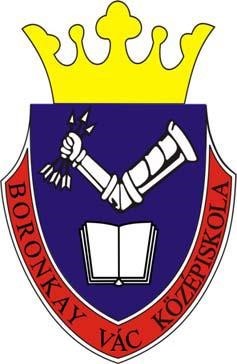
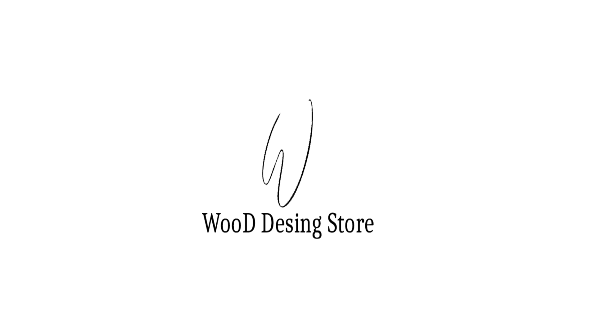
**VÁCI SZAKKÉPZÉSI CENTRUM**  
**BORONKAY GYÖRGY**  
**MŰSZAKI TECHNIKUM ÉS GIMNÁZIUM**



**VIZSGAREMEK**

***Wood Design Store***



Konzulens: Wiezl Csaba Készítette: Piszár Tamás

Forgács Tamás

# Konzultációs lap

Vizsgázók neve: Piszár Tamás

Forgács Tamás

Vizsgaremek címe: Wood Design Store

|  |  |  |
| --- | --- | --- |
| Sorszám | A konzultáció időpontja | A konzulens aláírása |
| 1. | 2023.10.17. |  |
| 2. | 2023.11.21. |  |
| 3. | 2023.12.13. |  |
| 4. | 2024.01.16 |  |
| 5. | 2024.02.20. |  |
| 6. | 2024.03.12 |  |

|  |  |  |
| --- | --- | --- |
| A vizsgaremek beadható: |  | A vizsgaremeket átvettem: |
| Vác, 2024. |  | Vác, 2024. |
| Konzulens |  | A szakképzést folytató intézmény felelőse |

Tartalomjegyzék

[Konzultációs lap 2](#_Toc1067662530)

[Témaválasztás Indoklása és Dokumentációja: Fa Termékek Webáruháza 4](#_Toc1747285581)

[Témaválasztás 5](#_Toc1806237999)

[1. Fejlesztői dokumentáció 5](#_Toc1951306559)

[Áttekintés 6](#_Toc628261315)

[Fő Komponensek 6](#_Toc1184617778)

[Adatbázis Kapcsolat (MySQL) 6](#_Toc19359784)

[Felhasználókezelés 6](#_Toc1792152934)

[Termékkezelés 6](#_Toc2004889971)

[Kosárkezelés 6](#_Toc2075244613)

[Felhasználói Felület 6](#_Toc240159164)

[Fontos Fájlok és Mappák 6](#_Toc1340785747)

[Kapcsolódás az adatbázishoz 7](#_Toc1465700391)

[Felhasználói bejelentkezés ellenőrzése 7](#_Toc1464775989)

[Felhasználói adatok lekérése 8](#_Toc266966508)

[Kosárba helyezés 8](#_Toc441102613)

[Kosár megjelenítése és kezelése 9](#_Toc1496808079)

[Fizetési folyamat 9](#_Toc1667764519)

[1.1Felhasználói bejelentkezési rendszer 9](#_Toc1626525765)

[1.2Felhasználói munkamenetek 10](#_Toc2006425213)

[1.3Adminisztrációs felületet 11](#_Toc268531269)

[1.4Új adminisztrátor 13](#_Toc504817866)

[1.5Felhasználói adatok frissítése 13](#_Toc1287457838)

[1.6Felhasználó végleges kitiltásának feloldása 14](#_Toc1442944589)

[1.7Felhasználó korlátozása 16](#_Toc1201661732)

[1.8Felhasználó szerkesztése 17](#_Toc456876698)

[1.9Admin adatok frissítése 18](#_Toc1047123091)

[1.10 Felhasználó Törlése 20](#_Toc1237152736)

[1.11 Felhasználók Listázása 20](#_Toc501889904)

[1.12 Fizetés és Szállítás Űrlap 22](#_Toc1802802640)

[1.13 Készlet Módosítása Űrlap 22](#_Toc1123632433)

[1.14 Kosár Tartalom 23](#_Toc265592046)

[Cél 24](#_Toc1475309046)

[Technikai Implementáció 24](#_Toc2137300326)

[Főbb Komponensek 24](#_Toc2074674905)

[Felhasznált Technológiák 24](#_Toc782824649)

[Fejlesztői Megjegyzés 24](#_Toc515809566)

[1.15 Bejelentkező Oldal 25](#_Toc785980934)

[Cél 25](#_Toc1190914382)

[Technikai Implementáció 25](#_Toc159343577)

[Főbb Komponensek 25](#_Toc159062450)

[Fejlesztői Megjegyzés 26](#_Toc1557867261)

[1.16 Felhasználó Bejelentkezés 26](#_Toc464541370)

[1.17 Felhasználó Regisztráció 27](#_Toc1629226069)

[Ez a PHP szkript a felhasználók regisztrációját végzi egy webalkalmazásban. 28](#_Toc88404531)

[1.18 Rólunk Oldal 28](#_Toc1163825965)

[Oldalstruktúra 29](#_Toc362606557)

[Technikai Részletek 29](#_Toc848303296)

[1.19 Termék hozzáadása űrlap PHP kódja: 30](#_Toc250850727)

[Funkciók 31](#_Toc1824396734)

[Adatmezők 31](#_Toc199090679)

[1.20 Termékek sorrendjének módosítása 31](#_Toc395629630)

[1.21 Termékek törlése 32](#_Toc1029375819)

[1.22 Termékek listázása 33](#_Toc999514029)

[2. Adatszerkezet 34](#_Toc1462553533)

[Bevezetés 35](#_Toc1359381059)

[Rendszerkövetelmények 35](#_Toc582474754)

[Folyamat 35](#_Toc480999905)

[Változók és Objektumok 36](#_Toc1137488116)

[Hibakezelés 36](#_Toc927121911)

[Biztonsági megfontolások 36](#_Toc1140190918)

[Az “Admin\_users” tábla 37](#_Toc256555714)

[Tábla struktúra 37](#_Toc120238577)

[Használat 38](#_Toc1636745652)

[Példa SQL lekérdezések 38](#_Toc307100083)

[A “termékek” tábla 38](#_Toc1629306259)

[Tábla struktúra 39](#_Toc1475750602)

[Használat 39](#_Toc1885591335)

[Biztonsági megfontolások 40](#_Toc611875512)

[A “users” tábla 41](#_Toc182293515)

[Használat 41](#_Toc526314762)

[Biztonsági megfontolások 41](#_Toc866414144)

[Példa phpMyAdmin műveletek 42](#_Toc1939236007)

[3. Felhasználói Dokumentáció 43](#_Toc1050734233)

[Áttekintés 43](#_Toc1091588264)

[Hogyan működik? 43](#_Toc827035314)

[Használati utasítások 43](#_Toc925115644)

[3.1 Főbb funkciók és tartalom 44](#_Toc1995433156)

[Használati utasítások 44](#_Toc2027553550)

[Fontos tudnivalók 44](#_Toc2067857233)

[Technikai részletek 44](#_Toc721615593)

[Kosárhasználati útmutató 45](#_Toc777442988)

[Egyéb Fontos Tudnivalók 45](#_Toc1093982766)

[3.2 Technikai Részletek és Frissítések 45](#_Toc1369413603)

[Kosárhasználati útmutató folytatása 45](#_Toc1083603799)

[Egyéb Fontos Tudnivalók (folytatás) 45](#_Toc908149653)

[Technikai Részletek 45](#_Toc1863359408)

[3.3 Jogi Tudnivalók és Adatvédelem 46](#_Toc960268131)

[Kapcsolat és További Információk 46](#_Toc416265075)

[Visszajelzés és Értékelés 46](#_Toc1325120436)

[4. Tesztelés 47](#_Toc2020653291)

[4.1 Tesztelési lépések 48](#_Toc1731639663)

[4.2 Tesztelési funkciók és dokumentáció 49](#_Toc143537161)

[Felhasználó bejelentkezésének ellenőrzése: 52](#_Toc1426440699)

[5. Irodalomjegyzék 60](#_Toc1510721615)

[6. Mellékletek 61](#_Toc966513002)

# Témaválasztás Indoklása és Dokumentációja: Fa Termékek Webáruháza

A Fa Termékek Webáruháza projekt célja egy online platform létrehozása, amely kifejezetten fa alapú termékekre specializálódik. Ennek az indoklásnak és dokumentációnak a célja, hogy részletesen bemutassa a projekt hátterét, céljait, funkcionalitását, valamint a technológiai megoldásokat és rendszerarchitektúrát.

## Témaválasztás

A fa termékek népszerűsége és keresletük növekedése napjainkban egyre inkább észrevehetővé válik. A természetes anyagok iránti igény és a fenntarthatóság fontossága miatt a fák és fa alapú termékek újra előtérbe kerülnek a lakberendezési és kerti kiegészítők piacán. Ennek megfelelően a Fa Termékek Webáruháza projekt egy olyan válasz a piacon tapasztalható igényekre, amely lehetővé teszi a vásárlók számára, hogy könnyen megtalálják és megvásárolják a kívánt fa termékeket egy kényelmes és inspiráló online környezetben.

**A témaválasztás indokai között szerepel:**

1. **Növekvő kereslet**: A fa alapú termékek iránti kereslet folyamatosan növekszik, ami a projekt sikeres piaci bevezetésének előfeltétele.
2. **Fenntarthatóság**: A fa termékek egyre inkább a fenntarthatóság és környezetvédelem szimbólumává válnak, amelyek elősegítik a természetes anyagokkal való kapcsolatot.
3. **Inspiráció és kényelem**: Az online webáruházak előnyeit kihasználva, a Fa Termékek Webáruháza lehetőséget nyújt az inspiráló vásárlási élményre és a kényelmes otthoni berendezésre.

# 1. Fejlesztői dokumentáció

Ez a dokumentáció a webshop PHP kódját részletesen bemutatja és dokumentálja. A kód egy egyszerű webáruházat valósít meg, amely lehetővé teszi a felhasználók számára a termékek böngészését, kosárba helyezését és a fizetést.

## Áttekintés

A PHP kód a következő főbb funkciókat valósítja meg:

* Felhasználók bejelentkezése és regisztrációja.
* Termékek listázása kategóriánként.
* Kosár kezelése: termékek hozzáadása, eltávolítása, kosár tartalmának megjelenítése.
* Átirányítás a fizetési oldalra.

## Fő Komponensek

### **Adatbázis Kapcsolat (MySQL)**

A **mysql** osztályt használva kapcsolódik az adatbázishoz, ahol tárolódnak a felhasználók és a termékek adatai.

### **Felhasználókezelés**

* Bejelentkezés: Az email cím alapján ellenőrzi a felhasználó jelszavát és bejelentkezteti, ha a jelszó helyes.
* Regisztráció: Új felhasználók regisztrálása az adatbázisban.

### **Termékkezelés**

* Termékek listázása: A készleten lévő termékek megjelenítése az adatbázisból.
* Kosárba helyezés: A felhasználók által kiválasztott termékek hozzáadása a kosárhoz.

### **Kosárkezelés**

* Kosár megjelenítése: A felhasználó kosarában lévő termékek listázása.
* Termék eltávolítása a kosárból.

### **Felhasználói Felület**

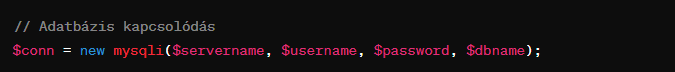
* HTML és CSS használata a felhasználói felület kialakításához.
* Reszponzív dizájn: A felület alkalmazkodik a különböző eszközök képernyőméretéhez.

## Fontos Fájlok és Mappák

* **index.php**: A főoldal, amelyen keresztül a felhasználók böngészhetik a termékeket és kezelhetik a kosarukat.
* **login.php**, **register.php**: Bejelentkezési és regisztrációs oldalak.
* **logout.php**: Kijelentkezési folyamatot kezeli.
* **fizetes.php**: Az átirányítás a fizetési folyamathoz.

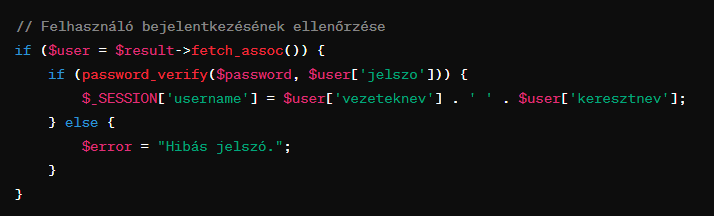
### **Kapcsolódás az adatbázishoz**

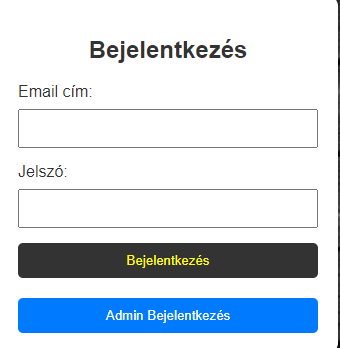
A kód kezdetén inicializáljuk a PHP session-t és csatlakozunk az adatbázishoz a MySQL adatbázis szerveren. A kapcsolatot felépítjük a megadott felhasználónévvel, jelszóval és adatbázisnévvel. Ha a kapcsolat nem sikerül, akkor hibaüzenetet jelenítünk meg és leállítjuk a programot.



### **Felhasználói bejelentkezés ellenőrzése**

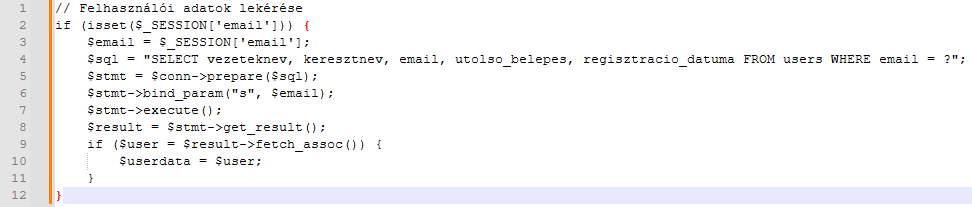
A felhasználói bejelentkezés ellenőrzését egy előre készített "users" táblában tárolt adatok alapján végezzük. A felhasználói e-mail alapján keresünk a táblában, és ha megtaláljuk a felhasználót, ellenőrizzük a megadott jelszót a táblában tárolttal. Ha a jelszó helyes, beállítjuk a felhasználó nevét a session változóba. Ha nem, akkor hibaüzenetet generálunk.





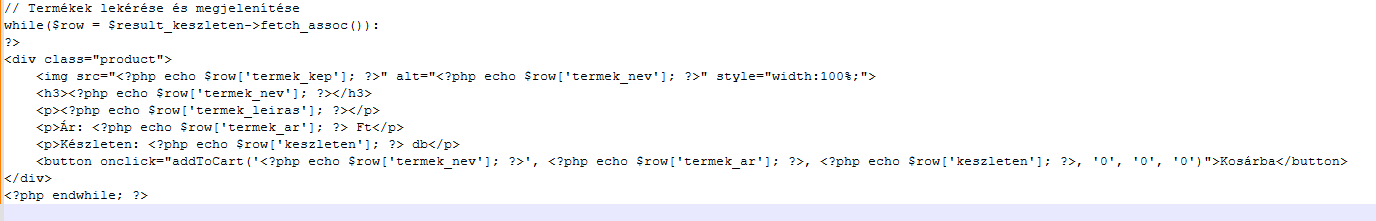
### **Felhasználói adatok lekérése**

Ha a felhasználó bejelentkezett, további adatokat kérhetünk le a felhasználóról, például vezetéknév, keresztnév, e-mail cím, utolsó bejelentkezés ideje stb. Ezeket az adatokat a "users" táblából kérjük le az e-mail cím alapján.



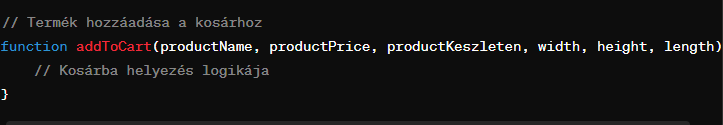
**Termékek megjelenítése**

A weboldal főoldalán a termékeket a "termekek" táblából kérjük le, amelyeket a "megjelenitesi\_hely" mező értéke alapján választunk ki. A termékek listáját a felhasználó számára megjelenítjük, beleértve a termék nevét, leírását, árát és készleten lévő darabszámát.



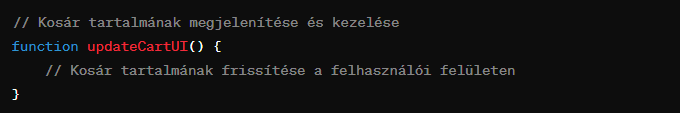
### **Kosárba helyezés**

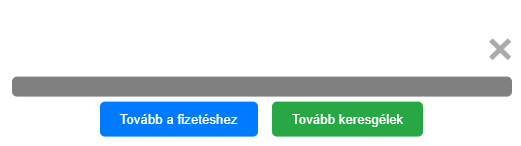
Minden termék mellett egy gomb található, amely lehetővé teszi a felhasználók számára, hogy a terméket a kosárba helyezzék. A gombra kattintva a termék adatait, például a nevét, árát és készletszámát összegyűjtjük, majd ezeket az adatokat a kosárba helyezéshez tároljuk. Amennyiben a termékből még van készleten, a kosárba helyezés sikeres lesz, egyébként hibaüzenet jelenik meg.



