DEVELOPER

팀원들이 가장 신뢰하는 개발자 **박주찬** 입니다.

강한 팀워크와 협업 중심의 태도로 3회 연속 **프로젝트 베스트 멤버** 선정

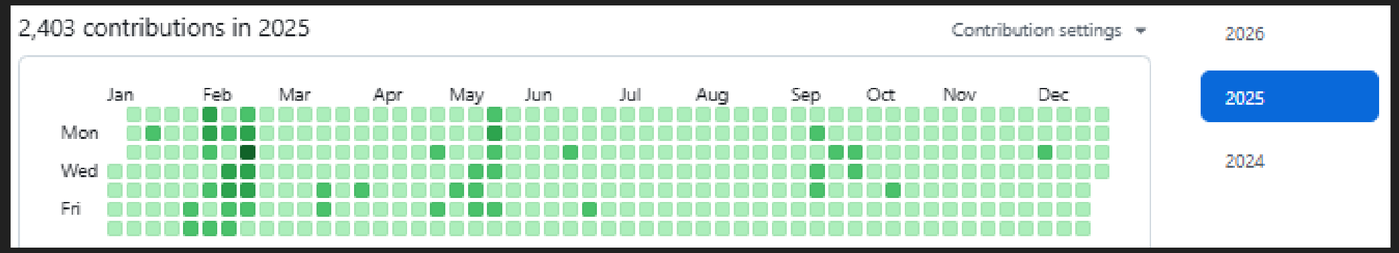
경력자들 사이에서 핵심 역할 담당

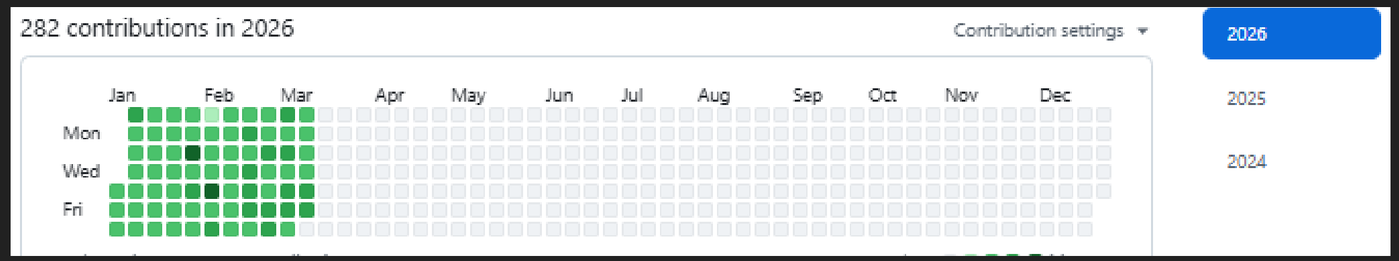
포기하지 않고 책임감있게 끝까지 일하는 **개발자 박주찬**입니다.

박주찬 PHONE: 010-8508-8650 MAIL: p990805@gmail.com Github: https://github.com/p990805

CONTENTS

ABOUT ME





2015학년도 1분기 선효 행모범상(선행부문)		2015.05.18.

표창장(모범학생 부문)		2017.08.23.

[공통PJT 베스트멤버]에 선정되신 것을 축하드립니다.
특화PJT 기간동안도 Best하게 보낼 여러분의 SSAFY 생활을 항상 응원하겠습니다❤️

[특화PJT 베스트멤버]로 선정되신 분들께 전달되는 기프티콘입니다.
남은 자율PJT도 BEST하게 보내실 여러분을 항상 응원하겠습니다.

[자율PJT 베스트멤버]로 선정되심을 축하드립니다.
앞으로도 여러분의 BEST한 앞날을 언제나 응원하겠습니다.

안녕하세요.
모범, 우수, 열정의 아이콘 박주찬입니다.

안녕하세요! 어디에서든 최선을 다해온 개발자, 박주찬입니다.
저는 소속된 곳에서 항상 책임감을 가지고 임해왔습니다.
그 결과 고등학교 시절부터 모범상과 표창장을 받아왔고,
SSAFY에서는 모든 프로젝트에서 베스트 멤버로 선정되었습니다.
혼자 잘하는 것이 아니라, 팀 안에서 분위기를 이끌고 함께 성과를 만들어 내는 것이 저의 강점입니다.

저는 Java와 Spring Boot를 중심으로 백엔드 개발에 깊은 관심을 가지고 있으며, 관련 부트캠프와 강의에 적극적으로 참여하고 있습니다.
개발 감각을 잃지 않기 위해 매일 코드를 작성하고 Github에 기록하는 것을 꾸준히 이어가고 있습니다.

Park JuChan

박주찬 (1999. 08. 05)

EDUCATION

2024. 02 신한대학교 치기공학과 졸업
2018. 03 신한대학교 치기공학과 입학
2018. 02 서울 문현 고등학교 졸업

AWARD

2025. 12 스파르타 최종 프로젝트 우수상
2025. 04 SSAFY 특화 프로젝트 우수상

EXPERIENCE

2025. 09 ~ 2025. 12 팀 스파르타 자바 단기 심화 과정(4기)
2024. 07 ~ 2025. 06 삼성 청년 SW · AI 아카데미 (SSAFY 12기)

LICENSE

2026. 03. 27 SQLD
2025. 12. 24 정보처리 기사
2023. 12. 18 지게차 기능사 (3톤 이상)
2021. 01. 21 운전 면허증 1종 보통

CONTACT

전화 010-8508-8650
메일 p990805@naver.com
주소 서울 특별시 송파구

SKILL

주로 사용하고 있습니다.

Java
Spring boot
JPA
Docker
MySQL
PostgreSQL
Git
React(vite JS)

사용해본 경험이 있습니다.

