데이터 수집을 위한 API 및 크롤링 1회차

🎯 **수업 목표**

1. API를 활용해 데이터 수집을 할 수 있다.

목차

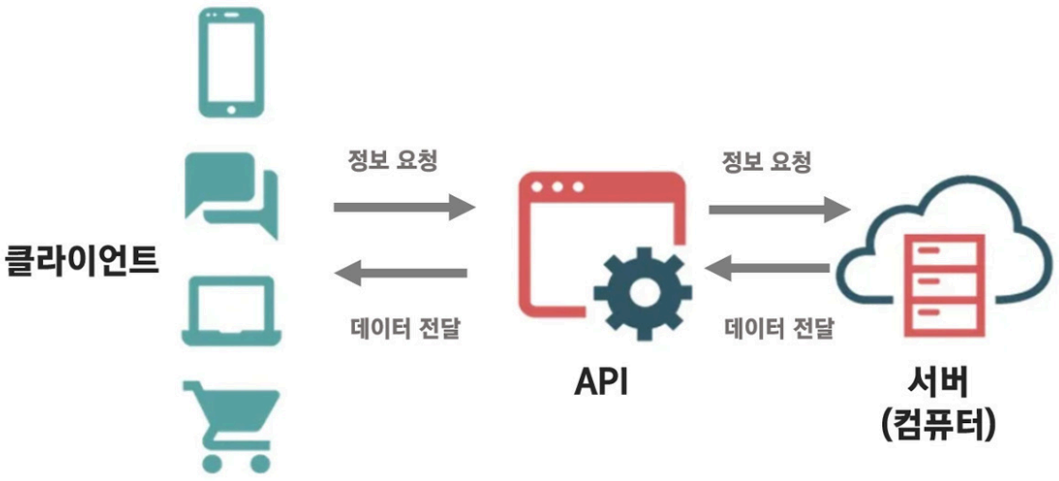
1. API의 이해
2. HTTP의 이해
3. REST API와 URL
4. 오픈 API를 통한 데이터 수집 절차
5. API를 통한 데이터 수집 실습 [실습 자료 미리 다운]
6. API를 통한 데이터 수집 과제

</> **모든 토글을 열고 닫는 단축키**
Windows : Ctrl + alt + t
Mac : ⌘ 알알 + ⌥ + t

▼ 1. API의 이해

API (Application Programming Interface)란?

- 프로그램이나 서비스 간에 기능이나 데이터를 주고받기 위한 인터페이스



API의 주요 역할

1. **서비스 연결**: 서로 다른 프로그램, 애플리케이션, 시스템 간의 통신을 가능하게 합니다.
2. **데이터 교환**: 서버와 클라이언트 간에 데이터를 주고받는 표준화된 방법을 제공합니다.

3. **기능 추상화**: 복잡한 기능을 단순한 인터페이스로 제공하여 개발자가 쉽게 활용할 수 있게 합니다.
4. **서비스 확장**: 기존 시스템의 기능을 외부에 공개하여 새로운 서비스 개발을 촉진합니다.

API 활용 사례

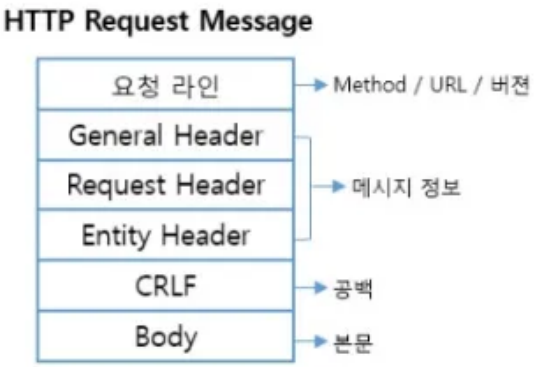
- **결제 시스템**
  - 카카오페이, 토스, 페이팔 등의 API를 통해 쇼핑몰에 안전한 결제 기능 통합
  - 예: 온라인 쇼핑몰에서 다양한 결제 방식 제공
- **지도 및 위치 서비스**
  - 구글 맵, 네이버 지도, 카카오맵 API를 활용한 위치 기반 서비스 개발
  - 예: 음식 배달 앱에서 실시간 배달 위치 추적
- **날씨 정보**
  - 기상청, OpenWeatherMap 등의 API를 통해 실시간 날씨 정보 제공
  - 예: 여행 앱에서 목적지 날씨 정보 표시
- **데이터 분석**
  - 구글 애널리틱스, 네이버 데이터랩 API를 활용한 데이터 수집 및 분석
  - 예: 웹사이트 방문자 통계 자동 수집 및 리포트 생성
- **공공 데이터 활용**
  - 정부나 공공기관에서 제공하는 API를 통해 다양한 공공 서비스 개발
  - 예: 공공데이터포털 API를 활용한 미세먼지 정보 앱

