ZÁRÓDOLGOZAT

Készítették:

Sándorné Feke Réka - Zádori Mónika

Konzulens:

Farkas Zoltán

Miskolc

2025.

Miskolci SZC Kandó Kálmán Informatikai Technikum

Miskolci Szakképzési Centrum

SZOFTVERFEJLESZTŐ- ÉS TESZTELŐ SZAK

Porta Rendszer

Sándorné Feke Réka - Zádori Mónika

2024-2025.

Tartalomjegyzék

Bevezetés

A **PortaRendszer** projekt egy innovatív iskolai adminisztrációs platform, amely a hagyományos jelenléti és felügyeleti rendszert ötvözi modern technológiával és pedagógiai szemléletű funkcionalitással. A projekt célja egy olyan webes és asztali alkalmazás létrehozása volt, amely lehetővé teszi az iskolai dolgozók számára, hogy valós időben nyomon kövessék a tanulók intézményen belüli mozgását, kezeljék a délutáni felügyeleteket, valamint adminisztrálják az osztálylétetést, a speciális napokat és a portai eseményeket egy biztonságos, többjogosultságú környezetben.

A **PortaRendszer** szimbolikus jelentéssel is bír: egyszerre utal a fizikai portára, amelyen keresztül a tanulók távoznak, valamint arra a digitális kapura, amelyen keresztül az iskolai működés minden fontos folyamata átláthatóvá válik. A fejlesztés során célunk az volt, hogy a pedagógiai és szervezési munkát hatékonyan támogató, strukturált rendszer jöjjön létre, amely nemcsak az iskolavezetésnek, hanem a tanároknak, portásoknak és adminisztrátoroknak is egyszerű és gyors megoldást nyújt a napi feladatok ellátására.

A projekt fejlesztése során a legújabb technológiákat alkalmaztuk, mint például az ASP.NET Core 8 WebAPI-t, az Entity Framework Core-t JWT-token alapú hitelesítéssel, valamint egy MySQL adatbázist a relációs adatok kezelésére. A rendszer frontendje React.js technológiával készült, mobil/tablet nézetre optimalizálva, Bootstrap és TailwindCSS támogatással. A fejlesztési folyamat során Visual Studio 2022 és Visual Studio Code környezetet, valamint GitHub-ot és Trello-t használtunk a verziókezeléshez és a projektmenedzsmenthez.

A következő fejezetek részletesen tárgyalják a projekt különböző aspektusait, beleértve a koncepciót, a tervezést, a fejlesztést, a tesztelést és a dokumentációt. Célunk, hogy átfogó képet adjunk a **PortaRendszer** kialakításáról és működéséről, valamint bemutassuk a mögötte álló csapat munkáját, technikai megvalósításait és pedagógiai elkötelezettségét.

Projekttervezés

A **PortaRendszer** projekt során a Scrum módszertant alkalmaztuk, amely agilis keretrendszerként heti sprintekben határozta meg a fejlesztési ciklusainkat. Ezek a rövid, jól tervezett szakaszok lehetővé tették számunkra a folyamatos fejlődést, a gyors reagálást a változásokra, valamint az új funkciók gyors tesztelését és bevezetését. A feladatok szervezésére és nyomon követésére **Trello-táblát** használtunk, ahol egyértelműen jelölni tudtuk a fejlesztési fázisokat, elvégzendő feladatokat és a tesztelési mérföldköveket. A csapaton belüli kommunikáció a személyes egyeztetésekre és közvetlen kapcsolatfelvételre épült, ami hatékony, gyors és jól strukturált munkavégzést eredményezett.

A projekt kulcsfontosságú mérföldkövei a következők voltak:

- **Adatbázis kialakítása:** A relációs adatbázis MySQL-ben készült el, az osztályok, tanulók, felhasználók és jogosultságok kezelésére.
- **Bejelentkezés és regisztráció:** Biztonságos JWT-alapú autentikáció került bevezetésre, szerepkör-alapú hozzáféréskezeléssel.
- **Osztálylléptetés:** A tanév eleji automatikus évfolyamsorolás, illetve évismétlő tanulók külön kezelése.
- **Felhasználói felületek szétválasztása:** Admin (igazgató, igazgatóhelyettes), Tanári, Portai felületek jogosultságalapú elérése.
- **Frontend tablet nézetre optimalizálása:** A rendszer mobilbaráttá tétele a tanári és portai felhasználók számára.

Feladatmegosztás és csapatmunka

A feladatok elosztása világosan és hatékonyan történt. A csapat két fő tagból állt, akik a következő munkaterületeken dolgoztak:

- **Sándorné Feke Réka**:
  - Az adatbázis szerkezetének megtervezése és implementálása MySQL-ben
  - A teljes backend fejlesztése ASP.NET Core 8 WebAPI technológiával
  - A jogosultságkezelés, tanulóadminisztráció, osztálylléptetés és fájlimport funkciók megvalósítása
  - A projekt Trello alapú dokumentálása, idővonalak kezelése, fejlesztési ciklusok ütemezése

- **Zádori Mónika**:
  - A frontend alkalmazás megtervezése és implementálása React.js technológiával
  - A felhasználói felületek kialakítása (Admin, Tanári, Portai), figyelembe véve a tablet-nézet és a reszponzív dizájn követelményeit
  - Az egyes funkciók (bejelentkezés, kijelentkezés, tanulólista, státuszváltozás, összevonás, udvarra ment státusz) vizuális megvalósítása
  - Bootstrap- és Tailwind-alapú stílusok integrálása

A fejlesztés során mindketten aktívan részt vettek a teljes fejlesztési ciklusban, a tervezéstől a megvalósításon át a tesztelésig. A sprintek zárásaként rendszeresen tartottunk értékelő megbeszéléseket, ahol visszatekintettünk az előző ciklus eredményeire, megbeszéltük a felmerült nehézségeket, és közösen döntöttünk a következő lépések prioritásairól. Ez a szoros együttműködés, valamint az egymás munkájának tiszteletben tartása és támogatása biztosította, hogy a projekt gördülékenyen és eredményesen haladhasson.

Technológiai eszközök és keretrendszerek

A **PortaRendszer** fejlesztése során kiemelt figyelmet fordítottunk arra, hogy a projekt megfeleljen a legújabb iparági követelményeknek és technológiai trendeknek. Célunk az volt, hogy olyan modern, megbízható és jól skálázható eszközöket használjunk, amelyek elősegítik a hatékony fejlesztést, valamint biztosítják a rendszer hosszú távú fenntarthatóságát.

A backend fejlesztéséhez az **ASP.NET Core 8 WebAPI** keretrendszert választottuk, amely lehetőséget biztosított a REST alapú végpontok kialakítására, valamint a biztonságos autentikációra és adatkezelésre. A korábbi ASP.NET Core verziókhoz képest a 8-as változat már teljeskörűen támogatta az új fejlesztési irányokat, így nem volt szükség kompromisszumokra. A keretrendszer stabilitása lehetővé tette, hogy a fejlesztés minden fázisában a funkciók tökéletesítésére és a felhasználói logikák kialakítására koncentrálhassunk.

A fejlesztés során minden adatot **MySQL** adatbázisban tároltunk, amelyhez az **Entity Framework Core** ORM technológiát használtuk. Az adatelérés LINQ-alapú lekérdezésekkel történt, az adatbázis séma scaffold segítségével generálódott, így az adatmodell és a kód közötti kapcsolat végig biztosított volt. A helyi fejlesztésekhez **XAMPP** környezetet alkalmaztunk.

