**UNIVERSITATEA BABEŞ-BOLYAI CLUJ-NAPOCA**
**FACULTATEA DE MATEMATICĂ ŞI INFORMATICĂ**
**SPECIALIZAREA INFORMATICĂ ÎN LIMBA ROMÂNĂ**

# LUCRARE DE LICENŢĂ

# Conectarea persoanelor bazată pe interese comune folosind sisteme de recomandare

**Conducători științifici**

**Asist. Drd. Coste Claudia-Ioana**
**Prof. univ. Dr. Andreica Anca-Mirela**

*Absolvent*
*Bucilă Mihai-Cristian*

2025

ABSTRACT

Abstract: un rezumat în limba engleză cu prezentarea, pe scurt, a conținutului pe capitole, punând accent pe contribuțiile proprii și originalitate

# Cuprins

# Capitolul 1

# Introducere

## 1.1 Motivație

## 1.2 Obiective

## 1.3 Structura lucrării

Această lucrare este structurată în 5 capitole care prezintă atât aspecte teoretice, cât și aspecte practice ale subiectului abordat. Capitolul 1 prezintă o introducere a temei propuse, cu atenția îndreptată spre obiectivele lucrării și motivația acesteia. În capitol 2 se oferă detalii despre câteva soluții existente ce rezolvă problema dată. Capitolul 3 este dedicat bazei teoretice. Soluția software se descrie în capitolul 4. În capitolul 5 sunt prezentate concluziile și perspectivele de viitor.

# Capitolul 2

# Soluții similare în domeniul sistemelor de recomandare

Sistemele de recomandare sunt folosite la scară largă în multe aplicații, oferind conținut personalizat și o experientă de utilizare îmbunătățită. Aplicațiile ce le folosesc variază, de la platforme de comerț online la rețele sociale. Scopul acestora este de a ajuta utilizatorii să obțină conținut relevant din cantitatea mare de informație care există, simplificând procesul de căutare. Astfel de soluții contribuie semnificativ creșterea veniturilor companiilor și la fidelizarea utilizatorilor. Algoritmii folosiți în astfel de sisteme sunt filtrarea bazată pe conținut, filtrarea colaborativă sau soluții hibride, fiecare cu avantajele și limitările sale. Pe lângă preferințele redate direct de utilizatori, sistemele moderne de recomandare țin cont și de context: locație, dispozitiv folosit sau momentul zilei. Cu toate acestea, tehnicile se confruntă cu anumite provocări, precum problema utilizatorilor noi, lipsa de diversitate în recomandări sau lipsa de transparență a algoritmilor.

## 2.1 Platforme de vânzări

### Amazon

Amazon este una dintre cele mai cunoscute platforme de comerț care utilizează sisteme de recomandare. Acesti algoritmi sunt integrați în întreaga aplicație, de la pagina principală până la paginile individuale ale produselor. Platforma folosește tehnologii avansate de recomandare și de analiză a datelor pentru a afla comportamentul utilizatorilor. Datele relevante pentru sistem sunt: produsele vizualizate, adăugate în coș sau cumpărate, recenzii, istoricul căutărilor, dar și comportamentul altor utilizatori cu profil similar. Toate aceste date sunt folosite pentru a crea profiluri detaliate de utilizator, cu rol crucial în oferirea de recomandări personalizate. Pe baza acestora, Amazon implementatează metode precum filtrarea bazată

pe conținut (pentru a recomanda produse similare celor accesate în trecut) și filtrarea colaborativă (cu scopul de a sugera produse populare în rândul utilizatorilor cu interese comune). Sistemul lor de recomandare este considerat un factor cheie în maximizarea eficienței vânzărilor, ajutând la retenția clienților și stimulând cumpărăturile recurente.

### Netflix

Netflix este un exemplu de serviciu de tip streaming care implementează algoritmi de recomandare pentru a îmbunătăți experiența utilizatorilor și pentru a crește nivelul de satisfacție al acestora. Fără acest sistem, platforma nu s-ar bucura de succesul pe care îl are. La fel ca Amazon, Netflix folosește o combinație de tehnici, adaptate pentru a oferi sugestii personalizate. Sistemul ia în considerare date ce privesc durata și istoricul vizionărilor, rating-ul acordat filmelor și interacțiunile anterioare. Impactul acestor tehnologii este semnificativ, astfel că Netflix reușește să crească timpul petrecut în aplicație și să mărească gradul de fidelizare a utilizatorilor. Mai mult, recomandările ajută abonații să descopere titluri noi, pe care probabil nu le-ar fi căutat, sporind aprecierea lor pentru platformă.

