

## 어린이가 작성한 이야기(일기) 기반 그림 동화 생성기 중간보고서



팀 : wUDC

201224544 차행철

202055610 최광진

201924511 오지웅

## 1. 요구조건 및 제약 사항 분석에 대한 수정사항

현재까지 프로젝트의 진행상황에 따르면, 착수보고서에서 다루었던 요구조건과 제약 사항 분석에 대한 수정사항은 없다. 현재 단계에서 구현중인 내용은 AI 모델 개선과, 모델의 내용을 서버를 거쳐 클라이언트 어플리케이션에 전달하는 기능이다. 이 과정에서 UI/UX는 사용자의 연령대를 고려하여 단순하고 직관적으로 설계중이고, 생성된 동화를 자체 플랫폼을 통해 공유하는 기능 역시 개발단계에 들어가있다. 또한, AI 모델은 모듈화된 파이프라인으로 구성하는 데 성공하였고 캐릭터의 일관성 역시 계속해서 개선되어지고 있다.

현재 다루고 있지 않은 내용인 '교육적 활용에 대한 확장성'은 핵심 기능의 구현이 끝난 후 추가적으로 다룰 예정이다.

## 2. 설계 상세화 및 변경 내역

### 2.1. 개발 언어 및 개발 도구

#### 2.1.1. 개발 언어

- Python 3.12+: AI 모델 개발 언어
- JavaScript: 웹 인터페이스 (향후 확장 예정)
- SQL: 데이터베이스 쿼리

#### 2.1.2. 개발 도구

- FastAPI: REST API 서버 개발 프레임워크
- Uvicorn: ASGI 기반 비동기 웹서버
- Pytest: 테스트 프레임워크
- Git: 버전 관리 시스템
- Docker: 컨테이너화 (향후 적용 예정)
- Unity API : 클라이언트 어플리케이션 개발

### 2.2. 시스템 구조

### 2.2.1. 사용자 인터페이스 (Front-end)

#### 기술스택

- Unity 6 (6000.0.44.f1)
- Plastic SCM (Version Control System)

### 2.2.2. 서버 측 애플리케이션 (Back-end)

#### 기술 스택

- Java 21, Spring Boot 3.x
- Gradle (Groovy DSL)
- application.properties 방식 설정
- 패키지 네임: com.wudc.storypool

#### 서버 환경

- Nginx: HTTPS Reverse Proxy (SSL 적용)
- Spring Boot App (JAR 실행)
- Redis: 이메일 인증/토큰 캐시
- MySQL (또는 MariaDB)

### 2.2.3. AI 파이프라인

#### 2.2.3.0. 요구 패키지

- Python 3.12+
- diffusers==0.34.0: 딥러닝 기반 이미지 생성 및 변환 모델(예: Stable Diffusion)을 쉽게 사용할 수 있도록 돕는 라이브러리
- easyocr==1.7.2: 다양한 언어를 지원하는 간편한 OCR(문자 인식) 라이브러리
- fastapi==0.115.14: Python으로 빠르고 효율적인 REST API 서버를 쉽게 개발할 수 있는 웹 프레임워크
- httpx==0.28.1: 비동기 및 동기 HTTP 요청을 지원하는 고성능 HTTP 클라이언트 라이브러리
- numpy==2.3.1: 고성능 수치 계산을 위한 배열 및 행렬 연산 라이브러리
- Pillow==11.2.1: Python에서 이미지 처리와 조작을 위한 라이브러리 (PIL의 후속)
- pydantic==2.11.7: 데이터 유효성 검사 및 설정 관리를 위한 타입 기반 모델링 라이브러리
- pytest==8.4.1: Python 테스트 코드 작성과 실행을 도와주는 테스트 프레임워크
- sentence\_transformers==4.1.0: 문장 임베딩 벡터 생성 및 문장 간 유사도 계산에 특화된 라이브러리
- torch==2.7.1: 딥러닝 모델 개발에 널리 쓰이는 PyTorch 라이브러리
- transformers==4.53.0: Hugging Face에서 제공하는 다양한 사전학습 NLP 모델 사용 라이브러리
- uvicorn==0.35.0: ASGI 기반 비동기 Python 웹서버, 주로 FastAPI와 함께 사용됨

