

UNIVERSITATEA „ALEXANDRU IOAN CUZA” IAȘI
FACULTATEA DE INFORMATICĂ



LUCRARE DE LICENȚĂ
MANAGEMENTUL LUCRĂRILOR DE LICENȚĂ

propusă de

Tanasă Nicoleta

Sesiunea: **iulie, 2018**

Coordonator științific

Colab. Olariu Florin

UNIVERSITATEA „ALEXANDRU IOAN CUZA” IAȘI
FACULTATEA DE INFORMATICĂ

MANAGEMENTUL LUCRĂRILOR DE LICENȚĂ

Tanasă Nicoleta

Sesiunea: **iulie, 2018**

Coordonator științific

Colab. Olariu Florin

Avizat,

Îndrumător Lucrare de Licență,

Titlul, Numele și prenumele _____

Data _____ Semnătura _____

DECLARAȚIE privind originalitatea conținutului lucrării de licență

Subsemntata Tanasă Nicoleta, având domiciliul în sat Stejaru, comuna Fărcașa, județul Neamț, născută la data de 06.12.1995 identificată prin CNP 2951206271543, absolventă a Universității „Alexandru Ioan Cuza” din Iași, Facultatea de Informatică, specializarea Infomatică, promoția 2017, 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: Managementul lucrărilor de licență, elaborată sub îndrumarea dl. Colab. Florin Olariu, 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, consimțind inclusiv la introducerea conținutului său 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 diploma 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 „*Managementul lucrărilor de licență*”, codul sursă al programelor și celelalte conținuturi (grafice, multimedia, date de test etc.) care însoțesc 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” 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.

Iași,

Absolvent Tanasă Nicoleta

Cuprins

1. Contribuții	15
2. Descrierea problemei.....	16
3. Abordări anterioare	17
4. Descrierea soluției.....	18
4.1 Principalele funcționalități	18
4.2 Diagrame	31
4.2.1 Diagrame pentru cazuri de utilizare	31
4.2.2 Diagrama structurii soluției.....	34
4.3 Modelare datelor	36
4.4 Comunicarea server-client.....	39

Introducere

Motivație

Susținerea lucrării de licență reprezintă un pas important în viața fiecărui student aflat în an terminal, aceasta dându-i încă o șansă de a demonstra că se pot pune în practică toate cunoștințele acumulate pe parcursul întregii facultății. El poate să realizeze și să arate, de asemenea, faptul că și-a asumat toate aceste cunoștințe în mod conștiincios și reușita asocierii lor într-un mod cât mai productiv să îi deschidă noi orizonturi.

Pentru a ajunge cu bine la momentul în care lucrarea de licență trebuie prezentată în fața comisiei de evaluatori, un student trebuie să se pregătească intens pe tot parcursul anilor de studii prin gândire, analiză, asociere și mai ales lucru practic. La început de an terminal, când se consideră că studentul are cunoștințe suficiente, începe efectiv perioada dezvoltării lucrării de licență. Acesta este un parcurs anevoios, cu multe ore de așteptare pe holurile facultății, cu multe email-uri trimise și, mai ales, cu multe întrebări la care studentul nu cunoaște răspunsurile.

Uneori studentul, poate din cauză că nu se consideră suficient de bine pregătit pentru a merge și a vorbi cu un anumit coordonator, pierde șansa de a realiza o lucrare excelentă și de a avea o colaborare minunată. Sau chiar, din cauza numărului mare de studenți, poate un profesor nu are șansa de a cunoaște fiecare student în parte și pierde și acesta, de asemenea, șansa unei colaborări eficiente din care să rezulte numai lucruri pozitive.

Chiar și după ce primul pas a fost făcut, și studentul are asignat un profesor coordonator și o temă pentru viitoare lucrare de licență, din cauza modului curent de viață, de cele mai multe ori, nici viitorul absolvent, nici coordonatorul științific al acestuia nu dispun de o perioadă de timp suficient de mare încât să asigure o colaborare ce acoperă în totalitate nevoile studentului și așteptările coordonatorului.

Datorită tuturor acestor motive, am ales să realizez o aplicație web care să faciliteze întreg procesul de supraveghere a dezvoltării acestei lucrări pentru profesorii coordonatori și procesul coordonat al studentului.

Obiective generale

Având ca principal mediu de utilizare cel universitar, studenții își vor putea crea un cont fiind restricționați de apartenența lor la o facultate. Deci aceștia vor folosi email-ul facultății curente pentru a se putea înregistra și mai apoi loga. De crearea conturilor coordonatorilor se va ocupa o persoană desemnată pentru acest lucru, rolul acesteia fiind de administrator.

Un student va avea la dispoziție o listă care va conține numele coordonatorilor disponibili pentru o anumită sesiune de licență și va putea aplica la unul dintre acei profesori urmând ca cererea lor să fie acceptată sau respinsă de către coordonator. În același timp, dacă un profesor are o idee prededefinită o va putea face publică astfel încât studenții interesați de subiecte/domenii să poată lua la cunoștință aceste aspecte.

Aplicarea unui student la un îndrumător va presupune, de asemenea, scrierea unui paragraf în care să ofere câteva detalii despre o posibilă idee pentru viitoarea temă sau va putea cere o sugestie în caz că nu are încă ceva clar în minte.

Acceptarea sau respingerea cererii unui student de către un coordonator va ține strict de acesta. El va avea posibilitatea să își păstreze criteriile de departajare pe care le-a utilizat până în momentul de față.

Un coordnator își va putea alocă un număr maxim de studenți, sub o limită stabilită în prealabil de reprezentanții facultății, pentru fiecare sesiune de prezentări în funcție de planurile proprii iar după cel acel număr va fi depășit niciun alt student nu va mai putea aplica pentru îndrumarea lui.

Dacă cererea unui student este acceptată el nu va mai putea aplica pentru un alt coordonator iar în caz contrar aplicabilitatea rămâne valabilă până când acesta își va găsi îndrumătorul, procesul de selectare a unui îndrumător continuând în același mod: aplicare cu o primă idee despre temă sau cererea unui sfat și mai apoi acceptare/respingere din partea coordonatorului.

În perioada dintre momentul aplicării, din partea studentului, și momentul acceptării/respingerii, din partea cadrului didactic, studentul va fi redirectionat către aceeași pagină în care sunt listate opțiunile domnilor profesori, dar opțiunea de a aplica din nou va fi blocată. În cazul în care la un termen hotărât de către reprezentanții facultății vor exista studenți

care nu au un profesor coordonator asociat atunci respectivii studenți se vor adresa secretariatului și își vor prezenta motivele pentru care doresc să fie coordonați de un cadru didactic dar nu au reușit să se înscrie în timp util, și vor rămâne să susțină lucrarea de licență într-o sesiune ulterioară.

După acest pas va începe efectiv interacțiunea coordonator-student. Fiecare student va putea accesa pagina coordonatorului său unde acesta va posta periodic anunțuri, termene limită și orice alt tip de conținut dorește să îl aducă la cunoștință studenților săi. Studenții vor putea, deasemenea, posta întrebări/nelămuriri pe această pagină, aceasta fiind publică pentru toți studenții înscriși la respectivul coordonator.

Dacă cei doi o să își dorească totuși o întâlnire față în față vor trebui doar să se puna de acord cu data și ora întâlnirii, aceste date fiind completate într-un loc special rezervat care va fi prestabilit de coordonator în funcție de orele sale disponibile.

Când studentul începe munca propriu-zisă, prima versiune a lucrării sale va fi încărcată într-un sistem de versionare și mai apoi un link către acest sistem va fi pus la dispoziție, urmând ca profesorul coordonator să o verifice și să acorde sfaturi de care studentul să țină cont mai departe. Pe parcursul evoluției lucrării, studentul nu va trebui decât să se asigure că toate modificările lui sunt actualizate în sistemul de versionare folosit.

Ceilalți studenți înscriși la același profesor coordonator vor putea de asemenea să vadă lucrările tuturor colegilor și eventual să își spună părerea pentru a îl ajuta pe studentul posesor al lucrării să o relizeze cât mai bine cu putință.

Atât coordonatorul cât și studentul vor avea acces la o serie de grafice realizate pe baza sistemului de versionare folosit de către student care vor furniza informații de tipul: de câte ori un student și-a actualizat lucrarea raportat la săptămână, ultimele actualizări ale lucrării sau proporția limbajelor de programare folosite. Aceste informații vor fi obținute din sistemul de versionare folosit de către student cu ajutorul GitHub API¹ (API ce expune informații referitoare la codul sursă pe baza acțiunilor utilizatorilor).

Fiecare student și coordonator va avea acces la o pagină special construită pentru a le furniza acestora informații despre perioada susținerii lucrării de licență, perioadele în care se desfășoră

¹ GitHub API: <https://developer.github.com/v3/>

înscriserile, documentele necesare și sfaturi utile pentru a putea fi siguri că totul va merge bine pe parcursul prezentării lucrării.

Pe lângă rolurile de student și coordonator, un alt rol secundar există în cadrul aplicației – cel de administrator. Acesta va avea responsabilitatea de a adăuga noi profesori coordonatori în sistem, de a șterge din coordonatorii existenți dacă aceștia vor deveni indisponibili sau de a modifica datele deja înregistrate în cazul în care va fi nevoie.

Scurtă descriere a soluției

Aplicația curentă fiind una accesibilă cu ajutorul internetului, soluția construită poate fi descrisă ca o aplicație de tip client-server cu următoarea componență:

- **Aplicația server:** reprezentată sub forma unui API REST² care expune extern informații, prin intermediul serviciilor, și care ajută la manipularea datelor;
- **Aplicația client:** reprezentată de o interfață grafică, ce utilizează serviciile oferite de către aplicația server, apelându-le prin intermediul protocolului HTTP³, și expune informațiile obținute către utilizatori oferindu-le, de asemenea, posibilitatea manipulării datelor;
- **Baza de date:** reprezentată de o bază de date de tip relațional, cu ajutorul căreia datele necesare aplicației sunt stocate și persistate.

Abordare tehnică





Limbaje de programare

.NET Core este versiunea complet rescrisă a principalului framework Microsoft, .NET Framework. Acesta are la bază limbajul C# și are ca principale avantaje faptul că poate rula pe mai multe platforme (nu doar Windows, cum eram obișnuiți, ci și pe Linux sau MacOS), compatibilitatea totală cu vechiul framework, menționat anterior, și faptul că există posibilitatea accesării codului sursă, acesta fiind public.

² Principii REST: <https://www.c-sharpcorner.com/uploadfile/kalisk/rest-fundamentals/>

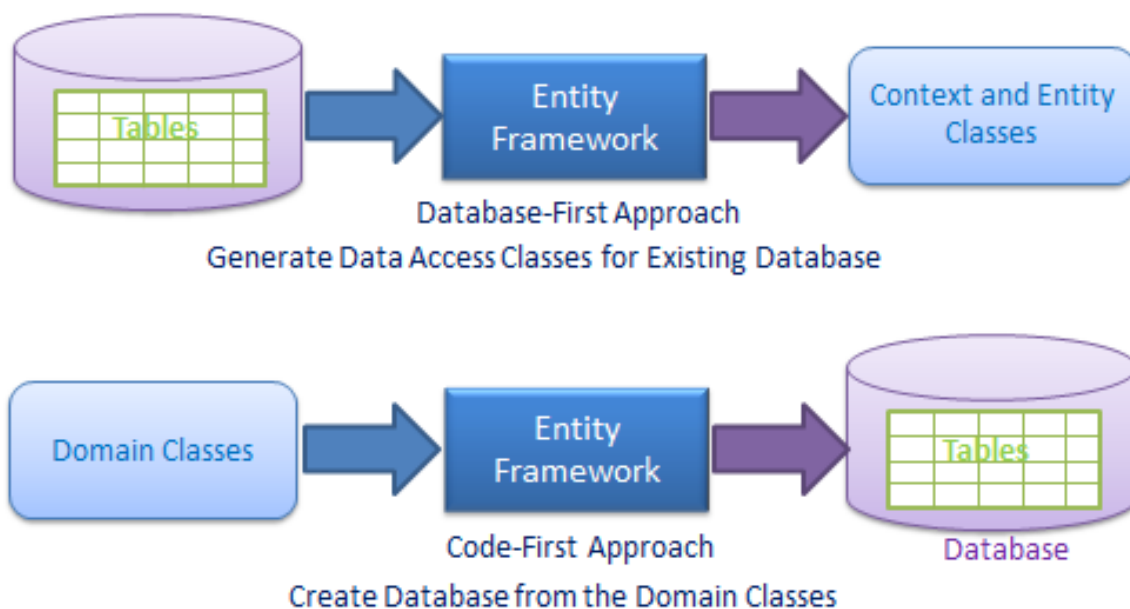
³ Hypertext Transfer Protocol: <https://developer.mozilla.org/en-US/docs/Web/HTTP>

ASP.NET Core este un framework folosit pentru crearea aplicațiilor de tip web sau cloud. Acesta este complet accesibil și disponibil pe site-ul GitHub⁴, site de găzduire al proiectelor software . Utilitatea principală a acestui framework este că aplicațiile dezvoltate cu ajutorul lui pot rula atât pe tehnologia .NET Core, cât și pe .NET Framework (fapt vizibil în *Figura 1*).

ASP.NET 4.6	ASP.NET Core 1.0
.NET Framework 4.6 	.NET Core 1.0   
.NET framework libraries	.NET core libraries

*Figura 1: ASP.NET vs. ASP .NET Core*⁵

Entity Framework Core este versiunea independentă de platformă al cadrului Entity Framework, acesta fiind un ORM⁶ (Object Relational Mapping) care permite dezvoltatorilor .NET să lucreze cu o bază de date folosind obiecte. Există două abordări posibile ale acestei versiuni, și anume Code-First (mai întâi se scriu modelele și mai apoi se generează tabelele bazei de date cu ajutorul lor) și Database-First (mai întâi se crează tabelele bazei de date și mai apoi se generează modelele aferente), după cum se observă în *Figura 2*.



⁴ GitHub: <https://github.com/>

⁵ Sursa: <http://www.jomendez.com/2017/02/15/asp-net-core-1-0/>

⁶ ORM: <https://searchwindevelopment.techtarget.com/definition/object-relational-mapping>

Figura 2: Abordări Entity Framework Core⁷

Din aceste două abordări, cea utilizată în cadrul aplicației curente este Entity Framework Core Code-First datorită faptului că permite utilizarea structurilor de tip clasă pentru a reprezenta modelul pe care se bazează Entity Framework pentru a efectua orice fel de operații la baza de date, codul fiind scris mai întâi și apoi generându-se modelul pe baza acestuia.

HTML (HyperText Markup Language) este un limbaj de marcare utilizat pentru crearea paginilor web ce pot fi afișate într-un browser scopul acestuia fiind de a prezenta informațiile (font, tabelă, paragraf) și putând fi construit folosind un simplu editor de texte. Elementele acestuia, reprezentate prin structuri numite tag-uri, alcătuiesc scheletul unei pagini, unele exemple fiind <head>, <body>, <header>, <footer>⁸.

