Retro Film Kincsek

Informatikai rendszerterv

Tartalom

[1 Informatikai rendszerterv sablon 3](#_Toc132352072)

[2 Fogalmak és rövidítések 3](#_Toc132352073)

[3 Architektúra 4](#_Toc132352074)

[3.1 Áttekintés 4](#_Toc132352075)

[3.2 Választott technológiák 13](#_Toc132352076)

[3.3 Függőségek 16](#_Toc132352077)

[4 Modulok 16](#_Toc132352078)

[4.1 Áttekintés 17](#_Toc132352079)

[4.2 Modulok részletei 17](#_Toc132352080)

[5 Perzisztencia 18](#_Toc132352081)

[5.1 A tárolás választott módja 18](#_Toc132352082)

[5.2 Koncepcionális adatmodell 18](#_Toc132352083)

[6 Folyamatok 21](#_Toc132352084)

[7. Interfészek 21](#_Toc132352085)

[8. Biztonság 21](#_Toc132352086)

[9. Üzemeltetést támogató képességek 21](#_Toc132352087)

[9.1. Konfiguráció 21](#_Toc132352088)

[9.2. Monitorozás 21](#_Toc132352089)

[9.3. Logolás 21](#_Toc132352090)

[9.4. Hibaelhárítás 22](#_Toc132352091)

[7 Általános felhasználó: 23](#_Toc132352092)

[8 Bejelentkezett felhasználó: 26](#_Toc132352093)

# Informatikai rendszerterv sablon

Verzió: 1.0

A projectmunkám egy filmajánló oldal, ami a régebbi alkotásokra specializálódott. A filmek azon rajongóinak készítettem ezt az oldalt, akik kifejezetten szeretik a régebbi filmeket, illetve azoknak, akik esetleg emlékeznek egy filmre fiatalabb korukból, viszont az idő homályában elfeledték annak a címét, esetleg csak egy színész vagy egy megjelenési évre emlékeznek. Erre a problémára hozna megoldást az oldalam, hiszen rengeteg szűrési lehetőség van, illetve az is könnyíti a keresést hogy csak 2000 előtti filmek szerepelnek az oldalon, ezért ha egy rendező több filmet is csinált később, már sokkal inkább leszűkült a találatok száma.

Azért választottam ezt a témát, mert régebben rendszeres program volt a filmnézés apukámmal, viszont ő sem emlékezett sokszor a film címére. Azt, hogy mikor látta a moziban vagy hogy milyen híresebb színész játszott benne azt meg tudta mondani, főleg az előbbit. Az IMDB-n viszont a rengeteg film miatt sokszor feladtuk a keresést, illetve az angol leírás sem segített minket közelebb az esti nézendő megtalálásához.

# Fogalmak és rövidítések

HTML5:

Az HTML5 egy olyan nyelv, amely lehetővé teszi a webfejlesztők számára, hogy jobban és könnyebben hozzáadjanak tartalmat és interakciókat az internetes oldalakhoz és webalkalmazásokhoz. Az HTML5 segít a fejlesztőknek abban, hogy videókat, animációkat és egyéb multimédiás tartalmakat használjanak az oldalakon, és a mobil eszközökkel is kompatibilisebbé teszi a weboldalakat. Az HTML5 sok új elemet és attribútumot vezetett be, amelyek segítségével a fejlesztők jobban és hatékonyabban tudnak dolgozni, és kreatívabb és jobban interaktív webalkalmazásokat hozhatnak létre.

CSS:

A CSS (Cascading Style Sheets) egy stílusleíró nyelv, amelyet a webfejlesztők használnak az internetes oldalak stílusának meghatározásához és formázásához. A CSS szétválasztja az oldal tartalmát és megjelenését, azaz lehetővé teszi a fejlesztők számára, hogy elkülönítsék az oldal tartalmát a formázástól.

MYSQL:

MySQL egy nyílt forráskódú, relációs adatbázis-kezelő rendszer. Az adatbázis-kezelő rendszerek olyan szoftverek, amelyek segítségével adatokat lehet tárolni, szervezni, kezelni és lekérdezni. Az adatokat táblákban tárolják, és a MySQL lehetővé teszi az adatok hatékony tárolását és keresését.

JAVAScript:

JavaScript egy szkriptnyelv, amely lehetővé teszi a weboldalak interaktív funkcióinak készítését. A JavaScript segítségével lehetőség van arra, hogy a weboldalak dinamikusabbá váljanak, többféle felhasználói interakcióval, animációkkal és más dinamikus elemekkel.

.NET:

.NET egy Microsoft által kifejlesztett szoftver platform, amely lehetővé teszi az alkalmazások fejlesztését és futtatását különböző operációs rendszereken, mint például a Windows, Linux és macOS.

EJS:

JS rövidítése az Embedded JavaScript-nek, azaz Beágyazott JavaScript-nek. Ez egy népszerű JavaScript sablonmotor, amely lehetővé teszi a fejlesztők számára, hogy dinamikus HTML tartalmat hozzanak létre JavaScript kód segítségével. Az EJS sablonokat sima JavaScript nyelven írják, és közvetlenül be lehet ágyazni őket HTML kódba. Az EJS egyszerű szintaxist biztosít a változók, ciklusok és feltételes állítások beágyazásához HTML sablonokba, ami lehetővé teszi dinamikus tartalom létrehozását adatbázisból vagy API-ból származó adatok alapján. Az EJS-t mind a kliensoldalon, mind a szerveroldalon használhatjuk, és gyakran használják Node.js és Express.js segítségével.

# Architektúra

## Áttekintés

Működési környezet:

Az alkalmazás Windows környezetre lett tervezve, tehát csak Windows operációs rendszerrel rendelkező felhasználók érik majd el. Egy .NET alkalmazást hoztam létre hogy biztosítsam az optimális teljesítményt.

Alkalmazásszerver:

Az alkalmazás egy NodeJS v16.13.2 alapú alkalmazásszerveren futtatható, amelynek feladata a felhasználó kéréseinek fogadás, feldolgozása, és a válaszok visszaküldése a kliens számára. Ez a szerver biztosítja a biztonságos és hatékony működését, valamint lehetőséget ad a különböző modulok és könyvtárak használata.

Adatbázisszerver:

Az alkalmazás adatbázisszervere egy MySQL adatbázis, amelyben a filmek és a felhazsnálók adatai egyaránt tárolásra kerülnek. Az adatbázisban található adatokhoz hozzáférnek a felhasználók, viszont változtatni csak kis szeletét tudják a táblázatok összességének.

Böngésző:

Az alkalmazás a felhasználók számára a böngészőjükön keresztül lesz elérhető. Az oldal kompatibilitását a legnépszerűbb böngészőkkel, mint a Chrome vagy a Firefox, teszteltük. Ezen felül, a felhasználói élmény érdekében reszponzív illetve dinamikus is.

