Intro

고병후 Software Developer

1998. 03. 25

bhwkd@naver.com / 01067466704

"과감하게 도전하고 섬세하게 고민하는 개발자 고병후입니다."

새로운 기술, 새로운 분야라도 성과에 대한 의지와 배움에 대한 즐거움으로 과감하게 도전합니다.
학부 졸업 후에는 'SFZ'라는 개발팀을 주도적으로 조직하여 다양한 프로젝트에 도전했으며,
미디어 스트리밍 중계 서버 개발과 ARM TrustZone을 활용한 보안 도어락 개발 등 새로운 영역에서도 성공적인 결과를 만들어왔습니다.
이러한 태도로 새로운 업무가 주어져도 빠르게 습득해 성과를 내기 위해 노력하고 있습니다.

또한, 하나의 프로젝트, 기능이라도 동작원리와 개선사항을 섬세하게 고민합니다.
예를 들어, 메시징 프로토콜이 적용되지 않은 WebSocket 통신에서 토큰 인증 문제를 해결하거나,
HTTP 요청 유형에 따라 다르게 나타나는 CORS 오류문구의 원인을 파악해 블로그에 정리하는 등의 활동을 해왔습니다.
이러한 과정을 통해 한층 더 깊이 있는 기술 역량을 갖춘 개발자로 성장하고자 노력하고 있습니다

Key Skills

- **Programming Languages: Java, C++, Python**
- **Web Development: Spring Boot, Django**
- **Database Management: MySQL**
- **Infrastructure: AWS (EC2, RDS, S3)**
- **Version Control: Git, GitHub**

Experienced Skills

- **Programming Languages: C, JavaScript**
- **Web Development: HTML, CSS, React, Node.js, Express**
- **Database Management: Redis**
- **Infrastructure: AWS (ALB, CloudFront), Nginx**
- **DevOps : Docker, GitHub Actions**

Education

경희대학교 국제캠퍼스, 컴퓨터공학과

2017. 03. 01 - 2023. 08. 16

카카오 테크 부트캠프, 클라우드 네이티브 제주 1기

2024. 04. 02 - 2024. 10. 11

Certifications

정보처리기사

2024. 06. 18

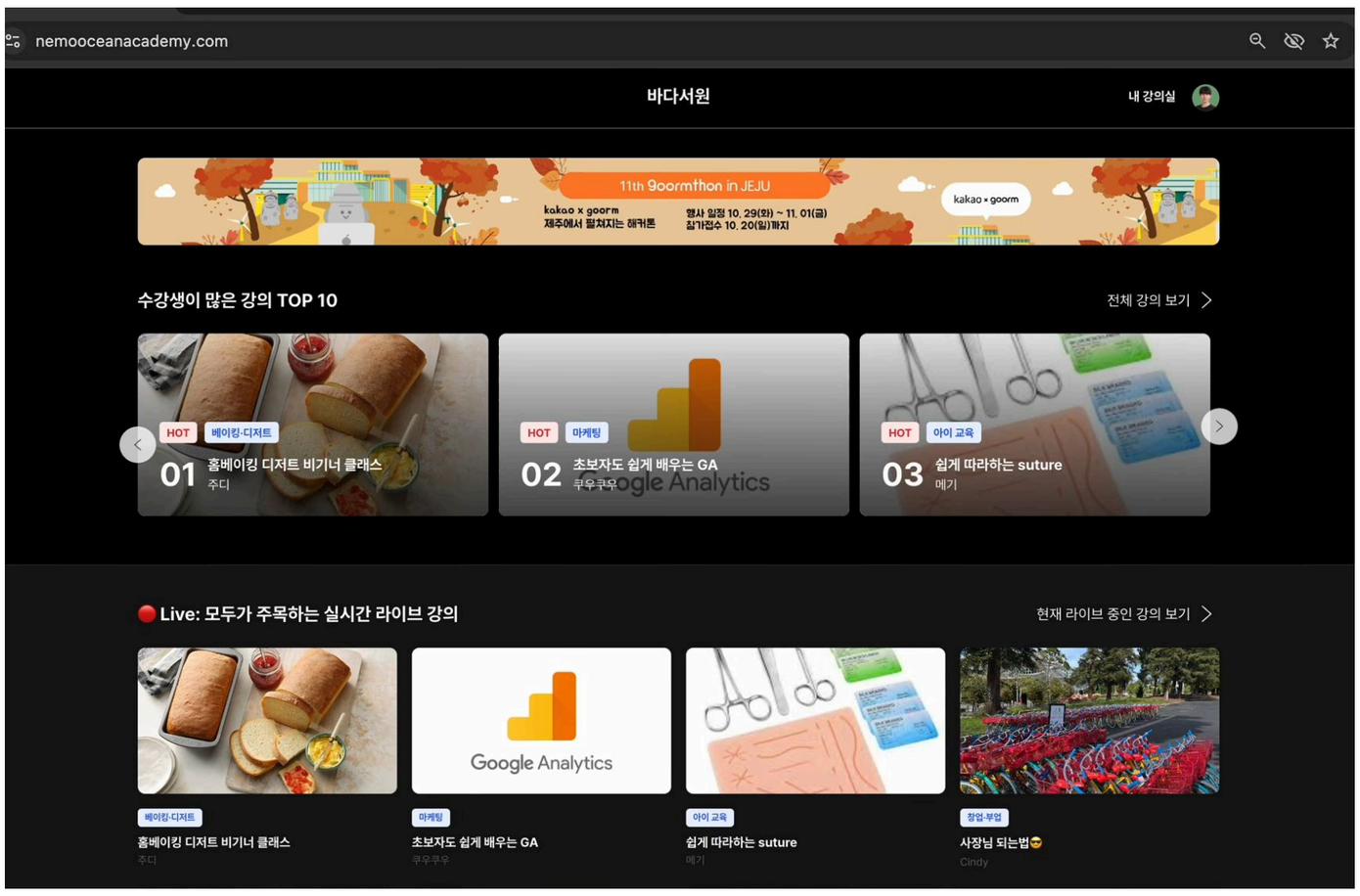
GitHub / Technical Blog

https://github.com/GoByeonghu

https://gobyeonghu.github.io/

바다서원 개인 라이브 강의 플랫폼

Project - 1



강사가 라이브 강의를 개설하고, 수강자들이 이를 수강 신청한 뒤 실시간 라이브 스트리밍 강의를 수강하는 서비스입니다.
라이브 강의에서는 강사의 웹캠 영상, 화면 공유 영상을 수강자들에게 실시간으로 송출하며, 참여자들이 실시간 채팅을 주고받을 수 있습니다.

