**PROJEKTMUNKA**

Pályakövető rendszer weboldal

**Mentorok: Készítette:**

Szabó Ákos Menyhárt Martin

Dantesz Tamás Kiss Szabolcs

Labbancz Balázs

NYÍREGYHÁZA

2023

Tartalomjegyzék

[Feladat megadása 2](#_Toc133850665)

[1. Felhasználók és igényeik / Funkció lista 3](#_Toc133850666)

[1.1 Látogató 3](#_Toc133850667)

[1.2 Bejelentkezett felhasználó 3](#_Toc133850668)

[1.3 Adminisztrátor 3](#_Toc133850669)

[2. Screenek 4](#_Toc133850670)

[3. Modellek 8](#_Toc133850671)

[4. Alkalmazások kiválasztása 11](#_Toc133850672)

[4.1 Front-end 11](#_Toc133850673)

[4.2 Backend 13](#_Toc133850674)

[4.3 Adatbázis 14](#_Toc133850675)

[4.4 Tesztelés 14](#_Toc133850676)

[5. Routing 15](#_Toc133850677)

[6. Implementáció 16](#_Toc133850678)

[7. Tesztelés 16](#_Toc133850679)

[8. Bevezetés, éles üzemmód 17](#_Toc133850680)

[9. Összegzés 17](#_Toc133850681)

# **Feladat megadása**

Vizsgaremekünkben a Nyíregyházi SZC egyik új projektjét szeretnénk megvalósítani, melynek keretében egy középiskola utáni pályakövetési rendszert fogunk elkészíteni. Projektünkben egy weboldalt szeretnénk megvalósítani, amely minden felhasználója számára képes megjeleníteni, hogy ki, mikor és melyik középiskolát fejezte be és ezután hol állt munkába, vagy melyik egyetemen tanul tovább.

Fontos megemlítenünk, hogy felhasználó csak SZC dolgozó lehet, így regisztrációra nem lesz lehetőség és csak az adminisztrátori joggal rendelkezők vehetnek fel új felhasználókat. Ez azt is jelenti, hogy az oldal nem publikus és tartalma csak bejelentkezés után érhető el. A felhasználóknak lehetősége lesz módosítani az adatbázist és különböző szempontok alapján szűrni azt, majd az adatokat exportálni Excel táblázatba vagy .csv fájlba.

# **1. Felhasználók és igényeik / Funkció lista**

Az alkalmazásban a felhasználók tanárok lesznek, akiket az OM azonosítójuk azonosít. Regisztráció olyan szinten létezik, hogy csak az admin szintű felhasználók tudnak a tanároknak fiókot létrehozni, előre megadott jelszavakkal, amiket a felhasználó később bármikor megváltoztathat. A tanárok csakis a saját osztályaikat láthatják, amikkel bejelentkezés után az alábbi műveleteket hajthatják végre.

## Látogató

* + - * Legyen képes bejelentkezni

## Bejelentkezett felhasználó

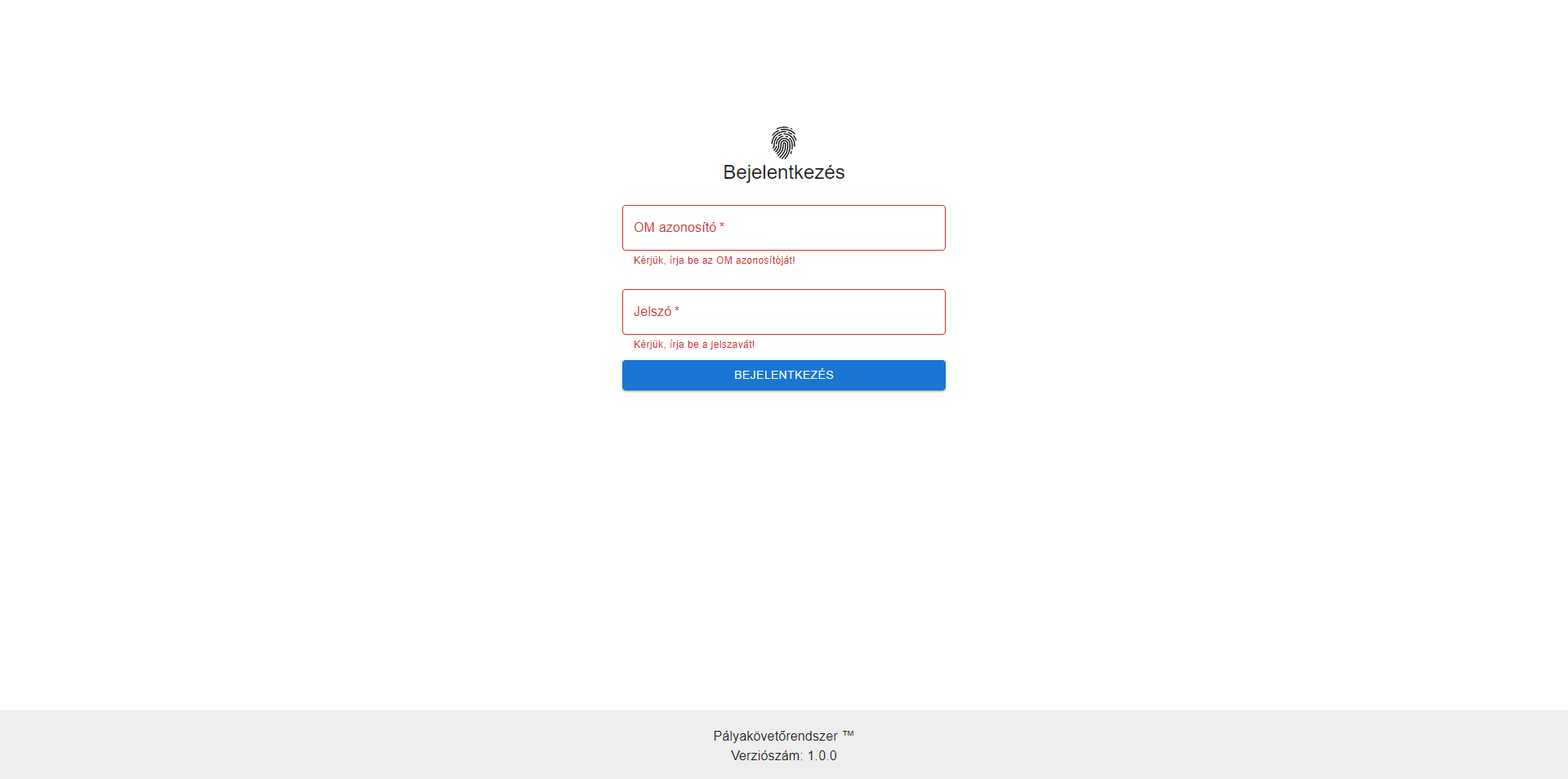
* + - * Legyen képes osztályt választani.
      * Legyen képes hozzáadni, módosítani, törölni, vagy különböző szempontok alapján szűrni a kiválasztott osztály tanulóit.
      * Legyen képes importálni CSV fájlból tanulókat a kiválasztott osztályba.
      * Legyen képes exportálni Excel és CSV fájlba kijelölt tanulókat és azok adatait a kiválasztott osztályból.
      * Legyen képes új, üres osztályt létrehozni.
      * Legyen képes jelszót változtatni.
      * Legyen képes kapcsolatfelvételre az oldal üzemeltetőjével.
      * Legyen képes sötét módba váltani az oldalt.
      * Legyen képes kijelentkezni.

