2025 년 전기 졸업과제 착수보고서

RAG 적용을 통한 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, 기작성된 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 MSA 구축 지원 플랫폼을 제안한다. 본 과제는 1) RAG 를 활용한 컨테이너 기반 마이크로서비스 배포 명세 생성 기능, 2) 조직 내부 데이터 중 운영 정보를 활용한 컨테이너 기반 마이크로서비스 약결합 방법, 3) 동적 운영정보 획득 및 분석을 통한 마이크로서비스 운영 환경 재구성 기술 제공을 목표로 수행한다.

2) 과제 세부 목표

① RAG 기반 마이크로서비스 배포 명세 생성 자동화

② 기능 동작 명세 기반 마이크로서비스 약결합 명세 생성 방법 도출

③ 운영 정보 기반 마이크로서비스 운영 환경 재구성 기술 제공

2. 대상 문제 및 요구사항 분석

1) 유사 시스템 분석

① k8sGPT: k8sGPT 는 Kubernetes 클러스터 내 리소스 상태를 스캔하고 문제를 진단하는 데 중점을 둔 도구다. LLM 기반으로 분석한 결과를 자연어로 설명해준다. kubectl 과 연동해서 클러스터 상태를 점검한다.

i) Kubernetes 클러스터의 문제 진단 및 설명

ii) 파드 상태, 이벤트 로그, 설정 오류 분석 후 원인 진단

② Robusta: 클러스터에서 발생하는 이벤트를 실시간으로 모니터링하여 감지된 문제에 대해 알림 채널과 통합해서 정보를 제공하고, 사용자 정의 핸들러로 문제 발생 시 자동 복구 스크립트를 실행할 수 있다.

i) 문제 발생 시 알림 전송 및 사전 정의된 대응 스크립트 실행

ii) Slack, Teams 등과 연동하여 경고 및 대응 자동화

③ Sveltos: Sveltos 는 컨테이너 배포 환경에서 리소스 상태를 지속적으로 평가하고, 사전 정의된 정책에 따라서 자동으로 수정 조치를 수행하는 도구이다.

i) 다중 클러스터 리소스 상태 자동 분석

ii) 정책 위반 감지 및 자동 수정

2) 문제점 분석

① 내부 데이터를 반영한 배포 명세 파일 수준의 생성 미지원: 기존 시스템들은 대부분 컨테이너 오케스트레이션의 API 를 통해 접근 가능한 리소스를 분석하거나 이벤트 로그를 진단하는 수준에 머물러 있다. 따라서 전문 지식이 부족한 사용자는 요구사항에 걸맞는 배포 명세를 작성하기 어렵다. 또한, 기존 시스템은 사내 커스텀 이미지, 내부 운영 정책 등 조직 내부 데이터에 대한 접근성이 떨어져 이를 반영하기 어렵다.

② 리소스 설정의 정량적 최적화 기능 부족: 기존 시스템들은 대부분 리소스 상태의 상태 진단은 가능하지만, CPU, 메모리, 요청/제한 값 등의 리소스 설정을 동적 운영 정보를 기반으로 최적화하고 개선을 제안하는 기능이 미흡하다.

③ 도메인 특화 지식 부족에 따른 진단의 비정확성: Robusta 와 Sveltos 와 같은 시스템은 알림이나 정책 기반 대응 기능은 제공하지만, 그 효과는 주로 단순 이벤트에 국한된다. 복잡한 서비스 간의 연관성, 커스텀 이미지 설정, 약결합된 마이크로서비스 구조 등 도메인에 특화 정보에 관련된 설정에 대해서는 정확한 진단이 어렵고, 표면적인 조치에 그치기 쉽다.

3) 시스템의 필요성

① 내부 데이터 기반 배포 명세 생성: 운영할 서비스에 대해 전문지식이 없는 사용자는 MSA의 약결합 구조와 동작의 논리성을 고려하여 배포 명세를 작성하기 매우 어렵다. 또한 배포 명세 작성과정에서 요구되는 컨테이너별 도메인 지식은 사용자에게 부담으로 이어질 수 있다. 따라서, 마이크로서비스 배포 명세를 생성하기 위하여 최신 내부 데이터를 실시간으로 검색하고 반영하는 방법이 필요하다.

② 운영 데이터 기반 리소스 최적화 및 서비스 안정성 확보: 배포 명세의 리소스 설정이 실제 운영 환경과 괴리가 있을 경우, 비효율적인 리소스 할당 문제를 유발한다. 따라서, 실제 운영 중 수집되는 로그 및 메트릭 데이터를 분석하여 컨테이너별 리소스 사용량에 기반한 권장 설정 값을 제안하는 방법이 필요하다.

③ 자동화된 문제 진단 및 수정안 제공: 기존 도구들은 클러스터 상태를 진단하거나 특정 이벤트 기반 알림을 제공하는 수준에 머무르며, 발생한 문제에 대해 구체적인 수정 방안은 도출하지 않는다. 따라서, 운영 중인 시스템에서 발생하는 마이크로서비스 배포 명세 오류를 탐지하고, 이를 기반으로 마이크로서비스 배포 명세의 수정 방안을 제안하는 방법이 필요하다.

4) 요구사항 분석

① 기능적 요구사항

표 1 은 시스템이 제공해야 할 기능들에 대한 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.

② 비기능적 요구사항

다음 표 2 는 시스템이 만족해야 할 비기능적 요구사항이다.

표 2. 비기능적 요구사항

요건	설명
성능	- 시스템은 로그 이상 탐지 주기를 최대 5 분 이내로 유지해야 함.
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.

5) 유스케이스 분석

① 유스케이스 다이어그램

그림 1 은 MSA 구축 지원 시스템의 유스케이스 다이어그램이다.

