2025 년 전기 졸업과제 중간보고서

RAG 를 활용한 컨테이너 기반 마이크로서비스 운영 환경 관리 지원 시스템

팀명 : 트리톤

201914116 김휘수

202055645 신세환

202255663 설종환

지도교수 : 염근혁 (인)

목차

1. 과제의 목표

1) 과제 배경

마이크로서비스 아키텍처(MSA, Microservice Architecture)는 시스템을 소규모의 독립적인 기능 요소로 구분하여 서비스를 배포할 수 있게 만드는 구조이다. MSA 는 주로 컨테이너 오케스트레이션 기술(예: Kubernetes, Docker Swarm)을 통해 구현되며, 컨테이너 오케스트레이션 기술을 활용하여 마이크로서비스 배포를 수행하기 위해서는 배포 명세(Kubernetes Template, Dockerfile 등)를 활용하여 컨테이너가 동작하는 환경을 구축할 필요가 있다.

컨테이너를 활용한 마이크로서비스 아키텍처는 마이크로서비스 간의 약결합을 통해 기능 흐름을 구축하게 된다. 컨테이너 기반 마이크로서비스 아키텍처 구축 시 약결합 수행을 위해 네트워크 인터페이스 명세, 컨테이너 이미지 명세 등을 정확하게 작성하고 배포해야 한다. 그러나, 기 작성된 명세를 활용하거나 컨테이너 오케스트레이션 기술에 대한 이해도가 부족한 사용자는 컨테이너 기반 마이크로서비스에서 발생하는 오류를 파악하고 해소하기 어렵다. 또한, LLM 과 같은 개발을 위한 보조 도구를 활용하더라도 조직 내부 데이터(커스텀 이미지, 핵심 운영 정책 등)는 접근하기 어렵기에 요구사항에 적합한 마이크로서비스 애플리케이션 개발이 어렵다는 문제가 있다. 추가적으로, 컨테이너 기반 마이크로서비스 배포 이전에는 알기 어려운 동적 운영 정보(CPU/메모리 사용량 등)에 대해 지원할 수 있는 체계가 부족하다.

따라서, 본 과제에서는 RAG(Retrieval-Augmented Generation)를 활용한 컨테이너 기반 마이크로서비스의 운영 지원 플랫폼을 제안한다. RAG 는 신뢰할 수 있는 외부 데이터베이스를 참조 및 활용하여 LLM 이 생성하는 답변의 정확도 및 활용성을 향상시키는 기법으로, 본 과제에서는 자연어 기반의 사용자 질의가 주어질 때 RAG 를 통해 조직의 내부 데이터와 동적 운영 정보를 활용하여 명세 생성과 수정안을 제공하는 컨테이너 기반 마이크로서비스 운영 지원 기술을 목표로 한다.

2) 과제 세부 목표

① RAG 기반 마이크로서비스 배포 명세 생성 자동화 기법 연구

② 기능 동작 명세 기반 마이크로서비스 약결합 명세 생성 방법 도출

③ 운영 정보 기반 마이크로서비스 운영 환경 재구성 및 관리 기술 제공

2. 요구사항 및 제약사항 분석에 대한 수정사항

1) 기존 요구사항 및 수정사항

① 기능적 요구사항

표 1 은 시스템이 제공해야 할 기능들에 대한 기존 요구사항을 나타낸다.

표 1 기능적 요구사항

기능		설명
사용자 정보 관리	사용자 정보 생성	사용자는 회원가입을 통해 ID, 비밀번호, SSH 접속을 위한 인증 키와 IP 주소, AI 서비스 API 키 정보를 입력해서 계정을 생성할 수 있어야 한다.
	사용자 정보 수정	사용자는 등록한 비밀번호, SSH 접속 정보, API 키를 수정할 수 있어야 한다.
	사용자 인증	사용자는 계정 정보를 통해 시스템에 로그인하고, 시스템은 인증된 사용자만 주요 기능에 접근할 수 있도록 해야 한다.
시스템 초기화	로그 수집 환경 구축	사용자는 ELK 스택 중 로그 수집기(Filebeat, Logstash)를 배포해야 하고, 시스템은 수집되는 로그를 벡터 DB 로 저장해야 한다.
	비공개 데이터 저장	사용자는 시스템에 사내 정책, 커스텀 이미지 등의 비공개 데이터를 업로드 해야 하고, 시스템은 비공개 데이터를 벡터 DB 로 저장해야 한다.
서비스 배포 관리	서비스 배포 파일 작성 지원	시스템은 사용자의 질의를 바탕으로, 내부 데이터 DB 및 웹 문서 검색 결과를 참고해서 각 설정 항목에 주석과 해설이 포함된 배포 파일을 새로 생성하거나 개선안을 제안할 수 있어야 한다.
	서비스 운영 모니터링	시스템은 수집된 로그에 대해 주기적으로 분석을 수행하여 오류 로그가 발생하거나 리소스 사용량의 분석 결과로 도출한 권장 설정 값보다 현재 리소스 사용량이 많을 경우 이상 동작으로 판단할 수 있어야 한다.
	배포 파일 수정 방안 제시	시스템은 이상 동작이 감지된 경우, 내부 데이터 DB, 로그 DB, 웹 문서 검색 결과를 참고하여 배포 파일 수정 방안을 사용자에게 제공해야 한다.
작업 이력 조회	사용 이력 조회	사용자는 시스템이 답변으로 제공한 배포 명세 이력을 조회할 수 있다.

표 2는 시스템이 제공해야 할 기능적 요구사항에 대한 추가사항이다.

표 2 기능적 요구사항 추가사항

기능		설명
사용자 정보 관리	사용자 정보 삭제	사용자는 등록한 계정을 삭제할 수 있어야 한다.
사용자 애플리케이션 관리	애플리케이션 등록	사용자는 관리할 MSA 애플리케이션을 시스템에 등록할 수 있어야 한다.
	애플리케이션 삭제	사용자는 등록된 프로젝트와 관련된 모든 정보(배포 이력, 내부 데이터 등)를 시스템에서 삭제할 수 있어야 한다.
시스템 초기화	SSH 인증 설정	사용자는 등록된 애플리케이션에 접근하기 위한 SSH 인증 키와 IP 주소 등의 정보를 설정할 수 있어야 한다.
서비스 배포 관리	서비스 배포 파일 수정 지원	사용자의 배포 파일이 문법 오류, 약결합 오류 등으로 배포에 실패하는 경우, 시스템은 비공개 데이터 DB 및 웹 문서 검색 결과를 참고해서 오류가 수정된 배포파일을 제공할 수 있어야 한다.
	서비스 배포 파일 동적 개선 지원	시스템은 이상 동작이 감지된 경우, 비공개 데이터 DB, 로그 DB, 웹 문서 검색 결과를 참고하여 배포 파일 수정 방안을 사용자에게 제공할 수 있어야 한다.
작업 이력 관리	작업 이력 삭제	사용자는 시스템이 답변으로 제공한 배포 명세 이력을 삭제할 수 있어야 한다.

② 비기능적 요구사항

다음 표 3 은 시스템이 만족해야 할 비기능적 요구사항이다.

표 3 비기능적 요구사항

요건	설명
성능	- 시스템은 로그 이상 탐지 주기를 최대 3 분 이내로 유지해야 함. - **시스템은 배포 명세에 대한 오류 식별 기능의 테스트 케이스 검증 단계에서 F1 score 0.8 이상 달성해야 함.** • $F1\_score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$ • $Precision = \frac{TP}{TP + FP}$ • $Recall = \frac{TP}{TP + FN}$ • TP: 테스트 시나리오에 오류가 있고 시스템이 오류를 탐지함 • FP: 테스트 시나리오에 오류가 없지만 시스템이 오류가 있다고 판단함 • FN: 테스트 시나리오에 오류가 없고 시스템이 오류가 없다고 판단함
신뢰성	- 시스템은 사용자의 입력 오류(유효하지 않은 API 키 또는 접근 자격)에 대해서도 오류 없이 예외를 처리하고 사용자에게 유의미한 오류 메시지를 제공해야 함.
보안성	- 시스템은 비공개 데이터, 로그 등 민감 데이터의 전송 구간 데이터 보안을 위해 TLS 1.2 이상을 사용하는 HTTPS 를 통해 통신을 암호화해야 함.
안정성	- 전체 기능을 독립적으로 구성해서 시스템 장애시에도 일부 기능은 제한된 상태에서 지속 운영될 수 있어야 함.
가용성	- 시스템은 정기 점검이나 예기치 못한 장애 발생 시에도 서비스 중단 시간을 최소화할 수 있도록 자동 복구 또는 빠른 수동 조치가 가능해야 한다.

3. 설계 상세화 및 변경 내역

1) 시스템 설계 및 명세

① UML 클래스 다이어그램

- 사용자 정보 관리

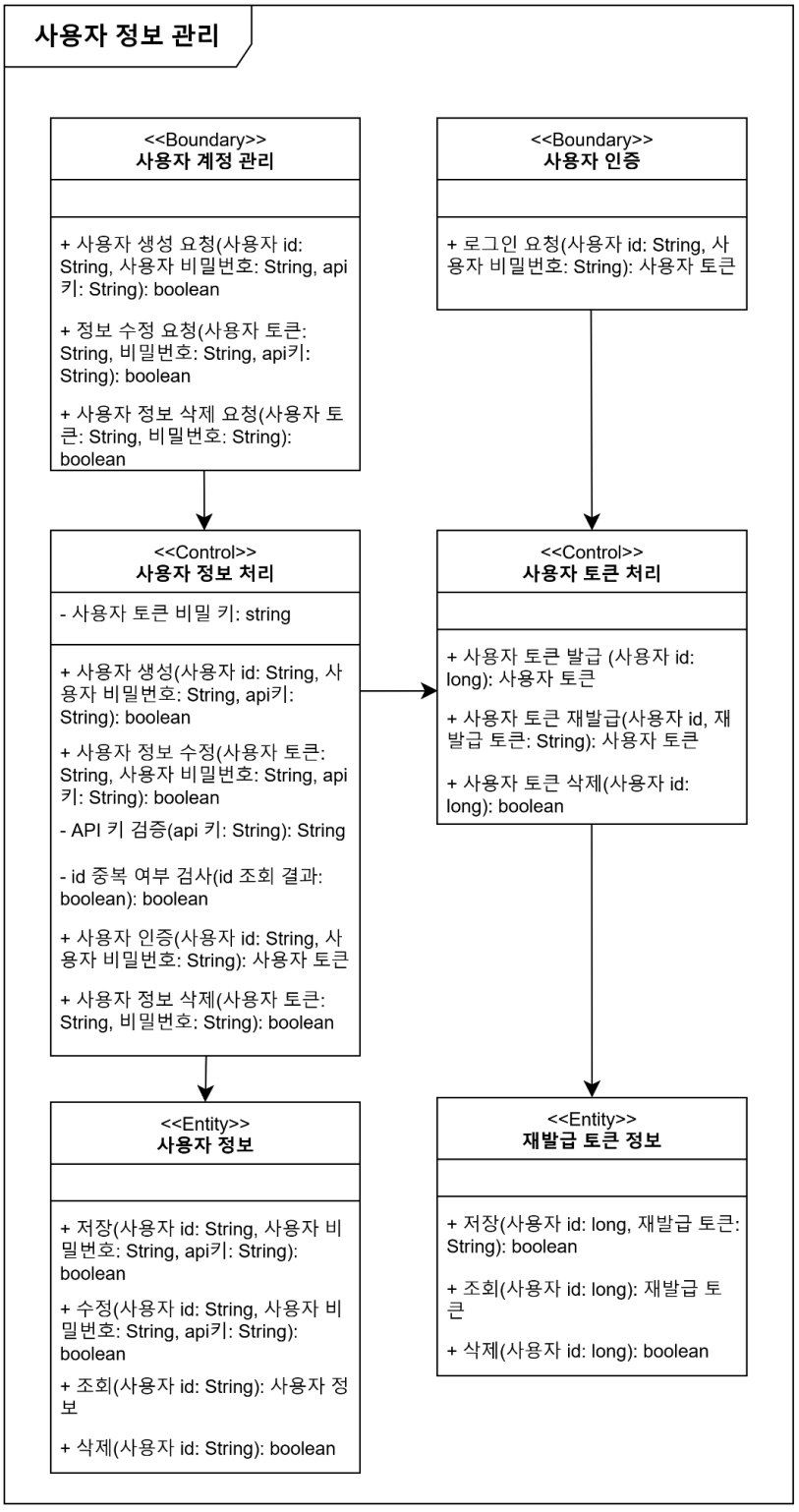

그림 1 사용자 정보 관리 클래스 다이어그램

그림 1 은 사용자 정보 관리에 필요한 요소들을 클래스 다이어그램으로 나타낸 것이다.

사용자 계정 관리 클래스는 사용자가 사용자 생성 요청이나 정보 수정 요청, 사용자 정보 삭제 요청을 진행할 때 필요한 정보를 입력받고 이를 사용자 정보 처리 클래스에 전달하는 인터페이스이다. 사용자 인증 클래스는 사용자가 로그인 시 인증 요청을 처리한다. 사용자 정보 처리 클래스에서 전달받은 사용자 데이터를 데이터베이스에 저장하는 로직을 호출하거나 사용자 인증에 관한 로직을 수행한다. 사용자 정보 클래스는 사용자 계정 데이터를 저장, 조회 및 수정하는 역할을 수행한다.

2025 전기 졸업과제

- 사용자 애플리케이션 관리

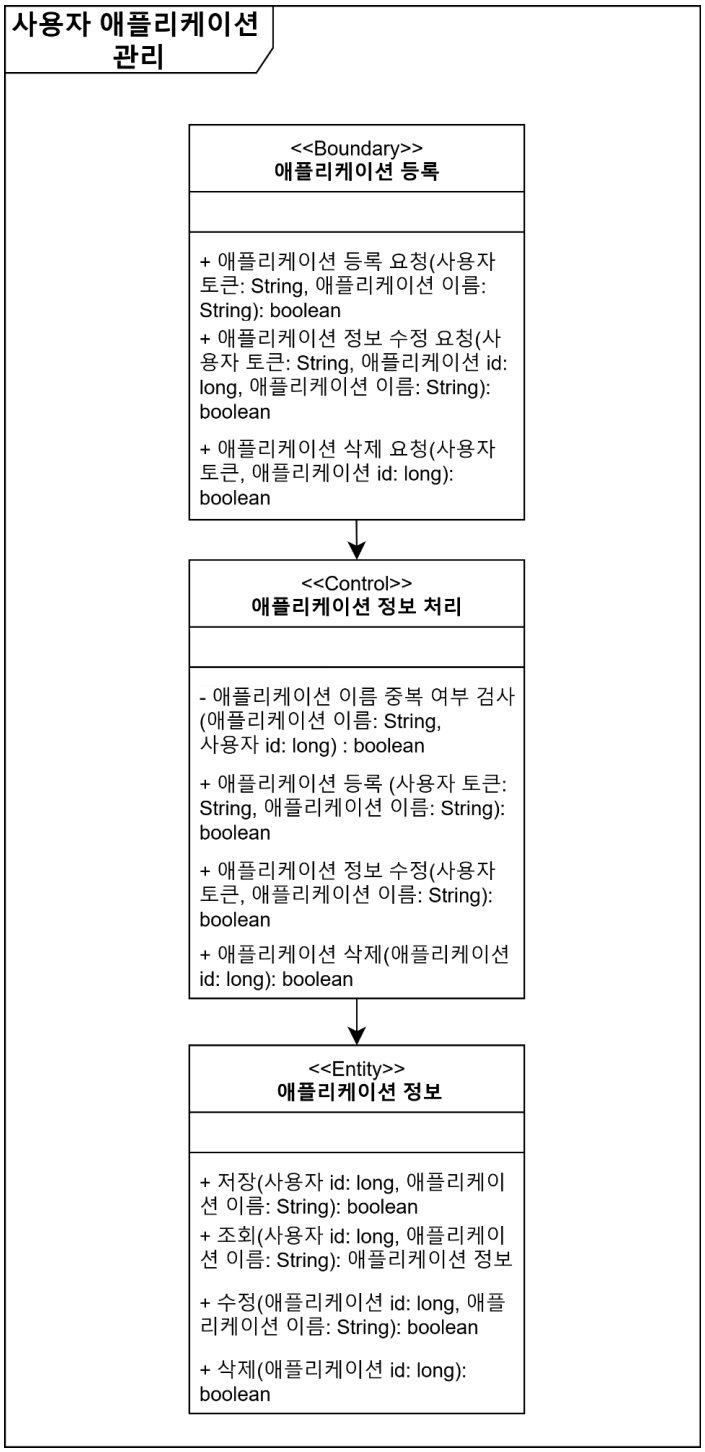

그림 2 사용자 애플리케이션 관리 클래스 다이어그램

그림 2 는 사용자 애플리케이션 관리에 필요한 요소들을 클래스 다이어그램으로 나타낸 것이다.