A frontend fejlesztése **React.js** segítségével valósult meg, amely napjaink egyik legnépszerűbb JavaScript-alapú UI könyvtára. A felület kialakítása során **Bootstrap** és **Tailwind CSS** is segítségünkre volt, amelyek lehetővé tették a reszponzív, könnyen átlátható és tabletre optimalizált kezelőfelület létrehozását. A React komponensek egyértelműen különválasztják az egyes felhasználói szerepkörökhöz tartozó nézeteket, így minden szereplő (adminisztrátor, tanár, portás) a számára releváns funkciókat érheti el.

A jogosultságkezeléshez **JWT (JSON Web Token)** alapú authentikációt alkalmaztunk, amely lehetővé teszi, hogy minden egyes kérés során ellenőrizzük a felhasználó szerepkörét, és kizárólag a számára engedélyezett végpontokat szolgáljuk ki. A tokenek digitálisan aláírtak, így hitelesek és nem manipulálhatók, ezzel biztosítva a rendszer integritását.

A fejlesztés során használt további eszközök:

- **Visual Studio 2022** (backend)
- **Visual Studio Code** (frontend)
- **Git / GitHub** (verziókövetés, külön ágak: frontend, backend, adatbázis)
- **Trello** (idővonal, feladatlista, fejlesztési mérföldkövek, képernyőképek dokumentálása)

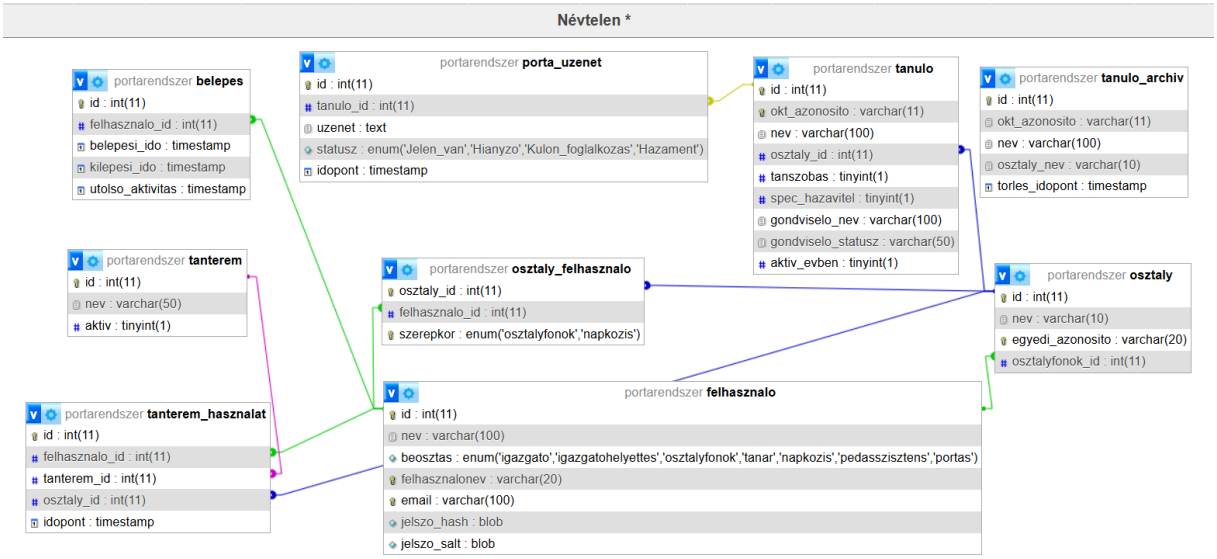
Adatbázis-tervezés

A **PortaRendszer** adatbázisának tervezésekor kiemelt figyelmet fordítottunk az átláthatóságra, a strukturáltságra és a hatékony relációs kapcsolatokra. Bár az adatbázis 12 táblát tartalmaz, az egyes táblák gondosan megtervezettek, és pontosan lefedik az iskolai adminisztrációhoz szükséges funkciókat.

A táblák elhatárolása és a jól meghatározott kapcsolatok lehetővé teszik, hogy az adatok gyorsan és pontosan lekérdezhetők legyenek, miközben az adatredundancia minimális marad. A logikusan felépített idegen kulcsos kapcsolatok biztosítják, hogy az adatok konzisztensen és biztonságosan kapcsolódjanak egymáshoz, például a tanulók, az osztályok, és a portai események relációi.

Adatbázisdiagram és magyarázata

Az alábbi ábra mutatja be a teljes adatbázis-modellt:



1.ábra – PortaRendszer adatbázis-modellje

Főbb táblák és szerepük:

- **tanulo**: tárolja az aktív tanulókat, osztálybesorolással, oktatási azonosítóval, gondviselői státuszokkal.
- **tanulo_archiv**: a törölt tanulók archívuma (pl. kiiratkozás után).
- **osztaly**: az osztályok nevét, egyedi azonosítóját és osztályfőnökét tartalmazza.
- **felhasznalo**: az iskolai dolgozók táblája, szerepkörrel, titkosított jelszóval és elérhetőségekkel.
- **osztaly_felhasznalo**: kapcsolótábla, amely tárolja, hogy egy felhasználó milyen szerepben tartozik egy osztályhoz (pl. osztályfőnök vagy napközis).
- **belepes**: a felhasználók napi be- és kilépési időpontjait rögzíti, ezzel lehetőség nyílik a jelenlét naplózására.
- **porta_uzenet**: a tanulók aktuális státuszát (jelen van, hiányzó, külön foglalkozáson, hazament) és hozzá tartozó üzenetet tartalmazza.
- **tanterem**, **tanterem_hasznalat**: rögzíti, hogy mely tanár/tanársegéd melyik tantermet használta, mikor és milyen osztállyal.

Kapcsolatok

Az adatbázis idegen kulcsos kapcsolatai biztosítják, hogy az adatstruktúra logikus és integrált legyen:

- A tanulo tábla osztaly_id mezője az osztaly táblára hivatkozik
- Az osztaly tábla osztalyfonok_id mezője a felhasznalo táblához kapcsol
- Az osztaly_felhasznalo kapcsolótábla kettős kapcsolatot tart fenn: felhasznalo_id → felhasznalo, osztaly_id → osztaly
- A tanterem_hasznalat hármas kapcsolattal dolgozik: felhasznalo, tanterem, osztaly

