

</> software

developer

portfolio

"왜"라는 질문을
가장 많이하는
송현광을
소개합니다.



궁금증이 많아서 문제의 원인을 파고들고,
그 과정에서 더 나은 해결책을 찾는 걸 즐깁니다.
"왜?"라는 질문이 결국 더 나은 시스템을 만든다고
믿습니다.

송현광 / Song hyunkwang
1997.04.01

experience

2025 1월 ~ 2025 12월 SSAFY

- Samsung Software AI Academy For Youth
- Python 트랙
- 알고리즘, SQL, Django, Vue, JavaScript
- 팀 프로젝트

2023 8월 ~ 2024 2월 Kepco Digital Bootcamp

- Java, Python, SQL
- ML/DL, 데이터 분석 및 시각화
- 팀 프로젝트

career

2024 4월 ~ 2024 9월

(주) 휴먼인텍 SW 연구실 재직

- 웹 애플리케이션 개발 및 유지보수
- RESTful API 설계 및 구현
- Git을 활용한 프로젝트 관리

Skills

- Java & Springboot
- Python & Django
- SQL
- JavaScript & TypeScript
- React & ReactNative
- Flutter
- Docker
- Kubernetes

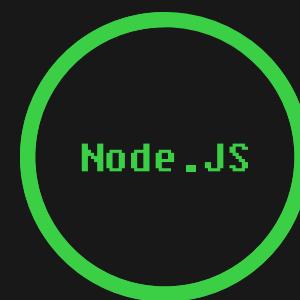
license

- 정보처리기사
- SQLD

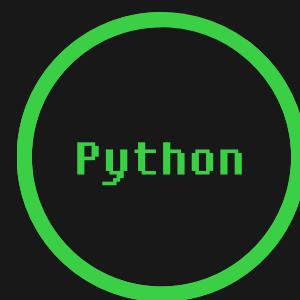
스킬 및 역량



Springboot



Nest.JS



Django



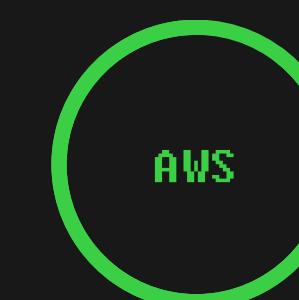
PostgreSQL



Docker



Jenkins



EC2, S3



Kubernetes



언어 사용 능력

Java(Springboot)와 JavaScript(Nest.JS)를 주력으로 다루며 개발의 즐거움을 알아가고 있습니다. 이해하고 넘어가기를 목표로 단순히 기능을 구현하는 것을 넘어 프레임워크와 SQL의 원리를 꾸준히 학습합니다.

개발자로서의 역량

문제가 발생했을 때 근본 원인을 분석하여 해결하는 데 집중하며 유지보수성과 확장성을 고려하여 SOLID 원칙 및 클린 코드를 적용하려 노력하고 있습니다. 또한 Docker 환경 이해 및 CI/CD 과정을 학습하며 시스템 전반에 대한 이해도를 키우고 있습니다.

Project Experience

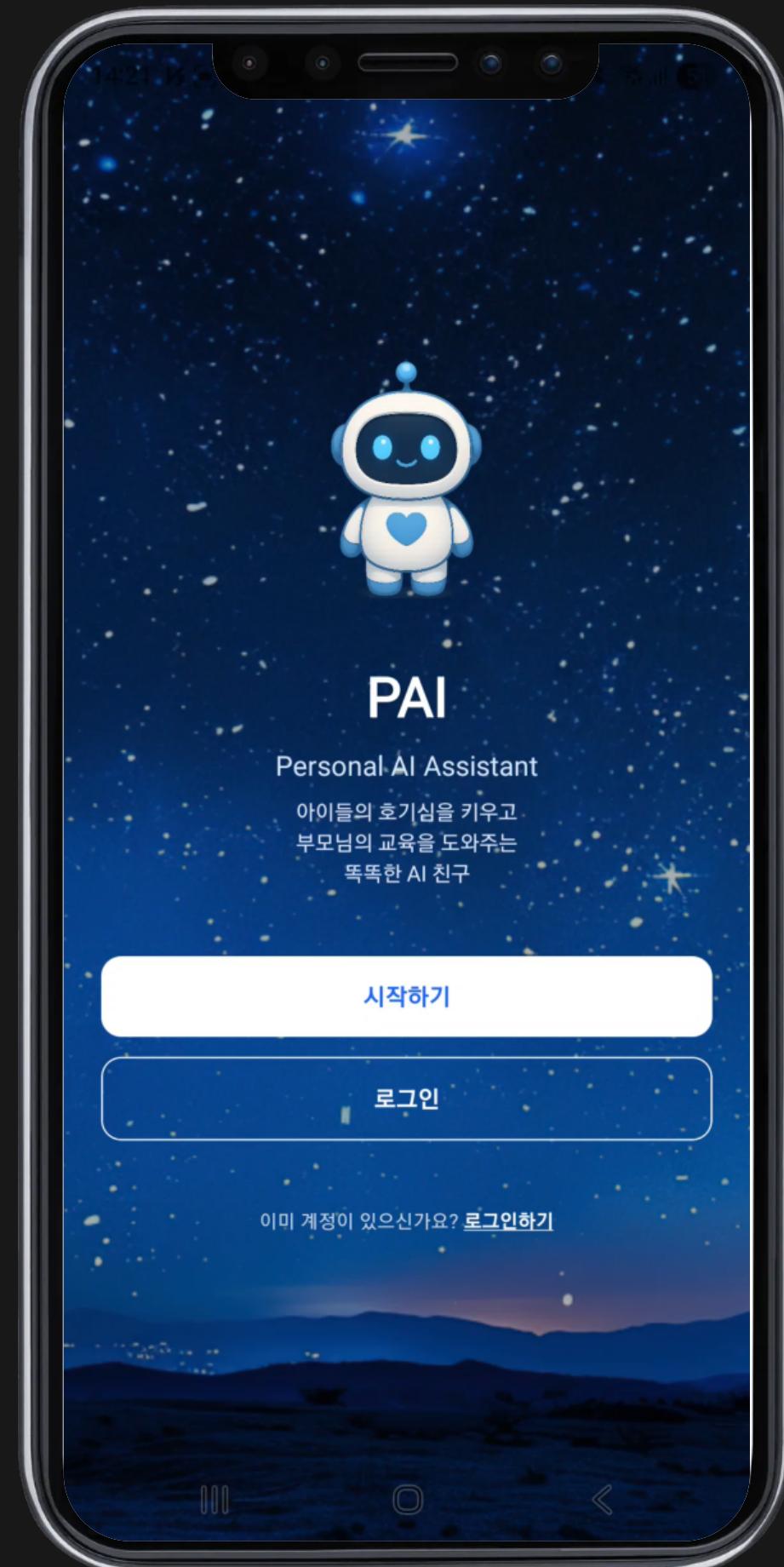
PAI (Parent & I): 아이 질문 기반 지능형 추천 시스템

PAI는 아이의 질문 데이터를 분석해 관심
사를 정교하게 추출하고,
그에 맞는 활동·학습 콘텐츠를 제공함으로
써 부모와 아이가 더 깊이 소통할 수 있도록
돕는 AI 기반 양방향 인터랙션 시스템입니다.

