Part 3. 멀티 LLM 기반 AI 에이전트 개발 프로젝트

솔트룩스 x 이코노미스트 x KG제로인 협력 프로젝트

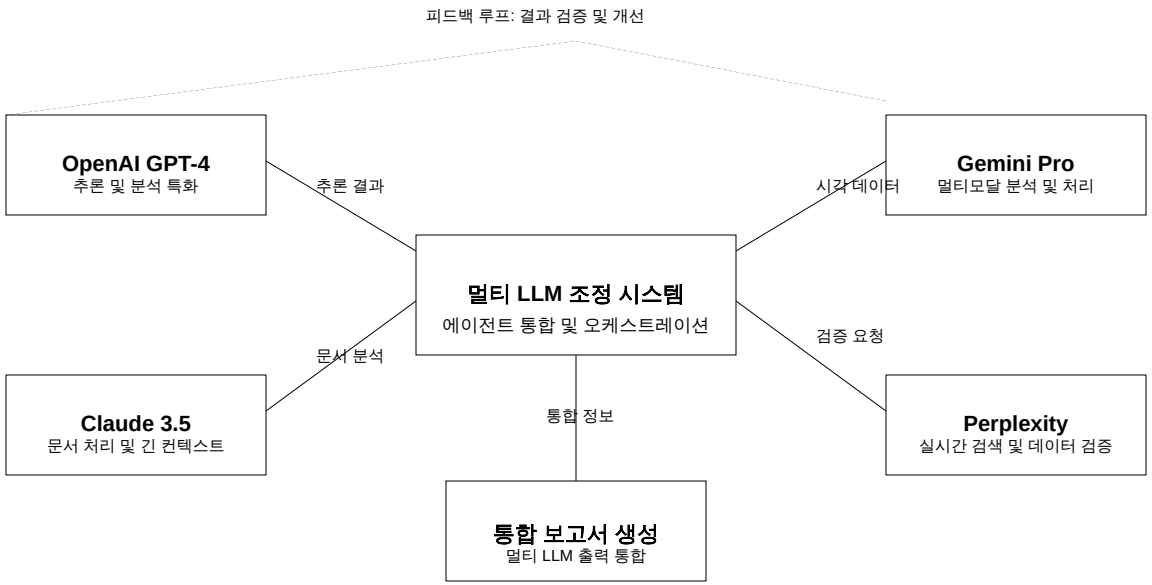
솔트룩스 **이코노미스트** **KG제로인**

2025년 7월

멀티 LLM 아키텍처 구성도

- **데이터 레이어**: 금융 데이터, 기업 공시, 뉴스 및 KG제로인 API를 통합하여 OpenAI, Claude, Gemini, Perplexity 연동
- **LLM 통합 레이어**: OpenAI(GPT-4)의 추론 능력, Claude의 긴 컨텍스트 처리, Gemini의 멀티모달 분석, Perplexity의 실시간 검색 기능 통합
- **AI 에이전트 레이어**: 멀티 LLM 기반 평가·리포트·어워드 전문 에이전트로 애널리스트 평가 자동화
- **서비스 레이어**: 사용자 친화적인 인터페이스로 AI 애널리스트 평가 결과와 어워드 관리 지원

멀티 LLM 기반 AI 에이전트 간 상호작용 구조



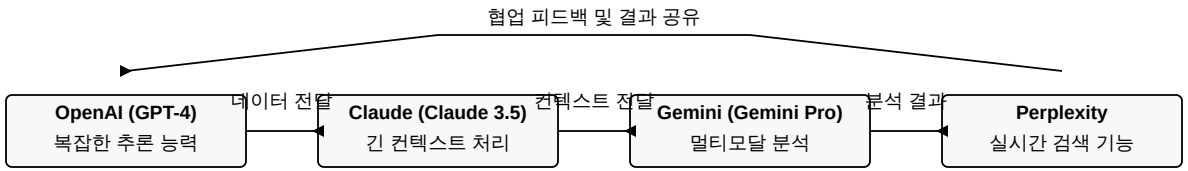
- **멀티 LLM 협업 구조**: 중앙집중형 + 분산협업 하이브리드 방식
  - 각 LLM이 특화된 영역에서 독립적으로 작업 수행
  - 중앙 조정 시스템이 작업 할당 및 결과 통합 역할 수행
  - 모델 강점 활용: 추론(GPT-4), 문서처리(Claude), 멀티모달(Gemini), 검색(Perplexity)
- **정보 흐름 특성**: 양방향 순환형 정보 처리
  - 실시간 데이터와 과거 데이터의 동적 통합 처리
  - LLM 간 교차 검증으로 정확도 향상 및 편향성 감소

