BUDAPESTI MŰSZAKI SZAKKÉPZÉSI CENTRUM

PETRIK LAJOS

KÉT TANÍTÁSI NYELVŰ VEGYIPARI, KÖRNYEZETVÉDELMI
ÉS INFORMATIKAI TECHNIKUM

SZOFTVERFEJLESZTŐ ÉS -TESZTELŐ TECHNIKUSI SZAKMA

5-0613-12-03

ASZTALI- ÉS WEBES SZOFTVERFEJLESZTÉS, ADATBÁZIS-KEZELÉS

IDŐTARTAM: 240 PERC

TARTALOMJEGYZÉK

KÖZPONTI INFORMÁCIÓK

A vizsgatevékenység során **a jelölt, a feladat kidolgozása közben offline ill. online dokumentációkat használhat, célirányosan elkészített segédanyagokat azonban nem.**

A - Szoftverfejlesztés és -tesztelés vizsgaremek vizsgarész	A vizsgázóknak minimum 2, maximum 3 fős fejlesztői csapatot alkotva kell a vizsgát megelőzően egy komplex szoftveralkalmazást lefejleszteniük.	*30 perc*	*55 pont*
B - Asztali- és webes szoftverfejlesztés, adatbázis-kezelés vizsgarész	A vizsgafeladat során a vizsgázónak egy számítógépes szoftverfejlesztési feladatokat tartalmazó feladatsort kell megoldania, amely tartalmaz backend, frontend, ill. desktop grafikus és konzolos funkcionalitást is.	*210 perc*	*65 pont*

ÁLTALÁNOS INFORMÁCIÓK A VIZSGATEVÉKENYSÉGRE VONATKOZÓAN

Felhasználható technológiák	PHP, C#, Java, Node.js, JavaScript/TypeScript, CSS, MariaDB
Javasolt alkalmazások	Visual Studio Code, IntelliJ IDEA
Javasolt technológiák	Laravel, NestJS, React, Vue.js, Bootstrap, JavaFX

Szoftverfejlesztő és -tesztelő technikusi szakma – 5-0613-12-03	
Asztali- és webes szoftverfejlesztés, adatbázis-kezelés	**Időtartam: 240 perc**

ASZTALI- ÉS WEBES SZOFTVERFEJLESZTÉS, ADATBÁZIS-KEZELÉS FELADATSOR

ÁLTALÁNOS TUDNIVALÓK

- Készítse el az alábbi alkalmazásokat, amelyekkel elvégezhetők az alábbi feladatok!
- Az alkalmazások elkészítéséhez tetszőleges fejlesztői környezetet, illetve programozási nyelvet használhat!
- Segítségül használhatók:
  - a gépre telepített offline help rendszerek
  - internetkapcsolat, amely használható:
    - általános keresésre;
    - online dokumentáció elérésére;
    - projekt keretek létrehozására, csomagok telepítésére.
    - Mindenfajta kommunikáció, vagy meg nem engedett segédanyag letöltése szigorúan tilos!

- **Kiadott források**:
  - books.sql – a „books” adatbázistábla szerkezete és kezdeti tartalma
  - szerzok.zip – a szerzők fényképei

- **Beadandó**: az összes projekt, a megoldások során létrejött kimeneti állományok, illetve a megoldás során létrehozott adatbázis exportja.
  - Mindezeket egyetlen tömörített fájlban töltse fel Teams feladat beadásaként!
  - Az állomány neve szoftvervizsga2024_sajatnev_osztaly legyen!
  - A beadott alkalmazások futtatható állapotúak legyenek!
  - A hibás vagy hiányos részeket kommentben hagyja a kódban, de jelezze, hogy az is a megoldás része!
  - A gyorsabb értékeléshez az elkészített programot futtatható állapotban hagyja megnyitva a számítógépén!

