**UNIVERSITATEA “ALEXANDRU IOAN CUZA” DIN IAȘI**

**FACULTATEA DE INFORMATICĂ**



## LUCRARE DE LICENȚĂ

„OnlineLibrary”

**propusă de**

***ALEXANDRESCU AURELIAN***

Sesiunea: *07, 2019*

**Coordonator științific**

Drd. Colab. Florin Olariu

**UNIVERSITATEA “ALEXANDRU IOAN CUZA” DIN IAȘI**

**FACULTATEA DE INFORMATICĂ**

„OnlineLibrary”

***ALEXANDRESCU AURELIAN***

Sesiunea**:** *07, 2019*

**Coordonator științific**

***Drd. Colab. FLORIN OLARIU***

DECLARAȚIE DE CONSIMȚĂMÂNT

Prin prezenta declar că sunt de acord ca Lucrarea de licență cu titlul „*OnlineLibrary*”, 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” 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.

Iași, *data*

Absolvent  *Alexandrescu Aurelian*

(semnătura în original)

ACORD PRIVIND PROPRIETATEA DREPTULUI DE AUTOR

Facultatea de Informatică este de acord ca drepturile de autor asupra programelor-calculator, în format executabil și sursă, să aparțină autorului prezentei lucrări, Aurelian Alexandrescu.

Încheierea acestui acord este necesară din următoarele motive:

[Se explică de ce este necesar un acord, se descriu originile resurselor utilizate în realizarea produsului-program (personal, tehnologii, fonduri) și aportul adus de fiecare resursă.]

Iași, data

Decan Prenume Nume Absolvent Prenume Nume

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(semnătura în original) (semnătura în original)

Cuprins

[LUCRARE DE LICENȚĂ 1](#_Toc12464788)

[Introducere 6](#_Toc12464789)

[Tema 6](#_Toc12464790)

[Motivația 6](#_Toc12464791)

[Gradul de noutate a temei 7](#_Toc12464792)

[Obiective generale ale lucrării 8](#_Toc12464793)

[Metodologia folosită 8](#_Toc12464794)

[Descrierea sumară a soluției 9](#_Toc12464795)

[Structura lucrării 9](#_Toc12464796)

[Contribuții 9](#_Toc12464797)

[Capitolul 1 – Descrierea problemei 11](#_Toc12464798)

[Capitolul 2 – Descrierea soluției 15](#_Toc12464799)

[Introducere 15](#_Toc12464800)

[Tehnologii folosite 15](#_Toc12464801)

[Descrierea aplicației 27](#_Toc12464802)

[Header 27](#_Toc12464803)

[Home 28](#_Toc12464804)

[My library 28](#_Toc12464805)

[Want to read 33](#_Toc12464806)

[Book 33](#_Toc12464807)

[Rented books 35](#_Toc12464808)

[Returned books 35](#_Toc12464809)

[Concluziile lucrării 35](#_Toc12464810)

[Direcții viitoare de cercetare 36](#_Toc12464811)

[Bibliografie 38](#_Toc12464812)

# Introducere

## Tema

Tema aleasă este reprezentată de o aplicație web în care utilizatorul poate să împrumute cărți de la alți utilizatori, și acesta la rândul lui să împrumute altora din cărțile ce le deține. Aplicația poate fi folosită și de bibliotecile autorizate, acestea urcându-și pe site toate cărțile pe care le dețin în bibliotecă. Aplicația are ca scop mutarea online a tot ce ține de împrumutul de cărți, nemaifiind nevoia de a căuta cartea propriu-zisă la bibliotecă.

## Motivația

Am ales această temă deoarece în momentul de față nu există nicio aplicație web care centralizează toate bibliotecile dintr-un oraș. În loc să se caute o carte prin fiecare bibliotecă în parte, cartea poate fi căutată în toate deodată, după care utilizatorul își poate alege biblioteca cea mai apropiată pentru a o împrumuta.

În societate este nevoie de o astfel de aplicație pentru că ne-ar economisi foarte mult timp atunci cand vrem să împrumutăm o carte pentru un anumit interval de timp. Totul s-ar face online iar bibliotecile n-ar mai exista decat fizic pentru ridicarea cărții și online pentru căutarea cărții. Cititorii în momentul actual, dacă doresc să citească o carte ce nu o dețin, trebuie să meargă pe la fiecare bibliotecă în parte pentru a o găsi. Unele biblioteci și-au extins aria și pe mediul online, cum ar fi Biblioteca Centrală Universitară „Mihai Eminescu” Iași. Aceasta a pus la dispoziție cititorilor ce dețin abonament la ea, să se poată căuta o carte online, ce aparține acestei biblioteci, arătând cititorului toate cărțile disponibile pentru a fi împrumutate. Odată împrumutată cartea, aceasta poate fi ridicată numai de la adresa bibliotecii. Sunt biblioteci cunoscute în Iași ce nu au o aplicație web, unde să pună la disponibilitatea cititorului, cărțile disponibile din aceasta, de exemplu Biblioteca Universitatii Tehnice “Gheorghe Asachi”, Biblioteca Academiei Române – Iaşi, Biblioteca Centrală U.M.F. “Gr.T.Popa” Iaşi. Ar fi un pas foarte mare pentru acestea, să aibe trecute cărțile în mediul online. Le-ar fi mai ușor atât pentru cititori, cât și pentru administratorii bibliotecii, să împrumute o carte, respectiv să o administreze. Și-ar deschide ariile de la câțiva clienți, la întreaga populație a unui oraș. În momentul de față doar anumiți oameni știu de ele, și doar aceștia le folosesc. Ar ajuta orașul enorm dacă bibliotecile respective, ar trece online. Cărțile acestora vor putea fi văzute de oricine dorește să caute o carte pe care ei le dețin.

Acum imaginați-vă numai cum ar fi o aplicație web care deține toate bibliotecile din oraș. Atunci când dorim să căutăm o carte, numai trebuie să o căutăm fizic în incinta bibliotecilor, sau să o căutăm online pentru bibliotecile ce au această disponibilitate, pur și simplu o căutăm pe OnlineLibrary, o aplicație web ce deține toate bibliotecile din oraș, cu toate cărțile în parte.

Deasemenea, un utilizator își poate creea propria bibliotecă, și își poate urca toate cărțile pe care le deține și este binevoit să le împrumute altora. Acest lucru poate fi transformat într-o mica afacere, pentru cei ce își doresc să contribuie la dezvoltarea societății.

Deci căutarea unei cărți pe OnlineLibrary, va include căutarea atât în toate bibliotecile din oraș, cât și în toate bibliotecile utilizatorilor binevoiți să ajute societatea urcând în aplicație cărțile pe care le dețin. Tot procesul de împrumut al cărții va fi afișat și salvat în baza de date a aplicației. Vor fi înregistrate toate împrumuturile de cărți atât pentru bibliotecă, cât și pentru utilizator. Fiecare pas de la căutarea de carte, la cererea acesteia, la ridicarea acesteia, la deținerea acesteia și la returnarea acesteia, vor fi afișate în aplicație. La finalul unui astfel de proces, se atribuie fiecărui utilizator în parte ce a împrumutat cartea, un scor ce va ajuta bibliotecile să vadă cât de serios este acesta. De asemenea atât cel care împrumută cartea, cât și cel care o dăruiește, au posibilitatea de a adăuga o recenzie pentru fiecare în parte. Aceste recenzii fiind disponibil de vizualizat în pagina lor de informații. Pe baza acestor recenzii, proprietarii de cărți își pot alege cititorii care doresc să împrumute cărți de la ei.

Unele biblioteci sunt încă necunoscute de majoritatea oamenilor. Această aplicație le face posibilitatea de a apărea pe site în urma unei căutări de carte, dacă biblioteca deține cartea respectivă. Odată ce a fost căutată o carte de către un utilizator, aceasta va apărea doar dacă există cel puțin o bibliotecă ce o deține. Dacă cartea aparține mai multor biblioteci, în momentul în care utilizatorul dorește să o împrumute, îi va apărea toate bibliotecile ce o dețin. Acesta având posibilitatea de a alege oricare bibliotecă dintre cele apărute. Utilizatorul se poate orienta după distanța dintre el și bibliotecă sau după recenziile făcute la biblioteca respectivă, date de alți utilizatori.

Consider că este nevoie în societatea noastră de o astfel de aplicație, deoarece din ce în ce mai multe lucruri ce se făceau înainte numai prin prezență fizică, acum au trecut online, acesta este pasul uriaș pentru împrumutul de cărți să treacă de la fizic la online, iar această aplicație web se potrivește perfect pentru necesitatea actuală.

