25.06.21 파일 정리

프로젝트 개요

- 프로젝트명: 리뷰 자동화 SaaS 서비스
- 주요 기능: 배민/요기요/쿠팡이츠 리뷰 크롤링 및 AI 답글 자동화
- URL: http://localhost:8000/
- 기술 스택: Python, FastAPI, Playwright, Supabase, OpenAl GPT-4

📂 전체 폴더 구조

Oth Deview releasure in the transfer in the tr
C:\Review_playwright/
├── 📄 .env # 환경변수
requirements.txt
├── 📄 run_server.bat
├── 📄 CLAUDE.md # 🔤 Claude 문서
├── 📄 SQL_playwright.txt # 🔤 DB 스키마 (루트로 이동)
├── 📄 test_date_parsing.py # 🔤
├── 📄 .gitignore
— 📄 .htaccess
README.md
— 📄 plan.md
— 📄 create_file.bat
├— <mark></mark> api/
│ ├── 📄 main.py # 🔤 FastAPI 메인 앱

```
– proper 
                        - 📁 services/
                       review_collector_service_sync.py # NEW
                        - ≝ routes/ # API 엔드포인트
                                                                                      # 🔤 (중복? 확인 필요)
                        - 📁 routers/
                       – 📁 schemas/
                       – 📁 auth/
               — 📁 utils/
           dependencies.py
                                                                                      # 구조 변경됨
            - 📁 web/
                    — 📁 static/
                                                                                 # 🔤 정적 파일 디렉토리
                        ---- css/
                       └── js/
                      − 📁 templates/ # 🔤 템플릿 디렉토리
                     ---- == auth/
                        — 📁 dashboard/
                         — 📁 reviews/
                                - 📁 stores/
                    base.html
                    index.html
                    store_register.html
         – 📁 config/
         – 📁 logs/
     --- 📁 scripts/
      — 📁 docs/
          – 📁 tests/
          – 📁 backups/
         – 📁 browser_data/
     — 📁 data/
         — i downloads/
          – 📁 temp/
           - 🣁 .vscode/
├── 📁 .git/
├── 📁 .claude/ # 🔤 Claude 관련 디렉토리
      — 📁 SHIRMP/
```

핵심 모듈 상세 구조

🧰 /api/crawlers/ - 크롤러 모듈

 /api/crawlers/ ├──
review_crawlers/ # 리뷰 수집 전문 ☆ □ run_sync_crawler.py # 메인 실행 스크립트 (배민) □ run_coupang_async_crawler.py # 쿠팡 리뷰 크롤링 실행 ✓ □ run_yogiyo_async_crawler.py # 요기요 리뷰 크롤링 실행 ✓ □ run_naver_async_crawler.py # 네이버 리뷰 크롤링 실행 ✓ □ baemin_sync_crawler.py # 배민 기본 크롤러 □ baemin_sync_review_crawler.py # 배민 리뷰 크롤러 □ baemin_review_crawler.py # 배민 비동기 버전 □ coupang_async_review_crawler.py # 쿠팡 동기 크롤러 ✓ □ coupang_sync_review_crawler.py # 코팡 등기 크롤러 ✓ □ yogiyo_async_review_crawler.py # 요기요 비동기 크롤러 ✓ □ yogiyo_sync_review_crawler.py # 요기요 동기 크롤러 ✓ □ windows_async_crawler.py # 요기요 동기 크롤러 ✓ □ review_crawler.py # 요기요 동기 크롤러 ✓ □ initpy # Windows 최적화 □ review_crawler_subprocess.py # 서브프로세스 크롤러 □ initpy
│ ├── ► store_crawlers/ # 매장 정보 수집 │ ├── ■ crawler_subprocess.py # 서브프로세스 실행 │ ├── ■ naver_store_crawler.py # 네이버 매장 정보 크롤러 ✔ │ ├── ■initpy

```
– 📁 cookies/
      ___ inaver/
        L— cookies_dd6f81d717.ison # 쿠키 파일
       - 📁 screenshots/
      └── ≝ naver/ # 네이버 스크린샷 (50+ 파일)
  – 📁 reply_managers/ 🧈 # 답글 관리 🌠 (모두 구현됨)
  ├── 📄 reply_manager.py   # 답글 관리 베이스 클래스
  ├── 📄 baemin_reply_manager.py # 배민 답글 관리 🔽
  ├── 📄 coupang_reply_manager.py # 쿠팡 답글 관리 🔽
  ├── 📄 yogiyo_reply_manager.py #요기요 답글 관리 🌠
  ├── 📄 naver_reply_manager.py # 네이버 답글 관리 🔽
  ____ init__.py
 — 📁 review_parsers/ # 리뷰 파싱 🔽
  ├── 📄 baemin_review_parser.py # 배민 리뷰 파서 🔽
  ├── 📄 coupang_review_parser.py # 쿠팡 리뷰 파서 🔽
  ├── 📄 naver_review_parser.py # 네이버 리뷰 파서 🔽
  ____init__.py
 — 📁 deprecated/ # 사용 중단
  baemin_windows_crawler.py
└── 📁 __pycache__/ # 컴파일된 파일들
```

