

추가 가이드(common)

폴더: Student/common/

AI 기반 리서치 또는 검색 결과를 단계별로 처리·병합·렌더링하는 **파이프라인용 유틸리티 코드**

- schemas.py: 데이터 형태 정의
- merge.py: 결과 병합
- writer.py: 시각적 리포트 작성
- fs_utils.py: 결과물 저장

단계	주요 파일	기능 요약	결과물
1	schemas.py	데이터 구조 및 계획 정의	Plan 및 모델
2	merge.py	웹 결과 + RAG 결과 병합	JSON 병합 결과
3	writer.py	병합 결과를 Markdown 리포트로 변환	마크다운 텍스트
4	fs_utils.py	Markdown 리포트 저장	.md 파일

추가 가이드(common)

폴더: Student/common/

1. fs_utils.py — 파일 저장 유틸리티

- 주요 기능: 결과물(markdown 리포트)을 자동으로 저장하는 도구입니다.
- 핵심 함수:
 - `_slugify(text)` : 파일 이름으로 쓸 수 있도록 문자열을 정제(공백 제거, 특수문자 필터링).
 - `save_markdown(query, route, markdown)` :
 - data/processed 폴더를 만들고,
 - 쿼리·라우트 정보를 조합한 파일명(날짜__route__slug.md)으로 markdown을 저장.
- 용도: **Day1~3 단계별 결과를 markdown 형태로 남기는 기능을 지원합니다.**
 → 즉, “리서치 결과 파일을 자동으로 기록하고 관리”하기 위한 기본 유틸입니다

추가 가이드(common)

폴더: Student/common/

2. merge.py — 웹 검색 결과와 RAG 결과 병합 로직

- 주요 기능: “웹 검색(Day1)”과 “RAG 기반 요약(Day2)” 결과를 정책에 따라 병합.
- 핵심 역할:
 - 웹 결과(web_payload)와 RAG 결과(rag_payload)를 입력받아,
 - 신뢰도(confidence) 계산,
 - 게이팅(gating) 통과 여부 확인,
 - 최종 병합 전략(strategy) 결정:
 - web_only, rag_only, web_plus_rag.
 - _web_confidence, _rag_confidence 함수를 통해 점수화.
 - merge_day1_day2()에서 두 결과를 통합한 JSON 스키마(merged_day1_day2) 생성.
- 용도:
 - “웹검색+RAG” 병렬 실행 결과를 하나로 합치는 중앙 병합기(core merger).
 - 예: “RAG 신뢰 낮으면 웹결과만, 신뢰 충분하면 둘 다 표시” 같은 정책 처리

추가 가이드(common)

폴더: Student/common/

3. schemas.py — 데이터 구조 정의

- 주요 기능: Day1~Day3 처리 과정에서 사용되는 입력/출력 데이터 스키마 정의.
- 내용 요약:
 - Day1Plan, WebResultItem: 웹 검색 관련 설정 및 결과 구조.
 - Day2Plan, RagContextItem: RAG 검색/요약 관련 설정.
 - Day3Plan, GovNoticeItem: 정부지원사업/공고 데이터 구조.
 - GovNoticeItemModel, GovNoticesModel: pydantic 모델로 API용 검증 가능.
- 용도:
 - 각 단계(Day1~Day3)의 파이프라인에서 JSON 형태로 데이터를 주고받을 때 일관된 구조와 타입 검증을 보장

추가 가이드(common)

폴더: Student/common/

4. writer.py — 마크다운 리포트 생성기

- 주요 기능: 검색 결과나 분석 결과를 마크다운 형태의 리포트로 렌더링.
- 주요 함수:
 - render_day1(): 웹 리서치 보고서(시세, 기업 정보, 관련 기사 등).
 - render_day2(): RAG 요약 및 근거(contexts) 테이블.
 - render_day3(): 정부 공고 리스트 및 첨부 요약.
 - render_enveloped(): 메타데이터(header/footer)를 포함한 완성본 생성.
- 용도:
 - 병합된 결과를 사람이 읽기 좋은 Markdown 문서로 변환.
 - fs_utils.save_markdown()과 함께 사용하여 자동 저장까지 담당