

Záródolgozat

Lukácsi Dániel



OKJ 54 213 05

— Szoftverfejlesztő —

5/13/A

Kecskeméti Szakképzési Centrum

Kandó Kálmán Technikum

Kecskemét

2021

FIGYELEM!

A program a fejlesztő tulajdona, amit a nemzetközi szerzői jogok védenek.
A program használati engedélyét kizárólagosan a verseny idejére biztosítom.
Továbbadása, forráskódjának visszafejtése a szerző engedélye nélkül tilos.

Kandó mozi

Tartalomjegyzék

A program általános specifikációja	4
Rendszerkövetelmények	5
Megvalósítás	8
Index.php.....	9
Fooldal.php	10
Foglal.php	12
Az eddigi oldalak kinézetei:.....	14
Filmajanlo.php	16
Musor.php	17
Bejelentkezo.php.....	23
Vizsgalo.php.....	24
Admin.php.....	25
Film, Előadás felvétele.....	27
A felhasználói dokumentáció	28
Az alkalmazott fejlesztői eszközök.....	28
Adatmodell leírása	29
A Tábla kapcsolatok.....	32
A filmek.fId=az eloadasok.FId-vel	32
A PHP adatbázis kapcsolódás	33
Tesztelés	34

A program általános specifikációja

A weboldal célja, hogy a diákok szabadidejükben megtekinthessék kedvenc filmjeiket az iskolában. Az oldal lehetővé teszi, hogy jegyet foglaljanak a filmekre. Megnézhetik a filmek előzetesét, rövid tartalmáról olvashatnak és minden információt megtekinthetnek ami vetítendő filmmel/filmekkel kapcsolatos. Lehetőség van visszajelzést is küldeni az oldalól, ha esetleg valami probléma merülne fel vagy akár élménybeszámolót is írhatnak róla.

A weboldal **HTML, CSS, Bootstrap, PHP és Javascript**-el készült. Egy menü vezérelt weboldallról van szó, amely 3-4 weblapból áll össze.

A foglalásokat, illetve a visszajelzéseket egy Admin felületen lehet megtekinteni, módosítani, törölni. Az Admin felületen lehetőségünk is van arra, hogy filmet/filmeket adjunk hozzá a megtekinthető filmek listájához. Persze minden adatával együtt. Az Admin felület kezelése is weboldalon történik. Egy külön weboldalon.

A weboldalt **Visual Studio Code**-ban készítettem és az adatbázis kezelését **XAMPP**-on végeztem el.

A **Visual Studio Code** a Microsoft több programozási nyelvet tartalmazó fejlesztőkörnyezete, amely az évek során egyre több új programnyelvvvel bővült.

A **XAMPP** szabad és nyílt forrású platformfüggetlen webservert-szoftvercsomag, amelynek legfőbb alkotóelemei az Apache webservert, a MariaDB (korábban a MySQL) adatbázis-kezelő, valamint a PHP és a Perl programozási nyelvek értelmezői (végrehajtó rendszerei). Ez a szoftvercsomag egy integrált rendszert alkot, amely webes alkalmazások készítését, tesztelését és futtatását célozza, és ehhez egy csomagban minden szükséges összetevőt tartalmaz. A rendszer egyik nagy előnye az összehangolt elemek könnyű telepíthetősége.

Rendszerkövetelmények

Hardveresen

A weboldal megtekinthető az interneten egy URL segítségével, szóval mondhatni bármely számítógépen megtekinthető vagy akár laptopon, mobilon is. A Nethely.hu lehetővé teszi, hogy a weboldal elérhető legyen.

De, ha módosítani szeretnénk rajta vagy akkor egy alap számítógépre szükségünk lesz.

Pl. amire szükségünk van:

- MSI A320M Pro-E alaplap: két DDR4-es memóriamodul kezelésére képes (vagyis a RAM akár 32 GB-ig is bővíthető), 12 USB-port, RJ45-csatlakozó, valamint DVI-D és VGA-port.
- AMD Athlon 200GE processzor
- Patriot PSD44G240041 4 GB RAM
- Kingston A400-as típusú, 240 GB tárhelyes SSD

Processzor	AMD Athlon 200GE
Processzorhűtő	Gyári
Alaplap	MSI A320M PRO-E
Memória	Patriot 4GB PSD44G240041
Grafikus kártya	CPU-ba integrált
SSD	Kingston A400 240 GB
Merevlemez	—
DVD/Blu-ray egység	—
PC-ház	Spire Maneo 1073/420 W
Tápegység	A házban

Szoftveresen

A fentiekben említett programokról lenne szó. Igazából bármelyik operációs rendszereken lefutnak. Tegyük fel a Windows 7, 8.1, 10....Linux és még sorolhatnám. Ez egyénfüggő, hogy ki melyik operációs rendszert szeretné használni.

A letöltendő programok pedig:

- XAMPP
- Visual Studio Code



A XAMPP letöltése, telepítése:

- Bármely verzióját meg is találhatjuk az interneten (én a legújabbat választottam).
- Telepítése során az első ablakon kilehet választani, hogy mely szervereket, programnyelveket telepítsük. Itt nyugodtan kiválaszthatjuk mindet.
- A következő lépésben a program telepítési helyét kell megadnunk. Innen már csak a felhasználói feltételek elolvasása, kipipálása után már telepíthetjük is a programot
- Az Apache és a MySQL indítása után az Adminra nyomjunk rá és már is elérhetővé válik számunkra az Adatbázisokkal való műveletek.



A Visual Studio Code letöltése, telepítése:

- Bármely verzióját meg is találhatjuk az interneten (én a legújabbat választottam).
- Szinte ennél is ugyanaz a séma, mint az előzőnél. A program telepítési helye, felhasználói feltételek, program nyelvének kiválasztása.
- Telepítés után szükséges nekünk pár bővítményt letöltenünk, ami később még a hasznunkra válhat. Itt gondolok az (Open in Browser, Open PHP).
- Settings.json file-ban átítni, beírni az elérési útvonalat a xampp-on történő futtatás miatt.
- Ez azt teszi lehetővé, hogy a programunk csak akkor fut a szerver is aktív.

```
"explorer.confirmDelete": false,  
"php.executablePath": "C:\\xampp\\php",  
"files.autoSave": "afterDelay",  
"open-php-html-js-in-browser.customUrlToOpen": "http://localhost/${%7BrelativeDirnameDocumentRoot%7D}/",  
"open-php-html-js-in-browser.documentRootFolder": "C:\\xampp\\htdocs\\",  
"open-php-html-js-in-browser.selectedBrowser": "Chrome",
```

Open-PHP-HTML-JS-in-browser: Document Root Folder

Base directory of your pages to serve from `http://localhost` domain (eg by default

C:\\xampp\\htdocs\\

Open-PHP-HTML-JS-in-browser: Remember Browser Selection

☒ Remember last browser selection. Uncheck and select Ask always... launch

Open-PHP-HTML-JS-in-browser: Selected Browser

Browser to open (Chrome, Firefox, ...)

Chrome

Megvalósítás

A weboldal **php menüvezérelt**. A PHP-ban készült weblapok paraméterezhetőek. Ennek megfelelően megoldható, hogy míg egy weblap kezdőoldala az `index.php` legyen, addig a további oldalak ugyanezen **index.php** oldal paraméterezésének eredményeként legyenek elérhetőek.

Ilyenkor a háttérben (vagyis a PHP programban) az történik, hogy az `index.php` kezdőoldal mindig betölti a weblap állandó elemeit (logó, fejléc, lábléc, menü, stb...), ám az egyes menüpontokhoz tartozó tényleges tartalom a paraméter értékétől függően kerül kiválasztásra és megjelenítésre. Erre azért van lehetőség, mert a PHP egy programozási nyelv (tehát lehetnek benne változók, és elágazások), míg a HTML csak jelölőnyelv (tehát nincsenek benne sem változók, sem elágazások).

Azért, hogy az `index.php`-ban szereplő forráskód ne legyen irreálisan hosszú, érdemes az egyes menüpontokhoz tartozó tartalmakat önálló fájlban megalkotni. Ahhoz pedig, hogy a paraméter értékétől függő tartalmat önálló forrásfájlból is beszerkeszthessük a főoldalt jelentő `index.php`-ba, a **PHP nyelv `include()` függvényét használhatjuk**:

```
<ul class="navbar-nav ml-auto">
  <li><a class="nav-link" href="index.php">Főoldal</a></li>
  <li><a class="nav-link" href="index.php?p=filmajanlo.php">Filmek</a></li>
  <li><a class="nav-link" href="index.php?p=musor.php">Műsor</a></li>
  <li><a class="nav-link" href="index.php?p=kapcsolat.php">Kapcsolat</a></li>
</ul>
<?php
  if(isset($_GET["p"])){
    include $_GET["p"];

  else include("fooldal.php");
?>
```

Ezáltal a többi weboldalhoz nem szükséges semmilyen HTML fejléc vagy hasonló, csak az oldal tartalmával kell foglalkoznunk. Ez a módszer megkönnyíti a későbbiekben azt is, ha módosítani akarunk a weboldalon.