SCSS (Sassy CSS) este un superset al limbajului CSS (Cascading Style Sheets) care se scrie în același mod dar permite folosirea caracteristicilor limbajului SASS (Syntactically Awesome Style Sheets), acesta fiind un preprocesor CSS. Utilitatea acestuia în construirea aplicației mele constă, în principal, în utilizarea variabilelor pentru memoria diferitelor elemente (culoare, font) care ajută la scrierea unui cod curat, ușor mentenabil și fără părți duplicate, având două exemple expuse în *Figura 3*, respectiv *Figura 4*.

```
$text-font: 'Lucida Sans';  
$text-color: ■rgba(94, 64, 56, 1);
```

Figura 3: Declararea unor variabile SCSS

```
.sidebar_content{  
  cursor: pointer;  
  color: $text-color;  
  font-family: $text-font;  
  font-size: $text-size-header;  
  margin: 1% 5%;  
  outline: none;  
}
```

Figura 4: Folosirea unor variabile SCSS

Angular este un framework structural construit în întregime cu ajutorul limbajului TypeScript, care este un superset al limbajului JavaScript, ce poate fi folosit pentru a crea partea de client a

⁷ Sursa: <http://www.entityframeworktutorial.net/efcore/entity-framework-core.aspx>

⁸ HTML pentru începători: <https://www.w3schools.com/html/>

unei aplicații. Avantajul major pe care l-am avut folosind ambele tehnologii a fost ușurința transmiterii informațiilor între server și pagina web, fiind posibilă stocarea datelor în obiecte ușor manipulabile, validarea facilă a datelor și eliminarea codului duplicat prin utilizarea de metode. În figura de mai jos (*Figura 5*) se găsesc menționate alte patru avantaje ale utilizării Angular.

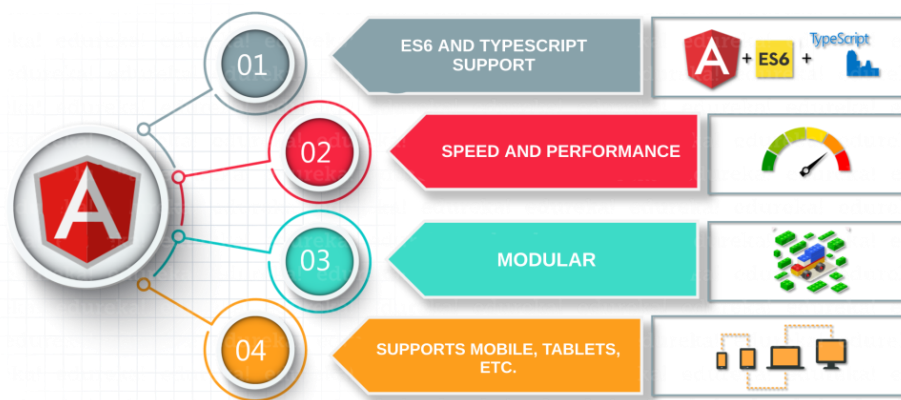


Figura 5: Avantajele utilizării Angular⁹

C# este un limbaj de programare orientat-obiect conceput de Microsoft la sfârșitul anilor 90. .NET Core permite dezvoltarea aplicațiilor folosind oricare din limbajele C#, F# sau Visual Basic. Eu am ales folosirea primului dintre ele deoarece este unul dintre cele mai puternice, pe care îl utilizează de milioane de alți oameni după cum poate fi observat în *Figura 7*.

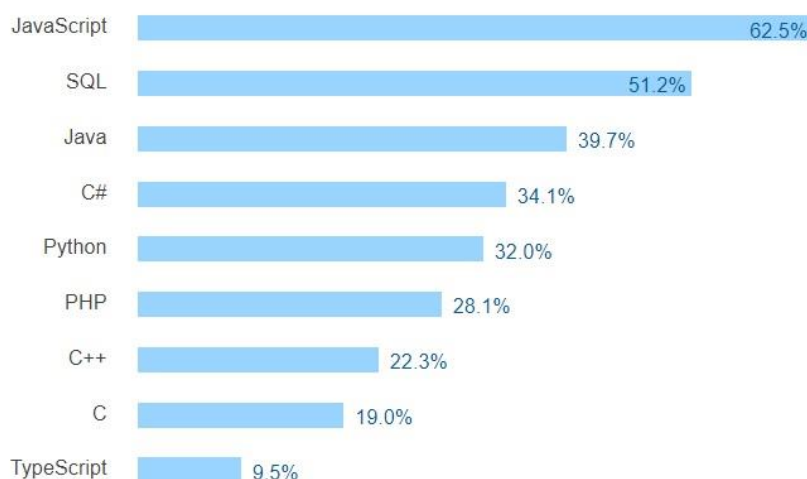


Figura 7: Fragment dintr-o imagine care conține cele mai populare tehnologii în 2017¹⁰

⁹ Sursa: <https://softwaredevelopment.ae/angular-best-solution-2018-web-app/>

¹⁰ Studiu, imagine: <https://insights.stackoverflow.com/survey/2017#mostpopular-technologies>

Instrumente software

Visual Studio Code¹¹ este un editor de cod sursă ușor de utilizat, dar puternic, care rulează pe desktop și este disponibil pentru Windows, MacOS și Linux. Dispune de suport încorporat pentru TypeScript, acesta fiind principalul motiv pentru care am ales să îl folosesc pentru dezvoltarea interfeței cu utilizatorul.

Visual Studio este, de asemenea, un editor de cod sursă, cu ajutorul căruia pot fi contruite diferite tipuri de aplicații (desktop, servicii, mobile). L-am utilizat datorită multitudinii de avantaje pe care acesta le oferă. Unele dintre ele sunt: sugestiile pe care acesta le oferă în timp real, modul în care sunt afișate și generate erorile de compilare și ușurința creării unei structuri optime de pentru aplicație.

Postman este o unealtă software care poate fi utilizată pentru a simula cererile HTTP care ar fi în mod normal trimise de către un client către server. Utilitatea lui a fost posibilitatea testării serviciilor oferite de către server, după cum se poate observa în *Figura 8*.

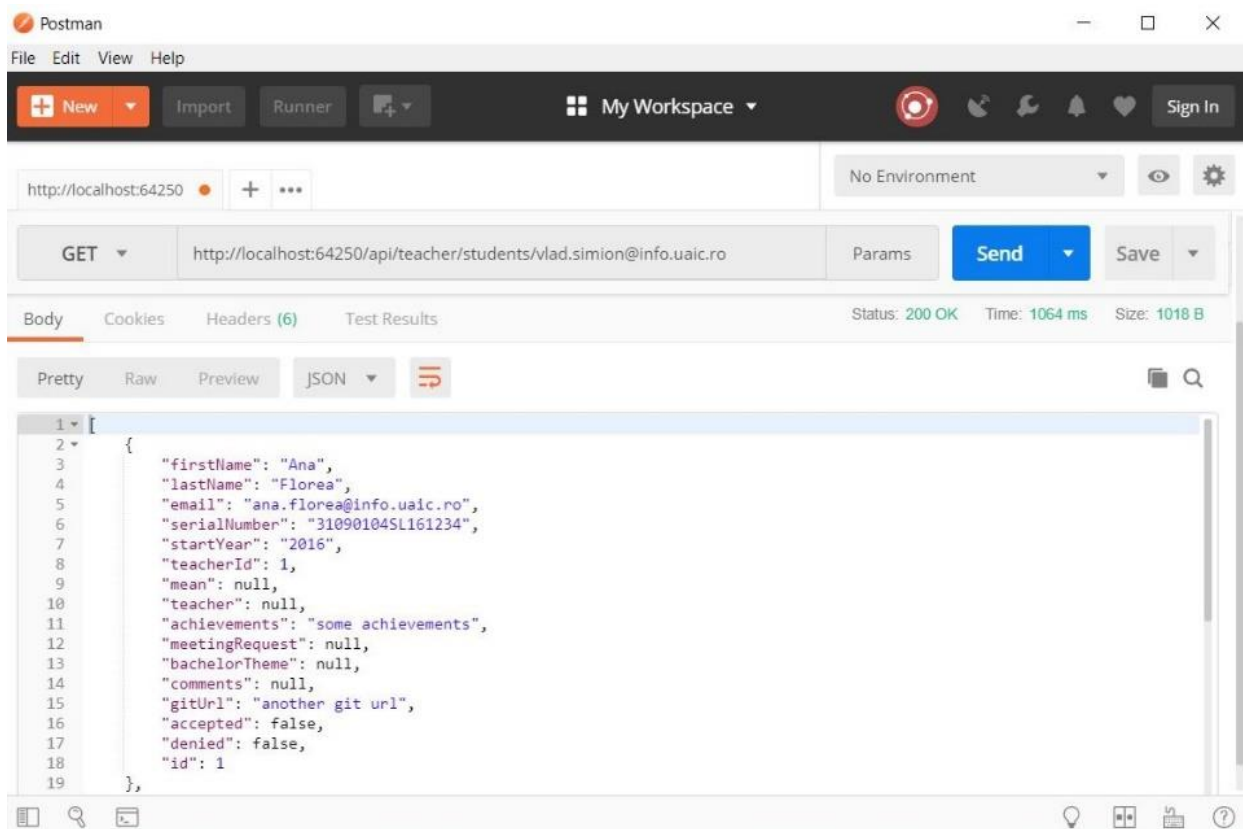


Figura 8: Cerere de tip GET și răspuns de la server, cu ajutorul Postman

¹¹ Documentație Visual Studio Code: <https://code.visualstudio.com/docs>

SourceTree este o aplicație care joacă rolul de client pentru sistemul de versionare Git¹². Lucrul cu sistemul de versionare este deci ușurat și încurajat cu ajutorul acestei interfețe grafice. Două din avantajele majore al folosirii acestora este siguranța pe care o oferă în stocarea codului sursă (și a orice altor documente) și posibilitatea folosirii oricărei versiuni anterioare de cod. Aceasta oferă cu ușurință informații despre toate modificările efectuate, inclusiv despre cele care se află în curs de modificare, lucru vizibil în *Figura 9*.

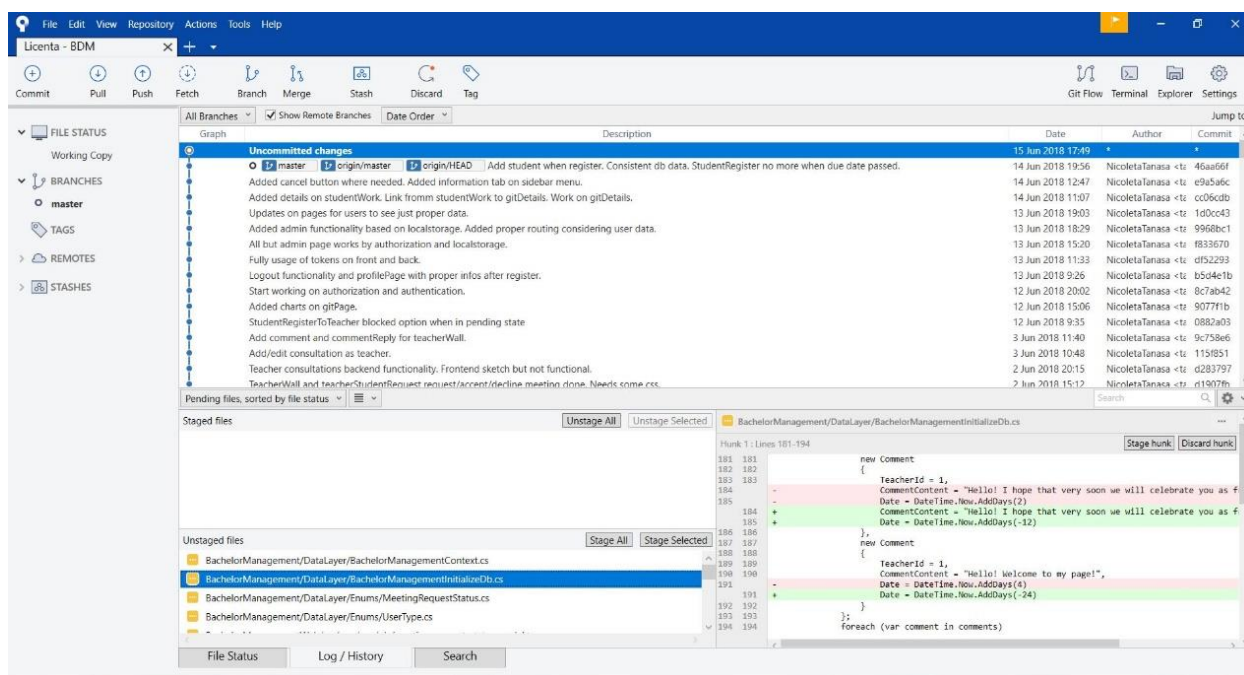


Figura 9: Interfața SourceTree

¹² Detalii GitHub: <https://www.atlassian.com/git/tutorials/what-is-git>

1. Contribuții

Contribuția personală în realizarea prezentei lucrări a constat în alegerea temei, propunerea ei către domnul profesor coordonator și mai apoi discutarea ideilor. Petrecând niște ani superbi în cadrul studenției mi-am propus ca lucrarea mea de licență să poată fi baza unei aplicații generice care să aibă scopul de a ajuta studenții. În acest sens m-am gândit inițial la o platformă pentru organizarea și gestionarea materialelor aferente cursurilor și seminariilor predate în cadrul facultății dar, având în vedere că domnii profesori deja folosesc anumite spații pentru a-și pune resursele la îndemâna studenților, m-am reorientat spre un tip de aplicație ce nu există încă în cadrul facultății și nici măcar în cadrul universității.

Pentru realizarea obiectivului, am creat o aplicație de tip client-server pe care am dezvoltat-o pe parcursul mai multor luni, folosind varii unelte și limbaje de programare. Inițial, am creat codul ce a generat structura bazei de date, apoi am creat câteva date ce populează baza menționată anterior, iar în cea mai mare parte a timpului am lucrat, în paralel, la construirea serverului și interfeței grafice.

Prin utilizarea unei multitudini de tehnologii moderne (spre exemplu Angular 4 și .NET core 2.0), respectarea bunelor practici atât în arhitectura cât și în implementarea aplicației și prin ușurința cu care aceasta poate fi extinsă, am reușit să creez o aplicație care să ajute studenții și chiar să îi provoace în a continua munca începută de mine.

Pe tot parcursul dezvoltării aplicației am folosit cunoștințe dobândite pe parcursul anilor de facultate, în cadrul mai multor materii, și anume:

- **Introducere în .NET:** limbajul C# și principii generale
- **Baze de date:** realizarea și managementul baze de date
- **Ingineria programării:** realizarea diagramelor și optimizarea modului de lucru
- **Rețele de calculatoare:** protocolul HTTP
- **Tehnologii web:** realizarea unei aplicații web și principii REST
- **Programare orientată obiect:** principiile programării orientate obiect

2. Descrierea problemei

În vremurile în care s-au pus bazele primelor universități din lume, în jurul anului 1100, profesorii erau forțați de către asociațiile studențești, formate în urma unor discuții cu cei aflați la conducerea orașelor, să respecte o serie de reguli, un exemplu fiind să nu lipsească de la cursuri fără aprobarea studenților. Aceștia trebuiau să se supună regulilor deoarece singurele lor venituri erau banii pe care studenții și familiile lor îi investeau în învățământ. Simțindu-se excluși din procesul de construire al acestor reguli, profesorii au început să impună de asemenea anumite reguli care să fie aplicate studenților, una dintre acestea fiind elaborarea unei lucrări de licență la finalul studiilor și prezentarea ei în fața unei comisii.

