# Proiect - Etapa 2 - POO TV

- **Data publicării:** 23.12.2022, 00:10
- **Data ultimei modificări:** 23.12.2022, 00:10
- **Deadline HARD:** 16.01.2023, 23:59
- **Responsabili:** Mihail Ungureanu [mailto:mihai2000.ungureanu@gmail.com], Sorina Anamaria Buf [mailto:sorina_anamaria.buf@stud.acs.upb.ro], Narcis-Florin Căroi [mailto:narciscaroi24@gmail.com], Mihnea-Andrei Bloțiu [mailto:mihnea.blotiu@stud.acs.upb.ro], Mihai Soare [mailto:mihai_daniel.soare@stud.acs.upb.ro], Eduard Marin [mailto:eduard.marin@stud.acs.upb.ro]
- **Repository schelet:** Github [https://github.com/oop-pub/oop-asignments/tree/master/proiect2]
- **Actualizări**:

# Obiective

- dezvoltarea unor abilități de bază de organizare și design orientat obiect;
- proiectarea unui design mentenabil, extensibil și reutilizabil, respectând astfel principalele obiective ale limbajelor OOP;
- înțelegerea scopului și contextului de utilizare ale design pattern-urilor studiate în cadrul laboratorului (și nu numai) în vederea alegerii și implementării pattern-ului potrivit pentru rezolvarea unei probleme reale;
- adăugarea de noi funcționalități, având la bază un codebase deja existent;
- respectarea unui stil de programare și de documentare a codului, în unitate cu practicile de codare folosite în prima etapă a aplicației.

# Etapa 1 Rewind

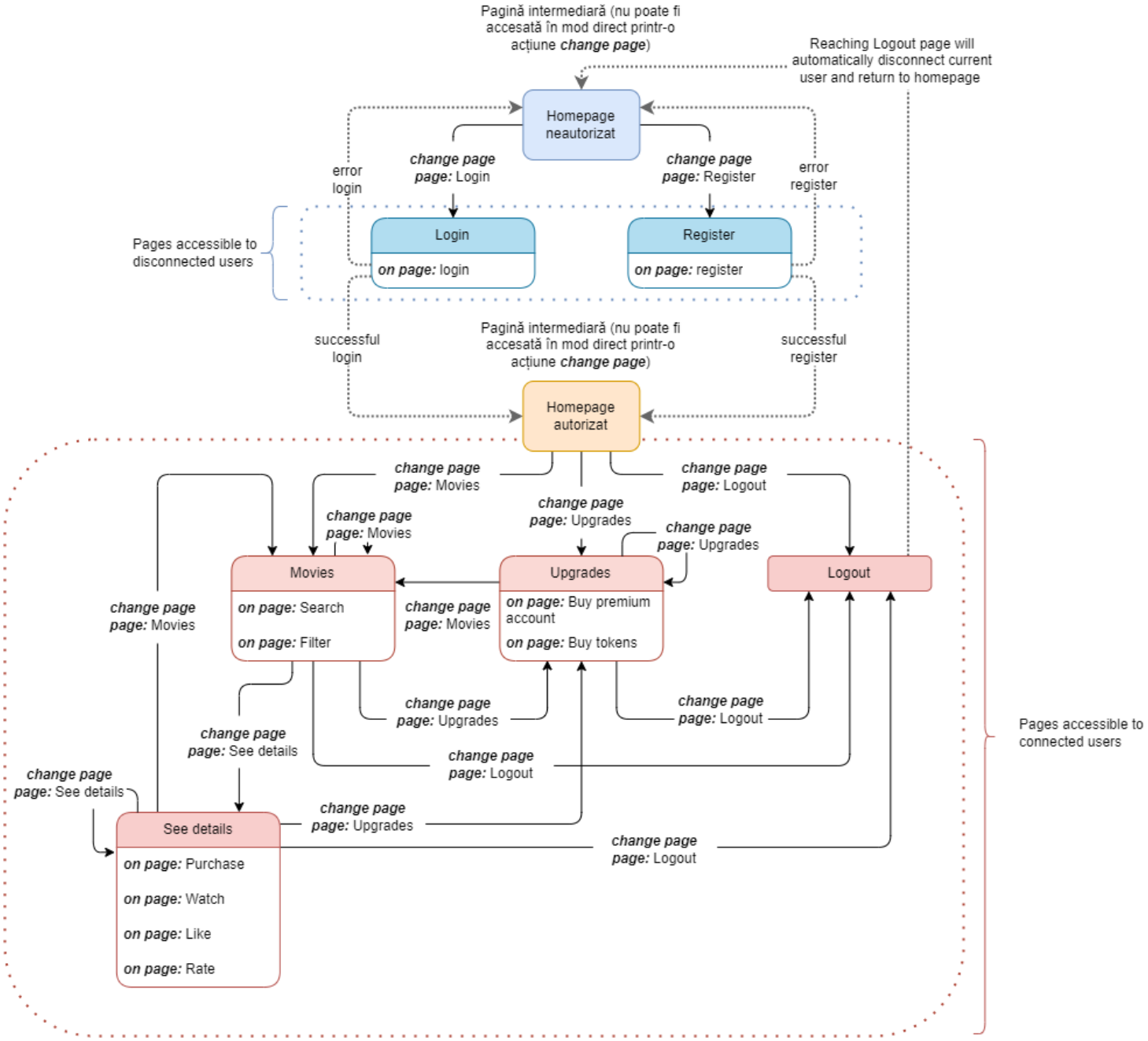
Înainte de a merge mai departe cu funcționalitățile introduse de această etapă, dorim să facem o scurtă recapitulare asupra primului stagiu al proiectului și să clarificăm astfel anumite aspecte, pentru a nu întâmpina dificultăți în etapa curentă.