#### 2.2.3.1. 아키텍처 특징

##### 1. 마이크로서비스 아키텍처

- 각 AI 모듈은 컨테이너화되어 독립 배포/확장 가능  
(예: OCR, Translator, StoryWriter, PromptMaker, ImageMaker)
- API Gateway는 통합 진입점 + 라우팅 + 인증/로깅

## 2. Factory + Strategy 패턴

- Factory: 각 Manager가 여러 구현체(vendor/model)를 선택 생성  
(예: OpenAI GPT, LLaMA, MarianMT)
- Strategy: AI 모델/서비스를 런타임에 교체 가능  
(예: LLaMA3 → Gemini, Stable Diffusion → SDXL)

## 3. 이미지 생성 프롬프트 최적화 아키텍처

- 생성된 이미지를 정량적 지표(CLIPScore, aesthetic score, identity similarity)를 바탕으로 재생성하거나 최종 선택을 결정하는 아키텍처
- SceneParser > PromptMaker > ImageMaker > EmbeddingManager > EvaluatorManager > ReRanker

## 4. llm + post processing 아키텍처

- llm 프롬프트 구조 특징:

one-shot: 모델에게 하나의 예시를 제공하고 패턴을 학습한 것 처럼 답변

CoT: 모델이 중간 추론 단계를 거치며 답을 도출하도록 지시

Self-consistency: 같은 질문에 대해 여러 답변을 생성한 후 가장 일관된 답을 선택

low Temperature: 모델 출력의 무작위성 정도를 조절하는 파라미터로 일관된 json 형식 출력을 위해서 0.3으로 두고 사용

multi-pass role prompting: 작업을 한 번에 끝내지 않고 여러 단계로 나눠 프롬프트 작성, 각 pass에서 다른 목표 수행

#### 2.2.3.2. 전체 플로우

사용자 일기 입력



OCR (손글씨 → 텍스트)



번역 (다국어 → 한국어/영어)



스토리 생성 (내용에 맞는 동화 생성)



장면 분석 (스토리 구조화) → 감정 분석 (장면의 감정 파악)



이미지 생성



완성된 그림동화를 사용자에게 제시

### 2.2.3.3. 엔드포인트

#### 헬스 체크

- `GET /health` - 시스템 상태 확인

#### OCR (광학 문자 인식)

- `POST /ocr/process` - 이미지 파일에서 텍스트 추출
- `POST /ocr/batch` - 여러 이미지 파일 일괄 처리

#### 번역

- `POST /translator/process` - 파일 번역
- `POST /translator/text` - 텍스트 번역

#### 감정 분석

- `POST /emotion/process` - 파일 감정 분석
- `POST /emotion/text` - 텍스트 감정 분석
- `POST /emotion/batch` - 여러 파일 일괄 감정 분석