Astfel, în scurt timp, mediul academic universitar a prins o formă asemănătoare cu cea din zilele noastre, necesitatea dezvoltării unei lucrări de licență datând, deci de acum aproximativ 900 de ani iar importanța acesteia rămânând aceeași.

În mod convențional, procesul de elaborare a lucrării de licență și, în cazul în care este necesar, a unei componente practice care să stea la baza acestei lucrări se realizează pe parcursul mai multor luni sub supravegherea unui coordonator, urmărind îndeaproape sfaturile acestuia. În acest sens au loc întâlniri periodice între cele două părți implicate în proces. Aceste perioade pot varia în funcție de student sau coordonator și se stabilesc într-un interval din zi în care ambii participanți sunt prezenți în facultate și au la dispoziție suficient timp pentru a discuta eventualele nelămuriri sau următorii pași ce trebuie urmați.

De cele mai multe ori, profesorul coordonator nu dispune de suficient timp în cadrul perioadei în care este la facultate pentru a oferi sprijin maxim studenților pe care îi îndrumă, iar studenții, din cauza examenelor, testelor sau a locului de muncă, nu reușesc mereu să ajungă cu toate nelămuririle către cadrul didactic aferent lor și, astfel informație prețioasă este pierdută și nevalorificată la adevăratul ei potențial.

3. Abordări anterioare

În acest moment nu există o aplicație folosită pe scară largă și care să poată să îmbunătățească acest proces iar parcursul urmat de către studenți depinde de fiecare coordonator științific, acesta alegându-și metoda prin care comunică aspecte cum ar fi: următoarea întâlnire pentru o nouă discuție, conținutul ce trebuie modificat, corectat sau eventual îmbunătățit, sfaturi pentru a-i ușura munca și pașii pe care trebuie să îi parcurgă până la următoarea lor discuție. Există totuși un site care găzduiește o platformă numită **EduSoft**¹³, aceasta fiind o asociație de tip ONG coordonată de către un specialist, sub îndrumarea căruia studenții pot alege să își elaboreze lucrarea de licență. Neajunsul acestei platforme este acela că numărul studenților care au folosit-o este extrem de redus iar din anul 2015 nu mai pare să existe activitate.

Datorită faptului că internetul și tehnologia, în general, au avut o evoluție foarte mare în ultimul timp, ele pot fi folosite cu succes pentru a implementa și, mai apoi, utiliza orice tip de aplicație menită să ne facă viața mai confortabilă și să ne ofere acces la tot felul de informații.

Am ales deci să construiesc un mod de a le ușura munca atât coordonatorilor cât și studenților prin a reduce cantitatea de timp pierdută prin aglomerația orașelor, prin așteptarea reciprocă pentru a avea o întâlnire dacă nu consideră că este neapărat necesară, prin discuțiile care pot divaga de la subiect și prin neînțelegerile ce pot apărea cu privire la anumite secțiuni din lucrare din cauza unor eventuale exprimări neclare. Aplicația este adresată studenților ciclului de licență și coordonatorilor acestora, venind în ajutorul lor cu tot felul de funcționalități gândite pentru a le ușura în special modul de manageriere a timpului și pentru a asigura o mai bună comunicare între cei doi, fiecare dintre aceștia putând actualiza datele și/sau fișierele în orice moment le permite timpul.

¹³ EduSoft: <https://www.edusoft.ro/>

4. Descrierea soluției

4.1 Principalele funcționalități

Prima pagină cu care utilizatorul va avea contact atunci când va dori să utilizeze aplicația curentă va fi cea care îi va permite logarea. Login. Aceasta se va realiza cu ajutorul unui email, ce aparține Facultății de Informatică Iași, și o parolă. În cazul în care credențialele utilizate nu se regăsesc în sistem, înseamnă că acel cont nu există sau că, din greșeală, au fost introduse credențiale invalide. Utilizatorul va fi notificat de acest lucru, dacă va fi cazul, și i se va permite să reintroducă un nou set de credențiale. Această pagină conține validări asupra adresei de email, mai exact aceasta să aparțină Facultății de Informatică și să fie asignată unui student existent. Email-ul va putea fi introdus atât cu majuscule cât și cu minuscule, sistemul gestionând astfel de situații prin compararea șirurilor de caractere convertite în minuscule.

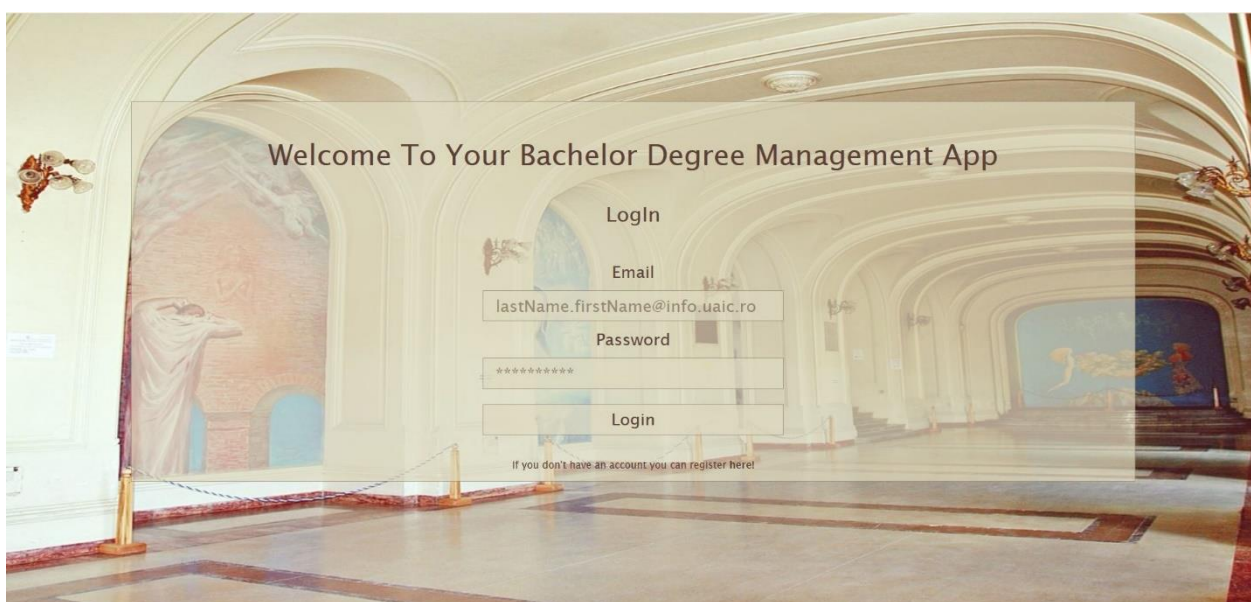


Figura : Pagină ce permite accesarea sistemului

În cazul în care contul pe care utilizatorul dorește să îl folosească nu există încă, situație explicată mai sus, acesta va putea să se înregistreze în cadrul sistemului și mai apoi să se logheze, existența noului cont fiind asigurată în caz de succes sau semnalată o eroare în caz contrar. Un utilizator nu va putea să se înregistreze cu ajutorul unei adrese de email care a mai fost folosită în prealabil de către alt cont. Register. Doar utilizatorii cu titlu de student își vor putea crea un cont

nou, pentru utilizatorii de tip profesor existând un administrator care se ocupă de crearea lor. Validările existente pe această pagină sunt atât referitoare la adresa de email cât și la parola introdusă. Adresa de email va trebui să fie una ce aparține Facultății de Informatică, să fie asignată unui student care își desfășoară în prezent studiile aici și să nu mai fi fost folosită la crearea niciunui alt cont în cadrul acestui sistem. Parola va trebui să aibă între șase și treizeci și două de caractere și să aibă în componență cel puțin o cifră, un caracter special, o literă mare și una mică. Orice fel de combinații care respectă cerințele anterioare vor fi acceptate. O dată introdusă o parolă validă, aceasta va trebui confirmată prin reintroducerea ei într-un câmp apropiat. Dacă oricare din validările adresei de email sau parolei vor fi încălcate, utilizatorul primi o notificare pentru a deveni conștient de acest lucru.

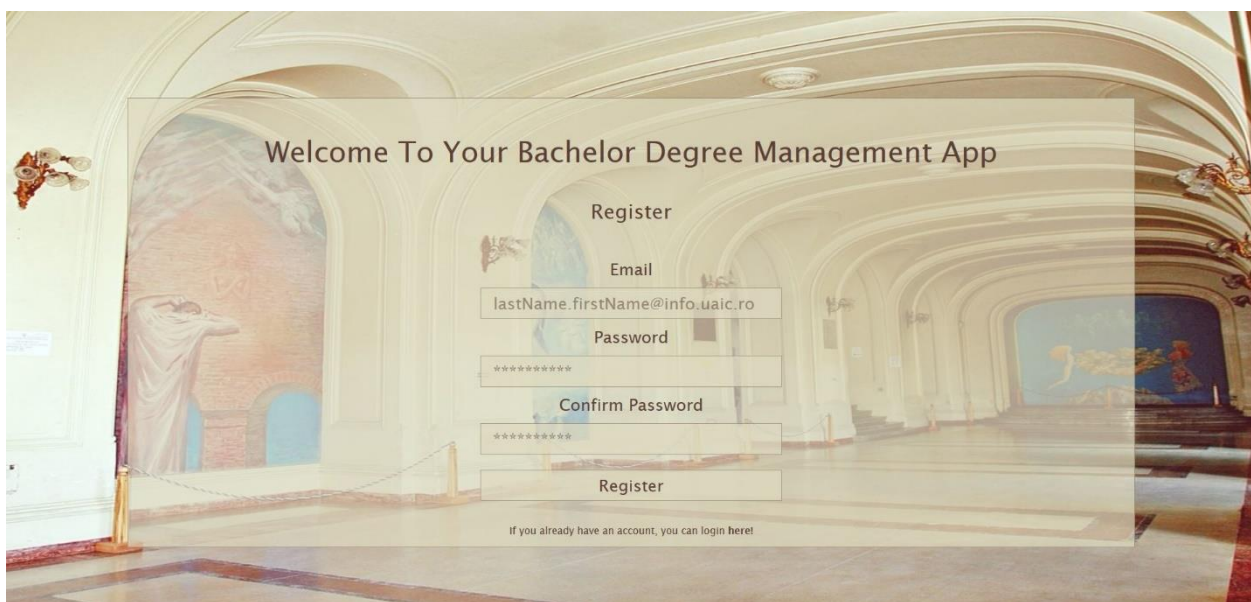


Figura : Pagină ce permite înregistrarea în sistem

Dacă logarea sau înregistrarea unui cont nou a reușit utilizatorii vor fi avea acces, în cazul în care vor dori, la pagina de bun venit, din cadrul căreia vor putea alege care să fie următoarele lor acțiuni. Figura. Acțiunile care vor urma pot fi: navigarea spre pagina în care profesorul postează diferite, naviarea spre pagina de profil a utilizatorului, navigarea către pagina care oferă informații despre munca fiecărui student, navigarea către pagina care oferă informații utile despre sesiune curentă de susținere a licenței (pagini accesibile atât pentru student cât și pentru profesor dar nu pentru administrator) și navigarea către pagina unde utilizatorul vede cererile de înscriere sau de întâlnire față în față (pagină accesibilă pentru profesor dar nu pentru student sau

administrator). Pentru utilizatorii de tip student, această pagină va deveni disponibilă abia după ce perioada specifică înscrierilor va trece și ei vor fi asignați către unul dintre coordonatorii disponibili.

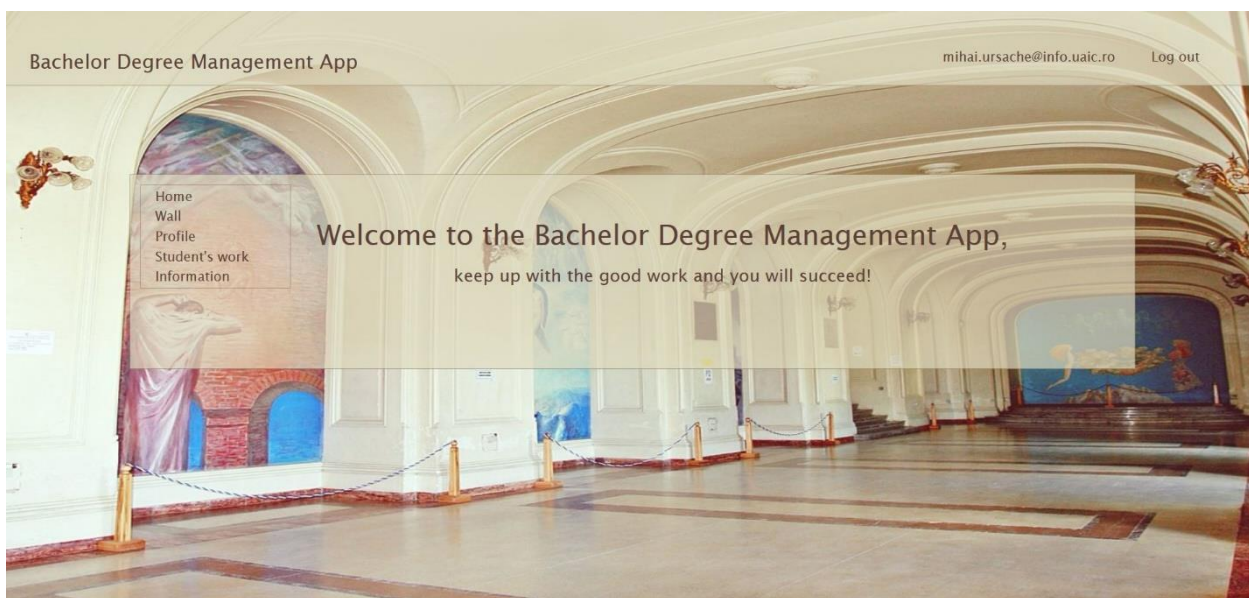


Figura : Pagină de “Bun venit”

O pagină similară cu ce de bun venit este cea de eroare, ce va apărea în cazul în care, din diverse motive, serverul nu va fi disponibil.