FELADATLEÍRÁS

Egy könyvtár kölcsönzés-nyilvántartó rendszerét kell elkészítenie, amely a könyvek, ill. a kölcsönzések adatait tartja nyilván.
A rendszer három fő komponensből kell álljon:
- egy webes backend alkalmazásból, amely REST API-n keresztül biztosít hozzáférést az adatokhoz
- egy böngészőben futó kliens alkalmazásból, amely a backend alkalmazást használja
- egy desktop/konzolos hibrid alkalmazásból

A könyvkatalógus egy része már digitalizálva lett (l. a mellékelt books.sql fájlt), amely az alábbi „books” adattáblát definiálja:
- id: egész szám, a könyv azonosítója, elsődleges kulcs, automatikusan kap értéket
- title: szöveg, a könyv címe
- author: szöveg, a könyv szerzője
- publish_year: egész, a kiadási év
- page_count: egész, a könyv hossza oldalban
- created_at: timestamp, amikor a rekord felvételre került az adatbázisban
- updated_at: timestamp, a rekord legutóbbi módosítása az adatbázisban

A backend és a desktop/konzolos alkalmazások ugyanazt a közös adatbázist használják.
Az alkalmazás csak belső, zárt hálózaton lesz elérhető, ezért autentikációt nem kell megvalósítani.

Szoftverfejlesztő és -tesztelő technikusi szakma – 5-0613-12-03	
Asztali- és webes szoftverfejlesztés, adatbázis-kezelés	**Időtartam: 240 perc**

1. feladat Backend alkalmazás

A feladatot egy webes programozási nyelvben kell megvalósítani. Ajánlott egy MVC keretrendszer, ill. egy hozzá tartozó ORM keretrendszer használata (pl. TypeScript/NestJS/TypeORM, PHP/Laravel/Eloquent, C#/ASP.NET/Entity Framework, Java/Spring Boot/Hibernate).

a) Készítsen egy üres projektet a választott backend keretrendszer segítségével. A projekt neve legyen **konyvtarbackend**! Töltse be a **books.sql** fájl tartalmát az adatbázis-kezelőbe!
b) Készítse el a kölcsönzéseket nyilvántartó **rentals** adattáblát, az alábbi oszlopokkal:
   - id: egész szám, a kölcsönzés azonosítója, elsődleges kulcs
   - book_id: egész szám, a hivatkozott könyv azonosítója, idegen kulcs (a books.id mezőre hivatkozik)
   - start_date: dátum, a kölcsönzés kezdete
   - end_date: dátum, a kölcsönzés vége

   Amennyiben a választott keretrendszer megköveteli, a tábla tartalmazhat még szükséges oszlopokat (pl. created_at stb.)
   Az adattáblát is az ORM keretrendszer segítségével hozza létre, ne SQL utasításokkal!
c) Hozza létre az adatbázis táblákhoz tartozó modell osztályokat! (Book és Rental)
d) A rentals táblát töltse fel véletlenül generált teszt-adatokkal, legalább 15 rekorddal! Ezt a backend keretrendszer seed-elés (vagy ekvivalens) funkciójával tegye meg!
   - Ha a keretrendszer nem rendelkezik ilyen funkcióval, elfogadható egy „POST /seed” végpont is, ami elvégzi az adatgenerálást.
e) Készítse ez az alábbi API végpontokat!
   Minden végpont JSON adatformában adja vissza a kimenetet.
   Hiba esetén a hiba okát jelezze:
   - A HTTP státusz kóddal, valamint
   - Egy JSON objektum segítségével szövegesen is
     - A keretrendszer által generált hiba-válaszok is megfelelők, amennyiben a feltételeknek megfelelnek.

   - **GET /api/books**
     Adja vissza az összes könyv alábbi 5 adatát: id, title, author, publish_year, page_count
     Az eredmény egy objektumokból álló lista legyen. Egy lehetséges megoldás:

```
{
      "data": [
            {
                  "id": 1,
                  "title": "Le Petit Prince",
                  "author": "Antoine de Saint-Exupéry",
                  "publish_year": 1943,
                  "page_count": 96
            },
            {
                  "id": 3,
                  "title": "Winnie-the-Pooh",
                  "author": "A. A. Milne",
                  "publish_year": 1926,
                  "page_count": 116
            }
      ]
}
```

