



날씨에 따른 음악 추천 PLAYLIST

샘플리

이준 채석
노빈 유훈



목 차

01 . 프로젝트 설명

02 . 플로우 차트

03 . 세부 코드

04 . 트러블 슈팅

05 . 시연 영상

06 . 팀원 소개 및 소감



01. 프로젝트 개요

프로젝트 목표 및 설명

하루를 시작할 때, 때로는 기분에 맞는 음악 한 곡이 그날의 분위기를 완전히 바꿔줍니다.
하지만 어떤 곡이 좋을지, 매번 고민하게 됩니다. "오늘의 날씨가 말해주는 분위기로 음악을 고르면 어떨까?"
그래서 만들었습니다!



"오늘의 날씨 분위기에 따른 노래 추천!!"

사용자의 위치는 특별시,광역시 등 전국의 주요 도시 40여개중
선택

현재 날씨 정보를 OpenWeatherMap 에서 받아와 출력



해당 날씨에 해당하는 키워드를 멜론 DJ에 해시태그로 검색하여
검색된 플레이리스트 중 노래 추천



01. 프로젝트 개요

구체적인 기능



위치 선택

- 사용자가 원하는 위치 선택



날씨 제공 및 노래 추천

- 선택한 위치와 시간대에 따라 실시간 날씨 정보 제공
- 날씨 분위기에 어울리는 노래 추천

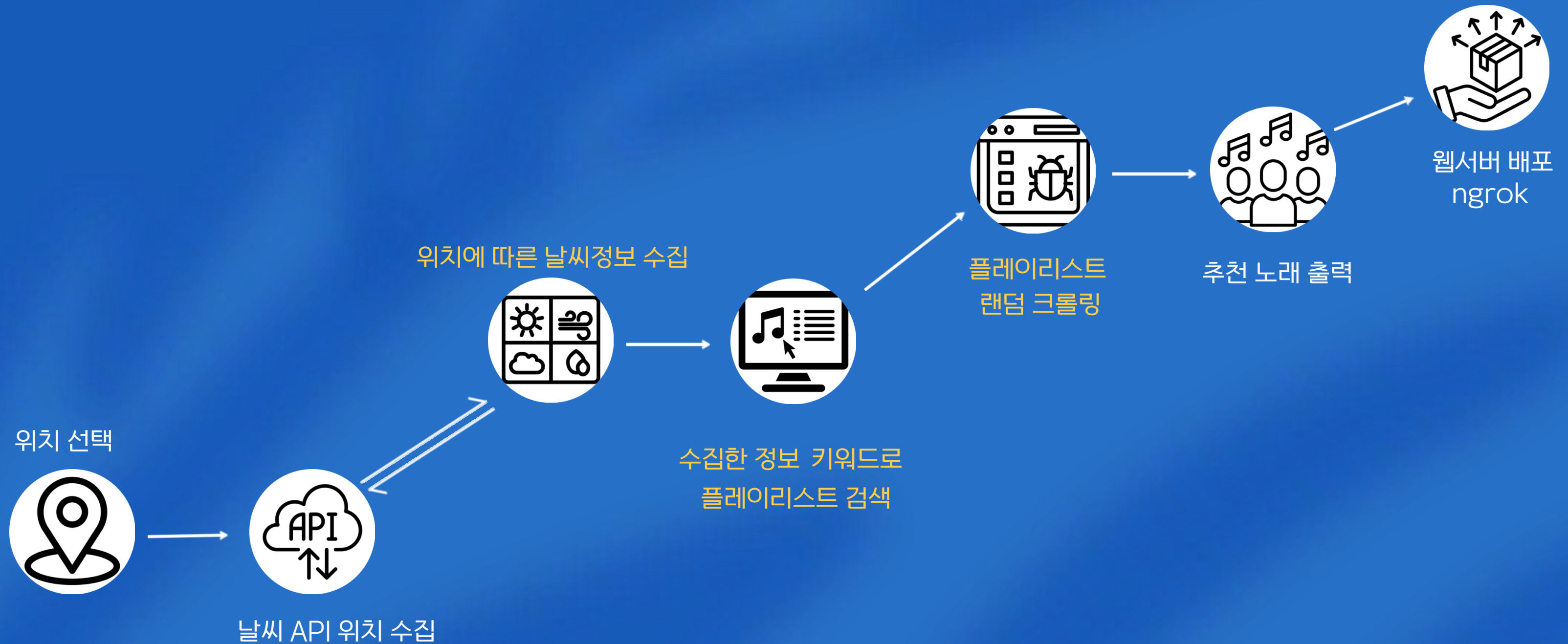


유튜브 검색

- 추천된 노래 제목을 클릭하면 해당 곡의 유튜브 검색 결과 화면으로 자동연결



02. FLOW CHART



03. 코드 구현

위치 선택 페이지 (home.html)

```
@app.route("/")
def home():
    return render_template("home.html")
```

