

BGéSzC

1111 Budapest

# Záró dolgozat

*watch.it*

Konzulens tanár:

Lipák Tibor

Készítette:

Bakos Bence, Dózsa Dávid, Breznyánszky Dávid

## Tartalom

1	Bevezetés .....	3
1.1	A felhasznált ismeretek .....	3
1.2	A felhasznált szoftverek .....	3
2	Felhasználói dokumentáció.....	4
2.1	A program általános specifikációja.....	4
2.2	Rendszerkövetelmények .....	4
2.3	A program telepítése.....	6
2.4	A program használatának a részletes leírása .....	7
3	Fejlesztői dokumentáció .....	23
3.1	Az alkalmazott fejlesztői eszközök .....	23
3.1.1	Adatmodell leírása .....	23
3.3	Részletes feladat-specifikáció, algoritmusok.....	25
3.4	Tesztelési dokumentáció .....	47
4	Összefoglalás .....	50
4.1	Önértékelés .....	50
4.2	Továbbfejlesztési lehetőségek .....	50
5	Felhasznált irodalom .....	51

# 1 Bevezetés

## **1.1 A felhasznált ismeretek**

Az oldalhoz az alábbi ismereteinket használtuk fel:

- HTML/CSS
- JavaScript/JSON
- PHPMailer
- MySQL/PhP
- GitBash

## **1.2 A felhasznált szoftverek**

Az oldal működéséhez hozzájáruló felhasznált szoftverek az alábbiak:

- XAMPP
- Microsoft Visual Studio Code
- GitBash Terminal
- phpMyAdmin
- Preferált böngésző

## 2 Felhasználói dokumentáció

### **2.1 A program általános specifikációja**

A weboldal elsődleges funkciója a filmböngészés. A watch.it weboldalon filmkritikát fogalmazhatunk meg több felhasználó között, értékelhetjük a filmeket/sorozatokat, amit egy összesített értékláncon keresztül a felhasználók átlag véleményét fejezi ki. A weboldalon szintén megtekinthetjük egy film vagy sorozat előzetesét (úgynevezett 'trailer'-t), ahol egy pár perces összefoglalót láthatunk a weboldalon feltüntetett filmekről és sorozatokról.

Az oldalon szert tehetünk egy profil regisztrálására. Lehetőségünk van különböző műfajok szerint keresni, szűrőket alkalmazni az ajánlásokhoz, valamint a főoldalon értesülni tudunk az újonnan megjelent filmekről és sorozatokról. A közösségi funkciók révén más film és sorozat kedvelők véleményeit is támogatathatjuk vagy nem értésünket kifejezhetjük egy tetszés, vagy nem tetszés gomb megnyomásával az adott kommenten mindezt anonim módon.

### **2.2 Rendszerkövetelmények**

#### *2.2.1 Hardver követelmények*

##### **Minimális rendszerkövetelmények:**

- **Processzor:** Intel Core i3-8100 / AMD Ryzen 3 1200
- **Memória:** 4 GB RAM
- **Tárolóhely:** 5 GB szabad tárhely HDD-n vagy SSD-n
- **Grafikus kártya:** Integrált grafika (Intel UHD Graphics 620 vagy AMD Radeon Vega 3)
- **Internetkapcsolat:** Minimum 5 Mbps
- **Kijelző felbontás:** 1280x720

### Ajánlott rendszerkövetelmények:

- **Processzor:** Intel Core i5-10400 / AMD Ryzen 5 3600
- **Memória:** 8 GB RAM
- **Tárolóhely:** 10 GB szabad tárhely SSD-n
- **Grafikus kártya:** NVIDIA GeForce GTX 1650 / AMD Radeon RX 5500 XT
- **Internetkapcsolat:** Minimum 25 Mbps stabil kapcsolat
- **Kijelző felbontás:** 1920x1080
- **Szükséges szoftverek:** Legújabb böngésző verziók, WebRTC támogatás, JavaScript engedélyezve, Adobe Reader, WebGL támogatás

### 2.2.2 Szoftver követelmények

#### Szoftver követelmények a Watch.IT oldal futtatásához/megtekintéséhez szerver oldalról:

- **Operációs rendszer:** Windows 10/11 (64-bit) / Linux Ubuntu 20.04 / macOS 10.15
- **Böngésző: (legalább)** Google Chrome 90+, Mozilla Firefox 85+, Microsoft Edge 90+, Apple Safari
- **Szükséges szoftverek:** JavaScript engedélyezett böngésző, WebRTC támogatás, Adobe Reader (PDF megtekintéshez)

#### Szoftver követelmények a Watch.IT oldal futtatásához/megtekintéséhez lokális/saját hálózatról:

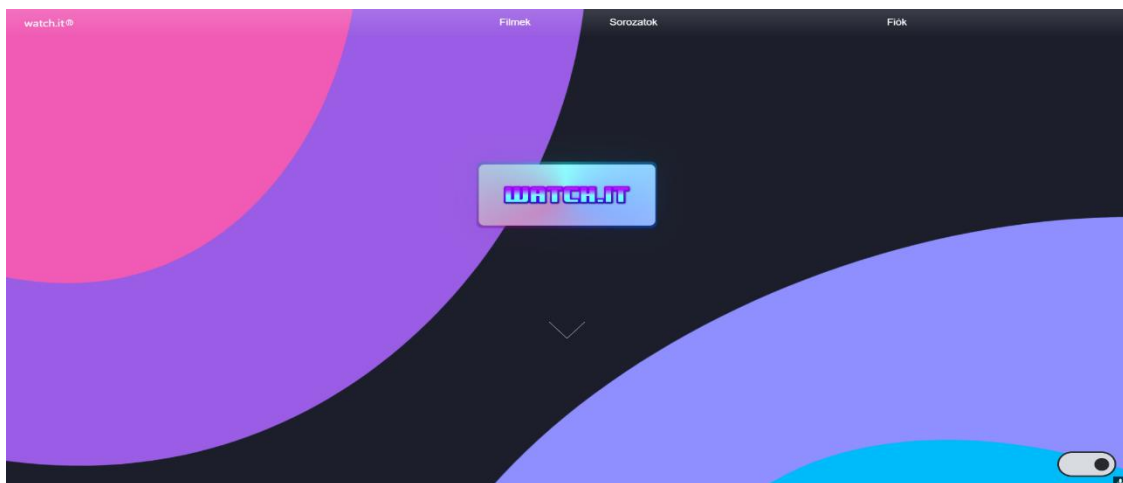
- **Operációs rendszer:** Windows 10/11 (64-bit) / Linux Ubuntu 20.04 / macOS 10.15
- **Böngésző: (legalább)** Google Chrome 90+, Mozilla Firefox 85+, Microsoft Edge 90+, Apple Safari
- **Szükséges szoftverek:** JavaScript engedélyezett böngésző, WebRTC támogatás, Adobe Reader (PDF megtekintéshez), XAMPP Control Panel, phpMyAdmin, Visual Studio Code (részletesebb betekintésért)

## 2.3 A program telepítése

Ahhoz, hogy meg tudjuk tekinteni a weboldalt, nincs más dolgunk, mint hogy elindítsuk a számítógépünkön vagy okoseszközünkön a kedvenc vagy éppen alapértelmezett internetes böngészőnket — ez lehet például a Google Chrome, a Mozilla Firefox, a Microsoft Edge vagy akár a Safari is. Miután megnyitottuk a böngészőt, a tetején található hosszúkás mezőbe, amit címsornak is nevezünk, be kell írunk a következő domain címet: [abakos.hu](http://abakos.hu)



Fontos, hogy pontosan írjuk be, szóközök és elgépelések nélkül! Ha ezzel megvagyunk, akkor nyomjuk le az **Enter** billentyűt a billentyűzeten, vagy kattintsunk rá a böngésző jobb oldalán található nagyító ikonra, ami a keresést vagy a betöltést indítja el, ez böngészőtől függően egy picit eltérhet. Ha minden rendben ment, és a cím helyesen lett beírva, akkor néhány pillanat múlva betöltődik a weboldal, és meg is jelenik előttünk a tartalma.



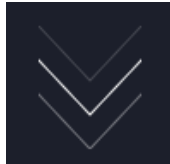
Ez a kezdőlap, más néven a **Főoldal**, ahonnan aztán tovább tudunk navigálni az oldal többi részére, például megnézhetjük az oldal alján a legújabb filmeket és sorozatokat, vagy böngészhetjük magunknak ezeket, illetve megnézhetjük egyéb „social” platformjait az oldalnak.

## 2.4 A program használatának a részletes leírása

### Főoldal

Amint a felhasználó megérkezik a főoldalunkra, egyből több mindenre is felbukkanhat. A legfelső sorban (navigációs bárban) láthatja a „Filmek”, „Sorozatok”, illetve a „Fiók” menüket.

Látják a főoldal animációt is, ami egyben a logónk is. Az alatt látható egy lefele mutató nyíl, amely arra szögezi a tekintélyünket, hogy a főoldalnak még nincs itt vége. Amint a felhasználó megnyomja az animált lefele nyilat,

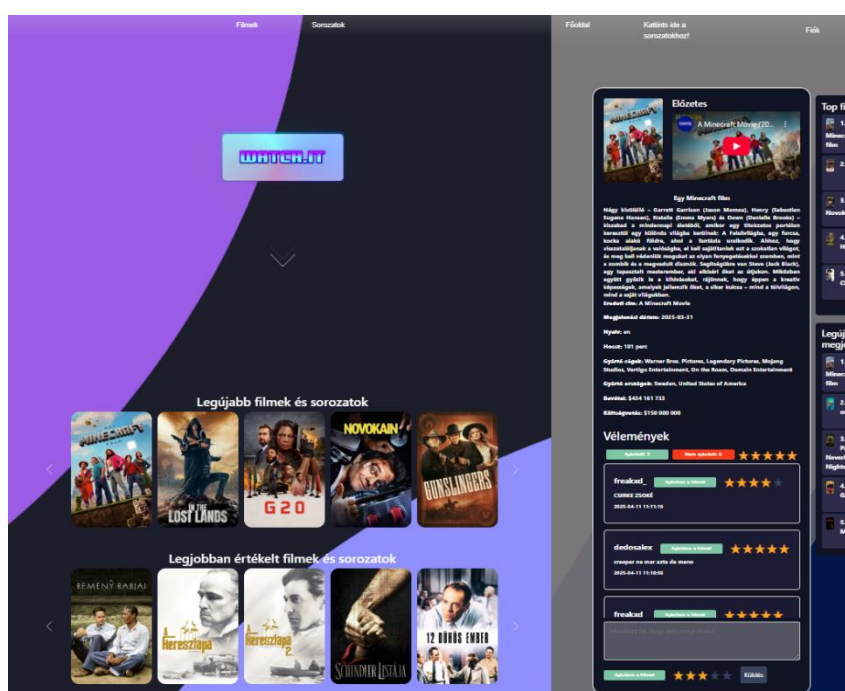


akkor leviszi őt a "Legújabb filmek és sorozatokhoz", illetve a „Legjobban értékelt filmekhez és sorozatokhoz”. A film és sorozat posztterek automatikusan változnak megadott időnként. Hogyha a felhasználó úgy dönt, akkor rá is tudja a posztterekre húzni a kurzort amennyiben számítógépen használja a weboldalt. Ekkor, úgynevezetten kiugró animációt fog látni, amiben az az adott film vagy sorozat látható, amelyre ráhúzta a kurzorát. A többi elem elhomályosul, hogy jobban tudjon az éppen kiválasztott filmre vagy adott sorozat poszterére fókuszálni.



