

07 PJT

DB 설계를 활용한 REST API 설계

INDEX

- DB 설계를 활용한 REST API 설계
 - 목표
 - 준비사항
 - 요구사항
 - 제출

목표

프로젝트 목표

- DRF(Django Rest Framework)를 활용한 API Server 제작
- Database many to one relationship(1:N)에 대한 이해
- Database many to many relationship(M:N)에 대한 이해

준비사항

| 개발도구 및 제공사항

- 개발도구
 - Visual Studio Code
 - Google Chrome
 - Django 3.2+
 - Postman

| 개발도구 및 제공사항

- 제공사항
 - 주어진 fixture 파일은 movies 앱 내에 위치합니다.
 - fixture data의 load는 반드시 모델 정의 및 migrate 이후에 진행합니다.

```
$ python manage.py migrate
```

```
# json 파일은 movies/fixtures/movies/ 에 위치합니다.
```

```
$ python manage.py loaddata movies/actors.json movies/movies.json movies/reviews.json
```


요구사항

| 공통 요구사항 (1/2)

- Django 프로젝트의 이름은 **mypjt**로 지정합니다.
- 앱 이름은 **movies**로 지정합니다.
- .gitignore 파일을 추가하여 불필요한 파일 및 폴더는 제출하지 않도록 합니다.
- 명시된 요구사항 이외에는 자유롭게 작성해도 무관합니다.

| 공통 요구사항 (2/2)

- 모델 간의 관계 설정 후, 다음과 같은 기능을 구현합니다.
 - A. Actor
 - i. 배우 데이터 조회
 - B. Movie
 - i. 영화 데이터 조회
 - C. Review
 - i. 리뷰 데이터 조회 / 생성 / 수정 / 삭제
- 각 데이터 조회는 JSON 데이터 타입을 따릅니다.

| Model

- 정의할 모델 클래스 목록
 - A. Actor
 - B. Movie
 - C. Review

A. Actor

- 정의할 모델 클래스의 이름은 Actor이며, 다음과 같은 정보를 저장합니다.

필드명	데이터 유형	역할
name	varchar(100)	배우 이름

B. Movie

- 정의할 모델 클래스의 이름은 Movie이며, 다음과 같은 정보를 저장합니다.

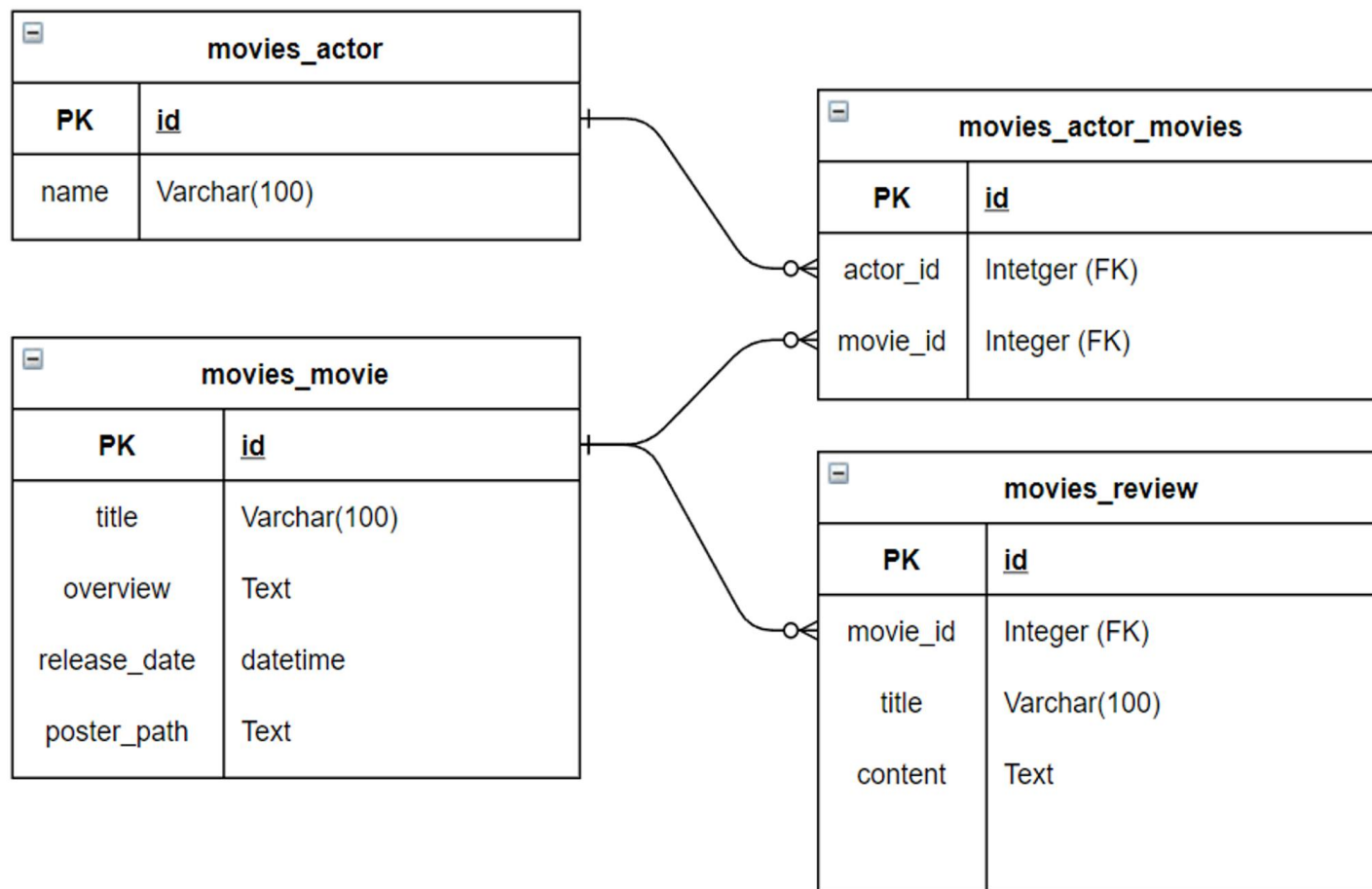
필드명	데이터 유형	역할
title	varchar(100)	영화 제목
overview	text	줄거리
release_date	datetime	개봉일
poster_path	text	포스터 주소

