

**UNIVERSITATEA BABEȘ-BOLYAI CLUJ-NAPOCA  
FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ  
SPECIALIZAREA INFORMATICĂ ÎN LIMBA  
ROMÂNĂ**

## **LUCRARE DE LICENȚĂ**

**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

---

The abundance of information on the internet has driven the rapid evolution of recommender systems. Their ability to provide personalized suggestions has led to widespread adoption in high-demand areas like e-commerce and social networking platforms, enhancing both user experience and retention.

This thesis presents the core techniques underlying recommendation systems — content-based filtering and collaborative filtering — and explores the challenges that can arise when implementing these methods, such as the cold start problem, synonymy and privacy concerns. The main objective of this work is to demonstrate the utility of a recommender system in the context of an application that matches users based on their shared hobbies. The application analyzes user-provided hobby data and uses a simplified matching algorithm to identify potential matches within a shared interest space.

This work is motivated by the idea that recommender systems can have an impact beyond commercial platforms and enhance social connectivity and community building. By combining the theoretical knowledge with a concrete implementation, the project shows how recommendation techniques can be adapted to support social engagement and personal development. The application serves not only as a case study, but also as a proof of concept for using technology to bring people together in a more intentional way.

This paper is organized into five chapters that address both theoretical background and practical solution. The first chapter introduces the topic, exposing the main objectives and the motivation behind the research. The second chapter reviews several existing recommendation system solutions and studies. Chapter 3 provides the theoretical concepts which are essential to understand the proposed approach. Chapter 4 presents the developed software solution. Finally, the last chapter summarizes the work and mentions potential directions for future work.

# Cuprins

<b>1</b>	<b>Introducere</b>	<b>1</b>
1.1	Obiective . . . . .	1
1.2	Motivație . . . . .	1
1.3	Structura lucrării . . . . .	2
1.4	Utilizarea instrumentelor de inteligență artificială generative . . . . .	2
<b>2</b>	<b>Soluții similare în domeniul sistemelor de recomandare</b>	<b>4</b>
2.1	Platforme de vânzări și streaming . . . . .	4
2.2	Rețele sociale . . . . .	5
2.3	Abordări existente în conectarea utilizatorilor bazată pe interese . . . . .	6
<b>3</b>	<b>Fundamente teoretice în sisteme de recomandare</b>	<b>7</b>
3.1	Filtrare bazată pe conținut . . . . .	7
3.1.1	Principiu de funcționare . . . . .	8
3.2	Filtrare colaborativă . . . . .	8
3.2.1	Abordare tehnică . . . . .	8
3.3	Metrici de similaritate . . . . .	9
3.3.1	Coeficientul Jaccard . . . . .	9
3.3.2	Similaritatea cosinus . . . . .	9
3.3.3	Corelația Pearson . . . . .	9
3.4	Provocări și limitări ale metodelor de recomandare . . . . .	10
3.4.1	Problema „Cold start” . . . . .	10
3.4.2	Sinonimie și polisemie . . . . .	10
3.4.3	Confidențialitate și securitate . . . . .	11
<b>4</b>	<b>Aplicație software</b>	<b>12</b>
4.1	Arhitectura aplicației . . . . .	12
4.1.1	Frontend . . . . .	12
4.1.2	Backend . . . . .	13
4.1.3	Sistemul de recomandare . . . . .	16
4.2	Funcționalități . . . . .	17

4.2.1	Cerințe funcționale . . . . .	17
4.2.2	Cerințe nefuncționale . . . . .	19
4.3	Manual de utilizare . . . . .	19
4.4	Testare și validare . . . . .	22
4.4.1	Metodologia de testare . . . . .	22
4.4.2	Scenarii de testare . . . . .	22
4.4.3	Rezultate . . . . .	23
4.4.4	Limitări și îmbunătățiri . . . . .	23
<b>5</b>	<b>Concluzii</b>	<b>24</b>
	<b>Bibliografie</b>	<b>25</b>

# Capitolul 1

## Introducere

În contextul erei digitale, cantitatea de informație disponibilă în mediul online a crescut exponențial, generând nevoia de metode eficiente de filtrare și de selecție a conținutului. Astfel, sistemele de recomandare au devenit un element indispensabil în numeroase arii, precum comerț electronic, platforme de streaming și rețele sociale. Aceste sisteme oferă clienților sugestii personalizate, contribuind la o experiență de utilizare superioară și la retenția acestora. Lucrarea de față surprinde potențialul unui astfel de sistem într-un context diferit, acela al interacțiunilor umane bazate pe interese similare.

### 1.1 Obiective

Principalul obiectiv al lucrării este de a evidenția rolul și utilitatea sistemelor de recomandare într-o aplicație a cărui scop este asocierea între persoane pe baza intereselor comune. Lucrarea își propune să exploreze și să compare principalele metode utilizate în recomandare: filtrarea bazată pe conținut și filtrarea colaborativă. De asemenea, se vor analiza și principalele obstacole asociate acestor tehnici, cum ar fi problema sinonimiei, confidențialitatea și încrederea în sistem. În plus, proiectul urmărește să demonstreze aplicabilitatea acestor concepte prin punerea lor în practică într-un sistem funcțional.

### 1.2 Motivație

În zilele noastre interacțiunile umane devin din ce în ce mai mediate de tehnologie, astfel nevoia de a crea conexiuni relevante și veritabile între persoane devine tot mai evidentă. Deși scopul inițial al rețelelor sociale era de a aduce mai aproape persoanele unele de altele, în prezent ele nu mai răspund la fel de eficient dorinței utilizatorilor de a întâlni persoane cu interese similare. Mai mult, studii de la Uni-

versitatea din Pennsylvania [1] au stabilit că utilizarea îndelungată a rețelelor sociale duce la stări de singurătate și anxietate. Experimentul în care au fost implicați 143 de studenți, împărțiți în două grupuri (unul care și-a limitat utilizarea platformelor sociale la 30 de minute, timp de 3 săptămâni și altul de control) a arătat efectele nocive pe care folosirea necontrolată le are asupra indivizilor. În urma acestuia, au fost observate scăderi semnificative ale stărilor de depresie și anxietate, alături de o îmbunătățire a stimei de sine. Persoanele care au participat la acest studiu au remarcat cât de importantă este folosirea conștientă a acestor platforme.

