

Szegedi Szakképzési Centrum  
Vasvári Pál Gazdasági és Informatikai  
Szakgimnáziuma

Szoftverfejlesztő OKJ 54 213 05 szakképesítés

Szakdolgozat

**Filmes adatbázis – filmkeresés adatbázisban és rendelés**

Készítette: Pónusz Richárd

Témavezető:

Bálint Róbert

Témavezető:

Gyuris Csaba



Szeged

2018

## Tartalom

Bevezetés.....	3
I. Fejlesztői dokumentáció.....	4
I.1. A fejlesztés eszközeiről.....	4
I.1.1. Az adatbázis.....	4
I.1.2. Az asztali Alkalmazás .....	4
I.1.3 A weboldal .....	5
I.1.4 Egyéb felhasznált eszközök .....	6
I.2 – A program felépítéséről.....	6
I.2.1 – Az adatbázis szerkezete.....	7
I.2.1.1 – A Movies tábla .....	9
I.2.1.2 – A Directors tábla .....	9
I.2.1.3 – A Users tábla.....	10
I.2.1.4 – A movieorder tábla.....	10
I.2.2 – Az asztali program .....	11
I.2.2.1 – Asztali alkalmazás: Login .....	11
I.2.2.2 - Asztali alkalmazás: Movies – Movie Tábla .....	11
I.2.2.3 - Asztali alkalmazás: Movies – Edit Tábla .....	12
I.2.2.4 - Asztali alkalmazás: Users tábla.....	12
I.2.2.5 - Asztali alkalmazás: Order Tábla.....	13
I.2.2.6 – Asztali Alkalmazás: Unit Teszt .....	13
I.2.3 – A webes Alkalmazás .....	14
I.2.3.1. – Webes Alkamazás: Az Index oldal .....	15
I.2.3.3. – Webes Alkamazás: Az About Us oldal .....	15
I.2.3.4 – Webes Alkalmazás: Registration, Login & Logout .....	15
I.2.3.5 – Webes Alkalmazás: Movie.php.....	16
I.2.3.6 – Webes Alkalmazás: basket.php, Order.php, Functions.php etc.....	18
I.2.3.7 – Webes Alkalmazás: Connect.php .....	19
I.3. Egyéb információk.....	19
I.3.1 Továbbfejlesztési javaslatok.....	19
I.3.2 Fejlesztés során történt változások.....	19
II. Felhasználói dokumentáció .....	20
II.1 A Program alapvető funkciói .....	20
II.1.2 A Program felhasználói szintjei .....	21

II.2 – Használati Útmutató.....	21
II.2.1. – Felhasználói Dokumentáció: Az Asztali Alkalmazás.....	21
II.2.1.1. – Asztali alkalmazás: Login .....	22
II.2.1.2 – Asztali Alkalmazás: Movies.....	22
II.2.1.3 – Asztali Alkalmazás: Edit Movies.....	24
II.2.1.4 – Asztali Alkalmazás: Users.....	24
II.2.1.5 – Asztali Alkalmazás: Orders.....	25
II.2.1.6 – Asztali Alkalmazás: About Us & Exit .....	25
II.2.2 – Felhasználói Dokumentáció: Webes alkalmazás .....	25
II.2.2.1. – Webes Alkalmazás: Fő oldal és AboutUs oldal .....	25
II.2.2.2. – Regisztráció és Login.....	26
II.2.2.3 – Webes Alkalmazás: Search Page, Basket Page .....	27
II.3 – Rendszerkövetelmény .....	27
III. Forrásmegjelölés .....	28
III.1 – Weboldalak, programkódok, szöveges források megjelölése .....	28
III.2 – Képek forrásainak megjelölése.....	28
III.2.1 – Az adatbázisban tárolt képek .....	28
III.2.2. – Az asztali alkalmazáshoz használt képek .....	28
III.2.3 – A webes alkalmazáshoz használt képek: .....	28
IV. Köszönetnyilvánítás.....	29

## BEVEZETÉS

A záródolgozatom témája egy olyan filmes adatbázis létrehozása, amely mozifilmeket tartalmaz és a lehetséges a megrendelésük online felületen. Két fajta elérhetőséget biztosítottam a filmekhez: webes és asztali. Mind a kettőt más funkciókkal láttam el különböző feladataik elvégzésére. A program legfőbb célja, hogy egy otthonos és mégis modern stílusban tudja a felhasználó megtalálni és megrendelni a filmeket. Leginkább otthoni használatra van tervezve, tehát bárki elkészítheti ez alapján a saját otthoni kis verzióját és azt feltölteni a saját filmjeivel.

Azért választottam ezt a témát mivel szeretem a filmeket és szinte napi szinten használlok különböző filmes oldalakat, amiken meglehet találni a keresett filmeket vagy akár meg is rendelni. A záródolgozatom legfőbb inspirációja az IMDb és a Rotten Tomatoes nevű oldal volt. Ezen oldalaknál figyelembe vettem, hogy személyes és közeli ismerősi köreimben milyen keresési feltételek alapján is keresnek a legtöbbet és úgy módosítottam ahogy azt a racionalitás és a használhatóság keretein belül maradjon.

A programot angol nyelven írtam, mivel jobban ki igazodom és kézre állóbb, ha angolul van minden megírva és a mivel a programokat is angol nyelven használom, így értelemszerű volt, hogy amit csak lehetséges volt, azt angolul írjam meg. Úgy érzem, hogy ez nem nehezíti meg az értelmezését, mivel leginkább képekkel dolgozik ez a típusú program és színekkel is megjelöltem a tovább haladáshoz való gombokat a legkönnyebb érthetőség érdekében.

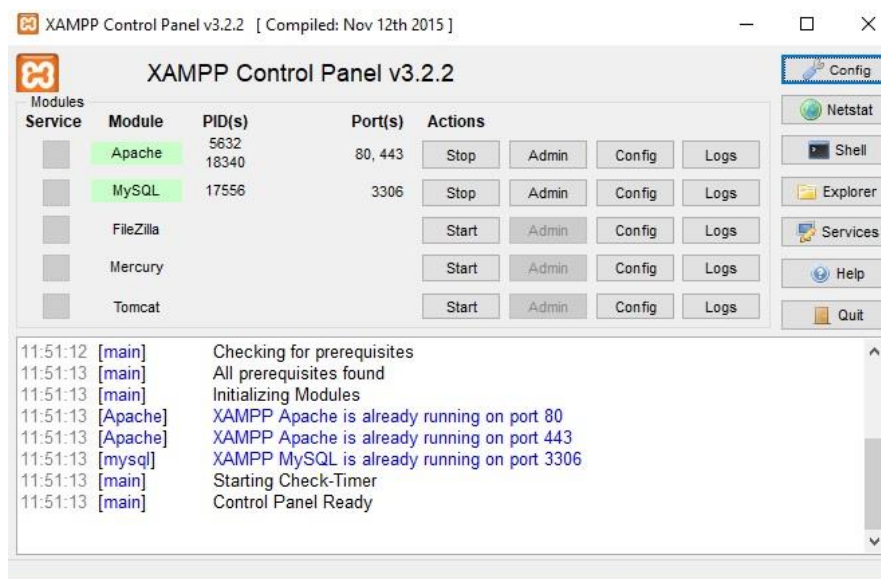
# I. FEJLESZTŐI DOKUMENTÁCIÓ

## I.1. A FEJLESZTÉS ESZKÖZEIRŐL

Az első és egyben egyik legfontosabb kérdés az volt, hogy milyen programozási nyelveken végezzem el a munkákat. Végül olyan fejlesztői programok mellett döntöttem, amiket a mai napig is népszerűek és professzionális célra is használnak, hogy a leghatékonyabban működjön a programozási feladat modern keretek között.

### I.1.1. AZ ADATBÁZIS

Az adatbázis felépítéséhez a **XAMPP** nevű programot használtam, ami egy ingyenes és nyílt forráskódú keresztplatformos web szerver csomag program. Ezt használva tudtam elérni a PhpMyAdmin modult, amit az Apache és MySQL modulok bekapcsolásával tudtam hozzáférhetővé tenni.



Kép #01 - XAMPP Control Panel

### I.1.2. AZ ASZTALI ALKALMAZÁS

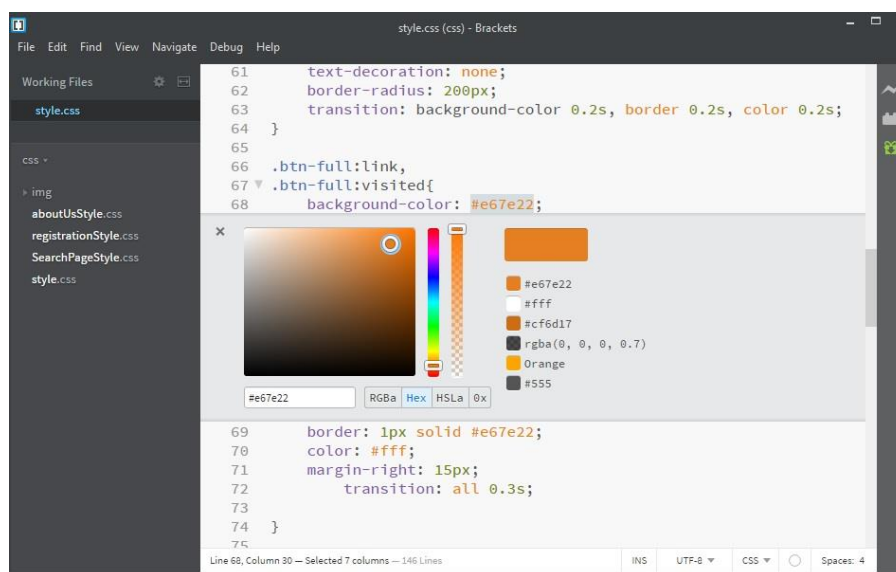
Az asztali alkalmazás fejlesztésénél a C# nyelv használata mellett döntöttem, mivel ebben a programozási nyelvben van a legtöbb és legtöbbet használt ismereteim. A fejlesztői környezetnek pedig a **Microsoft Visual Studio** mellett döntöttem.

