SAMSUNG SW AI ACADEMY FOR YOUTH

對外秘

프로젝트 명세서

파이썬을 활용한 API 데이터 수집

(파이썬 트랙)

“파이썬을 활용한 데이터 수집 기본”

1 회차

對外秘

목차

PJT 명	파이썬을 활용한 API 데이터 수집	
단계	01 PJT	
진행일자	2026.01.23	
예상 구현 시간	필수기능	5H
	심화기능	3H

1. 파이썬을 활용한 API 데이터 수집

1.1 목표

이번 '파이썬을 활용한 API 데이터 수집' 프로젝트의 목표는 다음과 같습니다.

- Python 기초 문법을 이해하고 활용할 수 있다.
- 외부 API(JSON) 사용법을 이해하고 활용할 수 있다.
- JSON 데이터를 분석하고, 원하는 형태로 가공할 수 있다.

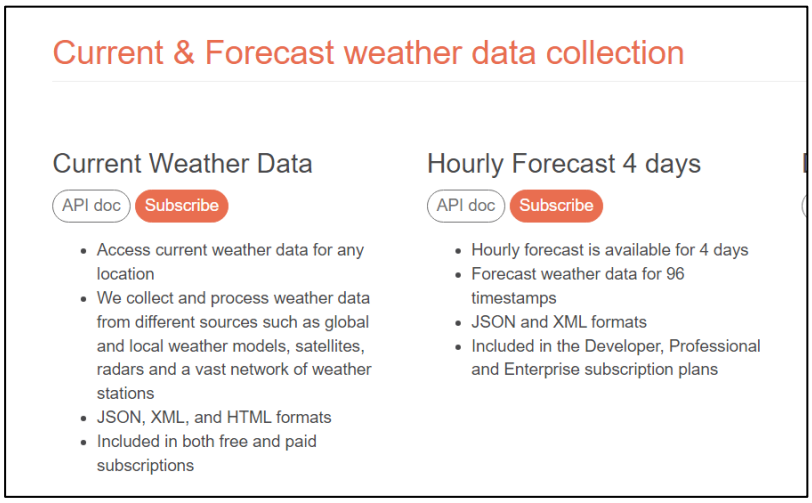
1.2 준비사항

1) 프로젝트 구조

- 프로젝트는 총 두 개의 폴더가 제공됨
  - skeleton/weather 폴더: 필수 요구사항 구현을 위한 스켈레톤 코드
  - skeleton/deposit 폴더: 심화 요구사항 구현을 위한 스켈레톤 코드

2) 사용 데이터

- OpenWeatherMap API: https://openweathermap.org/api
  - 무료 날씨 데이터 API
  - API 종류: Current Weather Data

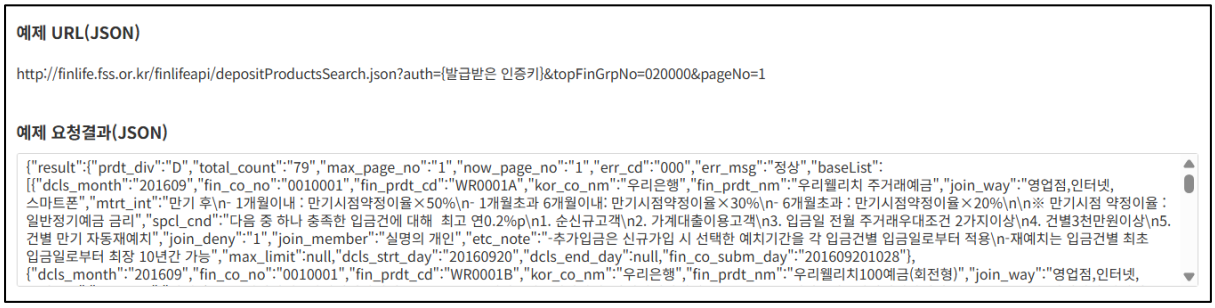


<그림 1> OpenWeatherMap API 공식 사이트 화면 일부

<그림 2> Current Weather Data 답변 예시

- 금융상품통합비교공시 API (심화)
  - https://finlife.fss.or.kr/finlife/main/contents.do?menuNo=700029
  - API 종류: 정기예금 API

