Webes Alkalmazások Fejlesztése

1. beadandó/9.1. feladat

Gonda Dávid
BIXU0S
gonda.david18@gmail.com

Feladat

Mozi

Készítsünk egy mozi üzemeltető rendszert, amely alkalmas az előadások, illetve jegyvásárlások kezelésére.
1. részfeladat: a webes felületen keresztül a nézők tekinthetik meg a moziműsort, valamint rendelhetnek jegyeket.
• A főoldalon megjelenik a napi program, azaz mely filmeket mikor vetítik a moziban, valamint kiemelve az öt legfrissebb (legutoljára felvitt) film plakátja.
• A filmet kiválasztva megjelenik annak részletes leírása (rendező, főszereplők, hossz, szinopszis), plakátja, továbbá az összes előadás időpontja.
• Az időpontot kiválasztva lehetőség nyílik helyfoglalásra az adott előadásra. Ekkor a felhasználónak meg kell adnia a lefoglalandó ülések helyzetét (sor, illetve oszlop) egy, a mozitermet sematikusan ábrázoló grafikus felületen. Egyszerre legfeljebb 6 jegy foglalható, és természetesen csak a szabad helyek foglalhatóak (amelyek nem foglaltak, vagy eladottak). A felhasználónak ezen felül meg kell adnia teljes nevét, valamint telefonszámát, ezzel véglegesíti a foglalást.

2. részfeladat: az asztali grafikus felületet az alkalmazottak használják a mozipénztárakban az előadások meghirdetésére, illetve jegyek kiadására.
• Az alkalmazott bejelentkezhet (felhasználónév és jelszó megadásával) a programba, illetve kijelentkezhet.
• Új film felvitelekor ki kell tölteni a film adatait (cím, rendező, főszereplők, hossz, szinopszis), valamint feltölthetünk egy képet plakátként.
• Új előadás meghirdetéséhez a felhasználónak ki kell választania a termet, valamint a filmet, és az időpont megadásával hirdetheti meg az előadást. A meghirdetéskor ügyelni kell arra, hogy az előadás ne ütközzön más előadásokkal az adott teremben (figyelembe véve a kezdés időpontját, illetve a film hosszát), illetve két előadás között legalább 15 percnek kell eltelnie a takarítás végett.
• A jegyvásárláshoz ki kell választani a filmet és az előadást. Ezt követően listázódnak a helyek (sor, oszlop, státusz). A szabad, illetve foglalt helyek eladhatóak, illetve a foglalt helyeket kiválasztva meg lehet tekinteni a foglaló adatait (név, telefonszám).

Az adatbázis az alábbi adatokat tárolja:
• filmek (cím, rendező, szinopszis, hossz, plakát, bevitel dátuma);
• termek (név, sorok száma, oszlopok száma);
• előadások (film, kezdő időpont, terem);

• helyek (előadás, terem, sor, oszlop, státusz , foglaló neve, foglaló telefonszáma);
• alkalmazottak (teljes név, felhasználónév, jelszó).

