

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

# **SZAKDOLGOZAT**

**Balogh Ádám – Schönberger Dominik**

**2023.**

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



**SZAKDOLGOZAT**

**Használt Sportszer**

Konzulens: Wiezl Csaba

Készítette: Balogh Ádám  
Schönberger Dominik

## Hallgatói nyilatkozat

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

---

Balogh Ádám

---

Schönberger Dominik

# Konzultációs lap

Vizsgázók neve: Balogh Ádám, Schönberger Dominik

Szaktervezési címe: Használt Sportszer Webshop

Program nyújtotta szolgáltatások:

- Bejelentkezés/regisztráció
- Saját hirdetés feltöltés/szerkesztés
- Sportszerek eladása
- Profil szerkesztés
- Hirdetés böngészés
- Hirdetések törlése
- Profilok megtekintése
- Profilok törlése
- admin profilok regisztrálása
- admin profilok változtatása

Sorszám	A konzultáció időpontja	A konzulens aláírása
1.	2022.10.10.	
2.	2022.11.14.	
3.	2022.12.12.	
4.	2023.01.16.	
5.	2023.02.13.	
6.	2023.03.13.	

A szaktervezési beadható:

Vác, 2023. ....

.....

Konzulens

A szaktervezési átvettem:

Vác, 2023. ....

.....

A szakképzést folytató  
intézmény felelőse

# Tartalomjegyzék

Hallgatói nyilatkozat.....	3
Konzultációs lap.....	4
Tartalomjegyzék .....	5
Témaválasztás .....	7
<b>1 Fejlesztői dokumentáció .....</b>	<b>8</b>
<b>1.1 Rövid alkalmazás ismertetés.....</b>	<b>8</b>
<b>1.2 Felhasznált alkalmazások .....</b>	<b>8</b>
1.2.1 Visual Studio Code .....	8
1.2.2 Brackets.....	8
<b>1.3 Felhasznált fejlesztői nyelvek .....</b>	<b>9</b>
1.3.1 PHP .....	9
1.3.2 HTML .....	9
1.3.3 CSS.....	9
1.3.4 Javascript.....	10
<b>1.4 Felhasznált technológiák .....</b>	<b>10</b>
1.4.1 PHPMailer.....	10
1.4.2 JSON .....	11
1.4.3 MySql.....	11
1.4.4 Xampp .....	11
1.4.5 Bootstrap 5 .....	11
<b>1.5 Használati esetmodell, szerepkörök.....</b>	<b>12</b>
<b>1.6 Adatbázis.....</b>	<b>12</b>
1.6.1 Adatszerkezet.....	12
1.6.2 Adattáblák .....	13
1.6.3 Utólag hozzáadott adattáblák.....	18
1.6.4 Adatbázis kapcsolatok.....	22
<b>1.7 Modulok és kód ismertetése .....</b>	<b>23</b>
1.7.1 Adatbázis csatlakozás.....	23
1.7.2 Admin felület.....	23
1.7.3 Authentication .....	25
1.7.4 CSS / Cascading Style Sheets.....	26
1.7.5 Képek.....	27
1.7.6 HTML kód nélküli funkciók.....	27
1.7.7 HTML oldalak/funkciók .....	33

1.8	Tesztelés/Tesztesetek .....	38
1.8.1	Tesztesetek.....	39
1.9	Továbbfejlesztési lehetőségek .....	40
2	Felhasználói dokumentáció.....	41
2.1	Alkalmazás rövid ismertetése .....	41
2.2	Rendszerkövetelmények .....	41
2.3	Alkalmazás telepítése, szükséges beállítások .....	41
2.3.1	XAMPP telepítése .....	41
2.3.2	XAMPP elindítása .....	42
2.3.3	A fájlok bemásolása a htdocs mappába .....	43
2.3.4	Adatbázis futtatása .....	43
2.3.5	Webáruház megnyitása.....	43
2.4	Alkalmazás használata.....	43
2.4.1	Navigáció, lábjegyzet.....	43
2.4.2	Regisztráció, Bejelentkezés .....	44
2.4.3	Főoldal.....	45
2.4.4	Kosaram.....	47
2.4.6	Rendelés leadása .....	48
2.4.7	Kontakt .....	49
2.4.8	Hirdetés feltöltése.....	50
2.4.9	Profilom.....	51
2.5	Felhasználható eszközök .....	52
3	Irodalomjegyzék .....	56
3.1	Internetes források .....	56
3.2	Könyvek.....	56
4	Mellékletek.....	57