## Adminisztrátor

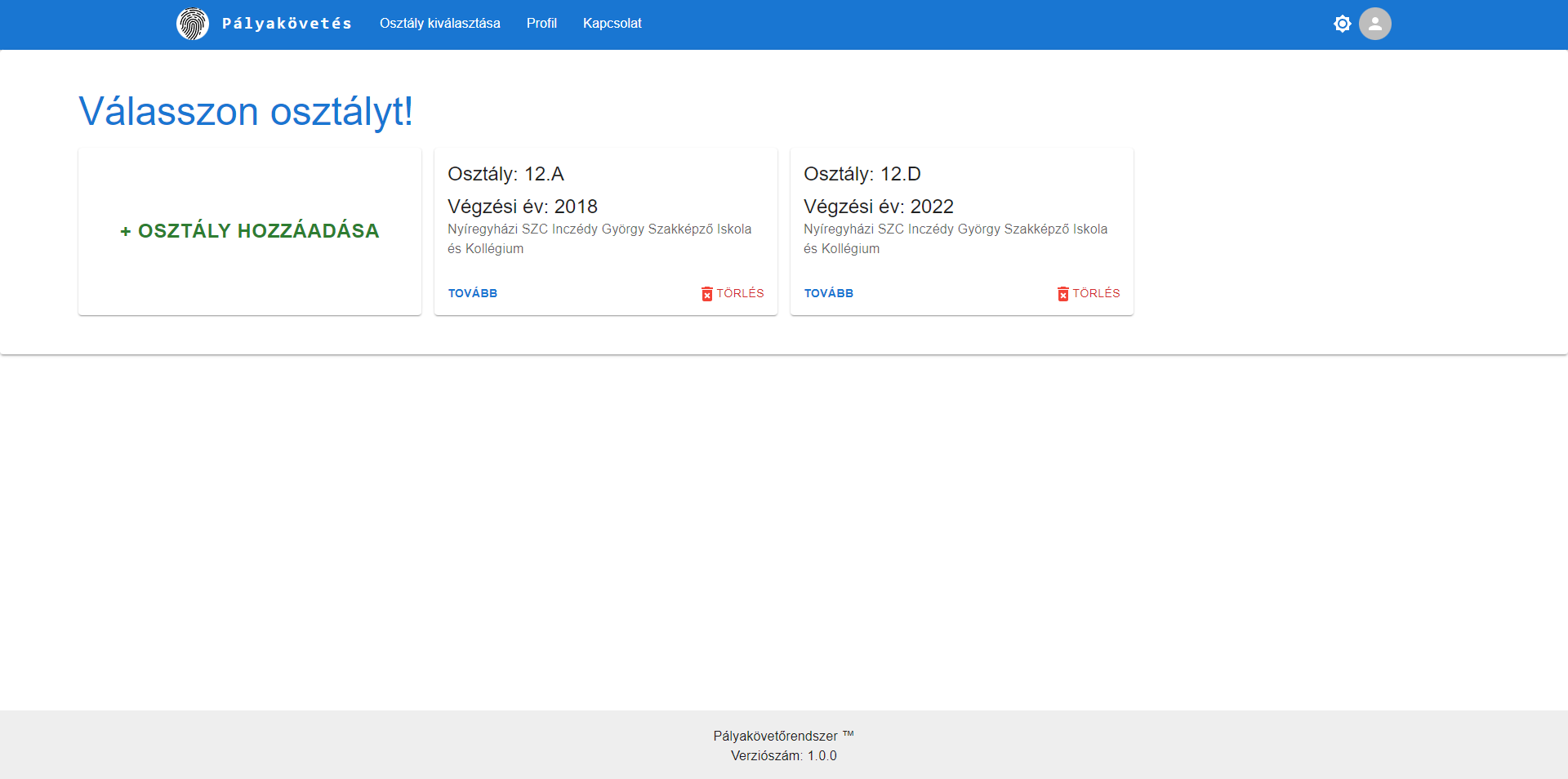
* + - * Legyen képes új felhasználókat felvenni.
      * Legyen képes felhasználókat törölni
      * Legyen képes saját jelszót változtatni.
      * Legyen képes sötét módba váltani az oldalt.

