Debreceni SZC Beregszászi Pál Technikum   
Debrecen, Jerikó utca 17, 4032

SZOFTVERFEJLESZTŐ ÉS -TESZTELŐ TECHNIKUS SZAKMA

Online webáruház és raktárkezelő alkalmazás

**Készítette:** Tokai-Kiss Máté

**Csapattagok:** Serdült Zoltán

Debrecen, 2022.

Tartalomjegyzék

[1 Bevezető 4](#_Toc100073680)

[2 Tervezés 4](#_Toc100073681)

[2.1 Elektronikus kereskedelem 4](#_Toc100073682)

[2.2 Logisztika és raktározás 4](#_Toc100073683)

[2.3 Adatbázis 5](#_Toc100073684)

[2.3.1 Termék 6](#_Toc100073685)

[2.3.2 Vásárló 7](#_Toc100073686)

[2.3.3 Rendelés 7](#_Toc100073687)

[2.3.4 Dolgozó 8](#_Toc100073688)

[2.3.5 Beérkezések 9](#_Toc100073689)

[3 Fejlesztői dokumentáció 10](#_Toc100073690)

[3.1 Szerverek 10](#_Toc100073691)

[3.2 Kódszerkesztő 11](#_Toc100073692)

[3.3 PHP, HTML 13](#_Toc100073693)

[3.3.1 Header 13](#_Toc100073694)

[3.3.2 Products 14](#_Toc100073695)

[3.3.3 Kosár 14](#_Toc100073696)

[3.3.4 Fiók 15](#_Toc100073697)

[4 Tesztelés 16](#_Toc100073698)

[4.1 Regisztráció 16](#_Toc100073699)

[4.2 Bejelentkezés 17](#_Toc100073700)

[4.3 Rendelés 17](#_Toc100073701)

[4.4 Fiók kezelése 18](#_Toc100073702)

[4.4.1 Saját adatok 18](#_Toc100073703)

[4.4.2 Jelszó változtatás 19](#_Toc100073704)

[5 Továbbfejlesztési lehetőségek 20](#_Toc100073705)

[5.1 CSS 20](#_Toc100073706)

[5.2 Regisztráció 20](#_Toc100073707)

[5.3 Kosár 20](#_Toc100073708)

[5.4 Hibaüzenetek 20](#_Toc100073709)

[5.5 Sorrend 20](#_Toc100073710)

[6 Felhasználói dokumentáció 20](#_Toc100073711)

[6.1 Felhasználói követelmények 20](#_Toc100073712)

[6.2 Program használata 21](#_Toc100073713)

[6.2.1 Regisztráció 21](#_Toc100073714)

[6.2.2 Bejelentkezés és fiók 22](#_Toc100073715)

[6.2.3 Vásárlás 24](#_Toc100073716)

[7 Összegzés 25](#_Toc100073717)

[Jegyzékek 26](#_Toc100073718)

[Ábrajegyzék 26](#_Toc100073719)

[Táblázatok 26](#_Toc100073720)

[Források 26](#_Toc100073721)

# Bevezető

A mi csapatunk Tokai-Kiss Máté és Serdült Zoltán egy már nagyon régi esetre próbálunk egy szoftvert csinálni a kereskedelem terén. A legtöbb üzlet foglalkozik raktározás és eladás részel és az informatika megjelenésével egyre hatékonyabb lett. Kezdetben még csak nyilvántartók és raktárkezelési programok léteztek, de az internet feltűnése nagyban megváltoztatta az üzletek világát. Lehetőség nyílt az internetes eladásoknak, a webshopok létre jöttének. Mi egy üzlet online kezelését szeretnénk megvalósítani a termékek felvételétől, az interneten való árusításáig, ami egy webshopként valósulna meg. Választásunkban sokat közrejátszott, hogy pár éve mind a ketten hasonló munkakörben dolgozunk, így van valamelyest rálátásunk a kereskedés és a logisztika működésére és szeretnénk hatékonyabbá tenni azzal, hogy virtuális világba ültetjük át, ezzel sokkal könnyebbé és kényelmesebbé téve a munkamenetet. A tanagyagok előrehaladtával egyre többször találkoztunk olyan, a mindennapi használatban alkalmazott programok, weboldalak megvalósításával, amelyeket a tanulópadon mi is ugyanúgy elkészítettünk, ezért elhatároztuk, hogy megpróbálunk egy hasonló szoftvert készíteni, amit a munkánk során is használunk. Mivel rengeteg online áruház ás raktár üzemel, így gondoltuk, hogy a saját vizsgaremek létrehozása során számos programot és weboldalt tudunk majd tanulmányozni, ami segítségével saját ízlésünkre tudjuk majd formálni a projektet. Mivel ezt a projektet nem egyedül kellett megcsinálni, pont jól jött, hogy viszonylag két nagy részre lehet felosztani, a raktározás részre és az online értékesítésre. Én vállaltam az online rész megvalósítása, Zoltán pedig az asztali alkalmazás rész elkészítését választotta, amit egy közösen megtervezett adatbázisból fogunk feltölteni adatokkal.

# Tervezés

## Elektronikus kereskedelem

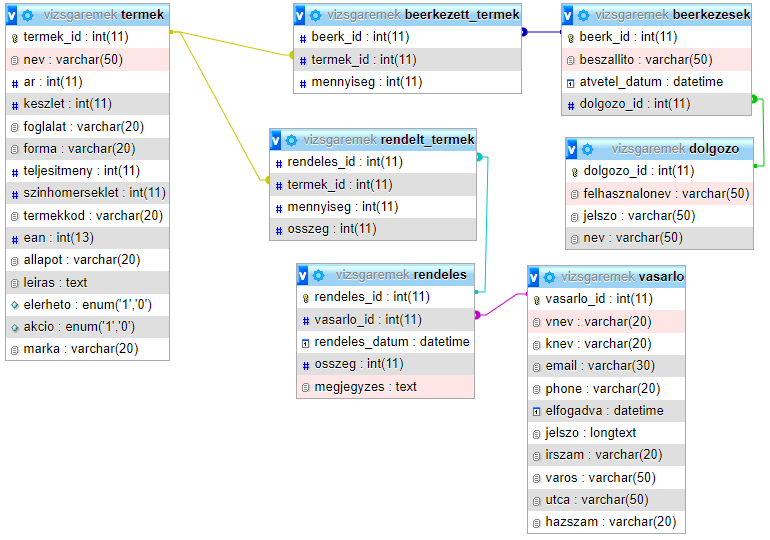
Elektronikus kereskedelem (e-kereskedelem) alatt az árucikkek interneten való kereskedését értjük, aminek a fő alapja az olyan honlap, ahol ez a folyamat végbemegy. Ezeket a honlapokat hívjuk webshopnak (internetes bolt, internetes áruház, online áruház). Eltérően a hagyományos áruházaktól a webshopokban a valós termékekkel fizikailag nem találkozik a vásárló, hanem általában kép formájában tudja megtekinteni. Általában a termékekhez tartozik egy részletes leírás, amely tartalmazza az áru pontos specifikációit. A legtöbb webáruház egy speciális piaci részesedést céloz meg a teljes piaci szegmensből, de természetesen vannak kivételek, mint például az Amazon. A mai napra igen kiforrott technológiai háttérrel rendelkezik a – közel harminc éves – online értékesítés. Rengeteg különböző módszert fejlesztettek ki a kereskedés megoldására, mint például az igen elterjedt kosaras rendszerű webshop, ahol a vevő egy virtuális kosárba tudja pakolni a megvásárolni kívánt termékeket, amelyet tetszés szerint akár ki is vehet a kosárból. A rendelés véglegesítéséhez szükség van egy előzetes regisztrációra a weboldalon, így rögzíteni lehet a szállításhoz kapcsolódó információkat, vagy cégek esetében az adószámot is, amelyek így a jövőbeni vásárlást könnyítik. A regisztrációval az összes vásárlást tárolni lehet, amit így a vásárló bejelentkezés után bármikor meg tud tekinteni.

