AI회계 프로젝트 기획서

AI 회계 자동화 시스템 – 통합 기획서 (v0.3 / 2025-04-17)

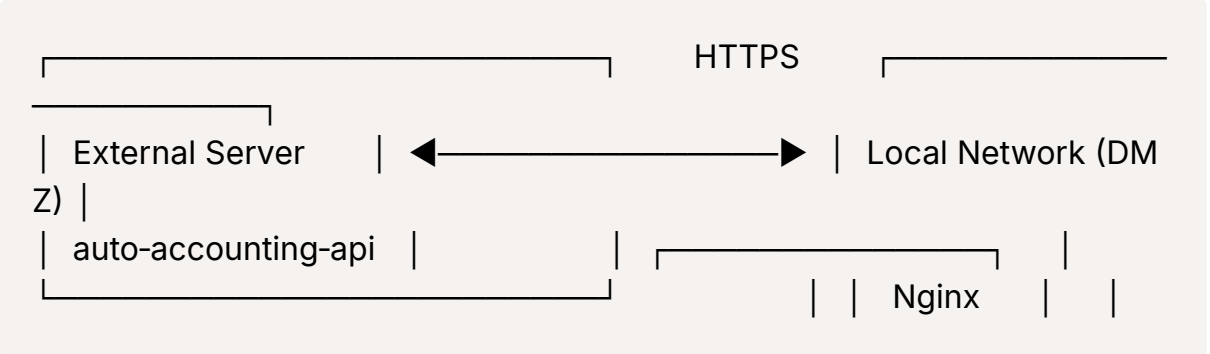
1. 프로젝트 개요

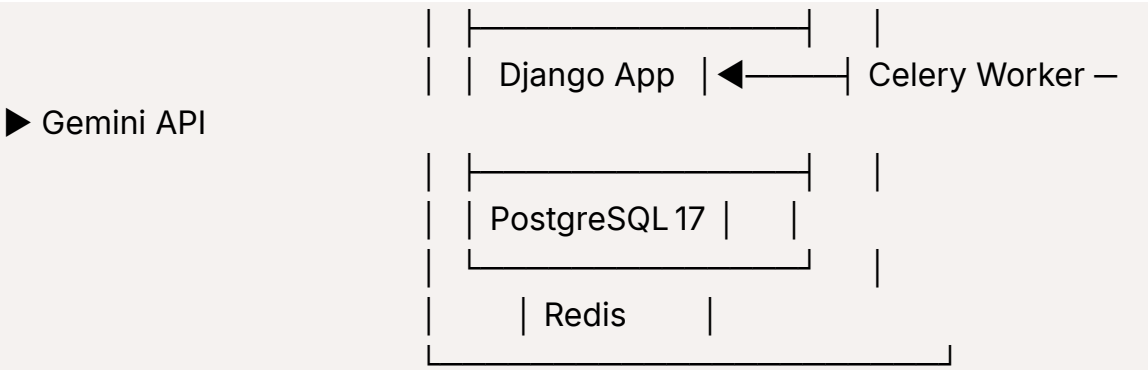
항목	내용
기간	2025-04-17 ~ 2025-04-21 (5-Day MVP)
스택	Python 3.10.6 · Django 4.1.0 · PostgreSQL 17
현황	**Day 1 완료** – Repo 생성, Docker Compose, DB 컨테이너 기동, 기본 앱 (users · accounting · i18n) 마이그레이션 적용
목표	인보이스 → 전표(JE) 자동 전환 → 재무제표 생성 + Bank Recon + 월마감 10-Step 자동화 + 대시보드 시각화

2. 범위 정의

구분	In-Scope	Out-of-Scope
문서	PDF · JPEG · PNG 인보이스, CAMT.053 v2	수기 영수증, HWP
DB	PostgreSQL 17	MySQL · NoSQL
언어	Python 3.10.6 + Django 4.1.0	PHP · Ruby
배포	Docker-Compose + Ubuntu 20.04	Kubernetes

3. 시스템 아키텍처 (On-Prem + External API)





- **배포 형태** : Django App·DB·Redis·Celery는 **기업 내부 로컬 서버**(DMZ VLAN)에서 구동
- **외부 호출** : `auto-accounting-api.company.com`(External Server)이 HTTPS 로 내부 Nginx 프록시를 호출해 회계 자동화 엔드포인트(`/api/v1/upload/` , `/api/v1/closing/run/` 등)를 사용
- **접근 제어** : 외부↔내부 통신은 방화벽 포트 하나만 열고,
  JWT + IP Allowlist + mTLS 적용

---

4. 데이터 모델 (요약 ERD) 데이터 모델 (요약 ERD)

```
# 마스터
Entity(entity_id PK)     Article(article_id PK)     Account(code PK)
  └ name                   └ description              └ name
  └ reg_number               └ unit_price               └ type
  └ country_code             └ vat_rate                 └ is_verified

# 거래 & 전표
Invoice(invoice_id PK) —▶  Entity FK
  └ issuance_date ...
JournalEntryHeader(je_id PK)
  └ doc_no UNIQUE
  └ trx_date
JournalEntryLine(line_id PK) —▶  je_id FK | account FK | article FK

# 다국어(i18n)
TranslationSource(src_id PK) — key · default_text
TranslationEntry(entry_id PK) — src_id FK · language_code · translated_text
Language(code PK) — name
```

```
# 로그 & 기타
UploadLog(file_id PK) · ClosingLog(close_id PK) ...
```