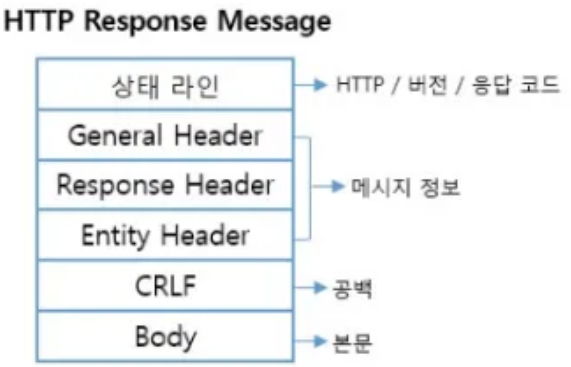
▼ 2. HTTP의 이해

HTTP (Hyper Text Transfer Protocol) 이란?

- 인터넷 환경에서 정보를 주고받기 위한 규칙

HTTP 요청 / 응답 구조





HTTP 요청(Request) 구조

요청 라인(Request Line)

- **구성 요소**: HTTP 메서드 + URL + HTTP 버전

일반 헤더(General Header)

- **역할**: 메시지 전송 시간, 연결 유지 방법, 캐싱 지시사항 등을 포함한 일반적인 정보

요청 헤더(Request Header)

- **역할**: 요청에 대한 추가 정보 및 클라이언트 정보 제공

엔티티 헤더(Entity Header)

- **역할**: 요청 본문에 대한 본문의 데이터 형식, 길이, 압축 방식 등 메타데이터 제공

본문(Body)

- **역할**: 서버로 전송할 실제 데이터

HTTP 응답(Response) 구조

상태 라인(Status Line)

- **구성 요소**: HTTP 버전 + 응답 코드 + 상태 메시지

일반 헤더(General Header)

- **역할**: 응답 생성 시간, 연결 상태 등 포함한 일반적인 정보

응답 헤더(Response Header)

- **역할**: 서버 정보 및 응답에 대한 추가 정보 제공

엔티티 헤더(Entity Header)

- **역할**: 응답 본문에 대한 메타데이터 제공

본문(Body)

- **역할**: 서버가 클라이언트에게 반환하는 실제 데이터

HTTP 요청 메소드

- `GET` : 정보를 가져올 때
- `POST` : 새로운 정보를 생성할 때
- `PUT/PATCH` : 정보를 수정할 때
- `DELETE` : 정보를 삭제할 때

HTTP의 요청 헤더

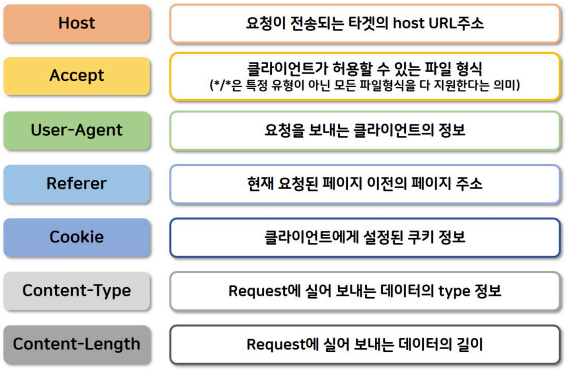
- HTTP 헤더는 웹서버와 브라우저 간의 통신에서 중요한 메타데이터를 전달하는 역할

HTTP Header 구조

```
POST /index.php HTTP/1.1
Host : www.whitehat.co.kr
Accept : text/html, */*;
User-Agent : Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko)
             Chrome/45.0.2454.101 Safari/537.36
Referer : http://www.whitehat.co.kr/
Cookie : PHPSESSID=6htmrajn1fab33s8o2jgns68m1;
Content-Length : 23
Content-Type : application/x-www-from-urlencoded

id=whitehat&pw=password
```

주요 HTTP 헤더 필드



Host

- **예시:** `Host: www.example.com`
- **활용**: 요청을 보내는 대상 서버의 도메인 이름이나 IP 주소를 지정합니다.

