



UNIVERSITATEA DIN
BUCUREŞTI



FACULTATEA DE
MATEMATICA ŞI
INFORMATICA

SPECIALIZAREA INFORMATICĂ

Lucrare de licență

APLICAȚIE WEB PENTRU PASIONAȚII DE MUZICĂ

Absolvent
Robert-Gabriel Barbu

Coordonator științific
Conf.dr. Mihai Prunescu

București, iunie 2024

Rezumat

Lucrarea prezentă are ca scop definirea funcționalităților, dar și a particularităților tehnice ale aplicației web MoreThanMusic.

Aplicația urmărește să îmbunătățească experiența utilizatorului în ascultare de muzică prin crearea unui mediu interactiv și care să îi ofere recomandări personalizate. De cele mai multe ori, utilizatorii sunt copleșiți de multitudinea de opțiuni muzicale disponibile, iar o aplicație care ajută la descoperirea de noi melodii prin diverse funcționalități și jocuri poate fi o abordare modernă a acestei probleme.

De asemenea, majoritatea utilizatorilor de platforme muzicale folosesc Spotify pentru streaming-ul de muzică. Integrarea API-urilor Spotify facilitează autentificarea și permite sincronizarea facilă cu Spotify pentru aplicația MoreThanMusic, fără a face utilizatorii să renunțe la o aplicație cu care deja sunt familiari și unde au deja playlist-uri și melodii adăugate la secțiunea de favorite.

Abstract

The present paper aims to define the functionalities and technical particularities of the MoreThanMusic web application.

The application seeks to enhance the user's music-listening experience by creating an interactive environment that offers personalized recommendations. Often, users are overwhelmed by the multitude of available music options, and an application that helps discover new songs through various functionalities and games can be a modern approach to this problem.

Furthermore, most users of music platforms use Spotify for music streaming. Integrating Spotify APIs facilitates authentication and allows for easy synchronization with Spotify for the MoreThanMusic application, without requiring users to give up an application they are already familiar with and where they have existing playlists and songs added to their favorites.

Cuprins

1	Introducere	7
1.1	Contextul aplicației	7
1.2	Motivația pentru aplicație	8
2	Preliminarii	9
2.1	Inspirație și studii ajutătoare	9
2.2	Structura proiectului	10
3	Tehnologii și librării folosite	12
3.1	Baza de date	12
3.1.1	SQLite	12
3.2	Backend	13
3.2.1	Node.js	13
3.2.2	Express.js	13
3.2.3	Spotify Web API	14
3.2.4	OpenAI API	14
3.3	Frontend	15
3.3.1	HTML5	15
3.3.2	CSS3	15
3.3.3	Javascript	15
3.4	Librării folosite	16
3.4.1	Express	16
3.4.2	Express-session	16

3.4.3	SQLite3	16
3.4.4	Dotenv	16
3.4.5	Axios	16
3.4.6	Request	17
3.4.7	Node-cron	17
3.4.8	Path	17
3.4.9	Url	17
4	Detalii ale implementării	18
4.1	Configurarea bazei de date	18
4.1.1	Descrierea entităților și a relațiilor	19
4.2	Controller-e	22
4.3	Rute	24
4.4	Configurarea principală	26
4.5	Comunicarea frontend-ului cu backend-ul	27
5	Prezentarea aplicației	28
5.1	Paginile principale	28
5.1.1	Autentificarea utilizatorului	28
5.1.2	Pagina de Home	29
5.1.3	Recomandări muzicale	31
5.1.4	Evaluarea melodiilor	32
5.1.5	Pagina de clasament	34
5.1.6	Gestionarea pieselor favorite	35
5.1.7	Jocul de Wordle	37
5.1.8	Pagina de profil	38
5.2	Fluxuri de utilizare	40
5.2.1	Fluxul autentificării utilizatorului	40
5.2.2	Fluxul recomandărilor muzicale	41
5.2.3	Fluxul evaluărilor melodiilor	42

5.2.4	Fluxul gestionării pieselor favorite	43
5.2.5	Fluxul jocului de Wordle	44
6	Concluzii	45
6.1	Concluziile aplicației	45
6.2	Dezvoltare ulterioară	46
	Bibliografie	48

Capitolul 1

Introducere

1.1 Contextul aplicației

Era digitalizării a influențat semnificativ majoritatea domeniilor, printre care și cel muzical. Deoarece vânzarea de albume fizice a scăzut semnificativ de la an la an, descoperirea de noi artiști și de noi melodii se face de cele mai multe ori pe platformele destinate streaming-ului de muzică. De asemenea, se pune un accent mai mare pe recomandările personalizate și experiența pe care o au utilizatorii pe aceste platforme, rezultând într-un număr de ascultători care crește constant. [21]

Compania cu cel mai mare impact asupra muzicii este Spotify, având milioane de utilizatori activi și care sunt deja familiari cu această platformă, continuând să o folosească în detrimentul celorlalte platforme. Deoarece se axează în special pe descoperirea de noi melodii și pe facilitarea acestui aspect, este greu de atins același impact prin o aplicație care are ca specific streaming-ul de muzică. [22]

Deoarece persoanele tind să se plăcătă de o rutină de explorare sau să fie copleșite de multitudinea albumelor și listelor diversificate din care pot să aleagă, acestea ajung să asculte încontinuu albumul cu care erau obișnuiti fără să aibă energia necesară de a descoperi ceva nou. Aplicația MoreThanMusic are ca scop interacțiunea utilizatorului prin diverse activități care pot duce la explorarea de noi artiști, melodii, genuri muzicale și este sincronizată cu profilul de Spotify al utilizatorului pentru a facilita adăugarea melodiiilor

îndrăgite într-un mediu cu care deja sunt familiari.

1.2 Motivația pentru aplicație

Motivația care stă la baza creării acestei aplicații provine din problema de care, atât eu, cât și multe alte persoane, s-au lovit. Problema este neintroducerea unor activități care să stârnească interesul unui utilizator de a continua să descopere melodii care sunt foarte ascultate la momentul de față, melodii și artiști noi sau pe care nu i-au mai ascultat de mult timp. Acest lucru m-a făcut ca o perioadă îndelungată să nu mai folosesc platforme de streaming deoarece aveam deja un album de melodii care îmi plăceau, dar de care ajunsesem să mă plăcătesc. În plus, atunci când încercam să descopăr melodii noi, o puteam face doar prin a căuta albume de recomandări noi, fără a fi un factor de divertisment la mijloc sau care să mă influențeze să fac acest lucru.

Capitolul 2

Preliminarii

2.1 Inspirație și studii ajutătoare

Pentru a mă asigura că funcționalitățile pe care doresc să le implementez sunt unele care să atragă utilizatorii și să fie diferite față de ceea ce este deja pe piață, am ales să mă documentez despre ceea ce este în tendințe în rândul utilizatorilor de aplicații web. În acest proces de documentare, am studiat comportamentul utilizatorilor, popularitatea anumitor tipuri de jocuri și interacțiuni online.

Astfel, ideea de implementarea unui joc de Wordle a provenit din popularitatea pe care acest joc a dobândit-o după achiziția acestuia de către New York Times. Jocul are la bază ghicirea unui cuvânt prin multiple încercări și indicii ajutătoare pentru a valida care litere sunt poziționate unde trebuie. [16] Am adaptat acest concept pentru aplicația mea, transformând Wordle într-un joc bazat pe muzică, unde utilizatorii trebuie să ghicească numele unui artist sau titlul unei piese. Această adaptare nu doar că menține utilizatorii implicați, dar și combină divertismentul cu învățarea unor noi informații muzicale.

De asemenea, muzica are un impact important asupra stării de fericire a unui individ. De cele mai multe ori, indivizii aleg să asculte melodii care să le reflecte emoțiile simțite la momentul respectiv, indicând o conexiune directă între starea de emoție și muzica ascultată. [2] Studiile au arătat că muzica poate influența și chiar schimba starea emoțională a unei persoane, ceea ce m-a inspirat să includ o funcționalitate de recomandări muzicale

personalizate. Acest sistem de recomandări personalizează experiența utilizatorului prin integrarea de OpenAI, asigurându-se că acesta primește sugestii de melodii care sunt în concordanță cu starea sa emoțională.

În această ordine de idei, deși aplicația dezvoltată de mine are similarități cu celelalte aplicații de pe piață, se diferențiază prin aspectele de interacțiune directă a utilizatorilor cu aplicația și funcționalitățile acesteia, funcționalități care se bazează pe tendințe și personalizare. Aceste funcționalități nu doar că oferă o interacțiune mai plăcută și adaptată preferințelor utilizatorului, dar și încurajează o utilizare continuă și cu interes față de aplicație.

2.2 Structura proiectului

Proiectul este structurat în partea de backend (Fig 2.2.1) și în parte de frontend (Fig 2.2.2).

```
└── MUSIC_APP
    ├── backend
    │   ├── config
    │   ├── controllers
    │   ├── node_modules
    │   ├── routes
    │   └── .env
    ├── database.sqlite
    ├── init-db.js
    ├── package-lock.json
    ├── package.json
    └── server.js
```

Figura 2.2.1: Structura backend-ului

```
└── frontend
    ├── css
    ├── html
    ├── images
    └── javascript
```

Figura 2.2.2: Structura frontend-ului

În backend avem următoarele componente:

- config - acest folder conține configurațiile necesare pentru baza de date.

- controllers - acest folder conține toate fișierele care gestionează logica aplicației.
- routes - acest folder conține fișierele care definesc rutele API-ului pentru diverse funcționalități ale aplicației.
- init-db.js - este un script pentru initializarea și configurarea tabelelor în baza de date SQLite.
- server.js - conține configurarea serverului Express, gestionarea sesiunilor, a rutelor și a token-urilor.
- .env - conține variabile de mediu.
- package.json - este fișierul generat automat de configurare pentru Node.js care specifică metadatele proiectului și dependențele.
- package-lock.json - este fișierul generat automat care înregistrează starea actuală a dependențelor.

În frontend avem următoarele componente:

- css - acest folder conține fișierele CSS care definesc stilurile paginilor web.
- html - acest folder conține fișierele HTML care definesc structura paginilor web.
- images - acest folder conține imaginile utilizate în aplicație pentru icons, elemente grafice și fundaluri.
- javascript - acest folder conține fișierele JavaScript care adaugă funcționalități dinamice și interactive ale paginilor web.

Capitolul 3

Tehnologii și librării folosite

3.1 Baza de date

3.1.1 SQLite

Pentru a integra o bază de date în cadrul proiectului meu, am ales să folosesc SQLite, care este o bibliotecă de software care oferă un sistem de gestionare al bazelor de date mult mai simplu din punct de vedere al configurației față de alte alternative și care nu are nevoie de server. [6]

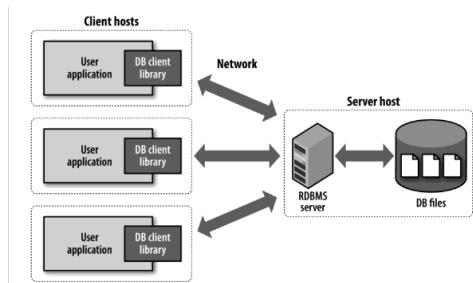


Figura 3.1.1.1: Arhitectura RD-BMS tradițională client/server [7]

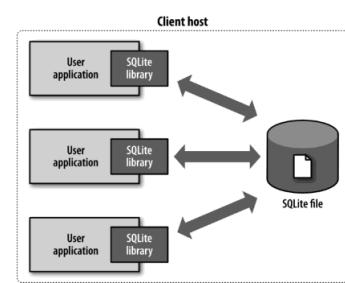


Figura 3.1.1.2: Arhitectura SQLite server-less [7]

Având o aplicație de dimensiuni mici, alegerea de a folosi SQLite a fost una optimă deoarece oferă performanțe bune pe acest tip de aplicații. În plus, deoarece nu necesită un server separat sau alte configurații complexe, a fost o alegere care a facilitat testarea rapidă și dezvoltarea ideilor pe care le-am avut pe parcursul implementării aplicației.

