Proiect GlobalWaves - Etapa 1 - Audio Player

- **Responsabili:** Andrei Oțetea [mailto:andreiotetea23@gmail.com], Costin-Andrei Vlad [mailto:costin_andrei.vlad@stud.acs.upb.ro], Andreea-Rebecca State [mailto:rebeccastate40@gmail.com]
- **Colaboratori:** Sorina-Anamaria Buf [mailto:sorinabuf@gmail.com], Ștefan Cocioran [mailto:stefancocioran@gmail.com], Florian-Luis Micu [mailto:miculuis1@gmail.com]

- **Data publicarii:** 05.11.2023, ora 23:00
- **Deadline HARD:** 25.11.2023, ora 7:59
- **Ultima modificare a cerinței:** 14.11.2023, ora 19:00 (clarificari extra next)
- **Ultima modificare a scheletului:** 07.11.2023, ora 17:27 (am adaugat un comentariu de TODO ca sa stiti de unde incepe implementarea)
- **Ultima modificare a testelor:** 14.11.2023, ora 10:30 (cateva ref-uri modificate minimal testele 16, 17)
- **Schelet:** https://github.com/oop-pub/oop-project-2023/tree/main/etapa1 [https://github.com/oop-pub/oop-project-2023/tree/main/etapa1]

- **Atentie!** Pentru partea de citire / afisare nu este necesara folosirea adnotarilor din libraria Jackson. Puteti utiliza o metoda similara cu cea prezentata de noi in schelet, si anume sa va folositi de clasa ObjectMapper. **Nu exista o metoda corecta sau gresita de a lucra cu input-ul / output-ul.** Scopul nostru in cadrul acestui proiect nu este sa va testam abilitatea de a lucra cu fisierele, ci sa va testam atat intelegerea cunostintelor de baza dobandite in cadrul cursurilor si a laboratoarelor, cat si modul de gandire intr-un limbaj de programare orientat obiect.

Obiective

- Familiarizarea cu Java și conceptele de bază ale POO.
- Fundamentarea practică a constructorilor și a agregării/moștenirii.
- Dezvoltarea unor abilități de bază de organizare și design orientat obiect.
- Scrierea unui cod cât mai generic, ce va permite ulterior adăugarea de noi funcționalități.
- Respectarea unui stil de codare și de comentare.
- Dezvoltarea aptitudinilor de depanare a problemelor în cod.

Descriere

Pentru acest proiect o să aveți de implementat o aplicație asemănătoare ca funcționalități cu Spotify, simulând diferite acțiuni făcute de utilizatori. Aceste acțiuni vor fi simulate folosind niște comenzi primite în fișierele de input. Astfel, perspectiva din care rezolvați tema este aceea a unui admin, ce percepe toate acțiunile realizate de utilizatori și poate genera diferite rapoarte legate de toți utilizatorii.

Entități

Fișier audio

Poate fi de 2 tipuri:

- melodie

- episod de podcast

În funcție de tipul de fișier, utilizatorul are mai multe sau mai puține moduri în care poate interactiona cu acesta.

```
{
     "name": "Shape of You",
     "duration": 233,
     "album": "Divide",
     "tags": [
       "#pop",
       "#mostlistenedthisyear",
       "#spotify"
     ],
     "lyrics": "The club isn't the best place to find a lover, So the bar is where I go (mm-mm)",
     "genre": "Pop",
     "releaseYear": 2017,
     "artist": "Ed Sheeran"
}
```

```
{
    "name": "Elon Musk Returns",
    "duration": 11927,
    "description": "Elon Musk, CEO of SpaceX and Tesla, returns to discuss various topics."
}
```

Colecții de fișiere audio

Sunt de 3 tipuri:

- library
- playlist
- podcast

Library-ul reprezintă totalitatea melodiilor din platforma. Toți utilizatorii platformei au acces la toată biblioteca de melodii.

Playlist-ul este o colecție de melodii creată de un utilizator. Un playlist poate fi public sau privat. Acesta este format din fișiere audio din library în funcție de preferințele utilizatorului.

