고정완

https://github.com/ghojeong
010-3878-1459
rhwjddhks@gmail.com

학력

컴퓨터공학 | UNIST

- 전기전자 컴퓨터공학과 성적우수 졸업 (수리과학 부전공)

성과 요약

- 호스팅 업체에서 AWS 클라우드로 인프라 전환 후 **안정성 99%** 개선
- 모놀리식에서 MSA 로 리팩토링 후 테스트 커버리지 86% 달성
- 인증 서버를 세션 기반에서 토큰 기반으로 변경 후 **인증시간 93%** 단축
- 쿼리튜닝 및 스키마 재설계 후 **서버비용 82%** 절약

실무 경험 (6 년차)

Backend Tech Lead | Dreamfora *(2021.08.25 –)*

- 사용기술: Java, Spring, Spring Security, JPA, MariaDB
- 미국, 영국 등, 전세계 200 만명이 사용하는 B2C 목표관리 앱을 개발
- Dreamfora 경력기술서 링크

Web Developer | Ecube Labs *(2018.10.15 – 2020.12.28)*

- 사용기술: TypeScript, React, Redux, Express.js
- 미국 볼티모어, 일본 이치카와, 한국 고양시와 계약한 B2B 스마트 시티 SaaS 를 개발
- Ecube Labs 경력기술서 링크

교육 경험 (5 년차)

Kotlin Reviewer | 우아한테크코스 *(2024.02.13 – 2024.06.24)*

- 우아한테크코스에서 크루들을 지도하는 Kotlin 리뷰어로 근무
- 리뷰 예시 링크: 람다와 함수형 프로그래밍, 테스트와 예외처리

Spring JPA Reviewer | NEXTSTEP *(2021.03 –)*

- WAS 와 ORM 프레임워크를 바닥부터 구현하도록 돕는 리뷰어로 근무
- 리뷰 예시 링크: 쿠키에 Null Object Pattern 도입, 멀티 쓰레드에서의 영속성 컨텍스트

세미나 강사 | 한빛미디어 *(2019.03 – 2019.06)*

- 개발 문서화를 주제로 한빛미디어의 후원을 받아 공감세미나 주니어를 개최
- 사전정의서, 요구사항 명세서, ERD, 프로세스 Flow Chart 문서 작성하는 법을 강의

