퀄리저널 프로젝트의 기술적 목표 및 우선순위

1. 프로젝트 개요 및 기본 목표

퀄리저널(QualiNews Keyword News) 프로젝트는 **“하루 하나의 키워드 뉴스”**를 발행하는 큐레이션 시스템이다. 편집자가 특정 키워드를 입력하면 시스템이 공식 뉴스, 학술 논문, 표준 문서, 기업 블로그, 커뮤니티 글 등 다양한 출처에서 내용을 수집한다【569795938803306†L3-L11】. 키워드와 관련된 기사 중 **최소 15 건 이상의 고품질 기사 카드를 선별**하여 발행하는 것이 목표로, 각 기사 카드는 제목·발행일·간략 요약·출처를 포함하고 과거와 최신 동향을 모두 다룬다【569795938803306†L3-L10】.

프로젝트는 JSON 형식으로 데이터를 저장한 뒤 편집자가 수집 결과를 검토-승인하고 Markdown/HTML 페이지로 최종 발행하는 구조이며【569795938803306†L12-L13】, 이를 통해 한 키워드에 대한 역사적 맥락과 최신 정보를 동시에 전달한다.

2. 핵심 구성 요소 및 작업 구조

프로젝트의 작업 루트는 하나의 디렉터리 안에 있으며, 주요 파일 및 폴더는 다음과 같다【569795938803306†L19-L37】:

구성 요소	역할 및 설명
orchestrator.py	기사 수집부터 발행까지의 전체 과정을 orchestration 하는 진입점 스크립트이다【569795938803306†L20-L23】. CLI 인자를 통해 키워드 수집 및 발행을 실행한다.
engine_core.py	기사 수집·평가(스코어링)·요약/번역·발행 등 세부 기능을 담당하는 핵심 모듈이다【569795938803306†L20-L23】.
config.json	품질 기준 및 크롤링 설정을 정의한다. 품질 게이트(QG) 설정, RSS 사용 여부, 커뮤니티 필터 등의 파라미터가 포함된다【569795938803306†L24-L26】.
editor_rules.json	도메인별 기본 가중치, 특정 키워드에 대한 추가 점수 등 편집·평가 규칙을 담는다【569795938803306†L27-L29】.
feeds/official_sources.json	산업 뉴스 사이트, 기술 블로그 등 공식 소스의 RSS 목록과 각 소스의 신뢰도 점수 등을 포함한다【569795938803306†L30-L31】.
feeds/community_sources.json	Reddit 서브레딧, 전문 포럼 등 커뮤니티 소스 목록과 필터 조건(예: 최소 추천수, 댓글수)을 정의한다【569795938803306†L32-L34】.
data/selected_keyword_articles.json	특정 키워드에 대한 수집 결과가 저장되는 JSON 파일로, 기사 메타데이터·요약·점수와 편

구성 요소	역할 및 설명
	집자가 선택 여부 및 코멘트를 기록한다【569795938803306†L35-L37】.

CLI 활용 예시

프로젝트는 명령줄 인터페이스를 통해 동작한다. 특정 키워드에 대한 뉴스를 수집하려면 다음 명령을 실행한다:

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

위 명령은 해당 키워드와 관련된 최신 기사를 크롤링하고 필터·평가를 거쳐 selected_keyword_articles.json 에 후보 기사 목록을 저장한다【569795938803306†L50-L55】. 편집자가 발행할 기사를 선택한 후에는 다음과 같이 발행한다:

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

이 명령은 selected_keyword_articles.json 에서 selected: true 로 표시된 기사만 모아 Markdown/HTML 뉴스 페이지를 생성하고 archive/키워드_YYYYMMDD.md 또는 .html 파일에 저장한다【569795938803306†L55-L59】.

3. 기능 단계별 기술 목표

3.1 키워드 기반 기사 수집 단계

1. **키워드 정의 및 동의어 확장** – 사용자가 입력한 키워드의 언어·약어·동의어를 keyword_synonyms.json 과 glossary 로 확장하여 수집 범위를 넓힌다. 향후 추가할 용어집과 동의어 목록은 번역 품질과 검색 범위를 향상시킬 수 있다【569795938803306†L114-L119】.

2. **공식 소스 수집** – official_sources.json 에 정의된 RSS 피드를 크롤링하여 공식 뉴스·블로그·표준 문서 등을 가져온다. 신뢰도 점수가 높은 도메인을 우선적으로 사용하며, 필요 시 external_rss.enabled 를 true 로 설정하고 외부 뉴스 API(Google News API 등)를 연동해 과거 뉴스까지 확장할 수 있다【569795938803306†L100-L104】.

3. **커뮤니티 수집** – Reddit 서브레딧 및 전문 포럼에서 키워드 관련 게시물을 가져온다. community_sources.json 에서 최소 추천수·댓글수, 키워드 포함 여부, 허용 도메인, 제목 길이 제한 등 필터를 설정해 노이즈를 줄인다【569795938803306†L76-L81】. 커뮤니티 글은 제목과 내용에 키워드가 포함되어야 하며, 채용·할인 등 홍보성 패턴은 제외한다【569795938803306†L76-L81】.

