 **45% 향상**, 주문응답시간 **700ms→300ms 단축**
- 사용자 체류시간 **1.8배 증가**, 운영비 **20% 절감**
- 안정적인 실시간 거래 구조로 이후 프로젝트(WebSocket 표준화) 기반 마련

7 Binance 일임형 서비스 “zolbo.ai” WEB 기획 및 개발

2021~2022

 **키워드:** Django · Chart.js · 금융도메인 학습 · 포인트 결산

[요약]

- 금융 지식 없이 1일 결산 로직을 직접 설계 후 Django + JS로 구현

- Binance API 연동, 자산 결산, 포인트 차감 로직 개발

[담당업무]

- Chart.js로 운용성과 시각화 및 백엔드 데이터 연동
- 일일 운용성과 기준의 포인트 결산 프로세스 구축

[성과]

- 데이터 시각화의 가치 체득, 금융 로직 직접 설계 경험
- '도메인 이해'의 중요성을 깨달으며 서비스 사고력 확장



(주) 강인함 | 신사업팀장

2020.02 ~ 2021.04 | 신사업 기획 · 개발 · 마케팅 총괄

분야: 온라인 교육 스타트업(교육컨설팅) / 웹사이트 및 CRM 자동화 구축



주요 성과

분야	주요 내용
웹사이트 개발	WordPress 직접 개발 (HTML/CSS/JS 수정, SEO 구조화)
CRM 프로세스	Google Spreadsheet 기반 상담-결제-등록 자동화
데이터 분석	GA4 + GTM 연동, 전환 이벤트 추적, KPI 대시보드 구축
디지털 마케팅	네이버·카카오·구글 광고 운영, ROAS 기반 캠페인 최적화
브랜딩 디자인	Photoshop·XD 기반 랜딩/배너·포스터 디자인
성과	월 트래픽 3.2배, 광고전환율 +104%, 분기 매출 2.5배 성장

기술 스택: WordPress · Google Spreadsheet · GA4 · GTM · HTML · CSS

[프로젝트 개요 및 목적]

- 교육컨설팅 브랜드 “강인함”의 디지털 전환 프로젝트로, 기존 오프라인 중심의 상담·결제·등록 프로세스를 온라인 자동화 시스템으로 전환하는 것을 목표로 함.

[주요 업무]

- WordPress 기반 홈페이지 구축 및 SEO 구조화
- Google Spreadsheet 기반 CRM 자동화 (상담 → 결제 → 등록 데이터 파이프라인 설계)
- GA4 + GTM 기반 상담 전환 추적 및 리포트 자동화
- 광고 캠페인(네이버/카카오/구글) 성과 분석 및 UX 개선
- 랜딩페이지, 배너, 포스터 등 온라인 브랜드 에셋 제작

[문제 해결 과정]

- CTA와 광고 타겟팅이 최적화되어 있지 않아 이탈률이 업종평균대비 2배 높음 → 홈페이지 랜딩 구성 최적화 및 키워드 셋팅 최적화(1차, 2차고객) 진행
- CRM 데이터가 산발적으로 관리되어 상담 이력 추적이 어려움 → Google Spreadsheet로 통합 관리
- 광고 효율 저하 → 전환율 기반 GA4 트래킹 도입 및 캠페인 리디자인

[성과]

- 초기 이탈률 70% 감소
- 광고 전환율 2.1% → 4.3%(+104%)
- 월 트래픽 3.2배 증가, 고객 응답속도 60% 단축

- CRM 자동화로 운영 리소스 50% 절감

소개

본질을 추구하는 5년차 프로젝트 개발자

[소개]

4년간 웹 서비스 개발에 집중해온 저는, 작년부터 기술적 한계를 넘어 “앱 생태계 확장”이라는 새로운 가능성을 직접 열어본 경험이 있습니다.

2024년, Next.js 기반 웹 프로젝트를 Capacitor로 래핑하여 Android/iOS 빌드 및 실기 테스트를 독자적으로 성공시켰습니다. Gradle, Manifest, Xcode 등 생소한 영역에서 하루에도 수십 번의 오류를 해결하며, 웹 자산이 PWA·Android·iOS를 모두 아우르는 하이브리드 구조로 확장되도록 구축했습니다.