### **Kosár megjelenítése és kezelése**

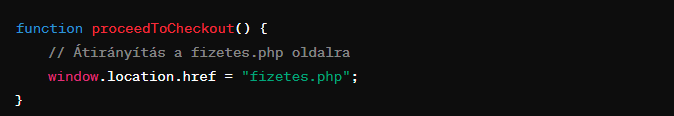
A felhasználói felületen egy ikonra kattintva megjelenik a kosár tartalma. Itt a felhasználók láthatják, hogy milyen termékeket helyeztek a kosárba, a mennyiségüket és az egyes termékek árát. Lehetőség van a termékek törlésére a kosárból is. A kosár tartalmát egy model ablakban jelenítjük meg.



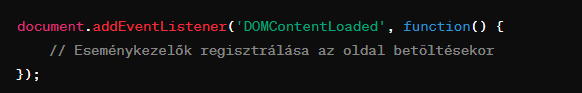


### **Fizetési folyamat**

A felhasználók tovább léphetnek a fizetési folyamathoz, amelyre egy gomb megnyomásával kerül sor. Ekkor átirányítjuk őket a "fizetes.php" oldalra, ahol a vásárlást folytathatják.

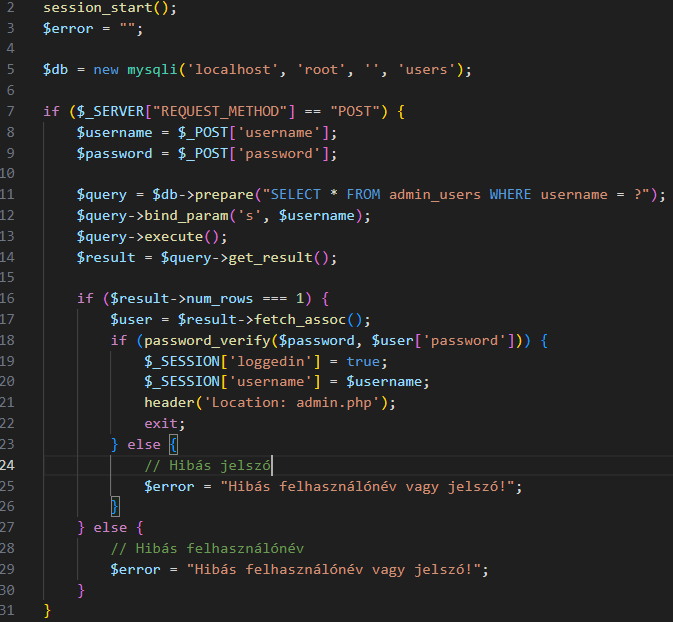


**Eseménykezelők:**



# 1.1 Felhasználói bejelentkezési rendszer

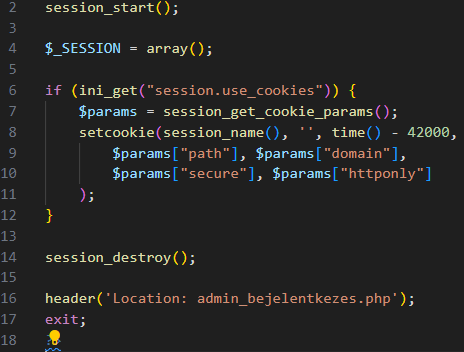
**Ez a PHP kód egy egyszerű felhasználói bejelentkezési rendszert valósít meg: Admin**

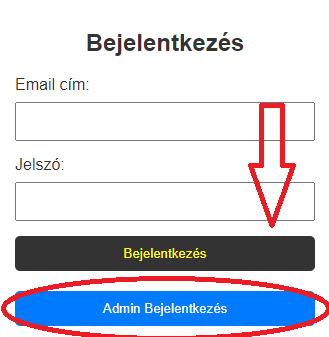
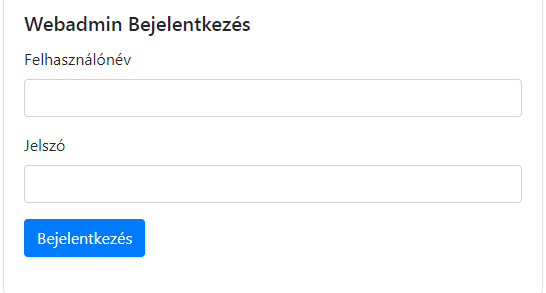


1. **session\_start()**: Ez a függvény indítja el a munkamenetet, ami lehetővé teszi az adatok tárolását az egyes látogatók számára.
2. **$error változó**: Ez a változó a bejelentkezési hibák kezelésére szolgál. Ha valamilyen hiba történik (pl. helytelen felhasználónév vagy jelszó), akkor ebbe a változóba kerül a hibaüzenet.
3. **Adatbázis csatlakozás létrehozása**: Az adatbázis kapcsolatot hoz létre az adatbázis szerverrel a mysqli osztály segítségével. A 'localhost', 'root', '', 'users' paraméterek az adatbázis szerver elérési útját, felhasználónevét, jelszavát és az adatbázis nevét adják meg. Ezt a kapcsolatot a **$db** változó tárolja.
4. **POST kérés kezelése**: Ellenőrzi, hogy a kérés POST metódussal érkezett-e a szerverre. Ha igen, akkor a beérkező felhasználónév és jelszó értékeket elmenti a **$username** és **$password** változókba.
5. **Felhasználó azonosítása az adatbázisban**: A felhasználó által megadott felhasználónév alapján lekérdezi az adatbázisból az adott felhasználó adatait. Ha a lekérdezés eredménye egyetlen sor, akkor a felhasználó létezik az adatbázisban, és a jelszava helyes.
6. **Jelszó ellenőrzése**: A **password\_verify()** függvény segítségével összehasonlítja a felhasználó által megadott jelszót a tárolt jelszó hash-szel az adatbázisban. Ha a jelszavak megegyeznek, akkor a bejelentkezés sikeres.
7. **Munkamenet változók beállítása**: Ha a bejelentkezés sikeres, akkor az **$\_SESSION['loggedin']** és **$\_SESSION['username']** változókat beállítja, majd átirányítja a felhasználót az adminisztrációs felületre (**admin.php**).
8. **Hibakezelés**: Ha a felhasználónév vagy a jelszó helytelen, akkor a **$error** változóban tárolja a hibaüzenetet, amit később megjeleníthet a felhasználónak.

# 1.2 Felhasználói munkamenetek

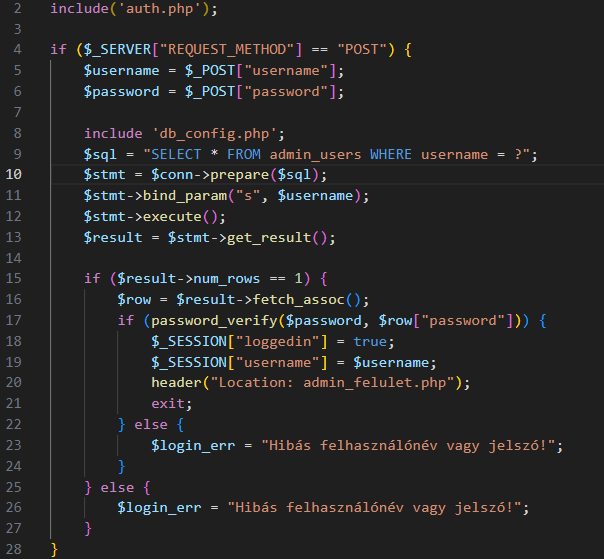
**Ez a PHP kód a felhasználói munkamenetek lezárását és a felhasználó kijelentkeztetését végzi el:**



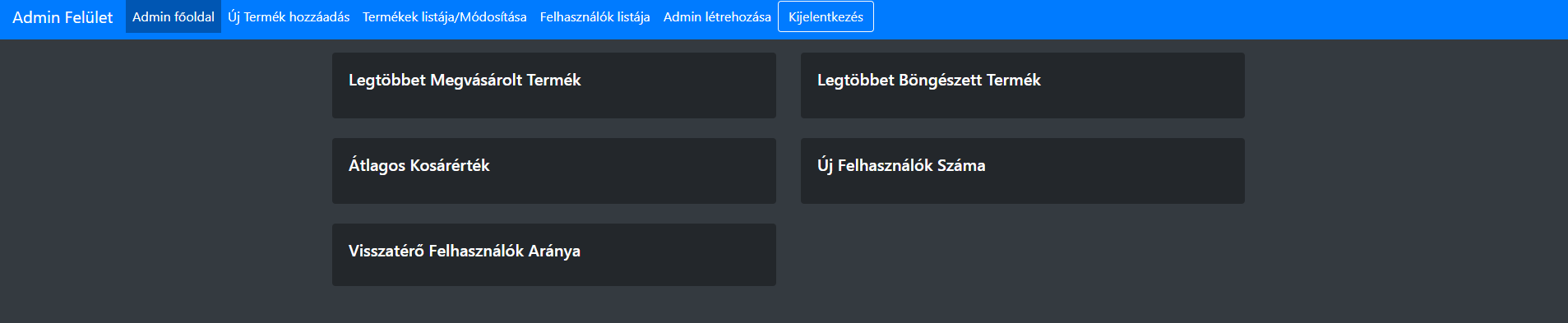
1. **session\_start()**: Elindítja a munkamenetet, hogy a PHP tudja kezelni a munkamenet változókat.
2. **$\_SESSION törlése**: A **$\_SESSION** globális tömb ürítése révén minden munkamenet változót töröl a felhasználói munkamenetből. Ez azért fontos, hogy teljesen megszüntesse a felhasználó bejelentkezett állapotát és más munkamenet-specifikus adatokat.
3. **session\_get\_cookie\_params() és setcookie()**: Ezek a függvények beállítják a munkamenet cookie-jának élettartamát, így a böngészőben tárolt munkamenet azonosító hamarosan lejár és érvénytelen lesz. Ez egyfajta biztonsági intézkedés, hogy megakadályozza a lejárt munkamenet cookie használatát.
4. **session\_destroy()**: Ez a függvény teljesen megszünteti a munkamenetet, beleértve a munkamenetet kezelő szerveroldali változók és a munkamenet azonosító törlését is.
5. ****
6. **header() és Location átirányítás**: Ez a sor átirányítja a felhasználót a bejelentkező oldalra (**admin\_bejelentkezes.php**). Ez gyakorlatilag visszairányítja a felhasználót a bejelentkező oldalra, miután sikeresen kijelentkezett.
7. **exit()**: Megállítja a script futását, miután az átirányítás megtörtént, így további felesleges kód futtatása nem következik be.

# 1.3 Adminisztrációs felületet

**Ez a PHP kód egy adminisztrációs felületet hoz létre, amely biztonságos bejelentkezést igényel:**

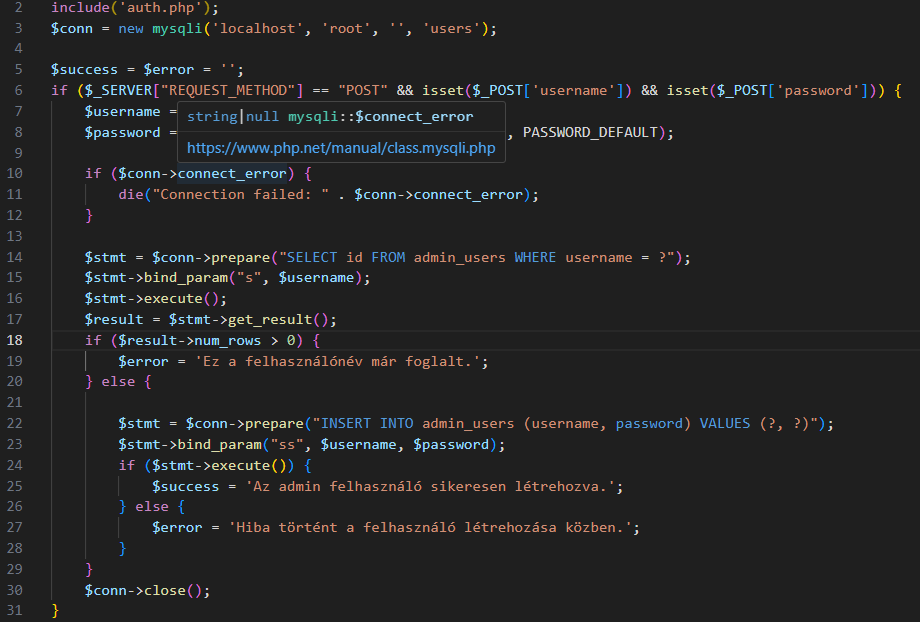


1. **Authentikációs ellenőrzés (include('auth.php'))**: Ez a sor hivatkozik egy olyan fájlra, amely tartalmazza az autentikációs ellenőrzést végző kódot. Ez valószínűleg azért van ott, hogy biztosítsa, hogy csak bejelentkezett felhasználók férjenek hozzá az adminisztrációs felülethez.
2. **POST kérés kezelése**: Ellenőrzi, hogy a kérés POST metódussal érkezett-e a szerverre. Ha igen, akkor a beérkező felhasználónév és jelszó értékeket elmenti a **$username** és **$password** változókba.
3. **Adatbázis csatlakozás létrehozása**: Az adatbázis kapcsolatot hoz létre az adatbázis szerverrel a **db\_config.php** fájlban tárolt beállítások alapján. Ezután lekérdezi az adatbázisból az adminisztrátor felhasználó adatait a megadott felhasználónév alapján.
4. **Jelszó ellenőrzése**: Ha van egyetlen találat a felhasználónevet tartalmazó rekordok között, akkor ellenőrzi a megadott jelszót a tárolt, titkosított jelszóval. Ha a megadott jelszó helyes, beállítja a munkamenet változókat a bejelentkezett állapot jelezésére, majd átirányítja a felhasználót az adminisztrációs felületre.
5. **Átirányítás a bejelentkezési oldalra**: Ha a bejelentkezés sikertelen, akkor egy hibaüzenetet állít be a **$login\_err** változóban, amit később megjeleníthet a felhasználónak.
6. **HTML kód az adminisztrációs felülethez**: Ez a rész a HTML kódot tartalmazza az adminisztrációs felület megjelenítéséhez. A Bootstrap keretrendszer segítségével történik a stílusozás, ami modern és responszív megjelenést biztosít az oldalnak.
7. **Navbar**: Az adminisztrációs felület felső sávjában található menüpontok, amelyek között lehet navigálni. Az aktív oldalt jelző osztályt a **basename($\_SERVER['PHP\_SELF'])** függvény segítségével állítja be.



# 1.4 Új adminisztrátor

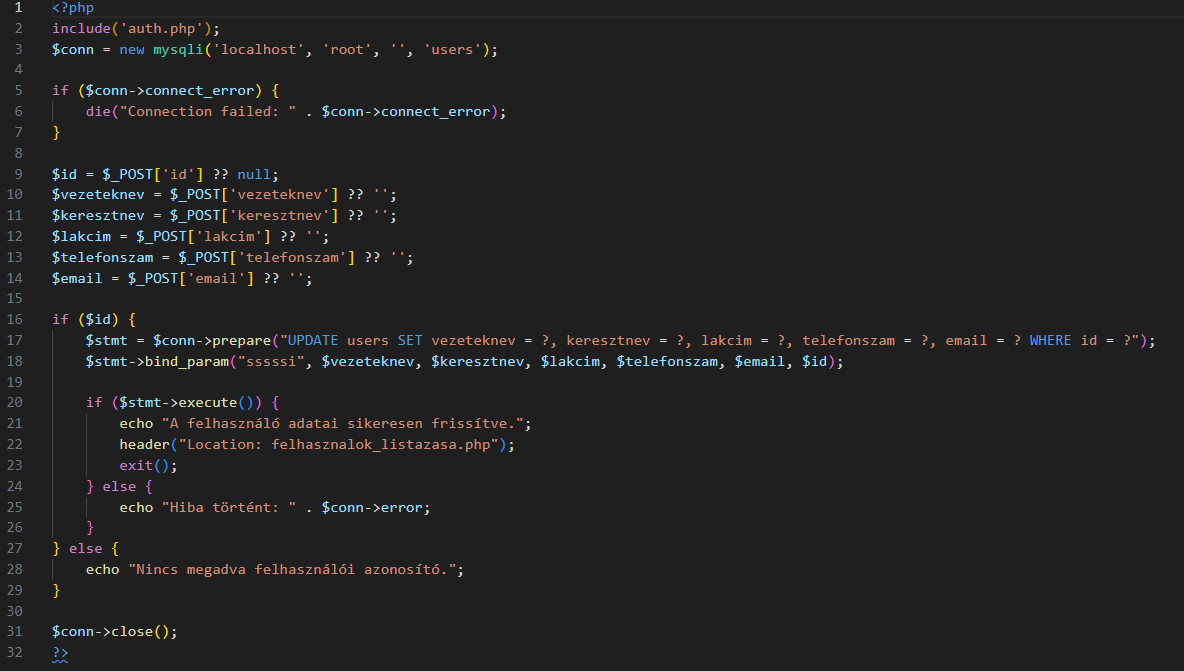
**Ez a PHP kód egy új adminisztrátor felhasználót hoz létre az adatbázisban, és egy egyszerű űrlapot jelenít meg az adminisztrációs felületen a felhasználó létrehozásához:**

