SK네트웍스 Family AI 과정 20기

데이터 전처리 학습된 인공지능 모델

산출물 단계	데이터 전처리
평가 산출물	데이터 전처리 학습된 인공지능 모델
제출 일자	2026년 2월 2일
깃허브 경로	SKN20-FINAL-5TEAM
작성 팀원	김황현, 최소영

1. 모델 목적

- **목적**: 'Engineering Gym' 플랫폼의 핵심인 실무형 연습 문제(의사코드, 디버깅, 시스템 설계)를 생성하고, 사용자가 제출한 답변들을 분석하여 정밀한 기술 평가 및 개인별 맞춤형 코칭 피드백을 제공하는 것을 목적으로 합니다. 단순히 정답 여부를 가리는 것을 넘어, 사용자가 겪는 논리적 오류를 탐지하고 에이전트를 통해 개선 방향을 제안하는 지능형 튜터 역할을 수행합니다.사용자의 해결 능력을 평가하기 위한 표준화된 데이터셋 및 가이드라인 구축.
- **핵심 기능**: 비정형 기술 지식(기술 블로그 등)을 AI 엔지니어의 관점으로 재해석하여 단계별(Step 1~5) 학습 콘텐츠로 변환.

2. 모델 아키텍처 설계

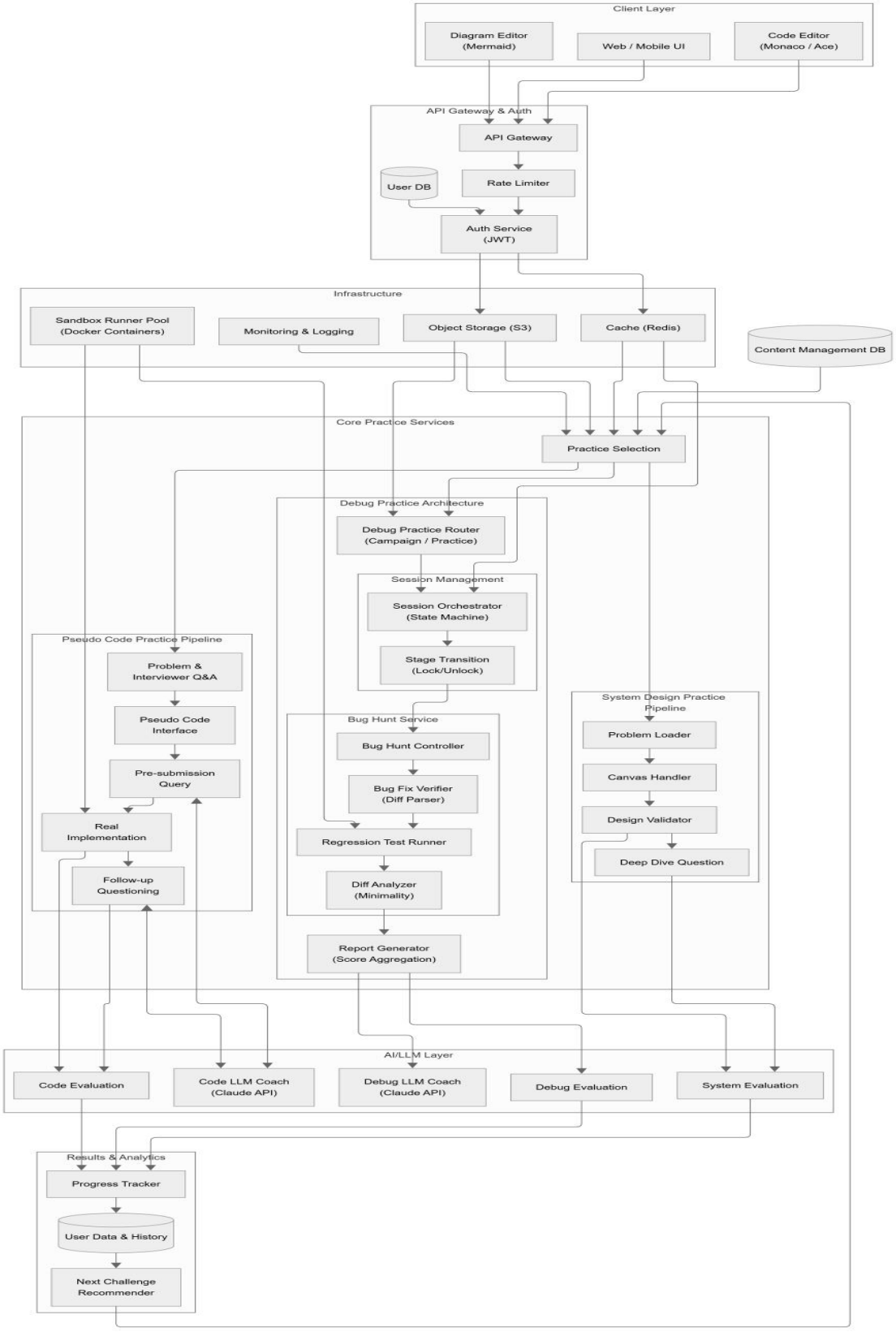
2.1 선정 모델: GPT-4o-mini 하이브리드 에이전트

- **이유**: 실시간 코드 실행 엔진(Pyodide)과 유기적으로 통신하며, 사용자의 의도를 파악하는 인스트럭션 튜닝 성능이 가장 우수하여 선정하였습니다.

