

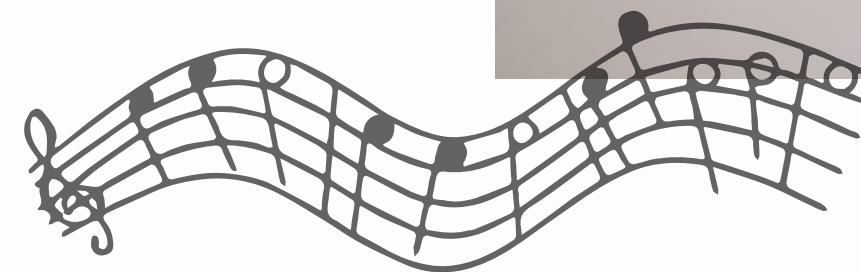

Music Recommend

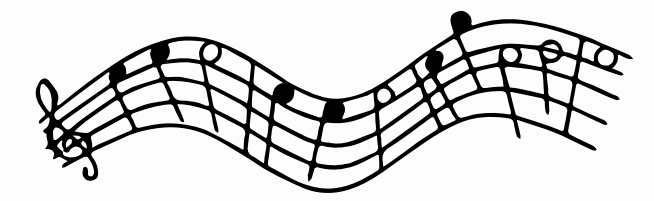
[HTTPS://GITHUB.COM/YOON1717/YOONCALENDAR](https://github.com/YOON1717/YOONCALENDAR)

윤 가 영

Contents

- 01. 프로젝트 주제
- 02. 주요 기능
- 03. 프로젝트 핵심 소스
- 04. 향후 개선점

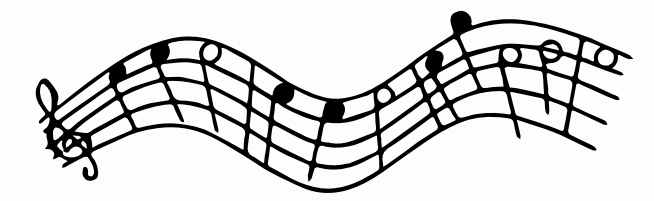




사용자의 입력에 따른 음악 추천 기능을 위한

멜론 사이트 크롤링으로 데이터를 수집,
NPL 자연어 처리와 임베딩을 통해
유사도가 높은 음악을 추천





HOME

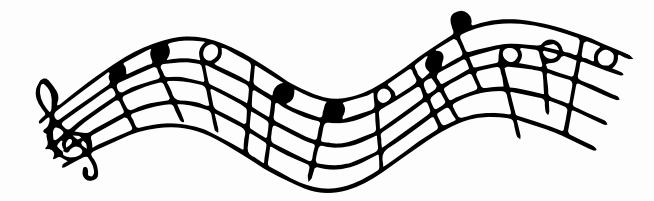
어떤 노래를 듣고싶나요

홍대에서 공연 봤을 때 정말이지 감동적이고 특별한 기분이었다.
뭔가 짜릿하고 신나는 기

추천

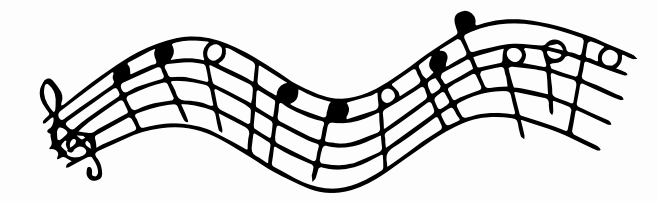
Copyright © Your Website 2023

사용자에게
듣고 싶은 노래에 대한 설명
혹은 어떤 기분인지
텍스트를 입력 받습니다.



라면인건가	▽
폰서트	▽
instagram	▽
값	▽
한여름밤의 꿀	▽
자니 (Feat. Dynamic Duo)	▽
쿠키 영상	▽
사이렌 Remix (Feat. UNEDUCATED KID, Paul Blanco)	▽
아주 NICE	▽
거리에서 (Feat. ASH ISLAND)	▽

사용자가 입력한 텍스트와
가사의 유사도가
가장 높은 10개의 곡을 추천합니다.



