



























25.06.21 파일 정리

C:\Review_playwright 프로젝트 전체 구조 (최종 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, OpenAI GPT-4

전체 폴더 구조

```
C:\Review_playwright/
├──  .env # 환경변수
├──  requirements.txt
├──  run_server.bat
├──  run_review_automation.py
├──  CLAUDE.md #  Claude 문서
├──  SQL_playwright.txt #  DB 스키마 (루트로 이동)
├──  test_date_parsing.py # 
├──  test_review_collection.py # 
├──  test_review_collector.py # 
├──  test_supabase_connection.py # 
├──  test_naver_crawler.py
├──  test_naver_review_crawler.py
├──  .gitignore
├──  .htaccess
├──  README.md
├──  plan.md
├──  create_file.bat
├──
├──  api/
├──   └──  main.py #  FastAPI 메인 앱
```

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

🎯 핵심 모듈 상세 구조

📁 /api/crawlers/ - 크롤러 모듈

```
/api/crawlers/
├── base_crawler.py          # 추상 베이스 크롤러
├── coupang_crawler.py       # 쿠팡이츠 크롤러 ✓
├── yogiyo_crawler.py        # 요기요 크롤러 ✓
├── naver_crawler.py         # 네이버 스마트플레이스 크롤러 ✓
├── __init__.py             # 크롤러 팩토리
├──
├── 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 비동기 크롤러 ✓
│   ├── coupang_sync_review_crawler.py # 쿠팡 동기 크롤러 NEW
│   ├── yogiyo_async_review_crawler.py # 요기요 비동기 크롤러 ✓
│   ├── yogiyo_sync_review_crawler.py # 요기요 동기 크롤러 NEW
│   ├── windows_async_crawler.py # Windows 최적화
│   ├── review_crawler_subprocess.py # 서브프로세스 크롤러
│   └── __init__.py
├──
├── store_crawlers/         # 매장 정보 수집
│   ├── crawler_subprocess.py # 서브프로세스 실행
│   ├── naver_store_crawler.py # 네이버 매장 정보 크롤러 ✓
│   └── __init__.py
├──
├── logs/                  # 크롤링 로그 (네이버 전용) ✓
│   ├── browser_data/
│   │   └── naver/
│   ├── browser_profiles/
│   │   └── naver/
│   │       └── profile_dd6f81d717/ # 프로필 데이터
```

```

├── cookies/
│   └── naver/
│       └── cookies_dd6f81d717.json # 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





