

PJT명	금융 상품 데이터를 활용한 REST API Server 구축	
단계	07 PJT	
진행일자	2025.11.14	
예상 구현 시간	필수기능	6H
	심화기능	2H

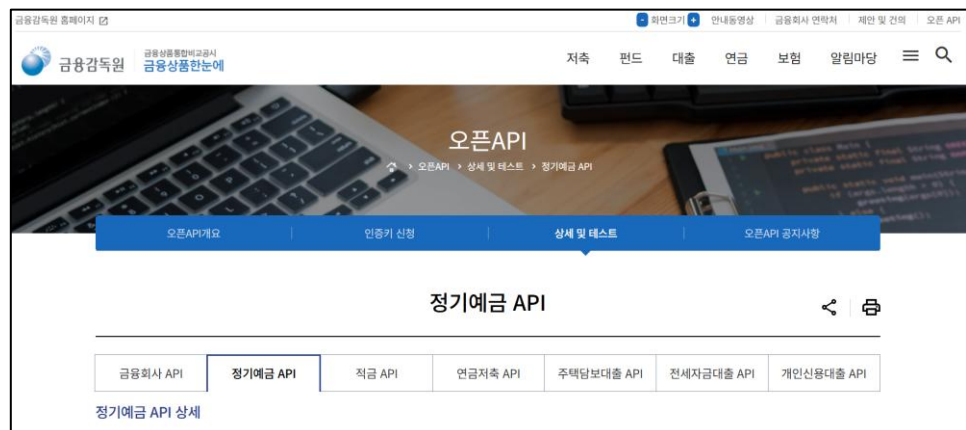
1. 목표

- Django 프레임워크를 활용하여 REST API Server를 구축할 수 있다.
- 외부 금융상품통합비교공시 API(정기예금 API)를 활용하여 데이터를 수집·저장할 수 있다.
- Django REST Framework(Serializer)를 활용해 JSON 형태의 응답을 반환할 수 있다.
- Postman을 활용해 REST API 요청 및 응답을 테스트할 수 있다.

2. 준비사항)

1) 사용 데이터 및 API

- 금융상품통합비교공시 API (정기예금 API)
 - [정기예금 데이터](#)
 - 주요 데이터: 정기예금 상품 정보 및 옵션 정보



2) 금융 감독원 API KEY

- [인증키 신청 링크](#)
 - 발급 절차
 - 회원가입 및 로그인
 - 상단 “인증키 신청” 클릭
 - 약관 동의 후 신상정보 입력
 - 사용 용도: WEB / 사용 URL: http://127.0.0.1:8000/ 입력
 - 휴대폰 및 이메일 인증 완료 후 API KEY 수신
- ◆ API KEY는 이메일로 전송됩니다.

The screenshot shows the '인증키 신청' (API Key Application) form. At the top, there's a section for '이용형태' (Usage Type) with radio buttons for '개인' (Individual) and '단체' (Corporate). Below this is a disclaimer about the number of API keys and a note about the application fee. The form is divided into several sections: '본인 확인' (Self-Verification) with a '본인 인증' button; '이메일' (Email) with fields for email and a verification code, and an '이메일 인증' button; '사용용도' (Usage Purpose) with radio buttons for 'WEB', 'APP', and 'WEB+APP', and a text field for '사용URL' (Usage URL) which contains 'http://'. A red box highlights the '사용용도' and '사용URL' fields. At the bottom right, there is a blue button labeled '인증키 신청'.

