Tema - VideosDB

- Responsabili: Maria Leoveanu, Radu Pavel, Ionela Nicuță, Bianca Ciuche, Florian-Luis Micu, Adelin Stanca, Mihai Burdușelu, Yasemin Menan, Răzvan Abagiu, Raimond Varga, Miruna-Maria Fătu
- Deadline: 28.11.2021 - 23:55
- Deadline hard: 5.12.2021 - 23:55
- Data publicării: 4.11.2021
- Data ultimei modificări: 4.11.2021

Obiective

În cadrul acestei teme veți implementa o platformă simplificată ce oferă informații despre filme și despre seriale. Utilizatorii pot să primească recomandări personalizate pe baza preferințelor.

Principalul scop al temei este de a vă familiariza cu scrierea codului în mod orientat pe obiect prin folosirea de concepte precum moștenirea, agregarea, polimorfismul și abstractizarea. De asemenea veți putea să vă exersați programarea în Java.

În cadrul acestui curs punem accentul pe calitatea codului, atât ca design, cât și ca aspect. De aceea, pentru toate temele vă încurajăm să folosiți tool-ul de analiză statică checkstyle, cu care vă puteți verifica respectarea code style-ului oficial al limbajului Java.

Descriere

Platforma pe care o veți implementa simulează acțiuni pe care le pot face utilizatorii pe o platforma de vizualizare de filme: ratings, vizualizare film, căutări, recomandări etc.

Entitățile pe care le veți modela vor fi:

- Video
  - De două tipuri: filme și seriale (shows). Diferența dintre ele este că serialele au sezoane.
  - In cazul serialelor, episoadele sezoanelor sunt nesemnificative in cerintele temei.
  - Toate videourile au în comun titlu, an lansare, unul sau mai multe genuri (e.g. comedie, thriller)
  - Sezoanele unui serial au asociat un număr, durata întregului sezon și un rating.
  - Filmele au durată și rating
- User (utilizator)
  - Are două categorii: standard și premium

- Are videouri favorite și videouri vizualizate

Datele pentru aceste entități sunt încărcate din fișierele JSON oferite ca intrare în teste. Ele sunt ținute într-un **Repository.**
În secțiunea Detalii de implementare vom oferi exemple pentru fiecare entitate și acțiunile efectuate pe ele.

Utilizatorii pot realiza următoarele trei tipuri de **acțiuni**: comenzi, query-uri și recomandări.

Execuția temei

1. Se încarcă datele citite din fișierul de test, în format JSON, în obiecte.
2. Se primesc secvențial acțiuni (comenzi, queries sau recomandări) și se execută pe măsură ce sunt primite, rezultatul lor având efect asupra Repository-ului
3. După executarea unei acțiuni, se afișează rezultatul ei în fișierul JSON de ieșire
4. La terminarea tuturor acțiunilor se termina și execuția programului.

Comenzile

Acestea reprezintă abilitatea unui utilizator de a realiza acțiuni directe, fiind de 3 tipuri diferite.

- **Favorite** - adaugă un video în lista de favorite videos ale acelui user, dacă a fost deja vizionat de user-ul respectiv.
- **View** - vizualizează un video prin marcarea lui ca văzut. Dacă l-a mai văzut anterior, se incrementează numărul de vizualizări ale acelui video de către user. Atunci cand se vizualizeaza un serial, se vizualizeaza toate sezoanele.
- **Rating** - oferă rating unui video care este deja văzut (la seriale se aplică la pentru fiecare sezon in parte(spre deosebire de vizionare, unde se face pe tot serialul)).
    - Ratingul diferă în funcție de tip - pentru seriale este pentru fiecare sezon in parte.
    - Ratingul poate fi dat o singură dată de către un utilizator. Pentru seriale poate fi o singură dată pe sezon.
    - Un serial are mai multe sezoane, fiecare putand primi cate o nota. Rating-ul se calculeaza ca o medie aritmetica a tuturor sezoanelor.

Query-urile

Acestea reprezintă căutări globale efectuate de utilizatori după actori, video-uri și utilizator. Rezultatele acestor query-uri sunt afișate ca output al testului.

Un query contine tipul de informație căutată: actor/video/user, criteriul de sortare și diverși parametri. În secțiunea Implementare aveți exemple pentru toate felurile de queries. Query-urile conțin ca parametru și ordinea sortării, dacă sortarea să se facă crescător sau descrescător. Criteriul de sortare secundar este ordinea alfabetică descrescătoare/crescătoare în funcție de ordinea de la primul criteriu.