## Témaválasztás

Az év elején alaposan átgondoltuk szakdolgozatunk témáját, és végül a webes irány mellett döntöttünk, hiszen mindketten nagyobb kedvvel és tapasztalattal rendelkezünk a webes alkalmazásokhoz szükséges nyelvek terén. Ezt követően el kellett döntenünk, milyen témában építjük fel az általunk tervezett webáruházat. A sport mindkettőnk életében meghatározó szerepet tölt be, és ráadásul nem találtunk olyan használt sporteszközökkel foglalkozó webáruházat, amely kifejezetten erre a célra fókuszál. Ezért úgy határoztunk, hogy ezt a témakört helyezzük a középpontba, és az általunk tervezett oldal lehetőséget biztosít a használt sporteszközök vásárlására és eladására egyaránt.

Célunk, hogy minél szélesebb körben ismertté és elérhetővé tegyük az általunk létrehozott webáruházat, és minél több visszajelzést kapjunk a felhasználóktól, hogy hatékonyan fejleszthessük az alkalmazást. Fontos számunkra, hogy a még felhasználható sporteszközök ne a szemétként kerüljenek ki, hanem olyan kezekbe kerüljenek, amelyeknek örömet okoznak és hosszú távon hasznosak lehetnek. Emellett szeretnénk támogatni azokat a sportolókat is, akik nem engedhetik meg maguknak a legújabb felszereléseket, de szívesen vásárolnának jó minőségű, áron aluli eszközöket.

Az általunk létrehozott webáruház kialakításában Schönberger Dominik vállalta a különböző funkciók megvalósítását, beleértve az adatbázissal való csatlakozást, a beléptetést, a regisztrációt, a profilok funkcióit, mint a kijelentkezés, profil adatok, illetve a kiszállítási cím módosítása, a hirdetések feltöltését és szerkesztését, az e-mail rendszert és az admin felületet. Balogh Ádám feladatai közé tartozott a frontend felépítése, reszponzív oldalak kialakítása, az adatbázis létrehozása és a Schönberger Dominik által írt funkciók beintegrálása projectmunkába. Fontosnak tartjuk megemlíteni, hogy a fejlesztés során csapatunk nem használt előre megírt weboldalakot, a template oldalak használata szembementek a mi elveinkkel, szerettük volna, hogyha a projekt 100%-ban a saját munkánk gyümölcse. Természetesen a feladatokban voltak átfedések, és amikor szükséges volt, segítettünk egymásnak. Csapatunk harmadik tagja Hajszter Botond volt, aki a tanév vége felé befejezte tanulmányait, így a rábízott feladatok is hirtelen a mi nyakunkba szakadtak. Ezen probléma kiküszöbölése kisebb-nagyobb sikerrel sikerült.

# 1 Fejlesztői dokumentáció

## 1.1 Rövid alkalmazás ismertetés

Alkalmazásunk egy sportszerekkel foglalkozó webshop. Felhasználóink kényelmesen vásárolhatnak legyenek bárhol az országban. Célunk az volt, hogy azokat az embereket is ösztönözzük a sportolásra akiknek nem áll módjukban megvásárolni a drágábbnál drágább sportszereket vagy csak nem szeretnék ennyi pénz áldozni rá.