폰서트

Melon

이건 세상에서 제일 비싼 단독 공연 가수는 나고 관객은 너 하나 화려한 막이 이제 곧 올라가기 전에 그저 몇 가지만 주의해줘요 세상에서 제일 편한 옷을 갈아 입고 제일 좋아하는 자리에 누워 배터리가 바닥나지 않게 조심하고 통화상태를 항상 유지해줘요 듣고 싶은 노래를 말 만해 everything 입이 심심할 때는 coffee popcorn anything 너무 부담주진 말고 편하게 들어줘 아님 내가 너무 오직 너를 웃게 하기 위한 코너 네가 너무 설레 잠 못 들게 만들 거야 : 별한 너의 취향을 알아 달콤한데 슬픈 듯 아찔하게 맞지 근데 다음 곡 그 날 그 노래 내가 너무 진지해 보여도 웃지마 누가 봐도 완벽한 노래 문에 틀리잖아 아직 나는 너무 떨리니까 오직 너에게만 감동적인 노래 무 설레 잠 못 들게 만들 거야 지금이야 크게 소리 질러 이 공연은 거의 문자로 너무나 아쉽지만 졸린 거 이미 알고 있어 기대해줘 마지막 곡 너를 웃게 하기 위한 코너 네가 너무 설레 잠 못 들게 만들 거야 지금이야 시 듣고 싶어 사실 내가 원해 네가 너무 설레 잠 못 들지 모르지만 앵콜 제일 비싼 단독공연 가수는 나고 관객은 너 하나

Melón

'미쳐버리겠다' 말이 나오는 비오의 추천곡!

