[NursingXR]Interpreter 통합 및 LLM기반 c# 코드 생성

혼합현실 2024.12.17 Tue

컴퓨터·소프트웨어 학과(미래자동차 SW 융합전공)

SELab 석사 1기 정지나

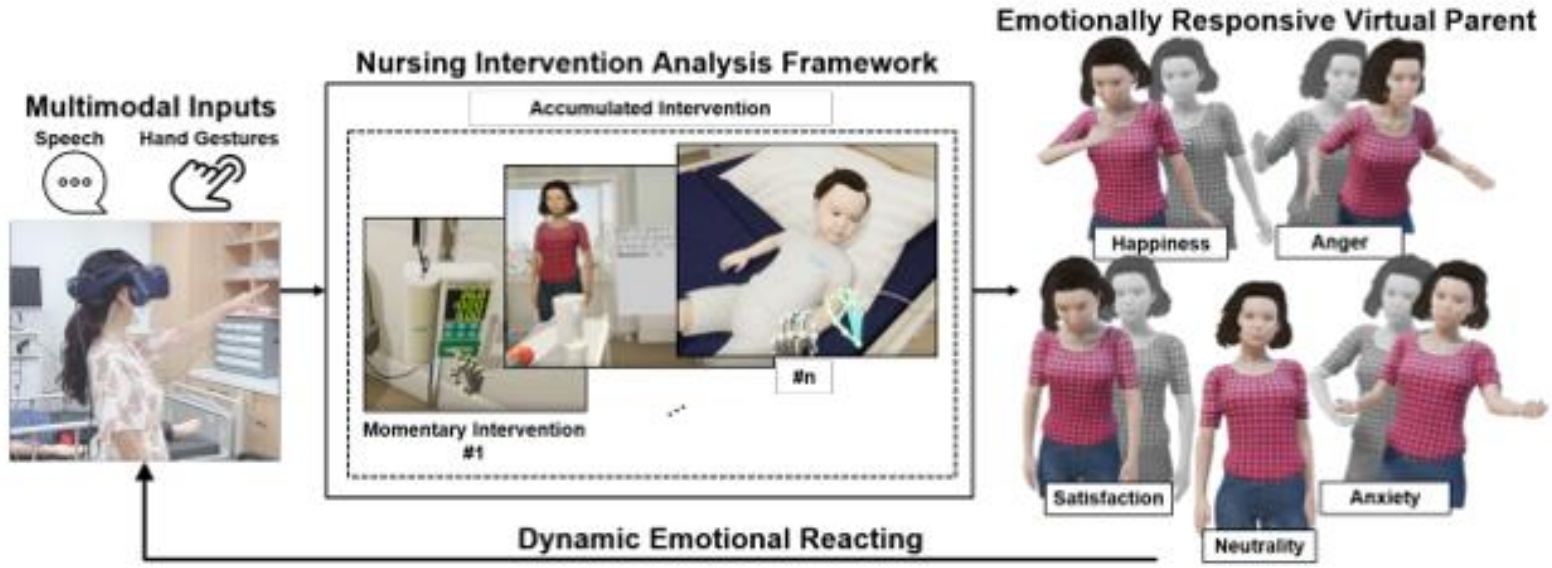
[NursingXR] Interpreter 통합 및 LLM 기반 c# 코드 생성

[목차]

NursingXR 이란?

간호의 본질에 충실한 인간적인 돌봄을 제공할 수 있는 간호인재를 양성하기 위해

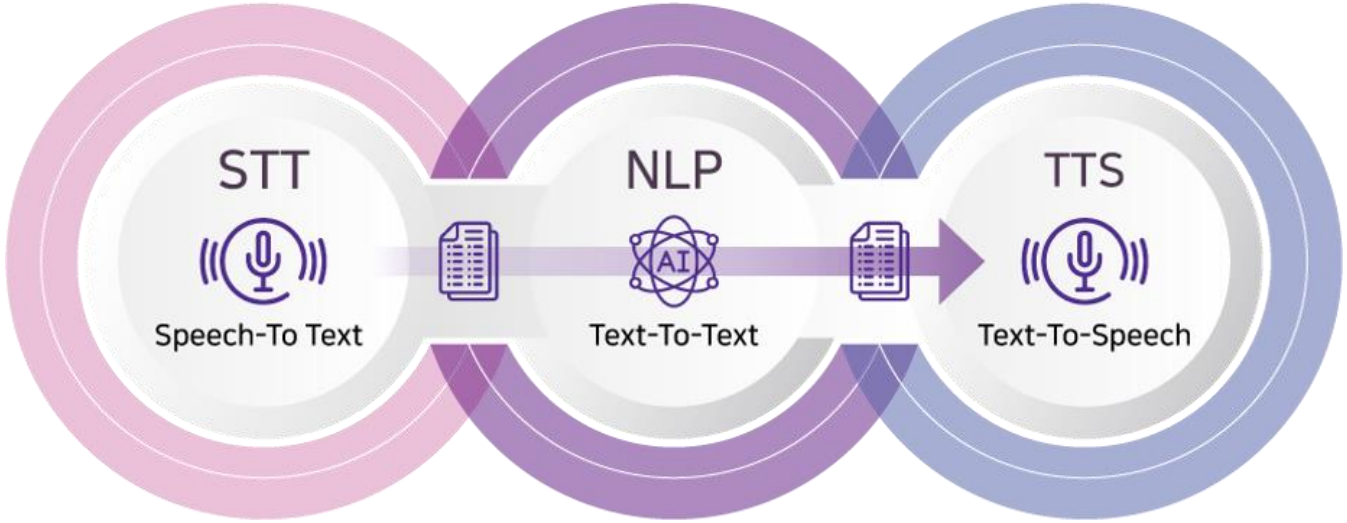
“AI 기반 감성 커뮤니케이션 환경을 제공하는 실감형 간호교육플랫폼” 을 제공하는 스마트 에듀테크 기업.