3.2 Backend

3.2.1 Node.js

Pentru partea de backend am folosit cu predominantă Node.js. Node.js este un mediu care permite utilizarea JavaScript atât pentru partea de frontend, cât și pentru partea de backend, lucru care permite scrierea unei aplicații complete în același limbaj de programare. [14]

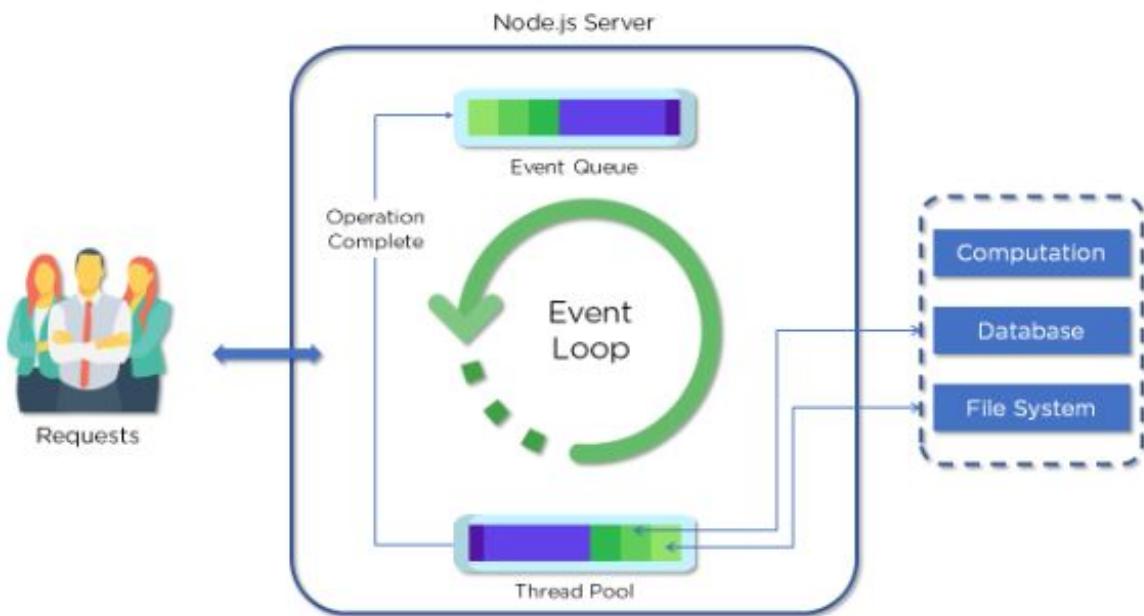


Figura 3.2.1.1: Arhitectura Node.js [18]

Am ales să folosesc Node.js deoarece este rapid și eficient și poate gestiona mai multe conexiuni în același timp și de asemenea, are o multitudine de module și pachete care mi-au permis să adaug sau să încerc funcționalități diferite fără nevoie de a scrie de la început codul corespondent acestora.

3.2.2 Express.js

Deoarece în aplicația mea folosesc un număr mare de rute, Express.js a fost folosit deoarece ajută la gestionarea ușoară a acestora și are o sintaxă clară și ușor de urmărit pentru potențialele erori care pot apărea.

Express.js e un framework pentru Node.js, aspect ce a ajutat la folosirea acestuia în cadrul aplicației mele, care ajută la procesul de dezvoltare al aplicațiilor web și al API-urilor. Aceasta este unul dintre cele mai utilizate framework-uri pentru Node.js tocmai pentru că ajută la gestionarea rutelor, cererilor și răspunsurilor într-un mod organizat, ducând la o eficiență mai mare. [8]

3.2.3 Spotify Web API

Spotify Web API este, în principiu, o modalitate care permite dezvoltatorilor să folosească funcționalități oferite de Spotify și să interacționeze cu datele furnizate de Spotify. Acesta oferă o multitudine de funcționalități care ajută la dezvoltarea unei aplicații de muzică, funcționalități precum căutarea de melodii, artiști, recomandări, gestionarea playlist-urilor. [20]

Având o aplicație care se axează pe muzică, am ales să folosesc Spotify Web API tocmai pentru a avea acces la o bază de date muzicale extinsă, la funcționalități diverse și la autentificarea prin contul de Spotify, plus ușurința integrării acestora deoarece există o documentație detaliată despre integrare.

În plus, pentru a folosi API-uri, e nevoie de un cont de dezvoltator și se oferă și acces la un dashboard al aplicației pe care am făcut-o. Acest dashboard m-a ajutat la monitorizarea utilizării API-ului, la gestionarea utilizatorilor, dar și la testare, deoarece se pot testa anumite cereri înainte de a le integra propriu-zis.

3.2.4 OpenAI API

OpenAI API oferă o modalitate prin care dezvoltatorii pot accesa modele de inteligență artificială create de OpenAI și le pot integra în aplicațiile lor. Funcționalitățile pe care le oferă sunt vaste, deoarece OpenAI API poate fi folosit de la generarea de text, răspunsuri la întrebări, până la analiză de date cu acuratețe mare. [12]

În aplicația dezvoltată de mine, am ales să folosesc OpenAI API pentru furnizarea unor recomandări personalizate pe baza unui nivel de fericire deoarece oferă răspunsuri

de calitate înaltă și are un set de date mult mai mare din care poate oferi recomandări. De asemenea, l-am folosit și pentru a ajuta la interacțiunea utilizatorilor cu aplicația, răspunsurile și recomandările fiind generate rapid și la fiecare apăsare pentru cerere.

3.3 Frontend

3.3.1 HTML5

HTML5 este ultima versiune a HTML, care este și limbajul standard pentru crearea paginilor web. Spre deosebire de predecesorul său, HTML5 aduce îmbunătățiri pentru multimedia și interactivitatea aplicațiilor web, dar este și o alternativă cu care mulți programatori sunt familiari pentru partea de frontend. [15]

3.3.2 CSS3

CSS3 este ultima versiune a CSS, care este folosit pentru stilizarea documentelor HTML. CSS3 aduce o multitudine de avantaje pe partea de stilizare, mai ales pentru flexibilitatea layout-urilor și crearea unui design care să atragă utilizatorii pe o aplicație. [19]

3.3.3 Javascript

Javascript reprezintă un limbaj de programare utilizat foarte des pentru paginile web. Aceasta oferă interactivitate aplicațiilor web și le face dinamice, dar comunică și asincron cu serverele, aspecte care îl fac o alternativă populară în rândul programatorilor. [3]

Alegerea de a folosi HTML5, CSS3 și Javascript pentru partea de frontend a venit din combinația acestora cu Node.js pentru partea de backend. Astfel, am reușit să dezvolt o aplicație web folosind în predominantă Javascript atât pentru backend, cât și pentru frontend, asigurând o modalitate mai eficientă și ușor de gestionat a codului și a dezvoltării aplicației într-un mod atractiv și modern.

3.4 Librării folosite

3.4.1 Express

Express este un framework web pentru Node.js, care simplifică dezvoltarea de aplicații web și API-uri. [5] În aplicație, Express gestionează routing-ul și configurarea serverului. Este folosit pentru a defini rutele care conectează diverse funcționalități ale backend-ului cu frontend-ul.

3.4.2 Express-session

Express-session este un middleware pentru gestionarea sesiunilor în Express. [5] În contextul aplicației mele, această librărie gestionează sesiuni de utilizator, ceea ce permite stocarea informațiilor despre autentificare și păstrarea stării utilizatorului între cereri.

3.4.3 SQLite3

SQLite3 este un sistem de manageriere al bazelor de date, care este ușor și integrat într-un singur fișier. [17] Aplicația utilizează SQLite3 pentru stocarea datelor utilizatorilor, pieselor favorite, rating-urilor și informațiilor de autentificare. Aceasta permite accesul rapid și eficient la date fără a necesita o configurare complexă a serverului de baze de date.

3.4.4 Dotenv

Dotenv este o librărie care ajută la încarcarea variabilele de mediu dintr-un fișier .env. [4] În aplicație este utilizată pentru gestionarea configurațiilor, cum ar fi cheile API și credențialele Spotify, fără a le expune direct în codul sursă.

3.4.5 Axios

Axios este o librărie folosită pentru efectuarea cererilor către servere externe. [1] Aplicația folosește Axios pentru a trimite cereri către Spotify Web API și OpenAI API,

pentru a obține recomandări muzicale și pentru autentificare.

3.4.6 Request

Request este o librărie simplă și ușor de utilizat pentru efectuarea cererilor HTTP. [13] Este folosită în aplicație pentru a face cereri către API-ul Spotify, pentru gestionarea autentificării și pentru obținerea informațiilor despre utilizatori și piese muzicale

3.4.7 Node-cron

Node-cron este o librărie pentru programarea anumitor task-uri în Node.js. [9] Această librărie este folosită în aplicație pentru a programa task-uri la o anumită perioadă de timp, cum ar fi actualizarea zilnică a pieselor virale în baza de date.

3.4.8 Path

Path este un modul din Node.js care se folosește pentru gestionarea căilor fișierelor și directoarelor. [10]

3.4.9 Url

Url este un modul din Node.js care se folosește pentru parsarea și manipularea URL-urilor. [11] Este folosit în aplicație pentru a construi și manipula URL-urile necesare pentru redirecționări și cereri API, mai ales în procesele de autentificare și autorizare cu Spotify.

Capitolul 4

Detalii ale implementării

4.1 Configurarea bazei de date

Pentru a integra SQLite în aplicație, am folosit pachetul sqlite3 din npm. Acest pachet permite interacțiunea ușoară cu bazele de date SQLite. Ulterior, am configurat conexiunea la baza de date SQLite și am creat un script care crează tabelele necesare.

Tabelele de care am avut nevoie pentru stocarea de informații care necesitau să fie păstrate într-o bază de date sunt:

- users - Conține date despre utilizatori, inclusiv un câmp pentru a stoca streak-ul de la Wordle.
- viral_songs - Conține informații despre piesele virale, care sunt folosite atât pentru rating, cât și pentru favorite.
- ratings - Ține evidența evaluărilor pieselor de către utilizatori.
- favorites - Stochează piesele pe care utilizatorii le-au marcat ca favorite.
- wordle - Ține evidența cuvintelor zilnice pentru Wordle și încercările utilizatorilor.
- last_update - Este utilizată pentru a urmări ultima dată când piesele virale au fost actualizate.

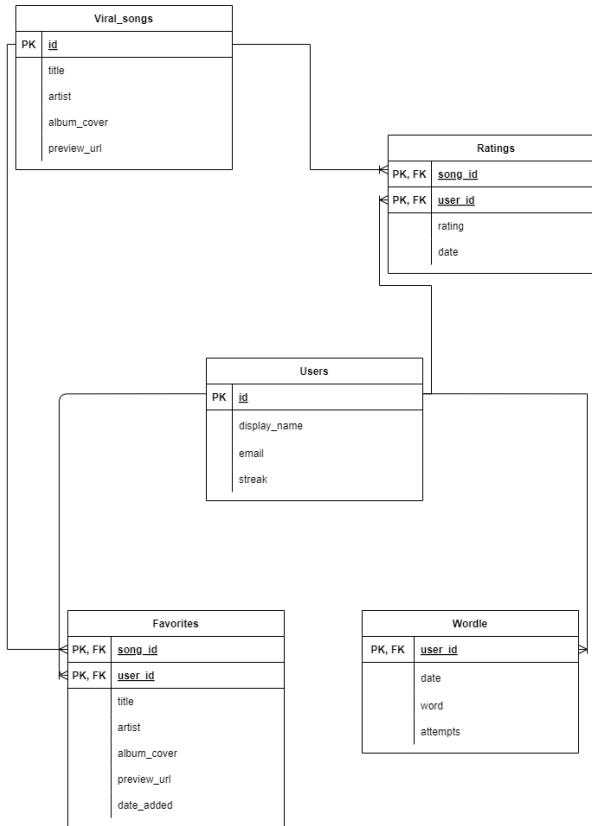


Figura 4.1.1: Diagrama ERD a bazei de date

4.1.1 Descrierea entităților și a relațiilor

Entitățile sunt:

1. users

- id: TEXT (Primary Key) - ID-ul unic al utilizatorului.
- display_name: TEXT - Numele de afișare al utilizatorului.
- email: TEXT - Email-ul utilizatorului.
- streak: INTEGER - Numărul curent de jocuri în care utilizatorul a ghicit corect artistul în jocul Wordle.

2. viral_songs

- id: TEXT (Primary Key) - ID-ul unic al piesei.
- title: TEXT - Titlul piesei.

- artist: TEXT - Artistul piesei.
- album_cover: TEXT - URL-ul imaginii albumului.
- preview_url: TEXT - URL-ul de previzualizare a piesei.

3. ratings

- song_id: TEXT - ID-ul piesei (Foreign Key legat de viral_songs).
- user_id: TEXT - ID-ul utilizatorului (Foreign Key legat de users).
- rating: INTEGER - Evaluarea dată piesei de către utilizator.
- date: TEXT - Data evaluării.

4. favorites

- song_id: TEXT - ID-ul piesei (Foreign Key legat de viral_songs).
- user_id: INTEGER - ID-ul utilizatorului (Foreign Key legat de users).
- title: TEXT - Titlul piesei.
- artist: TEXT - Artistul piesei.
- album_cover: TEXT - URL-ul imaginii albumului.
- preview_url: TEXT - URL-ul de previzualizare a piesei.
- date_added: TEXT - Data adăugării piesei la favorite.