#### 스토리 생성

- `POST /story/process` - 파일 기반 스토리 생성
- `POST /story/text` - 텍스트 기반 스토리 생성

#### 장면 분석

- `POST /scene/process` - 파일 기반 장면 분석
- `POST /scene/text` - 텍스트 기반 장면 분석

### 프롬프트 생성

- `POST /prompt/process` - 파일 기반 프롬프트 생성
- `POST /prompt/text` - 텍스트 기반 프롬프트 생성

### 이미지 생성

- `POST /image/process` - 파일 기반 이미지 생성
- `POST /image/text` - 텍스트 기반 이미지 생성

### 파이프라인

- `POST /pipeline/process` - 파일 기반 파이프라인 동작



### 2.2.3.3. 프로젝트 구조

#### 📁 프로젝트 구조

```
.storypool_ai_pipeline/
├── apis.py
├── main.py
├── requirements.txt
├── api_caller/
│   ├── api_caller_interface.py
│   └── api_caller_selector.py
├── emotion_classifier/
│   ├── emotion_classifier_interface.py
│   ├── emotion_classifier_manager.py
│   └── emotion_classifier_selector.py
├── image_maker/
│   ├── image_maker_interface.py
│   ├── image_maker_manager.py
│   └── image_maker_selector.py
├── llama_tools/
│   ├── llama_api_caller.py
│   └── llama_helper.py
├── object_analyst/
│   ├── object_analyst_interface.py
│   ├── object_analyst_manager.py
│   └── object_analyst_selector.py
├── ocr/
│   ├── easy_ocr.py
│   ├── ocr_interface.py
│   ├── ocr_manager.py
│   └── ocr_selector.py
├── prompt_maker/
│   ├── llama_prompt_maker.py
│   ├── prompt_maker_interface.py
│   ├── prompt_maker_manager.py
│   └── prompt_maker_selector.py
├── scene_parser/
│   ├── basic_scene_parser.py
│   ├── scene_parser_interface.py
│   ├── scene_parser_manager.py
│   └── scene_parser_selector.py
├── story_writer/
│   ├── llama_story_writer.py
│   ├── story_writer_interface.py
│   ├── story_writer_manager.py
│   └── story_writer_selector.py
├── translator/
│   ├── marian_translator.py
│   ├── nllb_translator.py
│   ├── translator_interface.py
│   ├── translator_manager.py
│   └── translator_selector.py
```