****

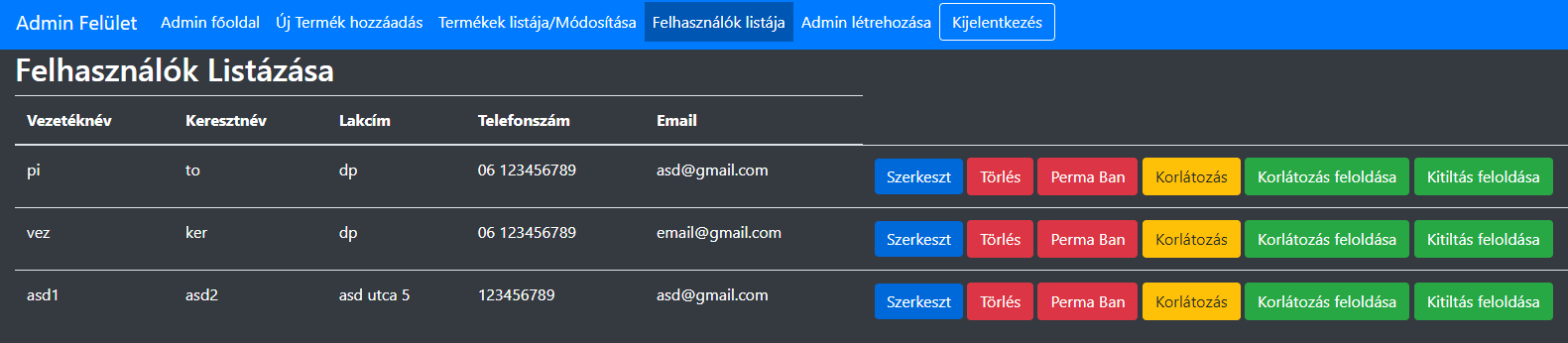
1. **Authentikációs ellenőrzés (include('auth.php'))**: Ez a sor hivatkozik egy olyan fájlra, amely tartalmazza az autentikációs ellenőrzést végző kódot. Ez biztosítja, hogy csak bejelentkezett felhasználók férjenek hozzá az adminisztrációs felülethez.
2. **Adatbázis csatlakozás létrehozása**: Az adatbázis kapcsolatot hoz létre az adatbázis szerverrel a **mysqli** osztály segítségével. Az adatbázis szerver elérési útját, felhasználónevét, jelszavát és az adatbázis nevét a megfelelő helyeken kell megadni.
3. **POST kérés kezelése és felhasználó létrehozása**: Ellenőrzi, hogy a kérés POST metódussal érkezett-e a szerverre és hogy a **username** és **password** mezők ki vannak-e töltve. Ezután ellenőrzi, hogy a megadott felhasználónév már létezik-e az adatbázisban. Ha nem létezik, akkor a megadott felhasználónevet és jelszót hozzáadja az adatbázishoz.
4. **HTML kód az új adminisztrátor létrehozásához**: Ez a rész a HTML űrlapot tartalmazza az adminisztrátor felhasználó létrehozásához. A Bootstrap keretrendszer segítségével történik a stílusozás, ami modern és responszív megjelenést biztosít az oldalnak.
5. **Siker és hibaüzenetek megjelenítése**: A sikeres vagy sikertelen művelet után megjelenik egy üzenet a felhasználónak a **$success** vagy **$error** változóban tárolt üzenet alapján.

# 1.5 Felhasználói adatok frissítése

**Ez a PHP kód felelős az adatbázisban egy felhasználó adatainak frissítéséért:**

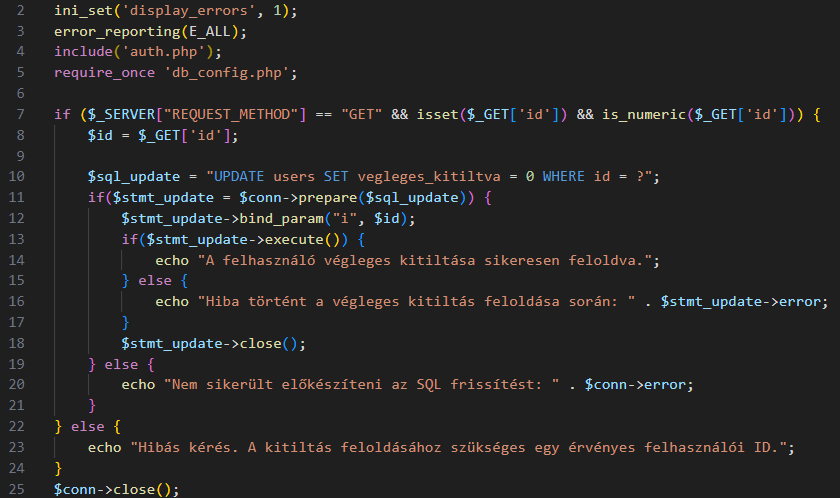
****

1. **Authentikációs ellenőrzés (include('auth.php'))**: Ez a sor hivatkozik egy olyan fájlra, amely tartalmazza az autentikációs ellenőrzést végző kódot. Ez biztosítja, hogy csak bejelentkezett felhasználók férjenek hozzá az adminisztrációs funkciókhoz.
2. **Adatbázis csatlakozás létrehozása**: Az adatbázis kapcsolatot hoz létre az adatbázis szerverrel a **mysqli** osztály segítségével. Ezután ellenőrzi, hogy sikeres volt-e a kapcsolódás. Ha nem sikerült, akkor kiírja a kapcsolódási hibát és megszakítja a script futását.
3. **POST adatok kezelése**: A beérkező POST adatokat elmenti változókba, vagy alapértelmezett értékeket állít be, ha a POST adatok nem érkeztek meg.
4. **Adatok frissítése az adatbázisban**: Ellenőrzi, hogy van-e megadva felhasználói azonosító. Ha van, akkor az adott felhasználó adatait frissíti az adatbázisban a kapott adatok alapján.
5. **Siker és hibaüzenetek megjelenítése**: Ha a frissítés sikeres, kiírja, hogy "A felhasználó adatai sikeresen frissítve." és átirányítja a felhasználót a felhasználók listázása oldalra. Ha hiba történt, kiírja az adatbázis hibáját.
6. **Adatbázis kapcsolat lezárása**: Bezárja az adatbázis kapcsolatot a **close()** metódussal.

****

# 1.6 Felhasználó végleges kitiltásának feloldása

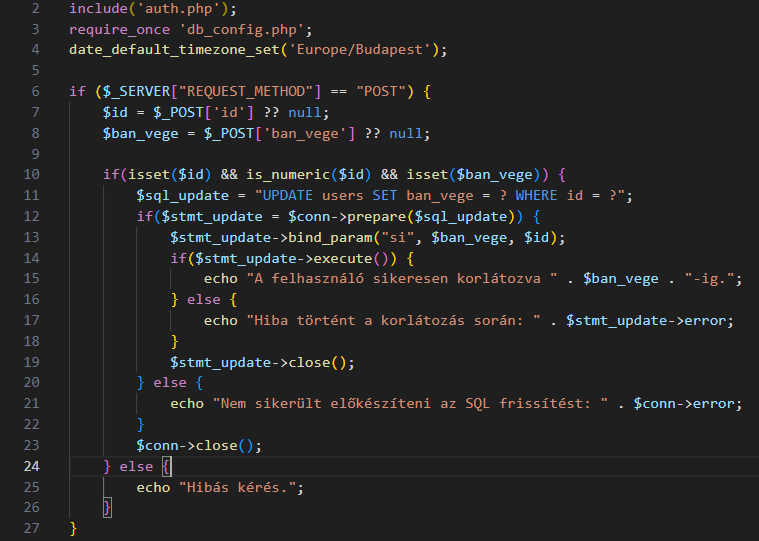
**Ez a PHP kód felelős egy felhasználó végleges kitiltásának feloldásáért az adatbázisban. Nézzük meg részleteiben:**

****

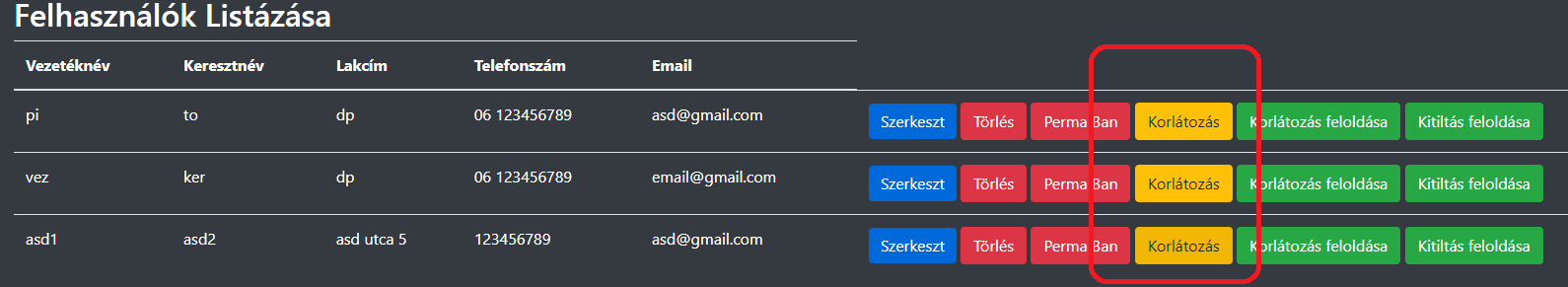
1. **Hibaüzenetek bekapcsolása (ini\_set('display\_errors', 1); error\_reporting(E\_ALL);)**: Ezek a sorok bekapcsolják és beállítják az összes hibajelzést, hogy az összes hibát és figyelmeztetést megjelenítsék, így könnyebben lehet hibákat keresni és javítani.
2. **Authentikációs ellenőrzés (include('auth.php'))**: Ez a sor hivatkozik egy olyan fájlra, amely tartalmazza az autentikációs ellenőrzést végző kódot. Ez biztosítja, hogy csak bejelentkezett felhasználók férjenek hozzá az adminisztrációs funkciókhoz.
3. **Adatbázis kapcsolat létrehozása (require\_once 'db\_config.php';)**: Ez a sor hivatkozik a **db\_config.php** fájlra, amely tartalmazza az adatbázishoz való kapcsolódáshoz szükséges beállításokat és kódokat.
4. **GET kérés kezelése és végleges kitiltás feloldása**: Ellenőrzi, hogy a kérés GET metódussal érkezett-e a szerverre, és hogy az érkező kérés tartalmazza-e a **id** paramétert, amely egy érvényes felhasználói azonosítót jelöl. Ha ezek a feltételek teljesülnek, akkor előkészíti az SQL lekérdezést a végleges kitiltás feloldására és végrehajtja azt az adatbázison keresztül. A sikeres végrehajtás után kiírja a sikerüzenetet, ha hiba történik, akkor pedig a hibaüzenetet.
5. **Hibás kérés kezelése**: Ha a GET kérés nem tartalmaz érvényes felhasználói azonosítót, akkor kiírja a hibaüzenetet.
6. **Adatbázis kapcsolat lezárása**: Bezárja az adatbázis kapcsolatot a **close()** metódussal.
7. 

# 1.7 Felhasználó korlátozása

**Ez a kód egy felhasználó korlátozását teszi lehetővé az adatbázisban, és egy űrlapot jelenít meg az adminisztrációs felületen a korlátozás beállításához:**



1. **Authentikációs ellenőrzés (include('auth.php'))**: Ez a sor hivatkozik egy olyan fájlra, amely tartalmazza az autentikációs ellenőrzést végző kódot. Ez biztosítja, hogy csak bejelentkezett felhasználók férjenek hozzá az adminisztrációs funkciókhoz.
2. **Adatbázis kapcsolat létrehozása (require\_once 'db\_config.php';)**: Ez a sor hivatkozik a **db\_config.php** fájlra, amely tartalmazza az adatbázishoz való kapcsolódáshoz szükséges beállításokat és kódokat.
3. **POST adatok kezelése és korlátozás beállítása**: Ellenőrzi, hogy a kérés POST metódussal érkezett-e a szerverre, és hogy a POST adatok tartalmazzák-e a szükséges **id** és **ban vege** paramétereket. Ha minden adat rendelkezésre áll, előkészíti az SQL lekérdezést a felhasználó korlátozásának beállítására az adatbázisban, majd végrehajtja azt. Sikeres végrehajtás esetén kiírja a sikerüzenetet, ellenkező esetben a hibaüzenetet.
4. **Hibás kérés kezelése**: Ha a POST kérés nem tartalmaz minden szükséges adatot, kiírja a hibaüzenetet.
5. **Adatbázis kapcsolat lezárása**: Bezárja az adatbázis kapcsolatot a **close()** metódussal.
6. **HTML űrlap a korlátozás beállításához**: Ez a rész a HTML űrlapot tartalmazza a felhasználó korlátozásának beállításához. A felhasználó az űrlapon kiválaszthatja a korlátozás végét, majd elküldheti az űrlapot a beállítás végrehajtásához.

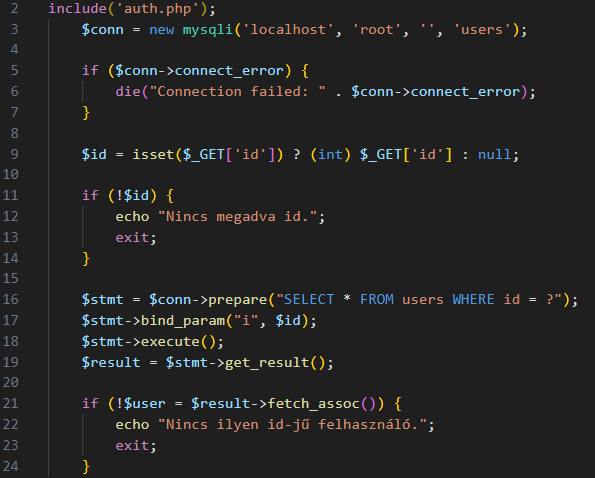


Ez a kód lehetővé teszi egy felhasználó korlátozását az adatbázisban, és egy felhasználóbarát űrlapot biztosít a korlátozás beállításához az adminisztrációs felületen.

****

# 1.8 Felhasználó szerkesztése

**Ez a kód egy felhasználó szerkesztésére szolgáló űrlapot jelenít meg az adminisztrációs felületen:**

****

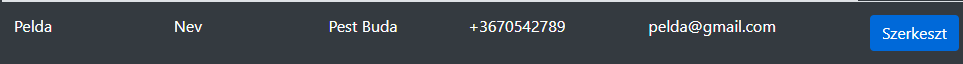
1. **Authentikációs ellenőrzés (include('auth.php'))**: Ez a sor hivatkozik egy olyan fájlra, amely tartalmazza az autentikációs ellenőrzést végző kódot. Ez biztosítja, hogy csak bejelentkezett felhasználók férjenek hozzá az adminisztrációs funkciókhoz.
2. **Adatbázis kapcsolat létrehozása**: A **$conn** változó egy adatbázis kapcsolatot hoz létre az adatbázis szerverrel. Ha a kapcsolat létrejön, akkor folytatódik a kód futása, ha nem, akkor kiírja a kapcsolódási hibát.
3. **GET adatok kezelése**: Az **$id** változóba eltároljuk a **$\_GET['id']** értékét, ha az létezik és szám típusú. Ha nincs megadva **id**, akkor kiírja, hogy nincs megadva id, és kilép a kódból.
4. **Felhasználó lekérdezése**: Létrehoz egy SQL lekérdezést a **users** táblából az adott **id** alapján. Ezután végrehajtja a lekérdezést, és az eredményt eltárolja a **$user** változóban.
5. **Adatok megjelenítése az űrlapon**: Az űrlapon megjelenítjük a felhasználó adatait az input mezőkben. Az alapértelmezett értékek a **$user** tömbből származnak, amelyet az adatbázisból kaptunk.
6. **HTML űrlap a felhasználó adatainak szerkesztéséhez**: Ez a rész tartalmazza az űrlapot, ahol a felhasználó adatait szerkeszthetjük. Az input mezőkben megjelenítjük a felhasználó jelenlegi adatait, és azokat lehet szerkeszteni.
7. 

# 1.9 Admin adatok frissítése

**Ez a PHP kód az adatbázisban egy felhasználó adatainak frissítését végzi el az adminisztrációs felületen keresztül:**

****

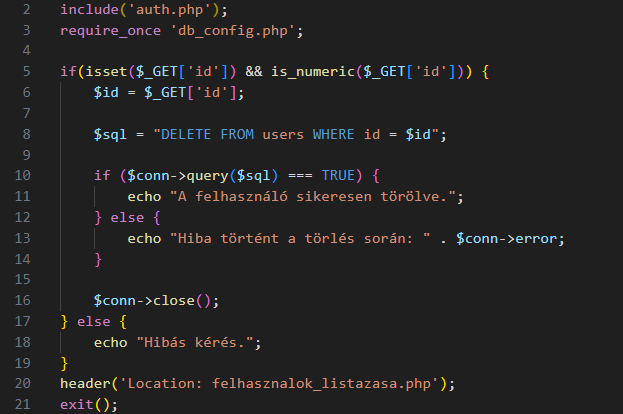
1. **Authentikációs Ellenőrzés**: Az első sor az **auth.php** fájl beillesztését végzi, amelyben az autentikációs ellenőrzés kódja található. Ez biztosítja, hogy csak bejelentkezett felhasználók férhessenek hozzá az adminisztrációs funkciókhoz.
2. **Adatbázis Kapcsolat Létrehozása**: Ezután a kód létrehoz egy kapcsolatot az adatbázissal a **mysqli** osztály segítségével. Ha a kapcsolat létrejött, a kód folytatódik, ha nem, akkor kiírja a kapcsolódási hibát és megszakítja a futást.
3. **POST Adatok Feldolgozása**: Az **$\_POST** tömbből kinyeri az adatokat, például az azonosítót, vezetéknevet, keresztnevet, lakcímet, telefonszámot és e-mail címet. Ezeket az adatokat az űrlapról kapja meg, amikor azokat elküldik.
4. **Adatok Frissítése az Adatbázisban**: Ezután egy **UPDATE** SQL lekérdezést hajt végre az adatbázison, amely frissíti a megadott azonosítójú felhasználó adatait. A **bind\_param** metódus segítségével paraméterezett lekérdezést készít, hogy elkerülje a SQL injection támadásokat.
5. **Siker/Fiaskó Üzenetek Megjelenítése**: Ha a lekérdezés sikeresen végrehajtódott, akkor kiírja a "A felhasználó adatai sikeresen frissítve." üzenetet, és átirányítja a felhasználót a **felhasznalok\_listazasa.php** oldalra. Ha hiba történt, akkor kiírja a hibaüzenetet.
6. **Adatbázis Kapcsolat Bezárása**: Végül bezárja az adatbázis kapcsolatot a **close** metódussal, hogy ne terhelje tovább a szerver erőforrásait.