5. wordle

- user_id: TEXT - ID-ul utilizatorului (Foreign Key legat de users).
- date: TEXT - Data jocului.
- word: TEXT - Cuvântul (artistul) pentru jocul Wordle.
- attempts: TEXT - Ultima încercare a utilizatorului pentru ziua respectivă.

6. last_update

- date: TEXT - Data ultimei actualizări a tabelului viral_songs.

Tabela last_update are un singur câmp, date, care stochează data ultimei actualizări a datelor din viral_songs. Scopul acestei tabele este de a preveni efectuarea de apeluri inutile către API-ul Spotify și actualizări ale bazei de date, dacă datele sunt deja la zi.

Relațiile între tabele sunt:

1. Relația între users și ratings

- Tip: Relație de tip One-to-Many
- Descriere: Fiecare utilizator poate evalua mai multe piese, dar fiecare evaluare este asociată unui singur utilizator. Relația este reprezentată prin coloana user_id din tabelul ratings, care face referire la coloana id din tabelul users.

2. Relația între viral_songs și ratings

- Tip: Relație de tip One-to-Many
- Descriere: Fiecare piesă poate primi evaluări de la mai mulți utilizatori, dar fiecare evaluare este asociată unei singure piese. Relația este reprezentată prin coloana song_id din tabelul ratings, care face referire la coloana id din tabelul viral_songs.

3. Relația între users și favorites

- Tip: Relație de tip One-to-Many
- Descriere: Fiecare utilizator poate avea mai multe piese favorite, dar fiecare piesă favorită este asociată unui singur utilizator. Relația este reprezentată prin coloana user_id din tabelul favorites, care face referire la coloana id din tabelul users.

4. Relația între viral_songs și favorites

- Tip: Relație de tip One-to-Many
- Descriere: Fiecare piesă poate fi favorizată de mai mulți utilizatori, dar fiecare piesă favorită este asociată unei singure piese. Relația este reprezentată prin

coloana song_id din tabelul favorites, care face referire la coloana id din tabelul viral_songs.

5. Relația între users și wordle

- Tip: Relație de tip One-to-Many
- Descriere: Fiecare utilizator poate juca jocul Wordle în fiecare zi, dar fiecare încercare zilnică este asociată unui singur utilizator. Relația este reprezentată prin coloana user_id din tabelul wordle, care face referire la coloana id din tabelul users.

În tabela de favorites se stochează și melodiile recomandate în funcție de nivelul de fericire dacă se apasă pe butonul de adăugare la favorite, dar deoarece acestea sunt generate de fiecare dată când utilizatorul accesează pagina, nu aveau nevoie de stocare într-un tabel separat.

4.2 Controller-e

Folderul controllers din structura proiectului conține logica aplicației. Acestea sunt responsabile pentru manipularea cererilor HTTP, interacțiunea cu baza de date și realizarea diverselor operațiuni necesare pentru funcționarea corectă a aplicației. Fiecare fișier din acest folder reprezintă un controller specific care gestionează o anumită parte a aplicației.

Descrierea Controller-elor:

1. authController.js

- Funcționalitate: Gestionarea autentificării utilizatorilor prin Spotify.
- Metode principale:
 - login: Redirectionează utilizatorul către pagina de autentificare Spotify dacă acesta nu e deja conectat pe Spotify.

- callback: Procesează codul de autentificare primit de la Spotify, obține tokenurile de acces și refresh, salvează datele utilizatorului în sesiune și baze de date, și redirecționează utilizatorul către pagina de start a aplicației.

2. favoritesController.js

- Funcționalitate: Gestionarea pieselor favorite ale utilizatorilor.
- Metode principale:
 - addFavorite: Adaugă o piesă la favoritele utilizatorului atât în baza de date locală, cât și în secțiunea "Liked Songs" de pe Spotify.
 - removeFavorite: Elimină o piesă din favoritele utilizatorului în ambele locații menționate mai sus.
 - getFavorites: Obține lista pieselor favorite ale utilizatorului din baza de date locală.

3. recommendationController.js

- Funcționalitate: Oferirea de recomandări muzicale personalizate utilizatorilor.
- Metode principale:
 - getRecommendations: Utilizează OpenAI API pentru a genera recoman-
 - dări de melodii bazate pe nivelul de fericire al utilizatorului. Obține detalii suplimentare despre melodii utilizând Spotify API.

4. songsController.js

- Funcționalitate: Gestionarea pieselor virale, evaluărilor și clasamentelor.
- Metode principale:
 - updateViralSongs: Actualizează lista pieselor virale din baza de date locală prin intermediul Spotify API..
 - getTopSongs: Obține lista pieselor virale pentru utilizatorul autentificat.
 - rateSong: Permite utilizatorilor să evaluateze piesele.

- getLeaderboard: Obține clasamentul pieselor bazat pe evaluările utilizatorilor.

5. userController.js

- Funcționalitate: Gestionarea profilului utilizatorului.
- Metode principale:
 - getUserProfile: Obține datele profilului utilizatorului din baza de date și le returnează în format JSON pentru afișare în frontend.

6. wordleController.js

- Funcționalitate: Gestionarea jocului Wordle bazat pe artiști muzicali.
- Metode principale:
 - getWordle: Obține cuvântul zilei pentru jocul Wordle și inițializează jocul pentru utilizator.
 - postWordle: Salvează încercările utilizatorului pentru cuvântul zilei.
 - updateStreak: Actualizează streak-ul utilizatorului în baza de date dacă acesta a ghicit corect cuvântul zilei.

4.3 Rute

În aplicație, rutele sunt punctele de intrare prin care cererile HTTP sunt gestionate și direcționate către funcțiile corespunzătoare din controller-e. Rutele sunt definite în fișierele din directorul routes, fiecare fișier fiind responsabil pentru un set specific de funcționalități. Rutele au rol de intermediari între cererile venite din partea utilizatorului (frontend) și logica aplicației definită în controller-e.

Rutele definite în aplicație sunt:

1. authRoutes.js

- GET /login: Apel către funcția login din authController, care redirecționează utilizatorul către pagina de autentificare Spotify.
- GET /callback: Apel către funcția callback din authController, care procesează răspunsul de la Spotify și gestionează token-urile de acces și refresh.

2. favoritesRoutes.js

- POST /api/add-favorite: Apel către funcția addFavorite din favoritesController, pentru a adăuga o piesă la favorite.
- POST /api/remove-favorite: Apel către funcția removeFavorite din favoritesController, pentru a elimina o piesă din favorite.
- GET /api/favorites: Apel către funcția getFavorites din favoritesController, pentru a obține lista pieselor favorite ale utilizatorului.

3. recommendationRoutes.js

- POST /api/recommendations: Apel către funcția getRecommendations din recommendationController, care returnează recomandări muzicale bazate pe nivelul de fericire al utilizatorului.

4. songsRoutes.js

- GET /api/update-viral-songs: Apel către funcția updateAndGetViralSongs din songsController, care actualizează și obține lista pieselor virale.
- GET /api/top-songs: Apel către funcția getTopSongs din songsController, pentru a obține lista pieselor virale pentru utilizatorul autentificat.
- POST /api/rate-song: Apel către funcția rateSong din songsController, pentru a permite utilizatorilor să evalueze piesele.
- GET /api/leaderboard: Apel către funcția getLeaderboard din songsController, pentru a obține clasamentul pieselor bazat pe evaluările utilizatorilor.

5. userRoutes.js

- GET /profile-info: Apel către funcția getUserProfile din userController, pentru a obține datele profilului utilizatorului și a le returna în format JSON.

6. wordleRoutes.js

- GET /api/wordle: Apel către funcția getWordle din wordleController, pentru a obține cuvântul zilei pentru jocul Wordle și a inițializa jocul pentru utilizator.
- POST /api/wordle: Apel către funcția postWordle din wordleController, pentru a salva încercările utilizatorului pentru cuvântul zilei.
- POST /api/update-streak: Apel către funcția updateStreak din wordleController, pentru a actualiza streak-ul utilizatorului dacă acesta a ghicit corect cuvântul zilei.

4.4 Configurarea principală

Fișierul server.js este componenta principală a aplicației, având rolul de a configura și inițializa serverul, de a gestiona sesiunile, de a defini rutele și de a seta diverse middleware-uri necesare pentru funcționarea corectă a aplicației.

În fișierul de server.js sunt efectuate următoarele aspecte ale aplicației:

- Importarea modulelor necesare.
- Inițializarea aplicației Express.
- Configurarea sesiunilor - Sesiunile sunt configurate folosind express-session, ceea ce permite păstrarea autentificării și a altor date de sesiune între diferite cereri HTTP.
- Parsarea JSON - Middleware-ul express.json() este utilizat pentru a permite aplicației să parseze cererile cu conținut JSON.
- Servirea conținutului static - Directorul frontend este setat pentru a servi fișiere statice(HTML, CSS, JavaScript, imagini).

- Definirea rutelor - Rutele sunt importate din diverse fișiere de rute și adăugate aplicației Express pentru a gestiona diferitele cereri HTTP.
- Definirea rutelor paginilor - Aceste rute gestionează redirectările către diverse pagini HTML.
- Pornirea Serverului - Serverul este configurat să asculte pe portul 8888.

4.5 Comunicarea frontend-ului cu backend-ul

Pentru a oferi interactivitate aplicației web, comunicarea dintre frontend și backend este un aspect important. În aplicația mea, această comunicare se realizează prin intermediul cererilor HTTP (GET și POST) care sunt gestionate de serverul Express (backend) și interfața de utilizator (frontend).

Capitolul 5

Prezentarea aplicației

5.1 Paginile principale

5.1.1 Autentificarea utilizatorului

Pagina de autentificare are ca scop permiterea utilizatorilor să se autentifice în aplicație folosind contul de Spotify.

Elemente cheie:

- Buton de autentificare: Redirecționează utilizatorii către pagina de autentificare Spotify.
- Mesaj de bun venit: Instruiește utilizatorii să se autentifice pentru a continua.
- Toggle pentru modul întunecat/luminos: Permite schimbarea între modurile de vizualizare.