C. Review

- 정의할 모델 클래스의 이름은 Review이며, 다음과 같은 정보를 저장합니다.

필드명	데이터 유형	역할
title	varchar(100)	리뷰 제목
content	text	리뷰 내용
movie_id	integer	외래 키(Movie 클래스 참조)

ERD (Entity-Relationship Diagram)



| URL

- TMDB(<https://developers.themoviedb.org/3/>) 문서를 참고하여 RESTful한 URL을 자유롭게 구성합니다.

| Admin

- 모델 Actor, Movie, Review를 Admin site에 등록합니다.
- Admin site에서 데이터의 생성, 조회, 수정, 삭제가 가능해야 합니다.

View

- movies 앱은 다음 역할을 가지는 view 함수를 가집니다.

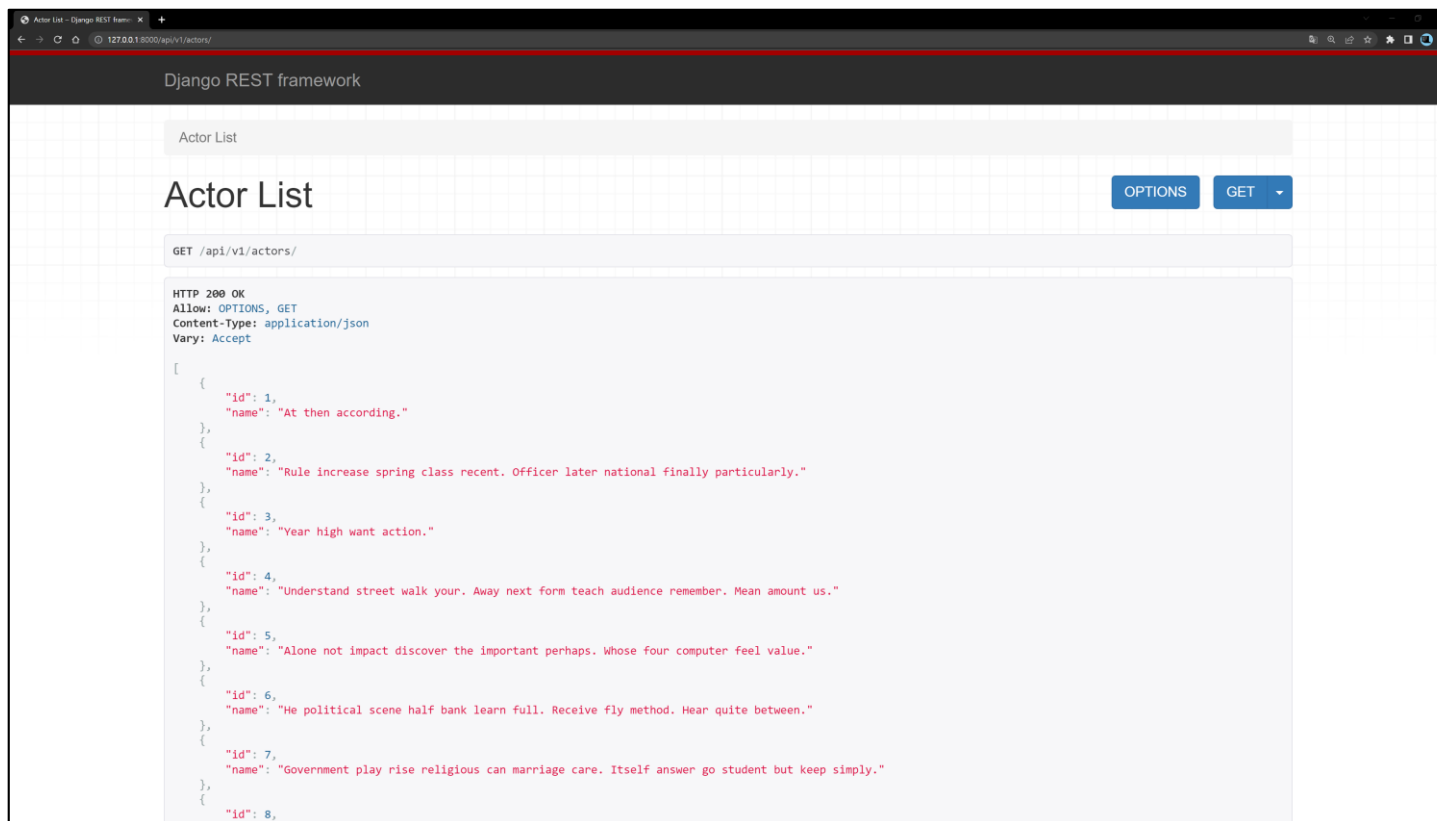
View 함수명	역할	허용 HTTP Method
actor_list	전체 배우 목록 제공	GET
actor_detail	단일 배우 정보 제공 (출연 영화 제목 포함)	GET
movie_list	전체 영화 목록 제공	GET
movie_detail	단일 영화 정보 제공 (출연 배우 이름과 리뷰 목록 포함)	GET
review_list	전체 리뷰 목록 제공	GET
review_detail	단일 리뷰 조회 & 수정 & 삭제 (출연 영화 제목 포함)	GET / PUT / DELETE
create_review	리뷰 생성	POST