Podcast-ul este o colecție formată din mai multe episoade ce au legătură între ele. Toate podcasturile vor fi specificate în fișierul de intrare, fiind accesibile în biblioteca de la începutul simularii. Episoadele sunt ordonate în funcție de poziția inițială specificată la input.

```
{
    "name": "The Joe Rogan Experience",
    "owner": "Joe Rogan",
    "episodes": [
      {
        "name": "Elon Musk Returns",
        "duration": 11927,
        "description": "Elon Musk, CEO of SpaceX and Tesla, returns to discuss various topics."
      },
      {
        "name": "Jordan Peterson",
        "duration": 9916,
        "description": "Dr. Jordan Peterson joins Joe to discuss psychology, philosophy, and current events."
      },
      {
        "name": "Neil deGrasse Tyson",
        "duration": 11276,
        "description": "Astrophysicist Neil deGrasse Tyson talks about the universe and science."
      },
      {
        "name": "Elon Musk on Mars",
        "duration": 10533,
        "description": "Elon Musk shares his vision for a human settlement on Mars."
      },
      {
        "name": "The Art of MMA",
        "duration": 8742,
        "description": "Joe Rogan and guests discuss the art and science of mixed martial arts."
```

```
        }
    ]
}
```

Search bar

Search bar-ul este folosit pentru a căuta în bibliotecă o anumită melodie sau un anumit playlist sau podcast în funcție de mai multe filtre:

- Melodii:
  - by name → se verifica dacă numele melodiei incepe cu textul specificat in filtru
  - by album → se da tot numele albmului si trebuie verificat daca melodia face parte din acel album
  - by tags → se va da o lista cu tag-uri si trebuie ca melodia sa aiba toate tag-urile specificate
  - by lyrics → se va da un cuvant sau o fraza; melodia trebuie sa contina respectivul sir ce caractere
  - by genre → se va specifica un gen muzical; se verifică dacă melodia face parte din acel gen muzical
  - by release year → e string, e.g. "<2000", ">2000", trebuie verificat dacă anul de lansare este mai mic, respectiv mai mare decat anul specificat
  - by artist → se specifică numele artistului, se verifică dacă melodia este creată de acel artist
- Playlists:
  - by name → se verifică dacă numele playlist-ului începe cu textul specificat
  - by owner → e numele unui utilizator, trebuie verificat dacă playlist-ul a fost creat de acel user
- Podcast:
  - by name → se verifică dacă numele podcast-ului începe cu textul specificat
  - by owner → e numele celui care a creat podcast-ul, trebuie verificat dacă podcast-ul a fost creat de acel creator

Pentru fiecare tip de căutare garantăm că cel puțin un câmp o să fie specificat. Ordinea în care se obțin rezultatele ține de poziția conținutului în lista cu toate acele elemente(adică în lista de melodii, în lista cu podcast-uri sau în listele cu playlist-uri în cazul fiecărui utilizator).

Music player

Pentru a rula diferite fișiere audio avem nevoie de un player. Acesta poate rula melodii din bibliotecă sau dintr-un playlist, fiind rulate pe rând. Podcasturile sunt inițial rulate de la primul episod, iar episoadele sunt rulate pe rând, iar de fiecare dată când se revine la ascultarea unui podcast, acesta va rula de la episodul și momentul din episod în care a rămas ultima oară. Episoadele sunt specificate în ordine la input.

User

Platforma noastra are mai mulți utilizatori. La aceasta etapă aceștia pot interacționa cu search bar-ul, cu player-ul și pot crea și interacționa cu playlist-uri. Fiecare utilizator are un username unic, de aceea, pentru fiecare comandă ce ține de un utilizator, o să specificăm username-ul acestuia.

```
{
    "username": "alice22",
    "age": 28,
    "city": "Los Angeles"
}
```

Timestamp

Pentru a simula caracterul real al alpicatiei, comenzile au un camp numit "timestamp", care precizeaza la ce secunda s-au executat fata de inceputul testului (momentul t0). Fiecare comandă este executată instantaneu într-un moment de timp. Timpul este comun pentru toți utilizatorii, si trece constant, indiferent de comenzile primite. Astfel, nu vom putea avea o comanda cu timestamp-ul "30" pentru "user1", urmata de o comanda cu timestamp-ul "20" pentru "user2" (o comanda cu timestampul "t" marcheaza faptul ca toti utilizatorii se afla la momentul de timp "t"). De asemenea, trebuie simulat ce se întâmplă între momentele de timp, în cazul nostru, starea în care se află player-ul utilizatorului curent (la ce track a ajuns, dacă s-a oprit rularea etc.).