- **LLM 간 상호작용 방식**
  - API 기반 표준화된 프롬프트 및 데이터 교환
  - ReAct 패턴과 Chain of Thought 방식의 협업 추론
  - 멀티모달 정보 통합: 텍스트, 수치, 차트 통합 분석
- **멀티 LLM 활용 이점**
  - 각 모델의 특화된 강점 활용으로 종합적 분석력 향상
  - 단일 LLM의 한계 보완 및 편향성 상호 검증
  - 실패 허용 시스템: 단일 모델 장애 시에도 작동 지속

멀티 LLM 기반 AI 에이전트 특화 기능

- **OpenAI (GPT-4)**: 복잡한 추론 능력, 정교한 논리 전개, 다단계 문제 해결, 금융 데이터 분석 및 해석
- **Claude (Claude 3.5)**: 긴 컨텍스트 처리(최대 200K 토큰), 대용량 리포트 분석, 상세한 문서 요약, 윤리적 판단
- **Gemini (Gemini Pro)**: 멀티모달 분석, 차트/그래프 인식, 이미지 데이터 해석, 시각 자료 기반 금융 분석
- **Perplexity**: 실시간 검색 기반 분석, 최신 금융 정보 반영, 시장 데이터 연동, 팩트 체킹 및 검증

멀티 LLM 간 상호작용 구조



- **협업 방식**: 순차적 처리와 병렬 처리를 조합한 하이브리드 방식
  - **복합 추론**: GPT-4의 추론 + Claude의 컨텍스트 능력 결합
  - **시각 분석**: Gemini의 시각 자료 해석 + Perplexity의 검증
- **통합 시스템 설계**: 멀티 LLM 협업 프레임워크
  - **데이터 공유**: 공통 벡터 데이터베이스 활용
  - **프롬프트 최적화**: 각 LLM 특성에 맞는 입력 설계

멀티 LLM 기반 복합 추론 알고리즘 설계

- **멀티 LLM 협업 추론**: OpenAI, Claude, Gemini, Perplexity 각 모델의 강점을 결합하여 시너지 효과를 극대화하는 복합 추론 시스템 구현

OpenAI (GPT-4)	Claude	Gemini	Perplexity
핵심 추론 및 분석력 담당, 재무제표 해석 및 투자 전략 개발	장문 컨텍스트 처리, 복잡한 재무보고서 및 연간 공시자료 종합분석	멀티모달 데이터 처리, 차트/그래프 분석 및 시각자료 기반 예측	실시간 정보 검색, 최신 시장 데이터 및 뉴스 반영 분석

- **다단계 크로스체킹 프로세스**: 각 LLM의 분석 결과를 교차검증하여 정확도와 신뢰도를 향상시키는 자기검증 알고리즘 적용
- **시나리오 기반 복합 추론**: 다양한 경제 시나리오를 생성하고 각 LLM이 시나리오별 영향 분석 후 종합적 투자 전략 도출
- **확률/리스크 평가 시스템**: 각 투자전략에 대한 성공확률, 위험도, 기대수익률을 AI 모델 앙상블 방식으로 계산하여 객관적 평가

멀티 LLM 기반 AI 애널리스트 평가 시스템 UI/UX 계획

- **멀티 LLM 통합 대시보드**: OpenAI, Claude, Gemini, Perplexity 각 모델의 분석 결과를 한눈에 비교할 수 있는 통합 인터페이스, 모델별 장단점 비교 시각화
- **사용자 역할별 맞춤형 인터페이스**: 펀드매니저, 기관투자자, 증권사, 일반 투자자 등 사용자별 최적화된 화면 구성과 AI 분석 결과 제공 방식
- **AI 평가 결과 시각화**: 복잡한 투자 데이터와 평가 지표를 각 LLM의 특성을 살려 시각화하여 제공 (GPT-4의 추론 과정, Claude의 장문 분석, Gemini의 차트 해석, Perplexity의 검색 결과)
- **인터랙티브 멀티모달 리포트**: 다양한 LLM의 분석 결과를 통합하여 심층적으로 살펴볼 수 있는 대화형 리포트 시스템, 모델 간 교차 검증 기능 제공
- **통합 사용자 경험**: PC, 태블릿, 모바일 등 다양한 디바이스에서 멀티 LLM 기반 분석 결과에 접근할 수 있는 반응형 디자인 및 일관된 사용자 경험 제공
- **모델 선택 및 가중치 조정 기능**: 사용자가 각 LLM 모델(OpenAI, Claude, Gemini, Perplexity)의 분석 결과 반영 비율을 조정할 수 있는 직관적인 인터페이스 제공