## 1.2 Felhasznált alkalmazások

### 1.2.1 Visual Studio Code

A **Visual Studio Code** ingyenes, nyílt forráskódú kódszerkesztő, melyet a Microsoft fejleszt Windows, Linux és macOS operációs rendszerekhez. Támogatja a hibakeresőket, valamint beépített Git támogatással rendelkezik, továbbá képes az intelligens kódkiegészítésre az IntelliSense segítségével. A VSCode-ban a felhasználók megváltoztathatják a kinézetet (témát), megváltoztathatják a szerkesztő gyorsbillentyű-kiosztását, az alapértelmezett beállításokat és még sok egyebet. Támogatja a kiegészítőket, melyek segítségével további funkciók, testreszabási lehetőségek érhetőek el. A VSCode az Electron nevű keretrendszeren alapszik, amellyel asztali környezetben futtatható Node.js alkalmazások fejleszthetőek.<sup>1</sup>

### 1.2.2 Brackets

A **Brackets** ingyenes nyílt forráskódú szöveg- és forráskódszerkesztő melyet HTML-ben, CSS-ben és JavaScriptben írtak kifejezetten webfejlesztéshez. A programot az Adobe fejlesztette és karbantartása a GitHub-on történik. A Brackets elérhető Macintos, Linux és Windows platformokra. 2021. szeptember 1-jén az Adobe befejezte a támogatást.<sup>2</sup>

---

<sup>1</sup> [https://hu.wikipedia.org/wiki/Visual\\_Studio\\_Code](https://hu.wikipedia.org/wiki/Visual_Studio_Code)

<sup>2</sup> <https://hu.wikipedia.org/wiki/Brackets>



## 1.3 Felhasznált fejlesztői nyelvek

### 1.3.1 PHP

A PHP elterjedt nyílt forráskódú szerver-oldali programozási nyelv, amely az SSI (Server Side Includes) és a Perl ötvözésére hivatott. Szintaktikája leginkább a C programozási nyelvéhez hasonlít. Megalkotója Rasmus Lerdorf. Mára azonban egy egész csapat foglalkozik a nyelvvel. Széles körben való elterjedtségét C programozási nyelvhez való hasonlóságának, sebességének és egyszerűségének köszönheti. A PHP nem csak szerveroldali alkalmazásoknál alkalmazható, bár asztali alkalmazások esetében nem terjedt el.<sup>3</sup>

### 1.3.2 HTML

A **HTML** egy leíró nyelv, melyet weboldalak készítéséhez fejlesztettek ki, és mára már internetes szabvánnyá vált a W3C (World Wide Web Consortium) támogatásával. Egy HTML dokumentum alapértelmezésként ISO-8859-1, azaz nyugat-európai kódolást használ. Gyakran előforduló hiba szokott lenni, hogy nincs beállítva a charset paraméter a fejléc content attribútumában, annak ellenére, hogy a dokumentum nem nyugat-európai kódolású szöveget tartalmaz.<sup>4</sup>

### 1.3.3 CSS

A CSS (Cascading Style Sheets) egy olyan leírónyelv, melynek segítségével különböző stíluslapokat hozhatunk létre és ágyazhatunk be HTML honlapjainkba. Ezek a stíluslapok befolyásolják az oldal megjelenését (design-ügyileg kiváló nyelv), meghatározhatjuk vele, hogy az egyes HTML tagek (bekezdések, táblázatok, címsorok stb.) hogyan jelenjenek meg, meghatározhatjuk a méretüket, színüket, stílusukat, típusukat stb.<sup>5</sup>

---

<sup>3</sup> [https://www.inf.u-szeged.hu/~gnemeth/adatbgyak/exe/MySQL\\_XAMPP\\_JDBC/a\\_php\\_nyelvi\\_elemeinek\\_rvid\\_ttekintse.html](https://www.inf.u-szeged.hu/~gnemeth/adatbgyak/exe/MySQL_XAMPP_JDBC/a_php_nyelvi_elemeinek_rvid_ttekintse.html)