Pentru rezolvarea **etapei 2** a proiectului, trebuie să aveți finalizate toate funcționalitățile descrise în cadrul **etapei 1**.

Noțiunile care vor fi recapitulate în cadrul acestei secțiuni **NU vor acoperi toate detaliile de care aveți nevoie pentru rezolvarea funcționalităților din etapa 1**. Această recapitulare are simplul rol de a clarifica anumite noțiuni care au prezentat dificultăți de rezolvare/înțelegere, în urma feedback-ului vostru pentru etapa precedentă. **Pentru finalizarea sau rezolvarea etapei 1 a proiectului, vă rugăm consultați pagina dedicată acestei teme**.

## Structura site-ului POO TV

În cadrul primei etape, ați avut de implementat structura platformei noastre de streaming video, care a constat în definirea paginilor care pot fi accesate de un utilizator și flow-ul de accesare al acestora.




Am introdus schema precedentă pentru a avea o imagine și mai clară asupra paginilor existente pe site-ul nostru, modul în care acestea pot fi accesate (în urma acțiunilor de tipul **change page**) precum și acțiunile care pot fi realizate în cadrul fiecărei pagini. (acțiunile **on page**).

Schema prezintă doar o recapitulare a flow-ului **deja existent** în urma rezolvării etapei 1, fără introducerea noilor funcționalități impuse de etapa 2. Funcționalitățile etapei 2 vor fi prezentate după încheierea acestei secțiuni.

Schema **NU** reprezintă o ierarhie de moștenire implicită a paginilor, ci modul în care acestea pot fi accesate.

Legenda schemei:

- chenarele reprezintă paginile site-ului POO TV, cu acțiunile **on page** realizabile în cadrul fiecărei pagini;
- săgețile care nu sunt punctate, reprezintă acțiunile posibile de navigare explicită de pe o pagină pe o altă pagină din cadrul platformei, în urma executării unei comenzi **change page**;
- săgețile punctate, în schimb, arată tot navigări între pagini, dar care nu sunt declanșate în mod explicit printr-o acțiune change page (condițiile trecerilor implicite dintre pagini sunt explicate în cadrul schemei).
- există și amănunte care nu au putut fi surprinde în cadrul schemei, cum ar fi constrângerile de execuție ale acțiunilor Purchase/Watch/Like/Rate de pe pagina Movies, deci, pentru mai multe detalii legate de flow-ul existent al aplicației

vă reamintim să consultați pagina dedicată etapei 1.

Orice nerespectare a schemei (încercarea de navigare pe o pagină unde nu putem ajunge din locul în care ne aflăm sau de executare a unei acțiuni nepermise paginii unde suntem) va genera afișarea unei erori la output.

# Lămuriri format input

## Acțiuni change page

Se păstrează același format de input explicat în cadrul etapei anterioare, deci puteți vedea exemplele din cadrul paginii dedicate etapei 1.