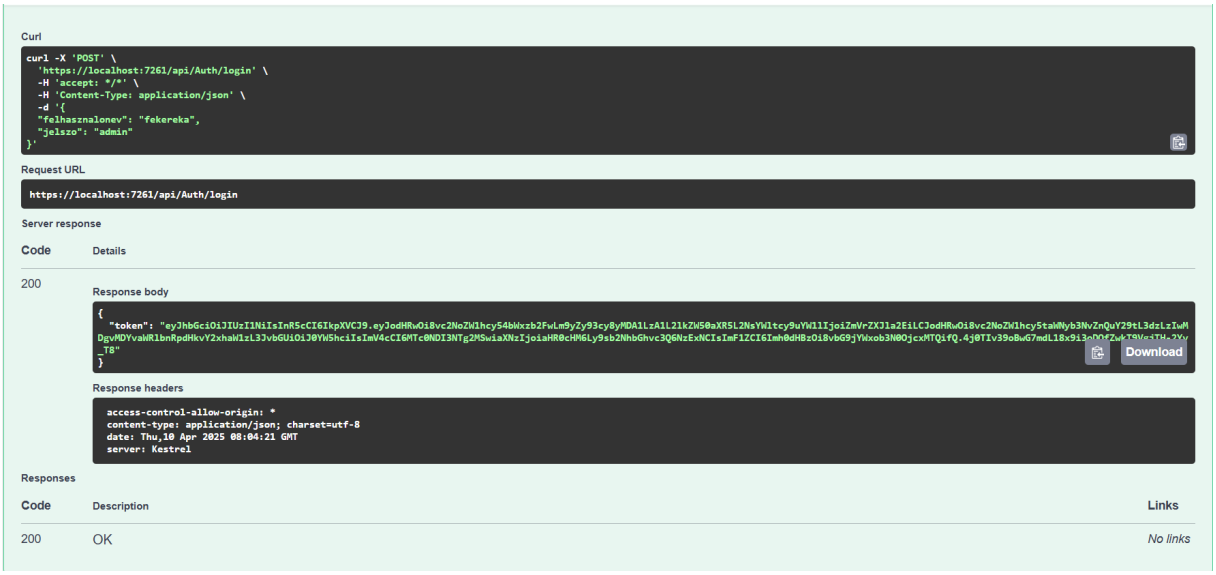
Adatmodellezési szempontok

A rendszerben szereplő adatok valós idejű kezelése miatt a timestamp alapú mezők különösen fontos szerepet kapnak. Ezeket használjuk a tanulók státuszváltozásainál, portai eseményeknél, tanteremhasználatnál és belépéseknél is. Az enum típusú mezők (pl. szerepkor, beosztas, statusz) garantálják, hogy az értékek csak az előre definiált lehetőségek közül választhatók.

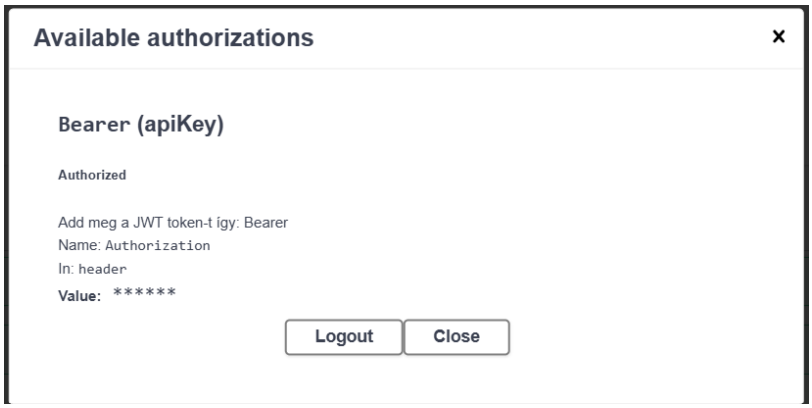
Legfontosabb végpontok és funkcióik

1. 🔐 POST/auth/login – Bejelentkezés

- **Leírás**: Hitelesíti a felhasználót, visszaad egy JWT tokent.
- **Kik használhatják**: Minden regisztrált felhasználó (tanár, portás, igazgató stb.)



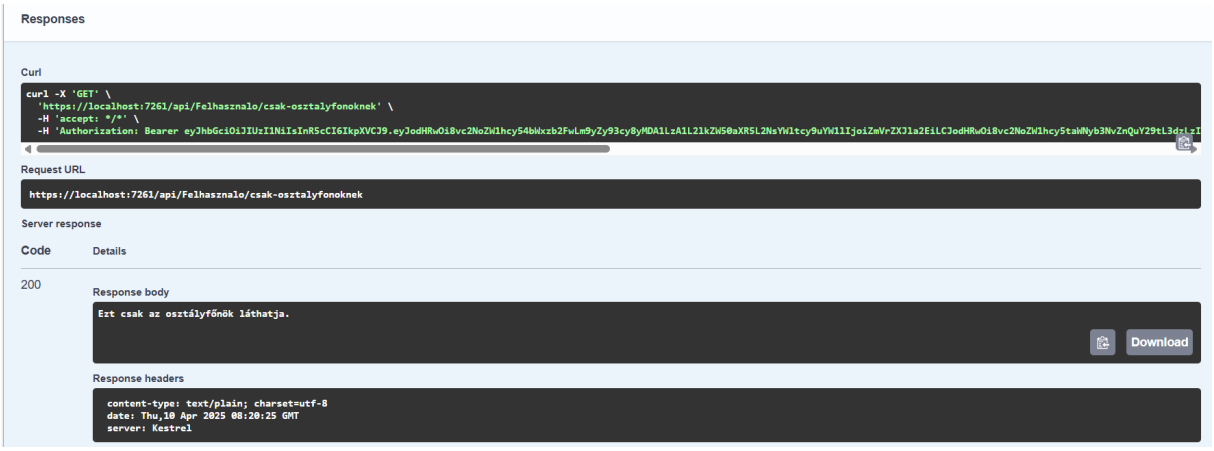
2.ábra – Bejelentkezés és a válaszként kapott token JSON kimenete



3. ábra - Sikeres hitelesítés Swagger felületen JWT tokennel

2. 👤 GET/felhasznalok – Felhasználók listázása

- **Leírás**: Az összes felhasználó megtekintése, jogosultság szerint szűrve.
- **Kik használhatják**: Csak igazgató, igazgatóhelyettes, osztályfőnök



4.ábra – Osztályfőnöki jogosultság

3. 🎓 GET /tanulok – Tanulók lekérdezése

- **Leírás**: Aktív tanulók listázása, osztály és név szerint szűrhető.
- **Kik használhatják**: Osztályfőnökök, napközisek, portások

5.ábra - Szűrés `osztaly_id` alapján + tanulók tömbjének megjelenítése válaszként

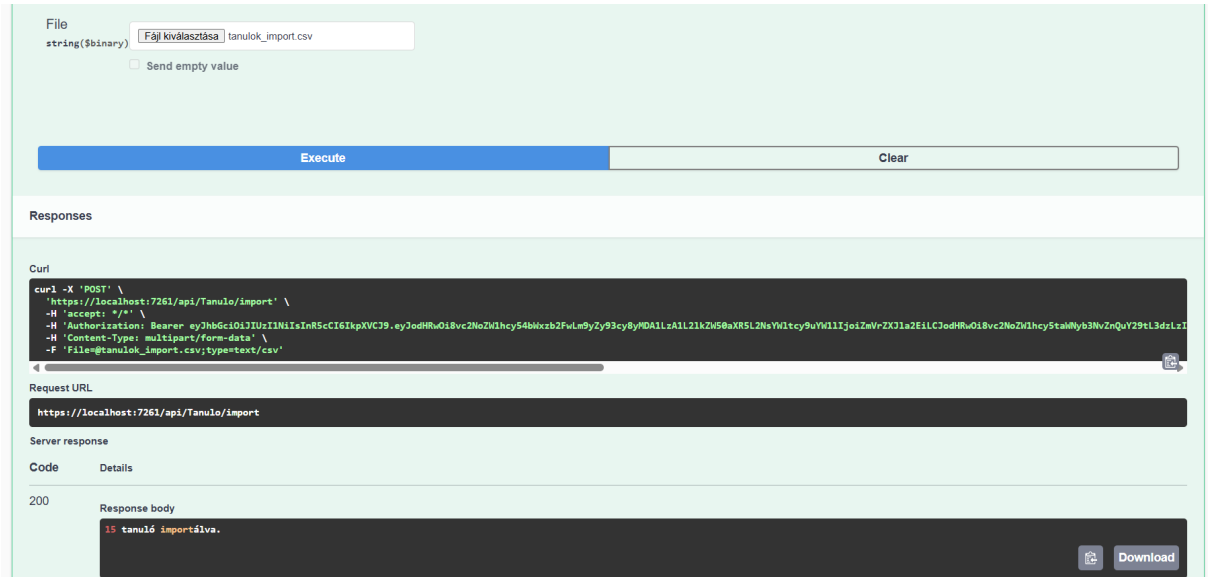
4. 🆕 POST /tanulok – Új tanuló rögzítése