Pentru actori:

- **Average** - primii N actori (N dat în query) sortați după media ratingurilor filmelor și a serialelor în care au jucat. Media este aritmetica si se calculeaza pentru toate videoclipurile(cu ratingul total diferit de 0) in care a jucat.
- **Awards** - toți actorii cu premiile menționate în query. Atenție, trebuie ca acei actori să fi primit toate premiile cerute, nu doar pe unele din ele. Sortarea se va face după numărul total de premii al fiecărui actor, după ordinea precizată în input. De exemplu, daca un actor deține doua tipuri de premii: `BEST_PERFORMANCE` câștigat de 3 ori și `BEST_SUPPORTING_ACTOR` câștigat de 4 ori, numărul total de premii este 7. Acesta este numărul față de care vor fi făcute sortările.
- **Filter Description** - toți actorii în descrierea cărora apar toate keywords-urile (case insensitive) menționate în query. Sortarea se va face după ordinea alfabetică a numelor actorilor, după ordinea precizată în input.

Pentru video-uri:

- **Rating** - primele N video-uri sortate după rating. Pentru seriale rating-ul se calculează ca media tuturor sezoanelor, cu conditia ca cel putin un sezon sa aiba rating, cele fara rating avand 0. Nu se vor lua în considerare video-urile care nu au primit rating.
- **Favorite** - primele N video-uri sortate după numărul de apariții în listele de video-uri favorite ale utilizatorilor
- **Longest** - primele N video-uri sortate după durata lor. Pentru seriale se consideră suma duratelor sezoanelor.
- **Most Viewed** - primele N video-uri sortate după numărul de vizualizări. Fiecare utilizator are și o structură de date în care ține cont de câte ori a văzut un video. În cazul sezoanelor se ia întregul serial ca și număr de vizualizări, nu independent pe sezoane.

Pentru utilizatori:

- **Number of ratings** - primii N utilizatori sortați după numărul de ratings pe care le-au dat (practic cei mai activi utilizatori)

Precizare: Daca numarul de raspunsuri este mai mic decat N, atunci se vor returna toate. Rezultatul este de forma: "Query result: [X]", unde X este o lista de elemente, care poate sa contina 0 elemente!

Recomandările

Acestea reprezintă căutări după video-uri ale utilizatorilor. Ele sunt particularizate pe baza profilului acestora și au la bază 5 strategii.

Strategiile de recomandare:
Pentru toți utilizatorii:

- **Standard** - întoarce primul video nevăzut de utilizator din baza de date, neexistand al doilea criteriu.
- **Best unseen** - întoarce cel mai bun video nevizualizat de acel utilizator. Toate video-urile sunt ordonate descrescător după rating și se alege primul din ele, al doilea criteriu fiind ordinea de aparitie din baza de date.

Doar pentru utilizatorii *premium*:

- **Popular** - întoarce primul video nevizualizat din cel mai popular gen (video-urile se parcurg conform oridinii din baza de date). Popularitatea genului se calculează din numărul de vizualizări totale ale videoclipurile de acel gen. In cazul in care toate videoclipurile din cel mai popular gen sunt vizionate de utilizator, atunci se trece la urmatorul gen cel mai popular. Procesul se reia pana se gaseste primul videoclip care nu a fost vizionat din baza de date.
- **Favorite** - întoarce videoclipul care e cel mai des intalnit in lista de favorite (care să nu fie văzut de către utilizatorul pentru care se aplică recomandarea) a tuturor utilizatorilor, al doilea criteriu fiind ordinea de aparitie din baza de date. Atentie! Videoclipul trebuie sa existe in cel putin o lista de videoclipuri favorite ale userilor.
- **Search** - toate videoclipurile nevăzute de user dintr-un anumit gen, dat ca filtru în input. Sortate este in ordine crescatoare după rating, al doilea criteriu fiind numele.

În cadrul recomandărilor (fără Search Recommendation), al doilea criteriu de sortare este ordinea din baza de date (adică ordinea de apariţie în câmpul "movies"/"shows" din database).

Doar în cadrul Search Recommendation al doilea criteriu de sortare este numele.

Atunci când nu se poate întoarce niciun video se afiseaza "XRecommendation cannot be applied!", altfel "XRecommendation result: ", unde X este numele strategiei.

