FT Developer

박 기 찬

코드로
나를
표현하다.

Portfolio
2019 - 2026

프로젝트

- 캄보디아 ITS 시범 구축 사업
- 안전공단 첨단모빌리티 현황조사 시각화
- 성남시 ITS 구축 사업
- 서울동행맵 앱 고도화
- 카본제로-캄보디아
- 러닝메이트:Running Mate
- Walk Planet
- Price Quotation Tool
- Leave Diary

경력

- 2024-01-26 ~ 유아이네트웍스 재직중

수상내역

- 2020 천안아산 지역 대학교 기업분석경진대회 장려상
- 2022 백석대학교 컴퓨터공학부 C언어프로그래밍 경진대회 동상
- 2022 백석대학교 컴퓨터공학부 JAVA프로그래밍 경진대회 은상

학력

- 2024 백석대학교 정보보호학과 졸업
- 2018 우신고등학교 졸업

자격증

- 2025 정보처리기사
- 2021 정보처리 산업기사

IT Developer 박기찬

Tech Stack

Frontend

- React(typescript)
- React Native(typescript)
- Flutter
- Tauri v2

Backend

- springboot
- node.js
- socket.io

Database

- PostgreSQL
- Tibero
- redis
- ClickHouse

Infra

- AWS (ec2, rds, route53, s3)
- nginx
- Tomcat
- Jenkins

IT Developer

산업기능요원

유아이네트웍스

- 2024-01-26 ~ 2025-12-29
- 2024-01-31 산업기능요원 편입
- 2024-06-13 ~ 2024-07-04 훈련소 수료
- 담당 업무:
  - 프론트엔드
    - React-typescript
  - 앱
    - Flutter
  - 백엔드
    - JAVA Spring Boot,
    - e-Government Framework
- 프로젝트:
  - 캄보디아 ITS 시범 구축 사업
  - 안전공단 첨단 모빌리티 현황조사 시각화
  - 성남시 ITS 구축 사업
  - 서울동행맵 앱 고도화
  - 카본제로-캄보디아

Table of Contents

01. PROJECT

박기찬의 프로그래밍 프로젝트

IT Developer

협업 프로젝트

01. 캄보디아 ITS 시범 구축 사업

02. 안전공단 첨단모빌리티 현황조사 시각화

03. 성남시 ITS 구축 사업

04. 서울동행맵 앱 고도화

05. 카본제로-캄보디아

사이드 프로젝트

06. 러닝메이트:RunningMate

07. Walk Planet

08. Price Quotation Tool

09. 이별일기

포트폴리오 웹사이트

제 프로젝트와 기술 경험을 보다 자세히 보고 싶으시다면
아래 링크에서 포트폴리오 페이지를 확인하실 수 있습니다.

URL : https://kichani.com/kichan/main

01. 캄보디아 **ITS** 시범 구축 사업

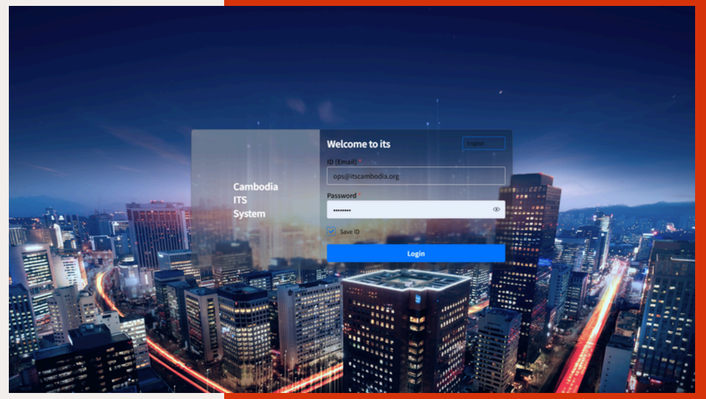
2025-03-06 ~ 2025-11-21

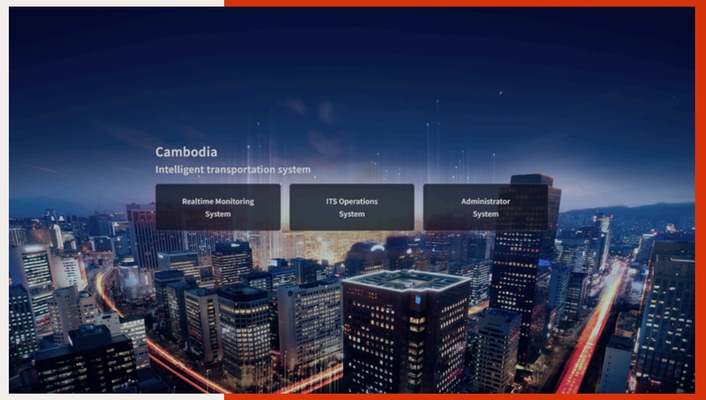
캄보디아 프놈펜의 교통량, 교차로 상태, 속도, CCTV 영상을 실시간으로 수집·분석하여 하나의 대시보드에서 시각화함으로써 효율적인 교통 관제를 지원하는 시범 구축 사업

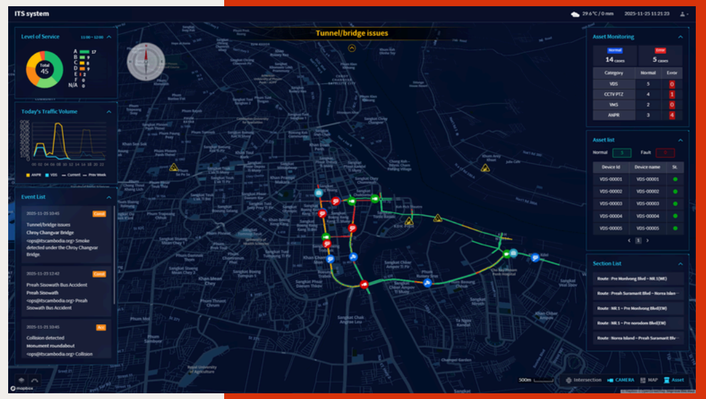
- 역할: 상황판 풀스택 개발 (Front-end / Back-end / WebSocket Server)
- 웹: React(TypeScript) 18 + Vite
- 앱 API: Spring Boot 3.4.1
- 서버: 사내 개발서버
- 데이터베이스: PostgreSQL (with PostGIS), ClickHouse, Redis
- CI/CD: Jenkins

캄보디아 ITS 시범 구축 사업

기간 (2024.03.06 ~ 2025.11.21) | 역할 (상황판 풀스택 개발)







개요

- 캄보디아 프놈펜의 교통량·신호·교차로 상태·속도·CCTV 영상 데이터를 실시간 수집·분석해 하나의 대시보드로 시각화.

Tech Stack

- FrontEnd: React 18, Vite, TypeScript, React Router, TanStack React Query, Mapbox GL JS, react-map-gl, deck.gl(interleaved mode), Emotion/styled-components, D3.js, amCharts, Axios, Socket.IO Client, EventSource(SSE), hls.js, react-hls-player, Video.js, zustand, react-i18next
- BackEnd: Spring Boot 3.4.1, Spring Security, MyBatis, QueryDSL 5.0.0, JWT, Swagger(SpringDoc), SSE(SseEmitter)
- WebSocket: Node.js + Socket.IO
- Database: PostgreSQL(PostGIS), ClickHouse, Redis
- Infra/DevOps: Docker, Jenkins, Nginx, Spring Cloud Gateway, Auth Server(SSO 연동)

Architecture

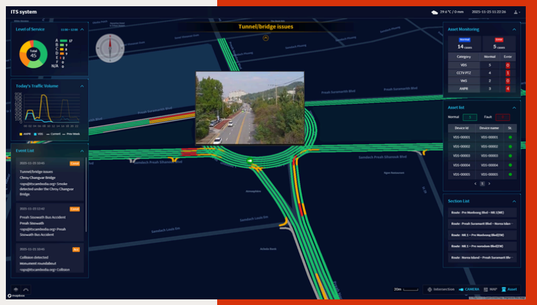
- React > Spring Cloud Gateway > Auth Server(JWT) > Spring Boot API > PostgreSQL
- React > Socket.IO Server(Node.js) > PostgreSQL LISTEN/NOTIFY
- React > SSE(Spring Boot) > ClickHouse

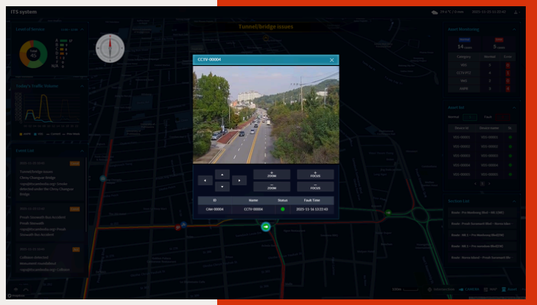
핵심 요구사항

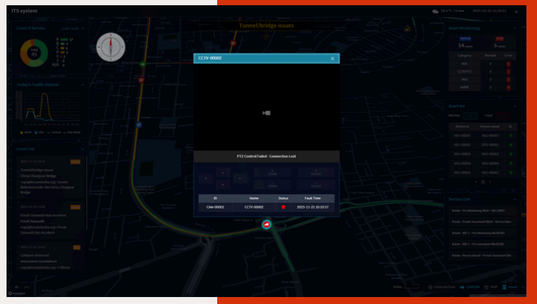
- 교통량·신호·교차로·CCTV 데이터를 실시간으로 모니터링 가능해야 했다.
- 다중 지표 기반 지도/차트 시각화와 빠른 화면 갱신 성능을 확보해야 했다.
- 초 단위 TickChart(주식 차트 형태)의 실시간 교통량 스트리밍을 지원해야 했다.
- 현지 외 국가에도 확장 가능한 범용 글로벌 플랫폼 구조가 필요했다.

핵심 개발사항

- 상황판 프론트엔드: React 18 + TypeScript + Vite 기반 SPA, Socket.IO 클라이언트 및 EventSource(SSE) 실시간 데이터 수신
- 상황판 백엔드: Spring Boot 3.4.1 RESTful API, MyBatis/QueryDSL 복합 쿼리, SSE 스트리밍 엔드포인트 구현
- WebSocket 서버: Node.js + Socket.IO + PostgreSQL LISTEN/NOTIFY 기반 실시간 메시징 서버
- DB 이중화: PostgreSQL(5분 단위 가공 데이터, 실시간 조회) + ClickHouse(초 단위 원천 데이터, 대용량 통계)
- SSE 실시간 스트리밍: ClickHouse 초 단위 데이터 폴링 → SseEmitter → EventSource → TickChart 업데이트
- MSA Gateway 연동: Spring Cloud Gateway를 통한 API 라우팅, 쿠키 기반 JWT 인증 필터 통합
- Auth 서버 연동: 기 구축된 Spring Boot 기반 JWT 인증 서버와 SSO 구현
- 글로벌 타임존 처리: 프론트엔드에서 국가 코드 전달 → 백엔드에서 UTC 기준 변환 → 통계 API 시간 범위 계산
- 지도 시각화: Mapbox GL JS + deck.gl interleaved mode로 레이어 렌더링 이슈 해결, 교통 데이터 오버레이
- 차트 및 통계: D3.js, amCharts를 활용한 다양한 시각화, SSE 기반 실시간 TickChart
- CI/CD: Jenkins 파이프라인 (빌드 → 테스트 → Docker 이미지 생성 → 컨테이너 배포)
- Docker 컨테이너화: Front-end, Back-end, WebSocket 서버 각각 Dockerfile 작성 및 배포 자동화
- 디바운싱 처리: PostgreSQL 고빈도 NOTIFY를 5초 디바운싱으로 WebSocket 트래픽 최적화
- 성능 테스트: Python으로 실제와 유사한 대량 교통 데이터 생성 프로그램 개발, PostgreSQL/ClickHouse INSERT를 통한 실시간·대용량 데이터 처리 시나리오 검증
- HLS 영상 스트리밍: 협력사 HLS 서버 연동, hls.js 기반 CCTV 실시간 영상 재생, 다중 영상 동시 모니터링 UI 구현

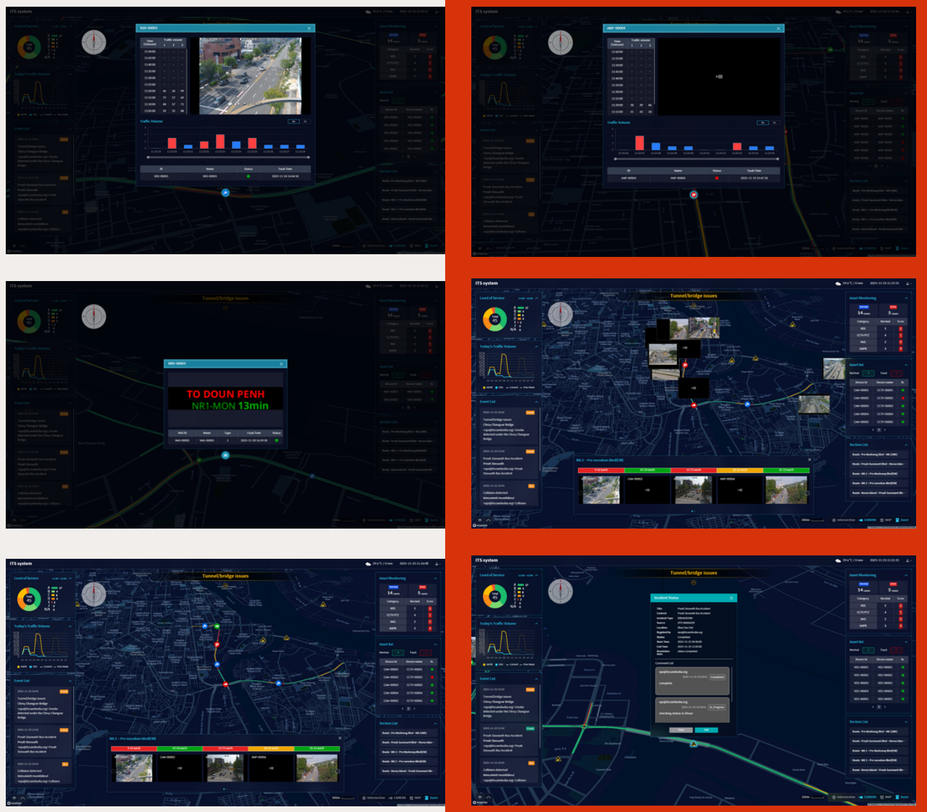






주요 이슈 & 해결

- 지도 레이어가 지도 레이블을 가리는 문제 → interleaved: true 옵션을 사용해 deck.gl 레이어를 Mapbox 내부 레이어 스택에 삽입하여 레이블과 도로명이 정상적으로 노출되도록 개선
- 여러 국가에 공통 적용 가능한 시간 관리 방식 부재 → DB·백엔드 시간을 UTC로 통일하고 프론트의 국가 코드 기반 타임존 변환 로직을 도입해 해외 ITS 사업 확장성 확보
- DB 변경사항을 실시간으로 대시보드에 반영하기 어려움 → PostgreSQL LISTEN/NOTIFY + Node.js Socket.IO 연동으로 DB 트리거 발생 시 즉시 WebSocket 브로드캐스팅
- 고빈도 실시간 이벤트로 인한 WebSocket 과부하 → 채널별 5초 디바운싱 처리로 다수 알림을 단일 알림으로 통합해 트래픽 및 리렌더링 부담 해소
- 1초/5초 교통량을 주식 차트처럼 실시간 스트리밍해야 하는 요구 → ClickHouse > SSE > EventSource 기반 단방향 스트리밍 구조로 TickChart 실시간 렌더링 구현
- MSA 환경 내에서의 JWT 인증 처리 복잡성 → Spring Cloud Gateway에서 쿠키 기반 JWT 검증 후 사용자 정보를 헤더에 포함해 각 서비스로 전달