# **2**. **Screenek**

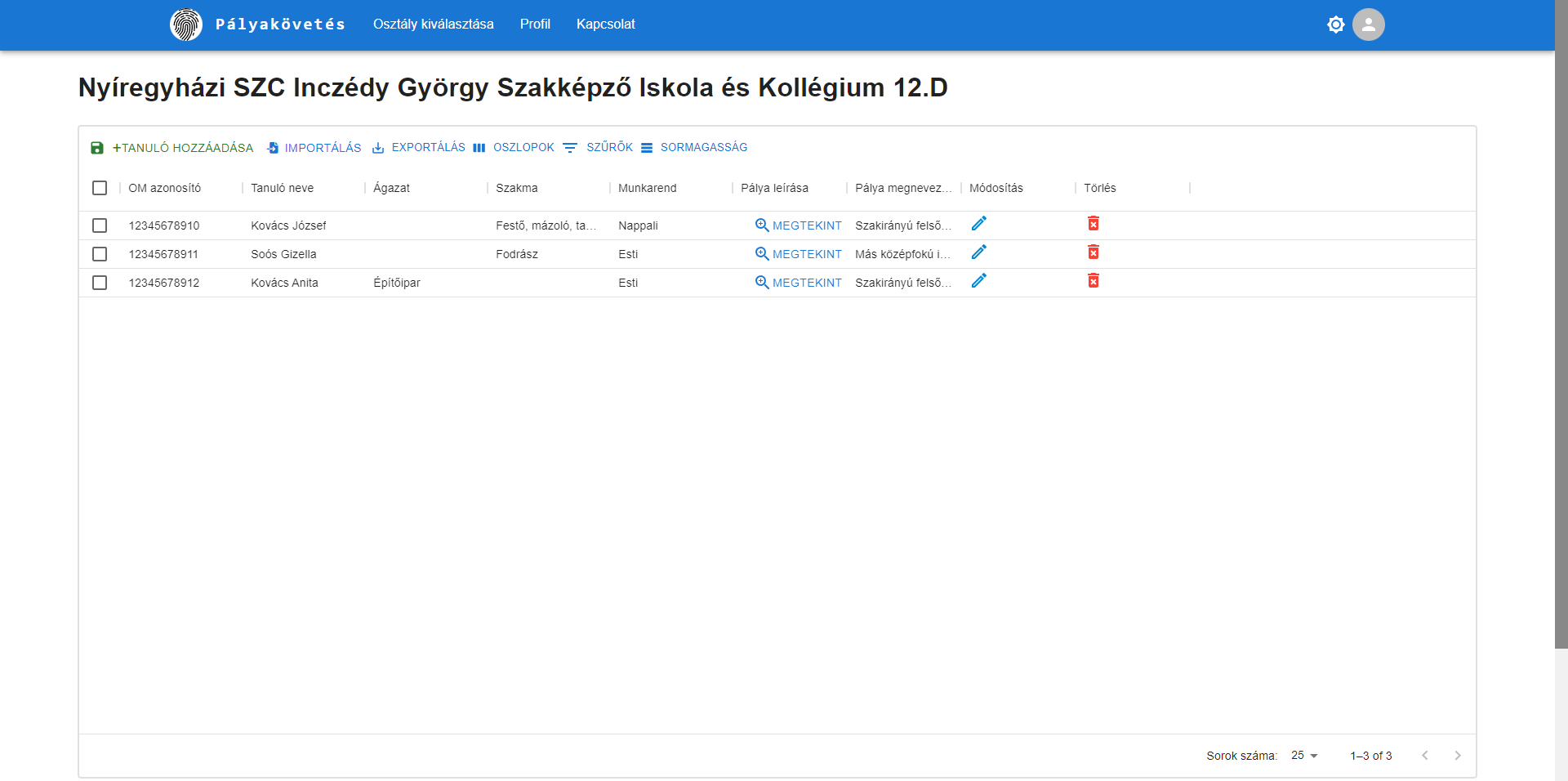
Letisztult megjelenésű bejelentkező oldal:



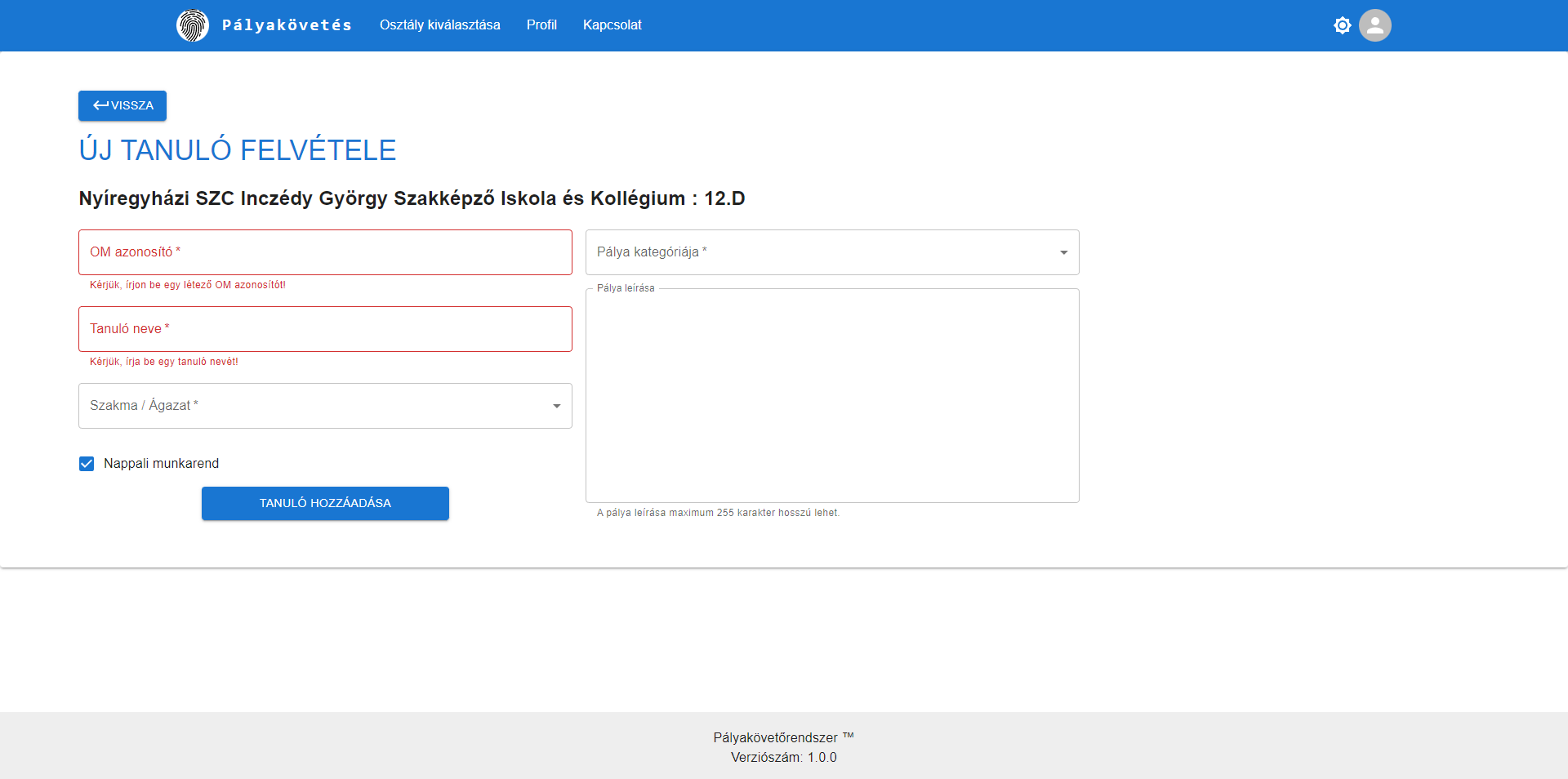
Bejelentkezés után az osztályválasztó felület:



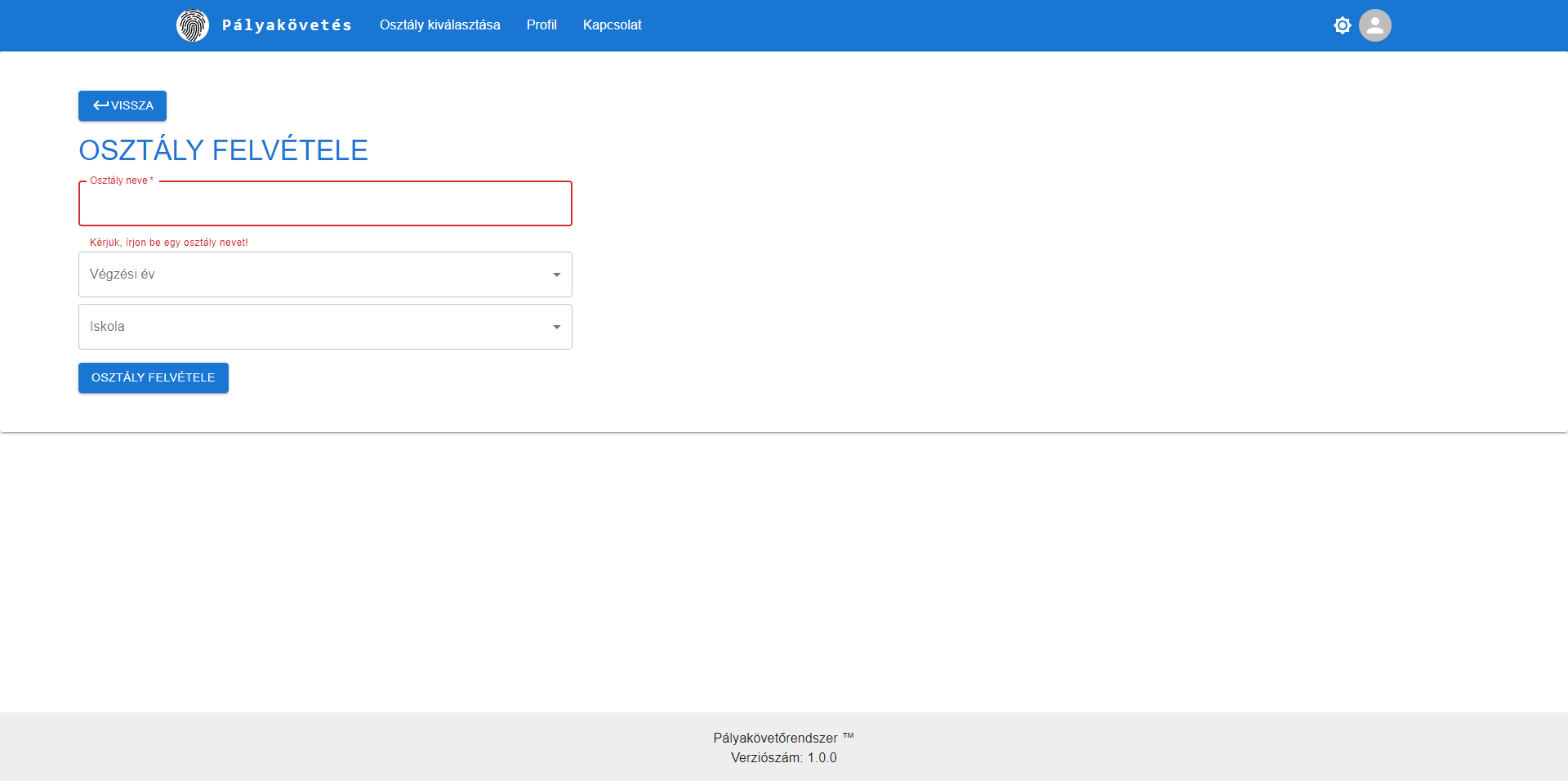
Egy kiválasztott osztály, ami tartalmazza az osztályba tartozó tanulókat:



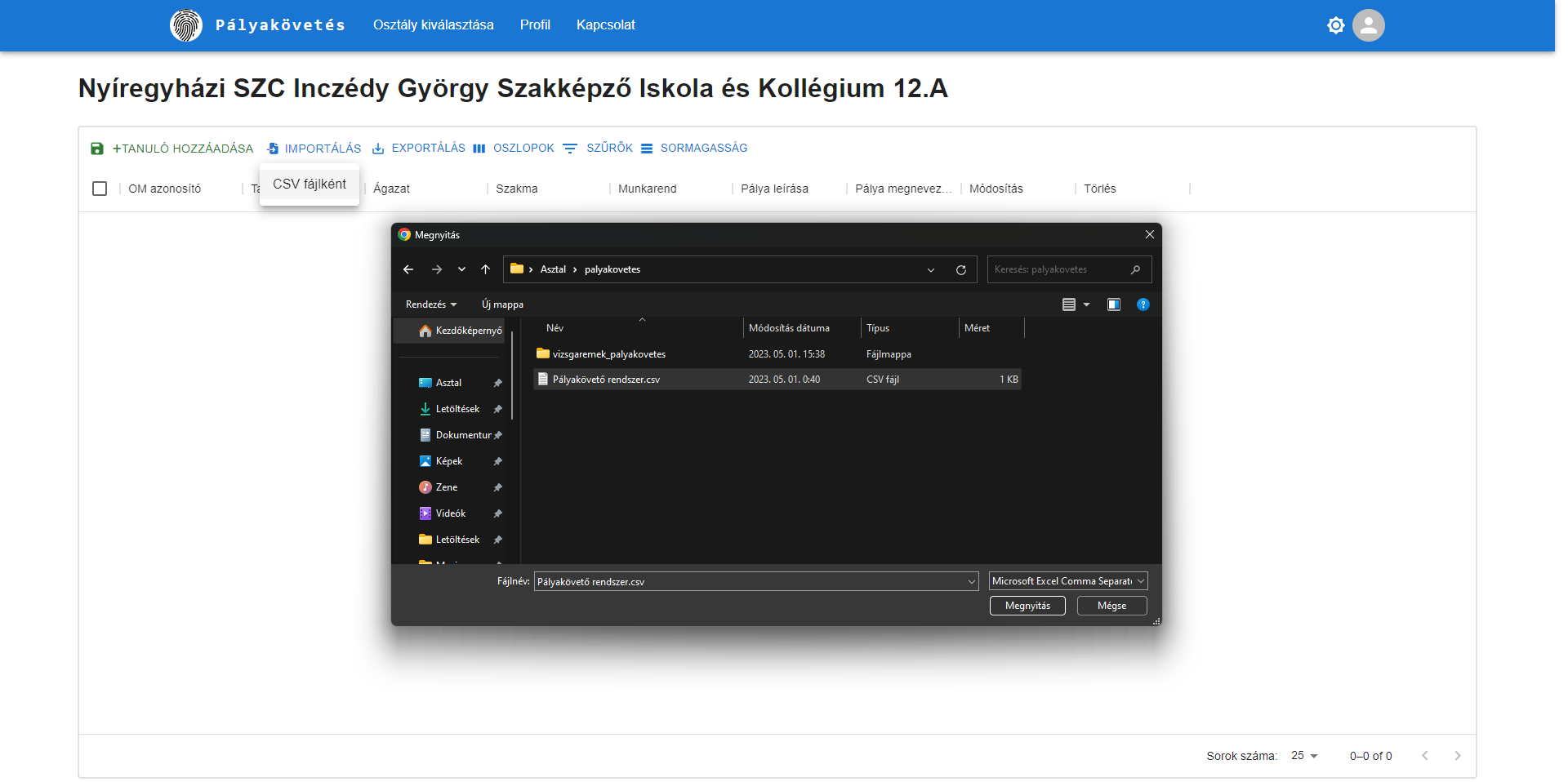
Egy új tanuló felvétele az adott osztályba, manuálisan:



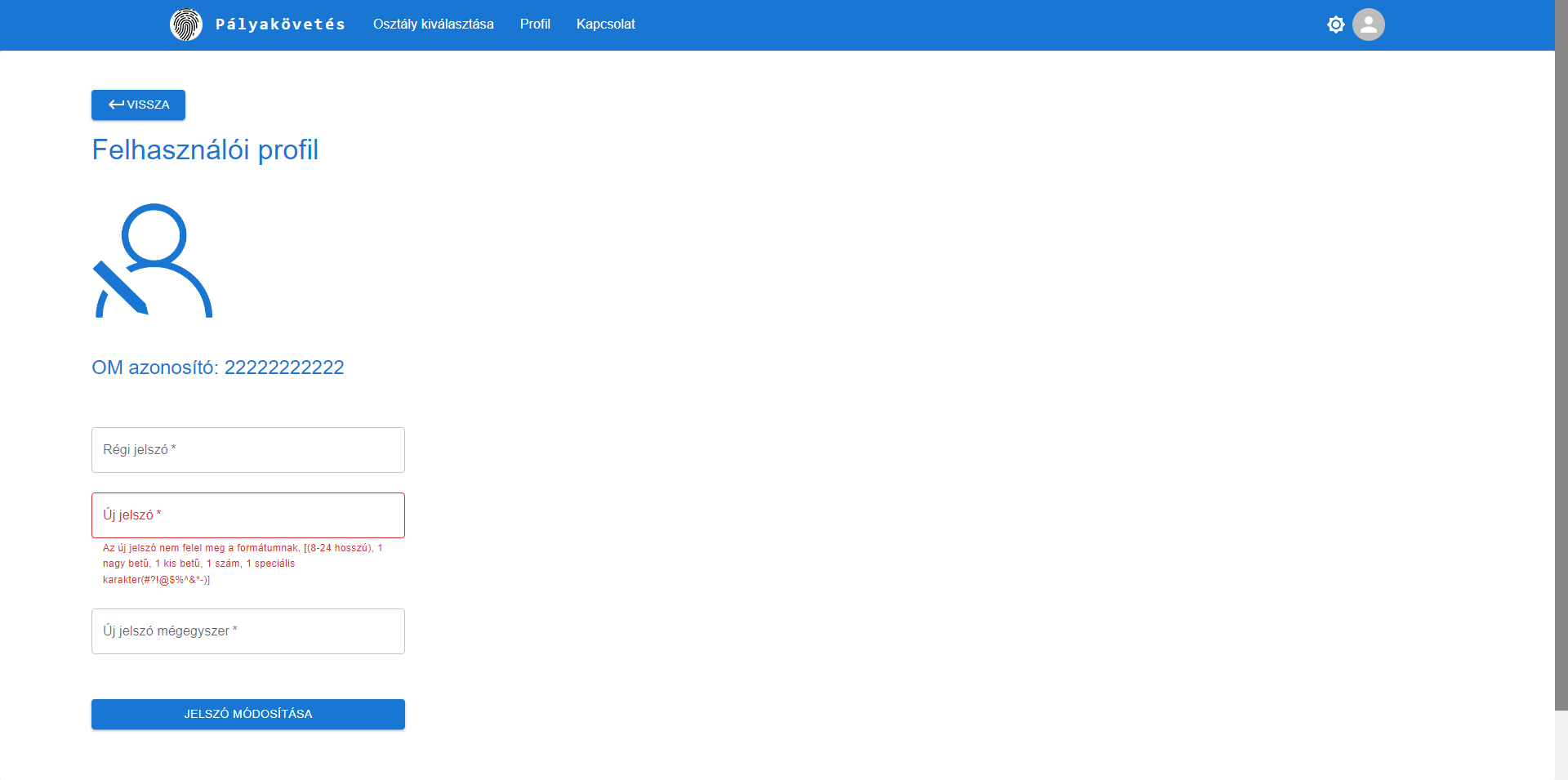
Egy új, üres osztály hozzáadása:



Tanuló(k) hozzáadása .csv fájlból:



Saját profil oldal, ahol a jelszóváltoztatás végezhető el:



Az admin felület tartalmazza a felhasználókat, amikkel törlési műveletek végezhetőek el:

A képen szöveg látható

Automatikusan generált leírás

# **3. Modellek**

Az adatok tárolására több modellre van szükség: iskola, felhasználó, osztály, tanuló, ágazat, szakma, és kategória.

**iskola** {

* id: int (2)
* nev: varchar (255)

}

**felhasznalo** {

* om\_azon: bigint (11)
* jelszo: varchar (60)
* admin: tinyint (1)

}

**osztaly** {

* id: int (4)
* iskolaid: int (2)
* felhasznalo\_om: bigint (11)
* nev: varchar (255)
* vegzesi\_ev: int (4)

}

**tanulo** {

* om\_azon: bigint (11)
* nev: varchar (255)
* osztalyid: int (4)
* nappali\_munkarend: tinyint (1)
* agazatid: int (3)
* szakid: int (3)

}

**palya** {

* id: int (11)
* diak\_om\_azon: bigint (11)
* kategoriaid: int (2)
* leiras: varchar (255)

}

**agazat** {

* id: int (3)
* nev: varchar (255)
* szam: varchar (255)

}

**szakma** {

* id: int (3)
* nev: varchar (255)
* szam: varchar (255)

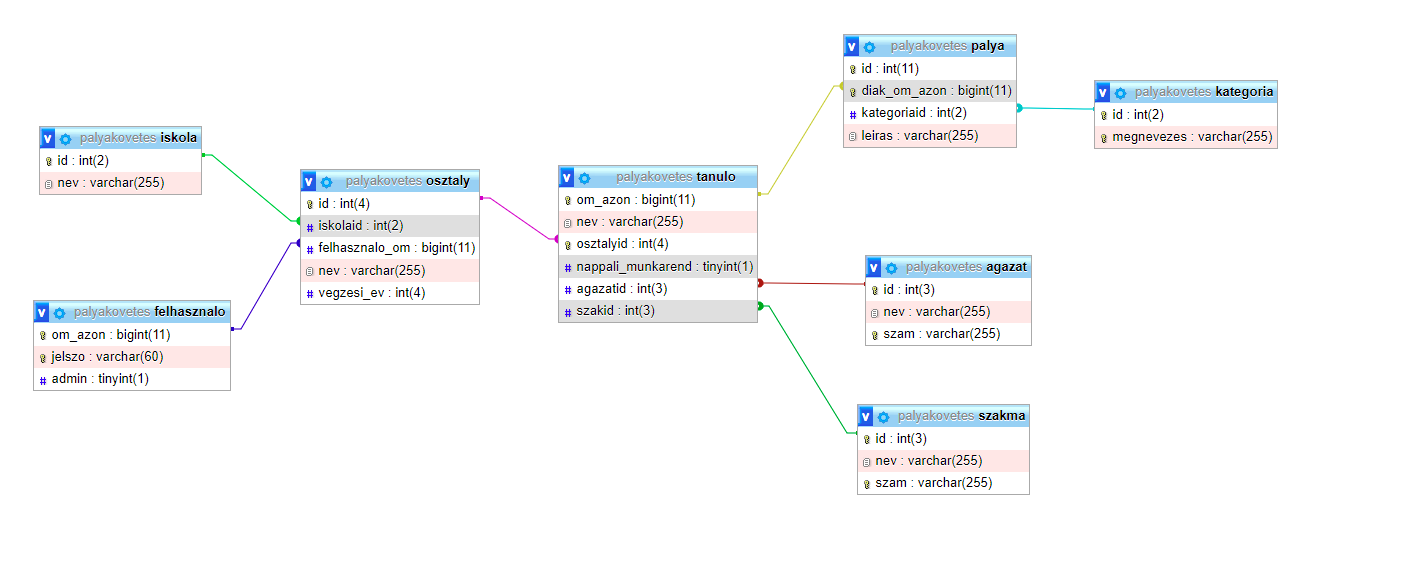
}

**kategoria** {

* id: int (2)
* megnevezes: varchar (255)

}

**Az adatbázis UML diagramja:**

****

# 4. **Alkalmazások kiválasztása**

## 4.1 Front-end

* **React**

A React az egyik nyílt forráskódú JavaScript könyvtár. Interaktív felhasználói felületek felépítésére szolgál. Ez egy hatékony, deklaratív és rugalmas könyvtár. A Model-View- Controller (MVC) V, azaz View elemével foglalkozik. Ez nem egy teljes keret, hanem csak egy előlapi könyvtár. Lehetővé teszi összetett felhasználói felületek létrehozását vagy elkészítését, összetevőkként ismert, elkülönített és apró kóddarabok felhasználásával.

(<https://hu.education-wiki.com/9050114-what-is-react>)

* **Javascript**

A JavaScript (röviden JS) egy objektumalapú scriptnyelv, amelyet elterjedten használnak weboldalakon, a web programozási nyelve. A Javascript futásideje egyszálú, ami azt jelenti, hogy egyszerre csak egy darab kódot képes végrehajtani. Ennek a megoldására használunk async függvényeket.

