

강희원 (Kang Heewon)

Software Engineer

유저가 겪는 문제를 해결하는 데서 멈추지 않고, 왜 반복되는지 추적해 더 나은 구조로 바꾸는 일을 좋아합니다.
문제를 온전히 해결하기 위해서라면 제품과 시스템의 경계를 가리지 않고 일합니다.

LOCATION	EMAIL	WEBSITE
Seoul, Korea	me@heewon.dev	https://heewon.dev
SKILLS React / Next.js Java / Spring GraphQL OpenTelemetry AWS / Kubernetes		

EXPERIENCE (총 6년 7개월)

CLASS101

2021.03 – Present (5년 7개월)

Auth & Activation · Software Engineer

2025.03 – Present

가입-인증 퍼널 개선

- 휴대폰 인증 실패를 문자 발송, 인증 방식, 발신 번호, Bot Detection 단계로 나눠 원인을 추적하고 문자 발송 Provider 다변화, MO 인증, 발신 번호 확대, Native CAPTCHA를 도입했습니다.
- 단일 Provider와 Web CAPTCHA에 집중돼 있던 실패 지점을 분산해 휴대폰 인증 실패율을 15%대에서 3%대로 낮췄습니다.
- 소셜 로그인 신규 가입자 중 광고 수신 미동의자에게 동의 화면을 한 번 더 노출하는 실험을 제안·주도해, 신규 가입자의 광고 수신 동의율을 8%p 개선했습니다.

브라우저 메인 스텔드 병목 개선

- 공용 디자인 시스템 Vibrant의 런타임 스타일 연산을 빌드 타임 CSS로 옮기는 정적 컴파일러를 설계·구현하고, 안전성을 증명할 수 없는 경로는 기존 런타임으로 fallback하도록 구성했습니다.
- 컴파일 전후 결과를 전수 대조하는 differential test와 browser parity-시각 회귀 테스트를 CI gate로 구축해 기존 동작을 보존하면서, 동일 코드의 compiler OFF/ON 비교에서 홈 스크립트 실행 시간을 16% 단축하고 런타임에서 스타일을 계산하는 요소를 3,752개에서 175개로 약 95% 줄였습니다.
- 프로덕션 CPU 프로파일링으로 상품 카드에서 사용되지 않는 이미지 팔레트 추출이 메인 스텔드를 점유하는 병목을 발견하고, 색상값을 사용하지 않는 경로에서는 연산 자체를 수행하지 않도록 변경해 홈 초기 로드의 팔레트 연산 18회와 추가 이미지 요청 18회를 제거했습니다.

Web-React Native-WebView를 연결하는 Client Journey 설계

- 재현이 어려운 클라이언트 문제를 `Session → Navigation → Meaningful Action → Network / Error → backend trace` 로 연결해 추적하는 Structured Client Journey를 설계했습니다.
- 모든 interaction을 수집하지 않고 Navigation, Meaningful Action, first-party Network, Error로 signal 범위를 제한하고, Web-React Native-WebView의 차이는 platform별 instrumentation으로 분리했습니다.
- 관측성 실패가 사용자 경로로 전파되지 않도록 fail-open을 원칙으로 두고, storage/cardinality budget을 초기 설계 제약으로 정의했습니다.

서비스마다 달랐던 인증 방식을 Session Cookie + 공통 SDK 기반으로 정리

- Cookie, Firebase Token, `x-user-id` 등 서비스마다 달랐던 인증 경로를 Session Cookie + 공통 Authentication SDK 기반으로 정리하고 서비스별 인증 처리를 공통화했습니다.
- Gateway가 전달한 사용자 ID를 그대로 신뢰하던 경로를 줄이고, 각 서비스가 공통 SDK를 통해 auth-service가 관리하는 실제 세션 상태를 일관된 방식으로 검증하도록 점진적으로 전환했습니다.
- Firebase Auth와 custom claim에 중복돼 있던 사용자-동의 데이터를 실제 도메인으로 이전해 인증 데이터와 비즈니스 데이터의 책임을 분리했습니다.

user-service 공통 조회 경로 성능 개선

- user-service에 reader/writer datasource routing을 적용하고 Federation User 조회의 중복 SQL을 제거해, production 기준 GraphQL p99 47%, HTTP p99 42%를 단축했습니다.

