전소명

✉ callendev20@gmail.com

github.com/Callein

linkedin.com/in/callein

INTRODUCE

Java(Spring Boot)와 Python(FastAPI)을 중심으로, 각 언어와 프레임워크의 장점을 살려 유연한 서비스를 만드는 데 관심이 많습니다. 익숙한 방식을 고집하기보다는, 리소스 부하의 특성에 따라 아키텍처를 효율적으로 분리하거나 상황에 맞는 인프라 구성으로 비용을 절감하는 등 주어진 상황에서 가장 효율적인 방법이 무엇인지 고민하며 개발해왔습니다.

백엔드 개발의 기본은 데이터의 정확함과 시스템의 안정성이라고 생각합니다. 동시성 이슈가 발생할 수 있는 환경에서 데이터가 안전하게 처리되도록 DB 락(Lock)이나 트랜잭션 관리에 신경 쓰고, 트래픽이 몰릴 때도 서비스가 멈추지 않도록 메시지 큐를 활용한 비동기 구조를 적용해 보았습니다. 또한 검색 엔진의 연산 비용을 획기적으로 줄여본 경험처럼, 시스템의 비효율적인 부분을 찾아 개선하는 과정을 즐깁니다.

개발할 때는 "이게 정말 최선일까?"를 자주 묻곤 합니다. AI를 토론 상대 삼아 제가 미처 생각하지 못한 엣지 케이스는 없는지 검토하고, 더 나은 구조를 탐구합니다. 기술적 고집보다는 팀과 비즈니스의 목표를 우선하며, 동료들과 함께 문제를 해결해 나가는 과정에서 성장하고 싶습니다.

SKILLS

BACKEND	• Java(Spring Boot, JPA, MyBatis) • Python (FastAPI, Django)		
Infra & DevOps	• Kubernetes • Docker	• RabbitMQ • GitHub Actions	• AWS
Database	• MySQL • MariaDB		
AI & Data	• FAISS • Annoy	• E5 • fastText	• Hugging Face
Testing & Tools	• Puppeteer • Selenium	• FFmpeg	

Projects

2025. 03 ~ 2025. 07

CMS & Transcode Worker (비동기 분산 미디어 엔진)

Core Repo
Worker Repo

백엔드 리드 개발자 (Worker 시스템 설계 및 구현, API 설계 및 구현)

EDA(Event-Driven Architecture) 설계 및 구현

- **[Problem]** 미디어 변환 작업(CPU Bound)이 API 서버(I/O Bound)와 결합되면, 대용량 업로드 시 전체 시스템 응답 지연 발생
- **[Strategy]**
  - **Decoupling:** RabbitMQ를 도입하여 업로드 요청(Producer)과 변환 작업(Consumer)을 비동기로 분리하고, Worker를 독립적으로 수평 확장가능한 구조로 설계
  - **Buffering:** 트래픽 폭주 시 메시지 큐에 작업을 버퍼링하여 서버 다운을 방지하고 안정적인 처리량 유지
- **[Result]** 동기 처리 대비 API 응답 속도를 대폭 개선하고, 자동 재연결 복구 로직으로 무중단 서비스 가용성을 확보

트랜잭션 제어를 통한 데이터 정합성 보장

- **[Problem]** 파일 삭제 요청이 동시에 발생할 때, S3 객체는 삭제되었으나, DB 메타데이터가 남는 고아 파일(Orphan File) 문제 발견
- **[Solution]** 삭제 트랜잭션 내 **비관적 락(Select ... For Update)**을 적용하여 동시 접근을 제어하고, S3 삭제 → DB 삭제 과정을 원자적으로 처리
- **[Impact]** 데이터 불일치 가능성을 원천 차단하여 스토리지 비용 누수 방지 및 데이터 신뢰성 확보

FFmpeg 파이프라인 최적화

- 기존 멀티 프로세스 방식(다운로드 → 변환 → 업로드 반복)의 비효율을 해결하기 위해 **Filter Complex**를 활용한 **Single-Pass 인코딩** 파이프라인 구축
- 메모리 상에서 스트림을 복제하여 480p/1080p 동시 변환을 수행, **Disk Read I/O를 50% 절감**하고 CPU 컨텍스트 스위칭 비용 최소화

`Spring Boot` `RabbitMQ` `Python` `Celery` `FFmpeg` `Kubernetes` `MariaDB` `MinIO`

2024. 05 ~ 2025. 06

Handong Feed Platform (교내 정보 큐레이션 플랫폼)

Platform Repo
Validator Repo
Tagger Repo

백엔드 개발자 (아키텍처 설계, Core/AI 서비스 개발)