## Acțiuni on page

În cazul acțiunilor de tipul on page, a existat o neconcordanță între formatul exemplificat pe ocw și cel din teste (ne cerem scuze pentru asta). Clarificăm și aici, că pentru o acțiune de tipul **on page** formatul de input **NU** va conține un field **page**, care să specifice pagina pe care se realizează acțiunea. Este de datoria voastră să vă salvați intern pagina pe care vă aflați și ce acțiuni aveți voie să executați în cadrul acesteia.

# Lămuriri format output

Formatul de output se păstrează și puteți vedea explicațiile integrale asupra conținutului acestuia pe pagina dedicată etapei precedente. În această secțiune vrem însă să revizuim ce tip de acțiuni generează scrierea de conținut la output.

Erorile generate de acțiuni nepermise se vor afișa indiferent dacă, în cazul execuției valide, tipul acțiunii care a generat eroarea nu scria la output.

Explicațiile care urmează au la baza formatul de input/output descris în cadrul etapei 1.

## Acțiuni change page

Nu toate schimbările de pagină generează scrierea la output. Cele care generează sunt:

- **change page** către pagina **Movies**

→ output-ul va reflecta filmele accesibile utilizatorului ajuns pe pagina *Movies*, în cadrul câmpului **currentMoviesList**

- **change page** către pagina **See details**

→ output-ul va reflecta filmul accesat prin pagina *See details*, în cadrul câmpului **currentMoviesList** (o listă care va conține doar filmul a cărei pagini a fost accesată)

## Acțiuni on page

Nu toate acțiunile realizabile pe o pagină generează scrierea la output. Cele care generează sunt următoarele:

### Pe pagina Login

- **login**

→ output-ul va reflecta utilizatorul nou conectat, în cadrul câmpului **currentUser**

### Pe pagina Register

- **register**

→ output-ul va reflecta utilizatorul nou înregistrat și conectat, în cadrul câmpului **currentUser**

### Pe pagina Movies

- **search**

→ output-ul va reflecta lista curentă a filmelor de pe pagina *Movies*, în urma filtrării executate de acțiunea search (ne amintim că acțiunea search este o comandă de filtrare în care condiția este ca titlul filmelor să înceapă obligatoriu cu string-ul oferit în câmpul de input **startsWith**), în cadrul câmpului **currentMoviesList**

- **filter**

→ output-ul va reflecta lista curentă a filmelor de pe pagina *Movies*, în urma filtrării executate de acțiunea filter (pe baza filtrelor primite la input), în cadrul câmpului **currentMoviesList**

Dacă acțiunile de căutare sau filtrare a filmelor nu generează găsirea vreunui film care să se potrivească filtrelor introduse, înseamnă că pe pagina Movies nu se mai afișează niciun film și implicit field-ul **currentMoviesList** va conține o listă goală.

### Pe pagina See details

- **purchase**

→ output-ul va reflecta filmul curent pe care ne aflăm, în cadrul câmpului **currentMoviesList** (o listă care va conține doar filmul a cărei pagini a fost accesată) și cumpărarea acestuia, fiind apendat la lista **purchasedMovies** din obiectul utilizatorului curent, **currentUser**

- **watch**

→ output-ul va reflecta filmul curent pe care ne aflăm, în cadrul câmpului **currentMoviesList** (o listă care va conține doar filmul a cărei pagini a fost accesată) și vizionarea acestuia, fiind apendat la lista **watchedMovies** din obiectul utilizatorului curent, **currentUser**

- **like**

→ output-ul va reflecta filmul curent pe care ne aflăm, în cadrul câmpului **currentMoviesList** (o listă care va conține doar filmul a cărei pagini a fost accesată) și aprecierea acestuia, fiind apendat la lista **likedMovies** din obiectul utilizatorului curent, **currentUser**

- **rate**

→ output-ul va reflecta filmul curent pe care ne aflăm, în cadrul câmpului **currentMoviesList** (o listă care va conține doar filmul a cărei pagini a fost accesată) și oferirea de rating acestuia, fiind apendat la lista **ratedMovies** din obiectul utilizatorului curent, **currentUser**

## Going forward To Etapa 2

Acum că avem toate detaliile platformei noastre clarificate (hopefully), putem sa ne apucăm de implementarea noilor funcționalități de care depinde etapa curentă de îmbunătățire a aplicației.