Comenzi search bar

Search

Se caută în funcție de filtre o melodie, un playlist sau un podcast. Melodiile sunt căutate în bibliotecă și playlist-urile sunt accesibile doar dacă sunt ale utilizatorului care a dat comanda sau sunt publice. Se returnează o listă cu primele 5 rezultate, iar în cazul în care sunt mai puține, se returnează toate rezultatele obținute. Filtrele pot varia de la o comandă de search la alta (unele campuri pot lipsi), dar pentru fiecare comanda de search, trebuie mentionat cel putin un filtru. De asemenea, căutarea se face din perspectiva utilizatorului, ceea ce înseamnă că 2 utilizatori pot avea rezultate diferite atunci când caută ceva.

Cand se ruleaza acesta comanda, se scoate sursa care era incarcata in player. Astfel, dupa un search, indiferent de rezultatul returnat de acesta, player-ul nu mai ruleaza nimic.

```
{
    "command": "search",
    "username": "alice22",
    "timestamp": 10,
    "type": "song",
    "filters": {
      "name": "Sta"
    }
}
```

```
{
    "command": "search",
    "username": "alice22",
    "timestamp": 63,
    "type": "playlist",
    "filters": {
      "owner": "alice22"
    }
}
```

```
{
    "command": "search",
    "username": "alice22",
    "timestamp": 100,
    "type": "podcast",
    "filters": {
      "name": "The"
    }
}
```

```
{
    "command" : "search",
    "user" : "alice22",
    "timestamp" : 10,
    "message" : "Search returned 3 results",
    "results" : [ "Stairway to Heaven", "Start Me Up", "Stargazing" ]
}
```

Select

Se selectează din listă una dintre opțiunile obținute la ultimul search, dacă aceasta a generat cel puțin un rezultat. De asemenea, dacă se selectează un rezultat ce are index-ul mai mare decât ultimul rezultat, se returnează o eroare. Primul rezultat are index-ul 1.

Mesaje posibile

1. Successfully selected {track_name}.
2. Please conduct a search before making a selection.
3. The selected ID is too high.

```
{
    "command": "select",
    "username": "alice22",
    "timestamp": 15,
```

```
    "itemNumber": 1
}
```

```
{
    "command" : "select",
    "user" : "alice22",
    "timestamp" : 15,
    "message" : "Successfully selected Stairway to Heaven."
}
```

Comenzi player

Load

Această comandă se poate da imediat după ce s-a selectat o melodie/un playlist/un podcast. Aceasta rulează playlist-ul, respectiv melodia de la început, iar în cazul podcast-ului, de unde a rămas în ultimul episod rulat.

Mesaje posibile

1. Playback loaded successfully.
2. Please select a source before attempting to load.
3. You can't load an empty audio collection!

```
{
    "command": "load",
    "username": "alice22",
    "timestamp": 20
}
```

```
{
   "command" : "load",
   "user" : "alice22",
   "timestamp" : 20,
   "message" : "Playback loaded successfully."
}
```

Această comandă poate fi rulată doar după ce utilizatorul a dat comanda select.

PlayPause

Această comandă face tranziția între starea de play și pause a player-ului. Dacă player-ul se află în starea de play, atunci în urma comenzii "playPause" acesta va trece în starea de pause. Dacă player-ul se află în starea de pause, atunci în urma comenzii "playPause" acesta va trece în starea de play.

Mesaje posibile

1. Playback paused successfully.
2. Playback resumed successfully.
3. Please load a source before attempting to pause or resume playback.

```
{
    "command": "playPause",
    "username": "alice22",
    "timestamp": 30
}
```

```
{
    "command" : "playPause",
    "user" : "alice22",
    "timestamp" : 30,
```

```
    "message" : "Playback paused successfully."
}
```

Această comandă poate fi rulată doar după ce utilizatorul a dat comanda load.