애플리케이션 등록 클래스는 사용자가 애플리케이션 정보를 등록, 수정, 삭제 시 관련 데이터를 입력받아 애플리케이션 정보 처리 클래스로 전달하는 인터페이스이다. 애플리케이션 정보 처리 클래스는 애플리케이션 등록, 수정 및 삭제 요청 로직을 수행한다. 애플리케이션 정보 클래스는 애플리케이션 관련 데이터를 저장, 조회, 수정 및 삭제하는 역할을 한다.

- 시스템 초기화

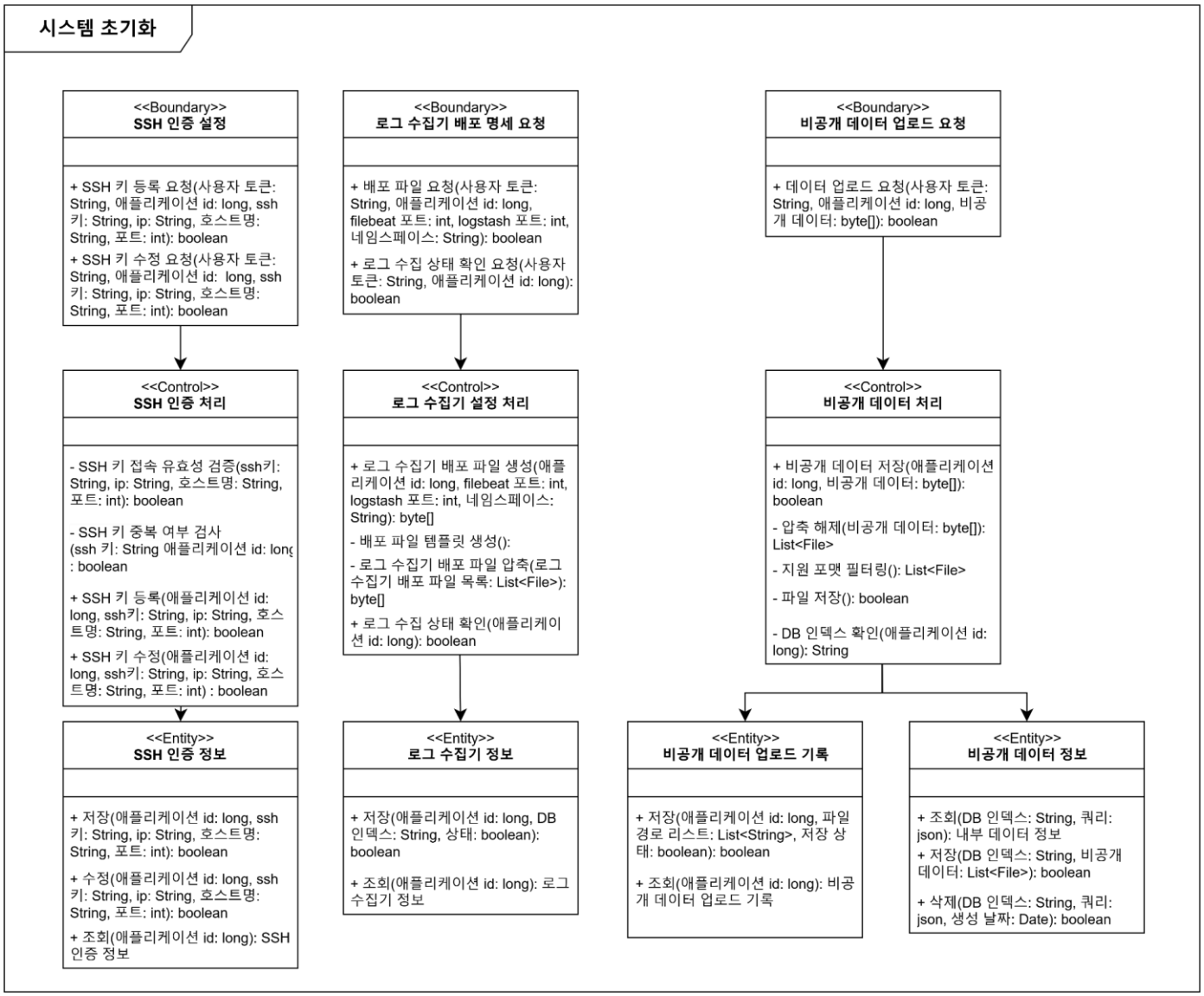

그림 3 시스템 초기화 클래스 다이어그램

그림 3 은 시스템의 초기화에 필요한 요소들을 나타낸 클래스 다이어그램이다.
SSH 인증 설정 클래스는 사용자 애플리케이션의 원격 접속 인증 정보를 설정할 때 사용하는 인터페이스이다. SSH 인증 처리 클래스는 SSH 인증 설정 클래스의 요청을 받아 등록이나 수정 시 입력된 SSH 키를 통해 테스트 접속을 시도하여 접속 유효성을 검증하고 처리한다. SSH 인증 정보는 애플리케이션 id, SSH 키, IP, 호스트명, 포트로 구성된다.

로그 수집기 배포 명세 요청 클래스는 사용자가 애플리케이션에 로그 수집기를 배포하고 관리하기 위해서 사용하는 인터페이스이다. 사용자가 filebeat 와 logstash 의 포트번호와 네임스페이스를 명시하여 요청을 하면 시스템이 그에 맞는 로그 수집기 배포 명세를 생성한 뒤 압축 파일 형태로 전송한다. 이 배포 명세를 통해 사용자 애플리케이션에 로그 수집기를 함께 배포하고 로그 수집 상태를 확인할 수 있다. 로그 수집기 정보는 애플리케이션 id, DB 인덱스, 수집기 상태로 구성된다.

비공개 데이터 업로드 요청 클래스는 사용자가 애플리케이션 배포 명세에 필요한 파일들을 시스템에 업로드하기 위해 사용하는 인터페이스이다. 사용자가 애플리케이션 id 와 파일을 업로드하면 시스템은 지원하는 포맷에 따라 파일들을 필터링하고 Elasticsearch 에 저장한 뒤 저장 기록을 로그로 저장한다. 비공개 데이터 업로드 기록은 파일 경로와 저장 상태로 구성되고 비공개 데이터 정보는 Elasticsearch 조회용 DB 인덱스로 구성된다.

- 배포 파일 관리

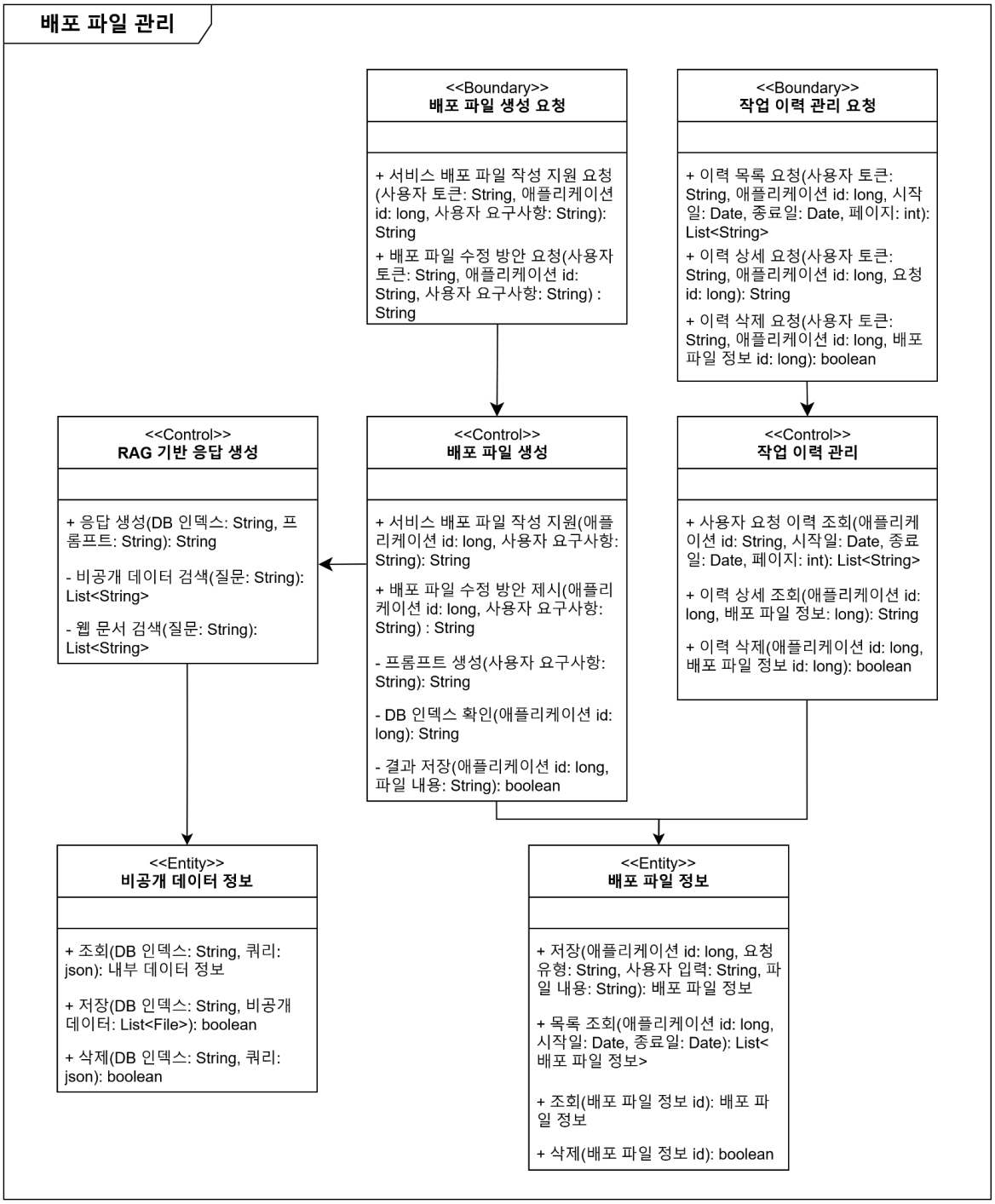

그림 4 배포 파일 관리 클래스 다이어그램

그림 4 는 배포 파일 관리 기능에 필요한 요소들을 클래스 다이어그램으로 나타낸 것이다.

배포 파일 생성 요청 클래스는 배포 파일 작성 지원이나 수정 요청을 받고 사용자 입력을 배포 파일 생성 클래스로 전달하는 인터페이스이다. 배포 파일 생성 클래스는 사용자 입력을 기반으로 프롬프트를 생성한 뒤 RAG 기반 응답 생성 클래스로 요청하여 배포 파일 작성과 수정에 필요한 답변을 얻고, 생성된 결과를 배포 파일 정보 클래스에 저장한다. RAG 기반 응답 생성 클래스는 로그 정보, 비공개 데이터, 웹 문서 검색 결과를 참조해서 프롬프트에 대한 응답을 생성한다. 로그 정보 클래스는 벡터 DB 에서 애플리케이션 로그를 조회 및 삭제 기능을 수행한다. 비공개 데이터 정보 클래스는 벡터 DB 에서 애플리케이션 내부 데이터 조회 및 삭제 기능을 수행한다.

작업 이력 관리 요청 클래스는 사용자가 기존에 요청한 배포 작업 이력 목록 및 상세 내용을 확인하거나 삭제할 수 있도록 요청 데이터를 작업 이력 조회 클래스로 전달한다. 작업 이력 관리 클래스는 사용자가 요청한 작업 이력 데이터를 반환 혹은 삭제하는 로직을 수행한다. 배포 파일 정보 클래스는 사용자 애플리케이션에서 요청한 작업 이력을 저장, 조회 및 삭제 기능을 수행한다.

2025 전기 졸업과제

- 서비스 운영 모니터링

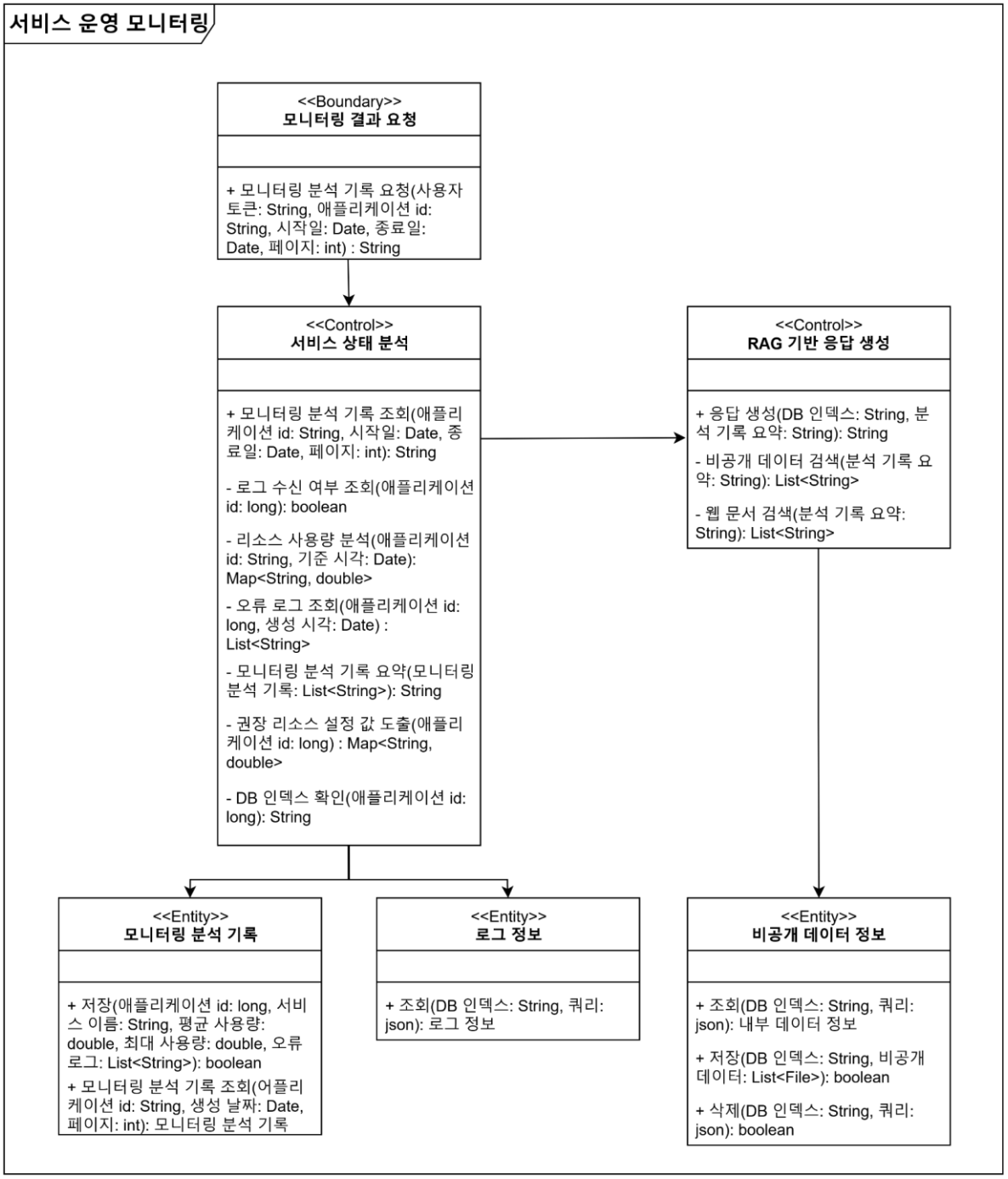

그림 5 서비스 운영 모니터링 클래스 다이어그램

그림 5 는 서비스 운영 모니터링에 필요한 요소들을 클래스 다이어그램으로 나타낸 것이다.