A könnyen kezelhetőség, ingyenesség, elterjedtség és saját tapasztalataim fontos szerepet játszódtak e döntésemben. A záródolgozathoz való projekt típusa pedig Windows Form Application.

A Visual Studio 2017-es verzióját használtam a fejlesztés kezdetétől a végéig. A Visual Studio verziószáma 15.0.27130.2010 volt tehát az ez alatti verziószámokkal való futtatásnál lehetséges bizonyos verziók közötti hiányosságokból akadó hibák megjelenése.

### I.1.3 A WEBOLDAL

A záródolgozat internetes fejlesztési részénél több program is rész vett. A programozás kisebbik része notepad++-ban és NetBeans-ben zajlott ingyenességük és elérhetőségük okán, de a legtöbbet használt fejlesztési eszközöm a **Brackets** volt. A Brackets-cel a leghatékonyabban és a legmodernebb eszközökkel tudtam elvégezni a különböző formázási és stílus feladatokat. A Brackets verziószáma a fejlesztés során végig az 1.11-es build volt.



kép #02 – Brackets CSS formázás közben

A záródolgozatom webes részének a teljes elkészültségéhez az alábbi programozási nyelveket használtam:

- **HTML:** az olyan oldalakhoz, amiknél csak egyszerű szöveg és kép megjelenítés volt a feladat.
- **Php:** az egyik legtöbbet használt elem, szükséges a regisztrációhoz, felhasználók nyomon követésére, rendelés elvégzésére és az adatbázisból való adatok lekérdezésére.

- **CSS:** Arra használtam, hogy az elemek a lehető legjobb pozíciókban helyezkedjenek el, a weboldal teljes felületét átdolgozza, hogy modern, stílusos és könnyen használható legyen.
- **JavaScript:** a megrendeléshez kapcsolódó szükséges folyamatok ellátását látja el.

A webes résznél használtam még több az alábbi ingyenesen is beszerezhető és nyílt forráskódú eszközöket:

- **Ion Icons:** A weben megjelenő különböző ikonokhoz használtam például a kosár ikon vagy az About Us oldalon lévő kisebb ikonok.
- **normalize.css:** gitHubról ingyenesen letölthető formázási útmutatókat mutató tábla, ami segít abban, hogy a megfelelő megjelenést biztosítsa a program.
- **grid.css:** A responsivegridsystem.com nevű oldalról ingyenesen letölthető nyílt forráskódú css fájl, ami elősegíti azt, hogy például az About us oldalon a 4 oszlop egymás mellett legyen és reszponzívak legyenek a képernyő méretváltoztatására.

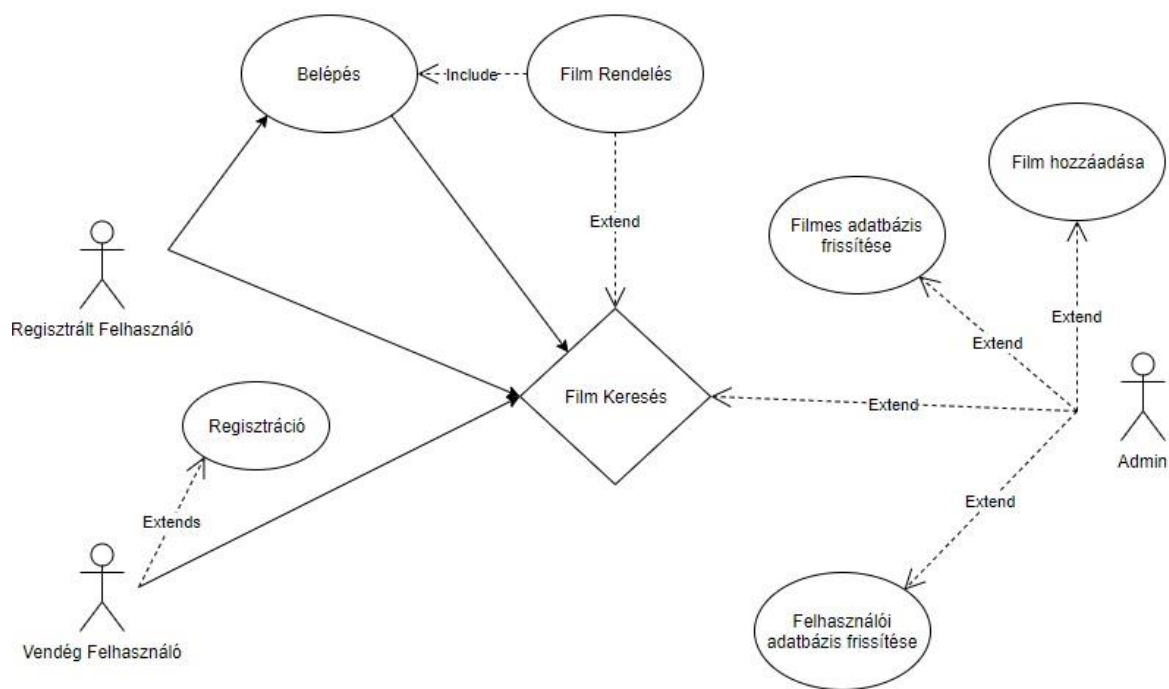
#### **I.1.4 EGYÉB FELHASZNÁLT ESZKÖZÖK**

- **Draw.io** – UML diagram elkészítéséhez használt ingyenes weboldal.
- **Google Chrome Developer Tools**

#### **I.2 – A PROGRAM FELÉPÍTÉSÉRŐL**

A fejlesztés során figyelembe kellett vennem több tényezőt is, hol bővíteni, hol eltávolítani kellett funkciókat/tulajdonságokat a programból ahhoz, hogy elérje a végső formáját. A legtöbb kérdés a felhasználóhoz kapcsolódott, hogy pontosan milyen adatok is legyenek szükségesek ahhoz, hogy sikeresen lehessen azonosítani, regisztrálni, bejelentkeztetni és megrendeltetni vele egy filmet. Ezt a problémát végül sikerült megoldani előzetes egyeztetések és alapos utánajárások után.

A lentebb található **Use-Case** diagram tökéletesen bemutatja, hogy milyen szinten milyen feladatok elérhetőek. A vendég felhasználó ha akar regisztrálhat, de kereshet anélkül is filmet, bár rendelni már nem tud mivel ahhoz sikeres belépés kell meglévő fiókkal.