- **POST /api/books**
  Hozzon létre egy új könyvet.
  A kérés törzse egy JSON objektum, amely tartalmazza az alábbi mezőket: title, author, publish_year, page_count. Pl.:

```
{
      "title": "Le Petit Prince",
      "author": "Antoine de Saint-Exupéry",
      "publish_year": 1943,
      "page_count": 96
}
```

  A végpont ellenőrizze, hogy a bemeneti adatok megfelelők-e:
  - Minden mező megadása kötelező
  - A kiadási év egész szám
  - Az oldalszám pozitív egész szám

  Validációs hiba esetén adjon vissza egy JSON objektumot, amely leírja a hiba okát, valamint egy megfelelő 4xx-es státusz kódot.
  Siker esetén adja vissza az új könyvet leíró JSON objektumot (l. GET /api/books végpontnál leírtakat), amiben szerepeljen az adatbázis-kezelő által generált id is! A státusz kód a **201 Created** legyen.
- **POST /api/books/{id}/rent**
  Könyv kölcsönzése: foglaljon le egy könyvet egy hétre, azaz hozzon létre egy új Rental rekordot, ahol:
  - A kezdődátum az aktuális dátum
  - A kölcsönzés vége a jelenlegi dátumhoz képest egy hét
  - A könyv azonosítója az URL-ben szereplő ID.

  A kérésnek nincs törzse, az {id} paraméter pedig egy egész szám, amely egy könyv azonosítóját jelenit.
  Ha nincs ilyen azonosítójú könyv, a végpont ezt **404 Not Found** státusz kóddal jelezze.
  Ha a könyv már foglalt (azaz már létezik foglalás az adott könyvre, amelynek az intervallumába beleesik az aktuális dátum), akkor a végpont ezt jelezze **409 Conflict** HTTP státusz kóddal, valamint a JSON kimeneten jelezze szövegesen is a hiba okát.
  Siker esetén JSON formátumban jelezze a foglalás kezdetét és végét:

```
{
      "id": 43,
      "book_id": 3,
      "start_date": "2022-04-01",
      "end_date": "2022-04-08"
}
```

Szoftverfejlesztő és -tesztelő technikusi szakma – 5-0613-12-03	
Asztali- és webes szoftverfejlesztés, adatbázis-kezelés	**Időtartam: 240 perc**

2. feladat Frontend alkalmazás

A feladatot JavaScript programozási nyelvben (vagy JavaScript-re forduló nyelvben) kell megvalósítani, egy frontend keretrendszer segítségével (pl. Vue.js, React, Angular). CSS keretrendszer (pl. Bootstrap, Tailwind) használata megengedett.
A feladat teljeskörű elkészítéséhez szükség van az 1. feladatban elkészített backend API végpontokra. Amennyiben valamelyiket nem tudta elkészíteni, a frontend alkalmazásban ettől függetlenül hívja meg a végpont URL-jét, az alkalmazás azonban dolgozhat tovább teszt adatokkal.
Az alkalmazás Egyoldalas Alkalmazás (Single Page Application) legyen, vagyis egyetlen funkció se járjon teljes oldal-újratöltéssel vagy böngésző navigációval.
A feladat elkészítése során, ahol lehetséges, használjon szemantikus HTML tag-eket CSS class-ok és id-k helyett!

a) Készítsen egy üres projektet a választott frontend keretrendszer segítségével. A projekt neve legyen **konyvtarfrontend**!
b) Az alkalmazás betöltésekor kérdezze le az /api/books végpont segítségével a könyvek adatait. Az így lekért könyvek alábbi adatait jelenítse meg:
   - Cím
   - Szerző
   - Kiadási év
   - Hossz
   - A szerzőt jelképező fénykép