User-Agent

- **예시:** `User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) Chrome/91.0.4472.124 Safari/537.36`
- **활용**: 서버는 이 정보를 통해 기기나 브라우저에 최적화된 콘텐츠를 제공할 수 있습니다. 모바일 기기인지 데스크톱인지 구분할 때도 사용됩니다

Authorization

- **예시:** `Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...`
- **활용**: API 사용 시 인증 정보를 전달할 때 사용 (JWT, 베이직 인증 등)

Cookie

- **예시:** `Cookie: session_id=abc123; user_preference=dark_mode`
- **활용**: 로그인 상태 유지, 사용자 설정 기억, 장바구니 정보 등 세션 관리에 사용됩니다.

HTTP 주요 응답 코드

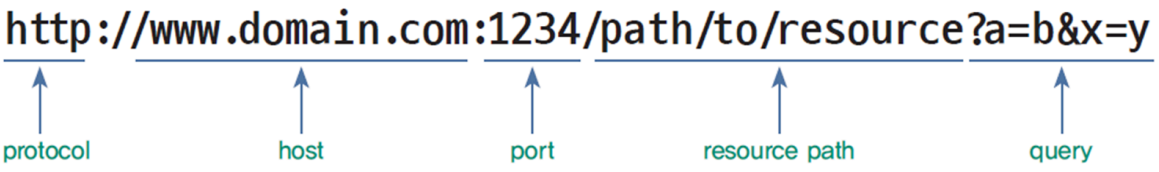
- `200번대`: 성공 (초록불)
- `400번대`: 클라이언트 측 문제 (노란불)
- `500번대`: 서버 측 문제 (빨간불)

▼ 3. REST API와 URL

REST API(Representational State Transfer)란?

- REST API는 인터넷에서 정보를 주고받는 가장 인기 있는 방식
- REST API는 HTTP 프로토콜을 **기반**으로 동작합니다.
- 자원(Resource)을 URI로 표현하고, HTTP 메서드로 자원에 대한 행위를 정의하는 방식입니다.

URL(Uniform Resource Locator) 구조



- **URL은 웹에서 특정 자원의 위치를 나타내는 주소**
- **기본 구조:** `프로토콜://호스트:포트/경로?쿼리스트링`

URL 구성 요소

- **프로토콜(Scheme)**: 통신 규칙

- **호스트(Host)**: 도메인 이름 또는 IP 주소
- **포트(Port)**: 서비스 접속 번호 (생략 가능, HTTP는 80, HTTPS는 443)
- **경로(Path)**: 자원의 위치를 나타내는 문자열
- **쿼리 스트링(Query String)**: 자원에 대한 추가 정보 (필터링, 검색어 등)

REST API의 HTTP 메서드별 역할

- **GET: 자원 조회 (읽기 전용, 안전한 작업)**
- POST: 자원 생성 (서버에 새 데이터 제출, 종종 데이터 수집에서도 활용됨)
- **PUT**: 자원 전체 수정 (지정된 자원의 모든 데이터 변경)
- **PATCH**: 자원 부분 수정 (지정된 자원의 일부 데이터만 변경)
- **DELETE**: 자원 삭제 (지정된 자원 제거)

▼ 4. 오픈 API를 통한 데이터 수집 절차

- **오픈 API란 외부에 공개된 응용 프로그램 인터페이스**로, 누구나 사용할 수 있도록 제공되는 API
- 웹 서비스, 데이터 제공 업체, 공공 기관 등에서 자신들의 **데이터나 서비스에 접근할 수 있는 방법을 표준화하여 제공**
- 대표적인 예: 공공데이터포털 API, 기상청 API, 네이버 검색 API, 카카오 지도 API 등

