이건

서버/백엔드 개발자 | 01031804727 | doolchong@gmail.com

Blog Github

저는 호기심과 열정으로 개발하는 백엔드 개발자입니다. Java와 Spring 프레임워크를 주로 사용하며, 기술적 문제를 깊이 분석하고 해결하는 것을 즐깁니다. 코드의 성능과 가독성, 유지보수성을 항상 고려하며, 팀원들과의 협업을 통해 더 나은 솔루션을 만들어내는 것을 중요하게 생각합니다. 새로운 기술 습득에 열정적이며, 지속적인 성장을 추구합니다.

경력 사항

더선한

인턴

2025.02~ 2025.03 (2개월)

기업 정보 크롤링 자동화

1. 대규모 기업 정보 수집 자동화 시스템 구축 (누적 43만 건)
- Levenshtein 거리 알고리즘 기반 중복 데이터 통합
- Spring Batch 기반 주기적 업데이트(주 단위)
2. 안정성 강화 설계
[에러 처리]
• 재시도: 일시적 네트워크 장애
• 건너뛰기: 비정형 데이터 예외
• 중단: 시스템 장애 등 치명적 오류
[복구 기능]
• 중단 시점부터 작업 재개 구현

프로젝트

Trend-Chat

2025.06~ 2025.08 (3개월)

Github

팀프로젝트(BE 2명)

프로젝트 소개

실시간 트렌드 수집 · 분석 · 알림과 인기 트렌트 키워드 기반 채팅 기능을 통합한 마이크로서비스 플랫폼입니다. **LangGraph 기반 멀티 에이전트 시스템**으로 요약/분류/생성 기능을 자

동화하고, Kafka 이벤트 흐름에 따라 **도구(tool)를 활용하는 자율형 LLM Agent**가 작동합니다.

적용 기술

- 프레임워크: Spring Boot, Spring Cloud Gateway, Next.js
- 데이터베이스/캐시: MySQL, Redis(Sentinel)
- 메시지 브로커: Kafka, SSE
- 배포/운영: Kubernetes, Docker, GitHub Actions, MetalLB, Ingress-Nginx, Prometheus/Grafana, Zipkin
- AI/에이전트: Python, LangChain/LangGraph, Groq Llama3, Google Gemini

---

역할

1. Kafka 기반 데이터 파이프라인(Producer/Consumer) 설계 및 구현
2. Google Trends 정적/RSS + Selenium 동적 크롤러 개발
3. Redis ZSET 상위 키워드 스코어링/TTL 동기화 알고리즘 설계
4. LangGraph FSM 에이전트 워크플로우·도구 연동(Serper, Trafilatura) 설계
5. OAuth2(JWT, RTR, 블랙리스트) + Spring Cloud Gateway 인가 구조 구축
6. K8s 배포 자동화(bootBuildImage→Docker Hub→Ingress/MetalLB), self-hosted runner 운영

트러블슈팅 및 성능개선

1. **LLM 프롬프트/토큰 폭발 문제 및 Supervisor 제거**
   LangGraph 기반 에이전트 실행 중 Supervisor 호출 단계에서 응답 지연 문제 발생
   - 각 Agent의 결과가 state['messages']에 누적되어 프롬프트 토큰이 3,000~4,000개 이상으로 폭증
   - 프롬프트 최소화 및 상태 관리 개선을 통해 평균 응답 시간 **30초 → 5초 이내로 단축**
   - Supervisor 노드를 제거하여 **아키텍처 단순화 및 안정성 향상**
2. **Kafka Consumer 중복 처리 문제**
   AI Agent 단계별 메시지가 중복 분배되어 순서 불일치 및 중복 처리 발생
   - 기본 Consumer Group이 병렬 분산 처리(Partition-based)를 전제로 작동
   - 단계별 토픽 분리 및 keyword_id 기반 파티셔닝 키를 강제 적용
   - 동일 키는 항상 같은 파티션으로 전달되어 **순서 보장 및 중복 처리 해소**
   - 파이프라인 안정성 확보 및 **메시지 처리 일관성 개선**
3. **쿠버네티스 마스터 노드 네트워크 장애 (ARP 캐시 오염)**
   로컬 Kubernetes 클러스터에서 마스터 노드만 SSH/Ping 불가 현상 발생
   - Host PC의 ARP 캐시에 잘못된 MAC 주소가 고착(Stale)된 것이 원인
   - arp -d <IP> 명령어로 캐시 삭제 후 즉시 복구 확인
   - 설정이 정상이더라도 **상태(캐시) 오염으로 장애가 발생할 수 있음**

- 네트워크 계층별 Top-Down 진단 방식을 통해 **문제 범위 신속 축소**

4. **쿠버네티스 클러스터 연쇄 장애 (NFS / PV-PVC / DNS / Cgroup)**
   단순 kubectl connection refused 에러에서 시작된 복합 인프라 장애 발생
   - 잘못된 DNS 설정, NFS 캐시 손상, PV Finalizer 교착 등 다중 원인이 복합 작용
   - --resolv-conf=/etc/resolv.conf 적용, cachefilesd 초기화, PV 재생성으로 문제 해결
   - 클러스터 전면 복구 및 **스토리지·네트워크 역할 분리 정책 수립**
   - 인프라 안정성과 운영 효율성 향상

FitNus

2024.10~ 2024.11 (2개월)

Github

팀프로젝트(BE 5명)

프로젝트 소개

FitNus는 기존에 월 단위로 운동을 결제해야 했던 시스템을 벗어나 일 단위로 원하는 날짜에 원하는 운동을 일정으로 등록하고 예약하는 서비스입니다.

