VIDEO DATABASE

DUȚU ALIN CĂLIN

CONTENTS

1 INTRODUCTION

1.1 Context

Since the debut of streaming services such as Netflix, HBO Max, and Disney+, people started to have more options at the press of a button.

For this particular reason, this project serves as a valuable resource, offering a blend of expert opinions and audience insights on a wide range of movies. The platform provides a space for thoughtful critique and community discussion, helping users navigate the diverse world of cinema.

1.2 Objectives

This project aims to provide an overview of Object-Oriented programming by creating a simple platform that delivers detailed information and diverse perspectives on each film and serial registered to enhance the movie-watching experience.

The following objectives are covered upon implementing the program:

- Familiarize with Object-Oriented Programming
- Understand and utilize basic OOP concepts: Aggregation, Inheritance, Polymorphism and Abstraction
- Adopt a specific coding design and style for Object Oriented Programming

2 PROPOSED SOLUTION

2.1 Solution presentation

The project aims to implement an information platform for movies and serials to practice and understand how to design and structure the code using the OOP concepts in mind.

There are 2 entities involved in the program's flow: videos and users.

The videos are split into movies and serials. Both types have common metadata such as title, year of release, and the multiple genres the video suits. However, the metadata for serials is split into seasons, including the number, duration, and rating. In contrast, the movies include only the duration and a rating in their metadata.

The users are split into standard and premium users. The premium users have access to exclusive strategies that provide better recommendations. In addition, users are also classified into normal users and actors.

Users can execute searches for movies they seek, visualize relevant information about any movie or serial, set the number of visualizations, view, and offer ratings for seen movies and serial seasons, add videos to a private list of favorites, and receive recommendations based on the user's category.

3 IMPLEMENTATION DETAILS

The proposed solution consists of a Java application that registers movies, serials, and users and processes multiple requests specific to the user's category.

Every entity and action is presented in a JSON format containing multiple fields with information relevant to the represented object.

3.1 Users

The fields relevant for a simple user are:

- Username - The name chosen by the user to be identified in the platform
- Subscription - The subscription type
- History - A list with the user's history that includes the accessed movies' names and the number of views
- Favourite - A list with a selection of movies added by the user

Example:

```
{
   "username": "madUnicorn3",
   "subscription": "BASIC",
   "history": [
       {
           "name": "SPF-18",
           "no_views": 3
       },
       {
           "name": "Waiting for the Barbarians",
           "no_views": 2
       },
       {
           "name": "The Circle",
           "no_views": 3
       },
       {
           "name": "Euphoria",
```

```
            "no_views": 3
        },
        {
            "name": "The 4400",
            "no_views": 3
        },
        {
            "name": "The Haunting of Hill House",
            "no_views": 2
        }
    ],

    "favourite": [
        "Waiting for the Barbarians",
        "The Circle"
    ]
}
```


3.2 Actors

The fields relevant for an actor are:

- Name - Actor's name
- Career Description - A brief description of the actor's career
- Filmography - A list of movies in which the actor had played
- Awards - A list of earned awards

Example:


```
{
    "name": "Johnny Depp",
    "career_description": "John Christopher \"Johnny\" Depp II (born June 9, 1963)...
    "filmography": [
        "Waiting for the Barbarians",
        "The Professor",
        "City of Lies",
        "Fantastic Beasts: The Crimes of Grindelwald",
        "London Fields",
        "Don Juan DeMarco",
        "Ed Wood",
        "Pirates of the Caribbean: Dead Men Tell No Tales"
```

```
    ],
    "awards": []
}
```

3.3 Movies

The fields relevant to a movie are:

- Name - The name of the movie
- Year - The launching year
- Duration - The duration in minutes
- Genres - A list of genres the movie fits into
- Actors - A list of actors that play in the movie

Example:

```
{
    "name": "The Child in Time",
    "year": "2018",
    "duration": 90,
    "genres": [
        "TV Movie",
        "Drama"
    ],
    "actors": [
        "Benedict Cumberbatch",
        "Stephen Campbell Moore",
        "Kelly Macdonald"
    ]
}
```