▼ 데이터 수집 절차

▼ 1) API 이용 준비

- **API 키 발급**
  - 해당 서비스에 회원가입 및 개발자 등록
  - API 키(인증 토큰) 발급 신청
  - 발급받은 API 키는 안전하게 보관 (환경 변수 등으로 관리)
- **API 문서 숙지**
  - API 엔드포인트 및 제공되는 데이터 종류 파악
  - 요청 방법 확인
  - 요청/응답 형식 및 매개변수 확인

▼ 예시

```
기상청 단기예보 API 주요 정보 정리

0. 요청 방법 : GET

1. 기본 URL: http://apis.data.go.kr/1360000/VilageFcstInfoService_2.0

2. 주요 엔드포인트:
  - 단기예보: /getVilageFcst
  - 초단기예보: /getUltraSrtFcst
  - 예보버전 조회: /getFcstVersion

3. 요청 파라미터:
  - serviceKey: 발급받은 API 키 (필수)
  - pageNo: 페이지 번호 (기본값 1)
  - numOfRows: 한 페이지 결과 수 (기본값 10)
  - dataType: 응답 데이터 타입 (XML/JSON)
```

```
  - base_date: 발표 날짜 (YYYYMMDD)
  - base_time: 발표 시간 (HHMM)
  - nx, ny: 예보지점 X, Y 좌표 (기상청 격자 좌표)

4. 응답 형식:
  {
   "response": {
    "header": {
     "resultCode": "00",
     "resultMsg": "NORMAL_SERVICE"
    },
    "body": {
     "dataType": "JSON",
     "items": {
      "item": [
       {
        "baseDate": "20230601",
        "baseTime": "0500",
        "category": "TMP",
        "fcstDate": "20230601",
        "fcstTime": "0600",
        "fcstValue": "22"
       },
       ...
      ]
     },
     "pageNo": 1,
     "numOfRows": 10,
     "totalCount": 60
    }
   }
  }

5. 응답 코드 의미 파악:
  - TMP: 1시간 기온 (°C)
  - REH: 습도 (%)
  - POP: 강수확률 (%)
  - PTY: 강수형태 (코드값)
  - PCP: 1시간 강수량 (mm)
```

- **사용 제한 확인**
  - 일일/분당 호출 제한 확인 (Rate Limit)
  - 무료/유료 플랜에 따른 기능 제한 확인
  - 데이터 사용 범위 및 이용약관 검토

  ▼ 예시

```
네이버 검색 API 사용 제한 정보

1. 호출 한도:
  - 기본 플랜: 하루 25,000회 API 호출 제한
  - 유료 플랜: Enterprise 요금제 (월 100만원부터)

2. 처리량 제한:
  - 초당 10회 호출 제한 (QPS 10)
```

```
   - 초과 시 429 Too Many Requests 오류 발생

3. 콘텐츠 제한:
   - 검색 결과는 최대 100개까지만 제공
   - 블로그 검색은 최대 1,000개까지 페이지네이션 가능

4. 데이터 사용 제한:
   - 네이버 및 검색결과 제공 출처 표시 필수
   - 검색 결과의 재배포 및 저장 금지
   - 상업적 용도로 활용 시 별도 제휴 필요

5. 플랜별 차이점:
   - 기본(무료): 기본 검색 결과만 제공
   - 유료: 추가 필드, 더 많은 결과, 더 높은 호출 한도
```

▼ 2) API 호출 및 데이터 수집

- **기본 요청 테스트**
  - 간단한 API 호출로 연결 및 응답 확인
  - 응답 형식과 구조 파악
- **본격적인 데이터 수집**
  - 필요한 파라미터를 포함한 API 요청 실행

▼ 3) 데이터 처리 및 저장

- **응답 데이터 파싱 및 정제**
  - JSON/CSV 형식의 응답을 객체로 변환
  - 필요한 데이터 추출 및 정제
- **데이터 검증**
  - 누락된 값이나 형식 오류 처리
- **데이터 변환 및 저장**
  - 수집 목적에 맞게 데이터 구조화 (딕셔너리, 데이터 프레임 등)
  - 적절한 형식으로 저장 (CSV, JSON, 데이터베이스 등)

▼ 5. API를 통한 데이터 수집 실습 [실습 자료 미리 다운]

[API KEY 발급 방법 및 실습 파일]

API Key 발급 가이드

📂 **실습 파일**
https://drive.google.com/file/d/109j7r5DG9lVnbBg4zXvOcdCITLN79zeh/view?usp=sharing

1. 네이버 API 실습

- 네이버에서 제공하는 검색 API를 활용하여 블로그 글과 이미지를 검색하고 결과를 가져오는 실습입니다.

