Gala Mozi Webalkalmazás

Készítő:
Papp Edina

Tartalomjegyzék

1. Bevezetés

1.1. Témaválasztás indoklása

Világunkban jelenleg virágzik a filmipar, tele vannak a mozik, sorra készülnek a magas költségvetésű, sikeresebbnél sikeresebb filmek, emiatt záródolgozatom témájául egy fiktív mozi honlapjának elkészítését választottam, mely egy modern és felhasználóbarát platformot nyújt a filmek kedvelőinek.

A projekt által megcélzott probléma a mozinál tapasztalt hosszú sorok, és az ezzel járó időveszteség. Ugyanis, ha előzetes tájékozódás nélkül érkezünk egy mozi elé, problémát jelenthet a megfelelő film és ülőhely kiválasztása, de fennáll az a lehetőség is, hogy érkezésünkkor az általunk kiválasztott előadásra már elfogytak a jegyek. Előbbi rossz filmélményt, utóbbi pedig frusztrációt eredményezhet. Míg, ha az otthonunk kényelméből, érkezés előtt felmérjük a kínálatot és akár még jegyet is foglalunk, a folyamat sokkal kényelmesebb, és a kiválasztott film is nagyobb eséllyel nyeri el a tetszésünket.

Projektem célja tehát a moziélmény könnyítése, hiszen ez mind a mozi üzemeltetője, mind a vendégek számára előnyös.

2. Fejlesztői dokumentáció

2.1. Fejlesztői környezet

Fejlesztői környezetként a Visual Studio Code-ot választottam, hiszen számos előnye miatt ideális a webfejlesztéshez. Elsőként kiemelendő a rendkívüli gyorsasága, hiszen minimális erőforrás igénye miatt a szoftver gyorsan indul.

Az, hogy a Visual Studio Code nyílt forráskódú, lehetőséget teremt a széleskörű testreszabásra és bővíthetőségre. Ennek köszönhetően könnyedén hozzáadhatunk különböző kiegészítőket a Visual Studio Code Marketplace-ről, ha fejlesztés közben szükségünk lenne rá.

A Visual Studio Code kiemelkedő támogatást nyújt a webfejlesztéshez, lehetővé téve HTML, CSS, JavaScript és PHP dokumentumok kényelmes szerkesztését. Ezen felül beépített Git támogatással rendelkezik, amely megkönnyíti a munkánk verzióinak követését.

Integrált hibakeresővel is rendelkezik, amely segítséget nyújt a kódban található hibák felderítésében.

A webalkalmazás jellege miatt az adatbáziskezelés is igen nagy szerepet játszik a működésben, így ennek megfelelően adatbázis-kezelő szoftvert is kellett választanom. A phpMyAdminra esett a választásom, hiszen ezzel már korábban is dolgoztam, és tapasztalatom szerint felülete könnyen átlátható.

2.2. Használt nyelvek, technológiák

Az alkalmazás fejlesztése során több különböző nyelvet és technológiát használtam, a minél hatékonyabb és esztétikusabb végeredmény elérése érdekében. Ezek között van a HTML, CSS, Bootstrap, PHP és Javascript.

2.2.1. HTML

A HTML, azaz a HyperText Markup Language (magyarul HiperSzöveg Jelölési Nyelv), az internetes tartalom strukturálására, jelölésére és megjelenítésére szolgáló kulcsfontosságú

webes technológia. Az HTML az internetes világ egyik alapvető eleme, hiszen meghatározza a weboldalak működését és külső megjelenését.

A HTML kifejezetten strukturált nyelv, amely jelöléseket használ, hogy meghatározzon egyes elemeket, például címsorokat, bekezdéseket, képeket és linkeket. A HTML elemei hierarchikusan rendeződnek, és egy dokumentumot alkotnak, ez képezi egy weboldal alapját. A böngészők szerepe, hogy értelmezzék, majd megjelenítsék a kódot.

A webalkalmazások fejlesztése során a HTML fontos szerepet játszik, mivel felelős a felhasználói felület szerkezetéért és megjelenítéséért. Az HTML-t keverhetjük más technológiákkal, (pl. CSS és Javascript), hogy dinamikusabb és interaktívabb felhasználói élményt biztosítson.

A projektem szempontjából a HTML szerepe lesz a felhasználói felület kialakítása, hiszen ezen keresztül fogom létrehozni az oldal alapstruktúráját, illetve határozom meg a szövegek elrendezését, illesztem be a képeket, hivatkozásokat és egyéb elemeket, mint például beviteli mezőket és gombokat.

2.2.2. CSS

A CSS, vagyis a Cascading Style Sheets, egy olyan stílusleíró nyelv, amelyet a weboldalak megjelenítésének formázására használunk.

Segítségével meghatározhatjuk a weboldal színeit, betűtípusait, háttérképeit, méreteit, térközeit és egyéb vizuális elemet. Készíthetünk vele animációkat és átmeneteket, és megadhatunk különböző tulajdonságokat felhasználói interakcióktól függően (pl.: hover, kattintás). Továbbá lehetőségünk van a már meglévő HTML elemeink elrendezésére, pozícionálására.

Az oldal reszponzivitásában is fontos szerepet tölt be, hiszen media querykkel szabályokat adhatunk meg, amik lehetővé teszik, hogy az oldalunk alkalmazkodjon a különböző kijelzőméretekhez.

