PJT명	**DB 설계를 활용한 REST API 설계**	
단계	[Django PJT]	
진행일자	2025.08.22	
예상 구현 시간	필수기능	5H
	추가기능	3H

1. 목표

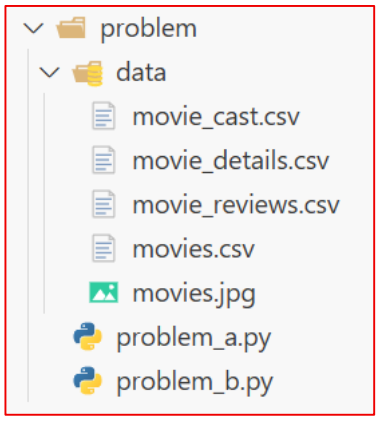
- Django REST framework(DRF)를 활용하여 RESTful API 서버를 제작할 수 있다.
- 데이터베이스 모델 관계(N:1, N:M)를 이해하고 설계에 적용할 수 있다.
- HTTP 요청 메서드와 상태 코드에 대한 이해를 바탕으로 API 기능을 구현할 수 있다.
- 영화, 배우, 리뷰, 장르 데이터를 조회하고, 리뷰 데이터를 생성, 수정, 삭제하는 API 서버를 구축한다.

2. 준비사항

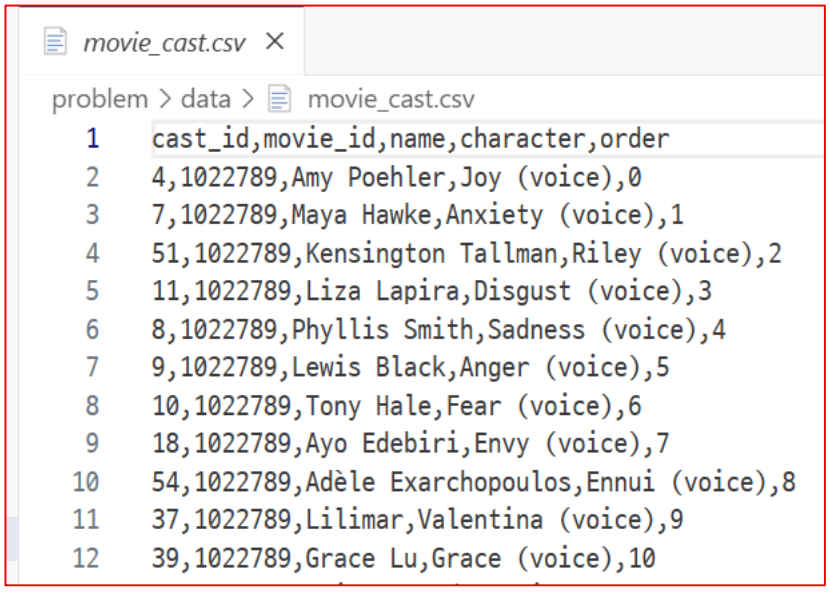
1) 프로젝트 구조

- 프로젝트는 문제 해결을 위한 스켈레톤 코드와 예시 코드를 포함하여 구성됩니다.
  1. problem 폴더
  2. examples 폴더

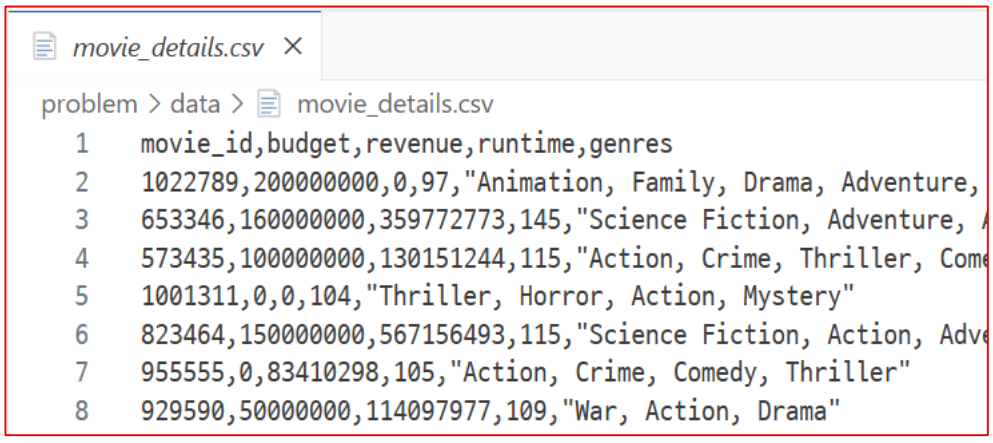
- problem 폴더
  - 요구사항 구현을 위한 스켈레톤 코드 파일 2개와 샘플 데이터 파일이 제공됩니다.



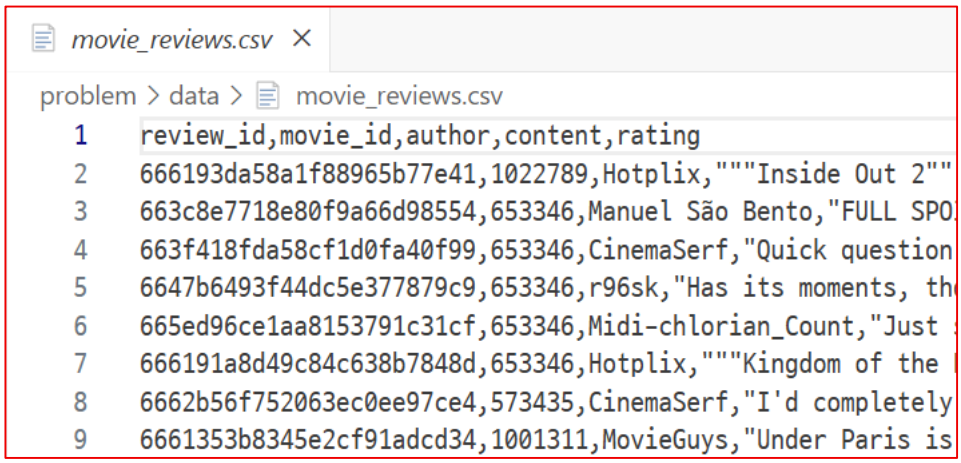
1. movie_cast.csv: csv 데이터 예시



2. movie_details.csv: csv 데이터 예시

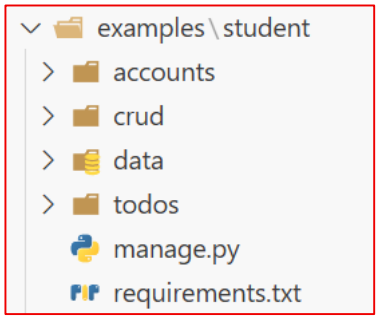


3. movie_reviews.csv: csv 데이터 예시



4. movies.csv: csv 데이터 예시

- examples 폴더
  - 프로젝트 해결에 도움이 될 수 있는 예시 코드가 포함되어 있습니다. 코드는 강의 시간에 함께 작성합니다.



