Large-Scale and Multi-Structured Databases

MOVIELAND

Cecchini Leonardo

Protopapa Matteo

Ruisi Rosario

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

UNIVERSITÀ DI PISA

CROSSLAB Innovation for industry 4.0

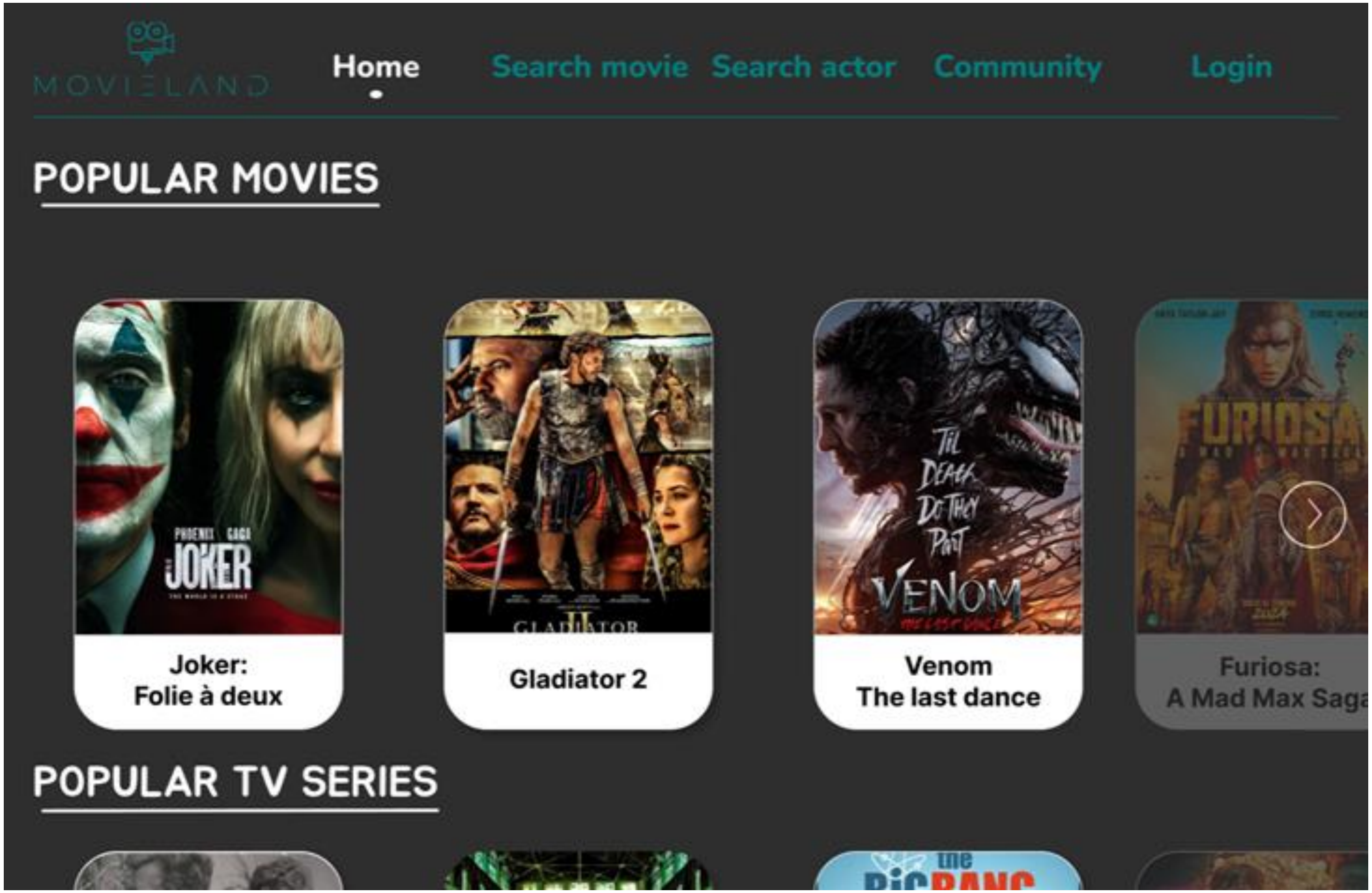
Application Highlights

MOVIELAND is a comprehensive platform for movies and TV series, enabling users to discover, review, and rate titles, connect with a community, and receive personalized recommendations. It offers features to manage profiles, favorites, watchlists, and perform advanced searches.

- **Personalized Profiles**: Enable users to save favorite genres, movies, and actors, or create a watchlist for future viewing.
- **Movie Reviews and Ratings**: Allow users to share their opinions on movies while providing managers with metrics to evaluate both user engagement and movies and actors performances.
- **User Interaction**: Introducing posts and comments to create discussions and connections around movies.
- **Personalized Recommendations:** Suggest movies, actors, and users with similar tastes based on preferences and ratings.
- **Advanced Search and Filters**: Facilitate precise searches by title, genre, and additional criteria.

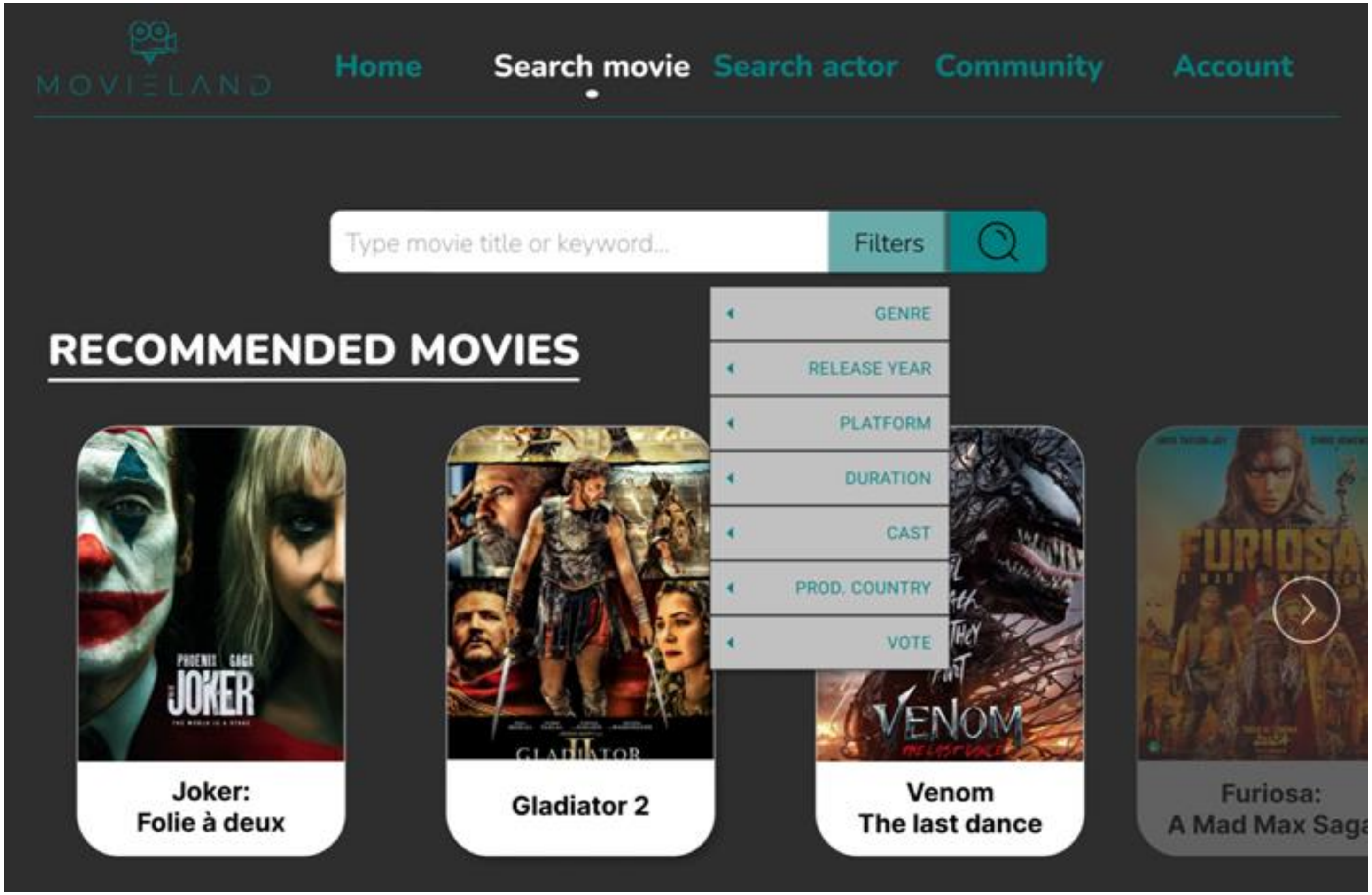
Actors (i)

UNREGISTERED USERS



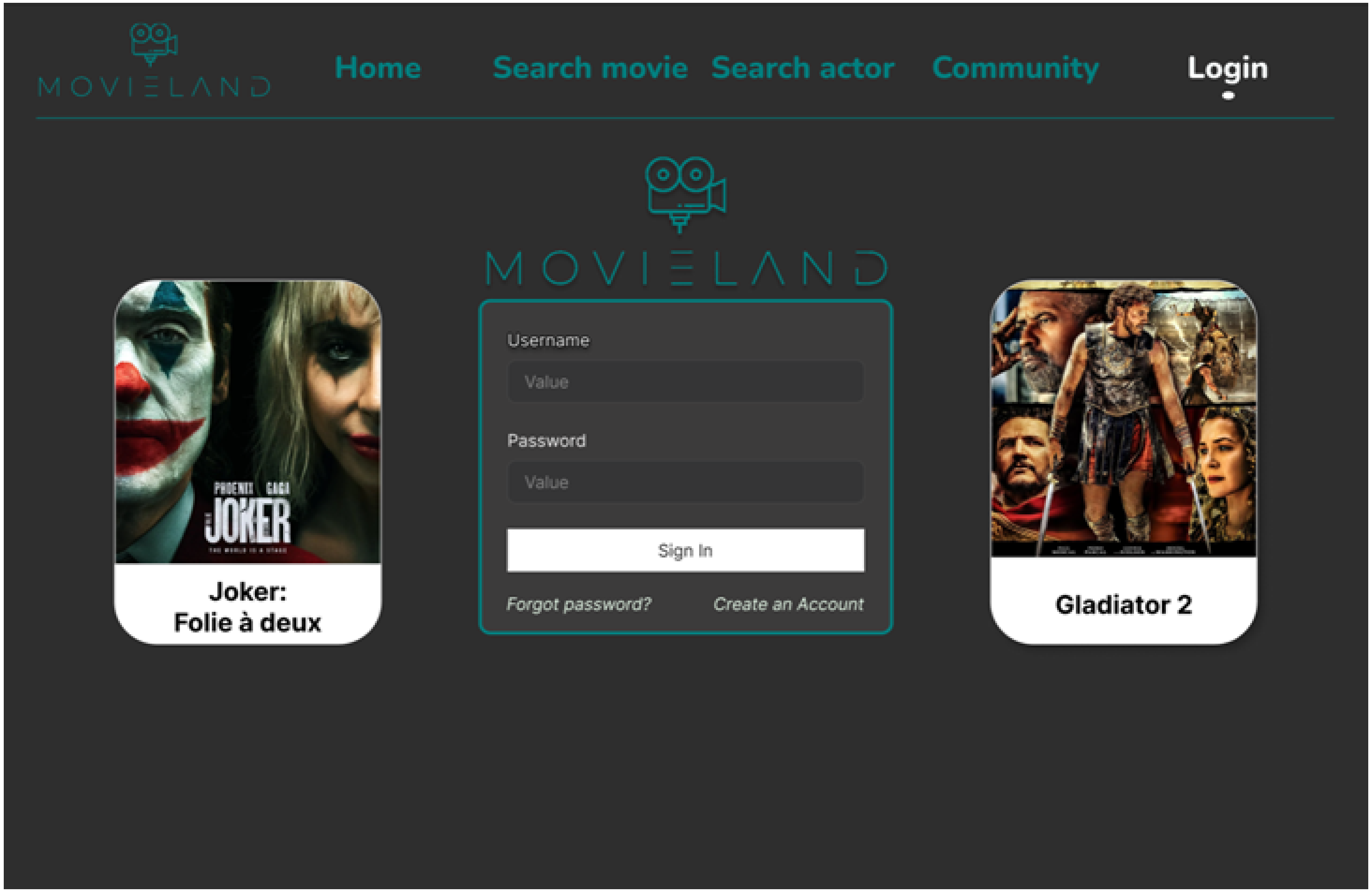
Actors (ii)