- **Leírás**: Új tanuló hozzáadása oktatási azonosítóval, névvel, osztályhoz rendelve.
- **Kik használhatják**: Csak igazgató, igazgatóhelyettes

6. ábra - Tesztadat beküldése

5. 📁 POST /tanulok/import – Tanulók CSV importálása

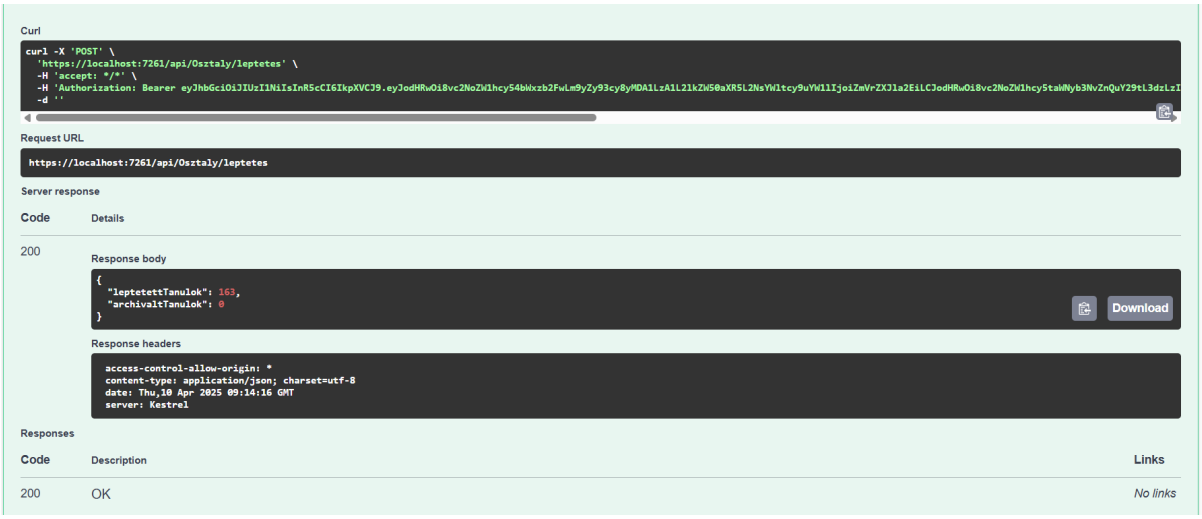
- **Leírás**: Tanulók tömeges feltöltése fájlból (pl. tanévkezdéskor)
- **Kik használhatják**: Igazgató, igazgatóhelyettes, osztályfőnök



7.ábra - Fájlválasztás mező és a válaszüzenet a sikeres importról

6. 🔄 POST /osztalyok/leptetes – Osztályléptetés

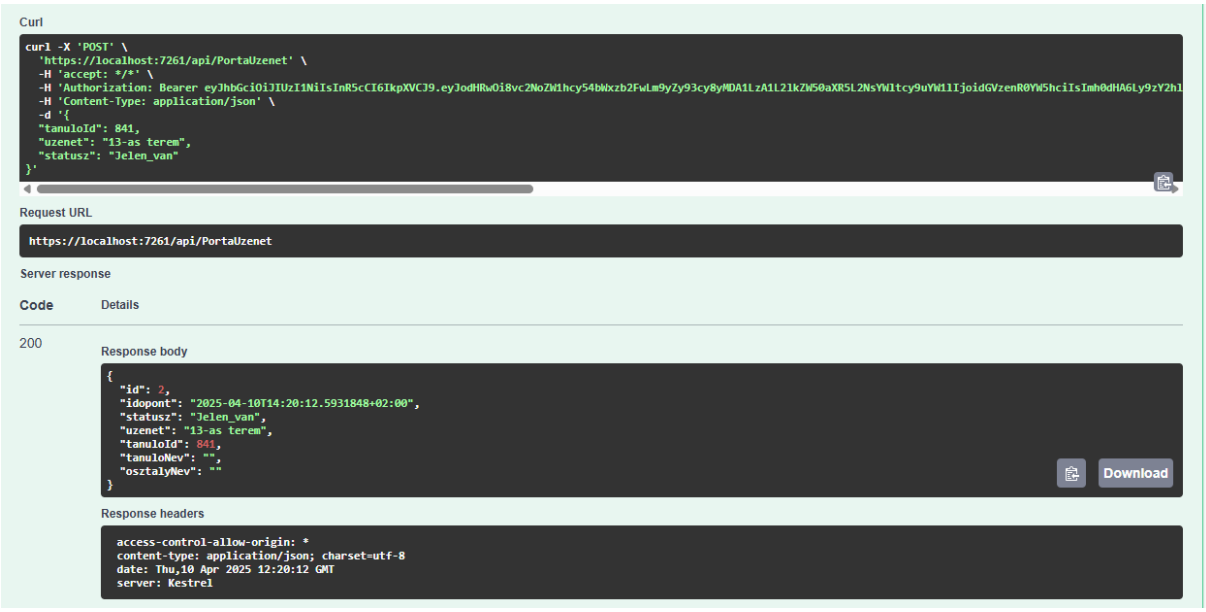
- **Leírás**: Tanulók évfolyamszintű automatikus továbbmozgatása.
- **Kik használhatják**: Csak igazgató, igazgatóhelyettes



8.ábra - Léptetett tanulók listája

7. 🏠 POST /porta-uzenet – Portai üzenet/státusz rögzítése

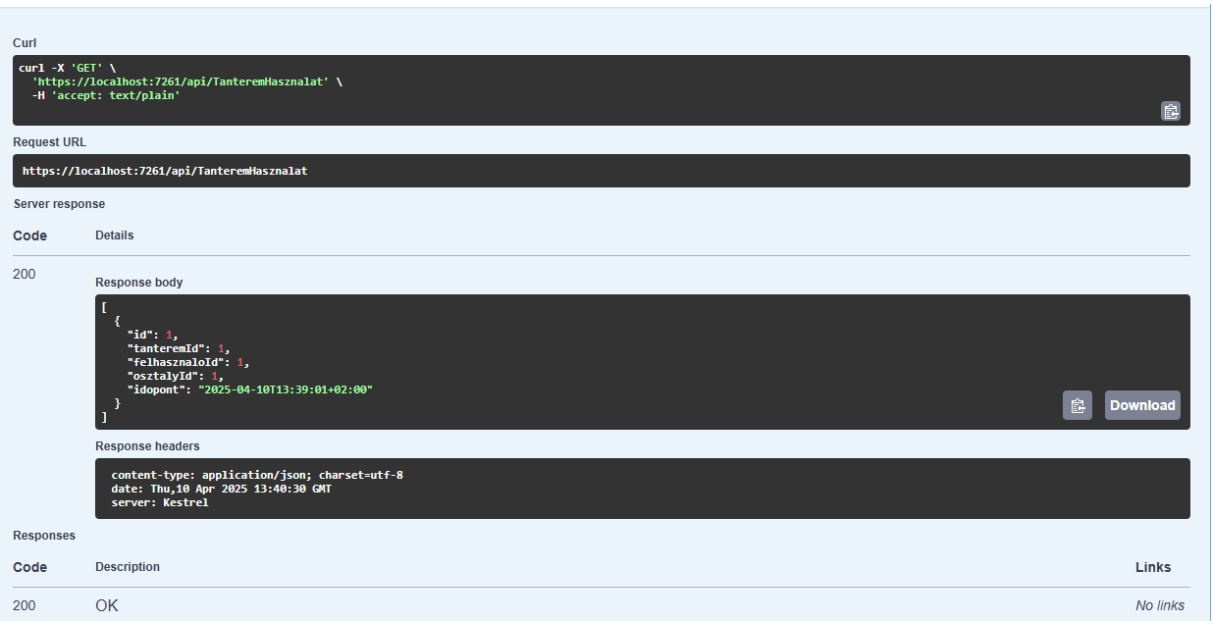
- **Leírás**: Jelzi, ha egy tanuló hazament, hiányzik vagy más programon van.
- **Kik használhatják**: Portás, tanár, napközis