🔍 주요 발견사항

1. 동기/비동기 크롤러 모두 구현

- 각 플랫폼별로 sync/async 버전이 모두 존재
- 실행 스크립트는 주로 async 버전 사용

2. 네이버 크롤링 로그 구조

- store_crawlers/logs/ 하위에 네이버 전용 로그 저장
- 브라우저 프로필, 쿠키, 스크린샷 체계적 관리
- 프로필 ID로 세션 관리 (profile_dd6f81d717)

3. 모든 플랫폼 지원 완료

- 배민, 쿠팡, 요기요, 네이버 모두 구현됨
- 각 플랫폼별 parser, reply_manager 구현 완료

🔧 개선 권장사항

1. pycache 정리

- .gitignore에 추가하여 버전 관리에서 제외
- 주기적으로 삭제 필요

2. 로그 파일 관리

- 네이버 스크린샷이 50개 이상 누적됨
- 오래된 로그 파일 자동 정리 로직 필요

3. 파일 명명 일관성

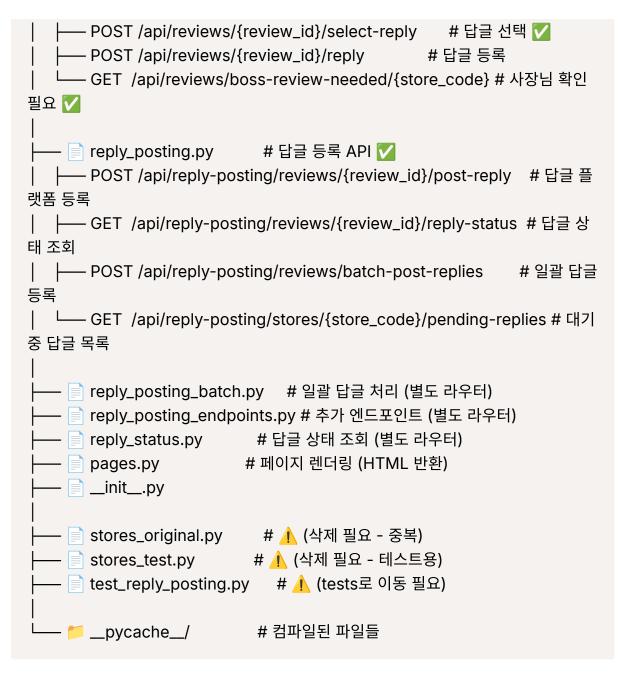
- sync/async 구분을 파일명에 명확히 표시
- 예: baemin_async_review_crawler.py (현재는 혼재)