<그림 3> 금융상품비교공시 API 소개 페이지



<그림 4> 정기예금 API 답변 예시

3) 개발언어 및 툴

- Python (Visual Studio Code)

4) 필수 라이브러리 / 오픈소스

- requests

1.3 작업 순서

1) 팀원과 같이 요구사항(기본/추가/심화)을 확인한다.
2) 각자 GitLab 에 프로젝트를 생성한다. (프로젝트 이름은 01-pjt 로 지정한다.)
3) 각 반 담당 강사님을 Maintainer 로 설정한다.
4) 페어와 함께 요구사항을 분석하고 필요한 코드를 파악한다.
5) 필수 요구사항을 구현한다.
6) [선택] 심화 과제에 대해 페어와 함께 정리 후 구현한다.
7) README 를 작성한다.
8) 제출 기한에 맞춰 모든 산출물이 GitLab 에 업로드한다.

1.4 요구사항

본 프로젝트는 외부 API(OpenWeatherMap, 금융상품통합비교공시 API)를 통해 데이터를 수집하고, 해당 데이터를 분석 및 가공하여 사용자 맞춤형 날씨 서비스로 확장할 수 있는 기반을 마련하는 것을 목표로 한다. 단순한 API 연동을 넘어서 데이터의 구조를 이해하고, 필요한 정보를 추출하며, 사용 목적에 맞게 가공하는 일련의 과정을 수행한다. 또한, 팀원과 협업하여 아래 요구사항을 구현하고, 향후 실사용 서비스를 위한 기술을 익힌다.

이번 관통 프로젝트는 금융 상품 비교 애플리케이션 프로젝트의 필수 기능 중 하나인 외부 API 를 통한 데이터 수집 및 가공 기능을 중점적으로 구현한다.

이를 통해 현실 데이터 기반의 서비스 설계 경험을 쌓고, 다양한 외부 데이터를 목적에 맞게 정제 및 가공하여 활용할 수 있는 데이터 처리 역량을 강화하는 것을 목표로 한다.

■ 요구사항 예시 (참고용)

- 아래의 내용을 참고하여 추가적인 아이디어에 대해 요구사항을 추가 또는 수정하여 기능을 구현한다. 단, 필수 기능은 반드시 구현해야 하며 수정할 수 없다.

對外秘

기능적 요구사항				
번호	분류	요구사항명	요구사항 상세	우선순위
F101	날씨	Key 값 출력	날씨 데이터의 응답을 JSON 형태로 변환 후 Key 값만 출력하도록 구성	필수
F102	날씨	원하는 값 추출	main, weather 키 값을 딕셔너리로 추출	필수
F103	날씨	Key 한글화	추출한 키 값을 한글 키로 변환한 새로운 딕셔너리 생성	필수
F104	날씨	섭씨 온도 추가	F03의 결과 딕셔너리에 온도 관련 값을 이용해 섭씨 데이터 필드 추가	필수
F105	날씨	생성형 AI 활용	OpenWeatherMap API에 대한 프롬프트 구성 및 실행	필수
F111	금융	Key 값 출력	전체 정기예금의 응답을 JSON 형태로 변환 후 Key 값만 출력하도록 구성	심화
F112	금융	정기예금 리스트 추출	전체 응답 중 정기예금 상품 리스트 정보만 출력	심화
F113	금융	옵션 정보 가공	전체 응답 중 정기예금 상품들의 옵션 리스트를 출력	심화
F114	금융	상품+옵션 통합	금융 상품 + 옵션 정보 딕셔너리로 가공	심화
F115	금융	생성형 AI 활용	금융 API 기반 프롬프트 작성 및 결과 출력	심화
…	…	…	…	…

비기능적 요구사항				
번호	분류	요구사항명	요구사항 상세	우선순위
NF101	환경설정	API Key 발급	API 사용을 위해 공식 사이트에서 API Key를 발급받고, 코드로 활용할 수 있도록 구성	필수
NF102	UX	코드 사용성	코드가 직관적으로 작성되어야 하며, 주요 로직에 주석이 포함되어야 함	필수
NF103	효율성	함수 구조화	기능별로 함수가 분리되어 있어야 하며, 코드 중복을 최소화해야 함	필수
NF104	문서화	README 작성	README.md에 구현 설명, 학습 내용, 느낀 점 등을 상세히 기록해야 함	필수

…	…	…	…	…

1. 기본 기능 (필수)

요구사항 구현 전, API Key 발급 및 공식 API 문서를 통해 구조와 사용법을 사전 확인한다.