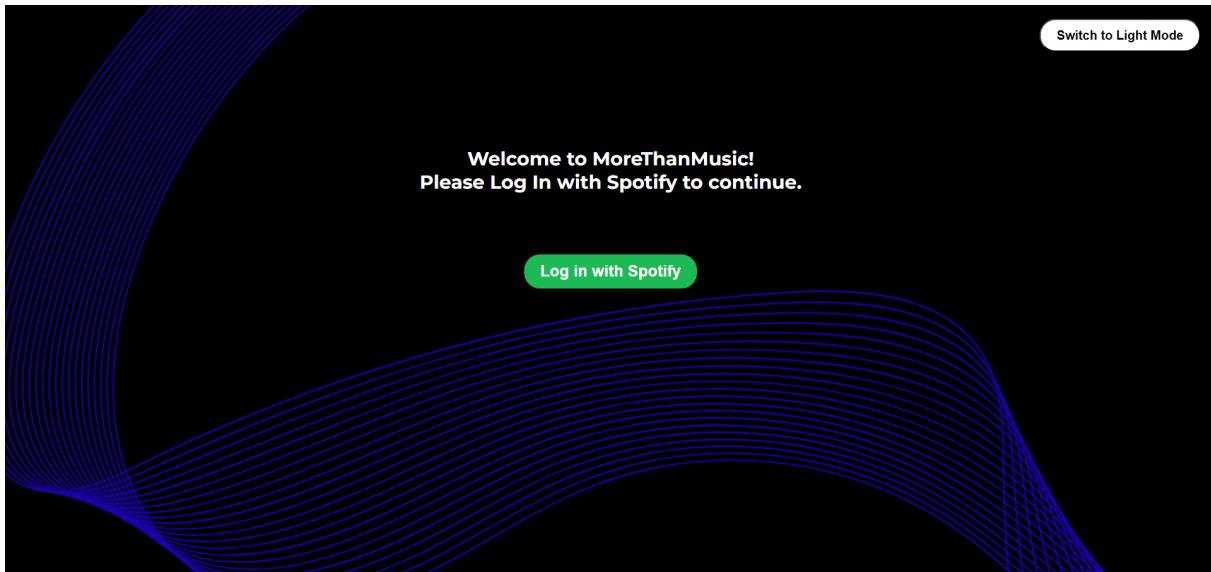


Figura 5.1.1.1: Pagina de autentificare - mod întunecat

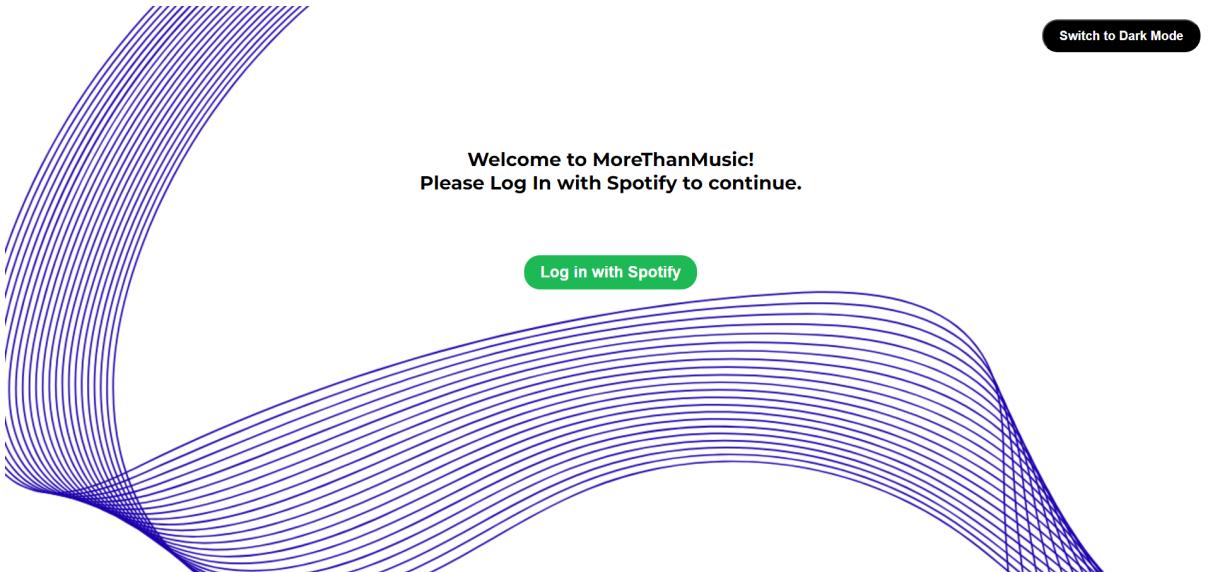


Figura 5.1.1.2: Pagina de autentificare - mod luminos

5.1.2 Pagina de Home

Pagina de Home are ca scop spațiul principal pentru utilizatori, oferind acces rapid la diverse funcționalități ale aplicației.

Elemente cheie:

- Buton de recomandări muzicale: Directionează către pagina de recomandări muzicale.
- Buton de favorite: Directionează către lista de melodii favorite ale utilizatorului.
- Buton de profil: Directionează către pagina de profil a utilizatorului.
- Buton de evaluare a melodiilor: Directionează către pagina unde utilizatorul poate evalua melodiile virale.
- Buton de joc Wordle: Directionează către jocul Wordle muzical.
- Toggle pentru modul întunecat/luminos: Permite schimbarea între modurile de vizualizare.

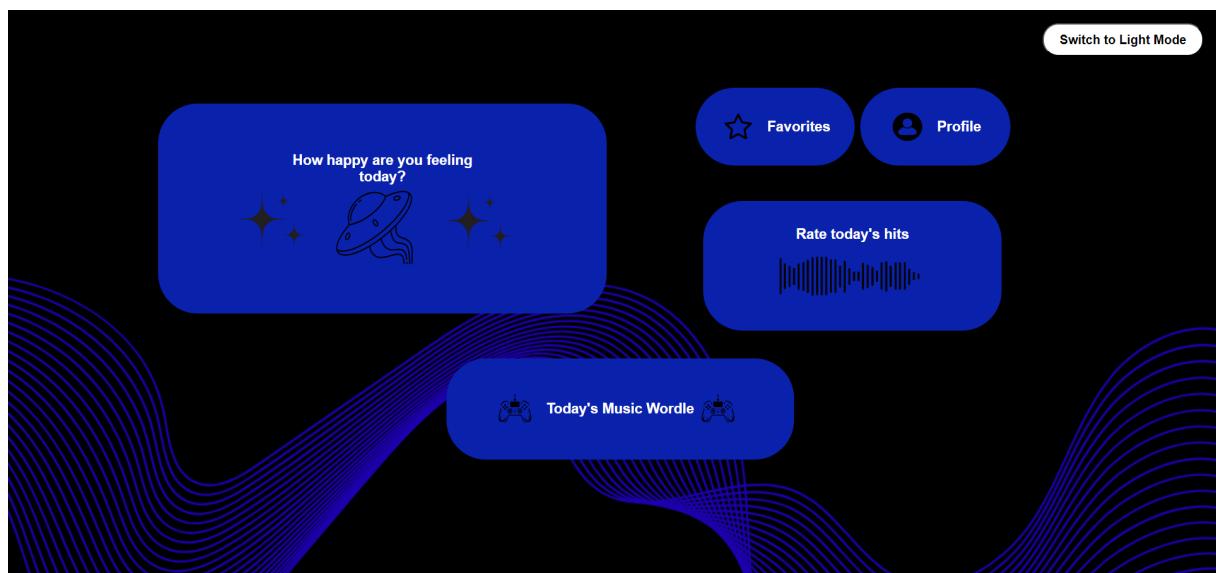


Figura 5.1.2.1: Pagina de Home - mod întunecat

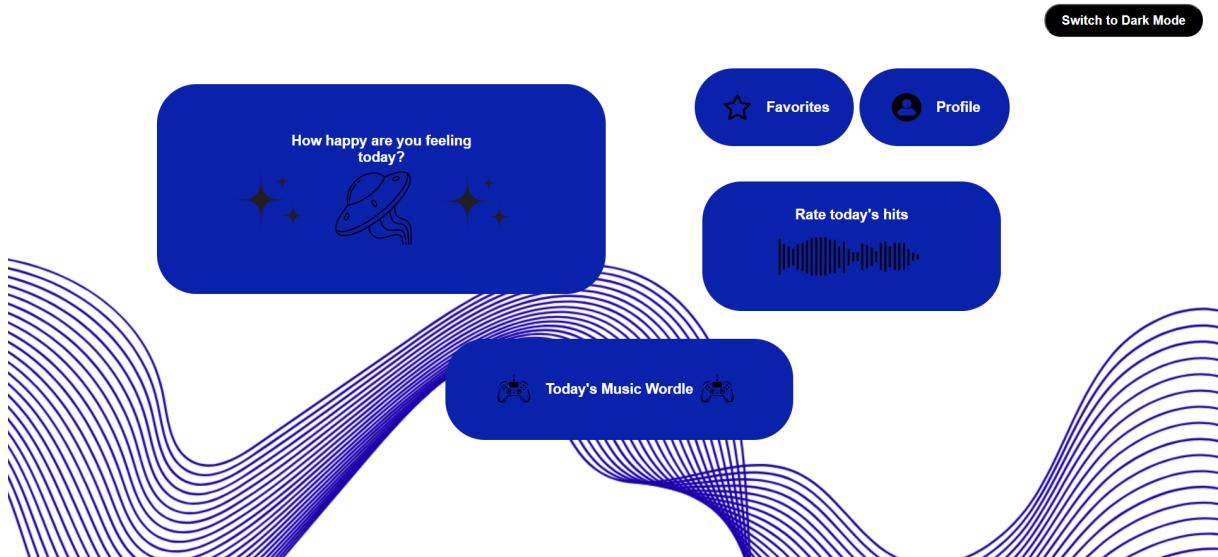


Figura 5.1.2.2: Pagina de Home - mod luminos

5.1.3 Recomandări muzicale

Pagina de recomandări muzicale oferă utilizatorilor recomandări muzicale personalizate bazate pe nivelul lor de fericire.

Elemente cheie:

- Input pentru nivelul de fericire: Utilizatorii introduc un număr între 1 și 10 pentru a indica cât de fericiți se simt.
- Buton de trimitere: Trimit nivelul de fericire pentru a primi recomandările muzicale corespunzătoare.
- Listă de recomandări: Afisează melodii recomandate cu opțiunea de a adăuga la favorite și Liked Songs pe Spotify.
- Toggle pentru modul întunecat/luminos: Permite schimbarea între modurile de vizualizare.
- Buton de Home: Directionează către pagina de Home.

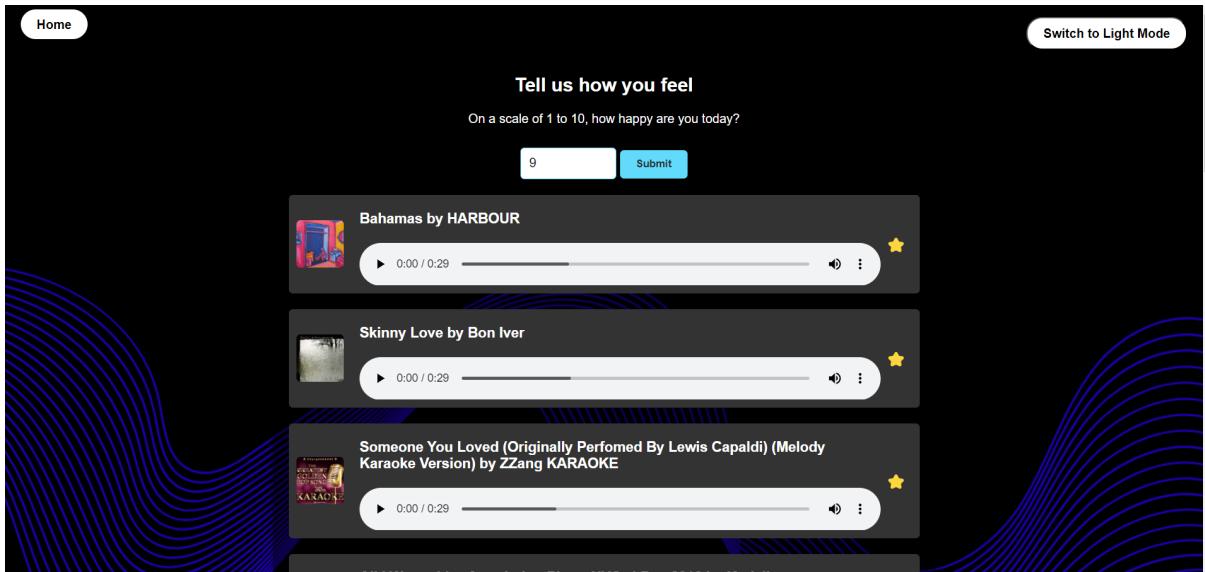


Figura 5.1.3.1: Pagina de recomandări - mod întunecat

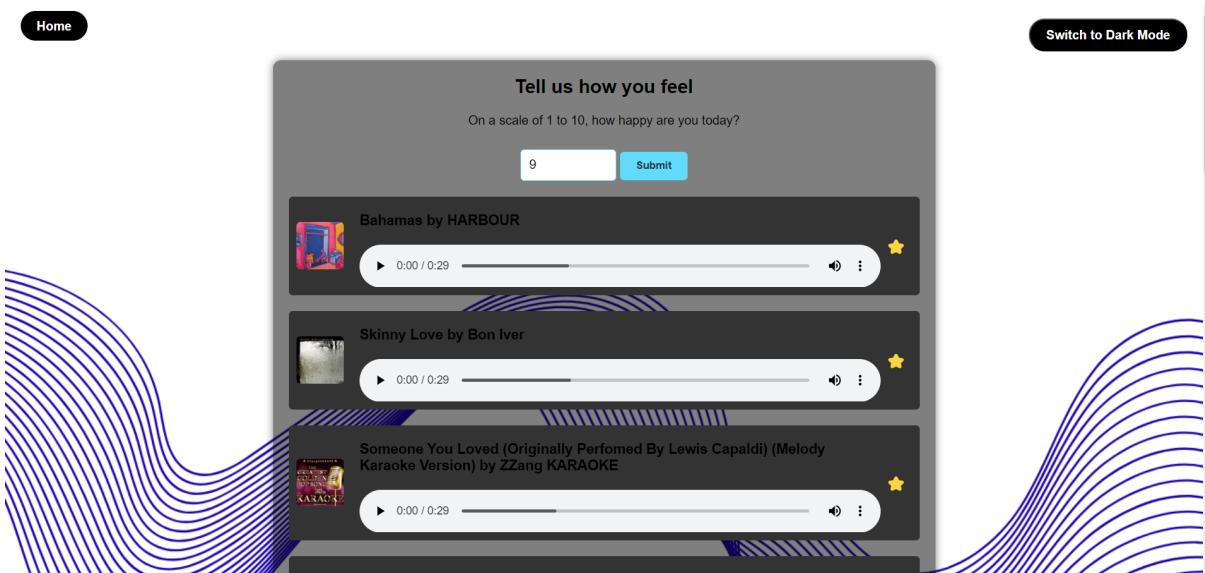


Figura 5.1.3.2: Pagina de recomandări - mod luminos

5.1.4 Evaluarea melodiilor

Pagina de evaluare a melodiilor permite utilizatorilor să evalueze melodile virale din ziua respectivă.