2025년 8월 - 2025년 9월(5주)

총 6명

풀스택 개발자 2명, AI 개발자 4명



// 기술스택

Backend:

Nest.js (TypeScript), FastAPI

Database:

PostgreSQL (Prisma ORM), Redis

Frontend:

React Native (TypeScript), Expo, Zustand, Tanstack Query

Infra:

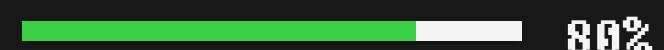
AWS EC2, S3, Docker, Nginx, GitLab CI

AI:

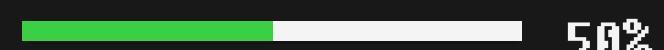
TTS - ElevenLabs

// 진도도

프론트엔드 개발



백엔드 개발

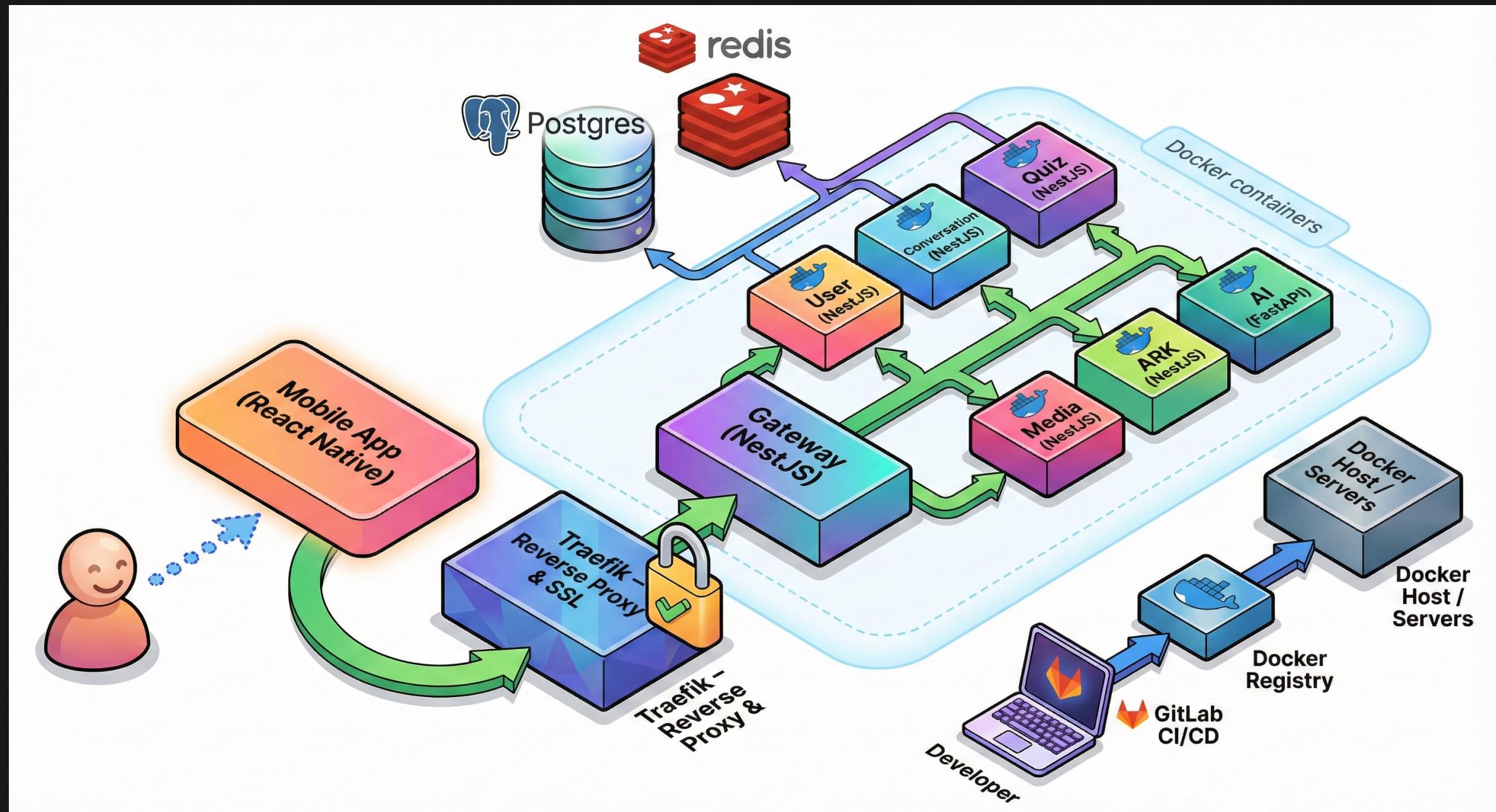


테스트 및 배포



시스템 아키텍처

// MSA & Docker & GitLab CI/CD



[MSA]

NestJS & FastAPI 결합:

Node.js의 높은 동시성 처리 능력과 Python의 AI 생태계를 모두 활용하기 위해 서비스 분리.

Traefik 게이트웨이:

단일 진입점을 통해 라우팅, 로드 밸런싱, SSL 인증을 중앙에서 효율적으로 관리.

[Docker]

서비스 컨테이너화:

6개 이상의 마이크로서비스와 데이터베이스를 독립된 컨테이너로 격리하여 실행.

이식성 확보:

OS에 구애받지 않고 어디서든 동일하게 실행 가능한 애플리케이션 환경 구축.

