

**Békéscsabai Szakképzési Centrum  
Trefort Ágoston Technikum  
Szakképző Iskola és Kollégium**

**Záródolgozat**

**Online Könyvkölcsönző oldal**

**Készítette: Dézsi János  
OKJ szám: 5421305**

**2024.**

# Tartalomjegyzék

|                                |    |
|--------------------------------|----|
| Bevezető .....                 | 1  |
| Rendszerkövetelmények.....     | 2  |
| Választott téma indoklása..... | 2  |
| A program telepítése .....     | 3  |
| Adatmodell Leírás .....        | 8  |
| Algoritmusok.....              | 9  |
| Használata .....               | 15 |
| Fejlesztés terv .....          | 19 |
| Összegzés .....                | 20 |
| Források.....                  | 22 |

# Bevezető

Üdv kedves *Felhasználó / Olvasó* !

A bemutatott weboldal amit majd lát/láthat a **Libar-E** nevet kapta azzal a céllal hogy az emberek akik nem mennének el könyvárba vagy nincs idejük akár telefonról vagy otthoni eszközükről is tudjanak a könyvek világába élni.

Mivel az oldal még csak nemrég lett készen így a választék nem a legszélesebb még de a közel jövőben tervezünk a könyvárakkal kapcsolatot létesíteni amivel el tudjuk azt érni hogy bárki bárholnan bármikor kölcsönözhet .

Rajtunk keresztül még akár online is olvashatja vagy egy cím megadása után eljuttatjuk önhöz az említett könyvet

De nagyon fontos hogy a rendszerünket ne használják ki így nekünk is vannak szabályaink amit be kell tartani az *Olvasónak / Felhasználónak* mivel ha ezeket a szabályokat nem tartják be akkor a weboldalról ki lesznek tiltva és pénz büntetésben lesznek részesítve hogy a szabályaink fontosságát és a könyvtárak büntetését ne lehessen elkerülni



## Rendszerkövetelmények

a Gép amin sikeresen tesztelve lett a program egy elég gyenge gépen történt meg

Processzor: Intelcore i-3 8100

Ram: 4 GGB ram

Op rendszer: Windows 10

Tárhely: Min 5gb

Programok: Node.js

XAMPP

Visual Studio Code

Tesztelés alapján a programnak lehet van minimálisabb specifikációja de a legtöbb alkalommal ezen a specifikáción elfutott hiba/probléma nélkül sajnos más gépeken nem nagyon volt esélyem tesztelni még így a leg gyakrabban használt specifikációt tudom megadni csak

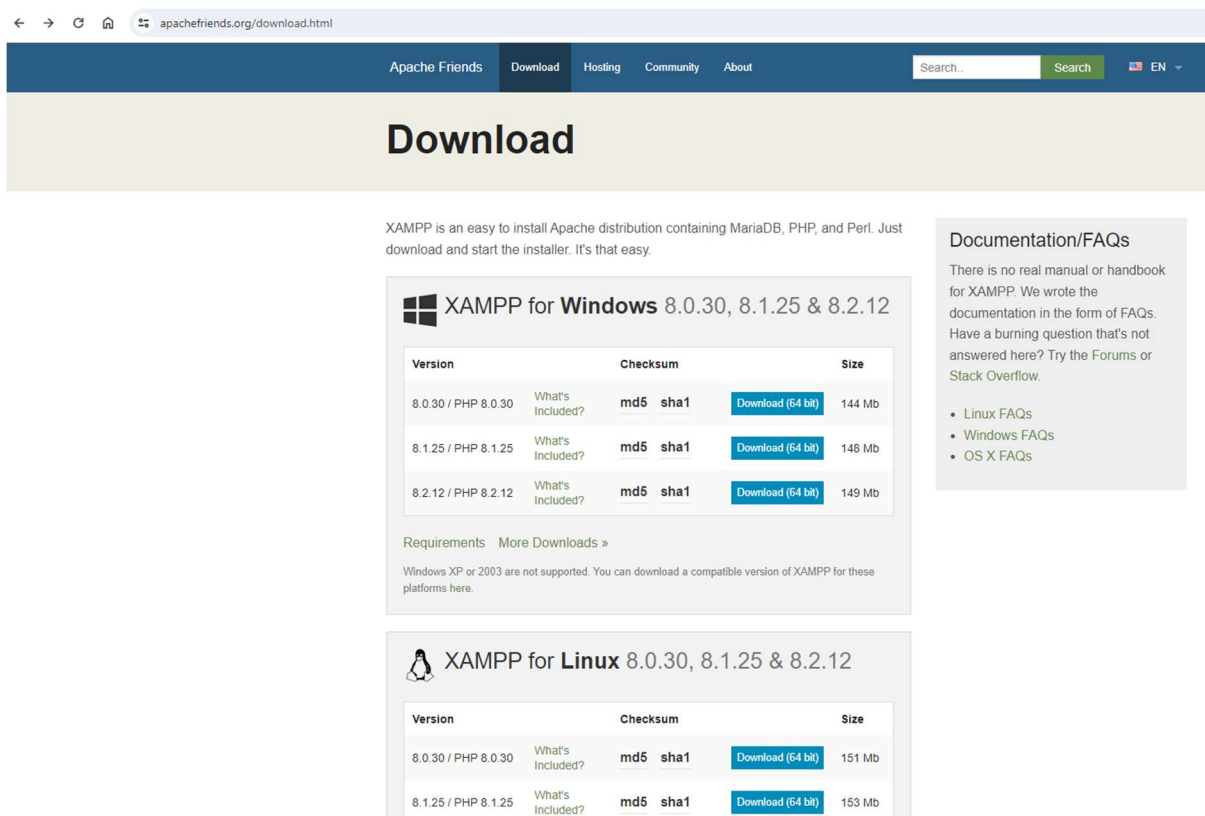
## Választott téma indoklása

A könyvek régen is érdekelték és hogyha például nem volt otthon egy könyv így mindig el kellett menni a könyvtárba de mint minden emberben bennem is van lustaság így nem mindig mentem el kölcsönözni a könyvet amit szerettem volna mert lusta voltam elmenni

De így a mostani tapasztalatommal megtudom csinálni azt hogy csináljak egy oldalt amivel lehet online kölcsönözni ezzel meg spórolva az időt és a fáradságot a könyvtárba elmenetellel

# A program telepítése

A működés első részéhez először is kell a program ami miatt működni fog az egész az adatbázis kezelő programunk ez esetben a xampp.



The screenshot shows the Apache Friends download page. The navigation bar includes 'Apache Friends', 'Download', 'Hosting', 'Community', and 'About'. A search bar is on the right. The main heading is 'Download'. Below it, a text block states: 'XAMPP is an easy to install Apache distribution containing MariaDB, PHP, and Perl. Just download and start the installer. It's that easy.'