준비사항

1. 네이버 개발자 센터(https://developers.naver.com) 회원가입 및 로그인
2. 애플리케이션 등록 (API 이용 신청)
3. 클라이언트 ID와 시크릿 발급

4. Python 및 VSCode 설치
5. VSCode 주피터 환경 설치

2. 공공데이터 API 실습 - 출입국 관광 통계

- 한국관광공사에서 제공하는 출입국 관광 통계 API를 활용하여 특정 국가의 관광객 수를 조회하는 실습입니다. 이 API를 통해 방한 외래관광객과 국민 해외관광객 통계를 수집할 수 있습니다.

준비사항

1. 공공데이터포털(https://www.data.go.kr) 회원가입 및 로그인
2. 출입국 관광 통계 API 활용 신청
3. 서비스 키(API 키) 발급
4. Python 및 VSCode 설치
5. VSCode 주피터 환경 설치

3. 유튜브 API 실습

- 유튜브에서 제공하는 데이터 API를 활용하여 동영상을 검색하고 상세 정보를 가져오는 실습입니다.

준비사항

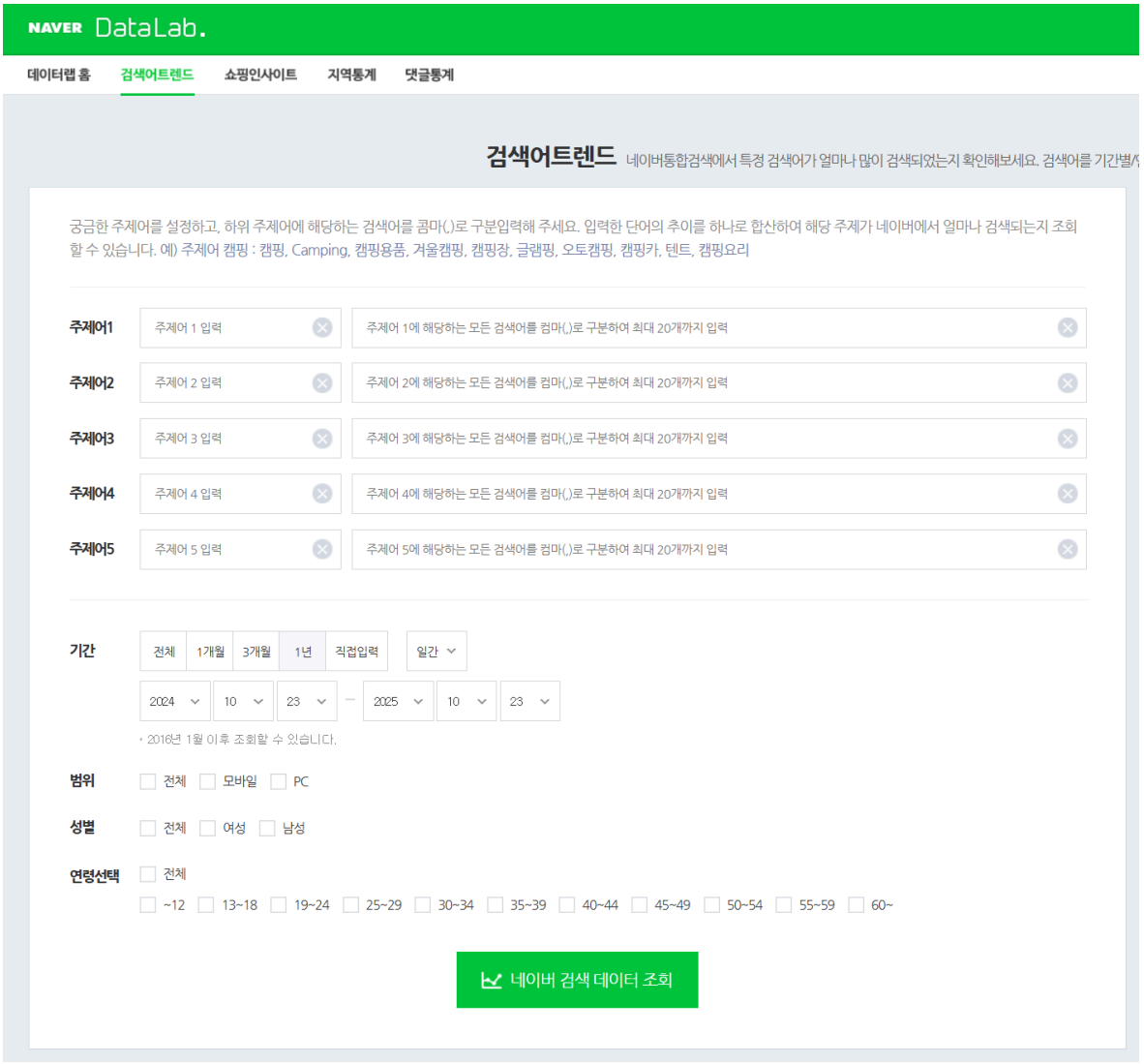
1. Google 개발자 콘솔(https://console.developers.google.com) 회원가입 및 로그인
2. 프로젝트 생성 및 YouTube Data API v3 활성화
3. 서비스 키(API 키) 발급
4. Python 및 VSCode 설치