모니터링 결과 요청 클래스는 사용자가 배포한 MSA 애플리케이션의 모니터링 분석 결과를 조회할 때 사용하는 인터페이스이다. 로그 정보는 DB 인덱스와 생성 날짜, 내용으로 구성된다. 서비스 상태 분석 클래스는 로그 수신이 정상적으로 이루어지는 MSA 애플리케이션들을 대상으로 애플리케이션의 컨테이너 별로 리소스 사용량 분석, 오류 로그 조회를 수행한다. 평균 리소스 사용량, 최대 리소스 사용량과 오류 로그 목록을 모니터링 분석 기록으로 저장한다.

권장 리소스 설정 값 도출에서는 컨테이너의 과거 평균 리소스 사용량을 기반으로 최소 보장 리소스 제한 값을, 최대 사용량 기반으로 최대 사용 리소스 제한 값을 도출한다. 오류 로그를 발견하거나 권장 리소스 설정 값과 현재 리소스 제한 값이 30% 이상 차이날 경우, 배포 파일 동적 수정을 위해 모니터링 분석 기록이 반영된 프롬프트를 생성한다. RAG 기반 응답 생성 클래스에서 생성된 프롬프트를 기반으로 비공개 데이터 검색, 웹 문서 검색을 수행한 후 개선안을 도출해서 사용자에게 제공한다.

② UML 시퀀스 다이어그램

- 사용자 정보 생성

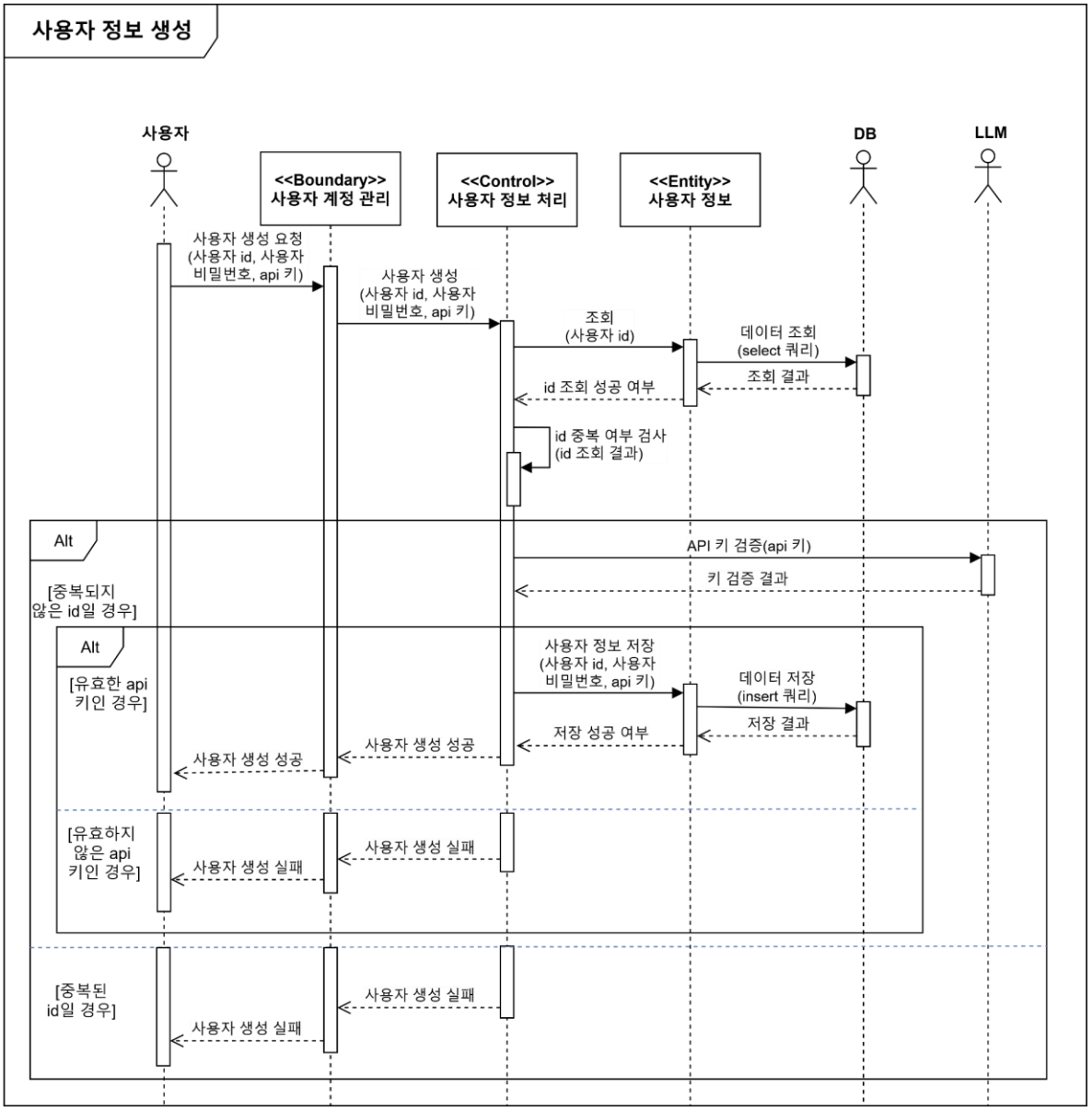

그림 6 사용자 정보 생성 시퀀스 다이어그램

그림 6 은 사용자가 시스템에 회원가입하는 시퀀스 다이어그램이다. 사용자는 id 와 비밀번호, API 키를 입력하고 회원가입을 요청한다. 시스템은 DB 를 조회하여 id 가 중복되는 id 인지 검사한다. 중복되지 않은 id 일 경우 API 키의 유효성을 검사한다. 유효한 API 키인 경우 사용자 정보를 DB 에 저장하고 회원가입 성공 메시지를 출력한다. Id 가 중복되었거나 API 키가 유효하지 않을 경우 회원가입 실패 메시지를 출력한다.

- 사용자 인증

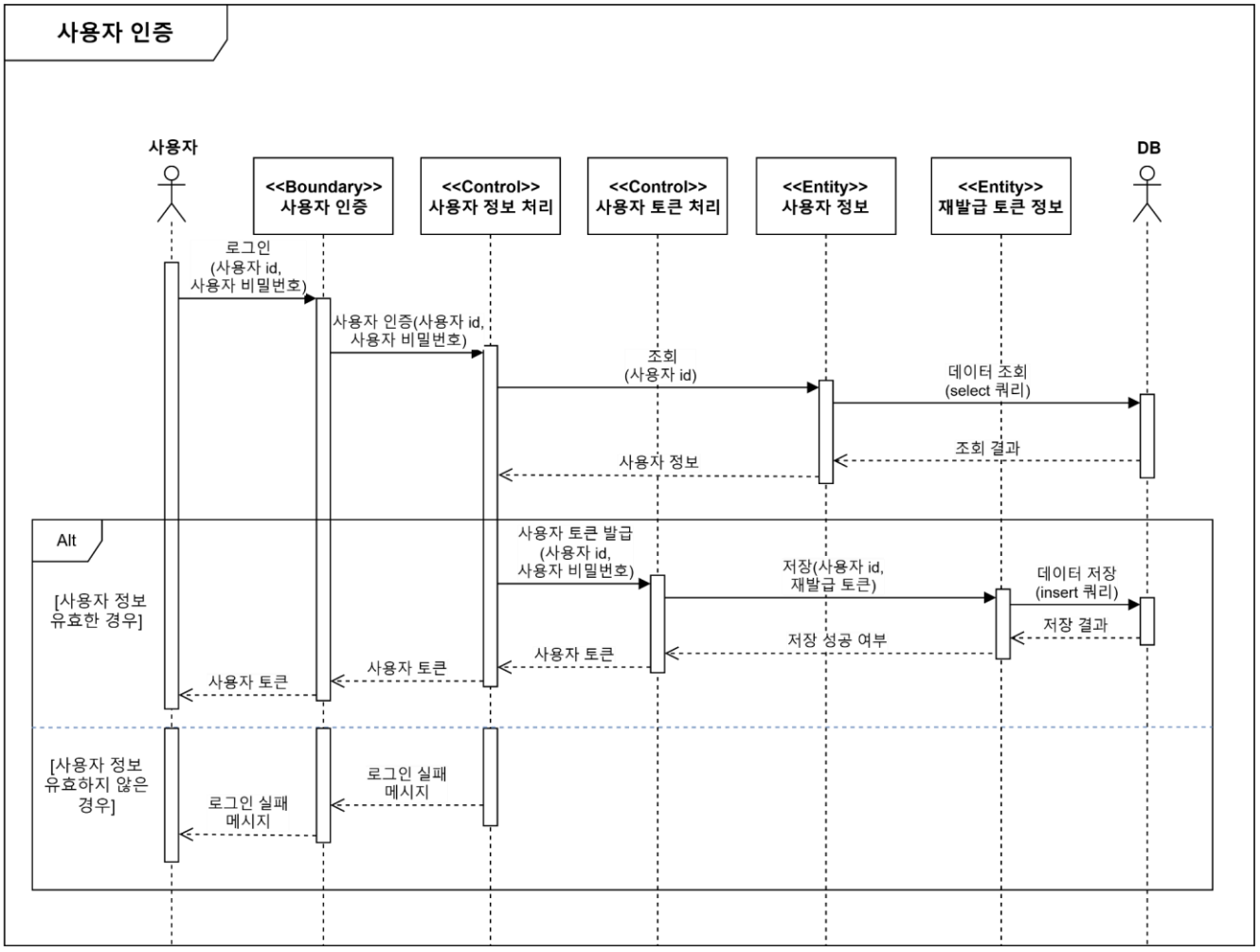

그림 7 사용자 인증 시퀀스 다이어그램

그림 7 은 사용자 인증 시퀀스 다이어그램이다. 사용자는 id 와 비밀번호를 입력하고 로그인 요청한다. 시스템은 DB 를 조회해서 유효한 사용자인지 판단한다. 유효한 사용자인 경우 사용자 토큰 처리에서 사용자 토큰을 발급한다. 이 때 재발급 토큰도 함께 발급하는데, 재발급 토큰은 DB 에 저장하고 사용자 토큰은 사용자에게 전달한다. 사용자 정보가 유효하지 않은 경우 로그인 실패 메시지를 출력한다.

- 사용자 계정 수정

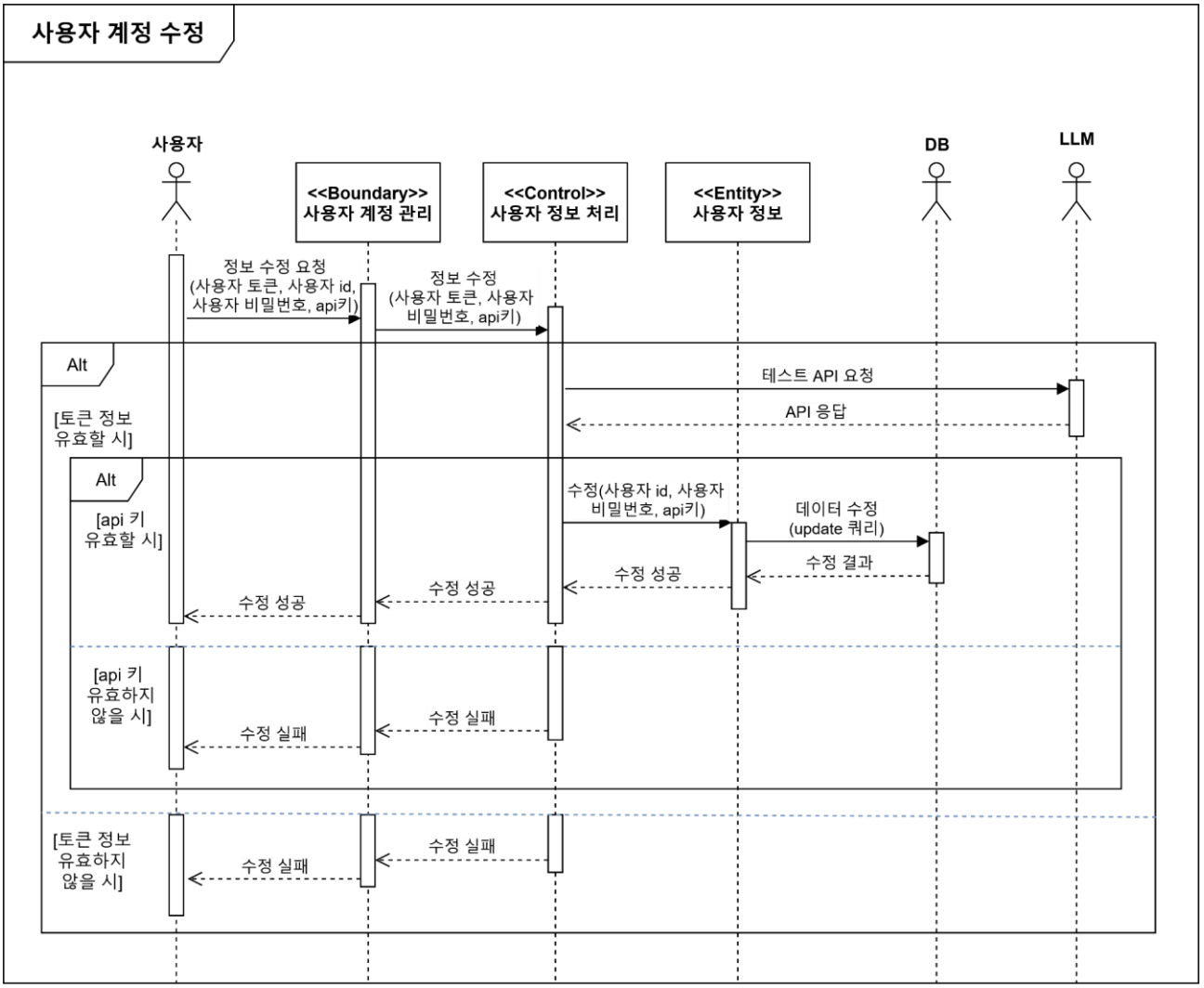

그림 8 사용자 계정 수정 시퀀스 다이어그램

그림 8 은 사용자 계정 수정 시퀀스 다이어그램이다. 사용자는 사용자 토큰과 변경하고자 하는 내용의 사용자 id, 사용자 비밀번호, api 키를 입력하고 정보 수정을 요청한다. 시스템은 토큰 정보가 유효할 시 API 키 검증을 수행한다. API 가 유효하다면 수정할 정보를 DB 에 저장하고 성공 응답을 반환한다. API 가 유효하지 않을 경우 실패 응답을 반환한다. 토큰 정보가 유효하지 않은 경우에도 실패 응답을 반환한다.

