

Projektmunka Dokumentálása

Mozi helyfoglaló webalkalmazás

Premozi

Premontrei Szakgimnázium és Technikum

Készthely

2025

Szoftverfejlesztő és -tesztelő szak

5-0613-12-03

Készítette:

Bató Bence, Magyarósi Bálint

## Tartalom

1.	Bevezetés .....	4
2.	Fontos programok.....	5
3.	Technológiai felépítés.....	5
4.	NuGet csomagok .....	6
	Biztonság és Hitelesítés.....	6
	Adatbázis és Entity Framework .....	6
	API és JSON Kezelés .....	6
	Tesztelés .....	6
	Egyéb Segédeszközök.....	7
5.	Adatbázis szerkezete.....	7
	Adatbázis modell .....	8
	Film tábla.....	9
	Terem tábla.....	9
	Vetítés tábla .....	9
	Foglalt székek .....	10
	VetítésSzékek tábla .....	10
	Httplogs .....	10
	Székek tábla.....	11
	FoglalásAdatok tábla.....	11
	Users tábla (felhasználók) .....	12
	Images tábla (képek) .....	12
	jegytipus tábla (jegyárak) .....	12
	Email2FACodes tábla (kétfélepcsős azonosítás kódok).....	13
	__EFMigrationsHistory tábla (migrációs előzmények) .....	13
	Összefoglalás .....	13
	Kapcsolatok .....	13
6.	REST API funkciók.....	15
	Felhasználókezelés (AuthController) .....	15
	Foglalások kezelése (FoglalasController) .....	16
	Termek kezelése (TeremController) .....	16
	Vetítések kezelése (VetitesController) .....	17
	Képek kezelése (ImageController).....	17
7.	Weboldal működése .....	18
8.	NUnit teszt .....	24
	AuthController tesztek .....	24

Regisztráció tesztelése (Register_ReturnsOk_WhenSuccessful): .....	24
Email megerősítés tesztelése (ConfirmEmail_ReturnsOk_WhenTokenIsValid): .....	24
Felhasználói adatok lekérése (GetUser_ReturnsUser_WhenAuthenticated):.....	24
Bejelentkezés állapotának ellenőrzése .....	24
Felhasználó törlésének jogosultságvizsgálata.....	24
FilmController tesztek.....	25
Filmlekérdezési tesztek .....	25
Filmkezelési műveletek.....	26
FoglalásController tesztek .....	27
Jegytípus kezelés tesztjei.....	27
Foglalás lekérdezési tesztek.....	27
CRUD műveletek tesztjei.....	28
TeremController tesztek .....	29
VetitesController tesztek .....	29
Módosítási műveletek tesztjei.....	29
Törlési műveletek tesztjei.....	30
9. Fejlesztési javaslatok .....	31
10. Összegzés/Reflexió .....	31
11. Források.....	32

## 1. Bevezetés

A szakmai vizsgaprojektünk egy mozijegy-foglaló rendszer, amelynek megvalósítását azért választottuk, mert csapatunk egy korábbi iskolai projekt során már dolgozott mozi témájú alkalmazáson, így ezt az ismeretet tovább tudtuk fejleszteni. A projekt célja egy olyan online rendszer létrehozása, amely lehetővé teszi a felhasználók számára, hogy kényelmesen és gyorsan lefoglalják helyeiket a kívánt vetítésekre.

A rendszer fejlesztése során kiemelt figyelmet fordítottunk a hatékony csapatmunkára és a feladatok megfelelő elosztására. Minden csapattag a saját erősségei és tapasztalatai alapján kapott szerepet a projektben, így biztosítva a munka gördülékenységét és a lehető legjobb eredmény elérését. A kommunikáció és az együttműködés kulcsfontosságú volt, rendszeresen tartottunk megbeszéléseket, ahol megosztottuk az előrehaladásunkat, megbeszéltük a felmerülő problémákat, és közösen kerestünk megoldásokat.

A jegyfoglaló rendszer egy webalapú alkalmazásként működik, amely egy MySQL adatbázisra épül. A felhasználók regisztrálhatnak, bejelentkezhetnek, és a rendszer segítségével kiválaszthatják a kívánt vetítést, majd lefoglalhatják helyeiket. Az adatok kezelése és a foglalási folyamat biztosítása egy REST API segítségével történik, amely biztosítja az adatok biztonságos és gyors elérését.

Dokumentációnk részletesen bemutatja a rendszer felépítését, működését és technológiai hátterét. Ismertetjük az alkalmazott fejlesztési eszközöket, az adatbázis szerkezetét, valamint a rendszer főbb funkcióit.

Összességében a projekt nem csak technikai kihívást jelentett számunkra, hanem lehetőséget adott arra is, hogy fejlesszük együttműködési készségeinket, megtanuljuk a hatékony munkamegosztás fontosságát, és valós fejlesztői környezetben szerezzünk tapasztalatot. Bízunk benne, hogy a mozijegy-foglaló rendszerünk egy jól működő, felhasználóbarát megoldást kínál a felhasználók számára.

## 2. Fontos programok

- Visual Studio 2022

A fő fejlesztői környezet, ahol írtuk, debugoltuk és futtattuk a programot, C# és ASP.NET keretrendszer használatával.

- XAMPP

A program adatbázisa egy lokális szerveren van csak jelenleg, ezt a XAMPP-al és a PhpMyAdmin-nal kezeljük (Apache, MySQL, PHP).

- Discord

Kommunikációs platformként szolgált, ahol gyorsan és egyszerűen megbeszéltük a feladatokat, megosztottuk a frissítéseket és tartottuk a kapcsolatot.

- GitHub

A verziókövetéshez és a csapatmunka koordinálásához használtuk, ahol a kódot tároltuk, megosztottuk és együtt dolgoztunk rajta bárholnan.

- ProtonVPN

A ProtonVPN segítségével tudtunk elérni bármilyen segítséget az interneten, ami az iskolai hálózaton néha le volt tiltva. (pl.: Stackoverflow, Discord, W3Schools)

## 3. Technológiai felépítés

- **Backend:** ASP.NET Core 8.0
- **Frontend:** React + Vite
- **Adatbázis:** MySQL
- **Adatbázis kommunikáció:** REST API

## **4. NuGet csomagok**

### **Biztonság és Hitelesítés**

BCrypt.Net-Next (4.0.3): Jelszavak biztonságos hash-elésére és tárolására (pl. regisztráció, bejelentkezés).

Microsoft.AspNetCore.Authentication.JwtBearer (8.0.2): JWT (JSON Web Token) alapú hitelesítés implementálásához API-okban.

Microsoft.IdentityModel.JsonWebTokens (8.6.1): JWT tokenek generálásához és ellenőrzéséhez.

### **Adatbázis és Entity Framework**

Microsoft.EntityFrameworkCore (8.0.2): ORM keretrendszer adatbázis-kezeléshez .NET-ben.

Pomelo.EntityFrameworkCore.MySql (8.0.2): MySQL támogatás Entity Framework Core-hoz.

MySql.Data (8.0.20) & MySql.Data.EntityFrameworkCore (8.0.20): MySQL adatbázis-kapcsolat létrehozásához és kezeléséhez.

Microsoft.EntityFrameworkCore.Design (8.0.2) & Tools (8.0.2): Adatbázis migrációk és scaffolding (pl. dotnet ef parancsok).

### **API és JSON Kezelés**

Microsoft.AspNetCore.Mvc.NewtonsoftJson (8.0.2): JSON serializálás/deserializálás testreszabásához (pl. dátumformátumok, ciklikus referenciák).

Swashbuckle.AspNetCore (6.6.2): Swagger/OpenAPI dokumentáció generálásához .

### **Tesztelés**

NUnit (4.3.2) & NUnit3TestAdapter (5.0.0): Unit tesztek írásához és futtatásához.

xunit (2.9.3) & xunit.runner.visualstudio (3.0.2): Alternatív tesztkeretrendszer CI/CD folyamatokhoz.

Moq (4.20.72): Mock objektumok létrehozásához unit tesztekben.

Microsoft.NET.Test.Sdk (17.13.0): Tesztelési keretrendszer támogatás.

## Egyéb Segédeszközök

Microsoft.AspNetCore.SpaProxy (8.0.2): SPA (pl. React, Vite) fejlesztéshez proxy szerverrel.

