

VÁCI SZAKKÉPZÉSI CENTRUM
BORONKAY GYÖRGY
MŰSZAKI TECHNIKUM ÉS GIMNÁZIUM

Vizsgaremek

Dallos Ruben, Herédi Gábor

2024.

VÁCI SZAKKÉPZÉSI CENTRUM
BORONKAY GYÖRGY
MŰSZAKI TECHNIKUM ÉS GIMNÁZIUM



Vizsgaremek

TornadoPC

Konzulens: Wiezl Csaba

Készítette: Dallos Ruben
Herédi Gábor

Hallgatói nyilatkozat

Alulírottak, ezúton kijelentjük, hogy a vizsgaremek saját, önálló munkánk, és korábban még sehol nem került publikálásra.

Dallos Ruben

Herédi Gábor

Konzultációs lap

Vizsgálók neve: Dallos Ruben, Herédi Gábor

Vizsgaremek címe: TornadoPC

Program nyújtotta szolgáltatások:

- Regisztráció - bejelentkezés
- Asztali adminisztrátori felület
- Konfiguráció összeállítás és megosztás
- Kedvencek - Kosár
- Rendelés

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

Hallgatói nyilatkozat	3
Konzultációs lap	4
Tartalomjegyzék.....	5
Témaválasztás.....	9
1 Fejlesztői dokumentáció.....	10
1.1 Munkamegosztás	10
1.2 Vizsgaremek bemutatása.....	11
1.3 Fejlesztői környezet.....	11
1.3.1 Visual Studio Code	11
1.3.2 Visual Studio 2022	11
1.3.3 XAMPP	11
1.3.4 Teams.....	12
1.3.5 Github	12
1.4 Felhasznált technológiák	13
1.4.1 HTML.....	13
1.4.2 CSS.....	13
1.4.3 PHP	13
1.4.4 JavaScript	14
1.4.5 jQuery	14
1.4.6 AJAX	14
1.4.7 REST API	15
1.4.8 MySQL	15
1.4.9 C#	16
1.5 Használati esetmodell	16

1.6	A program felépítése	17
1.7	Adatbázis	18
1.7.1	akciok tábla	20
1.7.2	arkategoriak tábla	21
1.7.3	ertekeles tábla	22
1.7.4	felhasznalok tábla	23
1.7.5	fizetesi_modok tábla	24
1.7.6	jellemzok tábla	25
1.7.7	kedvencek tábla	26
1.7.8	kepek tábla.....	27
1.7.9	konfig tábla	28
1.7.10	konfig_id tábla	29
1.7.11	kosar tábla.....	30
1.7.12	markak tábla	31
1.7.13	rendeles tábla	32
1.7.14	termekes tábla.....	33
1.7.15	termekes_akciok.....	35
1.7.16	termek_kategoriak tabla.....	36
1.7.17	tetelek tábla	37
1.8	Rendszerkövetelmények.....	38
1.9	Megfelelő indítás.....	39
1.10	A weboldal felépítése	39
1.10.1	Főoldal.....	40
1.11	Termékek oldal.....	42
1.11.1	termekesMegjelenit()	42
1.11.2	Megjelenítés	45

1.12	Kosár.....	46
1.13	Kedvencek.....	47
1.14	Belépés	48
1.15	Regisztráció.....	49
1.16	Konfiguráció összerakó.....	50
1.17	A konfiguráció megtekintő, és értékelő oldal	51
1.17.1	Card elemek	51
1.17.2	Ajax hívások	53
1.17.3	rating.php.....	54
1.17.4	limit.php.....	55
1.18	Tesztelés, tesztesetek	56
1.18.1	Unit tesztek:.....	56
1.18.2	Integrációs tesztek:	56
1.18.3	Funkcionalitási tesztek:.....	56
1.18.4	Megjelenés, felhasználói élmény tesztelése:	57
1.18.5	Regisztráció, bejelentkezés.....	57
1.18.6	Rendelés.....	59
1.19	Továbbfejlesztési lehetőségek	60
1.19.1	Felhasználói fiókok módosítása	60
1.19.2	E-mail	60
1.19.3	Ügyfélszolgálat.....	60
1.19.4	Rendelés visszaküldése	60
1.19.5	Akciók, leértékelések, kuponkód	61
1.19.6	Fizetési módok	61
1.19.7	Kompatibilitás és további elemek hozzáadása	61
1.19.8	Konfiguráció megosztó oldal továbbfejlesztése	61

2	Felhasználói dokumentáció	62
2.1	Bevezetés.....	62
2.2	Rendszerkövetelmények.....	62
2.3	Főoldal	63
2.4	Footer	63
2.5	Navigációs sáv	64
2.5.1	Termékek, Kategóriák	64
2.5.2	Regisztráció, Bejelentkezés.....	64
2.5.3	Konfigurációs rész	65
2.5.4	Kosár, kedvencek	65
2.6	Egy termék oldala.....	65
2.7	Kosár.....	66
2.8	Rendelés	66
2.9	Rendelési előzmények	67
2.10	Konfiguráció összeállító	68
2.11	Konfiguráció értékelő	68
3	Irodalomjegyzék	70
3.1	Internetes tartalmak	70
3.2	Könyvek	70
4	Melléklet.....	71

Témaválasztás

A vizsgaremekünk témájaként egy számítógép alkatrészeket, perifériákat, kiegészítőket és egyéb elektronikai eszközöket árusító webshopot választottunk. A tervezés első pillanataitól webes alkalmazásban gondolkodtunk. Rögtön egy számítógépes webshop volt az ötlete, ezért gyorsan neki tudtunk látni a munkálatoknak.

Nem akartuk, hogy a mi projektünk egy legyen a több ezer webshop között, ami már elkészült, ezért találtuk ki a számítógép konfiguráció építő, és megosztó részt is, hogy valamivel egyedivé és maradandóvá tegyük. A megosztott konfigurációkat mások láthatják és tudnak rá reagálni like, dislike formájában, szóval egy kezdetleges közösségi média oldalra hasonlítana. A megosztott konfigurációk között lehetne nézelődni, inspirációt szerezni és ezek alapján vásárolni is.

A weboldalt webshop részét Ruben készítette el. Itt megtalálhatóak kategóriák, amire kattintva a megfelelő termékeket láthatjuk. A termékekre rá tudunk keresni. Tudunk kedvencek közé és kosárba helyezni, továbbá rendelni és rendelési előzményeket megtekinteni.

Gábor feladata volt az adatbázis összeállítása, konfiguráció összeállítására és megosztására szolgáló oldalak elkészítése, regisztrációs és bejelentkező oldalak, valamint az adminisztrációs asztali alkalmazás létrehozása is.

1 Fejlesztői dokumentáció

1.1 Munkamegosztás

Dallos Ruben

- navigációs sáv
- kategóriák oldal
- termékek oldal
- egy termék oldala
- rendezés
- keresés
- kedvencek
- kosár
- rendelés

Herédi Gábor

- regisztráció oldal
- bejelentkezés oldal
- konfiguráció összeállító oldal
- konfiguráció megosztó oldal
- adminisztrátori felület (C#)

1.2 Vizsgaremek bemutatása

Vizsgaremekünket OOP (Objektumorientált Programozás) struktúra, és REST architektúra segítségével készítettük el. A fő szempont egy egyszerűen használható webshop és egy alap közösségi oldal. Törekedtünk a felhasználóbarát és modern megjelenésre.

1.3 Fejlesztői környezet

1.3.1 Visual Studio Code

A weboldal forráskódját a Visual Studio Code IDE-ben írtuk, mely a Microsoft egy ingyenes, nyílt forráskódú IDE felülete. Alapértelmezetten támogatja az alábbiak szerkesztését: HTML, JavaScript.¹



1.3.2 Visual Studio 2022

A Microsoft fejlesztőkörnyezete jelenleg számos programozási nyelvet támogat, például: F#, C++, C# és Visual Basic, valamint az XML-t is támogatja.²



1.3.3 XAMPP

Az XAMPP egy szoftvercsomag, amely segítségével a PHP nyelvhez szükséges helyi webservert hozza létre, amelyet az Apache old meg, továbbá a MySQL adatbázis elérése is a XAMPP használatával történik.³



¹ Forrás: https://hu.wikipedia.org/wiki/Visual_Studio_Code

² Forrás: https://hu.wikipedia.org/wiki/Microsoft_Visual_Studio

³ Forrás: <https://hu.wikipedia.org/wiki/XAMPP>

1.3.4 Teams

A Microsoft Teams a Microsoft által a Microsoft 365 termékcsalád részeként kifejlesztett csapatmunka-alkalmazás, amely munkaterületi csevegést és videokonferenciát, fájlátrolást, valamint saját és harmadik féltől származó alkalmazások integrációját kínálja.⁴



1.3.5 Github

A GitHub egy fejlesztői platform, amely lehetővé teszi a fejlesztők számára a kód létrehozását, tárolását, kezelését és megosztását. Git szoftvert használ, amely minden projekthez biztosítja a Git elosztott verziókezelését, plusz hozzáférés-vezérlést, hibakövetést, szoftverfunkciókéréseket, feladatkezelést, folyamatos integrációt és wikiket.

A kaliforniai székhelyű cég 2018 óta a Microsoft leányvállalata. Általában nyílt forráskódú szoftverfejlesztési projektek fogadására használják. 2023 januárjában a GitHub arról számolt be, hogy több mint 100 millió fejlesztővel és több mint 420 millió adattárral rendelkezik, köztük legalább 28 millió nyilvános adattárral. 2023 júniusában ez a világ legnagyobb forráskód-gazdája.



5

⁴ Forrás: https://en.wikipedia.org/wiki/Microsoft_Teams

⁵ Forrás: <https://hu.wikipedia.org/wiki/GitHub>

1.4 Felhasznált technológiák

1.4.1 HTML

A HTML (angolul: HyperText Markup Language, „hiperszoveges jelölőnyelv”) egy leíró nyelv, melyet weboldalak készítéséhez fejlesztettek ki, és mára már internetes szabvánnyá vált a W3C (World Wide Web Consortium) támogatásával. Az aktuális változata az 5.⁶



1.4.2 CSS

A CSS (Cascading Style Sheets, magyarul: „lépcsőzetes stíluslapok”) a számítástechnikában egy stíusleíró nyelv, mely a HTML vagy XHTML típusú strukturált dokumentumok megjelenését írja le. A CSS-t a weblapok szerkesztői és olvasói egyaránt használhatják, hogy átállítsák vele a lapok színét, betűtípusait, elrendezését, és más megjelenéshez kapcsolódó elemeit. A tervezése során a legfontosabb szempont az volt, hogy elkülönítsék a dokumentumok struktúráját (HTML vagy egy hasonló leíró nyelv) a dokumentum megjelenésétől (CSS).⁷



1.4.3 PHP

A PHP egy általános szerveroldali szkriptnyelv dinamikus weblapok készítésére. Az első szkriptnyelvek egyike, amely külső fájl használata helyett HTML oldalba ágyazható. A kódot a webszerver PHP feldolgozómodulja értelmezi, ezzel dinamikus weboldalakot hozva létre. Ma a The PHP Group tartja fenn és fejleszti.⁸



⁶ Forrás: <https://hu.wikipedia.org/wiki/HTML>

⁷ Forrás: <https://hu.wikipedia.org/wiki/CSS>

⁸ Forrás: <https://hu.wikipedia.org/wiki/PHP>

1.4.4 JavaScript

A JavaScript programozási nyelv egy objektumorientált, prototípus-alapú szkriptnyelv, amelyet weboldalakon elterjedten használnak. Ebből fejlődött ki a TypeScript, ami a JavaScript típusos változatának tekinthető.⁹



1.4.5 jQuery

A jQuery egy JavaScript-könyvtár, amelyet a HTML DOM-fa bejárásának és kezelésének, valamint az eseménykezelésnek, a CSS-animációknak és az AJAX-nak az egyszerűsítésére terveztek. Ez egy ingyenes, nyílt forráskódú szoftver, amely a megengedő MIT licencet használja. A jQuery szintaxisát úgy tervezték, hogy megkönnyítse a dokumentumban való navigálást, a DOM-elemek kiválasztását, az animációk létrehozását, az események kezelését és az AJAX-alkalmazások fejlesztését.¹⁰



1.4.6 AJAX

Az Ajax (Asynchronous JavaScript and XML) interaktív webalkalmazások létrehozására szolgáló webfejlesztési technika. Segítségével a weblap kis mennyiségű adatot cserél a szerverrel a háttérben, így a lapot nem kell újratölteni minden egyes alkalommal, amikor a felhasználó módosít valamit. Ez növeli a honlap interaktivitását, sebességét és használhatóságát.¹¹



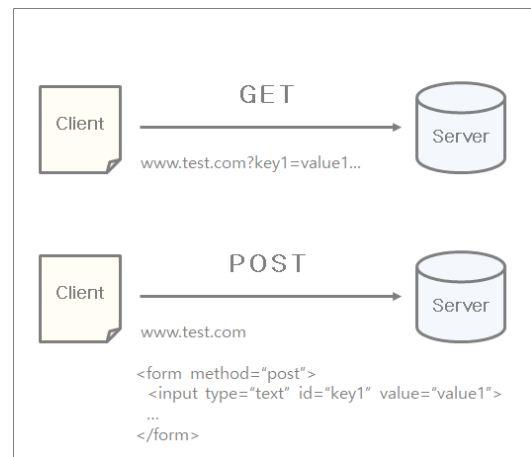
⁹ Forrás: <https://hu.wikipedia.org/wiki/JavaScript>

¹⁰ Forrás: <https://hu.wikipedia.org/wiki/JQuery>

¹¹ Forrás: [https://hu.wikipedia.org/wiki/Ajax_\(programoz%C3%A1s\)](https://hu.wikipedia.org/wiki/Ajax_(programoz%C3%A1s))

1.4.7 REST API

A REST (Representational State Transfer) szoftverarchitektúra típus. Egy REST típusú architektúra kliensekből és szerverekből áll. A kliensek kéréseket indítanak a szerverek felé, a szerverek kéréseket dolgoznak fel és a megfelelő választ küldik vissza. A kérések és a válaszok erőforrás-reprezentációk szállítása köré épülnek. Az erőforrás lényegében bármilyen koherens és értelmesen címezhető koncepció lehet.¹²



1.4.8 MySQL

A MySQL egy többfelhasználós, többszálú, SQL-alapú relációs adatbázis-kezelő szerver.