TypeScript
Kafka
Redis
PyQT5
AWS
Python
Pandas
Vue.js

개인 프로젝트
(PROTHSYNC)

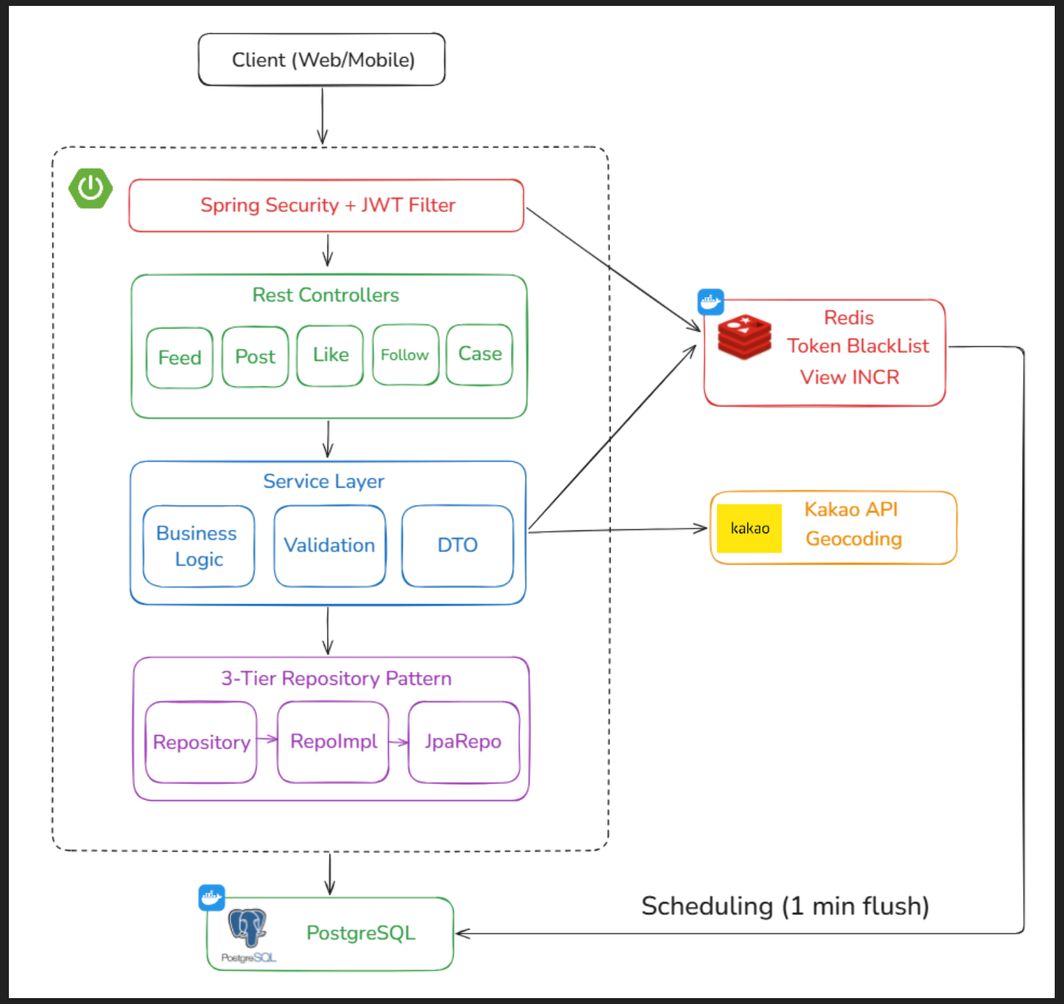
2026.01.26 ~ 03

치과 산업 종사자를 위한 전문가 네트워킹 & 외주 매칭 플랫폼
치과의사·기공소·기공사·학생 등 치과 산업 종사자들이 작업물을 공유하고
(Instagram-style 피드), 기공물 제작 외주를 의뢰·수주하며(크몽/숨고식 매칭),
위치 기반으로 주변 의뢰·사업체를 탐색할 수 있는 종합 플랫폼입니다

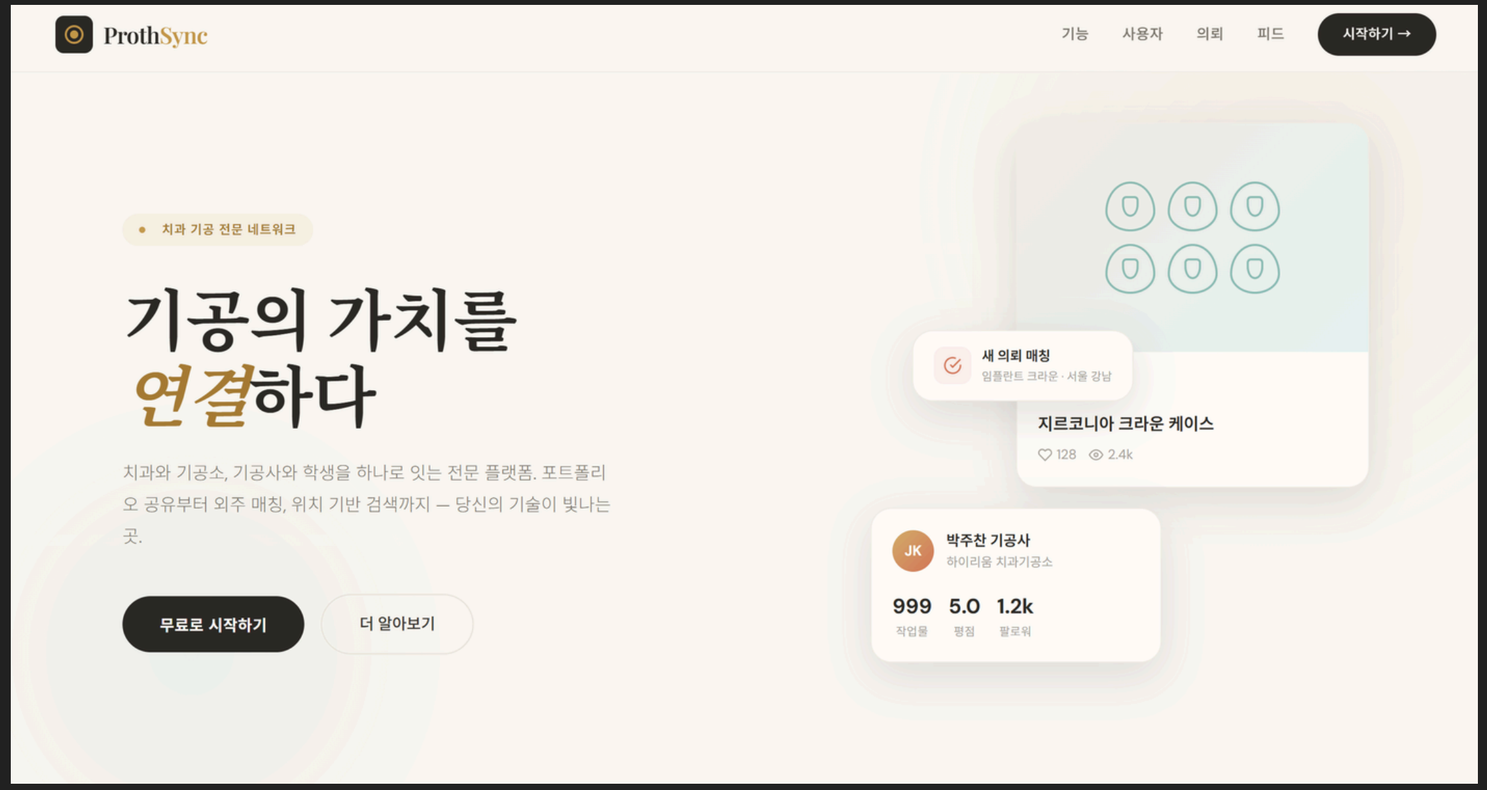
ProthSync (개인 프로젝트)