1) 사용 데이터

- TMDB API: 영화 데이터 수집을 위해 TMDB(The Movie Database) API를 활용합니다.
- data 폴더 내 제공된 CSV 파일

2) 개발언어 및 툴

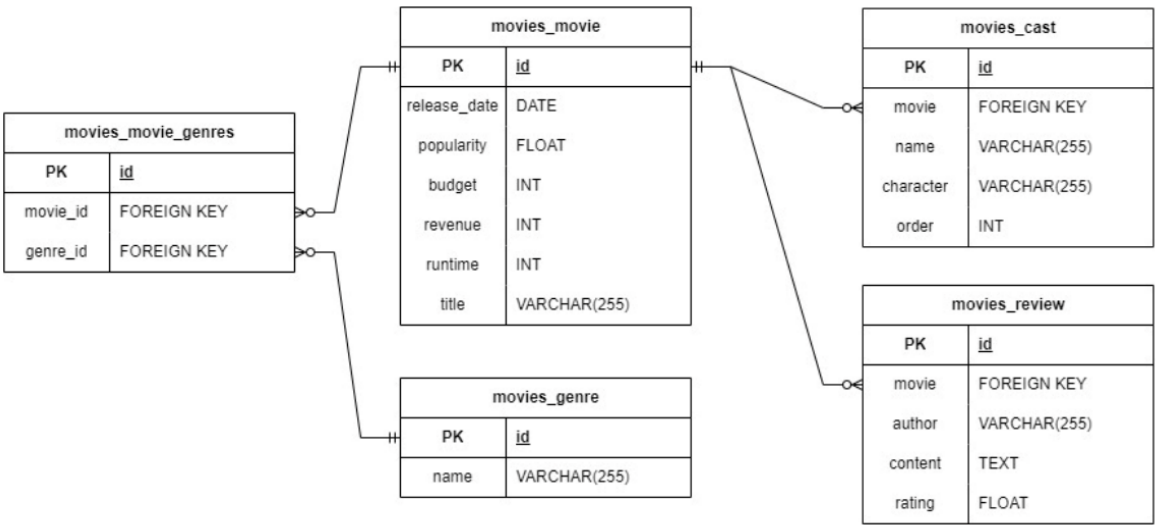
- Python 3.11+ / Visual Studio Code
- Django 5.2.x
- Django REST framework
- Visual Studio Code
- Postman

3) 필수 라이브러리 / 오픈소스

- Django
- Django REST framework

4) 모델 ERD 예시

- ERD 예시는 예시이므로, 반드시 완벽히 동일 해야하는 것은 아님



3. 작업 순서

1) 팀원과 같이 요구사항(필수/도전)을 확인하고, GitLab에 프로젝트를 생성합니다.
   - 프로젝트 이름은 `02-pjt`로 지정합니다.
   - 각 반 담당 강사님을 Maintainer로 설정합니다.
2) 제공된 `data` 폴더의 CSV 파일을 활용하여 DB 초기 데이터를 구축합니다.
3) movies 앱의 models.py에 명시된 모델과 관계를 정의하고 마이그레이션을 진행합니다.
4) 요구사항에 따라 urls.py와 views.py에 각 기능별 URL 엔드포인트와 로직을 구현합니다.
5) Postman을 사용하여 각 API가 요구사항에 맞게 올바르게 동작하는지 테스트합니다.
6) Postman을 사용하여 각 API가 요구사항에 맞게 올바르게 동작하는지 테스트합니다.
7) 작성한 코드들을 정리하고, README.md를 작성합니다.
8) README.md 작성이 완료되면 도전 과제를 진행합니다.
9) 제출 기한에 맞춰 모든 산출물이 GitLab에 업로드 될 수 있도록 합니다.

4. 요구사항

본격적인 영화 추천 커뮤니티 서비스 개발에 앞서, 서비스의 백엔드 API 서버를 구축하는 것을 목표로 합니다. 이를 위해 Django의 모델(Model) 기능을 활용하여 데이터베이스 스키마를 직접 설계하고, Django REST Framework(DRF)를 통해 클라이언트에게 데이터를 JSON 형식으로 제공하는 RESTful API를 구현해야 합니다.

개발자는 영화, 장르, 리뷰, 출연진 데이터를 관리하는 모델을 직접 정의하고, 각 데이터에 대한 조회(Read), 생성(Create), 수정(Update), 삭제(Delete)가 가능한 API 엔드포인트(Endpoint)를 구현하게 됩니다. 이 과정은 데이터베이스 설계와 서버 로직 구현을 직접 경험하며, 클라이언트-서버 구조의 핵심인 백엔드 시스템의 기초를 다지는 작업입니다.

팀원과 상의하여 아래 요구사항을 만족할 수 있도록 요구사항 명세서를 작성 및 구현합니다. 모든 서비스는 데이터를 저장하고 관리하는 서버가 필요하므로, 클라이언트의 요청에 응답하는 API 서버를 직접 설계하고 구축하는 연습을 해 봅시다.

- 요구사항 예시(참고용)
  - 아래의 내용을 참고하여 추가적인 아이디어에 대해 요구사항을 추가 또는 수정하여 기능을 구현한다. 단, **필수 기능은 반드시 구현해야 하며, 임의로 변경 할 수 없다**.

번호	분류	요구사항명	요구사항 상세	우선순위
기능적 요구사항				
F01	DB 모델링	모델링 및 관계 설정	영화, 배우, 리뷰, 장르 모델을 정의하고 N:1, N:M 관계를 올바르게 설정하는 기능	필수
F02	API 조회	전체 장르 목록 조회	`GET api/v1/genres/` 요청 시, 모든 장르 정보를 JSON 목록으로 반환하는 기능	필수
F03	API 조회	전체 영화 목록 조회	`GET api/v1/movies/` 요청 시, 모든 영화 정보를 JSON 목록으로 반환하는 기능	필수
F04	API 조회	단일 영화 상세 정보 조회	`GET api/v1/movies/<movie_pk>/` 요청 시, 특정 영화와 관련된 모든 정보를 반환하는 기능	필수
F05	API 조회	전체 리뷰 목록 조회	`GET api/v1/reviews/` 요청 시, 모든 리뷰 정보를 JSON 목록으로 반환하는 기능	필수
F06	API CUD	단일 리뷰 조회/수정/삭제	`api/v1/reviews/<review_pk>/` 엔드포인트에서 GET, PUT, PATCH, DELETE 메서드를 처리하는 기능	필수
F07	API 생성	특정 영화에 대한 리뷰 생성	`POST api/v1/movies/<movie_pk>/reviews/` 요청 시, 해당 영화에 대한 리뷰를 생성하는 기능	필수
F08	인증/권한	인증 및 권한 기능 구현	dj-rest-auth를 활용하여 회원가입/로그인 기능을 구현하고, 리뷰 작성 등 특정 기능에 대한 권한을 설정하는 기능	도전
F09	데이터 확장	영화 평점 통계 정보 추가	단일 영화 조회 시, 해당 영화의 리뷰들을 바탕으로 '평균 평점'과 '총 투표 수'를 계산하여 함께 제공하는 기능	도전

1) 기본(필수) 기능

A. 데이터베이스 모델링 및 관계 설정

데이터베이스의 기반이 될 각 모델의 필드와 모델 간의 관계를 설정합니다. 이 단계는 API 구현의 가장 기초가 됩니다.

