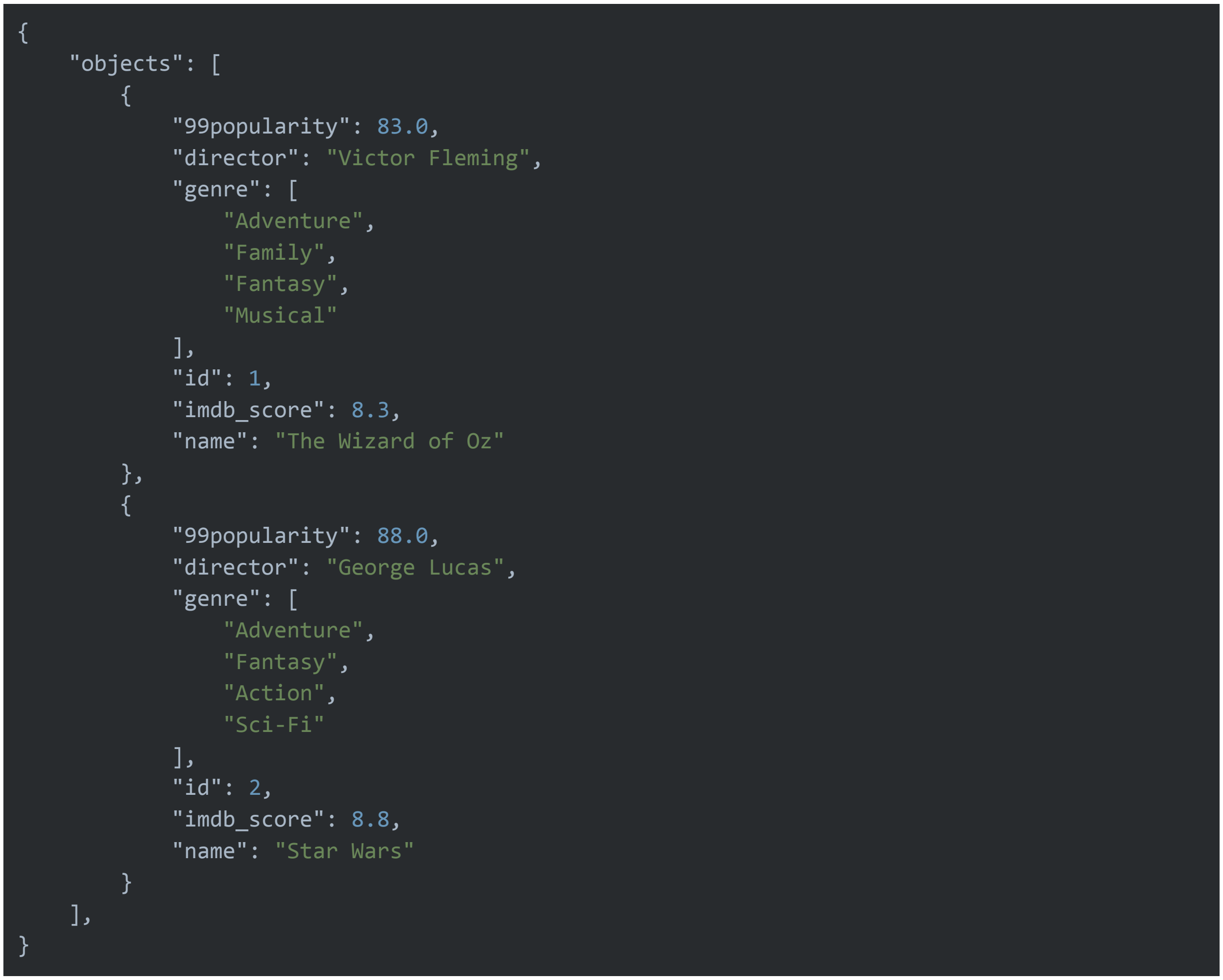
IMDB TASK

Task base URL: https://imdb-movies-api.herokuapp.com/
Python Framework: Flask

API endpoints:

1. List movies: anonymous users can get list of available movies
   endpoint : https://imdb-movies-api.herokuapp.com/movies

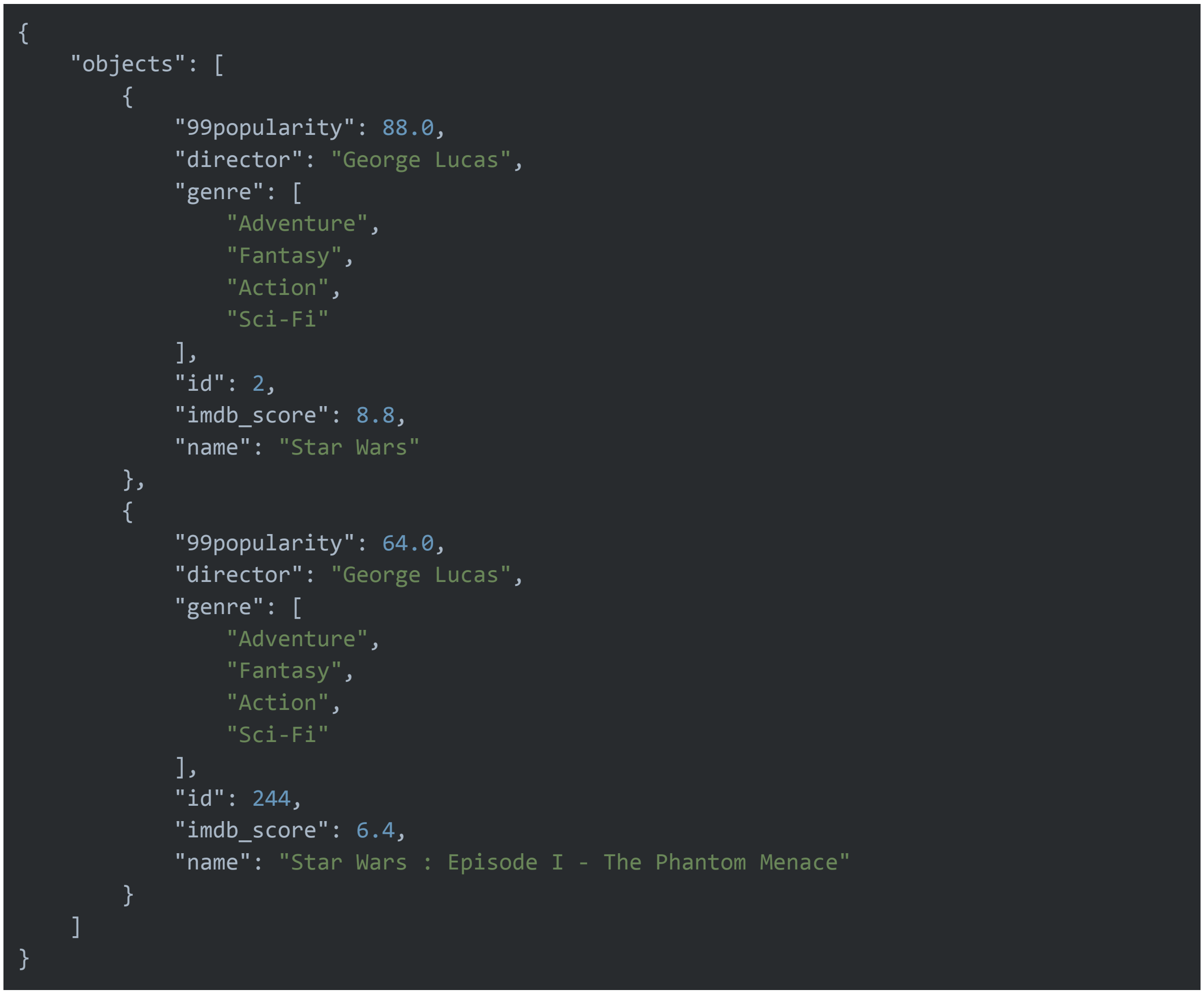
   Method : GET
   Response: 200 OK



```
{
    "objects": [
        {
            "99popularity": 83.0,
            "director": "Victor Fleming",
            "genre": [
                "Adventure",
                "Family",
                "Fantasy",
                "Musical"
            ],
            "id": 1,
            "imdb_score": 8.3,
            "name": "The Wizard of Oz"
        },
        {
            "99popularity": 88.0,
            "director": "George Lucas",
            "genre": [
                "Adventure",
                "Fantasy",
                "Action",
                "Sci-Fi"
            ],
            "id": 2,
            "imdb_score": 8.8,
            "name": "Star Wars"
        }
    ],
}
```

