Cuprins

[Introducere 2](#_Toc483916835)

[Motivaţie 2](#_Toc483916836)

[Context 2](#_Toc483916837)

[Cerinţe funcţionale 2](#_Toc483916838)

[Abordare tehnică 3](#_Toc483916839)

[Baza de date 3](#_Toc483916840)

[Backend 3](#_Toc483916841)

[Frontend 3](#_Toc483916842)

[Framework-uri 3](#_Toc483916843)

[Entity Framework Core 3](#_Toc483916844)

[Bootstrap 3](#_Toc483916845)

[Selenium 4](#_Toc483916846)

[Contribuţii 4](#_Toc483916847)

[Proiectare 5](#_Toc483916848)

[Arhitectura soluției 5](#_Toc483916849)

[Modelarea datelor 5](#_Toc483916850)

[Protocoale de comunicare client – server 5](#_Toc483916851)

[Interfața cu utilizatorul 5](#_Toc483916852)

[Implementare 5](#_Toc483916853)

[Manual de utilizare 5](#_Toc483916854)

[Referințe 5](#_Toc483916855)

# Introducere

## Motivaţie

În această lucrare voi propune o soluţie pentru dezvoltarea unei aplicaţii web, atât pentru client cât şi pentru server, care să ofere informaţii utilizatorilor despre cărţi pe care doresc să le împrumute şi nu le deţin sau nu le pot procura de la o bibliotecă sau librărie.

De asemenea, consider că ar putea fi considerată şi o aplicaţie socială deoarece utilizatorii vor avea posibilitatea de a cunoaşte persoane noi şi de a comunica cu acestea prin intermediul chat-ului din interiorul aplicaţiei.

## Context

Dat fiind faptul că, în momentul în care cineva doreşte să împrumute o carte are nevoie de permis la o bibliotecă, permis care nu se oferă în orice condiţii, spre exemplu, anumite biblioteci sunt doar universitare; oferă cărţi spre împrumut doar studenţilor/profesorilor sau anumite cărţi le poţi citi doar la sala de lectură. La o căutare pe google se poate observa că există destul de puţine asemenea aplicaţii, cea mai cunoscută fiind Bookster (o bibliotecă pentru companii din România).

Conform ultimelor statistici oficiale[[1]](#footnote-1) din anul 2011 interesul pentru lectură este destul de scăzut. De asemenea, datorită numărului continuu în creştere de utilizatori în mediul online[[2]](#footnote-2) consider că o asemenea aplicaţie online ar putea fi benefică.

## Cerinţe funcţionale

Pentru a avea un impact pozitiv dar şi pentru eventualitatea intrării pe piaţă a aplicaţiei, ea va trebui să respecte următoarele cerinţe funcţionale:

* Utilizatorul va avea posibilitatea de a-şi crea un cont nou
* Utilizatorul va putea folosi ca şi metode de autentificare Google, Facebook sau contul creat la inregistrare
* Utilizatorul va putea să caute cartea dorită în funcţie de anumite opţiuni: titlu, numele autorului, anul publicării cărţii
* Utilizatorul va putea să-şi editeze informaţiile personale
* Utilizatorul va putea să-şi managerieze cărţile pe care le deţine
* Utilizatorul va fi putea fi contactat prin email sau prin intermediul chat-ului
* Utilizatorul va putea să vadă ce utilizatori mai sunt prezenţi pe site şi cărţile pe care aceştia le deţin
* Utilizatorul va avea posibilitatea de a se loga/deloga

## Abordare tehnică

Baza de date

Pentru baza de date am ales soluţia oferită de cei de la Microsoft, şi anume, Microsoft SQL Server. Este o bază de date relaţională, care are capacitatea de stocare şi expunere a datelor pe baza unor cereri primite de la solicitant şi poate fi rulată pe acelaşi computer sau pe un computer din reţea.

### Backend

Pentru partea de backend voi folosit C#, un limbaj de programare functional, generic, orientat obiect, imperativ, declarativ. De asemenea, această componentă va face legătura între baza de date şi frontend. Tot aici se vor transmite notificările şi mesajele de la un utilizator către un altul.

### Frontend

La nivel de frontend se vor folosi AngularJS, framework de Javascript (limbaj de nivel înalt şi interpretat), care foloseşte desing pattern-ul MVC. De asemenea, AngularJS oferă posibilitatea de a crea elemente HTML proprii, altele decât cele specifice limbajului. Structura de bază a acestui framework este bazată pe servicii, controllere şi directive.

De asemenea, pentru a face partea vizuală cât mai plăcută pentru utilizatorul final se va folosi CSS, un standard pentru formatarea elementelor unui document HTML. Stilurile se pot atașa elementelor HTML prin intermediul unor fișiere externe sau în cadrul documentului prin elementul <style> și/sau atributul style. Cu alte cuvinte, CSS este canditatul ideal pentru partea de web design.