멜론차트

최신음악

장르음악

멜론DJ

멜론TV

스타포스트

매거

곡 정보

폰서트

10CM

앨범

4.0

발매일

2017.09.01

장르

인디음악, 포크/블루스

FLAC

Flac 16/24bit

♡ 226,242

↓ 곡 다운 >

가사

이건 세상에서

제일 비싼 단독 공연

가수는 나고 관객은 너 하나

화려한 막이

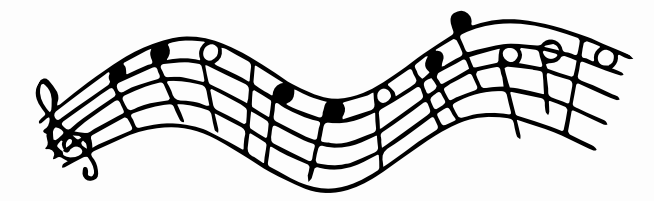
이제 곧 올라가기 전에

그저 몇 가지만 주의해줘요

세상에서 제일 편한

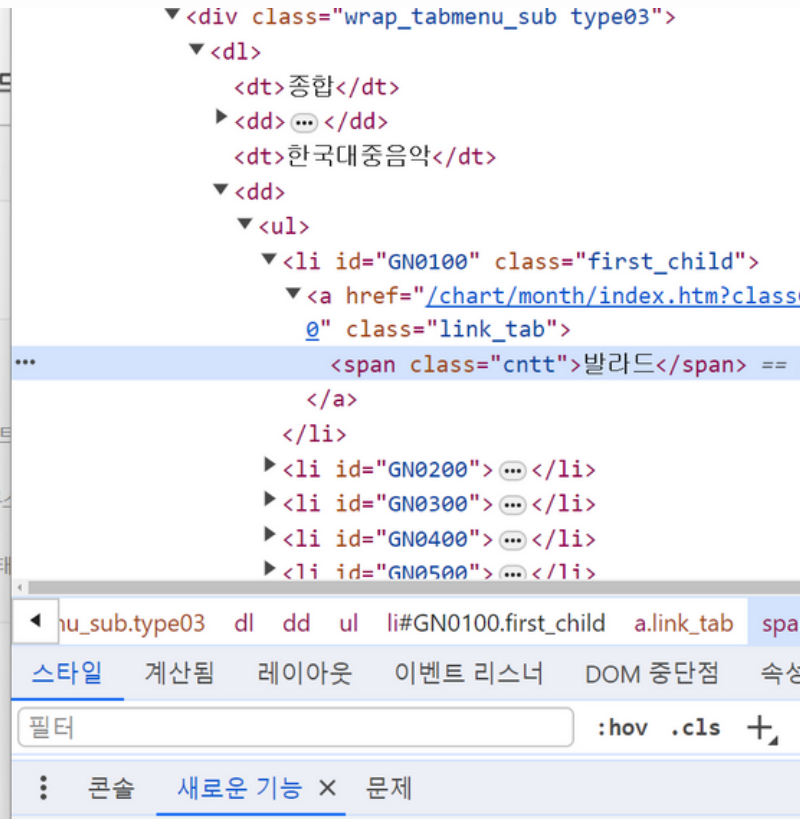
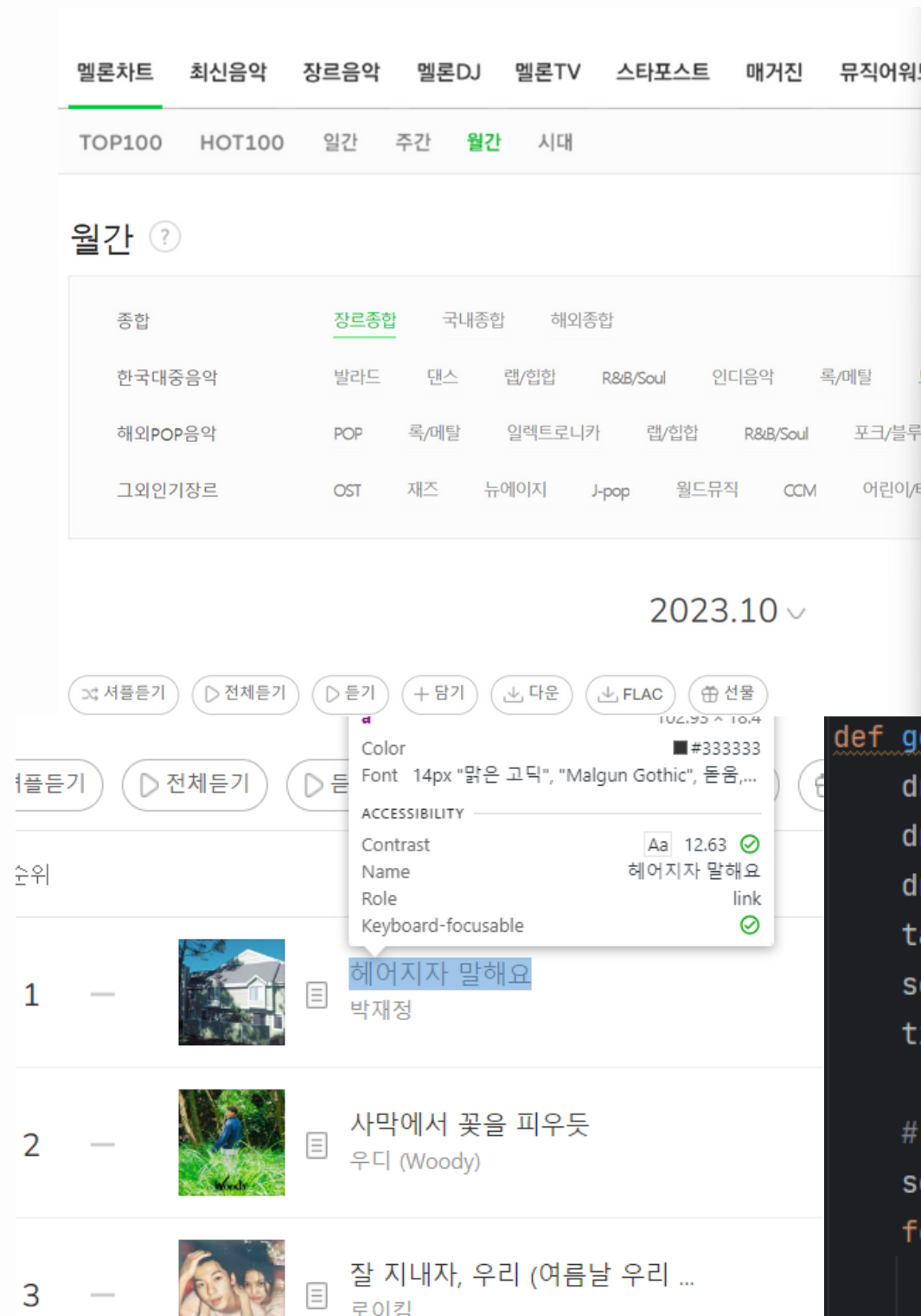
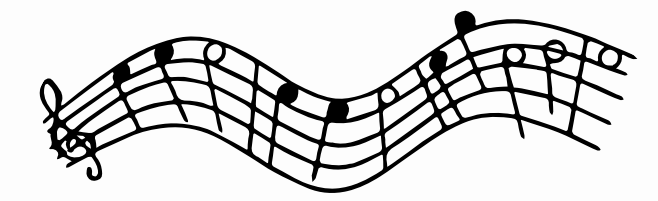
추천 리스트의 노래를 누르면 해당 곡의 가사 정보와 melon 의 노래 정보 페이지로 연결되는 버튼이 있습니다.