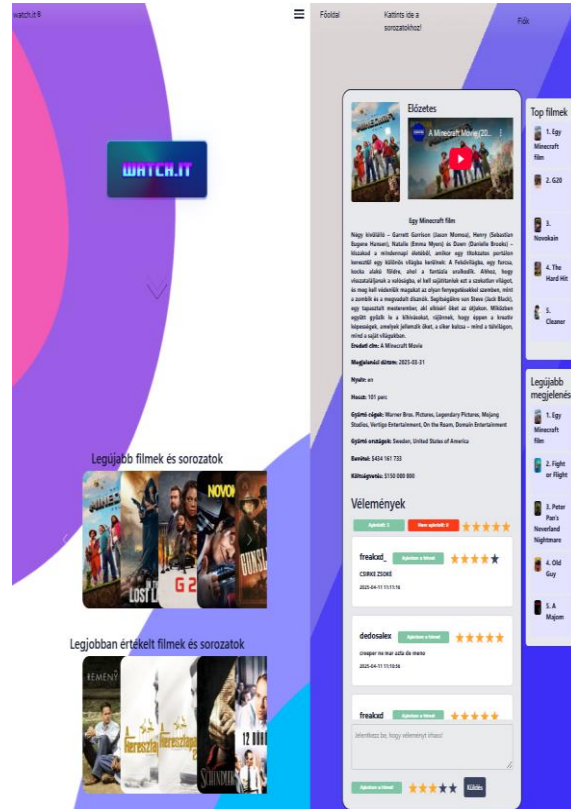
A főoldalon jobb alul még megtalálható a "Fekete-Fehér mód", amely személyes preferencia szerint be tudja a felhasználó állítani, hogy éppen milyen módban szeretné a weboldalunkat felhasználni.

### Fekete/Éjszakai mód

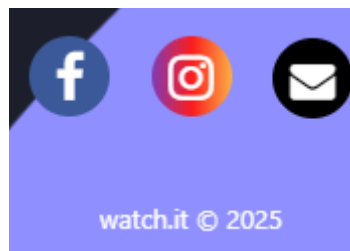




## Fehér/Világos mód



A Főoldal legalján a szociális média platformokat láthatja, amely alapján bekövethet minket „Facebook-on”, „Instagram-on”, illetve a felhasználó meg is tud minket keresni „Email-en” hogyha bármi kérdése van.



## Filmek (és sorozatok)

Főoldal

Kattints ide a sorozatokhoz!

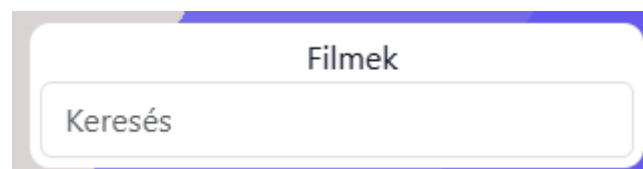
Megjelennek új gombok fent, a navigációs bárba. "Főoldal", "Kattints ide a sorozatokhoz".

Amint a Főoldal gombra nyom, vissza fogja Önt irányítani a Főoldalunkra.

Viszont hogyha a "Kattints ide a sorozatokhoz" gombra nyom, akkor átirányítjuk önt egy másik oldalra, ahol hasonló módon, mint itt, bátran kereshet személyes preferencia alapján rengeteg sorozat közül

Hogyha a felhasználó sikeresen megnyomta a Főoldalon a "FilmeK" gombot akkor átirányította Önt a FilmeK oldalra. Ezen az oldalon egyből meglát filmeKet, mindenféle kategóriából.

Viszont hogyha ennél részletesebben akarna keresni, akkor megetheti azt a "Keresés"-re nyomva. A keresés gombra kattintva, már betudja gépelni a keresni kívánt filmet (Cím szerint). Alatta látható a "Kategóriák" menüpont, ahol a felhasználó kategóriák alapján tudja leszűrni, hogy milyen típusú filmeK között akar keresni.



Amint rányomott valamelyikre, egyből betöltődnek az adott filmeK az Ön által megadott szűrés alapján. Utána láthatja, hogy a filmeK nagy részénél van egy maximum tíz soros (rövidített) leírás, amely segítséget adhat Önnek a tökéletes film megtalálásához. Itt is mint a főoldalon, hogyha ráviszi a filmnek a poszterére a kurzort, akkor előugrik az éppen nézett film poszter.

A "Kategória" menüpontnál a szám, azt jelöli, hogy az adott kategóriában hány film található meg.

## Kategóriák

**Akción (43542)**

**Kaland (24289)**

**Animáció (60603)**

**Vígjáték (149156)**

**Bűnügyi (36748)**

**Dokumentum (183367)**

**Dráma (248571)**

**Családi (28433)**

**Fantasy (24737)**

**Történelmi (18853)**

**Horror (60869)**

**Zenei (45888)**

**Rejtély (22123)**

**Romantikus (55405)**

**Sci-Fi (22599)**

**TV film (28394)**


**Thriller (51385)**

**Háborús (11461)**


**Western (9185)**

Ha lejjebb megyünk egy kicsit, akkor a "Kategóriák" menüpont alatt megláthatja a Top filmeket, illetve a Legújabb megjelenéseket.


### Top filmek




1. Egy Minecraft film




2. G20



3. Novokain




4. The Hard Hit




5. Cleaner


### Legújabb megjelenések




1. Egy Minecraft film




2. Fight or Flight



3. Peter Pan's Neverland Nightmare



4. Old Guy



5. A Majom

Ezek folyamatosan változnak, hogy éppen melyik az a film amelyik a legnépszerűbbek az elmúlt időben. A legújabb megjelenésnél pedig azokat a filmeket láthatja, amik a közelmúltban jelentek meg. Itt hogyha ráviszi a kurzort bármelyik Top filmre vagy Legújabb sorozatra, akkor egy halványabb színnel fogja önnek jelezni, hogy éppen melyiken van a kurzor. Ezekre is rá lehet nyomni, és elvezeti Önt az adott film adatlapjára.

Amint rányom egy kiválasztott filmre, akkor átirányítjuk Önt egy másik oldalra, ahol az éppen kiválasztott filmről részletesebben tudhat meg dolgokat.

Láthatja a jobb oldalon, hogy itt is megjelennek a "Top filmek" és a "Legújabb megjelenések", amelyek esetleg érdekelhetik a felhasználót

A poszter mellett megtekintheti az éppen kiválasztott filmnek az "Előzetesét", amely bemutatja, hogy mit is várhatunk a filmtől.

**Egy Minecraft film**

**Előzetes**

**A Minecraft Film (2025) | Trailer (4K)**

Megtekintheti itt: [YouTube](#)

Megnézzen... Megosztás

**Egy Minecraft film**

Négy Minecraft – Garrett Garrison (Jason Momoa), Henry (Sebastian Eugene Hansen), Natalia (Emma Myers) és Owen (Danielle Brooks) – kaland a videójáték világából, amikor egy véletlenes portálon keresztül egy különös világba kerülnek: A Földalatti világ, egy furcsa, kocka alakú világ, ahol a fantasztikus erővel rendelkező játékosok, akik a Minecraft világában élnek, a valósághoz, a földalatti világba kerülnek, és meg kell védeniük magukat az olyan fenyegetésektől, amilyenek, mint a kockák és a megsemmisítő szörnyek. Segítséghez van Steve (Jack Black), egy tapasztalt kalandor, aki elindul a kockák, Minecraft együttesével és a kockákhoz, rájönnek, hogy éppen a kockák kockáztatnak, amelyek jellemzői, a kockák kockáztatnak – mind a kockákban, mind a kockák világában.

**Érdekelt elmé: A Minecraft Film**

**Megjelenési dátum: 2025-03-31**

**Nyelv: en**

**Hossz: 101 perc**

**Gyártó cégek: Warner Bros. Pictures, Legendary Pictures, Mojang Studios, Vertigo Entertainment, On the Run, Domain Entertainment**

**Gyártó országok: Sweden, United States of America**

**Bevétel: \$434 161 733**

**Költségvetés: \$150 000 000**

**Vélemények**

**freakad\_** [Ajánlom a filmet](#) ★★★★★ **CSIRKE ZSÓRÉ**  
2025-04-11 11:11:18

**dedosalex** [Ajánlom a filmet](#) ★★★★★ **crapper na mar azia de memo**  
2025-04-11 11:10:56

**freakad\_** [Ajánlom a filmet](#) ★★★★★

**Ajánlom a filmet** ★★★★★ **Küldés**

**Top filmek**

1. Egy Minecraft film
2. G20
3. Novokuba
4. The Hard Hit
5. Chesser

**Legújabb megjelenések**

1. Egy Minecraft film
2. Fight or Flight
3. Peter Pan's Neverland Nightmare
4. Old Guy
5. A Mojom

Alatta láthatja a filmnek a címét, ami magyarosítva van, az alatt pedig egy hosszabb leírást kap a felhasználó a filmről. Kicsivel lejjebb megtalálja a címnek az eredeti címét, ami általában külföldi nyelveken van elnevezve.

Ha egy kiválasztott filmre rányomott, és benne van ezen az oldalon, akkor felül láthat egy "Vissza" gombot, amely animálva van amint a kurzort rá tetszik húzni. A gomb megnyomásával az előzőleg megtekintett oldalra fogja Önt visszairányítani.

A film leírása alatt megtalál néhány plusz információt, mint például, hogy mikor jelent meg a film "Megjelenési dátum".

A "Nyelv"-nél filmnek az eredeti nyelvét láthatja, milyen nyelven készült a film.

Alatta láthatja a "Hossz-t", ami az éppen kiválasztott filmnek az időtartamát percben megadva mutatja.

A Hossz alatt a "Gyártó Cég(ek)nek" a neveit láthatja a felhasználó az éppen kiválasztott filmnél. (Melyik cégek gyártották a filmet)

Az alatt a "Gyártó ország(ok)nak" a neveit láthatja a felhasználó az éppen kiválasztott filmnél. (Melyik ország gyártotta a filmet)

Alatta látható a filmből generált "Bevétel", ez nem a profitot jelzi, csak szimplán azt, hogy a film mennyi pénzt hozott be eddig.

Alatta látható a film "Költségvetése", ami azt jelenti, hogy mennyi pénzből sikerült elkészíteniük a filmet a készítőknak.

Ezek a kis információk alatt, megtalálhatja a felhasználó a "Vélemények" részt.

Ahol a többi felhasználó véleményeit fogja tudni elolvasni, illetve, hogy ajánlja-e az éppen kiválasztott filmet vagy nem. A felhasználónak van lehetősége csillagokkal jelezni, hogy 1 és 5 között mennyire tetszett neki a film. Ezekről a felhasználó láthat egy átlagolt pontozást, illetve látja, hogy hányan ajánlották a filmet anélkül, hogy megkeljen azt a felhasználónak számolni.

Hogyha a felhasználó már regisztrált, illetve be van jelentkezve, akkor már tud kommentet írni az összes film alá, tudja pontozni a csillagok alapján, hogy hányasra értékeli az adott

filmet. Illetve, ha tetszett valamelyik felhasználónak a hozzászólása, akkor van lehetősége megnyomni egy "Tetszik" vagy "Nem tetszik" gombot az adott felhasználó véleményénél.

