PARK MI HEE

Backend Developer

안녕하세요.
사용자에게 안정적인 서비스를 제공하는 것을 목표로 삼고있는 1년차 백엔드 개발자 박미희입니다.

안정적인 서비스

새로운 기술을 도입하기 전 PoC를 통해 검증한 후 실제 서비스에 적용합니다. 이를 통해 리스크를 최소화하고 안정적인 서비스를 구축합니다.

성장하는 개발자

서비스를 개선하기 위해 문제를 분석하고, 해결 방안을 찾기 위해 지속적으로 학습합니다.

협업하는 개발자

문제를 해결하기 위해 팀원과 협업하며 함께 성장하는 것을 선호합니다.
또한, 현업에서 기획자와의 관점차이를 조율하며 협업한 경험이 있습니다.

+82 010-9461-0649
mihee78952@naver.com

https://github.com/PMH2906
https://bellejoie.tistory.com/

PARK MI HEE

그동안의 활동, 자격증, 수상 정보 및 이력을 소개합니다.

ACTIVITIES

삼성 청년 SW 아카데미 8기 (총 1600시간 이수)

- 자바기반의 웹 개발교육 800시간이수
- 협업 프로젝트 다수 진행

CERTIFICATES

SQLD	2021.12
ADSP	2021.12
정보처리기사	2024.9

EDUCATION

세종대학교 졸업 - 지능기전공학부 무인이동체공학과 2022.08

AWARDS

SAFFY 자율 프로젝트 우수 2023.06
세계 시사 상식 학습 서비스 'Worldy'를 개발하여 우수 프로젝트로 선정

SAFFY 특화 프로젝트 우수 2023.04
전통주 빅데이터 추천 서비스 '술내음'을 개발하여 우수 프로젝트로 선정

SKILLS

Spring | JAVA | JPA | MySQL | Tibero | Redis | AWS EC2 | S3 | NginX | Docker | Git | Jira

CAREERS

Tmax Fintech
2023.07-ING

코어 뱅킹 시스템 2024.06-2024.10

MSA 기반의 코어 뱅킹 시스템

주요 기여

- MSA 아키텍처 설계
- SAGA 패턴 구현
- Transaction Outbox 패턴 구현

배달 서비스 공제 조합 2023.07-2024.03

배달 종사자에게 특화된 보험 서비스

주요 기여

- Redis를 도입하여 사고번호 채번 방식 고도화
- 낙관적 락을 도입하여 동시성 문제 개선
- 인덱스와 QueryDsl을 사용해 조회 쿼리 개선
- 데이터 암복호화 환경 고도화

PROJECT

대표 프로젝트를 간략하게 소개합니다.
자세한 기술은 프로젝트 별 페이지에 소개됩니다.

코어 뱅킹 시스템

MSA 기반의 코어 뱅킹 시스템

Tmax Fintech | 2024.06 - 2024.10

담당 업무

- 수신 파트 백엔드 개발

주요 기여

- MSA 아키텍처 설계
- SAGA 패턴 구현
- Transaction Outbox 패턴 구현

3p

배달 서비스 공제 조합

배달 종사자에게 특화된 보험 서비스

Tmax Fintech | 2023.07 - 2024.03

담당 업무

- 보상 기간계 시스템의 백엔드 개발

주요 기여

- Redis를 도입하여 사고번호 채번 방식 고도화
- 낙관적 락을 도입하여 동시성 문제 개선
- 인덱스와 QueryDsl을 사용해 조회 쿼리 개선
- 데이터 암복호화 환경 고도화

5p

술내음

사용자 맞춤 전통주 추천 서비스

삼성 청년 SW 아카데미 | 2023.02 - 2023.04(7주)

담당 업무

- DevOps / Backend

주요 기여

- 인프라 구축
- JWT 기반 소셜 로그인
- User RestAPI 구현

7p

코어 뱅킹 시스템

Tmax Fintech | 2024.06-2024.10

MSA 기반의 코어 뱅킹 시스템

팀원 총 8명 **성과** 코어 뱅킹 시스템에 MSA 도입

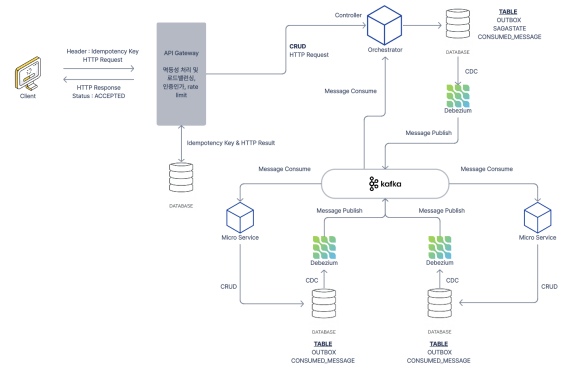
링크 https://github.com/PMH2906/CoreBank-MSA-PoC

