[스파르타코딩클럽] 웹개발 플러스 - 3주차

매 주차 강의자료 시작에 PDF파일을 올려두었어요!

▼ PDF 파일

▼ 단축키 모음

▼ 새로고침

- `F5`

▼ 저장

- Windows: `Ctrl` + `S`
- macOS: `command` + `S`

▼ 전체선택

- Windows: `Ctrl` + `A`
- macOS: `command` + `A`

▼ 잘라내기

- Windows: `Ctrl` + `X`
- macOS: `command` + `X`

▼ 콘솔창 줄바꿈

- `shift` + `enter`

▼ 코드정렬

- Windows: `Ctrl` + `Alt` + `L`
- macOS: `option` + `command` + `L`

▼ 들여쓰기

- `Tab`
- 들여쓰기 취소 : `Shift` + `Tab`

▼ 주석

- Windows: `Ctrl` + `/`
- macOS: `command` + `/`

[수업 목표]

1. Selenium을 이용해 브라우저를 제어하고 웹스크래핑을 할 수 있다.
2. 내 웹사이트에 네이버 지도를 넣을 수 있다.
3. 네이버 지도 API의 다양한 기능을 활용할 수 있다.

[목차]

모든 토글을 열고 닫는 단축키
Windows : `Ctrl` + `alt` + `t`
Mac : `⌘` + `⌥` + `t`

---

01. 3주차 이번주에 배울 것

▼ 브라우저 제어

- 내가 필요한 정보를 얻기 위해 로그인, 스크롤 내리기 등 브라우저를 동작시켜야 할 때, selenium 같은 브라우저 제어 프로그램을 이용할 수 있습니다.
- 웹스크래핑 뿐만 아니라 브라우저 제어 기능을 응용하면 정해진 시간에 게시판에 글을 작성하는 등 다양한 업무를 자동화하는 데 쓰일 수 있습니다.

▼ 이번주에 만들 웹서비스 구경하기

▼ **[코드스니펫] - 맛집지도 보러 가기**

```
http://spartacodingclub.shop/wp/matjip
```

02. 셀레니움으로 스크래핑하기 - 1

▼ 1) 스크래핑 복습 - 멜론 차트

▼ 웹스크래핑이란?

웹 스크래핑(web scraping)은 웹 페이지에서 우리가 원하는 부분의 데이터를 수집해오는 것을 뜻합니다.

- 한국에서는 같은 작업을 *크롤링 crawling* 이라는 용어로 혼용해서 쓰는 경우가 많습니다. 원래는 크롤링은 자동화하여 주기적으로 웹 상에서 페이지들을 돌아다니며 분류/색인하고 업데이트된 부분을 찾는 등의 일을 하는 것을 뜻해요.
- 구글 검색을 할 때는 **web scraping** 으로 검색해야 우리가 배우는 페이지 추출에 대한 결과가 나올 거예요!
- 참고
  Web Scraping(wikipedia) / Web Crawler(wikipedia)
  Web Scraping vs Web Crawling: What's the Difference?

- 우선 `requests`와 `beautifulsoup4`를 이용해서 멜론 차트를 스크래핑해봅시다.

▼ 이번주도 가장 먼저 할 일은 파이참으로 프로젝트 준비하기!

1. File > New project에 가서 project03 폴더 열기
2. Project Interpreter에서 가상환경에 필요한 패키지 설치하기(`requests`, `beautifulsoup4`, `pymongo`, `flask`, `selenium`)
3. project03 폴더 안에 templates, static 폴더 만들기

4. app.py 파일 만들기

▼ **[코드스니펫] - app.py**

```
from flask import Flask, render_template, request, jsonify, redirect, url_for
from pymongo import MongoClient


app = Flask(__name__)

client = MongoClient('내AWS아이피', 27017, username="아이디", password="비밀번호")
db = client.dbsparta_plus_week3


@app.route('/')
def main():
    return render_template("index.html")


if __name__ == '__main__':
    app.run('0.0.0.0', port=5000, debug=True)
```

5. templates 폴더 안에 index.html 파일 만들기
6. prac_scraping.py 파일 만들기

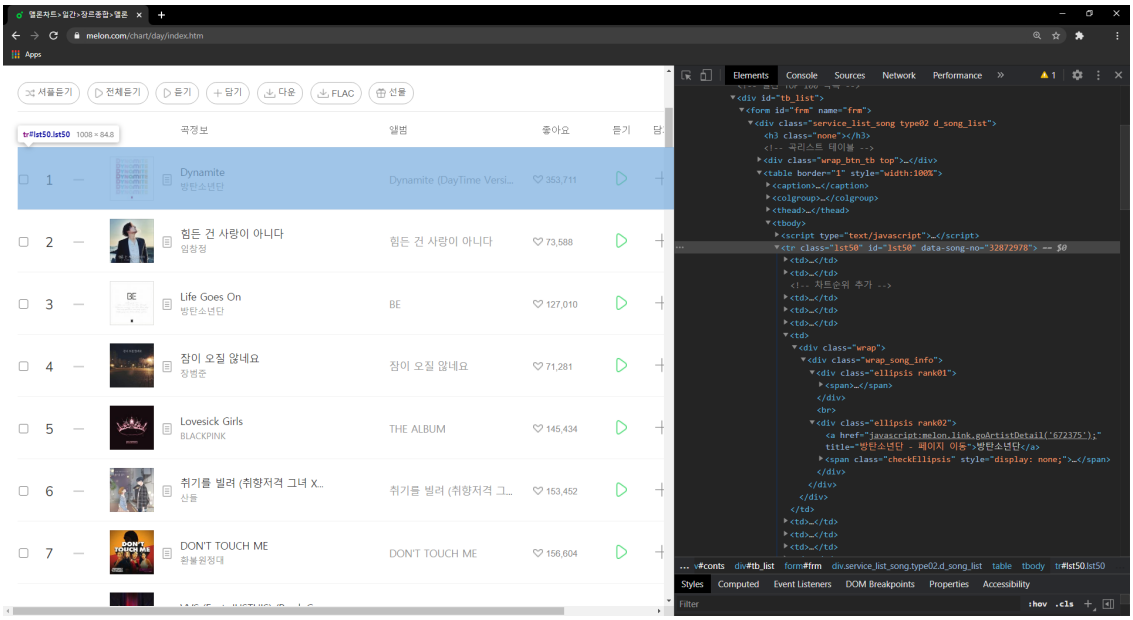
▼ **[코드스니펫] - prac_scraping.py**

```
import requests
from bs4 import BeautifulSoup


url = "https://www.melon.com/chart/day/index.htm"
headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0
data = requests.get(url, headers=headers)

req = data.text
soup = BeautifulSoup(req, 'html.parser')
```

- 크롬에 멜론 차트를 띄우고 개발자도구를 열어 HTML 구조를 살펴봅시다. 각 노래 제목, 가수 이름, 좋아요 수는 어떻게 가져와야 할까요?

▼ **[코드스니펫] - 멜론 차트**

```
https://www.melon.com/chart/day/index.htm
```



▼ **[코드스니펫] 선택자로 정보 출력하기**

```
songs = soup.select("#frm > div > table > tbody > tr")
print(len(songs))

for song in songs:
    title = song.select_one("td > div > div.wrap_song_info > div.rank01 > span > a").text
    artist = song.select_one("td > div > div.wrap_song_info > div.rank02 > span > a").text
    likes = song.select_one("td > div > button.like > span.cnt").text
    print(title, artist, likes)
```

```
100
Dynamite 방탄소년단
총건수
                                                    0


힘든 건 사랑이 아니다 임창정
총건수
                                                    0


Life Goes On 방탄소년단
총건수
                                                    0


잠이 오질 않네요 장범준
총건수
                                                    0
```

!? 앗 좋아요 수가 제대로 출력되지 않네요! 왜 그럴까요?

03. 셀레니움으로 스크래핑하기 - 2

▼ 2) 셀레니움 써서 스크래핑하기

- 멜론 차트처럼 동적인 웹페이지를 스크래핑할 때는 브라우저에 띄운 후 소스코드를 가져오는 방법을 써야합니다.

▼ 셀레니움 설치하기

▼ 크롬드라이버 다운로드

- 크롬 브라우저를 실제로 제어하는 chromedriver 파일을 다운로드 받아야합니다.
- 컴퓨터의 운영체제와 크롬 버전에 맞는 드라이버를 받아주세요.