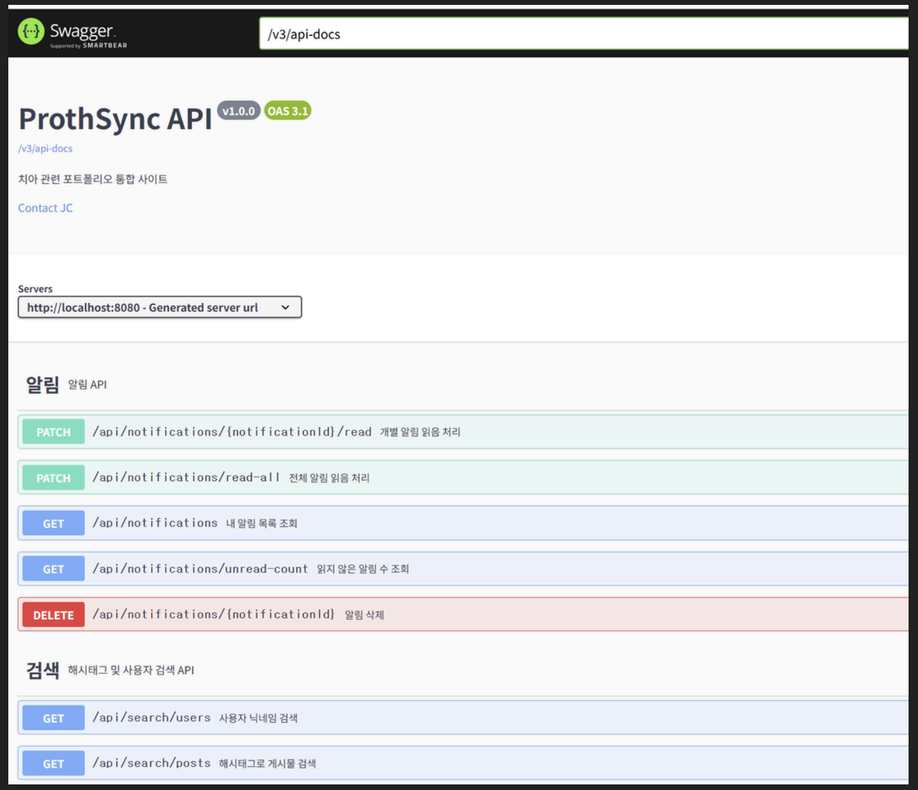
기술 스택

분류	기술
Language	Java17
Framework	Spring Boot 3.5
Security	Spring Security, JWT
Database	PostgreSQL, Spring Data JPA
Cache	Redis
Infra	Docker Compose
Build	Gradle

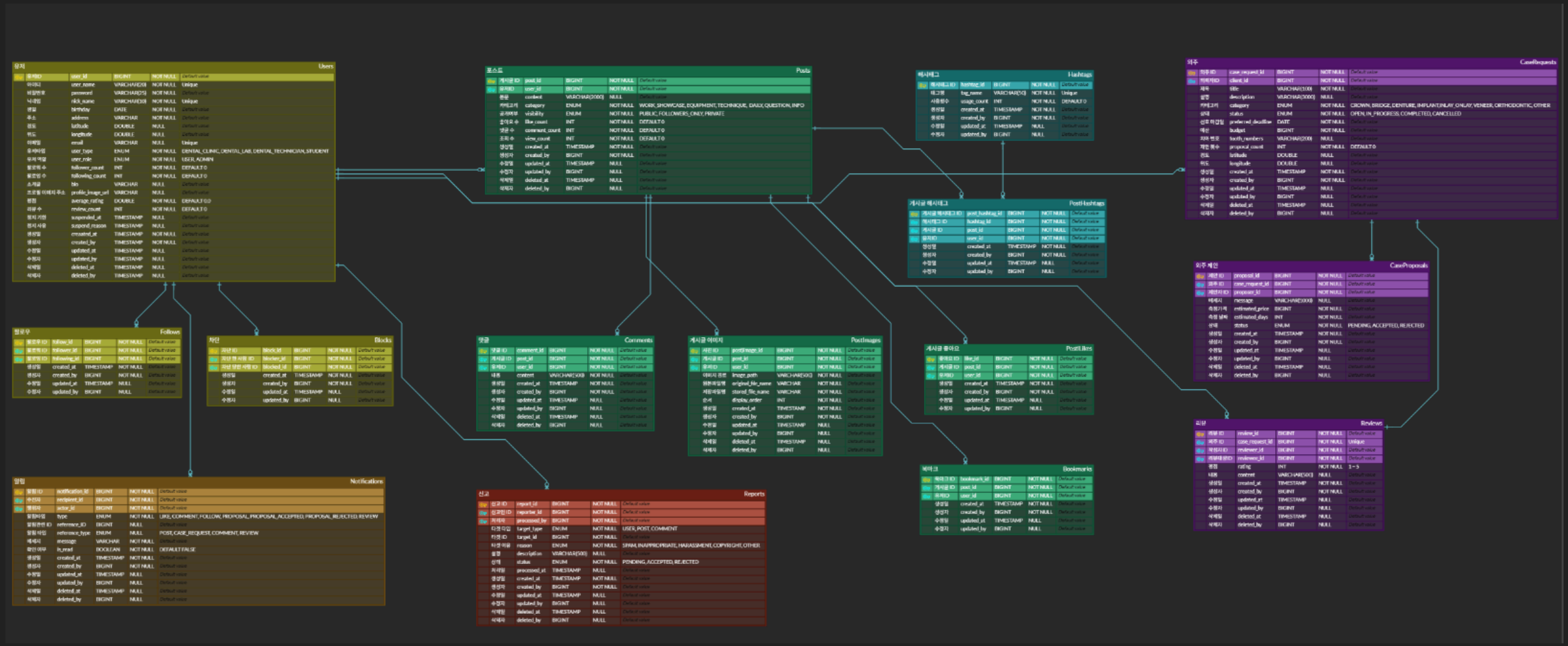


ProthSync (개인 프로젝트)