-> app.py와 home.html 연결

```
<div class="container">
<h1>오늘의 날씨 ☀️ 오늘의 노래 🎵 </h1>
<span> 현재 날씨에 어울리는 노래를 추천해드려요! </span>
<h3> 📍 현재 위치를 선택해주세요</h3>
<form action="/index" method="GET">
<label for="province">시/도 선택:</label>
<select id="province" onchange="updateCities()">
  <option value="">-- 시/도 선택 --</option>
  <option value="서울특별시">서울특별시</option>
  <option value="경기도">경기도</option>
  <option value="강원특별자치도">강원특별자치도</option>
```

-> form 태그를 통해 index.html와 연결

-> 1. select 태그를 통한 지역 선택

```
<br>
<button class="btn" type="submit"> 지금 날씨와 어울리는 노래는</button>
</form>
```

-> submit 태그를 통해 도시 이름이 쿼리 스트링 형식으로 URL에 붙어 전송
Ex) http://127.0.0.1:5000//index?city=서울특별시

```
<br>
<label for="city">시/군/구 선택:</label>
<select id="city" name="city">
  <option value="">-- 먼저 시/도를 선택하세요 --</option>
</select>

<script>
  const regionData = {
    "서울특별시": ["서울특별시"],
    "부산광역시": ["부산광역시"],
```

-> 2. select 태그를 통해 도시 선택



03. 코드 구현

위치 선택 페이지 (home.html)

오늘의 날씨 🌤️ 오늘의 노래 🎵

현재 날씨에 어울리는 노래를 추천해드려요!

📍 현재 위치를 선택해주세요

시/도 선택: 경기도 ▼

시/군/구 선택: -- 시/도 선택 --

지금 날씨와 어울리는 노래는

서울특별시

경기도

강원특별자치도

충청북도

충청남도

경상북도

경상남도

전북특별자치도

부산광역시

대구광역시

대전광역시

인천광역시

울산광역시

제주특별자치도

select 1 : 시/도 선택



오늘의 날씨 🌤️ 오늘의 노래 🎵

현재 날씨에 어울리는 노래를 추천해드려요!

📍 현재 위치를 선택해주세요

시/도 선택: 경기도 ▼

시/군/구 선택: 고양시 ▼

지금 날씨와 어울리는 노래는

부천시

용인시

수원시

화성시

하남시

남양주시

오산시

파주시

평택시

안양시

군포시

광명시

일산시

select 2 : 시/군/구 선택



```
<button class="btn" type="submit">  
지금 날씨와 어울리는 노래는  
</button>
```

지금 날씨와 어울리는 노래는



03. 코드 구현

@app.route("/index")

```
@app.route("/index")
def index():
    city = request.args.get("city")
    city_id = city_name_to_id.get(city)
    weather, temp = get_weather_by_city_id(city_id)
    keywords = weather_to_keywords(weather)
    songs = crawl_melon_dj(keywords)

    return render_template("index.html",
                           city=city,
                           weather=weather,
                           temp=temp,
                           keywords=keywords,
                           songs=songs)

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

- > city = http://127.0.0.1:5000//index?city=서울특별시
city = "서울특별시"



03. 코드 구현

OpenWeather API 실시간 날씨 정보 수집 (app.py)

```
city_name_to_id = {  
    "서울특별시": 1835847, "부산광역시": 1838519, "대구광역시": 1835327, "대전광역시": 1835224,  
    "광주광역시": 1841808, "인천광역시": 1843561, "울산광역시": 1833742, "고양시": 1842485,  
    "부천시": 1838716, "용인시": 1832427, "수원시": 1835553, "화성시": 1843847,  
    "하남시": 1897007, "남양주시": 1897122, "오산시": 1839652, "파주시": 1840898, "군포시" : 1842030,  
    "평택시": 1838343, "보령시": 1835447, "부여군": 1838508, "공주시": 1842616,  
    "논산시": 1840211, "예산군": 1832771, "천안시": 1845759, "청주시": 1845604,  
    "충주시": 1845033, "강릉시": 1843137, "인제군": 1843542, "원주시": 1833105,  
    "속초시": 1836553, "화천군": 1844045, "양양군": 1832809, "춘천시": 1845136,  
    "목포시": 1841066, "무안군": 1840982, "무주군": 1840942, "화순군": 1843841,  
    "신안군": 6395804, "익산시": 1843491, "진안군": 1846114, "임실군": 1843585,  
    "완주군": 1833466, "창원시": 1846326, "김해시": 1842943, "남해군": 1840454,  
    "밀양시": 1841149, "양산시": 1832828, "창녕군": 6903078, "함양군": 1844533,  
    "김천시": 1842944, "상주시": 1837706, "포항시": 1839071, "울진군": 1833105,  
    "구미시": 1842225, "제주특별자치도": 1846265  
}
```

50여개의 주요도시의 이름에 해당하는 도시 ID 딕셔너리 생성 (Openweatherhmap 사이트 참고)

EX) city = "서울특별시" > city_id = city_name_to_id.get(city) > city_id = 1835847



03. 코드 구현

OpenWeather API 실시간 날씨 정보 수집 (app.py)