그 결과, 사내 최초로 앱 빌드를 완수해 회사가 인앱 결제와 스토어 배포라는 새로운 비즈니스를 추진할 수 있는 기반을 마련했습니다.

웹 개발자였던 제게는 “처음 휴대폰에서 직접 실행된 앱”이 단순한 결과물이 아니라, 회사의 기술 신뢰를 높이고, 제 자신에게 한계를 넘을 수 있다는 확신을 준 계기였습니다.

[실패는 배움의 시작이었다]

초기 zolbo.ai(Binance 일임투자 서비스) 개발 당시, 펀드 결산 로직의 오류로 인해 결산 결과가 일치하지 않는 문제가 반복되었습니다. 결산일 기준가와 수익률 계산이 누락·역산 오류를 일으키며 고객의 수익률이 실제 펀드 성과와 달라졌습니다. 당시 저는 단순히 코드를 수정하는 대신, 문제의 본질이 ‘도메인을 모르고 개발했다는 것’에 있음을 깨달았습니다.

그래서 “무엇을 모르는지부터 정의하자”는 원칙 아래, 3주간 하루 다섯 가지 이상의 질문을 스스로 설정했습니다. 펀드 회계, 기준가, 좌수, 선입선출, 대차대조표 등 20여 개의 금융 개념을 논문과 도서를 통해 학습했고, 이를 토대로 입출금 시 마진 밸런스를 역산하는 스냅샷 혼합형 결산 로직을 직접 고안했습니다. 결과적으로 결산 오류율을 100%에서 0%로 낮추고, 성과 불일치 문제를 완전히 해결했습니다.

핵심 성과 요약

- 결산 오류율 100% → 0%
- 3주간 100회 이상 질문 & 2회 내부 공유 세션 진행
- 질문 중심 학습 문화 정착 → 온보딩 레퍼런스 문서화

이 경험은 단순한 기술 습득이 아니라, 학습 구조를 스스로 설계한 과정이었습니다. 매일 대표와 사수에게 질문하고, 팀원과 학습 내용을 공유하면서 ‘질문은 성장의 시작’이라는 문화가 자리 잡았습니다. GPT나 내부 자료가 전혀 없던 시기였기에, 논문·기사·YouTube 등 다양한 채널을 통해 정보를 수집하고, 정리된 내용을 후임 개발자가 쉽게 이해할 수 있도록 내부 레퍼런스 문서로 남겼습니다. 이 프로젝트를 통해 저는 “모르는 것을 빠르게 정의하고, 학습 가능한 체계로 전환하는 능력”이 개발자의 가장 중요한 역량임을 깨달았습니다. 문제를 해결한다는 것은 단순히 오류를 잡는 일이 아니라, 이해의 경계를 넓히는 일이었습니다.

“성장은 문제를 해결할 때가 아니라, 모르는 것을 인정할 때부터 시작된다.”

[팀의 집중력을 높이는 환경 설계자]

개발은 개인의 역량만으로 완성되지 않습니다. 저는 팀이 더 집중할 수 있는 환경을 설계하는 일에 가장 큰 보람을 느낍니다. 업무 효율을 높이는 것은 속도를 높이는 일이 아니라, 불필요한 소음을 제거해 몰입을 돕는 일이라고 믿습니다.

먼저 **회의 구조를 재설계**했습니다. 매일 진행하던 구두 데일리리 문서를 리포트로 전환해 불필요한 회의 시간을 60% 줄였고, 주간 회의는 전사 공지와 담당자별 스크럼 체계로 개편하여 실행률을 35% 향상시켰습니다. 회의가 시간 소비가 아니라 집중 시간 확보의 수단으로 바뀌었습니다.