Motivația personală din spatele acestei lucrări este dorința de a utiliza tehnologia nu doar pentru informare și divertisment, ci și pentru a stimula relațiile sociale. O cercetare din domeniu [2] confirmă ideea că tinerii formează relații de prietenie mai puternice cu persoane care au interese apropiate. Autorii arată cum sportul, muzica și activitățile extracurriculare facilitează interacțiunile sociale. Toate aceste dovezi susțin ideea unui sistem de asociere bazat pe hobby-uri pentru promovarea unui tip de interacțiune pozitivă bazată pe existența unor pasiuni comune între indivizi.

### **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, menționând studii și cercetări pentru susținerea acesteia. În capitol 2 se oferă detalii despre câteva soluții existente din domeniul sistemelor de recomandare, realizând o analiză comparativă ale acestor abordări. Capitolul 3 este dedicat principiilor fundamentale ale sistemelor de recomandare, oferind un punct de plecare pentru înțelegerea și implementarea soluției propuse. Soluția software se descrie în capitolul 4, care prezintă arhitectura sistemului, principalele funcționalități și un scurt ghid de utilizare. În capitolul 5 sunt prezentate concluziile și perspectivele de viitor.

### **1.4 Utilizarea instrumentelor de inteligență artificială generative**

Pe parcursul elaborării lucrării, au fost folosite unele instrumente bazate pe IA generativă, precum ChatGPT, Claude și GitHub Copilot. Aceste unelte s-au dovedit utile pentru realizarea unor căutări avansate legate de subiectele specifice și pentru reformularea ideilor în scopul coerenței și clarității. În zona dezvoltării software, Copilot și ChatGPT au contribuit la identificarea și remedierea rapidă a erorilor, dar și la

verificarea funcționalităților implementate prin reviziunea automată a modificărilor în cadrul sistemului de control al versiunilor. Este important de menționat că aceste instrumente nu înlocuiesc aportul uman asupra procesului de cercetare și scriere academică. Din acest motiv, responsabilitatea pentru validarea informațiilor, corectitudinea soluțiilor tehnice și asigurarea originalității aparține în totalitate autorilor lucrării.



## 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 experiență 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 la 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 și streaming

### Amazon

Amazon este una dintre cele mai cunoscute platforme de comerț care utilizează sisteme de recomandare. Acești 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 implementează 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 [3, 4].

## **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 [5, 6].

## **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 [7].

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, mărinid 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 [8].

Cu toate acestea, folosirea excesivă a algoritmilor poate conduce la crearea unor „bule informaționale” [9], 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.

## **2.3 Abordări existente în conectarea utilizatorilor bazată pe interese**

O direcție importantă în cadrul sistemelor de recomandare este reprezentată de metode de asociere între utilizatori, în funcție de interesele comune. Aceste abordări sunt des întâlnite în aplicații de socializare, unde rolul lor este de a facilita conexiunile relevante între persoanele cu profiluri similare.

În unul dintre articolele de specialitate [10], autorii propun să abordeze problema recomandării de prietenii prin dezvoltarea unei metode avansate de calcul a similarității între utilizatori. Obiectivul principal al algoritmului lor este de a conecta persoane cu interese comune care se află în aceeași zonă geografică, combinând astfel compatibilitatea bazată pe hobby-uri cu proximitatea fizică pentru a facilita interacțiunea dintre utilizatori. Metodologia propusă îmbunătățește metricile clasice de similaritate, integrând atât hobby-urile propriu-zise, cât și un rating pe care utilizatorii îl dau, nuanțare care ajută la evaluarea mai precisă a apropierii dintre două persoane. Autorii au evaluat performanțele metodei proiectate pe un eșantion de 286 de utilizatori prin compararea lor cu alte metode clasice de calcul ale similarității existente în literatură, obținând rezultate superioare în contextul specific.

# Capitolul 3

## Fundamente teoretice în sisteme de recomandare

Acest capitol prezintă fundamentele teoretice care stau la baza sistemelor de recomandare, punând accent pe metodele utilizate în practică: filtrarea bazată pe conținut și filtrarea colaborativă. De asemenea, sunt remarcate provocările cu care aceste sisteme se confruntă, precum problema „cold start”, ambiguitățile semantice și aspecte legate de securitate și confidențialitate. Aceste concepte teoretice oferă cadrul necesar pentru înțelegerea și dezvoltarea unui sistem de potrivire între persoane.

### 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 [11].

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 tehnici 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 [12].

## 3.2 Filtrare colaborativă

Filtrarea colaborativă este o metodă eficientă de recomandare, care se bazează pe preferințele utilizatorilor pentru a oferi sugestii relevante. În contrast cu filtrarea bazată pe conținut, filtrarea colaborativă utilizează date colectate din interacțiunile utilizatorilor cu scopul de a identifica puncte comune între aceștia. Principiul presupune că persoanele cu gusturi și comportamente asemănătoare vor aprecia și recomanda elemente similare. Există două modalități de abordare în acest tip de filtrare: filtrarea bazată pe utilizatori și filtrarea bazată pe itemi. Prima variantă generează recomandări pe baza preferințelor altor utilizatori cu profiluri similare, în timp ce a doua variantă analizează obiectele și le identifică pe acelea care au fost evaluate pozitiv de către utilizatorii asemănători [13].

### 3.2.1 Abordare tehnică

În mod concret, filtrarea presupune construcția unei matrici utilizator-item 3.1, unde rândurile reprezintă utilizatorii, iar coloanele obiectele (produse, cărți, filme etc.). Valorile din matrice reprezintă evaluările sau aprecierile persoanelor față de itemi. Pentru a determina similaritatea, se folosesc diferite metrice, precum Pearson, cosinus sau Jaccard. Pe baza acestora, algoritmul determină utilizatorii cei mai apropiați și creează un grup de persoane similare denumit vecinătate [11].

	Item <sub>1</sub>	Item <sub>2</sub>	.....	Item <sub>j</sub>	Item <sub>n</sub>
User <sub>1</sub>					
User <sub>2</sub>					
.					
User <sub>i</sub>					
User <sub>m</sub>					

Figura 3.1: Matrice utilizator-item [12]

### 3.3 Metrice de similaritate

Un pas esențial în oferirea unei recomandări este calculul similarității dintre utilizatori sau itemi. În continuare, vor fi prezentate cele mai importante metode de evaluare a asocierilor [14].

