

UNIVERSITATEA "ALEXANDRU-IOAN CUZA" DIN IASI

## FACULTATEA DE INFORMATICA



LUCRARE DE LICENTA

### Uixplorer - Aplicatie Web dedicata invatarii conceptelor UI/UX

propusa de

**Smântână Lavinia**

**Sesiunea:** iulie, 2024

Coordonator stiintific

**Conf. Dr. Vitcu Anca**

UNIVERSITATEA "ALEXANDRU-IOAN CUZA" DIN IASI

## FACULTATEA DE INFORMATICA

# Uixer - Aplicație Web dedicată învățării conceptelor UI/UX

Smântână Lavinia

Sesiunea: iulie, 2024

Coordonator științific

Conf. Dr. Vitcu Anca

Avizat,

Îndrumător lucrare de licență,

Conf. Dr. Vitcu Anca.

Data: .....

Semnătura: .....

## Declarație privind originalitatea conținutului lucrării de licență

Subsemnatul **Smântână Lavinia** domiciliat în **România, jud. Iași, mun. Iași, Strada Valea Adâncă nr. 76E**, născut la data de **14 noiembrie 2001**, identificat prin CNP **6011114226777**, absolvent al Facultății de informatică, **Facultatea de informatică** specializarea **informatică**, promoția 2024, declar pe propria răspundere cunoscând consecințele falsului în declarații în sensul art. 326 din Noul Cod Penal și dispozițiile Legii Educației Naționale nr. 1/2011 art. 143 al. 4 și 5 referitoare la plagiat, că lucrarea de licență cu titlul **Uixplorer - Aplicație Web dedicată învățării conceptelor UI/UX** elaborată sub îndrumarea domnului **Conf. Dr. Vitcu Anca**, pe care urmează să o susțin în fața comisiei este originală, îmi aparține și îmi asum conținutul său în întregime.

De asemenea, declar că sunt de acord ca lucrarea mea de licență să fie verificată prin orice modalitate legală pentru confirmarea originalității, consumând inclusiv la introducerea conținutului ei într-o bază de date în acest scop.

Am luat la cunoștință despre faptul că este interzisă comercializarea de lucrări științifice în vederea facilitării falsificării de către cumpărător a calității de autor al unei lucrări de licență, de diplomă sau de disertație și în acest sens, declar pe proprie răspundere că lucrarea de față nu a fost copiată ci reprezintă rodul cercetării pe care am întreprins-o.

Data: .....

Semnătura: .....

## **Declarație de consimțământ**

Prin prezenta declar că sunt de acord ca lucrarea de licență cu titlul **Uixplorer - Aplicație Web dedicată învățării conceptelor UI/UX**, codul sursă al programelor și celelalte conținuturi (grafice, multimedia, date de test, etc.) care însotesc această lucrare să fie utilizate în cadrul Facultății de informatică.

De asemenea, sunt de acord ca Facultatea de informatică de la Universitatea "Alexandru-Ioan Cuza" din Iași, să utilizeze, modifice, reproducă și să distribuie în scopuri necomerciale programele-calculator, format executabil și sursă, realizate de mine în cadrul prezentei lucrări de licență.

Absolvent **Lavinia Smântână**

Data: .....

Semnătura: .....

# Cuprins

<b>Motivație</b>	<b>2</b>
<b>Introducere</b>	<b>4</b>
<b>1 Analiza aplicației</b>	<b>6</b>
1.1 Aplicații similare existente . . . . .	6
1.1.1 Laws of UX . . . . .	6
1.1.2 Learn UX . . . . .	7
1.1.3 Uxcel . . . . .	8
1.2 Stabilirea specificațiilor . . . . .	9
<b>2 Arhitectura aplicației</b>	<b>11</b>
2.1 Arhitectura Model-View-Controller . . . . .	12
2.2 Scenariu de utilizare . . . . .	15
<b>3 Implementarea aplicației</b>	<b>17</b>
3.1 Backend . . . . .	17
3.1.1 Tehnologii folosite . . . . .	17
3.1.2 Implementarea Backend-ului . . . . .	20
3.1.3 Sistemul de recomandare de articole . . . . .	24
3.1.4 Baza de date . . . . .	26
3.2 Frontend . . . . .	28
3.2.1 Tehnologii utilizate . . . . .	28
3.2.2 Interfața aplicației . . . . .	29
<b>Concluzii</b>	<b>37</b>
<b>Bibliografie</b>	<b>38</b>

# Motivație

În contextul în care aplicațiile mobile, website-urile și alte platforme digitale devin din ce în ce mai dezvoltate din punct de vedere tehnologic, designerii trebuie să fie mereu la curent cu noile tendințe, metode și tehnologii pentru a crea produse care nu sunt doar funcționale, ci și atrăgătoare și intuitive.

Studentii și profesioniștii din domeniul designului se confruntă cu provocarea de a rămâne relevanti într-un peisaj tehnologic în continuă schimbare. Accesul la informații actualizate este esențial pentru a menține un nivel ridicat de competență și pentru a se adapta rapid la noile cerințe ale pieței. În plus, colaborarea și schimbul de idei între designeri pot duce la soluții mai creative și mai eficiente, contribuind la dezvoltarea unui portofoliu de succes.

De asemenea, industria designului UI/UX pune un accent considerabil pe înțelegerea comportamentului utilizatorului și pe crearea de interfețe care să îmbunătățească experiența acestuia. Acest lucru necesită nu doar o bază teoretică solidă, dar și o învățare continuă.

Aplicația Uixplorer vine ca un răspuns la necesitatea tot mai mare de resurse educaționale accesibile și de calitate pentru studentii și profesioniștii din acest domeniu, oferindu-le o platformă integrată de învățare, colaborare și actualizare constantă a cunoștințelor.

Prin dezvoltarea acestei aplicații, mi-am propus să ating următoarele obiective:

- Oferirea accesului la resurse educaționale actualizate și relevante în UI/UX design: Printr-un catalog diversificat și actualizat de articole, Uixplorer își propune să devină o sursă esențială pentru studenți și profesioniști, asigurându-le informațiile necesare pentru a rămâne la curent cu evoluțiile din domeniu și pentru a își îmbunătăți competențele.
- Construirea unei comunități active și colaborative: Forumul integrat în aplicație

are ca obiectiv principal crearea unui spațiu de discuții, unde utilizatorii pot învăța unii de la alții, pot împărtăși idei și pot primi feedback constructiv. De asemenea, aceștia au opțiunea de a adăuga într-o listă de prieteni alți utilizatori și de a avea discuții private cu ei.

- Personalizarea experienței de învățare: Sistemul de recomandare al aplicației urmărește să ofere utilizatorilor resurse adaptate intereselor și nevoilor lor individuale, contribuind astfel la o învățare mai eficientă și mai plăcută.

Prin atingerea acestor obiective, Uixerplorer se angajează să devină un instrument indispensabil pentru toți cei interesati de UI/UX design, sprijinindu-i în dezvoltarea și perfectionarea abilităților lor.

# Introducere

Tema principală a aplicației mele reprezintă crearea unui mediu educațional pentru persoanele interesate de UI/UX design. Fiind atrasă de Human-Computer Interaction în timpul facultății, am ales această temă din dorința personală de a avea o platformă dedicată învățării conceptelor teoretice de design.

Prin intermediul acestui site, doresc să ofer utilizatorilor o platformă unde pot găsi articole actuale și ușor de citit, care descriu problemele curente din domeniul design-ului și oferă soluții și best-practices pentru ele. De asemenea, am considerat că integrarea unui forum dedicat comunicării dintre utilizatori va fi un mare plus, deoarece va permite schimbul de idei, soluționarea problemelor și colaborarea pe proiecte comune.

În dezvoltarea acestei aplicații am folosit o serie de tehnologii și instrumente moderne pentru a asigura o experiență de utilizare optimă și o performanță ridicată. Pentru mediul de dezvoltare, am ales IntelliJ IDEA, un IDE robust și flexibil, care oferă suport avansat pentru multiple limbi de programare și framework-uri.

Pentru partea de backend, am utilizat Spring Boot, un framework popular pentru dezvoltarea aplicațiilor Java, care simplifică configurarea și implementarea serviciilor web. Am ales Spring Boot datorită capacitatea sa de a accelera procesul de dezvoltare prin configurarea automată și prin suportul excelent pentru crearea de aplicații scalabile și robuste.

Pentru partea de frontend, am integrat Thymeleaf, un motor de template-uri care permite crearea de pagini web dinamice într-un mod intuitiv și eficient. Acest lucru mi-a permis să îmbin datele și funcționalitățile aplicației cu prezentarea, oferind utilizatorilor o interfață atractivă și ușor de utilizat.

Limbajul de programare principal utilizat a fost Java, datorită fiabilității și versatilității sale. Pentru designul și structura paginilor web, am folosit HTML și CSS, asigurând astfel o prezentare vizuală clară a conținutului. Pentru a adăuga interacți-

vitate și a îmbunătăți experiența utilizatorilor, am integrat anumite funcționalități cu JavaScript.

În următoarele capitole, voi descrie structura și funcționalitățile aplicației Uixer.

Primul capitol începe cu dezvoltarea abordării temei propuse, unde voi prezenta anumite aplicații similare deja existente pe piață și voi explica cum am stabilit specificațiiile necesare site-ului meu.

Capitolul doi va cuprinde câteva detalii generale despre arhitectura unei aplicații și o prezentare a modelului folosit în proiect (Model-View-Controller), demonstrat cu un exemplu concret.

În final, capitolul trei va aborda detaliile de implementare, unde voi dezvolta mai multe despre tehnologiile utilizate în Backend-ul și Frontend-ul aplicației.

# Capitolul 1

## Analiza aplicației

Pentru a dezvolta cu succes o aplicație web care să răspundă la nevoile utilizatorilor, este nevoie de a analiza cu atenție care sunt cele mai importante funcționalități pe care vrei să le integrezi, în conformitate cu publicul țintă și nevoile acestuia. Așadar, o practică bună este cercetarea pieței și analizarea aplicațiilor similare deja existente. În timpul acestei etape, se realizează atât o analiză a concurenței, cât și evaluarea și identificarea obiectivelor proiectului.

### 1.1 Aplicații similare existente

#### 1.1.1 Laws of UX

Laws of UX este un site dedicat promovării de best-practices pe care designerii le pot lua în considerare atunci când construiesc interfețe pentru utilizatori.

Principalele funcționalități pe care acest site le include sunt:

- Un set de legi ale UX design-ului (precum Fitts' Law, Hick's Law, Law of Proximity etc.), prezentate printr-o imagine sugestivă și o scurtă descriere.
- Un catalog de articole care combină domeniul psihologiei și User Experience.
- Prezentarea cărții *Laws of Ux*, scrisă de Jon Yablonski (dezvoltatorul aplicației). În această pagină putem regăsi o prezentare generală a cărții, un rezumat al punctelor pe care aceasta le abordează și câteva review-uri oferite de către cititorii cărții.
- Secțiunea de magazin: Dezvoltatorul aplicației pune la dispoziție utilizatorilor opțiunea de a cumpăra un pachet de cărți cu 54 de principii psihologice și metode

UX, cât și anumite postere care reprezintă legile din UX design.