9.ábra – „Jelen_van” státusz + üzenet megjelenítése

8. 🏫 GET /tanterem-hasznalat – Teremhasználati adatok

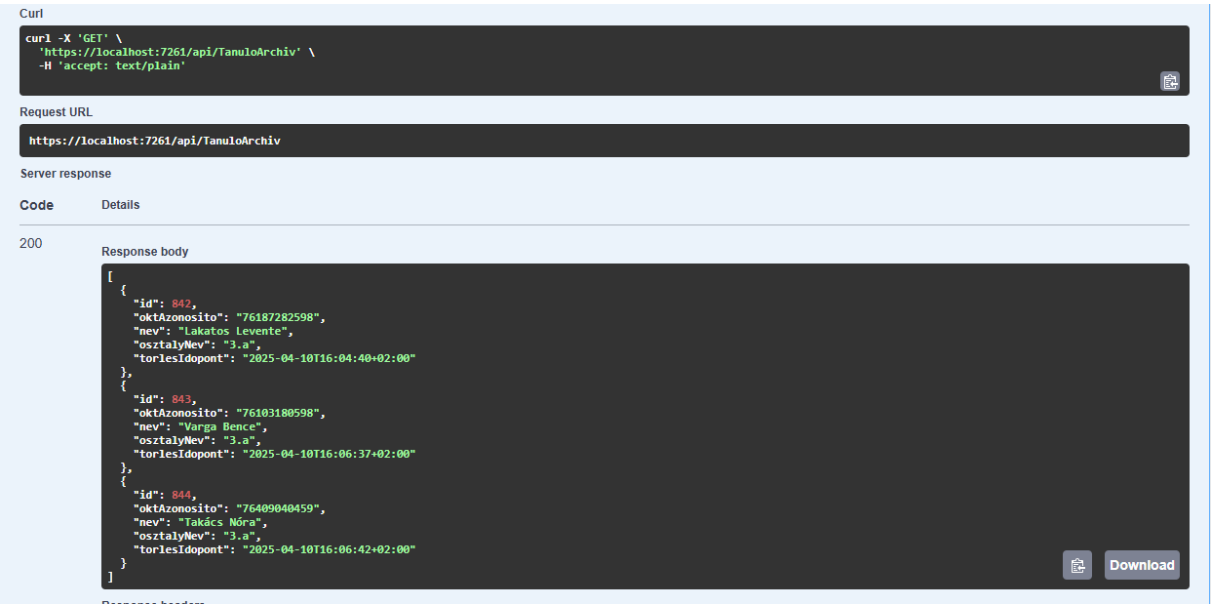
- **Leírás**: Tantermek használatának nyomon követése (ki használta, mikor, milyen osztállyal)
- **Kik használhatják**: Igazgató, helyettes, tanár



10.ábra - Válasz JSON tartalmazza a felhasznalo, tanterem, osztaly, idopont adatokat

9. 🗃️ GET /tanulok/archiv – Archivált tanulók lekérdezése

- **Leírás**: Korábban törölt vagy végzett tanulók listázása
- **Kik használhatják**: Igazgató, igazgatóhelyettes



11.ábra - Archivált tanulók listája, törlés dátuma is megjelenik

Felhasználói élmény (UX)

A **PortaRendszer** tervezése során kiemelt figyelmet fordítottunk arra, hogy a különböző felhasználói szerepkörök (portás, tanár, napközis, osztályfőnök, igazgató, adminisztrátor) gyorsan és hatékonyan tudják elérni a számukra fontos funkciókat.

Eszközhasználat és platformfüggetlenség

- A rendszer **tablet-kompatibilis**, mobilnézetben is tökéletesen működik.
- **Reszponzív felhasználói felület** React technológiával, Bootstrap és Tailwind segítségével.

Szerepkör alapú hozzáférések

- A belépést követően minden felhasználó kizárólag a saját szerepkörének megfelelő funkciókat látja.
- A portások egy kattintással rögzíthetik, ha egy diák elhagyta az iskolát.
- Az osztályfőnökök látják saját osztályuk tanulóinak státuszát, és beléphetnek a tanteremhasználati naplóba is.
- Az adminisztrátorok teljeskörű hozzáféréssel rendelkeznek.

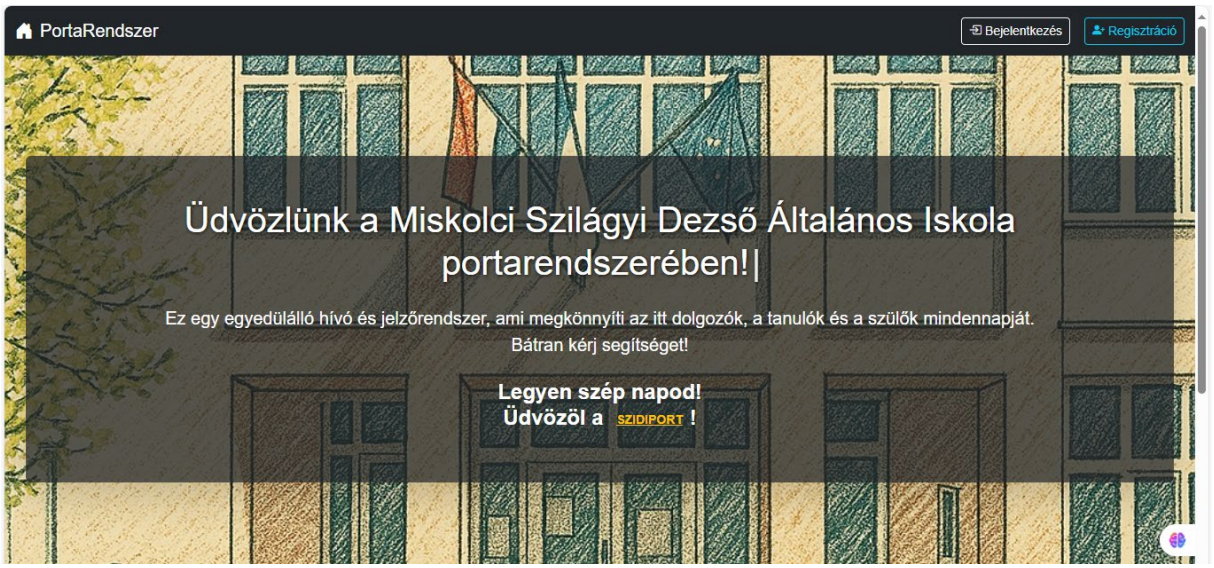
Egyszerű adatbevitel és gyors visszajelzés

- A státuszok előre definiált listából választhatók (hazament, külön foglalkozás stb.), így gyorsan rögzíthetők.
- A műveletek után a rendszer visszajelzést ad (sikeres mentés, hiba esetén hibaüzenet).
- Swagger UI-t biztosítunk a REST API tesztelésére fejlesztői oldalról.

Felhasználói visszajelzések alapján fejlesztett funkciók

- Osztályfőnök automatikus hozzárendelése új osztály létrehozásakor.
- Tanulók archiválása év végén vagy kiiratkozás esetén.
- Tanteremhasználat nyilvántartása egyszerű űrlappal.

