



# ZÁRÓDOLGOZAT

---

## KOLLÉGIUMI RÁDIÓ(BACKEND) FEJLESZTÉSE

---

Tanuló  
TELEKI KRISZTIÁN

Mentor  
SZABÓ BEATRIX

Mórahalom, 2024.



## Tartalomjegyzék

Bevezetés .....	3
1. Fejlesztői dokumentáció .....	4
1.1. A projekt céljának ismertetése .....	4
1.2. Az adatbázis ismertetése .....	4
1.3. Az oldalak ismertetése.....	6
1.3.1. Az oldal betöltése.....	6
1.3.2. Fő oldal .....	8
1.3.3. A regisztrációs oldal.....	15
1.4. Rendszerterv.....	16
2.1. Rendszerkövetelmények .....	17
2.2. Program használata .....	17
2.2.1. Bejelentkezés .....	17
2.2.2. Főmenü .....	17
2.2.3. Zene beküldése .....	18
2.2.4. Szavazások.....	18
2.2.5. Kilépés .....	18
2.2.6. Regisztráció .....	19
2.3. Hibajelentés .....	19
Összefoglalás .....	20
Irodalomjegyzék .....	21



## Bevezetés

A diplomamunkám egy egyedülálló projekt köré épül, amelynek középpontjában az iskolai közösségi élet egy különleges aspektusa áll: a kollégiumi ébredések megújítása. Ennek szellemében hoztam létre egy olyan webes platformot, amely újra definiálja a reggeli ébresztés fogalmát iskolám kollégiumában. A cél az volt, hogy létrehozzak egy olyan teret az interneten, ahol a kollégium lakói közösen választhatják meg az ébredéshez szóló zenéket, így minden reggel egy olyan dallamra ébredhetnek, amely összehozza és energiával tölti fel őket.

Ez a weboldal több mint egy egyszerű ébresztő szolgáltatás; ez egy közösségi élmény, ahol mindenki hozzájárulhat a reggeli hangulathoz. A platform felhasználóbarát felülete lehetővé teszi, hogy a regisztrált tagok könnyedén tölthessenek fel zenéket, és szavazhassanak a kollégiumi ébredés során lejátszandó dalokra.

Funkciók:

- Zene feltöltés:  
Regisztrált felhasználók egyszerűen tölthetnek fel zenéket YouTube linkek által.
- Szavazás:  
Minden zene rendelkezik egy számlálóval, amely a felhasználók kedvelésével növekszik és nem kedvelésével csökken.
- Rangsorolás:  
A zenék rangsora a felhasználók szavazatai alapján alakul ki, így a legkedveltebb zene kerül legelőre.



## 1. Fejlesztői dokumentáció

### 1.1. A projekt céljának ismertetése

A projekt célja egy webalkalmazás létrehozása, amely lehetővé teszi felhasználók számára, hogy zenét küldjenek be és ezekre szavazzanak. A fő cél az, hogy a közösség által legkedveltebb zenét rangsoroljuk és megjelenítsük az oldalon. Az oldal fő részét php-teszi ki, valamint a kinézetére css-t használtam. A weboldalt a Visual Studio Code fejlesztői környezet segítségével készítettem, és a XAMPP szoftvercsomagot alkalmaztam a helyi teszteléshez.

Az oldal célja, hogy a kollégium lakói szavazhassanak arról, hogy mi legyen a reggeli ébresztő. A következő funkciókat tartalmazza:

- Regisztráció: Mindenki saját felhasználói fiókkal rendelkezik. Ez lehetővé teszi, hogy mindenki egyszer szavazhasson minden zenére.
- Rangsorolás: A legtöbb szavazattal rendelkező zene a lista tetejére kerül.

### 1.2. Az adatbázis ismertetése

Az adatbázisom a linkek, a regisztráció és a szavazatok táblából áll, amelyeket most szeretnék bemutatni:

Az linkek tábla tartalmazza az alábbi oszlopokat:

- zeneid": A zene azonosítója.
- felhasznalonev: A felhasználó neve, aki a zenét küldte be.
- eloado: A zene előadója.
- cim: A zene címe.
- link: A zene youtube linkje.
- vote: Az adott zenén lévő szavazatok.




1	<b>zeneid</b> 	int(11)
2	<b>felhasznalonev</b>	varchar(255)
3	<b>eloado</b>	varchar(255)
4	<b>cim</b>	varchar(255)
5	<b>link</b>	varchar(255)
6	<b>vote</b>	int(11)

1.ábra – linkek tábla tartalma

A regisztráció tábla tartalmazza az alábbi oszlopokat:

- id: A felhasználó azonosítója.
- felhasznalonev: A felhasználó neve.
- teljesnev: A felhasználó vezeték- és keresztnéve.
- email: A felhasználó e-mail címe.
- jelszo: A felhasználó jelszava titkosítva.
- admin: Rendszergazda jogosultságát jelzi(0-nincs,1-van).