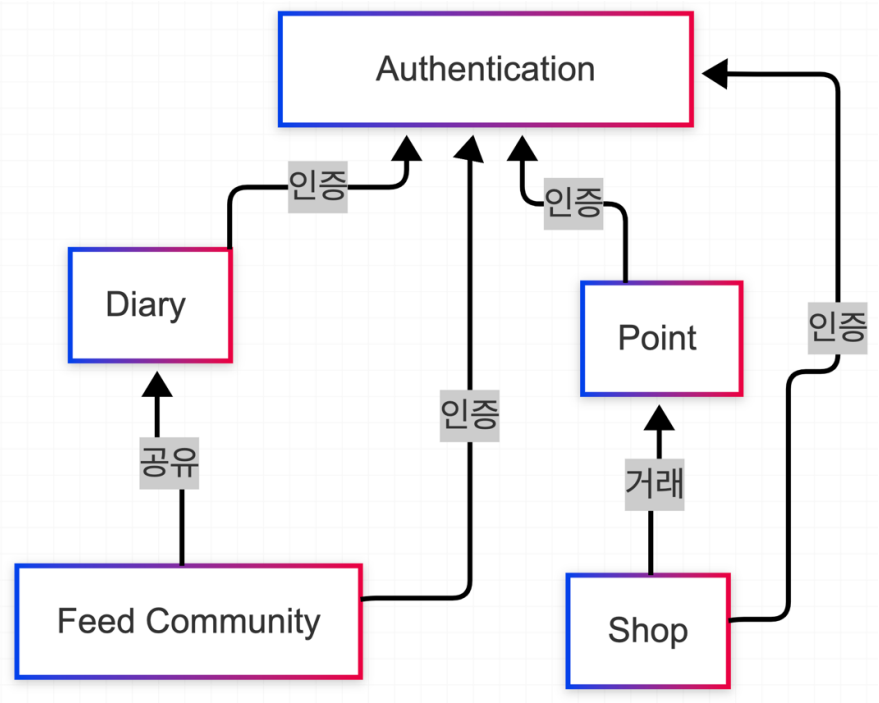
Dreamfora 경력기술

클라우드 전환 | 안정성 99% 개선

- 문제: 호스팅 업체에 문제가 생기면 서비스가 정지됨
- 원인: Single Point of Failure 가 되는 물리적 서버가 존재
- 해결책: 이중화, 망분리, CI/CD
  - 트래픽 폭증 시 서비스 정지의 원인이 되는 WAS 서버를 이중화
  - 보안 공격 시 데이터가 날아갈 수 있는 DB 를 Virtual Private Cloud 세팅을 통해 인터넷과 격리
  - Continuous Integration 시, 빌드와 테스트 성공 여부를 확인
  - 도커 이미지를 활용한 Rolling 방식의 무중단 배포 도입

MSA 도입 | 테스트 커버리지 86% 달성

- 문제: 신규 기능 추가 시, 전체 서비스에 버그가 생김
- 원인: 모놀리식 구조라 의도치 않은 Side Effect 가 발생
- 해결책: MSA 도입
  - Micro Service 의 기준: 의존성이 분리된 패키지가 별개의 물리적 서버와 Entry Point 를 가짐
  - 단일 패키지를 도메인을 기준으로 5 개의 패키지로 분리
    - 분리된 패키지 간의 의존성을 분리
  - MSA 로 리팩토링된 이후의 정상동작을 검증학기 위해 테스트 작성
    - 도메인 객체에는 단위 테스트를, 서비스 시나리오에는 인수 테스트를 작성
  - 분리된 패키지에 별개의 Entry Point 를 구현
  - 분리된 패키지마다 별개의 물리적 서버를 할당
  - 별개의 물리적 서버와 Entry Point 를 가졌다면, Mico Service 로 분리되었다고 정의 M
  - 트래픽 때문에 죽을 수 있는 Micro Service 는 이중화
  - 보안 상 중요 데이터를 다루는 Micro Service 는 망분리된 Private Cloud 로 격리



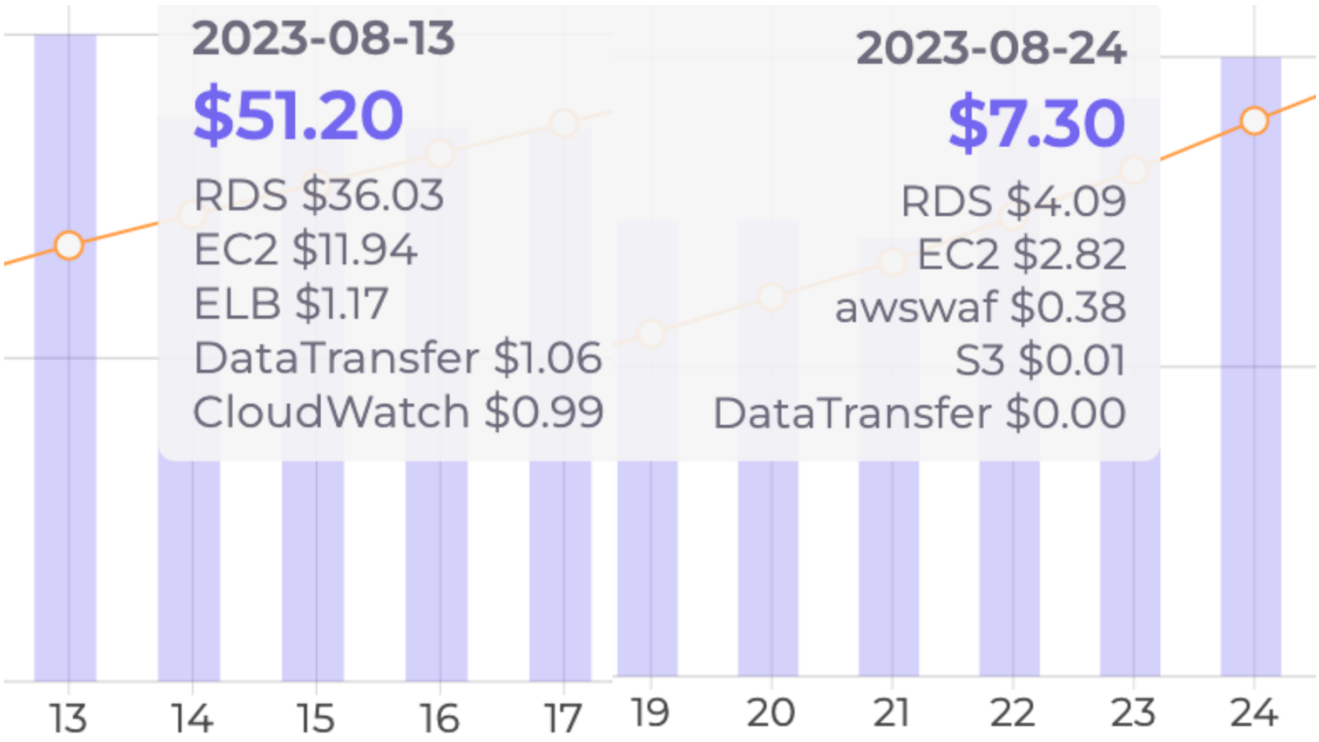
- 마이크로 서비스 설명
  - Authentication 서비스
    - 사용자 인증 및 권한을 부여
    - 모든 서비스와 상호작용
  - Diary 서비스
    - 매일의 목표수행 데이터를 자동 동기화
  - Feed Community 서비스
    - 게시글, 댓글, 대댓글, 좋아요를 남길 수 있는 게시판
    - Diary 에서 작성한 내용을 공유 가능
    - 트래픽이 많이 몰리기에 이중화
  - Point 서비스
    - 이벤트성, 환금성 포인트를 획득 및 소비
    - 포인트 획득 및 소비 내역을 조회
    - 보안상 중요하기에 Private Cloud 로 망분리
  - Shop 서비스
    - Point 를 소비하여 아이템을 거래
    - 아이템 거래 이력을 조회