ProthSync (개인 프로젝트)



ProthSync (개인 프로젝트)

핵심 기능

1. **Kakao Geocoding API를 통한 주소-> 위도,경도 변환**
2. **하버사인 공식을 이용한 주변 외주/ 사업체 위치 검색**
3. **인스타 기반 게시글 (이미지 다중 첨부, 카테고리, 해시태그) 소셜 피드**
4. **유저 관리, 신고 처리, 정지 관리 관련 관리자 대시보드**

ProthSync (개인 프로젝트)

설계 방향

1. **단방향 ID 참조 아키텍처**
   - JPA 연관관계 대신 ID 참조로 엔티티 간 결합도 최소화

2. **3-tier 레포지토리 패턴으로 데이터 접근 추상화**
   - Service가 JPA에 직접 의존하지 않도록 추상화 계층 분리

3. **피드 모델: Fan-out on Read(Pull Model) 선택**
   - 읽기 시점에 팔로잉 기반 피드를 조립하는 Pull 방식 채택

4. **Redis Write- Behind 패턴으로 조회수 처리**
   - Redis INCR 버퍼링 + 스케줄러 일괄 flush로 DB 부하 감소

5. **엔티티별 삭제 전략 분리(Soft Delete Vs Hard Delete)**
   - 비즈니스 엔티티는 Soft Delete, 토글 엔티티는 Hard Delete

ProthSync (개인 프로젝트)

트러블 슈팅1 : 피드 조회 시 N+1 문제 해결

문제 상황

- 팔로잉 피드, 북마크 목록 등을 조회할 때 게시글 수(N)에 비례하여 쿼리가 급증하는 문제를 발견하였습니다.
- 게시글 10건 조회 시 최대 **41회 이상의 쿼리**가 실행됐습니다.

문제 원인

- 게시글 조회한 후 각 게시글마다 이미지,해시태그,좋아요 여부, 북마크 여부를 개별적으로 조회하는 구조(4N+1) 였습니다.

```
게시글 N건 기준 총 4N+1 쿼리
posts = postRepository.findFeedPosts(followingIds) // 1회
for each post:
postImageService.getImages(post.id) // N회
hashtagService.getHashtags(post.id) // N회
postLikeRepository.existsLike(userId, post.id) // N회
bookmarkRepository.existsBookmark(userId, post.id) // N회
```

문제 해결

- 배치 IN 쿼리 + Map 룩업 패턴으로 전환 게시글 ID를 먼저 수집한 뒤 연관 데이터를 한 번에 조회하고, Map<Long, T>로 변환하여 O(1) 조회 시간이 나도록 변경

```
List<Long> postIds = posts.stream().map(Post::getPostId).toList();
Map<Long, List<PostImageResponseDTO>> imageMap  = postImageService.getImagesByPostIds(postIds); // 1회
Map<Long, List<HashtagResponseDTO>> hashtagMap  = hashtagService.getHashtagsByPostIds(postIds); // 1회
Set<Long> likedPostIds = new HashSet<>( postLikeRepository.findLikedPostIds(userId, postIds)); // 1회
Set<Long> bookmarkedPostIds = new HashSet<>( bookmarkRepository.findBookmarkedPostIds(userId, postIds)); // 1회
```

항목	Before	After
게시글 10건 조회 시 쿼리 수	41회 (4N +1)	5회 (고정)
게시글 100건 조회 시 쿼리 수	401회	5회(고정)

ProthSync (개인 프로젝트)

트러블 슈팅2 : 조회 수 처리: 매 요청 DB Update → Redis Write-Behind 패턴

문제상황

- 게시글 상세 조회 API를 호출할 때마다 조회수 UPDATE 쿼리가 발생하여, 인기 게시글에 트래픽이 집중되면 동일 행에 대한 DB **UPDATE 경합(Lock Contention)**이 발생했습니다.

문제 원인

- 조회 API 내부에서 **조회수 증가를 즉시 DB에 반영하는 구조**였기 때문에, 읽기 전용이어야 할 조회 API가 쓰기 트랜잭션을 사용했습니다.
- 1초에 **100명이 같은 게시글을 보면 100번의 UPDATE가 같은 행에 경합**하게 되어 DB 병목이 발생했습니다.

문제 해결

- Redis INCR로 조회수를 버퍼링하고, 스케줄러가 주기적으로 DB에 일괄 반영하는 Write-Behind 패턴 도입하였습니다.

항목	Before	**After**
조회 1건당 DB 쓰기	1회 UPDATE	0회 (Redis INCR만)
DB 쓰기 빈도	요청 당 1회	1분당 1회
조회 API 트랜잭션	읽기 + 쓰기 혼합	순수 읽기

팀 프로젝트
(KLP LOGISTICS)