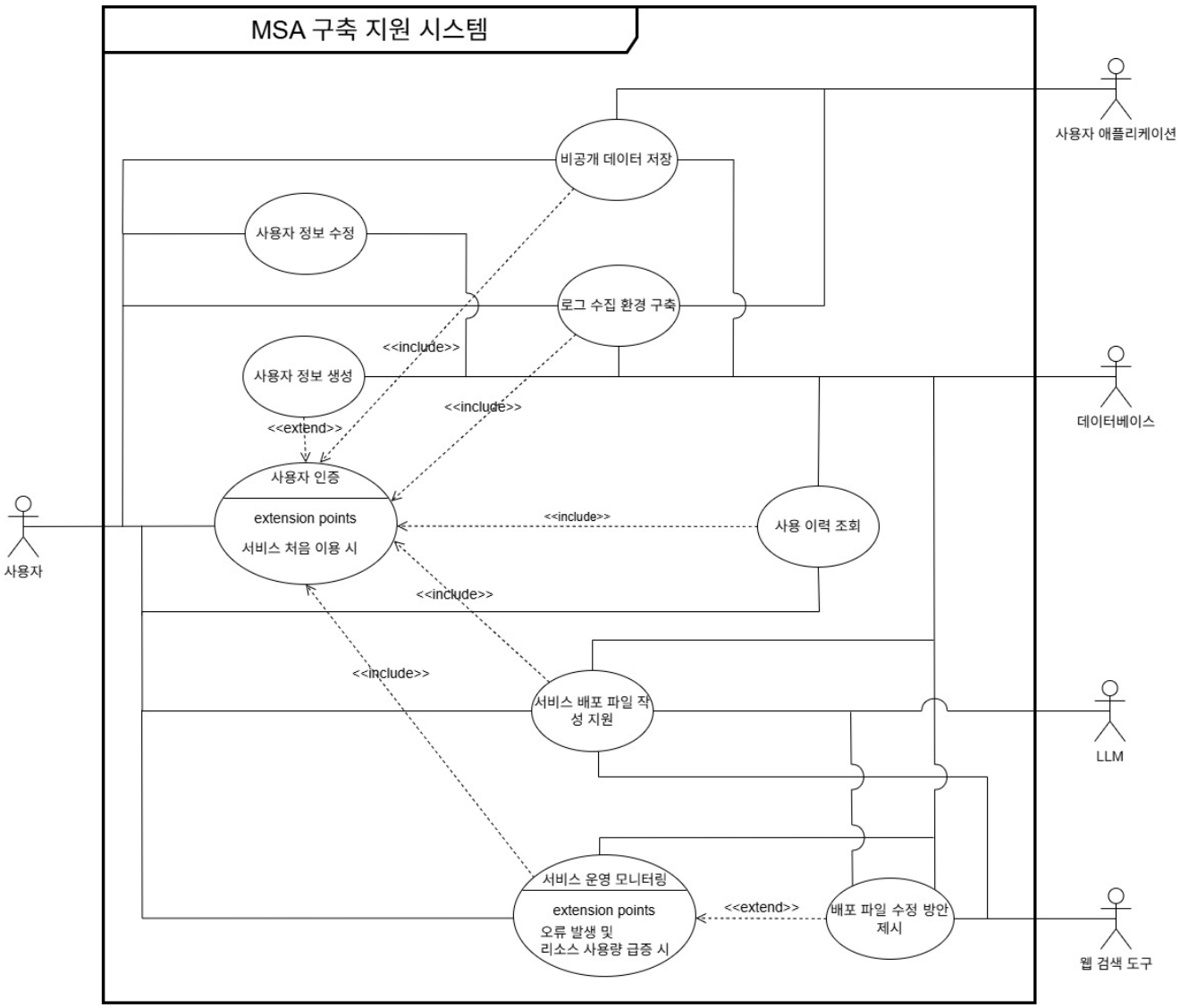

그림 1. MSA 구축 지원 시스템의 유스케이스 다이어그램

② Actor

표 3 은 Actor 에 대한 설명을 나타낸 표이다.

표 3. Actor

요건	설명
사용자	시스템을 사용하는 주체이다. 사용자 정보 생성 및 인증, 배포 명세 생성 및 수정 요청, 이력 조회 등 주요 기능을 사용한다.
사용자 애플리케이션	사용자의 MSA 클러스터 환경에서 실행된다. 시스템 초기화 시 ELK 구성 요소(Filebeat, Logstash)가 애플리케이션에 배포되고, 로그 수집 대상이 된다. 시스템은 SSH 를 통해 애플리케이션에 원격 접속한다.
데이터베이스	사용자 애플리케이션 로그 데이터, 내부 데이터를 벡터 DB 형태로 저장하며, AI 서비스 API 키, SSH 접근 자격, 사용자 이력 등 사용자 정보도 저장한다.
LLM	사용자 요청 및 로그 분석 요청 시 내부 데이터 DB, 웹 문서, 로그 DB 검색 결과를 참고해서 적절한 배포 명세 또는 설정 개선안을 생성한다.
웹 검색 도구	외부 기술 문서(StackOverflow, GitHub, Docker Hub)를 검색해 LLM 의 응답 생성을 보조한다.

③ 유스케이스 명세

● 사용자 정보 생성

유스케이스명	사용자 정보 생성
개요	사용자는 회원가입을 통해 ID, 비밀번호, SSH 접속을 위한 접근 자격과 IP 주소, AI 서비스 API 키 등의 정보를 입력해서 계정을 생성한다.
관련 액터	사용자, 데이터베이스
선행 조건	없음
이벤트 흐름	기본흐름: 1. 사용자는 ID, 비밀번호, SSH 접속 정보, AI 서비스 API Key 등 회원가입에 필요한 모든 정보를 입력 후 제출한다. 2. 시스템은 ID 중복 여부를 확인한다. 3. 시스템은 입력된 SSH 접속 정보로 접근 가능여부(ping 테스트)를 확인한다. 4. 시스템은 AI 서비스 API 키로 샘플 요청이 가능한지 확인한다. 5. 시스템은 사용자 계정을 데이터베이스에 저장한다. 대안흐름: 2-1. ID 가 데이터베이스에 이미 존재하는 경우 1. 시스템은 사용자에게 입력한 ID 가 이미 사용중임을 알리는 에러 메시지를 출력한다. 3-1. SSH 접속 정보로 접근이 불가능한 경우 1. 시스템은 사용자에게 SSH 접속 정보가 유효하지 않음을 알리는 에러 메시지를 출력한다. 4-1. AI 서비스 API 키로 샘플 요청이 불가능한 경우 1. 시스템은 사용자에게 API 키가 유효하지 않음을 알리는 에러 메시지를 출력한다.
후행 조건	사용자는 사용자 정보를 생성한다.

● 사용자 정보 수정

유스케이스명	사용자 정보 수정
개요	사용자는 등록한 비밀번호, SSH 접속 정보, API 키 등을 수정한다.
관련 액터	사용자, 데이터베이스
선행 조건	사용자가 로그인되어 있어야 한다.
이벤트 흐름	기본흐름: 1. 사용자가 사용자 정보(SSH 접속 정보, API 키 값)를 수정한 값을 시스템에 전달한다. 2. 시스템은 입력된 SSH 접속 정보로 접근 가능여부(ping 테스트)를 확인한다. 3. 시스템은 입력된 API 키로 샘플 요청이 가능한지 확인한다. 4. 수정된 정보가 데이터베이스에 반영된다. 대안흐름: 2-1. SSH 접속 정보로 접근이 불가능한 경우 1. 시스템은 사용자에게 SSH 접속 정보가 유효하지 않음을 알리는 에러 메시지를 출력한다. 3-1. AI 서비스 API 키로 샘플 요청이 불가능한 경우 1. 시스템은 사용자에게 API 키가 유효하지 않음을 알리는 에러 메시지를 출력한다.
후행 조건	데이터베이스는 수정된 사용자 정보를 저장한다.

● 사용자 인증

유스케이스명	사용자 인증
개요	사용자는 계정 정보를 통해 시스템에 로그인하고, 시스템은 인증된 사용자만 주요 기능에 접근할 수 있다.
관련 액터	사용자, 데이터베이스
선행 조건	없음
이벤트 흐름	기본흐름: 1. 사용자가 ID 와 비밀번호를 입력해 사용자 인증을 요청한다. 2. 시스템은 데이터베이스에서 해당 ID 의 사용자 정보를 조회한다. 3. 시스템은 해당 사용자의 비밀번호와 입력된 비밀번호를 비교하여 일치하는지 판단한다. 대안흐름: 3-1. 입력된 비밀번호가 데이터베이스 내부 정보와 일치하지 않는 경우 1. 시스템은 사용자에게 인증 정보가 일치하지 않다는 오류 메시지를 출력한다.
후행 조건	사용자는 로그인하여 시스템의 기능을 사용할 수 있다.