```
def get_weather_by_city_id(city_id):  
    url = f"https://api.openweathermap.org/data/2.5/weather?id={city_id}&appid={API_KEY}&units=metric"  
    res = requests.get(url).json()  
    if 'weather' not in res or 'main' not in res :  
        return "정보 없음", "정보 없음", "알 수 없음"  
    weather = res['weather'][0]["main"]  
    temp = round(res['main']['temp'],1)  
    return weather, temp
```

- &units=metric 을 통해 온도의 단위를 켈빈 (K) 에서 섭씨(°C)로 변경
- 온도 temp 는 소수 둘째 자리에서 반올림 (11.05 > 11.1)

Ex) weather = "Rain"
temp = 11.1

JSON

JSON format API response example

```
{  
  "coord": {  
    "lon": 7.367,  
    "lat": 45.133  
  },  
  "weather": [  
    {  
      "id": 501,  
      "main": "Rain",  
      "description": "moderate rain",  
      "icon": "10d"  
    }  
  ],  
  "base": "stations",  
  "main": {  
    "temp": 284.2,  
    "feels_like": 282.93,  
    "temp_min": 283.06,  
    "temp_max": 286.82,  
    "pressure": 1021,  
    "humidity": 60,  
    "sea_level": 1021,  
    "grnd_level": 910  
  },  
}
```



03. 코드 구현 날씨별 키워드 변환(app.py)

```
def weather_to_keywords(weather):  
    mapping = {  
        "Clear": "맑음",  
        "Clouds": "흐림",  
        "Rain": "비오는날",  
        "Snow": "눈오는날",  
        "Thunderstorm": "천둥번개",  
        "Drizzle": "이슬비",  
        "Fog": "안개"  
    }  
    return mapping.get(weather)
```

->날씨별 키워드 설정

멜론에서 곡 정보를 수집할 때 해시태그 검색용으로 활용

Ex) weather = "Rain" > return "비오는날"



03. 코드 구현

@app.route("/index")

```
@app.route("/index")
def index():
    ✓city = request.args.get("city")
    ✓city_id = city_name_to_id.get(city)
    ✓weather, temp = get_weather_by_city_id(city_id)
    ✓keywords = weather_to_keywords(weather)
    songs = crawl_melon_dj(keywords)

    return render_template("index.html",
                           city=city,
                           weather=weather,
                           temp=temp,
                           keywords=keywords,
                           songs=songs)

