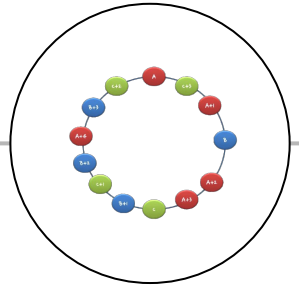
임형섭

Backend Developer

개발 프로젝트

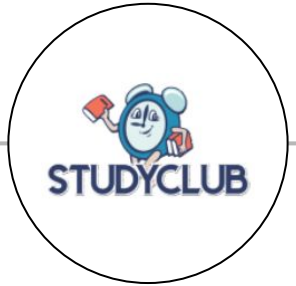


Consistent Hashing

백엔드

샤드 관리 및 효율적인
데이터 분배 시스템

2024.8 ~2024.9



Study-Club

백엔드

웹앱 스터디 모집 및 활동
통합 플랫폼 서비스

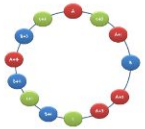
2024.6 ~2024.7

Doggiverse

백엔드

웹앱 반려인과 반려견을 위한
컨텐츠 기반 커뮤니티 서비스

2024.5 ~2024.6

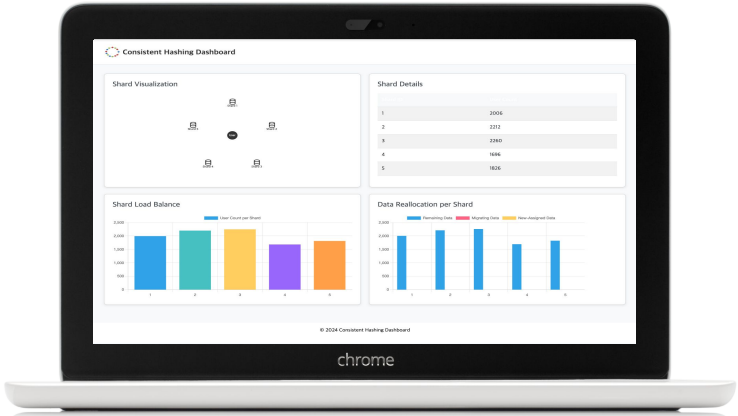


샤드 관리 및 효율적인 데이터 재분배 시스템

개발 기간 2024.8 ~2024.9

플랫폼 Web

개발 인원 1명



GitHub https://github.com/h3lim/Consistent-hashing

개발 환경

언어 Java(JDK 17), HTML/CSS, JavaScript

서버 Spring Boot 내장 톰캣 서버

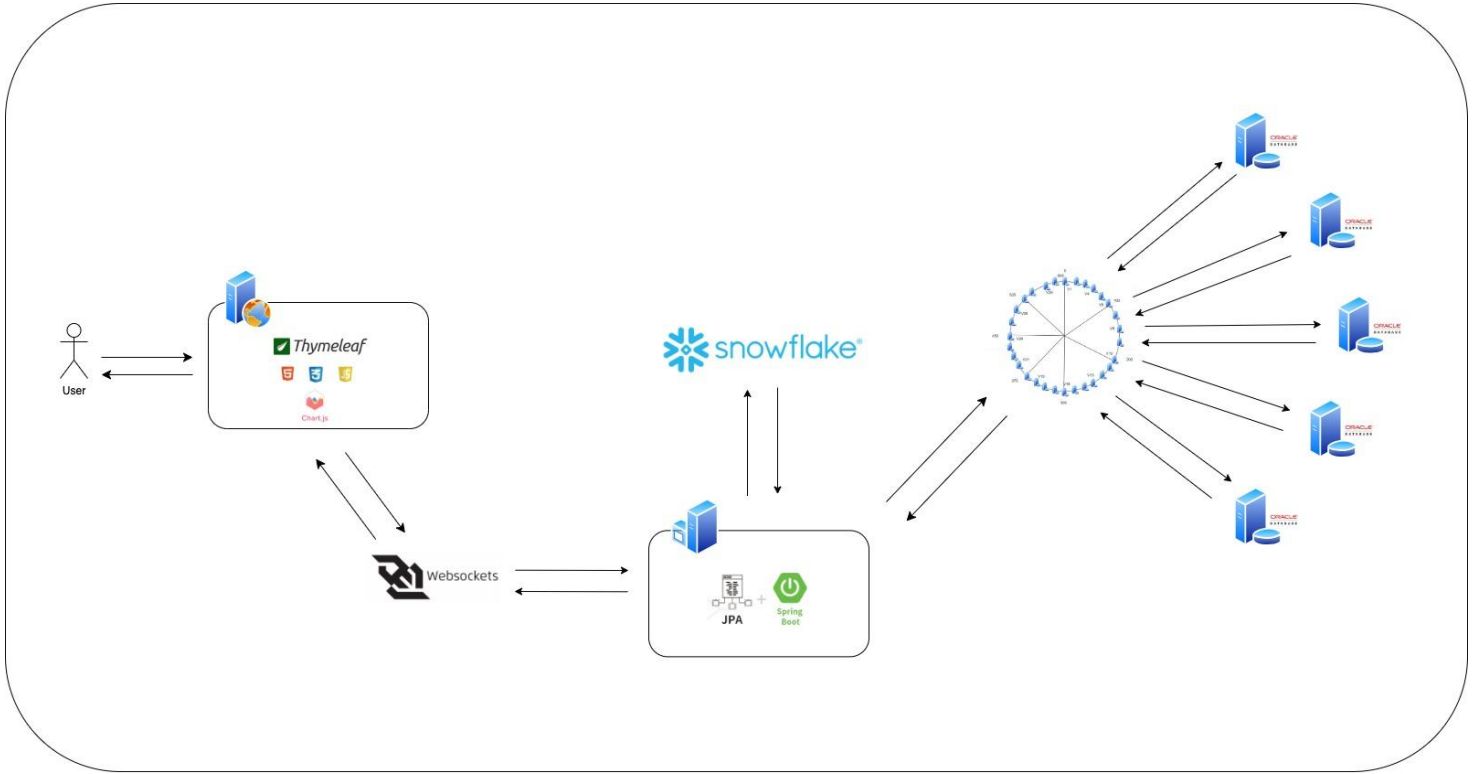
프레임워크 Spring Boot, JPA,Spring WebSocket

DB Oracle DB

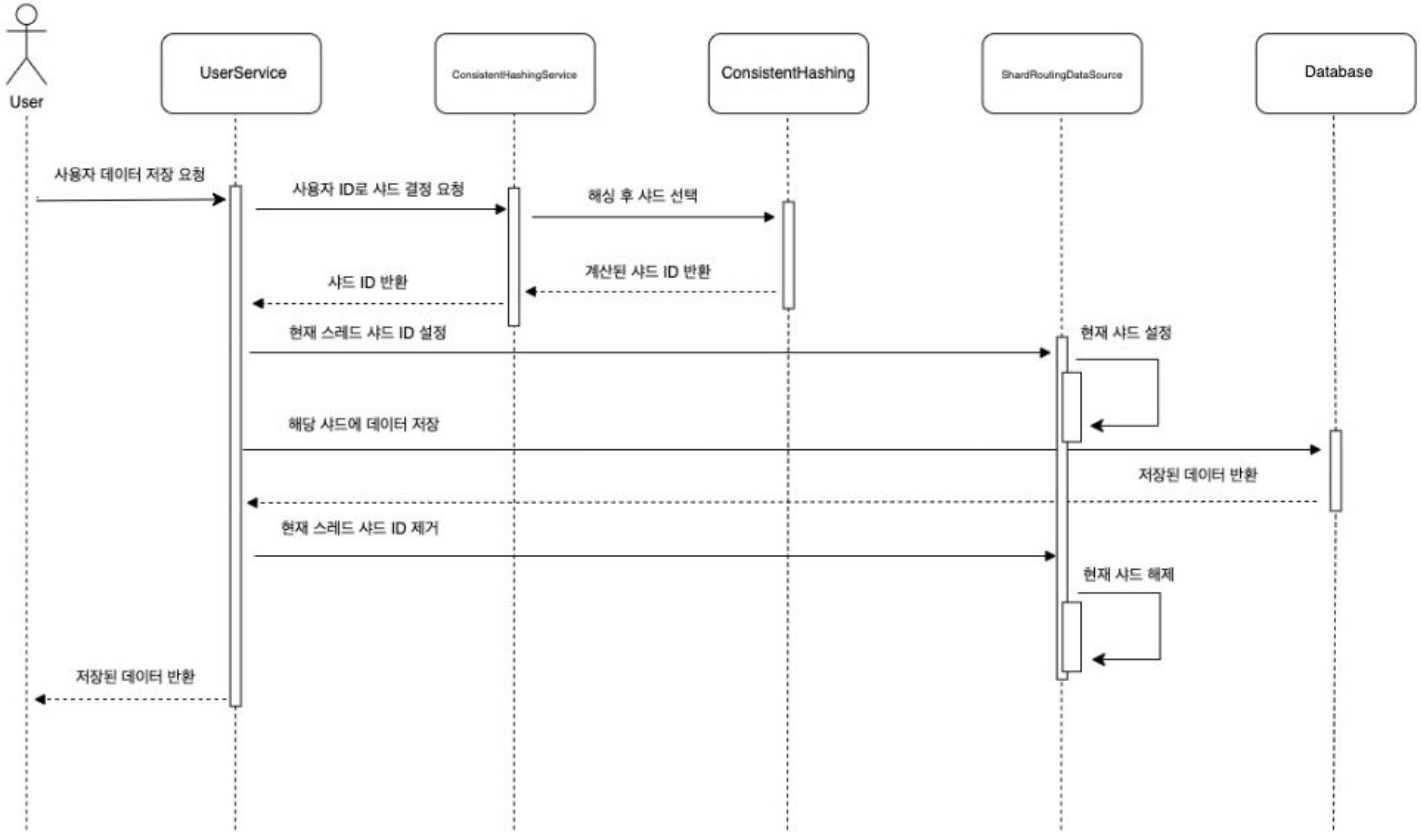
