**Baranya Vármegyei Szakképzési Centrum**

**Simonyi Károly Technikum és Szakképző Iskola**

**Vizsgaremek**

**Pécs**

**2025**

Készítették: Pál Rajmund, Futó Csaba

**Baranya Vármegyei Szakképzési Centrum**

**Simonyi Károly Technikum és Szakképző Iskola**

**Pécs**

**2025**

Szakma megnevezése: Szoftverfejlesztő és -tesztelő

A szakma azonosító száma: 5 0613 12 03

Készítették: Pál Rajmund, Futó Csaba

**Vizsgaremek**

Wimu Webshop

**2025**

**Eredetiség nyilatkozat**

Alulírottak: Futó Csaba, Pál Rajmund

A Baranya Vármegyei SzC Simonyi Károly Technikum és Szakképző Iskola, Szoftverfejlesztő és tesztelő végzős tanulói, büntetőjogi és fegyelmi felelősségünk tudatában nyilatkozunk és aláírásunkkal igazoljuk, hogy a(z)

Wimu

című vizsgaremek saját önálló munkánk, az abban hivatkozott szakirodalom felhasználása a forráskezelés szabályai szerint történt.

Tudomásul vesszük, hogy vizsgaremek esetén plágiumnak számít:

* szószerinti idézet közlése idézőjel és hivatkozás megjelölése nélkül,
* tartalmi idézet hivatkozás megjelölése nélkül,
* más publikált gondolatainak saját gondolatként való feltüntetése

Alulírott kijelentjük, hogy a plágium fogalmát megismertük, és tudomásul vesszük, hogy plágium esetén a vizsgaremekünk visszautasításra kerül.

Tanulók aláírása

Tartalom

[**1.** **Felhasználói dokumentáció** 7](#_Toc195559040)

[**1.1 Letöltés** 7](#_Toc195559041)

[**1.2 Adatbázis és fájl elhelyezés** 8](#_Toc195559042)

[**1.3** **Weboldal használata** 9](#_Toc195559043)

[**2. Bevezető** 15](#_Toc195559044)

[**2.1 Projekt célja és motiváció** 15](#_Toc195559045)

[**2.2 Tanultak és új ismeretek** 15](#_Toc195559046)

[**2.3 Későbbi tervek** 16](#_Toc195559047)

[**2.4 Csapatmunka és szerepkörök** 16](#_Toc195559048)

[**2.5 Összefoglalás** 16](#_Toc195559049)

[**3. Témaválasztás:** 17](#_Toc195559050)

[**3.1 Közös cél: egy életképes prototípus létrehozása** 17](#_Toc195559051)

[**3.2 Mit hoz a jővő?** 17](#_Toc195559052)

[**4. Az alkalmazott fejlesztői eszközök:** 18](#_Toc195559053)

[**4.1. Programozási és leíró nyelvek & keretrendszerek** 18](#_Toc195559054)

[**4.2. Fejlesztői környezetek & eszközök** 18](#_Toc195559055)

[**4.3. Design & Médiaeszközök** 18](#_Toc195559056)

[**4.4. Könyvtárak & Függőségek** 18](#_Toc195559057)

[**4.5. Egyéb eszközök** 18](#_Toc195559058)

[**4.6. Biztonsági eszközök** 18](#_Toc195559059)

[**4.7. Kompatibilitás & tesztelés** 19](#_Toc195559060)

[**5. Tervezési módszer** 19](#_Toc195559061)

[**5.1 Alkalmazás tervezése:** 19](#_Toc195559062)

[**5.2 Tervezési módszertan:** 19](#_Toc195559063)

[**5.3 Vizuális modellezés:** 19](#_Toc195559064)

[**5.4 OOP megvalósulása:** 20](#_Toc195559065)

[**6. Az adatbázis ábrája:** 21](#_Toc195559066)

[**6.1 Főbb táblák:** 21](#_Toc195559067)

[**6.2 Fő funkciók:** 22](#_Toc195559068)

[**6.3 Táblák kapcsolatai** 22](#_Toc195559069)

[**6.4 Indexek és optimalizálás** 23](#_Toc195559070)

[**6.5 Biztonsági intézkedések** 23](#_Toc195559071)

[**6.6 Jelentős funkciók és adatbázis integráció** 23](#_Toc195559072)

[**6.7 Későbbi fejlesztések** 24](#_Toc195559073)

[**7. Részletes feladatspecifikáció, algoritmusok** 25](#_Toc195559074)

[**7.1. belepes.php – Belépési folyamat** 25](#_Toc195559075)

[**7.2 Főbb függvények és metódusok specifikációja** 26](#_Toc195559076)

[**7.3 Vásárlói adatok validálása (2\_vasarloi\_adatok.php)** 27](#_Toc195559077)

[**7.4. Fizetési mód kiválasztása (3\_fizetesi\_modok.php)** 27](#_Toc195559078)

[**7.5. Rendelés összegzése (4\_rendeles\_osszegzes.php)** 28](#_Toc195559079)

[**7.6 Rendelés feldolgozása (5\_rendeles\_elkuld.php)** 29](#_Toc195559080)

[**7.7. Bejelentkezési rendszer (belepes.php)** 30](#_Toc195559081)

[**7.8 Algoritmusok leírása** 30](#_Toc195559082)

[**7.9 Rendelésfeldolgozás folyamata** 31](#_Toc195559083)

[**7.10. Termékkereső algoritmus (index.php)** 31](#_Toc195559084)

[**7.11. Kosár megtekintése (kosar\_megtekintes.php)** 32](#_Toc195559085)

[**7.12. Termék kosárba tétele (kosarba\_tesz.php)** 33](#_Toc195559086)

[**7.13. Profil módosítás (profil\_modosit.php és** 34](#_Toc195559087)

[**7.14. Regisztráció (reg.php és reg\_ellenoriz.php)** 35](#_Toc195559088)

[**7.15. Termék listázás és keresés (index.php)** 36](#_Toc195559089)

[**7.16. Algoritmusok folyamatábrákkal** 37](#_Toc195559090)

[**8.1. Bevezetés** 39](#_Toc195559091)

[**8.2. Tesztelési módszerek** 42](#_Toc195559092)

[**8.3 Fehér doboz tesztelés** 42](#_Toc195559093)

[**8.4. Tesztelési szintek** 43](#_Toc195559094)

[**8.5 Integrációs tesztelés** 43](#_Toc195559095)

[**8.6 Rendszertesztelés** 43](#_Toc195559096)

[**8.7. Tesztelési technikák** 44](#_Toc195559097)

[**8.8 Ekvivalencia particionálás** 44](#_Toc195559098)

[**8.9 Állapotgép-alapú tesztelés** 44](#_Toc195559099)

[**8.10. Tesztkörnyezet és eredmények** 44](#_Toc195559100)

[**8.11. Összegzés és javaslatok** 44](#_Toc195559101)

[**9. Fejlesztés** 45](#_Toc195559102)

[**9.1 Tényleges rendelés** 45](#_Toc195559103)

[**9.2 Számla kiállítás** 45](#_Toc195559104)

[**9.3 Email cím megerősítés** 45](#_Toc195559105)

[**9.4 Több termék feltöltés a webshopba** 45](#_Toc195559106)

[**9.5 Szállítási cím változtatás** 45](#_Toc195559107)

[**9.6 Admin felület fejlesztése** 45](#_Toc195559108)

[**10. Összegzés** 46](#_Toc195559109)

[**10.1 Szakmai fejlődés** 46](#_Toc195559110)

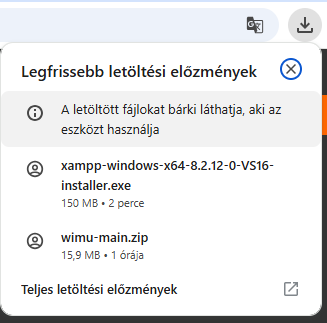
[**10.2 Jövőbeli célok** 47](#_Toc195559111)

[**11. Források** 47](#_Toc195559112)

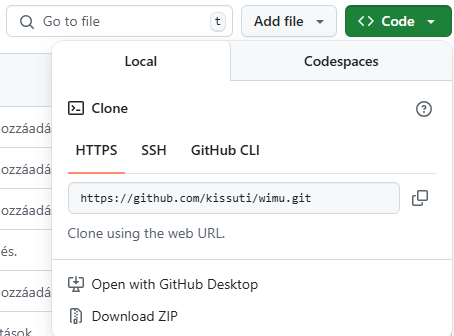
# **Felhasználói dokumentáció**

## **1.1 Letöltés**

**XAMPP letöltése**: <https://www.apachefriends.org/hu/index.html>  


A felhasználó operációs rendszerétől függően válassza ki a megfelelő letöltést. 

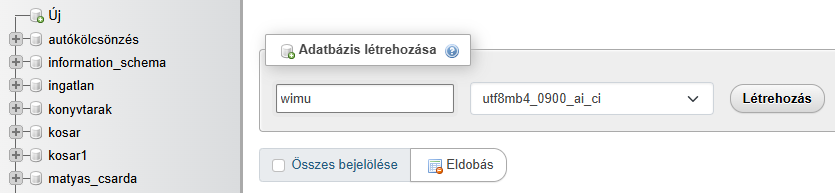
A letöltési legördülő menüben válassza ki a *XAMPP* telepítőjét, majd az alapbeállításokon telepítse fel a számítógépre.

A GitHub-ról töltse le a felhasználó a repository-t innen:

Letöltés után a felhasználó nevezze át a mappát erre: *wimu*

**1.2 Adatbázis és fájl elhelyezés**

El kell indítani az Apache-ot és a MySQL-t. Majd navigáljon a felhasználó el a fájlkezelőben ide:  
*C:/xampp/htdocs/*.  
Csomagolja ki a felhasználó a betömörített mappát és illessze be a fent említett helyre.  
A címsorba írja be: *localhost/phpmyadmin*

  
A *phpmyadmin* oldalán létre kell hozni a felhasználónak egy *wimu* nevezetű adatbázist.  
Az adatbázis létrehozásához kattintson rá a bal felül található „Új” gombra, majd az adatbázis nevének a *wimu* nevet adja.  
Létrehozás után, kattintson rá bal oldalon található már létrehozott *wimu* adatbázisra.