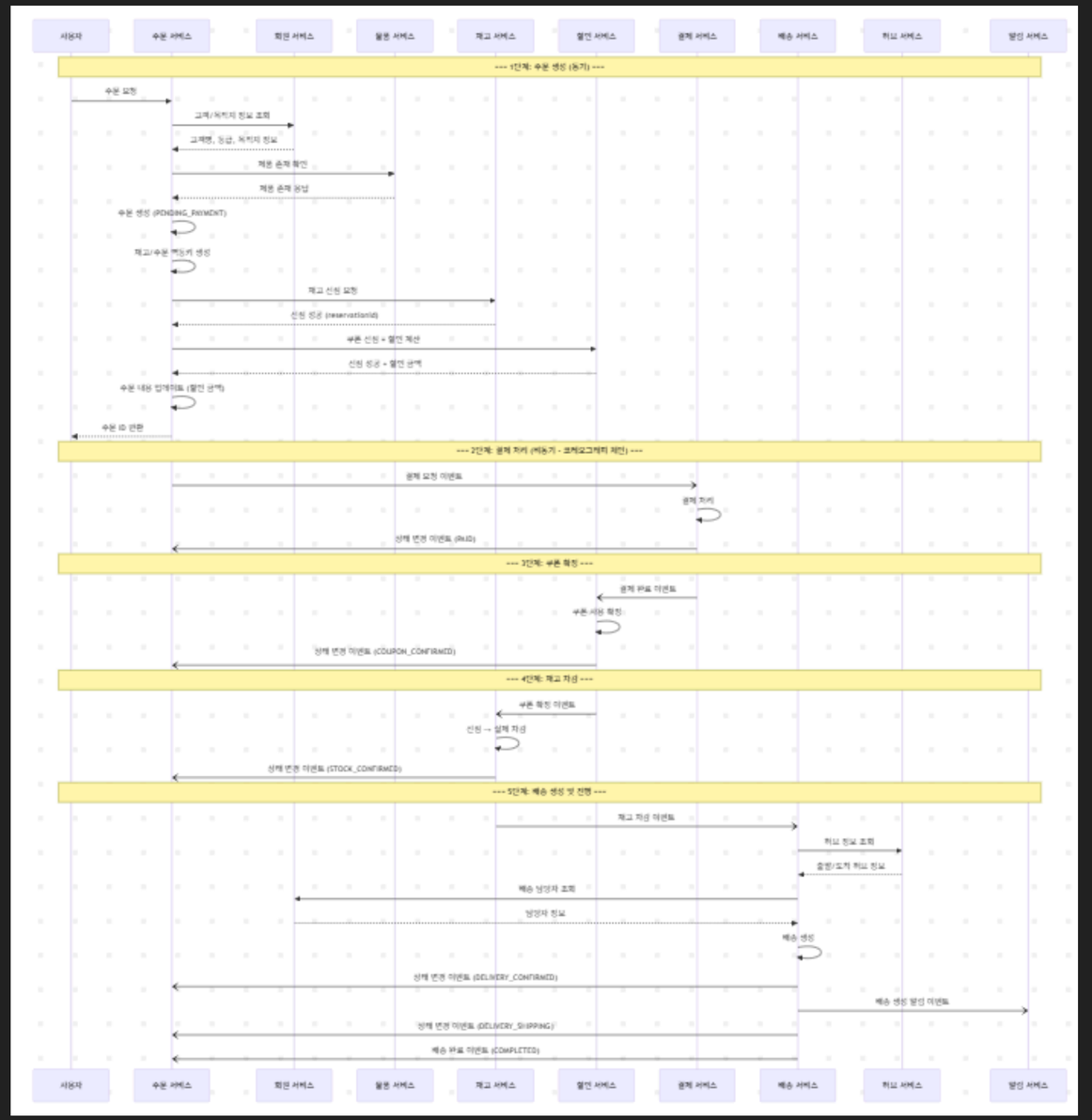
2025.10.21 ~ 12.30

B2B 물류 흐름을 기반으로 허브 간 배송 시스템을 설계하고, B2C 영역까지 확장하여 고객이 원하는 상품을 신속하게 주문·배송할 수 있는 서비스입니다.
물류 플랫폼 MSA 기반 주문 관리 시스템이며 주문 생성 부터 재고 확인, 결제, 쿠폰, 배송, 알림 까지 이벤트 기반 아키텍처 입니다.

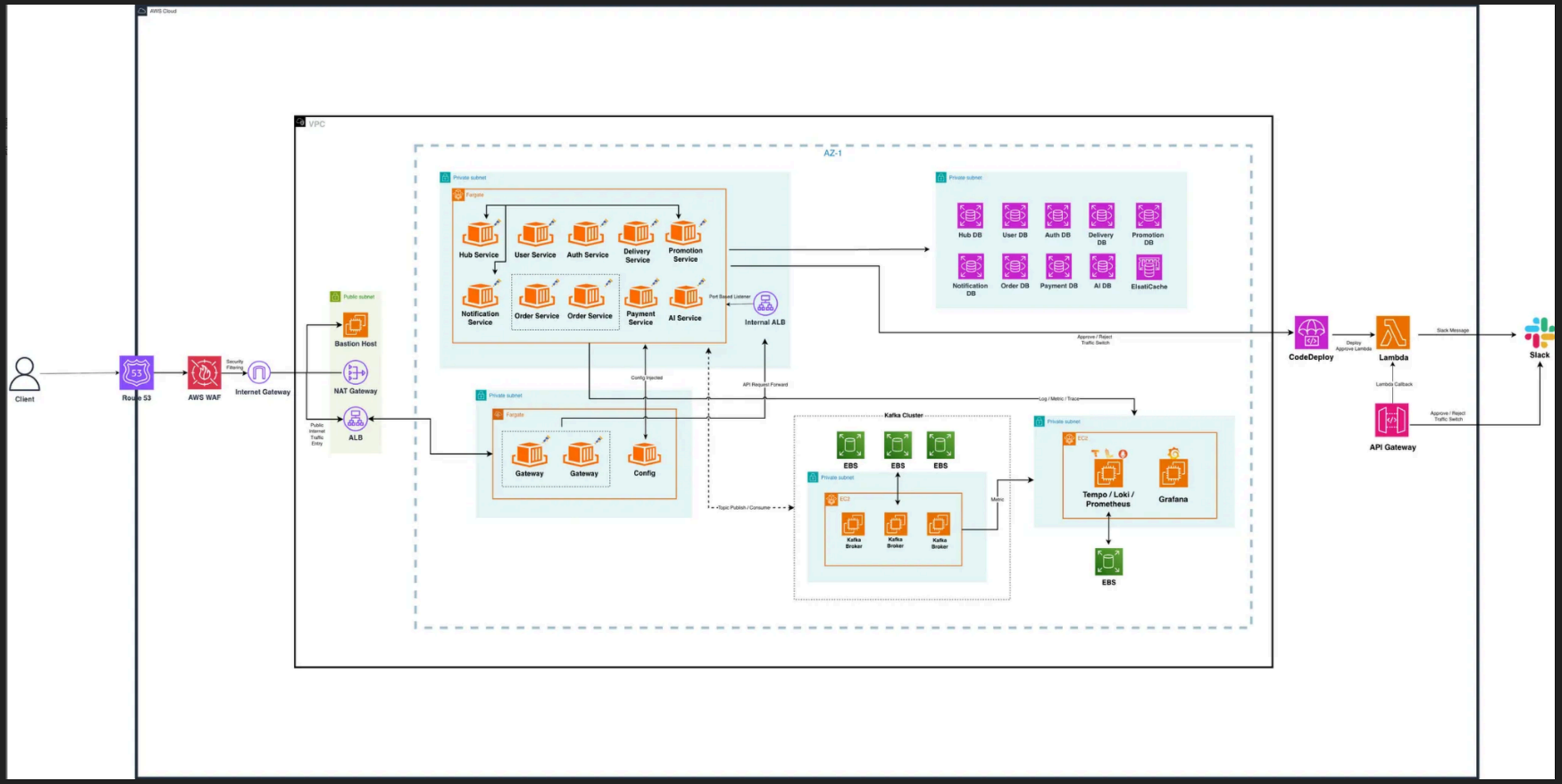
KLP 물류 프로젝트 (팀 프로젝트, 6명, Sparta 자바심화 4기)