**Működési környezet:** Windows környezet

* Windows .NET app
* Alkalmazásszerver - NodeJS v16.13.2
* MySQL adatbázisszerver 8.0
* Connector/NET 8.0.28
* Támogatott böngészők (Chrome, Firefox)
* Használt protokollok (http, https)
* SSL tanúsítvány
  + Teszteléshez használt: OpenSSL programmal generált önaláírt tanúsítvány
  + Az átadott vezióhoz ajánljuk valamelyik tanúsítványkiszolgálótól megvásárolni a használt domainhez (pl**. RetroFilmKincsek.hu**) tartozó tanúsítványt.
  + Tanúsítvány tároló mappa helye:

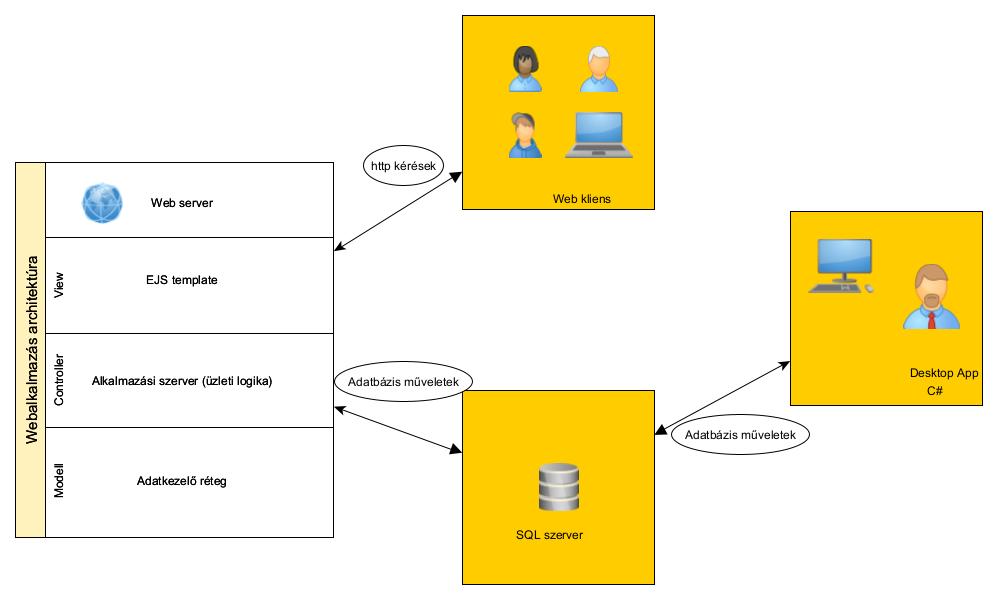
[telepítési mappa]\cert\localhost\  
\bin\bin.js file tartalmazza a használt https tanúsítvány elérhetőségét:

const options ={

key:fs.readFileSync(path.join(\_\_dirname,'../cert/localhost/localhost.decrypted.key')),

cert:fs.readFileSync(path.join(\_\_dirname,'../cert/localhost/localhost.crt'))

}



. ábra

**Rétegek**



**2**. ábra

**A Router Layer** tartalmazza az alkalmazás programozási felületének (API) útvonalait.

**A réteg felelőssége:**

Meghívja az alkalmazás szerver megfelelő metódusát és várja a válaszát. A választ küldjön a szervertől a böngészők felé.

**A modul helye:** /db/dboperation.js

**A réteg konponensei:**

* **index.js** – általános útvonalak kezelése
  + **URL :/**
    - Metódus: GET
    - render: index.ejs
    - Főoldal
  + **URL:** /filmek/:page - ***:page*** az oldalszám paramétere
    - **Metódus:** GET
    - **render:** page.ejs
    - **A filmeket listázza ki**
  + [URL: /szinesz/:nev](URL:/szinesz/nev) - :nev a színész neve.
    - Metódus: GET
    - render: actor.ejs
    - Azokat a filmeket listázza ki, amelyekben szerepelt az adott színész.
  + URL: /rendezo/:nev - :nev a rendező neve.
    - Metódus: GET
    - render: rendezo.ejs
    - Azokat a filmeket listázza ki, amelyeket rendezte az adott rendező.
  + URL: /kategoria/:nev - :nev a kategória neve.
    - Metódus: GET
    - render: kategoria.ejs
    - Azokat a filmeket listázza ki, amelyek az adott kategóriába vannak sorolva.
  + URL: /szures
    - Metódus: GET
    - render: szures.ejs
    - A felhasználó egy részletes keresőpanelt kap a képernyőjére
  + URL: /szures
    - Metódus: POST
    - render: talalatok.ejs
    - A felhasználó által megadott adatok alapján leszűri a filmek listáját majd visszatölti a felhasználónak
  + URL: /getActor/:nev – bevitt karaterek
    - Metódus: GET
    - render: actorlist.ejs
    - A szűrési oldalon, ha a felhasználó elkezdi gépelni a színész nevét a saját mezőjébe, akkor kiad egy listát, ami a további karakterek megadásával szűkül.
  + URL: /getCategory/:nev
    - Metódus: GET
    - render: categorylist.ejs
    - A szűrési oldalon, ha a felhasználó elkezdi gépelni a színész nevét a saját mezőjébe, akkor kiad egy listát, ami a további karakterek megadásával szűkül.
  + URL: /kereses
    - Metódus: POST
    - render: talalatok.ejs
    - A navigációs sávban található „Keresés” gomb hívja meg, lefut az adatbázisban található minden film minden adatán, majd amiben megtalálható az az adat amit a felhasználó megadott rendereli azokat a talalatok.ejs-ben.
  + URL: /addFavorite
    - Metódus: POST
    - redirect: adott film adatlapja
    - A filmet hozzárendeli a felhasználó kedvenc filmjei közé, majd visszaadja a film adatait.
    - Ha a felhasználó nincs bejelentkezve, a bejelentkezési oldalra irányítja át.
  + URL: /removeFavorite
    - Metódus: POST
    - redirect: adott film alaplapja
    - A filmet kiveszi a felhasználó kedvenc filmjei közé, majd visszaadja a film adatait.
  + URL :/film/:id – A film adatbázisban szereplő azonosítója
    - Metódus: GET
    - render: adatlap.ejs
    - A film adatait jeleníti meg a felhasználó számára.
* **admin.js**– adminisztrátor funkciókhoz tartozó útvonalak kezelése
  + **URL: /userlist/:page – page** az oldalszám paramétere
    - **Metódus: GET**
    - **render userlist.ejs**
    - **Az adminisztrátor megtekintheti az összes felhasználót**
* **users.js –** felhasználókezeléshez tartozó útvonalak kezelése
  + **URL: /user/kedvencek**
    - **Metódus: GET**
    - **render: kedvencek.ejs**
    - **A felhasználó kedvenceit tölti be**
  + **URL: /user/login**
    - **Metódus: GET**
    - **render: login.ejs**
    - **A bejelentkezési oldalt tölti be**
  + **URL: /user/reg**
    - **Metódus: GET**
    - **render: reg.ejs**
    - **A regisztrációs oldalt tölti be.**
  + **URL: /user/profil**
    - **Metódus: GET**
    - **render: profil.ejs**
    - **A felhasználó profiljának adatait jeleniti meg.**
  + **URL: /user/login**
    - **Metódus: POST**
    - **render: A felhasználó által legutóbb megtekintett ejs**
    - **A felhasználó bejelentkezéséhez szükséges adatokat küldi el a szerverre**
  + **URL: /user/reg**
    - **Metódus: POST**
    - **render: A felhasználó által legutóbb megtekintett ejs**
    - **A felhasználó regisztrációjához szükséges adatokat küldi el a szerverre**
  + **URL: /user/logout**
    - **Metódus: GET**
    - **render: A felhasználó által legutóbb megtekintett ejs**
    - **A felhasználót kijelentkezteti.**