4. **자료 저장** – 수집된 모든 기사는 원문 전체를 저장하지 않고 제목·요약·메타데이터(출처, 날짜 등)만 저장하여 데이터 용량과 저작권 문제를 줄인다【569795938803306†L71-L75】. 데이터는 JSON 포맷으로 selected_keyword_articles.json 에 기록된다【569795938803306†L35-L37】.

3.2 기사 평가 및 스코어링 단계

1. **도메인 가중치 적용** – editor_rules.json 에 정의된 도메인별 기본 가중치를 활용해 신뢰할 수 있는 출처에 높은 점수를 부여한다. 예를 들어 IPC 나 ECSS 도메인에 높은 가중치가 부여되고, 기술 블로그와 AI 관련 출처에도 점수 차등을 둔다【569795938803306†L27-L29】.

2. **커뮤니티 필터링** – config 파일의 community.filters 설정을 통해 키워드 토큰 수 (예: 두 개 이상의 토큰), 정규식 매칭, 허용 도메인, 최소 제목 길이, 차단 패턴 등을 조정할 수 있다【569795938803306†L76-L81】. Reddit 게시물은 최소 추천수·댓글수와 같은 추가 조건도 만족해야 한다【569795938803306†L32-L34】.

3. **점수 계산 및 랭킹** – 엔진은 키워드 일치도·도메인 가중치·인기 지표(업보트 수, 조회수 등)를 기준으로 기사 점수를 계산하고 상위 기사들을 선정한다. 선정 기준은 score_weights 및 norms 파라미터를 통해 튜닝할 수 있다.

3.3 요약 및 번역 단계

1. **요약** – 선정된 기사의 본문을 요약하여 제목과 함께 2–3 문장 이내로 압축한다. 요약 시 핵심 정보(사건·결과·기술 의미)를 전달하도록 한다.

2. **번역 및 용어 일관성** – 기사 원문이 영어 등 외국어인 경우 한국어로 번역한다. 번역 시 용어집(glossary.json)을 적용하여 전문 용어의 번역을 통일하고, 동의어 목록을 사용해 검색과 번역 품질을 높인다【569795938803306†L114-L119】.

3.4 편집자 승인 단계

1. **기사 선택** – 수집·평가된 기사 목록에서 편집자가 발행할 기사(최소 15 개)를 선택한다【569795938803306†L60-L65】. 15 개 미만을 선택하면 발행 단계에서 오류가 발생하거나 UI 상 발행 버튼이 비활성화된다【569795938803306†L60-L65】.

2. **편집자 코멘트 추가** – 승인된 각 기사에 대해 편집자는 한 줄짜리 editor_note 를 추가할 수 있으며, 이는 독자에게 맥락이나 중요도를 전달하는 메모 역할을 한다【569795938803306†L66-L68】. 빈 코멘트는 경고를 띄우는 등 UX 설계로 입력을 유도한다【569795938803306†L66-L69】.

3.5 발행 및 저장 단계

1. **발행 실행** – CLI 명령 python orchestrator.py --publish-keyword "키워드"를 사용하여 발행한다【569795938803306†L55-L59】. 엔진은 selected:true 로 표시된 기사들을 모아 Markdown/HTML 페이지를 생성한다.

2. **저장 경로** – 생성된 뉴스 페이지는 archive/키워드_YYYYMMDD.md 및 .html 파일에 저장되며, 팀 내부 또는 공개 웹사이트에 업로드할 수 있다【569795938803306†L55-L59】.

4. 향후 기술적 개선 목표 (장기 로드맵)

프로젝트 인수인계 문서에서는 다음과 같은 향후 TODO 를 제시한다:

1. **검색 엔진(Elasticsearch) 연동** – 현재는 JSON 또는 간단한 DB 에 기사를 저장하지만, 검색 최적화와 대용량 데이터 관리를 위해 Elasticsearch 연동을 고려한다

【569795938803306†L103-L107】. 이를 통해 키워드 색인, 날짜별 조회, 전문 검색 속도가 개선되고 트렌드 분석 기능 구현이 쉬워진다【569795938803306†L104-L106】.

2. **관리자 UI 개선** – 키워드 뉴스 발행을 더 편리하게 하기 위해 웹 기반 UI 연동이 필요하다. 예를 들어 Flask/FastAPI 백엔드에 수집/승인/발행을 위한 API 엔드포인트를 제공하고, React/Vue 프론트엔드로 키워드 입력→수집 트리거→기사 검토 및 선택→발행까지 전 과정을 웹에서 처리하도록 한다【569795938803306†L107-L113】. 이를 통해 여러 편집자의 협업, 상태 동기화, 발행 이력 관리가 가능해진다【569795938803306†L111-L113】.

3. **용어집 및 동의어 시스템 구축** – glossary.json 과 keyword_synonyms.json 을 충실히 채워 전문 용어 번역 통일과 키워드 수집 범위 확장을 도모한다. AI↔인공지능, EV↔전기차처럼 주요 용어의 매핑을 늘리면 수집 범위와 번역 품질이 크게 향상된다【569795938803306†L114-L119】.

