# 1. Bevezetés

A **MovieTreasure** egy közösségi filmmegosztó és értékelő oldal, amely az IMDB-hez hasonló élményt nyújt. A cél, hogy a felhasználók filmeket töltsenek fel, értékeljenek, kommenteljenek, és megosszák véleményüket. A projekt célja egy teljes backend és frontend rendszer készítése Laravel és Angular technológiákkal.

**Technológiák:**

* Backend: Laravel
* Frontend: Angular

A tesztelési terv célja a rendszer backend részének (API) manuális ellenőrzése, Postman segítségével. A frontend tesztelés külön dokumentációban szerepel.

# 2. Tesztkörnyezet

* Backend: Laravel REST API
* Adatbázis: MySQL
* Tesztelő eszköz: Postman
* Frontend: Angular (külön kezelve)

# 3. Tesztesetek

## 3.1 Jelszókezelés

**POST /vizsga-api/new-password**

* Sikeres jelszóváltás (200)
* Hibás régi jelszó (401)
* Érvénytelen új jelszó (422)
* Validáció:
  + Kötelező mezők ellenőrzése
  + Minimum jelszóhossz és karakterek

**POST /vizsga-api/forgot-password**

* Sikeres új jelszó generálás (200)
* Nem létező e-mail (404)
* E-mail küldés hibája esetén: a jelszónak ettől függetlenül generálódnia kell
* Generált jelszó: 10 karakteres random string

## 3.2 Értékelési rendszer

**POST /vizsga-api/add-rating**

* Új értékelés hozzáadása (201)
* Meglévő értékelés frissítése (200, action: updated)
* Érvénytelen értékelés (422 – 0 vagy 6)
* Validáció:
  + Rating: 1-5 közötti érték
  + Show ID létezése (nincs exists validáció)

**POST /vizsga-api/delete-rating**

* Saját értékelés törlése (200)
* Más felhasználó értékelése (403)
* Nem létező értékelés (404)

**GET /vizsga-api/avg-rating/{showid}**

* Átlag lekérdezése
* Nincs értékelés esetén: null
* Bejelentkezve: user\_rating is szerepel

**GET /vizsga-api/avg-ratings**

* Rendezés csökkenő sorrendben (átlag szerint)

## 3.3 Autentikáció

**POST /vizsga-api/login**

* Sikeres bejelentkezés (200)
* Hibás adatok (401)
* Válasz: success, message, token

**POST /vizsga-api/register**

* Egyedi e-mail
* Validáció: név, email formátum

**GET /vizsga-api/user**

* Be nem jelentkezett felhasználó: user mező null, státusz 200

**GET /vizsga-api/dashboard**

* Visszatérő user mező: csak bejelentkezve

**Jogosultságok minden műveletnél**

* Saját adat módosítása/törlése: engedélyezett
* Más felhasználó adatai: 403 Forbidden

## 3.4 Kommentelési rendszer

**POST /vizsga-api/add-comment**

* Sikeres kommentelés (201)
* Üres szöveg: 422

**POST /vizsga-api/delete-comment**

* Saját komment: törölhető (200)
* Más kommentje: 403
* Nem létező komment: 404

**GET /vizsga-api/get-show-comments**

* Legújabb kommentek elöl
* Felhasználó neve is szerepeljen (user\_name)

**POST /vizsga-api/update-comment**

* Sikeres frissítés után edited = 1

## 3.5 Feltöltés / Show-kezelés

**POST /vizsga-api/upload-show**

* Sikeres feltöltés (201)
* Validáció:
  + Mimes: jpg, jpeg, png, webp
  + Max méret: 5MB
* Fájl mentés: public/uploads/vizsga/
* season és episode alapértéke: 0

**POST /vizsga-api/update-show**

* Nem létező műsor: 404
* Más műsora: 403
* Képcsere esetén a régi kép törlése

**POST /vizsga-api/delete-show/{id}**

* Más műsora: 403
* Megjegyzés: a fájl fizikailag **nem** törlődik a szerverről

## 3.6 Keresés és szűrés

**POST /vizsga-api/search**

* Üres/szóköz → üres tömb
* Válasz mezők: id, title, type, image\_url

**POST /vizsga-api/filter**

* Üres/szóköz → üres tömb
* Több kategória: OR kapcsolat

## 3.7 Watchlist funkció

**POST /vizsga-api/add-watchlist/{showid}**

* Sikeres hozzáadás
* Már listán van → 409 (status: exists)

**POST /vizsga-api/remove-watchlist/{showid}**

* Nincs a listán → 404 (success: false)

**POST /vizsga-api/get-watchlist**

* Üres lista: 404
* Válasz: title, image\_url

## 3.8 API hibakezelés

* Hiányzó mezők → 422
* Érvénytelen típus / ID → 404
* Jogosulatlan hozzáférés → 403

## 3.9 Frontend tesztelés (külön dokumentumban)

* UI/UX: reszponzivitás, böngészőkompatibilitás
* Hibakezelés: hibák megjelenítése a felületen
* Integráció: például kommentek betöltése a részletes nézetben

# 4. Tesztelés a fejlesztés során

A tesztelés iteratív módon történt, a fejlesztés minden szakaszában. A Laravel API és Angular frontend integrációja is tesztelve lett, különös figyelemmel az autentikációra és jogosultságokra.

# 5. Összegzés

Ez a tesztelési terv lefedi a MovieTreasure projekt összes lényeges backend funkcióját. A frontend külön tesztelési tervet kap. A rendszer jól működik, a hibák megfelelően kezeltek, és az adatok biztonsága is biztosított.

**Megjegyzés:** A nagy mennyiségű adat kezelésére a lapozás bevezetése javasolt (pl. /get-ratings, /index).

**Tisztázandó végpont:** A /get-user-rating nincs a route-ok között, helyette az /avg-rating/{showid} tartalmazza a user\_rating mezőt bejelentkezés esetén.