IDE IntelliJ,SQL Developer,Postman

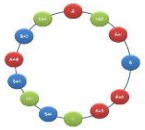
API,라이브러리 WebSocket API, Thymeleaf, Guava

시스템 아키텍처

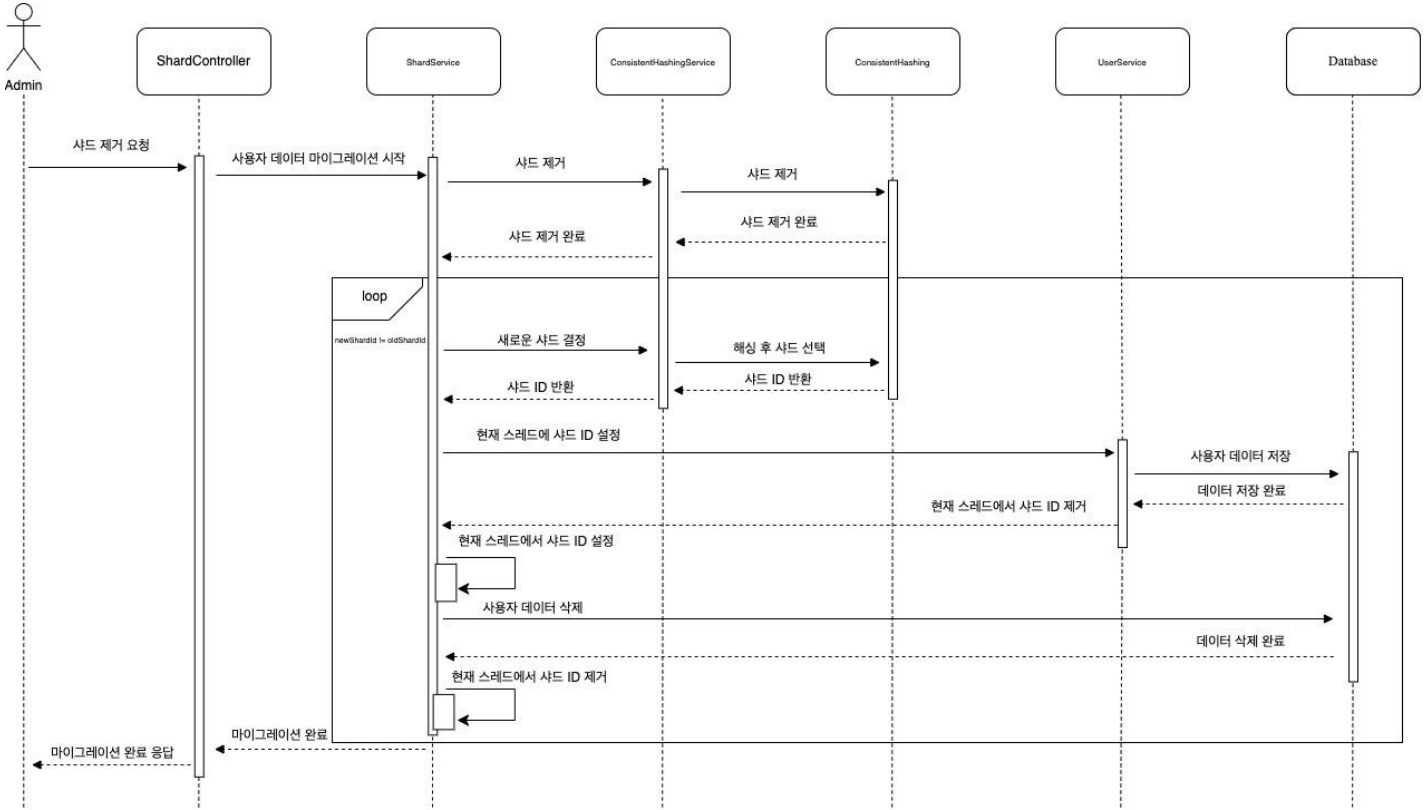

User
Thymeleaf
Chart.js
snowflake
Websockets
JPA
Spring Boot
ORACLE

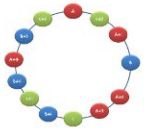
시퀀스 다이어그램 - 샤딩 비즈니스 로직



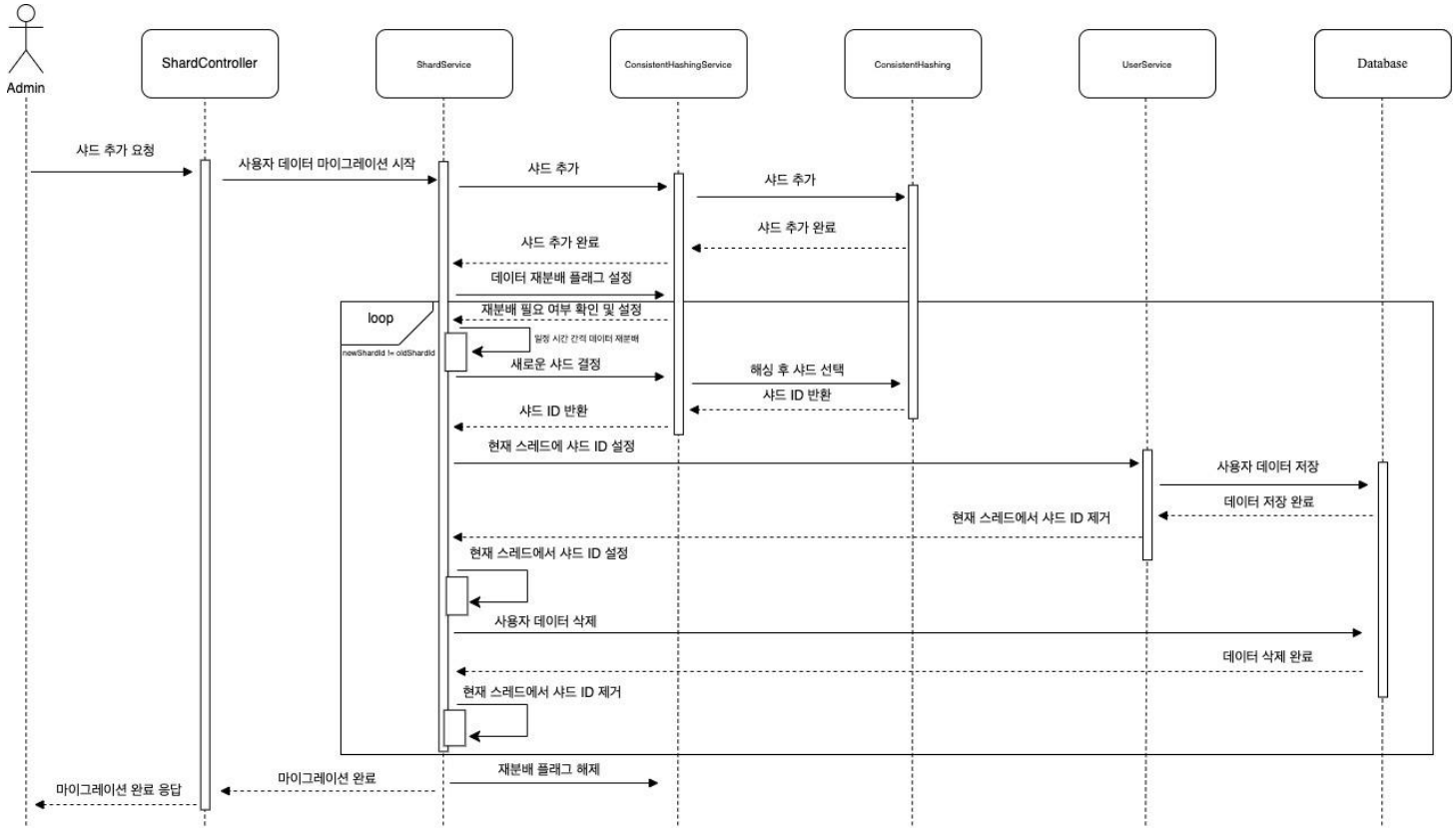


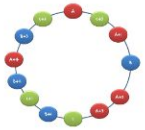
시퀀스 다이어그램 - 샤드 제거 및 데이터 재분배





시퀀스 다이어그램 - 샤드 추가 및 데이터 재분배



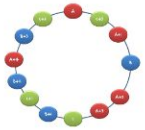


샤딩 핵심 비즈니스 로직

- Consistent Hashing을 사용하여 데이터를 분산하고, 여러 개의 데이터베이스 샤드에 저장합니다. 이 방식은 추가 및 제거되는 샤드에 따라 전체 데이터를 재분배하지 않고, 최소한의 데이터 이동만으로 처리가 가능합니다. 이를 통해 샤드를 동적으로 관리할 수 있으며, 시스템 확장성 및 안정성을 보장합니다.