Elemente cheie:

- Listă de melodii: Afisează melodiile virale cu opțiunea de a asculta un preview și de a adăuga la favorite, cât și la Liked Songs pe Spotify.
- Input de evaluare: Utilizatorii pot introduce un rating pentru fiecare melodie.
- Buton de evaluare: Permite salvarea evaluărilor introduse.
- Toggle pentru modul întunecat/luminos: Permite schimbarea între modurile de vizualizare.
- Buton de Home: Direcționează către pagina de Home.
- Buton de clasament al comunității : Direcționează către pagina unde se află topul general al utilizatorilor pentru melodiile virale din ziua respectivă.

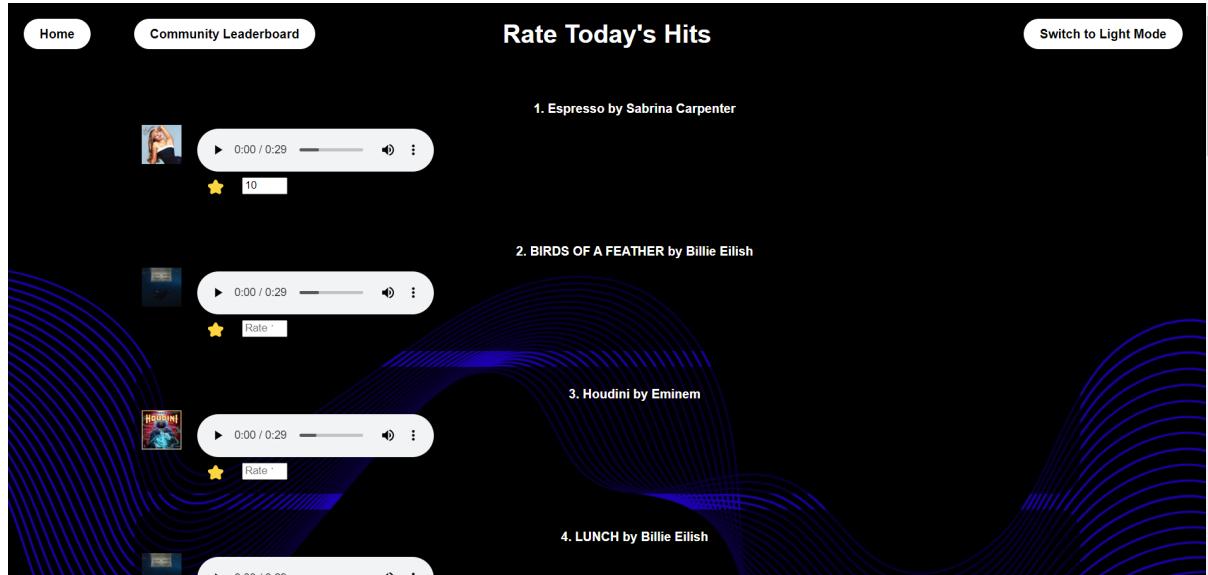


Figura 5.1.4.1: Pagina de evaluări ale melodiilor virale - mod întunecat

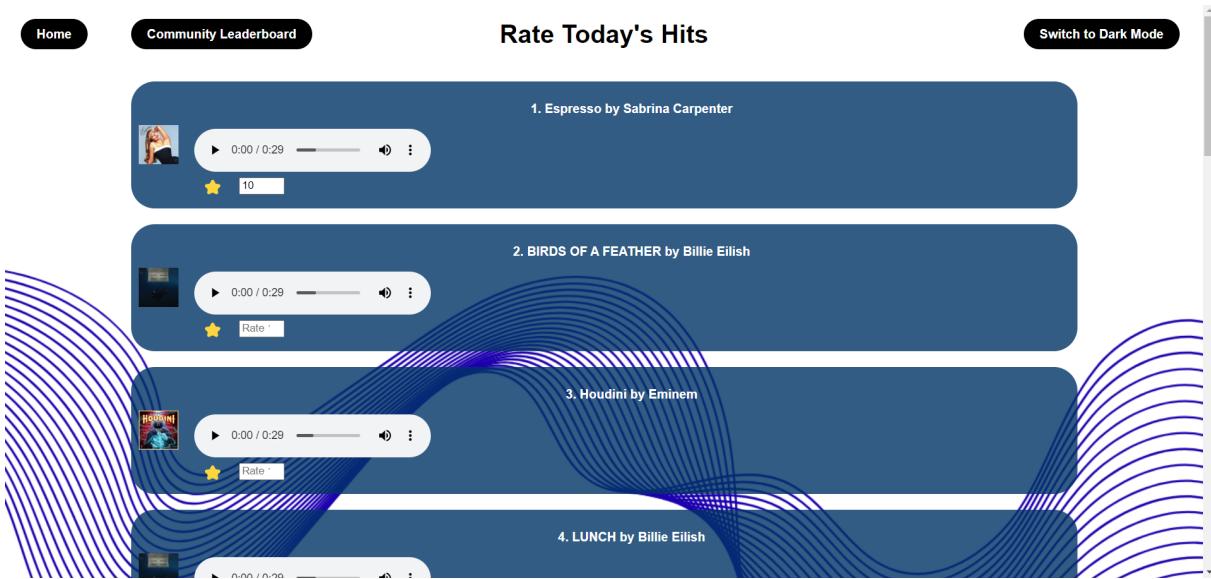


Figura 5.1.4.2: Pagina de evaluări ale melodiiilor virale - mod luminos

5.1.5 Pagina de clasament

Pagina de leaderboard este destinată afișării melodiiilor cu cele mai mari scoruri pe baza evaluărilor comunității pentru ziua curentă.

Elemente cheie:

- Listă de melodii: Afisează melodiiile virale cu opțiunea de a asculta un preview și scorul cumulat al comunității.
- Toggle pentru modul întunecat/luminos: Permite schimbarea între modurile de vizualizare.
- Buton de Back to Rating: Directionează către pagina de evaluare a melodiiilor.

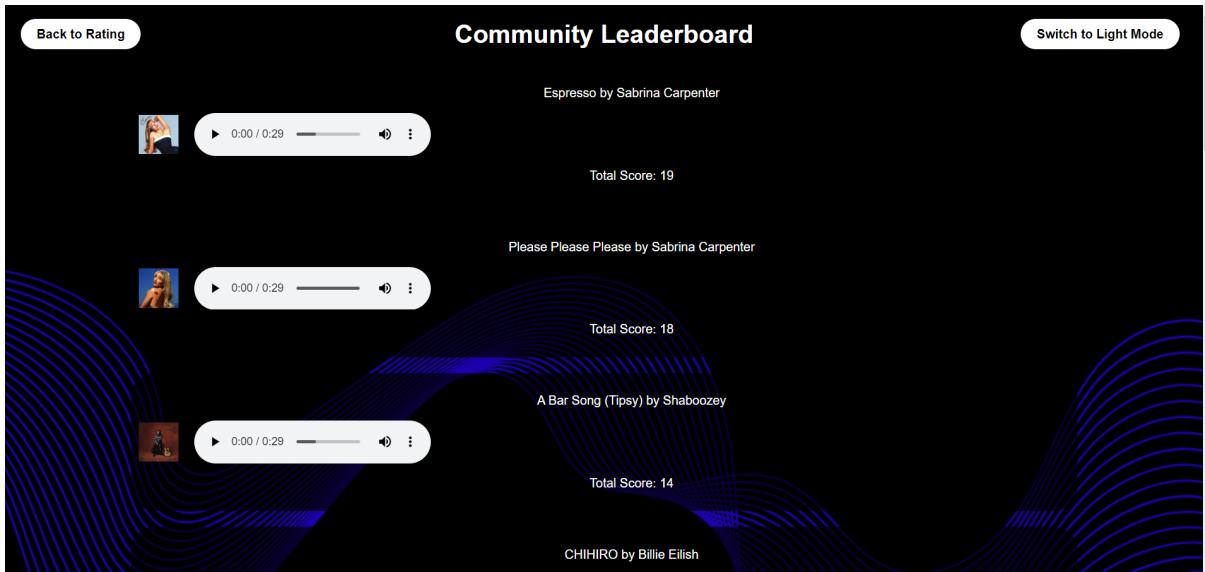


Figura 5.1.5.1: Pagina de clasament - mod întunecat

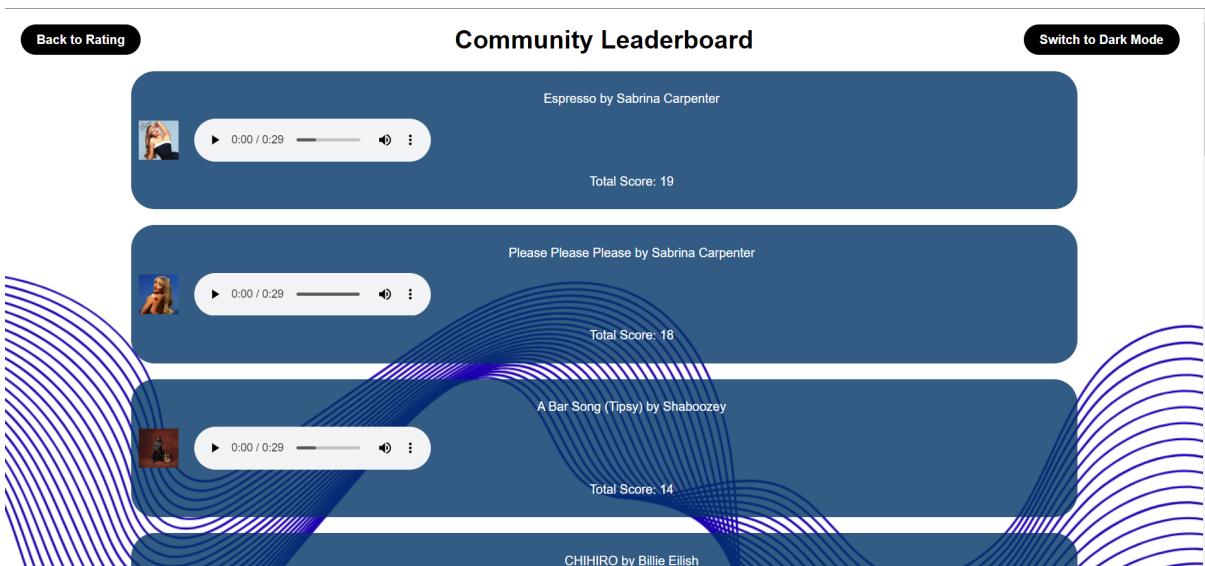


Figura 5.1.5.2: Pagina de clasament - mod luminos

5.1.6 Gestionaarea pieselor favorite

Pagina de favorite afișează melodiile pe care utilizatorii le-au adăugat la favorite în cadrul aplicației.

Elemente cheie:

- Listă de favorite: Afisează melodiile favorite ale utilizatorului.

- Buton de eliminare: Permite utilizatorilor să eliminate o melodie din lista de favorite, cât și din Liked Songs de pe Spotify.
- Toggle pentru modul întunecat/luminos: Permite schimbarea între modurile de vizualizare.
- Buton de Home: Direcționează către pagina de Home.