UNREGISTERED USERS



Actors (iii)

REGISTERED USERS



Username

Value

Password

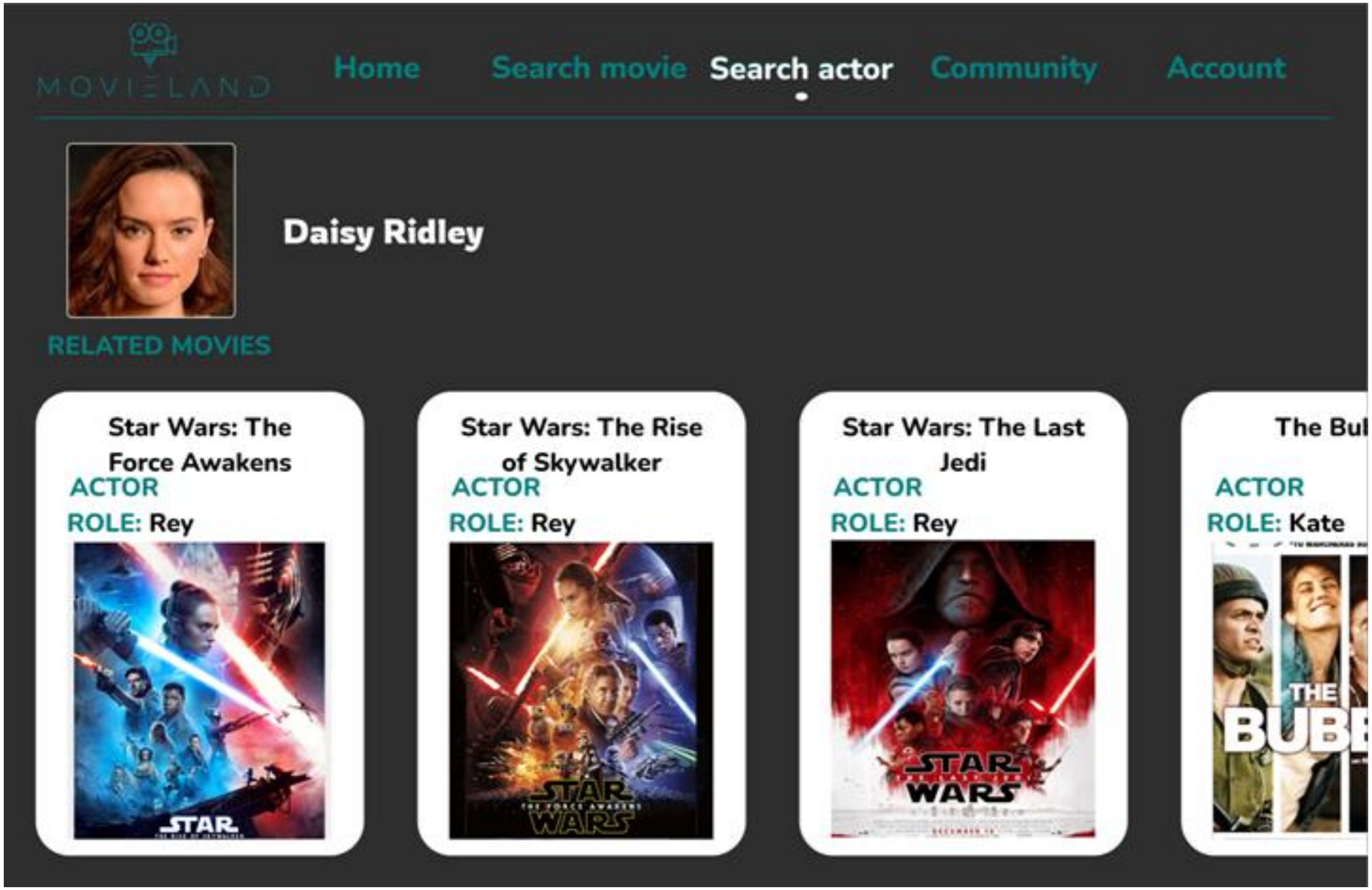
Value

Sign In

Forgot password? *Create an Account*

Actors (iv)

UNREGISTERED USERS



Daisy Ridley

RELATED MOVIES

Star Wars: The Force Awakens
ACTOR
ROLE: Rey

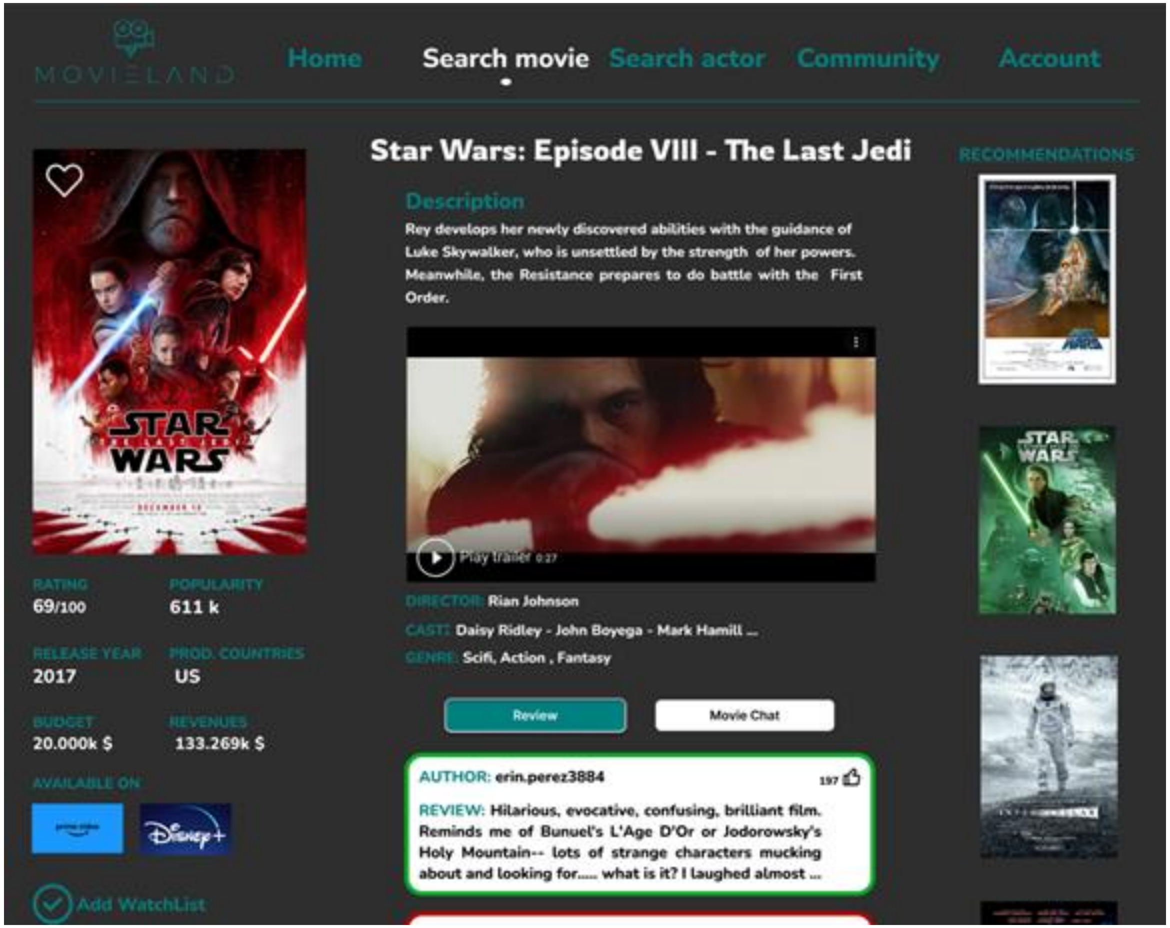
Star Wars: The Rise of Skywalker
ACTOR
ROLE: Rey

Star Wars: The Last Jedi
ACTOR
ROLE: Rey

The Bu
ACTOR
ROLE: Kate

Actors (v)

UNREGISTERED USERS



Star Wars: Episode VIII - The Last Jedi

Description

Rey develops her newly discovered abilities with the guidance of Luke Skywalker, who is unsettled by the strength of her powers. Meanwhile, the Resistance prepares to do battle with the First Order.

RATING 69/100

POPULARITY 611 k

RELEASE YEAR 2017

PROD. COUNTRIES US

BUDGET 20.000k $

REVENUES 133.269k $

AVAILABLE ON

DIRECTOR: Rian Johnson

CAST: Daisy Ridley - John Boyega - Mark Hamill ...

GENRE: Scifi, Action , Fantasy

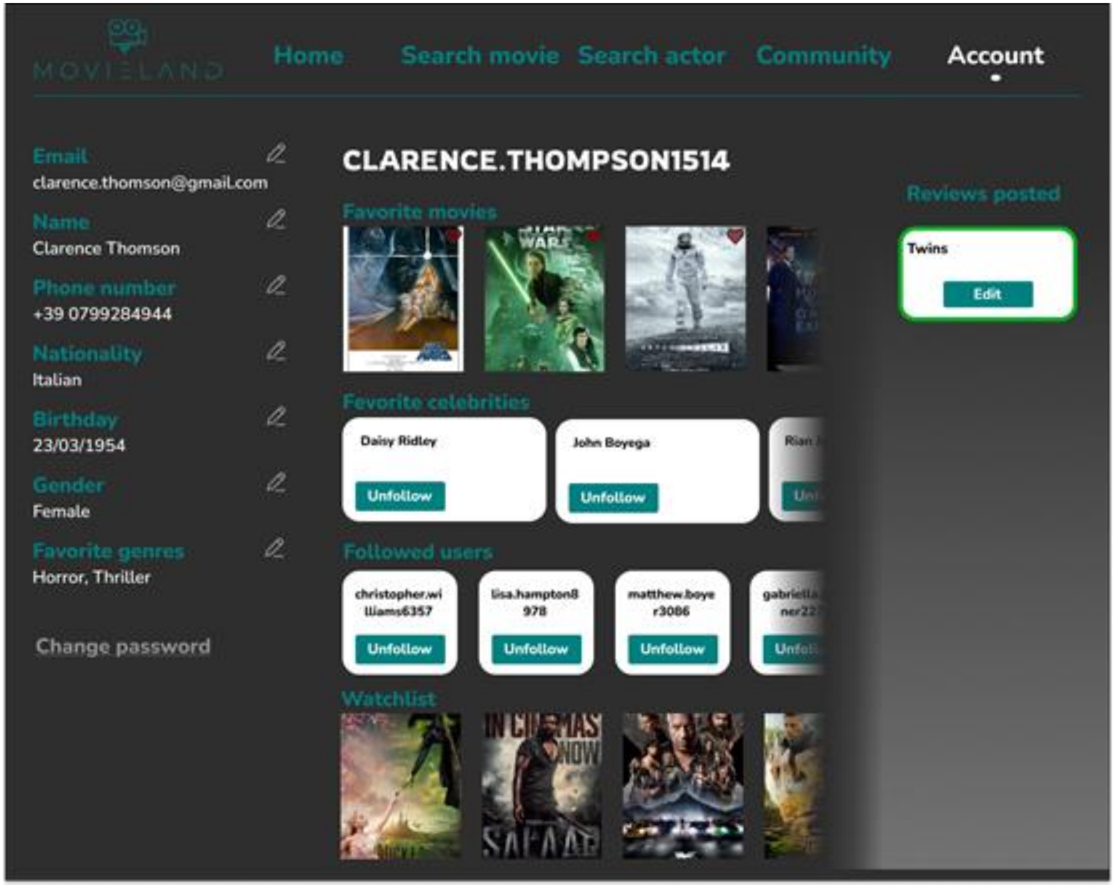
Review | Movie Chat

AUTHOR: erin.perez3884 197

REVIEW: Hilarious, evocative, confusing, brilliant film. Reminds me of Bunuel's L'Age D'Or or Jodorowsky's Holy Mountain-- lots of strange characters mucking about and looking for..... what is it? I laughed almost ...