Az oldalam formázására külső stíluslapot fogok alkalmazni, hiszen ez nagyobb projekt esetén segíti az átláthatóságot és megkönnyíti a módosítást.

2.2.3. Bootstrap

A Bootstrap egy ingyenes és nyílt forráskódú CSS keretrendszer, amelyet a Twitter fejlesztett ki. A fő célja, hogy segítse a fejlesztőket az egyszerű és gyors responszív weboldalak kialakításában. A Bootstrap előnye, hogy előre elkészített elemeket és komponenseket kínál, amelyeket könnyen beilleszthetünk weboldalunkba, anélkül, hogy a minden részletet nekünk kellene létrehozni.

A keretrendszer tartalmaz olyan többek között gombokat, űrlapokat, navigációs sávokat, kártyákat, amelyek egységes és esztétikus megjelenést kölcsönöznek az alkalmazásnak. Emellett a Bootstrap fejlesztői számára responszív tervezési lehetőségeket is biztosít, így az alkalmazások könnyen alkalmazkodhatnak különböző eszközök és képernyőméretek között.

A Bootstrap szerepe projektem során az oldal esztétikus, felhasználóbarát megjelenésének és reszponzív viselkedésének kialakítása a beépített stílusok és a grid system segítségével.

2.2.4. PHP és Javascript

A weboldal funkcionalitását két nyelv fogja biztosítani: a PHP és a Javascript.

PHP (Hypertext Preprocessor) egy programozási nyelv, melyet elsősorban dinamikus weboldalak és alkalmazások fejlesztéséhez használnak. Az elnevezése már önmagában is utal a szerepére, mivel az Hypertext Preprocessor azt jelzi, hogy a nyelv feldolgozza és előkészíti a hypertextet, vagyis a HTML-t, mielőtt azt a böngésző megjeleníti a felhasználó számára.

Szerveroldali programozási nyelv, ami azt jelenti, hogy a kód a szerveren fut, és ez a szerver generálja a válaszokat, amiket a böngésző eljuttat a klienshez.

A PHP legfontosabb szerepe a projektemben az adatbázissal való kapcsolata lesz, hiszen lehet vele adatokat lekérdezni, módosítani, vagy az adatbázisból adatokat beilleszteni. Dinamikus tartalom generálására is alkalmas, ilyen például az előbb említett adatbeillesztés. Emellett lehetőséget nyújt a felhasználói interakciók kezelésére, űrlapok feldolgozására, sütik kezelésére és más olyan funkciókra, amelyek segítik a webalkalmazások fejlesztését.

Az oldal PHP részét a XAMPP nevű szoftvercsomag segítségével fogom tesztelni.

A Javascript a PHP-val ellentétben kliensoldali programozási nyelv, tehát a böngészőben (kliensoldalon) fut. Ez teszi lehetővé, hogy interaktív felhasználói felületeket alakítsunk ki, dinamikus funkciókkal. A Javascript arra is jó, hogy kezeljük a felhasználói bemeneteket, eseményeket (pl.: kattintás, egérmozgatás), és ennek megfelelően frissítsük az oldal tartalmát. Aszinkron kommunikációra is használhatjuk, ami lehetővé teszi, hogy az oldal újratöltés nélkül frissüljön.

2.3. Adatbázis tervezet

A program megfelelő működéséhez egy adatbázisra és kilenc darab táblára lesz szükség. Minden tábla első mezője egy int típusú *id* mező, ami a tábla elsődleges kulcsa, és AUTO INCREMENT tulajdonsággal rendelkezik, vagyis automatikusan jön létre a felvitel során.

2.3.1. Rendezo, mufaj és besorolas tábla

A **rendezo**, **mufaj** és **besorolas** nevű táblák felépítése szinte megegyezik, hiszen mindhárom két mezőből áll. Első mezőjük egy *id* mező, második pedig a **rendezo** és **mufaj** tábla esetében varchar, a **besorolas** tábla esetében pedig int típusú, és a tábla nevével megegyező információt adják meg.

2.3.2. Film tábla

A **film** nevű adattábla a moziban vetített filmek adatait tartalmazza és nyolc mezővel rendelkezik. Az első az *id* mező, a második a *cim*, ami varchar típusú és a film címét tartalmazza. A harmadik a *leiras* mező, text típussal, ahova a film rövid leírása kerül. A *hossz* és *ev* mező int típusú, előbbibe a film hossza adható meg percben, míg utóbbiba a film megjelenésének időpontja. Az *url* mezőben a filmhez tartozó kép neve szerepel. Ezek után jönnek azon mezők, melyek egy másik táblához kapcsolják a film táblát. A *rendezoId* a **rendezo** tábla *id* mezőjéhez, a *mufajId* a **mufaj** tábla *id* mezőjéhez, a *besorolasId* pedig a **besorolas** tábla *id* mezőjéhez kapcsolódik. Ezek a mezők teszik lehetővé, hogy a filmek között rendező, műfaj és korbesorolás szerint böngésszünk.