구현 상세

- ConsistentHashing 클래스:
  - ConsistentHashing 클래스는 해시링을 관리하며, 주어진 노드(샤드) 수와 가상 노드 수를 이용하여 해시를 계산합니다. murmur3_32 해싱 알고리즘을 사용하여 사용자 ID나 데이터 키를 해시링에 매핑합니다.
  - addNode(int shard): 샤드를 추가하는 메서드로, 가상 노드를 기반으로 각 샤드를 해시링에 추가합니다.
  - removeNode(int shard): 샤드를 제거하는 메서드로, 제거된 샤드의 가상 노드를 해시링에서 삭제합니다.
  - getNode(long userId): 주어진 사용자 ID 또는 데이터 키를 해시링에 매핑하여 해당 샤드를 반환합니다.
- ConsistentHashingService 클래스:
  - 이 서비스 클래스는 실제 샤드에서 사용자의 데이터를 저장하거나 읽을 때, 해당 사용자가 속한 샤드를 결정합니다.
  - getShardForUser(Long userId): 사용자 ID를 입력받아 사용자가 속할 샤드를 결정하고, 해당 샤드에 데이터를 저장하거나 로드하는 역할을 합니다.
  - addShard(int shardId): 새로운 샤드를 추가할 때, 해시링에 샤드를 추가하고 샤드 분포를 업데이트합니다.

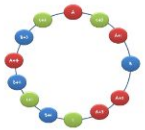


유니크 키값 생성

- 각 사용자의 고유 ID나 데이터 레코드의 유니크한 식별자를 생성하기 위해 Custom ID Generator를 구현했습니다. 이 ID 생성기는 Snowflake 알고리즘에서 영감을 받아 설계되었으며, 고유한 노드 ID, 타임스탬프, 시퀀스 값을 조합하여 분산 시스템에서도 충돌이 없는 식별자를 생성합니다.

구현 상세

- CustomIdGenerator 클래스:
  - 노드 ID는 각 서버나 샤드에 고유하게 할당된 값으로, 해당 노드에서 생성되는 모든 ID가 고유하도록 보장합니다.
  - 타임스탬프는 현재 시간을 기준으로 하여, 같은 시간 내에서도 여러 개의 고유 ID를 생성할 수 있습니다.
  - 시퀀스 값은 동일한 타임스탬프에서 중복된 ID 생성을 피하기 위해 추가된 값입니다.
- ID는 밀리초 단위의 타임스탬프, 노드 ID, 시퀀스 값을 결합하여 고유한 값을 생성합니다. 타임스탬프가 밀리초 단위로 제공되기 때문에 짧은 시간 안에 수백만 개의 고유 ID를 생성할 수 있습니다.



데이터 이동 (Shard Migration)

- 샤드를 추가하거나 제거할 때, 데이터는 새로운 샤드로 재분배됩니다. 이때 Consistent Hashing을 사용하여 최소한의 데이터 이동만으로 새 샤드에 데이터를 재배치하며, 서비스가 중단되지 않도록 비동기로 처리됩니다. 또한, WebSocket을 통해 실시간으로 데이터 이동 상황을 대시보드에서 확인할 수 있습니다.

구현 상세

- ShardService 클래스:
  - migrateUsersFromShard(int removedShardId): 샤드를 제거할 때, 해당 샤드에 저장된 데이터를 새롭게 할당된 샤드로 이동시키는 메서드입니다.
  - migrateUser(User user, int oldShardId): 개별 사용자의 데이터를 새로운 샤드로 이동시키는 로직입니다. 먼저, 사용자가 속한 샤드가 제거되면 새로운 샤드를 찾아 해당 사용자의 데이터를 저장합니다.
- WebSocket 연동:
  - sendShardUpdates(): 일정 시간마다 샤드의 데이터 분포 및 이동 상황을 WebSocket을 통해 실시간으로 전송하여, 관리자는 이를 대시보드에서 모니터링할 수 있습니다.

정적 데이터베이스 관리 문제

매번 테스트를 진행할 때마다 각 샤드에 데이터를 등록하고 설정하는 과정에서 반복적으로 어려움을 겪었습니다. 특히, 새로운 샤드를 추가하거나 제거할 때 수동으로 등록해야 하며, 설정의 일관성이 유지되지 않아 테스트가 복잡해지고 시간이 소요되었습니다.

증상

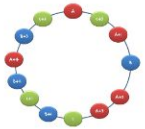
- 테스트를 실행할 때마다 각 데이터베이스(샤드)에 데이터를 일일이 등록 필수
- 데이터베이스가 많아질수록 수동 설정 시간이 늘어나고, 테스트 환경을 준비하는 과정에서 빈번한 오류가 발생
- 새로운 샤드를 추가하거나 제거할 때마다 설정을 수동으로 변경해야 하는 번거로움

원인 분석

- 데이터베이스 설정이 코드 내에 하드코딩되어 있어, 샤드가 변경될 때마다 이를 코드에서 수정해야 하는 비효율적인 방식
- 애플리케이션 실행 시 자동으로 샤드를 등록하거나 삭제하는 기능이 없었기 때문에, 테스트 환경에서 추가적인 샤드 설정이 필요

해결 방법

- Spring의 @ConfigurationProperties를 사용하여 설정 파일에서 샤드 정보를 동적으로 로드할 수 있도록 수정했습니다. 이를 통해, 애플리케이션 실행 시 자동으로 샤드들을 설정 파일에 정의된 대로 동적으로 등록하고 관리
- ShardDataSourceProperties 클래스를 활용해 모든 샤드를 동적으로 로드하고 관리할 수 있도록 설정



데이터 일관성 문제 (샤드 추가/제거 시)

샤드를 추가하거나 제거할 때, 사용자 데이터가 새로운 샤드로 올바르게 이동하지 않거나 일부 데이터가 유실되는 문제를 겪었습니다. 이는 특히 많은 데이터를 처리할 때 더 두드러졌으며, 데이터 이동 과정에서 일관성이 유지되지 않는 상황이 발생했습니다.

증상

- 샤드 제거 시 데이터가 다른 샤드로 옮겨지지 않고 사라지는 현상
- 샤드를 추가한 후에도 기존 데이터가 새로운 샤드로 재분배되지 않는 문제
- 데이터 이동이 일관되지 않아 일부 사용자 데이터가 여러 샤드에 중복 저장되거나, 반대로 데이터가 없는 상태로 남는 현상

원인 분석

- 데이터 이동 중 트랜잭션 관리를 제대로 하지 않아, 일부 데이터가 제대로 처리되지 않거나 중간에 데이터 이동이 실패하여 일관성이 깨짐
- 실시간으로 데이터 이동을 처리할 때, 많은 데이터를 한 번에 처리하려고 하면서 시스템 부하가 발생했고, 그로 인해 데이터 이동이 불완전

해결 방법

- 데이터 마이그레이션을 처리할 때, 트랜잭션을 사용하여 하나의 샤드에서 데이터 이동 작업이 완전히 끝난 후에만 성공 처리되도록 했습니다. 트랜잭션이 실패할 경우, 해당 샤드의 모든 작업이 롤백되도록 설계하여 데이터 일관성을 보장
- 많은 데이터를 한 번에 옮기는 대신, 사용자 데이터를 분할하여 순차적으로 마이그레이션하도록 설계했습니다. 이를 통해 시스템 부하를 줄이고 안정적인 데이터 이동을 보장