1	<b>id</b> 	int(11)
2	<b>felhasznalonev</b>	varchar(255)
3	<b>teljesnev</b>	varchar(255)
4	<b>email</b>	varchar(255)
5	<b>jelszo</b>	varchar(255)
6	<b>admin</b>	int(11)

2.ábra – regisztráció tábla tartalma

A szavazatok tábla tartalmazza az alábbi oszlopokat:

- zid: A zene azonosítója.
- fid: A zenét beküldő felhasználó azonosítója.
- vote: Ha a felhasználó az adott zenét kedvelte akkor az érték „fel”, ha nem kedvelte akkor az érték „le” lesz.



1	zid	int(11)
2	fid	int(11)
3	vote	varchar(10)

3.ábra – szavazatok tábla tartalma

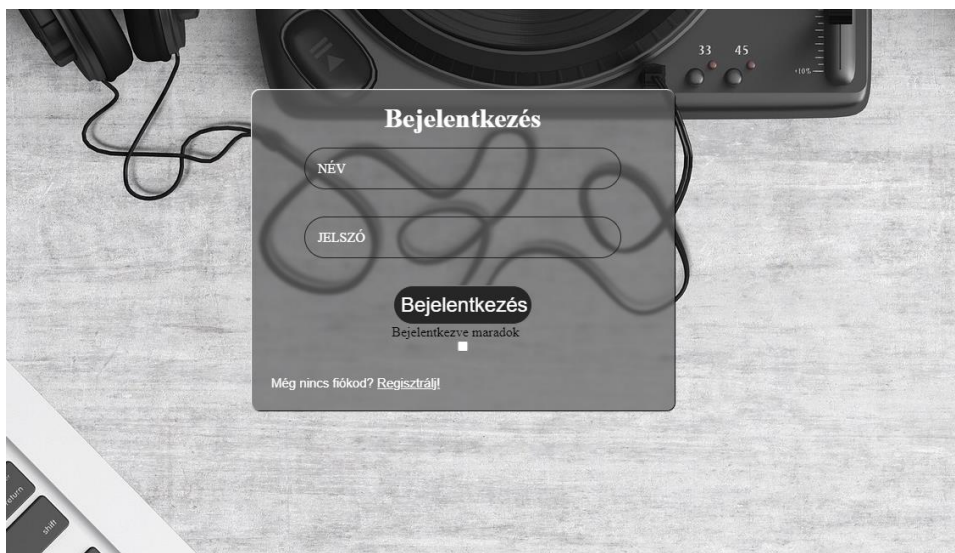
### 1.3. Az oldalak ismertetése

Az oldal három részre osztható:

- Regisztrációs oldal: Ezen az oldalon lehet regisztrálni még nem létező felhasználóval. Az oldal ellenőrzi az e-mail formátumot, valamint, hogy létezik-e a felhasználó/e-mail cím.
- Bejelentkezés oldal: Ezen az oldalon a felhasználóknak lehetőségük van bejelentkezni a rendszerbe. Itt megadják a felhasználónevet és jelszót, majd az oldal ellenőrzi az azonosító adatok helyességét az adatbázisból.
- Fő oldal: Ezen az oldalon lehet beküldeni a zenét a zene előadója, a címe és a YouTube zene linkjét megadva, valamint itt jelenik meg a rangsorolt lista.

#### 1.3.1. Az oldal betöltése

Az oldal megnyitásakor a bejelentkező oldal jelenik meg. Ezen az oldalon kell megadni a bejelentkezéshez szükséges adatokat és a bejelentkezve maradok funkciót, valamint erről az oldalról lehet átmenni a regisztrációs oldalra.



4.ábra – A bejelentkezés oldal

A következő programrész felelős a felhasználó bejelentkezés ellenőrzésére és bejelentkezve maradásért:

```
if(isset($_POST["submitLogin"])){
    $loginName = trim($_POST["username"]);
    $pw = trim($_POST["password"]);
    try{
        if(empty($loginName) || empty($pw)){
            throw new UserException("Felhasználó és jelszó is kötelező");
        }
        require_once("dbconn.php");
        if(empty($dbconn)){
            throw new UserException("Adatbázis kapcsolódási hiba");
        }
        $sqlLogin = "SELECT id, felhasznalonev, jelszo FROM regisztracio WHERE felhasznalonev =:felhasznalonev";
        $queryLogin = $dbconn->prepare($sqlLogin);
        $queryLogin->bindValue("felhasznalonev",$loginName);
        $queryLogin->execute();
        if($queryLogin->rowCount()==0){
            throw new UserException("Hibás felhasználónév");
        }

        $user = $queryLogin->fetch(PDO::FETCH_ASSOC);
        if(!password_verify($pw,$user["jelszo"])){
            throw new UserException("Hibás jelszó");
        }

        $_SESSION["user"] = array("felhasznalonev"=>$user["felhasznalonev"],"id"=>$user["id"]);
        if(isset($_POST["marad"])){
            setcookie("userId",$user["id"],time()+60*60*24);
        }
        header("location:index.php");
        exit();
    }catch(UserException $e){
        $error = "Bejelentkezési hiba: ".$e->getMessage();
    }
}
```

