PORTFOLIO

끊임없이 새로워지는 기술과 환경,그 안에서 빠르게 적응하고 성과를 내는 유연한 김세헌입니다

KIM, SEHEON

#FULL STACK

kshandy0221@gmail.com
010-3738-5871

김세헌

FULL STACK

생년월일 : 2001. 02. 21

주소 : 서울 강서구 양천로 62길 41

이메일 : kshandy0221@gmail.com

전화번호 : 010-3738-5871

GitHub: https://github.com/NewOld21

Velog: https://velog.io/@shk0221/posts

학력

2020. 03 ~ 2025. 02 성결대학교 컴퓨터공학과

2017. 03 ~ 2020. 02 마포고등학교 졸업

수상 및 자격

정보처리기사 필기 합격 (2024.07)

국가우수이공계 장학생 (2023.05)

스킬 – USE WELL

- Backend: Python, TypeScript , JavaScript, Node.js, Nest.js,
- Frontend: React, HTML, CSS
- Database: PostgreSQL
- Tooling: GitHub, Slack, Notion

스킬 – HAVE USE

- Backend: C, Java, Spring
- Frontend: Tailwind CSS
- Database: MySQL, MongoDB
- Tooling: GitHub, Figma, Postman

프로젝트 : WaveFlow

프로젝트 정보

- 주소 : https://waveflow.pro
- 기간 : 1개월
- 팀원 : 5명
- 사용기술 : Nest.js, React, PostgreSQL, TypeScript, Tailwind CSS, Python 등
- GItHub : https://github.com/Team-Honey-Badgers

역할

Backend 개발

- NestJS 기반 RESTful API 설계 및 구현
- 데이터베이스 설계
- JWT 인증 및 Google OAuth 연동
- AWS S3 파일 업로드, 이메일 발송 시스템

Frontend 개발

- React 기반 음악 협업 플랫폼 UI 구현(DashBoard,Track)
- WebSocket 실시간 협업 기능
- 사용자 인터페이스 및 상태 관리
- 접근 권한 가드

배경 및 필요성

- 음악 작업자들은 버전 관리를 찐막, 최최종 같은 파일명으로 대체하고 있음
- 협업 시 카카오톡, 이메일, 구글 드라이브 등 비효율적인 방식에 의존
- 개발자들의 Git처럼, 음악 협업에 특화된 형상 관리 도구의 부재를 해결하고자 함

개요

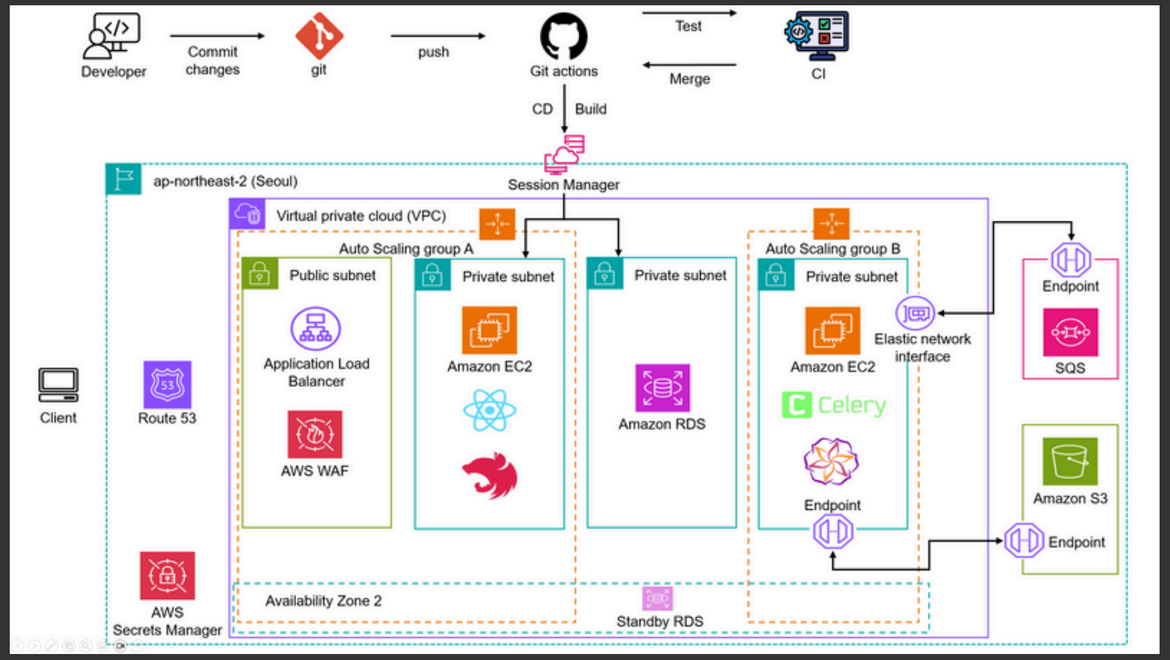
- 멀티트랙 기반 실시간 협업 오디오 편집 플랫폼
- Stage → Stem → Version 구조로 버전 관리 및 리뷰 워크플로우 지원
- 실시간 알림, 댓글, 오디오 비교 기능까지 통합한 종합 오디오 협업 환경 제공

