

MUSIC STREAMING WEBSITE

음악 스트리밍 웹사이트 개발



TEAM. STREAMING

2019117865 김나형
2017111502 김민영
2014016014 오강산
2019113779 옥명주



INDEX

1.담당 업무

2.개발 배경 및 목표

3.개발 내용

- UI

- 주요 기능

 - (1) 음악 플레이어

 - (2) 플레이리스트

 - (3) 검색창

- 서버 구동 및 배포

4. 시연

5. 기대 효과

6. 개선 사항





1. 담당업무





1. 담당업무



김나형 – 팀장

UI, Playlist, Search,
Click Play 구현



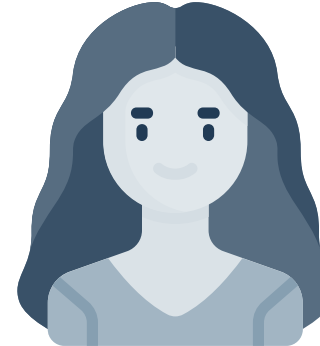
오강산

component 분리, 서버 구성 및 배포,
PPT 제작 및 발표



김민영

component 분리,
PPT 템플릿 제작



옥명주

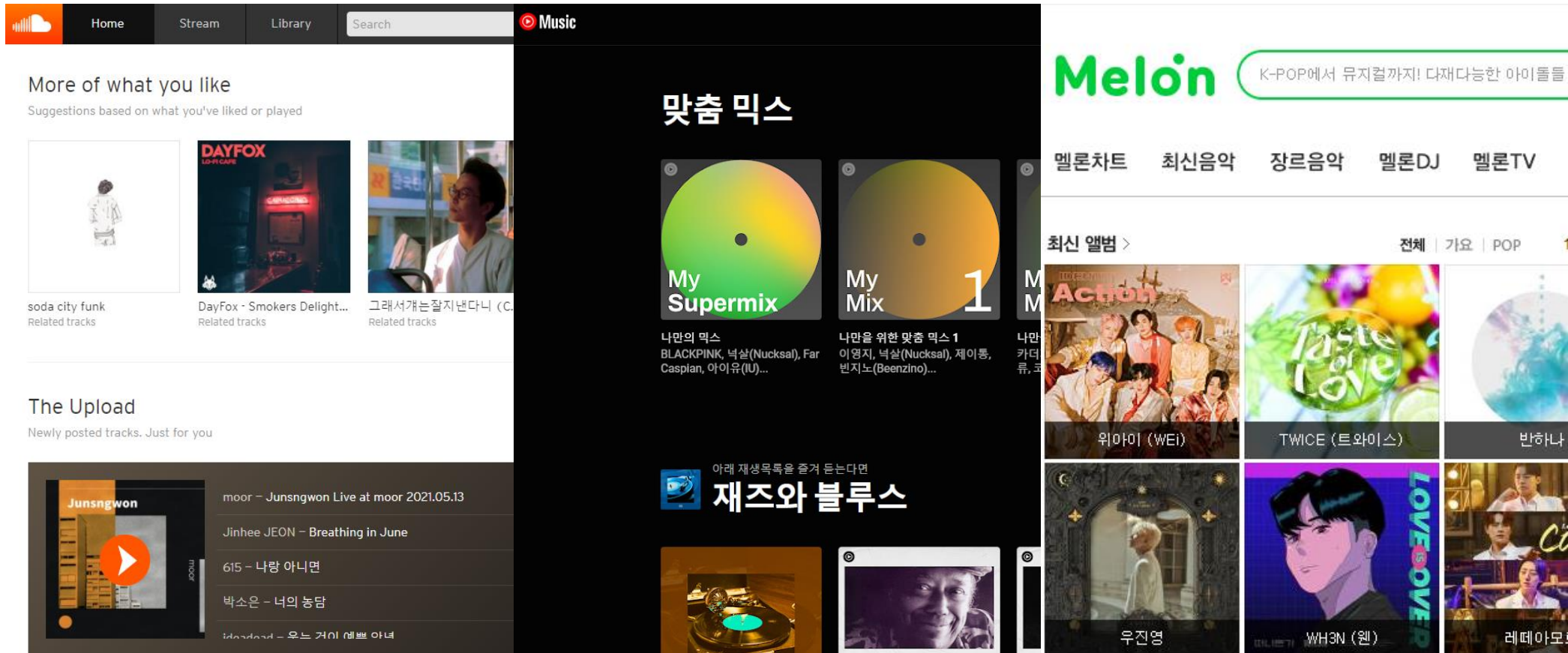
음악 재생, 플레이리스트 전체 재생,
Search 구현



2. 개발 배경 및 목표



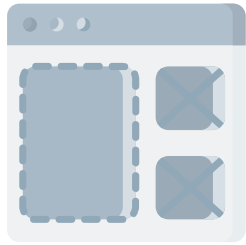
2. 개발 배경 및 목표 - 개발 배경



▲ 다양한 음악 스트리밍 사이트



2. 개발 배경 및 목표 - 개발 목표



메인 페이지



음악 플레이어



음악 검색

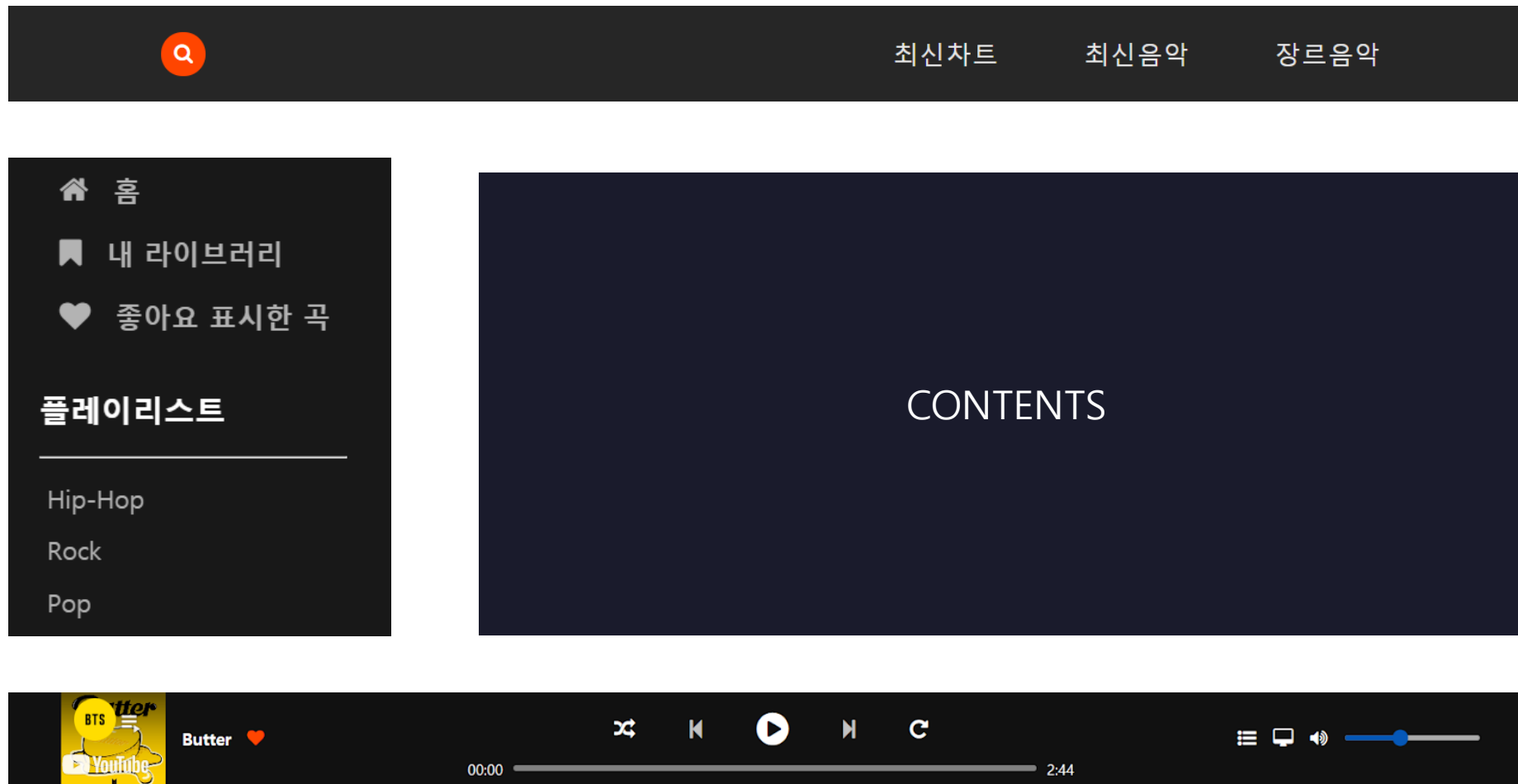


음원 차트 리스트



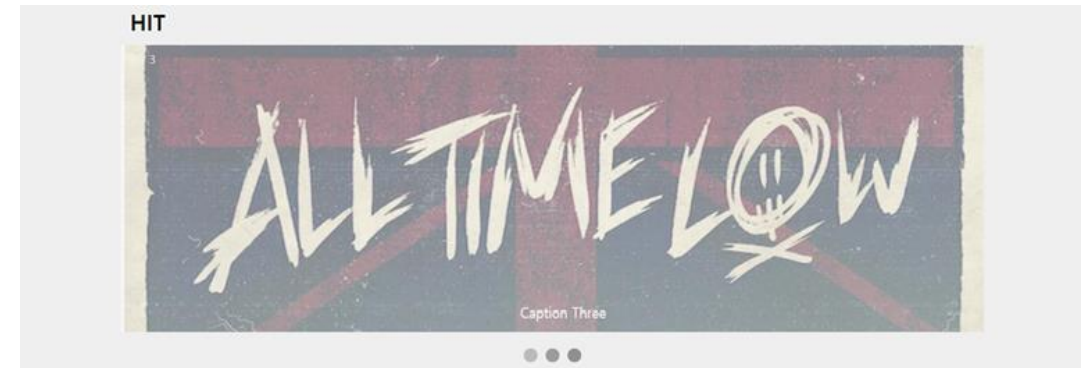
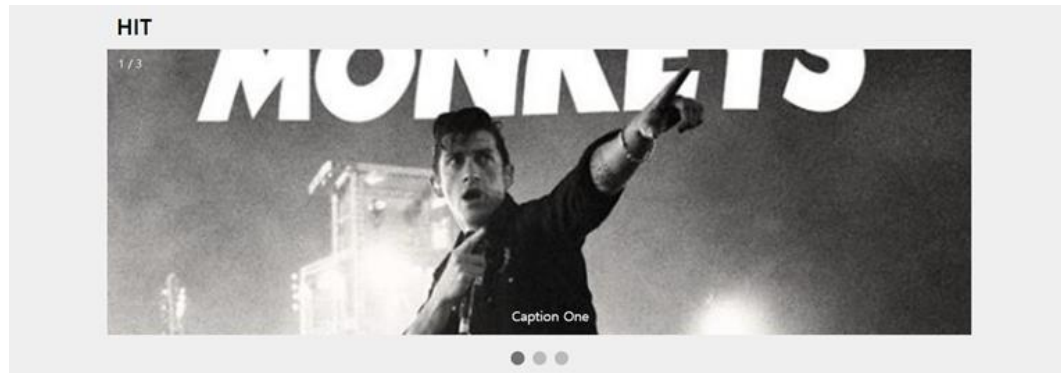
3. 개발 내용

