정보컴퓨터공학부 2025 전기 졸업과제 발표

RAG를 활용한 컨테이너 기반 마이크로서비스 운영 환경 관리 지원 시스템

팀명: 트리톤

201914116 김휘수
202055645 신세환
202255663 설종환

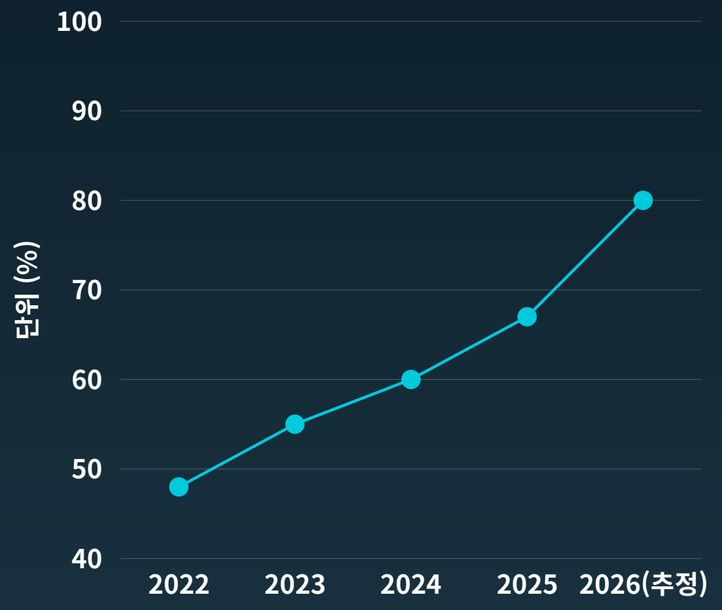
목 차

01.

연구 배경

01.
기업의 LLM & MSA 도입 증가

LLM(대형 언어 모델) 도입 비율



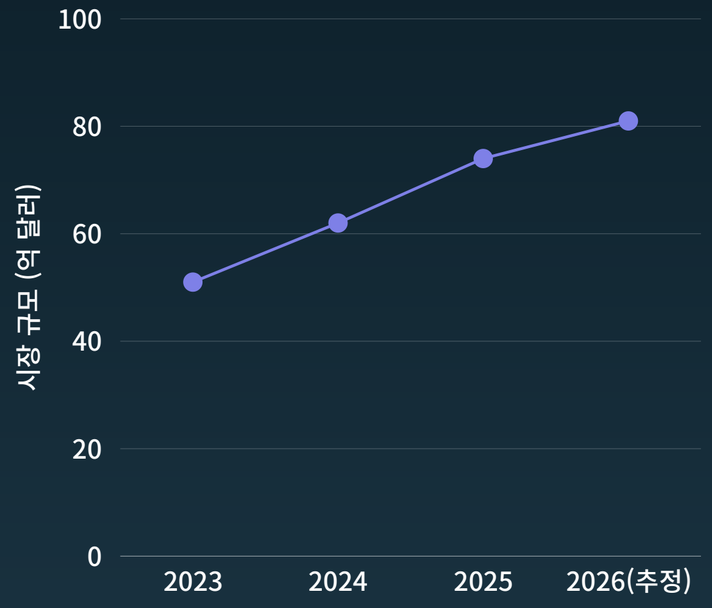
출처: Menlo, Hostinger

최근 기업 환경에서,

LLM 도입은 약 **80%**
MSA 시장규모는 80억 달러

MSA와 LLM의 확산은
운영 방식의 변화를
만들어내고 있음

MSA 시장 규모



출처: The Business Research Company

01.

MSA (Microservice Architecture)의 특징

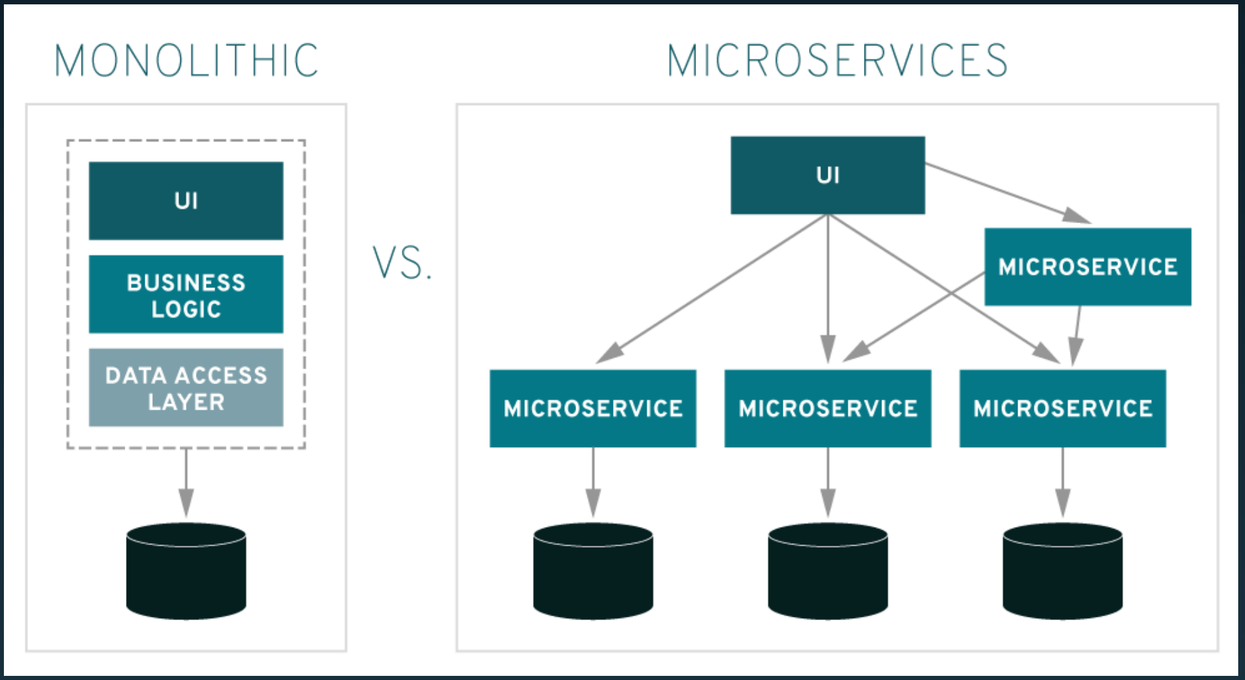


그림 1 Monolithic 구조 vs MSA

다수의 마이크로서비스 & 의존 관계
규모 확장 시 모놀리식보다 **복잡한 구조**

배포 명세 작성 난이도 & **운영** 복잡성 증가

MSA 배포·운영 과정에 LLM 도구 도입 시
업무 효율성 개선 기대

02.

