

Szakképesítés neve: Szoftverfejlesztő

OKJ száma: 54 481 02 0010 54 04

Záródolgozat

Filc weboldal

Ferenczi Tímea
témavezető

Szeredi Ákos
13/A

Békéscsaba, 2024.

Tartalomjegyzék

Bevezetés	3
Választott téma indoklása	4
Felhasználói dokumentáció.....	5
Rendszerkövetelmények	5
Hardverkövetelmények	5
Szoftverkövetelmények.....	5
A program és ahhoz szükséges környezet letöltése és telepítése.....	6
A program használatának részletes leírása	9
Regisztráció, bejelentkezés	9
Navigációs sáv, bal oldali menü	9
Főoldal	10
Értékelések oldal	10
Órarend oldal	10
Mulasztások oldal	11
Iskola oldal.....	11
Fejlesztői dokumentáció	14
Alkalmazott fejlesztői eszközök	14
Adatmodell leírása	16
Algoritmusok	20
ProtectedRoute	20
Bejelentkező algoritmus.....	21
Egyedi token (kulcs) generálása	22
Tesztelés.....	23
Fejlesztési lehetőségek.....	25
Összefoglalás	26
Irodalomjegyzék	27

Bevezetés

A záróvizsgám beadandójához témának egy iskolai weboldalt választottam. A weboldal célja, hogy egyszerűbbé és hatékonyabbá tegye az iskolai élet adminisztratív feladatait. Az alkalmazásban lehetőség nyílik órarend létrehozására és kezelésére, iskolai adminisztrációs feladatok elvégzésére, értékelések és érdemjegyek tárolására, felvételére. A weboldal lehetőséget nyújt arra, hogy több iskola is egyszerre használni tudja teljesítmény kiesése nélkül. A tanároknak kevesebb időt kell adminisztrációval tölteniük, így több időt tudnak a tanításra fordítani. A diákok és a szülők számára is könnyebbé válik az információkhoz való hozzáférés.

Választott téma indoklása

A File témaválasztása mögött, személyes tapasztalataim alapján éreztem szükségét egy olyan alkalmazásnak, ami képes egy iskolai élet szerteágazó feladatait kezelni. Úgy érzem, hogy nincs ehhez hasonló versenyképes alkalmazás a piacon a Krétán kívül, így úgy döntöttem elkészítem a saját ilyen feladatokat kezelni képes alkalmazásomat.

Régóta fejleszték és csinálók már weboldalakat, programokat és alkalmazásokat, viszont eddig ehhez a projekthez még nem volt hasonló nagyságrendű. A maga nehézsége, bonyolultsága és kihívása miatt választottam ezt a projektet témaként.

Felhasználói dokumentáció

Rendszerkövetelmények

Hardverkövetelmények

Minimális rendszerkövetelmény:

- **CPU:** 1.5 GHz processzor vagy jobb
- **RAM:** 2 GB
- **Szükséges terület:** 3 GB
- **Operációs rendszer:** Windows 10

Optimális rendszerkövetelmény:

- **CPU:** 1.7 GHz processzor vagy jobb
- **RAM:** 4 GB
- **Szükséges terület:** 4 GB
- **Operációs rendszer:** Windows 10

Szoftverkövetelmények

Az oldal használatához kivétel nélkül bármely böngésző alkalmas (Edge, Google Chrome, Brave, Opera, stb.). Az oldal magától alkalmazkodik a képernyő méretéhez, emiatt nem szükséges képernyőbeállításokat módosítani, mert a weblap reszponzív.

Szükséges programok: NodeJS (lehetőleg legújabb verzió, v20), Xampp.

A program és ahhoz szükséges környezet letöltése és telepítése

A weboldal eléréséhez böngészőre van szükség, ha ezzel nem rendelkezünk telepítenünk kell egy számunkra szimpatikusot. Mivel az egész fejlesztés és tesztelés Chrome-ban történt, ezért én a Google Chrome böngészőt ajánlom a stabil és gyors működés eléréséhez. A böngésző telepítője mellékelve megtalálható a CD-n (ChromeSetup.exe), egyszerűen és könnyen feltelepíthető bármely felhasználó számára (1. ábra).

A telepítő megnyitása után nincs más tennivaló csak megvárni a telepítési folyamatot, a folyamat teljesen automatikusan lezajlik.



1. ábra

A weboldal adatbázisához szükségünk lesz egy MySQL-t futtató környezetre, ebben az esetben phpMyAdmin-ra és egy Apache szerverre, ami a Xampp szoftver segítségével futtatható. Ha nincs Xampp szoftverünk, akkor ebben az esetben le kell telepíteni. A program telepítője szintén megtalálható a mellékelt CD-n. (xampp-windows-x64.....installer.exe)

A telepítő futtatása után ki kell választanunk a könyvtárat, ahova szeretnénk telepíteni. Ez a könyvtár alapból és általában a fő merevlemezen van (C://xampp), más tennivalónk nincs, csak várni, amíg a telepítés befejeződik.

Ezt követően az Xampp megnyitása után, el kell indítani a MySQL és Apache szerverünket, ez a két szerver felel a weboldal adatbázisáért. Első indításkor automatikusan létrehozza a program az adatbázist, nekünk csak adatokkal kell majd felölteni.

