



# Django REST Framework

## | Background

- ✓ Web Framework
- ✓ Django

## | Goal

- ✓ DRF(Django REST Framework)에 대한 이해

## | Problem

- ❖ DRF를 활용하여 음악 정보 관련 REST API 서버를 구축하시오.
- ❖ 프로젝트는 my\_api, 앱은 music으로 만들어 진행한다.
- ❖ Postman을 활용하여 각 HTTP Method에 맞는 요청을 테스트하시오.

## 1. Model & Admin

- 작성 할 모델 및 필드 정보
- admin 페이지를 활용하여 가수, 해당 가수가 부른 노래를 2개씩 작성한다.

```
class Artist(models.Model):
    name = models.CharField(max_length=200)

class Music(models.Model):
    artist = models.ForeignKey(Artist, on_delete=models.CASCADE)
    title = models.CharField(max_length=200)
```

# Django REST Framework



## 2. Serializer

### ArtistListSerializer

- 모든 가수의 정보를 반환하기 위한 Serializer
- id, name 필드 출력

### ArtistSerializer

- 상세 가수의 정보를 생성 및 반환하기 위한 Serializer
- id, name, music\_set, music\_count 필드 출력  
(music\_count 필드는 music\_set을 count한 결과이다.)

### MusicListSerializer

- 모든 음악의 정보를 반환하기 위한 Serializer
- id, title 필드 출력

### MusicSerializer

- 상세 음악의 정보를 생성 및 반환하기 위한 Serializer
- id, title, artist 필드 출력



### 3. url & view

#### GET & POST api/v1/artists/

- GET 요청인 경우 모든 가수의 id와 name 컬럼을 JSON으로 응답한다.
- POST 요청인 경우 가수의 정보를 생성한다.
  - 검증에 성공하는 경우 가수의 정보를 DB에 저장하고 생성된 가수의 정보와 201 Created를 응답한다.
  - 검증에 실패하는 경우 400 Bad Request를 응답한다.

#### GET api/v1/artists/<artist\_pk>/

- 특정 가수의 모든 컬럼을 JSON으로 응답한다.
- 특정 가수의 노래 정보와 노래의 개수 정보를 함께 응답한다.

#### POST api/v1/artists/<artist\_pk>/music/

- 특정 가수의 음악의 정보를 생성한다.
- 검증에 성공하는 경우 음악의 정보를 DB에 저장하고 생성된 음악의 정보와 201 Created를 응답한다.
- 검증에 실패하는 경우 400 Bad Request를 응답한다.

#### GET api/v1/music/

- 모든 음악의 id와 title 컬럼을 JSON으로 응답한다.

#### GET & PUT & DELETE api/v1/music/<music\_pk>/

- GET 요청인 경우 특정 음악의 모든 컬럼을 JSON으로 응답한다.
- PUT 요청인 경우 특정 음악의 정보를 수정한다.
  - 검증에 성공하는 경우 수정된 음악의 정보를 DB에 저장한다.
  - 검증에 실패하는 경우 400 Bad Request를 응답한다.
  - 수정이 완료된 이후에 수정된 음악의 정보를 응답한다.
- DELETE 요청일 경우 특정 음악의 정보를 삭제한다.
  - 삭제가 완료된 이후에 삭제한 음악의 id와 204 No Content를 응답한다.

❖ 각 요청에 해당하는 JSON 응답 결과와 **views.py**, **serializers.py** 코드를 마크다운에 작성하여 제출하시오.