## Modificare output

Noile acțiuni pe care le vom introduce în această etapă vor presupune o mică modificare realizată asupra formatului de output, mai exact, introducerea unui nou câmp în cadrul obiectului **currentUser**.

```
{
  "error" : null,
  "currentMoviesList" : [ ],
  "currentUser" : {
    "credentials" : {
      "name" : "Eduard",
      "password" : "secret",
      "accountType" : "standard",
      "country" : "Romania",
```

```
      "balance" : "50"
    },
    "tokensCount" : 0,
    "numFreePremiumMovies" : 15,
    "purchasedMovies" : [ ],
    "watchedMovies" : [ ],
    "likedMovies" : [ ],
    "ratedMovies" : [ ],
    "notifications": []
  }
}
```

După cum puteți observa în exemplul anterior, singura modificare care apare este adăugarea unui array **notifications**, în cadrul obiectului **currentUser**. Restul field-urilor rămân neschimbate și își păstrează însemnătatea și rolul din prima etapă a proiectului.

Popularea câmpului **notifications** va fi explicată în cadrul fiecărei acțiuni ce necesită modificarea acestuia. În principiu, fiecare user va trebui să stocheze o coadă de notificări ce va fi populată de-a lungul execuției, în urma anumitor acțiuni efectuate.

Coada de notificări se va prezerva pentru un utilizator de-a lungul execuției întregului program, inclusiv între logări diferite ale acestuia. Asta înseamnă că reconectarea unui user nu va genera golirea listei de notificări, iar noile notificări se vor adăuga la cele deja existente.

## Movies Database

I've seen it all… asta a fost principala plângere din partea utilizatorilor site-ului vostru, care, într-o seară de procrastinare a temei la SO, au vizualizat toate filmele din baza voastră de date și au făcut switch rapid pe NETFLIX, că deadline-ul la temă e oricum departe.

Ca să-i aduceți înapoi pe site, deschideți The Pirate Bay și uploadați pe POO TV toate filmele Marvel pe care le găsiți. Ba mai mult, ca să nu supraîncărcați baza de date, vreți să aveți și posibilitatea ca ocazional să ștergeți filmele mai puțin populare.

Nu în ultimul rând, ca să vă țineți consumatorii mulțumiți, le oferiți acestora posibilitatea de abonare gratuită la anumite movie topics și implicit de a primi notificări în momentul update-urilor efectuate asupra bazei de filme.

### Subscribe

În această etapă, vom adăuga un nou tip de acțiune, **subscribe**, prin care utilizatorul curent conectat va avea posibilitatea de a se abona la un anumit gen de filme.

Această acțiune va putea fi efectuată doar aflându-ne pe pagina **see details** a unui film. Un utilizator **va putea da subscribe doar la genurile filmului pe pagina căruia se află** (de exemplu, dacă am navigat pe pagina see details a filmului "Titanic", utilizatorul poate da subscribe doar la genurile filmului "Titanic", aflându-se pe această pagină). Încercarea de a efectua o acțiune subscribe pe o altă pagină decât see details sau la un alt gen de film decât cele reprezentative filmului pe care l-am accesat va genera afișarea unei erori la output (și implicit, subscripția nu va fi efectuată).

Această comandă **NU** va modifica pagina pe care ne aflăm.

```
{
  "type": "subscribe",
  "subscribedGenre": "Action"
}
```

Inputul acestei comenzi va conține numele acțiunii, **"subscribe"**, și genul la care se abonează utilizatorul. Un utilizator se poate abona la un singur gen de filme printr-o acțiune de subscribe.

Deși o acțiune de subscribe generează abonarea la un singur topic, un utilizator poate fi abonat în același timp la mai multe genuri de filme în urma execuției mai multor comenzi de tipul subscribe.

Încercarea de abonare la un topic la care utilizatorul este deja abonat va genera o eroare.

Dacă nu există erori, nu se va afișa nimic la output după efectuarea acestui tip de comandă, fiind de datoria voastră să vă salvați intern categoriile cinematografice la care se abonează utilizatorii.

Subscripțiile unui user se păstrează între conectări diferite. Adică, dacă un user s-a logat/înregistrat pe platformă, în timpul sesiunii pe site s-a abonat la un număr X de topicuri, într-o următoare conectare a utilizatorului, cele X subscripții se păstrează, la care se vor mai putea adăuga abonările realizate în timpul sesiunii curente și așa mai departe. Pe scurt, pentru fiecare utilizator, subscripțiile se păstrează între sesiuni diferite de conectare.

## Database

Acum, ne dorim să putem modifica baza noastră de filme, pentru a avea mereu un conținut up to date, care să nu ne plictisească utilizatorii.