3) 환경변수 파일(.env) 설정

- `django-environ` 패키지 설치
 - `pip install django-environ`
- `.env` 파일 예시
 - `API_KEY="발급받은_인증키"`
- `.gitignore`에 `.env` 파일 추가하여 외부 노출 방지
- `settings.py` 에서 환경변수 불러오기 예시

```
import environ
env = environ.Env()
environ.Env.read_env()
API_KEY = env('API_KEY')
```

4) 개발언어 및 툴

- Python, Django, Django REST Framework

5) 필수 라이브러리 / 오픈소스

- Requests
- Django / djangorestframework
- Django-environ
- Sqlite3

3. 작업 순서

- 1) 팀원과 같이 요구사항(기본/심화)을 확인한다.
- 2) 개발 사항 및 수정(개선) 등 할 일을 정하고 역할을 분담한다.
- 3) 요구사항에 맞는 기능을 구현하고 테스트한다.
- 4) 팀원과 함께 추가적인 아이디어를 논의하고 정리한다.
- 5) 추가 아이디어를 적용하여 프로젝트를 완성한다.
- 6) 산출물을 정리하여 GitLab에 반영한다.

4. 요구사항

이번 관통 프로젝트는 금융상품통합비교공시 API(정기예금 API)를 활용하여 **금융상품정보 API Server**를 구축하는 것을 목표로 한다.

외부 금융감독원 API로부터 데이터를 수집하고, Django REST Framework 기반의 서버를 통해 데이터를 가공·저장·조회하는 RESTful 서비스로 구현한다. 이 프로젝트는 서비스 구조를 **REST API 방식으로 설계**하여, 클라이언트가 요청 시 JSON 형태로 금융상품 데이터를 제공하는 **백엔드 핵심 기능**을 완성하는 것을 중점으로 한다.

또한, 환경변수(.env)와 django-environ을 활용하여 **API KEY 보안 관리**를 적용하고, Postman을 통해 API 호출 및 응답 결과를 테스트한다.

심화 단계에서는 생성형 AI를 활용하여 **금융상품 관련 더미 데이터를 생성 및 검증**하는 기능을 추가로 구현한다.

● 요구사항 예시

- 아래에 제시된 기능 명세에 맞는 기능을 구현한다.
- 필수 기능은 반드시 구현해야 하며, 심화 기능은 팀 내 기획에 따라 선택적으로 구현한다.

번호	분류	요구사항명	요구사항 상세	우선순위
기능적 요구사항				
F01	데이터 수집	정기예금 상품 및 옵션 목록 저장	금융상품통합비교공시 API를 활용해 정기예금 상품 및 옵션 정보를 수집하고 DB에 저장한다.	필수
F02	데이터 조회	전체 정기예금 상품 목록 출력	DB에 저장된 모든 정기예금 상품 정보를 JSON 형태로 반환한다.	필수
F03	데이터 입력	정기예금 상품 추가	요청 본문(POST 요청)을 통해 새로운 금융상품 데이터를 DB에 추가한다.	필수
F04	데이터 조회	특정 상품 옵션 리스트 출력	상품 코드(fin_prdt_cd)를 기준으로 해당 상품의 옵션 정보를 조회하여 반환한다.	필수
F05	데이터 분석	최고 금리 상품 조회	모든 상품 중 최고 우대금리가 가장 높은 상품과 그 옵션을 함께 반환한다.	필수
F06	생성형 AI	더미 데이터 생성	생성형 AI를 활용하여 실제 은행상품 형태의 더미 데이터를 생성하고 JSON 형태로 제출한다.	심화
...
비기능적 요구사항				
NF01	환경 설정	API Key 관리	금융감독원 API Key를 발급받고 .env 파일로 환경변수를 관리한다.	필수
NF02	보안성	외부 Key 비공개 처리	.gitignore에 .env 파일을 등록하여 API Key가 외부에 노출되지 않도록 한다.	필수
NF03	개발 효율성	코드 구조화	Django의 앱 구조를 준수하며, Model / View / Serializer / URL 구조를 명확히 구분한다.	필수
NF04	문서화	README 작성	구현 과정, 학습 내용, 어려웠던 점, 느낀	필수

			점 등을 README.md에 상세히 기록한다.	
NF05	테스트	Postman 검증	Postman을 활용해 각 API 엔드포인트의 요청 및 응답 결과를 검증한다.	필수
...

1) 기본(필수) 기능

본 프로젝트는 금융상품통합비교공시 API를 활용하여 **정기예금 상품 정보를 수집·저장하고, REST API 형태로 제공하는 금융상품정보 서버**를 구현하는 것을 목표로 한다.