Microsoft.VisualStudio.Web.CodeGeneration (8.0.2): Automatikus kódgenerálás (pl. scaffolding).

SendGrid (9.29.3): E-mail küldés (pl. regisztrációs megerősítések, jelszó-visszaállítás).

SixLabors.ImageSharp (3.1.7): Képek feldolgozására (pl. átméretezés, formátumkonverzió).

System.Linq.Async (6.0.1): Aszinkron LINQ műveletek (pl. IEnumerable kezelés).

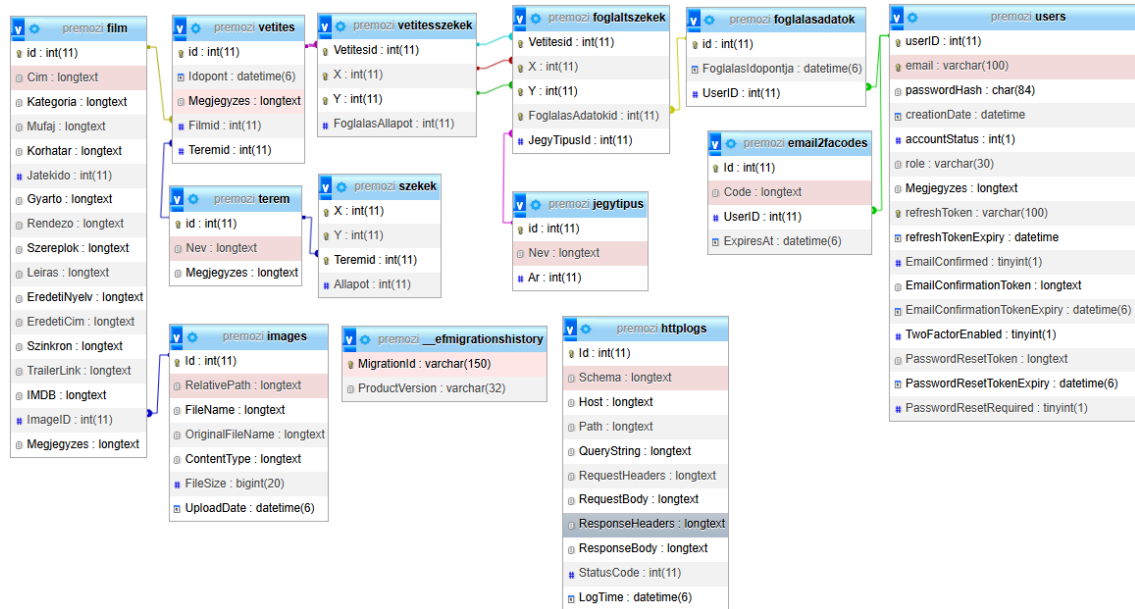
## 5. Adatbázis szerkezete

Az adatbázis a következő táblákból áll:

- **users:** Felhasználók adatait tárolja.
- **film:** A mozi filmjeinek adatait tartalmazza.
- **vetites:** A vetítések adatait tartalmazza.
- **terem:** A mozitermek adatait tartalmazza.
- **email2facodes:** Emailés kétfaktoros hitelesítés ideiglenes kódjai.
- **foglalasadatok:** Felhasználók által létrehozott foglalások.
- **foglaltszekek:** Foglalt székek adatai (milyen jeggyel, melyik vetítésen).
- **httplogs:** API kérések naplózása (hibakereséshez, nyilvántartáshoz).
- **images:** Filmplakátok és egyéb képek tárolása.
- **jegytipus:** Jegytípusok (pl. diák, felnőtt) és áraik.
- **szekek:** A termekben lévő székek elhelyezkedése és állapota (pl. elérhető-e).
- **vetitesszekek:** Vetítésenként, mely székek foglaltak vagy szabadok.

- **\_\_efmigrationshistory**: EntityFrameworkCore automatikusan generált log táblája



## Adatbázis modell



Az adatbázisunk a mozi weboldalának jól működő hátteret biztosít, lehetővé teszi a zavartalan működést.



## Film tábla

Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
id 	int(11)			Nem	Nincs		AUTO_INCREMENT
Cím	longtext	utf8mb4_general_ci		Nem	Nincs		
Kategória	longtext	utf8mb4_general_ci		Nem	Nincs		
Műfaj	longtext	utf8mb4_general_ci		Nem	Nincs		
Korhatár	longtext	utf8mb4_general_ci		Nem	Nincs		
Jatekido	int(11)			Nem	Nincs		
Gyarto	longtext	utf8mb4_general_ci		Nem	Nincs		
Rendezo	longtext	utf8mb4_general_ci		Nem	Nincs		
Szereplok	longtext	utf8mb4_general_ci		Nem	Nincs		
Leiras	longtext	utf8mb4_general_ci		Nem	Nincs		
EredetiNyelv	longtext	utf8mb4_general_ci		Nem	Nincs		
EredetiCim	longtext	utf8mb4_general_ci		Nem	Nincs		
Szinkron	longtext	utf8mb4_general_ci		Nem	Nincs		
TrailerLink	longtext	utf8mb4_general_ci		Nem	Nincs		
IMDB	longtext	utf8mb4_general_ci		Nem	Nincs		
ImageID 	int(11)			Nem	Nincs		
Megjegyzes	longtext	utf8mb4_general_ci		Nem	Nincs		

A moziban vetített filmek adatait tartalmazza (pl. cím, műfaj, leírás).

## Terem tábla

Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
id 	int(11)			Nem	Nincs		AUTO_INCREMENT
Név	longtext	utf8mb4_general_ci		Nem	Nincs		
Megjegyzes	longtext	utf8mb4_general_ci		Nem	Nincs		

A mozi termeit rögzíti (név, elhelyezkedés).

## Vetítés tábla

Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
id 	int(11)			Nem	Nincs		AUTO_INCREMENT
Idopont	datetime(6)			Nem	Nincs		
Megjegyzes	longtext	utf8mb4_general_ci		Nem	Nincs		
Filmid 	int(11)			Nem	Nincs		
Teremid 	int(11)			Nem	Nincs		

A mozi vetítéseinek időpontjait, helyszíneit, műsorát rögzíti.

## Foglalt székek

Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
Vetitesid 	int(11)			Nem	Nincs		
X 	int(11)			Nem	Nincs		
Y 	int(11)			Nem	Nincs		
FoglalasAdatokid 	int(11)			Nem	Nincs		


Rögzíti, hogy ki melyik széket foglalta le és milyen típusú jeggyel. Kompozit kulcsát a hozzá tartozó vetítés szék kompozit kulcsa és a foglalás id-je teszi ki.

## VetítésSzékek tábla

Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
Vetitesid 	int(11)			Nem	Nincs		
X 	int(11)			Nem	Nincs		
Y 	int(11)			Nem	Nincs		
FoglalasAllapot	int(11)			Nem	Nincs		

Nyomon követi, hogy mely székek foglaltak vagy szabadok egy adott vetítésen. A felhasználó a szabad székek közül választhat (1-es kódú állapot – a 0-ás kód az elérhetetlen, a 2-es kód a már lefoglalt székeket jelöli). Kompozit kulcsát a hozzá tartozó vetítés id-je és a szék koordinátái teszik ki.

## Httplogs

Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
Id 	int(11)			Nem	Nincs		AUTO_INCREMENT
Schema	longtext	utf8mb4_general_ci		Nem	Nincs		
Host	longtext	utf8mb4_general_ci		Nem	Nincs		
Path	longtext	utf8mb4_general_ci		Nem	Nincs		
QueryString	longtext	utf8mb4_general_ci		Nem	Nincs		
RequestHeaders	longtext	utf8mb4_general_ci		Nem	Nincs		
RequestBody	longtext	utf8mb4_general_ci		Nem	Nincs		
ResponseHeaders	longtext	utf8mb4_general_ci		Nem	Nincs		
ResponseBody	longtext	utf8mb4_general_ci		Nem	Nincs		
StatusCode	int(11)			Nem	Nincs		
LogTime	datetime(6)			Nem	utc_timestamp()		


A rendszer működését naplózza (hibakereséshez, nyilvántartáshoz). A felhasználó számára láthatatlan, de segít a problémák gyors és hatékony megoldásában.