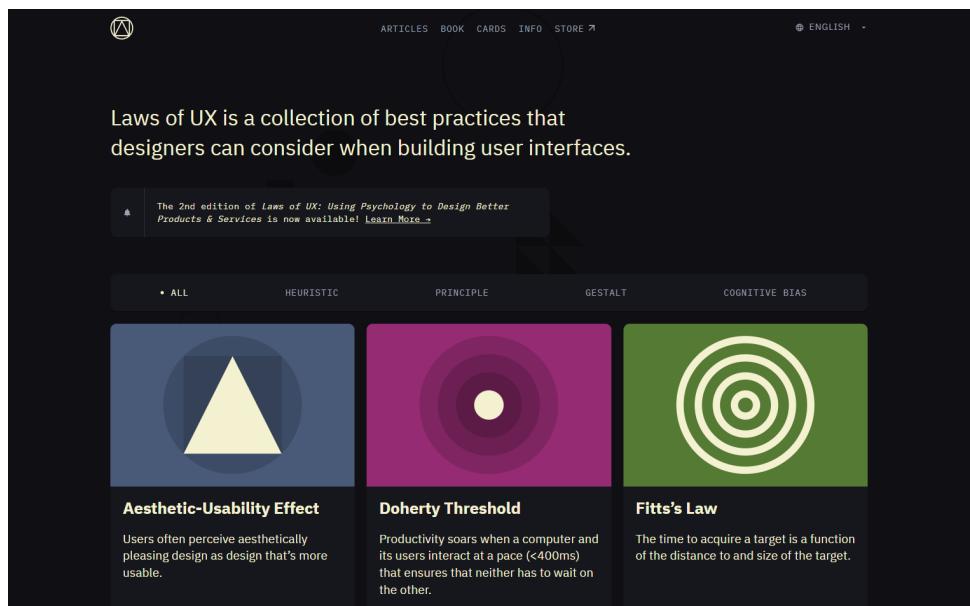


Figura 1.1: Laws of UX

### 1.1.2 Learn UX

Learn UX, site web dezvoltat de Greg Rog, oferă utilizatorilor săi o varietate de cursuri video, accesibile contra cost printr-o subscripție lunară. Aplicația este proiectată atât pentru începători, cât și pentru cei avansați care doresc să învețe anumite caracteristici mai puțin cunoscute.

Câteva dintre cursurile disponibile sunt:

- The basics: Design for Screens, Usability, User Testing.
- Design Workflow: Sketch App, Adobe XD, Figma.
- Design & Prototype: InVisio Studio, Principle App, Flinto.
- Handoff & Tools: Zeplin, Abstract.
- HTML and CSS Workshop.
- Webflow: Webflow Basics & Advanced.

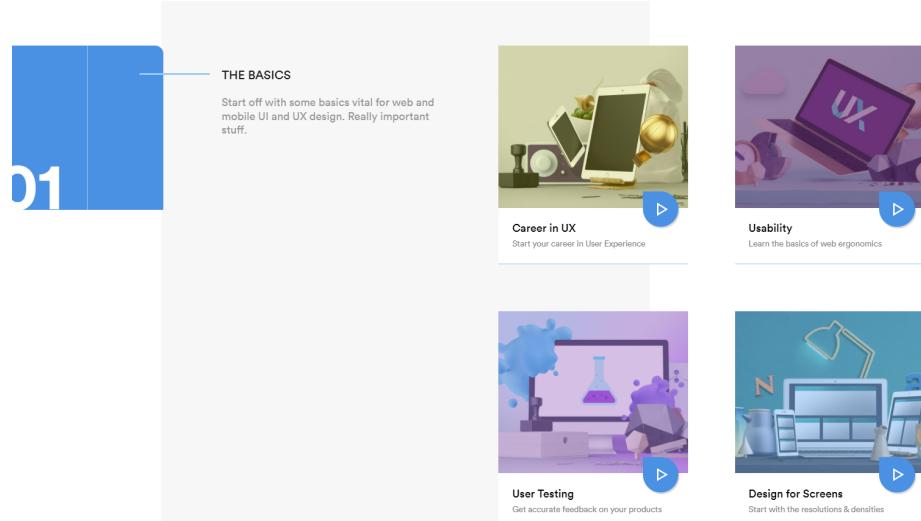


Figura 1.2: Learn UX

### 1.1.3 Uxcel

Uxcel este o platformă de învățare profesională care ajută persoanele să își dezvolte abilitățile de design UI prin cursuri interactive, lecții, evaluări și proiecte de design. A fost recunoscută ca fiind "Duolingo pentru învățarea designului UX" și este populară atât în rândul designerilor profesioniști, cât și al studentilor.

**Explore**  
Discover the best UX and UI learning resources online.

**Courses**

- UX Design Foundations** POPULAR  
Master the fundamentals of UX design to build a solid foundation in design principle... Beginner • 6 hours
- Introduction to Figma**  
Master Figma's core functions from the ground up. Explore the basics and beyond ... Beginner • 5 hours
- Design Terminology**  
This course is a perfect introduction to design terminology for all professional... Beginner • 3 hours

**Design Briefs**

- UX/UI Case Study for Inclusive Land...** LIVE  
Test your skills by creating a case study for a landing page that adheres to diversity an... \$2,500+ in prizes Ends Jun 30
- Promotional Media Design for Uxcel ...**  
Put your visual design skills into practice by designing a set of promotional materials fo... Content Strategy • Intermediate
- Analytical Dashboard for Work Man...**  
Demonstrate your interaction design and content strategy skills by designing an... Interaction Design • Advanced

Figura 1.3: Uxcel

O statistică de pe site menționează faptul că 84% din utilizatorii Uxcel își îmbună-

tătesc abilitățile lor de design în doar 3 luni de la folosirea aplicației.

Uxcel oferă atât o variantă gratuită ce dispune de multe cursuri, cât și varianta Pro, contra cost, ce include toate cursurile disponibile pe site, certificări, evaluări și multe altele.

## 1.2 Stabilirea specificațiilor

Unul dintre primii pași în dezvoltarea unei noi aplicații este definirea concretă a specificațiilor dorite. Consider că un rol important în realizarea unei aplicații orientate spre un public țintă specific este crearea de Personas.

Personas sunt personaje fictive, create pe baza cercetărilor efectuate pentru a reprezenta diferitele tipuri de utilizatori care ar putea utiliza serviciul/site-ul într-un mod similar. Crearea acestora va ajuta la înțelegerea nevoilor, experiențelor, comportamentelor și obiectivelor utilizatorilor.

**Andrei Popescu**  
35 / M

Andrei lucrează în domeniul designului UI/UX de cinci ani. A absolvit facultatea de Design Grafic și a făcut tranzitia către UI/UX prin diverse cursuri online și bootcampuri. Este pasionat de crearea interfețelor intuitive și îmbunătățirea experiențelor utilizatorilor.

**Detalii**

Vârstă:	28 ani
Nume:	Andrei Popescu
Ocupație:	UI/UX Designer la o companie de tehnologie de dimensiuni medii

**Obiective**

- Să fie la curent cu ultimele tendințe și bune practici în designul UI/UX.
- Să își îmbunătățească abilitățile de design prin învățare continuă.
- Să primească feedback și inspirație pentru proiectele sale curente.

**Frustrări**

- Găsirea de conținut de calitate și relevant printre informațiile abundente online.
- Lipsa unei platforme centralizate pentru a discuta idei și a primi feedback profesional.
- Limitări de timp din cauza programului său de lucru aglomerat.

**Caracteristici**

Andrei este o persoană curioasă și creativă, mereu în căutarea de noi provocări și experiențe. În timpul liber, îi place să fotografieze peisaje urbane și naturale, combinându-și pasiunea pentru artă vizuală cu plăcerea de a călători. Andrei este și un mare fan al literaturii science fiction, găsindu-și inspirația în cărțile autorilor clasici ai genului.

**Tehnologii**

- Dispozitive: Laptop cu Windows, telefon Android
- Platforme: Canva, Google Analytics, Trello, Slack

Figura 1.4: Exemplu de Persona pentru aplicația Uixplorer

Spre deosebire de proiectarea produselor, serviciilor și soluțiilor pe baza preferințelor echipei de proiectare, a devenit o practică să se adune cercetări și să se personifice tendințe și modele specifice în date sub formă de Personas.

Personas nu descriu persoane reale, ci sunt compuse pe baza datelor reale colectate de la mai multe persoane. Personas adaugă o notă umană la cercetarea și stabilirea unor specificații pentru o aplicație. Crearea de profiluri Persona ale utilizatorilor tipici

sau atipici (extremi) ajută la înțelegerea tiparele din partea de cercetare și analizare, care sintetizează tipurile de persoane pentru care se proiectează aplicația.

După ce am analizat nevoile utilizatorilor, am înțeles că o mare problemă al unui designer (atât începător, cât și avansat) este lipsa feedback-ului și a comunicării între designeri. Soluția mea pentru această problema este integrarea unui forum care stă la dispozitie pentru orice utilizator și le oferă un loc unde își pot expune proiectele personale sau unde pot primi răspunsuri la întrebările lor. De asemenea, Uixplorer oferă posibilitatea de a adăuga în lista de prieteni utilizatori din toată lumea și de a putea avea discuții private cu ei.

O altă funcționalitate importantă în aplicația mea este sistemul de recomandare implementat în server. Pe bază ce un utilizator citește anumite articole, acesta va primi constant un număr de articole recomandate, pe care nu le-a descoperit încă. În cazul utilizatorilor care abia s-au înregistrat în aplicație, se va oferi o recomandare după top-ul articolelor citite de către ceilalți utilizatori. Acest sistem de recomandare ajută la explorarea rapidă a cât mai multor articole disponibile în catalog.

Pentru o experiență diferită în citirea articolelor, am folosit un API (SpeechSynthesis de la Web Speech API) pentru a facilita citirea articolelor de către un robot, prin apăsarea unui buton de play. Astfel, utilizatorii pot asculta conținutul în loc să-l citească, oferindu-le o alternativă practică și accesibilă, mai ales în situații când au altă fereastră deschisă sau pur și simplu preferă să consume informația audio.

Un alt lucru pe care mi-am dorit să îl adaug în aplicația mea este un Dashboard, care va conține progresul utilizatorului și îi va afișa articolele pe care acesta deja le-a parcurs. De asemenea, tot în Dashboard vor fi disponibile și Badge-urile pe care utilizatorul le are. Acestea au ca rol să încurajeze utilizatorii să navigheze pe site-ul Uixplorer cât mai des.

# Capitolul 2

## Arhitectura aplicației

Arhitectura aplicației reprezintă structura generală a unei aplicații software, descriind modul în care diferitele componente ale acesteia interacționează între ele și cu mediul exterior. Aceasta încorporează o gamă largă de principii de proiectare, de la modele la framework-uri, jucând un rol indispensabil în determinarea performanței, scalabilității și funcționalității generale a unei aplicații.

Pe scurt, arhitectura unei aplicații poate fi privită ca o hartă sau ca o schiță. Această schiță oferă o vedere de ansamblu asupra componentelor și interacțiunilor într-o aplicație.

Este esențială pentru a garanta că aplicația funcționează conform planului și îndeplinește cerințele funcționale și nefuncționale. Arhitectura aplicației reprezintă baza dezvoltării software, orientând dezvoltatorii în fiecare etapă a procesului.

Fiecare aplicație web este formată dintr-un frontend și un backend.

Frontend-ul, cunoscut și sub numele de client-side, reprezintă tot ceea ce vede și interacționează utilizatorul în interiorul browserului său. Acesta este scris în HTML, CSS și JavaScript. Backend-ul este cunoscut ca partea de server a aplicației. Nu este accesibilă utilizatorilor; aceasta stochează și manipulează datele, procesează cererile HTTP care, în esență, aduc datele (text, imagini, fișiere etc.) solicitate de utilizator. Spre deosebire de frontend, pentru a scrie backend-ul unei aplicații web pot fi folosite multe limbiage, cum ar fi Java, Python, PHP, JavaScript, C# etc.