기술 스택

분류	기술
Language	Java17
Framework	Spring Boot 3
Messaging	Apache Kafka(Event Driven Architecture)
Security	Spring Security, JWT
Database	PostgreSQL(Schema Separation), Spring Data JPA
Cache	Redis
Infra	Docker Compose, Eureka Service Discovery
Build	Gradle

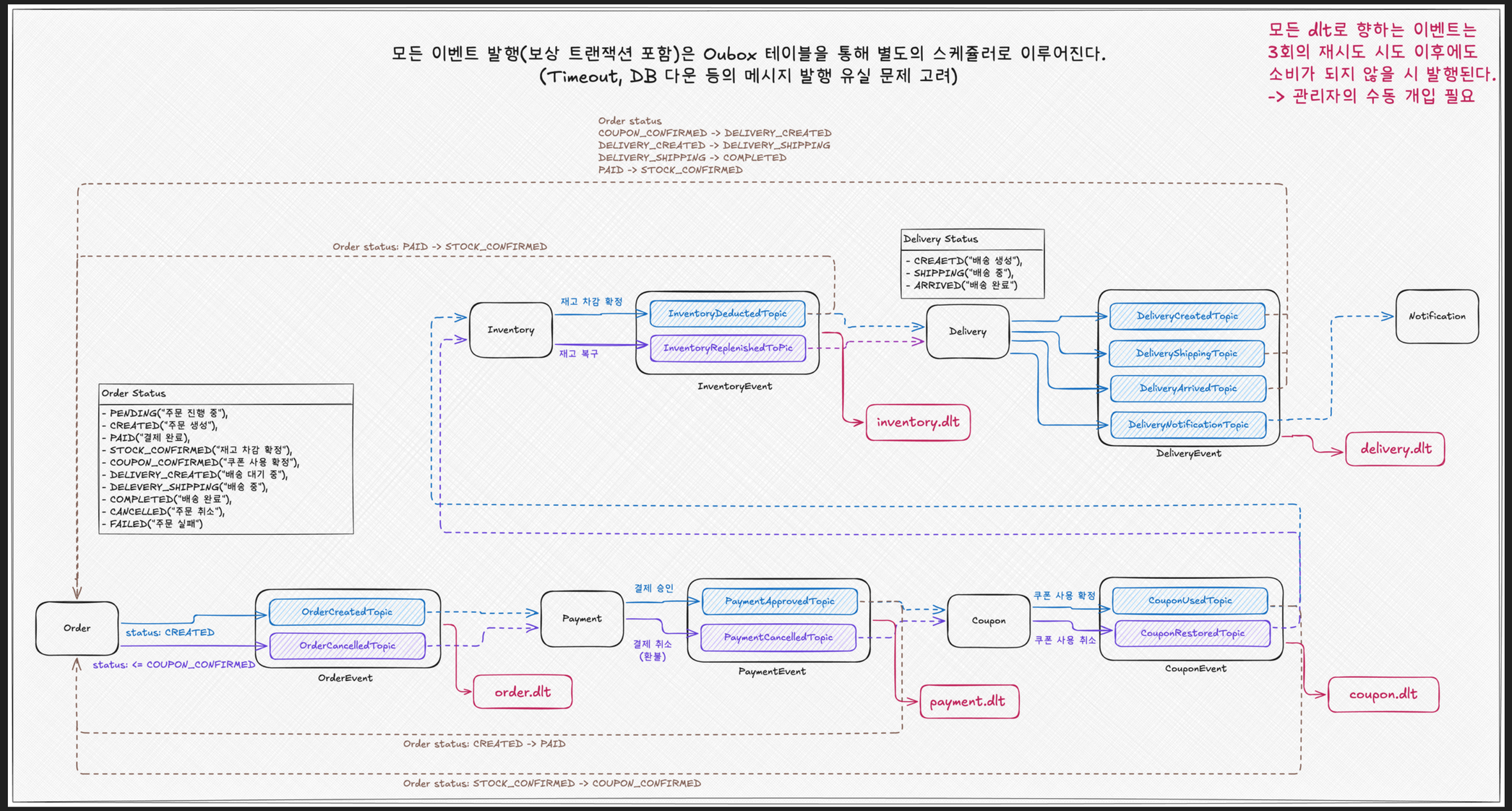


KLP 물류 프로젝트 (팀 프로젝트, 6명, Sparta 자바심화 4기)



KLP 물류 프로젝트 (팀 프로젝트, 6명, Sparta 자바심화 4기)

KLP 물류 프로젝트 (팀 프로젝트, 6명, Sparta 자바심화 4기)



KLP 물류 프로젝트 (팀 프로젝트, 6명, Sparta 자바심화 4기)

내가 한 일

1. **주문 도메인 개발**
   - 주문 생성/수정/취소/삭제 CRUD API 구헌 및 Swagger 문서화
   - Facade 패턴 적용으로 복잡한 주문 생성 과정을 오케스트레이션

2. **FeignClient → Kafka Event-Driven Architecture 전환 담당**
   - 5개의 마이크로 서비스 간 비동기 이벤트 통신 체계를 설계 및 구현
   - @RetryableTopic + DLT 패턴으로 이벤트 소비 재시도 및 Dead Letter 처리
   - @KafkaHandler로 하나의 Listener에서 성공/실패 이벤트를 타입별로 분기 처리

