[파이썬 트랙] 4회차 월말평가

| Background

Django Rest API
Django Framework

| Goal

API Service를 구현할 수 있다.

| 환경 설정

1) 개발환경 : python 3.9+, Django 3.2.X, vscode, Postman
2) 프로젝트 다운로드 : 프로젝트 압축 파일을 다운로드 후 압축 해제한다.
3) 가상 환경 설정 : 파이썬 가상 환경을 생성한다.
4) 패키지 설치 : 동봉된 requirements.txt에 기재된 패키지를 3)에서 생성한 가상 환경에서 설치한다.
5) 코드 작성시 작성한 코드에 대한 해설을 주석 형태로 상세하게 작성한다.
6) Postman을 활용하여 작성한 코드를 테스트할 수 있다.
7) 제출 : venv 폴더를 제외한 프로젝트 폴더를 압축하고 지역0반_홍길동 형식으로 이름을 변경하여 제출한다.

| 주의사항

1) python manage.py migrate 명령어를 이용하여 DB를 생성합니다.
2) 주어진 내용을 제외한 어떠한 문서나 소스 코드도 참고 할 수 없습니다.
3) views.py에 작성 되어있는 데코레이터 수정을 금지합니다.
4) 문제에 명시된 파일이 있는 경우, 반드시 해당 파일을 수정합니다.

소스코드 유사도 판단 프로그램 기준 코드가 유사하여 부정 행위로 판단 시, 0점 처리 및 학사 기준에 의거 조치 실시 예정

파이썬 트랙] 4회차 월말평가

| 문제1. MusicListSerializer (15)

GET방식의 /api/v1/music/ 요청이 들어올 때, 모든 음악 리스트를 조회할 수 있는 기능을 완성하려 한다. 이 때 아래 조건에 맞게 음악 정보를 직렬화 할 수 있는 클래스를 완성하시오.

1. **music/serializers.py** 에서 **MusicListSerializer** 클래스를 수정하여 코드를 완성합니다.
2. 음악 데이터는 **모든 정보들이 아닌 title과 release_date 만** JSON에 담겨지도록 합니다.

| 문제2. music_list POST 요청 (15)

POST방식의 /api/v1/music/ 요청이 들어올 때, 데이터베이스에 음악 정보를 저장하는 음악 등록 기능을 완성하시오.

1. **music/views.py** 의 **music_list**에서 **MusicSerializer**를 활용하여 저장 기능을 구현합니다.
2. 등록할 데이터의 유효성 검사가 이루어져야 합니다.
3. 정상적으로 음악을 등록했다면, 등록된 음악 정보를 JSON 형태로 반환해야 합니다.

파이썬 트랙] 4회차 월말평가

| 문제3. music_detail GET 요청 (10)

GET 방식의 /api/v1/music/<music_pk>/ 요청이 들어올 때, 등록된 음악의 모든 정보를 JSON 형태로 반환 받을 수 있도록 음악 상세 조회 기능을 완성하시오.

1. **music/views.py** 에서 **music_detail** 함수를 수정하여 코드를 완성합니다.
2. 해당 pk의 음악 정보가 존재하지 않으면, 404 상태 코드를 반환해야 합니다.

| 문제4. music_detail DELETE 요청(10)

DELETE 방식의 /api/v1/music/<music_pk>/ 요청이 들어올 때, 해당 음악 정보를 삭제하는 기능을 완성하시오.

1. **music/views.py** 에서 **music_detail** 함수를 수정하여 코드를 완성합니다.
2. 음악 정보를 정상적으로 삭제하였다면, **key**를 **'delete', value**를 **music_pk**로 갖는 JSON 데이터를 반환해야 합니다.
   (예시: 5번글이 정상적으로 삭제되었다면 {'delete': 5} 반환)

| 문제5. music_detail PUT 요청 (10)

PUT 방식의 /api/v1/music/<music_pk>/ 요청이 들어올 때, 음악 정보를 수정하는 기능을 완성하시오.

