

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:
 - o a gépre telepített offline help rendszerek
 - o 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:**
 - o members.sql – a „members” adatbázistábla szerkezete és kezdeti tartalma
 - o kepek.zip – a feladat megoldásához szükséges képek
- **Beadandó:** az összes projekt, a megoldások során létrejött kimeneti állományok, illetve a megoldás során létrehozott adatbázis exportja.
 - o Mindezeket egyetlen tömörített fájlban tölts fel Teams feladat beadásaként!
 - o Az állomány neve szoftvervizsga2022proba_sajatnev_osztaly legyen!
 - o A beadott alkalmazások futtatható állapotúak legyenek!
 - o A hibás vagy hiányos részeket kommentben hagyja a kódban, de jelezze, hogy az is a megoldás része!
 - o A gyorsabb értékeléshez az elkészített programot futtatható állapotban hagyja megnyitva a számítógépén!

FELADATLEÍRÁS

Egy könyvklub tagjait nyilvántartó rendszert kell elkészítenie, amely tagok, ill. a tagdíj befizetések 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 tagok listáját az alábbi „members” adattábla definiálja:

- id: egész szám, a tag azonosítója, elsődleges kulcs, automatikusan kap értéket
- name: szöveg, a tag azonosítója
- gender: karakter, az értéke lehet 'M' (férfi), 'F' (nő), vagy NULL (nem kívánta megadni)
- birth_date: születési dátum
- banned: logikai, a tag ki van-e tiltva a klubból vagy sem
- created_at: timestamp, a tag csatlakozási ideje
- 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.

1. feladat Backend alkalmazás

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

- a) Készítsen egy üres projektet a választott backend keretrendszer segítségével. A projekt neve legyen **bookclubbackend**! Töltse be a **members.sql** fájl tartalmát az adatbázis-kezelőbe!
- b) Készítse el a tagdíj befizetéseket nyilvántartó **payments** adattáblát, az alábbi oszlopokkal:
 - **id**: egész szám, a befizetés azonosítója, elsődleges kulcs
 - **member_id**: egész szám, a hivatkozott tag azonosítója, idegen kulcs (a members.id mezőre hivatkozik)
 - **amount**: a befizetett összeg, egész szám
 - **paid_at**: a befizetés ideje, dátum/idő

Amennyiben a választott keretrendszer megköveteli, a tábla tartalmazhat még szükséges oszlopokat (pl. **created_at** stb.)

Az adattáblát adatbázismigráció segítségével hozza létre!

- c) Hozza létre az adatbázis táblákhoz tartozó modell osztályokat! (Member és Payment)
- d) A payments táblát tölts fel véletlenül generált teszt-adatokkal, legalább 15 rekorddal! Ezt a backend keretrendszer seed-elés (vagy ekvivalens) funkciójával tegye meg!
- 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
 - o A keretrendszer által generált hiba-válaszok is megfelelőek, amennyiben a feltételeknek megfelelnek.

- **GET /api/members**

Adja vissza az összes tag alábbi 5 adatát: **id**, **name**, **gender**, **birth_date**, **created_at**

Az eredmény egy objektumokból álló lista legyen. Egy lehetséges megoldás:

```
{
  "data": [
    {
      "id": 1,
      "name": "Hajdu Mátyás",
      "gender": "M",
      "birth_date": "2007-02-12",
      "created_at": "2022-01-03T07:17:01Z"
    },
    {
      "id": 2,
      "name": "ifj. Balogh Endrené",
      "gender": null,
      "birth_date": "2007-01-04",
      "created_at": "2022-04-01T02:31:09Z"
    },
    {
      "id": 5,
      "name": "id. Török Zsóka PhD",
      "gender": "F",
      "birth_date": "1978-01-06",
      "created_at": "2022-03-09T15:15:21Z"
    }
  ]
}
```

- **POST /api/members**

Hozzon létre egy új könyvklub tagot.

A kérés törzse egy JSON objektum, amely tartalmazza az alábbi mezőket: name, gender (opcionális), birth_date. Pl.:

```
{
  "name": "Török Géza",
  "gender": "M",
  "birth_date": "1997-04-04",
}
```

A végpont ellenőrizze, hogy a bemeneti adatok megfelelők-e:

- A name és a birth_date mezők megadása kötelező
- A birth_date dátum formátumú
- A gender mező, ha létezik, 'M' vagy 'F' karakterek egyikének kell lennie!