   A fényképeket a mellékelt **szerzok.zip** tömörített fájlban találja, ezeket másolja egy olyan mappába, amely a frontend alkalmazás számára elérhető. A kép megnevezése minden esetben **szerzőnev.jpg**, pl. „Sabina O'Connell” szerző esetében a fájlnév „Sabina O'Connell.jpg”.
   Az adatok megjelenítésekor az alábbi szempontokat vegye figyelembe:
   - A könyvek címe HTML címsorként szerepeljenek!
   - Reszponzivitás: desktop nézetben három, tablet nézetben kettő, mobil nézetben egy könyv adata jelenjen meg soronként!
   - A könyvek adatai vizuálisan különüljenek el egymástól (pl. szegély vagy térköz segítségével)!
   - A megjelenítésre egy mintát talál a feladatsor végén.
c) A táblázat alatt jelenítsen meg egy űrlapot, amely segítségével a fenti 4 adat megadható (új kép feltöltését nem kell megvalósítani), valamint egy „Új könyv” feliratú gombot. A gombra kattintáskor:
   - Próbáljon meg létrehozni egy új könyvet a megadott adatokkal, az /api/books végpont használatával!
   - Amennyiben a backend alkalmazás validációs hibát jelez, ezeket jelenítse meg a felhasználónak! A hibaüzenetet formázza meg úgy, hogy a hibaüzenet jellege egyértelmű legyen!
   - Sikeres létrehozás esetén töltse újra a táblázatot (hogy a legfrissebb adatokkal dolgozhasson), és az űrlapot állítsa alaphelyzetbe.
   - A bemeneti mezők típusai legyenek az adott adattípusnak megfelelők!
d) A táblázatot egészítse ki „Kölcsönzés” feliratú gombokkal, amely minden könyv kártyájában/cellájában jelenjen meg! A gombra kattintva hívja meg a /api/books/{id}/rent végpontot. Siker esetén az oldal tetjén jelenjen meg egy „Sikeres foglalás!” felirat, ellenkező esetben pedig írja ki a backend által visszaadott hibaüzenet.
e) Egészítse ki az alkalmazást:
   - Egy fejléccel, amely tartalmazza:
     - Az alkalmazás megnevezését: Petrik Könyvtár Nyilvántartó (ez legyen HTML címsor) Ez legyen a teljes oldal címe is!
     - Egy vízszintes navigációs sávot, amely két linket tartalmazzon: Új könyv felvétele – görgessen le az „Új könyv” űrlaphoz Petrik honlap – a https://petrik.hu/ weboldalra mutasson
   - Egy lábléccel, amelyben szerepeljen az alkalmazás készítőjének (azaz az Ön) neve.

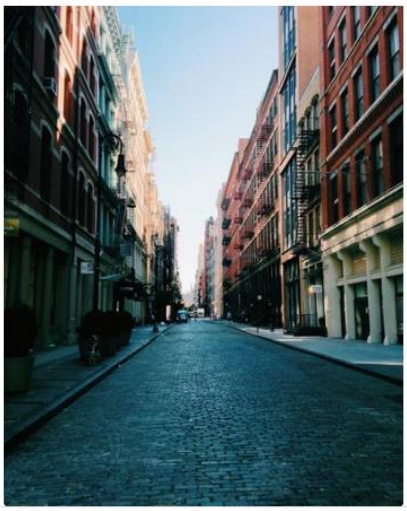
Szoftverfejlesztő és -tesztelő technikusi szakma – 5-0613-12-03	
Asztali- és webes szoftverfejlesztés, adatbázis-kezelés	**Időtartam: 240 perc**

Új könyv felvétele Petrik honlap

Petrik Könyvtár Nyilvántartó

Hordómese
Jonathan Swift
Kiadási év: 1991
Hossz: 187 oldal

Robinson Crusoe
Daniel Defoe
Kiadási év: 1850
Hossz: 201 oldal



Mindenmindegy Jakab
Denis Diderot
Kiadási év: 1968
Hossz: 71 oldal

Készítette: Ifj. Minta Vizsga

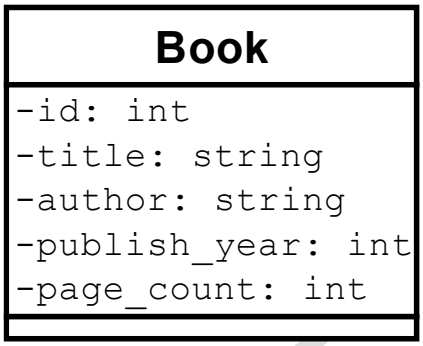
Szoftverfejlesztő és -tesztelő technikusi szakma – 5-0613-12-03	
Asztali- és webes szoftverfejlesztés, adatbázis-kezelés	**Időtartam: 240 perc**