## Gradul de noutate a temei

În ceea ce privește gradul de noutate al temei, pot spune că încă nu există nicio aplicație web care centralizează toate bibliotecile dintr-un oraș, permițând de asemenea utilizatorilor să creeze propria lor bibliotecă pentru a mări numărul de cărți disponibile pentru a fi împrumutate. În urma căutării unei cărți de către utilizator, acestuia o să îi apară lista cu biblioteci ce dețin cartea respectivă. După ce și-a ales biblioteca, acestuia să îi apară drumul cel mai scurt către locația bibliotecii.

## Obiective generale ale lucrării

Obiectivele generale ale lucrării sunt:

* Înregistrarea în aplicație a tuturor bibliotecilor dintr-un oraș
* Scăderea timpului de căutare a unei cărți de către cititor, acesta nemaifiind nevoit să caute în fiecare bibliotecă în parte fizic, poate căuta în toate deodată online
* Creșterea substanțială de cărți pentru împrumutat, în momentul în care un utilizator își poate poate crea propria sa bibliotecă
* O mai bună administrare a împrumuturilor de cărți pentru o bibliotecă, aceasta reușind să filtreze utilizatorii care doresc să împrumute o carte, pe baza recenziilor atribuite de alte biblioteci și a scorului acestora

## Metodologia folosită

Ca primă fază, împreună cu profesorul îndrumător, au fost stabilite cerințele funcționale ale aplicației. După care a urmat ca aceste cerințe să fie împărțite pe task-uri și rezolvate în funcție de gradul de importanță. Înainte de a începe rezolvarea taskurilor, a trebuit făcut mediul de lucru utilizând pe partea de client Angular iar pe partea de server Spring. Bazele de date fiind construite în MariaDB. Am folosit git ca și versionare a proiectului, pentru a avea un workflow cât mai ușor de urmărit. Repository-ul fiind public, profesorul îndrumător mi-a oferit sfaturi oricând aveam nevoie.

Realizarea proiectului a durat aproximativ 6 luni, lucrându-se alternativ între back-end și front-end. Designul web al aplicației a fost făcut încă de la început, acesta reușind în mare parte să fie respectat până la sfârșit. În mod general se făceau mai întâi serviciile back-end, după care serviciile front-end, pentru testarea acestora. Pentru o mai bună înțelegere a codului, am adăugat comentarii pe alocuri iar codul a fost scris cât mai curat.

## Descrierea sumară a soluției

Soluția pentru această problemă ar fi o aplicație web în care printr-o simplă căutare de carte, utilizatorului să-i apară toate bibliotecile ce dețin acea carte, acesta fiind nevoit să aleagă să împrumute cartea de la oricare bibliotecă dorește. Deasemenea utilizatorului îi pot apărea cărți deținute de alți utilizatori, aceștia fiind binedispuși să o împrumute.

## Structura lucrării

În lucrare voi discuta despre vechea problemă din societate, în care un om trebuie să caute o carte la bibliotecă personal și nu online, și rezolvarea acesteia. Lucrarea este structurată pe două mari capitole:

* Descrierea problemei
* Descrierea soluției

În descrierea problemei voi face comparație între împrumutul de cărți digitale și împrumutul de cărți tipărite. Voi specifica cum aplicațiile ce se ocupă cu împrumutarea de cărți digitale, sunt într-o continuă creștere, pe când cele ce se ocupă cu împrumutarea de cărți digitale, stagnează. Majoritatea cititorilor preferă să țină o carte în mână atunci când citesc.

În descrierea soluției voi prezenta aplicația web ce rezolvă problema descrisă mai sus. Voi prezenta fiecare funcționalitate în parte și voi explica modul în care ajută utilizatorul. De asemenea voi prezenta tehnologiile folosite în această aplicație web, și de ce le-am ales pe acestea.

După capitolele menționate mai sus, voi specifica ce mai poate fi adus în această aplicație pe viitor, care este direcția ei și în cât timp ar putea fi pe piață. În concluzie voi prezenta impactul pe care l-ar face aplicația, odată lansată pe piață. Urmările și consecințele acesteia, nu vor trece nemenționate.

## Contribuții

Am realizat o aplicație web care se ocupă cu împrumutul de cărți din bibliotecile înregistrate pe acesta. Aplicația are ca scop reducerea timpului a procesului de împrumut a unei cărți, de asemenea o mai bună administrare a bibliotecii de către administratorul acesteia. De asemenea un utilizator ce dorește să împrumute o carte, poate alege dintre varietatea de biblioteci înregistrate pe aplicație, ce dețin cartea respectivă, oferindu-i calea cea mai rapidă de la locația sa curentă, la locația bibliotecii.

Aplicația poate fi folosită de oricine dorește să împrumute o carte, sau oricine dorește să ofere o carte drept împrumut. Utilizatorul își poate crea propria bibliotecă, punându-și cărțile pe care le deține, în aplicație. Procesul de împrumut a unei cărți realizat în această aplicație este unul sigur, deoarece cei care administrează biblioteca își pot alege utilizatorii în funcție de un scor, sau de recenziile atribuite acestora, pe pagina de informații a utilizatorului. Recenziile fiind atribuite de administratorii de biblioteci, în urma finalizării unui împrumut.

# Capitolul 1 – Descrierea problemei

Pe piața din România nu există încă astfel de aplicații care să centralizeze toate bibliotecile din țară, însă sunt anumite biblioteci ce au trecut pe mediul online în ceea ce privește împrumutul de cărți. De asemenea există aplicații web ce oferă utilizatorului o carte ebook. Aceasta este o metodă prin care poți citi o carte fără să trebuiască să o deții fizic, dar pe un dispozitiv cu memorie internă. Mai jos voi preciza câteva dintre aplicațiile web ce oferă utilizatorului ocazia de a citi cartea dorită.

### BCU Iași

Din punctul meu de vedere BCU Iași este singura aplicație din acest oraș care se apropie cel mai tare de aplicația web OnlineLibrary. Dar aceasta are de asemenea și foarte multe lipsuri.

* Căutarea unei cărți
* Afișarea dacă această carte este disponibilă
* Afișarea informațiilor necesare despre o carte
* Adaugarea de recenzii pe o carte
* Cererea de împrumut a acesteia
* Trimiterea de email de la utilizator la bibliotecă și de la bibliotecă la utilizator

### Biblioteca Metropolitană București

Biblioteca Metropolitană București este una din bibliotecile care sunt și online și fizic, care are ca și funcționalități următoarele servicii gratuite:

* Cuprinde împrumutarea de cărți ebook precum: înregistrări sonore, video, DVD-uri și CD-uri, periodice și documente on-line destinate oricărei categorii de utilizatori
* Oferă servicii gratuite de împrumut de carte
* Organizează programe culturale și educaționale diverse
* Rezervarea locurilor în sală de lectură

Aceasta are și servicii speciale pentru utilizatori contra cost precum:

* împrumut interbibliotecar intern și/sau internațional, în baza unei solicitări prealabile și cu condiția acoperirii de solicitant a costurilor de transport al documentelor solicitate
* Digitalizarea și transmiterea prin e-mail sau pe suportul solicitantului a fragmentelor din documente aflate în colecțiile bibliotecii
* Cercetări bibliografice și referințe complexe la cerere în baza unei solicitări prealabile și în limitele colecției B.M.B.

### Biblioteca central universitară „Lucian Blaga” Cluj-Napoca

Din punctul meu de vedere biblioteca central universitară „Lucian Blaga”, are servicii care fac ca această bibliotecă să se apropie cel mai mult de OnlineLibrary, și este cea mai completă dintre toate celelalte biblioteci din România. Aceasta are aceleași servicii de gratuitate și de contra cost pentru utilizatori ca și B.M.B, dar are în plus următoarele servicii:

* Închirierea unei săli de conferință
* Scanarea, fotocopierea documentelor din colecțiile speciale

Concurența acestor aplicații web ce oferă posibilitatea să împrumuți cartea de acasă și să o ridici fizic de la locația bibliotecii sunt aplicațiile web ce oferă cărțile în format ebook. Am enumerate o listă mai jos cu astfel de situri:

### Google Books

Google Books a apărut on-line înca din anul 2004 sub denumirea de „Google căutare de cărți” sau „Google Print”. Acesta este un serviciu Google care la momentul căutării unei cărți pe aplicație caută în conținutul cărților textul respectiv. Google Books este pe departe cea mai bună aplicație web, ce are la bază împrumutul de cărți digitale. Aceasta are următoarele funcționalități:

* Căutarea unei cărți
* Dacă cartea nu este protejată prin drepturi de autor sau editorul a dat permisiunea, utilizatorii vor putea să vadă o previzualizare a cărții și, în unele cazuri, întregul text
* Dacă este în domeniul public, utilizatorii sunt liberi să o descarce o copie PDF
* Crearea propriei biblioteci
* Scrierea de recenzii pentru o carte
* Istoricul căutărilor recente, cărților achiziționate și a cărților citite
* Propunerea de cărți de citit pentru utilizator printr-un algoritm ce folosește inteligența artificială

### BookRix

BookRix este o platformă gratuită de publicare proprie, care oferă servicii de distribuție electronică scriitorilor independenți. Utilizatorii acestei aplicații pot să-și împărtășească scrisul, să se conecteze cu alți cititori și să descopere cărți și autori noi. Toate într-un singur loc. Această aplicație față de Google Books promovează mai bine autorul unei noi cărți având astfel serviciul de Book-Bestseller. Un serviciu ce arată utilizatorilor care este cea mai bine vândută carte la momentul actual. Cu toate că are acest serviciu în plus, totalul de cărți din BookRix nu se compară cu multitudinea de cărți din Google Books.