```

📁 /api/services/
├── 📄 ai_service.py           # GPT-4 답글 생성 ★
├── 📄 ai_reply_service.py     # AI 답글 서비스 (중복) ⚠️ - 동기용
├── 📄 review_collector_service.py # 리뷰 수집 통합 관리 (비동기) ★
├── 📄 review_collector_service_sync.py # 리뷰 수집 (동기) NEW
├── 📄 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
├── 📄 encryption.py          # 암호화 처리
├── 📄 error_logger.py         # 에러 로깅
├── 📄 __init__.py
└── 📁 platforms/             # 플랫폼별 답글 등록 서비스
  
```

- | | —  baemin_subprocess.py # 배민 답글 등록 실행
- | | —  coupang_subprocess.py # 쿠팡 답글 등록 실행 
- | | —  yogiyo_subprocess.py # 요기요 답글 등록 실행 
- | | —  naver_subprocess.py # 네이버 답글 등록 실행 
- | | —  baemin_reply_manager.py # 배민 답글 매니저
- | | —  __init__.py
- | —  __pycache__/ # 컴파일된 파일들

/api/routes/ - API 엔드포인트

-  /api/routes/
 -  auth.py # 인증 API 
 - POST /api/auth/register # 회원가입
 - POST /api/auth/login # 로그인
 - GET /api/auth/me # 내 정보
 - POST /api/auth/refresh # 토큰 갱신  (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
 - 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
- 

- | | — POST /api/reviews/{review_id}/select-reply # 답글 선택 ✓
- | | — POST /api/reviews/{review_id}/reply # 답글 등록
- | | — GET /api/reviews/boss-review-needed/{store_code} # 사장님 확인 필요 ✓
- | | —  reply_posting.py # 답글 등록 API ✓
- | | — POST /api/reply-posting/reviews/{review_id}/post-reply # 답글 플랫폼 등록
- | | — GET /api/reply-posting/reviews/{review_id}/reply-status # 답글 상태 조회
- | | — POST /api/reply-posting/reviews/batch-post-replies # 일괄 답글 등록
- | | — GET /api/reply-posting/stores/{store_code}/pending-replies # 대기 중 답글 목록
- | | —  reply_posting_batch.py # 일괄 답글 처리 (별도 라우터)
- | | —  reply_posting_endpoints.py # 추가 엔드포인트 (별도 라우터)
- | | —  reply_status.py # 답글 상태 조회 (별도 라우터)
- | | —  pages.py # 페이지 렌더링 (HTML 반환)
- | | —  __init__.py
- | | —  stores_original.py # ⚠ (삭제 필요 - 중복)
- | | —  stores_test.py # ⚠ (삭제 필요 - 테스트용)
- | | —  test_reply_posting.py # ⚠ (tests로 이동 필요)
- | | —  __pycache__/ # 컴파일된 파일들

/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 ✓

Token	# JWT 토큰 ✓
TokenData	# 토큰 데이터 ✓
PasswordChange	# 비밀번호 변경 ✓
PasswordReset	# 비밀번호 재설정 ✓
PasswordResetConfirm	# 비밀번호 재설정 확인 ✓
EmailVerification	# 이메일 인증 ✓
UserStats	# 사용자 통계 ✓
store.py	# 매장 스키마 ✓ (11개 클래스)
PlatformEnum	# 지원 플랫폼 열거형 ✓ (배민, 요기요, 쿠팡, 네이버)
StoreStatus	# 매장 상태 열거형 ✓ (active, inactive, suspended)
StoreType	# 매장 유형 열거형 ✓ (delivery_only, dine_in)
StoreRegisterRequest	# 매장 등록 요청 ✓
StoreInfo	# 매장 정보 ✓
PlatformStore	# 플랫폼 매장 ✓
PlatformStoresResponse	# 플랫폼 매장 목록 응답 ✓
StoreRegisterResponse	# 매장 등록 응답 ✓
StoreListResponse	# 매장 목록 응답 ✓
StoreUpdateRequest	# 매장 설정 업데이트 ✓
StoreCrawlRequest	# 매장 크롤링 요청 ✓
review_schemas.py	# 리뷰 스키마 ✓ (5개 클래스 + 1개 Config)
ReviewCollectRequest	# 리뷰 수집 요청 ✓
ReviewCollectResponse	# 리뷰 수집 응답 ✓
ReviewResponse	# 리뷰 정보 ✓
Config	# Pydantic 설정 클래스
ReplyRequest	# 답글 요청 ✓
ReplyResponse	# 답글 응답 ✓
__init__.py	
__pycache__	# 컴파일된 파일들

/api/auth/ - 인증/권한


```

/api/auth/
├── jwt.py # JWT 토큰 관리 ✅ (6개 함수)
│   ├── verify_password() # 비밀번호 검증 ✅
│   ├── get_password_hash() # 비밀번호 해싱 ✅
│   ├── create_access_token() # 액세스 토큰 생성 ✅
│   ├── verify_token() # 토큰 검증 ✅
│   ├── create_refresh_token() # 리프레시 토큰 생성 (7일) ✅
│   └── verify_refresh_token() # 리프레시 토큰 검증 ✅
├── utils.py # 인증 유틸리티 ✅ (8개 함수)
│   ├── verify_password() # 비밀번호 검증 (중복) ⚠️
│   ├── get_password_hash() # 비밀번호 해싱 (중복) ⚠️
│   ├── get_current_user() # 현재 사용자 조회 ✅
│   ├── get_current_active_user() # 활성 사용자 필터 ✅
│   ├── get_admin_user() # 관리자 권한 확인 ✅
│   ├── generate_user_code() # 사용자 코드 생성 (USR001) ✅
│   ├── generate_subscription_code() # 구독 코드 생성 ✅
│   └── check_store_permission() # 매장 권한 확인 ✅
├── __init__.py
└── __pycache__/ # 컴파일된 파일들

```

🔍 주요 발견사항

1. 함수 중복 문제 ⚠️

- `verify_password()` 와 `get_password_hash()` 가 `jwt.py`와 `utils.py`에 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

```


-  playwright_helper.py # Playwright 브라우저 관리 ✓
- PlaywrightHelper 클래스 # 브라우저 인스턴스 관리 ✓
 - `__init__()` # 초기화
 - `initialize()` # Playwright 초기화 ✓
 - `create_browser()` # 브라우저 생성 ✓
 - `get_browser()` # 브라우저 재사용 ✓
 - `close_browser()` # 브라우저 종료 ✓
 - `cleanup()` # 리소스 정리 ✓
 - `_get_browser_path()` # 브라우저 경로 탐색 ✓
 - `_install_browser()` # 브라우저 자동 설치 ✓
 - `_on_browser_disconnected()` # 연결 해제 핸들러 ✓
- 편의 함수들 (싱글톤 패턴) ✓
 - `get_playwright_helper()` # 싱글톤 인스턴스
 - `create_browser()` # 브라우저 생성 래퍼
 - `get_browser()` # 브라우저 가져오기 래퍼
 - `cleanup_playwright()` # 정리 래퍼
-  `__init__.py`
-  `__pycache__` # 컴파일된 파일들