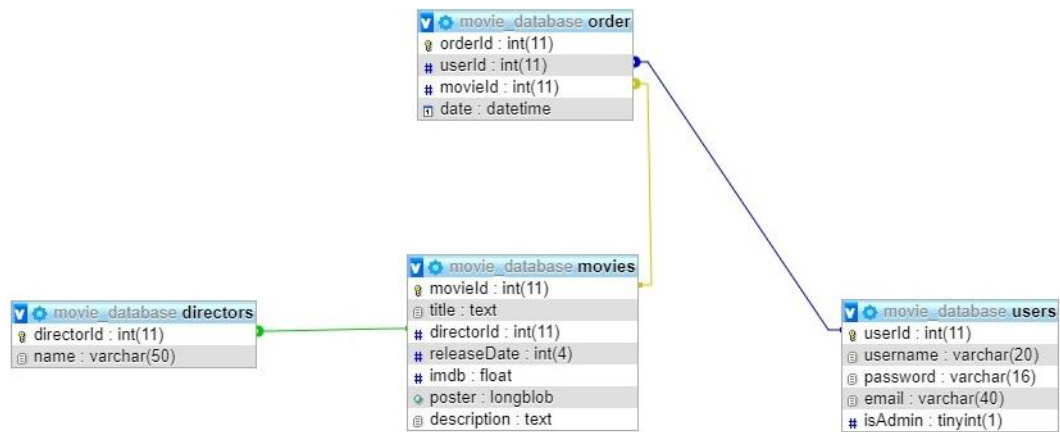
kép 03# - Use-Case Diagram

## I.2.1 – AZ ADATBÁZIS SZERKEZETE

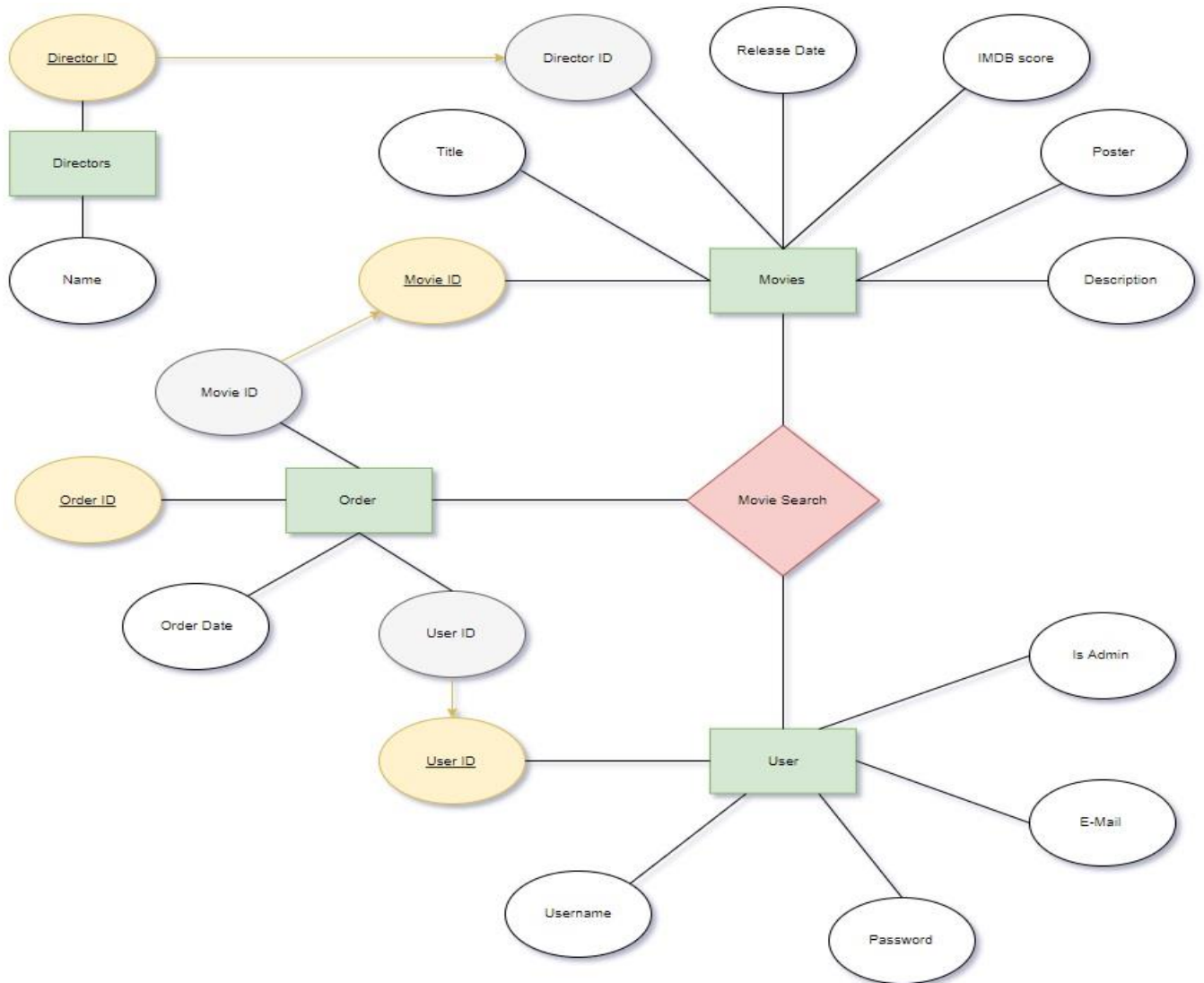
Az adatbázishoz végül négy táblát készítettem az alábbi csoportosítások szerint:

- **Movies:** Ebben a táblában helyezkedik el az összes az összes filmekhez kapcsolódó anyag.
- **Directors:** Az összes rendező neve szerepel ebben.
- **Users:** ebben a felhasználók kerülnek rögzítésre.
- **movieorder:** A sikeres film megrendeléseket tartalmazza.





kép #04 - Az adatbázis felépítése 01



kép #05 - Az adatbázis felépítése 02

### I.2.1.1 – A Movies tábla

A filmeknek az alábbi tulajdonságokat adtam:

1. **MovieId:** Egy integer, automatikusan növekvő szám, ami minden filmhez egyedi, ez a film sorszáma. Ez a mező a kulcs ebben a táblában.
2. **Title:** A film címe, Text típus mellett döntöttem, mivel szerepelhetnek olyan film címek, amik elég hosszúak és több mondatból is állhatnak, ami belekavarhatna a program helyes működésébe ha Varchar típussal dolgozok.
3. **DirectorId:** Egy integer szám, amiben egy rendezőhöz rendelt azonosítót rejti, minden rendezőnek egy darab azonosítója szerepel itt, amit a DirectorId táblán belül kötünk össze, ahol a neve is szerepel a rendezőnek.
4. **ReleaseDate:** Integer szám. A film kijövetelének évét határozza meg.
5. **IMDB:** Float típusú szám. Az imdb.com oldalon lévő filmekhez tartó értékelési pont. Az értékelés értéke 1.0-tól 10.0-ig terjedhet csak.
6. **Poster:** longblob típus, amiben képeket tárolok. A Blob használatával egy adatbázisba fog kerülni minden adat, ami egy filmhez kell. LongBlob verziót használtam, ezzel megkönnyítve a film plakátjainak szánt képek kiválogatását mivel a LongBlob a legnagyobb tárképességét befogadó formátuma a Blob-nak. A Blob használatával igaz hogy megkönnyítettem az elhelyezést, de egy másik akadállyal kellett szembesülnöm, mégpedig hogy az így kiexportált sql adatbázis állományomat nem tudja alapból kezelni egyik XAMPP program sem, így szükségessé téve azt, hogy át kell írni az adatbázis importálás előtt a php.ini-t. Módosításnak az alábbi két sort kell átírni a z egyenlőségjel utáni értékekre:  
a, post\_max\_size=30M  
b upload\_max\_filesize=20M
7. **Descrpition:** Text típus. Minden filmnek külön leírása van, ahol bármilyen szöveg szerepelhet.

### I.2.1.2 – A Directors tábla

Itt csak 2 mező szerepel mivel a célja a normalizálás volt, ezzel elkerülve, hogy feleslegesen többször tárolja ugyanazokat az adatokat az adatbázis. A mezők:

1. **DirectorId:** Egy integer, automatikusan növekvő szám, ami minden rendezőhöz egyedi, ez a rendezőknek a sorszáma.  
Ez a mező a kulcs ebben a táblában, és ugyanezen a néven található a movies táblában is, ezáltal megkönnyítve a kettő tábla összekapcsolását.
2. **Name:** Varchar típus. A rendezőnek a neve szerepel benne. Az adatbázisban szereplő filmekhez konkrétan úgy válogattam össze, hogy legyen olyan rendező, aki sokszor szerepel, és olyan is, aki csak egyszer szerepel a jobb reprezentáltság érdekében.

### I.2.1.3 – A Users tábla

A felhasználókat az alábbi mezőkben szerepeltettem:

1. **UserId:** Egy integer, automatikusan növekvő szám, ami minden felhasználóhoz egyedi. A felhasználóknak a sorszámát jelöli. Ez a mező a kulcs ebben a táblában.
2. **Username:** Varchar típus maximum 20 karakternyi hosszúságú lehet. A felhasználónevet tartalmazza.
3. **Password:** Varchar típus, maximum 16 karakternyi hosszú lehet. A user ID-hoz tartozó jelszót tartalmazza.
4. **E-mail:** Varchar típus, maximum 40 karakternyi hosszú lehet. A user ID-hoz tartozó E-mail címet határozza meg.
5. **IsAdmin:** Boolean változó. Azt jelöli, hogy a felhasználó rendelkezik-e adminisztrátori jogokkal. Regisztráció során mindenkinek alapból a 0-ás érték kiosztva, tehát nem rendelkeznek adminisztrátori joggal. Az admin rendelkezik egyedül 1-es értékkel ami azt jelenti, hogy alap esetben ő az egyedüli adminisztrátor. SQL parancscsal megváltoztatható.

### I.2.1.4 – A movieorder tábla

Leginkább egy gyűjtő táblának használt, ami arra lett létrehozva, hogy sikeresen tudjon az esetleges üzlet tulajdonosa vagy adminisztrátora ellenőrizni a sikeres bejövő megrendeléseket. Fő feladata, hogy össze kösse a megrendelő userId-t a megrendelni vágyó movieId-val amit egy orderId-ben mentek el. Plusz dologként hozzáraktam a megrendelés időpontját. A movieorder tábla mezői:

6. **OrderId:** Egy integer, automatikusan növekvő szám, ami minden rendeléshez egyedi, ez a rendelés sorszáma. Ez a mező a kulcs ebben a táblában.
1. **UserId:** Integer szám. A users táblával van összekötve, így azonosítva hogy pontosan melyik userID-val rendelkező felhasználó rendelte is meg a filmet.

2. **MovieId:** Integer szám. A megrendelni kívánt filmet azonosítja a movies táblában lévő movieId-val van összeköttetésben.
3. **Date:** Date típus. A megrendelés pontos időpontját azonosítja. Benne foglalja a megvásárlás évét, hónapját és napját.