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

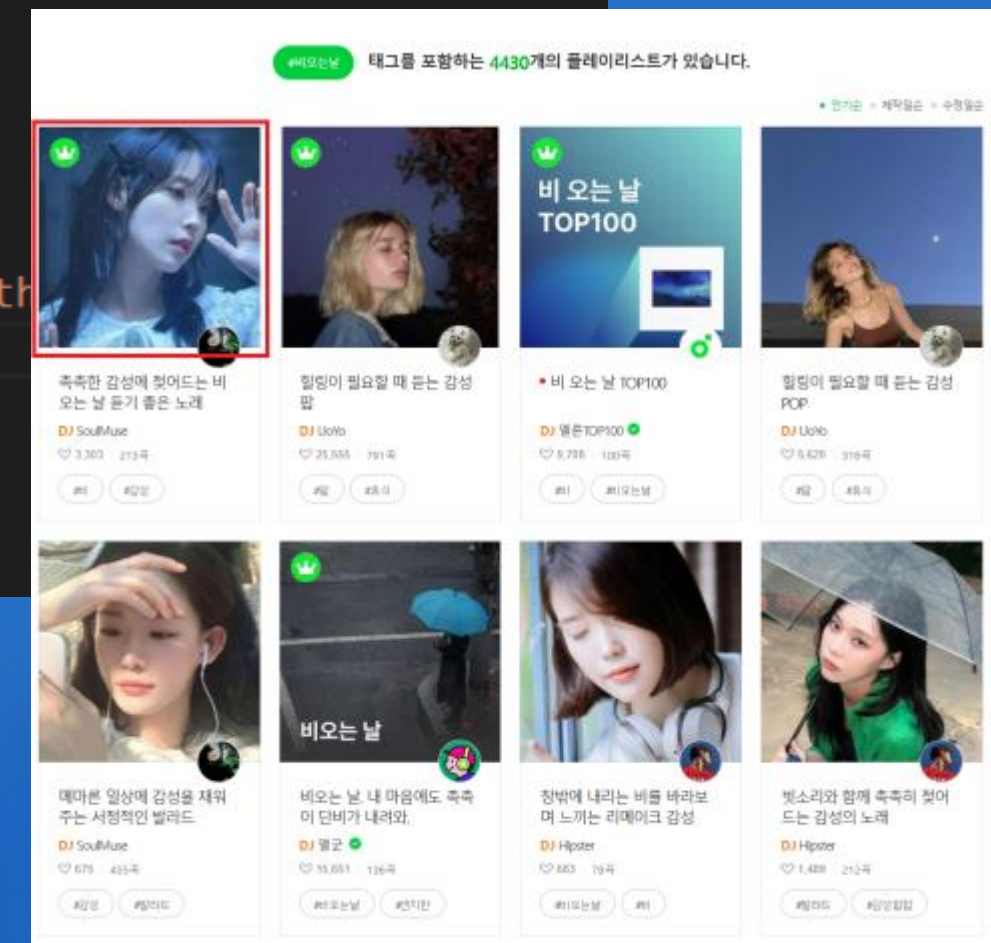


03. 코드 구현

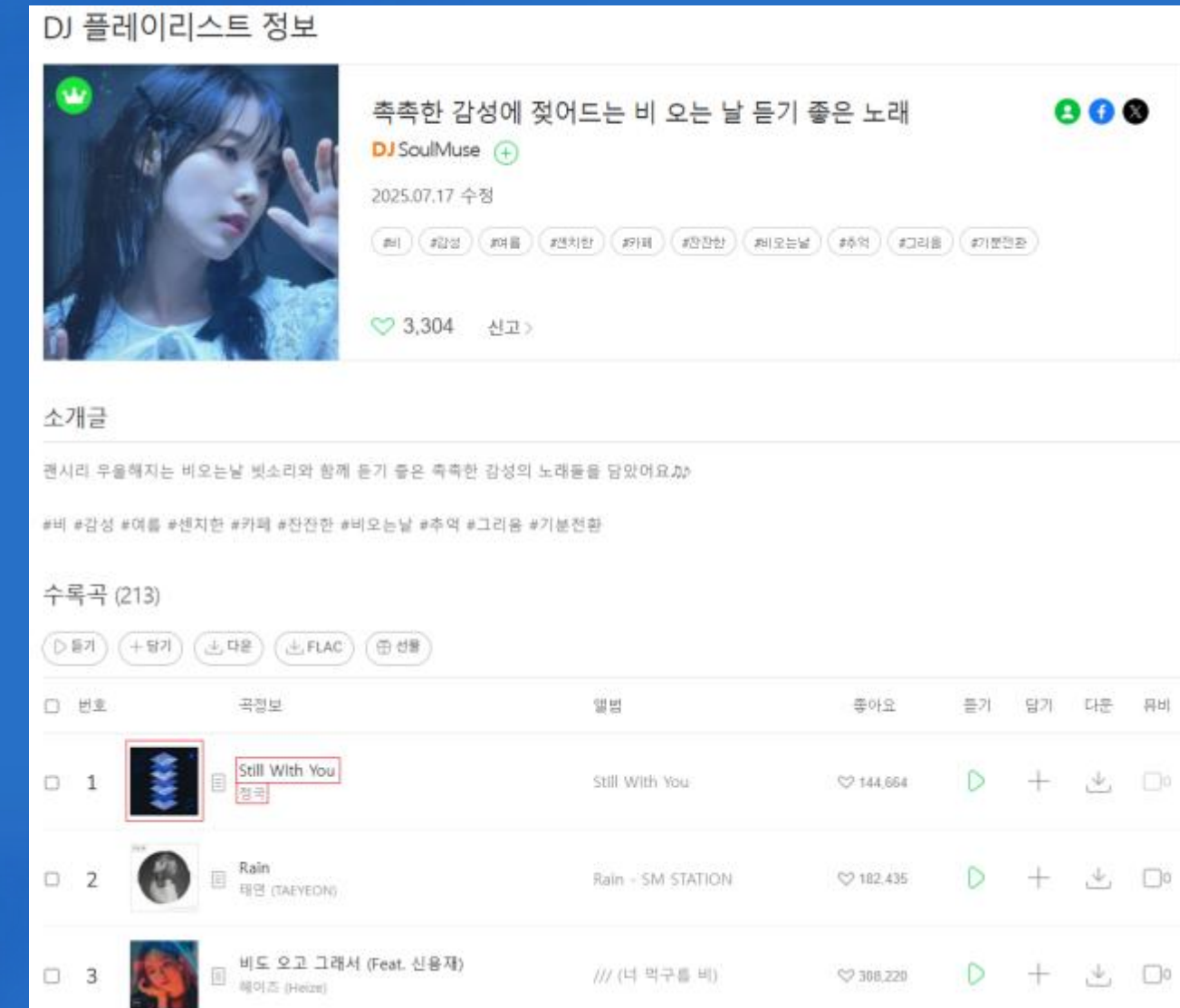
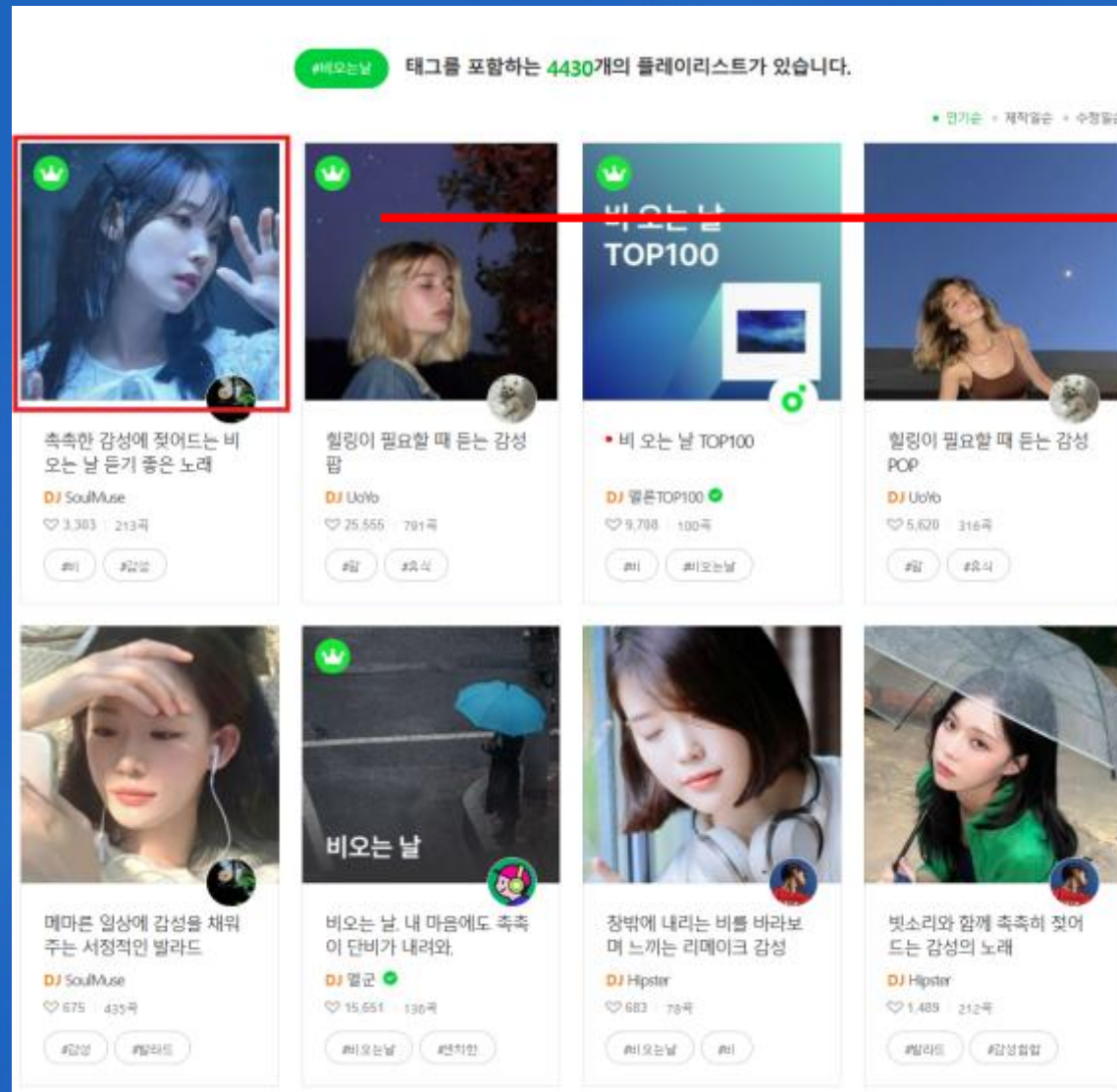
곡정보 수집 (app.py)

```
def crawl_melon_dj(keyword):  
    driver = create_driver()  
    results = []  
    try:  
        driver.get(f"https://www.melon.com/dj/djfinder/djfinder_inform.htm?djSearchType=T&djSearchKeyword=%23비오는날")  
        time.sleep(0.2)  
  
        playlist_items = driver.find_elements(By.CSS_SELECTOR, "#djPlylstList > div > ul > li")  
        playlist_count = len(playlist_items)  
        playlist_random = random.randint(1, playlist_count)  
        playlist_selector = f"#djPlylstList > div > ul > li:nth-child({playlist_random}) > div.th  
        playlist = driver.find_element(By.CSS_SELECTOR, playlist_selector) # 플레이리스트 선택  
        playlist.click()  
        time.sleep(0.3)
```