2025 전기 졸업과제

- 사용자 계정 삭제

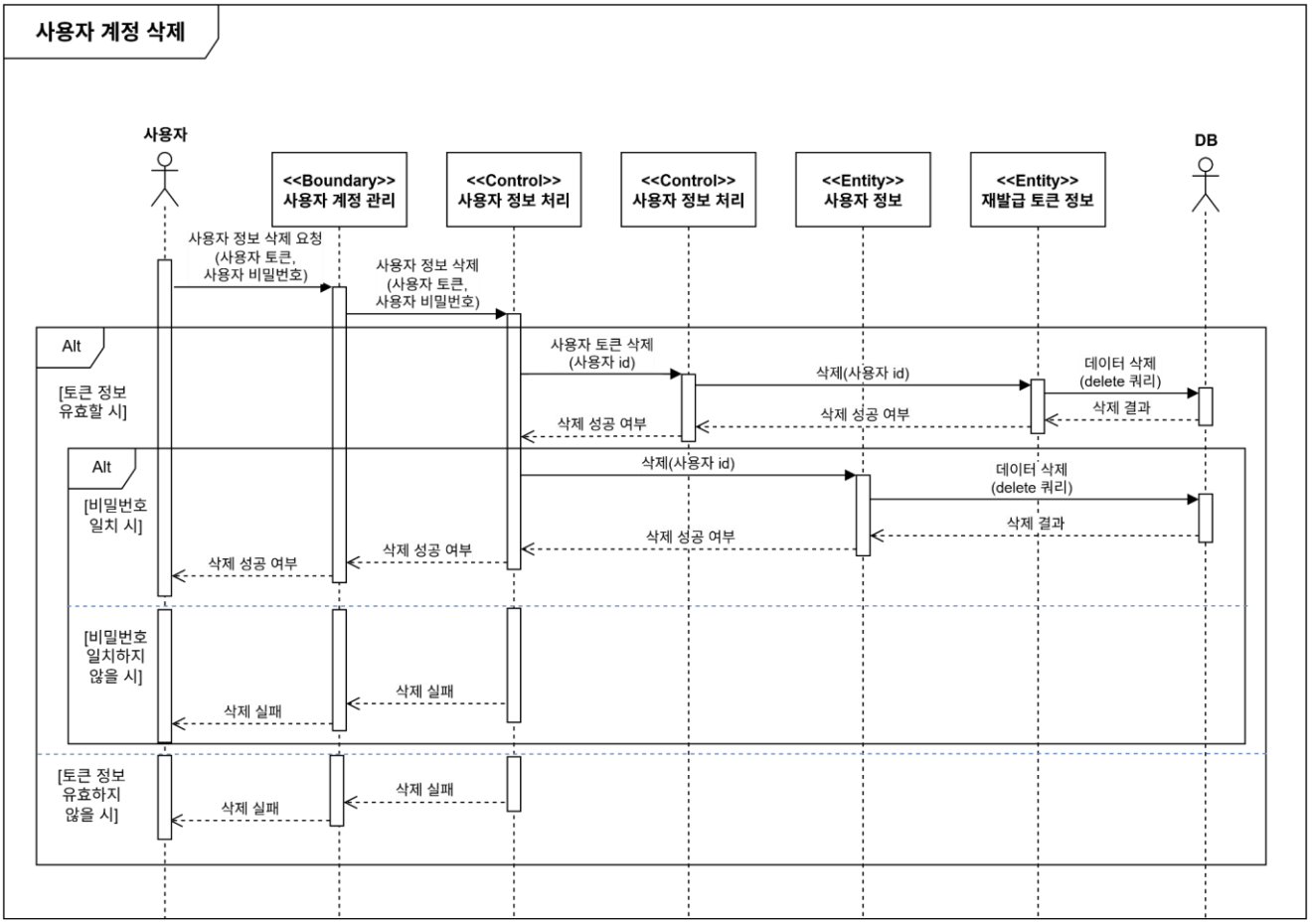

그림 9 사용자 계정 삭제 시퀀스 다이어그램

그림 9 는 사용자 계정 수정 시퀀스 다이어그램이다. 사용자는 사용자 토큰과 사용자 비밀번호를 통해 사용자 정보 삭제를 요청한다. 시스템은 토큰의 만료 여부를 판단하고 유효할 시 비밀번호 일치 여부를 확인한다. 일치할 경우 삭제 후 삭제 성공 응답을 반환한다. 비밀번호가 일치하지 않거나 토큰 정보가 유효하지 않은 경우, 삭제 실패 응답을 반환한다.

2025 전기 졸업과제

- 애플리케이션 등록

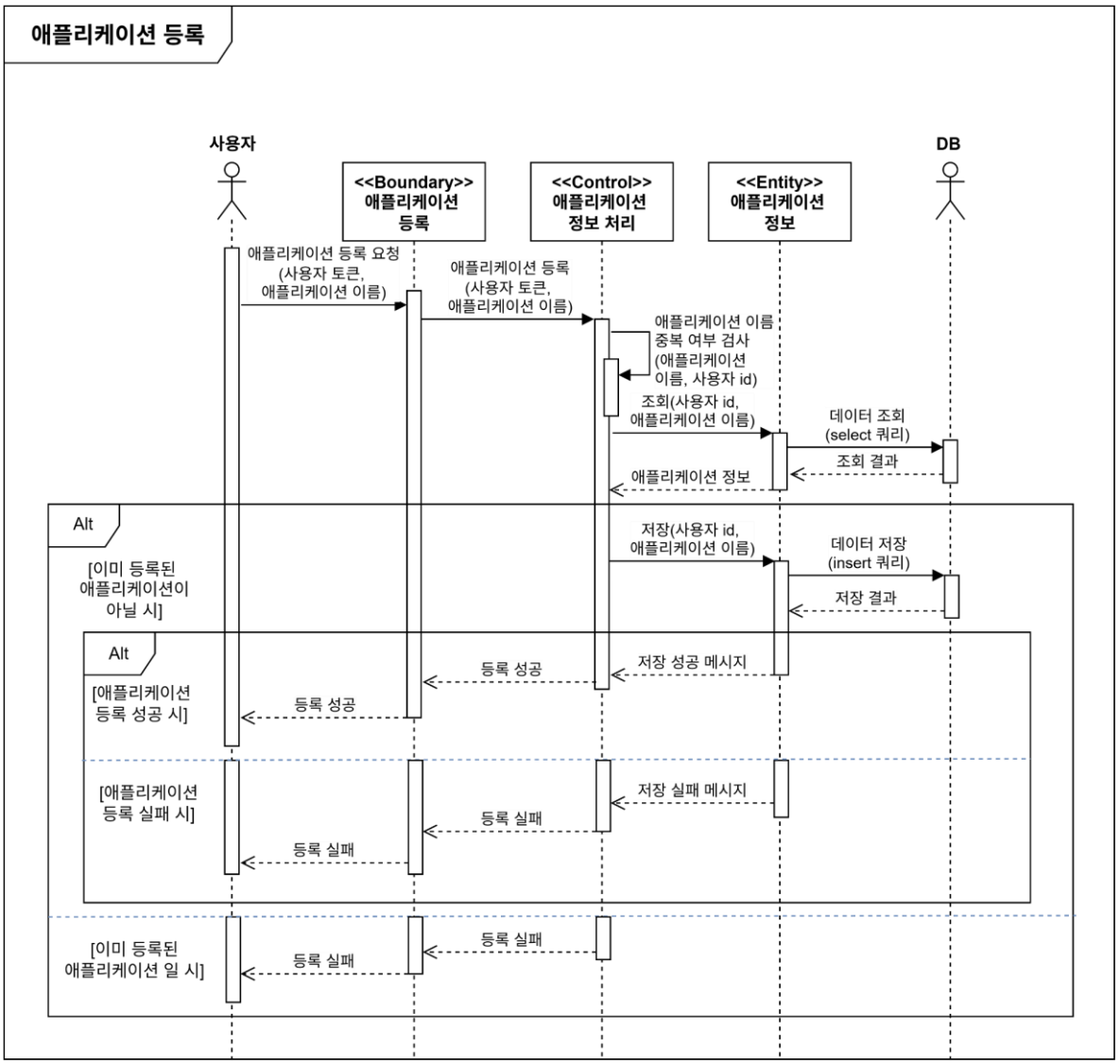

그림 10 애플리케이션 등록 시퀀스 다이어그램

그림 10 은 사용자 시스템에 애플리케이션을 등록하는 시퀀스 다이어그램이다. 사용자는 애플리케이션 이름을 입력하고 등록을 요청한다. 시스템은 이미 등록된 애플리케이션인지 검사한다. 등록되지 않았을 경우 시스템에 애플리케이션 정보를 저장을 시도하고 저장 성공 시 등록 성공 응답을 반환한다. 등록이 실패하거나 이미 등록되어 있을 경우 등록 실패 응답을 반환한다.

- 애플리케이션 정보 수정

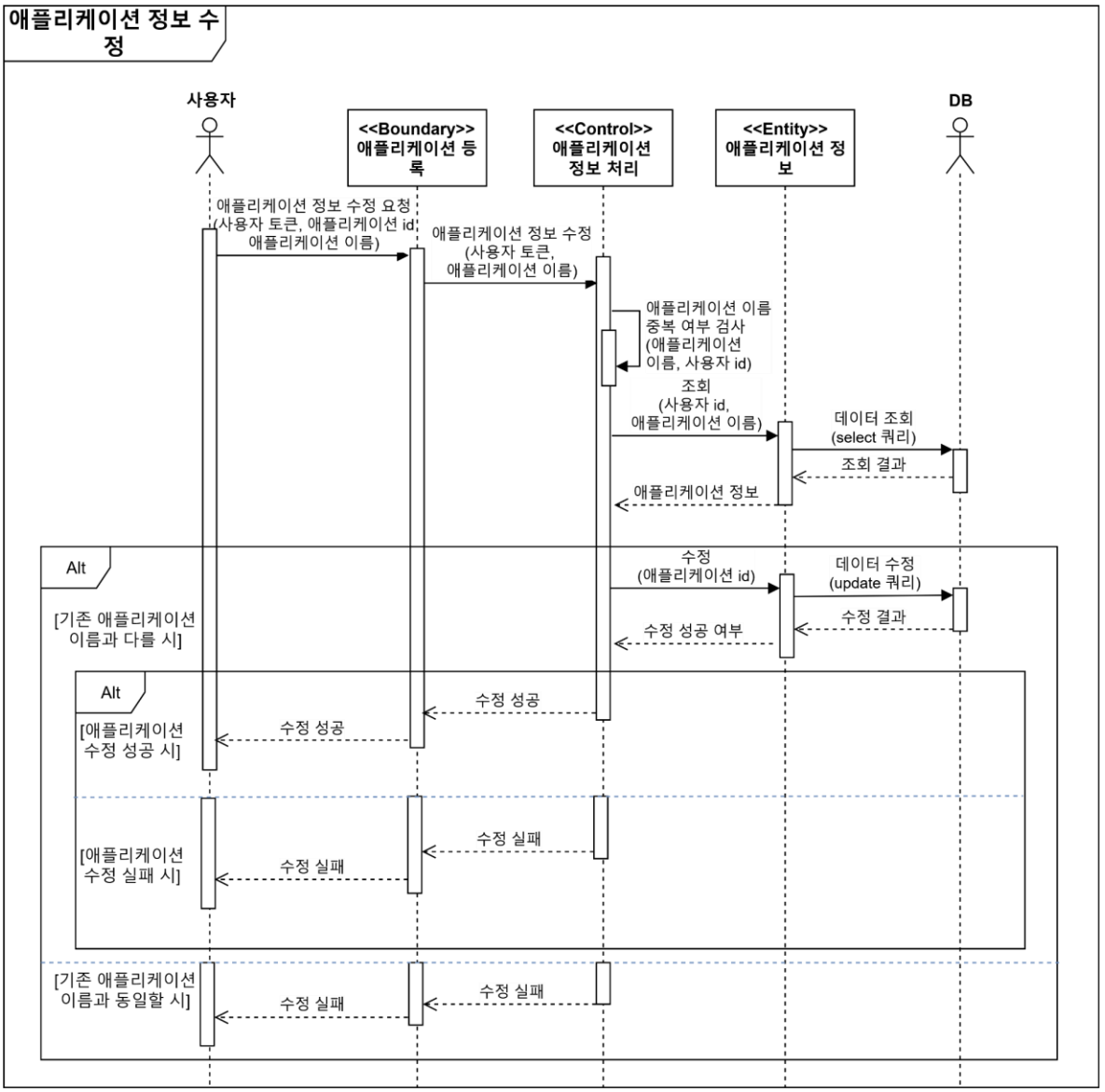

그림 11 애플리케이션 정보 수정 시퀀스 다이어그램

그림 11 은 애플리케이션의 정보를 수정하는 시퀀스 다이어그램이다. 사용자는 수정하고자 하는 애플리케이션의 이름을 입력하고 수정을 요청한다. 시스템은 기존 애플리케이션 이름과 동일한지 검사한다. 동일하지 않을 경우 애플리케이션 수정을 시도하고 성공 시 수정 성공 응답을 반환한다. 수정이 실패하거나 이름이 동일한 경우 수정 실패 응답을 반환한다.

- SSH 키 등록 요청

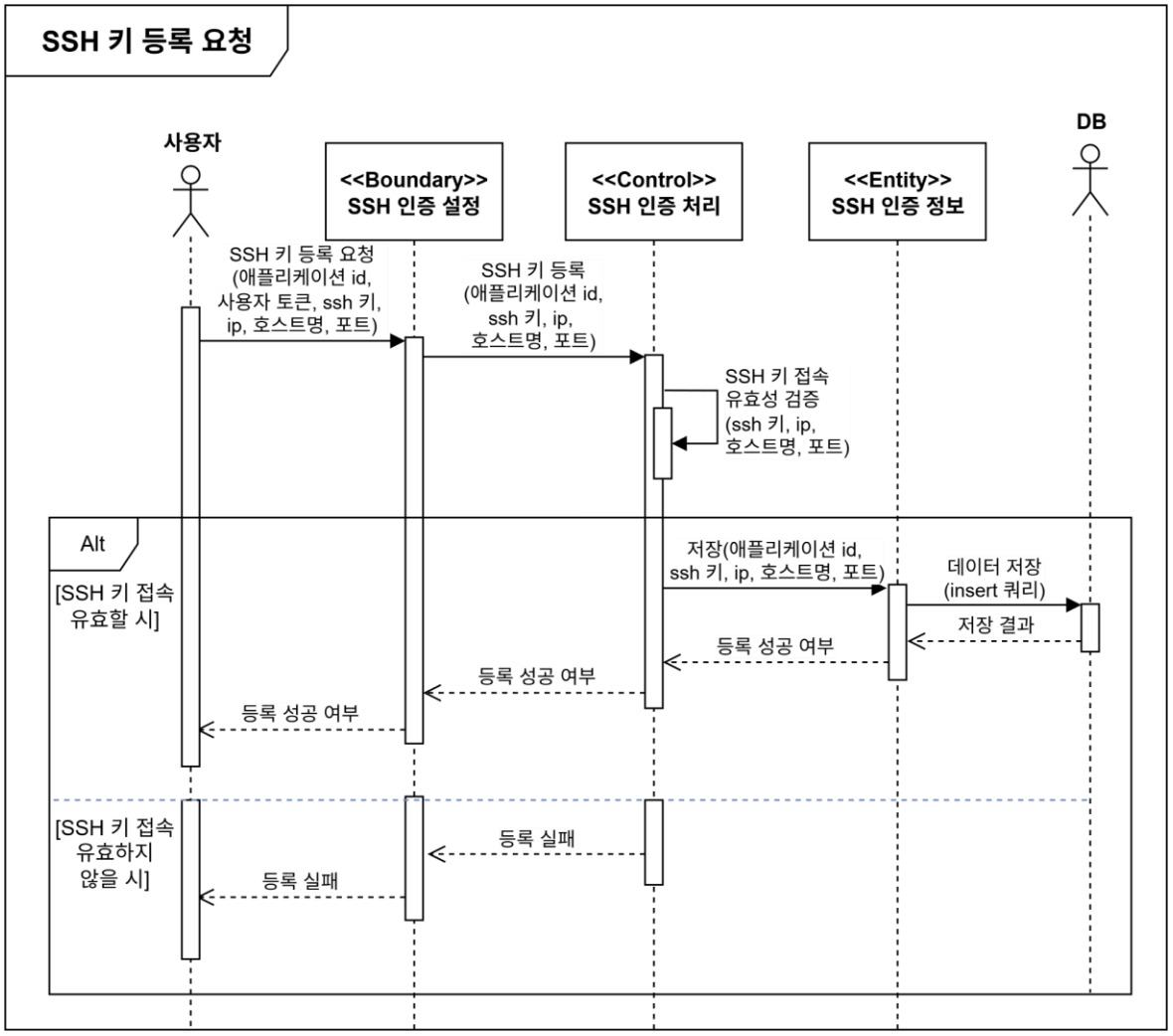

그림 12 SSH 키 등록 요청 시퀀스 다이어그램

그림 12 애플리케이션의 SSH 키의 등록을 요청하는 시퀀스 다이어그램이다. 사용자는 애플리케이션 이름, SSH 키와 ip, 호스트명, 포트 정보를 입력하고 등록을 요청한다. 시스템은 SSH 키가 유효한지 검사한다. 유효한 키인 경우 DB 에 SSH 인증 정보를 저장하고 등록 성공 여부를 반환한다. SSH 키가 유효하지 않을 경우 등록 실패 응답을 반환한다.