주요기능

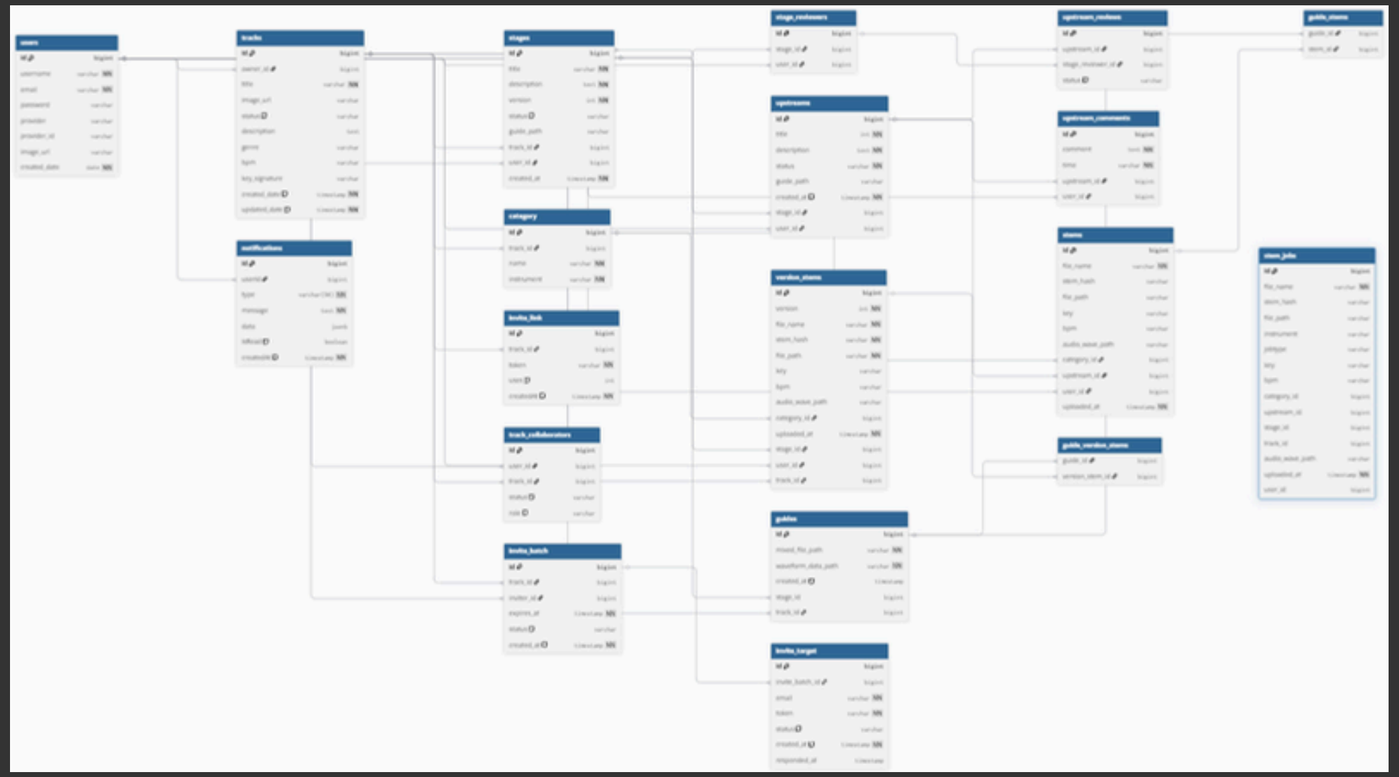
- 멀티트랙 오디오 스트리밍 및 재생 기능 제공
- 실시간 협업자 초대 및 권한 관리 시스템 구현
- 단계별 리뷰 워크플로우 및 버전 관리 지원
- 실시간 알림, 댓글, PR 기반 오디오 비교 및 승인 기능 제공

