UNIVERSITATEA BABEŞ-BOLYAI CLUJ-NAPOCA
FACULTATEA DE MATEMATICĂ ŞI INFORMATICĂ
SPECIALIZAREA INFORMATICA

LUCRARE DE LICENŢĂ

SISTEM PENTRU RECOMANDARE DE FILME

Conducător ştiinţific
[Grad, titlu si nume coordonator]

Absolvent
RADU DRAGOS STEFAN

2021

ABSTRACT

Abstract: un rezumat în limba engleză cu prezentarea, pe scurt, a conţinutului pe capitole, punând accent pe contribuţiile proprii şi originalitate

Cuprins

Capitolul 1

Capitolul 3: Tehnologiile de pe Back-end

1.1 Node Js

Node js este un ecosistem de JavaScript , open-source , cross-platform si care este bazat pe motorul V8 de la Chrome acesta executand cod JavaScript in afara unui browser web. Node.js permite dezvoltatorilor să folosească JavaScript pentru a scrie instrumente de linie de comandă și pentru scripturi de pe partea de server, rularea scripturilor pe partea de server pentru a produce conținut dinamic al paginii web înainte ca pagina să fie trimisă în browserul web al utilizatorului.

JavaScript a fost conceput ca un limbaj de programare cu un singur fir care rulează într-un browser. A fi un singur fir înseamnă că numai un singur set de instrucțiuni este executat în orice moment în același proces (browserul, în acest caz, sau doar fila curentă în browserele moderne). Acest lucru a făcut lucrurile mai ușoare pentru implementare și pentru dezvoltatorii care folosesc limba. JavaScript a fost inițial un limbaj util numai pentru adăugarea unor interacțiuni la paginile web, validări de formulare și așa mai departe - nimic care să nu necesite complexitatea multithreading-ului. Cum funcționează sub capotă este destul de interesant. În comparație cu tehnicile tradiționale de servire web în care fiecare conexiune (cerere) generează un fir nou, preluând RAM de sistem și, în cele din urmă, maximizând cantitatea de RAM disponibilă, Node.js funcționează pe un singur fir, utilizând apeluri non-blocante, permițându-i să suporte zeci de mii de conexiuni simultane ținute în bucla evenimentului

Cand vorbim de node js este foarte important de mentionat este suportul incorporat pentru gestionarea pachetelor utilizand NPM(node package manager), un instrument care vine în mod implicit la fiecare instalare Node.js. Ideea modulelor NPM este destul de similară cu cea a Ruby Gems : un set de componente reutiliza-

bile disponibile public, disponibile prin instalare ușoară printr-un depozit online, cu gestionarea versiunilor și dependenței. Ecosistemul modulului este deschis tuturor și oricine își poate publica propriul modul care va fi listat în depozitul npm.

1.2 Javascript

În cea mai mare parte a vieții sale, limbajul de programare JavaScript a trăit în interiorul web browsere. A început ca un limbaj de scriptare simplu pentru modificarea detaliilor mici ale pagini web, dar a devenit un limbaj complex, cu o mulțime de aplicații și biblioteci. Mulți furnizori de browsere precum Mozilla și Google au început să pompeze resurse în timpii de rulare rapid JavaScript, iar browserele au obținut motoare JavaScript mult mai rapide ca un rezultat.În 2009, a apărut Node.js. Node.js a scos V8, puternicul motor JavaScript Google Chrome, din browser și i-a permis să ruleze pe servere.În browser, dezvoltatorii nu au avut de ales decât să aleagă JavaScript. Pe lângă Ruby, Python, Java, și alte limbi, dezvoltatorii ar putea alege acum JavaScript atunci când dezvoltă aplicații de pe server.Este posibil ca JavaScript să nu fie limbajul perfect pentru toată lumea, dar Node.js are adevărat beneficii. În primul rând, motorul V8 JavaScript este rapid, iar Node.js încurajează un stil de codare asincron, făcând codul mai rapid evitând în același timp coșmarurile cu mai multe fire. JavaScript avea, de asemenea, o mulțime de biblioteci utile datorită popularității sale. Cu exceptia Cel mai mare beneficiu al Node.js este capacitatea de a partaja codul între browser și server. Dezvoltatorii nu trebuie să facă niciun fel de schimbare de context atunci când trec de la client și Server.