3. feladat Konzolos alkalmazásrész, az adatszerkezet kialakítása

A feladatot C# vagy Java programozási nyelvben kell megvalósítani. A megoldáshoz szükséges az 1. feladatban létrehozott és kiegészített adatbázis.

a) Hozzon létre egy új projektet **KonyvtarAsztali** néven!

b) Hozzon létre egy Book nevű osztályt egy könyv adatainak a kezeléséhez az alábbi osztálydiagramm alapján:



- Az adattagokhoz készítsen get property-ket, vagy getter metódusokat!
- Hozzon létre a **Book** osztályban paraméteres konstruktort, amely a paraméterek értékével inicializálja az adattagokat!
- Amennyiben későbbi feladatok megoldásához szükséges, úgy bővítse az osztályt megfelelő adattagokkal és metódusokkal.

c) Hozzon létre egy **Statistics** nevű osztályt a konzolos feladatok elvégzéséhez!

- A program indításakor amennyiben `--stat` parancssori argumentum lett megadva úgy a konzolos alkalmazásrész induljon el.
  - Amennyiben nem tudja kezelni a parancssori argumentumokat úgy a konzolos alkalmazásrész számára külön projektet hozzon létre **KonyvtarAsztaliKonzol** néven
- Az osztály rendelkezzen egy `books` nevű adattaggal, amely egy **Book** típusú objektumokat tartalmazó lista, amely lehetővé teszi a könyvtár összes könyvének a kezelését.
- Írjon a **Statistics** osztályban függvényt, amely beolvassa az adatbázisban lévő könyveket, és a beolvasott adatok alapján feltölti a books listát a könyvekkel.
- Ha nem sikerül kapcsolódni az adatbázishoz, a program adjon hibaüzenetet, és a program futása szakadjon meg!
- Hozzon létre a **Statistics** osztályban konstruktort vagy statikus függvényt, amely végrehajtja a beolvasást, majd elvégzi a további részfeladatokat.

d) Hozzon létre eljárásokat és függvényeket a Statistics osztályban az alábbi részfeladatok elvégzéséhez. Az eredményt írja ki a konzolra a mintának megfelelően.

- Határozza meg az 500 oldalnál hosszabb könyvek számát.
- Döntse el, hogy szerepel-e az adatok között 1950-nél régebbi könyv.
- Határozza meg és írja ki a leghosszabb könyv adatait.
- Határozza meg és írja ki a legtöbb könyvvel rendelkező szerzőt.
- Kérjen be a konzolról egy könyv címet. Határozza meg, hogy ki az adott könyv szerzője. Ha a megadott címmel nem szerepel könyv, akkor „Nincs ilyen könyv” üzenet jelenjen meg.
- Minta:

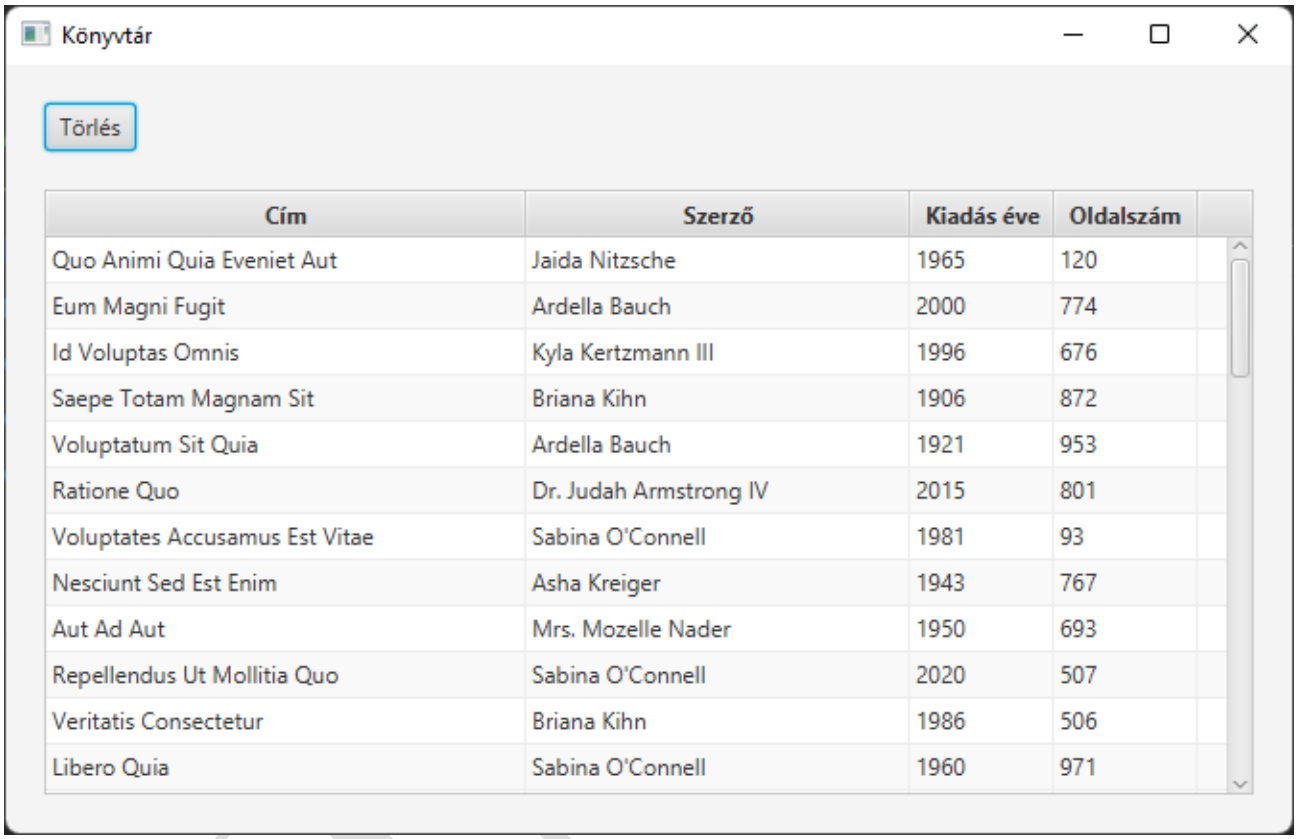
```
500 oldalnál hosszabb könyvek száma: 35
Van 1950-nél régebbi könyv
A leghosszabb könyv:
      Szerző: Kyla Kertzmann III
      Cím: Libero Voluptas Unde Iure
      Kiadás éve: 1959
      Oldalszám: 1000
A legtöbb könyvvel rendelkező szerző: Briana Kihn
Adjon meg egy könyv címet: Quo Animi Quia Eveniet Aut
Az megadott könyv szerzője Jaida Nitzsche
```

Szoftverfejlesztő és -tesztelő technikusi szakma – 5-0613-12-03	
Asztali- és webes szoftverfejlesztés, adatbázis-kezelés	**Időtartam: 240 perc**

4. feladat Grafikus alkalmazásrész

A megoldást a 3. feladat folytatásaként kell elvégezni, vagyis a két feladat végeredménye egyetlen projekt legyen!

a) A projekt grafikus megjelenítést végző osztályában hozza létre a felület elemeit úgy, hogy az alábbi mintához hasonló megjelenítést tegyen lehetővé!



- Az ablak bal felső sarkában helyezzen el egy gombot „Törlés” felirattal
- Helyezzen el a gomb alá egy tároló komponenst, amelybe a könyvek adatait táblázatos formában tudja listázni.
  - A megjelenítésre szolgáló komponensnek nem kötelező rendelkeznie fejléccel.

b) Az alkalmazásrész indulásakor töltse fel a listát az adatbázisban lévő könyvek adataival.
- A listázáshoz használja fel az előző feladatban létrehozott **Book** osztályt.
- Ha nem sikerül kapcsolódni az adatbázishoz, a program felugró ablakban adjon hibaüzenetet. A felugró ablak bezárásakor a teljes program álljon le.