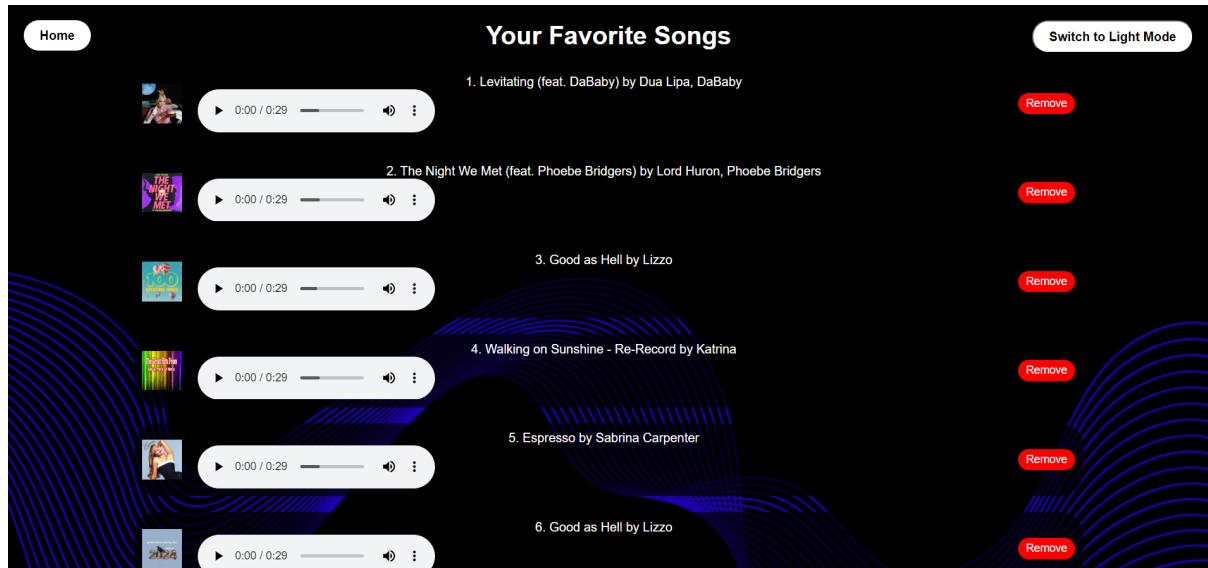


Figura 5.1.6.1: Pagina de favorite - mod întunecat

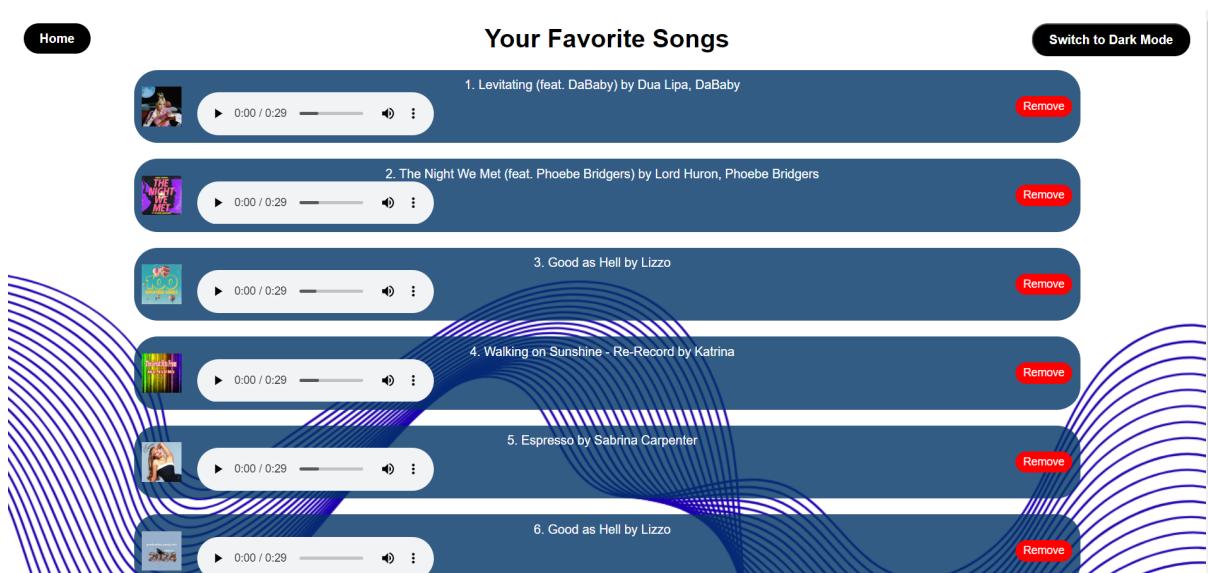


Figura 5.1.6.2: Pagina de favorite - mod luminos

5.1.7 Jocul de Wordle

Pagina jocului de Wordle muzical oferă utilizatorilor un joc tip Wordle adaptat pentru muzică, unde trebuie să ghicească numele unui artist.

Elemente cheie:

- Grilă Wordle: Afisează rândurile și celulele pentru introducerea literelor.
- Buton de verificare: Permite verificarea încercării curente.
- Indicator de stare: Afisează mesaje de succes sau eroare în funcție de rezultatul încercării.
- Toggle pentru modul întunecat/luminos: Permite schimbarea între modurile de vizualizare.
- Buton de Home: Directionează către pagina de Home.

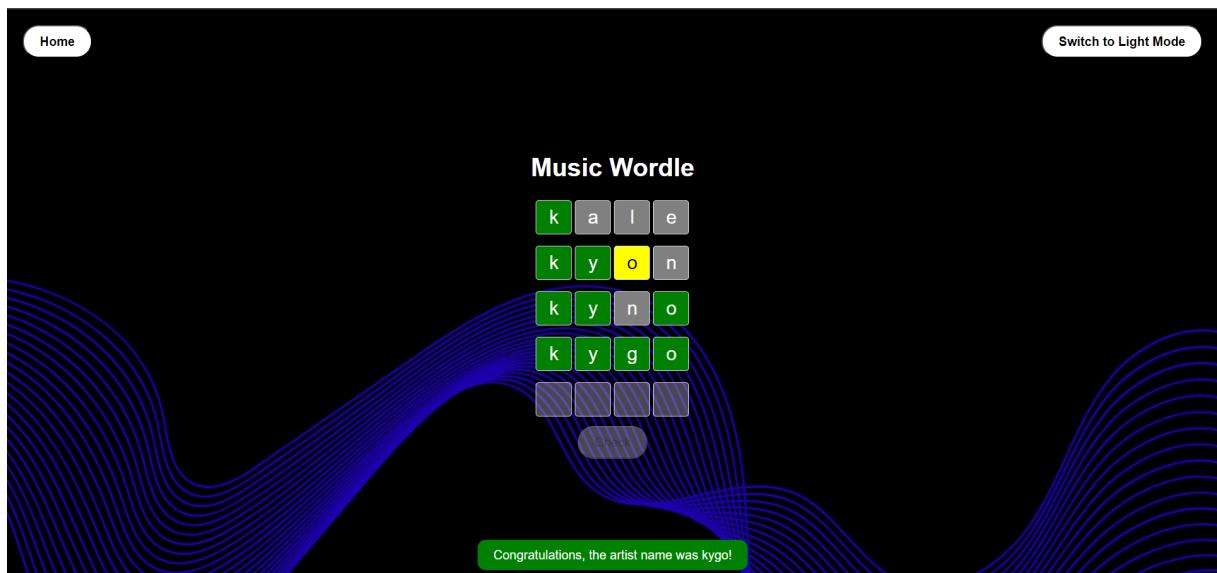


Figura 5.1.7.1: Pagina de Wordle cu artiști - mod întunecat

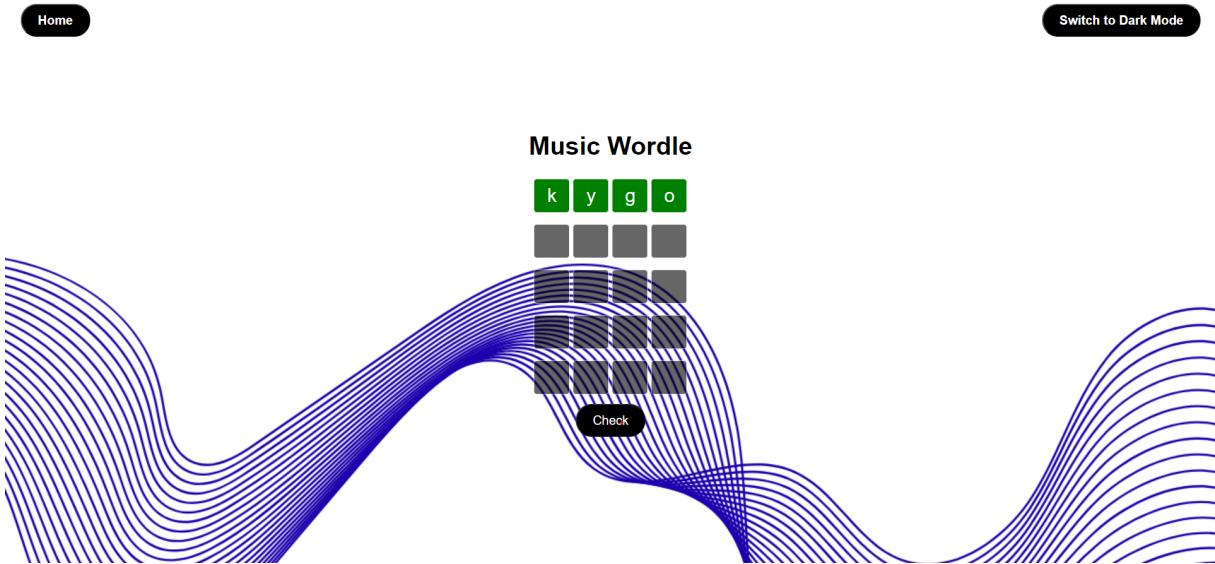


Figura 5.1.7.2: Pagina de Wordle cu artiști - mod luminos

5.1.8 Pagina de profil

Pagina de profil afișează informațiile de pe contul de Spotify ale utilizatorului, precum nume și email, și numărul de jocuri de Wordle muzical rezolvate.

Elemente cheie:

- Informații de profil: Nume, email.
- Streak Wordle: Afișează numărul de jocuri în care utilizatorul a ghicit corect artistul din Wordle.
- Toggle pentru modul întunecat/luminos: Permite schimbarea între modurile de vizualizare.
- Buton de Home: Directionează către pagina de Home.
- Buton de Back to Authentication: Directionează către pagina de Autentificare, unde utilizatorul se poate deconecta de pe contul de Spotify și să încerce un alt cont.

