| **서지민**  **Backend Engineer** | **Phone |** 010-4705-3052  **Email |** jm.seo.1017@gmail.com  **Github** | <https://github.com/Lauiee>  **Blog |** <https://velog.io/@hm9720> |
| --- | --- |

**Summary**

* **문제를 정의하고 해결**하는 과정에서 직관보다 **근거에 기반한 사고**를 중시하는 백엔드 개발자 입니다.
  + 좌석 중복 선점을 방지하기 위해 Redis 기반 분산락을 적용하고, Lettuce와 Redisson을 비교 테스트 후 **성능 수치에 근거**해 **Lettuce를 선택**하였습니다.
* **단기간에 목표를 명확히 설정하고 성과를 내는 데 강점**이 있습니다.
  + 실제로 **5주 안에 프로젝트를 완성**하는 부트캠프에 두 차례 참여하여 **모두 수상**한 경험이 있습니다.

**Experience**

**TickeTeam** 2025.04 ~ 2025.07

사용자가 응원하는 팀의 홈·원정 경기를 통합 조회하고, 티켓 예매까지 가능한 야구 티켓팅 서비스

**Tech-Stack |** Java 17**,** Spring Boot(3.3.9), Spring Data JPA, MySQL, Redis, Grafana, k6

**개발 내용 |**

* 좌석 중복 선점을 방지하기 위해 Redis 기반 분산락을 적용 [[notion](https://apricot-anchovy-f57.notion.site/240ae7033dfb8052ac2fc1bfc2fef3b9?source=copy_link)]
  + Redis 기반 분산락 구현 과정에서 Lettuce와 Redisson의 비교를 통해 Lettuce 선택
    - 구현된 좌석 선점 시스템 특성상 좌석 선점 실패 시 재시도 및 락 대기 없이 즉시 응답 반환
    - 두 클라이언트의 가장 큰 차이는 락 대기 방식에서 발생하는 Redis 부하 차이지만,   
      본 시스템은 락 대기를 하지 않기 때문에 해당 차이를 고려하지 않음
    - 라이브러리 선택 기준을 성능에 두고 비교 테스트 진행
    - VU 1000, 락 획득 시나리오, 30초 부하 테스트 결과 **Lettuce**가 근소 우위  
      (RPS: **727.68**/s vs**648.02**/s | 평균 응답 시간: **7.38**ms vs **9.06**ms)
  + 락 획득 이후, 좌석 ID 기반의 Redis 키에 사용자 선점 정보를 저장하여 선점 상태를 관리
    - 생성된 캐시의 TTL을 이용해 선점 시간을 관리
    - 선점 시간은 티켓팅 시퀀스의 예상 소요 시간(5분)과 여유 시간(2분)을 반영해 **총 7분**으로 설정.
  + 캐시 만료로 선점 정보가 사라진 좌석은, Spring Scheduler를 활용해 선점 상태를 해제하고 예매 가능 상태로 복구
  + 분산락과 캐시의 조합으로 동시성 문제를 효과적으로 제어
* Grafana k6 기반 부하 테스트를 통한 성능 개선 [[notion](https://apricot-anchovy-f57.notion.site/21eae7033dfb8087bff9d0969bb03fc4)]
  + 부하테스트 기준점
    - 최대 5,000VU 환경에서 안정적으로 실패 없이 요청을 처리하는 것에 집중
    - 최초 테스트 진행 후, 실패율 **1%**이하, 요청 성공률 **99%** 이상, RPS **200/s** 기준점으로 설정
  + 사용자의 예약 내역 조회에서 발생한 N+1 문제를 Fetch Join을 활용하여 해결
    - 예약 내역 조회 시 각 예약에 연관된 티켓 정보를 접근할 때마다 추가 쿼리가 발생하는 N+1 문제가 발생
    - 이를 해결하기 위해 연관된 티켓 정보를 한 번의 쿼리로 함께 가져올 수 있도록 **Fetch Join**을 적용
    - VU 20, 30초 부하 테스트를 통해 Fetch Join 적용 전, 후 성능 비교  
      (평균 응답 시간 **182.28ms → 101.32ms**, 약 **44.4% 개선)**
  + BCryptPasswordEncoder의 반복적인 해싱 연산으로 인해 CPU 사용량이 증가, 이로 인해 로그인 병목 현상 발생
    - 로그인 시 사용자가 입력한 비밀번호를 비교하는 과정에서 BCryptPasswordEncoder의 matches() 메서드 사용
    - matches() 메서드는 검증 과정에서 해싱 연산을 거치며, 설정된 강도에 따라 연산량 기하급수적으로 증가
    - Docker 기반 애플리케이션 서버 스케일 아웃으로 CPU 사용률 분산 및 병목 완화

| **링크**: [GitHub](https://github.com/TEAM-JOE-TICKETEAM/TickeTeam-backend) |
| --- |