## 2.2 Rețele sociale

### Facebook

Facebook este una dintre cele mai utilizate rețele sociale la nivel mondial. Sistemele de recomandare sunt esențiale în modul în care platforma afișează noutățile, sugerează prieteni sau grupează pagini relevante pentru fiecare persoană. Algoritmii aplicației analizează numeroși factori precum interacțiunile anterioare (reacții, comentarii, distribuiri), timpul petrecut pe anumite tipuri de postări, relațiile dintre utilizatori, dar și preferințele implicite, deduse din comportament. Aceste informații sunt prelucrate în timp real pentru a determina ce conținut este cel mai relevant. Se poate observa că acest mod de personalizare influențează modul în care utilizatorii percep realitatea digitală, întrucât conținutul afișat este filtrat și ordonat în funcție de relevanță, nu de cronologie sau obiectivitate.

Un alt aspect important este faptul că platforma nu oferă doar recomandări explicite (sugestii de prieteni sau grupuri), ci și implicite, prin modul în care organizează și filtrează fluxul de informații. Acestea optimizează și afișarea reclamelor, asigurând o potrivire mai bună între interesele utilizatorilor și ofertele companiilor, mărind astfel eficiența campaniilor publicitare și veniturile platformei. Astfel, persoanele care utilizează aplicația sunt predispuse să petreacă mai mult timp ex-

plorând conținut, interacționând cu prietenii și postările acestora, ceea ce determină un nivel ridicat de implicare și angajament față de platformă. Prin utilizarea acestor tehnologii, Facebook nu doar îmbunătățește experiența individuală, ci și crește implicarea utilizatorilor, contribuind la menținerea unei baze de date active.

Cu toate acestea, folosirea excesivă a algoritmilor poate conduce la crearea unor ”bule informaționale”, unde utilizatorii sunt expuși doar la un anumit tip de conținut, ce le confirmă convingerile existente, limitând diversitatea opiniilor și a informațiilor la care o persoană poate avea acces. În tot acest timp, personalizarea exagerată influențează percepția asupra realității și poate reduce capacitatea de a descoperi perspective noi sau diferite.

# Capitolul 3

# Fundamente teoretice în sisteme de recomandare

## 3.1 Filtrare bazată pe conținut

Filtrarea bazată pe conținut este o tehnică de recomandare care analizează atributele unui element sau ale unei persoane pentru a genera sugestii. În general, această metodă este utilizată cu scopul de a oferi recomandări personalizate în funcție de profilul utilizatorilor. Sistemul identifică similarități între profiluri și sugerează elemente sau persoane care corespund cel mai bine preferințelor utilizatorului. [1]

Această metodă este des utilizată în platforme precum Spotify, YouTube sau Netflix, în special pentru a recomanda conținut similar cu cel deja consumat. De exemplu, un utilizator care ascultă muzică jazz va primi mai frecvent recomandări din aceeași zonă muzicală, fără a fi nevoie de comparații cu preferințele altor utilizatori. Avantajul major al acestui tip de filtrare constă în caracterul său personalizat, axat strict pe gusturile fiecărui utilizator, fără a fi influențat de tendințele generale ale comunității.

Cu toate acestea, filtrarea bazată pe conținut are și unele limitări. Una dintre cele mai notabile este lipsa diversității — sistemul poate rămâne ”prizonier” în tiparele stabilite inițial și poate recomanda doar conținut foarte similar cu ceea ce a fost deja consumat. În plus, eficiența sistemului depinde de calitatea datelor și de granularitatea atributelor folosite pentru descrierea obiectelor.

### 3.1.1 Principiu de funcționare

Metoda se bazează pe construirea unui profil pentru fiecare utilizator, care reflectă preferințele sale în funcție de conținutul anterior accesat sau evaluat. Acest profil este comparat cu descrierile altor elemente disponibile, iar sistemul recomandă acele elemente care prezintă cele mai mari similarități. De regulă, se folosesc teh-

nici de procesare a limbajului natural, metode probabilistice sau modele de învățare automată. Una dintre particularitățile filtrării bazate pe conținut este faptul că nu are nevoie de date despre alți utilizatori pentru a funcționa. Recomandările sunt generate exclusiv pe baza preferințelor și comportamentului. Acest lucru oferă o flexibilitate ridicată, în special atunci când predilecțiile se schimbă. Pe de altă parte, această abordare presupune existența unor descrieri detaliate ale elementelor din sistem, ceea ce poate deveni o provocare atunci când aceste informații lipsesc sau sunt dificil de structurat. [2]

## 3.2 Filtrare colaborativă

## 3.3 Provocări și limitări ale metodelor de recomandare

Deși sistemele de recomandare bazate pe conținut și colaborative au un rol esențial în personalizarea experienței utilizatorilor, acestea se confruntă cu o serie de limitări care pot afecta eficiența și precizia recomandărilor. Printre cele mai întâlnite provocări se numără lipsa datelor suficiente pentru utilizatorii noi, dificultatea de a surprinde preferințele în schimbare și complexitatea modelării relațiilor între elemente sau utilizatori. În continuare sunt prezentate câteva dintre aceste probleme, împreună cu implicațiile lor asupra performanței sistemelor de recomandare.