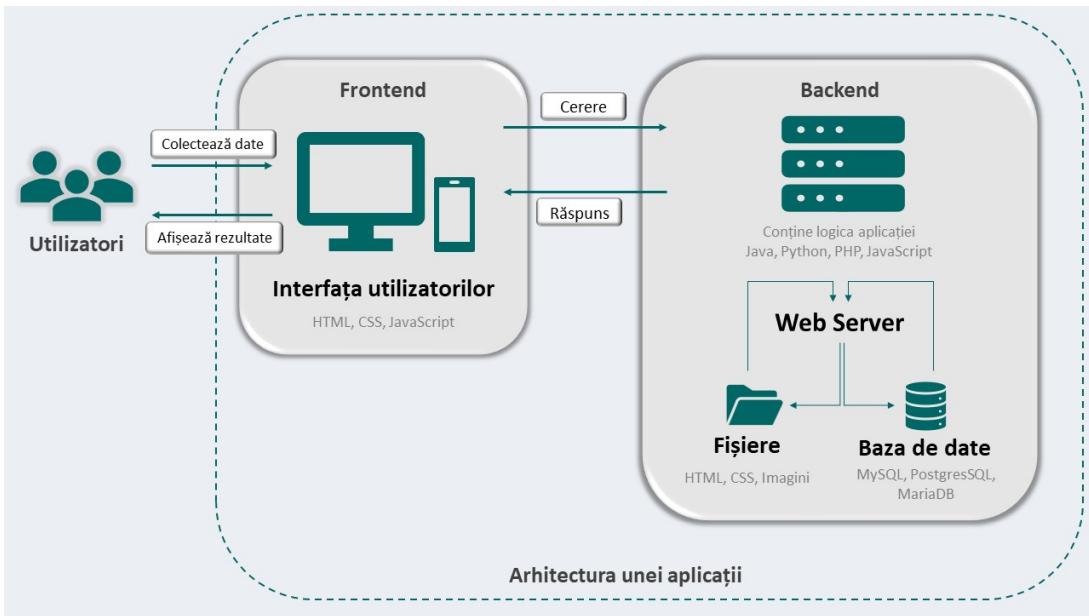


Figura 2.1: Arhitectura Aplicației

Principalele aspecte ale arhitecturii aplicației includ:

- Modularitatea: Împărțirea aplicației în module distincte, fiecare având o responsabilitate clară.
- Scalabilitatea: Capacitatea aplicației de a gestiona o creștere a numărului de utilizatori și a volumului de date.
- Performanța: Asigurarea unui timp de răspuns rapid și a unei experiențe fluide pentru utilizatori.
- Securitatea: Protejarea datelor utilizatorilor și prevenirea accesului neautorizat.
- Mantenabilitatea: Facilitarea actualizării și întreținerii aplicației prin cod clar și documentație adecvată.

## 2.1 Arhitectura Model-View-Controller

MVC (Model-View-Controller) este un model arhitectural care formează structura de bază a multor aplicații software. Acest model împarte o aplicație în trei componente distincte, dar interconectate.

Cele trei părți ale MVC sunt:

- Model: definește structura datelor și gestionează logica.

- View: se ocupă de interfața cu utilizatorul și de prezentarea datelor.
- Controller: intermediarul dintre model și view. Preia cererea și instruiește modelul să execute anumite acțiuni.

Tipul acesta de arhitectură separă logica aplicației de afișare. Pe lângă faptul că creează un cadru solid pentru aplicațiile web, asigură și organizarea clară a diferitelor aspecte ale aplicației, simplificând astfel scalabilitatea viitoare.

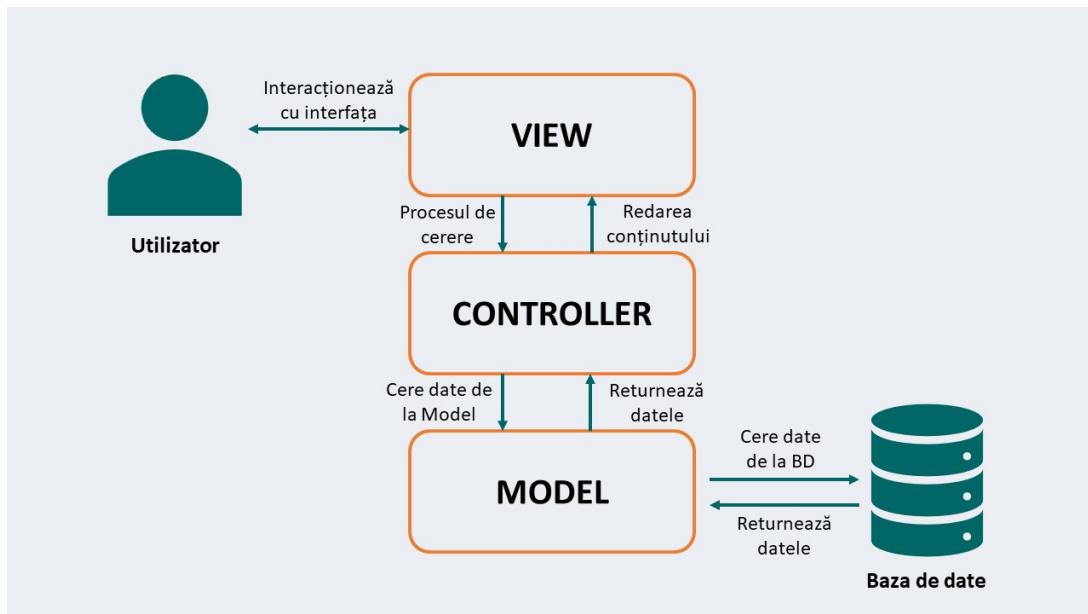


Figura 2.2: Modelul MVC

Un posibil flux de lucru într-o aplicație MVC poate fi reprezentat prin următorii pași:

- Utilizatorul face o cerere (request): În acest timp, utilizatorul interacționează cu interfața aplicației (View), fie prin trimiterea unui formular sau apăsarea unui anumit buton. Cererea HTTP va fi transmisă către server.
- Preluarea cererii de către Controller: Serverul/Backend-ul primește cererea de la client și o trimit la funcția corespunzătoare din Controller, care determină ce acțiune trebuie să facă.
- Interacțiunea dintre Controller și Model: Controller-ul este cel care solicită informații necesare de la Model, iar acesta gestionează logica. Dacă cererea necesită date din baza de date, Modelul va fi cel care va efectua interogările bazei de date și va returna rezultatele către Controller.

- Pregătirea răspunsului în View: Controller-ul preia toate datele primite de la Model și le transmite înapoi către componenta View, care va utiliza datele primite pentru a genera interfața cu utilizatorul.
- Răspunsul către client: View-ul final este trimis către utilizator sub forma unui răspuns HTTP, iar browser-ul este cel care va reda conținutul primit.

În aplicația web Uixplorer, componentele MVC sunt împărțite astfel: la nivelul backend-ului aplicației, sunt definite modelele în pachetul "models" (precum AppUser, Article, Comment etc.), iar funcționalitatea de controller este asigurată în pachetul "controllers". View-urile se regăsesc în frontend-ul aplicației (fișierele templates, static), fiind implementate folosind HTML, CSS și Javascript.

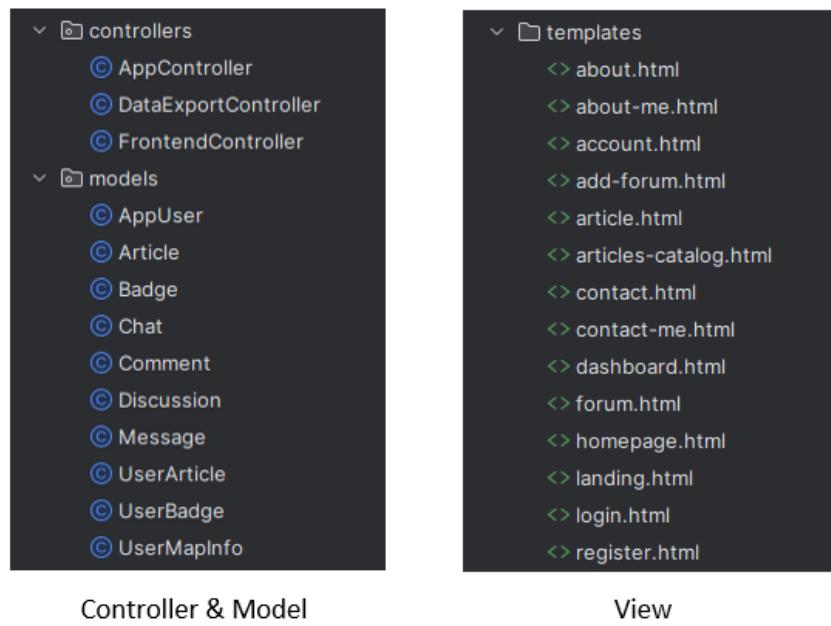


Figura 2.3: Modelul MVC în Uixplorer

Un exemplu concret, urmând cei 5 pași exemplificați mai sus, ar fi următorul:

- Un utilizator accesează pagina "/articles".
- Controller-ul pentru pagina de articole primește cererea de la utilizator.
- Modelul interoghează baza de date pentru a obține lista completă de articole.
- Datele rezultate din interogare sunt transmise către View, care generează o pagină HTML cu lista de articole.
- Pagina HTML este trimisă către browser-ul utilizatorului.

## 2.2 Scenariu de utilizare

În continuare, voi defini interacțiunea utilizatorului cu sistemul pentru aplicația Uixplorer. Am ilustrat aceste scenarii prin diagrama use-case în figura de mai jos:



Figura 2.4: Diagramă Use-Case

Un utilizator ce nu este autentificat are opțiunea de a vizualiza paginile about (unde sunt prezentate detalii despre site) și contact, de înregistrare în care își creează un cont nou și cea de autentificare în care introduce datele corespunzătoare pentru a intra în contul existent.

După autentificare, acesta va intra în Homepage, de unde va putea naviga pe paginile About, Contact, Articles, Forum, Account, Dashboard, folosind link-urile din header-ul paginii. Tot aici utilizatorul va primi de la sistem o recomandare personalizată de trei articole pe care nu le-a mai citit și o recomandare cu privire la adăugarea în lista de prieteni a utilizatorilor care locuiesc în aceeași zonă geografică. De asemenea, în header-ul paginii își va putea vizualiza lista actuală de prieteni și, printr-un click,

poate deschide conversațiile cu prietenii săi, unde poate vizualiza istoricul mesajelor sau poate trimite un mesaj nou.

În meniul de articole, utilizatorul are opțiunea de a căuta un articol după titlul său sau de a selecta un anumit articol, pe care îl poate citi sau poate folosi robotul integrat pentru a asculta conținutul articolului. După finalizare, poate apăsa pe un buton pentru a marca articolul ca fiind citit. În pagina Forum-ului, utilizatorul poate vizualiza discuțiile actuale dintre ceilalți utilizatori și poate contribui fie printr-un comentariu la o discuție existentă, fie prin începerea unei noi discuții.

Utilizatorul va mai avea la dispoziție o secțiune de administrare a contului, unde acesta își va putea modifica datele contului (precum email, parolă, poza de profil). De asemenea, în Dashboard-ul utilizatorului, acesta va putea vizualiza articolele citite până la momentul curent și badge-urile câștigate prin citirea articolelor și folosirea aplicației web.

# Capitolul 3

## Implementarea aplicației

După finalizarea procesului de planificare și proiectare a aplicației, următorul pas este implementarea acestor decizii în codul sursă, transformând astfel ideile și concepțele în funcționalități reale.

În acest capitol voi scoate în evidență modul în care a fost dezvoltată această aplicație web și voi vorbi despre tehnologiile pe care am ales să le folosesc.