API 사용 시점에 따라 결과물은 문서 예시와 다르게 출력될 수 있다.

생성형 AI 를 제외한 전체 기능은 함수 단위로 구현하고, 사용자 입력 없이 실행 가능한 구조로 작성한다. (생성형 AI 는 생성형 AI 서비스가 제공하는 화면에서 진행)

1) 응답 데이터에서 Key 값들만 따로 추출하여 출력

① 요구사항 번호 : F101

② 현재 날씨 데이터를 API 를 통해 가져온다.

③ API 의 응답 중 Key 값들만 따로 출력하여 상세 구조를 파악한다.

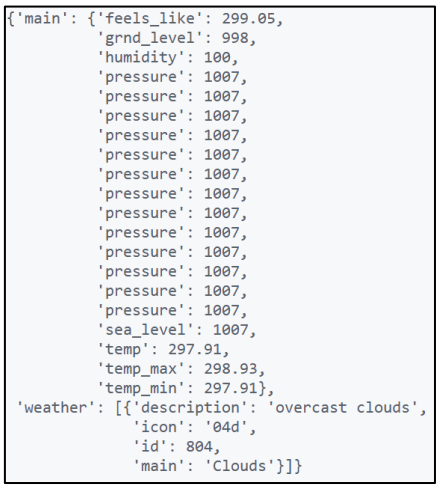
```
dict_keys(['coord', 'weather', 'base', 'main', 'visibility', 'wind', 'clouds',
'dt', 'sys', 'timezone', 'id', 'name', 'cod'])
```

<그림 5> 현재 날씨 데이터 Key 값 출력 예시

2) 특정 데이터만 추출하여 딕셔너리로 구성하여 출력

① 요구사항 번호 : F102

② Key 값이 (main, weather) 인 데이터만 따로 딕셔너리로 구성하여 출력한다.

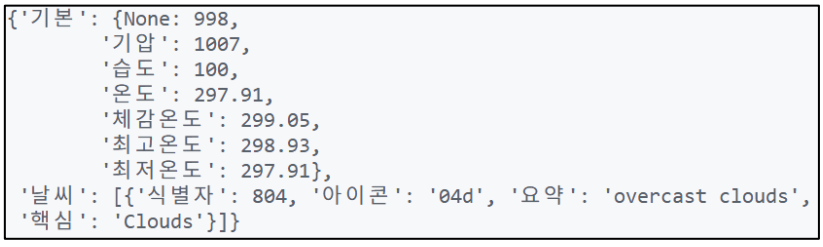


<그림 6> main, weather 데이터 출력 예시

3) Key 값들을 모두 한글로 변환

① 요구사항 번호 : F103

② Key 값들을 모두 한글로 변환한 새로운 딕셔너리를 구성하여 출력한다.



<그림 7> 한글 변환 후 딕셔너리 출력 결과 예시

4) 데이터 가공 (섭씨 온도 추가)

① 요구사항 번호 : F104

② F103 의 결과 딕셔너리에 온도 관련 값을 이용해 섭씨 데이터 필드 추가


```
{'기본': {None: 998,
        '기압': 1007,
        '습도': 100,
        '온도': 297.91,
        '온도(섭씨)': 24.76,
        '체감온도': 299.05,
        '체감온도(섭씨)': 25.9,
        '최고온도': 298.93,
        '최고온도(섭씨)': 25.78,
        '최저온도': 297.91,
        '최저온도(섭씨)': 24.76},
 '날씨': [{'식별자': 804, '아이콘': '04d', '요약': 'overcast clouds',
 '핵심': 'Clouds'}]}
```


<그림 8> 온도(섭씨) 데이터 추가

5) 생성형 AI 활용하기

① 요구사항 번호 : F105

② 생성형 AI 가 OpenWeatherMap API 를 이해하고 설명할 수 있도록 프롬프트를 구성한다.