A szoftver eredeti fejlesztője a svéd MySQL AB cég, amely kettős licenccel tette elérhetővé a MySQL-t; választható módon vagy a GPL szabad szoftver licenc, vagy egy zárt (tulajdonosi) licenc érvényes a felhasználásra. 2008 januárjában a Sun felvásárolta 800 millió dollárért a céget. 2010. január 27-én a Sun felvásárolta az Oracle Corporation, így a MySQL is az Oracle tulajdonába került.¹³



¹² Forrás: <https://hu.wikipedia.org/wiki/REST>

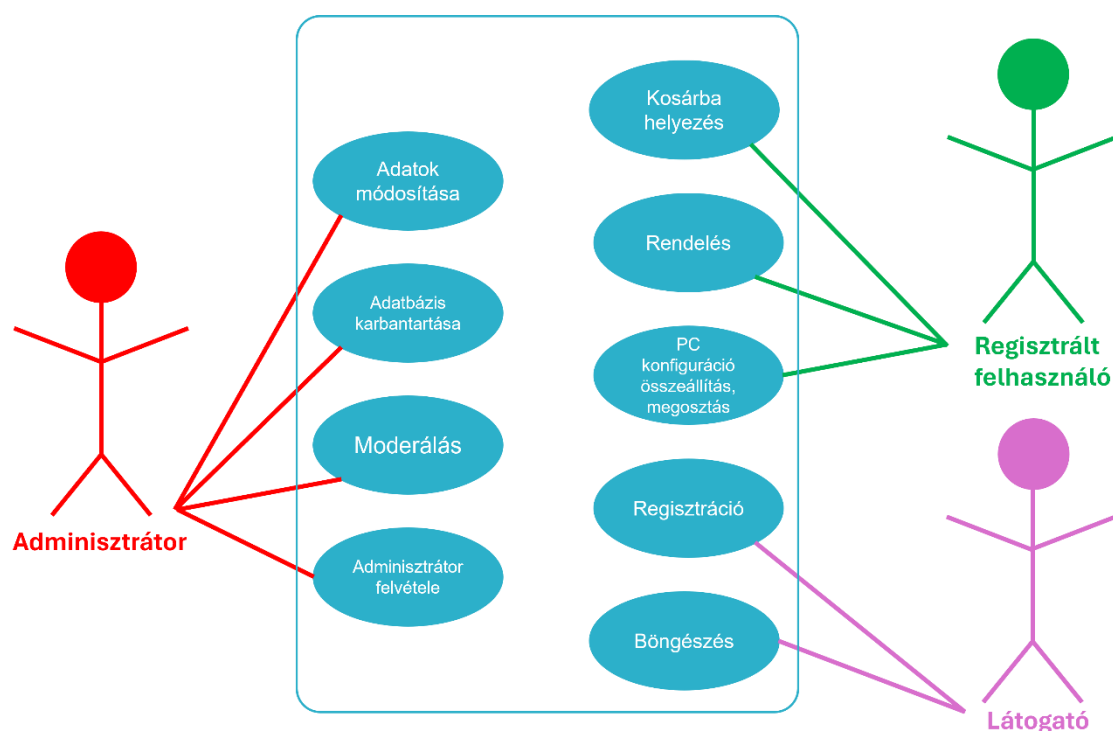
¹³ Forrás: <https://hu.wikipedia.org/wiki/MySQL>

1.4.9 C#

A C# egy általános célú, magas szintű programozási nyelv, amely több paradigmát is támogat. A C# magába foglalja a statikus gépelést, az erős gépelést, a lexikális hatókörű, kötelező, deklaratív, funkcionális, általános, objektum-orientált (osztály-alapú) és komponens-orientált programozási tudományokat.¹⁴



1.5 Használati esetmodell



¹⁴ Forrás https://hu.wikipedia.org/wiki/C_Sharp

1.6 A program felépítése

A **'TornadoPC'** mappa tartalmazza a weboldal működéséhez létfontosságú alkotóelemeket például: minden szükséges PHP fájlt és több almappát is.

'css' almappában található egy CSS fájl, amely minden oldal modern és reszponzív megjelenéséért felel a Bootstrap 5-tel együttműködve.

'js' almappa tartalmazza a főként jQuery fájlokat, amelyek segítségével hozunk létre AJAX hívásokat, kosárba vagy kedvencek közé helyezést és törlést is általuk végezzük, illetve a kosárban található termék mennyiségének növelésére, csökkentésére is itt található jQuery scripteket használunk.

A **'kepek'** mappa további almappákat és képeket is tartalmaz. Az almappákban megtalálható képek vannak megjelenítve a weboldalon, az adatbázisban a képeknek csak az elérési útvonala van elmentve és megjelenítésre felhasználva. Egyik almappa a **'kategóriák'**, amelyben az egyes kategóriák képei tekinthetők meg. Másik almappa a **'termékek'**, amelyben a termék neve a termék azonosítóját kapta, amely egyszerűsíti a képek feltöltését, feldolgozását. A többi kép, amelyek nincsenek almappába helyezve, több helyen is megjelenhetnek, például a kosár, kedvencek, keresés gombok ikonjai és a háttérkép is itt szerepel.

1.7 Adatbázis

10.4.25-MariaDB

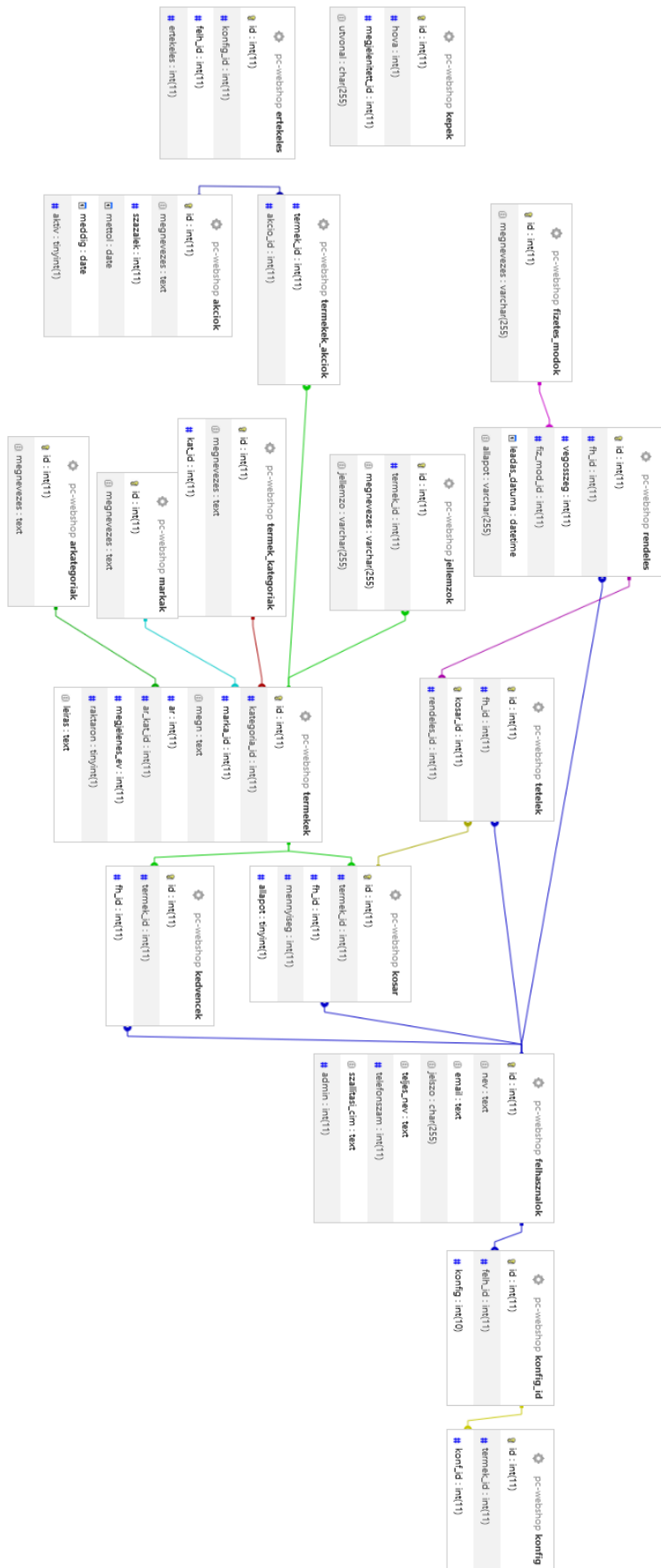
Adatbázis neve: tornado-pc

Tárolómotor: InnoDB

Alapértelmezett illesztés: utf8_hungarian_ci

Adattáblák:

- akciok
- arkategoriak
- ertekeles
- felhasznalok
- fizetes_modok
- jellemzok
- kedvencek
- kepek
- konfig
- konfig_id
- kosar
- markak
- rendeles
- termekek
- termekek_akciok
- termék kategoriak
- tetelek



1.7.1 akciók tábla

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
<input type="checkbox"/> 1	id	int(11)			Nem	Nincs		AUTO_INCREMENT	Módosítás Eldobás Több
<input type="checkbox"/> 2	megnevezes	text	utf8_hungarian_ci		Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/> 3	szazalek	int(11)			Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/> 4	mettol	date			Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/> 5	meddig	date			Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/> 6	aktiv	tinyint(1)			Nem	Nincs			Módosítás Eldobás Több

id: A tábla azonosítója, elsődleges kulcs

megnevezes: Az akció megnevezése

szazalek: Az akció értéke

mettol: Az akció kezdetének dátuma

meddig: Az akció végének dátuma

aktiv: Az akció érvényességének meghatározója

```
CREATE TABLE `akciok` (  
  `id` int(11) NOT NULL,  
  `megnevezes` text NOT NULL,  
  `szazalek` int(11) NOT NULL,  
  `mettol` date NOT NULL,  
  `meddig` date NOT NULL,  
  `aktiv` tinyint(1) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;
```

```
ALTER TABLE `akciok`  
  ADD PRIMARY KEY (`id`);
```

```
ALTER TABLE `akciok`  
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
```

Ez a funkció még nincs használatban, de felkészítettük rá az adatbázist.

1.7.2 arkategoriak tábla

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
<input type="checkbox"/>	1	id	int(11)		Nem	Nincs		AUTO_INCREMENT	Módosítás Eldobás Több
<input type="checkbox"/>	2	megnevezes	text	utf8_hungarian_ci	Nem	Nincs			Módosítás Eldobás Több

id: A tábla azonosítója, elsődleges kulcs

megnevezes: Az árkategória megnevezése

```
CREATE TABLE `arkategoriak` (  
    `id` int(11) NOT NULL,  
    `megnevezes` text NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;  
ALTER TABLE `arkategoriak`  
    ADD PRIMARY KEY (`id`);  
ALTER TABLE `arkategoriak`  
    MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
```

1.7.3 értékes tábla

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
<input type="checkbox"/>	1 id	int(11)			Nem	Nincs		AUTO_INCREMENT	Módosítás Eldobás Több
<input type="checkbox"/>	2 konfigur_id	int(11)			Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/>	3 felh_id	int(11)			Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/>	4 értékes	int(11)			Nem	Nincs			Módosítás Eldobás Több

id: A tábla azonosítója, elsődleges kulcs

konfig_id: Az értékelt konfiguráció azonosítója

felh_id: Az értékelő felhasználó azonosítója

ertekeles: Az értékelés értéke, 1 = like, 2 = dislike

```
CREATE TABLE `ertekeles` (  
  `id` int(11) NOT NULL,  
  `konfig_id` int(11) NOT NULL,  
  `felh_id` int(11) NOT NULL,  
  `ertekeles` int(11) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;  
ALTER TABLE `ertekeles`  
  ADD PRIMARY KEY (`id`);  
ALTER TABLE `ertekeles`  
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
```

1.7.4 felhasználók tábla

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
<input type="checkbox"/> 1	id	int(11)			Nem	Nincs		AUTO_INCREMENT	Módosítás Eldobás Több
<input type="checkbox"/> 2	nev	text	utf8_hungarian_ci		Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/> 3	email	text	utf8_hungarian_ci		Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/> 4	jelszo	char(255)	utf8_hungarian_ci		Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/> 5	teljes_nev	text	utf8_hungarian_ci		Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/> 6	telefonszam	int(11)			Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/> 7	szallitasi_cim	text	utf8_hungarian_ci		Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/> 8	admin	int(11)			Nem	Nincs			Módosítás Eldobás Több

id: A tábla azonosítója, elsődleges kulcs

nev: A felhasználó megadott felhasználóneve

email: A felhasználó e-mail címe

jelszo: A felhasználó jelszava sha1 algoritmussal titkosítva

teljes_nev: A felhasználó teljes neve

telefonszam: A felhasználó telefonszáma

szallitasi_cim: A felhasználó szállítási címe

admin: A felhasználó státusza, 0 = átlagos felhasználó, 1 = admin

```
CREATE TABLE `felhasznalok` (  
  `id` int(11) NOT NULL,  
  `nev` text NOT NULL,  
  `email` text NOT NULL,  
  `jelszo` char(255) NOT NULL,  
  `teljes_nev` text NOT NULL,  
  `telefonszam` int(11) NOT NULL,  
  `szallitasi_cim` text NOT NULL,  
  `admin` int(11) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;  
ALTER TABLE `felhasznalok`  
  ADD PRIMARY KEY (`id`);  
ALTER TABLE `felhasznalok`  
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
```

