Külkereskedelmi Technikum

**Byte Brigade**

**Dokumentáció**

**BOOKHUB**

**online használtkönyv-csere platform**



Készítették:

Balázs Réka, Gerecs Diána, Csajbók-Reményi László

Budapest

2024

**Tartalomjegyzék**

1.Bevezetés**1**

1.1 Témaválasztás indoklása2

1.2 AI által generált képek használata3

1.3 Hasonló weboldal3

1.4 A project jelenlegi állapota és a jövője3

2. Fejlesztői dokumentáció5

2.1Fejlesztői környezet5

2.2 Fejlesztéshez használt operációs rendszerek, böngészők és nyelvek6

2.3 Adatbázis9

2.4 Változók frontendben és backendben11

2.5 Osztályok elnevezése a Bulma keretrendszeréhez igazodva12

2.6 Backend megvalósítása SpringBoot alkalmazással17

2.7 User regisztrációja és jogosultságai18

2.8 Fejlesztési lehetőségek19

3 Tesztelés13

4 Felhasználói dokumentáció18

4.1 Böngészés a honlapon18

4.2 Regiztráció a honlapon19

4.3 Elérhető funkciók a bejelentkezés után19

4.4 Könyv felajánlása21

4.5 Könyv lefoglallása22

4.6 Hibajelzések jelentése23

5. Összefoglalás24

**1 Bevezetés**

**1.1 Témaválasztás**

*„A lélek hőmérője: a könyv.  Mondd, meg mit olvasol és megmondom: mi vagy. Megmondom mi az akaratod, célod, vágyad, álmod.” (Juhász Gyula)*

A Byte Brigade által megálmodott és megalkotott Bookhub vállalat weblapja alkalmas arra, hogy azokat az embereket kösse össze, akik szeretnek olvasni, és szívesen adják tovább azt, amit már elolvastak, vagy keresgélnek azopk között a könyvek között, amik elérhetőek. Ezáltal ismeretségek születhetnek azonos érdeklődésű emberek között. Az azonos érdeklődésű emberek ()

Mindeközben felhasználóink felelősséggel viseltetnek a környezet felé, tisztában vannak azzal, hogy rajtunk múlik, mit adunk tovább a gyermekeinknek. Fejlesztőként ehhez azzal járulhatunk hozzá, hogy lehetőséget biztosítunk arra, hogy a saját könyveiket szabadon felajánlják, cserélgessék egymás között, mindezt olyan rendezett körülmények között, amely egyszerre védi a felhasználók adatait, és minél inkább csökkenti a lehetőségét annak, hogy valaki visszaéljen a felajánlók jóindulatával.

Szeretnénk elérni, hogy amikor valaki tovább szeretné adni a könyveit bármilyen oknál fogva, akkor ne ütközzön abba az akadályba, hogy nem tudja kinek, és emiatt egy jó állapotú könyv a hulladékban végzi, pedig más szívesen olvasná, csak nincs rá módja, hogy megvegye., esetleg nem is tud róla, hogy másnak a birtokában van, de már nincsen szüksége rá. Mindezeken túl olyan könyveket is megóvhatnánk a kidobástól, amiket már csak használtan lehet megszerezni egy-egy antikváriumban, pedig az oktatás során egyszerűen nélkülözhetetlen.

Nagy mennyiségű könyvek cseréjét az adminoknak lehetne jelezni, az adminisztrációs terhek csökkentése érdekében. Leginkább felsőfokú oktatási intézményekben jellemző, hogy a tantárgy oktatási színvonalát jelentősen csökkentheti egy-egy már el nem érhető könyv, amit a tanulók csak addig használnak, amíg meg nem szerzik az oklevelet.

**1.2 AI által generált képek használata**

A tervezgetés kezdeti fázisában elkezdtünk AI által generált képeket gyártani Hamer kialakult az egyetértés abban, hogy ezek megfelelő alapot fognak adni a vizuális megjelenéshez.