3. 개발 내용 - UI





3. 개발 내용 - UI



▲ Vanilla js를 이용한 애니메이션 효과



3. 개발 내용 - 주요기능 (1) 음악플레이어



YouTube > Data API

Home Guides Reference Samples Support

Overview

- Activities
- Captions
- ChannelBanners
- Channels
- ChannelSections
- Comments
- CommentThreads
- I18nLanguages
- I18nRegions
- Members
- MembershipsLevels
- PlaylistItems
- Playlists
- Search
- Subscriptions
- Thumbnails
- VideoAbuseReportReasons
- VideoCategories
- Videos
- Watermarks
- GuideCategories
- Standard Query Parameters
- YouTube Data API Errors

Table of contents

Calling the API

Resource types

- Activities
- Captions
- ChannelBanners
- ChannelSections
- Channels
- CommentThreads
- Comments
- GuideCategories
- I18nLanguages
- I18nRegions
- Members
- MembershipsLevels
- PlaylistItems
- Playlists
- Search
- Subscriptions
- Thumbnails
- VideoAbuseReportReasons
- VideoCategories
- Videos
- Watermarks

Home > Products > YouTube > Data API > Reference

Rate and review

API Reference

The YouTube Data API lets you incorporate functions normally executed on the YouTube website into your own website or application. The lists below identify the different types of resources that you can retrieve using the API. The API also supports methods to insert, update, or delete many of these resources.

This reference guide explains how to use the API to perform all of these operations. The guide is organized by resource type. A resource represents a type of item that comprises part of the YouTube experience, such as a video, a playlist, or a subscription. For each resource type, the guide lists one or more data representations, and resources are represented as JSON objects. The guide also lists one or more supported methods (LIST, POST, DELETE, etc.) for each resource type and explains how to use those methods in your application.

Calling the API

The following requirements apply to YouTube Data API requests:

1. Every request must either specify an API key (with the `key` parameter) or provide an OAuth 2.0 token. Your API key is available in the [Developer Console's API Access](#) pane for your project.
2. You **must** send an authorization token for every insert, update, and delete request. You must also send an authorization token for any request that retrieves the authenticated user's private data.

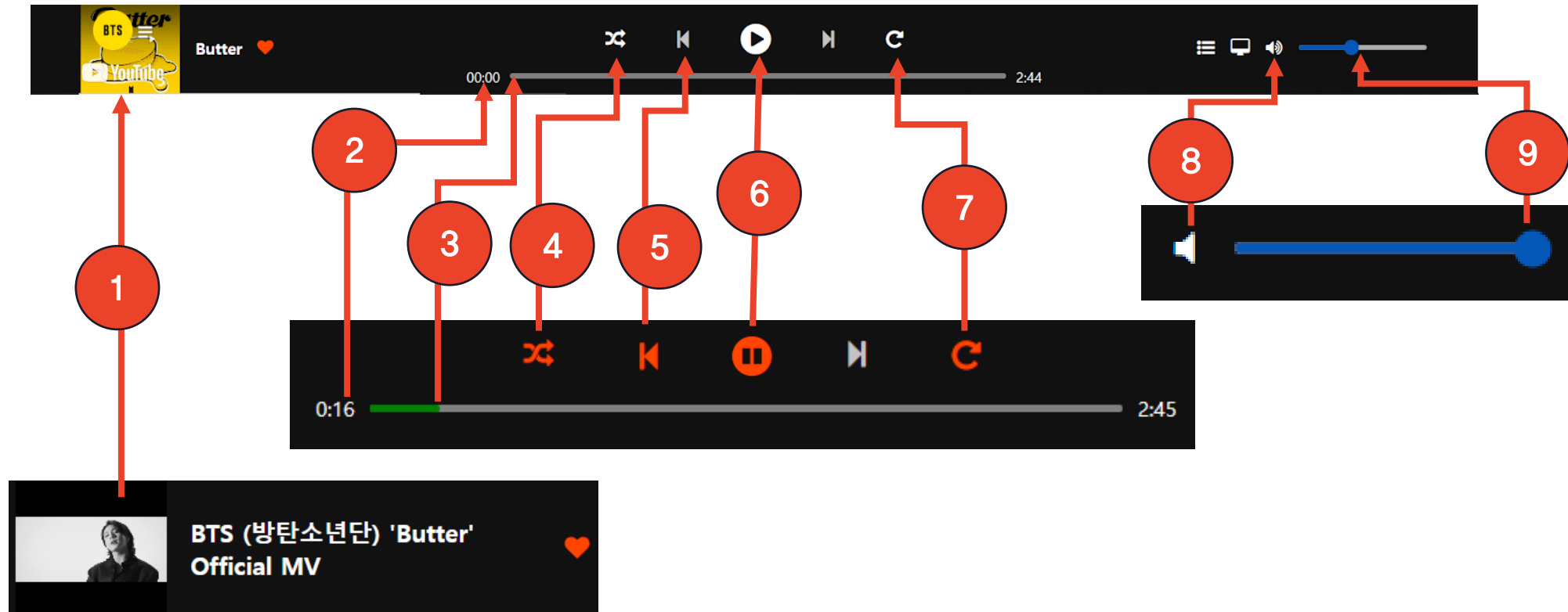
In addition, some API methods for retrieving resources may support parameters that require authorization or may contain additional metadata when requests are authorized. For example, a request to retrieve a user's uploaded videos may also contain private videos if the request is authorized by that specific user.