▼ **[코드스니펫] - 크롬브라우저 버전 확인하기**

```
chrome://settings/help
```

▼ **[코드스니펫] - 크롬 드라이버 다운로드 링크**

```
https://chromedriver.storage.googleapis.com/index.html
```

- 압축을 풀어서 chromedriver 파일을 project03 폴더 안에 넣어주세요.

- 파이썬으로 이 크롬 드라이버를 동작시킬 selenium 패키지는 이미 설치했죠?

- 아까 짠 스크래핑 코드에 아래처럼 코드를 바꿔 써주면 셀레니움 적용 끝!

▼ **[코드스니펫] 셀레니움 적용하기**

```
from bs4 import BeautifulSoup
from selenium import webdriver
from time import sleep
```

```
driver = webdriver.Chrome('./chromedriver')  # 드라이버를 실행합니다.


url = "https://www.melon.com/chart/day/index.htm"
# headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3
# data = requests.get(url, headers=headers)

driver.get(url)  # 드라이버에 해당 url의 웹페이지를 띄웁니다.
sleep(5)  # 페이지가 로딩되는 동안 5초 간 기다립니다.

req = driver.page_source  # html 정보를 가져옵니다.
driver.quit()  # 정보를 가져왔으므로 드라이버는 꺼줍니다.

# soup = BeautifulSoup(data.text, 'html.parser')
soup = BeautifulSoup(req, 'html.parser')  # 가져온 정보를 beautifulsoup으로 파싱해줍니다.

songs = soup.select("#frm > div > table > tbody > tr")
print(len(songs))

for song in songs:
    title = song.select_one("td > div > div.wrap_song_info > div.rank01 > span > a").text
    artist = song.select_one("td > div > div.wrap_song_info > div.rank02 > span > a").text
    likes = song.select_one("td > div > button.like > span.cnt").text
    print(title, artist, likes)
```

- 결과가 이런 식으로 나오는군요!

```
100
Dynamite 방탄소년단
총건수
353,741
힘든 건 사랑이 아니다 임창정
총건수
73,599
Life Goes On 방탄소년단
총건수
```

- '총건수'를 지우기 위해서는 이렇게 해줄 수 있습니다.

▼ [코드스니펫] - 총건수 지우기

```
likes_tag = song.select_one("td > div > button.like > span.cnt")
likes_tag.span.decompose()  # span 태그 없애기
likes = likes_tag.text.strip()  # 텍스트화한 후 앞뒤로 빈 칸 지우기
```

▼ 3) 브라우저 제어 - 스크롤, 버튼

- 단순히 HTML을 띄우는 것 뿐만 아니라 셀레니움을 이용해서 스크롤, 버튼 클릭 등 다양한 동작을 할 수 있습니다.
- 네이버 이미지 검색창을 예시로 써서 브라우저 제어를 해보겠습니다.

▼ [코드스니펫] - 네이버 이미지 검색창 스크래핑 코드

```
from bs4 import BeautifulSoup
from selenium import webdriver
from time import sleep


driver = webdriver.Chrome('./chromedriver')

url = "https://search.naver.com/search.naver?where=image&sm=tab_jum&query=%EC%95%84%EC%9D%B4%EC%9C%A0"
driver.get(url)
sleep(3)

req = driver.page_source
driver.quit()

soup = BeautifulSoup(req, 'html.parser')
images = soup.select(".tile_item._item ._image._listImage")
print(len(images))

for image in images:
    src = image["src"]
    print(src)
```

▼ 스크롤 내리기

- 셀레니움에서 스크롤을 내리고 싶을 땐 아래와 같은 코드를 이용합니다.

▼ **[코드스니펫] - 1000픽셀 만큼 스크롤 내리기**

```
driver.execute_script("window.scrollTo(0, 1000)")  # 1000픽셀만큼 내리기
```

- 화면의 맨 밑까지 내리고 싶다면 이렇게 해줄 수 있습니다.

▼ **[코드스니펫] - 맨 밑까지 내리기**

```
sleep(1)
driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
sleep(10)
```

04. 네이버 지도 API

▼ 4) API 정보 보기

- 네이버 지도 API 기술문서에 가면 각 요소의 사용법 뿐만 아니라 다양한 예시를 볼 수 있습니다.

▼ **[코드스니펫] - 네이버 지도 API 기술문서 링크**

```
https://navermaps.github.io/maps.js.ncp/docs/
```

▼ 5) 사용 신청하기

▼ (0) 만약 해외 거주 중이고 네이버 아이디가 없으시다면?

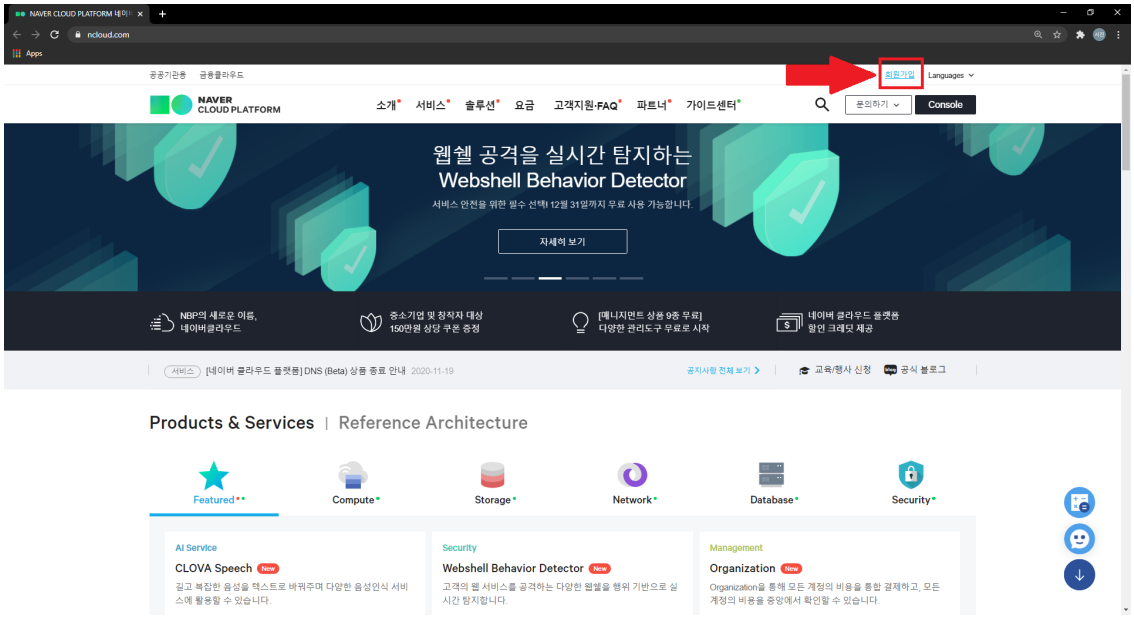
> 👉 키는 저희가 아래에 준비해드렸습니다!
> 아래 (1)~(5) 를 건너뛰고 바로 6) html에 넣어보기 로 넘어가세요.😎

```
3k4qgpyp2e # Client ID
JRVCqPc1ghKstibbccUW1qDzcyTw9GAr0R16lY83 # Client Secret
```