Repeat

Inițial repeat este în starea în care nu se repetă nimic. În funcție de tipul conținutului pe care îl rulăm în momentul respectiv, comanda repeat ciclează prin diferite stări. Practic din starea 0 se trece in starea 1, din 1 in 2 si din 2 in 0, starea efectiva tinand de tipul continutul care este rulat de player in acel moment. Astfel avem cazurile:

- Dacă rulam ceva din playlist, avem stările:
  - 0 - no repeat
  - 1 - repeat all → după ce termină toate melodiile din listă, începe din nou cu prima melodie
  - 2 - repeat current song → după ce termină melodia curentă, o reîncepe
- Dacă rulam o melodie din bibliotecă sau un podcast, avem stările:
  - 0 - no repeat
  - 1- repeat once → după ce termină fișierul curent, îl mai rulează o data
  - 2 - repeat infinite → rulează fișierul curent la nesfârșit

Mesaje posibile

1. Repeat mode changed to {repeat_state}.
2. Please load a source before setting the repeat status.

```
{
    "command": "repeat",
    "username": "alice22",
    "timestamp": 31
}
```

```
{
    "command" : "repeat",
    "user" : "alice22",
    "timestamp" : 31,
    "message" : "Repeat mode changed to repeat once."
}
```

Această comandă poate fi rulată doar după ce utilizatorul a dat comanda load.

Shuffle

Comanda shuffle poate fi folosită **doar atunci când ascultăm un playlist**.

Când se încarcă ceva în player, acesta preia track-urile și le rulează în ordine. Pentru a rula în altă ordine, se dă comanda shuffle, care trece prin cele 2 stări. Pentru un playlist rulat în modul shuffle se stabilește ordinea în care se ruleaza melodiile din acesta folosindu-se de seed-ul specificat în comandă. Practic, dacă avem 10 melodii, o să considerăm că avem id-uri de la 0 până la 9 în aceasta ordine și le amestecăm folosindu-ne de clasa Random și de seed-ul primit. Noul vector semnifică ordinea în care urmează melodiile, dacă shuffle este activ.

Presupunand ca suntem la a k-a melodie din playlist (unde k sunt indicii melodiilor date în ordinea din input), avem 2 cazuri:

1. Dacă shuffle e dezactivat, urmează melodia k+1, în cazul în care repeat e dezactivat
2. Dacă shuffle e activat, ne uităm în vectorul de indici amestecați, identificăm unde se află k în listă și selectăm id-ul care se află în acea listă după k ca indice al noii melodii din playlist-ul curent

Mesaje posibile

1. Shuffle function activated successfully.
2. Shuffle function deactivated successfully.
3. The loaded source is not a playlist.

4. Please load a source before using the shuffle function.

```
{
    "command": "shuffle",
    "username": "bob35",
    "timestamp": 850,
    "seed": 1024
}
```

```
{
    "command" : "shuffle",
    "user" : "bob35",
    "timestamp" : 850,
    "message" : "Shuffle function activated successfully."
}
```

Această comandă poate fi rulată doar după ce utilizatorul a dat comanda load.

Forward/Backward

Comenzile forward și backward pot fi rulate doar atunci când user-ul ascultă un podcast. Prin aceste comenzi acesta poate avansa sau devansa cu 90 de secunde. Dacă au trecut mai puțin de 90 de secunde și se rulează backward, player-ul rămâne la începutul episodului curent. Dacă sunt mai puțin de 90 de secunde până se termină episodul și se rulează forward, începe automat următorul episod, de la început.

Mesaje posibile

1. Skipped forward successfully.
2. The loaded source is not a podcast.
3. Please load a source before skipping forward.
4. Rewound successfully.
5. Please select a source before rewinding.

Această comandă poate fi rulată doar după ce utilizatorul a dat comanda load.

```
{
    "command" : "forward",
    "user" : "bob35",
    "timestamp" : 1050
}
```

```
{
    "command" : "forward",
    "user" : "bob35",
    "timestamp" : 1050,
    "message" : "Please load a source before skipping forward."
}
```

```
{
    "command": "backward",
    "username": "bob35",
    "timestamp": 1390
}
```

```
{
    "command" : "backward",
    "user" : "bob35",
    "timestamp" : 1390,
    "message" : "Rewound successfully."
}
```

Like

Această comandă poate fi rulată doar dacă player-ul rulează o melodie în momentul respectiv. Utilizatorul poate să dea like melodiei curente, iar dacă i-a dat deja like, își retrage like-ul. Acesta formează o lista cu melodiile apreciate ce este verificată prin altă comandă.