#### 3.3.1 Coeficientul Jaccard

Coeficientul Jaccard [15] este o metrică folosită cu precădere la compararea seturilor. Este definit ca raportul între mărimea intersecției și cea a uniunii celor două seturi 3.1.

$$Jaccard(A, B) = \frac{|A \cap B|}{|A \cup B|} \quad (3.1)$$

#### 3.3.2 Similaritatea cosinus

Similaritatea cosinus [16] este metodă utilizată frecvent pentru a măsura asemănarea dintre doi vectori. Formula 3.2 se bazează pe unghiul dintre vectorii respectivi într-un spațiu n-dimensional, unde  $A_i$ , respectiv  $B_i$  reprezintă valorile vectorilor de pe poziția  $i$ .

$$\cos(A, B) = \frac{\sum_1^n A_i \cdot B_i}{\sqrt{\sum_1^n A_i^2 \cdot \sum_1^n B_i^2}} \quad (3.2)$$

#### 3.3.3 Corelația Pearson

Corelația Pearson [16] exprimă gradul de relație liniară dintre două variabile vectoriale, fiind utilizată pentru a evalua variația a două seturi de date. Formula 3.3 este

adesea folosită pentru a compara intensități sau ratinguri, unde  $\bar{A}_i$  și  $\bar{B}_i$  reprezintă media aritmetică a valorilor vectorilor.

$$Pearson(A, B) = \frac{\sum_1^n (A_i - \bar{A}) \cdot (B_i - \bar{B})}{\sqrt{\sum_1^n (A_i - \bar{A})^2 \cdot \sum_1^n (B_i - \bar{B})^2}} \quad (3.3)$$

### 3.4 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.

#### 3.4.1 Problema „Cold start”

Problema „Cold start” [17] 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, când un obiect nou este adăugat în sistem, el nu este evaluat, deci nu poate fi recomandat. 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 [11].

#### 3.4.2 Sinonimie și polisemie

În cadrul sistemelor de recomandare, sinonimia apare atunci când doi termeni diferiți descriu același concept (de exemplu, „movie” și „film”, „football” și „soccer”). Acești termeni sunt tratați ca entități distincte, afectând performanța recomandărilor, deoarece similaritatea latentă dintre interese nu este exploatată. Pentru a contracara fenomenul, literatura de specialitate propune tehnici care extind vocabularul cu ter-

meni echivalenți sau identifică automat legături între date, astfel încât acestea să fie grupate în funcție de sensul lor. Totuși, soluțiile nu sunt perfecte, pentru că acești termeni pot avea sensuri ușor diferite față de cele intenționate și pot duce la o asociere irelevantă [18].

Pe lângă sinonimie, polisemia este o problemă similară care afectează generarea de recomandări. În acest caz, un singur termen are mai multe înțelesuri, în funcție de context, iar sistemul poate face asocieri eronate. Aceste două probleme favorizează apariția erorilor în recomandare și evidențiază nevoia unor mecanisme semantice de interpretare a datelor care să țină cont de nuanțele limbajului natural [19].

### 3.4.3 Confidențialitate și securitate

Confidențialitatea și securitatea sunt două concepte ce vizează efectele sistemului de recomandare asupra utilizatorilor. Dat fiind faptul că sistemele de recomandare se bazează pe colectarea și procesarea datelor sensibile, precum preferințe și istoric de vizualizare, acestea ridică probleme majore în ceea ce privește confidențialitatea și securitatea. Păstrarea acestor date fără protecție poate conduce la scurgeri de informații, afectând eficiența sistemului și încrederea utilizatorilor [20]. Pentru prevenirea expunerii datelor, un articol de specialitate recomandă utilizarea unor metode criptografice avansate, precum criptarea homomorfă, care permite sistemului să recomande fără ca el să aibă acces la informația reală a utilizatorilor [21].

Aceste măsuri nu protejează doar datele utilizatorilor în sine, ci și dezvoltă un nivel de încredere ridicat. Persoanele în cauză sunt sigure că datele lor nu sunt exploatate în mod negativ.



# 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 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. Aceste componente au roluri bine definite și comunică între ele pentru a asigura funcționarea aplicației. Frontend-ul gestionează interacțiunea cu utilizatorul, backend-ul coordonează logica aplicației și accesul la date, iar sistemul de recomandare oferă sugestii pe baza datelor oferite de server. Arhitectura, așa cum este prezentată în figura 4.1, prezintă structura modulară ce permite separarea clară a responsabilităților.

#### 4.1.1 Frontend

Partea de frontend este dezvoltată în Ionic React [22], tehnologie ce îmbină librăria React [23], scrisă în limbajul de programare JavaScript, utilizată pentru a construi interfețe utilizator și Ionic [24], 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.

Interfața utilizator a aplicației este compusă din rute, pagini și componente. Paginile corespund unor ecrane complete din aplicație, iar componentele sunt părți