1.7.5 fizetes_modok tábla

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
<input type="checkbox"/> 1	id	int(11)			Nem	Nincs		AUTO_INCREMENT	Módosítás Eldobás Több
<input type="checkbox"/> 2	megnevezes	varchar(255)	utf8_hungarian_ci		Nem	Nincs			Módosítás Eldobás Több

id: A tábla azonosítója, elsődleges kulcs

megnevezes: A fizetési mód megnevezése

```
CREATE TABLE `fizetes_modok` (  
  `id` int(11) NOT NULL,  
  `megnevezes` varchar(255) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;  
INSERT INTO `fizetes_modok` (`id`, `megnevezes`) VALUES  
(1, 'készpénz - utánvét'),  
(2, 'bankkártya - utánvét'),  
ALTER TABLE `fizetes_modok`  
  ADD PRIMARY KEY (`id`);  
ALTER TABLE `fizetes_modok`  
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
```


1.7.6 jellemzok tábla

#	Név	Típus	Indexelés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
<input type="checkbox"/>	1 id 	int(11)			Nem	Nincs		AUTO_INCREMENT	 Módosítás  Eldobás Több
<input type="checkbox"/>	2 termék_id 	int(11)			Nem	Nincs			 Módosítás  Eldobás Több
<input type="checkbox"/>	3 megnevezes	varchar(255)	utf8_hungarian_ci		Nem	Nincs			 Módosítás  Eldobás Több
<input type="checkbox"/>	4 jellemzo	varchar(255)	utf8_hungarian_ci		Nem	Nincs			 Módosítás  Eldobás Több

id: A tábla azonosítója, elsődleges kulcs

termek_id: A termék azonosítója, külső kulcs

megnevezes: A jellemző megnevezése

jellemzo: A jellemző leírása

```
CREATE TABLE `jellemzok` (  
  `id` int(11) NOT NULL,  
  `termek_id` int(11) NOT NULL,  
  `megnevezes` varchar(255) NOT NULL,  
  `jellemzo` varchar(255) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;  
ALTER TABLE `jellemzok`  
  ADD PRIMARY KEY (`id`),  
  ADD KEY `termek_id` (`termek_id`);  
ALTER TABLE `jellemzok`  
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;  
ALTER TABLE `jellemzok`  
  ADD CONSTRAINT `jellemzok_ibfk_1` FOREIGN KEY (`termek_id`) REFERENCES `termekek` (`id`);
```

1.7.7 kedvencek tábla

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
<input type="checkbox"/>	1 id	int(11)			Nem	Nincs		AUTO_INCREMENT	Módosítás Eldobás Több
<input type="checkbox"/>	2 termék_id	int(11)			Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/>	3 fh_id	int(11)			Nem	Nincs			Módosítás Eldobás Több

id: A tábla azonosítója, elsődleges kulcs

termék_id: A termék azonosítója, külső kulcs

fh_id: A felhasználó azonosítója, külső kulcs

```
CREATE TABLE `kedvencek` (  
  `id` int(11) NOT NULL,  
  `termék_id` int(11) NOT NULL,  
  `fh_id` int(11) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;  
ALTER TABLE `kedvencek`  
  ADD PRIMARY KEY (`id`),  
  ADD KEY `termék_id` (`termék_id`),  
  ADD KEY `fh_id` (`fh_id`);  
ALTER TABLE `kedvencek`  
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;  
ALTER TABLE `kedvencek`  
  ADD CONSTRAINT `kedvencek_ibfk_1` FOREIGN KEY (`termék_id`) REFERENCES `termekek` (`id`),  
  ADD CONSTRAINT `kedvencek_ibfk_2` FOREIGN KEY (`fh_id`) REFERENCES `felhasznalok` (`id`);
```

1.7.8 kepek tábla

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
<input type="checkbox"/>	1	id	int(11)		Nem	Nincs		AUTO_INCREMENT	Módosítás Eldobás Több
<input type="checkbox"/>	2	hova	int(1)		Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/>	3	megjelenített_id	int(11)		Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/>	4	utvonal	char(255) <i>utf8_hungarian_ci</i>		Nem	Nincs			Módosítás Eldobás Több

id: A tábla azonosítója, elsődleges kulcs













hova: A kép megjelenítésének helye, 1 = kategória, 2 = termék

megjelenített_id: A kép megjelenítéséhez szükséges azonosító

utvonal: A kép elérési útvonala

```
CREATE TABLE `kepek` (  
  `id` int(11) NOT NULL,  
  `hova` int(1) NOT NULL,  
  `megjelenített_id` int(11) NOT NULL,  
  `utvonal` char(255) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci  
ALTER TABLE `kepek`  
  ADD PRIMARY KEY (`id`);  
ALTER TABLE `kepek`  
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
```

1.7.9 konfigurációs tábla

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
<input type="checkbox"/>	1 id 	int(11)			Nem	Nincs		AUTO_INCREMENT	 Módosítás  Eldobás  Több
<input type="checkbox"/>	2 termék_id 	int(11)			Nem	Nincs			 Módosítás  Eldobás  Több
<input type="checkbox"/>	3 konf_id 	int(11)			Nem	Nincs			 Módosítás  Eldobás  Több

id: A tábla azonosítója, elsődleges kulcs

termék_id: A konfigurációhoz tartozó termék azonosítója, külső kulcs

konf_id: A konfiguráció azonosítója, amihez a termék tartozik, külső kulcs

```
CREATE TABLE `konfig` (  
  `id` int(11) NOT NULL,  
  `termék_id` int(11) NOT NULL,  
  `konf_id` int(11) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;  
ALTER TABLE `konfig`  
  ADD PRIMARY KEY (`id`),  
  ADD KEY `termék_id` (`termék_id`, `konf_id`),  
  ADD KEY `konf_id` (`konf_id`);  
ALTER TABLE `konfig`  
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;  
ALTER TABLE `konfig`  
  ADD CONSTRAINT `konfig_ibfk_1` FOREIGN KEY (`konf_id`)  
  REFERENCES `konfig` (`id`);
```

1.7.10 konfigur_id tábla

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
<input type="checkbox"/>	1	id	int(11)		Nem	Nincs		AUTO_INCREMENT	Módosítás Eldobás Több
<input type="checkbox"/>	2	felh_id	int(11)		Nem	Nincs			Módosítás Eldobás Több

id: A konfiguráció azonosítója, elsődleges kulcs

felh_id: A konfiguráció melyik felhasználóhoz tartozik, külső kulcs

```
CREATE TABLE `konfig_id` (  
  `id` int(11) NOT NULL,  
  `felh_id` int(11) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;  
ALTER TABLE `konfig_id`  
  ADD PRIMARY KEY (`id`),  
  ADD KEY `felh_id` (`felh_id`);  
ALTER TABLE `konfig_id`  
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;  
ALTER TABLE `konfig_id`  
  ADD CONSTRAINT `konfig_id_ibfk_1` FOREIGN KEY (`felh_id`) REFERENCES `felhasznalok` (`id`);
```

1.7.11 kosar tábla

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
<input type="checkbox"/>	1 id	int(11)			Nem	Nincs		AUTO_INCREMENT	Módosítás Eldobás Több
<input type="checkbox"/>	2 termék_id	int(11)			Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/>	3 fh_id	int(11)			Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/>	4 mennyiség	int(11)			Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/>	5 állapot	tinyint(1)			Nem	Nincs			Módosítás Eldobás Több

id: A tábla azonosítója, elsődleges kulcs

termek_id: A termék azonosítója, külső kulcs

fh_id: A felhasználó azonosítója, külső kulcs

mennyiség: Az adott termék mennyisége

allapot: A kosárban levő termék állapota, 0 = nem megrendelt (megjelenik kosárban), 1 = megrendelt

```
CREATE TABLE `kosar` (  
  `id` int(11) NOT NULL,  
  `termek_id` int(11) NOT NULL,  
  `fh_id` int(11) NOT NULL,  
  `mennyiség` int(11) NOT NULL,  
  `allapot` tinyint(1) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;  
ALTER TABLE `kosar`  
  ADD PRIMARY KEY (`id`),  
  ADD KEY `termek_id` (`termek_id`),  
  ADD KEY `fh_id` (`fh_id`);  
ALTER TABLE `kosar`  
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;  
ALTER TABLE `kosar`  
  ADD CONSTRAINT `kosar_ibfk_3` FOREIGN KEY (`termek_id`) REFERENCES `termekek` (`id`),  
  ADD CONSTRAINT `kosar_ibfk_4` FOREIGN KEY (`fh_id`) REFERENCES `felhasznalok` (`id`);
```

1.7.12 markak tábla

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
<input type="checkbox"/>	1	id	int(11)		Nem	Nincs		AUTO_INCREMENT	Módosítás Eldobás Több
<input type="checkbox"/>	2	megnevezes	text	utf8_hungarian_ci	Nem	Nincs			Módosítás Eldobás Több

id: A tábla azonosítója, elsődleges kulcs

megnevezes: A márka megnevezése

```
CREATE TABLE `markak` (  
  `id` int(11) NOT NULL,  
  `megnevezes` text NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;  
ALTER TABLE `markak`  
  ADD PRIMARY KEY (`id`);  
ALTER TABLE `markak`  
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
```

1.7.13 rendeles tábla

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
<input type="checkbox"/>	1 id	int(11)			Nem	Nincs		AUTO_INCREMENT	Módosítás Eldobás Több
<input type="checkbox"/>	2 fh_id	int(11)			Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/>	3 vegosszeg	int(11)			Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/>	4 fiz_mod_id	int(11)			Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/>	5 leadas_datuma	datetime			Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/>	6 allapot	varchar(255)	utf8_hungarian_ci		Nem	Nincs			Módosítás Eldobás Több

id: A tábla azonosítója, elsődleges kulcs

fh_id: A felhasználó azonosítója, külső kulcs

vegosszeg: A fizetendő végösszeg

fiz_mod_id: A fizetési mód azonosítója, külső kulcs

leadas_datuma: A leadás dátuma

allapot: A rendelés állapota

```
CREATE TABLE `rendeles` (  
  `id` int(11) NOT NULL,  
  `fh_id` int(11) NOT NULL,  
  `vegosszeg` int(11) NOT NULL,  
  `fiz_mod_id` int(11) NOT NULL,  
  `leadas_datuma` datetime NOT NULL,  
  `allapot` varchar(255) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;  
ALTER TABLE `rendeles`  
  ADD PRIMARY KEY (`id`),  
  ADD KEY `fiz_mod_id` (`fiz_mod_id`),  
  ADD KEY `fh_id` (`fh_id`);  
ALTER TABLE `rendeles`  
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;  
ALTER TABLE `rendeles`  
  ADD CONSTRAINT `rendeles_ibfk_3` FOREIGN KEY  
  (`fiz_mod_id`) REFERENCES `fizetes_modok` (`id`),  
  ADD CONSTRAINT `rendeles_ibfk_4` FOREIGN KEY (`fh_id`)  
  REFERENCES `felhasznalok` (`id`);
```


1.7.14 termékek tábla

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
<input type="checkbox"/>	1 id 	int(11)			Nem	Nincs		AUTO_INCREMENT	 Módosítás  Eldobás Több
<input type="checkbox"/>	2 kategoria_id 	int(11)			Nem	Nincs			 Módosítás  Eldobás Több
<input type="checkbox"/>	3 marka_id 	int(11)			Nem	Nincs			 Módosítás  Eldobás Több
<input type="checkbox"/>	4 megn	text	utf8_hungarian_ci		Nem	Nincs			 Módosítás  Eldobás Több
<input type="checkbox"/>	5 ar	int(11)			Nem	Nincs			 Módosítás  Eldobás Több
<input type="checkbox"/>	6 ar_kat_id 	int(11)			Nem	Nincs			 Módosítás  Eldobás Több
<input type="checkbox"/>	7 megjelenes_ev	int(11)			Nem	Nincs			 Módosítás  Eldobás Több
<input type="checkbox"/>	8 raktaron	tinyint(1)			Nem	Nincs			 Módosítás  Eldobás Több
<input type="checkbox"/>	9 leiras	text	utf8_hungarian_ci		Nem	Nincs			 Módosítás  Eldobás Több

id: A tábla azonosítója, elsődleges kulcs

kategoria_id: A kategória azonosítója, külső kulcs

marka_id: A termék márkájának azonosítója, külső kulcs

megn: A termék megnevezése

ar_kat_id: A termék árkategóriájának azonosítója, külső kulcs

megjelenes_ev: A termék megjelenésének éve

raktaron: A termékből van-e raktáron? (Még nem működő funkció, fejlesztés alatt)

leiras: A termék leírása

```

CREATE TABLE `termekek` (
  `id` int(11) NOT NULL,
  `kategoria_id` int(11) NOT NULL,
  `marka_id` int(11) NOT NULL,
  `megn` text NOT NULL,
  `ar` int(11) NOT NULL,
  `ar_kat_id` int(11) NOT NULL,
  `megjelenes_ev` int(11) NOT NULL,
  `raktaron` tinyint(1) NOT NULL,
  `leiras` text NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungar-
ian_ci;
ALTER TABLE `termekek`
  ADD PRIMARY KEY (`id`),
  ADD KEY `kategoria_id` (`kategoria_id`),
  ADD KEY `marka_id` (`marka_id`),
  ADD KEY `ar_kat_id` (`ar_kat_id`);
ALTER TABLE `termekek`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
ALTER TABLE `termekek`
  ADD CONSTRAINT `termekek_ibfk_1` FOREIGN KEY (`marka_id`)
REFERENCES `markak` (`id`),
  ADD CONSTRAINT `termekek_ibfk_3` FOREIGN KEY (`katego-
ria_id`) REFERENCES `termek_kategoriak` (`id`),
  ADD CONSTRAINT `termekek_ibfk_4` FOREIGN KEY
(`ar_kat_id`) REFERENCES `arkategoriak` (`id`);

