



# 생성형 AI 서비스, 게이트웨이로 쉽게 시작하기

## 우아한형제들의 AI API 게이트웨이 개발

- 생성형 AI 기술의 효율적 활용을 위한 게이트웨이 개발 배경
- 다양한 AI 서비스를 손쉽게 통합하는 API 게이트웨이 아키텍처
- 자격증명, 템플릿, 프록시, 파이프라인 등 주요 기능 소개

# 개발 배경: 생성형 AI의 확산과 변화

생성형 AI 도구의 확산은 크게 두 가지 측면에서 큰 변화를 가져옴

기술 민주화: AI/ML 전문 지식이 없는 일반 사용자도 간단한 프롬프트

입력만으로 고품질 콘텐츠 생성 가능

비즈니스 가속화: 자체 모델 개발이나 파인튜닝 없이도 빠르게 AI 기반

서비스 개발 및 출시 가능

우아한형제들의 AI 도입 현황:

- 회사 전반에 걸쳐 다양한 업무와 서비스 개발에 생성형 AI 활용 중
- 이를 사용하는 구성원과 적용 서비스 수 꾸준히 증가 신규 서비스 기획 단계부터 생성형 AI 활용 적극 검토

## 대표 활용 사례

### 메뉴 이미지 개선 기능

- 저품질 메뉴 이미지 품질 향상
- 과도하게 확대 촬영된 이미지의 배경 영역 확장(아웃페인팅)
- Google VertexAI Imagen 활용
- AI 기술이 서비스 품질 향상에 직접 기여하는 사례



이러한 AI 활용 확대에 따라 생성형 AI 기술을 더욱 효율적으로 개발할 수 있는 환경 구축 필요성 대두

# 문제 정의: 생성형 AI 활용의 현실적 과제

**인터뷰 기반 요구사항 분석:** 생성형 AI를 적극 활용하는 구성원들과의 인터뷰를 통해 어려움과 개선 필요사항 도출

## 6대 핵심 주제:

- **API Gateway:** 생성형 AI 시스템의 핵심 인프라 구성 요소, 반복 개발 작업 최소화, 시스템 안정성, 보안성
- **Experimental Feedback:** 생성 결과 품질 개선을 위한 사용자 피드백 수집, 반영
- **LLM Serving:** 대규모 언어 모델의 효율적 운영 및 관리
- **Prompt Experiment:** 효과적인 프롬프트 개발과 최적화를 위한 실험 환경
- **Hybrid Search:** 어휘 검색과 의미적 검색을 결합한 고급 검색 시스템  
<https://cloud.google.com/vertex-ai/docs/vector-search/about-hybrid-search?hl=ko>
- **RAG Pipeline:** 외부 지식을 활용한 응답 품질 향상 파이프라인

## API 게이트웨이 개발 우선 선정 배경:

다양한 생성형 AI 서비스(AWS Bedrock, Azure OpenAI, GCP Imagen) 활용 ↑  
프롬프트와 호출 로직의 파편화로 인한 관리 어려움

자격증명과 프롬프트의 중앙화된 관리 필요성 증대

## 직접 개발의 필요성 (cf. Kong, Dify.ai)

맞춤형 요구사항 충족: 특정 AI 워크로드에 대한 빠른 지원 가능

유연성 확보: 빠르게 변화하는 AI 기술에 신속한 대응

비용 및 보안: 상용 솔루션 대비 장기적 비용 절감 및 보안 강화

통합 용이성: 기존 인프라와 워크플로우와 원활한 통합

# 해결과제 : AI API 게이트웨이의 필요성

## 해결해야 할 기술적 과제



### 자격증명 관리

다양한 API별 인증정보  
보안과 갱신 문제



### 중복 개발

동일 기능의 반복 구현  
로그/민감정보 처리



### 배포 비효율

설정값 변경에도 코드  
수정 및 재배포



### 프롬프트 관리

최적 프롬프트 공유  
버전 관리의 어려움

## 자격증명 관리 복잡성

AWS 시크릿 매니저 통해 관리 제공하고 있지만

- 신규 서비스 개발시, 신규 보안 암호 생성 요청 필요
- 수정 후, 즉각적인 익스터널 시크릿 갱신 위한 배포
- 서비스 단위 보안 암호 수 지속 증가
- 자격 증명 히스토리 관리, 추적 어려움, 정책, 시스템