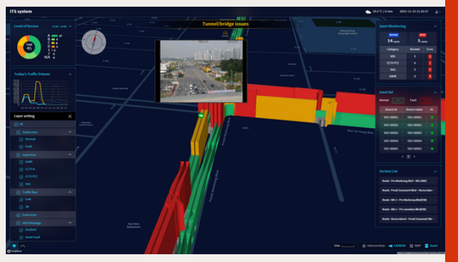
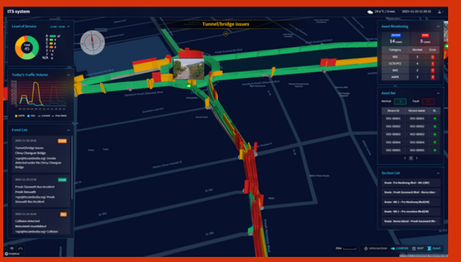


설계

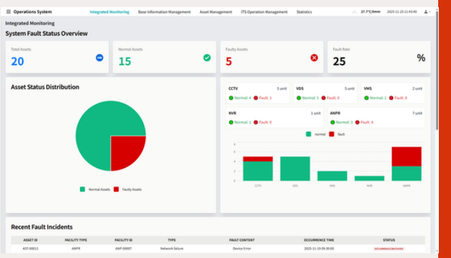
- 기술 선정 기준: 유행이 아닌 ‘문제를 가장 확실하게 해결할 수 있는지’를 기준으로 기술 스택을 선택.
- 지도 시각화: MapboxOverlay + interleaved 모드로 Mapbox GL과 deck.gl을 동일 WebGL 스택에서 렌더링해 레이블 가림 문제 방지.
- 실시간 통신: Kafka 대신 PostgreSQL LISTEN/NOTIFY + Socket.IO를 채택하여 요구사항 대비 최적 비용·저지연 구조 구현.
- 초 단위 스트리밍: TickChart는 양방향 통신이 불필요하므로 WebSocket 대신 SSE 기반 단방향 실시간 스트리밍으로 구성.
- 시간 관리 전략: 백엔드·DB를 UTC 고정, 프론트 국가코드 기반 타임존 변환으로 해외 다국가 확장성 확보.
- 핵심 설계 원칙: 다양한 기술 도입보다 ‘왜 이 기술이어야 했는가’를 설명할 수 있는 선택을 우선.

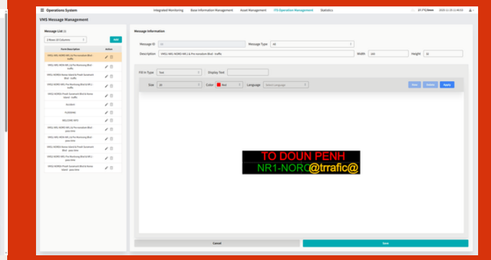
성과

- 상황판 전체를 단독 개발하여 실시간 교통 모니터링 대시보드 구축 완료
- 해외 여러 ITS 사업에 범용 적용 가능한 UTC 기반 글로벌 플랫폼 설계
- PostgreSQL LISTEN/NOTIFY + Socket.IO 조합으로 저지연 실시간 데이터 전파 시스템 구현
- SSE 기반 실시간 TickChart로 VDS/ANPR 장비의 1초/5초 단위 교통량 시각화
- ClickHouse 도입으로 대용량 통계 쿼리 성능 대폭 향상 및 DB 이중화로 역할 분산
- deck.gl interleaved mode 적용으로 이전 프로젝트의 레이어 렌더링 이슈 해결
- MSA Gateway/Auth 서버 연동으로 쿠키 기반 SSO 통합 인증 환경 구축
- Jenkins + Docker 기반 CI/CD 파이프라인으로 배포 자동화 및 안정성 향상
- 디바운싱 처리를 통한 WebSocket 트래픽 최적화로 클라이언트 성능 개선







한국교통안전공단

02. 안전공단 첨단모빌리티 현황조사 시각화

2024-11-10 ~ 2025-03-21

국가 법정 ‘첨단모빌리티 현황조사’ 결과를 한눈에 보여주는 인터랙티브 시각화 대시보드로, 지자체 정책 의사결정을 지원합니다.

- 역할: 원천 데이터 가공, 첨단모빌리티 현황조사 시각화
- 웹: React(TypeScript)
- 데이터 가공: Python
- CI/CD: Jenkins(개발시에만)

URL: https://www.kotsa.or.kr/ptc/common/cdSheet.do

안전공단 첨단모빌리티 현황조사 시각화

기간 (2024.11.10 ~ 2025.03.21) | 역할 (데이터 가공·시각화 단독 개발)

개요

- 국가 법정 '첨단모빌리티 현황조사' 결과를 한눈에 보여주는 인터랙티브 대시보드.

Tech Stack

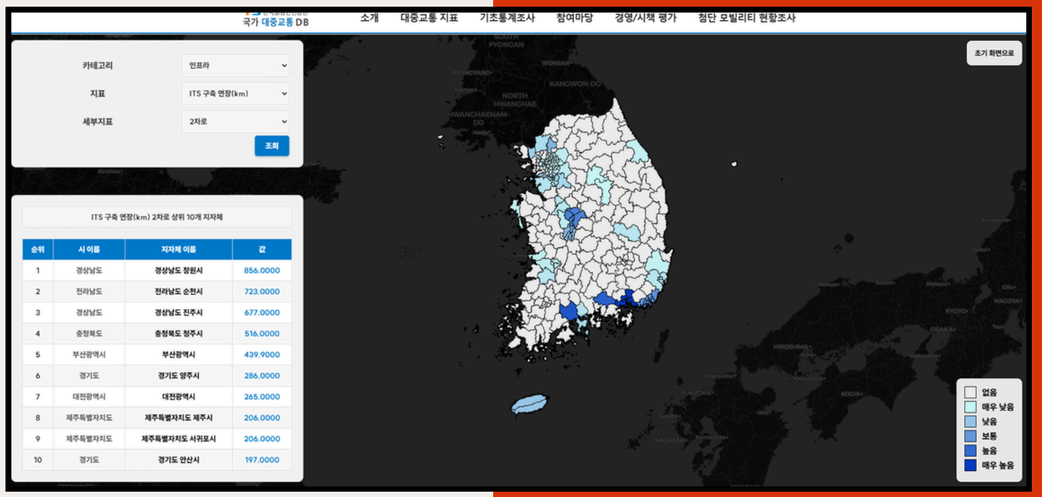
- FrontEnd: React(TypeScript), deck.gl(GeoJson/Text/Tile)
- Data Processing: Python(ETL)
- Infra/DevOps: Jenkins(개발시), OSM Dark Tiles(지도 베이스)

Architecture

- Python ETL(Excel → GeoJSON) > React(deck.gl 시각화)

핵심 요구사항

- Excel 원천 데이터를 ETL 파이프라인으로 가공해야 했다.
- 지표(만족도, 수요응답형, 앱 사용, PSI, 인프라)를 표준화해야 했다.
- 라이센스 비용 없는 운영을 해야 했다.



핵심 개발사항

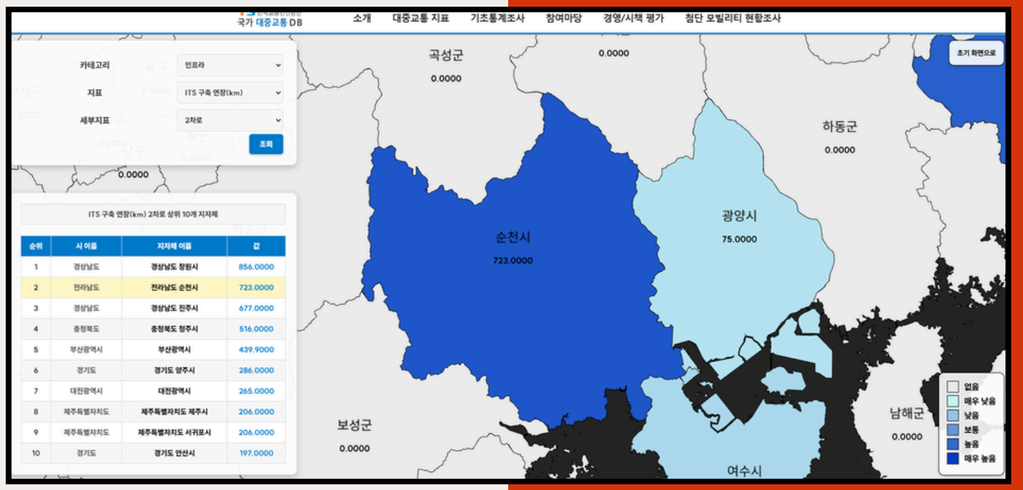
- 데이터 파이프라인(ETL): Excel → GeoJSON 변환, 좌표계(WGS84) 정합·센트로이드 계산, 스키마 표준화.
- 정적 데이터 제공: 가공 결과를 JSON으로 산출해 프론트에서 즉시 로딩 가능하게 구성.
- 지도 시각화: deck.gl GeoJsonLayer로 영역 채움, TextLayer로 시군구 라벨·값 표시.
- 탐색/인터랙션: Top 10 테이블과 지도 연동(행 클릭 시 해당 지자체로 fly-to).
- 성능 최적화: useMemo와 updateTriggers를 활용해 필요한 경우만 리렌더링.

설계

- ETL 중심 구조: 원천 Excel을 Python으로 전처리 후 JSON으로 제공, 프론트 단 단순화.
- 데이터 시각화 전략: 라이선스 이슈 없는 OSM 타일 사용, deck.gl 레이어 기반으로 확장성 확보.

성과

- 대시보드 탐색 성능과 가독성이 개선돼 정책 활용성이 높아짐.
- 운영 환경에서 라이선스 비용 없이 안정적인 지도 시각화 운영 가능.



03. 성남시 ITS 구축 사업

2024-11-10 ~ 2025-03-21

성남시의 교통량·신호·교차로 상태·CCTV 영상 정보 등을 실시간으로 수집·분석하고, 이를 대시보드 형태로 시각화하여 효율적인 교통관제와 관리를 지원

- 역할: 성남시 ITS 대시보드 풀스택 개발
- 웹: React(TypeScript) 18 + Vite
- 앱 API: Spring Boot 2
- 서버: 성남시청 폐쇄망
- 데이터베이스: Tibero
- CI/CD: Jenkins

성남시 ITS 구축 사업

기간 (2024.11.10 ~ 2025.03.21) | 역할 (대시보드 풀스택 개발)

개요

- 성남시의 교통량·신호·교차로 상태·CCTV 영상 데이터를 실시간 수집·분석해 대시보드로 시각화.
- 교통관제 인력이 효율적으로 모니터링·대응할 수 있도록 지원하는 시스템.

Tech Stack

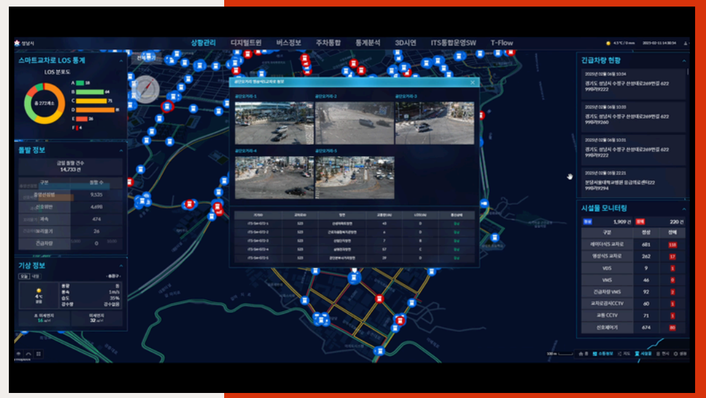
- FrontEnd: React 18, Vite, TypeScript/JavaScript, React Router, TanStack React Query, Mapbox GL JS, react-map-gl, deck.gl, Kakao Maps SDK, Emotion/styled-components, D3.js, Axios, Video.js, React HLS Player, amCharts
- BackEnd: Spring Boot 2, MyBatis, QueryDSL 5.0.0, JWT, Swagger(SpringDoc), HikariCP
- WebSocket: Spring Boot STOMP
- Database: Tibero(ITS/BIS/Dashboard 3 스키마)
- Infra/DevOps: 성남시청 폐쇄망, Nginx, Jenkins

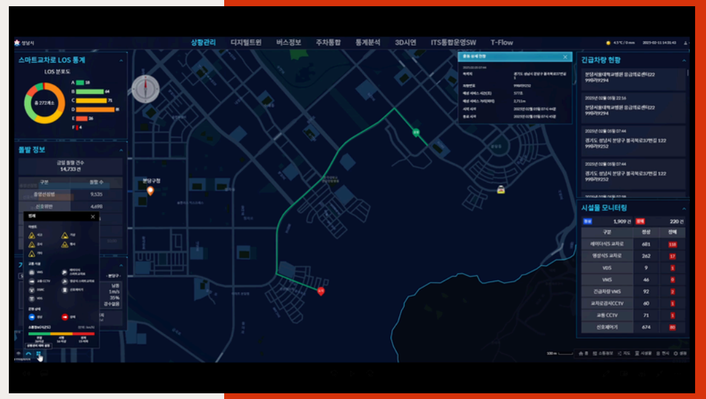
Architecture

- React(FrontEnd) > Spring Boot API/STOMP(WebSocket) > Tibero DB

핵심 요구사항

- 교통량 그래프와 CCTV 영상을 실시간으로 제공해야 했다.
- 교차로 상세 UI와 실시간 차량 데이터 성능을 최적화해야 했다.
- 대규모 WebSocket(STOMP) 메시지 처리에서 안정성을 확보해야 했다.
- 통계분석 페이지에서 다양한 지표·필터 기반 데이터 시각화를 지원해야 했다.