2. Search movie: anonymous users can search for movie
   endpoint : https://imdb-movies-api.herokuapp.com/movies?search=Star Wars

   Method: GET
   Response: 200 OK



```
{
    "objects": [
        {
            "99popularity": 88.0,
            "director": "George Lucas",
            "genre": [
                "Adventure",
                "Fantasy",
                "Action",
                "Sci-Fi"
            ],
            "id": 2,
            "imdb_score": 8.8,
            "name": "Star Wars"
        },
        {
            "99popularity": 64.0,
            "director": "George Lucas",
            "genre": [
                "Adventure",
                "Fantasy",
                "Action",
                "Sci-Fi"
            ],
            "id": 244,
            "imdb_score": 6.4,
            "name": "Star Wars : Episode I - The Phantom Menace"
        }
    ]
}
```

3. Admin Login:
   endpoint : https://imdb-movies-api.herokuapp.com/login

   Method : POST
   Request headers:
   - Content-Type: application/json

   Request body:

```
{
       "username": "davidXavier",
       "password": "imdb@1234"

}
```

Response: 200 OK

```
{
    "success": true,
    "token":
"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJleHAiOjE2MTYxNzUwOTMsImlhdCI6MTYxNjE3MTQ5Mywic3
ViIjoyfQ._ZCDqI6rQ_HVNIjtaCxqz-r6XnIx0OmuJ_Wi2E1jCEM",
    "token_expiry": "2021-03-19T17:31:33.021481"
}
```

Response contains `token`, send this token in `Authorization` header in each request for all admin following apis. token is valid for 1 hour

4. Add movie: only admin user can add new movie

   Endpoint: https://imdb-movies-api.herokuapp.com/movies/add

   Method: POST
   Request headers:

   - Authorization: token received while admin login

   Request body:

```
{
        "name": "My New Movie",
        "director": "David Xavier",
        "99popularity": 85.0,
        "imdb_score": 8.5,
        "genre": [
                "Action",
                "Sci-fi"
        ]
}
```

   Response: 200

5. Remove movie: only admin user can remove existing movie
   Endpoint : https://imdb-movies-api.herokuapp.com/movies/remove/<id>

   Requires <id> of movie from list movies endpoint

   Method: DELETE
   Request headers:

   - Authorization: token received while admin login

   Response: 200

6. Edit movie: only admin user can edit existing movie
   Endpoint : https://imdb-movies-api.herokuapp.com/movies/edit/<id>

   Requires <id> of movie from list movies endpoint
   Method: PATCH
   Request headers:
   - Authorization: token received while admin login

   Request body:

```
{
        "name": "My New Movie 2",
        "director": "David Xavier",
        "99popularity": 85.0,
        "imdb_score": 8.5,
        "genre": [
                "Action",
                "Sci-fi"
        ]
}
```

   Response: 200 OK

Database scaling:
- Most of the traffic would be towards read access to movie db table
- We can consider to add read replicas of db instance which can decrease read access latency time

Request time:
- In order to process large number of requests CPU resources also needs to be added
- Other thing can be done is to process API requests async like Sanic or FastAPI