Pentru a realiza acest lucru, vom adăuga încă un tip de acțiune, **database**, care, **va putea fi efectuată indiferent de pagina pe care ne aflăm**.

Fiind o acțiune care afectează baza de date de filme, nu utilizatorii, va putea fi executată indiferent de pagina pe care ne aflăm și implicit indiferent de existența unui utilizator conectat (practic, în orice moment al execuției acțiunilor administratorul site-ului va putea modifica baza de filme).

Această comandă **NU** va modifica pagina pe care ne aflăm.

Acest tip de comanda va avea două posibile features: **add** și **delete**.

### Database add

În momentul în care se execută o acțiune de tipul **database add**, un nou film va fi adăugat în baza de date. Puteți privi acțiunea ca fiind executată de un administrator extern al bazei de date, care nu va afecta pagina pe care sunteți.

De asemenea, prin intermediul acestei comenzi, utilizatorii care sunt abonați (prin intermediul acțiunii **subscribe** explicate anterior) la cel puțin unul dintre genurile filmului nou adăugat vor fi notificați de adăugare.

```
{
  "type": "database",
  "feature": "add",
  "addedMovie": {
    "name": "Matrix 2",
    "year": "1998",
    "duration": 131,
    "genres": [
      "Action",
      "Comedy",
      "Crime"
    ],
    "actors": [
      "Laurence Fishburne",
      "Halle Berry",
      "Keanu Reeves"
    ],
    "countriesBanned": [
      "Ireland"
    ]
  }
}
```

Inputul pentru acest tip de comandă va conține tipul acțiuni (field-ul **type** cu valoarea **database**) și tipul operației efectuate asupra bazei de date (field-ul **feature** cu valoarea **add**). Obiectul **addedMovie** va conține toate field-urile de descriere ale noului film adăugat.

Încercarea de adăugare a unui film deja existent în baza de date va genera afișarea unui erori la output. Verificarea existenței unui film în baza de date se va face pe baza **numelui filmului**. Este suficient ca două filme să aibă același nume pentru a fi considerate echivalente.

Acțiunea database add va adăuga un singur film în baza de date (implicit, va primi un singur film în input). Se pot adăuga mai multe filme în baza de date, în urma efectuării mai multor acțiuni de tipul **database add**.

În momentul în care se realizează o acțiune database add, **TOȚI** utilizatorii care sunt abonați la cel puțin unul dintre genurile filmului, și care nu se află într-una dintre țările banate ale movie-ului, vor fi notificați de adăugarea noului film, și implicit își vor apendui la o coadă de notificări acest eveniment.

Vor fi notificați **TOȚI** utilizatorii care respectă constrângerile menționate mai sus, indiferent dacă aceștia sunt conectați sau nu.

```
"notifications" : [ {
    "movieName" : "Matrix 2",
    "message" : "ADD"
} ]
```

Notificarea de adăugare a unui film este caracterizată prin numele filmului (field-ul **movieName**), și tipul acțiunii, **ADD**, care s-a efectuat (field-ul **message**).

Reamintim, constrângerile de țară banată se păstrează, astfel că vor fi notificați doar acei utilizatori care sunt abonați la unul dintre genurile filmului, și care **nu se află într-una dintre țările banate ale filmului**.

Dacă nu există erori, nu se va afișa nimic la output după efectuarea acestui tip de comandă, fiind de datoria voastră să actualizați baza de date și să notificați în mod corect utilizatorii (și implicit să adăugați la coada lor de notificări evenimentul).

## Database delete

Acțiunea **database delete** va reprezenta inversul comenzii **database add** (so shocking, I know) și va efectua ștergerea unui film din baza de date. La fel ca în cazul acțiunii precedente, puteți să o vedeți ca fiind executată de un administrator extern al bazei de date, care nu va afecta pagina pe care vă aflați.

Utilizatorii care au achiziționat filmul care urmează a fi șters vor fi notificați de ștergerea acestuia și vor primi înapoi resursele folosite pentru cumpărarea filmului.

Dacă contul curent al utilizatorului este **premium**, i se va restitui acestuia un film gratuit, incrementându-se astfel numărul lui de premium free movies (nu se va ține cont dacă utilizatorul cu cont premium a cumpărat într-adevăr filmul șters folosind tokeni sau folosindu-și unul dintre filmele lui premium gratuite). În schimb, userii **standard** vor primi înapoi valoarea în tokeni a unui film (egală cu **2 tokeni**).