3. The API supports the OAuth 2.0 authentication protocol. You can provide an OAuth 2.0 token in either of the following ways:
 - Use the `access_token` query parameter like this: `?access_token=oauth2-token`
 - Use the HTTP `Authorization` header like this: `Authorization: Bearer oauth2-token`

▲ Youtube API reference page



3. 개발 내용 - 주요기능 (1) 음악플레이어



▲ 구현된 음악 플레이어 기능



3. 개발 내용 - 주요기능 (1) 음악플레이어

동영상 플레이어 로드

API의 JavaScript 코드가 로드된 후 API는 `YT.Player` 개체를 구성하여 페이지에 동영상 플레이어를 삽입할 수 있는 지점에서 `onYouTubeIframeAPIReady` 함수를 호출합니다. 아래 HTML 발췌 부분에서는 위 예제의 `onYouTubeIframeAPIReady` 함수를 보여줍니다.

```
var player;
function onYouTubeIframeAPIReady() {
  player = new YT.Player('player', {
    height: '360',
    width: '640',
    videoId: 'M7lc1UVf-VE',
    events: {
      'onReady': onPlayerReady,
      'onStateChange': onPlayerStateChange
    }
  });
}
```

▲ Youtube API reference

```
var player;
function onYouTubeIframeAPIReady() {
  audioPlayerInit();
  player = new YT.Player(players, {
    height: '100',
    width: '100',
    playerVars: {
      listType: 'playlist',
      autoplay: 1,
      controls: 0,
      fs: 0,
      loop: 1,
      list: 'PL4fGSI1pDJn6puJdseH2Rt9sMvt9E2M4i',
    },
    events: {
      'onReady': onPlayerReady,
      'onStateChange': onPlayerStateChange,
    }
  });
}
```