2.3.3. Vetites tábla

A **vetites** táblában tároljuk a moziban aktuális vetítések adatait. A táblában megtalálható egy *id* mező, *datum* mező date típussal, egy *ido* mező time típussal, és egy *filmId* és *teremId* nevű mező. Utóbbi kettő összeköttetésben áll másik táblákkal. A *filmId* a **film** tábla *id* mezőjével, a *teremId* pedig a **terem** tábla *id* mezőjével.

2.3.4. Terem tábla

A **terem** tábla a moziban megtalálható termeket tárolja egy *id* és egy *ferohely* mező segítségével. Ez a tábla lehetővé teszi számunkra, hogy tisztában legyünk egy adott előadásra vonatkozó maximum eladható jegyek számával.

2.3.5. Felhasznalo tábla

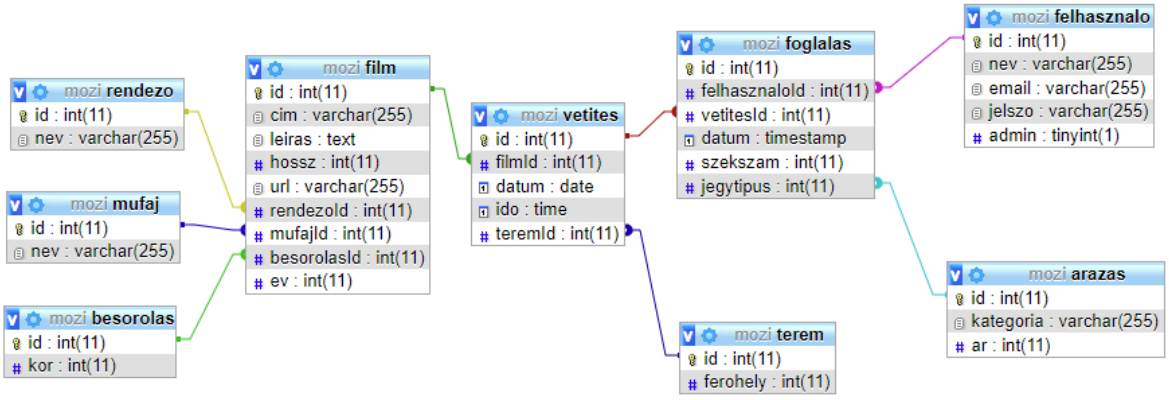
A webalkalmazás biztosítja a felhasználók adatainak tárolását, így egy **felhasznalo** nevű táblára is szükségünk van. A tábla tartalmaz egy *id* mezőt, egy *nev, email* és *jelszo* mezőt varchar típusban, ahol a felhasználónév, email cím és jelszó tárolódik. A táblában található egy *admin* mező tinyint típusban, melynek beállításával (0 vagy 1), megadhatjuk, hogy a felhasználó rendelkezik-e admin jogosultsággal vagy sem.

2.3.6. Foglalas tábla

A **foglalás** táblában tároljuk a felhasználók által tett foglalásokat. A táblában található egy *id* mező, egy *felhasznaloId* mező, ami a **felhasznalo** tábla *id* mezőjével van kapcsolatban. A *vetitesId* nevű mező a **vetites** tábla *id* mezőjével áll kapcsolatban. Szeretnénk tárolni, hogy melyik foglalás mikor történt, így van egy timestamp típusú *datum* mezőnk is. Az int típusú *szekszam* mezőben tároljuk, hogy hányas széket foglalta le a felhasználó, a *jegytipus* nevű mezőben pedig az árkategória van meghatározva, az **arazas** tábla *id* mezőjével való kapcsolat által.

2.3.7. Arazas tábla

Az **arazas** tábla a jegykategóriák tárolására szolgál, három mezője az *id*, a varchar típusú *kategoria* és az int típusú *ar* mező.



Mozi adatbázis táblái és azok kapcsolatai

2.4. Szerkezet

2.4.1. Változók

A program működéséhez elengedhetetlen a szuperglobális változók használata.

A bejelentkező űrlap elküldése POST metódussal történik, amint a felhasználó a bejelentkezés gombra kattint. Ezután a $_POST változót használom az űrlap által elküldött adatok elérésére. Ezek az értékek aztán részt vesznek a bejelentkezési folyamatban és az adatbázisból való lekérdezésben.

A felhasználó bejelentkezésekor elindul egy munkamenet a session_start() utasítással, ami lehetővé teszi a $_SESSION változók használatát. Ezzel valósul meg a felhasználó adatainak tárolására, továbbá a felhasználó típusának (felhasználó vagy admin) megállapítása. A későbbiekben egy $_SESSION['user_id'] (vagy admin esetén $_SESSION['admin_id']) nevű változóval történik meg a felhasználó azonosítása.

Az oldalak közötti információátvitelt $_GET változók segítségével végzem, ilyen például a filmek és vetítések azonosítójának átadására URL-en keresztül.

2.4.2. Osztályok

A weboldal fejlesztése során elengedhetetlen volt az osztályok használata, mind Javascript aszinkron adatbázisműveleteknél, mind az egyszerű PHP adatbázisműveleteknél. Aszinkron kérést abban az esetben alkalmaztam, ha valamilyen bonyolultabb felhasználói művelet történt, vagy dinamikus tartalom megjelenítésére volt szükség, ezzel szemben PHP-ban történő műveleteket akkor végeztem, amikor egyszerűbb felhasználói interakciók történtek. A Javascriptben és PHP-ban történő adatbázis-műveletekhez egymástól független osztályokat használtam.