Actors (vi)

REGISTERED USERS



Email
clarence.thomson@gmail.com

Name
Clarence Thomson

Phone number
+39 0799284944

Nationality
Italian

Birthday
23/03/1954

Gender
Female

Favorite genres
Horror, Thriller

Change password

CLARENCE.THOMPSON1514

Favorite movies

Favorite celebrities

Daisy Ridley — Unfollow

John Boyega — Unfollow

Rian J — Un

Followed users

christopher.williams6357 — Unfollow

lisa.hampton8978 — Unfollow

matthew.boyer3086 — Unfollow

gabriella.ner22 — Unfoll

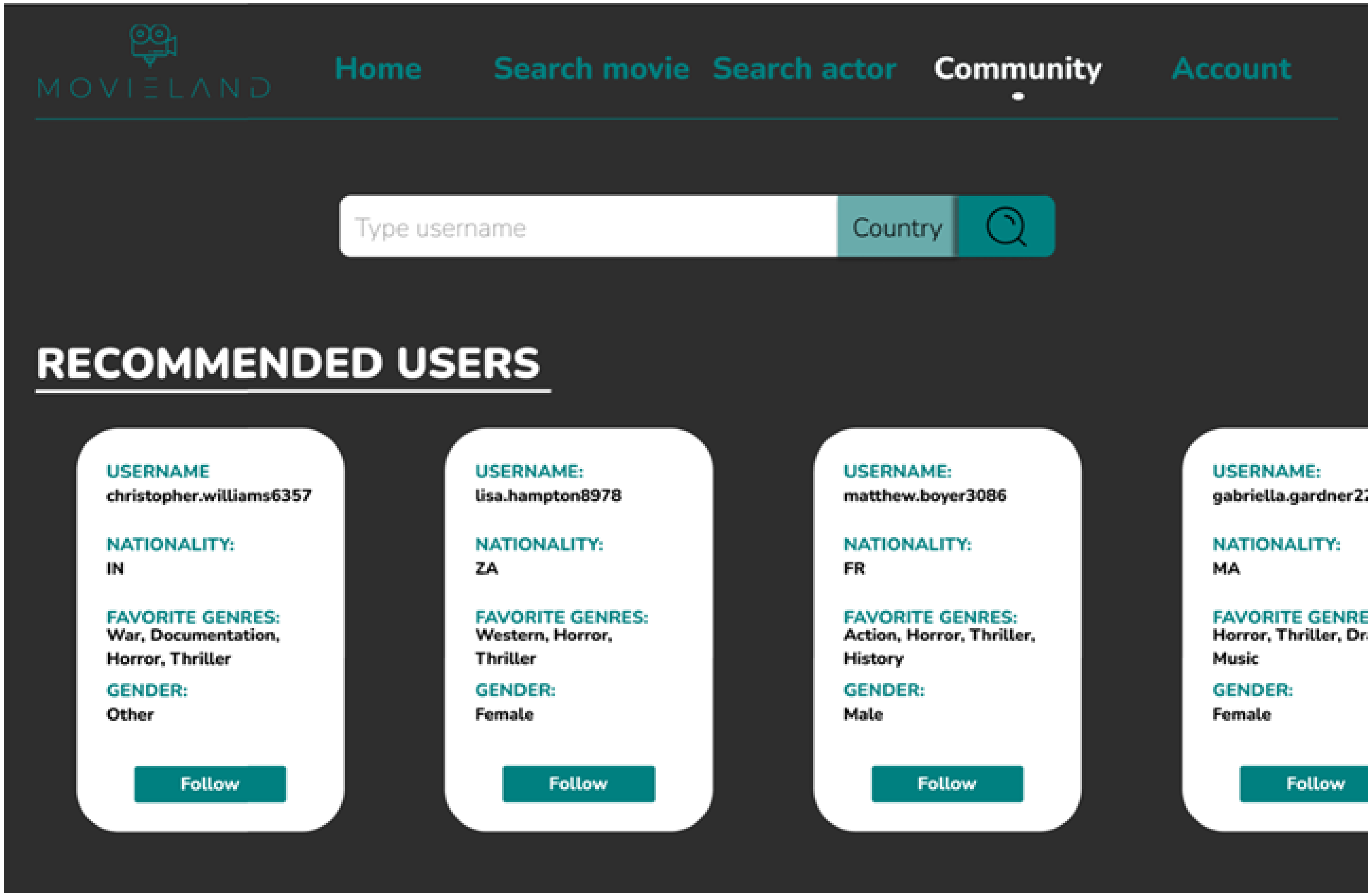
Watchlist

Reviews posted

Twins — Edit

Actors (vii)

REGISTERED USERS



RECOMMENDED USERS

USERNAME
christopher.williams6357

NATIONALITY:
IN

FAVORITE GENRES:
War, Documentation, Horror, Thriller

GENDER:
Other

USERNAME:
lisa.hampton8978

NATIONALITY:
ZA

FAVORITE GENRES:
Western, Horror, Thriller

GENDER:
Female

USERNAME:
matthew.boyer3086

NATIONALITY:
FR

FAVORITE GENRES:
Action, Horror, Thriller, History

GENDER:
Male

USERNAME:
gabriella.gardner2

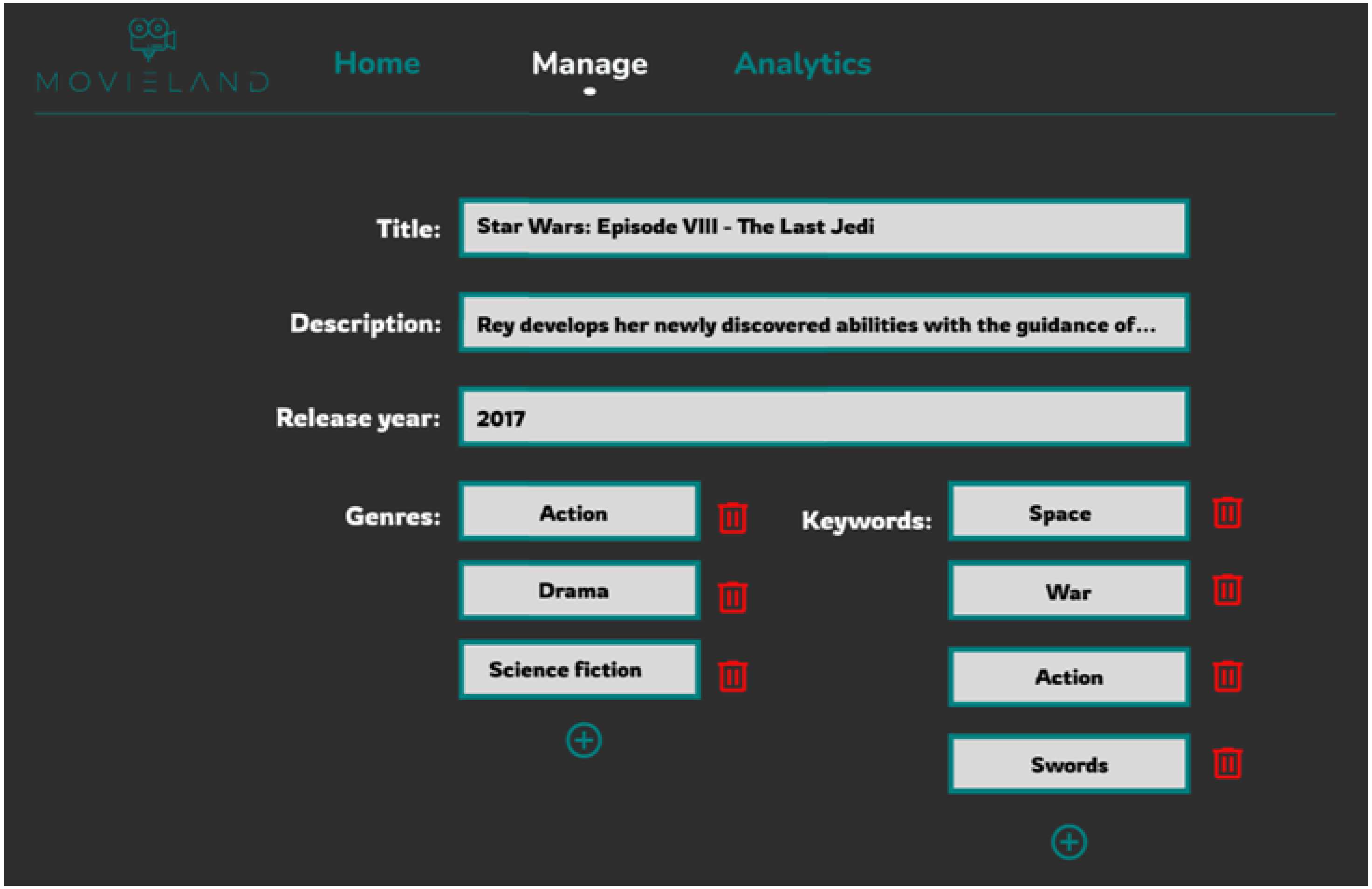
NATIONALITY:
MA

FAVORITE GENRE
Horror, Thriller, Dr
Music

GENDER:
Female

Actors (viii)