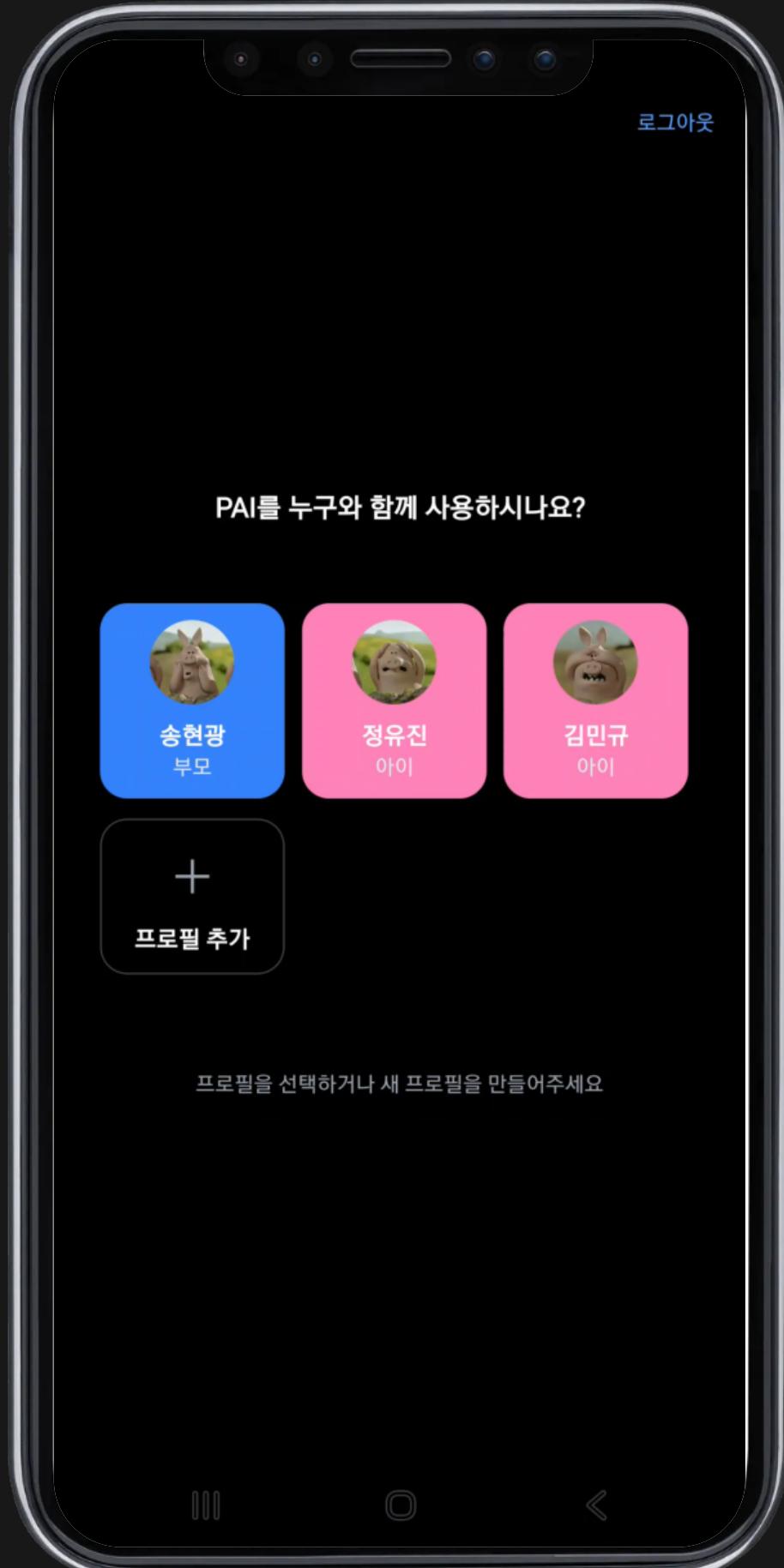
[GitLab CI/CD]

지속적 통합/배포:

GitLab Runner를 활용해 소스 코드 변경 사항을 실시간으로 감지하고 자동으로 서버에 배포

개발 생산성 향상:

반복적인 배포 작업을 자동화하여 개발 효율성 극대화 및 빠른 피드백 루프 달성.



주요 기능 및 담당 업무

// 프로필 선택 스크린(Insight-service)

[개요]

사용자가 앱에 진입할 때 사용할 프로필을 선택하는 화면
한 계정에서 여러 가족 구성원의 프로필을 관리하며, 부모/자녀 역할에
따라 다른 UI/UX를 제공

[핵심 기술]

보안 강화:
bcrypt 12 rounds PIN 해싱
Redis Token Versioning

성능 최적화:
Zustand 캐싱으로 API 호출 최소화
JWT 기반 권한 체크
Promise.all로 이미지 병렬 로딩

[트러블 슈팅]

문제 : 프로필 이미지 로딩 지연으로 UI 깜빡임
해결: Promise.all을 사용하여 모든 이미지 URL을 병렬로 패칭
성과: 4개 프로필 이미지 기준 로딩 시간 1.2초 → 0.3초 (75% 단축)
UI 깜빡임 현상 해결

문제: 로그아웃 시 & 프로필 전환 시 이전 토큰이 계속 유효함
해결 Redis 기반 Token Versioning 도입
성과: 로그아웃 & 프로필 전환 시 토큰 즉시 무효화



주요 기능 및 담당 업무

// 아이 관심사 분석 스크린(Insight-service)

[개요]

자녀 대화에서 AI가 추출한 키워드를 기반으로 아이의 관심사를 자동으로 분석하고 점수화하는 시스템

[핵심 기술]

점수 계산 공식: $3.0 + \min(\text{언급횟수} * 0.5, 3.0)$
최소 3.5점(1회), 최대 6.0점 (6회 이상)

시간 감쇠 공식: $\text{기존 점수} * (0.5 ^ (\text{경과일수}/7))$

[트러블 슈팅]