A küldés gomb mellett találhatja azt a gombot, amely, alaphelyzetbe "ajánlom a filmet" van, viszont hogyha megnyom akkor átvált arra, hogy "nem ajánlom"

A csillag pontozás (1-5ig működik), az Ön belátása szerint tudja értékelni ilyen formában az kiválasztott filmet, amit esetleg látott már és más felhasználóknak akar segíteni, hogy érdemes-e esetleg megnéznie a másik felhasználónak.

A "Küldés" gomb megnyomásával sikeresen posztolni fogja az adott film alá az Ön által írt megjegyzést.

Az oldal legalján, megtalálhatja a lapozó rendszerünket, amely megnyomásával akár 500 oldalnyi különböző film vagy sorozatokat tudunk Önnek mutatni.

Az oldal láblécében megtalálja itt is a "Szociális Média" felületeinket és a "Fekete-Fehér módot". (Főoldal leírásában többet olvashat erről)

### 3. Sorozatok

Megjelennek új gombok fent, a navigációs bárba. "Főoldal", "Kattints ide filmekhez".

Amint a Főoldal gombra nyom, vissza fogja Önt irányítani a Főoldalunkra.

Viszont hogyha a "Kattints ide a filmekhez" gombra nyom, akkor átirányítjuk önt egy másik oldalra, ahol hasonló módon, mint itt, bátran kereshet személyes preferencia alapján rengeteg film közül

Hogyha a felhasználó sikeresen megnyomta a Főoldalon a "Sorozatok" gombot akkor átirányította Önt a Sorozatok oldalra. Ezen az oldalon egyből megtalálja a sorozatokat, mindenféle kategóriából.

Viszont hogyha ennél részletesebben akarna keresni, akkor megetheti azt a "Keresés"-re nyomva. A keresés gombra kattintva, már betudja gépelni a keresni kívánt sorozatot (Cím szerint). Alatta látható a "Kategóriák" menüpont, ahol a felhasználó kategóriák alapján tudja leszűrni, hogy milyen típusú sorozatok között akar keresni.

Amint rányomott valamelyikre, egyből betöltődnek az adott sorozatok az Ön által megadott szűrés alapján. Utána láthatja, hogy a sorozatnak nagy részénél van egy maximum tíz soros (rövidített) leírás, amely segítséget adhat Önnek a tökéletes sorozat megtalálásához.

Itt is mint a főoldalon, hogyha ráviszi azt akármelyik sorozatra a poszterére a kurzort, akkor előugrik az éppen nézett film poszter.

Viszont itt nem fog elhomályosodni a többi.

A "Kategória" menüpontnál a szám, azt jelöli, hogy az adott kategóriában hány sorozat található meg az oldalunkon.

Ha lejjebb megyünk egy kicsit, akkor a "Kategóriák" menüpont alatt megláthatja a Top sorozatokat, illetve a Legújabb megjelenéseket. Ezek folyamatosan változnak, hogy éppen melyik az a sorozat, amelyek a legnépszerűbbek az elmúlt időben. A legújabb megjelenésnél pedig azokat a filmeket láthatja, amik a közelmúltban jelentek meg. Itt hogyha ráviszi a kurzort bármelyik Top sorozatra vagy Legújabb megjelenésre, akkor egy halványabb színnel fogja önnek jelezni, hogy éppen melyiken van a kurzor.

Ezekre is rá lehet nyomni, és elvezeti Önt az adott sorozat adatlapjára.

Amint rányom egy kiválasztott sorozatra, akkor átirányítjuk Önt egy másik oldalra, ahol az éppen kiválasztott sorozatról részletesebben tudhat meg dolgokat.

Láthatja a jobb oldalon, hogy itt is megjelennek a "Top sorozatok" és a "Legújabb megjelenések", amelyek esetleg érdekelhetik a felhasználót

A poszter mellett megtekintheti az éppen kiválasztott sorozatnál az "Előzetesét", amely bemutatja, hogy mit is várhatunk a sorozattól.

Alatta láthatja a sorozatnak a címét, ami magyarosítva van, az alatt pedig egy hosszabb leírást kap a felhasználó a sorozatról. Kicsivel lejjebb megtalálja a címnek az eredeti címét, ami általában külföldi nyelveken van elnevezve.

Ha egy kiválasztott sorozatra rányomott, és benne van ezen az oldalon, akkor felül láthat egy "Vissza" gombot, amely animálva van amint a kurzort rá tetszik húzni. A gomb megnyomásával az előzőleg megtekintett oldalra fogja Önt visszairányítani.

A sorozat leírása alatt megtalál néhány plusz információt, mint például, hogy mikor volt a sorozatnak az "Első adásának dátuma".

Alatta megtalálhatja, az "Értékelést", amely megmutatja önnek, hogy hányan szavaztak és 1-10ig milyen értékelésű az éppen kiválasztott sorozat.

A "Nyelv"-nél sorozat az eredeti nyelvét láthatja, milyen nyelven készült a sorozat.



A Nyelv alatt megtalálhatja, a sorozatnak "Státuszát", amely azt jelzi a felhasználónak, hogy a sorozatot még mindig forgatják, készítenek új epizódokat vagy pedig már befejezték, és nem várható több évad/epizód vagy hogyha "Returning Status", akkor az azt jelenti, hogy a jövőben fog készülni több epizód az adott sorozatról.

Alatta láthatja a "Hossz-t", ami az éppen kiválasztott sorozatnak az időtartamát percben megadva mutatja.

A Hossz alatt a "Gyártó Cégek-nek" a neveit láthatja a felhasználó az éppen kiválasztott sorozatnál. (Melyik cégek gyártották a sorozatot)

Az alatt a "Gyártó országok-nak" a neveit láthatja a felhasználó az éppen kiválasztott sorozatnál. (Melyik ország gyártotta a sorozatot)

Az alatt az "Epizódok száma", amely azt jelzi, hogy az adott sorozatból, hány epizód készült el összesen.

Alatta az "Évadok száma", amely azt jelzi, hogy az adott sorozatból, hány évadot készítettek.

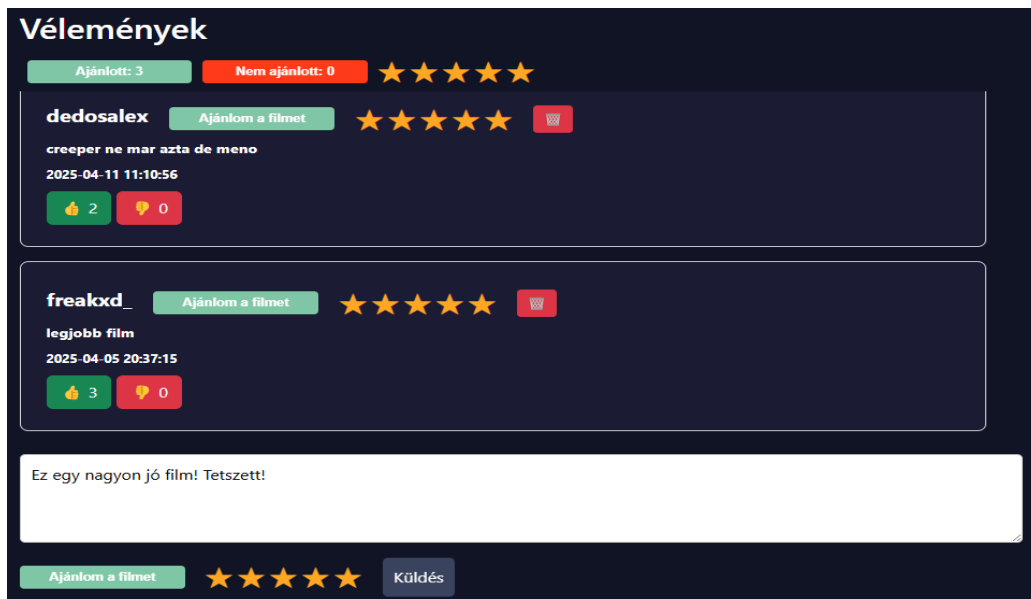
A "Weboldalon" megtalálhatja, a hivatalos weboldalát a sorozatnak.

Ezek a kis információk alatt, megtalálhatja a felhasználó a "Vélemények" részt.

Ahol a többi felhasználó véleményeit fogja tudni elolvasni, illetve, hogy ajánlja-e az éppen kiválasztott sorozatot vagy nem. A felhasználónak van lehetősége csillagokkal jelezni, hogy 1 és 5 között mennyire tetszett neki a sorozat. Ezekről a felhasználó láthat egy átlagolt pontozást, illetve látja, hogy hányan ajánlották a sorozatot anélkül, hogy megkeljen azt a felhasználónak számolni.

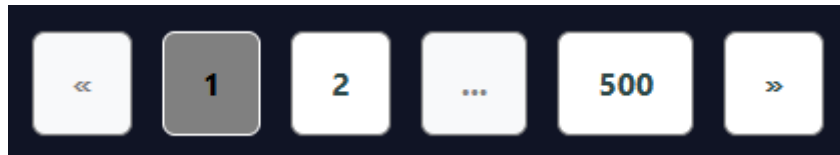
Hogyha a felhasználó már regisztrált, illetve be van jelentkezve, akkor már tud kommentet írni az összes film alá, tudja pontozni a csillagok alapján, hogy hányasra értékeli az adott sorozatot. Illetve, ha tetszett valamelyik felhasználónak a hozzászólása, akkor van lehetősége megnyomni egy "Tetszik" vagy "Nem tetszik" gombot az adott felhasználó véleményénél.

A küldés gomb mellett található azt a gombot, amely, alaphelyzetbe "ajánlom a sorozatot" van, viszont hogyha megnyom akkor átvált arra, hogy "nem ajánlom"



A csillag pontozás (1-5ig működik), az Ön belátása szerint tudja értékelni ilyen formában az kiválasztott sorozatot amit, ha látott már és más felhasználóknak akar segíteni, hogy érdemes-e megnéznie a másik felhasználónak. A "Küldés" gomb megnyomásával sikeresen posztolni fogja az adott film alá az Ön által írt megjegyzést.

Az oldal legalján, megtalálhatja a lapozó rendszerünket, amely megnyomásával akár 500 oldalnyi sorozatot tudunk Önnek kínálni.

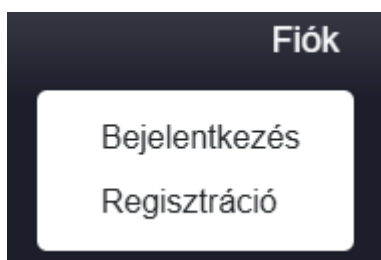


Az oldal láblécében megtalálja itt is a "Szociális Média" felületeinket és a "Fekete-Fehér módot". (Főoldal leírásában többet olvashat erről)

## Fiók

A fiók leugró menü az összes oldalon megtalálható.

Amint rányom a "Fiók" felíratra. Megjelenik kettő választási lehetőség Önnél.



"Bejelentkezés-re" rányomva, hogyha már van egy fiókja, amit beregisztált a weboldalunkra, akkor a helyes felhasználónév, jelszó megadásával fog tudni majd belépni. Amint rányom a bejelentkezés gombra, fog majd jönni az Ön által megadott email címre egy email, amiben egy 6 jegyű kódot fog látni. Azt a kódot majd legyen szíves beírni. Ezután már bent fogja magát találni az oldalunkon!