▼ (1) 네이버 클라우드 플랫폼에서 회원가입하기

▼ **[코드스니펫] - 네이버 클라우드 플랫폼 링크**

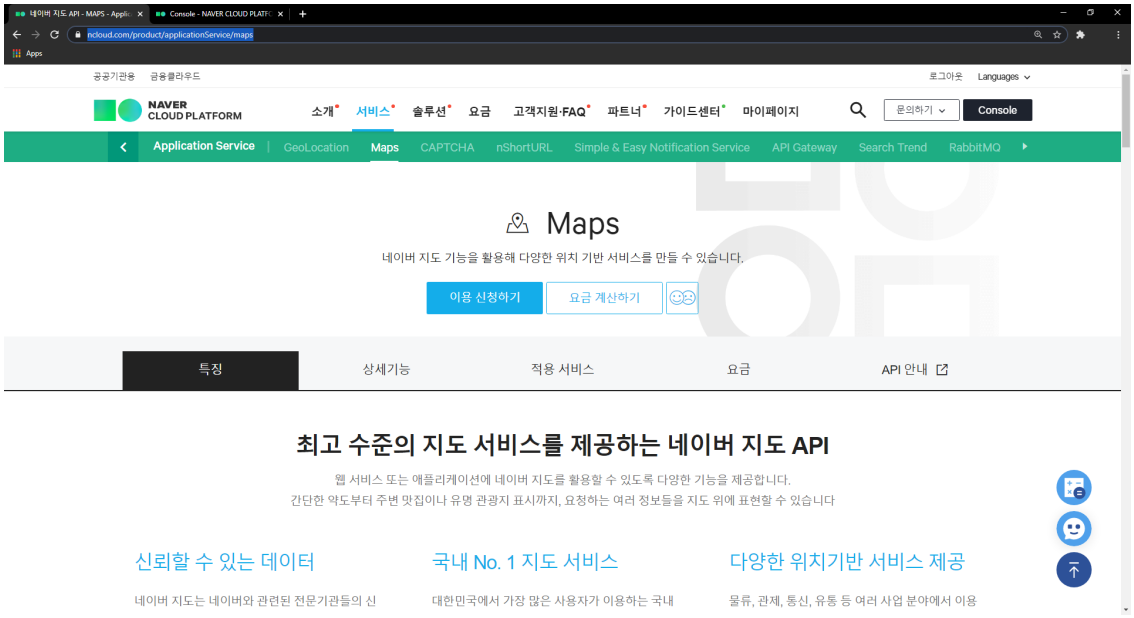
```
https://www.ncloud.com/
```



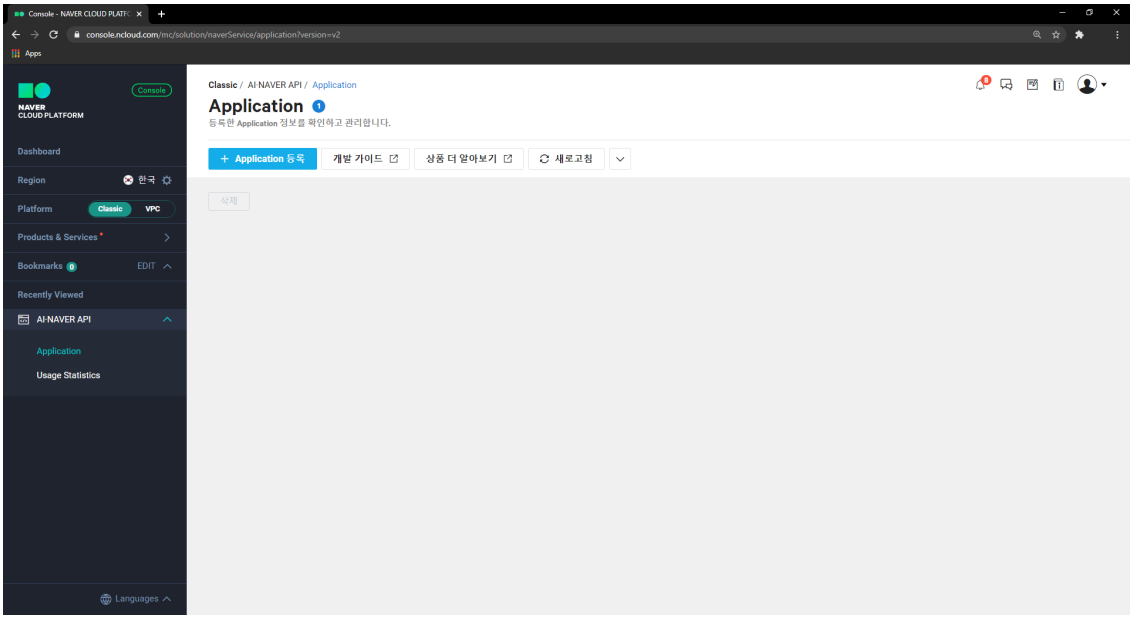
▼ (2) 지도 API 링크에서 이용 신청하기

▼ **[코드스니펫] - 네이버 지도 API 링크**

```
https://www.ncloud.com/product/applicationService/maps
```

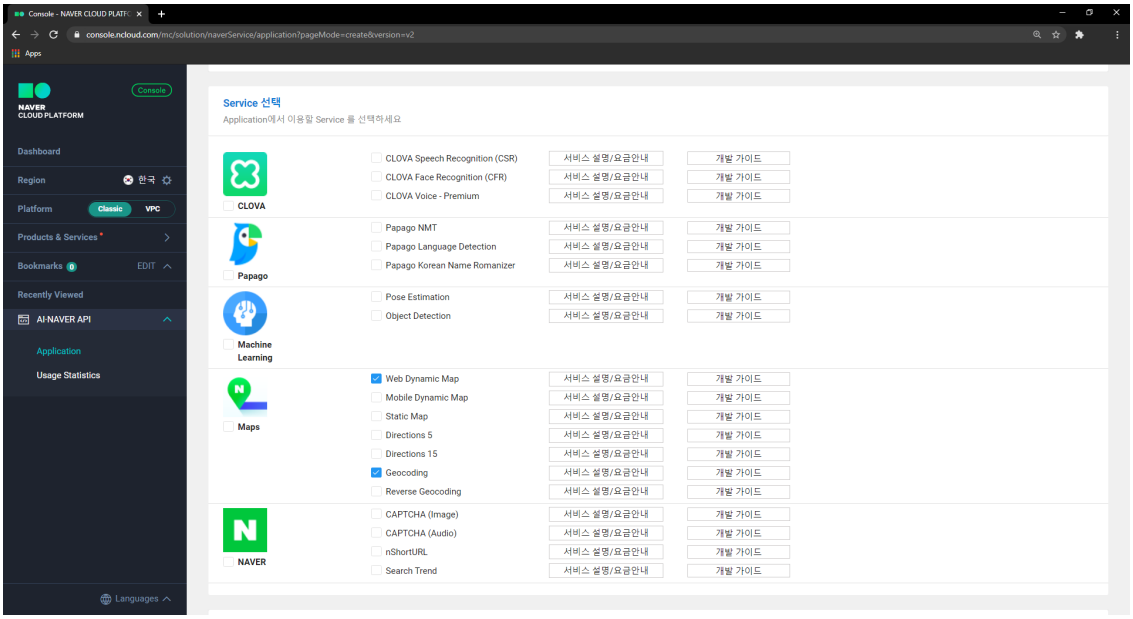


▼ (3) 'Application 등록' 클릭하기

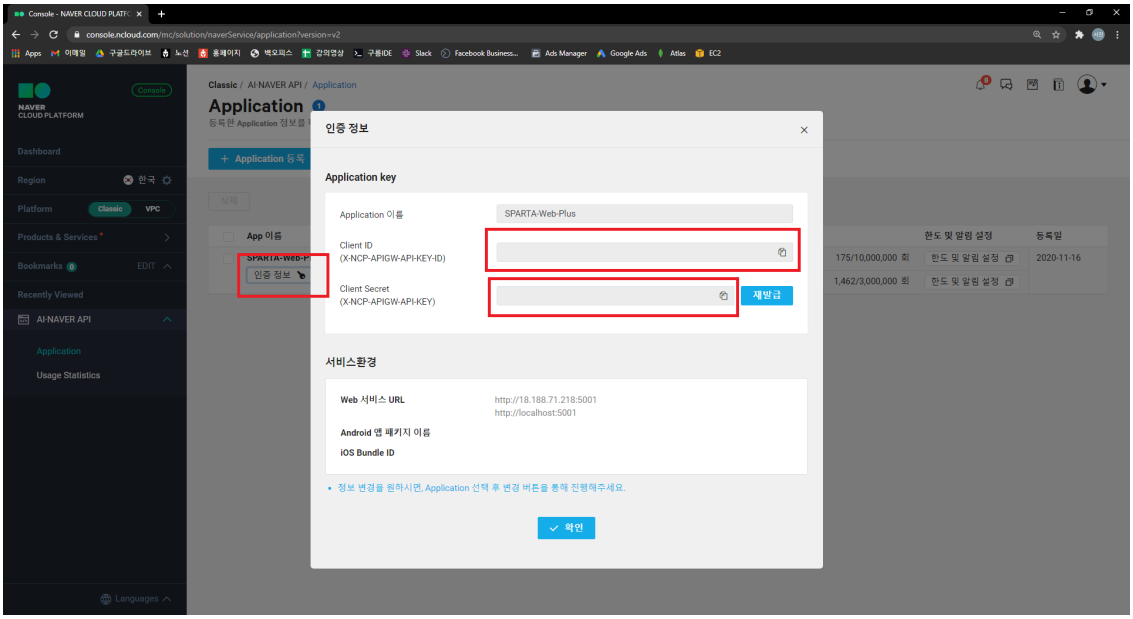


▼ (4) 설정 정보 입력하기

- Application 이름 : 영문, 숫자, - 조합의 원하는 이름 입력 (ex. SPARATA-Web-Plus)
- Maps : Web Dynamic Map와 Geocoding 체크
- Web 서비스 URL : http://localhost:5000 입력 후 '+ 추가' 버튼 클릭
- 입력 완료 후 '등록' 클릭



▼ (5) 지도 API 인증 아이디 확인



▼ 6) html에 넣어보기

▼ templates 폴더에 prac_map.html을 만들어 시작코드를 붙여넣어주세요.