A téma alapját a mackók adták, mégpedig azért, mert a fejlesztés korai szakaszában Gerecs Diána fejlesztő javaslatára bevezettük a rendszeres gumicukor fogyasztást, ami többnyire mackó formájú volt. Így aztán a mackók uralták a témákat a képeknél. Külön-külön generáltuk a képeket Csajbók-Reményik László és Balázs Réka. Az online találkozók alkalmával közösen döntöttük el, hogy melyik felületre melyik kép kerülhet.

A képek generálása közben sokat mmegtanultunk arról, hogy hogyan lehet és érdemes kiadni az utasításokat olyan módon a mesterséges intelligenciának, hogy az olyan képeket generáljon, amik közel állnak az elképzeléseinkhez. A végeredmény magáért beszél, azok a külső szemlélők, akik megtekintették a weblapot ezekkel a képekkel egyöntetűen elismerően nyilatkoztak róluk.

**1.3 Hasonló weboldal**

Jelenleg hasonló működő weboldalt nem találtunk, ami Magyarországon működik, ingyenesen elérhető, és használható. Azonban több olyan fórumot is találtunk, ahol igény mutatkozik arra, hogy jó lenne egy ilyen szolgáltatás, ahol a használt könyveket cserélhetik a regisztrált felhasználók.

Több nagyvárosban vannak kihelyezett ingyenes könyvmegállók, és ezeket szívesen használják az emberek. Ide azonban személyesen kell elmenni, és azok számára, akik közelében nincsen ilyen kihelyezve nem jelent megoldást. Arról nem beszélve, hogy sokkal egyszerűbb úgy keresni a tartalmak között, hogy csak akkorveszi fel valaki a kapcsolatpt a felajánlóval, ha az ő számára érdekes könyv van a kínálatban. A kihelyeztt könyvek esetében azonban ez a kényelmesebb áttekintése a választéknak egyszerűen nem lehetséges.

**1.4 A projekt jelenlegi állapota és jövője**

A projekt jelenlegi állapotában csak egy elképzelt vállalat egy jól felépített szerethető weboldallal, olyan képekkel, amiket mindenki szívesen nézeget, és aminek sokan örülnének, ha megvalósulna. Egyelőre nem kerestünk hozzá támogatót, és nem tervezzük azt, hogy ezt valóban életre keltsük, azonban nem is zárjuk ki teljesen ennek az lehetőségét.

A korlátokat egyértelműen az anyagi forrás megteremtése okozza, illetve az, hogy jelenleg egyikünk sem dolgozik főállásban fejlesztőként, így nehézséget okozna a fenntartása és üzemeltetése annak ellenére, hogy akiknek megmutattuk tetszetősnek találták, és látnának benne fantáziát.

**2. Fejlesztői dokumentáció**

**2.1 Fejlesztői környezet (frontend és backend)**

A hozzáférési lehetőségek úgy vannak kialakítva, hogy minden admin hozzáfér a forráskódokhoz, aki meghívást kapott a közös Github tárolóhoz. A fejlesztés közben verzió követést alkalmaztunk, az aktuális fejlesztés feltöltése előtt letöltöttük a további változtatásokat, ezen kívül rendszeresen tartottunk online megbeszéléseket zoom-on.

Először a Github-ról kell letölteni a program aktuális állapotát. Az Angulár 16.0.1-es verziót használtuk, ehhez igazodjon a többi letöltött program kompatibilis verziója.