## I.2.2 – AZ ASZTALI PROGRAM

Az asztali programot három fő különböző részre osztottam. A program fő részeihez való sikeres eléréséhez először is egy sikeres adminisztrátori belépés kell, mivel az adminisztrátor tud szerkeszteni adatokat. Ha sikeres volt a belépés, akkor az adminisztrátor 3 lehetséges tabPage-re tud rámenni, azok között váltogatni és megnézni vagy akár módosítani bizonyos adatokat. Kilépés teljes mértékben a programból a piros Exit gombbal lehetséges ami a menuStrip-en helyezkedik el, emellett kaphat a felhasználó a program verziójáról az About Us gombra kattintva.

### I.2.2.1 – Asztali alkalmazás: Login

Úgy gondoltam, hogy a felhasználó csak akkor láthassa a teljes projektet, hogy ha már be van jelentkezve, addig rejtve marad a fő form. Kettő adatot kérek be tőle, a felhasználónevét (username) és a jelszavát (password). Úgy lett a program megtervezve, hogy csak az alap felhasználónév és jelszó kombinációval tud az adminisztrátor belépni. A sikeres belépéshez szükséges adatok:

- Username: admin
- Password: admin

### 1.2.2.2 - Asztali alkalmazás: Movies – Movie Tábla

Az asztali alkalmazás szíve a Movies tábla ahol az adatbázisból lekéri a program a filmek majdnem összes adatát és különféle módon megjeleníti azokat.

Az oldal legfontosabb eleme, a középen lévő dataGridView. Amint a felhasználó rákattint a **Load** gombra, a program betölti az adatbázisból az összes adatot a DataGridView-ba és meg is jeleníti egy részét ott (movie ID, Title, Director name, Release Date, IMDB score). A Description is a DataGridView-ba töltődik be, de ez az oszlop el van rejtve a felhasználó elől és inkább úgy döntöttem, hogy egy külön readOnly textboxba helyezem el, így javítva a láthatóságát és praktikusságát. Mellékeltem a Load gomb adatbázishoz való lekérés kód egy kisebb részletét.

```

1. private void buttonLoad_Click(object sender, EventArgs e)
2.     {
3.         Database d = new Database();
4.         mdi = d.connect();
5.         mdi.open();
6.         moviesDT = mdi.getToDataTable("SELECT movies.movieId, movies.title, directors.name,
movies.releaseDate, " +
7.             " movies.imdb, movies.poster, movies.description FROM directors, movies WHERE
movies.directorId LIKE " +
8.             " directors.directorsId" );
9.
10.        dataGridViewMovies.DataSource = moviesDT;
11.        this.dataGridViewMovies.Columns["poster"].
Visible = false;
12.        this.dataGridViewMovies.Columns["description"].Visible = false;

```

A DataGridView-on belül jelenleg csak egy darab sort lehet kiválasztani és az ahhoz tartozó adatokat megnézni. A Description TextBox (Leírás) automatikusan megjeleníti a kiválasztott filmhez tartozó leírást és a DataGridView és a Description Box közötti részen pedig megjelenik a kiválasztott film neve amit meg egy egyszerű Label jelenít meg.

A **Delete** gomb törli az adott filmet a listából, de ahhoz hogy mutassa az adatokat, újra választani kell filmet egy kattintással. A **New** gombra kattintva felhossa az edit formot amin keresztül pedig egy új film adható hozzá a listához. Az Edit gomb nem használható ezen formában, mellékeltem felette a leírást, hogy dupla kattintással működik az Edit tábla felhozása.

### 1.2.2.3 - Asztali alkalmazás: Movies – Edit Tábla

Az Edit tábla akkor jelenik meg, ha a felhasználó duplán rákattint a DataGridView-on belül egy filmre. Ekkor megjelenik a film adatai, amelyeket tud majd szerkeszteni. A movie Id-ját nem engedtem, hogy átírja, ezzel a módszerrel kiküszöbölve hogy több azonos MovieId-val rendelkező film legyen. A movieId-t emiatt a biztonsági okból csak egy egyszerű Label jeleníti meg. A Save gombra kattintva visszatér az eredeti movies táblára, a már mentett adatokkal. Itt lehetséges az új film hozzáadása is, ekkor érték nélküli textboxok láthatóak és kitölthetőek, amiket a Save gomb menti is és elküldi az adatbázisnak

### 1.2.2.4 - Asztali alkalmazás: Users tábla

Itt listView-ot jelenít meg, benne a már regisztrált felhasználók majdnem összes adatával (user ID, username, password, email), kivétel az isAdmin ami itt rejtve marad. A passwordok

hashelve jelennek meg biztonsági okokból kifolyólag, kivéve az adminisztrátoroké a könnyebb belépés érdekében.

### 1.2.2.5 - Asztali alkalmazás: Order Tábla

A megjelenítési mód ListView, és a felhasználó itt is csak megnézni tud adatokat, szerkeszteni nem. Kijelzi a rendelés azonosítóját, hogy melyik ID-val rendelkező user rendelte és melyik movie ID-val rendelkező filmet a rendelés időpontjával egyetemben. Azért döntöttem a movieId mellett, és nem a film címe mellett, mivel több azonos című film is lehet az adatbázisban (reboot-ok és remake-ek), és a movieId pedig leszűkíti hogy pontosan melyik id-val rendelkező filmről van szó. Egy rendelésben több film is lehet, ezért több film együttes megrendelésénél többször is megjelenik a listView-ban, csak a movieId fog változni, más adat nem. A movieorder adatbázis tábla összes adatát megjeleníti, ezzel is kikerülve a phpMyAdmin vagy SQL lekérdezés felesleges használatát.

### 1.2.2.6 – Asztali Alkalmazás: Unit Teszt

A program helyes működése érdekében egy unit (egység) tesztet is hozzáadtam, ami a Login formon megjelenő helyes input adatok bevitelét vizsgálja. A Unit test ezen része azt vizsgálja meg, hogy a felhasználó a belépéshez szükséges adatoknál gépelt e valamilyen adatot az input textbox mezőkbe. Ha üresen hagyta a felhasználó, akkor egy piros felkiáltójel fog megjelenni a textbox mellett, ami jelzi, hogy nem lehet üresen hagyni a mezőt, valamilyen adatot feltétlen meg kell hogy adjon.

```
1. [TestClass()]
2. public class CheckTests
3. {
4.     [TestMethod()]
5.     public void checkNameIsNotEmptyTest()
6.     {
7.         Check cs = new Check();
8.         Assert.IsFalse(cs.checkNameIsNotEmpty(""));
9.     }
10.
11.     [TestMethod()]
12.     public void checkPasswordInNotEmptyTest()
13.     {
14.         Check cs = new Check();
15.         Assert.IsTrue(cs.checkPasswordInNotEmpty("any"));
16.     }
17. }
```

### I.2.3 – A WEBES ALKALMAZÁS

A Movie Database Webes részére a regisztráció és a megrendelés fő vonulatán megy, ezért leginkább PHP kóddal folyt az oldal fejlesztése. A regisztrációt és a Logint (Bejelentkezést) Php használatával oldottam meg, míg a kosárba helyezést Php és JavaScript segítségével.

Az oldalakat a fő mappába helyeztem el, de létrehoztam egy resources és egy vendors mappát is. A **resources** mappában az én általam elkészített/letöltött programok és képek vannak. Ezen mappán belül három almappa található: css, img és js. A css mappában találhatóak a css fájlok és egy img mappa amiben csak a háttérképeket tárolok. A js mappában pedig a rendeléshez szükséges javascript fájl található. Ezzel az elrendezéssel minden könnyen és gyorsan megtalálható és könnyen átrendezhető is szinte kizárva a kavarodást a fájlnevekkel vagy formátumokkal.

A **Vendors** mappában található a külsődleges fájlok, az itt található css mappában az internetről letöltött nyílt forráskódú és ingyenes segéd grid-ek és normalizáló css fájlok találhatóak, míg a fonts-ban az ikonok találhatóak.

A weboldalakat Mozilla Firefox Quantum böngészőnek 59.0.1-es verzióját használtam fő tesztelési felületnek, de emellett még a Google Chrome 65.0.3325.181 verzióját használtam hibaellenőrzésre. A Google Chrome Troubleshoot funkciója kifejezetten hasznos alkalmazás volt a hibák nagy részének a megtalálásában.

A weboldalnál úgy döntöttem, hogy a fejléct (Menu Bar-t) külön kezelem, és lehívom minden oldalra a **session** használata miatt. A sessionnel nyomon tudom követni, hogy a felhasználó jelenleg milyen fázisban tart:

- nincs bejelentkezve (logged out)
- be van jelentkezve (logged in)

Ha még nincs bejelentkezve, akkor a menüsor jobb oldalán a Registration és a Login oldalak jelennek meg. Az index oldalon meghagytam a középen lévő nagyobb registration gombot a jobb hatás érdekében, nem szükséges ott figyelni, hogy login létrejött-e sikeresen vagy sem. A másik kiemeltebb gomb a search movie, mivel ez egy olyan lényeges elem, amit vendég és regisztrált felhasználó egyaránt használhat.

Ha viszont már be van jelentkezve a felhasználó, akkor a logout (kijelentkezés) és a basket (kosár) php oldalak jelennek meg. Ezzel elkerülöm a menu bar túlzsúfoltságát, hogy mindegy hogy be van-e jelentkezve vagy sem, ott kellene hogy legyen külön a login, logout, basket, registration és nem csak design szempontjából megkapóbb, de praktikusabb is kevesebb elemmel dolgozni a menu bar-ban. A session követés végigfut az egész weboldalon, még ott is ahol nincs igazából lényegi funkciója (index.php és aboutUs.php) a menu bar folyamatossága miatt, így igaz hogy azok php oldalaknak minősülnek, de az egyetlen phpval rendelkező funkciójuk a menu bar.