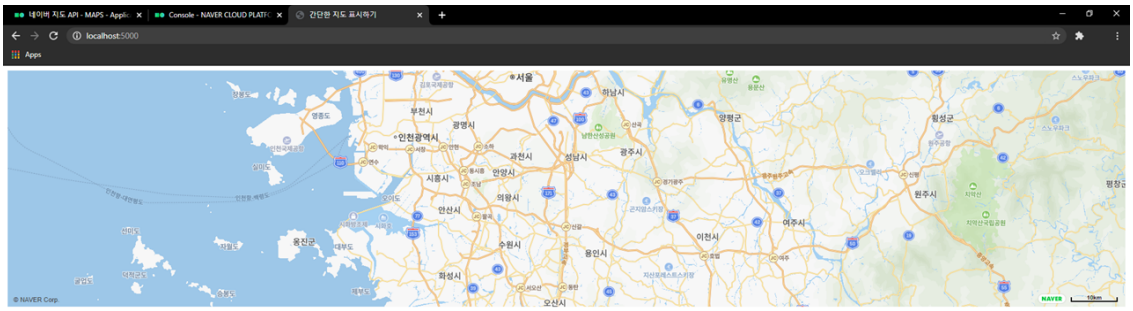
▼ **[코드스니펫] - prac_map.html**

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport"
              content="width=device-width, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0, user-scalable=no">
        <title>간단한 지도 표시하기</title>
        <script type="text/javascript"
                src="https://openapi.map.naver.com/openapi/v3/maps.js?ncpClientId=YOUR_CLIENT_ID"></script>
        <script src=" https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>

        <style>
            #map {
                width: 100%;
                height: 400px;
            }
        </style>

        <script>
            $(document).ready(function () {
                let map = new naver.maps.Map('map', {
                    center: new naver.maps.LatLng(37.4981125, 127.0379399),
                    zoom: 10
                });
            })
        </script>
    </head>
    <body>
        <div id="map"></div>
    </body>
</html>
```

⚠ 네이버 지도 관련 자바스크립트 파일 주소 중 `YOUR_CLIENT_ID` 부분에 콘솔에서 확인한 인증 정보를 넣어야합니다.

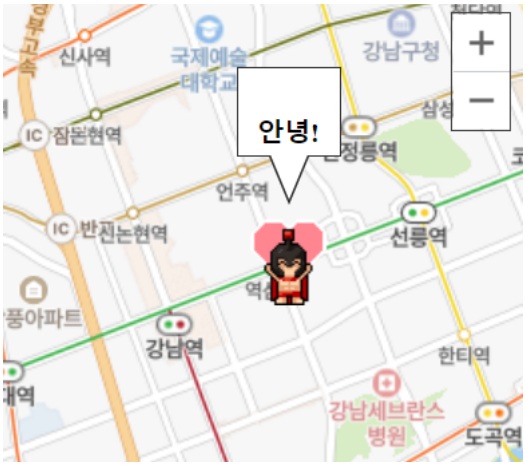
- 브라우저에 띄워볼까요? 앗, 인증 실패 에러가 나는군요! 아까 http://localhost:5000에서만 쓰겠다고 했는데, 파이참에서 바로 브라우저로 띄우면 주소가 http://localhost:63342/week03/templates/prac_map.html?... 이런 식이라 그렇습니다.

- 이번에는 app.py에서 main()에 연결된 html을 prac_map.html로 바꾸고 app.py를 실행한 후 다시 지도를 띄워봅시다. 이렇게 잘 나오면 성공!



05. 네이버 지도 연습하기

▼ 7) 네이버 지도 갖고 놀기



▼ 확대/축소 버튼 넣기

- map의 옵션을 추가해주면 됩니다.

▼ **[코드스니펫] 확대/축소 버튼**

```
let map = new naver.maps.Map('map', {
    center: new naver.maps.LatLng(37.4981125, 127.0379399),
    zoom: 10,
    zoomControl: true,
    zoomControlOptions: {
        style: naver.maps.ZoomControlStyle.SMALL,
```

```
        position: naver.maps.Position.TOP_RIGHT
    }
});
```

▼ 마커 띄우기

- 지도에 마커를 띄우기 위해서는 `marker` 오브젝트를 만들어주어야 합니다.
- `marker`를 얹을 지도(`map`)와 경위도 좌표를 명시해주세요.

▼ **[코드스니펫] 마커 띄우기**

```
let marker = new naver.maps.Marker({
    position: new naver.maps.LatLng(37.4981125, 127.0379399),
    map: map
});
```

▼ 마커 이미지 바꾸기

- 마커 이미지를 다른 모양으로 바꾸고 싶다면, `marker`에 옵션으로 넣어주세요. 이미지 파일은 static 폴더에 넣어주면 되겠죠?

▼ **[코드스니펫] - 마커 이미지**

```
https://s3.ap-northeast-2.amazonaws.com/materials.spartacodingclub.kr/webplus/week03/rtan_heart.png
```

▼ **[코드스니펫] - 마커 이미지 바꾸기**

```
let marker = new naver.maps.Marker({
    position: new naver.maps.LatLng(37.4981125, 127.0379399),
    map: map,
    icon: "{{ url_for('static', filename='rtan_heart.png') }}"
});
```

▼ 정보창 띄우고 닫기

▼ **[코드스니펫] infoWindow 만들고 열기**

```
let infowindow = new naver.maps.InfoWindow({
    content: `<div style="width: 50px;height: 20px;text-align: center"><h5>안녕!</h5></div>`,
});
infowindow.open(map, marker);
```

▼ **[코드스니펫] infoWindow 닫기**

```
infowindow.close();
```

▼ **[코드스니펫] 마커를 누를 때마다 infoWindow 여닫기**

```
naver.maps.Event.addListener(marker, "click", function () {
    console.log(infowindow.getMap()); // 정보창이 열려있을 때는 연결된 지도를 반환하고 닫혀있을 때는 null을 반환
    if (infowindow.getMap()) {
        infowindow.close();
    } else {
        infowindow.open(map, marker);
    }
});
```

06. 프로젝트 3: 맛집 지도

▼ 8) 문제 분석 - 완성작부터 보기

▼ **[코드스니펫] - 맛집지도 보러 가기**

```
http://spartacodingclub.shop/wp/matjip
```

▼ 9) API 설계하기

> ? 필요한 기능들을 생각해볼까요?

- 맛집 정보 스크래핑
- 지도 보여주기
- 각 맛집 별 마커, 정보창, 카드 만들고 서로 연결하기

▼ 10) 프로젝트 준비 - app.py 준비하기

▼ **[코드스니펫] - app.py**

```
from flask import Flask, render_template, request, jsonify, redirect, url_for
from pymongo import MongoClient


app = Flask(__name__)

client = MongoClient('내AWS아이피', 27017, username="아이디", password="비밀번호")
db = client.dbsparta_plus_week3


@app.route('/')
def main():
    return render_template("index.html")


@app.route('/matjip', methods=["GET"])
def get_matjip():
    # 맛집 목록을 반환하는 API
    return jsonify({'result': 'success', 'matjip_list': []})

if __name__ == '__main__':
    app.run('0.0.0.0', port=5000, debug=True)
```

▼ 11) 프로젝트 준비 - index.html 준비하기

▼ **[코드스니펫] - index.html**