🧊 /api/services/ - 비즈니스 로직

<pre>/api/services/</pre>
├── 📄 ai_service.py # GPT-4 답글 생성 🜟
├── 📄 ai_reply_service.py # Al 답글 서비스 (중복) <u>↑</u> - 동기용
├── review_collector_service.py # 리뷰 수집 통합 관리 (비동기) ★
├── 📄 review_collector_service_sync.py # 리뷰 수집 (동기) 🔤
├── 📄 reply_posting_service.py # 답글 등록 서비스 🜟
├── reply_service.py # 답글 관리
├── 📄 review_service.py # 리뷰 관리
├── 📄 review_processor.py # 리뷰 처리
├── 📄 supabase_service.py # DB 연결 관리 🜟
├── 📄 supabase_extension.py # DB 확장 기능
├── 📄 supabase_extension_methods.py # DB 확장 메서드
├── 📄 user_service.py # 사용자 관리
├── 📄 database.py # DB 초기화
├──
├── 📄 error_logger.py # 에러 로깅
— initpy
├──

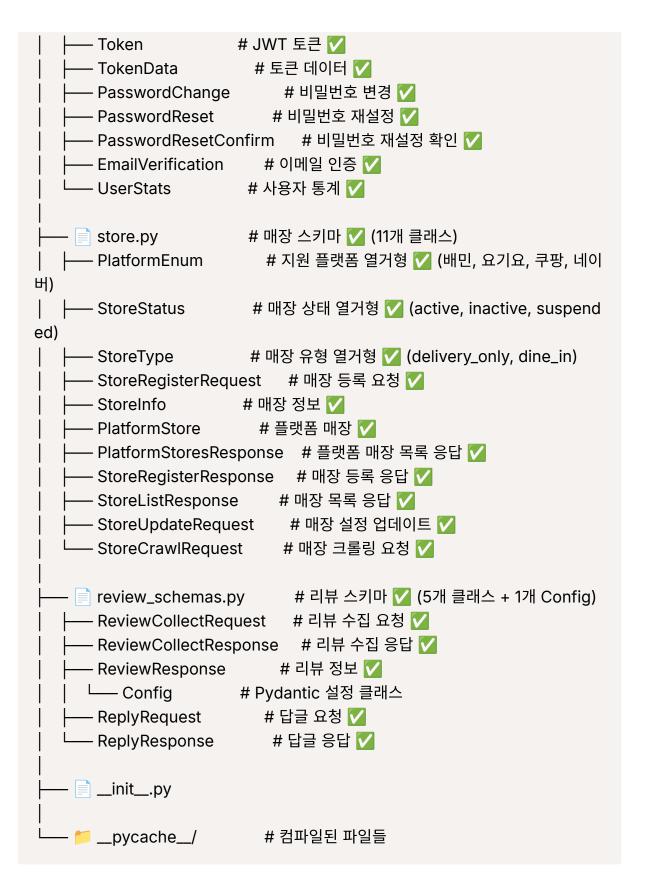
ブ /api/routes/ - API 엔드포인트

```
/api/routes/
               # 인증 API 🔽
 — 📄 auth.py
   ├── POST /api/auth/register # 회원가입
   ├── POST /api/auth/login # 로그인
     — GET /api/auth/me # 내 정보
   └── POST /api/auth/refresh # 토큰 갱신 <a>[☑</a> (구현되어 있음)
  — 📄 stores.py # 매장 관리 API 🔽
   ├── GET /api/stores/platforms # 플랫폼 목록
     — POST /api/stores/crawl # 매장 크롤링
   ├── POST /api/stores/register # 매장 등록
   ---- GET /api/stores # 매장 목록
   ├── GET /api/stores/{store_code} # 특정 매장 조회
   ├─ PUT /api/stores/{store_code} # 매장 수정
   L— DELETE /api/stores/{store_code} # 매장 삭제
   - 📄 reviews.py # 리뷰 관리 API 🌠 (11개 엔드포인트)
   ├── POST /api/reviews/collect # 리뷰 수집
     — POST /api/reviews/collect/all # 전체 리뷰 수집
     — GET /api/reviews/{store_code} # 리뷰 조회
   ├── GET /api/reviews/{store_code}/debug # 디버그 정보 조회 🔽
    --- GET /api/reviews/stats/{store_code} # 리뷰 통계 🔽
     – POST /api/reviews/{review_id}/generate-reply # AI 답글 생성 🔽
     — POST /api/reviews/{review_id}/regenerate-reply # AI 답글 재생성
├── GET /api/reviews/{review_id}/generation-history # 생성 이력 조회
```