Index.php

Ezen az oldalon helyezkednek el a weboldal „alapjai”. A html kód head részében helyezkednek el az oldal formázásához szükséges hivatkozások. Gondolok itt a .css fájlokra vagy a bootrapes fájlokra is. Továbbá, ha scriptek is szükségesek azokat is a head részben szoktam elhelyezni.

```
<link rel="stylesheet" href="style/styles.css">
```

A navigációs menü formázását nagyrészt bootstrap-el valósítottam meg. navbar, navbar-expand-lg, navbar-collapse, navbar-nav, ml-auto. A tageken belüli linkeket nav-link class osztálykijelölővel láttam el. Az ilyen háttérszíneket általában a html-ben már beállítom mert így nekem jobban átláthatóak. Minden mást css-ben végzek el, mint pl: A linkek színét, kurzor reakciókra való átszínezés stb. stb...

```
<script>
window.addEventListener('load',Menu)

function Menu(){
    let array=document.links;
    let userUrl=document.URL;
    for (let index = 0; index < array.length; index++) {
        if(array[index].href == userUrl){
            document.links[index].style.color = "red";
        }
    }
}
</script>
```

A script kód használatával annak a weboldalnak a link-je amin épp rajta vagyunk beállít neki egy színt (Ez esetben a piros). Így megtudjuk különböztetni, hogy melyik oldalon is vagyunk éppen.

A footert bg-dark osztálykijelölővel láttam el, ami a háttérszínét sötétszürkére álltja. Azon belül a szövegek elrendezéséhez szintén bootstrapet használtam.

Pl.:

```
<div class="col-lg-8 col-md-8 col-sm-12">
<div class="col-lg-4 col-md-4 col-sm-12">
```

Ezek segítenek a szövegek elhelyezését ,arányosan ossza el a weboldalon.

A footer-ben lévő linkekre a target=”_BLANK”-ot állítottam be, hogy új oldalon nyissa meg a linket.

Fooldal.php

A php tagben a config.php-ra van egy hivatkozás, hogy az oldalhoz készített adatbázissal lehessen majd dolgozni: require_once "config.php"; Egy változó értékének megvan adva az sql lekérdezés. Létre van hozva egy üres változó, amivel később az eredményt iratjuk ki. Az extract() lehetővé teszi azt, hogy az adatbázisban szereplő mezőnevekre változóként hivatkozzunk.

```
$adatok.="
    <div class='movie-box'>
        <img src='img/{ $fImg}' alt=''>
        <div class='movie-info '>
            <h3>{ $fCim}</h3>
            <a href=foglal.php?id={ $fId}>Foglalás</a>
        </div>
    </div>";}
```

Az oldalon létrehoztam egy fejléct, ami egy képből áll és a kép közepén az oldalról egy rövid bemutatkozó szöveget írtam. A h-100 osztálykijelölő a szélességet a lap szerint igazítja, vagyis 100%-osra.

```
<header id="fejlec" style="background-image:url(img/mozihatter.jpg);">
    <div class="container-fluid h-100">
        <div class="row h-100 align-items-center">
            <div class="col-12 text-center text-light" style="-webkit-text-
stroke: 1px black;">
                <h1>Kandó Mozi</h1>
                <h2 >Üdvözöllek a weboldalon, ahol sulí után vagy akár a lyukasórák között
megnézhetsz egy filmet a kínálatunkból a barátaiddal!!!</h2>
                <h2>Foglaláshoz görgess lejjebb!!</h2>
            </div>
        </div>
    </div>
</header>
```

fejléc CSS formázása: Kép ne ismétlődjön.

```
#fejlec{
    height: 100vh;
    min-height: 400px;
    background-repeat: no-repeat;
    background-size: cover;
}
```

A html tagben az **adatok** változóra hivatkozva megkapjuk eredménynek az oldalon, ami benne van. A movie-box, movie-info, movie-show-container osztálykijelölőkkel formáztam az oldalt css-ben.

- **.movie-box** { position: relative; margin: 10px 0; }
- **.movie-box .movie-info** { padding: 50% 0; position: absolute; top: 50%; left: 50%; transform: translate(-50%, -50%); -ms-transform: translate(-50%, -50%); text-align: center; height: 100%; width: 100%; opacity: 0; transition: .7s ease; background-color: black; }
- **.movie-show-container** { margin-top: 0; padding: 0 10%; }
- **.movie-show-container>h3** { color: #969696; text-align: left; padding: 0 10px 0; }
- **.movie-show-container>h3:after** { content: "; display: block; height: 3px; width: 7%; background: black; position: relative; bottom: -10px; }
- **.movies-container** { display: grid; grid-column-gap: 10px; grid-template-columns: auto auto auto auto auto auto; padding: 10px 0; }

```
<div class="movie-show-container">
  <h3>Foglalj most!</h3>
  <div class="movies-container">
    <?=$adatok?>
  </div>
</div>
```

A **movies-container**-en belül írtam ki az adatokat. A filmek borítóképét és nevét. Itt lehetőség is van a film foglalásának az oldalra átmenni. A kurzort, ha a kép fölé visszük feldob egy „foglalás” linket, ami oda átirányít.

Ennek a css kódja:

- **.movie-box:hover .movie-info** { opacity: 0.8; }
- **.movie-info>a** { display: inline-block; padding: 0.35em 1.2em; border: 0.1em solid white; border-radius: 0.13em; box-sizing: border-box }
- **.movie-info>a** { ; text-decoration: none; font-weight: 300; color: white; text-align: center; transition: all 0.2s; }
- **.movie-info>a:hover** { color: black; background-color: white; }

Foglal.php

Ezen az oldalon három táblára volt szükség az adatbázisból. A **filmek** adattáblából kiírtam az megjelenítendő oldal adatait egy táblázatban. Az **eloadások** adattáblára a select-ekben volt szükség. És a **foglalások adattábla**. Ebbe menti el a foglaló személy adatait meg, hogy melyik filmre melyik időpontban foglalt.

```
$id = $_GET['id'];
$sql = "SELECT * FROM filmek WHERE fId = $id";
$stmt = $db->query($sql);
$row = $stmt->fetch();
```

A foglalt cím id-jét lekérve egyenlővé tesszük az sql parancsban, így a megfelelő film adatait listázza ki.

A html részében csak elég lesz így hivatkoznunk rá! `<?php echo $row['adat']; ?>`

A selectekhez az eddig használt módszerre volt szükség.

```
$sql="select * from eloadasok where fId=$id group by EDatum";
$stmt = $db->query($sql);
$strDatumok="";
while($sor = $stmt->fetch()){
    extract($sor);
    $strDatumok.="<option value='$EDatum'>$EDatum</option>";
}
```

A Dátumot és Időt sql parancs segítségével írtam ki és az optionbe átadtam ezt neki értéknek, így foglaláskor ezt az értéket menti el.

Az oldalon egy foglal nevű gombra hivatkozva mentettem ki az adatokat az adatbázisba, aminek a programkódja így néz ki:

```
if(isset($_POST['foglal'])){
    extract($_POST);
    $sql="INSERT INTO
    foglalas (fogCim,fogDate,fogTime,
    fogVnev,fogKnev,fogEmail,fogTelefon)
    VALUES ('".$_row['fCim']."'.'".$_POST["nap"]."',
    '".$_POST["ora"]."', '".$_POST["VezNev"]."',
    '".$_POST["KerNev"]."', '".$_POST["Email"]."',
    '".$_POST["tSzam"]."')";

    $stmt=$db->exec($sql);
}
```

A form method="post"-ban lévő inputok nevére hivatkozva kimentti az értékeket.

Az oldal tetején láthatjuk a film címet és a jobb felső sarokban egy ikont, ami egy kattintással visszavisz minket az előző oldalra egy scripttel. Az elrendezést bootstrap-el csináltam a méretarányos felbontásos elrendezéssel, mint korábban. Hogy az ikon látható legyen, a head-en belül hivatkoztam egy linkre. Az ikon a **Font Awesome Introduction**-ból van. Egy i tag class-ban megadtam ezt az értéket: „fa fa-times”. 24px-es méretet és darkred színt beállítva. A táblázat páros és páratlan sorainak színei eltérnek a **table**, **table-striped**, **table-dark** osztálykijelölők alkalmazásával.

