ChatGPT

퀄리저널 (QualiNews Keyword News) 프로젝트 인수인계 문서

프로젝트 개요 및 목표

퀄리저널은 **하루 하나의 키워드**를 중심으로 여러 출처의 콘텐츠를 수집하여 큐레이션 형태의 뉴스를 발행하는 시스템입니다. 편집자가 특정 키워드를 입력하면 해당 키워드와 관련된 **공식 뉴스, 학술 논문, 표준 문서, 기업 블로그, 커뮤니티 글** 등을 폭넓게 수집하고, 그 중 **최소 15건 이상의 고품질 기사 카드**를 선정하여 특별 호 형태로 발행하는 것이 목표입니다 [1]. 각 기사 카드는 제목, 발행일, 간략 요약, 출처 등의 정보를 담고 있으며, 키워드와 연관된 과거부터 최신까지의 내용을 아우릅니다.

본 프로젝트는 한 키워드에 대한 **역사적 맥락과 최신 동향**을 동시에 보여줄 수 있도록 설계되었습니다. 기사 선정 시 다양한 유형과 관점을 포함해 **정보 제공, 논쟁거리, 전문가 평가, 제품 소개, 커뮤니티 토론** 등 다각적인 콘텐츠를 담도록 합니다 [2]. 수집된 모든 기사는 JSON 형식으로 저장·정리되고, 편집자가 내용을 검토하여 승인한 후 Markdown/HTML 페이지로 최종 발행됩니다.

작업 루트 경로

프로젝트 파일은 한 폴더에 모여 있으며, 작업을 진행할 때 해당 폴더를 **루트 디렉토리**로 사용합니다. 예시 경로 (Windows): `C:\Users\userW\Desktop\퀄리저널`. 이 디렉토리를 기준으로 아래에 소개하는 파일들과 폴더 구조가 구성되어 있습니다.

핵심 파일 및 폴더 구조

프로젝트의 주요 파일과 폴더 구조는 다음과 같으며, 각 파일의 역할은 아래와 같습니다:

- `orchestrator.py` – 전체 수집부터 발행까지의 과정을 orchestration하는 진입점 스크립트입니다. 이 스크립트를 통해 키워드별 기사 **수집** 및 **발행** 작업을 실행합니다.
- `engine_core.py` – 핵심 엔진 모듈로, 기사 **수집**, **평가(스코어링)**, **요약/번역**, **발행** 등의 세부 기능을 담당합니다. 각 단계별 로직이 구현되어 있으며 orchestrator가 이를 호출합니다.
- `config.json` – 품질 기준 및 크롤링 설정을 담은 구성 파일입니다. **품질 게이트(QG)** 설정, RSS 사용 여부, 크롤링 파라미터 등이 정의되어 있습니다. 예를 들어 커뮤니티 글 필터 기준이나 외부 뉴스 검색 사용 여부 등이 포함됩니다.
- `editor_rules.json` – 편집/평가 규칙이 정의된 파일입니다. 도메인별 기본 **가중치(점수)**, 특정 키워드에 대한 추가 점수, 주요 편집 가이드라인, 기본 **editor_note** 템플릿 등이 들어 있습니다. (예: 신뢰할 만한 도메인 목록과 그 가중치, 중요한 키워드에 대한 점수 부여 등)
- `feeds/official_sources.json` – 공식 소스들의 RSS 목록을 정의한 파일입니다. 산업 뉴스 사이트, 기술 블로그 등의 RSS 피드 URL과 메타정보를 포함합니다 (예: 각 소스의 URL, 명칭, 신뢰도 점수 등).
- `feeds/community_sources.json` – 커뮤니티/포럼 등 **비공식 소스** 목록 파일입니다. Reddit 서브레딧, 전문 포럼 등의 API 엔드포인트나 RSS 피드, 크롤링 URL을 정의하고, 커뮤니티 특유의 필터 조건(예: 최소 추천수, 댓글수 등)을 설정합니다.
- `data/selected_keyword_articles.json` – 키워드 수집 결과가 저장되는 JSON 파일입니다. 특정 키워드에 대해 수집된 기사들의 메타데이터, 요약, 점수 등이 이 파일에 기록됩니다. 편집자가 어떤 기사를 선택(approve)했고 어떤 코멘트를 달았는지도 이 JSON에 반영됩니다.