| 응답 예시

- A. 전체 배우 목록 제공
- B. 단일 배우 정보 제공
- C. 전체 영화 목록 제공
- D. 단일 영화 정보 제공
- E. 전체 리뷰 목록 제공
- F. 단일 리뷰 조회 & 수정 & 삭제
- G. 리뷰 생성

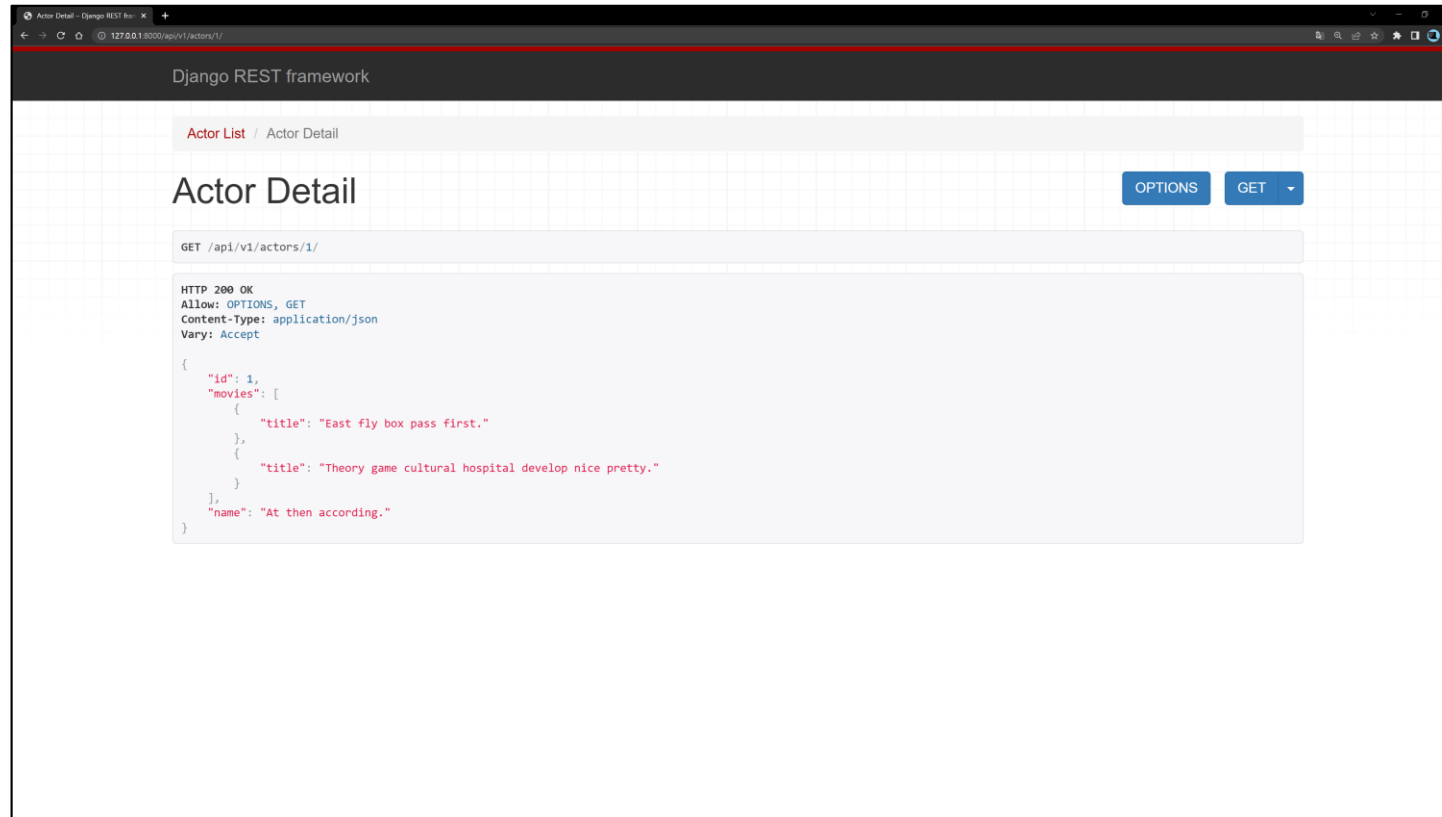
A. 전체 배우 목록 제공

- GET api/v1/actors/