Ahhoz, hogy maga az oldalunk futni tudjon a NodeJS szoftverre lesz szükség, a program legújabb verziójának telepítője a CD-n megtalálható (node-v20...msi). A telepítő megnyitása után el kell fogadni az általános szerződési feltételeket, majd pedig ki kell választani a könyvtárat, ahova a programot szeretnénk telepíteni. Ezután nincs más dolgunk, mint megvárni a telepítést.

A telepítés után minden szoftver rendelkezésre áll, hogy elindítsuk a programunkat. A program elindítása a következőképpen történik.

 backend	2024. 04. 04. 16:33	Fájlmappa
 frontend	2024. 04. 04. 16:37	Fájlmappa

2. ábra

A program főkönyvtárában található két mappa, az egyik mappa 'backend' a háttérfolyamatokért szolgál, lépünk be a mappába és nyissuk meg a '.env' fájlt.



```
.env - Notepad
File Edit Format View Help
DATABASE_URL="mysql://root:@localhost:3306/filc"
JWT_SECRET="szupertitkos"
```

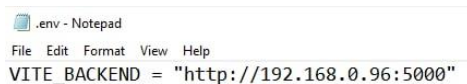
3. ábra

Ebben a fájlban találunk két környezeti változót. Az első változó az adatbázis csatlakozásához szükséges, ide kell megadni az adatbázis belépéséhez szükséges adatokat. (felhasználónév, jelszó, elérési cím, port), ha ezen nem lett változtatva akkor így hagyhatjuk a változókat. A 'JWT_SECRET' változó a 'jsonwebtoken' modulunk jelszavaként felel.

Ezek után nyissuk meg a 'backend' mappájában a terminált, ezt úgy tehetjük meg, hogy a fájl elérési sávba beírjuk, hogy 'cmd' aztán nyomunk egy entert. A terminálba a következőket kell beírni:

- npm i – ez a parancs felel azért, hogy a modulok letöltve naprakészek legyenek. Mivel a modulok már le vannak töltve, ezért ezt a parancsot nem kell feltétlen használni.
- npx prisma db push – ez a parancs az adatbázis naprakész frissítéséért felel, ezzel a paranccsal tudunk ráfrissíteni az MySQL adatbázisunkra bármilyen szerkezeti változtatás esetén.
- npm run dev – ezzel a paranccsal indítható el a 'backend' oldal, elindítás után más tennivalónk nincs.

A 'backend' elindítása után, lépünk vissza a program főkönyvtárába és nyissuk meg a 'frontend' mappát, ebben a mappában található a weboldal kinézeti része. Szintén nyissuk meg a '.env' fájlt, csak úgy mint a 'backend'-nél ahogy tettük.



```
.env - Notepad
File Edit Format View Help
VITE_BACKEND = "http://192.168.0.96:5000"
```

4. ábra

Ez a változó azért felel, hogy a 'frontend' oldal meg tudja találni a 'backend' oldalt. A 'backend' szervere alából 5000-es porton fut, így a port elé csak egy ip címet kell megadni, ahonnan elérhető lesz a szerver.

Maradjunk a 'frontend' mappában és nyissuk meg a terminált, azon a módon ahogy azt a 'backend' oldalon csináltuk.

- `npm i` – ez a parancs felel azért, hogy a modulok letöltve naprakészek legyenek. Mivel a modulok már le vannak töltve, ezért ezt a parancsot nem kell feltétlen használni.
- `npm run dev` – ezzel a parancssal indítható el a 'frontend' oldal, elindítás után nincs más tennivaló.

Ha mindkét programrész fut, akkor a weboldalunkat a 'http://localhost:5173'-as címen érhetjük el.

A program használatának részletes leírása

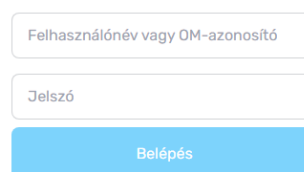
Az oldal bejelentkezés nélkül nem elérhető el, nincsen bejelentkezés nélküli tartalom az oldalon. A funkciókat csak bejelentkezett felhasználók érhetik el.

Regisztráció, bejelentkezés

Nincs regisztrációra lehetőség az oldalon, mivel ez egy iskola által vezetett és menedzselt alkalmazás, ezért csak bizonyos felhasználóknak van lehetőségük új felhasználót regisztrálni vagy felvenni az oldalon. Így az alkalmazás elindításakor a kezdő oldalon csak bejelentkezés funkció található.


Az oldalra bejelentkezni diákként OM azonosítóval és jelszóval, vagy tanárként felhasználónévvel és jelszóval lehet (5. ábra). A felhasználók belépéséhez a felhasználói név meghatározásakor az ékezet nélküli kisbetűs szöveg ajánlott.