<sup>4</sup> <https://hu.wikipedia.org/wiki/HTML>

<sup>5</sup> <https://prog.hu/cikkek/100430/bevezetes-a-css-alapjaiba>

### 1.3.4 Javascript

A JavaScript, mint neve is mutatja, egy script nyelv, amit legelőször a Netscape Navigator 2.0 támogatott. A Javascript a WWW-ért született. Jelenleg nem is létezik más implementációja, mint ami a böngészőkben és a webszerverekben fut. A Sun és a Netscape által kifejlesztett parancsnyelvnek két változata létezik. Az ún. LiveWire lehetővé teszi, hogy szerver oldali alkalmazásokat készítsünk, ami hasonló a CGI-hez (Common Gateway interface). A másik a kliens oldali, ami lehetővé teszi egyszerűbb programozási feladatok elkészítését a Weben (pl. mozgó HTML elemek, interaktív funkciók). A kliens és a szerver oldali rész egy külön világ, szintakszisuk ugyan megegyezik, de a programozói interfész teljesen más. A kliens oldali JavaScript a böngészőkön keresztül jelenik meg számunkra. Mégpedig úgy, hogy a JavaScript kódot az általános HTML kód közé ékeljük. Amikor a böngésző megjeleníti a HTML kódot, akkor a `<script>` és `</script>` közé írt kódrészleteket nem megjeleníti, hanem értelmezi, és futtatja, ezáltal dinamikus viselkedést kölcsönöz az oldalnak. (Van lehetőség a JavaScript kódot egy külön állományba helyezni és a HTML-ből pusztán hivatkozni erre az állományra.)<sup>6</sup>

## 1.4 Felhasznált technológiák

### 1.4.1 PHPMailer

A PHPMailer egy kódkönyvtár, amely biztonságosan és egyszerűen küld e-maileket PHP-kódon keresztül egy webszerverről. 2001-től a PHPMailer az egyik legnépszerűbb megoldása az e-mail küldéseknek a PHP-n belül.<sup>7</sup>

---

<sup>6</sup> <http://nyelvek.inf.elte.hu/leirasok/JavaScript/index.php?chapter=1>

<sup>7</sup> <https://en.wikipedia.org/wiki/PHPMailer>

### 1.4.2 JSON

A különféle adatokkal való munkavégzés ismerete fontossá vált. A programozóknak, fejlesztőknek és informatikai szakembereknek át kell vinniük a feltöltött adatstruktúrákat bármely nyelvről olyan formátumba, amelyet más nyelvek és platformok is felismernek. A JavaScript Object Notation (JSON) az az adatsere-formátum, amely ezt lehetővé teszi.<sup>8</sup>

### 1.4.3 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.<sup>9</sup>

### 1.4.4 Xampp

Aki adatbázissal támogatott webszervert szeretne üzemeltetni, annak össze kell hangolnia a webszerver, az adatbázisszerver és a PHP szolgáltatásait, valamint a kommunikációs portokat. A XAMPP fejlesztői felismerték a lehetőséget, hogy ezeket a beállításokat mindenkinek el kell végeznie, viszont nem biztos, hogy mindenki rendelkezik adminisztrátori jogokkal és ilyen szintű ismeretekkel, ezért összeállítottak egy keretrendszert, amelyet kezdő fejlesztők is könnyedén használhatnak.<sup>10</sup>

### 1.4.5 Bootstrap 5

A Bootstrap 5 a Bootstrap legújabb verziója, amely a legnépszerűbb HTML, CSS és JavaScript keretrendszer reszponzív webhelyek létrehozásához.<sup>11</sup>

---

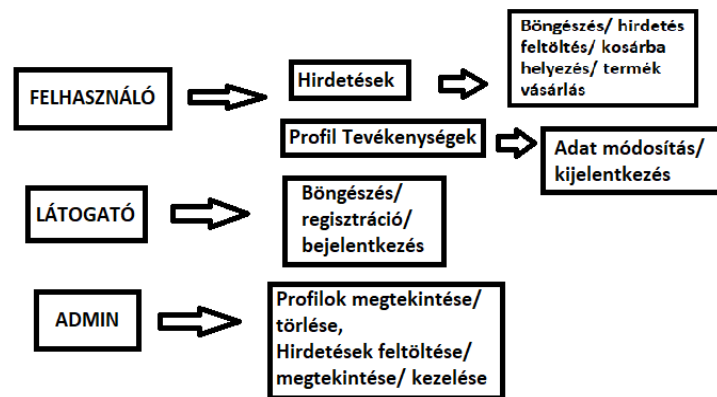
<sup>8</sup> <https://www.oracle.com/database/what-is-json/>