**2.2 Fejlesztéshez használt operációs rendszerek:**

A képen Színesség, Téglalap, képernyőkép, tér látható

Automatikusan generált leírás Windows

A képen fekete, kör, fekete-fehér, csillagászat látható

Automatikusan generált leírás Debian

**Fejlesztéshez használt böngészők:**

 Edge

A képen kör, Színesség, Grafika, Grafikus tervezés látható

Automatikusan generált leírás Chrome

A képen kör, Színesség, gömb, bolygó látható

Automatikusan generált leírás Chromium

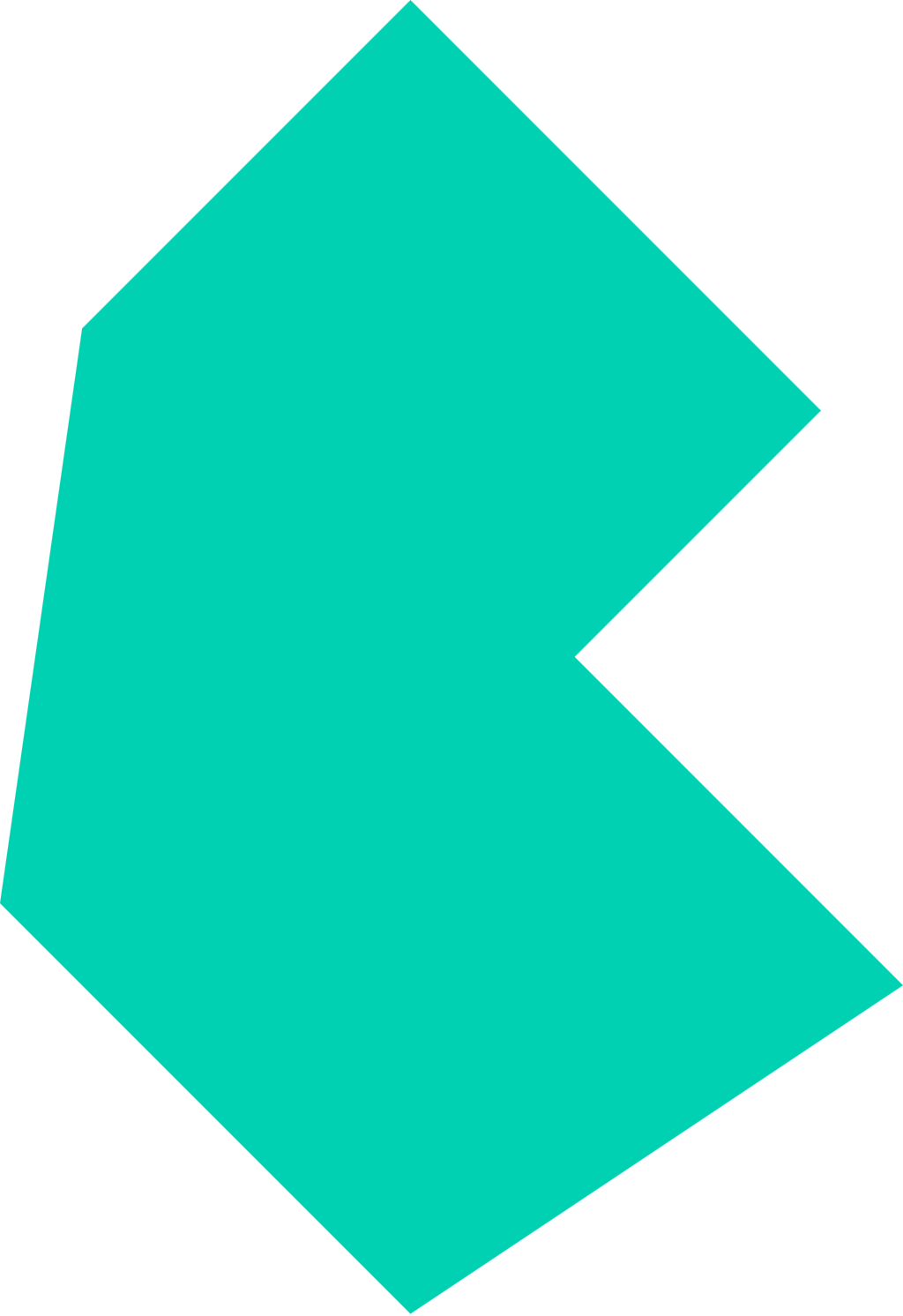
**Fejlesztéshez használt programok:**

A képen szimbólum, Betűtípus, Grafika, képernyőkép látható

Automatikusan generált leírás IntelliJ

A képen szimbólum látható

Automatikusan generált leírás Angular 16.0.1.