There are two main sections for download:

### XAMPP for Windows 8.0.30, 8.1.25 & 8.2.12

| Version             | Checksum                  | Size   |
|---------------------|---------------------------|--------|
| 8.0.30 / PHP 8.0.30 | What's Included? md5 sha1 | 144 Mb |
| 8.1.25 / PHP 8.1.25 | What's Included? md5 sha1 | 148 Mb |
| 8.2.12 / PHP 8.2.12 | What's Included? md5 sha1 | 149 Mb |

Each row has a 'Download (64 bit)' button. Below the table are links for 'Requirements' and 'More Downloads »'. A note states: 'Windows XP or 2003 are not supported. You can download a compatible version of XAMPP for these platforms here.'

### XAMPP for Linux 8.0.30, 8.1.25 & 8.2.12

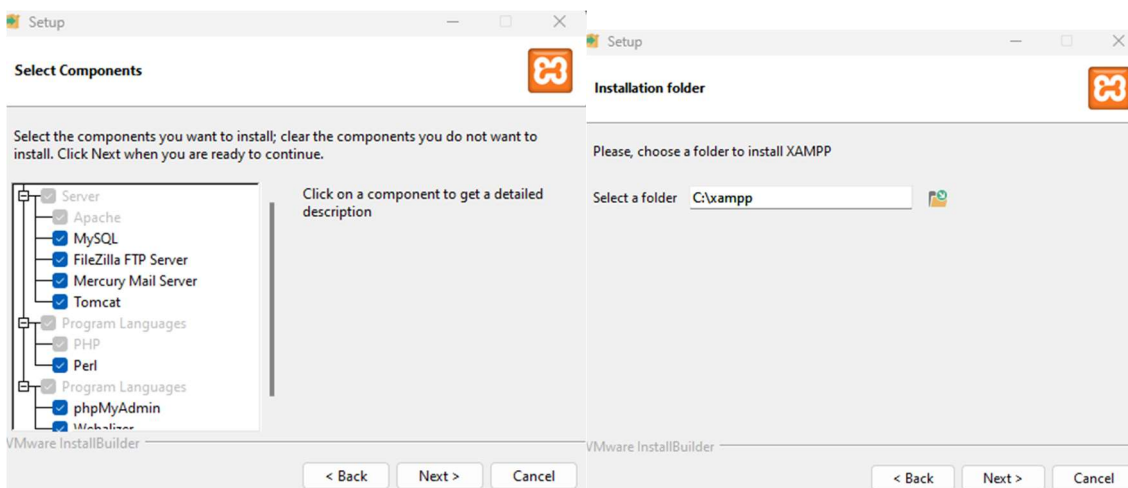
| Version             | Checksum                  | Size   |
|---------------------|---------------------------|--------|
| 8.0.30 / PHP 8.0.30 | What's Included? md5 sha1 | 151 Mb |
| 8.1.25 / PHP 8.1.25 | What's Included? md5 sha1 | 153 Mb |

Each row has a 'Download (64 bit)' button.

On the right, there is a 'Documentation/FAQs' section with the text: 'There is no real manual or handbook for XAMPP. We wrote the documentation in the form of FAQs. Have a burning question that's not answered here? Try the Forums or Stack Overflow.'

- Linux FAQs
- Windows FAQs
- OS X FAQs

Miután leszedtük a telepítőt fel is kell telepítenünk a programot válasszuk ki a beállításokat amiket szeretnénk a telepítéssel és hogy hova töltsse le ahol foglalja majd a számítógépen a helyet



The screenshot shows the XAMPP installation wizard. The left window is titled 'Select Components' and contains a tree view of components to be installed. The right window is titled 'Installation folder' and asks for a folder to install XAMPP.

**Select Components:**

- Server
  - Apache
  - MySQL
  - FileZilla FTP Server
  - Mercury Mail Server
  - Tomcat
- Program Languages
  - PHP
  - Perl
- Program Languages
  - phpMyAdmin
  - Webalizer

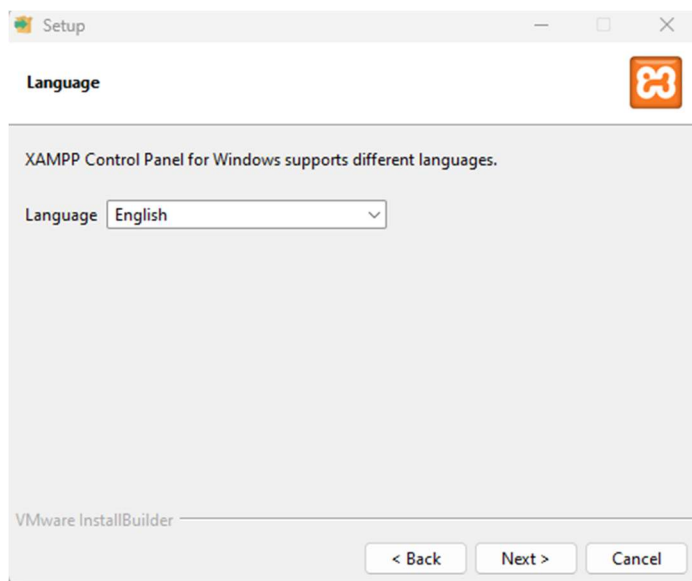
**Installation folder:**

Please, choose a folder to install XAMPP

Select a folder: C:\xampp

Both windows have '< Back', 'Next >', and 'Cancel' buttons at the bottom.

A következő lépés a nyelv kiválasztás lesz az személyes hogy milyen nyelven akarjuk használni mivel van olyan ember aki angolul jobban preferálja mivel jobban eligazodik de van ember aki inkább maradna az anyanyelvénél



és ezek után a beállítások után a program el is kezdi magát telepíteni a gépre ami után már csak a programot kell elindítani

A másik program ami kelleni fog az a Visual Studio Code amit a <https://code.visualstudio.com/docs/?dv=win64user/> linken tölthetünk le, válasszuk ki hogy melyik platformra akarjuk letölteni amint ez megvan



amint letöltött a telepítő mennyünk végig az általános beállításokon amiket be akarunk állítani mint például telepítési hely azokat állítsuk be és startra kész az alkalmazás

A Visual Studio Code amivel tudjuk futtatni magát a programot és azon belül utána a megfelelő mappába kell letölteni még külön csomagokat amik működtetni fogják az egészet a kódokat itt a telepítőbe

Kódok:

```
npm i
```

```
npm i express
```

```
npm i cors
```

```
npm i nodemon --save-dev
```

```
npm init -y
```

```
npm i prisma --save-dev
```

```
npx prisma
```