A dark blue login form titled "Bejelentkezés" with a close button (X) in the top right corner. It contains two input fields: "Felhasználónév" (Username) with a person icon and "Jelszó" (Password) with a key icon. Both fields have a red outline and a red exclamation mark icon on the right. Below the fields is a checkbox labeled "Maradj bejelentkezve" (Stay logged in) and a link "Elfelejtett jelszó?" (Forgot password?). At the bottom, there is a green "Bejelentkezés" (Login) button and a link "Nincs még fiókod?" (Don't have an account yet?).

Hogyha elfelejtette a jelszót, amellyel regisztrált az oldalunkra, bármikor rányomhat az "Elfelejtett jelszó?" lehetőségre, ezután az email címét fogja kérni, amellyel regisztrált. Utána kapni fog egy kódot emailbe, ami beírásával felhoz egy oldalt, ahol megtudja változtatni az Ön elfelejtett jelszavát.

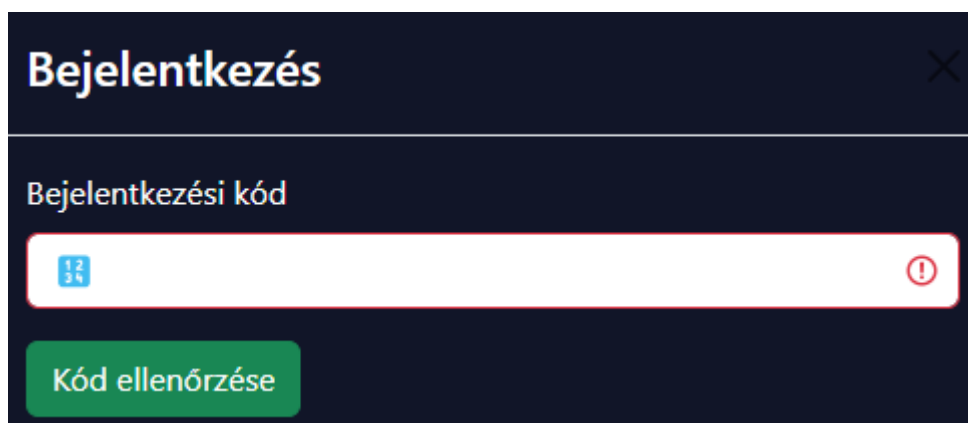
Hogyha esetleg véletlen a "Bejelentkezésre" nyomott volna és nincsen még fiókja, akkor sincs semmi baj. Hiszen jobb alul láthatja a "Nincs még fiókod?" lehetőséget, amint arra rányom, átirányítjuk Önt a regisztrációs oldalunkra

"Regisztráció" fülnél, amint rányom 3 adatot fogunk Öntől kérni.

Felhasználónevét, Jelszavát, email címét (olyan email címet adjon meg amely az Öné, két faktoros azonosító kódot arra az Email címre küldjük)

Hogyha rányomott arra, hogy regisztrál, akkor fog kapni egy kódot emailbe, amellyel sikeresíti a regisztrációt.

Ezután tovább tud menni a bejelentkezésre, ahol a beeregisztrált felhasználónévvel, illetve az Ön által kreált jelszavának megadásával fog tudni bejelentkezni. Ilyenkor is fog kapni egy Kétfaktoros azonosító kódot emailbe, ami 6 számjegyű. E kód beírásával, már sikeresen bent van az oldalon



**watch.it** <noreply.watch.it@gmail.com>

címzett: én ▾

Kedves dozsza,

Az Ön bejelentkezési kódja: **707290**

Üdvözlettel,

[watch.it](https://watch.it)

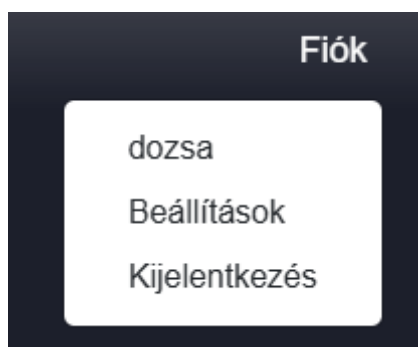
← Válasz

→ Továbbítás



Hogyha úgy kívánja, akkor hogyha megnyomja a gombot, hogy "Maradj bejelentkezve", akkor nem kell aggódni. Az oldal nem fogja Önt kijelentkezteni kilépésnél anélkül, hogy Ön tenné meg azt.

Amint sikeresen bejelentkezett, és ismét megnyomja a "Fiók" menüt a navigációs báron, akkor egy legördülő menübe fogja látni az Ön felhasználónevét, beállításokat és a kijelentkezést.



A beállítások megnyomása után a "Profil" alatt van Önnek lehetősége egy másik felhasználónevet adni magának tetszése szerint.

A dark blue modal window titled 'Beállítások' in white text, with a close button (X) in the top right corner. On the left is a sidebar with two buttons: 'Profil' and 'Biztonság'. The main area shows the heading 'Új felhasználónév' above a text input field. The input field contains a person icon and a red warning icon. Below the input field is a green button labeled 'Mentés'.

A "biztonság" megnyomása után pedig van Önnek lehetősége az email címét megváltoztatni, hogyha úgy tartja. Illetve hogyha új jelszót akar magának akkor arra is van lehetősége megváltoztatni. Viszont itt tudni kell a régi jelszavát is, alatta tudja megadni az újat. Ugyan ez az emailnél, kérjük adja meg azt az email címet amellyel regisztrált az oldalunkra és adja meg az újat amelyre megakarja változtatni az email címét.

## Beállítások

Profil

Biztonság

Régi jelszó

Új jelszó

Mentés

Régi e-mail cím

Új e-mail cím

Mentés

## 3 Fejlesztői dokumentáció

### 3.1 Az alkalmazott fejlesztői eszközök

A weboldal elkészítéséhez az alábbi fejlesztői eszközöket használtuk: PHP, JavaScript, HTML, CSS, SQL, JSON. Az alábbi fejlesztői környezetben Visual Studio Code, phpMyAdmin Segédprogramok: GitHub, Paint, XAMPP, API <https://www.themoviedb.org/> -ről Mindent jogszerűen használtunk tudomásunk szerint. Az API-nál csak annyi volt a kikötés, hogy nem kereshetünk pénzt az oldalon. Egyébként szabad felhasználású mindenki felé az API

#### 3.1.1 Adatmodell leírása

Az adatbázis három fő táblából áll: account, comments és commentlikes.

Ezek a táblák a felhasználói fiókokat, a hozzászólásokat és a hozzászólásokhoz kapcsolódó lájkokat kezelik.

Az alábbiakban bemutatom a táblák szerkezetét, a közöttük lévő kapcsolatokat

#### account Tábla

**Ez a tábla a felhasználói fiókok adatait tárolja. Az alábbi mezőket tartalmazza:**

- **id:** INT (11), elsődleges kulcs, automatikusan generált azonosító a felhasználó számára.
- **username:** VARCHAR (30), a felhasználó neve, amely egyedi azonosítónak szolgál a felületen.
- **password:** VARCHAR (255), a felhasználó jelszava titkosított formában.
- **email:** VARCHAR (255), a felhasználó e-mail címe, amely egyedi kell legyen.
- **role:** INT (2), a felhasználó szerepét jelöli (pl. admin, felhasználó).
- **created\_at:** DATETIME, a fiók létrehozásának időpontja.
- **verification\_code:** INT (6), a fiók aktiválásához szükséges ellenőrző kód.

## **comments Tábla**

**Ez a tábla a felhasználók által írt hozzászólásokat tárolja. Az alábbi mezőket tartalmazza:**

- **id:** INT(11), elsődleges kulcs, automatikusan generált azonosító a hozzászólás számára.
- **user\_id:** INT(11), idegen kulcs, amely a account tábla id mezőjére hivatkozik, és azt jelzi, hogy melyik felhasználó írta a hozzászólást.
- **movie\_id:** INT(11), a film azonosítója, amelyhez a hozzászólás kapcsolódik.
- **series\_id:** INT(11), a sorozat azonosítója, amelyhez a hozzászólás kapcsolódik (opcionális, ha a hozzászólás sorozathoz tartozik).
- **comment:** TEXT, a hozzászólás szövege.
- **rating:** INT(1), a felhasználó által adott értékelés (pl. 1-5 skálán).
- **recommended:** TINYINT(1), logikai érték, amely jelzi, hogy a felhasználó ajánlja-e a tartalmat (1: igen, 0: nem).
- **created\_at:** TIMESTAMP, a hozzászólás létrehozásának időpontja.

## **commentlikes Tábla**

**Ez a tábla a hozzászólásokhoz kapcsolódó lájkokat tárolja. Az alábbi mezőket tartalmazza:**

- **id:** INT (11), elsődleges kulcs, automatikusan generált azonosító a lájk számára.
- **user\_id:** INT (11), idegen kulcs, amely a account tábla id mezőjére hivatkozik, és azt jelzi, hogy melyik felhasználó lájkolta a hozzászólást.
- **comment\_id:** INT (11), idegen kulcs, amely a comments tábla id mezőjére hivatkozik, és azt jelzi, hogy melyik hozzászólást lájkolta a felhasználó.
- **liked\_at:** TIMESTAMP, a lájk időpontja.

## **Kapcsolatok a Táblák Között**

**Az adatbázis táblái között a következő kapcsolatok állnak fenn:**

- **account és comments:** Egy felhasználó (account) több hozzászólást (comments) is írhat. Ez egy egy-több kapcsolat, amelyet a user\_id idegen kulcs biztosít a comments táblában.



- **account és commentlikes:** Egy felhasználó (account) több hozzászólást is lájkolhat (commentlikes). Ez szintén egy egy-több kapcsolat, amelyet a user\_id idegen kulcs biztosít a commentlikes táblában.
- **comments és commentlikes:** Egy hozzászólás (comments) több lájkot (commentlikes) kaphat különböző felhasználóktól. Ez egy egy-több kapcsolat, amelyet a comment\_id idegen kulcs biztosít a commentlikes táblában.

### 3.3 Részletes feladat-specifikáció, algoritmusok

#### Az értékelés rendszer



Az értékelés rendszer lehetővé teszi a felhasználók számára, hogy csillagok segítségével értékeljék a filmeket.

```
<div class="rating">
  <input value="5" name="rate" id="star5" type="radio">
  <label title="5 csillag" for="star5">★</label>
  <input value="4" name="rate" id="star4" type="radio">
  <label title="4 csillag" for="star4">★</label>
  <input value="3" name="rate" id="star3" type="radio" checked="">
  <label title="3 csillag" for="star3">★</label>
  <input value="2" name="rate" id="star2" type="radio">
  <label title="2 csillag" for="star2">★</label>
  <input value="1" name="rate" id="star1" type="radio">
  <label title="1 csillag" for="star1">★</label>
</div>
```

#### Működése:

A csillagok rádiógombokként vannak definiálva, amelyeket a label elemek vizuálisan jelenítenek meg.

##### 1. **input[type="radio"]:**

- A csillagok értékeléséhez rádiógombokat használunk.
- Az id és a for attribútumok kapcsolják össze az input elemet a label-lel.
- A value attribútum az értékelés értékét tárolja (1-től 5-ig).

##### 2. **label:**

- A csillagok vizuális megjelenítésére szolgál.
- A title attribútum tooltipként jeleníti meg az értékelés szöveges leírását.