Orice utilizator care a achiziționat un film care va fi șters trebuie să îl elimine atât din lista filmelor pe care le-a cumpărat, cât și din celelalte liste care reflectă operații ale userului pe filmul respectiv (watch, like, rate). Practic, un film care a fost șters nu mai poate apărea nici în baza de date, nici în listele de filme purchased/watched/liked/rated ale utilizatorilor.

```
{
  "type": "database",
  "feature": "delete",
  "deletedMovie": "Titanic"
}
```

Inputul pentru acest tip de comandă va conține tipul acțiunii (field-ul **type** cu valoarea **database**) și tipul operației efectuate asupra bazei de date (field-ul **feature** cu valoarea **delete**). Obiectul **deletedMovie** va conține numele filmului care trebuie șters.

Încercarea de ștergere a unui film inexistent în baza de date va genera afișarea unui erori la output.

Acțiunea database delete va șterge un singur film din baza de date (implicit, va primi un singur film în input). Se pot șterge mai multe filme din baza de date, în urma efectuării mai multor acțiuni de tipul **database delete**.

În momentul în care se realizează o acțiune database delete, **TOȚI** utilizatorii care au cumpărat filmul respectiv vor fi notificați de ștergere și implicit își vor apendui la o coadă de notificări acest eveniment.

Vor fi notificați **TOȚI** utilizatorii care respectă constrângerile menționate mai sus, indiferent dacă aceștia sunt conectați sau nu.

```
"notifications" : [ {
    "movieName" : "Titanic",
```

```
    "message" : "DELETE"
} ]
```

Notificarea de ștergere a unui film este caracterizată prin numele filmului (field-ul **movieName**), și tipul acțiunii, **DELETE**, care s-a efectuat (field-ul **message**).

Dacă nu există erori, nu se va afișa nimic la output după efectuarea acestui tip de comandă, fiind de datoria voastră să actualizați baza de date și să notificați în mod corect utilizatorii (și implicit să adăugați la coada lor de notificări evenimentul).

## Acțiunea de back între pagini

Ne propunem să adăugăm un nou element de UI pe fiecare pagină a site-ului nostru, butonul de **back**. Să navighezi pe un site fără posibilitatea de a te întoarce pe pagina anterioară poate fi destul de frustrant, nu-i așa? De aceea, am decis să includem acest feature și în cadrul aplicației noastre.

Această acțiune poate fi văzută ca inversul acțiunii de tip **change page**, dar în loc să navigați către următoarea pagină primită în input, vă întoarceți pe pagina pe care v-ați aflat înainte de efectuarea ultimei comenzi de tip change page.

```
{
  "type": "back"
}
```

În cazul acestei acțiuni nu există un exemplu propriu-zis de output, întrucât întoarcerea pe o anumită pagină va genera același output ca o acțiune de tipul **change page** către pagina respectivă.

Daca ajungeți pe pagina **homepage** în urma unei acțiuni de back, nu este necesar să afișați ceva la output, întrucât am considerat că nu putem realiza o acțiune change page explicită către acestă pagină (este suficient să salvați intern faptul ca vă aflați pe pagina homepage).

Putem privi navigarea între pagini ca pe o stivă în care introducem paginile în care am navigat cu succes. În momentul în care realizăm o acțiune de **back**, simulăm o acțiune de tipul **change page** validă către ultima pagină scoasă din stivă.

Acțiunea de back este specifică doar unui **user logat**, altfel va genera o eroare. Astfel, începerea populării stivei de pagini se va face doar în momentul conectării utilizatorului.

Întoarcerea pe pagina de **login/register** nu generează deconectarea userului curent. Cum doar un user deconectat are access la aceste pagini, o acțiune de back efectuată către paginile de **login/register** va genera o eroare.

Dacă nu mai există pagini în stivă, comanda întoarce eroare, iar acțiunea de back nu se efectuează (rămâneți pe aceeași pagină).

Pe lângă întoarcerea pe pagina de Homepage neautentificat și deconectarea user-ului curent, acțiunea de **logout** generează golirea stivei de pagini. Orice acțiune de back efectuată după un logout fără reconectarea implicită a unui user va genera o eroare.

## Recomandări user premium

Pentru a reaprinde vânzările abonamentelor, în această etapă le oferim utilizatorilor premium și mai multe beneficii, sub forma de recomandări personalizate.