### Székek tábla

Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
X 	int(11)			Nem	Nincs		
Y 	int(11)			Nem	Nincs		
Teremid 	int(11)			Nem	Nincs		
Allapot	int(11)			Nem	Nincs		




A termék székeinek elrendezését tárolja. Ez adja a mintát a vetítések kiosztásának generálásához. Kompozit kulcsát a hozzá tartozó terem id-je és a szék koordinátái teszik ki.

### FoglalásAdatok tábla

Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
id 	int(11)			Nem	Nincs		AUTO_INCREMENT
FoglalasIdopontja	datetime(6)			Nem	Nincs		
UserID 	int(11)			Nem	Nincs		

A felhasználók által létrehozott foglalásokat tárolja.

## Users tábla (felhasználók)

Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
userID 	int(11)			Nem	Nincs		AUTO_INCREMENT
email 	varchar(100)	utf8mb4_general_ci		Nem	Nincs		
passwordHash	char(84)	utf8mb4_general_ci		Nem	Nincs		
creationDate	datetime			Nem	Nincs		
accountStatus	int(1)			Nem	Nincs		
role	varchar(30)	utf8mb4_general_ci		Nem	Nincs		
Megjegyzes	longtext	utf8mb4_general_ci		Nem	Nincs		
refreshToken 	varchar(100)	utf8mb4_general_ci		Igen	NULL		
refreshTokenExpiry	datetime			Igen	NULL		
EmailConfirmed	tinyint(1)			Nem	Nincs		
EmailConfirmationToken	longtext	utf8mb4_general_ci		Igen	NULL		
EmailConfirmationTokenExpiry	datetime(6)			Igen	NULL		
TwoFactorEnabled	tinyint(1)			Nem	Nincs		
PasswordResetToken	longtext	utf8mb4_general_ci		Igen	NULL		
PasswordResetTokenExpiry	datetime(6)			Igen	NULL		
PasswordResetRequired	tinyint(1)			Nem	Nincs		

A felhasználók adatait tárolja. A felhasználó ezzel a fiókkal tud bejelentkezni, jegyet foglalni, és személyre szabott élményt kap.

## Images tábla (képek)

Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
Id 	int(11)			Nem	Nincs		AUTO_INCREMENT
RelativePath	longtext	utf8mb4_general_ci		Nem	Nincs		
FileName	longtext	utf8mb4_general_ci		Nem	Nincs		
OriginalFileName	longtext	utf8mb4_general_ci		Nem	Nincs		
ContentType	longtext	utf8mb4_general_ci		Nem	Nincs		
FileSize	bigint(20)			Nem	Nincs		
UploadDate	datetime(6)			Nem	Nincs		

A filmekhez tartozó plakátok és egyéb képek adatait tárolja. A képek a szerveren vannak tárolva, amik futáskor a backenden, statikus úton keresztül érthetőek el.

## jegytípus tábla (jegyárak)

Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
id 	int(11)			Nem	Nincs		AUTO_INCREMENT
Nev	longtext	utf8mb4_general_ci		Nem	Nincs		
Ar	int(11)			Nem	Nincs		

Meghatározza a jegyek típusait és árait (pl. diák, felnőtt). A felhasználó ennek alapján választhat kedvezményes vagy normál jegyet.

### Email2FACodes tábla (kétlépcsős azonosítás kódok)

Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
Id 	int(11)			Nem	Nincs		AUTO_INCREMENT
Code	longtext	utf8mb4_general_ci		Nem	Nincs		
UserID 	int(11)			Nem	Nincs		
ExpiresAt	datetime(6)			Nem	Nincs		

Biztonsági kódokat tárol Email-es kétlépcsős azonosításhoz. A felhasználó ezzel erősítheti meg bejelentkezését.

### \_\_EFMigrationsHistory tábla (migrációs előzmények)

Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
MigrationId 	varchar(150)	utf8mb4_general_ci		Nem	Nincs		
ProductVersion	varchar(32)	utf8mb4_general_ci		Nem	Nincs		

Az EntityFrameworkCore automatikusan generált log táblája.

## Összefoglalás

A felhasználó a **users** táblán keresztül regisztrál, a **film** és **vetites** táblák alapján kiválaszt egy filmet és időpontot, majd a **szekek** és a **foglaltszekek** segítségével lefoglalja a helyét. A **jegytipus** meghatározza az árát, a **foglalasadatok** pedig tárolja a rendelését. Mindez a mozijegy-vásárlás zökkenőmentes élményét biztosítja.

## Kapcsolatok

### Felhasználó tábla kapcsolatai

**users ↔ email2facodes**

Kapcsolat típusa: 1:N (egy felhasználónak több kétlépcsős azonosító kódja lehet)

**users ↔ foglalasadatok**

Kapcsolat típusa: 1:N (egy felhasználó több foglalást hozhat létre)

### Film tábla kapcsolatai

**images ↔ film**

Kapcsolat típusa: 1:N (egy képhez több film is tartozhat, de egy filmnek csak egy képe van)

**film ↔ vetites**

Kapcsolat típusa: 1:N (egy filmhez több vetítés tartozhat)

### **Vetítés tábla kapcsolatai**

**terem ↔ vetites**

Kapcsolat típusa: N:1 (több vetítés történhet egy teremben)

**vetites ↔ vetitesszek**

Kapcsolat típusa: 1:N (egy vetítéshez több szék tartozhat)

### **Foglalás táblák kapcsolatai**

**foglalasadatok ↔ foglaltszek**

Kapcsolat típusa: 1:N (egy foglalás több széket tartalmazhat)

**foglaltszek ↔ vetitesszek**

Kapcsolat típusa: 1:1 (egy foglalt szék egy vetítés székére hivatkozik)

**foglaltszek ↔ jegytipus**

Kapcsolat típusa: N:1 (a foglalt székeknek van jegytípusa)

### **Terem tábla kapcsolatai**

**terem ↔ szek**

Kapcsolat típusa: 1:N (egy teremben több szék van)

**terem ↔ vetites**

Kapcsolat típusa: 1:N (egy teremben több vetítés történhet)

## 6. REST API funkciók

A rendszer REST API-t használ az adatok kezelésére. Az alábbi fő végpontok érhetőek el:

### Felhasználókezelés (AuthController)

POST /api/auth/login – Bejelentkezés

POST /api/auth/register – Regisztráció

GET /api/auth/confirm-email – Email cím megerősítése

POST /api/auth/verify-email-2fa – 2FA kód ellenőrzése

POST /api/auth/resent-2fa-code – 2FA kód újraküldése

POST /api/auth/enable-email-2fa – 2FA engedélyezése

POST /api/auth/disable-email-2fa – 2FA letiltása

GET /api/auth/getUser – Felhasználó adatainak lekérése (csak saját adatok)

GET /api/auth/getUserAdmin/{id} – Felhasználó adatainak lekérése (admin, bárki adatai)

GET /api/auth/get – Összes felhasználó lekérése (admin)

POST /api/auth/queryUsers – Felhasználók szűrése (admin, végül nem használt)

PATCH /api/auth/editUser – Felhasználó adatainak módosítása

PATCH /api/auth/editUserAdmin – Felhasználó adatainak módosítása(admin, végül nem használt)

PATCH /api/auth/editPassword – Jelszó módosítása

GET /api/auth/checkIfLoggedIn – Bejelentkezési állapot ellenőrzése

DELETE /api/auth/logout – Kijelentkezés

POST /api/auth/checkIfAdmin – Admin jogosultság ellenőrzése

DELETE /api/auth/deleteUser/{id} – Felhasználó törlése

PATCH /api/auth/change-password – Jelszó megváltoztatása

PATCH /api/auth/force-password-change/{id} – Jelszóváltoztatás kényszerítése (admin)

PATCH /api/auth/change-status/{id} – Felhasználó állapotának módosítása (admin)

POST /api/auth/request-password-reset/{userId} – Jelszó-visszaállítási kérés (admin)

GET /api/auth/verify-password-reset – Jelszó-visszaállítási token ellenőrzése

POST /api/auth/complete-password-reset – Jelszó visszaállítása

### **Filmek kezelése (FilmController)**

POST /api/film/query – Filmek szűrése (végül nem használt)

GET /api/film/get – Összes film lekérése

GET /api/film/get/{id} – Film lekérése ID alapján

POST /api/film/add – Új film hozzáadása (admin)

PATCH /api/film/edit – Film módosítása (admin)

DELETE /api/film/delete/{id} – Film törlése (admin)