- 요구사항 번호: F01
- 핵심 관계:
  - Movie - Cast (1:N): 하나의 영화는 여러 명의 출연진을 가질 수 있습니다. Cast 모델이 Movie의 외래 키를 가집니다.
  - Movie - Review (1:N): 하나의 영화는 여러 개의 리뷰를 가질 수 있습니다. Review 모델이 Movie의 외래 키를 가집니다.
  - Movie - Genre (M:N): 하나의 영화는 여러 장르에 속할 수 있고, 하나의 장르는 여러 영화를 포함할 수 있습니다. Django의 ManyToManyField를 사용하여 구현합니다.
- Movie 모델 명세

Column name	Type	Description
id	IntegerField (Primary Key)	영화 고유 ID
title	CharField	영화 제목
release_date	DateField	개봉일
popularity	FloatField	인기도
budget	IntegerField	예산
revenue	IntegerField	수익
runtime	IntegerField	상영 시간(분)

- Cast 모델 명세

Column name	Type	Description
id	IntegerField (Primary Key)	출연진 고유 ID
movie	ForeignKey to Movie	출연 영화 (Movie 참조)
name	CharField	배우 이름
character	CharField	배역 이름
order	IntegerField	출연 순서

- Review 모델 명세

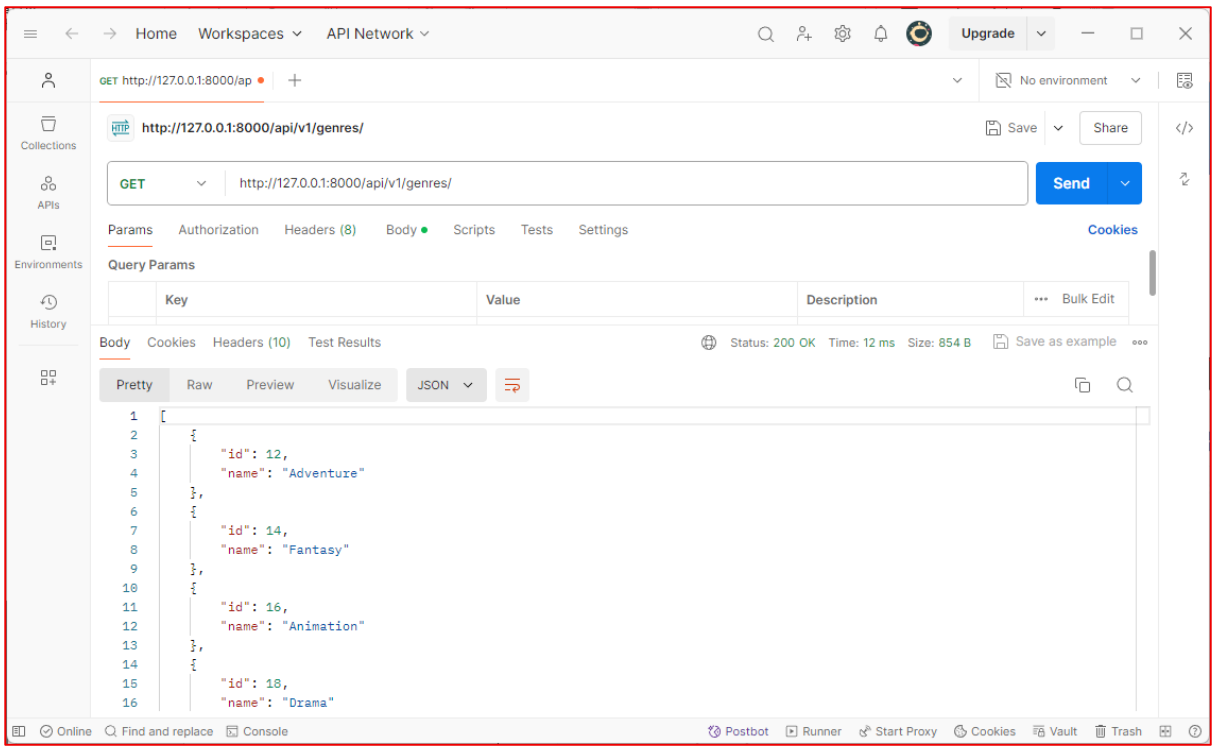
Column name	Type	Description
id	IntegerField (Primary Key)	리뷰 고유 ID
movie	ForeignKey to Movie	리뷰 대상 영화 (Movie 참조)
author	CharField	작성자
content	TextField	리뷰 내용
rating	FloatField	평점

B. 전체 장르 목록 조회

데이터 조회를 위한 각 GET 엔드포인트를 구현합니다. 각 API는 지정된 형식의 JSON 데이터를 반환해야 합니다.

DB에 저장된 모든 장르의 목록을 반환합니다.