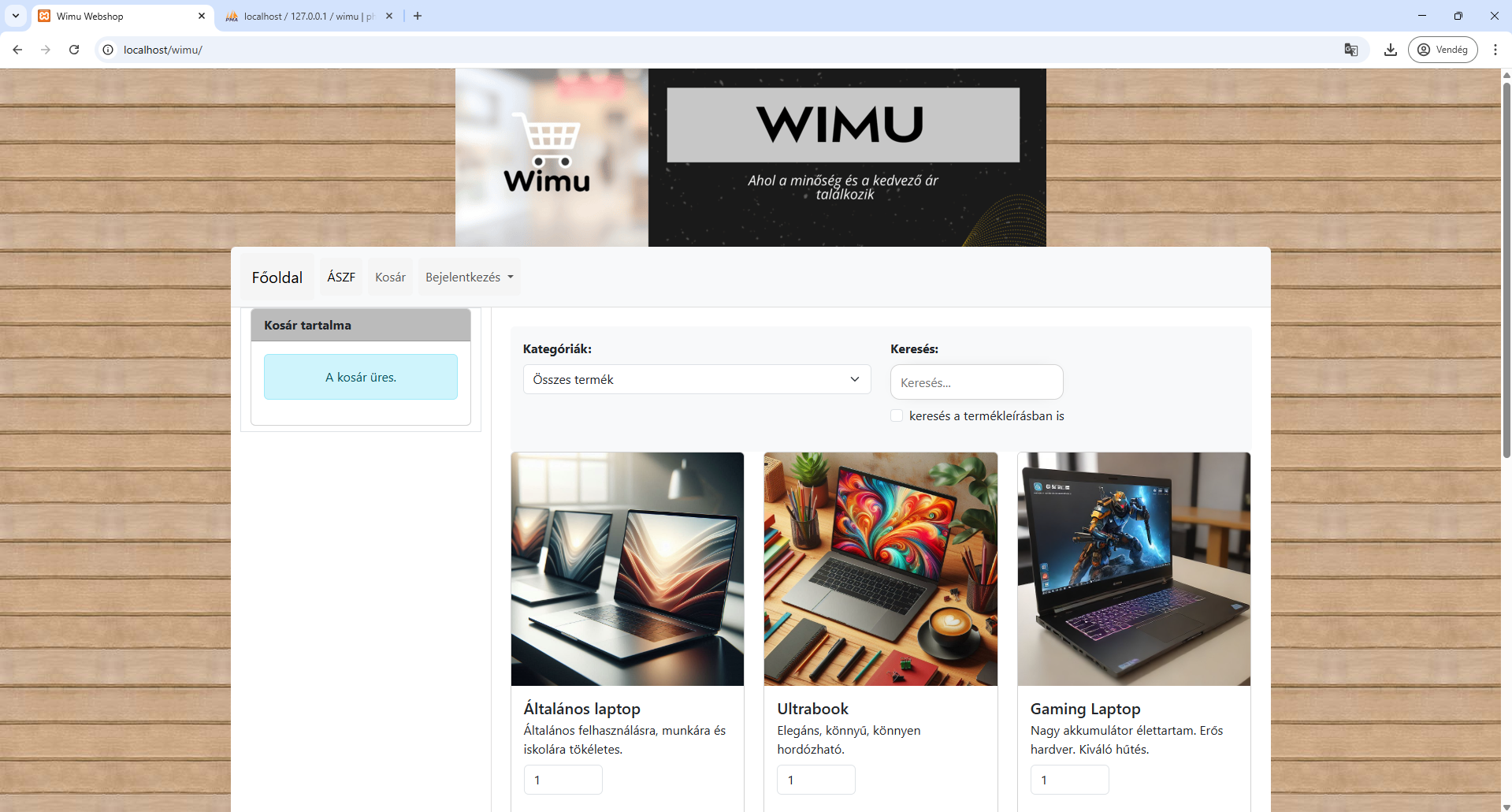
A felhasználó kattintson az *SQL* gombra.  


Az adatbázist a letöltött mappában találhatja a felhasználó és illessze be: *wimu.sql*

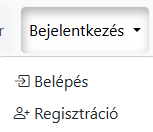
Majd nyomjon rá az *Inditás* gombra.

* 1. **Weboldal használata**

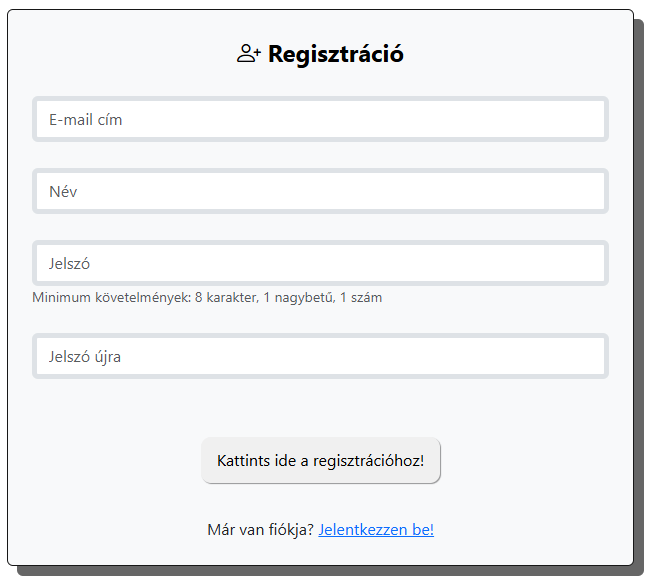
A felhasználó nyisson egy új lapot a böngészőben és írja be az alábbi címet, hogy elérhesse a weboldalt: *localhost/wimu*



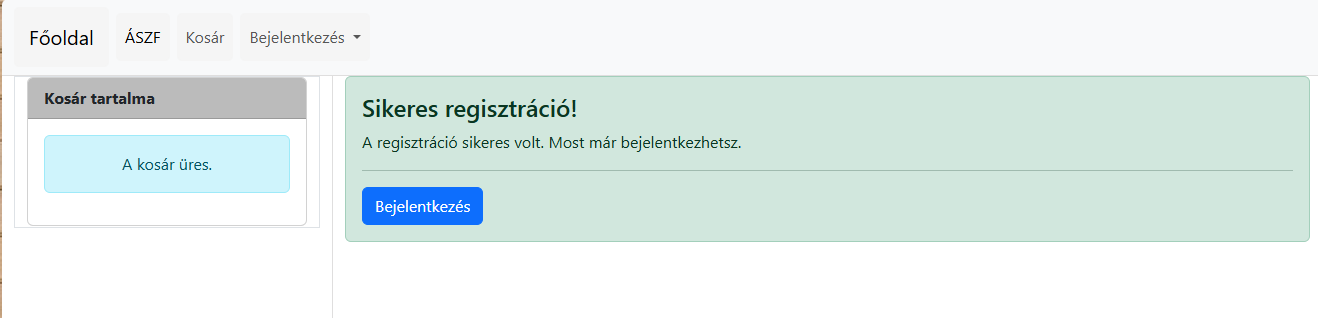
**Regisztráció:**



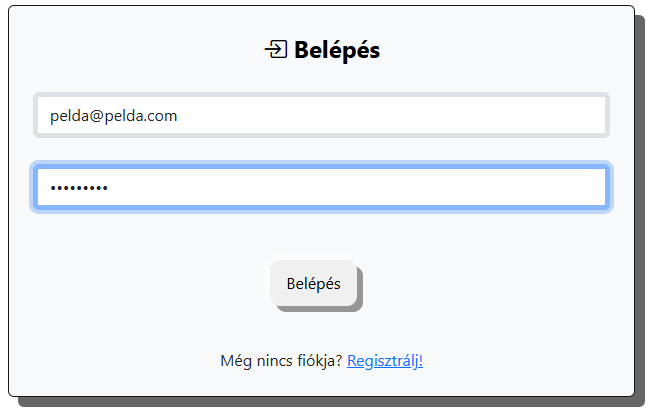
A legördülő menüben „Regisztráció” gomb megnyomásával hozhat létre felhasználót magának.

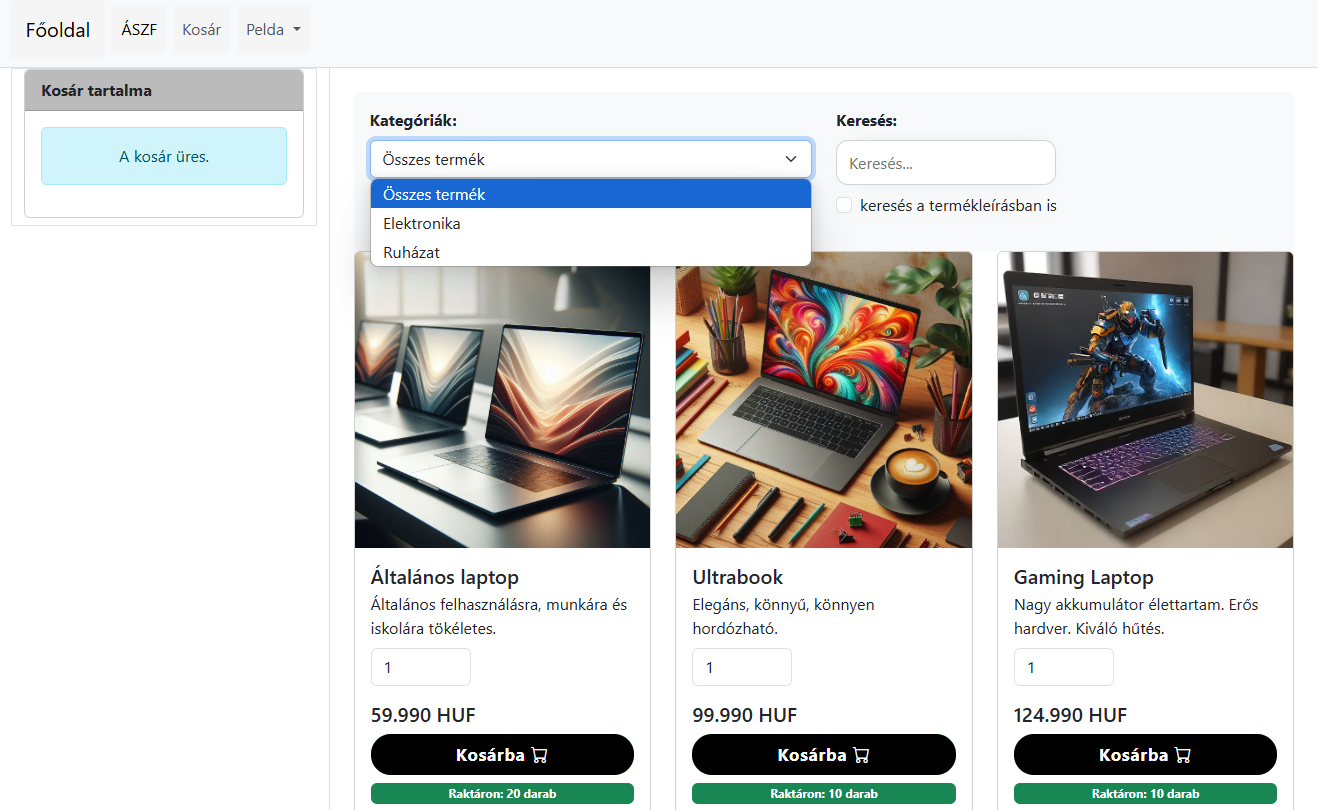


Az adatok kitöltése után regisztrálhat a gomb megnyomásával.