기술

- **Backend**

  SpringBoot | Java | JPA | Tibero | Kafka

아키텍처



담당 업무

- 수신 파트 백엔드 개발

주요기여

- MSA 아키텍처 설계
- SAGA 패턴 구현
- Transaction Outbox 패턴 구현

대표 트러블 슈팅

1 SAGA 패턴 구현

✔문제

MSA를 도입하면서 기존의 Monolithic 방식으로는 분산 트랜잭션의 원자성을 유지하기 어려웠습니다.

✔해결

- 분리된 애플리케이션과 DB 환경에서 해당 문제를 해결하기 위해 Orchestration기반의 SAGA 패턴을 도입했습니다.
- SAGA Orchestrator서버를 추가하여 SAGA와 분산 트랜잭션을 관리하였습니다.

✔결과

- MSA환경에서 분산 트랜잭션의 원자성을 유지하기 위한 방법을 배웠습니다.
- Framework 없이 SAGA Orchestrator를 구현하면서 분산 트랜잭션의 보상 로직을 이해했습니다.

코어 뱅킹 시스템

Tmax Fintech | 2024.06-2024.10

MSA 기반의 코어 뱅킹 시스템

대표 트러블 슈팅

2 Transaction Outbox 패턴 구현

✔문제

Event Driven Architecture를 도입하며 서비스 간 IPC가 발생할 때, 데이터베이스 변경과 메시지 발행 간의 원자성이 깨지는 문제가 발생했습니다.

✔해결

- 해당 문제를 개선하기 위해 Transaction Outbox 패턴을 도입하였습니다.
- 서비스 간 IPC가 필요한 경우, Outbox 테이블에 데이터를 삽입하고, Outbox 테이블의 변경 사항을 감지한 Debezium 커넥터가 설정된 Kafka의 토픽에 메시지를 발행하도록 구현했습니다.

✔결과

- Debezium에서 제공하는 Transaction Outbox 코드를 기반으로 코어 뱅킹 시스템에 적용하여 트랜잭션의 원자성을 유지했습니다.

배달 공제 조합 서비스

Tmax Fintech | 2023.07-2024.03

배달 종사자에게 특화된 보험 서비스
여러 공제상품을 설계하고, 가입 및 보상 등의 서비스를 제공

팀원 총 8명(보상팀 백엔드 8명)

성과 보상 기간계 시스템 1차 개발 및 QA 완료

링크 -

기술

- **Backend**
  SpringBoot | Java | JPA | Tibero

아키텍처

-

담당 업무

- 보상 기간계 시스템의 백엔드 개발
- 모바일 사고 현황 조회 API 개발
- 현장출동보고서 관련 API 개발

주요기여

- Redis를 도입하여 사고번호 채번 방식 고도화
- 낙관적 락을 도입하여 동시성 문제 개선
- 인덱스와 QueryDsl을 사용해 조회 쿼리 개선
- 데이터 암복호화 환경 고도화

대표 트러블 슈팅

1 Redis를 도입하여 채번 방식 고도화

기술 블로그 링크) https://bellejoie.tistory.com/29

✔문제

고유한 사고번호를 채번하기위해 DB에 존재하는 사고번호의 MAX값+1 방식의 트리거를 사용했습니다. 그러나 트랜잭션이 동시에 발생할 경우, 동일한 사고번호가 중복 생성되어 하나를 제외한 나머지 트랜잭션이 롤백되었습니다.

✔해결

- 싱글 스레드인 Redis에 사고번호의 MAX 값을 저장하여 동일한 사고번호가 채번되는 문제를 개선했습니다.
- Redis의 장애를 대비해 Redis Cluster를 구성했습니다.
- Redis가 모두 다운되면 기존의 MAX+1 방식의 프로시저를 이용하여 사고번호를 채번합니다.

✔결과

- JMeter를 통해 테스트한 결과, 동시 요청을 1/N 확률로 처리하던 것을 100% 확률로 처리하도록 개선했습니다.