* **HTML**

A HTML (Hypertext Markup Language) egy leíró nyelv, melyet weboldalak elkészítésére használhatunk.

* **MUI**

A MUI egy nyílt forráskódú keretrendszer (framework) amely egy React alapú komponens könyvtár, ami a Google Material Design elveit implementálja**.** Nagyon könnyen, és minimális energia befektetéssel tudunk jól kinéző, bármilyen képernyőméreten szépen megjelenő weboldalakat készíteni a MUI segítségével.

* **Axios**

Az Axios egy könyvtár, amelyet HTTP -kérések küldésére használnak a böngészőből Node és Express.js platformon keresztül.

* **papaparse**

A Papa Parse .csv fájlok importálást, majd JSON objektummá alakítását teszi lehetővé.

* **file-saver**

A File Saver komponens lehetővé teszi egy megadott adatokkal feltöltött fájl generálását és a kliensre való letöltését.

* **sheetjs**

Exceles táblázatok egyszerű írását,olvasását teszi lehetővé javascriptes környezetben.

## 4.2 Backend

* **Node.js**

A Node.js egy nyíltforrású, többplatformos, szerveroldali JavaScript futtatókörnyezet a **Google Chrome V8 JavaScript***-*motoron alapulva. Leginkább webes applikációk, dinamikus weboldalak fejlesztésére használják, de akár szerveroldali feldolgozó-szkriptek készítésére is kiválóan alkalmas.

* **Express**

Az **Express** (amit gyakran **Express.js**-nek, vagy **ExpressJS**-nek is hívnak) egy szerver oldali (back end) applikáció fejlesztői keretrendszer Node.js-hez. Használata és terjesztése ingyenes, szabad szoftver az MIT licensze alatt. Elsősorban webes applikációkat és API-kat fejleszthetünk vele, és rendkívül sikeres életútja során viszonylag hamar az általános webes applikációs keretrendszerré vált Node.js alatti fejlesztések esetében.

* **Bcrypt –** jelszó hash-elés

A Bcrypt egy népszerű és megbízható módszer a jelszavak titkosítására.

* **Nodemon**

A nodemon egy olyan eszköz, amely segít a **Node.js** alapú alkalmazások fejlesztésében azáltal, hogy automatikusan újraindítja a csomópont-alkalmazást, ha a rendszer fájlváltozásokat észlel a könyvtárban.

* **MySQL**

A MySQL egy rendkívül népszerű **relációs adatbázis-kezelő rendszer***.* A MySQL-t gyakran használják adatbázisként a Node-dal épített projektekhez és alkalmazásokhoz.

* **JSON Web Token - JWT**

A JWT egy olyan szabvány, amely kompakt és önálló módszert határoz meg az információk biztonságos továbbításához az ügyfél és a szerver között, mint JSON objektumot. A kompakt méret megkönnyíti a tokenek átadását URL-en, POST paraméteren vagy egy HTTP fejlécen keresztül.

* **http-status-codes – NPM**

A http kódok jelentésének szöveges megjelenítése átláthatóbbá teszik a kódot.

* **Dotenv**

A dotenv module a környezeti változókat a .env fájlból betölti a process.env változóba, mely a NodeJS process objektumának env property-je, így innen már bármelyik fájl hozzáfér majd a változó értékeihez**.**

(<https://czetojozsef.hu/dotenv-modul-haszn%C3%A1lata-a-env-f%C3%A1jl-adatainak-beh%C3%BAz%C3%A1s%C3%A1ra-nodejs-ben>)

## 4.3 Adatbázis

* **XAMPP**

Egy platformfüggetlen webszerver-szoftvercsomag, azok számára, akik adatbázissal támogatott webszervert szeretnének üzemeltetni, a webes alkalmazások készítését, tesztelését és futtatását oldja meg.

* **MariaDB**

A MariaDB Server az egyik legnépszerűbb nyílt forráskódú relációs adatbázis.

## 4.4 Tesztelés

* **Postman**

A Postman egy komplett eszköztár API fejlesztők számára. Olyan eszköz, amelyet az API-k megbízható teszteléséhez használhatunk. A Postman leegyszerűsíti az API életciklusának minden lépését, és egyszerűsíti az együttműködést.

# 5. Routing

**POST auth/register**

|  |  |
| --- | --- |
| tervezett feladat | Regisztráció |
| bemenet | OM azonosító, jelszó |
| kimenet | Visszajelzés a sikerességről |
|  |  |

**POST auth/login**

|  |  |
| --- | --- |
| tervezett feladat | Bejelentkezés |
| bemenet | OM azonosító, jelszó |
| kimenet | Felhasználónév és egy jogosultságot meghatározó érték (isAdmin), sikerességről való visszajelzés |

**GET auth/logout**

|  |  |
| --- | --- |
| tervezett feladat | Kijelentkezés |
| bemenet | - |
| kimenet | Visszajelzés a sikerességről |

**GET classes/class\_chooser**

|  |  |
| --- | --- |
| tervezett feladat | Osztályok kilistázása, ami az adott felhasználóhoz tartozik |
| bemenet | - |
| kimenet | Sikerességről való visszajelzés, az osztályok listája |

**POST classes/createClass**

|  |  |
| --- | --- |
| tervezett feladat | Üres osztály létrehozása |
| bemenet | Az iskola id-je, a tanár OM azonosítója, az osztály neve, és az osztály végzési éve |
| kimenet | Sikerességről való visszajelzés |

**POST classes/deleteClass**

|  |  |
| --- | --- |
| tervezett feladat | Egy adott osztály és a hozzá tartozó tanulók adatainak törlése |
| bemenet | Az osztály id-je |
| kimenet | Sikerességről való visszajelzés |

**GET categories/getCategories**

|  |  |
| --- | --- |
| tervezett feladat | Kategóriák listázása |
| bemenet | - |
| kimenet | Sikerességről való visszajelzés |

**GET categories/getProfessions**

|  |  |
| --- | --- |
| tervezett feladat | Szakmák listázása |
| bemenet | - |
| kimenet | Sikerességről való visszajelzés |