## Logisztika és raktározás

Raktározásnak az áru tárolására, állagának megóvására, a készletek elhelyezésére szolgáló tevékenységeket nevezzük. Leggyakrabban természetesen kiszolgáló szerepe van: a vállalkozások által megtermelt értékek tárolója. Célja a nyersanyag biztonságos tárolása a felhasználásáig, fizikai tulajdonságainak megőrzése. A raktárakat, raktározás rendszereket elsősorban a különböző munkafolyamatok gazdaságos kapacitási különbségeinek kiegyenlítése teszi szükségessé, mint például a beszerzéskori optimális mennyiség. A raktározás sajátos szerepet tölt be a kitermeléstől a felhasználásig terjedő komplex folyamatrendszerben. A raktárak a részfolyamatok összekötő elemei. A raktározás történhet nyitott és zárt helyen. Zárt helyen tárolják például az elektronikai eszközöket, világítástechnikai termékeket (izzók, LED-szalagok). A raktározás általában valamely üzlet zavartalan és folyamatos működését, valamint az eszközök, nyersanyagok, félkész- és késztermékek szakszerű, veszteségmentes megőrzését szolgálja mindaddig, míg azok felhasználásra, illetve értékesítésre kerülnek. A raktározás folyamata a következő részfolyamatokat foglalja magába: az áru átvétele, az áru tárolása, kezelése, az áru kiadása. Az egész folyamat kezdete az áruátvétel, ahol rögzíteni lehet a beszállítót, az áruátvételt végző személyt, időpontot, pontos mennyiséget. Az áruátvétel után a termékeket raktározni kell.

## Adatbázis

A tervezést alapvetően meghatározza, hogy milyen információs egységekkel szeretnénk dolgozni, illetve az egyes adatelemek között milyen mélyebb összefüggés adott. Ezen lépések egyike sem formalizálható, nincs általános algoritmus a megoldásukra, hiszen az adatbázisokban bármilyen ismeretet leírhatunk, tárolhatunk, lényegében korlátozás nélkül. Minden fontosabb tulajdonságot rögzíteni kell, amely a későbbiekben segítségünkre lesznek az adatok feldolgozása során és elkerülendő a redundancia.



1. ábra: Adatbázis táblái

### Termék

1. táblázat: Termék adatszótár

|  |
| --- |
| **termek** |
| **Oszlop** | **Típus** | **Null** | **Idegen kulcs** | **Megjegyzés** |
| termek\_id *(Primary)* | int(11) | Nem |  | AUTO\_INCREMENT, egyedi azonosító |
| nev | varchar(50) | Nem |  | a termék neve |
| ar | int(11) | Nem |  | a termék ára |
| keszlet | int(11) | Nem |  | raktári készlet |
| foglalat | varchar(20) | Nem |  | a termék foglalatja |
| forma | varchar(20) | Nem |  | a termék formája |
| teljesitmeny | int(11) | Nem |  | a termék wattban mért teljesítménye (Watt) |
| szinhomerseklet | int(11) | Nem |  | az izzó színhőmérséklete (Kelvin) |
| termekkod | varchar(20) | Nem |  | a termék egyedi szöveges azonosítója |
| ean | int(13) | Nem |  | ean-kód |
| allapot | varchar(20) | Nem |  |  |
| leiras | text | Nem |  | részletes, szöveges leírás |
| elerheto | enum('1', '0') | Nem |  | webshopban való megjelenítés |
| akcio | enum('1', '0') | Nem |  |  |

A termékeket tartalmazó táblának elsődleges kulcsnak a „termek\_id” oszlopot választottuk. Ezeknek az értéke minden rekord esetében eltérő. A „nev” oszlopban fogjuk tárolni a termék megnevezését (pl. Globe A60 8W E27 4000K) ezért típusa 50 karakter hosszú varchar. Az „ar” oszlopnak neveztük el a vételárat tartalmazó oszlopot, amelyet integer típusú változóként rögzítünk. A „keszlet” oszlop fogja jelenteni a fizikális raktárban található és rendelkezésre álló termékek mennyiségét, típusa integer. A „foglalat” és „forma” oszlop a fényforrások két főbb tulajdonságát reprezentálják, nevükből adódóan a foglalatot (E27, E14, G4, GU10..) és a formát (gömb, gyertya, mini-gömb..) ezért típusuk 20 karakter hosszú varchar. A „teljesitmeny” oszlop az izzó watt-ban mért teljesítményét írja le, típusának integert választottunk. Fontos a fényforrásnak a színhőmérsékletét is rőgzíteni, amely a vörös színtől a sárgán és a fehéren át a kékig terjedő spektrumon kiválasztott szín. A színhőmérséklet skálán az értékeket kelvinben jelöljük, ezért az integer típusú változó tökéletes választás. A „termekkod” oszlopban a termék szöveges egyedi azonosítóját tároljuk amelyet a termék tulajdonságai alapján lehet megalkotni. Első része a formát jelöli (gömb – g, gyertya – c, mini-gömb - mg), majd a foglalat(e14,e27) ezt követi a teljesítmény és végül a színhőmérséklete, ezért típusa 20 karakteres varchar. Az európai gyártmánykódot az „ean” oszlopban tároljuk. Ez egy olyan vonalkódrendszer amelyet kiskereskedelmi árucikkek jelzésére használnak és 13 számjegyből áll. A „leiras” oszlop a terméket hosszan és részletesen leíró szöveget tartalmazza, így adattípusának text-et választottunk. Az „elerheto” oszlop a termék elérhetőségét jelzi a webshop felé. Értéke 1 (elérhető) vagy 0 (nem elérhető), adattípusa enum('1’,’0’). A leíráshoz hasonlóan az „akcio” az akciókat jelzi, típusa szintúgy enum(’1’,’0’).