STUDYCLUB

웹앱 스터디 모집 및 활동 통합 플랫폼 서비스

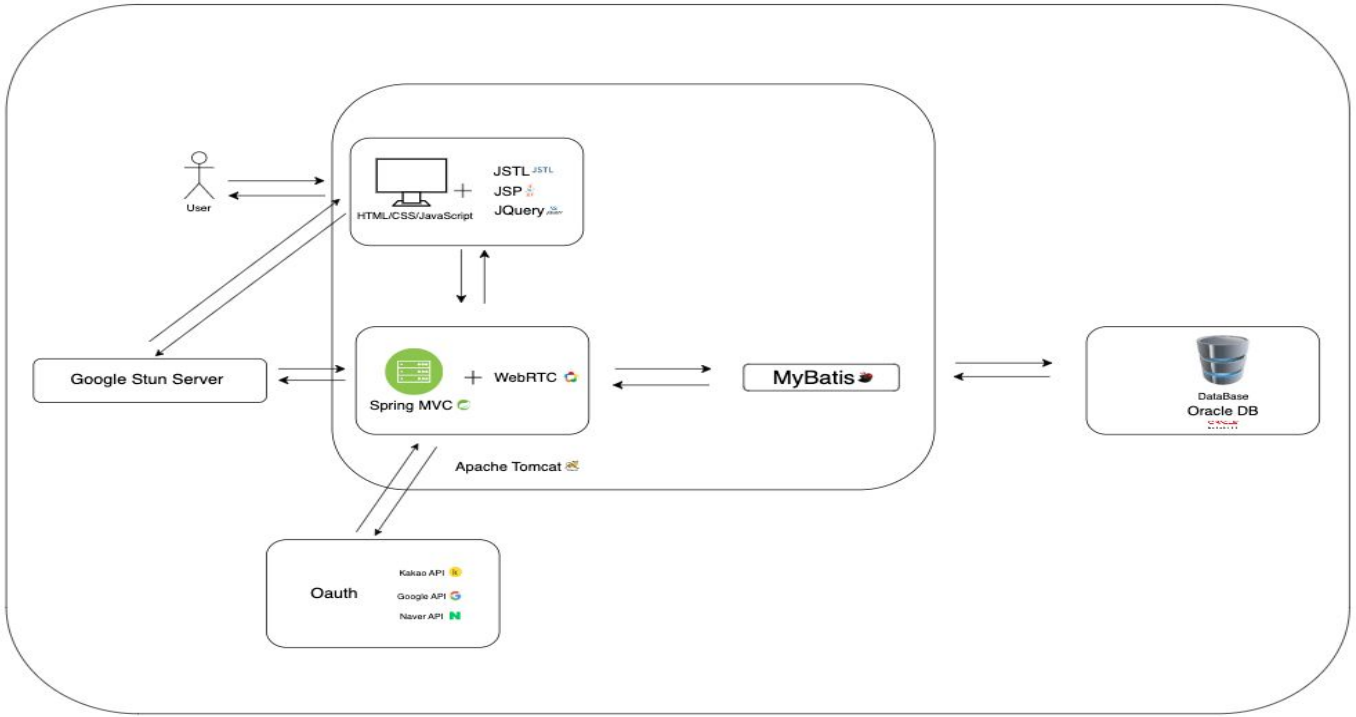
GitHub https://github.com/h3lim/StudyClub

개발 기간	2024.7 ~2024.8
플랫폼	Web
개발 인원	4명
담당 역할	Web RTC를 통한 1대1 화상면접 구현, 퀴즈 기능 구현, 서비스 기획 및 방향성 설정

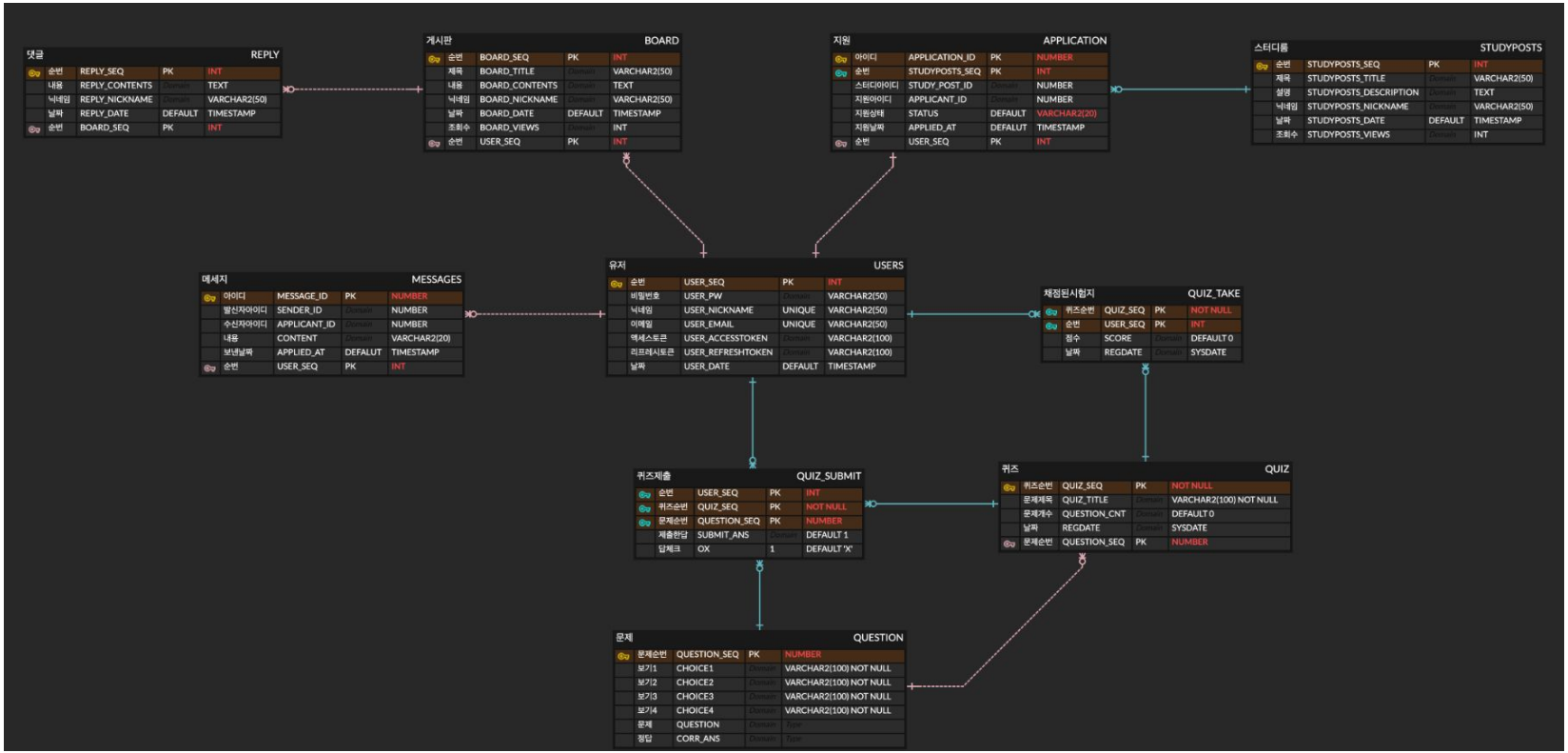
개발 환경

언어	Java(JDK 17), HTML/CSS, JavaScript
서버	Apache Tomcat 9.0, WebSocket, WebRTC
프레임워크	Spring MVC, Spring Boot, Spring Security, MyBatis 3.4.6
DB	Oracle DB
IDE	IntelliJ,SQL Developer
API,라이브러리	KAKAO/NAVER/GOOGLE Login API, JQuery, WebRTC API, WebSocketAPI, Log4j, SLF4J,Validation