 Bulma keretrendszer

A képen szimbólum, Grafika, tervezés látható

Automatikusan generált leírás XAMPP

A képen szöveg, embléma, Betűtípus, tervezés látható

Automatikusan generált leírás.. PHPMyAdmin

A képen óra, multimédia, tervezés látható

Automatikusan generált leírás Terminátor

 Chocolatey

**Fejlesztéshez használt programozási nyelvek:**

**A képen fekete, sötétség látható

Automatikusan generált leírás HTML**

**A képen tervezés, művészet látható

Automatikusan generált leírás alacsony megbízhatósággal JAVA**

**A képen Grafika, Színesség, zöld látható

Automatikusan generált leírás CSS**

A képen embléma, Betűtípus, szimbólum, szöveg látható

Automatikusan generált leírás **PHP**

 JavaScript

**Fejlesztéshez használt eszközök:**

Asztali számítógép,

laptop

A fejlesztésnél kiemelt fontosságú volt a használt programok kijelölt verzióinak használata. Célszerű minden automatikus frissítés lehetőséget kikapcsolni a gépeken, illetve nem engedélyezni az Angulár a SpringBoot részére az automatikus frissítést vagy feltöltést a GitHub-ra. A SpringBoot ezt a project indításakor megkérdezi, akkor kell válaszként azt a lehetőséget választani, hogy programozás közben döntjük el, mikor akarjuk feltölteni a változásokat a GitHub repositoryba.

A vizuális megjelenítésnél következetesen AI által generált képeket használtunk, célszerű esetleges bővítés esetén ehhez következetesen ragaszkodni.

A Bulma keretrendszerben használt card elemek biztosítják a képek az egységes megjelenését a honlapon.

A honlap különböző komponensekből áll, minden egyes komponensnek van html, css, spec,ts és ts fálja. Ezeket a fájlokat az Angulár hozza létre a komponens létehozásával együtt.

Az egyes komponensekhez tartozó html fájlok tartalmazzák a komponensek felépítését , a css fájlokban lehet a megjelenítésüket módosítani. A ts fájlok tartalmazzák a teszteket.

Az angulárhoz a Bulma keretrendszert telepítettük, ezekkel formáztuk az oldalakat. A html fájlokban az egyes elemekhez tartozó osztályok elnevezése annak megfelelő, hogy milyen megjelenést szántunk nekik.

**2.3 Adatbázis**

A *Bookuser* adattábla, amely tárolja a regisztrál felhasználók adatait. a ki-és bejelentkezéshez szükséges. Elnevezése egyértelműen utal arra, hogy a regisztrált felshasználó egyben könyvtulajdonos, hiszen vagy felajánl könyvet, vagy nagy valószínűséggel éppen foglalni fog egyet és ebből a célból regisztrál.

Az első mező az *id* “AUTO\_INCREMENT” tulajdonsággal rendelkezik, automatikusan jön létre a regisztrációkor. Hossza 11 karakter lehet, ez bőven elegendő az esetleges felhasználók adatainak tárolásához. A tulajdonság *int* váltózóként van tárolva, mely az integer rövidítése, egész számok tárolására szolgál.

Az int változóban tárolt adatok kis méretű adatok, kevés helyet igényelnek az adatbázisban. Az egyik legelterjedtebb változó az adatbázisokban, az adatbázis kezelő rendszerek támogatják a használatát. Nem ismétlődhet ez a tulajdonság, ugyanaz az *id* nem lesz kiosztva a felhasználó adatainak törlése után sem másik regisztrált felhasználó részére. Egyértelmű azonosítást tesz lehetővé, használható idegen kulcsként más adattáblával történő összekötéshez.