작업 기간

2024. 07 - 2024. 10

구성원

6명 (Backend 4, Frontend 1, Design 1)

깃헙

https://github.com/GoByeonghu/5-nemo-oceanAcademy-be

기술 스택

Back-End

Spring Boot · Spring WebSocket · Spring Batch
Node.js · Express.js · Socket.IO · Mediasoup

Front-End

React · TypeScript

Database

MySQL · Redis

주요기능

- WebRTC를 이용한 실시간 스트리밍
  - 웹캠 영상, 화면 공유 영상, 마이크 음성, 화면 공유 음성 송수신
  - Mediasoup 기반 SFU구조 비디오, 오디오 스트리밍
- WebSocket를 이용한 실시간 채팅
  - Stomp 기반 실시간 채팅
- Top 10 인기강의 순위 연산
  - 최근에 만들어지고, 수강생 수가 많은 강의에 순위 점수 부여
  - 매일 03시 순위 변경 수행
- OAuth 인증(Kakao)
- 강의 CRUD
- 수강신청 기능

역할

Back-End 개발

- Spring을 이용한 REST API 개발
- 미디어 중계서버 개발
- 인기강의 순위 연산 Batch 시스템 개발
- Redis Cache 도입
- Sentry 도입을 통해 실시간 오류 감지 및 모니터링 시스템 구축

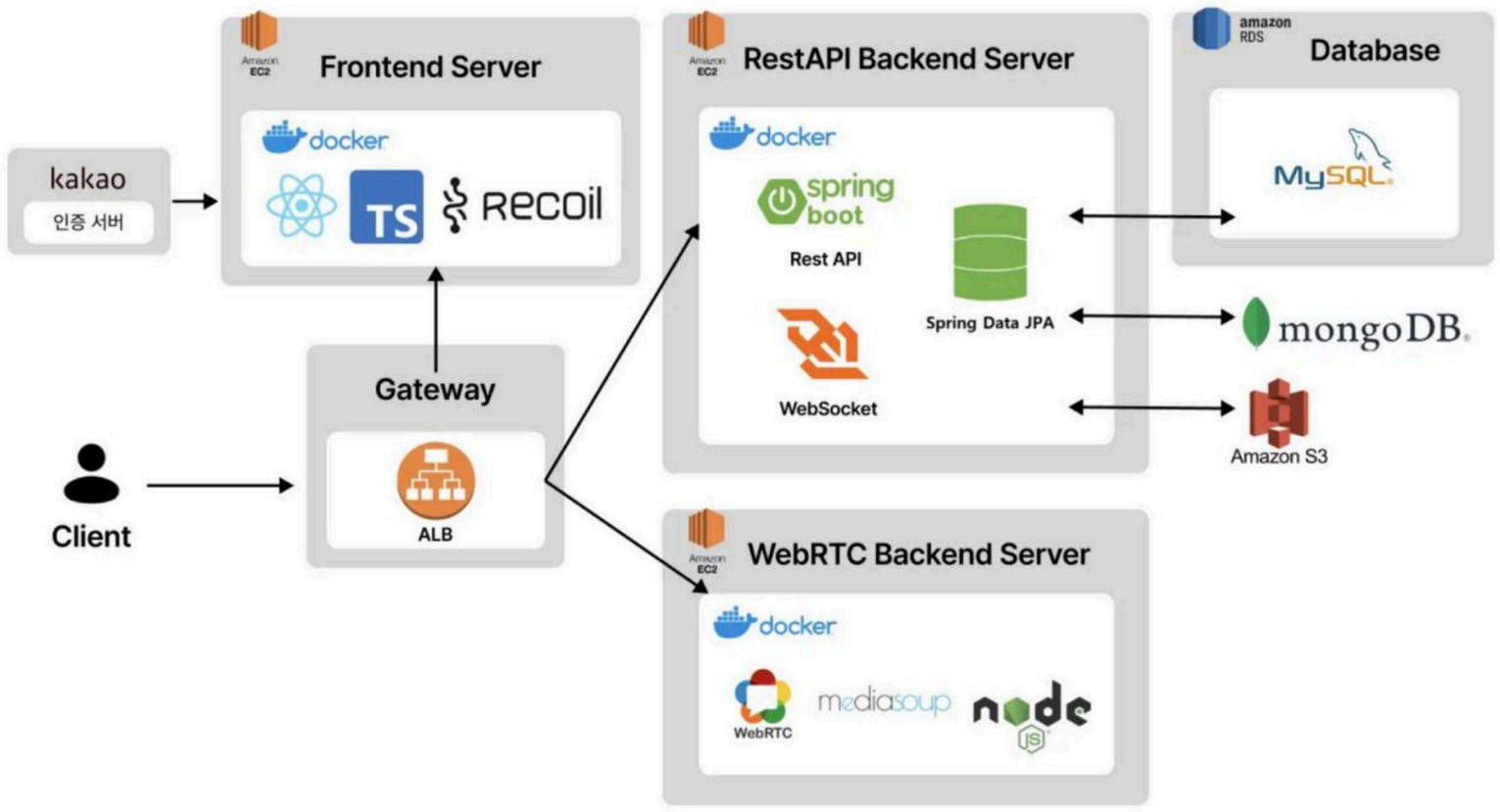
성과

- Redis Cache 도입으로 API 응답 시간 837ms >228ms 3.7배 개선
- Presigned URL 도입으로 서버의 Network I/O 50KB 기준 91% 절감
- Mediasoup 중계서버 도입으로 동시 참여 유저 수 11.4배 증가