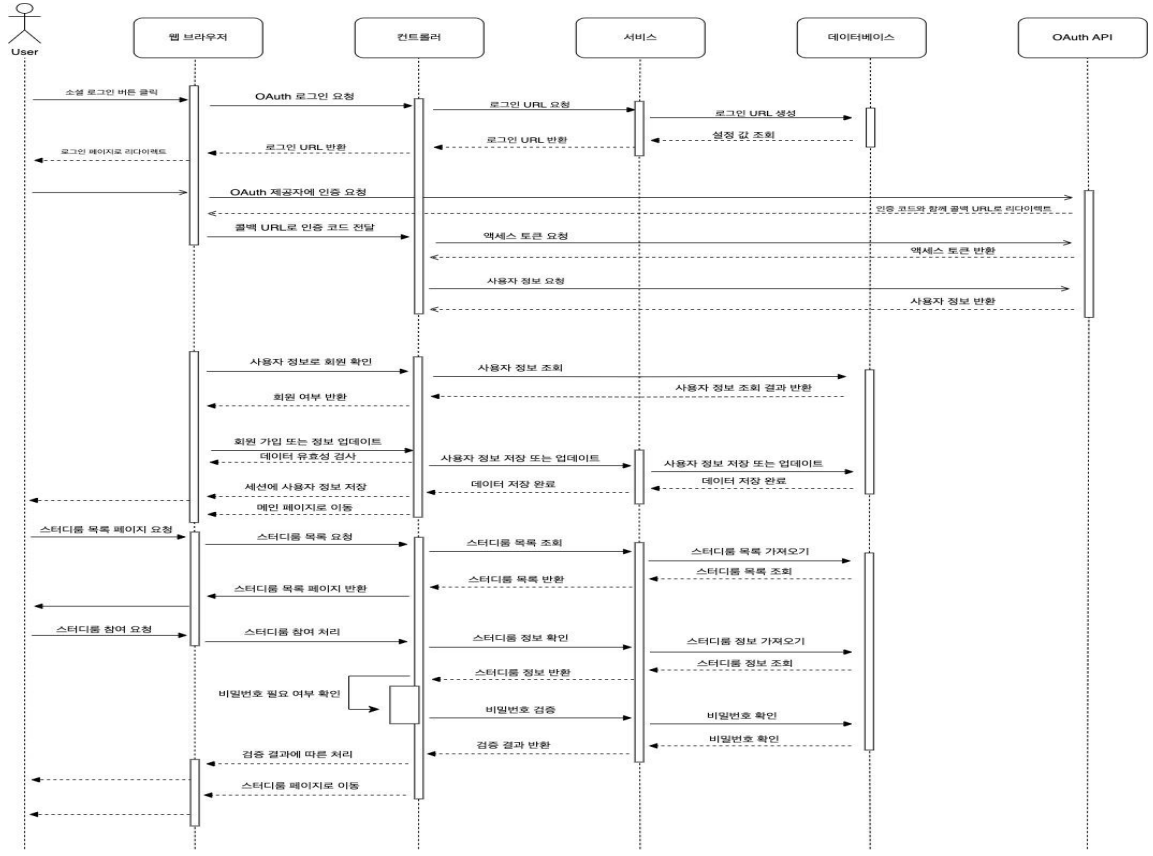
시스템 아키텍처


User
HTML/CSS/JavaScript
JSTL
JSP
JQuery
Google Stun Server
Spring MVC
WebRTC
MyBatis
DataBase
Oracle DB
Apache Tomcat
Oauth
Kakao API
Google API
Naver API

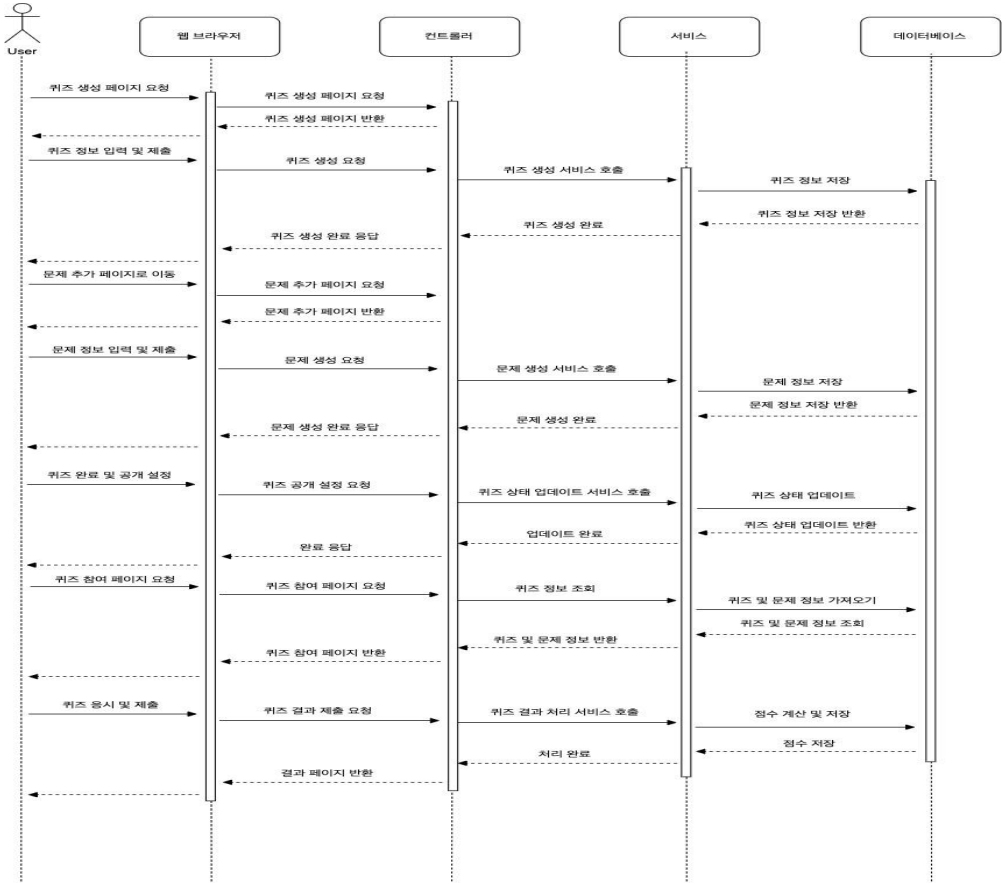
ERD 설계


댓글 REPLY
순번 REPLY_SEQ PK INT
내용 REPLY_CONTENTS TEXT
닉네임 REPLY_NICKNAME VARCHAR2(50)
날짜 REPLY_DATE DEFAULT TIMESTAMP
순번 BOARD_SEQ PK INT
게시판 BOARD
순번 BOARD_SEQ PK INT
제목 BOARD_TITLE VARCHAR2(50)
내용 BOARD_CONTENTS TEXT
닉네임 BOARD_NICKNAME VARCHAR2(50)
날짜 BOARD_DATE DEFAULT TIMESTAMP
조회수 BOARD_VIEWS INT
순번 USER_SEQ PK INT
지원 APPLICATION
아이디 APPLICATION_ID PK NUMBER
순번 STUDYPOSTS_SEQ PK INT
스터디아이디 STUDY_POST_ID NUMBER
지원아이디 APPLICANT_ID NUMBER
지원상태 STATUS DEFAULT VARCHAR2(20)
지원날짜 APPLIED_AT DEFALUT TIMESTAMP
순번 USER_SEQ PK INT
스터디룸 STUDYPOSTS
순번 STUDYPOSTS_SEQ PK INT
제목 STUDYPOSTS_TITLE VARCHAR2(50)
설명 STUDYPOSTS_DESCRIPTION TEXT
닉네임 STUDYPOSTS_NICKNAME VARCHAR2(50)
날짜 STUDYPOSTS_DATE DEFAULT TIMESTAMP
조회수 STUDYPOSTS_VIEWS INT
메세지 MESSAGES
아이디 MESSAGE_ID PK NUMBER
발신자아이디 SENDER_ID NUMBER
수신자아이디 APPLICANT_ID NUMBER
내용 CONTENT VARCHAR2(20)
보낸날짜 APPLIED_AT DEFALUT TIMESTAMP
순번 USER_SEQ PK INT
유저 USERS
순번 USER_SEQ PK INT
비밀번호 USER_PW VARCHAR2(50)
닉네임 USER_NICKNAME UNIQUE VARCHAR2(50)
이메일 USER_EMAIL UNIQUE VARCHAR2(50)
액세스토큰 USER_ACCESSTOKEN VARCHAR2(100)
리프레시토큰 USER_REFRESHTOKEN VARCHAR2(100)
날짜 USER_DATE DEFAULT TIMESTAMP
채점된시험지 QUIZ_TAKE
퀴즈순번 QUIZ_SEQ PK NOT NULL
순번 USER_SEQ PK INT
점수 SCORE DEFAULT 0
날짜 REGDATE SYSDATE
퀴즈제출 QUIZ_SUBMIT
순번 USER_SEQ PK INT
퀴즈순번 QUIZ_SEQ PK NOT NULL
문제순번 QUESTION_SEQ PK NUMBER
제출한답 SUBMIT_ANS DEFAULT 1
답체크 OX 1 DEFAULT 'X'
퀴즈 QUIZ
퀴즈순번 QUIZ_SEQ PK NOT NULL
문제제목 QUIZ_TITLE VARCHAR2(100) NOT NULL
문제개수 QUESTION_CNT DEFAULT 0
날짜 REGDATE SYSDATE
문제순번 QUESTION_SEQ PK NUMBER
문제 QUESTION
문제순번 QUESTION_SEQ PK NUMBER
보기1 CHOICE1 VARCHAR2(100) NOT NULL
보기2 CHOICE2 VARCHAR2(100) NOT NULL
보기3 CHOICE3 VARCHAR2(100) NOT NULL
보기4 CHOICE4 VARCHAR2(100) NOT NULL
문제 QUESTION
정답 CORR_ANS

시퀀스 다이어그램 - OAuth 로그인 및 스터디룸 참여



시퀀스 다이어그램 - 퀴즈 생성 및 참여



WebRTC를 통한 실시간 화상 회의 기능

- STUDYCLUB 프로젝트에서는 스터디룸 내에서 사용자들이 실시간으로 화상 회의를 진행할 수 있도록 WebRTC 기술을 활용하였습니다. 이를 통해 원격에 있는 사용자들이 마치 한 공간에 있는 것처럼 소통하고 협업할 수 있습니다.