### 3. **checked:**

- Az alapértelmezett kiválasztott értéket jelöli (pl. 3 csillag).

```
.rating input[type="radio"] {
  appearance: none;
  -webkit-appearance: none;
  -moz-appearance: none;
  background-color: transparent;
  width: 0;
  height: 0;
  margin: 0;
  position: absolute;
}

.rating label {
  font-size: 30px;
  color: #ffa723;
  cursor: pointer;
  transition: color 0.3s ease;
}

.rating input:checked + label {
  color: #ffa723;
}

.rating label:hover,
.rating label:hover ~ label {
  color: #ff9e0b;
}
```

### Működése:

A rádiógombok alapértelmezett stílusa el van rejtve, és a csillagok színe hover és kiválasztott állapotban változik.

#### 1. **appearance: none:**

- Eltávolítja a rádiógomb alapértelmezett stílusát, így csak a csillagok láthatók.

#### 2. **font-size:**

- Meghatározza a csillagok méretét (30px).

#### 3. **color:**

- Az alapértelmezett csillagszín (#ffa723) és hover állapot színe (#ff9e0b).

#### 4. **input:checked + label:**

- A kiválasztott csillag színét állítja be.

## 5. label: hover ~ label:

- Hover állapotban az aktuális csillag és az előtte lévő csillagok színe változik.

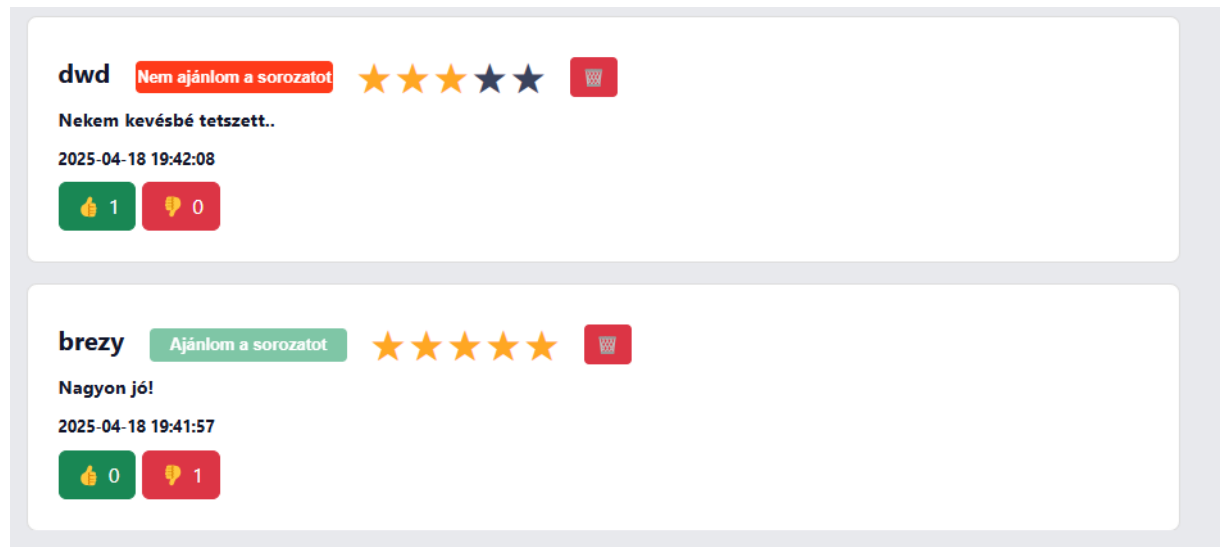
```
//vélemény lementése
if (commentForm) {
  commentForm.addEventListener('submit', function (event) {
    event.preventDefault();
    const comment = commentText.value.trim();
    const rating = document.querySelector('.rating input[name="rate"]:checked').value;
    const recommended = recommendedCheckbox.checked ? 1 : 0;
    const movieId = new URLSearchParams(window.location.search).get('id');

    if (comment && rating && recommended !== null) {
      fetch('../backend/save_comment.php', {
        method: 'POST',
        headers: {
          'Content-Type': 'application/json'
        },
        body: JSON.stringify({ movie_id: movieId, comment: comment, rating: rating, recommended: recommended })
      })
        .then(response => response.json())
        .then(data => {
          if (data.status === 'success') {
            commentText.value = '';
            document.querySelector('.rating input[name="rate"]:checked').checked = false;
            recommendedCheckbox.checked = false;
            loadComments(movieId);
          }
        })
    }
  });
}
```

### Működése:

Az értéket amit megad a felhasználó átküldjük php-ba, majd az lementi az adatbázisba.

## A Komment rendszer



A komment rendszer lehetővé teszi a felhasználók számára, hogy véleményeket írjanak a filmekhez, és azokat megjelenítse az oldalon.

```
<div id="comments">
  <h2>Vélemények</h2>
  <div id="comments-section" class="scrollspy-example" style="height: 400px; overflow-y: scroll;">
    <div id="comments-container"></div>
  </div>
  <form id="comment-form">
    <textarea rows="8" id="comment-text" placeholder="Írja meg véleményét..."></textarea>
    <br>
    <button type="submit">Küldés</button>
  </form>
</div>
```

### Működése:

Az űrlap és a kommentek megjelenítésére szolgáló konténer.

1. **#comments-container:**

- A beküldött kommentek megjelenítésére szolgál.

2. **#comment-form:**

- Az űrlap, amely lehetővé teszi a felhasználók számára, hogy új kommenteket írjanak.

3. **textarea:**

- A felhasználó itt írhatja meg a véleményét.

```
#comments-section {
  height: 400px;
  overflow-y: scroll;
  border: 1px solid #ccc;
  padding: 10px;
  background-color: #f9f9f9;
}

#comment-form textarea {
  width: 100%;
  margin-bottom: 10px;
  padding: 10px;
  border-radius: 5px;
  border: 1px solid #ccc;
}

#comments-container {
  margin-top: 10px;
}
```

### Működése:

A kommentek stílusát és a görgethető területet határozza meg.

1. **#comments-section:**

- Görgethető terület a kommentek megjelenítésére.

2. **textarea:**

- A szövegmező stílusát határozza meg, például szélesség, margó és szegély.

3. **#comments-container:**

- A kommentek megjelenítésére szolgáló konténer.

```

//vélemény lementése
if (commentForm) {
  commentForm.addEventListener('submit', function (event) {
    event.preventDefault();
    const comment = commentText.value.trim();
    const rating = document.querySelector('.rating input[name="rate"]:checked').value;
    const recommended = recommendedCheckbox.checked ? 1 : 0;
    const movieId = new URLSearchParams(window.location.search).get('id');

    if (comment && rating && recommended !== null) {
      fetch('../backend/save_comment.php', {
        method: 'POST',
        headers: {
          'Content-Type': 'application/json'
        },
        body: JSON.stringify({ movie_id: movieId, comment: comment, rating: rating, recommended: recommended })
      })
        .then(response => response.json())
        .then(data => {
          if (data.status === 'success') {
            commentText.value = '';
            document.querySelector('.rating input[name="rate"]:checked').checked = false;
            recommendedCheckbox.checked = false;
            loadComments(movieId);
          }
        })
    }
  });
}
}

```

### Működése:

Ha van vélemény, csillag és értékeli-e vagy sem, akkor lementi a véleményt adatbázisba.

## Film/Sorozat rendszer

Főoldal

Kattints ide a sorozatokhoz!

Fiók

Filmek

Keresés

Kategóriák

Akcio (43596)

Kaland (24334)

Animacio (60674)

Vigjatek (149532)

Bunugyi (36810)

Dokumentum (183900)

Drama (249060)

Csaladi (28480)

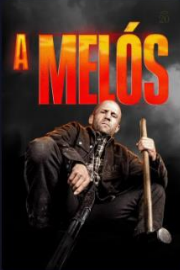
Fantasy (24796)

Torteneelmi (18887)

Horror (60980)


Zenei (45950)

Dejatek (23183)




**A melós**

Levon egyszerű életre vágyik. Otthagyja a terrorrelhárítást, és egy építkezésen dolgozik. Jól megvan a kollégáival, és szereti a tulaját, aki családtagként kezeli. Ezért veszi rosszul, amikor a tulaj kamaszlánya egy bulis este után nem megy haza. A lány nyomába ered. Az egyszerű ügynek látszó eset...




**A Kárhozott Vidék**

A történet középpontjában Gray Alys (Milla Jovovich), egy hatalmas és rettegett boszorkány áll, akit egy királynő megbíz, hogy szerezzen számára egy mágikus képességet, amely lehetővé teszi az alakváltást vérfarkassá. Gray Alys társa a rejtélyes vándor, Boyce (Dave Bautista)...



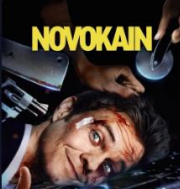
**Egy Minecraft film**

Négy kivüálló – Garrett Garrison (Jason Momoa), Henry (Sebastian Eugene Hansen), Natalie (Emma Myers) és Dawn (Danielle Brooks) – kiszakad a mindennapi életéből, amikor egy titokzatos portálon keresztül egy különös világba kerülnek: A Felsővilágba, egy furcsa, kock...




**G20**


Amikor ostrom alá kerül a G20-csúcs, az Egyesült Államok elnöke, Danielle Sutton válik az első számú célponttá. Miután sikerül kicsúsznia a támadók markából, túl kell járnia az ellenség eszén, hogy megvédje a családját, az országát és a világ vezetőit ebben az izgalmasban bővelkedő...




**NOVOKAIN**



**GUNSLINGERS**





**WOMAN IN THE YARD**

A filmek/sorozatok betöltéséért felelő rendszer, amiket egy ingyenes, TMDB API-ból nyerünk ki és jelenítünk meg.

A kategóriák angol nyelvről magyar nyelvre fordítva.

```
//kategóriák magyar nevekkal
const categoryNames = {
  "Action": "Akció ",
  "Adventure": "Kaland",
  "Animation": "Animáció",
  "Comedy": "Vígjáték",
  "Crime": "Bűnügyi",
  "Documentary": "Dokumentumfilm",
  "Drama": "Dráma",
  "Family": "Családi",
  "Fantasy": "Fantasy",
  "History": "Történelmi",
  "Horror": "Horror",
  "Music": "Zenei",
  "Mystery": "Misztikus",
  "Romance": "Romantikus",
  "Science Fiction": "Sci-fi",
  "TV Movie": "TV film",
  "Thriller": "Thriller",
  "War": "Háborús",
  "Western": "Western"
};
```

A kategóriák betöltéséért felelős kód, ami kilistázza az összes kategóriák és ha választunk egyet, akkor az alapján tölti be a filmeket.

```
//kategóriák betöltése
fetch(categoryUrl)
  .then(response => response.json())
  .then(data => {
    if (data.genres) {
      data.genres.forEach(genre => {
        const categoryItem = document.createElement('div');
        categoryItem.className = 'category2';

        const categoryApiUrl = `https://api.themoviedb.org/3/discover/movie?api_key=${apiKey}&with_genres=${genre.id}&language=hu-HU`;
        fetch(categoryApiUrl)
          .then(response => response.json())
          .then(movieData => {
            const movieCount = movieData.total_results;
            categoryItem.textContent = `${categoryNames[genre.name]} || genre.name} (${movieCount})`;
          })

        categoryItem.dataset.genreId = genre.id;
        categoryItem.addEventListener('click', () => {
          loadMoviesByCategory(genre.id);
        });
        categoryList.appendChild(categoryItem);
      });
    }
  });

//kategóriák alapján betölti a filmet
function loadMoviesByCategory(genreId) {
  selectedGenreId = genreId;
  currentPage = 1;
  loadMovies(currentPage);
}
```