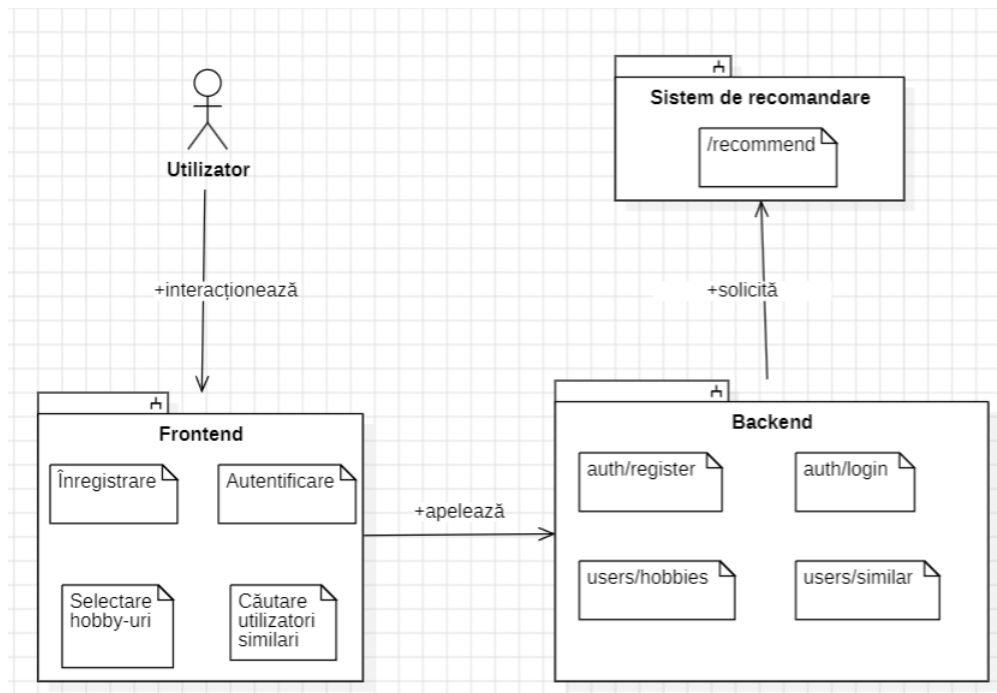


Figura 4.1: Diagrama de arhitectură

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 [25], 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 4.2 sunt definite clar relațiile între obiecte.

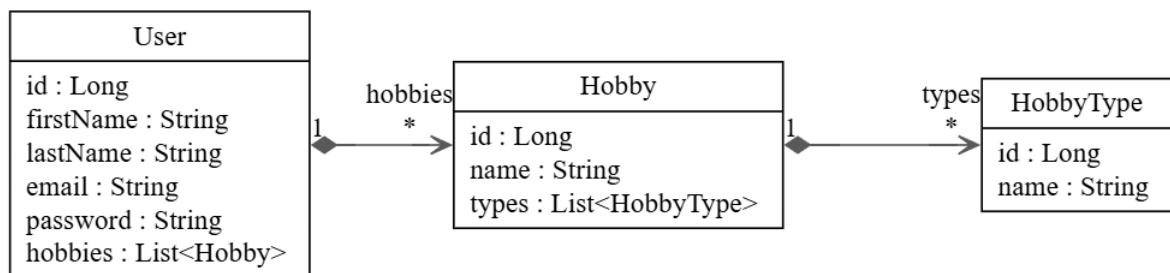


Figura 4.2: Diagrama de clase

## 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 [26], 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 [27], permițând manipularea entităților Java ca tabele din baza de date, așa cum se poate vedea în figura 4.3.

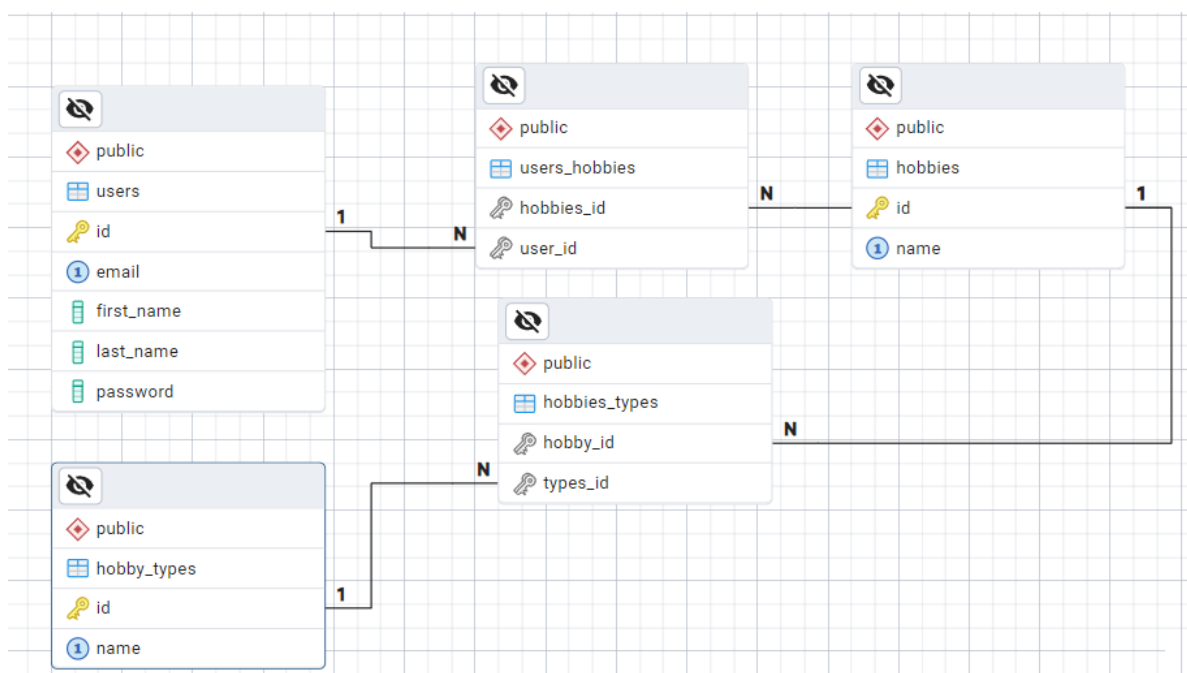


Figura 4.3: Diagrama bazei de date

## Securitate

Aplicația implementează câteva mecanisme minime 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

Pentru a determina gradul de similaritate între utilizatori pe baza hobby-urilor comune, sistemul folosește coeficientul de similaritate Jaccard. Această metodă compară seturile de hobby-uri ale fiecărui utilizator și calculează raportul dintre elementele comune și totalitatea elementelor distincte din cele două seturi, rezultând o valoare reală între 0 și 1. Metoda este simplă și intuitivă, fiind adecvată pentru compararea mulțimilor neordonate de termeni. Implementarea funcției care calculează similaritatea Jaccard este prezentată în secvența de cod 4.1.

Principalele argumente care au stat la baza alegerii acestei modalități de calcul al similarității sunt eficiența sa computațională, prin faptul că nu necesită resurse mari de calcul, dar și compatibilitatea sa cu datele binare, precum cele legate de hobby-uri (un hobby poate să fie prezent sau absent). De asemenea, există o simetrie în relația dintre utilizatori: similaritatea dintre utilizatorul A și B este aceeași cu cea dintre utilizatorul B și A, favorizând constituirea unei rețele coerente între persoane.

```

1 def jaccard_similarity(set1, set2):
2     if not set1 or not set2:
3         return 0.0
4     intersection = len(set1.intersection(set2))
5     union = len(set1.union(set2))
6     return intersection / union

```

Secvența de cod 4.1: Algoritmul de calcul al similarității Jaccard

#### 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 [28, 29] și Hugging-Face [30]. 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 serverul Java și sistemul de recomandare, care este implementat separat, scris în Python, folosind framework-ul

Flask [31]. 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 4.4 evidențiază aceste funcționalități.