Cărțile tipărite pierd tot mai mult teren în fața cărților digitale. Fiecare persoană preferă un anumit mod de a citi o carte: de pe tabletă, eReader sau din carte. Mai jos am listat o serie de avantaje și dezavantaje în cadrul ambelor variante de cărți:

|  |  |
| --- | --- |
| Cartea tipărită | |
| Avantaje | Dezavantaje |
| poți folosi un semn de carte pentru a găsi ușor pagina unde ai rămas. În cazul eBook-urilor, dacă dispozitivul se resetează, pagina la care ai rămas se pierde | pentru a o citi seara sau noaptea, ai nevoie de o sursă de lumină |
| cartea nu are baterie care se descarcă | unele cărți pot fi foarte voluminoase și grele, acestea fiind incomod de ținut în mână |
| îți poți face însemnări prin sublinierea frazelor care ți-au plăcut | hârtia se învechește, îngălbenește și capătă un miros vechi. Totuși acest lucru îi poate ridica valoarea |
| cartea tipărită o poți împrumuta și altora | cărțile tipărite sunt mai scumpe decât varianta lor eBook |
| cărțile tipărite pot deveni relicve de familie |  |
| cartea tipărită nu poate fi ștearsă din greșeală |  |
| o bibliotecă mare, îți înfrumusețează casa, iar cărțile electronice nu-ți pot face asta |  |

|  |  |
| --- | --- |
| Cartea digitală | |
| Avantaje | Dezavantaje |
| unele cărți pot fi găsite gratuite în versiunea de eBook | bateria dispozitivului se poate descărca, exact atunci când ai mai mare nevoie |
| are o greutate mai mică decât o carte groasă | în eBook-uri nu poți face însemnări |
| e mai comod să ții în mână o tabletă sau un eReader, de pe care să citești, decât o carte |  |
| eBook-urile sunt mai ieftine |  |
| îți poți regla intensitatea luminii de pe dispozitivul cu care citești |  |
| într-o tabletă sau eReader, îți poți păstra peste sute de cărți pe care le ai la îndemână oricând |  |

### Concluzie

În ceea ce privește cărțile digitale, aplicațiile web ce se ocupă cu astfel de cărți se dezvoltă foarte rapid, sunt într-o continua creștere. Dar în ceea ce privește cărțile tipărite, aplicațiile web ce se ocupă cu astfel de cărți stagnează. Bibliotecile există încă din anul 300 î.Hr., și de atunci urmează același tipar în care cititorul trebuie să caute cartea fizic la bibliotecă pentru a vedea dacă aceasta o deține. Aplicațiile din ziua de azi se folosesc de mediul online pentru a căuta o carte cum ar fi cele menționate mai sus (BCU, BMB etc.), dar căutarea se face numai în incinta acelei biblioteci. Au trecut de bariera online, dar nu au avansat foarte mult. O aplicație ideală ar fi una care deține toate bibliotecile cu cărțile acestora la un loc, pentru a face căutarea unei cărți mult mai ușoară pentru cititor. Societatea noastră are nevoie de o astfel de aplicație, de aceea voi descrie soluția acestei probleme mai jos.

# Capitolul 2 – Descrierea soluției

## Introducere

Pentru a rezolva problema descrisă mai sus, societatea are nevoie de o aplicație web ce cuprinde pentru fiecare oraș în parte, toate bibliotecile publice și private din acesta. Aplicația web OnlineLibrary se potrivește perfect pentru o astfel de necesitate, aceasta permițând bibliotecilor să-și urce cărțile pe care le dețin fizic, pe aplicație. Utilizatorul nu va mai fi nevoit să caute cartea pe care și-o dorește din bibliotecă, în bibliotecă. Acesta o poate căuta pe OnlineLibrary, iar dacă vreo bibliotecă deține cartea respectivă, utilizatorul poate să o împrumute de la distanță, iar în momentul aprobării împrumutului de către bibliotecă, utilizatorului i se arată ruta cea mai scurtă către locația bibliotecii, din locația lui curentă.

De asemenea utilizatorii își pot crea propria bibliotecă, unde își pot urca cărțile pe care le dețin. Acesta este un plus aplicației deoarece numărul de oferte crește. Există posibilitatea ca o bibliotecă să nu dețină cartea dorită, dar un alt utilizator să o aibă. Astfel, pentru cei ce vor să împrumute cărți doar pentru o anumită perioadă de timp, OnlineLibrary este locul perfect pentru cititori.

## Tehnologii folosite

### Java

Java este unul dintre cele mai populare limbaje de programare din lume. Acesta este rapid, de încredere și sigur. Potrivit Oracle, compania ce deține Java, rulează pe cel puțin 3 miliarde de dispozitive. Deoarece numărul de dezvoltatori Java este un număr foarte mare, iar Java rulează pe atât de multe dispozitive, putem spune că acest limbaj de programare nu va dispărea prea curând.

La fel ca limba română, Java are un set de reguli care determină modul, în care instrucțiunile sunt scrise. Aceste reguli sunt cunoscute ca sintaxa sa. Odată ce un program a fost scris, instrucțiunile de nivel înalt sunt traduse în coduri numerice pe care computerele le pot înțelege și executa.

Caracteristici ale limbajului de programare Java:

#### Cross-platform

Java a fost construit cu filozofia „ write once, run anywhere” („scrie o dată, alerga oriunde”) (WORA). Codul Java (cod Java și librării pure) pe care îl scrieți pe o singură platformă (sistemul de operare) se va difuza pe alte platforme fără modificări cum ar fi Windows, Linux, Mac OS etc.Pentru a rula Java, este utilizată o mașină abstractă numită Java Virtual Machine (JVM). JVM execută Java bytecode. Apoi, CPU execută JVM. Deoarece toate JVM-urile funcționează exact la fel, același cod funcționează și pe alte sisteme de operare, făcând Java independent de platformă.

#### Limbaj de programare orientat pe obiecte

Există diferite stiluri de programare. Obiectiv orientat este una dintre stilurile de programare populare. În programarea orientată pe obiecte, o problemă complexă este împărțită în seturi mai mici prin crearea obiectelor. Acest lucru face ca codul dvs. să fie reutilizabil, are beneficii de proiectare și face ca codul să fie mai ușor de întreținut.

#### Java este rapid

În trecut versiunile Java au fost criticate pentru că sunt prea lente. Dar lucrurile s-au schimbat de atunci. Noile JVM-uri sunt mult mai rapide, aici fiind problema principala din trecut, iar procesorul care le execută devine din ce în ce mai puternic. În momentul de față Java este printre cele mai rapide limbaje de programare existente.

#### Java este sigur

În cee ace privește securitatea aplicațiilor Java oferă diferite funcții. Caracteristicile principale de nivel înalt pe care le ocupă Java sunt:

* o gestionare automată a memoriei care duce la reducerea corupției și vulnerabilitățile memoriei
* oferă o platformă sigură pentru dezvoltarea și rularea aplicațiilor
* oferă o comunicare sigură protejând confidențialitatea dar și integritatea datelor trimise

#### Bibliotecă standard mare

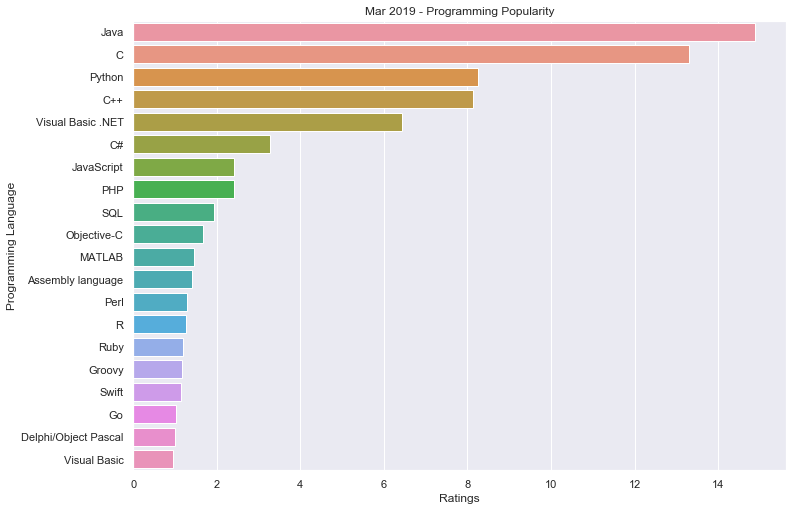
Având în vedere că Java folosește o bibliotecă uriașă standard, acesta este folosit pe scară largă de mulți dezvoltatori. Aceasta are sute de metode și clase din diferite pachete ce ajută dezvoltatorii de software să scrie cod cât mai puțin și curat. De exemplu, unele dintre cele mai populare librării sunt:

* java.lang pentru caracteristici avansate ale matricelor, șirurilor, vectorilor și altor clase
* java.io pentru citirea din fișier sau scrierea acestuia, precum și tratarea excepțiilor
* java.util pentru tot ce ține de date și timp, expresii regulate, structuri de date și altele

#### Concluzie

Am folosit Java 8 deoarece este o lansare revoluționară a acestei platforme de dezvoltare. Acesta include un upgrade foarte mare la modelul de programare Java și o evoluție coordonată a limbajului JVM, Java și a bibliotecilor. Java 8 include caracteristici de productivitate, ușurință în utilizare, programare îmbunătățită, securitate și performanță.