NursingXR 이란?

NLP의 정확성과 **생성형 AI의 창의성**이 만나 탄생한 최적의 학습 환경

- NLP(자연어 처리) : 인간의 언어를 해석, 조작 및 이해하는 능력을 컴퓨터에게 부여하는 기계 학습 기술



과제 개요

“ LLM을 활용하여 Unity 환경에서 C# 코드를 자동으로 생성, 분석, 테스트, 그리고 통합하는 시스템을 구축하는 것”

- LLM(Large Language Model): 방대한 양의 데이터로 사전 학습된 초대형 딥 러닝 모델

[주요 구성 요소]

1. LLM통합
   - Unity Editor 에 LLM 서버와 통신할 수 있는 인터페이스를 구축
   - Editor 은 유니티 내 컴포넌트나 변수들을 읽어와서 개발자의 기능 요청 사항과 함께 LLM으로 전달
2. 코드 생성 및 분석
   - 요청된 기능을 LLM이 코드로 생성하면 이를 유니티로 전송 후 스크립트 파일로 저장하여 유니티 내 반영

과제 요구 사항 분석

1. LLM 서버 연동
   - OPEN AI ChatGPT LLM 서버와 안정 적인 연결 구현
2. 코드 설명 및 생성
   - 기존 코드 설명

     : LLM을 활용해 기존 C# 코드에 대한 상세 주석 및 설명 자동 생성
   - 새로운 코드 생성

     : 기능 요구 사항을 LLM에 전달해 자동으로 C# 코드 생성

     ex) 간호사가 청진기를 드는 동작 관련 코드의 구체적 기능 요구 설명
3. 사용자 Interface 개발(GUI)
   - Unity Plugin 형태로 통합하여, GUI 창을 통해 Prompt를 입력하고 생성된 LLM응답 확인후 반영 여부 선택

과제 요구 사항 분석

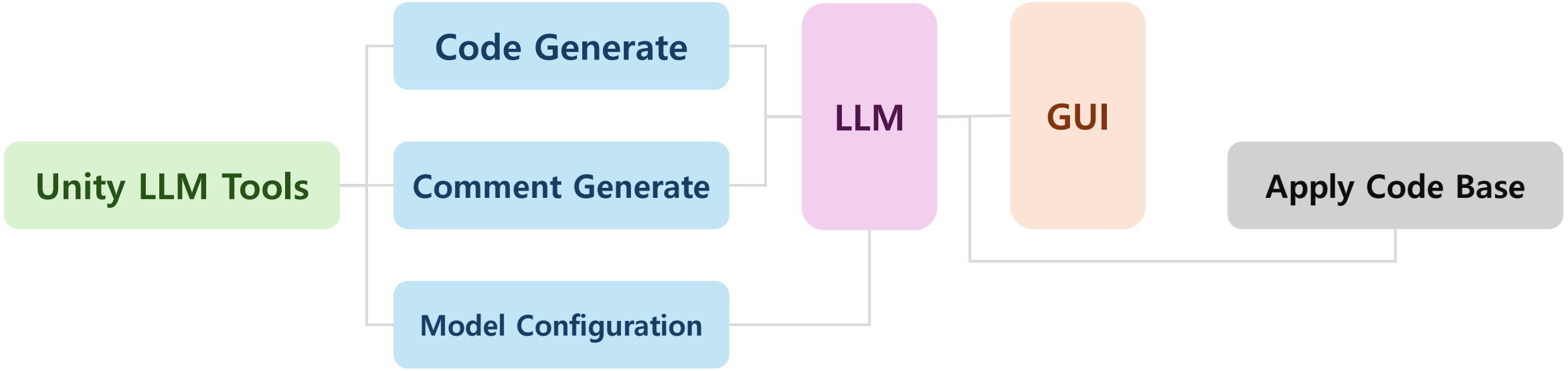
1. LLM 서버 연동
   - OPEN AI ChatGPT LLM 서버와 안정 적인 연결 구현
2. 코드 설명 및 생성
   - 기존 코드 설명

