


김명재 이력서

■ 기본사항

	성명	(한글) 김명재 (남) (영문) KIM MYEONG JAE
	생년월일	1994.07.30
	연락처	010-7212-0959
	이메일	mkmj730@gmail.com
	주소	서울특별시 양천구 신정동 876-3

■ 학력 및 경력사항 (개발 총 경력 : 6년)

학 력 사 항	학 교 명	성적/등수(기준)	전 공	입학년도	졸업년도	졸업여부
	호계고등학교	전교 2등/250명	인문계(문과)	2010	2013	졸업
	충남대학교	학점 3.26/4.5	자유전공학부(행정학사)	2013	2017	졸업
경 력 사 항	근무처	직위/직책	담당업무	입사년월	퇴사년월	퇴직사유
	(주)알케미랩	선임연구원/팀장	프론트엔드 개발리더	2021.04	재직중	-
	(주)강인함	이사/실장	신규사업총괄	2020.02	2021.04	커리어전환
	수도기계화보병사단	중위/소대장	병력관리	2017.03	2019.07	전역

■ 경력세부사항

근무처	기간	세부 내용
(주)알케미랩	2024.09~2025.09	가상자산 일임투자 서비스 하이브리드 앱개발 • Next.js + Capacitor 기반 WEB/APP(PWA·Android·iOS) 통합 빌드 구조 설계 및 구축 • Upbit API를 연동하여 실시간 자산 조회, 운용 ON/OFF 제어, 자동매매 기능 구현 • GA4, 인앱결제(Google Play Billing), WebSocket 기반 실시간 데이터 반영 구조 개발 • 금융 UI/UX에 최적화된 온보딩 프로세스(BottomSheet·Tab) 구조 설계로 사용자 직관성 향상
	2023.10~2024.06	가상자산 거래소 WTS(트레이딩 대시보드) 개발 • 차트/호가창/체결내역 실시간 연동 구현 (WebSocket 기반) • 주문·포지션·손익·위험관리 모듈 UI 통합 • TanStack Query + Zustand로 Server/Client 상태 이원화 관리 구조 설계
	2023.02~2023.07	가상자산 펀드마켓 솔루션 대시보드 개발 • 펀드 운용지표, 성과차트, 가입현황 시각화 대시보드 설계 • 관리자/운용사/투자자용 Role 기반 접근 시스템 구현 • DRF 백엔드와의 연동을 위한 API Wrapper Service 작성
	2022.11~2022.11	자사 홈페이지 리뉴얼 및 브랜드 아이덴티티 기획 • Django + Javascript + TailwindCss 기반 웹사이트 개발 • 회사미션·비전·전략·핵심가치 체계를 새롭게 정의하고, 이를 웹사이트에 반영하는 리뉴얼 프로젝트 주도

	2022.05~2022.10	가상자산 자문형 서비스 화면 개발 <ul style="list-style-type: none"> • 딥러닝 데이터 기반 물타기주문(6호가) 기능 개발 • 반응형 UI/UX 구축 (모바일·웹 통합 지원)
	2022.02~2022.04	가상자산 옵션 트레이딩 MTS 개발 <ul style="list-style-type: none"> • 차트/옵션 호가 데이터 실시간 시각화 • 사용자 인터랙션(시장가/지정가/수정) 최적화 • 주문 체결/취소 프로세스의 비동기 이벤트 핸들링 개선
	2021.12~2021.01	자사 홈페이지 개발 <ul style="list-style-type: none"> • Django + Javascript + CSS 기반 웹사이트 개발
(주)강인함	2020.10~2021.04	신사업 홈페이지 개발 및 CRM 프로세스 구축 <ul style="list-style-type: none"> • Wordpress 기반 신사업 웹사이트 개발 • GoogleSpreadSheet 기반 리드 관리 CRM 시스템 개발 • 상담, 결제현황, 서류관리 입시 컨설팅 프로세스 자동화

■ 자격 및 특기사항

날짜	자격증	점수/급수	발급기관
2023.07	SQL개발자(SQLD자격)	최종합격	한국데이터베이스진흥센터
2024.06	투자자산운용사	최종합격	금융투자협회
2025.08	정보처리기사	필기합격	한국산업인력공단