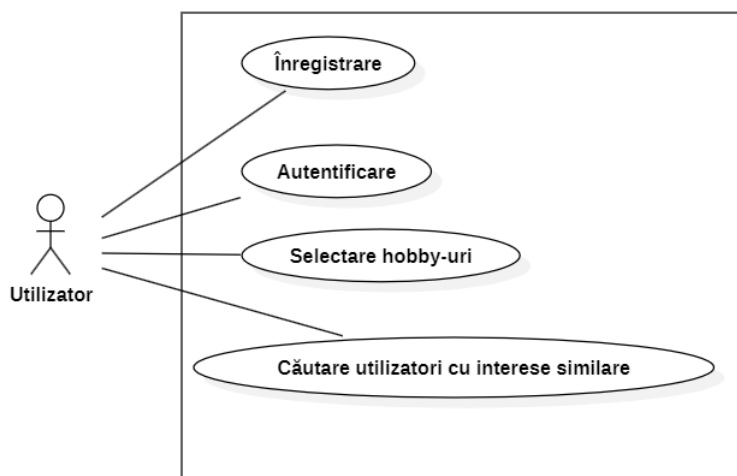


Figura 4.4: Diagrama cazurilor de utilizare

### 4.2.1 Cerințe funcționale

#### 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 utilizatorului 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.

### **Autentificarea utilizatorului**

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 redirectionat către pagina principală.

### **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.

### **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 tuturor 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.

### 4.2.2 Cerințe nefuncționale

Pe lângă funcționalitățile oferite utilizatorului, sistemul proiectat trebuie să respecte o serie de cerințe nefuncționale care asigură calitatea generală a produsului. Referitor la interfața utilizator, aceasta trebuie să fie intuitivă și ușor de utilizat pentru a se putea adresa unui număr mare de utilizatori. În plus, aplicația trebuie să fie portabilă datorită faptului că este construită cu ajutorul framework-ului Ionic. De asemenea, cerințele impun ca sistemul să aibă un mecanism robust de securitate în care accesul la funcționalitățile interne ale aplicației să fie restricționate și autorizate doar în prezența unor token-uri valide. În cele din urmă, se dorește ca aplicația să fie scalabilă, ușor de întreținut și de actualizat, pentru a permite extinderea funcționalităților în viitor.

## 4.3 Manual de utilizare

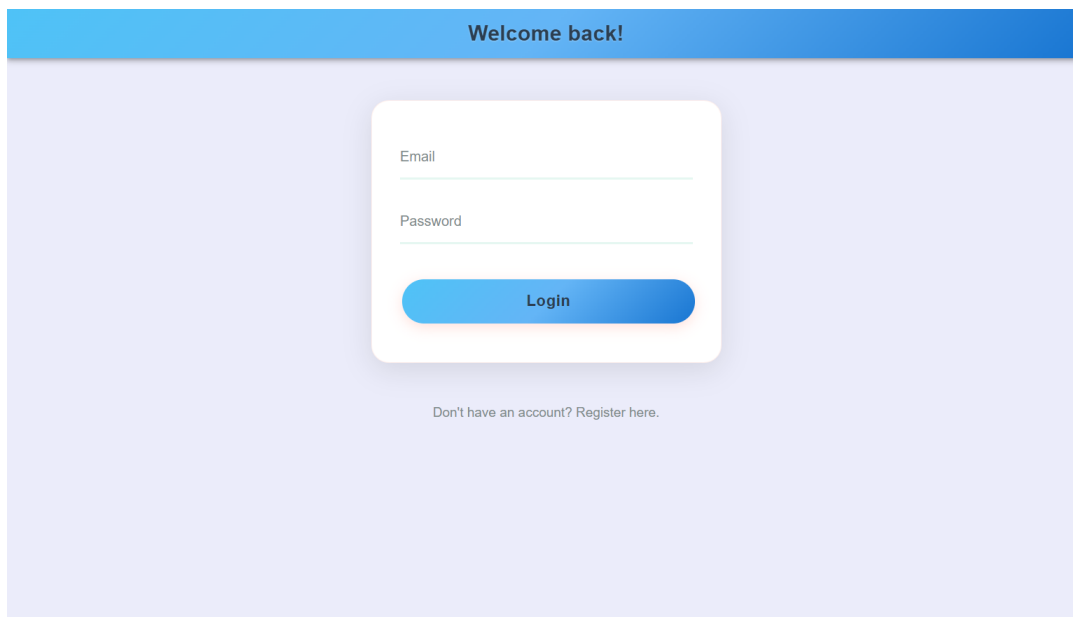
La prima interacțiune cu aplicația, utilizatorul are acces la două ecrane, la cel de logare și la cel de înregistrare. În funcție de calitatea user-ului (nou sau înregistrat) poate alege să se înregistreze sau să se autentifice.

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 4.5. La apăsarea butonului de autentificare, sistemul verifică datele și redirecționează utilizatorul către pagina principală sau notifică cu privire la erorile apărute.

Figura 4.6 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.