YOON
GAYOUNG



음악 추천 아코디언 옆에는
사용자의 입력어와
가사의 유사도가 높은 50개 곡의
장르 정보를 방사형 그래프로
시각화 해 보여줍니다.

03. 프로젝트 핵심 소스 / 웹 크롤링



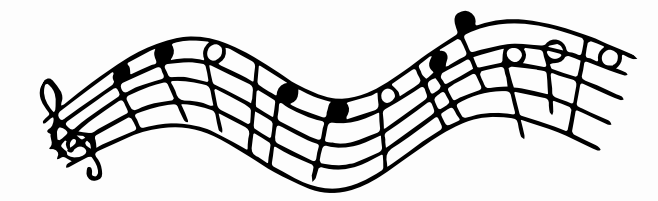
'GN0100' : 발라드
'GN0200' : 댄스
'GN0300' : 랩/힙합
'GN0400' : R&B/SOUL
'GN0500' : 인디음악
'GN0600' : 록/메탈
'GN0700' : 트로트
'GN0800' : 포크/블루스

멜론 차트 페이지의
html 요소 속성 분석

selenium의 webdriver를 활용
8개 장르에 대한
100개 곡 고유번호 수집

pandas를 활용
장르 | 곡 번호
xlsx 파일로 저장

```
def get_genre(query):  
    driver = webdriver.Chrome("../chromedriver.exe")  
    driver.implicitly_wait(3)  
    driver.get(url + query)  
    table = driver.find_element(By.TAG_NAME, 'table')  
    soup = BeautifulSoup(driver.page_source, 'html.parser')  
    time.sleep(1)  
  
    # 데이터 수집 및 리스트에 추가  
    song_elements = soup.find_all('tr', attrs={'data-song-no': True})  
    for song in song_elements:  
        song_no = song['data-song-no']  
        print(f'data-song-no: {song_no}')  
        song_list.append({'keyword': query, 'song-no': song_no})
```

곡 정보



Love Lee

AKMU (악뮤)

앨범 Love Lee
발매일 2023.08.21
장르 댄스
FLAC Flac 16/24bit

♡ 109,013 ⬇️ 곡 다운 >

div#d_video_summary.lyric 1008 × 170

You know
내 스타일이 아닌 음악을 들어도
You know

```
# 가사 정보를 가져오기 위해 해당 div를 찾음
lyric_div = soup.find('div', {'id': 'd_video_summary'})

lyric = ""
for con in lyric_div.contents:
    # html을 가져옴
    if 'NavigableString' in str(type(con)):
        if lyric != "":
            lyric += " "
        lyric += con.text.strip()
print(lyric)

# 제목 정보를 가져오기 위해 해당 div를 찾음
title_div = soup.find('div', {'class': 'song_name'})
title = title_div.find('strong').find_next_sibling(string=True).text
print(title)

song_detail_list.append({'keyword': key,
                        'song_no': code,
                        'title': title,
                        'lyric': lyric})

driver.close()
```

```
<div class="section_lyric">
  <div class="page_header bline">☹️</div>
  <div class="wrap_lyric" id="lyricArea">
    <div class="lyric" id="d_video_summary"> == $0
      <!-- height:auto; 로 변경시, 확장됨 -->
      " You know"
      <br>
      "내 스타일이 아닌 음악을 들어도"
      <br>
      "You know"
      <br>
      "좋아하지 않는 음식을 먹어도"
      <br>
      "우산 없이 비가 와 홀딱 다 젖어도 좋아"
      <br>
      "I love it because I love you"
      <br>
      <br>
      "우리 관계 디비디비딩"
      <br>
      "매일 봐도 처음같이 비기비기닝"
      <br>
      "춤추고 싶어 빌리빌리진"
      <br>
      "우리 앞 우리 옆 시기시기질투"
```

section.my_fold div div#conts div.section_lyric div#lyricArea.wrap_lyric div#d_v
콘솔 새로운 기능 × 문제
Highlights from the Chrome 119 update