### 3.3.1 Problema "Cold start"

Problema "Cold start" reprezintă o situație care este des întâlnită în domeniul sistemelor de recomandare. Aceasta apare atunci când nu există suficiente date despre un utilizator sau un obiect pentru a face recomandări precise. În cazul unui utilizator nou, sistemul nu dispune de informații suficiente despre acesta, făcând dificilă furnizarea sugestiilor. În mod similar, în cazul unui obiect nou, Astfel, se poate ajunge la recomandări inexacte ce afectează negativ experiența utilizatorului. Soluțiile pentru această problemă includ utilizarea tehnicilor de recomandare bazate pe conținut, care nu depind de interacțiuni anterioare, în detrimentul recomandărilor colaborative sau combinarea mai multor metode pentru o acuratețe îmbunătățită în etapele incipiente de folosire. Concret, sistemul ar trebui fie să ofere posibilitatea unui utilizator nou să evalueze anumite articole sau să întrebe explicit despre gusturile acestuia pentru a-i construi un profil, fie să dea recomandări preliminare bazate pe informații demografice sau alte date disponibile. [1]

### 3.3.2 Sinonimie

### 3.3.3 Confidențialitate și securitate

# Capitolul 4

# Aplicație software

Capitolul de față descrie aplicația software realizată, prezentând structura sa generală, principalele componente și modul în care acestea comunică pentru a oferi funcționalitățile propuse. Sunt oferite detalii despre arhitectură și sunt explicate în detaliu principalele funcționalități. Capitolul se încheie cu un scurt manual de utilizare, conceput să ghideze utilizatorul în folosirea aplicației.

## 4.1 Arhitectura aplicației

Aplicația este alcătuită din 3 componente: interfața utilizator (frontend), partea de server (backend) și sistemul de recomandare. Partea de frontend este dezvoltată în Ionic React[3], tehnologie ce îmbină librăria React[4], scrisă în limbajul de programare JavaScript, utilizată pentru a construi interfețe utilizator și Ionic[5], un instrument ce ajută la dezvoltarea aplicațiilor cross-platform. Folosind Ionic, aplicația poate fi instalată pe mai multe platforme — web, mobile (Android, iOS), desktop — utilizând același cod sursă. Această abordare eficientizează dezvoltarea și întreținerea aplicațiilor, permițând o experiență de utilizare uniformă, indiferent de dispozitiv folosit.

### 4.1.1 Frontend

Interfața utilizator a aplicației este compusă din rute, pagini și componente. Paginile corespund unor ecrane complete din aplicație, iar componentele sunt părți reutilizabile ale aplicației, precum un formular sau buton personalizat. Componentele primesc proprietăți și returnează elemente de interfață utilizator. Rutele definesc ce pagină trebuie afișată atunci când utilizatorul accesează o anumită adresă (URL). De exemplu, ruta `/login` afișează pagina de autentificare, ruta `/register` afișează pagina de înregistrare ș.a.m.d.

Frontend-ul comunică cu serverul prin apeluri HTTP, datele fiind transmise în formatul JSON. În cadrul aplicației a fost creat fișierul `api.ts` care grupează toate cererile care sunt trimise către backend. Acest serviciu oferă funcții precum `register`, `login` și `saveHobbies`. Aplicația folosește biblioteca Axios, o soluție populară și eficientă pentru manipularea cererilor HTTP.

### 4.1.2 Backend

Partea de server este scrisă în Java utilizând Spring Boot[6], un instrument care ajută la simplificarea procesului de dezvoltare al aplicațiilor. Acest framework permite crearea rapidă de API-uri REST, o integrare facilă cu baza de date și o organizare logică a codului în componente. Alegerea acestei tehnologii a fost motivată de ecosistemul extins pe care îl oferă, configurarea redusă și structura modulară.

Proiectul este organizat conform principiilor arhitecturii stratificate, adaptate pentru o aplicație REST. Așadar, codul sursă este împărțit în mai multe directoare (pachete) care separă responsabilitățile logice:

- controller: include clase care gestionează cererile HTTP venite de la frontend. Acestea definesc rutele API și interacționează direct cu serviciile.
- service: conține implementările operațiilor specifice fiecărei funcționalități (înregistrare utilizator, adăugare hobby etc.). Acest strat realizează legătura între controller și repository.
- repository: definește interfețe care extind `JpaRepository`. Acestea se concentrează asupra operațiilor CRUD (Create, Read, Update, Delete) cu efect direct asupra bazei de date.
- domain: conține entitățile din domeniul aplicației. Acestea sunt adnotate cu specificații precum `@Entity`, `@Table` etc. pentru a le asocia tabelelor din baza de date.