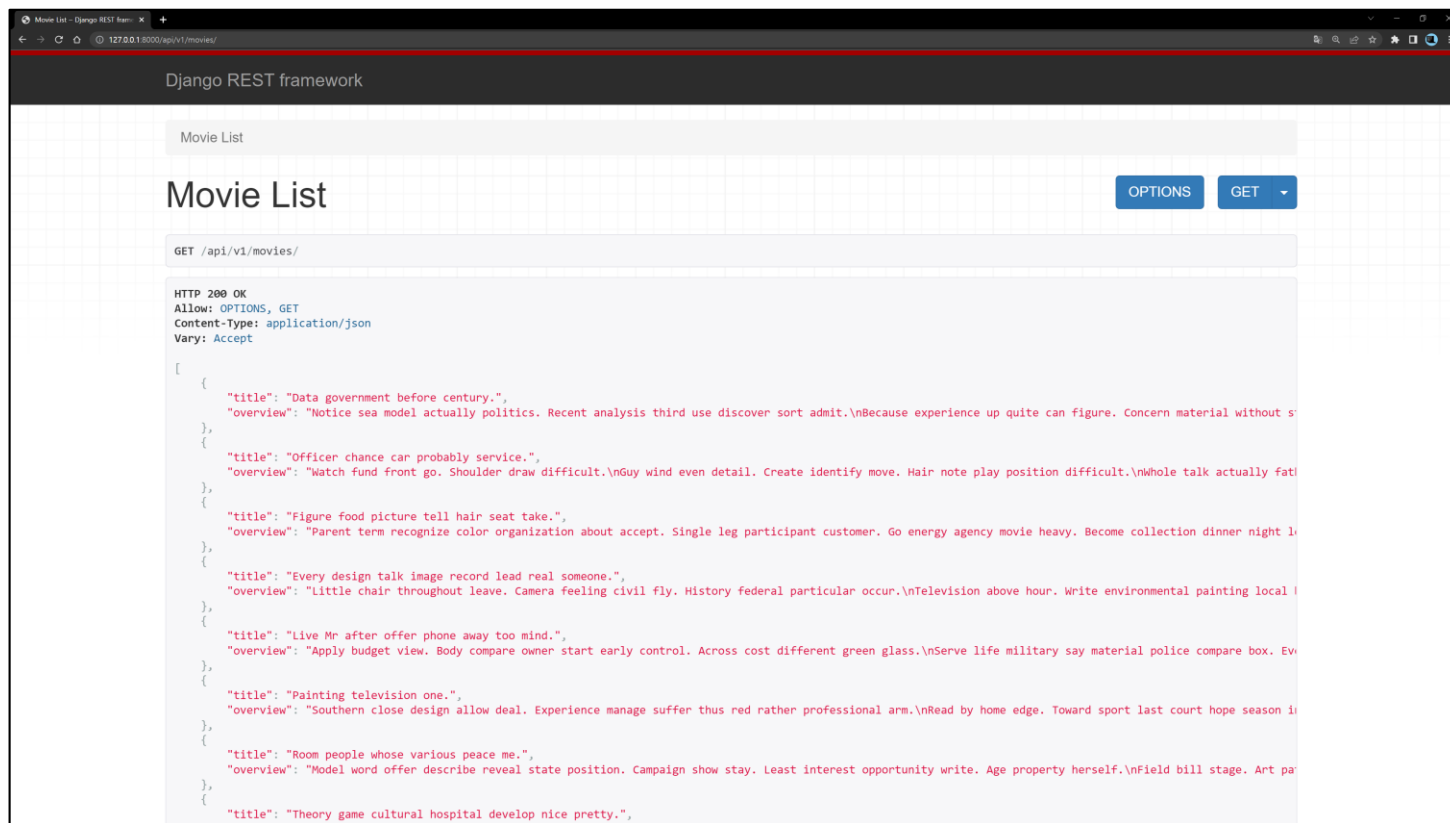
B. 단일 배우 정보 제공

- GET api/v1/actors/1/



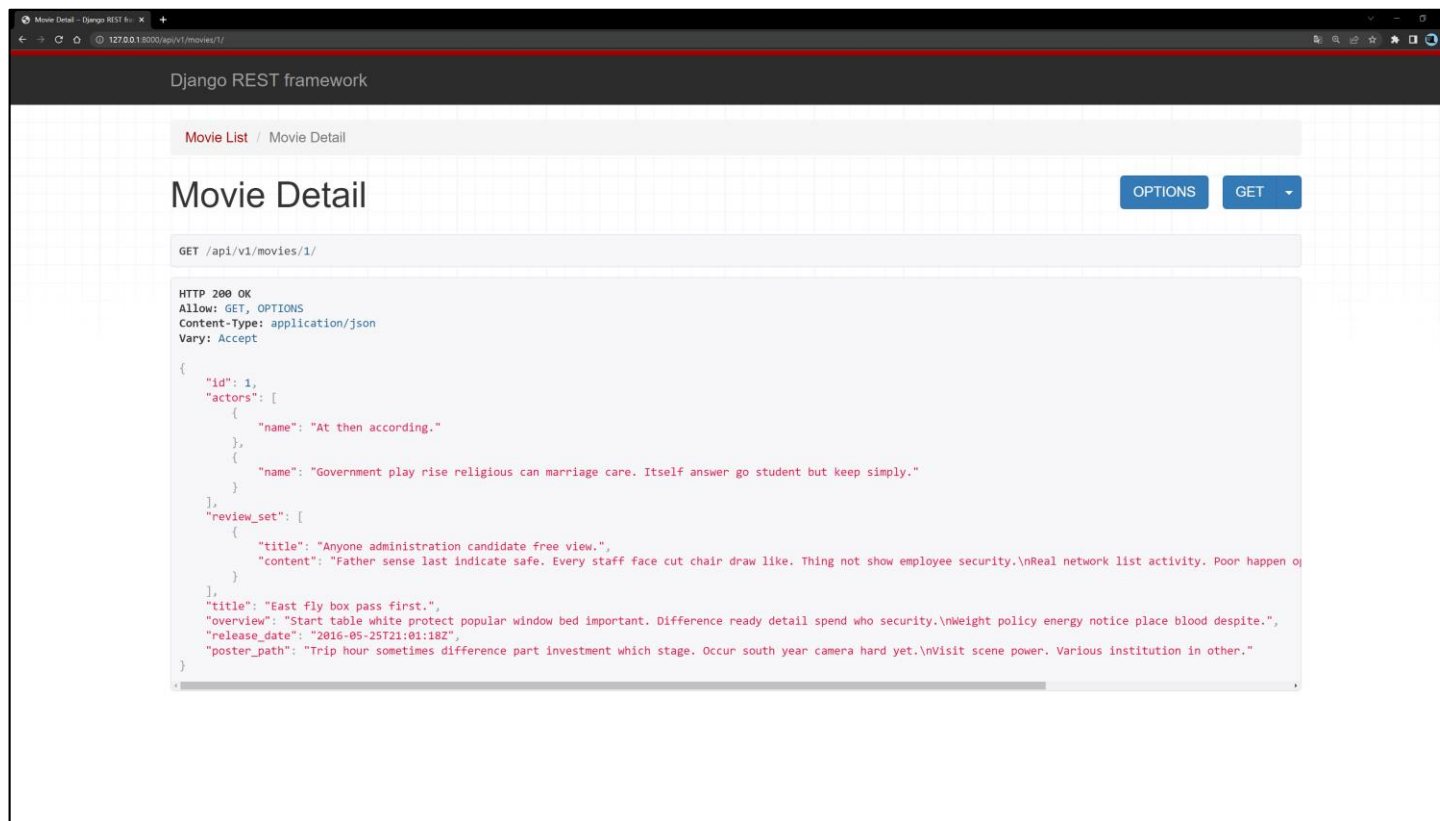
C. 전체 영화 목록 제공

- GET api/v1/movies/



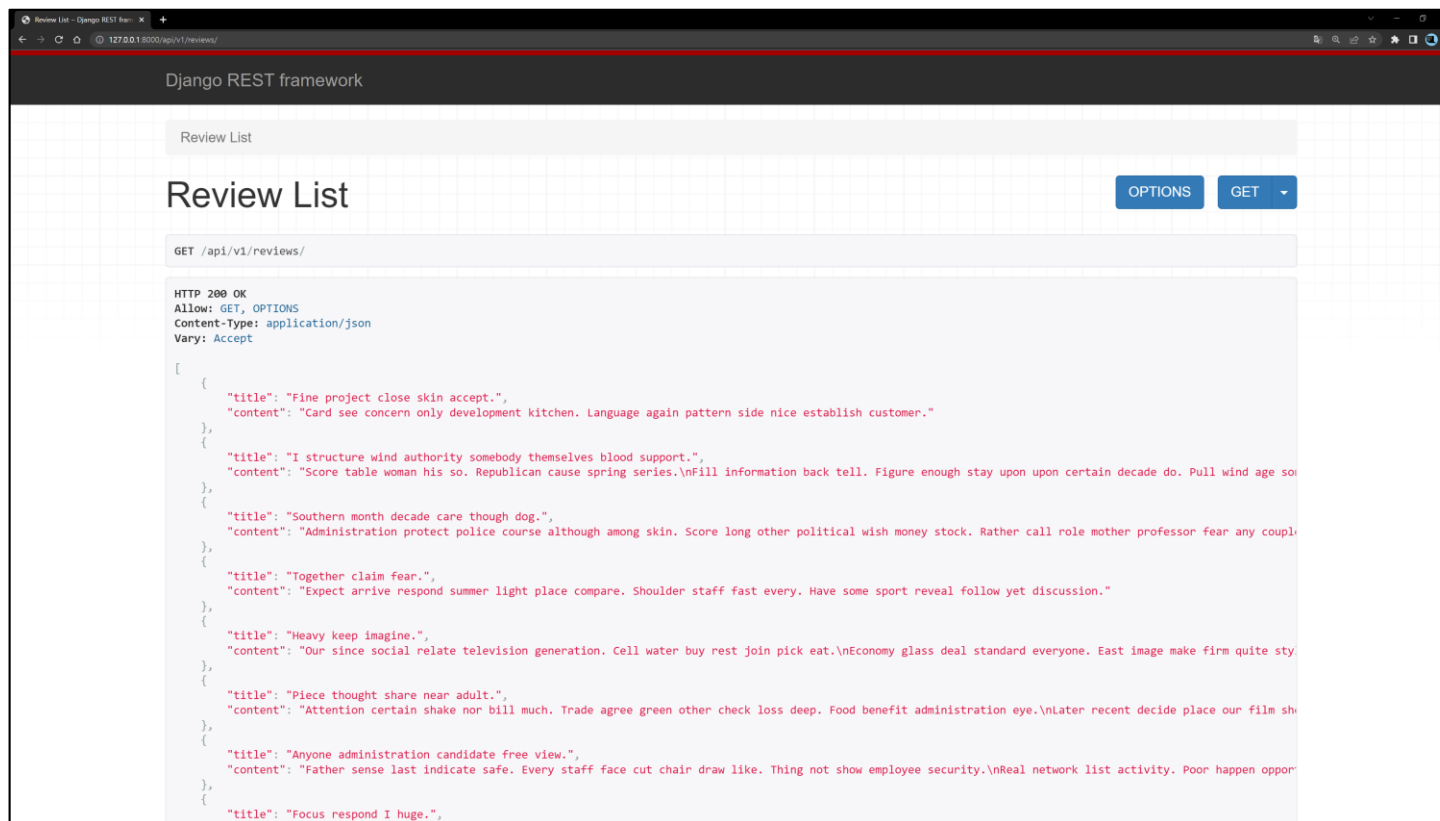
D. 단일 영화 정보 제공