리소스 효율을 극대화한 Polyglot MSA 아키텍처

- **[Problem]** 단일 서버 구조에서 I/O 요청(API)과 CPU 집약적 연산(벡터화, AI 태깅)이 자원을 동시에 사용하면 전체 응답 지연 발생
- **[Strategy]** 트래픽 처리가 주된 Core API는 **Spring Boot**, 벡터 연산 및 AI 로직은 **FastAPI**와 **Python Script**로 분리하는 Polyglot MSA 적용. 각 서비스 간 결합도를 낮추기 위해 REST API 통신을 강제하고 **RBAC** 기반 **API Gateway** 구축
- **[Result]** 서비스별 리소스를 격리하여 연산 부하 시에도 API 응답 속도를 보장하고, 단일 서버 기준 **CPU 점유율을 30% 미만**으로 최적화

TF-IDF/Annoy 기반 실시간 중복 탐지 및 데이터 압축

- **[Goal]** 8,883건 이상의 수집 데이터 중 무의미한 중복 데이터(재업로드 등)를 걸러내고, 유사한 정보끼리 묶어 사용자에게 높은 밀도의 정보를 제공해야 함
- **[Strategy]**
  - **Warm-Start Indexing**: 최근 14일치 데이터를 메모리에 로드하여 **TF-IDF Vectorizer**와 **Annoy Index**를 웜스타트
  - **Classification**: 입력 메시지의 벡터 거리를 계산하여 **Duplicate**(거리 0.0), **Similar**(임계값 이하), **New**(임계값 이상)로 분기 처리하는 정교한 파이프라인 구현
  - **Async Processing**: 대량의 데이터 유입 시 병목을 막기 위해 Asyncio를 활용한 비동기 벌크(Bulk) 처리 구현
- **[Result]** 건당 **0.9s** 이내의 고속 분류 성능을 확보하고, 전체 데이터의 **70%를 중복/유사 게시물로 자동 필터링**하여 사용자에게 높은 밀도의 정보를 제공

PII 보호 및 결함 허용(Fault Tolerance) 시스템

- **[Problem]** 교내 공지, 홍보물 특성상 이름, 전화번호 등 민감 정보(PII)가 포함될 수 있어, 이를 외부 LLM(Gemini 등)에 전송 시 보안 리스크 존재
- **[Strategy]**
  - **PII Masking:** KoELECTRA(NER) 모델과 정규표현식을 결합한 하이브리드 마스킹 파이프라인을 구축하여 LLM 전송 전 민감 정보 원천 차단
  - **Fault Tolerance:** 외부 AI 태깅 서비스 장애 시 요청을 유실하지 않고 DB에 기록 후, 시스템 복구 시 자동 재시도하는 로직 구현
- **[Result]** 개인정보 유출 리스크를 차단하고, 외부 의존성 장애 상황에서도 데이터 정합성을 유지하는 안정적인 파이프라인 확보

Serverless 기반 이벤트 구동 워크플로우

- **[Problem]** 태깅 작업은 간헐적으로 발생하므로, 이를 위해 별도의 워커 서버를 24시간 상시 가동하는 것은 명백한 **유휴 자원낭비**
- **[Strategy]** 상시 서버 대신, 데이터 수집 및 1차 검증이 완료된 시점에만 **GitHub Actions Runner**를 트리거하여 태깅 작업을 수행하고 종료하는 **On-Demand Serverless 패턴** 도입
- **[Result]** 서버 대기 시간 동안 발생하는 불필요한 인프라 비용을 제거하여 운영 비용 최소화

`Spring Boot` `FastAPI` `JPA/MyBatis` `MySQL` `Docker` `GitHub Actions` `Vector Search` `Asyncio`

2025. 01 ~ 2025. 08

RAGvertise (AI 광고 추천 엔진)

RAGvertise Repo

산학 연구원, 메인 개발자 (AI 추천 엔진 및 검색 API 개발, 성능 최적화 주도)

KTL 공식 성능 인증 획득 (Latency 0.25s, Recall@5 0.87)

- 한국산업기술시험원(KTL)을 통해 매칭 알고리즘의 응답 속도와 정확도에 대한 공인 성적서 획득 (ISO/IEC 17025)

검색 엔진 고도화 (V1 → V3)

- **[Goal]** 유저(광고주)가 원하는 느낌(추상적 쿼리)을 입력하면 최적의 광고 포트폴리오를 검색
- **[Strategy]**
  - **Phase 1 (V1. 단순 결합):** LLM으로 추출한 메타데이터(설명, 스타일 등)를 하나의 텍스트로 결합하여 검색했으나, 필드별 중요도가 희석되어 매칭 정확도가 낮은 한계에 직면
  - **Phase 2 (V2. 다중 인덱스):** 정확도 개선을 위해 5개 필드(Desc, What, How 등)를 개별 인덱스로 분리하여 가중치를 부여해 검색 품질은 높였으나, 연산량이 5배 증가하여 응답 속도가 저하되는 병목 현상 발생
  - **Phase 3 (V3. Weighted Late Fusion)**: 벡터 결합 단계에서 각 필드의 가중치를 선반영하는 **Weighted Late Fusion** 기법 도입. 5회의 연산을 **1회 내적 연산으로 통합**하여 구조적 비효율 해결
- **[Result]** 검색 정확도(Recall@5) **0.87**을 유지하면서도 연산 비용 75% 절감 및 레이턴시 **0.25s** 달성