● 로그 수집 환경 구축

유스케이스명	로그 수집 환경 구축
개요	사용자는 ELK 스택 중 로그 수집기(Filebeat, Logstash)를 배포해야 하고, 시스템은 수집되는 로그를 벡터 DB 로 저장한다.
관련 액터	사용자, 데이터베이스, 사용자 애플리케이션
선행 조건	사용자는 시스템에 로그인해야 한다.
이벤트 흐름	기본흐름: 1. 시스템은 사용자의 애플리케이션에서 시스템의 데이터베이스로 로그를 전송할 수 있는 로그 수집기(Filebeat, Logstash) 배포 파일을 제공한다. 2. 사용자는 해당 배포 파일을 사용자 애플리케이션에 적용하여 로그 수집기를 구성한다. 3. 시스템은 데이터베이스에 로그 수집 여부를 확인하는 쿼리를 수행하여, 로그 수집기가 정상 동작 하는지 확인한다. 대안흐름: 3-1. 사용자 애플리케이션의 로그가 일정 시간동안 수집되지 않을 경우 1. 시스템은 사용자에게 상태 점검을 요청하는 메시지를 출력한다. 예외흐름: 3-2. 로그 확인 쿼리 중 오류가 발생한 경우 1. 시스템은 데이터베이스 연결 상태를 점검하고 오류 내용을 기록한다.
후행 조건	시스템은 사용자 애플리케이션 모니터링을 수행한다.

● 비공개 데이터 저장

유스케이스명	비공개 데이터 저장
개요	사용자는 시스템에 사내 정책, 커스텀 이미지 등의 비공개 데이터를 업로드 해야 하고, 시스템은 비공개 데이터를 벡터 DB 로 저장한다.
관련 액터	사용자, 데이터베이스, 사용자 애플리케이션
선행 조건	사용자는 시스템에 로그인해야 한다.
이벤트 흐름	기본흐름: 1. 사용자는 내부 데이터(zip 파일)를 업로드한다. 2. 시스템은 zip 파일을 사용자별 임시 디렉토리에 압축 해제한다. 3. 시스템은 해제된 파일 중, 지원하지 않는 포맷(.exe, .sh, .bat 등)을 제외한 나머지를 데이터베이스에 저장한다. 4. 시스템은 사용자에게 처리 결과 메시지를 출력한다. 대안흐름: 3-1. 압축 해제된 파일 중 저장 가능한 포맷의 파일이 존재하지 않는 경우 1. 시스템은 사용자에게 저장 가능한 데이터가 없다는 메시지를 출력한다. 예외흐름: 3-2. 데이터베이스에 저장 중 오류가 발생한 경우 1. 시스템은 저장 작업을 롤백하고 사용자에게 저장 실패 메시지를 출력한다.
후행 조건	시스템은 내부 데이터를 데이터베이스에 저장한다.

● 서비스 배포 파일 작성 지원

유스케이스명	서비스 배포 파일 작성 지원
개요	시스템은 사용자의 질의를 바탕으로, 내부 데이터 DB 및 웹 문서 검색 결과를 참고해서 각 설정 항목에 주석과 해설이 포함된 배포 파일을 새로 생성하거나 개선안을 제안한다.
관련 액터	사용자, 웹 검색 도구, 데이터베이스, LLM
선행 조건	사용자는 시스템에 로그인해야 한다.
이벤트 흐름	기본흐름: 1. 사용자가 배포하고자 하는 서비스 설명 또는 약결합 개선이 필요한 배포 파일에 대한 설명을 자연어로 입력한다. 2. 시스템은 사용자 입력을 기반으로 데이터베이스에서 유사도를 기반으로 내부 데이터를 조회한다. 3. 웹 검색 도구를 활용하여 사용자 입력에 대한 자료를 탐색한다. 4. 조회된 내부 데이터, 탐색된 웹 검색 자료, 사용자 입력을 기반으로 마이크로서비스 배포 명세와 배포 명세에 대한 설명을 생성하기 위한 프롬프트를 작성한다. 5. 작성된 프롬프트를 기반으로 LLM 에 생성을 요청한다. 6. 시스템은 생성된 배포 명세와 설명을 사용자에게 제공한다. 예외흐름: 2-1. 데이터베이스 조회 중 오류가 발생한 경우 1. 시스템은 데이터베이스 연결 상태를 점검하고 오류 내용을 기록한다.
후행 조건	사용자는 시스템이 제시한 배포 명세 파일을 참고할 수 있다.

● 서비스 운영 모니터링

유스케이스명	서비스 운영 모니터링
개요	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 판단한다.
관련 액터	사용자, 데이터베이스
선행 조건	사용자가 로그인한 상태이며, 사용자 애플리케이션 로그 수집이 정상적으로 수행되어야 한다.
이벤트 흐름	기본흐름: 1. 시스템은 데이터베이스에서 로그인 된 사용자의 로그를 3 분 주기로 조회한다. 2. 시스템은 해당 사용자의 로그에 대해 텍스트 기반 분석을 수행한다. 3. 시스템은 사용자 애플리케이션의 평균 리소스 사용량을 기반으로 권장 설정 값을 도출한다. 대안흐름: 3-1. 사용자 애플리케이션에 오토 스케일링이 이미 적용된 경우 1. 시스템은 리소스 사용량 패턴을 분석해 리소스 한계값 조정 제안을 제공한다. 예외흐름: 2-1. 오류 로그를 발견한 경우 1. 시스템은 문제를 해결할 수 있는 오류 개선안을 사용자에게 제공한다. 3-2. 리소스 사용량이 권장 설정 값을 지속적으로 초과하는 경우 1. 시스템은 사용자 애플리케이션에 리소스 개선안을 제안한다.
후행 조건	없음

● 배포 파일 수정 방안 제시

유스케이스명	배포 파일 수정 방안 제시
개요	시스템은 이상 동작이 감지된 경우, 내부 데이터 DB, 로그 DB, 웹 문서 검색 결과를 참고하여 배포 파일 수정 방안을 사용자에게 제공한다.
관련 액터	사용자, 데이터베이스, LLM, 웹 검색 도구
선행 조건	서비스 모니터링 결과, 오류 또는 리소스 사용량이 권장 설정 값을 초과하는 경우
이벤트 흐름	기본흐름: 1. 시스템은 데이터베이스에서 오류 상황과 관련된 자료를 검색한다. 2. 시스템은 웹 검색 도구를 활용하여 오류 상황과 관련된 자료를 검색하고 수집한다. 3. 시스템은 수집된 정보들을 기반으로 배포 명세 수정 프롬프트를 생성한다. 4. LLM 은 생성된 프롬프트를 기반으로 적절한 배포 명세 수정과 설명을 생성한다. 5. 시스템은 생성된 배포 명세와 설명을 사용자에게 제공한다. 예외흐름: 1-1. 데이터베이스 조회 중 오류가 발생한 경우 1. 시스템은 데이터베이스 연결 상태를 점검하고 오류 내용을 기록한다.
후행 조건	사용자에게 적용 가능한 개선안이 제공된다.