다음으로 **협업 툴**의 한계를 해결했습니다. Slack의 휘발성 메시지로 인해 결정 이력이 사라지는 문제가 있었기에, Swit 도입을 제안하고 팀을 설득하여 Slack → Swit 마이그레이션 전체를 주도했습니다. 그 결과 커뮤니케이션 효율이 40% 향상되었고, 업무 이력 추적률이 **0%에서 95%**로 개선되었습니다. 채팅 중심의 협업 문화를 기록 중심의 협업 구조로 바꾸어, 누구나 의사결정의 맥락을 쉽게 공유할 수 있게 했습니다.

마지막으로 배포 프로세스를 **자동화**했습니다. AWS Amplify와 GitLab CI/CD를 조합하여 무중단 자동배포 파이프라인을 구축하고, 브랜치별 자동 빌드·QA 검증 체계를 만들어 배포 시간을 80% 단축, QA 처리 속도를 두 배 이상 높였습니다. 이후 팀원들은 반복적인 수동 작업에서 벗어나 문제 해결과 개선에 더 집중할 수 있는 환경을 갖추게 되었습니다.

“팀의 생산성 향상은 개인의 문제해결이 아니라, 집중할 수 있는 구조를 설계하는 일이다.”

[하나부터 열까지 스스로 성장한 여정]

처음부터 끝까지 스스로 부딪히며 성장한 경험이 있습니다. TradingView 기반의 실시간 차트 시스템을 구현할 당시, 문서가 불충분하고 내부 버전은 이미 지원이 종료된 상태였습니다. 저는 “누구도 모른다면, 직접 해석한다”는 마음으로 라이브러리의 핵심 소스코드를 한 줄씩 분석하고, 각 함수의 호출 구조와 반환값을 `console.log`로 추적하며 동작 원리를 복원했습니다.

그 결과, WebSocket 렌더링 지연 시간을 **3.2초에서 0.7초로 단축**하고, 실시간 데이터 반영 속도를 개선해 사용자 체류 시간을 2배 이상 증가시켰습니다. 문서보다 코드를 먼저 믿고, 코드보다 사용자의 반응을 먼저 본 경험이었습니다.

이후 OKX → Bybit 마이그레이션 프로젝트**에서는 거래소 간 API 구조가 완전히 달라 단순 포팅이 불가능한 상황이었습니다. OKX의 세분화된 계약 단위(Contract·Coin·USDT)가 Bybit에서는 통합되어, 수량 계산과 주문 체결 로직 전체를 새로 설계해야 했습니다. 저는 공식·비공식 문서를 모두 수집해 API 응답을 1:1 매핑하고, 코인/USDT 기준으로 환산하는 표준화 파서(Parser)를 직접 구현했습니다.

또한 Bybit이 포지션 정보를 WebSocket으로 제공하지 않는 한계를 극복하기 위해, REST 기반 자산 데이터와 Ticker WebSocket을 결합한 혼합형 스트림 구조를 새로 설계했습니다. 이를 통해 데이터 부하를 30% 이상 줄이고, 거래 오류율을 0.3% 이하로 안정화시켰으며, 예상 기간의 절반인 3주 만에 전체 마이그레이션을 완수했습니다.

이 두 프로젝트는 공통적으로 “문서를 읽는 개발자”에서

“시스템을 해석하는 개발자”

로 성장한 순간이었습니다. 누가 알려주지 않아도, 필요한 답을 스스로 찾고 구조를 새로 세우는 과정 속에서 저는 진짜 성장의 의미를 배웠습니다

[문제 해결은 시스템의 한계를 이해하는 것]

거래소 서비스 초기, Django 기반 PWA에서 네트워크가 불안정할 때 차트·주문 모듈이 반복적으로 오류를 일으키는 문제가 있었습니다. 단순한 성능 이슈로 보이지만, 실제로는 Django의 SSR 특성과 비동기 요청 간 캐시 불일치로 인해 매 요청마다 비캐시 리소스를 새로 로드하는 구조적 한계가 원인이었습니다.

저는 이를 “속도 문제”가 아니라 “사용자 신뢰 하락 거래 중단 매출 저하”라는 비즈니스 병목으로 정의하고, Service Worker Cache + Browser Cache + Django Cache를 병행 적용했습니다. 정적 리소스 재렌더링을 최소화하고, 네트워크가 끊겨도 JS·CSS·차트 데이터가 로컬 캐시에서 즉시 복구되도록 구조를 설계했습니다.