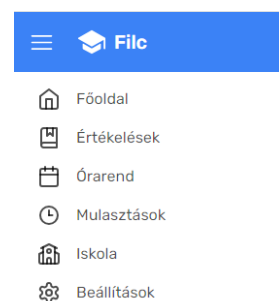
Filc



5. ábra

Navigációs sáv, bal oldali menü

Bejelentkezés után egyből szembejön velünk a navigációs menü a képernyő bal oldalán (6. ábra). Ebben a menüben léphetünk tovább az oldal többi funkciójára a hozzáférésünktől függően. Van külön menüpont csak diákoknak, csak tanároknak, valamint csak adminisztrátoroknak. A navigációs sáv lekicsinyíthető ikonokra, illetve nagyítható  gombbal. A belépett felhasználó neve és jogosultsági szintje (tanár, diák, admin) a navigációs sáv jobb oldalán látható, ahol a profil menü lenyitásával tud kijelentkezni a felhasználó.



6. ábra

Főoldal

A bejelentkezés után a főoldalra irányít át minket az oldal. Ezen az oldalon találhatjuk az iskola elérhetőségi adatait (címe, weboldala, telefonszáma, email címe, igazgatója).

Bármilyen adatbevitelt követően a felhasználónak frissítenie kell az oldalt. Az adatok feltöltésekor a tanárok feltöltésével kell kezdeni, mivel így tudjuk majd az osztályok meghatározásakor az osztályfőnököt is kiválasztani. Ezt követően történhet a diákok adatainak rögzítése.

Adminisztrátori jogosultságot csak MySQL adatbáziskezelőn belül lehetséges beállítani, erre az oldalon egyelőre nincs lehetőség.

Értékelések oldal

Diákként ezen az oldalon látjuk az aktuális teljesítményünket, kapott értékeléseinket havi bontásban, de lehetőségünk van megnézni a havi vagy éppen egész tanéves átlagunk is.


Tanárként ez az oldal bőven több funkciót és lehetőséget hordoz magával, mivel itt tudjuk a diákok munkáját kiértékelni és dokumentálni. Tanárként az értékelés oldalon az oldal tetején megjelennek az általa tanított osztályok és tantárgyak összekapcsolva. Az értékelni kívánt osztály és tantárgy kártyájára kattintva megjelennek a tanított tanulók nevei. 'Új értékelés' gombra kattintva a beviteli felülethez jutunk, ahol megadhatjuk az értékelés dátumát, súlyát, majd ezt követően a tanulók nevével egysorban elhelyezett 5 pontos értékelési táblában kiválaszthatjuk az érdemjegyet. Lehet szövegesen is értékelni, amennyiben a diák munkája netán értékelhetetlen vagy nem adott be munkát.

Ezen az oldalon a tanár csak a saját diákjait látja, akit éppen tanít. Csak az osztályfőnökök látják a saját osztályuk teljes teljesítményét és értékeléseit.

Órarend oldal

Ezen az oldalon látják a diákok az adott aktuális órarendjüket napokra és órákra bontva, hogy fel tudjon készülni a következő napra. Itt tudják megnézni az aznapi/a már megtartott órák témáját és munkáját.

Tanárként láthatjuk a saját órarendünket, csak úgy, mint a diákok, valamint az osztályfőnökök az osztályuk órarendjét is meg tudják tekinteni. A tanárok letudják könyvelni ezen a felületen az óráikat, valamint utána szerkeszteni is tudják azokat. Lekönyvelésnél hiányzást is tudunk rögzíteni, amit a diák azonnal lát és értesül róla. A lekönyvelést követően az órarendben az óra színe átvált zöld háttérre. A lekönyveletlen órák piros háttérrel jelennek

meg az órarendben. Kötelességük azon a héten az összes órát lekönyvelni, különben lekönyveletlen, hiányos marad, mivel heti zárásra van beállítva az alkalmazás. A heti zárás előtt a lekönyvelt órák még változtathatók mind az óra témájában, mind a hiányzók tekintetében. A hiányzás a tanulók nevével egyvonalban elhelyezett váltógombon rögzíthető. Adminisztrátorként látjuk az összes osztály óráját. Ha az adminisztrátori jogosultságunk mellett még tanárként és osztályfőnökként is rendelkezünk jogosultsággal, akkor az Órarend oldal tetején először az osztályunk órarendjét, majd a saját órarendünket látjuk és ezt követi az összes többi osztály órarendje. Új órákat is tudunk felvenni a  gombra kattintást követően (. ábra) vagy éppen törölni, ami oldal frissítése után módosul.


Óra hozzáadása

Óra kezdete 00:00

Óra vége 00:00

Hozzáadás

7. ábra

Az órarendben a tanítási naphoz történő tantárgy és tanár rögzítését is a  gombra kattintással kezdeményezhetjük, ahol a megjelenő ablakban legördülő listából választhatjuk ki a tantárgyat a hozzá kapcsolódó tanárral együtt.

Mulasztások oldal

Diákként ezen az oldalon látjuk a saját hiányzásainkat napokra és tanórákra rendezve, a hiányzás státuszával együtt.

Osztályfőnökként itt tudjunk nyomon követni a saját osztályunk diákjainak mulasztásait, szintén napokra és tanórákra rendezve. A tanuló hiányzását a tanóra mellett jobb oldalon megjelenő legördülő menüből tudjuk változtatni az alábbiakra: igazolatlan, orvosi igazolt, szülői igazolt, iskolai igazolt státuszra.