Mai jos am pus un grafic cu lista de cele mai populare limbaje de programare din luna mai 2019, conform indicelui TIOBE. Indicele comunitar de programare TIOBE este un indicator al popularității limbajelor de programare. Acesta vine de la „[*The Importance of Being Earnest*](https://en.wikipedia.org/wiki/The_Importance_of_Being_Earnest)*”,* ceea ce înseamnă *„ importanța de a fi mai bine*”. Acesta a fost creat, și este menținut de compania TIOBE din Olanda.



*Figura 1 - Grafic indice* TIOBE *martie 2019*

Figure - Grafic indice TIOBE martie 2019

### Java Spring Framework

Framework-urile sunt corpuri mai mari ce sunt constituite din mai multe clase, cu cod pre-scris la care dezvoltatorii își adaugă propriul lor cod, pentru a rezolva o problemă dintr-un anumit domeniu. Dezvoltatorii se folosesc de framework-uri deoarece acestea conțin funcționalități deja scrise, pe care dezvoltatorii le adaugă în proiectul lor. Aceștia apelează metode care folosesc modelul Observer.

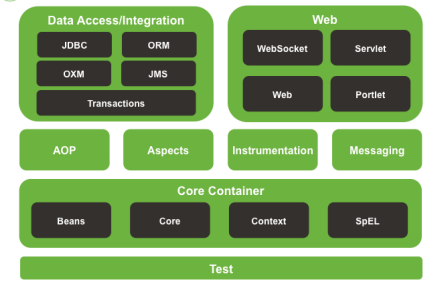
Java Spring este de asemenea un framework ce facilitează crearea de aplicații enterprise Java. Are tot ce este nevoie pentru a-l folosi într-un mediu de afaceri, cu support pentru Maven, Groovy si Kotlin ca limbi alternative pe JVM. Într-o întreprindere mare, aplicațiile există pentru o perioadă lungă de timp acestea fiind nevoite să ruleze pe un server de aplicații JDK, îmbunătățirea la o alta versiune fiind necontrolată de dezvoltatori. Spring este open-source, iar în spate are o comunitate uriașă de dezvoltatori, ce își spun părerea la orice problemă existent din cadrul framework-ului. Au ajutat la dezvoltarea cu succes a acestui framework încă de când a apărut.

Punctul forte al acestui framework stă în infrastructură. Spring are grijă de aceasta, în timp ce dezvoltatorii se pot concentra pe aplicația propriu-zisă. Spring-ul permite să construiți aplicații în *„ plain old Java objects”(POJOs)* și să aplicați servicii de întreprindere asupra acestora. Această capabilitate se aplică modelului de programare Java SE si Java EE.

Următoarele exemple m-au ajutat pe mine ca dezvoltator, să beneficiez de platforma Spring:

* printr-o simplă apelare a unei metode Java într-o tranzacție din baza de date, fără să trebuiască să folosesc API-urile de tranzacție
* am făcut o metodă Java locală care se ocupă de mesajele de eroare oricând este nevoie, fără să folosesc API-urile JMS
* am făcut diferite metode locale Java pentru apelarea serviciilor din aplicație, fără să folosesc API-urile JMX
* am facut o metodă locală de trimitere de email-uri, fără să folosesc API-urile pentru trimiterea email-urilor

Spring framework este organizat din aproximativ 20 de module, ce au la bază anumite funcționalități. Aceste module stau la baza framework-ului și sunt grupate în Core Container, Data Access / Integration, Web, AOP (Aspect Oriented Programming), Instrumentation, Messaging și Test, având diagrama de mai jos ca exemplu.



*Figura 2 - Arhitectura Spring*

Figure - Arhitectura Spring

#### Core Container

Modulele spring-beans și spring-core sunt modulele fundamentale ale framework-ului, incluzând caracteristicile IoC și Dependency Injection, caracteristici pe care Spring-ul se bazează cel mai mult. Folosesc în proiect Dependency Injection pentru a *„injecta”* în clase servicii ce au loc în altele. Este cel mai folosit modul de către dezvoltatori, deoarece acesta face legăturile între toate nivelele de arhitectură ale aplicației.

#### Data Access/Integration

Acest modul cuprinde tranzacțiile datelor din baza de date, respectiv Repository. În acest proiect folosesc modul JPA, un modul care permite dezvoltatorului să *„culeagă” sau să adauge* informații din baza de date sub forma unor tranzacții. Acest modul protejează proiectul de SQL Injections, având metode distinctive pentru afișarea, adăugarea, editarea și ștergerea datelor din baza de date, fără a folosi Query-uri SQL native.

Deasemenea, datorită acestui modul, baza de date poate fi generate automat, folosind adnotarea @Entity asupra claselor ce le considerăm a fi tabele în baza de date. Acestea se folosesc și la operațiile aplicate pe baza de date. Prin urmare la nivel de Repository se folosesc clasele de tip entitate.

#### Web

Modulul Web este legătura cu clientul. Toate requesturile date de acesta sunt ascultate în modul web, prin intermediul unui Controller. Sunt asignate oricărui Controller anumite end-point-uri (rute) care preia requestul de la client, și îl duce mai departe către servicii. Datorită acestui modul, este posibilă comunicarea dintre front-end și back-end. Pentru parsarea JSON-ului de la client către server se folosesc clasele de tip VO (Value Object). Sunt clase simple Java ce au ca, cheie și valoare, datele din JSON-ul trimis de la client. Deoarece pentru nivelul Repository se folosesc entitățile, am fost nevoit să convertesc clasele VO în Entity prin crearea de metode locale ce aparțin serviciului de conversie.

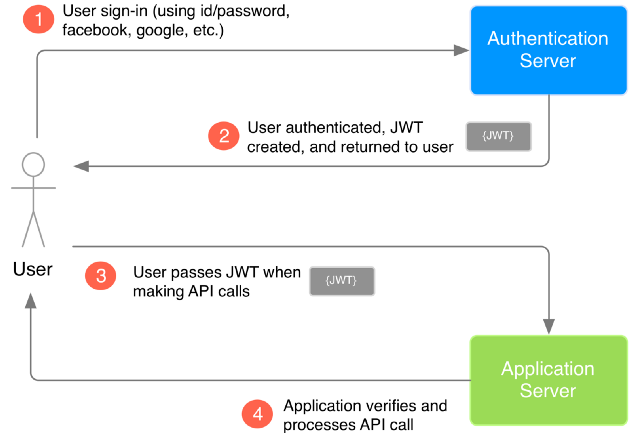
#### Concluzie

În concluzie, framework-ul Spring este cel mai potrivit pentru partea de server în cadrul lucrării de licență. Relaționarea cu baza de date este sigură și rapidă iar toate operațiile de adaugare, editare, ștergere sau afișare se fac aproape instant, acest lucru fiind ideal pentru aplicația web OnlineLibrary. Datorită framework-ului Spring, am reușit să fac diferite funcționalități din aplicație, fără să mă folosesc de alte API-uri publice. Acest framework face ca munca dezvoltatorului să fie cât mai curată, nelăsându-l să modifice clasele pe care le are framework-ul, doar apelându-le și adăugând cod peste, duce la o aplicație cât mai simplă de înțeles în cadrul codului. Fiind scrisă cât mai structurat, va fi mai ușor dezvoltatorilor să lucreze din nou pe cod dacă este nevoie, mai ales în cazul în care apare un bug. Metodele fiind ușor de înțeles și accesibile.