MANAGER



Title: Star Wars: Episode VIII - The Last Jedi

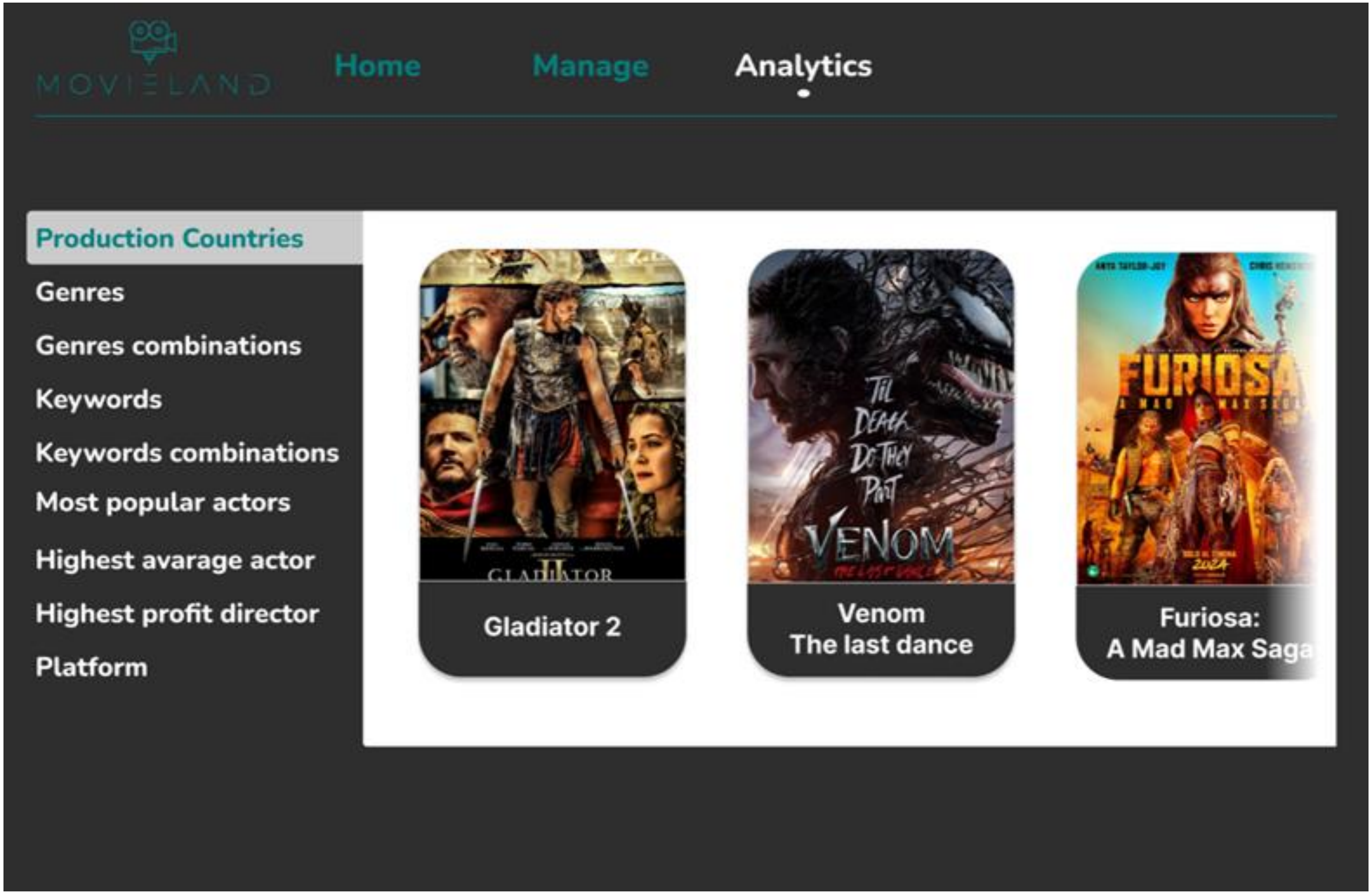
Description: Rey develops her newly discovered abilities with the guidance of...

Release year: 2017

Genres: Action, Drama, Science fiction

Keywords: Space, War, Action, Swords

Actors (ix)


MOVIELAND
Home
Manage
Analytics
Production Countries
Genres
Genres combinations
Keywords
Keywords combinations
Most popular actors
Highest avarage actor
Highest profit director
Platform
Gladiator 2
Venom
The last dance
Furiosa:
A Mad Max Saga

Dataset Description

Source:

Movies and TV series from Netflix, Disney+, Amazon Prime Video, Paramount, and Apple TV, sourced from Kaggle, IMDb Non-Commercial Datasets, Movies Posts and Comments, and IMDb Movie Reviews from Kaggle. The user dataset was generated using AI-based synthetic data generation techniques.

Volume:

340 MB.

Pre-processing:

The original dataset of movies and celebrities have been filtered of useless informations to create more compact datasets. Users informations have been generated using ai as well as connections between them and other entities.

Description:

It includes detailed information about movies and TV series, including cast details, across major streaming platforms, as well as post, comments and reviews about movies and users informations.

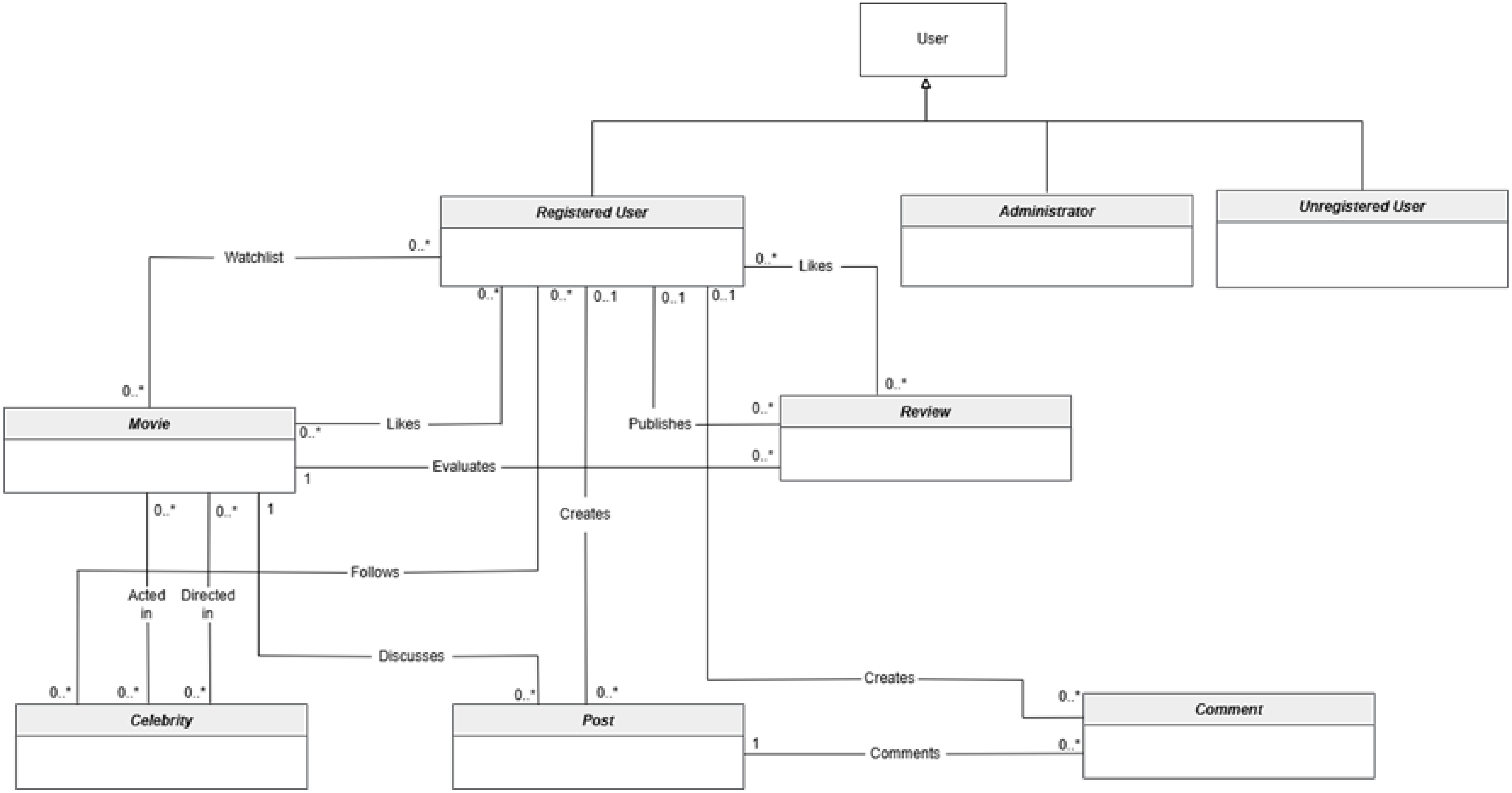
Variety:

The dataset spans diverse genres, languages, and streaming platforms, offering a wide range of content for analysis.

Velocity:

Updates to the dataset occur frequently, due to the hypothetical continuous stream of posts and reviews generated by users.

UML Design Class Diagram



DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

UNIVERSITÀ DI PISA

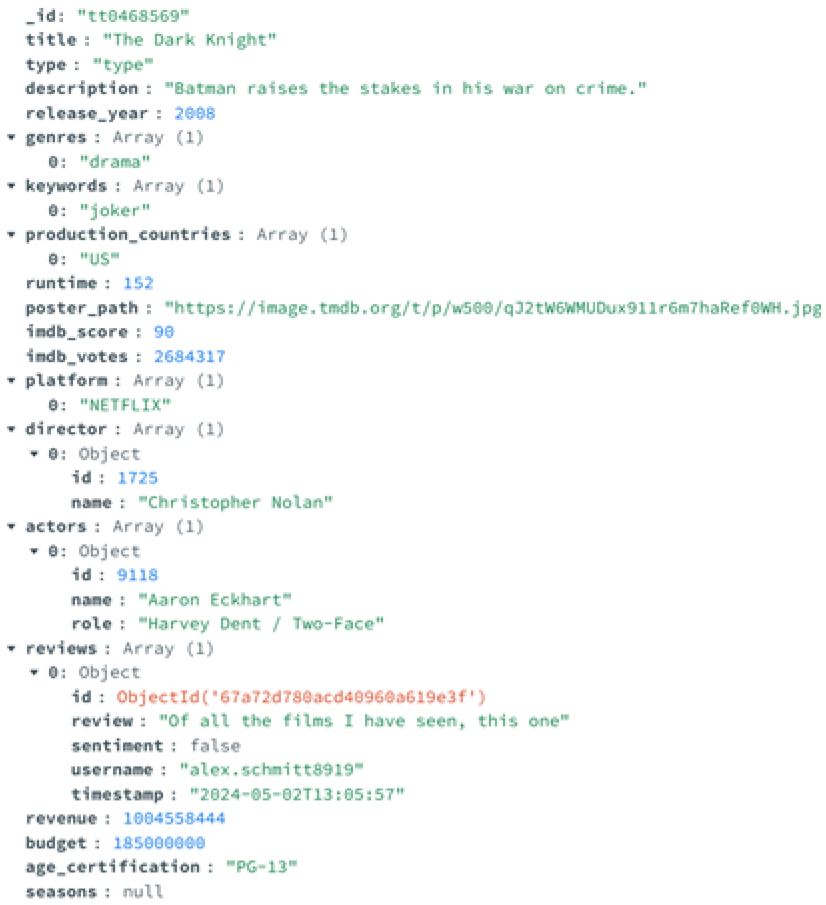
CROSSLAB Innovation for industry 4.0

Application non-functional requirements

- **High Availability**: The system should be available 24/7 with minimal downtime.
- **Scalability**: Easily scalable to handle increased traffic.
- **Fault Tolerance**: Minimize single points of failure.
- **Performance**: Ensure low latency and fast content loading.
- **Data Security**: Encrypt user passwords and use secure data exchanges.
- **Reliability**: The system should be resilient to failures, with regular backups.