아키텍처 및 ERD

시스템 아키텍처



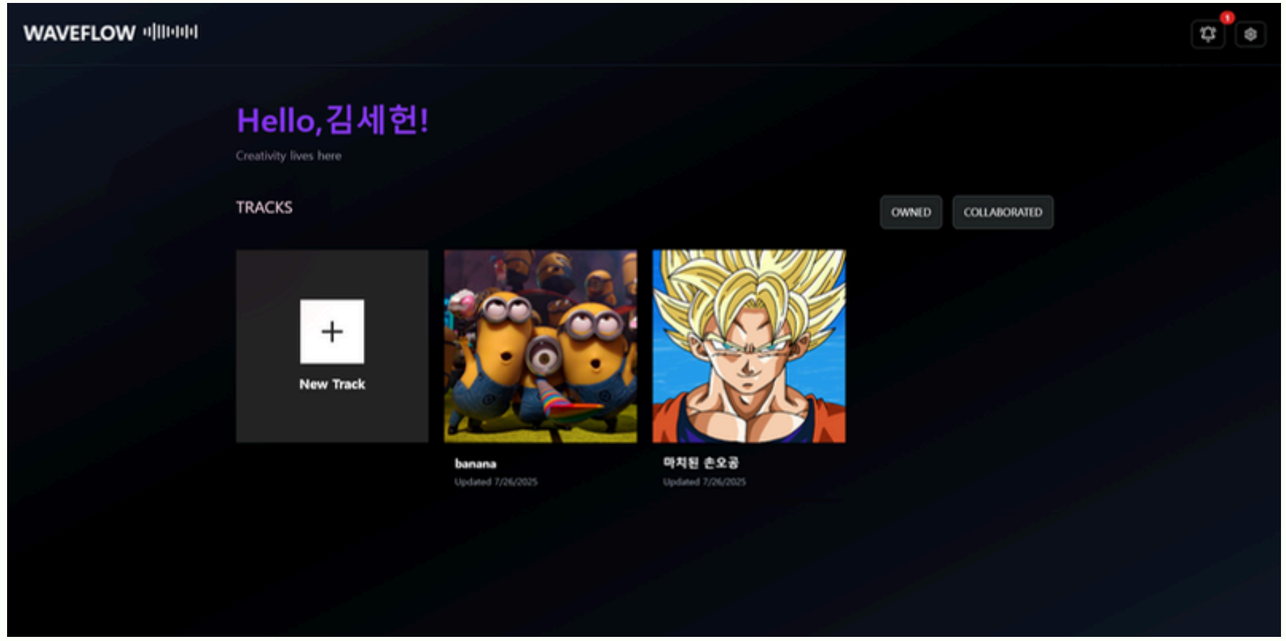
ERD



자세한 ERD 내용 : https://velog.io/@shk0221/WaveFlow-ERD

구현 기능 및 개발과정

대시보드 페이지



중점 구현 기능

JWT 기반 인증/인가 시스템
실시간 알림 시스템
페이지별 접근 권한 검증

대시보드 페이지

- 트랙 CRUD 시스템 (생성/조회/수정/삭제)
- 멀티파트 파일 업로드를 통한 Stem 파일 관리
- 반응형 트랙 카드 UI 및 전체 페이지 레이아웃