- ◆ 생성형 AI 가 답변해주는 내용과 공식 문서를 직접 비교하고 차이점이 있다면 정확한 답변을 할 수 있도록 프롬프트를 구성한다.

③ 생성형 AI 가 답변할 수 있는 날씨와 관련된 질문을 자유롭게 구성하고, 정답을 얻을 수 있도록 프롬프트 구성한다.

- ◆ 질문 예시: 서울의 3 일 후 날씨를 알기 위해서 어떻게 요청을 보내야 해?

④ 생성형 AI 대화와 결과물을 캡처하여 함께 제출한다.

- ◆ AI 서비스 종류와 결과물(AI 답변, 코드 출력 결과 등)은 자유롭게 구성한다.

2. 심화 기능 (선택)

추후 금융 상품 비교 애플리케이션 구현을 위해 금융감독원 금융상품통합비교공시 API 를 활용하여 정기예금 상품 정보 및 옵션 정보를 수집하고, 해당 데이터를 정제 및 가공하여 활용 가능한 데이터 형태로 변환하는 로직을 구현.

사용자는 응답 받은 JSON 데이터에서 원하는 키를 추출하고, 한글화 및 추가 가공을 통해 정보를 읽기 쉽게 정리할 수 있어야 한다.

금융상품통합비교공시 API 구조 및 사용법 문서를 반드시 확인 후 심화 기능을 구현한다.

6) 응답 데이터에서 Key 값들만 따로 추출하여 출력

① 요구사항 번호 : F111

② 정기예금 상품 데이터를 API 를 통해 가져온다.

③ API 의 응답 중 Key 값들만 따로 출력하여 상세 구조를 파악한다.

```
dict_keys(['prdt_div', 'total_count', 'max_page_no', 'now_page_no',
'err_cd', 'err_msg', 'baseList', 'optionList'])
```

<그림 9> 정기 예금 데이터 Key 값 출력 예시

7) 상품 리스트 정보 출력

① 요구사항 번호 : F112

② 새로운 딕셔너리에 정기예금 **상품 리스트 정보만 구성**하여 출력한다.

◆ 전체 데이터는 상품 정보와 금리 정보로 이루어져 있다.


```
[{'dcls_end_day': None,
  'dcls_month': '202506',
  'dcls_strt_day': '20250709',
  'etc_note': '- 가입기간: 1~36개월\n'
              '- 최소가입금액: 1만원 이상\n'
              '- 만기일을 일,월 단위로 자유롭게 선택 가능\n'
              '- 만기해지 시 신규일 당시 영업점과 인터넷 홈페이지에 고시된 계약기간별
금리 적용',
  'fin_co_no': '0010001',
  'fin_co_subm_day': '202507091007',
  'fin_prdt_cd': 'WR0001B',
  'fin_prdt_nm': 'WON플러스예금',
  'join_deny': '1',
  'join_member': '실명의 개인',
  'join_way': '인터넷,스마트폰,전화(텔레뱅킹)',
  'kor_co_nm': '우리은행',
  'max_limit': None,
  'mtrt_int': '만기 후\n'
              '- 1개월이내 : 만기시점약정이율×50%\n'
              '- 1개월초과 6개월이내: 만기시점약정이율×30%\n'
              '- 6개월초과 : 만기시점약정이율×20%\n'
              '\n'
              '※만기시점 약정이율 : 일반정기예금 금리',
  'spcl_cnd': '해당사항 없음'},
 {'dcls_end_day': '99991231',
  'dcls_month': '202506',
  'dcls_strt_day': '20250620',
```


<그림 10> 상품 정보 목록 출력 예시

8) 옵션 정보 리스트 출력

① 요구사항 번호 : F113

② 각 상품의 옵션 정보만 따로 추출하여 리스트로 가공 후 출력한다.