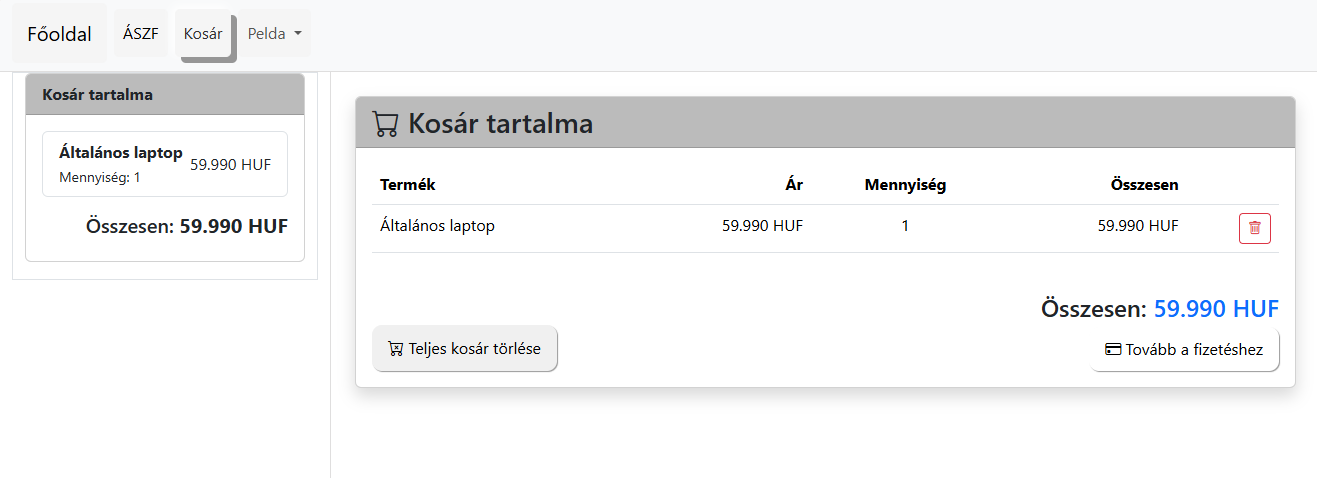


Majd ezzel az üzenettel jelzi a weboldal, hogy sikeresen regisztrált, aztán be is jelentkezhet a felhasználó az új fiókjával.





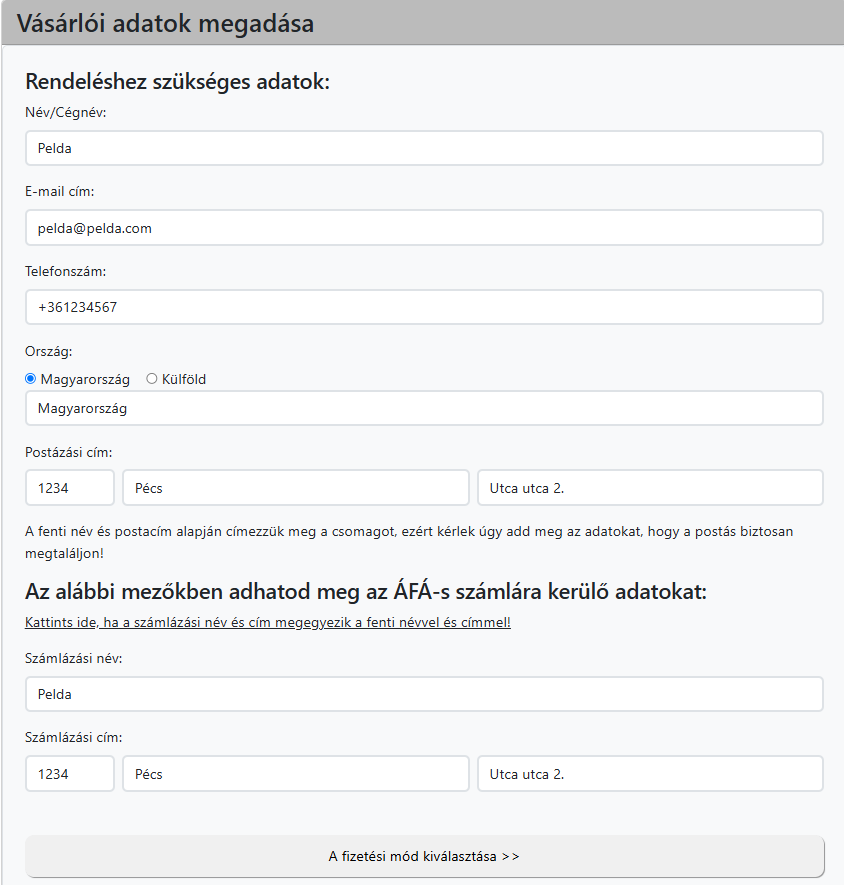
A felhasználó tud keresni a kínálatok között kategóriákra bontva, valamint név szerint is.  
Ha a felhasználónak megtetszett egy termék, akkor „Kosárba” gomb megnyomásával a kosárba helyezheti.



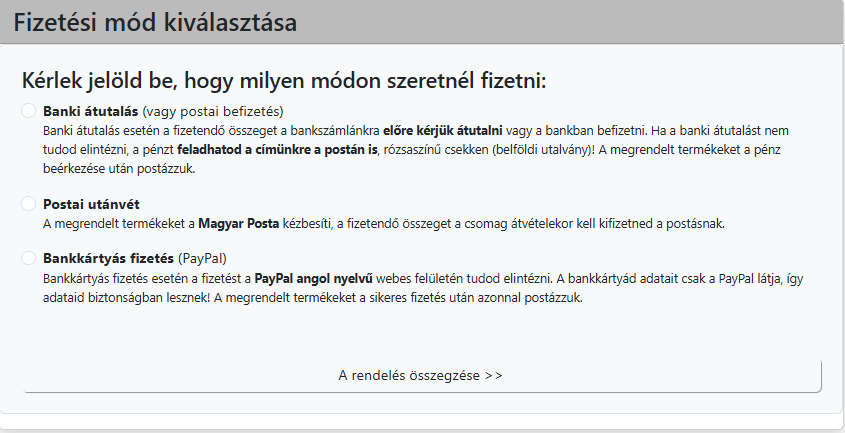
A menüsoron a „Kosár” gomb megnyomásával elnavigálhat a kosarának a tartalmához.  
Itt tudja a felhasználó megtekinteni a kosarának a tartalmát. A kosarat ki is lehet üríteni a bal alsó sarokban lévő gombbal, de ha csak egy bizonyos terméket szeretne eltávolítani, akkor a jobb oldalon található piros gomb megnyomásával tudja ezt végrehajtani. Amennyiben a kívánt termékek szerepelnek a felhasználó kosarában, „Tovább a fizetéshez” gomb megnyomásával haladhat tovább.



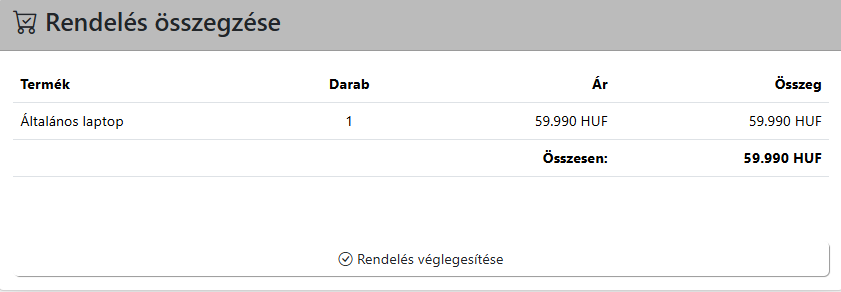
A mennyiség megváltoztatására is van lehetőség.



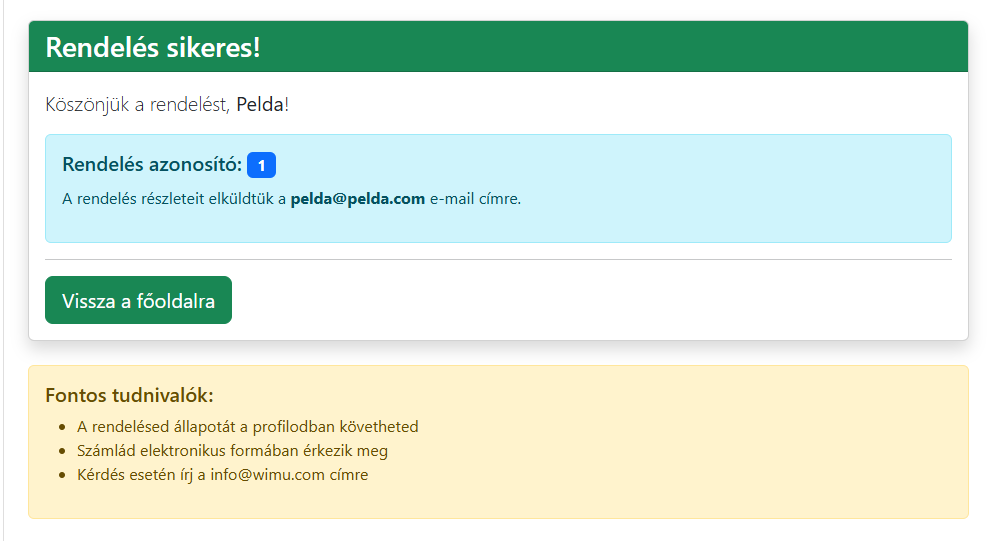
A felhasználónak itt ki kell tölteni a személyes információival.



Itt a felhasználónak az alábbi fizetési opciók közül kell választania.



A felhasználó át tudja tekinteni a végleges rendelését és a lenti gombbal tudja véglegesíteni azt, ami majd a rendszerbe felkerül.



A rendelés végén kap egy visszajelzést az oldaltól, hogy a rendelés sikeresen megtörtént és felkerült az adatbázisba. (számla fejlesztés alatt)

# **2. Bevezető**

## **2.1 Projekt célja és motiváció**

A Wimu Webshop létrehozásával célunk egy egyszerű, ugyanakkor hatékony online vásárlási platform fejlesztése volt, amelyen keresztül elsajátíthattuk a webalkalmazások teljes életciklusát. A projektet azért választottuk, mert lehetőséget biztosított számunkra a gyakorlati ismeretek mélyítésére, az adatbázis-tervezéstől kezdve a felhasználói felület optimalizálásán át egészen a modern webfejlesztési technológiák alkalmazásáig. A kezdeti technikai kihívások – mint például a kosárkezelés és a session-kezelés – leküzdése után sikeresen implementáltunk összetettebb funkciókat is, többek között különböző fizetési módok integrálását, valamint a raktárkészlet pontos nyilvántartását. A fejlesztési folyamat során kiemelt figyelmet fordítottunk a skálázhatóságra, a biztonságos adatkezelésre, valamint a felhasználói élmény folyamatos javítására, hogy a végeredmény egy professzionális és megbízható webshop legyen.