```
<!DOCTYPE html>
<html>

    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport"
              content="width=device-width, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0, user-scalable=no">
        <title>스파르타코딩클럽 | 맛집 검색</title>
        <script type="text/javascript"
                src="https://openapi.map.naver.com/openapi/v3/maps.js?ncpClientId=YOUR_CLIENT_ID&submodules=geocoder"></script

        <!-- Bootstrap CSS -->
        <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
              integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
              crossorigin="anonymous">

        <!-- Optional JavaScript -->
        <!-- jQuery first, then Popper.js, then Bootstrap JS -->
        <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
        <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
                integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
                crossorigin="anonymous"></script>
        <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
                integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
                crossorigin="anonymous"></script>
        <style>
            .wrap {

            }
```

```
            .banner {

            }

            .matjip-list {

            }

            #map {
                width: 100%;
                height: 50vh;
                margin: 20px auto 20px auto;
            }
        </style>
        <script>
            let y_cen = 37.4981125   // lat
            let x_cen = 127.0379399  // long
            let map;
            $(document).ready(function () {
                map = new naver.maps.Map('map', {
                    center: new naver.maps.LatLng(y_cen, x_cen),
                    zoom: 12,
                    zoomControl: true,
                    zoomControlOptions: {
                        style: naver.maps.ZoomControlStyle.SMALL,
                        position: naver.maps.Position.TOP_RIGHT
                    }
                });
            })
        </script>

    </head>

    <body>
        <div class="wrap">
            <div class="banner"></div>
            <div id="map"></div>

            <div class="matjip-list" id="matjip-box">
                <div class="card" id="card-0">
                    <div class="card-body">
                        <h5 class="card-title"><a href="#" class="matjip-title">혼가츠</a></h5>
                        <h6 class="card-subtitle mb-2 text-muted">일식</h6>
                        <p class="card-text">서울 마포구 와우산로21길 36-6 (서교동)</p>
                        <p class="card-text" style="color:blue;">생방송 투데이</p>
                    </div>
                </div>
            </div>
        </div>

    </body>

</html>
```

⚠ Client ID 넣는 것 잊지 마세요!

▼ 12) 프로젝트 준비 - 배너 이미지 준비하기

▼ **[코드스니펫] - 배너 이미지**

```
https://s3.ap-northeast-2.amazonaws.com/materials.spartacodingclub.kr/webplus/week03/banner.jpg
```

07. 맛집 정보 스크래핑하기

▼ 13) 스크래핑해 올 사이트 살펴보기

▼ 맛집 정보는 SBS TV 맛집 사이트에서 스크래핑해오겠습니다.

▼ **[코드스니펫] - SBS TV 맛집 링크**

```
http://matstar.sbs.co.kr/location.html
```

- 각 카드 안에 식당의 이름, 주소, 출연 프로그램, 카테고리 등의 정보가 있습니다.
- 페이지 하단의 버튼을 눌러 더 많은 맛집 정보를 받아올 수 있습니다.

▼ 14) 셀레니움으로 스크래핑하기

▼ [코드스니펫] - scraping.py

```
from selenium import webdriver
from bs4 import BeautifulSoup
import time
from selenium.common.exceptions import NoSuchElementException
from pymongo import MongoClient
import requests


client = MongoClient('내AWS아이피', 27017, username="아이디", password="비밀번호")
db = client.dbsparta_plus_week3

driver = webdriver.Chrome('./chromedriver')

url = "http://matstar.sbs.co.kr/location.html"

driver.get(url)
time.sleep(5)

req = driver.page_source
driver.quit()

soup = BeautifulSoup(req, 'html.parser')
```

▼ [코드스니펫] - 각 식당에 해당하는 카드 선택

```
places = soup.select("ul.restaurant_list > div > div > li > div > a")
print(len(places))
```

▼ [코드스니펫] - 식당 이름, 주소, 카테고리, 출연 프로그램과 회차 정보를 출력하기

```
for place in places:
    title = place.select_one("strong.box_module_title").text
    address = place.select_one("div.box_module_cont > div > div > div.mil_inner_spot > span.il_text").text
    category = place.select_one("div.box_module_cont > div > div > div.mil_inner_kind > span.il_text").text
    show, episode = place.select_one("div.box_module_cont > div > div > div.mil_inner_tv > span.il_text").text.rsplit(" ", 1)
    print(title, address, category, show, episode)
```

08. 맛집 정보 좌표로 변환하기

▼ 15) 추가 정보 받기

- 맛집을 지도 위에 나타내기 위해서는 경위도 좌표가 필요합니다. 다행히 네이버에서 제공하는 API 중에 주소를 좌표로 변환해주는 `geocoding` API가 있습니다.

  ▼ [코드스니펫] - geocoding API 참조서 링크

  ```
  https://api.ncloud-docs.com/docs/ai-naver-mapsgeocoding-geocode
  ```

- 사용 신청은 전에 지도 API 신청하면서 같이 했기 때문에, 바로 사용할 수 있습니다.
- 요청을 보낼 때 Client ID와 Client Secret Key 모두 보내주어야합니다.

  ▼ [코드스니펫] - 네이버 클라우드 플랫폼 콘솔

  ```
  https://console.ncloud.com/mc/solution/naverService/application?version=v2
  ```

  ▼ [코드스니펫] - Geocoding 연결하기

```
headers = {
    "X-NCP-APIGW-API-KEY-ID": "[내 클라이언트 아이디]",
    "X-NCP-APIGW-API-KEY": "[내 클라이언트 시크릿 키]"
}
r = requests.get(f"https://naveropenapi.apigw.ntruss.com/map-geocode/v2/geocode?query={address}", headers=headers)
response = r.json()
```

- 주소에 오류가 있어 결과를 하나도 받지 못하는 경우가 있으므로 결과가 있을 때만 값을 출력하도록 합니다.

▼ [코드스니펫] - 결과 출력하기

```
if response["status"] == "OK":
    if len(response["addresses"])>0:
        x = float(response["addresses"][0]["x"])
        y = float(response["addresses"][0]["y"])
        print(title, address, category, show, episode, x, y)
    else:
        print(title, "좌표를 찾지 못했습니다")
```

09. 맛집 정보 DB에 저장하기

▼ 16) 여러 페이지 스크래핑하기

👉 이제 버튼을 클릭하여 더 많은 맛집 정보를 받아올 수 있도록 해보겠습니다!

▼ [코드스니펫] - 더 보기 버튼의 선택자로 버튼 클릭하기

```
btn_more = driver.find_element_by_css_selector("#foodstar-front-location-curation-more-self > div > button")
btn_more.click()
time.sleep(5)
```

▼ [코드스니펫] - 더 보기 버튼을 10번 누르려면?

```
for i in range(10):
    try:
        btn_more = driver.find_element_by_css_selector("#foodstar-front-location-curation-more-self > div > button")
        btn_more.click()
        time.sleep(5)
    except NoSuchElementException:
        break
```

▼ [코드스니펫] - DB에 저장하기

```
doc = {
    "title": title,
    "address": address,
    "category": category,
    "show": show,
    "episode": episode,
    "mapx": x,
    "mapy": y}
db.matjips.insert_one(doc)
```

▼ [코드스니펫] - 스크래핑 완성 코드

```
from selenium import webdriver
from bs4 import BeautifulSoup
import time
from selenium.common.exceptions import NoSuchElementException
from pymongo import MongoClient
import requests


client = MongoClient('내AWS아이피', 27017, username="아이디", password="비밀번호")
```

```
db = client.dbsparta_plus_week3

driver = webdriver.Chrome('chromedriver')

url = "http://matstar.sbs.co.kr/location.html"

driver.get(url)
time.sleep(5)
for i in range(10):
    try:
        btn_more = driver.find_element_by_css_selector("#foodstar-front-location-curation-more-self > div > button")
        btn_more.click()
        time.sleep(5)
    except NoSuchElementException:
        break
req = driver.page_source
driver.quit()

soup = BeautifulSoup(req, 'html.parser')

places = soup.select("ul.restaurant_list > div > div > li > div > a")

print(len(places))
for place in places:
    title = place.select_one("strong.box_module_title").text
    address = place.select_one("div.box_module_cont > div > div > div.mil_inner_spot > span.il_text").text
    category = place.select_one("div.box_module_cont > div > div > div.mil_inner_kind > span.il_text").text
    show, episode = place.select_one("div.box_module_cont > div > div > div.mil_inner_tv > span.il_text").text.rsplit(" ", 1)

    headers = {
        "X-NCP-APIGW-API-KEY-ID": "[내 클라이언트 아이디]",
        "X-NCP-APIGW-API-KEY": "[내 클라이언트 시크릿 키]"
    }
    r = requests.get(f"https://naveropenapi.apigw.ntruss.com/map-geocode/v2/geocode?query={address}", headers=headers)
    response = r.json()
    if response["status"] == "OK":
        if len(response["addresses"]) > 0:
            x = float(response["addresses"][0]["x"])
            y = float(response["addresses"][0]["y"])
            print(title, address, category, show, episode, x, y)
            doc = {
                "title": title,
                "address": address,
                "category": category,
                "show": show,
                "episode": episode,
                "mapx": x,
                "mapy": y}
            db.matjips.insert_one(doc)
        else:
            print(title, "좌표를 찾지 못했습니다")
    else:
        print(response["status"])
```

10. 웹사이트 모습 만들기

▼ 17) 배너, 지도, 카드영역 만들고 꾸미기

▼ [코드스니펫] - HTML