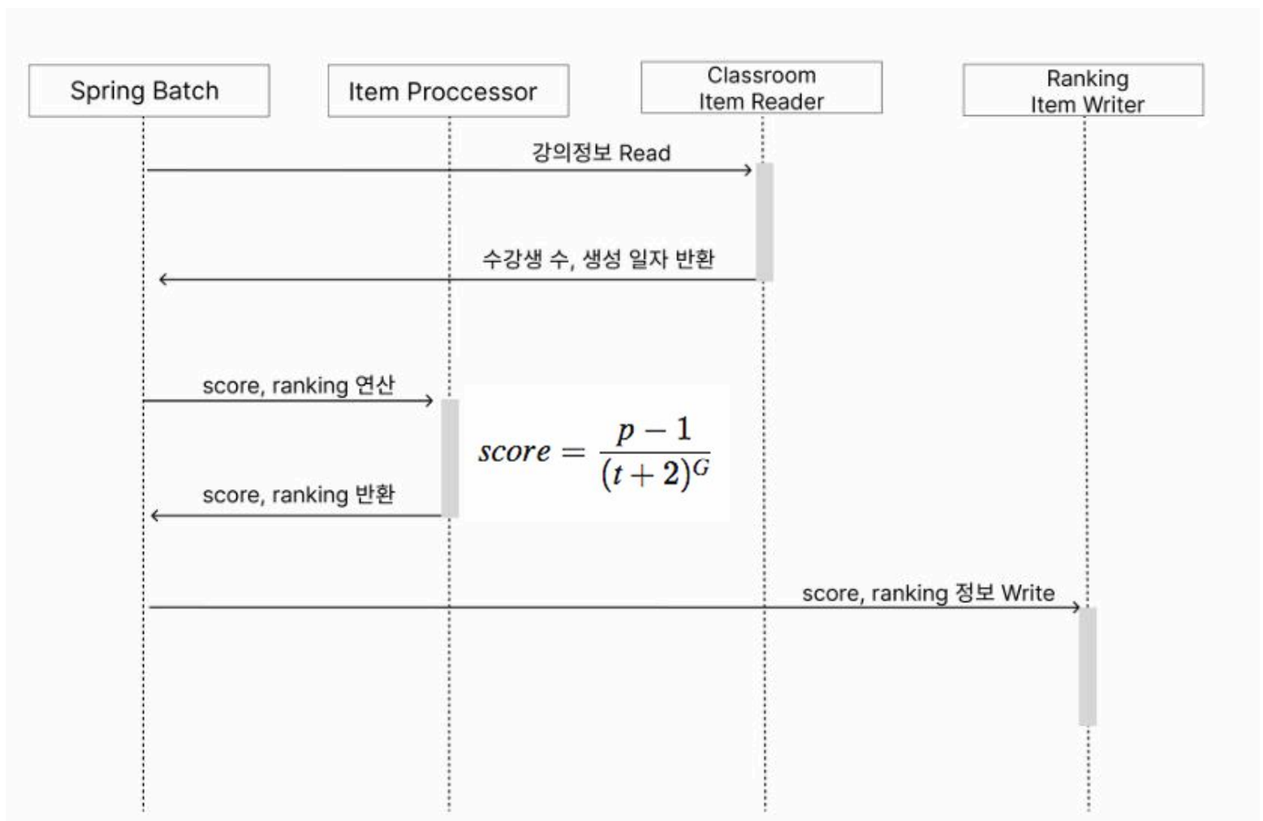
바다서원 개인 라이브 강의 플랫폼

Project - 1

Architecture



인기강의 순위 연산 Batch 시스템 개발



배치 시스템 설계

- 하루 한 번 강의들의 인기 score를 연산하여 인기 순위를 도출하는 배치 시스템을 개발하였습니다.
  - 트래픽이 적은 03시에 일괄 작업을 수행하기 위해 Spring Scheduler를 사용하였습니다.
  - 모든 Class 테이블의 레코드에 대해 연산을 수행할 때, 트랜잭션을 보장하고 재시도를 보장하기 위해 Spring Batch를 사용하였습니다.
- MySQL의 index와 Redis를 활용하여 score 연산 결과 기반 인기 순위 측정 오버헤드를 감소시켰습니다.
  - 강의 레코드의 score 컬럼에 index를 설정하여 ORDER BY에서 full scan을 방지하였습니다.
  - Redis List에 순위를 캐싱하여 LPUSH로 저장 후 LRANGE 0 9로 가져오므로서 순위 조회 성능을 향상시켰습니다.

인기순위 알고리즘 (Popularity Ranking Algorithm)

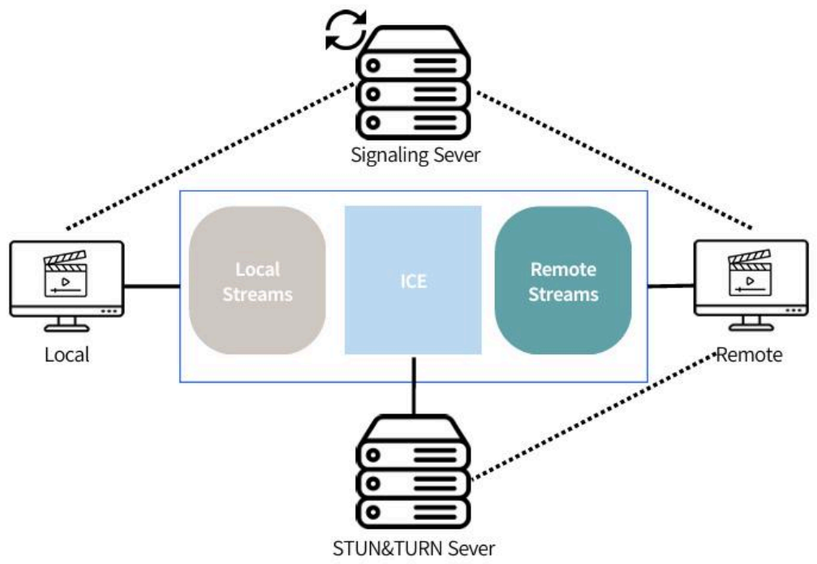
- 최근에 개설된 강의와 수강생 수가 많은 강의일수록 높은 순위를 부여하는 알고리즘을 적용하였습니다.
  - Hacker News Algorithm를 적용하여 수강생 수가 많을수록 높은 점수를 부여하였습니다.
  - 또한, Aging 기법을 적용하여 시간이 지날수록 점수가 감소하도록 하여,
    수강생 수가 극단적으로 많은 하나의 강의가 순위를 독점하는 것을 방지하였습니다.
  - score = (P - 1) / (t + 2)^G 수식을 적용하였으며,
    p: 게시물에 대한 호감도(수강생 수), t: 게시물이 게시된 시간과 현재 시간 사이의 차이, G: 중력 계수(1.8)을 적용하였습니다.