## **2.2 Tanultak és új ismeretek**

*A fejlesztés során elsajátítottuk:*

**Adatbázis-kezelés**: MySQL táblák tervezése (pl. termékek, felhasználók, rendelések).

**Biztonsági technikák**: Jelszavak titkosítása (Hash), SQL injection elleni védelem.

**Folyamatkezelés**: Felhasználói regisztráció, kosár tartalmának mentése, rendelés véglegesítése.

**Fizetési integráció**: PayPal SDK használata bankkártyás tranzakciókhoz.

**Reszponzív design**: Bootstrap segítségével mobilbarát felületek kialakítása.

## **2.3 Későbbi tervek**

**Jövőbeli fejlesztési céljaink**:

**Email-es regisztráció**: Jelszó-visszaállítás és regisztráció megerősítése.

**Rendeléskövetés**: Felhasználók számára áttekinthető szállítási állapot.

**Admin felület**: Termék- és rendeléskezelés intuitív vezérlőpultja.

**Értékelési rendszer**: Vásárlói visszajelzések megjelenítése.

## **2.4 Csapatmunka és szerepkörök**

*A csapatban a feladatok így oszlottak meg:*

**Csaba:** Backend fejlesztés (PHP, adatbázis logika, rendelésfeldolgozás).

**Rajmund:** Frontend tervezés (HTML/CSS, Bootstrap).

A kommunikációhoz Discordot használtunk. A kódot GitHubon tároltuk.

*Projektszervezési Eszközök*

**Discord (Kommunikáció):** Feladatok szétosztása és állapotuk nyomon követése (pl. "Kosár implementálása", "Fizetési módok hozzáadása").

**GitHub:** Kódmegosztás és együttműködés (branch-ek használata).

**Messenger:** Megbeszéltük, hogymikor csináljuk a vizsgaremeket.

## **2.5 Összefoglalás**

A Wimu Webshop projekt lehetővé tette, hogy egy valós alkalmazás fejlesztésének minden szakaszában részt vegyünk. A nehézségek (pl. adatbázis-frissítések szinkronizálása) megoldása közben mélyebb betekintést kaptunk a webes rendszerek működésébe. A projekt továbbfejlesztésével célunk, hogy egy teljes körű, felhasználóbarát webshoppá nőjön ki.

# **3. Témaválasztás:**

*Miért ezt választottuk?*

Én Csaba azért ezt szerettem volna választani, mert nagyon sok webshop van a világban, de én is meg akartam nézni a folyamatát, hogy hogyan épül fel egy webshop.

Én Rajmund azért ezt választottam, mert mindig is érdekelt az üzleti oldala a dolgoknak, illetve mindig akartam egy saját boltot/webshopot nyitni.

## **3.1 Közös cél: egy életképes prototípus létrehozása**

Bár különböző motivációk vezettek minket, a cél egyértelmű volt: egy működő webshop prototípus, ami:

**Technikailag megalapozott**: Biztonságos, skálázható, könnyen bővíthető.

**Felhasználóbarát**: Intuitív navigáció, reszponzív design, gyors betöltés.

**Üzletileg releváns**: Valós igényeket szolgál (pl. címkezelés külföldre szállításhoz).

A projekt során megtanultuk, hogy a kommunikáció és a kompromisszumok nélkülözhetetlenek egy csapatban. Például amikor Rajmund egy összetett animációt akart a kosárhoz, de Csaba rámutatott, hogy az lelassítaná az oldalt, kompromisszumként egyszerűbb, de hatékony megoldást választottunk.

## **3.2 Mit hoz a jővő?**

A webshop jelenleg egy **MVP** (Minimum Viable Product), de alapozóként szolgál további fejlesztésekhez:

**Email-es regisztráció**: Jelszó-visszaállítás és automatikus értesítések implementálása.

**Analitika**: Google Analytics integráció a vásárlói viselkedés nyomon követésére.

**SEO-optimalizálás**: A termékoldalak tartalmának finomhangolása keresőmotorok számára.

Csaba számára ez a projekt egy technikai referenciapont, Rajmundnak pedig egy üzleti alapkő. Mindketten abban reménykedünk, hogy a jövőben ezt a prototípust egy teljes értékű online vállalkozássá lehet fejleszteni.

# **4. Az alkalmazott fejlesztői eszközök:**

## **4.1. Programozási és leíró nyelvek & keretrendszerek**

**PHP**: A backend logika (pl. kosárkezelés, felhasználókezelés) megvalósítása.

**MySQL:** Relációs adatbázis a termékek, felhasználók és rendelések tárolására.

**HTML/CSS/JavaScript:** Frontend felület reszponzív design-nal és interaktív elemekkel.

**Bootstrap 5**: Gyors és egységes felhasználói felület kialakítása (pl. kártyák, navigációs sáv).

## **4.2. Fejlesztői környezetek & eszközök**

**Visual Studio Code**: Fő IDE kódírásra, debugolásra (PHP, HTML, CSS, JS).

**XAMPP**: Lokális szerver környezet (Apache, MySQL, PHP) teszteléshez.

**phpMyAdmin**: Adatbázisok kezelése, SQL lekérdezések futtatása.

**Git & GitHub**: Verziókövetés és csapatmunka koordinálása.

## **4.3. Design & Médiaeszközök**

**PiktoChart**: Banner szerkesztése (pl. img/banner2.png).

**Leonardo.ai, MS Copilot**: Termék képek

**Bootstrap Icons**: Ikongyűjtemény a felhasználói felülethez.

## **4.4. Könyvtárak & Függőségek**

**jQuery:** Dinamikus űrlapkezelés és AJAX kérések (pl. kosár frissítése).

**Popper.js & Bootstrap JS**: Interaktív elemek (dropdown menük, tooltip-ek).

## **4.5. Egyéb eszközök**

**Messenger/Discord**: Feladatok szétosztása és állapotkövetés.

**Google Chrome DevTools**: Frontend hibakeresés.

## **4.6. Biztonsági eszközök**

**BCrypt:** Jelszavak titkosítása (password\_hash() és password\_verify()).

**Prepared Statements**: SQL injection elleni védelem (pl. mysqli->prepare()).

**Session Hardening**: Cookie-k biztonságos beállításai (httponly, secure flag-ek).

## **4.7. Kompatibilitás & tesztelés**

**Cross-Browser Tesztelés**: Chrome, Firefox, Safari.

**Reszponzív Design Tesztelés**: Mobil-, tablet- és asztali nézet.

# **5. Tervezési módszer**

## **5.1 Alkalmazás tervezése:**

A Wimu Webshop tervezése során a felhasználói igények és üzleti célok összehangolására fókuszáltunk. A tervezés három fő szakaszra oszlott:

**Követelményelemzés**: A vásárlók és adminisztrátorok igényeinek feltárása (pl. egyszerű regisztráció, kosárkezelés, rendeléskövetés).

**Rendszerfelépítés**: A háromrétegű modell (frontend, backend, adatbázis) kialakítása.

*Funkcionális specifikációk:*

Felhasználói szerepkörök (vendég, regisztrált felhasználó, admin).

Modulok (termékkatalógus, kosár, fizetési rendszer).

A tervezés során kiemelt szerepet kapott a reszponzív design, hogy a webshop minden eszközön (mobil, tablet, asztali) optimálisan működjön.

## **5.2 Tervezési módszertan:**

A projekt agilis módszertant követett, amely lehetővé tette a folyamatos visszajelzés alapján történő iteratív fejlesztést.

**Csapatmunka**: Megbeszélés alapján összedolgoztunk, hogy mikor ér rá a másik kolléga.

**Felhasználó tesztelés**: Felhasználóként bejelentkeztünk leteszteltünk főbb funkciókat.

## **5.3 Vizuális modellezés:**

A rendszer vizuális reprezentációja kulcsfontosságú volt a tervezésben:

*Bootstrap-ek:*

Főoldal elrendezése (termékkártyák, navigációs sáv).

Kosár oldal felépítése (táblázatos nézet, gombok).

*Adatbázis séma tervezése:*

ER-diagram készült a táblák és kapcsolatok szemléltetésére (pl. arucikk ↔ kategoriak, ugyfel ↔ rendelesek).

Normalizálás az ismétlődések elkerülése végett.

*Folyamatábrák:*

Belépési és rendelési folyamatok vizualizációja.

## **5.4 OOP megvalósulása:**

Bár a projekt főként PHP-t használ, objektumorientált elemeket is építettünk bele:

**Adatbázis kezelés OOP stílusban**:

A mysqli osztály használata prepared statement-ekkel:

$parancs = $kapcsolat->prepare("SELECT \* FROM ugyfel WHERE email=?");

$parancs->bind\_param("s", $email);

$parancs->execute();

Transaction kezelés a rendeléseknél

**Osztálytervek jövőbeli bővítéshez**:

**User osztály**: Felhasználói adatok és jogosultságok kezelése.

**Cart osztály**: Kosár műveletek (hozzáadás, törlés, frissítés).

**Product osztály**: Termékek lekérdezése és szűrése.

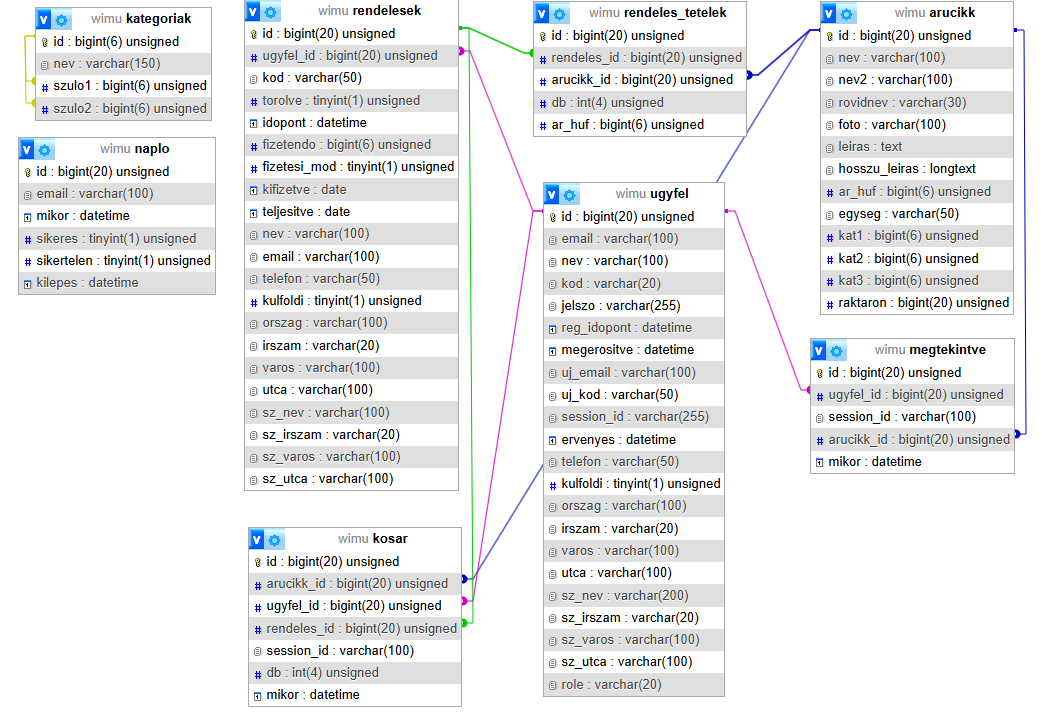
*Öröklés és egységbezárás:*

A Payment főosztályból származó alosztályok (pl. PayPalPayment, BankTransfer) a fizetési módok kezelésére. (fejlesztés alatt)

Privát metódusok biztonságos adatkezeléshez (pl. jelszó titkosítás).