• 애저 오픈AI : AZURE\_OPENAI\_ENDPOINT, AUZRE\_OPENAI\_API\_KEY

• GCP 버텍스AI : credential.json

• AWS 베드록 : IAM Role ARN (+ Security Token Service(STS))

• 네이버 클라우드 플랫폼 : invoke\_url, secret\_key

## 중복 개발 필수 요소

생성형 AI 활용 개발은 외부 API를 호출하는 것으로 시작하지만

- 개인 식별 정보 유출 차단 및 이를 활용한 결과 생성 차단 필요. 따라서 민감 정보 요청 거부나 해당 정보를 마스킹 처리 후 외부 생성형 API에 전달

- 입출력 데이터 활용을 위한 로깅 저장소 생성/관리 필요, 표준 입출력 정의 및 적재 로직 구현, 프롬프트와 설정값 등 파라미터들로 로깅 필요

- API 호출의 제한(Quota/Call Rate) 발생, 예외 상황과의 구분을 위해 예외 처리와 폴백 로직 구현 필요, 대용량 트래픽이 예상되는 경우, 생성형 AI API 리소스를 여러 개 생성하고 호출 시점에 어떤 리소스를 요청 할지에 대한 로직 구현

## 비효율적인 배포

게이트웨이 개발 전에는 자격증명, 프롬프트, 모델 배포명, API 버전 등의 속성값, 상세 파라미터들을 코드 또는 리소스 파일로 관리 외부 API 호출에 필요한 값들이 수시로 변하기때문에 프롬프트와 같은 리소스 파일 수정을 위해 재배포 이 루어지는 것은 비효율적

게이트웨이에서는 동적 라우팅과 내부 리소스 관리 기능 구현

→ API 게이트웨이를 통해 개선하였음

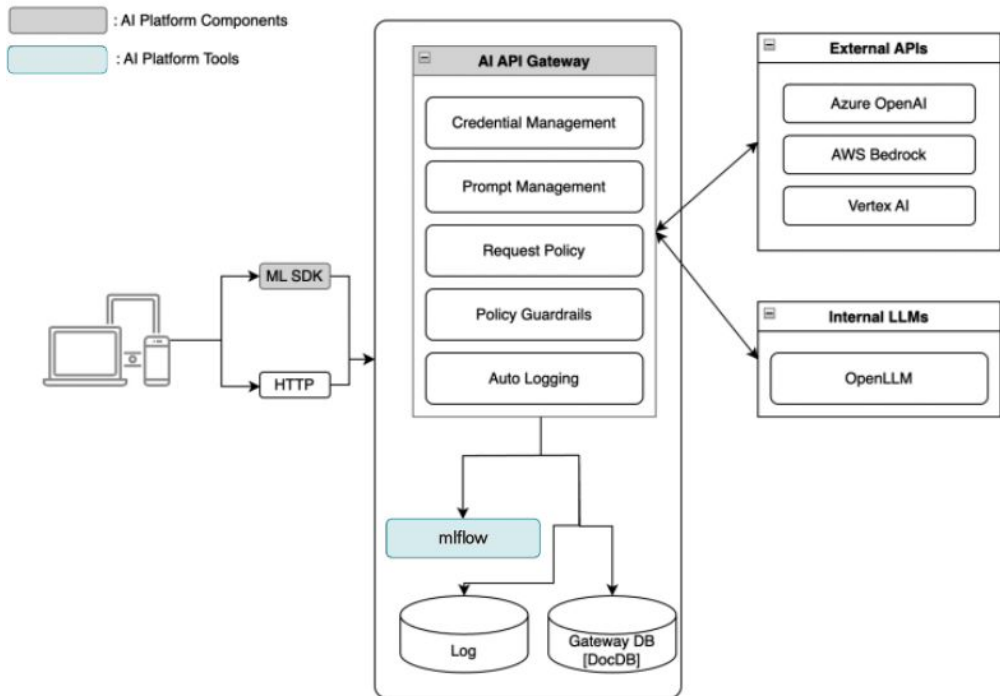
# AI API 게이트웨이 아키텍처

**게이트웨이 개요:** 생성형 AI API 호출을 중앙 집중화하여 자격증명 관리, 프롬프트 관리, 로깅, 비용 모니터링 등 공통 기능을 통합 제공

## 핵심 기능:

- 멀티 클라우드 자격증명 통합 관리
- 템플릿 기반 프롬프트 버전 관리
- 프록시/파이프라인을 통한 호출 간소화
- LangChain 호환성 지원