```
<div class="banner">
    <div class="d-flex flex-column align-items-center"
         style="background-color: rgba(0,0,0,0.5);width: 100%;height: 100%;">
        <h1 class="title mt-5 mb-2">스파르타 맛집 지도</h1>
    </div>
</div>
```

▼ [코드스니펫] - CSS

```
.wrap {
    width: 90%;
    max-width: 750px;
    margin: 0 auto;
}

.banner {
    width: 100%;
    height: 20vh;
```

```
    background-image: url("{{ url_for('static', filename='banner.jpg') }}");
    background-position: center;
    background-size: contain;
    background-repeat: repeat;

}

h1.title {
    color: white;
    font-size: 3rem;
}

.matjip-list {
    overflow: scroll;
    width: 100%;
    height: calc(20vh - 30px);
    position: relative;
}

.card-title, .card-subtitle {
    display: inline;
}
```

▼ 18) 구글 웹폰트 적용하기

▼ [코드스니펫] - 구글 웹폰트 링크

```
https://fonts.google.com/?subset=korean
```

- 마음에 드는 글씨체를 선택한 후, 링크 태그와 CSS 규칙을 복사해 붙여넣습니다.

▼ [코드스니펫] - 주아체 link 태그

```
<link rel="preconnect" href="https://fonts.gstatic.com">
<link href="https://fonts.googleapis.com/css2?family=Jua&display=swap" rel="stylesheet">
```

▼ [코드스니펫] - 주아체 CSS

```
h1.title {
    font-family: 'Jua', sans-serif;

    color: white;
    font-size: 3rem;
}

h5 {
    font-family: 'Jua', sans-serif;
}
```

▼ 19) DB에서 맛집정보 받아오기

▼ [코드스니펫] - 서버

```
@app.route('/matjip', methods=["GET"])
def get_matjip():
    # 맛집 목록을 반환하는 API
    matjip_list = list(db.matjips.find({}, {'_id': False}))
    # matjip_list 라는 키 값에 맛집 목록을 담아 클라이언트에게 반환합니다.
    return jsonify({'result': 'success', 'matjip_list': matjip_list})
```

▼ [코드스니펫] - 클라이언트

```
function get_matjips() {
    $('#matjip-box').empty();
    $.ajax({
        type: "GET",
        url: '/matjip',
        data: {},
        success: function (response) {
            let matjips = response["matjip_list"]
            for (let i = 0; i < matjips.length; i++) {
                let matjip = matjips[i]
                console.log(matjip)
```

```
            }
        }
    });
}
```

▼ 20) 카드 만들기

- 각 맛집 별로 카드 하나 씩 만드는 함수를 만듭니다.

```
function make_card(i, matjip) {
    let html_temp = `<div class="card" id="card-${i}">
                        <div class="card-body">
                            <h5 class="card-title"><a href="#" class="matjip-title">${matjip['title']}</a></h5>
                            <h6 class="card-subtitle mb-2 text-muted">${matjip['category']}</h6>
                            <p class="card-text">${matjip['address']}</p>
                            <p class="card-text" style="color:blue;">${matjip['show']}</p>
                        </div>
                    </div>`;
    $('#matjip-box').append(html_temp);
}
```

11. 정보 추가하기

▼ 21) 마커 띄우기

👉 markers 라는 리스트 아래에 marker 를 만들어 넣은 뒤,
markers 를 한 번에 지도에 표시해야 합니다.

▼ [코드스니펫] - 맛집 정보 marker 로 만들고 markers 에 저장하기

```
function make_marker(matjip) {
    let marker = new naver.maps.Marker({
        position: new naver.maps.LatLng(matjip["mapy"], matjip["mapx"]),
        map: map
    });
    markers.push(marker)
    return marker
}
```

▼ 22) 정보창 띄우기

👉 marker 의 정보를 보여줄 infoWindows 라는 리스트도 만들어볼까요?

▼ [코드스니펫] - infoWindows 만들기

```
function add_info(i, marker, matjip) {
    let html_temp = `<div class="iw-inner">
                        <h5>${matjip['title']}</h5>
                        <p>${matjip['address']}
                        </div>`;
    let infowindow = new naver.maps.InfoWindow({
        content: html_temp,
        maxWidth: 200,
        backgroundColor: "#fff",
        borderColor: "#888",
        borderWidth: 2,
        anchorSize: new naver.maps.Size(15, 15),
        anchorSkew: true,
        anchorColor: "#fff",
        pixelOffset: new naver.maps.Point(10, -10)
    });
    infowindows.push(infowindow)
    naver.maps.Event.addListener(marker, "click", function (e) {
        if (infowindow.getMap()) {
            infowindow.close();
        } else {
            infowindow.open(map, marker);
```

```
        }
    });
}
```

▼ **[코드스니펫] - infoWindows CSS 설정하기**

```
.iw-inner {
    padding: 10px;
    font-size: smaller;
}
```

▼ **[코드스니펫] - infoWindows를 가운데 오게 하기**

```
map.setCenter(infowindow.position)
```

12. 고급 기능 쓰기

▼ 23) 카드 보이게 스크롤 움직이기

▼ **[코드스니펫] - 카드 보이게 스크롤 움직이기**

```
$("#matjip-box").animate({
    scrollTop: $("#matjip-box").get(0).scrollTop + $(`#card-${i}`).position().top
}, 2000);
```

▼ 24) 카드 제목 클릭했을 때 정보창 띄우기

👉 a 태그를 클릭했을 때 새창이 뜨는게 아니라 javascript 함수를 실행하고 싶다면?
→ <a href="javascript:실행함수명()"> ...

▼ **[코드스니펫] - i번째 카드를 눌렀을 때 해당 정보창을 열고 닫는 기능**

```
function click2center(i) {
    let marker = markers[i]
    let infowindow = infowindows[i]
    if (infowindow.getMap()) {
        infowindow.close();
    } else {
        infowindow.open(map, marker);
        map.setCenter(infowindow.position)
    }
}
```

▼ **[코드스니펫] - make_card 안의 a 태그 수정**

```
<a href="javascript:click2center(${i})" class="matjip-title">${matjip['title']}</a>
```

▼ 25) og태그, favicon 넣기

- 이번주도 완성도 있는 사이트를 위해 Open Graph 태그와 favicon을 넣어봅시다. og 이미지는 배너 이미지를 복사해서 사용하고 favicon은 아래 파일을 다운 받아 static 폴더에 넣어줍시다.

  ▼ **[코드스니펫] - favicon**

  ```
  https://s3.ap-northeast-2.amazonaws.com/materials.spartacodingclub.kr/webplus/week03/favicon.ico
  ```

- index.html 위에 링크를 첨부합니다.

  ▼ **[코드스니펫] - og태그와 favicon 링크**

```
<meta property="og:title" content="스파르타 맛집 지도"/>
<meta property="og:description" content="mini project for Web Plus"/>
<meta property="og:image" content="{{ url_for('static', filename='og_image.jpg') }}"/>
<link rel="shortcut icon" href="{{ url_for('static', filename='favicon.ico') }}" type="image/x-icon">
<link rel="icon" href="{{ url_for('static', filename='favicon.ico') }}" type="image/x-icon">
```

▼ 26) 완성 코드

▼ [코드스니펫] - app.py

```
from flask import Flask, render_template, request, jsonify, redirect, url_for
from pymongo import MongoClient


app = Flask(__name__)

client = MongoClient('내AWS아이피', 27017, username="아이디", password="비밀번호")
db = client.dbsparta_plus_week3