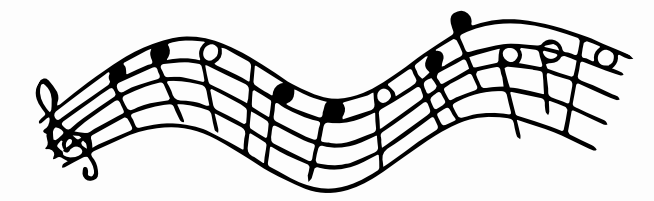
A	B	C	D	E	F	G	H	I	J	K
keyword	song_no	title	lyric							
GN0100	36382580	헤어지자 말해요	헤어지자고 말하려 오늘 너에게 가다가 우리 추억 생각해 봤어 처음 본 네 얼굴 마주친							
GN0100	36616378	사막에서 꽃을 피우듯	리에 첫눈을 보다가 문득 고백했던 그 순간 가보고 싶었던 식당 난생처음 준비한 선물							
GN0100	36595401	잘 지내자, 우리 (여름날	헤어지자 말해요 나는 사실 그대에게 좋은 사람이 아네요 그대 이제 날 떠난다 말해요							
GN0100	36518341	나에게 그대만이	면 나는 어쩔 수 없을 걸 문득 너의 사진 보겠지 새로 사귄 친구 함께 웃음 띤 네 얼굴 !							
GN0100	4446485	너의 모든 순간	먼저 헤어지자 말해요 나는 사실 그대에게 좋은 사람이 아네요 그대 이제 날 떠난다 말							
GN0100	34061322	사랑은 늘 도망가	볼 수 있을까 이기적인 거 나도 잘 알아 그땐 그럴 수밖에 없던 어린 내게 한 번만 더 기							
GN0100	34657844	사랑인가 봐	게 좋은 사람이 되고 싶었어 영영 다신 못 본다 해도 그댈 위한 이 노래가 당신을 영원히							

멜론 개별 곡 페이지
html 요소 속성 분석

selenium의 webdriver를 활용
수집한 곡 번호 해당 페이지
제목, 가사 수집

html 문법의 가사
BeautifulSoup을 활용
태그 제거

xlsx 파일로 저장



```

for i, line in enumerate(lyrics):
    try:
        d = okt.pos(line, norm=True, stem=True)
        r = []
        for word in d:
            if not word[1] in ['Josa', 'Eomi', 'Punctuation']:
                r.append(word[0])
        rl = (" ".join(r)).strip()
        result.append(rl)
        result_titles_list.append(titles_list[i])
        result_keyword_list.append(keyword_list[i])
        result_song_no_list.append(song_no_list[i])
    except Exception as ee:
        print(ee)

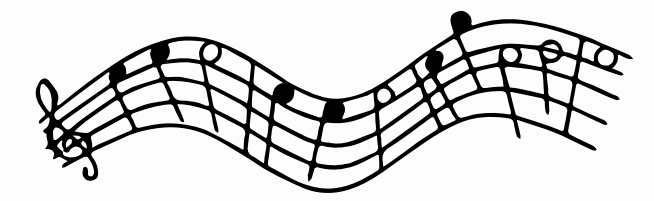
```