**A Controller Layer** kezeli az alkalmazás üzleti logikáját. Ez azt jelenti, hogy az adatokat átalakítják vagy kiszámítják, hogy megfeleljenek az adatbázismodell követelményeinek, mielőtt elküldik őket a kiszolgálóra. A controller réteg a router layerbe lett integrálva.

**Adatkezelő réteg (Modells)**

Hozzáféréssel rendelkezik az adatbázishoz.

Az adatbáziskapcsolathoz szükséges paramétereket a **db/dbconfig.js** file tartalmazza:

const config = {

    connectionLimit: 10,

    password: 'root',

    user: 'root',

    database: 'keptar',

    host: 'localhost',

    port: 3306,

    dateStrings: true

  };

A telepítést követően a paramétereket helyi sajátosságoknak megfelelően kell beállítani.

A telepített verziónál ajánlatos egy korlátozott jogosultságokkal rendelkező felhasználó létrehozása a MySQL szerverben és ezzel a felhasználóval kell csatlakozni.

**host:** amennyiben az SQL szerver és a webkiszolgáló külön gépen kerül telepítésre, itt kell megadni a MySQL szervergép hostnevét

**user:** a MySQL szerverhez kapcsolódó felhasználó neve

**password:** MySQL user jelszava

**A réteg felelőssége:**

Az adatok lekérdezése, létrehozása, törlése vagy szerkesztése.Ez az a réteg, ahol a szerver logikájából érkező összes kérést és választ kezeljük. Az adatelérési (Data Access Layer) metódusok változókat kapnak a controller rétegtől. Ezek a változók szükségesek az SQL lekérdezésekhez. Az SQL szervertől kapott válasz alapján ígéretet (Promise) küld, amely egy objektum.

Az objektum a szerver válaszának rendelkezésre állása után kerül meghatározásra. Ha a kérés sikeres, a DAL ígéretet egy objektum oldja fel, amely visszatér a szolgáltatási réteghez, amely maga is visszatér az útválasztási réteghez. Amikor az útválasztási réteg megkapja a szolgáltatási réteg által visszaadott objektumot, az útválasztási réteg JSON formátumban küldi el az objektumot a View rétegnek.

**A modul helye:** /db/dboperation.js

A réteg Komponensei:

* SelectFilmek(pageNo)
  + Lekérdez az összes film közül azt a 18-at amelyiket az oldal most tölti be
  + paramétere: pageNo – az oldal száma
* SelectFilmekIndex
  + Lekérdezi azokat a filmeket amelyeknél a „Kezdolapon” igaz az adatbázisban.
* SelectFilmekIMDB
  + Lekérdezi az adatbázisban 20 legmagasabb IMDB ponttal rendelkező filmet
* SelectSzineszFilm(szineszNeve)
  + Lekérdezi azokat a filmeket amikben a megadott színész szerepel
  + paramétere: - szineszNeve: A keresett színész neve
* SelectRendezoFilm(rendezoNeve)
  + Lekérdezi azokat a filmeket amiket az adott rendező rendezett
  + paramétere: - rendezoNeve: A keresett rendező neve
* SelectKategoriaFilm(KategoriaNeve)
  + Lekérdezi azokat a filmeket amik az adott kategóriába vannak sorolva
  + paramétere: - KategoriaNeve: A keresett kategória neve
* SelectKategoria(kateg)
  + Lekérdezi az összes kategóriát, amiben szerepel a paraméter
  + paramétere – kateg: A felhasználó által beírt karakterlánc
* SelectSzinesz(nev)
  + Lekérdezi az összes színész nevét, amiben szerepel a karakterlánc
  + paramétere – nev: A felhasználó által beírt karakterlánc
* SearchFilm(MagyarCim,Rendezo,Ev,EredetiCim,Kategoria,Szinesz)
  + Lekérdezi a felhasználó által megadott paraméterek alapján a filmek listáját
  + paraméterei:
    - MagyarCim: A filmek magyar címe
    - Rendezo: A filmek rendezője
    - Ev: A filmek készítésének éve
    - EredetiCim: A filmek eredeti címe
    - Kategoria: A filmek kategoriaja
    - Szinesz: a filmben szereplő színészek
* SearchFilmAll(Kereses)
  + Lekérdezi azokat a filmeket amelyek tartalmazzák a paramétert, bármelyik adatukban.
  + paramétere: - Kereses: a felhasználó által megadott karakterlánc
* SelectOne(movieId)
  + Lekérdezi az egesz view-ból az adott film adatait
  + paramétere: - movieId: filmek azonosítója
* SelectActors(movieId)
  + Lekérdezi a filmhez tartozó színészek nevét a kapcsolótábla segítségével
  + paramétere: - movieId: filmek azonosítója
* SelectCategory(movieId)
  + Lekérdezi a film kategóriái a kapcsolótábla segítségével
  + paramétere: - movieId: filmek azonosítója
* SelectFavorite(user\_id, filmId)
  + Lekérdezi, hogy az adott film az adott felhasználóval szerepel-e a kedvenckapcsolas táblában
  + paramétere:
    - user\_id: A felhasználó azonosítója
    - filmId: A film azonosítója
* InsertFavorite(user\_Id, filmId)
  + Beilleszti a kedvenckapcsolas táblába a film és felhasználó azonosítók párosát
  + paraméterei:
    - user\_id: A felhasználó aonosítója
    - filmId: A film azonosítója
* DeleteFavorite(user\_id,filmId)
  + Kitörli a kedvenckapcsolas táblából a film és felhasználó azonosítók párosát
* SelectUserFavorite(user\_id)
  + Kilistázza azokat a filmeket, amelyek össze vannak kötve a felhasználóval a kedvenckapcsolas táblában
  + paramétere: - user\_id: a felhaszáló azonosítója
* InsertUser(username, password,name,email)
  + Regisztrációkor beilleszti a felhasználó adatait a user táblába.
  + paraméterei:
    - username: Felhasználónév
    - password: Jelszó, ezt az sql függvény titkosítja, és csak a titkosítottat tárolja el az adattáblában
    - name: A felhasználó teljes neve
    - email: A felhasználó email-címe
* UpdateUser(id, Username,Password,Neve,Email)
  + A regisztrált felhasználó adatait frissíti a user táblában
  + paraméterei:
    - id: a felhasználó azonosítója
    - Username: Felhasználónév
    - Password: Jelszó, ezt az sql függvény titkosítja, és csak a titkosítottat. tárolja el az adattáblában
    - Name: A felhasználó teljes neve
    - Email: A felhasználó email-címe
* VerifyUser(username password)
  + Ellenőrzi hogy a felhasználó helyes felhasználónév és jelszó párost adott-e meg.
* SelectUser(pageNo)
  + A user táblát jeleníti meg 50-essével
* SelectOneUser(user\_id)
  + A felhasználó adatait kérdezi le az azonosító alapján.