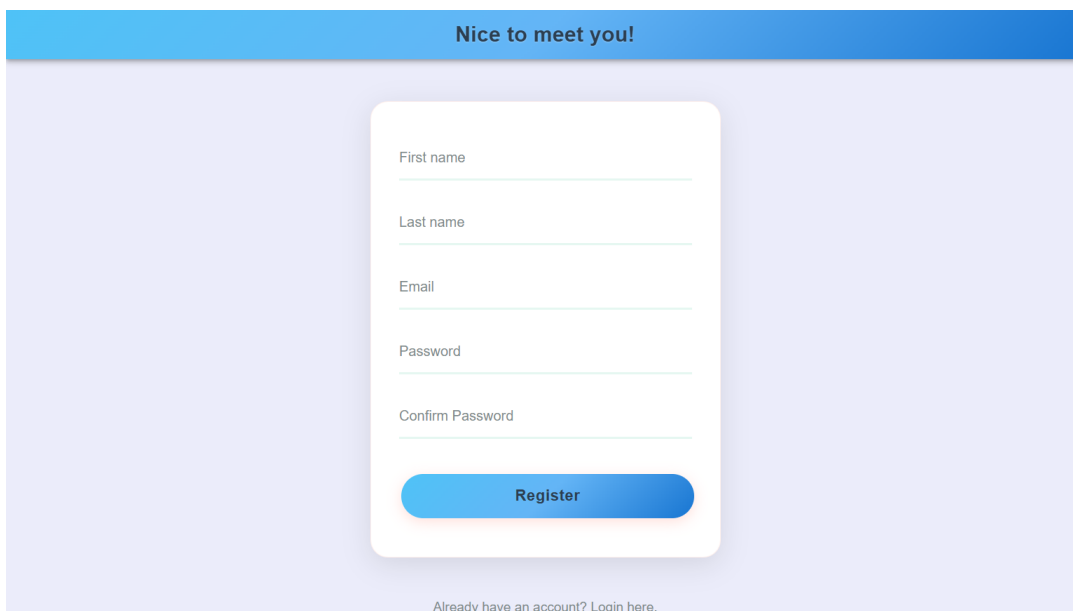
În cazul în care utilizatorul logat nu are adăugate hobby-uri, un ecran dedicat acestei acțiuni va fi afișat imediat după logare, așa cum se poate observa în figura





The login page features a blue header with the text "Welcome back!". Below the header is a light purple background. In the center, there is a white rounded rectangle containing two input fields labeled "Email" and "Password". Below these fields is a blue rounded button labeled "Login". At the bottom of the white rectangle, there is a link that says "Don't have an account? Register here."

Figura 4.5: Pagina de autentificare



The registration page features a blue header with the text "Nice to meet you!". Below the header is a light purple background. In the center, there is a white rounded rectangle containing five input fields labeled "First name", "Last name", "Email", "Password", and "Confirm Password". Below these fields is a blue rounded button labeled "Register". At the bottom of the white rectangle, there is a link that says "Already have an account? Login here."

Figura 4.6: Pagina de înregistrare

4.7. 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, acestea pot fi salvate prin apăsarea butonului Save.

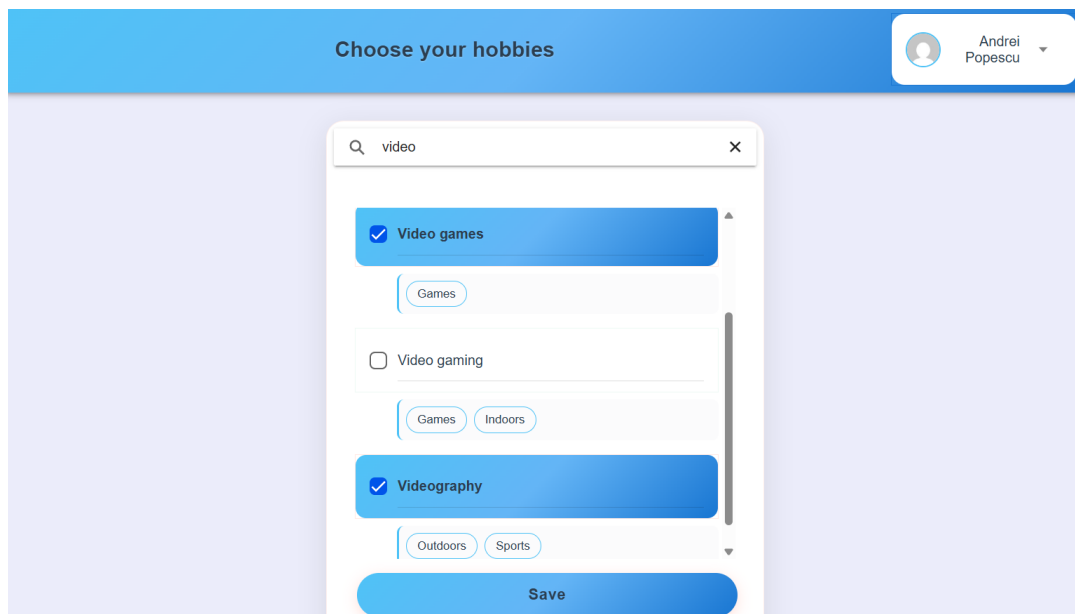


Figura 4.7: Pagina afișată utilizatorilor fără hobby-uri selectate

Dacă utilizatorul logat are hobby-uri selectate, se prezintă pagina de căutare a utilizatorilor cu hobby-uri asemănătoare, conform figurii 4.8. Pentru căutare, user-ul apasă butonul dedicat de pe ecran și se afișează lista utilizatorilor recomandați. Această listă este ordonată descrescător după scorul potrivirii.