### JWT

JWT (JSON Web Token) este un obiect JSON definit în RFC 7519, folosit pentru transmiterea în siguranță a informațiilor dintre client și server. Token-ul este compus dintr-un antet, datele transmise și o semnătură. Acesta folosește un secret (prin intermediul algoritmului HMAC) sau perechea de chei publice/private utilizând ECDSA ori RSA.

Pentru a arăta cum și de ce am folosit JWT în această aplicație, am dat ca exemplu diagrama de mai jos. Entitățile din această diagram sunt utilizatorul, serverul de aplicații și serverul de autentificare. Serverul de autentificare, fiind integrat în Spring, îi trimite utilizatorului tokenul JWT numai dacă acesta a introdus corect datele de intrare în cont. Cu acest token, utilizatorul poate accesa în siguranță aplicația. Odată autentificat cu succes, Angular primește tokenul utilizatorului pe care îl salvează în „local storage”. Fiecare request trimis către Spring, va avea în header acest token. Spring va ști de fiecare dată cine accesează end-point-urile, pe baza acestui token.



*Figura 3 - Request de autentificare folosind JWT*

Figure - Request de autentificare folosind JWT

#### Concluzie

Am folosit pentru securitatea aplicației pachetul JWT deoarece este unul dintre cele mai sigure astfel de pachete. JWT se integrează cu Spring foarte ușor, ceea ce duce la o mai bună utilizare a acestuia de către dezvoltatori.

### Maven

Maven este un cuvânt idiș ce are ca însemnătate „acumulator de cunoștințe”. Integrat cu Java, are ca rol simplificarea proceselor de construire a unui proiect. Este o modalitate de a folosi JAR-uri în diferite proiecte. Pentru a nu introduce JAR-urile unul câte unul, Maven se ocupă de asta introducând el automat toate JAR-urile de care este nevoie în proiect. Maven poate fi folosit prin integrare cu Java Spring. Acesta are un fișier pom.xml unde sunt declarate toate dependențele din proiect. Odată trecută dependența în fișier, Maven aduce JAR-ul declarant în acea dependență, automat.

În acest proiect am folosit dependențe precum spring-boot (pentru a integra în proiect, framework-ul Spring), spring-security (pentru a oferi siguranță end-point-urilor accesate de afară), jjwt (pentru generarea de token în cazul intrării în cont al unui user), javax-mail (pentru trimiterea de email-uri), mariadb (pentru conexiunea cu baza de date) și multe altele. De asemenea în acest fișier sunt trecute și numele proiectului cu versiunea de Java stabilită. Voi afișa mai jos dependențele folosite în acest proiect.



#### Concluzie

Am folosit Maven deoarece framework-ul Spring merge foarte bine cu acesta, Maven având acces la toate dependențele necesare pentru Spring Framework. Aceste dependențe pot fi adăugate sau șterse cu ușurință, după necesitatea developerului.

### MariaDB

Pentru a se potrivi cu lumea modernă de astăzi, MariaDB adoptă o abordare fundamentală bazată pe baze de date. Operațiunile pot depinde acum de o singură bază de date completă, indiferent de hardware. MariaDB oferă agilitate operațională fără egal, fără a sacrifica caracteristicile cheie ale întreprinderii.

MariaDB este, fără îndoială, unul dintre cele mai populare servere de bază de date din lume. Originar a fost creat de dezvoltatori MySQL. Acesta este open-source și ca bază de date relațională oferă o interfață SQL pentru accesul de date. Versiunile mai recente de MariaDB, include de asemenea și caracteristici precum JOSN și GIS. Este folosit într-o gamă largă de aplicații, de la aplicații bancare la site-uri web făcute în Wordpress. Este înlocuitorul de tip drop-in, îmbunătățit, pentru MySQL. MariaDB este folosită pentru rapiditatea, scalabilitatea și robustivitatea sa, bazându-se pe un ecosystem bogat de motoare de stocare și pluginuri, ceea ce îl face să fie foarte versatile pentru o gamă largă de cazuri de utilizare.

Tabelele create în baza de date folosită pentru OnlineLibrary sunt:

* books
* aici sunt stocate cărțile din aplicație
* tabelul este folosit în cazul adăugării de cărți și în cazul căutării de carte
* libraries
  + sunt stocate toate bibliotecile create
  + tabelul este folosit în cazul adăugării de biblioteci și de afișare a acestora
* libraries\_books
* aici sunt stocate cărțile unei biblioteci
* tabelul este folosit pentru a afișa/edita/șterge cărțile din biblioteci
* libraries\_books\_rented
* aici sunt stocate cărțile împrumutate
* tabelul este folosit pentru a afișa cărțile ce doresc a fi împrumutate și cărțile împrumutate deja
* libraries\_book\_returned
* aici sunt stocate cărțile returnate
* tabelul este folosit pentru a afișa istoricul tuturor împrumuturilor
* libraries\_reviewes
* aici sunt stocate recenziile bibliotecilor
* tabelul este folosit pentru a adăuga recenzii bibliotecilor și pentru afișarea acestora
* notifications
* aici sunt stocate notificările
* tabelul este folosit pentru a alerta utilizatorul dacă are o notificare
* queue
* aici sunt stocate datele userilor și cărțile de la bibliotecă
* tabelul este folosit în cazul în care cartea dorită să fie împrumutată nu există (utilizatorul va trece într-o coadă de așteptare, și va fi alertat când îi vine rândul să împrumute cartea respectivă)
* books\_reviews
* aici sunt stocate recenziile cărților
* tabelul este folosit pentru a adăuga recenzii cărților și pentru afișarea acestora
* users
* aici sunt stocați utilizatorii înregistrați
* tabelul este folosit la înregistrarea unui cont nou, intrarea în cont și la afișarea de date despre utilizator
* users\_reviews
* aici sunt stocate recenziile utilizatorilor
* tabelul este folosit pentru a adăuga recenzii utilizatorilor și pentru afișarea acestora
* want\_to\_reads
* aici sunt stocate cărțile pe care utilizatorul dorește să le citească pe viitor
* tabelul este folosit pentru adăugarea, afișarea și ștergerea acestora

#### Concluzie

Am decis să folosesc MariaDB ca și bază de date deoarece proiectul este construit pe baza mai multor legături prin intermediul unor chei străine. Cum MariaDB este o ramură a MySQL-ului, aceasta este perfectă pentru relaționări între tabele. Pot fi aplicați și indecși având în vedere că aplicația se bazează mult pe căutare, iar MariaDB excelează la acest lucru.

### Angular

Angular ca și Java Spring este un framework open-source, acesta fiind folosit doar pentru crearea de aplicații client, folosind HTML, CSS și TypeScript. În esențial, este o structură care ajută dezvoltatorii web să creeze aplicații mai rapid, oferind un număr de obiecte și servicii care facilitează dezvoltatorii aplicației, acestea fiind scrise în TypeScript. Fiind creat și întreținut de Google, framework-ul Angular este sigur de folosit, actualizat și îmbunătățit de la o versiune la alta.

Structura angular se bazează pe module. Un modul reprezintă o multitudine de componente. Toate aplicațiile angular au la bază un modul, unde sunt importate alte module cu componente corespunzătoare pentru necesitățile dezvoltatorului, precum și Spring care are fișierul pom.xml, unde își declară dependențele. Cum Spring-ul folosește JAR-urile așa și Angular folosește modulele. De asemenea în module mai sunt declarate și directivele, componentele și serviciile.

Exemple de module folosite în aplicație:

* HttpClientModule
* Acest modul permite clientului să facă requesturi de tip GET, POST, PUT, DELETE către aplicația construită pe Spring pe back-end
* Poate fi folosit și pentru apelarea de API-uri publice de ex. Google Books, de unde caut cartea pentru a fi adăugată în bibliotecă
* AgmCoreModule
* Acest modul folosește API-ul de la Google Maps, prin intermediul unei chei
* Este folosit pentru afișarea în pagină a unei hărți cu locația bibliotecii, în momentul în care utilizatorul dorește să închirieze o carte
* AgmDirectionModule
* Ca și AgmCoreModule, acest modul folosește API-ul de la Google Maps, pentru afișarea de rute pe hartă
* Este folosit pentru a indica drumul cel mai scurt de la locația curentă a utilizatorului până la locația bibliotecii

După cum am spus mai sus, modulele reprezintă o multitudine de componente. O componentă are în structura sa un template HTML, un fișier pentru stilizare CSS și un fișier de tip TypeScript, unde se definește componenta și sunt apelate servicii.