토큰 기반 인증 도입 | **인증시간 93% 단축**

- 문제: 인증 시 8 초가 넘는 시간이 소요되는 경우가 존재
- 원인: 세션의 풀스캔
  - 세션으로 메모리가 아닌 Storage 를 사용
  - Session TTL 설정이 잘못되어, 8 천만개의 세션이 휘발되지 않고 쌓임
  - 세션에 인덱스가 걸리지 않아 풀스캔
- 해결책: Sessionless 인증 도입
  - 인증에 필요한 데이터를 세션이 아닌 JWT 로 관리
  - 보안을 위해 accessToken 과 refreshToken 을 분리
    - accessToken: 보안을 위해 일주일마다 갱신
    - refreshToken: 일주일마다의 accessToken 갱신에 사용됨

쿼리튜닝 및 스키마 재설계 | **서버비용 82% 절약**

- 문제: 검색 시 메모리와 CPU 자원을 많이 소모
- 원인: 검색 시 다루는 테이블의 데이터 크기가 큼
- 해결책: Horizontal Partitioning
  - 주 단위로 테이블을 분리
  - 현재 테이블에는 1 주일 동안 생성된 데이터만 남기고, 나머지는 주 단위로 분리한 테이블로 이동
  - 1 달이 지난 데이터는 비용이 싼 아카이빙 스토리지 서버로 이동하여 비용 절감
  - 현재 테이블이 아닌 과거의 테이블은 Pagination 을 통해 과거의 데이터를 찾는 경우에만 사용
  - 아카이빙된 데이터 때문에 성능 이슈가 생기면, 현재 테이블로 데이터를 수작업으로 이동시킴



데이터 마이그레이션 | **마이그레이션 시간 75% 단축**

- 문제: 데이터 마이그레이션에 1 시간 소요
- 원인: JPA 의 N+1 쿼리
  - 마이그레이션 로직의 검증을 위해 JPA 와 테스트 코드를 활용
  - 설정된 Batch Size 보다 많은 데이터를 마이그레이션 시, N+1 쿼리가 발생
- 해결책: EntityManager 의 커스텀
  - rewriteBatchedStatements 옵션을 활용
  - Persistable 인터페이스를 상속하여 isNew 를 항상 true 로 설정