     : LLM을 활용해 기존 C# 코드에 대한 상세 주석 및 설명 자동 생성
   - 새로운 코드 생성

     : 기능 요구 사항을 LLM에 전달해 자동으로 C# 코드 생성

     ex) 간호사가 청진기를 드는 동작 관련 코드의 구체적 기능 요구 설명
3. 사용자 Interface 개발(GUI)
   - Unity Plugin 형태로 통합하여, GUI 창을 통해 Prompt를 입력하고 생성된 LLM응답 확인후 반영 여부 선택

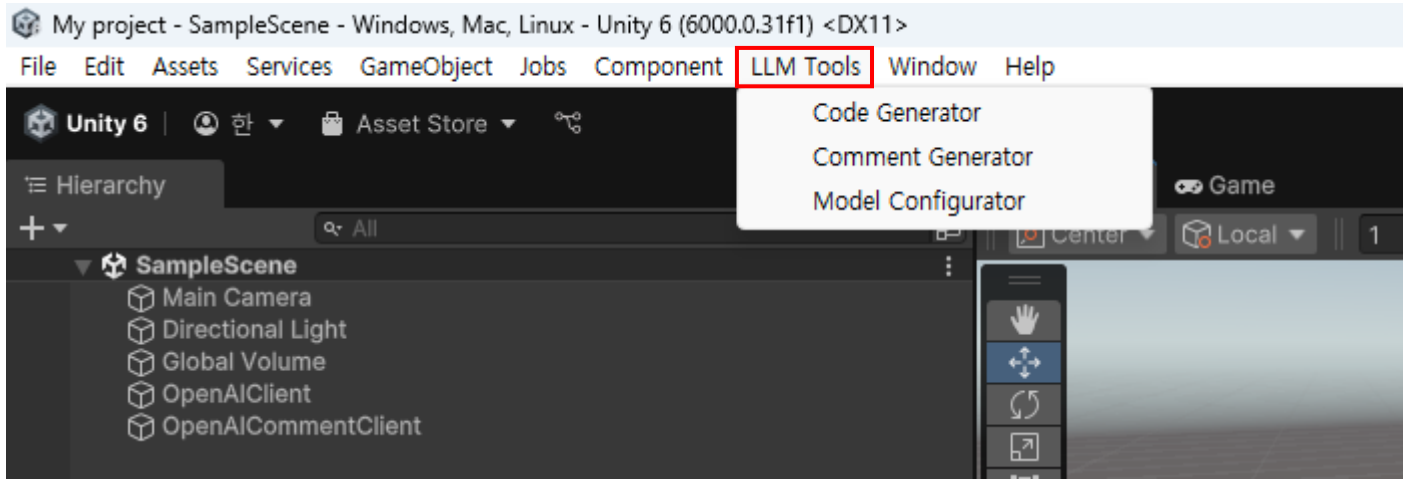
프로젝트 기능 아키텍처


Unity LLM Tools
Code Generate
Comment Generate
Model Configuration
LLM
GUI
Apply Code Base

프로젝트 기능

Unity LLM Tools

- LLM 과 API 통신 하며 코드베이스를 토대로 코드 생성 및 주석 생성 작업에 특화된 Tool
- Unity 에디터 내에서 플러그인 형태로 제공