Pe lângă aceste pachete clasice, proiectul include și alte componente utile:

- dto (Data Transfer Object): conține clase care precizează formatul datelor transmise între client și server.
- mapper: cuprinde clase cu metode statice folosite la conversia obiectelor de tip domeniu în obiecte de tip DTO și vice-versa.
- exception: definește clase care se ocupă de tratarea erorilor. Acestea extind clasa `Exception` cu scopul de a crea excepții personalizate.
- security: gestionează autentificarea și autorizarea. Oferă implementare pentru JWT (Json Web Token).

### Model de date

Entitățile de bază ale aplicației sunt User, Hobby și HobbyType. Fiecare dintre acestea sunt reprezentate printr-o clasă Java, asociată unei tabele din baza de date. În figura 4.1 sunt definite clar relațiile între obiecte.

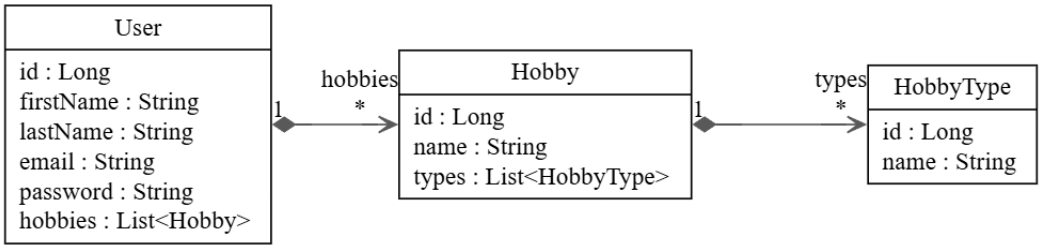


Figura 4.1: Diagrama de clase

### Baza de date

Conectarea la baza de date se face cu ajutorul fișierului de configurare `application.properties`, unde sunt specificate datele de acces precum URL, utilizator și parolă. Baza de date folosită este PostgreSQL[7], consacrată pentru fiabilitate și performanță. Pentru asocierea obiectelor Java cu tabelele bazei de date s-a utilizat Java Persistence API împreună cu framework-ul Hibernate[8], permițând manipularea entităților Java ca tabele din baza de date, așa cum se poate vedea în figura 4.2.