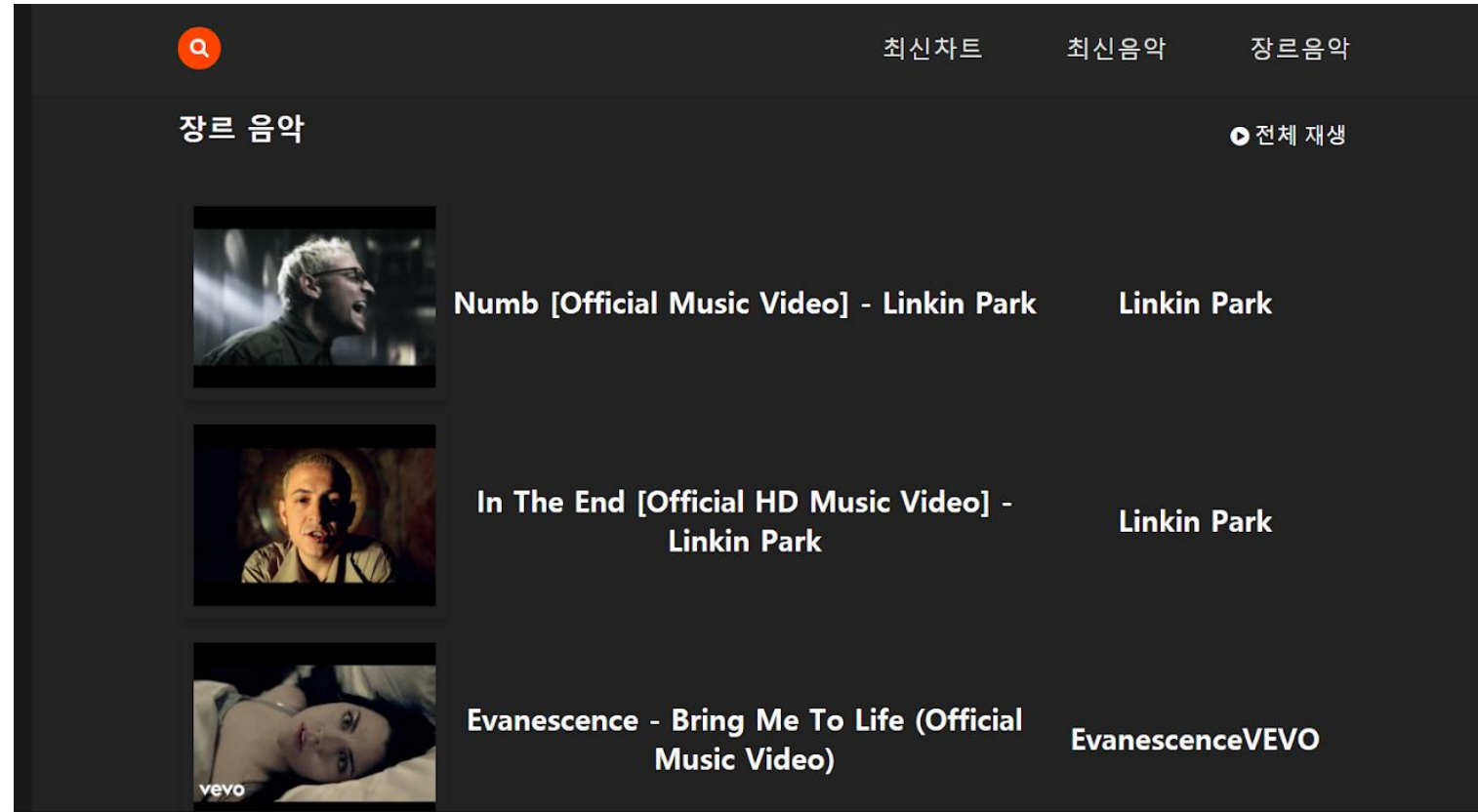
▲ API 적용 코드



3. 개발 내용 - 주요기능 (2) 플레이리스트

```
{
  "kind": "youtube#playlistItem",
  "etag": etag ✎,
  "id": string ✎,
  "snippet": {
    "publishedAt": datetime ✎,
    "channelId": string ✎,
    "title": string ✎,
    "description": string ✎,
    "thumbnails": {
      (key) ✎: {
        "url": string ✎,
        "width": unsigned integer ✎,
        "height": unsigned integer ✎
      }
    },
    "channelTitle": string ✎,
    "playlistId": string ✎,
    "position": unsigned integer ✎,
    "resourceId": {
      "kind": string ✎,
      "videoId": string ✎
    }
  }
}
```

▲ Youtube Playlist API JSON 반환값


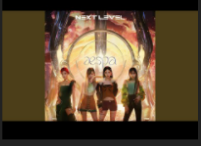
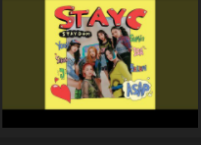



▲ 결과 출력 화면



3. 개발 내용 - 주요기능 (2) 플레이리스트

최신 차트

	Butter	BTS - Topic
	Next Level	aespa - Topic
	ASAP (ASAP)	STAYC - Topic
	Dun Dun Dance	OH MY GIRL - Topic

▲ 플레이리스트

전체 재생

```
function playlistAll(playlist_Id){
  player.stopVideo();
  player.loadPlaylist({
    'list': playlist_Id,
    'listType': 'playlist',
    'index': 0,
    'startSeconds': 0,
    'suggestedQuality': 'small'
  });
}
```

▲ 전체 음악 플레이리스트 재생

```
function clickPlay(vid){
  player.stopVideo();
  player.loadVideoById({
    'videoId': vid,
    'startSeconds': 0,
    'suggestedQuality': 'small'
  });
}
```

