

코드 크래프트 AI



(주)유틸플러스인터랙티브 / (주)인텔리코드

목차

1. 메타버스 도입 현황
2. 프로젝트 진행
3. 프로젝트 내용
4. 코드 생성 워크플로우
5. 검색 기능 고도화
6. 서비스 실행

1. 메타버스 도입 현황

- 메타버스 생성AI 사용 사례
 - 로블록스(Roblox)
 - 가상 환경에서 코드 생성 기능 제공
 - 에픽 게임즈(Epic Games)
 - Unreal Engine AI 확장
 - 스페이셜(Spatial)
 - 3D 오브젝트 및 코드 생성
 - 마인크래프트(Minecraft)
 - AI로 명령어 및 스크립트 생성
 - AI 기반 모드 제작
 - 브이알챗(VRChat)
 - Script Assistants



2. 프로젝트 진행

사전 기획 및 설계

- 개발 계획 수립
 - * 다른 메타버스 서비스의 생성AI 코드 생성 기능 사례 분석
 - * 디토랜드 내에서의 코드 생성 기능 계획 분석
 - * 목표, 범위, 일정, 리소스 등의 분석 설계
- 서비스 및 이벤트 기획
 - * 실제 사용자를 대상으로 시나리오 기반 기획
 - * 코드 생성 UI/UX 분석 및 기획
- 상세 설계
 - * UI/UX 디자인 프로토타입 제작 및 테스트 결과 분석
 - * 데이터베이스 구조 설계 진행
 - * 기능별 기술 스펙 분석 및 설계 진행

메타버스 개발

- AI 학습 데이터 설계 및 생성
 - * 코드 생성을 위한 기반 학습 데이터 조사
 - * 자체 SDK/API 학습 데이터 포맷 분석
 - * 학습 데이터 수집 및 정형화 진행
- AI 학습 데이터 서버 구축 및 고도화
 - * GPU 서버를 활용한 RAG 시스템 구축
 - * 학습 데이터 저장을 위한 벡터 데이터베이스 시스템 구축
 - * RAG 시스템을 위한 파이프라인 프로세스 구성
 - * 고도화된 Retrieval을 위한 다중 검색 시스템 구성
- AI Coding Assistant 모듈 구현
 - * 사용자 UI/UX에서의 코딩 생성 기능 연동 개발
 - * 사용자 사용 API 개발 진행
- 저작 도구 내 AI Coding 연동
 - * 저작 도구 내 AI 기능 기획 및 구현
 - * 다양한 사용자 UX를 지원하는 AI 코딩 기능 개발

QA 및 검증

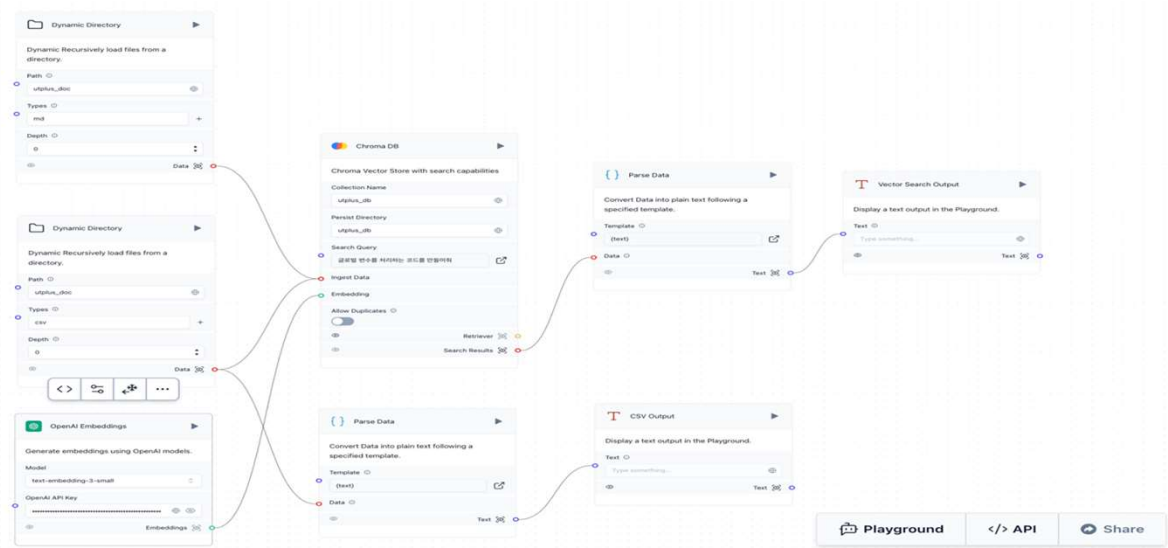
- 내부 QA와 외부 FGT 검증
 - * 내부에서 사용자 시나리오 기반으로 테스트 진행
 - * 외부 사용자 그룹을 대상으로 시나리오 기반 테스트 진행
 - * 테스트 결과 수집 및 분석
- 산출물 최종 검수 및 결과 보고
 - * 산출물 최종 픽스 및 패키징 진행
 - * 내부 프로젝트 결과 보고
- 버그 픽스 및 서비스 안정화
 - * 다양한 테스트를 통한 에러/버그 수집 및 분석
 - * 보고된 버그 픽스 개발 진행
 - * 시스템 구조 안정화 진행