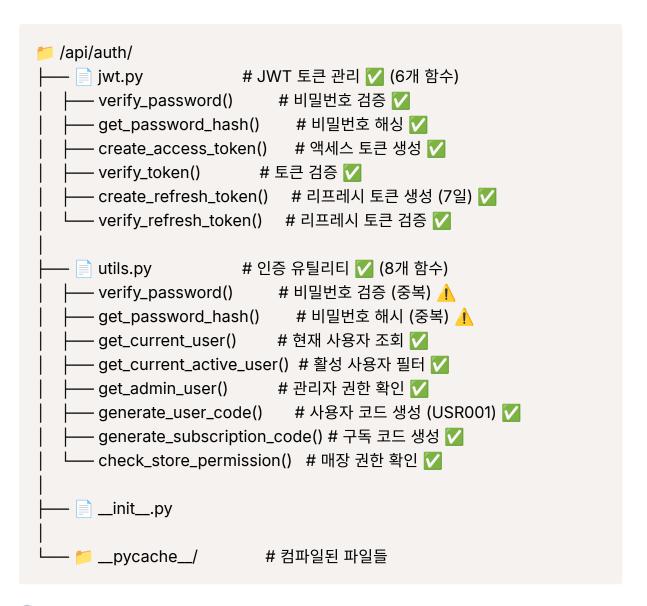


📁 /api/schemas/ - 데이터 스키마

```
    /api/schemas/
    ├── auth.py # 인증 스키마 ✔ (14개 클래스)
    ├── UserBase # 사용자 기본 정보 ✔
    ├── UserCreate # 회원가입 요청 ✔
    ├── UserUpdate # 사용자 정보 수정 ✔
    ├── User # 사용자 응답 모델 ✔
    ├── UserInDB # DB 사용자 정보 (비밀번호 포함) ✔
    ├── LoginRequest # 로그인 요청 ✔
    ├── LoginResponse # 로그인 응답 ✔
```



🧰 /api/auth/ - 인증/권한



🔍 주요 발견사항

1. 함수 중복 문제 🔔

- verify_password() 와 get_password_hash() 가 jwt.py와 utils.py에 중복 존재
- 동일한 기능이므로 하나로 통합 필요

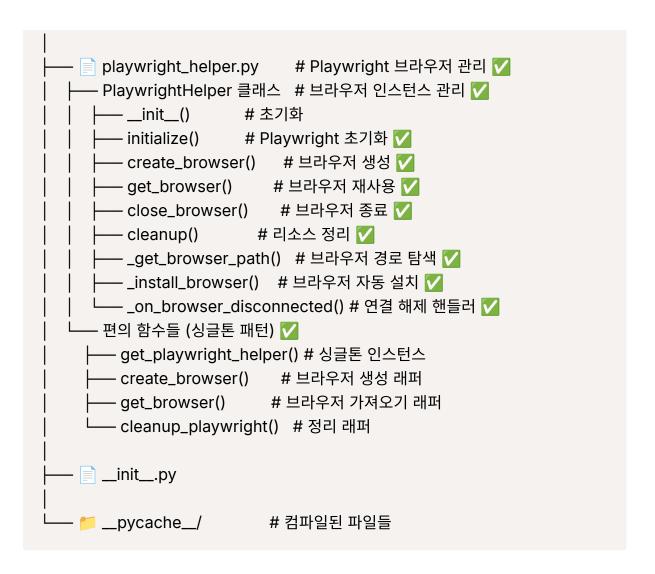
🖿 api/dependencies.py (별도 위치) 🔽

```
python
# 주요 함수들:
├---- get_db()# Supabase 클라이언트 반환
├---- async_wrapper()# 동기→비동기 래퍼
```