수집한 원본가사 형태소 분석
konlpy Okt 활용
조사, 어미, 문장부호 제거

analyzed_lyric 컬럼 생성

csv 파일로 저장

A	B	C	D	E	F	G	H
genre	song_no	title	analyzed_lyric				
GN0100	36382580	헤어지자 말해요	헤어지자 말 하다 오늘 너 가다가 우리 추억 생각 하다 보다 처음 보다 네 얼굴 마주				
GN0100	36616378	사막에서 꽃을 피우듯	거리 첫 눈 보다 문득 고백 하다 그 순간 가보다 싶다 식당 난생처음 준비 선물 고맙				
GN0100	36595401	잘 지내자, 우리 (여름날 우	말 하다 나 사실 그대 좋다 사람 아 네 그대 이제 날 떠나다 말 하다 잠시 이 행복 느				
GN0100	36518341	나에게 그대만이	사진 보다 새롭다 사귀다 친구 함께 웃음 띤 네 얼굴 보다 말 하다 수 없다 묘 감정 들				
GN0100	4446485	너의 모든 순간	좋다 사람 아 네 그대 이제 날 떠나다 말 하다 잠시 이 행복 느끼다 고맙다 하다 번은				
GN0100	34061322	사랑은 늘 도망가	다 어리다 내다 하다 번만 더 기회 주다 그 대다 정말 사랑 하다 말 하다 나 사실 그대				
GN0100	34657844	사랑인가 봐	위 이 노래 당신 영원하다 사랑 하다 테				

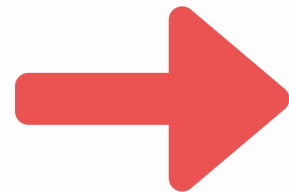


```
# CSV 파일 읽기
file_path = 'analyzed.csv'
df = pd.read_csv(file_path, encoding='cp949')

# UTF-8 인코딩으로 저장
output_file_path = 'analyzed_utf8.csv'
df.to_csv(output_file_path, index=False, encoding='utf-8')
```

csv 파일 읽어오는 것에서
지속적인 에러 발생

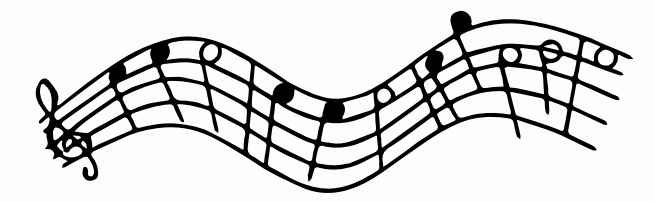
```
genre,song_no,title,analyzed_lyric
GN0100,36382580,헤어지자 말해요,헤어지
GN0100,36616378,사막에서 꽃을 피우듯,
GN0100,36595401,"잘 지내자, 우리 (여
GN0100,36518341,나에게 그대만이,더 이
GN0100,4446485,너의 모든 순간,이윽고
GN0100,34061322,사랑은 늘 도망가,눈물
GN0100,34657844,사랑인가 봐,너 함께
```



```
genre,song_no,title,analyzed_lyric
GN0100,36382580,헤어지자 말해요,헤어지
GN0100,36616378,사막에서 꽃을 피우듯,
GN0100,36595401,"잘 지내자, 우리 (여
GN0100,36518341,나에게 그대만이,더 이
GN0100,4446485,너의 모든 순간,이윽고
GN0100,34061322,사랑은 늘 도망가,눈물
GN0100,34657844,사랑인가 봐,너 함께
```

encoding cp949로 되어있음
확인

utf-8로 인코딩 후 저장



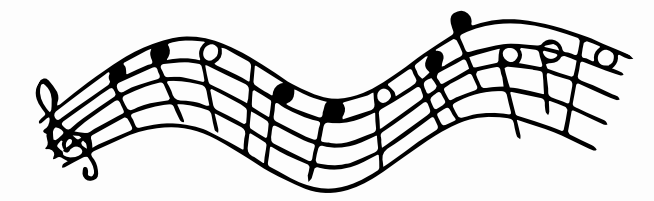
```
df = pd.read_csv('analyzed_utf8.csv')
df.head()

data = df['analyzed_lyric'].values
okt = Okt()
result = []
for line in data:
    d = okt.pos(line, norm=True, stem=True)
    r = []
    for word in d:
        if not word[1] in ['Josa', 'Eomi', 'Punctuation']:
            r.append(word[0])
    result.append(r)
fasttext_model = fasttext.FastText(result, vector_size=100, window=8, min_count=2, sg=1)
fasttext_model.save('music.model')
```