A PHP adatbázisműveletek végrehajtásához két osztályt alkalmaztam: az **Adatbazis** osztály felel az adatbázis-kezelésért, míg a **Mozi** osztály az alkalmazási logikát valósítja meg.

Az **Adatbazis** osztály a konstruktorban inicializálja az adatbáziskapcsolatot, és két fő metódussal rendelkezik: adatokLekerese() és Modosit(). Az adatokLekerese() segítségével adatokat lekérdezhetünk az adatbázisból, míg a Modosit() módosító műveleteket hajt végre az adatbázison, például beszúrás vagy frissítés esetén. Az osztály képes kezelni az SQL hibákat is, biztosítva a megbízható adatbázis-kezelést.

A **Mozi** osztály egyszerűbb felhasználói műveleteket kezel, például bejelentkezést és regisztrációt a login() és register() metódusokkal. Emellett filmekkel kapcsolatos műveleteket is végrehajt, mint például filmek lekérése. Az osztály az **Adatbazis** osztályt használja az adatbázis-kezeléshez, így különválasztja az alkalmazási logikát és az adatbázis-interakciót, ami javítja a kód átláthatóságát és karbantarthatóságát.

Az aszinkron kérésekhez további három osztályt alkalmaztam.

Az **Adatbazis** osztály felelős az olyan adatbázis-műveletek végrehajtásáért, mint az adatok lekérése vagy módosítása. A Modosit() metódus az adatbázisra vonatkozó módosításokat végzi, például INSERT, UPDATE vagy DELETE parancsokat hajt végre, majd visszatér a művelet eredményével. A adatLekeres() metódus az adatbázisból történő adatlekéréseket végzi, például SELECT parancsokat, és visszatér az eredményhalmazzal.

A **Mozi** osztály az alkalmazási logikát valósítja meg, és különböző funkciókat rendelkezik, mint például a foglalas_rogzites(). Ez a metódus a foglalások rögzítését végzi az adatbázisban. A rögzítéshez szükséges adataok a felhasználói azonosító, a vetítés azonosítója, a szék száma és a jegy típusa. A művelet végrehajtása után a metódus visszaadja a művelet eredményét JSON formátumban.

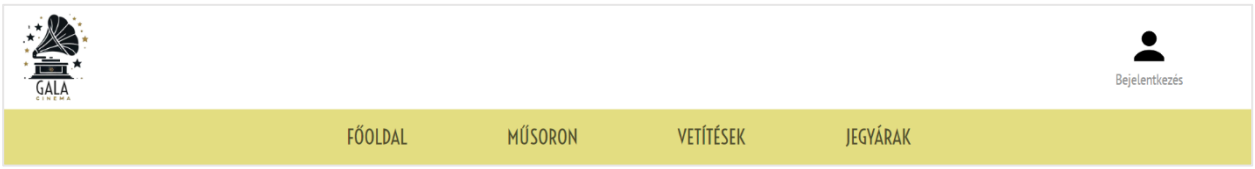
Az **Utvonal** osztály felelős az URL-ek értelmezéséért és az azokhoz tartozó funkciók kiválasztásáért. Az URL végétől függően kiválasztja és meghívja a megfelelő metódust a **Mozi** osztályból, majd visszaadja az eredményt.

Ez a struktúra lehetővé teszi a weboldal aszinkron JavaScript kérésekkel kapcsolatos könnyű kezelését, az adatok hatékony lekérdezését és módosítását az adatbázisban, valamint a felhasználói interakciók kezelését és az adminisztrációs funkciók biztosítását.

2.4.3. Oldal szerkezete

Az webalkalmazás minden aloldal esetén három szerkezeti elemből áll: *header*, *main* és *footer*.

A *header* elem kétféleképpen nézhet ki, attól függően, hogy a felhasználó be van-e jelentkezve. Ha bejelentkezés nélkül böngészi az oldalt, a jobb felső sarokban lévő ikon alatt a "Bejelentkezés" szó található. Viszont, ha a felhasználó már be van jelentkezve, a jobb felső sarokban az ikon alatt a kijelentkezés szó található, illetve az ikonra kattintva megtekintheti az addigi foglalásait. Ezen kívül a header elem minden esetben tartalmazza az oldal logóját, melyre kattintva a felhasználó a főoldalra lesz irányítva, és egy navigációs sávot, melyen az oldal menüpontjai találhatóak. Ez segít a felhasználónak az oldal elemei között eligazodni.



Az oldal navigációs sávja, bejelentkezett felhasználó nélkül

Az oldal *main* részében találhatóak a fő tartalmi elemek. A "Főoldal" esetén ez egy *carousel* elem, amiben aktuális filmek reklámját, promóciókat tekinthet meg a felhasználó. A "Műsoron" menüpont egy rácsszerkezetet tartalmaz két oszloppal, melyek egyikében a mozi által aktuálisan vetített filmeket, illetve ezek adatlapját tekinthetjük meg. A másikban az említett filmek között tudunk szűrést végezni rendező, műfaj és korhatár-besorolás szerint. A "Vetítések" menüpont tartalmaz egy idővonalat 10:00-21:00 óráig. Ezen idővonal mentén jelennek meg először az aznapi, majd keresés után a választott időpontban vetített filmek. A "Jegyárak" menüpont a kétféle kategóriájú jegy megnevezését és árát tartalmazza, táblázatba foglalva.