@app.route('/')
def main():
    return render_template("index.html")

if __name__ == '__main__':
    app.run('0.0.0.0', port=5000, debug=True)
```

▼ [코드스니펫] - index.html

```
<!DOCTYPE html>
<html>

    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport"
              content="width=device-width, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0, user-scalable=no">
        <title>스파르타코딩클럽 | 맛집 검색</title>
        <meta property="og:title" content="스파르타 맛집 지도"/>
        <meta property="og:description" content="mini project for Web Plus"/>
        <meta property="og:image" content="{{ url_for('static', filename='og_img.png') }}"/>
        <link rel="shortcut icon" href="{{ url_for('static', filename='favicon.ico') }}" type="image/x-icon">
        <link rel="icon" href="{{ url_for('static', filename='favicon.ico') }}" type="image/x-icon">
        <script type="text/javascript"
                src="https://openapi.map.naver.com/openapi/v3/maps.js?ncpClientId=YOUR_CLIENT_ID&submodules=geocoder"></script

        <!-- Bootstrap CSS -->
        <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
              integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
              crossorigin="anonymous">

        <!-- Optional JavaScript -->
        <!-- jQuery first, then Popper.js, then Bootstrap JS -->
        <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
        <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
                integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
                crossorigin="anonymous"></script>
        <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
                integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
                crossorigin="anonymous"></script>
        <link rel="preconnect" href="https://fonts.gstatic.com">
        <link href="https://fonts.googleapis.com/css2?family=Jua&display=swap" rel="stylesheet">
        <style>
            .wrap {
                width: 90%;
                max-width: 750px;
                margin: 0 auto;
            }

            .banner {
                width: 100%;
                height: 20vh;
                background-image: url("{{ url_for("static", filename="banner.png") }}");
                background-position: center;
                background-size: contain;
                background-repeat: repeat;

            }

            h1.title {
```

```
            font-family: 'Jua', sans-serif;

            color: white;
            font-size: 3rem;
        }

        h5 {
            font-family: 'Jua', sans-serif;
        }

        .matjip-list {
            overflow: scroll;
            width: 100%;
            height: calc(20vh - 30px);
            position: relative;
        }

        .card-title, .card-subtitle {
            display: inline;
        }

        #map {
            width: 100%;
            height: 50vh;
            margin: 20px auto 20px auto;
        }

        .btn-sparta {
            color: #fff;
            background-color: #e8344e;
            border-color: #e8344e;
        }

        .iw-inner {
            padding: 10px;
            font-size: smaller;
        }
    </style>
    <script>
        let y_cen = 37.4981125   // lat
        let x_cen = 127.0379399  // long
        let map;
        let markers = []
        let infowindows = []
        $(document).ready(function () {
            map = new naver.maps.Map('map', {
                center: new naver.maps.LatLng(y_cen, x_cen),
                zoom: 12,
                zoomControl: true,
                zoomControlOptions: {
                    style: naver.maps.ZoomControlStyle.SMALL,
                    position: naver.maps.Position.TOP_RIGHT
                }
            });

            get_matjips()
        })

        function get_matjips() {
            $('#matjip-box').empty();
            markers = []
            infowindows = []
            $.ajax({
                type: "GET",
                url: `/matjip`,
                data: {},
                success: function (response) {
                    let matjips = response["matjip_list"]
                    console.log(matjips.length)
                    for (let i = 0; i < matjips.length; i++) {
                        let matjip = matjips[i]
                        make_card(i, matjip)
                        let marker = make_marker(matjip)
                        add_info(i, marker, matjip)
                    }
                }
            });
        }

        function make_marker(matjip) {
            let marker = new naver.maps.Marker({
                position: new naver.maps.LatLng(matjip["mapy"], matjip["mapx"]),
                map: map
            });
            markers.push(marker)
            return marker
        }
```

```
            function make_card(i, matjip) {
                let html_temp = `<div class="card" id="card-${i}">
                                    <div class="card-body">
                                        <h5 class="card-title"><a href="javascript:click2center(${i})" class="matjip-title">$
                                        <h6 class="card-subtitle mb-2 text-muted">${matjip['category']}</h6>
                                        <p class="card-text">${matjip['address']}</p>
                                        <p class="card-text" style="color:blue;">${matjip['show']}</p>
                                    </div>
                                </div>`;
                $('#matjip-box').append(html_temp);
            }

            function add_info(i, marker, matjip) {
                let html_temp = `<div class="iw-inner">
                                    <h5>${matjip['title']}</h5>
                                    <p>${matjip['address']}
                                    </div>`;
                let infowindow = new naver.maps.InfoWindow({
                    content: html_temp,
                    maxWidth: 200,
                    backgroundColor: "#fff",
                    borderColor: "#888",
                    borderWidth: 2,
                    anchorSize: new naver.maps.Size(15, 15),
                    anchorSkew: true,
                    anchorColor: "#fff",
                    pixelOffset: new naver.maps.Point(10, -10)
                });
                infowindows.push(infowindow)
                naver.maps.Event.addListener(marker, "click", function (e) {
                    console.log("clicked", infowindows.length)
                    if (infowindow.getMap()) {
                        infowindow.close();
                    } else {
                        infowindow.open(map, marker);
                        map.setCenter(infowindow.position)
                        $("#matjip-box").animate({
                            scrollTop: $("#matjip-box").get(0).scrollTop + $(`#card-${i}`).position().top
                        }, 2000);
                    }
                });
            }

            function click2center(i) {
                let marker = markers[i]
                let infowindow = infowindows[i]
                if (infowindow.getMap()) {
                    infowindow.close();
                } else {
                    infowindow.open(map, marker);
                    map.setCenter(infowindow.position)
                }
            }
        </script>

    </head>

    <body>
        <div class="wrap">
            <div class="banner">
                <div class="d-flex flex-column align-items-center"
                     style="background-color: rgba(0,0,0,0.5);width: 100%;height: 100%;">
                    <h1 class="title mt-5 mb-2">스파르타 맛집 지도</h1>
                    <button type="button" onclick="get_matjips()" class="btn btn-sparta">
                        새로고침하고 더 많은 맛집 보기
                    </button>
                </div>
            </div>
            <div id="map"></div>

            <div class="matjip-list" id="matjip-box">
                <div class="card" id="card-0">
                    <div class="card-body">
                        <h5 class="card-title"><a href="#" class="matjip-title">혼가츠</a></h5>
                        <h6 class="card-subtitle mb-2 text-muted">일식</h6>
                        <p class="card-text">서울 마포구 와우산로21길 36-6 (서교동)</p>
                        <p class="card-text" style="color:blue;">생방송 투데이</p>
                    </div>
                </div>
            </div>
        </div>

    </body>

</html>
```

13. 3주차 끝 & 숙제 설명

정보창에 즐겨찾기 표시 기능을 추가해봅시다! 즐겨찾기된 맛집의 마커 모양도 바꿔볼까요?

▼ **[코드스니펫] - 마커 예시 1**

```
https://s3.ap-northeast-2.amazonaws.com/materials.spartacodingclub.kr/webplus/week03/marker-default.png
```

▼ **[코드스니펫] - 마커 예시 2**

```
https://s3.ap-northeast-2.amazonaws.com/materials.spartacodingclub.kr/webplus/week03/marker-liked.png
```

▼ 힌트

- EC2에서도 네이버 지도가 잘 작동하게 하기 위해서는 콘솔에서 내 아이피를 URL 등록을 해주어야합니다.
- ▼ 즐겨찾기 아이콘을 넣기 위해서는 Font Awesome을 임포트해와야합니다.
  - ▼ **[코드스니펫] - Font Awesome 임포트**

```
<link href="//maxcdn.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css" rel="stylesheet">
```

- 도큐먼트를 업데이트할 때 필드를 제거하기 위해서는 `"$unset"`을 사용합니다.

14. 3주차 숙제 답안 코드

▼ **[코드스니펫] - 3주차 숙제 답안 코드**

전체 코드

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/19b08bfe-e824-4fdf-8a33-f92790ac1481/static.zip

▼ app.py

```
from flask import Flask, render_template, request, jsonify, redirect, url_for
from pymongo import MongoClient


app = Flask(__name__)

client = MongoClient('내AWS아이피', 27017, username="아이디", password="비밀번호")
db = client.dbsparta_plus_week3