3. 프로젝트 내용

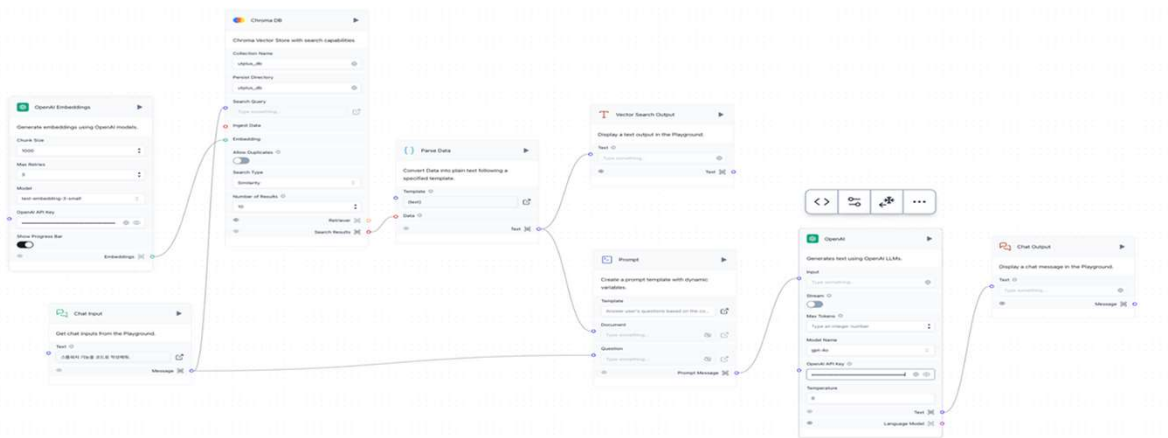
01	독창성	<ul style="list-style-type: none">• 자동 코드 생성• RAG 기반 자체 데이터 반영• 커스텀 코드 도움 기능
02	전문성	<ul style="list-style-type: none">• 오류 감지 및 수정• 자체 데이터 기반 코드 제공• 사용자 채팅 기능
03	수익가능성	<ul style="list-style-type: none">• 창의성 향상• 유료 기능 별도 제공• 자원 한도 제공으로 유료화
04	기술보유	<ul style="list-style-type: none">• 다양한 검색을 통한 검색 기능 고도화• 자체 임베딩 기능을 통한 검색 기능 제공• 자동화된 데이터 업데이트 기능
05	산업기여	<ul style="list-style-type: none">• 향상된 효율성 및 생산성• 코드 최적화• 혁신과 신기술 촉진