5.ábra – A bejelentkezéshez szükséges program részlet



A bejelentkezés után az oldal saját magának küld egy POST kérelmet, mely megvizsgálja a megadott adatokat és ha valamelyik üres akkor egy kivétellel kezeli és tudatja a felhasználóval, hogy töltsön ki minden mezőt. Ez után lekérdezi az adatbázisból az adatokat ellenőrizve, hogy létezik-e ilyen felhasználónév és a hozzá tartozó jelszó. Ha mindent rendben talált eltárolja az információkat egy SESSION-ben. Itt állítja be a bejelentkezés határidejét is abban az esetben, ha bejelentkezve szeretnénk maradni. Mindezek után az oldal át irányít a fő oldalra.

### 1.3.2. Fő oldal

Bejelentkezés után az oldal meg jeleníti a felhasználó nevet, amivel bejelentkeztünk és a zene beküldésére vonatkozó részt. valamint ez alá a rangsorolt listát.

6.ábra – Fő oldal





A zene beküldés is hasonlóan POST kérelmet küld saját magának:

```
if(isset($_POST['submitBekuld'])){
    $link = trim($_POST["link"]);
    $eloado = trim($_POST['eloado']);
    $cim = trim($_POST['cim']);
    try{
        if(empty($link) || empty($eloado) || empty($cim)){
            throw new UserException("Minden mezőt ki kell tölteni");
        }
        require_once("dbconn.php");
        if(empty($dbconn)){
            throw new UserException("Adatbázis kapcsolódási hiba");
        }

        if (strpos($link, "youtube.com/watch?v=") !== false) {
            $link2 = substr($link, strpos($link, "youtube.com/watch?v=") + strlen("youtube.com/watch?v="));
        } elseif (strpos($link, "youtu.be/") !== false) {
            $id_start = strpos($link, "youtu.be/") + strlen("youtu.be/");
            $id_end = strpos($link, "?si=");
            if ($id_end !== false) {
                $link2 = substr($link, $id_start, $id_end - $id_start);
            } else {
                $link2 = substr($link, $id_start);
            }
        }
        $linkid = $link2;
        $link_check = $dbconn->prepare("SELECT link FROM linkek WHERE link LIKE :link");
        $link_check->bindValue("link", '%' . $linkid . '%');
        $link_check->execute();

        if ($link_check->rowCount() != 0) {
            throw new UserException("Ezt a zene már be lett küldve");
        }

        $ell = '/^https:\\\\www\\.youtube\\.com\\/\\/';
        $ell2 = '/^https:\\\\\\/youtu\\.be\\/\\/';
        if (!preg_match($ell,$link) && !preg_match($ell2,$link)) {
            throw new UserException("Kérlek érvényes youtube linket küldj be");
        }

        $sql = "INSERT INTO linkek(felhasznalonev,eloado,cim,link)
            VALUES(:felhasznalonev,:eloado,:cim,:link)";

        $querynewLink = $dbconn->prepare($sql);
        $querynewLink->bindValue("felhasznalonev",$SESSION['user']['felhasznalonev'],PDO::PARAM_STR);
        $querynewLink->bindValue("eloado",$eloado,PDO::PARAM_STR);
        $querynewLink->bindValue("cim",$cim,PDO::PARAM_STR);
        $querynewLink->bindValue("link",$link,PDO::PARAM_STR);
        $querynewLink->execute();
        echo "<script>alert('A zene sikeresen beküldve');</script>";
    }catch(UserException $e){
        $error = "Beküldési hiba: ".$e->getMessage();
    }
}
```

7.ábra – Zene beküldés

Minden mező kitöltése után a beküldött linket ellenőrzi, hogy az adatbázisban szerepel-e és ellenőrzi, hogy ténylegesen YouTube linket küldtek-e be. Ha nem szerepel az adatbázisban és helyes a link formátuma, akkor feltölti az adatbázist a megfelelő adatokkal.

