**Szakképesítés neve:** ……………………………………

**OKJ száma:** …………………

**SZAKDOLGOZAT**

SZAKDOLGOZAT CÍME

Témavezető:  
**Beke Béla**

Készítette:  
**Pataki Dávid Ferenc**

Debrecen  
2023

# Tartalomjegyzék

[Tartalomjegyzék 2](#_Toc127901388)

[1 Bevezetés 4](#_Toc127901389)

[1.1 Miért ezt választottam? 4](#_Toc127901390)

[1.2 Köszönetnyilvánítás 4](#_Toc127901391)

[2 Felhasználói dokumentáció 5](#_Toc127901392)

[2.1 Rendszerkövetelmény 5](#_Toc127901393)

[2.2 Weboldal használata 5](#_Toc127901394)

[3 Fejlesztői dokumentáció 6](#_Toc127901395)

[3.1 Használt npm csomagok 6](#_Toc127901396)

[3.2 Telepítés 7](#_Toc127901397)

[3.3 Backend 7](#_Toc127901398)

[3.3.1 Tervezési minta (Architektúra) 7](#_Toc127901399)

[3.4 Adatbázis 8](#_Toc127901400)

[3.4.1 User tábla 8](#_Toc127901401)

[3.4.2 Login tábla 9](#_Toc127901402)

[3.4.3 Follow tábla 9](#_Toc127901403)

[3.5 Algoritmusok 10](#_Toc127901404)

[3.6 Tesztdokumentáció 10](#_Toc127901405)

[3.7 Fejlesztői környezet 10](#_Toc127901406)

[4 Összefoglalás 11](#_Toc127901407)

[5 Irodalomjegyzék 12](#_Toc127901408)

# Bevezetés

A záródolgozatom témája egy filmeket értékelő és ajánló platform, ahol a felhasználók bejegyzéseket írhatnak megtekintett filmjeikről, hogy megosszák élményüket másokkal is. A felhasználók követhetnek más fiókokat, hogy könnyedén lássák azok értékeléseit.  
A weblap funkciói közt említést érdemel még a filmek keresése név, értékelések és műfajai alapján.

## Miért ezt választottam?

Ezt a témát választottam, mert szeretek filmeket nézni és érdekesnek találtam a filmekről való vélemények összegyűjtését és megosztását. Továbbá úgy véltem, egy ilyen weboldal hasznos lehet mások számára is, hogy segítsen nekik a film választásában.

## Köszönetnyilvánítás

# Felhasználói dokumentáció

## Rendszerkövetelmény

## Weboldal használata

# Fejlesztői dokumentáció

A weboldal fejlesztéséhez [**Node.js**](https://nodejs.org/en/)-t alkalmaztam. Azért ezt használtam mivel a JavaScript programozási nyelvet jobban elsajátítottam szabad időmben, mint más interaktív weboldal készítésére használt nyelveket, például a PHP-t. Abban is előnyösebb nekem ez a környezet, mert a Frontend - Backend kódot jobban tudom szeparálni.

A JavaScript programozási nyelvet kiegészítem a **TypeScript** nevezetű szintaxis és statikus típus ellenőrző készlettel, a fölösleges hibák elkerülése érdekében.

A Node.js-en belül a [**Next.js**](https://nextjs.org/) **keretrendszert** használom, ami a React frontend könyvtárat kiegészíti egy Express-hez hasonló backend RESTful API készítő csomaggal.

A [**React**](https://reactjs.org/) a *Meta* által létrehozott JavaScript könyvtár, mellyel könnyen lehet készíteni interaktív felhasználói felületeket.

Adatbázisnak a MySQL alapú **MariaDB**-t alkalmaztam.

## Használt npm csomagok

A fentieken kívül használtam még másmilyen különböző úgynevezett npm csomagokat kisebb problémák megoldására:

* bcrypt: Egy Kriptografikus könyvtár, amely a jelszók biztonságosabb eltárolása érdekében használok.
* jsonwebtoken: Röviden JWT, egy token alapú autentikációs módszer, amely a session-el ellentétben a kliensen tárolja a bejelentkezett fiók adatait. A szerver csak egy kriptografikus kulcs segítségével validálja a klienstől kapott tokent.
* cookies-next: Next.js-ben nincs alapból cookie-k módosítására lehetőség, de van erre kifejlesztett csomag a készítőktől.
* mysql2: Az adatbázis kapcsolat létrehozására és felhasználására használt könyvtár

## Telepítés

A project optimális futása érdekében erősen ajánlott a [Fejlesztői környezet](#_Fejlesztői_környezet)-ben megjelölt Node.js verzió használata. Régebbi, ritka esetben akár új verziók is, képesek előre nem látható problémákat okozni, amelyek akadályozhatják a program futását.