▼ 6. API를 통한 데이터 수집 과제

과제 1: 네이버 트렌드 API를 활용한 데이터 수집

개요

네이버 데이터랩에서 제공하는 검색어 트렌드 API를 활용하여 특정 키워드의 검색량 추이를 분석하는 과제입니다. 시간에 따른 검색어 관심도 변화를 확인하고 데이터를 수집하여 저장합니다. 아래의 사이트에서 검색어 트렌드를 입력해서 결과를 확인할 수 있습니다.



https://datalab.naver.com/keyword/trendSearch.naver

준비사항

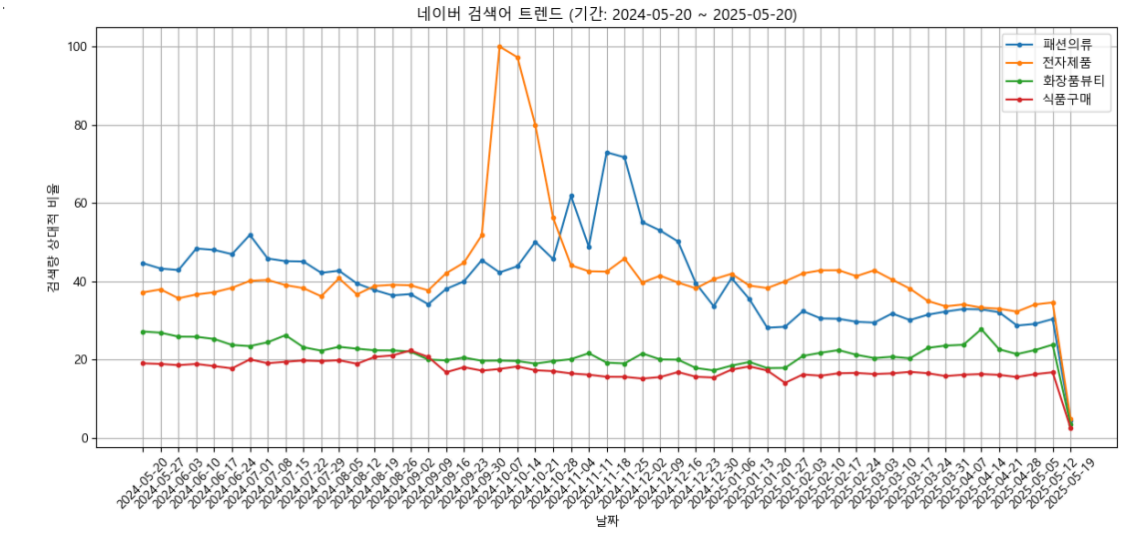
1. 네이버 개발자 센터(https://developers.naver.com) 회원가입 및 로그인
2. 애플리케이션 등록 (데이터랩(검색어 트렌드) API 사용 신청)
3. 클라이언트 ID와 시크릿 발급
4. Python 설치 및 requests 라이브러리 설치
5. VSCode 주피터 환경 설치