A következő programrész az adatbázisból lekérdezi az összes linket és át alakítja azokat, hogy csak egy bizonyos része maradjon meg amit az iframe tag kezelni tud, és ezeket egy tömbbe helyezi.



```

require_once("dbconn.php");
$sqlZenek = "SELECT link,felhasznalonev,zeneid,vote FROM linkek ORDER BY vote DESC";
$zenek = $dbconn->query($sqlZenek);
$album = array();
if ($zenek->rowCount() > 0) {
    while($row = $zenek->fetch(PDO::FETCH_ASSOC)) {
        $link = $row["link"];
        $felhasznalonev = $row["felhasznalonev"];
        $zeneid = $row["zeneid"];
        $vote = $row["vote"];
        if (strpos($link, "youtube.com/watch?v=") !== false) {
            $link = substr($link, strpos($link, "youtube.com/watch?v=") + strlen("youtube.com/watch?v="));
        } elseif (strpos($link, "youtu.be/") !== false) {
            $id_start = strpos($link, "youtu.be/") + strlen("youtu.be/");
            $id_end = strpos($link, "?si=");
            if ($id_end !== false) {
                $link = substr($link, $id_start, $id_end - $id_start);
            } else {
                $link = substr($link, $id_start);
            }
        }
        $album[] = array("link" => $link, "nev" => $felhasznalonev, "zeneid" => $zeneid, "vote" => $vote);
    }
}

```

8.ábra – Zenék rangsorolása

A következő rész a szavazás megoldása:

```

if(isset($_POST["submit"])){
    $voteid = explode(";", $_POST["submit"]);
    require_once("dbconn.php");

    $query = "SELECT * FROM szavazatok WHERE fid = :fid AND zid = :zid";
    $stmt = $dbconn->prepare($query);
    $stmt->bindValue("fid", $_SESSION["user"]["id"]);
    $stmt->bindValue("zid", $voteid[0]);
    $stmt->execute();
    $result = $stmt->fetch(PDO::FETCH_ASSOC);

    if (!$result) {
        $insert_query = "INSERT INTO szavazatok (zid, fid) VALUES (:zid, :fid)";
        $insert_stmt = $dbconn->prepare($insert_query);
        $insert_stmt->bindValue("fid", $_SESSION["user"]["id"]);
        $insert_stmt->bindValue("zid", $voteid[0]);
        $insert_stmt->execute();
        $szavazat = array("zid" => $voteid[0], "fid" => $_SESSION["user"]["id"], "vote" => "null");
    } else {
        $szavazat = array("zid" => $result['zid'], "fid" => $result['fid'], "vote" => $result['vote']);
    }

    if($voteid[1] == "+" && $szavazat['vote'] == "le"){
        foreach($album as $elem){
            if($elem["zeneid"] == $voteid[0]){
                $vote = $elem["vote"] + 2;
            }
        }
    }
}