적용 기술

- 프레임워크: Spring Boot, Spring Batch
- 데이터베이스: MySQL, Redis
- 메시지 브로커: SSE, Kafka
- 검색 엔진: Elasticsearch
- 인증/보안: Spring Security, JWT
- 배포 환경: AWS (EC2, RDS, S3, ECR, VPC, ALB, AutoScaling), Docker, GitHub Actions, Terraform
- 외부 API: 카카오페이, 카카오 로그인, 카카오 지도
- 고가용성: Redis Cluster

역할

1. 모임, 멤버, 일정 기능 CRUD
2. 일정 등록 기능 동시성 제어 및 성능 개선
3. 정산 기능 - 대용량 데이터 처리를 위해 스프링 배치 사용
• 일 단위 센터 이용내역 기록
• 월 단위 센터별 정산
4. 모놀리식 구조에서 멀티 모듈로 구조를 변경
5. Slack API를 이용한 개발자 알림

트러블슈팅 및 성능개선

1. **모듈별 의존 관계를 재정비**

기존 구조에서 'notification' 모듈에서 처리해야 할 Kafka 파티션을 'service' 모듈이 가져가는 문제가 발생

- 'notification'과 'service' 모듈 간의 불필요한 의존성을 제거
- 모든 모듈이 'common' 모듈만을 의존하도록 수정
- 모듈 간의 의존 관계가 단순화되었으며, 유지 보수성과 확장성이 향상됨

2. **일정 등록(예약) 동시성 제어 및 성능 개선**

일정 등록(예약) 기능 부하 테스트 진행 중 동시성 문제 발생

- 동시성 제어를 위해 Redis 분산락을 이용했으나, **TPS가 108.6/sec**로 성능이 떨어짐
- Redis atomic 연산을 활용하여 개선 후 **TPS가 522.3/sec**로 개선됨
- Redis Lua Script를 활용하여 개선 후 **TPS가 550/sec**로 개선됨

3. **일 단위 센터 이용내역 기록 성능 개선(10만건 기준)**

대용량 데이터 처리를 위해 Spring Batch를 활용하여 기능 구현 → 처리 속도 9m45.6s(기준)

- **멀티스레드** 적용 → 처리 속도가 4m40.99s로 기준 대비 **52.01% 개선**
- **파티셔닝** 적용 → 처리 속도가 4m37.8s로 기준 대비 **52.56% 개선**
- **쿼리메소드 saveAll** 적용 → 처리 속도가 2m19.2s로 기준 대비 **76.22% 개선**

AI 해충 진단 어플리케이션

2023.10~ 2023.11 (2개월)

팀프로젝트(BE 2명, AI 1명, FE 1명)

프로젝트 소개

YOLO 기반의 해충 진단 모델을 학습하고 Flask 기반 추론 서버를 구축한 프로젝트입니다. React Native 앱에서 이미지를 업로드하면, Flask 서버가 실시간으로 해충 종류를 판별해 JSON 형태로 진단 결과를 반환합니다.

적용 기술

- 프레임워크: PyTorch, Flask, React Native
- AI/모델링: YOLOv5

역할

1. AI 허브 및 자체 촬영 데이터 기반 학습 파이프라인 구성 (데이터 증강: Flip, Hue, Brightness 등)
2. PyTorch/Ultralytics YOLOv8으로 모델 학습 및 성능 평가(mAP@0.5)
3. Flask 기반 추론 서버 구축 및 앱-서버 간 REST API 통신 구현
4. 모델·API·앱 구조를 분리하여 배포 효율성 확보

성과

- **평균 mAP 0.94 (IoU 0.5)** 달성
- **모델·API·앱 분리 구조로 업데이트/배포 용이성 확보**
- **클래스 불균형 개선**을 통해 소수 클래스 탐지 정확도 30% 이상 향상

트러블슈팅

데이터셋 내 클래스 불균형으로 일부 해충(예: 복숭아진딧물)의 탐지율 저하 문제 해결

- AI 허브 데이터를 활용한 **소수 클래스 증강**
- **클래스별 증강 비율 조정**으로 학습 기회 균형화
- **Loss Weight 보정**으로 과적합 완화
  → 평균 **mAP 0.87 → 0.94로 개선**, 탐지 성능 향상 확인

대외활동 및 수상

스파르타 내일배움캠프 Spring 6기

2024.07~ 2024.11 (5개월)
2024.09.02 ~ 2024.09.06 뉴스피드 프로젝트
2024.09.19 ~ 2024.09.25 아웃소싱 프로젝트
2024.10.14 ~ 2024.10.18 심화 프로젝트
2024.10.21 ~ 2024.11.22 최종 프로젝트
내일배움캠프 Spring 6기 모범상

학력 사항

세종대학교

컴퓨터공학과 학사
2017.03~ 2024.02(졸업)

2017년 교내 제4회 SW경시대회(세종코딩챌린지위크) 장려상
2023년 제16회 창의설계경진대회 장려상