개발과정

보안: JWT 토큰 기반 stateless 인증, 가드 패턴으로 라우트 보호
실시간성: WebSocket + JWT 검증으로 안전한 실시간 통신
권한 관리: 트랙별 협업자 권한 체계 구현

대시보드 페이지
Backend

- NestJS로 Track/Stem 엔티티 설계, AWS S3 Presigned URL 기반 대용량 파일 업로드 API 구현

Frontend

- React로 드래그앤드롭 파일 업로드, 실시간 업로드 진행률 표시, 트랙 관리 인터페이스 구현

통합

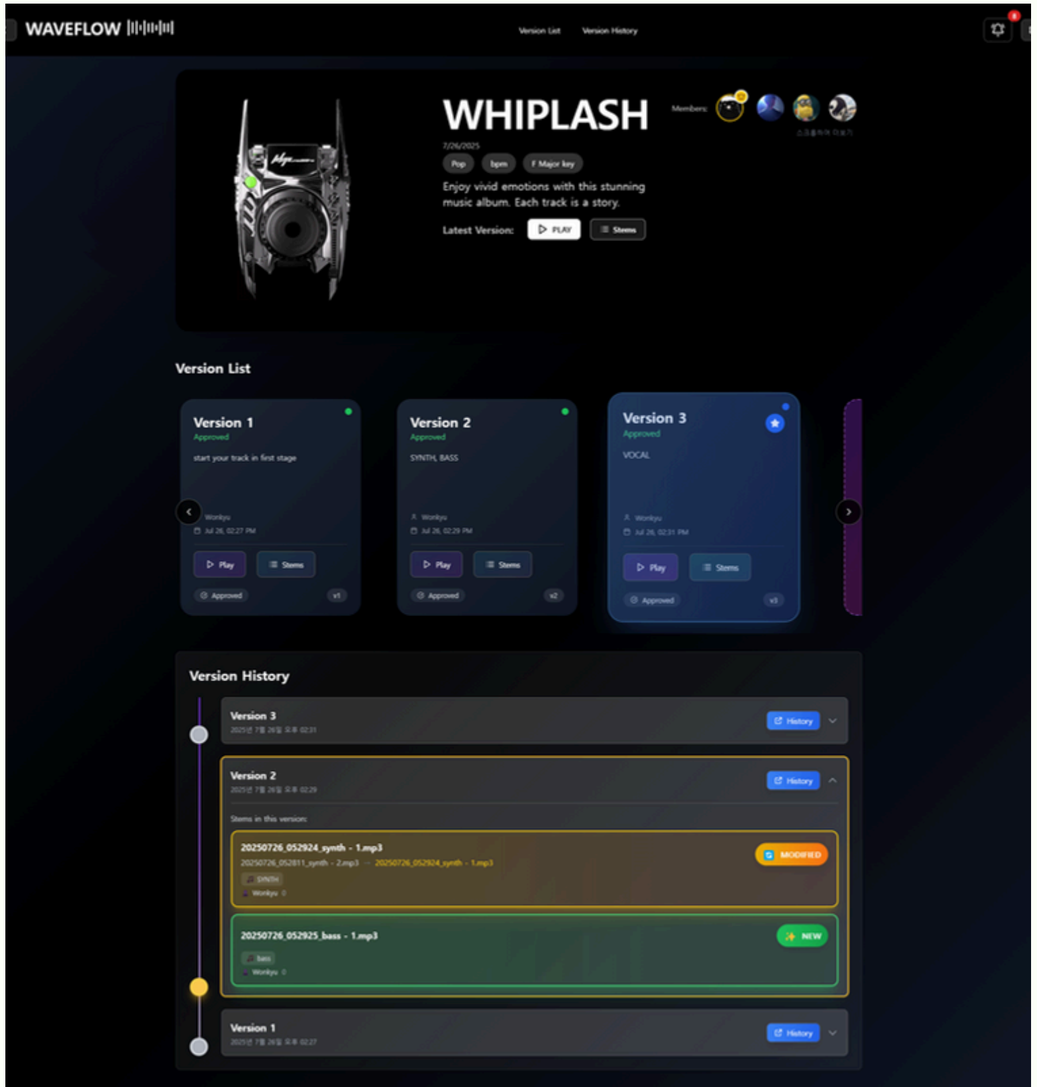
- RESTful API와 프론트엔드 상태 관리를 연동하여 실시간 트랙 목록 업데이트