A /next.config.js állományban érdemes átírni az adatbázis hitelesítő adatait.

databaseHost: elérés cím  
databaseUser: felhasználói fiók  
databasePassword: felhasználó jelszava  
databaseDatabase: adatbázis neve

A /scripts mappában található telepítési és indítási scriptek Linux-ra és Windows-ra, ha a parancssor túl rémisztő.

* ./install: telepíti a npm csomagokat, létrehozza az adatbázist és dinamikusan generál titkos szerver oldali tokeneket a kriptografikus műveleteknek.
* ./start-dev: Elindítja a fejlesztői környezetet, ebben a módban a projectben történő változtatások egyből megváltoznak a weboldalon is, viszont ez fölösleges rendszer erőforrásokat és optimalizálatlan kódot futtat ezért a weboldal lassabbnak tűnhet.
* ./build & ./start-prod: Ezzel a két scripttel lehet egy optimalizáltabb környezetet létrehozni a projectnek, viszont minden változtatás után újra le kell futtatni a építés parancsot.

## Backend

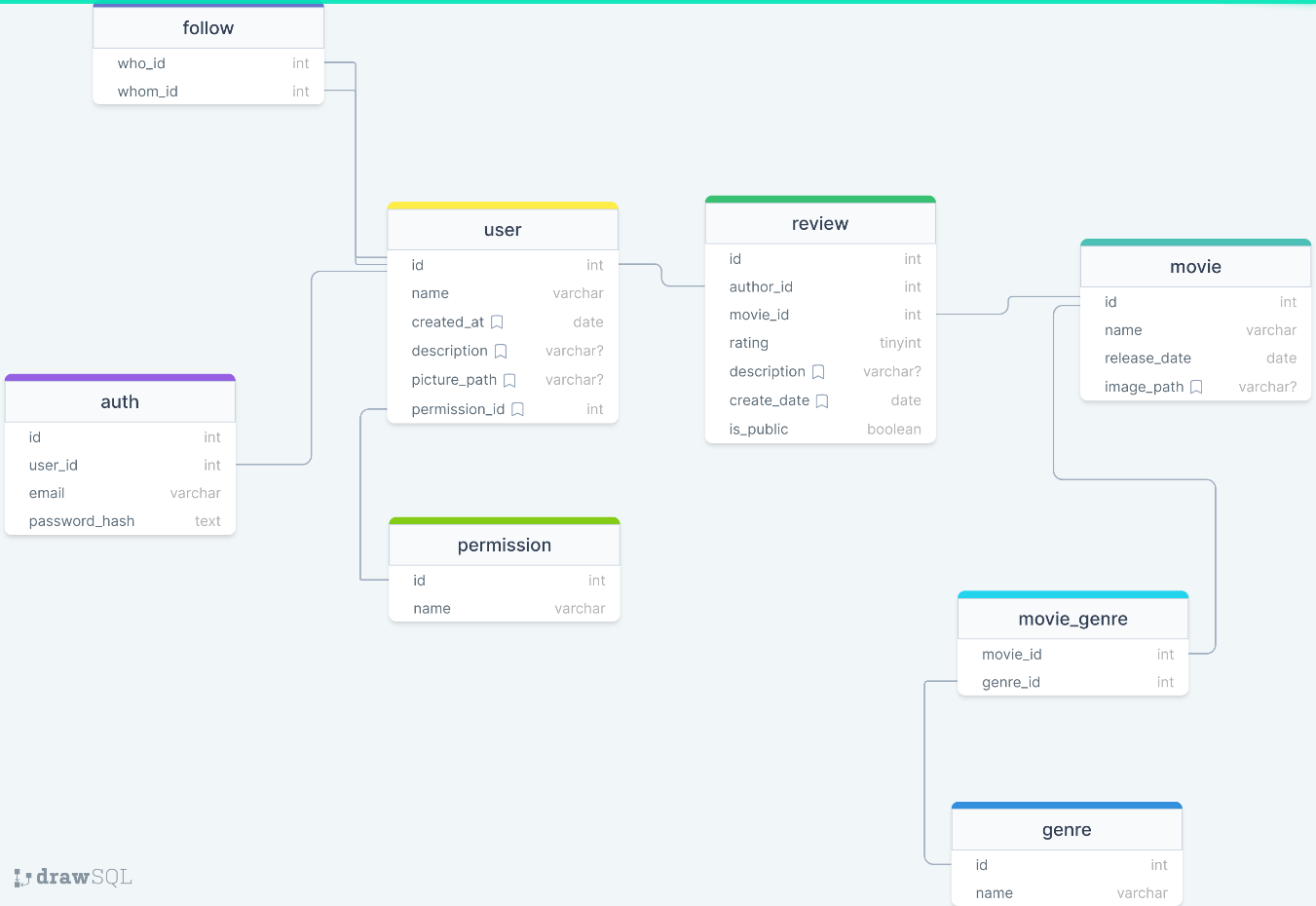
### Tervezési minta (Architektúra)