단어 추출하여 리스트에 저장

전처리된 데이터를 사용하여
FastText 모델을 학습

music.model로 저장



```

music_recommend
├── .pytest_cache
├── data_crawling
├── static
│   ├── abc.png
│   ├── scripts.js
│   └── styles.css
├── templates
│   ├── index.html
│   └── store.html
└── app.py
    ├── last_merged_data.csv
    ├── lyrics_embeddings.pkl
    ├── music.model
    └── music.model.wv.vectors

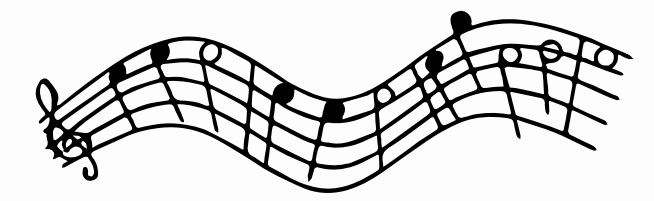
@app.route(rule: "/", methods=["GET", "POST"])
def index():
    if request.method == "POST":
        # 폼 데이터를 처리하고 원하는 동작을 수행하세요
        content = request.form["content"]
        top_10_songs = remove_duple_top_songs(num: 10, content)
        top_50_songs = get_top_songs(num: 50, content)
        gen_rate = get_genre_rate(top_50_songs)
        img_data = make_graph(gen_rate)

        return render_template(template_name_or_list: "store.html",
                               top_10_songs=top_10_songs,
                               top_50_songs=top_50_songs,
                               gen_rate=gen_rate,
                               img_data=img_data) # top_songs 변수를 store.html로 전달
    else:
        return render_template("index.html") # top_songs

@app.route(rule: "/store", methods=["GET", "POST"])
def store():
    # 다른 페이지(예: store.html)의 뷰 코드
    return render_template("store.html")

```

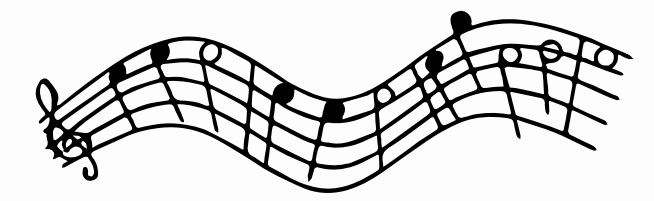
웹 서비스를 위해
flask 활용



```
def get_embedding(text, model):
    words = text.split()
    word_embeddings = [model.wv[word] for word in words if word in model.wv]
    if not word_embeddings:
        return np.zeros(model.vector_size)
    return np.mean(word_embeddings, axis=0)

def preprocess_text(text):
    okt = Okt()
    result = []
    d = okt.pos(text, norm=True, stem=True)
    r = []
    for word in d:
        if not word[1] in ['Josa', 'Eomi', 'Punctuation']:
            r.append(word[0])
    # 토큰화된 단어들을 다시 문자열로 결합
    return ' '.join(r)
```

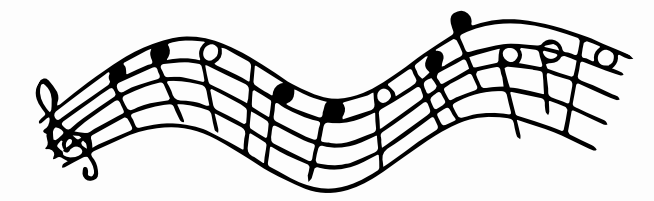
사용자에게 입력 받은 텍스트
model에 학습시킨
가사 데이터와 동일하게 전처리



```
def remove_duple_top_songs(num, content):
    # 사용자 입력의 임베딩 생성
    user_embedding = get_embedding(preprocess_text(content), model)
    # 저장된 임베딩 로드
    lyrics_embeddings = pd.read_pickle('lyrics_embeddings.pkl')
    # 코사인 유사도 계산
    similarity_scores = cosine_similarity([user_embedding], list(lyrics_embeddings['embedding']))[0]
    # 유사도 점수에 따라 내림차순으로 정렬
    lyrics_data['similarity_score'] = similarity_scores
    sorted_songs = lyrics_data.sort_values(by='similarity_score', ascending=False)
    # 중복 song_no를 제거하고 상위 10개의 곡 선택
    top_songs = []
    seen_song_no = set() # 이미 선택한 song_no를 추적하기 위한 집합
    for _, row in sorted_songs.iterrows():
        song_no = row['song_no']
        if song_no not in seen_song_no:
            top_songs.append(row.to_dict())
            seen_song_no.add(song_no)
            if len(top_songs) == num:
                break
    return top_songs
```