구현 상세

- 시그널링 서버 구축: WebRTC 연결을 설정하기 위해 필요한 시그널링 과정을 처리하기 위해 CamController를 구현하였습니다.
- STUN 서버 사용: NAT 방화벽 뒤에 있는 클라이언트 간의 연결을 지원하기 위해 공용 STUN 서버를 활용하였습니다.
- WebSocket을 통한 통신: 클라이언트와 서버 간의 실시간 메시지 전달을 위해 WebSocket 프로토콜을 사용하였습니다.
- 클라이언트 측 구현:
    - JavaScript의 RTCPeerConnection API를 사용하여 P2P 연결을 설정하였습니다.
    - 사용자의 미디어 스트림(오디오 및 비디오)을 캡처하고 상대방에게 전송하였습니다.
    - 시그널링 데이터를 주고받기 위해 WebSocket을 통해 서버와 통신하였습니다.

퀴즈 생성 및 참여 기능

- STUDYCLUB에서는 사용자들이 직접 퀴즈를 생성하고, 다른 사용자들이 해당 퀴즈에 참여하여 학습 내용을 복습할 수 있는 기능을 제공합니다. 이를 통해 사용자 간의 지식 공유와 학습 동기 부여를 촉진합니다.

구현 상세

- 퀴즈 생성 과정:
  - 사용자가 퀴즈 제목, 설명 등을 입력하여 퀴즈를 생성합니다.
  - 생성된 퀴즈에 여러 개의 문제를 추가할 수 있습니다.
  - 각 문제는 질문, 보기(선택지), 정답 등의 정보를 포함합니다.
- 퀴즈 참여 과정:
  - 다른 사용자는 공개된 퀴즈 목록에서 원하는 퀴즈를 선택하여 응시합니다.
  - 퀴즈를 풀고 답안을 제출하면 점수가 계산되어 결과가 표시됩니다.
- 데이터베이스 설계:
  - Quiz 테이블: 퀴즈의 기본 정보를 저장합니다.
  - Question 테이블: 각 퀴즈에 속한 문제들을 저장합니다.
  - UserGrade 테이블: 사용자의 퀴즈 응시 결과를 저장합니다.

퀴즈 기능의 ERD 설계 고민

퀴즈 기능을 구현하면서 데이터베이스의 ERD(엔티티 관계 다이어그램)를 설계하는 데 시간이 많이 소모되었습니다. 특히 다음과 같은 부분에서 고민했습니다.

고민

- **퀴즈와 문제 간의 관계 설정**: 하나의 퀴즈에는 여러 개의 문제가 포함되며, 각 문제는 특정 퀴즈에 속합니다. 이를 어떻게 효율적으로 표현할 것인가?
- **사용자 응시 결과 저장 방법**: 사용자가 퀴즈를 응시한 결과(점수, 제출한 답안 등)를 어떻게 저장하고 관리할 것인가?

해결 방법

- 퀴즈와 문제 간의 관계 설정: Quiz 테이블과 Question 테이블을 생성하고, Quiz 테이블의 quiz_id를 Question 테이블에서 외래 키로 참조하도록 설계. 이를 통해 1 대 N 관계를 명확하게 표현
- 사용자 응시 결과 저장 방법: UserGrade 테이블을 만들어 사용자의 퀴즈 응시 결과를 저장. userNickname, quizSeq, score, submitDate 등의 컬럼을 포함하여 어떤 사용자가 어떤 퀴즈를 언제 응시했고, 몇 점을 받았는지 추적할 수 있도록 설계

결과

- 데이터베이스 구조를 명확하게 설계함으로써 퀴즈 기능을 구현하는 데 필요한 데이터 처리를 효율적으로 수행
- 사용자 응시 결과를 체계적으로 관리하여 학습 성과를 추적하고 피드백을 제공 O

비동기 WebRTC 작업에서 send() 함수에 await 누락으로 인한 문제

send() 함수에 await가 누락된 상태로 WebRTC 신호를 전송하면서, 비동기 작업의 순서가 어긋나는 현상이 발생했습니다.
예를 들어, createOffer() 함수에서 setLocalDescription이 완료되지 않은 상태에서 send()가 실행되며, 미완성 신호가 전달되는 상황이 빈번했습니다.

증상

- 신호 교환 과정에서 offer와 answer 전송이 제대로 이루어지지 않아 연결이 불안정하거나 실패하는 현상 존재
- 상대방의 비디오나 오디오 스트림이 전송되지 않거나, 실시간 통신 지연이 발생

원인 분석

- await 없이 send()가 호출됨으로써, setLocalDescription이 완료되기 전에 신호가 전송되어 수신자 측에서 예상하지 못한 신호를 받음
- 신호 전송의 순서와 타이밍이 어긋나면서 연결이 불안정해지고, 일부 신호가 누락되는 문제가 발생

해결 방법

- 각 비동기 작업이 완료된 후 send() 함수가 실행되도록 await 키워드를 추가하여 순서를 보장
- createOffer()에서 setLocalDescription이 완료된 후 await send()로 신호를 전송하여, 수신자 측에서 올바른 신호를 받을 수 있도록 설정

1대1 화상 면접시 상대방 화면 미송출 - HTTPS 및 SSL 인증서 문제

STUDYCLUB 1:1 면접 시스템에서 구현된 WebRTC 기반의 영상 통화 기능이 제대로 작동하지 않는 문제가 발생했습니다. 상대방의 화면이 송출되지 않거나 일시적으로 멈추는 현상이 빈번하게 발생했습니다.

증상

- 상대방 화면이 송출되지 않음. 브라우저에서 WebRTC 연결 차단
- 콘솔에서 WebRTC 연결 실패 로그 발생

원인 분석

- WebRTC는 보안이 확보된 연결(HTTPS)에서만 동작하지만, 서버에 SSL 인증서가 없거나 잘못 설정된 경우, WebRTC 통신이 불가능해졌습니다. 특히 최신 브라우저들은 HTTP 통신에서 WebRTC를 차단하기 때문에, 영상이 송출되지 않는 현상이 발생

해결 방법

- SSL 인증서를 발급받아 Tomcat에 등록 -> SSL 인증서를 server.xml에 설정하여 HTTPS를 활성화
- HTTP로 접속하더라도 HTTPS로 자동 리다이렉트되도록 설정
- 브라우저에서 HTTPS로 정상 접속되도록 설정. 이를 통해 WebRTC 연결이 안전하게 이루어짐

1대1 화상 면접시 상대방 화면 미송출 - 포트 포워딩 및 방화벽 문제

STUDYCLUB 1:1 면접 시스템에서 구현된 WebRTC 기반의 영상 통화 기능이 제대로 작동하지 않는 문제가 발생했습니다. 상대방의 화면이 송출되지 않거나 일시적으로 멈추는 현상이 빈번하게 발생했습니다.

증상

- 상대방의 화면이 송출되지 않거나, 송출이 일시적으로 중단되는 현상이 발생
- 방화벽이나 라우터에서 특정 포트가 차단되어 있어 WebRTC 연결이 중단

원인 분석

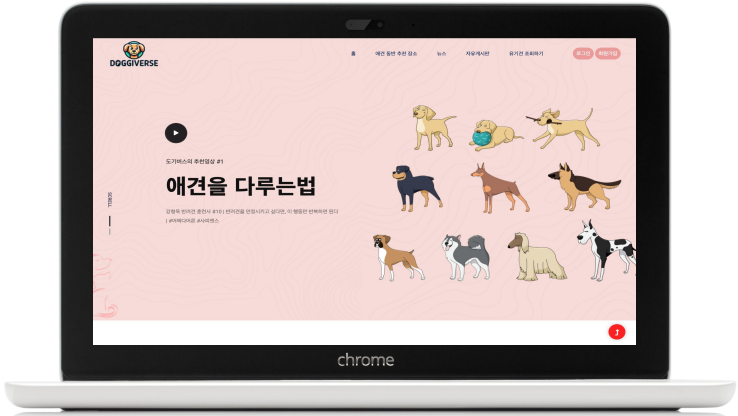
- WebRTC 통신은 특정 포트를 통해서 이루어집니다. 하지만 포트 포워딩이 설정되지 않거나 방화벽 규칙에서 해당 포트가 차단되면 P2P 연결이 불가능해져 영상 송출이 차단
- Tomcat 서버와 Oracle DB에서 포트 포워딩이 제대로 설정되지 않았거나 방화벽에서 관련 포트(HTTP: 8087, HTTPS: 8447, Oracle DB: 1521)가 차단된 경우 발생

