손호영

Backend Engineer

010-8586-4134 | tls1234568@naver.com | github.com/pyrimidine02 | portfolio.noraneko.cc

안녕하세요. **6,375명** 사용자와 일 평균 **1,400+ DAU**를 기록하는 프로덕션 서비스를 설계·운영하고 있는 백엔드 개발자 손호영입니다.

알고리즘 문제풀이에서 길러온 “어떤 입력이 들어와도 깨지지 않는 로직”에 대한 고민이 자연스럽게 서비스 아키텍처와 데이터 모델링으로 확장되었고, 지금은 일상의 불편함을 “백엔드 구조로 어떻게 풀 수 있을까”라는 관점에서 바라보고 있습니다.

RailNetwork

서울 지하철 실시간 운행 정보를 다루는 **RailNetwork**는 이 관점이 처음으로 실서비스 코드에 적용된 경험입니다. 서울교통공사 공식 앱을 Proxyman으로 리버스 엔지니어링해 공개되지 않은 SMSS API 엔드포인트를 복원하고, 문서 없이 응답 스키마와 제약을 직접 정의했습니다. 열차 상태가 숫자 코드로만 내려오는 환경에서 이를 “역 도착/출발/지연/운행 종료” 도메인 상태로 재해석하고, 시간표와 결합해 지연 시간을 계산하는 알고리즘을 설계했습니다.

SMSS 장애 시 열린데이터 API로 자동 전환하는 이중 폴백 시스템을 구축해 서비스 안정성을 확보했고, Promise.all 병렬 처리로 전체 노선을 동시 업데이트하도록 개선했습니다. Redis 캐싱으로 응답 성능을 최적화하고, Firebase Analytics 지역 분석(서울 68%, 경기 22%)을 통해 데이터 기반 의사결정에 활용하는 경험도 쌓았습니다. 이 과정에서 체득한 “API가 바뀌더라도 도메인 모델은 안정적으로 유지되도록” 설계하는 감각이 이후 아키텍처·DB·캐싱을 고민할 때 기준점이 되었습니다.

Girls Band Tabi

기획부터 설계·구현까지 책임지고 있는 **Girls Band Tabi** (서버 운영 중, 앱 내부 테스트)에서는 “미래에 생길 문제를 설계 단계에서 선제적으로 막는다”는 기준으로 아키텍처를 잡았습니다. 여러 애니·밴드 프로젝트를 한 백엔드에서 통합 관리하면서 데이터가 계속 쌓이는 구조이므로, Kotlin + Spring Boot 3 + Spring Modulith 기반 이벤트 드리븐 아키텍처로 5개 도메인을 느슨한 결합으로 분리했

습니다.

데이터베이스 설계에서는 “데이터를 어떻게 조회할 것인가”를 먼저 상정한 뒤 스키마를 잡았습니다. 프로젝트·유닛·장소·이벤트를 명확한 엔티티로 두고, 조회 패턴이 필요한 곳에서만 관계를 테이블로 풀었습니다. PostGIS Geography 타입으로 구면 좌표 연산을 처리하면서, 활성 장소만 인덱싱하는 부분 인덱스로 검색 비용을 조절했습니다. Flyway 마이그레이션으로 스키마 변경을 버전 관리하고, HikariCP 커넥션 풀을 최적화해 DB 운영 안정성도 확보했습니다.

캐싱은 Caffeine L1 / Redis L2 이중 캐시를 cache-aside 패턴으로 조합하고, 도메인 단위 캐시 키 네임스페이스로 정밀 무효화를 설계했습니다. Resilience4j 서킷 브레이커로 장애 전파를 차단하고, Cloudflare R2에는 일일 대역폭/월간 요청 제한을 두어 인프라 비용을 통제하고 있습니다. 코드 품질 측면에서는 Detekt/Ktlint 정적 분석과 Kover 커버리지(서비스 계층 **80%** 목표)를 CI 파이프라인에 통합해 품질 게이트를 자동화했습니다.

결론

Girls Band Tabi는 아직 실 트래픽 기반의 성능 튜닝이 이뤄진 단계는 아닙니다. 그러나 RailNetwork에서 **6,375명** 사용자의 실 트래픽 경험을 쌓았고, Girls Band Tabi에서는 그 경험을 바탕으로 아키텍처·DB·캐싱·비용 보호·코드 품질까지 선제적으로 설계하고 있습니다.

실 트래픽에서 문제를 발견하고 해결하는 힘과, 문제가 터지기 전에 구조로 막는 설계력—이 두 가지를 갖춘 백엔드 개발자로 귀사에서 성장하겠습니다. 감사합니다.

손호영 드림