Iskola oldal

Ezen az oldalon az iskola neve alatt vízszintesen elhelyezkedő színes gombok egy-egy felületre irányítanak: diákok, tanárok, felhasználók, osztályok, tantárgyak. A gombokon a

megnevezés alatt látható az adott csoport rekordjainak száma. A gombok alatt alaphelyzetben a diákokat mutatja a rendszer.

Tantárgyak

A tantárgyak felvétele a név megadásával történik. A tantárgyak listázásakor a műveleti oszlopban ikonok segítségével választhatjuk ki a tantárgy módosítását vagy törlését. A tantárgy módosítása úgy történik, hogy a tantárgy nevére kattintunk, majd az új tantárgy név begépelését követően, a tantárgy mezőjéből kikattintunk és ezután a mentés gombra nyomunk. Egy oldalon 5 tantárgy jeleníthető meg, lapozó nyilak segítségével tudunk lépegetni a tantárgyak között, illetve keresőmező segítségével konkrétan tantárgyra tudunk keresni. A kereső sávval egyvonalban található az 'Exportálás' gomb, melynek segítségével tudjuk a megjelenített tantárgyakat kinyomtatni vagy lementeni CSV formátumba. A megjelenített tantárgy neve alapján rendezhető növekvő, illetve csökkenő sorrendbe.

Osztályok

Az osztályok felvétele az osztály azonosítójának és osztályfőnök megadásával történik. Az osztály azonosítója bármilyen karaktert és számot tartalmazhat, illetve lehet szöveges adat is. Az osztály rögzítését követően egyelőre nincs lehetőség az osztály azonosítójának és osztályfőnökének módosítására és törlésére. Egy oldalon 5 osztály jeleníthető meg, lapozó nyilak segítségével tudunk lépegetni az osztályok között, illetve keresőmező segítségével konkrétan osztályra tudunk keresni. A kereső sávval egyvonalban található az 'Exportálás' gomb, melynek segítségével tudjuk a megjelenített osztályokat kinyomtatni vagy lementeni CSV formátumba. A megjelenített adatok (osztály azonosítója, osztályfőnöke) alapján rendezhető növekvő, illetve csökkenő sorrendbe.

Tanár

A tanárok felvétele a tanár nevének, születési dátumának, telefonszámának, felhasználónevének, jelszavának megadásával történik. Megadható az is, hogy milyen tantárgyat tanít és mely osztályban osztályfőnök. Egy tanár több tantárgyat is taníthat. Egy tanár egy osztálynál lehet osztályfőnök. A tanár rögzítését követően egyelőre nincs lehetőség az adatainak módosítására és törlésére. Egy oldalon 5 tanár jeleníthető meg, lapozó nyilak segítségével tudunk lépegetni az tanárok között, illetve keresőmező segítségével konkrétan tanárra tudunk keresni. A kereső sávval egyvonalban található itt is az 'Exportálás' gomb, melynek segítségével tudjuk a megjelenített tanárokat kinyomtatni vagy lementeni CSV

formátumba. A megjelenített adatok (tanár neve, születési dátuma, telefonszáma, felhasználóneve) alapján rendezhető növekvő, illetve csökkenő sorrendbe. Tanárok regisztrálásakor a regisztrációval egyidőben megadható a tanított tantárgy is. A tanárok megtekintésekor gördítősávval tudunk rátekinteni a tanárhoz kapcsolt valamennyi tulajdonsághoz.

Diák

A diákok felvétele a diák nevének, születési dátumának, telefonszámának, OM azonosítójának, osztályának megadásával történik. A jelszót automatikusan generálja a program a tanuló születési dátumából (pl.: 2005-02-06). A diák rögzítését követően egyelőre nincs lehetőség az adatainak módosítására és törlésére. Egy oldalon 5 diák jeleníthető meg, lapozó nyilak segítségével tudunk lépegetni a diákok között, illetve keresőmező segítségével konkrétan diákra tudunk keresni. A kereső sávval egyvonalban található az 'Exportálás' gomb, melynek segítségével tudjuk a megjelenített diákokat kinyomtatni vagy lementeni CSV formátumba. A megjelenített adatok (diák neve, OM azonosítója, osztálya, születési dátuma) alapján rendezhetők növekvő, illetve csökkenő sorrendbe ezen az oldalon is az adatsorok.

Felhasználó

Egy oldalon 5 felhasználó jeleníthető meg, lapozó nyilak segítségével tudunk lépegetni a felhasználók között, illetve keresőmező segítségével konkrétan felhasználóra tudunk keresni. A kereső sávval egyvonalban található az 'Exportálás' gomb, melynek segítségével tudjuk a megjelenített felhasználókat kinyomtatni vagy lementeni CSV formátumba. A megjelenített adatok (felhasználó neve, jogosultsága) alapján rendezhető növekvő, illetve csökkenő sorrendbe.