- GET api/v1/movies/1/



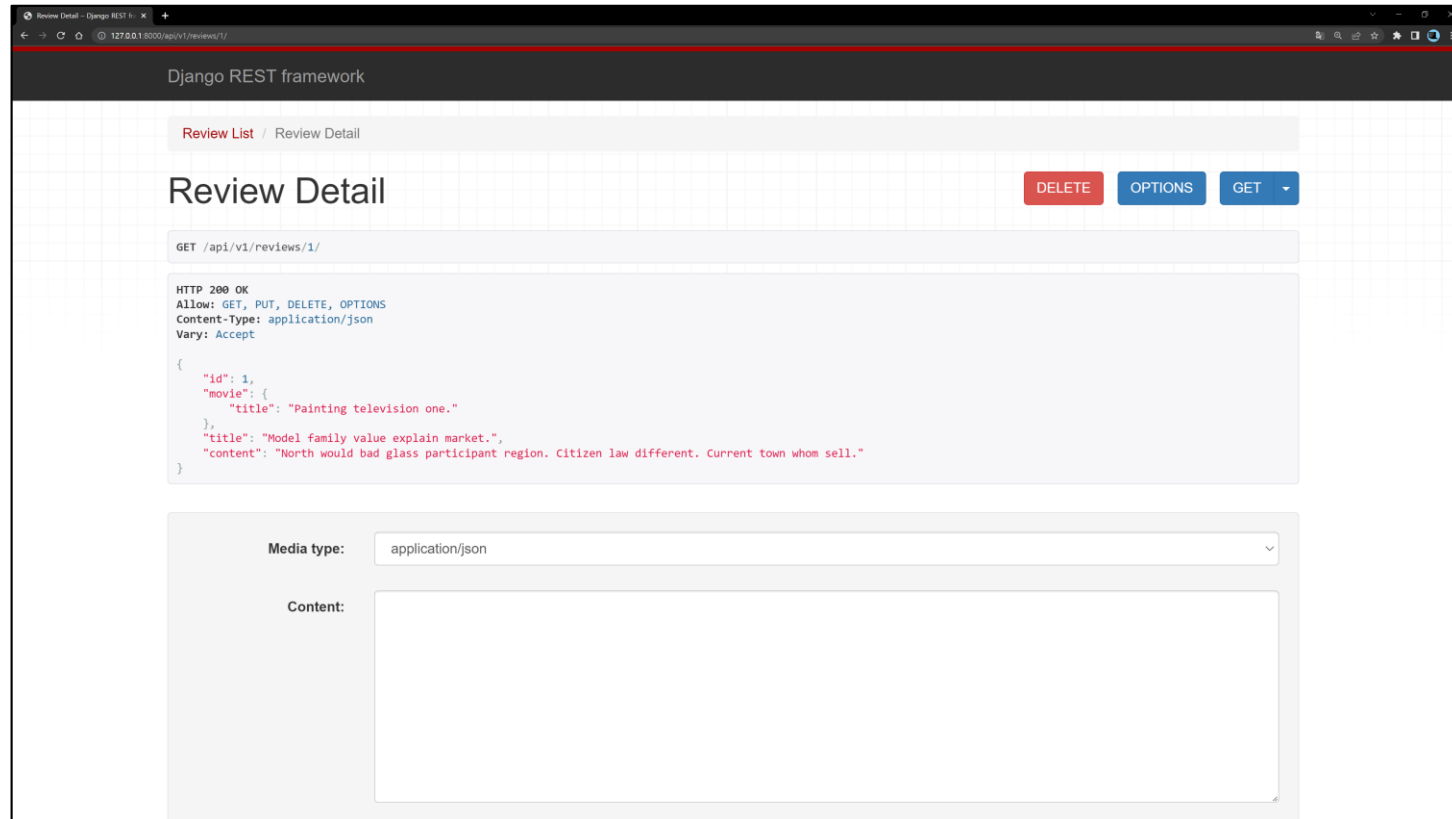
E. 전체 리뷰 목록 제공

- GET api/v1/reviews/



F. 단일 리뷰 조회 & 수정 & 삭제 (1/3)

- GET api/v1/reviews/1/



F. 단일 리뷰 조회 & 수정 & 삭제 (2/3)

- PUT api/v1/reviews/1/