## A filmek/sorozatok betöltéséért felelős kód

```
//filmek betöltése, maximum 500 oldal az api miatt
function loadMovies(page = 1) {
  if (page < 1) page = 1;
  if (page > 500) page = 500;

  let apiUrlWithGenre = apiUrl;
  if (selectedGenreId) {
    apiUrlWithGenre = `https://api.themoviedb.org/3/discover/movie?api_key=${apiKey}&with_genres=${selectedGenreId}&language=hu-HU&page=${page}`;
  } else {
    apiUrlWithGenre = `${apiUrl}&page=${page}`;
  }

  fetch(apiUrlWithGenre)
    .then(response => response.json())
    .then(data => {
      moviesContainer.innerHTML = '';
      if (data.results) {
        data.results.forEach(movie => {
          const movieElement = document.createElement('div');
          movieElement.className = 'col-md-3 movie';
          movieElement.innerHTML = `
            
            <h3 class="movie-title">${movie.title}</h3>
            <p class="movie-overview limited-overview">${movie.overview}</p>
          `;
          movieElement.addEventListener('click', () => {
            window.location.href = `filmek?id=${movie.id}`;
          });
          moviesContainer.appendChild(movieElement);
        });
        setupPagination(data.page, data.total_pages);
      }
    })
  })
}
```

```
function loadTvShows(page = 1) {
  if (page < 1) page = 1;
  if (page > 500) page = 500;

  let apiUrlWithGenre = apiUrl;
  if (selectedGenreId) {
    apiUrlWithGenre = `https://api.themoviedb.org/3/discover/tv?api_key=${apiKey}&with_genres=${selectedGenreId}&language=hu-HU&page=${page}`;
  } else {
    apiUrlWithGenre = `${apiUrl}&page=${page}`;
  }

  fetch(apiUrlWithGenre)
    .then(response => response.json())
    .then(data => {
      tvContainer.innerHTML = '';
      if (data.results) {
        data.results.forEach(tv => {
          if (!/^[\u0000-\u007F]+$/.test(tv.name)){
            const tvElement = document.createElement('div');
            tvElement.className = 'col-md-3 tv';
            tvElement.innerHTML = `
              
              <h3 class="tv-title">${tv.name}</h3>
              <p class="tv-overview limited-overview">${tv.overview}</p>
            `;
            tvElement.addEventListener('click', () => {
              window.location.href = `sorozatok?id=${tv.id}`;
            });
            tvContainer.appendChild(tvElement);
          }
        });
        setupPagination(data.page, data.total_pages);
      }
    })
  })
}
```

Egy adott film/sorozat oldalának betöltése ha rányom a felhasználó,  
?id=xxxxxx URL-el, előzetessel és véleményekkel.

```
//egy adott film betöltése
function loadMovieById(movieId) {
  const movieApiUrl = `https://api.themoviedb.org/3/movie/${movieId}?api_key=${apiKey}&language=hu-HU`;
  const videoApiUrl = `https://api.themoviedb.org/3/movie/${movieId}/videos?api_key=${apiKey}`;

  fetch(movieApiUrl)
    .then(response => response.json())
    .then(movie => {
      moviesContainer.innerHTML = '';
      const movieElement = document.createElement('div');
      movieElement.className = 'col-md-4 movie';
      movieElement.innerHTML = `
        
      `;
      moviesContainer.appendChild(movieElement);

      fetch(videoApiUrl)
        .then(response => response.json())
        .then(data => {
          if (data.results.length > 0) {
            const trailer = data.results.find(video => video.type === 'Trailer' && video.site === 'YouTube');
            if (trailer) {
              const trailerElement = document.createElement('div');
              trailerElement.className = 'col-md-8 trailer';
              trailerElement.innerHTML = `
                <h4>Előzetes</h4>
                <iframe width="100%" height="355" src="https://www.youtube.com/embed/${trailer.key}" frameborder="0" allowfullscreen></iframe>
              `;
              moviesContainer.appendChild(trailerElement);
            }
          } else {
            const trailerElement = document.createElement('div');
            trailerElement.className = 'col-md-8 trailer';
            trailerElement.innerHTML = `
                <h4>Előzetes</h4>
                <p>Nincs elérhető előzetes.</p>
              `;
            moviesContainer.appendChild(trailerElement);
          }

          const movieElement2 = document.createElement('div');
          movieElement2.className = 'col-md-12';
          movieElement2.innerHTML += `
            <h3 class="movie-title">${movie.title}</h3>
            <p class="movie-overview">${movie.overview}</p>
            <p class="movie-original-title"><strong>Eredeti cím:</strong> ${movie.original_title}</p>
            <p class="movie-release-date"><strong>Megjelenési dátum:</strong> ${movie.release_date}</p>
            <p class="movie-original-language"><strong>Nyelv:</strong> ${movie.original_language}</p>
            <p class="movie-runtime"><strong>Hossz:</strong> ${movie.runtime} perc</p>
            <p class="movie-production-companies"><strong>Gyártó cégek:</strong> ${movie.production_companies.map(company => company.name).join(', ')}</p>
            <p class="movie-production-countries"><strong>Gyártó országok:</strong> ${movie.production_countries.map(country => country.name).join(', ')}</p>
            <p class="movie-revenue"><strong>Bevétel:</strong> ${movie.revenue.toLocaleString()}</p>
            <p class="movie-budget"><strong>Költségvetés:</strong> ${movie.budget.toLocaleString()}</p>
          `;
          moviesContainer.appendChild(movieElement2);
        })
      .then(() => loadComments(movieId));
    })
}
```

```

//egy adott sorozat betöltése
function loadTvById(tvId) {
  const tvApiUrl = 'https://api.themoviedb.org/3/tv/${tvId}?api_key=${apiKey}&language=hu-HU';
  const videoApiUrl = 'https://api.themoviedb.org/3/tv/${tvId}/videos?api_key=${apiKey}';

  fetch(tvApiUrl)
    .then(response => response.json())
    .then(tv => {
      if (!/^[\u0000-\u007F]+$/.test(tv.name)) {
        tvContainer.innerHTML = '';
        const tvElement = document.createElement('div');
        tvElement.className = 'col-md-4 tv';
        tvElement.innerHTML = `
           response.json())
          .then(data => {
            if (data.results && data.results.length > 0) {
              const trailer = data.results.find(video => video.type === 'Trailer' && video.site === 'YouTube');
              if (trailer) {
                const trailerElement = document.createElement('div');
                trailerElement.className = 'col-md-8 trailer';
                trailerElement.innerHTML = `
                  <h4>Előzetes</h4>
                  <iframe width="560" height="315" src="https://www.youtube.com/embed/${trailer.key}" frameborder="0" allowfullscreen></iframe>
                `;
                tvContainer.appendChild(trailerElement);
              }
            } else {
              const trailerElement = document.createElement('div');
              trailerElement.className = 'col-md-8 trailer';
              trailerElement.innerHTML = `
                <h4>Trailer</h4>
                <p>Nincs elérhető előzetes.</p>
              `;
              tvContainer.appendChild(trailerElement);
            }
            const tvElement = document.createElement('div');
            tvElement.className = 'col-md-12';
            tvElement.innerHTML = `
              <h3 class="tv-title">${tv.name}</h3>
              <p class="tv-overview">${tv.overview}</p>
              <p><strong>Eredeti cím:</strong> ${tv.original_name}</p>
              <p><strong>Első adás dátuma:</strong> ${tv.first_air_date}</p>
              <p><strong>Értékelés:</strong> ${tv.vote_average} (${tv.vote_count} szavazat)</p>
              <p><strong>Nyelv:</strong> ${tv.original_language}</p>
              <p><strong>Státusz:</strong> ${tv.status}</p>
              <p><strong>Tagline:</strong> ${tv.tagline}</p>
              <p><strong>Gyártó cégek:</strong> ${tv.production_companies.map(company => company.name).join(', ')}</p>
              <p><strong>Gyártó országok:</strong> ${tv.production_countries.map(country => country.name).join(', ')}</p>
              <p><strong>Epizódok száma:</strong> ${tv.number_of_episodes}</p>
              <p><strong>Évadok száma:</strong> ${tv.number_of_seasons}</p>
              <p><strong>Weboldal:</strong> <a href="${tv.homepage}" target="_blank">${tv.homepage}</a></p>
            `;
            tvContainer.appendChild(tvElement);

            loadComments(tvId);
          })
        })
    })
}

```

A vélemények betöltéséért felelős kód

```

//völveszték betöltése
function loadComments(movieId) {
  fetch('../backend/get_comments.php?movie_id=${movieId}')
    .then(response => response.json())
    .then(data => {
      commentsContainer.innerHTML = '';
      if (data.comments && data.comments.length > 0) {
        let totalRating = 0;
        let recommendedCount = 0;
        let notRecommendedCount = 0;

        data.comments.forEach((comment, index) => {
          totalRating += parseInt(comment.rating, 10);
          if (comment.recommended) {
            recommendedCount++;
          } else {
            notRecommendedCount++;
          }

          const commentElement = document.createElement('div');
          commentElement.className = 'custom-comment-card';
          commentElement.id = `comment-${index + 1}`;
          commentElement.innerHTML = `
            <div class="custom-comment-card-body">
              <div class="comment-header">
                <div class="custom-comment-card-title">${comment.username}</div>
                <div class="checkbox-wrapper-18" style="pointer-events: none;">
                  <input type="checkbox" id="cb-${index}" class="tgl tgl-flip" ${comment.recommended ? 'checked' : ''} disabled>
                  <label for="cb-${index}" data-tg-on="Ajánlom a filmet" data-tg-off="Nem ajánlom a filmet" class="tgl-btn"></label>
                </div>
                <div class="rating" style="pointer-events: none;">
                  ${[5, 4, 3, 2, 1].map(value => `
                    <input value="${value}" name="rate-${index}" id="star${value}-${index}" type="radio" ${comment.rating == value ? 'checked' : ''} disabled>
                    <label title="text" for="star${value}-${index}"></label>
                  `).join('')}
                </div>
                ${current_user_role === 1 || parseInt(comment.user_id, 10) === parseInt(current_user_id, 10) ? `
              <div class="delete-comment-container">
                <button class="delete-comment-btn btn btn-danger btn-sm" data-comment-id="${comment.id}">✖ </button>
                <div class="confirm-buttons" style="display: none;">
                  <button class="confirm-delete-btn btn btn-sm btn-success" data-comment-id="${comment.id}"></button>
                  <button class="cancel-delete-btn btn btn-sm btn-secondary" style="background-color: red;"></button>
                </div>
              </div>
            ` : ''}
            </div>
            <p class="custom-comment-card-text">${comment.comment}</p>
            <p class="custom-comment-card-text"><small>${comment.created_at}</small></p>
            ${status === 'logged in' ? `
              <div class="like-dislike-container">
                <button class="like-btn btn btn-success" data-comment-id="${comment.id}">
                  ▲ ${comment.like_count}
                </button>
                <button class="dislike-btn btn btn-danger" data-comment-id="${comment.id}">
                  ▼ ${comment.dislike_count}
                </button>
              </div>
            ` : ''}
          </div>
        `;
        commentsContainer.appendChild(commentElement);
      });

      const averageRating = Math.round(totalRating / data.comments.length);
      reviewSummary.innerHTML = `
        <div class="checkbox-wrapper-18">
          <input type="checkbox" class="tgl tgl-flip" checked disabled>
          <label data-tg-on="Ajánlott: ${recommendedCount}" data-tg-off="Ajánlott: ${recommendedCount}" class="tgl-btn" style="margin-left: 7px;"></label>
        </div>
        <div class="checkbox-wrapper-18" style="margin-left: 18px;">
          <input type="checkbox" class="tgl tgl-flip" disabled>
          <label data-tg-off="Nem ajánlott: ${notRecommendedCount}" class="tgl-btn" style="color: red;"></label>
        </div>
        <div class="rating">
          ${[5, 4, 3, 2, 1].map(value => `
            <input value="${value}" name="average-rate" id="average-star${value}" type="radio" ${averageRating == value ? 'checked' : ''} disabled>
            <label title="text" for="average-star${value}"></label>
          `).join('')}
        </div>
      `;
    } else {
      commentsContainer.innerHTML = `<p>Ezen a filmen nincs még vélemény.</p>`;
      reviewSummary.innerHTML = '';
    }
  });
}