1.3 MySQL

MySQL este un sistem de gestiune al bazelor de date relațional, produs de compania suedeză MySQL AB și distribuit sub Licență Publică Generală GNU. Este cel mai popular SGBD open-source la ora actuală, fiind o componentă cheie a stivei LAMP(Linux, Apache, MySQL, PHP).

Deși este folosit foarte des împreună cu limbajelel de programare JAVA, PHP, cu MySQL pot construi aplicații în orice limbaj major. Există multe scheme API disponibile pentru MySQL ce permit scrierea aplicațiilor în numeroase limbaje de programare pentru accesarea bazelor de date MySQL. O interfață de tip ODBC denumită MyODBC permite altor limbaje de programare ce folosesc această interfaţă, să utilizeze bazele de date MySQL cum ar fi ASP sau Visual Basic (Molnar).

În multe cărți de specialitate este precizat faptul că MySQL este destul de ușor de învățat și folosit în comparație cu multe din aplicațiile de gestiune a bazelor de

date, ca exemplu comanda de ieşire fiind una simplă și evidentă: „exit" sau „quit".

Pentru a administra bazele de date MySQL se poate folosi modul linie de comanda sau se pot descărca de pe internet diferite programe ce creează o interfață grafică: MySQL Administrator şi MySQL Query Browser. Un alt instrument de management al acestor baze de date este aplicaţia SQL Manager.

Serverul de baze de date MySQL este foarte rapid, fiabil și ușor de utilizat. Inițial a fost dezvoltat pentru a manipula baze de date de dimensiuni mari mult mai rapid decât soluțiile existente.

MySQL Database Software este un sistem client/server ce constă într-un server MySQL multi-threaded care suportă diferite programe client și biblioteci, unelte administrative şi o gamă largă de interfețe pentru programarea aplicațiilor.

1.4 Python

Python este un limbaj de programare interpretat, orientat spre obiecte, la nivel înalt, cu semantică dinamică. Structurile sale de date încorporate la nivel înalt, combinate cu tastarea dinamică și legarea dinamică, îl fac foarte atractiv pentru dezvoltarea rapidă a aplicațiilor, precum și pentru utilizarea ca limbaj de scriptare sau lipici pentru a conecta componentele existente împreună. Sintaxa simplă, ușor de învățat a Python accentuează lizibilitatea și, prin urmare, reduce costul întreținerii programului. Python acceptă module și pachete, ceea ce încurajează modularitatea programului și reutilizarea codului. Interpretul Python și biblioteca standard extinsă sunt disponibile sub formă sursă sau binară fără taxe pentru toate platformele majore și pot fi distribuite în mod liber.

Adesea, programatorii se îndrăgostesc de Python din cauza productivității crescute pe care o oferă. Deoarece nu există nicio etapă de compilare, ciclul de editare-test-depanare este incredibil de rapid. Depanarea programelor Python este ușoară: o eroare sau o intrare greșită nu va provoca niciodată o eroare de segmentare. În schimb, atunci când interpretul descoperă o eroare, ridică o excepție. Când programul nu prinde excepția, interpretul imprimă o urmă de stivă. Un depanator la nivel de sursă permite inspectarea variabilelor locale și globale, evaluarea expresiilor arbitrare, setarea punctelor de întrerupere, trecerea prin cod o linie la un moment dat și așa mai departe. Depanatorul este scris chiar în Python, mărturisind puterea introspectivă a lui Python. Pe de altă parte, deseori cel mai rapid mod de a depana un program este să adăugați câteva instrucțiuni de tipărire la sursă: ciclul rapid de editare-test-depanare face această abordare simplă foarte eficientă.

1.5 Flask

Flask este un framework web scris în Python care este clasificat ca un microframe, deoarece nu necesită anumite instrumente sau biblioteci.Nu are un strat de abstractizare a bazei de date, validarea formularelor sau orice alte componente în care bibliotecile terțe preexistente oferă funcții comune. Cu toate acestea, Flask acceptă extensii care pot adăuga caracteristici ale aplicației ca și cum ar fi implementate în Flask în sine. Există extensii pentru mapere relaționale obiect, validarea formularelor, gestionarea încărcărilor, diverse tehnologii de autentificare deschise și mai multe instrumente comune legate de cadru.