- 요구사항 번호: F02
- 사전 준비:
  - genre 데이터 데이터 수집
  - CSV 파일로 export
  - DB에 데이터 삽입

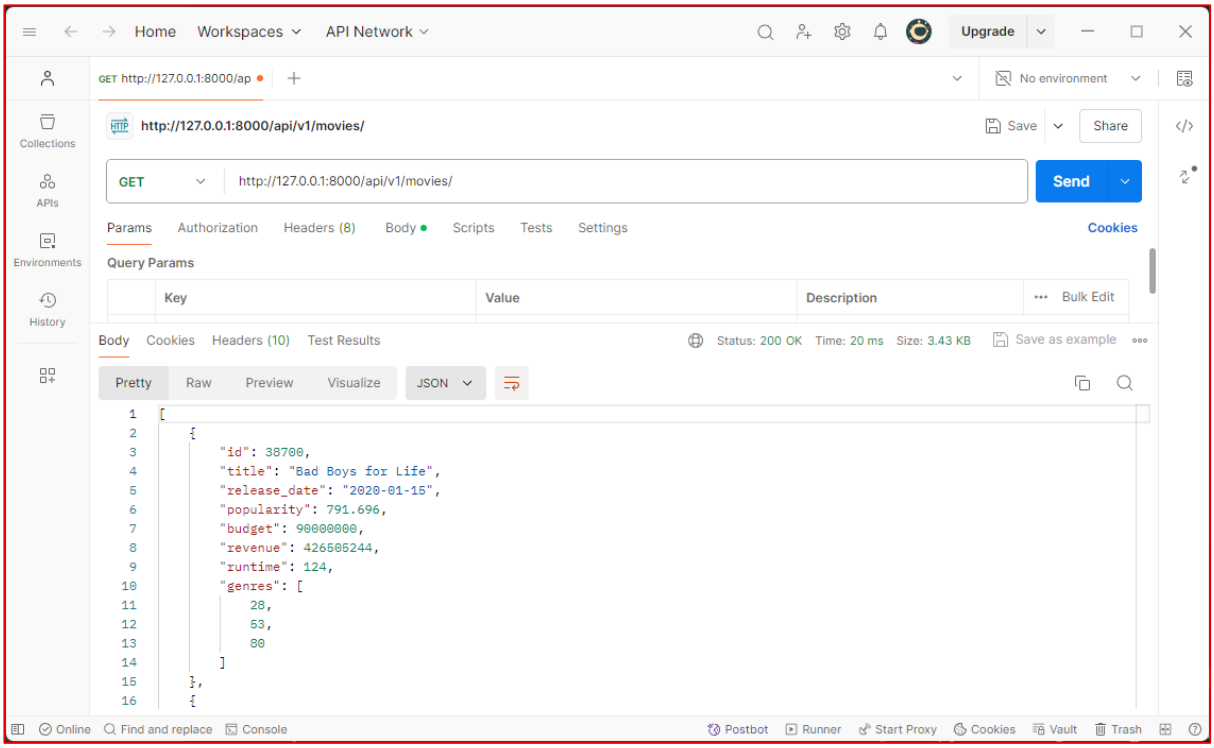


C. 전체 영화 목록 조회

데이터 조회를 위한 각 GET 엔드포인트를 구현합니다. 각 API는 지정된 형식의 JSON 데이터를 반환해야 합니다.

DB에 저장된 모든 영화의 목록을 반환합니다. 각 영화 객체에는 해당 영화의 장르 id 목록이 포함되어야 합니다.

- 요구사항 번호: F03
- 사전 준비:
  - movies.csv와 movie_details.csv 데이터 통합
  - genre 정보는 M:N 관계임에 유의

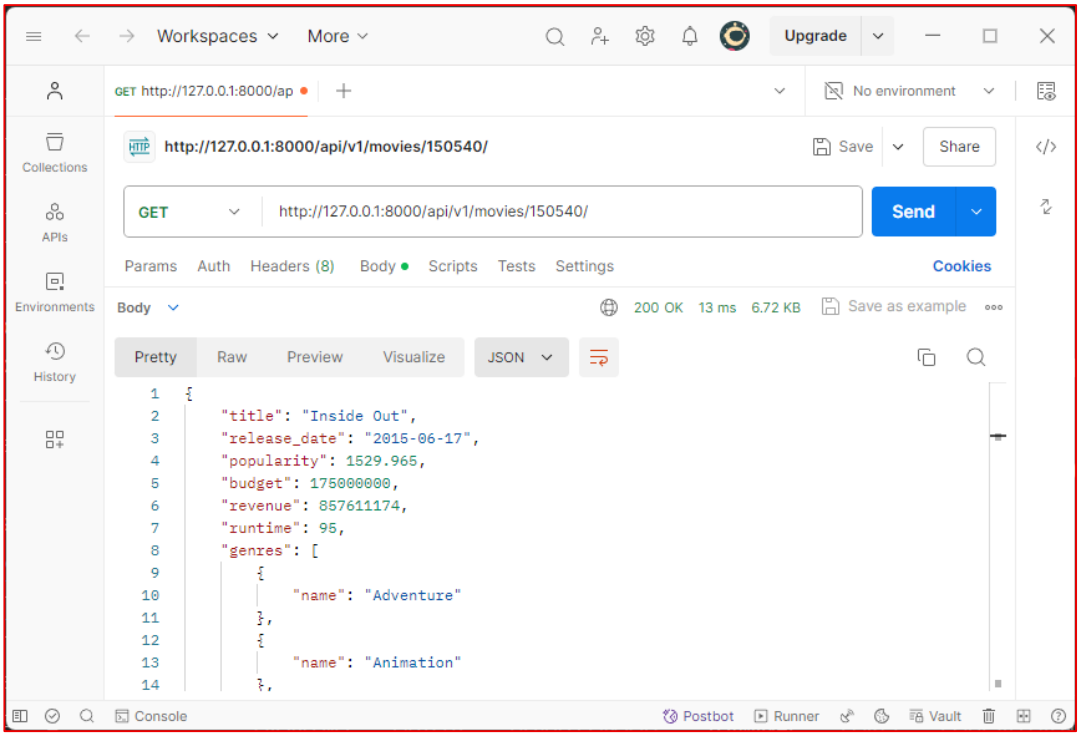


D. 단일 영화 상세 정보 조회

데이터 조회를 위한 각 GET 엔드포인트를 구현합니다. 각 API는 지정된 형식의 JSON 데이터를 반환해야 합니다.

특정 영화(movie_pk)의 모든 상세 정보와, 해당 영화에 종속된 출연진(cast_set), 리뷰(review_set), 장르(genres) 정보를 함께 반환합니다.

- 요구사항 번호: F04
- 사전 준비:
  - movie 상세 조회시 Movie를 참조 중인 모든 데이터를 반환하여야 함
    A. movie를 참조중인 Cast, Review의 모든 필드
    B. M:N 관계를 맺고 있는 Genre의 name
  - genre 정보는 M:N 관계임에 유의