@app.route('/')
def main():
    return render_template("index.html")


@app.route('/matjip', methods=["GET"])
def get_matjip():
    matjip_list = list(db.matjips.find({}, {"_id": False}))
    # 맛집 목록을 반환하는 API
    return jsonify({'result': 'success', 'matjip_list': matjip_list})


@app.route('/like_matjip', methods=["POST"])
def like_matjip():
    title_receive = request.form["title_give"]
```

```
    address_receive = request.form["address_give"]
    action_receive = request.form["action_give"]
    print(title_receive, address_receive, action_receive)

    if action_receive == "like":
        db.matjips.update_one({"title": title_receive, "address": address_receive}, {"$set": {"liked": True}})
    else:
        db.matjips.update_one({"title": title_receive, "address": address_receive}, {"$unset": {"liked": False}})
    return jsonify({'result': 'success'})


if __name__ == '__main__':
    app.run('0.0.0.0', port=5000, debug=True)
```

▼ index.html

```
<!DOCTYPE html>
<html>

    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport"
              content="width=device-width, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0, user-scalable=no">
        <title>스파르타코딩클럽 | 맛집 검색</title>
        <script type="text/javascript"
                src="https://openapi.map.naver.com/openapi/v3/maps.js?ncpClientId=내클라이언트아이디넣기&submodules=geocoder"></scr

        <!-- Bootstrap CSS -->
        <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
              integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
              crossorigin="anonymous">
        <link href="//maxcdn.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css" rel="stylesheet">

        <!-- Optional JavaScript -->
        <!-- jQuery first, then Popper.js, then Bootstrap JS -->
        <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
        <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
                integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
                crossorigin="anonymous"></script>
        <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
                integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
                crossorigin="anonymous"></script>
        <link rel="preconnect" href="https://fonts.gstatic.com">
        <link href="https://fonts.googleapis.com/css2?family=Jua&display=swap" rel="stylesheet">
        <style>
            .wrap {
                width: 90%;
                max-width: 750px;
                margin: 0 auto;
            }

            .banner {
                width: 100%;
                height: 20vh;
                background-image: url("{{ url_for("static", filename="banner.jpg") }}");
                background-position: center;
                background-size: contain;
                background-repeat: repeat;

            }

            h1.title {
                font-family: 'Jua', sans-serif;

                color: white;
                font-size: 3rem;
            }

            h5 {
                font-family: 'Jua', sans-serif;
            }

            .matjip-list {
                overflow: scroll;
                width: 100%;
                height: calc(20vh - 30px);
                position: relative;
            }

            .card-title, .card-subtitle {
                display: inline;
            }
```

```
            #map {
                width: 100%;
                height: 50vh;
                margin: 20px auto 20px auto;
            }

            .btn-sparta {
                color: #fff;
                background-color: #e8344e;
                border-color: #e8344e;
            }

            .iw-inner {
                padding: 10px;
                font-size: smaller;
            }

            i {
                color: #e8344e;
            }

            i:hover {
                cursor: pointer;
            }
        </style>
        <script>
            let y_cen = 37.4981125;   // lat
            let x_cen = 127.0379399;  // long
            let map;
            let markers = [];
            let infowindows = [];
            $(document).ready(function () {
                map = new naver.maps.Map('map', {
                    center: new naver.maps.LatLng(y_cen, x_cen),
                    zoom: 12,
                    zoomControl: true,
                    zoomControlOptions: {
                        style: naver.maps.ZoomControlStyle.SMALL,
                        position: naver.maps.Position.TOP_RIGHT
                    }
                });

                get_matjips()
            });

            function get_matjips() {
                $('#matjip-box').empty();
                for (let i = 0; i < markers.length; i++) {
                    markers[i].setMap(null);
                    infowindows[i].close()
                }
                markers = [];
                infowindows = [];

                $.ajax({
                    type: "GET",
                    url: `/matjip`,
                    data: {},
                    success: function (response) {
                        let matjips = response["matjip_list"];
                        console.log(matjips.length);
                        for (let i = 0; i < matjips.length; i++) {
                            let matjip = matjips[i];
                            make_card(i, matjip);
                            let marker = make_marker(matjip);
                            add_info(i, marker, matjip)
                        }
                    }
                });
            }

            function make_marker(matjip) {
                let marker_img = '';
                if ("liked" in matjip) {
                    marker_img = '{{ url_for("static", filename="marker-liked.png") }}'
                } else {
                    marker_img = '{{ url_for("static", filename="marker-default.png") }}'
                }
                let marker = new naver.maps.Marker({
                    position: new naver.maps.LatLng(matjip["mapy"], matjip["mapx"]),
                    map: map,
                    icon: marker_img
                });
                markers.push(marker);
                return marker
            }
```

```
            function make_card(i, matjip) {
                let html_temp = ``;
                if ("liked" in matjip) {
                    html_temp = `<div class="card" id="card-${i}">
                                    <div class="card-body">
                                        <h5 class="card-title"><a href="javascript:click2center(${i})" class="matjip-title">$
                                        <h6 class="card-subtitle mb-2 text-muted">${matjip['category']}</h6>
                                        <i class="fa fa-bookmark" onclick="bookmark('${matjip['title']}', '${matjip['address'
                                        <p class="card-text">${matjip['address']}</p>
                                        <p class="card-text" style="color:blue;">${matjip['show']}</p>
                                    </div>
                                </div>`;
                } else {
                    html_temp = `<div class="card" id="card-${i}">
                                    <div class="card-body">
                                        <h5 class="card-title"><a href="javascript:click2center(${i})" class="matjip-title">$
                                        <h6 class="card-subtitle mb-2 text-muted">${matjip['category']}</h6>
                                        <i class="fa fa-bookmark-o" onclick="bookmark('${matjip['title']}', '${matjip['addres
                                        <p class="card-text">${matjip['address']}</p>
                                        <p class="card-text" style="color:blue;">${matjip['show']}</p>
                                    </div>
                                </div>`;
                }


                $('#matjip-box').append(html_temp);
            }

            function add_info(i, marker, matjip) {
                let html_temp = `<div class="iw-inner">
                                    <h5>${matjip['title']}</h5>
                                    <p>${matjip['address']}
                                    </div>`;
                let infowindow = new naver.maps.InfoWindow({
                    content: html_temp,
                    maxWidth: 200,
                    backgroundColor: "#fff",
                    borderColor: "#888",
                    borderWidth: 2,
                    anchorSize: new naver.maps.Size(15, 15),
                    anchorSkew: true,
                    anchorColor: "#fff",
                    pixelOffset: new naver.maps.Point(10, -10)
                });
                infowindows.push(infowindow);
                naver.maps.Event.addListener(marker, "click", function (e) {
                    console.log("clicked", infowindows.length);
                    if (infowindow.getMap()) {
                        infowindow.close();
                    } else {
                        infowindow.open(map, marker);
                        map.setCenter(infowindow.position);
                        $("#matjip-box").animate({
                            scrollTop: $("#matjip-box").get(0).scrollTop + $(`#card-${i}`).position().top
                        }, 2000);
                    }
                });
            }

            function click2center(i) {
                let marker = markers[i];
                let infowindow = infowindows[i];
                if (infowindow.getMap()) {
                    infowindow.close();
                } else {
                    infowindow.open(map, marker);
                    map.setCenter(infowindow.position)
                }
            }

            function bookmark(title, address, action) {
                $.ajax({
                    type: "POST",
                    url: "/like_matjip",
                    data: {
                        title_give: title,
                        address_give: address,
                        action_give: action
                    },
                    success: function (response) {
                        if (response["result"] == "success") {
                            get_matjips()
                        }
                    }
                })
            }
        </script>
```

```
    </head>

    <body>
        <div class="wrap">
            <div class="banner">
                <div class="d-flex flex-column align-items-center"
                     style="background-color: rgba(0,0,0,0.5);width: 100%;height: 100%;">
                    <h1 class="title mt-5 mb-2">스파르타 맛집 지도</h1>
                    <button type="button" onclick="get_matjips()" class="btn btn-sparta">
                        새로고침하고 더 많은 맛집 보기
                    </button>
                </div>
            </div>
            <div id="map"></div>

            <div class="matjip-list" id="matjip-box">
            </div>
        </div>

    </body>

</html>
```