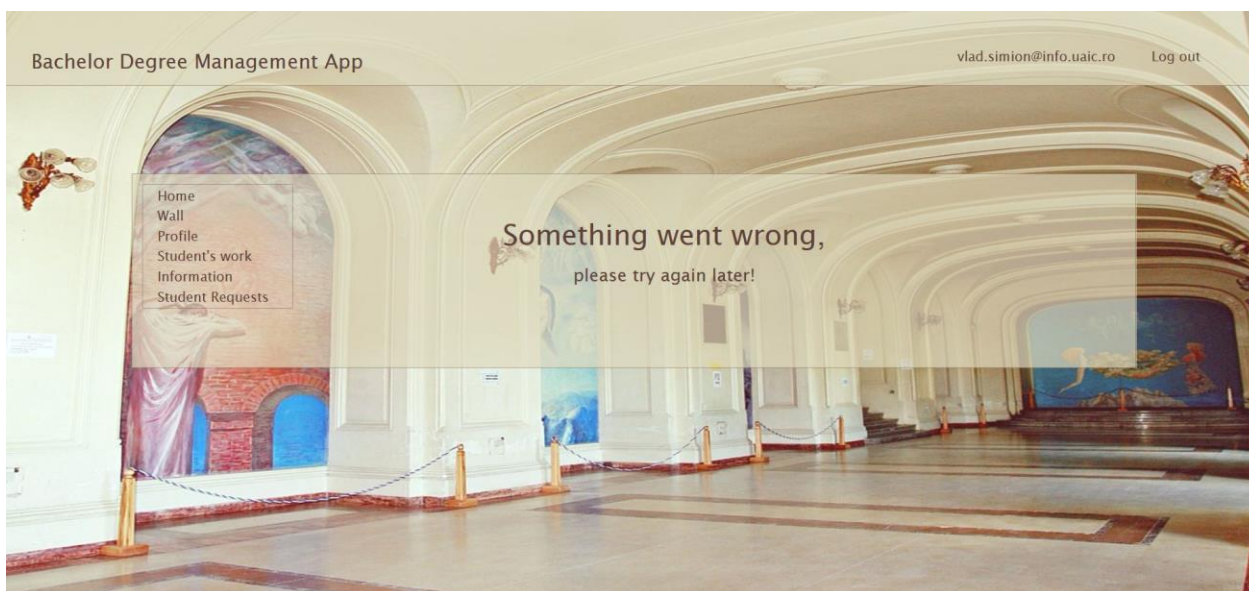


Figura : Pagină de eroare

Inițial, pentru toată durata perioadei de înscrieri, studentul va ajunge pe o pagină care îi va permite să își aleagă un profesor coordonator dintr-o listă completată de către cadrele didactice. Va putea analiza opțiunile disponibile și, în funcție de aptitudinile și dorințele lui, va putea continua procesul de selecție apăsând butonul vizibil (ca fiind inactiv) în *Figura*. Dacă studentul aplică la unul dintre coordonatorii listați, va fi redirecționat spre o pagină pe care va adăuga anumite detalii și, mai apoi, reîntors pe aceeași pagină destinată înscrierilor dar cu opțiunea de aplicare blocată până când cadrul didactic îi va aproba sau respinge cererea de înscriere. De asemenea, înregistrarea corespondentă profesorului la care a aplicat va fi evidențiată.

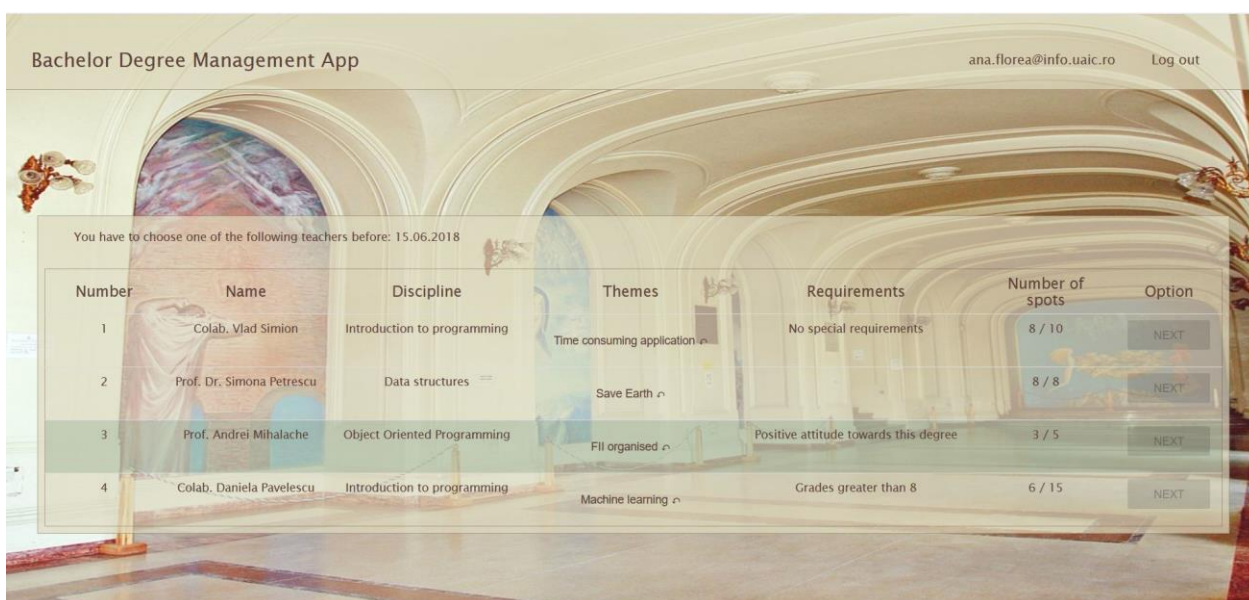


Figura : Pagină pentru înscrierea la un coordonator, cu opțiune blocată

Pentru utilizatorii de tip profesor, pentru aceeași perioadă a înscrierilor, va fi afișată lista cererilor studenților înscriși până în momentul respectiv. Aceștia vor putea vizualiza anumite detalii furnizate de către student, pe baza cărora vor decide dacă aprobă sau nu cererea studentului. Aceștia vor avea în fiecare moment la dispoziție informații despre numărul de studenți pe care îi mai poate accepta și totalul numărului de studenți pe care vrea să îi accepte în sesiunea curentă, informație ce acesta o va completa în prealabil. După ce perioada înscrierilor trece, lista cererilor de înscriere va fi înlocuită de lista cererilor de întâlniri în persoană pe care

studenții le vor putea solicita, care va conține adresa de email a studentului, data și ora la care a făcut cererea și statusul curent al cererii (în așteptare, aprobat sau refuzat). *Figura*

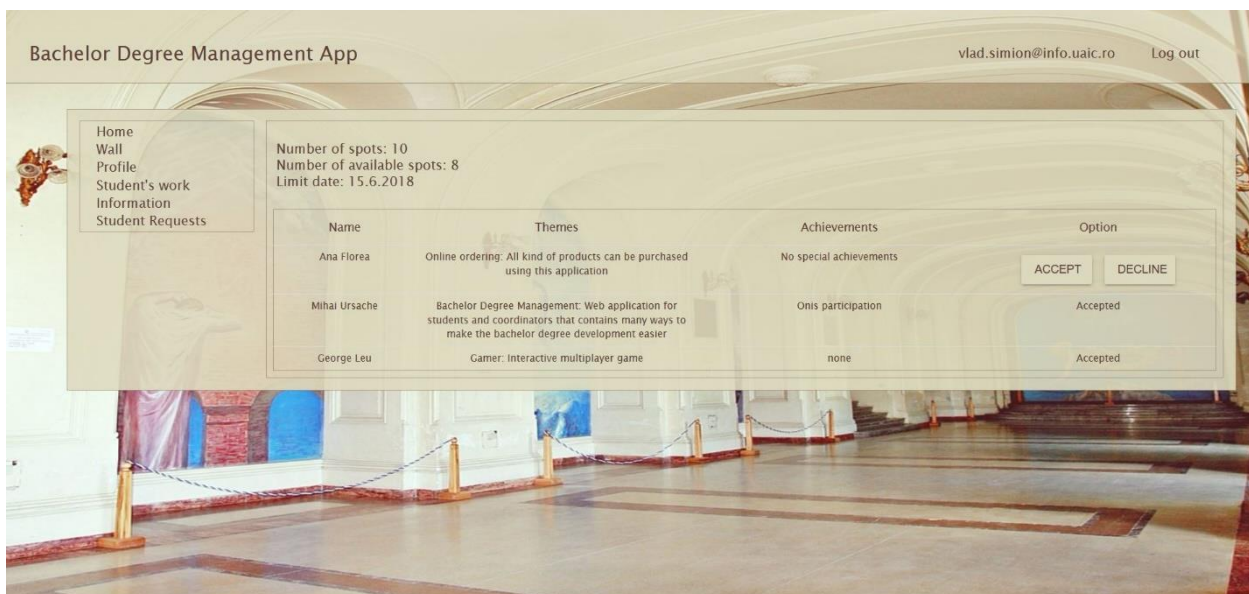


Figura : Pagină pentru vizualizarea cererilor studenților în timpul perioadei de înscriere

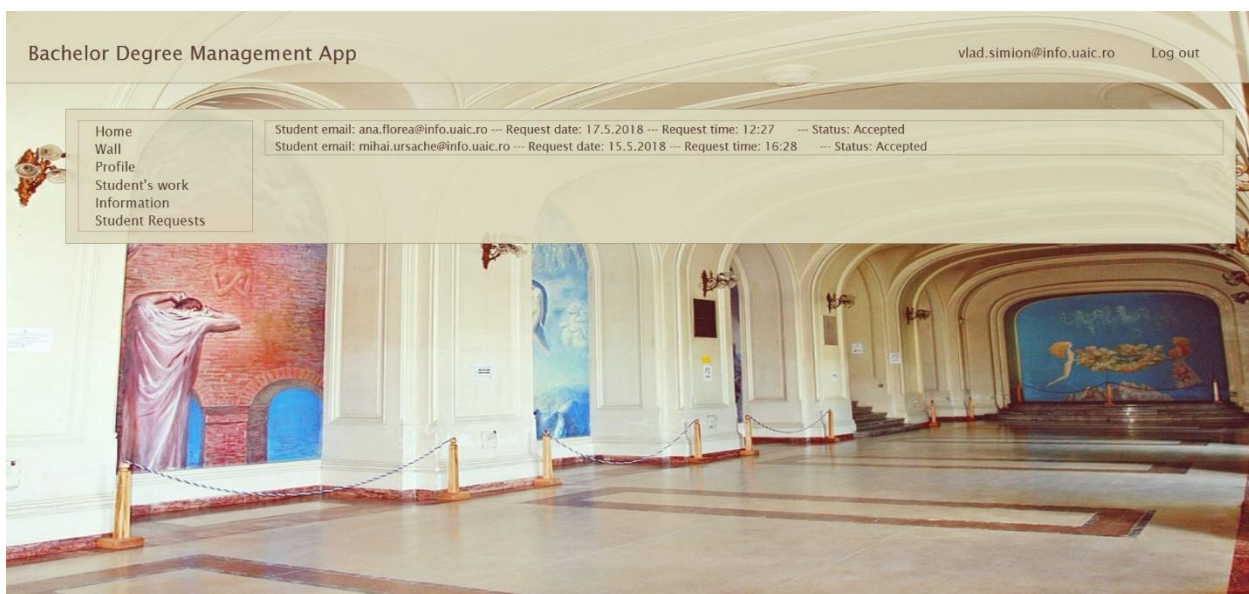
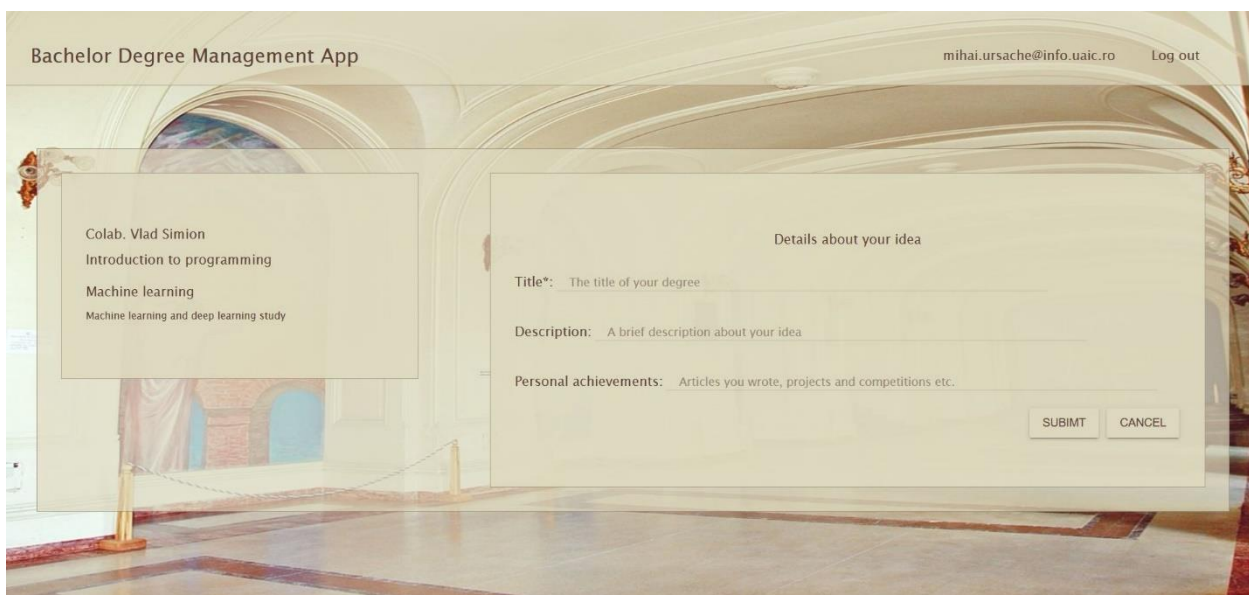


Figura : Pagină pentru vizualizarea cererilor studenților după perioada de înscriere

Așa cum menționam anterior, după ce un student selectează unul din domnii profesori coordonatori disponibili, va fi redirecționat către o pagină în cadrul căreia va putea oferi mai multe detalii despre ideea pe care o are. *Figura*. El va completa date despre titlul lucrării, o mică

descriere și, dacă va dori, va menționa performanțe obținute în cadrul facultății. Acestea din urmă ar putea însemna impresionarea profesorului coordonator ales. În partea stânga va avea disponibile informații despre profesorul coordonator la care studentul dorește să se înscrie.



The screenshot displays the 'Bachelor Degree Management App' interface. At the top, the title 'Bachelor Degree Management App' is on the left, and the email 'mihai.ursache@info.uaic.ro' and a 'Log out' link are on the right. The background is a blurred image of a grand hall. On the left, a sidebar lists the user's name 'Colab. Vlad Simion' and four course topics: 'Introduction to programming', 'Machine learning', and 'Machine learning and deep learning study'. The main area is titled 'Details about your idea' and contains three text input fields: 'Title*' (with a placeholder 'The title of your degree'), 'Description' (with a placeholder 'A brief description about your idea'), and 'Personal achievements' (with a placeholder 'Articles you wrote, projects and competitions etc.'). At the bottom right of the form are two buttons: 'SUBMIT' and 'CANCEL'.

Figura : Pagină pentru adăugarea detaliilor necesare lucrării de licență

Atunci când un cadru didactic va fi adăugat în sistem de către administrator, i se vor atribui numai un set prestabilit de informații, profesorul trebuind, doar în cazul în care vor accesa sistemul (se vor loga) pentru prima dată, să adauge mai multe detalii despre disponibilitatea și cerințele lui, așa cum este vizibil în *Figura*. Unele din aceste date vor putea fi editate ulterior dacă se va dori. Datele editabile sunt cele care corespund zilei și intervalului din zi în care domnii profesori coordonatori sunt disponibili pentru a avea întâlniri în persoană cu studenții. Tot profesorul coordonator va trebui să stabilească un număr maxim de studenți pe care îi va coordona (acesta fiind limitat superior de un număr hotărât de conducerea facultății și introdus în sistem de către administrator), niște cerințe minime pe care dorește ca studenții să le îndeplinească (dacă are astfel de cerințe și dorește să le facă publice), un titlu și o descriere pentru o lucrare de licență astfel încât studenții își vor putea da seama care este gradul de dificultate și care sunt așteptările pe care un profesor le are și o zi din săptămână, respectiv un interval orar din acea zi în care dânsul va fi disponibil pentru întâlniri în persoană cu studenții, în caz că aceștia solicită acest lucru.