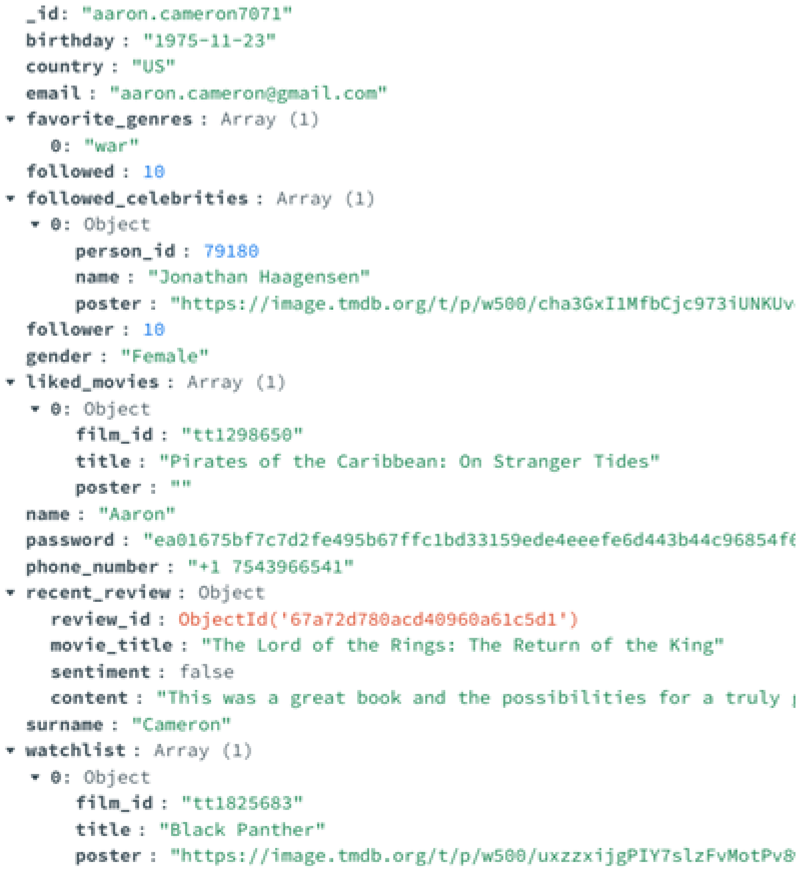
Document DB Design

Movies



```
  _id: "tt0468569"
  title : "The Dark Knight"
  type : "type"
  description : "Batman raises the stakes in his war on crime."
  release_year : 2008
▾ genres : Array (1)
    0: "drama"
▾ keywords : Array (1)
    0: "joker"
▾ production_countries : Array (1)
    0: "US"
  runtime : 152
  poster_path : "https://image.tmdb.org/t/p/w500/qJ2tW6WMUDux911r6m7haRef0WH.jpg
  imdb_score : 90
  imdb_votes : 2684317
▾ platform : Array (1)
    0: "NETFLIX"
▾ director : Array (1)
  ▾ 0: Object
      id : 1725
      name : "Christopher Nolan"
▾ actors : Array (1)
  ▾ 0: Object
      id : 9118
      name : "Aaron Eckhart"
      role : "Harvey Dent / Two-Face"
▾ reviews : Array (1)
  ▾ 0: Object
      id : ObjectId('67a72d780acd40960a619e3f')
      review : "Of all the films I have seen, this one"
      sentiment : false
      username : "alex.schmitt8919"
      timestamp : "2024-05-02T13:05:57"
  revenue : 1004558444
  budget : 185000000
  age_certification : "PG-13"
  seasons : null
```

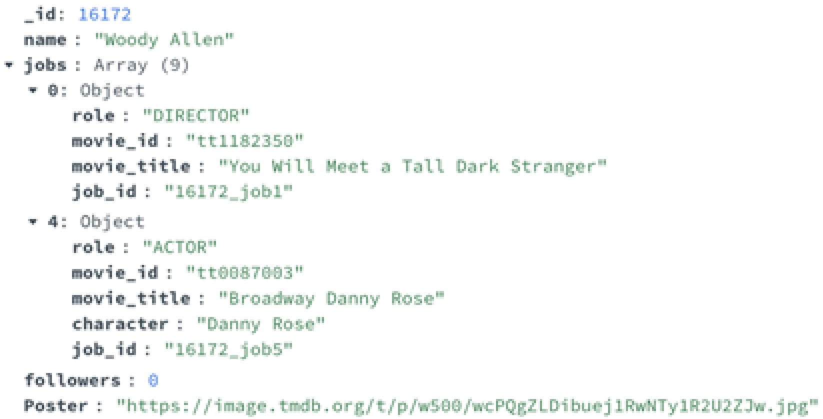
Users



```
  _id: "aaron.cameron7071"
  birthday : "1975-11-23"
  country : "US"
  email : "aaron.cameron@gmail.com"
▾ favorite_genres : Array (1)
    0: "war"
  followed : 10
▾ followed_celebrities : Array (1)
  ▾ 0: Object
      person_id : 79180
      name : "Jonathan Haagensen"
      poster : "https://image.tmdb.org/t/p/w500/cha3GxI1MfbCjc973iUNKUv
  follower : 10
  gender : "Female"
▾ liked_movies : Array (1)
  ▾ 0: Object
      film_id : "tt1298650"
      title : "Pirates of the Caribbean: On Stranger Tides"
      poster : ""
  name : "Aaron"
  password : "ea01675bf7c7d2fe495b67ffc1bd33159ede4eeefe6d443b44c96854f
  phone_number : "+1 7543966541"
▾ recent_review : Object
    review_id : ObjectId('67a72d780acd40960a61c5d1')
    movie_title : "The Lord of the Rings: The Return of the King"
    sentiment : false
    content : "This was a great book and the possibilities for a truly
  surname : "Cameron"
▾ watchlist : Array (1)
  ▾ 0: Object
      film_id : "tt1825683"
      title : "Black Panther"
      poster : "https://image.tmdb.org/t/p/w500/uxzzxijgPIY7slzFvMotPv
```

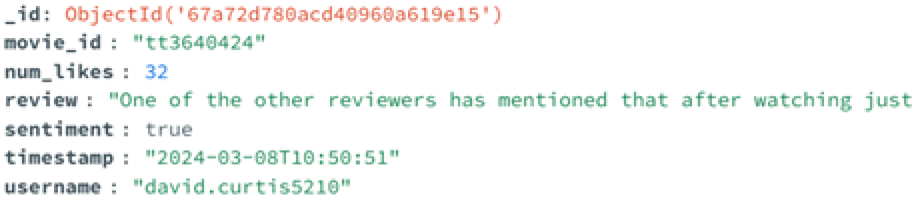
Document DB Design

Celebrities



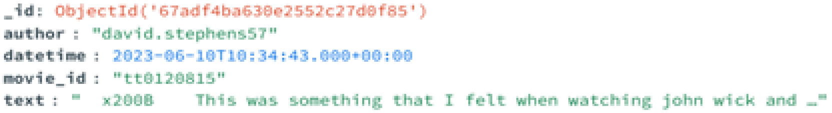
```
  _id: 16172
  name : "Woody Allen"
▾ jobs : Array (9)
  ▾ 0: Object
      role : "DIRECTOR"
      movie_id : "tt1182350"
      movie_title : "You Will Meet a Tall Dark Stranger"
      job_id : "16172_job1"
  ▾ 4: Object
      role : "ACTOR"
      movie_id : "tt0087003"
      movie_title : "Broadway Danny Rose"
      character : "Danny Rose"
      job_id : "16172_job5"
  followers : 0
  Poster : "https://image.tmdb.org/t/p/w500/wcPQgZLDibuej1RwNTy1R2U2ZJw.jpg"
```

Reviews



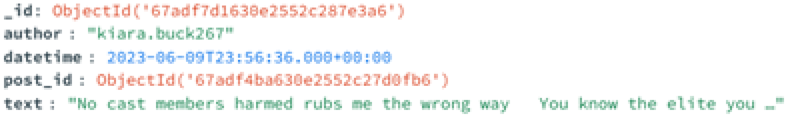
```
_id: ObjectId('67a72d780acd40960a619e15')
movie_id : "tt3640424"
num_likes : 32
review : "One of the other reviewers has mentioned that after watching just
sentiment : true
timestamp : "2024-03-08T10:50:51"
username : "david.curtis5210"
```

Post



```
_id: ObjectId('67adf4ba630e2552c27d0f85')
author : "david.stephens57"
datetime : 2023-06-10T10:34:43.000+00:00
movie_id : "tt0120815"
text : "  x200B    This was something that I felt when watching john wick and …"
```

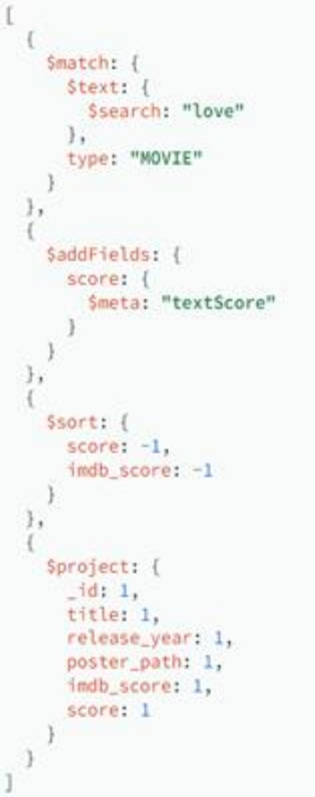
Comments



```
_id: ObjectId('67adf7d1630e2552c287e3a6')
author : "kiara.buck267"
datetime : 2023-06-09T23:56:36.000+00:00
post_id : ObjectId('67adf4ba630e2552c27d0fb6')
text : "No cast members harmed rubs me the wrong way   You know the elite you …"
```

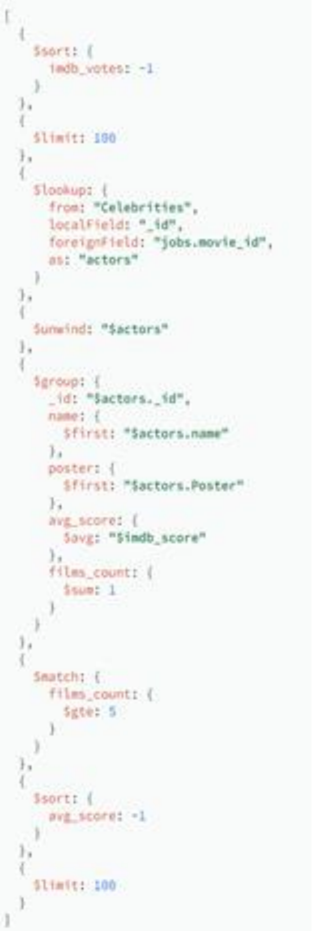
Document DB Design

TEXT SEARCH



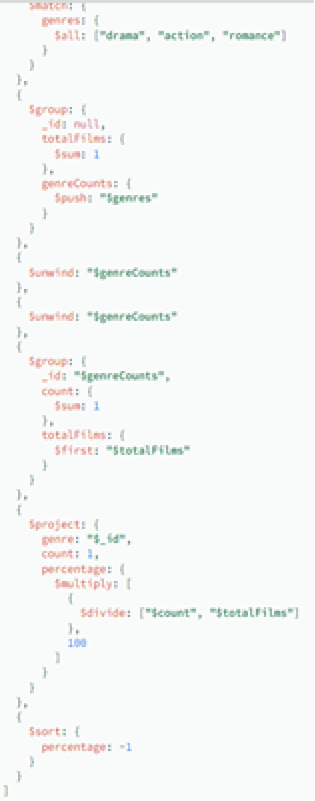
```
[
  {
    $match: {
      $text: {
        $search: "love"
      },
      type: "MOVIE"
    }
  },
  {
    $addFields: {
      score: {
        $meta: "textScore"
      }
    }
  },
  {
    $sort: {
      score: -1,
      imdb_score: -1
    }
  },
  {
    $project: {
      _id: 1,
      title: 1,
      release_year: 1,
      poster_path: 1,
      imdb_score: 1,
      score: 1
    }
  }
]
```

ACTORS AVERAGE MOVIE VOTES



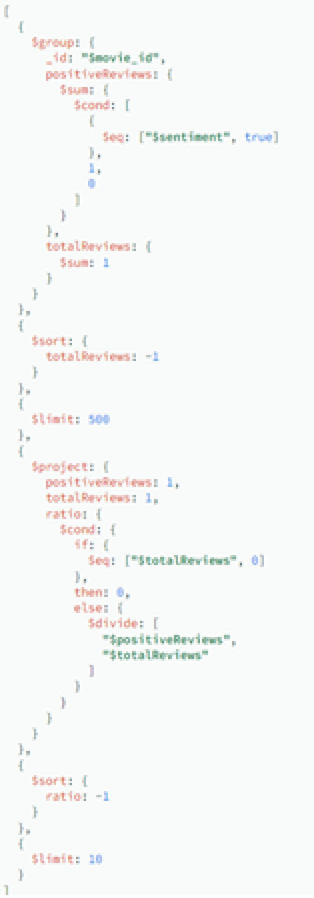
```
[
  {
    $sort: {
      imdb_votes: -1
    }
  },
  {
    $limit: 100
  },
  {
    $lookup: {
      from: "Celebrities",
      localField: "_id",
      foreignField: "jobs.movie_id",
      as: "actors"
    }
  },
  {
    $unwind: "$actors"
  },
  {
    $group: {
      _id: "$actors._id",
      name: {
        $first: "$actors.name"
      },
      poster: {
        $first: "$actors.Poster"
      },
      avg_score: {
        $avg: "$imdb_score"
      },
      films_count: {
        $sum: 1
      }
    }
  },
  {
    $match: {
      films_count: {
        $gte: 5
      }
    }
  },
  {
    $sort: {
      avg_score: -1
    }
  },
  {
    $limit: 100
  }
]
```