Capitolul 2

Capitolul 4: Tehnologiile de pe Front-end

2.1 React

Des considerat un JavaScript framework precum Angular sau Vue.js, React este de fapt o biblioteca de tip open source. Este folosit mai ales pentru interfate web mari, complexe dar si pentru aplicatii single-paged. Creat de Jordan Walke, un software engineer la Facebook, a fost implementat rapid in news feed-ul Facebook, in 2011. Un an mai tarziu, Instagram, aplicatia detinuta de Facebook, a implementat React si de aici incepe povestea. In ziua de azi, sute de mii (poate chiar milioane) de website-uri sunt sutinute de aceasta librarie si alte mii se nasc in fiecare zi. De fapt, de la lansarea React, am observat o crestere exploziva in utilizarea de librarii JavaScript mici ca dimensiuni, dar puternice. Utilizatorii vor sa utilizeze din ce in ce mai mult pagini web mai rapide si mai dinamice, in timp ce dezvoltatorii opteaza pentru medii moderne si flexibile fara tone de linii de cod in pachet. De aceea ReactJS este o alegere evidenta pentru multi. Ca sa explicam de ce, haideti sa recapitulam principalele motive pentru care folosim React. Desi considerat un JavaScript framework precum Angular sau Vue.js, React este de fapt o biblioteca de tip open source. Este folosit mai ales pentru interfate web mari, complexe dar si pentru aplicatii single-paged.

Primul lucru care face atat de multi oameni sa foloseasca ReactJS in proiectele lor este probabil simplitatea sa. React este o biblioteca JavaScript, astfel incat daca un dezvoltator este familiarizat cu functiile JS, va avea un start mai usor cu ReactJS. Cu această biblioteca, dezvoltatorii definesc interfete cu o sintaxa asemanatoare HTML numită JSX. Drept urmare, este produs cod HTML și CSS. API-ul React este foarte mic, dar puternic si tot ce trebuie safaci inainte de a incepe este sa inveti cateva functii de baza. Un pic din curba de învatare apare cand doriti sa utilizati React cu

alte biblioteci JS, cum ar fi Redux, Material UI sau Enzyme. Desi nu fac parte din stiva React, astfel de biblioteci adauga functii suplimentare si va permit sa gestionati mai usor componentele React. Cele mai comune biblioteci sunt bine documentate si nu ar trebui sa creeze probleme niciunui dezvoltator.

React JS folosește câteva extensii numite JSX și Virtual DOM pentru a crea interfețe utilizator. Acestea permit dezvoltatorilor să le creeze pentru majoritatea platformelor și, datorită faptului că pot vedea rezultatele codului lor instantaneu, pot avea o vizualizare și o înțelegere mai bună a ceea ce fac.React JS folosește ceva numit JSX. Aceasta este o extensie a React care permite utilizarea HTML cu JavaScript și acest lucru face ca codul dvs. să fie mai versatil. React este compatibil cu majoritatea browserelor moderne, ceea ce îi ajută pe dezvoltatori să își schimbe și să testeze DOM pe diferite platforme.Virtual DOM: DOM este prescurtarea pentru Document Object Model, care este o interfață de program de aplicație (API). Permite programelor să citească conținutul oricărui site web, astfel încât să poată fi modificat în funcție de preferințele și nevoile programatorului. Orice site web care nu folosește React JS folosește HTML pentru a modifica și modifica DOM-ul său. Datorită implementării JSX, React JS poate avea un DOM virtual, care este o copie a DOM-ului original utilizat de aplicație sau site-ul web. Principala diferență dintre DOM și DOM-ul virtual este că primul arată doar modificările după ce pagina este încărcată din nou, în timp ce acesta din urmă le arată în timp real, fără a fi nevoie să reîncărcați.