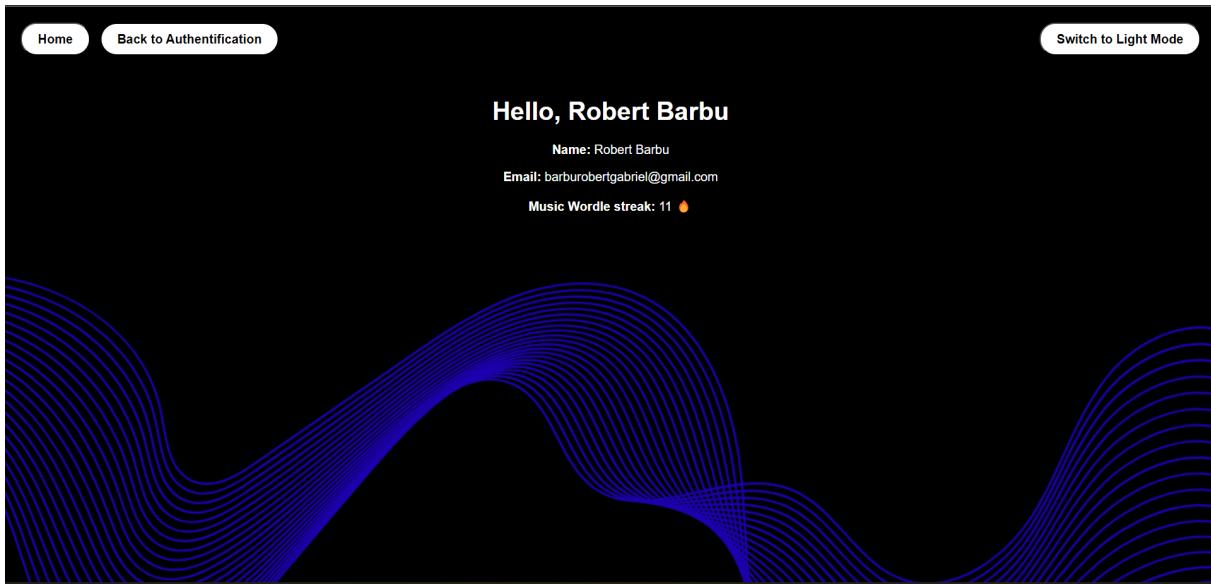


Figura 5.1.8.1: Pagina de profil - mod întunecat

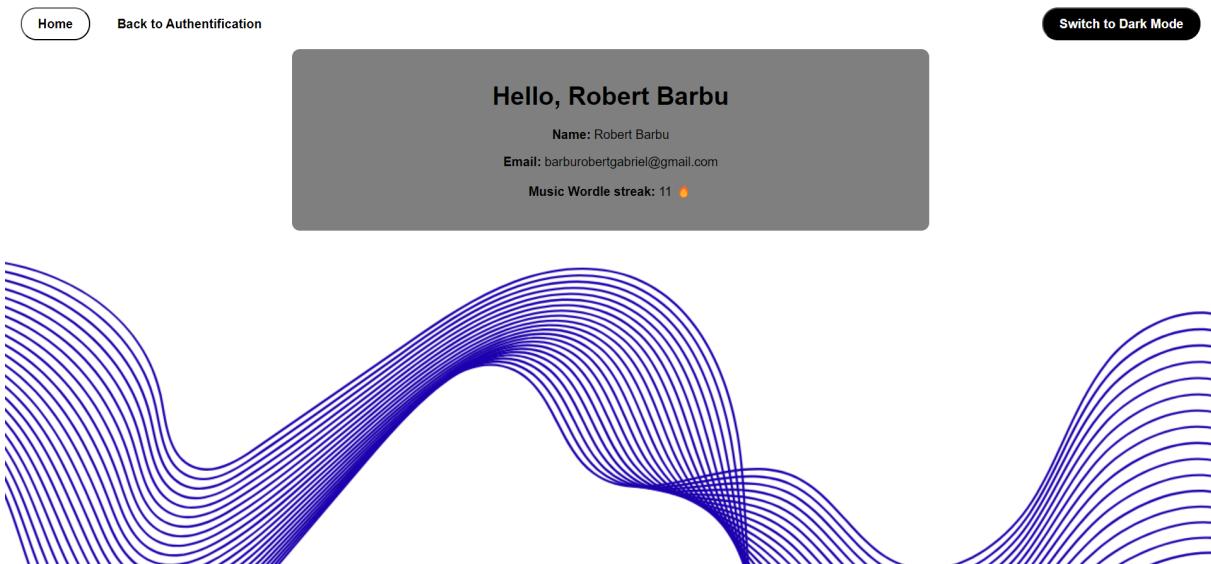


Figura 5.1.8.2: Pagina de profil - mod luminos

5.2 Fluxuri de utilizare

5.2.1 Fluxul autentificării utilizatorului

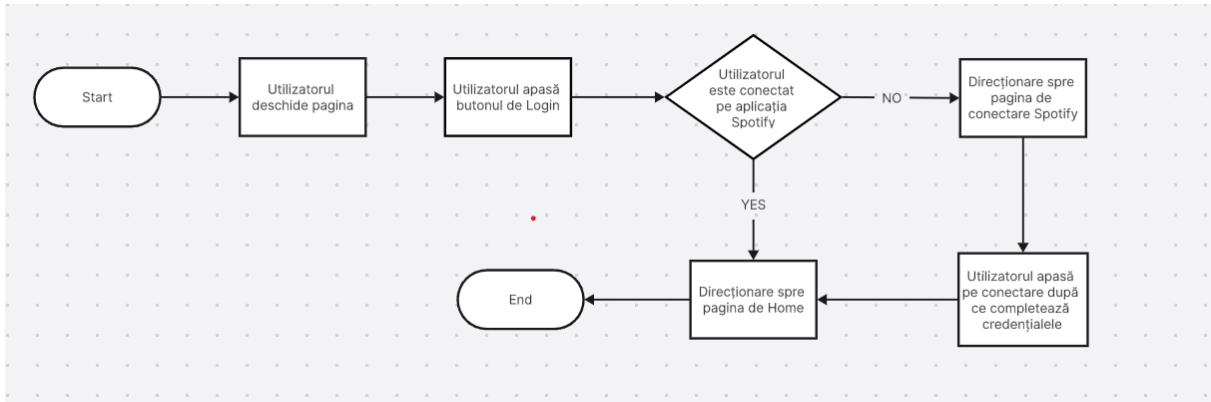


Figura 5.2.1.1: Diagrama fluxului autentificării

Pentru a se autentifica pe aplicație, utilizatorul trebuie să apese pe butonul de Login de pe pagina principală.

Dacă utilizatorul este deja autentificat pe aplicația de Spotify, nu mai trebuie să introducă credențialele și este direcționat la pagina de Home, de unde poate accesa funcționalitățile aplicației.

Dacă utilizatorul nu este deja autentificat pe aplicația de Spotify, acesta va fi direcționat pe pagina de conectare Spotify, unde își va introduce credențialele și va apăsa pe conectare, iar mai apoi va fi direcționat pe pagina de Home.

5.2.2 Fluxul recomandărilor muzicale

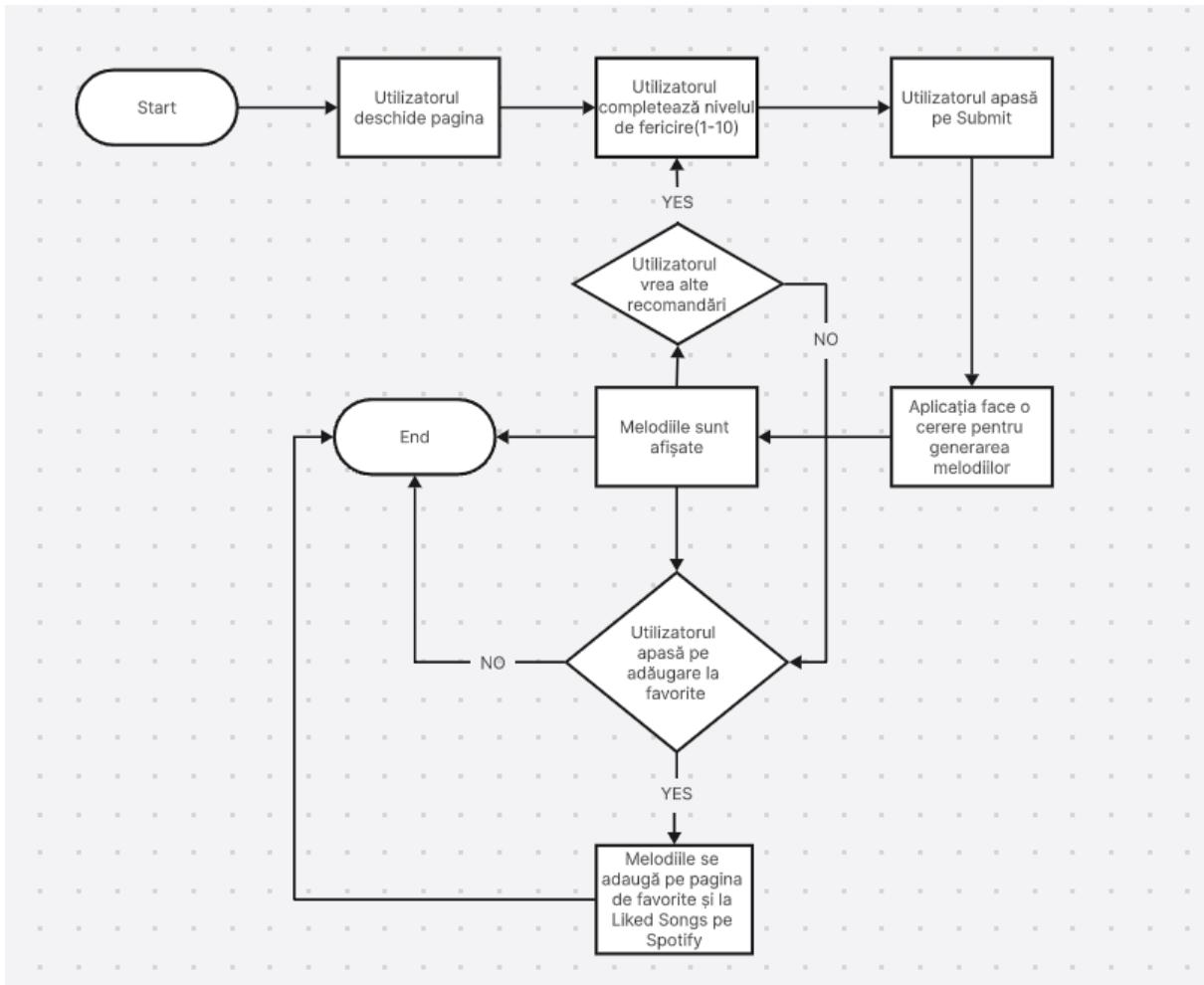


Figura 5.2.2.1: Diagrama fluxului paginii de recomandări

Pentru un utilizator care a deschis pagina de recomandări, va trebui să completeze cu un număr de la 1 la 10 nivelul de fericire pe care îl are la momentul respectiv. După ce acest număr a fost completat, utilizatorul apasă butonul de Submit și îi va apărea o listă cu 15 melodii, pe care poate alege să le asculte sau să le adauge la favorite. De asemenea, utilizatorul poate reveni oricând pe pagină și să trimită un nou nivel de fericire.

5.2.3 Fluxul evaluărilor melodiilor

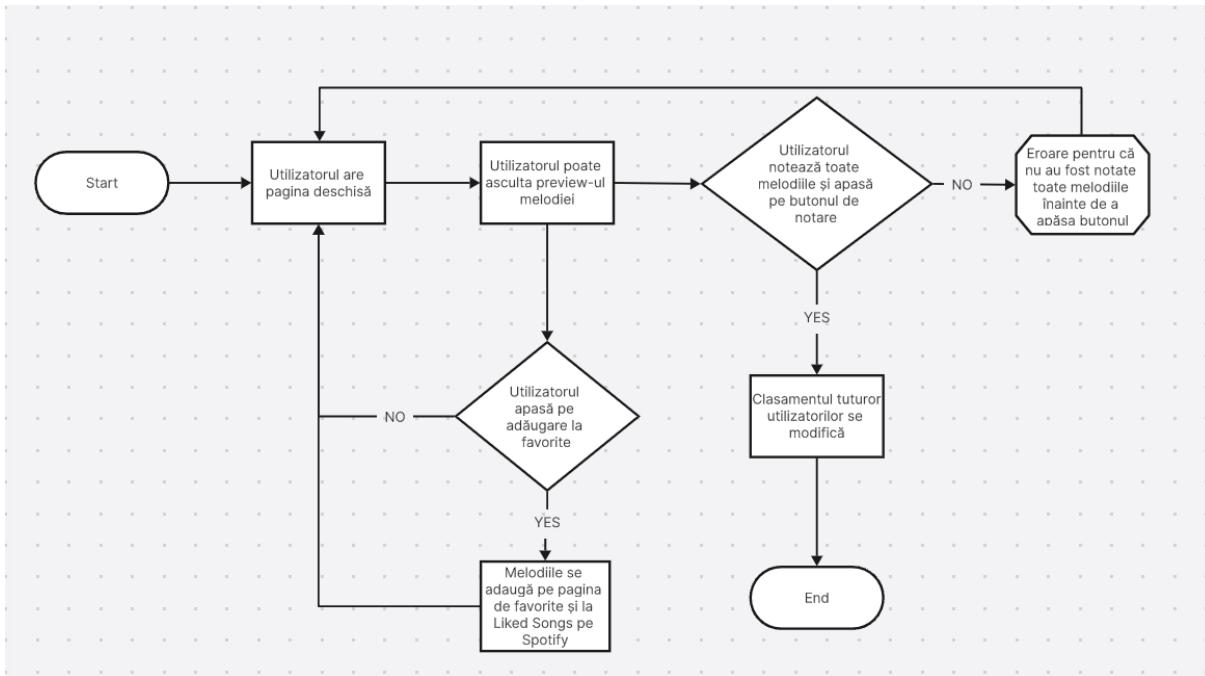


Figura 5.2.3.1: Diagrama fluxului paginii de evaluare a melodiilor

Când utilizatorul deschide pagina de evaluare a melodiilor virale, îi vor apărea top 15 melodii din tendințele Spotify din ziua respectivă. Utilizatorul poate alege să asculte preview-ul unei melodii sau să o adauge la favorite, dar dacă vrea să le noteze, trebuie să le dea notă tuturor 15. Dacă utilizatorul nu a notat toate 15 melodii, dar a apăsat pe butonul de notare, va apărea un mesaj de eroare pentru a evita scăderea în clasamentul general a unor melodii care nu au fost notate. Se poate nota doar o dată pe zi, iar mai apoi, dacă utilizatorul încearcă să noteze din nou, punctajele lui nu vor fi adăugate la clasament.