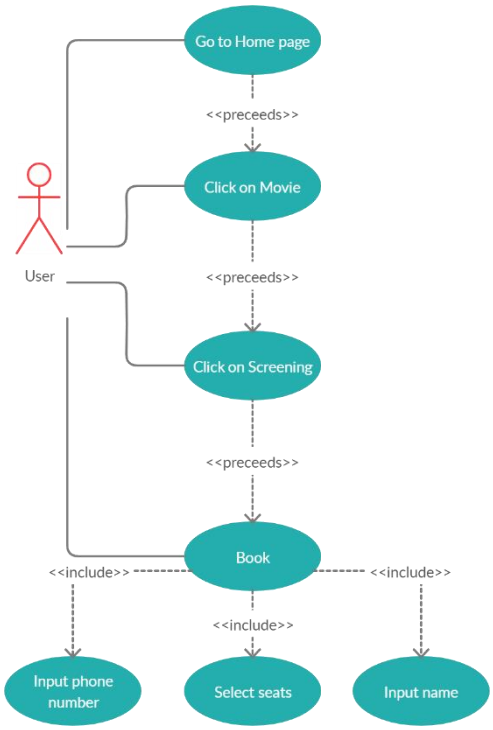
Elemzés

- Az oldal 3 fő oldalból fog állni: egy **Főoldalból**, ahol kilistázzuk a napi programokat; egy **Film** oldalból, ahol a kiválasztott film adatait, illetve az előadások időpontját/termét lehet látni/kiválasztani; és egy **Foglalási** odlalból, itt megjelenítjük a termet, amin kattintással lehet kiválasztani a foglalt helyeket, majd pedig név és telefonszám megadásával után lehet megerősíteni a foglalást.
- A feladatot *ASP.NET Core*-ban valósítjuk meg, *MVC* architektúrában.
- A foglalás után átirányítjuk a felhasználót egy **Siker** oldalra, ahonnan 4 másodperc után visszairányítjuk a főoldalra.
- A vetítéseket, termeket, üléseket, és filmeket egy adatbázisban fogjuk tárolni.
- Az adatbázis számunkra 4 fontosabb táblából fog állni: **Seats**, itt tároljuk vetítésekre bontva a székeket, amiknek megadjuk a vetítés azonosítóját, a terem azonosítóját, a teremben való elhelyezkedésüket, az aktuális állapotukat (*Szabad, Foglalt, Eladva*), a foglaló nevét és számát, a két utóbbi kezdedben *NULL,* ezek cask foglalás esetén adhatóak meg; **Screenings**, itt tároljuk a film azonosítóját, a kezdőidőpontot, valamint a terem azonosítóját; **Movies,** itt tároljuk a film címét, a rendező nevét, egy rövid szinopszist, a film hosszát, a plakátot, valamint a film bevitelének dátumát; **Rooms,** itt tároljuk a termek nevét, a sorok/oszlopok számát.
- Még egy táblát kell megemlítenünk, mégpedig az **Employees** táblát, de erről nem ejtenék most szót, mivel ez csak a következő részfeladatban less aktuális.