4. 코드 생성 워크플로우

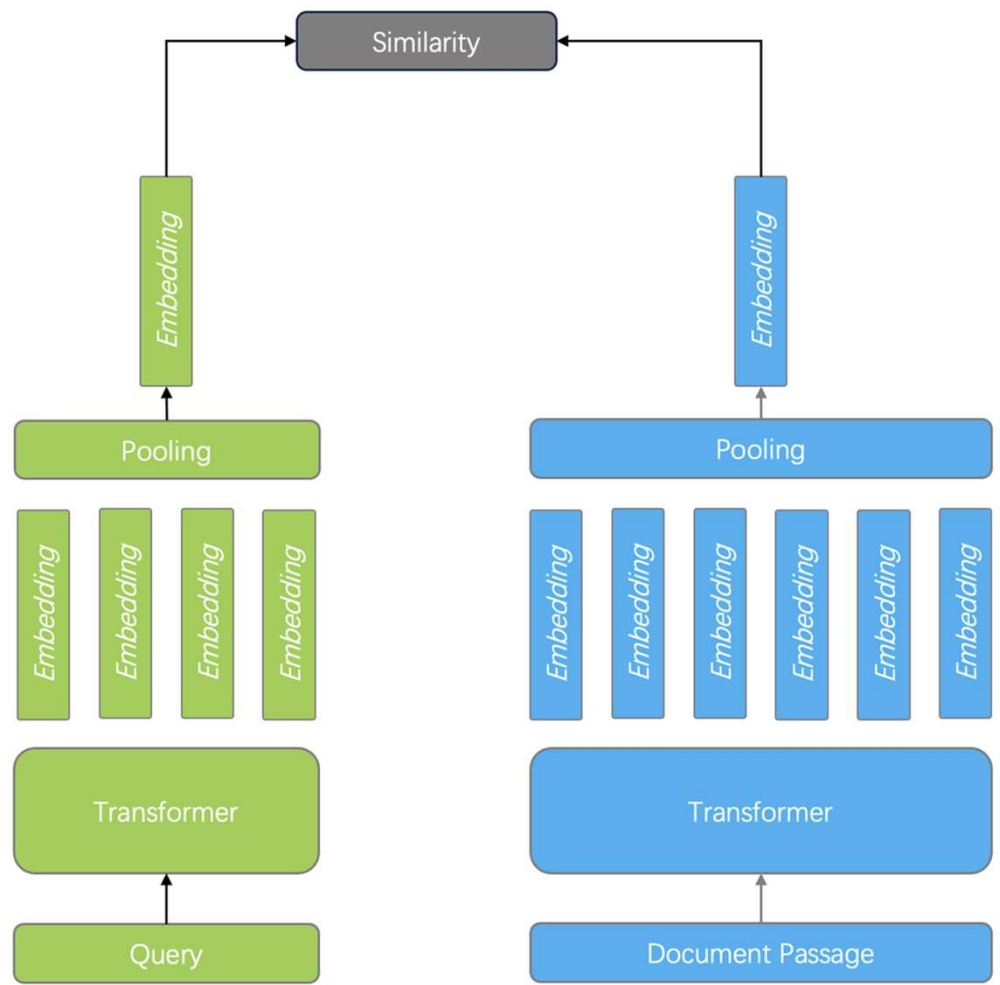
- 코드 데이터 생성
 - 벡터 데이터베이스 사용
 - 임베딩 모델 사용
 - 커스텀 노드 사용
 - 노드 워크플로우 구조
 - 벡터 DB 교체 가능
 - 임베딩 모델 교체 가능



- 코드 생성 기능
 - 벡터 데이터베이스 사용
 - LLM 모델 사용
 - 외부/내부 모델 사용
 - 노드 워크플로우 구조
 - LLM 모델 교체 가능
 - 리랭커 모델 교체 가능



5. 검색 기능 고도화



Dense search는 정보 검색(Information Retrieval) 분야에서 고차원 벡터를 활용하여 질의(Query)와 문서(Document) 간의 유사성을 계산하는 방법입니다. 주로 텍스트, 이미지, 오디오와 같은 비정형 데이터를 다룰 때 사용됩니다. 이 방법은 질의와 문서의 의미적 유사성을 평가하는 데 탁월하여 기존의 단어 기반 검색 방법보다 더 정교한 검색 결과를 제공할 수 있습니다.