KLP 물류 프로젝트 (팀 프로젝트, 6명, Sparta 자바심화 4기)

내가 한 일

3. **Transactional Outbox pattern**
   - Kafka 메시지 발행의 신뢰성을 보장하기 위해 Transactional Outbox Pattern 구현
   - Exponential BackOff + Jitter로 1초 ->2초->..->5분 으로 무분별한 재시도 방지
   - Stuck Event Recovery : 30초마다 실행하여 3분 이상 Publishing 상태인 이벤트 복구

4. **Saga Pattern (ChoreoGraphy + Orchestration)**
   - 분산 트랜잭션의 일관성을 보장하기 위해 Saga 패턴을 구현했습니다.
   - 정방향 트랜잭션은 Choreography로 보상 트랜잭션은 Orchestration 방식으로 실행
   - 정방향 : 주문 → 재고/쿠폰 선점→ 결제 → 재고/쿠폰 확정→ 배송→ 알림
   - 보상 : 주문 취소 → 재고 복원 + 결제 취소 + 쿠폰 복원

ProthSync (개인 프로젝트)

설계 방향

1. 이벤트 기반 아키텍처를 선택한 이유

- **동기 방식**에서는 연계 서비스가 추가될 때 마다 주문 확정까지의 응답 시간이 늦어졌습니다.
- 주문 생성과는 전혀 관계 없는 알림이 실패하더라도 주문 확정에 실패하는 문제가 있었습니다.
- 그래서 주문 생성이라는 책임에만 집중하고, 후속 처리는 **비동기 이벤트**로 처리하여 서비스 간 결합도를 낮추고 개별 서비스의 장애가 전파되지 않도록 설계 했습니다.

2. order.topic VS order.created → order.created

- **이벤트별 개별 토픽 방식**을 선택했습니다.
- 각 서비스가 필요한 토픽만 구독하여 불필요한 메세지 소비를 줄이고 DLT에서 불필요한 이벤트 처리를 방지하였습니다.

ProthSync (개인 프로젝트)

설계 방향

3. Outbox Pattern을 도입한 이유

- Outbox Pattern이 없이는 이벤트 발행 후 소비 서비스가 장애로 다운되거나 Kafka 자체에 문제가 생기면 **메세지가 소실**되어 **서비스 간 데이터 일관성이 깨지는 문제** 발생하였습니다.
- Outbox Pattern을 도입하여 비즈니스 로직과 이벤트 저장을 하나의 DB 트랜잭션으로 묶어 원자성을 보장하였습니다.
- **Exponentiial Backoff + Jitter를 적용하여 Kafka 장애 시 무분별한 재시도로 인한 시스템 부하를 방지**했습니다.

4. Saga Pattern 중 하나로 통일하지 않은 이유

- 주문 생성 전 반드시 성공해야 하는 필수 작업이 존재하는데 이를 **동기식으로 실패하면 주문을 생산하지 않아 불필요한 보상 트랜잭션을 줄일 수 있고**, 후속작업은 각 서비스가 이벤트를 발행/ 구독하여 자율적으로 처리하게 하여 서비스 결합도를 낮췄습니다.

KLP 물류 프로젝트 (팀 프로젝트, 6명, Sparta 자바심화 4기)

트러블 슈팅1 : 이벤트를 Record로 만들어 발생한 역직렬화 실패

문제상황

- 이벤트 객체의 불변성을 보장하기 위해 Java Record로 생성했습니다.
- 그런데 Kafka로 이벤트를 수신할 때 **Cannot construct instance (no Creators, like default constructor, exist)** 에러가 발생했습니다.

문제 원인

- Jackson은 기본적으로 기본 생성자로 빈 객체를 만든 후 setter로 필드를 주입하는 방식으로 역직렬화 합니다.
- 하지만 Record는 기본 생성자와 Setter가 없고 Java는 컴파일 할때 생성자의 파라미터 값을 보존하지 않아 Jackson이 JSON의 어떤 필드를 생성자의 어떤 파라미터에 넣어야 할지 알 수 없어 역질렬화에 실패했습니다.

문제 해결

- JacksonConfig라는 파일을 만들어 **ParameterNamesModule을 등록**하여 Jackson이 생성자 파라미터 이름을 읽을 수 있도록 설정하였습니다.
- build.gradle에 **-parameters 컴파일 옵션을 추가**하여 컴파일 시 파라미터 이름이 바이트코드에 보존되도록 햇습니다.
- 이를 통해 Record의 생성자 파라미터를 통한 역직렬화가 가능해졌습니다.

KLP 물류 프로젝트 (팀 프로젝트, 6명, Sparta 자바심화 4기)

트러블 슈팅2 : N+1 쿼리 성능 저하

문제상황

- K6 부하 테스트에서 주문 목록 조회 API의 응답 시간이 비정상적으로 높게 측정되었습니다.
- 페이지당 20건 조회 시 DB 쿼리가 21회 이상 발생하고 있었습니다.

문제 원인

- Order와 OrderItem이 OneToMany 관계인데, 조회 메서드에서 Spring Data JPA의 기본 메서드를 사용하고 있었습니다.
- 기본 메서드는 Order만 조회하고 OrderItem은 가져오지 않기 때문에, 이후 응답을 만들 때 order.getOrderItems()를 호출하면 JPA가 Order마다 OrderItem을 조회하는 추가 쿼리를 실행하여 N+1 문제가 발생했습니다.

문제 해결

- JPQL FETCH JOIN을 적용하여 Order를 조회할 때 OrderItem을 한 번의 쿼리로 함께 가져오도록 변경했습니다.
- Page 쿼리에는 countQuery를 분리하여 카운트 시 불필요한 JOIN이 발생하지 않도록 최적화 했습니다.

KLP 물류 프로젝트 (팀 프로젝트, 6명, Sparta 자바심화 4기)

트러블 슈팅3 : 주문 생성 실패 시 보상 이벤트도 롤백되는 문제

문제상황

- 주문 생성 과정에서 재고 선점은 성공했지만 이후 단계 에서 실패했을 때, 재고 선점을 해제하기 위한 ORDER_FAILED 이벤트가 DB에 저장되지 않는 문제가 발생하였습니다.
- 재고는 이미 다른 서비스에서 선점 완료된 상태인데 실패 이벤트가 없으니 주문이 없는데 재고가 잠겨있는 데이터 불일치 상태가 발생했습니다.