#### **I.2.3.1. – Webes Alkalmazás: Az Index oldal**

Az index.php (fő oldal) célja hogy bevezesse és megmutassa a felhasználóknak, hogy milyen lehetőségei vannak. Középen ki van emelve regisztráció és a keresés gombok, CSS-ben külön kiemelve hogy látványos legyen ez a két funkció hogy megragadja a felhasználó figyelmét. Az index oldal csak html és css elemeket tartalmaz. Külön figyelmet fordítottam a középen megjelenő hero-text-box részlegre mivel ez a főoldal „hőse”, ezt látja minden felhasználó először és a legfontosabb css formázásokat erre kellett irányítani először, és innét „továbbvinni” a már megszerzett css stílusformázási adatokat a többi css dokumentumra.

#### **I.2.3.3. – Webes Alkalmazás: Az About Us oldal**

Az About Us (Rólunk) oldal fő célja ismertetni a felhasználót az oldallal. Ez az oldal csak html és css elemeket használ. Az oszlopokat egy külsődleges (grid.css) fájl kezeli a legjobb responzivitás érdekében. A külső grid css lehetővé teszi azt, hogy az oszlopok számát könnyen lehessen növelni vagy éppen csökkenteni tetszés szerint. A responzivitás érdekében viszont át kellett állítanom hogy mikortól váltsanak át egymás melletti vagy éppen az egymás alatti elrendezésbe. Ezt az értéket 1080px-nél határoztam meg, mivel ez adja a lehető legkényelmesebb olvasási módot.

#### **I.2.3.4 – Webes Alkalmazás: Registration, Login & Logout**

Ezen oldalak fő mozgatórugója a php. A felhasználótól a regisztrációkor három adatot kérünk be (username, password és email) amiket különböző módon ellenőrzünk függvények, preg\_match, hosszúság és egyéb validitások kapcsán. Password Confirm (jelszómegegyeztetés) is tartalmaz az oldal, ami egy egyszerű egyenlőségvizsgálattal ellenőrzi, hogy van-e egyezés a fentebbi, elsődlegesen beírt jelszóval.



```

1.  function test($str)
2.  {
3.      $str = trim($str);
4.      $str = strip_tags($str);
5.      $str = stripslashes($str);
6.      return $str;
7.  }
8.  $usernameErr = $passwordErr = $emailErr = "";
9.
10. if($_SERVER['REQUEST_METHOD'] == "POST"){
11.    //USERNAME
12.    if(isset($_POST['username']) && !empty($_POST['username']) && $_POST['username'] == test($_POST['username']) && preg_match("/^[a-zA-Z]*$/", $_POST['username']) && strlen($_POST['username']) <= 40)
13.    {
14.        $username = $_POST['username'];
15.    }
16.    else
17.    {
18.        $usernameErr = "Wrong username!";
19.    }

```

A Login (Bejelentkező oldal) csak a felhasználónevet és a jelszót kéri, majd sikeres belépés után a session megjegyzi a belépőnek a userId-ját. A Logout oldal meg megszünteti a jelenleg aktív állapotban lévő session-t a session\_destroy() használatával, majd visszairányít az index.php oldalra.

### I.2.3.5 – Webes Alkalmazás: Movie.php

A movie.php feladata elsődlegesen a posterek megjelenítése, filterek beállítása és a kosár ikonra kattintva a basketbe (kosárba) helyezése a kiválasztott filmeknek. Minden fontosabb cselekvés a functions.php-n belül történik a movieList function-t végrehajtva.

```

1. function movieList($link){
2.     $sql = "SELECT * FROM movies";
3.     $add = "";

```

A filtereket egy SQL paranccsal működnek. Az alapvető SQL lekérdező string változatlan marad, és az \$add végzi el a filter feladatát, amit a végén pedig hozzáadunk az alap \$sql-hez. A reset filterben üres marad az \$add változó tehát filter nélküli alaphelyzetbe hozza az oldalt, pont mint az eredeti betöltésnél. Az IMDb pontszám szerinti filtert egy Switch oldja meg míg a cím szerinti filter szinte teljes egészében sql parancs végzi el. A switch mellett döntöttem, mivel a felhasználók többsége mindig egy bizonyos pont feletti értékekre kíváncsi, így felesleges túlbonyolítása lett volna a programnak. Select paramtérnél 5.0 alatti értéket nem adtam meg, mivel alapesetben nincs is 5.0 alatti film az adatbázisban, és nem szokás

felhasználói szinten ez alá menni, mivel az átlag felhasználóknak csak igen kicsi százaléka érdekelt lenne ilyen kritikus pont alá menni.

A lentebbi kód részleten látható, hogy hogyan is működik a másik filter, a film címe alapján való keresés. Itt a felhasználó egy input mezőbe írhatja bele, hogy mire keressen rá az oldal, és akár szón belüli találatokat is kilistázni fogja. Tehát ha a felhasználó azt írja be, hogy „ars” akkor az (alap esetben létrehozott adatbázisban) az összes Star Wars filmek kilistázza.

```
1.  print'
2.      <div class="movie-filter-section"><p>Filter by Movie Title</p></div>
3.      <form action="'. $_SERVER['PHP_SELF']. '" method="get">
4.      <input type="text" name="movie-title" placeholder="Movie Title" required>
5.
6.      <input class="button-submit" type="submit" value="Search by title"
   name="title">
7.      </form>;
8.
9.      print'</div>;
10.
11.
12.      if(isset($_GET['movie-title'])) {
13.          $movieTitle = $_GET['movie-title'];
14.
15.          $add = " WHERE title LIKE '%" . $movieTitle . "%' ";
16.      }
```

A movie.php egyik legnagyobb kihívása a képek megjelenítése volt, mivel a blobban tárolt képeket sokkal nehezebben lehetett megjeleníteni, mintha egy külön fájlból lettek volna csak beolvasva. Az alábbi kód részlettel végül is sikerült működésre hozni a képek lehívását és egyből átkonvertálását.

```
1.  while($row=mysqli_fetch_assoc($image)){
2.      echo '<div class="movieContainer">'.
3.      '<div class="movie-title">'.
4.      $row['title'].
5.      '</div><i class="ion-android-cart movie-cart">';
8.      echo ' id=';
9.      echo $row['movieId'];
10.     echo '></i></div>';
11. }
```

Mint ahogy a fentebbi kódrészletben is látható, több echora lett bontva a lekérdezés, mivel a ‘ és a “ kezelése php-ben könnyen nehézségekbe ütközhető esemény a php sajátos tag lezárása miatt. A képeket egy containerben helyeztem el, CSS használatával pedig reszponzívvá tettem a képek megjelenítését. A kép alatti kosár ikon rendelkezik alap esetben azzal a movieId-val, ami alatt elhelyezkedik, így rendelés esetén azonnal be van azonosítva a film.

### I.2.3.6 – Webes Alkalmazás: **basket.php, Order.php, Functions.php** etc

Az egyik legbonyolultabb része a webes alkalmazásnak a rendelési funkció. Itt található a legtöbb programozási nyelv használva: html, css, php és javascript. A html és css feladata ugyanazok voltak eddig, mint a többi oldalon, tehát csak az egyszerűbb adatok megjelenítése és formázása. A php szükséges lett az adatbázisból való adatok lehívására (pl. poster, film cím) és a kosár tartalmának megtekintésére. A javascript fájl a kosár tartalmának a változásait vizsgálja.

Ha a már bejelentkezett felhasználó kosárba helyezett filmet a movie.php-ról, akkor az a basket.php-n fog feltűnni. Egy table-be jelennek meg az adatok, úgy mint a megrendelni kívánt film címe, majd ugyanabban a sorban egy Delete (törlés) gomb, ami csak azt az egy filmet törli, amelyik sorban jelenleg elhelyezkedik. Úgy alakítottam, hogy hiába kattint a kosár ikonra többször a felhasználó, akkor is csak egy darabot tud belehelyezni a kosárba, ezt nem tudja se megkerülni, se átjátszani az egyszeri felhasználó. A végső árat úgy számolja ki az oldal, hogy megnézi hány sornyi film lett belerakva, és azt megszorozza 5-tel, mivel minden film ára egységesen \$5. A Delete esetén automatikusan újratölti az oldalt egy `window.location.reload()` paranccsal. Kosárürítésnél (Empty Cart) mindent kivesz a kosárból, de a session megmarad.

Javascript összesen 4 funkciót lát el:

- **movie-cart:** a kiválasztott filmnek a kosárba helyezése id alapján.
- **Delete:** teljesen törli a kosárból a kiválasztott id-val rendelkező filmet.
- **Empty-basket:** Kosár ürítése, mindent töröl a kosárból.
- **Order-button:** a film megrendeléséhez szükséges funkciót látja el.