Fejlesztői dokumentáció

Alkalmazott fejlesztői eszközök

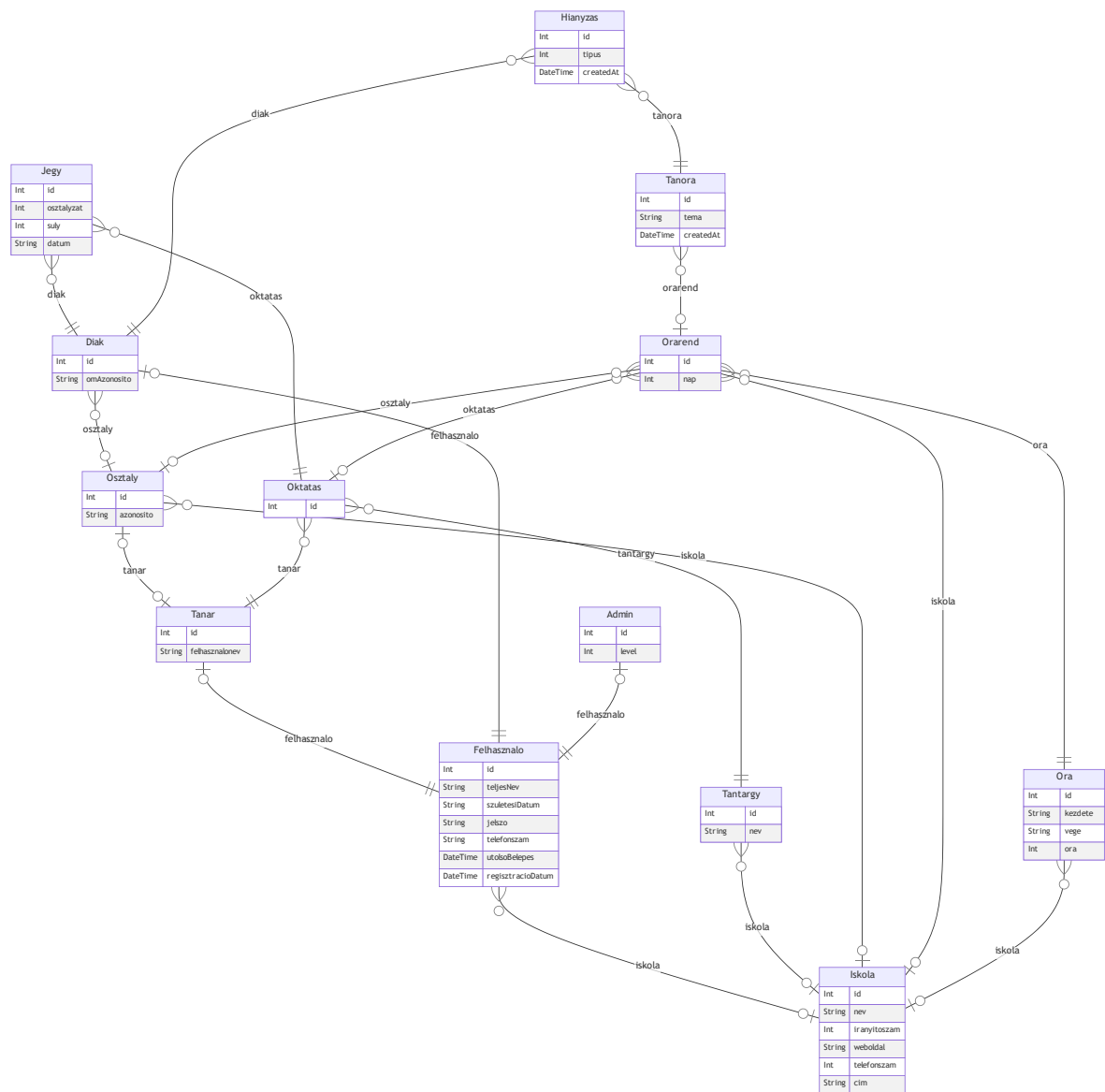
A projekt fejlesztéséhez **ReactJS** és **ExpressJS** keretrendszert használtam, a **Visual Studio Code** szerkesztőben. Az adatbázis kezelésére **MySQL**-t használtam. A fejlesztési folyamatot a **Git** verziókezelő rendszerrel követtem nyomon. A **Node.js** környezetet és az **NPM** csomagrendszert használtam a modulok telepítéséhez és a projekt build-eléséhez. **Webpack**-et a **Javascript** modulok kötegeltté konvertálásához, a **Babelt** a régebbi böngészőkkel való kompatibilitáshoz, az **ESLint**et a kód minőségének ellenőrzéséhez, a **Prettier**et pedig a kód formálásához használtam.

A projekthez használt modulok

- Prisma – ez a modul felel az adatbázis egyszerű és gyors működéséért. Lehetővé teszi az objektum-orientált fogalmakat a bonyolult, hosszabb SQL lekérdezések helyett.
- Jsonwebtoken – egyedi kulcsok (tokenek) titkosítására és megfejtésére használják, jelen esetben a bejelentkezéshez elengedhetetlen.
- Vite – által létrehozott fájlok sokkal kisebbek, mint a hagyományos modulcsomagolók (bundlerek) által létrehozott fájlok. Ez csökkenti a weboldal betöltési idejét és felel a ReactJS optimalizálásáért.
- Axios – felel az adatok küldéséért és lekérdezéséért. Szintaxisa könnyen és egyszerűen megérthető, valamint átlátható.
- TailwindCSS – segítségével gyorsan létrehozható egy modern reszponzív és jól kinéző weboldal anélkül, hogy egyéni CSS-t kellene írni.
- React-hot-toast – stílusos üzenetek megjelenítésére szolgál, visszajelzést nyújt a felhasználónak.
- React-icons – előre elkészített ikonbázis, használata feldobhatja a weboldal kinézetét.
- Moment – dátumok egyszerűbb kezelésére szolgál.
- React-router – az oldalak közötti lépegetést és navigációt segíti. Könnyen navigálhatnak a felhasználók a különböző funkciók között.
- DaisyUI – előre legyártott stílusos komponensek és funkciók használatára szolgál. Alapjaként elengedhetetlen a TailwindCSS.