**View réteg:**

A html oldal előállításáért (render) **ejs template engine** a felelős.

A modul helye: **/views** mappa

A réteg felelőssége:

A JSON formátumban megkapott paraméterek és az ejs file-ban meghatározott sablon alapján előállítja a kimeneti HTML formátumot. A generált html oldal szövegét visszaadja a útválasztási rétegnek.

A réteg Komponensei:

A filmeket megjelenítő EJS:

* filmek.ejs: - Megmutatja az összes adatbázisban szereplő filmet
* card.ejs: - A filmek kártyáit hozza létre, ez van meghívva a filmek.ejs-ben

A szűréshez használt EJS-ek:

* szures.ejs: - Betölt egy form-ot, amit a felhasználó tölthet ki értelemszerűen.
* actorlist.ejs: A szűrő form-on a színészek nevének betöltését segíti
* categorylist: - A szűrő form-on a kategóriák nevének betöltését segíti
* talalatok.ejs: - A szűrés eredményeként kapott filmetek tölti be.
* actor.ejs: - A filmek adatlapján található linken keresztül a színészek alapján szűrt filmek megjelenítése
* kategoria.ejs: - A filmek adatlapján található linken keresztül a kategória alapján szűrt filmek megjelenítése
* rendezo.ejs: - A filmek adatlapján található linken keresztül a rendező alapján szűrt filmek megjelenítése

Felhasználói EJS-ek:

* reg.ejs: - Regisztrációs form megjelnítése
* login.ejs: - Bejelentkezési form megjelnítése
* kedvencek.ejs: - A bejelentkezett felhasználó által kedvelt filmek kártyái
* profil.ejs: - A bejelentkezett felhasználó adatlapját megjeleníti

Egyéb EJS-ek:

* error.ejs: A hiba kiírására használt ejs
* index.ejs: A főoldal
* navbar.ejs: A navigációs sávot tartalmazza, ez van meghívva minden egyéb oldalon
* userlist.ejs: Az adminisztrátori jogosultsággal rendelkező felhasználók szárára megjeleníthető a regisztrált felhasználók.

## Választott technológiák

Miért ezt választottam?

* Web-es technológiát választottam, hogy egyszerre akár több száz felhasználót ki lehessen szolgálni, és ezen előnyeit szeretném kihasználni.
* Desktopot, korlátozott számú felhasználó használja az adminisztrátori funkciókat. Egyszerre többen is hozzáférhetnek az adatbázishoz, viszont ez elegendő ha a lokális hálózatról elérhető.
* Az aszinkron programozási technológiát választottam, hogy több száz felhasználó esetén is egyidejűleg ki tudja szolgálni a felhasználókat. Valamint az interneten keresztül bárhol el lehessen érni a szolgáltatást.

**Desktop applikáció:**

* Fejlesztői környezet:
  + Visual Studio
  + MySQL for Visual Studio
* Programnyelv: C#
* Operációs rendszer: Windows
* Keretrendszer: .NET framework 4.8 Advanced Services
* ADO.NET programkönyvtár
* Connector/NET for MySQL 8.0.28 X86

**Web-es app:**

* Fejlesztői környezet:
  + NodeJS
* Programnyelv: Javascript
* Operációs rendszer: Windows / Linux

Könyvtárak:

* + bootstrap: 5.2.3
    - A Bootstrap keretrendszer nyílt forráskódú és ingyenes, segítségével a weboldalunk kinézetét egyszerűen és gyorsan elkészíthetjük. Olyan előre elkészített stílusokat, elrendezések tartalmaz amelyeket saját CSS vagy JavaScript kód írása nélkül használhatunk.
  + cookie-parser: 1.4.4
    - A Cookie-parser egy olyan modul, amely kezeli a weboldalon használt sütiket. A sütik olyan kisméretű adatfájlok, amelyek lehetővé teszik a weboldalnak, hogy ha felhasználót bejelentkezve hagyják az eszközén, azért mert az eszközön tárolja az ehhez szükséges adatokat.
  + corps: 0.1.0
  + cors: 2.8.5
    - Cross-Origin Resource Sharing (CORS) egy biztonsági mechanizmus, amely a webes támadásoktól védi meg a felhasználót. Egy weboldal akkor használja, ha más oldalakról is szeretne információt betölteni.
  + debug: 2.6.9
    - A Debug segít a fejlesztőknek, a hibakeresést végez az alkalmazásban. Segítségével a fejlesztők könnyen és rendszeresen megtalálhatják az általuk vétett hibákat a kódban.
  + ejs: 3.1.8
    - Az EJS (Embedded JavaScript) olyan sablonrendszer, amely lehetővé teszi az adatok dinamikus változtatását a weboldalon. Segítségével az oldalon dinamikusan lehet megjeleníteni, vagy változtatni az adatokat. Az EJS-nek köszönhetően lehet az oldal a felhasználó számára személyre szabott.
  + express: 4.16.1
    - Az Express egy olyan keretrendszer, amely rengeteg előre elkészített funkciót és modult tartalmaz. Ezek segítségével a fejlesztők egyszerűen létrehozhatnak új funkciókat saját alkalmazásukban.
  + express-session: 1.17.3
    - Az Express-session egy olyan modul, amely segít az alkalmazásoknak kezelni a felhasználói munkameneteket. Általában a felhasználói információk tárolására használják, például a bejelentkezés után elmenti a felhasználó adatait, hogy később az oldal könnyebben elérhesse azt.
  + fs: 0.0.1-security
    - Az fs a „file system” kifejezésre utal, ez a modul lehetővé teszi az alkalmazásnak, hogy fájlokhoz férjen hozzá a számítógépről.
  + http-errors: 1.6.3
    - Az http-errors modul egy Node.js könyvtár, amely lehetővé teszi a fejlesztők számára, hogy HTTP hibakódokat használjanak kivételek kezeléséhez a Node.js alkalmazásaikban.Ez azt jelenti, hogy az http-errors modul segítségével könnyedén dobhatunk kivételt a különböző HTTP hibakódokkal, például a 404-es vagy a 500-as hibakóddal kapcsolatosan.
  + https: 1.0.0
    - Az HTTPS (HyperText Transfer Protocol Secure) az HTTP protokoll biztonságosabb változata, amely titkosítja az adatokat a szerver és a kliens közötti kommunikáció során. Az HTTPS az SSL (Secure Sockets Layer) vagy a TLS (Transport Layer Security) protokollt használja a titkosított kommunikáció biztosításához. Az HTTPS titkosítja az adatokat, így a kommunikáció biztonságosabbá és védettebbé válik. Az HTTPS használata során a kliens és a szerver közötti kommunikáció titkosított csatornán keresztül történik, és csak a kommunikáló felek képesek olvasni az adatokat.
  + jquery: 3.6.3
    - A jQuery egy JavaScript könyvtár, amely megkönnyíti a webfejlesztőknek a dinamikus weboldalak fejlesztését. A jQuery segítségével a webfejlesztők egyszerűbben manipulálhatják a HTML elemeket és a CSS stílusokat, egyszerűbben hozhatnak létre animációkat, kezelhetik az eseményeket (például a kattintásokat és a beviteli mezők bevitelét), és könnyedén kommunikálhatnak a szerverrel AJAX segítségével.
  + morgan: 1.9.1
    - A morgan modul lehetővé teszi, hogy a fejlesztők az alkalmazás fejlesztése alatt naplózási információkat gyűjtsenek, mint például a beérkező kérések, azok státusza vagy időtartama.
  + multer: 1.4.5
    - A multer egy olyan eszköz, amely lehetővé teszi, hogy a felhasználó képeket, videókat vagy egyéb fájlokat töltsön fel a szerverre. A segítségével könnyedén kezelhető a fájlok mérete, típusa vagy a kiterjesztése, illetve a felmásolt fájl helyének megadása.
  + mysql: 2.18.1
    - A mysql modul lehetővé teszi a MYSQL adatbázisokkal való kommunikációt.
  + mysql2: 3.1.0
    - A mysql2 egy Node.js modul, amelyet MySQL adatbázisokkal való kommunikációhoz használnak. A mysql2 az eredeti mysql modulhoz képest hatékonyabb és több funkcionalitást kínál.A mysql2 a Node.js áramlás (stream) API-ját használja, ami hatékonyabb adatkezelést tesz lehetővé, különösen nagy adatmennyiségek esetén.
  + nodemon: 2.0.20
    - A nodemon egy Node.js alkalmazás, amely figyeli az alkalmazás fájljait, és újraindítja az alkalmazást minden alkalommal, amikor az egyik fájl módosul. A nodemon konfigurálható, és számos opciót és beállítást kínál, például a fájlok figyelési mappájának meghatározását, a figyelő ciklusának a beállítását, és az alkalmazás újraindításának módját.
  + path: 0.12.7
    - Az path modul hasznos segédfüggvényeket kínál a fájlok és mappák elérési útvonalainak kezeléséhez, fájlnevek és kiterjesztések feldolgozásához, elérési utak összekapcsolásához és ellenőrzéséhez, és sok más hasznos funkciót kínál.

## Függőségek

1. MySQL Server 8.0.31 adatbázis szerver
2. Hardver követelmény:
   1. Processzor: 1 GHz vagy gyorsabb processzor.
   2. RAM: 1 GB a 32-bites rendszerhez vagy 2 GB a 64-bites rendszerhez
   3. Háttértároló: 16 GB a 32-bites rendszerhez vagy 20 GB a 64-bites rendszerhez.
   4. Videókártya: DirectX 9 vagy újabb WDDM 1.0 driverrel.
   5. Megjelenítő: 1024 x 600 vagy nagyobb felbontás.
3. **Desktop app:**
   1. Operációs rendszer: Windows 10
   2. Keretrendszer: .NET framework 4.8 Advanced Services
4. **Web App:**
   1. Windows / Linux
   2. nodeJS
   3. package.json állományban lévő függőségek (telepítés: npm install)
   4. tanúsítvány, amennyiben van saját

# Modulok

A rendszer két modulból áll:

* Desktop alkalmazás
* Webkiszolgáló app

## Áttekintés

A szoftver architektúrája az alábbi komponensekre bomlik:

* Desktop alkalmazás
* Webkiszolgáló app

## Modulok részletei

Desktop alkalmazás – adminisztrációs modul

* Törzsadatok
  + Filmek
  + Besorolás
  + Kategóriák
  + Év
  + Hossz
  + Kedvenckapcsolás
  + Rendező
  + Színész
  + Színészkapcsoló
  + Felhasználók
* Kilépés

**Webes alakalmazás**

* Felhasználókezelés
  + Regisztráció
    - A felhasználók regisztrációjának kezelése.
  + Bejelentkezés
    - A regisztrált felhasználók bejelentkezésének megvalósítása.
  + Profil
    - A bejelentkezett felhasználók adatainak megjelenítése.
  + Kijelentkezés
    - A bejelentkezett felhasználók kiléptetése a weboldalról.
* Kereső
  + A felhasználó megkötések nélkül kereshet.
* Filmek megjelenítése
  + Az adatbázisban található filmek megjelenítése
* Szűrés
  + A felhasználó kötöttebben kell, hogy keressen, viszont több információt is megadhat egyszerre
* Kedvencek
  + A felhasználó kedvencekhez adhatja az általa kedvelt filmeket így ezeket könnyebben megtalálhatja

# Perzisztencia

Adattárolás:

* MySQL adatbázis
* Képek tárolása a fájlrendszer mappájában

## A tárolás választott módja

MySQL:

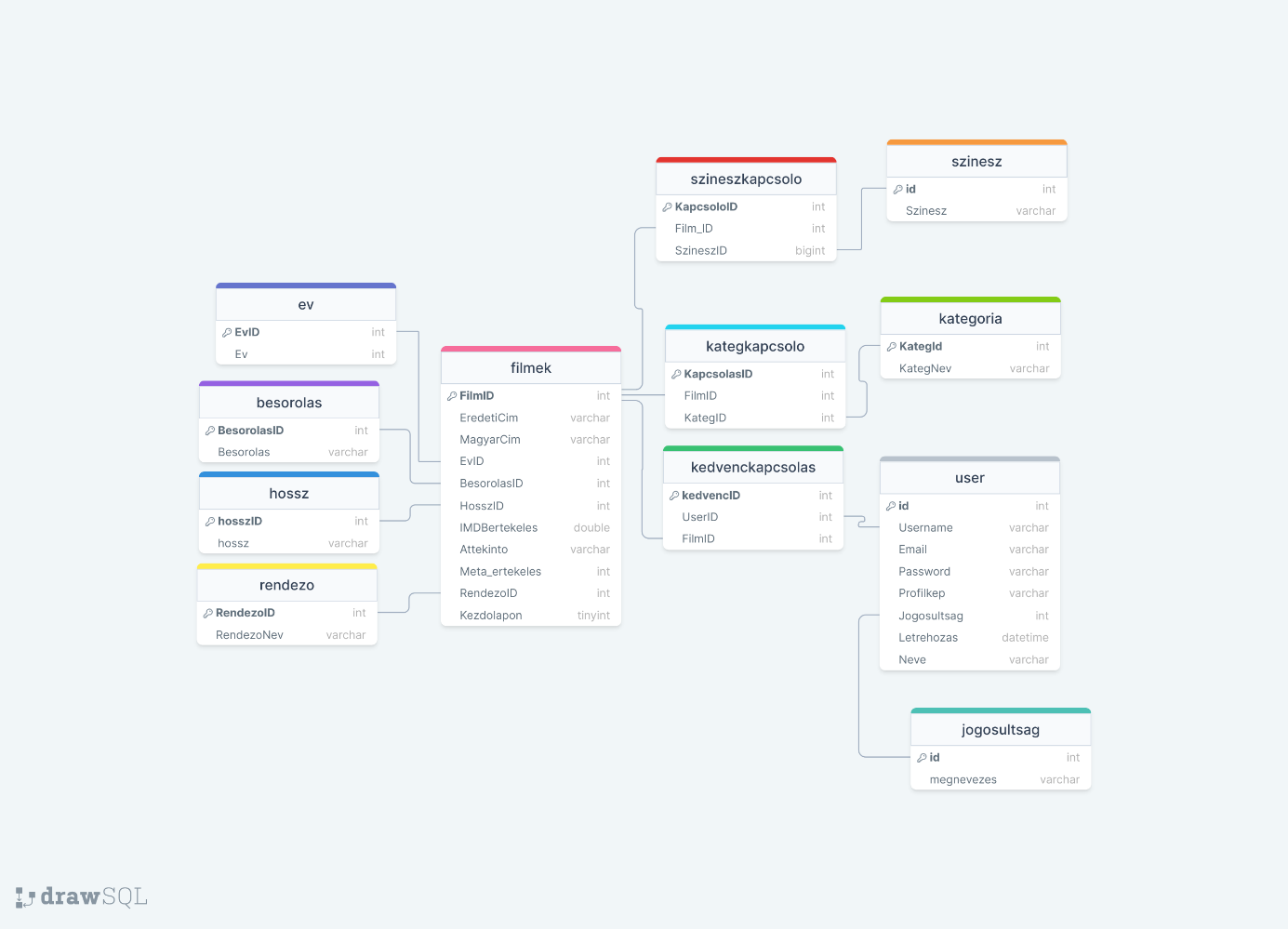
MySQL egy relációs adatbázis-kezelő rendszer (RDBMS), amelyet adatok tárolására és lekérdezésére használnak. Gyakran használják webalkalmazásokban az adatok kezeléséhez és dinamikus tartalom biztosításához.

Fájlok, képek tárolása:

* Az alkalmazás erre a célra kijelölt mappákban történik:
  + Felhasználói profil képei: [alkalmazás mappája]\public\profil
  + A filmek képei: [alkalmazás mappája]\public\images

## Koncepcionális adatmodell

* Általános
  + filmek
    - FilmID INT NOT NULL PRIMARY KEY,
    - EredetiCim VARCHAR(150) DEFAULT NULL,
    - MagyarCim VARCHAR(150) DEFAULT NULL,
    - EvID INT DEFAULT NULL,
    - BesorolasID INT DEFAULT NULL,
    - HosszID INT DEFAULT NULL,
    - IMDBertekeles DOUBLE DEFAULT NULL,
    - Attekinto VARCHAR(2000) DEFAULT NULL,
    - Meta\_ertekeles INT DEFAULT NULL,
    - RendezoID INT DEFAULT NULL,
    - Kezdolapon TINYINT(1) DEFAULT NULL,
  + besorolas
    - BesorolasID INT NOT NULL PRIMARY KEY,
    - Besorolas VARCHAR(40) DEFAULT NULL,
  + ev
    - EvID INT NOT NULL PRIMARY KEY,
    - Ev INT DEFAULT NULL,
  + hossz
    - hosszID INT NOT NULL PRIMARY KEY,
    - hossz VARCHAR(40) DEFAULT NULL,
  + kategoria
    - KategId INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,
    - KategNev VARCHAR(255) DEFAULT NULL,
  + rendezo
    - RendezoID INT NOT NULL PRIMARY KEY,
    - RendezoNev VARCHAR(40) DEFAULT NULL,
  + szinesz
    - id INT NOT NULL PRIMARY KEY,
    - Szinesz VARCHAR(40) DEFAULT NULL,
* Kapcsoló táblák
  + kategkapcsolo
    - KapcsolasID INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,
    - FilmID INT NOT NULL,
    - KategID INT NOT NULL,
  + kedvenckapcsolas
    - kedvencID INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,
    - UserID INT NOT NULL,
    - FilmID INT NOT NULL,
  + szineszkapcsolo
    - KapcsoloID INT NOT NULL PRIMARY KEY,
    - Film\_id INT DEFAULT NULL,
    - SzineszID INT DEFAULT NULL,
* Felhasználó táblák
  + jogosultsag
    - id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,
    - megnevezes VARCHAR(255) NOT NULL,
  + user
    - id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,
    - Username VARCHAR(255) NOT NULL,
    - Email VARCHAR(50) NOT NULL,
    - Password VARCHAR(255) NOT NULL,
    - Profilkep VARCHAR(255) DEFAULT 'profile1.jpg',
    - Jogosultsag INT NOT NULL DEFAULT 1,
    - Letrehozas DATETIME DEFAULT (now()),
    - Neve VARCHAR(255) NOT NULL,



. ábra

Adatkapcsolatok:

* filmek
  + RendezoID = rendezo.RendezoID
  + EvID = ev. EvID
  + BesorolasID = besorolas.BesorolasID
  + HosszI = hossz.hosszID
  + FilmID = kategkapcsolo.FilmID
  + FilmID = szineszkapcsolo.Film\_id
* kategkapcsolo
  + kategoria.KategId = kategkapcsolo.KategID
* szineszkapcsolo
  + szinesz.id = szineszkapcsolo.SzineszID
* kedvenckapcsolo
  + FilmID = filmek.FilmID
  + UserID = user.id
* user
  + Jogosultsag = jogosultsag.id

# Folyamatok

Filmek bemutatása: A filmek a „Filmek” fül alatt töltődnek be. Elsőre csak 18 film jelenik meg, viszont, lefelé görgetéssel töltődik be a többi szintén ilyen csoportokban. A filmek részletes adatát a címre rákattintva a film adatlapján találhatjuk

Regisztráció: A „Profil” fül alatt található egy „Regisztráció” gomb, ami betölti a regisztrációs oldalt. Itt megjelenik egy form, ahol ki kell tölteni mind az 5 beviteli mezőt. Ezek közül csak 4 kerül elmentésre azután, hogy a program ellenőrizte, hogy a jelszavak megegyeznek-e.