****

Ez a kód hatékonyan frissíti a felhasználó adatait az adatbázisban az adminisztrációs felületen keresztül, és kezeli az esetleges hibákat is.

Az űrlap teteje

# 1.10 Felhasználó Törlése

Ez a PHP szkript lehetővé teszi felhasználók törlését az adatbázisból az azonosítójuk alapján:

#### **Használati Utasítások**

1. **Bejelentkezés**: Győződjön meg róla, hogy bejelentkezett a rendszerbe és rendelkezik megfelelő jogosultságokkal a felhasználók törléséhez.
2. **Kérés Küldése**: Nyissa meg a böngészőt, és adja meg a szkript URL-jét a következő formátumban: **http://weboldal.hu/torles.php?id=[felhasználó azonosítója]**. A **[felhasználó azonosítója]** helyére írja be a törlendő felhasználó egyedi azonosítóját az adatbázisban.
3. **Megerősítés**: Nyomja meg az "Enter" billentyűt a kérés elküldéséhez.

#### Figyelmeztetések

* **Azonosító Helyesbítése**: Ellenőrizze, hogy a megadott azonosítóval rendelkező felhasználó valóban törölhető-e, és nincs-e kapcsolatban más adatokkal, amelyeket meg kell őrizni.

# 1.11 Felhasználók Listázása

Ez a PHP szkript lehetővé teszi a rendszerben tárolt felhasználók listázását, és lehetőséget biztosít a felhasználókkal kapcsolatos műveletek elvégzésére, például szerkesztésre, törlésre, korlátozásra és kitiltásra.



#### Használati Utasítások

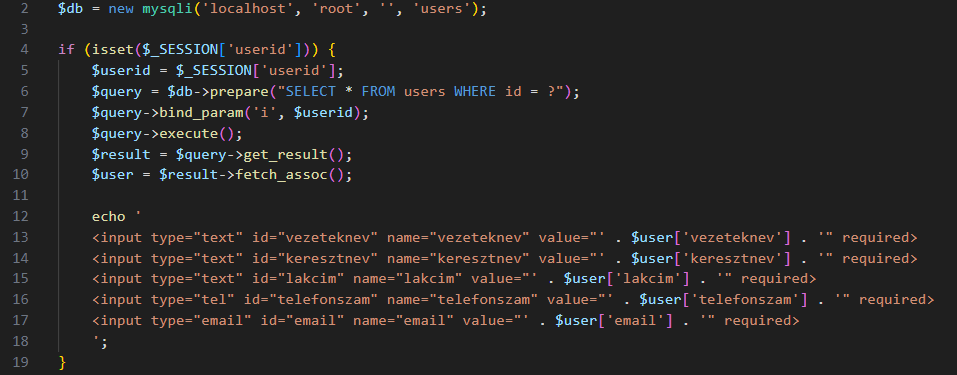
1. **Bejelentkezés**: Győződjön meg róla, hogy bejelentkezett a rendszerbe, és rendelkezik a megfelelő jogosultságokkal a felhasználók kezeléséhez.
2. **Felhasználók Listázása**: Nyissa meg a böngészőt, és adja meg a szkript URL-jét a következő formátumban: **http://weboldal.hu/felhasznalok\_listazasa.php**.
3. **Felhasználók Megtekintése**: A felhasználók listája megjelenik a weboldalon táblázatos formában. A táblázat minden sorában egy felhasználó adatai találhatók, beleértve a vezetéknevet, keresztnevet, lakcímet, telefonszámot és e-mail címet.
4. **Felhasználókkal Kapcsolatos Műveletek**: Minden felhasználó sorának végén különböző gombok találhatók, amelyek lehetővé teszik a felhasználókkal kapcsolatos műveletek elvégzését:
   * **Szerkesztés**: A "Szerkeszt" gombra kattintva a felhasználó adatait szerkesztheti.
   * **Törlés**: A "Törlés" gombra kattintva a felhasználót törölheti az adatbázisból. A törlés előtt megjelenik egy megerősítő párbeszédablak.
   * **Perma Ban**: A "Perma Ban" gombra kattintva a felhasználót véglegesen kitilthatja az adatbázisból. A kitiltás előtt megjelenik egy megerősítő párbeszédablak.
   * **Korlátozás**: A "Korlátozás" gombra kattintva korlátozhatja a felhasználó hozzáférését bizonyos funkciókhoz.
   * **Korlátozás Feloldása**: A "Korlátozás feloldása" gombra kattintva feloldhatja a felhasználó korlátozását. A feloldás előtt megjelenik egy megerősítő párbeszédablak.
   * **Kitiltás Feloldása**: A "Kitiltás feloldása" gombra kattintva feloldhatja a felhasználó kitiltását. A feloldás előtt megjelenik egy megerősítő párbeszédablak.
5. **Kijelentkezés**: Ha végezte a műveleteket, kattintson a "Kijelentkezés" gombra a biztonságos kijelentkezéshez.

#### Figyelmeztetések

* **Adatvesztés Veszélye**: A "Törlés" és "Perma Ban" műveletek véglegesen eltávolítják a felhasználót az adatbázisból, és nem lehet visszavonni. Győződjön meg róla, hogy ezeket a műveleteket csak akkor hajtja végre, ha biztos benne, hogy szükséges.

# 1.12 Fizetés és Szállítás Űrlap

Ez a PHP szkript felelős a felhasználóknak a szállítási és fizetési információk megadásáért egy webshopban.

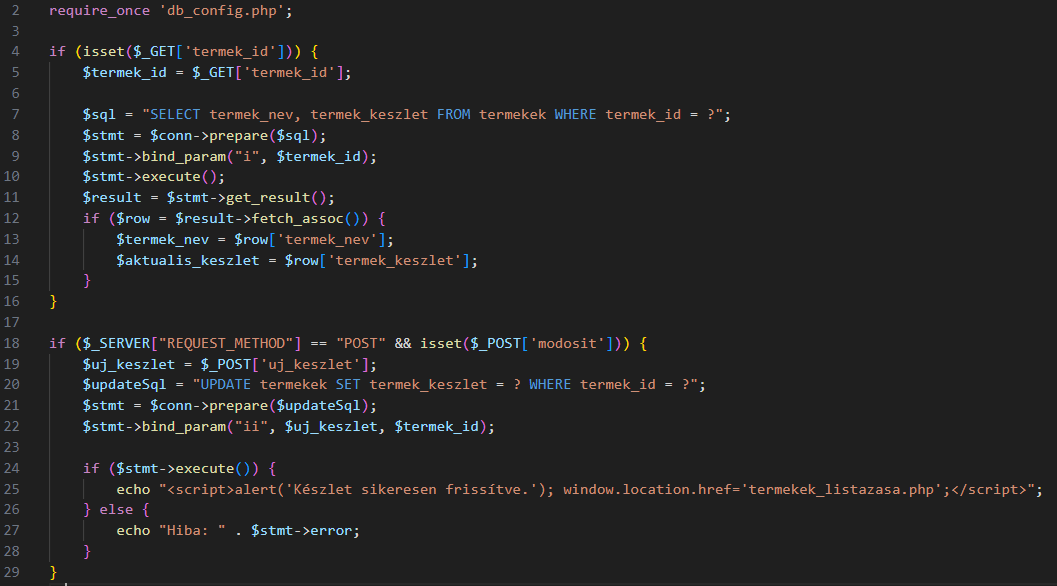


#### Használati Utasítások

1. **Bejelentkezés**: Amennyiben be van jelentkezve a rendszerbe, a szükséges adatok részben előre kitöltődnek a felhasználói adatok alapján.
2. **Szállítási és Fizetési Adatok Megadása**: Töltse ki a szállítási és fizetési információkhoz szükséges mezőket az űrlapon. Szükséges mezők a következők: vezetéknév, keresztnév, e-mail cím, telefonszám, adószám, város, utca, irányítószám, szállítási és fizetési módszer választása.
3. **Kosár Tartalmának Megtekintése**: A jobb oldalon megjelenítésre kerül a kosár tartalma, amely tartalmazza a kiválasztott termékeket, azok darabszámát és árát.
4. **Összesítés és Továbblépés**: Az összesítésben látható a kosárban található termékek végösszege. Ha minden szükséges adatot megadott, kattintson a "Tovább a fizetési oldalra" gombra a további lépésekhez.

# 1.13 Készlet Módosítása Űrlap

Ez a PHP szkript lehetővé teszi a felhasználók számára egy termék készletének módosítását egy webáruházban.



#### Használati Utasítások

1. **Termék Kiválasztása**: Amennyiben a termék készletét szeretné módosítani, először ki kell választania a megfelelő terméket. Ehhez használja a "Termék ID" paramétert a URL-ben, például: http://weboldal.hu/keszlet\_modositasa.php?termek\_id=[termék\_id], ahol [termék\_id] a kívánt termék azonosítója az adatbázisban.
2. **Új Készlet Megadása**: Töltse ki az "Új Készlet" mezőt a kívánt mennyiségű készlet megadásához.
3. **Módosítás Végrehajtása**: Kattintson a "Módosít" gombra a készlet módosításának végrehajtásához.

#### Figyelmeztetések

* **Adatellenőrzés**: Győződjön meg róla, hogy a megadott készlet mennyiség érvényes és pozitív egész szám.
* **Adatbiztonság**: Mindig ellenőrizze, hogy a termék azonosítója és a készlet mennyisége helyes és megfelelően módosul-e az adatbázisban.
* **Megerősítés**: Győződjön meg arról, hogy a módosítás végrehajtását megelőzően megfelelően beállította az új készletmennyiséget.

# 1.14 Kosár Tartalom

### Cél

Az oldal célja, hogy megjelenítse a felhasználó kosarában lévő termékeket, valamint lehetőséget biztosítson a felhasználónak a fizetési folyamat elindítására.



### Technikai Implementáció

* Az oldal HTML, CSS és JavaScript nyelveken készült.
* Az HTML struktúra megfelelően tagolt, és a **<meta>** címkék beállítása a megfelelő karakterkódolást és nézet méretet biztosítja.
* A CSS stílusokat a **<style>** címkékben az oldal tetején találhatóak.
* A JavaScript funkciók az oldal alján találhatóak, az elemek megjelenítése és a fizetési folyamat kezelése céljából.

### Főbb Komponensek

1. **Kosár Tartalom Megjelenítése:**
   * Az oldal betöltésekor azonnal megjeleníti a felhasználó kosarában lévő termékeket vagy egy üzenetet, ha a kosár üres.
   * Ha a kosár nem üres, akkor minden termék nevét és mennyiségét listázza ki.
2. **Fizetési Folyamat:**
   * A "Tovább a fizetéshez" gombra kattintva egy egyszerű üzenetablak jelenik meg, ami jelzi a fizetési folyamat megkezdését. Ebben az implementációban a fizetési folyamat nem történik meg az oldalon, csak egy üzenet jelenik meg.

### Felhasznált Technológiák

* **HTML:** Az oldal struktúrájának és tartalmának leírására.
* **CSS:** Az oldal megjelenésének formázására.
* **JavaScript:** Az interaktivitás biztosítása és a felhasználói műveletek kezelése céljából.
* **Viewport Meta Tag:** A mobilbarát elrendezés és a megfelelő nézet méret beállítása.

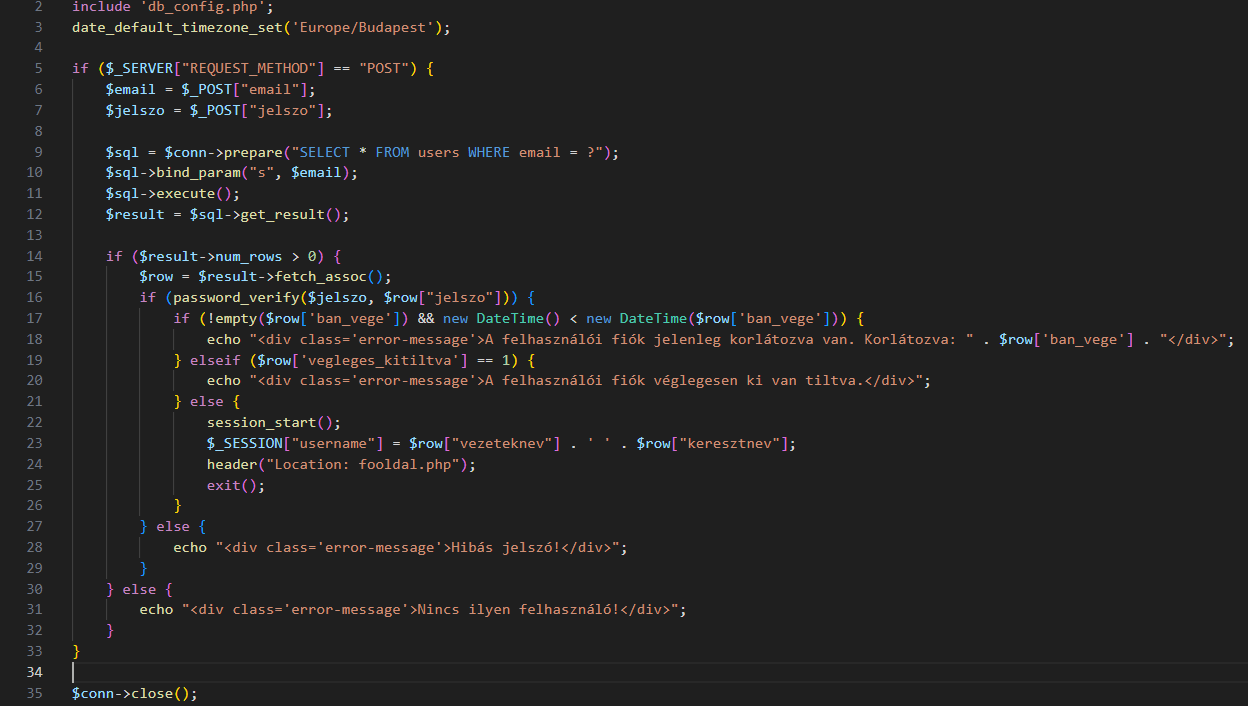
### Fejlesztői Megjegyzés

* Az oldal jelenleg csak egy egyszerű funkcióval rendelkezik, de könnyen bővíthető más funkcionalitásokkal, például a termékek törlésével vagy módosításával a kosárból.
* A fizetési folyamatot megvalósító JavaScript funkciók és a fizetési API integrációja további fejlesztési lehetőségeket kínál a jövőben.

# 1.15 Bejelentkező Oldal

### Cél

Az oldal célja, hogy lehetővé tegye a felhasználók számára a bejelentkezést. Ellenőrzi a felhasználó által megadott adatok helyességét, és hibaüzeneteket jelenít meg, ha szükséges.



### Technikai Implementáció

* Az oldal PHP és HTML nyelveken készült.
* Az adatbáziskapcsolatot a **db\_config.php** fájl tartalmazza.
* Az adatok ellenőrzése és a bejelentkezési folyamat a PHP kód segítségével történik.
* Az oldal megjelenése és stílusa a CSS stíluslapban van definiálva.

### Főbb Komponensek

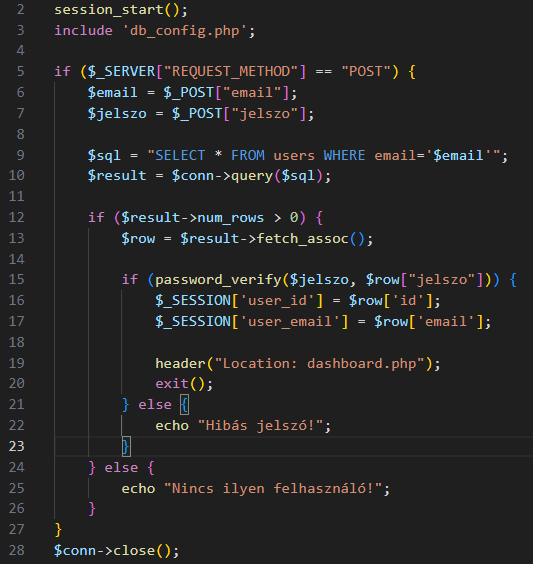
1. **Bejelentkezési Űrlap:**
   * Az űrlap tartalmazza az email cím és jelszó mezőket, valamint egy "Bejelentkezés" gombot.
   * Ha a felhasználó hibás adatokat ad meg, hibaüzenet jelenik meg az űrlap alatt.
2. **Bejelentkezési Folyamat:**
   * A bejelentkezési adatokat ellenőrzi az adatbázisban tárolt adatokkal.
   * Ha a bejelentkezés sikeres, az oldal átirányít a főoldalra (**fooldal.php**).
3. **Admin Bejelentkezés Gomb:**
   * Az "Admin Bejelentkezés" gomb egy másik oldalra irányít, ahol az adminisztrátorok be tudnak jelentkezni.