기존 문제점

02.

조직 내부 데이터 기반 명세 작성의 어려움

배경

- LLM의 학습 데이터 **의존성**
- 배포 **명세 요구사항 파악 및 반영 필요**
- 배포를 위해 각 서비스의 **도메인 지식** 필요

문제점

- 외부 자동화 도구의 **내부 데이터 반영** 어려움
- 배포 담당자가 **모든 도메인 지식**을 갖추기 어려움
- **대규모 MSA**의 배포 명세 작성 **정확도 저하**

02.
파인튜닝의 한계

배경

- 새로운 데이터마다 **모델 전체 재학습** 과정 필요
- **답변 출처** 제시 불가

문제점

- 재학습 시간으로 인해 **최신 데이터 반영 지연**
- 재학습 과정에서 발생하는 **높은 비용**과 **시간** 소요
- **신속성** 확보와 **정확성** 검증의 어려움

02.

로그 및 리소스 메트릭 기반 개선 기능 부족

배경

- 기존 시스템(Prometheus, Grafana 등)은 리소스 상태 확인·시각화 중심

문제점

- 운영 로그·리소스 메트릭 **활용 기능** 부족
- 운영 데이터 기반 **배포 명세 개선 기능** 부재

03.

연구 목표

03.
연구 목표

- 배포 명세 작성을 위한 조직 내부 데이터의 정의와 분류 기준 제시
- RAG 기반 배포 명세 생성 기능 구현
- 에러 로그 및 리소스 메트릭 수집을 통한 배포 명세 동적 개선 방안 제시

04.

연구 내용

04.
조직 내부 데이터의 정의와 분류 기준 제시

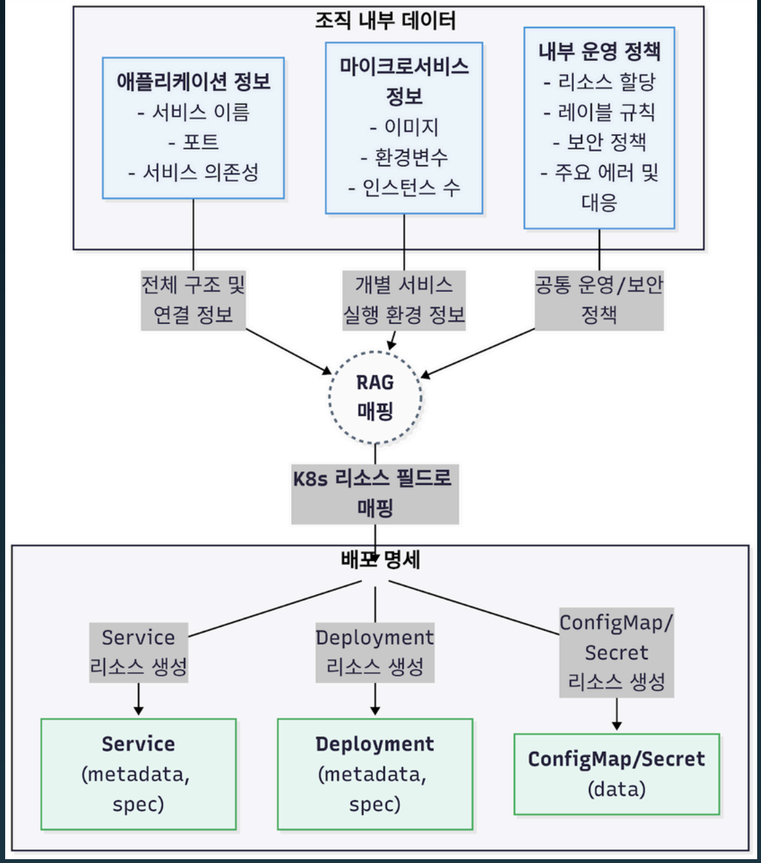


그림 1 조직 내부 데이터를 활용한 배포 명세 생성 과정

구성 항목 / 정보 요소	데이터 항목	배포 명세 필드	데이터 항목 정의
애플리케이션 정보	서비스 이름	metadata .name	애플리케이션을 구분하기 위한 고유 이름
	내부 서비스 포트	spec .ports	서비스 간 통신에 사용되는 포트
	의존 서비스	data	애플리케이션을 구성하는 마이크로서비스 목록
	외부 노출 방식	spec.type	외부에서 서비스에 연결하기 위한 공개 방식