Bejelentkezés: A „Profil” fül alatt megtalálható a „Bejelentkezés” gomb, ami betölti a bejelentkezési oldalt. Itt egy form található, ami kéri hogy a felhasználó írja be a felhasználónevét és jelszavát amivel regisztrált. Amennyiben ez sikertelen, a felhasználót értesíti a program. Ha egy felhasználó nem regisztrált, van egy „[Még nem regisztrált?](https://localhost/user/reg)" szöveg sárgával, ez átirányítja a regisztrációs oldalra.

Profil: A felhasználó megtekintheti az adatait belépés után a „Profil” fül alatt megtalálható egy gomb, rajta a felhasználóneve és a profilképe, rákattintva egy a felhasználó minden adatát, kivéve a jelszavát, tartalmazó mező jelenik meg. Itt lehetősége van az adatainak a megváltoztatására.

Kedvencek: A felhasználó hozzáadhat a fiókjához általa kedvelt filmeket, ezeket a program egy táblában eltárolja. A „Kedvencek” fül alatt megjelenítheti, illetve a film adatlapján is megtalálható.

Kijelentkezés: A felhasználó a „Profil” fül alatt talál egy „Kijelentkezés” gombot, amire rákattintva törli a jelenlegi session-t így csak az alap funkciókat éri el.

Ábrajegyzék

[1. ábra 5](#_Toc132352682)

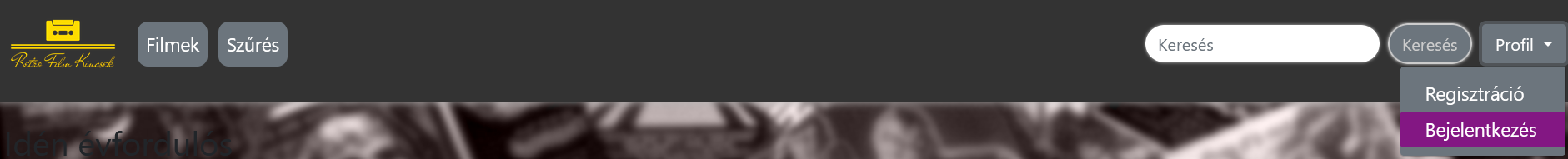
[**2**. ábra 6](#_Toc132352683)

[3. ábra 20](#_Toc132352684)

Felhasználói dokumentáció

# Általános felhasználó:

A weboldal megnyitásakor a felhasználó egy navigációs felületet láthat a lap tetején. Ez a terület ugyanaz marad addig, amíg be nem jelentkezik, vagy nem regisztrál. Itt található bal oldalon a logónk erre rákattintva, mindig a főoldalt kapja vissza felhasználó. Mellette található egy „Filmek” gomb, megnyomása betölti az oldalon található összes filmet. A következő gomb egy szűrő gomb, ami átirányítja a felhasználót egy oldalra, ahol a filmek több adata alapján is tud keresni. Ezen kívül található még egy kereső sáv is a jobb oldalon. Ebbe a felhasználó bele tudja írni a keresett film, eredeti címét, magyar címét, korhatár besorolását, egyik színészét vagy kategóriáját, rendezőjét vagy esetleg valamilyen részletet, amit tartalmazhat a leírás, viszont ez minden adatát megvizsgálja egy filmnek, így nem olyan pontos a keresés. Ezen kívül már csak egy lenyíló fül található, „Profil” felirattal, itt lehet bejelentkezni vagy regisztrálni amennyiben ez még nem történt meg

.

A főoldalon két sorban találhatóak filmek. A sorok fölött jelzi az oldal, hogy melyik sáv miért érdekes. Az elsőben találhatóak azok a filmek, amik idén ünnepli gyártásuk évfordulóját a filmek között 10-20-30 éves és még régebbi filmek jelennek meg. A második sáv pedig azt a 20 filmet tartalmazza, amiknek a legnagyobb az Metacritic pontja, amit kritikusok általi értékelés után kapja. Az oldalon bármelyik filmnek a címére rákattintva, betölti a film adatlapját. A sávokon található 2-2 nyíllal ellátott gomb. A gombok a nyíl irányába mozgatja a kártyákat egyszerre egyet, az adott gomb csak a saját kártyáit mozgatja.

Az oldal ami betölti, az összes filmet, a felhasználó eszközének mérete alapján tölti be a filmeket egy sorokba, minél kisebb a kijelző annál kevesebbet. A oldalon lefelé görgetve találhatjuk meg a filmek kártyáját, a kártyán megtalálható a film eredeti plakátja, illetve a címe. Itt is ha rákattint a felhasználó a film címére, akkor az oldal betölt az adatlapot.

A „Szűrés” gomb lenyomása után, betölt az oldal egy olyan ablakot ahol, a filmek adataira részletesen rá lehet keresni, az adott mező kitöltésével. Minden mező szürkével jelzi, hogy milyen információt kell oda bevinni, és a felhasználó dönthet, hogy mennyi információt szeretne megadni a keresett filmről. Az év mezőbe csak számokat lehet megadni, azokat is csak akkor ha az oldalon található olyan film ami abban az évben készült. A színészt és a kategóriát ki kell választani egy listából, viszont ha a felhasználó elkezdi beírni az adatot, akkor a listában már csak azok az elemek jelennek meg amik tartalmazzák a felhasználó által megadott részletet. Amennyiben kiválasztott egy kategóriát vagy színészt az adott mezőből megjelenik még egy, a kategóriából így maximum háromra, a színészek közül maximum négyre lehet szűrni. Amennyiben a felhasználó kitöltötte a számára ismert mezőket, a „Szűrés” gombra, ami az adatokkal megegyező szürke mezőben található, megjelennek a megadott feltételeknek megfelelő filmek.