문제: 오래된 관심사 누적

해결: 스케줄러 적용

라이브러리: @nestjs/schedule

실행 주기: 트래픽이 적은 새벽 3시에 실행

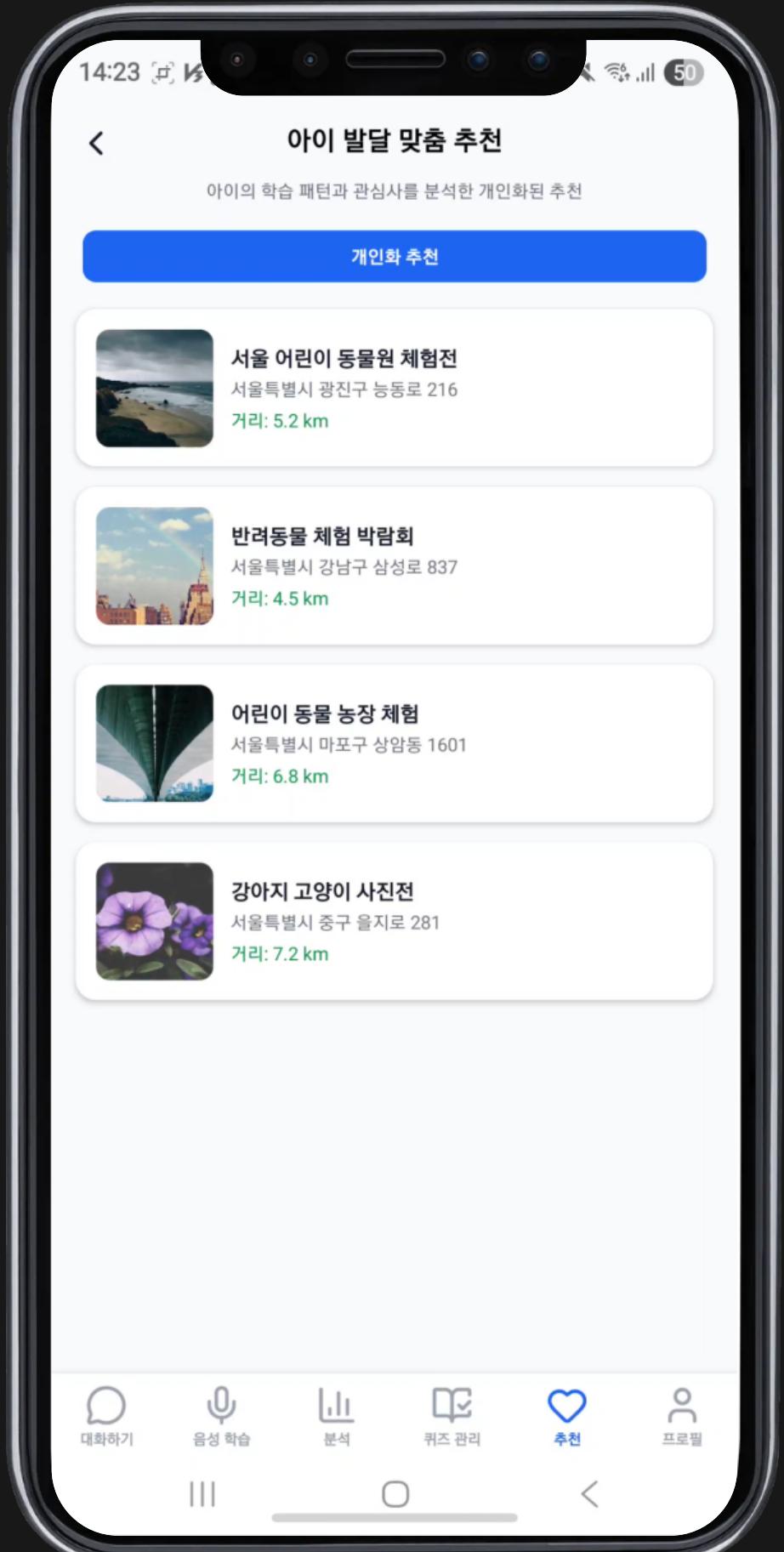
삭제 조건: 14일 이상 업데이트 안 된 관심사 && 점수 1.0 미만인 관심사

성과: DB 용량 감소

문제: 오래된 관심사가 계속 상위에 남음

해결: 시간 감쇠 로직으로 최신 관심사에 비중 할당

성과: 최근 관심사가 상위권에 정확히 반영됨



주요 기능 및 담당 업무

// 관심사 기반 추천 스크린(Insight-service)

[개요]

아이의 최상위 관심사와 가족 위치 정보를 기반으로 한국관광공사 API를 활용하여 맞춤형 콘텐츠를 추천하는 시스템

[핵심 기술]

Haversine 공식 (거리 계산)

Korea tourism API 연동

키워드 매칭 알고리즘

정규식 사용 : \b{keyword}\b

[트러블 슈팅]

문제 : 매 요청마다 외부 API 호출로 인한 응답 지연

해결: Redis 캐싱 도입

성과: 응답시간 80% 단축

문제: 키워드 부분 매칭 오류

해결: 정규식 단어 경계 사용(\b\b) 적용

성과: 정확도 40% → 95%

Project Experience

PAI 리팩터링

MSA, DDD,

kubernetes

서비스의 확장성과 유지보수성을 확보하기 위해 Hexagonal Architecture 와 Domain-Driven Design(DDD)을 적용하여 Clean Architecture 구조로 재구성했습니다.

2025년 10월 - 2025년 12월(ing)

총 2명

풀스택 개발자 2명

// 주요 변경점

아키텍처 개선:

Hexagonal Architecture:

비즈니스 로직과 인프라 계층 분리로 테스트 용이성과 코드 품질 개선

MSA 아키텍처 고도화:

단일 서버를 7개의 독립적인 마이크로 서비스로 분리하여 유지보수성 향상

Domain-Driven-Design 적용:

Entity와 Value Object 패턴으로 도메인 로직 명확화

Shared-types Package:

NPM 패키지로 타입들을 배포하여 백엔드와 프론트엔드 간 타입 일관성 보장, 코드 재사용성 62% 증가

Kubernetes 도입:

독립 배포, 자동 스케일링, 무중단 업데이트

// 마이크로서비스 구성

User Service: 인증/인가, 프로필 관리, 가족 그룹 관리

Media Service: 미디어 업로드/조회, S3 연동, 갤러리 관리

Insight Service: 아이 관심사 분석 및 추천

Quiz Service: 퀴즈 생성/채점

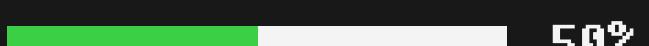
AI Service: VQA, TTS, LLM 통합

Conversation Service: 대화 세션, Redis 캐싱

Shared-types: NPM 패키지 타입 공유

// 기여도

프론트엔드 개발



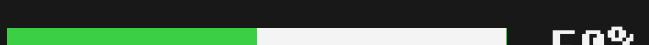
50%

백엔드 개발



50%

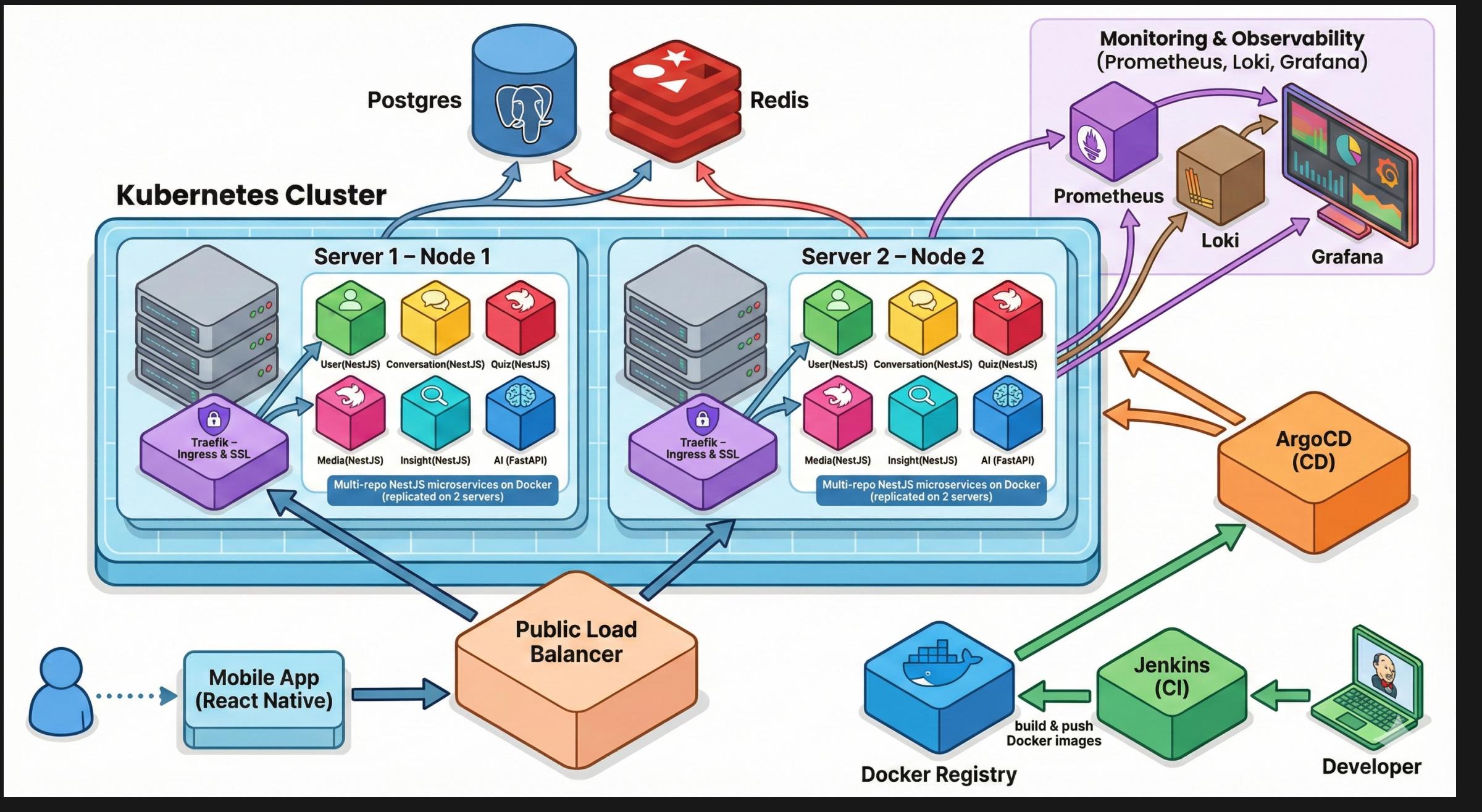
테스트 및 배포



50%

시스템 아키텍처

// MSA & DDD & Kubernetes



[Kubernetes] (진행중)

고가용성 클러스터 구축:

멀티 노드(Node 1, 2) 환경과 Traefik Ingress를 통해 트래픽을 효율적으로 분산하고, 단일 장애점(SPOF)을 제거하여 서비스 가용성 극대화.

GitOps 기반 자동화:

ArgoCD를 활용하여 쿠버네티스 매니페스트 변경 사항을 실시간으로 동기화하고, 선언적 인프라 관리를 통해 운영 복잡도 최소화.

[DDD]

도메인 중심의 경계 설정:

비즈니스 복잡도를 해결하기 위해 핵심 도메인을 기준으로 Bounded Context를 명확히 정의하여 서비스 분리.

[MSA]

최적의 기술 스택 융합:

Node.js(NestJS)의 높은 동시성 처리 능력과 Python(FastAPI)의 강력한 AI 생태계를 결합하여, 각 서비스 목적에 맞는 최적의 기술을 적용.

독립적 확장성:

AI 분석, 캐싱, 유저 관리 등 각 도메인 서비스가 독립적으로 배포 및 스케일링 가능하여, 특정 기능의 트래픽 급증에도 전체 시스템의 안정성 보장.



주요 기능 및 담당 업무

// 아이 관심사 분석 스크린(Insight-service)

[개요]

자녀 대화에서 AI가 추출한 키워드를 기반으로 아이의 관심사를 자동으로 분석하고 점수화하는 시스템

[핵심 기술]

점수 계산 공식: $3.0 + \min(\text{언급횟수} * 0.5, 3.0)$
최소 3.5점(1회), 최대 6.0점 (6회 이상)

시간 감쇠 공식: $\text{기존 점수} * (0.5 ^ (\text{경과일수}/7))$

[트러블 슈팅]

문제: 오래된 관심사 누적

해결: 스케줄러 적용

라이브러리: @nestjs/schedule

실행 주기: 트래픽이 적은 새벽 3시에 실행

삭제 조건: 14일 이상 업데이트 안 된 관심사

성과: DB 용량 감소

문제: 오래된 관심사가 계속 상위에 남음

해결: 시간 감쇠 로직으로 최신 관심사에 비중 할당

성과: 최근 관심사가 상위권에 정확히 반영됨

Project Experience

CHAMBER: 실시간 구역별 스마트 공조 관리 시스템

멀티존 실내 환경을 정밀하게 묘사하는 물리 시뮬레이터와 심층 강화학습(SAC) 기반의 지능형 제어 시스템을 통합하여, 개인맞춤형 쾌적 환경 제공과 에너지 소비 절감을 동시에 달성하는 차세대 스마트 공조 시스템