#### 2.2.3.4. 모듈별 상세구조

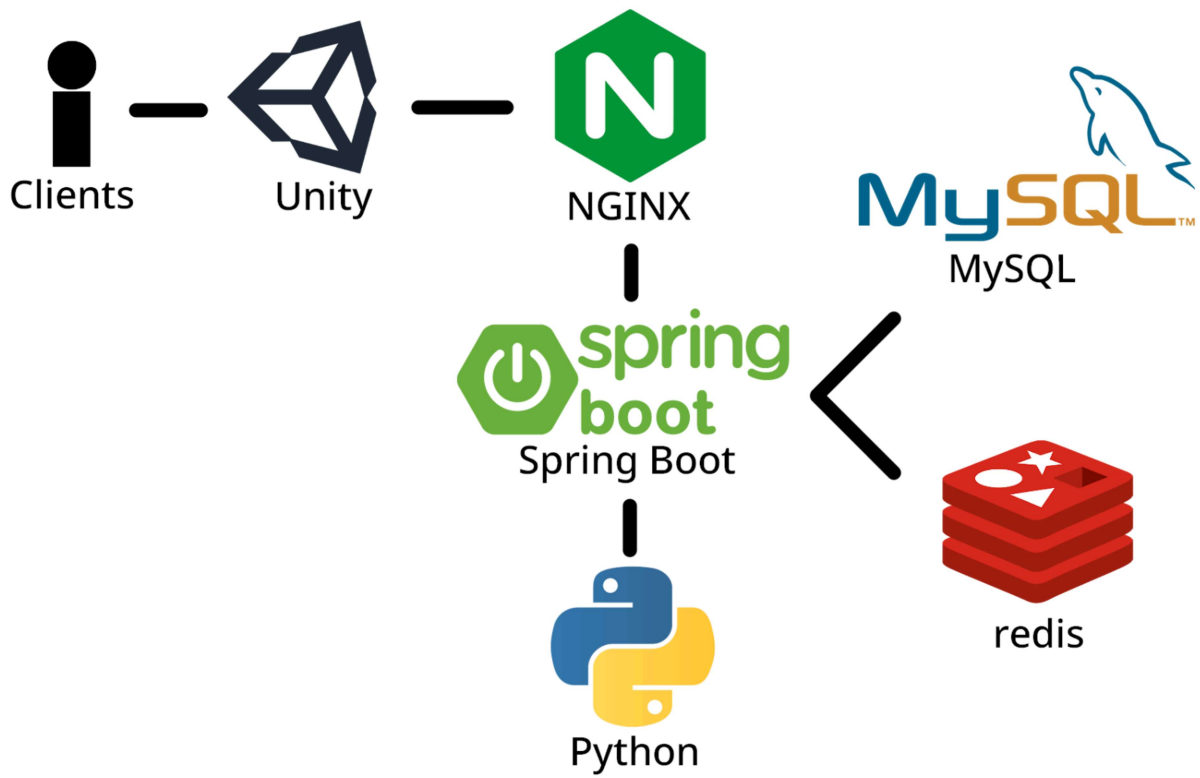
##### 공통구조

- interface: 기능을 추상화한 계약(Contract) 역할, 추상 메서드(abstract method)를 선언하여, 실제 구현체들이 반드시 구현해야 하는 메서드를 정의
- selector: 인터페이스 구현체(실제 기능을 하는 클래스)를 선택/생성하는 팩토리 역할, 인자로 받은 키워드나 설정값에 따라 적합한 구현체를 반환
- manager: 비즈니스 로직 담당, 인터페이스 구현체의 메서드를 호출하여 실제 작업 수행, 외부에서 전달받은 데이터 처리 (ex. 파일 경로에서 이미지 읽기)
- 구현체: 기능의 구현 로직을 담당하며, 외부에서 호출되는 interface에서 선언된 메서드를 통해 동작 수행.
- selector 입력 str 바탕으로 구현체 생성 > manager에 주입 > manager는 인터페이스 형태 사용

##### 상세구조 Github Link

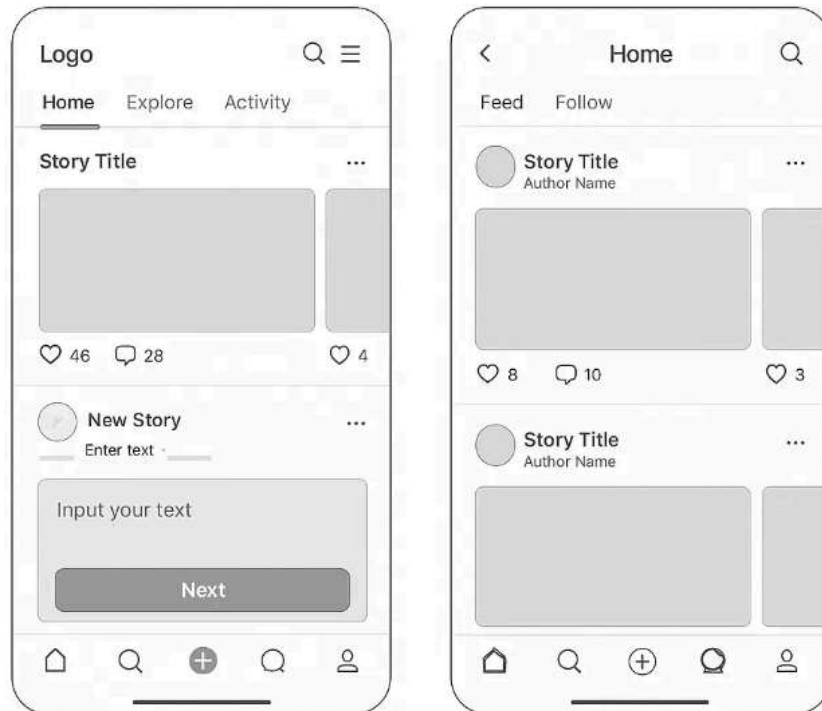
[https://github.com/jiwoong5/storypool\\_ai\\_pipeline/edit/main/README.md](https://github.com/jiwoong5/storypool_ai_pipeline/edit/main/README.md)