Django 프레임워크와 Django REST Framework를 기반으로 외부 API 연동, 데이터베이스 관리, 직렬화(Serializer), RESTful 응답 구조를 학습하며, Postman을 통한 API 테스트 및 응답 검증 과정을 경험한다.

이를 통해 **단순한 데이터 출력이 아닌 실제 API 서버 구축 경험**을 쌓고, 향후 금융상품 추천 서비스의 백엔드 기초를 마련한다.

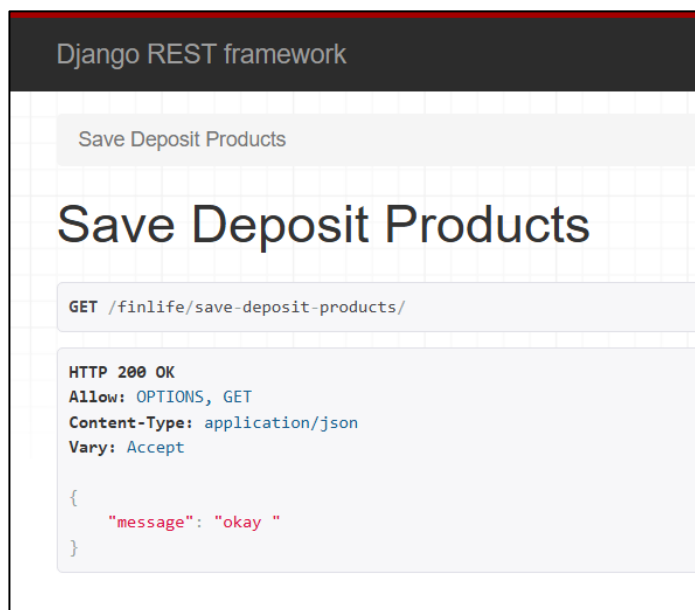
A. 정기예금 상품 및 옵션 목록 저장 (요구사항 F01)

- 금융감독원 금융상품통합비교공시 API(정기예금 API)를 호출하여 전체 상품 목록과 옵션 데이터를 수집한다.
- requests 라이브러리를 사용하여 API 응답(JSON)을 받아온다.
- 수집된 데이터는 Django ORM을 통해 각각의 테이블에 저장한다.
 - 상품 정보 → **DepositProducts** 모델
 - 옵션 정보 → **DepositOptions** 모델
 - ◆ DepositProducts를 외래키로 참조
- 데이터 저장 시 금리가 비어 있을 경우 -1로 처리한다.
- 결과물 예시