• 게이트웨이 핵심 기능과 호출 흐름 •



## 호출 방식

**HTTP 호출:** 간단한 프록시/파이프라인, 외부 서버 연동에 적합

**ML SDK 호출:** 복잡한 전처리/후처리, LangChain 연동에 적합

**다중 자격증명:** 트래픽 분산 및 Call Rate Limit 회피 지원

## 지원 클라우드 서비스

 <b>AWS</b> Bedrock	 <b>Azure</b> OpenAI	 <b>Google</b> VertexAI(Imagen)	 <b>Naver</b> Cloud Platform
--	---	--	---

## 지원 기능

프로젝트

자격증명 관리

템플릿 관리

프록시 생성/관리

파이프라인 생성/관리

랜체인 호환성 지원

공용 대시보드

# 프로젝트 및 자격증명 관리

## 프로젝트 기반 API Key 관리

- 프로젝트 단위 API Key 발급
  - 사용자 → 프로젝트 생성 요청 → API Key 발급
- API Gateway 접근 제어
  - 모든 요청에 대해 API Key 유효성 검증
- 권한 제한 & 보안 강화
  - 무분별한 호출 및 리소스 수정/삭제 방지
- 환경별 독립 운영
  - 베타와 운영 환경은 프로젝트 및 API Key 분리 관리

## 자격증명 관리 프로세스

자격증명 관리 기능이란?

- 자격증명 값: AI API(예: AWS Bedrock, Azure OpenAI, GCP Vertex AI 등)를 호출할 때 필요한 Access Key, Secret Key, Token 같은 인증 정보 → 기존에는 AWS Secrets Manager에 별도 저장·관리


게이트웨이가 해주는 일

1. 게이트웨이에 자격증명 등록
2. 통합 관리
  - 게이트웨이가 API 호출 시 자동으로 해당 인증 값을 꺼내 사용.
  - 필요하면 게이트웨이에서 수정·삭제하면 즉시 서비스에 반영.
3. 멀티 클라우드 지원
  - AWS Bedrock, Azure OpenAI, GCP Vertex AI, NCP 등 여러 클라우드 AI 서비스 호출 가능.
  - AWS Bedrock은 STS(Security Token Service)라는 내부 제어 방식을 활용해 호출을 관리. 필요할 때만 짧은 유효기간의 임시 키를 발급

# 프로젝트 및 자격증명 관리

## 자격증명 등록하는 과정인 듯...

• 게이트웨이에서 제공하는 자격증명 관리 기능 •

**access\_info**  **required** NCPAccessInfo (object) or AzureOpenAIAccessInfo (object) or AWSAccessInfo (object) or GCPAccessInfo (object)  
(Access Info)  
자격증명 정보. service 별로 포맷이 상이함에 주의

Any of

NCPAccessInfo

AzureOpenAIAccessInfo

AWSAccessInfo

GCPAccessInfo

**endpoint** **required**

string (Endpoint)  
Azure OpenAI endpoint

**api\_key** **required**

Api Key (string) or Api Key (string) (Api Key)  
Azure OpenAI API key

# 템플릿 관리

- 생성형 AI 결과의 품질과 직결되는 프롬프트의 유연한 변경과 체계적인 관리
- 게이트웨이에 등록된 템플릿을 공개, 구성원들의 노하우 공유
- 템플릿 생성은 **plain text**와 **message blocks** 포맷을 모두 지원해 간단한 템플릿과 복잡한 템플릿의 등록과 사용을 모두 지원

