Discover your treasures hidden within!

Assignment: Movie Review Service

The assignment is to implement a service related to movie reviews, and following is the description of it. You can choose *any* language (preferably JAVA) that you are comfortable with. Please push your code to Github and share the link with us when you reply.

Overview

The movie review service collects reviews for movies from its users. Later these movie reviews are used to derive insights which helps in enriching the lives of its customers with entertainment.

Internal Capabilities

1. Users of the service can review only movies which are released so far, they cannot review upcoming movies.
2. Users can give a review-score between 1 to 10. (Higher the score the better the liking for the movie). Currently we are not allowing a user to review the same movie multiple times.
3. The service by default on-boards a user as a 'viewer'. Later a 'viewer' can be upgraded to a 'critic' after he/she has published more than 3 reviews for various movies.
4. Critics are considered as experts in the judgement here, so critics reviews will be captured with more weightage. i.e. 6 review rating of a critic will be considered as 12 (2x) NOTE: Older reviews by the user as `viewer` shall not be affected.
5. **Good To Have:** Based on the users behaviour, the service should give the capability to plugin more *user upgradation policies*. Eg. User -> Critic -> Expert -> Admin.

Requirements

Each of the following features needs to be implemented:

1. Adding users and movies.
2. User to review a movie.
3. List top *n* movies by total review score by 'critics' in a particular genre.
4. Average review score in a particular year of release.
5. Average review score for a particular movie.

Expectations

1. Make sure that you have working and demonstrable code for all the above requirements.
2. Feature requirements should be strictly followed. Work on the expected output first and then add good-to-have features of your own.
3. Use of proper abstraction, separation of concerns is required.
4. Code should easily accommodate new requirements with minimal changes.
5. Proper exception handling is required.
6. Code should be modular, readable and unit-testable.

Important Notes:

- **Do not use any database store**, use in-memory data structure.
- Do not write an API on top of the service methods.
- Do not create any UI for the application.
- Do not build a Command line interface, executing test cases or via the simple Main function should be sufficient.

Sample Test Cases:

1. **Onboard 10 movies onto your platform in 3 different years.**
    a. Add Movie("Don" released in Year 2006 for Genres “Action” & “Comedy”)
    b. Add Movie("Tiger" released in Year 2008 for Genre “Drama”)
    c. Add Movie("Padmaavat" released in Year 2006 for Genre “Comedy”)
    d. Add Movie("Lunchbox" released in Year 2021 for Genre “Drama”)
    e. Add Movie("Guru" released in Year 2006 for Genre “Drama”)
    f. Add Movie("Metro" released in Year 2006 for Genre “Romance”)
2. **Add users to the system:**
    a. Add User(“SRK”)
    b. Add User(“Salman”)
    c. Add User(“Deepika”)
3. **Add Reviews:**
    a. add_review(“SRK”, “Don”, 2)
    b. add_review(“SRK”, “Padmavaat”, 8)
    c. add_review(“Salman”, “Don”, 5)
    d. add_review(“Deepika”, “Don”, 9)
    e. add_review(“Deepika”, “Guru”, 6)
    f. add_review(“SRK”,”Don”, 10) - *Exception multiple reviews not allowed*

g. add_review("Deepika", "Lunchbox", 5) - *Exception movie yet to be released*
h. add_review("SRK", "Tiger", 5). *Since 'SRK' has published 3 reviews, he is promoted to 'critic' now.*
i. add_review("SRK", "Metro", 7)