### Vásárló

2. táblázat: Vásárló adatszótár

|  |
| --- |
| **vasarlo** |
| **Oszlop** | **Típus** | **Null** | **Idegen kulcs** | **Megjegyzés** |
| vasarlo\_id *(Primary)* | int(11) | No |  | AUTO\_INCREMENT, egyedi azonosító |
| vnev | varchar(20) | No |  | vezetéknév |
| knev | varchar(20) | No |  | keresztnév |
| email | varchar(30) | No |  | email-cím |
| phone | varchar(20) | No |  | telefonszám |
| elfogadva | datetime | No |  | ÁSZF elfogadásának dátuma |
| jelszo | longtext | No |  | hashelt jelszó |
| irszam | varchar(20) | No |  | irányítószám |
| varos | varchar(50) | No |  |  |
| utca | varchar(50) | No |  |  |
| hazszam | varchar(20) | No |  |  |

A vásárló táblában fogjuk tárolni a leendő vásárlók lényegesebb tulajdonságait. Az első oszlop „vasarlo\_id” elsődleges kulcs, a vásárlót azonosító egyedi szám, típusa integer. A teljes nevet két különböző oszlopban tároljuk, külön a vezetéknevet és külön a keresztnevet. Mindkettő típusa 20 karakter hosszú varchar. Email címek tárolásara egy 30 karakter hosszú varchar típusú oszlopot választottunk. A telefonszámok különböző megadásai végett(például: +36, 06) a varchar típust preferáltuk az integer helyett. Fontos rögzíteni, hogy a vásárló elfogadta az ÁSZF-et. Erre egy datetime típusú változót választottunk, így az elfogadás dátumát is rögzíteni tudjuk. A jelszavak biztonságos tárolására longtext típus mellett döntöttünk, mivel ez egy hashelt string lesz. A szállítási címet négy részre bontottuk és mindegyiknek a varchar típust választottuk.

### Rendelés

A rendelés megoldására két táblát hoztunk létre. Az egyik táblában magát a rendelést rögzítjük egy másikban pedig az adott rendeléshez tartozó rendelt termékeket.

3. táblázat: Rendelés adatszótár

|  |
| --- |
| **rendeles** |
| **Oszlop** | **Típus** | **Null** | **Idegen kulcs** | **Megjegyzés** |
| rendeles\_id *(Primary)* | int(11) | No |  | AUTO\_INCREMENT, egyedi azonosító |
| vasarlo\_id | int(11) | No | vasarlo -> vasarlo\_id |  |
| rendeles\_datum | datetime | No |  |  |
| osszeg | int(11) | No |  | teljes összeg |
| megjegyzes | text | No |  |  |

Ennek a táblának az elsődleges kulcsa a „rendeles\_id”, ami rekordonként egyedi és típusa integer. A következő oszlop egy idegen kulcs, a vasarlo tábla „vasarlo\_id” oszlopa, minek a segítségével tárolni tudjuk a vásárló azonosítóját. A rendelés leadásának a dátumát egy datetime típusú változóban rögzítjük „rendeles\_datum” néven. A rendelés teljes összegét egy integer típusú „osszeg” oszlopban mentjük el. Lehetősége lesz még a vásárlónak a rendeléshez kapcsolódó egyéb megjegyzések hozzáfűzésére. Mivel a megjegyzések hosszabb szövegek is lehetnek típusának text-et választottuk.

4. táblázat: Rendelt termék adatszótár

|  |
| --- |
| **rendelt\_termek** |
| **Oszlop** | **Típus** | **Null** | **Idegen kulcs** | **Megjegyzés** |
| rendeles\_id | int(11) | No | rendeles -> rendeles\_id |  |
| termek\_id | int(11) | No | termek -> termek\_id |  |
| mennyiseg | int(11) | No |  |  |
| osszeg | int(11) | No |  | részösszeg |

A megrendelt termékek táblájának nincs elsődleges kulcsa. A „rendles\_id” oszlopa idegen kulcs, ami a rendeles tábla „rendles\_id” oszlopával egyezik meg. A „termek\_id” szintén idegen kulcs, ami a termek tábla „termek\_id” oszlopát reprezentálja. A két idegen kulcs segítségével rögzíteni tudjuk az adott rendeléssel vásárolt termékeket. Az adott termékből megrendelt mennyiséget „mennyiseg” oszlopban tároljuk integer típusként. Egy termék részösszegét is rögzítjük az „osszeg” oszlopban, típusa int.

### Dolgozó

5. táblázat: Dolgozó adatszótár

|  |
| --- |
| **dolgozo** |
| **Oszlop** | **Típus** | **Null** | **Idegen kulcs** | **Megjegyzés** |
| dolgozo\_id *(Primary)* | int(11) | No |  | AUTO\_INCREMENT, egyedi azonosító |
| felhasznalonev | varchar(50) | No |  |  |
| jelszo | varchar(50) | No |  |  |
| nev | varchar(50) | No |  |  |

A „dolgozo” táblát a program a raktári beléptetéshez fogja használni. A „dolgozo\_id” egy integer típusú autoincrement elsődleges kulcs ami csak azonosításra és illesztésekre használatos. A „felhasznalonev” egy varchar típusú 50 karakter hosszúságú karakterlánc amit az admin ad meg, az épp a programot használni akaró dolgozó embernek nincs semmi megkötöttség a bevitelhez lehet kis-nagy betű és szám. A „jelszo” a „felhasznalonev” mező párja, ez egy varchar típusú 50 karakter hosszúságú karakterlánc nincs semmi megkötöttség a bevitelhez, lehet kis-nagy betű. A „nev” a „dolgozo” tábla egyedének az értelmezhetőbb beazonosítására van, ez egy varchar típusú 50 karakter hosszúságú karakterlánc.

### Beérkezések

6. táblázat: Beérkezések adatszótár

|  |
| --- |
| **beerkezesek** |
| **Oszlop** | **Típus** | **Null** | **Idegen kulcs** | **Megjegyzés** |
| beerk\_id *(Primary)* | int(11) | No |  | AUTO\_INCREMENT, egyedi azonosító |
| beszallito | varchar(50) | No |  | Az áru beszállitója |
| atvetel\_datum | datetime | No |  |  |
| dolgozo\_id | int(11) | No | dolgozo -> dolgozo\_id | a dolgozó azonosítása |

A beérkezések rögzítésére a rendelt termékekhez hasonlóan két táblát alkalmazunk. Minden beérkezés kap egy egyedi azonosítót, amit a „beerk\_id” oszlopban rögzítünk. Típusa integer és értékét az adatbázis adja meg. A beszállítók nevét egy varchar típusú változóban tároljuk. Fontos elmenteni a beérkezés átvételének dátumát és az átvételt végző dolgozót. Ezt egy idegen kulccsal oldjuk meg, ami a „dolgozo” tábla „dolgozo\_id” oszlopa.

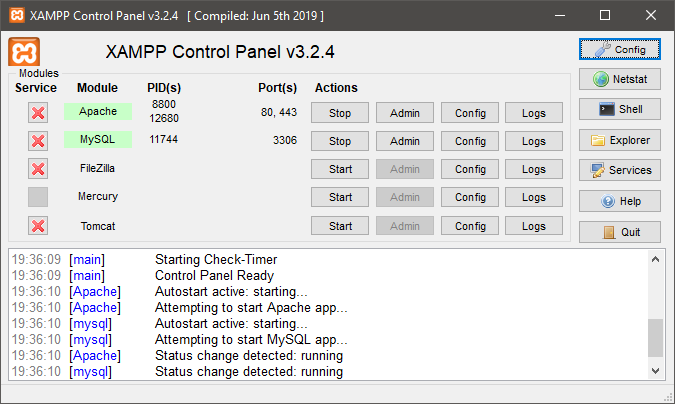
7. táblázat: Beérkezett termékek adatszótár

|  |
| --- |
| **beerkezett\_termek** |
| **Oszlop** | **Típus** | **Null** | **Idegen kulcs** | **Megjegyzés** |
| beerk\_id | int(11) | No | beerkezesek -> beerk\_id | beérkezés azonosítója |
| termek\_id | int(11) | No | termek -> termek\_id | termék azonosítója |
| mennyiseg | int(11) | No |  |  |

A beérkezett termékek táblájának nincs elsődleges kulcsa. A „beerk\_id” oszlopa idegen kulcs, ami a beerkezesek tábla „beerk\_id” oszlopával egyezik meg. A „termek\_id” szintén idegen kulcs, ami a termek tábla „termek\_id” oszlopát reprezentálja. A két idegen kulcs segítségével rögzíteni tudjuk az adott beérkezéssel érkezett termékeket. Az adott termékből beérkező mennyiséget „mennyiseg” oszlopban tároljuk integer típusként.