A *footer* elem minden esetben az oldal alján található, és a szerzői információt, vagyis a készítő nevét adja meg.

A *header* és *footer* elemek az **include** paranccsal kerülnek megadásra az oldal minden részében.

2.5. Algoritmusok

2.5.1. Regisztráció

Az oldal működése során végig szükség van adatbázisra, így a regisztráció során felvitt adatok is itt tárolódnak. Az adatbázishoz szükséges SQL műveleteket és függvényeket két külön fájlban (sql_muveletek.php, sql_fuggvenyek.php) tárolom el.

A regisztráció folyamata akkor kezdődik meg, amikor a felhasználó rákattint a **$_POST['button_reg']** hivatkozású gombra. Ezután megvizsgálásra kerül, hogy van-e olyan mező, amely nem került kitöltésre. Amennyiben ez az állítás igaz, egy *echo 'Kérem töltse ki a megadott mezőket!'* üzenettel jelezzük ezt a felhasználó felé. Ha a felhasználó minden mezőt kitöltött, a program megvizsgálja, hogy a kitöltés megfelelően történt-e meg. Deklarálunk egy **error** nevű változót, amelybe a kitöltési hibák száma kerül. A hibák lehetnek: a név mező teljes nevet tartalmaz-e, a két jelszó egyezik-e, a jelszó hossza legalább nyolc karakter. Ha a változó tartalma nulla, az adatok felvitelre kerülnek az adatbázisba. A jelszó titkosítása az adatbázisba való beszúráskor történik, SHA2 függvénnyel. Az erre vonatkozó SQL parancs a fent említett sql_muveletek.php-ban található.

2.5.2. Bejelentkezés

A bejelentkezési folyamat szintén gombhoz kötött. Amint a **button_login** megnyomásra kerül, kezdetét veszi az ellenőrző folyamat. Ha mindkét mező ki lett töltve, a **$login_data** nevű változóba lekérjük az adatbázisból annak a felhasználónak az adatait, amelyben az email cím egyezik a bejelentkezésnél megadottal. Ha a $login_data egy tömb, akkor elindítjuk a munkamenetet, majd megnézzük, hogy a megadott és a tárolt jelszó egyezik-e. Ha igen, újabb vizsgálat következik, melyben megállapításra kerül a felhasználó típusa. Ha admin, akkor az oldal az admin.php oldalra irányítja tovább, ellenkező esetben pedig a home.php-ra.

2.5.3. Filmkínálat megjelenítése

A filmkínálat megjelenítésére vonatkozó kód a movies.php nevű fájlban található. A megfelelő menüpontra kattintva az oldal betöltődésekor megkezdődik a filmek lekérése. Az adatok lekérése aszinkron kéréssel és adatbázis lekérdezésekkel történik. Az oldal betöltése után szűrhetjük a filmeket műfaj, rendező és korhatár-besorolás szerint. Ez szintén aszinkron kéréssel történik. A szűrés folyamatánál négy Javascript függvény van jelen: rendezoLeker(), mufajLeker(), korLeker(), kereses(). Az első hárommal feltöltjük azon <select> elemeket, amelyekkel később a szűrést végezzük, utóbbival pedig történik a szűrés.

Minden film plakátját és címét egy külön elemben jelenítjük meg. A elemekre egy, a film azonosítóját is tartalmazó, a film.php oldalra mutató hivatkozás kerül (<a href="film.php?id='.$x['id'].'">), ez teszi lehetővé, hogy egyesével meg lehessen tekinteni a filmek adatlapját.

2.5.4. Vetítések megjelenítése

A vetítések megjelenítése a filmkínálat megjelenítéséhez hasonlóan történik. Az erre vonatkozó kód a vetitesek.php-ban található. Az async function megjelenit() függvény a backend-től lekéri az adatokat az aktuális dátum alapján, majd dinamikusan frissíti a megjelenített filmeket az oldalon. A vetítések minden esetben, időpontjuk szerint kerülnek

elhelyezésre, az oldal bal oldalán található idővonal megfelelő részébe. Az oldalon tizenkettő "orak" osztállyal ellátott elem található, melyeken belül újabb elem helyezkedik el, azonosítója pedig egy szám tíztől-huszonegyig, ami az órát jelöli. Ez a jelölés teszi lehetővé, hogy a vetítések a megfelelő helyre kerüljenek. A film plakátját, címét és a vetítés időpontját jelenítjük meg. A film plakátjára és címére egy, a film adatlapjára mutató hivatkozást helyezünk (<a href="film.php?id='.$x['filmid'].'">), az időpontra pedig egy, a foglaláshoz átirányító linket (<a href="vetites.php?id='.$x['id'].'">).