GENRE ANALYTICS



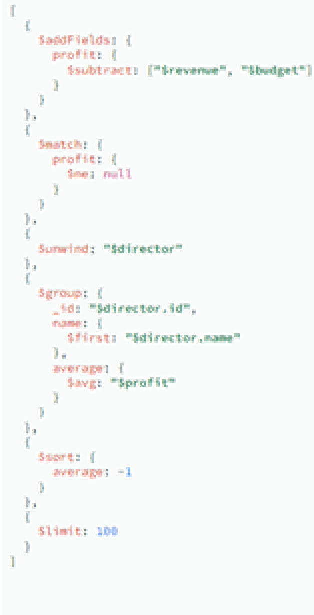
```
    $match: {
      genres: {
        $all: ["drama", "action", "romance"]
      }
    }
  },
  {
    $group: {
      _id: null,
      totalFilms: {
        $sum: 1
      },
      genreCounts: {
        $push: "$genres"
      }
    }
  },
  {
    $unwind: "$genreCounts"
  },
  {
    $unwind: "$genreCounts"
  },
  {
    $group: {
      _id: "$genreCounts",
      count: {
        $sum: 1
      },
      totalFilms: {
        $first: "$totalFilms"
      }
    }
  },
  {
    $project: {
      genre: "$_id",
      count: 1,
      percentage: {
        $multiply: [
          {
            $divide: ["$count", "$totalFilms"]
          },
          100
        ]
      }
    }
  },
  {
    $sort: {
      percentage: -1
    }
  }
]
```

MOVIES WITH BEST REVIEW RATIO



```
[
  {
    $group: {
      _id: "$movie_id",
      positiveReviews: {
        $sum: {
          $cond: [
            {
              $eq: ["$sentiment", true]
            },
            1,
            0
          ]
        }
      },
      totalReviews: {
        $sum: 1
      }
    }
  },
  {
    $sort: {
      totalReviews: -1
    }
  },
  {
    $limit: 500
  },
  {
    $project: {
      positiveReviews: 1,
      totalReviews: 1,
      ratio: {
        $cond: {
          if: {
            $eq: ["$totalReviews", 0]
          },
          then: 0,
          else: {
            $divide: [
              "$positiveReviews",
              "$totalReviews"
            ]
          }
        }
      }
    }
  },
  {
    $sort: {
      ratio: -1
    }
  },
  {
    $limit: 10
  }
]
```

HIGHEST PROFITING DIRECTORS



```
[
  {
    $addFields: {
      profit: {
        $subtract: ["$revenue", "$budget"]
      }
    }
  },
  {
    $match: {
      profit: {
        $ne: null
      }
    }
  },
  {
    $unwind: "$director"
  },
  {
    $group: {
      _id: "$director.id",
      name: {
        $first: "$director.name"
      },
      average: {
        $avg: "$profit"
      }
    }
  },
  {
    $sort: {
      average: -1
    }
  },
  {
    $limit: 100
  }
]
```

MongoDB Replica Set Configuration

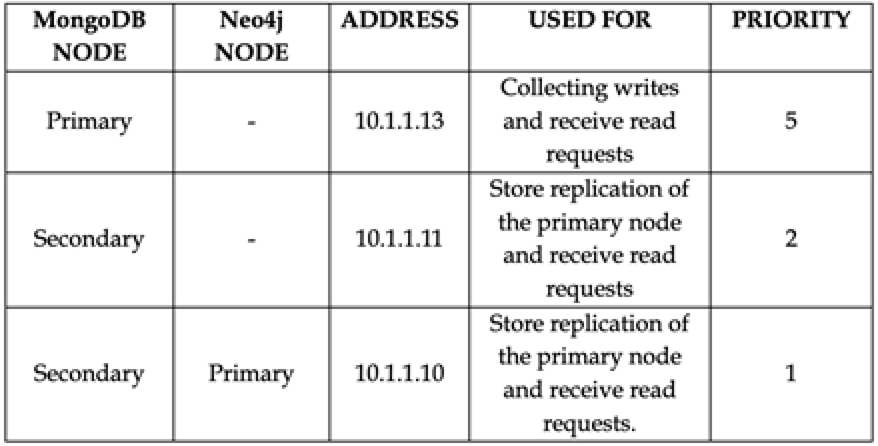
We chose to prioritize data availability over consistency, considering the nature of the social-media-like application.

MongoDB Replication:

- ➢ Read-Preferences: Nearest
- ➢ Write-Concern: W1

Neo4j Replication:

Deployed on a separate machine to prevent a single point of failure and ensure better request distribution.



MongoDB NODE	Neo4j NODE	ADDRESS	USED FOR	PRIORITY
Primary	-	10.1.1.13	Collecting writes and receive read requests	5
Secondary	-	10.1.1.11	Store replication of the primary node and receive read requests	2
Secondary	Primary	10.1.1.10	Store replication of the primary node and receive read requests.	1

MongoDB Indexes

MOVIES

- title (Text, priority: 10)
- keywords (Text, priority: 2)
- imdb_score

CELEBRITIES

- name (Text, priority: 10)
- character (Text, priority: 4)

USERS

- name (Text, priority: 5)
- surname (Text, priority: 10)
- username (Text, priority: 7)
- country

POSTS

- movie_id
- author

COMMENTS

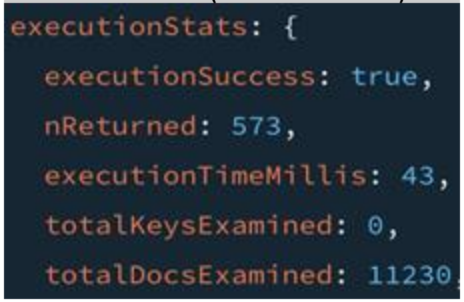
- post_id
- author

REVIEWS

- movie_id

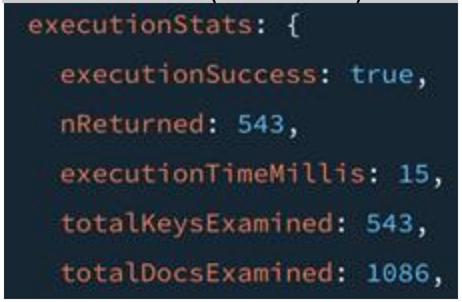
Query performances examples with and without indexes

Movie Search (without index)



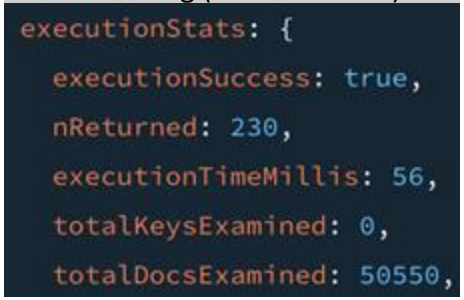
```
executionStats: {
  executionSuccess: true,
  nReturned: 573,
  executionTimeMillis: 43,
  totalKeysExamined: 0,
  totalDocsExamined: 11230
```

Movie Search (with index)



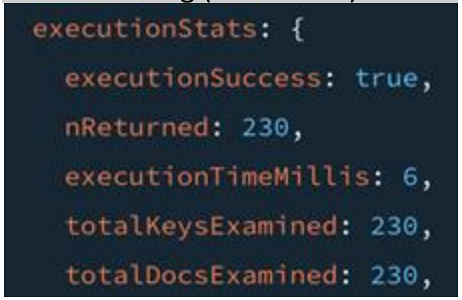
```
executionStats: {
  executionSuccess: true,
  nReturned: 543,
  executionTimeMillis: 15,
  totalKeysExamined: 543,
  totalDocsExamined: 1086,
```

Post Filtering (without index)



```
executionStats: {
  executionSuccess: true,
  nReturned: 230,
  executionTimeMillis: 56,
  totalKeysExamined: 0,
  totalDocsExamined: 50550,
```

Post Filtering (with index)



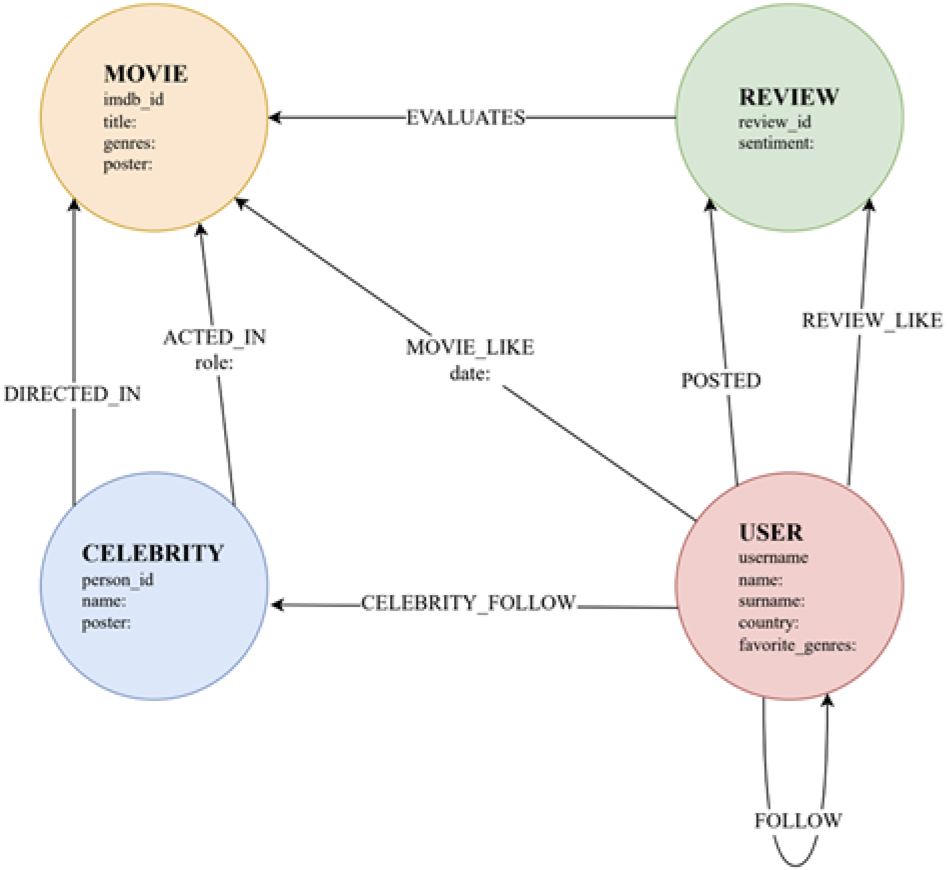
```
executionStats: {
  executionSuccess: true,
  nReturned: 230,
  executionTimeMillis: 6,
  totalKeysExamined: 230,
  totalDocsExamined: 230,
```

Discussion on MongoDB Data Sharding

We selected the potential ***sharding methodologies*** *based on the most frequent queries to optimize performance and balance data distribution.*