### Fejlesztői Megjegyzés

* Az oldal egyszerű és felhasználóbarát felületet kínál a bejelentkezéshez.
* A kódot könnyen bővíthetővé lehet tenni további funkciókkal, például elfelejtett jelszó visszaállításával vagy többfaktoros azonosítással.

# 1.16 Felhasználó Bejelentkezés

Ez a PHP szkript az alkalmazás felhasználóinak bejelentkezését végzi:



#### Működés

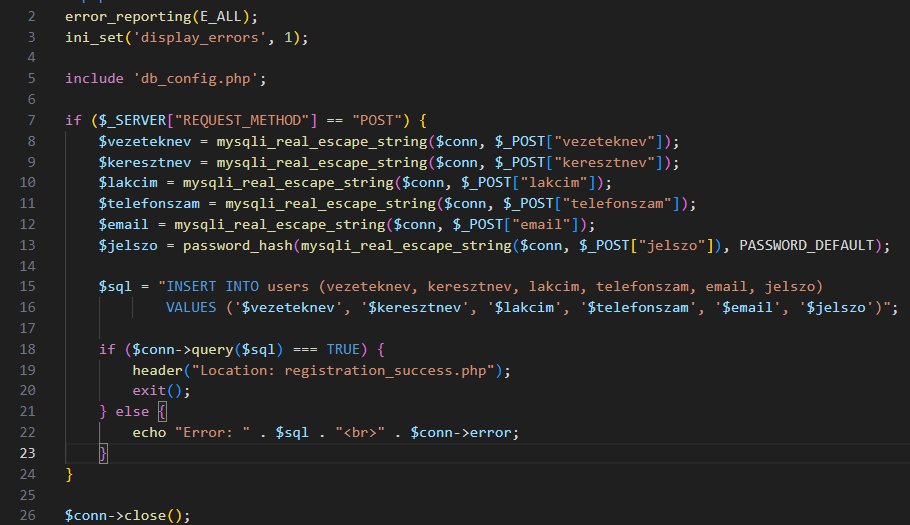
1. **Munkamenet inicializálása:** A szkript először inicializálja vagy folytatja a munkamenetet, hogy tárolni tudja a bejelentkezett felhasználó adatait.
2. **Adatbázis kapcsolat:** A szkript csatlakozik az adatbázishoz a **db\_config.php** konfigurációs fájl segítségével.
3. **Űrlapkezelés:** Ellenőrzi, hogy a beérkező kérés POST metódussal érkezett-e. Ha igen, beolvassa és sanitizálja az űrlapról érkező e-mail cím és jelszó adatokat.
4. **Felhasználó ellenőrzése:** A szkript lekérdezi az adatbázisból a megadott e-mail címmel rendelkező felhasználót. Ha talál eredményt, ellenőrzi a megadott jelszót a tárolt jelszóval.
5. **Bejelentkezés kezelése:** Ha a jelszó helyes, beállítja a felhasználó munkameneti változóit, majd átirányítja a felhasználót a vezérlőpulthoz (**dashboard.php**). Ha a bejelentkezés sikertelen, hibaüzenetet jelenít meg.
6. **Adatbázis kapcsolat lezárása:** A szkript lezárja az adatbáziskapcsolatot.

#### Fájlok

* **db\_config.php:** Az adatbáziskapcsolat konfigurációs beállításait tartalmazó fájl.
* **dashboard.php:** A felhasználó vezérlőpultját kezelő fájl.

# 1.17 Felhasználó Regisztráció

Ez a PHP szkript a felhasználók regisztrációját végzi egy webalkalmazásban.



#### Működés

1. **Hibaüzenetek bekapcsolása:** Az **error\_reporting** és **ini\_set** függvények beállításával az összes hibaüzenet megjelenítése engedélyezett, ami segít a hibakeresésben.
2. **Adatbázis kapcsolat:** A szkript csatlakozik az adatbázishoz a **db\_config.php** konfigurációs fájl segítségével.
3. **Űrlapkezelés:** Ellenőrzi, hogy a beérkező kérés POST metódussal érkezett-e. Ha igen, beolvassa és sanitizálja az űrlapról érkező adatokat.
4. **Felhasználói adatok beszúrása:** A szkript beszúrja a felhasználó által megadott adatokat az adatbázisba. A jelszó titkosításra kerül a **password\_hash** függvény segítségével.
5. **Beszúrás eredményének kezelése:** Ha a beszúrás sikeres, az űrlapról átirányítja a felhasználót a sikeres regisztrációt jelző oldalra (**registration\_success.php**). Ha a beszúrás sikertelen, hibaüzenetet jelenít meg.
6. **Adatbázis kapcsolat lezárása:** A szkript lezárja az adatbáziskapcsolatot.

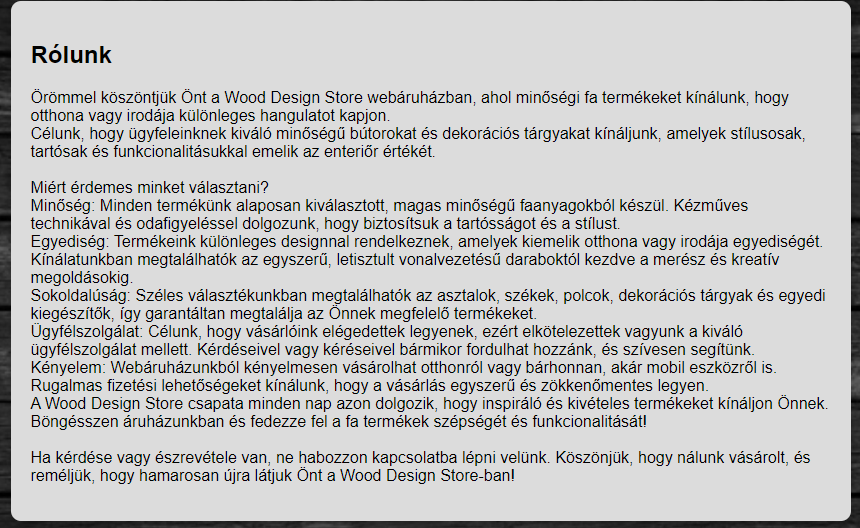
#### Fájlok

* **db\_config.php:** Az adatbáziskapcsolat konfigurációs beállításait tartalmazó fájl.
* **registration\_success.php:** A sikeres regisztrációt jelző oldal.

#### Megjegyzés

A kódban fontos megjegyezni a felhasználó által megadott adatok sanitizálását, hogy megelőzzük a SQL injection támadásokat. A jelszavak titkosítása pedig az adatbiztonság fontos eleme.

# 1.18 Rólunk Oldal



A Rólunk oldal a Wood Design Store webáruház része, amelynek célja bemutatni a vállalatot, annak tevékenységét és értékeit a látogatók számára.

## **Oldalstruktúra**

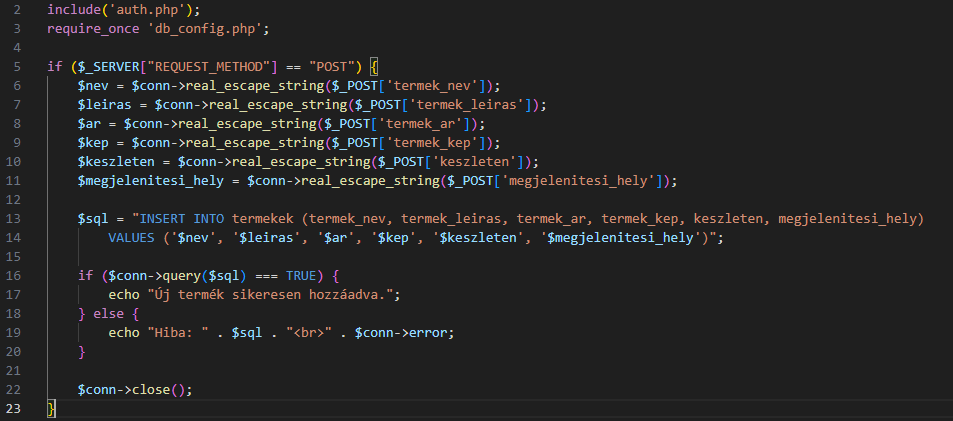
Az oldal struktúrája a következő elemekből áll:

* **HTML Fájl:** A **rolunk.php** nevű HTML fájl tartalmazza az oldal kódját és szerkezetét.
* **Fő Címe:** Az oldal fő címe "Rólunk".
* **Navigációs Menü:** A navigációs menü a weboldal többi részére mutató linkeket tartalmaz.
* **Tartalomtéma:** A tartalom részletesen bemutatja a Wood Design Store vállalatot, annak céljait, termékeinek jellemzőit és az ügyfélszolgálatot.

## **Technikai Részletek**

* **Dokumentum Típusa:** Az oldal HTML5 dokumentumtípust használ.
* **Nyelvi Támogatás:** Az oldal a magyar nyelvet használja.
* **Karakterkódolás:** A karakterkódolás UTF-8.
* **Megjelenítési Beállítások:** A viewport beállítása lehetővé teszi az oldal megfelelő megjelenítését különböző eszközökön.
* **Stíluslapok:** A megjelenésért felelős stíluslapok külső forrásból történő betöltése a **https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css** címről.
* **Stílusok:** Az oldal stílusai a **<style>** elemen belül vannak definiálva.

# 1.19 Termék hozzáadása űrlap PHP kódja:



### **Funkciók**

* **Adatok Felvitele:** Az űrlap lehetővé teszi a termék nevének, leírásának, árának, képének, készletmennyiségének és megjelenítési helyének megadását.
* **Adatok Ellenőrzése:** A beérkező adatokat megfelelően ellenőrzi és validálja az űrlap, hogy biztosítsa az adatok integritását és hibamentes feldolgozását.
* **Adatok Tisztítása:** A beérkező adatokat megfelelően tisztítja és formázza a PHP kód, hogy elkerülje a biztonsági problémákat, például az SQL injection támadásokat.
* **Adatok Beszúrása:** Az űrlap elküldésekor a PHP kód beszúrja az új termék adatait az adatbázisba, biztosítva ezzel a frissített készletet és az új termék megjelenését a webáruházban.
* **Visszajelzés:** A felhasználók visszajelzést kapnak az űrlap elküldését követően arról, hogy a művelet sikeres volt-e vagy sikertelen.

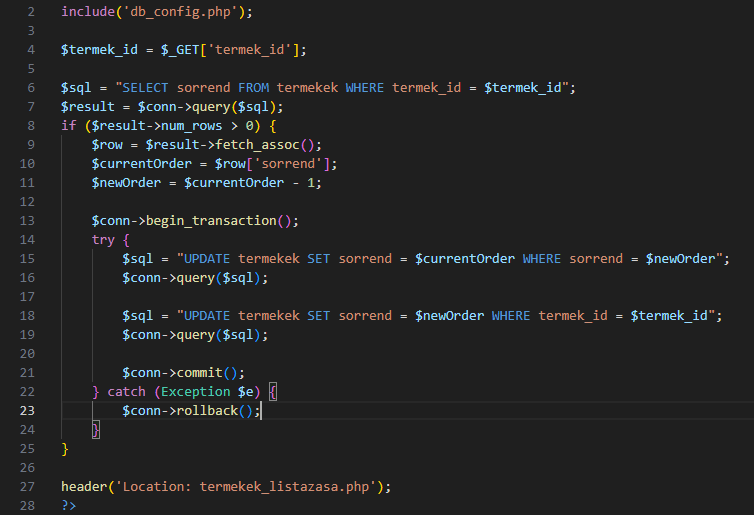
### **Adatmezők**

Az űrlap az alábbi adatmezőket tartalmazza a termék részletes adatainak megadásához:

* **Termék Név:** A termék nevének megadása.
* **Leírás:** Részletes leírás a termékről.
* **Ár:** A termék ára.
* **Kép URL:** A termék képének URL-je.
* **Készleten:** A rendelkezésre álló készlet mennyisége.
* **Megjelenítési Hely:** A termék megjelenítési helyének kiválasztása a webáruházban.

1.20 Termékek sorrendjének módosítása

Ez a PHP kód egy olyan funkciót valósít meg, amely lehetővé teszi a termékek sorrendjének módosítását az adatbázisban. A kód az alábbiak szerint működik:

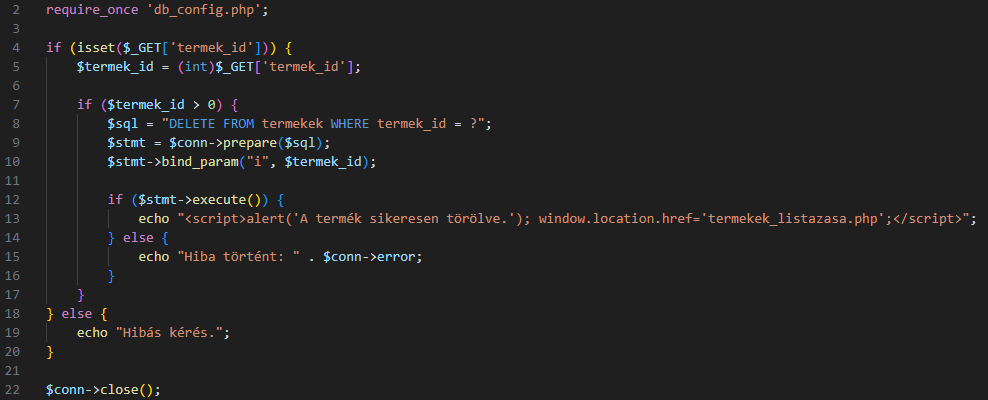


1. Adatbázis csatlakozás: Először is, a kód importálja az adatbázis konfigurációs fájlt (**db\_config.php**), hogy kapcsolódjon az adatbázishoz.
2. URL paraméter beolvasása: A kód beolvassa a **termek\_id** paramétert az URL-ből, amely azon termék azonosítóját tartalmazza, amelynek a sorrendjét módosítani szeretnénk.
3. Sorrend frissítése: A kód lekérdezi az adatbázisból az adott termékhez tartozó sorrendet. Ezután megpróbálja csökkenteni a termék sorrendjét egyel az előző sorrend alapján. A tranzakció biztosítja, hogy mindkét frissítés csak akkor kerüljön végrehajtásra, ha mindkettő sikeres.
4. Hiba kezelése: Ha valamilyen hiba történik a tranzakció végrehajtása közben, akkor a tranzakciót visszavonja (rollback), hogy ne legyenek inkonzisztens adatok az adatbázisban.
5. Átirányítás: Végül a kód átirányítja a felhasználót a termékek listázása oldalra (**termekek\_listazasa.php**), hogy a felhasználó láthassa a frissített sorrenddel rendelkező termékeket.

Ez a funkció lehetővé teszi a termékek sorrendjének egyszerű módosítását az adminisztrátorok számára, ami segíthet a webáruházban való tartalom szervezésében és megjelenítésében.

1.21 Termékek törlése

Ez a PHP kód egy olyan funkciót valósít meg, amely lehetővé teszi egy adott termék törlését az adatbázisból. A kód a következő lépéseket hajtja végre:

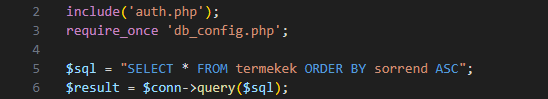


1. Adatbázis csatlakozás: Először is, a kód importálja az adatbázis konfigurációs fájlt (**db\_config.php**), hogy kapcsolódjon az adatbázishoz.
2. Paraméterek beolvasása: A kód ellenőrzi, hogy a **termek\_id** paraméter meg van-e adva az URL-ben. Ha igen, akkor az értékét átalakítja egész számmá.
3. Termék törlése: Ha a **termek\_id** érvényes és nagyobb nullánál, akkor a kód előkészíti és végrehajtja a törlési műveletet az adatbázisban. Ehhez először SQL lekérdezést készít a **DELETE** utasítással, majd az értéket paraméterként köti a lekérdezéshez.
4. Visszajelzés: Ha a törlés sikeres, akkor üzenet jelenik meg, amely értesíti a felhasználót a sikerességről, és átirányítja a felhasználót a termékek listázása oldalra. Ellenkező esetben hibaüzenet jelenik meg.
5. Adatbázis kapcsolat lezárása: Végül a kód bezárja az adatbázis kapcsolatot.

Ez a funkció lehetővé teszi az adminisztrátorok számára, hogy könnyen és hatékonyan töröljék a termékeket az adatbázisból, ami segíthet a webáruház tartalmának frissítésében és karbantartásában.

1.22 Termékek listázása

Ez a PHP kód egy olyan funkciót valósít meg, amely lehetővé teszi az aktív megjelenő termékek listázását és módosítását az adminisztrátorok számára. A kód a következő lépéseket hajtja végre:



1. Adatbázis csatlakozás: Először is, a kód importálja az adatbázis konfigurációs fájlt (**db\_config.php**), hogy kapcsolódjon az adatbázishoz.
2. Termékek lekérdezése: A kód lekéri az összes terméket az adatbázisból a **termekek** táblából, rendezve a **sorrend** mező alapján növekvő sorrendben.
3. HTML kimenet: A kód HTML kódot generál a termékek listázásához és megjelenítéséhez a weboldalon. Minden termék adatait táblázat formájában jeleníti meg, beleértve a termék nevét, leírását, árát, képét és egy sor műveletet, amelyek lehetővé teszik a termék sorrendjének módosítását, módosítását vagy törlését.
4. Műveletek hivatkozásai: A műveleteket végző gombok hivatkozásokat tartalmaznak a szükséges PHP fájlokra (**termek\_sorrend\_le.php**, **termek\_sorrend\_fel.php**, **termek\_modositasa.php**, **termek\_torlese.php**), amelyek felelősek a termékek sorrendjének módosításáért, módosításáért vagy törléséért. Ezen kívül a törléshez egy megerősítési párbeszédablakot is megjelenít a felhasználónak.
5. Stílusok és formázás: A kód CSS stílusokat tartalmaz a megjelenítés formázásához, például a háttérszín, a betűszín, a táblázat formázása és a gombok stílusának meghatározásához.