2.2 TypeScript

TypeScript a fost făcut public pentru prima dată în octombrie 2012 (la versiunea 0.8), după doi ani de dezvoltare internă la Microsoft. La scurt timp după anunț, Miguel de Icaza a lăudat limba însăși, dar a criticat lipsa de sprijin IDE matur în afară de Microsoft Visual Studio, care nu era disponibil pe Linux și OS X în acel moment. Astăzi există sprijin în alte IDE, în special în Eclipsă, printr-un plug-in contribuit de Palantir Technologies. Diversi editori de text, inclusiv Emacs, Vim, Webstorm, Atom și a Microsoft Cod Visual Studio acceptă, de asemenea, TypeScript.TypeScript este o limbaj de programare dezvoltat și întreținut de Microsoft. Este un sintactic strict superset de JavaScript și adaugă opțional tastarea statică la limbă. TypeScript este conceput pentru dezvoltarea de aplicații mari și transcompilează la JavaScript.Deoarece TypeScript este un superset de JavaScript, programele JavaScript existente sunt, de asemenea, programe TypeScript valabile.TypeScript poate fi folosit pentru a dezvolta aplicații JavaScript pentru ambele partea clientului și partea de server executare (ca și în cazul Node.js sau Deno). Există mai multe opțiuni disponibile pentru transcompilare. Poate fi folosit fie TypeScript Checker implicit, sau Babel compilatorul poate fi invocat pentru a converti TypeScript în JavaScript.TypeScript

acceptă fișiere de definiție care pot conține informații de tip ale bibliotecilor JavaScript existente, la fel C ++ fișierele antet poate descrie structura existentului fișiere obiect. Aceasta permite altor programe să utilizeze valorile definite în fișiere ca și cum ar fi entități TypeScript tipizate static. Există fișiere antet terțe pentru biblioteci populare, cum ar fi jQuery, MongoDB, și D3.js. Anteturi TypeScript pentru Node.js sunt de asemenea disponibile module de bază, care permit dezvoltarea programelor Node.js în TypeScript.

Capitolul 3

Specificari Back-end

Pe partea de back end sunt prezente doua servere, unul scris in JavaScript cu ajutorul framework-ului Node js care serveste partea de front-end cu toate datele de care are nevoie si un server scris in Python cu ajutorul framework-ului Flask cu ajutorul caruia aflu filme recomandate pentru un film sau o lista de filme.

3.1 Baza de date

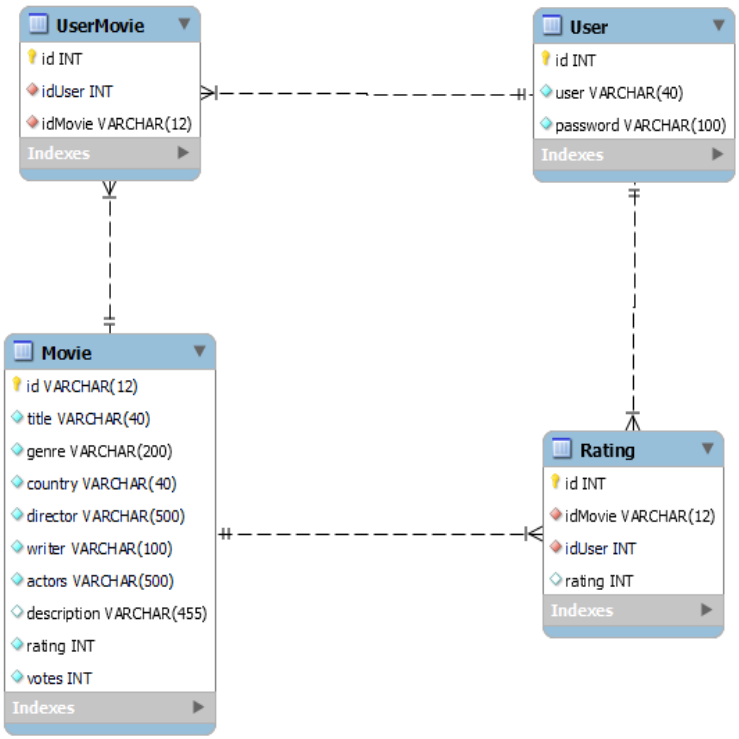

Figura 3.1: Diagrama baza de date.

Un lucru foarte important in dezvoltarea aplicatiei a fost alegerea bazei de date, am optat pentru pentru MySql versiunea 8.0. In tabelul de User voi salva datele

de inregisrare a unui utilizator , in tabelul Movie sunt prezente datele care descriu un film prin colloanele:title,genre,country,directors,writers,actors,description,votes si rating, in tabelul Rating salvez datele unui rating dat de un user pe laga valoare ratingului salvez si id-urile de la film si user pentru a stii cine da rating si la ce film totodata pe partea de back-end fac si o actualizare a rating ului si numarului de voturi din tabelul Movie asta pentru o a face aplicatia sa se miste mai rapid, iar in tabelul UserMovie salvez id ul de la un user si id ul de la un film reprezentand un film pe care utilizatorul l-a vizualizat.