### 2.3. 전체 구조도



## 2.4. 시스템 흐름

### - 클라이언트

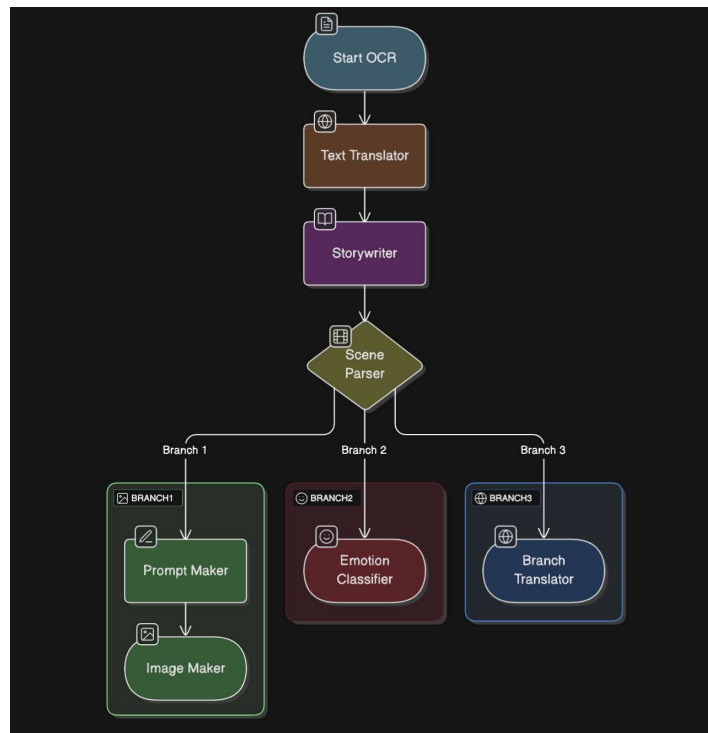


동화 제작에 필요한 기능(일기 캡처, 텍스트 인식등) 제공  
사용자가 만든 동화들을 시각적으로 표시  
여러 사용자들 사이에서 작성한 동화를 공유하는 플랫폼 제공

### - 서버

nginx - 리버스 프록시 : 클라이언트의 요청, 정보를 서버에 전달  
spring boot : 백엔드 로직 처리(ex. 사용자 인증, DB연동)  
mysql : 사용자의 기본정보 및 기타 파일 저장  
redis : 사용자의 로그인 정보 관리

## - AI with Python



OCR(easyocr) : 사용자가 작성한 일기를 캡처한 이미지에서 텍스트 추출  
 translator(marianmt): LLM 적용을 위해 한영 번역  
 story writer(llama 3.2:3b): 텍스트를 기반으로 이야기 제작  
 scene parser(user function): 스토리를 여러 장면으로 구분  
 prompt maker(llama 3.2:3b): 장면에 맞는 그림생성을 위한 프롬프트 생성  
 image maker(ghibli diffusion, dream shaper): 이미지 생성  
 emotion classifier(minilm): 텍스트에서 각 장면의 주요 감정 추출  
 translator(nllb): 사용자에게 제공하기 위해 다시 한글로 번역

### 2.5. 변경 내역

- scene parser : 스크립트 기반 -> llm 기반으로 변경
- kss 모듈 > simple\_split 을 구현하여 적용
- OCR, STT, TTS : Google Cloud platform AI5.2. 시스템 구조5.2.1. 사용자 인터페이스 (Front-end)
- AWS 클라우드 서비스를 활용하여 확장성과 유연성을 높이고 비용을 절감
- Amazon EC2 인스턴스에서 여러 개의 Docker컨테이너를 구성하여 애플리케이션을 배포

- Spring 프레임워크로 백엔드 서버를 구성하여 RestfulAPI통신과 트래픽에 따른 멀티스레딩을 활용
- 기존 기출문제들을 크롤링하여 Amazon RDS로PostgreSQL의 Master와 Slave 인스턴스를 구성해저장하고, 백엔드 서버와의고가용성 통신을 지원함
- Redis를 캐시 서버로 사용하여 자주 접근되는데이터를 인-메모리에 로드하여 애플리케이션의성능(반응 시간)을 높임
- Amazon S3를 사용하여 문제 이미지 및 듣기 음성과같은 비정형 데이터를 보관

### 3. 갱신된 과제 추진 계획

개발구분	세부항목	5월(완료)	6월(완료)	7월(진행중)	8월	9월
기획	요구사항 분석					
	데이터 수집					
AI 로직 설계	개별 모듈 테스트					
	모듈 최적화					
	파이프라인 구축 및 api 구조 설계					
서버 개발	User Auth 등 사용자 인증 기본 API					
	AI 연동 서비스					
	StoryDraft / Story API 구현					
	Post/Comment/Like 소셜 API 구현					
클라이언트 앱 개발	UI/UX 로직 설계					
	이미지, 애니메이션 제작					
	클라이언트 로직 개발					
테스트 & 배포	트러블 슈팅					
	플레이스토어 & 앱스토어 출시					
최종보고서 작성						