그림 2 내부 데이터 항목과 배포 명세 필드 연관관계 일부

조직 내부 데이터를 정보 요소의 **목적과 범위**에 따라 분류

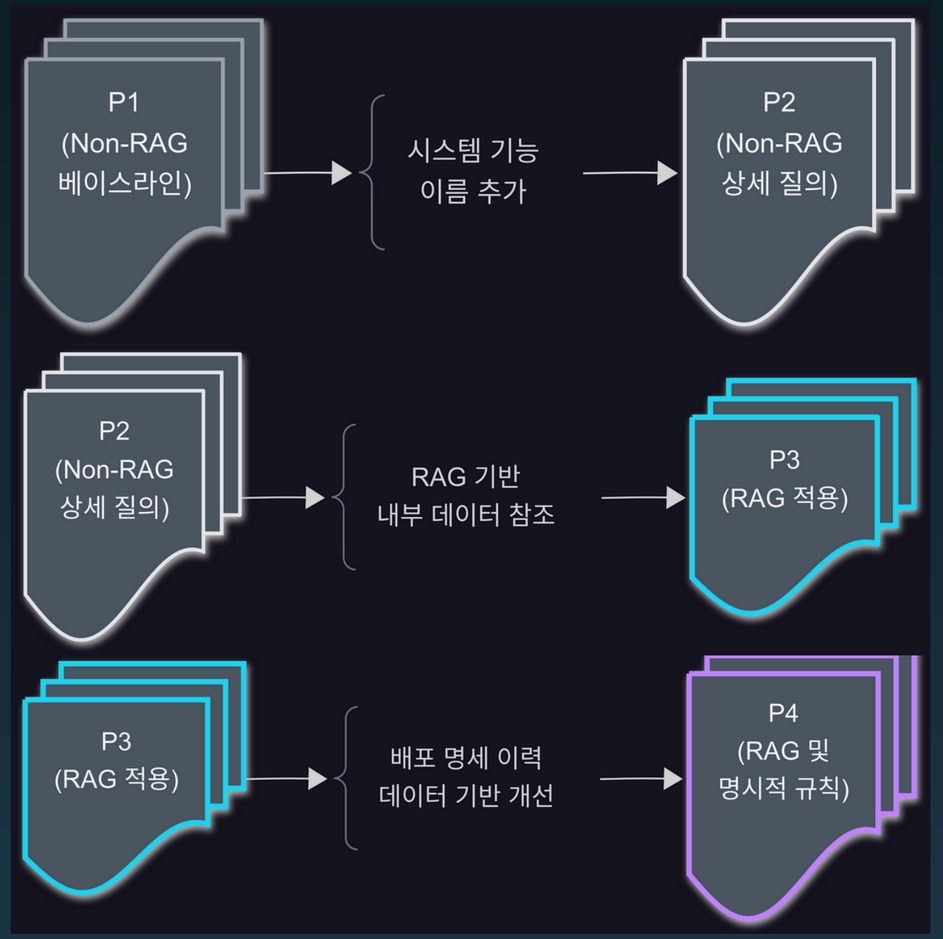
각 데이터의 항목과 배포 명세 필드 간의 **연관 관계 분석**

시스템이 설정한 기준에 따라 분류된 **데이터를 참조**하여 **배포 명세를 생성**할 수 있는 방법을 도출

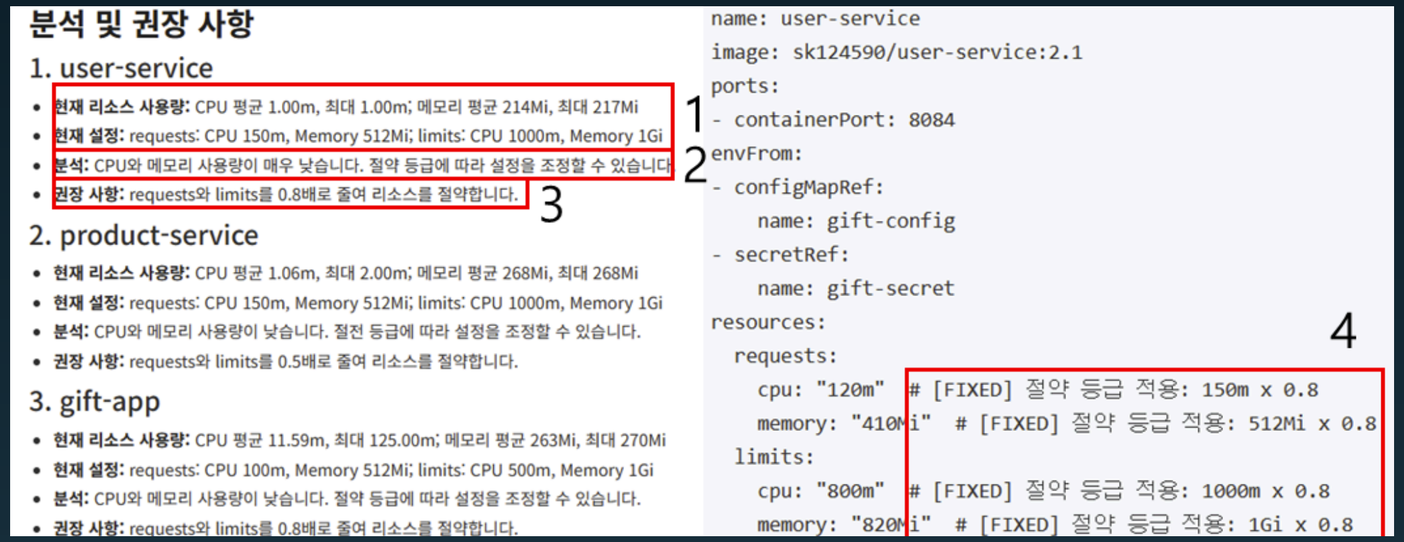
04.
RAG 기반 배포 명세 생성

최종 프롬프트	검증 목표
P1 (Non-RAG 베이스라인)	Non-RAG 환경에서 단순 질의의 성능 측정
P2 (Non-RAG + 상세 질의)	Non-RAG 환경에서 상세 질의의 구조 생성 능력 검증
P3 (RAG 적용)	RAG 적용 시 조직 내부 데이터 반영 정확성 능력 검증
P4 (RAG + 명시적 규칙)	명시적 규칙의 문법 및 정책 준수 효과 검증

그림 1 단계별로 개선된 프롬프트



04.
에러 로그 및 리소스 메트릭 수집을 통한 배포 명세 동적 개선



분석 및 권장 사항

1. user-service

- **현재 리소스 사용량:** CPU 평균 1.00m, 최대 1.00m; 메모리 평균 214Mi, 최대 217Mi
- **현재 설정:** requests: CPU 150m, Memory 512Mi; limits: CPU 1000m, Memory 1Gi
- **분석:** CPU와 메모리 사용량이 매우 낮습니다. 절약 등급에 따라 설정을 조정할 수 있습니다.
- **권장 사항:** requests와 limits를 0.8배로 줄여 리소스를 절약합니다.