A *username* tulajdonság 50 karakter hosszú lehet, szabadon választott a felhasználó regisztrációjánál. Ez a hosszúság valószínűleg elég lesz egy átlagos felhasználónév megadásához. A felhasználó a felhasználónév és a jelszó megadásával tud belépni a honlapra, könyveket felajánlani és foglalni.

A password tulajdonság tárolja a regsiztrált felhasználó által választott jelszót, amely szintén szabadon választott, hossza maximum 255 karakter.

A *Books* adattábla a regisztrált felhasználók által felajánlott könyvek adatait tárolja. Az *id* tulajdonság automatikusan jön létre, annak beállítása illetve utólagos módosítása nem lehetséges. Az *owner\_id* tartalma azonos a bookusers tábla *id*-vel, elsődleges kulcsként funkcionál, biztosítja a két adattábla közötti kapcsolatot.

A képen szöveg, képernyőkép, szoftver, Számítógépes ikon látható

Automatikusan generált leírás

**2.4 Változók és mezők elnevezése a programban**

A frontendben és a backendben is a tiszta kódnak megfelelő változóneveket használtunk, amelyek egyértelműen utalnak az adott változó céljára és használatára.

Frontend deklarált változónevek:

* user (regisztrált felhasználó)
* name(regisztrált felhasználó neve)
* email(regisztrált felhasználó emailcíme)
* password(regisztrált felhasználó jelszava)
* county(regisztrált felhasználó által jelölt megye)

Backend deklarált változónevek:

* book (felajánlott könyv)
* id (könyv id-ja)
* title (könyv címe)
* author (könyv szerzője)
* publisher (felajánló regisztrált felhasználó)
* category (könyv kategóriája)
* county (regisztrált felhasználó által jelölt megye)
* quality (könyv állapota)

A regisztrációkor megadott adatokat tároljuk változókban, azoknak vizsgálata csak a formai követelményeknek való megfelelés szempontjából szükséges. Azok valódiságát semmilyen szempontból nem vizsgáljuk, tartalmukért a regisztrált felhasználó a felelős.

**2.5 Osztályok elnevezése a Bulma keretrendszer szerint**

Például az alábbi kódrészletben a container osztály használatával a div elem felveszi a konténer tulajdonságait. és a tartalom középre igazítva jelenik meg igazodva az oldal szélességéhez

A column class segítségével a "has-text-primary-dark" egy osztálynevet jelöl, amelyet a Bulma CSS keretrendszer használ. Ez a stílus beállítja a szöveg színét sötét árnyalatú kékre. A Bulma dokumentációban kereshetető ki a megfelelő színárnyalat, amit látni szeretnénk a megjelenő honlapon. A „columns” class hozzáadásával az alárendelt div elem a Bulma grid rendszerének megfelelően jelenik meg, ami sokkal esztétikusabb megjelenítést tesz lehetővé az eredeti html kódhoz képest egy egyszerű kódmódosítással.

A margin értéket sem szükséges külön beállítani a css kódban, az mt-2 class hozzáadásával margin-top értéket ad az oszlopoknak, így a felső margót növelve beállítja a megfelelő távolságot a honlapon megjelenő elemek között. A column class az oszlopok egymás mellé rendezésében segít.

A card class kártyaszerű megjelenést ad a benne elhelyezett elemeknek. Ez a kártya szerű megjelenés leegyszerűsítette a képek megjelenését és formázását az oldalon, egységes megjelenést tett lehetővé.

Az alábbi szín a következő színkódot jelöli a táblázat szerint: "hsl(171, 100%, 29%)", amely a HSL (Hue, Saturation, Lightness) színmodell alapján határozza meg a szín árnyalatát, telítettségét és világosságát. Ebben az esetben a szín egy kék árnyalata.

A title class használatával a paragraph elem kep kiemelt megjelenést további formázás nélkül is.