## 4. 구성원별 진척도

### 차행철

1. 로그인과 회원가입에 필요한 UI,UX 및 로직 (구현 완료)
2. 사용자가 원하는 동화를 선택할 수 있는 환경 제작 (일부 구현 완료)
3. 선택된 동화의 내용을 화면에 알맞게 표시 (일부 구현 완료)
4. 동화의 핵심 내용을 사용자에게 제시하기 위한 배치 알고리즘(진행 중)
5. 사용자의 연령대에 맞춘 아트, 애니메이션 제작 및 반영 (진행 중)
6. 클라이언트와 서버 사이의 데이터 주고 받기(구현 완료)

### 최광진

Spring Boot 기반 백엔드 서버 구현 내용

최소 구현 목표 Domain

1. Post (DB테이블 설계완료, API 구현 완료)
2. Comment (DB테이블 설계완료, API 구현 완료)
3. Like (DB테이블 설계완료)
4. StoryDraft (DB 테이블 설계 중)
5. DeviceToken (구현중)
6. Notification (구현중)
7. Fairytale (DB 테이블 설계 중)
8. Bookmark (DB테이블 설계완료)
9. User (DB테이블 설계완료, API 구현 완료)
- 10.Auth (구현 완료)

협업에 무리가 없도록 로컬용 임시 서버 구현 병행중. 이후 배포시 운영서버와 연결 예정



## 오지웅

1. ocr selector, manager, interface, easy ocr 구현체 및 api 구현완료
2. translator selector, manager, interface, marian, nllb 구현체 및 api 구현완료
3. storywriter selector, manager, interface, llama 구현체 및 api 구현완료
4. scene parser selector, manager, interface, llama 구현체 및 api 구현완료
5. prompt maker selector, manager, interface, llama 구현체 및 api 구현완료
6. image maker selector, manager, interface, dream\_shaper, ghibli\_diffusion  
구현체 및 api 구현완료
7. ai\_processing\_pipeline 구현 및 api 완료

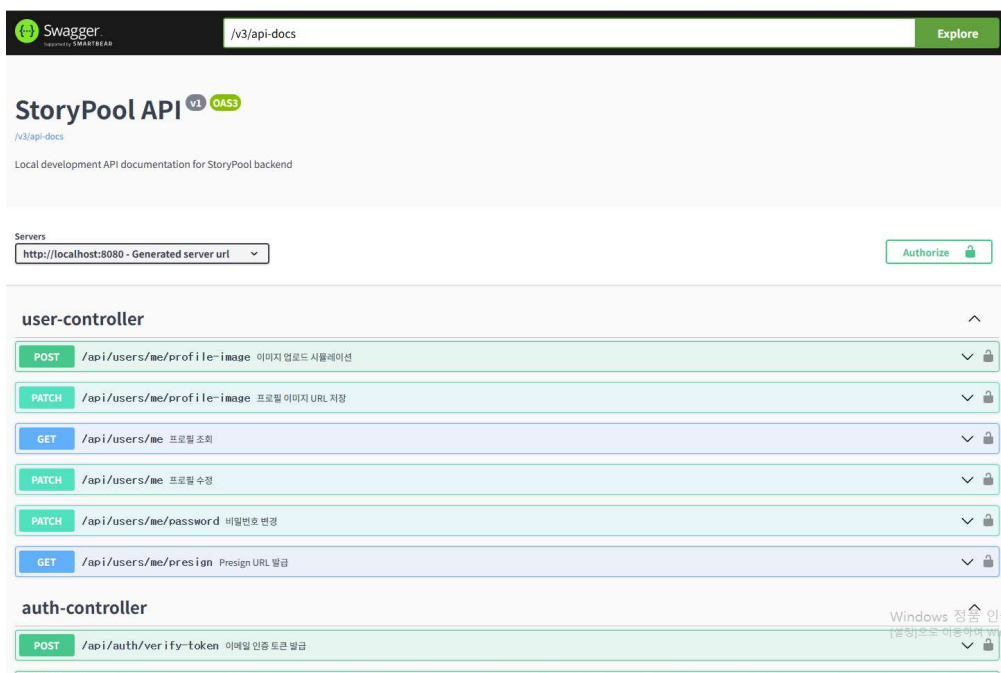
## 5. 보고 시점까지의 과제 수행 내용 및 중간 결과

### 1. 유니티 기반 모바일 빌드 클라이언트



로그인 / 회원가입 UI

### 2. 스프링 부트 기반 백엔드 서버



개발한 API 문서 페이지입니다.