2.5.5. Foglalás menete

A foglalás menete három oldalon keresztül történik: vetites.php, reservation.php, osszegzes.php. Az oldalak között az adatok átadása POST metódussal történik. A vetites.php oldalon a felhasználó megadja a foglalni kívánt jegyek számát. A "Tovább" gombra kattintva ezek az adatok átadásra kerülnek a reservation.php oldalnak, ahol a felhasználó kiválaszthatja ülőhelyeit. Fontos, hogy egy széket csak egy ember foglalhasson le, így ez egy adatbázis-lekérdezéssel ellenőrzésre kerül. Az ülőhely kiválasztása után a felhasználó az osszegzes.php oldalra érkezik, ahol összesítve láthatja a lefoglalni kívánt jegyeket, azok székszámát és a végösszeget. A "Foglalás véglegesítése" feliratú gombra kattintva az adatok aszinkron kéréssel felvitelre kerülnek az adatbázisba. Sikeres foglalás esetén a felhasználó a success.php oldalra kerül átirányításra.

2.5.6. Felhasználó foglalásainak megjelenítése

Betöltésekor a program lekérdezi a Mozi osztály segítségével a felhasználó által foglalt mozijegyeket a user_foglalasok() metódussal, és megjeleníti azokat a táblázatban. Minden sor tartalmazza a film címét, a vetítés dátumát, a székszámot, a foglalás dátumát, az árat és egy "Törlés" gombot, amivel a felhasználó a foglalását tudja lemondani. Ha a foglalás dátuma korábbi, mint az aktuális dátum, akkor a "Törlés" gomb letiltásra kerül, annak érdekében, hogy a felhasználó ne tudja törölni a már lezajlott vetítésekre tett foglalásokat. A "Törlés" gombra kattintva a foglalasTorles(elem) függvény fut le. Ez először megerősítést kér a felhasználótól. Ha a megerősítés megtörtént, elküldi a törölni kívánt foglalás azonosítóját a szervernek. A szerver válaszát megkapva, ha a művelet sikeres, az oldal újratöltődik.

2.6. Admin felület

Az adminisztrátor felületet a sima felhasználói felülethez hasonlóan, bejelentkezéssel érhetjük el. A felületen lehetőségünk van kezelni a felhasználókat, vetítéseket, filmeket.

2.6.1. Filmek kezelése

A felület első aloldala a "Filmek" menüpont alatt található movies_admin.php. Az oldalon egy táblázatban jelennek meg a rendelkezésre álló filmek adatai: a cím, a leírás, a hossz, a rendező, a műfaj, a korhatár és az év. Az adatok megjelenítése, módosítása és törlése aszinkron kérésekkel történik az adatbázisból. Minden filmhez tartozik egy törlés és egy módosítás gomb, amelyekkel az admin törölheti vagy módosíthatja a filmek adatait.

A film törlése gombra kattintva először egy megerősítő üzenet jelenik meg. Ha az adminisztrátor megerősíti a törlést, akkor a rendszer ellenőrzi, hogy a filmhez tartoznak-e vetítések vagy foglalások. Ha vannak, akkor először azokat törli, majd magát a filmet is. Ez az adatbázisban található kapcsolatok miatt lényeges. Ha a törlés sikeres volt, az oldal automatikusan frissül, hogy az adminisztrátor láthassa a frissített filmek listáját.

A film módosítása gombra kattintva az adminisztrátort átirányítja az oldal a film adatainak szerkesztésére szolgáló oldalra, ahol lehetőségük van a film adatainak módosítására.

Az oldal felső részében két további gomb található, amelyekkel az adminisztrátorok új rendezőt vagy új filmet adhatnak hozzá az adatbázishoz. Mindkét gombra kattintva az adminisztrátorok egy új oldalra lesznek átirányítva, ahol megadhatják az új rendező vagy film adatait.

2.6.2. Vetítések kezelése

A vetítéseket a "Vetítések" menüpontra kattintva tudja az adminisztrátor kezelni. Az erre vonatkozó kód a vetitesek_admin.php fájlban található. Működése hasonló a movies_admin.php-on található filmek kezeléséhez. Az oldalon található táblázatban jelennek meg a vetítések adatai: a cím, a dátum és a pontos idő. Minden vetítéshez tartozik egy törlés és egy módosítás gomb, amelyekkel az adminisztrátor törölheti vagy módosíthatja a vetítések adatait.

A vetítés törlése gombra kattintva először egy megerősítő üzenet jelenik meg, amelyben az admin megerősítheti a törlés szándékát. Megerősítés esetén a rendszer először ellenőrzi, hogy a vetítéshez tartoznak-e foglalások. Ha igen, akkor először azokat törli, majd csak ezután a vetítést. Ha a törlés sikeres volt, az oldal automatikusan frissül, hogy az adminisztrátor láthassa a frissített vetítések listáját.

A vetítés módosítása gombra kattintva az adminisztrátorok átirányítást kapnak a vetítés adatainak szerkesztésére szolgáló oldalra, ahol lehetőségük van a vetítés adatainak frissítésére.

Az oldal felső részében található egy gomb, amely segítségével az adminisztrátor új vetítést adhat hozzá az adatbázishoz. A gombra kattintva az adminisztrátor egy új oldalra lesz átirányítva, ahol megadhatja az új vetítés adatait.

Az adatlekérések az adatbázisból aszinkron módon történnek a JavaScript fetch API segítségével, amely lehetővé teszi az adatok aszinkron betöltését és megjelenítését az oldalon anélkül, hogy újratöltődne az egész oldal. Ez a megközelítés felhasználóbarátabb élményt nyújt az adminisztrátornak, és növeli a webalkalmazás hatékonyságát.