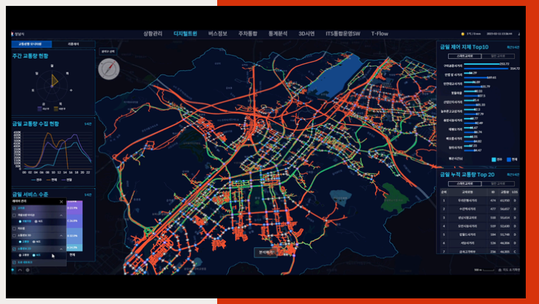


핵심 개발사항

- 교통량 그래프 API: 교차로·접근로·시간버킷 단위 집계 스키마 정의 → Spring Boot + MyBatis/QueryDSL로 API 제공.
- 프론트 연동/시각화: React Query로 API 바인딩, amCharts로 팝업 내 그래프 렌더링 및 로딩/에러 처리 구현.
- 데이터 정합성 확보: 집계 기준(시간대·장비 상태 코드·조인 방식) 재정의 후 쿼리 리팩토링, 대시보드 표출값 불일치 문제 해결.
- 지도 나침반 위젯: Mapbox bearing 기반 나침반 구현, 방위 표시와 클릭 스냅 기능으로 상황판 공간 인지성 강화.
- 긴급차량 경로 보정: 속도·거리 기반 비정상 좌표 자동 감지·보정, 경로 단절 없이 매끄럽게 연결.
- 실시간 차량 데이터 최적화: 초기 렌더링 10초 → STOMP 메시지 중복 제거(Set) + useMemo 캐싱 적용, <1초 단축.
- 메모리 안정화: WebSocket(STOMP) 데이터 슬라이딩 윈도 전략 적용, 장시간 실행에서도 Out of Memory 방지.
- 통계분석 페이지: amCharts 도입·표준화, 기간/권역/교차로 필터 적용, 교차로 방향(동·서·남·북) 시각화 컴포넌트 구현.
- PDF/Excel 내보내기: 화면 조건(필터/기간)을 반영해 동일 데이터로 산출물 생성, 교차로 이미지+표 동시 포함.

주요 이슈 & 해결

- 초기 차량 렌더링 지연(10초 이상) → STOMP 메시지 중복 제거 + useMemo 캐싱 → 1초 미만으로 단축.
- 장시간 구동 시 메모리 누수 발생 → 슬라이딩 윈도 방식으로 최신 데이터만 유지 → 장시간 안정 구동 확보.
- Mapbox 팝업이 deck.gl에 가려짐 → 기획자와 협의해 우측 패널 UI로 대체 → 일정 지연 방지.
- 교통량 지표 불일치 → 집계 기준 재정의 및 쿼리 리팩토링 → API·화면 데이터 정합성 확보.



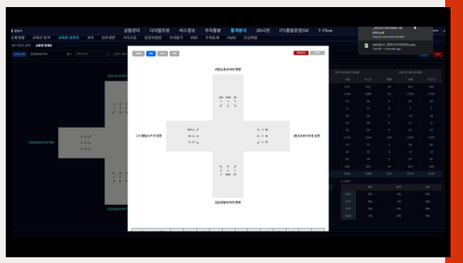
설계

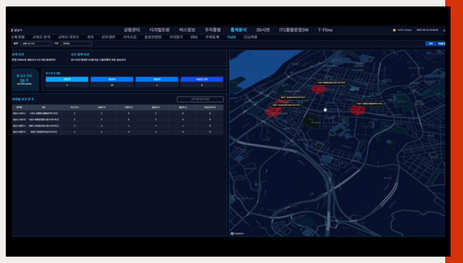
- 실시간 통신(WebSocket/STOMP): Spring Boot STOMP를 채택, 클라이언트 구독/발행 구조로 안정적인 실시간 데이터 전송.
- UI 전략: deck.gl 오버레이 대신 패널 UI를 적용, 유지보수성과 일정 안정성 확보.
- 차트 표준화: amCharts를 공통 컴포넌트화, 필터·상태 변경 시 로딩/에러 처리 일관성 확보.
- 산출물 구조: PDF·Excel 내보내기에 동일 데이터 소스를 반영, 운영 효율성 강화.

성과

- 실시간 차량 데이터 렌더링 속도를 10초 → <1초로 단축.
- STOMP 데이터 메모리 관리 개선으로 장시간 구동 안정성 확보.
- 통계·시각화 페이지 표준화로 유지보수성 강화.
- 대시보드 데이터 정합성 확보로 운영 신뢰도 제고.








서울동행맵

04. 서울동행맵 앱 고도화

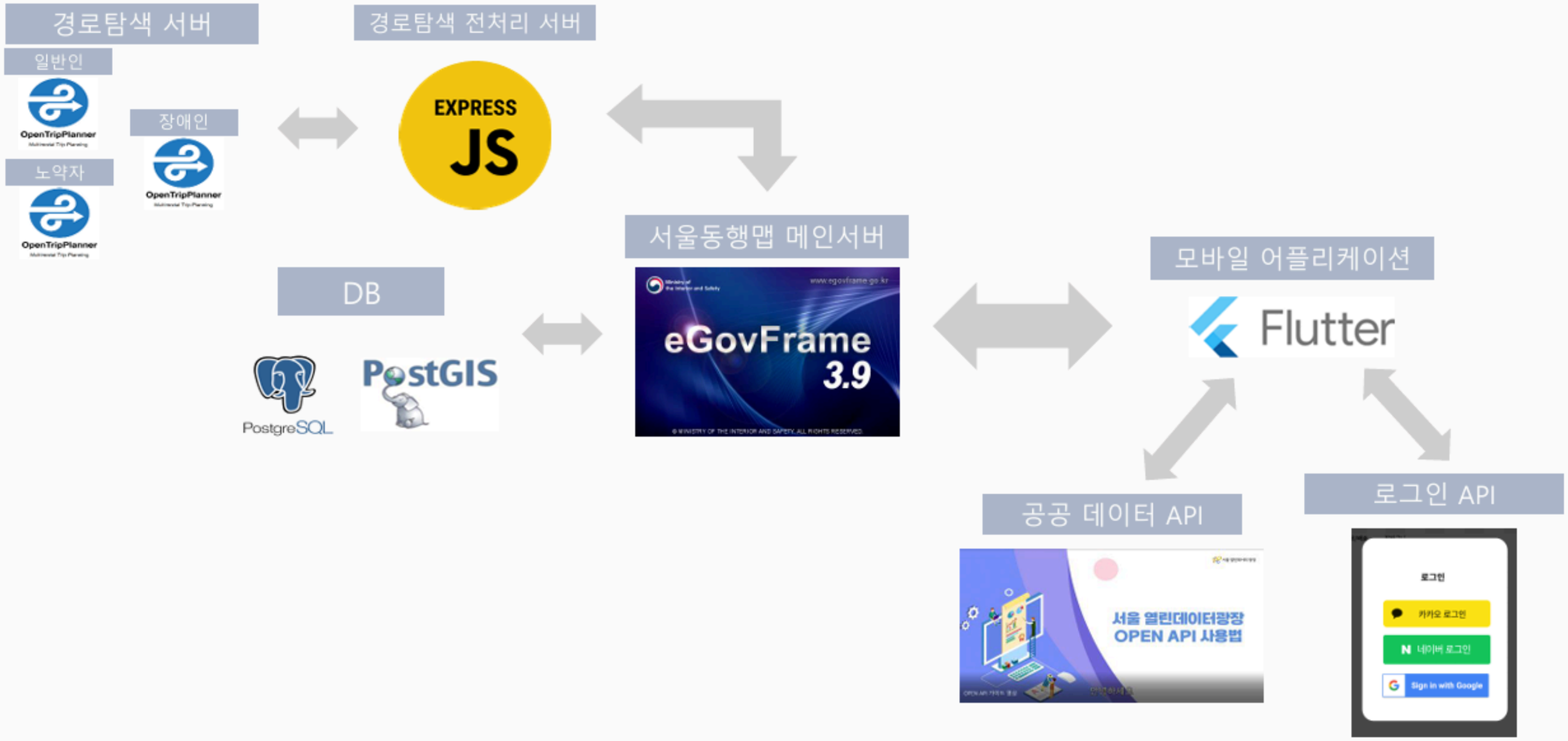
2024-08-20 ~ 2024-11-30

교통 약자에게 맞춤형 교통 정보를 제공해 대중교통 접근성을 높이고 이용을 활성화 하기 위한 교통 약자 앱

- 역할: 앱 개발 고도화, 전체적인 프로젝트 유지 보수
- 앱: Flutter
- 앱 API: e-Government Framework
- 서버: 상암 에스플렉스 센터 폐쇄망
- 데이터베이스: PostgreSQL, PostGIS
- CI/CD: API는 직접 빌드하여 폐쇄망에 배포 앱은 플레이스토어, 앱스토어, 원스토어에 배포

Play Store: https://play.google.com/store/apps/details?id=kr.go.seoul.mydata&hl=ko
App Store: https://apps.apple.com/kr/app/%EC%84%9C%EC%9A%B8%EB%8F%99%ED%96%89%EB%A7%B5/id1555649324
One Store: https://m.onestore.co.kr/ko-kr/apps/appsDetail.omp?prodId=0000753480

서울동행맵 프로젝트 구성



서울동행맵 앱 고도화

기간 (2024.08.20 ~ 2024.11.30) | 역할 (앱 고도화, 유지보수)

개요

- 교통 약자에게 맞춤형 대중교통 정보를 제공해 접근성을 높이는 앱.
- Flutter 기반 앱, 폐쇄망 API, OTP(OpenTripPlanner) 미들웨어를 연동해 서비스 품질을 개선.

Tech Stack

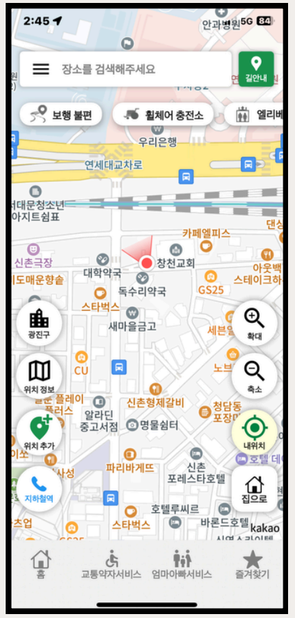
- FrontEnd(App): Flutter
- BackEnd(API): e-Government Framework
- Middleware: OTP(OpenTripPlanner), 경로탐색 미들웨어
- Database: PostgreSQL(+PostGIS)
- Infra: 상암 에스플렉스 센터 폐쇄망, Jenkins(로컬 빌드/배포)

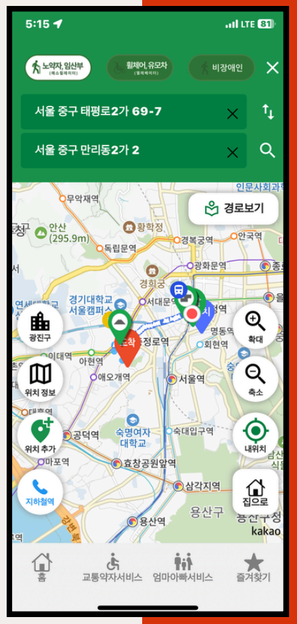
Architecture

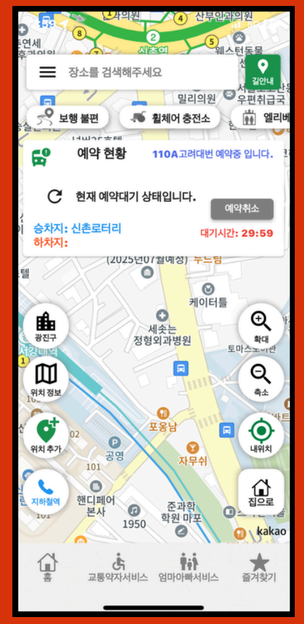
- Flutter(App) > API(e-Gov Framework) > Middleware(OTP) > DB(PostgreSQL+PostGIS)

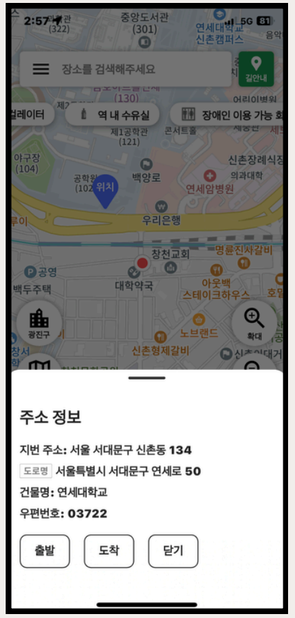
핵심 요구사항

- 스크린리더 기반 접근성을 강화해야 했다.
- 지도 중심 UX의 한계를 보완할 대체 UI가 필요했다.
- OTP 기반 경로 탐색 품질을 개선해야 했다.
- 저상버스·편의시설 등 교통 약자 특화 정보를 제공해야 했다.









핵심 개발사항

- 접근성 개선: 버튼/이미지 라벨링, 읽기 순서 정리, 중복 읽힘 제거.
- 지도 대체 UX: 마커 목록·상세를 패널 위젯으로 제공, 전용 확대/축소 버튼 추가.
- 시각적 개선: 색상 대비 ≥3:1 조정, 활성/선택 상태를 색상+굵기로 병행 표현.
- 교통 약자 기능: 내 위치 실시간 토글, 저상버스 예약 현황 갱신, 편의시설(경사로·복지시설·충전소·장애인 화장실) 필터 제공.
- 길찾기 UI 개선: 소요시간 막대에 노선 정보 표기, 앱 시작 시 공지 팝업 제공.
- 경로탐색 파이프라인: GTFS + OSM 기반 Graph.obj 빌드, router-config.json 튜닝(환승/대기/도보 가중치)으로 탐색 정확도 개선.

주요 이슈 & 해결

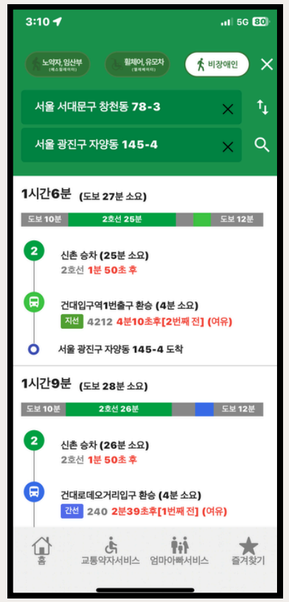
- 지도 접근성 한계 → 위젯 패널로 대체 UI를 제공해 스크린리더 사용자도 탐색 가능.
- 저상버스 예약 현황 지연 → 갱신 버튼 및 API 로직 추가로 실시간 반영 가능.
- 경로 탐색 품질 저하 → OTP 파라미터(환승·대기·도보 가중치) 조정으로 경로 품질 개선.

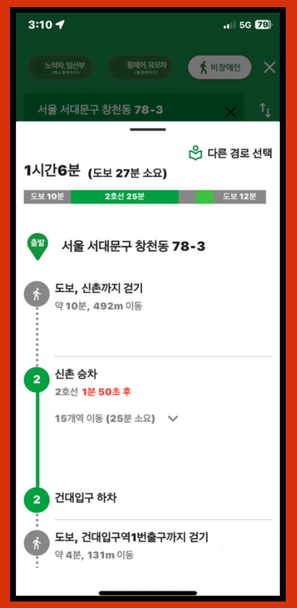
설계

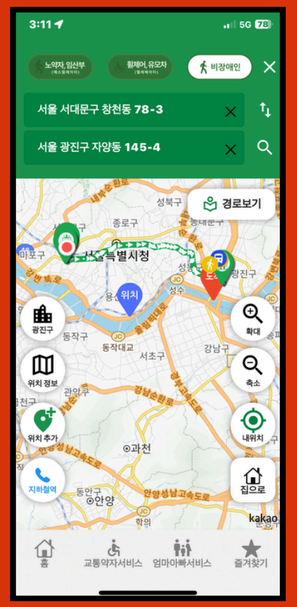
- 접근성 전략: 스크린리더·시각 대비를 병행 적용해 시각장애인 사용성을 강화.
- OTP 미들웨어: 경로 탐색 전용 계층으로 구성해 다중 소스(버스·지하철·도보)를 통합 처리.
- 데이터 소스 관리: 버스는 사내 DB, 지하철은 웹 크롤링, 도보 네트워크는 my-T 편집 툴로 가공.