3.2 Server Node Js

Serverul de Node Js il folosesc pentru a face inregistrare, login, returanrea tuturor filmelor care contin un numit sub titlu, adaugarea unui film la lista utilizatorului, returnarea a tuturor filmelor unui utilizator si adaugarea de rating a unui film. Cu ajutorul pachetului Express stabilesc rutele si pornesc serverul. Cand se face un request pe o ruta, se apeleaza functia asignata din Service, care are rol de a modela datele in functie de cum avem nevoie. Pentru a comunica cu baza de date ne folosim de Repository, acesta face legatura cu baza de date.

3.2.1 Inregistrare

Procesul de inregistrare este unul simplu, pasii fiind urmatorii:

1. Se trimit datele catre server
2. Se valideaza datele primite
3. Daca datele sunt valide se verifica daca numele utilizatorului se afla in baza de date deoarece numele trebuie sa fie unic pentru fiecare utiliztor
4. Daca totul este bine pana la acest pas se face has la parola pentru o mai buna securitate

```
//Hash password
const salt = await bcrypt.genSalt(10);
const hasedPassword = await bcrypt.hash(password, salt);
const user = new User(username, hasedPassword);
```

Figura 3.2: Has la parola.

5. Următorul pas este de a apela functția din repositoy care adaugă un user in baza de date.

```
const addUser = async (user) => {
  const { username, password } = user;
  const command = `INSERT INTO User (user, password) VALUES ('${username}', '${password}')`;
  const prom = new Promise((resolve, reject) => {
    pool.query(command, (err, data) => {
      if (err) throw Error;
      resolve(data);
    });
  });
};
```

Figura 3.3: Adaugare user in baza de date.

6. Dacă adăugarea s-a efectuat cu succes se returneaza un messaj de succes altfel se returnează un mesajul de eroare.

3.2.2 Autentificare

Pașii pentru procesul de autentificare sunt următorii

1. Se trimit datele catre server
2. Se valideaza datele primite
3. Se verifica daca username-ul se afla in baza de date

```
const findUser = async (username) => {
  const command = `Select * from User Where user = '${username}'`;
  const prom = new Promise((resolve, reject) => {
    pool.query(command, (err, data) => {
      if (err) throw Error;
      resolve(data);
    });
  });
  return prom;
};
```

Figura 3.4: Cautare dupa username in baza de date

4. Daca se găsește user-ul in baza de date se verifică parola trimisă in requst cu cea din baza de date apoi se trimite raspunsul in functie de potrivirea celor doua parole, daca rapunsul este valid se va trimite id-ul user-ului care va reprezenta token-ul pentru front-end, daca parolele nu corespund se va trimite status 400 si un mesaj de eroare.

```
//Verify
const databaseUser = new User(userList[0].user, userList[0].password);
const id = userList[0].id;
const validPass = bcrypt.compare(databaseUser.password, password);
if (!validPass) response.status(400).send({ error: "Password is incorect" });
response.status(200).send({ status: "ok", token: id });
```

Figura 3.5: Verificare parola si trimitere raspuns

3.2.3 Lista filmelor care contin un subtitlu

Pașii pentru returnarea listii sunt:

1. Dupa se se apeleaza ruta specifica se ia din body titlul
2. Se apeleaza functioa din repository care returneaza lista
3. In repository se iau toate filmele din baza de date apoi se parcurg pentru a pastra doar filmele care contin subtitlul dat.

```
const searchMovies = async (title) => {
  const command = `SELECT * FROM Movie`;
  const prom = new Promise((resolve, reject) => {
    pool.query(command, (err, data) => {
      if (err) reject(err);
      const result = data.reduce((acc, elem) => {
        if (elem.title.includes(title)) acc.push(elem);
        return acc;
      }, []);
      resolve(result);
    });
  });
  return prom;
};
```

Figura 3.6: Functia din repository care cauta filmele