■ Vscode sqlite extension으로 저장 확인

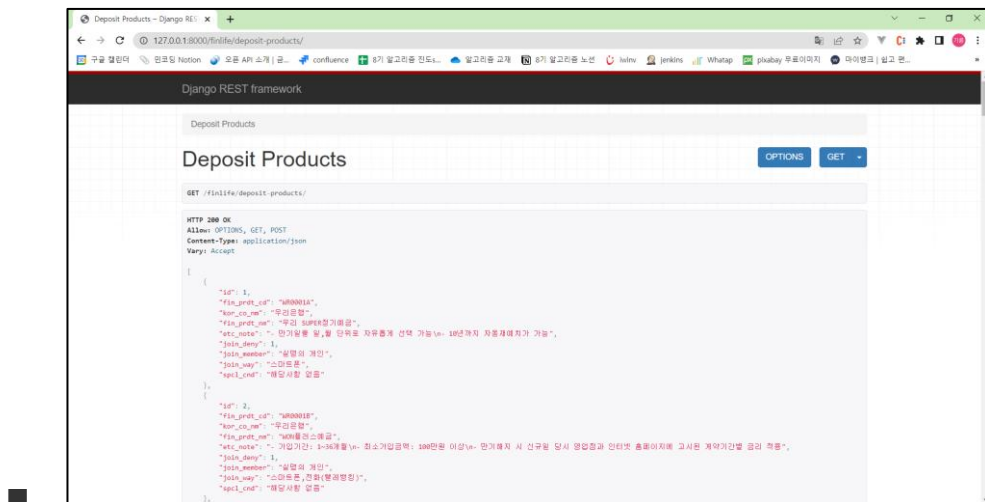
	id	fin_prdt_cd	kor_co_nm	fin_prdt_nm	etc_note
	Filter...	Filter...	Filter...	Filter...	Filter...
1	1	WR0001B	우리은행	WON플러스예금	- 가입기간: 1~36개월 - 최소가입금액
2	2	00320342	한국스탠다드차타드은행	e-그린세이브예금	디지털채널 전용상품 (인터넷, 모바일)
3	3	10511008000996000	아이엘뱅크	iM주거대출예금(첫만납고객형)	계좌당 가입 최저한도 : 100만원
4	4	10511008001004000	아이엘뱅크	iM행복파트너예금(일반형)	계좌당 가입 최저한도 : 100만원
5	5	10511008001166004	아이엘뱅크	iM함께예금	계좌당 가입 최저한도 : 100만원
6	6	10511008001278000	아이엘뱅크	iM스마트예금	계좌당 가입 최저한도 : 100만원
7	7	01030500510002	부산은행	LIVE정기예금	1. 가입금액 : 1천만원 이상 2. 가입기
8	8	010305005600002	부산은행	더(The) 특판 정기예금	1. 가입금액 : 1백만원 이상 제한없음
9	9	010305006000002	부산은행	더(The) 레벨업 정기예금	1. 가입금액 : 1백만원 이상 제한없음
10	10	TD11300027000	광주은행	미즈월복리정기예금	1. 가입기간 : 1년이상 3년제 2. 가입금
11	11	TD11300031000	광주은행	스마트모아Dream정기예금	1. 가입기간 : 1개월이상 3년제 2. 최소

■ 응답 예시



B. 전체 정기예금 상품 목록 출력 (요구사항 F02)

- DB에 저장된 **모든 정기예금 상품 정보를 JSON 형태로 반환**한다.
- Django REST Framework의 Serializer를 활용해 직렬화한다.
- 상품별 주요 필드(kor_co_nm, fin_prdt_nm, join_way, spcl_cnd 등)를 응답에 포함한다.
- 결과물 예시



C. 정기예금 상품 추가 (요구사항 F03)

- 사용자가 직접 금융상품 정보를 추가할 수 있는 API를 구현한다.
- POST 요청을 통해 전달된 JSON 데이터를 기반으로 DB에 새로운 상품을 저장한다.
- 정상 저장 시 "데이터 삽입 성공" 메시지를 반환한다.
- 필드 누락, 중복 코드 등 예외 상황에 대한 적절한 오류 메시지를 반환한다.

- 데이터 예시

```
{

  "fin_prdt_cd": "TEST001",

  "kor_co_nm": "SSAFY은행",

  "fin_prdt_nm": "그레잇예금",

  "join_deny": 1,

  "join_member": "실명의 개인",

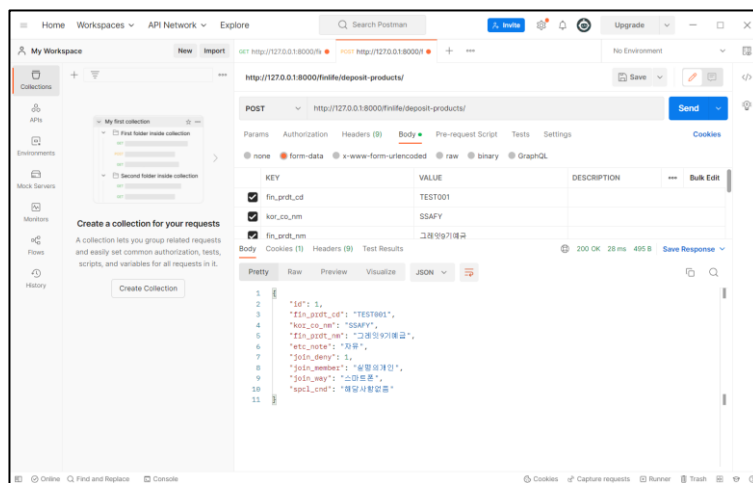
  "join_way": "스마트폰",

  "spcl_cnd": "해당사항 없음"

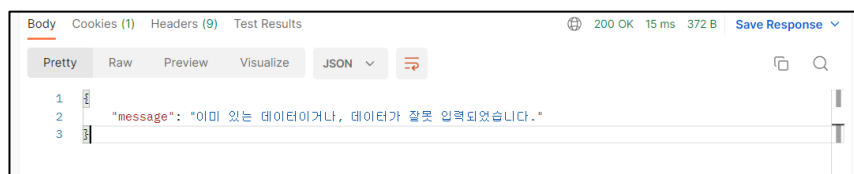
}
```