### **Foglalások kezelése (FoglalasController)**

GET /api/foglalas/getJegyTipus – Jegytípusok lekérése

GET /api/foglalas/get – Összes foglalás lekérése (admin)

GET /api/foglalas/getByVetites/{vid} – Foglalások lekérése vetítés alapján

GET /api/foglalas/getByUser/{uid} – Foglalások lekérése felhasználó alapján

POST /api/foglalas/add – Új foglalás létrehozása

PATCH /api/foglalas/edit – Foglalás módosítása (végül nem használt)

DELETE /api/foglalas/delete/{id} – Foglalás törlése

### **Termek kezelése (TeremController)**

GET /api/terem/get – Összes terem lekérése



GET /api/terem/get/{id} – Terem lekérése ID alapján

POST /api/terem/add – Új terem hozzáadása (admin)

PATCH /api/terem/edit – Terem módosítása (admin)

DELETE /api/terem/delete/{id} – Terem törlése (admin)

### Vetítések kezelése (VetitesController)

GET /api/vetites/get – Összes vetítés lekérése

GET /api/vetites/get/{id} – Vetítés lekérése ID alapján

POST /api/vetites/add – Új vetítés hozzáadása (admin)

PATCH /api/vetites/edit – Vetítés módosítása (admin)

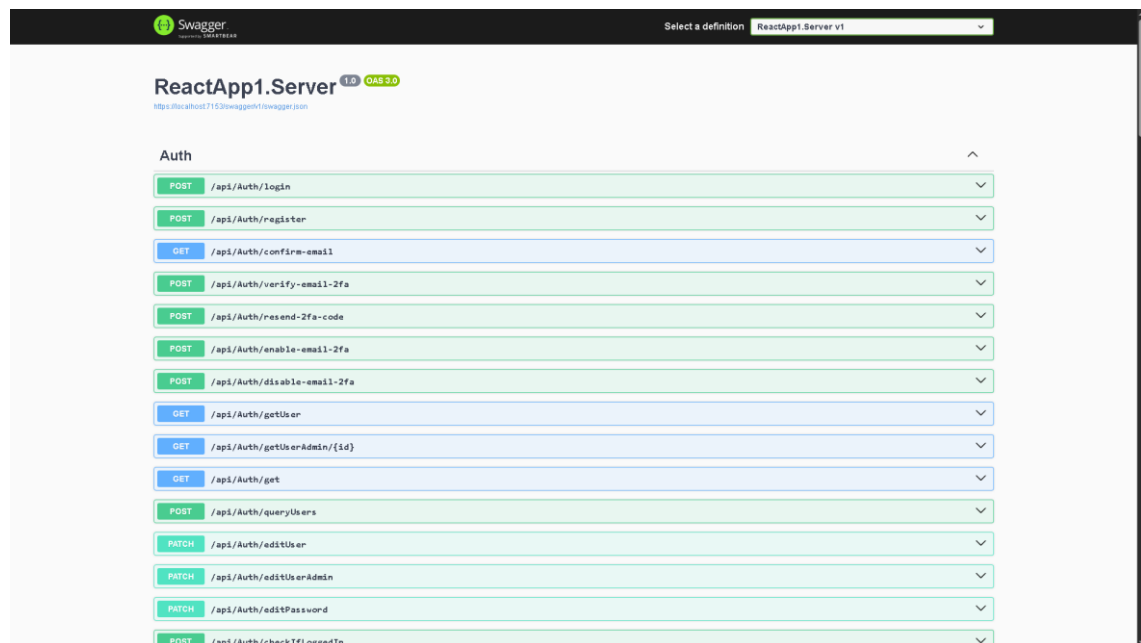
DELETE /api/vetites/delete/{id} – Vetítés törlése (admin)

### Képek kezelése (ImageController)

GET /api/images/get – Minden kép adatainak lekérése (admin)

POST /api/images/upload – Képfeltöltés (csak az api számára hozzáférhető

végpont)



(Képernyőkép a Swagger-ról, amivel teszteltük az API-t)

## 7. Weboldal működése

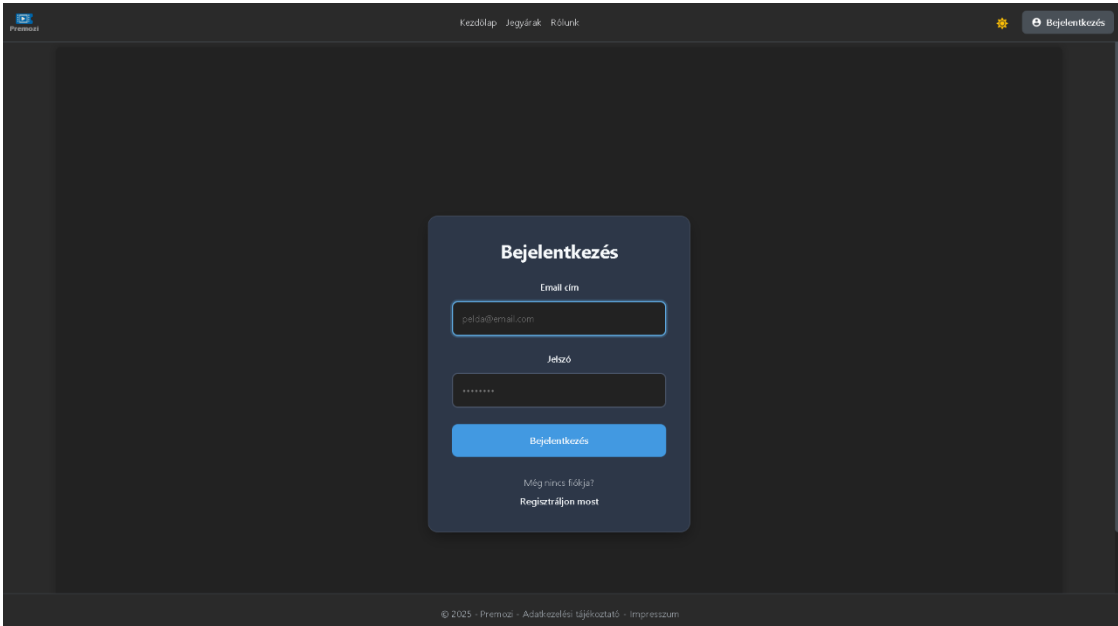
A weboldal egy reszponzív felhasználói felületet biztosít, amely az alábbi fő oldalakat tartalmazza:

- Bejelentkezés és regisztráció

Elérés út: <https://localhost:60769/account/login>

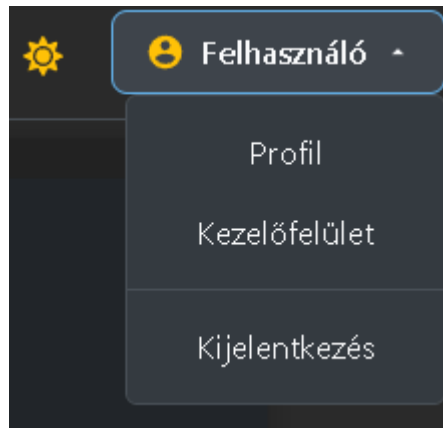
A felhasználó be tud jelentkezni, egy letisztult, bejelentkezési felületen.

Ha nincs fiókja, az alsó „Nincs fiókja? Regisztráljon most!” -ra kattintva, átirányítódik a regisztrációs felületre, ahol létrehozhatja fiókját, amit utána egy emailben kapott linkkel erősíthet meg.