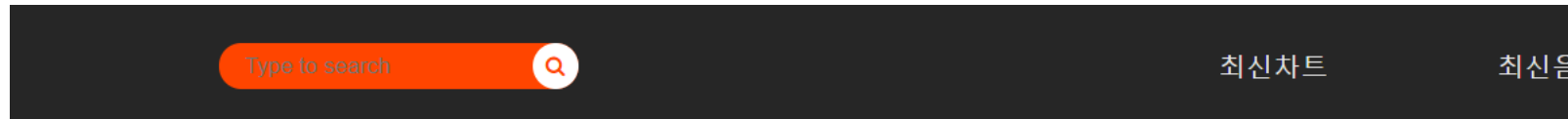
▲ 음악 선택 재생



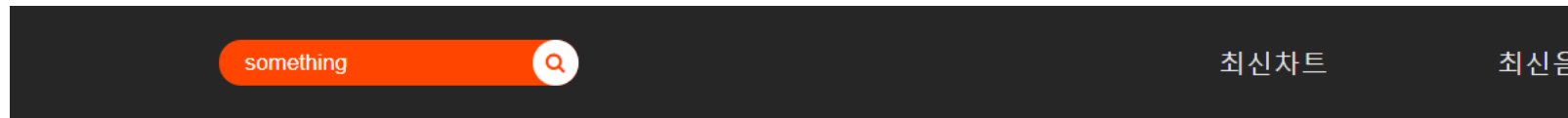
3. 개발 내용 - 주요기능 (3) 검색창



▲ 기본 형태



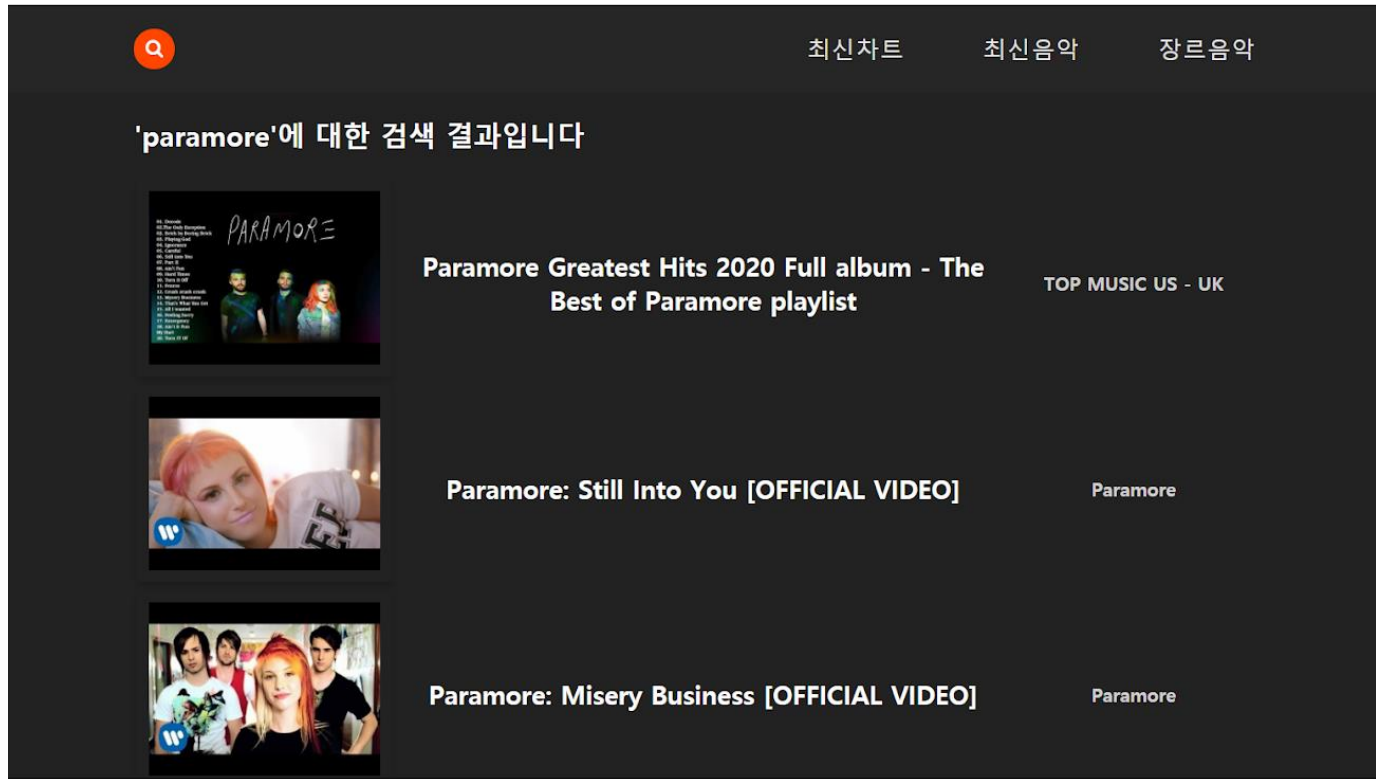
▲ hover



▲ 글자 입력



3. 개발 내용 - 주요기능 (3) 검색창



▲ 검색 결과 페이지

Youtube 검색 API
적용 코드

```
function sgetYouTube() {
    var url = decodeURI(location.href);
    var search = url.slice(url.indexOf('?') + 1, url.length);
    let albumDiv = $('#searchlist');
    let result = $('#searchresult');
    result.append("'" + search + "'에 대한 검색 결과입니다");
    $.ajax({
        type: "GET",
        dataType: "JSON",
        url: "https://www.googleapis.com/youtube/v3/search",
        data: { "key": "AIzaSyBnTgWwE0h0JKYooTahPdejU3LWBh2Ja4s",
            "part": "snippet",
            "maxResults": 50,
            "q": search
        }, // "kakao"를 search로 변경해야함.
        contentType: "application/json",
        success: function (jd) {
            let {items} = jd;
            let tempData = "";

            tempData += '<div class="albums"><table>';
            for(item of items){
                tempData += `
                <tr class="youtubeId" id="${item.id.videoId}">
                <th></th>
                <th><span class="title">${item.snippet.title}</span></th>
                <th><span>${item.snippet.channelTitle}</span></th>
                </tr>`;
            }
            tempData += '</table></div>'

            console.log(tempData);
            albumDiv.append(tempData);
        },
    });
}
```



3. 개발 내용 - 서버 구동 및 배포



```
const express = require('express')
const app = new express()
let port = process.env.PORT;
if (port == null || port == ""){
  port = 4000;
}
app.listen(port, ()=>{
  console.log('App listening ...')
})
app.get('/', homeController)
app.get('/recentChart', recentChartController)
app.get('/newestChart', newestChartController)
app.get('/genreChart', genreChartController)
app.get('/search', searchController)
```

▲ 서버 구동 코드



```
remote: ----> Build succeeded!
remote: ----> Discovering process types
remote:
remote: ~    Mis-cased procfile detected; ignoring.
remote: ~    Rename it to Procfile to have it honored.
remote:
remote:    Procfile declares types      -> (none)
remote:    Default types for buildpack -> web
remote:
remote: ----> Compressing...
remote:    Done: 35.4M
remote: ----> Launching...
remote:    Released v8
remote:    https://whispering-refuge-35717.herokuapp.com/ deployed to Heroku
remote:
remote: Verifying deploy... done.
```

▲ heroku 배포 성공 로그



4. 시연



4. 시연

<https://whispering-refuge-35717.herokuapp.com/>

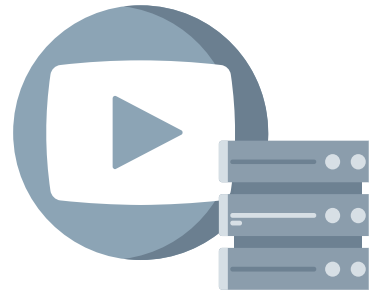
▲ heroku 배포 링크



5. 기대효과



5. 기대효과



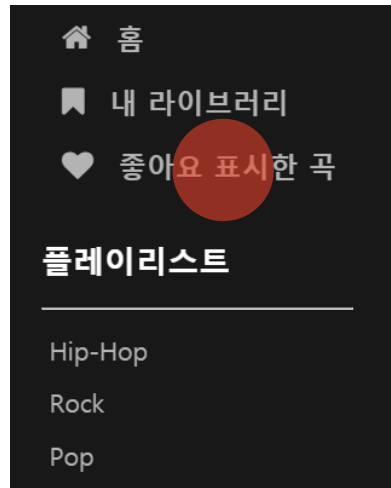
- ▶ 유튜브 DB에 접근, 다양한 음원 청취 가능
- ▶ 유튜브 DB와 연동된 Playlist의 정보 접근 가능