2025년 7월 - 2025년 8월(7주)

총 6명

프론트엔드 개발자 1명,
백엔드 개발자 2명, 임베디드 개발자 3명

//기술스택

Backend:

Python & FastAPI

Database:

PostgreSQL, SQLAlchemy

Frontend:

Web-application

React Native (TypeScript), Expo, Zustand,
Tanstack Query, Zustand

Mobile-application

Flutter & Dart

Infra:

AWS EC2, Docker, Nginx, traefik

AI:

Python, PyTorch, OpenCV, Gymnasium,
Stable-Baseline3, Numpy, SAC 알고리즘,
MQTT, YOLOv8n-Pose

Hardware:

Jetson Orin Nano, Raspberry Pi

//성과

실시간 WebSocket 연동

접근성 지원

색맹 모드 4종 구현

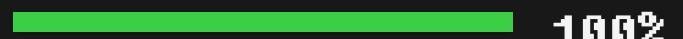
큰 텍스트 지원

장치 제어 기능 및 상태 모니터링 기능 구현

크로스 플랫폼 어플리케이션 구현

//기여도

프론트엔드 개발



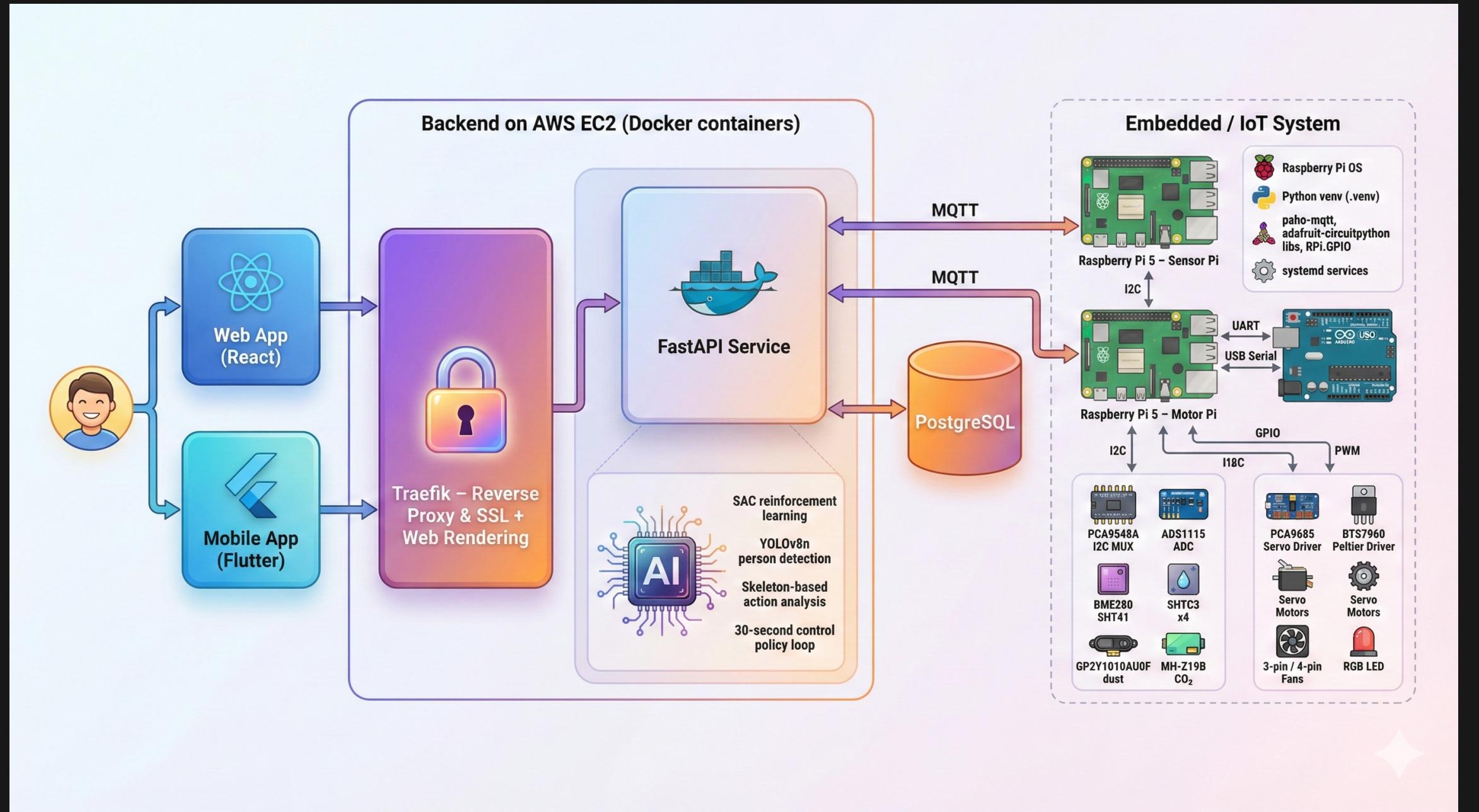
100%

테스트 및 배포



100%

시스템 아키텍처



//React & Flutter, FastAPI

[MSA]

NestJS & FastAPI 결합:

Node.js의 높은 동시성 처리 능력과 Python의 AI 생태계를 모두 활용하기 위해 서비스 분리.

Traefik 게이트웨이:

단일 진입점을 통해 라우팅, 로드 밸런싱, SSL 인증을 중앙에서 효율적으로 관리.

[Docker]

서비스 컨테이너화:

6개 이상의 마이크로서비스와 데이터베이스를 독립된 컨테이너로 격리하여 실행.

이식성 확보:

OS에 구애받지 않고 어디서든 동일하게 실행 가능한 애플리케이션 환경 구축.