성과

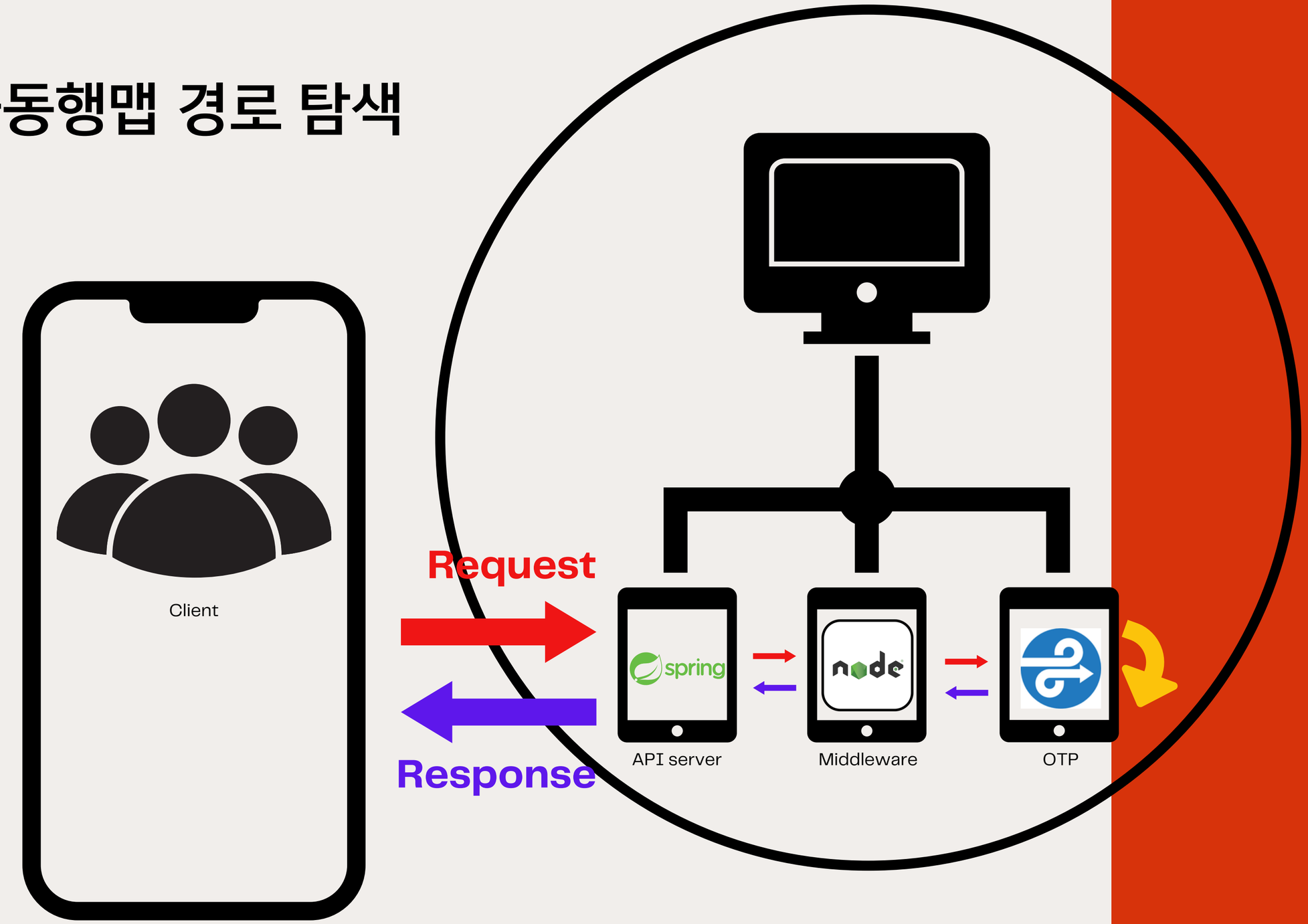
- 스크린리더와 대체 위젯 UX를 통해 접근성을 확보.
- OTP 튜닝과 다중 소스 통합으로 경로 탐색 정확도 향상.
- 교통 약자 대상 편의 기능 추가로 실사용 탐색 흐름 개선.







서울동행맵 경로 탐색


Client
Request
Response
spring
node
API server
Middleware
OTP

CARBON ZERO

05. 카본제로 - 캄보디아

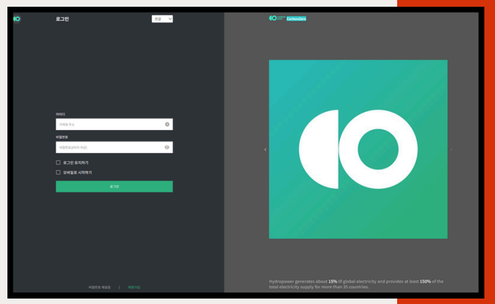
2024-01-31 ~ 2024-08-10

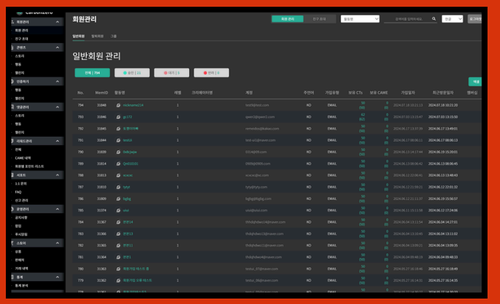
기후 행동을 통해 CTs(포인트)를 얻어 CTreeMall(교환소)에서 CTs로 물건을 살 수 있는 기후 행동 앱

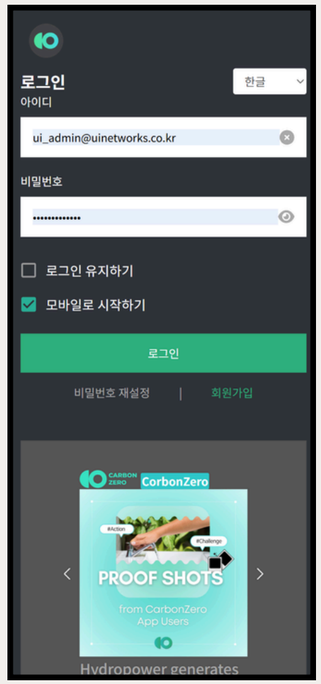
- 역할: 관리자 페이지 풀 스택 개발, 앱 API 개발, 서버 및 데이터베이스 구축(EC2, RDS, nginx, SSL, 로드밸런서)
- 앱: Flutter
- 관리자 페이지: React(typescript)
- 앱 API, 관리자 페이지 API: Spingboot 2.5.6
- 서버: AWS-EC2(Amazon Linux 2)
- 데이터베이스: AWS-RDS(PostgreSQL), Redis
- CI/CD: Jenkins(2.440.2 버전)

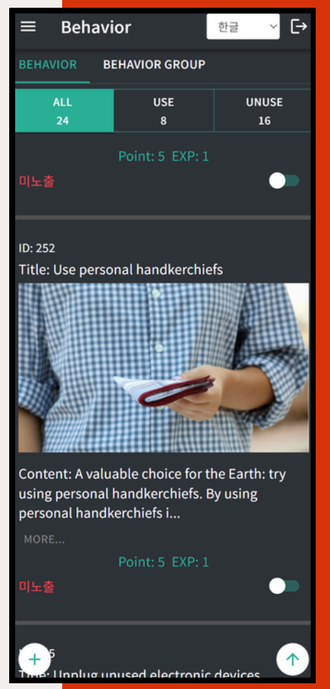
카본제로 – 캄보디아

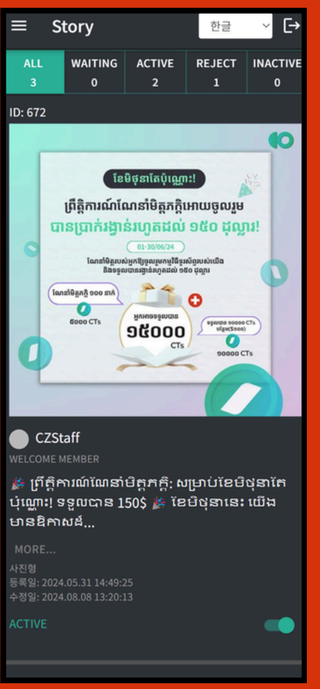
기간 (2024.01.31 ~ 2024.08.10) | 역할 (풀스택: 앱 API, 관리자 페이지, 서버/DB 구축)











Tech Stack

- FrontEnd: React(TypeScript), MUI, styled-components, Redux Toolkit, Redux Saga, Persist, i18next, moment, Axios
- BackEnd: Spring Boot 2.5.6, JPA/QueryDSL, MyBatis, JWT, Swagger
- App: Flutter
- Database: AWS RDS (PostgreSQL), Redis
- Infra: AWS EC2(Amazon Linux 2), Nginx, Route53, ACM(SSL), ALB, Jenkins, S3(+Lambda)

Architecture

- Route53 → ALB(ACM TLS) → Nginx/EC2 → App(API) · RDS(PostgreSQL) · Redis · S3(+Lambda)

핵심 요구사항

- 앱과 관리자 페이지를 분리 운영해야 했다.
- 글로벌 사용자를 위해 다국어(i18n)를 지원해야 했다.
- 대규모 데이터 테이블과 무한 스크롤 UX를 안정적으로 제공해야 했다.
- 포인트 정책·랭킹 모듈을 확장 가능한 구조로 개발해야 했다.
- AWS 기반 인프라에서 무중단 운영 및 CI/CD 체계를 갖춰야 했다.

핵심 개발사항

- 다국어 표준화: react-i18next + JSON 번역팩, LocalStorage 기반 사용자 언어 고정.
- 모바일 운영 모드: 로그인만 반응형, 나머지는 모바일 전용 화면 별도 개발.
- 대용량 테이블: 서버사이드 정렬·페이지네이션 전환으로 성능 안정화.
- 무한 스크롤 개선: 프리패치(pre-fetch) + TOP 버튼 제공.
- 포인트 정책 리팩토링: 복잡한 보상 로직을 구조화해 유지보수성 확보.
- 랭킹 모듈 구현: MyBatis + CTE(WITH) 기반, 개인/소속별 주간·누적 랭킹 지원.
- 보안/세션: JWT 기반 Access/Refresh 토큰 전략 적용.
- 커뮤니티 신고·숨김 기능: 출시 블로커를 단기 내 해결.
- 인프라 설계: AWS EC2/Nginx + Route53/ACM/ALB, Redis 세션 관리, S3+Lambda 이미지 리사이징, Jenkins CI/CD.

주요 이슈 & 해결

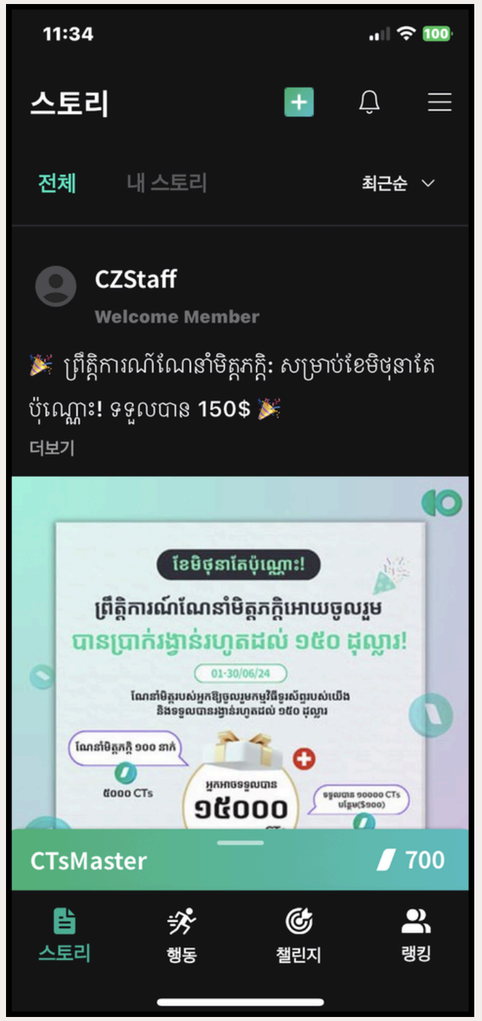
- i18n 언어 고정 문제 → react-i18next 표준화 + LocalStorage 저장으로 해결.
- MUI DataGrid 성능 저하 → 서버사이드 정렬/페이지네이션으로 전환, 성능 안정화.
- 무한 스크롤 지연 → 프리패치 + TOP 버튼 적용, UX 개선.
- 랭킹 계산 정확성 부족 → MyBatis + CTE 쿼리 리팩토링, 정확성 및 유지보수성 강화.
- 보상 악용 취약점 → 당일 인증 여부 검증 추가, 중복 보상 차단.
- 스토어 출시 반려(신고·숨김 부재) → 커뮤니티 신고/숨김 기능 급구현, 출시 완료.

설계

- 캐시/세션: Redis를 Docker로 운영해 JWT·이메일 토큰 저장소로 사용.

성과

- 다국어 환경에서 안정적 운영 구조 확보.
- 대용량 테이블과 무한 스크롤 UX 성능 개선.
- 랭킹·포인트 정책 구조화로 운영 효율성 강화.
- AWS 기반 무중단 배포와 CI/CD 파이프라인 확보.



러닝메이트
첫 게시 안내

함께 달리는 러너들의
기록 공간!

06. 러닝메이트:RunningMate

2025-01-14 ~ 2025-08-12

러닝 기록/인사이트(속도·히트맵·요일 경로)와 페이스·크루·채팅·팔로우, 랭킹을 제공하는 커뮤니티형 러닝 앱.

- 앱: React Native
- 앱 API: Spring-boot 3
- 웹소켓: node.js(socket.io)
- 서버: iwinv(CentOS 9)
- 데이터베이스: CentOS서버 내에 구축(PostgreSQL, PostGis), redis
- CI/CD: 윈도우 로컬 환경에 Jenkins 구축, 앱은 플레이스토어에 배포

Play Store: https://play.google.com/store/apps/details?id=com.runningmateapp

러닝메이트: RunningMate

기간 (2025.01.14 ~ 2025.08.12) | 역할 (개인프로젝트)

개요

- 러닝 기록과 분석(속도·히트맵·요일별 경로), 소셜 기능(게시글·팔로우·채팅·랭킹)을 통합한 커뮤니티형 러닝 앱.
- 실시간 트래킹과 데이터 분석, 커뮤니티 기능을 결합해 러너들의 참여 동기 부여를 강화.