바다서원 개인 라이브 강의 플랫폼

Project - 1

미디어 중계서버 도입

영상 및 오디오 스트리밍에 WebRTC 사용

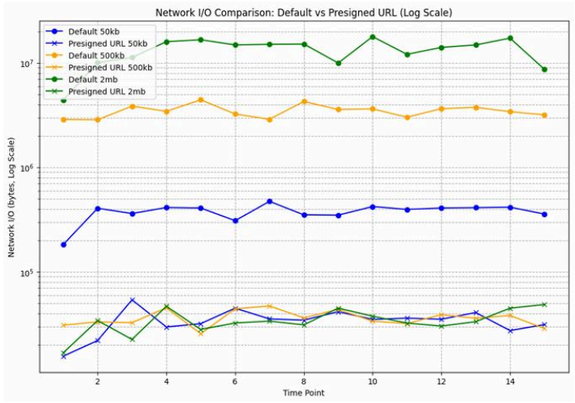


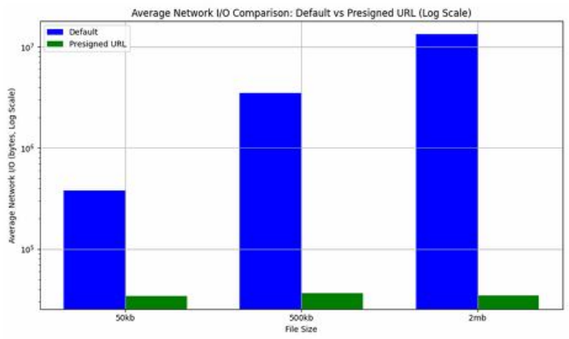
- 별도의 프로그램이나 확장 프로그램 없이, 브라우저 내장 기능만으로 브라우저 간 P2P 통신을 구현하기 위해 WebRTC를 적용하였습니다.
- NAT 뒤에 있는 사설 IP를 가진 Client 간 P2P 연결 설정하는 작업을 수행했습니다.
  - Signaling Server를 통해 연결 대상을 파악하고, STUN 서버를 사용하여 NAT을 통과할 수 있는지 확인합니다.
  - 만약 STUN을 이용한 P2P 연결이 불가능한 경우, TURN 서버를 사용하여 간접적인 통신을 수행합니다.
  - UDP 기반의 P2P 연결이 성립되면, 이를 통해 스트리밍을 수행합니다.

왜 미디어 중계서버를 두어야 하는가?

- 스트리밍 수신자의 네트워크 대역폭 사용량을 줄이기 위하여 미디어 중계서버를 도입한 SFU 구조를 도입하였습니다.
  - Mesh 구조에서는 수신자가 송신자와 1:1로 연결해야 하므로, 송신자는 N명의 수신자만큼 업링크를 형성해야 하는 부담이 있었습니다.
  - 서비스에서 스트리밍 대상은 웹캠 영상, 화면 공유 영상, 마이크 음성, 화면 음성의 총 4가지 미디어 스트림이었습니다.
  - VP8(비디오) 및 Opus(오디오) 코덱을 사용했으며, 60 FPS, HD(720p) 환경에서 단일 수신자에게 스트리밍할 때 평균 7167kbps의 대역폭이 소모되었습니다.
  - 따라서, N명의 유저에게 스트리밍하기 위해서는 총 N × 7167kbps의 대역폭이 필요했습니다.
  - 이를 개선하기 위해 Mediasoup 기반 미디어 중계 서버를 도입하여, 송신자가 단 한 번의 업링크만 형성하는 SFU(Selective Forwarding Unit) 구조로 변경하였습니다.
  - 그 결과, 수신자는 8000kbps 미만의 대역폭만으로 다수의 유저에게 스트리밍이 가능해졌습니다.

Presigned URL 도입

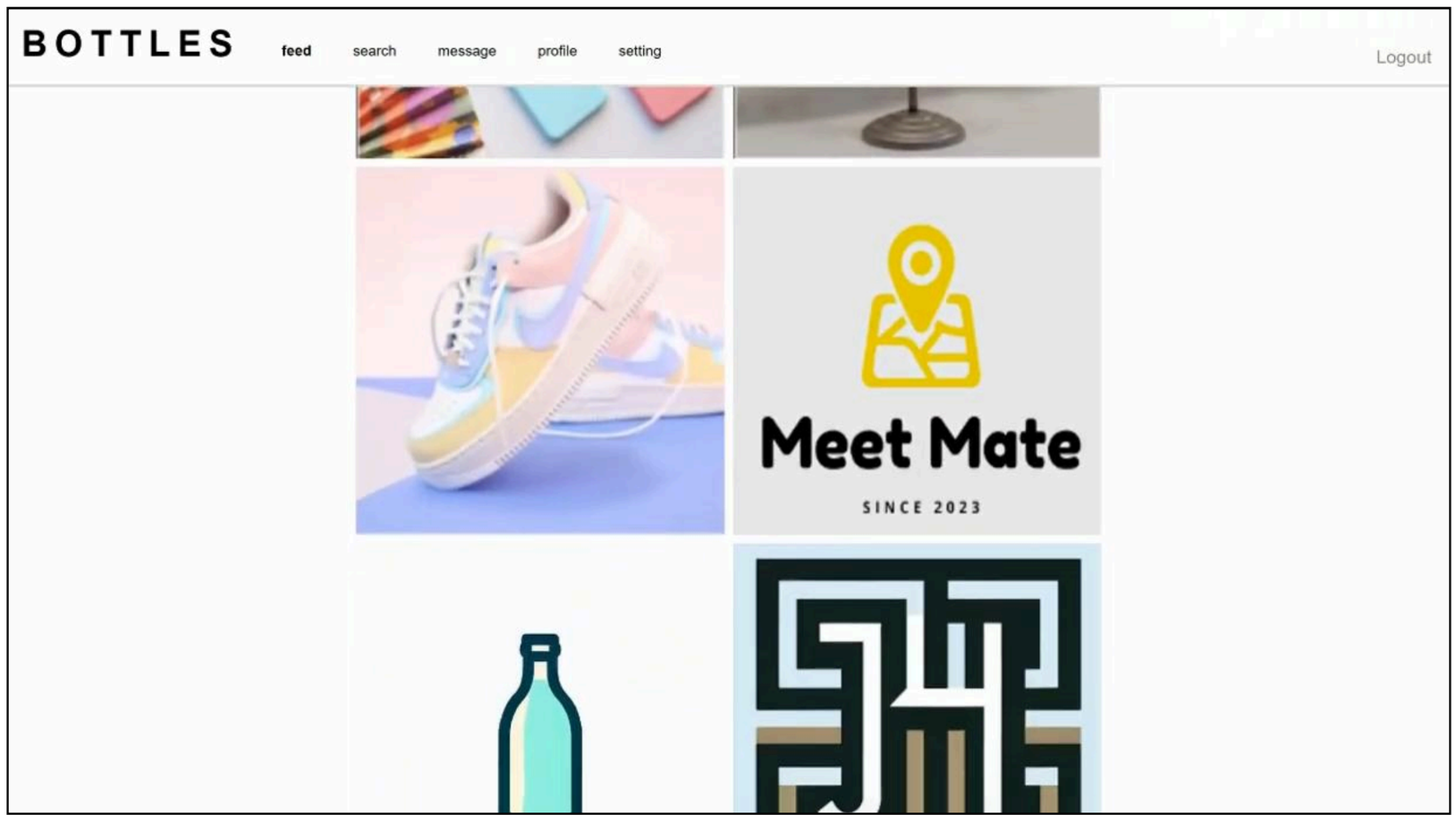




- S3에 이미지를 업로드하는 기능에 Presigned URL을 도입하였습니다.
  - 기존에 클라이언트에서 백엔드로, 그리고 백엔드가 S3로 이미지 업로드 하는 구조는 이미지 전송을 이중으로 수행했습니다.
  - 이것을 개선하고자 백엔드가 S3로부터 서명받은 URL을 획득하고 이를 클라이언트에 전송해 클라이언트는 S3로 이미지 전송을 1회만 수행하는 구조로 바꾸었습니다.
  - 결과적으로 이미지 업로드 기능에서 백엔드의 Network I/O 부담이 50KB 기준 91% 감소했습니다.