### 3.1 Backend

#### 3.1.1 Tehnologii folosite

##### Java

Java este un limbaj de programare orientat pe obiecte, robust, sigur și independent de platformă. A fost lansat în 1995 de către compania Sun Microsystems și la momentul actual este una dintre cele mai folosite tehnologii pe baza căruia sunt construite aplicații pentru toate segmentele industriei software.

Am ales Java pentru backend-ul proiectului meu datorită avantajelor sale: portabilitate (codul scris în Java poate fi rulat pe orice sistem cu JVM), performanță (optimizează codul la timpul de rulare), securitate (oferă conceptul de gestionare a excepțiilor și are caracterisici de securitate integrate, precum gestionarea memoriei și verificarea riguroasă a tipurilor), orientat pe obiecte (se bazează pe conceputul de "obiecte", care sunt instanțe ale claselor și pot conține atât date, cât și metode pentru a manipula acele date, ce facilitează reutilizarea codului) și pentru framework-urile sale, precum Spring Security care oferă soluții integrate pentru autentificarea și autorizarea utilizatorilor.

Platforma standard de lucru Java este Java SE (Standard Edition) și reprezintă versiunea de bază a Java. Aceasta furnizează o colecție largă de API-uri și biblioteci care oferă funcționalități de bază pentru dezvoltarea unei aplicații folosind Java, precum gestionarea colecțiilor cu pachetul "java.util" sau rezolvarea operațiilor matematice cu numere mari cu "java.math".

## **Spring Boot**

Java Spring Framework este un framework popular, open-source, la nivel de întreprindere, pentru crearea de aplicații independente, de producție, care rulează pe mașina virtuală Java (JVM). Acest framework este compatibil cu Java SE și se folosește de funcționalitățile de care platforma standard dispune și le extinde prin adăugarea unor caracteristici avansate.

Java Spring Boot este o extensie care face ca dezvoltarea aplicațiilor web să fie mai rapidă și mai ușoară prin intermediul a trei capacitați de bază: autoconfigurare, o abordare avizată a configurării și posibilitatea de a crea aplicații de sine stătătoare.

Spring Boot este ideal pentru crearea de arhitecturi de microservicii, deoarece oferă suport pentru dezvoltarea serviciilor RESTful și integrarea ușoară cu alte servicii. De asemenea, acesta simplifică gestionarea dependențelor, asigurând compatibilitate și versiuni stabile ale bibliotecilor utilizate.

Spring Security este cel care se ocupa de securizarea aplicațiilor Java care folosesc Framework-ul Spring. Acesta oferă un set cuprinzător de funcționalități de securitate, precum autentificare și autorizare sau configurarea accesului la resurse pentru diferite tipuri de utilizatori. Astfel, Spring Security permite dezvoltatorilor să implementeze măsurile de securitate dorite cu un efort minim prin folosirea unor concepte, precum SecurityFilterChain și AuthenticationManager.

Prin urmare, utilizarea limbajului Java, împreună cu framework-ul Spring, mi-a permis să construiesc o aplicație eficientă, securizată și ușor de întreținut.

## **IntelliJ IDEA**

IntelliJ IDEA este un IDE puternic, dezvoltat de JetBrains, cunoscut pentru eficiență și productivitatea sa. Un mare avantaj constă în faptul că IDE-ul oferă completare automată intelligentă, navigare rapidă în cod și refactorizare automată, ajutând

astfel dezvoltatorii să economiească timp și să le reducă posibilele erori. De asemenea, oferă suport pentru o varietate de plugin-uri care se integrează cu ale instrumente și servicii folosite în dezvoltare și extind funcționalitățile IDE-ului.

Pe lângă Java, IntelliJ IDEA oferă suport pentru o varietate de limbaje de programare, precum Kotlin, Scala, HTML, CSS, Javascript și multe altele.

## Python

Python este un limbaj de programare versatil, cunoscut pentru simplitatea și lisibilitatea sa. Câteva dintre avantajele sale care m-au făcut să aleg Python pentru a implementa anumite funcționalități în proiectul meu sunt: scriptarea și automatizarea (Python este ideal pentru a scrie scripturi de automatizare, reducând efortul manual și minimizând erorile; gama largă de biblioteci puternice pentru analiză de date și machine learning (Pandas, NumPy, scikit-learn/sklearn).

Am folosit Python în aplicația mea pentru a construi sistemul de recomandare datorită ecosistemului său bogat de biblioteci și framework-uri dedicate machine learning-ului. În particular, am utilizat **scikit-learn** (sau sklearn), o bibliotecă pentru învățarea automată și analiza de date. Am folosit 'TfidfVectorizer' pentru vectorizarea textului și funcția 'cosine\_similarity' pentru calculul similarității între articole, oferind astfel o recomandare personalizată utilizatorilor.

De asemenea, am utilizat **pandas** pentru manipularea datelor, prin crearea unor DataFrame-uri (structuri de date similare cu tabelele din bazele de date) ce reprezintă datele articolelor. Cu **flask**, am creat o aplicație web la portul 5000, care apoi este apelată de Controller-ul din Java. O altă bibliotecă pe care am folosit-o în sistemul de recomandare este **requests**, care m-a ajutat să iau datele și informațiile despre articolele citite de la un server extern (din clasa DataExportController a aplicației Java).

```
String url = "http://localhost:5000/recommend";
HttpHeaders headers = new HttpHeaders();
headers.setContentType(MediaType.APPLICATION_JSON);

Map<String, Long> requestBody = new HashMap<>();
requestBody.put("user_id", userId);

HttpEntity<Map<String, Long>> entity = new HttpEntity<>(requestBody, headers);
ResponseEntity<Article[]> response = restTemplate.postForEntity(url, entity, Article[].class);

List<Article> recommendedArticles = Arrays.asList(Objects.requireNonNull(response.getBody()));

model.addAttribute("articlesRec", recommendedArticles);
```

Figura 3.1: Folosirea sistemului de recomandare în Java

### 3.1.2 Implementarea Backend-ului

Capitolul trecut am vorbit despre tipul de arhitectură folosit în aplicatiea mea, Model-View-Controller. În acest caz, fiecare componentă va îndeplini un rol specific și va interacționa cu celelalte într-un mod bine definit. În continuare voi prezenta în detaliu clasele și funcționalitățile pe care le-am implementat în partea de server/backend. În figura de mai jos poate fi observată structura proiectului din IntelliJ:

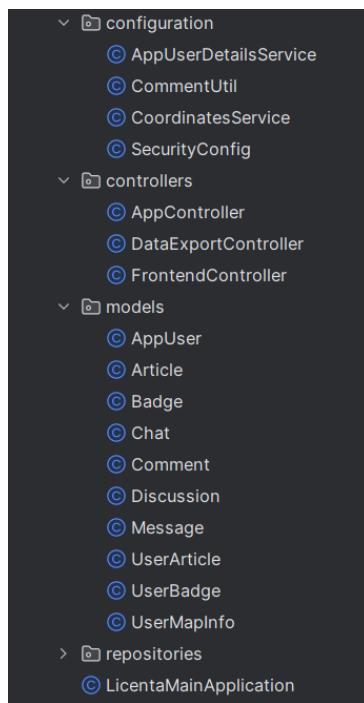


Figura 3.2: Structura proiectului

În pachetul **configuration** se regăsesc clasele de configurare ale aplicatiei, esențiale pentru stabilirea securității și gestionarea utilizatorilor.

Prin urmare, clasa SecurityConfig este responsabilă pentru configurarea securității în aplicatie și aceasta include configurații pentru autentificarea și autorizarea utilizatorilor și accesul acestora la resurse. Scopul acestei clase este să protejeze aplicația împotriva accesului neautorizat. În cazul aplicatiei mele, utilizatorii care nu au cont pe site-ul Uixplorer au acces doar la pagina principală, cea de contact, about și serviciul de înregistrare, spre deosebire de utilizatorii autentificați care pot vizualiza restul site-ului (articolele, forumul etc.).

Clasa AppUserDetailsService implementează UserDetailsService din Spring Security și este responsabilă cu încărcarea detaliilor utilizatoril din baza de date și transformarea acestora în obiecte de tip AppUser (modelul meu de utilizator). Clasa su-

prăscăie metoda `loadUserByUsername`, cea care se ocupă cu autentificarea utilizatorilor care au deja cont în aplicație.

Clasa `CommentUtil` oferă o singură metodă statică și este folosită pentru a converti URL-urile dintr-un text în link-uri HTML (care au tag-ul `<a>`). Aceasta este folosită atunci când un utilizator adaugă un comentariu (care conține un link) la o discuție din forum.

Tot în acest pachet se află și clasa `CoordinatesService`, cea care se ocupă de găsirea coordonatelor unui utilizator după câmpurile 'country' și 'state' din baza de date și calcularea distanțelor dintre utilizatori. Pentru a obține coordonatele (latitudinea și longitudinea) am folosit API-ul Nominatim OpenStreetMap, care extrage aceste coordonate pentru un anumit stat dintr-o țară. Pentru a calcula distanța dintre doi utilizatori, am folosit distanța Manhattan, care este suma distanțelor absolute dintre latitudinile și longitudinile dintre cei doi utilizatori. În continuare, am implementat funcția `getTop3ClosestUsers`, care se folosește de distanța Manhattan pentru a returna o listă cu cei mai apropiati trei utilizatori față de un anumit utilizator. Am implementat aceasta clasă pentru a putea oferi o recomandare utilizatorilor de a adăuga în lista lor de prieteni utilizatori care se află în aceeași zonă geografică.

Pachetul **controllers** conține trei clase/controllere care gestionează cererile HTTP și returnează răspunsurile adecvate.

Clasa `AppController` este un controller REST care gestionează cererile făcute de către un utilizator cu rolul 'Admin', precum `getAllUser`, care returnează toți utilizatorii din baza de date, sau `getAllArticles`.

`DataExportController` gestionează cererile pentru exportul de date, folosit pentru integrarea cu sistemul de recomandare scris în Python. În cazul site-ului meu, pentru a putea oferi o recomandare personalizată, sistemul are nevoie de lista de articole deja citite de către utilizator.

```

# toate articolele din baza de date
def get_articles():
    response = requests.get('http://localhost:2024/api/articles')
    return response.json()

# articolele citite de utilizator
def get_user_articles(user_id):
    response = requests.get(f'http://localhost:2024/api/users/{user_id}/articlesRead')
    return response.json()

# articole citite de toti utilizatorii din tabela user_article
def get_all_user_articles():
    response = requests.get('http://localhost:2024/api/user_articles')
    return response.json()

```

Figura 3.3: Metodele din DataExportController apelate în Python

Clasa FrontendController este responsabilă cu gestionarea cererilor care implică interfața utilizatorilor și logica de prezentare. Metodele din aceasta clasă vor returna fișiere HTML, Backend-ul aplicației fiind integrat cu Frontend-ul, folosind Thymeleaf. Astfel, în FrontendController vom stabili atributele pentru modelul paginii și sesiunea curentă a unui utilizator.

În continuare, avem pachetul **models** care conține modelele folosite în proiect. Un model este una dintre componentele fundamentale ale arhitecturii MVC și este cel care interacționează cu baza de date pentru a obține și manipula date, care apoi sunt furnizate către View și Controller.

Clasa AppUser reprezintă un utilizator al aplicației. Aceasta implementează interfața UserDetails pentru a folosi formatul standard de autentificare și autorizare din Spring Security. Clasa are câmpuri pentru informațiile personale (numele, email-ul, parola, poza de profil, țara și statul, data nașterii), roluri, lista de prieteni, articolele citite și badge-urile utilizatorului. Parola este criptată înainte de a fi salvată în baza de date folosind BCryptPasswordEncoder. Atunci când un utilizator se înregistrează, vor fi calculate și câmpurile pentru latitudine și longitudine.