멀티 LLM 개발 일정표

- **기획 및 착수단계(2025년 하반기)**: 9월 멀티 LLM 프로젝트 킥오프, 10월 OpenAI, Claude, Gemini, Perplexity API 연동 설계, 12월 통합 프로토타입 개발 완료
- **개발 및 검증단계(2026년 상반기)**: 1-2월 멀티 LLM 조정 시스템 개발 및 내부 테스트, 3월 API 성능 및 비용 최적화, 5-6월 멀티 LLM 통합 추론 엔진 고도화, 7월 대규모 벤치마크 테스트
- **서비스 론칭(2026년 하반기)**: 9월 멀티 LLM 기반 애널리스트 평가 시스템 베타 오픈, 10-11월 LLM별 특화기능 및 성능 최적화, 12월 정식 서비스 론칭 및 첫 어워드 시상
- **책임 조직 구성**: 솔트룩스(멀티 LLM 통합 시스템 개발), 이코노미스트(어워드 기획 및 리서치 체계 수립), KG제로인(데이터 제공 및 평가지표 구체화)
- **주요 마일스톤**: 2025년 12월 4개 LLM 통합 프로토타입 완성, 2026년 3월 LLM 간 추론 조율 시스템 구축, 2026년 7월 LLM 성능 비교 분석 완료, 2026년 12월 첫 멀티 LLM 기반 어워드 시상

투자 비용 계획

- **개발 인력 및 인프라 비용**: 총 **4억 원** (AI 엔지니어 4명, 데이터 사이언티스트 2명, 최소 인프라 구성)
- **데이터 구매 및 외부 비용**: 총 **2억 원** (금융 데이터 API 및 KG제로인 데이터 선별적 구매)
- **멀티 LLM API 사용료**: 총 **1.8억 원** (OpenAI 6천만원, Claude 5천만원, Gemini 4천만원, Perplexity 3천만원)
- **멀티 LLM 통합 개발 비용**: 총 **1.2억 원** (API 연동, 통합 인터페이스, 프롬프트 엔지니어링, 데이터 처리 파이프라인)
- **연간 운영 및 고도화 비용**: 총 **2.5억 원/년** (필수 업데이트, 모델 파인튜닝, API 버전 호환성 유지)
- **예상 ROI 및 회수 계획**: 1.5년 내 투자금 회수 목표, 어워드 브랜드 가치 및 기술 라이센싱 통한 추가 수익 창출

총 초기 투자 비용: 9.5억 원 (10억 원 이내 예산 준수)

기대효과 및 결론

- **글로벌 경쟁력 확보**: 멀티 LLM 기반 AI 애널리스트 어워드는 기술 혁신을 통한 국내 금융 리서치 시장의 글로벌 경쟁력 강화에 기여
- **투자자 의사결정 혁신**: 정확한 데이터 분석과 복합추론 능력을 통해 투자자들에게 최적화된 포트폴리오 구성 및 투자 전략 제시
- **금융 리서치 패러다임 전환**: 인간 애널리스트와 AI의 협업 모델을 구축하여 리서치의 품질, 속도, 정확성 모두 향상
- **시장 신뢰도 향상**: 객관적인 평가 체계와 투명한 데이터 기반 분석으로 애널리스트 평가의 공정성과 신뢰성 제고
- **미래 로드맵**: 2026년 파일럿 운영을 시작으로 지속적인 기능 고도화 및 글로벌 스탠더드 AI Investment Award로 발전 추진

Cursor AI 개발 프롬프트: 시스템 개요·아키텍처·원칙

시스템 개요 및 모노레포 구조

아래 요구사항을 기반으로 멀티 LLM(OpenAI, Claude, Gemini, Perplexity) 기반 AI 애널리스트 평가 플랫폼의 모노레포를 생성하고 초기 셋업을 진행하라.