További css beállítások a selectekre, inputokra és táblázatra:

(színek,méretetek,keretek, margók)

```
.table-container{
    margin-top:25px;
    margin-left: 10px;
    margin-right: 10px;
}

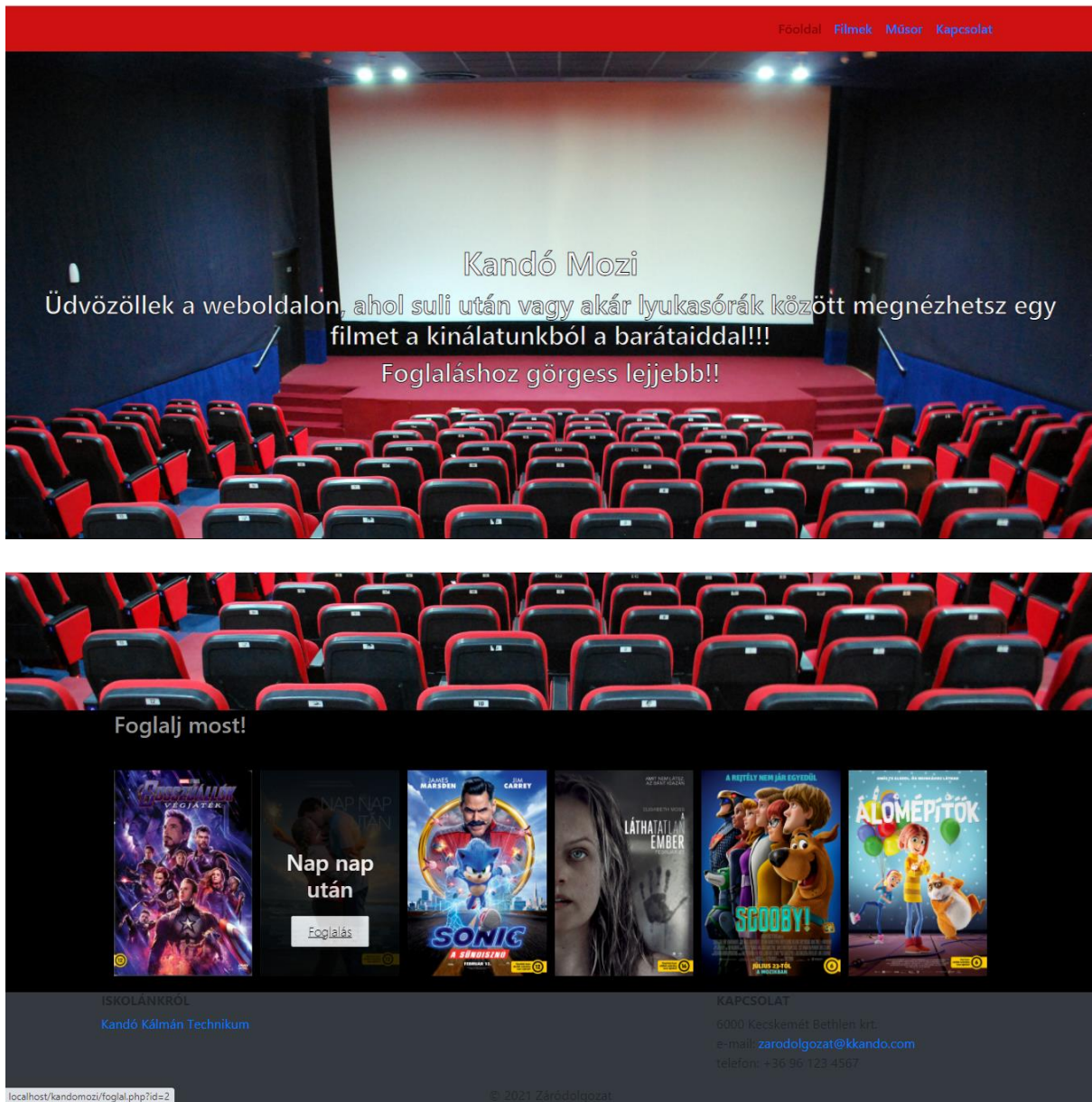
.foglalas-container{
    margin-top:25px;
    margin-left: 40px;
    margin-right: 40px;
}
select{
    padding: 6px 12px;
    background-color:grey;
    color: darkred;
    border: 2px solid black;
}

input[type=text] {
    background-color:grey;
    font-size:18px;
    margin:5px;
}

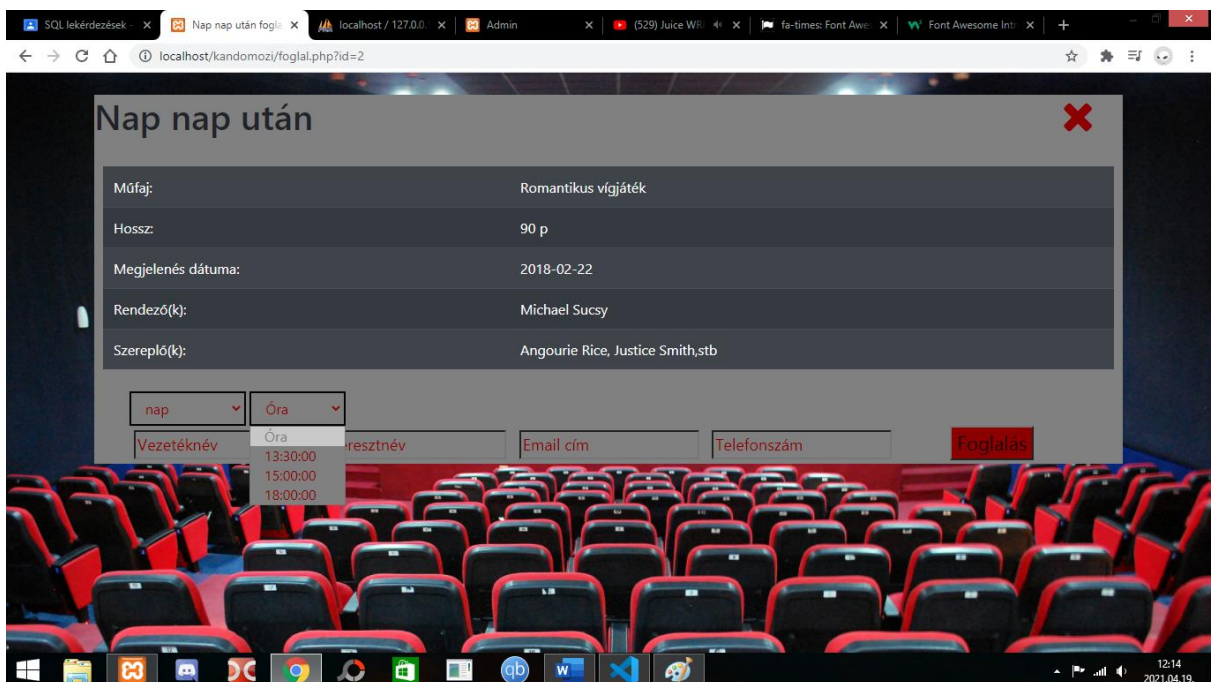
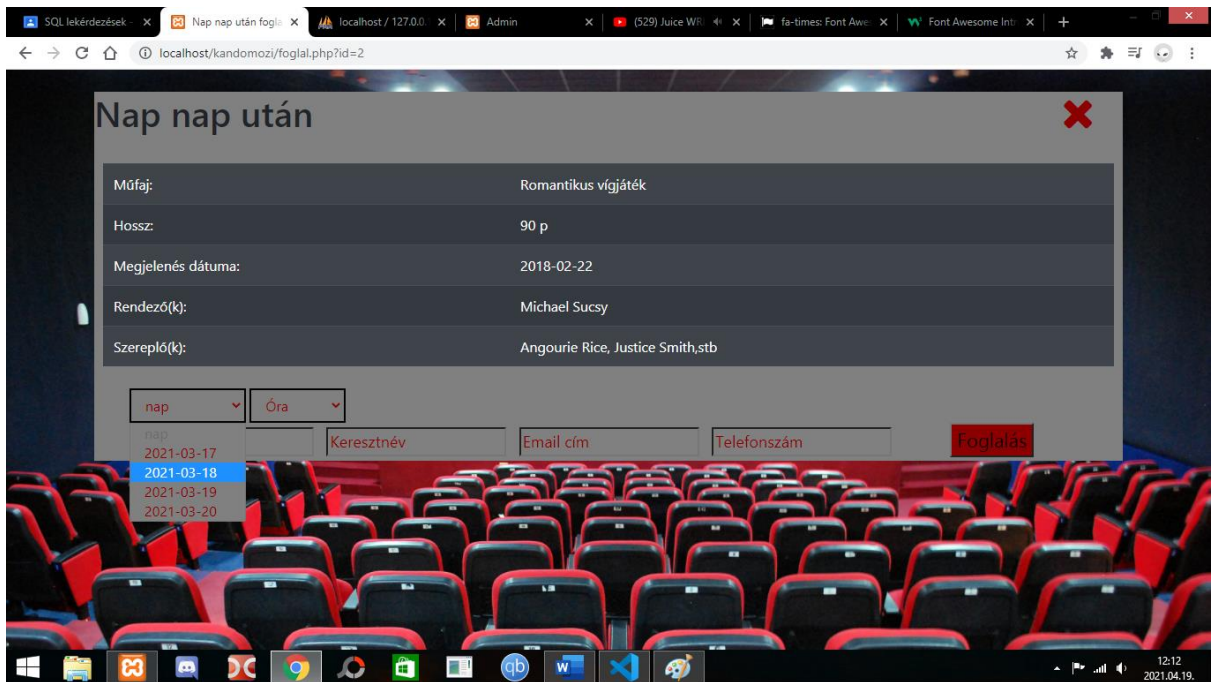
textarea[type=text] {
    background-color:grey;
}
input[type=button] {
    background-color: darkred;
    font-size:22px;
    cursor: pointer;
}
```

Az eddigi oldalak kinézetei:

fooldal.php



foglal.php



Filmajanlo.php

Az oldalon a filmek adattáblára volt szükség. Egy sql lekérdezéssel, majd az eddigi módszerrel valósítottam meg az adatok kiíratását.

```
$sql="select * filmek;";
$adatok="";
$stmt=$db->query($sql);
while($sor=$stmt->fetch()){
    extract($sor);
    $adatok.="
```

Az oldalon a filmek rövid tartalmáról olvashatunk és beágyazott linkeket láthatunk, ezek a filmeknek az előzeteseik.