<div class="container">

  <div class="columns mt-2">

    <div class="column">

      <div class="card has-background-primary-light">

        <div class="card-content">

          <p class="title has-text-primary-dark has-text-primary-dark">

            "Olvass, cserélj, játssz."

          </p>

**2.6 Backend megvalósítása SpringBoot alkalmazással**

A backend a SpringBoot alkalmazásban lett megvalósítva. A SpringBoot használata nagyban egyszerűsítette a backend megvalósítását beépített mechanizmusok segítségével. Ezeknek a funkciónak az eléréséhez importokat kell az egyes fájlok elején használni.

A program belépési pontja az alábbiakban található ***BackendApplication.java*** fájl. A *public static void main(String[] args* metódus van benne definiálva, amely a program végrehajtásának kezdetén indul el. Ez a metódus hívja meg a Spring Boot alkalmazás indításához szükséges *SpringApplication.run(BackEndApplication.class, args)* függvényt.

A fájl tartalmazza továbbá az *@SpringBootApplication* annotációt is, ami azt jelzi, hogy ez a fájl a Spring Boot alkalmazás fő konfigurációs osztálya. Az annotáció kombinálja a *@Configuration, @EnableAutoConfiguration, és @ComponentScan* annotációkat, ezáltal automatikusan konfigurálja a Spring alkalmazást és betölti az összes szükséges beállítást. Ez teszi lehetővé, hogy az alkalmazás könnyedén futtatható legyen a Spring Boot alapértelmezett beállításaival.

A fájlban található *@Bean* annotáció pedig egy *PasswordEncoder* típusú bean létrehozására szolgál. Ez a bean a jelszavak titkosítására szolgáló *BCryptPasswordEncoder* példányt hozza létre, amelyet később a biztonsági konfigurációban lehet használni.

package com.geolidth.BackEnd;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.context.annotation.Bean;  
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;  
import org.springframework.security.crypto.password.PasswordEncoder;  
  
@SpringBootApplication  
public class BackEndApplication {  
  
 @Bean  
 public PasswordEncoder passwordEncoder() {  
 return new BCryptPasswordEncoder();  
 }  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(BackEndApplication.class, args);  
 }  
  
}

Az alábbiakban található a **BookCotroller.java** fájl tartalma, amely a könyvekkel kapcsolatos HTTP kérések kezéséért felelős. A fájlban a REST API funkciói valósulnak meg a következő módon:

1. *Könyvek lekérdezése: Az @GetMapping* annotáció segítségével a *getBooks* metódus felelős az összes könyv lekérdezéséért.
2. *Könyv részleteinek lekérdezése: Az @GetMapping("/{id}")* annotációval ellátott *getById* metódus felelős egy adott könyv részleteinek lekérdezéséért az azonosító alapján.
3. *Könyv mentése: Az @PostMapping* annotációval ellátott *save* metódus felelős az új könyv mentéséért. A kliens által küldött új könyv adatait egy *NewBook* objektumban kapja meg, és aztán a *bookService* segítségével elmenti az adatbázisba.
4. *Könyv frissítése: Az @PutMapping("/{bookId}")* annotációval ellátott *updateBook* metódus felelős egy meglévő könyv frissítéséért. A kliens által küldött frissített könyv adatokat egy *UpdateBook* objektumban kapja meg, és aztán a *bookService* segítségével frissíti az adatbázisban.
5. *Könyv törlése: Az @DeleteMapping("/{bookId}")* annotációval ellátott *deleteBook* metódus felelős egy könyv törléséért az azonosító alapján.

package com.geolidth.BackEnd.Controllers;  
  
import com.geolidth.BackEnd.models.dao.BookUser;  
import com.geolidth.BackEnd.models.dto.UpdateBook;  
import com.geolidth.BackEnd.models.dao.Book;  
import com.geolidth.BackEnd.models.dto.NewBook;  
import com.geolidth.BackEnd.services.BookService;  
import org.springframework.http.HttpStatus;  
import org.springframework.http.ResponseEntity;  
import org.springframework.security.core.Authentication;  
import org.springframework.web.bind.annotation.\*;  
  
  
import java.util.List;  
import java.util.Optional;  
  