Az adattábla created_at mezője az aktuális dátum/idő legyen!

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 tagot leíró JSON objektumot (l. GET /api/members végpontnál leírtakat), amiben szerepeljen az adatbázis-kezelő által generált id is! A státusz kód a **201 Created** legyen.

- **POST /api/members/{member}/pay**

Tagdíj befizetés rögzítése: fizesse be egy tag tagdíját, azaz hozzon létre egy új Payment rekordot, ahol:

- A paid_at az aktuális dátum
- A fizetés összege 5000 Ft
- A tag azonosítója az URL-ben szereplő ID.

A kérésnek nincs törzse, az {member} paraméter pedig egy egész szám, amely egy tag azonosítóját jelenti.

Ha nincs ilyen azonosítójú tag, a végpont ezt **404 Not Found** státusz kóddal jelezze.

Ha a tag már fizetett az elmúlt 30 napban (azaz már létezik befizetés, amelyet az adott tag végzett, és a fizetési dátuma az aktuális dátumhoz képesti 30 napba esik), 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 vissza a befizetett összeget, és a fizetés idejét:

```
{
  "id": 43,
  "member_id": 3,
  "amount": 5000,
  "paid_at": "2022-04-07T11:01:58Z"
}
```

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-újrátöltéssel 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 **bookclubfrontend!**
- b) Az alkalmazás betöltésekor kérdezze le az `/api/members` végpont segítségével a tagok adatait. Az így lekért tagok alábbi adatait jelenítse meg:
 - Név
 - Születési dátum
 - A tag csatlakozási ideje
 - A tag nemét jelképező kép

A képeket a mellékelt **kepek.zip** tömörített fájlban találja, ezeket másolja egy olyan mappába, amely a frontend alkalmazás számára elérhető. A három kép: **male.png** (férfi), **female.png** (nő), **other.png** (nincs megadva).

Az adatok megjelenítésekor az alábbi szempontokat vegye figyelembe:

 - A tagok nevei HTML címsorként szerepeljenek!
 - Reszponzivitás: desktop nézetben három, tablet nézetben kettő, mobil nézetben egy tag adata jelenjen meg soronként!
 - A tagok 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 tagfelvételi űrlapot, amely segítségével a szükséges 3 adat megadható, valamint egy „Tagfelvétel” feliratú gombot. A gombra kattintáskor:
 - Próbáljon meg létrehozni egy új tagot a megadott adatokkal, az `/api/members` 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 megfelelőek!
- d) A taglistát egészítse ki „Tagdíj befizetés” feliratú gombokkal, amely minden tag kártyájában/cellájában jelenjen meg! A gombra kattintva hívja meg az `/api/members/{member}/pay` végpontot. Sikeres esetén a gomb alatt jelenjen meg egy „Sikeres befizetés!” felirat, ellenkező esetben pedig írja ki a backend által visszaadott hibaüzenet.
- e) Egészítse ki az alkalmazást:
 - Egy fejléccel, amely tartalmazza:
 - o Az alkalmazás megnevezését: Petrik Könyvklub (ez legyen HTML címsor)
Ez legyen a teljes oldal címe is!
 - o Egy vízszintes navigációs sávot, amely két linket tartalmazzon:
Új tag felvétele – görgessen le a „Tagfelvétel” űrlaphoz
Petrik honlap – <https://petrik.hu/> weboldalra mutasson
 - Egy lábléccel, amelyben szerepeljen az alkalmazás készítőjének (azaz az Ön) neve.