하이브리드 임베딩 및 데이터 신뢰도 확보

- **[Problem]** 5가지 검색 필드 중 Desc/Full은 문장형 문맥인 반면, What/How/Style은 단어형 키워드 이므로, 이를 단일 모델(E5)로 임베딩할 경우, 짧은 단어의 의미적 특성을 충분히 반영하지 못해 매칭 정확도 한계 발생
- **[Strategy]** 문장형 필드(Desc)는 문맥 파악에 강한 **E5**를, 단어형 필드(Style 등)는 형태소 및 단어 유사도 분석에 강한 **fastText**를 적용하는 이원화 전략 수립
- **[Result]** 입력 데이터 분류 정확도(ROUGE-1 F1) **0.98**을 기록하여 검색 품질 신뢰도 확보

외부 서비스 연동 안정성 및 데이터 정합성 확보

- **[Problem]** 외부 LLM API(Gemini)의 호출 제한(Rate Limit) 초과 및 잡담이 섞인 비정형 응답으로 인한 파이프라인 중단 리스크 존재
- **[Strategy]**
  - **Rate Limiter:** 분당 요청 수(RPM)를 엄격히 제어, 429 에러 원천 차단
  - **Fallback Parsing:** LLM이 잡답이 섞인 비정형 응답을 반환하더라도 Regex 기반 파서로 정형화된 JSON 데이터 구조를 강제하도록 설계
- **[Result]** 외부 API 의존성이 높은 환경에서도 예외 상황을 최소화하여 안정적으로 동작하는 견고한 추천 엔진 구축

`FastAPI` `Python` `FAISS` `Vector Search` `React` `NumPy`

Experience(Internship)

2024.12~2025.02

구도투자자문(Software Engineer Intern) - QA Automation

투자 데이터 정합성 교차 검증 자동화

- 포트폴리오 UI 표출 수치와 실제 재무제표(Statement) 원본 간의 오차를 감지하는 자동화 검증 도구 개발
- 데이터 불일치 항목을 JSONL 로그로 자동 적재하여 투자 정보의 신뢰성 리스크를 사전에 차단

검증 프로세스 최적화 및 업무 효율 개선

- **[Problem]** 기존 Selenium 기반 테스트의 무거운 실행 속도와 반복 로그인 문제로 점검 효율 저하
- **[Solution]** Puppeteer(CDP)로 기술 스택 전환 및 세션 재사용(User Data Dir) 로직 구현으로 초기화 시간 단축
- **[Impact]** 9가지 핵심 점검 항목을 100% 자동화하여 일일 QA 소요 시간을 70% 절감

비개발 직군을 위한 사내 QA 도구화

- 개발 지식이 없는 동료도 손쉽게 정합성 검증을 수행할 수 있도록 CLI 메뉴 및 Batch Script(.bat) 실행 환경 구축

`Puppeteer` `Selenium` `QA Automation` `Shell Script`

Education

2020.03 – 2026.02 <졸업 예정>

한동대학교

AI / 컴퓨터공학 심화 전공 (3.81 / 4.5)

2025.08 – 2025.12

LeTourneau University (Exchange)

Computer Science

Awards & Activities

2025. 09

한동대학교 스마트 어플리케이션 공모전 우수상 CMS & Transcode Worker

교내 동아리의 효율적인 운영과 활동 기록 지식의 영속적 보존을 위한 Web App 및 미디어 엔진 개발

2025. 08 ~ 2025. 12

제3기 한미 첨단분야 청년교류 지원사업 장학생 (산업통상자원부 / 한국산업기술진흥원)

미국 LeTourneau University (Texas) 파견 (Software Engineering, Network Security 수학)
Project: LLM 기반 로컬 뉴스 콘텐츠 자동 선별 및 요약 시스템 설계 연구

2025. 08

KTL 소프트웨어 품질 성능 인증 (ISO/IEC 17025)한국산업기술시험원 (Korea Testing Laboratory)

RAGvertise (AI 추천 엔진) 공식 성능 시험 성적서 획득
매칭 알고리즘 Latency 0.25s, Recall@5 0.87 달성 인증

2025. 01

COSS 전북대 올림피아드 (LLM 정확도 향상) 최우수상 (빅데이터 혁신융합대학 7개 대학 연합)

30% 성능의 모델을 RAG/FAISS/Sentence Transformers를 활용해 96.1%까지 최적화
참가 20개 팀 중 1위 (Top Award) 수상

2024. 11

한동대학교 스마트 어플리케이션 공모전 대상 Handong Feed Platform

교내 정보 통합을 위한 MSA 기반 큐레이션 플랫폼 개발

2021. 12

한동대학교 스마트 어플리케이션 공모전 대상 HUT

불규칙한 대중교통 문제를 해결하기 위한 교내 택시 동승 매칭 O2O 플랫폼 (Flutter, Firebase)