Tech Stack

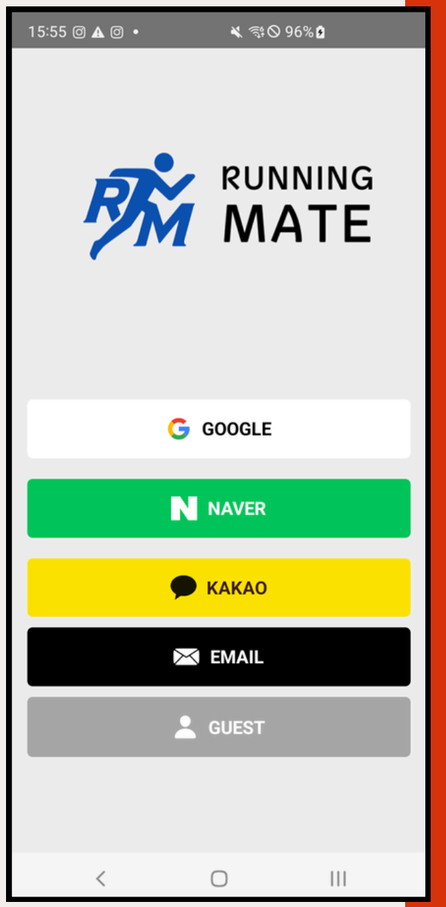
- FrontEnd(App): React Native(0.76.6), TypeScript, React Navigation, React Query, Zustand, FlashList, RN Maps, Notifee, Image Picker/Resizer, ViewShot, Share
- BackEnd(API): Spring Boot 3.4.1, JPA, QueryDSL, MyBatis, PostgreSQL(+PostGIS), Spring Security + JWT, Swagger
- WebSocket: Node.js(Socket.IO)
- Database: PostgreSQL(+PostGIS), Redis
- Infra/DevOps: iwinv(CentOS 9), Jenkins(Windows 로컬), Google Play 배포

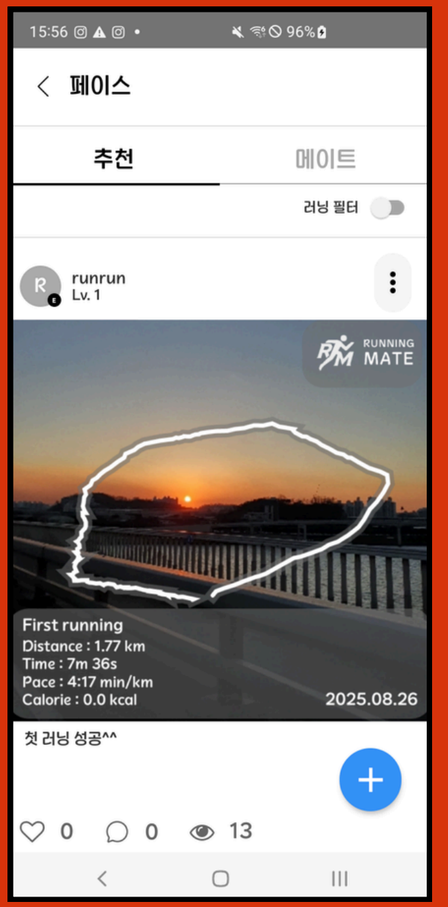
Architecture

- React Native(App) > Spring Boot API > PostgreSQL(+PostGIS) / Redis / Socket.IO(WebSocket)

핵심 기능

- 실시간 트래킹: GPS 데이터를 기반으로 안정적인 경로 기록 및 시각화.
- 분석: 월간 히트맵, 요일별 경로, 평균 속도·페이스 등 데이터 인사이트 제공.
- 소셜 기능: 게시글(페이스), 팔로우, 크루 채팅 지원.
- 랭킹 시스템: 좋아요·댓글·조회·시간을 종합 반영한 점수 기반 순위.
- 공유/광고: 코스 이미지 캡처·SNS 공유, AdMob 배너 및 전면 광고.



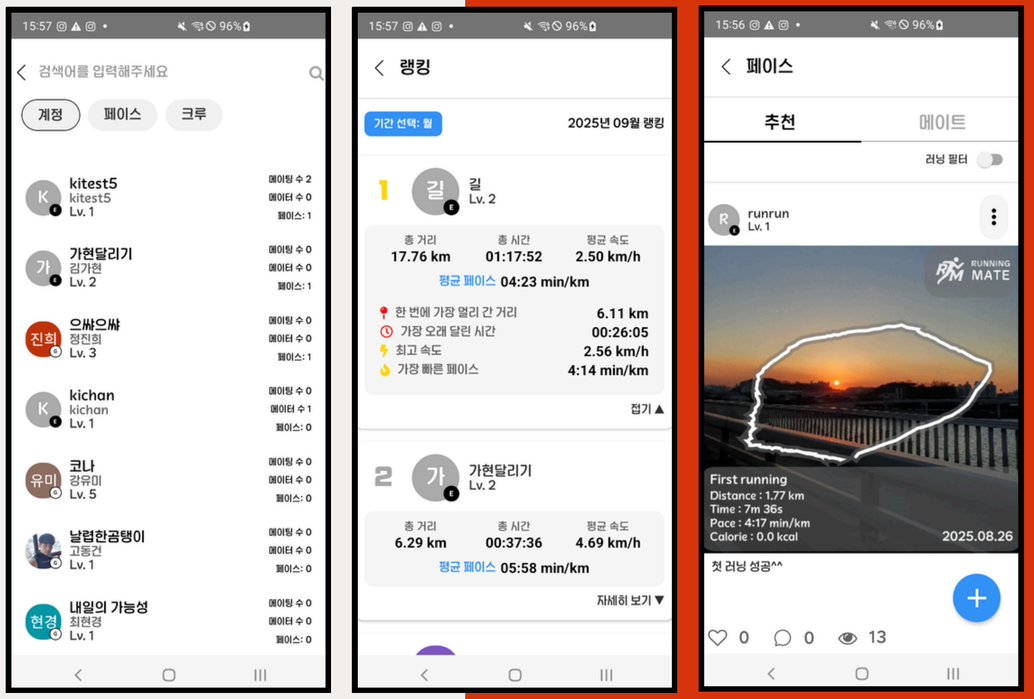


핵심 개발사항

- 지도/트래킹 안정화: RN Maps의 초기 region 흔들림 문제를 공식 문서(onMapReady 가이드) 기반으로 보정. 속도·거리 기반 아웃라이어 필터링으로 좌표 '튐' 현상 제거.
- 백그라운드 기록 유지: Foreground Service + Notifee 알림, 배터리 최적화 예외 설정으로 장시간 러닝 기록 안정화.
- 리스트 최적화: FlatList의 성능 한계를 FlashList로 교체. React Native 공식 문서를 참고해 프리패치 로직 적용 → 대규모 피드도 지연 없이 탐색 가능.
- OAuth 안정화: 네이버 로그인 RN 0.76.6 환경에서 크래시 발생 → GitHub 이슈 분석 후 patch-package를 적용해 초기화 타이밍을 수정, 안정적 로그인 흐름 확보.
- 채팅 unread 계산 최적화: 채팅방에 latest_seq, 사용자별 last_read_seq를 관리해 두 값을 비교. → 안 읽은 메시지 수와 읽은 인원을 즉시 계산, 조인 최소화.
- 랭킹 성능 개선: Materialized View + 인덱스 튜닝, pg_cron으로 주기적 갱신 → 대규모 환경에서도 일정한 응답 속도 확보.
- 대용량 조회 최적화: Keyset 페이지네이션으로 OFFSET 기반 지연 제거, 10만+ 데이터 기준에서도 <1초 응답 유지.
- 이미지 공유 기능: Polyline을 SVG/뷰로 렌더 → ViewShot 캡처 → 갤러리 저장 및 인스타 스토리 공유 기능 구현.

주요 이슈 & 해결

- 초기 지도 좌표 흔들림 → RN Maps 공식 문서(onMapReady 가이드) 참고 후 region 보정 → 첫 렌더 안정화.
- 좌표 '튐' 현상 → 속도·거리 기반 아웃라이어 필터 적용 → 경로 단절 최소화.
- 무한 스크롤 지연 → FlashList + 프리패치 적용(Shopify 가이드 참고) → 스크롤 UX 개선.
- 네이버 OAuth 크래시 → GitHub 이슈 분석 후 patch-package로 초기화 타이밍 수정 → 앱 정상화.
- 채팅 unread 계산 부하 → last_read_seq vs latest_seq 비교 방식으로 단순화 → DB 부하 최소화.

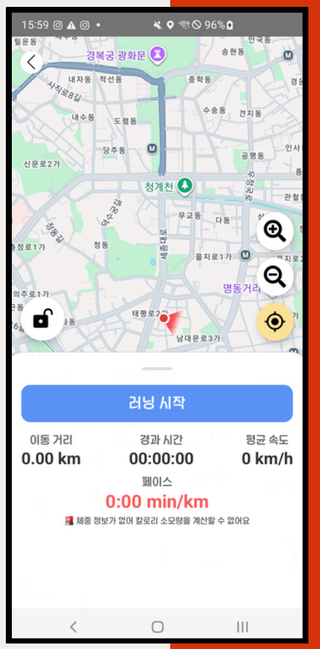


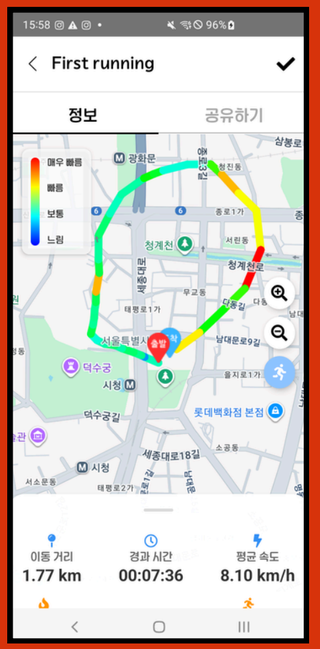
설계

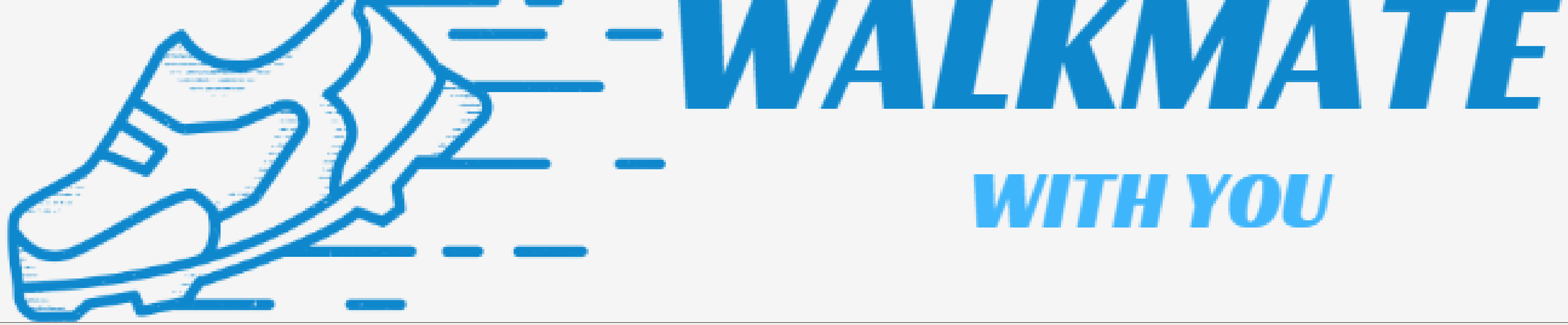
- 실시간 통신 선택: Spring Boot STOMP는 RN 환경에서 호환성과 유지보수 부담이 커서 제외. Socket.IO를 선택해 모바일 환경에서 재연결·네임스페이스 관리가 단순하고 안정적으로 동작.
- 채팅 seq 구조: 채팅방 단위로 latest_seq를 기록, 사용자별 last_read_seq 저장. 두 값을 비교해 unread와 읽은 인원을 효율적으로 계산.
- 보안/세션 관리: Redis 기반 JWT 세션 관리로 중복 로그인 차단, 블랙리스트 전략으로 만료·재사용 제어.
- 랭킹 구조: Materialized View 기반 캐싱으로 실시간성과 성능을 동시에 확보.

성과

- 지도·트래킹 안정화 및 소셜 기능 통합으로 사용자 체감 성능 대폭 개선.
- GitHub 이슈/공식 문서 기반 문제 해결 경험을 통해 기술 역량 검증.
- 채팅 seq 구조로 읽음/안읽음 계산 성능 최적화.
- 10만+ 데이터 환경에서도 <1초 응답 성능 달성.
- 구글 플레이 스토어 정식 배포 완료.







07. Walk Planet

2024-06-01 ~ 2024-08-12

걸음 수를 측정하여 걸음 수를 기록하고 포인트로 전환하여 앱 내의 캐릭터인 WALK MATE를 꾸미는 앱

- 앱: React Native
- 웹뷰: React-typescript
- 앱 API: Spring-boot 2.5.6
- 서버: iwinv(CentOS 9)
- 데이터베이스: CentOS서버 내에 구축(PostgreSQL)
- CI/CD: 윈도우 로컬 환경에 Jenkins 구축, 앱은 플레이스토어에 배포

Web View URL: https://kichani.com/chan-web/login

- Email: Test
- Password: (입력하지 않습니다)

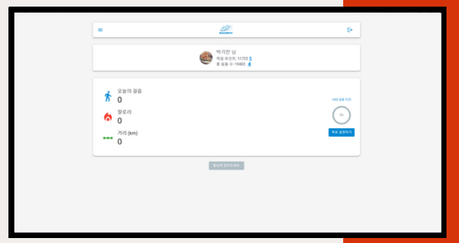
Play Store: https://play.google.com/store/apps/details?id=com.walkplanet

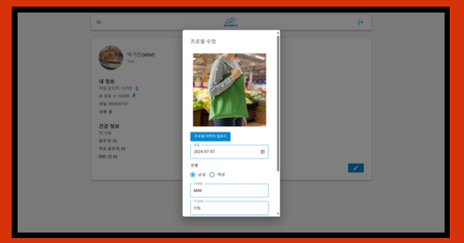
Walk Planet

기간 (2024.06.01 ~ 2024.08.12) | 역할 (개인프로젝트)

개요

- 사용자의 걸음 수를 측정해 포인트로 전환하고, 앱 내 캐릭터(WALK MATE)를 꾸밀 수 있는 러닝/헬스케어 앱.
- 모바일 센서 기반의 걸음 측정과 React Native–WebView 연동 구조를 통해 앱과 웹이 동기화되는 경험을 제공.