-

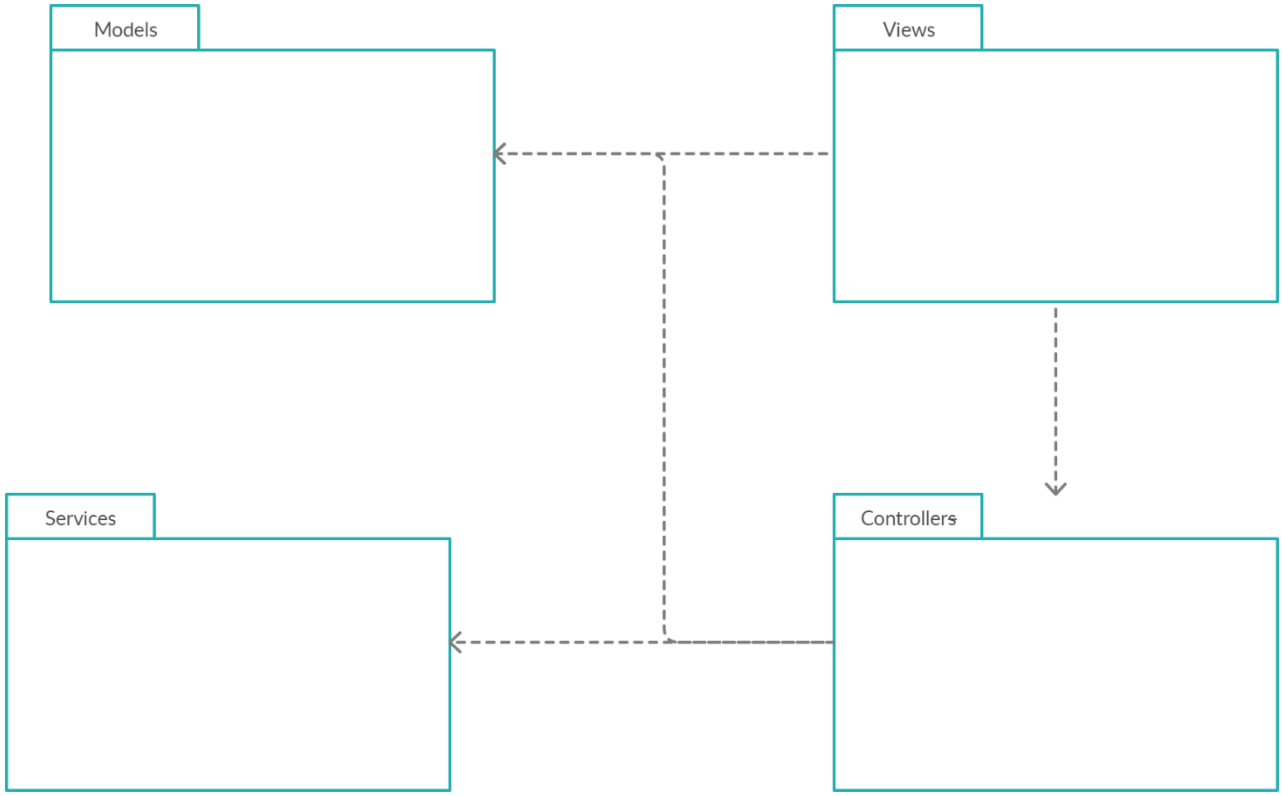
Felhasználói esetek

- A felhasználó akármikor a *Cinema* linker kattintva visszamehet a kezdőoldalra.
- Rákattinthat egy vetítésre, ez átirányítja az adott film oldalára.
- A film oldalán kiválaszthat egy specifikus vetítést, ez átirányítja a foglalási oldalra.
- A foglalási oldalon kiválaszthat legfelljebb 6 szabad ülést, beírhatja a nevét/telefonszámát, majd rákattinthat a *Book* gombra, ami lefoglalja a kiválasztott üléseket.