- 결과물 예시

- POSTMAN으로 저장된 결과를 확인합니다.

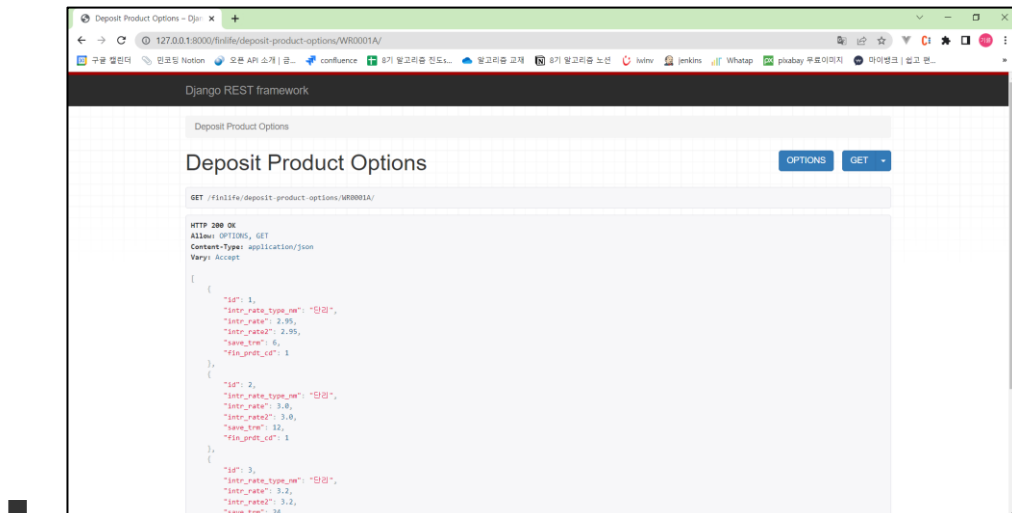


- 삽입 성공/실패 시 적절한 출력 제공



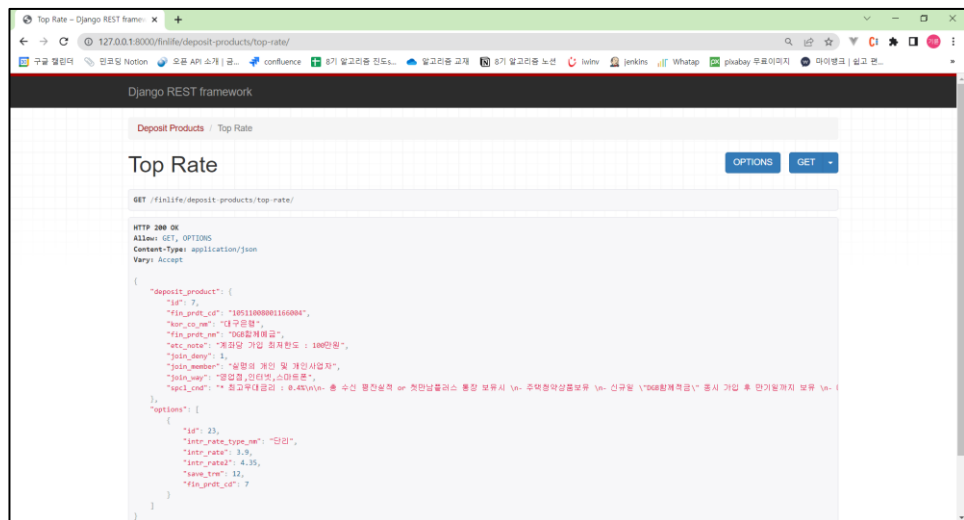
D. 특정 상품 옵션 리스트 출력 (요구사항 F04)