Detalii pentru implementare

Structura orientativa

Pentru un design al temei mai clar și mai corect, vă recomandăm o organizare cât mai logică a claselor și a pachetelor. Pentru a vă oferi o imagine de ansamblu a execuției temei, este important sa înțelegeți că orice acțiune are un efect asupra repository-ului și al workflow-ului programului. De exemplu, o acțiune executata de un utilizator poate afecta interpretarea și comportamentul viitoarelor acțiuni/rezultate.

Scheletul de cod

În rezolvarea temei va fi nevoie de folosirea unor obiecte pentru stocarea și maparea datelor primite în format JSON. Scheletul temei vă oferă mai multe clase și enum-uri utile pentru rularea temei, citirea și parsarea testelor. Acestea se regăsesc în cadrul scheletului astfel:

- Clase de tip Enum : `ActorsAwards`, `Genre`. Acestea conțin informații despre tipurile de premii, respectiv toate genurile de videouri posibile.
- Clasele de input din cadrul pachetului `fileio` : Acestea se vor folosi pentru parsarea datelor de input. Astfel preluați toate informațiile necesare pentru a crea propriile clase pentru rezolvarea temei.
- Clasa `Test` în interiorul căreia se poate testa individual rezolvarea, pe un anumit fișier de input.
- Clasa `Main` care este punctul de intrare pentru logica de rezolvare a temei.
- In cazul unor warning-uri de tip unchecked este recomandat sa puneti supress.

Pentru a putea scrie în fişier se pot folosi metodele din clasa Writer:

- closeFile - scrie în fişier toate datele din JSONArray-ul dat ca parametru şi închide fişierul
- writeFile populează un JSONArray cu date după formatul de input. Id este id-ul acţiunii şi message este mesajul rezultat în cadrul acţiunii id.

Pentru a putea scrie în fişier este nevoie de popularea unui singur JSONArray.

Scheletul conține și un fișier **README** cu detalii despre rulare și fișierele oferite.

Exemple date de intrare

Entități

Utilizator

Click pentru exemplu

Actor

Click pentru exemplu

Video - film

Click pentru exemplu

Video - serial

Click pentru exemplu

Comenzi

Adaugă videoul *The Circle* în lista de **videouri favorite** ale utilizatorului *madUnicor3*

Click pentru exemplu de comanda favorite

Adaugă videoul *False identidad* în lista de **videouri vizualizate** ale utilizatorului *madUnicorn3*

Click pentru exemplu de comanda view

Adaugă rating 9 pentru sezonul 1 al show-ului *The Haunting of Hill House* în **lista de ratings** pentru show-uri ale utilizatorului kindLocust2

Click pentru exemplu de comanda rating

Queries

Oferă primii *17* **actori** sortați în ordine crescătoare în funcție de **media** acestora.

Click pentru exemplu

Oferă o listă cu **actorii** ce au câștigat **premiile** *BEST_PERFORMANCE* și *BEST_SUPPORTING_ACTOR* sortate în ordine descrescătoare.

Click pentru exemplu

Oferă o listă cu **actorii** ce au în **descriere** cuvintele cheie *actress* și *producer* sortați în ordine crescătoare.

Click pentru exemplu

Întoarce o listă cu primele 6 **filme** ce sunt din anul *2019* și au genul *Action* sortate descrescător după **raiting**.

Click pentru exemplu

Întoarce o listă cu primele 9 **filme** favorite din anul *1994* cu genul *Romance* sortate descrescător după numărul de apariții în lista de filme **favorite** ale utilizatorilor.

Click pentru exemplu

Întoarce o listă cu primele 8 **filme** din anul *2017* cu genul *Comedy* sortate crescător după **lungimea** lor.

Click pentru exemplu

Întoarce o listă cu primele 7 **filme** din anul *2017* cu genul *Drama* sortate descrescător după **numărul de vizualizări**.

Click pentru exemplu

Întoarce o listă cu primii 10 **utilizatori** sortați crescător după **numărul de ratings** pe care le-au dat.

Click pentru exemplu

Recomandări

Întoarce primul **video nevăzut** de utilizatorul lyingRice4.

Click pentru exemplu

Întoarce **cel mai bun** video nevizualizat de utilizatorul solidEagle6.

Click pentru exemplu

Întoarce **video-ul** cel mai vizionat din **cel mai popular** gen pentru utilizatorul culturedCurlew5.

Click pentru exemplu

Întoarce **filmul** care e **cel mai des întâlnit** în lista de favorite pentru utilizatorul aboardMackerel5.