과제 요구사항

0. 네이버 데이터랩 검색어 트렌드 API활용 방법에 대해 숙지하세요.
    - https://developers.naver.com/docs/serviceapi/datalab/search/search.md#%ED%86%B5%ED%95%A9-%EA%B2%80%EC%83%89%EC%96%B4-%ED%8A%B8%EB%A0%8C%EB%93%9C
    - 주의 사항 : 보통 `get()` 함수로 호출 데이터를 가져왔으나 여기에서는 `post()` 함수로 데이터를 가져와야합니다. (API 활용 가이드 내용 및 예시 코드 확인)
1. 네이버 데이터랩 검색어 트렌드 API를 활용하여 다음 작업을 수행하세요:
    - 최소 3개 이상의 키워드에 대한 검색 트렌드 데이터 수집
    - 최근 1년간의 주간 트렌드 데이터 수집

- JSON 형식으로 응답 데이터 저장

2. 수집한 데이터를 분석하여:
   - 키워드별 검색량 추이 비교 시각화



▼ 힌트 코드

```
# 네이버 트렌드 API를 이용한 검색어 트렌드 데이터 수집
import requests
import json
import pandas as pd
import matplotlib.pyplot as plt
import os
from datetime import datetime, timedelta

# API 인증 정보 설정
client_id = "YOUR_ID"
client_secret = "YOUR_SECRET"

# 저장 폴더 생성
output_dir = "./naver_trend_data"
os.makedirs(output_dir, exist_ok=True)

# 현재 날짜 기준으로 1년 전까지의 기간 설정
end_date = datetime.now().strftime("%Y-%m-%d")
start_date = (datetime.now() - timedelta(days=365)).strftime("%Y-%m-%d")

# 분석할 키워드 그룹 설정
keyword_groups = [
  {
    "groupName": "패션의류",
    "keywords": ["남성의류", "여성의류", "아우터", "원피스", "티셔츠", "청바지", "패딩"]
  },
  {
    "groupName": "전자제품",
    "keywords": ["스마트폰", "노트북", "태블릿", "이어폰", "스마트워치", "게이밍", "블루투스"]
  },
  {
```

```
        "groupName": "화장품뷰티",
        "keywords": ["스킨케어", "립스틱", "선크림", "파운데이션", "마스크팩", "향수", "클렌징"]
    },
    {
        "groupName": "식품구매",
        "keywords": ["배달음식", "밀키트", "건강식품", "간편식", "쌀", "과일", "신선식품"]
    }
]

# API 요청 헤더 설정
headers = {
    "X-Naver-Client-Id": client_id,
    "X-Naver-Client-Secret": client_secret,
    "Content-Type": "application/json"
}

# API 엔드포인트
api_url = "https://openapi.naver.com/v1/datalab/search"

# 요청 본문 데이터
request_body = {
    "startDate": start_date,
    "endDate": end_date,
    "timeUnit": "week",  # 주간 단위로 데이터 요청
    "keywordGroups": keyword_groups,
}

# API 요청 실행
response = requests.post(
    api_url,
    headers=headers,
    data=json.dumps(request_body)
)

if response.status_code == 200:
    print("API 요청 성공!")
    result_data = response.json()
```

과제 2: 카카오 맵 API를 활용한 할리스 카페 위치 정보 수집

개요

카카오 맵 API를 활용하여 할리스 카페 매장의 주소를 위도와 경도 좌표로 변환(지오코딩)하고, 이 데이터를 수집하여 저장하는 과제입니다.