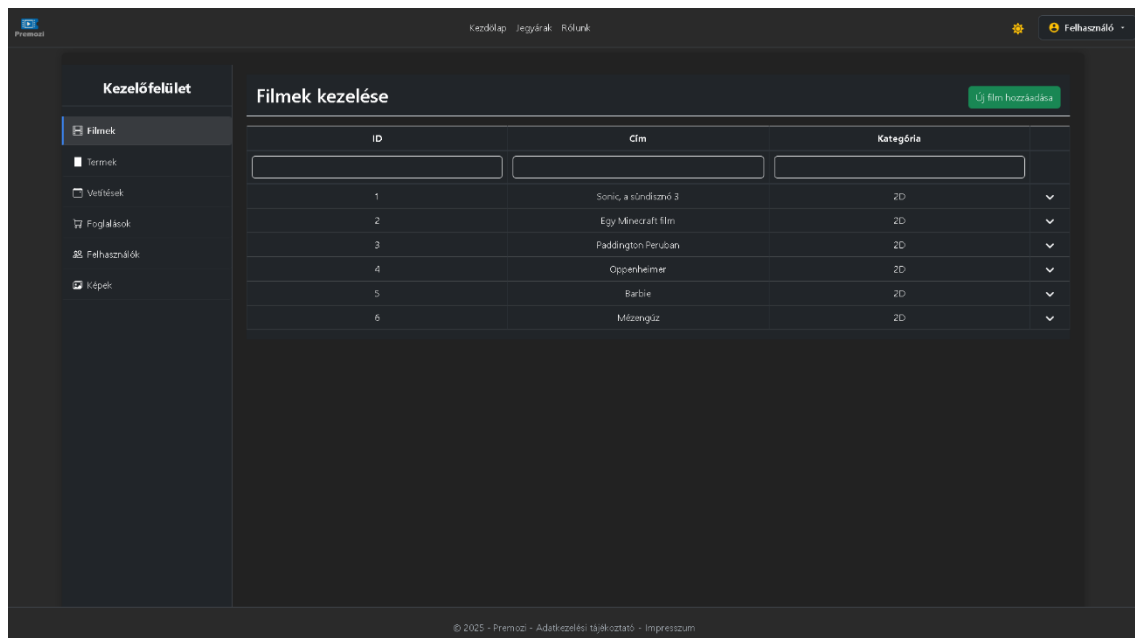


The screenshot displays a web application interface for login. At the top, a navigation bar includes links for 'Kezdőlap', 'Jegyzárak', and 'Rólunk', along with a 'Bejelentkezés' button. The main content area features a central login form titled 'Bejelentkezés'. This form contains two input fields: 'Email cím' (with the placeholder 'pelda@email.com') and 'Jelszó' (with a masked password '\*\*\*\*\*'). Below these fields is a blue 'Bejelentkezés' button. At the bottom of the form, there is a link that reads 'Még nincs fiókja? Regisztráljon most!'. The footer of the page contains the text '© 2025 - Premozi - Adatkezelési tájékoztató - Impresszum'.

- Ha admin fiókkal jelentkezünk be, az admin fiókkal a következők tehetőek:
  - Felhasználók kezelése/módosítása
  - adatok kezelése/módosítása/törlése

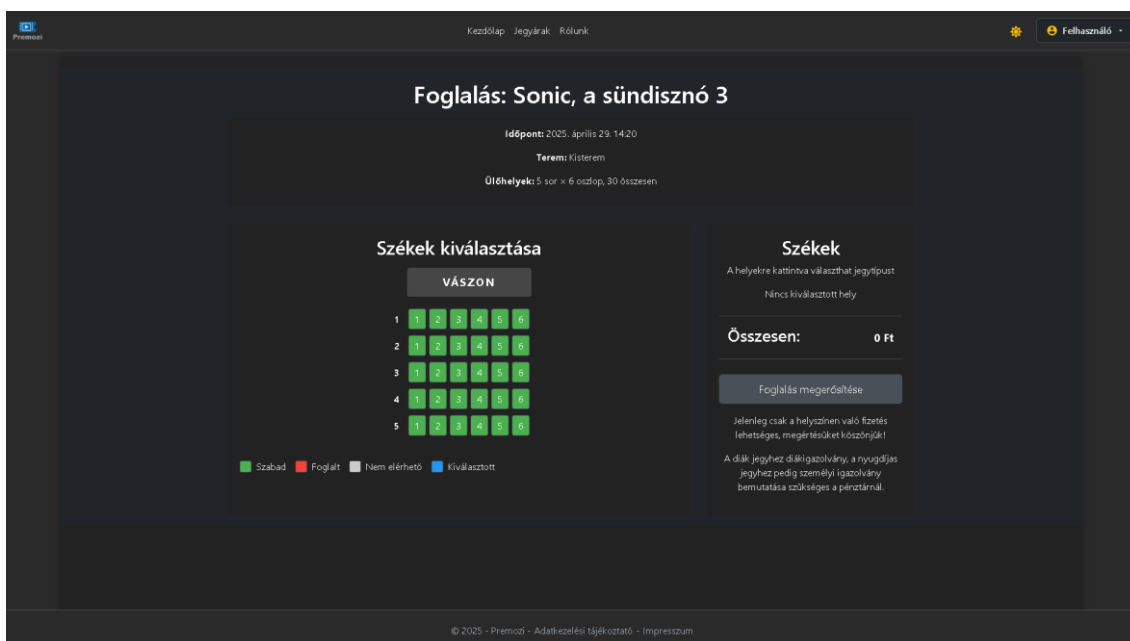
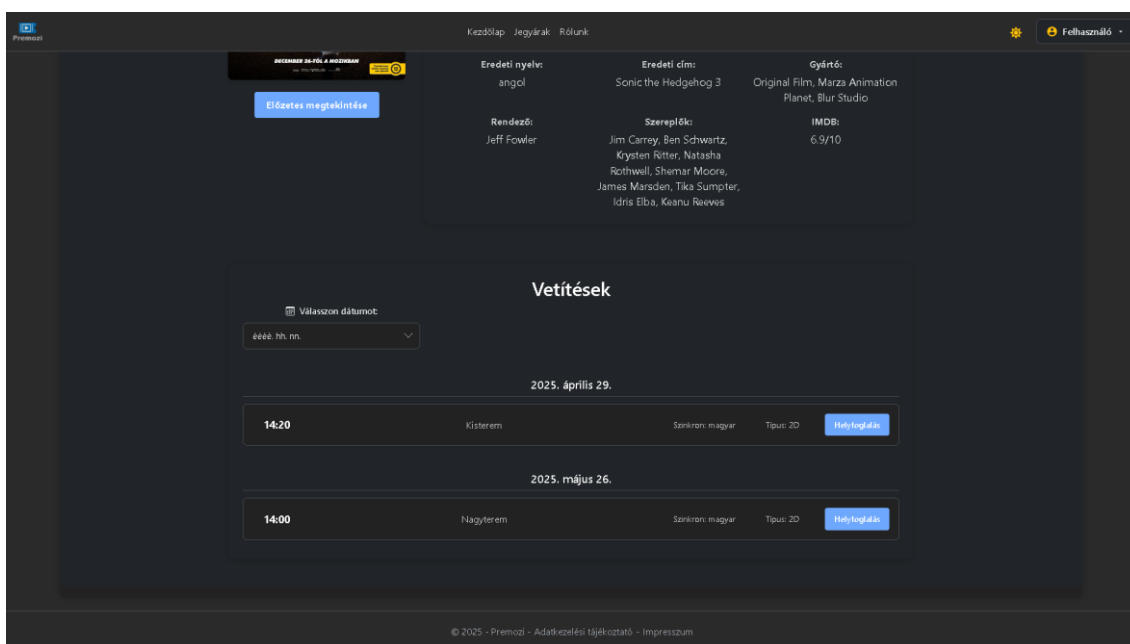


Ezeket a funkciókat egy csak az adminisztrátorok számára elérhető kezelőfelületen érhetőek el. A felhasználókat törölni nem tudja az adminisztrátor, de felfüggeszteni igen. A felhasználó maga tudja törölni fiókját.



- Helyfoglalás
  - A helyfoglaláshoz választani kell egy filmet a listán, majd rányomni a helyfoglalás gombra. Ez után megjelenik a helyfoglalás felület, ha be van jelentkezve – ha nincs, a bejelentkező felületre irányít át.

A helyfoglaláshoz választani kell egy több szék, szükség esetén beállítani a jegy típusát, majd rákattintani a foglalás gombra. Ezután egy sikeres foglalást jelző oldalra irányít át az oldal, ami 10 másodperc után visszairányít a kezdőlapra. Emellett a sikeres foglalásról emailt is kap a felhasználó.