# Fejlesztői dokumentáció

## Szerverek



2. ábra: XAMPP vezérlő-panel

Mivel vizsgaremeknek egy webes alkalmazás megalkotását választottam, ezért mindenféleképp szükség van egy webszerverre. Ennek a megoldására a XAMPP-ot választottam, így nincs szükség egy másik eszközre a szerver és a kliensoldali fejlesztéshez. Ez a programcsomag tartalmazza az Apache webszervert, ami segítségével lehet futtatni a PHP fájlokat. A PHP egy szkriptnyelv, mely segítségével dinamikus weblapokat lehet készíteni. A PHP nyelv segítségével a szerver-oldalon tudunk létrehozni HTML állományt és majd az adatok feldolgozása után kapja meg a böngésző a kért tartalmat. Ezt a folyamatot a szerver egy úgynevezett interpreter(értelmező) segítségével oldja meg. Rengeteg olyan feladat végezhető el , amelyek a kliens-oldalon biztonsági kockázatot jelentene, ilyenek például a bejelentkezés, adatbáziskezelés, fájlkezelés.

A képen szöveg látható

Automatikusan generált leírás

3. ábra: Webszerver tulajdonságai

A XAMPP tartalmazza a szükséges adatbázist is, ami jelen esetben a MariaDB. Ez egy SQL-alapú relációsadatbázis-kezelő szerver, amely a MySQL elágazásából jött létre, amikor az Oracle felvásárolta a MySQL-t, de teljesen kompatibilis vele. Verziószámozása követi a MySQL-t , azonban a 5.5-ös verzió után saját számozást használ, amely 10-től kezdődik. Leginkább a liszenszelésben tér el a MySQL-től, míg a MySQL nagyrésze nyílt és bárki számára hozzáférhető, de tartalmaz zárt forráskódú forrásrészleteket , addig a MariaDB ezeket a részeket szabadon hozzáférhető részekkel váltották ki ( a különbségek az alábbi linken találhatóak: https://mariadb.com/kb/en/mariadb-vs-mysql-features/ ).

A képen szöveg látható

Automatikusan generált leírás

4. ábra: Adatbázis tulajdonságai

## Kódszerkesztő

Mivel az alkalmazásomhoz szükség volt HTML, CSS, JavaScript és PHP-ban való fejlesztésre, ezért a választásom a Visual Studio Code kódszerkesztőre esett. Ez egy ingyenes nyílt forráskódú kódszerkesztő, amelyet a Microsoft fejleszt. Támogatja a hibakeresőket és beépített Git támogatással rendelkezik. Az alkalmazás elérhető Windows, Linux és OS X operációs rendszereken. A gyors kódoláshoz nagy segítséget nyújt a beépített IntelliSense, ami lehetővé teszi az intelligens kódkezelést. Ez számos támogató funkciót foglal magában, mint például a kódkiegészítés, függvények és eljárások paraméterlistája. A Visual Studio Code IntelliSense-t a JavaScript, HTML, CSS fájlok szerkesztésénél használtam. A szerkesztő ablakot a [Ctrl + Space] billentyűkombinációval tudjuk elérni.

A képen szöveg látható

Automatikusan generált leírás

5. ábra: IntelliSense kódkiegészítés

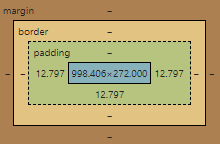
Rengeteg külön telepíthető bővítménnyel rendelkezik. A HTML kód gyorsabb elkészítéséhez az Auto Close Tag bővítményt használtam. Az elérési utak egyszerű kezeléséhez a Path Intellisense bővítményt telepítettem. Ezen felül a felület nagy mértékben testreszabható, így változtatni lehet a kinézetet(témát), megváltoztathatjuk a gyorsbillentyű-kiosztást. Számomra nagyon fontos funkció volt, a szerkesztési csoportok. Ennek segítségével az alkalmazást több részre lehet felosztani, amivel egyszerre több fájl kódját is látni lehet. A HTML dokumentumok megjelenítésének leírására a CSS-t használtam. Főbb célja, hogy a megjelenést és a tartalmat szétválassza. Segítségével külön fájlba tárolni lehet a HTML elemek stílusát, így egy sokkal átláthatóbb kódot kapunk és mivel a böngészők gyakran gyorsítótárban tárolják ezeket a stíluslapokat, így a hálózati forgalom is csökkenthető.

A képen szöveg, monitor, képernyőkép, képernyő látható

Automatikusan generált leírás

6. ábra: Visual Studio Code felülete

A weboldal kliensoldali teszteléséhez a Google által fejlesztett böngészőt a Google Chrome 99.0.4844.82-es verzióját használtam. Számos fejlesztést segítő funkcióval rendelkezik ez a böngésző. Megtekinthető az adott HTML elem dobozmodellje, valamint a rá vonatkozó CSS szabály.

A képen szöveg, képernyő, képernyőkép látható

Automatikusan generált leírás

7. ábra: Box Modell a Chrome-ban 8. ábra: CSS tulajdonságok

JavaScript kódok írása során is segítségemre volt ez a böngésző, mivel ennek a szkriptnyelvnek a futási környezete általában egy webböngésző, azon belül is a JavaScript-motor. A Chrome a V8-JavaScript-motort használja, amit a Google fejlesztett. Ennek a motornak az egyik előnye, hogy a szkriptet nem interpretálva futtatja, hanem natív gépi kódot készít a futtatás előtt, így gyorsabb működést eredményez. Fejlesztés során a Chrome Console funkciójával ellenőrizni tudjuk a JavaScript kívánt működését és esetleges hiba esetén a hibaüzenetek segítenek a feltárásban.

A képen szöveg, képernyő, fekete, képernyőkép látható

Automatikusan generált leírás

9. ábra: JavaScript konzol

## PHP, HTML

A projektem megvalósítása során a leginkább alkalmazott nyelvként a PHP-t használtam. Az egyik legfontosabb funkció a MySQLi függvényei. Segítségével elérhetővé vált a weboldal számára a relációs adatbázis. Megvalósítás során egy külön fájlt hoztam létre a csatlakozáshoz connect.php néven. Ebben a PHP fájlban létrehoztam egy $mysqli nevű változót, amelynek értékként létrehoztam egy mysqli objektumot. A mysqli osztálynak négy paramétert adtam meg. Az első a szerver címe, majd az adatbázist használó felhasználónév, harmadik paraméterként szerepel a jelszó, majd végül az adatbázis neve. Ezután egy feltételes elágazásban megvizsgálom, hogy történt-e valami hiba. A connect\_errno függvény visszatérési értéke a kapcsolódási kísérlet legutolsó hibakódja. Sikeres kapcsolódás esetén értéke nulla. Sikertelen kapcsolódás esetén kiíratom a „Hiba a csatlakozáskor!” sztringet a hibakeresés megkönnyítése érdekében, majd bezárom a szkriptet.