프로젝트 기능

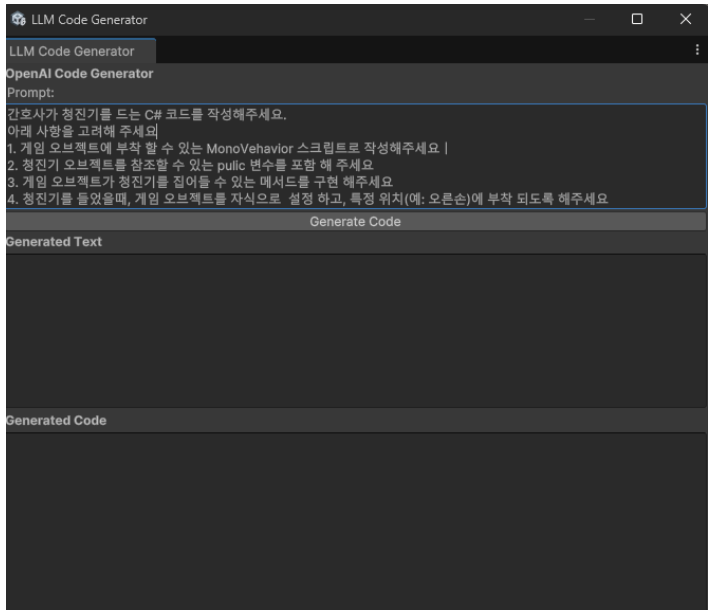
Code Generate

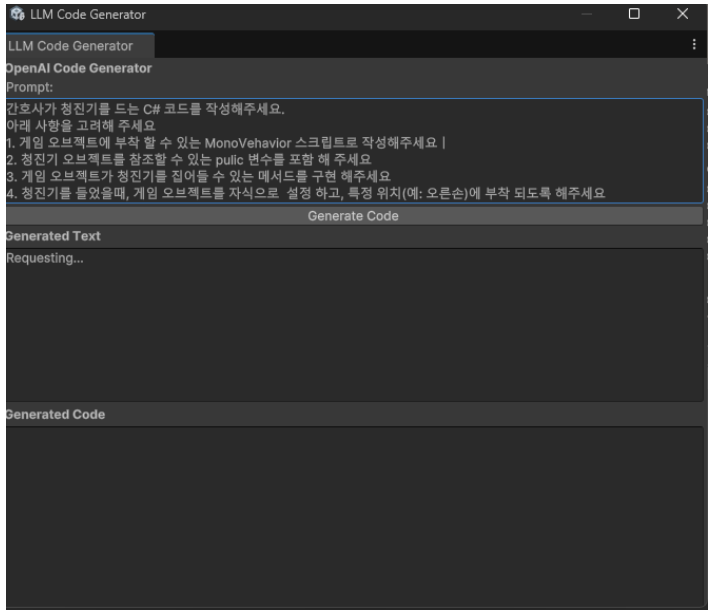
- OpenAI GPT 모델과 연동하여 개발자가 원하는 동작이나 기능에 대한 C# 코드를 자동으로 생성
  - 반복적이거나 단순한 작업 자동화.
  - 개발 시간 단축.
- 유저 Prompt 입력 → LLM (OpenAI API) → C# 코드 생성 → Unity Editor 저장 및 GUI 반영

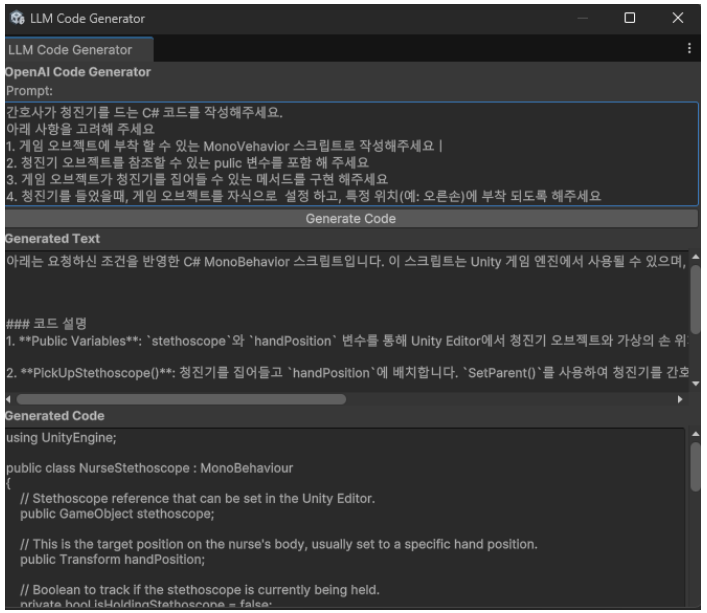
프로젝트 기능

Code Generate

- 사용자가 입력한 Prompt 와 C# 코드 생성에 맞춘 System Prompt를 LLM API에 입력
- LLM 응답에서 Text 와 Code 를 분리하여 저장 및 Plugin GUI 에 반영