준비사항

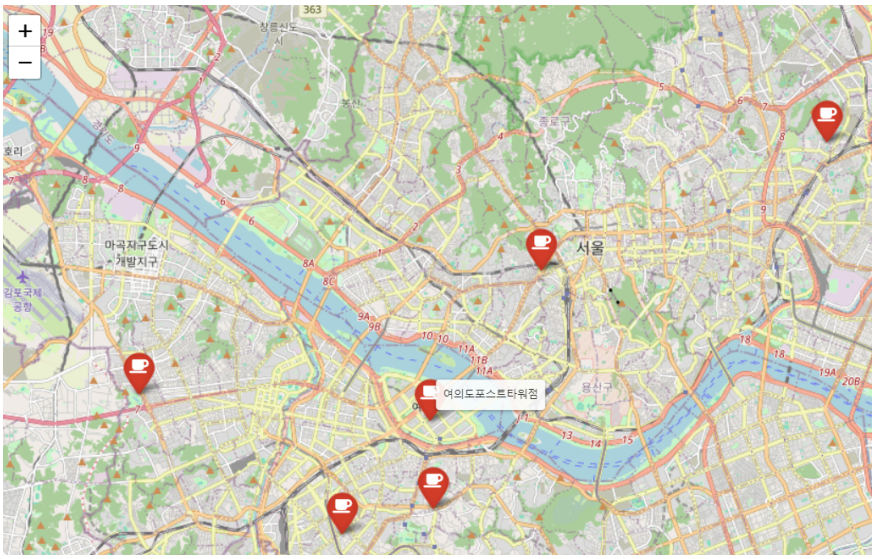
1. 카카오 개발자 사이트(https://developers.kakao.com) 회원가입 및 로그인
2. 애플리케이션 등록 및 API 키 발급
3. REST API 키 확인
4. Python 설치 및 requests 라이브러리 설치
5. VSCode 주피터 환경 설치

과제 요구사항

0. 카카오 로컬 API의 주소 검색 기능 API활용 방법에 대해 숙지하세요.
    - https://developers.kakao.com/docs/latest/ko/local/dev-guide
1. 카카오 로컬 API의 주소 검색 기능을 활용하여:
    - 제공된 할리스 카페 주소 목록에 대한 위도, 경도 정보 수집

      hollys_stores.csv

    - 주소, 매장명, 위도, 경도를 포함한 데이터 저장
    - 오류 발생 시 적절한 예외 처리 구현
2. 수집한 데이터 처리:
    - CSV 파일로 정리하여 저장
    - 선택사항: 수집한 좌표를 지도에 표시(folium 라이브러리 활용)



▼ 힌트 코드

```
# 카카오 맵 API를 활용한 할리스 카페 위치 정보 수집
import requests
import pandas as pd
import time
import os
from datetime import datetime
import folium
from folium.plugins import MarkerCluster

# API 키 설정
API_KEY = "YOUR_API_KEY"  # 카카오 개발자 사이트에서 발급받은 REST API 키로 변경

# 헤더 설정
headers = {
    "Authorization": f"KakaoAK {API_KEY}"
}

# 지오코딩 API 엔드포인트
```

```
api_url = "https://dapi.kakao.com/v2/local/search/address.json"


def geocode_address(address):
    """
    주소를 위도와 경도로 변환하는 함수

    Args:
        address (str): 지오코딩할 주소

    Returns:
        dict or None: 위도, 경도 정보를 담은 딕셔너리, 오류 시 None 반환
    """
    params = {"query": address}

    try:
        response = requests.get(api_url, headers=headers, params=params)

        # API 응답 상태 확인
        if response.status_code == 200:
            result = response.json()

            # 검색 결과가 있는 경우
            if result.get("documents"):
                # 첫 번째 결과 사용
                location = result["documents"][0]

                # 도로명 주소가 있으면 도로명 주소 사용, 없으면 지번 주소 사용
                road_address = location.get("road_address")
                if road_address:
                    address_name = road_address.get("address_name", "")
                else:
                    address_name = location.get("address", {}).get("address_name", "")

                # 위도, 경도 반환
                return {
                    "lat": float(location["y"]),  # 위도
                    "lng": float(location["x"]),  # 경도
                    "address_found": address_name  # 찾은 주소
                }
            else:
                print(f"주소를 찾을 수 없음: {address}")
                return None
        else:
            print(f"API 요청 실패: {response.status_code} - {response.text}")

            # 429 에러(Too Many Requests)인 경우 재시도 안내
            if response.status_code == 429:
                print("API 호출 한도를 초과했습니다. 잠시 후 다시 시도하세요.")

            return None

    except Exception as e:
        print(f"지오코딩 중 오류 발생: {e}")
        return None
```