Premiozi

Kézdőlap

Jegylárak

Rőlünk

Felhasználó

Foglalás: Sonic, a sündisznó 3

Időpont: 2025. április 29. 14:20

Terem: Kisterem

Ülőhelyek: 5 sor x 6 oszlop, 30 összesen

Székek kiválasztása

VÁSZON

1

1

2

3

4

5

6

2

1

2

3

4

5

6

3

1

2

3

4

5

6

4

1

2

3

4

5

6

5

1

2

3

4

5

6

Szabad

Foglalt

Nem elérhető

Kiválasztott

Székek

A helyekre kattintva válasszhat jegytípust

3. sor 3. szék

Normál (2000 Ft)

Diák (1500 Ft)

Nyugdíjas (1200 Ft)

3. sor 4. szék

Diák (1500 Ft)

Összesen: 3500 Ft

Foglalás megerősítése

Jelenleg csak a helyszínen való fizetés lehetséges, megértésüket köszönjük!

A diák jegyhez diákigazolvány, a nyugdíjas jegyhez pedig személyi igazolvány bemutatása szükséges a pénztárnál.

© 2025 - Premiozi - Adatkezelési tájékoztató - Impresszum

Premiozi

Kézdőlap

Jegylárak

Rőlünk

Felhasználó

Köszönjük a foglalását!

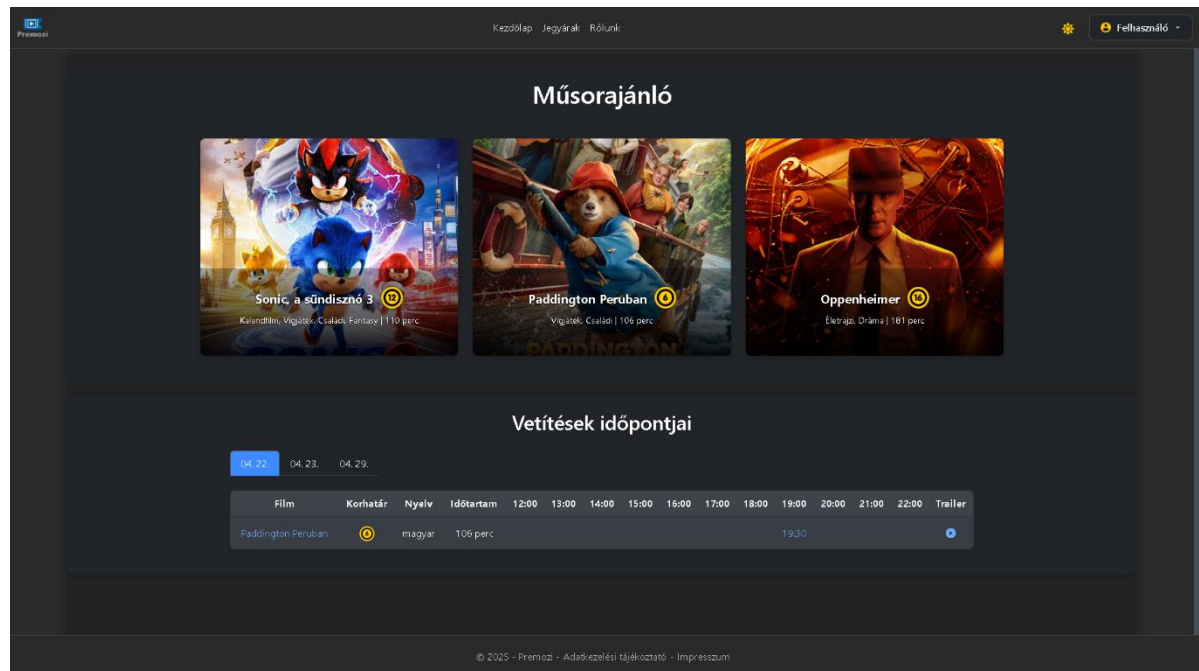
A foglalását sikeresen rögzítettük. A részleteket tartalmazó emailt elküldtük a megadott címre.

Átírányítás a főoldalra 9 másodperc múlva...

Vissza a főoldalra most

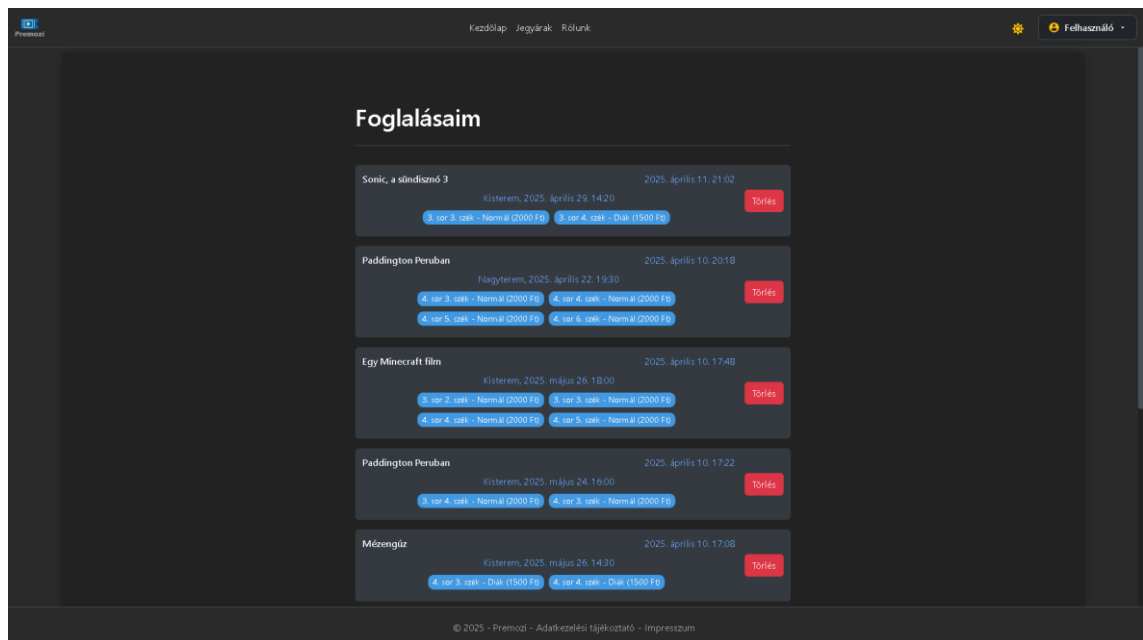
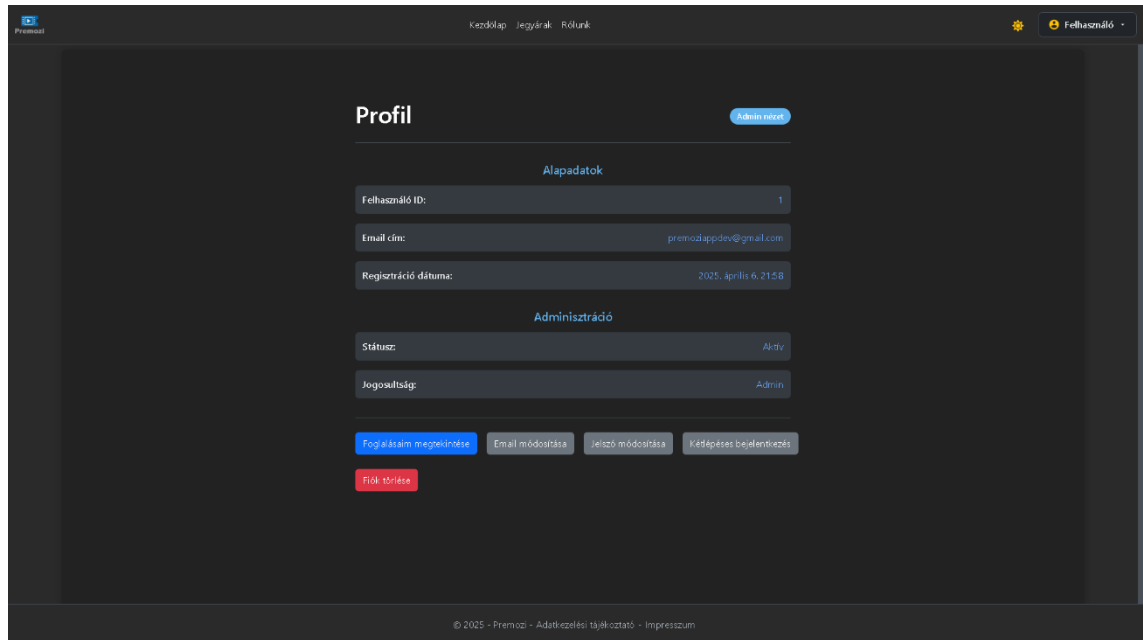
© 2025 - Premiozi - Adatkezelési tájékoztató - Impresszum

- Vetítés időpontok és filmek listázása
  - A Főoldalunkon megjelenik az összes 30 napon belüli vetés



- Filmmel kapcsolatos adatok megjelenítése
  - Ha kiválasztunk egy filmet, akkor megjelenítődnek a filmmel kapcsolatos adatok
- Helyfoglalás adott vetítésre
  - Ha kiválasztottunk egy filmet, akkor a film adatok alatt megjelennek a vetítés időpontok, ahol, ha kiválasztunk egy időpontot, tudunk helyet foglalni.