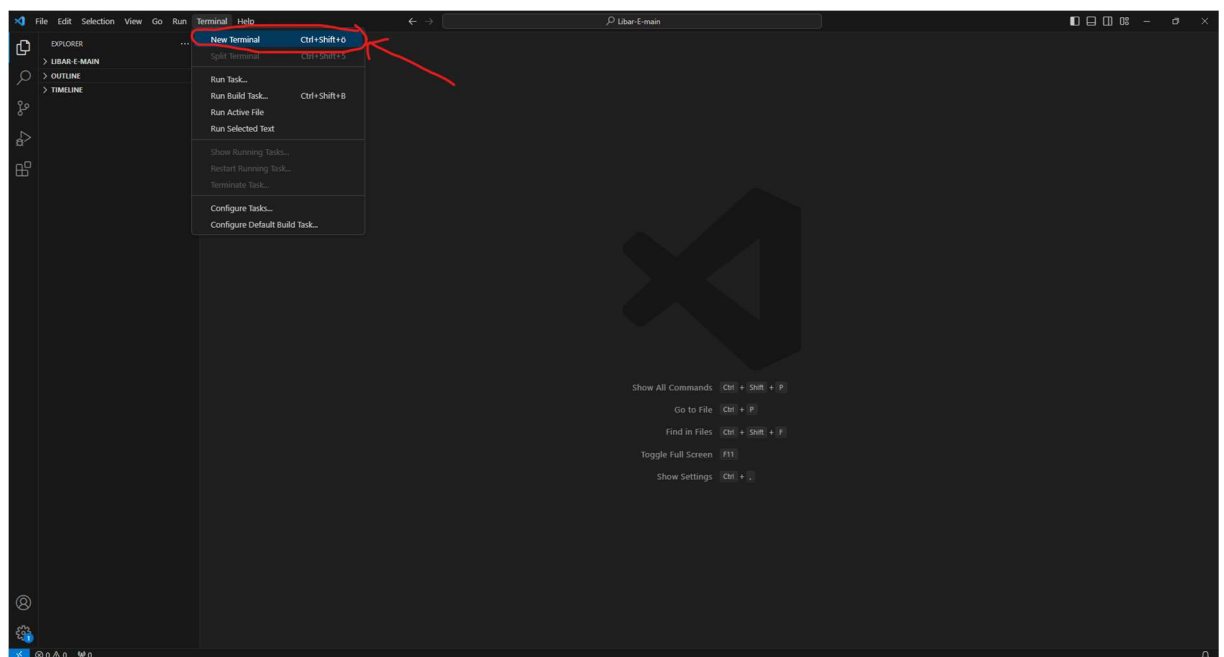
```
npx prisma migrate dev --name init
```

 (ha megvan már az adatváltás tervünk/schema ami már benne van alapból)

```
npm i @prisma/client
```

az `i` betű az `install` rövidítése (`install` = telepítés/letöltés)

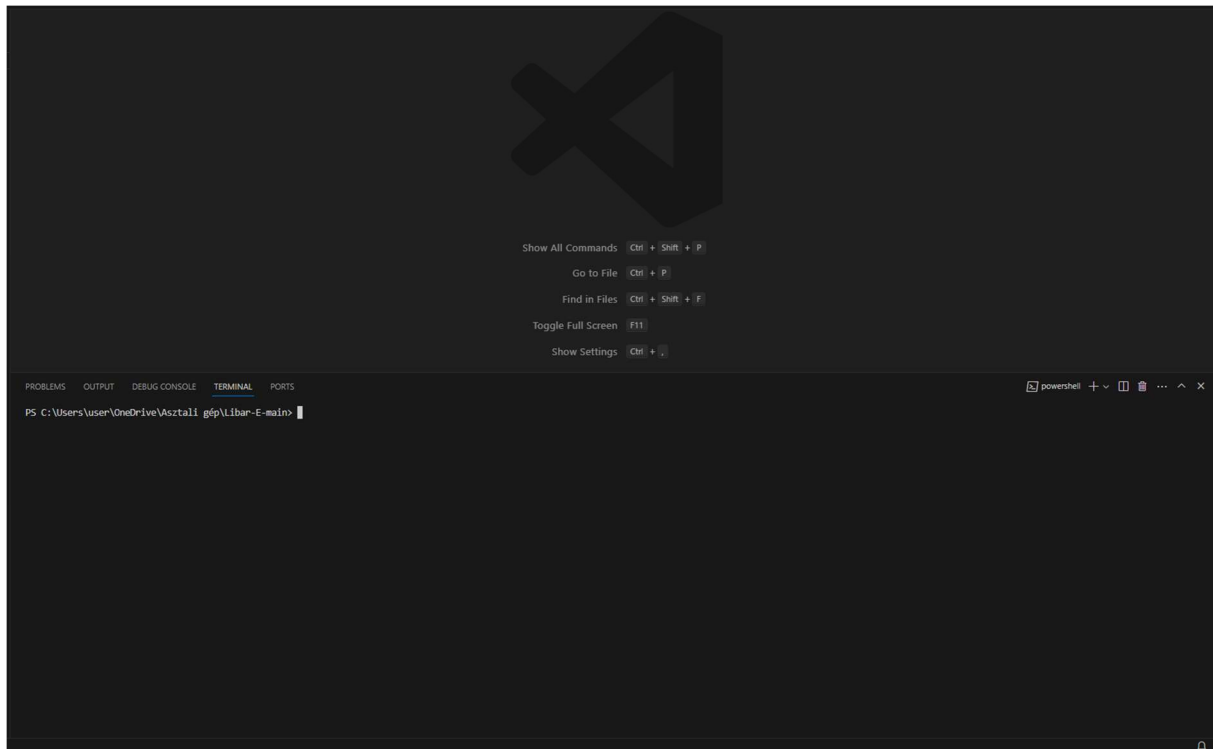
A felül látott parancsokat a Visual Studio Codeon belül kell beírni egy külön helyre ami végre hajtja ezeknek a letöltését és miután azok megvannak van egy működőképes programunk



A kiemelt *New Terminal* gomb megnyomásával megnyitjuk ezt a külön helyet

Ez a külön hely a terminál úgy működik mint egy parancssor vagy máshogyan megnevezve olyan mint egy kis számítógép amit használhatunk arra hogy a programunkat futtatni tudjuk és kezelni

A Terminál jó a hibakezelésre is mivel abba segít ha hibába ütközünk hogy ki írja pontosan hogy mi a hiba és hogy hol a hiba és hogy mikor történt a hiba



Miután az végrehajtottuk az előző lépéseket készen állunk arra hogy az előbb megadott parancsokat megadjuk hogy a programhoz való dolgok letöltődjenek és a programot működésre bírjuk és ezeket mind a megadott mappába ahol a program többi része fut

Ha mindent jól telepítettünk és az npm start nevű parancs használatával elindítjuk akkor a megadott mappából az alábbi képen látható szöveget/ üzenetet kéne kapnunk

```
PS C:\Users\diak\Desktop\konyvtar> cd backend
PS C:\Users\diak\Desktop\konyvtar\backend> npm start

> backend@1.0.0 start
> nodemon index.js

[nodemon] 3.0.3
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node index.js`
listening on 8000
```

.env fájl létrehozása

```
DATABASE_URL="mysql://root:@localhost:3306/konyvtar"
```

```
JWT_SECRET = "szupertitkostitok"
```