Mesaje posibile

1. Like registered successfully.
2. Unlike registered successfully.
3. Please load a source before liking or unliking.
4. Loaded source is not a song.

```
{
    "command": "like",
    "username": "bob35",
    "timestamp": 205
}
```

```
{
    "command" : "like",
    "user" : "bob35",
    "timestamp" : 205,
    "message" : "Like registered successfully."
}
```

Această comandă poate fi rulată doar după ce utilizatorul a dat comanda load.

Next

Când se rulează comanda next, player-ul trece la următorul track, luând în calcul starea în care se află (status shuffle, status repeat). Dacă a ajuns la finalul unui playlist/podcast sau dacă a ajuns la finalul melodiei rulate din bibliotecă(e cazul în care rulăm doar o melodie), iar condiția de repeat specifică faptul că nu se repetă, atunci player-ul se pune pe pauză după ce s-a terminat playlist-ul/podcast-ul și rămâne fără vreo sursă încărcată cu care să poată interacționa.

Mesaje posibile

1. Skipped to next track successfully. The current track is {track_name}.
2. Please load a source before skipping to the next track.

Această comandă poate fi rulată doar după ce utilizatorul a dat comanda load.

Dacă după ce dați next nu mai aveți niciun track de rulat, se dă mesajul: **Please load a source before skipping to the next track.**.

```
{
    "command": "next",
    "username": "bob35",
    "timestamp": 590
}
```

```
{
    "command" : "next",
    "user" : "bob35",
    "timestamp" : 590,
    "message" : "Skipped to next track successfully. The current track is Under Pressure."
}
```

Prev

Când se apasă prev, putem să avem următoarele situații:

- player-ul trece la începutul fișierului audio dacă a trecut cel puțin o secundă din conținutul acestuia
- dacă nu a trecut nicio secundă din fișierul audio curent, player-ul trece la fișierul audio precedent

- dacă player-ul este la prima melodie din playlist/primul episod din podcast/într-o melodie din biblioteca, se reia de la început conținutul curent

Mesaje posibile

1. Returned to previous track successfully. The current track is {track_name}.
2. Please load a source before returning to the previous track.

```
{
    "command": "prev",
    "username": "bob35",
    "timestamp": 710
}
```

```
{
    "command" : "prev",
    "user" : "bob35",
    "timestamp" : 710,
    "message" : "Returned to previous track successfully. The current track is Start Me Up."
}
```

- Dacă se rulează această comandă, player-ul începe să ruleze conținutul, chiar dacă înainte se afla în starea de pauză.
- Această comandă poate fi rulată doar după ce utilizatorul a dat comanda load.

AddRemoveInPlaylist

Când se rulează această comandă se adaugă în playlist-ul specificat melodia curentă. Dacă aceasta există deja, se elimină din playlist-ul specificat. Playlist-urile sunt indexate de la 1, primul index fiind corespondent primului playlist creat.

Mesaje posibile

1. Successfully added to playlist.
2. Successfully removed from playlist.
3. The loaded source is not a song.
4. The specified playlist does not exist.
5. Please load a source before adding to or removing from the playlist.

```
{
    "command": "addRemoveInPlaylist",
    "username": "alice22",
    "timestamp": 24,
    "playlistId": 1
}
```

```
{
    "command" : "addRemoveInPlaylist",
    "user" : "alice22",
    "timestamp" : 24,
    "message" : "Successfully added to playlist."
}
```

Această comandă poate fi rulată doar după ce utilizatorul a dat comanda load.

Garantam ca aceasta comanda nu va fi rulata de un utilizator pe un playlist pe care il are incarcat in player.