- **리포지토리 구조:** 모노레포 (Turborepo 또는 pnpm workspace)
- **패키지 구성:**
  - `apps/web` - Next.js + TypeScript + Tailwind
  - `apps/api` - FastAPI + uvicorn
  - `packages/shared` - TS/Python 공용 스키마
  - `apps/worker` - Celery 백그라운드 작업
  - `infra` - Docker Compose, IaC 템플릿
- **핵심 모듈:** 포털(리서치/기업/어워드/평가), 에이전트 콘솔, 데이터 파이프라인, 멀티 LLM 오케스트레이션

기술 스택 구성

- **Frontend:** Next.js 14, TypeScript, TailwindCSS, React Query, Zod
- **Backend:** FastAPI, SQLModel/SQLAlchemy, PostgreSQL, Redis, Celery
- **AI/ML:** OpenAI API, Anthropic API, Google Gemini API, Perplexity API, FAISS/pgvector
- **DevOps:** Docker, Kubernetes, OpenTelemetry, Prometheus, Loki

디자인 원칙 및 코딩 규칙

디자인 원칙:

- **전면 흑백:** 컬러 사용 금지, 모든 UI 요소는 흑백으로만 구성
- **직선 구조:** 모든 도형과 다이어그램은 사각형과 직선만 사용 (곡선 금지)
- **반응형:** 모바일부터 데스크톱까지 모든 화면 크기 지원
- **접근성:** ARIA 속성 적용, 키보드 내비게이션 지원
- **다국어:** 한글 UI 기본, 영문 폴백 지원

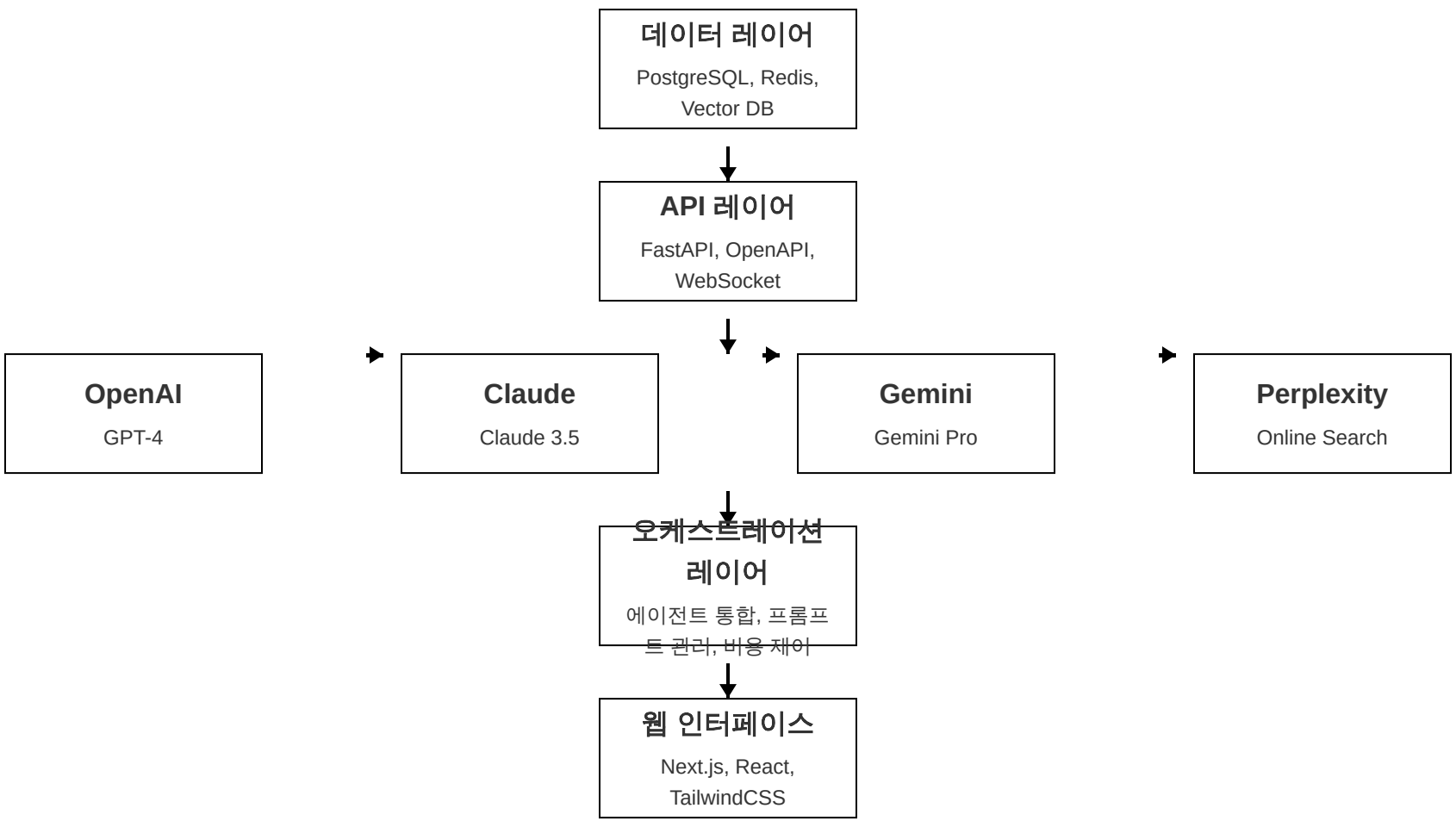
코딩 규칙:

- **린트/포맷:** ESLint, Prettier, ruff, black 적용
- **테스트:** pytest, vitest로 단위/통합 테스트 구현
- **컨벤션:** conventional commits, GitHub Flow 전략

보안 및 비용 관리

- **환경 분리:** .env 파일로 개발/스테이징/프로덕션 환경 분리
- **비용 상한:** LLM API 사용량 쿼터 및 예산 상한 설정
- **민감정보:** 기업 재무, 애널리스트 개인정보 마스킹 처리
- **인증/권한:** JWT + RBAC 기반 세분화된 접근 제어
- **모델 요청:** 프롬프트 인젝션 방지, 입력 샌드박싱
- **컨텐츠:** 출력 필터링, 저작권 검증, 인용 확인

멀티 LLM 기반 아키텍처 구성도



프론트엔드·포털(예: FnGuide 유사) 화면 설계 프롬프트

> Next.js(앱 라우터)로 흑백 UI 포털을 구현하라. 곡선 금지, 사각형 컴포넌트와 직선 구분선만 사용.

주요 화면

1) 대시보드

지표 대시보드, 알림 센터, 작업큐 상태, 최근 활동 로그

`/dashboard`

2) 애널리스트 목록/프로필

성과 지표, 리포트 아카이브, 추천 히스토리, 정확도 시각화

`/analysts` `/analysts/[id]`

3) 리포트

목록, 상세 보기, 미리보기, PDF 다운로드, 섹션별 분석

`/reports` `/reports/[id]`

4) 기업 DB

기업개요, 재무정보, 이벤트 타임라인, 애널리스트 커버리지

`/companies` `/companies/[ticker]`

5) 평가/어워드 현황

스코어보드, 평가기준 설정, 수상 히스토리, 카테고리별 순위

`/awards` `/scores`

6) 에이전트 콘솔

에이전트 상태 모니터링, 작업 실행/로그, 설정 관리

`/agents` `/agents/[id]/logs`

공통 컴포넌트

- **테이블**: 가상 스크롤, 정렬/필터/컬럼 토글 기능, CSV/XLSX 내보내기
- **검색바**: 자동완성, 필터 드롭다운, 최근 검색어 저장
- **페이징**: 번호 기반, 무한 스크롤 옵션, 페이지당 항목 수 설정
- **모달**: 직사각형 형태, 키보드 접근성, 뒤로가기 연동
- **토스트**: 상단 또는 우측 하단 알림, 자동 소멸, 상태별 아이콘
- **Stepper**: 직선형 단계 표시, 현재/완료 상태 구분
- **아이콘**: 단색, 직선 스타일, SVG 기반

상태관리 & 폼

React Query: 서버 상태 관리, 캐시 전략, staleTime 설정
폼 처리: Zod + react-hook-form, 유효성 검증, 에러 메시지
상태 패턴: 로딩/에러/성공/빈데이터 처리 일관성

테이블 스타일

- 흑백 컬러 스키마: 컬러 사용 금지
- 셀 테두리: 1px 실선
- 행간(compact): 셀 패딩 최소화
- 헤더 구분: 굵은 테두리로 구분
- 정렬: 숫자 우측, 텍스트 좌측 정렬
- 행 호버: 배경색 약간 어둡게 변경
- 선택된 행: 테두리 강조