Session követés minden oldalon van, a belépett felhasználó csak saját magának és a saját maga adataira tud megrendelni filmet. Keys (kulcsok) létrehozásával és while ciklus használatával végigmegy az kosár tartalmán (ahol csak a movieId van, a title máshol van lekérdezve a function-on belül) és végül elkészül az order.php segítségével a teljes lekérdezés, amit fel akarunk tölteni. Utána SQL paranccsal fel is töltjük a movieorder táblába a megfelelő adatokkal. Az SQL parancs végéből a while ciklus miatti folyamatos ismétlődés miatt az utolsó sornál hiba bukkanhatna fel az utolsó movieid utáni vessző miatt, ezért használtam a `substr($sql, 0, -2);` parancsot. A substr a jövőbeli SQL lehívás végéből kivonja az utolsó kettő karakter, ami miatt helyes lesz az SQL parancs és el is lehet küldeni az adatbázisba.

### **I.2.3.7 – Webes Alkalmazás: Connect.php**

Egy feladata van: az adatbázishoz való sikeres kapcsolódás biztosítása

## **I.3. EGYÉB INFORMÁCIÓK**

### **I.3.1 Továbbfejlesztési javaslatok**

A program esetleges bővítésénél szóba jöhet először is az adatbázis fejlesztése/kibővítése. Ha úgy kívánná egy esetleges felkérés, akkor át lehetne alakítani a programot úgy, hogy az adatbázisban a képeket nem LargeBlob-ban tárolja, hanem stringként a linket a képhez, és azt a programmal együtt tárolni, nem az adatbázisban. Ez attól is függ, hogy ki, hogy látja össze a programját, én ezt a lehetőség egy esetleges nagyobb bővítés során venném csak figyelembe.

Másik fejlesztési lehetőség hogy új film felvételénél képet is lehessen választani, asztról betölteni vagy internetes linken keresztül és úgy jelenítődne meg a film plakátja, de én ezt a lehetőség sem javasolnám sem biztonsági sem pedig praktikussági szempontból (rossz link küldés, már nem létező link, vírus stb). Lehetséges opció még esetlegesen az asztali funkciók áthelyezését webes felületre és fordítva.

### **I.3.2 Fejlesztés során történt változások**

A program kezdeti és végleges formája között lényeges különbségek vannak, amiket a fejlesztés során kellett módosítanom, hogy a leghatékonyabb és legmodernebb stílusú alkalmazást készítek.

Az egyik ilyen lényeges különbség, hogy az adatbázisból eltüntettem a profil táblát. A profil tábla össze lett volna kötve a users táblával és annak lett volna a „folytatása” kibővített adatokkal. Megjelentek volna a felhasználó személyes adatai (Teljes név, születési hely, születési időpont, lakcím) ám ezek elvetése mellett döntöttem, mivel ezek teljesen felesleges adatok ahhoz, hogy valaki egy streaming linket megvegyen. Az E-mail cím is eredetileg a profil táblában volt, az volt az egyedüli amit átmentettem a profil táblából a users táblába.

Másik változás a bővítése volt a movies táblának. Eredetileg csak szöveges megjelenítésű rövid tartalmak kaptak volna helyet benne, de a felhasználóbarátsági szempontból a képek bevezetése szóba és az kapta a prioritást is benne, habár nehezebb képekkel dolgozni mint a szöveges tartalmakkal de egy ilyen típusú (filmes) projekthez elengedhetetlen a képek használata.

## II. FELHASZNÁLÓI DOKUMENTÁCIÓ

### II.1 A PROGRAM ALAPVETŐ FUNKCIÓI

A **Movie Database** nevű program legfontosabb funkciója a filmek keresése, ami több szinten is elérhető. Az adatbázisba alaptól fel van töltve információkkal, webes felületen plakát, asztali alkalmazásnak pedig a különböző keresési funkciók miatt gyorsan és könnyen megtalálható bármelyik film.

Az **asztali alkalmazásban** fő része a Movies oldal ahol a filmek szerepelnek. Itt gyorsan megtalálhatja a felhasználó bármelyik filmet, hisz elég csak rákattintani a keresni kívánt opcióra például IMDb pontszám és automatikusan növekvő vagy éppen csökkenő sorrendbe helyezi őket, pont úgy, ahogy azt felhasználó szeretné. Mivel a leírás külön van elhelyezve a betöltő felületen kívül, ezért arra nem lehetséges a keresés semmilyen formában. Módosítani filmeket lehetséges az asztali alkalmazásban, mivel alapvetően le lett korlátozva, hogy csak adminisztrátori jogkörrel lehet belépni, így aki rendelkezik azzal a joggal, az azonnali teljes hozzáférést kap a programhoz és az adatbázis filmes részének módosításához is. Itt tudja módosítani egy filmnek csakis az előre meghatározott és megengedett adatait. Bármilyen más változtatáshoz SQL paranccsal vagy phpMyAdmin-al történik.

A **webes felület** leginkább a regisztrációra és a filmek megvásárlására összpontosít. Itt tud az átlag felhasználó belépni ha már regisztrált de csak akkor nincs lefoglalva az a username amit megadott. A vásárlást pedig csak bejelentkezett felhasználó tudja elvégezni. A filmek vásárlásánál nincs mennyiség megadva, mivel nem lehetséges házhoz rendelni mivel egy felhasználó csak egy filmet csak egyszer tud megvenni, mivel (a korral haladva) már nincs fizikai másolat amit megrendelhet, hanem egy streaming linket küldünk arra az e-mail címre amit a regisztráció során megadott. A filmek jelenleg egység áron lehet megvenni, nincsenek az árak között különbségek, minden egységesen \$5, és minden filmből csak egy darabot vehet. Ha a felhasználó már helyezte a kosarába terméket, akkor a kosár tartalmát bármikor meg tudja nézni ha a basket földre kattint, persze az csak akkor érhető el, ha már be van jelentkezve a felhasználó. Az adminisztrátor és átlag felhasználó között webes felületen nincs eltérés, ugyanazok a funkciók érhetőek el mind a két fél számára.

## II.1.2 A PROGRAM FELHASZNÁLÓI SZINTJEI

Felhasználói típusoknál hármat különböztethetünk meg: vendég felhasználó, regisztrált felhasználó és adminisztrátor.

A **vendég felhasználó** csak keresni tud a filmek között és regisztrálni a webes felületen, semmilyen szerkesztési jogkörrel nem rendelkezik. Asztali alkalmazásba nem tud belépni.

A **regisztrált felhasználó** már be belépés után ugyanúgy elérhető a webes alkalmazásban a keresés funkció de már tud rendelni is filmet. Mivel csak a regisztrált felhasználó és az adminisztrátor rendelkezik adatbázisban lévő E-mail címmel, ami szükséges a rendeléshez, ezért csak ettől a felhasználói szinttől lehetséges a rendelés. Asztali alkalmazásba nem tud belépni.

Az **adminisztrátor** a legtágabb jogkörrel rendelkező személy, eredetileg kettő személyt adtam meg ebbe a jogkörbe, de lehetséges mások hozzáadása, amit viszont az adatbázison belül lehet csak elintézni egy mező átírásával. Teljes körű hozzáférése van az asztali alkalmazáshoz ahol tud hozzáadni, módosítani és törölni filmeket.

## II.2 – HASZNÁLATI ÚTMUTATÓ

A következő pontokban az lesz bemutatva, hogy a felhasználó hogy és milyen módon tudja használni az asztali és a webes alkalmazást ahhoz, hogy a lehető legtöbbet hozza ki belőle. A Movie Database asztali és webes helyes működése érdekében szükségeltetik a XAMPP program futtatása az Apache és a MySQL modul bekapcsolásával. Szükséges még egy, a záródolgozathoz mellékelt adatbázis (movie\_database.sql) importálása a helyi hálózatra, amit viszont csak egy átírással lehetséges importálni a XAMPP-on belül a config fülön elérhető php.ini-ben kell átírni a következő kettő sort az egyenlőség utáni értékekre:

a, post\_max\_size=30M

b upload\_max\_filesize=20M

### II.2.1. – Felhasználói Dokumentáció: Az Asztali Alkalmazás

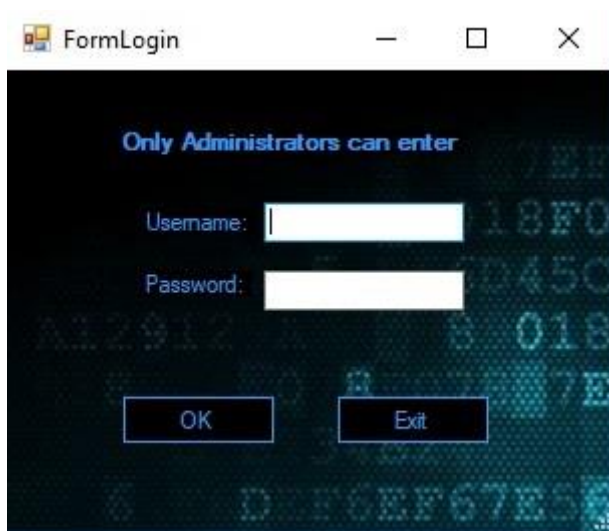
Az asztali program egy külön telepítő által számítógépre felrakható program amihez szükséges hogy a setup.exe futtatása és a telepítés helyes elvégzése és a XAMPP folyamatosan fusson akár a háttérben is.