Ez a funkció lehetővé teszi az adminisztrátorok számára, hogy könnyen kezeljék és módosítsák a termékek listáját a webáruházban, biztosítva ezzel a felhasználóknak a friss és aktuális termékkínálatot.

2. Adatszerkezet

## **Bevezetés**

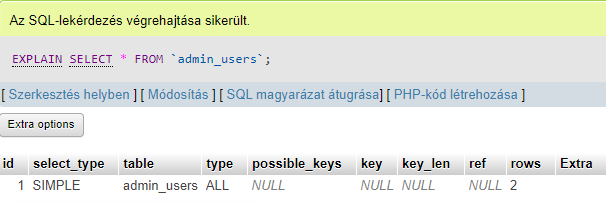
Ez a részletes dokumentáció bemutatja a PHP nyelven írt MySQL adatbázisból történő felhasználói adatok lekérésének folyamatát. A kód a felhasználói munkamenet során tárolt email cím alapján keres egy felhasználót az adatbázisban, majd visszaadja az ehhez a felhasználóhoz tartozó adatokat.

## **Rendszerkövetelmények**

* PHP 5.6 vagy újabb
* MySQL szerver



## **Folyamat**

* **Felhasználói munkamenet ellenőrzése**: Először is meg kell győződni arról, hogy a felhasználó be van-e jelentkezve, és ha igen, akkor van-e tárolt email címe a munkamenetben. Ez biztosítja, hogy csak bejelentkezett felhasználók adatait kérjük le.
* **Adatbáziskapcsolat felépítése**: A kódban először fel kell építeni egy kapcsolatot az adatbázissal. Ez általában egy előre definiált kapcsolatobjektumon (**$conn**) keresztül történik.
* **SQL lekérdezés előkészítése**: A felhasználóhoz tartozó adatok lekéréséhez SQL lekérdezést kell készíteni. A lekérdezésben paraméterként a felhasználó email címét használjuk.
* **SQL lekérdezés végrehajtása**: A lekérdezést először előkészítjük (**prepare()** metódus), majd a paramétereket hozzárendeljük (**bind\_param()** metódus). Ezután végrehajtjuk a lekérdezést (**execute()** metódus).
* 
* **Eredményfeldolgozás**: A lekérdezés végrehajtása után megkapjuk az eredményt. Ha van találat (a felhasználó email címe alapján), akkor ezt feldolgozzuk és eltároljuk a **$userdata** változóban.

## **Változók és Objektumok**

* **$\_SESSION['email']**: A felhasználó munkamenetben tárolt email címe.
* **$conn**: Az adatbáziskapcsolatot reprezentáló objektum.
* **$email**: A felhasználó email címe, amely alapján keresünk az adatbázisban.
* **$sql**: Az SQL lekérdezés string formában, amely a felhasználó adatainak lekérésére szolgál.
* **$stmt**: Az előkészített SQL állítás (statement) objektuma.
* **$result**: Az SQL lekérdezés végrehajtása után kapott eredmény objektuma.
* **$userdata**: A felhasználó adatait tartalmazó asszociatív tömb, ha van találat.

## **Hibakezelés**

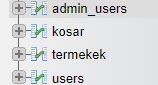
A kód jelenleg nem tartalmaz hibakezelést. Fontos lenne ellenőrizni a lekérdezés eredményét (**$result**), hogy megbizonyosodjunk arról, hogy sikeres volt-e a lekérdezés végrehajtása. Ezenkívül érdemes lenne kezelni az esetleges adatbázis hibákat is.

## **Biztonsági megfontolások**

A jelenlegi kód sebezhető lehet SQL befecskendezésre. Fontos, hogy mindig használjunk paraméterezett lekérdezéseket (mint itt a **bind\_param()**), hogy megakadályozzuk az ilyen típusú támadásokat.

2.1 Adattáblák: Szakdoga

- admin\_users  
- kosar  
- termekek  
- users



## Az “Admin\_users” tábla

## 

### 

### **Tábla struktúra**

Az "admin\_users" tábla a következő mezőket tartalmazza:

**id**: Az adminisztrátor egyedi azonosítója (Primary Key). Ez egy automatikusan növekvő egész szám típusú mező.

**username**: Az adminisztrátor felhasználóneve. Egyedi kell hogy legyen és maximum 255 karakter hosszú string típusú mező.

**password**: Az adminisztrátor jelszava. Biztonsági okokból érdemes hashelve tárolni (pl. bcrypt) és maximum 255 karakter hosszú string típusú mező.

**email**: Az adminisztrátor e-mail címe. Egyedi kell hogy legyen és maximum 255 karakter hosszú string típusú mező.

**created\_at**: Az adminisztrátor létrehozásának időbélyegzője. Ez egy dátum/idő típusú mező, amely automatikusan kitöltődik az adminisztrátor létrehozásakor.

### **Használat**

Az "admin\_users" tábla lehetővé teszi az adminisztrátorok hozzáadását, módosítását, törlését és lekérdezését az adminisztrációs rendszeren keresztül. Az alábbi műveleteket végezhetjük el ezzel a táblával:

* **Adminisztrátorok létrehozása**: Új adminisztrátorokat lehet létrehozni a "username", "password" és "email" mezők kitöltésével. A "created\_at" mező automatikusan kitöltődik a létrehozáskor.
* **Adminisztrátorok frissítése**: Az adminisztrátorok adatait lehet frissíteni, például felhasználónév vagy jelszó módosításával.
* **Adminisztrátorok törlése**: Az adminisztrátorokat lehet törölni az adatbázisból.
* **Adminisztrátorok lekérdezése**: Lekérdezhetjük az összes adminisztrátor adatait, vagy keresést végezhetünk az adminisztrátorok között.

### **Példa SQL lekérdezések**

#### **Adminisztrátorok lekérdezése:**



#### **Új adminisztrátor hozzáadása:**

“INSERT INTO admin\_users (username, password, email) VALUES ('admin', 'hashed\_password', 'admin@example.com');”

#### **Adminisztrátor adatainak frissítése:**



#### **Adminisztrátor törlése:**



## A “termékek” tábla

### **Tábla struktúra**

A "termékek" tábla a következő mezőket tartalmazza:



**id**: A termék egyedi azonosítója (Primary Key). Ez egy automatikusan növekvő egész szám típusú mező.

név: A termék neve. Maximum 255 karakter hosszú string típusú mező.

**leírás**: A termék leírása. Szöveg típusú mező.

**ár**: A termék ára. Decimális típusú mező, ami a termék árát tárolja.

**készleten**: A termék készletmennyisége. Egész szám típusú mező, ami a termék darabszámát tárolja.

**image\_url**: A termék képének elérési útja. Szöveg típusú mező, ami a kép elérési útvonalát tárolja.

### 

### **Használat**

A "products" tábla lehetővé teszi a termékek hozzáadását, módosítását, törlését és lekérdezését a webáruházban. Az alábbi műveleteket végezhetjük el ezzel a táblával:

* **Termékek létrehozása**: Új termékeket lehet létrehozni a "név", "leírás", "ár", "készleten" és "image\_url" mezők kitöltésével.

Nyissa meg a phpMyAdmin felhasználói felületét.

Válassza ki a "permékek" táblát a bal oldali navigációs panelből.

Kattintson az "Insert" fülre az adatok beillesztéséhez.

Adja meg az új termék adatait az egyes mezőkben.

Kattintson az "OK" gombra a művelet végrehajtásához.

* **Termékek frissítése**: A termékek adatait lehet frissíteni, például név, leírás, ár, készletmennyiség vagy kép módosításával.

Nyissa meg a phpMyAdmin felhasználói felületét.

Válassza ki a "termékek" táblát a bal oldali navigációs panelből.

Kattintson az "Edit" fülre az adatok szerkesztéséhez.

Módosítsa a termék adatait az egyes mezőkben.

Kattintson az "OK" gombra a művelet végrehajtásához.

* **Termékek törlése**: A termékeket lehet törölni az adatbázisból.

Nyissa meg a phpMyAdmin felhasználói felületét.

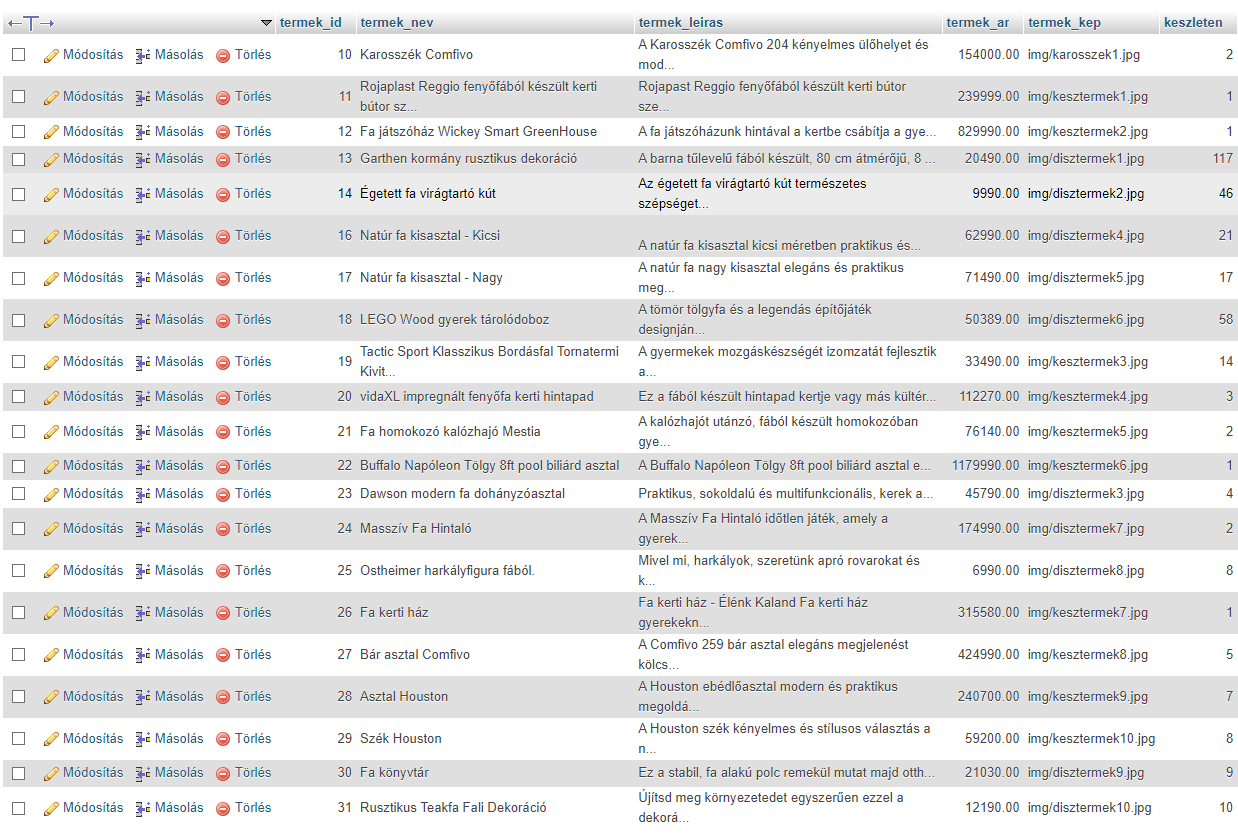
Válassza ki a "termékek" táblát a bal oldali navigációs panelből.

Jelölje ki azokat a termékeket, amelyeket törölni szeretne a táblából.

A táblázat alatt válassza ki a "Törlés" lehetőséget a kiválasztott termékek törléséhez.

* **Termékek lekérdezése**: Lekérdezhetjük az összes termék adatait, vagy keresést végezhetünk a termékek között.

A "termékek" tábla lehetővé teszi a termékek kezelését egy webáruházban. Ez a dokumentáció részletesen bemutatta a tábla struktúráját, használatát és biztonsági megfontolásait, valamint bemutatott példa műveleteket a phpMyAdmin segítségével a termékek kezelésére.



### **Biztonsági megfontolások**

Fontos biztonsági megfontolások az alábbiak:

* Paraméterezett lekérdezések használata: Mindig használjunk paraméterezett lekérdezéseket az adatbázishoz való hozzáférés során a SQL injection támadások elkerülése érdekében.
* Jogosultságkezelés: Biztosítsuk, hogy csak az adminisztrátorok számára elérhető funkciókat csak az adminisztrátorok tudják elérni.

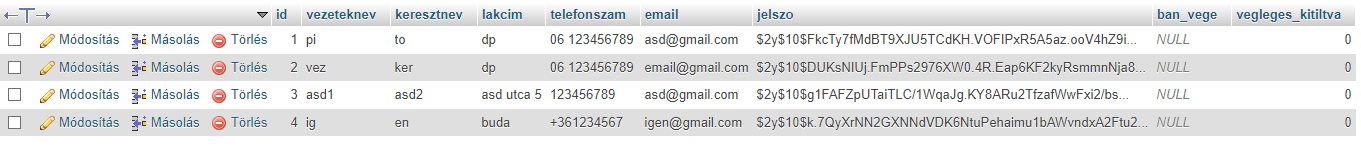
## A “users” tábla

A "users" tábla használatához az alábbiak szükségesek:

* MySQL szerver
* PHP vagy más programnyelv a kapcsolat kialakításához és az adatok kezeléséhez

### **Használat**

A "users" tábla lehetővé teszi a felhasználók hozzáadását, módosítását, törlését és lekérdezését a webalkalmazásban. Az alábbi műveleteket végezhetjük el ezzel a táblával:



* **Felhasználók létrehozása**: Új felhasználókat lehet létrehozni a "first\_name", "last\_name", "phone", "address", "email", "password" és "status" mezők kitöltésével.
* **Felhasználók frissítése**: A felhasználók adatait lehet frissíteni, például kereszt- vagy vezetéknév, telefonszám, lakcím, e-mail cím, jelszó vagy státusz módosításával.
* **Felhasználók törlése**: A felhasználókat lehet törölni az adatbázisból.
* **Felhasználók lekérdezése**: Lekérdezhetjük az összes felhasználó adatait, vagy keresést végezhetünk a felhasználók között.
* **Felhasználó státuszának változtatása**: A felhasználó státuszát lehet változtatni (pl. bannolás, korlátozás, feloldás stb.).



### **Biztonsági megfontolások**

Fontos biztonsági megfontolások az alábbiak:

* A jelszavak biztonságos tárolása: Használjunk erős hashelést a jelszavak tárolására, például bcrypt algoritmust.
* Paraméterezett lekérdezések használata: Mindig használjunk paraméterezett lekérdezéseket az adatbázishoz való hozzáférés során a SQL injection támadások elkerülése érdekében.
* Jogosultságkezelés: Az adminisztrátoroknak megfelelő jogosultsággal kell rendelkezniük a felhasználók státuszának változtatásához.

## Példa phpMyAdmin műveletek

#### **Felhasználók létrehozása:**

1. Nyissa meg a phpMyAdmin felhasználói felületét.
2. Válassza ki a "users" táblát a bal oldali navigációs panelből.
3. Kattintson az "Insert" fülre az adatok beillesztéséhez.
4. Adja meg az új felhasználó adatait az egyes mezőkben.
5. Kattintson az "OK" gombra a művelet végrehajtásához.

#### **Felhasználók frissítése:**

1. Nyissa meg a phpMyAdmin felhasználói felületét.
2. Válassza ki a "users" táblát a bal oldali navigációs panelből.
3. Kattintson az "Edit" fülre az adatok szerkesztéséhez.
4. Módosítsa a felhasználó adatait az egyes mezőkben.
5. Kattintson az "OK" gombra a művelet végrehajtásához.

#### **Felhasználók törlése:**

1. Nyissa meg a phpMyAdmin felhasználói felületét.
2. Válassza ki a "users" táblát a bal oldali navigációs panelből.
3. Jelölje ki azokat a felhasználókat, akiket törölni szeretne a táblából.
4. A táblázat alatt válassza ki a "Delete" lehetőséget a kiválasztott felhasználók törléséhez.

**Összegzés**

A "users" tábla lehetővé teszi a felhasználók kezelését egy webalkalmazásban. Ez a dokumentáció részletesen bemutatta a tábla struktúráját, használatát és biztonsági megfontolásait, valamint bemutatott példa műveleteket a phpMyAdmin segítségével a felhasználók kezelésére.

# 3. Felhasználói Dokumentáció

### **Áttekintés**

A "Webadmin Bejelentkezés" oldal lehetővé teszi a webadminisztrátorok számára a webadmin felület elérését és a bejelentkezést. Ez az oldal biztosítja a biztonságos hozzáférést a webes adminisztrációs felülethez, ahol a weboldal tartalmát és beállításait lehet módosítani.

### **Hogyan működik?**

**Bejelentkezési adatok megadása**: A felhasználónak meg kell adnia a felhasználónevét és a jelszavát a megfelelő mezőkbe.

**Bejelentkezés gomb megnyomása**: Miután a felhasználó megadta a bejelentkezési adatokat, meg kell nyomni a "Bejelentkezés" gombot.

**Hitelesítés**: Az oldal ellenőrzi a megadott felhasználónevet és jelszót az adatbázisban tárolt adatokkal.

**Hitelesítés eredménye**: Ha a megadott adatok helyesek, a felhasználó sikeresen bejelentkezik az adminisztrációs felületre, és átirányításra kerül a "admin.php" oldalra.