1. Use Case diagram

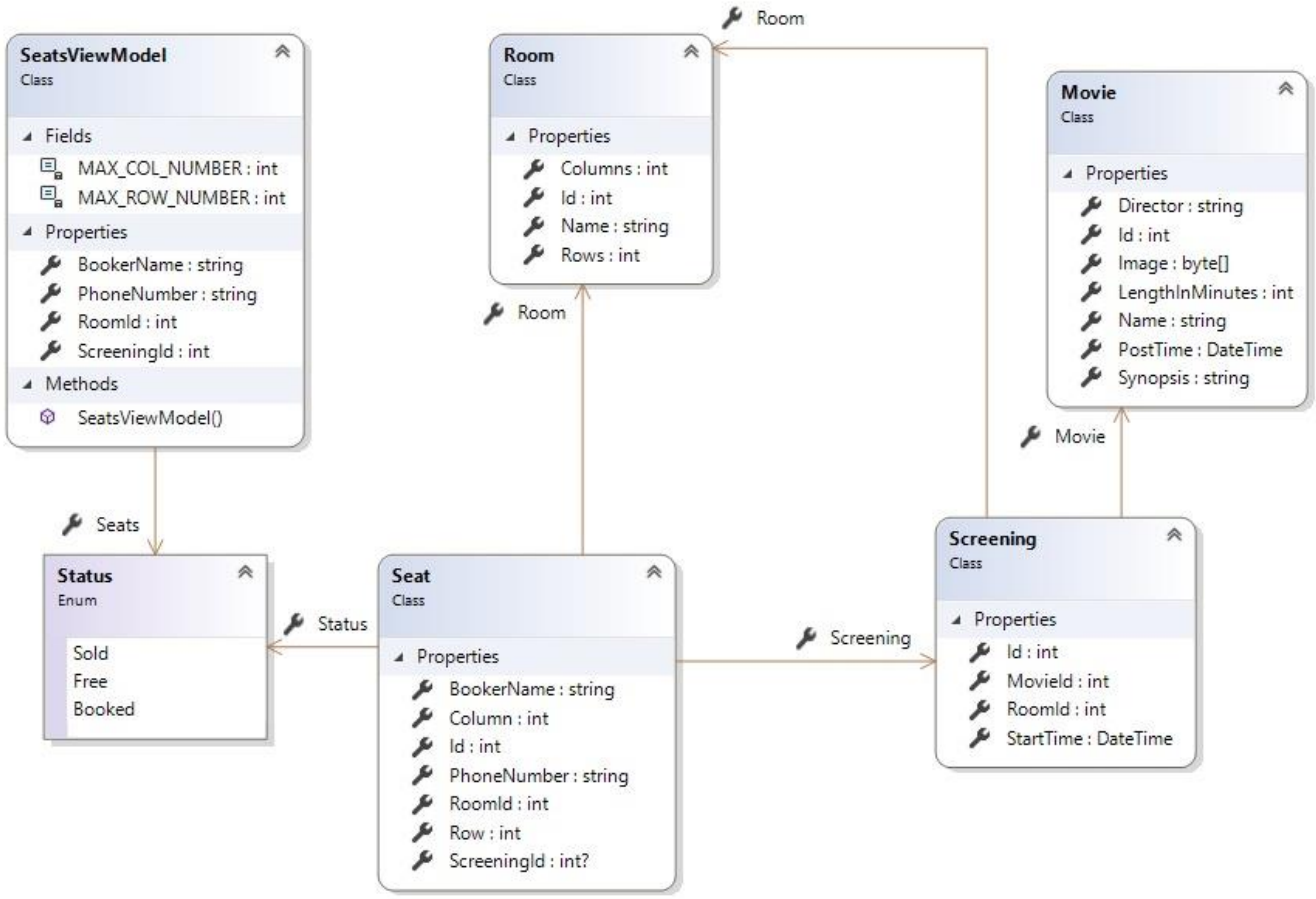
Osztálydiagram



2. Komponens diagram

A program 4 nagyobb egységből áll, egy *Controllers*ből, egy *Models*ből, egy *Views*ból, és egy *Services* rétegből.

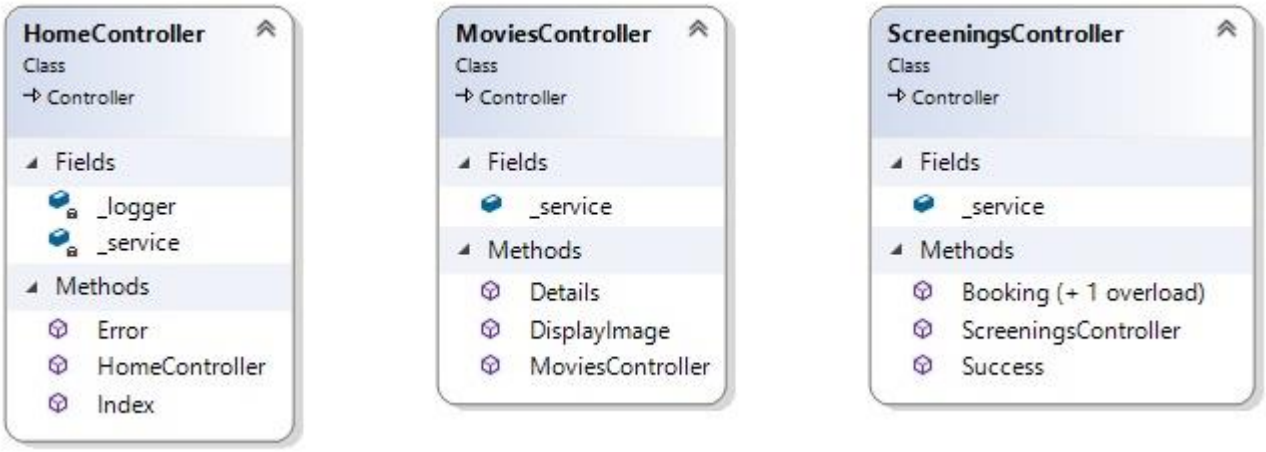
- **Model:**



3. Model class diagram

A *modell* tartalmazza az adatok/entity-k reprezentációját.
A *Room* osztály tartalmazza a termek tulajdonságait.
A *Movie* osztály tartalmazza a filmek tulajdonságait.
A *Seat* osztály tartalmazza az ülések tulajdonságait, aminek a *Status* tulajdonságát egy **Status** enumerátoral írjuk le.
A *Screening* pedig a vetítések tulajdonságát tartalmazza.
A *SeatViewModel* nézetmodell segíti a foglalási adatok helyes befogadását a felhasználótól. Itt írjuk le a bemenetek formai követelményeit.

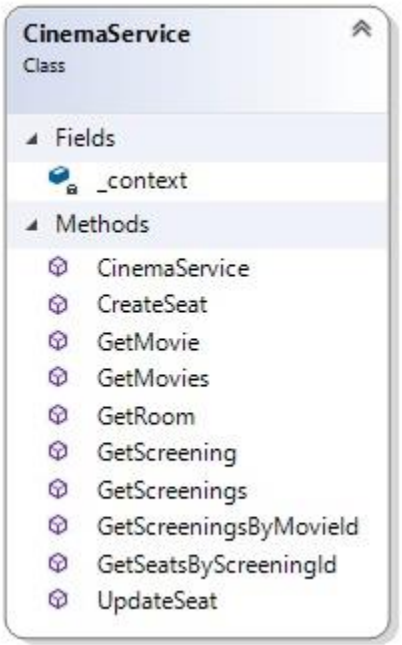
- **Controllers:**



4. Controller class diagram

A vezérlők, amelyek elérést adnak a weboldalhoz.
A *HomeController* jeleníti meg a napi programot, illetve az 5 legfrissebb film plakátját.
A *MoviesController* jeleníti meg egy film tulajdonságait, illetve sorolja fel az összes vetítési időpontot.
A *ScreeningsController* valósítja meg a foglalást egy adott vetítésre.