실행 예시 (명령어)

퀄리저널 프로젝트는 CLI를 통해 수집 및 발행을 수행합니다. 아래는 사용 예시입니다:

- **키워드 기사 수집**: 원하는 키워드에 대한 뉴스를 수집하려면 다음과 같이 실행합니다.

```
python orchestrator.py --collect-keyword "IPC-A-610"
```

위 명령은 `"IPC-A-610"` 키워드와 관련된 모든 소스의 최신 기사를 크롤링하고, 필터링 및 평가를 거쳐 `selected_keyword_articles.json`에 후보 기사 목록을 저장합니다.

- **키워드 뉴스 발행**: 수집된 기사 중 편집자가 발행할 목록을 확정했다면, 다음 명령으로 발행합니다.

```
python orchestrator.py --publish-keyword "IPC-A-610"
```

이 명령은 `selected_keyword_articles.json` 내 **selected:true**로 표시된 기사들만 모아 Markdown/HTML 뉴스 페이지를 생성합니다. 생성된 결과물은 설정에 따라 지정된 폴더(예: `archive/키워드_YYYYMMDD.md/html`)에 저장됩니다.

편집자 승인 및 조건

수집 프로세스가 완료되면 편집자는 관리자 UI나 JSON 파일을 직접 편집하여 발행할 기사를 선택하게 됩니다. **발행 전 최소 15개 이상의 기사를 승인**(selected 필드를 true로 표시)해야 발행이 가능하도록 설계되어 있습니다 [3]. 만약 15개 미만을 선택하면 발행 단계에서 오류가 나거나 UI상 발행 버튼이 비활성화되도록 구현되어 있습니다. **15~20개 정도의 기사 선정**이 권장되며, 너무 많거나 적지 않도록 균형을 맞춥니다.

또한 각 승인된 기사에는 **editor_note(편집자 한줄 코멘트)**를 달 수 있습니다. 이 코멘트는 독자에게 해당 기사의 맥락이나 중요도를 전달하는 짧은 메모로, 필수는 아니지만 권장 사항입니다. UI에서 코멘트 입력란이 제공되며, 편집자가 코멘트를 남기지 않고 넘어가는 경우를 방지하기 위한 UX 장치(예: 빈 값일 때 경고나 placeholder 안내문구 등)도 고려되어 있습니다.

주의 사항 및 팁

- **본문 전체 저장 금지**: 수집된 기사들의 **원문 전체를 저장하지 않도록** 주의해야 합니다. 시스템은 제목, 요약, 메타데이터(출처, 날짜 등)만 저장하며, 본문 전문은 저장/노출하지 않습니다 [4]. 이는 데이터 용량 관리와 저작권 문제를 피하기 위한 것으로, 이미 구현된 수집 모듈들도 **링크와 요약만 보존**하도록 되어 있습니다. 새로 코드를 수정하거나 소스를 추가할 때도 이 원칙을 지켜주세요.
- **커뮤니티 필터 조정**: 커뮤니티 글의 경우 노이즈가 많으므로 `config.json`의 설정을 통해 필터링 기준을 관리합니다. 예를 들어 **키워드 포함 여부(require_keyword)**, 최소 키워드 토큰 수, 허용 도메인 목록, 제목 길이 제한, 차단할 제목 패턴(예: 홍보성 글) 등을 설정할 수 있습니다 [5]. 현재 기본값으로 Reddit 등에서는 키워드가 제목/내용에 반드시 포함되도록 하고, 채용, 할인 등의 단어가 들어간 글은 제외하는 등 규칙이 적용되어 있습니다. 필요에 따라 이 필터 기준을 완화하거나 조절하여 **커뮤니티 결과가 너무 적게 나오는 경우** 대응할 수 있습니다 [6].
- **외부 뉴스 API 사용**: Google News 등 **외부 뉴스 검색 API** 연동 기능이 존재하지만, 기본적으로는 **비활성화**되어 있습니다. `config.json`의 `external_rss.enabled` 값이 false로 설정되어 있으며 [7], 외부 API를 활용한 과거 기사 검색은 초기 구현에서는 제외되었습니다. 이 기능을 사용하려면 CLI 인자 (`--`

`external-rss` 등)나 설정을 통해 활성화해야 합니다. 활성화 시 Google News API/Bing News API 등을 통해 키워드와 관련된 과거 뉴스도 자동 수집할 수 있지만, 구현 복잡도가 올라가므로 **필요할 때만 한정적으로 사용**하도록 권장됩니다.