<sup>9</sup> <https://hu.wikipedia.org/wiki/MySQL>

<sup>10</sup> [https://www.inf.u-szeged.hu/~gnemeth/adatbgyak/exe/MySQL\\_XAMPP\\_JDBC/a\\_xampp\\_keretrendszer\\_s\\_hasznlata.html](https://www.inf.u-szeged.hu/~gnemeth/adatbgyak/exe/MySQL_XAMPP_JDBC/a_xampp_keretrendszer_s_hasznlata.html)

<sup>11</sup> <https://www.w3schools.com/bootstrap5/>

## 1.5 Használati esetmodell, szerepkörök



## 1.6 Adatbázis

### 1.6.1 Adatszerkezet

**MySQL kliens verziója:** 4.9.0.1

**Adatbázis neve:** sportszer\_ab

**Tárolómotor:** InnoDB

**Alapértelmezett illesztés:** utf8\_hungarian\_ci

**SQL parancs:**

```
CREATE DATABASE IF NOT EXISTS `sportszer_ab`  
DEFAULT CHARACTER SET utf8 COLLATE utf8_hungarian_ci;
```

## 1.6.2 Adattáblák

### 1.6.2.1 'allapotok' tábla

A termék lehetséges állapotát tartalmazza (pl.: új, használt).

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
<input type="checkbox"/>	1	allapotok_id	int(4)		Nem	Nincs		
<input type="checkbox"/>	2	allapotok_megn	varchar(20) utf8_hungarian_ci		Nem	Nincs		

- allapotok\_id: elsődleges kulcs a termék állapotának azonosítója
- allapotok\_megn: a termék állapotának megnevezése

```
CREATE TABLE IF NOT EXISTS `allapotok` (  
  `allapotok_id` int(4) NOT NULL,  
  `allapotok_megn` varchar(20) COLLATE utf8_hungarian_ci  
  NOT NULL,  
  PRIMARY KEY (`allapotok_id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8  
COLLATE=utf8 hungarian ci;
```

### 1.6.2.2 'markak' tábla

A termék márkáját tartalmazza

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
<input type="checkbox"/>	1	markak_id	int(6)		Nem	Nincs		
<input type="checkbox"/>	2	markak_megn	varchar(40) utf8_hungarian_ci		Nem	Nincs		

- markak\_id: elsődleges kulcs, a márka azonosítója
- markak\_megn: a márka megnevezése

**Tábla létrehozása:**

```
CREATE TABLE IF NOT EXISTS `markak` (  
  `markak_id` int(6) NOT NULL,  
  `markak_megn` varchar(40) COLLATE utf8_hungarian_ci  
  NOT NULL,  
  PRIMARY KEY (`markak_id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8  
COLLATE=utf8_hungarian_ci;
```

### 1.6.2.3 'nemek' tábla

A termék célközönségét tartalmazza a nem alapján (férfi, női, unisex)

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
<input type="checkbox"/> 1	nemek_id	int(4)			Nem	Nincs		
<input type="checkbox"/> 2	nemek_megn	varchar(50)	utf8_hungarian_ci		Nem	Nincs		

- nemek\_id: elsődleges kulcs, a nem azonosítója
- nemek\_megn: a nem megnevezése

**Tábla létrehozása:**

```
CREATE TABLE IF NOT EXISTS `nemek` (  
  `nemek_id` int(4) NOT NULL,  
  `nemek_megn` varchar(50) COLLATE utf8_hungarian_ci  
  NOT NULL,  
  PRIMARY KEY (`nemek_id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8  
COLLATE=utf8_hungarian_ci;
```

### 1.6.2.4 'sportagak' tábla

A termék sportági besorolását tartalmazza

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
<input type="checkbox"/> 1	sportagak_id	int(6)			Nem	Nincs		AUTO_INCREMENT
<input type="checkbox"/> 2	sportagak_megn	varchar(60)	utf8_hungarian_ci		Nem	Nincs		

- sportagak\_id: elsődleges kulcs, a sportág azonosítója
- sportagak\_megn: a sportág megnevezése