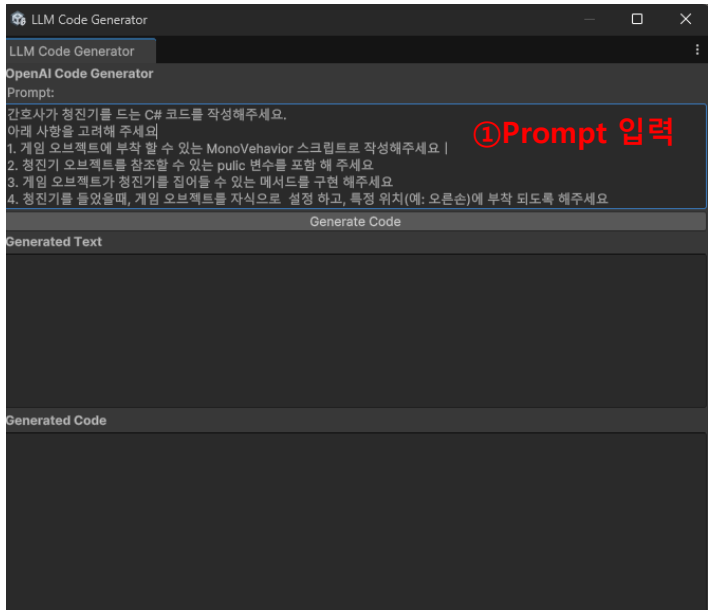


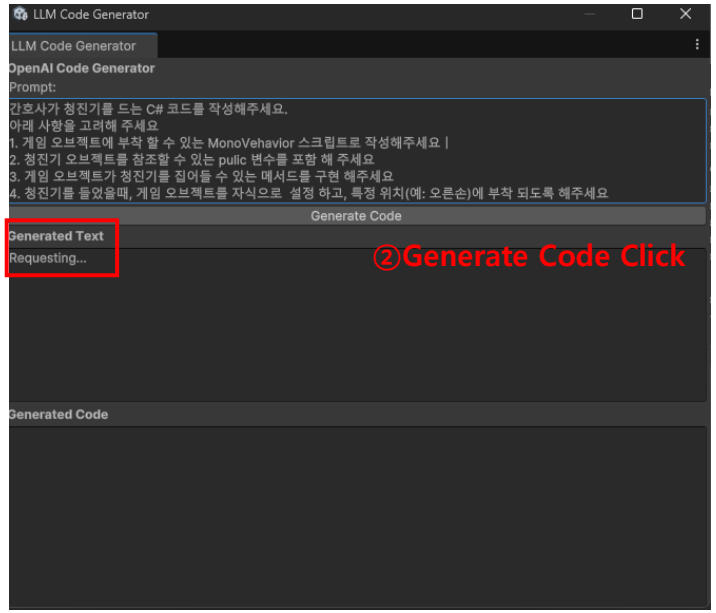


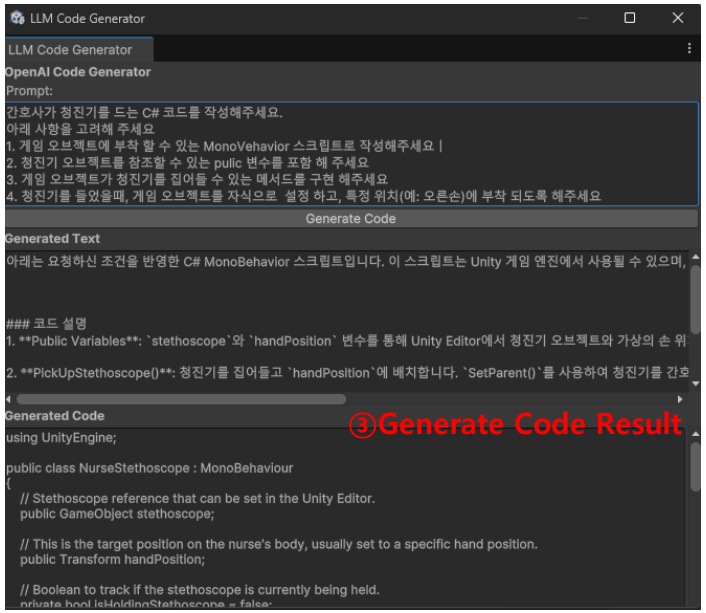
프로젝트 기능

Code Generate

- 사용자가 입력한 Prompt 와 C# 코드 생성에 맞춘 System Prompt를 LLM API에 입력
- LLM 응답에서 Text 와 Code 를 분리하여 저장 및 Plugin GUI 에 반영







프로젝트 기능

Comment Generate

- OpenAI API를 활용하여 기존 C# 코드에 자동으로 주석을 추가하는 기능

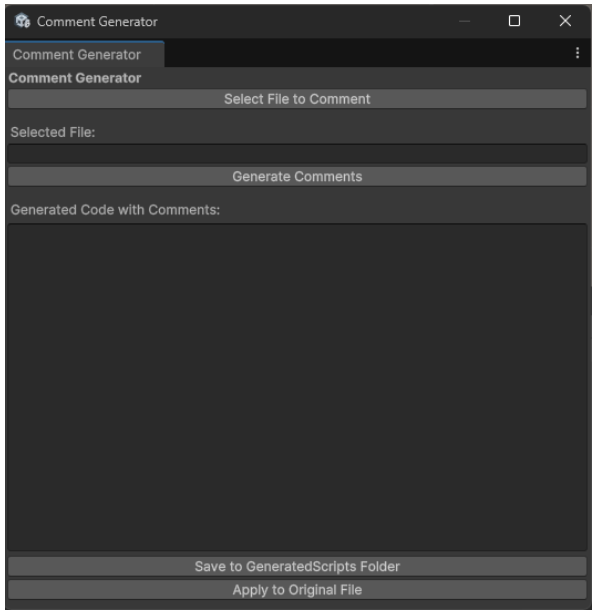
[목적]

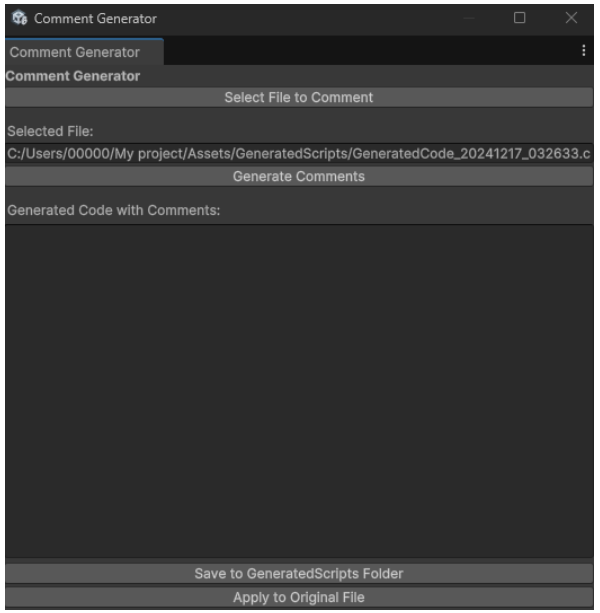
- 코드 가독성 및 유지보수성 향상.
- 협업 시 코드 이해도 증대.