5.2.4 Fluxul gestionării pieselor favorite

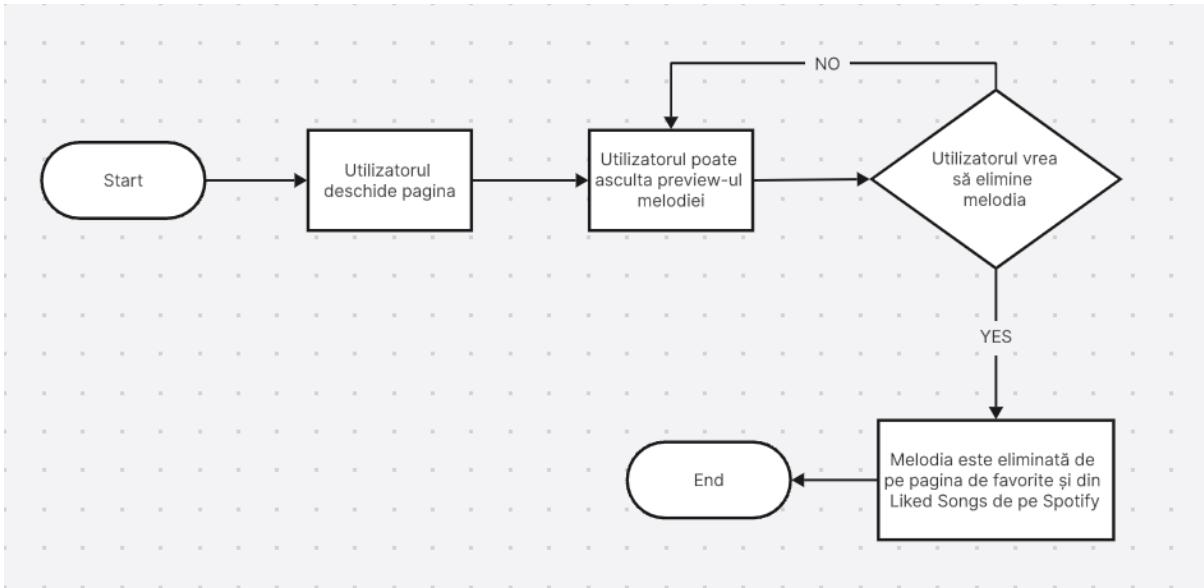


Figura 5.2.4.1: Diagrama fluxului paginii de favorite

Dacă utilizatorul se află pe pagina de favorite, poate asculta melodiile pe care acesta le-a adăugat la favorite. Când o melodie este adăugată la favorite, se adaugă și la Liked Songs de pe Spotify pentru a îmbunătăți interacțiunea dintre cele două aplicații.

În cazul în care unui utilizator nu îl mai place o melodie pe care acesta a adăugat-o la favorite, o poate elimina de pe pagina de favorite și atunci se elimină și din Liked Songs de pe Spotify.

5.2.5 Fluxul jocului de Wordle

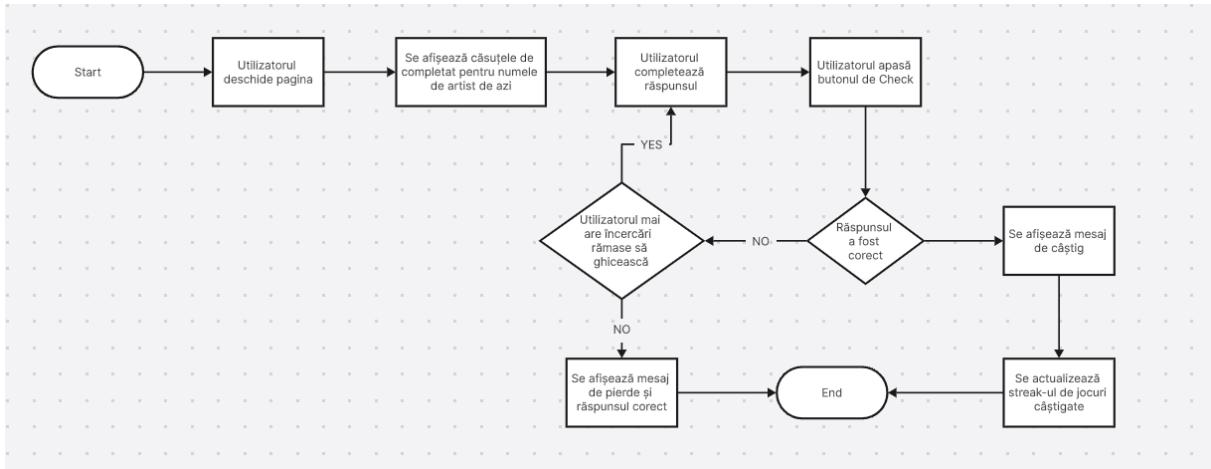


Figura 5.2.5.1: Diagrama fluxului jocului de Wordle

Jocul de Wordle are câte un nume de artist nou în fiecare zi, și nu poate fi jucat de mai multe ori în aceeași zi. Dacă utilizatorul alege să joace acest joc, va trebui să introducă în căsuțe numele artistului care crede că s-ar potrivi și să apese pe butonul de verificare.

Dacă utilizatorul a ghicit artistul, se va afișa un mesaj de câștig, iar scorul de jocuri câștigate care apare pe profil va crește.

Dacă utilizatorul nu a ghicit artistul, dar încă are încercări rămase, se poate uita la culorile căsuțelor de la încercarea anterioară (galben înseamnă că litera există în cuvânt dar nu e pusă unde trebuie, verde înseamnă că litera e unde trebuie, iar gri înseamnă că litera nu există în cuvânt) și mai poate trimite un nou răspuns. În cazul în care încercările au fost epuizate, se va afișa un mesaj de pierdere.

Capitolul 6

Concluzii

6.1 Concluziile aplicației

În dezvoltarea aplicației MoreThanMusic, am reușit să ating obiectivele propuse inițial și să implementez funcționalități care să ajute la interacțiunea cu utilizatorii.

Aplicația combină elemente de divertisment, cum ar fi jocul Wordle bazat pe muzică, cu funcționalități utile precum recomandările muzicale personalizate și gestionarea pieselor favorite. Folosind tehnologii moderne și eficiente, precum Node.js pentru backend și HTML, CSS și JavaScript pentru frontend, am construit o platformă distractivă și intuitivă.

Realizări cheie:

- Autentificare integrată cu Spotify: Utilizatorii se pot autentifica rapid și securizat folosind conturile lor Spotify, facilitând accesul la funcționalitățile aplicației fără a crea un nou cont.
- Recomandări muzicale personalizate: Prin integrarea API-ului OpenAI, aplicația oferă recomandări muzicale adaptate stării emoționale a utilizatorilor, îmbunătățind experiența de ascultare a muzicii.
- Evaluarea melodiilor: Utilizatorii pot evalua melodiile ascultate, contribuind la crearea unui leaderboard comunitar și oferind feedback valoros pentru alții utilizatori.

- Gestionarea pieselor favorite: Funcționalitatea de adăugare și eliminare a pieselor din lista de favorite permite utilizatorilor să își gestioneze preferințele muzicale într-un mod convenabil.
- Jocul Wordle: Jocul Wordle aduce un element de gamificare în aplicație, făcând interacțiunea cu platforma mai plăcută și mai atractivă pentru utilizatori.

Folosirea Node.js pentru backend a permis construirea unei aplicații funcționale și scalabile, în timp ce SQLite a oferit o soluție simplă și eficientă pentru gestionarea bazei de date. Pentru frontend, am utilizat HTML5, CSS3 și JavaScript, asigurând o interfață modernă și responsive. Integrările cu API-urile Spotify și OpenAI au fost esențiale pentru funcționalitățile cheie ale aplicației, iar librăriile utilizate (Express, Axios, Request, etc.) au simplificat considerabil dezvoltarea.

În concluzie, dezvoltarea aplicației MoreThanMusic a fost un proiect care mi-a permis să îmbin tehnologia cu pasiunea pentru muzică, și din care a realizat o aplicație unde utilizatorii pot descoperi melodii noi prin funcționalități interactive.

6.2 Dezvoltare ulterioară

În ceea ce privește dezvoltarea ulterioară a aplicației MoreThanMusic, am în vedere o serie de îmbunătățiri și extinderi care să aducă un plus de valoare utilizatorilor și să crească atraktivitatea platformei.

Potrivită funcționalități:

- Algoritmi de recomandare avansați: Ar necesita perfecționarea algoritmilor de recomandare muzicală, integrând modele de machine learning care să analizeze mai profund preferințele utilizatorilor și să ofere sugestii și mai precise.
- Personalizare extinsă: Aș adăuga opțiuni suplimentare de personalizare, permitând utilizatorilor să își ajusteze interfața și funcționalitățile aplicației conform preferințelor personale. De exemplu, aș include teme personalizate și widget-uri configurabile.

- Funcționalități sociale: Un aspect important ar fi adăugarea de funcționalități sociale care să permită utilizatorilor să interacționeze între ei, precum implementarea de opțiuni de partajare a playlist-urilor, comentarii și like-uri pentru piese și liste de redare.
- Gamificare extinsă: Aș extinde elementele de gamificare prin adăugarea de noi jocuri și provocări muzicale, concursuri și badge-uri pentru utilizatori. Astfel, utilizatorii vor fi motivați să se implice activ în aplicație.
- Notificări personalizate: Încă un aspect care ar ajuta interacțiunea frecventă este integrarea unui sistem de notificări care să informeze utilizatorii despre noile recomanări muzicale, actualizări ale clasamentului sau noi provocări din jocurile muzicale.

Bibliografie

- [1] *Axios - Getting Started*, URL: <https://axios-http.com/docs/intro> (accesat în 4.6.2024).
- [2] Rana Shabbir Ahmad; Akhtar Nasreen; North Adrian Charles, „Relationship between Interest in Music, Health and Happiness”, în *Journal of Behavioural Sciences* 21 (2011), p. 48, URL: <https://openurl.ebsco.com/EPDB%3Agcd%3A2%3A3832498/detailv2?sid=ebsco%3Aplink%3Ascholar&id=ebsco%3Agcd%3A63021786&crl=c>.
- [3] Douglas Crockford, *JavaScript: The Good Parts*, O'Reilly Media, Inc, 2008, pp. 1–4, ISBN: 978-0-596-51774-8.
- [4] *Dotenv*, URL: <https://www.npmjs.com/package/dotenv> (accesat în 4.6.2024).
- [5] *Express/Node introduction*, URL: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction (accesat în 4.6.2024).
- [6] Jay Kreibich, *Using SQLite*, O'Reilly Media, Inc, 2010, pp. 1–7, ISBN: 978-0-596-52118-9.
- [7] Jay Kreibich, *Using SQLite*, O'Reilly Media, Inc, 2010, p. 3, ISBN: 978-0-596-52118-9.
- [8] Azat Mardan, *Express.js Guide: The Comprehensive Book on Express.js*, Azat Mardan, 2014, pp. 1–10, ISBN: 978-1494269272.
- [9] *Node-cron*, URL: <https://www.npmjs.com/package/node-cron> (accesat în 4.6.2024).
- [10] *Node.js v22.2.0 documentation*, URL: <https://nodejs.org/api/path.html> (accesat în 4.6.2024).

- [11] *Node.js v22.2.0 documentation*, URL: <https://nodejs.org/api/url.html> (accesat în 4.6.2024).
- [12] „OpenAI documentation”, în *OpenAI Platform* (), URL: <https://platform.openai.com/docs/overview> (accesat în 3.6.2024).
- [13] *Requests Library*, URL: <https://docs.robotframework.org/docs/different-libraries/requests> (accesat în 4.6.2024).
- [14] Diogo Resende, *Node.js High Performance*, Packt Publishing Ltd, 2015, pp. 2–11, ISBN: 978-1-78528-614-8.
- [15] Bruce Lawson; Remy Sharp, *Introducing HTML5*, Peachpit Press, 2012, pp. 1–21, ISBN: 978-0-321-78442-1.
- [16] Qiuyu Shen, „Analysis on game popularity and difficulty: – An empirical study based on Wordle”, în *Highlights in Science, Engineering and Technology* 56 (2023), pp. 1–13, URL: <https://drpress.org/ojs/index.php/HSET/article/view/9811>.
- [17] *SQLite - Documentation*, URL: <https://sqlite.org/docs.html> (accesat în 4.6.2024).
- [18] Taha Sufiyan, „Understanding Node.js Architecture”, în *Simplilearn* (2023), URL: <https://www.simplilearn.com/understanding-node-js-architecture-article> (accesat în 3.6.2024).
- [19] Craig Grannell; Victor Sumner; Dionysios Synodinos, *The Essential Guide to HTML5 and CSS3 Web Design*, Apress, 2012, pp. 10–16, ISBN: 978-1-4302-3786-0.
- [20] „Web API - Retrieve metadata from Spotify content, control playback or get recommendations”, în *Spotify for Developers* (), URL: <https://developer.spotify.com/documentation/web-api> (accesat în 3.6.2024).
- [21] Josh Wilson, „The Age Of Digital; Music Executive Reacts To The Impact Of Digitalization In The Music Industry”, în *Forbes* (2022), URL: <https://www.forbes.com/sites/joshwilson/2022/09/14/the-age-of-digital-music-executive-reacts-to-the-impact-of-digitalization-in-the-music-industry/?sh=31554ff8537b> (accesat în 2.6.2024).

- [22] Rüya Yönak, „How Spotify has changed the way we listen to music”, în *Audiodioxide* (2019), URL: <https://audiodioxide.com/articles/how-spotify-has-changed-the-way-we-listen-to-music/> (accesat în 2.6.2024).