동적 템플릿에 주입되는 값은 **(key)** 형태로  
지정하며, 템플릿을 통한 요청 시 **(key)**에  
해당하는 값을 전달해 최종 프롬프트를 생성

버전 관리

서비스 호출 시 요청

```
// message block 형식의 템플릿
// 정적 프롬프트 템플릿
POST /v1/templates
x-project-name: {{project_name}}
x-api-key: {{api_key}}

{
  "name": "example-message-blocks",
  "template": [
    {
      "role": "system",
      "content": "You are a helpful assistant."
    },
    {
      "role": "user",
      "content": "2024년 올림픽은 어디서 개최했어?"
    },
    {
      "role": "assistant",
      "content": "파이썬입니다."
    },
    {
      "role": "user",
      "content": "가장 많은 메달을 딴 국가는 어디야?"
    }
  ]
}

// plain text 형식의 템플릿(message block 형식으로 변환되어 저장)
// 동적 프롬프트 템플릿
POST /v1/templates
x-project-name: {{project_name}}
x-api-key: {{api_key}}

{
  "name": "example-plain-text",
  "template": "({style} 스타일로 {user_input}에 대해 설명해줘)"
}
```

```
// 템플릿 테스트 호출
POST /v1/templates/example-plain-text
x-project-name: {{project_name}}
x-api-key: {{api_key}}

{
  "style": "웃긴",
  "user_input": "파이썬"
}

// 템플릿 테스트 응답
{
  "template": {
    "name": "example-plain-text",
    "template": [
      {
        "role": "user",
        "content": "({style} 스타일로 {user_input}에 대해 설명해줘)"
      }
    ],
    "version": "1"
  },
  "queries": {
    "style": "웃긴",
    "user_input": "파이썬"
  },
  "prompt": [
    {
      "role": "user",
      "content": "웃긴 스타일로 파이썬에 대해 설명해줘"
    }
  ]
}
```

```
// 템플릿 업데이트 후의 템플릿 정보
{
  "name": "example-plain-text",
  "template": [
    {
      "role": "user",
      "content": "({style} 말투로 {user_input}에 대해 알려줘)"
    }
  ],
  "version": "2",
  "versions": {
    "1": [
      {
        "role": "user",
        "content": "({style} 스타일로 {user_input}에 대해 알려줘)"
      }
    ],
    "2": [
      {
        "role": "user",
        "content": "({style} 말투로 {user_input}에 대해 알려줘)"
      }
    ]
  }
}
```

```
POST /v1/services/azure_openai/chat_completions
x-project-name: {{project_name}}
x-api-key: {{api_key}}

{
  "deployment_name": "gpt-4o",
  "api_version": "2024-02-01",
  "template": {
    "name": "example-plain-text",
    "queries": {
      "style": "웃긴",
      "user_input": "파이썬"
    }
  },
  "params": {
    "temperature": 0.7,
    "max_tokens": 50
  }
}
```



# 프록시 생성/관리

## 생성형 AI API 적용을 위한 핵심 요소

- 자격증명 (access credentials)
- 프롬프트 (prompt)
- 파라미터 (parameters)

## 프록시 기능의 목적과 이점

1. 요청 양식 간소화
  - 복잡한 API 요청을 단일 프록시 호출로 단순화
  - 예: 클라이언트는 프록시 이름만 지정하면 됨
2. 비개발자 (데이터 사이언티스트)도 수정 가능
  - 프롬프트나 파라미터를 엔지니어 개입 없이 조정 가능
3. 다중 자격증명 관리
  - 여러 자격증명 등록 가능
    - 트래픽 쉼/속도 제한 등의 이슈 완화
    - 게이트웨이가 내부적으로 적절한 자격증명 자동 선택

```
// 프록시 생성 요청
POST /v1/proxy
x-project-name: {{project_name}}
x-api-key: {{api_key}}

{
  "name": "auzre_opanai_proxy",
  "credential_names": [
    "azure_openai1",
    "azure_openai2",
    "azure_openai3"
  ],
  "policy": "random",
  "service": "auzre_openai",
  "template": "tech-blog-example",
  "options": {
    "deployment_name": "gpt-4o",
    "api_version": "2024-02-15-preview",
    "params": {
      "temperature": 0.7,
      "max_tokens": 50
    }
  }
}
```

```
// 프록시 호출
POST /v1/proxy/auzre_opanai_proxy/chat_completion
x-project-name: {{project_name}}
x-api-key: {{api_key}}

{
  "queries": {
    "style": "웃긴",
    "user_input": "파이썬"
  }
}
```

```
// 버텍스AI 이미지생성에 대한 프록시 생성
POST /v1/proxy
x-project-name: {{project_name}}
x-api-key: {{api_key}}

{
  "name": "vertex-ai-imagegeneration-proxy",
  "credential_names": [
    "gcp_cred"
  ],
  "service": "gcp_vertex_ai",
  "policy": "random",
  "options": {
    "model_name": "imagen-3.0-generate-001",
    "params": {
      "negative_prompt": "steam, close up",
      "number_of_images": 1,
      "aspect_ratio": "1:1",
      "language": "en",
      "safety_filter_level": "block_few",
      "person_generation": "dont_allow"
    }
  }
}
```