기술 스택 선정 이유

- AWS S3: 대용량 오디오 파일의 안정적 저장 및 CDN 활용
- Presigned URL: 서버 부하 없이 클라이언트에서 직접 업로드
- JWT: 확장 가능한 stateless 인증
- WebSocket: 실시간 협업 환경 구축

구현 기능 및 개발과정

트랙 페이지



중점 구현 기능

트랙 페이지

- 트랙 정보 시스템: 트랙 메타데이터 표시 및 실시간 업데이트
- 버전 관리 시스템: 전체 버전 리스트 조회 및 타임라인 형태 히스토리
- 원클릭 롤백: 이전 버전으로 즉시 복원 기능
- Stem 파일 관리: 버전별 개별 Stem 파일 모달 뷰어 및 재생
- 통합 오디오 플레이어: 멀티트랙 동시 재생 및 개별 제어
- Stage 생성 시스템: 리뷰 단계 생성 및 워크플로우 관리

개발과정

트랙 페이지

Backend

- 데이터베이스 설계: Track-Version-Stem 관계형 모델 구현, 버전 히스토리 추적을 위한 스키마 최적화
- API 엔드포인트: 트랙 상세 조회, 버전 관리, 롤백, Stem 파일 처리 등 15+ 엔드포인트 구현
- 버전 관리 로직: 트랙 상태 변경 시 자동 버전 생성, 롤백 시 데이터 무결성 보장
- Stage 시스템: 리뷰 워크플로우를 위한 Stage 엔티티 및 비즈니스 로직 구현

Frontend

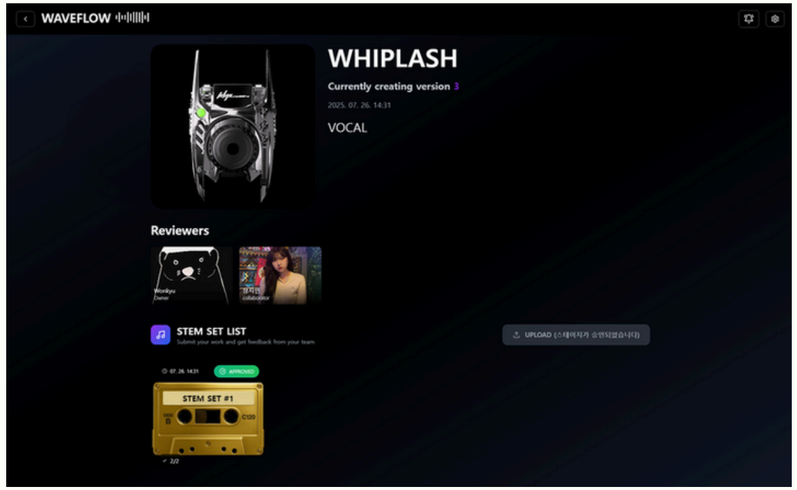
- 페이지 레이아웃: 트랙 정보, 버전 리스트, 히스토리를 직관적으로 배치한 반응형 UI 설계
- 버전 히스토리: 타임라인 형태의 시각적 버전 추적, 각 버전별 변경사항 표시
- 모달 시스템: Stem 파일 상세 보기 모달의 데이터 연동 및 API 통합 (디자인 제외)
- 인터랙션: 드래그앤드롭, 원클릭 롤백, 실시간 상태 업데이트