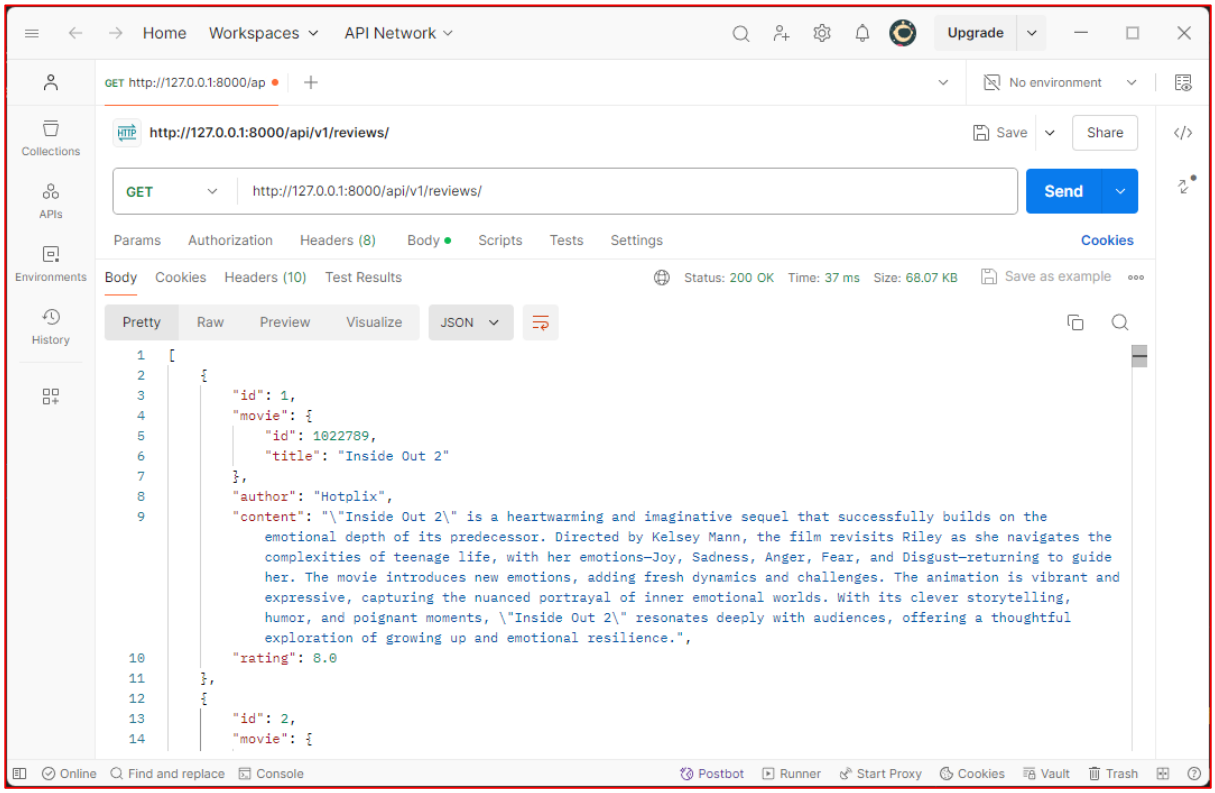

History
Pretty
Raw
Preview
Visualize
JSON
{
"name": "Family"
}
],
"cast_set": [
{
"name": "Richard Kind",
"character": "Bing Bong (voice)",
"order": 5
}
],
"review_set": [
{
"author": "Fatota",
"content": "This is the most incredible movie I've ever seen :)",
"rating": 10.0
},
{
"author": "Andres Gomez",
"content": "Another great movie from Pixar. The story in entangling and is structured in a master way to show us in a nice recreation how the mind works and emotions like sadness are important for a healthy life. A must to be seen.",
"rating": 8.0
},
{
Console
Postbot
Runner
Vault

E. 전체 리뷰 목록 조회

데이터 조회를 위한 각 GET 엔드포인트를 구현합니다. 각 API는 지정된 형식의 JSON 데이터를 반환해야 합니다.

모든 리뷰 목록을 반환합니다. 각 리뷰 객체에는 리뷰가 달린 영화의 id와 title이 포함되어야 합니다.

- 요구사항 번호: F05
- 사전 준비:
  - 전체 리뷰 목록 조회시 review의 모든 정보를 제공하여야 함
  - 단, 참조중인 movie의 경우, id와 title만 제공되어야 함



F. 단일 리뷰 조회/수정/삭제

[GET]: 특정 리뷰(review_pk)의 상세 정보를 반환합니다.

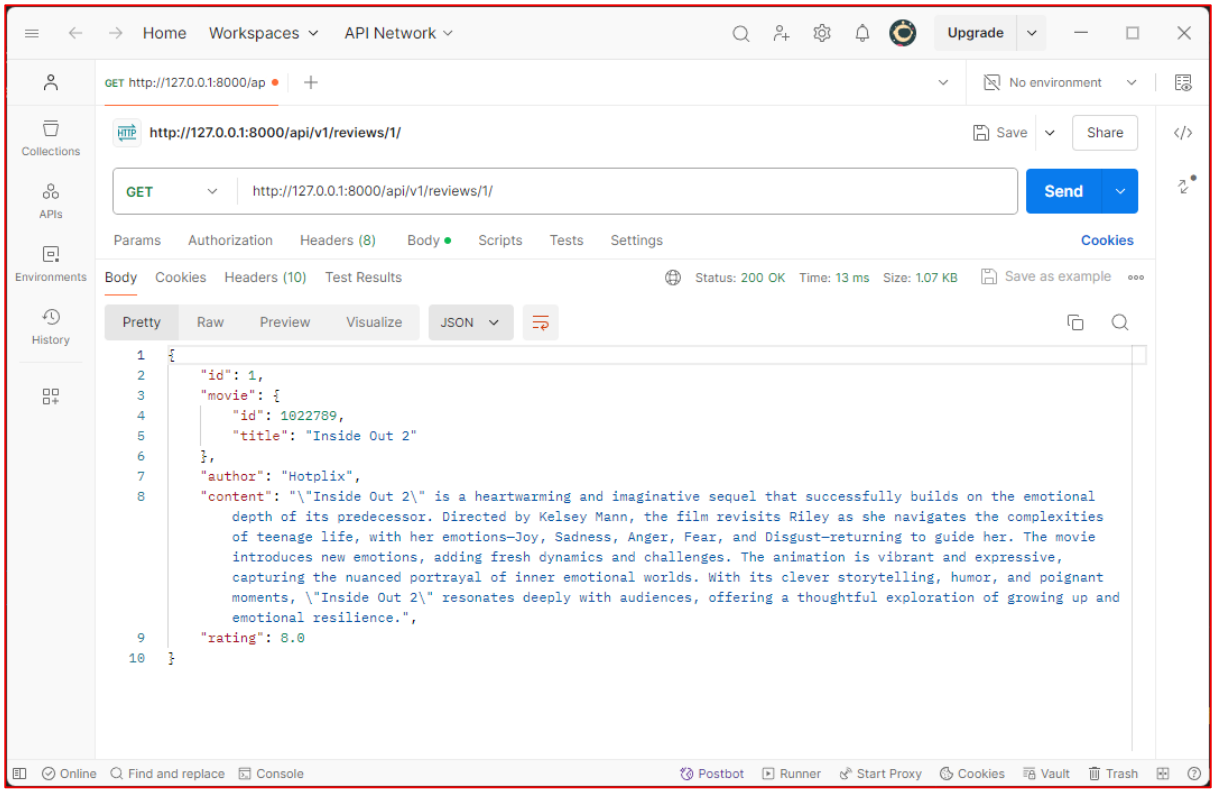
[PUT]: 특정 리뷰의 모든 필드(author, content, rating)를 요청받은 값으로 교체합니다. 일부 필드만 보내면 오류가 발생해야 합니다.

[PATCH]: 특정 리뷰의 일부 필드만 요청받은 값으로 수정합니다.