접근성 요구사항

- 키보드 내비게이션: 모든 기능 Tab 접근 가능
- 명도 대비(흑/백): 최소 4.5:1 비율 준수
- 포커스 스타일: 시각적 구분 명확화
- ARIA 레이블: 모든 상호작용 요소에 적용
- 한글 UI 기본, 영문 폴백 지원

UI/UX 디자인 규칙

- 직사각형 레이아웃: 모든 모서리는 90도 각도
- 일관된 여백: 8px 증분 사용
- 모노크롬: 흑백 + 회색 음영만 사용
- 타이포그래피: 제목 24/20/18px, 본문 16/14px
- 반응형 설계: 모바일(320px) - 데스크탑(1920px)
- 대시보드: 모듈식 카드 그리드
- 스켈레톤 UI: 로딩 상태 명확히 표시

백엔드 API·데이터 스키마 프롬프트

FastAPI로 REST+WebSocket API를 설계·구현하라. OpenAPI 스펙 자동화 및 RBAC 적용.

엔터티(초안)

데이터베이스 주요 엔터티:

- users
- analysts
- report_sections
- filings
- recommendations
- criteria
- agents
- job_logs
- vectors
- roles
- reports
- companies
- forecasts
- awards
- scores
- jobs
- datasets

핵심 스키마 예

analysts

```
id, name, firm, sector, experience_years, profile_url
```

reports

```
id, analyst_id, company_id, date, title, source_url,
parsed_json(JSONB), embedding_vector
```

companies

```
id, ticker, name_kr, name_en, sector, market_cap,
fundamentals(JSONB)
```

API 예

GET `/analysts` - 전체 애널리스트 목록 조회

GET `/analysts/{id}` - 특정 애널리스트 상세정보

GET `/reports?analyst=...` - 특정 조건 리포트 검색

POST `/reports/parse` - 리포트 파싱 작업 요청

POST `/awards/run` - 어워드 선정 평가 실행

POST `/scores/recompute` - 스코어 재계산

POST `/agents/run` - AI 에이전트 작업 실행

WS `/jobs/{job_id}/stream` - 작업 실시간 상태 스트림

인증 및 배치 작업

인증: JWT(Access/Refresh), 역할: admin, editor, viewer

배치/큐: Celery 기반 작업 처리

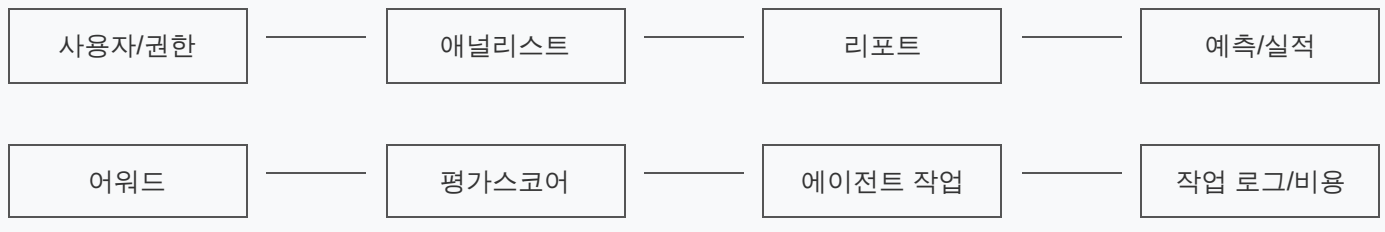
작업 종류:

- parse_report - 리포트 파싱 및 구조화
- fetch_company - 기업 데이터 수집
- compute_score - 애널리스트 성과 점수 계산
- run_award - 어워드 선정 프로세스 실행

검색: pg_trgm/tsvector, 벡터검색(pgvector)로 유사 문서 조회

마이그레이션: Alembic 스크립트 생성

데이터 관계 다이어그램 (ERD)



산출물: ERD(사각형+직선 연결), OpenAPI 문서, 샘플 시드 스크립트

멀티 LLM 연동·오케스트레이션 프롬프트

OpenAI, Anthropic, Google, Perplexity를 어댑터 패턴으로 추상화하고 작업별 라우팅/앙상블을 구현하라.