### II.2.1.1. – Asztali alkalmazás: Login

A program elindítása után az elsődleges dolog ami azonnal felbukkan, és nem lehet kikerülni az a Login (Bejelentkező) képernyő. Itt csak az admin jogkörrel rendelkező léphet be, de ezt jelzi is a program a tetejénél. Alapvető belépéshez szükséges adatok:

Username (Felhasználónév): admin

Password (Jelszó): admin



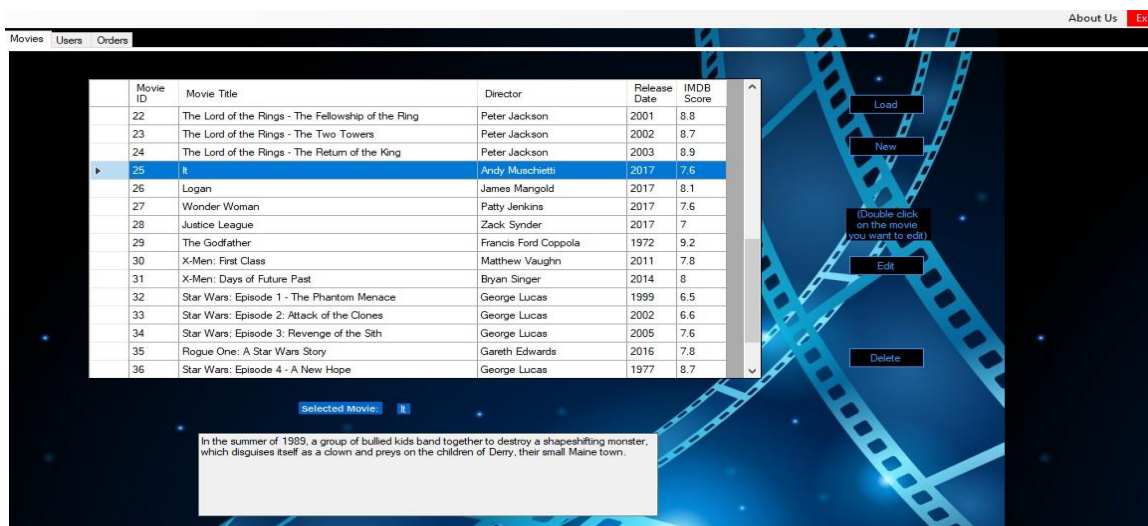
kép #06 - Asztali: Login

Ha beírta a helyes adatokat, akkor az OK gombra kell kattintani és sikeres belépés után már a teljes program elérhető is lesz. Ha a felhasználó mégis ki szeretne lépni, akkor az Exit gombra kell kattintania és a program ki is lépteti a programból teljesen.

### II.2.1.2 – Asztali Alkalmazás: Movies

A legtöbbet használt része a programnak, itt találhatóak a filmek és azok adatai, illetve a módosítások is csak ezen a fülön lehetségesek.

A Program alapvetőlegesen nem tölti be azonnal az adatbázis-t, először is a felhasználónak rá kell kattintania a **Load** (Töltés) gombra, ekkor betöltődik az adatbázisból az összes adat. Az adatbázisból való adatok lekérése után már lehetséges a módosítások. Itt minden film megtalálható ami szerepel az adatbázisban, tehát ha valaki keres egy filmet de nem találja, akkor azt a filmet a New (Új) gombra kattintva már hozzá is tudja majd adni azonnal a rendszerhez. Egyszerre viszont csak egy sor kijelölése engedett, több adat megtekintése így ilyen módon sajnos még nem lehetséges .



kép #07 - Movies tábla

Egyik lehetőség a filmek törlése a **Delete** (Törlés) gombbal. Ekkor ki kell hogy legyen választva egy film amit törölni, rá kell kattintani a Delete gombra, a rendszer kiírja hogy biztos törölni szeretnénk e, és ha igen, akkor a rendszer már törli is a listából azt a filmet és az összes hozzá kapcsolódó adatát.

Ott van még az **Edit** (Szerkeszt) gomb, ami úgy működik, hogy arra a filmre kell kattintani amit szerkeszteni szeretnénk és felhossa a szerkesztő oldalt, ahol ez meg is valósítható. Maga a gomb nem hozza be az Edit ablakot, tehát mondhatni hogy díszként funkcionál de szerepe az az, hogy mutassa a felhasználónak, hogy ilyen opció is lehetséges, és felette meg is találja angolul a leírást, hogy hogyan tudja előhozni az Edit táblát.

Az oldal működés szempontjából úgy működik, hogy kattintással ki lehet választani, hogy melyik filmről szeretnénk információt szerezni és utána betöltődnek az adatok. Az adatok többsége látszódik a fő formon, és itt is lehetséges a **szűrési beállítások** például ha az IMDB score-ra vagy a Release Date-re kattintunk akkor automatikusan csökkenő illetve növekvő sorrendbe állítja a filmeket így megkönnyítve a kiszemelt film megtalálását.

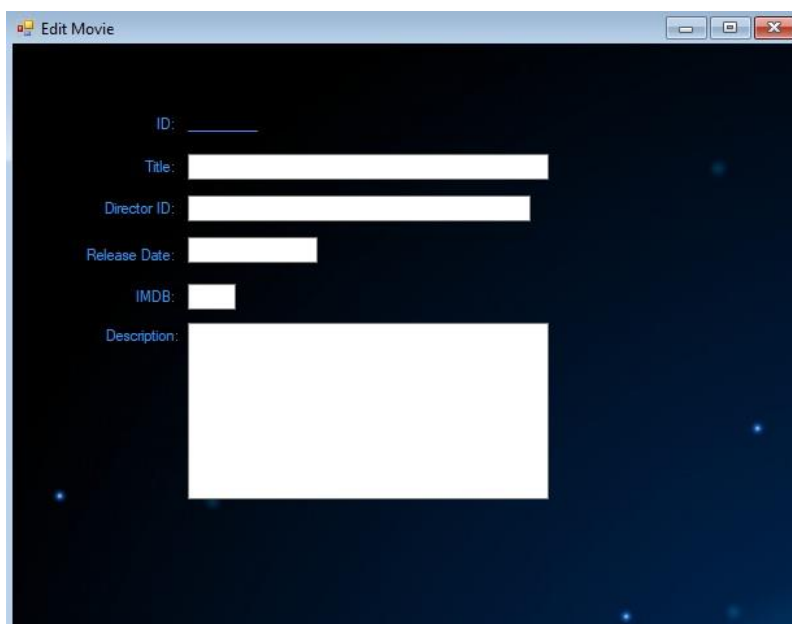
A fő betöltődő formon található a filmnek az azonosítója (Movie ID), a címe (Movie Title), a Rendező neve (Director), Release Date (Megjelenési dátum) és végül IMDB rating (pontszám). A képernyő közepén legalul található a Description (Leírás), míg a fő adatfolyamatok és a Description között a kiválasztott film címe jelenik meg a Selected Movie: után.



### II.2.1.3 – Asztali Alkalmazás: Edit Movies

Ez az oldal akkor érhető el, hogy ha az adminisztrátor vagy duplán rákattint egy filmre vagy a **New** gombra kattint. Első esetben a kiválasztott film adatai jelennek meg, míg a New gomb esetén egy üres adatokkal teli Edit ablak jelenik meg, amit ki kell töltenie.

Miután a felhasználó kitöltötte ez az adatlapot utána a **Save** (Mentés) gombra kell rákattintani és az adatok elmentődnek az adatbázisban. Ha a felhasználó mégse szeretne módosítani akkor a jobb felső sarokban lévő x-re kattintva kiléphet a módosítási ablakból.



*Kép #08 - Edit Movie oldal*

### II.2.1.4 – Asztali Alkalmazás: Users

A Users táblának a lényege, hogy megjelenítse a regisztrált felhasználóknak majdnem az összes adatát, amit a regisztráció során megadtak. Az itt megtalálható adatok:

- **User ID** (Felhasználót azonosító sorszám)
- **Username** (Felhasználónév) – Ez minden esetben csak egyedi lehet, nem osztható több felhasználó ugyanazon a néven
- **Password** (Jelszó): Csak is kódolva (Hashelve) jelennek meg, tehát az Admin ezen keresztül nem láthatja a jelszót biztonsági célból, a megtekintéshez az adatbázisból közvetlen kell az adatokat lekérdezni és lefordítani.
- **E-mail cím:** a megadott E-mail cím amire a vásárlás során el lesz küldve a Streaming Link a filmhez.

Movies	Users	Orders	
User ID	Username	Password	E-mail
1	admin	admin	admin@gmail.com
3	asd	asd	asd@gmail.com
4	aaa	aaa	aaa@gmail.com
5	simpleuser	simplepassword	simpleemail@gmail.com

kép #09 - Users tábla példa adatokkal

### II.2.1.5 – Asztali Alkalmazás: Orders

Az Orders tábla lényege, hogy meg lehessen tekinteni, hogy jelenleg milyen rendelések voltak eddig megtéve.

Az alábbi adatok szerepelnek ezen az oldalon:

- **Order ID** (Rendelés Azonosító): Magát a rendelést azonosító sorszám. Felhasználót, kiválasztott filmet és a rendelés időpontját veszi figyelembe, több film egyszeri megrendelésénél egyedül a movieId fog változni.
- **User ID** (Felhasználó Azonosító): Azt jelöli, hogy melyik User Id-val rendelkező felhasználó adta le a rendelést.
- **Movie ID** (Film Azonosító): A megrendelni kívánt filmet azonosítja.
- **Date** (Dátum): A megrendelés dátumát mutatja meg.