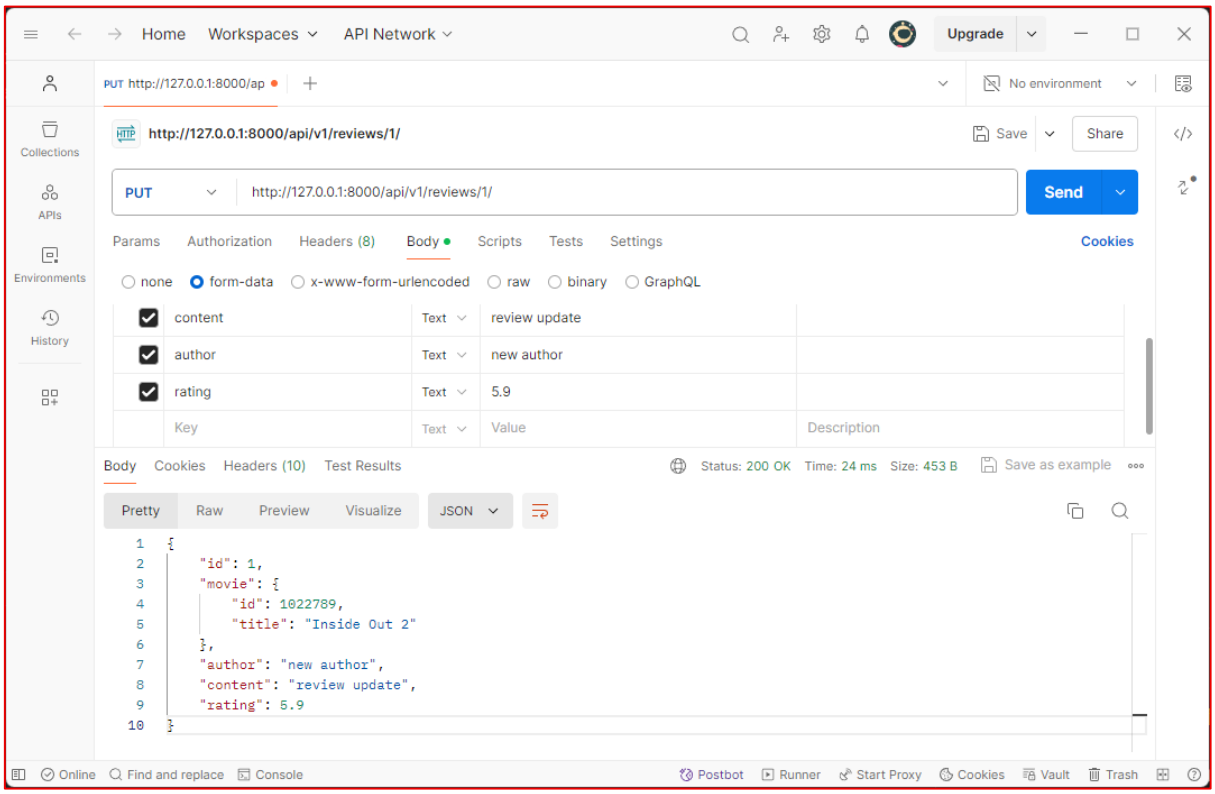
[DELETE]: 특정 리뷰를 DB에서 삭제합니다.

- 요구사항 번호: F06
- 사전 준비:
  - 단일 리뷰 조회
    A. 단일 리뷰 조회시 review의 모든 정보를 제공하여야 함

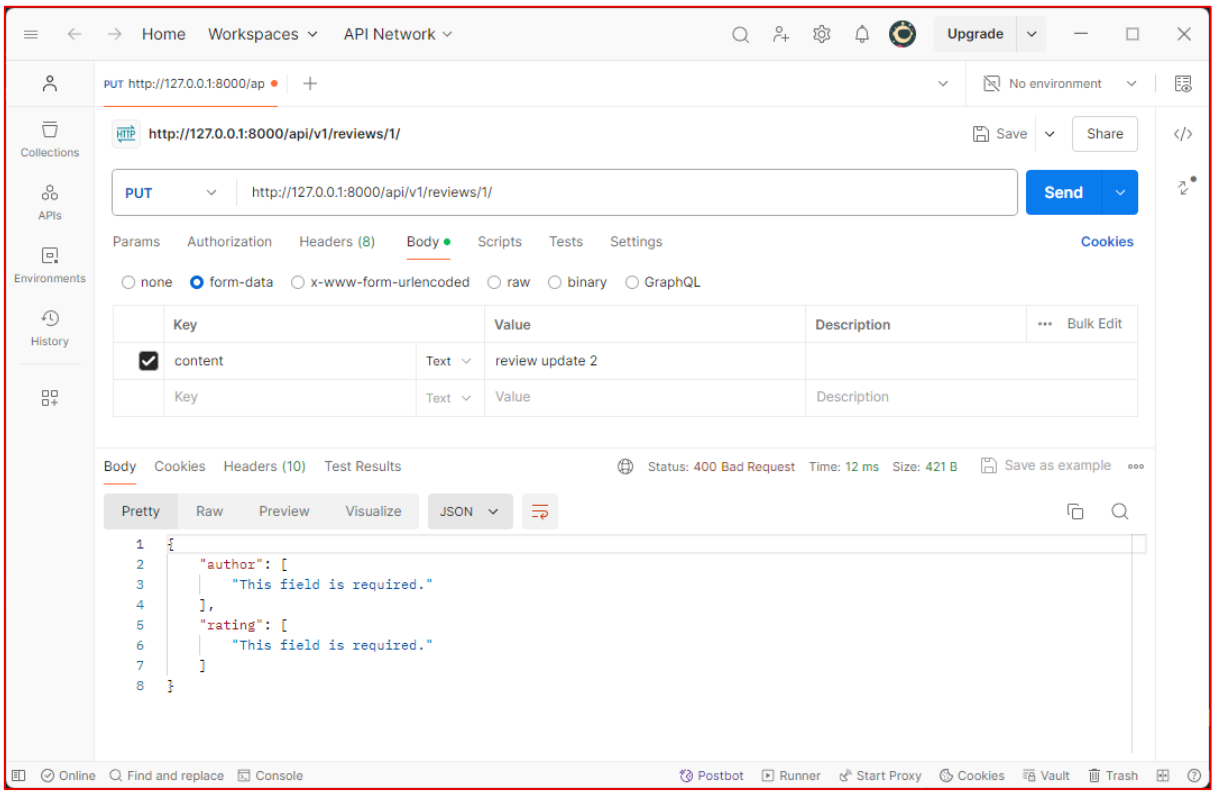
    B. 단, 참조중인 movie의 경우, id와 title만 제공되어야 함