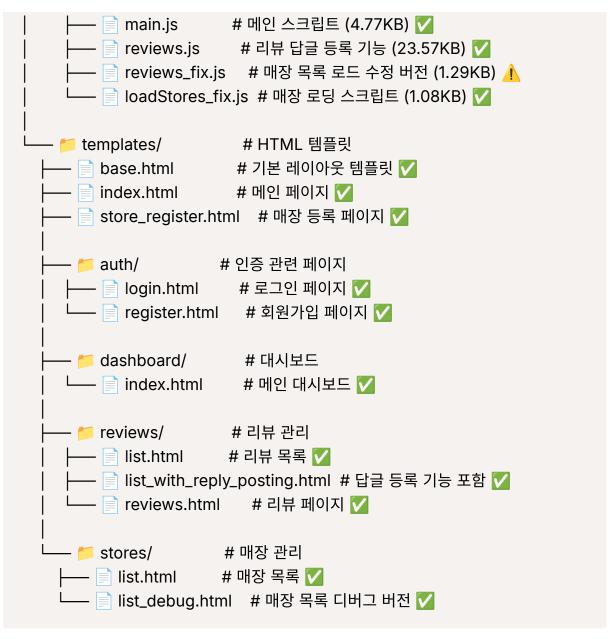
```
── retry_on_connection_error()# 연결 재시도 데코레이터
├── get_current_user()# 현재 사용자 조회 (JWT 검증)
├── get_current_user_optional()# 선택적 사용자 조회
├── get_current_active_user()# 활성 사용자 필터
├── get_admin_user()# 관리자 권한 확인
├── get_supabase_service()# Supabase 서비스
├── get_database_service()# DB 서비스 (별칭)
├── get_reply_posting_service()# 답글 등록 서비스
└── get_openai_client()# OpenAI 클라이언트
```

🧰 /api/utils/ - 유틸리티

```
/api/utils/
 — 📄 date_parser.py # 날짜 파싱 🔽 (2개 함수)
     – parse_relative_date()   # 상대적 날짜 → YYYY-MM-DD
     ├── '오늘', '어제', '그제' 처리 🔽
     ├── 'N일 전', 'N개월 전', 'N주 전' 처리 🔽
     ├── 'YYYY.MM.DD', 'YYYY-MM-DD' 형식 처리 🔽
     └── 'MM월 DD일' 형식 처리 🔽
     — test_date_parser()   # 테스트 함수 ️
  — 📄 error_handler.py 🧈 # 통합 에러 처리 🌠 (3개 클래스 + 1개 메인 클래
스)
     – ErrorCategory 클래스 🖊 에러 카테고리 상수 🔽
     – ErrorSeverity 클래스 🔰 # 심각도 상수 (low, medium, high, critical) 🌠
     – ErrorType 클래스 # 에러 타입 상수 🔽
     – ErrorHandler 클래스 🧼 통합 에러 처리 🔽
      — log_error() # 일반 에러 로깅 🔽
     ├── log_login_error() # 로그인 에러 전용 🔽
       — log_crawling_error() # 크롤링 에러 전용 🔽
     ├── log_reply_error() # 답글 등록 에러 전용 🔽
       — log_api_error()
                        # API 에러 전용 🔽
     └── _save_to_local_file() # 로컬 파일 저장 <
     – 편의 함수들 (싱글톤 패턴) 🔽
     ├── log_login_error() # 싱글톤 래퍼
       - log_crawling_error() # 싱글톤 래퍼
       – log_reply_error() # 싱글톤 래퍼
     └── log_api_error() # 싱글톤 래퍼
```



📁 /web/ - 프론트엔드 UI (업데이트됨)



• JavaScript 파일 중복 문제 🔔

- o auth.js (11.92KB) 6/9 수정된 메인 버전
- auth_new.js (12.69KB) 6/14 생성된 개선 버전
- auth_updated.js (12.47KB) 6/14 생성된 최신 버전
- 3개의 유사한 파일이 공존 → 통합 필요

• reviews 관련 파일

- o reviews.js (23.57KB) 메인 파일 (가장 큼)
- o reviews_fix.js (1.29KB) 수정 버전 (작음)
- 기능 확인 후 통합 필요