Bottles 익명 고민상담 SNS

Project - 2



이미지, 비디오, 텍스트로 자유롭게 구성한 게시글을 올리고 서로 댓글을 달며 유저간에 팔로우를 할 수 있는 SNS 입니다.
추가로 익명 모드를 추가하여, 익명 게시글로 작성된 피드는 작성자를 익명으로 유지한 체 랜덤한 유저에게 노출됩니다.
이를 수신한 유저는 마찬가지로 익명 모드의 상태로 댓글을 달고, 서로 간에 실시간 채팅을 수행할 수 있습니다.

작업 기간

2023. 09 - 2024. 02

구성원

5명 (Backend 2, Frontend 2, Android 1)

깃헙

https://github.com/GoByeonghu/Bottles_BE

기술 스택

Back-End

Django Django Rest Framework Django Channels
Nginx daphne

Front-End

React TypeScript

Database

MySQL

주요기능

- 이미지, 비디오, 텍스트로 자유롭게 구성한 피드 CRUD
  - 사용자가 원하는 구성과 순서로 피드를 설정 가능
- WebSocket를 이용한 실시간 채팅
  - Python 기반의 WAS에서 Websocket 연결 수행
- 익명모드 전환 & 게시글 랜덤 매칭
  - 유저 계정과 1:1로 매핑되는 익명 계정 부여
  - 익명 계정으로 작성한 게시글은 사용자가 감추어지고,
    게시글이 랜덤한 유저에게 노출되며,
    익명 게시글과 매칭된 유저의 반응(댓글 등)도 익명으로 표기
- 유저간 팔로우
- 유저 아이디 기반 검색 및 자동완성 검색어 추천 기능
- 댓글 작성 및 댓글의 답글 달기 기능
  - 게시글에 댓글이 달리면 해당 댓글에 대하여 답글을 남긴다.
  - 게시글의 답글이 계층을 이루며 저장

역할

Back-End 개발

- Django(DRF)를 이용한 REST API 개발
- Websocket 실시간 채팅 개발
- Pillow를 활용한 사용자 이미지 리사이징 컴포넌트 개발
- JWT 인증 인가 구현
- 관계형 Database 설계
- 백오피스 시스템 개발