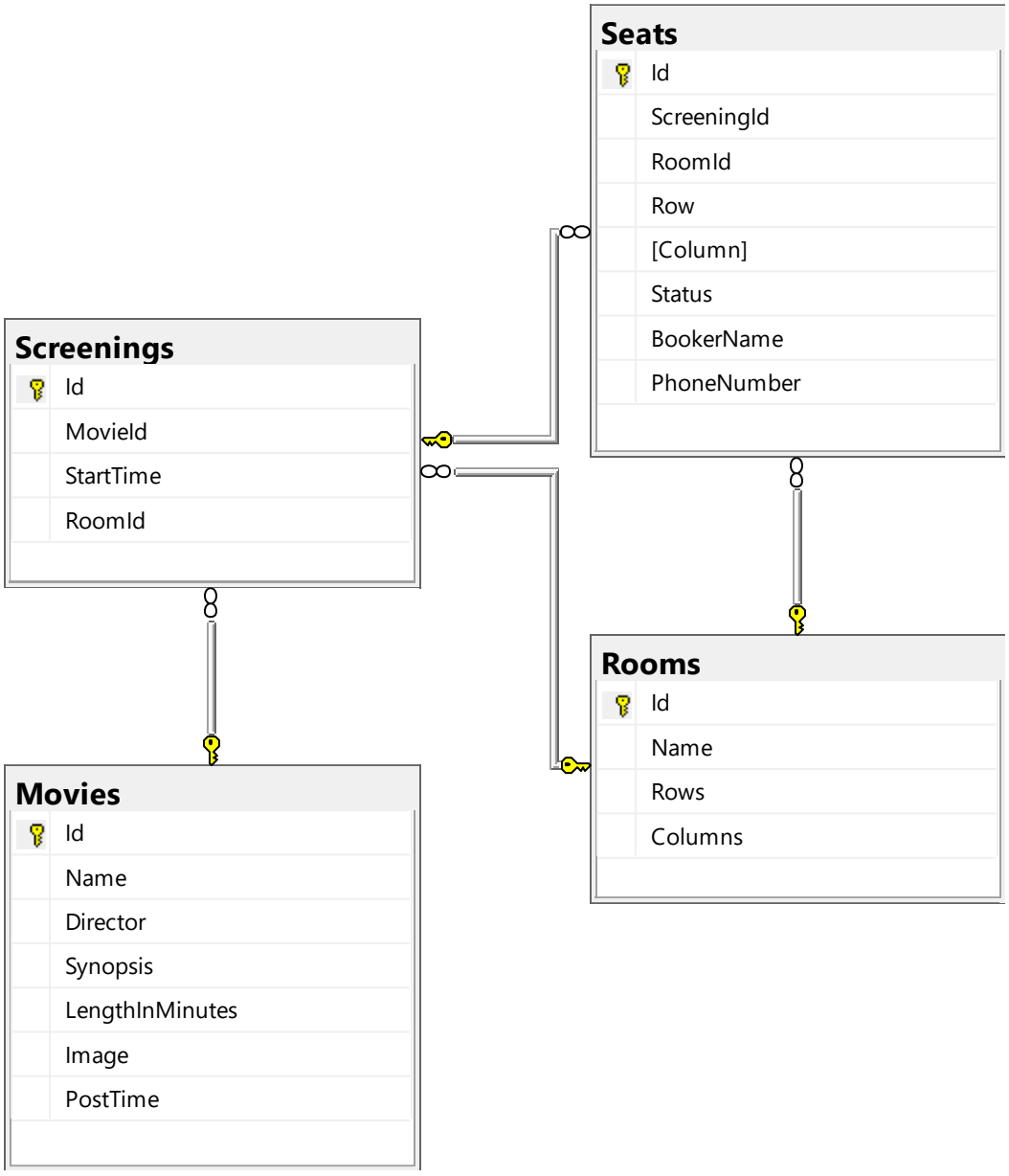
- **Services:**



5. Services class diagram

Az adatbázisoperációkat valósítja meg.

Adatbázis



6. Adatbázis entity relationship diagram

Az adatbázisunk négy táblából áll (öt, de az egyiket az első részfeladatban nem használjuk.

A táblák a már fentebb leírt entitásmodellek alapján tárolják az adatokat.