**COLEGIUL NAȚIONAL “CAROL I”**

**PROFIL REAL**

**SPECIALIZARE MATEMATICĂ-INFORMATICĂ**

**LUCRARE DE ATESTAT**

CINEMATOGRAFIA

ELEV **Stoica Mihai-Nicusor**

Clasa **a XII-a B**

PROFESOR ÎNDRUMĂTOR **Coravu Laura**

**CRAIOVA 2023**

**Continut:**

1. **Introducere**
2. **Tehnologii utilizate**
3. **Structura proiectului**
4. **Cerinte de sistem**
5. **Bibliografie**
6. **Introducere**

Tema proiectului este *Cinematografia* si este realizat sub forma unei aplicatii web de tip Single Page Application ce are ca scop recomandarea de filme si cantece similare cu titlurile pe care user-ul le ofera ca input. UI-ul aplicatiei este format din doua carduri: IMDb si Spotify, care sunt utilizate in functie de contextual aplicatiei. Cardul din stanga reprezinta input-ul, iar cel din dreapta recomandarile calculate in functie de input.

Contextul aplicatiei se schimba prin actionarea butonului *switch*(cel verde din centru), oferind user-ilor o experienta familiara cu UI-ul din Google Translate.

1. **Tehnologii utilizate**
* HTML
* CSS
* JavaScript
* Node.js
* Express.js
* VaderSentiment.js
* Spotify API
* TMDb API
1. **Structura Proiectului**

****

Folderul *expressApp* contine aplicatia propriu-zisa

*Index.html* este singurul fisier html necesar paginii. Acesta contine componentele UI-ului(containerele) pentru sectiunile IMDb si Spotify si butonul de switch.



*stylesheets/style.css* un fisier css obisnuit care aplica fonturi culori tranzitii animatii paginii web



*public/javascripts/contextSwitch.js* este scriptul referit in *index.html,* acesta are rolul de a realiza comunicarea intre client si server. Actualizeaza UI-ul, da post request-urilor(input-ul) pe endpointurile serverului si fetch pentru a primi raspunsurile(recomandari). 

*app.js* efectiv codul ce ruleaza pe server. In metoda post are loc procesarea inputului. Textul introdus de user prin request este verificat pentru a decide daca este un link de pe spotify, un link de pe IMDb sau un ID de pe IMDb cu ajutorul unor regex-uri. Dupa aceea utilizam metodele din *songRecommendations.js*  si *movieRecommendations.js*. Rezultatele le trimitem ca rezultat inapoi pe client.

*public/javascripts/genreApproximationValues.js* contine valori normale pentru a asocia feature-uri audio ale unor cantece cu genuri de film

*public/api/apiKeys.js* contine cheile de autentificare pentru api-urile folosite

*public/javascripts/songRecommendations.js* contine metodele responsabile pentru generarea de recomandari de cantece pe baza parametrului *movie*(film obtinut din input). Se extrag genurile filmului si se calculeaza scoruri medii pentru track-uri bazate pe valorile approximate. Pentru a obtine o legatura tematica intre cantec si film folosim VaderSentiment, un AI lightweight care aplica algoritmi de natural language processing pentru a extrage valenta afectiva a descrierii filmului pe o scara intre -1.0 si 1.0, unde -1.0 este maxim negative si 1.0 maxim pozitiv. Valenta este setata si ea ca parametru in *searchOptions*. Folosind API-ul Spotify cautam recomandari potrivite acestor optiuni.



*public/javascripts/movieRecommendations.js* face fix acelasi lucru, dar in sens invers, cautand filme cu API-ul TMDb folosindu-se de feature-urile de pe track-ul de input.

1. **Cerinte de sistem**

Deoarece partile cele mai complexe ale aplicatiei ruleaza pe serverside, singura cerinta de system este un browser.

Pentru a rula serverul este sufficient un virtual machine pe github codespaces (dual core, 4GB ram), pe care sa fie instalat Node.js cu extensiile potrivite.

1. **Bibliografie**

[**https://developers.themoviedb.org/3**](https://developers.themoviedb.org/3)

[**https://developer.spotify.com/**](https://developer.spotify.com/)

[**https://code.visualstudio.com/docs/nodejs/nodejs-tutorial**](https://code.visualstudio.com/docs/nodejs/nodejs-tutorial)

[**http://expressjs.com/en/api.html**](http://expressjs.com/en/api.html)

[**https://nodejs.org/api/**](https://nodejs.org/api/)

[**https://github.com/cjhutto/vaderSentiment**](https://github.com/cjhutto/vaderSentiment)