Click pentru exemplu
Întoarce o listă cu toate **filmele nevăzute** de utilizatorul finickyZebra8 din genul *Action & Adventure*.
Click pentru exemplu

Diverse detalii

- Rating - e un double. Folosiți Double.compare pt comparare
- Se vor realiza obiecte custom pentru fiecare entitate necesară, toate datele fiind parsate de clasele ce se regăsesc în pachetul fileio. (ex. ActionInputData)
- Am folosit termenul "lista de " pentru filmele vizualizate, favorite etc. Este la latitudinea voastra însa ce structură de date alegeți sa folosiți pentru a ține aceste date. Mai multe detalii despre structurile de date din Java găsiți în laboratorul Colectii
- În cazul claselor în care se reține inputul parsat, acestea pot fi modificate, însa noi recomandăm crearea altor clase, pentru a nu strica citirea datelor de intrare din greșeala. Ramane la latitudinea voastă cum implementați!

Rulare

Rularea temei se va realiza exclusiv prin intermediul clasei Main, de acolo fiind apelat și checkstyle-ul. Toate testele se regăsesc în directorul **test_db** din cadrul scheletului temei.

Evaluare

Punctajul constă din:

- 80p implementare - trecerea testelor
- 10p coding style (vezi checkstyle)
- 5p README clar, concis, explicații axate pe design (flow, interacțiuni)
- 5p folosire git pentru versionarea temei

Pe pagina Indicații pentru teme găsiți indicații despre scrierea readme-ului și depunctările generale pentru teme

Depunctarile pentru **designul și organizarea codului** se vor scădea din punctajul testelor. Dacă vor apărea depunctari specifice temei în momentul evaluării, nemenționate pe pagina cu depunctări generale, ele se vor încadra în limitele de maxim 15 pentru design, 5p pentru readme. Dacă tema nu respecta cerințele, sau are zero design OOP atunci se pot face depunctari suplimentare.
Folosirea git pentru versionare va fi verificata din folderul .git pe care trebuie să îl includeți în arhiva temei. Punctajul se va acorda dacă ați făcut minim 3 commit-uri relevante și cu mesaj sugestiv. **NU** este permis să aveți repository-urile de git publice până la deadline-ul hard.
Teste temei au punctaj diferentiat pe dificultatea lor:

- 21 teste de tip comanda unica - 2p fiecare

- 5 teste de fail - 4 au 2p, 1 are 3p
- 9 teste complexe - 3p fiecare

Bonusuri: La evaluare, putem oferi bonusuri pentru design foarte bun, cod bine documentat dar și pentru diverse elemente suplimentare alese de voi.

- Pentru a vă ușura munca și pentru un cod cât mai lizibil, vă recomandăm utilizarea stream-urilor și a design pattern-urilor. (De asemenea, acestea vă pot scăpa de bucăți lungi și repetitive de cod.)

Temele vor fi testate împotriva plagiatului, **inclusiv sursele din anii trecuți**. Orice tentativă de copiere va duce la **anularea punctajului** de pe parcursul semestrului şi **repetarea materiei** atât pentru sursă(e) cât şi pentru destinație(ii), fără excepție.

Checkstyle

Unul din obiectivele temei este învățarea respectării code-style-ului limbajului pe care îl folosiți. Aceste convenții (de exemplu cum numiți fișierele, clasele, variabilele, cum indentați) sunt verificate pentru temă de către tool-ul checkstyle.
Pe pagina de Recomandări cod găsiți câteva exemple de coding style.
Dacă numărul de erori depistate de checkstyle depășește 30, atunci punctele pentru coding-style nu vor fi acordate. Dacă punctajul este negativ, *acesta se trunchiază la 0*.
Exemple:

- `punctaj_total = 100` și `nr_erori = 200` ⇒ `nota_finala = 90`
- `punctaj_total = 100` și `nr_erori = 29` ⇒ `nota_finala = 100`
- `punctaj_total = 80` și `nr_erori = 30` ⇒ `nota_finala = 80`
- `punctaj_total = 80` și `nr_erori = 31` ⇒ `nota_finala = 70`

Upload temă

Arhiva pe care o veţi urca pe VMChecker va trebui să conţină în directorul rădăcină:

- fișierul `README`
- folder-ul src cu pachetele și cu fișierele .java
- folderul .git

Resurse și linkuri utile

- Schelet de cod
- Indicații pentru teme
- Recomandări coding style & javadoc