### II.2.1.6 – Asztali Alkalmazás: About Us & Exit

Az főképernyő jobb felső sarkában található két gomb. Az About Us-ra (Rólunkra) kattintva megjelenik az éppen futó program neve, verziószáma, készítés ideje és készítője. Az Exit gombra kattintva pedig teljesen kilép a programból.

## II.2.2 – FELHASZNÁLÓI DOKUMENTÁCIÓ: WEBES ALKALMAZÁS

A Movie Database internetes eléréséhez szükséges a XAMPP program folyamatos futtatása a megfelelő adatbázis importálása szintén szükséges hogy a program regisztráció, vásárlás és keresési funkciói elérhetőek legyenek.

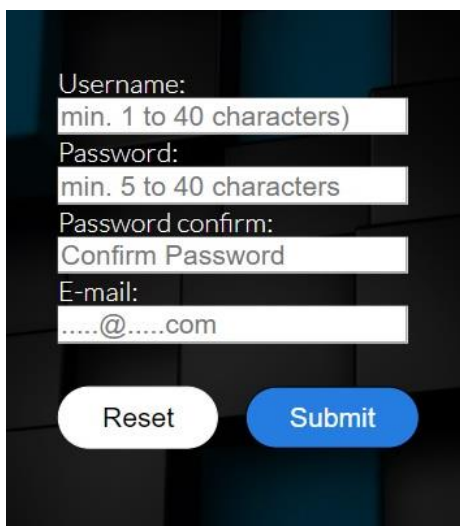
### II.2.2.1. – Webes Alkalmazás: Fő oldal és AboutUs oldal

Ezeknek az oldalaknak a fő célja, hogy bemutassa és ismertesse a felhasználót az oldal működésével. Az index oldal célja, hogy rávezesse a felhasználót a regisztrációra, és a keresésre. Az About Us oldal pedig ismerteti az oldalon elérhető funkciókat.

### II.2.2.2. – Regisztráció és Login

A regisztráció során kötelező kitölteni minden mezőt a megfelelő értékekkel, tehát ha hibásan tölti ki a regisztrálni kívánt felhasználó, akkor nem engedi a rendszer regisztrálni azokkal az adatokkal, amiket megadott. Különleges karakterek nem megengedettek. A kitölteni kívánt mezők a **regisztrációs űrlapon** (registration.php) a következők:

- **Username** (Felhasználónév): angol ABC betűit és 0-9-ig terjedő számokat fogadja el a rendszer, maximum 40 karakternyi hosszúságú lehet.
- **Password** (Jelszó): Minimum 5, Maximum 40 karakternyi hosszúságú lehet
- **Password Confirm** (Jelszó megerősítés): Ugyanannak kell lennie, mint a fentebb lévő password-nek
- **E-mail**: kell benne lennie kukac (@) jelnek és pontnak. Helyes példa: valaki@gmail.com

A screenshot of a web registration form. It features four input fields: 'Username:' with a hint 'min. 1 to 40 characters', 'Password:' with a hint 'min. 5 to 40 characters', 'Password confirm:' with a hint 'Confirm Password', and 'E-mail:' with a hint '.....@.....com'. Below the fields are two buttons: a white 'Reset' button and a blue 'Submit' button. The background is dark and textured.

kép #10 - Webes Regisztráció képernyő 1

A webes admin belépéshez:

username: admin2

password: admin2

A regisztrációt a Submit (Küldés) gombra kattintva küldi el feldolgozásra. A reset (visszaállítás) gombbal törli az összes eddig bevitt adatot.

A **login** képernyőn csak a felhasználónevet és a jelszót kell helyes megadni a helyes belépéshez.

### II.2.2.3 – Webes Alkalmazás: Search Page, Basket Page

A Movie Database egyik legkomplexebb webes része. A felhasználó vásárolni tud rajta filmeket tetszés szerint. Egy felhasználó viszont 1 db-ot tud venni egy filmből, mivel nincs fizikai kópia, hanem egy streaming linket vesz, amit az E-mail címre küldünk el neki. Ez egyszerre természettudatos hozzáállást segíti és egyben költséghatékony is mivel nem kell raktárat fent tartani a filmek tárolására. A filmek ára egységesen 5\$ dollár, ez kérés esetén változtatható.

Ha a felhasználó a kosár ikonra rákattint, akkor bekerül a kosárba, amit a „**Place Order**” gombra kattintva már meg is rendelhet. Az **Empty Cart** gombbal törlésre kerül a kosár tartalma. A rendelést vagy az adatbázisban lehet megnézni, vagy az asztali alkalmazás Orders táblájában.

## II.3 – RENDSZERKÖVETELMÉNY

A program helyes működtetéséhez nem szükséges erős hardverrel rendelkező számítógép. A programok egy eredeti Windows 10 64bit Home operációs rendszeren teszteltem, ahol gond nélkül futott. Úgy vélem hogy egy Windows 7-es operációs rendszertől kezdődően felfelé már probléma és hiba nélkül futna a program minimális rendszerkövetelménnyel. Internet hiánya esetlegesen a weboldal néhány funkciójában tehet kárt, de mivel helyi hálózaton fut, ezért nem kellene, hogy problémát okozzon.

A legfőbb problémát a helytelen verziójú böngésző/futtató programok okozhatják, amik miatt lehetséges, hogy a program esetlegesen hibásan tölt le adatokat.

A minimum rendszerkövetelményeket így a Windows 7 minimum rendszerkövetelményeihez igazítottam:

- 1 gigahertz (GHz) vagy gyorsabb 32-bit (x86) vagy 64-bit (x64) processzor
- 1 gigabyte (GB) RAM (32-bit) vagy 2 GB RAM (64-bit)
- 16 GB elérhető merevlemezterület (32-bit) vagy 20 GB (64-bit)
- DirectX 9-et támogató grafikus meghajtó eszköz WDDM 1.0 vagy magasabb szintű driver

## **III. FORRÁSMEGJELÖLÉS**

### **III.1 – WEBOLDALAK, PROGRAMKÓDOK, SZÖVEGES FORRÁSOK MEGJELÖLÉSE**

- Install Forge – asztali alkalmazás telepítésénél használt program
- imdb.com -- a példaoldal. Poster képek és a leírások forrása
- Elvp2.vasvari.hu -- az oldalon fellelhető különböző hivatalos oktatási anyagok
- quickhighlighter.com – kódrészletek kiemelésére
- database.cs – Gyuris Csaba tanár úr által biztosított segédanyag

### **III.2 – KÉPEK FORRÁSAINAK MEGJELÖLÉSE**

#### **III.2.1 – Az adatbázisban tárolt képek**

Az összes filmhez kapcsolódó alapvető képet (postereket) az imdb.com weboldalról származnak, a megfelelő filmet megkeresve 2018. február 14-ei dátummal ellenőrizve.

#### **III.2.2. – Az asztali alkalmazáshoz használt képek**

Az asztali alkalmazáshoz tartozó képek forrásai:

- Login form háttérkép:  
<https://static.slo-tech.com/67143.jpg>
- Movie Project/Edit Movie/Movie/About Us/Order/Users háttérkép:  
<http://magazin.nekunkszol.hu/wp-content/uploads/2013/07/filmek2.jpg>

#### **III.2.3 – A webes alkalmazáshoz használt képek:**

A webes alkalmazáshoz tartozó képek forrásai:

- Index/About Us oldalak háttérképei:  
[http://static.tumblr.com/c6a6d577644c58db8da2f780981d62d9/iaqnu6g/1hsogi0kv/tumblr\\_static\\_ck4r2t6qqzccggg4cs44owcoc\\_2048\\_v2.jpg](http://static.tumblr.com/c6a6d577644c58db8da2f780981d62d9/iaqnu6g/1hsogi0kv/tumblr_static_ck4r2t6qqzccggg4cs44owcoc_2048_v2.jpg)
- Registration/Login oldalak háttérképei:  
[http://freshwallpapers.info/uploads/posts/2017-04/10\\_cube\\_abstract.jpg](http://freshwallpapers.info/uploads/posts/2017-04/10_cube_abstract.jpg)
- Movie.php/basket.php oldalak háttérképei:  
<https://hdwallsource.com/img/2014/7/matte-black-wallpaper-29553-30272-hd-wallpapers.jpg>

## **IV. KÖSZÖNETNYILVÁNÍTÁS**

Ezúton is szeretném megköszönni segítőkész tanárain tanácsait, építő jellegű kritikáit, melyek segítettek abban, hogy a szakdolgozat elérje a jelenlegi formáját. A munka során jelentkező szakmai kérdésben segítségemre voltak a témavezető tanáiraimon: Bálint Róbertnek és Gyuris Csabának.

## TANULÓI NYILATKOZAT

Alulírott **Pónusz Richárd** (tanuló neve), Szegedi Szakképzési Centrum Vasvári Pál Gazdasági és Informatikai Szakgimnáziuma tanulója kijelentem, hogy a „**Filmes adatbázis - Filmkeresés adatbázisban és rendelés**” című záródolgozat a saját munkám.

Kelt:

---

aláírás