- SSH 키 수정 요청

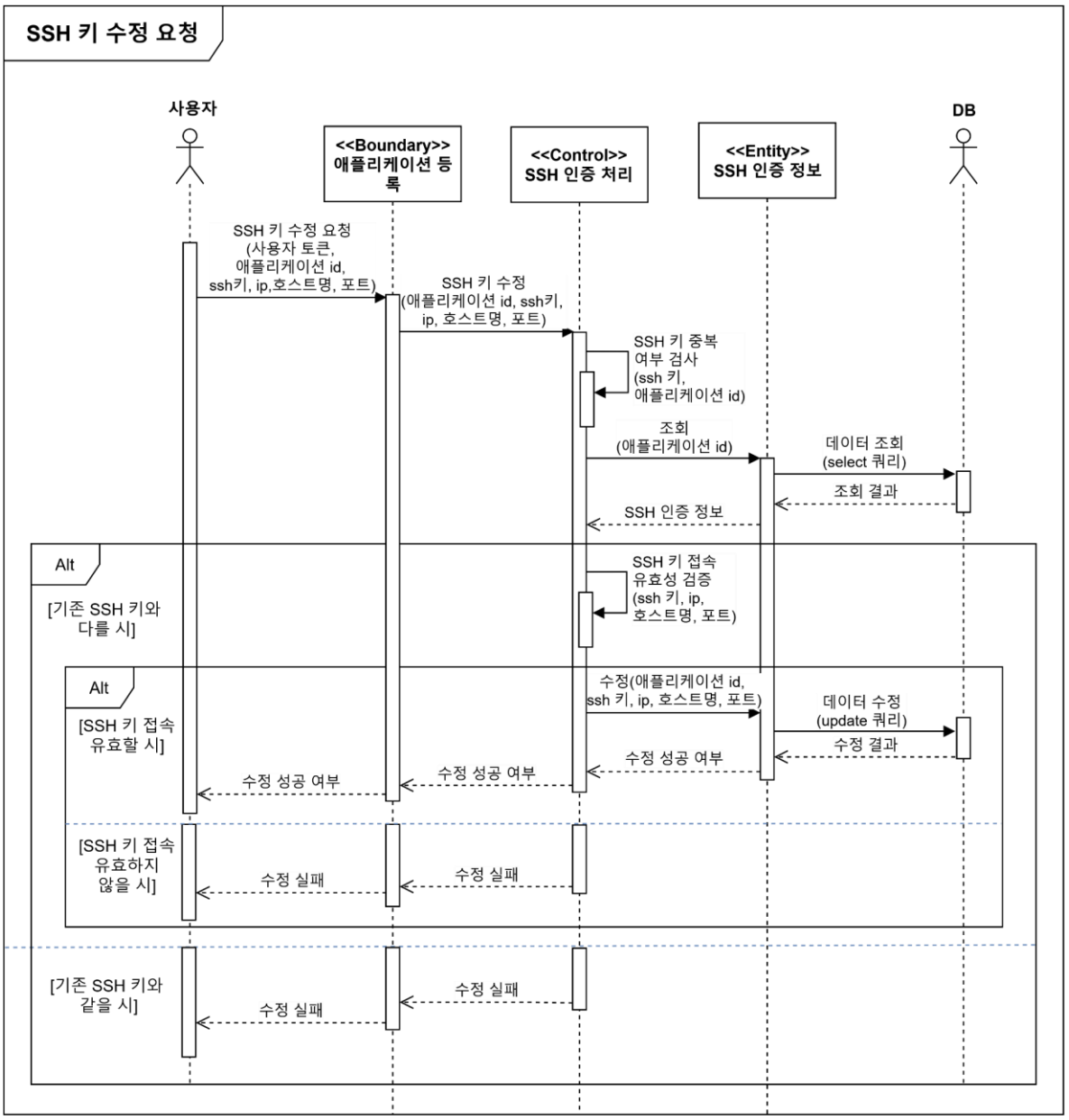

그림 13 SSH 키 수정 요청 시퀀스 다이어그램

그림 13 은 애플리케이션의 SSH 정보를 수정하는 시퀀스 다이어그램이다. 사용자는 수정하고자 하는 애플리케이션의 이름, SSH 키와 ip 를 입력하고 수정을 요청한다. 시스템은 기존 SSH 키와 동일한지 검사한다. 동일하지 않을 경우 SSH 키가 유효한지 검사한다. 유효한 키인 경우 DB 에 SSH 인증 정보를 저장하고 수정 성공 여부를 반환한다. SSH 키가 유효하지 않거나 기존 SSH 와 동일한 경우 수정 실패 응답을 반환한다.

- 로그 수집기 배포

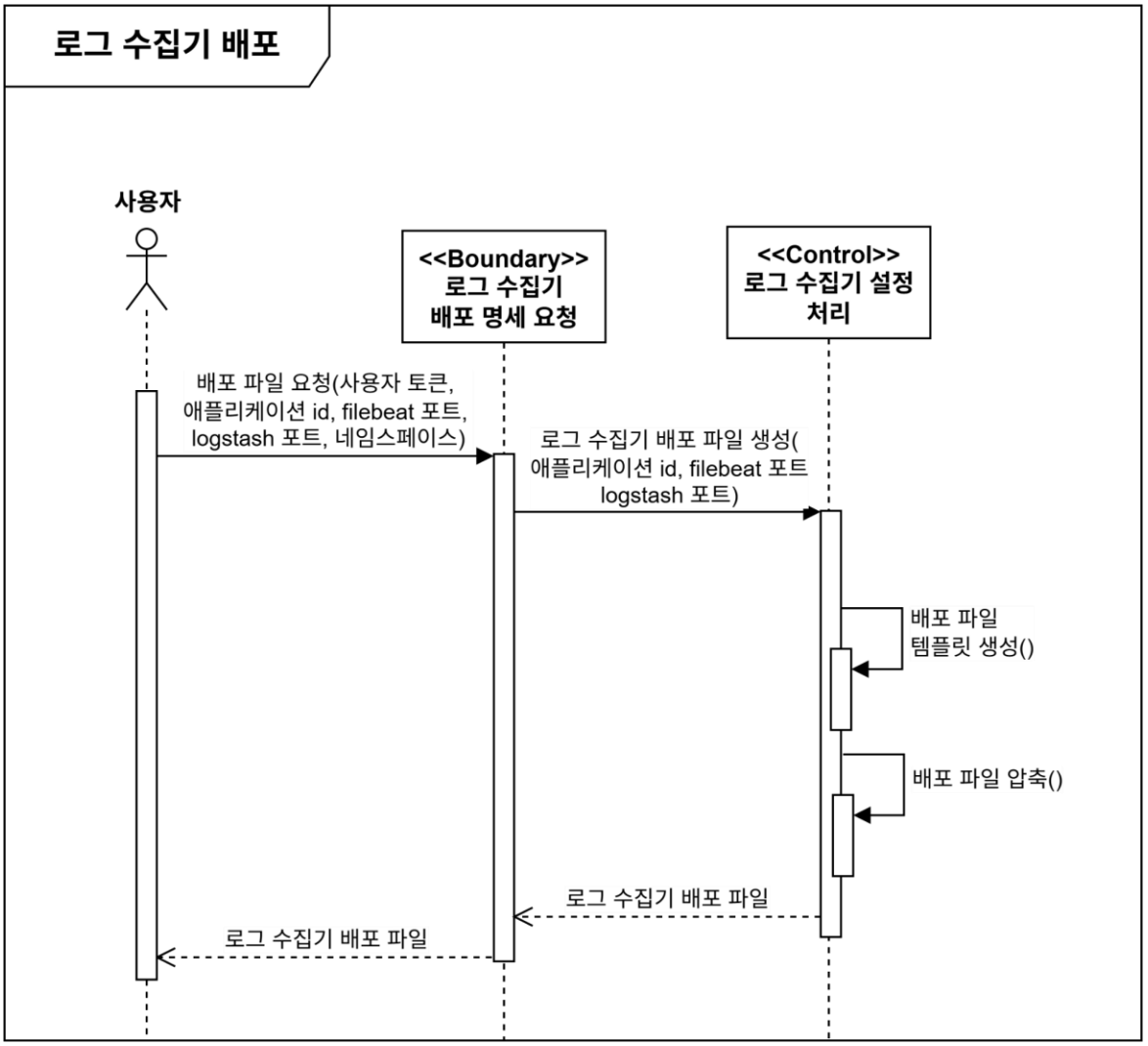

그림 14 로그 수집기 배포 시퀀스 다이어그램

그림 14 는 로그 수집기 배포 파일을 요청하는 시퀀스 다이어그램이다. 사용자는 filebeat 포트, logstash 포트를 입력하고 수정을 요청한다. 시스템은 포트 번호를 참조하여 배포 파일 템플릿을 생성하고 압축파일로 변환한다. 사용자는 압축된 파일을 다운로드한다.

- 로그 수집 상태 확인

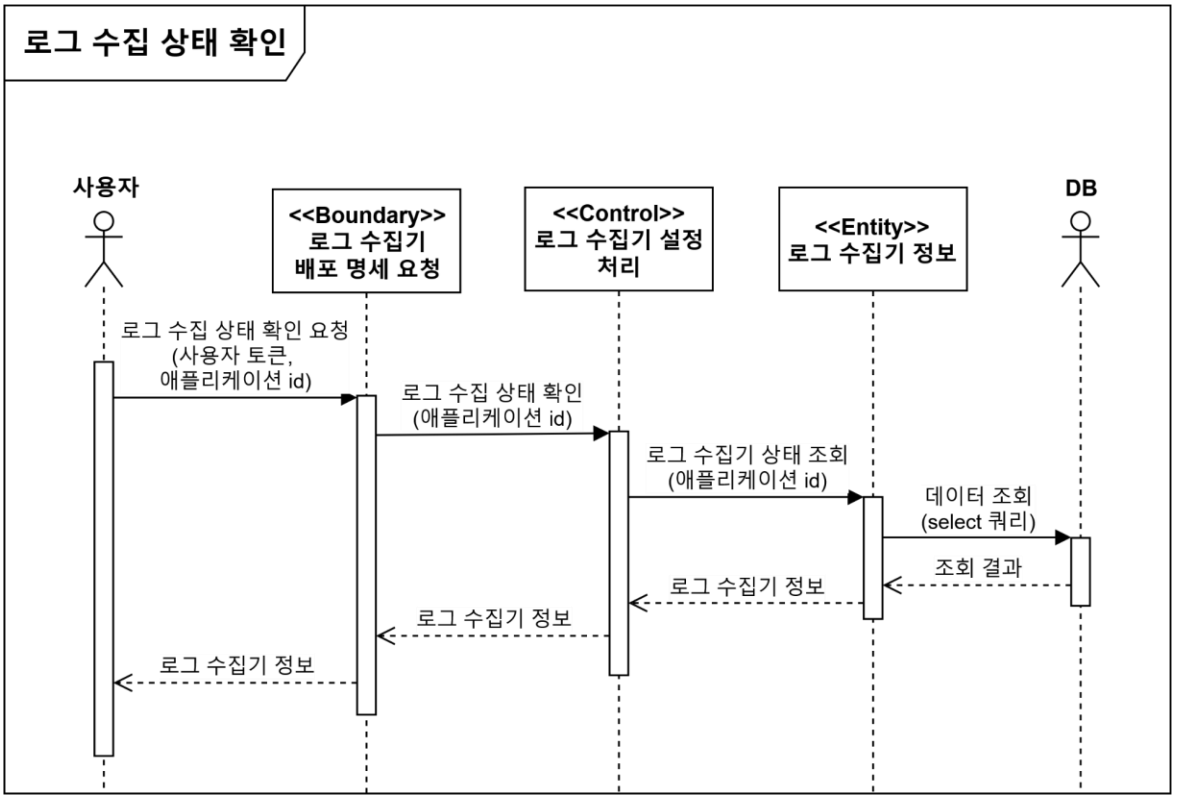

그림 15 로그 수집 상태 확인 시퀀스 다이어그램

그림 15 는 로그 수집 상태를 확인하는 시퀀스 다이어그램이다. 사용자는 운영 로그를 확인하고자 하는 애플리케이션의 이름을 입력하고 확인을 요청한다. 시스템은 DB 를 조회하여 로그 수집기 상태를 확인한다. 조회 정보가 반환되면 사용자에게 로그 수집기 정보를 출력한다.