```

1.7.15 termekek_akciok








#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
<input type="checkbox"/>	1 termék_id 🔑	int(11)			Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/>	2 akcio_id 🔑	int(11)			Nem	Nincs			Módosítás Eldobás Több

termek_id: A termék azonosítója, külső kulcs

akcio_id: Az akció azonosítója, külső kulcs

```
CREATE TABLE `termekek_akciok` (  
  `termek_id` int(11) NOT NULL,  
  `akcio_id` int(11) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;  
ALTER TABLE `termekek_akciok`  
  ADD KEY `termek_id` (`termek_id`),  
  ADD KEY `akcio_id` (`akcio_id`);  
ALTER TABLE `termekek_akciok`  
  ADD CONSTRAINT `termekek_akciok_ibfk_2` FOREIGN KEY  
  (`termek_id`) REFERENCES `termekek` (`id`),  
  ADD CONSTRAINT `termekek_akciok_ibfk_3` FOREIGN KEY (`ak-  
cio_id`) REFERENCES `akciok` (`id`);
```

1.7.16 termék_kategoriak tabla

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
<input type="checkbox"/>	1 id 	int(11)			Nem	Nincs		AUTO_INCREMENT	 Módosítás  Eldobás Több
<input type="checkbox"/>	2 megnevezes	text	utf8_hungarian_ci		Nem	Nincs			 Módosítás  Eldobás Több
<input type="checkbox"/>	3 kat_id	int(11)			Nem	Nincs			 Módosítás  Eldobás Több

id: A tábla azonosítója, elsődleges kulcs

megnevezes: A kategória megnevezése

kat_id: A kategóriának megadott azonosító

```
CREATE TABLE `termek_kategoriak` (  
  `id` int(11) NOT NULL,  
  `megnevezes` text NOT NULL,  
  `kat_id` int(11) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;  
ALTER TABLE `termek_kategoriak`  
  ADD PRIMARY KEY (`id`);  
ALTER TABLE `termek_kategoriak`  
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
```

1.7.17 tetelek tábla

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
<input type="checkbox"/> 1	id	int(11)			Nem	Nincs		AUTO_INCREMENT	Módosítás Eldobás Több
<input type="checkbox"/> 2	fh_id	int(11)			Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/> 3	kosar_id	int(11)			Nem	Nincs			Módosítás Eldobás Több
<input type="checkbox"/> 4	rendeles_id	int(11)			Nem	Nincs			Módosítás Eldobás Több

id: A tábla azonosítója, elsődleges kulcs

fh_id: A felhasználó azonosítója, külső kulcs

kosar_id: A kosár azonosítója, külső kulcs

rendeles_id: A rendelés azonosítója, külső kulcs

```
CREATE TABLE `tetelek` (  
  `id` int(11) NOT NULL,  
  `fh_id` int(11) NOT NULL,  
  `kosar_id` int(11) NOT NULL,  
  `rendeles_id` int(11) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;  
ALTER TABLE `tetelek`  
  ADD PRIMARY KEY (`id`),  
  ADD UNIQUE KEY `kosar_id` (`kosar_id`),  
  ADD KEY `rendeles_id` (`rendeles_id`),  
  ADD KEY `fh_id` (`fh_id`);  
ALTER TABLE `tetelek`  
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;  
ALTER TABLE `tetelek`  
  ADD CONSTRAINT `tetelek_ibfk_1` FOREIGN KEY (`kosar_id`) REFERENCES `kosar` (`id`),  
  ADD CONSTRAINT `tetelek_ibfk_3` FOREIGN KEY (`rendeles_id`) REFERENCES `rendeles` (`id`),  
  ADD CONSTRAINT `tetelek_ibfk_4` FOREIGN KEY (`fh_id`) REFERENCES `felhasznalok` (`id`);
```

1.8 Rendszerkövetelmények

Mesterremekünk webshop lévén, szükség van egy mai modern böngészőre, lehet ez Google Chrome, Mozilla Firefox, Opera és Opera GX is. Szükséges lehet néhány bővítmény telepítése a technológiák miatt, és a JavaScript futtatását is engedélyezni kell. A webes alkalmazás egy adatbázis-kezelő rendszert is használ, amely telepítése szükséges a helyi vagy webszerverre, amiről az alkalmazást szeretnénk futtatni. Futtatáshoz szükséges vezetékes, vagy vezeték nélküli internetkapcsolat. Alkalmazásunkat helyi szerveren fogjuk bemutatni, ezért szükségünk van egy webalkalmazás kiszolgáló helyi alkalmazásra, ami az Apache http szerver, a MySQL adatbázis kezelő rendszer és ezen kívüli más komponensek integrált csomagja, ami jelen esetben a Xampp lesz.

Követelmények a Xampp által biztosított webszerverhez:

- Windows, Linux vagy Mac OS operációs rendszer
- 5-8 GB tárhely (Minél több új terméket adunk kínálatunkhoz, úgy változhat a tárhely igény)
- Internetböngésző, javasolt a Mozilla Firefox vagy Google Chrome
- MySQL legfrissebb verziója
- PHP legfrissebb verziója

1.9 Megfelelő indítás

Alkalmazásunk a dokumentáció írásakor még nem található meg semmilyen webszerveren, ezért egy GitHub link szükséges a letöltéshez. Az előbb említett Xampp legfrissebb verzióját is szükséges telepíteni az alkalmazásunk pontos és megfelelő működéséhez. A Xampp magában foglalja az Apache http szerveret, és a MySQL adatbáziskezelő rendszert is. Ez egy egyszerű telepítő-programmal, néhány kattintással telepíthető egyszerű folyamat.

1. A GitHubos linkről letöltjük a forrásfájlokat, amiket a Xampp miatt szigorúan, a C meghajtó 'xampp' mappáján belül a 'htdocs' mappába helyezzük el.
2. Elindítjuk a Xampp Control Panelt, ezen belül pedig, az Apache és MySQL modulokat.
3. Böngészőben meg kell nyitnunk a <http://localhost/phpmyadmin/> oldalt, és importálnunk kell a 'tornado-pc.sql' fájlt.
4. Végül elindítjuk az alkalmazásunkat. Beírjuk a címet: 'localhost', és kész. Webes alkalmazásunk készen áll a bemutatásra!

A webszerveren vagy localhost-on való futtatás ugyanúgy működik, nincs köztük működésbeli különbség, csak furcsa lehet, hogy egy gépről megy a rendelés, és az adminisztrátori funkciók is ott működnek. A védés idejére igyekszünk oldalunk egy webcímen való megosztására.

1.10A weboldal felépítése

A weboldal az objektum orientált programozás és a tiszta kód elvei szerint készült. Több osztályt használunk a webshop működéséhez. 'Webshop' osztály tartalmazza a webshop működéséhez szolgáló metódusokat. 'DB' osztályban találhatóak az adatbázis csatlakozást létrehozó és az SQL utasítások futtatására használható metódusok. 'Felhasznalo' osztály a regisztráló és bejelentkező űrlapok megjelenítésére és feldolgozására szolgáló metódusokat tartalmazza. A 'Rendeles' osztályban pedig a rendeléshez szükséges ada-

tok bekérő és feldolgozó, illetve a rendelési előzményeket megjelenítő metódusok vannak elhelyezve. HTML, PHP és JavaScript nyelveket használtunk, a frontend saját CSS és Bootstrap 5 segítségével valósítottuk meg, a reszponzivitásra is odafigyelve.

1.10.1 Főoldal

```
<head>
|   <?php $webshop->head(); ?>
</head>

<body>
|   <?php $webshop->navbar(1); ?>
|
|   <div class="container-fluid tartalomnak">
|       <div class="content">
|           <?php
|               $webshop->kategoriakMegjelenit();
|               if (isset ($_POST['kategoria_id'])) {
|                   header('Location: termek.php');
|               }
|           ?>
|       </div>
|   </div>
|
|   <footer>
|       <?php $webshop->footer(); ?>
|   </footer>
```

Létrehoztunk az 'index.php' fájlban egy '\$webshop' objektumot és ezen keresztül illesztjük be a 'head()' metódus által a <head> részben szükséges adatok és például a saját CSS és Bootstrap linkelés is, így, ha bármilyen változtatást szükséges eszközölnünk, akkor elég lesz csak a metódusban egyszer módosítani, nem kell minden oldalon. Továbbá ilyen módon jelenik meg a navigációs sáv, a kategóriák és a lábléc is.

A kategóriák három nagyobb főkategóriába vannak osztva, a három főkategória alá tartozó kategóriák kártyái egy-egy lenyíló részen belül tekinthetők meg, ahonnan tovább lehet navigálni az adott kategóriába tartozó termékekhez. Ugyanez a három főkategória a további kategóriáival a navigációs sávon is elérhető. Minden kategória egyedi ID-vel, azonosítóval rendelkezik, GET kérésen keresztül paraméterként átadásra kerül ez az ID és megjelennek a termékek egy-egy kártyán.


```

public function accordion($cime, $kat_id = null)
{
    $sqlKategoriak = "SELECT * FROM termek_kategoriak
                    WHERE kat_id = $kat_id
                    ORDER BY 2";

    $kategoriak = $this->sqlAssoc($sqlKategoriak);
    echo "
<div class='accordion-item'>
  <h2 class='accordion-header'>";
    if ($kat_id == 1) {
        $collapse_id = "collapseOne";
        $aria_expanded = true;
        $show = "show";
    } elseif ($kat_id == 2) {
        $collapse_id = "collapseTwo";
        $aria_expanded = false;
    } elseif ($kat_id == 3) {
        $collapse_id = "collapseThree";
        $aria_expanded = false;
    }
    echo "
    <button class='accordion-button collapsed fw-bolder' type='button' data-bs-toggle='collapse' data-bs-target='#$collapse_id'
    aria-expanded='$aria_expanded' aria-controls='$collapse_id'>
      $cime
    </button>
  </h2>
  <div id='$collapse_id' class='accordion-collapse collapse'>
    if (isset($show))
      echo "$show";
    echo "' data-bs-parent='#accordionExample'>
    <div class='accordion-body'>";

    echo "<div class='row'>";
    foreach ($kategoriak as $kategoria) {
        echo "
        <div class='col-sm-6 col-lg-4 col-xl-3 my-3'>
          <div class='card mb-4 h-100 shadow-sm border border-info'>
            ";
        $this->image($kategoria['id'], "kat");
        echo "<div class='card-body'>
          <div class='clearfix mb-3'>
            <span class='kategorianak kozepre float-start border border-info'>{$kategoria['megnevezes']}</span>
          </div>
          <a href='termekek.php?kategoria_id={$kategoria['id']}' class='btn border-info d-block w-50 mx-auto'>Tovább</a>";
        echo "
        </div>
      </div>
    </div>";
    }
    echo "
  </div>
</div>";
}

public function kategoriakMegjelenit()
{
    echo "
    <div class='accordion cucc' id='accordionExample'>";
    $this->accordion('PC Alkatrészek', 1);
    $this->accordion('Perifériák', 2);
    $this->accordion('Egyéb', 3);
    echo "
  </div>";
}

```

1.11Termékek oldal

1.11.1 termekekMegjelenit()

Ez a metódus a következő funkciókat látja el:

- rendezés
- lapozás
- adatbázisból lekérdezzük a termék adatait
- keresés
- megjelenítés kártyákkal

Bemeneti paraméternek meg kell adni a jelenlegi oldal számát (\$p). Két opcionális paramétert fogad: '\$kategoria_id', a kiválasztott kategória ID-je, és '\$keresett', ami a keresett kulcsszó lesz.