**Hibás bejelentkezés**: Ha a megadott felhasználónév vagy jelszó helytelen, a felhasználó hibaüzenetet kap, és nem sikerül bejelentkezni.

### **Használati utasítások**

**Felhasználónév és jelszó megadása**: Adja meg a felhasználónevét és a jelszavát a megfelelő mezőkbe.

**Bejelentkezés**: Kattintson a "Bejelentkezés" gombra a bejelentkezéshez.

**Hiba esetén**: Ha a bejelentkezés nem sikerül, ellenőrizze a megadott felhasználónevet és jelszót, majd próbálja meg újra.

A "fooldal.php" fájl a webshop főoldalát tartalmazza. Ez az oldal az első, amit a felhasználó lát, amikor a webshopra navigál. A főoldal tartalmazza a webshop logóját, a navigációs menüt, promóciós szekciókat, kiemelt termékeket és a láblécet.

### **3.1 Főbb funkciók és tartalom**

**Fejléc (Header)**: A fejléc tartalmazza a webshop logóját, ami visszavezet a főoldalra, és az oldal tetején jelenik meg.

**Navigációs Menü**: A navigációs menü lehetővé teszi a felhasználók számára, hogy a különböző oldalak között navigáljanak a webshopban. A menüpontok a következők:

* + Főoldal
  + Kész Termékek
  + Dísztermékek
  + Rólunk
  + Kapcsolat
  + Bejelentkezés
  + Regisztráció

**Promóciós Szekció**: A promóciós szekcióban aktuális akciókat vagy promóciókat mutat be a webshop.

**Kiemelt Termékek**: A főoldalon kiemelt termékek láthatóak, amelyek gyors hozzáférést biztosítanak a legnépszerűbb vagy legújabb termékekhez.

**Lábléc (Footer)**: A lábléc az oldal alján található, tartalmazza a webshop nevét, a jogi nyilatkozatot és egyéb linkeket.

### **Használati utasítások**

* **Navigáció**: A navigációs menü segítségével böngészhet a webshop különböző részei között.
* **Termékek megtekintése**: Kattintson a kiemelt termékek címére vagy az "Add to Cart" gombra a részletes termékinformációk megtekintéséhez.
* **Bejelentkezés/Regisztráció**: Ha már van fiókja, jelentkezzen be, vagy regisztráljon egy új fiók létrehozásához.

### **Fontos tudnivalók**

* **Termékinformációk**: A kiemelt termékeket csak a termék nevére kattintva vagy az "Add to Cart" gombra kattintva lehet megtekinteni.
* **Fiók szükségessége**: A vásárláshoz be kell jelentkeznie vagy regisztrálnia kell egy fiókot.

### **Technikai részletek**

* **Nyelv és Technológia**: Az oldal HTML, CSS és JavaScript nyelven van írva, és egy PHP alapú webshop backenddel kommunikál.
* **Responszív dizájn**: Az oldal reszponzív dizájnt használ, ami lehetővé teszi az optimális megjelenítést különböző eszközökön és képernyőméreteken.

### **Kosárhasználati útmutató**

* **Kosár Tartalma**: A "Kosár Tartalma" részben láthatja a kosárba helyezett termékeket, azok árait és mennyiségét.
* **Törlés a Kosárból**: Ha szeretné eltávolítani a terméket a kosárból, kattintson a "Töröl" gombra a termék mellett.
* **Összesen**: Az összesített végösszeget láthatja az "Összesen" sorban.

### **Egyéb Fontos Tudnivalók**

* **Kapcsolat és Támogatás**: Ha további kérdései vagy problémái vannak az oldal használata során, vagy szeretné megosztani visszajelzéseit, kérjük, lépjen kapcsolatba a webshop ügyfélszolgálatával.
* **Adatvédelem és Biztonság**: Az oldal adatvédelmi és biztonsági irányelveiről és gyakorlatáról további információkat találhat a webshop adatvédelmi tájékoztatójában.

### **3.2 Technikai Részletek és Frissítések**

* **Frissítések és Karbantartás**: Az oldal fejlesztése és karbantartása folyamatos, hogy biztosítsa a lehető legjobb felhasználói élményt és biztonságot.
* **Hibabejelentés**: Ha technikai hibákat vagy problémákat észlel az oldal használata során, kérjük, jelezze azokat a webshop ügyfélszolgálatának.

### **Kosárhasználati útmutató folytatása**

* **Kijelentkezés**: Ha befejezte a vásárlást és további teendői nincsenek az oldalon, kérjük, használja a "Kijelentkezés" lehetőséget a biztonságos kilépéshez az oldalról.

### **Egyéb Fontos Tudnivalók (folytatás)**

* **Felhasználási Feltételek**: Az oldal használata során elfogadja és betartja a webshop által előírt felhasználási feltételeket és szabályokat. Kérjük, olvassa el a Felhasználási Feltételeket az oldalon található linkeken keresztül.
* **Frissítések és Hibajavítások**: Az oldal folyamatos fejlesztés alatt áll, és időről időre frissítések és hibajavítások lesznek végrehajtva a jobb felhasználói élmény és a biztonság érdekében.

### **Technikai Részletek**

* **Adatvédelem és Biztonság**: A webshop nagy hangsúlyt fektet az adatvédelemre és a biztonságra. Az Ön adatait bizalmasan kezeljük, és csak az oldal működéséhez és a vásárlási folyamat teljesítéséhez szükséges mértékben használjuk fel.
* **Technikai Támogatás**: Ha technikai kérdései vagy problémái merülnek fel az oldal használata során, kérjük, vegye fel a kapcsolatot az oldal üzemeltetőjének technikai támogatási csapatával.

### **3.3 Jogi Tudnivalók és Adatvédelem**

* **Cookie-k használata**: Az oldal cookie-kat használ az Ön webes élményének személyre szabása és javítása érdekében. A cookie-k információkat tárolnak az Ön böngészőjében, például az oldal beállításait vagy bejelentkezési adatait, hogy ne kelljen újra beírnia azokat az oldal újbóli meglátogatásakor.
* **Adatvédelem**: Az Ön személyes adatait bizalmasan kezeljük és nem adjuk át harmadik félnek. Az adatok védelme érdekében az oldal biztonsági intézkedéseket alkalmaz, hogy megelőzze az illetéktelen hozzáférést, a változtatást vagy a jogosulatlan adatokhoz való hozzáférést.

### **Kapcsolat és További Információk**

* **Ügyfélszolgálat**: Ha további kérdései vagy észrevételei vannak az oldallal vagy a vásárlási folyamattal kapcsolatban, kérjük, lépjen kapcsolatba az oldal ügyfélszolgálatával. A kapcsolatfelvétel módjait megtalálhatja az "Elérhetőségek" vagy "Kapcsolat" menüpontokban.
* **Hírlevél**: Iratkozzon fel hírlevelünkre, hogy értesüljön az új termékekről, akciókról és egyéb fontos információkról. A feliratkozási lehetőséget az oldal alján található űrlapon keresztül érheti el.

### **Visszajelzés és Értékelés**

* **Vásárlói Visszajelzések**: Nagyra értékeljük visszajelzéseit és észrevételeit az oldallal kapcsolatban. Ha elégedett vagy elégedetlen volt az élményével, kérjük, ossza meg velünk véleményét az oldal visszajelzési űrlapján keresztül.
* **Termékértékelések**: Kérem, ossza meg velünk véleményét a vásárolt termékekről. Az Ön visszajelzése segít más vásárlóknak a döntésben, és hozzájárul az oldal fejlesztéséhez és a szolgáltatásaink minőségének javításához.

# 4. Tesztelés

A kódban van egy bejelentkezési funkció, amely lehetővé teszi a felhasználók számára a webadmin felületre való bejelentkezést. Ez a funkció tesztelésre szorul annak érdekében, hogy megbizonyosodjunk róla, hogy helyesen működik-e. A következő lépésekkel lehetne tesztelni:

# 

Ezek a kiemelt részek azok, amelyek fontosak a bejelentkezési funkció helyes működésének ellenőrzéséhez és teszteléséhez. Az itt lévő tesztek segítenek biztosítani, hogy a bejelentkezési folyamat megfelelően működik, és kezeljék az esetleges hibákat vagy kivételeket a felhasználói interakció során.

# 4.1 Tesztelési lépések

**1.** **Adatbázis ellenőrzése**: Ellenőrizze, hogy az adatbázis tartalmaz-e adminisztrátor felhasználókat, és hogy a jelszavak helyesen vannak-e titkosítva.

**2.** **Felhasználói adatok bevitele**: Töltse ki a bejelentkezési űrlapot a meglévő adminisztrátor felhasználó adataival (felhasználónév és jelszó).

**3.** **Bejelentkezési funkció tesztelése**: Nyomja meg a "Bejelentkezés" gombot, hogy elküldje az adatokat a szervernek és végre legyen hajtva a bejelentkezési funkció.

**4.** **Helyes bejelentkezés ellenőrzése**: Ha a bejelentkezés sikeres volt, ellenőrizze, hogy a felhasználó át lett-e irányítva az adminisztrációs felületre.

**5.** **Hibás bejelentkezés ellenőrzése**: Ha a bejelentkezés sikertelen volt (hibás felhasználónév vagy jelszó), ellenőrizze, hogy megjelenik-e a megfelelő hibaüzenet a felhasználónak.

**6.** **Hibás adatok bevitele**: Tesztelje, hogy mi történik, ha hibás adatokat ad meg a bejelentkezési űrlapon, például üres mezők vagy helytelen formátumú adatok.

**7.** **Adatbázis lekérdezés ellenőrzése**: Győződjön meg róla, hogy a felhasználói adatok helyesen vannak-e lekérdezve az adatbázisból a megadott felhasználónév alapján.

**8.** **Jelszó ellenőrzése**: Ellenőrizze, hogy a megadott jelszó helyes-e a tárolt jelszóval összehasonlítva, és hogy a felhasználó bejelentkezett-e a helyes jelszóval.

**9.** **Felhasználói munkamenet kezelése**: Győződjön meg róla, hogy a felhasználói munkamenet megfelelően kezelésre került-e, és hogy a felhasználó belépett-e a rendszerbe, miután helyesen megadta az adatokat.

Ezek a lépések segítenek ellenőrizni és biztosítani a bejelentkezési funkció helyes működését az adminisztrációs felületen keresztül.

# 4.2 Tesztelési funkciók és dokumentáció

**Munkamenet-kezelés ellenőrzése:**

**Munkamenet törlése**: A **session\_destroy()** függvény segítségével a munkamenetet teljesen töröljük. Ezt ellenőrizhetjük úgy, hogy megnézzük, hogy a felhasználó kilépése után a munkamenet változók üresek-e.

**Cookie-k törlése**: Ha a PHP beállítások engedélyezik a sütik használatát a munkamenetek azonosítására (**session.use\_cookies** beállítás), akkor a **setcookie()** függvénnyel a munkamenet cookie-jait is töröljük. Ezt ellenőrizhetjük a böngésző sütijeinek beállításaiban vagy azáltal, hogy megnézzük a HTTP kéréseket és válaszokat.

**Dokumentáció:**

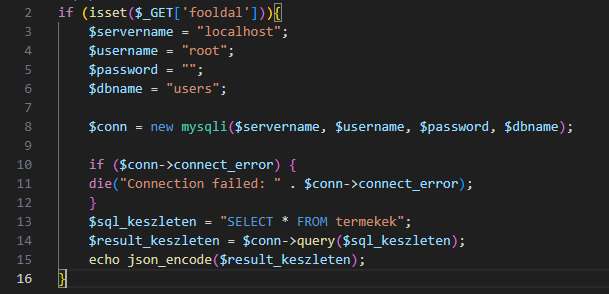
* **Munkamenet törlése**: A munkamenetek törlése a felhasználó kijelentkezésekor fontos biztonsági intézkedés, hogy ne lehessen az előző munkamenet adataira hivatkozni a kijelentkezés után. A **session\_destroy()** függvény felelős a munkamenet teljes törléséért.
* **Cookie-k törlése**: Ha a munkamenet-kezelés a PHP beállításaiban sütiket használ (**session.use\_cookies**), akkor a **setcookie()** függvény segítségével töröljük ezeket a sütiket. Ez biztosítja, hogy a böngésző ne tárolja tovább a munkamenet azonosítóját.

**Tesztelési eljárások:**

**Manuális tesztelés**: Futtassuk a kódot, majd ellenőrizzük, hogy a munkamenet változók üresek-e a kilépés után, és a böngészőben nincsenek-e a munkamenethez kapcsolódó sütik.

**Automatizált tesztelés**: Használjunk automatizált tesztelési keretrendszert (pl. PHPUnit), hogy ellenőrizzük a kijelentkezési folyamatot, beleértve a munkamenetek törlését és a sütik törlését is.

A megadott kódban egy adatbázisból lekérdezett adatokat JSON formátumban visszaadó funkció található, amely akkor fut le, ha a **$\_GET['fooldal']** változó értéke megtalálható. A kód az adatbázis kapcsolódásával kezdődik, majd lekérdezi a "termekek" tábla minden rekordját, és a lekérdezés eredményét JSON formátumban visszaadja.



#### **Adatok lekérése és JSON formátumban visszaadása:**

**Adatbázis kapcsolódás ellenőrzése**: Ellenőrizzük, hogy sikeresen létrejött-e a kapcsolat az adatbázissal. Ha nem sikerült a kapcsolódás, akkor megfelelő hibaüzenetet kell kiírni.

**Adatok lekérése**: A kód lekérdezi a "termekek" tábla összes rekordját.

**JSON formátumban visszaadás**: Az adatokat JSON formátumban visszaadjuk a kliensnek. A JSON formátum lehetővé teszi az adatok egyszerű és strukturált átvitelét.

#### **Tesztelési eljárások:**

**Manuális tesztelés**: Kérjük le a "fooldal" paraméterrel az oldalt, majd ellenőrizzük, hogy a kapott válasz JSON formátumban tartalmazza-e a "termekek" tábla adatait.

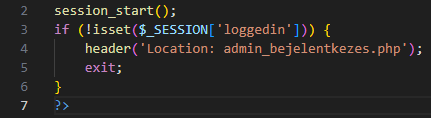
**Automatizált tesztelés**: Használjunk automatizált tesztelési keretrendszert (pl. PHPUnit vagy Selenium), hogy ellenőrizzük a funkció helyes működését különböző körülmények között.

## Felhasználó bejelentkezésének ellenőrzése:

**Munkamenet ellenőrzése**: Ellenőrizzük, hogy van-e érvényes munkamenet (**$\_SESSION['loggedin']**) a felhasználó számára. Ha nincs, az azt jelenti, hogy a felhasználó nincs bejelentkezve.

**Átirányítás**: Ha a felhasználó nincs bejelentkezve, akkor átirányítjuk őt a bejelentkező oldalra (**admin\_bejelentkezes.php**).

**Kilépés a szkriptből**: Az **exit** utasítás megszakítja a kód további végrehajtását, így csak a bejelentkezett felhasználók férhetnek hozzá az adminisztrációs felülethez.



**Felhasználó azonosítása és ellenőrzése:** Először az oldal ellenőrzi, hogy a **GET** kérésben szereplő **id** paraméter megfelelő-e, és szám-e. Ezután lekéri az adott felhasználóhoz tartozó **vegleges\_kitiltva** mező értékét az adatbázisból. Ha a felhasználó már véglegesen ki van tiltva (**vegleges\_kitiltva == 1**), akkor a végrehajtás leáll, és hibaüzenetet ad ki. Ha nem található a felhasználó az adatbázisban, szintén hibaüzenetet ad ki.

**Felhasználó végleges kitiltása:** Ha minden ellenőrzés sikeres volt, akkor végrehajtódik a felhasználó végleges kitiltása az adatbázisban. Az **UPDATE** SQL utasítással beállítja a **ban\_vege** értékét **NULL**-ra és a **vegleges\_kitiltva** értékét **1**-re a megadott felhasználóhoz tartozó rekordon. Ezután kiírja, hogy a művelet sikeres volt.

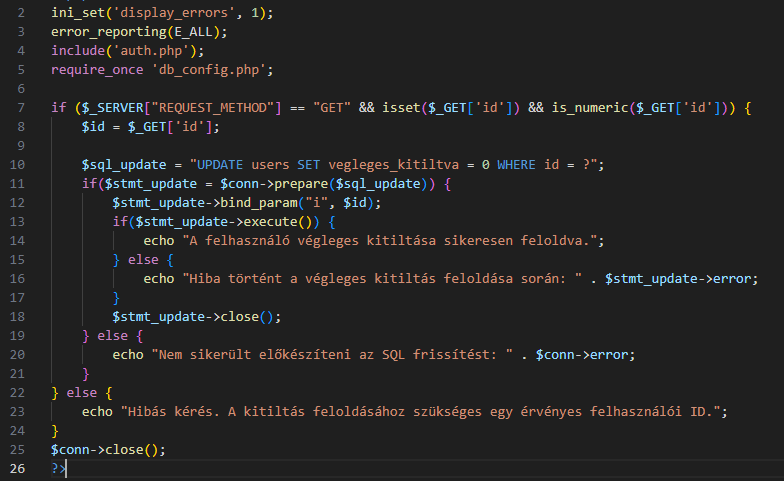
**Átirányítás:** Végül az oldal átirányítja a felhasználót a **felhasznalok\_listazasa.php** oldalra, ahol a felhasználók listája található.



Az első rész az **ini\_set** és **error\_reporting** függvények hívása a hibák kijelzésének beállítására és az összes hiba megjelenítésére szolgál. Ez segíthet a hibakeresésben fejlesztés közben, de éles környezetben érdemes lenne kikapcsolni.