# 파이프라인 생성 관리

## 단일 호출 한계

- 단순 API 한 번 호출로는 복잡한 서비스 구성 어려움

## 복합 시나리오 유형

- 동일 API 반복 호출 → 프롬프트 고도화로 해결 가능
- 다른 API 연속 호출 → 프롬프트만으로는 해결 불가

## 프록시 기능 한계

- 서버 없이 구현할 경우 연속 호출 시 한계 존재
- 

## → 파이프라인 기능 제공

- 게이트웨이 자체에서 간단한 순차 호출 시나리오 지원
- 예: 한국어 → 영어 번역 → 이미지 생성

0번째 인덱스(첫 번째) 프록시의 결과에서 특정 필드 값(0.response.choices.0.message.content) 을 프롬프트로 해 이미지를 생성하는 예제

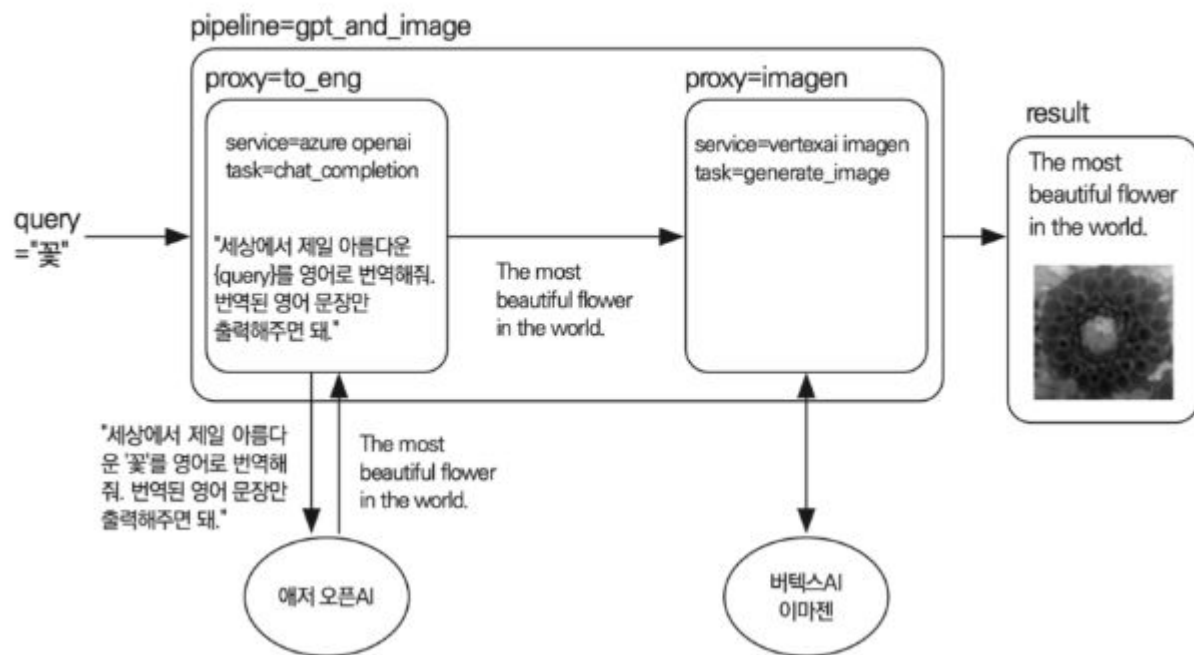
```
POST /v1/pipelines
x-project-name: {{project_name}}
x-api-key: {{api_key}}
{
  "name": "gpt_and_imagen",
  "pipeline": [
    {
      "proxy": "to_eng",
      "task": "chat_completion"
    },
    {
      "proxy": "imagen",
      "task": "generate_image",
      "inputs": {
        "prompt": "0.response.choices.0.message.content"
      }
    }
  ]
}
```