데이터 연동 및 상태 관리

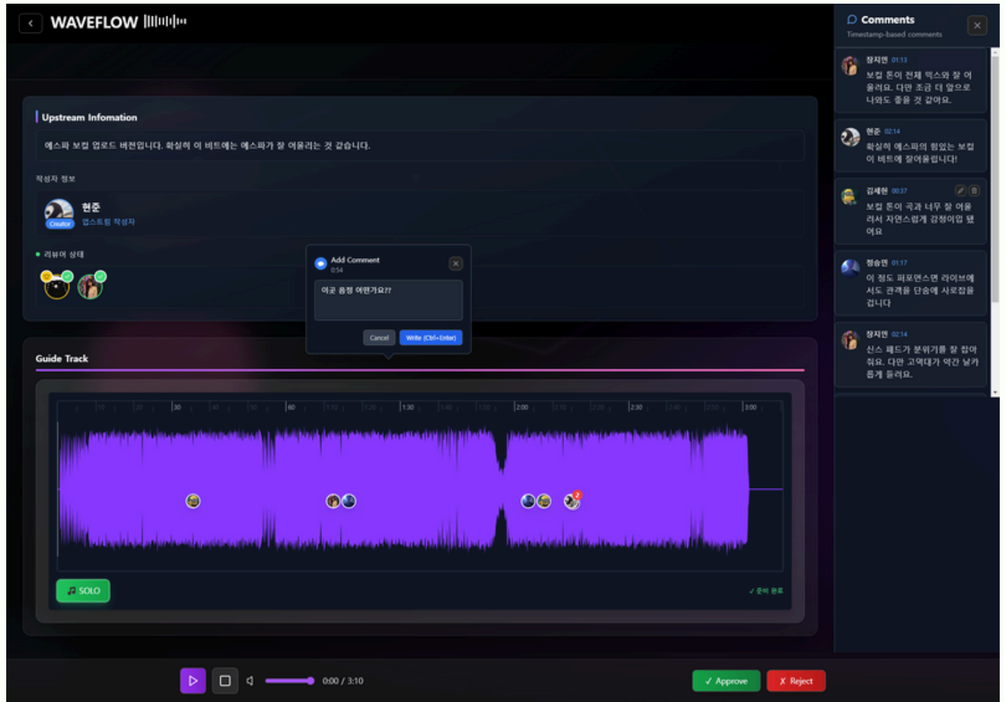
- API 통합: 모든 트랙 관련 API 엔드포인트와 프론트엔드 연결
- 실시간 업데이트: 버전 변경, 롤백 시 즉시 UI 반영
- 에러 핸들링: 파일 업로드 실패, 네트워크 오류 등 예외 상황 처리

구현 기능 및 개발과정

스테이지 페이지



스템세트 리뷰 페이지



중점 구현 기능

스테이지 페이지

- Stem 파일 업로드 시스템
- 트랙/스테이지 정보 조회 API
- Stem-set 데이터 관리

스템세트 리뷰 페이지

- 리뷰 상태 관리, 승인/거부 처리
- 댓글 시스템
- Upstream 데이터 조회
- 리뷰 버튼 연동 및 데이터 페칭

개발과정

스테이지 페이지

Backend

- 파일 업로드 : AWS S3 기반 Stem 파일 업로드 엔드포인트 구현
- 데이터 조회 : Stage 상세 정보 및 연관된 Track 데이터 조회 API 개발
- Stem-set 관리 : 리뷰용 Stem 파일 그룹 관리 로직 구현

스템세트 리뷰 페이지

- 리뷰 시스템: Review 엔티티 설계, 승인/거부 상태 변경 로직 구현
- 댓글 기능: 리뷰별 댓글 CRUD API 개발
- 데이터 연동: Upstream 데이터 조회 및 리뷰 상태 실시간 업데이트
- Frontend 연동: 리뷰 버튼 클릭 시 API 호출 및 상태 반영

트러블 슈팅

분석 서버 부하 분산을 위한 개별 파일 완료 처리

상황

- 클라이언트가 여러 파일을 병렬 업로드한 뒤, 전체 업로드 완료 후 일괄 서버 알림
- 서버는 모든 파일을 한 번에 SQS로 분석 서버에 전달
- 분석 서버에 분석 요청이 몰리며 순간적인 부하 집중 발생