**MySql**: Adatbáziskapcsolat egyszeres példányosítása.

# **6. Az adatbázis ábrája:**



## **6.1 Főbb táblák:**

* **Termékek** (arucikk)
  + Adatok: név, ár, készlet, kategóriák (3 szint), leírások.
  + Példa: Laptopok, workstationok.
* **Kategóriák** (kategoriak)
  + Hierarchikus struktúra (pl. Elektronika → Laptopok → Gaming).
  + Szülő-mezőkkel (szulo1, szulo2).
* **Felhasználók** (ugyfel)
  + Regisztráció, címek, titkosított jelszó (BCrypt), admin jogok (role).
* **Kosár** (kosar)
  + Session vagy felhasználóhoz kötött, ideiglenes adatok a rendelés előtt.
* **Rendelések** (rendelesek + rendeles\_tetelek)
  + Fizetési módok, címek, egyedi kód (67e7c1d056397).
  + Tételek: termék ID, darabszám, ár.



## **6.2 Fő funkciók:**

* **Többszintű kategóriák**: Szűrés és navigáció egyszerűsítése.
* **Kosárkezelés**: Session-alapú vagy felhasználói fiókhoz kötött.
* **Rendelésfeldolgozás**: Szállítási/számlázási címek, külföldi támogatás.
* **Biztonság:** Titkosított jelszavak, bejelentkezési napló (naplo).

## **6.3 Táblák kapcsolatai**

Az adatbázis relációit idegen kulcsok (Foreign Keys) biztosítják:

*Termék → Kategóriák:*

ALTER TABLE arucikk ADD FOREIGN KEY (kat1) REFERENCES kategoriak(id);

Egy termék legfeljebb 3 kategóriába sorolható (kat1, kat2, kat3).

*Kosár → Termékek:*

ALTER TABLE kosar ADD FOREIGN KEY (arucikk\_id) REFERENCES arucikk(id);

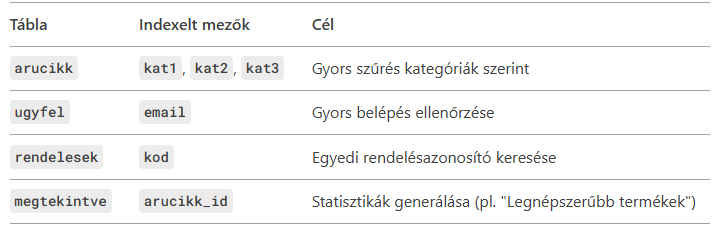
Egy kosár tétel mindig egy létező termékre hivatkozik.

*Rendelések → Felhasználók:*

ALTER TABLE rendelesek ADD FOREIGN KEY (ugyfel\_id) REFERENCES ugyfel(id);

Minden rendelés egy regisztrált felhasználóhoz tartozik.

## **6.4 Indexek és optimalizálás**

A teljesítmény növelése érdekében stratégiai indexek lettek létrehozva:

## **6.5 Biztonsági intézkedések**

**Jelszavak titkosítása**:

A ugyfel.jelszo mezőben BCrypt hashelés kerül használatra.

Példa: $2y$10$... – a 10 a titkosítási költség faktort jelöli.

**SQL Injection védelem**:

*Prepared statement-ek használata PHP-ben:*

$parancs = $kapcsolat->prepare("SELECT \* FROM ugyfel WHERE email=?");

$parancs->bind\_param("s", $email);

*Munkamenet kezelés:*

A ugyfel.session\_id és ervenyes mezők biztosítják a jogosulatlan hozzáférés kiszűrését.

## **6.6 Jelentős funkciók és adatbázis integráció**

*Többszintű kategóriák*

**Megvalósítás**: A kategoriak tábla hierarchikus struktúrája lehetővé teszi:

Akár 3 szintű kategória fa (pl. Elektronika → Laptopok → Gaming).

Rugalmas szűrés a terméklistákban (kat1, kat2, kat3 mezők).

*Kosárkezelés*

Vendég vásárlók: A session\_id menti a kosarat 7 napig (DELETE FROM kosar WHERE mikor<'$egyhete').

Készlet ellenőrzés: A kosar és arucikk táblák tranzakcióban frissülnek, hogy elkerüljük a túlértékesítést.

*Rendeléskövetés*

Státuszok: A rendelesek tábla fizetesi\_mod és teljesitve mezői nyomon követik a fizetést és szállítást.

## **6.7 Későbbi fejlesztések**

**Elavult kosarak törlése**: Heti automatikus törlés cron job segítségével.

**Full-text keresés**: Az arucikk táblában a termékleírások gyors keresése.

**Többnyelvű támogatás**: Új mezők (pl. leiras\_en) a lokalizációhoz.

Ez az adatbázis szerkezet biztosítja a Wimu Webshop skálázhatóságát és a jövőbeli bővítések lehetőségét.

# **7. Részletes feladatspecifikáció, algoritmusok**

## **7.1. belepes.php – Belépési folyamat**

Funkció: Felhasználó hitelesítése e-mail és jelszó alapján.

*Paraméterek:*

$\_POST['email']: Felhasználó e-mail címe.

$\_POST['jelszo']: Felhasználó jelszava.



**Működés**:

JavaScript validálja az e-mail formátumot és a jelszó meglétét.

Szerver oldalon ellenőrzi az adatbázisban a felhasználó létezését (SELECT \* FROM ugyfel WHERE email='$email').

Hash-elt jelszó ellenőrzése password\_verify() függvénnyel.

Naplózza a sikertelen próbálkozásokat (napló tábla).

Ha 3 sikertelen próbálkozás van 30 percen belül, tiltja a belépést.

Sikeres belépés esetén beállítja a session-t és cookie-kat, átirányít a főoldalra.

Visszatérési érték: HTTP átirányítás hibákra (hiba=1, tilos=1) vagy sikeres belépés esetén index.php.

## **7.2 Főbb függvények és metódusok specifikációja**

*1. Kosár tartalmának betöltése (1\_kosar\_tartalma.php)*

**Funkció**: A felhasználó kosarában lévő termékek megjelenítése, összegzés.



*Paraméterek:*

session\_id: A felhasználó session azonosítója (ha nincs bejelentkezve).

ugyfel\_id: A bejelentkezett felhasználó azonosítója.

*Adatbázis lekérdezés:*

SELECT \* FROM kosar WHERE session\_id = ? OR ugyfel\_id = ?;

**Kimenet**: HTML tábla a termékekkel, mennyiséggel és összesített árakkal.

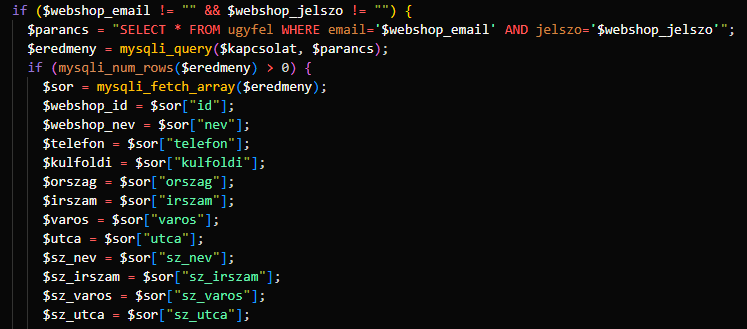
## **7.3 Vásárlói adatok validálása (2\_vasarloi\_adatok.php)**

*JavaScript függvény* ellenoriz()*:*

**Funkció**: Ellenőrzi, hogy minden kötelező mező ki van-e töltve.

**Ellenőrzött mezők**: Név, telefonszám, cím, számlázási adatok.

**Hibakezelés**: Alert üzenetek üres mezők esetén.



**Adatbázis frissítés**:

UPDATE ugyfel SET nev=?, telefon=?, ... WHERE id=?;

Paraméterek: Felhasználó által megadott adatok (pl. $\_POST['nev'], $\_POST['telefon']).

## **7.4. Fizetési mód kiválasztása (3\_fizetesi\_modok.php)**

JavaScript függvény ellenoriz():

**Funkció**: Ellenőrzi, hogy legalább egy fizetési mód ki van-e választva.

**Integráció**: PayPal SDK használata bankkártyás fizetéshez.

**Konverzió**: HUF → USD fix árfolyammal (pl. osszeg / 350).

*Adatátvitel:*

<input type="hidden" name="fizet" value="1|2|3">

Értékek: 1 (banki átutalás), 2 (utánvét), 3 (PayPal).



## **7.5. Rendelés összegzése (4\_rendeles\_osszegzes.php)**

**Funkció**: Megjeleníti a felhasználó kosarát és a kiválasztott fizetési módot.

*Adatbázis lekérdezés:*

SELECT arucikk.nev, kosar.db, arucikk.ar\_huf

FROM kosar

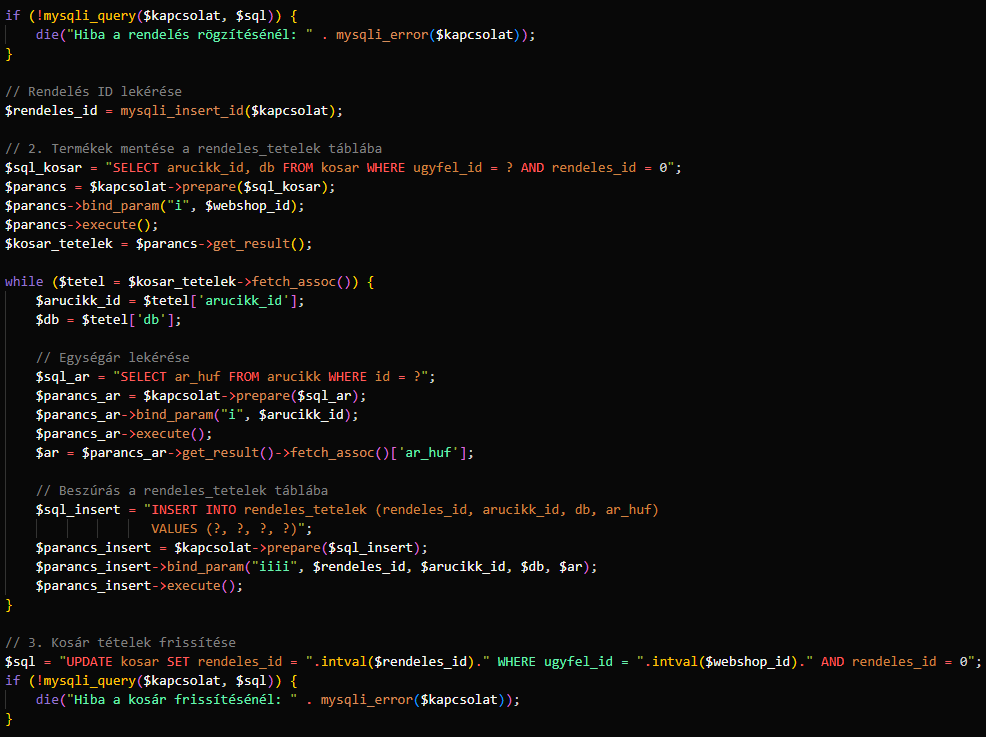
JOIN arucikk ON kosar.arucikk\_id = arucikk.id

WHERE ugyfel\_id = ?;



**Kimenet**: Tábla a termékek részleteivel és egy véglegesítő gomb.

## **7.6 Rendelés feldolgozása (5\_rendeles\_elkuld.php)**

****

*Lépések:*

*Rendelés rögzítése:*

INSERT INTO rendelesek (ugyfel\_id, kod, fizetendo, ...);

*Termékek mentése:*

INSERT INTO rendeles\_tetelek (rendeles\_id, arucikk\_id, db, ar\_huf);

*Raktárkészlet frissítése:*

UPDATE arucikk SET raktaron = raktaron - ? WHERE id = ?;

*Kosár ürítése:*

DELETE FROM kosar WHERE ugyfel\_id = ?;

**Kimenet**: Sikeres rendelés üzenet és visszaigazoló e-mail szimuláció.

## **7.7. Bejelentkezési rendszer (belepes.php)**

**Funkció**: Felhasználó hitelesítése és session kezelése.

*Ellenőrzések:*

**Jelszó ellenőrzés**: password\_verify($jelszo, $sor['jelszo']).

*Brute force védelem:*

SELECT COUNT(\*) FROM naplo WHERE email=? AND mikor > ? AND sikertelen=1;

Blokkolás, ha 3+ sikertelen próbálkozás van 30 percen belül.