- **sql-ben még:**

```
<div class='filmekajanlo'>
    <div class='col-md-12 col-12'>
        <div class='card bg-secondary text-light'>
            <div class='card-body'>
                <h1 class='card-title'>{$fCim}</h4>
                <p class='card-text text-justify'>
                    {$fTartalom}
                </p>
                <iframe class='img-fluid w-
100' src='{$fLink}' frameborder='0' allow='accelerometer; autoplay; clipboard-
write; encrypted-media; gyroscope; picture-in-
picture' allowfullscreen></iframe>
            </div>
        </div>
    </div></div>
";}
```

Az adatokat egy **section**-ben írtam ki. Az oldalon minden formázást bootstrap-el oldottam meg(közepre igazítások,méretezés,elrendezés).

- **A filmekajánló-n elvégzett css formázások(padding-ek):**

```
.filmekajanlo{
    padding-top: 10px;
    padding-bottom:10px;
    padding-left: 120px;
    padding-right:120px ;
}
.img-fluid{
    max-height: auto;
}
```


Musor.php

Ezen az oldalon a filmek és az előadások adattáblára volt szükség. Egy select segítségével kiválasszuk, hogy melyik napon szeretnénk megnézni a vetítéseket. Miután kiválasztottuk, egy táblázatba kiírja nekünk a film címét és az időpontokat.

PHP parancs a select-ig:

```
$sql="select EDatum from eloadasok group by EDatum;";
$stmt=$db->query($sql);
$d_eredmeny="";
$musorok="";
$aznap="";

while($sor=$stmt->fetch()){
    extract($sor);
    $d_eredmeny.="<option value='{ $EDatum }'>{ $EDatum }</option>";
}
```

A submit gombra kattintva el is végzi nekünk a parancs, hogy kiírja a napot h1-es betűmérettel középre igazítva és az adatokat egy táblázatba.

```
if(isset($_POST['btn'])){
    $EDatum=$_POST['datum'];
    $sql="SELECT fCim,Eido1,Eido2,Eido3,Eido4 from filmek inner join eloadasok
on filmek.fId=eloadasok.FId where EDatum='{ $EDatum }' order by fCim;";
    $stmt=$db->query($sql);

    while($sor=$stmt->fetch()){
        extract($sor);
        $aznap="{ $EDatum } napi vetítéseink";
        $musorok.="
        <tr><td>{ $fCim }</td><td>{ $Eido1 }</td>
        <td>{ $Eido2 }</td><td>{ $Eido3 }</td>
        <td>{ $Eido4 }</td></tr>";
    }
}
```

A táblázatot ezen az oldalon is „table table-striped table-dark” osztálykijelölővel láttam el. A select-et, submit gombot, h1-et és a táblázatot is css-ben formáztam.

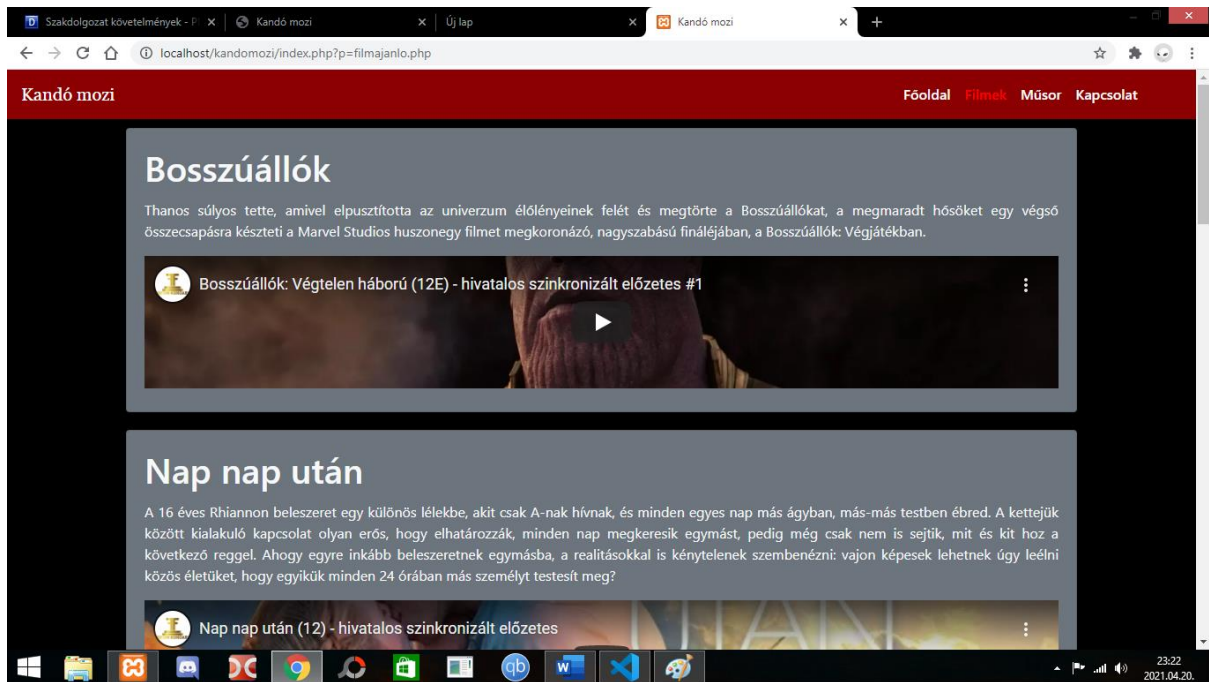
Az oldalon elhelyeztem az index.php oldalon már elhelyezett footer-t is és beállítottam neki a fixed-botton osztálykijelölőt, hogy maradjon mindig fixen az oldal legalján.

A musor.php-n elvégzett css formázások:

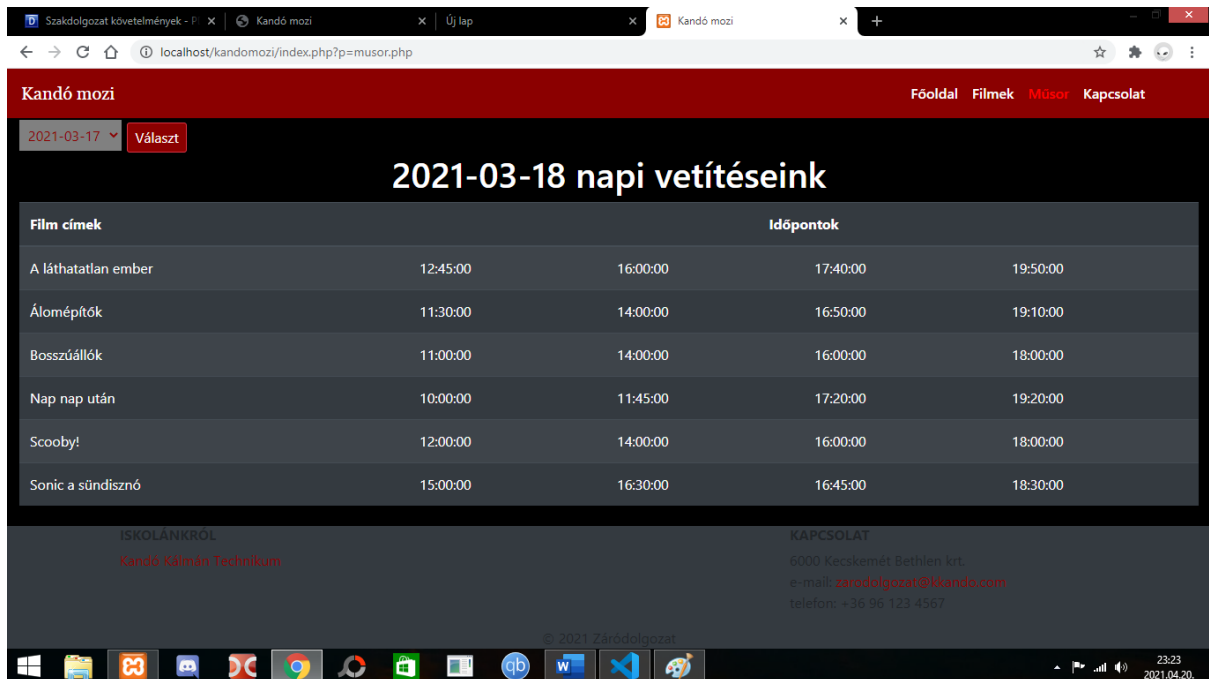
```
.aznap{
    text-align: center;
    color:white;
}
#datum{
    margin-left:12px !important;
    background-color:grey !important;
    color: darkred !important;
    border: 2px solid black !important;
    padding: 6px;
}
input[type=submit] {
    background-color: darkred;
    font-size:16px;
    cursor: pointer;
}
.musortabla{
    border-collapse: collapse;
    width: 98% !important ;
}
.center {
    margin-left: auto;
    margin-right: auto;
}
```

- **h1:** Középre igazítás + fehér betűszín
- **datum id-vel ellátott select:** Szürke háttér, balról a margó 12px-el bentebb, betűszín bordó, 2px-es fekete keret
- **submit gomb:** Bordó betűszín, szöveg méret 16px, cursor-t, ha rávisszük megváltozik(rámutat)
- **táblázat:** 98%-os szélesség, szegély beállítás
- **.center:** Középre igazítás.

Filmajanlo.php oldal kinézete



Musor.php oldal kinézete



Kapcsolat.php

A visszajelzések adattáblára volt szükség ehhez az oldalhoz. Az inputokban és a textareaban lévő begépelte üzenetet elküldheted. Ez amolyan üzenő felület bármilyen problémával vagy szimplán a pozitív visszajelzésekhez az oldalról.