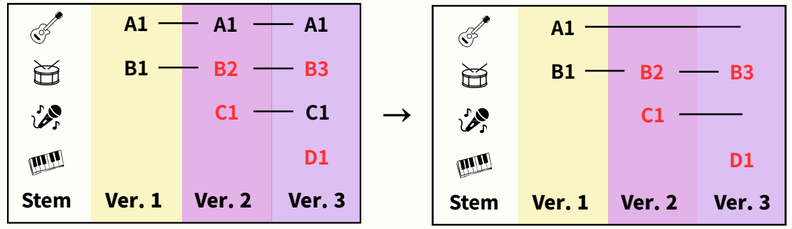
해결

- 개별 파일 완료 시점마다 서버에 알림 전송하도록 구조 변경
- 서버는 해당 파일을 즉시 SQS로 분석 서버에 전송
- 분석 요청이 시간차를 두고 분산
- 그 결과, 분석 서버 부하 완화처리와 속도 및 안정성 향상

해당 경험을 통해 알게된 점

- 병렬 구조에서도 이벤트 타이밍 조정만으로 부하를 효과적으로 분산할 수 있음을 경험
- 단순한 기능 구현보다, 흐름을 제어하는 설계가 시스템 성능에 더 큰 영향을 미침
- 실시간 처리와 안정성을 동시에 고려한 구조가 서비스 운영 품질에 직결된다는 점을 체감

증분 스냅샷 도입



상황

- 초기에는 Stage마다 전체 Stem 파일을 스냅샷 형태로 복제 저장하고 있었음
- 동일한 Stem 파일이 중복 보관되며 스토리지 낭비가 발생함
- 롤백 시 전체 스냅샷을 일괄 복원해야 했기에 복구 시간이 지연되는 문제가 있었음

해결

- Stage마다 수정되거나 추가된 Stem만 별도로 저장하는 증분 스냅샷 구조로 변경
- 변경 없는 파일은 이전 Stage의 레퍼런스를 참조하도록 설계
- 롤백 시 Cascade Delete를 활용해 이후 Stage만 삭제하고, 증분된 스냅샷만 복원
- 그 결과, 버전별 저장 용량 약 60% 감소, 복구 속도 약 2배 개선

해당 경험을 통해 알게된 점

- 기능 구현 외에도 스토리지 효율성과 복구 성능까지 고려한 구조 설계의 중요성을 체감
- 전체 복제보다 증분 저장이 더 나은 선택이 될 수 있다는 아키텍처적 인사이트를 얻게 됨
- 실제 사용 흐름(롤백, 병합 등)을 가정한 설계가 현실적인 문제 해결로 직결됨
- 기술적 개선이 사용자 경험과 운영 효율 모두에 긍정적 영향을 준다는 점을 직접 경험

프로젝트 성과 및 회고

프로젝트 성과

항목	내용
실시간 음악 협업 플랫폼	멀티트랙 오디오 스트리밍과 버전 관리 시스템 구현
RESTful API 및 DB 모델링	S3 멀티파트 업로드 및 Presigned URL 활용으로 안정적 전송 처리
분석 서버 부하 분산	30개 이상의 API 구현과 Track-Version-Stem-Review 구조 설계
실시간 통신 시스템 구축	JWT 인증 기반 WebSocket으로 알림 및 협업 기능 실시간 처리

프로젝트 회고

아쉬운 점	개선 방향
파일 검증 로직 미흡	파일 형식/크기 제한, 바이러스 스캔, S3 버킷 CORS 및 접근 정책 강화
DB 조회 시 N+1 문제와 인덱스 부족으로 성능 저하 발생	인덱스 최적화, Eager Loading, Redis 기반 캐싱 적용
모니터링 및 이상 탐지 미흡	응답 시간/쿼리 로깅, 비정상 API 호출 감지, 보안 이벤트 알림 시스템 구축

CONTACT

kshandy0221@gmail.com
010-3738-5871

THANK YOU