Pentru crearea efectivă a paginilor se va folosi HTML. Este un [limbaj de marcare](https://ro.wikipedia.org/wiki/Limbaj_de_marcare" \o "Limbaj de marcare) utilizat pentru crearea paginilor web ce pot fi afișate într-un [browser](https://ro.wikipedia.org/wiki/Browser) (sau navigator). Scopul HTML este mai degrabă prezentarea informațiilor – paragrafe, fonturi, tabele ș.a.m.d. – decât descrierea semanticii documentului. Specificațiile HTML sunt dictate de [World Wide Web Consortium](https://ro.wikipedia.org/wiki/World_Wide_Web_Consortium) (W3C).

### Framework-uri

#### Entity Framework Core

Este o tehnologie folosită pentru a accesa o bază de date, o versiune independentă de platforma de dezvoltare, extensibilă şi mai uşoară comparativ cu Entity Framework. EF Core este un mapper de obiecte relaţionale (ORM, mai pe scurt), care le oferă dezvoltatorilor de .NET posibilitatea de a lucra cu o bază de date folosind obiecte .NET. Totodată, elimină nevoia de a scrie cod pentru a accesa baza de date, cod pe care în mod normal un programator oricum ar fi nevoit să-l scrie. EF Core are şi suport pentru mai multe motoare de baze de date.

#### Bootstrap

Este un framework pentru web pe partea de frontend gratuit şi open-source, care ajută la realizarea design-ului pentru site-uri sau aplicaţii web. Conţine anumite şabloane de design - HTML şi CSS - pentru elementele vizibile în interfaţa utilizatorului dar şi opţional, extensii Javascript. Spre deosebire de multe alte framework-uri web, se concentrează doar pe dezvoltarea frontend-ului.

#### Selenium

Este un framework de testare software portabil folosit pentru aplicații de tip web. Oferă o unealtă pentru înregistrare/playback pentru teste fără a învăța un limbaj de testare specific. De asemenea, oferă posibilitatea de a scrie teste în diverse limbaje de programare cum ar fi: C#, Java, PHP, Python, Ruby ș.a. . Testele pot rula pe browserele modern. Dezvoltarea lor se poate realiza atât pe Windows cât și pe Linux sau OS X. Un alt avantaj este că este open-source și poate fi descărcat gratuit.

# Contribuţii

Lucrarea este structurată în patru capitole mari, fiecare din ele tratând, pe rând, procesul de dezvoltare din cele două perspective principale(client şi server). Se va oferi motivaţia pentru anumite decizii luate pe parcursul proceseului de dezvoltare dar şi detalii relativ la implementare sau eventuale diferenţe care există comparând elemente similare. De asemenea, se vor oferi detalii despre asigurarea calităţii produsului şi a funcţionalităţii pe termen scurt şi lung. Cele patru capitole sunt enumerate mai jos:

* Capitolul 1: Dezvoltarea aplicației client
* Capitolul 2: Dezvoltarea aplicației server
* Capitolul 3: Baza de date
* Capitolul 4: Testarea funcţionalităţilor şi asigurarea calităţii

În cadrul dezvoltării aplicaţiei client s-au avut în vedere aspecte precum:

* Să fie user-friendly şi intuitivă
* Să fie rapidă
* Să fie scalabilă în funcţie de dispozitivul folosit
* Utilizarea de design-pattern-uri pentru separarea responsabilităţilor

Pe partea de server s-a avut în vedere ca aplicaţia să facă faţă unui volum consistent de utilizatori, să fie sigură și să păstreze confidențialitatea informațiilor despre utilizatori sau cărți:

* Folosirea autentificării și autorizării
* Cererile procesate asincron sau sincron
* Utilizarea principiilor SOLID
* Utilizarea de design-pattern-uri pentru separarea responsabilităţilor

În ceea ce priveşte baza de date s-a dorit a fi o bază de date de dimensiuni reduse dar robustă:

* Legăturile între tabele folosind chei străine
* Utilizarea de chei primare compuse din alte chei străine

Testarea funţionalităţii şi asigurarea calităţii are ca scopuri esenţiale asigurarea faptului că aplicaţia respectă cerinţele funcţionale şi posibilitatea de dezvoltare, îmbunătăţire, aspecte ce pot fi aduse pe viitor:

* Crearea de unit teste pentru fiecare modul şi layer în parte
* Crearea de teste de integrare pentru verificarea funcţionalităţii modulelor împreună
* Crearea de teste automate pentru verificarea funcţionalităţii din punctul de vedere al utilizatorului

# Proiectare

## Arhitectura soluției

## Modelarea datelor

## Protocoale de comunicare client – server

## Interfața cu utilizatorul

# Implementare

# Manual de utilizare

# Referințe

1. Obiceiuri lectură - http://www.ires.com.ro/articol/172/obiceiurile-de-lectura-ale-romanilor [↑](#footnote-ref-1)
2. Navigare pe Internet - http://ec.europa.eu/eurostat/statistics-explained/index.php/Information\_society\_statistics\_-\_households\_and\_individuals/ro [↑](#footnote-ref-2)