**REFACTORY** 2024.12 ~ 2025.02

사용자에게 AI 코드리뷰를 제공하고, 리뷰 히스토리를 통한 통계 및 분석을 제공하는 크롬 익스텐션

**Tech-Stack |** Python,Django(5.0.6), MySQL, Celery, RabbitMQ

**개발 내용 |**

* Celery와 RabbitMQ를 활용한 코드리뷰 비동기 처리
  + PR에 포함된 변경된 파일의 수에 비례하여 코드리뷰 시간이 증가하는 문제 발생
  + 각 파일 분석 작업을 Celery Task로 분리하고, RabbitMQ를 통해 큐에 전달하여 여러 워커가 병렬로 처리하도록 구성
  + 평균 코드리뷰 시간 **70%**가량 단축하여 사용자 대기 시간 감소 및 시스템 부하 감소
* SSE를 활용한 실시간 코드리뷰 구현
  + AI 분석이 완료될 때까지 결과를 기다려야 하는 동기적 구조로 인해, 사용자의 경험이 저하되는 문제가 발생
  + SSE를 도입해 AI 분석 결과를 연결된 클라이언트에게 청크 단위로 실시간 전송하는 방식으로 변경
  + 사용자 대기 시간을 최소화하고 리뷰를 즉시 확인할 수 있도록 하여 사용자 경험 개선

| **링크**: [Medium](https://medium.com/@syt47053052/%EC%8B%A4%EB%A6%AC%EC%BD%98%EB%B0%B8%EB%A6%AC-%EC%9C%88%ED%84%B0-%EB%B6%80%ED%8A%B8%EC%BA%A0%ED%94%84-%ED%8C%80-d-%EB%A6%AC%ED%8C%A9%ED%86%A0%EB%A6%AC-9e1e6c41c940) / [GitHub](https://github.com/2024-Winter-BootCamp-TeamD) |
| --- |

**LAWBOT** 2024.06 ~ 2024.08

사용자의 계약서를 검토하고, 계약서 공유와 전자서명을 통한 비대면 계약을 제공하는 서비스

**Tech-Stack |** Python, Django(5.0.6), MySQL, Pinecone, AWS KMS

**개발 내용 |**

* RAG를 활용한 검토 성능 향상
  + 검색 증강을 위해 다양한 법령 데이터를 벡터화하고, 클라우드 벡터 DB에 저장하여 검색 가능한 형태로 구성
  + LLM에 RAG를 적용하여, 관련 법령을 기반으로 보다 정확하고 신뢰도 높은 검토 결과를 도출
* 전자서명이 포함된 계약서의 보안 강화 [[blog](https://velog.io/@hm9720/PYTHON-KMS-%EA%B3%84%EC%95%BD%EC%84%9C-%ED%8C%8C%EC%9D%BC-%EB%B3%B4%ED%98%B8%EB%A5%BC-%EC%9C%84%ED%95%9C-%EC%95%94%ED%98%B8%ED%99%94-%EB%A1%9C%EC%A7%81)]
  + AWS KMS와 Cryptography라이브러리를 활용해 전자서명이 포함된 계약서 파일을 암호화
  + AWS KMS를 통해 발급받은 암호화 키를 이용해 S3에 업로드되는 계약서 파일 암호화 진행
  + 암호화된 파일은 서버를 통한 복호화를 거쳐야만 원본 내용 확인 가능

| **링크**: [Medium](https://medium.com/@jinu.mekkong/lawbot-927a2d19b273) / [GitHub](https://github.com/2024-Summer-BootCamp-TeamK) |
| --- |

**Skills**

**Language**  | Java, Python

**Database**  | MySQL

**Library / Framework** | Django, Spring Boot

**DevOps** | Docker, Github Actions, Nginx, GCP

**Activities**

2024.06 ~ 2024.08 / **2024 하계 Techeer Silicon Valley SW Bootcamp** 수료

2024.08 ~ 현재 진행 중 / **Techeer 8기** - 실리콘밸리 개발자의 SW 개발자 커리어 그룹 멤버

2024.12 ~ 2025.02 / **2024 동계 Techeer Silicon Valley SW Bootcamp** 수료

**Education**

성결대학교

정보통신공학과 | 2019.03 - 2025.02

학점 | 4.26