● 사용 이력 조회

유스케이스명	사용 이력 조회
개요	사용자가 시스템 내에서 이전에 수행한 서비스 배포 지원 요청 또는 개선 작업 등의 이력을 확인할 수 있다.
관련 액터	사용자, 데이터베이스
선행 조건	사용자가 로그인한 상태여야 한다.
이벤트 흐름	기본흐름: 1. 시스템은 데이터베이스에서 사용자의 작업 이력 조회한다. 2. 시스템은 조회한 이력들을 최신순으로 정렬한다. 3. 시스템은 이력 목록을 사용자에게 출력한다. 대안흐름: 1-1. 사용자의 작업 이력이 없을 경우 1. 시스템은 사용자에게 작업 이력이 없다는 내용을 출력한다.
후행 조건	사용자는 목록에서 특정 항목을 선택하여 상세 내용을 확인할 수 있다.

3. 현실적 제약 사항 분석 결과 및 대안

1) 현실적 제약 사항

① 실제 본 시스템을 이용하는 다수의 사용자를 모집하기가 어렵다.

② 시스템에 업로드 할 실제 산업 현장에서 사용되는 조직 내부 데이터를 구하기 어렵다.

③ 웹 기반 검색 수행 시 배포 명세와 관련된 모든 정보를 탐색하기는 어렵다.

2) 대안

① 팀원들이 10 개의 테스트용 계정을 생성해서 사용자 역할을 대체한다.

② 실제 내부 데이터 대신, 생성형 AI(ChatGPT)를 활용해 만든 더미 데이터를 문서화해서 사용한다.

③ 웹 검색 도메인을 배포 파일과 관련된 사이트(Docker Hub, GitHub, Stack Overflow)로 제한한다.

4. 시스템 구성

1) 시스템 구성

그림 2 는 본 시스템의 전체 구성을 나타낸 그림이다.

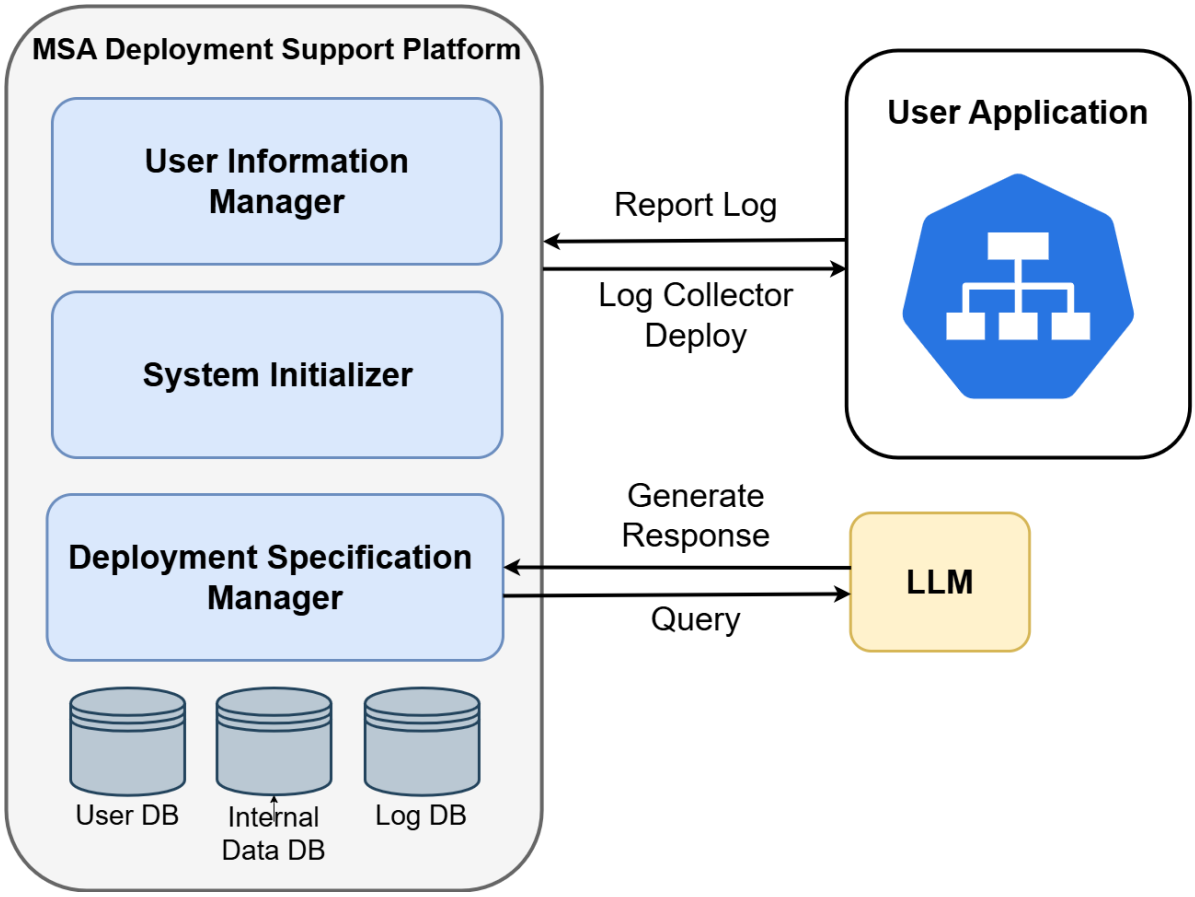

그림 2. 시스템 구성도

2) 개발 환경

그림 3 은 시스템 개발 환경의 구성도를 나타낸다.