- 비공개 데이터 업로드

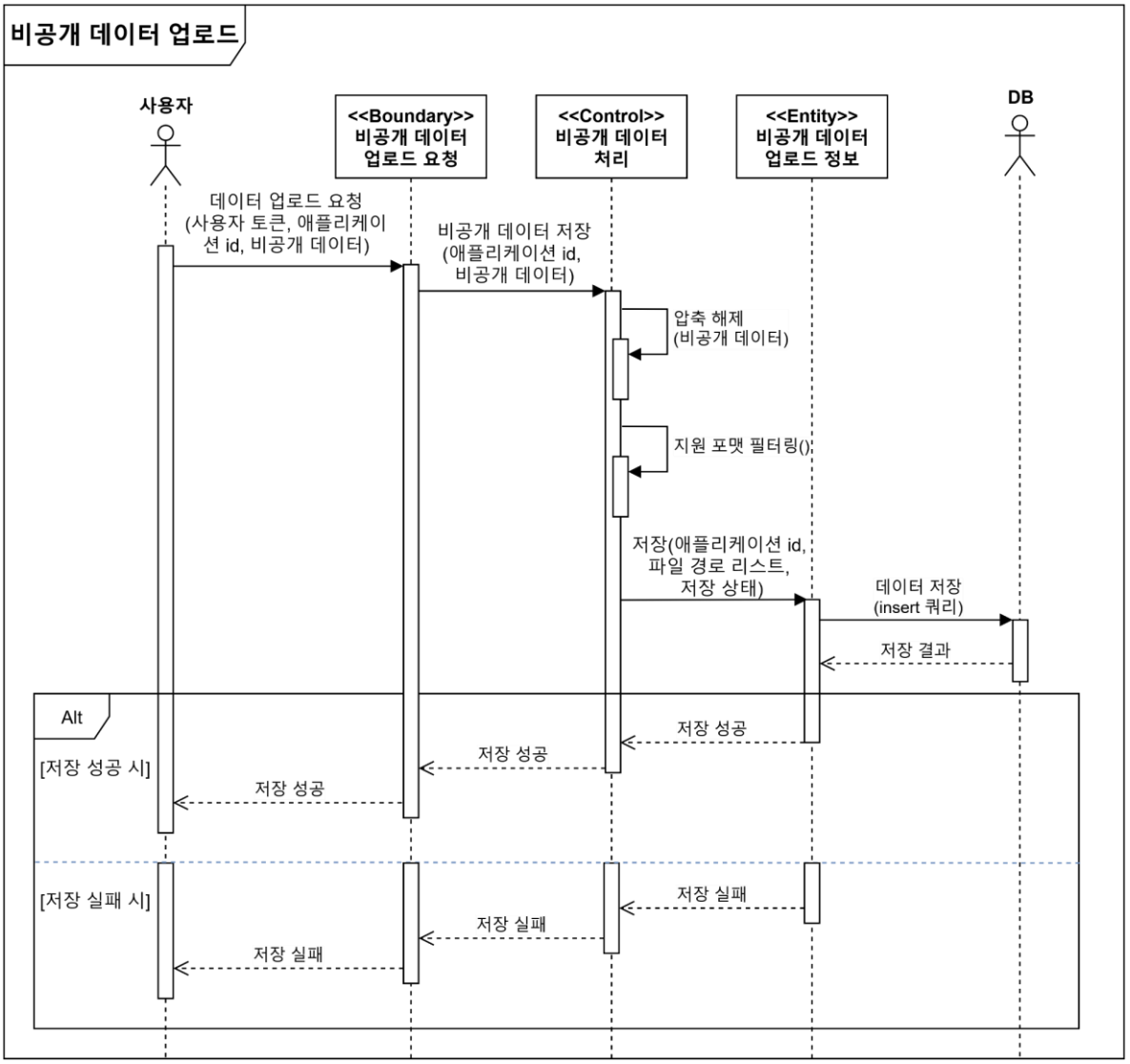

그림 16 비공개 데이터 업로드 시퀀스 다이어그램

그림 16 은 시스템에 비공개 데이터를 업로드하는 시퀀스 다이어그램이다. 사용자는 애플리케이션 id 와 비공개 파일들을 압축파일 형태로 입력하고 업로드를 요청한다. 시스템은 업로드된 파일을 압축 해제하고 지원하는 파일 형식을 필터링한 뒤 DB 에 저장을 시도한다. 저장 성공 시 저장 성공 응답을 반환한다. 저장 실패 시 저장 실패 응답을 반환한다.

- 서비스 운영 모니터링

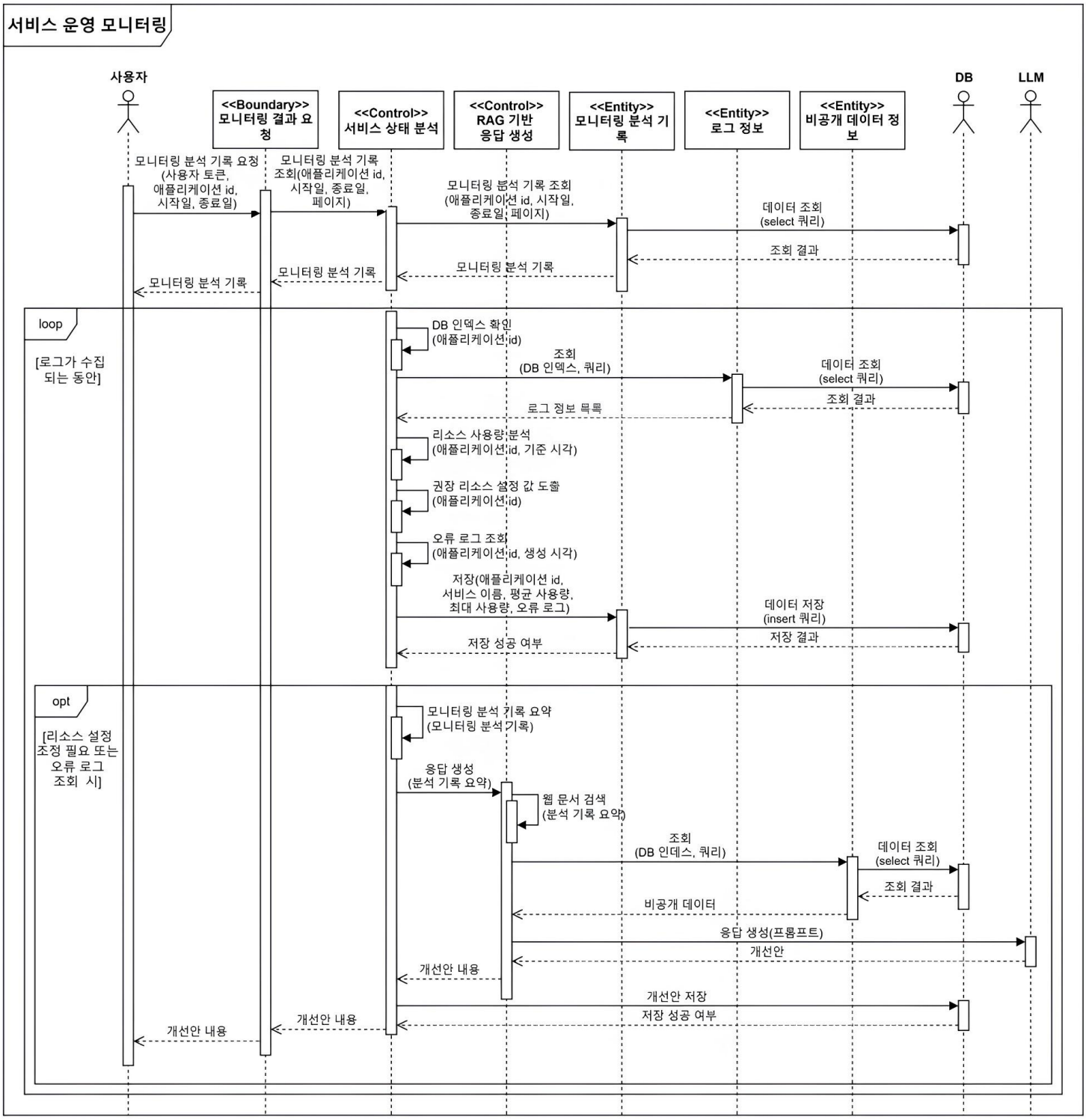

그림 17 서비스 운영 모니터링 시퀀스 다이어그램

그림 17 은 서비스 운영 모니터링을 시퀀스 다이어그램이다. 시스템은 로그가 수집되는 애플리케이션을 대상으로 주기적으로 리소스 사용량 분석을 수행한다. 또한 과거 리소스 사용량 중 평균 리소스 사용량, 최대 리소스 사용량을 기반으로 각각 최소 보장 리소스, 최대 사용 리소스 설정 값을 도출한다. 오류 로그 조회에서는 로그 레벨이 ERROR 인 로그들을 애플리케이션 별로 수집한다.

만약 현재 리소스 설정 값이 권장 리소스 설정 값에 비해 30% 이상 차이나거나, 오류 로그가 조회된 경우 이상 탐지로 간주한다. 이상 탐지 시 모니터링 분석 기록을 요약한다. 이후 리소스 권장 설정 값과 모니터링 분석 기록 요약을 포함한 분석 기록 요약을 RAG 응답 생성 요청을 위한 입력으로 사용한다. RAG 기반 응답 생성에서는 웹 문서 검색과 비공개 데이터 검색을 통해 프롬프트를 생성하고 최종적으로 LLM 에 응답 생성을 요청한다. 생성된 개선안은 DB 에 저장되고, 사용자에게 출력된다.

2025 전기 졸업과제

- 서비스 배포 파일 작성 지원

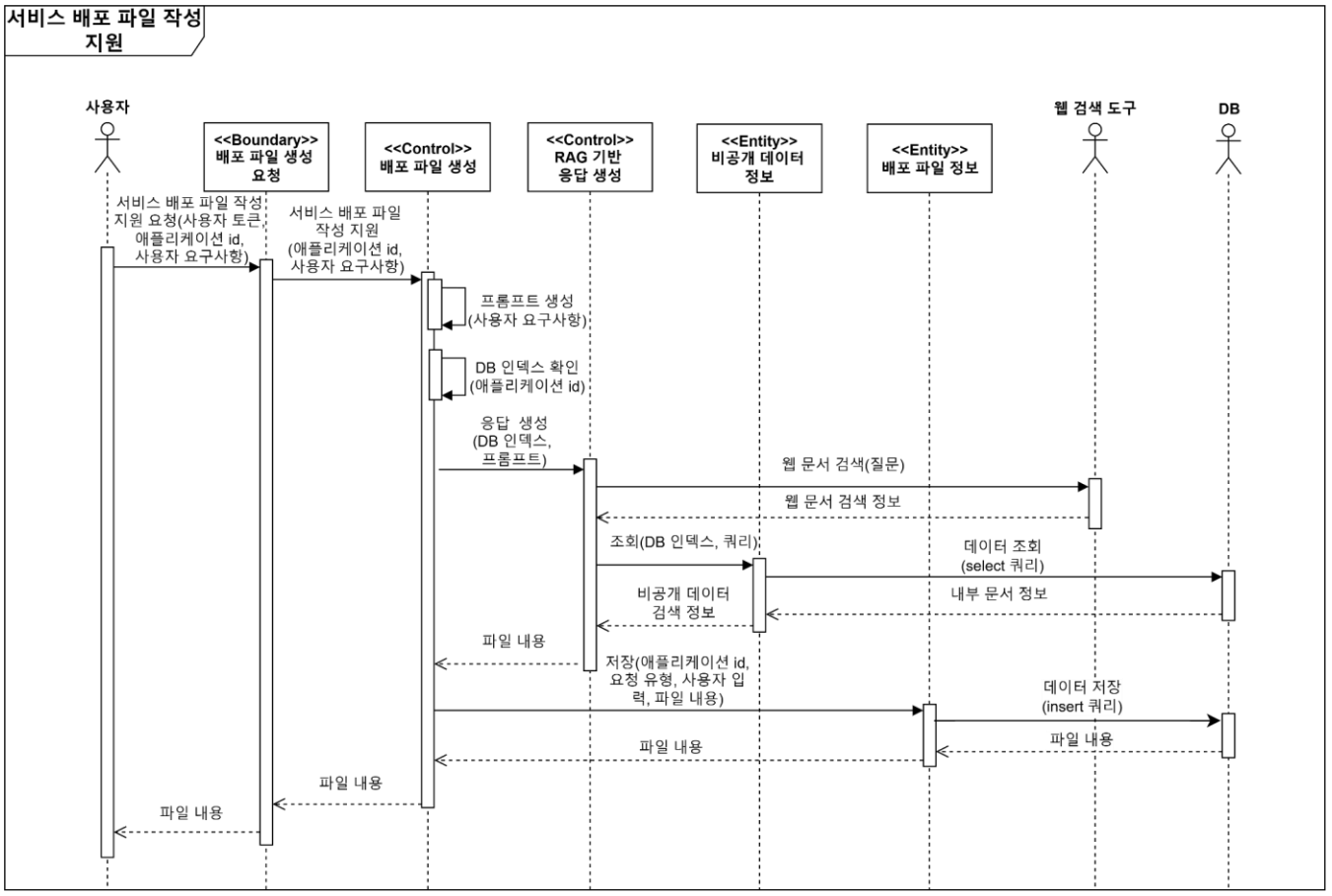

그림 18 서비스 배포 파일 작성 지원 시퀀스 다이어그램

그림 18 은 서비스 배포 파일을 생성하는 시퀀스 다이어그램이다. 사용자는 애플리케이션 id 와 요구사항을 입력하고 배포 파일 작성 지원을 요청한다. 시스템은 사용자 요구사항을 기반으로 프롬프트를 생성한다. 이후 웹 문서와 내부 문서를 검색하여 배포 파일을 생성한다. 시스템은 생성된 배포 파일을 DB 에 저장하고 사용자에게 출력한다.

- 서비스 배포 파일 수정 지원

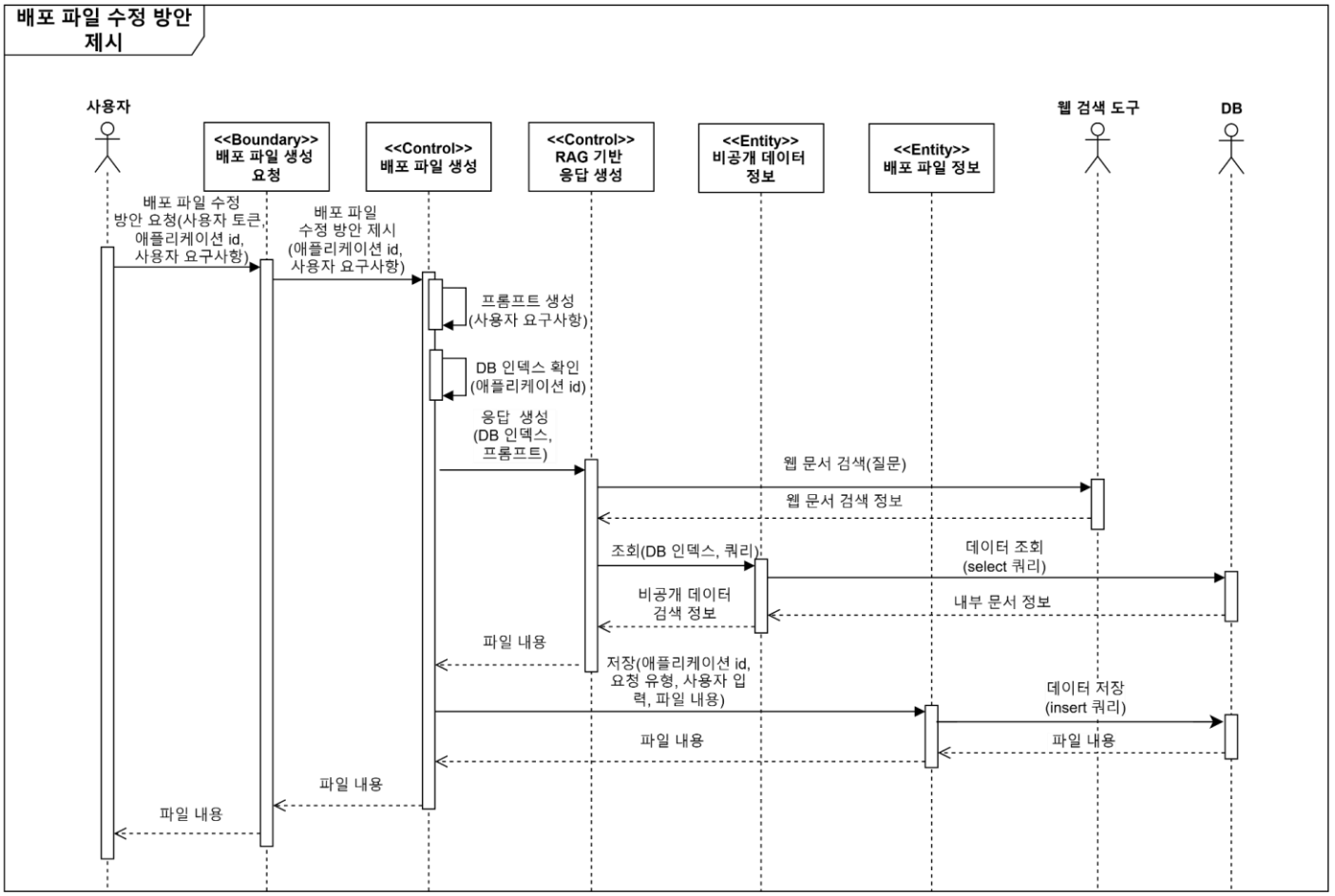

그림 19 서비스 배포 파일 수정 지원 시퀀스 다이어그램

그림 19 는 서비스 배포 파일을 수정하는 시퀀스 다이어그램이다. 사용자는 애플리케이션 id 와 요구사항을 입력하고 배포 파일 수정 지원을 요청한다. 시스템은 사용자 요구사항을 기반으로 프롬프트를 생성한다. 이후 웹 문서와 내부 문서를 검색하여 배포 파일을 생성한다. 시스템은 생성된 배포 파일을 DB 에 저장하고 사용자에게 출력한다.