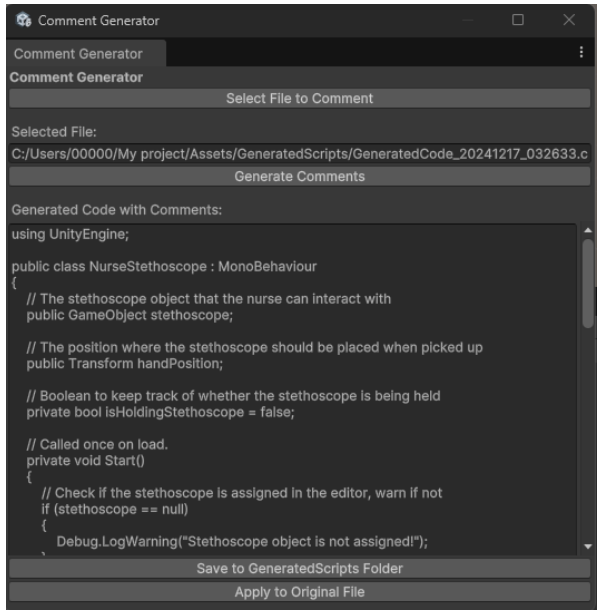
프로젝트 기능

Comment Generate

- LLM 과 API 통신 하며 Unity 내에서 코드베이스를 토대로 코드 생성 및 주석 생성 작업에 특화된 Plugin
- 생성 Scripts 폴더에 저장 하기, 원본 코드에 반영 해서 저장 하기