Tartalmaz három különböző lekérdezést, amelyek azt a célt szolgálják, hogy három funkció közül a megfelelőt tudjuk hasznosítani (összes, kiválasztott kategória és keresett kulcsszó termékei).

```
public function termekekMegjelenit($p, $kategoria_id = null, $keresett = null)
{
    $rendez = "2"; //alapértelmezetten abc sorrend
    if (isset($_SESSION['rendez']) && isset($_POST['rendez'])) {
        $_SESSION['rendez'] = $_POST['rendez'];
    } elseif ((isset($_POST['rendez'])) && (!isset($_SESSION['rendez']))) {
        $_SESSION['rendez'] = '0';
    }
    echo "
    <form id='rendezForm' method='POST' action='{$_SERVER['REQUEST_URI']}'>
    <select class='form-select w-25' name='rendez' onchange='rendezeshez()'>
        <option value='0' ">
            if ($_SESSION['rendez'] == '0') {
                echo 'selected';
            }
            echo ">ABC sorrend</option>
        <option value='1' ">
            if ($_SESSION['rendez'] == '1') {
                echo 'selected';
            }
            echo ">Ár csökkenő</option>
        <option value='2' ">
            if ($_SESSION['rendez'] == '2') {
                echo 'selected';
            }
            echo ">Ár növekvő</option>
    </select>
    </form>";

    if (isset($_SESSION['rendez'])) {
        switch ($_SESSION['rendez']) {
            case '0':
                $rendez = 2; //abc sorrend
                break;
            case '1':
                $rendez = '4 DESC'; //ár csökkenő
                break;
            case '2':
                $rendez = '4 ASC'; //ár növekvő
                break;
        }
    }
}
```

1.11.1.1 Rendezés

Az alapértelmezetten ABC sorrendben jelennek meg a termékek, de a felhasználó egy legördülő listából tud választani további kettő rendezési mód közül: ár csökkenő, ár növekvő. Annak érdekében, hogy megmaradjon a rendezés lapozás vagy oldalfrissítés esetén is, el kell menteni a választott módot egy SESSION változóba. Ezután a megfelelő rész

a '\$rendez' változóban kerül elmentésre, amely pedig az SQL utasításokban lesz felhasználva.

1.11.1.2 Lapozás

```
public function lapozas($p, $egyoldal, $darabszam)
{
    $start = ($p - 1) * $egyoldal;
    $lapok = ceil($darabszam / $egyoldal);
    $url = strtok($_SERVER['REQUEST_URI'], '&');

    echo "
    <div class='egysor'>
    <ul class='pagination'>
    <li class='page-item'><a href='{ $url }&p=1' class='page-link'>&laquo;</a></li>";

    $darab = floor($darabszam / $egyoldal);
    $kezdolap = max(1, min($p - floor($darab / 2), $lapok - $darab + 1));

    for ($i = $kezdolap; $i < $kezdolap + $darab && $i <= $lapok; $i++) {
        echo "<li class='page-item'><a href='{ $url }&p=$i' class='page-link'>$i</a></li>";
    }
    echo "<li class='page-item'><a href='{ $url }&p=$lapok' class='page-link'>&raquo;</a></li>
    </ul>
    </div>";
    return array($egyoldal, $start);
}
```

A lapozás lehetséges az első, utolsó oldalra, valamint az oldalszámokra kattintva is. Ki van számolva a kezdőérték, ami mindig változik a lapozás által és az oldalak száma. Ki van nyerve az URL olyan formában, hogy utána lehessen illeszteni a lapozás oldalszámát. Ha oldalszámra kattintunk, akkor kiszámolja hány oldalra lesz szükség és megjeleníti ezeket. Visszatérési értéke egy tömb, amely tartalmazza, hogy egy oldalon mennyinek kell megjeleníteni és a kezdőértéket, amelyek az SQL utasítás LIMIT záradékában lesznek felhasználva.

1.11.1.3 Összes termék lekérdezése

```
if ($kategoria_id == 0) { //összes termék
    $sql = "SELECT Count('id') AS id FROM termekek";
    $sv = parent::sqlSelect($sql);
    $darabszam = $sv['id'];

    $mennyi = 12;
    $lapozas = $this->lapozas($p, $mennyi, $darabszam);
    $egyoldal = (int) $lapozas[0];
    $start = (int) $lapozas[1];

    $sql = "SELECT t.id, m.megnevezes, t.megn, t.ar
    FROM termekek AS t
    INNER JOIN markak AS m ON m.id = t.marka_id
    ORDER BY $rendez
    LIMIT $start, $egyoldal";
}
```

Amikor a navigációs sávon kiválasztjuk az 'Összes termék' menüpontot. Mint a neve is mutatja az összes adatbázisban tárolt termék megjelenik, ekkor átadásra kerül a kategória ID, ami nulla lesz. Jelen esetben 12 darab termék lesz látható egy oldalon, eszerint a beállítás szerint.

1.11.1.4 Kiválasztott kategória termékei

```
} elseif ($kategoria_id > 0) { //kiválasztott kategória termékei
    $sql = "SELECT Count('id') AS id FROM termekek WHERE kategoria_id = $kategoria_id";
    $sv = parent::sqlSelect($sql);
    $darabszam = $sv['id'];

    $mennyi = 4;
    $lapozas = $this->lapozas($p, $mennyi, $darabszam);
    $egyoldal = (int) $lapozas[0];
    $start = (int) $lapozas[1];

    $sql = "SELECT t.id, m.megnevezes, t.megn, t.ar
            FROM termekek AS t
            INNER JOIN markak AS m ON m.id = t.marka_id
            WHERE t.kategoria_id = $kategoria_id
            ORDER BY $rendez
            LIMIT $start, $egyoldal";
```

Amennyiben a metódus megkapott kategória azonosítója nagyobb, mint nulla, akkor az adott kategória termékei lesznek lekérdezve.

1.11.1.5 Keresett kulcsszó termékei

```
} elseif (isset($keresett)) { //keresett kulcsszó termékei
    $sql = "SELECT Count('t.id') AS id
            FROM termekek AS t
            INNER JOIN markak AS m ON m.id = t.marka_id
            WHERE 1 AND CONCAT_WS(' ', m.megnevezes, t.megn) LIKE '%$keresett%'";
    $sv = parent::sqlSelect($sql);
    $darabszam = $sv['id'];

    $mennyi = 4;
    $lapozas = $this->lapozas($p, $mennyi, $darabszam);
    $egyoldal = (int) $lapozas[0];
    $start = (int) $lapozas[1];

    $sql = "SELECT t.id, m.megnevezes, t.megn, t.ar
            FROM termekek AS t
            INNER JOIN markak AS m ON m.id = t.marka_id
            WHERE 1 AND CONCAT_WS(' ', m.megnevezes, t.megn) LIKE '%$keresett%'
            ORDER BY $rendez
            LIMIT $start, $egyoldal";
```

Ha a navigációs sáv keresősávját használjuk, akkor a keresett termékek lekérdezésére az itt látható módon kerül sor.

A termékek a megfelelő SQL utasítás kiválasztása után végezetül egy asszociatív tömbbe kerülnek, a saját metódusunk által, amely így használható:

```
$termek = parent::sqlAssoc($sql);
```

1.11.2 Megjelenítés

```
echo "<div class='row'>";
foreach ($termek as $termek) {
    echo "
        <div class='col-sm-6 col-lg-4 col-xl-3 my-3'>
            <div class='card mb-4 h-100 shadow-sm border border-info'>;
                $this->image($termek['id'], 'termek');
            </div>
            <div class='card-body'>
                <div class='clearfix mb-3'>
                    <span class='markanok float-start border border-info'>{$termek['megnevezes']}</span>
                </div>
                <h3 class='card-title'>{$termek['megn']}</h3>
                <p class='float-start ar'>" . number_format($termek['ar'], 0, ',', ' ') . " Ft</p>
                <div class='sor'>
                    <button class='kosarGomb btn' data-termek-id='{$termek['id']}'><img class='kosar' src='kepek/shopping-cart.png'></button>
                    <button class='kedvencekGomb btn' data-termek-id='{$termek['id']}'><img class='sziv' src='kepek/heart.png'></button>
                    <div class='vasarlas_gomb'>
                        <a href='termek.php?id={$termek['id']}' class='btn btn-info vasarlas_gomb'>Tovább</a>
                    </div>
                </div>
            </div>
        </div>
    ";
}
echo "</div>";
```

A termékek kártyán jelennek meg képpel, megnevezéssel és árral. Kosárba és kedvencek közé innen is helyezhetők. Minden termékről megjeleníthető egy részletes, külön oldal, amelyhez a 'Tovább' gombbal navigálhatunk el.

1.12 Kosár

```
if (empty($stermekek)) {
    echo "<h1>A kosár üres!</h1>";
} else {
    echo "
    <table class='table table-hover table-striped tablazat mx-auto'>
    <tr class='text-center'>
        <th class='border' colspan=2>Termék</th>
        <th class='border'>Ár</th>
        <th class='border'>Mennyiség</th>
    </tr>";
    foreach ($stermekek as $stermek) {
        $sar = $stermek['ar'] * $stermek['mennyiseg'];
        echo "
        <tr>
            <td>";
            $this->image($stermek['termek_id'], 'termek');
            echo "</td>
            <td class='text-align-right fs-5'>
                <a href='termek.php?id={$stermek['termek_id']}'>{$stermek['nev']}</a>
            </td>
            <td class='text-align-right'>
                . number_format($sar, 0, ',', ' ') . " Ft</td>
            <td class='text-align-right'>
                <div class='mennyisegnek mx-auto'>
                    <span class='menny border border-primary rounded'>{$stermek['mennyiseg']} db</span>
                    <button class='btn btn-primary csokk_menny_gomb' value='{$stermek['id']}'>-</button>
                    <button class='btn btn-primary nov_menny_gomb' value='{$stermek['id']}'>+</button>
                    <button class='btn btn-outline-danger torol' value='{$stermek['id']}'>Törlés</button>
                </div>
            </td>
        </tr>";
    }
    echo "
    </table>
    <div class='vegosszeg w-75 mx-auto py-2'>
        <h2>Végösszeg: " . number_format($vegosszeg, 0, ',', ' ') . " Ft</h2>
        <button class='btn btn-danger teljes_kosar_torol' >Teljes kosár törlése</button>
        <a class='btn btn-info float-end' href='rendeles.php'>Tovább a rendeléshez</a>
    </div>
    ";
}
```

A kosár elemei adatbázisban vannak elmentve, ezek vannak megjelenítve egy táblázatban. Abban az esetben, ha a bejelentkezett felhasználó nem helyezett semmit a kosárba, 'A kosár üres!' felirat jelenik meg. Megjeleníti a táblázatban a termék képét, megnevezését, árát, mennyiségét. A táblázat alatt megtalálható a végösszeg, egy 'Teljes kosár törlése' és egy 'Tovább a rendeléshez' gomb. A termék ára és a végösszeg dinamikusan változnak a darabszám módosításának függvényében. Van lehetőség a mennyiség növelésére, csökkentésére 1-99 között, egy termék törlésére és a teljes kosár törlésére, ezen funkciók egy-egy gomb segítségével érhetők el.

```

$(document).ready(function() {
    $('#kosarGomb').click(function() {
        var termekId = $(this).data('termek-id');
        var termekMenny;

        if ($('#menny').val() < 1 || typeof($('#menny').val()) === "undefined") {
            termekMenny = 1;
        } else if (typeof($('#menny').val()) !== "undefined") {
            if ($('#menny').val() > 99) {
                termekMenny = 99;
            } else {
                termekMenny = $('#menny').val();
            }
        }

        $.ajax({
            type: 'POST',
            url: 'kosar_feldolgoz.php',
            data: { termek_id: termekId, termek_menny: termekMenny },
            success: function(response) {
                var valasz = JSON.parse(response);
                if (!valasz.success) {
                    alert(valasz.message);
                }
            },
            error: function(error) {
                alert('Hiba történt: ' + error);
            }
        });
    });
});

```

A kosárba helyezés a termék ID-je alapján történik, mennyiséggel együtt. A termék mennyisége a 'menny' azonosítójú beviteli mezőtől függ. Amennyiben a beírt érték nulla vagy negatív szám, esetleg nincs megadva érték, akkor mindenképpen 1 lesz. Akkor is ez lesz érvényes, ha csak egy kosárba helyez gomb található, nincsen beviteli mező a darabszámnak, például kedvencek közül kosárba helyezés. Ha érvényes a szám akkor ellenőrizzük, hogy ne legyen 99-től nagyobb az érték, végül, ha megfelelő a beírt darabszám, akkor lesz elmentve. Egy AJAX hívás segítségével átadásra kerül a termék azonosítója és a mennyiség is, ezután következik az adatbázisba mentés. Amennyiben sikeres vagy sikertelen, megjelenik 'alert()' formájában egy üzenet.

1.13 Kedvencek

A kedvencek közé helyezett termékek szintén adatbázisban vannak elmentve és táblázatban megjelenítve, hasonlóan a kosárhoz, csak egyszerűbben. Nincsen mennyiség eltárolva, ebből adódóan nem lehet módosítani sem a darabszámot. A kedvencek között lévő termékeket el lehet távolítani és kosárba helyezni, ebben az esetben egy darab kerül át a kosárba, viszont továbbra is megmarad a kedvencek között.

1.14 Belépés

```
public function bejelentkezesForm()
{
    echo "
    <div class='container_form'>
        <form class='formnak' method='POST' action='{$_SERVER['PHP_SELF']}'>
            <div class='head'>
                <span>Bejelentkezés</span>
            </div>
            <div class='inputs'>
                <input type='text' name='fnev' placeholder='Felhasználónév' required>
                <input type='password' name='jelsz' placeholder='Jelszó' required>
            </div>
            <input type='submit' class='button' value='Bejelentkezés'>
        </form>
        <div class='form-footer'>
            <p>Regisztráljon itt! <a href='reg.php'>Regisztráció</a></p>
        </div>
    </div>
    ";
}
public function bejelentkezes()
{
    if (isset($_POST['fnev']) && isset($_POST['jelsz'])) {
        $fnev = $_POST['fnev'];
        $j = $_POST['jelsz'];
        $_SESSION['o']=0;
        $jelszo = sha1($j);

        $sql = "SELECT * FROM felhasznalok WHERE nev = '$fnev' AND jelszo = '$jelszo'";
        $ell = parent::sqlAssoc($sql);

        if (count($ell) > 0) {
            header("Location: index.php");
        } else {
            echo "<script>alert('Nem megfelelő a felhasználónév vagy jelszó')</script>";
        }
        foreach ($ell as $e) {
            $_SESSION['fid'] = $e['id'];
            $_SESSION['fnev'] = $fnev;
        }
    }
}
```

Ez a kódrészlet a bejelentkezés oldal fő részét hozza létre. A bejelentkezesForm() eljárás létrehozza a bejelentkezési adatok beírására alkalmas űrlapot, és a bejelentkezés gombot. A bejelentkezes() metódus magát a bejelentkezési folyamatot hajtja végre. Először is ellenőrzi, hogy a felhasználó megadta-e a felhasználó nevét, és jelszavát. Ha ez a feltétel teljesült, akkor fut tovább a kód. A megadott felhasználónevet, és jelszót elmenti

egy-egy változóba, ezután egy sql parancs futtatásával megnézi, hogy az adatbázis tartalmaz-e ilyen néven regisztrált, és ezzel a jelszóval rendelkező felhasználót. Ha tartalmaz, akkor átirányít az 'index.php' oldalra, és a felhasználónevet elmenti munkamenet változóként.