## 요청 양식

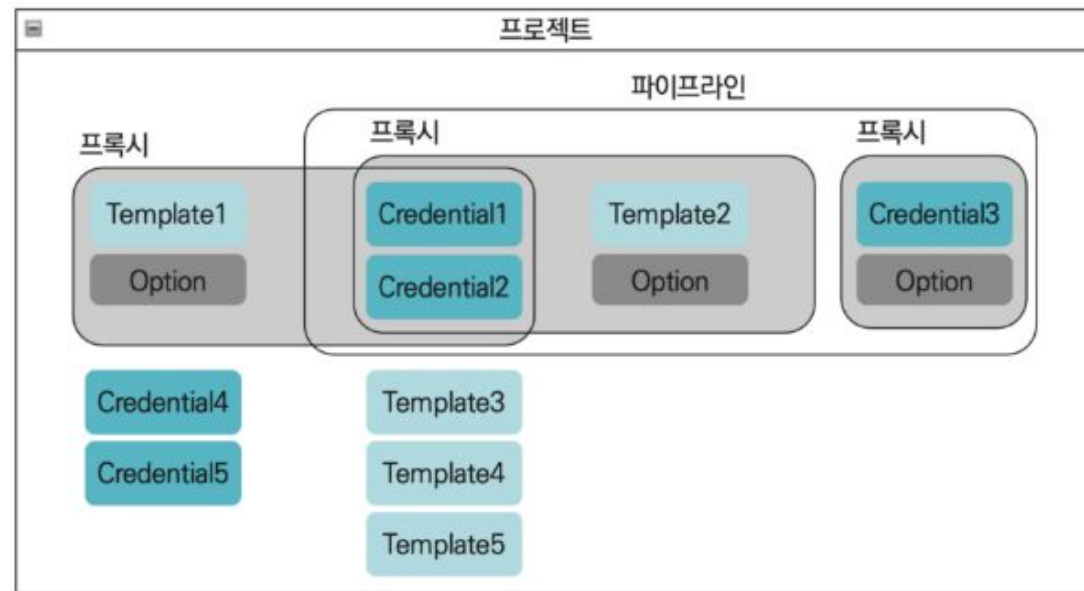
```
POST /v1/pipelines/gpt_and_imagen/run
x-project-name: {{project_name}}
x-api-key: {{api_key}}
{
  "queries": {
    "query": "꽃"
  },
  "params": {
  }
}
```

# 파이프라인 생성 관리

• 게이트웨이 내부 흐름도 •



• 프로젝트 리소스 구조 •



# 랭체인 호환성 지원

## 1. 배경

- 생성형 AI API 단일 호출을 넘어서 **RAG 등 고급 방법론 활용** 증가
- **LangChain** 사용률 증가, 이에 맞는 게이트웨이 연동 필요

## 2. 비효율 발생 사례

- 게이트웨이 결과를 LangChain에 수동 이식 → **중복 개발**
- 프롬프트/파라미터 설정을 별도 코드로 재작성 필요

## 3. 해결책: ML SDK 제공

- **AiApiGatewayClient**로 LangChain과 **pipe 연산자 (|)** 연동 가능
- 프록시 설정 시 서비스 제공자 교체 (**Azure ↔ AWS 등**) 도 간편
- **비동기 지원** → 클라이언트 서버에서도 사용 적합

```
from woowa_ml_sdk.client.ai_api_gateway import
from woowa_ml_sdk.client.ai_api_gateway import AiApiGatewayClient
from woowa_ml_sdk.util.config import Config

config = Config("beta", "project_name")

llm = AiApiGatewayClient(
    config=config,
    api_key=os.getenv("AI_API_GATEWAY_API_KEY"),
    service="azure_openai",
    credential="azure_openai_cred", # 생략 시 게이트웨이에서 임의 선택
    template="...",
    params={...}
)

prompt = ChatPromptTemplate.from_messages(
    [("system", "you are a bot"), ("human", "{input}")]
)

chain = prompt | llm

chain.invoke("hello")
```