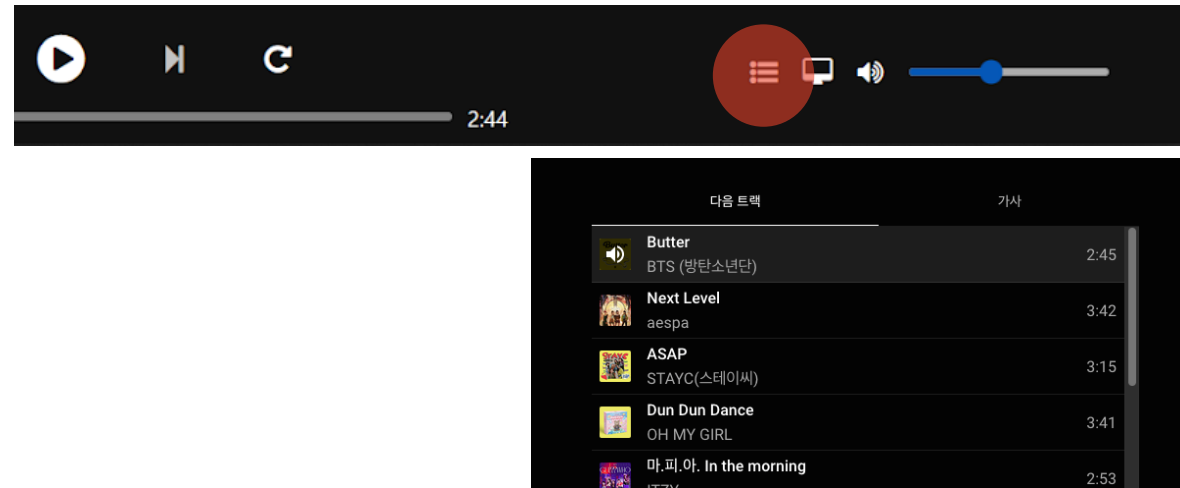
6. 개선 사항



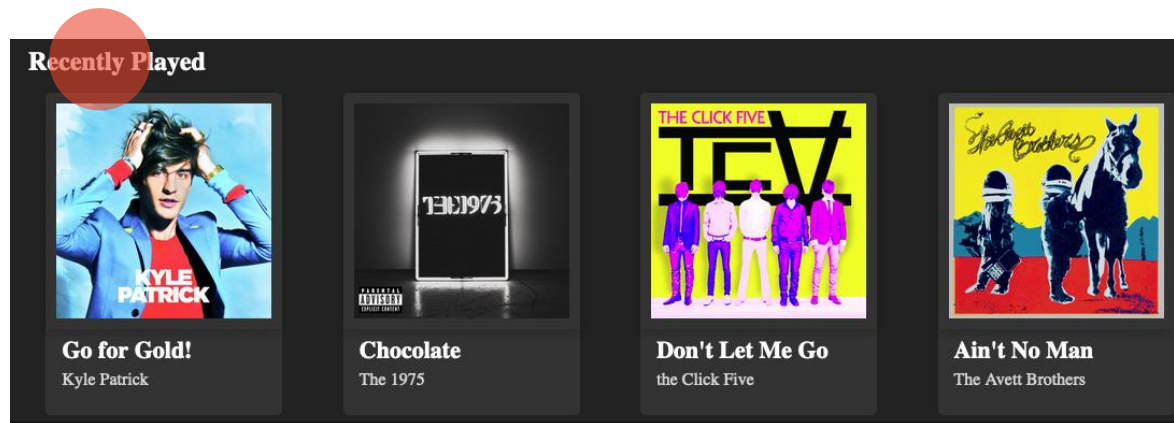
6. 개선 사항



▲ 플레이리스트 / 좋아요 리스트



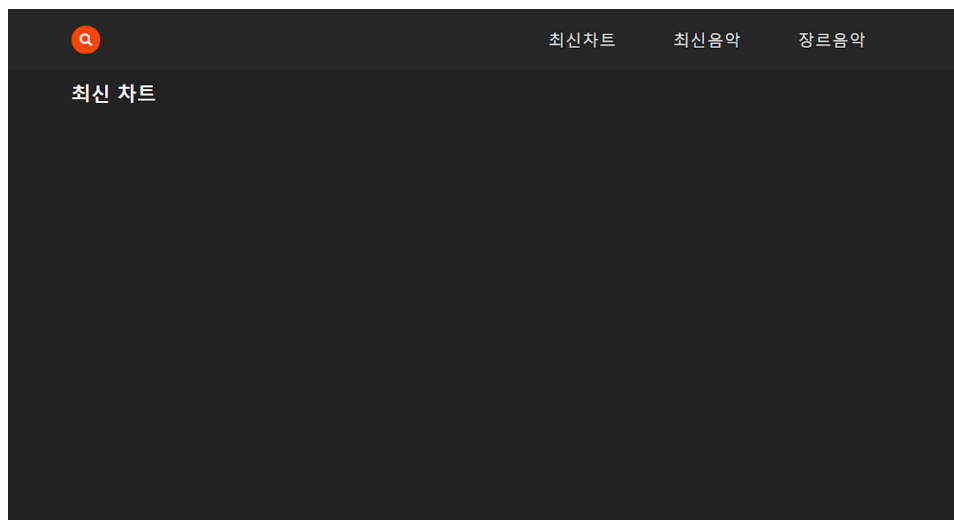
▲ 현재 재생목록 확인



◀ 최근 재생목록



6. 개선 사항



```

✖ www.googleapis.com/y...com%2FrecentChart:1
Failed to load resource: the server
responded with a status of 403 ()

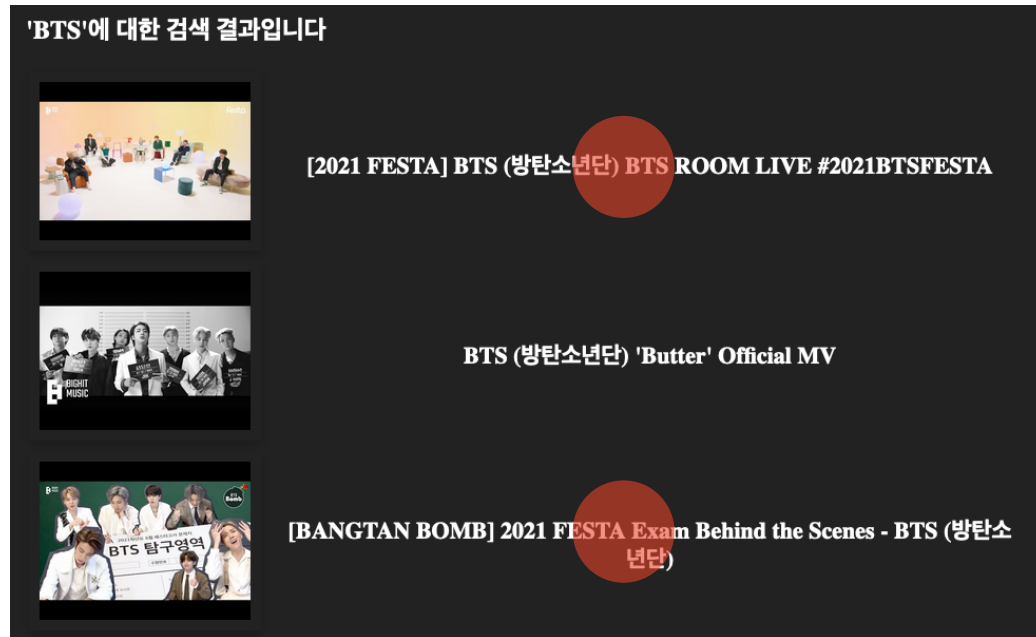
✖ www.googleapis.com/y...E0m&maxResults=50:1