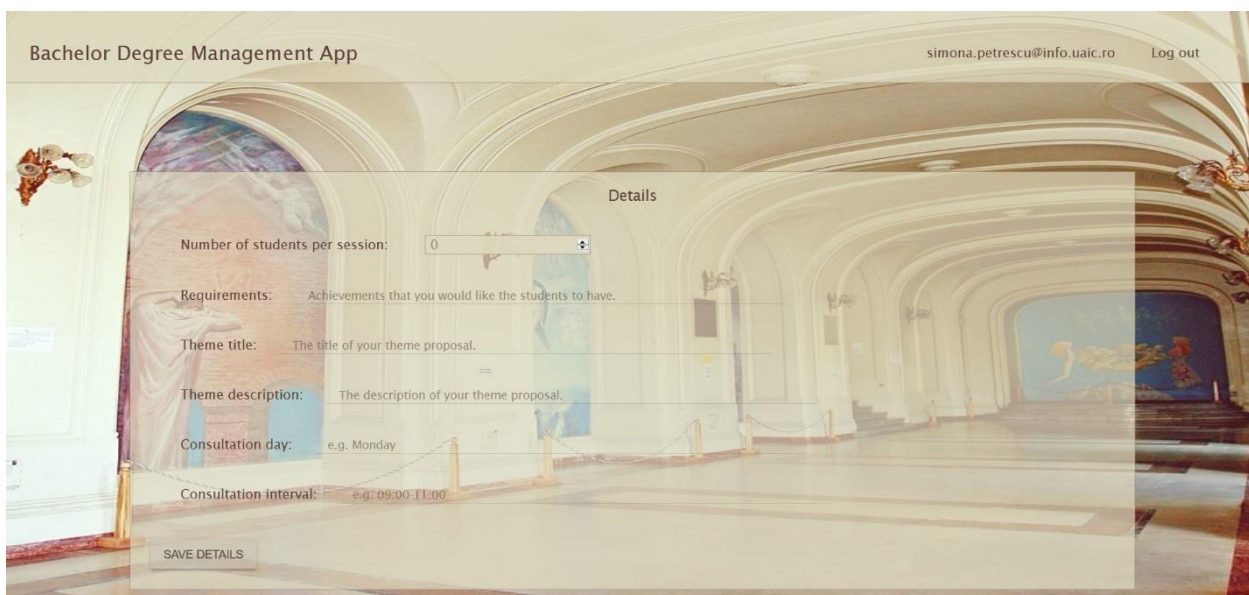


Figura : Pagină pentru adăugarea detaliilor și cerințelor unui cadru didactic

După ce perioada destinată înscrierilor va trece, utilizatorii vor avea acces la o pagină ce conține majoritatea informațiilor pe care un student le-ar dori. Vor avea legături către toate celelalte paginile, vor putea adăuga/anula o cerere pentru o întâlnire în persoană cu coordonatorul, vor putea vizualiza toate noutățile coordonatorului și iniția postări sau adăuga comentarii la postările deja existente. Profesorii coordonatori vor avea la dispoziție aceleași informații, mai puțin posibilitatea de a adăuga/anula o cerere pentru o întâlnire în persoană.

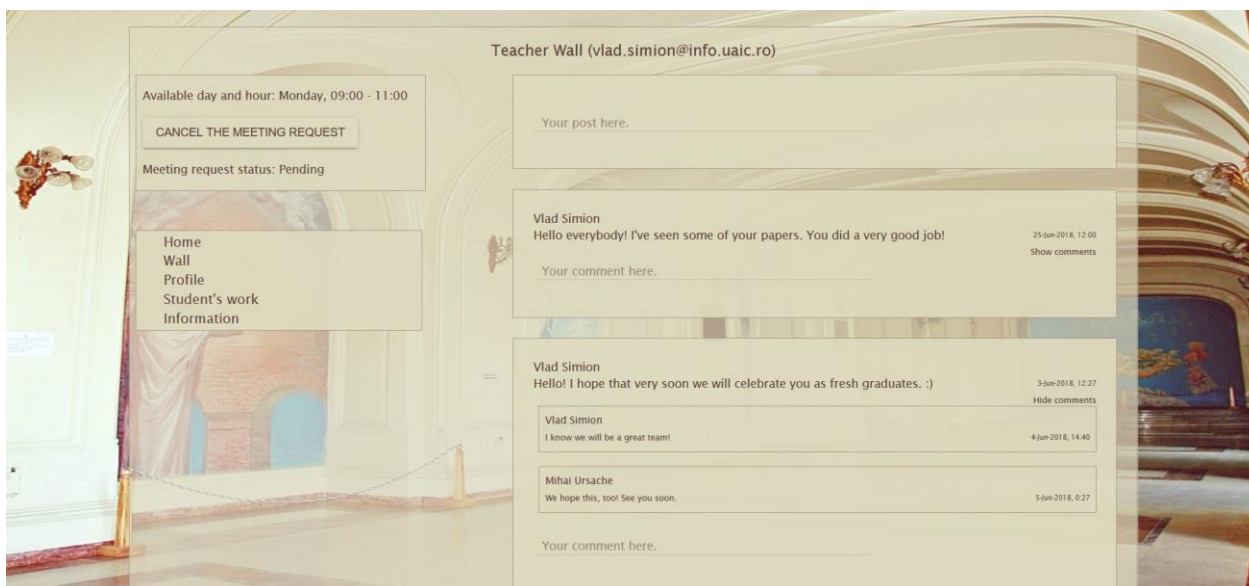


Figura : Pagină pentru adăugarea vizualizare detaliilor și cerințelor unui cadru didactic

Atât profesorul coordonator, cât și studenții vor avea acces la o serie de informații despre toți candidații înscriși la același profesor. Aceste detalii sunt: numele, performanțele din cadrul facultății, adresa la care codul sursă este disponibil și ultimele 5 actualizări făcute de fiecare.

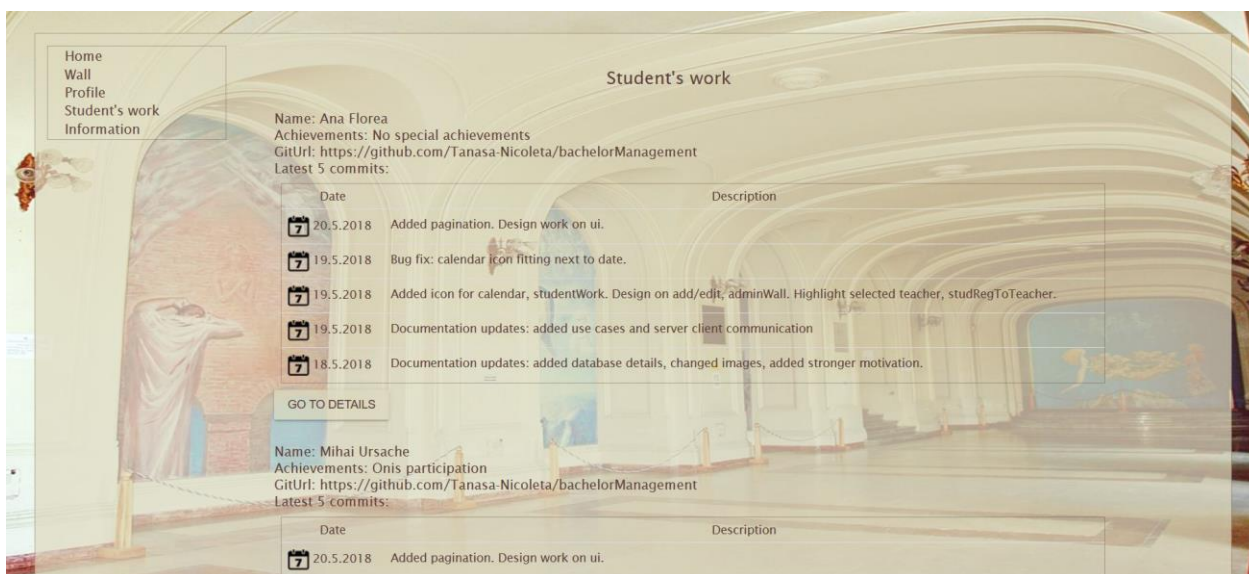


Figura : Pagină pentru vizualizare detaliilor fiecărui student

Pentru fiecare student care apare pe pagina menționată anterior, se va putea naviga spre o pagină specifică acestuia în care vor fi prezentate informații mult mai detaliate cu privire la munca pe care acesta a demarat-o pe parcursul ultimelor luni. Se vor putea vedea informații despre codul sursă, cum ar fi: adresa la care acesta poate fi accesat și vizualizat, o diagramă care va expune procentajul limbajelor de programare folosite (diagramă de tip disc cu ajutorul căreia informațiile se percep foarte ușor și care folosește culori calde pentru a oferi utilizatorului o experiență plăcută), o diagramă ce va expune progresul, măsurat ca și număr de linii de cod șterse și adăugate în ultimele 10 săptămâni (scara acestei diagrame este 1:100, ceea ce se traduce ca înmulțirea cu 100 a tuturor valorilor situate pe diagramă, culorile folosite sunt roșu pentru numărul de linii șterse respectiv verde pentru numărul de linii adăugate, acestea intercalându-se când este cazul și oferind utilizatorului o vedere de ansamblu exactă și rapidă) și ultimele 30 de actualizări pe care le-a făcut în dezvoltarea lucrării sale de licență (acestea fiind organizate în grupuri de câte cinci, disponibile în acest format cu ajutorul paginării ce are opțiuni pentru pagina anterioară, pagina următoare și trei dintre pagini disponibile pentru a fi selectate).

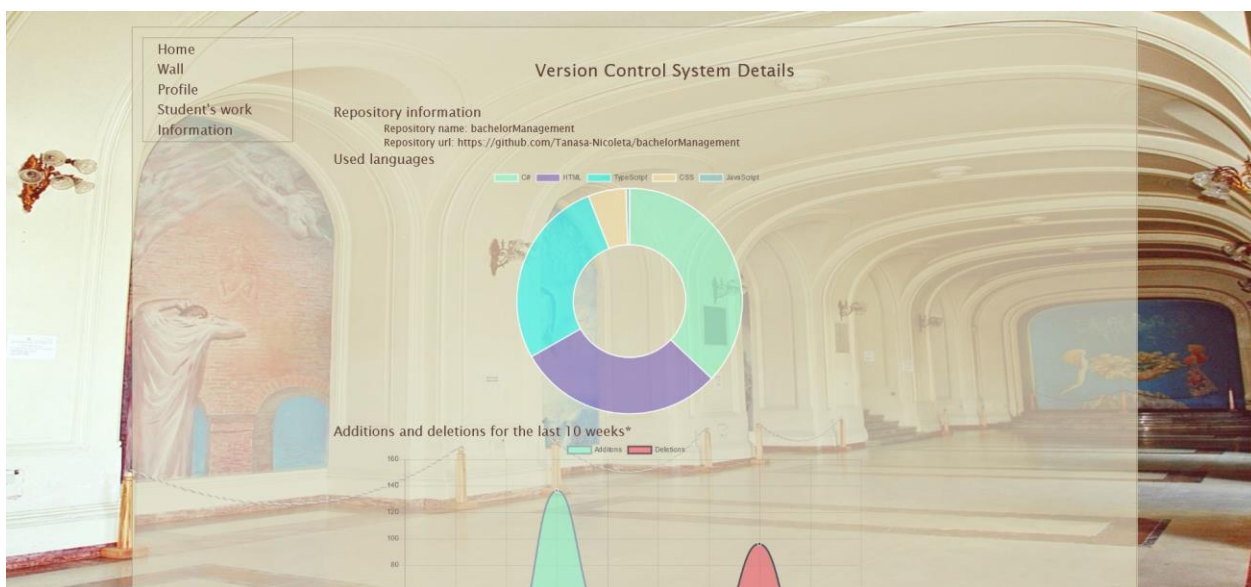


Figura : Pagină pentru vizualizare detaliilor unui student ales



Figura : Pagină pentru vizualizare detaliilor unui student ales – paginare

Fiecare utilizator, exceptând administratorul sistemul, va avea acces la o pagină cu informații specifice profilului său. Studenții vor avea acces la informații de tipul: nume, numele profesorului coordonator asignat, numele temei lucrării de licență, descrierea temei respective, adresa la care aceasta poate fi găsită, numărul matricol, anul în care studentul a început studiile în cadrul facultății și mediile lui.

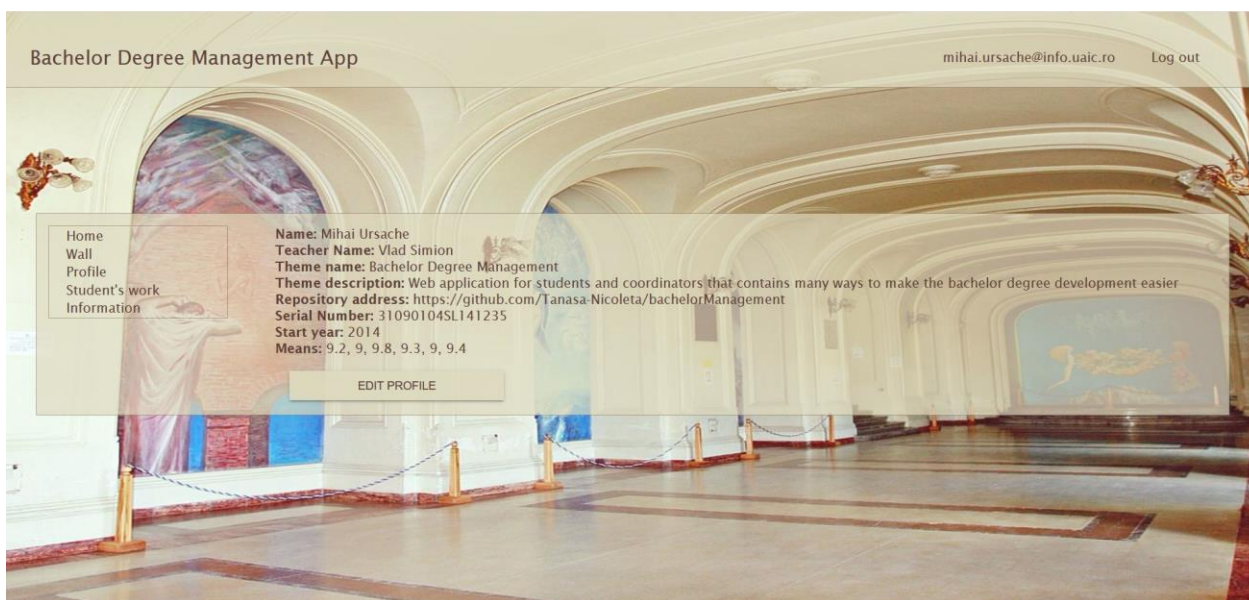


Figura : Pagină pentru vizualizarea profilului unui utilizator de tip student

De asemenea, utilizatorului de tip student i se va permite editare unei anumite selecții de informații cum ar fi: adresa la care se găsește codul său sursă, titlul lucrării pe care o dezvoltă și descrierea acestei lucrări.