- Mui – Material UI által elkészített felületek használatát segíti, hatalmas előnye hogy könnyen testre szabható.

Adatmodell leírása



Felhasznalo

- Id
- teljesNev
- szulesetiDatum
- jelszo – titkosított jelszó
- telefonszam
- utolsóBelepes – utolsó belépés dátuma
- regisztracioDatum – regisztráció dátuma
- iskola – a felhasználóhoz tartozó iskola
- tanar – a felhasználóhoz tartozó tanár tábla, ha tanár
- diak – a felhasználóhoz tartozó diák tábla, ha diák

admin - a felhasználóhoz tartozó admin tábla, ha admin

Tantargy

nev – a tantárgy neve

iskola – a tantárgyhoz tartozó iskola

oktatasok – a tantárgyat oktató tanárok

Tanar

felhasznalo – a tanárhoz tartozó felhasználó

felhasznalonev – a belépéshez szükséges felhasználónév

osztaly – a tanár osztálya, ha osztályfőnök

oktatasok – a tanár által tanított tantárgyak

Osztaly

azonosito – az osztály azonosítója pl. 9/A

tanar – az osztályfőnök

iskola – az osztályhoz tartozó iskola

diakok – az osztály diákjai

orak – az osztály órarendje

Diak

omAzonosito – a belépéshez szükséges oktatási azonosító

osztaly – a diák osztálya

felhasznalo – a diákhoz tartozó felhasználó

jegyek – a diák jegyei

hianyzasok – a diák hiányzásai

Oktatas

tanar – az oktatáshoz tartozó tanár

tantargy – az oktatáshoz tartozó tantárgy

jegyek – kiosztott érdemjegyek

orak – a tantárgyhoz beosztott órák

Jegy

osztályzat – az érdemjegy

suly – az osztályzat súlya, alapból 100%

datum – az osztályzás dátuma

oktatas – az osztályzatot kiadó oktató

diak – az értékelt diák

Admin

level – az adminisztrátor szintje

felhasznalo – az adminhoz tartozó felhasználó

Orarend

nap – a hét napja számban megadva, pl 1 = Hétfő

ora – az órához tartozó csengetési óra

oktatas – az órához tartozó oktató

iskola – az órához tartozó iskola

osztaly – az órarendhez tartozó osztály

tanorak – az órához tartozó lekönyvelt órák

Ora

kezdete – a csengetés kezdete, pl 7:45

vege – a csengetés vége, pl 8:30

ora – az óra száma, pl 1

orarendek – ehhez a csengetéshez tartozó órák

iskola – a csengetéshez tartozó iskola

Iskola

nev – az iskola neve

iranyitoszam – az iskola irányítószáma

weboldal – az iskola weboldalának címe

telefonszam – az iskola telefonszáma

cím – az iskola helyrajzi címe

felhasznalok – az iskolához tartozó felhasználók

orarendek – az iskolához tartozó órarend

osztalyok – az iskolához tartozó osztályok

orak – az iskolához tartozó csengetési rend
tantargyak – az iskolához tartozó tantárgyak

Tanora

orarend – a lekönyvelt tanórához tartozó óra
tema – a tanóra témája
hiányzasok – a tanórán lekönyvelt hiányzó diákok

Hianyzas

tipus – a hiányzás típusa számban megadva, 0 = Igazolatlan, 1 = Orvosi igazolt, 2 = Szülői igazolt, 3 = Iskolai igazolt
diak – a hiányzó diák
tanora – a tanóra amelyiken a hiányzást rögzítették

Algoritmusok

ProtectedRoute

Az alábbi kódrészlet egy olyan komponenst mutat meg, ami megakadályozza a jogosulatlan felhasználókat bizalmas és személyes adatok hozzáféréséhez. Mielőtt a felhasználó egy oldalra lépne át, ez az algoritmus megnézi és megvizsgálja a jogosultságait, ha nincs elegendő joga a megnyitni kívánt oldalhoz az algoritmus automatikusan visszairányítja a bejelentkező vagy főoldalra. Az algoritmus több paramétert is befogad. A tanár (teacher) paramétert akkor használjuk, amikor egy olyan oldalt szeretnénk levédeni amihez csak tanári jogosultságokkal rendelkező felhasználó férhet hozzá. Admin paramétert pedig abban az esetben használunk, ha a levédeni kívánt oldal csak adminisztrátori jogokkal látogatható.

```
const ProtectedRoute = ({ children, teacher, admin }) => {
  const { user } = useAuthContext()

  if (!user) {
    return <LoginPage />
  }
  if (teacher && !user?.tanar) {
    return (
      <Navbar>
      | <HomePage />
      </Navbar>
    )
  }
  if (admin && !user?.admin) {
    return (
      <Navbar>
      | <HomePage />
      </Navbar>
    )
  }
  return children
}

export default ProtectedRoute
```