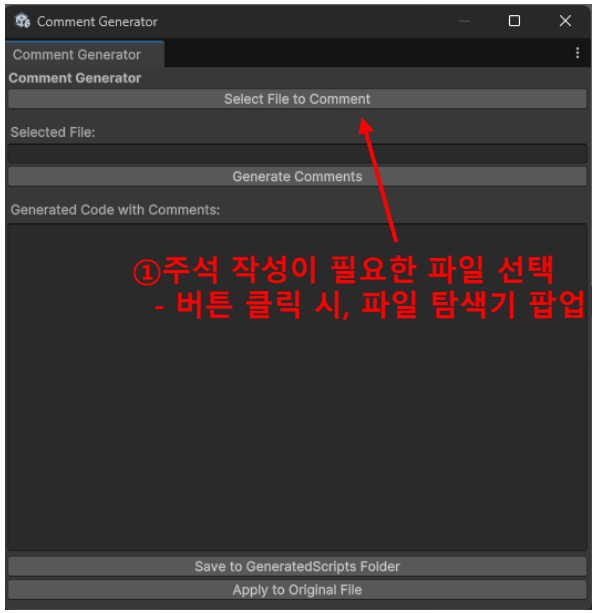


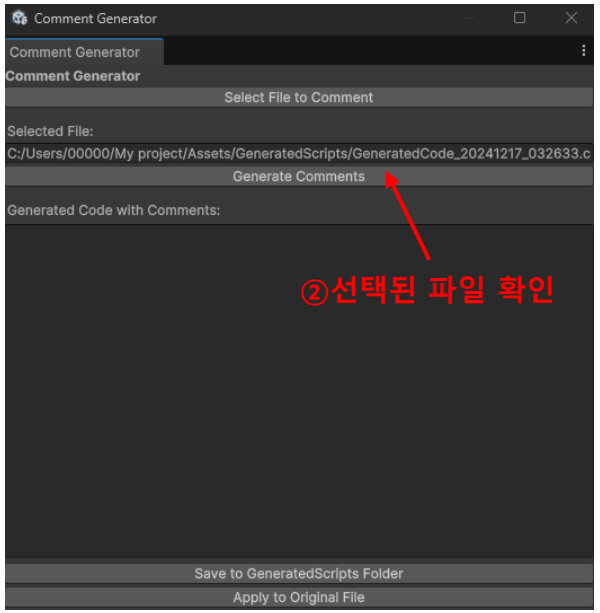


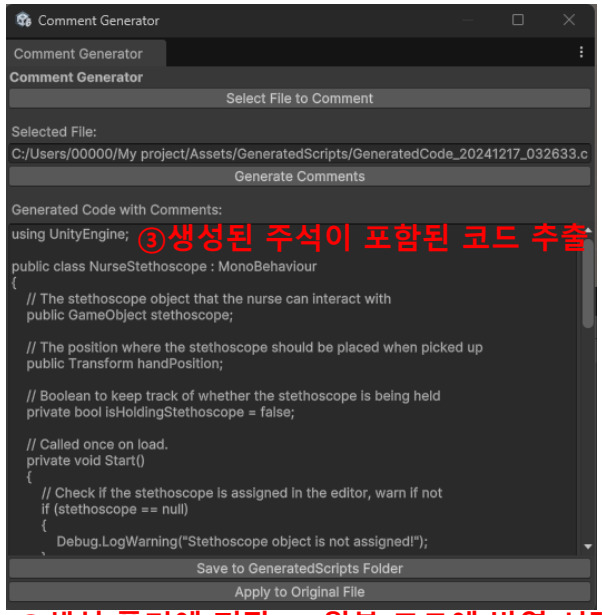
프로젝트 기능

Comment Generate

- LLM 과 API 통신 하며 Unity 내에서 코드베이스를 토대로 코드 생성 및 주석 생성 작업에 특화된 Plugin
- 생성 Scripts 폴더에 저장 하기, 원본 코드에 반영 해서 저장 하기





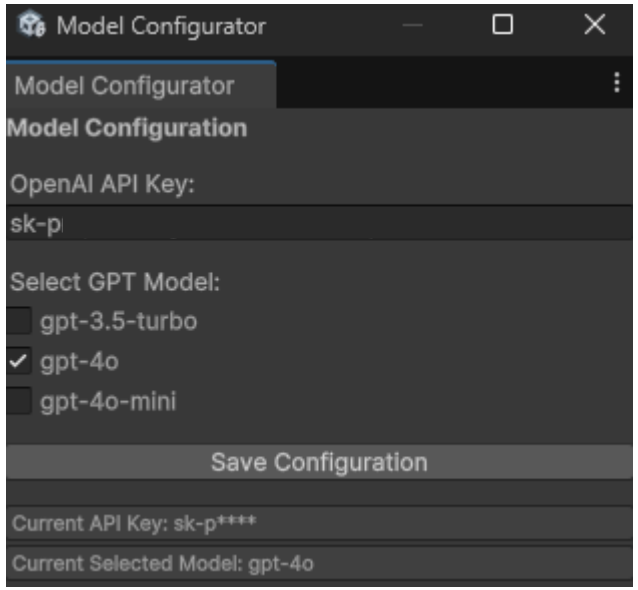


④생성 폴더에 저장 or 원본 코드에 반영 선택

프로젝트 기능

Model Configuration

- LLM(OpenAI API)과의 통신을 위해 API Key 및 모델 설정을 관리하는 기능.
- Unity 에디터 내 GUI 환경에서 설정을 쉽게 변경하고 저장 가능.



[목적]

- API Key를 안전하게 관리.
- 필요에 따라 GPT 모델(gpt-3.5-turbo, gpt-4o 등)을 선택.
- 설정을 한 번 저장하면 플러그인의 다른 기능(Code Generator, Comment Generator)에서 공통으로 사용.

프로젝트 기능

LLM

- OpenAI API와의 통신을 통해 다양한 작업을 수행.
- System Prompt를 통해 모델이 특정 역할과 목적에 맞게 작업을 수행하도록 지시.
  - System Prompt?
    - LLM이 특정 작업을 수행하도록 명확하게 역할과 목표를 설정.
    - System Prompt를 사용하면 작업에 특화된 결과를 얻을 수 있음.

- Code Generate 기능과 Comment Generate 기능에 각각 적합한 System Prompt를 생성하여 LLM의 작업 수행을 도움.

프로젝트 기능

LLM

- Code Generate System Prompt

```
"You are an assistant that generates C# code. "
"Ensure the code is clean, well-commented, and organized into a single C# file."
```

- Comment Generate System Prompt

```
"You are an assistant that adds comments to existing C# code. "
"Your job is to add clear and useful comments to the code without modifying the original structure. "
"Return the updated code with only the comments added."
```

프로젝트 기능

Apply Code Base

- LLM이 생성한 코드 또는 주석을 기존 코드베이스에 반영하는 기능
- 사용자 작업 흐름을 단순화하고, 수동으로 복사/붙여넣기를 할 필요 없이 자동으로 코드 파일을 업데이트

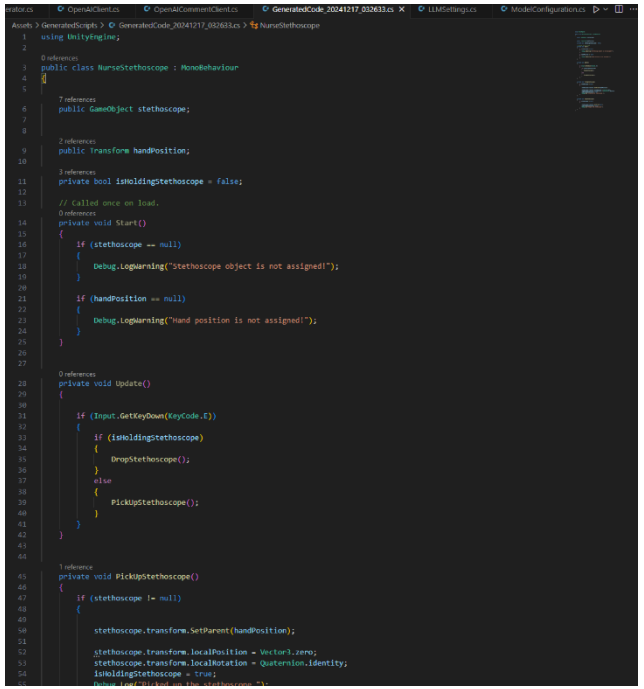
[목적]