Failed to load resource: the server
responded with a status of 403 ()

유튜브 요청 에러: search.js:45
유튜브 요청 에러: recentYTAPI.js:44
  
```

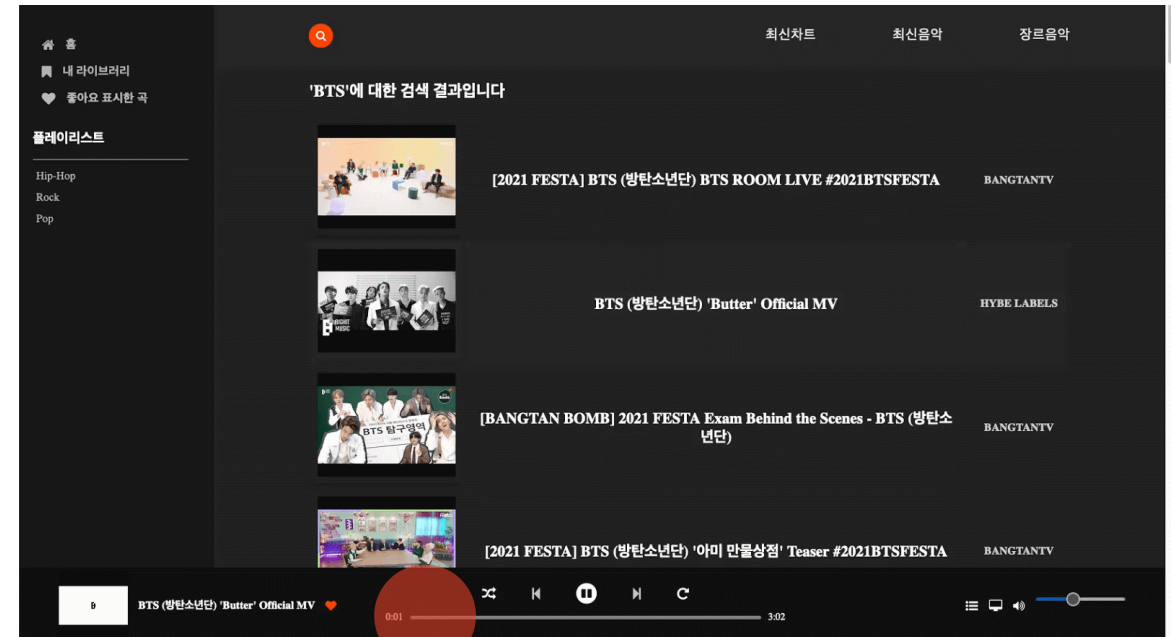
▲ Youtube API 요청 할당량 초과시 화면 / 에러 로그



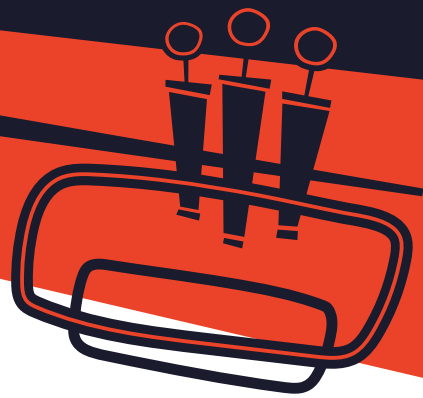
6. 개선 사항



▲ 음악이 아닌 영상도 같이 검색 결과 출력



▲ 화면 전환 시, 플레이어 재생이 초기화



Q & A

감사합니다



TEAM. STREAMING

2019117865 김나형
2017111502 김민영
2014016014 오강산
2019113779 옥명주