Php kód:

```
if(isset($_POST['beir'])){
    extract($_POST);
    $sql="insert into visszajelzes values (null,'{$Vnev}',
    '{$Knev}','{$Email}','{$Szov}')";
    $stmt=$db->exec($sql);
}
```

A html form="post"-ban lévő kitöltési mezők értékét a submit gomb lenyomásával továbbítja/eltárolja. Az inputoknak ugyanazt a nevet kell adni, mint az adatbázisban lévő adatok mezőinek.

Ez az oldal bal oldalán helyezkedik el (css-ben majd kitérek ehhez).

```
<div class="contact-us-container">
    <div class="contact-us-section contact-us-section1">
        <h1>Kapcsolat</h1>
        <p>Adatok</p>
        <form method="POST">
            <input type="text" placeholder="Vezetéknév" name="Vnev"><br>
            <input type="text" placeholder="Keresztnév" name="Knev" ><br>
            <input type="text" placeholder="E-mail cím" name="Email"><br>
            <textarea type="text" placeholder="Üzenet írása..."
            name="Szov" rows="10" cols="30"></textarea><br>
            <button type="submit" name="beir"
            value="submit">Küldés</button>
        </form>
    </div>
```

A div-eket, inputokat,textarea-t, submit gombot (mindent a css-ben formáztam).

```
<div class="contact-us-section contact-us-section2">
    <h1>Rólunk</h1>
    <h3>Telefon</h3>
    <p><a href="">+36 20 216 2568</a></p>
    <h3>Cím</h3>
    <p>6000 Kecskemét, Bethlen krt. 63</p>
    <h3>E-mail</h3>
    <p><a href="">zarodolgozat@kkando.com</a></p>
</div>
</div>
```

Az oldal jobb oldalán az elérhetőségek láthatóak, hasonlóan a footer-hez. Ezeknek a formázása is mind css-ben történt. Az oldal legalján egy beágyazott google térkép található, amin látható az iskola címe.

```
<div class="gmap_canvas">
<iframe id="gmap_canvas" src="https://www.google.com/maps/embed?pb=!1m14!1m8!1m3!1d12963.60370814023!2d19.689910372545842!3d46.91723122728676!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x0%3A0x76e03484e50b2f33!2zS2Vjc2t1bc0pdGkgU1pDIEthbmTDsyBLw6FsbC0hbiBUZWNoYm1rdW0!5e0!3m2!1shu!2shu!4v1608203223214!5m2!1shu!2shu"
frameborder="0" scrolling="no" marginheight="0" marginwidth="0">
</iframe>
</div>
```

A css formázások: Szövegszínek, margók méretezése, szélesség, magasság beállítás százalékosan, pixelesen.

```
.contact-us-container {
    display: grid;
    grid-gap: 25px;
    grid-template-areas: 'contact-form contact-form contact-info';
    margin: 30px 15%;
}

.contact-us-section {
    overflow: hidden;
}

::placeholder {
    color: darkred;
}

.contact-us-section h1 {
    color: white;
    padding: 20px 0;
    margin-bottom: 20px;
}

.contact-us-section p {
    color: white;
}

.contact-us-section1 form * {
    padding: 10px;
    margin: 10px 0;
}

.contact-us-section1 form button {
    background-color: darkred;
    color: white;
    border: none;
    cursor: pointer;
}
```

```

.contact-us-section2 {
  grid-area: contact-info;
}

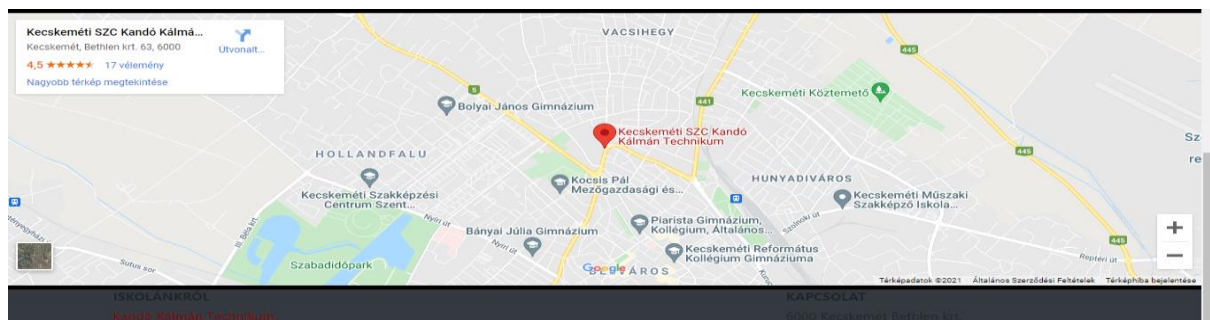
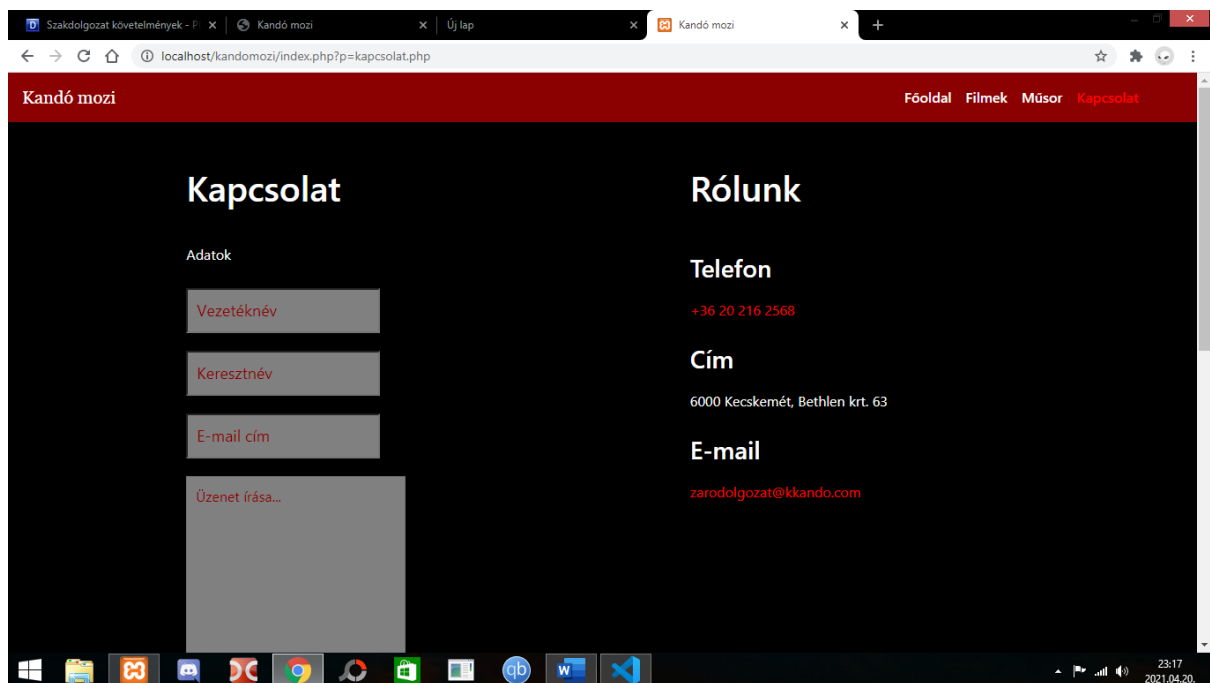
.contact-us-section2 h3 {
  color: white;
  padding: 10px 0;
  margin-bottom: 10px;
}

.contact-us-section2 p{
  color:white;
}

#gmap_canvas {
  width: 100%;
  height: 400px;
}

```

A kapcsolat.php oldal kinézete:



Bejelentkezo.php

Először is...Ahhoz, hogy az **admin.php**-ra kerüljünk a navigációs menü bal oldalán lévő Kandó mozi linkre kell kattintanunk ami átirányít minket egy bejelentkező oldalra(**bejelentkezo.php**). Itt, ha megadjuk a helyes adatokat, beléphetünk az admin felületre.

```
<?php require_once "vizsgalo.php"; ?>
```

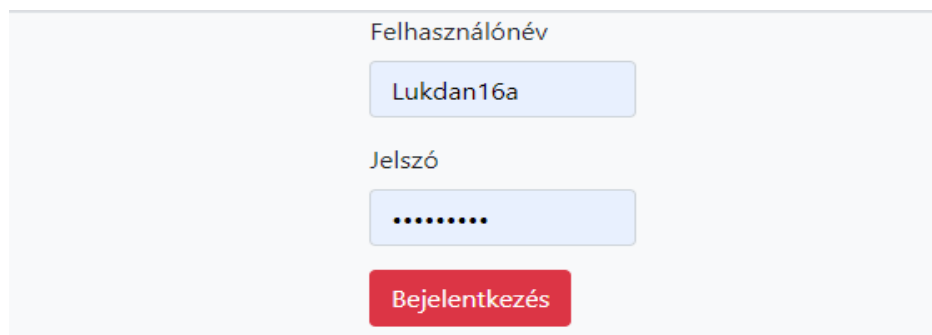
Az elején hivatkozunk egy vizsgalo.php-ra, ez ellenőrzi majd az adatainkat.

```
<div class="container border bg-light">
<div class="row justify-content-center">
  <div class="col-2">
    <form action="" method="POST">
      <div class="form-group">
        <label>Felhasználónév</label>
        <input class="form-control" type="text"
          name="username" value="" />
      </div>

      <div class="form-group">
        <label>Jelszó</label>
        <input class="form-control" type="password"
          name="password" value="" />
      </div>

      <div class="form-group">
        <input type="submit"
          name="submit" class="btn btn-danger" value="Bejelentkezés" />
      </div>
    </form>
  </div>
</div>
```

Az oldalon csak bootstrap formázásokat végeztem form-control,col-2,bg-light,justify-content-center. Ezek a középre igazítást,középre rendezést és háttérszínt állítják be.