주요 변경사항:

1. 폴더 구조 변경

- static/ 폴더에 CSS, JS 등 정적 리소스 분리
- templates/ 폴더에 HTML 템플릿 구성
- 기능별로 하위 폴더 구성 (auth, dashboard, reviews, stores)

2. 파일 중복 및 버전 관리 이슈 🙏

- auth.js, auth_new.js, auth_updated.js 버전별 파일 정리 필요
- reviews.js, reviews_fix.js 수정 버전 통합 필요
- list_debug.html 디버그용 파일 분리

3. 위치 이상 파일 👃

- /static/reviews.html templates/reviews로 이동 필요
- /templates/reviews/loadStores_fix.js static/js로 이동 필요

4. 누락된 리소스

• images/ 폴더가 없음 (로고, 아이콘 등 필요시 추가)

페이지별 기능:

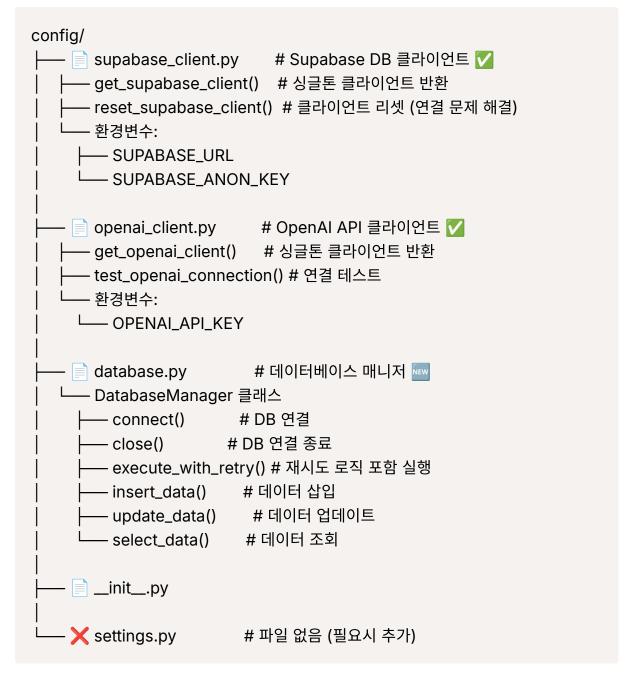
- 인증: login.html, register.html
- 대시보드: dashboard/index.html (메인 통계 및 요약)
- 매장 관리:
 - o store_register.html (신규 등록)
 - stores/list.html (목록 조회)
- 리뷰 관리:
 - o reviews/list.html (기본 목록)
 - o reviews/list_with_reply_posting.html (답글 등록 기능 포함)

개선 필요사항:

- 1. 중복 파일 정리 (auth_.js, reviews.js)
- 2. 파일 위치 정리 (reviews.html, loadStores_fix.js)

- 3. images 폴더 생성 및 리소스 추가
- 4. 일관된 파일 명명 규칙 적용

🧰 /config/ - 설정 파일 (업데이트됨)



주요 특징:

1. supabase_client.py

- 싱글톤 패턴으로 클라이언트 관리
- 환경변수 검증 및 에러 처리

• 연결 문제 시 리셋 기능 제공

2. openai_client.py

- OpenAI 클라이언트 싱글톤 관리
- API 키 유효성 검증
- 연결 테스트 기능 포함

3. database.py

- DatabaseManager 클래스로 DB 작업 추상화
- 자동 재시도 로직 (최대 3회)
- 에러 타입별 처리 및 로깅
- 연결 실패 시 자동 클라이언트 리셋
- 비동기 작업 지원

환경변수 요구사항 (.env):

```
env
# Supabase
SUPABASE_URL=your_supabase_url
SUPABASE_ANON_KEY=your_supabase_anon_key

# OpenAl
OPENAI_API_KEY=your_openai_api_key

# 추가 필요한 설정들 (settings.py에 추가 가능)
JWT_SECRET_KEY=your_jwt_secret_key
ENCRYPTION_KEY=your_encryption_key
ENVIRONMENT=development
```