Tech Stack

- FrontEnd(Web): React(TypeScript), MUI, React Query, Redux
- App: React Native(WebView), Android Native(SensorManager STEP_DETECTOR, AlarmManager, Foreground Service, Notification)
- BackEnd(API): Spring Boot 2.5.6 (REST API)
- Database: PostgreSQL
- Infra: iwinv(CentOS 9), Nginx, Jenkins(Windows 로컬)

Architecture

- React Native(App: WebView) > React(Web) > Spring Boot API > PostgreSQL

핵심 기능

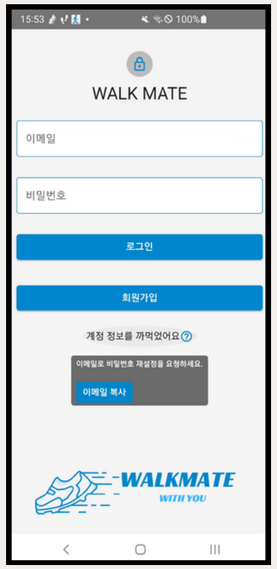
- 걸음 측정: Android STEP_DETECTOR 센서를 기반으로 하루 걸음 수 기록.
- 백그라운드 유지: Foreground Service로 앱 종료 시에도 카운트 지속.
- 데이터 동기화: RN ↔ WebView ↔ Native 간 양방향 통신으로 걸음 수 일관성 유지.
- 목표/누적 관리: StepGoals, StepHistory, StepTotal 도메인 모델을 설계해 하루 목표·이력·누적 데이터를 구분 관리.
- 포인트 전환/꾸미기: 누적 걸음을 포인트로 전환, 캐릭터 꾸미기에 활용.

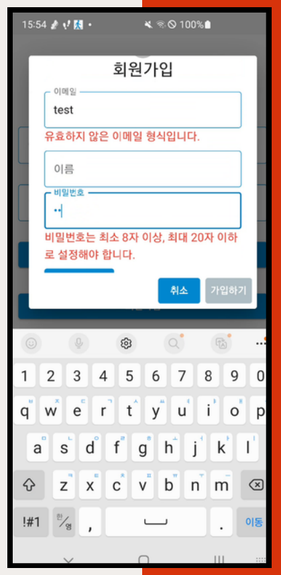
핵심 개발사항

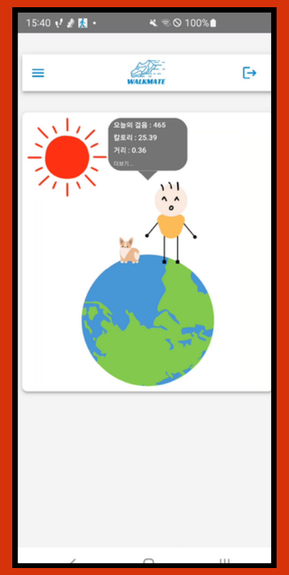
- 센서 선택/튜닝: 가속도계는 민감도 한계로 오차가 커 STEP_DETECTOR로 전환. Foreground Service와의 궁합이 좋아 안정적인 카운팅 확보.
- 자정 초기화 로직: AlarmManager 브로드캐스트로 매일 00:00에 카운터 리셋.
- 데이터 정합성: 로그인 시 DB 값 → RN → Native로 역주입, 로그아웃 시 현재 걸음 수를 DB에 저장해 상태 불일치 방지.
- 이미지 업로드 경량화: 모바일 업로드 실패를 해결하기 위해 blueimp-load-image로 리사이즈/EXIF 보정 후 업로드.
- 웹 데이터 패칭 개선: useState/useEffect 중심 구조를 React Query로 전환해 캐싱/중복요청 방지, 로딩 플리커 개선.
- 배포 자동화: Jenkins + Publish over SSH로 FE 빌드와 BE JAR을 원격 서버에 배포 자동화.

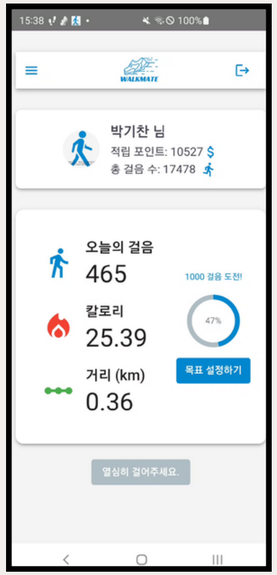
주요 이슈 & 해결

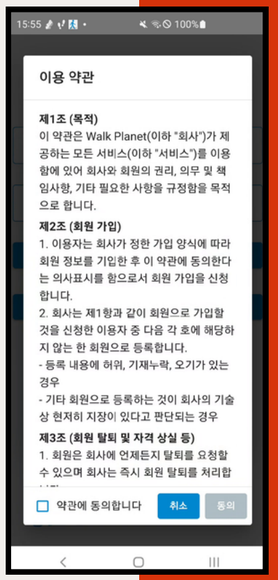
- 센서 정확도 문제 → 가속도계 기반 카운팅 오차 발생 → STEP_DETECTOR로 교체해 정확도 개선.
- 앱 종료 시 카운팅 중단 → Foreground Service와 상시 알림 적용 → 백그라운드에서도 지속 측정 가능.
- 자정 이후 걸음 수 초기화 불가 → AlarmManager 브로드캐스트로 00:00 카운터 리셋 → 매일 정확한 초기화 확보.
- 앱-웹-네이티브 간 값 불일치 → 로그인/로그아웃 시점에 DB ↔ RN ↔ Native 값 동기화 → 데이터 일관성 보장.
- WebView SSL 오류 → 초기 배포 시 중간 인증서 미설치로 WebView 차단 발생 → fullchain 설치로 해결.

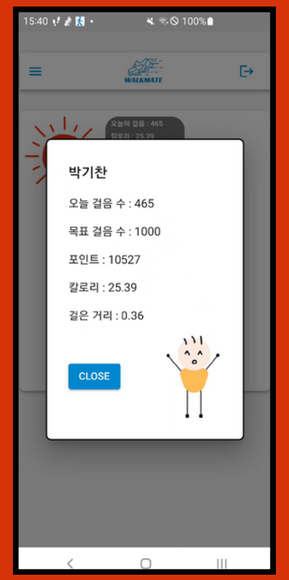










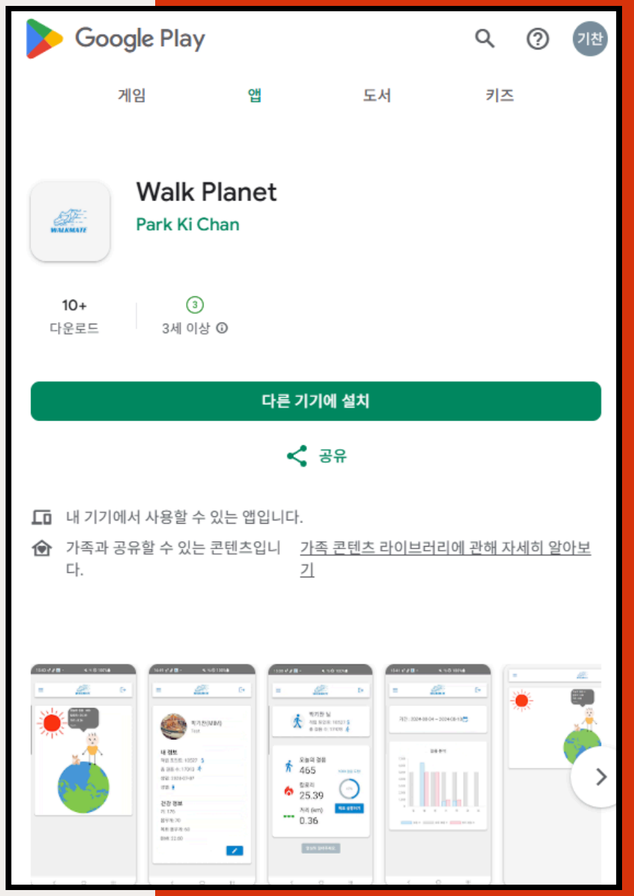


설계

- 걸음 수 구조: StepGoals(목표), StepHistory(일자별 이력), StepTotal(누적)로 분리해 확장성과 데이터 정합성 확보.
- 통신 구조: RN ↔ WebView, RN ↔ Native 양방향 브릿지 통신으로 동기화 오류 최소화.
- 인프라: 비용 최소화를 위해 S3 대신 서버 로컬 uploads/ 경로를 사용, Nginx로 단순 퍼블릭 서빙.

성과

- 모바일 센서 기반 걸음 측정 정확도와 백그라운드 지속성을 확보.
- 하루 목표/이력/누적 데이터 일관성 유지로 사용자 신뢰도 강화.
- React Query 전환으로 데이터 패칭 UX 개선.
- 구글 플레이 스토어 정식 배포 완료.



08. Price Quotation Tool

2025-06-09 ~ 2025-08-26

대용량 엑셀 품목을 초보자도 빠르게 견적서로 만들 수 있게 한 Windows 데스크톱 견적 툴

- 윈도우 앱: Tauri + React + TypeScript + Vite
- 토큰 키 생성기: Python

Price Quotation Tool (Windows, Tauri)

기간 (2025.06.09 ~ 2025.08.26) | 역할 (개인프로젝트)

개요

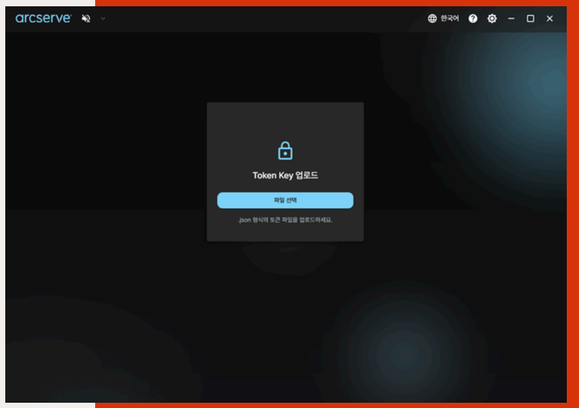
- 대용량 엑셀 품목을 빠르게 견적서로 변환하는 Windows 데스크톱 툴.
- 오프라인/로컬 환경에서 누구나 동일한 품질의 견적서를 산출할 수 있도록 구현.
- Electron 기반 프로토타입을 Tauri로 전환해 설치·배포 용량과 성능을 대폭 개선.

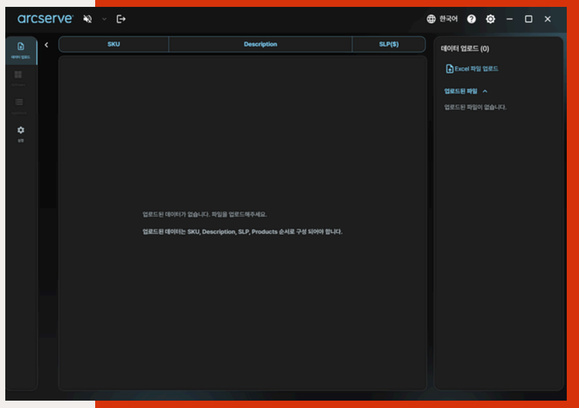
Tech Stack

- Desktop: Tauri v2(Windows), React 19, TypeScript, Vite
- UI/상태: MUI v7, Zustand, Framer Motion, notistack
- Excel I/O: xlsx(읽기), ExcelJS(생성), file-saver(다운로드)
- Virtualization/DND: @tanstack/react-virtual, react-window, @hello-pangea/dnd
- Storage: @tauri-apps/plugin-store
- 보안/토큰: Python(토큰 키 생성기, 암복호화 유틸)
- 기타: React Router, i18next, Tauri opener, use-sound

핵심 기능

- 엑셀 업로드/매핑: 파일 업로드 시 품목/옵션 자동 매핑, 정규식 기반 네이밍 규칙으로 단일/2종 세트 구분.
- 테이블 커스터마이즈: 컬럼 드래그&드롭, On/Off 토글, 사용자 설정 자동 저장·복원.
- 대용량 대응: 수천 행 데이터를 가상 스크롤로 부드럽게 처리.
- 다국어/온보딩: i18next 기반 다국어 지원, 초보자 가이드/도움말 제공.
- 내보내기: ExcelJS로 견적 파일 생성·다운로드, 클라이언트 양식 규칙 반영.
- 사용성 강화: 라이트/다크 모드, 사운드 피드백, 커스텀 윈도우 버튼, 스플래시 화면.





핵심 개발사항

- Electron → Tauri 전환: 런타임 경량화(100MB+ → ~6MB), 설치·업데이트 속도 개선.
- 리스트/테이블 최적화: react-virtual, react-window 가상화와 메모이제이션 적용.
- 정규식 매핑 로직: 품목 네이밍 규칙을 정규식으로 정의해 자동 매핑 정확도 향상.
- 스토리지 관리: Tauri Store로 사용자별 컬럼·테마·가이드 진행도 영속화.
- 보안 처리: 모든 데이터 로컬 처리, Python 토큰 키 발급기로 암호화/복호화 검증.

주요 이슈 & 해결

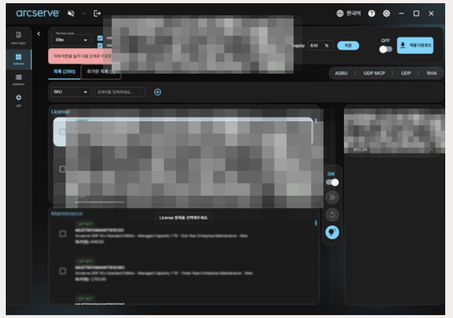
- Electron 빌드 용량 과대(100MB+) → Tauri 전환으로 ~6MB 확보, 배포 부담 해소.
- 대용량 테이블 버벅임 → react-virtual/메모이제이션 적용 → 수천 행도 원활히 렌더링.
- 사용자 입력 오류 → 정규식 기반 자동 매핑으로 실수 최소화, 견적 흐름 단순화.
- 스토리지 불안정 → Tauri Store 도입으로 사용자 설정 안정적 보관.

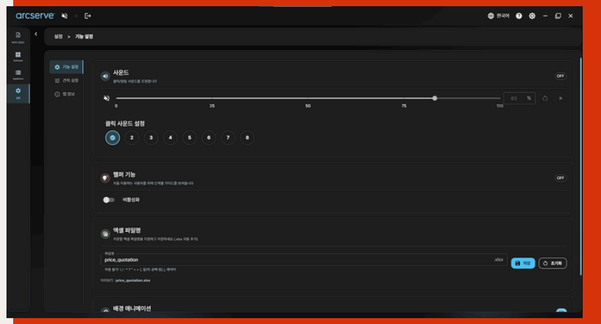
설계

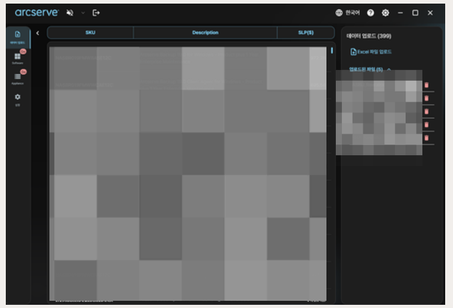
- 런타임/플랫폼 전환:
  - Electron은 Node+Chromium 번들로 용량·메모리 사용량이 과다(100MB+).
  - Tauri는 OS WebView + Rust 백엔드를 활용해 패키지 용량 ~6MB, 낮은 메모리 사용, 빠른 구동 속도를 확보.
- 보안: 민감 데이터는 네트워크 전송 없이 로컬에서만 처리. Python 기반 토큰 키 발급기와 엡 내 암복호화 로직으로 검증.
- 로컬 우선 아키텍처: Excel 업로드/생성, 사용자 설정 저장까지 모든 기능을 로컬에서 수행, 오프라인 환경에서도 완전 동작.
- UI 전략: 테이블 + 드래그&드롭 + 다크 모드, 초보자도 직관적으로 사용 가능.