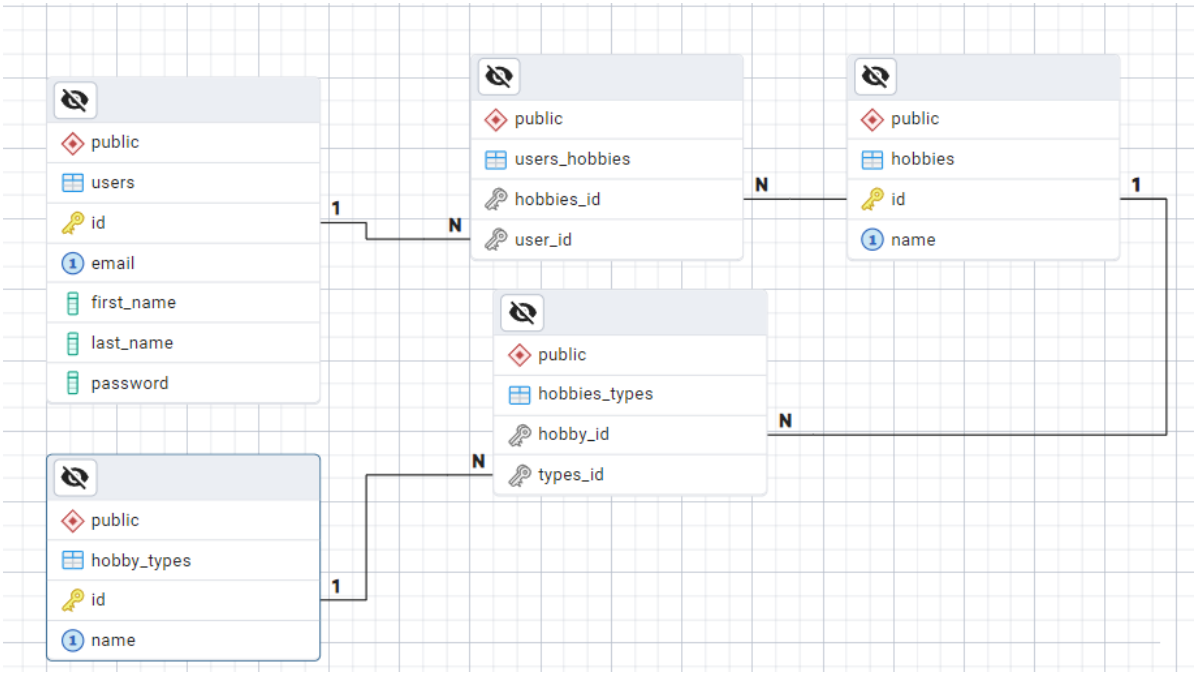


Figura 4.2: Diagrama bazei de date

#### Securitate

Aplicația implementează câteva mecanisme minimale de asigurare a securității și de protejare a accesului la resursele interne. În ceea ce privește parola asociată contului, aceasta este stocată sub formă criptată, folosind algoritmul BCrypt, iar pentru siguranța parolei, aplicația impune o lungime minimă de 8 caractere a acesteia. Atunci când utilizatorul se autentifică cu succes, primește un JSON Web Token (JWT). Acest token este folosit pentru identificarea ulterioară a utilizatorului în cererile către server. Deoarece majoritatea rutelor din aplicație sunt restricționate, utilizatorul are nevoie de un token valid pentru a le accesa, ceea ce asigură un control al accesului la funcționalitățile interne ale aplicației.

### 4.1.3 Sistemul de recomandare

#### Setul de date

Pentru a popula baza de date cu o listă de hobby-uri cât mai bogată și realistă, s-au fuzionat trei seturi de date disponibile pe platformele Kaggle[9], [10] și HuggingFace [11]. După îmbinarea acestora, a fost nevoie de o prelucrare atentă a setului de date pentru a standardiza formatul. Numele hobby-urilor au fost normalizate, iar în cazul în care au apărut duplicate, acestea au fost eliminate, însă tipurile lor au fost comasate. Tipurile hobby-urilor au fost la rândul lor prelucrate, deoarece erau exprimate inconsistent, separate fie prin virgulă, fie prin conjuncția ”and”, iar duplicatele au fost excluse. În urma procesării, s-a obținut un set de aproximativ 730 de hobby-uri care a fost introdus în baza de date pentru ca utilizatorii să poată selecta hobby-urile dintr-o listă structurată.

#### Comunicare cu serverul Java

Recomandările se realizează printr-o comunicare între serverului Java și sistemul de recomandare, care este implementat separat, scris în Python, folosind framework-ul Flask [12]. Acest serviciu extern este responsabil cu procesarea datelor și calculul gradului de asemănare. Atunci când utilizatorul accesează pagina de căutare a recomandărilor, aplicația trimite către serverul Python informații despre hobby-urile acestuia, precum și despre restul utilizatorilor existenți. Sistemul analizează datele și returnează o listă de utilizatori cu interese comune, ordonați în funcție gradul de similaritate.

Alegerea de a implementa separat acest sistem de recomandare a fost motivată de dorința de a păstra componenta independentă față de restul aplicației. Această abordare dă posibilitatea de a reutiliza codul într-un alt context, precum recomandarea de produse, cărți sau filme.

## 4.2 Funcționalități

Aplicația se concentrează asupra a patru acțiuni esențiale: înregistrarea unui cont, autentificarea unui utilizator, selectarea hobby-urilor dorite și căutarea altor utilizatori cu hobby-uri similare. Diagrama din figura 4.3 evidențiază aceste funcționalități.

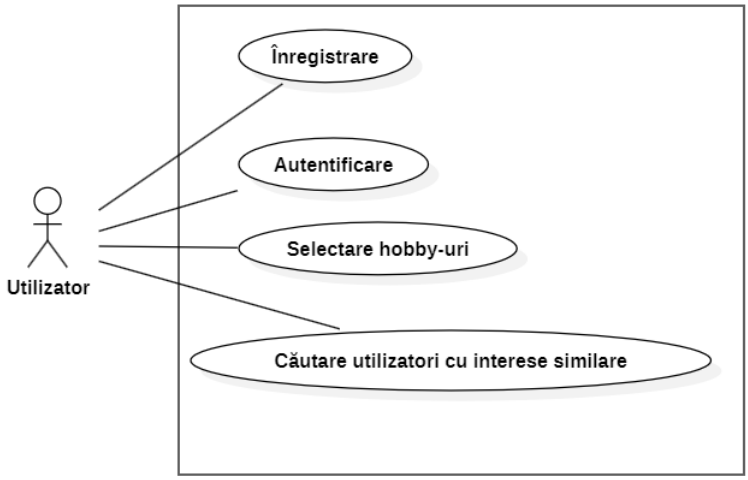


Figura 4.3: Diagrama cazurilor de utilizare

### 4.2.1 Înregistrarea utilizatorului

Această funcționalitate permite unui user nou să își creeze cont în aplicație. Pagina conține un formular cu câmpurile: nume, prenume, email sau parolă. La acționarea butonului Register, aplicația validează datele scrise de utilizator, iar în cazul în care nu există erori, trimite o cerere HTTP de tip POST către endpoint-ul `/auth/register` expus de backend. În caz contrar, programul oferă user-ului feedback imediat prin mesaje de eroare sugestive. Pe backend, datele se revalidează pentru a asigura securitatea și integritatea acestora. Contul utilizatorul se salvează în baza de date doar dacă nu există deja unul cu aceeași adresă de email, altfel se aruncă o excepție corespunzătoare. Serverul trimite răspuns, după caz, fie datele user-ului creat, fie un mesaj de eroare.