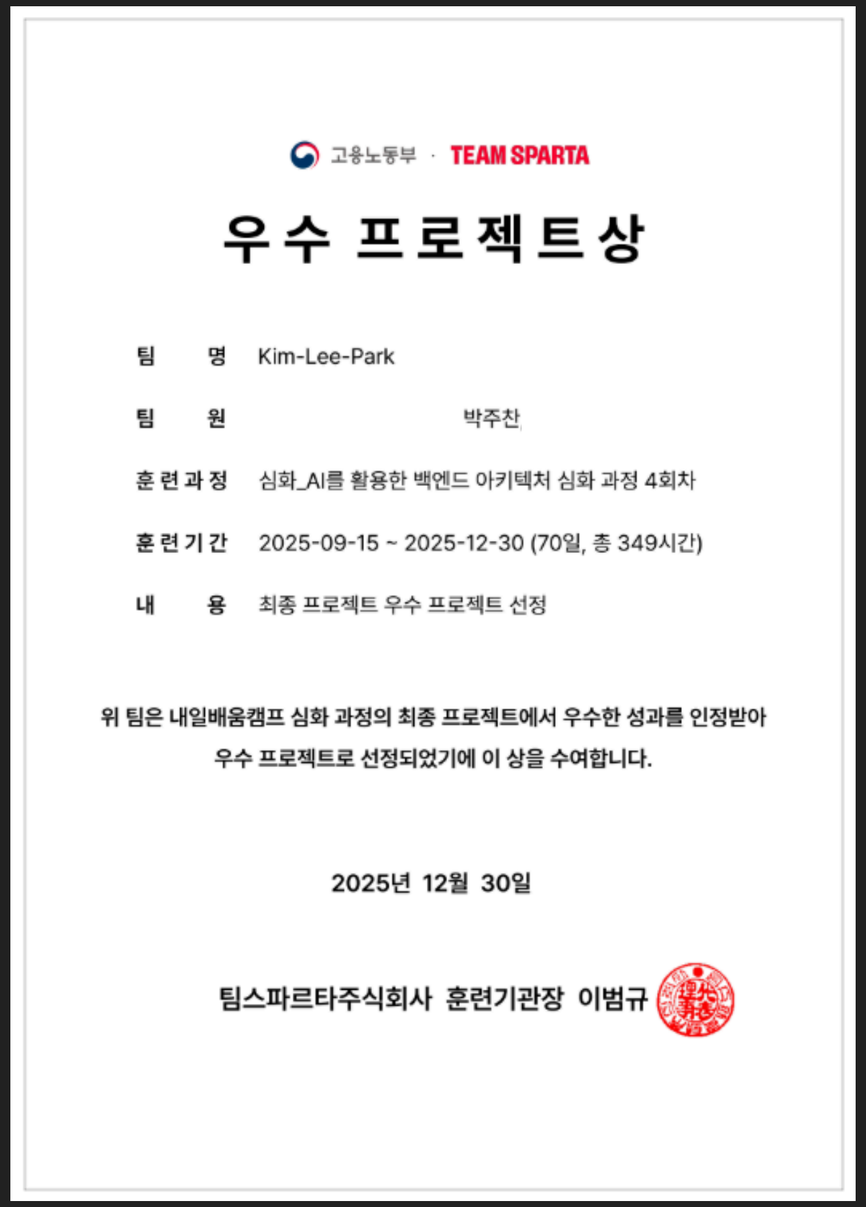
문제 원인

- createOrder() 메서드 전체가 하나의 트랜잭션에 감싸져있었습니다.
- catch문에서 ORDER_FAILED 이벤트를 outbox에 저장한 후 throw로 예외를 던지면 **전체 트랜잭션이 롤백**을 시켜서 **catch문 안에서 작성했던 이벤트**가 없어져 버리는 현상이였습니다.

문제 해결

- 실패 이벤트 저장을 위한 saveFailedEvent() 메서드를 별도로 만들었습니다.
- **@Transactional(propagation = Propagation.REQUIRES_NEW)**를 적용했습니다.
- **REQUIRES_NEW는 기존 트랜잭션을 일시 중단하고 완전히 새로운 트랜잭션을 생성하여 독립적으로 커밋**되어 메인 트랜잭션이 롤백되어도 ORDER_FAILED 이벤틑 DB에 남어있게 됩니다.

KLP 물류 프로젝트 (팀 프로젝트, 6명, Sparta 자바심화 4기)



고용노동부 · TEAM SPARTA

우수 프로젝트상

팀 명	Kim-Lee-Park
팀 원	박주찬
훈련과정	심화_AI를 활용한 백엔드 아키텍처 심화 과정 4회차
훈련기간	2025-09-15 ~ 2025-12-30 (70일, 총 349시간)
내 용	최종 프로젝트 우수 프로젝트 선정

위 팀은 내일배움캠프 심화 과정의 최종 프로젝트에서 우수한 성과를 인정받아
우수 프로젝트로 선정되었기에 이 상을 수여합니다.

2025년 12월 30일

팀스파르타주식회사 훈련기관장 이범규

Other Projects

1. 음식 주문 플랫폼(TDD)(Sparta, 6인)(09/15 ~ 10/20)

- 장바구니 담당
- 리뷰 담당

2. 삼성 전자 기업연계 프로젝트 (생산 계획 최적화 프로그램)(SSAFY,6인)(04/14 ~ 05/30)

- 팀장
- PyQT5 담당 (모든 페이지)

3. 꿈 일기 영상화 프로젝트 (우꿈다)(SSAFY,6인)(02/24 ~ 04/11)

- 프론트 담당 (모든 페이지 TailwindCSS를 이용한 디자인)
- DockerCompose + Jenkins를 이용한 CI/CD 담당
- 우수상

4. 유기동물 보호소 라이브 방송 플랫폼 (T-ara)(SSAFY,6인)(01/06 ~ 02/21)

- 팀장
- 프론트 담당 (메인 페이지, 로그인, 회원가입, 마이페이지, 예약페이지, 지도 담당, 후원가능 동물 페이지 담당)

THANK YOU.

박주찬 PHONE: 010-8508-8650 MAIL: p990805@gmail.com Github: https://github.com/p990805