Kezdő oldal

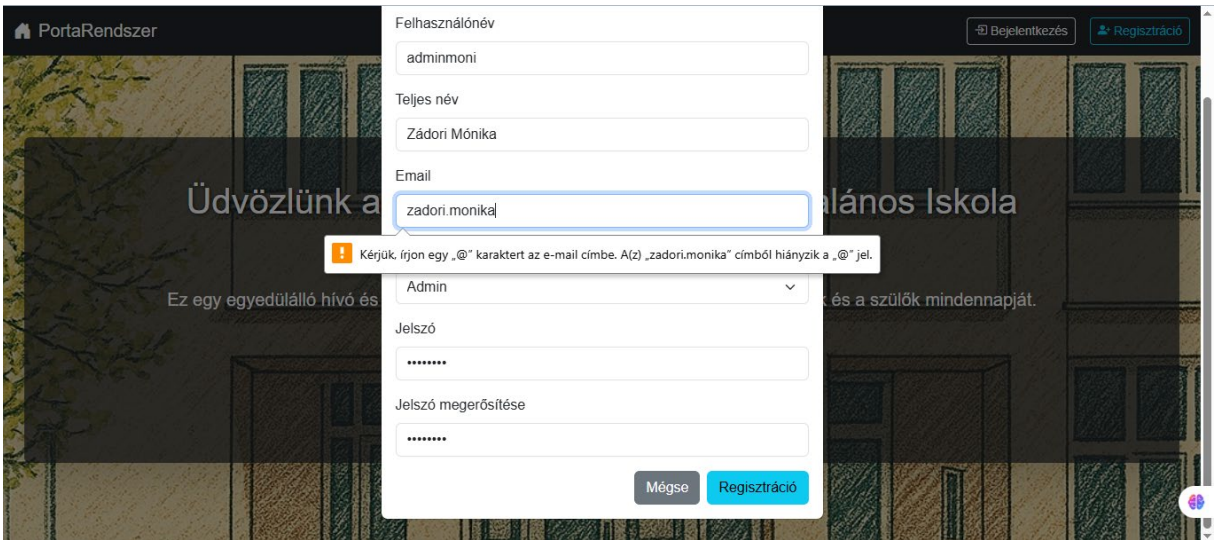


A **PortaRendszer** nyitóoldala egyedi, illusztrált háttérrel és meleg hangvételű fogadtatással fogadja a látogatókat:

- Az üdvözlő szöveg személyes hangulatot teremt, az iskolai közösséghez való tartozás érzését erősíti.
- Rövid bemutatkozás tájékoztatási célból jelenik meg.
- A nyitólapon egyértelműen elhelyezett Bejelentkezés és Regisztráció gombok találhatók, amelyek elérhetőek mobil/tablet nézetben is.
- A „Legyen szép napod!” üzenet pozitív és bizalomgerjesztő hangot üt meg.

Ez a kezdőfelület nemcsak technikai, hanem érzelmi szempontból is fogadja a felhasználót, különösen fontos ez egy oktatási intézményben.

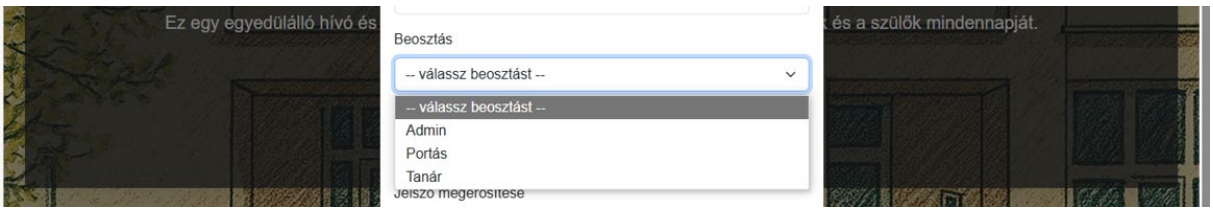
Regisztrációs felület



A rendszer első használatakor a felhasználóknak lehetőségük van saját fiók létrehozására a **Regisztráció gomb** segítségével. A gomb megnyomását követően egy modális ablak jelenik meg, amely letisztult, középre igazított elrendezéssel vezeti végig a felhasználót a szükséges adatok megadásán.

Kötelező mezők:

- **Felhasználónév** – Egyedi azonosító, amelyet a bejelentkezéshez használ a rendszer.
- **Teljes név** – A felhasználó teljes neve, amely megjelenik különböző felületeken, például osztályfőnök esetén az osztály mellett.
- **Email** – Kapcsolattartási cím, jövőbeli értesítésekhez, hitelesítéshez előkészítve.
- **Beosztás** – Legördülő menüből választható, jelenleg a következő opciókkal:



- **Jelszó és jelszó megerősítése** – A jelszó biztonságos, kétszeri megadásával erősíti meg a felhasználó az adatok helyességét.

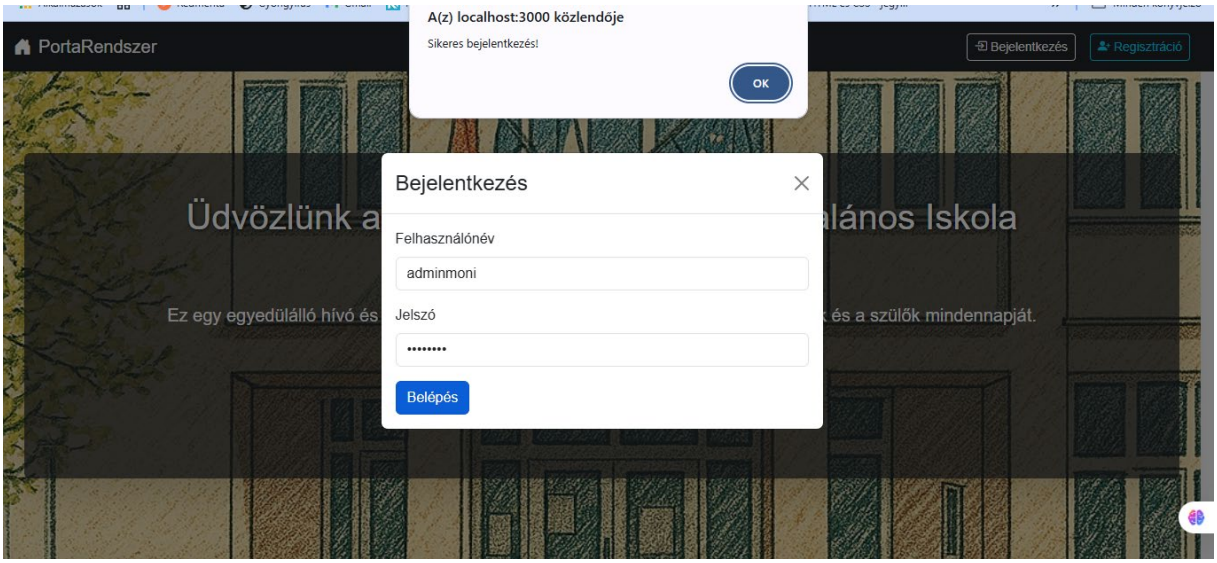
UI elemek és működés:

- A modális ablak bármikor bezárható a jobb felső sarokban található × gombbal.
- A „Regisztráció” gomb lenyomására a rendszer backendjéhez kerülnek beküldésre az adatok.
- Sikertelen beküldés esetén (pl.: ha hiányzik egy mező vagy a jelszók nem egyeznek), a felület nem frissül, és a háttérrendszer hibaüzenete jelenik meg.
- Sikeres regisztráció után a felhasználó automatikusan átirányításra kerülhet a szerepköre szerint meghatározott oldalra (/redirect).