- 작업 이력 관리

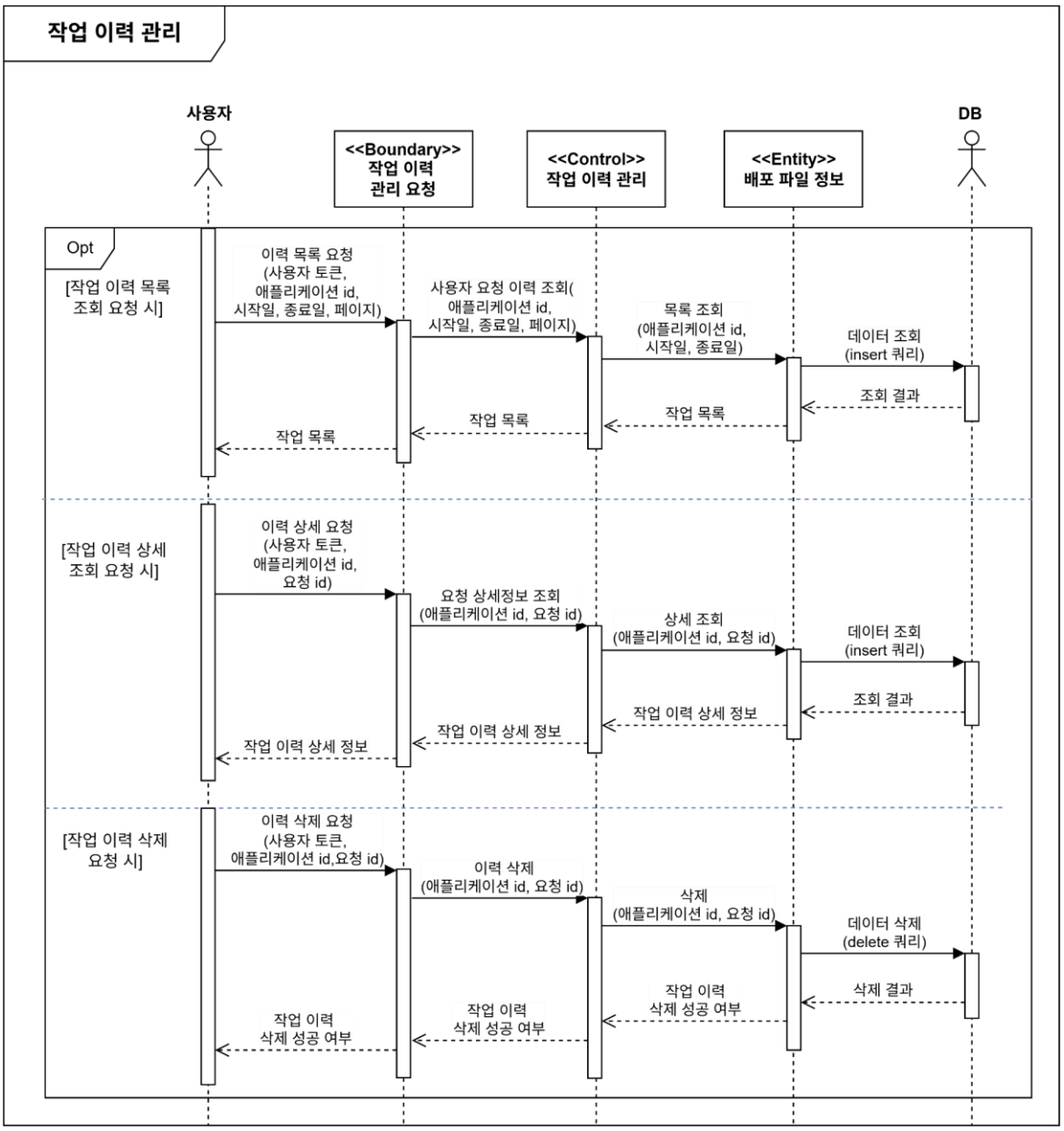

그림 20 작업 이력 관리 시퀀스 다이어그램

그림 20 은 작업 이력을 관리하는 시퀀스 다이어그램이다. 사용자가 작업 이력 목록 조회를 요청하고자 하는 경우 애플리케이션 id 와 조회하고자 하는 날짜의 범위를 설정하고 조회를 요청한다. 시스템은 날짜를 기준으로 DB 에서 배포 파일을 조회한다. DB 에서 작업 목록이 반환되면 사용자에게 작업 목록을 출력한다. 사용자가 작업 이력 상세 조회를 요청하고자 하는 경우 애플리케이션 id 와 요청 id 를 입력하고 조회를 요청한다.

시스템은 요청 id 를 기준으로 DB 에서 배포 파일을 조회한다. DB 에서 작업 이력 상세 데이터가 반환되면 사용자에게 작업 이력 상세 정보를 출력한다. 사용자가 작업 이력 삭제를 요청하고자 하는 경우 애플리케이션 id 와 요청 id 를 입력하고 삭제를 요청한다. 시스템은 요청 id 를 기준으로 DB 에서 작업이력을 삭제한다. 삭제가 완료되면 사용자에게 작업이력 삭제 성공 응답을 반환한다.

2) 시스템 구성도

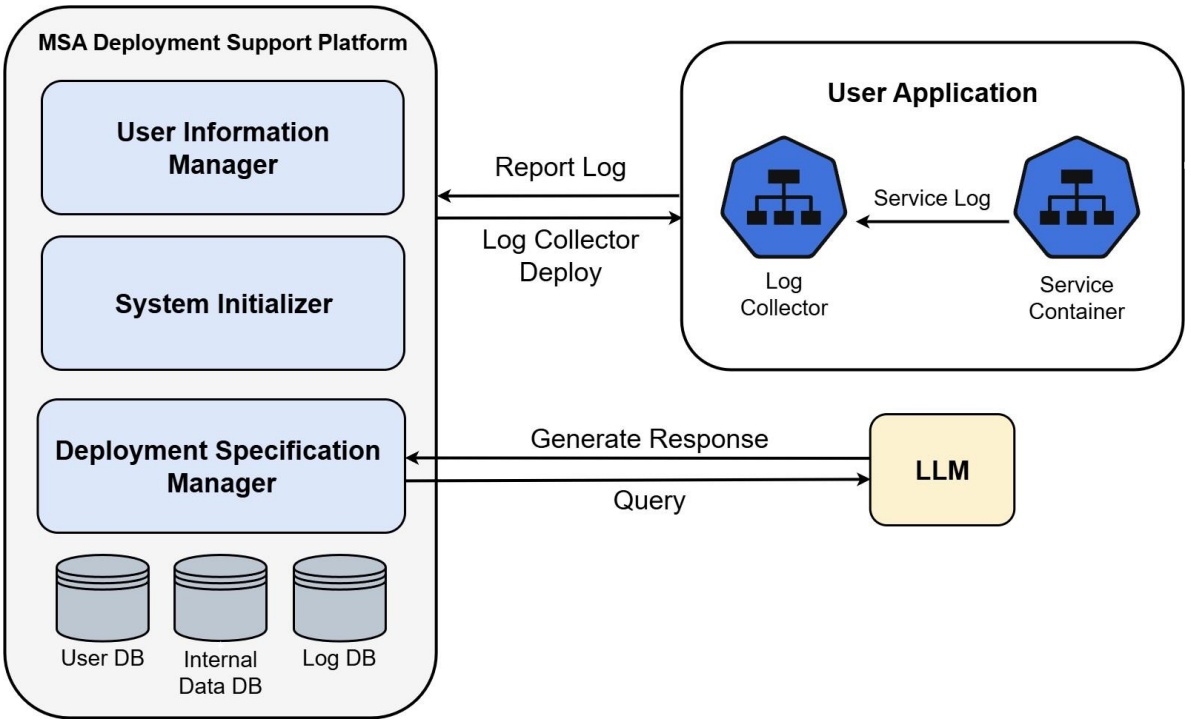

그림 21 시스템 구성도

그림 21 은 시스템 구성도를 나타낸다. 사용자는 시스템 활용을 위해 웹 기반 플랫폼에 접근한다. 시스템은 사용자 정보 관리 서버, 시스템 초기화 서버, 배포 명세 관리 서버로 구성된다. 각 서버는 데이터베이스, 사용자 애플리케이션과 외부 LLM 과 통신하며 기능을 수행한다.

① User Information Manager

사용자 정보와 애플리케이션 정보를 관리한다. 사용자가 로그인하고 시스템을 이용할 때 JWT 토큰을 기반으로 사용자 인증을 수행한다. 사용자는 토큰을 통한 인증을 받아야만 시스템의 기능을 이용할 수 있다.

② System Initializer

사용자 애플리케이션에 시스템의 배포 명세 생성을 위한 초기화를 수행한다. 사용자의 요청을 받아 로그 수집기 배포 명세를 제공한다. 또한 사용자 애플리케이션 원격 접속을 위한 SSH 인증 정보를 입력받고 원격 접속 인터페이스를 제공한다. 그리고 사용자가 업로드한 내부 데이터를 Internal Data DB 에 저장한다.

③ Deployment Specification Manger

사용자의 요청에 따라 배포 명세 파일을 생성하거나 기존 명세를 수정한다. 사용자의 질의를 기반으로 프롬프트를 생성하고 내부 데이터, 외부 문서 등을 참조하여 RAG 기반 답변을 생성한다. 또한 시스템 운영 중 이상이 탐지되었을 경우 모니터링 분석 기록을 활용하여 배포 명세 수정 방안을 제안한다. 생성된 모든 배포 명세는 저장되어 사용자가 조회가능하다.

④ User DB

사용자의 로그인 정보, API 키, 애플리케이션 정보 그리고 각 애플리케이션의 SSH 키, 모니터링 분석 기록과 배포 명세 생성 기록 정보를 저장하는 데이터베이스이다.

⑤ Internal Data DB

사용자가 업로드한 애플리케이션 배포 명세 생성에 필요한 내부 데이터를 저장하는 데이터베이스이다. 업로드된 데이터는 Elasticsearch 에 의해 벡터화된 후 저장되어 RAG 기반 답변 생성 시 유사도가 높은 데이터를 참조한다.

⑥ Log DB

시스템에 의해 사용자 애플리케이션 배포되어 수집한 로그를 저장하는 데이터베이스이다. 로그는 Elasticsearch 에 저장되고 시스템에 의해 주기적으로 분석된다. 만약 이상 동작이 감지된 경우 시스템은 배포 파일 수정을 제안한다.

⑦ User Application

사용자가 배포하고자 하는 혹은 이미 배포되어 있는 컨테이너 기반 MSA 애플리케이션이다. 사용자는 시스템에 요청하여 얻은 로그 수집기 배포 명세를 애플리케이션과 함께 배포하여 시스템이 운영 로그를 수집하도록 한다. 시스템은 전송받은 로그를 분석하여 모니터링 분석 기록을 생성한다.

⑧ LLM

RAG 기반 배포 명세 생성에 활용되는 외부 LLM 이다. 시스템은 사전에 정의된 프롬프트와 내부 데이터를 기반으로 API 를 통해 응답 생성을 요청하고 LLM 이 생성한 응답을 반환 받은 후 사용자에게 출력한다.

4. 구성원 별 개발 진척도

표 4 는 구성원 별 개발 진척도를 나타낸다.

표 4 구성원 별 개발 진척도

이름	진척도
김휘수	1. RAG 파이프라인 개발 - 토크나이저, 임베딩 모델 선정 및 테스트 - LangSmith 와 연동하여 체인 실행 흐름을 모니터링 및 디버깅 2. RAG 서버 구축 - Flask 를 사용한 API 엔드포인트 개발 - Spring 서버로부터 사용자 질의를 받아 LLM(gpt-3.5-turbo)에서 응답 생성 후 반환
신세환	1. API 개발 - Thymeleaf 기반 대시보드 레이아웃 설계 - Apache MINA SSHD 를 통한 SSH 원격 접속 인증 기능 개발 - Spring Security 기반 사용자 인증 기능 개발 - Blowfish 알고리즘 기반 비밀번호 암호화 기능 구축 2. 로그 수집기 배포 기능 개발 - Filebeat 와 Logstash 를 활용하여 애플리케이션의 로그를 Elasticsearch 로 실시간 전송하기 위한 배포 파일 작성 3. 테스트용 MSA 애플리케이션 개발 - 서비스 간 약결합이 적용된 테스트 애플리케이션 개발
설종환	1. API 개발 - 메인, 로그인, 회원가입 페이지 구현 - 사용자/프로젝트 관리 서비스 기능 개발 2. 비공개 데이터 업로드 - RAG 지식의 기반 데이터를 저장하기 위한 클라우드 환경 구축 및 데이터 저장소(Elasticsearch) 배포 - 클라우드 환경에 배포된 데이터 저장소 전처리 및 저장 기능 설계

5. 과제 수행 내용 및 중간 결과

1) 기본 레이아웃 및 인증 시스템

홈 화면, 로그인, 회원가입 페이지에 대한 UI 를 Thymeleaf 와 CSS 를 이용하여 구현하였다. 사용자는 아이디, 비밀번호, API 키 정보를 입력하여 회원가입을 진행한 후 로그인을 통해 홈 화면에 접근할 수 있으며, 이후 개인 프로젝트 등록을 할 수 있다.