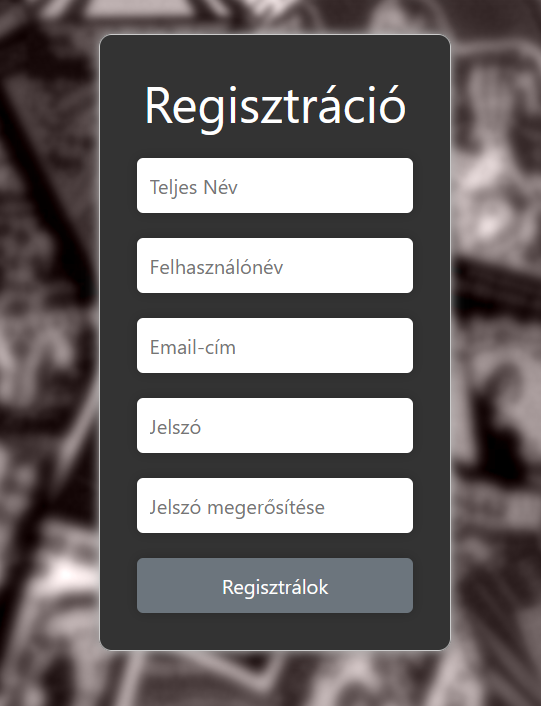
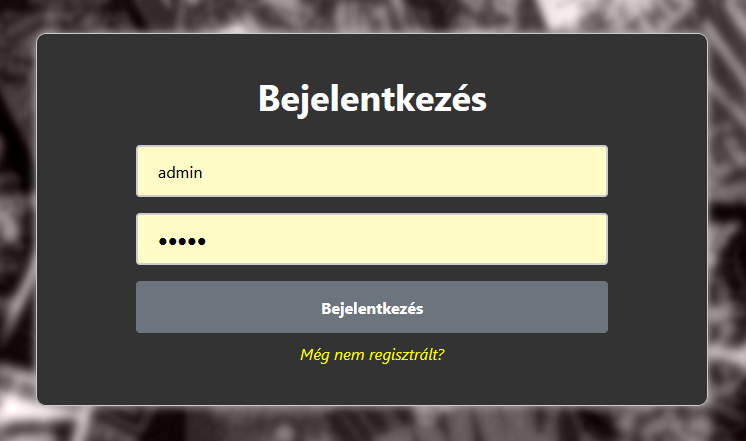


A navigációs sávon való „Keresés” és a szűrési oldalon a gomb megnyomását követően a megadott információk alapján megjelennek a filmek, amelyekre igazak a feltételek, olyan külsővel, ahogy azt az összes filmeket megjelenítéskor láthat a felhasználó. Itt is igaz, hogy ha a felhasználó rákattint a film címére, akkor betölti az adatlapját a filmnek.



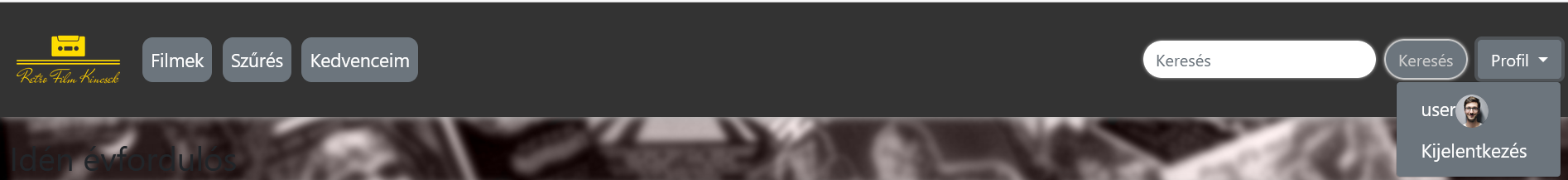
A filmek adatlapja megmutatja, az adott film magyar címét. Ismét megtalálható a film eredeti plakátja. A plakát mellett a részletes információkat láthatja a felhasználó, az eredeti címét, a műfajait, a kiadásának évét, a 2022 szeptemberi IMDB pontját mutatja, a rendező nevét, 4 színészt, illetve egy rövid leírást. A címen kívül ami ezen az oldalon sárgával jelenik meg, arra a felhasználó rákattinthat, és az adott kritériumnak megfelelően, megjelennek a filmek. A filmek adatai felett megtalálható egy sárga csillag aminek csak a körvonala a sárga, ha erre rákattintunk automatikusan átvisz az oldal a bejelentkezési oldalra. Később ennek a gombnak más feladata lesz.



A „Bejelentkezés” és a „Regisztráció” gombra kattintva hasonló kinézetű oldal jelenik meg. Itt ki kell tölteni a mezőket, minden mező jelzi, hogy mit kell a felhasználónak beleírnia, majd a dobozhoz tartozó gombot megnyomva belépteti, vagy regisztrálja és belépteti az oldalra. Az oldal mindig felkínálja a „Még nem regisztrált?” gombot, amire ha rákattint a felhasználó átirányítja a regisztrációs felületre.

# Bejelentkezett felhasználó:

A felhasználónak aki bejelentkezett a navigációs sáv, megváltozik annyival, hogy a szűrés gomb mellett megjelenik egy „Kedvenceim” gomb, ami betölti a felhasználó által a kedvencekhez adott filmek listáját, olyan külsővel mint az összes film megjelenítésekor, ahhoz hogy hozzá tudjunk adni egy filmet a kedvenceink közé, a korábban már említett sárga csillagot tudjuk használni. Amennyiben a csillagnak továbbra is csak a körvonalát látjuk, nincs bent a kedvenceink között, ilyenkor ha rákattintunk hozzáadja. Viszont ha a csillag teli sárga, akkor azt jelzi, hogy a filmet már korábban hozzáadtuk, de egy kattintással rá ismét ki tudjuk venni onnan. A profil nevű legördülő lista lenyitásával, két gomb jelenik meg, az egyik a felhasználó, felhasználónevét illetve profilképét jeleníti meg, a másik pedig egy „Kijelentkezés” gomb. A felső gombra kattintva a felhasználó profiljának adatai jelennek meg. A „Kijelentkezés” gombra kattintva pedig kilépteti a felhasználót



A felhasználó adatlapja hasonlóan külsővel rendelkezik mint a filmek adatlapja. A felhasználó nevét sárgával jeleníti meg a mező tetején, alatta a profilképét látja. A kép mellett megjelenik a felhasználónév, az email cím, a neve és hogy mikor hozta létre a fiókot. Alul sárgával pedig egy „Adatok megváltoztatása” gomb található, amire rákattintva betölt egy a regisztrációhoz hasonló oldal, itt is a mezők jelzik, hogy hova kell írni azt amit meg szeretnénk változtatni, ha beírta azokat az adatokat amiket szeretne megváltoztatni a gomb megnyomásával véglegesítheti a változtatásait, és ismét a profilját látja a már megváltoztatott adatokkal.