```



```
}

$insert_query = "UPDATE szavazatok SET vote=:vote WHERE zid=:zid AND fid=:fid";
$insert_stmt = $dbconn->prepare($insert_query);
$insert_stmt->bindValue("vote", "fel", PDO::PARAM_STR);
$insert_stmt->bindValue("fid", $_SESSION["user"]["id"], PDO::PARAM_STR);
$insert_stmt->bindValue("zid", $voteid[0], PDO::PARAM_STR);
$insert_stmt->execute();

$sql = "UPDATE linkek SET vote =:vote WHERE zeneid=:zeneid";
$queryUpdate = $dbconn->prepare($sql);
$queryUpdate->bindValue("vote", $vote, PDO::PARAM_STR);
$queryUpdate->bindValue("zeneid", $voteid[0], PDO::PARAM_STR);
$queryUpdate->execute();
header("location:index.php");

}else if($voteid[1] == "+" && $szavazat['vote'] == "null"){
    foreach($album as $elem){
        if($elem["zeneid"] == $voteid[0]){
            $vote = $elem["vote"] + 1;
        }
    }
}

$insert_query = "UPDATE szavazatok SET vote=:vote WHERE zid=:zid AND fid=:fid";
$insert_stmt = $dbconn->prepare($insert_query);
$insert_stmt->bindValue("vote", "fel", PDO::PARAM_STR);
$insert_stmt->bindValue("fid", $_SESSION["user"]["id"], PDO::PARAM_STR);
$insert_stmt->bindValue("zid", $voteid[0], PDO::PARAM_STR);
$insert_stmt->execute();

$sql = "UPDATE linkek SET vote =:vote WHERE zeneid=:zeneid";
$queryUpdate = $dbconn->prepare($sql);
$queryUpdate->bindValue("vote", $vote, PDO::PARAM_STR);
$queryUpdate->bindValue("zeneid", $voteid[0], PDO::PARAM_STR);
$queryUpdate->execute();
header("location:index.php");

}

if($voteid[1] == "-" && $szavazat['vote'] == "fel"){
    foreach($album as $elem){
        if($elem["zeneid"] == $voteid[0]){
            $vote = $elem["vote"] - 2;
        }
    }
}

$insert_query = "UPDATE szavazatok SET vote=:vote WHERE zid=:zid AND fid=:fid";
$insert_stmt = $dbconn->prepare($insert_query);
$insert_stmt->bindValue("vote", "le", PDO::PARAM_STR);
$insert_stmt->bindValue("fid", $_SESSION["user"]["id"], PDO::PARAM_STR);
```



```
$insert_stmt->bindValue("zid",$voteid[0],PDO::PARAM_STR);
$insert_stmt->execute();

$sql = "UPDATE linkek SET vote =:vote WHERE zeneid=:zeneid";
$queryUpdate = $dbconn->prepare($sql);
$queryUpdate->bindValue("vote",$vote,PDO::PARAM_STR);
$queryUpdate->bindValue("zeneid",$voteid[0],PDO::PARAM_STR);
$queryUpdate->execute();
header("location:index.php");

}else if($voteid[1] == "-" && $szavazat['vote'] == "null"){
    foreach($album as $elem){
        if($elem["zeneid"] == $voteid[0]){
            $vote = $elem["vote"] - 1;
        }
    }
}

$insert_query = "UPDATE szavazatok SET vote=:vote WHERE zid=:zid AND fid=:fid";
$insert_stmt = $dbconn->prepare($insert_query);
$insert_stmt->bindValue("vote","le",PDO::PARAM_STR);
$insert_stmt->bindValue("fid",$_SESSION["user"]["id"],PDO::PARAM_STR);
$insert_stmt->bindValue("zid",$voteid[0],PDO::PARAM_STR);
$insert_stmt->execute();

$sql = "UPDATE linkek SET vote =:vote WHERE zeneid=:zeneid";
$queryUpdate = $dbconn->prepare($sql);
$queryUpdate->bindValue("vote",$vote,PDO::PARAM_STR);
$queryUpdate->bindValue("zeneid",$voteid[0],PDO::PARAM_STR);
$queryUpdate->execute();
header("location:index.php");
}
}
```

A programrész lekérdezi az adatbázisból azt a sort, ami meg egyezik a jelenlegi felhasználóval és azzal a zenével amire szavazott. Abban az esetben, ha nincs ilyen sor még akkor létrehoz egy null értékkel rendelkező szavazatot, amelyet később változtat meg. A szavazásnál megvizsgálja, hogy fel vagy leszavaztunk ezt követően ellenőrzi, hogy szavaztunk-e már arra a zenére. Abban az esetben, ha még nem szavaztunk akkor a zene szavazatot növeli, illetve csökkenti eggyel attól függően, hogy mire szavazott. Ha már szavazott az illető akkor ugyan arra a szavazatra nem tud még egyszer szavazni csak az ellenkezőre ilyenkor kettővel növeli, illetve csökkenti a szavazástól függően.

Betöltéskor az oldal le ellenőrzi, hogy a felhasználó admin jogosultságú-e:



```
require_once("dbconn.php");  
$sqlAdmin = "SELECT admin FROM regisztracio WHERE id=:id";  
$stmA = $dbconn->prepare($sqlAdmin);  
$stmA->bindValue("id",$_SESSION["user"]["id"]);  
$stmA->execute();  
$admin = $stmA->fetch(PDO::FETCH_ASSOC);
```