Felhasználóbarát megközelítés:

A regisztrációs felület jól átlátható, mobilbarát kialakítású, Bootstrap keretrendszert használ. A mezők intuitívan kitölthetők, a legördülő menü csak érvényes szerepköröket enged kiválasztani, így kizárja a nem várt vagy nem támogatott szerepköröket.

Bejelentkező felület



A regisztrációt követően vagy korábban létrehozott felhasználói fiókkal a rendszer használata a Bejelentkezés funkción keresztül történik. A kezdőoldal jobb felső sarkában található **Bejelentkezés gomb** megnyomásával egy modális (popup) ablak nyílik meg, amely a minimál design elveit követve csak a legszükségesebb adatokat kéri be.

Beviteli mezők:

- **Felhasználónév** – A korábban regisztrált azonosító, amellyel a rendszer a jogosultságokat is kezeli.
- **Jelszó** – A titkosított módon tárolt jelszó, csillaggal takart formában jelenik meg.

UI elemek és működés:

- A × gombbal a felhasználó bármikor bezárhatja az ablakot, és visszatérhet a kezdőoldalra.
- A Belépés gomb lenyomása után a rendszer az API backenddel kommunikál, ellenőrizve a felhasználónév és jelszó helyességét.
- Sikertelen belépés esetén hibaüzenet jelenik meg (pl. „Ismeretlen hiba történt”, vagy „Hibás felhasználónév vagy jelszó”).
- Sikeres bejelentkezés után a felhasználó szerepköre szerint automatikusan átirányításra kerül a megfelelő felületre:
    - admin → /admin
    - portas → /portas
    - tanar → /tanar

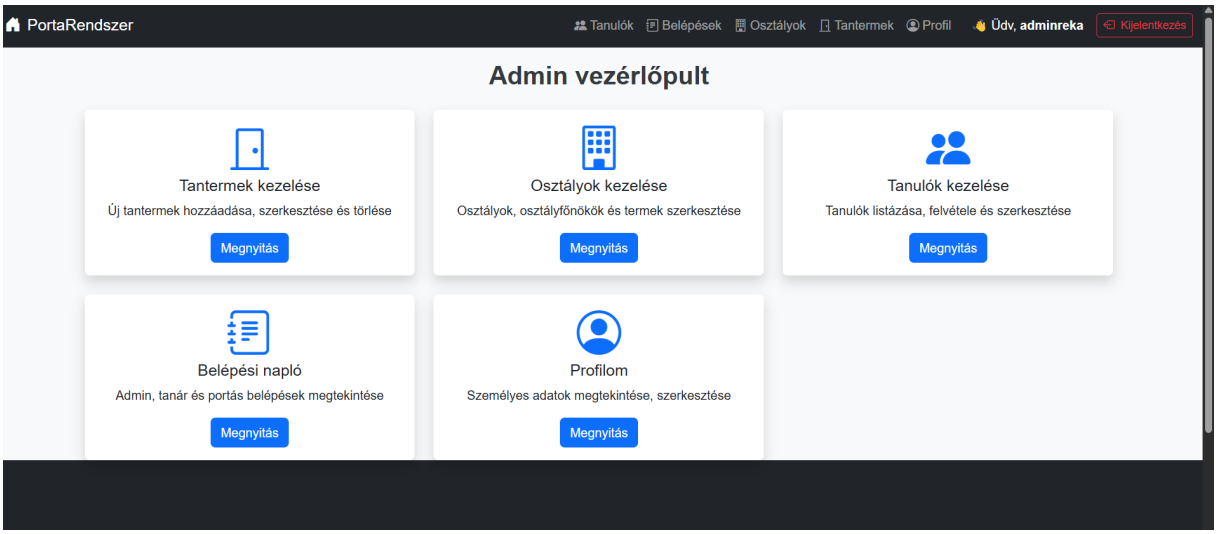
Biztonsági megfontolások:

- A jelszó minden esetben rejtett formában jelenik meg (password input).
- A rendszer csak akkor enged belépést, ha mindkét mező helyesen van kitöltve.
- A sikeres autentikáció során JWT token kerül mentésre a localStorage-be, amely a későbbi jogosultság-ellenőrzésekhez szükséges.

Felhasználóbarát élmény:

A bejelentkező felület egyszerű és gyors használatot biztosít, tableten és érintőképernyős eszközökön is jól működik. A letisztult elrendezés és az azonnali visszajelzés a hibákról biztosítja, hogy a felhasználó pontosan tudja, mi történik bejelentkezéskor.

Admin felület



Sikeres bejelentkezés után, ha a felhasználó szerepköre admin, automatikusan a **/admin** útvonalra irányítódik, ahol megjelenik az **Admin vezérlőpult**.
Ez a központi kezelőfelület minden adminisztrátori jogosultságot biztosító funkciót átlátható, **kártyás elrendezésben** kínál.

Elérhető funkciók:

Funkció	Leírás
Tantermek kezelése	Új tantermek felvétele, meglévők módosítása és törlése. A tantermek nevét, típusát és elérhetőségét lehet beállítani.
Osztályok kezelése	Osztályok létrehozása, szerkesztése. Osztályfőnök és tanterem hozzárendelés lehetséges.
Tanulók kezelése	Teljes tanulói adatbázis kezelés: új tanulók felvitele, lista megtekintése, adatlap szerkesztése. Importálási lehetőség CSV fájlból.
Belépési napló	Naplózott belépések megtekintése admin, tanár és portás szerepkörök szerint, dátum és felhasználó alapján.
Profilom	Az admin saját adatai: név, email, jelszó módosítása, szerepkör megtekintése.

UI jellemzők:

- A felület **reszponzív**, tehát tableten is jól használható.

- Az egyes kártyákon jól látható ikon segíti az azonosítást.
- Minden kártya alján egy kék Megnyitás gomb található, amely az adott modul részletes oldalára navigál.

Navigáció:

A felső navigációs sáv minden oldalon elérhető, amely gyors elérést biztosít a legfontosabb modulokhoz:

- **Tanulók**
- **Belépések**
- **Osztályok**
- **Tanterem**
- **Profil**
- valamint a **Kijelentkezés gomb** is itt található.

Felhasználóbarát működés:

- Minden admin funkció világosan elhatárolt.
- Az adminisztrátor pontosan látja, hol milyen műveleteket végezhet.
- A rendszer a hibákat és sikereket **toast üzenetekkel** jelzi (pl. „Sikeres törlés”, „Hiányzó mező”).

Tanterem lista

Az adminisztrátorok számára elérhető a **Tanterem lista** oldal, amely a Tanterem nevű entitás adatainak kezelésére szolgál.

A felület célja, hogy egyszerűen és gyorsan lehessen tantermeket:

- Hozzáadni
- Szerkeszteni
- Törölni

Felület felépítése

- **Cím:** „Tanterem lista”
- **Új tanterem hozzáadása** gomb: a lap tetején található, kék színű, + ikonnal
- **Tanterem kártyák**: minden tanterem külön dobozban jelenik meg

Minden doboz tartalmazza:

- A tanterem **nevét vagy számát**
- **Szerkesztés gombot** (sárga ceruza ikon)
- **Törlés gombot** (szürke kuka ikon, piros kerettel)

Tanterem típusok

A rendszerben egyaránt szerepelnek:

- **Számozott tantermek** (pl. 1., 2., 3. … 15.)
- **Elnevezett tantermek**: informatika, könyvtár, technika, tornaterem, foglalkoztató

Ez lehetővé teszi, hogy az iskola különböző típusú helyiségei egységesen kezelhetők legyenek.