**GET categories/getSectors**

|  |  |
| --- | --- |
| tervezett feladat | Ágazatok listázása |
| bemenet | - |
| kimenet | Sikerességről való visszajelzés |

**GET schools/getSchools**

|  |  |
| --- | --- |
| tervezett feladat | Az iskolák kilistázása |
| bemenet | - |
| kimenet | Sikerességről való visszajelzés |

**POST students/getStudent**

|  |  |
| --- | --- |
| tervezett feladat | Egy tanuló keresése OM-azonosító alapján |
| bemenet | A tanuló OM-azonosítója |
| kimenet | Sikerességről való visszajelzés, az adott tanuló adatainak kilistázása |

**POST students/studentList**

|  |  |
| --- | --- |
| tervezett feladat | Egy osztály tanulóinak lekérése |
| bemenet | Az adott osztály id-je |
| kimenet | Sikerességről való visszajelzés, az adott osztály tanulói és adatainak listája |

**POST students/getPalya**

|  |  |
| --- | --- |
| tervezett feladat | Egy tanuló pályájának listázása |
| bemenet | A tanuló OM-azonosítója |
| kimenet | Sikerességről való visszajelzés, az adott tanuló pályakövetési adatainak kilistázása |

**POST students/addStudent**

|  |  |
| --- | --- |
| tervezett feladat | Egy tanuló és a hozzá tartozó pálya hozzáadása az adatbázishoz |
| bemenet | A tanuló OM-azonosítója, neve, osztálya, munkarendje, ágazata VAGY szakja, iskola utáni elhelyezkedésének kategóriája, elhelyezkedésének bővebb leírása |
| kimenet | Sikerességről való visszajelzés |

**POST students/editStudent**

|  |  |
| --- | --- |
| tervezett feladat | Egy tanuló és a hozzá tartozó pálya szerkesztése |
| bemenet | A tanuló OM-azonosítója, neve, osztálya, munkarendje, ágazata VAGY szakja, iskola utáni elhelyezkedésének kategóriája, elhelyezkedésének bővebb leírása |
| kimenet | Sikerességről való visszajelzés |

**POST students/deleteStudent**

|  |  |
| --- | --- |
| tervezett feladat | Egy tanuló és a hozzá tartozó pálya törlése az adatbázisból |
| bemenet | A tanuló OM-azonosítója |
| kimenet | Sikerességről való visszajelzés |

**GET users/userList**

|  |  |
| --- | --- |
| tervezett feladat | A felhasználók kilistázása, kivéve a jelenleg bejelentkezett admin. |
| bemenet | - |
| kimenet | Sikerességről való visszajelzés, a felhasználók listája |

**POST users/deleteUser**

|  |  |
| --- | --- |
| tervezett feladat | Egy felhasználó és a hozzá tartozó osztályok törlése az adatbázisból. |
| bemenet | Az törölni kívánandó felhasználó OM-azonosítója |
| kimenet | Sikerességről való visszajelzés |

**POST users/updatePassword**

|  |  |
| --- | --- |
| tervezett feladat | A bejelentkezett felhasználó módosíthatja a saját jelszavát |
| bemenet | A régi (jelenlegi) jelszó, új jelszó |
| kimenet | Sikerességről való visszajelzés |

# 6. Implementáció

Az implementáció a tervezést követő folyamat. A tervezés során megtörténik a feladatokra bontás, valamint az egyes feladatokban érintett funkcionalitás és szerkezet modellezése. Az implementáció során a modellekből készítünk működő, futtatható kódot.

# 7. Tesztelés

A felhasználó szempontjából a rendszer futtatásához elegendő bármely olyan operációs rendszer, amelyen futtatható HTML, CSS és JavaScript technológiával készült alkalmazás megjelenítésére képes böngésző.

A fejlesztés alatt a rendszer tesztelése asztali számítógépeken, illetve laptokon több böngészőben (Google Chrome 112.0.5615.138, Microsoft Edge 112.0.1722.48) ráadásul több asztali operációs rendszereken, név szerint Windows 10 Professional és Windows 11 Professional operációs rendszerek alatt történt. Továbbá mobil eszközön Android és iOS operációs rendszer alatt is teszteltük.

Szerveroldalról nézve a rendszernek szüksége van egy olyan platformra, ami MySQL szervereket képes futtatni, a fejlesztés során egy helyi localhost-al oldottuk meg.

A feladat végrehajtása során folyamatosan jelen volt a tesztelés. Már a statikus oldal megtervezésekor és készülésekor tesztelve volt a megjelenés reszponzivitása, az olvashatóság, a láthatóság.

Az adatbázis tartalommal feltöltése után az oldal tartalmi része is tesztelve, ellenőrizve volt. Ezt Postman-ban valósítottuk meg, amely azonnali visszajelzést adott a fejlesztés sikerességéről, esetleges hibáiról. A felhasználói és adminisztrátori fiókok Postman tesztjeit kimentettük külön gyűjteményekbe, amik a **backend** mappán belül a **postman tesztek** nevű mappában találhatóak meg.

# 8. Bevezetés, éles üzemmód

Az elkészült munka megtekintéséhez szükséges alkalmazások telepítését, és a weboldal üzembe helyezését a felhasználói dokumentáció részletesen lépésről-lépésre leírja.

# 9. Összegzés

Az általunk fejlesztett REST API alapú alkalmazás a CRUD (create, read, update és delete) funkciókat kihasználva mutatja be a kliens-szerver közötti HTTP kommunikációt.

Fontos szempont volt a fejlesztés során a jogkörök kezelése, az admin és regisztrált felhasználók jogosultságainak elkülönítése az alkalmazás különböző részeihez.

Az elkészített alkalmazás nem tökéletes, fejlesztésekre és javításokra szorul a jövőben. Sajnálatos módon néhány funkció a rövid fejlesztési idő miatt nem került bele a szoftverbe. Ilyen hiányosság például az .xlsx fájlból történő importálás (Az alkalmazás jelenleg csak .csv fájlokból képes adatokat importálni), vagy az adatok nyomtatása közvetlenül az oldalról.

Az alkalmazás GitHub repository-ja, history-val együtt:

<https://github.com/imbrainlezz/palyakovetes>