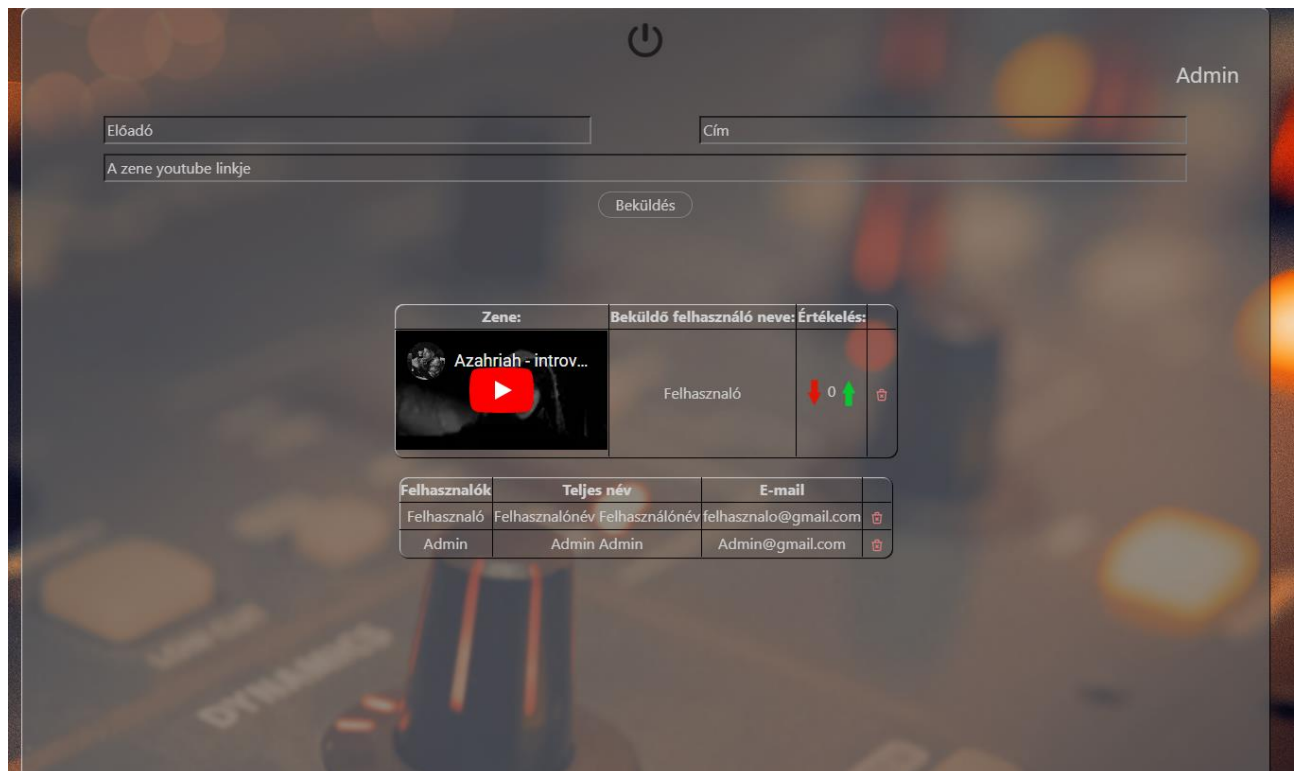
9.ábra –Admin jog

Adatbázisból lekérdezi a felhasználó adatait és az alapján eldönti, hogy rendelkezik-e admin jogosultsággal és egy változóban eltárolja.

A lista kiírásakor, ha admin jogosultság van kibővíti a táblázatot egy törlés gombbal, amivel az adott zenét és a hozzá tartozó adatokat törli az adatbázisból. Egy felhasználókból álló listát is létre hoz ilyenkor és ez által felhasználókat is lehet törölni.

```
if(isset($_POST["torol"])){  
    $zid = $_POST["torol"];  
    require_once("dbconn.php");  
  
    $sqlDeleteMusic = "DELETE FROM linkek WHERE zeneid = :id";  
    $stmtDeleteMusic = $dbconn->prepare($sqlDeleteMusic);  
    $stmtDeleteMusic->bindValue("id",$zid,PDO::PARAM_STR);  
    $stmtDeleteMusic->execute();  
  
    $sqlDeleteVotes = "DELETE FROM szavazatok WHERE zid = :id";  
    $stmtDeleteVotes = $dbconn->prepare($sqlDeleteVotes);  
    $stmtDeleteVotes->bindValue("id",$zid,PDO::PARAM_STR);  
    $stmtDeleteVotes->execute();  
    header("location:index.php");  
}  
if(isset($_POST["Ftorol"])){  
    $nev = $_POST["Ftorol"];  
    require_once("dbconn.php");  
  
    $sqlDeleteMusic = "DELETE FROM regisztracio WHERE felhasznalonev = :nev";  
    $stmtDeleteMusic = $dbconn->prepare($sqlDeleteMusic);  
    $stmtDeleteMusic->bindValue("nev",$nev,PDO::PARAM_STR);  
    $stmtDeleteMusic->execute();  
    header("location:index.php");  
}
```

10.ábra – Törlés rész



11.ábra – Admin szemszög



### 1.3.3. A regisztrációs oldal

A regisztrációs oldalon egy egyszerű űrlap kitöltése után szintén POST kérelmet küld saját magának.

The image shows a registration form titled "Regisztráció" (Registration) overlaid on a background of a vinyl record. The form contains the following elements:

- A title "Regisztráció" in a large, bold, serif font.
- Four input fields with rounded ends, each containing a label in all caps: "FELHASZNÁLÓNÉV" (Username), "VEZETÉKNÉV" (Surname), "KERESZTNÉV" (First Name), and "E-MAIL".
- A fifth input field with rounded ends labeled "JELSZÓ" (Password).
- A dark, rounded rectangular button with the text "Regisztráció" in white.
- At the bottom, the text "Van már fiókod? [Jelentkezz be!!](#)" (Do you already have an account? [Sign in!!](#)).