Vizsgalo.php

```
function try_to_login(){
    // Behozzuk a másik fájlban létrehozott változót,
    // hogy tudjuk megváltoztatni az értékét
    global $is_logged_in;
    // Ha bejelentkezhethetünk
    if( can_try_login() == FALSE )
    {return;}
    // Ha a felhasználó név vagy a jelszó nem talál , hiba üzenet mutatása
    if( check_login() == FALSE )
    {echo "<div class='row justify-content-
center'><h1>Helytelen adatok</h1></div>";
        return;}
    // Ha minden tökéletes, akkor bejelentkezés, és üzenet mutatása
    echo "<div class='row justify-content-
center'><h1>Sikeres bejelentkezés</h1></div>";
    $is_logged_in = TRUE;}
function can_try_login(){
    // Ha üres a $_POST akkor nincs mit csinálni
    if( empty( $_POST ) )
    {return FALSE;}
    // Ha nem érkezett username akkor nincs mit csinálni
    if( ! isset( $_POST["username"] ) )
    {return FALSE;}
    // Ha nem érkezett jelszó akkor nincs mit csinálni
    if( ! isset( $_POST["password"] ) )
    {return FALSE;}
    // Ha minden OK
    return TRUE;
}
function check_login(){
    $username = "Lukdan16a";
    $password = "Lukdan16a";
    // Ha nem talál a felhasználó név, hiba
    if( $_POST["username"] != $username )
    {return FALSE;}
    // Ha nem talál a jelszó, hiba
    if( $_POST["password"] != $password )
    {return FALSE;}
    // Minden ok, bejelentkezhethet
    return TRUE;
}
```

Ha az bekért adatok megegyeznek megjelenik egy link, amire ha rákattintunk már az admin felületen is vagyunk.

Admin.php

Itt az összes táblára szükség volt, hiszen itt kezelünk mindent. Filmek felvétele, kiírása, törlése. Foglalások kiírása, törlése. Vetítések hozzáadása, kiírása, törlése és az üzenetek kiírása, törlése. A navigációs menü itt úgy működik, hogyha rákattintunk pl. a filmek felvétele fülre, akkor megkeresi a filmfelvétel id-vel ellátott div-et és oda irányít és ugyanígy a többi füllel is...

Az oldal elrendezése sablonosnak tűnhet, mert az is. A foglalások, üzenetek, filmek, vetítések ugyanazon a módon működnek, csak más sql parancsokkal és változókkal.

Minden egy **container border p-12 class**-ba van rakva és minden táblázat **table table-hover table-fixed-border** és **table-responsive scrollable**-be van rakva. Ezek keretezik a diveket táblázatban fix keretet állítanak be és a táblázat responsive lesz.

Az egyetlen css formázások az oldalon:

```
.scrollable{ height:400px; overflow:scroll; }
a{ color:white; }
a:hover{ color:red; }
```

Görgethető táblázat + fix 400px magasság, link fehér színe, cursorral meg piros.

```
$adat=$db->query("select * from foglalas;");
while($sor=$adat->fetch()){
    extract($sor);
    $foglalasok.="<tr><td><a class='btn btn-
danger' href='deleteFoglalas.php?id={$fogId}'>Törlés</a></td>
    <td>{$fogCim}</td><td>{$fogDate}</td><td>{$fogTime}</td><td>{$fogVnev}</td>
    <td>{$fogKnev}</td><td>{$fogEmail}</td><td>{$fogTelefon}</td></tr>";
}
```

Amint említettem mind a négy adat kiírás megegyező szintaxissal rendelkezik csak más sql lekérdezéssel és mezőnevekkel. A táblázatban minden adatsor mellé bal szélre létrehoz egy piros törlés gombot amelynek az adatsor id-jét hozzárendeljük. Mind a négy kiírásnak egy külön **deleteValami.php**-ja van.

- deleteEloadas.php
- deleteFilm.php
- deleteFoglalas.php
- deleteUzenet.php

Az összes delete---.php szintaxisa is megegyezik.

Egy két példa: Előadások és film törlések.

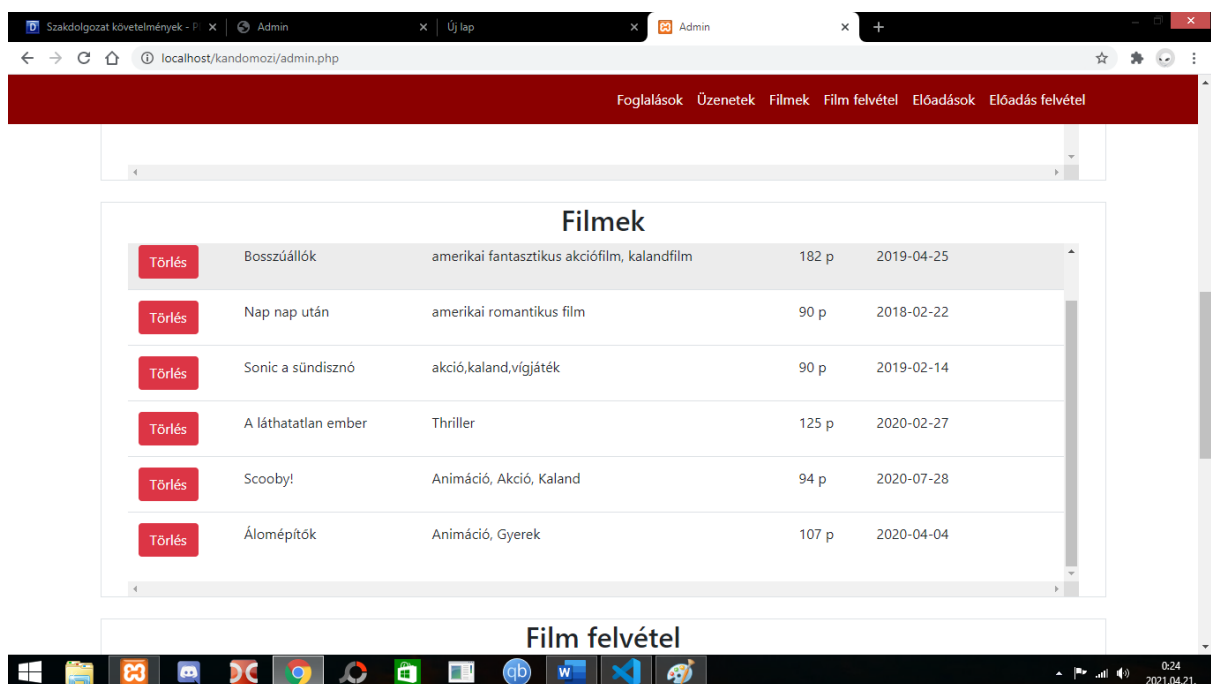
```
<?php
require_once "config.php";
print_r($_GET);
$id=$_GET['id'];
$sql="delete from eloadasok where EId={$id}";
$db->exec($sql);

header("Location:admin.php");
?>
```

```
<?php
require_once "config.php";
print_r($_GET);
$id=$_GET['id'];
$sql="delete from filmek where fId={$id}";
$db->exec($sql);

header("Location:admin.php");
?>
```

A törlésgombhoz hozzárendelt id-t egyenlővé tesszük egy id változóval majd az sql lekérdezésben kitöröljük azt az id-t ahol ez az id található. ha ez megtörténik, visszairányít minket az admin.php-ra. Persze, csak egy oldal frissítést veszünk belőle észre.



Film, Előadás felvétele

```
//film hozzaadas sql//
    if(isset($_POST['filmfelvetel'])) {
        extract($_POST);
        $sql="insert into filmek values (null,'{$fCim}',
        '{$fImg}','{$fMufaj}','{$fHossz}','{$fDatum}','{$fRendezok}',
        '{$fSzereplok}','{$fTartalom}','{$fLink}')";
        $stmt=$db->exec($sql);

        if($stmt)
            $msg1="Sikeres adatbeírás.";
        else $msg1="Nem sikerült";}

//Eloadas felvetele//
$sql="select * from eloadasok inner join filmek
on filmek.fId=eloadasok.FId group by fCim;";
$stmt = $db->query($sql);
$filmeklista="";
while($sor = $stmt->fetch()){
    extract($sor);
    $filmeklista.="<option value='{$fId}'>{$fCim}</option>"; }

    if(isset($_POST['eloadasfelvetel'])) {
        extract($_POST);
        $sql="insert into eloadasok values (null,'{$fId}','{$EDatum}',
        '{$Eido1}','{$Eido2}','{$Eido3}','{$Eido4}')";
        $stmt=$db->exec($sql);

        if($stmt)
            $msg2="Sikeres adatbeírás.";
        else $msg2="Nem sikerült";
    }
}
```

A submit gomb lenyomása után az inputok értékét elmenti az adatbázisba. Ha sikeres ha nem kapunk egy üzenetet ennek a végkimenetéről. Egy oldal frissítés után meg már láthatjuk is a beírt adatainkat.