- 사용자가 특정 금융상품의 상품 코드(fin_prdt_cd)를 입력하면, 해당 상품과 연결된 옵션 정보(금리, 기간, 유형 등)를 반환한다.
- DB에서 외래키 관계를 기반으로 데이터를 조회하여 Serializer로 JSON 변환한다.
- 요청 URL 예시
 - /finlife/deposit-product-options/<str:fin_prdt_cd>/
- 응답 예시
 - fin_prdt_cd의 옵션 리스트
- 결과물 예시



E. 최고 금리 상품 조회 (요구사항 F05)

- 모든 금융상품 옵션 중 최고 우대금리(intr_rate2)가 가장 높은 상품을 찾아 반환한다.
- 해당 옵션과 연결된 상품 정보를 함께 제공한다.
- 가입 기간(save_trm)에 상관없이 최고 금리가 기준이 된다.
- 결과물 예시



2) 심화 기능

- 사용자에게 적합한 금융 상품을 추천하기 위해서는 **충분한 학습 데이터**가 필요하지만, 실제로 수집 가능한 데이터는 매우 제한적이다.
- 이에 따라, 수집된 **실제 금융 상품 데이터와 생성형 AI를 결합하여 더미 데이터를 생성**한다.
- 생성된 더미 데이터는 **향후 추천 알고리즘 학습 및 검증 과정에서 활용**할 수 있다.

F. 생성형 AI 기반 더미 데이터 생성 (요구사항 F06)

- 생성형 AI(OpenAI, Gemini 등)를 활용하여 실제 은행 상품 형태의 더미 데이터를 생성한다.
- AI가 생성한 데이터를 JSON 형식으로 구성하여 dummy_data.json 형태로 저장한다.
- **생성된 데이터의 구조와 타입이 실제 DB 모델과 동일하도록 검증**한다.
- **생성된 데이터 고려 사항**
 - 실제로 추천 알고리즘 등에 사용할만한 현실성 있는 데이터인가?
 - 결측치나 타입 오류는 존재하지 않는가?
 - 중복 데이터는 없는가?
- 생성된 dummy_data.json 을 함께 제출한다.
 - 생성한 데이터는 향후 추천 알고리즘 구현 시 활용할 수 있다.
 - 생성 데이터의 개수는 제한 없이 자유롭게 구성한다.

5. 참고자료

- 금융상품통합비교공시 메인 페이지
 - <https://finlife.fss.or.kr/finlife/main/contents.do?menuNo=700029>
- Django Docs
 - <https://docs.djangoproject.com/en/5.2/>
- Django Rest Framework Docs
 - <https://www.django-rest-framework.org/>
- Postman
 - <https://www.postman.com/>

6. 결과

최종적으로 제출해야 할 항목은 아래와 같다

1. 구현 소스 코드

- Django 프로젝트 전체 소스코드
- 환경변수 파일(.env)은 제외하고 제출한다.

2. 실행 결과 캡처본

- 각 요구사항(F01~F06)에 대한 **API 실행 결과 화면(Postman 또는 Browser 출력 화면)** 캡처본

3. README.md

- 구현 기능 설명, 학습 내용, 느낀 점 포함

위 내용들을 모두 포함하여 GitLab에 업로드 한다.

GitLab 프로젝트 이름은 프로젝트 번호 + _pjt 로 지정한다 (07_pjt)