3.4 Serials

The fields relevant to a movie are:

- Name - The name of the serial
- Year - The launching year
- Cast - A list of actors that play in the movie
- Genres - A list of genres the movie fits into
- Number of seasons

- Seasons - A list of objects representing each season which contains the current season number and the duration in minutes

Example:

```
{
    "name": "Marvel's Cloak & Dagger",
    "year": "2018",
    "cast": [
        "Aubrey Joseph",
        "Olivia Holt"
    ],
    "genres": [
        "Action & Adventure",
        "Drama",
        "Sci-Fi & Fantasy"
    ],
    "number_of_seasons": 2,
    "seasons": [
        {
            "current_season": 1,
            "duration": 44
        },
        {
            "current_season": 2,
            "duration": 44
        }
    ]
}
```

3.5 Commands

There are 3 types of commands:

- Favourite - Adding a new video to the user's favorite list
- View - Registers that a user viewed a specific video
- Rating - Registers a user's rating for a specific video

3.5.1 Favourite

The Favourite command checks for 3 things before adding a new video:

- Checks if the user is registered
- Checks if the video exists in the user's history
- Checks if the video is not already in the user's favorite list

If one of these checks is false, then the command is canceled and there will be an error log with the problem shown in the output. Otherwise, the requested video will be added to the user's favorite list.

3.5.2 View

The View command simply checks if the video exists. If it does, either a new entry is created in the user's history with one view or increments the views of an existing entry.

3.5.3 Rating

The Rating command checks 3 conditions before adding a new video:

- Checks if the user is registered
- Checks if the video is registered
- Checks if the user hasn't already rated the movie

If these conditions are met, the rating is going to be registered, the rating number is going to be incremented and the average rating recalculated.

3.6 Queries

There are different query parameters based on the objects searched and the criteria are as follows:

- Author
  - Average - First N actors sorted by the average rating of the movies and serials they played
  - Awards - The actors with awards sorted by the number of awards acquired
  - Description - The actors with descriptions that contain the filtering word (case insensitive)
- Videos
  - Rating - The first N videos sorted by rating. The videos with no rating are not considered.
  - Favorite - The first N videos sorted by the number of favorites

  - Longest - The first N videos sorted by their length
  - Most Viewed - The first N videos sorted by the number of views
- Users
  - Number of ratings - The first N users sorted by the given number of ratings

3.7 Recommendations

The recommendations are provided based on the strategy chosen by the user. For all users, there are 2 strategies available:

- Standard - returns the first video unseen by the user
- Best Unseen - returns the best video unseen by the user based on rating

Additionally, the premium users can choose from 3 additional recommendation strategies:

- Popular - The first unseen video from the most popular genre based on the number of views
- Favorite - The most favorite video based on the number of saves in the favorite lists
- Search - All the unseen videos from a provided genre sorted by rating

3.8 Testing

For testing the proposed solution, the following methods have been used:

- Checker - Used for testing general cases
- Manual testing - Used for testing corner cases

4 CONCLUSIONS

In conclusion, this program aims to provide an overview of Object-Oriented programming by creating a simple platform that provides detailed information and diverse perspectives on any video registered to enhance the movie-watching experience. The app offers basic commands, different query modalities, and recommendations based on the user type to facilitate the experience of organizing and developing a program that uses the Object-Oriented programming paradigm.

Bibliography

[1] POO team @ Polytechnic University of Bucharest. Tema - VideosDB. `https://ocw.cs.pub.ro/courses/poo-ca-cd/arhiva/teme/2020/tema`. The homework is written in Romanian; Latest access: 15.08.2024;.

[2] POO team @ Polytechnic University of Bucharest. Curs - Programare Orientata Obiect. `https://archive.curs.upb.ro/2020/course/view.php?id=10544`. The course is written in Romanian; Latest access: 15.08.2024;.