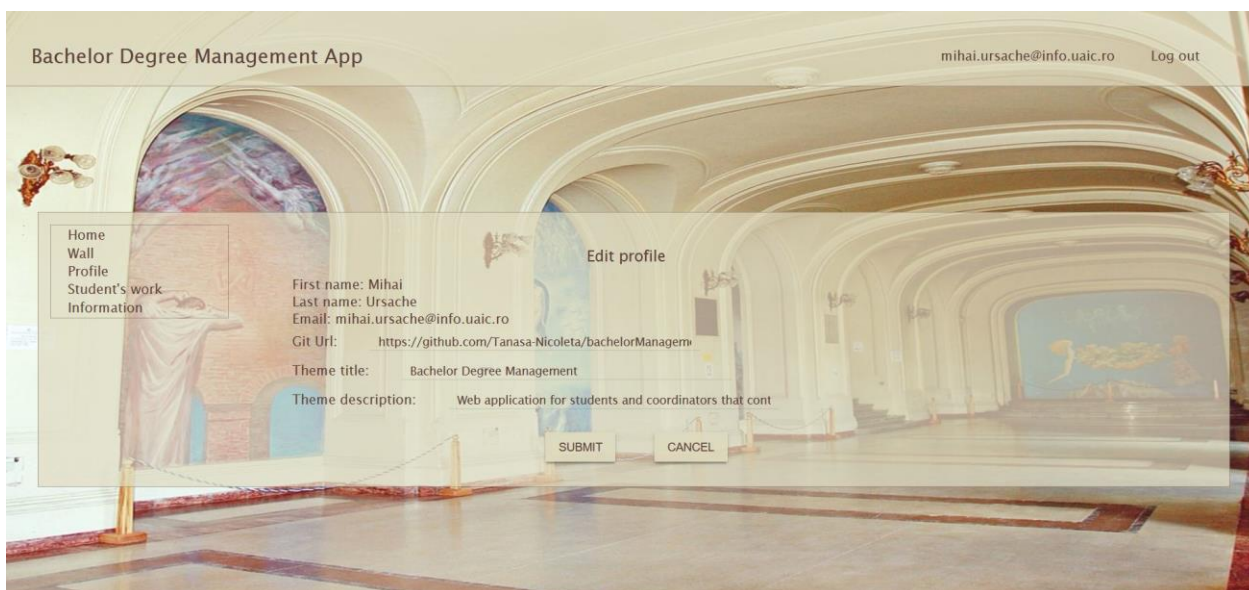


Figura : Pagină pentru editarea profilului unui utilizator de tip student

La fel ca în cazul studenților, și cadrele didactice vor avea acces la o pagină de tip profil în care vor avea informații despre numele și titlul lor, cerințele pe care le au de la studenți, ziua și ora în care au disponibilitate pentru întâlniri în persoană cu studenții, numele și descrierea temei propuse de către ei, numele, titlul, descrierea și adresa lucrării de licență a fiecărui student înscris spre coordonarea de către acesta. Ei vor putea edita numai informațiile care privesc ziua și ora disponibilă pentru întâlniri cu studenții.

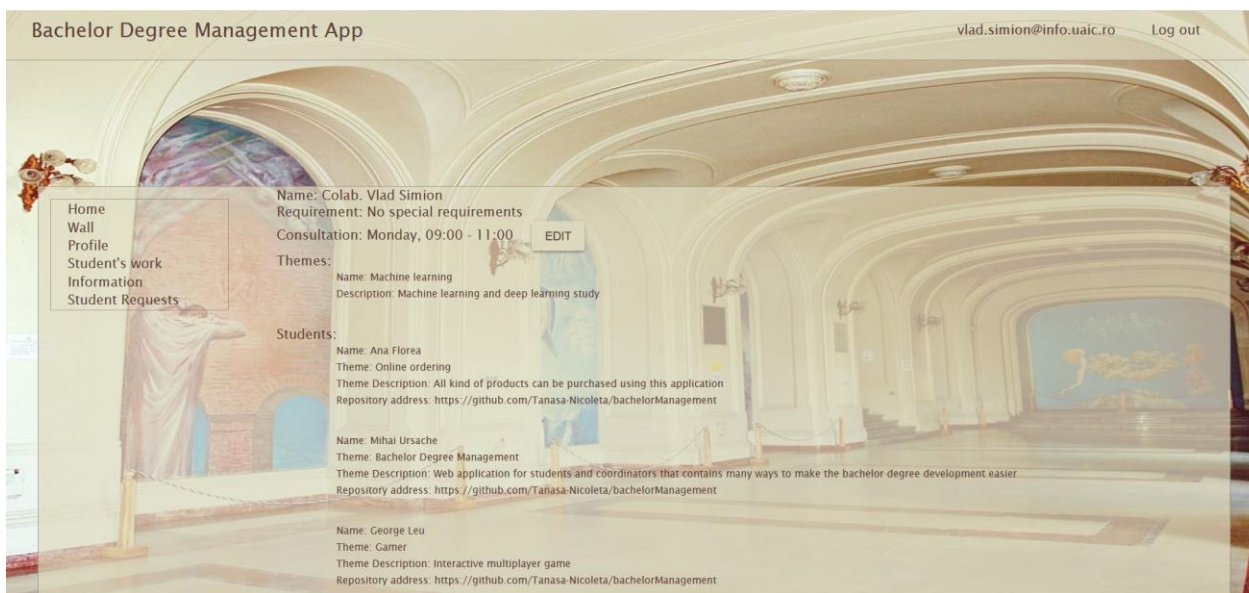


Figura : Pagină pentru vizualizarea profilului unui utilizator de tip profesor

Utilizatorii de tip profesor și cei de tip student vor avea, de asemenea, acces la o pagină care va oferi informații utile. Acestea vor fi despre perioadele de înscriere, de evaluare și de susținere a prezentării propriu-zise a lucrării de licență și documentele necesare pentru înscrierea în cadrul sesiunii curente. Vor putea vizualiza și câteva sfaturi de redactare și prezentare, menite să le ușureze tot traseul de pregătire a lucrării de licență, ca parte practică dar și ca parte teoretică. Aceste informații vor fi indentice cu cele care sunt postate pe pagina oficială a Facultății de Informatică din Iași.

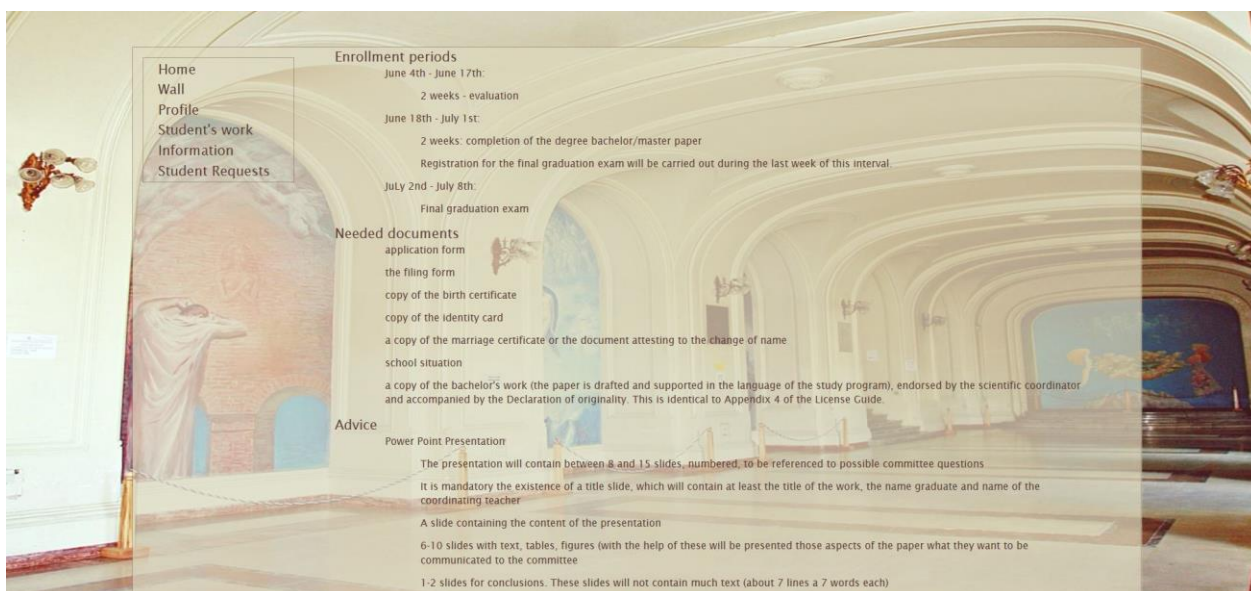


Figura : Pagină pentru vizualizarea informațiilor disponibile pentru sesiunea curentă

Un al treilea rol existent în cadrul aplicației este, după cum s-a menționat anterior, cel de administrator de sistem. Responsabilitățile acestuia vor fi de a menține lista de profesori mereu actualizată prin adăugarea, respectiv ștergerea conturilor cadrelor didactice, după caz. Va exista un singur cont de administrator, acesta putând fi împărțit de mai multe persoane, dacă se va dori.

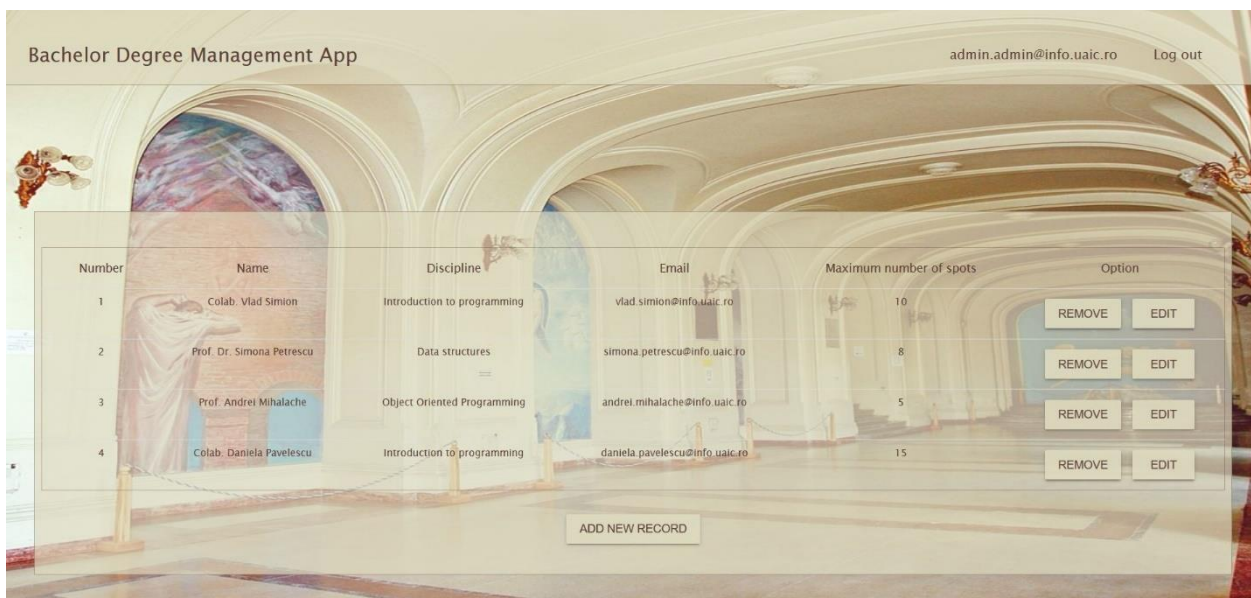


Figura : Pagină vizualizată de către administrator la accesul în sistem

Pentru a adăuga o nouă înregistrare de tip profesor coordonator, administratorul trebuie să dețină, în prealabil, informații despre numele, prenumele, adresa de email, disciplina pe care o predă și gradul profesorului. El va adăuga, de asemenea, nu număr maxim de studenți pe care un profesor îi va putea accepta în sesiunea curentă. Acest număr reprezintă limita superioară la care poate ajunge un profesor cu numărul de studenți coordonați. În cazul în care, din diferite motive, administratorul nu vrea să finalizeze operația tocmai începută (și anume cea de a insera un cadru didactic nou în sistem) va avea la dispoziție un buton ce îi va permite să o anuleze.

Number	Name	Discipline	Email	Maximum number of spots	Option
1	Colab. Vlad Simion	Introduction to programming	vlad.simion@info.uaic.ro	10	REMOVE EDIT
2	Prof. Dr. Simona Petrescu	Data structures	simona.petrescu@info.uaic.ro	8	REMOVE EDIT
3	Prof. Andrei Mihalache	Object Oriented Programming	andrei.mihalache@info.uaic.ro	5	REMOVE EDIT
4	Colab. Daniela Pavelescu	Introduction to programming	daniela.pavelescu@info.uaic.ro	15	REMOVE EDIT

Add new record

First name:

Last name:

Email:

Discipline:

Job title:

Maximum number of students:

SUBMIT CANCEL

Figura : Pagină care permite adăugarea unei noi înregistrări de tip profesor, disponibilă administratorului de sistem

Dacă va alege opțiunea de a edita o înregistrare de tip profesor, administratorului de sistem îi va fi permisă actualizarea doar a unei serii de date, adică modificarea numelui, prenumelui, disciplinei predate, gradului didactic și a numărului de studenți aferent sesiunii curente. La fel ca și în cazul adăugării unei înregistrări noi, există posibilitatea opririi operației de actualizare care este în progres dacă, din diferite motive, se dorește acest lucru. Atât adăugarea cât și actualizarea datelor au loc nominal (pentru un singur profesor la un moment dat) iar schimbările efectuate vor fi imediat vizibile pentru administrator.