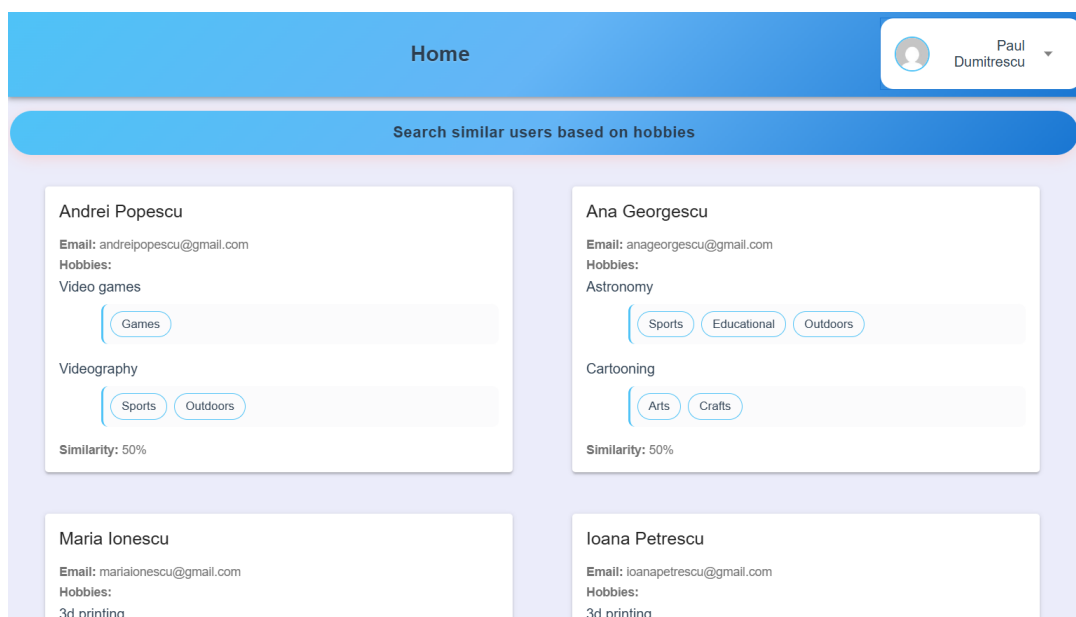


Figura 4.8: Pagina de căutare a utilizatorilor cu interese comune

De asemenea, utilizatorul se poate deconecta din aplicație. La apăsarea butonu-

lui ilustrat în figura 4.9, user-ul va fi redirectionat către pagina de autentificare.

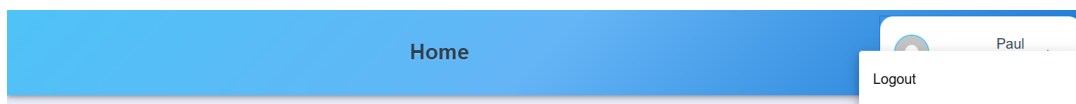


Figura 4.9: Deconectarea utilizatorului

## 4.4 Testare și validare

Pentru validarea funcționalităților aplicației dezvoltate, aceasta a fost testată pe platforma web, utilizând browsere precum Chrome și Brave. Deși framework-ul Ionic oferă posibilitatea de a rula aplicația și pe alte platforme (mobile și desktop), testarea s-a concentrat pe web. Procesul de testare a cuprins verificarea aplicației în întregime, cu asigurarea că toate scenariile de utilizare au fost luate în calcul.

### 4.4.1 Metodologia de testare

#### Testare manuală

Această abordare a implicat testarea sistematică a întregii aplicații prin interacțiunea directă cu interfața. Fiecare funcționalitate a fost testată individual pentru a asigura că este întocmai cu specificațiile: procesul de înregistrare și autentificare, adăugarea hobby-urilor, generarea recomandărilor și navigarea prin diferitele secțiuni ale aplicației. Au fost testate atât cazurile de utilizare standard cât și scenariile de eroare pentru a verifica comportamentul în situații limită.

#### Testare de interfață

Această metodă s-a concentrat pe evaluarea caracteristicilor vizuale și a experienței utilizatorului. Au fost verificate elementele de design din întreaga aplicație și modul în care interfața răspunde la schimbările de dimensiune ale ecranului. Suplimentar, s-a verificat funcționarea corectă a elementelor interactive și consistența tranzițiilor între pagini.

### 4.4.2 Scenarii de testare

Pentru validarea aplicației prin testare manuală, au fost create și executate mai multe scenarii de testare care acoperă toate funcționalitățile sistemului. Scenariile au fost împărțite în două categorii: scenarii pozitive, care testează comportamentul normal al aplicației, și scenarii negative, care verifică reacția programului la situații

excepționale. Fiecare dintre aceste scenarii a fost executat prin navigarea manuală în aplicație. Printre scenariile testate se numără: înregistrarea unui utilizator nou prin completarea formularului cu date valide și invalide, autentificarea unui utilizator cu credențiale corecte și incorecte, adăugarea hobby-urilor și generarea recomandărilor pentru diferite profiluri de utilizator. Această modalitate de testare a avut scopul de a identifica principalele erori ale funcționalităților și problemele legate de utilizabilitate și experiența utilizatorului.

#### **4.4.3 Rezultate**

Procesul de testare manuală a surprins funcționarea corectă a majorității funcționalităților. Aplicația gestionează corect atât cazurile de succes cât și cele de eroare. Datorită mesajelor clare de eroare pe care sistemul le afișează utilizatorilor, aceștia pot remedia ușor situațiile în care apar probleme. În linii mari, testarea a confirmat faptul că aplicația răspunde cerințelor funcționale și nefuncționale stabilite anterior.

#### **4.4.4 Limitări și îmbunătățiri**

Deși procesul de verificare a validat sistemul proiectat, au fost identificate limitări ale acestor metode. Multitudinea de cazuri de testare și combinațiile scenariilor fac ca această practică să fie dificilă și susceptibilă la erori umane. În plus, pe măsură ce aplicația crește, testarea manuală devine consumatoare de timp și greu de gestionat. Ca modalități de îmbunătățire, se recomandă implementarea unui sistem de testare automată și includerea celorlalte platforme în scenariile de testare.

# Capitolul 5

## Concluzii

Această lucrare a avut ca scop dezvoltarea unei aplicații software care favorizează interacțiunea dintre utilizatorii cu interese comune, prin folosirea unei metode de asociere a persoanelor bazată pe similaritatea hobby-urilor. Am analizat diversi algoritmi utilizați în sistemele de recomandare existente, atât în domeniul comerțului online, cât și al rețelelor sociale, pentru a aprofunda modul în care pot fi aplicate cu scopul conectării între persoane.

Aplicația realizată permite utilizatorilor să își aleagă hobby-urile și să descopere alți utilizatori cu interese similare. Asocierea se bazează pe identificarea unui scor de similaritate, calculat în funcție de hobby-urile comune. Pentru a îmbunătăți relevanța rezultatelor, algoritmul poate fi optimizat în mai multe direcții: integrarea unor date demografice, precum vârsta, genul și locația utilizatorilor, ponderarea hobby-urilor în funcție de frecvența lor în rândul persoanelor — hobby-urile comune ca muzica sau desenul pot avea un impact redus asupra calculului similarității, în timp ce cele rare sau specifice pot fi considerate mai semnificative în definirea profilului unui utilizator, dar și luarea în calcul a categoriilor de hobby-uri — pot exista compatibilități chiar dacă două persoane nu împărtășesc exact aceleași hobby-uri, dar au interese legate de activități de același tip.

Privind aplicația, există câteva direcții de dezvoltare care includ îmbunătățirea interfeței grafice pentru oferirea unei experiențe intuitive utilizatorilor, precum și adăugarea unei funcționalități de comunicare, care să permită utilizatorilor cu interese similare să inițieze conversații.