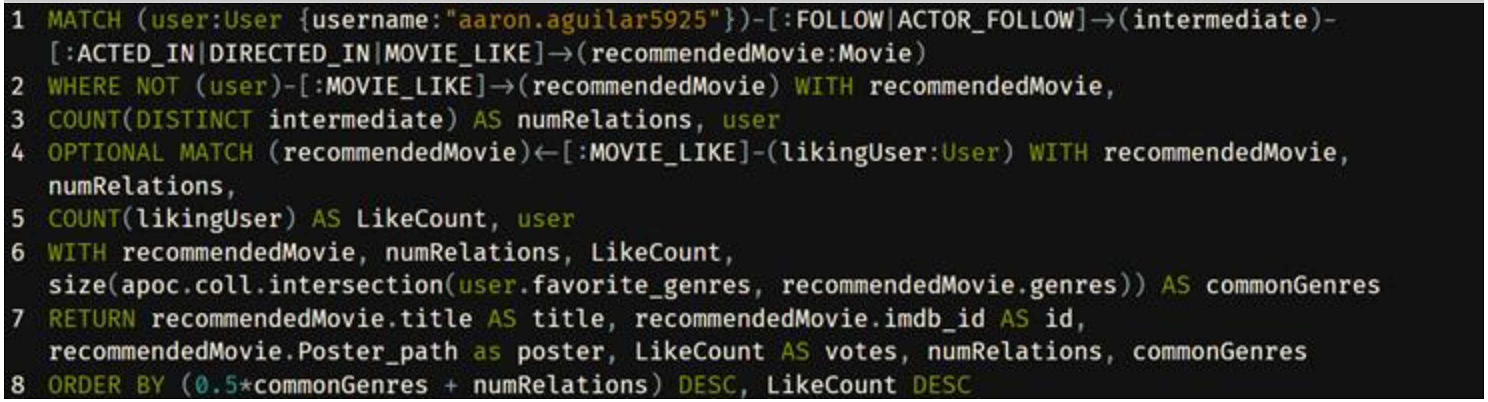
Movies	hashing (imdb_id) Alphabetical (title)
Celebrities	hashing (_id) Alphabetical (name)
Users	hashing (_id) Alphabetical (name)
Reviews	hashing (movie_id)
Posts	hashing (movie_id)
Comments	hashing (post_id)

Graph DB Design



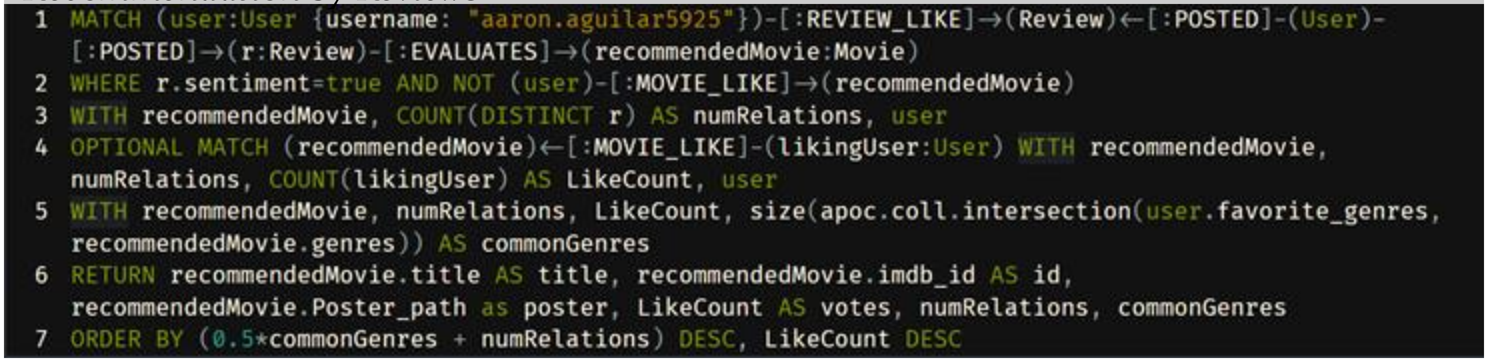
Graph DB Design

Movie Recommendation



```
MATCH (user:User {username:"aaron.aguilar5925"})-[:FOLLOW|ACTOR_FOLLOW]→(intermediate)-
[:ACTED_IN|DIRECTED_IN|MOVIE_LIKE]→(recommendedMovie:Movie)
WHERE NOT (user)-[:MOVIE_LIKE]→(recommendedMovie) WITH recommendedMovie,
COUNT(DISTINCT intermediate) AS numRelations, user
OPTIONAL MATCH (recommendedMovie)←[:MOVIE_LIKE]-(likingUser:User) WITH recommendedMovie,
numRelations,
COUNT(likingUser) AS LikeCount, user
WITH recommendedMovie, numRelations, LikeCount,
size(apoc.coll.intersection(user.favorite_genres, recommendedMovie.genres)) AS commonGenres
RETURN recommendedMovie.title AS title, recommendedMovie.imdb_id AS id,
recommendedMovie.Poster_path as poster, LikeCount AS votes, numRelations, commonGenres
ORDER BY (0.5*commonGenres + numRelations) DESC, LikeCount DESC
```

Recommendation by Reviews



```
MATCH (user:User {username: "aaron.aguilar5925"})-[:REVIEW_LIKE]→(Review)←[:POSTED]-(User)-
[:POSTED]→(r:Review)-[:EVALUATES]→(recommendedMovie:Movie)
WHERE r.sentiment=true AND NOT (user)-[:MOVIE_LIKE]→(recommendedMovie)
WITH recommendedMovie, COUNT(DISTINCT r) AS numRelations, user
OPTIONAL MATCH (recommendedMovie)←[:MOVIE_LIKE]-(likingUser:User) WITH recommendedMovie,
numRelations, COUNT(likingUser) AS LikeCount, user
WITH recommendedMovie, numRelations, LikeCount, size(apoc.coll.intersection(user.favorite_genres,
recommendedMovie.genres)) AS commonGenres
RETURN recommendedMovie.title AS title, recommendedMovie.imdb_id AS id,
recommendedMovie.Poster_path as poster, LikeCount AS votes, numRelations, commonGenres
ORDER BY (0.5*commonGenres + numRelations) DESC, LikeCount DESC
```

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

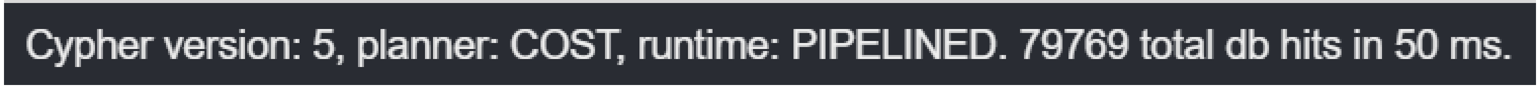
UNIVERSITÀ DI PISA

CROSSLAB Innovation for industry 4.0

Neo4j Indexes

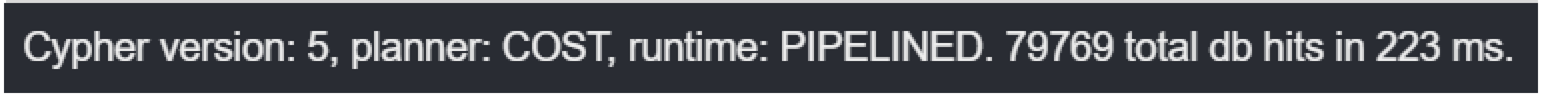
No indexes have been used in Neo4j due to the predominant use of graph traversal in queries and the low frequency of filtering by attributes

Movie recommendation query With indexes



```
Cypher version: 5, planner: COST, runtime: PIPELINED. 79769 total db hits in 50 ms.
```

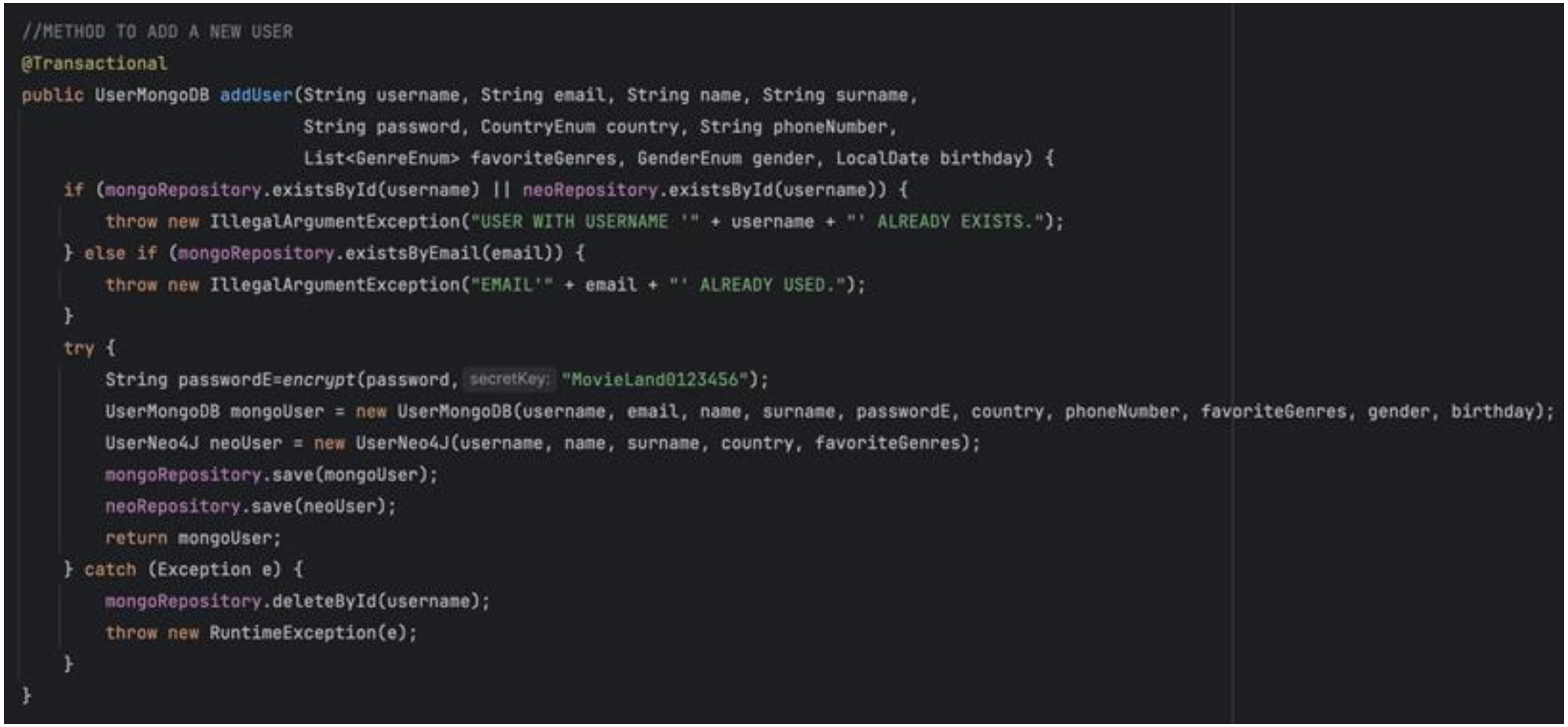
Movie recommendation query Without indexes



```
Cypher version: 5, planner: COST, runtime: PIPELINED. 79769 total db hits in 223 ms.
```

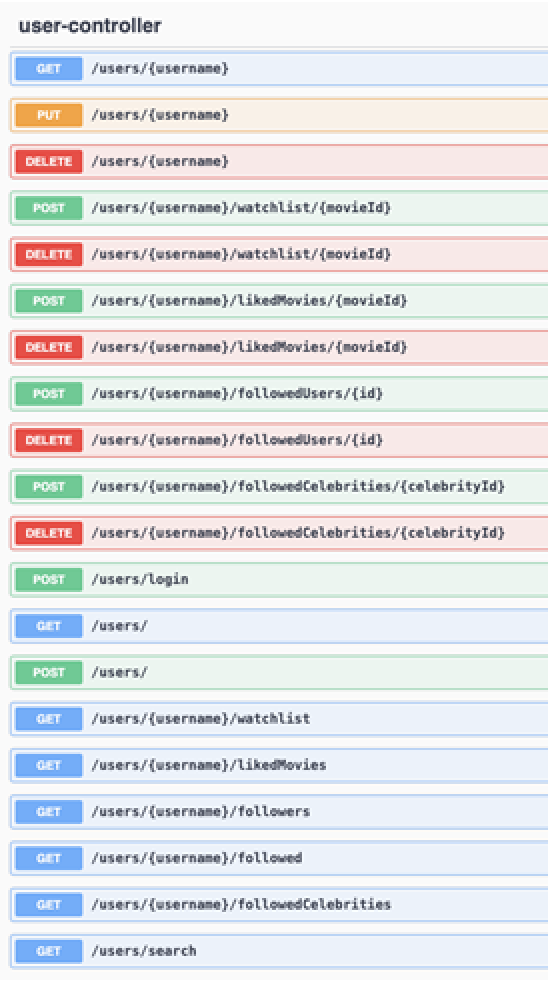
Handling Intra-DB Consistency

Database consistency has been ensured by treating operations as transactions and implementing rollback strategies. Functions to check for inconsistencies have also been created for the manager.