**Cookie-k:** webshop\_email és webshop\_jelszo 7 napig érvényesek.

## **7.8 Algoritmusok leírása**

**1. Kosár frissítési algoritmus**

Cél: A kosárban lévő termékek mennyiségének módosítása.

*Lépések:*

A felhasználó módosítja a mennyiséget a kosár oldalon.

Űrlap elküldése a mit=modosit paraméterrel.

*Adatbázis frissítés:*

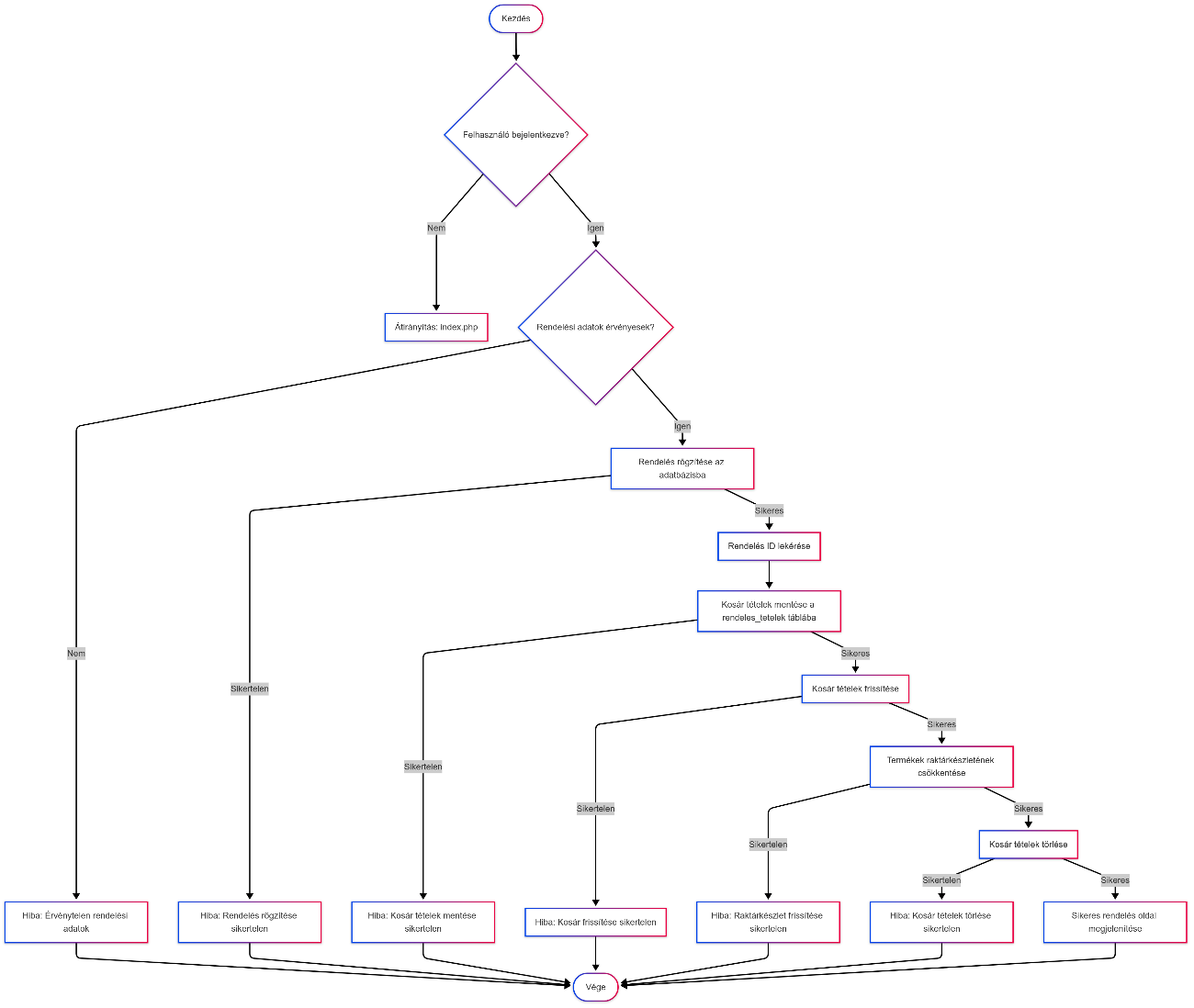
foreach ($\_POST as $kosar\_id => $db) {

UPDATE kosar SET db = ? WHERE id = ?;

}

Összeg újra számolása és átirányítás.

## **7.9 Rendelésfeldolgozás folyamata**



## **7.10. Termékkereső algoritmus (index.php)**

*Megjegyzések:*

**Biztonsági hiányosságok:** SQL injection kockázat néhány lekérdezésben (pl. 5\_rendeles\_elkuld.php).

Jelszavak tárolása nyers formában helyett password\_hash() használata ajánlott.

**Optimalizálási lehetőségek:**

Tranzakciók használata rendelésfeldolgozásnál (ACID tulajdonságok biztosítása).

Cache-elés a gyakran lekérdezett termékadatokhoz.

## **7.11. Kosár megtekintése (kosar\_megtekintes.php)**

**Funkció**: A felhasználó kosarában lévő termékek megjelenítése, törlési műveletek kezelése, összegzés.

*Paraméterek*:

$\_POST['torles']: "egy" vagy "osszes" érték (egy elem vagy teljes kosár törlése).

$\_POST['arucikk\_id']: Törlendő termék azonosítója.

$\_POST['db']: Törlendő mennyiség (csak egy elem törlése esetén).

*Működés:*

Session/Cookie alapján felhasználó azonosítása:

Ha bejelentkezett (ugyfel\_id), különben session\_id használata.

*Kosár tartalmának lekérdezése:*

SELECT \* FROM kosar

WHERE (ugyfel\_id = ? OR session\_id = ?) AND rendeles\_id = 0;

*Raktár visszaállítás minden tételre:*

UPDATE arucikk SET raktaron = raktaron + ? WHERE id = ?;

*Kosár ürítése:*

DELETE FROM kosar WHERE (ugyfel\_id = ? OR session\_id = ?);

*Raktár visszaállítás:*

UPDATE arucikk SET raktaron = raktaron + ? WHERE id = ?;

*Tétel eltávolítása:*

DELETE FROM kosar WHERE (ugyfel\_id = ? OR session\_id = ?) AND arucikk\_id = ? LIMIT 1;

**Kimenet**: HTML tábla a kosár tartalmával és összesített árral.

## **7.12. Termék kosárba tétele (kosarba\_tesz.php)**

**Funkció**: Termék hozzáadása a kosárhoz vagy mennyiség frissítése.

*Paraméterek:*

$\_POST['arucikk\_id']: Termék azonosítója.

$\_POST['db']: Hozzáadandó mennyiség.

*Készlet ellenőrzése:*

SELECT raktaron FROM arucikk WHERE id = ?;

Ha kert\_mennyiseg > raktaron: hibaüzenet.

Felhasználó azonosítása: Bejelentkezett (ugyfel\_id) vagy vendég (session\_id).

*Ha a termék már szerepel a kosárban:*

UPDATE kosar SET db = db + ? WHERE (ugyfel\_id = ? OR session\_id = ?) AND arucikk\_id = ?;

*Új termék esetén:*

INSERT INTO kosar (arucikk\_id, ugyfel\_id/session\_id, db) VALUES (?, ?, ?);

*Raktár frissítése:*

UPDATE arucikk SET raktaron = raktaron - ? WHERE id = ?;

**Visszatérés**: JSON válasz (success: true vagy hibaüzenet).

**7.13. Profil módosítás (profil\_modosit.php és profil\_modosit\_2.php)**

**Funkció**: Felhasználói adatok és jelszó módosítása.

*Paraméterek:*

$\_POST['nev']: Új név.

$\_POST['email']: Új email.

$\_POST['uj\_jelszo'] és $\_POST['uj\_jelszo\_2']: Új jelszó és megerősítés.

*JavaScript:*

*Üres mezők ellenőrzése.*

Email formátum ellenőrzése (helyescim()).

Jelszó egyezés ellenőrzése.

Szerver oldali ellenőrzés:

*Email egyediség:*

SELECT id FROM ugyfel WHERE email = ? AND id != ?;

Jelszó hash-elés: password\_hash().

*Adatbázis művelet:*

UPDATE ugyfel SET nev = ?, email = ?, jelszo = ? WHERE id = ?;

*Kimenet*:

Sikeres frissítés: $\_SESSION['siker'] üzenet.

Hiba: $\_SESSION['hiba'] üzenet.

## **7.14. Regisztráció (reg.php és reg\_ellenoriz.php)**

**Funkció**: Új felhasználó regisztrálása.

*Paraméterek:*

$\_POST['emailcim']: Felhasználó email címe.

$\_POST['nev']: Teljes név.

$\_POST['jelszo'] és $\_POST['jelszo2']: Jelszó és megerősítés.

Email formátum, jelszó egyezés, üres mezők.

*Jelszó komplexitás (8 karakter, nagybetű, szám):*

preg\_match('/^(?=.\*[A-Z])(?=.\*\d).{8,}$/', $jelszo);

*Email egyediség:*

SELECT \* FROM ugyfel WHERE email = ?;

*Adatbázis művelet:*

INSERT INTO ugyfel (email, nev, jelszo, reg\_idopont) VALUES (?, ?, ?, ?);

*Kimenet:*

**Sikeres regisztráció**: Átirányítás bejelentkezésre.

**Hiba**: Üzenet a hibáról (pl. "Email már foglalt").

## **7.15. Termék listázás és keresés (index.php)**

**Funkció**: Termékek szűrése kategóriák és keresési feltételek alapján.

*Paraméterek:*

$\_POST['kat1'], $\_POST['kat2'], $\_POST['kat3']: Kategóriák.