A következő rész az **if ($\_SERVER["REQUEST\_METHOD"] == "GET" && isset($\_GET['id']) && is\_numeric($\_GET['id']))** feltétel ellenőrzi, hogy a kérés metódusa GET-e, van-e 'id' nevű paramétere a kérésnek és ez az 'id' numerikus-e. Ha mindezek igazak, akkor a kód folytatódik, egyébként hibát dob.

Ezután a kód végrehajtja a feloldást a felhasználó végleges kitiltásának (amennyiben az 'id' érvényes és numerikus). Ha a kitiltás sikeresen feloldódik, akkor a kód kiírja a sikerüzenetet, egyébként a hibát.

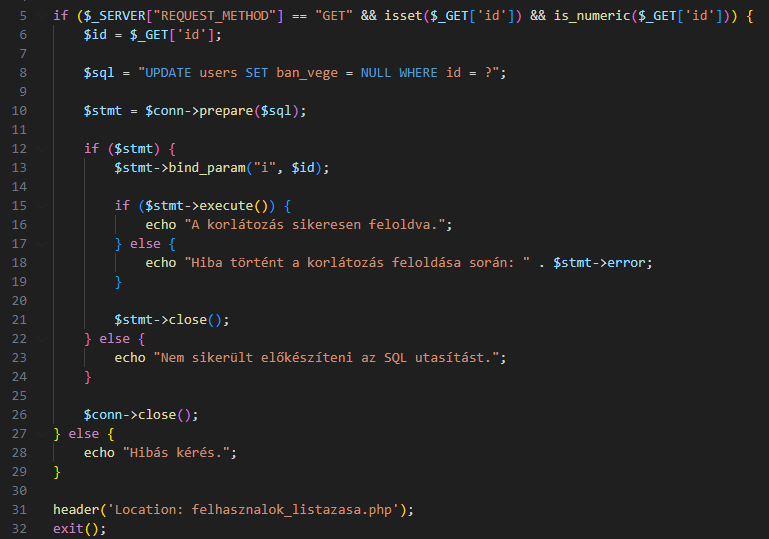


if ($\_SERVER["REQUEST\_METHOD"] == "GET" && isset($\_GET['id']) && is\_numeric($\_GET['id'])) { $id = $\_GET['id'];

Az első rész az **if ($\_SERVER["REQUEST\_METHOD"] == "GET" && isset($\_GET['id']) && is\_numeric($\_GET['id']))** feltétel ellenőrzi, hogy a kérés metódusa GET-e, van-e 'id' nevű paramétere a kérésnek, és ez az 'id' numerikus-e. Ha mindezek igazak, akkor a kód folytatódik, egyébként hibát dob.

Ezután a kód végrehajtja a korlátozás feloldását a felhasználókban (amennyiben az 'id' érvényes és numerikus). Ha a feloldás sikeresen megtörténik, akkor a kód kiírja a sikerüzenetet, egyébként a hibát.

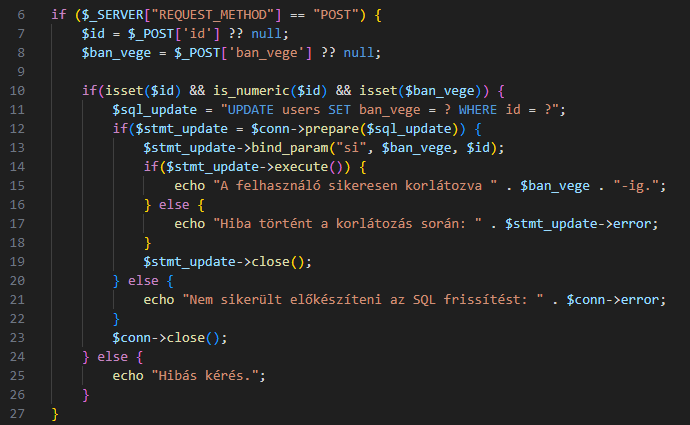
Végül a kód a felhasználók listájának oldalára irányítja a felhasználót, majd kilép. Ez azt jelenti, hogy ha a kód eléri ezt a pontot, akkor már nincs szükség további kódvégrehajtásra, és azonnal továbblendül a felhasználó a listázási oldalra. Ez az utolsó sor tesztelhető funkciót nyújt, mivel irányítja a felhasználót egy másik oldalra, ami azt jelenti, hogy a kód működése kipróbálható.



Az első rész az **if ($\_SERVER["REQUEST\_METHOD"] == "POST")** feltétel ellenőrzi, hogy a kérés metódusa POST-e. Ha igen, akkor a kód folytatódik, különben nem történik semmi.

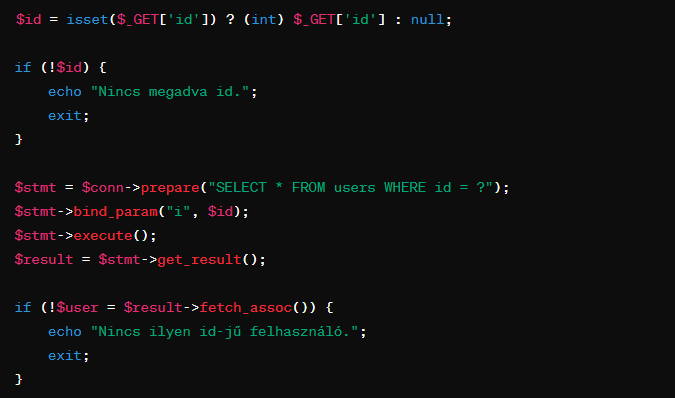
Ezt követően az adatokat ellenőrzik és előkészítik a frissítéshez. Ha az adatok érvényesek és a frissítés sikeres, akkor kiírják a sikerüzenetet, egyébként hibát jelentenek.

Az oldal alján egy űrlap található, amely lehetővé teszi a felhasználó korlátozását. A form elküldésekor az űrlap azonosítója és a korlátozás vége POST kérést küld a saját oldalra. Ez tesztelhető működést biztosít a felhasználó korlátozásához, valamint az űrlap adatok POST kéréssel történő továbbításához.



if ($\_SERVER["REQUEST\_METHOD"] == "POST") { $id = $\_POST['id'] ?? null; $ban\_vege = $\_POST['ban\_vege'] ?? null;

Ez a részlet biztosítja, hogy a szükséges adatok rendelkezésre állnak, mielőtt megjelenítenék vagy módosítanák a felhasználó adatait.



Az első részben ellenőrzik a kapcsolódást az adatbázishoz. Ha nem sikerül a kapcsolódás, akkor hibát dob és kilép a program.

A következő részben ellenőrzik, hogy van-e megadva **id**. Ha nem, akkor hibát dob és kilép.

Ezután lekérdezik a felhasználót az adatbázisból az adott **id** alapján. Ha nincs ilyen felhasználó az adatbázisban, akkor hibát dob és kilép.

A HTML rész a szerkesztőfelületet tartalmazza, ahol megjelenítik a felhasználó adatait egy űrlapon, amelyet a felhasználó módosíthat. Ez tesztelési célokat szolgál, mivel lehetővé teszi a felhasználó adatainak módosítását, majd az adatok elküldésével a **felhasznalo\_frissitese.php** oldalnak lehetőséget ad az adatok frissítésére az adatbázisban.

**Ez a részlet a következőt teszi:**

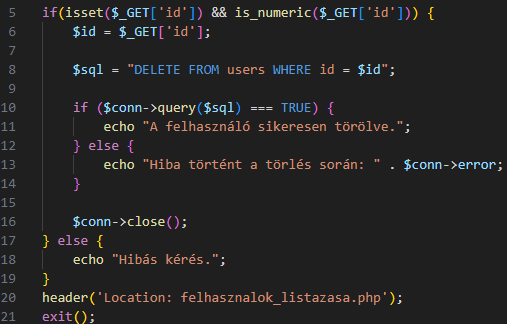
Ellenőrzi, hogy van-e **id** paraméter a GET kérésben.

Ha van, akkor megpróbálja az **id** értékét integerré konvertálni.

Ellenőrzi, hogy az **id** értéke nem üres és nem nulla.

Lekéri a felhasználó adatait az adatbázisból az **id** alapján.

Ellenőrzi, hogy a lekérdezés sikeres volt-e és hogy van-e találat. Ha nincs, akkor hibaüzenetet dob.

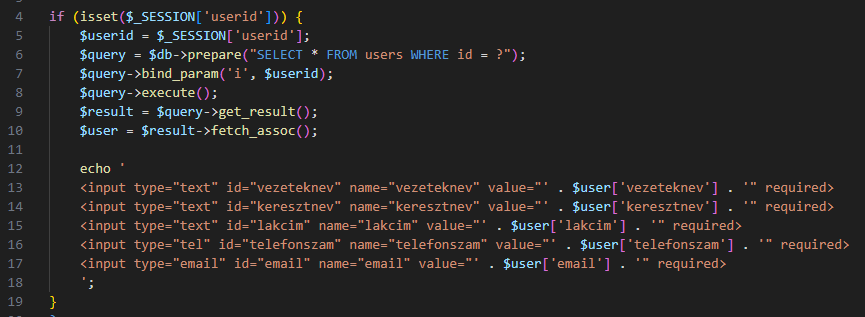


Az első részben ellenőrzik, hogy van-e **id** paraméter a GET kérésben és hogy az **id** numerikus-e. Ha nincs vagy nem numerikus, akkor hibát jelez és kilép a program.

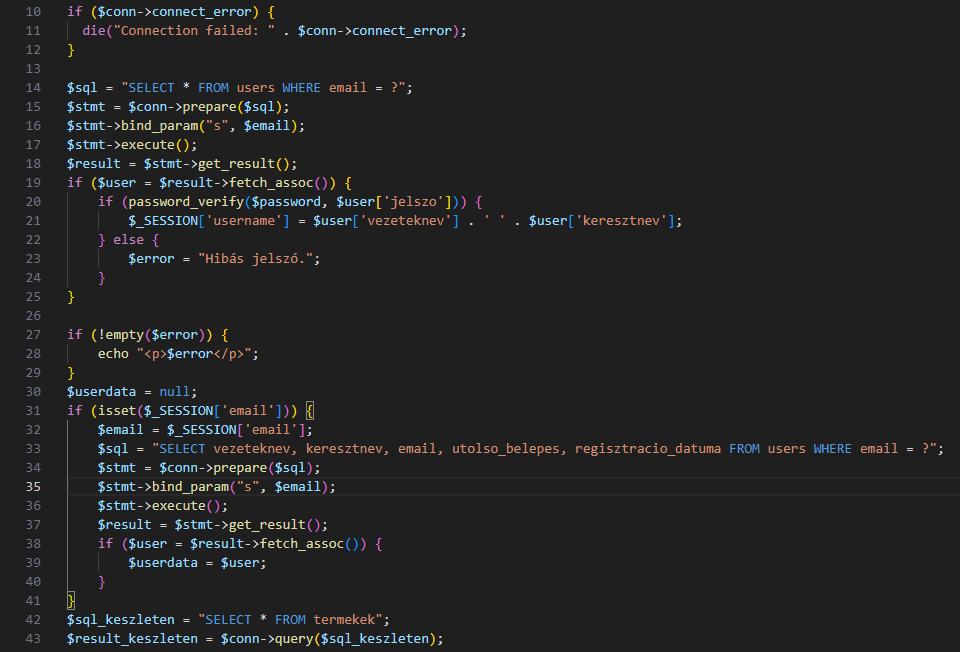
Ezután előkészítik a törlési műveletet az adatbázisban az adott **id** alapján.

A kód végrehajtja a törlési műveletet az adatbázisban, és ellenőrzi, hogy a művelet sikeres volt-e. Ha igen, akkor kiírja a sikerüzenetet, egyébként hibát jelez.

Végül a kód átirányítja a felhasználót a **felhasznalok\_listazasa.php** oldalra, ami azt jelenti, hogy ha a kód ezt a pontot eléri, akkor már nincs szükség további kódvégrehajtásra, és azonnal továbblendül a felhasználó a listázási oldalra. Ez tesztelhető működést nyújt, mivel irányítja a felhasználót egy másik oldalra.



1. Először kapcsolódik az adatbázishoz.
2. Ellenőrzi, hogy van-e **$\_SESSION['userid']** érték, ami azt jelzi, hogy a felhasználó be van jelentkezve.
3. Ha a felhasználó be van jelentkezve, akkor lekérdezi a felhasználó adatait az adatbázisból az **id** alapján.
4. Az adatokat megjeleníti az űrlapon előre kitöltött mezőkben.



Ebben a kódrészletben többféle tesztelési funkcionalitás is található:

**Felhasználó bejelentkeztetés ellenőrzése**: A kód megvizsgálja, hogy a felhasználó be van-e jelentkezve azáltal, hogy ellenőrzi, hogy létezik-e **$\_SESSION['username']**. Ha igen, akkor megjeleníti a felhasználó nevét a fejlécben, és lehetővé teszi a kijelentkezést. Ellenkező esetben megjeleníti a bejelentkezési és regisztrációs linkeket.

**Kosárba helyezés funkció**: A kód lehetővé teszi a termékek kosárba helyezését azáltal, hogy hozzáad egy eseményfigyelőt minden "Kosárba" gombhoz, amely a termékhez tartozik. Amikor a felhasználó rákattint a gombra, a termék adatait hozzáadja a kosárhoz, amelyet a helyi tárolásban tárol, és frissíti a kosár tartalmát az oldalon.

**Kosár megjelenítése**: A kód megjeleníti a kosár tartalmát egy modális ablakban, amelyet a felhasználó fel tud hívni az oldalon található "Kosaram" gombbal. A kosár tartalma dinamikusan frissül a felhasználó által hozzáadott és eltávolított tételek alapján.

Ezek a tesztelési funkciók segítenek abban, hogy a weboldal működése és felhasználói élménye megfelelő legyen. Az ellenőrzések és funkciók segítenek a hibák felderítésében és azok kezelésében, valamint a weboldal funkcionalitásának tesztelésében és javításában.

**Egységtesztek**: Ezek az apró tesztek ellenőrzik az egyes függvények működését. Például, ha lenne egy függvény a készletmódosítás végrehajtására, akkor annak tesztje ellenőrizné, hogy a megfelelő adatokat módosítja-e az adatbázisban.

**Integrációs tesztek**: Ezek a tesztek a rendszer részeinek együttes működését ellenőrzik. Például, egy integrációs teszt ellenőrizhetné, hogy az űrlap megfelelően kommunikál-e az adatbázissal, és helyesen frissíti-e a készletet.

**Elfogadási tesztek**: Ezek a tesztek az alkalmazás teljes funkcionalitását ellenőrzik a felhasználói szempontból. Például, az egyik ilyen teszt ellenőrizhetné, hogy a felhasználó az űrlapon keresztül meg tudja-e változtatni a termék készletét, és megkapja-e a megfelelő visszajelzést a módosításról.

# 5. Irodalomjegyzék

1. **Frontend technológiák:**
   * Flavio Copes: "Front-End Developer Handbook 2019"
   * Jon Duckett: "HTML and CSS: Design and Build Websites"
   * Kyle Simpson: "You Don't Know JS" (könyvsorozat)
   * Eloquent JavaScript: <https://eloquentjavascript.net/>
2. **Backend technológiák:**
   * Robin Nixon: "Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5"
   * Node.js Documentation: https://nodejs.org/en/docs/
   * Express.js Documentation: <https://expressjs.com/>
3. **JavaScript és jQuery:**
   * David Flanagan: "JavaScript: The Definitive Guide"
   * Douglas Crockford: "JavaScript: The Good Parts"
   * jQuery Documentation: https://api.jquery.com/
   * MDN Web Docs: JavaScript Guide: <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide>
4. **PHP és MySQL:**
   * Matt Zandstra: "PHP Objects, Patterns, and Practice"
   * David Sklar: "Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5"
   * PHP Documentation: <https://www.php.net/docs.php>
   * MySQL Documentation: <https://dev.mysql.com/doc/>
5. **CSS és stílusrendszerek:**
   * Eric A. Meyer, Estelle Weyl: "CSS: The Definitive Guide"
   * Jonathan Snook, Steve Smith, Cameron Adams, et al.: "The Art and Science of CSS"
   * Sass Documentation: https://sass-lang.com/documentation
   * Bootstrap Documentation: https://getbootstrap.com/docs/
6. **Adatbázisok és adatmodellálás:**
   * Abraham Silberschatz, Henry F. Korth, S. Sudarshan: "Database System Concepts"
   * Rick F. van der Lans: "Introduction to SQL"
   * MySQL Documentation: <https://dev.mysql.com/doc/>
   * PHPMyAdmin Documentation: https://docs.phpmyadmin.net/

Ezek a források segítettek a frontend, backend, JavaScript, PHP, CSS, PHPMyAdmin és MySQL témájú ismereteink elmélyítésében és a Wood Design Store webshop elkészítésében.

# 6. Mellékletek

https://www.w3schools.com/sql/  
https://www.w3schools.com/sql/sql\_intro.asp

https://www.w3schools.com/sql/sql\_create\_db.asp

https://www.reddit.com

https://codefinity.com/get-started/spa/v16/spv16/quarter?utm\_source=google&utm\_medium=cpc&utm\_campaign=20955067105&utm\_content=161127022780&utm\_term=coding&gad\_source=1&gclid=CjwKCAjwoa2xBhACEiwA1sb1BMhk3dyIIBPpw52gJNpoUQdSowHt\_zfmYKUByVUPR-P9lw5FBeDsRxoCJrkQAvD\_BwE

https://www.phptutorial.net

https://discord.com

https://hu.wikipedia.org/

https://github.com/bbLDxd/Vizsgaremek.git

git@github.com:bbLDxd/Vizsgaremek.git

# 

# 