환경 설정 및 어댑터 패턴

환경변수:

```
OPENAI_API_KEY
ANTHROPIC_API_KEY
GOOGLE_API_KEY
PERPLEXITY_API_KEY
LLM_BUDGET_DAILY
```

어댑터 인터페이스:

- openai_adapter, anthropic_adapter, gemini_adapter, perplexity_adapter
- 통합 인터페이스: generate(), embed(), tools()

라우팅 전략

- **생성/요약:** OpenAI→Claude 폴백
- **장문 컨텍스트:** Claude 우선
- **멀티모달:** Gemini 우선
- **실시간 검색:** Perplexity→OpenAI 리라이트

출력 포맷 및 품질 관리

출력 포맷:

- JSON Schema 기반의 strict 모드(함수호출/툴콜 선호)
- 체인오브소트 노출 금지·간단 근거 요약만

비용/안정성:

- 레이트리밋, 지수백오프, 재시도 메커니즘
- 캐시(Redis), 결과 해시 검사
- 샌드박스 테스트

관측:

- 요청/응답/비용 로깅
- 샘플링 기반 품질평가(BLEU/ROUGE/정확도)
- E2E 회귀 테스트

프롬프트 템플릿 레이어

System 프롬프트

역할 정의, 규칙, 톤 설정

↓

Task 프롬프트

목표, 입력 스키마, 작업 정의

↓

Guard 프롬프트

법률/윤리적 가이드, 금칙어

산출물

어댑터 코드:

- 4개 어댑터 모듈
- 공통 인터페이스 구현
- 인증 및 에러 핸들링

통합 Router:

- 작업 유형별 최적 LLM 선택
- 병렬/직렬 실행 로직
- 결과 검증 및 폴백

샘플 워크플로우:

- YAML 기반 정의
- 재사용 가능한 파이프라인
- 주요 작업별 예시 구현

AI 에이전트 정의·워크플로우 프롬프트

아래 에이전트들을 정의·구현하라. 각 에이전트에 대해 입력/출력 스키마(JSON), 사용 툴, 호출 모델, 품질지표, 실패시 폴백, 엔드포인트/잡 스케줄을 명시.

1. 평가 에이전트 (Evaluation)

목적: 애널리스트별 정확도/시의성/커버리지 산출 → scores 테이블 업데이트

입력: analyst_id, period, metrics[]

출력: {accuracy: float, timeliness: float, coverage: float, composite: float}

툴: DB 쿼리, 통계 라이브러리, 성과 메트릭 계산기

호출 모델: OpenAI + Claude 크로스체크(결과 검증)

엔드포인트: POST /agents/evaluation/run, 매일 23:00 배치 작업

2. Award 에이전트 (Award)

목적: 연간/분기별 카테고리별 후보 선별→근거 생성→우승자 산출

입력: {year: int, quarter?: int, categories: string[]}

출력: {winners: [{category, analyst_id, evidence}], runners_up: [...]}

툴: 점수 집계기, 순위 알고리즘, 텍스트 근거 생성기

호출 모델: Claude(장문 컨텍스트) → OpenAI(검증)

엔드포인트: POST /agents/award/run, 분기별/연간 스케줄

3. 리포트 생성 에이전트 (Report Generation)

목적: 기업/섹터 분석 리포트 자동 생성(요약/차트 지정), JSON 구조 출력

입력: {company_id/sector_id, report_type, depth_level, data_sources: []}

출력: {sections: [], summary: "", charts: [], recommendations: []}

툴: 차트 생성기, 표 생성기, 코퍼스 검색, 문서 템플릿

호출 모델: OpenAI GPT-4 (추론) + Claude (검증)

엔드포인트: POST /agents/report/generate, 온디맨드

4. 리포트 파싱 에이전트 (Report Parsing)

목적: PDF/HTML 파싱→섹션/표 추출→정규화→임베딩 적재

입력: {report_url, report_type, source_format, analyst_id}

출력: {sections: [], tables: [], entities: [], embedding_vectors: []}

툴: PDF 파서, HTML 추출기, 표 구조화, 벡터 임베딩

호출 모델: Gemini(멀티모달) + OpenAI 임베딩

엔드포인트: POST /agents/report/parse, 업로드시 트리거

5. 기업정보 검증 에이전트 (Company Verification)

목적: 공시/포털 대조로 데이터 정합성 검증

입력: {company_id, verification_fields: [], sources: []}

출력: {verified: bool, discrepancies: [], corrected_data: {}, confidence: float}

툴: 공시 크롤러, 웹 스크래퍼, 데이터 비교기, 규칙 엔진

호출 모델: Perplexity(검색) + Claude(판단)

엔드포인트: POST /agents/company/verify, 매일 08:00 배치

6. 실적 검증 에이전트 (Performance Verification)