2. product-service

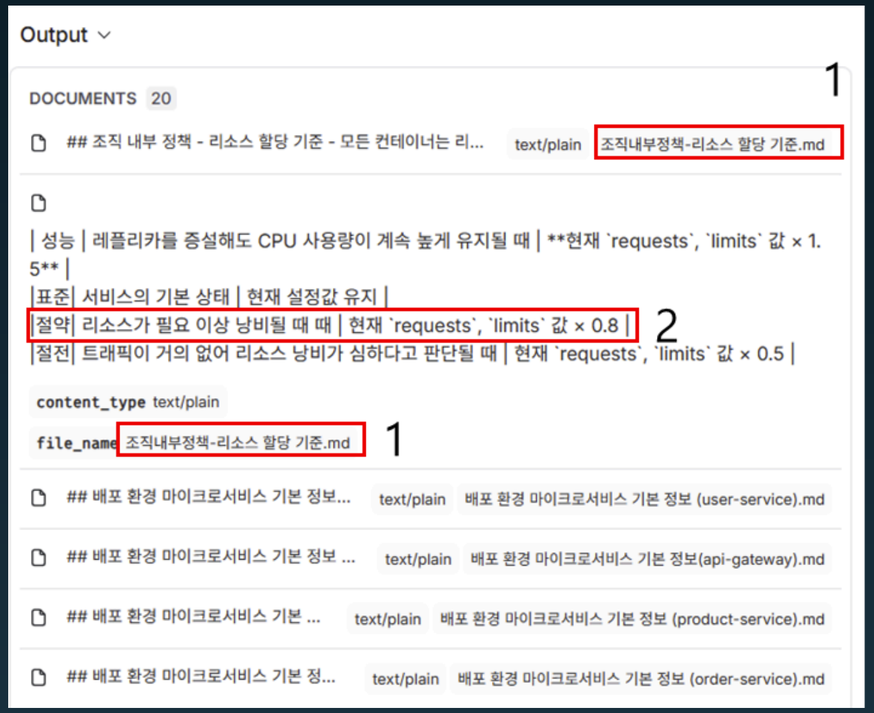
- **현재 리소스 사용량:** CPU 평균 1.06m, 최대 2.00m; 메모리 평균 268Mi, 최대 268Mi
- **현재 설정:** requests: CPU 150m, Memory 512Mi; limits: CPU 1000m, Memory 1Gi
- **분석:** CPU와 메모리 사용량이 낮습니다. 절전 등급에 따라 설정을 조정할 수 있습니다.
- **권장 사항:** requests와 limits를 0.5배로 줄여 리소스를 절약합니다.

3. gift-app

- **현재 리소스 사용량:** CPU 평균 11.59m, 최대 125.00m; 메모리 평균 263Mi, 최대 270Mi
- **현재 설정:** requests: CPU 100m, Memory 512Mi; limits: CPU 500m, Memory 1Gi
- **분석:** CPU와 메모리 사용량이 낮습니다. 절약 등급에 따라 설정을 조정할 수 있습니다.
- **권장 사항:** requests와 limits를 0.8배로 줄여 리소스를 절약합니다.

```
name: user-service
image: sk124590/user-service:2.1
ports:
- containerPort: 8084
envFrom:
- configMapRef:
    name: gift-config
- secretRef:
    name: gift-secret
resources:
  requests:
    cpu: "120m"  # [FIXED] 절약 등급 적용: 150m x 0.8
    memory: "410Mi"  # [FIXED] 절약 등급 적용: 512Mi x 0.8
  limits:
    cpu: "800m"  # [FIXED] 절약 등급 적용: 1000m x 0.8
    memory: "820Mi"  # [FIXED] 절약 등급 적용: 1Gi x 0.8
```

그림 1 모니터링 상세 이력 화면 일부



Output

DOCUMENTS 20

조직 내부 정책 - 리소스 할당 기준 - 모든 컨테이너는 리... text/plain 조직내부정책-리소스 할당 기준.md

성능	레플리카를 증설해도 CPU 사용량이 계속 높게 유지될 때	**현재 `requests`, `limits` 값 × 1.5**
표준	서비스의 기본 상태	현재 설정값 유지
절약	리소스가 필요 이상 낭비될 때 때	현재 `requests`, `limits` 값 × 0.8
절전	트래픽이 거의 없어 리소스 낭비가 심하다고 판단될 때	현재 `requests`, `limits` 값 × 0.5

content_type text/plain

file_name 조직내부정책-리소스 할당 기준.md

배포 환경 마이크로서비스 기본 정보... text/plain 배포 환경 마이크로서비스 기본 정보 (user-service).md

배포 환경 마이크로서비스 기본 정보 ... text/plain 배포 환경 마이크로서비스 기본 정보(api-gateway).md

배포 환경 마이크로서비스 기본 ... text/plain 배포 환경 마이크로서비스 기본 정보 (product-service).md

배포 환경 마이크로서비스 기본 정... text/plain 배포 환경 마이크로서비스 기본 정보 (order-service).md

그림 2 개선안 도출에 사용된 조직 내부 데이터 및 유사도 순위

05.

사례 연구

05.

사례 연구 대상: '선물하기 서비스'

상품 목록 조회

그림 1 전체 상품 목록 조회 화면

위시 리스트 추가

그림 2 위시 리스트 화면

상품 주문

그림 3 선물을 위한 주문 화면

05.