해결 방법

- WebRTC 통신을 위한 포트(8447, 8087, 1521)을 방화벽과 라우터에서 허용
- 방화벽 설정에서포트가 허용되었는지 확인, 방화벽 규칙을 추가하여 포트를 개방
- 방화벽 인바운드 규칙에 특정 포털 포트 추가하여 TCP에 적용

DOGGIVERSE

웹앱 반려인과 반려견을 위한 컨텐츠 기반 커뮤니티 서비스



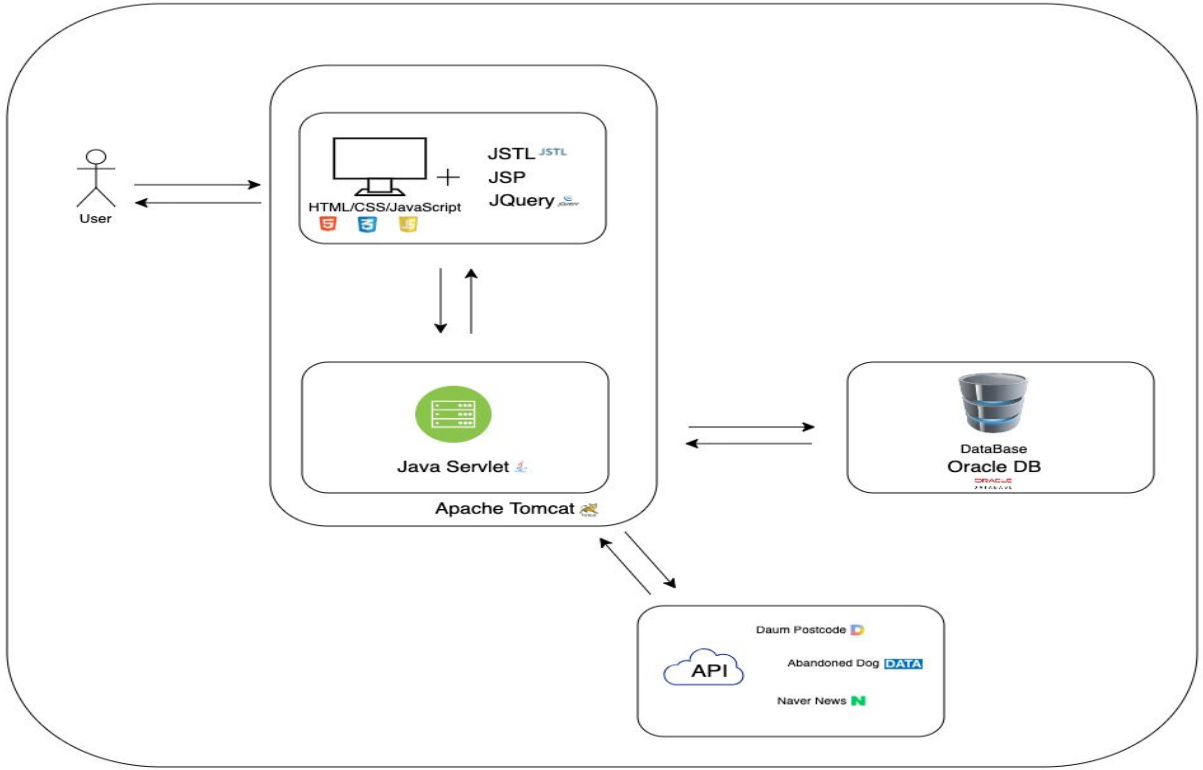
개발 기간	2024.5 ~2024.6
플랫폼	Web
개발 인원	3명
담당 역할	회원가입 및 마이페이지, 서비스 기획 및 방향성 설정

GitHub https://github.com/h3lim/Pet-Community-Web

개발 환경

언어	Java(JDK 17), HTML/CSS, JavaScript
서버	Apache Tomcat 9.0
프레임워크	Java Servlets & JSP
DB	Oracle DB
IDE	Eclipse,IntelliJ,SQL Developer
API,라이브러리	공공 유기견 API, Daum Postcode API, Gson,JSoup, JSTL,JQuery

시스템 아키텍처


User
HTML/CSS/JavaScript
JSTL
JSP
JQuery
Java Servlet
Apache Tomcat
DataBase
Oracle DB
API
Daum Postcode
Abandoned Dog DATA
Naver News

ERD 설계

USERS

Key	Logical	Physical	Domain	Type	Allow Null	Default Value	
PK	유저명	1	USERNAME	VARCHAR2(50)	N	PRIMARY KEY	
	비밀번호	2	PASSWORD	VARCHAR2(100)	N		
	이메일	3	EMAIL	VARCHAR2(100)	N	UNIQUE	
	이름	4	FULLNAME	VARCHAR2(100)	Y		
	닉네임	5	NICKNAME	VARCHAR2(50)	Y		
	개이름	6	DOG_NAME	VARCHAR2(50)	Y		
	날짜	7	CREATED_AT	TIMESTAMP	Y		DEFAULT CURRENT_TIMESTAMP

MY_BOARD

Key	Logical	Physical	Domain	Type	Allow Null	Default Value	
PK	시퀀스	1	BOARD_ID	NUMBER	N	PRIMARY KEY	
	제목	2	TITLE	VARCHAR2(30)	N		
	내용	3	CONTENT	VARCHAR2(2000)	Y		
	날짜	4	REG_DATE	SYSDATE	Y		
	조회수	5	HIT	NUMBER	N		DEFAULT 0
FK	유저명	6	USERNAME	VARCHAR2(50)	N	FOREIGN KEY	

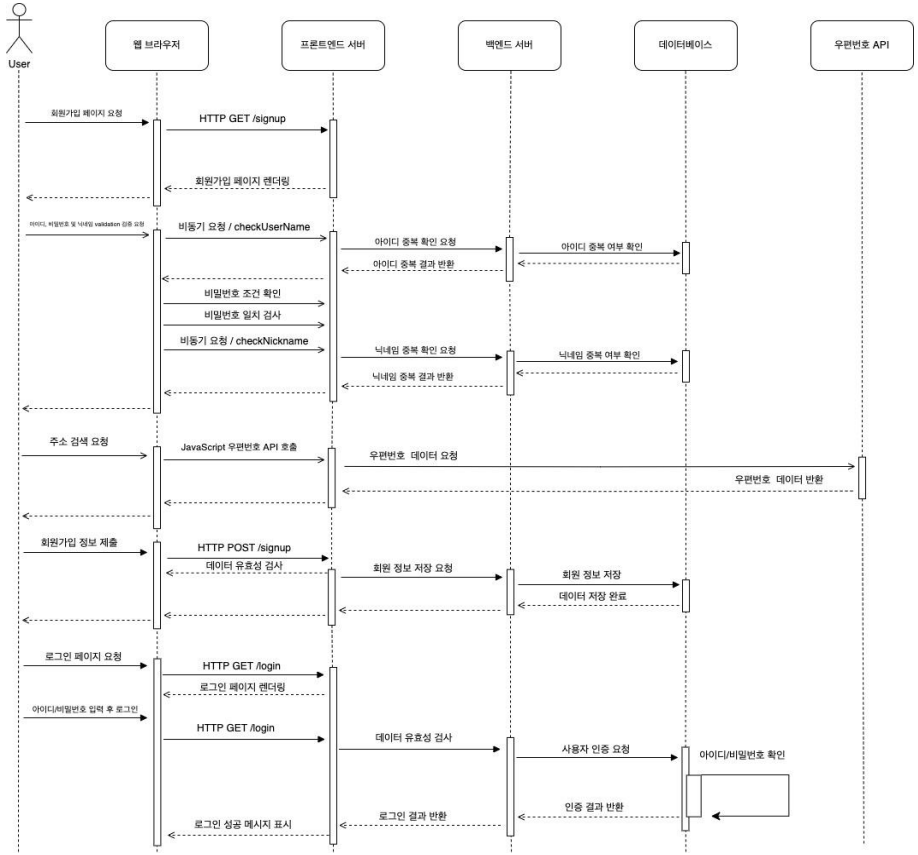
REPLY

Key	Logical	Physical	Domain	Type	Allow Null	Default Value	
PK, FK	유저명	2	USERNAME	VARCHAR2(50)	N	FOREIGN KEY	댓글을 작성한 사용자를 식별하는 외래 키
PK	시퀀스	1	RSEQ	NUMBER	N	PRIMARY KEY	댓글을 식별하는 고유키
	시퀀스	3	BOARD_ID	VARCHAR2(30)	N	FOREIGN KEY	댓글이 작성된 게시글을 식별하는 외래 키 -> REG_ID
	내용	4	REPLY	VARCHAR2(100)	Y		
	날짜	5	REGDATE	DATE	N		