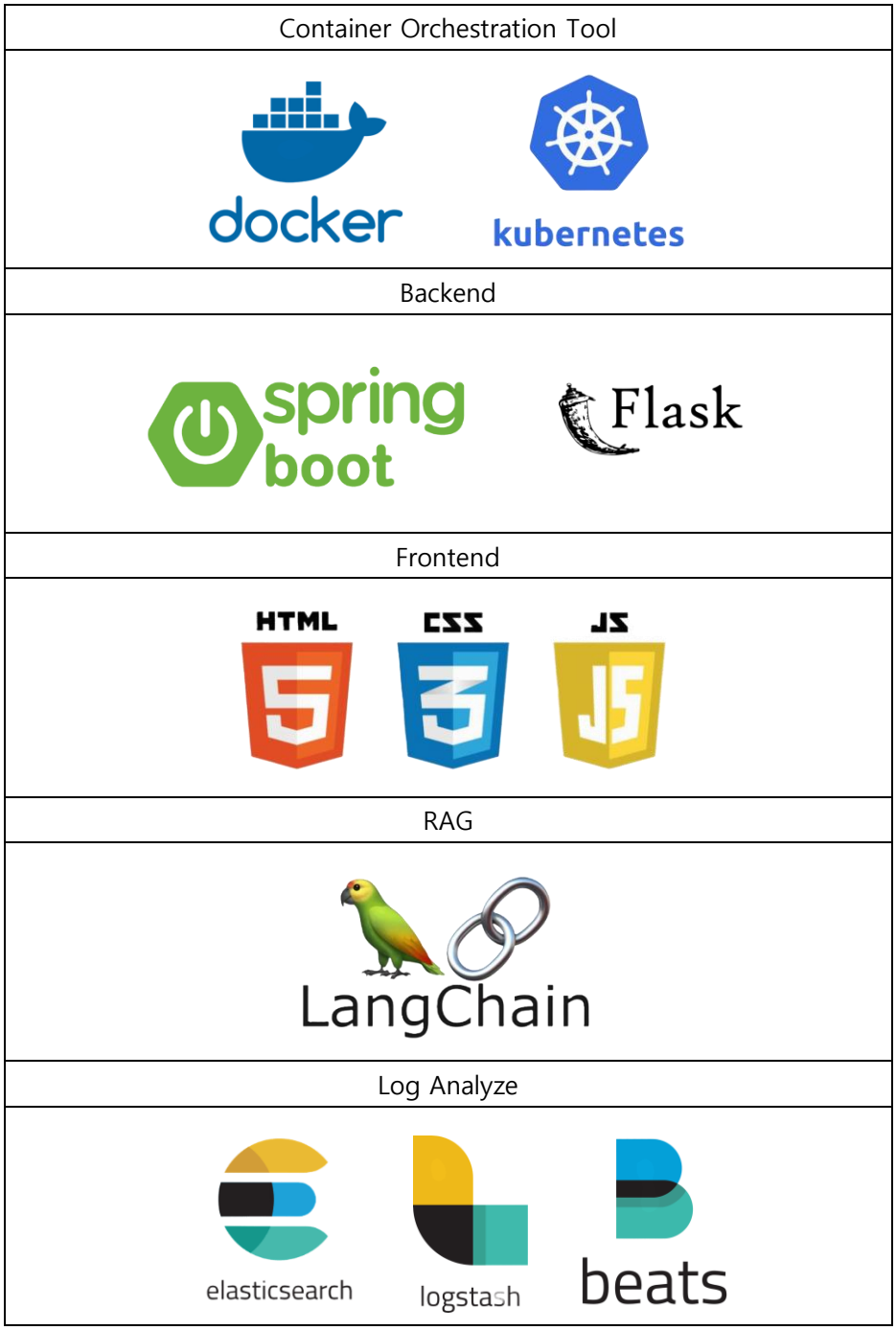

그림 3. 시스템 개발 환경 구성도

표 4 는 시스템 개발에서 사용되는 관련 기술들을 나타낸다.

표 4. 개발 환경

개발환경	사용기술
소프트웨어 형상관리(SCM)	Github
컨테이너 오케스트레이션 도구	Docker, Kubernetes
Back-end	Spring Boot, Flask
Front-end	HTML, CSS, JavaScript
RAG	LangChain
로그 분석	Elasticsearch, Logstash, Filebeat

3) 사용 기술

① Docker

Docker 는 애플리케이션 실행에 필요한 환경을 하나의 컨테이너 이미지로 패키징하여 어디서든 동일한 방식으로 실행되도록 보장하는 오픈소스 기반 컨테이너 플랫폼이다. 운영체제 레벨에서 격리된 환경을 제공하여 개발 환경과 운영 환경 간의 차이를 제거하고 배포 및 관리를 단순화한다.

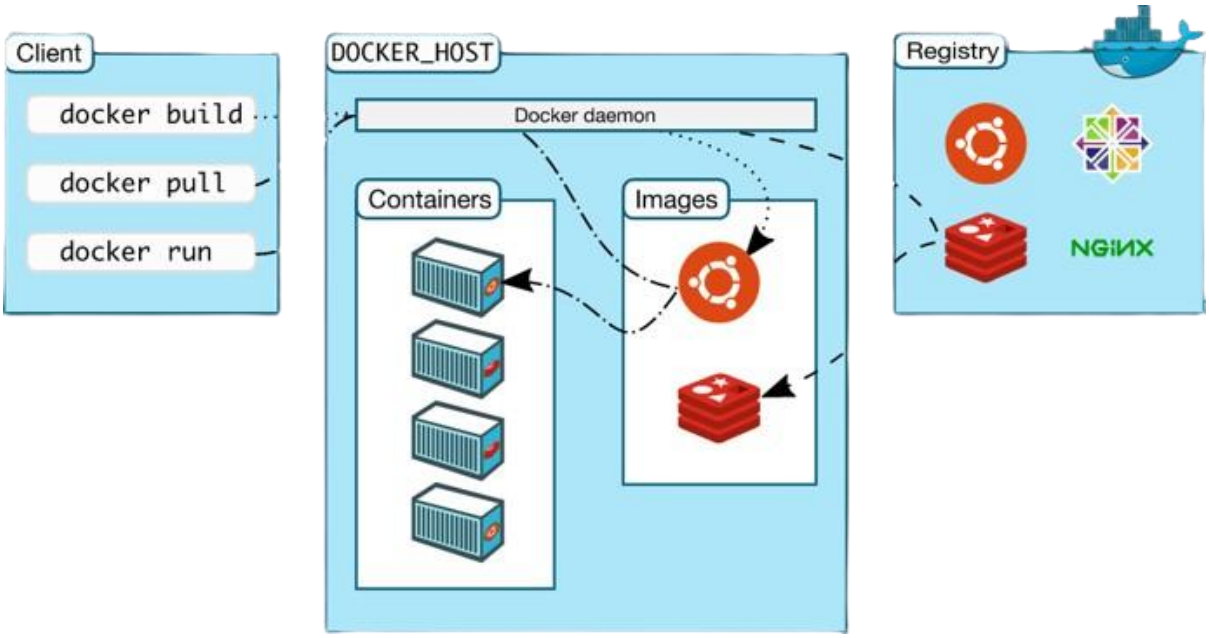

그림 4. 도커 아키텍처

Docker 는 다음과 같은 주요 구성 요소로 이루어져 있다:

i) Docker Daemon
도커 데몬은 도커의 핵심 엔진으로, 도커 API 요청을 수신하고 이미지, 컨테이너, 네트워크, 볼륨 등을 관리한다. 시스템 백그라운드에서 실행되며 컨테이너 실행, 정지, 삭제 같은 명령을 처리한다.

ii) Docker Client
사용자가 도커 명령어(CLI)를 통해 도커 데몬과 상호작용할 수 있게 해주는 클라이언트 인터페이스다.

iii) Docker Registry
이미지를 저장하고 관리하는 중앙 저장소로, 기본적으로 Docker Hub 를 사용하며 사내 프로젝트에서는 프라이빗 레지스트리를 운영할 수 있다. 컨테이너 이미지는 이 레지스트리에서 다운로드되거나 업로드된다.

iv) Docker Image
Dockerfile 에 정의된 명령들을 계층적으로 빌드하여 만들어지는 실행 환경 스냅샷이다. 이 이미지를 기반으로 컨테이너가 생성된다.

v) Container
컨테이너는 이미지의 실행 인스턴스로, 격리된 공간에서 애플리케이션을 실행한다. 호스트 OS 를 공유하면서도 독립된 환경을 갖기 때문에 리소스를 효율적으로 사용할 수 있다.

vi) Docker Network
컨테이너 간 통신을 위한 네트워크 환경을 제공한다. 각 컨테이너는 브리지, 호스트, 오버레이 네트워크 등을 통해 서로 통신할 수 있다.

vii) Docker Volume
호스트 시스템에서 컨테이너 데이터를 안전하게 보관하기 위한 방법이다. 컨테이너 재시작 시에도 데이터 유실이 없도록 지원한다.