GraphQL Hive를 CI/CD에 연결해 schema 변경을 배포 전에 검증

- GraphQL Hive Schema Registry를 Federation composition과 schema check/publish 과정에 연결해 backend subgraph의 계약 변경을 CI/CD에서 검증하도록 했습니다.
- Web과 Mobile의 독립적인 배포 주기 때문에 저장소 검색만으로 안전성을 판단하기 어려웠던 deprecated field는 production operation usage가 소진된 뒤 제거하도록 운영 방식을 정리했습니다.
- Federation 운영과 Schema Registry 활용 경험을 GraphQL on Production으로 정리해 Backend Chapter에 공유했습니다.

Distributed tracing 복구 및 Spring Batch tracing 확장

- 일부 span이 유실되거나 호출 관계가 올바르게 연결되지 않던 문제를 application instrumentation → container → OpenTelemetry Collector → Jaeger 경로까지 추적해 trace의 parent-child 관계와 수집 경로를 복구했습니다.
- OpenTelemetry auto-instrumentation이 충분히 커버하지 못하던 Spring Batch 5에 Job / Step span과 read, write, skip, commit, rollback 집계 metric을 제공하는 공통 tracing 모듈을 구현해 여러 backend 서비스에서 재사용할 수 있도록 했습니다.

CI/CD 플랫폼의 성능과 안정성 개선

- GitHub Actions의 build-test-typecheck-preview 실행 경로와 cache 전략을 개선해 Preview build 시간을 최대 58% 단축했습니다.
- CI workload를 AMD에서 ARM runner로 이전해 container image build 시간을 약 39% 단축하고 runner compute 단가를 약 15% 절감했습니다.
- 11개 repository에 분산돼 있던 EFS 기반 Gradle cache를 self-hosted Actions cache로 통합해 cache 경로를 단일화하고, 공유 archive 손상으로 여러 PR이 cold build로 전환되던 실패 경로를 제거했습니다.
- 반복되던 runner DiskPressure를 cache I/O와 Kubernetes node storage까지 추적해 m7gd local NVMe 기반으로 runner storage를 전환했습니다.

Engineering Platform · Software Engineer

2024.02 — 2025.03

제품 개발 흐름에 통합한 Web E2E 자동화 체계

- Playwright 기반 Web E2E를 제품 코드와 CI/CD에 통합해 별도의 QA 자산이 아니라 제품 개발자가 기능과 함께 관리하는 회귀 테스트 구조를 만들었습니다.
- 핵심 사용자 journey를 배포 과정에서 자동 검증하고 Frontend 조직에서 공통으로 사용할 수 있는 E2E 개발 기반을 마련했습니다.

수작업 계약 관리 업무를 시스템으로 전환

- 문서와 수작업에 의존하던 계약 정보를 도메인 모델과 Admin 기능으로 옮겨 계약 등록·조회·상태 관리가 가능한 시스템으로 전환했습니다.
- 계약 정보를 구조화된 데이터로 관리해 상품·정산 등 후속 업무에서 활용할 수 있는 기반을 만들었습니다.

레거시 서비스와 인프라를 점진적으로 제거

- 실제 호출 관계와 배포 의존성을 확인한 뒤 consumer drain → dependency 제거 → traffic 전환 → infrastructure 제거 순서로 오래된 서비스와 중복 인프라를 정리했습니다.
- 한 번에 교체하기보다 migration 단위를 작게 유지해 제품 개발을 지속하면서 점진적으로 전환했습니다.
- LLM 기반 전자책 생성 POC를 end-to-end로 구현해 콘텐츠 제작 과정에서 자동화할 단계와 사람의 판단을 남겨둘 단계를 나눠 제품 적용 가능성을 검증했습니다.