La finalul șirului de acțiuni, utilizatorul premium conectat va primi o recomandare de film, pe baza genurilor filmelor cele mai apreciate de acesta.

Aceste recomandări nu vor funcția ca o nouă acțiune primită la input, ci, va fi de datoria voastră ca la finalul execuției tuturor acțiunilor din input, să afișați la output recomandarea de film pentru utilizatorul premium conectat.

Recomandările sunt specifice doar **userilor premium conectați**. Dacă la finalul execuției acțiunilor nu există un user conectat sau acesta nu este premium, nu se va oferi nicio recomandare și, implicit, nu se va afișa nimic la output.

Algoritm de generare recomandare:

- realizarea unui top al genurilor apreciate de către utilizator, prin sortarea în mod descrescător a acestora în funcție de numărul de like-uri;

Numărul de like-uri al unui gen de film se calculează ca fiind numărul filmelor apreciate de utilizatorul conectat care fac parte din categoria genului respectiv.

Un gen de film trebuie să aibă cel puțin un like pentru a face parte din topul genurilor apreciate de utilizator (nu vom lua în calcul mai departe genurile care nu au primit nicio apreciere din partea utilizatorului).

În caz de egalitate a numărului de like-uri, al doilea criteriu de sortare va fi sortarea lexicografică ascendentă după numele genurilor.

- sortarea filmelor din baza de date descrescător în funcție de numărul total de like-uri;

Ne referim aici la toate filmele **vizibile** utilizatorului pe pagina *Movies* (fără aplicarea vreunei sortări sau filtrări pe acestea).

În caz de egalitate, se păstrează ordinea apariției filmelor în baza de date (dacă două sau mai multe filme au același număr de like-uri, ordinea apariției lor în lista sortată va fi dată de ordinea apariției în lista originală).

- găsirea primului film din baza de date cu cel mai mare număr de like-uri care nu a mai fost văzut de către utilizator și care face parte din genul de film cel mai apreciat de acesta;

- dacă nu există un film nevizualizat din categoria celui mai apreciat gen, se va trece la următorul cel mai apreciat gen al utilizatorului și continuăm așa până când, fie am găsit un film care îndeplinește datele problemei, fie am epuizat genurile utilizatorului.

Reamintim că **NU** există un input pentru acest tip de acțiune, recomandarea va fi realizată de voi la finalul execuției tuturor acțiunilor.


```
{
  "error" : null,
  "currentMoviesList" : null,
  "currentUser" : {
    "credentials" : {
      "name" : "Eduard",
      "password" : "secret",
      "accountType" : "premium",
      "country" : "Romania",
      "balance" : "150"
    },
    "tokensCount" : 40,
    "numFreePremiumMovies" : 14,
    "purchasedMovies" : [ {
      "name" : "Titanic",
      "year" : "2019",
      "duration" : 131,
      "genres" : [ "Action", "Thriller", "Crime" ],
      "actors" : [ "Laurence Fishburne", "Halle Berry", "Keanu Reeves" ],
      "countriesBanned" : [ "Russia" ],
      "numLikes" : 1,
      "rating" : 0.00,
      "numRatings" : 0
    } ],
    "watchedMovies" : [ {
      "name" : "Titanic",
      "year" : "2019",
      "duration" : 131,
      "genres" : [ "Action", "Thriller", "Crime" ],
      "actors" : [ "Laurence Fishburne", "Halle Berry", "Keanu Reeves" ],
      "countriesBanned" : [ "Russia" ],
      "numLikes" : 1,
      "rating" : 0.00,
      "numRatings" : 0
    } ],
    "likedMovies" : [ {
      "name" : "Titanic",
      "year" : "2019",
      "duration" : 131,
      "genres" : [ "Action", "Thriller", "Crime" ],
```

```
      "actors" : [ "Laurence Fishburne", "Halle Berry", "Keanu Reeves" ],
      "countriesBanned" : [ "Russia" ],
      "numLikes" : 1,
      "rating" : 0.00,
      "numRatings" : 0
    } ],
    "ratedMovies" : [ ],
    "notifications" : [ {
      "movieName" : "John Wick: Chapter 3 - Parabellum",
      "message" : "Recommendation"
    } ]
  }
}
```

Formatul de output nu se modifică, dar în cazul recomandărilor câmpurile **error** și **currentMoviesList** nu vor avea însemnătate, așa că le vom pune pe **null**. Modificările vor apărea în cadrul câmpului **currentUser**, mai exact în array-ul **notifications**, unde va fi adăugată recomandarea de film, caracterizată prin numele filmului (field-ul **movieName**) și mesajul de notificare, **“Recommendation”** (field-ul message).