## Használt eszközök

A program elkészítéséhez használt programozási nyelv az javascript volt és Visual studio Code ba dolgoztam a képeket én készítettem a gépen alpból megtalálható Képmetsző programmal adatbázis-kezelésben xampp meg prisma volt használva és a dokumentáció megíráshoz Microsoft Wordot használtam

A program megírásához használt eszközök amik segítettek működtetni ezt:

node.js

expressjs

CORS

prisma.io

nodemon.io

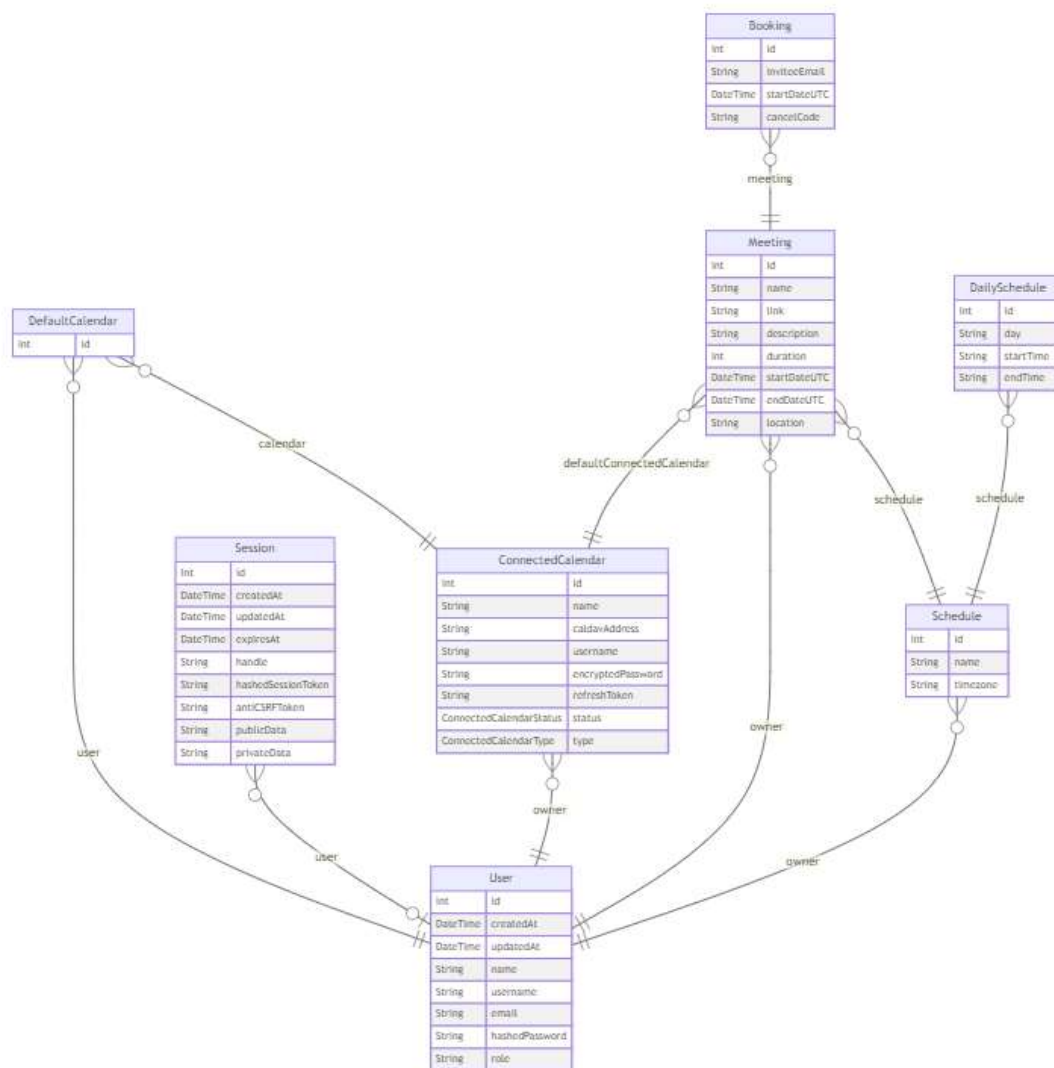
argon2

tailwind

jsonwebtoken

# Adatmodell Leírás

Az adatbázis kapcsolatai



A program **kölcsönzőkből** indul ki mivel a *felhasználónak/olvasónak* kell egy profil amivel tudnak kölcsönözni a profil adatai segítenek abba hogy rendszerünk megfelelően működjön mivel a lakcím a pénz büntetéssel és egyéb dolgokkal segít az email cím az ember azonosítója a telefonszámuk is azért kell hogy el tudjuk önöket érni ne csak emailen keresztül hanem azonnal is

A következő a **könyvek** tábla amibe egyértelműen a könyvek vannak eltárolva amiket lehet kölcsönözni

A **kölcsönzött** tábla számon tartja hogy a kölcsönzéseket hogy melyik könyvek vannak ki kölcsönözve és meddig mekkora határidővel és mikortól érvényes a kölcsönzése és hogy a könyv vissza lett e már hozva vagy még nem vagy esetleg határidőn átment e. a tábla még tartalmaz két mezőt ami 2 azonosító mező az egyik a kölcsönző táblából a kapcsolat miatt hozzáfér az ottani azonosítóhoz amivel az bejelentkezett illető látja a saját kölcsönzéseit. a másik azonosító az a könyv táblából az azonosító amit szintén egy kapcsolattal érünk el

Az utolsó tábla a **késés** tábla ami számon tartja hogy ki hányszor és mennyit késett a mivel a késő emberekkel szabály szerint kell eljárni (a weboldal fő oldalán található a szabályzat) hogy az illető ne kövessen el ilyet többször és hogy a károkat fizesse ki

A késés táblába 3 azonosító is van és mivel több az egyhez kapcsolat így elég egy kapcsolatot létrehozni és így a többi azonosító is kapcsolódik így a kölcsönzés azonosítójával ki lehet keresni melyik kölcsönzés volt az amivel késett az illető és hogy melyik könyv volt az

Röviden megmagyarázva a kölcsönző táblából kezdünk hogy a felhasználónak legyen egy profilja amivel hozzá férhet a weboldalhoz ezek után miután a weboldalhoz kapott hozzáférést amivel a könyvek oldalhoz hozzáfér és keresheti a kívánt könyvet/könyveket, amint a könyvet kiválasztotta a kölcsönzés oldalon tudja a kölcsönzését befejezni ahol amint a kölcsönzése felkerült az adatbázisba a kölcsönzéseim oldalnál a kölcsönzött táblából le lesznek kérve a kölcsönzéseik azonosító által és ki lesz írva az oldalra

A kapcsolatok ezt segítik röviden leírva

## Algoritmusok

Az algoritmusok avagy külön kódok feladatai:

```
const {PrismaClient} = require("@prisma/client")
const jwt = require('jsonwebtoken');
const argon2 = require('argon2');
const prisma = new PrismaClient();
```

Először is ez a 4 sor segít abban hogy a felhasználónak legyen védett hozzáférése a weboldalhoz és a prisma kliens fölvegye az adatbázisba a regisztrálni kívánó illetőt.