A felhasználói dokumentáció

Azért választottam ezt a témát, mert engem személyesen is nagyon érdekelnek a mozik, filmek. Szinte én is mindig ezeket az oldalakat nézegetem, hogy melyik filmeket lenne érdemes megnézni és már foglalom is le rá a jegyet. Ezért gondoltam azt, hogy az iskolámnak egy ilyen oldalt hozzak létre, ha tényleg megvalósulna ez a délutáni filmezés az iskolán belül.

Az oldal csak a számítógép elrendezésére van tervezve, vagyis máshol nem responsive. Ezért gondolom azt, hogy ezzel fejleszteném tovább. Responsive legyen a legtöbb eszközön. Telefonon, tableten, gépen és gépen egyaránt lehessen megtekintetni az oldalt. Mivel nem tudni előre az oldal sikerességét, így a jegyfoglalás mértékét nem szabályoztam, de később kitérnék erre. Például egy székfoglalás részt hoznék létre javascript-el vagy php-val és persze kiírná az eddigi lefoglalt helyeket. A későbbiekben php-val email-es felülettel is foglalkoznék. Mármint foglalás után egy sablon email-t küldene a foglaló email címére a foglalás adatairól.

Az alkalmazott fejlesztői eszközök

A weboldal készítéséhez html, javascript, php, css programnyelveket használtam. A Visual Studio Code-ban dolgoztam. Az adatbázis-kezelő rendszer pedig a Xampp phpmyadmin felületét használtam. Az oldalon segített az iskolában megtanult anyagok, netes felületek, mint pl. <https://www.w3schools.com> (Ez amolyan programozási nyelvek alap dolgait leíró oldal). Hasonló weboldal, ami a bootstrap alap dolgait írja le a <https://getbootstrap.com>. Igazából rengeteg oktató videó is segített amiket leginkább a www.youtube.com-on találtam/néztem meg.



Adatmodell leírása

Egy adatbázis (**kandomozi**) 4 adattáblával. Az adatbázis és a táblák kódolása is **utf8_hungarian_ci**.


Tábla	Művelet	Sorok	Típus	Illesztés	Méret	Felülírás
<input type="checkbox"/> eloadasok	★ Tartalom Szerkezet Keresés Beszúrás Kiürítés Eldobás	17	InnoDB	utf8_hungarian_ci	64.0 KB	-
<input type="checkbox"/> filmek	★ Tartalom Szerkezet Keresés Beszúrás Kiürítés Eldobás	6	InnoDB	utf8_hungarian_ci	16.0 KB	-
<input type="checkbox"/> foglalas	★ Tartalom Szerkezet Keresés Beszúrás Kiürítés Eldobás	0	InnoDB	utf8_hungarian_ci	16.0 KB	-
<input type="checkbox"/> visszajelzes	★ Tartalom Szerkezet Keresés Beszúrás Kiürítés Eldobás	3	InnoDB	utf8_hungarian_ci	16.0 KB	-
4 tábla	Összesen	26	InnoDB	utf8_general_ci	112.0 KB	0 B

Az eloadasok tábla szerkezete:

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
1	Eld 🔑	int(11)			Nem	Nincs		AUTO_INCREMENT
2	Fid 🔑	int(11)			Nem	Nincs		
3	EDatum	date			Nem	Nincs		
4	Eido1	time			Igen	NULL		
5	Eido2	time			Igen	NULL		
6	Eido3	time			Igen	NULL		
7	Eido4	time			Igen	NULL		

- ◆ Eld(ez az elsődleges kulcs, int(11) típusú és auto_increment)
- ◆ Fid(index,int(11))
- ◆ EDatum(date típusú)
- ◆ Eido1-2-3-4(time típusú és nem kötelező kitöltésűek)

Filmek tábla szerkezete:

fId 	int(11)		Nem	Nincs	AUTO_INCREMENT
fCim	varchar(100)	utf8_hungarian_ci	Nem	Nincs	
fImg	varchar(150)	utf8_hungarian_ci	Nem	Nincs	
fMufaj	varchar(50)	utf8_hungarian_ci	Nem	Nincs	
fHossz	int(11)		Nem	Nincs	
fDatum	date		Nem	Nincs	
fRendezok	varchar(50)	utf8_hungarian_ci	Nem	Nincs	
fSzereplok	varchar(50)	utf8_hungarian_ci	Nem	Nincs	
fTartalom	varchar(1000)	utf8_hungarian_ci	Nem	Nincs	
fLink	varchar(100)	utf8_hungarian_ci	Nem	Nincs	

- ◆ fId(ez az elsődleges kulcs, int(11) típusú és auto_increment)
- ◆ fCim(varchar(100) és utf8_hungarian_ci kódolás)
- ◆ fImg(varchar(150) és utf8_hungarian_ci kódolás)
- ◆ fMufaj(varchar(50) és utf8_hungarian_ci kódolás)
- ◆ fHossz(int(11) típusú)
- ◆ fDatum(date típusú)
- ◆ fRendezok(varchar(50) és utf8_hungarian_ci kódolás)
- ◆ fSzereplok(varchar(50) és utf8_hungarian_ci kódolás)
- ◆ fTartalom(varchar(1000) és utf8_hungarian_ci kódolás)
- ◆ fLink(varchar(100) és utf8_hungarian_ci kódolás)

Foglalas tábla szerkezete:

fogId 	int(11)	Nem	Nincs	AUTO_INCREMENT
fogCim	varchar(100) utf8_hungarian_ci	Nem	Nincs	
fogDate	date	Nem	Nincs	
fogTime	time(6)	Nem	Nincs	
fogVnev	varchar(100) utf8_hungarian_ci	Nem	Nincs	
fogKnev	varchar(100) utf8_hungarian_ci	Nem	Nincs	
fogEmail	varchar(100) utf8_hungarian_ci	Nem	Nincs	
fogTelefon	varchar(12) utf8_hungarian_ci	Nem	Nincs	

- ◆ fogId(ez az elsődleges kulcs, int(11) és auto_increment)
- ◆ fogCim(varchar(100) és utf8_hungarian_ci kódolás)
- ◆ fogDate(date típusú)
- ◆ fogTime(time(6) típusú)
- ◆ fogVnev(varchar(100) és utf8_hungarian_ci kódolás)
- ◆ fogKnev(varchar(100) és utf8_hungarian_ci kódolás)
- ◆ fogEmail(varchar(100) és utf8_hungarian_ci kódolás)
- ◆ fogTelefon(varchar(12) és utf8_hungarian_ci kódolás)

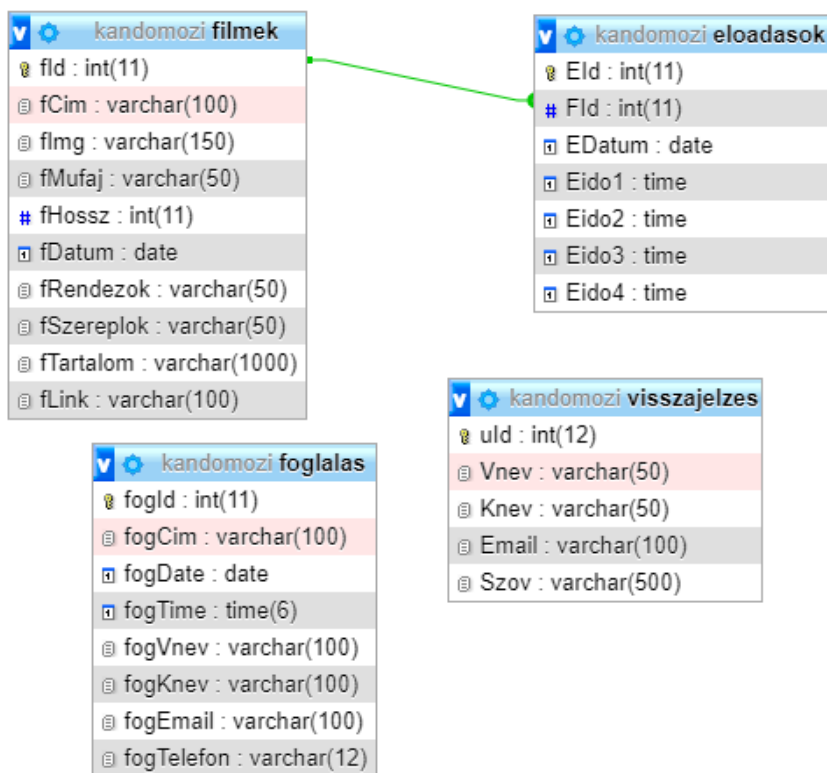
Visszajelzes tábla szerkezete:

uld 🔑	int(12)		Nem	Nincs	AUTO_INCREMENT
Vnev	varchar(50)	utf8_hungarian_ci	Nem	Nincs	
Knev	varchar(50)	utf8_hungarian_ci	Nem	Nincs	
Email	varchar(100)	utf8_hungarian_ci	Nem	Nincs	
Szov	varchar(500)	utf8_hungarian_ci	Nem	Nincs	