c) Tegye lehetővé a könyvek törlését!
- A „Törlés” gombra kattintva a listából kiválasztott könyv kerüljön eltávolításra az adatbázisból.
- Amennyiben nincs könyv kiválasztva akkor felugró ablakban jelenjen meg „Törléshez előbb válasszon ki könyvet” üzenet.
- Ha ki lett választva könyv, akkor a törlés előtt jelenjen meg egy megerősítő ablak „Biztos szeretné törölni a kiválasztott könyvet?” felirattal.
- A törlést csak akkor hajtsa végre, ha a felhasználó a felugró ablakon megfelelő gombra kattintott.
- A törlés sikerességéről vagy sikertelenségéről adjon visszajelzést. Sikertelen törlés esetén megfelelő hibaüzenetet jelenítsen meg.
- A sikeresen eltávolított könyv a listából is kerüljön törlésre.

Szoftverfejlesztő és -tesztelő technikusi szakma – 5-0613-12-03	
Asztali- és webes szoftverfejlesztés, adatbázis-kezelés	**Időtartam: 240 perc**

ÉRTÉKELŐ LAP

Vizsgázó neve	.....

1. feladat – Backend alkalmazás	**15**	
a) Létrehozta a projektet, ***KonyvtarBackend*** néven, az adatbázist betöltötte.	1	
b) **Rentals** adattábla		
– Létrehozta **rentals** adattáblát a megfelelő oszlopokkal és adattípusokkal	1	
– A projekt tartalmazza a hozzá tartozó adatbázis-migrációt	1	
– A táblát lehet seed-elni, amely min. 15 véletlen adatot tartalmazó rekordot hoz létre	2	
c) Létrehozta a Book és Rental osztályokat	2	
d) Listázás API végpont		
– A megfelelő URL-en elérhető, a kimenet JSON formátumú	1	
– Az adatbázisban található könyvek adatait adja vissza	1	
e) Új könyv API végpont		
– A megfelelő URL-en elérhető, a kimenet JSON formátumú	1	
– A bemenetet validálja, hiba esetén JSON formátumban jelzi a hiba okát, a HTTP státusz megfelelő	1	
– Sikeres létrehozás esetén visszaadja az új könyv adatait, a HTTP státusz megfelelő	1	
f) Kölcsönzés API végpont		
– A megfelelő URL-en elérhető, a kimenet JSON formátumú	1	
– JSON formátumban jelzi a hibás foglalásokat, a http státusz megfelelő	1	
– A kimenet tartalmazza az újonnan létrehozott foglalás adatait	1	

2. feladat – Frontend alkalmazás	**25**	
2.1 – Az alkalmazás működése	**15**	
a) Létrehozta a projektet, ***KonyvtarFrontend*** néven, az alkalmazás szintaktikai hiba mentes, fordítható.	1	
b) Könyvek listázása		
– A könyvek kért adatai megjelennek a böngészőben	2	
– Minden könyvnél megjelenik a szerző fényképe	1	
– Az adatokat a backend API végpont segítségével töltötte be	1	
c) Új könyv form		
– A táblázat alatt megjelenik négy megfelelő típusú input mező. Form tag nem szükséges	1	
– A form elküldi az adatait a backend API-nak a megfelelő formátumban	2	
– Hiba esetén megjelenik a validációs hibaüzenet	1	
d) Foglalás gomb		
– Megjelenik minden könyv cellájában	1	
– A gombra kattintva meghívja a megfelelő backend végpontot	1	
– Hiba esetén megjelenik a hibaüzenet	1	
e) Fejléc, lábléc		
– A fejléc tartalmazza az oldal nevét és a kért linkeket	1	
– A láblécben szerepel a tanuló neve	1	
– A <title> tag is tartalmazza az oldal nevét	1	
2.2 – Reszponzív, szemantikusan helyes weboldal	**10**	
f) Szemantikus HTML		
– A fejléc <header>, a lábléc <footer> tag-et használ	1	