Status

Se afișează starea în care se află player-ul user-ului curent.

```
  {
    "command": "status",
    "username": "alice22",
    "timestamp": 59
  }
```

```
{
    "command" : "status",
    "user" : "alice22",
    "timestamp" : 59,
    "stats" : {
      "name" : "The Power of Design",
      "remainedTime" : 3065,
      "repeat" : "No Repeat",
      "shuffle" : false,
      "paused" : false
    }
}
```

Comenzi playlist

CreatePlaylist

Se creeaza un playlist gol pentru un user. Dacă utilizatorul are un playlist cu acest nume, se primește un mesaj de eroare. Playlistul va avea inițial vizibilitatea setata pe public.

Mesaje posibile

1. Playlist created successfully.
2. A playlist with the same name already exists.

```
{
    "command": "createPlaylist",
    "username": "alice22",
    "timestamp": 5,
    "playlistName": "Playlist bengos"
}
```

```
{
    "command" : "createPlaylist",
    "user" : "alice22",
    "timestamp" : 5,
    "message" : "Playlist created successfully."
}
```

SwitchVisibility

Un playlist poate fi public sau privat. Când este creat acesta este public, ceea ce înseamnă că este vizibil pentru toți utilizatorii. Dacă se rulează această comandă, playlist-ul devine privat, dacă este public, respectiv public, dacă este privat.

Mesaje posibile

1. Visibility status updated successfully to {true/false}.
2. The specified playlist ID is too high.

```
{
    "command": "switchVisibility",
    "username": "carol19",
    "timestamp": 1130,
    "playlistId": 100
}
```

```
{
    "command" : "switchVisibility",
    "user" : "carol19",
    "timestamp" : 1130,
    "message" : "The specified playlist ID is too high."
}
```

FollowPlaylist

După ce se selectează un playlist via searchBar, un utilizator îi poate da follow. În cazul în care utilizatorul are deja playlistul la follow, atunci comanda va avea efectul de unfollow. Un playlist ce nu este deținut de utilizatorul curent e accesibil doar atunci când este public.

Mesaje posibile

1. Playlist followed successfully.
2. Playlist unfollowed successfully.
3. The selected source is not a playlist.
4. Please select a source before following or unfollowing.
5. You cannot follow or unfollow your own playlist.

```
{
    "command": "follow",
    "username": "carol19",
    "timestamp": 1050
}
```

```
{
    "command" : "follow",
    "user" : "carol19",
    "timestamp" : 1050,
    "message" : "Please select a source before following or unfollowing."
}
```

Această comandă poate fi rulată doar după ce utilizatorul a dat comanda select.

ShowPlaylists

Se vor afișa toate melodiile din toate playlist-urile deținute de utilizator.

```
{
    "command": "showPlaylists",
    "username": "alice22",
    "timestamp": 65
}
```

```
{
    "command" : "showPlaylists",
    "user" : "alice22",
    "timestamp" : 65,
    "result" : [ {
       "name" : "Playlist bengos",
       "songs" : [ "The Unforgiven" ],
       "visibility" : "public",
       "followers" : 0
    } ]
}
```

Statistici utilizatori

ShowPreferredSongs

Se va afișa o listă cu toate melodiile la care utilizatorul a dat like.

```
{
    "command": "showPreferredSongs",
    "username": "carol19",
    "timestamp": 1000
}
```

```
{
  "command" : "showPreferredSongs",
  "user" : "carol19",
  "timestamp" : 1000,
  "result" : [ "Bohemian Rhapsody" ]
}
```

Statistici generale

GetTop5Songs

Se va afișa o listă cu primele 5 melodii din bibliotecă care au primit cele mai multe like-uri de la utilizatori. În cazul în care sunt mai multe melodii cu același număr de like-uri, se alege în funcție de ordinea acestora în bibliotecă.

```
  {
    "command": "getTop5Songs",
    "timestamp": 3300
  }
```

```
{
    "command" : "getTop5Songs",
    "user" : null,
    "timestamp" : 3300,
    "result" : [ "Bohemian Rhapsody", "Shape of You", "Don't", "Stairway to Heaven", "Money Trees" ]
}
```

GetTop5Playlists

Se va afișa o listă cu primele 5 playlist-uri publice care au primit cele mai multe follow-uri de la utilizatori. În caz de egalitate alegem cel mai vechi playlist. Se garantează ca 2 playlist-uri nu au fost create în același timp.

```
{
    "command": "getTop5Playlists",
    "timestamp": 2560
}
```

```
{
    "command" : "getTop5Playlists",
    "user" : null,
    "timestamp" : 2560,
    "result" : [ "My first playlist", "Just for fun", "Felt cute might delete later", "Listen on repeat" ]
}
```

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 utile pentru citirea și rularea temei. Acestea se regăsesc în cadrul scheletului astfel:

- 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 Main care este punctul de intrare pentru logica de rezolvare a temei.