**Tábla létrehozása:**

```
CREATE TABLE IF NOT EXISTS `sportagak` (  
  `sportagak_id` int(6) NOT NULL,  
  `sportagak_megn` varchar(60) COLLATE utf8_hungarian_ci  
  NOT NULL,  
  PRIMARY KEY (`sportagak_id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8  
COLLATE=utf8_hungarian_ci;
```

#### 1.6.2.5 'szinek' tábla

A termék domináns színét jelöli

	#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
<input type="checkbox"/>	1	szinek_id	int(4)			Nem	Nincs		
<input type="checkbox"/>	2	szinek_megn	varchar(20)	utf8_hungarian_ci		Nem	Nincs		

- szinek\_id: a domináns szín azonosítója
- szinek\_megn: a domináns szín megnevezése

Tábla létrehozása:

```
CREATE TABLE IF NOT EXISTS `szinek` (  
  `szinek_id` int(4) NOT NULL,  
  `szinek_megn` varchar(20) COLLATE utf8_hungarian_ci  
  NOT NULL,  
  PRIMARY KEY (`szinek_id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8  
COLLATE=utf8_hungarian_ci;
```

#### 1.6.2.6 'tipusok' tábla

A termék típusát jelöli (pl.: cipő, melegítő felső)

	#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
<input type="checkbox"/>	1	tipusok_id	int(6)			Nem	Nincs		
<input type="checkbox"/>	2	tipusok_megn	varchar(40)	utf8_hungarian_ci		Nem	Nincs		

- tipusok\_id: a termék típusának azonosítója
- tipusok\_megn: a termék típusának megnevezése

Tábla létrehozása:

```
CREATE TABLE IF NOT EXISTS `tipusok` (  
  `tipusok_id` int(6) NOT NULL,  
  `tipusok_megn` varchar(40) COLLATE utf8_hungarian_ci  
  NOT NULL,  
  PRIMARY KEY (`tipusok_id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8  
COLLATE=utf8_hungarian_ci;
```

#### 1.6.2.7 'hirdetesekek' tábla

Ez a tábla egy átfogó tábla, tartalmazza a feltöltött hirdetések adatait.

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
<input type="checkbox"/>	1 id 🔑	int(11)			Nem	Nincs		AUTO_INCREMENT
<input type="checkbox"/>	2 állapotok_id 🔑	int(11)			Nem	Nincs		
<input type="checkbox"/>	3 markak_id 🔑	int(11)			Nem	Nincs		
<input type="checkbox"/>	4 nemek_id 🔑	int(11)			Nem	Nincs		
<input type="checkbox"/>	5 sportagak_id 🔑	int(11)			Nem	Nincs		
<input type="checkbox"/>	6 szinek_id 🔑	int(11)			Nem	Nincs		
<input type="checkbox"/>	7 tipusok_id 🔑	int(11)			Nem	Nincs		
<input type="checkbox"/>	8 ar	int(11)			Nem	Nincs		
<input type="checkbox"/>	9 leiras	varchar(500)	utf8_hungarian_ci		Nem	Nincs		
<input type="checkbox"/>	10 kep1	text	utf8_hungarian_ci		Nem	Nincs		
<input type="checkbox"/>	11 kep2	text	utf8_hungarian_ci		Nem	Nincs		
<input type="checkbox"/>	12 kep3	text	utf8_hungarian_ci		Nem	Nincs		
<input type="checkbox"/>	13 kep4	text	utf8_hungarian_ci		Nem	Nincs		
<input type="checkbox"/>	14 kep5	text	utf8_hungarian_ci		Nem	Nincs		

- id: elsődleges kulcs, a hirdetés azonosítója
- állapotok\_id: idegen kulcs, az állapot azonosítója
- markak\_id: idegen kulcs, a márka azonosítója
- nemek\_id: idegen kulcs, a nem azonosítója
- sportagak\_id: idegen kulcs, a sportág azonosítója
- szinek\_id: idegen kulcs, a domináns szín azonosítója
- tipusok\_id: idegen kulcs, a termék típusának azonosítója
- ar: a termék ára
- leiras: a termék leírása
- kep1: a hirdetés 1. képe, ez a hirdetés „profilképe”
- kep2: a hirdetés 2. képe
- kep3: a hirdetés 3. képe
- kep4: a hirdetés 4. képe
- kep5: a hirdetés 5. képe