날씨에 맞는 키워드를 멜론 DJ 해시태그에 검색 -> 플레이리스트를 랜덤으로 선택



03. 코드 구현 곡정보 수집 (app.py)



랜덤 플레이리스트 선택 후 1번곡 ~ 20번곡까지 랜덤 3개를 골라 results 에 제목,가수,이미지를 추가



03. 코드 구현

곡정보 수집 (app.py)

```
for i in random.sample(range(1,20),3):
    title_selector = f"#frm > div > table > tbody > tr:nth-child({i}) > td:nth-child(5) > div > div > div.ellipsis.rank01 > span > a"
    artist_selector = f"#frm > div > table > tbody > tr:nth-child({i}) > td:nth-child(5) > div > div > div.ellipsis.rank02 > a"
    album_img_selector = f"#frm > div > table > tbody > tr:nth-child({i}) > td:nth-child(3) > div > a > img"

    title = driver.find_element(By.CSS_SELECTOR, title_selector).text.strip()
    artist = driver.find_element(By.CSS_SELECTOR, artist_selector).text.strip()

    album_img_element = driver.find_element(By.CSS_SELECTOR, album_img_selector)
    album_img_url = album_img_element.get_attribute("src")

    results.append({
        "title": title,
        "artist": artist,
        "album_img": album_img_url
    })

except Exception as e:
    print("크롤링 오류:", e)
finally:
    driver.quit()

return results
```



03. 코드 구현

최종 저장(app.py)

```
@app.route("/index")
def index():
    city = request.args.get("city")
    city_id = city_name_to_id.get(city)
    weather, temp = get_weather_by_city_id(city_id)
    keywords = weather_to_keywords(weather)
    songs = crawl_melon_dj(keywords)

    return render_template("index.html",
                           city=city,
                           weather=weather,
                           temp=temp,
                           keywords=keywords,
                           songs=songs)

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

city = http://127.0.0.1:5000//index?city=서울특별시

weather = "Rain"

temp = 11.1

songs =

[

 {"title": "Still With You", "artist": "정국", "album_img": "..."},

 {"title": "Butter", "artist": "방탄소년단", "album_img": "..."},

 {"title": "???", "artist": "???", "album_img": "..."}]

]



03. 코드 구현 날씨 정보 및 플레이리스트 출력 (index.html)

```
<div class="container">
  <h2> {{ city }}</h2>

  <div class="weather-info">
    <p>
      <strong>
        {% if weather.lower() == 'clear' %}
        맑음
        {% elif weather.lower() == 'clouds' %}
        흐림
        {% elif weather.lower() == 'rain' %}
        비
        {% elif weather.lower() == 'snow' %}
        눈
        {% elif weather.lower() == 'thunderstorm' %}
        천둥번개
        {% elif weather.lower() == 'drizzle' %}
        이슬비
        {% elif weather.lower() == 'atmosphere' %}
        안개
        {% else %}
        정보 없음
        {% endif %}
      </strong>
    </p>
    <p><strong>{{ temp }}</strong></p>
  </div>
```

-> 날씨 상태에 따른 아이콘 출력

-> 날씨 상태 및 온도 텍스트 출력

-> 크롤링 된 노래 출력