Exemple de componente folosite în aplicație:

* Book
* Această componentă este folosită pentru afișarea informațiilor despre o carte
* La inițializarea acesteia în DOM, trimite automat un request GET către partea de back-end, cu parametrul de tip number „bookId”, Spring-ul trimițând ca și răspuns un obiect VO de tip book. La primirea acestui răspuns, Angular parsează JSON-ul și construiește HTML-ul afișat clientului, cu datele primite
* Mai conține și componenta „review-list” ce se ocupă de afișarea recenziilor de la carte după parametrul „bookId”
* Login
* Această componentă este folosită pentru intrarea în cont a utilizatorilor
* La completarea input-urilor din HTML, componenta Login trimite un JSON către Spring cu datele de autentificare ale utilizatorului respective. Pentru securitate Spring folosește „token-uri” jwt, însemnând că la o autentificare cu succes, Spring-ul trimite înapoi către Angular un token de autorizare, folosit pentru a accesa celelalte „end-point-uri” din back-end
* Want to read
* Această componentă este folosită pentru afișarea cărților din „Want to read”
* Trimite request-uri de tip GET și DELETE către Spring, pentru afișarea cărților din secțiunea „Want to read”, respective pentru ștergerea unei cărți din această secțiune

#### Concluzie

Am folosit Angular în principiu pentru că este un framework open-source și cross-platform. Pe lângă cele menționate mai sus, acesta este foarte rapid, încărcând componentele aproape instant. Structura proiectului create cu Angular este simplă și ușor de urmărit. Deoarece proiectul este structurat pe componente, dezvoltatorului îi este mult mai ușor pentru a depista o problemă în cazul în care apare una, aceste componente fiind independente una față de cealaltă. Serviciile de obicei sunt create în interiorul componentelor și sunt apelate prin intermediul funcționalității de Dependency Injection, precum Spring. Deoarece Angular folosește componente, la accesarea aplicației se încarcă în DOM o singură pagină HTML, componentele fiind apoi adăugate sau eliminate din DOM după cum acționează utilizatorul. Deasemenea datele sunt încărcate în pagină fară a mai trebui dat refresh la pagină.

### Concluzie tehnologii folosite

Din punctul meu de vedere framework-urile Angular și Spring sunt cele mai potrivite pentru aplicația web OnlineLibrary. Arhitectura celor doua framework-uri se aseamănă, acest lucru fiind ideal pentru dezvoltatorii acestei aplicații. Pe partea de client, Angular este unul dintre cele mai rapide framework-uri, acesta fiind mereu actualizat și îmbunătățit de Google. Legătura cu Spring-ul se face prin trimiterea de requesturi de tip GET, POST, PUT, DELETE către end-point-urile disponibile afișate în Controller-ul Spring. Acestea fiind trecute printr-un filtru de către JWT pentru a asigura securitatea aplicației. Totul se face prin apelarea unor metode, ce sunt independente unele față de altele. Requestul pleacă din Angular și se duce mai departe către Spring. Acesta interceptându-l pe baza unor Controllere, trimițând mai departe către nivelul de Business, la care acesta accesează nivelul Repository, ce stă la baza tranzacțiilor cu baza de date construită în MariaDB.

## Descrierea aplicației

Aplicația web OnlineLibrary poate fi folosită de orice persoană. Acesta trebuie doar să se înregistreze. Toate rutele sunt blocate, astfel încât doar utilizatorii care nu sunt intrați în cont, nu pot folosi aplicația. Singurele pagini accesibile fiind „Login” și „Register”.

În momentul înregistrării pe site a unui utilizator, după ce a completat forma de înregistrare conform cerințelor afișate, acesta este redirecționat către pagina de „Login” pentru a intra în cont. Utilizatorul poate intra în cont doar cu email-ul și parola, alese în momentul înregistrării. În urma introducerii corecte a acestora, utilizatorul va intra automat în cont, iar dacă datele au fost introduse incorect, utilizatorului îi va apărea mesaj de eroare.



Proiectul este structurat în mai multe componente. Prin schimbarea de rute accesând diferite butoane din site, se vor schimba componentele. Voi lista mai jos lista de componente pe care o folosesc în aplicație:

## Header

Odată intrat în cont, utilizatorului i se va schimba meniul de navigare de sus, în loc de login/register îi vor apărea tab-urile „My library”, „Want to read”, „Notifications” și „Settings”, și redirecționat pe pagina de căutare a unei cărți, aceasta fiind pagina principală a aplicației. Componenta header reprezintă meniul principal de navigare, ce nu va dispărea niciodată indiferent de componenta accesată.

Meniu de navigație înainte de intrarea în cont:

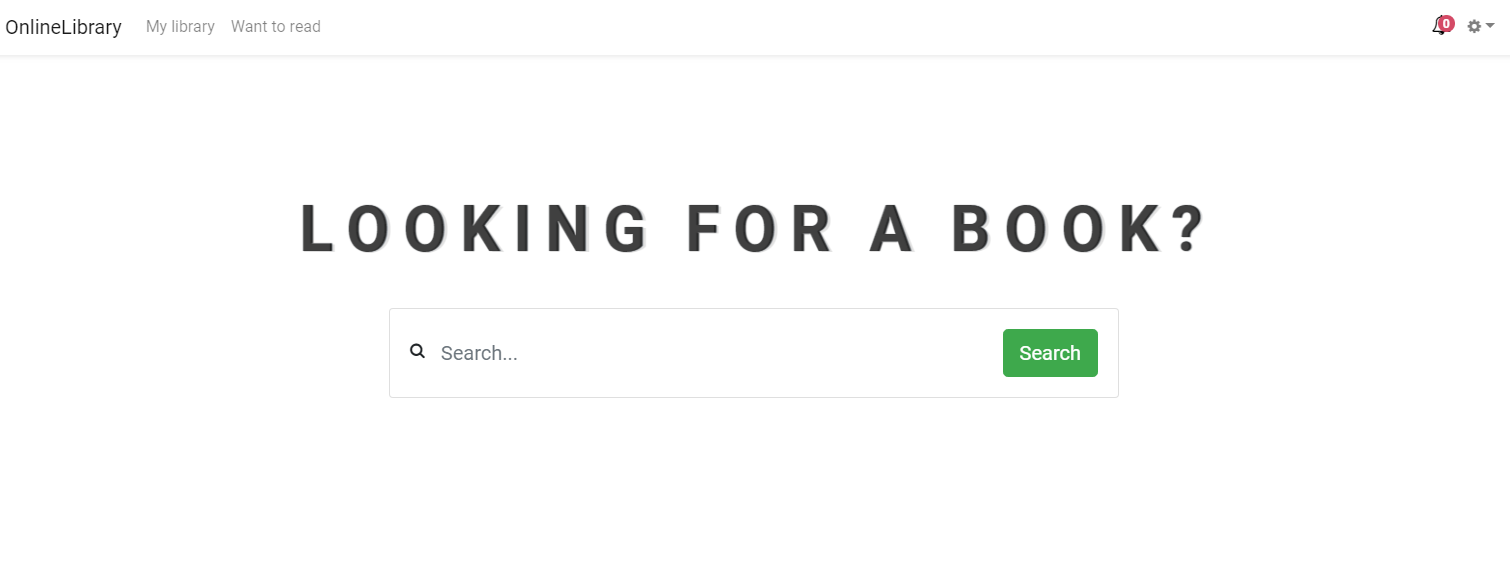
Meniu de navigație după intrarea în cont:



## Home

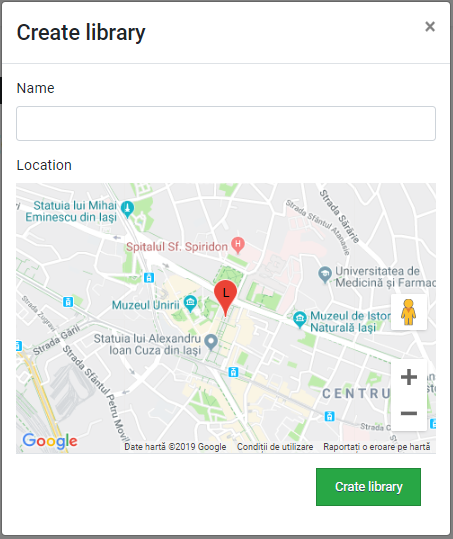
În această componentă, utilizatorul va putea să caute o carte după titlu, serviciu ce va returna prin intermediul Spring-ului o listă de cărți din baza de date. Fiecare carte primită din baza de date are link către pagina cărții respective.

Este componenta principală a aplicației, utilizatorul fiind redirecționat aici imediat ce intrarea în cont a fost cu succes.