10개 곡 추천을 위한
유사도 가장 높은 순위 매김
(장르 중복) 중복 song_no 제거

10개 곡 화면에 출력



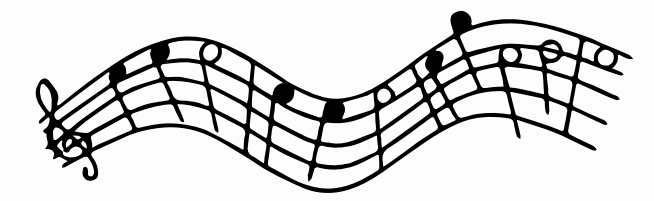
```
def get_genre_rate(list):
    genre_rate = [] # 결과를 담을 리스트 초기화
    for song_info in list:
        genre = song_info['genre']
        genres_list.append(genre)
    print(genres_list)
    # 각 장르의 갯수 계산
    genre_counts = Counter(genres_list)
    # 전체 곡 수
    total_songs = len(genres_list)
    # 각 장르의 비율 계산
    genre_ratios = {genre: count / total_songs for genre, count in genre_counts.items()}
    # 결과 출력
    for genre in possible_genres:
        count = genre_counts.get(genre, 0)
        ratio = round(genre_ratios.get(genre, 0.0), 2)
        print(f'Genre: {genre}, Count: {count}, Ratio: {ratio:.2%}')
        genre_rate.append({'Genre': genre, 'Count': count, 'Ratio': ratio})
    return genre_rate
```

```
def make_graph(rate):
    genre_names = {...}

    # 장르 코드를 장르명으로 변환
    for item in rate:
        item['Genre'] = genre_names.get(item['Genre'], item['Genre'])
    # 'Genre'와 'Ratio'를 추출하여 리스트로 저장
    genre_list = [item['Genre'] for item in rate]
    ratio_list = [item['Ratio'] * 100 for item in rate]
    # categories에 'Genre' 추가
    categories = [*genre_list, genre_list[0]]
    # grade1에 'Ratio' 추가
    grade1 = [*ratio_list, ratio_list[0]]
    # 나머지 그래프 그리는 코드
    label_loc = np.linspace(start=0, stop=2 * np.pi, num=len(grade1))
    plt.figure(figsize=(8, 8))
    ax = plt.subplot(polar=True)
    plt.xticks(label_loc, labels=categories, fontsize=13)
    ax.plot(label_loc, grade1, label='my mode', linestyle='dashed', color='lightcoral')
    ax.fill(label_loc, grade1, color='lightcoral', alpha=0.3)
    ax.legend()
    img_buffer = io.BytesIO()
    plt.savefig(*args: img_buffer, format='png')
    img_buffer.seek(0)
    img_data = base64.b64encode(img_buffer.read()).decode()
    # 이미지 데이터를 HTML 페이지로 전달
    return img_data
```

장르 추천을 위한
유사도가 높은 50개 곡의
장르 비율

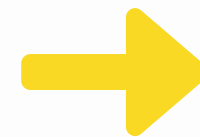
matplotlib 활용
방사형 그래프 이미지 저장
화면에 출력



한 정 적 인
데 이 터

melon 월간 차트
8개 장르 100개 곡

학습의 기준이 될
데이터가 한정적



풍 부 한
데 이 터

melon
다양한 차트에서
song_no 수집해서

다양하고
정확한 곡 추천

관 련 없 는
추 천

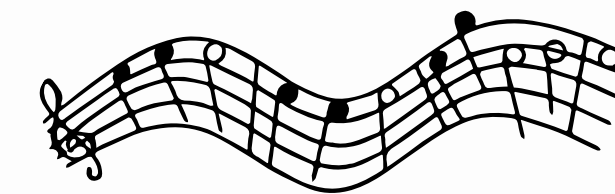
사용자의 입력과
유사성이 떨어지는
곡 추천



정 확 한
추 천

Word2Vec,
GloVe와 같은
다양한 모델 시도

하이퍼 파라미터
튜닝



Thanks!