2.6.3. Felhasználók kezelése

A felhasználók kezelését a "Felhasználók" menüpont alatt tehetjük meg. Az oldal fő tartalma egy táblázat. Ide történik a rendszer összes felhasználójának kilistázása, azonosítóval, a névvel és e-mail címmel. Minden felhasználó sorában található egy "Törlés" gomb, melynek segítségével az adminisztrátor törölheti az adott felhasználó adatait az adatbázisból. Amikor az adminisztrátor a "Törlés" gombra kattint, először egy megerősítő ablak jelenik meg, amelyben az adminisztrátor megerősítheti a törlés szándékát. Ha az adminisztrátor megerősíti a törlést, a rendszer először ellenőrzi, hogy a felhasználóhoz kapcsolódnak-e foglalások. Ha igen, akkor azokat törli először, majd magát a felhasználót is. Ha a törlés sikeres volt, az oldal automatikusan frissül, hogy az adminisztrátor láthassa a frissített felhasználók listáját.

3. Felhasználói dokumentáció

3.1. Hardver- és szoftverkövetelmények

A fejlesztés során az egyik kulcsfontosságú lépés a hardver- és szoftverkövetelmények meghatározása. Ez a folyamat nélkülözhetetlen a projekt megfelelő működése szempontjából. A hardverkörnyezet kiválasztása a processzor, a memória és a tárhelyre vonatkozó követelményeket foglalja magában.

Ami a processzort illeti, többmagos processzorra van szükség, amely képes párhuzamos feldolgozásra. Ezen kívül, a megfelelő mennyiségű RAM biztosítása szintén fontos, hiszen ez határozza meg a futó alkalmazások és folyamatok kezelésének hatékonyságát. Ezen projekt esetében minimum 8 GB-ra van szükség. A gyors adatelérés és feldolgozás érdekében pedig SSD alapú tárhelyre van szükség, ideálisan legalább 256 GB kapacitással.

A szoftverkövetelmény legfontosabb része a fejlesztői környezet, hiszen ennek segítségével szerkeszthető és nyitható meg a kód. A Microsoft által fejlesztett Visual Studio Code tökéletes választás erre a célra, hiszen a JavaScript, HTML és PHP kódok hatékony kezelhetőek vele.

A projektem tartalmaz szerveroldali programozási nyelvet, így egy web szerver is elengedhetetlen, enélkül elképzelhetetlen lenne a dinamikus webalkalmazás létrehozása és tesztelése saját gépen. Erre a feladatra megfelel a XAMPP szoftvercsomag, mivel egyszerű telepítése révén könnyen integrálható az Apache web szerverrel, a MySQL adatbáziskezelő rendszerrel és a PHP-val.

A program adatait adatbázisban tároljuk, így ezek kezelésére is szükség van egy adatbáziskezelő felületre. Ilyen a phpMyAdmin, melynek felülete átlátható, segítségével könnyedén kezelhető és megtekinthető az adatbázis és egyszerű a különböző műveletek végrehajtása.

A webböngésző számos fontos funkciót tölt be egy webalkalmazás szempontjából, hiszen többek között itt kerül a webes tartalom és felhasználói felület megjelenítésre, itt történik meg a HTML, CSS és Javascript értelmezése és a kliensoldal végrehajtása, hibakeresésre pedig használhatjuk a beintegrált fejlesztői eszközöket. Népszerűbb böngészők a Chrome vagy a Firefox.

3.2. Előkészületek

A program bármelyik egyszerű böngészőből elérhető (pl.: Google Chrome, Mozilla Firefox, Microsoft Edge), így telepítésre nincs szükség. Bizonyosodjon meg róla, hogy készüléke kapcsolódik-e az internetre, ugyanis enélkül a program nem használható.

3.3. Regisztráció

Az oldal eléréshez alapvetően nem, de ha jegyet szeretne foglalni, akkor felhasználói fiókkal kell rendelkeznie. Enélkül csak a filmkínálatot tudja megtekinteni. A regisztrációhoz kattintson a jobb felső sarokban található "Bejelentkezés" feliratra. Ekkor az oldal átirányítja a bejelentkező felülethez, ahol a "Regisztráció" feliratra kattintva megkezdheti a regisztráció folyamatát. Az oldal betöltése után egy űrlapot kell kitölteni, ami kérni fogja a teljes nevét, az email címét, és a jelszavát két alkalommal, hogy megbizonyosodjon róla, biztosan jól adta-e meg. Az email cím mezőbe valós címet adjon meg, és olyat, amit rendszeresen használ, hiszen ezzel tud majd bejelentkezni, és ide fognak érkezni a foglalásaival kapcsolatos tudnivalók. Ha

minden mezőt kitöltött, kattintson a "Regisztráció" feliratú gombra. Amennyiben a regisztrációs sikeres volt, a képernyőn a "Sikeres művelet" felirat jelenik meg.

3.4. Bejelentkezés

Amennyiben nem rendelkezik felhasználói fiókkal, tekintse meg a Regisztráció részt. Ha már korábban regisztrált, mindössze annyi a dolga, hogy a jobb felső sarokban lévő "Bejelentkezés" feliratra kattint, majd az oldal betöltése után az ön előtt látható belépési űrlapot kitöltse. Az email cím mezőbe írja be az email címét, a jelszó mezőbe pedig a korábban megadott jelszavát, majd nyomja meg a "Bejelentkezés" gombot.