배달 공제 조합 서비스

Tmax Fintech | 2023.07-2024.03

배달 종사자에게 특화된 보험 서비스
여러 공제상품을 설계하고, 가입 및 보상 등의 서비스를 제공

대표 트러블 슈팅

2 동시성 제어

✔문제

데이터의 정합성을 보장하기 위해 JPA의 영속성 컨텍스트를 활용하여 트랜잭션 격리 수준을 REPEATABLE READ로 설정했습니다. 그러나 특정 데이터를 동시에 수정할 때 데이터가 유실되는 lost update 문제가 발생했습니다.

✔해결

- 동시성 문제를 해결하기 위해 JPA의 낙관적 락을 도입했습니다.
- version 컬럼을 추가하여 변경된 데이터를 commit하는 시점에 엔티티의 version정보와 DB의 version을 비교했습니다. 일치하지 않으면 트랜잭션을 롤백 처리하여 중복 업데이트를 방지했습니다.

✔결과

- 버전이 일치하지 않으면 트랜잭션을 롤백하여 lost update 문제를 방지했습니다.
- 락 메커니즘을 이해하였고, 시스템의 요구사항과 서버의 성능을 고려하여 적절한 락을 도입하는 방법을 배웠습니다.

3 암복호화 환경 개선

✔문제

기존에는 NativeQuery에 암복호화 함수를 사용하여 암복호화를 진행했습니다. 해당 방식은 다음 문제를 야기했습니다.

- 암복호화 쿼리 추가 발생
- 암호화 컬럼를 누락할 가능성 높음
- QueryDSL 사용의 제약

✔해결

- AttributeConverter와 @Convert을 사용해서 엔티티와 DB 간의 암복호화를 자동으로 수행했습니다.
- AttributeConverter에는 Dual 테이블을 활용하여 DB의 암복호화 함수의 결과값을 반환했습니다.

✔결과

- 기존에 발생하던 문제가 개선되었고, 프로젝트의 유지 보수성도 향상되었습니다.
- @Converter는 엔티티 또는 엔티티를 통한 컬럼 조회 시에만 동작한다는 점을 알게 되었습니다.

술내음

삼성 청년 SW 아카데미 | 2023.02-2023.04(7주)

우리 술을 알리고, 취향에 맞는 전통주를 추천해주는 사용자 맞춤 전통주 추천 서비스

팀원 총 6명(Backend3명/Frontend3명)

성과 우수상 수여, 600명의 사용자 경험

링크 https://github.com/SSAFY707/SULNAEEUM

기술

- **Backend**
  SpringBoot | Java | JPA | MySQL | RabbitMQ | SpringSecurity
- **DevOps**
  AWS EC2 | S3 | Jenkins | Nginx | Docker | DockerCompose

아키텍처

담당 업무

- 인프라 구축
- JWT 기반 소셜 로그인
- User RestAPI 구현

주요기여

- Docker/컨테이너 기반의 서비스 배포
- Jenkins 기반의 CI/CD구축
- 무중단 배포
- JWT 기반의 소셜 로그인 구현

대표 트러블 슈팅

1 Blue-Green 무중단 배포

기술 블로그 링크) https://bellejoie.tistory.com/31

✔문제

서비스의 새로운 버전을 배포할 때 서비스의 다운타임이 발생했습니다.

✔해결

- Blue-Green 방식의 무중단 배포를 구현했습니다.
- Nginx에서 upstream 서버를 설정하여 새롭게 배포된 서버로 로드 밸런싱을 수행했습니다.
- 추가로 GitLab과 Jenkins를 연결하여 CI/CD 환경을 자동화했습니다.

✔결과

- 서비스를 다운타임 없이 지속적으로 사용할 수 있도록 개선했습니다.

술내음

삼성 청년 SW 아카데미 | 2023.02-2023.04(7주)

우리 술을 알리고, 취향에 맞는 전통주를 추천해주는 사용자 맞춤 전통주 추천 서비스

대표 트러블 슈팅

2 CORS 오류

✔문제

로컬 환경의 프론트에서 배포된 백엔드 서버로 API를 요청하는 과정에서 CORS 오류가 발생했습니다.

✔해결

- Simple Request 요청은 Springboot의 WebMvcConfigurer 파일에 와일드카드 연산자를 사용하여 모든 주소에 대한 요청을 허용했습니다.
- header에 JWT를 추가하여 발생한 OPTION 메서드 기반의 Preflight Request 요청은 SpringSecurity Config 파일에 요청 허용을 명시하여 해결했습니다.

✔결과

- 프론트엔드 개발자가 로컬에서 CORS 오류 없이 테스트할 수 있도록 개선했습니다.
- Simple Request와 Preflight Request의 차이점과, 각 요청 방식에 따라 CORS 문제를 해결하는 방법을 배웠습니다.