성과

- 설치·배포 용량을 Electron 대비 90% 이상 절감(100MB+ → ~6MB).
- 견적 작성 속도와 정확도를 개선해 업무 효율성 강화.
- 초보자도 몇 번의 클릭으로 견적을 완성할 수 있는 사용성 확보.







09. 이별 일기

2024-08-13 ~ 2024-08-19

이별한 사람들을 위한 일기 웹사이트 입니다. 하루에 한번 이별 일기와 이별 지수를 기록할 수 있습니다.

- 웹뷰: React-typescript
- 앱 API: Spring-boot 2.5.6
- 서버: iwinv(CentOS 9)
- 데이터베이스: CentOS서버 내에 구축(PostgreSQL)
- CI/CD: 윈도우 로컬 환경에 Jenkins 구축

이별 일기 사이트

- URL: https://kichani.com/leave/login

이별일기 (Leave Diary)

기간 (2024.08.13 ~ 2024.08.19) | 역할 (개인프로젝트)

개요

- 이별한 사람들을 위한 감정 기록 웹서비스.
- 하루 한 번 이별 일기를 작성하고, 이별 지수와 디데이를 기록·확인할 수 있도록 구현.

Tech Stack

- FrontEnd: React(TypeScript), React Query, Redux Store, MUI
- BackEnd: Spring Boot, JPA, JWT
- Database: PostgreSQL(iwinv 서버)
- Infra: Nginx, Jenkins(로컬), systemd 서비스

Architecture

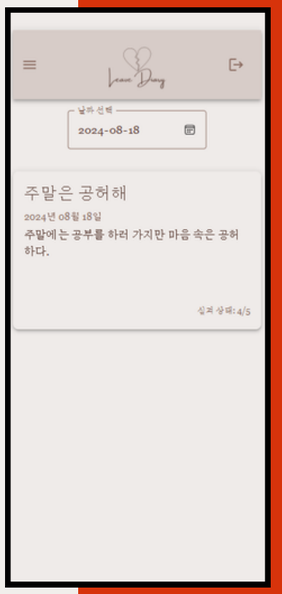
- React(Frontend) > Spring Boot API > PostgreSQL

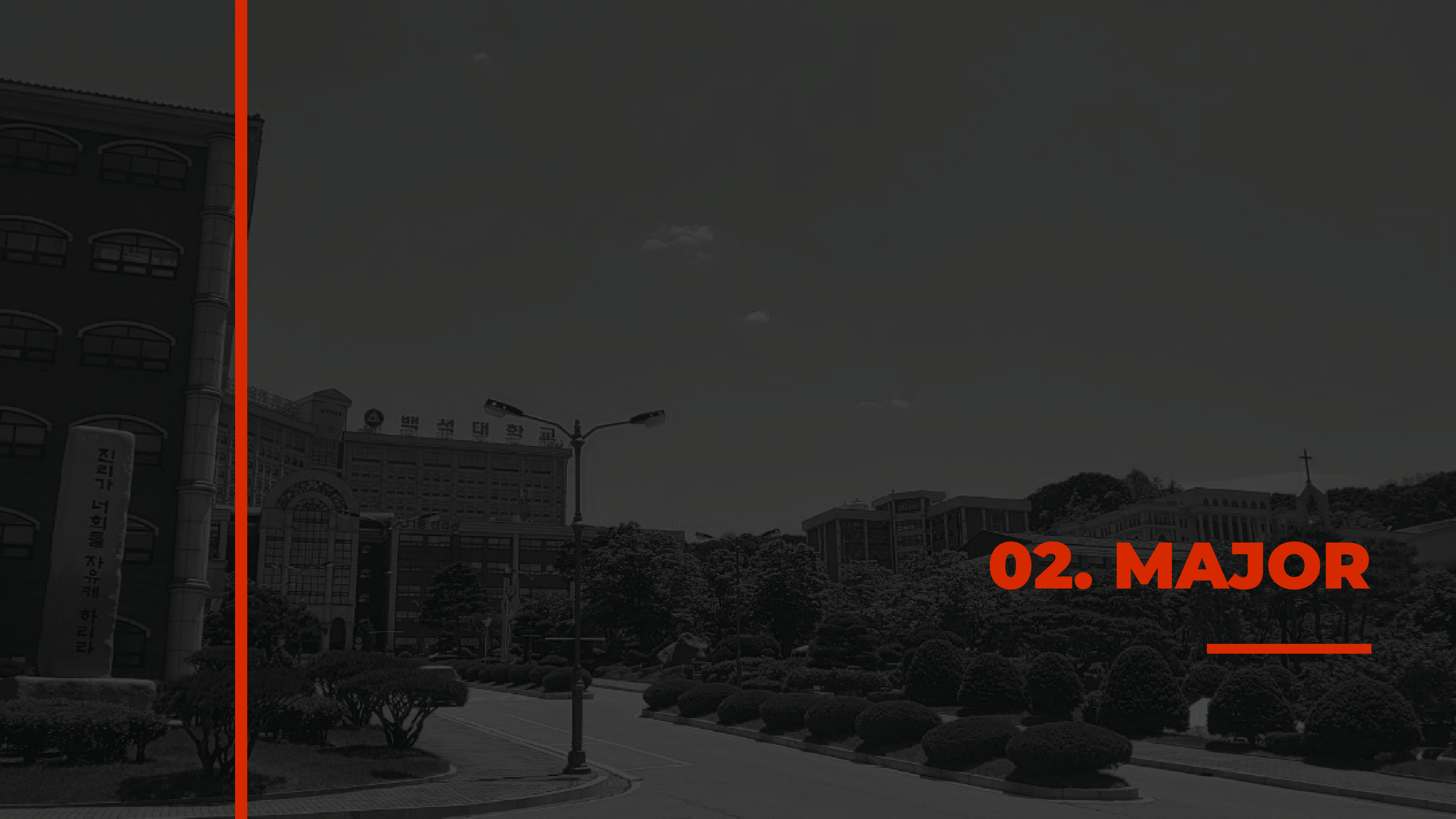
핵심 기능

- 하루 1회 작성 제한 정책(당일 중복 작성 차단).
- 무한 스크롤(useInfiniteQuery + useInView).
- 정렬 옵션(최신순, 오래된순).
- 이별 디데이 계산 및 프로필 관리.

핵심 개발사항

- API 레벨에서 하루 1회 작성 정책 검증 로직 추가.
- 무한 스크롤 데이터 로딩 최적화(React Query 캐시 + 프리패치).
- Jenkins + SSH 배포 자동화, systemd로 서버 자동 실행 구성.





02. MAJOR

정보처리기사

관리번호: 20250912090 6-08-814859

국 가 기 술 자 격 증

자격번호: 25202070630D

자격종목: 정보처리기사

성 명: 박기찬

생년월일: 1999년 08월 03일

위 사람은 「국가기술자격법」에 따른 국가기술자격을 취득하였음을 증명합니다.

합격 연월일: 2025년 09월 12일

발급 연월일: 2025년 09월 12일

과학기술정보통신부

※ 본 국가기술자격증은 「국가기술자격법」 제23조에 따라 과학기술정보통신부장관의 위탁을 받은 한국산업인력공단 이사장이 확인 · 발급함.

한국산업인력공단 이사장

◆ 본 확인서는 인터넷으로 발급되었으며 원본의 진위여부는 반드시 한국산업인력공단(http://www.Q-Net.or.kr)에서 조회,확인하시기 바랍니다. 문서의 진위 여부는 발급 신청일로부터 90일간 확인 가능합니다.

자격증 정보:

- 자격증 명칭: 정보처리기사 자격증
- 취득 일자: 2025-09-12
- 발급 기관: 한국산업인력공단

정보처리산업기사

관리번호: 20220529171 8-08-068864

국 가 기 술 자 격 증

· 자격번호: 21213020378W

· 자격종목: 정보처리산업기사

· 성 명: 박기찬

· 생년월일: 1999년 08월 03일

위 사람은 「국가기술자격법」에 따른 국가기술자격을 취득하였음을 증명합니다.

· 합격 연월일: 2021년 11월 26일

· 발급 연월일: 2022년 05월 29일

과학기술정보통신부

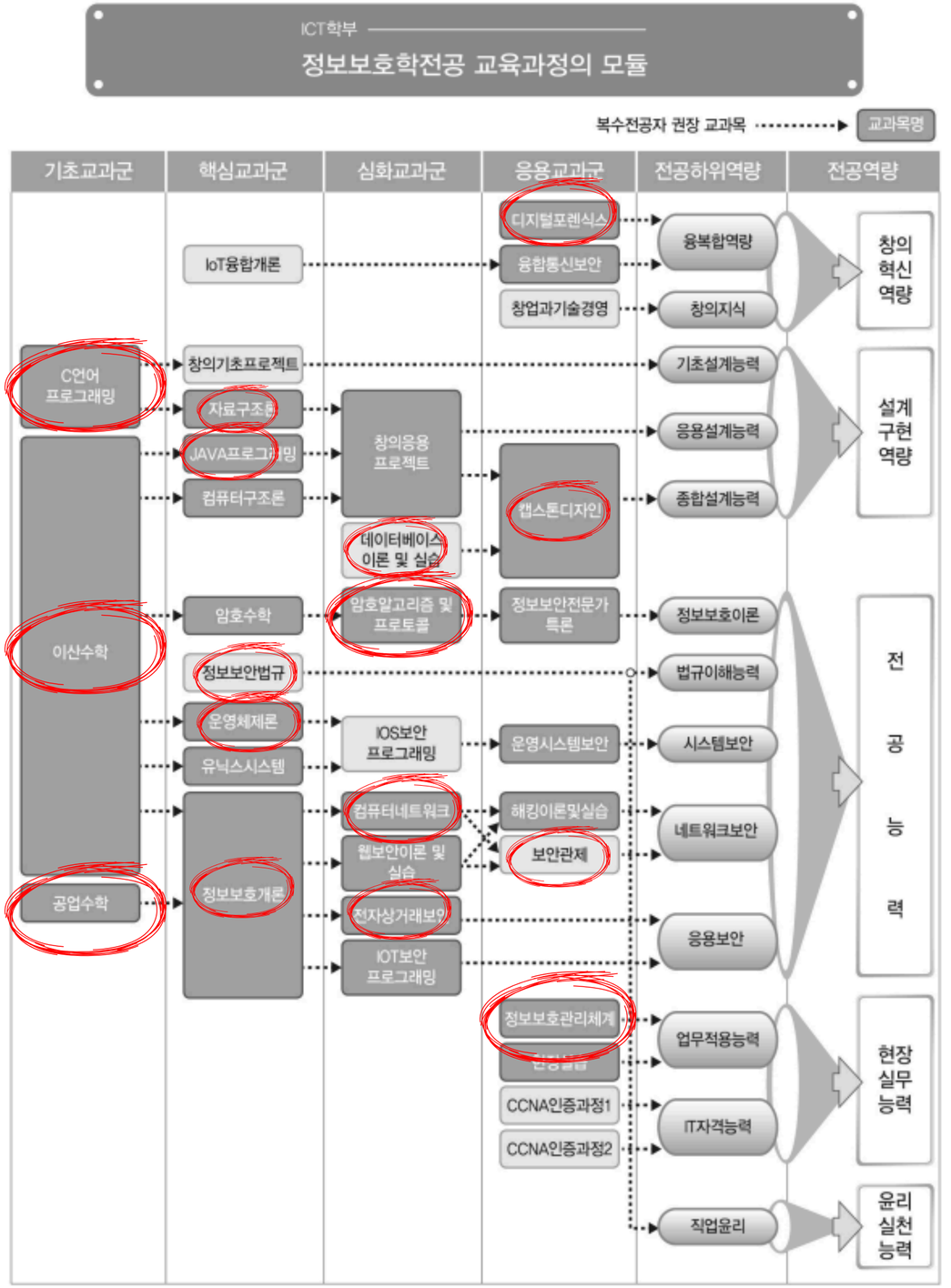
※ 본 국가기술자격증은 「국가기술자격법」 제23조에 따라 과학기술정보통신부장관의 위탁을 받은 한국산업인력공단 이사장이 확인 · 발급함.

한국산업인력공단 이사장

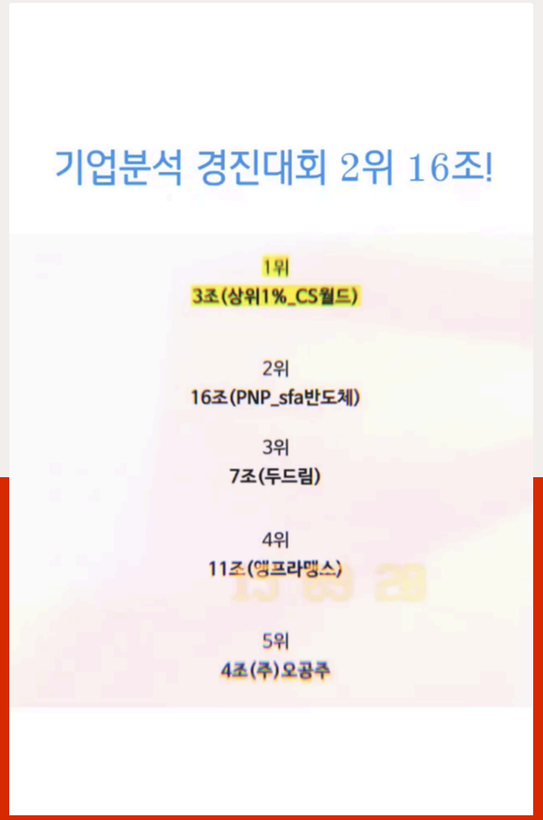
자격증 정보:

- 자격증 명칭: 정보처리산업기사 자격증
- 취득 일자: 2021-11-26
- 발급 기관: 한국산업인력공단

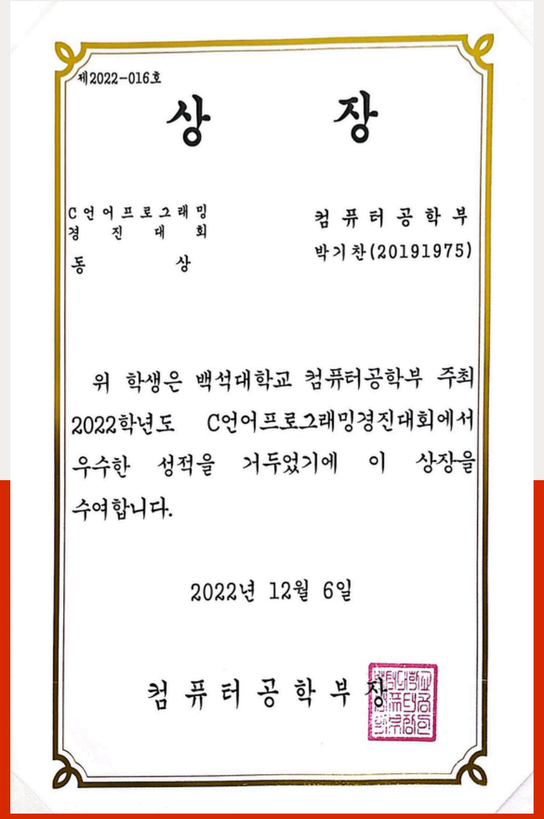
전공 커리큘럼
이수 내역


ICT학부
정보보호학전공 교육과정의 모듈
복수전공자 권장 교과목
교과목명
기초교과군
핵심교과군
심화교과군
응용교과군
전공하위역량
전공역량
C언어 프로그래밍
이산수학
공업수학
IoT융합개론
창의기초프로젝트
자료구조론
JAVA프로그래밍
컴퓨터구조론
암호수학
정보보안법규
운영체제론
유닉스시스템
정보보호개론
창의응용 프로젝트
데이터베이스 이론 및 실습
암호알고리즘 및 프로토콜
IOS보안 프로그래밍
컴퓨터네트워크
웹보안이론 및 실습
전자상거래보안
IOT보안 프로그래밍
디지털포렌식스
융합통신보안
창업과기술경영
캡스톤디자인
정보보안전문가 특론
운영시스템보안
해킹이론및실습
보안관제
정보보호관리체계
CCNA인증과정1
CCNA인증과정2
융복합역량
창의지식
기초설계능력
응용설계능력
종합설계능력
정보보호이론
법규이해능력
시스템보안
네트워크보안
응용보안
업무적용능력
IT자격능력
직업윤리
창의 혁신 역량
설계 구현 역량
전 공 능 력
현장 실무 능력
윤리 실천 능력

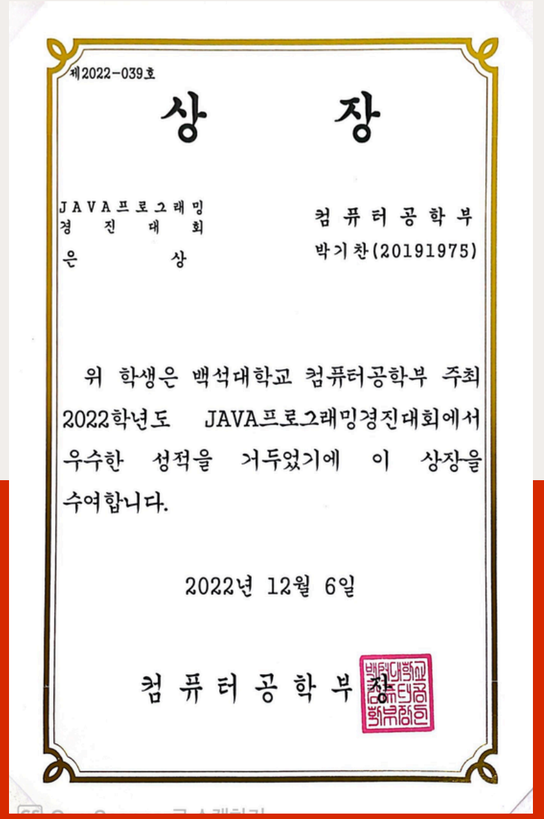
교내
경진대회



기업분석 경진대회
2등



C언어 프로그래밍
경진대회 동상



JAVA 프로그래밍
경진대회 은상

WORD IT SHOW

WORLD IT SHOW는 IT에 관련된 여러 기업들이 참여하는 전시회입니다.
분야별로 관람할 수 있게 여러 부스들이 있었습니다.

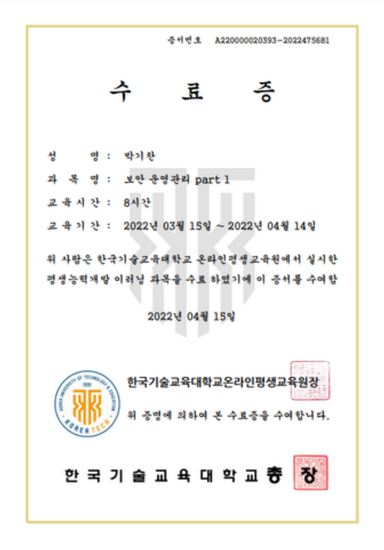
소프트웨어는 물론이고 하드웨어 부분도 볼 수 있었습니다.

크게 ICTConvergence, Smart Living & Health Care, Robotics, Digital Twin & Metaverse, Block Chain & Security,
IntelligentMobility의 종류가 전시돼 있었습니다.

개인적으로 AI분야의 전시품목이 많은걸 느꼈습니다.

전시회를 관람하고 AI가 지금의 트렌드라는 것을 느낄 수 있었습니다.

한국기술교육대학교


증서번호 A220000020393-2022475681

수 료 증

성 명 : 박기찬
과 목 명 : 보안 운영관리 part 1
교 육 시 간 : 8시간
교 육 기 간 : 2022년 03월 15일 ~ 2022년 04월 14일

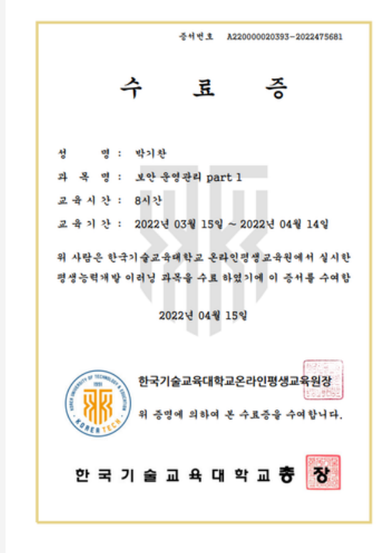
위 사람은 한국기술교육대학교 온라인평생교육원에서 실시한 평생능력개발 이러닝 과목을 수료 하였기에 이 증서를 수여함

2022년 04월 15일

한국기술교육대학교온라인평생교육원장

위 증명에 의하여 본 수료증을 수여합니다.

한 국 기 술 교 육 대 학 교 총 장


증서번호 A220000020393-2022475681

수 료 증

성 명 : 박기찬
과 목 명 : 보안 운영관리 part 1
교 육 시 간 : 8시간
교 육 기 간 : 2022년 03월 15일 ~ 2022년 04월 14일

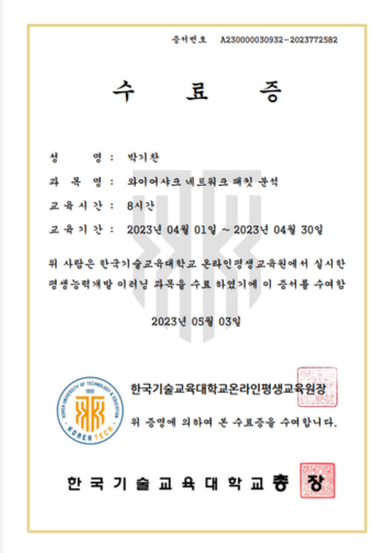
위 사람은 한국기술교육대학교 온라인평생교육원에서 실시한 평생능력개발 이러닝 과목을 수료 하였기에 이 증서를 수여함

2022년 04월 15일

한국기술교육대학교온라인평생교육원장

위 증명에 의하여 본 수료증을 수여합니다.

한 국 기 술 교 육 대 학 교 총 장


증서번호 A230000030932-2023772582

수 료 증

성 명 : 박기찬
과 목 명 : 와이어샤크 네트워크 패킷 분석
교 육 시 간 : 8시간
교 육 기 간 : 2023년 04월 01일 ~ 2023년 04월 30일

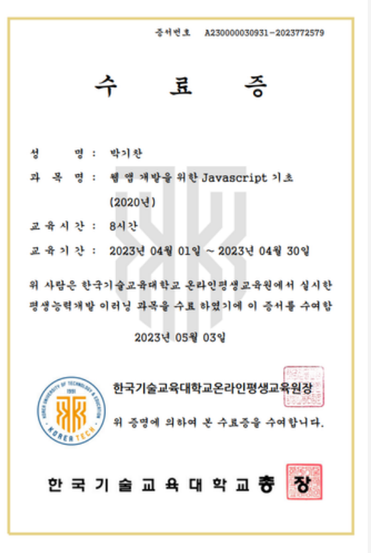
위 사람은 한국기술교육대학교 온라인평생교육원에서 실시한 평생능력개발 이러닝 과목을 수료 하였기에 이 증서를 수여함

2023년 05월 03일

한국기술교육대학교온라인평생교육원장

위 증명에 의하여 본 수료증을 수여합니다.

한 국 기 술 교 육 대 학 교 총 장


증서번호 A230000030931-2023772579

수 료 증

성 명 : 박기찬
과 목 명 : 웹 앱 개발을 위한 Javascript 기초 (2020년)
교 육 시 간 : 8시간
교 육 기 간 : 2023년 04월 01일 ~ 2023년 04월 30일

위 사람은 한국기술교육대학교 온라인평생교육원에서 실시한 평생능력개발 이러닝 과목을 수료 하였기에 이 증서를 수여함

2023년 05월 03일

한국기술교육대학교온라인평생교육원장

위 증명에 의하여 본 수료증을 수여합니다.

한 국 기 술 교 육 대 학 교 총 장

한국 기술 교육 대학교에서 제공하는 평생능력 개발 이러닝 과목으로
보안 운영관리 part1, 보안 운영관리 part2, 와이어샤크 네트워크 패킷 분석, 웹 앱 개발을 위한 Javascript 기초 를 이수했습니다.

03. ADDITIONAL

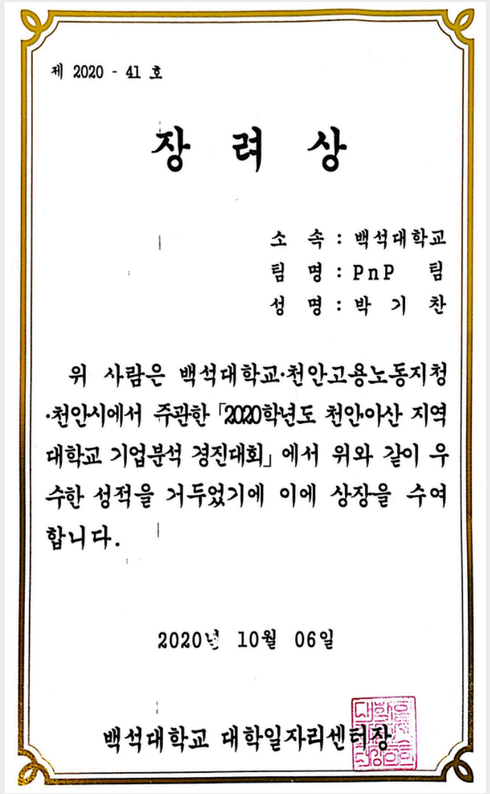
천안 아산지역 대학교경진대회

대학교 별로 2팀씩 선발하여 참가할 수 있는 천안 아산지역 기업분석 경진대회에 참가하여 장려상을 받았습니다.

SFA반도체 기업을 분석하였습니다. 기업 개요, 기업 분석, 취업 전략을 분석했습니다.

취업 전략으로 직접 모의 입사 지원서를 작성했습니다.

2학년 때에 입사지원서를 작성해 봄으로써 미리 취업 전략을 세울 수 있는 경험이었습니다.



제 2020 - 41 호

장 려 상

소 속 : 백석대학교
팀 명 : PnP 팀
성 명 : 박 기 찬

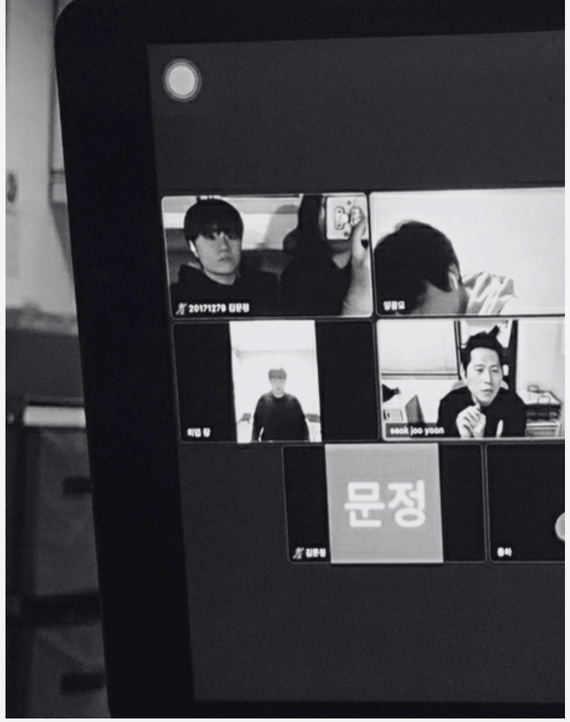
위 사람은 백석대학교·천안고용노동지청·천안시에서 주관한 「2020학년도 천안아산 지역대학교 기업분석 경진대회」에서 위와 같이 우수한 성적을 거두었기에 이에 상장을 수여합니다.

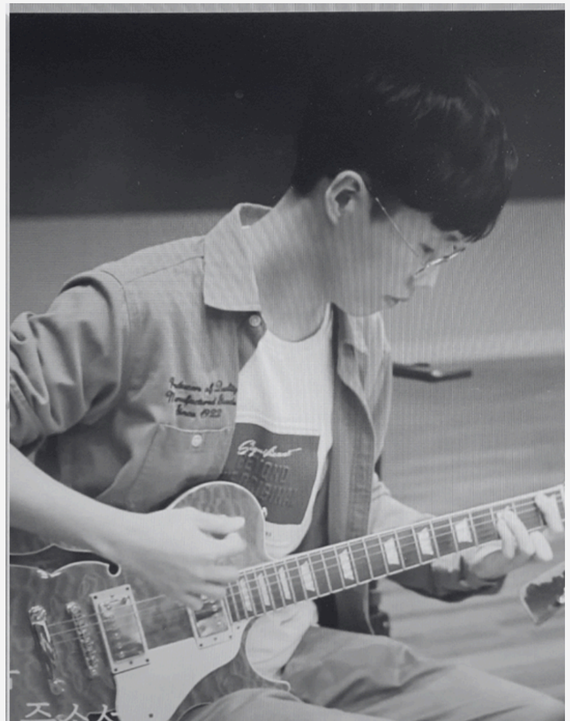
2020년 10월 06일

백석대학교 대학일자리센터장

기업 분석 경진대회 장려상

ICPC 동아리





ICT학부의 기독교 동아리입니다.
위 사진은 코로나로 인해 비대면으로 동아리 모임을 진행하는 사진입니다.
또 다른 사진은 제가 채플 찬양팀에 속했을 때 기타를 치는 사진입니다.

IT Developer

박 기 찬

rlgusrlcks00@gmail.com

010-4855-9646

https://blog.naver.com/qkrrlcks_99

https://github.com/rlgusrlcks00

IT Developer

THANK YOU