Működés és logika

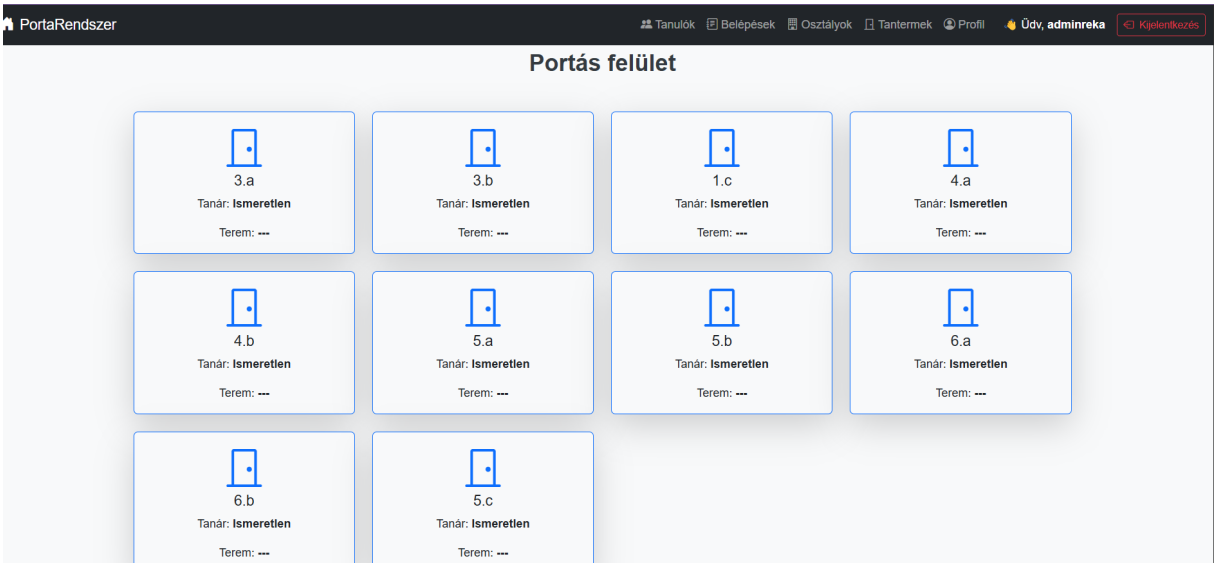
- **Új tanterem létrehozása** során a felhasználó egy űrlapot tölt ki, ahol a tanterem nevét adja meg.
- **Szerkesztés** során módosítható a tanterem megnevezése.
- **Törlés** gombbal az adott tanterem véglegesen eltávolítható az adatbázisból (ha nincs hozzárendelve osztályhoz, egyéb entitáshoz).

A műveletek során a rendszer automatikusan frissíti a listát az adatbázisból, így a felhasználónak nem kell újratöltenie az oldalt.

Jogosultság

Ez az oldal kizárólag admin szerepkörrel rendelkező felhasználók számára érhető el. Más szerepkör (pl. portás, tanár) nem fér hozzá.

Portás felület



A portás szerepkörrel bejelentkező felhasználó automatikusan a **/portas** útvonalra kerül átirányításra. Itt egy kifejezetten egyszerű, gyors áttekintést nyújtó felület fogadja őt, amely az **összes osztályról** vizuális listát mutat.

Felület célja

A portás felület fő célja, hogy **gyorsan információhoz jusson** az osztályokról:

- Milyen osztályok vannak az iskolában?
- Ki az osztályfőnök?
- Melyik teremben van az osztály?

Megjelenés

Az osztályokat egy kártyás elrendezésben jeleníti meg:

- **Osztály neve** (pl. „3.a”)
- **Tanár** – az osztályfőnök neve (ha nincs hozzárendelve: *Ismeretlen*)
- **Terem** – az osztály tanterme (ha nincs megadva: ---)

A kártyák ikonos megjelenítéssel vizuálisan is elhatárolódnak egymástól.

Funkcionalitás

Ez a felület **csak megtekintésre** szolgál – portás szerepkörű felhasználó:

- Nem tud módosítani adatokat
- Nem tud új bejegyzést rögzíteni
- Nem láthat érzékeny adatokat a tanulókról

Ez a korlátozás biztonsági szempontból indokolt, és illeszkedik a portások jogosultsági szintjéhez.

Navigáció és kényelem

- A **felső menüsáv** elérhető marad itt is, így gyorsan átnavigálhat más nézetekbe (pl. *Belépések* vagy *Profil*).
- A Kijelentkezés gomb mindig kéznél van.
- A felület mobil/tablet nézetre is optimalizált.

Tesztelés

A **PortaRendszer** fejlesztése során kiemelt figyelmet fordítottunk az alkalmazás megbízhatóságára, biztonságára és felhasználóbarát működésére. A tesztelési folyamatokat már a fejlesztés korai szakaszától kezdve integráltuk, hogy a hibák idejében észlelhetőek és javíthatóak legyenek.

Backend tesztelése

A backend rendszer fejlesztése során számos **manuális** és **automatikus tesztelési lépést** hajtottunk végre. Az alábbi főbb funkciók esetében különösen nagy hangsúlyt fektettünk a helyes működés ellenőrzésére:

- **Felhasználók regisztrációja és bejelentkezése**: minden szerepkör (admin, tanár, portás) jogosultságkezelésének ellenőrzése
- **Osztályok, tantermek, tanulók CRUD műveletei**: a rekordok hozzáadása, szerkesztése, törlése és listázása
- **Belépések naplózása és státuszüzenetek kezelése**: a státuszváltozások megfelelő naplózása és visszakereshetősége

- **Tanulók átléptetése és archiválása**: kiemelten figyeltük, hogy a nyolcadikos tanulók megfelelően archiválásra kerüljenek, míg a többiek automatikusan a következő évfolyamba lépjenek

Ezek a tesztek segítettek az **adatintegritás megőrzésében** és az **üzemszerű működés** fenntartásában.

Swagger API tesztek

A fejlesztés során a Swagger felületet is aktívan használtuk a végpontok kipróbálására és tesztelésére. Ezzel:

- gyorsan tesztelhettük az API-k válaszait,
- ellenőrizhettük a jogosultsági szinteket,
- valamint egyszerűen reprodukálhattuk a hibákat.

Frontend tesztelés

A React-alapú frontend esetében külön figyelmet fordítottunk arra, hogy:

- a jogosultsági szintek megfelelő oldalakra irányítsák a felhasználókat (admin, tanár, portás felület elválasztása),
- a felhasználói interakciók (űrlapok, gombok, modális ablakok) hiba nélkül működjenek,
- mobil/tablet nézetben is használható legyen a rendszer (reszponzív kialakítás).

Irodalomjegyzék

1. **ASP.NET Core web API,** Webcím: https://learn.microsoft.com/en-us/aspnet/core/web-api/?view=aspnetcore-8.0, Letöltés dátuma: 2025.03.03.
2. **Entity Framework Core,** Webcím: https://learn.microsoft.com/en-us/ef/core/, Letöltés dátuma: 2025.03.03.
3. **JWT-token authentikáció,** Webcím: https://jwt.io/introduction, Letöltés dátuma: 2025.03.22.
4. **MySQL hivatalos dokumentáció,** Webcím: https://dev.mysql.com/doc/, Letöltés dátuma: 2025.03.05.
5. **React – Hivatalos dokumentáció,** Webcím: https://react.dev/, Letöltés dátuma: 2025.03.14.
6. **Bootstrap – Hivatalos dokumentáció,** Webcím: https://getbootstrap.com/, Letöltés dátuma: 2025.03.14.
7. **Professional C# and .NET: 2021 Edition**, Szerző: Christian Nagel, Kiadás éve: 2021, Kiadó: Wrox, ISBN: 978-1119797203.