```
<h2> ♫ 오늘 날씨에 어울리는 음악</h2>
<ul>
  {% for song in songs %}
  <li>
    <table border="0" cellspacing="10">
      <tr>
        <td></td>
        <td style="text-align: left;">
          <a href="https://www.youtube.com/results?search_query={{ (song.title + ' ' + song.artist) | url_encode }}"
            target="_blank" style="margin-left: 4px; display: inline-block;">
            ▶ {{ song.title }}
          </a><br/>
          <span class="artist-name">{{ song.artist }}</span>
        </td>
      </tr>
    </table>
  </li>
  </ul>
  <button class="btn" onclick="location.reload()">다른 노래 추천받으세요</button>
```



04. 트러블 슈팅 발생 문제



HTTP 위치 정보 제한

보안상의 이유로 HTTP 환경에서의
위치접근 차단



로딩 지연

앨범 정보 크롤링 시 일부 요소가 제대로 출력 안됨

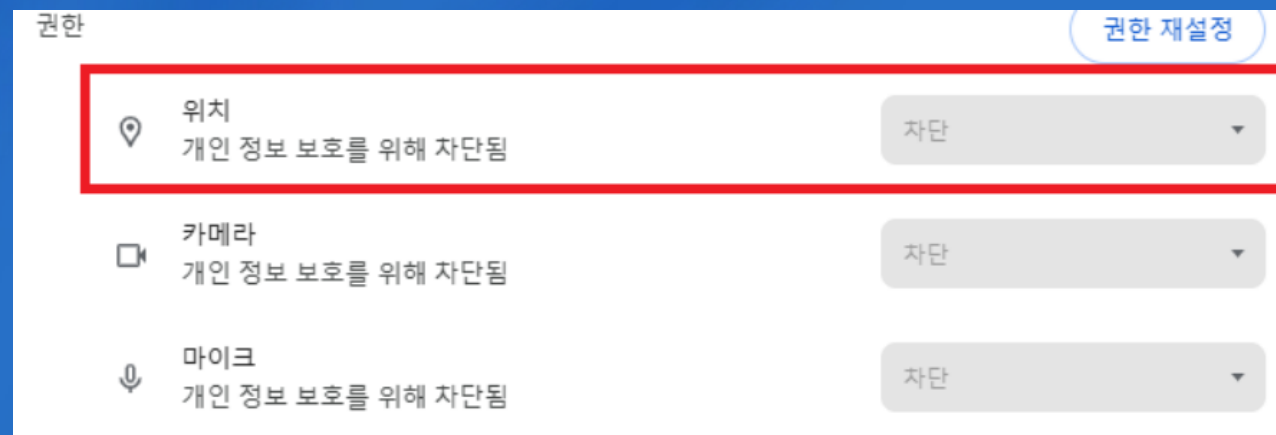


04. 트러블 슈팅 문제 해결 - HTTP 보안 제한



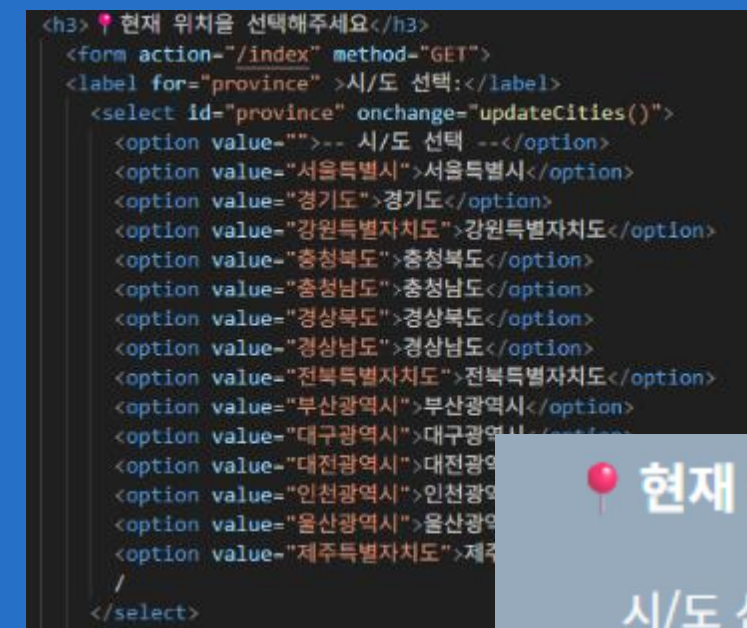
HTTP 위치 정보 제한

외부 디바이스로 접속시 HTTP 보안 정책에 따른
위치 정보 차단 상황 발생



위치 직접 선택

사용자가 직접 현재 위치 지정



현재 위치를 선택해주세요

시/도 선택: 제주특별자치도 ▼

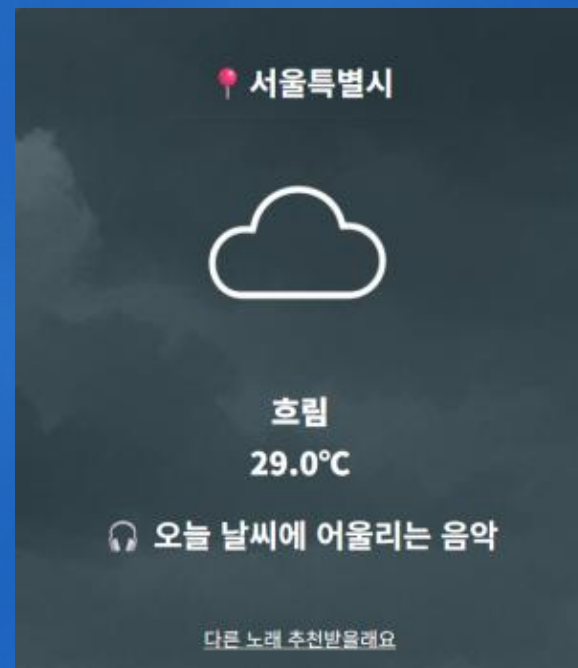
시/군/구 선택: 제주특별자치도 ▼



04. 트러블 슈팅 문제 해결 - 음원정보 로딩 지연

💡 로딩 지연

Selenium이 페이지 랜더링 보다 먼저 크롤링 시도
Melon에서 받아오는 음악정보 누락 발생



- > 노래 정보가 뜨지 않음

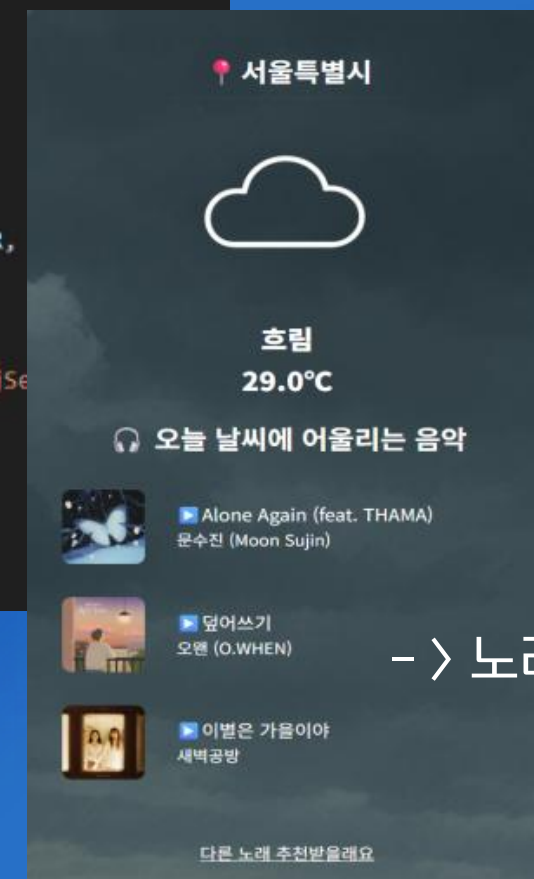
💡 DELAY 값 추가

time.sleep 을 통해 Selenium의 요소 로딩 시간 확보