1. **music/views.py** 에서 **music_detail** 함수를 수정하여 코드를 완성합니다.
2. 수정할 데이터의 유효성 검사가 이루어져야 합니다.
3. 음악 정보 수정이 정상적으로 완료되면, 수정된 음악 정보를 JSON 형태로 반환해야 합니다.

파이썬 트랙] 4회차 월말평가

| 문제6. ReviewSerializer (10)

음악 리뷰 정보를 직렬화 할 수 있는 클래스를 완성하시오.

1. **music/serializers.py**의 **ReviewSerializer** 클래스를 수정하여 코드를 구현합니다.
2. 리뷰 정보는 id, content, created_at, updated_at, music 등의 모든 필드들이 포함되어야 합니다.
3. music 필드는 **참조하고 있는 음악의 모든 정보**가 포함되어야 합니다.
4. music 필드는 사용자가 입력할 수 없도록 **read only** 가 되도록 설정합니다.

| 문제7. review_list GET 요청 (10)

GET방식의 /api/v1/review/ 요청이 들어올 때 모든 리뷰 정보를 JSON 형태로 반환하는 전체 리뷰 조회 기능을 완성하시오.

1. **music/views.py**의 **review_list** 함수를 수정하여 코드를 구현합니다.
2. **ReviewSerializer**를 활용하여 구현합니다.
3. 음악 정보에 상관없는 모든 리뷰 정보를 반환해야 합니다.

| 문제8. review_create POST 요청 (5)

POST방식의 /api/v1/music/<music_pk>/review/ 요청이 들어올 때 데이터베이스에 음악 리뷰 정보를 저장하는 리뷰 작성 기능을 완성하시오.

1. **music/views.py**의 **review_create** 함수를 수정하여 코드를 완성합니다.
2. **ReviewSerializer**를 활용하여 구현합니다.
3. music 필드는 요청하는 사용자가 직접 입력하지 않습니다.
4. 데이터의 유효성 검사가 이루어져야 합니다.
5. 유효성 검사를 통과하지 못하면 해당 정보와 함께 400 상태 코드가 반환되야 합니다.
6. 리뷰 생성 후, 생성된 리뷰 정보를 JSON 형태로 반환해야 합니다.
7. JSON 데이터와 함께 자원이 생성되었음을 나타내는 201 상태 코드도 같이 반환해야 합니다.

파이썬 트랙] 4회차 월말평가

| 문제9. review_detail GET 요청 (5)

GET 방식의 /api/v1/review/<review_pk>/ 요청이 들어올 때, 해당 리뷰의 모든 정보를 JSON 형태로 반환하도록 댓글 상세 조회 기능을 완성하시오.

1. **music/views.py**의 **review_detail** 함수를 수정하여 코드를 완성합니다.
2. 조회하는 리뷰가 존재하지 않을 경우, 404 상태 코드를 반환해야 합니다.

| 문제10. review_detail DELETE 요청 (5)

DELETE 방식의 /api/v1/review/<review_pk>/ 요청이 들어올 때, 해당 리뷰를 삭제하는 기능을 완성하시오.

1. **music/views.py**의 **review_detail** 함수를 수정하여 코드를 완성합니다.
2. 리뷰가 정상적으로 삭제되었다면, **key**를 **'delete', value**를 **review_pk**로 갖는 JSON 데이터를 반환해야 합니다.
   (예시: 5번 리뷰가 정상적으로 삭제되었다면 {'delete': 5} 반환)
3. JSON과 함께 204 상태 코드도 같이 반환해야 됩니다.
4. 삭제 대상 리뷰가 존재하지 않을 경우, 404 상태 코드를 반환해야 합니다.

| 문제11. MusicReviewCntSerializer (5)

음악 정보 상세 조회시 해당 음악에 작성된 리뷰의 개수를 직렬화하여 추가로 제공할 수 있도록 serializers.py의 MusicReviewCntSerializer 클래스를 수정하여 추가 기능을 완성하시오.

1. **field**명은 반드시 **review_count**로 정의합니다.
2. 추가한 review_count 필드와 Music의 모든 필드가 반환되도록 구성합니다.
3. **GET 방식의 /api/v1/music/<music_pk>/** 요청으로 결과를 확인할 수 있습니다.