- ♦ uld(ez az elsődleges kulcs, int(12) és auto_increment)
- ♦ Vnev(varchar(50) és utf8_hungarian_ci kódolás)
- ♦ Knev(varchar(50) és utf8_hungarian_ci kódolás)
- ♦ Email(varchar(100) és utf8_hungarian_ci kódolás)
- ♦ Szov(varchar(50) és utf8_hungarian_ci kódolás)

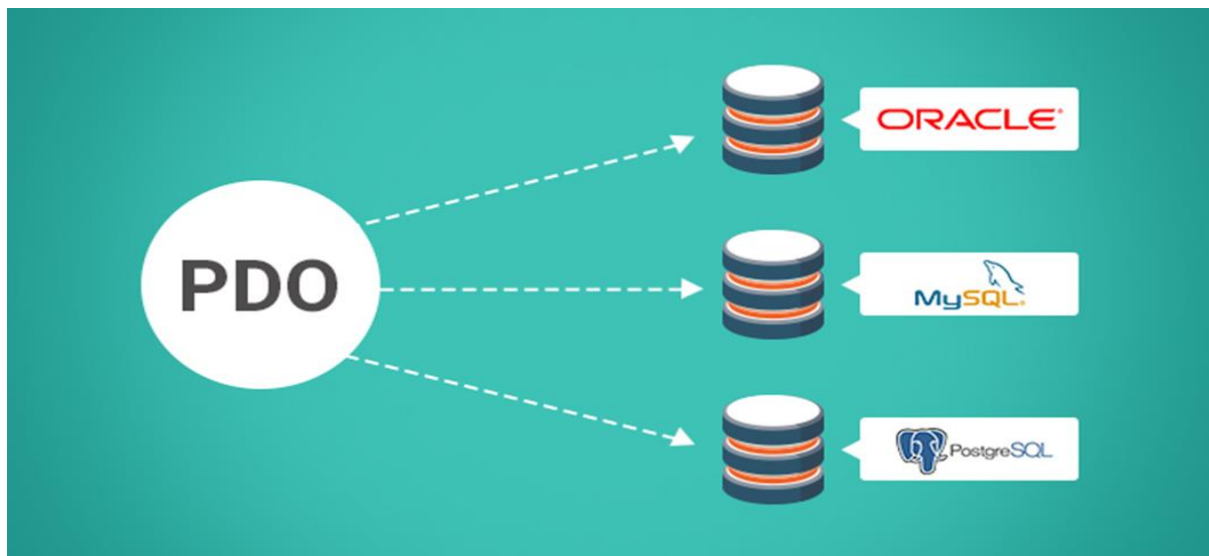
A Tábla kapcsolatok

A filmek.fld=az eloadasok.Fld-vel



A PHP adatbázis kapcsolódás

Az adatbázis absztrakciós réteg. Az adatbázis absztrakció lényege az adatbázis absztrakciós réteg fogalmán keresztül érthető meg. Enélkül egy adatbázis alapú alkalmazás a forráskód tetszőleges részein szétszórva tartalmaz olyan kódrészeket, amelyek az adott natív adatbázissal való kommunikációért felelősek. Ezzel szemben az adatbázis absztrakciós réteg segítségével a kommunikáció központilag menedzselve valósulhat meg. Erre is létezik többféle megoldás, de talán a legelterjedtebb a PDO (PHP Data Objects).



Ezeket alkalmaztam:

PDO FŐBB UTASÍTÁSAI

- `$pdo = new PDO("adatbázis_kapcsolat");`: kapcsolódás az adatbázishoz
- `$pdo = null`: kapcsolat bontása
- `$stmt = $pdo->query("sql");`: lekérdezések (`SELECT`) futtatása
- `$db = $pdo->exec("sql");`: nem lekérdezések esetén
- `$stmt = $pdo->prepare("sql");`: paraméteres SQL utasítások előkészítése
- `$siker = $stmt->execute(paraméter_tömb);`: az előkészített SQL utasítás futtatása a paraméterek behelyettesítésével.

Tesztelés

Szükségünk van a composer letöltésére és a phpUnit használatára. Egy app és egy tests könyvtárat hozzunk létre. Én a Visual Studio Code Terminal-jában dolgoztam.

Ezekkel a parancsokkal megkapjuk a phpUnit verziójának a számát.

```
→ composer require --dev phpunit/phpunit ^9  
→ ./vendor/bin/phpunit --version  
PHPUnit 9.0.0 by Sebastian Bergmann and contributors.
```

A PHP verziókat ezzel a paranccsal: `<?php phpinfo(); ?>`

Miután ezekkel megvagyunk hozzunk létre egy phpUnit.xml-t.

```
<?xml version="1.0" encoding="UTF-8"?>  
  
<phpunit bootstrap="vendor/autoload.php"  
colors="true"  
verbose="true"  
stopOnFailure="false">  
  
<testsuites>  
  
<testsuite name="Test suite">  
  
<directory>tests</directory>  
</testsuite>  
</testsuites>  
</phpunit>
```

Az app könyvtárba hozzuk létre azokat a fájlokat, amelyeket tesztelni szeretnénk. Én a config.php-mat tesztelem:

```
<?php
$host = 'localhost';
$db_name = 'kandomozi';
$db_username = 'root';
$db_password = '';
$options = [PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION, PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC];
try{
    $db = new PDO("mysql:host=$host;dbname=$db_name; charset=utf8",$db_username,$db_password,$options);
}catch(PDOException $e) {
    //echo "hiba:". $e->getMessage();
    //echo "<br>";
    echo "!!! az adatbazis kapcsolodas sikertelen !!!";
    exit;
}
?>
```

A Terminal-ban már csak annyi a dolgunk, hogy leteszteljük. Először direkt átirom és megnézem, hogy dob e rá ki hibát. Az options-nél pl. kitörlöm a végéről az s-t.

```
$db_username = 'root';
$db_password = '';
$options = [PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC, PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8"];
try{
    $db = new PDO("mysql:host=$host;dbname=$db_name; charset=utf8",$db_username,$db_password,$options);
}catch(PDOException $e) {
    //echo "hiba:". $e->getMessage();
    //echo "<br>";
}
```

A hibakódunk meg is jelent:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

There was 1 error:

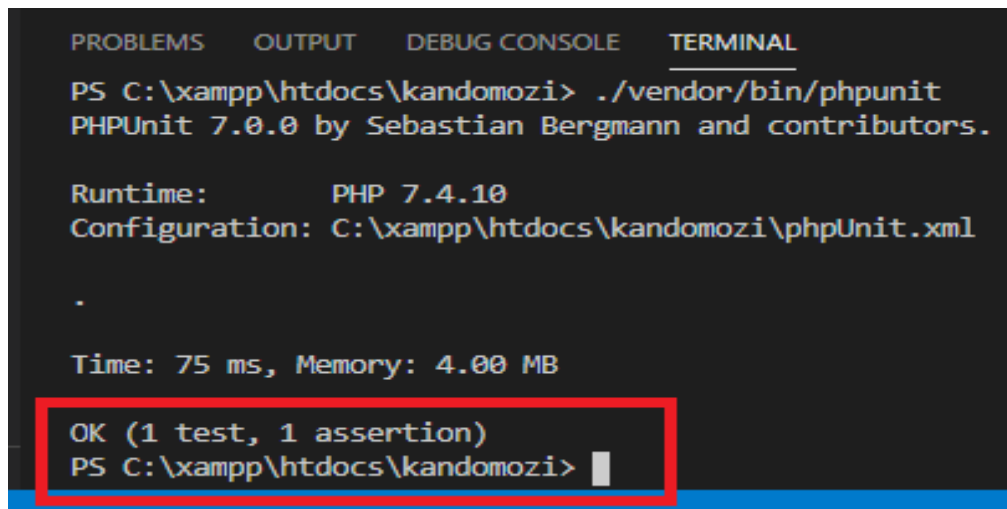
1) ConfigTests::testConnectionOK
Undefined variable: option

C:\xampp\htdocs\kandomozi\app\config.php:9
C:\xampp\htdocs\kandomozi\tests\configDatabaseTest.php:5

ERRORS!
Tests: 1, Assertions: 0, Errors: 1.
PS C:\xampp\htdocs\kandomozi>
```

És most normálisan letesztelve:

A config.php-al minden rendben van!



The image shows a terminal window with a dark background and light-colored text. At the top, there are four tabs: 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', and 'TERMINAL', with 'TERMINAL' being the active tab. The terminal content shows a command prompt where the user has run the PHPUnit command. The output displays the PHPUnit version (7.0.0), the PHP version (7.4.10), and the configuration file path. Below this, a single dot indicates a successful test run. The execution time is 75 ms and memory usage is 4.00 MB. The final line, 'OK (1 test, 1 assertion)', is highlighted with a red rectangular box. The prompt 'PS C:\xampp\htdocs\kandomozi>' is visible at the bottom of the terminal.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

PS C:\xampp\htdocs\kandomozi> ./vendor/bin/phpunit
PHPUnit 7.0.0 by Sebastian Bergmann and contributors.

Runtime:       PHP 7.4.10
Configuration: C:\xampp\htdocs\kandomozi\phpUnit.xml

.

Time: 75 ms, Memory: 4.00 MB

OK (1 test, 1 assertion)
PS C:\xampp\htdocs\kandomozi>
```