5. 주요 모듈 & 책임

모듈	경로	핵심 책임
users	`users/`	Django 기본 `AbstractUser` 확장 (준비됨)
i18n	`i18n/`	`TranslationSource/Entry` 모델·템플릿 태그 (코드 완료)
accounts	`accounts/`	(추후) 사용자 Auth UI·권한
uploader	`uploader/`	다중 파일 업로드 & `UploadLog` 저장
parser	`parser/`	Gemini Vision OCR · 인보이스 JSON 매핑
masters	`masters/`	Entity · Article auto-sync (`get_or_create`)
je	`je/`	Debit/Credit 생성 · `doc_no` 발행 · COA 체크
bank_recon	`bank_recon/`	CAMT.053 v2 파서 → 거래 매칭 · 조정분 JE 생성
closing	`closing/`	월마감 10-Step 실행 · `ClosingLog` 저장
fs	`fs/`	Trial Balance → PL · BS · CF 집계
dw	`dw/`	JE · FS Fact 적재 (ClickHouse or PG View)
dashboard	`dashboard/`	Streamlit/HTMX 대시보드 · 알림
api	`api/v1/`	Swagger(OpenAPI) REST · JE · FS · Recon · Closing

6. REST API 스케치

Method	Path	설명
POST	`/api/v1/upload/`	인보이스 파일 업로드
GET	`/api/v1/je/{doc_no}/`	전표 단건 조회
GET	`/api/v1/fs/`	재무제표(PL · BS · CF) 조회
POST	`/api/v1/bank/recon/`	CAMT.053 업로드 & 매칭 실행
POST	`/api/v1/closing/run/`	월마감 10-Step 실행

7. 프로세스 파이프라인

#	모듈	핵심 로직	산출물

0	uploader	파일 저장 → `UploadLog(PENDING)`	원본 파일
1	parser	OCR + Schema 매핑	Invoice Dict
2	masters	거래처·품목 `get_or_create`	Entity · Article
3	je	Debit/Credit 계산 + doc_no 발행	JEHeader + JELines
4	gl	GL 잔액 실시간 갱신	Trial Balance
5	fs	TB → 재무제표 집계	PL · BS · CF
6	bank_recon	CAMT 레코드 → JE 매칭	Recon Report
7	closing	10-Step 자동 분개 · 검증	ClosingLog
8	dw	Fact 적재 → ClickHouse	je_fact · fs_fact
9	analytics	KPI 시각화 · 알림	대시보드 UI

> 시그널 : 모델 save post-signal 로 JE→GL, GL→FS 단계 호출

---

8. 핵심 정책 & 규칙

- **COA 시드** : `data/coa_kifrs.yml` 자동 로드, 없는 계정은 `9xxx-TEMP` 로 생성· `is_verified=False`
- **ACID** : JE 생성 ~ GL posting 동일 트랜잭션
- **Bank Recon** : 매칭 실패분 `unmatched=True` → 알림 & 수동 처리
- **월마감** : Celery Beat 스케줄러, 완료 시 Slack 웹훅 통지
- **i18n** : `{{ _('welcome_title') }}` 식 템플릿 태그, 관리자 UI 로 번역 수정

---

9. 일정 (5-Day MVP – 설계·모듈·조립)

Day	상태	주요 목표	산출물
Day 1	⏳	**기획·설계**• 요구사항 상세화·모듈 분할 전략 확정• ERD · COA 시드 검증• `project_spec.md` 작성	확정 기획서, 세부 모듈 목록, ERD 다이어그램
Day 2		**임시 모듈 개발 (파트 1)**• `uploader/` , `parser/` 기능 독립 구현• pytest 통과	`temp_modules/uploader.py` , `parser.py` , 테스트 리포트

Day 3		**임시 모듈 개발 (파트 2)**• `je/builder.py`, `bank_recon/` 기본 구현• pytest 통과, `module_manifest.yml` 완성	`temp_modules/je.py`, `bank_recon.py`, manifest
Day 4		**모듈 조립 & 통합**• temp 모듈을 본 프로젝트에 이식• 업로드→OCR→JE 파이프라인 연결• Bank Recon UI 카드	통합 코드, 매칭율 리포트, `/admin` 업로드 기능
Day 5		**월마감 10-Step & MVP 데모**• closing 스케줄러 실행, 재무제표 생성• Streamlit 대시보드 데모	`closing/log`, Screencast, 데모 영상

10. 보안 & 컴플라이언스

- 업로드 파일 *SSE-S3* 암호화 예정 (MinIO)
- 외부 API↔로컬 Nginx 구간 **mTLS + WAF** 적용
- JWT Oauth2 토큰 + IP 허용목록으로 외부 서버만 호출 허용
- PII 필드 응답 시 마스킹 필터
- 운영 단계 전까지 **VPN IP 화이트리스트** 추가 적용