1.15 Regisztráció

```
public function regisztracioForm()
{
    echo "
    <div class='container_form'>
        <form class='formnak' method='POST' action='{$_SERVER['PHP_SELF']}'>
            <div class='head'>
                <span>Regisztráció</span>
            </div>
            <div class='inputs'>
                <input type='text' name='fnev' placeholder='Felhasználónév' required>
                <input type='email' name='email' placeholder='Email' required>
                <input type='password' name='jelsz' placeholder='Jelszó' required>
            </div>
            <input type='submit' class='button' value='Regisztráció'>
        </form>
        <div class='form-footer'>
            <p>Regisztrált már? <a href='login.php'>Bejelentkezés</a></p>
        </div>
    </div>
    ";
}

public function regisztracio()
{
    if (isset($_POST['fnev']) && isset($_POST['email']) && isset($_POST['jelsz'])) {
        $fnev = $_POST['fnev'];
        $email = $_POST['email'];
        $j = $_POST['jelsz'];

        $jelszo = sha1($j);
        $sql = "INSERT INTO felhasznalok (nev,email,jelszo) VALUES ('$fnev', '$email', '$jelszo')";

        try {
            parent::sqlQuery($sql);
            echo "<script>alert('Regisztráció sikeres!')</script>";
        } catch (Exception $e) {
            echo "<script>alert('Hiba: '+$e)</script>";
        }
    }
}
```

Ez a kódrészlet a regisztráció oldal fő részét hozza létre. A regisztracioForm() eljárás létrehozza a regisztráláshoz szükséges adatok beírására alkalmas űrlapot, és a regisztráció gombot. A regisztracio() metódus ellenőrzi, hogy a felhasználó megadott-e minden adatot, ha ez a feltétel teljesült, akkor fut tovább. A megadott adatokat átadja egy-egy változónak, a jelszót titkosítja sha1 hash titkosításban. Ezután lefuttat egy paraméteres sql parancsot, ami siker esetén felugró ablakban megjeleníti 'Regisztráció sikeres', hiba esetén pedig ugyanilyen felugró ablakban megjeleníti a hibát.

1.16 Konfiguráció összerakó

A konfigurációépítő oldalon a felhasználók összeállíthatják álmaik számítógépét, vagy azt a mit a pénztárcájuk megenged. Ezután menthetik az adatbázisba, és a kosárhoz is hozzáadhatják.

```
<form action="bekuld.php" method="post">
  <table>
  <?php
    $sql = 'SELECT * FROM termek_kategoriak WHERE kat_id=1';

    $stmtKat = $webshop->sqlQuery($sql);
    $kategoriak = $stmtKat->fetchAll(PDO::FETCH_ASSOC);
    $kat_db = 0;
    foreach ($kategoriak as $kategoria) {
      $kat_db++;
    }

    $sql = "SELECT t.id, t.kategoria_id, CONCAT(m.megnevezes, ' - ', t.megn) AS nev
            FROM termek AS t
            INNER JOIN markak AS m ON m.id=t.marka_id";
    $term = $webshop->sqlQuery($sql);
    $termek = $term->fetchAll(PDO::FETCH_ASSOC);

    $szam = 0;
    foreach ($kategoriak as $kategoria) {
      if ($kategoria['kat_id'] == 1) {
        echo "<tr>
                <td>
                    {$kategoria['megnevezes']} :
                </td>
                <td>
                    <select name='$szam' id='{ $kategoria['megnevezes']} '>
                </td>
            ";
        $szam++;
        foreach ($termek as $termek) {
          if ($termek['kategoria_id'] == $kategoria['id']) {
            echo "<option value='{ $termek['id']} '>{$termek['nev']}</option>";
          }
        }
      }
    }
    echo "</td>";
  ?>
</table>
  <input type="submit" value="Beküldés" name="sub">
</form>
```

A fenti kódrészlet létrehoz egy űrlapfelületet, ebben egy kettő oszlopos táblázattal. A táblázat bal oldali oszlopában a kategóriák jelennek meg, jobb oldalt pedig a termékek egy lenyíló select input mezőben. A megjelenítéshez kettő darab foreach ciklust használtam. A külső foreach ciklus a táblázat adott sorának kettő oszlopáért felelős, a belső

foreach ciklus pedig a select input mezőkhöz való opciók megadásáról gondoskodik. Az űrlap beküldésekor lefut a 'bekuld.php' fájl. Ez felelős az adatbázisban való rögzítésért.

A beküldés gombra nyomás után a 'Post' tömb 'sub' mezője kap egy értéket. A feltételvizsgálat azt vizsgálja, hogy a felhasználó megnyomta-e a gombot. Az első sql parancs hozzáad egy új adatsort a 'konfig_id' táblához. A for ciklusban lévő többször lefutó sql kód a 'konfig' táblához ad hozzá adatot. Minden lefutáskor rögzíti a konfiguráció azonosítóját, és a hozzáadott alkatrész azonosítóját is.

```
<?php
require_once("fuggvenyek.php");
session_start();

if ($_SERVER["REQUEST_METHOD"]=="POST") {
    if (isset($_POST["sub"])) {
        $felh_id=$_SESSION["fid"];
        $webshop = new Webshop();

        $sql = "INSERT INTO konfig_id (felh_id) VALUES('$felh_id')";

        $webshop->sqlQuery($sql);
        $sql = "SELECT LAST_INSERT_ID() AS id ";
        $sv=$webshop->sqlSelect($sql);
        $konfig_id=$sv["id"];

        for ($i = 0; $i < count($_POST)-1; $i++) {
            $sql = "INSERT INTO konfig (termek_id, konf_id) VALUES({$_POST[$i]}, $konfig_id)";

            $webshop->sqlQuery($sql);
        }

        $_SESSION["info"]="Sikeresen hozzáadva.";
        header("location:konfig_ossze.php");
    }
}
```

1.17A konfiguráció megtekintő, és értékelő oldal

1.17.1 Card elemek

Ezt az oldalt csak a bejelentkezett felhasználók tudják elérni egy menüpontról. Itt saját, és más felhasználók által összerakott konfigurációkat lehet megtekinteni, és értékelni. A központi div tárolóban oldalanként négy darab konfiguráció jelenik meg. Minden konfiguráció egy külön Bootstrap-es 'card' elemként jelenik meg, a like és dislike gombbal együtt.

```

for ($j = 0; $j < count($vtomb); $j++) {
    echo "<tr class='border'><td class='border'>";
    $sql = "SELECT megnevezes
            FROM termekek AS t
            INNER JOIN termek_kategoriak AS k
            ON t.kategoria_id=k.id
            WHERE t.id={$vtomb[$j]['termek_id']}";
    $kat = $webshop->sqlAssoc($sql);

    echo $kat[0]['megnevezes'] . "</td><td class='border'>";
    $sql = "SELECT CONCAT(m.megnevezes, ' - ', t.megn) AS nev
            FROM termekek AS t
            INNER JOIN markak AS m
            ON m.id=t.marka_id
            WHERE t.id={$vtomb[$j]['termek_id']}";
    $tomb = $webshop->sqlAssoc($sql);
    echo $tomb[0]['nev'] . "</td></tr>";
}

```

A fenti kódsor adja hozzá a kártyákhoz a konfiguráció alkatrészeit táblázatos formában. Hasonlóan, mint a konfiguráció összerakó oldalon, de a legnagyobb különbség az, hogy itt nincsenek 'select' mezők.

```

$sql1="SELECT COUNT(konfig_id) AS likes FROM erkeles WHERE erkeles=1 AND konfig_id=$i";
$sql2="SELECT COUNT(konfig_id) AS dislikes FROM erkeles WHERE erkeles=2 AND konfig_id=$i";
$tomb=$webshop->sqlQuery($sql1);
$like=$tomb->fetchAll(PDO::FETCH_ASSOC);
$tomb=$webshop->sqlQuery($sql2);
$dislike=$tomb->fetchAll(PDO::FETCH_ASSOC);

$fid=$_SESSION['fid'];
$sql="SELECT erkeles FROM erkeles WHERE konfig_id=$i AND felh_id=$fid";
$tomb=$webshop->sqlQuery($sql);
$select=$tomb->fetchAll(PDO::FETCH_ASSOC);

```

Ez a kódrészlet felelős azért, hogy lekérje az adott konfigurációhoz a like-ok és dislike-ok számát, illetve meghatározni, hogy a jelenleg bejelentkezett felhasználó értékelte-e a konfigurációt, és ha igen, mivel értéke.

1.17.2 Ajax hívások

A like - dislike funkció működését AJAX hívásokkal oldottuk meg. Egy adott konfigurációhoz tartozó gombok kapnak egy külön 'like' és 'dislike' osztályt, innen tudja a program, hogy melyik AJAX hívást kell végrehajtani.

```
$(".like").click(function () {  
    var id = this.id;  
    $.ajax({  
        url: "rating.php",  
        type: "POST",  
        data: { felh_id: fid, konf_id: id[0], ert: 1 },  
        success: function (response) {  
            location.reload();  
        }  
    });  
});  
  
$(".dislike").click(function () {  
    var id = this.id;  
    $.ajax({  
        url: "rating.php",  
        type: "POST",  
        data: { felh_id: fid, konf_id: id[0], ert: 2 },  
        success: function (response) {  
            location.reload();  
        }  
    });  
});
```

A 'fid' változó a bejelentkezett felhasználó azonosítója az adatbázisban, az 'id[0]' a megnyomott gomb azonosítójának első karaktere, ezt adja tovább, mint a konfiguráció azonosítója. A mentett értékelés értéke kettő lehet. Ha a felhasználó like-olt, akkor '1', ha dislike-olt, akkor '2'.

1.17.3 rating.php

Mindkettő AJAX hívás ugyanarra a PHP fájlra küldi tovább az adatokat, és ebből a fájlból indulnak az adatbázisműveletek is. A felhasználó módosíthatja az értékelését, vagy törölheti, ha ugyanarra a gombra nyom, amit már választott.

```
$felh_id = $_POST['felh_id'];
$konf_id = $_POST['konf_id'];
$ert = $_POST['ert'];

$sql = "SELECT COUNT(id) AS ratings
        FROM erkekeles
        WHERE konfig_id=$konf_id AND felh_id=$felh_id";
$r = $webshop->sqlAssoc($sql);

$sql = "SELECT erkekeles AS ert
        FROM erkekeles
        WHERE konfig_id=$konf_id AND felh_id=$felh_id";
$adott = $webshop->sqlAssoc($sql);

if ($r[0]['ratings'] < 1) {
    $sql = "INSERT INTO erkekeles(konfig_id, felh_id, erkekeles) VALUES('$konf_id','$felh_id','$ert')";
    $webshop->sqlQuery($sql);
}

if ($r[0]['ratings'] > 0) {
    if ($adott[0]['ert'] != $ert) {
        $sql = "UPDATE erkekeles SET erkekeles=$ert WHERE konfig_id=$konf_id AND felh_id=$felh_id";
        $webshop->sqlQuery($sql);
    } else {
        $sql = "DELETE FROM erkekeles WHERE konfig_id=$konf_id AND felh_id=$felh_id";
        $webshop->sqlQuery($sql);
    }
}
```

A kód kezdetben megnyitja, a munkamenetet, és külön változóba teszi a '\$_POST' tömb elemeit, ezután kettő adatbázis-lekérdezést hajt végre. Az első lekérdezés lekérdezi, hogy az adott felhasználó értékelte-e már az adott konfigurációt, a második lekérdezés az értékelt konfiguráció értékelését kérdezi le. Ezután feltételvizsgálattal eldönti az adott felhasználó értékelésének számát az adott konfiguráción. 0 érték esetén egyszerűen hozzáadja az adatbázishoz a megfelelő adatokat. Ha a felhasználó már értékelte az adott konfigurációt, akkor frissíti az adatbázis adatsorát, ahol a konfiguráció és a felhasználó azonosítója azonos a megadottakkal. Ellenkező esetben törli az adatbázis azon adatsorát, ahol az előző feltételek teljesülnek.

