BKSZC Pogány Frigyes Technikum

Sorozatnapló

Készítették:  
Orosz Ákos  
Ur Bálint

A téma:  
Egy sorozatnapló weboldalt csináltunk ahol az emberek regisztráció után be tudnak jelentkezni, majd sorozatot felvinni és beállítani, hogy hol tartanak. A sorozatokat módosítani, illetve törölni is lehet.

Célja az oldalnak:

Célunk hogy felhasználó barát felületet hozzunk létre azon emberek számára, akik mindennapi tevékenységeik mellett el tudják menteni epizódra pontosan a kedvenc sorozataikat, hogy el ne felejtsék őket.

Kinek hasznos?

Aki szeretné szépen levezetve számon tartani sorozatait annak jó választás lehet az oldal.

Fejlesztői dokumentáció

Fejlesztői környezet:

A projekt elkészítéséhez főleg Visual Studio Code-ot használtunk, ami a kódoláshoz kellett. A backend elkészítéséhez Express NodeJS volt használva, illetve Xampp kontroll panelját használtuk azon belül is az Apache és a MySQL funkcióit, ami az adatbázis csatlakoztatásához kellett.  
  
Elindítás:

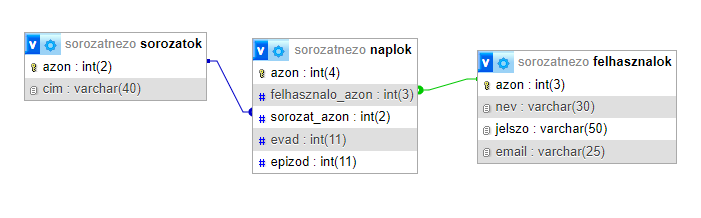
Ügyeljünk rá hogy a xampp-ban el van indítva a Apache és a MySQL illetve a Visual Studio Code-ban nyissunk meg egy új terminált.  
Miután letöltöttük a node elemeket (npm i) a backend mappában, adjuk ki a „node app.js” parancsot és máris elérhetővé válik a program a 3000. porton.

GitHub repository:

A projekt ebben a repository-ban érhetők el:

<https://github.com/Almaleee/BalintAkosproject>

## Az adatbázis elkészítése:



## A „naplok” adatai:

***naplok (azon, cím, felhasznalo\_azon, sorozat\_azon, evad, epizod)***

Az előrehaladás azonosítója (automatikusan növekedik), ez a kulcs

A naplózott felhasználó azonosítója, külső kulcs

A naplózott sorozat azonosítója, külső kulcs

Az előre haladásban résztvevő felhasználó sorozat évadsorszáma

Az előre haladásban résztvevő felhasználó sorozatának évadon belüli epizódsorszáma

## A „felhasznalok” adatai:

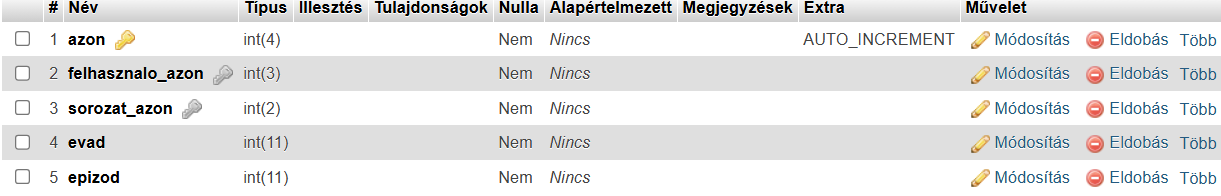
***felhasznalok (azon, nev, jelszo, email)***

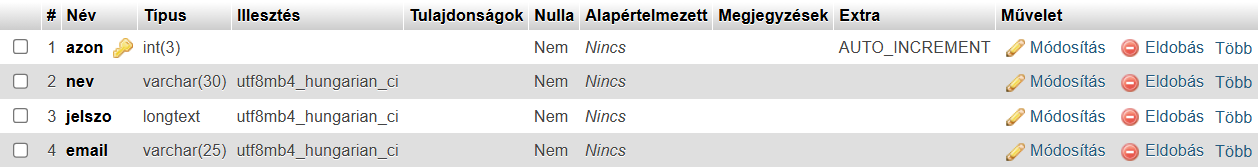
A felhasználó azonosítója (automatikusan növekedik), ez a kulcs