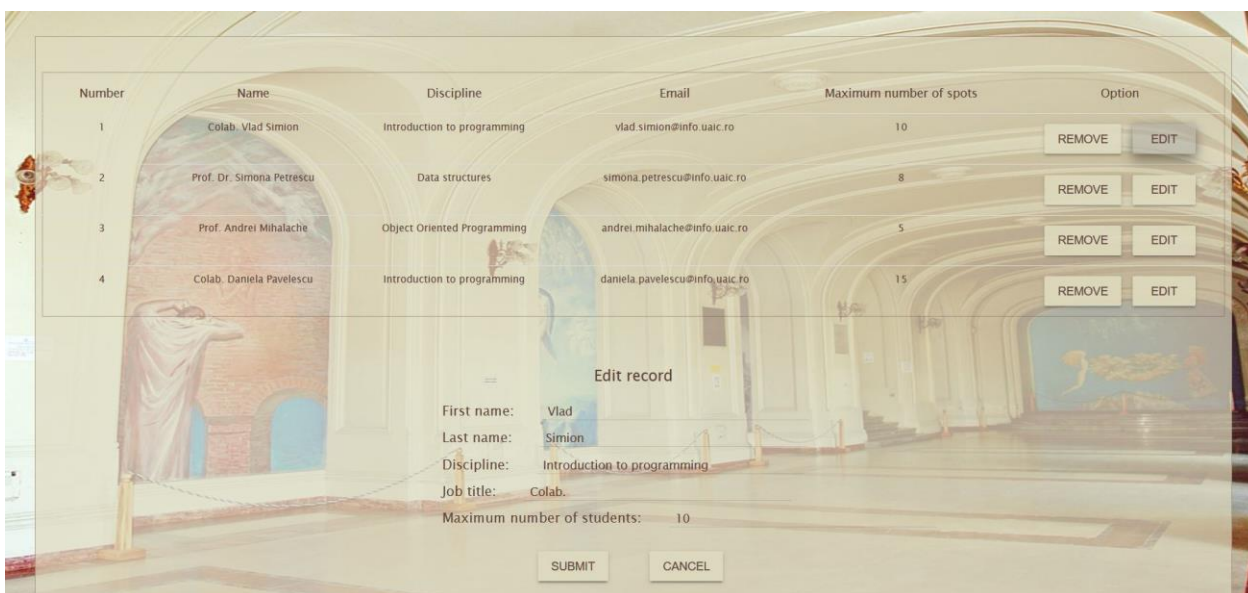


Figura : Pagină care permite editarea unei înregistrări de tip profesor, disponibilă administratorului de sistem

4.2 Diagrame

4.2.1 Diagrame pentru cazuri de utilizare

Fiecare utilizator trebuie să fie înregistrat în sistem înainte de a putea face orice fel de acțiune. După cum menționam anterior, în cadrul aplicației există definite trei tipuri de utilizatori: student, profesor coordonator și administrator de sistem. Fiecare dintre aceștia pot demara mai multe tipuri de acțiuni astfel interogând sau modificând baza de date asociată aplicației.

Toate cele trei tipuri de utilizatori pot vizualiza diferite tipuri de liste și pot acționa asupra elementelor acestora (adăugare, ștergere, editare, acceptare, refuzare) iar utilizatorii de tip student sau profesor coordonator pot să vizualizeze și editeze profilul personal și doar să îl vizualizeze pe cel al celorlalți participanți.

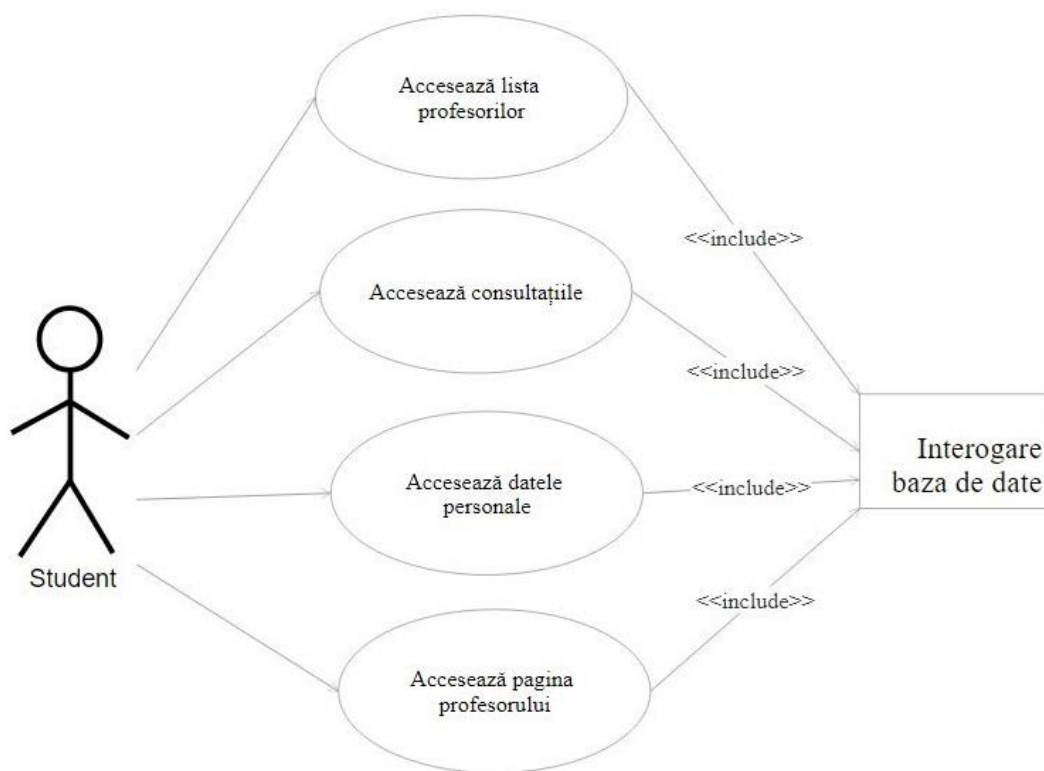


Figura : Acțiunile care generează interogarea bazei de date pentru studenți

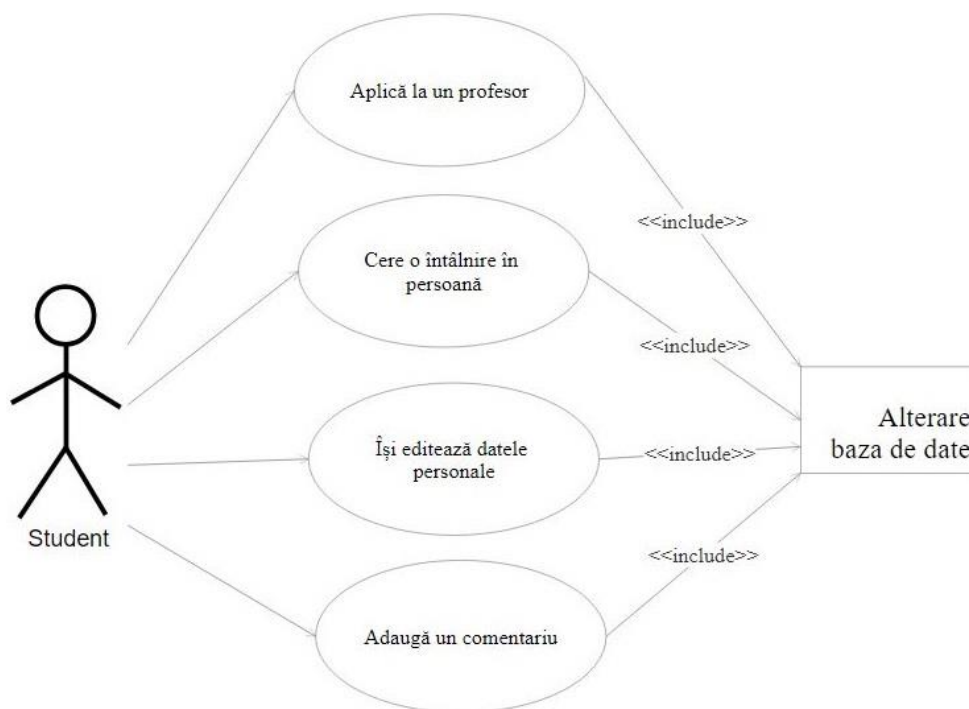


Figura : Acțiunile care generează alterarea bazei de date pentru studenți

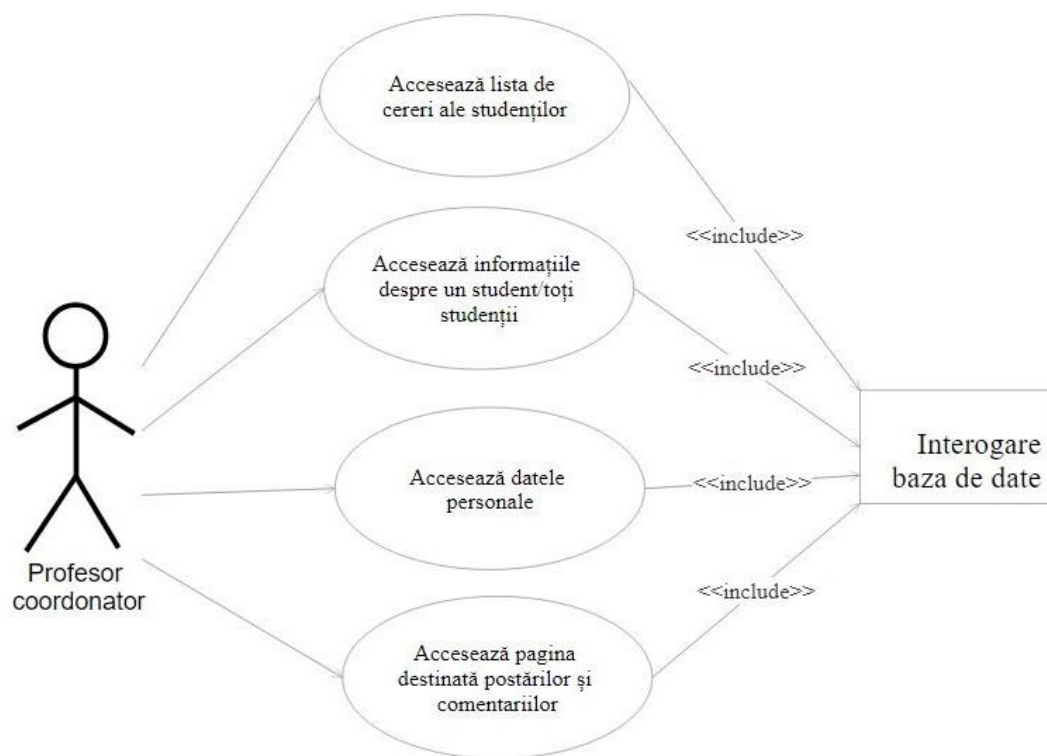


Figura : Acțiunile care generează interogarea bazei de date pentru profesori

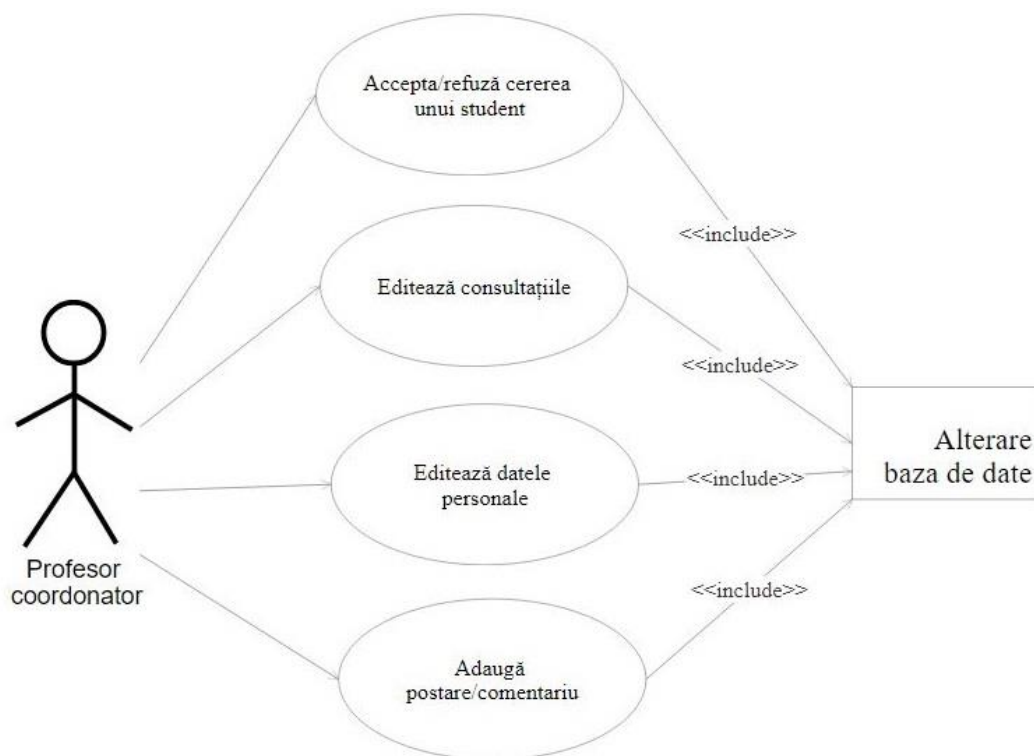


Figura : Acțiunile care generează alterarea bazei de date pentru profesori



Figura : Acțiunile care generează interogarea/alterarea bazei de date pentru administrator

4.2.2 Diagrama structurii soluției

Soluția este structurată pe mai multe nivele asigurându-se astfel, un mod optim de a se face managementul și mentenanța codului sursă. Fiecare nivel este reprezentat de un proiect care, la rândul lui este împărțit în dosare și subdosare pentru a permite localizarea ușoară a fișierelor.

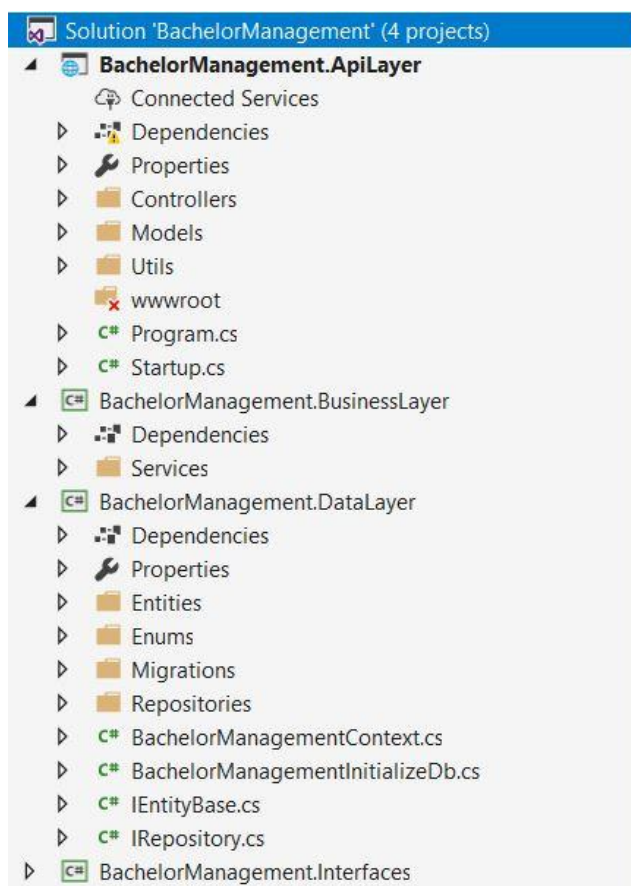


Figura : Structura soluției, Visual Studio

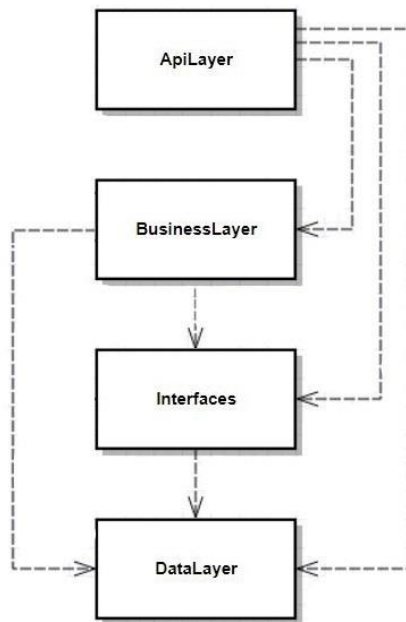


Figura : Structura soluției – proiecte și dependențe

ApiLayer – acesta este nivelul destinat API-ului REST construit. Acesta conține mai multe Controllere (clase ce expun servicii către partea de client a aplicației și care pot fi accesate cu ajutorul protocolului HTTP), mai multe modele (clase care descriu cum ar trebui să arate obiectele pe care serviciile le primesc de la partea client sau le trimit către partea client) și o clasă (în cadrul dosarului Utils) cu ajutorul căreia se verifică faptul că formatul token-ului primit de la utilizatori este corect acest lucru eficientizând timpul de răspuns (această verificare se face imediat ce o cerere ajunge către server și în caz că nu sunt îndeplinite condițiile necesare se trimite imediat un răspuns corespunzător către client, fără a se mai executa cererea în sine).