@RestController  
@RequestMapping("/books")  
public class BookController {  
 private final BookService bookService;  
 public BookController(BookService bookService) {  
 this.bookService = bookService;  
 }  
  
 @GetMapping  
 public ResponseEntity<List<Book>> getBooks(@RequestParam Optional<String> q) {  
 List<Book> books = bookService.getBooks(q);  
 return ResponseEntity.*status*(HttpStatus.*OK*).body(books);  
 }  
 @GetMapping("/{id}")  
 public ResponseEntity<Book> getById(@PathVariable int id) {  
 Book book = bookService.getById(id);  
 return ResponseEntity.*status*(HttpStatus.*OK*).body(book);  
 }  
  
 @PostMapping  
 public ResponseEntity<Book> save(@RequestBody NewBook newBook, Authentication auth) {  
 BookUser user = (BookUser) auth.getPrincipal();  
 Book savedBook = bookService.save(user.getId(), newBook);  
 return ResponseEntity.*status*(HttpStatus.*CREATED*).body(savedBook);  
  
 }  
 @PutMapping("/{bookId}")  
 public ResponseEntity<Book> updateBook(  
 @PathVariable int bookId,  
 @RequestBody UpdateBook updateBook, Authentication auth) {  
 BookUser user = (BookUser) auth.getPrincipal();  
 Book updatedBook = bookService.updateBook(user.getId(), bookId, updateBook);  
 return ResponseEntity.*status*(HttpStatus.*OK*).body(updatedBook);  
 }  
  
  
 @DeleteMapping("/{bookId}")  
 public ResponseEntity<?> deleteBook(@PathVariable Integer bookId, Authentication auth) {  
 BookUser user = (BookUser) auth.getPrincipal();  
 bookService.deleteBook(user.getId(), bookId);  
 return ResponseEntity.*status*(HttpStatus.*NO\_CONTENT*).build();  
 }  
  
  
 }

**2.7 User regisztráció és jogosultságok**

**2.8 Tesztelés**

**2.9 Fejlesztési lehetőségek**

**2.8 Fejlesztési lehetőségek**

Megfelelő komponensek hozzáadásával a honlap alkalmassá tehető regisztrált felhasználólk közötti közvetlen kommunikációra is forum felület kialakításával. Ez azonban mindenféleképpen erőforrás igényes, folyamatos felügyeletet igényel. Szükséges szűrőprogramok alkalmazása a kommunkáció stílusának és céljának ellenőrzésére . Ebben az esetben a felhasználói feltételek olyan kialakítása válna szükségessé amellyel a legtöbb visszaélés megakadályozható.

Lehetséges az adminok részére blog kialakítása, ahol személyes élményeket oszthatnának meg a felhasználókkal elősígítve az oldal aktív használatát.

Személyes találkozók megszervezéséhez nyújthat segítséget egy olyan menüpont, ahol az adott évben tervezett személyes találkozások láthatók. Ez megjelenhet továbbá egy

**3 Felhasználói dokumentáció**

**3.1. Honlap használata**

A weboldal használata lehetséges asztali gépről, laptopról, táblagépről, vagy mobiltelefonról. Webböngésző telepítésén kívül más program telepítésére és futtatására nem lesz szükséged, csak internetkapcsolatra.