A képen szöveg látható

Automatikusan generált leírás

10. ábra: connect.php

### Header

Az összes dokumentum, amely tartalmat jelenít meg a felhasználó számára tartalmazza a header.php fájlt. Ez a fájlt tartalmazza a HTML dokumentum alapszerkezetének fejléc részét. Itt határozható meg az oldal nyelve, karakterkódolása, címe, az alkalmazott stíluslapokra való hivatkozás. A másik fontos rész a HTML törzsének eleje. Itt a <header> tag-ben hozom létre az weboldal fejlécét. Ez a fejléc két részből áll. Az egyik részét <div class=”login-bar”> tag tartalmazza. Található itt egy hivatkozás, amellyel a kosarat lehet elérni és egy űrlap, amely a bejelentkezés státusza alapján változik. Kijelntkezett állapotban két beviteli mező(e-mail, jelszó) található a bejelentkezéshez és két gomb a regisztrációs űrlap elérésére és a bejelentkezésre. Bejelentkezve a saját fiókunk elérésére egy gomb jelenik meg és egy másik gomb a kijelentkezésre. Bejelentkezés során session változóként tárolódnak a vásárló rendeléshez szükséges információi. A fejléc másik része a navigációs hivatkozásokat tartalmazza. Megoldásomban egy ötelemű rendezetlen listát használtam. Az első elem az akciós termékek elérésére szolgál. A következő három elem létrehozása hasonló elven alapul. A PHP segítségével egy lekérdezést hajtok végre, ami segítségével az adatbázisban létező összes formát ki tudom listázni, majd ezt a listát felhasználva létrehozom a hivatkozásokat.

A képen szöveg látható

Automatikusan generált leírás

11. ábra: header.php

### Products

A termékek böngészésére és az online kosárba való helyezésre készült oldal két fontosabb részre lehet osztani. Az egyik a termékek szűrésére szolgáló mező, a másik pedig a megjelenítendő termékeket tartalmazó táblázat. Az első megoldására a header-ben alkalmazott elv alapján itt is egy listát hozok létre. Ezek a listaelemek tartalmaznak radio típusú beviteli mezőket, az adatbázis-lekérdezés alapján kapott találatoknak megfelelően. Mindegyik elem „onclick” eseményét felhasználva AJAX segítségével változnak a táblázatban szereplő adatok. A táblázat rendelkezik egy azonosítóval, a könnyeb azonosítás végett. A táblázat feltöltése egy MySQLi lekérdezéssel kezdődik. Az így kapott lista alapján generálódik a tartalom. Minden sor tartalmaz egy hivatkozást, amely az adott terméket részletesebben leíró oldalra irányít át. A sor utolsó eleme a „kosárba” gomb, mely segítségével az adott termékből egy darabot belehelyez a virtuális kosárba.

A képen szöveg látható

Automatikusan generált leírás

12. ábra: products.php

### Kosár

A virtuális kosár megjelenítésének alapjául egy asszociatív tömböt alkalmaztam, amelyet egy session változóban tároltam, így az értékét tárolni lehet több oldal közt is. A tömb létrehozása és feltöltése a kosarba.php szkript futtatása során történik. Első lépésként az URI-ban szereplő lekérdezés alapján deklarálja az adott termék azonosítóját. Ezután a kosár session változó létezését ellenőrzi, ha létezik, akkor hozzáad egyet az adott kulcs értékéhez, ha még nem létezik, akkor létrehoz egy asszociatív tömböt az adott kulccsal értékként eggyel.

A képen szöveg látható

Automatikusan generált leírásA képen szöveg látható

Automatikusan generált leírás

13. ábra: kosarba.php-termék-azonosító 14. ábra: kosarba.php-kosárba helyezés

Ennek a szkriptnek a párja a korsarbol.php. Futása során segítségével csökkenthető a tömbben lévő kulcsok értéke. Az előző szkripthez hasonlóan ebben is az URI-ban tárolódik a termék azonosítója. A program megvizsgálja az adott kulcshoz tartozó értéket. Abban az esetben, ha az értéke egy, kitörli az adott kulcsot és a hozzá tartozó értéket a tömbből, más esetben az értéket eggyel csökkenti.

A képen szöveg látható

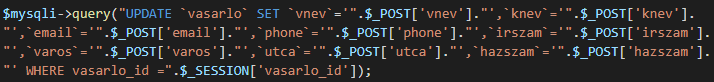
Automatikusan generált leírás

15. ábra: kosarbol.php

A kosár generálása során ezt a tömböt járja be a program. Üres kosár esetén egy egy-soros táblázat jelenik meg „a kosár jelenleg üres” felirattal. Termékekkel rendelkező kosár esetén egy foreach ciklus segítségével minden egyes elemnek létrehoz egy sort, amelyben megjelenik a termék neve, ami egy hivatkozás a termék részletesebb leírására, a termék egységára, a rendelt mennyiség és egy-egy gomb a mennyiség növelésére, csökkentésére. Utolsó sorban egy gomb jelenik meg, amely átirányít a rendelés feladásához. A gomb a bejelentkezés alapján változik. Kijelentkezve inaktívvá válik a gomb, hiszen csak regisztrált felhasználó tud rendelni. A véglegesítés során a vásárló ellenőrizni tudja az adatok helyességét, a rendelt terméket, látható a végösszeg és igény szerint megjegyzést is lehet írni az adott rendeléshez.

### Fiók

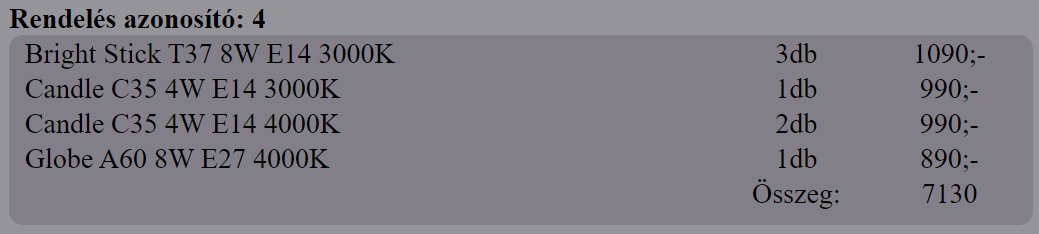
A saját fiók oldalon három funkció közül választhat a vásárló. Ezek navigálására három gomb áll rendelkezésre. Ezek a gombok AJAX segítségével változtatják a megjelenített tartalmat. Az első funkció az adatok megjelenítése és ezeknek a változtatása. Session változók segítségével betöltéskor a szkript feltölti az űrlapot a bejelentkezett vásárló adataival. Változtatás esetén a „Mentés” gombra kattintva a friss adatokat rögzíti a program az adatbázisban.



16. ábra: Felhasználó adatainak módosítása

A második funkció a leadott rendeléseket listázza ki. AJAX segítségével a jelenlegi oldalon maradva létrehoz egy táblázatot, amelyben soronként a rendelés azonosítója, dátuma, a rendeléshez tartozó megjegyzés és a végösszeg található. Minden sorhoz beállít egy „onclick” eseményt, ami segítségével megtekinthető az adott rendelés tételekre lebontva.