/web/ - 프론트엔드 UI (업데이트됨)

-  /web/
 -  static/ # 정적 리소스
 -  css/ # 스타일시트
 -  style.css # 메인 스타일시트 ✓
 -  reviews.css # 리뷰 페이지 전용 스타일 ✓
 -  js/ # JavaScript
 -  api-config.js # API 설정 및 엔드포인트 (11.73KB) ✓
 -  auth.js # 인증 관련 기능 (11.92KB)  6/9 수정 ✓
 -  auth_new.js # 인증 기능 개선 버전 (12.69KB)  6/14 생성
 -  auth_updated.js # 인증 기능 최신 버전 (12.47KB)  6/14 생성

main.js	# 메인 스크립트 (4.77KB) ✓
reviews.js	# 리뷰 답글 등록 기능 (23.57KB) ✓
reviews_fix.js	# 매장 목록 로드 수정 버전 (1.29KB) ⚠
loadStores_fix.js	# 매장 로딩 스크립트 (1.08KB) ✓
templates/	# HTML 템플릿
base.html	# 기본 레이아웃 템플릿 ✓
index.html	# 메인 페이지 ✓
store_register.html	# 매장 등록 페이지 ✓
auth/	# 인증 관련 페이지
login.html	# 로그인 페이지 ✓
register.html	# 회원가입 페이지 ✓
dashboard/	# 대시보드
index.html	# 메인 대시보드 ✓
reviews/	# 리뷰 관리
list.html	# 리뷰 목록 ✓
list_with_reply_posting.html	# 답글 등록 기능 포함 ✓
reviews.html	# 리뷰 페이지 ✓
stores/	# 매장 관리
list.html	# 매장 목록 ✓
list_debug.html	# 매장 목록 디버그 버전 ✓

• JavaScript 파일 중복 문제 ⚠

- `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

- `reviews.js` (23.57KB) - 메인 파일 (가장 큼)
- `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/reviews`로 이동 필요
- `/templates/reviews/loadStores_fix.js` - `static/js`로 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:
 - `store_register.html` (신규 등록)
 - `stores/list.html` (목록 조회)
- 리뷰 관리:
 - `reviews/list.html` (기본 목록)
 - `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)

```
config/
├──  supabase_client.py    # Supabase DB 클라이언트 
│   ├── get_supabase_client() # 싱글톤 클라이언트 반환
│   ├── reset_supabase_client() # 클라이언트 리셋 (연결 문제 해결)
│   └── 환경변수:
│       ├── SUPABASE_URL
│       └── SUPABASE_ANON_KEY
├──  openai_client.py      # OpenAI API 클라이언트 
│   ├── get_openai_client()   # 싱글톤 클라이언트 반환
│   ├── test_openai_connection() # 연결 테스트
│   └── 환경변수:
│       └── OPENAI_API_KEY
├──  database.py          # 데이터베이스 매니저 
│   └── DatabaseManager 클래스
│       ├── connect()      # DB 연결
│       ├── close()        # DB 연결 종료
│       ├── execute_with_retry() # 재시도 로직 포함 실행
│       ├── insert_data()   # 데이터 삽입
│       ├── update_data()   # 데이터 업데이트
│       └── select_data()   # 데이터 조회
├──  __init__.py
└──  settings.py          # 파일 없음 (필요시 추가)
```

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

# OpenAI
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()
```

핵심 프로세스 흐름

1. 리뷰 수집 프로세스

```
mermaid
graph LR
  A[웹/스케줄러] → B[review_collector_service]
  B → C[BaeminSyncReviewCrawler]
  C → D[네트워크 API 캡처]
  D → E[BaeminReviewParser]
  E → F[Supabase 저장]
```

2. AI 답글 생성 프로세스

```
mermaid
graph LR
  A[리뷰 선택] → B[ai_service]
  B → C[매장 정책 조회]
  C → D[GPT-4 API]
  D → E[품질 검증]
  E → F[DB 저장]
```

3. 답글 등록 프로세스

```
mermaid
graph LR
```

```
A[답글 승인] → B[reply_posting_service]
B → C[subprocess 실행]
C → D[baemin_subprocess.py]
D → E[Playwright 자동화]
E → 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

실행 방법

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