Recomandarea filmului va fi apenduită la lista de notificări, fără să modifice datele existente în prealabil în obiect. Restul datelor referitoare la utilizator rămân nemodificate și se afișează la output ca în cazul oricărei alte acțiuni.

```
{
  "error" : null,
  "currentMoviesList" : null,
  "currentUser" : {
    "credentials" : {
      "name" : "Mihail",
      "password" : "discret",
      "accountType" : "premium",
      "country" : "Russia",
      "balance" : "1337"
    },
    "tokensCount" : 0,
    "numFreePremiumMovies" : 15,
    "purchasedMovies" : [ ],
    "watchedMovies" : [ ],
    "likedMovies" : [ ],
    "ratedMovies" : [ ],
    "notifications" : [ {
      "movieName" : "No recommendation",
      "message" : "Recommendation"
    } ]
  }
}
```

În cazul în care nu a fost găsită o recomandarea, în loc de numele filmului se va scrie string-ul **“No recommendation”**. Restul outputului rămâne neschimbat, ca în cazul exemplului anterior.

# Indicații

- Separați conceptele de sine stătătoare în clase separate, nu le îmbinați - clasele ar trebui să aibă un sigur rol.
- Adaptați agregarea și moștenirea la situație, grupați pe cât posibil informația și acțiunile comune în clase generale.
- Nu vă apucați să scrieți direct; alocați timp modelării și abstractizării, pentru că altfel vă puteți trezi cu o temă muncitorească, cu mult cod din care să nu înțelegeți prea multe și pe care să-l extindeți greu.
- Vă recomandăm să porniți implementarea de la codul scris în prima etapă a proiectului, la care să adăugați noile funcționalități, dar se poate trimite și o implementare scrisă de la zero, care să nu păstreze codul primei părți, dar care să conțină funcționalitatea primei părți.
- **Etapa a doua se poate trimite fără să fi trimis prima etapă a proiectului**, însă va fi nevoie să se implementeze și funcționalitățile primei etape pentru a putea primi punctajul total pe teste; în acest caz se va primi punctaj doar pe a

doua etapă.
- Verificați periodic această pagină, deoarece scheletul/cerința pot suferi modificări în urma unor erori din partea noastră.

## Testarea soluției

Pentru testarea soluției, rulați funcția **main** a clasei **Test**. Aceasta va rula atât testele, cât și checkstyle-ul. Pentru rularea checkerului, aveți nevoie ca proiectul vostru să aibă încărcate bibliotecile pentru citirea fișierelor JSON. Mai multe detalii aici.

## Evaluare

Punctajul constă din:

- 60p implementare - trecerea testelor
- 10p coding style (vezi checkstyle)
- 15p design și organizare (folosire design patterns)
- 10p README clar, concis, explicații axate pe design (flow, interacțiuni)
- 5p utilizare Git (minimum 3 commit-uri personale, relevante pentru flow-ul proiectului)

- Folosirea git pentru versionare va fi verificată 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.

Pentru a se oferi punctaj maxim pentru implementare este necesară folosirea a **minim 4 design pattern-uri diferite** in total pentru cele 2 etape.

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

Depunctările pentru **designul și organizarea codului** se vor scădea din punctajul testelor. Dacă vor apărea depunctări specifice temei în momentul evaluării, nemenționate pe pagina cu depunctări generale, ele se vor încadra în limitele de maxim 15 pentru design, 10p pentru README. Dacă tema nu respectă cerințele, sau nu are design OOP, atunci pot apărea depunctări suplimentare.

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

Temele vor fi testate împotriva plagiatului. 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 [https://checkstyle.sourceforge.io/].

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 = 125` și `nr_erori = 200` ⇒ `nota_finala = 115`
- `punctaj_total = 125` și `nr_erori = 29` ⇒ `nota_finala = 125`
- `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 [https://vmchecker.cs.pub.ro/ui/#POO] va trebui să conţină în directorul rădăcină:

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

## Resurse și linkuri utile

- Schelet de cod [https://github.com/oop-pub/oop-asignments/tree/master/proiect2]
- Indicații pentru teme
- Recomandări coding style & javadoc

poo-ca-cd/teme/proiect/etapa2.txt · Last modified: 2022/12/23 00:11 by sorina_anamaria.buf