Așadar, lucrarea evidențiază potențialul unui sistem de recomandare axat pe interese comune pentru facilitarea interacțiunilor sociale. Optimizările propuse la nivel de algoritm și îmbunătățirile suplimentare aduse aplicației pot transforma aplicația într-o platformă eficientă de conectare între persoane cu profiluri compatibile.

# Bibliografie

- [1] M. G. Hunt, R. Marx, C. Lipson, and J. Young, "No more fomo: Limiting social media decreases loneliness and depression," *Journal of Social and Clinical Psychology*, vol. 37, no. 10, pp. 751–768, 2018.
- [2] Z. Xiao, J. Li, and G. Zhou, "Do common interests of students play a role in friendship?" *Procedia computer science*, vol. 131, pp. 733–738, 2018.
- [3] M. Z. Ahmed, A. Singh, A. Paul, S. Ghosh, and A. K. Chaudhuri, "Amazon product recommendation system," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 11, no. 3, 2022.
- [4] B. Smith and G. Linden, "Two decades of recommender systems at amazon.com," *Ieee internet computing*, vol. 21, no. 3, pp. 12–18, 2017.
- [5] M. Chiny, M. Chihab, O. Bencharef, and Y. Chihab, "Netflix recommendation system based on tf-idf and cosine similarity algorithms," *no. Bml*, pp. 15–20, 2022.
- [6] C. A. Gomez-Urbe and N. Hunt, "The netflix recommender system: Algorithms, business value, and innovation," *ACM Transactions on Management Information Systems (TMIS)*, vol. 6, no. 4, pp. 1–19, 2015.
- [7] E.-A. Baatarjav, S. Phithakkitnukoon, and R. Dantu, "Group recommendation system for facebook," in *On the Move to Meaningful Internet Systems: OTM 2008 Workshops: OTM Confederated International Workshops and Posters, ADI, AWeSoMe, COMBEK, EI2N, IWSSA, MONET, OnToContent+ QSI, ORM, PerSys, RDDDS, SEMELS, and SWWS 2008, Monterrey, Mexico, November 9-14, 2008. Proceedings.* Springer, 2008, pp. 211–219.
- [8] I. Heimbach, J. Gottschlich, and O. Hinz, "The value of user's facebook profile data for product recommendation generation," *Electronic Markets*, vol. 25, no. 2, pp. 125–138, 2015.
- [9] T. T. Nguyen, P.-M. Hui, F. M. Harper, L. Terveen, and J. A. Konstan, "Exploring the filter bubble: the effect of using recommender systems on content diver-

- sity," in *Proceedings of the 23rd international conference on World wide web*, 2014, pp. 677–686.
- [10] G. Tsakalakis and P. Koutsakis, "Improved user similarity computation for finding friends in your location," *Human-centric Computing and Information Sciences*, vol. 8, pp. 1–17, 2018.
- [11] 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.
- [12] 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>
- [13] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, "Collaborative filtering recommender systems," in *The adaptive web: methods and strategies of web personalization*. Springer, 2007, pp. 291–324.
- [14] M. S. D. Sondur, M. A. P. Chigadani, and S. Nayak, "Similarity measures for recommender systems: a comparative study," *Journal for Research*, vol. 2, no. 3, 2016.
- [15] S. Bag, S. K. Kumar, and M. K. Tiwari, "An efficient recommendation generation using relevant jaccard similarity," *Information Sciences*, vol. 483, pp. 53–64, 2019.
- [16] L. Al Hassanieh, C. Abou Jaoudeh, J. B. Abdo, and J. Demerjian, "Similarity measures for collaborative filtering recommender systems," in *2018 IEEE Middle East and North Africa Communications Conference (MENACOMM)*. IEEE, 2018, pp. 1–5.
- [17] B. Lika, K. Kolomvatsos, and S. Hadjiefthymiades, "Facing the cold start problem in recommender systems," *Expert systems with applications*, vol. 41, no. 4, pp. 2065–2073, 2014.
- [18] F. Mansur, V. Patel, and M. Patel, "A review on recommender systems," in *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*. IEEE, 2017, pp. 1–6.
- [19] P. Lops, M. De Gemmis, and G. Semeraro, "Content-based recommender systems: State of the art and trends," *Recommender systems handbook*, pp. 73–105, 2011.

- [20] W. Huang, B. Liu, and H. Tang, "Privacy protection for recommendation system: a survey," in *Journal of Physics: Conference Series*, vol. 1325, no. 1. IOP Publishing, 2019, p. 012087.
- [21] S. Badsha, X. Yi, and I. Khalil, "A practical privacy-preserving recommender system," *Data Science and Engineering*, vol. 1, pp. 161–177, 2016.
- [22] D. Co., "Ionic React documentation," <https://ionicframework.com/docs/react>, 2024, accesat la: 2025-02-21.
- [23] Meta Platforms, Inc. and affiliates, "React documentation," <https://react.dev/reference/react>, 2022, accesat la: 2025-02-21.
- [24] D. Co., "Ionic documentation," <https://ionicframework.com/docs>, 2023, accesat la: 2025-02-21.
- [25] Broadcom, "Spring Boot documentation," <https://docs.spring.io/spring-boot/>, 2024, accesat la: 2025-02-17.
- [26] The PostgreSQL Global Development Group, "PostgreSQL documentation," <https://www.postgresql.org/docs/>, 2024, accesat la: 2025-02-19.
- [27] Commonhaus Foundation, "Hibernate documentation," <https://hibernate.org/orm/documentation/6.5/>, 2024, accesat la: 2025-02-19.
- [28] A. Raj, "A Comprehensive Collection of Hobbies," <https://www.kaggle.com/datasets/mrhell/list-of-hobbies>, 2022, accesat la: 2025-01-25.
- [29] Dawid9632, "Hobbies & interests with categories," <https://www.kaggle.com/datasets/dawid9632/hobbies-interests-with-categories>, 2022, accesat la: 2025-01-25.
- [30] A. Ugurcan, "Hobbies Dataset," <https://huggingface.co/datasets/alperugurcan/Hobbies>, 2024, accesat la: 2025-01-25.
- [31] Pallets, "Flask documentation," <https://flask.palletsprojects.com/en/stable/>, 2024, accesat la: 2025-03-03.