② Kubernetes

Kubernetes 는 컨테이너화된 애플리케이션을 자동으로 배포, 관리, 확장 및 복구하는 오픈소스 컨테이너 오케스트레이션 플랫폼이다. 복잡한 마이크로서비스 환경을 안정적으로 운영할 수 있도록 설계되었다.

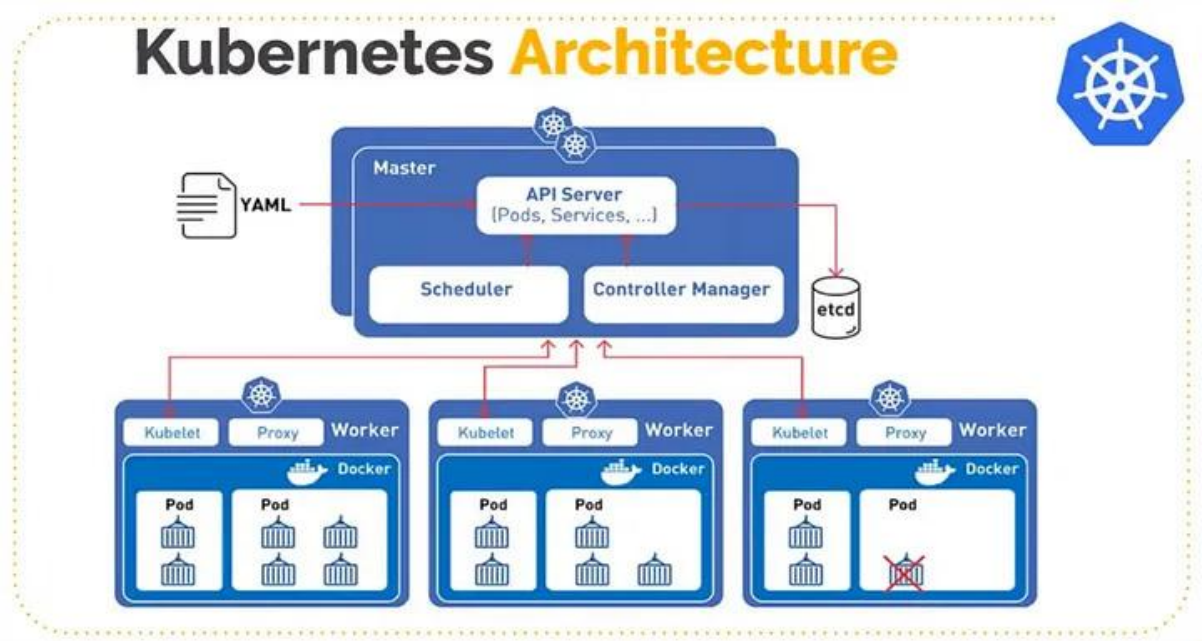

그림 5. 쿠버네티스 아키텍처

쿠버네티스는 다음과 같은 주요 구성 요소로 이루어져 있다.

i) Master Node
클러스터의 중앙 제어 장치 역할을 한다. 전체 클러스터의 상태를 추적하고, 각 워커 노드에 작업을 할당하는 기능을 담당한다.

- API Server: 클러스터 외부와 내부에서 발생하는 모든 요청을 수신하고 처리하는 진입점이다.
- Scheduler: 새로 생성된 파드를 어떤 워커 노드에 배치할지 결정한다.
- Controller Manager: 파드 수 유지, 노드 상태 확인 등 다양한 제어 루프를 수행한다.
- etcd: 클러스터의 상태 정보를 저장하는 분산 키-값 저장소이다.

ii) Worker Node
컨테이너가 배포되어 실행되는 노드다.

- kubelet: 마스터 노드로부터 명령을 수신해 파드를 생성하고 상태를 모니터링한다.
- kube-proxy: 파드 또는 외부 네트워크 간의 통신을 중계한다.
- Container Runtime: Docker 등 실제 컨테이너를 실행하는 런타임이다.

iii) Pod

가장 작은 배포 단위로, 하나 이상의 컨테이너가 실행되는 단위를 말한다. 동일한 네트워크 네임스페이스와 볼륨을 공유한다.

iv) Deployment

파드의 상태(예: 레플리카 수, 컨테이너 이미지 버전 등)를 사용자가 정의하면, Kubernetes 가 이를 자동으로 관리하는 리소스이다. 실제 애플리케이션이 지속적으로 안정적으로 운영되도록 한다.

v) Service

동적으로 생성되고 소멸되는 파드들을 외부에서 고정된 주소로 접근할 수 있도록 해주는 추상화 객체다. 클러스터 외부에서 내부 파드에 접근할 수 있도록 라우팅 역할을 수행한다.

③ Filebeat

Filebeat 는 애플리케이션 로그, 사용자 정의 로그 등 다양한 로그 파일을 실시간으로 수집하여 중앙 시스템으로 전송하는 경량 로그 수집 도구이다. 각 서버에 에이전트 형태로 설치되어 지정된 로그 파일을 Logstash 또는 Elasticsearch 로 전달한다. 효율적인 로그 수집 환경을 구성하는 데 사용된다. Filebeat 의 주요 구성 요소는 다음과 같다.