- Foglалások megjelenítése
  - A foglalt helyeinket megtudjuk nézni a profilunkon belül a „Foglalásaim megtekintése” gombnál



A weboldal az adatokat a REST API-on keresztül kéri le és tölti fel az adatbázisba.

## 8. NUnit tesztek

### AuthController tesztek

AuthControllerTests (5)	666 ms
✓ CheckIfLoggedIn_ReturnsCorrectState_WhenAuthenticated	17 ms
✓ ConfirmEmail_ReturnsOk_WhenTokenIsValid	26 ms
✓ DeleteUser_ReturnsForbidden_WhenNotAuthorized	1 ms
✓ GetUser_ReturnsUser_WhenAuthenticated	605 ms
✗ Register_ReturnsOk_WhenSuccessful	17 ms

#### Regisztráció tesztelése (Register\_ReturnsOk\_WhenSuccessful):

- Ellenőrzi, hogy a regisztráció sikeresen lefut-e és megfelelő választ ad-e vissza
- Mockolt szolgáltatásokkal szimulálja a regisztrációs folyamatot
- Ez hibával tér vissza, de valójában jó, mert létezik ilyen fiók.

#### Email megerősítés tesztelése (ConfirmEmail\_ReturnsOk\_WhenTokenIsValid):

- Ellenőrzi, hogy érvényes token esetén sikeresen megerősíti-e az email címet
- A token dekódolását és feldolgozását teszteli

#### Felhasználói adatok lekérése (GetUser\_ReturnsUser\_WhenAuthenticated):

- Hitelesített felhasználó esetén ellenőrzi a felhasználói adatok helyes visszaadását
- Claim-ek alapján történő azonosítást tesztel

#### Bejelentkezés állapotának ellenőrzése

##### (CheckIfLoggedIn\_ReturnsCorrectState\_WhenAuthenticated):

- Ellenőrzi, hogy a rendszer helyesen detektálja-e a bejelentkezés állapotát
- A hitelesítési állapot változásait figyeli

#### Felhasználó törlésének jogosultságvizsgálata

##### (DeleteUser\_ReturnsForbidden\_WhenNotAuthorized):

- Ellenőrzi, hogy nem admin felhasználó ne tudjon más felhasználót törölni
- Jogosultságkezelést tesztel



## FilmController tesztek

✓ FilmControllerTests (11)	27 ms
✓ AddFilm_ReturnsBadRequest_WhenUnsuccessful	1 ms
✓ AddFilm_ReturnsOkResult_WhenSuccessful	4 ms
✓ DeleteFilm_ReturnsBadRequest_WhenUnsuccessful	2 ms
✓ DeleteFilm_ReturnsOkResult_WhenSuccessful	1 ms
✓ EditFilm_ReturnsBadRequest_WhenUnsuccessful	2 ms
✓ EditFilm_ReturnsOkResult_WhenSuccessful	1 ms
✓ GetFilm_WithId_ReturnsBadRequest_WhenFilmDoesNotExist	2 ms
✓ GetFilm_WithId_ReturnsOkResult_WhenFilmExists	1 ms
✓ GetFilm_WithoutId_ReturnsOkResult	1 ms
✓ QueryFilm_ReturnsBadRequest_WhenServiceReturnsNull	1 ms
✓ QueryFilm_ReturnsOkResult_WithValidRequest	11 ms

## Filmlekérdezési tesztek

### QueryFilm\_ReturnsOkResult\_WithValidRequest

- Ellenőrzi, hogy a QueryFilm metódus helyesen visszaad-e egy filmlistát érvényes szűrőparaméterek esetén.
- Biztosítja, hogy a szűrt lekérdezések helyesen működjenek.

### QueryFilm\_ReturnsBadRequest\_WhenServiceReturnsNull

- Ellenőrzi a hibakezelést, ha a szolgáltatás null értéket ad vissza.
- Biztosítja, hogy a rendszer ne omoljon össze hibás adatok esetén.

### GetFilm\_WithoutId\_ReturnsOkResult

- Ellenőrzi, hogy a GetFilm metódus helyesen visszaad-e egy filmlistát, ha nincs ID megadva.
- Biztosítja, hogy az összes film lekérdezése helyesen működjön.

### GetFilm\_WithId\_ReturnsOkResult\_WhenFilmExists

- Ellenőrzi, hogy a GetFilm metódus helyesen visszaad-e egy filmet, ha az ID létezik.
- Biztosítja, hogy az egyedi filmlekérdezés helyesen működjön.

### GetFilm\_WithId\_ReturnsBadRequest\_WhenFilmDoesNotExist

- Ellenőrzi a hibakezelést, ha a film nem létezik.

- Biztosítja, hogy a rendszer ne omoljon össze, ha a film nem található.

## **Filmkezelési műveletek**

### **AddFilm\_ReturnsOkResult\_WhenSuccessful**

- Ellenőrzi, hogy a AddFilm metódus helyesen visszaad-e egy sikeres üzenetet, ha a film hozzáadása sikerült.
- Biztosítja, hogy a film hozzáadása helyesen működjön.

### **AddFilm\_ReturnsBadRequest\_WhenUnsuccessful**

- Ellenőrzi a hibakezelést, ha a film hozzáadása sikertelen.
- Biztosítja, hogy a rendszer helyesen kezelje a hibákat.

### **EditFilm\_ReturnsOkResult\_WhenSuccessful**

- Ellenőrzi, hogy a EditFilm metódus helyesen visszaad-e egy sikeres üzenetet, ha a film módosítása sikerült.
- Biztosítja, hogy a film módosítása helyesen működjön.

### **EditFilm\_ReturnsBadRequest\_WhenUnsuccessful**

- Ellenőrzi a hibakezelést, ha a film módosítása sikertelen.
- Biztosítja, hogy a rendszer helyesen kezelje a hibákat.

### **DeleteFilm\_ReturnsOkResult\_WhenSuccessful**

- Ellenőrzi, hogy a DeleteFilm metódus helyesen visszaad-e egy sikeres üzenetet, ha a film törlése sikerült.
- Biztosítja, hogy a film törlése helyesen működjön.

### **DeleteFilm\_ReturnsBadRequest\_WhenUnsuccessful**

- Ellenőrzi a hibakezelést, ha a film törlése sikertelen.
- Biztosítja, hogy a rendszer helyesen kezelje a hibákat.

## FoglalasController tesztek

✓ FoglalasControllerTests (12)	23 ms
✓ AddFoglalas_ReturnsBadRequest_WhenUnsuccessful	6 ms
✓ AddFoglalas_ReturnsOkResult_WhenSuccessful	2 ms
✓ DeleteFoglalas_ReturnsBadRequest_WhenUnsuccessful	2 ms
✓ DeleteFoglalas_ReturnsOkResult_WhenSuccessful	1 ms
✓ EditFoglalas_ReturnsBadRequest_WhenUnsuccessful	1 ms
✓ EditFoglalas_ReturnsOkResult_WhenSuccessful	1 ms
✓ GetFoglalas_AdminRole_ReturnsListOfFoglalas	1 ms
✓ GetFoglalasByUser_ReturnsBadRequest_WhenNotFound	2 ms
✓ GetFoglalasByUser_ReturnsListOfFoglalas_WhenExists	1 ms
✓ GetFoglalasByVetites_ReturnsBadRequest_WhenNotFound	1 ms
✓ GetFoglalasByVetites_ReturnsListOfFoglalas_WhenExists	2 ms
✓ GetJegyTipus_ReturnsListOfJegyTipus	3 ms