BusinessLayer – acesta este nivelul destinat logicii ce are loc pentru a face răspunsul la cererea clientului posibil. Conține mai multe servicii, împărțite pe entități (cont, student, profesor, comentariu etc.) ce au responsabilitatea de a merge către locul de stocare a datelor și a efectua operații de interogare, adăugare, ștergere sau editare a acestora.

Interfaces – este nivelul care face, într-o anumită măsură, legătura dintre nivelul de date și nivelul de logică. Acesta conține doar interfețe ce descriu comportamentul principal pe care nivelul de logică trebuie să îl implementeze și care folosește obiecte definite în nivelul de date.

DataLayer – este nivelul responsabil cu manipularea bazei de date. Aici se regăsește codul aferent entităților (după cum am menționat anterior, mai întâi s-a creat codul corespunzător

entităților și, mai apoi, acestea au fost generate), un repository generic (pe care toate repositoryele entităților îl moștenesc), repositoryele fiecărei entități, un context cu ajutorul căruia se construiește baza de date, o clasă ce inițializează baza de date prin introducerea unor valori în cazul în care aceasta este goală și un dosar cu migrări (Entity Framework funcționează cu ajutorul acestor migrări care sunt, practic, un istoric al modificărilor aduse structurii bazei de date).

4.3 Modelare datelor

Datele necesare dezvoltării și manipularii aplicației sunt salvate într-o bază de date de tip relațional (adică formată dintr-un ansamblu de tabele și legăturile/relațiile dintre ele).

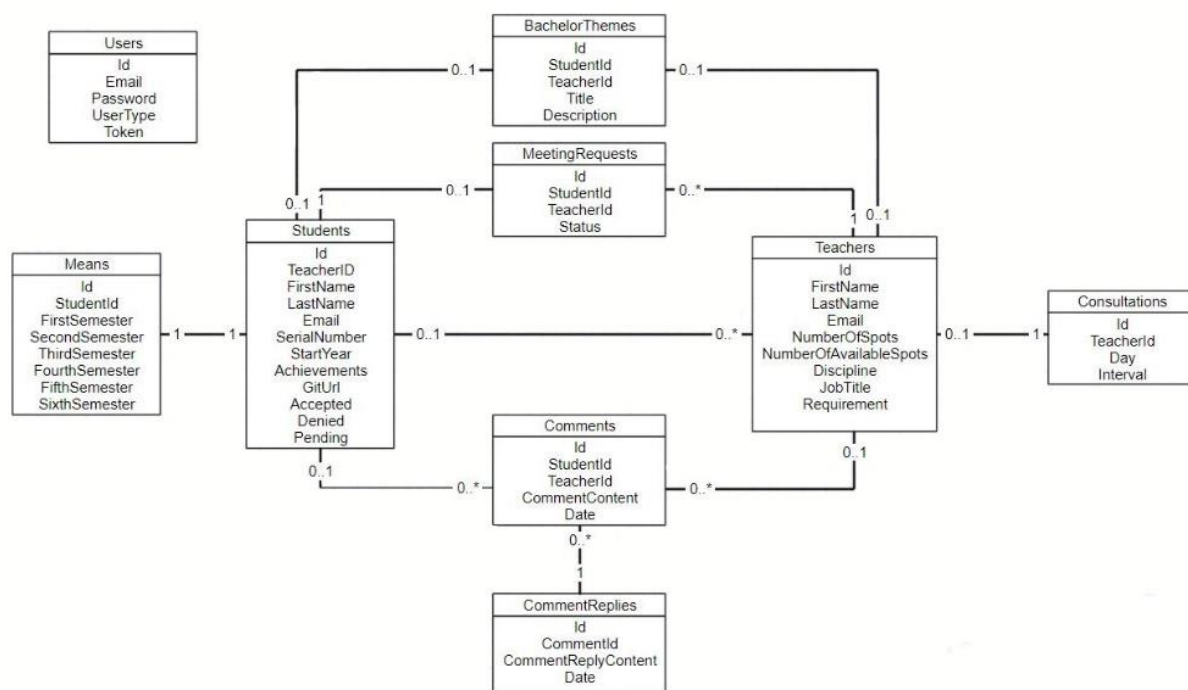


Figura : Diagrama bazei de date

Entități

Users – este tabela care definește orice utilizator ce intra în contact cu aplicația și aparține Facultății de Informatică din Iași. Fiecare utilizator va avea un email, o parolă, un tip de utilizator (din cele trei posibile: student, profesor, administrator) și un semn (eng. token) cu ajutorul căruia se verifică autentificarea lui înainte de fiecare operație.

Students – este tabela care definește studenții care se înregistrează în aplicație. Aceștia au informații de tipul: nume complet, email, număr matricol, anul începerii facultății, reușitele lor din timpul facultății, o adresă la care va putea fi găsită lucrarea lor de licență și trei câmpuri indicatoare (dacă studentul este acceptat de către profesor, dacă este refuzat de către profesor sau dacă încă este în stare de așteptare pentru un răspuns din partea profesorului).

Teachers – este tabela care definește profesorii care pot juca rolul de coordonator. Aceștia au informații de tipul: nume complet, email, numărul maxim de studenți pe care îi poate coordona, numărul de studenți pe care dorește să îi coordoneze (limitat superior de numărul maxim de studenți), disciplina pe care o predau, gradul pe care îl au ca și cadre didactice și cerințele minime pe care studenții trebuie să le îndeplinească pentru a-i accepta.

Means – este tabela asociată numai studenților care conține informații despre mediile pe care un student le are din primul semestru de facultate și până în semestrul curent, numărul maxim de semestre fiind șase.

BachelorThemes – este tabela care definește temele lucrărilor de licență ce vor fi realizate în sesiunea curentă. Acestea conțin informații despre deținătorul temei, despre titlul acesteia și o mică descriere asociată ei.

MeetingRequests – este tabela care definește statusul unei întâlniri solicitate de student către profesorul său coordonator, acesta putând fi: în așteptare, acceptat sau respins.

Comments – este tabela care definește comentariile ce pot fi adăugate, atât de studenți cât și de coordonatori pe pagina aferentă coordonatorilor. Aceasta conține informații despre autorul comentariului, conținutul comentariului și data la care acesta a fost adăugat.

CommentReplies – este tabela care definește răspunsurile la comentariile detaliate mai sus. Structura este asemănătoare cu cea a tablei aferente comentariilor dar mai are în plus id-ul comentariului la care este asociat răspunsul.

Consultations – este tabela asociată doar profesorilor coordonatori și definește o zi și un interval din acea zi în care cadrul didactic este disponibil pentru a primi solicitări de întâlniri în persoană cu studenții pe care îi coordonează.

Legături între entități

Students – Teachers: este o relație de tip unul la mai mulți. Un student poate avea un singur profesor coordonator, însă un cadru didactic poate îndruma mai mulți studenți.

Students – MeetingRequests: este o relație de tip unul la unul. Un student poate avea, la un moment dat, o singură cerere de întâlnire personală cu profesorul său coordonator și o astfel de întâlnire poate avea asignat un singur student.

Students – Means: este o relație de tip unul la unul. Un student poate avea o singură înregistrare corespondentă care să îi indice notele pe parcursul semestrelor petrecute în cadrul facultății și o înregistrare de tip medii poate aparține unui singur student.

Students – BachelorThemes: este o relație de tip unul la unul. Un student poate avea o singură temă de licență asignată și o temă de licență poate aparține unui singur student.

Students – Comments: este o relație de tip unul la mai mulți. Un student poate avea mai multe înregistrări de tip comentariu asociate dar un comentariu poate avea un singur autor, mai exact un singur student.

Teachers – BachelorThemes: este o relație de tip unul la unul. Un profesor coordonator poate avea o singură temă de licență asignată și, în același timp, o temă de licență poate aparține unui singur cadru didactic.

Teachers – Comments: este o relație de tip unul la mai mulți. Un profesor coordonator poate avea mai multe înregistrări de tip comentariu asociate dar un comentariu poate avea un singur autor, mai exact un singur profesor.

Teachers – Consultations: este o relație de tip unul la unul. Un profesor coordonator poate avea un singur slot de timp asignat la un moment dat și, în același timp, o înregistrare de tip consultație poate aparține unui singur cadru didactic.

Comments – CommentReplies: este o relație de tip unul la mai mulți. Un comentariu poate avea mai multe înregistrări de tip răspuns, dar un răspuns poate aparține unei singure înregistrări de tip comentariu.

4.4 Comunicarea server-client

Comunicarea între cele două entități, server și client, se face cu ajutorul protocolului HTTP. Fiind un protocol utilizat pe scară largă, aplicația de tip client se poate schimba în orice moment, chiar fiind posibilă existența mai multor clienți de tipuri diferite în același timp (un client poate fi o aplicație web care folosește un motor de căutare, un client poate fi o aplicație de tip mobil sau o aplicație special concepută pentru a simula cererile unui client către server cum ar fi Postman, aplicație desktop folosită în timpul dezvoltării curetei lucrări de licență). Acesta pune la dispoziție o sumedenie de metode prin care se pot obține date iar cele folosite în aplicația curentă sunt:

- metoda GET: este o metodă cu ajutorul căreia se pot cere date de la o rută specificată. Așa cum se observă în *Figura* și în *Figura* metoda are nevoie de o rută pe care o va apela utiliza aplicația client pentru a obține datele necesare și, posibil, de niște parametri. Serverul poate răspunde cu ajutorul unui obiect sau doar cu o variabilă de orice tip.

```
[HttpGet("getAccessToken/{email}")]  
public IActionResult GetAccessToken(string email)
```

Figura : Signatura unei metode de tip get GET, AccountController

```
const resp = this.http.get('http://localhost:64250/api/student/teacher/')
```

Figura : Parte din apelul unei metode de tip get GET

- metoda POST: este o metodă cu ajutorul căreia se adaugă noi date. Serverul are nevoie să primească de la client un obiect populat cu anumite date pe care, mai apoi, le prelucrează în diferite moduri și, în cele din urmă, inserează o nouă înregistrare în baza de date.

```
[HttpPost("register")]  
public IActionResult Register([FromBody] AccountDto accountDto)
```

Figura : Signatura unei metode de tip POST, AccountController

```
const resp = this.http.post('http://localhost:64250/api/account/login')
```

Figura : Parte din apelul unei metode de tip get POST

- metoda PUT: este o metodă cu ajutorul căreia, de cele mai multe ori, se actualizează datele deja existente în sistem. În cazul în care datele care se vor a fi actualizate nu există, metoda PUT preia din rolul metodei POST și inserează o nouă înregistrare. La fel ca și metoda anterioară, și aceasta are nevoie ca aplicația client să trimită un obiect prepopulat cu datele necesare pe care serverul le manageriază și decide care sunt cele ce trebuiesc actualizate sau dacă este nevoie de crearea unei înregistrări noi.

```
[HttpPut("editTeacher")]  
public IActionResult EditTeacher([FromBody] TeacherDto teacherDto)
```

Figura : Signatura unei metode de tip PUT, AdminController

```
const resp = this.http.put('http://localhost:64250/api/student/editStudent')
```

Figura : Parte din apelul unei metode de tip PUT

Concluzii

Prezenta lucrare de licență demonstrează faptul că folosind o suită bine aleasă de tehnologii și instrumente software se poate realiza o aplicație web, cu scară de utilizare medie dar cu posibilitate de extindere, care să fie utilizată cu succes în mediul academic și care să ușureze modul de lucru al utilizatorilor săi.

Având în vedere că tehnologia se află într-o continuă dezvoltare, cel mai benefic pentru noi devine faptul că o putem utiliza oricând, merintându-se a fi create tot felul de aplicații care să ne ajute în atingerea acestui scop.

Așadar, aplicația “**Managementul lucrărilor de licență**” se pretează a veni ca sprijin atât pentru generațiile viitoare de studenți ce vor trece pragul Facultății de Informatică din Iași, cât și pentru cadrele didactice ce activează în cadrul acesteia. Aceasta reușește să ajute toți utilizatorii să economisească timp prețios, să se descopere între ei (studenții au acces la ideile domnilor profesori coordonatori și invers) și să asigure o colaborare eficientă și armonioasă.

Versiunea curentă a aplicației poate fi îmbunătățită în mai multe direcții:

- se poate adăuga funcționalitatea întâlnirilor în grupuri de persoane (nu doar întâlnirile student - coordonator), studenții putându-și acorda sfaturi între ei;
- având în vedere că aplicația este deja integrată cu GitHub API, se pot aplica diverși algoritmi asupra datelor extrase pentru a putea face predicții privind rata de succes a lucrării;
- tot din existența colaborare cu GitHub API, se poate implementa un mecanism care să declanșeze anumite acțiuni atunci când un student își actualizează codul sursă și acesta ajunge să fie încărcat în sistemul de versionare;
- funcționalitatea curentă se poate extinde pentru a avea utilizatori din toată Universitatea Alexandru Ioan Cuza din Iași sau poate din toată țara, dacă aplicația se dovedește a fi utilă, momentan ea acceptând doar utilizatori ce au o legătură directă cu Facultatea de Informatică din Iași.

Bibliografie

1. Principii REST: <https://www.c-sharpcorner.com/uploadfile/kalisk/rest-fundamentals/>
2. Hypertext Transfer Protocol: <https://developer.mozilla.org/en-US/docs/Web/HTTP>
3. GitHub: <https://github.com/>
4. ASP NET Core: <http://www.jomendez.com/2017/02/15/asp-net-core-1-0/>
5. ORM: <https://searchwindevelopment.techtarget.com/definition/object-relational-mapping>
6. Entity Framework Core: <http://www.entityframeworktutorial.net/efcore/entity-framework-core.aspx>
7. HTML pentru începători: <https://www.w3schools.com/html/>
8. Angular : <https://softwaredevelopment.ae/angular-best-solution-2018-web-app/>
9. Cele mai populare tehnologii: <https://insights.stackoverflow.com/survey/2017#mostpopular-technologies>
10. Documentație Visual Studio Code: <https://code.visualstudio.com/docs>
11. Detalii GitHub: <https://www.atlassian.com/git/tutorials/what-is-git>
12. Edusoft: <https://www.edusoft.ro/>