1.17.4 limit.php

A nagyon sok konfiguráció betöltésénél való problémákat, a megjelenített konfigurációk limitálásával oldottuk meg. Mindig csak négy darab konfigurációt jelenít meg az oldalon. Bejelentkezéskor létrejön egy 'o' változó a munkamenetben. Ez adja meg az oldalszámot, ez változik minden lapozásnál, vagyis, ha a felhasználó átmegy az oldal más részére, és visszatér, ugyanott folytatja a konfigurációk böngészését.

```
<?php
session_start();
require_once ("fuggvenyek.php");
$webshop = new Webshop();

$sql = "SELECT id FROM konfig_id";
$szam_Tomb = $webshop->sqlAssoc($sql);

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    if (isset($_POST["k"])) {
        if (count($szam_Tomb) > 4 * ($_SESSION['o'] + 1)) {
            $_SESSION['o']++;
            header("location:konfig.php");
            $_SESSION['osszes_konfig'] = count($szam_Tomb);
        } else {
            header("location:konfig.php");
            $_SESSION['osszes_konfig'] = count($szam_Tomb);
        }
    }
    if (isset($_POST["e"])) {
        if ($_SESSION['o'] > 0) {
            $_SESSION['o']--;
            header("location:konfig.php");
        } else {
            header("location:konfig.php");
        }
    }
}
}
```

Az oldalváltó gombok egy 'e' és 'k' azonosítót kaptak, ezeket adják tovább a 'limit.php' fájlban.

1.18 Tesztelés, tesztesetek

A munkavégzés közben többször teszteltük az elkészített kódokat. Az alábbi tesztelési módszereket, technikákat végeztük el weboldalunk kódjainak tesztelésére.

1.18.1 Unit tesztek:

A unit tesztek arra összpontosítanak, hogy az egyes komponensek vagy egységek (modulok, osztályok, objektumok) jól működnek-e elkülönülve. A unit tesztek segítenek ellenőrizni az egyes funkciók helyes működését és stabilitását, könnyebb hibákat találni az egyes komponensekben.

1.18.2 Integrációs tesztek:

Az integrációs tesztek azt vizsgálják, hogy a különálló egységek együttesen tudnak-e helyesen működni. Az integrációs tesztek segítenek abban, hogy az egyes komponensek közötti interakciók megfelelően működjenek, és elkerüljék az esetleges összeütközéseket vagy hibákat.

1.18.3 Funkcionalitási tesztek:

A weboldalunkon létrehozott funkciókat is teszteltük, annak érdekében, hogy minden hibát kiszűrjünk, úgy működnek-e, ahogyan elterveztük.

- **Kategóriák és termékek:** Ellenőrizni kell, hogy a termékek jól vannak-e csoportosítva kategóriákba, illetve mindig a megfelelő termékek jelennek-e meg.
- **Keresés, rendezés, lapozás:** Termékek között lehetősége van a felhasználónak keresni, a termékeket rendezni, például: ár szerint növekvő, csökkenő rendezés. Lapozás, annak érdekében, hogy ne legyen túlterhelve a weboldal, amiért az összes termék meg lenne jelenítve, ezért csak néhány található meg egy oldalon. Ezeknek a funkcióknak a tesztelését is elvégeztük.
- **Kosár, kedvencek és megrendelés:** A tesztelések alkalmával meg kell győződni róla, hogy a vásárlók sikeresen hozzáadhatnak-e termékeket a kosárhoz, kedvencekhez, megtekinthetik-e a tartalmát ezeknek, és sikeresen leadhatnak-e megrendeléseket.

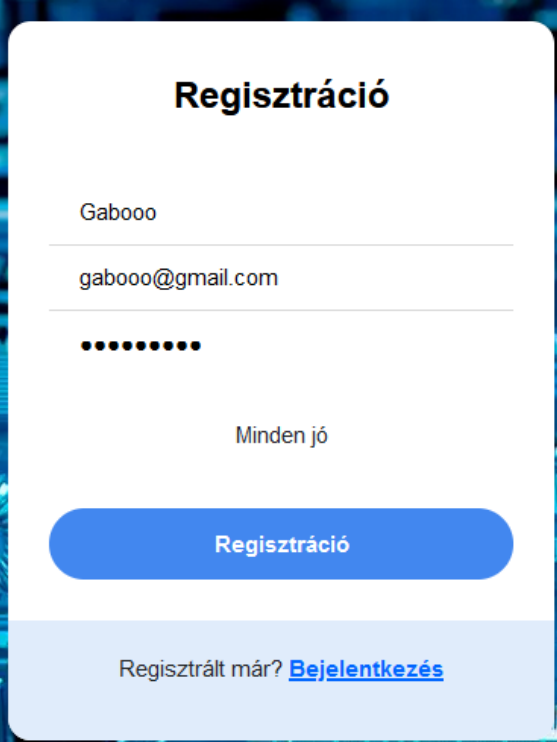
1.18.4 Megjelenés, felhasználói élmény tesztelése:

A megjelenésre is próbáltunk odafigyelni, egy modern, letisztult, felhasználóbarát kinézetet szerettünk volna létrehozni. Nem szerettük volna, hogy legyen olyan, ami a felhasználói élményt leronthatná.

- **Reszponzivitás:** Weboldalunkat minden eszközön és különböző méretű képernyőkön is leteszteltük. Azt néztük, hogy úgy jelenik-e meg az oldal, hogy semmi ne legyen takarásban, elcsúszva, nem esztétikusan megjelenítve vagy a funkciók működése hibás lenne-e.
- **Használhatóság:** Arra törekedtünk, hogy bizonyos funkciók több helyről is elérhetőek legyenek. A kategóriák megjelennek a navigációs sávon és a főoldalon is, ezzel is segítve a vásárlót, hogy a megfelelő kategória, termék megtalálása gyorsabb legyen, több helyről is elérhető legyen. A kereső használatával is segíteni próbáljuk a vásárlást. Ha megtalálta a terméket, akkor kosárba és kedvencek közé is helyezheti anélkül, hogy a termék részletes oldalát megnyitná, ahol természetesen szintén megtalálhatóak ezek a funkciók. Ezeket is mind le kellett ellenőrizni, hogy úgy működjenek, ahogyan szerettük volna.

1.18.5 Regisztráció, bejelentkezés

Nagy hangsúlyt fektettünk a regisztráció szolgáltatás, és az adatbáziskezelő asztali alkalmazás bejelentkezési felületének tesztelésére. Az oldalra való regisztráció oldalán a jelszónak minimum 8 karakter hosszúnak kell lennie, és tartalmaznia kell legalább egy kisbetűt, egy nagybetűt, egy számot és egy különleges karaktert. Az oldal kiírja, a jelszó jelenlegi hosszát. Tesztelésként 'Gabooo' néven fogok regisztrálni, a 'gabooo@gmail.com' email címmel, és 'Gabooo_20' jelszóval.



The screenshot shows a registration form with the title "Regisztráció". It contains input fields for the username "Gabooo", the email address "gabooo@gmail.com", and a password field represented by dots. Below the password field, it says "Minden jó". At the bottom of the form is a blue button labeled "Regisztráció". Below the button, there is a link that says "Regisztrált már? [Bejelentkezés](#)".

A 'Minden jó' felirat a jelszóra utal, ezt azt jelenti, hogy minden feltétel teljesült, és minden szükséges dolgot tartalmaz a jelszó. A regisztráció gombra nyomva, egy JavaScript 'alert()' ablakot fog megjeleníteni, ebben a regisztráció sikerességéről szóló üzenettel.



Abban az esetben, amikor a jelszó nem felel meg a megadott követelményeknek, egy másik tesztesetre is szükség van, mert meg kell vizsgálni a teljesség kedvéért. Felhasználónévként megadott adat a 'Dabdab' lesz, 'dabdab@gmail.com' email címmel, és 'dabdab11' jelszóval. Ebben az esetben az oldal nem változik, de a 'Regisztráció' feliratú gomb nem lesz kattintható.

A screenshot of a web registration form titled 'Regisztráció'. The form has three input fields: the first contains 'Dabdab', the second contains 'dabdab@gmail.com', and the third is masked with dots. Below the password field, there is a feedback message: 'A jelszó megfelelő hosszúságú'. At the bottom of the form is a large blue button with a white mouse cursor icon and the text 'Regisztráció'. Below the form, there is a light blue footer area with the text 'Regisztrált már?' followed by a blue link 'Bejelentkezés'. The entire form is set against a background of blue circuitry.

1.18.6 Rendelés

A kosárba helyezett termékeket meg szeretnénk rendelni, akkor a személyes adatok, szállítási cím megadása után, abban az esetben, ha a rendelés sikeres, akkor egy pipa kell, hogy fogadja a vásárlót és az adatai a jobb oldalon meg kell, hogy jelenjenek.



Sikeresen megrendelve!

Megrendelő neve
Kiss István
Elérhetőségei
istvan1234@gmail.com
0670883322
Szállítási cím
2687 Bercel, Bástya út 8.
Végösszeg
1 214 175 Ft
<small>A főoldalon a 'Rendelési előzmények' menüpontra kattintva megtekinthetők a rendelés tételei.</small>

Viszont, ha valami probléma adódik a rendelés közben, akkor ez az oldal lesz látható. Ekkor csak a főoldalra van lehetőség visszamenni.



Sikertelen megrendelés!

1.19 Továbbfejlesztési lehetőségek

Sok funkciót szerettünk volna még implementálni a weboldalunkba, csak idő hiányában és egyéb okokból kifolyólag nem sikerült véghez vinnünk, ezért a továbbfejlesztések közé kerültek.

1.19.1 Felhasználói fiókok módosítása

Szerettünk volna egy menüpontot, ahol egy űrlapon a felhasználó tudná módosítani a személyes adatai, szállítási címet, elérhetőségeit. Profilkép feltöltésére is szeretnénk biztosítani lehetőséget, amely megjelenne a navigációs sávon és számítógép konfiguráció megosztása esetén a többi konfiguráció között is, a feltöltő neve mellett.

1.19.2 E-mail

E-mailen keresztül értesítenénk a felhasználót például: regisztrálás esetén egy visszaigazolás lenne szükséges, adatok módosítása, rendelés, visszaküldés esetén egy visszajelzést kapna.

1.19.3 Ügyfélszolgálat

A legtöbb webáruház rendelkezik saját ügyfélszolgálattal, ahol lehet segítséget kérni, észrevételt, problémát, panaszt jelezni, érdeklődni a megrendelt termékek iránt, visszaküldést lebonyolítani. Ilyen és hasonló műveletek elvégzésére nyújtana megoldást egy webes felület létrehozása.

1.19.4 Rendelés visszaküldése

Amennyiben a vásárolt eszköz hibás, megsérült szállítás közben, vagy egyéb okból adódóan nem működik megfelelően. Biztosítanánk lehetőséget a termék visszaküldésére ügyfélszolgálaton keresztül, esetleg egy megfelelő menüpont kiválasztásával. Visszaküldés esetén a felhasználó visszakapná a termék összegét vagy egy kupont, amely a következő rendelésnél jelentene kedvezményt.

1.19.5 Akciók, leértékelések, kuponkód

Az adatbázis fel lett készítve az akciók létrehozására, kezelésére, csak idő hiányában nem került kidolgozásra. Meg lehetne adni százalékkal a kedvezmény mértékét, egy időintervallumot, ami között lenne érvényes vagy pedig egy aktív mezővel is lehetne inaktíválni.

Kuponkód járhatna az első rendelés után, visszaküldött termék után, szezonálisan. Több fajta is lenne, bizonyos mértékű kedvezmény a végösszezből, ingyenes szállítás biztosítása, esetleg több hasonló termék vásárlása esetén alacsonyabb áron megszerezhetővé válnának.

1.19.6 Fizetési módok

Szeretnénk a későbbiekben bővíteni fizetési lehetőségeinket, bankkártyát szeretnénk elfogadni, esetleg API felhasználásával újabb módszereket biztosítani, például: PayPal, SimplePay.

1.19.7 Kompatibilitás és további elemek hozzáadása

Konfiguráció összeállítás esetén kompatibilitás ellenőrzésére is lenne lehetőség, adatbázisba mentenénk a szükséges paramétereket, illetve hozzáadhatók lennének nem kötelező elemek, például: perifériák, kiegészítők, kötelezők közül több mint egy elem is.

1.19.8 Konfiguráció megosztó oldal továbbfejlesztése

Hasznos lenne egy keresés funkció egy felhasználóra vagy a keresett terméket tartalmazó konfigurációk szűkítéséhez, szűrési lehetőség alapján dátumnak, márkának, kompatibilitáshoz szükséges adatoknak megfelelően jelennének meg az oldalon, rendezés történne dátum és népszerűség szerint (like, dislike aránya, megtekintések száma alapján).

2 Felhasználói dokumentáció

2.1 Bevezetés

Alkalmazásunk elindításához, és beüzemeléséhez, a már említett, és bemutatott telepítési folyamatot kell végrehajtani. A telepítéshez lapozzon vissza a 'Megfelelő indítás' bekezdésre. Itt részletesen, és érthetően megtalálható a telepítés folyamata.