## Jegytípus kezelés tesztei

### GetJegyTipus\_ReturnsListOfJegyTipus

- Jegytípusok listájának lekérdezésének tesztelése

## Foglalás lekérdezési tesztek

### GetFoglalas\_AdminRole\_ReturnsListOfFoglalas

- Admin jogosultságú foglaláslistázás tesztelése
- Ellenőrzi, hogy megfelelően kapjuk-e vissza a foglalásokat

### GetFoglalasByVetites\_ReturnsListOfFoglalas\_WhenExists

- Vetítéshez tartozó foglalások lekérdezése
- Ellenőrzi, hogy nem üres lista érkezik-e vissza

### GetFoglalasByVetites\_ReturnsBadRequest\_WhenNotFound

- Nem létező vetítéshez tartozó foglalások kezelése
- Ellenőrzi a BadRequestObjectResult visszakapását

### GetFoglalasByUser\_ReturnsListOfFoglalas\_WhenExists

- Felhasználóhoz tartozó foglalások lekérdezése

- Ellenőrzi a felhasználó foglalási listát

#### **GetFoglalasByUser\_ReturnsBadRequest\_WhenNotFound**

- Nem létező felhasználói foglalások kezelése
- Ellenőrzi a BadRequestObjectResult-t

### **CRUD műveletek tesztjei**

#### **AddFoglalas\_ReturnsOkResult\_WhenSuccessful**

- Foglalás létrehozásának tesztelése
- Ellenőrzi az OkObjectResult-ot a sikerüzenettel

#### **AddFoglalas\_ReturnsBadRequest\_WhenUnsuccessful**

- Cél: Sikertelen foglalás készítés kezelése
- Ellenőrzi a BadRequestObjectResult-ot

#### **EditFoglalas\_ReturnsOkResult\_WhenSuccessful**

- Foglalás módosításának tesztelése
- Ellenőrzi a sikerüzenetet a válaszban

#### **EditFoglalas\_ReturnsBadRequest\_WhenUnsuccessful**

- Sikertelen módosítás kezelése
- Ellenőrzi a BadRequestObjectResult-ot

#### **DeleteFoglalas\_ReturnsOkResult\_WhenSuccessful**

- Foglalás törlésének tesztelése
- Ellenőrzi a sikerüzenetet

#### **DeleteFoglalas\_ReturnsBadRequest\_WhenUnsuccessful**

- Sikertelen törlés kezelése
- Ellenőrzi a BadRequestObjectResult-ot

## TeremController tesztek

TeremControllerTests (2)	14 ms
DeleteTerem_ReturnsBadRequest_WhenUnsuccessful	5 ms
DeleteTerem_ReturnsOk_WhenSuccessful	9 ms

### DeleteTerem\_ReturnsOk\_WhenSuccessful

- Sikeres terem törlésének tesztelése
- Ellenőrzi, a visszatérési érték típusát
- Ellenőrzi, hogy az eredmény OkObjectResult típusú
- Összehasonlítja, hogy a visszaadott üzenet megegyezik-e a mockolt sikerüzenettel

### DeleteTerem\_ReturnsBadRequest\_WhenUnsuccessful

- Sikertelen terem törlésének tesztelése
- Ellenőrzi, hogy a visszatérési érték típusa ActionResult<ErrorModel>
- Ellenőrzi, hogy az eredmény BadRequestObjectResult típusú

## VetitesController tesztek

VetitesControllerTests (4)	14 ms
DeleteVetites_ReturnsBadRequest_WhenUnsuccessful	3 ms
DeleteVetites_ReturnsOk_WhenSuccessful	4 ms
EditVetites_ReturnsBadRequest_WhenUnsuccessful	1 ms
EditVetites_ReturnsOk_WhenSuccessful	6 ms

## Módosítási műveletek tesztjei

### EditVetites\_ReturnsOk\_WhenSuccessful

- Sikeres vetítesmódosítás tesztelése
- Ellenőrzi a visszatérési érték típusát (ActionResult<ErrorModel>)
- Ellenőrzi, hogy az eredmény OkObjectResult típusú
- Ellenőrzi a visszaadott hibaüzenet tartalmát

### EditVetites\_ReturnsBadRequest\_WhenUnsuccessful

- Sikertelen vetítésmódosítás tesztelése
- Ellenőrzi a visszatérési érték típusát
- Ellenőrzi, hogy az eredmény BadRequestObjectResult típusú
- Ellenőrzi a hibaüzenet tartalmát

## **Törlési műveletek tesztjei**

### **DeleteVetites\_ReturnsOk\_WhenSuccessful**

- **Cél:** Sikeres vetítéstörlés tesztelése
- **Előkészítés:**
  - Mockolja a deleteVetites metódust, hogy "Sikeres törlés" üzenettel térjen vissza vetítés ID=1 esetén
- **Végrehajtás:**
  - Meghívja a controller DeleteVetites metódusát vetítés ID=1 paraméterrel
- **Ellenőrzés:**
  - Ellenőrzi a visszatérési érték típusát
  - Ellenőrzi, hogy az eredmény OkObjectResult típusú
  - Ellenőrzi a sikerüzenet tartalmát

### **DeleteVetites\_ReturnsBadRequest\_WhenUnsuccessful**

- Sikertelen vetítéstörlés tesztelése
- Ellenőrzi a visszatérési érték típusát
- Ellenőrzi, hogy az eredmény BadRequestObjectResult típusú
- Ellenőrzi a hibaüzenet tartalmát

## 9. Fejlesztési javaslatok

- A backend és a lekérdezések optimalizálása és biztonságosabbá tétele
- Web szerverre való feltöltés és futtatás (folyamatban, ha sikeres, akkor a <https://premozi.site> oldalon lesz elérhető a projekt)
- Fizetési lehetőség
- Frontend optimalizálása, a design javítása
- Adatbázis optimalizálása
- Docker konfigurálása és az onnan való futtatás

## 10.Összegzés/Reflexió

A mozijegy-foglaló webalkalmazás fejlesztése során számos kihívással és tanulsággal gazdagodtunk. A projekt nem csupán technikai ismeretek bővítését jelentette számunkra, hanem lehetőséget adott a csapatmunka és a problémamegoldó készségeink fejlesztésére is. Az alkalmazás elkészítése során sikerült egy olyan rendszert létrehoznunk, amely lehetővé teszi a felhasználók számára a mozijegyek gyors és kényelmes foglalását, miközben biztonságos és hatékony adatkezelést biztosít.

A projekt során kiemelt figyelmet fordítottunk az adatbázis tervezésére és a REST API implementálására. Az Entity Framework és a MySQL kombinációja kiválóan bizonyított az adatok kezelésében, míg a JWT token-alapú hitelesítés lehetővé tette a biztonságos felhasználókezelést. A frontend React és Vite segítségével készült, ami reszponzív és felhasználóbarát felületet eredményezett. A tesztelés során az NUnit keretrendszert használtuk, ami segített a kód megbízhatóságának és stabilitásának biztosításában.

A csapatmunka kulcsfontosságú volt a projekt sikerében. Minden csapattag a saját erősségei alapján vállalt feladatokat, és a rendszeres megbeszélések lehetővé tették a problémák gyors megoldását. A kommunikációhoz a Discord-ot, a verziókövetéshez pedig a GitHub-ot használtuk, ami nagyban hozzájárult a munka hatékonyságához. A ProtonVPN segítségével pedig akadálytalanul tudtunk elérni külső forrásokat, mint például a StackOverflow-t vagy a W3Schools-t.

A projekt során felmerülő nehézségek, például az adatbázis normalizálása vagy a kétlépcsős hitelesítés implementálása, lehetőséget adtak a kreatív problémamegoldásra és a tanulásra. Ezek a tapasztalatok értékesebbé tették számunkra a fejlesztési folyamatot.

Összességében elmondható, hogy a projekt sikeresen teljesítette a kitűzött célokat, és egy jól működő, felhasználóbarát alkalmazást eredményezett. A jövőben további fejlesztésekkel, például egy adminisztrációs felülettel vagy értesítési rendszerrel, még jobbra tehető a rendszer. A projekt nem csupán technikai ismereteket adott, hanem megerősítette bennünket abban, hogy a csapatmunka és a kitartás kulcsfontosságú a sikerhez.

## **11. Források, segítség**

- ChatGPT. Deepseek, Cody
- W3Schools
- Frostbyte
- Stackoverflow
- React dokumentáció (19.1)
- Youtube videók
- Egyéb, a programkódban, kommentben jelzett források