### 4.2.2 Autentificarea utilizatorul

Pentru a accesa restul funcționalităților, utilizatorul trebuie să se autentifice. Pagina dedicată logării cuprinde un formular cu două câmpuri: adresa de email și parolă.

În urma apăsării butonului Login, aplicația validează local datele introduse. În lipsa erorilor, se inițiază o cerere HTTP POST către endpoint-ul `/auth/login`. Pe partea de server, datele se verifică din nou, iar dacă acestea sunt corecte și corespund unui cont existent, se trimite un răspuns ce conține datele utilizatorului logat și un token JWT generat. În cazul unor date invalide (parolă incorectă, email inexistent), sistemul oferă un mesaj de eroare specific. După autentificare, token-ul JWT este stocat, iar utilizatorul este redirecționat către pagina principală.

### 4.2.3 Alegerea hobby-urilor

După ce utilizatorul este autentificat, aplicația verifică dacă acesta are hobby-uri. Dacă nu sunt identificate hobby-uri pentru user-ul logat, programul prezintă ecranul de alegere a hobby-urilor. Acesta conține un buton pentru salvare, o bară de căutare și listă de componente ce afișează numele și tipul fiecărui hobby, alături de un checkbox. Lista hobby-urilor este primită de la server prin intermediul unui apel HTTP de tip GET către endpoint-ul `/users/hobbies`. Utilizatorul poate filtra lista după numele hobby-urilor folosind bara de căutare. După ce user-ul selectează hobby-urile dorite și apasă butonul Save, se trimite o cerere HTTP de tip POST pentru a salva hobby-urile selectate.

### 4.2.4 Căutarea utilizatorilor cu hobby-uri similare

Dacă s-au găsit hobby-uri pentru utilizatorul autentificat, atunci aplicația afișează pagina de căutare a utilizatorilor cu hobby-uri asemănătoare. Aceasta prezintă un buton pentru căutare cu nume sugestiv. La apăsarea acestuia se trimite o cerere HTTP către endpoint-ul `/users/similar`. Backend-ul creează o listă cu ID-urile tutoror utilizatorilor și hobby-urile acestora, din care se elimină utilizatorul logat. Această listă și utilizatorul logat se transmit sistemului de recomandare, printr-o metodă HTTP către endpoint-ul `/recommend`. Sistemul de recomandare calculează similaritatea hobby-urilor dintre utilizatorul logat și ceilalți utilizatori și returnează ca răspuns lista de ID-uri ale utilizatorilor recomandați, memorând pentru fiecare un scor al potrivirii. Partea de server primește de la sistemul de recomandare lista sortată descrescător după acel scor. Pe baza fiecărui ID din listă, programul caută informațiile legate de utilizator (nume, e-mail și hobby-urile), pe care le asociază cu scorul primit. Lista completă este transmisă interfeței utilizator pentru afișare. Utilizatorului logat i se vor prezenta utilizatorii recomandați, vizualizând pentru fiecare numele, e-mailul, lista hobby-urilor și scorul similarității.

## 4.3 Manual de utilizare

La prima interacțiune cu aplicația, utilizatorul are acces la două ecrane, la cel de logare și la cel de înregistrare. În funcție de calitatea user-ului (nou sau înregistrat) poate alege să se înregistreze sau să se autentifice.

Pentru a se loga, utilizatorul trebuie să introducă email-ul și parola cu care s-a înregistrat în prealabil, așa cum se poate vedea în figura 4.4. La apăsarea butonului de autentificare, sistemul verifică datele și redirecționează utilizatorul către pagina principală sau notifică cu privire la erorile apărute.