■ 처우 및 근로조건

희망직무	프론트엔드 개발	희망급여	추후 협의
희망근무지	서울·경기·인천	종전급여	4600

■ 자기소개서

업 무 방 식	[일의 원칙] 가상자산 및 핀테크 분야에서 기획부터 개발, 배포까지 서비스 전 과정을 직접 경험해온 프론트엔드 개발자입니다. 저는 아래와 같은 3단계의 방법으로 일을 추진합니다.
	1. 킥오프 모든 프로젝트는 “이 일을 왜 하는가, 누구를 위해 하는가”라는 질문에서 시작합니다. 개발을 하기 전에 목적을 먼저 정의하고, 그 목적이 팀과 고객의 방향성에 맞는지부터 확인합니다. 이후 변하지 않는 것과 변하는 것, 할 수 있는 것과 할 수 없는 것에 대해 정의합니다.
	2. 추진 방식 프로젝트 내부에 흔들리지 않는 유스케이스와 가설을 세우고, 관련 서비스와 주변 레퍼런스를 빠르게 조사합니다. 핵심 UI/UX를 프로토타입으로 만들어 실제로 ‘손에 잡히는 형태’로 검증하고, 이 과정을 바탕으로 구체적인 기획과 개발 전략을 수립합니다. 이후 디자이너, 기획자, 개발팀과 함께 방향을 다듬으며 완성도를 높여갑니다.
	3. 개발 Next.js, TypeScript, Zustand, Tailwind, Capacitor 기반의 하이브리드 앱(PWA/Android/iOS) 개발 경험을 바탕으로, 사용자 중심의 인터랙션과 안정적인 배포 환경을 구현해왔습니다. 서비스 기획과 기술 구현을 모두 경험했기에, 기술적 의사결정과 사용자 경험 개선을 균형 있게 이끌 수 있습니다.

실 패 극 복	<p>[실패는 배움의 시작이었다]</p> <p>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.</p>
	<p>핵심 성과 요약</p> <ul style="list-style-type: none">- 결산 오류율 100% → 0%- 3주간 100회 이상 질문 & 2회 내부 공유 세션 진행- 질문 중심 학습 문화' 정착 → 온보딩 레퍼런스 문서화 <p>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.</p> <p>"성장은 문제를 해결할 때가 아니라, 모르는 것을 인정할 때부터 시작된다."</p>
팀 생 산 성 개 선	<p>[팀의 집중력을 높이는 환경 설계자]</p> <p>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.</p> <p>먼저 회의 구조를 재설계했습니다. 매일 진행하던 구두 데일리리 문서형 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.</p> <p>다음으로 협업 툴의 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.</p> <p>마지막으로 배포 프로세스를 자동화했습니다. 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.</p> <p>"팀의 생산성 향상은 개인의 문제해결이 아니라, 집중할 수 있는 구조를 설계하는 일이다."</p>

<div>코 드 분 석</div> <div>및</div> <div>적 용 경 험</div>	<p>[하나부터 열까지 스스로 성장한 여정]</p> <p>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.</p> <p>그 결과, WebSocket 렌더링 지연 시간을 3.2초에서 0.7초로 단축하고, 실시간 데이터 반영 속도를 개선해 사용자 체류 시간을 2배 이상 증가시켰습니다. 문서보다 코드를 먼저 믿고, 코드보다 사용자의 반응을 먼저 본 경험이었습니다.</p> <p>이후 OKX → Bybit 마이그레이션 프로젝트에서는 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.</p> <p>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.</p> <p>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</p>
<div>주 도 적</div> <div>문 제 해 결</div>	<p>[문제 해결은 시스템의 한계를 이해하는 것]</p> <p>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.</p> <p>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.</p> <p>그 결과</p> <ul style="list-style-type: none"> - 페이지 로딩 속도 6.8초 -> 2.1초 (69% 개선) - 차트 오류율 80% 감소 - 사용자 거래 유지율 15% 상승(이탈률 유의미적 감소) <p>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.</p>

<div>기 역 에</div> <div>남 는</div> <div>개 발 경 험</div>	<p>[사용자의 불안을 해결해야한다]</p> <p>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 가상자산 투자를 망설이고 있다는 사실을 확인했습니다.</p> <p>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 리디자인했습니다. 단순히 성능이 좋은 서비스를 만드는 것이 아니라, “기술이 사람의 감정을 설계할 수 있는가”라는 질문에 처음으로 진지하게 고민해볼 수 있었습니다.</p> <p>현장에서 얻은 데이터는 서비스의 철학을 바꾸는 근거가 되었습니다. 설문에 응한 사람 중 70%가. “비트코인 지금이야 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.</p> <p>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.</p> <p>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.</p> <p>“개발은 기능을 위한 도구가 아니라, 사람의 불안을 해결하는 언어다.”</p>
<div>끝 으 로</div>	<p>[정체성]</p> <p>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.</p>