Article reprezintă un articol în aplicație. Clasa are câmpuri pentru titlu, cele trei secțiuni de text și imagini, precum și sursa din care provine articolul. Acestea permit prezentarea articolelor într-un format detaliat.

Clasa Badge reprezintă o insignă ce poate fi obținută de utilizatori pe parcurs ce folosesc site-ul. Aceasta este definită printr-un nume și o imagine. Un exemplu de

Badge este "Log in for 5 days in a row", câștigat de utilizatorii care se autentifică pe site-ul Uixplorer cinci zile consecutive.

Comment definește un comentariu postat într-o discuție din forum-ul aplicației. Are câmpuri pentru mesajul comentariului, discuția de care aparține, ora și ziua la care a fost postat și câteva detalii despre utilizatorul care a adăugat comentariul (numele și poza).

Clasa Discussion reprezintă o discuție inițiată de către un utilizator în forum. Aceasta are un mesaj principal, o listă de comentarii asociate, ora și data postării și detalii despre utilizatorul care a postat discuția pe forum.

Clasa Message definește un mesaj ce aparține unei conversații dintre doi utilizatori. Aceasta conține textul mesajului, id-ul și poza utilizatorului care a trimis mesajul și conversația de tip Chat de care aparține.

Chat reprezintă o discuție dintre doi utilizatori care sunt prieteni. Prin urmare, clasa are câmpuri pentru cei doi utilizatori de tip AppUser și lista de mesaje dintre ei, de tip Message.

Clasele UserArticle și UserBadge reprezintă entități de legătură între AppUser și Article (indică ce articole au fost citite de către utilizatori), respectiv AppUser și Badge (indică ce badge-uri au fost obținute de către utilizatori).

Clasa UserInfoMap este folosită pentru a stoca doar câteva câmpuri despre un AppUser, necesare pentru a crea harta ce arată de unde sunt utilizatorii înregistrați pe site. Astfel, câmpurile necesare de la un utilizator vor fi numele lui, latitudinea și longitudinea.

În pachetul **repositories** am opt interfețe/repozitoare (UserRepo, ArticleRepo, BadgeRepo, CommentRepo, DiscussionRepo, UserArticleRepo, ChatRepo, MessageRepo), care reprezintă mecanismele de abstractizare a interacțiunilor cu baza de date. Fiecare interfață extinde interfața JpaRepository din Spring Data JPA, care oferă metode CRUD (Create, Read, Update, Delete) și alte operațiuni personalizate pentru gestionarea entităților din baza de date. Câteva metode standard din JpaRepository sunt: save, findById, findAll, deleteById. Aceste repozitoare sunt apoi injectate în servicii și controlere pentru a facilita gestionarea datelor din aplicație.

### 3.1.3 Sistemul de recomandare de articole

Sistemele de recomandare au ajuns tot mai populare și dorite în viața noastră de zi cu zi. Site-uri precum Youtube sau Netflix se folosesc de astfel de sisteme pentru a oferi utilizatorilor săi o experiență cât mai plăcută în navigarea pe site. Scopul acestor sisteme este de a recomanda unui utilizator ce și-ar putea dori să vizualizeze în funcție de videoclipurile sau filmele pe care le-a vizionat deja.

Un sistem de recomandare poate fi mai complicat, cum ar fi cele care utilizează deep learning, dar poate fi și simplu și se poate baza pe similaritatea dintre obiecte, calculând similaritatea cosinusului.

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}},$$

Figura 3.4: Funcția cosine\_similarity, sursă: Medium

Am ales să implementez în Uixplorer un sistem de recomandare personalizată bazată pe conținut (Content-Based Filtering) folosind Python. Aceasta va prelua articole din baza de date și va face recomandări de articole pentru utilizatori bazate pe similaritatea conținutului dintre articolele sale citite și cele necitite.

M-am folosit de librăria Pandas pentru a vectoriza articolele și pentru a crea un DataFrame din articolele obținute. Am concatenat titlul articolului cu conținutul său și am transformat rezultatul într-o matrice TF-IDF.

În continuare, m-am folosit de funcția cosine\_similarity din librăria scikit-learn care calculează similaritatea cosinus între articole folosind matricea TF-IDF și returnează o matrice de similaritate. Similaritatea cosinusului este o funcție de măsurare bazată pe distanța cosinusului dintre două obiecte și poate fi utilizată în sistemele de recomandare, cum ar fi cele care recomandă filme și cărți, iar, în cazul meu, articole. Sistemul va calcula scorurile de similaritate și va returna articolele cu cele mai mari scoruri de similaritate care nu au fost citite încă de utilizator. Pentru utilizatorii care nu au citi niciun articol, sistemul le va recomanda cele mai populare articole bazate pe frecvența citirii lor de către ceilalți utilizatori.

```

@app.route('/recommend', methods=['POST'])
def recommend():
    user_id = request.json.get('user_id')
    articles = get_articles()
    user_articles = get_user_articles(user_id)

    articles_df, tfidf_matrix = preprocess_articles(articles)
    cosine_sim = calculate_similarity(tfidf_matrix)

    user_articles_df = pd.DataFrame(user_articles)

    if user_articles_df.empty:
        all_user_articles = get_all_user_articles()
        recommended_articles = get_top_articles(all_user_articles, articles_df, top_n=3)
    else:
        recommended_articles = get_recommendations(user_articles_df, articles_df, cosine_sim, top_n=3)

    return jsonify(recommended_articles.to_dict(orient='records'))

```

Figura 3.5: Sistemul de recomandare

O altă metodă populară pentru a oferi recomandări personalizate sunt sistemele bazate pe colaborare (Collaborative Filtering), care se bazează pe comportamentul și preferințele utilizatorilor pentru a sugera obiecte similare cu cele apreciate de alți utilizatori cu gusturi similare. Acestea pot fi:

- User-Based: funcționează pe principiul că utilizatorii care au acordat evaluări similare unui anumit articol vor avea, cel mai probabil, aceeași preferință și pentru alte articole. Astfel, această metodă se bazează pe găsirea de similarități dintre utilizatori.
- Item-Based: se utilizează similaritatea dintre articole pe care un utilizator le-a apreciat în trecut pentru a oferi o recomandare.

Deși metoda Collaborative Filtering are multe avantaje, prin faptul că personalizarea este la un nivel mai ridicat, fiind bazată pe comportamentul și preferințele reale ale utilizatorilor, există și anumite dezavantaje, precum problema de a oferi o recomandare unor utilizatori noi care nu au suficient istoric de activitate, sau nevoie de a stoca un volum mare de date despre interacțiunile utilizatorilor cu articolele.

Am ales să folosesc în aplicația mea un sistem recomandare bazat pe conținut deoarece este mai potrivit pentru un site cu un număr redus de utilizatori activi și nu necesită informații detaliate despre utilizatori și preferințele lor, ci doar despre conținutul articolelor.

### 3.1.4 Baza de date

O bază de date este o colecție de date stocate într-un sistem informatic și controlată, de obicei, de un sistem de gestionare a bazelor de date (SGBD). Datele sunt modelate în tabele, ceea ce face ca interogarea și procesarea lor să fie eficiente. Limbajul de interogare structurat (SQL) este utilizat în mod obișnuit pentru interogarea și scrierea datelor.

În cadrul aplicației mele am ales să folosesc MySQL. MySQL este un sistem de gestionare a bazelor de date relationale (RDBMS) open-source dezvoltat de Oracle. Este cel mai popular SGBD, utilizat de companii cu nevoi masive de stocare a datelor, cum ar fi Facebook, Netflix, Uber și multe altele. Bazele de date relationale împart, organizează și stochează datele în diferite tabele.

Uixer folosește o bază de date formată din zece tabele, în care am stocat toate informațiile necesare pentru realizarea tuturor funcționalităților. În continuare voi explica rolul fiecărei tabele și asocierile dintre acestea.

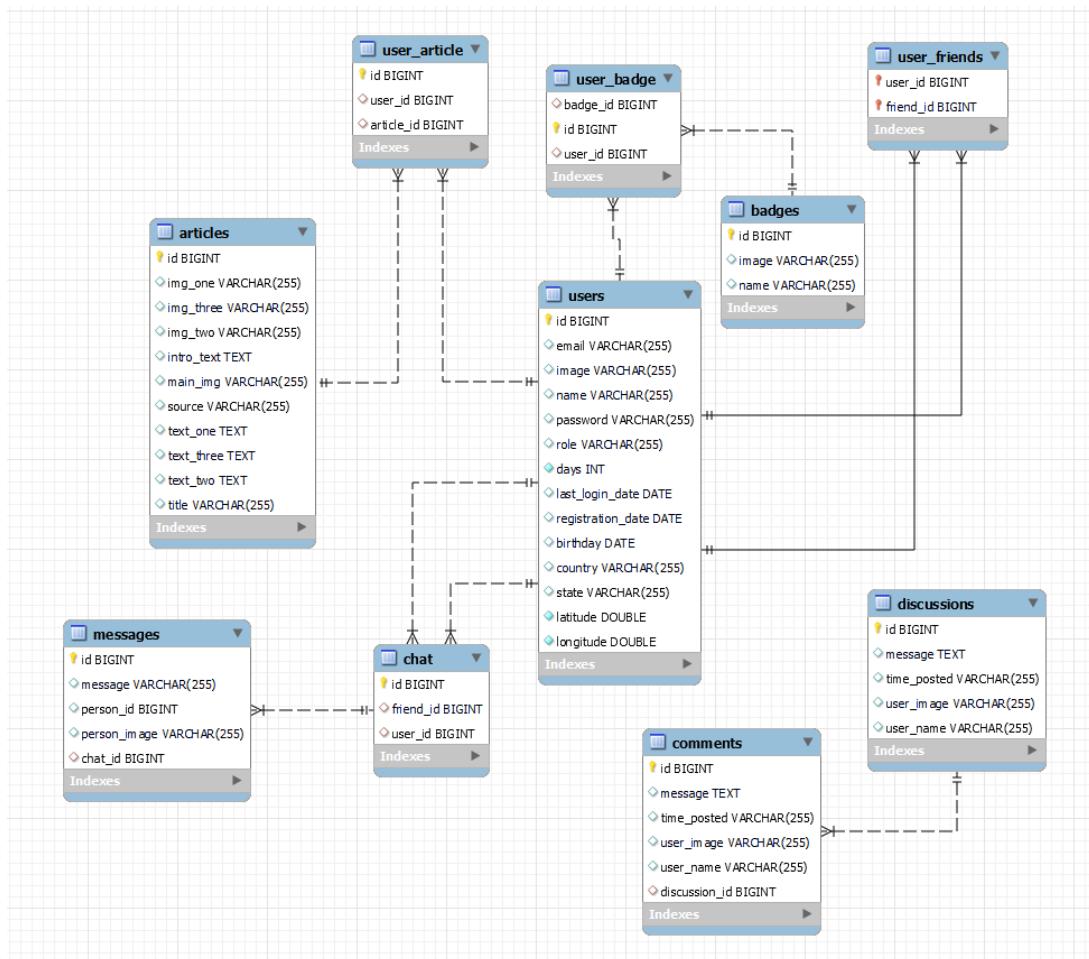


Figura 3.6: Diagramă generată de MySQL

- **users**: conține datele despre utilizatori. Coloanele tablei sunt: id (cheia primară a tablei), email, nume, parolă, poza de profil, rol, data când s-a logat ultima dată, data înregistrării, numărul de zile consecutive de autentificare. Tabela users este esențială pentru gestionarea autentificării și a autorizării utilizatorilor în aplicație.
- **articles**: reține informațiile articolelor. Conține coloane pentru: id (cheia primară), imaginea principală a articolului, introducerea, cele trei secțiuni de imagini și de text și sursa de unde a fost preluat.
- **discussions**: face referire la discuțiile din spațiul de forum. Coloanele regăsite aici sunt: id (cheia primară), mesajul discuției, data la care a fost postată, imaginea și numele utilizatorului care a postat discuția.
- **comments**: reprezintă un comentariu din cadrul unei discuții. Coloanele sale sunt: id (cheia primară), mesajul comentariului, numele și imaginea utilizatorului care a scris comentariul și id-ul discuției de care aparține, tabela comments și discussions fiind conectate pentru a permite fiecărui comentariu să fie asociat unei discuții.
- **badges**: stochează informații despre badge-urile pe care utilizatorii le pot căpăta. Conține coloanele id (cheia primară), numele badge-ului și poza.
- **user\_article**: este o tabelă de asociere între un users și articles și are rolul de a stoca articolele pe care un utilizator le-a citit. Aceasta conține coloanele: id (cheia primară), user\_id (id-ul utilizatorului), article\_id (id-ul articolului citit).
- **user\_badge**: este o tabelă de asociere între users și badges și are rolul de a stoca badge-urile pe care un utilizator le deține. Tabela are coloanele: id (cheia primară), user\_id (id-ul utilizatorului), badge\_id (id-ul badge-ului pe care utilizatorul îl deține).
- **user\_friends**: este o tabelă de asociere între doi utilizatori, marchează faptul că cei doi sunt prieteni pe site.
- **chat**: reprezintă o conversație dintre doi utilizatori care sunt prieteni și are coloanele: id (cheie primară), id-ul utilizatorului, id-ul prietenului și îi este asociată și o listă de mesaje de tip 'messages'.