Pentru a înțelege mai bine cum funcționează citirea/scrie în fișierele JSON vă recomandăm să citiți Json & Jackson [https://ocw.cs.pub.ro/courses/poo-ca-cd/laboratoare/tutorial-json-jackson].

Output-ul nu trebuie formatat ca în ref-uri, fiindcă se verifică conținutul obiectelor și array-urilor JSON, nu textul efectiv. Cu toate acestea, dacă folosiți Jackson, vă recomandăm să utilizați **PrettyPrinter** Documentație PrettyPrinter [https://fasterxml.github.io/jackson-databind/javadoc/2.7/com/fasterxml/jackson/databind/ObjectMapper.html#writerWithDefaultPrettyPrinter()]. Totodată, pentru a înțelege cum se poate realiza **scrierea în fișierele JSON de output**, vă sugerăm să consultați JSON Array [http://makeseleniumeasy.com/2020/05/16/rest-assured-tutorial-27-how-to-create-json-array-using-jackson-api-objectmapper-createarraynode/].

Aveți în folder-ul **"lib"** toate dependințele necesare pentru rularea temei, mai exact bibliotecile Jackson.

Execuția temei

1. Se citesc listele cu useri, melodii si podcast-uri, in format JSON - e facuta deja citirea in schelet.
2. Se citesc comenzile si sunt puse in obiecte.
3. Se primesc secvențial comenzi și se execută pe măsură ce sunt primite.
4. După executarea unei comenzi, se afișează rezultatul ei în fișierul JSON de ieșire.
5. La terminarea tuturor comenzilor se termină și execuția programului.

- După ce clonați repo-ul de pe GitHub, vă rugăm să vă faceți un repository propriu privat în care să vă puneți doar conținutul folder-ului **"etapa1"** de pe repo-ul echipei de POO. Dacă nu puneți folder-ul cu tema la alta cale, **nu o să puteți** să faceți schimbări in Git, deoarece vă aflați în rădăcina repository-ului echipei de POO.
- Pentru ca checker-ul să funcționeze trebuie să deschideți tema din Intellij la calea unde se află folderele **"src"**, **"lib"**, **"ref"**, **"input"**. Aveți folder-ul **.idea** pregenerat ca să vă ajute in acest sens. De asemenea, fișier-ul **.iml** contine calea către bibliotecile Jackson. Dacă aveți probleme stergeți folder-ul .idea si fișierul .iml si generațile voi din nou din Intellij.
- **Citirea comenzilor si afisarea rezultatelor trebuie facuta de voi!!!**

Recomandări

- Tema a fost concepută pentru a vă testa cunoștințele dobândite în urma parcurgerii laboratoarelor 1-4, aceasta putând fi rezolvată doar cu noțiunile invățate din acele laboratoare.
- Pentru depanarea diferențelor dintre output-ul vostru si fișierele ref, vă recomandăm acest site [https://www.jsondiff.com/].
- Verificați periodic această pagină, deoarece scheletul/cerința pot suferi modificări în urma unor erori din partea noastră.

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

Pentru folosirea tool-ului **Git** vă punem la dispoziție un tutorial actualizat și amplu despre el la acest link și aveți de asemenea și un tutorial despre comenzile pe care puteți să le dați din IntelliJ la acest link.

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, 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.

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.
- **Aveți grijă să nu puneți pe Vmchecker fișiere .idea sau .iml**.

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 = 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 [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
- folderul .git

Resurse și linkuri utile

- Schelet de cod [https://github.com/oop-pub/oop-project-2023/tree/main/etapa1]
- Indicații pentru teme
- Recomandări coding style & javadoc