17. ábra: Rendelés részletei

Az utolsó funkció egy egyszerű jelszóváltoztatás. A program bekéri a jelenlegi jelszót, majd a kívánt új jelszót kétszer. Megfelelő adatok bevitele után a kívánt jelszó hash-elve rögzül az adatbázisban.

A képen szöveg látható

Automatikusan generált leírás

18. ábra: Jelszó módosítása

# Tesztelés

## Regisztráció

A bejelentkezés során számos hibalehetőség áll fent a sok beviteli mezők miatt. A hibák észlelésére az URL lekérdezéseit alkalmaztam. A sikeres regisztrációhoz az összes mező kitöltése szükséges. Ezt a feladatot kétszer is ellenőrzi a program, egyszer a „required” attribútum használatával az input tageknél, majd a PHP szkript is ellenőrzi a mezők ürességét. Az e-mail formai követelményeit egyszer az email típusú html tag ellenőrzi, majd a filter\_var függvény segítségével is ellenőrzi a szkript.

A képen szöveg látható

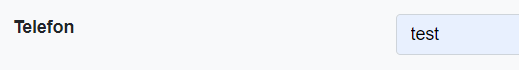
Automatikusan generált leírás

A jelszavak esetében csak azt kell ellenőrizni, hogy megegyeznek-e.



Telefonszámok esetében az ellenőrzést a PHP szkriptben egy reguláris kifejezés segítségével végzi a program. Ellenőrzi a megfelelő formátumot és hogy csak számok szerepelnek-e.

A képen szöveg látható

Automatikusan generált leírás 



19. ábra: Telefonszám ellenőrzése

Utolsó lépésben egy SQL-lekérdezést hajtunk végre. Segítségével ellenőrizni lehet, hogy a kiválasztott e-mail cím már használatban van-e. Abban az esetben, ha a lekérdezés eredményeként nulla sort kapunk, az azt jelzi, hogy az e-mail nem foglalt.

A képen szöveg látható

Automatikusan generált leírás

20. ábra: Email-cím ellenőrzése

Az összes beviteli mező ellenőrzése után a jelszó hash-elése következik, majd az adatok rögzítése az adatbázisba. A folyamat sikerességét az URL-ben megjelenő lekérdezés alapján tudjuk ellenőrizni.



## Bejelentkezés

Bejelentkezés során 3 lehetőséget különböztettem meg. Az egyik, ha nincsenek kitöltve a mezők, a másik kettő pedig, ha téves a jelszó vagy az e-mail cím. Ezeknek az észlelésére az URL-ben lévő lekérdezéseket használtam.



21. ábra: Bejelentkezési hibák

## Rendelés

A rendelés alapvető feltétele, hogy a felhasználó bejelentkezve legyen. A kosár megtekinthető kijelentkezve, de a többi funkció belépést követel. Megpróbálhatja elérni a szükséges weboldalakat, de ezek weboldalak betöltése előtt a PHP szkript ellenőrzi, hogy a felhasználó be van-e jelentkezve. Kijelentkezett állapotban ezek az oldalak visszairányítanak a kezdőlapra.

A képen szöveg látható

Automatikusan generált leírás

Egy másik hibalehetőség, ha nem létező terméket szeretnének a kosárba helyezni, rendelni. Ezt például úgy lehet elérni, ha a HTML dokumentumot szerkesztik. Chrome böngészőben például jobb klikk, vizsgál-ra kattintva.



Az „onclick” esemény paraméterének módosításával egy olyan termék kerül a kosárba, ami nem létezik. Ennek megfelelően a kosárban megjelenik ár és név nélkül.



22. ábra: Kosár

Ezután a véglegesítés során is betölti az oldal. A végösszeg kiszámolása során nem okoz gondot, hiszen soronként egy SQL-lekérdezés segítségével határozza meg az összeget a program, ha nem talál az azonosítóval terméket akkor az összeg nem változik.

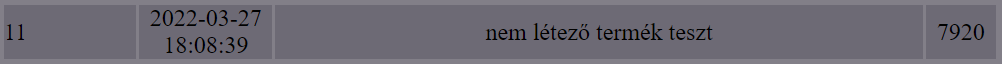
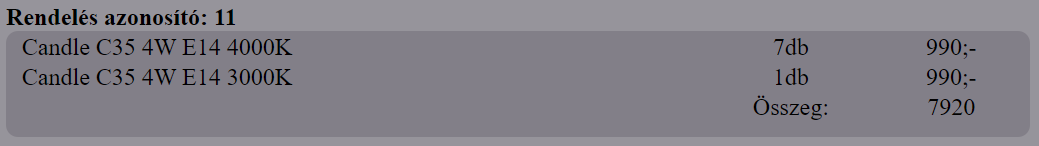
A képen szöveg látható

Automatikusan generált leírás A képen szöveg látható

Automatikusan generált leírás

23. ábra: Rendelés véglegesítése

Véglegesítésre kattintva a rendelés sikeres. Ellenőrzésképp a fiók, rendeléseim menüpontban megtekinthető a rendelés. Mivel az adatbázisban nem létezik termék az adott azonosítóval ezért a tétel nem is kerül rögzítésre.

## Fiók kezelése

### Saját adatok

Sikeres regisztráció után megváltoztathatja a felhasználó az adatait. Fontos, hogy a regisztráció során itt is ellenőrizzük a bevitt adatok helyességét. Ezek alapján az e-mail cím és a telefonszám formai követelményeit érdemes ellenőrizni. Ezeket a PHP szkript ugyanolyan módon ellenőrzi, mint ahogyan az említett regisztráció során végzi.

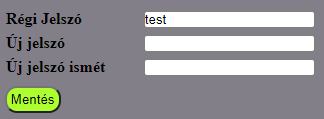
A képen szöveg látható

Automatikusan generált leírás  

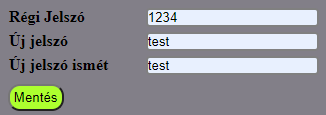
### Jelszó változtatás

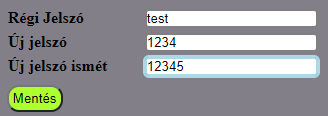
Jelszavak váltása során meg kell adni a jelenlegi jelszót, majd az új jelszót kétszer. Ennek az ellenőrzését a PHP szkript végzi el. Megpróbálható a jelszó változtatása csak a régi jelszó megadásával és a többi mező üresen hagyásával, hisz ebben az esetben az új jelszavak megegyeznek.

(A példában használt input mezők típusa „text” az olvashatóság miatt és a jelszó „test”)

Ebben az esetben az üres mezők hibával talakozunk. Rossz vagy eltérő jelszavak esetén szintúgy észleli a program a hibát és nem változtat semmit.

# Továbbfejlesztési lehetőségek

Ebben a fejezetben arról a lehetőségekről szeretnék pár szót ejteni, amelyekre az idő végessége miatt nem tudtam, vagy a jelenleg rendelkezésre álló eszközökkel nem tudtam, de a jövőben érdemes megvalósítani.

## CSS

A weboldal jelenlegi grafikai dizájnjával nem teljesen veszi figyelembe az olyan eszközöket, amelyek nem rendelkeznek elég nagy kijelzővel, ezért ebben az esetben néhány elem csak az oldal vízszintes görgetésével érhető el. Ennek megoldására lehetne használni a media query-ket számos kijelző méretét figyelembe véve és külön-külön optimalizálni.