```

```

//vélemények betöltése
function loadComments(tvId) {
  fetch('./backend/get_comments.php?series_id=${tvId}')
    .then(response => response.json())
    .then(data => {
      commentsContainer.innerHTML = '';
      if (data.comments && data.comments.length > 0) {
        let totalRating = 0;
        let recommendedCount = 0;
        let notRecommendedCount = 0;

        data.comments.forEach((comment, index) => {
          totalRating += parseInt(comment.rating, 10);
          if (comment.recommended) {
            recommendedCount++;
          } else {
            notRecommendedCount++;
          }
        });

        const commentElement = document.createElement('div');
        commentElement.className = 'custom-comment-card';
        commentElement.id = `comment-${index + 1}`;
        commentElement.innerHTML = `
          <div class="custom-comment-card-body">
            <div class="comment-header">
              <div class="custom-comment-card-title">${comment.username}</div>
              <div class="checkbox-wrapper-18" style="pointer-events: none;">
                <input type="checkbox" id="ch-${index}" class="tgl tgl-flip" ${comment.recommended ? 'checked' : ''} disabled>
                <label for="ch-${index}" data-tg-on="Ajánlás a sorozatot" data-tg-off="Nem ajánlás a sorozatot" class="tgl-btn"></label>
              </div>
              <div class="rating" style="pointer-events: none;">
                ${[5, 4, 3, 2, 1].map(value => `
                  <input value="${value}" name="rate-${index}" id="star${value}-${index}" type="radio" ${comment.rating == value ? 'checked' : ''} disabled>
                  <label title="text" for="star${value}-${index}"></label>
                `).join('')}
              </div>
              ${current_user_role === 1 || parseInt(comment.user_id, 10) === parseInt(current_user_id, 10) ? `
                <div class="delete-comment-container">
                  <button class="delete-comment-btn btn btn-danger btn-sm" data-comment-id="${comment.id}">✖</button>
                  <div class="confirm-buttons" style="display: none;">
                    <button class="confirm-delete-btn btn btn-sm btn-success" data-comment-id="${comment.id}"></button>
                    <button class="cancel-delete-btn btn btn-sm btn-secondary" style="background-color: red;">X</button>
                  </div>
                </div>
              ` : ''}
            </div>
            <p class="custom-comment-card-text">${comment.comment}</p>
            <p class="custom-comment-card-text"><small>${comment.created_at}</small></p>
            ${status === 'logged_in' ? `
              <div class="like-dislike-container">
                <button class="like-btn btn btn-success" data-comment-id="${comment.id}">
                  ▲ ${comment.like_count}
                </button>
                <button class="dislike-btn btn btn-danger" data-comment-id="${comment.id}">
                  ▼ ${comment.dislike_count}
                </button>
              </div>
            ` : ''}
          </div>
        `;
        commentsContainer.appendChild(commentElement);
      });

      const averageRating = Math.round(totalRating / data.comments.length);
      reviewSummary.innerHTML = `
        <div class="checkbox-wrapper-18">
          <input type="checkbox" class="tgl tgl-flip" checked disabled>
          <label data-tg-on="Ajánlott: ${recommendedCount}" data-tg-off="Ajánlott: ${recommendedCount}" class="tgl-btn" style="margin-left: 7px;"></label>
        </div>
        <div class="checkbox-wrapper-18" style="margin-left: 18px;">
          <input type="checkbox" class="tgl tgl-flip" disabled>
          <label data-tg-off="Nem ajánlott: ${notRecommendedCount}" class="tgl-btn" style="color: red;"></label>
        </div>
        <div class="rating">
          ${[5, 4, 3, 2, 1].map(value => `
            <input value="${value}" name="average-rate" id="average-star${value}" type="radio" ${averageRating == value ? 'checked' : ''} disabled>
            <label title="text" for="average-star${value}"></label>
          `).join('')}
        </div>
      `;

      if (data.comments.length > 10) {
        const scrollSpy = new bootstrap.ScrollSpy(document.body, {
          target: '#comments-section'
        });
      }
    } else {
      commentsContainer.innerHTML = `<p>Nincs még vélemény.</p>`;
      reviewSummary.innerHTML = '';
    }
  })
}

```

A vélemények mentéséért felelős kód

```

//vélemény írása, mentése
if (commentForm) {
  commentForm.addEventListener('submit', function (event) {
    event.preventDefault();
    const comment = commentText.value.trim();
    const rating = document.querySelector('.rating input[name="rate"]:checked').value;
    const recommended = recommendedCheckbox.checked ? 1 : 0;
    const seriesId = new URLSearchParams(window.location.search).get('id');

    if (comment && rating && recommended !== null) {
      fetch('../backend/save_comment.php', {
        method: 'POST',
        headers: {
          'Content-Type': 'application/json'
        },
        body: JSON.stringify({ series_id: seriesId, comment: comment, rating: rating, recommended: recommended })
      })
        .then(response => response.json())
        .then(data => {
          if (data.status === 'success') {
            commentText.value = '';
            document.querySelector('.rating input[name="rate"]:checked').checked = false;
            recommendedCheckbox.checked = false;
            loadComments(seriesId);
          }
        })
    }
  });
}
}

```

```

//vélemény lementése
if (commentForm) {
  commentForm.addEventListener('submit', function (event) {
    event.preventDefault();
    const comment = commentText.value.trim();
    const rating = document.querySelector('.rating input[name="rate"]:checked').value;
    const recommended = recommendedCheckbox.checked ? 1 : 0;
    const movieId = new URLSearchParams(window.location.search).get('id');

    if (comment && rating && recommended !== null) {
      fetch('../backend/save_comment.php', {
        method: 'POST',
        headers: {
          'Content-Type': 'application/json'
        },
        body: JSON.stringify({ movie_id: movieId, comment: comment, rating: rating, recommended: recommended })
      })
        .then(response => response.json())
        .then(data => {
          if (data.status === 'success') {
            commentText.value = '';
            document.querySelector('.rating input[name="rate"]:checked').checked = false;
            recommendedCheckbox.checked = false;
            loadComments(movieId);
          }
        })
    }
  });
}
}

```

A vélemények törléséért felelő kód

```

//vélemény törlése
commentsContainer.addEventListener('click', function (event) {
    const target = event.target;

    if (target.classList.contains('delete-comment-btn')) {
        const deleteContainer = target.closest('.delete-comment-container');
        const confirmButtons = deleteContainer.querySelector('.confirm-buttons');
        confirmButtons.style.display = 'block';
        target.style.display = 'none';
    }

    if (target.classList.contains('confirm-delete-btn')) {
        const commentId = target.dataset.commentId;

        fetch('../backend/delete_comment.php', {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json'
            },
            body: JSON.stringify({ comment_id: commentId })
        })
        .then(response => response.json())
        .then(data => {
            if (data.status === 'success') {
                const movieId = new URLSearchParams(window.location.search).get('id');
                loadComments(movieId);
            }
        })
    }

    if (target.classList.contains('cancel-delete-btn')) {
        const deleteContainer = target.closest('.delete-comment-container');
        const confirmButtons = deleteContainer.querySelector('.confirm-buttons');
        const deleteButton = deleteContainer.querySelector('.delete-comment-btn');
        confirmButtons.style.display = 'none';
        deleteButton.style.display = 'inline-block';
    }
});

```



```

//vélemény törlése
commentsContainer.addEventListener('click', function (event) {
  const target = event.target;

  if (target.classList.contains('delete-comment-btn')) {
    const deleteContainer = target.closest('.delete-comment-container');
    const confirmButtons = deleteContainer.querySelector('.confirm-buttons');
    confirmButtons.style.display = 'block';
    target.style.display = 'none';
  }

  if (target.classList.contains('confirm-delete-btn')) {
    const commentId = target.dataset.commentId;

    fetch('../backend/delete_comment.php', {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json'
      },
      body: JSON.stringify({ comment_id: commentId })
    })
      .then(response => response.json())
      .then(data => {
        if (data.status === 'success') {
          const tvId = new URLSearchParams(window.location.search).get('id');
          loadComments(tvId);
        }
      })
  }

  if (target.classList.contains('cancel-delete-btn')) {
    const deleteContainer = target.closest('.delete-comment-container');
    const confirmButtons = deleteContainer.querySelector('.confirm-buttons');
    const deleteButton = deleteContainer.querySelector('.delete-comment-btn');
    confirmButtons.style.display = 'none';
    deleteButton.style.display = 'inline-block';
  }
});

```

Like/dislike gombok működéséért felelős kód



```
//like/dislike gombok működése
commentsContainer.addEventListener('click', function (event) {
  const target = event.target;

  if (target.classList.contains('like-btn') || target.classList.contains('dislike-btn')) {
    const commentId = target.dataset.commentId;
    const likeType = target.classList.contains('like-btn') ? 1 : 0;

    fetch('../backend/like_comment.php', {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json'
      },
      body: JSON.stringify({ comment_id: commentId, like_type: likeType })
    })
    .then(response => response.json())
    .then(data => {
      if (data.status === 'success') {
        const movieId = new URLSearchParams(window.location.search).get('id');
        loadComments(movieId);
      }
    })
  }
});
```

```
//like/dislike gombok eseménykezelője
commentsContainer.addEventListener('click', function (event) {
  const target = event.target;

  if (target.classList.contains('like-btn') || target.classList.contains('dislike-btn')) {
    const commentId = target.dataset.commentId;
    const likeType = target.classList.contains('like-btn') ? 1 : 0;

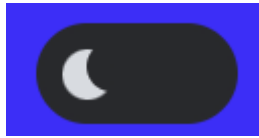
    fetch('../backend/like_comment.php', {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json'
      },
      body: JSON.stringify({ comment_id: commentId, like_type: likeType })
    })
    .then(response => response.json())
    .then(data => {
      if (data.status === 'success') {
        const tvId = new URLSearchParams(window.location.search).get('id');
        loadComments(tvId);
      }
    })
  }
});
```

Keresés a filmek/sorozatok között-ért felelős kód

```
//keresés
function searchMoviesByTitle(title) {
  const searchApiUrl = `https://api.themoviedb.org/3/search/movie?api_key=${apiKey}&query=${title}&language=hu-HU`;
  fetch(searchApiUrl)
    .then(response => response.json())
    .then(data => {
      moviesContainer.innerHTML = '';
      if (data.results) {
        data.results.forEach(movie => {
          const movieElement = document.createElement('div');
          movieElement.className = 'col-md-3 movie';
          movieElement.innerHTML = `
            
            <h3 class="movie-title">${movie.title}</h3>
            <p class="movie-overview limited-overview">${movie.overview}</p>
          `;
          movieElement.addEventListener('click', () => {
            window.location.href = `filmek?id=${movie.id}`;
          });
          moviesContainer.appendChild(movieElement);
        });
      }
    });
}
```

```
//keresés
function searchTvByTitle(title) {
  const searchApiUrl = `https://api.themoviedb.org/3/search/tv?api_key=${apiKey}&query=${title}&language=hu-HU`;
  fetch(searchApiUrl)
    .then(response => response.json())
    .then(data => {
      tvContainer.innerHTML = '';
      if (data.results) {
        data.results.forEach(tv => {
          if (!/^[\u0000-\u007F]+/.test(tv.name)) {
            const tvElement = document.createElement('div');
            tvElement.className = 'col-md-4 tv';
            tvElement.innerHTML = `
              
              <h3 class="tv-title">${tv.name}</h3>
              <p class="tv-overview">${tv.overview}</p>
            `;
            tvElement.addEventListener('click', () => {
              window.location.href = `sorozatok?id=${tv.id}`;
            });
            tvContainer.appendChild(tvElement);
          }
        });
      }
    });
}
```

## Fekete/Fehér mód rendszer



A fekete/fehér mód lehetővé teszi a felhasználók számára, hogy a weboldal színvilágát sötét vagy világos módra állítsák.

```
<div class="toggle-switch">
  <label class="switch-label">
    <input type="checkbox" class="checkbox" id="fekete-feher" checked>
    <span class="slider"></span>
  </label>
</div>
```

### Működése:

A kapcsoló, amely lehetővé teszi a felhasználók számára a mód váltását.