Az alábbi böngészők bármelyikét elindíthatod:

A képen kör, Színesség, Grafika, Grafikus tervezés látható

Automatikusan generált leírás  A képen rajzfilm látható

Automatikusan generált leírásA képen kör, Színesség, gömb, bolygó látható

Automatikusan generált leírás

Dupla kattintással tudod megnyitni a böngészőt és a következő url-t kell beírnod a kereső mezőbe:

A Bookhub weboldala egy különleges lehetőséget biztosít minden könyvszerető embernek, hogy a könyveid ne csak porosodjanak a polcodon, hanem új otthont és új gazdát találjanak. Fedezd fel a könyvcsere kalandját, ahol nem csak olvasol, hanem megiszthatod és cserélheted is a kedvenc történeteidet. csak jelentkezz be, és kezdődjön a kaland!

**3.2 Honlap funkciói**

A képen szimbólum, Grafika, képernyőkép, Téglalap látható

Automatikusan generált leírás Erre az ikonra kattintva minden oldalról a főoldalra jutsz vissza. A főoldalon az oldal tetején megjelenő menüben tudsz választani a honlap szolgáltatásai között.

A könyvek a főoldalon töltődnek be, itt megtekintheted a választékot bejelentkezés nélkül is. Válassz egy könyvet, jelezd a tulajdonosnak, és beszéljétek meg hogyan tudod átvenni. Töltsd fel régi könyveidet, és várd, hogy jelentkezzen érte az új tulajdonos!

Regisztrációhoz a Sign up menüpontban tudod megadni az adataidat. Az adatok valódisága a te felelősséged.

A képen képernyőkép látható

Automatikusan generált leírás

Adatvédelem: az ön adatait(nevét, címét, elérhetőségét) csak akkor ismerheti meg más felhasználó, amikor jelzi önnek, hogy valamelyik könyvét szeretné átvenni, és felveszik a kapcsolatot egymással. Kérjük járjon el figyelmesen, hogy kivel veszi fel a kapcsolatot. A felhasználó felelőssége, hogy valós adatokat adjon meg, azt a weboldal adminjainak nem áll módjában ellenőrizni. Visszaélés esetén kérem jelezze az adminoknak, hogy meg tudjuk tenni a további intézkedéseket.

Ha nem tudsz választani, kérd meg a zsákbamackót, hogy segítsen!

A képen szöveg, képernyőkép, szoftver, Webhely látható

Automatikusan generált leírás

Ha kiváncsi vagy az alapítókra válaszd az about menüpontot.

A kapcsolat menüpontban nyugodtan üzenhetsz az adminisztrátoroknak bármilyen problémával vagy kéréssel. Ne habozz igénybevenni ezt a lehetőséget, ha valamit nem találsz, vagy Vannak ötleteid, hogyn fejleszthetnénk tovább a honlapot. Tudod jelezni nekünk továbbá a tömeges cserére vonatkozó kérésedet is, így nem kell minden egyes könyvet egyesével felrögzítened.

A könyvek cseréjének gyakorlati megvalósítását segítendő a könyveknél jelezve van a megye, ahol átvehető személyesen .

Minden olyan könyvnél tudod az átvételi szándékot, amelyik a honlapon látható. A már lefoglalt könyvek eltűnnek a kínálatból, így nem tudsz olyan könyvet véletlenül lefoglalni, amit más már lefoglalt magának.

**3.3Hibajelzések, elérhetőségek**

Hibás adatok a regisztrációnál: Kérlek ellenőrizd, hogy a név mezőben nem szerepel-e téves karakter, a cím mezőben nem szerepel-e hibás karakter, illetve minden mezőt kitöltöttél-e a regisztrációs felületen?

Hibás adatok a könyvnél: Ellenőrizd újra, hogy a könyv címe nem túl hosszú-e? Ha nem fér ki a rendelkezésre álló mezőben, használj rövidítést.

Amennyiben a honlap hibásan jelenik meg, ellenőrizd, hogy a böngésződ frissítve van, és valóban a legújabb verziót használod.

Az adminok elérhetőségét az about menüpont alatt találod, a honlapon keresztül tudsz nekünk üzenetet küldeni. Ne habozz jelezni bármilyen hibát, amit találsz!

**4.Összefoglalás**