## Regisztráció

A regisztrációs űrlap gördülékenyebbé tételét is megszerettem volna valósítani JavaScript, mind PHP segítségével. Az egyik lehetőség, hogy futás közben jelezné az oldal a hibás bevitelt, például a telefonszám vagy az e-mail cím során. Egy másik hasznos funkció, ha sikertelen próbálkozás után az oldal visszatöltené a megfelelő beviteli mezők értékeit.

## Kosár

A termékek kosárba helyezése során jelenleg nincs lehetőség egyszerre több darab megadására. Nagy tételű rendelés esetén akár ez a vásárlót eléggé zavarni tudja. A kosárból való törlésre is érdemes lenne létrehozni egy gombot, hogy ne kelljen többször csökkenteni a tétel értéket. Ezek a funkciók mind az oldal használhatóságát növelnék.

## Hibaüzenetek

Az oldal a sikertelen próbálkozásokat, például bejelentkezés esetén csak az URL-ben jelzi. A felhasználó könnyeb tájékozódását segítené néhány az oldalon olvasható hibaüzenet.

## Sorrend

A termékek megjelenítése során segítséget tudna nyújtani a vásárlónak, ha tetszés szerint sorba lehetne rendezni a termékeket. Megoldható lenne például a termékek megjelenítésénél használt SQL-lekérdezéshez az „ORDER BY” záradék hozzáadásával, ha a felhasználó rákattint a táblázat első sorában található mezők valamelyikére, így lehetőség lenne például ár szerint növekvő, vagy csökkenő sorrendbe állítani a termékeket.

# Felhasználói dokumentáció

## Felhasználói követelmények

A program használatához szükség van egy olyan eszközre, amely képes futtatni egy webböngészőt, amely megfelel a HTML5 és CSS3 követelményeinek. A JavaScript futtatását engedélyezni kell a böngészőben. Ajánlott a Chrome böngésző legújabb verziójának használata, amely több operációs rendszeren is működik.

## Program használata

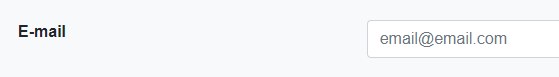
### Regisztráció

Az oldalon történő vásárláshoz elengedhetetlen, hogy a vásárló regisztrált fiókkal rendelkezzen. Ehhez a folyamathoz szükséges első lépés, hogy rákattintunk az oldal jobb felső sarkában található piros „Regisztráció” gombra.

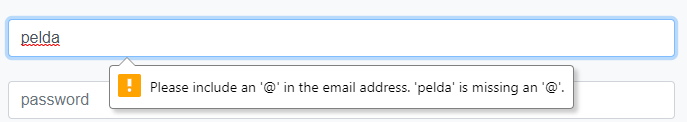
A képen szöveg látható

Automatikusan generált leírás

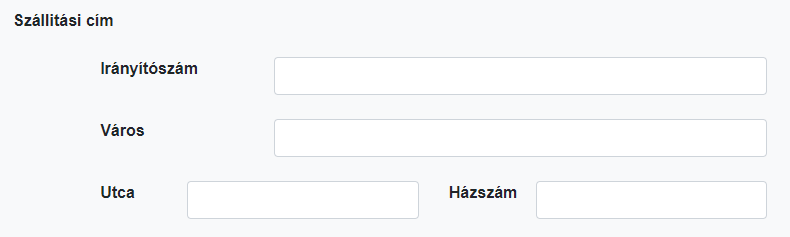
Erre a gombra kattintva egy másik oldalra irányít minket a weboldal. Ezen az oldalon tudjuk megadni a szükséges adatokat a sikeres regisztrációhoz. Az adatok bevitele tagoltan történik. Soronként más-más adatot szükséges megadni, melyek kitöltésében segítenek a sorok elején látható feliratok. Minden mező kitöltése kötelező.



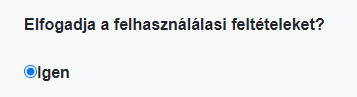
Az első sorban a név megadása szükséges. A teljes név két részre van bontva. Először a vezetéknév, majd a keresztnév megadása szükséges. Következő sorban az e-mail cím megadása történik. Formátuma a felhasználónév @ domain (például [pelda@email.com](mailto:pelda@email.com)). Kitöltésben az oldal is segítséget nyújt. Az e-mail cím segítségével lehet majd később bejelentkezni.



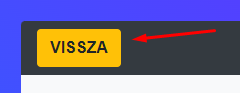
Következő két mező a jelszó megadására szolgál. A saját védelmünk érdekében ajánlott a legalább nyolc karakter hosszú, különleges karaktereket (számok, szimbólumok) tartalmazó jelszó. A jelszó kétszeri megadása azt a célt szolgálja, hogy kizárja az elgépelés lehetőségét, hiszen arra kevés az esély, hogy ugyanazt kétszer elgépeljük. A következő mező a telefonszám megadását teszi lehetővé. A telefonszámban csak számok szerepelhetnek, kivéve az első karaktert, amely lehet (+) is. Legalább hét karakter és legfeljebb 15 karakter hosszú lehet. A következő részben a szállítási cím megadására van lehetőség. Négy részre van tagolva. Első az irányítószám, majd a város és utána az utca és házszám.



Legutolsó sorban a hozzájárulásunkat szükséges megerősíteni, hogy elfogadjuk a felhasználási feltételeket.

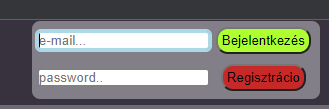


Abban az esetben, ha már rendelkezünk saját fiókkal, akkor a bal felső oldalon található Vissza gombra kattintva visszatalálunk a webshopba.



### Bejelentkezés és fiók

A bejelentkezéshez a jobb felső sarokban található mezőket lehet használni. A felső mező, amelyben szürkén az „e-mail…” szöveg olvasható az e-mail cím megadására szolgál, alatta lévő mező pedig a jelszónak van létre hozva. A szükséges információk begépelése után „Enter”-t ütve, vagy a zöld „Bejelentkezés” gombra kattintva tudunk belépni.



Sikeres bejelentkezés esetén ez a mező helyett megjelenik két gomb, a szürke „Fiók” és a piros „Kijelentkezés”.

A képen szöveg látható

Automatikusan generált leírás

Kijelentkezésre kattintva a rendszer kiléptet minket. A fiók gombra kattintva az oldal átirányít minket a fiókunk kezeléséhez használható oldalra. Itt három funkció található, melyek az oldal bal szélén érhetőek el.

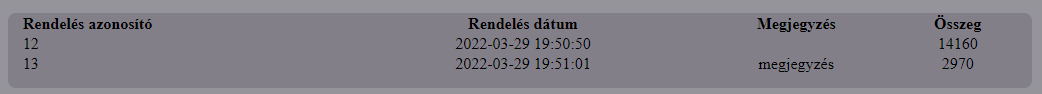
A képen szöveg látható

Automatikusan generált leírás

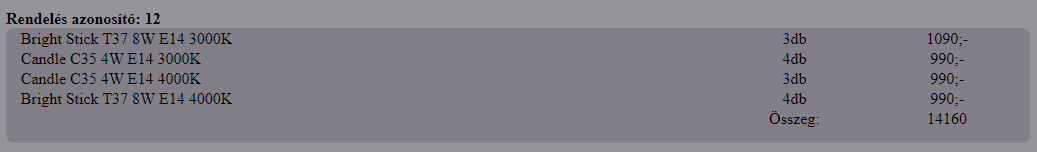
Az első funkció az adataink ellenőrzésére és változtatására szolgál. Az adatok változtatásának mentéséhez a zöld „Mentés” gombra szükséges kattintani.