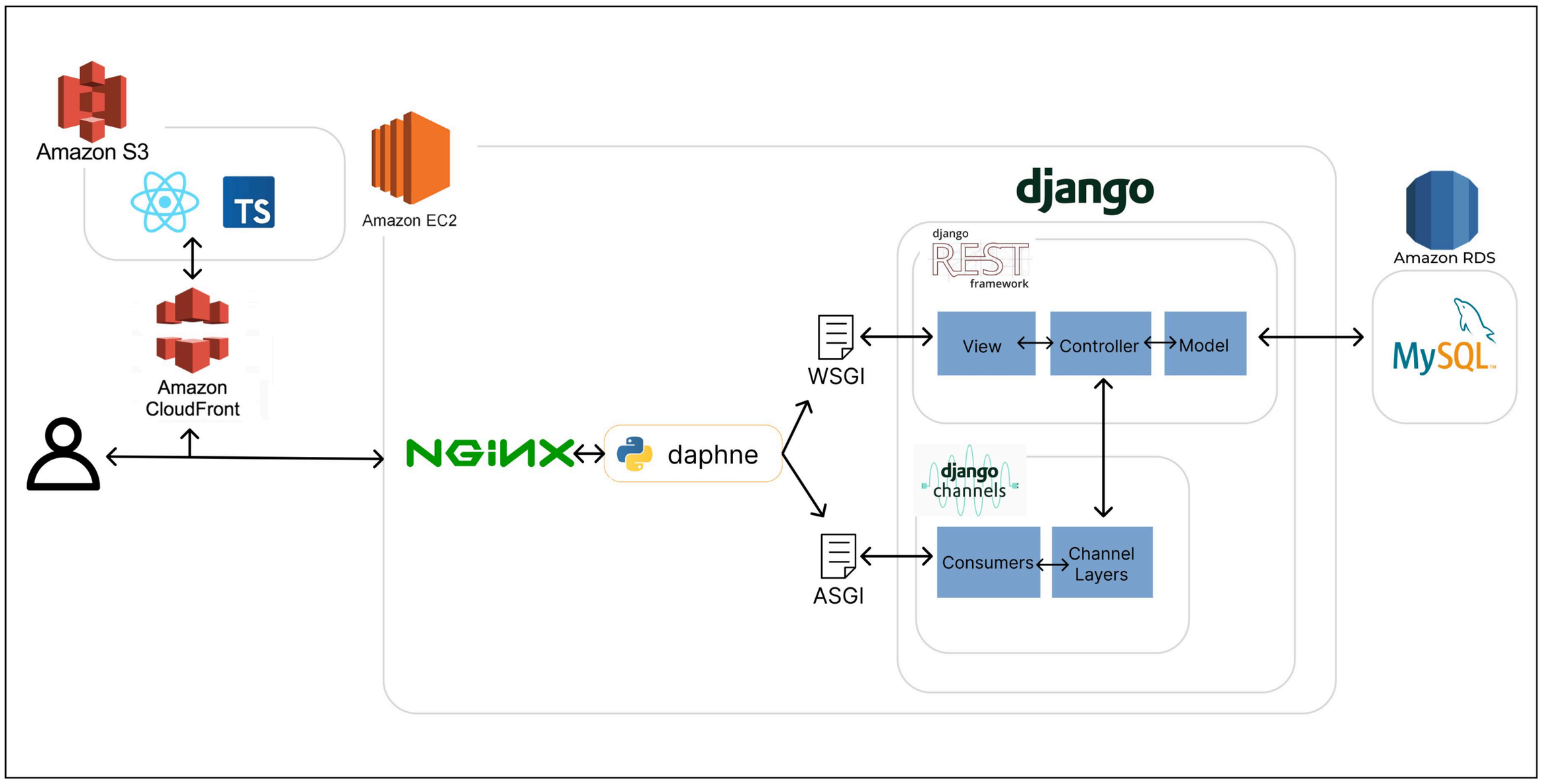
성과

- webP사용으로 이미지 저장용량 28% 절감
- Websocket 인증인가 오류 해결
- index 미사용 쿼리 식별 및 개선

Bottles 익명 고민상담 SNS

Project - 2

Architecture

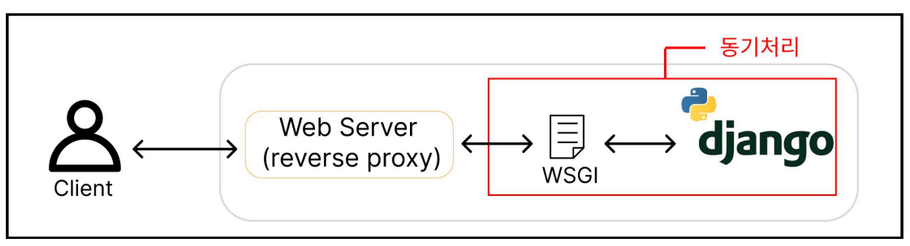


Python기반 WAS 환경 Websocket 채팅 개발

CGI 설정에 따른 Python Web Application Server의 WebSocket 통신 문제 인지

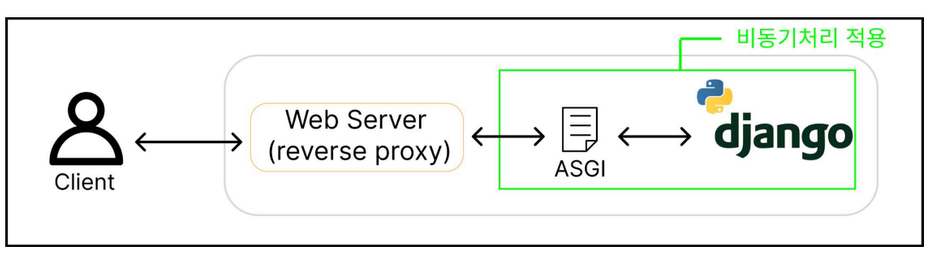
- WebSocket 적용 이전의 Django 사용방법에서 문제를 인지하였습니다.
  - 기존의 Django WAS는 요청을 동기 처리하였습니다.
  - 그 결과 WebSocket을 적용하면 연결을 유지하는 동안 하나의 프로세스가 계속해서 점유되는 문제가 예상되었습니다.

WSGI 동작 원리



- Web Server는 요청에 대해 동기 처리를 수행합니다.
  - WebSocket 연결을 수행하면, 연결이 끊어질 때까지 하나의 프로세스가 점유된 채로 유지됩니다.

ASGI 사용으로 비동기 처리 적용



- ASGI를 설정하여 비동기 처리를 적용했습니다.
  - Daphne 서버를 사용하여 WebSocket 프로토콜은 ASGI로, HTTP 요청은 WSGI로 처리하도록 설정하였습니다.
  - Django Channels를 활용해 WebSocket을 구현하고 비동기 처리를 적용하여 채팅 서버 개발을 완료했습니다.

이미지 리사이징 모듈 개발

사용자 업로드 이미지 리사이징 모듈의 코덱을 변경하여 저장 공간 28% 절감

- 기존의 JPG 및 PNG 코덱을 WebP로 통일하여 저장 효율을 개선했습니다.
  - 사용자 업로드 이미지를 JPG 70%, PNG 30%로 상정하여 테스트 환경을 구축하였습니다.
  - Pillow 라이브러리를 활용한 리사이징 모듈을 통해 WebP로 변경하여 저장하도록 자동화하였습니다.
  - 그 결과 저장 공간이 28% 절감되었습니다.

Bottles 익명 고민상담 SNS

Project - 2

ERD

Index 미사용 쿼리 개선

```
queryset = User.objects.filter(Q(username__icontains=search_query) | Q(name__icontains=search_query))
print(queryset.query)
```

```
SELECT * FROM users
WHERE username LIKE '%검색어%'
   OR nickname LIKE '%검색어%';
```

Slow Query 인식

- 사용자 아이디, 이름 기반 자동 완성 추천 기능을 위한 쿼리에서 비효율을 발견하였습니다.
  - Django 쿼리 로그를 확인해 보았습니다.
  - 그 결과 두 가지 문제 점을 발견했습니다.

1. LIKE 구문의 % 연산자 위치 수정

- LIKE 구문의 % 연산자 위치를 조정하여 검색 성능을 최적화하고, 인덱스를 활용할 수 있도록 개선하였습니다.
  - 기존의 쿼리 구문은 LIKE %검색어%와 같이 검색어 앞에도 %연산자가 위치하여 Index를 활용하지 못하는 구조였습니다.
  - 이를 개선하기 위해 username(id) 혹은 nickname(name)에 대한 검색은 앞 글자부터 하도록 규정하였습니다.
  - 아래와 같이 기존의 __icontains를 __startwith로 바꾸어 '검색어%' 가 적용되어 index 사용이 가능하도록 하였습니다.

```
queryset = User.objects.filter(Q(username__startswith=search_query) | Q(name__startswith=search_query))
```

```
SELECT * FROM users
WHERE username LIKE '검색어%'
   OR nickname LIKE '검색어%';
```

2. WHERE 절의 OR 연산자 사용문제 개선

- WHERE 절에서 OR 연산자를 사용하던 쿼리를 각 조건별로 개별 검색한 후, UNION을 통해 결과를 합치는 방식으로 최적화하였습니다.
  - 기존 쿼리는 WHERE 절에서 OR 연산자를 사용하여 인덱스가 적용되지 않는 문제가 있었습니다.
  - 이를 해결하기 위해 검색 대상 칼럼별로 개별 조회하여 index를 활용한 후, UNION을 통해 결과를 합치는 방식으로 최적화하였습니다.

```
users_with_matching_username = User.objects.filter(username__startswith=search_query)
users_with_matching_name = User.objects.filter(name__startswith=search_query)
filtered_users = users_with_matching_username.union(users_with_matching_name)
```

OpenSesame ARM Trustzone 활용 도어락

Project - 3

키패드를 누르지 않아도 동작하거나 스마트폰과 연동하여 잠금을 해제하는 등의 기능을 하는 스마트 도어락을 보안성 있게 설계한 프로젝트입니다.
보안 성능을 높인 스마트 도어락의 구현을 위해 ARM TrustZone을 사용하여 설계하였습니다.
또한, 보안성을 확보한 통신 방식을 설계하기 위해 PGP를 도입하였습니다.