Website Platform · Software Engineer

2023.01 — 2024.02

SSR 페이지 생성 비용을 사용자 요청에서 분리해 초기 로딩 성능 개선

- 페이지 생성 비용으로 응답이 지연되던 경로에서 stale HTML을 먼저 응답하고 SQS worker가 background regeneration을 수행하도록 변경했습니다.
- 페이지 재생성을 request critical path에서 분리해 TTFB를 줄이고 FCP를 개선했습니다.

탐색 응답 10초 → 100ms 미만

- request-time에 여러 도메인의 데이터를 조합하던 탐색 경로를 Browse Service와 Elasticsearch 기반 read model로 분리해 응답을 10초에서 100ms 미만으로 단축했습니다. 적용 이후 관측된 수감 전환율은 10%p 증가했습니다.

상품·탐색 페이지의 검색 노출을 위한 SEO 기반 구축

- Serverless 기반 Sitemap Service와 공통 생성 프레임워크를 설계·구현해, 대량 URL 분할과 다국어 sitemap index 생성·S3 배포 과정을 공통화했습니다.
- PDP, 검색어 기반 SRP, 카테고리 PLP, Upcoming 상품 등 주요 공개 페이지로 sitemap 적용 범위를 확장하고, handle 기반 카테고리 페이지와 Topic List 내부 링크를 구성해 검색 엔진이 상품·탐색 경로를 발견할 수 있는 기반을 마련했습니다.

GraphQL Platform을 Apollo Federation 2로 전환

- Apollo Server/Gateway와 TypeGraphQL, Codegen 등 관련 tooling의 호환성을 정리하며 Apollo Federation 2로 점진적으로 전환했습니다.

Creator Platform · Software Engineer

2021.08 — 2023.01

글로벌 서비스를 위한 i18n Backend 구축

- Lokalise에서 동기화한 번역을 MongoDB에서 관리하고 runtime에는 CDN cache를 통해 제공하도록 쓰기·읽기 경로를 분리한 i18n Backend를 설계·구현해 p95 응답시간 10ms대를 유지했습니다.
- Lokalise 변경을 webhook으로 동기화하고 CDN cache를 갱신해 애플리케이션 재배포 없이 번역 변경을 반영할 수 있도록 했습니다.

신규 상품 출시와 Product Domain 확장

- 원포인트 클래스 출시를 end-to-end로 개발하고, 정규 클래스 중심이던 Product Domain을 새로운 상품 유형까지 수용할 수 있도록 확장했습니다.
- 대규모 React Router 애플리케이션을 제품 개발과 병행해 Next.js로 점진적으로 전환했습니다.
- 반복되는 장애 대응 절차를 온콜 플레이북으로 정리했습니다.

Internal Tools & Culture · Software Engineer

2021.03 — 2021.08

- 신규 입사자가 필요한 정보를 직접 찾을 수 있도록 사내 Onboarding Web을 개발했습니다.
- 구성원 간 감사와 피드백을 주고받는 사내 칭찬 시스템 Tangerine을 개발했습니다.

Onwave

2020.10 — 2021.03 (6개월)

Co-founder

2020.10 — 2021.03

- 개발자디자이너를 위한 IT 콘텐츠 미디어 Featured를 기획하고 출시했습니다.
- React 기반 여행 가이드북 Travelkit에서 콘텐츠 제작, 여행 코스 추천, 결제 기능을 개발했습니다.

CLASS101

2020.03 — 2020.10 (8개월)

Subscription / Signature TF · Frontend Developer

2020.03 — 2020.10

- 구독 서비스의 구매 퍼널과 상품 구조를 개선하고 신규 구독 상품을 개발했습니다.

Splitz

2019.01 — 2020.03 (1년 3개월)

Founder / Software Engineer

2019.01 — 2020.03

- Kubernetes-Docker 기반 Bare-metal Container Orchestration Platform prototype을 개발했습니다.
- GitLab CI/CD 기반 빌드-배포 자동화 환경을 구축했습니다.

Freelancer

2018.01 — 2019.12 (2년)

Software Engineer

2018.01 — 2019.12

- PHP 기반 웹 서비스를 개발하고 Bare-metal 서버를 운영했습니다.

EDUCATION

Ajou University

2019 — Present

Software Engineering · 재학 중