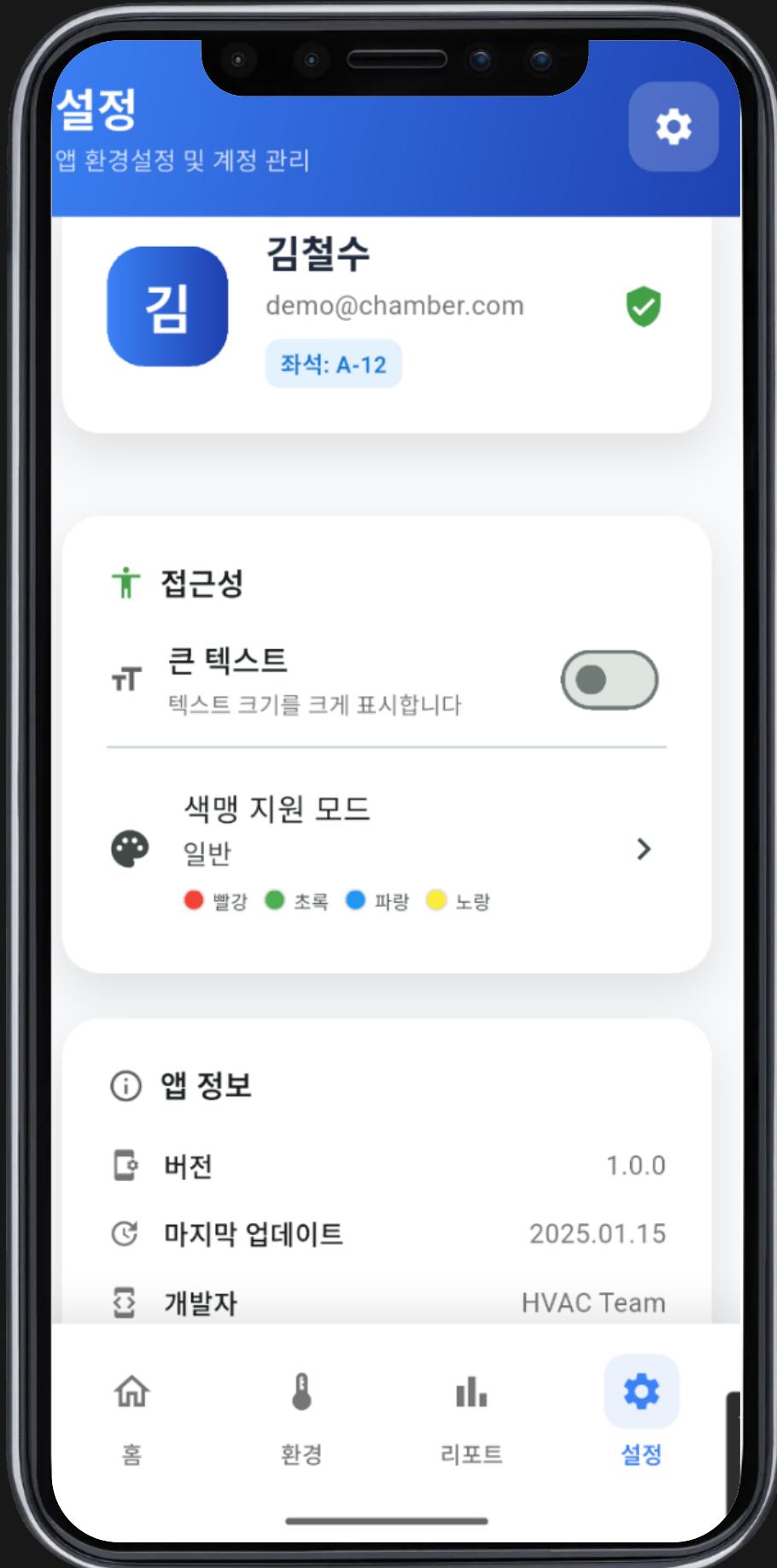
[MSA]

NestJS & FastAPI 결합:

Node.js의 높은 동시성 처리 능력과 Python의 AI 생태계를 모두 활용하기 위해 서비스 분리.

Traefik 게이트웨이:

단일 진입점을 통해 라우팅, 로드 밸런싱, SSL 인증을 중앙에서 효율적으로 관리.



주요 기능 및 담당 업무

// 설정 스크린(Mobile) - 접근성 (색맹 지원)

[개요]

색맹 사용자를 위해 HSV 색상 공간 변환 알고리즘을 적용한 전역 접근성 시스템
적록/청황 색맹 모드를 실시간으로 전환하며, 모든 UI 요소에 자동 적용

[핵심 기술]