A felhasználó neve

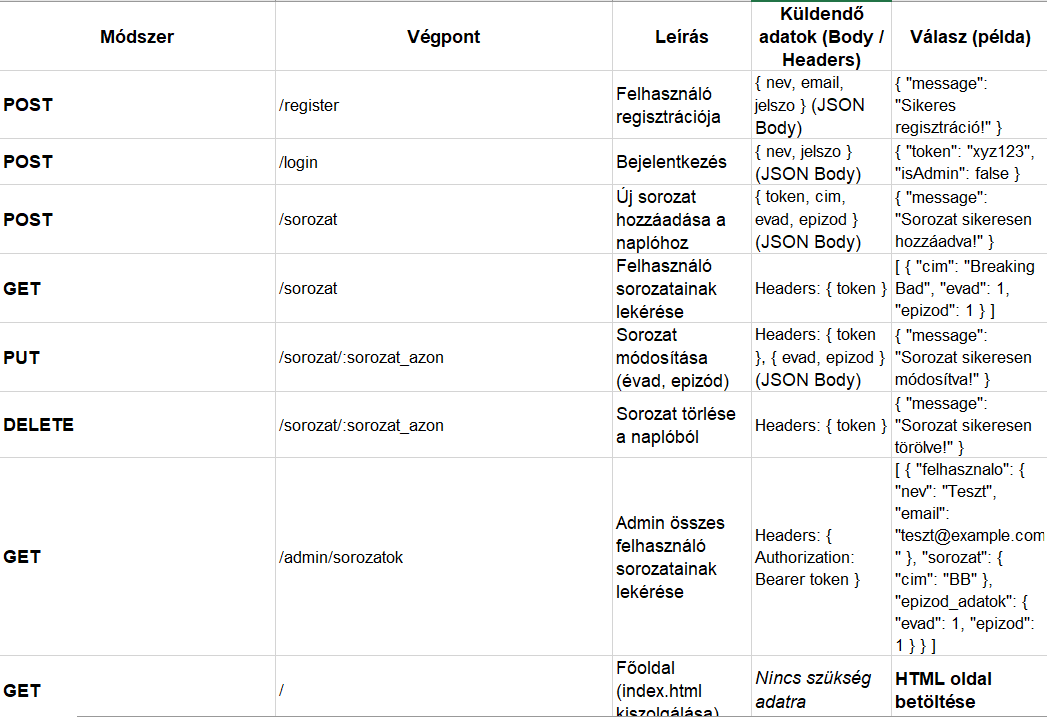
A felhasználó jelszava

A felhasználó email címe





Api végpontok:



Regisztráció backend és frontend részről:  
  
Backend:  
A kliens által küldött **név**, **e-mail cím** és **jelszó** adatait a req.body tartalmazza. A jelszót biztonsági okokból a szerver **bcrypt** könyvtárral hashelve (titkosítva) tárolja. Az adatok az „felhasznalok” táblába kerülnek az adatbázisban egy **INSERT** SQL utasítással.

Sikeres regisztráció esetén a szerver **201 (Created)** státuszkóddal egy "Sikeres regisztráció!" üzenetet küld vissza. Ha hiba történik (pl. adatbázis-hiba), a szerver **500 (Internal Server Error)** státuszkóddal egy "A regisztráció sikertelen." üzenetet küld a kliensnek.

Frontend:  
A függvény a HTML oldalon található beviteli mezőkből (regName, regEmail, regPassword) lekéri a felhasználó által megadott adatokat, majd egy **POST** kérést küld a ${apiUrl}/register végpontra. A kérés törzse JSON formátumban tartalmazza a nevet, e-mail címet és jelszót. A szervertől érkező választ JSON-ként dolgozza fel, majd a kapott üzenetet vagy hibaüzenetet egy felugró ablakban (alert) jeleníti meg.

Bejelentkezés backend és frontend részről:

Backend:  
Ha a felhasználónév „admin”, a szerver összehasonlítja a megadott jelszót a környezeti változóban (process.env.ADMIN\_PASSWORD) tárolt admin jelszóval. Ha egyezik, egy **JWT tokent** generál, amely az admin jogosultságot tartalmazza, és **1 óráig érvényes**. Hibás jelszó esetén **400 (Bad Request)** hibaválaszt küld.

Egyéb esetben a szerver az adatbázisból („felhasznalok” tábla) lekéri a megadott nevű felhasználót. Ha nem található, **400 (Bad Request)** státuszkóddal hibát küld vissza. Ha a felhasználó létezik, a megadott jelszót **bcrypt** segítségével összehasonlítja az adatbázisban tárolt hash-elt jelszóval.

Hiba esetén a szerver **500 (Internal Server Error)** státuszkóddal "Szerverhiba." üzenetet küld.

Frontend:

Sikeres bejelentkezés esetén a kapott **JWT tokent** a **localStorage**-ban tárolja. Ha a felhasználó „admin”, meghívja a loadSeriesForAdmin() függvényt, egyébként a loadSeries() függvényt. Ezt követően elrejti a bejelentkezési és regisztrációs felületet, és megjeleníti a sorozatok kezeléséhez szükséges elemeket. Sikertelen bejelentkezés esetén egy felugró ablakban (alert) megjeleníti a szerver által küldött hibaüzenetet.

Admin a backendben:  
Ez a végpont az „admin” felhasználó számára lehetővé teszi a sorozatok és a felhasználói naplók lekérdezését. A kéréshez szükséges **JWT token** a Authorization fejlécben érkezik. A token érvényességének ellenőrzése után, ha a felhasználó admin jogosultsággal rendelkezik, lekérdezi az adatbázisból a felhasználókhoz tartozó sorozatokat és azok epizódjait.