A Prisma Client segít abba hogy az adatbázissal lehessen kapcsolatot létesíteni és fenn tartani azt és adatot vigyünk fel az adatbázisba

a jwt másnéven a jsonwebtoken a token rendszert nyújtja amivel a felhasználó tud biztonságosan igazolni hozzáférést kapni az oldalhoz

az argon2 nyújtja a felhasználó kódjának a biztonságát mivel titkosítja a jelszót

```
const generateToken = (id) => {
  return jwt.sign({id}, process.env.JWT_SECRET, {expiresIn: '12h'});
}

const register = async (req, res) => {

  const {email, jelszo, telszam, lakcim, nev} = req.body;

  if(!email || !jelszo || !telszam || !nev){
    res.json({message: "Hiányos adatok!"});
    return;
  }

  const kolcsonzo = await prisma.kolcsonzo.findUnique({
    where: {
      email: email
    }
  });

  if(kolcsonzo){
    res.json({message: 'Az email cím már foglalt'});
    return;
  }

  const hasheltJelszo = await argon2.hash(jelszo);

  const ujKolcsonzo = await prisma.kolcsonzo.create({
    data: {
      email: email,
      nev: nev,
      telszam: telszam,
      lakcim : lakcim,
      jelszo: hasheltJelszo
    }
  });

  const token = generateToken(ujKolcsonzo.id);
  res.json({token: token});
}
```

a generateToken biztosítja a felhasználó biztonságát mivel ad neki egy külön hozzáférési azonosítót ad neki ami 12 óránként lejár

a register a regisztrációt kezeli ahol egy email csak egyszer szerepelhet és a kódja titkosítva van az argon 2 nevezetű csomagnak köszönhetően

```

const login = async(req, res) => {

  const {email, jelszo} = req.body;
  if(!email || !jelszo){
    res.json({message: 'Hiányzó adatok'});
  }

  const kolcsonzo = await prisma.kolcsonzo.findUnique({
    where: {
      email: email
    }
  });

  if(!kolcsonzo) {
    res.json({message: 'Felhasználó nem található'});
    return;
  }

  if(!(await argon2.verify(kolcsonzo.jelszo, jelszo))){
    res.json({message: 'Nem megfelelő a jelszó!'});
    return;
  }

  const token = generateToken(kolcsonzo.id);
  res.json({
    message: 'Sikeres bejelentkezés',
    token: token
  });
}

```

a képen a login azaz a bejelentkezés látható ami csak akkor ad ki token-t az illetőnek ha megfelelő jelszót ad meg és a belépéshez csak email és jelszó kell

```

const addKonyv = async (req, res) => {
  const {cim, iro, kategoria, kiadasDatum} = req.body;

  const ujKonyv = await prisma.konyvek.create({
    data: {
      cim: cim,
      iro: iro,
      kategoria: kategoria,
      kiadasDatum: kiadasDatum
    }
  });

  res.json({message: "Sikeres termékfelvitel", ujKonyv});
}

const getKönyv = async (req, res) => {
  const konyvek = await prisma.konyvek.findMany({});
  res.json({konyvek});
}

```

A könyveknél az addKönyv azért van hogyha egy új könyvet kéne felvenni az adatbázisba és az alatta lévő getKönyv arra van hogy az adatbázisból megkeresse a könyveket a táblából ennek a segítségével lehet megjeleníteni a könyveket a weboldalon

```
const placeKolcsonzes = (req, res) => {
  const {konyvek} = req.body;
  console.log(req.body);
  const kolcsonzo = req.kolcsonzo;
  console.log(kolcsonzo)
  try{
    konyvek.forEach(async (konyv) => {
      const now = new Date();
      now.setDate(now.getDate()+30)
      for(let i = 0; i < konyv.id; i++) {
        const kolcsonzes = await prisma.kolcsonzesek.create({
          data: {
            kcs_id: Number(kolcsonzo.id),
            kny_id: Number(konyv.id),
            hatarido: new Date(
              now.getFullYear(),
              now.getMonth(),
              now.getDate()
            )
          }
        });
      }
    });
  } catch(err){
    res.json(err.message)
  }
}
```

A kód ami megoldja hogy lehessen kölcsönözni

kölcsönzés jelenleg csak úgy működik hogy az az napi dátumtól kezdődik a kölcsönzés a kiválasztott határidőig amit erősen ajánlok hogy pontosan állítson be és hozza vissza időbe arra amit megadott mert ha nem tudja ezt megtenni a szabályzat alapján kell eljárni

```
const getkolcsonzes = async (req, res) => {

  const kolcsonzo = req.kolcsonzo;

  const getKonyv = await prisma.kolcsonzesek.findMany({
    where: {
      kcs_id: kolcsonzo.id,
    },
    select: {
      kivitel: true,
      hatarido : true,
      konyv: {
        select: {
          cim: true,
          iro: true,
          kategoria: true,
        }
      }
    }
  });

  res.json({
    kolcsonzesek: getKonyv
  })
}
```

Ez a kód ami miatt a weboldalon lehet látni a saját kölcsönzéseket

a kód lekéri a felhasználó azonosítójával ellátott rendelést aki éppen be van jelentkezve és ki írja a kölcsönzéseket

```
const jwt = require('jsonwebtoken');

const protect = async (req, res, next) => {
  let token;
  if (req.headers.authorization && req.headers.authorization.startsWith('Bearer ')) {
    try {
      token = req.headers.authorization.split(' ')[1];
      const idFromToken = jwt.verify(token, process.env.JWT_SECRET);
      req.kolcsonzo = await prisma.kolcsonzo.findUnique({
        where: {
          id: idFromToken.id
        },
        select: {
          id: true,
          email: true,
          nev: true,
          telszam: true,
        }
      });
      next();
    } catch (err) {
      res.status(401).json({ message: 'Gondok vannak' });
    }
  }

  if (!token) {
    res.json({ message: 'Be kell jelentkezni!' });
  }
}
```

a kód ami miatt a felhasználó biztonságba van a weboldalon

Routes:

```
1 const express = require('express');
2 const router = express.Router();
3 const {protect} = require('../middlewares/authMiddleware');
4
5
6 const {register, login ,getKolcsonzo} = require('../controllers/kolcsonzoController');
7
8 router.post('/register', register);
9 router.post('/login', login);
0 router.get('/me', protect,getKolcsonzo )
1
2 module.exports = router;
```