Bejelentkező algoritmus

```
router.post("/login", async (req, res) => {
  const { username, password } = req.body
  const student = await prisma.diak.findFirst({
    where: {
      omAzonosito: username
    },
    include: {
      felhasznalo: true
    }
  })

  const teacher = await prisma.tanar.findFirst({
    where: {
      felhasznalonev: username
    },
    include: {
      felhasznalo: true
    }
  })

  const user = student || teacher

  if (!user) {
    res.status(400).send("Nem létezik ilyen felhasználó!")
  }

  bcrypt.compare(password, user?.felhasznalo.jelszo, (err, result) => {
    if (result) {
      res.status(200).json({
        token: generateAccessToken(user.felhasznalo.id)
      })
    } else {
      res.status(400).send("Helytelen felhasználónév vagy jelszó!")
    }
  })

  res.status(401).send("Hiba történt!")
})
})
```

Ez a kódrészlet a program egyik legfontosabb és legmeghatározóbb eleme. Ez felel a bejelentkezés sikerességéért vagy éppen sikertelenségéért. Amikor a felhasználó a bejelentkezés gombra nyom, a kinézetért és stílusos felületért szolgáló ‘frontend’ oldal átküldi a felhasználó által beírt adatokat a háttérfolyamatokat kezelő ‘backend’ oldalnak. A ‘backend’ oldal megkapja az adatokat, tanár esetében felhasználónevet és jelszót, diák esetében OM azonosítót és jelszót. Ezután megkeresi, hogy létezik-e olyan felhasználó az adatbázisban, ami megegyezik a megadott adatokkal, ha megegyezik, akkor továbblép a következő fázisra. Ha viszont nem talál az adatbázisban a megadott felhasználónévvel és jelszóval adatot, akkor a felhasználó a ‘Nem létezik ilyen felhasználó!’ üzenetet kapja. Megegyező adatok esetén a program összehasonlítja a megadott jelszót az adatbázisban szereplő felhasználó jelszával, miután a két jelszó egyezik, a program generál egy egyedi és sajátos kulcsot a felhasználó számára és visszaküldi a ‘frontend’ oldalnak. Más különben a felhasználó nem tud belépni és a ‘Helytelen felhasználónév vagy jelszó!’ hibaüzenetet kapja.

Egyedi token (kulcs) generálása

```
const generateAccessToken = (id) => {  
  return jwt.sign({id}, process.env.JWT_SECRET, {  
    expiresIn: "1d"  
  })  
}  
  
export default generateAccessToken
```

Sikeres bejelentkezés esetén a 'backend' oldal ezt az eljárást használja az egyedi kulcs legenerálására. A generálás a 'jsonwebtoken' modul segítségével jön létre. A modul lehetővé teszi, hogy a letöltése után kulcsok létrehozására, valamint leellenőrzésére használják. A létrehozáshoz szükségünk van egy 'jwt secret' azaz egy titkos és biztonságos jelszóra, amit a modul arra használ, hogy a jelszó használatával ellenőrizi és létrehozza a kulcsokat. Hibás jelszó megadása esetén a modul nem lesz képes ezekre a folyamatokra. Amint elkészül az egyedi kulcs, az algoritmus visszaadja a 'backend' oldalnak.

Bejelentkezés esetén a felhasználó egyedi kulcsa 1 napig érvényes, azután a program megint kérni fogja a bejelentkezésre. Mindezt biztonsági okokból választottam, mivel egy iskolaszintű alkalmazásnál ez a védelem elengedhetetlen.

Tesztelés

Manuális tesztelés során új diák felvitelekor nem adtunk meg osztályt és telefonszámot. Az osztály megadásának hiánya miatt kidobott a rendszerből (8. ábra).



```
C:\Windows\system32\cmd.exe

PrismaClientKnownRequestError:
Invalid "prisma.felhasznalo.create()" invocation:

An operation failed because it depends on one or more records that were required but not found. No 'Osztaly' record(s) (
needed to inline the relation on 'Diak' record(s)) was found for a nested connect on one-to-many relation 'DiakToOsztaly
'.
    at In.handleRequestError (C:\Users\bewei\Documents\filc\backend\node_modules\@prisma\client\runtime\library.js:122:6
854)
    at In.handleAndLogRequestError (C:\Users\bewei\Documents\filc\backend\node_modules\@prisma\client\runtime\library.js
:122:6188)
    at In.request (C:\Users\bewei\Documents\filc\backend\node_modules\@prisma\client\runtime\library.js:122:5896)
    at async 1 (C:\Users\bewei\Documents\filc\backend\node_modules\@prisma\client\runtime\library.js:127:10871)
    at async file:///C:/Users/bewei/Documents/filc/backend/controllers/register.js:13:18 {
  code: 'P2025',
  clientVersion: '5.11.0',
  meta: {
    modelName: 'Felhasznalo',
    cause: "No 'Osztaly' record(s) (needed to inline the relation on 'Diak' record(s)) was found for a nested connect on
one-to-many relation 'DiakToOsztaly'."
  }
}

Node.js v20.12.1
[nodemon] app crashed - waiting for file changes before starting...
```

8. ábra

A hibát javítottam, a program figyelmeztet, hogy az osztály megadása kötelező a diák felvitelekor.