Ha a token érvénytelen vagy lejárt, a szerver **401 (Unauthorized)** hibát küld. Ha a felhasználó nem admin, akkor **403 (Forbidden)** választ ad. Sikeres lekérdezés esetén a sorozatok adatai JSON formátumban kerülnek visszaküldésre. Hiba esetén a szerver **500 (Internal Server Error)** hibát küld a hiba részleteivel.

Admin felülete:   
Pontosan ugyan olyan, mint a felhasználóké, csak az admin látja minden felhasználó összes sorozatait.

Sorozatok kezelése frontendben:

Hozzáadás:  
A **cim**, **evad** és **epizod** adatokat a megfelelő beviteli mezőkből (seriesTitle, seriesSeason, seriesEpisode) lekéri, és egy **POST** kérést küld a ${apiUrl}/sorozat végpontra. A kérés törzse JSON formátumban tartalmazza a token-t és a sorozat adatait.

A válaszban kapott üzenetet vagy hibaüzenetet egy felugró ablakban (alert) jeleníti meg. Ha sikeres a sorozat hozzáadása, a loadSeries() függvény frissíti a sorozatok listáját.

Módosítás:

A felhasználó által megadott új **evad** és **epizod** értékeket a prompt() segítségével kéri be. Ha mindkét értéket megadja a felhasználó, egy **PUT** kérést küld a ${apiUrl}/sorozat/${sorozat\_azon} végpontra, amely tartalmazza a frissített évad- és epizódadatokat, valamint a **JWT tokent** a kérés fejlécében.

A szerver válasza alapján a módosítás sikerességét vagy hibáját egy felugró ablakban (alert) jeleníti meg. Ha a módosítás sikeres, a loadSeries() függvény frissíti a sorozatok listáját.

Törlés:

A felhasználót egy **confirm** párbeszédablakban megkérdezi, hogy biztosan törölni akarja-e a sorozatot. Ha a felhasználó megerősíti, egy **DELETE** kérést küld a ${apiUrl}/sorozat/${sorozat\_azon} végpontra, a törlendő sorozat azonosítójával és a **JWT tokennel** a kérés fejlécében.

A szerver válasza alapján a törlés sikerességét vagy hibáját egy felugró ablakban (alert) jeleníti meg. A törlés után a loadSeries() függvény frissíti a sorozatok listáját.

Felhasználói dokumentáció

Hogyan érheted el a weboldalt?

Bármilyen okoseszközzel el tudod érni az oldalt, ami képes egy böngésző futtatására. Ilyen például:

* laptop
* számítógép
* okostelefon
* tablet

Használd bármelyiket nyugodtan.

Fontos megjegyezni, hogy bármely böngészővel el lehet érni az oldalt.

Lépések:

1. Nyisd meg a böngészőt.
2. Üsd be az URL linket.
3. Ez után már kezdheted is.

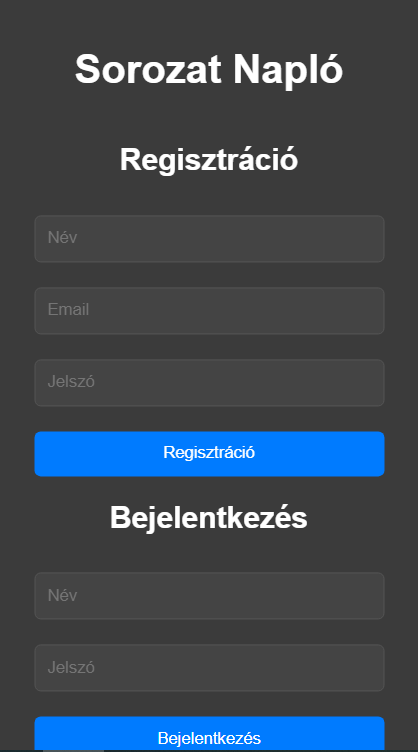
A „Rólunk” oldal:

A fent középen lévő „Rólunk” gombbal többet is megtudhatsz az oldalról és rólunk.



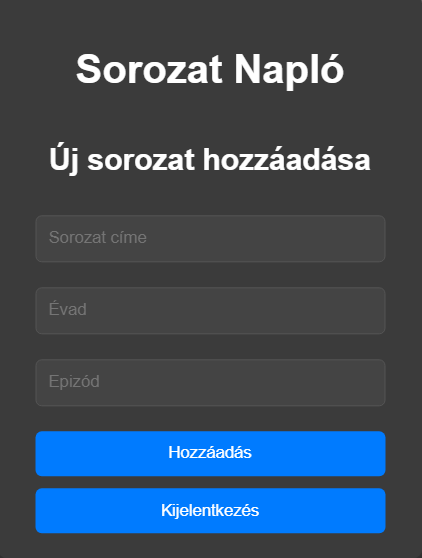
A Főoldal:

A Főoldal felülete egyszerű, így te is át tudod látni:



Bejelentkezés után:

Bejelentkezés után hozzá tudod adni elkezdett sorozataidat, és ha végeztél ki is tudsz jelentkezni.



Gombok:

Ha haladtál sorozataiddal módosítani tudod őket, vagy ha befejezted, akkor ki is tudod törölni.