# 공용 대시보드

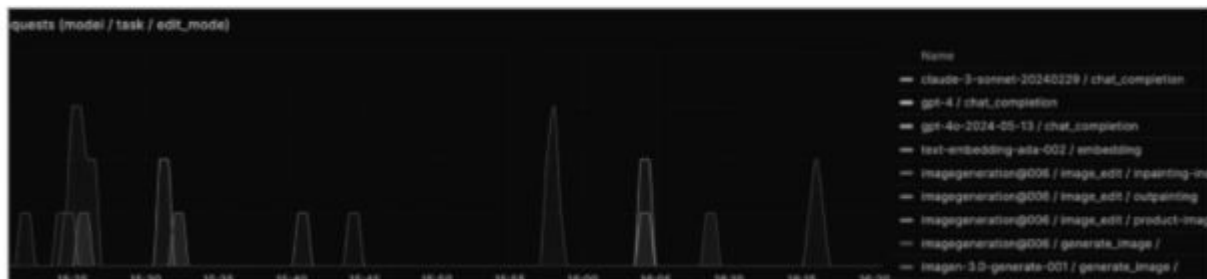
## 배경

### - 비용 추적 관리 필요

- **NLP vs Vision:** 도메인별 과금 방식 상이
- **모델/제공자**에 따라 단가 차이 큼

게이트웨이에서는 프로젝트 또는 모델 단위로 비용과 비례하는 메트릭을 한눈에 볼 수 있는 대시보드를 제공

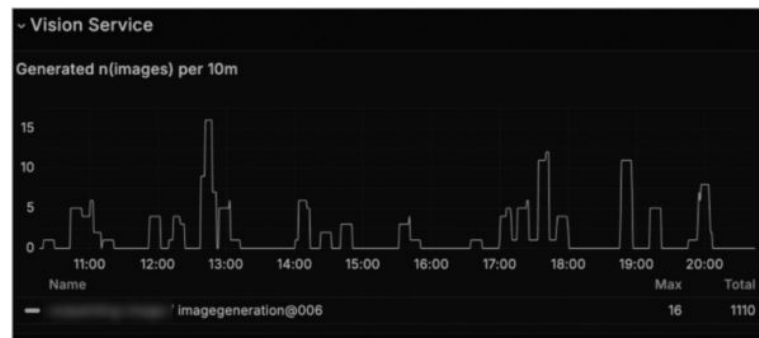
### • (모델, 작업)별 요청량 •



### • LLM 서비스의 (프로젝트, 모델)별 요청량 및 입출력 토큰 수 •



### • Vision 서비스의 (프로젝트, 모델)별 생성된 이미지 수 •



# 향후 계획

## 1. 현재 상태

- 게이트웨이 기능은 **API** 기반으로만 제공
- 사용자는 **Postman** 또는 직접 호출 코드 구현 필요  
→ 비개발자 접근성과 사용성에 한계

## 2. 개선 방향: AI 스튜디오와 통합

- AI플랫폼 내 **AI 스튜디오** 웹 기반 인터페이스 제공 예정
- 주요 기능:
  - 리소스 관리: 프로젝트, 자격증명, 템플릿 등
  - **API 요청 실행**: 웹에서 직접 테스트 및 실행
  - **템플릿 허브**: 프롬프트 템플릿 공유 및 협업

## 3. 기대 효과

- 개발자 외 사용자도 손쉽게 게이트웨이 활용
- 템플릿 품질 향상 및 표준화
- 협업 기반의 생성형 AI 서비스 개발 생산성 향상