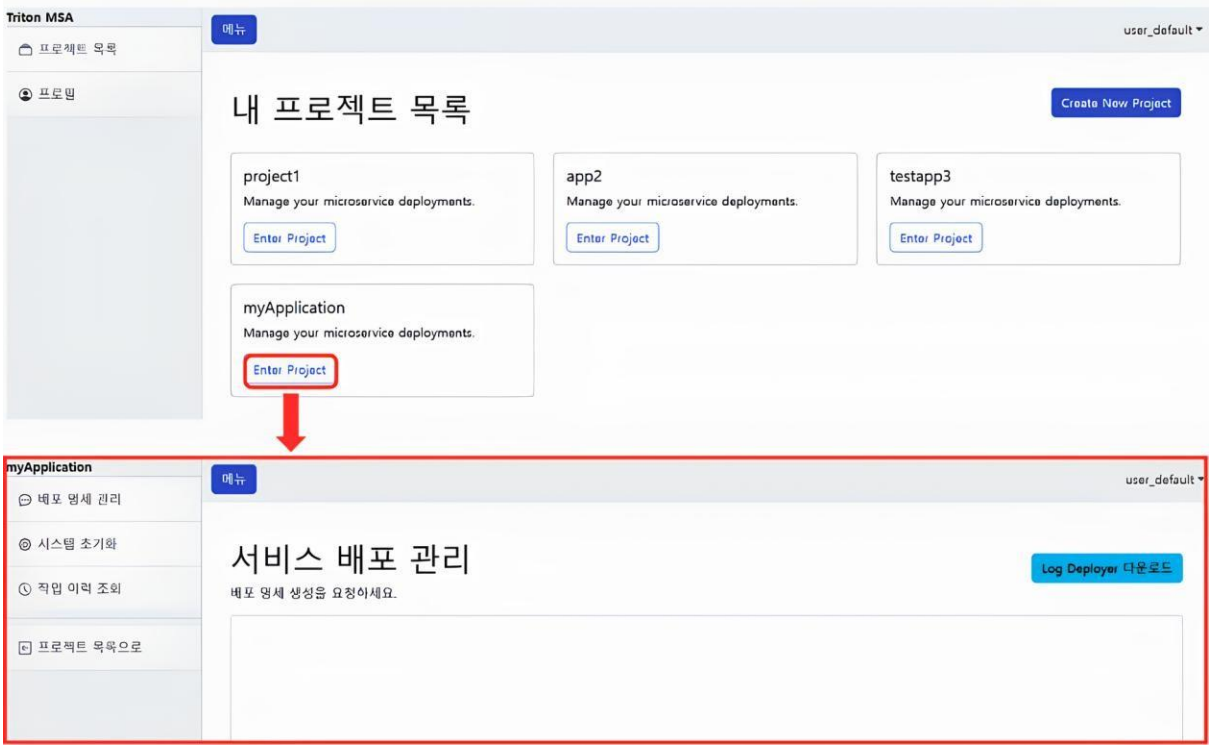

그림 22 기본 레이아웃 UI

그림 22 는 프로젝트 목록 화면과 프로젝트를 클릭할 시 전환되는 서비스 배포 관리 화면 등 기본 레이아웃 UI 를 나타낸다. 템플릿 구조를 레이아웃, 헤더, 사이드바로 나누어 공통 요소를 효율적으로 관리하도록 구성하였다. 또한 Spring Security 를 통해 세션 기반의 기본 인증 체계를 우선 적용하였으며, 추후 JWT 기반 토큰 인증 방식으로 변경하여 마이크로서비스 간 인증 연계도 문제가 없도록 할 계획이다.

2) 로그 수집기 배포파일 제공 기능

사용자 프로젝트에서 발생하는 로그를 저장하고 관리하기 위해서, 사용자는 로그 수집기 배포파일을 제공받을 수 있어야 한다. Filebeat, Logstash 의 수집기 템플릿들을 YAML 형식으로 구성하였으며, 사용자는 대시보드에서 배포파일을 다운로드 받을 수 있다. 그림 23 은 로그 수집기 배포파일 zip 파일을 생성하는 코드를 나타낸다.

```
public byte[] generateDeploymentZip(Long projectId) throws IOException {
    try(ByteArrayOutputStream baos = new ByteArrayOutputStream();
    ZipOutputStream zos = new ZipOutputStream(baos)) {
        addToZipFromResource(zos, "01-namespace.yaml", "log_templates/namespace.yaml");
        addToZipFromResource(zos, "02-filebeat-config.yaml", "log_templates/filebeat-config.yaml");
        addToZipFromResource(zos, "03-filebeat.yaml", "log_templates/filebeat.yaml");
        addToZipFromResource(zos, "05-logstash.yaml", "log_templates/logstash.yaml");

        String logstashConfigContent = generateLogstashConfig(projectId);
        addToZipFromString(zos, "04-logstash-config.yaml", logstashConfigContent);

        zos.finish();
        return baos.toByteArray();
    }
}
```

그림 23 로그 수집기 배포파일 zip 파일 생성 코드

사용자 애플리케이션 로그를 Filebeat 가 수집하여 Logstash 에 전달하고, Logstash 가 최종적으로 Elasticsearch 에 저장할 때, 각 프로젝트 별로 별도의 인덱스에 저장되어 개별 관리될 필요가 있다. 따라서 배포파일을 다운로드 할 때 사용자 애플리케이션의 id 를 참조해서 받아오고, 이를 Elasticsearch 에 저장할 인덱스로 지정해주는 것으로 구현하였다.

이를 통해 운영 중인 서비스의 로그를 중앙 서버인 Elasticsearch 에 수집하고 분석할 수 있는 기반을 마련하였다. 추후, 자신의 환경에 맞게 일부 설정(예: 포트 변경 등)을 수정한 후 배포할 수 있도록 구현할 예정이다.

3) 사용자 프로젝트 관리 기능 개발

사용자는 여러 개의 프로젝트(애플리케이션)를 등록하여 관리할 수 있다. 각 프로젝트에는 SSH 키, 비공개 데이터, 사용자별 채팅 내역 등 배포와 운영에 필요한 핵심 정보가 함께 저장된다. 그림 24 는 프로젝트 엔티티가 어떻게 구현되어 있는지를 나타낸다.

```
public class Project {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private long Id;

    @Column(nullable = false)
    private String name;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "user_id")
    private User user;

    @Embedded
    private SshInfo sshInfo;

    @OneToMany(mappedBy = "project", cascade = CascadeType.ALL, orphanRemoval = true)
    private List<ChatHistory> chatHistoryList = new ArrayList<>();

    @OneToMany(mappedBy = "project", cascade = CascadeType.ALL, orphanRemoval = true)
    private List<PrivateData> privateData = new ArrayList<>();
}
```

그림 24 프로젝트 엔티티 코드

이를 위해 사용자와 프로젝트 간에는 1:N 관계로 설계하였고, 각 프로젝트는 고유 ID 를 기준으로 로그 수집기, 배포 파일 관리 지원 등과 연동될 수 있다. 추후에 프로젝트 단위로 모니터링 분석 기록과 배포 이력도 함께 관리할 수 있도록 확장해나갈 계획이다.

4) SSH 키 유효성 검사 기능 개발

사용자가 입력한 SSH 접속 정보를 검증하는 기능을 Apache MINA SSHD 를 사용하여 개발하였다. 그림 25 는 SSH 키 유효성을 검사하는 로직이 어떻게 구현되었는지를 나타낸다.

```
try (ClientSession session = client.connect(sshInfo.getHostname(), sshInfo.getSshIpAddress(), sshInfo.getPort())
        .verify(CONNECT_TIMEOUT).getSession()) {
    session.addPublicKeyIdentity(keyPair);

    session.auth().verify(AUTH_TIMEOUT);

    boolean isAuthenticated = session.isAuthenticated();
    Files.delete(tempKeyFile);
    return isAuthenticated;
}
```

그림 25 SSH 키 유효성 검사 코드

호스트명, IP 주소, 포트 번호를 조회하여 session 획득 가능여부를 체크한다. session 획득이 가능하다면 verify 함수를 사용해서 인증 성공 여부를 체크하고 결과를 반환하는 구조이다.

5) 배포 파일 관리 Chat 인터페이스

사용자가 자연어로 질문을 입력하면, 시스템이 배포 명세를 작성하거나 오류 수정안을 제공할 수 있는 Chat 인터페이스 기본 구조를 우선 설계하였다. 그림 26 은 배포 파일 관리 Chat 인터페이스 UI 가 어떻게 구성되어 있는지를 나타낸다.

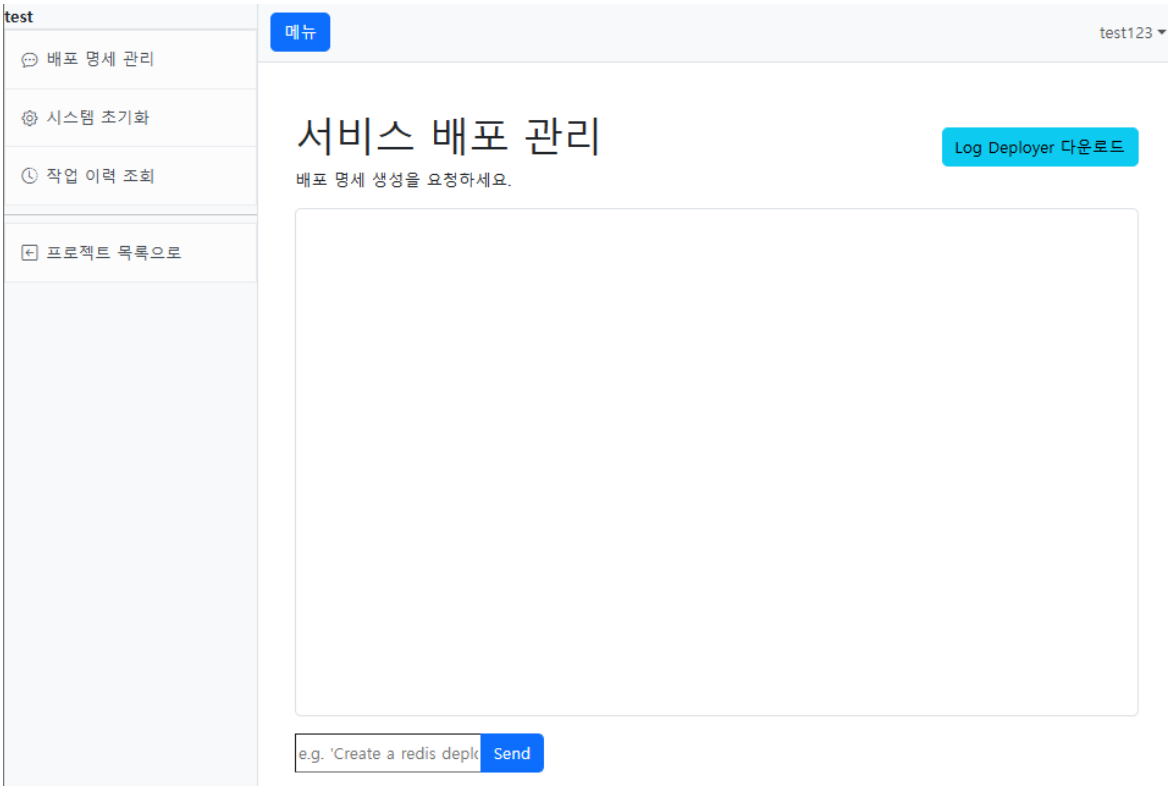

그림 26 배포 파일 관리 Chat 인터페이스 UI

사용자는 각 프로젝트별로 연결된 채팅 화면에서 배포와 관련된 질문을 입력할 수 있고, 시스템은 사용자가 입력한 질문과 비공개 데이터, 웹 문서 검색 결과를 바탕으로 권장 배포 명세 파일을 생성할 수 있는 로직을 마련해 두었다.

현재는 배포파일을 생성해주는 RAG 엔진과의 연동은 아직 되어있지 않은 상태이며, 추후 연동할 계획이다.

6) Splitter, Embedder

내부 문서를 처리하기 위한 Splitter 로써 openAI 의 tiktoken_encoder 를 사용하였다. tiktoken_encoder 는 는 OpenAI 에서 제공하는 토크나이저로, 텍스트를 GPT 계열 LLM 이 이해할 수 있는 토큰 단위로 인코딩해준다. GPT 모델과 동일한 방식으로 텍스트를 토큰화하므로 프롬프트 길이 및 요금 계산, 입력 제한 등에 대해 정확하게 대응 가능하다. 또한 LangChain 의 RecursiveCharacterSplitter 와 함께 사용하면, 토큰 수 기준으로 문서를 나누는 데 유리하여 LLM 입력을 최적화 할 수 있다. 그리고 Embedding 모델로써 openAI 의 기본값인 `text-embedding-ada-002` 모델을 사용하였다.

7) Vector Store (Chroma)

문서 검색을 위한 핵심 요소로, 사내 배포 가이드, YAML 예제, 운영 규칙 등의 문서를 분할한 후 벡터 임베딩하여 저장하는 역할을 수행한다. 임베딩은 비용이 많이 들기 때문에 최초로 파일을 임베딩한 후 Vector Store 에 저장하여 비용을 절감하고 검색 속도를 향상시켰다. 그림 27 은 문서를 로드하고 텍스트를 청크로 분할하여 임베딩한 후 벡터스토어에 저장하고 유사도 검색을 수행하는 과정을 나타낸 코드이다.

```python
from langchain.document_loaders import UnstructuredFileLoader
from langchain.text_splitter import CharacterTextSplitter
from langchain.vectorstores import Chroma
from langchain.embeddings import OpenAIEmbeddings

loader = UnstructuredFileLoader("./files/chapter_one.txt")
splitter = CharacterTextSplitter.from_tiktoken_encoder(
    separator="\n",
    chunk_size = 600,
    chunk_overlap = 100,
)

docs = loader.load_and_split(text_splitter=splitter)

embeddings = OpenAIEmbeddings()

vector_store = Chroma.from_documents(docs, embeddings)
```

그림 27 파일 로더 및 임베더

8) LangSmith

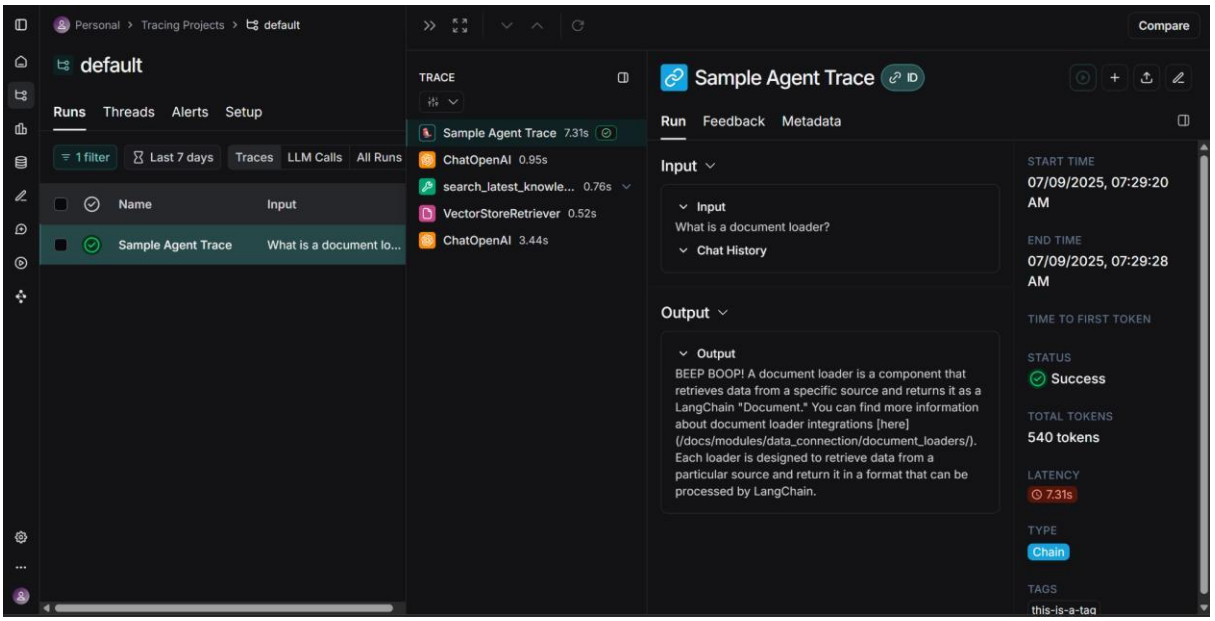

그림 28 LangSmith 대시보드 화면

그림 28 은 LangSmith 대시보드 화면을 나타낸다. LangChain 기반 시스템은 다양한 컴포넌트의 연결로 구성되기 때문에, 프롬프트 품질과 실행 결과를 체계적으로 분석할 수단이 필요하다. 따라서 LangChain 의 실험/디버깅/로그 분석 플랫폼인 LangSmith 를 사용하여 프롬프트 입출력, Retriever 검색 결과, LLM 응답 등을 시각적으로 추적하고 로그로 관리할 수 있게 하였다.

이를 통해 LLM 의 응답 정확성, 검색 문서 반영 여부, 체인별 오류 로그 등을 직관적으로 확인하고, 프롬프트의 개선 근거를 확보하여 시스템 품질 개선에 활용한다.