### 3. AI 모델 및 API


<p>2025년 7월 6일, 일요일 _ 숨 막히는 모험의 하루</p> <p>오늘은 평범하게 시작될 줄 알았다: 아침부터 햇살은 따갑게 내리쬘었고 나는 그저 시원한 카페에 앉아 책이나 읽으려 했다. 그런데, 운명이란 참 이상하다</p> <p>카페로 향하면 길목에서 골목 안쪽에서 들려오는 고양이 울음소리를 들었다. 뭔가 이상했다 단순한 울음소리가 아니었다 두려움에 찬, 애절한 호기심과 걱정이 뒤섞여 나는 소리를 따라 골목 안으로 들어갔다</p> <p>그곳에는 작은 고양이와 낡은 담장 틈에 알밤이 켜진 채 몸부림치고 있었다 녀석의 눈빛이 나를 바라보았다 도와달라는 듯한 간절하 눈빛.</p> <p>손을 뻗어 조심스레 고양이를 꺼내려는 순간; 담장 위에서 쭈뼛 한 마리가 튀어나왔다 고양이 이는 깜짝 놀라 더 몸을 비틀며 울부짖었고 나는 순간 당황해 손을 멈췄다 하지만 이내 정신을 가다듬고 조심스럽게 담장을 흔들어 고양이의 배내는 데 성공했다.</p> <p>고양이는 자유를 찾자마자 내 품에 안락 한참을 뉘었다: 그렇게 우리는 잠시 골목 한가운 데서 숨을 고르며 앉아 있었다:</p> <p>그 순간 뒤쪽에서 누군가 다가오는 인기척이 들렸다. 돌아보니 동네에서 유명한 할아버지 한 분이셨다 그분은 나를 뵈며 보더니 미소를 지으며 말했다</p> <p>"자네, 참 착한 청년이군: 이 고양이는 우리 동네가 지켜주는 행운의 고양이라네"</p> <p>그 말을 들은 나는 괜히 뿌듯해졌다</p> <p>오늘은 단순한 하루가 아니었다 작은 생명을 구했고 골목 어귀에서 예상치 못한 따뜻함을 느꼈다</p> <p>이 작은 모험이 내게 오래 기억될 것 같다</p> <p>울음 발을</p>	<p>Sunday, July 6, 2025. By the way, fate is so strange that when you go to a cafe, there's no cat's call in the back of the alley. Fearful, curious and concerned, mixed up in the air, the little cat in the alley, struggling with its front paws in the old fence, looking at his eyes and begging for help. As soon as the hand reached out and carefully tried to pull out the cat frame; a rat on the fence bursts up, and then, as suddenly as it was startled, it staggered more and more, and as soon as I was in a state of panic, I stopped, but within a short time, I managed to shake my mind and carefully to pull the cat out of the fence. As soon as I find my cat's freeness, I can't hold my hand: so we don't sit in the alley for a moment, just sitting there in the middle of the room, there's a man who's coming. And when I look back, one of the most famous of my grandfathers in town, he looks at me, and he looks at me, and he's neglected, and he says, "Well, this is a good young man, this is our lucky cat." And I'm so proud of that, and today I feel a little life and a little warm and unexpected in the alley, and I remember this little adventure for a long time.</p>
ocr 결과	translator 결과
<pre>-- It was a sweltering Sunday afternoon on July 6, 2025, and I had just finished a long day of work. Feeling a bit down, I decided to grab a cup of coffee at my favorite cafe in town. As I sat down at a small table by the window, I noticed a tiny cat stuck in an old fence behind the alley.  I couldn't help but feel a pang of sympathy for the poor animal. I slowly reached out my hand, trying not to startle it further. As soon as my fingers made contact with the cat's fur, a rat on the fence burst up, its head eyes fixed on us. I froze, expecting it to attack, but instead, it suddenly scurried and staggered. Like it had been electrocuted or something. I was in a state of panic, my heart racing, but then I managed to shake off my fear and carefully pulled the cat out of the fence.  The moment I felt the paw land safely on the ground, I couldn't hold back my tears of joy. The little creature looked up at me with its big green eyes, as if thanking me for its rescue. We sat there in the alley, side by side, enjoying the warm sunshine and the soft breeze.  Just as we were starting to relax, a man appeared from nowhere. His weathered face creased into a smile. He was one of my grandfather's old friends from town, a man I had always heard stories about but never met in person.  