12.ábra – Regisztrációs oldal

A regisztráció során ellenőrizzük a felhasználónevet, hogy meggyőződjünk róla, hogy még nem szerepel az adatbázisban. A regisztráció során továbbá ellenőrizzük az e-mail címet annak érdekében, hogy megbizonyosodjunk arról, hogy megfelelő formátumban van-e megadva, valamint, hogy még nem szerepel-e az adatbázisban. Ha mindkét ellenőrzés (felhasználó-név és e-mail) sikeresen lefut, akkor az adatbázisba rögzítjük az új felhasználó adatait.



```
if (isset($_POST["submitRegister"])){
    $felhasznalonev = trim($_POST["felhasznalonev"]);
    $teljesnev = trim($_POST["vezeteknev"]." ".$_POST["keresztnév"]);
    $email = trim($_POST["email"]);
    $jelszo = trim($_POST["jelszo"]);

    try{
        if(empty($felhasznalonev) || empty($teljesnev) || empty($jelszo) || empty($email)){
            throw new UserException("Minden adat kötelező");
        }
        require_once("dbconn.php");

        $stmt_check = $dbconn->prepare("SELECT felhasznalonev FROM regisztracio WHERE felhasznalonev = :felhasznalonev");
        $stmt_check->bindValue("felhasznalonev", $felhasznalonev);
        $stmt_check->execute();

        if ($stmt_check->rowCount() != 0) {
            throw new UserException("Felhasználónév már foglalt");
        }
        if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
            throw new UserException("Rossz email formátum");
            exit();
        }

        $stmt_check_email = $dbconn->prepare("SELECT email FROM regisztracio WHERE email = :email");
        $stmt_check_email->bindValue("email", $email);
        $stmt_check_email->execute();

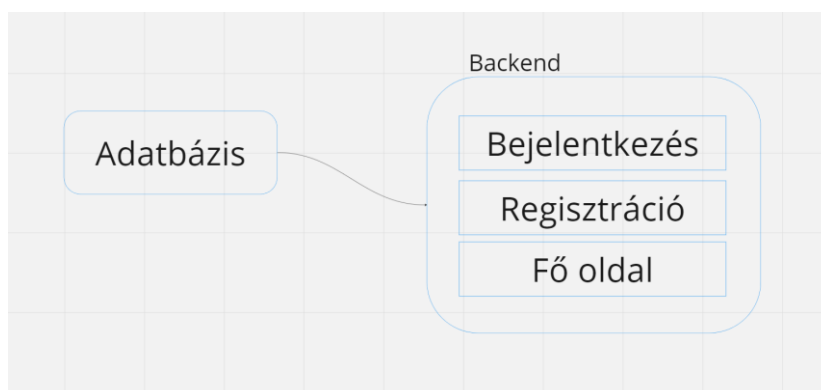
        if($stmt_check_email->rowCount() != 0){
            throw new UserException("Ezzel az email címmel már regisztráltak");
        }

        $pwhash = password_hash($jelszo, PASSWORD_DEFAULT);
        $sql = "INSERT INTO regisztracio(felhasznalonev,teljesnev,email,jelszo)
            VALUES(:felhasznalonev,:teljesnev,:email,:jelszo)";

        $querynewUser = $dbconn->prepare($sql);
        $querynewUser->bindValue("felhasznalonev",$felhasznalonev,PDO::PARAM_STR);
        $querynewUser->bindValue("teljesnev",$teljesnev,PDO::PARAM_STR);
        $querynewUser->bindValue("jelszo",$pwhash,PDO::PARAM_STR);
        $querynewUser->bindValue("email",$email,PDO::PARAM_STR);
        $querynewUser->execute();
        header("location:login.php?s");
        exit();
    }catch(UserException $e){
        $error = "Sikertelen regisztráció: ".$e->getMessage();
    }catch(PDOException $e){
        $error = "Adatbázis mentési hiba: ".$e->getMessage();
    }
}
```

13.ábra – Regisztrálás

## 1.4. Rendszerterv



14.ábra – Rendszerterv





## 2. Felhasználói dokumentáció

Ez az oldal a kollégiumi reggeli ébresztő zenéjének kiválasztására lett létrehozva. Az oldal felhasználóbarát és egyszerűen kezelhető felülettel rendelkezik.

### 2.1. Rendszerkövetelmények

Az oldal minden internetkapcsolattal rendelkező eszközön elérhető, amelyen található valamilyen webböngésző.