ebből több is van de elég egy példa hogy az előző kódokat kieszedtem a fájlokból és ezeken a címeken keresztül majd el lehet érni őket

```
const express = require('express');
const cors = require('cors');

const app = express();
const PORT = 8000;

app.use(express.json());
app.use(cors());
app.use('/kolcsonzesek', require('../backend/routes/kolcsonzesekRoutes'))
app.use('/kolcsonzo', require('../backend/routes/kolcsonzoRoutes'))
app.use('/konyv', require('../backend/routes/konyvekRoutes'))

app.listen(PORT, () => {
  console.log(`listening on ${PORT}`);
});

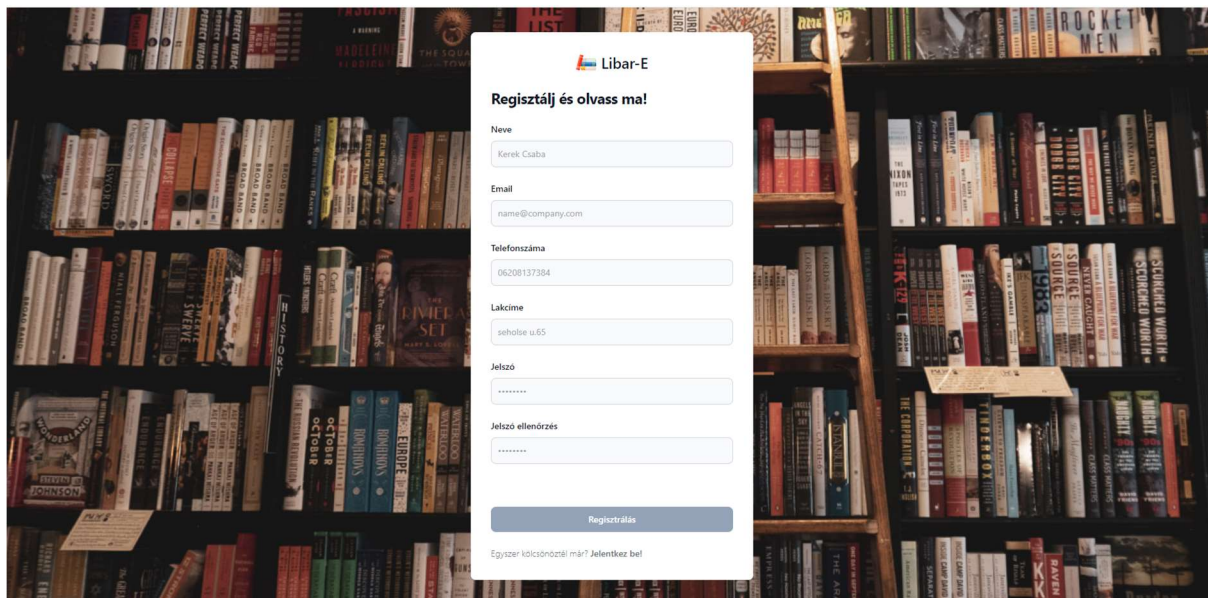
app.get('/', (req, res) => {
  res.json({message: 'Hello'});
});
```

Ez pedig a központja az oldalaknak ahonnan eligazodik a weboldal



# Használata

Miután sikeresen telepítetted és beállítottad a programot, könnyedén hozzáférhetsz a könyvkölcsönző weboldalhoz, hogy élvezd az online könyvkölcsönzés kényelmét és előnyeit.



The image shows a registration form for 'Libar-E' overlaid on a background of bookshelves filled with books. The form is titled 'Regisztrálj és olvasd ma!' and includes the following fields: 'Nev' (Name) with the value 'Kerek Csaba', 'Email' with the value 'name@company.com', 'Telefonszáma' (Phone number) with the value '06208137384', 'Lakcíme' (Address) with the value 'seholse u.55', 'Jelszó' (Password) with a masked value '\*\*\*\*\*', and 'Jelszó ellenőrzés' (Password confirmation) with a masked value '\*\*\*\*\*'. There is a 'Regisztrálás' (Registration) button at the bottom. Below the button, there is a small text link: 'Egyszer kölcsönözöl már? Jelentkez bel'.

Az oldal teljes eléréséhez először is regisztrálni kell egy fiókot. A regisztráció egy egyszerű folyamat, amelyhez csak pár adat megadása szükséges. Ezután regisztrálással végzett és el is kezdheti használni a weboldalt

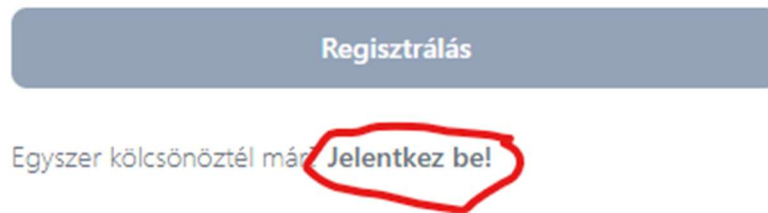
A regisztrációnál a nevet általánosságban kérjük hogy legyen milyen néven megszólítani hogyha szeretnénk önt valamiért és valamilyen módon elérni ha olyan dolog történne

Az emailje alapján tudjuk azonosítani a felhasználót és ezzel kapjuk meg az egyik főbb elérési módunkat ami alapján tudunk üzeni ha olyan eset történne ami miatt el kéne érünk a felhasználót

A telefonszáma szintúgy az elérése miatt kel sürgősebb esetnél amikor már az email nem elég vagy mivel gyorsan kell megoldani az esetet

A lakcímét azért kérjük el mert ha a szabályaink megsértése miatt esetleg pénz büntetést kéne kiadnunk akkor a házához fogjuk a számlát küldeni és emailbe is ki fogjuk küldeni

Ha már van profilja



az alábbi linkre nyomjon rá és átvizsi a belépés oldalára

A bejelentkezésnél csak az emailjét és a jelszavát kérjük hogy biztosítani tudjuk hogy ugyan az a felhasználó jelentkezett be aki csinálta a profilt és nem feltörték

így ezzel a módszerrel csak az emailt nem lehet többször felhasználni egy profil csinálásához így nevet és jelszót is lehet többször felhasználni hogy az egyszerű jelszavak a minden napi élet kedvéért ne legyenek foglalva hisz nem minden ember fogja felírni magának a jelszót vagy megjegyezni a nagyon hosszúakat

A bejelentkezésnél pedig csak az email meg a jelszó kell de ha nem lenne profilja akkor ugyanott ahol előbb található volt a gomb van megint egy gomb ami átvizs a regisztrációs oldalra

Amint regisztrál az adat el lesz tárolva az adatbázisba de ha csak belépett akkor kap egy szimpla tokent és élvezheti a weboldalt

Amint sikerült belépnie/regisztrálnia azonnal a fő oldalra fogja dobni



A fő oldalon lehet látni a szabályzatot ami amit erősen ajánlok elolvasni mivel egy fontos része a weboldalnak és az alapján működik a weboldal is

a szabályok:

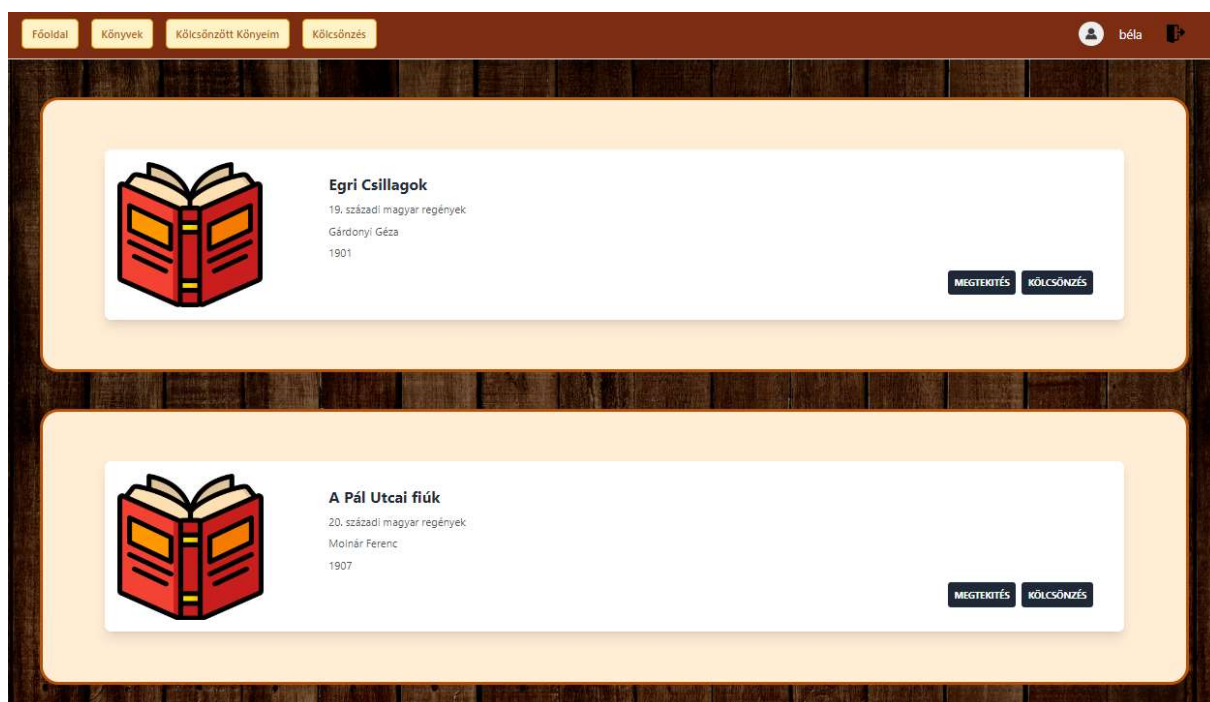
**Késés:** Késés esetén az olvasó kap 1 jelzést amit a profilja alatt láthat és pénzbüntetést szabunk ki 5 jelzés után a felhasználót kitiltjuk

**Könyvek:** A könyvek nem megfelelő állapotba visszajuttatása pénz büntetést és 1 jelzést von maga után ha a könyv egyáltalán nem jelenik meg nálunk egy bizonyos idő után súlyosabb pénz büntetés fogja érni és teljes kitiltás

**Információ:** Téves Információ esetén csak jelezzen az nekünk és segítettünk önnek ennek a javításával de ha téves információt használ rendelésre és nem tervezi változtatni akkor kitiltásban részesítjük

Amint a könyveknél sikerült kiválasztania melyik könyvet szeretné ki kölcsönözni a kölcsönzés gomb megnyomásával átmegy arra az oldalra ahol teljesen befejezheti a kölcsönzést

A könyvek oldal ahol lehet kutatni a könyvek között

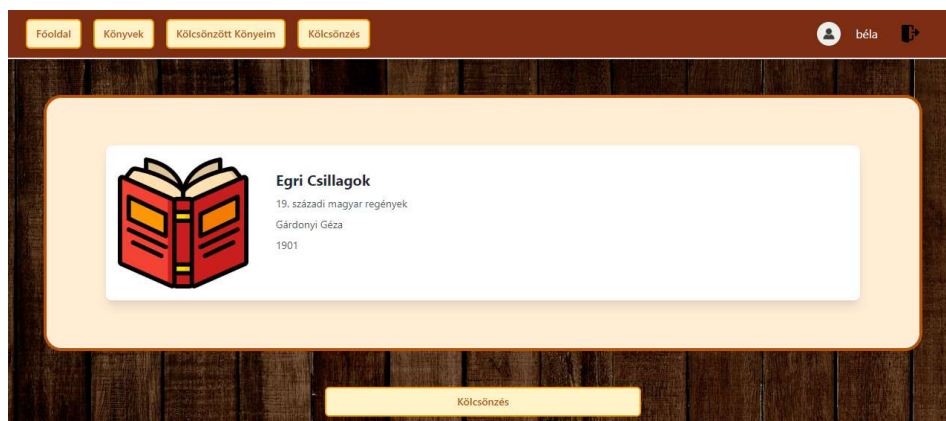


a könyveknek a címe , kategóriája, írója és kiadási dátuma van megjelenítve  
2 gomb van a megtekintés és a kölcsönzés

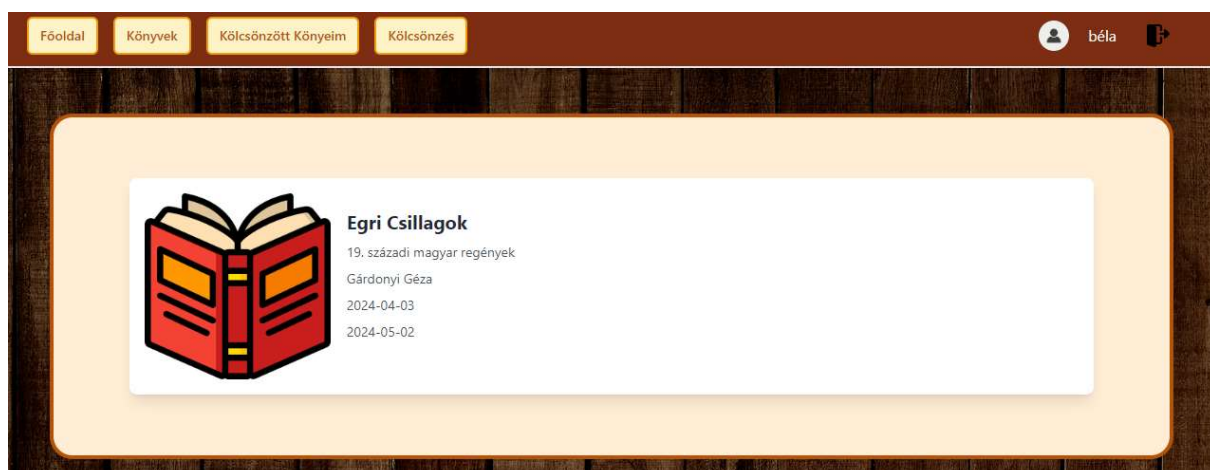
A megtekintés a könyvről egy részletesebb oldalt láthat majd a könyvről leírással és  
egyéb más információval