He approached us slowly, his eyes twinkling with amusement, and said, "Well, this is a good young man, this is our lucky cat." His voice was warm and familiar. Like a gentle breeze on a summer day. He looked at me with a mix of pride and affection, and I felt a little life and a little warm and unexpected in the alley, and I remember this little adventure for a long time.  As we sat there in the alley, looking in the warmth and light, I realized that sometimes life surprises us with unexpected moments of joy and connection. And I knew that this little adventure would stay with me forever, a reminder of the power of kindness and a little bit of magic in the world.  --</pre>	<pre>{   "action": "pull_cat",   "message": "The cat was stuck in the fence.",   "timestamp": "2025-07-06T15:27:00Z",   "processing_time": null,   "output_offset": null,   "scenes": [     {       "scene_number": 1,       "scene_title": "The Cat in the Alley",       "characters": [         "Cat",         "Fence",         "Rat"       ],       "location": "Alley",       "time": "Afternoon",       "mood": "Concerned",       "summary": "I grabbed a cup of coffee at the cafe and noticed a cat stuck in an old fence behind the alley.",       "dialogue_count": 0,       "start_char": null,       "end_char": null     },     {       "scene_number": 2,       "scene_title": "The Cat in the Alley",       "characters": [         "Cat",         "Fence",         "Rat"       ],       "location": "Alley",       "time": "Afternoon",       "mood": "Surprised",       "summary": "I pulled the cat out of the fence and it thanked me with its big green eyes.",       "dialogue_count": 1,       "start_char": null,       "end_char": null     },     {       "scene_number": 3,       "scene_title": "The Cat in the Alley",       "characters": [         "Cat",         "Fence",         "Rat"       ],       "location": "Alley",       "time": "Afternoon",       "mood": "Grateful",       "summary": "A man appeared and complimented me on rescuing the cat, showing pride and affection.",       "dialogue_count": 1,       "start_char": null,       "end_char": null     }   ] }</pre>
story writer 결과	scene parser 결과
<pre>"scenes": [   {     "scene_number": 1,     "prompt": "Concerned young adult woman with heart-shaped face, long brown hair tied in a ponytail, wearing a fitted white button-down shirt and dark jeans, sits at a small table in a bustling cafe, looking worried, soft afternoon light casting a warm glow."   },   {     "scene_number": 2,     "prompt": "Surprised young adult woman with heart-shaped face and long brown hair tied in a ponytail, wears the same fitted white button-down shirt and dark jeans as before, sits at a table in the alley, holding a cat with big green eyes."   },   {     "scene_number": 3,     "prompt": "Grateful young adult woman with heart-shaped face and long brown hair tied in a ponytail, wears the same fitted white button-down shirt and dark jeans as before, stands beside a man with a kind expression, both looking out at the city."   } ], "total_scenes": 3</pre>	
prompt maker 결과	image maker 결과 1



image maker 결과 2



image maker 결과 3

#### 참고자료

- [Investigating Prompt Engineering in Diffusion Models](<https://arxiv.org/abs/2211.15462>)

- [The Chosen One: Consistent Characters in Text-to-Image Diffusion Models]([https://arxiv.org/abs/2311.10093?utm\\_source=chatgpt.com](https://arxiv.org/abs/2311.10093?utm_source=chatgpt.com))