사례 연구 선정 이유

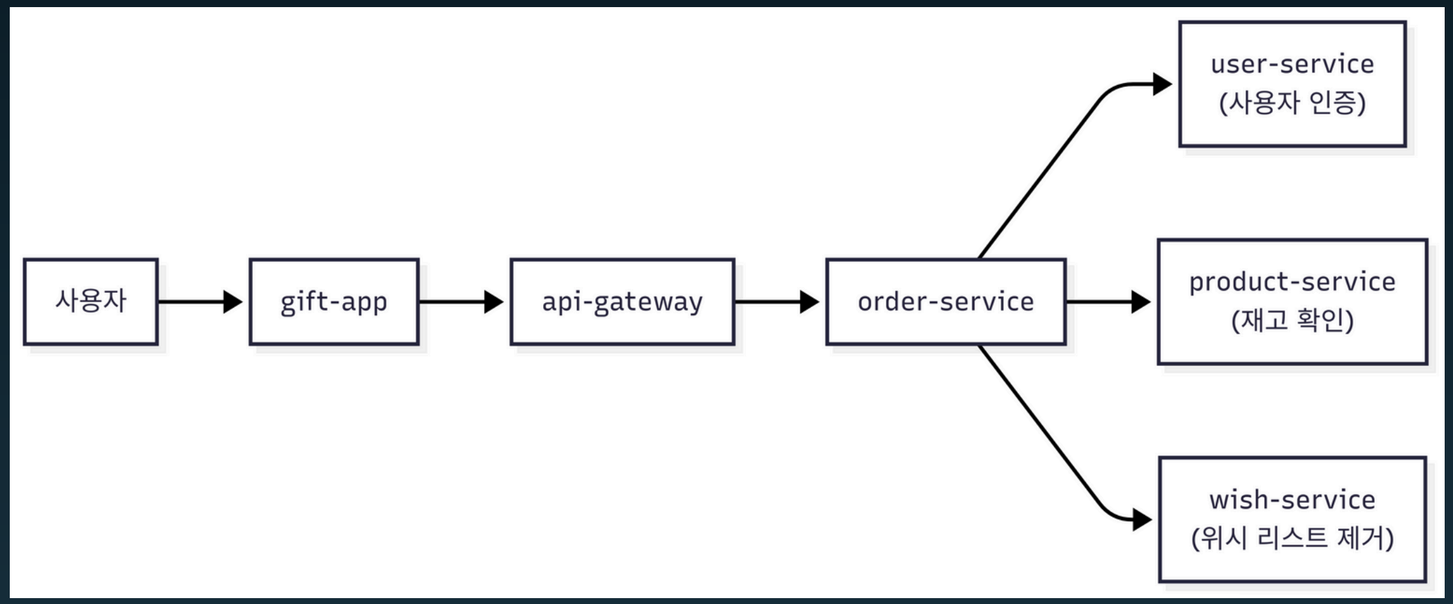


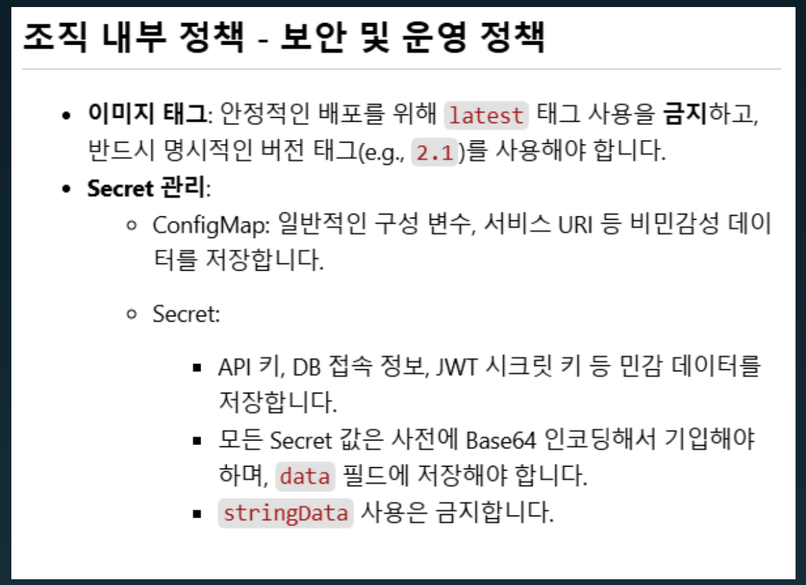
그림 1 상품 주문 흐름도

1. 서비스 의존성을 반영한 배포 명세 생성 검증

- **하나의 기능을 수행하기 위해 여러 마이크로서비스가 의존관계를 가짐**

05.

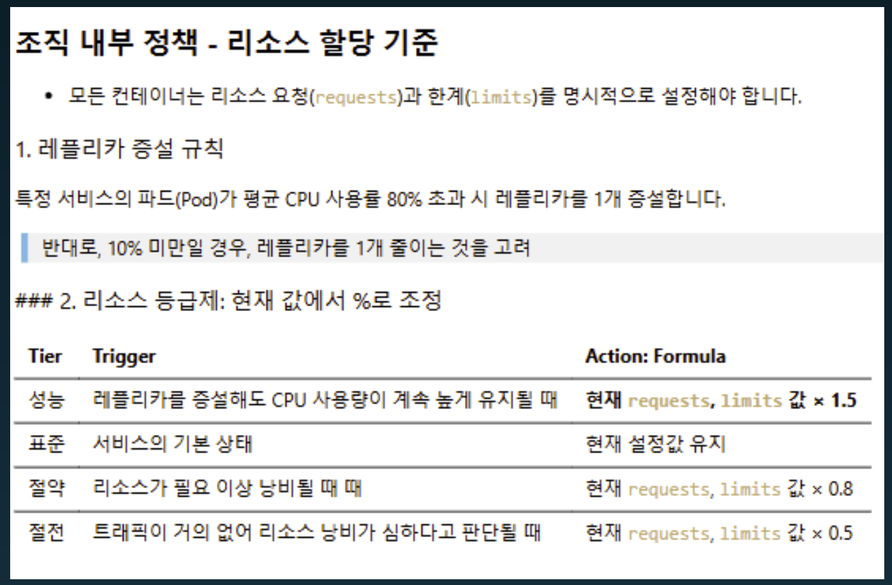
사례 연구 선정 이유



조직 내부 정책 - 보안 및 운영 정책

- **이미지 태그**: 안정적인 배포를 위해 `latest` 태그 사용을 **금지**하고, 반드시 명시적인 버전 태그(e.g., `2.1`)를 사용해야 합니다.
- **Secret 관리**:
  - ConfigMap: 일반적인 구성 변수, 서비스 URI 등 비민감성 데이터를 저장합니다.
  - Secret:
    - API 키, DB 접속 정보, JWT 시크릿 키 등 민감 데이터를 저장합니다.
    - 모든 Secret 값은 사전에 Base64 인코딩해서 기입해야 하며, `data` 필드에 저장해야 합니다.
    - `stringData` 사용은 금지합니다.

그림 1 보안 및 운영 정책



조직 내부 정책 - 리소스 할당 기준

- 모든 컨테이너는 리소스 요청(`requests`)과 한계(`limits`)를 명시적으로 설정해야 합니다.

1. 레플리카 증설 규칙

특정 서비스의 파드(Pod)가 평균 CPU 사용률 80% 초과 시 레플리카를 1개 증설합니다.

> 반대로, 10% 미만일 경우, 레플리카를 1개 줄이는 것을 고려

2. 리소스 등급제: 현재 값에서 %로 조정

Tier	Trigger	Action: Formula
성능	레플리카를 증설해도 CPU 사용량이 계속 높게 유지될 때	**현재 `requests`, `limits` 값 × 1.5**
표준	서비스의 기본 상태	현재 설정값 유지
절약	리소스가 필요 이상 낭비될 때 때	현재 `requests`, `limits` 값 × 0.8
절전	트래픽이 거의 없어 리소스 낭비가 심하다고 판단될 때	현재 `requests`, `limits` 값 × 0.5

그림 2 리소스 할당 기준

2. 조직 운영 정책 준수 여부 확인

- 조직의 구체적인 운영 정책을 배포 명세에 올바르게 적용하는지 검증 가능

05.
'선물하기 서비스' 조직 내부 데이터 일부

MSA 애플리케이션 정의

배포 환경 MSA 애플리케이션 정의 - 1

- **서비스 설명**
  - **서비스 이름**: 선물하기 서비스 (Gift Service)
  - **구성 마이크로서비스 목록**:
    - `api-gateway`: 모든 MSA 요청에 대한 인증 및 라우팅을 담당하는 관문 서비스입니다.
    - `user-service`: 사용자 회원가입, 로그인, 정보 관리를 담당합니다.
    - `product-service`: 상품 정보, 재고, 옵션을 관리합니다.
    - `wish-service`: 사용자의 위시리스트를 관리합니다.
    - `order-service`: 상품 주문 및 카카오톡 메시지 발송을 처리합니다.
    - `gift-app`: 사용자를 위한 웹 UI를 제공하는 BFF(Backend for Frontend) 서비스입니다.