Mulasztások rögzítésekor Nagy Árpád és László Erika lett hiányzónak rögzítve (9. ábra), de a mulasztások megtekintésekor nem mutatta László Erika hiányzását.

Osztályközösség-építés

Szerkeszthető

Tanóra témája

Ügyfélkapu regisztráció

Jelenlét

Nagy Árpád ☒

László Erika ☒

Kovács Kristóf ☐

Mentés

9. ábra

Órarendösszeütközés miatt keletkezett a hiba.

Minden 'backend' oldal felőli végpont (API) átesett legalább egy teszten, ez azt takarja, hogy minden végpont tesztelve lett a Visual Studio Code kódszerkesztőben a Thunder Client nevezetű bővítmény segítségével. Ebben a bővítményben megadhatjuk a végpont hivatkozását (pl.: <http://localhost:5000/auth/login>). Ezt követően saját magunk által megadott adatokkal tesztelhetjük az elérési pontot, mindezt az oldalon bármiféle felület használata nélkül.

Fejlesztési lehetőségek

A program a későbbiekben bővíthető a felhasználók további adataival. Például a diákok esetében további személyes adatokat is rögzíteni lehet, mint születési helye, anyja leánykori neve, gondviselője, lakcíme, e-mail címe, tanulói jogviszonyra vonatkozó adatok, tankötelezettség figyelemmel követése. Ifjúságvédelemre vonatkozó adatok, mint például a rendszeres gyermekvédelmi támogatás, vagy hátrányos, halmozottan hátrányos helyzet rögzítésére is lehetőséget lehetne biztosítani.

Bővíteni lehet még Kollégium modullal, hogy a kollégiumi ellátásban részesülő tanulók nyomon követhetők legyenek. A kollégiumi nevelőtanárok rögzítésével megoldható lenne, hogy figyelemmel tudják kísérni a diákok tanulmányi előmenetelét és tanórai hiányzásait, így azonnal értesülnének a tanulókkal kapcsolatos minden fontos információról. Ezzel lecsökkenthető lenne az információáramlás ideje, már nem lenne arra szükség, hogy az osztályfőnök keresse a diákok kollégiumi nevelőtanárát a hiányzások jelzése, vagy bukásra állás ténye miatt.

Fejleszthető lenne még a diákétkeztetésre vonatkozó Menza modullal, mellyel kiegészítve a mulasztások miatti étkezések lemondása megkönnyítené jelentősen az étkezésnyilvántartással foglalkozók munkáját.

A további adminisztratív feladatokat könnyíthetné, ha lehetőség lenne olyan levelek generálására, melyek például a szülőket értesítik a gyermekük hiányzásáról, vagy figyelmeztetik arról, hogy tanulmányi eredménye alapján bukásra áll. További felhasználóként a szülőket is be lehetne vonni, hogy elektronikus úton tudjanak tájékozódni a gyermekükkel kapcsolatban.

Összefoglalás

A záróvizsgám témájának egy olyan weboldal készítését tűztem ki célként, mely megkönnyíthetné az adminisztratív feladatok elvégzését és egy oldalon lekérdezhetővé, átláthatóvá tenné a teljes iskolában folyó munkát. A projekt készítése során kihívást jelentett az adatbázis megtervezése, kezelőfelületek megalkotása. Úgy gondolom, hogy munkám során alaposan elmélyedhettem a backend fejlesztésben.

Természetesen érzem, hogy olyan témát választottam, mely még bőven tovább fejleszthető, hiszen a weboldal elkészítésére rendelkezésre álló idő a téma összetettsége miatt kevés volt számomra. Úgy érzem, hogy projektmunkámmal bővíthettem tudásomat és tapasztalatot szerezhettem arról, hogy milyen nehézségekkel és kihívásokkal találkozhatnak szoftverfejlesztőként.

Végül szeretnék köszönetet mondani szakmai tanáraimnak az útmutatásaikért, segítségükért és családomnak a támogató háttér biztosításáért.

Irodalomjegyzék

1. Dokumentáció záródolgozathoz, vizsgaprojekthez
<https://infojegyzet.hu/webszerkesztes/dokumentacio/>
2. A szoftverfejlesztés. Tesztelés
https://www.informatika-programozas.hu/informatika_java_programozas_gyakorlat_2_szoftverfejlesztes_5a.html
3. Prisma
<https://www.prisma.io/>
4. ExpressJS
<https://expressjs.com/>

Plágium-nyilatkozat

Alulírott SZEREDI ÁKOS (név)

Gyula..... (szül.hely)

2005. 02. 06..... (szül.idő)

Bányai Erzsébet (anya neve)

jelen nyilatkozat aláírásával kijelentem, hogy a

Filc weboldal..... 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, 20hónap

Tanuló aláírása