향후 TODO 및 권장 작업

- **Elasticsearch 연동** (선택 사항): 현재 키워드별 기사 데이터는 JSON 또는 간단한 DB로 저장됩니다. 추후 검색 최적화나 대용량 데이터 관리를 위해 **Elasticsearch**와의 연동을 고려할 수 있습니다. Elasticsearch를 도입하면 키워드 색인, 날짜별 조회, 전문 검색 등이 빨라지고 향후 **트렌드 분석** 기능도 구현하기 용이합니다.
- **관리자 UI 개선**: 키워드 뉴스 발행을 편리하게 하기 위해 **웹 기반 UI** 연동을 계획하는 것이 좋습니다. 예를 들어 Flask/FastAPI 백엔드에 키워드 수집/승인/발행을 위한 API 엔드포인트를 추가하고, React/Vue 등으로 프론트엔드 대시보드를 구성하여 **키워드 입력 -> 수집 트리거 -> 기사 검토 및 선택 -> 발행**까지 모두 웹에서 처리할 수 있게 할 수 있습니다 [8] [9]. 현재 `selected_keyword_articles.json`을 불러와 편집하는 기능, 체크박스로 기사 선택, 코멘트 입력 필드, 발행 버튼 클릭 시 API 호출 등 **관리자용 워크플로우**를 구현하는 것이 향후 과제입니다. 이를 통해 여러 편집자 협업, 상태 동기화, 발행 이력 관리 등이 수월해집니다.
- **용어집 및 동의어 활용**: 프로젝트에는 용어 일관성 유지를 위한 **glossary.json(용어집)**이나 키워드 **동의어 목록(keyword_synonyms.json)**을 활용할 수 있는 기반이 마련되어 있습니다. 예를 들어 요약/번역 단계에서 Glossary를 적용하여 전문 용어의 번역을 통일하고 [10] [11], 키워드 수집 단계에서 한글 키워드에 대한 영어 약어/동의어도 함께 검색하도록 지원할 수 있습니다. 앞으로 이 파일들을 잘 채워두면 (예: AI ↔ 인공지능, EV ↔ 전기차 등의 매핑) 수집 범위를 넓히고 번역 품질을 높이는 데 큰 도움이 될 것입니다.
- **기타 개선**: 그 외에도 코드 전반적인 **리팩토링**을 통해 안정성과 확장성을 높이고, Docker를 사용한 배포, cron/스케줄러를 통한 자동화, 에러 로그 관리, 테스트 케이스 작성 등의 작업이 권장됩니다. 또한 발행된 키워드 뉴스 페이지를 독자가 쉽게 접근하도록 **프론트엔드 개편**(예: 메인 사이트에 키워드 뉴스 섹션 추가, SEO 최적화)도 고려해볼 수 있습니다.

以上이 퀄리저널 프로젝트에 대한 개요와 핵심 인수인계 사항입니다. 새로운 참여자는 이 문서를 참고하여 환경을 설정하고, 주어진 구조와 규칙에 따라 개발을 이어가시면 됩니다. 궁금한 사항이나 추가 자료는 내부 위키 또는 이전 개발자에게 문의하시기 바랍니다. 성공적인 프로젝트 진행을 기원합니다! [1] [2]

---

[1] [3] [8] [9] 1003퀄리뉴스 _키워드 뉴스_ 기능 설계 및 구현 전략.pdf
file://file-XoVMv2VGBLyHnFVetrmoGx

[2] 1003_2A역사 큐레이션형 키워드 뉴스 포맷을 실현하려면.pdf
file://file-495Xc4vLRe2iNSrwexdNLY

[4] 1003_3A 하루 한 키워드” 역사 큐레이션형 퀄리뉴스를 구축하기 위한 구체적 개발 가이드입니다.pdf
file://file-92Q3asQNjsGDWWy73Fkbpt

[5] [7] config.json
file://file_00000000cc0861f5bbcd2b62fa17db80

[6] 1003_1A퀄리 뉴스 제안.pdf
file://file-9ckyN6UsWeG3crtkDBreya

[10] [11] 1003_5A퀄리뉴스를 전면적으로 리팩터링하여 안정적이고 확장 가능한 시스템으로 구축하려면.pdf
file://file-C1X8pMSvAVd6Xs392FEnwq