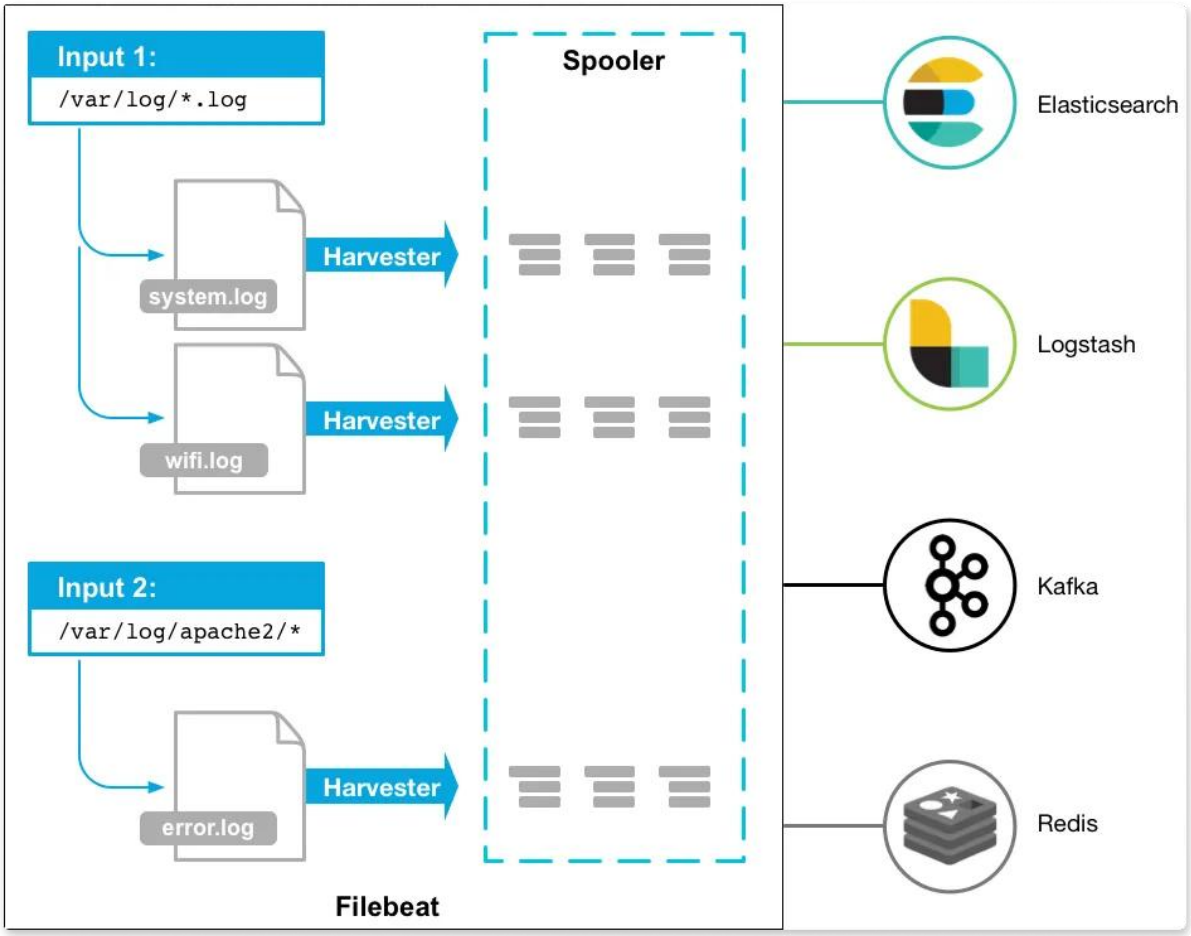

그림 6. Filebeat 아키텍처

i) Prospector / Input
로그 수집 대상 파일을 정의한다. 지정된 경로에 따라 각 로그 파일마다 Harvester를 생성한다.

ii) Harvester
실제로 로그 파일을 읽는 구성 요소이다. 파일의 변경 사항을 지속적으로 추적하는 방식으로 로그 데이터를 수집한다.

iii) Spooler
수집된 로그 데이터를 메모리 버퍼에 임시로 저장한다. 버퍼 단위로 데이터를 묶어서 다음 단계로 전달한다.

iv) Publisher
Spooler로부터 전달받은 데이터를 Logstash, Elasticsearch 등의 외부 시스템에 전송한다. 이 과정에서 포맷 변환이나 인코딩 처리가 가능하다.

④ Logstash

Logstash 는 실시간 데이터 수집 및 처리 파이프라인 구성 도구이다. 다양한 형식의 로그 데이터를 입력 받아 가공한 후, 출력 대상(Elasticsearch, 파일, 데이터베이스)으로 전송한다. Logstash 의 주요 구성 요소는 다음과 같다.

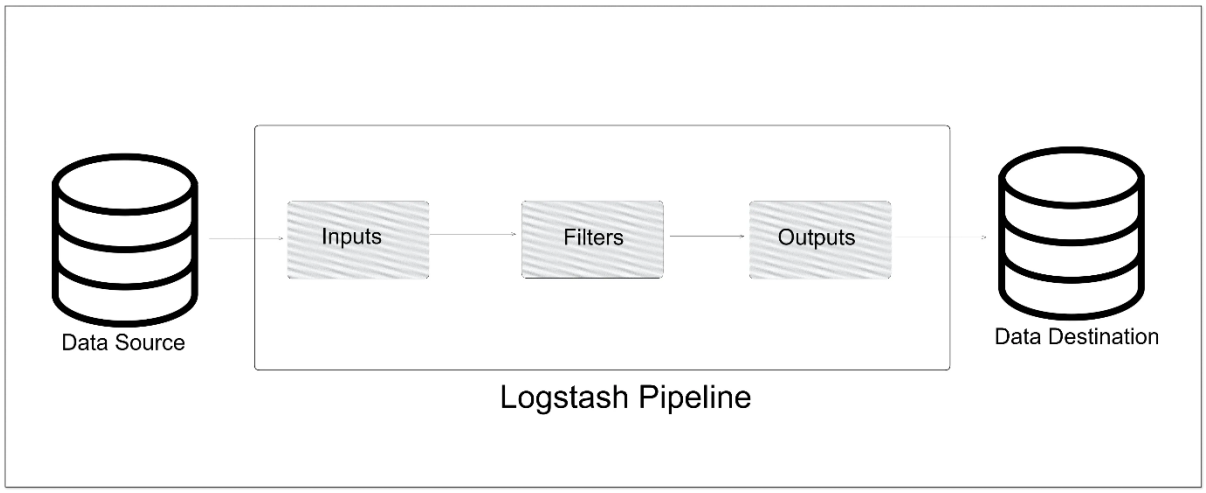

그림 7. Logstash Pipeline

i) Input Plugin
로그 데이터를 수신하는 단계이다. Filebeat, TCP, UDP, Kafka 등 다양한 방식으로 데이터를 수신할 수 있다.

ii) Filter Plugin
수신된 데이터를 변환하거나 특정 필드를 추출하는 중간 처리 단계다. 다양한 플러그인(Grok, mutate, date, geoip, kv 등)들 활용하여 로그 포맷을 정제하고 구조화 할 수 있다.

iii) Output Plugin
처리된 데이터를 출력 대상(Elasticsearch, 파일, 데이터베이스)으로 전송한다.

⑤ Elasticsearch

Elasticsearch 는 JSON 기반 분산 검색 및 분석 엔진으로, 다양한 구조의 데이터를 실시간으로 검색할 수 있도록 설계된 시스템이다. Apache Lucene 을 기반으로 하고, RESTful API 를 통해 데이터의 저장, 조회, 집계를 처리한다. Elasticsearch 의 주요 구성 요소는 다음과 같다.

i) Node
Elasticsearch의 실행 단위로, 데이터 저장 및 검색이 가능하다. 여러 노드가 하나의 클러스터를 형성한다.

ii) Cluster
여러 노드의 집합이다. 클러스터 이름으로 식별할 수 있고, 데이터와 Index를 통합적으로 관리한다.

iii) Index
데이터를 저장하는 논리적 단위이다. 하나의 인덱스는 다수의 Shard로 구성될 수 있다.

iv) Document
실제로 저장되는 데이터 단위이며, JSON형식으로 표현된다. 각각의 Document는 필드와 값을 포함하고, 텍스트, 수치, 벡터 값을 가질 수 있다.

v) Shard & Replica
Shard는 Index를 나누어 저장하기 위한 물리적 단위이다. Replica는 Shard의 복제본으로, 고가용성과 데이터 안정성을 보장하기 위해 사용한다.

Elasticsearch 는 벡터 데이터 기반의 유사도 검색 기능이 추가되면서 벡터 데이터베이스로도 활용되고 있다. 벡터 검색 지원 기능은 다음과 같다.