A kölcsönzés gomb eltárolja hogy melyik könyvet szeretné kikölcsönözni amit a  
kölcsönzés oldalnál látni

A kölcsönzés oldal ahol egyértelműen kölcsönözhet



A kölcsönzés oldalon a kiválasztott könyvet/könyveket kikölcsönözhet



És végül a kikölcsönzött könyvek ahol a felhasználó láthatja a saját kikölcsönzött könyveit és hogy mettől meddig vannak kölcsönözve

a határidő fontos pontja még mindig a szabályok miatt

## Fejlesztés terv

A Késes megoldása elsőnek mivel nem volt időm sajnálatos módon rá a profil megnézés a könyveknek rendesen képek a könyveknek leírás vagy megnézni a könyveket

A profil megtekintést és magát a szabály rendszert nem sikerült mivel szintúgy idő miatt megoldani és egyéb kódokat nem tudtam rendesen hozzáadni sajnos de megvannak csinálva

a kódok amit említettem:

```

const getSpecifiedKönyvKategoria = async (req, res) => {
  const kategoria = req.params.kategoria;

  const konyvek = await prisma.konyvek.findMany({
    where: {
      kategoria: kategoria
    }
  });

  res.json(konyvek);
}

const getSpecifiedKönyvIro = async (req, res) => {
  const iro = req.params.iro;

  const konyvek = await prisma.konyvek.findMany({
    where: {
      iro: iro
    }
  });

  res.json(konyvek);
}

const getSpecifiedKönyvCim = async (req, res) => {
  const cim = req.params.cim;

  const konyvek = await prisma.konyvek.findMany({
    where: {
      cim: cim
    }
  });

  res.json(konyvek);
}

const getSpecifiedKönyvid = async (req, res) => {
  const id = req.params.id;

  const konyvek = await prisma.konyvek.findFirst({
    where: {
      id: Number(id)
    }
  });

  res.json(konyvek);
}

```

a könyvek leszűrése író, kategória, cím vagy azonosító alapján

a kölcsönzés idő tartamának beállítása

## Összegzés

A Libar-E könyvkölcsönző online platformot azzal a céllal hoztuk létre, hogy lehetővé tegyük az emberek számára, hogy könnyedén hozzáférjenek a könyvek világához, bármikor és bárhol, a nap 24 órájában. A projekt során hangsúlyt fektettünk az egyszerűsége, a

kényelemre és a biztonságra, hogy a felhasználók teljes mértékben élvezhessék az online könyvkölcsönzés előnyeit.

Az oldal létrehozásának indítéka az volt, hogy megkönnyítsük az emberek számára az olvasásra való lehetőséget, anélkül, hogy azoknak el kellene menniük a könyvtárba. A könyvkölcsönző weboldal lehetővé teszi a felhasználók számára, hogy böngésszenek a különböző könyvek között, kölcsönözzenek, és élvezzék az olvasás örömét otthonról vagy akár útközben is.

A projekt során kiemelt figyelmet fordítottunk az alkalmazás biztonságára és a felhasználói élményre. A regisztráció és bejelentkezés folyamata biztonságos és egyszerű, hogy a felhasználók mindig védettek legyenek az adatvédelem terén. Emellett a felhasználói profil lehetőséget biztosít személyre szabott könyvkölcsönzési preferenciák beállítására, és a korábbi kölcsönzések nyomon követésére.

Az oldal kialakítása és navigációja intuitív és felhasználóbarát, hogy a felhasználók könnyen megtalálják a keresett könyveket és funkciókat. A főoldalon található ajánlások és legnépszerűbb könyvek listája segít a felhasználóknak az új olvasnivalók felfedezésében, míg a kategóriák és címek szerinti rendezés lehetővé teszi a pontosabb keresést.

Az online könyvkölcsönző weboldal jelenleg alapvető funkcionálitással rendelkezik, azonban a jövőben terveink között szerepel további funkciók és fejlesztések bevezetése. Ide tartozik a kölcsönzések pontosabb nyomon követése, a könyvek részletes leírásainak hozzáadása, valamint a profilok bővítése és a szabályrendszer továbbfejlesztése.

Az eddigi tapasztalatok alapján úgy véljük, hogy a Libar-E könyvkölcsönző weboldal jelentős előrelépést jelent az olvasás és könyvkölcsönzés terén, és reményeink szerint hozzájárul az emberek kényelmesebb és élvezetesebb olvasási élményéhez.

Az összefoglalóban megemlített céljaink és fejlesztési terveink alapján továbbra is elkötelezettek vagyunk az online könyvkölcsönzés platformjának folyamatos fejlesztése mellett, hogy még több embert szolgálhassunk ki az olvasás szeretetének és örömeinek terjesztésében.

## Források

1. Express.js hivatalos: <https://expressjs.com/>
2. CORS (Cross-Origin Resource Sharing): <https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS>
3. Prisma ORM (Object-Relational Mapping): <https://www.prisma.io/>
4. Nodemon: <https://nodemon.io/>
5. Argon2: <https://www.npmjs.com/package/argon2>
6. JSON Web Token (JWT): <https://jwt.io/>
7. Tailwind CSS: <https://tailwindcss.com/>

Ezek a források hozzájárultak a Libar-E könyvkölcsönző weboldal fejlesztéséhez és a dokumentáció elkészítéséhez, biztosítva a szükséges információkat és eszközöket a projekt megvalósításához.



# Plágium-nyilatkozat

Alulírott ..... (név)  
..... (szül.hely)  
..... (szül.idő)  
..... (anyja neve)

jelen nyilatkozat aláírásával kijelentem, hogy a

..... című záródolgozat

(a továbbiakban: dolgozat) önálló munkám, a dolgozat készítése során betartottam a szerzői jogról szóló 1999. évi LXXVI. tv. szabályait, valamint a Békéscsabai Szakképzési Centrum által előírt, a dolgozat készítésére vonatkozó szabályokat.

Tudomásul veszem, hogy a dolgozat esetén plágiumnak/szerzői jogsértésnek számít:

- szó szerinti idézet közlése idézőjel és hivatkozás megjelölése nélkül;
- tartalmi idézet hivatkozás megjelölése nélkül;
- más szerző publikált gondolatainak saját gondolatként való feltüntetése

Kijelentem továbbá, hogy a dolgozat készítése során az önálló munka kitétel tekintetében a konzulenszt, illetve a feladatot kiadó oktatót nem tévesztettem meg.

Jelen nyilatkozat aláírásával tudomásul veszem, hogy amennyiben bizonyítható, hogy a dolgozatot nem magam készítettem vagy a dolgozattal kapcsolatban szerzői jogsértés ténye merül fel, a Békéscsabai Szakképzési Centrum a dolgozatot elégtelennek minősíti.

Békéscsaba, 20 .....hó .....nap

Tanuló aláírása