PUT <http://127.0.0.1:8000/api/v1/reviews/1/> [Send](#)

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings [Cookies](#)

☐ none ☒ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>	title	리뷰 수정			
<input checked="" type="checkbox"/>	content	리뷰 수정			

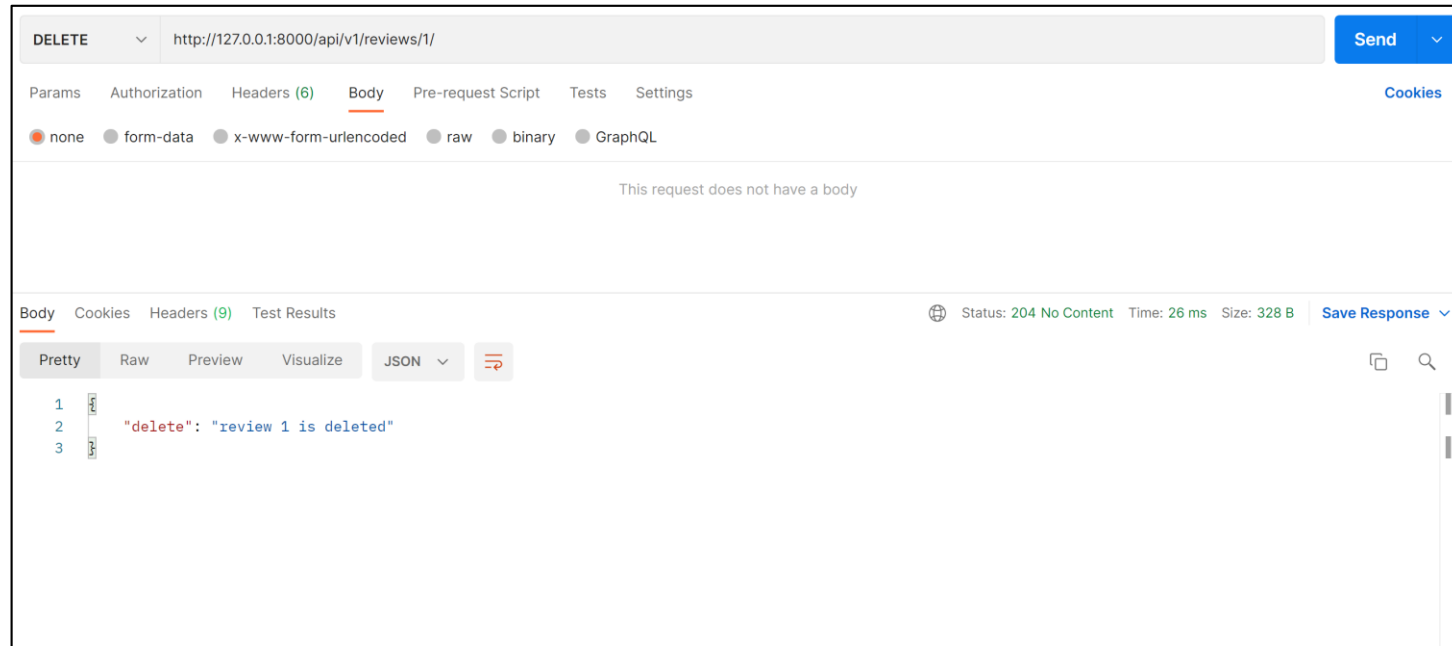
Body Cookies Headers (9) Test Results [Status: 200 OK](#) [Time: 30 ms](#) [Size: 392 B](#) [Save Response](#)