그림 1 MSA 애플리케이션 정의 - 1

마이크로서비스 기본 정보

배포 환경 마이크로서비스 기본 정보 (order-service)

- **컨테이너 이미지 및 버전 정보**: `sk124590/order-service:2.1`
- **마이크로서비스 포트**: `8083`
- **환경 변수**:
  - **ConfigMap (gift-config)**
    - `SERVICE_PRODUCT_URI`
    - `SERVICE_WISH_URI`
    - `SERVICE_USER_URI`
    - `FRONT_DOMAIN`
  - **Secret (gift-secret)**
    - `KAKAO_CLIENT_ID`
    - `KAKAO_CLIENT_SECRET`
- **리소스 설정**:
  - requests: `cpu: "100m", memory: "512Mi"`
  - limits: `cpu: "700m", memory: "1Gi"`
- **복제 수 (Replicas)**: 1

그림 2 마이크로서비스 기본 정보(order-service)

조직 내부 정책

조직 내부 정책 - 보안 및 운영 정책

- **이미지 태그**: 안정적인 배포를 위해 `latest` 태그 사용을 **금지**하고, 반드시 명시적인 버전 태그(e.g., `2.1`)를 사용해야 합니다.
- **Secret 관리**:
  - ConfigMap: 일반적인 구성 변수, 서비스 URI 등 비민감성 데이터를 저장합니다.
  - Secret:
    - API 키, DB 접속 정보, JWT 시크릿 키 등 민감 데이터를 저장합니다.
    - 모든 Secret 값은 사전에 Base64 인코딩해서 기입해야 하며, `data` 필드에 저장해야 합니다.
    - `stringData` 사용은 금지합니다.

그림 3 조직 내부 정책(보안 및 운영)