1. **input[type="checkbox"]:**
  - A felhasználó ezzel a kapcsolóval válthat a fekete/fehér mód között.
2. **id="fekete-feher":**
  - Azonosító, amelyet a JavaScript használ az állapot figyelésére.
3. **span.slider:**
  - A vizuális kapcsoló megjelenítésére szolgál.

```
body.darkmode {
  background-color: #000;
  color: #fff;
}

body.darkmode a {
  color: #ffa723;
}

.toggle-switch {
  position: fixed;
  top: 10px;
  right: 10px;
}

.switch-label .slider {
  width: 40px;
  height: 20px;
  background-color: #ccc;
  border-radius: 20px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

.checkbox:checked + .slider {
  background-color: #ffa723;
}
```

A lementett színskálák (az oldal alapértelmezetten fehér, az az a root classban indul)

```
:root{
  --base-color: white;
  --base-variant: #e8e9ed;
  --text-color: #111528;
  --filser-color: white;
  --secondary-text: #232738;
  --primary-color: #3a435d;
  --accent-color: #0071ff;
  --bejreg-color: #ffffff;
  --comment-color: #ffffff;
  --indexsvg-color: #e3e3ff;
  --brezyhover-color: #bababa;
}

.darkmode{
  --base-color: #1c1f2b;
  --base-variant: #101425;
  --text-color: #ffffff;
  --filser-color: #111528;
  --secondary-text: #a4a5b8;
  --primary-color: #3a435d;
  --accent-color: #0071ff;
  --bejreg-color: #111528;
  --comment-color: rgb(27, 28, 51);
  --indexsvg-color: #2d2d4a;
  --brezyhover-color: #3d3d65;
}
```

Működése:

Meghatározza a sötét mód stílusait és a kapcsoló megjelenését.

1. **body.darkmode:**

- A sötét mód stílusait határozza meg (fekete háttér, fehér szöveg).

2. **toggle-switch:**

- A kapcsoló pozícióját rögzíti az oldal tetején.

3. **.checkbox:checked + .slider:**

- Ha a kapcsoló be van kapcsolva, a slider színe megváltozik.

```

$(document).ready(function () {

    // fekete-fehér mód

    let darkmode = localStorage.getItem('darkmode');
    const feketeFeher = document.getElementById("fekete-feher");

    const enableDarkmode = () => {
        document.body.classList.add('darkmode');
        localStorage.setItem('darkmode', 'active');
        feketeFeher.checked = true;
    };

    const disableDarkmode = () => {
        document.body.classList.remove('darkmode');
        localStorage.setItem('darkmode', 'null');
        feketeFeher.checked = false;
    };

    if (darkmode === "active") {
        enableDarkmode();
    } else {
        disableDarkmode();
    }

    if (feketeFeher) {
        feketeFeher.addEventListener("change", () => {
            if (feketeFeher.checked) {
                enableDarkmode();
            } else {
                disableDarkmode();
            }
        });
    }
}

```

#### Működése:

Figyeli a rendszer ha a checkbox be van jelölve akkor az oldal színsémája megváltozik.

## 3.4 Tesztelési dokumentáció

### Legalább 3 különböző tesztet részletes bemutatása

#### *Tesztet 1.1: Értékelés rendszer működése*

- **Felhasználói tevékenység:** A felhasználó kiválaszt egy csillagot az értékelés rendszerben, majd elküldi az értékelését.
- **Várt eredmény:** A kiválasztott csillag kiemelkedik, és az értékelés értéke megjelenik a konzolon vagy a szerveren.
- **Kapott üzenet:**
  - Sikeres beküldés esetén: "Kommentje és csillag megjelenik a megadott értékkel"
  - Ha nem választott csillagot: "A komment nem jelenik meg"
- **Teendő:**
  - Sikeres üzenet esetén nincs további teendő.
  - Hibaüzenet esetén a felhasználót figyelmeztetni kell, hogy válasszon egy csillagot.

#### *Tesztet 1.2: Komment rendszer működése*

- **Felhasználói tevékenység:** A felhasználó beír egy kommentet, majd rákattint a "Küldés" gombra.
- **Várt eredmény:** A komment megjelenik a kommentek listájában, és a szövegmező kiürül.

- **Kapott üzenet:**
  - Sikeres beküldés esetén: "Komment sikeresen hozzáadva."
  - Ha üres a szövegmező: "A komment mező nem lehet üres!"
- **Teendő:**
  - Sikeres üzenet esetén nincs további teendő.
  - Hibaüzenet esetén a felhasználót figyelmeztetni kell, hogy írjon a mezőbe!

### *Teszt eset 1.3: Fekete/Fehér mód váltása*

- **Felhasználói tevékenység:** A felhasználó bekapcsolja a fekete/fehér módot a kapcsolóval.
- **Várt eredmény:** Az oldal színvilága sötét módra vált, és a kapcsoló állapota megváltozik.
- **Kapott üzenet:**
  - Sikeres váltás esetén: "Az oldal színsémát vált, sötét vagy világos lesz"
  - Ha a kapcsoló nem működik: -
- **Teendő:**
  - Sikeres üzenet esetén nincs további teendő.
  - Hiba esetén ellenőrizni kell a JavaScript működését és a kapcsoló állapotát.

## **Normál teszteset, extrém teszteset**

- **Forgatókönyv:** A felhasználó beír egy kommentet, kiválaszt egy csillagot, és beküldi az értékelését.
- **Eredmény:** A komment és az értékelés sikeresen megjelenik az oldalon.
- **Következtetés:** A rendszer normál használat mellett megfelelően működik.
- **Extrém teszteset:**
- **Forgatókönyv:** A felhasználó üres kommentet próbál beküldeni, vagy egyszerre több csillagot próbál kiválasztani.



- **Eredmény:**
  - Üres komment esetén hibaüzenet jelenik meg: "A komment mező nem lehet üres!"
  - Több csillag kiválasztása nem lehetséges, mivel a rádiógombok csak egy értéket engednek.
- **Következtetés:** A rendszer megfelelően kezeli a hibás felhasználói tevékenységeket.

### **A tesztelés során kiderült hibák felsorolása**

1. A menün lévő animációk mely, ha rávisszük kurzorunkat a lehetőségekre. Akkor a 'Fiók' fülnél nem jelenik meg az animáció a meghívott 'Bootstrap'-es „dropdown list” script miatt.
  - Megoldás: a 'Fiók' fül helyett egy új oldalt kellene megnyitnia, mely csakis a felhasználó belépésére és regisztrálására van lehetősége.

### **Tesztelési módszerek**

#### *1. Fekete doboz tesztelés:*

- A rendszer működését a felhasználói interakciók alapján teszteltük, anélkül, hogy a belső kódot vizsgáltuk volna.
- Példa: Komment beküldése és a megjelenítés ellenőrzése.

#### *2. Fehér doboz tesztelés:*

- A rendszer belső működését vizsgáltuk, például a JavaScript függvények és a CSS változók helyes működését.
- Példa: A fekete/fehér mód váltásának kódjának ellenőrzése.

#### *3. Stresszteszt:*

- Több komment gyors egymás utáni beküldése, hogy ellenőrizzük a rendszer teljesítményét.
- Eredmény: A rendszer megfelelően kezelte a terhelést.

## 4 Összefoglalás

### **4.1 Önértékelés**

Szerintünk nagyjából sikerült elérni a célunkat a weboldal fejlesztésekor. Pár dolgot, amit még bele akartunk vinni, azt nem igazán tudtuk egyelőre megvalósítani. Ha még több időt teszünk bele, akkor megvalósíthatóak lettek volna azok a célok, mint például a watchlist. Szerintünk sokat tudtunk tanulni ebből a kis projectből, hiszen megízleltük, hogy milyen is csapatban dolgozni, időre elkészíteni egy projectet. Voltak nehézségek, de megküzdöttük azokat. Lustaság, motiváció hiánya megjelent, de beszélgetések után ennek sehol nem volt nyoma. Alig várjuk, hogy ennél csak jobban bele tudjuk magunkat ásni a programozás / fejlesztés világába, és még több tapasztalatot szerezzünk a jövőben. Megtanultuk, hogy inkább az elején kell megnyomni a dolgokat, belevinni a munkát minthogy a vége fele legyen az, hogy kapkodás lesz, ez is egy jó lecke, hogy nem szabad az utolsó pillanatokra hagyni a dolgokat.

### **4.2 Továbbfejlesztési lehetőségek**

Egy teljes felhasználói profil kialakítása. Profilképpel, leírással magadról, hogy lásd milyen kommenteket írtál melyik film vagy sorozat alá.

Watchlist, egy olyan külön gomb minden sorozatnál, filmnél, amelyet hogyha megnyomsz és be vagy jelentkezve akkor elmenti a saját profilodhoz, és egy külön weboldalon megtudod tekinteni, hogy melyik filmeket vagy sorozatot akarod megtekinteni a közeljövőben.

Egy kifejlesztett support rendszer, amely alapján bármikor tudnak írni nekünk a felhasználók, bármilyen hiba esetén.

Egy esetleges fórum, ahol a filmekkel és sorozatokkal kapcsolatosan tudnak a felhasználók kommunikálni, kibeszélni adott témát

## 5 Felhasznált irodalom

[The Movie Database](#) – A filmeket és a sorozatokat tartalmazó ingyenes API-t biztosító weboldal.

[Bootstrap 5](#) – Alapvető reszponzivitásért felelős keretrendszer, amiből például a navbar-t, felbukkanó modal-okat is használjuk több helyen.

[W3Schools](#) – Sok segítséget nyújtó weboldal, amiből például a Fetch API-t is könnyen el lehet sajátítani, de persze sok minden másra is használtuk.

[Uiverse.io](#) – Animációkért, gombokért és egyéb más Felhasználói Felülettel kapcsolatos dolgokért felelős weboldal, ahol ingyenesen elérhető jól funkcionáló dolgok találhatók.