- 작업 시간 절약 및 개발 효율 향상
- 기존 코드와 일관성 유지
- 오류 최소화 (수동 작업을 줄임)

프로젝트 기능

Apply Code Base

- “Comment Generate” 기능에서 “Apply to Original Code” 선택 시, 자동으로 주석이 원본 코드와 병합되어 반영



```
using UnityEngine;

public class NurseStethoscope : MonoBehaviour
{
    public GameObject stethoscope;

    public Transform handPosition;

    private bool isHoldingStethoscope = false;

    // Called once on load.
    private void Start()
    {
        if (stethoscope == null)
        {
            Debug.LogWarning("Stethoscope object is not assigned!");
        }

        if (handPosition == null)
        {
            Debug.LogWarning("Hand position is not assigned!");
        }
    }

    private void Update()
    {
        if (Input.GetKeyDown(KeyCode.E))
        {
            if (isHoldingStethoscope)
            {
                DropStethoscope();
            }
            else
            {
                PickUpStethoscope();
            }
        }
    }

    private void PickUpStethoscope()
    {
        if (stethoscope != null)
        {
            stethoscope.transform.SetParent(handPosition);

            stethoscope.transform.localPosition = Vector3.zero;
            stethoscope.transform.localRotation = Quaternion.identity;
            isHoldingStethoscope = true;
```

```
using UnityEngine;

public class NurseStethoscope : MonoBehaviour
{
    // The stethoscope object that the nurse can interact with
    public GameObject stethoscope;

    // The position where the stethoscope should be placed when picked up
    public Transform handPosition;

    // Boolean to keep track of whether the stethoscope is being held
    private bool isHoldingStethoscope = false;

    // Called once on load.
    private void Start()
    {
        // Check if the stethoscope is assigned in the editor, warn if not
        if (stethoscope == null)
        {
            Debug.LogWarning("Stethoscope object is not assigned!");
        }

        // Check if the hand position is assigned in the editor, warn if not
        if (handPosition == null)
        {
            Debug.LogWarning("Hand position is not assigned!");
        }
    }

    // Called once per frame.
    private void Update()
    {
        // Check for user input, pressing the 'E' key
        if (Input.GetKeyDown(KeyCode.E))
        {
            // Toggle between picking up and dropping the stethoscope
            if (isHoldingStethoscope)
            {
                DropStethoscope();
            }
            else
            {
                PickUpStethoscope();
            }
        }
    }

    // Method to pick up the stethoscope
    private void PickUpStethoscope()
    {
        if (stethoscope != null)
        {
            // Set the stethoscope as a child of the hand position
            stethoscope.transform.SetParent(handPosition);

            // Set local position and rotation to align the stethoscope with the hand
            stethoscope.transform.localPosition = Vector3.zero;
            stethoscope.transform.localRotation = Quaternion.identity;

            // Update the boolean to indicate the stethoscope is being held
            isHoldingStethoscope = true;
```

프로젝트 결과

[주요 기능 정리]

- Code Generate
  - 사용자의 자연어 입력(prompt)을 바탕으로 C# 코드를 자동으로 생성.
  - 특화된 시스템 프롬프트를 통해 깔끔하고 단일 파일로 구성된 코드 제공.
- Comment Generate 기능
  - 기존 C# 코드에 논리적이고 명확한 주석을 자동으로 추가 및 반영
  - 코드 구조를 변경하지 않고 주석만 반영.
- Model Configuration
  - API Key 입력 및 OpenAI 모델 선택 기능.
  - 토글 UI를 통해 gpt-3.5-turbo, gpt-4o, gpt-4o-mini 모델 선택 가능

프로젝트 결과

[기대 효과 및 성과]

항목	기대 효과 및 성과
개발 시간 단축	수작업 없이 코드 생성 및 주석 추가 → 개발 효율 향상.
코드 품질 향상	명확한 주석과 구조화된 코드 제공 → 유지보수성 개선.
초보자 접근성	자연어 입력만으로 코드 작성 가능 → 기술 장벽 감소.
협업 개선	코드의 이해도 증대 → 팀 간 원활한 협업 지원.
자동화 프로세스	기존 코드에 자동 반영 → 수동 복사/붙여넣기 오류 방지.

프로젝트 결과

[한계점]

- Code Generate 기능에서는 자동 반영 기능 미비
    - LLM 응답이 완벽하지 않을 수 있어 개발자의 확인이 필요.

[작업물]

https://github.com/jung-gina/NursingXR_ICPBL_Project

[NursingXR] Interpreter 통합 및 LLM 기반 c# 코드 생성

Q & A