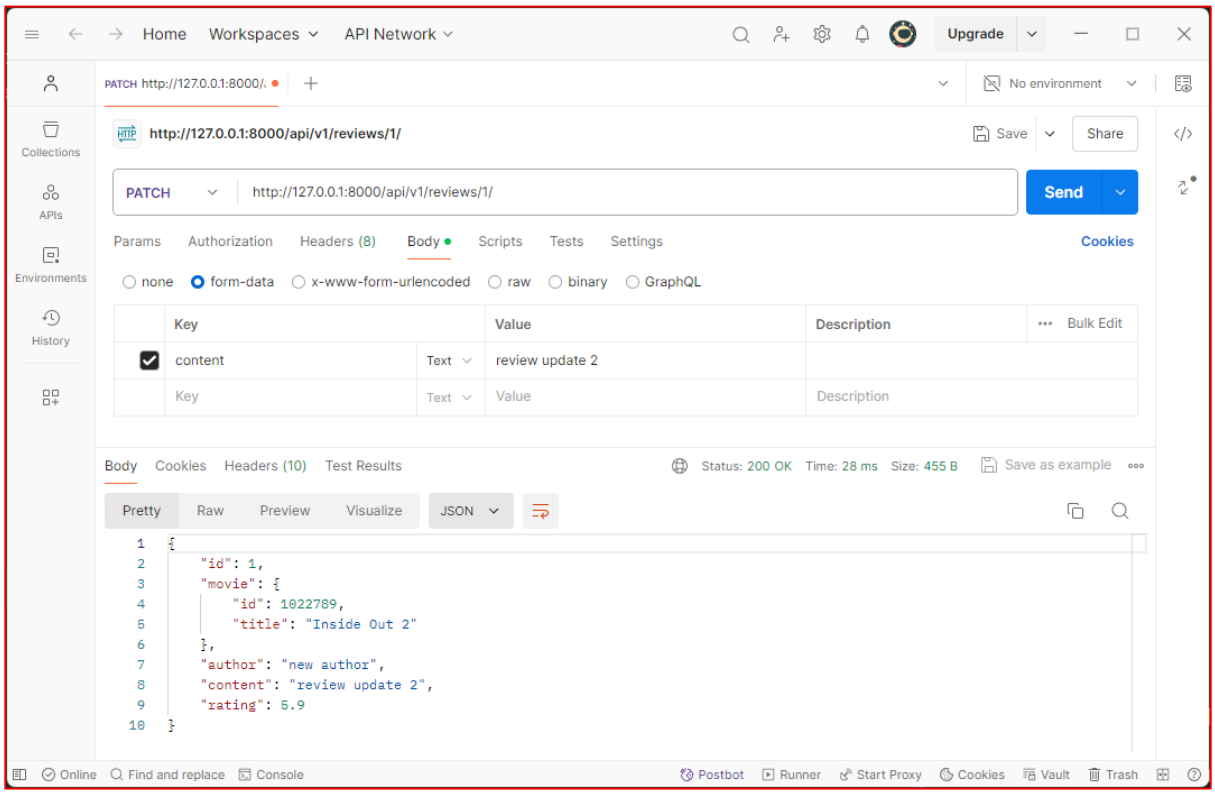
- 단일 리뷰 수정 – 전체 필드



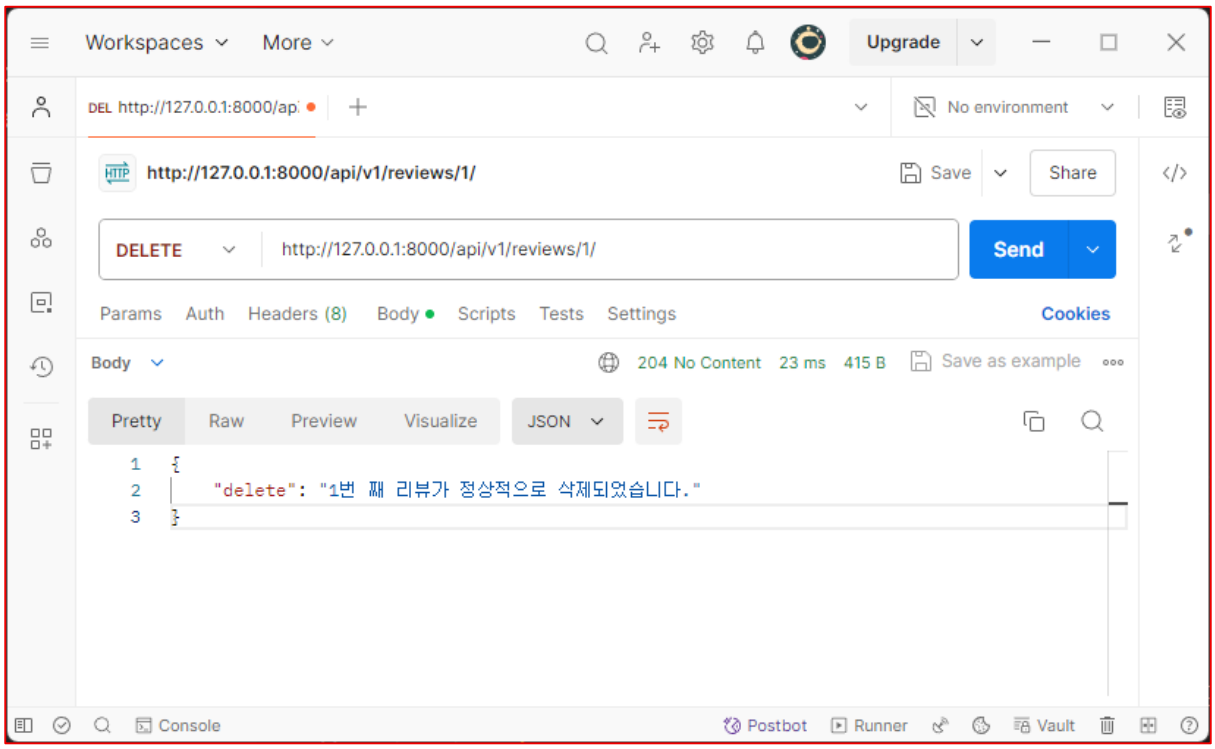
- 단일 리뷰 수정 – 일부 필드 실패



- 단일 리뷰 수정 – 일부 필드 성공



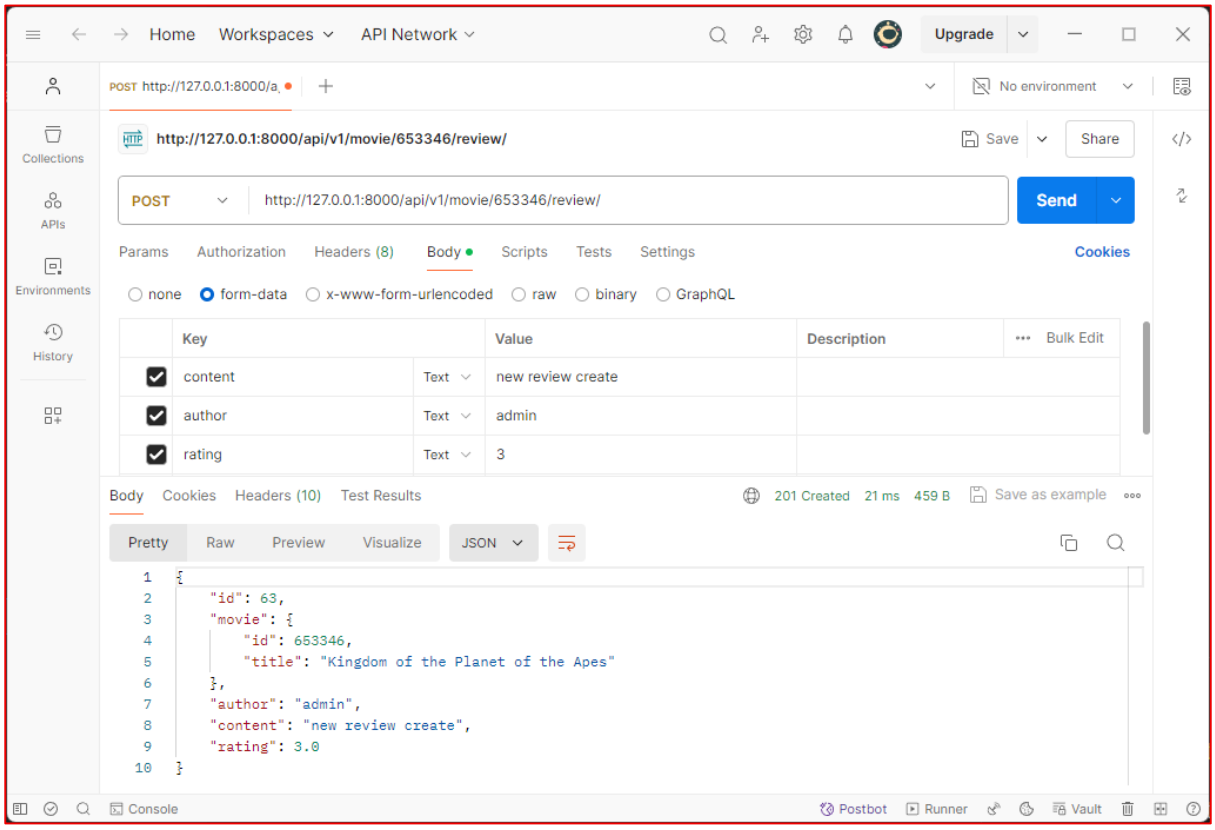
- 단일 리뷰 삭제



G. 특정 영화에 대한 리뷰 생성

movie_pk에 해당하는 영화에 새로운 리뷰를 생성합니다.

- 요구사항 번호: F07



2) 도전 과제

기본 기능 구현 후, 수집한 데이터를 바탕으로 다음 두 도전과제 요구사항을 해결합니다.

A. 인증 및 권한 기능 구현

사용자 관리 시스템을 도입하여 API에 인증(Authentication) 및 권한(Authorization) 기능을 추가합니다. 이를 통해 특정 기능을 보호하고 사용자별로 데이터를 관리할 수 있게 됩니다.

- 요구사항 번호: F08
- 구현 위치: accounts 앱 생성, dj-rest-auth 라이브러리 설정, settings.py 수정, Review 모델 및 Serializer 수정
- **핵심 기능:**
  - **회원가입/로그인:** dj-rest-auth를 사용하여 /accounts/signup/ 및 /accounts/login/ 엔드포인트를 구현합니다. 로그인 성공 시 인증 토큰(Token)이 발급되어야 합니다.
  - **인증 적용:** 회원가입/로그인을 제외한 모든 API 엔드포인트는 인증된 사용자(토큰을 가진 사용자)만 접근할 수 있도록 권한을 설정합니다.
  - **모델 관계 변경:** Review 모델의 기존 author(문자열) 필드를 User 모델을 참조하는 ForeignKey로 교체하여, 어떤 유저가 리뷰를 작성했는지 명확히 연결합니다.
  - **권한 설정:** 리뷰의 수정/삭제는 해당 리뷰를 작성한 사용자 본인만이 수행할 수 있도록 권한을 제한합니다.

B. 영화 평점 통계 정보 추가

단일 영화 상세 정보 조회 시, 해당 영화에 연결된 모든 리뷰 데이터를 동적으로 집계하여 평균 평점과 총 투표 수 정보를 추가로 제공합니다.

- 요구사항 번호: F09
- 구현 위치: movies/serializers.py 또는 movies/views.py의 단일 영화 조회 로직 수정

- **핵심 기능:**
  - **데이터 집계:** 특정 영화(movie_pk)에 연결된 모든 Review 객체들의 rating 필드 값을 가져옵니다.