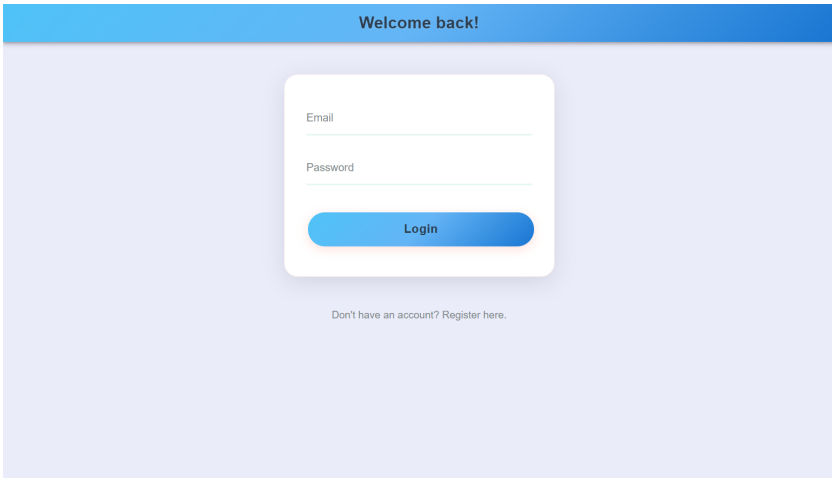


Figura 4.4: Pagina de autentificare

Figura 4.5 ilustrează pagina de înregistrare, unde utilizatorul trebuie să completeze formularul cu numele, adresa de e-mail și parola. La acționarea butonului de înregistrare, sistemul validează informațiile furnizate și creează contul utilizatorului sau afișează mesaje corespunzătoare în cazul unei erori. După o înregistrare cu succes, user-ul este redirecționat către pagina de logare.

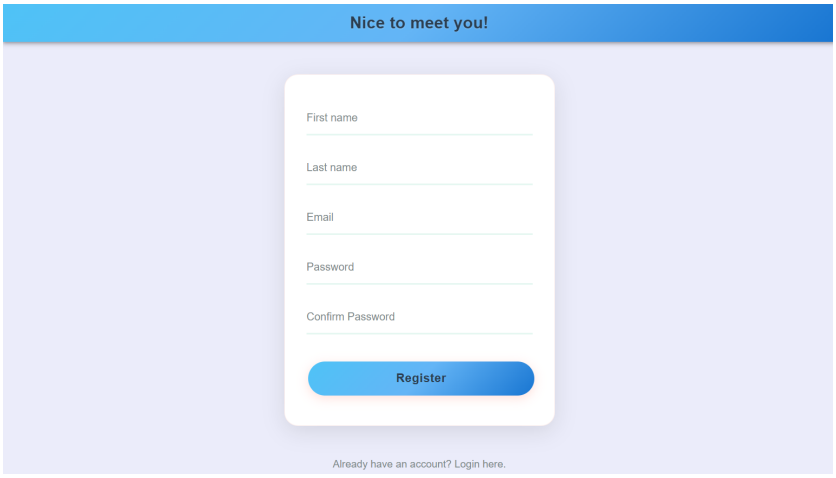


Figura 4.5: Pagina de înregistrare

În cazul în care utilizatorul logat nu are adăugate hobby-uri, un ecran dedicat acestei acțiuni va fi afișat imediat după logare, așa cum se poate observa în figura 4.6. Aici user-ul poate alege din lista de hobby-uri existente, prin intermediul unui checkbox. De asemenea, hobby-urile pot fi filtrate după nume, folosind bara de căutare. După selectare, aceastea pot fi salvate prin apăsarea butonului Save.

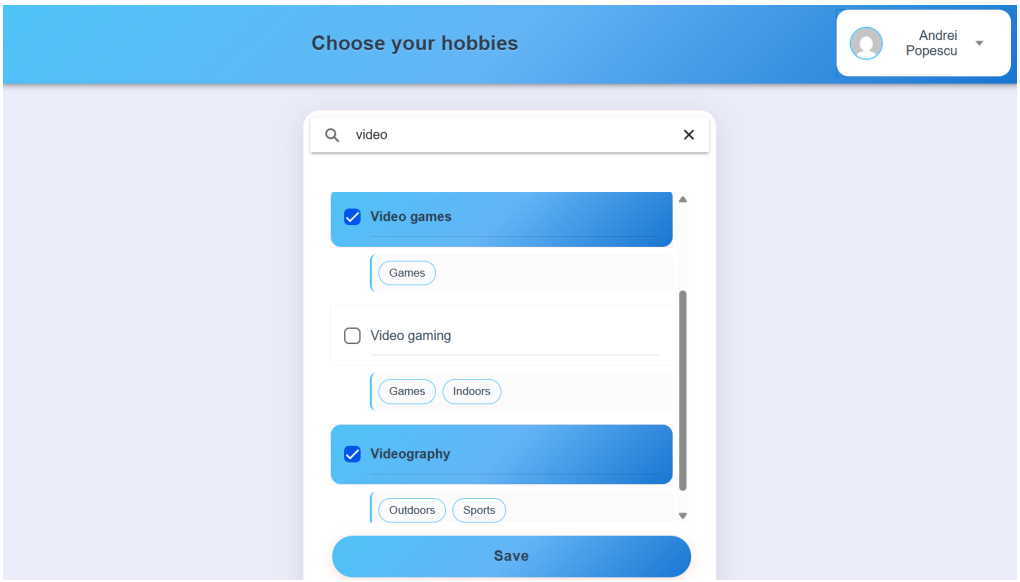


Figura 4.6: Pagina afișată utilizatorilor fără hobby-uri selectate

Dacă utilizatorul logat are hobby-uri selectate, se prezintă pagina de căutare a utilizatorilor cu hobby-uri asemănătoare, conform figurii 4.7. Pentru căutare, user-ul apasă butonul dedicat de pe ecran și se afișează lista utilizatorilor recomandați. Această listă este ordonată descrescător după scorul potrivirii.