– Az oldal fő tartalma használja <main>, <section>, <article> tag-ek egyikét	1	
– A navigációs linkek <nav> tag-en belül találhatók	1	
– Az oldal címe <h1>, a könyvek címe <h1>-nél alacsonyabb szintű heading	1	
g) Formázás, reszponzív megjelenés		
– Mobil nézetben egymás alatt helyezkednek el a könyvek adatai	1	
– Tablet nézetben soronként 2 helyezkedik el	1	
– Desktop nézetben soronként 3 helyezkedik el	1	
– A kártyák térközzel vagy szegéllyel elkülönülnek egymástól	1	
– A navigációs linkek egy sorban találhatók	1	
– A hibaüzenetek betűszínnel, háttérszínnel stb. elkülönülnek a többi tartalomtól	1	

3. feladat – Az adatszerkezet kialakítása, konzolos alkalmazásrész elkészítése	**15**	
a) Létrehozta a projektet, ***KonyvtarAsztali*** néven. A program szintaktikai hiba mentes, futtatható.	1	
b) Létrehozta a **Book** osztályt. Helyesen definiálta az osztály adattagjait és létrehozott az adattagok számára gettereket. Létrehozott paraméteres konstruktort az adattagok inicializálásához.	1	
c) Konzolos alkalmazásrész - **Statistics** osztály		
– Létrehozta a **Statistics** osztályt	1	
– A konzolos alkalmazásrész a --stat parancssori argumentum megadásával indul. Ellenkező esetben a grafikus alkalmazásrész indul el.	1	
– Az osztály rendelkezik egy függvénnyel, amely lekérdezi és eltárolja az adatbázisban tárolt könyveket, egy books nevű lista adattagba.	1	
– Ha nem sikerül kapcsolódni az adatbázishoz, akkor hibaüzenet jelenik meg a konzolon és a program futása megszakad	1	
d) Statisztikai feladatok		
– A részfeladatok számára külön eljárásokat és függvényeket hoz létre. Minden függvénynek egy meghatározott feladata van.	1	
– A kiírások a mintának megfelelőek	1	
– Helyesen határozza meg az 500 oldalnál hosszabb könyvek számát	1	
– Helyesen dönti el, hogy szerepel-e az adatok között 1950-nél régebbi könyv.	1	
– Helyesen határozza meg a leghosszabb könyvet	1	
– Helyesen határozza meg a legtöbb könyvvel rendelkező szerzőt	2	
– Bekér a konzolról egy könyv címet, ezt eltárolja a feladat megoldásához.	1	
– A bekért könyv cím alapján megfelelően határozza meg, hogy hányszor lett a könyv kikölcsönözve.	1	

Szoftverfejlesztő és -tesztelő technikusi szakma – 5-0613-12-03	
Asztali- és webes szoftverfejlesztés, adatbázis-kezelés	**Időtartam: 240 perc**

4. feladat – Grafikus alkalmazásrész	**10**	
a) Vizuális felület kialakítása		
– Elhelyezett egy nyomógombot a kiválasztott könyv törléséhez	1	
– Felvett egy megfelelő komponenst (Java: ListView, TableView / C#: ListBox, ListView, DataGrid) az könyvek táblázatszerű megjelenítéséhez	1	
b) Adatok listázása		
– A program indulásakor a feltölti a listát az adatbázisból kiolvasott könyvekkel	1	
– Ha nem sikerül kapcsolódni az adatbázishoz, akkor hibaüzenet jelenik meg egy felugró ablakban, a hibaüzenetet bezárásával az alkalmazás leáll	2	
c) Könyv törlése		
– Gombkattintásra, ha nincs kiválasztva könyv, akkor felugró ablakban hibaüzenet jelenik meg	1	
– Ha ki lett választva könyv, akkor felugróablak jelenik meg a művelet megerősítéséhez.	1	
– Ha a felhasználó azt választotta, hogy biztos szeretné törölni a könyvet akkor a könyv törlésre kerül az adatbázisból, ellenkező esetben nem történik semmi.	1	
– A törlés sikerességéről vagy sikertelenségéről a felugróablakban figyelmeztet. Sikertelen törlés esetén kiírja annak okát is.	1	
– Törlés után a lista tartalma frissül.	1	

Pontszám	**65**	

Javítás Dátuma: 2024.	**Érdemjegy** .....

Vizsgabizottsági tag aláírás .....