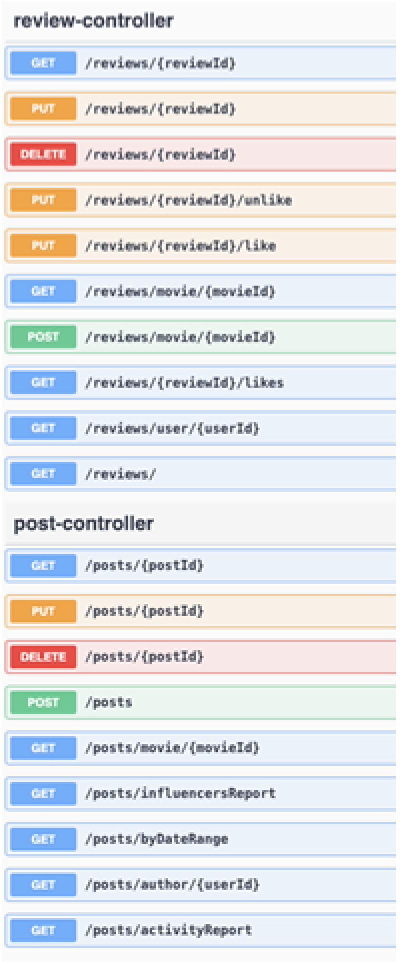
```java
//METHOD TO ADD A NEW USER
@Transactional
public UserMongoDB addUser(String username, String email, String name, String surname,
                           String password, CountryEnum country, String phoneNumber,
                           List<GenreEnum> favoriteGenres, GenderEnum gender, LocalDate birthday) {
    if (mongoRepository.existsById(username) || neoRepository.existsById(username)) {
        throw new IllegalArgumentException("USER WITH USERNAME '" + username + "' ALREADY EXISTS.");
    } else if (mongoRepository.existsByEmail(email)) {
        throw new IllegalArgumentException("EMAIL'" + email + "' ALREADY USED.");
    }
    try {
        String passwordE=encrypt(password, secretKey: "MovieLand0123456");
        UserMongoDB mongoUser = new UserMongoDB(username, email, name, surname, passwordE, country, phoneNumber, favoriteGenres, gender, birthday);
        UserNeo4J neoUser = new UserNeo4J(username, name, surname, country, favoriteGenres);
        mongoRepository.save(mongoUser);
        neoRepository.save(neoUser);
        return mongoUser;
    } catch (Exception e) {
        mongoRepository.deleteById(username);
        throw new RuntimeException(e);
    }
}
```

Swagger UI REST APIs documentation



user-controller

- GET /users/{username}
- PUT /users/{username}
- DELETE /users/{username}
- POST /users/{username}/watchlist/{movieId}
- DELETE /users/{username}/watchlist/{movieId}
- POST /users/{username}/likedMovies/{movieId}
- DELETE /users/{username}/likedMovies/{movieId}
- POST /users/{username}/followedUsers/{id}
- DELETE /users/{username}/followedUsers/{id}
- POST /users/{username}/followedCelebrities/{celebrityId}
- DELETE /users/{username}/followedCelebrities/{celebrityId}
- POST /users/login
- GET /users/
- POST /users/
- GET /users/{username}/watchlist
- GET /users/{username}/likedMovies
- GET /users/{username}/followers
- GET /users/{username}/followed
- GET /users/{username}/followedCelebrities
- GET /users/search

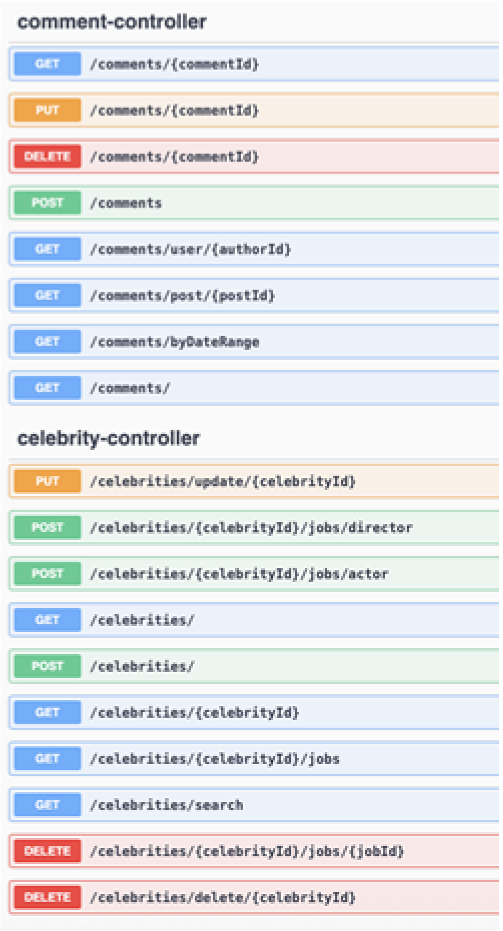


review-controller

- GET /reviews/{reviewId}
- PUT /reviews/{reviewId}
- DELETE /reviews/{reviewId}
- PUT /reviews/{reviewId}/unlike
- PUT /reviews/{reviewId}/like
- GET /reviews/movie/{movieId}
- POST /reviews/movie/{movieId}
- GET /reviews/{reviewId}/likes
- GET /reviews/user/{userId}
- GET /reviews/

post-controller

- GET /posts/{postId}
- PUT /posts/{postId}
- DELETE /posts/{postId}
- POST /posts
- GET /posts/movie/{movieId}
- GET /posts/influencersReport
- GET /posts/byDateRange
- GET /posts/author/{userId}
- GET /posts/activityReport



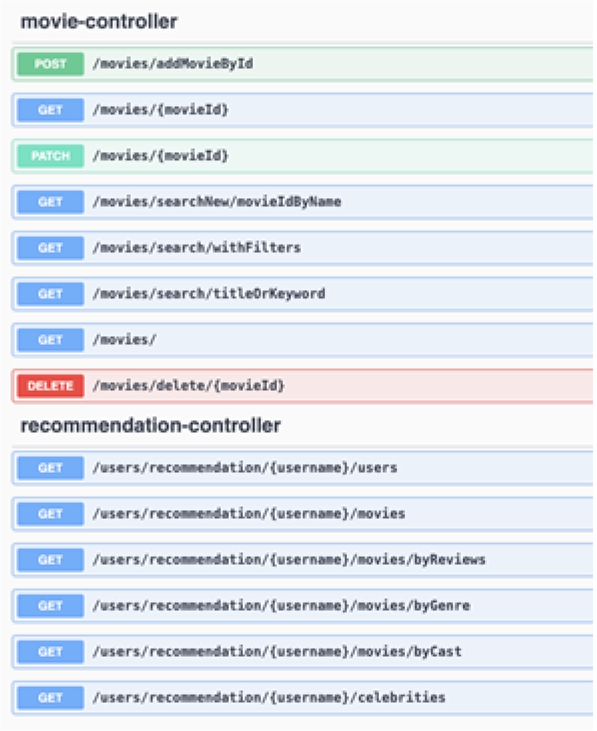
comment-controller

- GET /comments/{commentId}
- PUT /comments/{commentId}
- DELETE /comments/{commentId}
- POST /comments
- GET /comments/user/{authorId}
- GET /comments/post/{postId}
- GET /comments/byDateRange
- GET /comments/

celebrity-controller

- PUT /celebrities/update/{celebrityId}
- POST /celebrities/{celebrityId}/jobs/director
- POST /celebrities/{celebrityId}/jobs/actor
- GET /celebrities/
- POST /celebrities/
- GET /celebrities/{celebrityId}
- GET /celebrities/{celebrityId}/jobs
- GET /celebrities/search
- DELETE /celebrities/{celebrityId}/jobs/{jobId}
- DELETE /celebrities/delete/{celebrityId}

Swagger UI REST APIs documentation

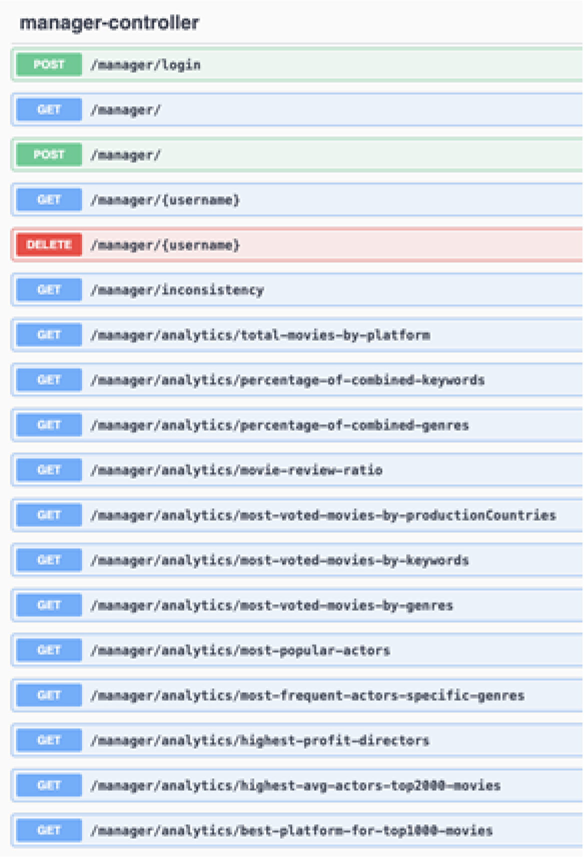


movie-controller

```
POST    /movies/addMovieById
GET     /movies/{movieId}
PATCH   /movies/{movieId}
GET     /movies/searchNew/movieIdByName
GET     /movies/search/withFilters
GET     /movies/search/titleOrKeyword
GET     /movies/
DELETE  /movies/delete/{movieId}
```

recommendation-controller

```
GET     /users/recommendation/{username}/users
GET     /users/recommendation/{username}/movies
GET     /users/recommendation/{username}/movies/byReviews
GET     /users/recommendation/{username}/movies/byGenre
GET     /users/recommendation/{username}/movies/byCast
GET     /users/recommendation/{username}/celebrities
```



manager-controller

```
POST    /manager/login
GET     /manager/
POST    /manager/
GET     /manager/{username}
DELETE  /manager/{username}
GET     /manager/inconsistency
GET     /manager/analytics/total-movies-by-platform
GET     /manager/analytics/percentage-of-combined-keywords
GET     /manager/analytics/percentage-of-combined-genres
GET     /manager/analytics/movie-review-ratio
GET     /manager/analytics/most-voted-movies-by-productionCountries
GET     /manager/analytics/most-voted-movies-by-keywords
GET     /manager/analytics/most-voted-movies-by-genres
GET     /manager/analytics/most-popular-actors
GET     /manager/analytics/most-frequent-actors-specific-genres
GET     /manager/analytics/highest-profit-directors
GET     /manager/analytics/highest-avg-actors-top2000-movies
GET     /manager/analytics/best-platform-for-top1000-movies
```

Thank you for your attention!

MOVIELAND

DIPARTIMENTO DI
INGEGNERIA
DELL'INFORMAZIONE

IN SUPREMÆ DIGNITATIS
1343
UNIVERSITÀ DI PISA

CROSSLAB
Innovation for industry 4.0