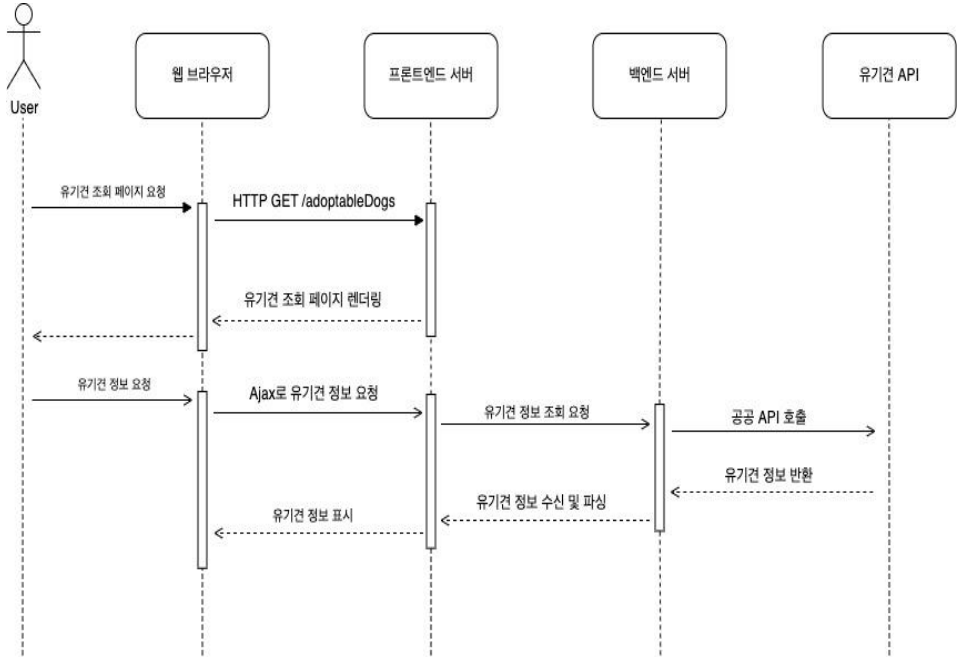
DOGGIVERSE

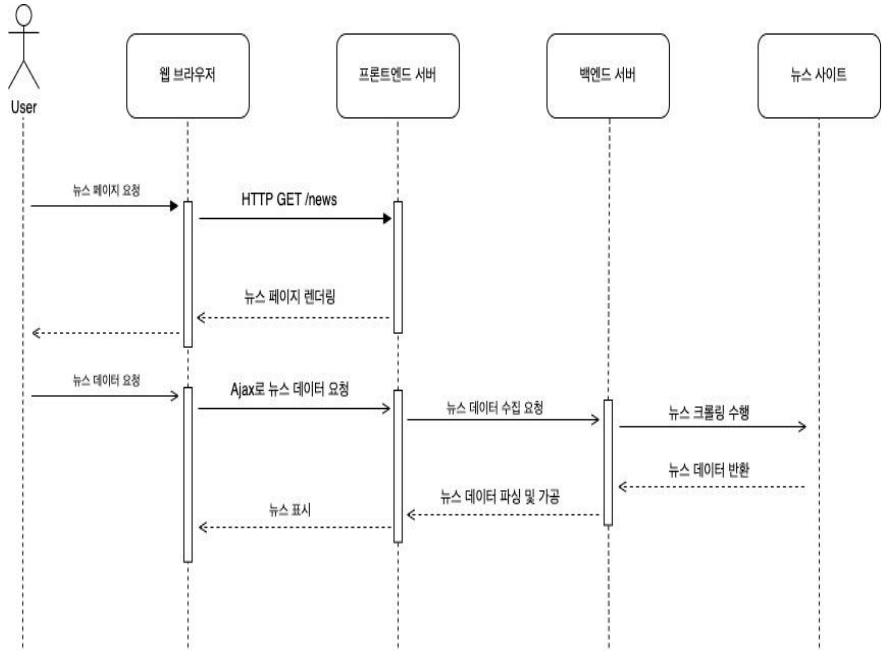
서비스 기획 및 방향성 설계 (3/4)

시퀀스 다이어그램 - 회원가입 및 로그인



시퀀스 다이어그램 - 유기견 조회, 반려견 관련 뉴스 조회





뉴스 크롤링 기능

- 사용자가 최신 반려견 관련 뉴스를 확인할 수 있도록 네이버 뉴스에서 데이터를 크롤링하여 제공합니다. 이를 위해 JSoup 라이브러리를 활용하여 웹 페이지를 파싱하고 필요한 정보를 추출합니다.

구현 상세

- NaverNewsCrawl 클래스에서 JSoup을 사용하여 네이버 뉴스 페이지의 HTML 코드를 가져옵니다.
- 가져온 HTML 문서에서 원하는 요소(예: 기사 제목, 링크, 이미지, 요약 등)를 CSS 선택자(selector)를 통해 추출합니다.
- 추출한 데이터는 NewsVO 객체에 담아 리스트로 관리합니다.
- CrawlServlet 서블릿이 클라이언트의 뉴스 요청을 처리하고, Gson을 사용하여 JSON 형태로 데이터를 반환합니다.
- News.jsp 페이지에서 AJAX를 통해 서블릿으로부터 데이터를 받아와 동적으로 뉴스 내용을 표시합니다.

유기견 정보 조회 기능

- 공공데이터포털의 유기동물 정보를 실시간으로 받아와 사용자에게 제공합니다. DogDataFetcher 클래스를 통해 외부 공공 API와 통신하여 데이터를 수집하고, 이를 사용자에게 보여줍니다.

구현 상세

- DogDataFetcher 클래스에서 HTTP 요청을 통해 공공데이터포털의 유기동물 정보를 가져옵니다.
- API 호출 시 필요한 서비스 키와 요청 파라미터를 URL에 포함합니다.
- 응답 받은 XML 데이터를 JSON으로 변환한 후, JSONObject를 사용하여 데이터를 파싱합니다.
- 추출한 데이터는 HashMap 객체에 담아 리스트로 관리합니다.
- dog.jsp 페이지에서 사용자로부터 시작일과 종료일을 입력받아 데이터를 조회하고, 페이지네이션을 구현하여 데이터를 표시합니다.

네이버 뉴스 크롤링 중 API Rate Limit 문제

네이버 뉴스 검색 페이지에서 데이터 크롤링 시, 서버에서 허용하는 최대 요청 횟수를 초과하면 Rate Limit에 걸려서 차단 가능성 존재. 특히, 여러 번 연속적인 요청이 발생하는 경우에 서버가 요청을 제한하거나 IP 차단 가능성 존재

증상

- 일정 시간 내에 여러 번 호출한 후, 페이지에 데이터가 로드되지 않거나 크롤링 자체가 실패하는 현상이 발생합니다. Jsoup.connect()가 IOException을 반환하거나, 네트워크 차단 알림이 발생

원인 분석

- 빈번한 요청은 서버에서 스팸 요청으로 간주 가능성 높아짐
- 요청 간 딜레이가 부족하여 API Rate Limit 초과 상황이 발생

해결 방법

- 코드에서 Thread.sleep(REQUEST_DELAY);를 사용하여 요청 간에 1초 지연을 적용해 일부 Rate Limit 문제를 완화 -> 요청 간 지연을 적절히 추가하여 서버의 부하를 줄이는 효과
- 특정 시간 내 동일한 요청의 중복 호출을 방지하기 위해 캐싱을 추가 -> 캐싱을 통해 자주 조회되는 데이터는 일정 시간 동안 재사용하여 API 호출 횟수를 줄임

유기견 API 데이터 응답 지연 및 간헐적 오류

API 호출 시 응답이 지연되거나 간헐적으로 호출 오류가 발생하여 데이터를 가져오지 못하는 문제가 발생

증상

- 간헐적 오류 발생 시 특정 오류 메시지가 페이지에 자주 출력되었고, 이를 해결하지 않으면 사용자에게 오류 메시지만 반복적으로 보이는 상황이 발생
- API에서 데이터를 반환하는 데 오랜 시간이 걸려 페이지 로딩 시간이 길어졌고, 일부 요청은 결국 타임아웃으로 실패

원인 분석

- 공공 API의 경우, 서버 상태나 응답 속도가 사용량에 따라 변동되며 비정상 상태가 발생할 가능성 존재

해결 방법

- 요청 실패 시 일정 시간 대기 후 재요청을 수행하도록 MAX_RETRIES와 RETRY_DELAY_MS 값을 지정하여 코드에 추가
- 재시도 횟수는 3번으로 제한해 과도한 요청을 방지하고, 대기 시간은 2초로 설정해 서버에 무리가 가지 않도록 설정