$\_POST['mitkeres']: Keresett kulcsszó.

$\_POST['sorrend']: Rendezési szempont (ár, név, népszerűség).

*Kategóriák hierarchikus szűrése:*

Ha kat3 > 0: kat1, kat2, és kat3 alapján.

Ha kat2 > 0: kat1 és kat2 alapján.

Alapértelmezett: kat1 vagy összes termék.

*Keresés a termék nevében, leírásában:*

SELECT \* FROM arucikk

WHERE (nev LIKE ? OR leiras LIKE ?)

AND katszuro\_feltetel

ORDER BY sorrend;

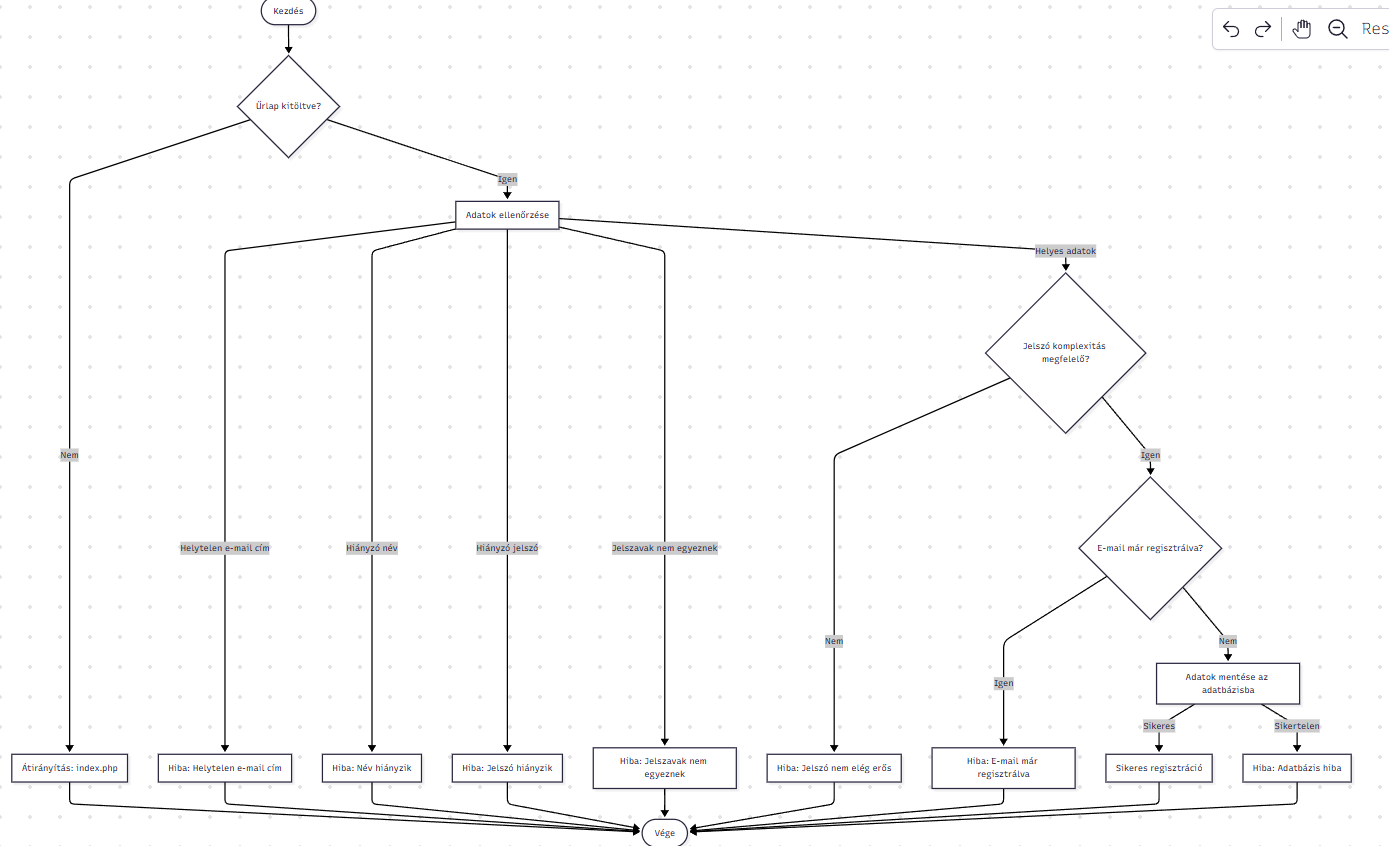
*Pagináció:*

oldal és laponként paraméterek kezelése.

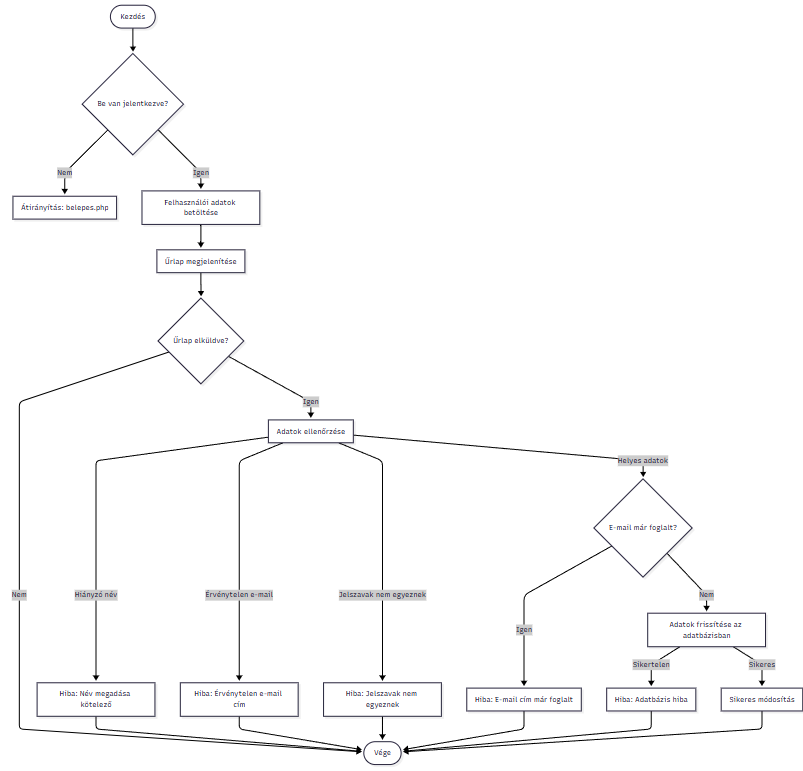
**Kimenet**: Termékek megjelenítése kártyákon, keresési eredmények frissítése.

## **7.16. Algoritmusok folyamatábrákkal**

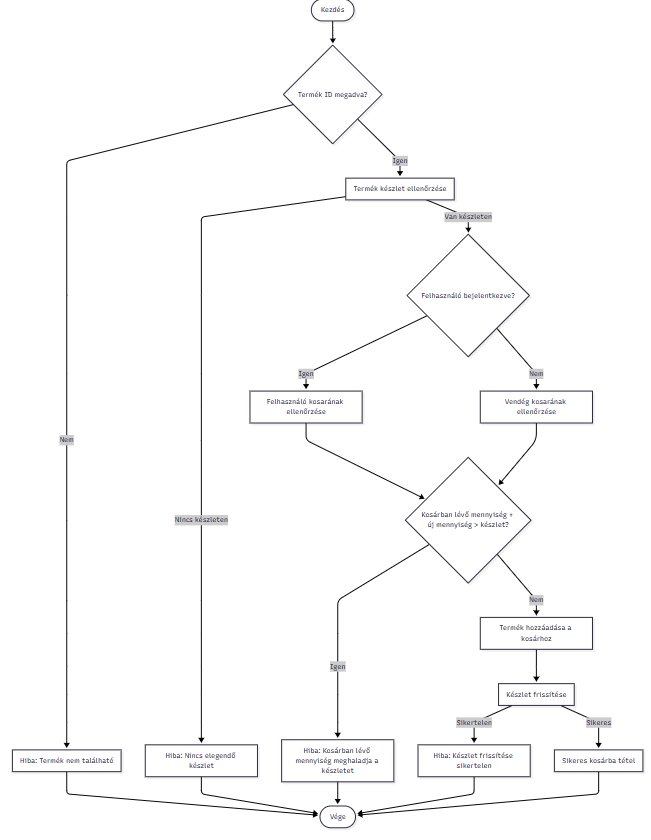
*1. Regisztrációs Folyamat:*



*2. Profil Módosítás:*



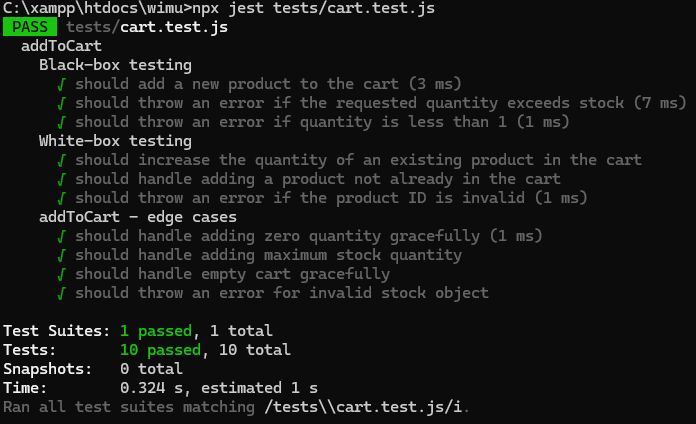
*3. Kosárba Tétel:*

**8. Unit teszt**

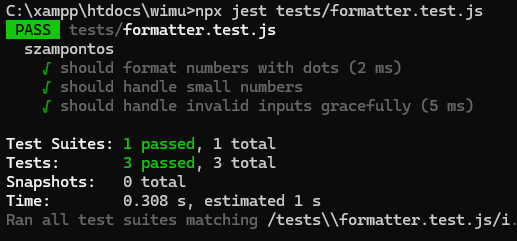
## **8.1. Bevezetés**

A dokumentáció ezen része bemutatja a különböző tesztelési módszereket, szinteket és technikákat egy webalkalmazás tesztelési folyamatában. A tesztelés a következő modulokra fókuszál:

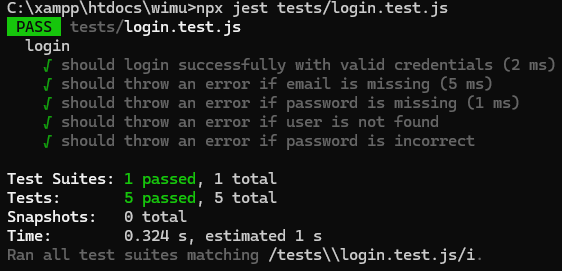
*Kosárkezelés* (cart.js)