A második funkció az előző rendeléseink megtekintésére szolgál, amelyet a „Rendeléseim” gombra kattintva érünk el. Ezután az oldal betölti az összes megrendelésünket.



Valamelyik sort bal kattintással kiválasztva, megjelenik az adott rendeléshez tartozó termékek darabja és részösszege.



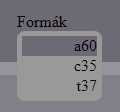
Az utolsó funkció a jelszó megváltoztatására használható. Az első sorban a jelenlegi megváltoztatni kívánt jelszó megadása szükséges. A következő két sorban az új jelszó ismételt megadása szükséges. A változtatás véglegesítése a zöld „Mentés” gombra kattintva történik.



### Vásárlás

A termékeket megjelenítő oldalt, a fejlécben található menüsor segítségével érhetjük el. Lehetőség van csak az akciós termékek megjelenítésére. A kurzort az adott menüpontra irányítva egy almenü jelenik meg, mely segítségével részletesebben tudunk szűrni a termékek közt. Az „Összes” menüpontra kattintva az összes elérhető termék megjelenik.

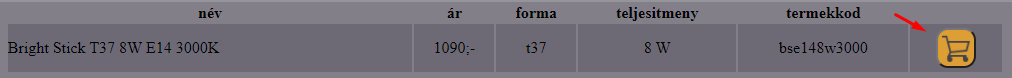




Az oldal bal szélén található menü segítségével részletes szűrést van lehetőségünk beállítani. A szűrés eltávolításához az „Összes” gombra kattintva van lehetőség.



A kiválasztott terméket a táblázat jobb szélén található kosár gombra kattintva helyezhetjük a kosárba.



A kosár megtekintéséhez a jobb felső sarokban található sárga „KOSÁR” gombra szükséges kattintani. Itt látható a jelenleg kosárban lévő összes termék. Ezeknek a mennyiségét csökkenteni, a piros (-) gombbal, növelni pedig a zöld (+) gombbal van lehetőség. A termékek alatt található zöld „RENDEL” gombra kattintva az oldal átirányít minket az adataink ellenőrzésére szolgáló oldalra.



Ezen a részen ellenőrizni tudjuk a személyes adatainkat. Látható az összes rendelni kívánt termék, a hozzájuk tartozó darabszámmal és részösszeggel, valamint a teljes összeg is. Megadható tetszés szerint a rendeléshez megjegyzés is. A zöld „VÉGLEGESÍTÉS” gombra kattintva leadható a rendelés.

A képen szöveg látható

Automatikusan generált leírás

# Összegzés

A projektmunka teljes megvalósítása során rengeteg új tapasztalathoz tudtam hozzájutni, amelyeket egyéni projekt elkészítése mellett nem lett volna lehetőségem megtapasztalni. Az adatbázis tervezése során néhány akadályba ütköztünk, amely rávilágított mennyire fontos és meghatározó része a programoknak a pontosan és jól megtervezett adatbázis, hiszen több szempontnak is eleget kell tennie. Segítségével mindkét platformon fejlesztett alkalmazás zökkenőmentesen tud futni. Sajnos a kellő fejlesztési tapasztalat hiánya miatt, a fejlesztés tervezési szakaszában túl sok mindent akartam megvalósítani és a véges idő végett megvalósításuk elmaradt. A hiányzó funkciók helyett ezért más leegyszerűsített megoldásokat kellet megvalósítanom. A programozás során kezdő fejlesztőként nem igazán alkalmaztam a kommentelést. Kellemetlenül tapasztaltam, amikor egy már hónapja megírt kódrészlet megtekintése során, mondhatni előröl megkellett fejteni a program működését. Megtapasztalhattam mennyire fontos a kellően összehangolt csoportmunka, a feladatok megfelelő felosztása, a pontos egyeztetések, hisz ezek hiányában az egész projekt készítése áll. A program elkészítése során rengeteg új, eddig számomra ismeretlen szakmai megvalósításokat tanultam. Új technológiák elsajátítása a jövőbeni fejlesztéseim gördülékenyebbé tételét is segítette.

# Jegyzékek

## Ábrajegyzék

[1. ábra: Adatbázis táblái 5](#_Toc99646495)

[2. ábra: XAMPP vezérlő-panel 10](#_Toc99646496)

[3. ábra: Webszerver tulajdonságai 10](#_Toc99646497)

[4. ábra: Adatbázis tulajdonságai 11](#_Toc99646498)

[5. ábra: IntelliSense kódkiegészítés 11](#_Toc99646499)

[6. ábra: Visual Studio Code felülete 12](#_Toc99646500)

[7. ábra: Box Modell a Chrome-ban 8. ábra: CSS tulajdonságok 12](#_Toc99646501)

[9. ábra: JavaScript konzol 13](#_Toc99646502)

[10. ábra: connect.php 13](#_Toc99646503)

[11. ábra: header.php 14](#_Toc99646504)

[12. ábra: products.php 14](#_Toc99646505)

[13. ábra: kosarba.php-termék-azonosító 14. ábra: kosarba.php-kosárba helyezés 14](#_Toc99646506)

[15. ábra: kosarbol.php 15](#_Toc99646507)

[16. ábra: Felhasználó adatainak módosítása 15](#_Toc99646508)

[17. ábra: Rendelés részletei 15](#_Toc99646509)

[18. ábra: Jelszó módosítása 16](#_Toc99646510)

[19. ábra: Telefonszám ellenőrzése 16](#_Toc99646511)

[20. ábra: Email-cím ellenőrzése 17](#_Toc99646512)

[21. ábra: Bejelentkezési hibák 17](#_Toc99646513)

[22. ábra: Kosár 18](#_Toc99646514)

[23. ábra: Rendelés véglegesítése 18](#_Toc99646515)

## Táblázatok

[1. táblázat: Termék adatszótár 6](#_Toc99646724)

[2. táblázat: Vásárló adatszótár 7](#_Toc99646725)

[3. táblázat: Rendelés adatszótár 7](#_Toc99646726)

[4. táblázat: Rendelt termék adatszótár 8](#_Toc99646727)

[5. táblázat: Dolgozó adatszótár 8](#_Toc99646728)

[6. táblázat: Beérkezések adatszótár 9](#_Toc99646729)

[7. táblázat: Beérkezett termékek adatszótár 9](#_Toc99646730)

## Források

<http://www.kiservinegon.hu/2010/09/webshop-tortenelem-az-e-kereskedelem.html>

<https://hu.wikipedia.org/wiki/E-kereskedelem>

<https://iconscout.com/icon/cart-374>

<https://mariadb.com/kb/en/mariadb-vs-mysql-features/>

<https://code.visualstudio.com/docs/editor/intellisense>

<https://v8.dev/>

<https://www.php.net/manual/en/class.mysqli.php>

<https://support.google.com/chrome/a/answer/7100626>