**Tábla létrehozása:**

```
CREATE TABLE IF NOT EXISTS `hirdetesek` (  
  `id` int(11) NOT NULL,  
  `allapotok_id` int(11) NOT NULL,  
  `markak_id` int(11) NOT NULL,  
  `nemek_id` int(11) NOT NULL,  
  `sportagak_id` int(11) NOT NULL,  
  `szinek_id` int(11) NOT NULL,  
  `tipusok_id` int(11) NOT NULL,  
  `ar` int(11) NOT NULL,  
  `leiras` varchar(500) COLLATE utf8_hungarian_ci NOT  
NULL,  
  `telszam` varchar(40) COLLATE utf8_hungarian_ci NOT  
NULL,  
  `kep1` text COLLATE utf8_hungarian_ci NOT NULL,  
  `kep2` text COLLATE utf8_hungarian_ci NOT NULL,  
  `kep3` text COLLATE utf8_hungarian_ci NOT NULL,  
  `kep4` text COLLATE utf8_hungarian_ci NOT NULL,  
  `kep5` text COLLATE utf8_hungarian_ci NOT NULL,  
  PRIMARY KEY (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8  
COLLATE=utf8_hungarian_ci;
```

### 1.6.3 Utólag hozzáadott adattáblák

A következő 4 tábla a weboldal fejlesztésének későbbi szakaszában lett hozzáadva az adatbázishoz. Ezen táblák kapcsolatainak felépítése, javítása a továbbfejlesztés része.

#### 1.6.3.1 'admin' tábla

Az fejlesztői felülethez szükséges adattábla

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
<input type="checkbox"/> 1	<b>id</b>	int(10)			Nem	Nincs		AUTO_INCREMENT
<input type="checkbox"/> 2	<b>name</b>	varchar(20)	utf8_hungarian_ci		Nem	Nincs		
<input type="checkbox"/> 3	<b>password</b>	varchar(50)	utf8_hungarian_ci		Nem	Nincs		

- id: elsődleges kulcs, a fejlesztői(admin) profil azonosítója
- name: a fejlesztői profil neve
- password: a profil jelszava

**Tábla létrehozása:**

```
CREATE TABLE IF NOT EXISTS `admin` (  
  `id` int(10) NOT NULL,  
  `name` varchar(20) COLLATE utf8_hungarian_ci NOT NULL,  
  `password` varchar(50) COLLATE utf8_hungarian_ci NOT  
  NULL  
  PRIMARY KEY (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8  
COLLATE=utf8_hungarian_ci;NOT NULL
```

#### 1.6.3.2 'felhasznalo' tábla

A regisztrált felhasználók adatait tartalmazó tábla

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
<input type="checkbox"/> 1	<b>id</b>	int(4)			Nem	Nincs		AUTO_INCREMENT
<input type="checkbox"/> 2	<b>name</b>	varchar(30)	utf8_hungarian_ci		Nem	Nincs		
<input type="checkbox"/> 3	<b>email</b>	varchar(50)	utf8_hungarian_ci		Nem	Nincs		
<input type="checkbox"/> 4	<b>number</b>	varchar(50)	utf8_hungarian_ci		Nem	Nincs		
<input type="checkbox"/> 5	<b>password</b>	varchar(60)	utf8_hungarian_ci		Nem	Nincs		
<input type="checkbox"/> 6	<b>address</b>	varchar(300)	utf8_hungarian_ci		Nem	Nincs		

- id: elsődleges kulcs, a felhasználó azonosítója
- name: a felhasználó neve
- email: a felhasználó e-mail címe
- number: a felhasználó telefonszáma
- password: a felhasználó jelszava
- address: a felhasználó lakcíme