추가 권장사항:

settings.py 파일 생성 제안:

```
python
# config/settings.py
```

```
import os
from typing import Optional
from pydantic import BaseSettings
class Settings(BaseSettings):
# 앱 기본 설정
  app_name: str = "리뷰 자동화 SaaS"
  app_version: str = "1.0.0"
  environment: str = "development"
# API 설정
  api_prefix: str = "/api"
  api_version: str = "v1"
# 보안 설정
  jwt_secret_key: str
  jwt_algorithm: str = "HS256"
  access_token_expire_minutes: int = 1440# 24시간
# 데이터베이스
  supabase_url: str
  supabase_anon_key: str
# 외부 API
  openai_api_key: str
# 크롤링 설정
  headless_browser: bool = True
  browser_timeout: int = 60000# 60초
# 로깅 설정
  log_level: str = "INFO"
  log_dir: str = "C:/Review_playwright/logs"
  class Config:
    env_file = ".env"
    case sensitive = False
```

```
settings = Settings()
```

❷ 핵심 프로세스 흐름

1. 리뷰 수집 프로세스

```
mermaid graph LR A[웹/\Delta 케줄러] \rightarrow B[review\_collector\_service] \\ B \rightarrow C[BaeminSyncReviewCrawler] \\ C \rightarrow D[네트워크 API 캡처] \\ D \rightarrow E[BaeminReviewParser] \\ E \rightarrow F[Supabase 저장]
```

2. AI 답글 생성 프로세스

```
mermaid graph LR A[ \text{리뷰 } \text{ } \text{d택} ] \rightarrow \text{B[ai\_service]} B \rightarrow \text{C[매장 } \text{정책 } \text{ } \text{Z$i]} C \rightarrow \text{D[GPT-4 } \text{API]} D \rightarrow \text{E[품질 } \text{검증]} E \rightarrow \text{F[DB } \text{저장]}
```

3. 답글 등록 프로세스

```
mermaid
graph LR
```

A[답글 승인] → B[reply_posting_service]

B → C[subprocess 실행]

 $C \rightarrow D[baemin_subprocess.py]$

D → E[Playwright 자동화]

 $E \rightarrow F[상태 업데이트]$

🔡 데이터베이스 구조

주요 테이블

• users: 사용자 정보

• platform_reply_rules: 매장 정보 및 답글 정책

• reviews: 수집된 리뷰

• reply_generation_history: AI 답글 생성 이력

• error_logs: 시스템 에러 로그

• subscriptions: 구독 정보

• usage_tracking: 사용량 추적



1. 개발 환경 설정

bash # 1. 가상환경 생성 python -m venv venv venv\Scripts\activate

2. 패키지 설치 pip install -r requirements.txt

3. 환경변수 설정 (.env) SUPABASE_URL=your_supabase_url SUPABASE_ANON_KEY=your_supabase_key

OPENAI_API_KEY=your_openai_key ENCRYPTION_KEY=your_encryption_key

2. 서버 실행

bash

방법 1: 배치 파일 run_server.bat

방법 2: 직접 실행

uvicorn main:app --reload --port 8000

3. 수동 테스트

bash # 리뷰 수집 테스트 cd api\crawlers\review_crawlers python run_sync_crawler.py

DB 연결 테스트 python scripts\test_supabase.py

🔪 개선 필요사항

1. 코드 정리

- 중복 파일 삭제 (stores_original.py, stores_test.py)
- reply_posting 관련 파일 통합
- 테스트 파일 /tests/ 폴더로 이동

2. 구조 개선

- /api/models/ 폴더 생성 및 SQLAlchemy 모델 정의
- 서비스 레이어 중복 제거 (ai_service vs ai_reply_service)

• 에러 핸들링 통일

3. **기능 추가**

- 요기요, 쿠팡이츠 크롤러 구현
- 실시간 모니터링 대시보드
- 웹훅 알림 시스템

4. 보안 강화

- API Rate Limiting
- 입력 검증 강화
- CORS 설정 최적화