  - **평균 평점 계산:** 가져온 모든 rating 값의 평균을 계산합니다. (Django ORM의 Avg 집계 함수 사용 권장)
  - **총 투표 수 계산:** rating 값이 있는 Review 객체의 총 개수를 계산합니다. (Django ORM의 Count 집계 함수 사용 권장)
  - **Serializer 필드 추가:** SerializerMethodField 등을 활용하여 average_rating과 vote_count라는 두 개의 새로운 필드를 Serializer에 추가하고, 위에서 계산된 값을 반환하도록 구현합니다.

5. 참고자료

- Django 공식 문서 (4.2)
  - 프로젝트의 기반이 되는 Django 프레임워크의 공식 문서입니다. 모델, ORM, 뷰 등 핵심 개념을 확인할 수 있습니다.
  - https://docs.djangoproject.com/en/4.2/
- Django REST Framework 공식 문서
  - DRF의 Serializer, View, 인증, 권한 등 API 서버 구축에 필요한 모든 기능의 사용법을 상세히 다룹니다.
  - https://www.django-rest-framework.org/
- dj-rest-auth 공식 문서
  - 도전 과제인 인증 기능 구현에 사용되는 dj-rest-auth 라이브러리의 공식 문서입니다.
  - https://dj-rest-auth.readthedocs.io/en/latest/
- MDN - REST 소개
  - RESTful API가 무엇인지, 그 구조와 원칙에 대해 이해하는 데 도움이 되는 가이드입니다.
  - https://developer.mozilla.org/ko/docs/Glossary/REST
- Postman Learning Center
  - 프로젝트 요구사항을 테스트하는 데 사용되는 API 클라이언트 도구인 Postman의 공식 사용법 안내서입니다.
  - https://learning.postman.com/docs/getting-started/introduction/

6. 결과

제출 기한은 진행일 18시까지이므로 제출 기한을 지킬 수 있도록 합니다. 제출은 GitLab을 통해서 이루어집니다.

- **산출물과 제출**
  - 단계별로 구현 과정 중 학습한 내용, 어려웠던 부분, 새로 배운 것들 및 느낀 점을 상세히 기록한 README.md
  - 완성된 각 문제 별 소스코드 및 실행 화면 캡쳐본
  - 프로젝트 이름은 02-pjt로 지정, 각자의 계정에 생성할 것
  - 각 반 담당 강사님을 Maintainer로 설정

- 끝 -