2.2 아키텍처 개요

계층	구성 요소	역할
입력층	User Response Parser	사용자가 작성한 텍스트 및 아키텍처 JSON 데이터를 파싱
전처리층	Semantic Validator	한글 형태소 분석 및 불용어 제거를 통해 핵심 논리 키워드 추출
평가층	Evaluation Engine	ai_view.py 내 정의된 시스템 프롬프트에 따른 정량적/정성적 채점
출력층	Feedback Generator	채점 결과와 연동된 '꼬리 질문' 및 맞춤형 힌트 제공

2.3 아키텍처 시각화



2.4 설계 근거

● **시맨틱 매칭 도입**: 초보 개발자가 사용하는 다양한 한글 표현(예: "넣는다", "추가한다",

"append")을 동일한 논리 단위로 인식하여 오채점을 방지함.

- **비동기 처리 구조**: 사용자 제출 후 대기 시간을 최소화하기 위해 백엔드와 AI 레이어 간 비동기 API 통신을 최적화함.

3. 모델 활용 및 최적화 요약

직접적인 가중치 학습 대신, 모델이 우리가 원하는 결과를 내도록 하는 프롬프트 최적화(Prompt Optimization) 과정을 수행했습니다.

- **프롬프트 기법**:
  - Role Playing: "너는 AI 엔지니어다"라는 역할을 부여하여 실무 밀착형 문제 생성 유도.
  - Few-shot: 모델이 JSON 형식을 깨뜨리지 않도록 실제 서비스에서 사용되는 stage.js나 test.json의 형식을 예시로 제공.
- **성능 평가 지표**:
  - 데이터 일관성: 모든 문제가 1~5단계의 점진적 난이도를 유지하는지 확인.
  - 형식 정합성: 생성된 데이터가 추가 작업 없이 프론트엔드 코드에 바로 적용 가능한지 검증.

4. 저장 및 배포

4.1 문제 저장

- **저장 방식**: 시스템 성능 최적화를 위해 생성된 데이터를 JSON 형태의 정적 파일로 저장하여 관리.
- **주요 파일명**:
  - stage.js: Logic Mirror 트랙용 의사코드 문제 데이터
  - progressive-problem.json: Bug Detective 트랙용 디버깅 문제 데이터
  - test.json: 시스템 아키텍처 설계 및 평가 루브릭 데이터

4.2 인프라 및 환경 정보

- **데이터베이스**: PostgreSQL (reset_db.sql을 통한 초기화)를 사용하여 사용자 학습 이력 및 평가 점수 영구 저장.

- **배포 방식**: Docker Hub 기반의 컨테이너 배포 체계를 구축하였습니다.
  프론트엔드(Nginx/Vue.js)와 백엔드(Django)를 Docker 컨테이너로 각각 독립화하고,
  Docker Compose를 통해 환경 변수

4.3 저장 및 호출 형식

항목	설명
저장 파일명	pseudo_eval_engine_v2.json (Prompt & Logic Bundle)
저장 형식	JSON-based Logic Structure & API Interaction Wrapper
모델 로드 예시	axios.post('/api/core/ai-evaluate/', { quest_title, user_logic })

4.4 모델 사양 요구 사항

- **프레임워크**: Django REST Framework (Backend) + Vue.js (Frontend)
- **환경 설정**: docker-compose.yml을 통해 API Server, DB, Frontend 서비스 일괄 관리
- **네트워크**: 외부 AI API(OpenAI) 호출을 위한 Outbound 트래픽 허용 필요

5. 종합 평가 및 활용 방안

- **모델 안정성**: 서버 사이드 렌더링과 독립된 AI 레이어를 구축하여 높은 가용성 확인.
- **일반화 가능성**: 신규 퀘스트 추가 시 stages.js의 메타데이터 수정만으로 즉시 확장 가능.
- **향후 활용**:
  - 사용자 성장 로그 분석을 통한 개인화된 '약점 극복 리포트' 자동 생성.
  - 기업용 '엔지니어링 코딩 테스트' 모듈로의 B2B 솔루션 확장.

6. 결론 및 향후 계획

- **결론**: 별도의 고비용 모델 학습 없이도, 정교한 프롬프트 설계를 통해 전문적인 엔지니어링 교육 콘텐츠를 확보할 수 있음을 입증함.
- **향후 계획**:
  - 생성된 문제 데이터셋을 기반으로, 추후 사용자의 답변 패턴을 분석하여 개인별 취약점을 보완하는 추천 모델로 고도화 예정.
  - 데이터 보안을 위해 모든 데이터에 Soft Delete(use_yn) 및 암호화 정책 유지.