4. **코드 리팩터링 및 자동화** – 엔진 전반을 리팩터링하여 안정성과 확장성을 높이고 Docker 컨테이너를 통한 배포, cron/스케줄러를 통한 자동화, 에러 로그 관리, 테스트 케이스 작성 등을 진행한다【569795938803306†L120-L123】. 또한 SEO 최적화를 고려한 프런트엔드 개편과 키워드 뉴스 섹션 추가도 권장된다【569795938803306†L121-L123】.

5. 단기 우선순위 및 즉시 실행 과제

다음 과제들은 현재 프로젝트 상태를 빠르게 개선하기 위해 우선적으로 수행해야 한다. 우선순위는 중요도와 시급성을 기준으로 나열하였다.

1. **환경 설정 및 데이터 소스 업데이트**
   - 작업 루트 디렉터리를 정확히 설정하고 주요 스크립트(orchestrator.py, engine_core.py)가 정상 동작하는지 확인한다.
   - official_sources.json 과 community_sources.json 의 RSS 피드와 API 엔드포인트를 최신 상태로 업데이트한다. 공식 소스는 IPC, ECSS, NASA 등 사용자 관심 분야의 뉴스를 충실히 포함해야 한다.
   - config.json 의 커뮤니티 필터를 검토하여 키워드 토큰 수, 허용 도메인, 차단 패턴 등이 실제 수집 목표에 적합한지 조정한다【569795938803306†L76-L81】.
2. **키워드 동의어 및 용어집 구축**
   - keyword_synonyms.json 에 핵심 키워드의 영어 약어·한글 표현·스페이스 버전 등을 체계적으로 등록한다. 이는 키워드 수집 단계에서 더 많은 관련 기사를 확보하고 번역 일관성을 높이는 데 필수적이다【569795938803306†L114-L119】.
   - glossary.json 을 신규로 작성하여 J-STD-001, IPC-A-610 등 표준 용어의 한글 번역을 정리하고, 번역 단계에서 일관되게 적용한다.
3. **스코어링 및 필터 튜닝**
   - editor_rules.json 의 도메인 가중치를 검토하고, 전자 생산·우주 기준 등 신뢰도가 중요한 도메인에 적절한 가중치를 부여한다【569795938803306†L27-L29】.
   - config.json 의 community.filters 를 테스트하면서 최소 제목 길이·키워드 정규식·차단 패턴이 실제 노이즈를 얼마나 줄이는지 검증한다【569795938803306†L76-L81】. 필요하면 가중치와 임계값을 조정한다.
4. **편집자 워크플로우 개선**

- 초기 단계에서는 JSON 파일을 직접 편집하여 기사를 선택할 수 있지만, 빠른 작업을 위해 간단한 스크립트나 작은 웹 UI 를 만들어 selected_keyword_articles.json 을 로드·수정·저장하는 기능을 제공한다.
- 기사 선택 시 15 개 이상을 반드시 선택하도록 유효성 검사를 추가하고, editor_note 를 입력하지 않은 경우 경고 메시지를 표시하는 등 기본적인 UX 를 구현한다【569795938803306†L60-L69】.

5. **발행 및 배포 자동화**
   - 발행 스크립트를 cron 잡이나 CI 파이프라인에 설정하여 지정된 시각에 자동으로 뉴스가 발행되도록 한다. 발행된 Markdown/HTML 파일을 웹사이트나 인트라넷에 자동 배포하는 절차를 확립한다.
   - Docker 컨테이너로 환경을 배포하면 팀원이 쉽게 실행 환경을 재현할 수 있으며, 오류 로그 및 모니터링 설정도 용이하다【569795938803306†L120-L123】.
6. **장기 개선 계획 수립**
   - Elasticsearch 연동과 관리자 UI 구축 같은 장기 목표를 로드맵에 명시하고, 필요 인력·일정·기술 스택을 계획한다【569795938803306†L103-L113】.
   - 코드 품질 향상을 위해 리팩터링·테스트 케이스 작성·모듈화 등을 지속적으로 진행한다【569795938803306†L120-L123】.

6. 결론

퀄리저널 프로젝트는 신뢰성 있는 공식 소스와 커뮤니티 자료를 함께 활용해 특정 키워드에 대한 깊이 있는 뉴스 레터를 매일 발행하는 것을 목표로 한다. 인수인계 문서에서 제시한 구조와 규칙을 따르면, 새로운 팀원도 빠르게 환경을 이해하고 운영을 이어갈 수 있다. 단기적으로는 환경 설정, 데이터 소스 업데이트, 스코어링 튜닝, 편집자 워크플로우 개선을 우선 수행해야 하며, 장기적으로는 Elasticsearch 연동, 관리자 UI 구축, 용어집 시스템 강화, 코드 리팩터링 등을 통해 시스템의 확장성과 품질을 높이는 것이 바람직하다.