작업 기간

2022. 09 - 2022. 12

구성원

2명

깃헙

https://github.com/GoByeonghu/OpenSesame

기술 스택

Systems Programming

C

Security & Cryptography

OpenSSL

Networking

WIFI Ad-Hoc

Embedded Systems

Raspberry Pi 3B+ QEMU

주요기능

- 사용자 계정 도어락 등록
- WIFI Ad-Hoc을 이용한 도어락 개폐 명령 송수신
  - 모바일 단말과 도어락 단말이 WIFI Ad-Hoc을 이용하여 통신
- 개폐 명령 유효성 판단
- 개폐 명령 및 사용자 정보 암복호화
- ARM Trustzone을 활용한 보안성 강화

역할

설계 & 논문 작성 & 기능 구현

- 사용기술 연구 및 아키텍쳐 설계
- 논문 작성
- OpenSSL을 이용한 사용자 정보 암복호화 모듈 및
  사용자 명령 유효성 판단 컴포넌트 개발

성과

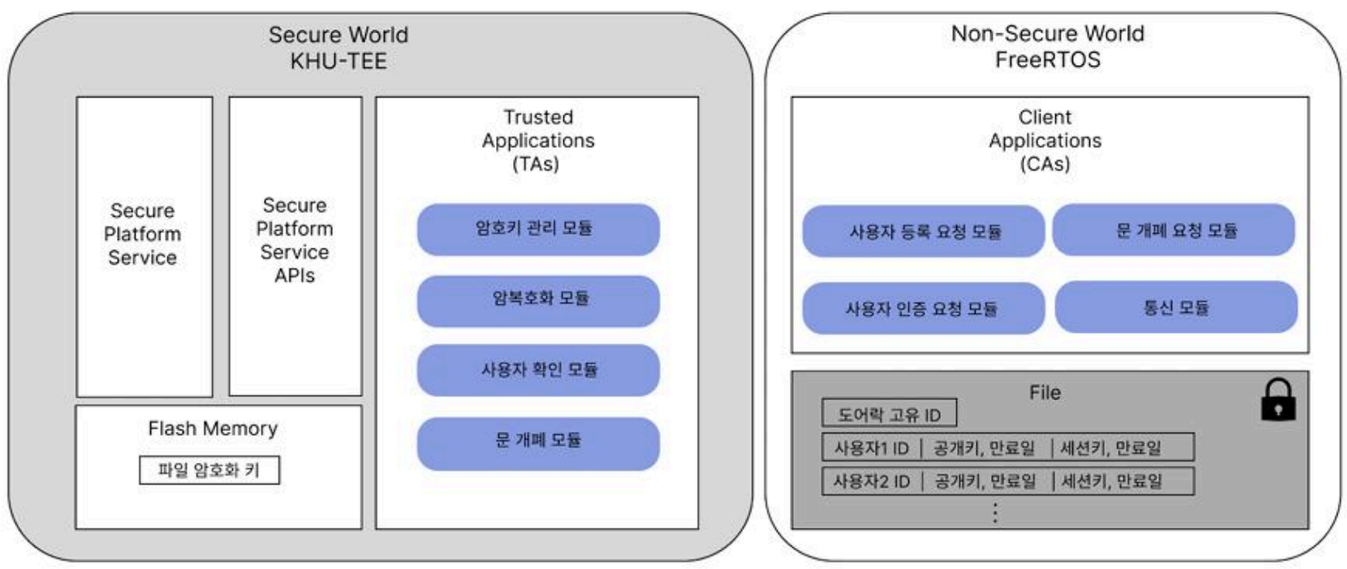
- 2022 한국소프트웨어종합학술대회 제1저자 논문등록 및 발표자 선정
- QEMU에서 사용가능한 Prebuilt Binaries 파일 오픈소스 공개