05.
생성된 배포 명세 파일 일부

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: gift-secret
data: # From 조직 내부 정책
  JWT_SECRET: WW4ya2ppYmRkRkFXdG5QSjJBRmxMOFdYbW9oSk1DdmlnUWdnYUV5cGE1RT0=
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: order-service
spec:
...
  template:
  ...
    spec:
      containers:
      - name: order-service
        image: sk124590/order-service:2.1
        ports:
        - containerPort: 8083
        env:
        - name: SERVICE_PRODUCT_URI
          valueFrom:
            configMapKeyRef:
              name: gift-config
              key: SERVICE_PRODUCT_URI
        - name: SERVICE_WISH_URI
          valueFrom:
            configMapKeyRef:
              name: gift-config
              key: SERVICE_WISH_URI
...
        resources:
          requests:
            cpu: "150m"
            memory: "512Mi"
          limits:
            cpu: "1000m"
            memory: "1Gi"
```

1

2

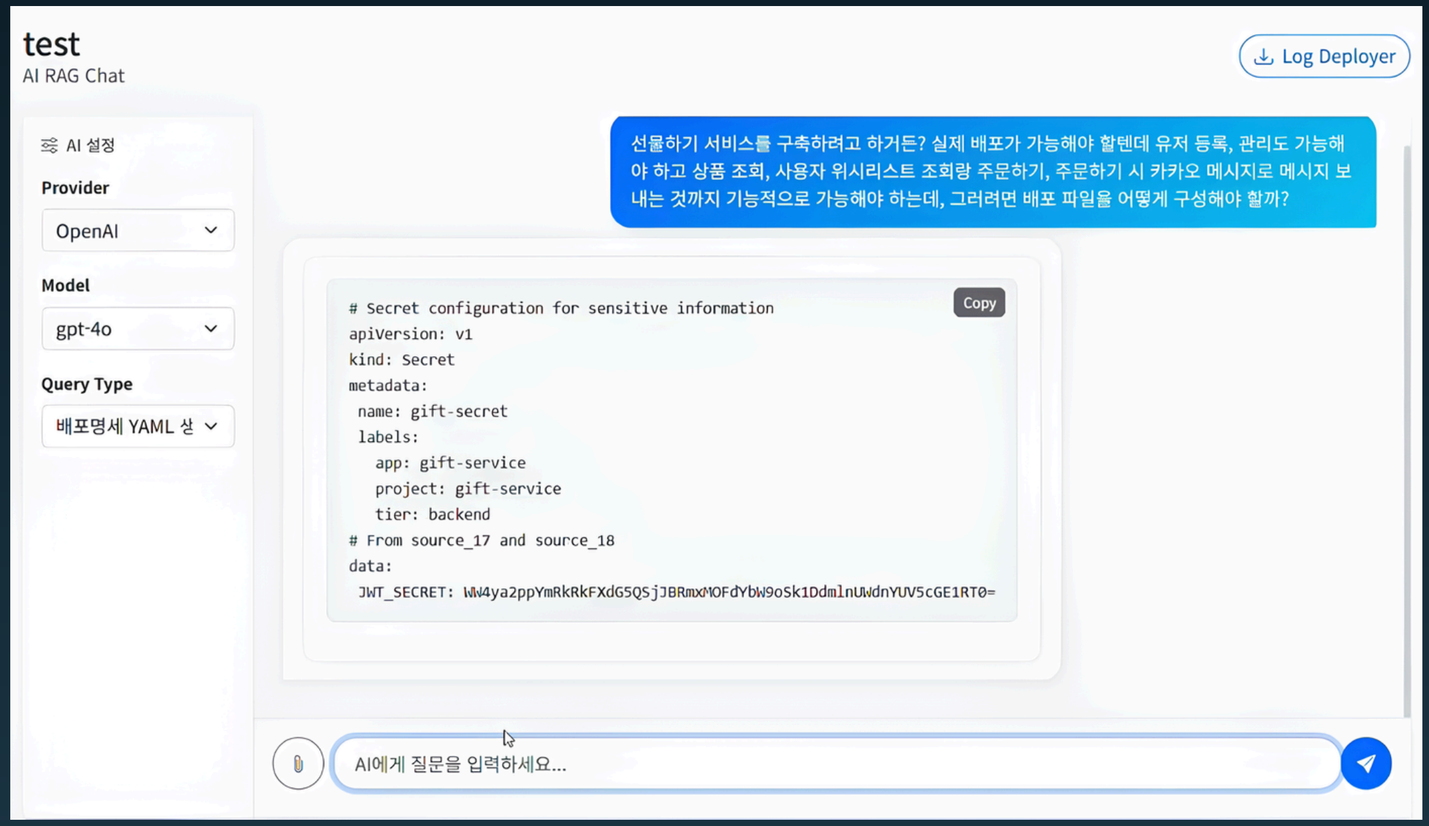
3

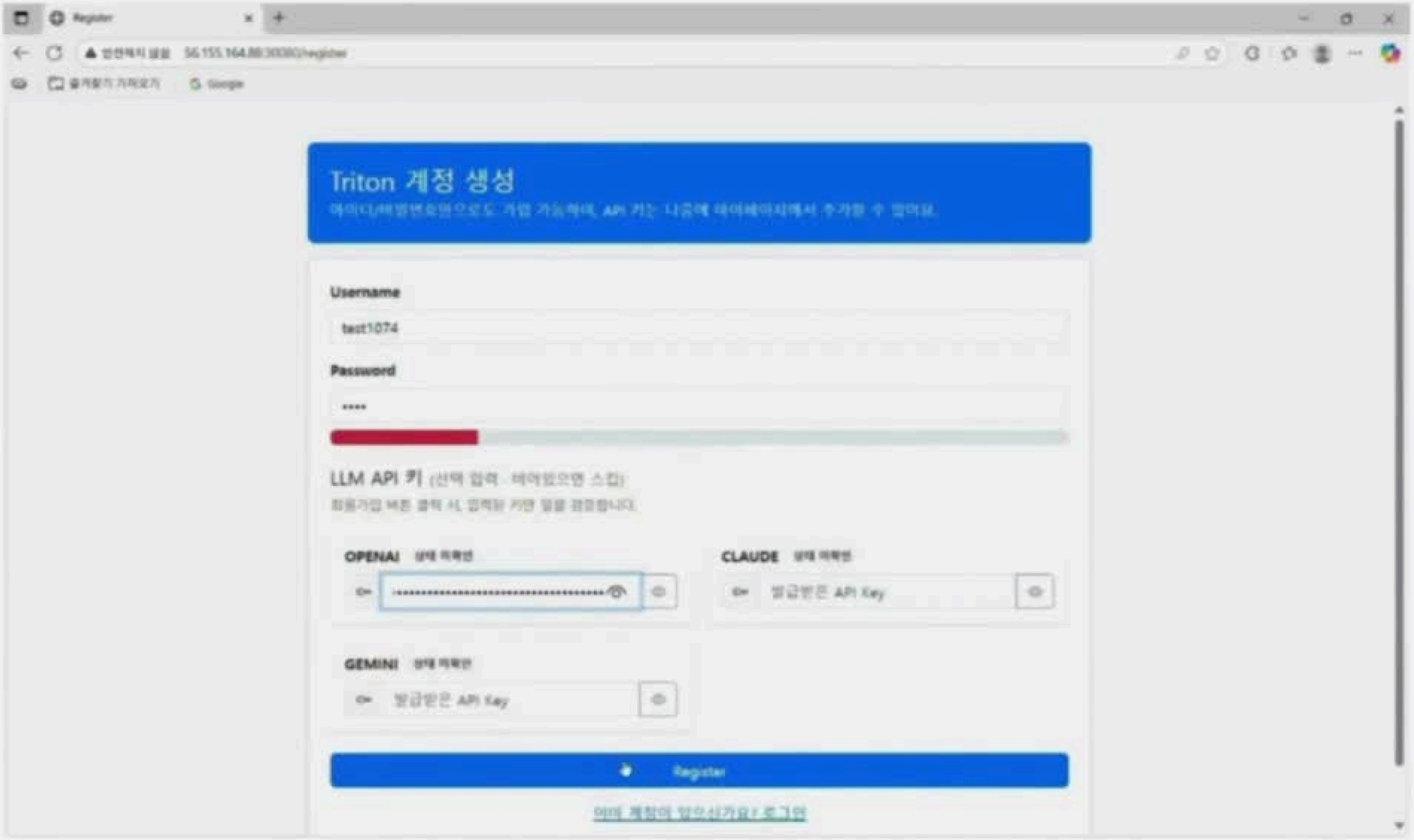
1 - **보안 정책**에 따라 Secret 리소스에서 data 필드 사용

2 - **마이크로서비스 정보**에 따라 정확한 포트번호, 컨테이너 이미지, 의존성(URI 환경변수)을 설정

3 - **리소스 정책**에 따라 requests, limits 값 반영

05.
시연 영상


test
AI RAG Chat
Log Deployer
AI 설정
Provider
OpenAI
Model
gpt-4o
Query Type
배포명세 YAML 생
선물하기 서비스를 구축하려고 하거든? 실제 배포가 가능해야 할텐데 유저 등록, 관리도 가능해야 하고 상품 조회, 사용자 위시리스트 조회랑 주문하기, 주문하기 시 카카오 메시지로 메시지 보내는 것까지 기능적으로 가능해야 하는데, 그러려면 배포 파일을 어떻게 구성해야 할까?
Copy