A backend kód struktúrája a Model-Service-Controller (Modell-Szolgáltatás-Vezérlő) architektúrán (röviden MSC architektúra) alapul. Minden API kérés ezen a „vezetéken” megy át. Három fontos részből áll:

1. Controller: A beérkező kéréseket irányítja a megfelelő irányba, ezen a rétegen történik a http kérésének szintaktikai ellenőrzése, és a http válaszok visszaküldése.
2. Service: Ezen a rétegen történik a komplexebb ellenőrzések (pl. felhasználó név létezik e már az adatbázisban), ez a réteg konkrétan nem foglalkozik az adatbázissal, sem a http kapcsolatokkal, de viszont egy fontos átmeneti réteg a következő rétegnek.
3. Model: Feladata az adatok lekérdezése az adatbázistól és annak értelmezése.

Ahogy lehet látni minden rétegnek meg van a saját felelőssége. Kisebb projektekben túlzás ilyen architektúrákat alkalmazni, lassítja a fejlesztési sebességét, viszont ezzel a tervezési mintával javítható a szoftver skálázhatósága és karbantarthatósága.

## Adatbázis



### User tábla

Ez a tábla a felhasználók alap adatait tárolja.

|  |  |
| --- | --- |
| * **ID**: szám típusú, automatikusan generált, egyedi elsődleges kulcs. * **UserID**: (TODO: ez még nem biztos, hogy megmarad). * **Username**: Szöveg típusú, a fiók felhasználóneve. * **DateJoin**: Dátum típusú, automatikusan generált, a fiók létrehozásának dátumát jelöli. * **Picture**: Bináris objektum, (TODO: ezt lehet, hogy nem adatbázisba tárolom) a felhasználó profilképe. * **Description**: Szöveg típusú, felhasználó leírása, nem kötelező. * **PermissionID**: Idegen kulcs, ami meghatározza a felhasználó jogait. |  |

### Login tábla

Ez a tábla tartalmazza a felhasználók authentikációs adatait.

* **UserID**: Idegen kulcs, melyik felhasználó belépési adatai.
* **Email**: Szöveg típusú, a felhasználó email címe.
* **PasswordHash**: Fix hosszúságú szöveg, a jelszavakat soha nem tároljuk egy per egy az adatbázisban, hanem a jelszót odaadjuk egy egy-irányú enkriptációs algoritmusnak és csak az eredményt tároljuk.
* **PasswordSalt**: Fix hosszúságú szöveg, random generált szöveg, amit a jelszóhoz adunk enkriptálás előtt.

A jelszó tárolásáról a [Algoritmusok](#_Algoritmusok)(TODO) fejlécben többet megtudhatunk.

### Follow tábla

Követések feljegyzésére szolgáló tábla.

* **whoUserID**: Idegen kulcs, melyik felhasználó követ
* **whomUserID**: Idegen kulcs, melyik felhasználót követi

## Algoritmusok

## Tesztdokumentáció

## Fejlesztői környezet

# Összefoglalás

# Irodalomjegyzék

* <https://nextjs.org/docs>
* <https://reactjs.org/>
* <https://stackoverflow.com/>