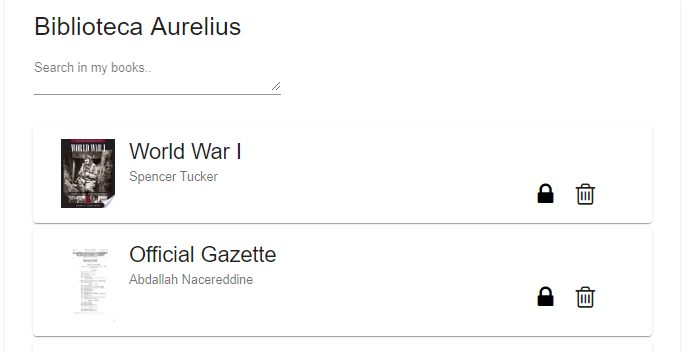


## My library

Accesând butonul „My Library” din meniul principal de navigare, utilizatorul va fi redirecționat pe această componentă. Inițial utilizatorul nu are decât posibilitatea de a crea o bibliotecă. Accesând butonul „Add library” acesta va putea să-și creeze propria bibliotecă, completând formularul de adăugare nume bibliotecă și locația acesteia. Locația este dată de locația curentă a utilizatorului. Prin opțiunea de drag and drop, utilizatorul își poate schimba locația bibliotecii, în cazul în care aceasta nu este dată exactă. Acest lucru se face prin intermediul unei hărți Google Maps.



În momentul creării de bibliotecă, se vor încărca alte două componente: în partea stângă un meniu de tip sidebar, iar în partea dreaptă o listă cu cărțile din bibliotecă. Fiecare carte din bibliotecă având opțiunea de a fi ștearsă, sau blocată pentru alte căutări. De asemenea dacă dăm click pe carte, Angular va încărca componenta ce ține de afișarea cărții respective.

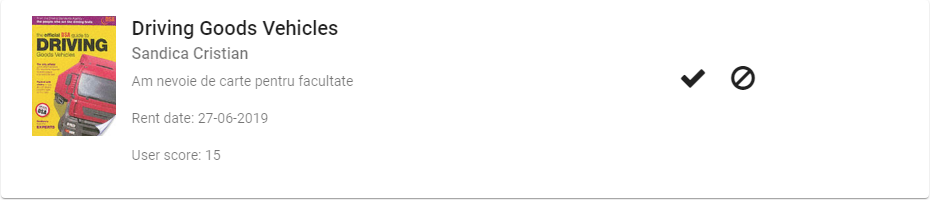


Meniul de tip sidebar din partea stângă este reprezentată de alte componente ce țin de administrarea bibliotecii, iar butoanele pe care aceasta le conține au următoarele funcționalități:

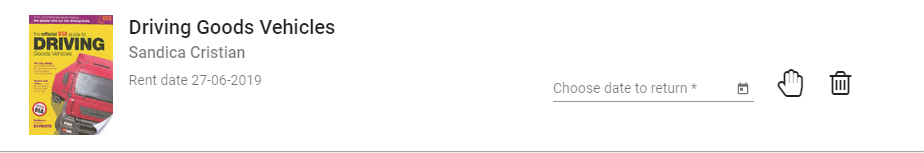
* Add a book
* accesând acest buton, o să apară o fereastră de tip modal ce conține un input pentru introducerea titlului cărții, pe care dorim să o adăugăm în bibliotecă
* odată completat input-ul, se execută automat serviciul de căutare în Google Books, acesta afișându-ne primele 5 cărți ce apar în sugestii
* carte odată selectată din listă, poate fi adăugată în bibliotecă accesând butonul „Add selected book”
* în urma acestui proces, carea nou adăugată va apărea în componenta bibliotecii, aceasta apărând în căutări, fiind disponibilă pentru a fi împrumutată



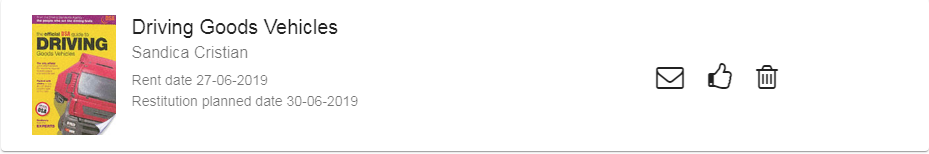
* Confirmations
* accesând acest buton, în partea dreaptă a paginii se va încarca componenta ce se ocupă cu administrarea cererilor de împrumut de carte
* aici sunt afișate toate cererile de la clienți, administratorul bibliotecii având ca opțiune acceptarea cererii de împrumut, sau respingerea acesteia. Pentru ambele acțiuni, utilizatorul va primi o notificare cu un mesaj, în funcție de acțiunea aleasă de administratorul bibliotecii
* administratorul bibliotecii i se vor afișa într-o listă cărțile ce doresc a fi împrumutate, utilizatorii care o doresc, scorul acestora și data la care cititorul dorește să o ridice



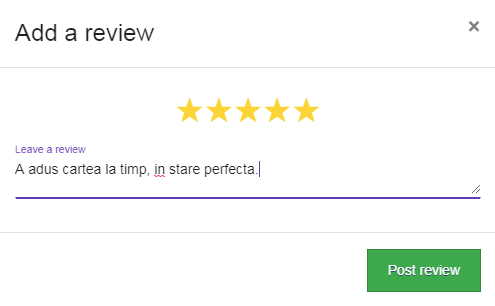
* Rents
* accesând acest buton, în partea dreaptă a paginii se va încărca componenta ce se ocupă cu împrumutul de carte, odată ce a fost aprobată cererea
* aici sunt afișate toate cărțile cu utilizatorii în parte, administratorul bibliotecii având ca opțiune ștergerea împrumutului și setarea acestuia ca fiind complet cu cartea ridicată de cititor. Înainte de a seta cartea ca fiind ridicată, acesta alege împreună cu cititorul data returnării cărții



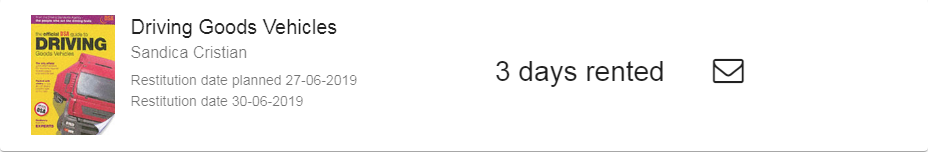
* dacă cartea a fost setată ca fiind ridicată, acțiunile administratorului de bibliotecă se schimbă în setarea cărții ca fiind adusă, trimiterea de email cititorului și ștergerea acesteia. Totodată administratorului de bibliotecă i se va afișa câte zile disponibile mai are utilizatorul, până o aduce



* odată adusă cartea înapoi la bibliotecă, administratorul acesteia trebuie să apese butonul de confirmare a aducerii cărții. Odată accesat, se va deschide o fereastră modală unde acesta poate să îi scrie utilizatorului o recenzie, recenzie ce o să apară pe pagina de informații a acestuia, iar scorul va fi actualizat automat



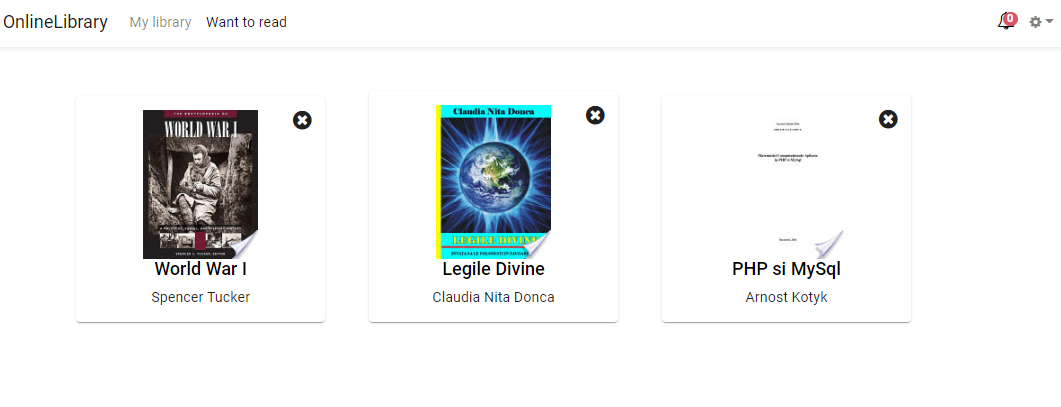
* History
* accesând acest buton, în partea dreaptă a paginii se va încărca componenta ce se ocupă cu administrarea tuturor împrumuturilor finalizate în cadrul bibliotecii respective
* administratorul bibliotecii poate să trimită email utilizatorilor înregistrați în istoric. De asemenea acesta poate să vadă perioada de timp în care a fost împrumutată cartea respectivă



* Import books
* accesând acest buton, administratorul bibliotecii poate adăuga mai multe cărți deodată
* acesta trebuie să urce un fișier csv ce conține titlul și autorul cărții

## Want to read

Accesând butonul „Want to read” din meniul principal de navigare, se va încărca componenta ce afișează utilizatorului lista lui de cărți adăugate pentru a fi citite mai târziu. Butonul adăugării unei cărți în această listă se află pe pagina de informații a unei cărți, fiecare utilizator având o astfel de listă. Acesta o poate elimina prin apăsarea butonului de X, situat deasupra cărții.