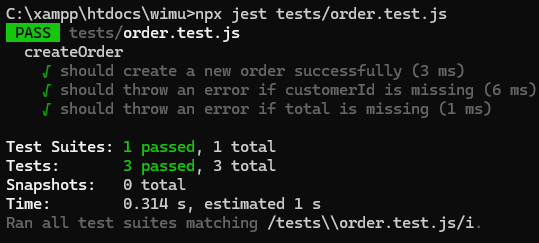
*Számformázás* (formatter.js)

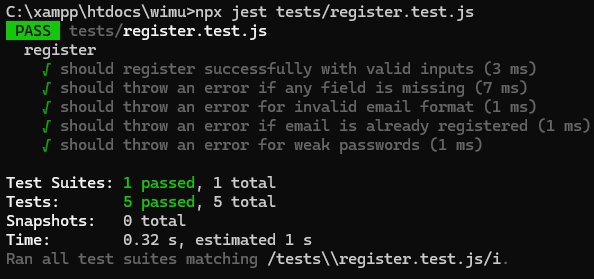


*Bejelentkezés* (login.js)

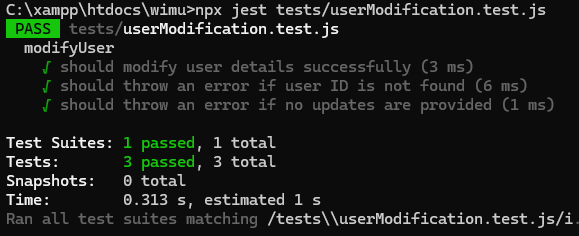


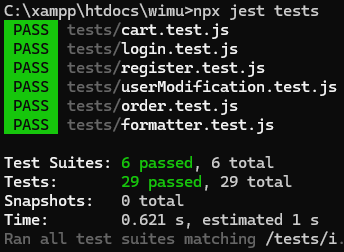
*Rendeléskezelés* (order.js)



*Regisztráció* (register.js)

*Felhasználómódosítás* (userModification.js)



*Teljes teszt futtatása:*

## **8.2. Tesztelési módszerek**

*Fekete Doboz Tesztelés*

A bemenet-kimenet viselkedést vizsgálja, anélkül, hogy a belső implementációt ismerné.

*Példák:*

*Kosárkezelés* (cart.test.js):

should add a new product to the cart: Ellenőrzi, hogy új termék hozzáadása esetén a kosár tartalmazza-e az új elemet.

should throw an error if quantity is less than 1: Ellenőrzi, hogy a rendszer hibát dob-e, ha a mennyiség érvénytelen.

*Bejelentkezés* (login.test.js):

should login successfully with valid credentials: Ellenőrzi, hogy érvényes hitelesítő adatok esetén sikeres bejelentkezés történik-e.

## **8.3 Fehér doboz tesztelés**

A belső logikára és struktúrára fókuszál, például feltételes ágak, ciklusok vagy hibakezelés tesztelésére.

*Példák:*

*Kosárkezelés* (cart.test.js):

should increase the quantity of an existing product in the cart: Ellenőrzi, hogy a rendszer frissíti-e a mennyiséget, ha a termék már szerepel a kosárban.

should throw an error if the product ID is invalid: Ellenőrzi a hibakezelést érvénytelen termékazonosító esetén.

*Számformázás (formatter.test.js):*

should handle negative numbers: Ellenőrzi, hogy a függvény helyesen formáz-e negatív számokat.

## **8.4. Tesztelési szintek**

*Unit Tesztelés*

Egyedi függvények vagy modulok tesztelése elszigetelten.

*Példák:*

Számformázás (formatter.test.js):

should format numbers with dots: Ellenőrzi a szampontos függvényt különböző bemenetekkel (pl. 1000 → 1.000).

*Regisztráció* (register.test.js):

should throw an error for weak passwords: Ellenőrzi a jelszóerősség követelményeit.

## **8.5 Integrációs tesztelés**

Modulok közötti interakciók tesztelése (jelenlegi példák főleg unit tesztek, de példa integrációs tesztre):

Rendelés létrehozása (order.test.js):

should create a new order successfully: Ellenőrzi, hogy a rendelés létrehozása frissíti-e a rendelések listáját.

## **8.6 Rendszertesztelés**

A teljes alkalmazás végpontokon keresztül történő tesztelése (nincs benne a példákban, de példa lehetne):

Bejelentkezés + Kosárkezelés:

Egy felhasználó bejelentkezik, hozzáad egy terméket a kosarához, majd létrehoz egy rendelést.

## **8.7. Tesztelési technikák**

*Határérték-elemzés*

**Kosárkezelés** (cart.test.js):

should handle adding maximum stock quantity: Ellenőrzi, hogy a rendszer engedélyezi-e a maximális raktárkészlet hozzáadását (pl. 10 darab).

## **8.8 Ekvivalencia particionálás**

*Bejelentkezés* (login.test.js):

**Érvényes partíció**: test@example.com + password123 → Sikeres bejelentkezés.

**Érvénytelen partíció**: unknown@example.com + bármilyen jelszó → "User not found" hiba.

## **8.9 Állapotgép-alapú tesztelés**

*Felhasználómódosítás* (userModification.test.js):

should modify user details successfully: Ellenőrzi, hogy a felhasználói adatok frissülnek-e a megfelelő állapotváltozással.

## **8.10. Tesztkörnyezet és eredmények**

**Keretrendszer**: Jest (JavaScript tesztelési keretrendszer).

**Tesztlefedettség**: A unit tesztek lefedik a kritikus funkciókat (pl. hibakezelés, formázás).

*Talált Hibák:*

A szampontos függvény nem kezeli a tizedes számokat konzisztensen (pl. 1234.56 → 1.234.56, ami nem szabványos).

A modifyUser függvény nem ellenőrzi az érvénytelen mezőket (pl. role: 'invalid\_role').

## **8.11. Összegzés és javaslatok**

**Erősségek**: A tesztek átfogóan lefedik az alapvető funkciókat és hibakezeléseket.

*Fejlesztési lehetőségek:*

Bővíteni kell az integrációs teszteket (pl. kosár + rendelés létrehozása).

Implementálni kell tizedes számok kezelését a szampontos függvényben.

**Következő lépések:**Tesztelési automatizáció bevezetése CI/CD folyamatokba.

# **9. Fejlesztés**

## **9.1 Tényleges rendelés**

A rendelés még fejlesztés alatt van, még nincsen beépítve, hogy ténylegesen lehessen rendelni.

## **9.2 Számla kiállítás**

A számlát gyorsan be lehet integrálni a webshopba. Csak még fejlesztés alatt van.

## **9.3 Email cím megerősítés**

Egy levelet küld a felhasználónak, aki megadta az email címét, majd ott meg kell erősítenie az ügyfélnek. Így működne a jelszó/email cím változtatás az oldalon, hogy meg kell erősítenie.

## **9.4 Több termék feltöltés a webshopba**

Több kategóriát és termékeket szeretnénk feltölteni az oldalra, hogy még több rendelés legyen.

## **9.5 Szállítási cím változtatás**

Most úgy van, hogy ha rendel az ügyfél, akkor megadja a szállítási és számlázási adatokat. Majd szeretnénk ezen változtatni úgy, hogy többféle szállítási címeket tudjon megadni a felhasználó és módosítani azt.

## **9.6 Admin felület fejlesztése**

Az admin felületnek a fejlesztése. Mint például, hogy a rendeléseket a felületen lehessen elfogadni vagy elutasítani. A felhasználók adatainak módosítása, törlése.

# **10. Összegzés**

## **10.1 Szakmai fejlődés**

*Webfejlesztési technológiák elsajátítása:*

A projekt során mélyreható tapasztalatot szereztünk a PHP és MySQL alapú backend-fejlesztésben. Kiemelten foglalkoztunk a felhasználókezelés, a kosárrendszer működésének kialakításával, valamint a hatékony SQL lekérdezések tervezésével és optimalizálásával. Megismerkedtünk a biztonságos programozási gyakorlatokkal, különös tekintettel az SQL injection elleni védekezésre (prepared statements alkalmazása), a jelszavak biztonságos tárolására és kezelésére (pl. password\_hash, password\_verify), valamint a session-ök megfelelő kezelésére.

*Komplex rendszertervezés:*

A webshop moduláris felépítése során fontos szempont volt az egyes funkciók – regisztráció, kosárkezelés, rendelés – összehangolt működésének biztosítása. Ennek megvalósítása hozzájárult rendszerszintű gondolkodásunk fejlődéséhez, és betekintést adott a skálázható alkalmazások tervezési elveibe.

*Frontend fejlesztés és UX:*

A felhasználói felület reszponzív kialakítását Bootstrap segítségével végeztük, míg a dinamikus funkciók – mint például a kosár valós idejű frissítése – JavaScript és AJAX alkalmazásával valósultak meg. A felhasználói élmény javítását célzó fejlesztések között szerepeltek a JavaScript-alapú űrlapvalidációk, hibakezelési mechanizmusok, valamint vizuális visszajelzések tervezése, amelyek segítik az átlátható és gördülékeny vásárlást.

*Tesztelési gyakorlat:*

A projekt során különböző tesztelési módszereket alkalmaztunk a szoftver megbízhatóságának növelése érdekében. Használtunk fekete doboz tesztelést a főbb funkciók – például kosárba helyezés, jelszóváltoztatás – működésének ellenőrzésére, valamint unit teszteket az egyes komponensek hibáinak proaktív azonosítására.

## **10.2 Jövőbeli célok**

*A rendszer funkcionalitásának bővítése:*

A közeljövőben tervezzük további fizetési lehetőségek – például Stripe vagy közvetlen bankkártyás fizetés – integrálását, illetve egy ajánlórendszer bevezetését, amely felhasználói viselkedési minták alapján személyre szabott termékjavaslatokat kínálna.

*Optimalizálás és technológiai fejlesztés:*

A teljesítmény növelése érdekében célunk az adatbázis indexelése, valamint gyorsítótárazási megoldások bevezetése a gyakran lekérdezett adatok esetében. Emellett hosszú távú célként szerepel a felhasználói élmény jelentős javítása egy modern Single Page Application (SPA) architektúra megvalósításával React vagy Vue.js keretrendszer segítségével.

*Tesztelési folyamatok fejlesztése:*

A jelenlegi manuális tesztelési eljárásokat szeretnénk automatizálni, bevezetve integrációs teszteket, amelyek a különálló modulok együttműködését vizsgálják. Továbbá célunk CI/CD eszközök – például GitHub Actions – alkalmazása a fejlesztési folyamatok hatékonyabbá tételére.

*Biztonság fokozása:*

A rendszer biztonságának növelése érdekében tervezzük kétlépcsős azonosítás (2FA) bevezetését, valamint rendszeres biztonsági auditok és automatikus sebezhetőségi vizsgálatok alkalmazását a felhasználói adatok védelmének biztosítására.

# **11. Források**

<https://chatgpt.com/> (Termékek leírása, kód segítség)

<https://create.piktochart.com/> (Fejléc)

<https://getbootstrap.com/> (Dízájnolás)

<https://copilot.microsoft.com/> (AI generált képek a termékekről)

<https://jestjs.io/> (Unit Tesztelés)

<https://mermaid.js.org/> (Folyamat ábrák)