그 결과

- 페이지 로딩 속도 6.8초 → 2.1초 (69% 개선)
- 차트 오류율 80% 감소
- 사용자 거래 유지율 15% 상승(이탈률 유의미적 감소)

이 경험을 통해 초기 데이터가 많아질 경우에 사용자 경험 측면에서의 Django SSR의 한계를 체감하고, React 기반 CSR 구조 전환의 필요성을 직접 제안했습니다. 이는 이후 Next.js 기반 하이브리드 앱으로 발전하는 기술적 전환점이 되었고, “문제 해결은 시스템의 한계를 이해하는 것에서 출발한다”는 제 첫번째 개발 철학으로 자리 잡았습니다.

[사용자의 불안을 해결해야한다]

2025년 9월, 코엑스에서 열린 동아 재테크쇼에서 약 500명의 투자자(잠재 고객)를 대상으로 직접 리서치를 진행했습니다. 그 결과, 대부분의 사용자가 기술적 문제나 보다 **‘전문성 부족으로 인한 심리적 불안’**으로 인해 가상 자산 투자를 망설이고 있다는 사실을 확인했습니다.

이 인사이트를 기반으로 비즈니스 모델을 ‘AI 자동투자’ → ‘안심패스(심리 안정 구독형+학습형)’으로 전환했습니다. 기존의 “기술 편의 중심 UX”를 버리고, “사용자의 불안을 해소하며 신뢰를 쌓는 경험 구조”로 리디자인했

습니다. 단순히 성능이 좋은 서비스를 만드는 것이 아니라, “기술이 사람의 감정을 설계할 수 있는가”라는 질문에 처음으로 진지하게 고민해볼 수 있었습니다.

현장에서 얻은 데이터는 서비스의 철학을 바꾸는 근거가 되었습니다. 설문에 응한 사람 중 70%가. “비트코인 지 금이야 사!”라고 답하면서도 실제 행동으로 옮기지 못했고, 가장 큰 불안 요인은 가격 출렁임(35%), 전문성 부족 (25.4%), 거래소 불신(18.3%)임을 확인할 수 있었습니다. 이를 통해 zolbo가 해결해야 할 핵심은 ‘수익률’이 아니라 ‘심리적 신뢰’라는 점을 명확히 인식했습니다.

이후 TF팀을 구성해 개발·기획·디자인이 함께 “사용자가 배우며 안심할 수 있는 서비스 구조”를 만들기 위해 협업했습니다. 내부 익명 설문 결과, 구성원 90% 이상이 새로운 방향성에 공감했고, 서비스는 기술 중심에서 사용자 중심으로 재탄생할 수 있었습니다.

비록 회사의 경영 여건으로 서비스는 10월에 종료되지만, 이 프로젝트는 팀 전체가 “우리가 왜 이 서비스를 만드는가”라는 질문에 다시 집중하는 계기가 되었습니다. 성과는 사라질 수 있지만, 그 과정에서 얻은 사용자 중심 철학과 조직의 공감대는 남았습니다.

“기술은 기능을 위한 도구가 아니라, 사람의 불안을 해결하는 언어다.”

[정체성]

저는 단순히 코드를 작성하는 개발자가 아니라, “문제를 정의하고, 구조적인 해결책을 찾는 프로덕트 개발자 (Product Developer)”입니다. 기술적 한계를 넘고, 팀의 집중력을 높이며, 사용자의 불안을 해결하며 성장해왔습니다. 앞으로도 서비스의 본질을 탐구하며, 사용자의 신뢰와 경험을 설계하며 회사의 매출에 기여하는 사람이 되고 싶습니다.

학력

충남대학교 자유전공 학부(2013~2017)

학점: 3.36/4.5

울산 호계 고등학교(2010~2013)

자격증

투자자산운용사 2024년 6월 | 식별번호 24-010053

SQLD 2023년 7월 | 식별번호 SQLD-049006432

병역

육군 중위 만기 전역 2019년 7월 | 수도기계화 보병사단(맹호부대) 소대장