3.5. Promóciók megtekintése

A mozi aktuális promócióinak megtekintéséhez nem szükséges bejelentkeznie.

Az oldal betöltésekor alapértelmezetten a főoldalt látja. Itt találja közvetlen az aktuálisan érvényben lévő promóciókat, kedvezményeket, új filmek reklámját. A promóciók néhány másodperceként maguktól váltakoznak, de ha lépkedni szeretne közöttük, kattintson a kép szélén található nyilak valamelyikére, attól függően, hogy melyik irányba szeretne haladni.

3.6. Filmkínálat megtekintése

Az aktuális filmkínálat megtekintéséhez nem szükséges bejelentkeznie.

Az oldal betöltésekor alapértelmezetten a főoldalt látja, tetején az oldal szekcióival. Kattintson a "Műsoron" menüpontra. Itt láthatja felsorolva a mozi aktuális kínálatát. A bal oldali szűrők használatával tud szűrni a filmek között. Amennyiben valamelyik film felkeltette az érdeklődését és részletesebb tájékoztatást szeretne róla, kattintson a film képére vagy címére. Így elolvashatja a film adatlapját, mely tartalmazza a megjelenés évét, a rendezőt, a film hosszát és egy rövid leírást.

3.7. Jegyárak megtekintése

Az jegyárak megtekintéséhez nem szükséges bejelentkeznie.

Az oldal betöltésekor alapértelmezetten a főoldalt látja, tetején az oldal szekcióival. Kattintson a "Jegyárak" menüpontra. Itt táblázatba foglalva tekintheti meg a mozi jegykategóriáinak megnevezését és árát.

3.8. Jegyfoglalás

Bejelentkezés nélkül is tud böngészni a vetítések között, de a jegyfoglaláshoz be kell jelentkeznie. Amennyiben problémába ütközik, tekintse meg a Bejelentkezés részt.

Belépés után, a főoldalt látja, melynek tetején az oldal szekcióival. Kattintson a "Vetítések" menüpontra. Betöltés után itt láthatja az aznapi vetítéseket. Amennyiben másik időpont iránt érdeklődik, használja a jobb felső sarokban található dátumszűrő-mezőt. Új dátum megadása után megtekintheti a vetítéseket időrendi sorrendben. A könnyebb tájékozódás érdekében figyelje az oldal bal oldalán található idővonalat. A film plakátjára kattintva megtekintheti a film adatlapját, míg a vetítés időpontjára kattintva megkezdheti a foglalást.

A foglalást a vetítés kezdete előtt legalább egy órával tudja megtenni.

Kattintás után a foglalás első lépéseként meg kell adnia, hogy a kétféle jegytípus közül melyikből mennyit szeretne lefoglalni. Mennyiség kiválasztása nélkül az oldal nem engedi tovább a jegyfoglalás következő lépésére.

Ha megadta a kívánt mennyiséget, kattintson a "Tovább" feliratú gombra. Következő lépésként kiválaszthatja ülőhelye pontos helyét, a jobb oldalon található legördülő lista segítségével. A helyválasztást segíti a bal oldali ábra, amely ábrázolja a moziterem befogadóképességét, jelöli a már foglalt székeket (szürke), szabad székeket (zöld) és az ön által aktuálisan kiválasztott (sárga) széket is, beszámozva. Ha kiválasztotta a megfelelő ülőhelyet, kattintson a "Tovább" feliratú gombra. Amennyiben nem választ ki ülőhelyet, az oldal nem engedi tovább a jegyfoglalás következő lépésére.

Az oldal ezután összesítve megjeleníti a választott jegyek számát, fajtáját, ülőhelyének számát. Ellenőrizze az adatokat. Ha mindent rendben talált, kattintson a "Foglalás véglegesítése" feliratú gombra. Sikeres foglalás esetén az oldal megjelenít egy "*Sikeres foglalás! […]*" kezdetű szöveget. A foglalt jegyeit megtekintheti a jobb felső sarokban található profil menüpont alatt, a felhasználó ikonra kattintva.

3.9. Korábbi foglalások megtekintése

Korábbi foglalásainak megtekintéséhez jelentkezzen be az oldalra. Amennyiben problémába ütközik, tekintse meg a Bejelentkezés részt.

Meg tudja tekinteni korábbi jegyfoglalásait. Ehhez kattintson jobb felső sarokban található felhasználó ikonra. Ezután időrendi sorrendben láthatja korábbi foglalásait. Amennyiben még nem volt foglalása, a *'Még nincs foglalás felirat jelenik meg'*.

Lehetősége van foglalása lemondására, legkésőbb az előadást megelőző napig. Amennyiben le szeretne mondani egy foglalást, kattintson a foglalás sorának végén található "Törlés" feliratú gombra. Ekkor az oldal megerősítést kér, elfogadás után pedig megtörténik a lemondás.

4. Források

https://hu.wikipedia.org/wiki/HTML

https://hu.wikipedia.org/wiki/CSS

https://hu.wikipedia.org/wiki/PHP

https://hu.wikipedia.org/wiki/JavaScript

https://code.visualstudio.com/docs