- **messages**: stochează informațiile despre un mesaj din cadrul unei conversații dintre doi utilizatori. Coloanele sale sunt: id (cheie primară), textul mesajului, id-ul chat-ului de care aparține și id-ul și poza utilizatorului care a trimis mesajul.

## 3.2 Frontend

### 3.2.1 Tehnologii utilizate

Frontend-ul aplicației Uixplorer a fost dezvoltat folosind HTML, CSS și JavaScript pentru a crea o interfață atractivă și ușor de folosit pentru utilizatori. Pentru a integra frontend-ul cu backend-ul am utilizat Thymeleaf.

- **HTML** (HyperText Markup Language) este baza structurii paginilor mele web, fiind utilizat pentru a defini elementele fundamentale ale interfeței, precum butoane, formulare și alte componente necesare pentru interacțiunea utilizatorului. Structura unei pagini HTML se poate împărtăși în trei părți: header (unde se află linkurile de navigare a paginilor), main (conținutul paginii) și footer (aici, de obicei, se regăsesc linkurile de contact sau alte detalii precum număr de telefon, adresă).
- **CSS** (Cascading Style Sheets) este folosit pentru a stiliza paginile HTML și pentru a crea un aspect vizual plăcut și consistent. În cazul proiectului meu, am creat o paletă de culori și am setat anumite dimensiuni de fonturi (în fisierul style.css) pentru a menține consistența interfeței.
- **JavaScript** a fost utilizat pentru a adăuga interactivitate, precum afișarea unui meniu ascuns sau afișarea dinamică a unor statistici de pe site. De asemenea, am folosit JavaScript pentru a integra în proiectul meu un API, SpeechSynthesis, un serviciu care permite utilizatorilor să asculte conținutul unui articol, în loc să îl citească.
- **Thymeleaf** este un motor de şablonare pentru Java, ce mi-a permis să generez pagini HTML dinamice pe partea de server. Thymeleaf m-a ajutat să inserez datele direct din obiectele Java în şabloanele HTML, precum detalii despre utilizatori, facilitând astfel actualizarea dinamică a conținutului paginilor. Controllerele din aplicație vor primi cererile HTTP și vor returna şabloanele Thymeleaf completate cu datele necesare.

### 3.2.2 Interfața aplicației

În continuare, voi face o scurtă prezentare a funcționalităților proiectului meu și voi prezenta interfața paginilor.

- **Landing**

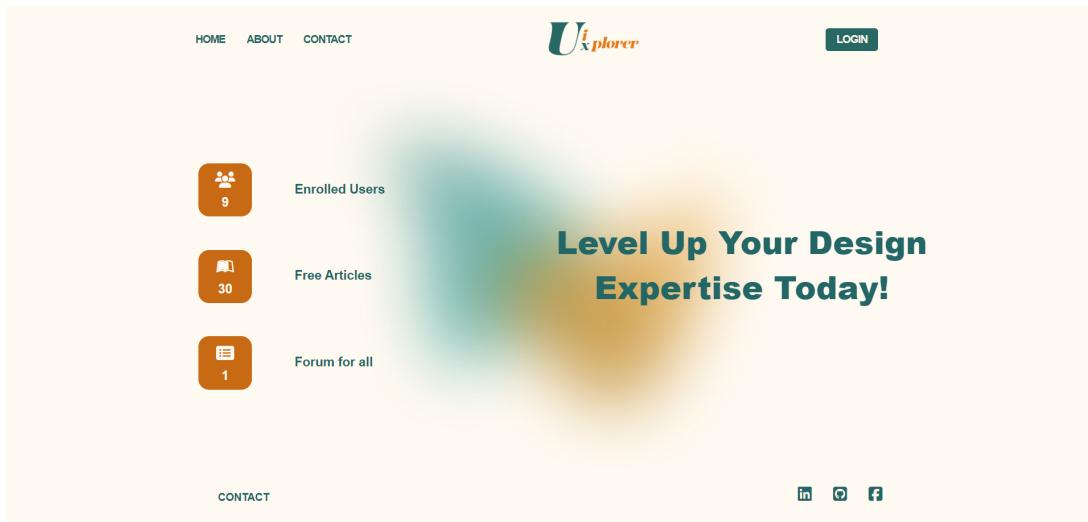


Figura 3.7: Pagina principală

Un utilizator neautentificat va avea acces doar la paginile about, contact și opțiunea de a se înregistra. În această pagină sunt afisate câteva statistici despre numărul de utilizatori și articole din aplicație.

- **Login**

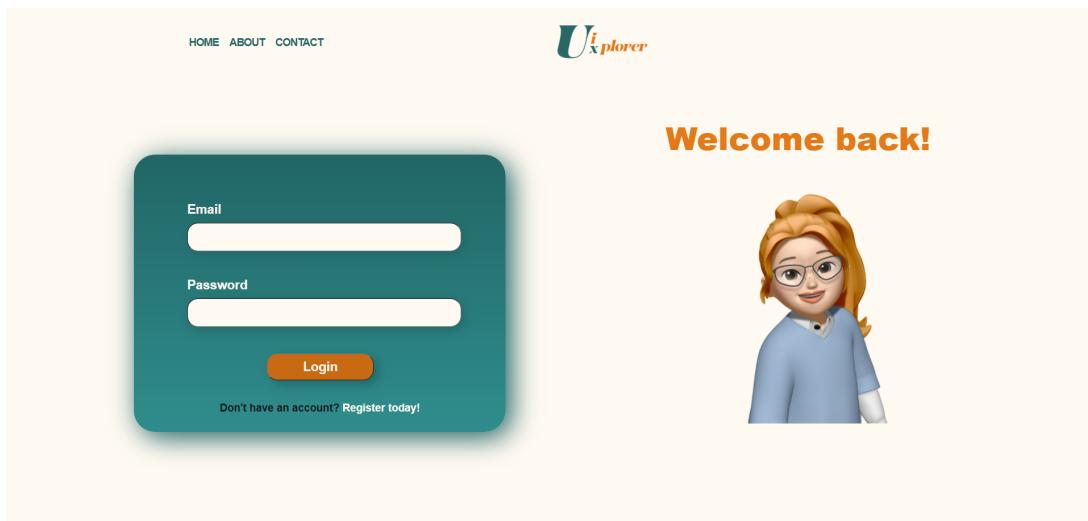


Figura 3.8: Pagina de logare

Un utilizator care are deja un cont pe site-ul Uixplorer, va trebui să își introducă adresa de email și parola pentru a se autentifica.

- **Register**

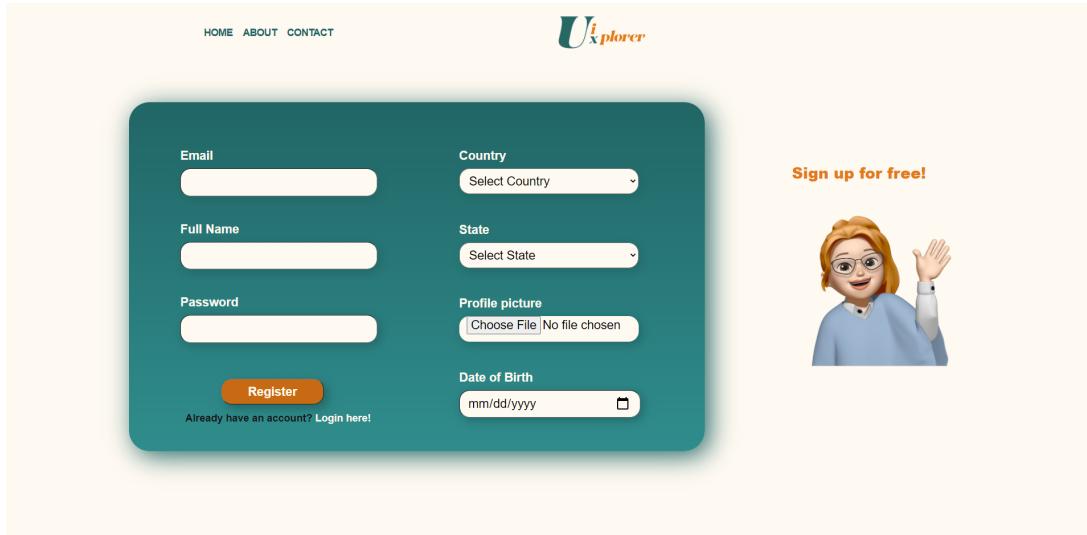


Figura 3.9: Pagina de înregistrare

Un utilizator care dorește să se înregistreze pe site va trebui să introducă o adresă de email, numele său, o parolă de cel puțin opt caractere (care să conțină minim o literă scrisă cu majuscule, o cifră și un caracter special), țara și statul de unde provine, o poză de profil și data nașterii.

- **Homepage**

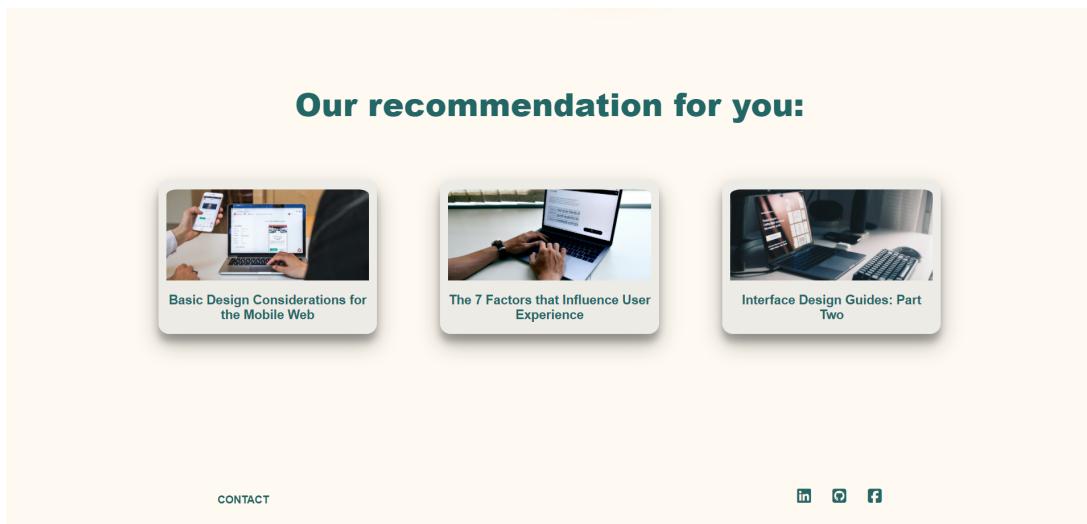


Figura 3.10: Recomandări de articole

După autentificare, utilizatorul va fi redirecționat către homepage, unde va putea vizualiza aceleasi statistici vizibile în Figura 3.7 (pagina Landing) și unde îi vor fi afișate recomandările de articole primite de la server. (Figura 3.10).