Secret configuration for sensitive information
apiVersion: v1
kind: Secret
metadata:
name: gift-secret
labels:
app: gift-service
project: gift-service
tier: backend
From source_17 and source_18
data:
JWT_SECRET: WW4ya2ppYmRkRkFXdG5QSjJBRmxMOFdYbW9oSk1DdmlnUWdnYUV5cGE1RT0=
AI에게 질문을 입력하세요...



API 키는 회원가입 버튼 클릭 시 검증되며,
검증에 실패하면 회원가입이 진행되지 않습니다.

06.

연구 결과

06.
연구 결과

내부 문서 반영에 따른 배포 성공률 증가

P1 → P2
상세 질의를 통해 배포 명세의 구조 생성 확인

P2 → P3
RAG를 적용해 배포 명세 필드에 정확한 값 주입 확인

P3 → P4
배포 명세 이력 데이터에 기반하여 프롬프트 템플릿에
명시적 규칙을 추가함으로써 특수 오류 케이스 극복

최종 프롬프트를 활용하여 배포 성공률 **85%**(17/20)를 달성하였으며, 이는 DevOps 성능 평가 지표인 **DORA의 Elite 등급 기준을 충족**하는 결과임

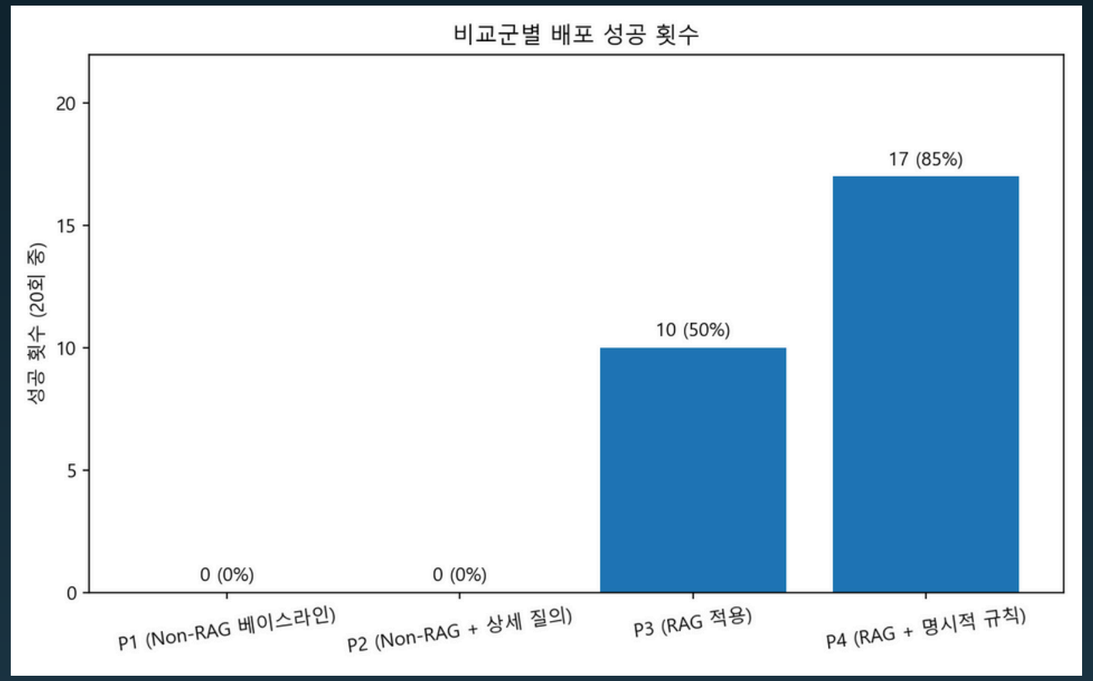


그림 2 비교군별 배포 성공 횟수

06.
연구 결과

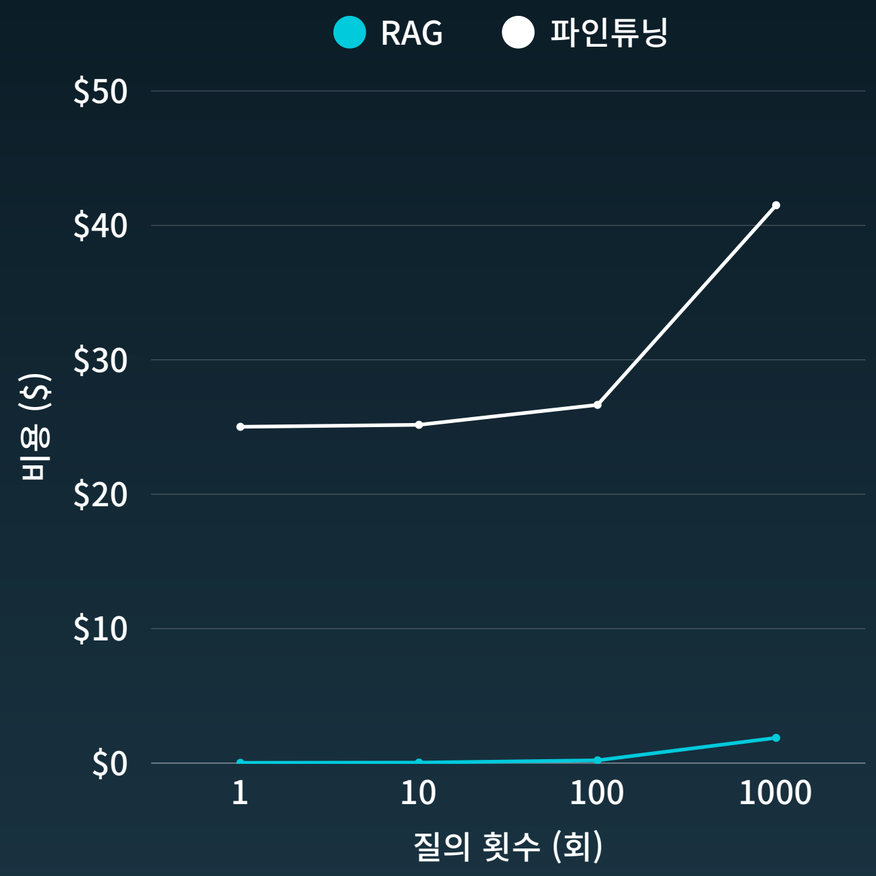
LLM 기반 접근 방법의 비용 절감

RAG는 문서를 최초로 1번만 임베딩하고 유사도 검색을 통해 필요한 문맥만 사용함

이로 인해 모델 호출당 비용이 감소하고, 대규모 문서 처리 시에도 효율적인 비용 구조를 유지 가능

동일한 데이터 활용 환경에서 전체 운영 비용을 크게 절감 가능

기술 / 비용 항목	RAG	파인 튜닝
초기 비용 (1M 토큰 당)	임베딩: $0.02 text-embedding-3-small	모델 학습: $25
입력 비용 (1M 토큰 당)	$0.15 (GPT-4o)	$3.75 (GPT-4o Fine Tuning)
출력 비용 (1M 토큰 당)	$0.60 (GPT-4o)	$15 (GPT-4o Fine Tuning)



07.

연구 결론

07.

RAG 기반 배포 명세 자동화

주요 연구 성과

RAG 기반 배포 명세 자동화로
담당자 **지식 의존도** 및 수작업 **오류** 감소

향후 기대 효과

배포 **복잡성 완화** 및 **운영 효율성** 극대화

정확성·신뢰성 강화된 운영 지원 체계 확립

07.

로그·메트릭 기반 동적 개선

주요 연구 성과

로그·메트릭 기반 **동적 개선** 기능으로 **운영 안정성** 및 **리소스 효율** 향상

향후 기대 효과

실시간 데이터 반영 및
개선 프로세스 자동화로 **비용 절감** 기대

로그 **관찰 및 수집**에 머물던 기존의 **수동적인** 시스템의 한계를 극복

감사합니다