핵심 개념

- 1. 임베딩(Embedding):
 - 텍스트 또는 문서를 고차원 벡터 공간으로 변환하는 과정입니다. 이 과정에서 단어의 의미를 반영하여 단어 또는 문서를 벡터로 표현합니다.
 - 대표적인 임베딩 기법으로는 Word2Vec, GloVe, BERT 등이 있습니다.
- 2. 유사도 계산:
 - 질의와 문서가 임베딩된 벡터 공간에서 서로 얼마나 가까운지를 측정하여 유사도를 계산합니다. 일반적으로 코사인 유사도(Cosine Similarity)를 사용합니다.
- 3. Dense Retriever:
 - 텍스트의 의미를 반영한 벡터를 생성하여 질의와 문서 간의 유사도를 계산하는 검색 모델입니다. BERT와 같은 딥러닝 모델을 활용하여 텍스트를 임베딩하고, 질의와 문서를 같은 벡터 공간에서 비교합니다.

5. 검색 기능 고도화

TF-IDF

TF-IDF is a measure of originality of a word by comparing the number of times a word appears in a doc with the number of docs the word appears in.

$$TF-IDF = TF(t, d) \times IDF(t)$$

Term frequency

Number of times term t appears in a doc, d

Inverse document frequency

$\log \frac{1 + n}{1 + df(d, t)}$

of documents

Document frequency of the term t

$$score(D, Q) = \sum_{i=1}^n IDF(q_i) * \frac{f(q_i, D) * (k_1 + 1)}{f(q_i, D) + k_1 * (1 - b + b * \frac{|D|}{avgdl})}$$

문서 D 에서 q_i 의 term frequency

문서 D 의 길이

파라미터

문서 집합의 평균 문서 길이

TF-IDF

TF-IDF는 전체 문서에서 빈출되는 단어의 중요도는 낮다고 판단하고, 특정 문서에서만 빈출되는 단어는 중요도가 높다고 판단합니다.

즉, TF-IDF 값은 단어의 중요도와 비례하는데, TF-IDF 값이 클수록 해당 단어의 중요도가 높은 것이죠.

예를 들어, '합니다', '에서', '가'와 같이 서술어나 조사와 같은 불용어는 대부분의 문서에서 나타나기 때문에 TF-IDF 값은 낮아집니다.

BM25

BM25 모델의 TF-IDF와의 차이점

1. BM25는 문서 A와 B의 검색어 빈도수가 같을 때 문서의 길이가 길수록 낮은 score를 가진다.

2. 다른 문서에 잘 등장하지 않는 단어 a를 포함한 문서는 a의 빈도수가 높지 않아도 높은 score를 가진다.

6. 서비스 실행

← Vector Search Output

Category: 클라
Prompt: 메쉬의 텍스처를 변경하는 코드를 알려줘
ResultLuaCode: Workspace.Cube.TextureID = 10579
OtherDescription:
Category: 클라
Prompt: 텍스트 위젯의 자동 줄바꿈을 변경하는 코드를 알려줘
ResultLuaCode: local text = Workspace.ScreenUI.Text
text.AutoWrapText = true
OtherDescription:
Category: 클라
Prompt: 위젯의 채우기를 변경하는 코드를 알려줘
ResultLuaCode: local Frame = Workspace.ScreenUI.Frame
Frame.Fill = true
OtherDescription:
Category: 클라
Prompt: 캐릭터 위치에 사운드를 생성하는 코드를 알려줘
ResultLuaCode: player:CreateSound(Workspace.Sound)
OtherDescription:
Category: 클라
Prompt: 사운드를 정지하는 코드를 알려줘
ResultLuaCode: Game:StopSound(Workspace.Sound)
OtherDescription:
Category: 클라
Prompt: 사운드를 재생하는 코드를 알려줘
ResultLuaCode: --플레이 할 사운드, 생성 위치
Game:PlaySound(Workspace.Sound, Vector.new(0, 0, 0))
OtherDescription:
Category: 클라



- 벡터 검색
- 채팅 실행
- 코드 생성