Tot aici, utilizatorului îi va fi recomandată o listă mai lungă de prieteni pe care poate să îi adauge și trei sugestii personalizate de prieteni (care sunt din aceeși zonă din punct de vedere geografic).

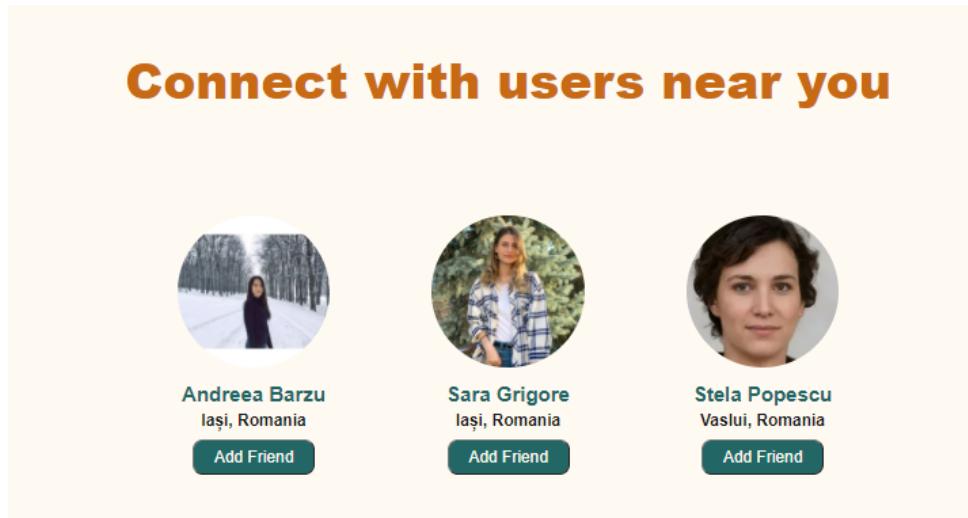


Figura 3.11: Recomandări de prieteni

De asemenea, am adăugat o hartă a utilizatorilor înregistrați din întreaga lume, folosind Leaflet, o bibliotecă JavaScript open-source utilizată pentru a crea hărți interactive. Harta este reprezentată prin iconițe (obiecte de tip UserMapInfo) în locurile unde se află câte un utilizator înregistrat în aplicație.

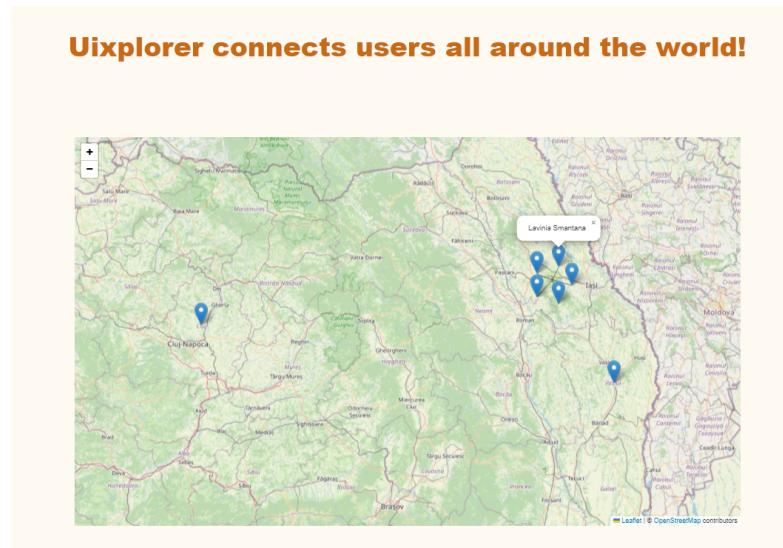


Figura 3.12: Harta utilizatorilor înregistrați

Odată ce un utilizator este înregistrat, acesta își va putea vizualiza lista de prieteni apăsând pe o iconă din header-ul paginilor și, de acolo, poate deschide conversația cu prietenii săi, unde va putea vizualiza istoricul mesajelor sau trimite un mesaj nou.

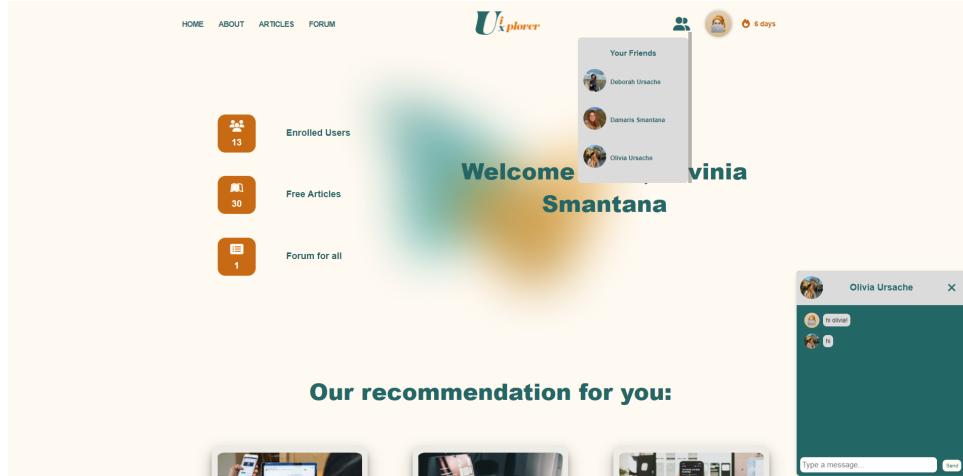


Figura 3.13: Lista de prieteni și o conversație între doi utilizatori

- **About**



Figura 3.14: Pagina About

Pagina About conține câteva informații despre site-ul Uixer, motivația din spatele dezvoltării acestui site și informații despre dezvoltatorul aplicației.

- Contact

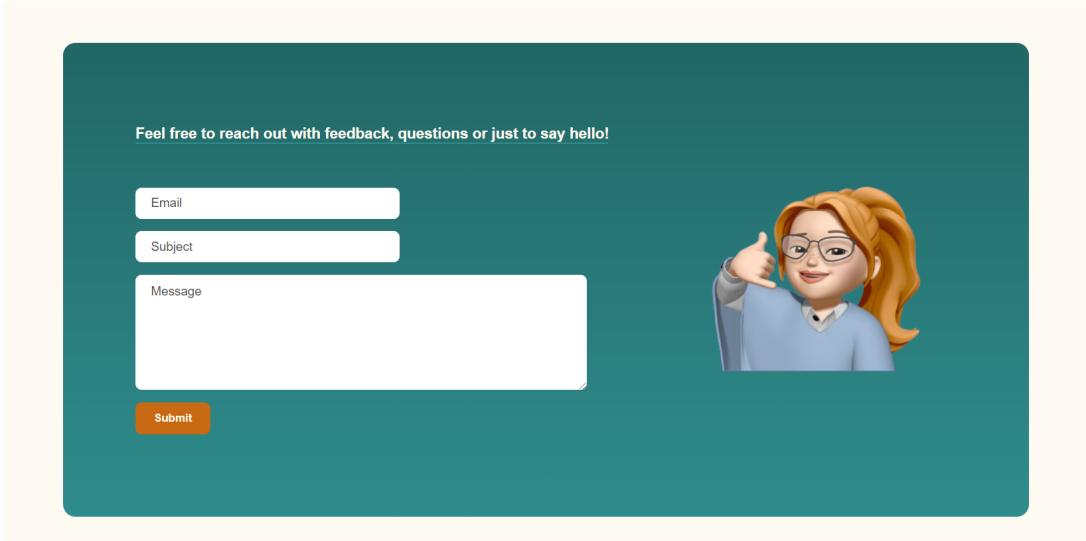


Figura 3.15: Formularul de contact

În pagina de contact am adăugat un formular folosind API-ul de la Web3Forms. Astfel, utilizatorii pot lua contact cu dezvoltatorul aplicației și pot trimite întrebări sau sugestii legate de site.

- Meniul de Articole

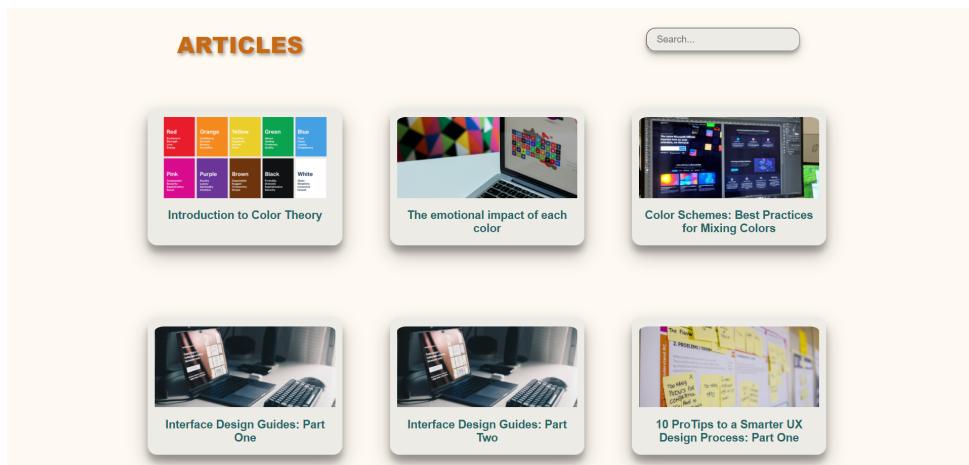


Figura 3.16: Navigarea articolelor

Aplicația Uixplorer oferă un catalog bogat și în continuă dezvoltare de articole ce prezintă concepte de UI/UX design. În meniul de articole putem vizualiza imaginea principală a unui articol și titlul acestuia, iar deasupra meniului avem un search bar pentru a putea căuta un articol după titlul său.

- Afisarea unui Articol

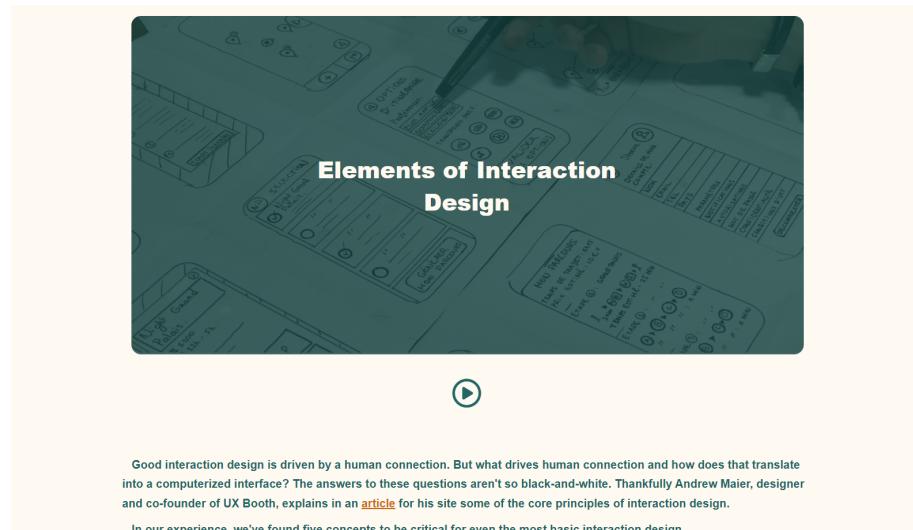


Figura 3.17: Afisarea unui Articol

Din meniul de articole, utilizatorii pot accesa câte un articol individual, acestea fiind reprezentate printr-un titlu, o imagine principală, conținutul său și sursa de unde a fost preluat. Sub poza articolului poate fi observat un buton de play, care este cel care activează funcția de speech și permite utilizatorilor să asculte conținutul unui articol. După ce termină de citit/ascultat, utilizatorii pot apăsa pe buton Finish Reading pentru a marca faptul că articolul respectiv a fost citit. Această acțiune ajută sistemul să ofere recomandări cât mai precise pentru utilizatori.