OpenSesame ARM Trustzone 활용 도어락

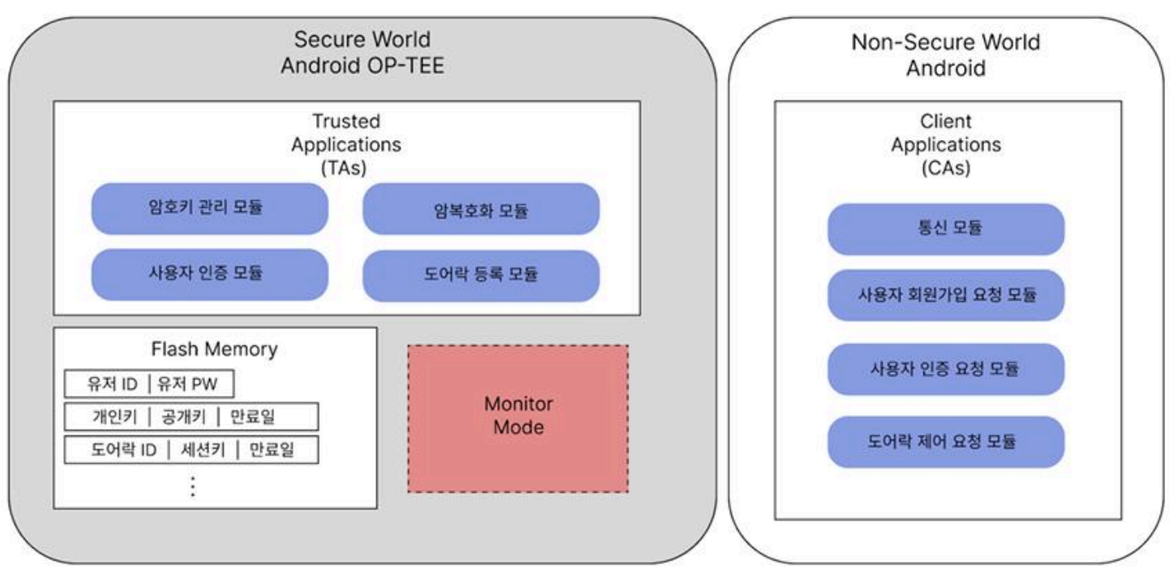
Project - 3

Architecture

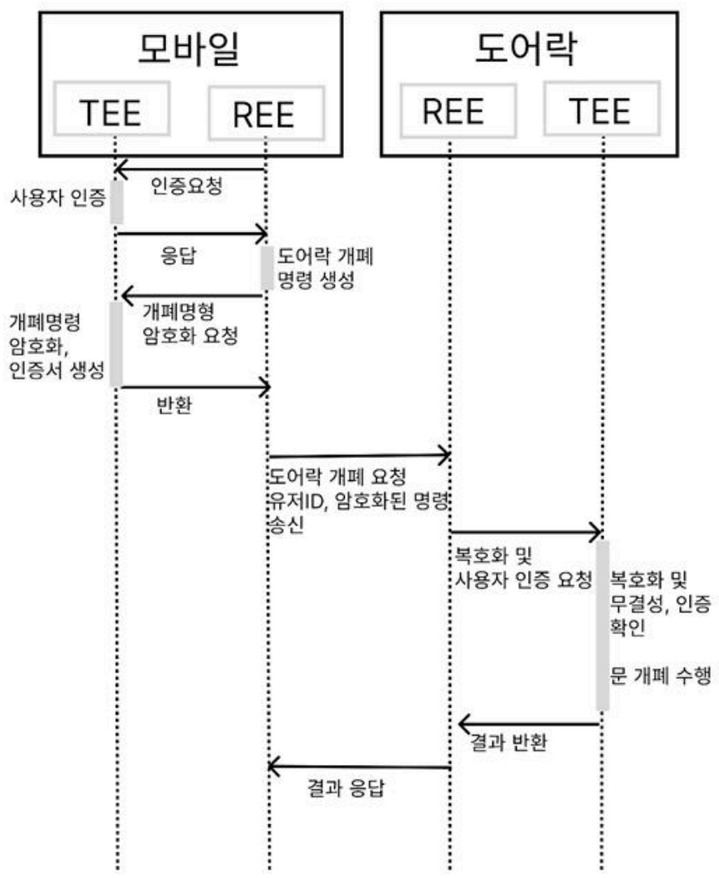
Door Lock System



Mobile System



OP-TEE



OP-TEE를 적용한 통신 프로세스 설계

- TEE(신뢰 실행 환경)에는 보안상 민감한 모듈을 배치했습니다.
  - 암복호화 모듈, 인증/인가 모듈, 도어락 개폐 모듈 등의 해킹에 민감한 요소를 배치하였습니다.
- REE(일반 실행 환경)에는 민감성이 적은 모듈을 배치했습니다.
  - Wi-Fi 통신 모듈 등 직접 데이터를 송수신하는 모듈을 배치했습니다.

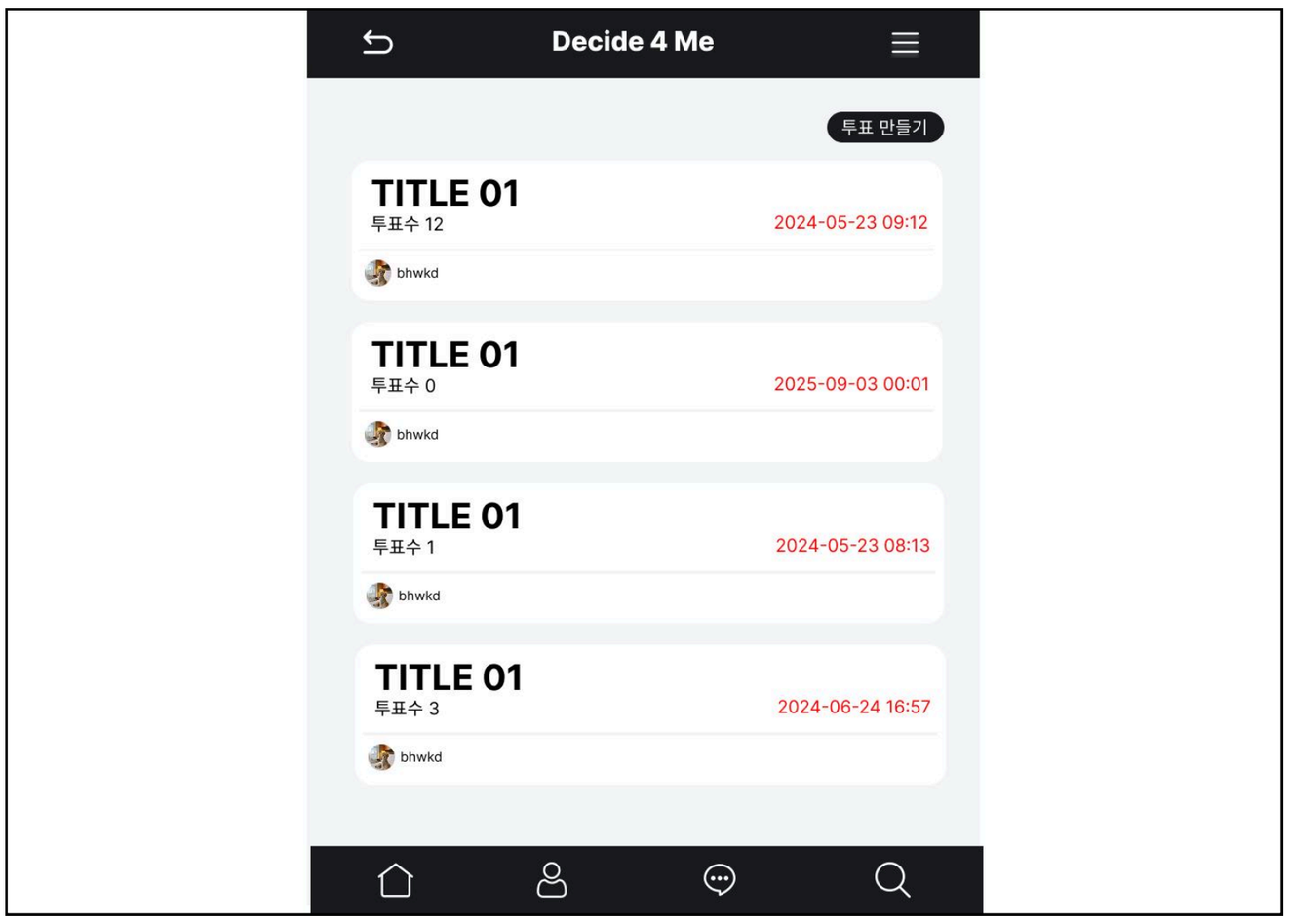
PGP

OpenSSL과 C언어를 사용하여 PGP를 수행하는 암복호화 모듈을 개발하였습니다.

- SHA-256을 이용해 데이터 무결성을 보장하였습니다.
- AES를 이용해 데이터 기밀성을 보장하였습니다.
- RSA를 이용해 사용자 인증을 수행하였습니다.

Decide4Me 결정을 도와주는 투표 웹사이트

Project - 4



사용자가 결정이 고민되는 사안에 대하여 원하는 만큼의 선택지를 두고 데드라인을 설정하여 게시글을 올리면
다른 유저가 투표를 진행하고, 데드라인을 넘어가면 최다 득표를 표기하는 결정도움 사이트입니다.

작업 기간

2024. 05 - 2024. 06 (MVP 개발)
2025. 02 - 현재 진행 중 (고도화 진행 중)

구성원

1명 (개인 프로젝트)

깃헙

https://github.com/GoByeonghu/Decide4Me_BE/tree/develop

기술 스택

Back-End

Spring Boot Spring Security
Nginx

Front-End

React TypeScript

Database

MySQL Redis

주요기능

- Kakao OAuth 로그인
- 원하는 갯수로 투표 선택지를 설정하며 게시글 생성
- 투표 게시글 데드라인 알림

역할

Back-End 개발

- Spring Boot를 이용한 REST API 개발
- Spring Security 이용한 인증 인가 개발
- 관계형 Database 설계

Front-End 개발

- React 이용한 프론트엔드 개발