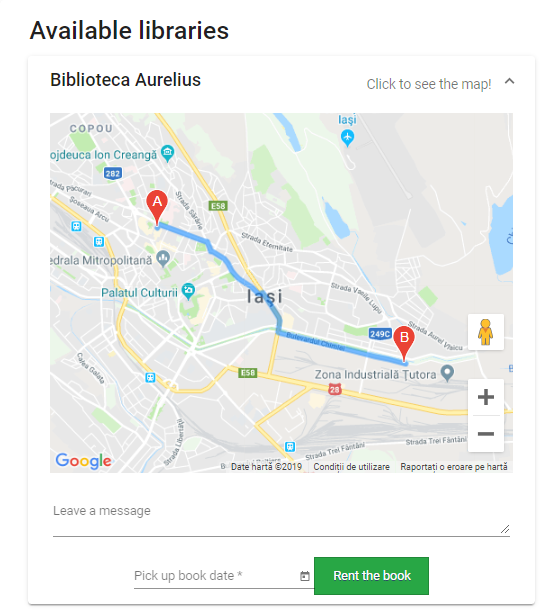


## Book

Această componentă se ocupă cu informațiile afișate despre o carte. Aceasta conține titlul cărții, imaginea de copertă a acesteia, autorul, o scurtă descriere și o listă de recenzii. Recenziile pot fi adăugate de orice utilizator.

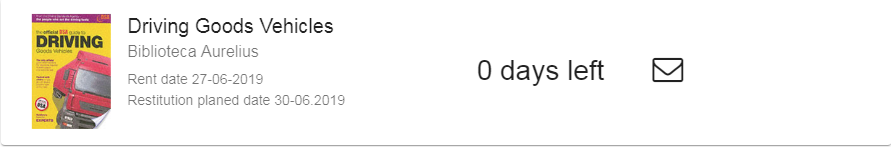


În această componentă mai sunt situate și butoanele „Add want to read” și „Rent the book”. Apăsarea primului buton adaugă cartea în listă de cărți pe care cititorul dorește să le citească, iar apăsarea butonului al doilea încarcă în pagină o componentă ce afișează toate bibliotecile ce dețin cartea respectivă, și este disponibilă. Utilizatorul poate să aleagă din multitudinea de biblioteci, pe cea care este cea mai apropiată. La alegerea unei biblioteci, acesta poate să vadă ruta cea mai apropiată, de la locația sa curentă până la locația bibliotecii. Pentru a continua procesul de împrumut de carte, utilizatorul trebuie să selecteze data la care acesta dorește să ridice cartea, iar opțional un mesaj. În momentul apăsării butonului „Rent the book”, cererea sa este trimisă către administratorul bibliotecii, acesta reușind să o vadă în notificări.



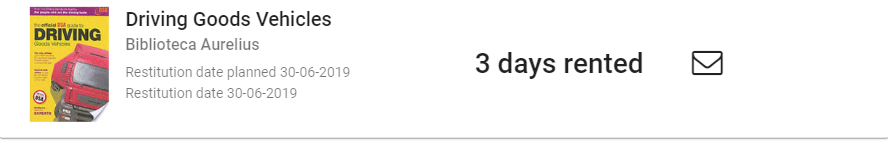
## Rented books

Această componentă se găsește în meniul de navigare principal, în dropdown-ul din partea dreaptă. Aici utilizatorul poate să-și vadă cărțile ce sunt în prezent împrumutate, cu tot cu informațiile bibliotecii. Componenta este folosită pentru a-l ajuta pe cititor să țină evidența cărților împrumutate. De asemenea aici poate să vadă și câte zile mai are rămase.



## Returned books

La fel ca și „Rented books”, această componentă se găsește tot în dropdown-ul din partea dreaptă a meniului principal. Aici utilizatorul își poate vedea istoricul de cărți împrumutate.



La setarea unei cărți ca fiind adusă, de către administratorul bibliotecii respective, utilizatorul ce a împrumutat cartea, va primi o notificare. Apăsând pe aceasta, va putea să ofere o recenzie bibliotecii de la care a împrumutat cartea.

# Concluziile lucrării

Toți am dat peste un moment în viața noastră, când aveam nevoie de o carte. Primul lucru pe care îl facem este să căutăm cartea respectivă în bibliotecile cunoscute din apropierea noastră. De multe ori biblioteca nu are cartea respectivă, acest lucru aflându-se abia în momentul în care o cerem. Astfel pierdem foarte mult timp cu căutarea cărți. Pentru rezolvarea acestei probleme, utilizatorii ar putea accesa pe internet o aplicație web, pe care să caute cartea dorită. Aplicația le va afișa lista cu biblioteci în care este găsită cartea. De asemenea drumul cel mai scurt până la biblioteca respectivă.

În opinia mea, societatea noastră are nevoie de o astfel de aplicație web. Având în vedere că lucrurile se mută încetul cu încetul pe internet, iar procesul unui împrumut de carte dintre un cititor și o bibliotecă a rămas același de sute de ani, OnlineLibrary este aplicația perfectă pentru îmbunătățirea serviciilor de împrumut de carte. Aceasta ar ajuta extrem de mult societatea, cititorii reușind să-și găsească cărțile mult dorite doar printr-o simplă căutare. Oricine poate să-și creeze o bibliotecă, în care să-și urce cărțile pe care dorește să le împrumute altora. Acest lucru face ca anumite cărți ce sunt mai rare, să fie găsite mai ușor. Utilizatorul poate găsi cartea dorită chiar și la doi pași de acesta, dacă există o bibliotecă creată de un altul din apropiere.

Această aplicație este benefică pentru noi oamenii, și ar trebui pusă în aplicare cât mai rapid. Prea mult timp am stat fără să facem nimic în acest caz. Ne-am mulțumit cu căutarea cărții într-o bibliotecă pe care o cunoaștem. A venit timpul să schimbăm lucrurile și să o punem în aplicare. Toți oamenii vor fi de acord cu o astfel de aplicație, și chiar bucuroși dacă aceasta va fi în piață. Toți căutăm soluții cât mai rapide pentru toate probleme noastre, aplicația web OnlineLibrary fiind perfectă pentru problema închirierii unei cărți.

## Direcții viitoare de cercetare

Mai jos am descris o listă cu viitoare funcționalități și îmbunătățiri ale aplicației:

### Mobile

În prezent aplicația este doar pentru web. Utilizatorii trebuie să o acceseze prin intermediul unui browser. Pe viitor aceasta ar putea avea și o versiune mobile. Utilizatorilor le-ar fi mult mai ușor să urmărească traseul către bibliotecă de pe telefon.

### Book of the day

Acest serviciu arată în Home cartea zilei. Cartea care a fost împrumutată cel mai mult în ziua respectivă. De asemenea ar putea fi și „Book of the month” și „Book of the year”.

### Chat live

Un chat live ar fi foarte bun pentru această aplicație. Momentan utilizatorii și bibliotecarii pot să comunice între ei prin intermediul email-urilor. Un chat live între administratorul bibliotecii și cititor ar fi mult mai rapid.

### Roluri

În prezent nu există decât administratorul unei biblioteci și utilizator. Rolurile ar fi o funcționalitate în plus adusă aplicației foarte benefică. Acest serviciu ar putea adăuga noi administratori la o bibliotecă cu anumite permisiuni: confirmare împrumut, ștergere împrumut, ștergere carte etc.

# Bibliografie

* <http://iasifun.ziaruldeiasi.ro/biblioteci/22266/>
* <http://www.bcu-iasi.ro/>
* <https://www.bibmet.ro/>
* <https://mariadb.org/about/>
* <https://mariadb.com/about-us/>
* <https://maven.apache.org/what-is-maven.html>
* <https://www.thoughtco.com/what-is-java-2034117>
* <https://www.programiz.com/java-programming>
* <http://ea.md/cartile-tiparite-sau-cele-electronice/>
* <https://www.google.com/googlebooks/about/>
* <https://www.bcucluj.ro/>
* <https://www.bookrix.com>
* <https://towardsdatascience.com/visualize-programming-language-popularity-using-tiobeindexpy-f82c5a96400d>
* <https://angular.io/features>
* <https://community.sensenet.com/blog/2017/08/09/why-jwt>
* <https://docs.spring.io/spring/docs/3.0.0.M4/reference/html/ch01s02.html>
* <https://docs.spring.io/autorepo/docs/spring-framework/5.0.0.M1/spring-framework-reference/pdf/spring-framework-reference.pdf>
* <https://www.java.com/en/download/faq/whatis_java.xml>
* <https://howtodoinjava.com/java/basics/what-is-java-programming-language/>
* <https://developers.google.com/maps/documentation/directions/start>
* <https://developers.google.com/maps/documentation/>