- Forum

**Forum**

[Start discussion](#)

Lavinia Smartana: Hi everyone! I've been working on a new app and am currently focused on improving the user onboarding experience. I want to ensure that new users understand how to use the app effectively and feel comfortable right from the start. I'm considering a series of interactive tutorials and tooltips, but I'm not sure if this is the best approach. Has anyone here implemented successful onboarding strategies? What methods did you find most effective in keeping users engaged and reducing the drop-off rate? Any tips or examples would be greatly appreciated!

Deborah Ursache: Great question! In my experience, combining a simple, concise tutorial with contextual tooltips works best. Too much information upfront can overwhelm users. Instead, try breaking it down into manageable chunks and guide users through key features as they encounter them. A welcome message with a brief overview can also help set the stage. Good luck!

Andreea Barzu: Check this out: <https://www.uxpin.com/studio/ebooks/ux-design-definitive-beginner-guide/>

Damaris Smartana: Check this out: <https://www.uxpin.com/studio/ebooks/ux-design-definitive-beginner-guide/>

Figura 3.18: Pagina Forum-ului

Am integrat un forum în această aplicație pentru a încuraja utilizatorii să își împărtășească ideile sau întrebările între ei. În aceasta pagină, un utilizator poate să înceapă o nouă discuție sau să vizualizeze discuțiile actuale și să adauge un comentariu la ele.

- **Adăugarea unei discuții în Forum**

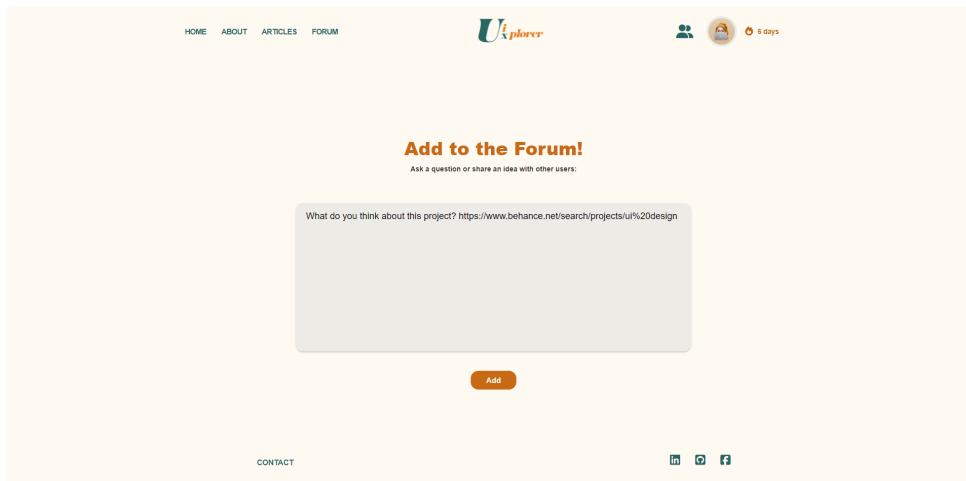


Figura 3.19: Adăugarea unei discuții

În pagina Forum, dacă utilizatorul apasă pe butonul 'Start discussion', acesta va intra pe pagina din poza de mai sus, unde va putea adăuga o nouă discuție în Forum, pe care o va putea vizualiza și la care se va putea alătura orice utilizator înregistrat.

- **Contul utilizatorului**

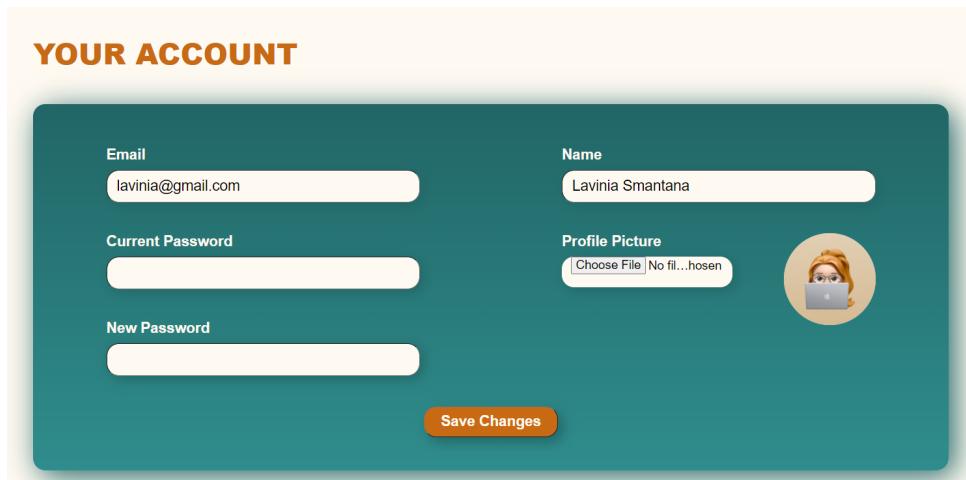


Figura 3.20: Contul unui utilizator

În pagina aceasta utilizatorul își va putea vedea datele contului său (numele, adresa de email, poza de profil) și va putea să își modifice aceste date, inclusiv parola, care poate fi schimbată doar dacă utilizatorul își știe parola actuală.

- **Dashboard**

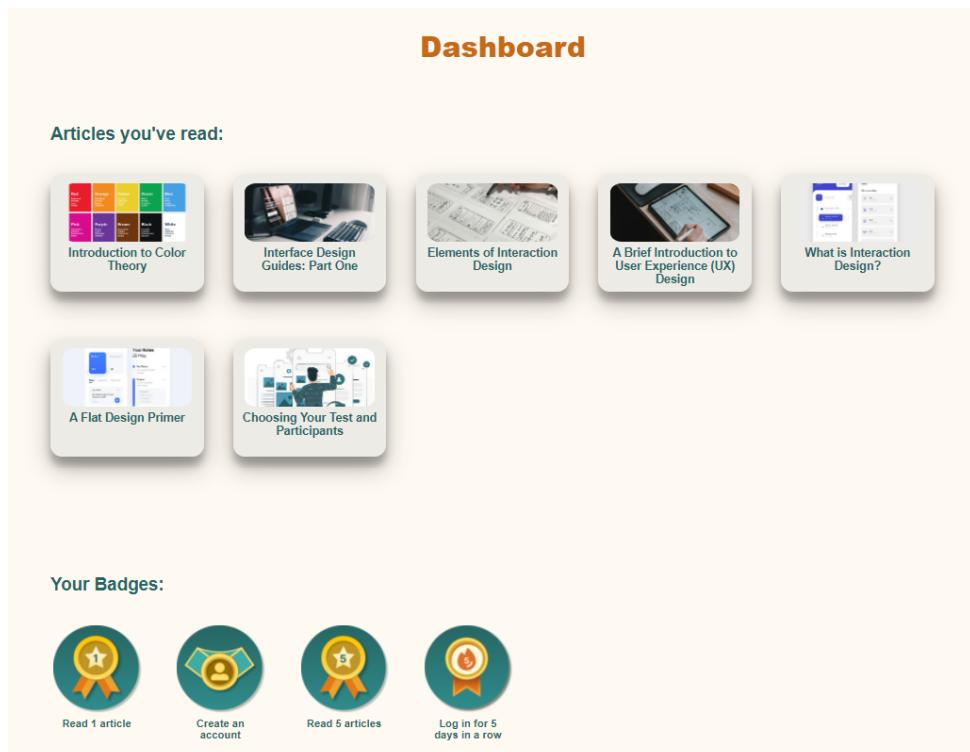


Figura 3.21: Pagina Dashboard

Am adăugat un dashboard în cadrul proiectului meu, unde utilizatorii vor putea vedea care sunt articolele pe care deja le-au citit și, de asemenea, ce badge-uri a căpătat până la momentul actual. Aceștia vor putea câștiga badge-uri pe parcurs ce folosesc aplicația.

# Concluzii

În timpul facultății, am dezvoltat o pasiune pentru UI/UX design și am realizat cât de importante sunt detaliile unei interfeței și modul în care preferințele publicului țintă influențează acestea. Cercetând platformele deja existente, am observat că pe lângă nevoia constantă de cursuri, articole sau materiale video actuale, persoanelor din acest domeniu le lipsește un loc unde să poată primi recomandări după nevoile lor și unde să poată descoperi alți designeri, fie ei începători sau avansați.

Uixplorer reprezintă rezultatul încercării de a crea o platformă de e-learning care să acopere aceste nevoi și, în același timp, să fie intuitivă și ușor de folosit, în care un utilizator să nu trebuiască să petreacă ore întregi navigând printre cursuri sau materiale.

În dezvoltarea acestui site am întâmpinat anumite provăcări, precum adaptarea la noile tehnologii folosite sau luarea deciziilor de proiectare, însă am reușit să îmi ating scopul final și să creez un produs relevant și util pentru designerii din ziua de azi.

Consider că aplicația încă mai poate fi îmbunătățită și că există multe funcționalități care ar putea fi adăugate în viitor, precum: un sistem de recomandare mai avansat, care să urmărească și preferințele utilizatorilor care fac parte din aceeași categorie atunci când oferă o recomandare sau un forum mai detaliat, care să fie împărtit pe categorii pentru a găsi discuțiile cele mai relevante. De asemenea, pe lângă conceptele teoretice reprezentate de articole, aplicația ar putea fi îmbunătățită prin implementarea unor exerciții practice, care să îi ajute pe utilizatori să aplice lucrurile învățate.

În concluzie, deși există loc de îmbunătățiri, consider că aplicația mea aduce o soluție excelentă pentru designeri și că Uixplorer este mai mult decât o simplă platformă educațională, ci este un loc unde pasionații de UI/UX design pot învăța, interacționa și progrresa în cariera lor.

# Bibliografie

- Personas – A Simple Introduction: <https://www.interaction-design.org/literature/article/personas-why-and-how-you-should-use-them>
- The MVC Architecture: <https://medium.com/@sadikarahmantanisha/the-mvc-architecture-97d47e071eb2>
- The Fundamentals of Web Application Architecture: <https://reinvently.com/blog/fundamentals-web-application-architecture/>
- Cristian Frăsinaru, *Curs Practic de Java*
- Collaborative Filtering Vs Content-Based Filtering for Recommender Systems: <https://analyticsindiamag.com/collaborative-filtering-vs-content-based-filtering-for-recommender-systems/>
- The cosine similarity and its use in recommendation systems: <https://naomy-gomes.medium.com/the-cosine-similarity-and-its-use-in-recommendation-systems-cb2ebd811ce1>
- What is Database? Types of Databases: <https://www.geeksforgeeks.org/what-is-database/>
- What is MySQL and what is it used for: <https://planetscale.com/learn/articles/what-is-mysql>