◆ 옵션 정보: 금리유형, 금리, 기간 등


```
[{'금융상품코드': 'WR0001B',
  '저축 금리': 2.5,
  '저축 기간': '1',
  '저축금리유형': 'S',
  '저축금리유형명': '단리',
  '최고 우대금리': 2.5},
 {'금융상품코드': 'WR0001B',
  '저축 금리': 2.5,
  '저축 기간': '3',
  '저축금리유형': 'S',
  '저축금리유형명': '단리',
  '최고 우대금리': 2.5},
 {'금융상품코드': 'WR0001B',
  '저축 금리': 2.5,
  '저축 기간': '6',
  '저축금리유형': 'S',
```


<그림 11> 정기 예금 데이터 옵션 리스트 출력 예시

9) 상품, 옵션 데이터 통합

① 요구사항 번호 : F114

② 모든 데이터를 관리할 수 있도록 하나의 금융 상품에 여러 옵션 정보를 포함하는 구조로 가공하여 딕셔너리를 생성한다

◆ 예시 구조: [{금융회사 1, 상품명, 옵션: [옵션 1, 옵션 2, …]}, {금융회사 2, 상품명 2, 옵션: [옵션 1, 옵션 2, …]}, …]


```
[{'금리정보': [{'저축 금리': 2.5,
              '저축 기간': '1',
              '저축금리유형': 'S',
              '저축금리유형명': '단리',
              '최고 우대금리': 2.5},
             {'저축 금리': 2.5,
              '저축 기간': '3',
              '저축금리유형': 'S',
              '저축금리유형명': '단리',
              '최고 우대금리': 2.5},
             {'저축 금리': 2.5,
              '저축 기간': '6',
              '저축금리유형': 'S',
              '저축금리유형명': '단리',
              '최고 우대금리': 2.5},
             {'저축 금리': 2.5,
              '저축 기간': '12',
              '저축금리유형': 'S',
              '저축금리유형명': '단리',
              '최고 우대금리': 2.5},
             {'저축 금리': 2.45,
              '저축 기간': '24',
              '저축금리유형': 'S',
              '저축금리유형명': '단리',
              '최고 우대금리': 2.45},
             {'저축 금리': 2.45,
              '저축 기간': '36',
              '저축금리유형': 'S',
              '저축금리유형명': '단리',
              '최고 우대금리': 2.45}],
  '금융상품명': 'WON플러스예금',
  '금융회사명': '우리은행'},
```


<그림 12> 하나의 금융 상품에 대한 통합 데이터 출력 예시

10) 생성형 AI 활용하기

① 요구사항 번호 : F115

② 생성형 AI가 금융상품통합비교공시 API를 이해하고 설명할 수 있도록 프롬프트를 구성한다.

- ◆ 생성형 AI가 답변해주는 내용과 공식 문서를 직접 비교하고 차이점이 있다면 정확한 답변을 할 수 있도록 프롬프트를 구성한다.

③ 금융상품통합비교공시의 다양한 API (적금, 대출 등)를 확인하고, 다양한 API를 생성형 AI가 정확한 답변을 할 수 있도록 프롬프트를 구성한다.

- ◆ 생성형 AI가 답변해주는 내용과 공식 문서를 직접 비교하고 차이점이 있다면 정확한 답변을 할 수 있도록 프롬프트를 구성한다.

- ◆ 질문 예시: 금리가 가장 높은 적금 상품을 가져오려면 어떻게 해야 해? 예상되는 결과와 함께 알려줘

<그림 13> 금융상품통합비교공시가 제공하는 API 목록

1.5 참고자료

- Requests 패키지 활용법 참고 문서 - MDN
  https://developer.mozilla.org/ko/docs/Web/API/Request
- OpenWeatherMap API 공식 문서 (Current weather data)
  https://openweathermap.org/current
- 금융감독원 금융상품통합비교공시 API (정기예금 data)
  https://finlife.fss.or.kr/finlife/api/fdrmDpstApi/list.do?menuNo=700052

1.6 프로젝트 결과

최종적으로 제출해야 할 항목은 아래와 같다

■ 산출물과 제출

1) 구현 소스 코드

◆ 완성된 각 문제 별 소스코드 및 실행화면 캡쳐본

◆ 콘솔 출력 결과 등

2) README.md

◆ 단계별로 구현 과정 중 학습한 내용, 어려웠던 부분, 새로 배운 것들 및 느낌 점을 상세히 기록

3) 위 내용을 모두 포함하여 GitLab에 업로드한다.

◆ GitLab 프로젝트 이름은 프로젝트 번호 + pjt 형식으로 지정한다 (01_pjt)

- 以上