[Új tag felvétele](#) [Petrik honlap](#)

Petrik Könyvklub

Hajdu Mátyás

Született: 2007-02-12

Csatlakozott: 2022-01-03T07:17:01Z



ifj. Balogh Endrené

Született: 2007-01-04

Csatlakozott: 2022-04-01T02:31:09Z



id. Török Zsóka PhD

Született: 1978-01-06

Csatlakozott: 2022-03-09T15:15:21Z

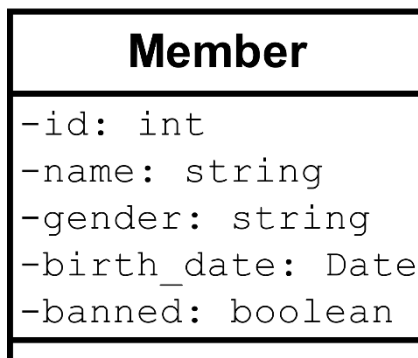


Készítette: Ifj. Minta Vizsga

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.

- Hozzon létre egy új projektet **Bookclub-desktop** néven!
- Hozzon létre egy **Member** osztályt a klubtagok adatainak kezeléséhez az alábbi osztálydiagramm alapján:



- Az adattagokhoz készítsen get property-ket, vagy getter metódusokat!
 - Hozzon létre a **Member** osztályban paraméteres konstruktort, amely a paraméterek értékével inicializálja az adattagokat!
 - Amennyiben későbbi feladatok megoldásához szükséges, úgy bővítse az osztályt megfelelő adattagokkal és metódusokkal.
 - A dátum számára javasolt adattípus Java-ban LocalDate, C#-ban DateTime.
- Hozzon létre egy **Statisztika** osztályt a konzolos feladatok elvégzéséhez!
 - A program indításakor amennyiben --stat parancssori argumentum lett megadva úgy a konzolos alkalmazásrész induljon el.
 - o Amennyiben nem tudja kezelni a parancssori argumentumokat úgy a konzolos alkalmazásrész számára külön projektet hozzon létre **Bookclub-desktop-cli** néven
 - Az osztály rendelkezzen egy members nevű adattaggal, amely egy **Member** típusú objektumokat tartalmazó lista, amely lehetővé teszi a klub összes tagjának a kezelését.
 - Írjon a **Statisztika** osztályban függvényt, amely beolvassa az adatbázisban lévő klubtagokat, és a beolvasott adatok alapján feltölti a „members” listát a klubtagokkal.
 - Ha nem sikerül kapcsolódni az adatbázishoz, a program adjon hibaüzenetet, és a program futása szakadjon meg!
 - Hozzon létre a **Statisztika** osztályban konstruktort vagy statikus függvényt, amely végrehajtja a beolvasást, majd elvégzi a további részfeladatokat.
 - Hozzon létre eljárásokat és függvényeket a Statisztika osztályban az alábbi részfeladatok elvégzéséhez. Az eredményt írja ki a konzolra a mintának megfelelően.
 - Határozza meg a kitiltott klubtagok számát.
 - Döntse el, hogy szerepel-e az adatok között 18 évnél fiatalabb személy.
 - Határozza meg és írja ki a legidősebb klubtag nevét és születésnapját.
 - Határozza meg és írja ki nemenként csoportosítva a tagok számát.
 - Kérjen be a konzolról egy nevet. Határozza meg, hogy az adott személy ki van-e tiltva a klubból. Ha a megadott névvel nem szerepel klubtag, akkor „Nincs ilyen tagja a klubnak” üzenet jelenjen meg.
 - Minta:

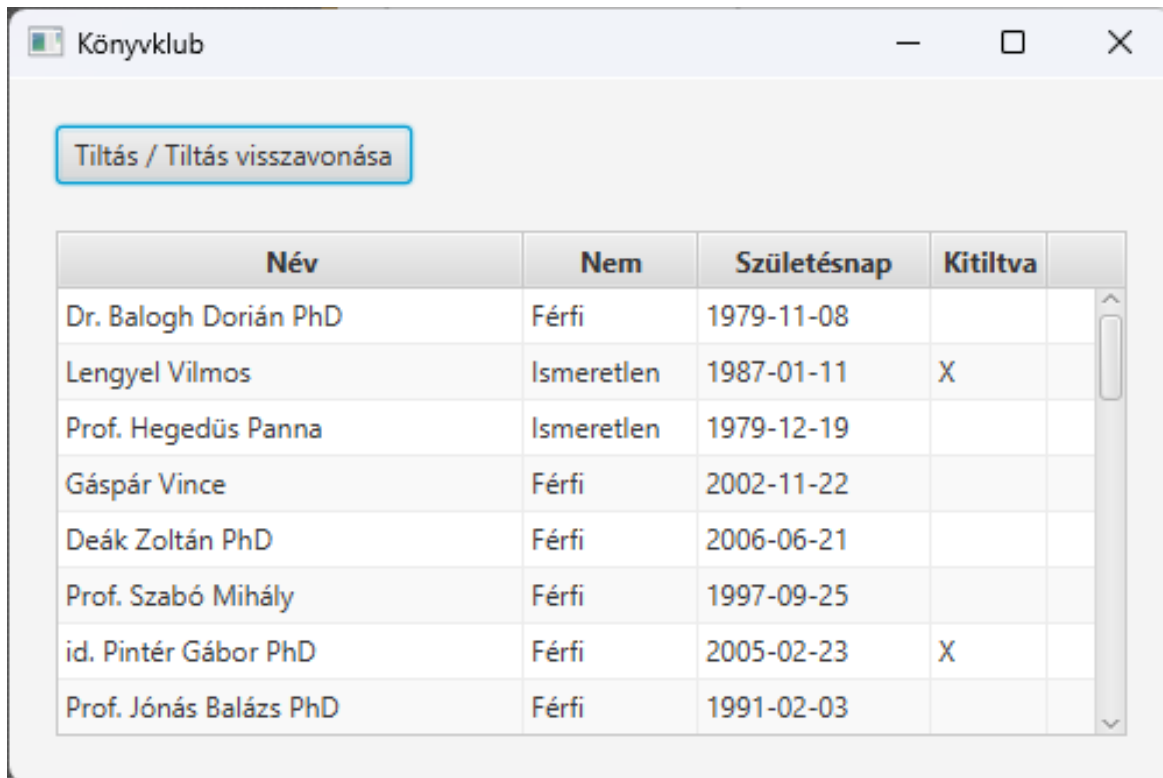
```

Kitiltott tagok száma: 4
Van a tagok között 18 évnél fiatalabb személy.
A legidősebb klubtag: Jakab Hanga (1972-06-14)
Tagok száma:
    Nő: 6
    Férfi: 14
    Ismeretlen: 10
Adjon meg egy nevet: Gáspár Vince
A megadott személy nincs kitiltva
  
```

4. feladat Grafikus alkalmazásrész – klubtagok listázása és kizárása

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 „Tiltás / Tiltás visszavonása” felirattal
- Helyezzen el a gomb alá egy tároló komponenst, amelybe a klubtagok adatait táblázatos formában tudja listázni.
 - o A megjelenítésre szolgáló komponensnek nem kötelező rendelkeznie fejléccel. Ebben az esetben a listában „X” helyett „kitiltva” felirat jelenjen meg kitiltás esetén.
- b) Az alkalmazásrész indulásakor töltse fel a listát az adatbázisban lévő klubtagok adataival.
 - A listázáshoz használja fel az előző feladatban létrehozott **Member** 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 tagok kitiltását és a tiltás visszavonását!
 - A „Tiltás / Tiltás visszavonása” gombra kattintva a listából kiválasztott klubtag kitiltásának értéke változzon (igazról hamisra vagy hamisról igazra).
 - Amennyiben nincs klubtag kiválasztva akkor felugró ablakban jelenjen meg „Tiltás módosításához előbb válasszon ki klubtagot” üzenet.
 - Ha ki lett választva klubtag, akkor az állapot módosítása előtt jelenjen meg egy megerősítő ablak, melynek felirata a kitiltás állapotától függően az alábbiak egyike legyen:
 - o „Biztos szeretné kitiltani a kiválasztott klubtagot?”
 - o „Biztos szeretné visszavonni a kiválasztott klubtag tiltását?”
 - A műveletet csak akkor hajtsa végre, ha a felhasználó a felugró ablakon megfelelő gombra kattintott.
 - A művelet sikerességéről vagy sikertelenségéről adjon visszajelzést. Sikertelen művelet esetén megfelelő hibaüzenetet jelenítsen meg.
 - A kitiltás vagy tiltás visszavonás esetén a listában is módosuljon az állapot.