SV 색상 공간 변환 알고리즘

Extension 패턴으로 전역 적용
Dart Extension 활용

영구 저장

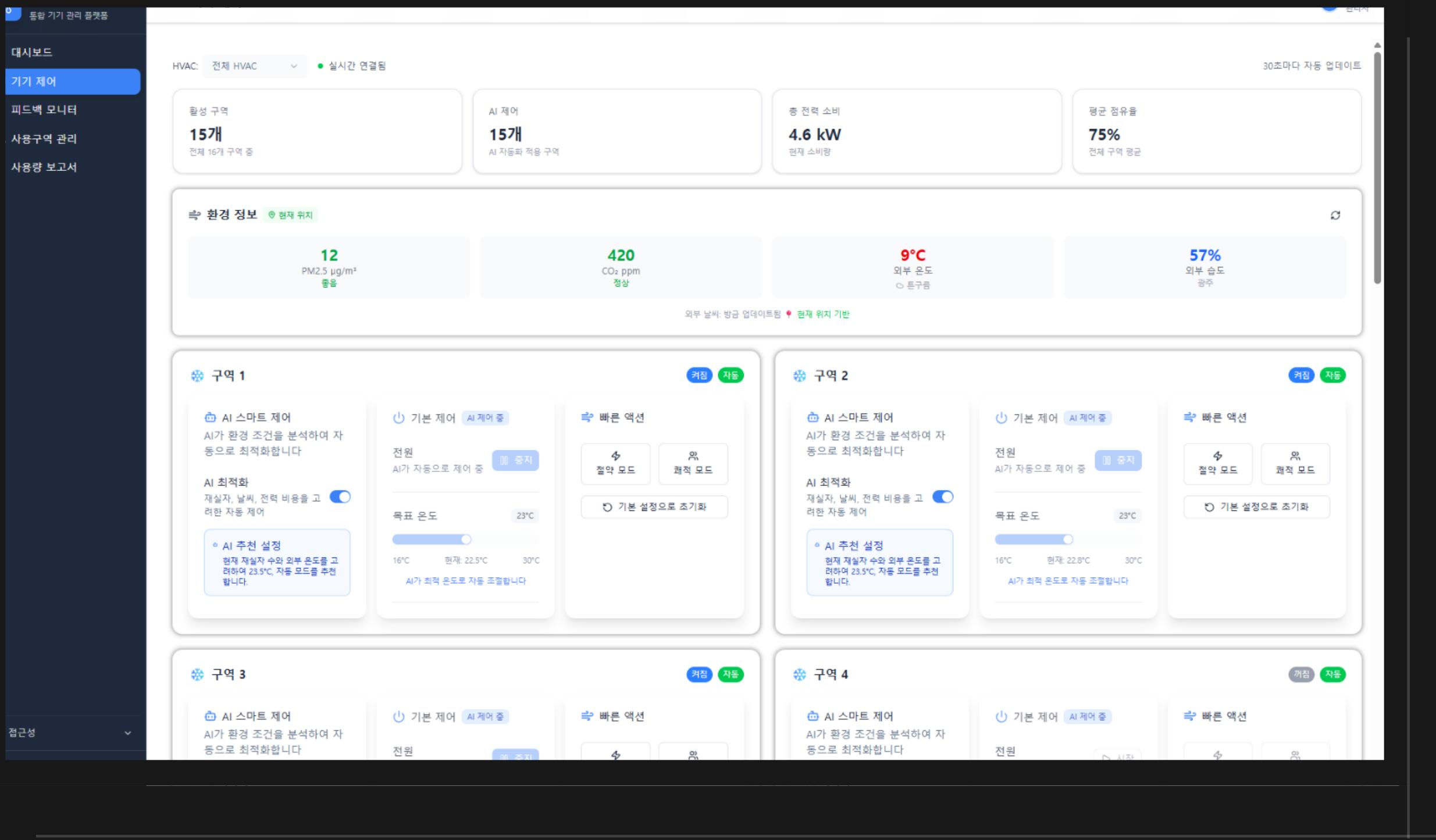
SharedPreferences 사용 -> 앱 재시작 시 자동 적용

[트러블 슈팅]

문제 : 일부 UI 요소만 색맹 모드 적용됨

해결: SV 색상 공간 변환 + Extension으로 전역 적용

성과: 모든 UI 요소 색맹 지원: 100%,
코드 재사용성: Extension으로 보일러플레이트 제거



주요 기능 및 담당 업무

// Device Control Page (Web) - WebSocket 동기화

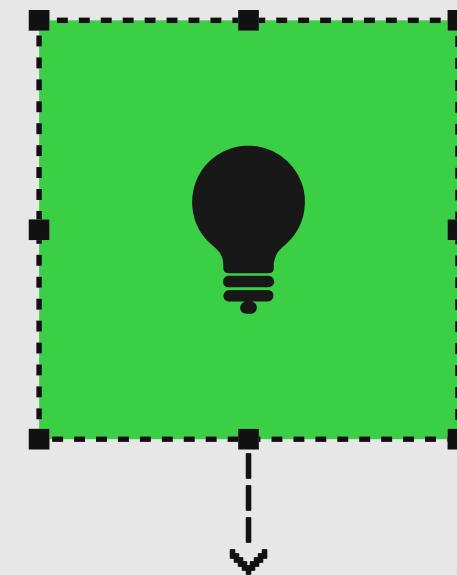
[개요]
관리자가 HVAC 구역별 온도를 조정하는 동안 WebSocket으로 실시간 데이터가 수신되어도 사용자 입력값을 보존하는 시스템.
5초 윈도우 기반 사용자 변경 추적으로 입력 손실 0건 달성.

[핵심 기술]
사용자 변경 추적 (5초 윈도우)
WebSocket 데이터 병합 알고리즘
Dart Extension 활용
WebSocket Provider Factory 패턴

[트러블 슈팅]
문제 : 관리자가 온도 조정 중 WebSocket 데이터로 덮어씌워짐
해결: 5초 윈도우 내 사용자 변경 추적, WebSocket 데이터와 병합
성과: 사용자 입력 보존율: 100%,
입력 손실: 0건,
메모리 효율: 5초 후 자동 정리

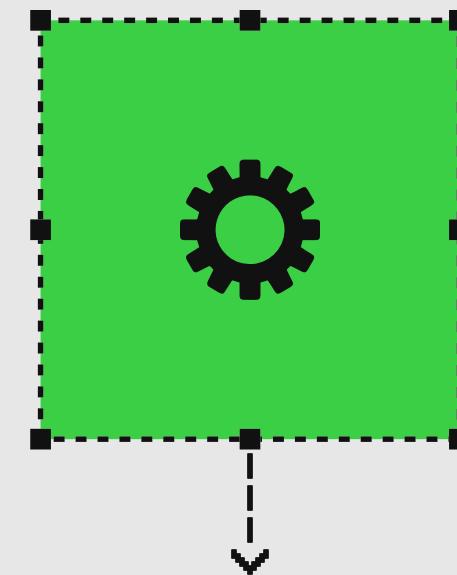
Motivation

지원동기



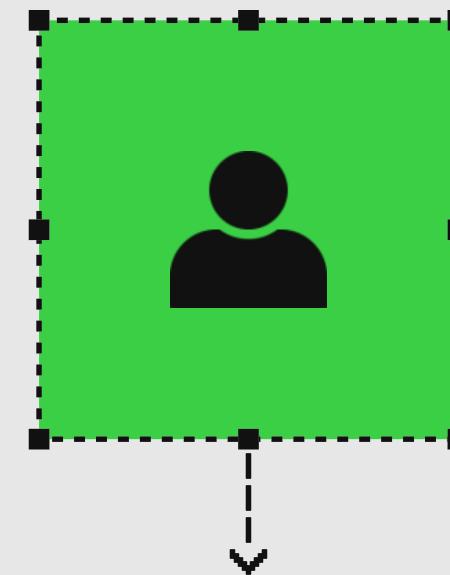
//문제 해결

복잡한 문제를 해결하는 과정에서
느끼는 성취감과 즐거움이
저를 개발자로서 성장하게 만든
원동력입니다.



//기술적 도전

새로운 기술적 도전에
끊임없이 도전하며 이를 통해
더 나은 솔루션을 찾는 것이
저의 개발 철학입니다.



//사용자 중심

사용자의 필요를 깊이 이해하고
이를 기반으로 직관적이고
유용한 제품을 만드는 것이
제가 추구하는 목표입니다.

Contact me

감사합니다.

- 01 {E-mail}
shk9521@naver.com
- 02 {Mobile}
010-5221-8476
- 03 {Website}
<https://www.ggalong.me/>