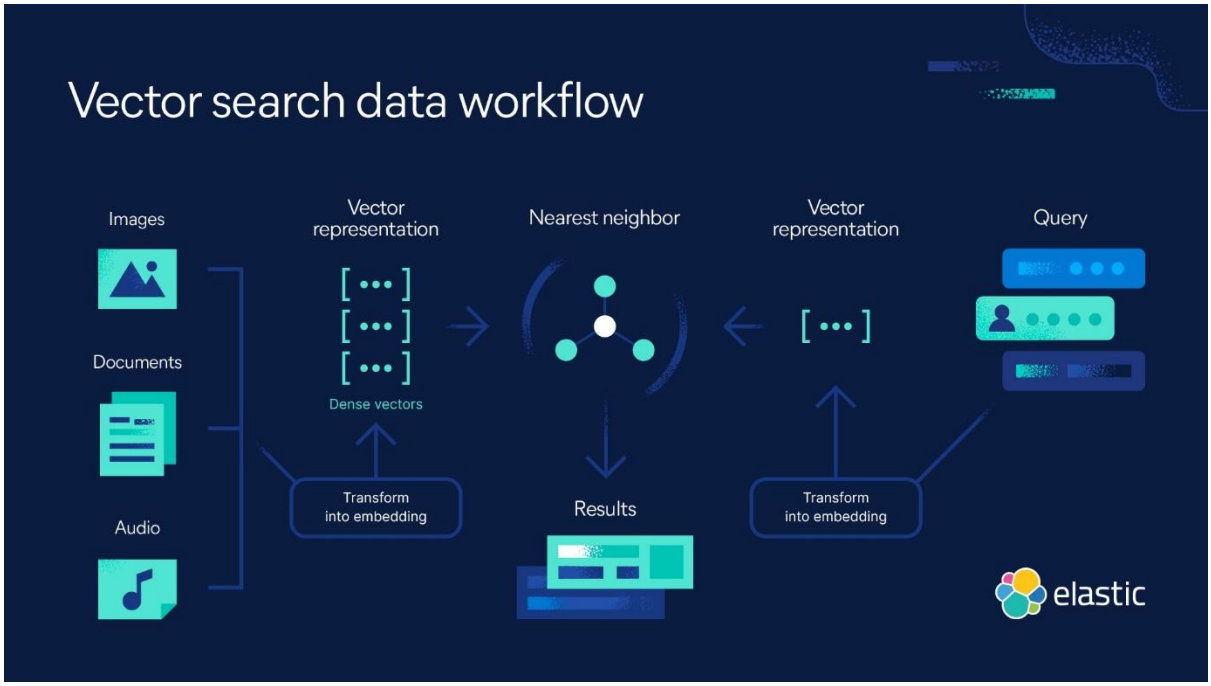

그림 8. Vector search data workflow

i) dense_vector 필드 타입
벡터를 문서 내 필드로 저장할 수 있고, 최대 16,384차원의 실수형 벡터를 지원한다.

ii) KNN(K-Nearest Neighbors) 검색
Elasticsearch의 knn_search API를 사용하여, 특정 벡터와 가장 유사한 문서(nearest neighbor)를 찾아낼 수 있다.

iii) Hybrid Search 지원
텍스트 기반 BM25 검색과 벡터 기반 검색을 함께 수행해서 검색 품질을 높일 수 있다.

⑥ Langchain

LangChain(랭체인)은 대규모 언어 모델(LLM)을 활용한 애플리케이션 개발에 특화된 오픈소스

프레임워크다. 특히 LangChain 은 LLM 과 외부 도구(문서, DB, API 등)를 결합해 사용자 질의에 대해 문서 검색, 도구 호출, 대화 흐름 유지 등을 지원하는 기능을 제공한다. LangChain 의 대표적인 기능은 다음과 같다.

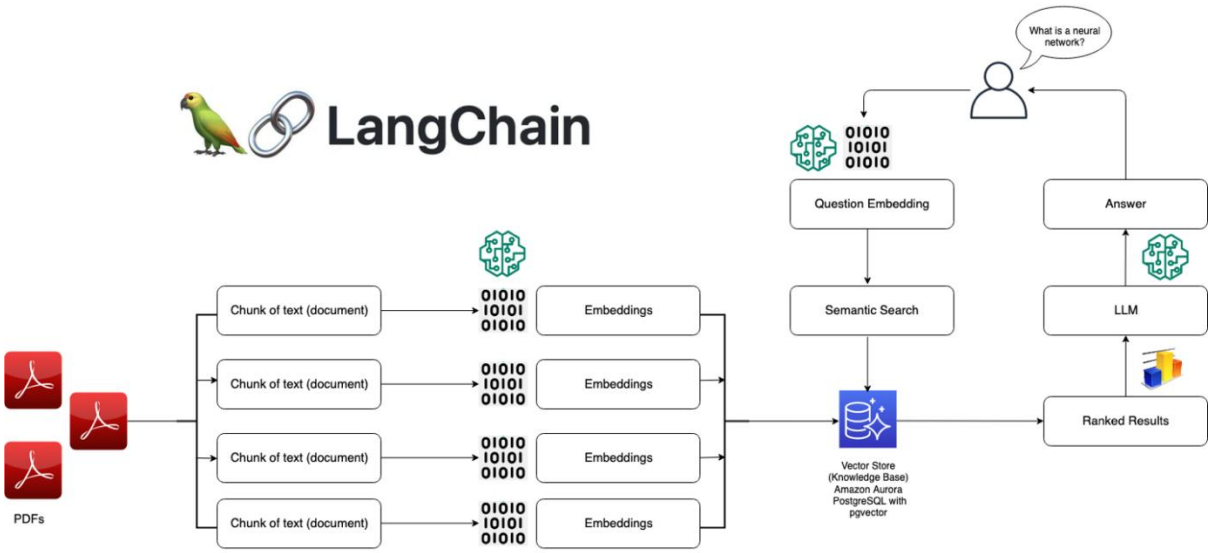

그림 9. LangChain 기반 RAG 아키텍쳐

i) Prompt Template
LLM에게 줄 입력 프롬프트를 템플릿화하여 구성하여 재사용할 수 있다.

ii) Document Loader
PDF, HTML, Github 등 다양한 소스에서 문서를 불러화 처리할 수 있다.

iii) Embeddings
문서를 벡터화하여 벡터DB에서 검색할 수 있게 한다.

iv) Retriever
사용자 질의와 유사한 문서를 벡터DB에서 찾아주는 모듈이다.

v) Chain
LLM 호출, 문서 검색, 응답 생성 등을 순차적으로 연결해 구성할 수 있게 하는 기능이다.

5. 개발 일정 및 역할 분담

1) 개발 일정

표 5 는 시스템 개발 일정을 나타낸 표이다.

표 5. 개발 일정

기간 / 수행 내용	5 월		6 월				7 월					8 월				9 월		
	3	4	1	2	3	4	1	2	3	4	5	1	2	3	4	1	2	3
필요 지식 습득	All																	
테스트용 MSA 애플리케이션 개발			All															
API 개발					신세환 설종환													
RAG 기반 답변 생성 파이프라인 구축					김휘수													
로그 수집기 배포 기능 개발									신세환									
내부 데이터 처리 기능 개발									설종환									
RAG Agent 개발									김휘수									
운영 모니터링 시스템 개발											신세환							
웹 UI 개발													All					
테스트 및 보완														All				
최종보고서 작성																All		

2) 역할 분담

표 6 은 개인별 시스템 역할 분담을 나타낸 표이다.

표 6. 역할 분담

이름	역할
공통	필요 지식 습득, 테스트용 MSA 애플리케이션 개발, 웹 UI 개발, 테스트 및 보완, 최종보고서 작성
김휘수	RAG 기반 답변 생성 파이프라인 구축, RAG Agent 개발
신세환	API 개발, 로그 수집기 배포 기능 개발, 운영 모니터링 시스템 개발
설종환	API 개발, 내부 데이터 처리 기능 개발