### 2.2. Program használata

A weboldal használata a következő:

#### 2.2.1. Bejelentkezés

Amikor meglátogatja a weboldalt, először a bejelentkezési oldal tárul elé. Adja meg felhasználónevét és jelszavát a megfelelő mezőkbe, majd kattintson a "Bejelentkezés" gombra. Amennyiben szeretne bejelentkezve maradni, jelölje be a „Bejelentkezve maradok” opciót.



15.ábra – Bejelentkezés rendszergazdaként

#### 2.2.2. Főmenü

A bejelentkezés után a fő menü jelenik meg, ahol a zene beküldésére szolgáló felület, alatta pedig a felhasználók által rangsorolt zenék listája található.



### 2.2.3. Zene beküldése

A zene beküldéséhez három mezőt kell kitölteni:

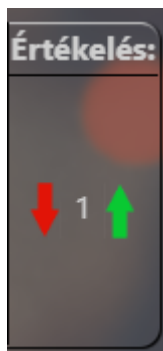
1. Előadó: Ide írja be a zene előadójának nevét.
2. Cím: Ide írja be a zene címét.
3. YouTube link: Ide másolja be a zene YouTube linkjét.

Ezeket a mezőket kitöltve és a "Beküldés" gombra kattintva tudja elküldeni a zenét a rangsorolásra.

16.ábra –Zene beküldése

### 2.2.4. Szavazások

Minden felhasználó csak egyszer szavazhat minden zenére: vagy „fel” (ha tetszik neki) vagy „le” (ha nem tetszik neki). Egy adott felhasználó csak egyféle szavazatot adhat le egy adott zene mellett.



17.ábra –Szavazás

### 2.2.5. Kilépés

A kilépéshez az oldal tetején található kikapcsoló gombra kell kattintani.



18.ábra –Kilépés gomb



### 2.2.6. Regisztráció

A regisztráció során meg kell adni a következő adatokat:

- **Felhasználónév:** Ez a név fog megjelenni az általa beküldött zenék mellett.
- **Vezetéknév:** A családnévedet kell megadni.
- **Keresztnév:** A keresztnévét kell megadnia.
- **E-mail cím:** Az e-mail címét kell megadnia.
- **Jelszó:** Egy jelszót kell meg adnia.

Ezek az adatok szükségesek a regisztrációhoz és a későbbi belépéshez az oldalon. A felhasználónév fog megjelenni az általa beküldött zenék mellett a rangsorolási folyamat során.

### 2.3. Hibajelentés

Az oldal minden hiba esetén értesíti a felhasználót a következőképpen:

- Ha valamilyen adatot rosszul vagy hiányosan tölt ki a regisztráció vagy más funkciók során, akkor az oldal megjelenít egy értesítést vagy hibaüzenetet a felhasználónak.
- Az értesítés vagy hibaüzenet pontosan jelzi, hogy mi a probléma vagy mi az hiányzik. Például: „Sikertelen regisztráció: Minden adat kötelező”, „Sikertelen regisztráció: Rossz email formátum”, „Bejelentkezési hiba: Hibás felhasználónév”.
- Az ilyen hibaüzenetek segítenek a felhasználónak gyorsan korrigálni a hibát, és folytatni az adott műveletet vagy funkciót.



## Összefoglalás

Nagyon elégedett vagyok a munkámmal. A felhasználói dokumentáció részletesen bemutatja az oldal minden részét és használatát. Az oldal használatát részletesen ismertettem, minden lépést és funkciót áttekintettem, és szemléltető képeket is beillesztettem a dokumentációba. A dokumentáció tartalmazza a lehetséges hibajelzéseket, amelyek segítenek a felhasználónak azonosítani és megoldani a problémákat.

Az oldal kifejezetten azért jött létre, hogy segítse a kollégiumok mindennapi reggeli felkészülését. Ennek értelmében az oldal alkalmas több kollégium számára is, és mindegyikük saját egyedi listát állíthat össze és használhat. Az oldal könnyen kezelhető és egyszerű felülete révén szinte bárki könnyedén használhatja.

Összességében úgy vélem, hogy az elkészült munkám hasznos eszköz lesz a diákok mindennapi életében. Az oldal továbbfejlesztése számos lehetőséget kínál a diákok számára.

Íme néhány javaslat a továbbfejlesztési lehetőségekre:

1. Napi listák: Minden nap egy új lista kerülne létrehozásra, amely által változatosabb lennének az ajánlott zenék.
2. Egyéni listák: Minden felhasználó magának hozhat létre listákat.
3. Jelszó frissítés: A felhasználó meg tudja változtatni a jelszavát.



## Irodalomjegyzék

W3Schools (<https://www.w3schools.com/php/> )

Kozmajer Viktor(2011): PHP és MySQL az alapoktól