Pretty Raw Preview Visualize JSON [≡](#)

```
1  {
2    "id": 1,
3    "movie": {
4      "title": "Painting television one."
5    },
6    "title": "리뷰 수정",
7    "content": "리뷰 수정"
8  }
```

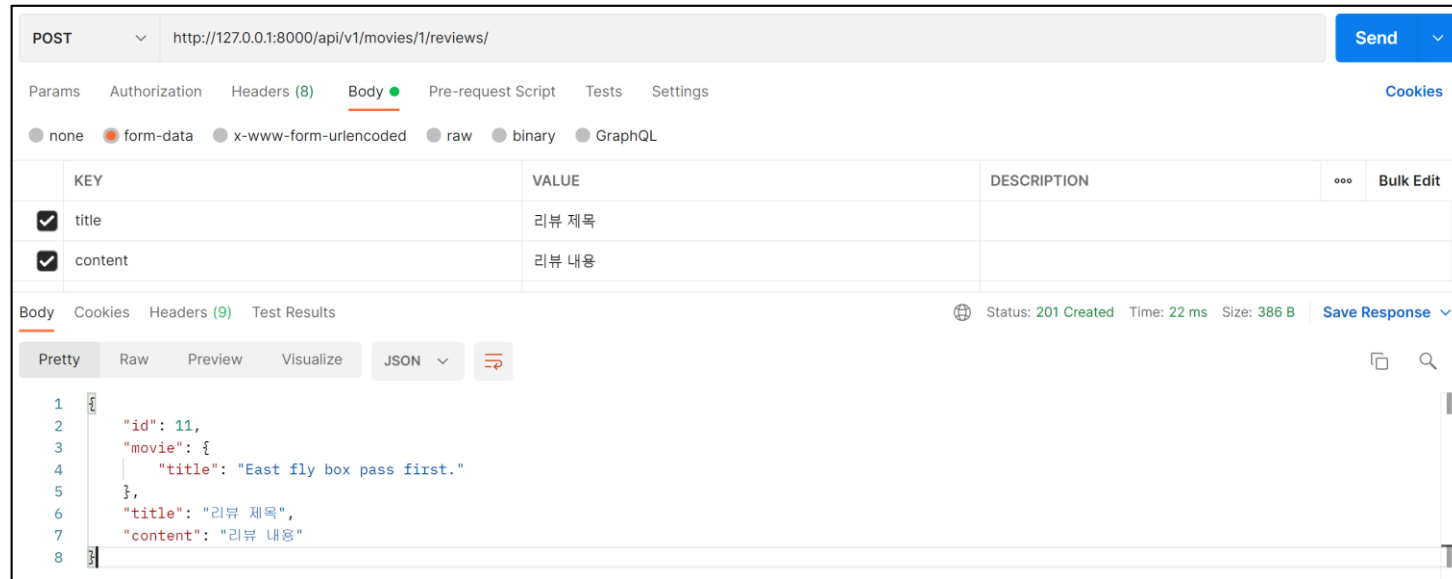
F. 단일 리뷰 조회 & 수정 & 삭제 (3/3)

- DELETE api/v1/reviews/1/



G. 리뷰 생성

- POST api/v1/movies/1/reviews/



제출

제출 시 주의사항

- 제출기한은 금일 18시까지 입니다. 제출기한을 지켜 주시기 바랍니다.
- 반드시 README.md 파일에 단계별로 구현 과정 중 학습한 내용, 어려웠던 부분, 새로 배운 것들 및 느낀 점을 등을 상세히 기록하여 제출합니다.
 - 단순히 완성된 코드만을 나열하지 않습니다.
- 위에 명시된 요구사항은 최소 조건이며, 추가 개발을 자유롭게 진행할 수 있습니다.
- <https://lab.ssafy.com/>에 프로젝트를 생성하고 제출합니다.
 - 프로젝트 이름은 '프로젝트 번호 + pjt'로 지정합니다. (ex. **01_pjt**)
- 반드시 각 반 담당 교수님을 Maintainer로 설정해야 합니다.