2.2 Rendszerkövetelmények

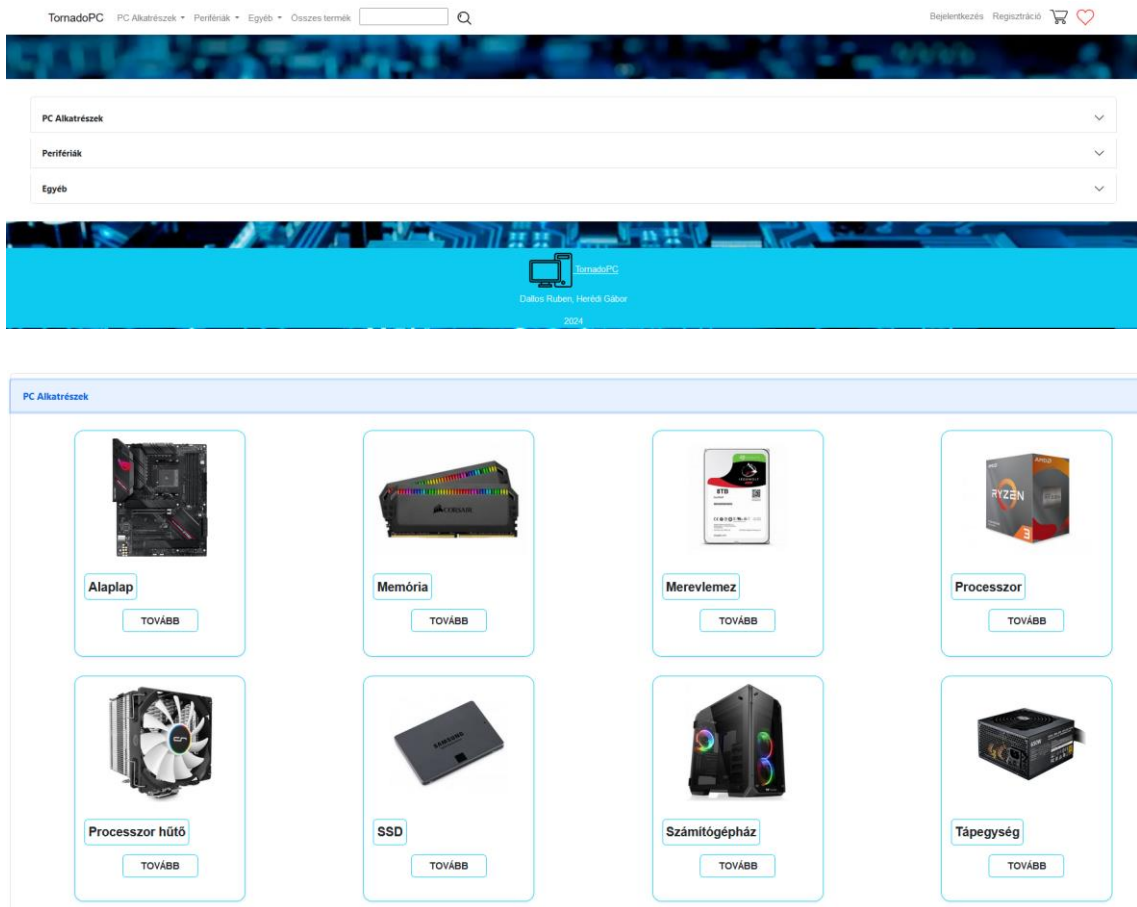
Mesterremekünk webshop lévén, szükség van egy mai modern böngészőre, lehet ez Google Chrome, Mozilla Firefox, Opera és Opera GX is. Szükséges lehet néhány bővítmény telepítése a technológiák miatt, és a JavaScript futtatását is engedélyezni kell. A webes alkalmazás egy adatbázis-kezelő rendszert is használ, amely telepítése szükséges a helyi vagy webszerverre, amiről az alkalmazást szeretnénk futtatni. Futtatáshoz szükséges vezetékes, vagy vezeték nélküli internetkapcsolat. Alkalmazásunkat helyi szerveren fogjuk bemutatni, ezért szükségünk van egy webalkalmazás kiszolgáló helyi alkalmazásra, ami az Apache http szerver, a MySQL adatbázis kezelő rendszer és ezen kívüli más komponensek integrált csomagja, ami jelen esetben a Xampp lesz.

Követelmények a Xampp által biztosított webszerverhez:

- Windows, Linux vagy Mac OS operációs rendszer
- 5-8 GB tárhely (Minél több új terméket adunk kínálatunkhoz, úgy változhat a tárhely igény)
- Internetböngésző, javasolt a Mozilla Firefox vagy Google Chrome
- MySQL legfrissebb verziója
- PHP legfrissebb verziója

2.3 Főoldal

A weblapunk megnyitásakor, a főoldal az első dolog, ami a szemünk elé tárul. Itt több fontos dolog is található, mint például a navigációs sor, vagy szakmai néven hívva 'navbar' is. Ez a navigációs sáv majdnem az összes oldalon elérhető, mert egy fontos része az oldalon való navigálásnak. A főoldalon jelennek meg a termékkategóriák, az adott kategóriára kattintva átugorhatunk az adott kategóriába tartozó termékek megtekintésére.



2.4 Footer

Az összes oldalon található egy lábléc, azaz 'footer' is. Ezen található a készítő neve, a készítés éve, és egy hivatkozás a főoldalra.

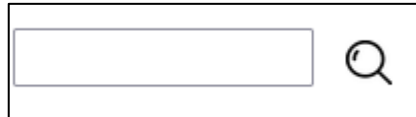
2.5 Navigációs sáv

A navigációs soron található az oldalon való navigáció fő része. Kettő változata van, egy akkor, amikor a felhasználó nincs bejelentkezve, és amikor a felhasználó bejelentkezett.

2.5.1 Termékek, Kategóriák

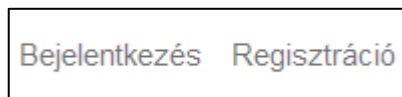


A 'navbar' első része a termékek, és a kategóriák közti navigációra használható. A kategóriáknál lenyíló elemekből lehet választani, vagy az összes termék is megtekinthető, az 'Összes termék' menüpontra kattintva.



A termékek közötti keresésre is van lehetőség, a 'navbar' ezen részével. Ez egy keresősáv, amibe a felhasználó beírja egy termék nevét, vagy márkáját, és rányom a nagyító gombra, majd előhossa a keresésnek megfelelő termékeket.

2.5.2 Regisztráció, Bejelentkezés



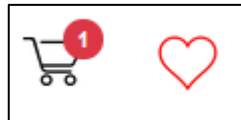
A Bejelentkezés vagy Regisztráció menüpontra kattintva, a megfelelő oldalra ugrik. Ez a rész csak akkor jelenik meg, ha a felhasználó nincs bejelentkezve.

2.5.3 Konfigurációs rész

Konfigurációk Konfiguráció összerakó

Erről a részből a számítógépkonfigurációkkal kapcsolatos oldalakra lehet látogatni. A 'Konfigurációk' menüpontra kattintva a saját, és más felhasználók által megosztott konfigurációk tekinthetők meg, a 'Konfiguráció összerakó' oldalon pedig saját konfigurációt lehet összeállítani.

2.5.4 Kosár, kedvencek



Ezen gombokkal juthatunk el a kosárba, vagy a kedvenc termékek oldalra. Ezek csak bejelentkezett felhasználók által használhatók.

2.6 Egy termék oldala

Egy adott termékre kattintva az oldal átvész a termék oldalára. Itt látható egy kép a termékről, az ára, illetve az adott termék megadott tulajdonságai is. Kedvencek közé mentheti ezt a terméket és megadható egy darabszám is, amennyire a vásárlónak szüksége lenne, annyit helyezhet a kosárba.



Intel Core i7-13700KF

178 190 Ft



Kedvencek közé:

Jellemzők

Processzor foglalat	Intel LGA-1700
Processzor órajel	2.5 GHz
Maximum órajel	5.4 GHz
Magok száma	16 db
Szálak száma	24 db

2.7 Kosár

A kosárban a felhasználó megtekintheti azokat a termékeket, amelyeket hozzáadott a kosárhoz. Itt tud törölni egy-egy terméket és tudja a mennyiséget növelni és csökkenteni. Ennek megfelelően dinamikusan változik a termék ára és a végösszeg is.

Termék	Ár	Mennyiség
	Asus ROG MAXIMUS Z790 HERO 273 390 Ft	1 db - + Törölés
	GIGABYTE Z790 AORUS Elite AX 110 291 Ft	1 db - + Törölés

Végösszeg: 383 681 Ft

Teljes kosár törlése

Tovább a rendeléshez

A 'Teljes kosár törlése' gombra kattintva egy kattintással a teljes tartalom törölhető, a 'Tovább a rendelésre' gombra kattintva, pedig a rendelés oldalra dob át.

2.8 Rendelés

Ezen az oldalon adhatja meg a felhasználó a szállítási címet, telefonszámot és választhatja ki a fizetési módot.

Teljes név

Citrom Cecil

Irányítószám

2600

Település

Vác

Szállítási cím

Petőfi Sándor u. 68.

Telefonszám

06301112233

Fizetési módok

▼

Rendelés



Sikeresen megrendelve!

Megrendelő neve

Halo Lajos

Elérhetőségei

halo.lajos86@gmail.com

06702223344

Szállítási cím

2600 Vác, Kossuth tér 1.

Végösszeg

383 681 Ft

A főoldalon a 'Rendelési előzmények' menüpontra kattintva megtekinthetők a rendelés tételei.

Sikeres rendelés esetén ez az oldal fogadja a felhasználót, itt megtekintheti a megadott adatait és visszatérhet a főoldalra.

2.9 Rendelési előzmények

'Rendelési előzmények' menüpontra kattintva megtekinthetővé válnak a leadott rendelések. Ezekről néhány információ és a rendelt termékek láthatóak.

Rendelési előzmények

Dátum: 2024-04-25 18:26:44

Végösszeg: 383 681 Ft

Név	Ár	Mennyiség
Asus ROG MAXIMUS Z790 HERO	273 390 Ft	1 db
GIGABYTE Z790 AORUS Elite AX	110 291 Ft	1 db

Dátum: 2024-04-25 18:28:51

Végösszeg: 349 580 Ft

Név	Ár	Mennyiség
Sony PlayStation 5	349 580 Ft	2 db

2.10 Konfiguráció összeállító

Ez az oldal csak bejelentkezett felhasználóknak elérhető. Itt tudnak saját számítógép-konfigurációt összeállítani a webshop kínálatában levő hardverelemekből. Az elemeket egy legördülő menüből lehet kiválasztani minden egyes típusnál.

Processzor :	Intel - Core i7-13700KF
Videókártya :	Intel - Core i7-13700KF AMD - Ryzen 5 5500
Alaplap :	Intel - Core i5-12400F
Memória :	Intel - Core i9-14900K
Merevlemez :	AMD - Ryzen 7 7700X AMD - Ryzen 9 7950X3D
SSD :	Samsung - 970 EVO Plus SSD
Tápegység :	Corsair - RM750x
Processzor hűtő :	Noctua - NH-D15
Számítógépház :	NZXT - H510

Az összeállítás után kettő lehetősége van a felhasználónak. Hozzáadni a megosztott konfigurációkhoz a 'Beküldés' gombra kattintva, vagy beküldeni, és a hardverelemeket a kosárhoz adni a 'Beküldés és kosár' gombra kattintva.

Konfiguráció összeállító

Processzor :	Intel - Core i7-13700KF
Videókártya :	AMD - Radeon RX 7900 XTX
Alaplap :	Asus - ROG MAXIMUS Z790 HERO
Memória :	KINGSTON - FURY Beast DIMM 2x32GB
Merevlemez :	Western Digital - WD Blue 1TB WD10EZRX
SSD :	Samsung - 970 EVO Plus SSD
Tápegység :	Corsair - RM750x
Processzor hűtő :	Noctua - NH-D15
Számítógépház :	NZXT - H510

[Beküldés](#) [Beküldés és kosár](#)


2.11 Konfiguráció értékelő


Ez az oldal csak bejelentkezett felhasználók számára elérhető. Itt lehet megtekinteni a saját és más felhasználók által létrehozott konfigurációkat, és értékelni egy like-al, vagy

dislike-al. Egyszerre maximum négy darab konfiguráció jelenhet meg. Mindegyik konfiguráció külön elemként jelenik meg az oldalon. Az oldal jelzi, hogy a felhasználó mivel értékelte az adott konfigurációt, és annak megfelelően színezi az adott gombot.

Gobar

Processzor	Intel - Core i7-13700KF
Videókártya	AMD - Radeon RX 7900 XTX
Alaplap	Asus - ROG MAXIMUS Z790 HERO
Memória	KINGSTON - FURY Renegade RGB DIMM 2x8GB
Merevlemez	Seagate - Exos X20 3.5" HDD
SSD	Samsung - 970 EVO Plus SSD
Tápegység	EVGA - SuperNOVA 650 G5
Processzor hűtő	Noctua - NH-D15
Számítógépház	Fractal Design - Design Meshify C

 1

 0

Az oldal alján található kettő gomb, amelyek a megjelenített konfigurációk váltására szolgálnak. Ezekkel lehet lapozni az előző, vagy következő oldalra. Középen pedig megjelenik, hogy a felhasználó hányadik oldalon van jelenleg.

3 Irodalomjegyzék

3.1 Internetes tartalmak

- **W3schools:** <https://www.w3schools.com/>
- **Stackoverflow:** <https://stackoverflow.com/>
- **Bootstrap:** <https://getbootstrap.com/>
- **PHP:** <https://www.php.net/>
- **JavaScript:** <https://www.javascript.com/>
- **jQuery:** <https://jquery.com/>
- **MySQL:** <https://www.mysql.com/>

3.2 Könyvek

- Robert C. Martin: **Tiszta kód** (2008)
- David Powers: **PHP Object-Oriented Solutions** (2010)
- Matt Beaumont: **PHP eCommerce** (2010)
- Richard Blum: **PHP, MySQL, & JavaScript All-in-One For Dummies** (2018)
- Jennifer Niederst Robbin: **Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics** (2018)
- Christian Nagel: **Professional C# and .NET - 2021 Edition** (2021)
- Rudolf Pecinovský: **OOP - Learn Object Oriented Thinking and Programming** (2013)

4 Melléklet

- **Projektünk GitHub linkje:**

<https://github.com/DallosRub/TornadoPC>