목적: 예측 vs 실제, 지표별 MAPE/Bias 산출

입력: {analyst_id, company_id, period, metrics: []}

출력: {mape: float, bias: float, hit_rate: float, metrics_detail: []}

툴: 통계 라이브러리, 예측 평가기, 시계열 비교기

호출 모델: OpenAI + Gemini(차트 검증)

엔드포인트: POST /agents/performance/verify, 분기 실적 발표 후

7. 추천종목 추적 에이전트 (Stock Tracking)

목적: 추천 이후 수익률/최대낙폭/샤프지수 추적

입력: {recommendation_id, tracking_period, benchmark_id}

출력: {returns: float, max_drawdown: float, sharpe: float, benchmark_diff: float}

툴: 시계열 분석기, 성과 계산기, 위험 측정기

호출 모델: OpenAI + Perplexity(시장 변동 이슈 검색)

엔드포인트: POST /agents/tracking/calculate, 매일 시장 종료 후

8. 데이터 수집 에이전트 (Data Collection)

목적: 크롤링/스크래핑(robots 준수, 재시도/백오프)

입력: {sources: [], data_types: [], date_range: {}, filters: {}}

출력: {collected_data: [], stats: {}, errors: [], next_cursor: string}

툴: Playwright, Selenium, RSS 파서, API 클라이언트

호출 모델: Gemini(구조 인식) + Perplexity(검증)

엔드포인트: POST /agents/data/collect, 스케줄링(Airflow)

9. 멀티 LLM 오케스트레이터 (Orchestrator)

목적: 작업 분해/라우팅/앙상블/크로스체크

입력: {task: string, context: {}, params: {}, quality_threshold: float}

출력: {result: {}, confidence: float, models_used: [], reasoning: []}

툴: 작업 분해기, 모델 라우터, 결과 앙상블러, 일관성 검사기

호출 모델: 모든 LLM + 자체 라우팅 로직

엔드포인트: POST /agents/orchestrator/run, 시스템 내부 호출

10. 포트폴리오 분석 에이전트 (Portfolio Analysis)

목적: 리밸런싱 제안, 리스크 한도 체크

입력: {portfolio_id, holdings: [], constraints: {}, target_return: float}

출력: {rebalance_actions: [], risk_metrics: {}, expected_return: float}

툴: 포트폴리오 최적화기, 리스크 분석기, 상관관계 행렬

호출 모델: OpenAI(추론) + Claude(검증) + Perplexity(시장 조건)

엔드포인트: POST /agents/portfolio/analyze, 주간/온디맨드

11. 애널리스트 리포트 AI 에이전트 (Analyst Report Agent)

목적: 고성능 PDF 파서로 애널리스트 리포트 구조화 → 표/차트/이미지/텍스트 추출

입력: {report_file: binary, report_type: string, extraction_targets: ["tables", "charts", "text", "formulas"]}

출력: {structured_content: {}, tables: [], charts: [], images: [], formulas: [], text_blocks: [], metadata: {}}

특수 기능: VLM 차트인식, LLM 보정, 한/영 인코딩 처리, 수식 OCR, 표 구조화, 이미지 텍스트 추출

툴: PyMuPDF/PDFPlumber, OCR(Tesseract+EasyOCR), 차트 인식기(Gemini Vision), 표 추출기(Tabula), 수식 파서(LaTeX), 한글 인코딩 컨버터

호출 모델: Gemini(멀티모달) + Claude(텍스트 컨텍스트) + OpenAI(구조화) + OCR 엔진

엔드포인트: POST /agents/report/analyze, 리포트 업로드시 자동 트리거

파싱 파이프: 1) 메타데이터 추출 → 2) VLM 레이아웃 분석 → 3) 표/이미지/차트 영역 식별 → 4) OCR 텍스트 추출 → 5) LLM 보정(오류수정) → 6) 구조화

공통 가드레일 및 표준

사내 정책/저작권/개인정보 준수, 출처/근거 필드 포함, 한국어 기본 출력

입출력 스키마는 JSON Schema 기반으로 엄격하게 유효성 검사

모든 에이전트는 실패 처리 로직과 로깅, 비용 추적, 재시도 전략 포함

레이트리밋 준수, 결과 캐싱, 데이터 마스킹 표준 적용

품질 메트릭 측정: 정확도, 정밀도, 재현율, F1, 응답시간 등