```
# 멜론 DJ 플레이리스트 크롤링
def crawl_melon_dj(keyword):
    driver = create_driver()
    results = []
    try:
        driver.get("https://www.melon.com")
        time.sleep(0.2)

        dj_menu = driver.find_element(By.CSS_SELECTOR, 'div.dj-menu')
        dj_menu.click()
        time.sleep(0.2)

        search_input = driver.find_element(By.ID, 'djSearchInput')
        search_input.clear()
        for ch in keyword:
            search_input.send_keys(ch)
            time.sleep(0.1)
            search_input.send_keys(Keys.RETURN)
            time.sleep(0.2)
```



- > 노래 정보 정상 출력



05. 시연 영상



날씨 기반 음악 추천
핸드폰 카메라로 찍어보세요!



06. 팀원 역할 및 소감



이 ○ 준

flask 서버 개발/데이터 처리 / 발표

이번 프로젝트를 통해 실시간으로 데이터를 수집하고 크롤링하는 과정을 직접 해볼 수 있어 재미있었습니다. 팀원들이 각자 맡은 역할을 잘 수행해 주고 많이 도와줘서 고마웠고, 서로에 대해 더 잘 알 수 있어서 행복했습니다.



노 ○ 빈

웹페이지 구성(html) 제작/ 데이터 처리/ PPT 제작

Python을 활용해 프로그램을 제작하는 것은 처음이라 매우 흥미로웠습니다. 특히 웹 크롤링을 통해 정보를 수집하고 데이터를 처리하는 과정에서 반도체 설계과정에 응용할 수 있을 것이라는 기대감이 생겼습니다. Python를 시작으로 다양한 프로그래밍 언어를 습득할 수 있는 기반을 마련하였다고 생각합니다.



채 ○ 석

웹페이지(html) 제작/데이터 처리 / 발표

수업시간 때 배운 날씨 데이터와 음악 추천 기능을 연동해보며 웹 자동화의 기초를 직접 경험할 수 있었습니다. 실시간 데이터를 기반으로 콘텐츠를 자동으로 보여주는 방식이 흥미로웠고, 실제 서비스 구현 과정에 대해 이해할 수 있는 좋은 시간이었습니다.



유 ○ 훈

크롤링 구현 / 데이터 처리 /PPT 제작

이번 프로젝트를 통해 날씨 API와 멜론 크롤링을 결합하여, 날씨에 어울리는 음악을 추천하는 웹서비스를 구현할 수 있었습니다. 외부 데이터를 실시간으로 연동하고, 크롤링을 통해 정보를 수집하는 과정에서 웹 개발과 데이터 처리의 흐름을 경험할 수 있어 뜻 깊은 시간이었습니다.



+ 팀원들의 추천곡



이 ○ 준

그날 본 꽃의 이름을 우리는 모른다 OST -
Secret Base



채 ○ 석

the score - Unstoppable



노 ○ 빈

최유리 - 사랑



유 ○ 훈

터치드 - Hi Bully