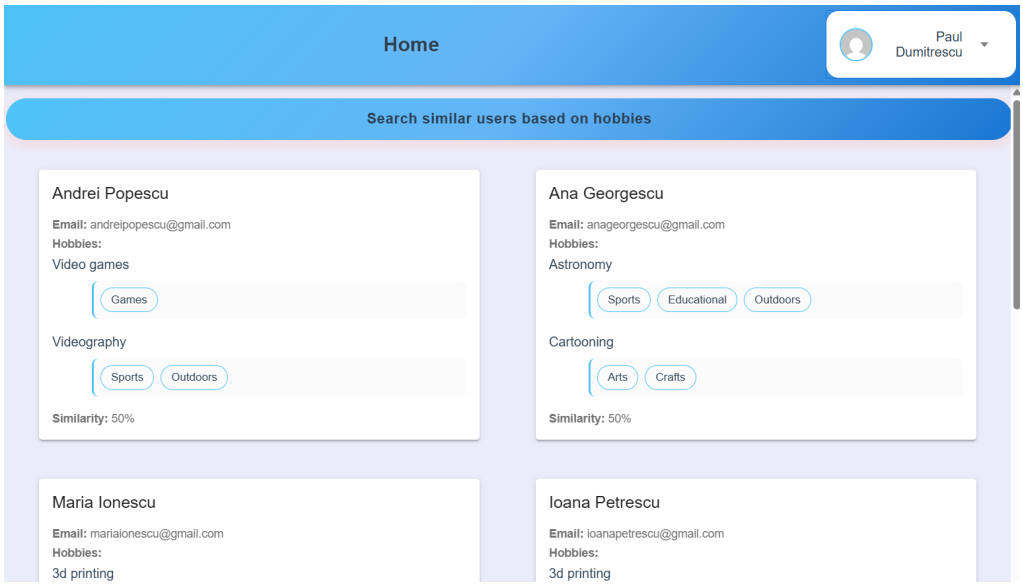


Figura 4.7: Pagina de căutare a utilizatorilor cu interese comune

De asemenea, utilizatorul se poate deconecta din aplicație. La apăsarea butonului ilustrat în figura 4.8, user-ul va fi redirecționat către pagina de autentificare.


Figura 4.8: Deconectarea utilizatorului

# Capitolul 5

# Concluzii

# Bibliografie


[1] P. Kumar and R. S. Thakur, "Recommendation system techniques and related issues: a survey," *International Journal of Information Technology*, vol. 10, pp. 495–501, 2018.

[2] F. Isinkaye, Y. Folajimi, and B. Ojokoh, "Recommendation systems: Principles, methods and evaluation," *Egyptian Informatics Journal*, vol. 16, no. 3, pp. 261–273, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1110866515000341

[3] D. Co., "Ionic React documentation," https://ionicframework.com/docs/react, 2024, accesat la: 2025-02-21.

[4] Meta Platforms, Inc. and affiliates, "React documentation," https://react.dev/reference/react, 2022, accesat la: 2025-02-21.

[5] D. Co., "Ionic documentation," https://ionicframework.com/docs, 2023, accesat la: 2025-02-21.

[6] Broadcom, "Spring Boot documentation," https://docs.spring.io/spring-boot/, 2024, accesat la: 2025-02-17.

[7] The PostgreSQL Global Development Group, "PostgreSQL documentation," https://www.postgresql.org/docs/, 2024, accesat la: 2025-02-19.

[8] Commonhaus Foundation, "Hibernate documentation," https://hibernate.org/orm/documentation/6.5/, 2024, accesat la: 2025-02-19.

[9] A. Raj, "A Comprehensive Collection of Hobbies," https://www.kaggle.com/datasets/mrhell/list-of-hobbies, 2022, accesat la: 2025-01-25.

[10] Dawid9632, "Hobbies & interests with categories," https://www.kaggle.com/datasets/dawid9632/hobbies-interests-with-categories, 2022, accesat la: 2025-01-25.

[11] A. Ugurcan, "Hobbies Dataset," https://huggingface.co/datasets/alperugurcan/Hobbies, 2024, accesat la: 2025-01-25.

[12] Pallets, “Flask documentation,” https://flask.palletsprojects.com/en/stable/, 2024, accesat la: 2025-03-03.