#### Tábla létrehozása:

```
CREATE TABLE IF NOT EXISTS `felhasznalo` (  
  `id` int(4) NOT NULL,  
  `name` varchar(30) COLLATE utf8_hungarian_ci NOT NULL,  
  `email` varchar(50) COLLATE utf8_hungarian_ci NOT  
NULL,  
  `number` varchar(50) COLLATE utf8_hungarian_ci NOT  
NULL,  
  `password` varchar(60) COLLATE utf8_hungarian_ci NOT  
NULL,  
  `address` varchar(300) COLLATE utf8_hungarian_ci NOT  
NULL  
  PRIMARY KEY (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8  
COLLATE=utf8_hungarian_ci;
```

#### 1.6.3.3 'cart' tábla

Minden felhasználó kosárba tett termékeinek az adatait tartalmazza

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
<input type="checkbox"/> 1	<b>id</b> 	int(100)			Nem	Nincs		AUTO_INCREMENT
<input type="checkbox"/> 2	<b>user_id</b>	int(100)			Nem	Nincs		
<input type="checkbox"/> 3	<b>hir_id</b>	int(100)			Nem	Nincs		
<input type="checkbox"/> 4	<b>marka</b>	varchar(100)	utf8_hungarian_ci		Nem	Nincs		
<input type="checkbox"/> 5	<b>ar</b>	int(10)			Nem	Nincs		
<input type="checkbox"/> 6	<b>termektipus</b>	varchar(100)	utf8_hungarian_ci		Nem	Nincs		
<input type="checkbox"/> 7	<b>kep1</b>	text	utf8_hungarian_ci		Nem	Nincs		

- **id**: elsődleges kulcs, a kosár azonosítója
- **user\_id**: a felhasználó azonosítója, aki a terméket a kosárba helyezte
- **hir\_id**: azon hirdetés, termék azonosítója, melyet a felhasználó a kosárba tett
- **marka**: a kosárba helyezett termék márkája
- **ar**: a kosárba helyezett termék ára
- **termektipus**: a kosárba helyezett termék típusa (pl.: cipő, melegítő felső)
- **kep1**: a kosárba helyezett termék elsőszámú képe

### Tábla létrehozása:

```
CREATE TABLE IF NOT EXISTS `cart` (  
  `id` int(100) NOT NULL,  
  `user_id` int(100) NOT NULL,  
  `hir_id` int(100) NOT NULL,  
  `marka` varchar(100) COLLATE utf8_hungarian_ci NOT  
NULL,  
  `ar` int(10) NOT NULL,  
  `termektipus` varchar(100) COLLATE utf8_hungarian_ci  
NOT NULL,  
  `kep1` text COLLATE utf8_hungarian_ci NOT NULL,  
  PRIMARY KEY (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8  
COLLATE=utf8_hungarian_ci;) ENGINE=InnoDB DEFAULT  
CHARSET=utf8 COLLATE=utf8_hungarian_ci;
```

#### 1.6.3.4 'rendelesek' tábla

A felhasználó megrendelt termékeinek adatait, illetve a felhasználó néhány adatát tartalmazza

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
<input type="checkbox"/>	1 id 	int(100)			Nem	Nincs		AUTO_INCREMENT
<input type="checkbox"/>	2 user_id	int(100)			Nem	Nincs		
<input type="checkbox"/>	3 nev	varchar(20)	utf8_hungarian_ci		Nem	Nincs		
<input type="checkbox"/>	4 telszam	varchar(20)	utf8_hungarian_ci		Nem	Nincs		
<input type="checkbox"/>	5 email	varchar(50)	utf8_hungarian_ci		Nem	Nincs		
<input type="checkbox"/>	6 modszer	varchar(50)	utf8_hungarian_ci		Nem	Nincs		
<input type="checkbox"/>	7 cim	varchar(500)	utf8_hungarian_ci		Nem	Nincs		
<input type="checkbox"/>	8 ossz_rendeles	varchar(1000)	utf8_hungarian_ci		Nem	Nincs		
<input type="checkbox"/>	9 ossz_ar	int(100)			Nem	Nincs		
<input type="checkbox"/>	10 datum	date			Nem	current_timestamp()		
<input type="checkbox"/>	11 fizetes	varchar(30)	utf8_hungarian_ci		Nem	fizetés nem történt meg		

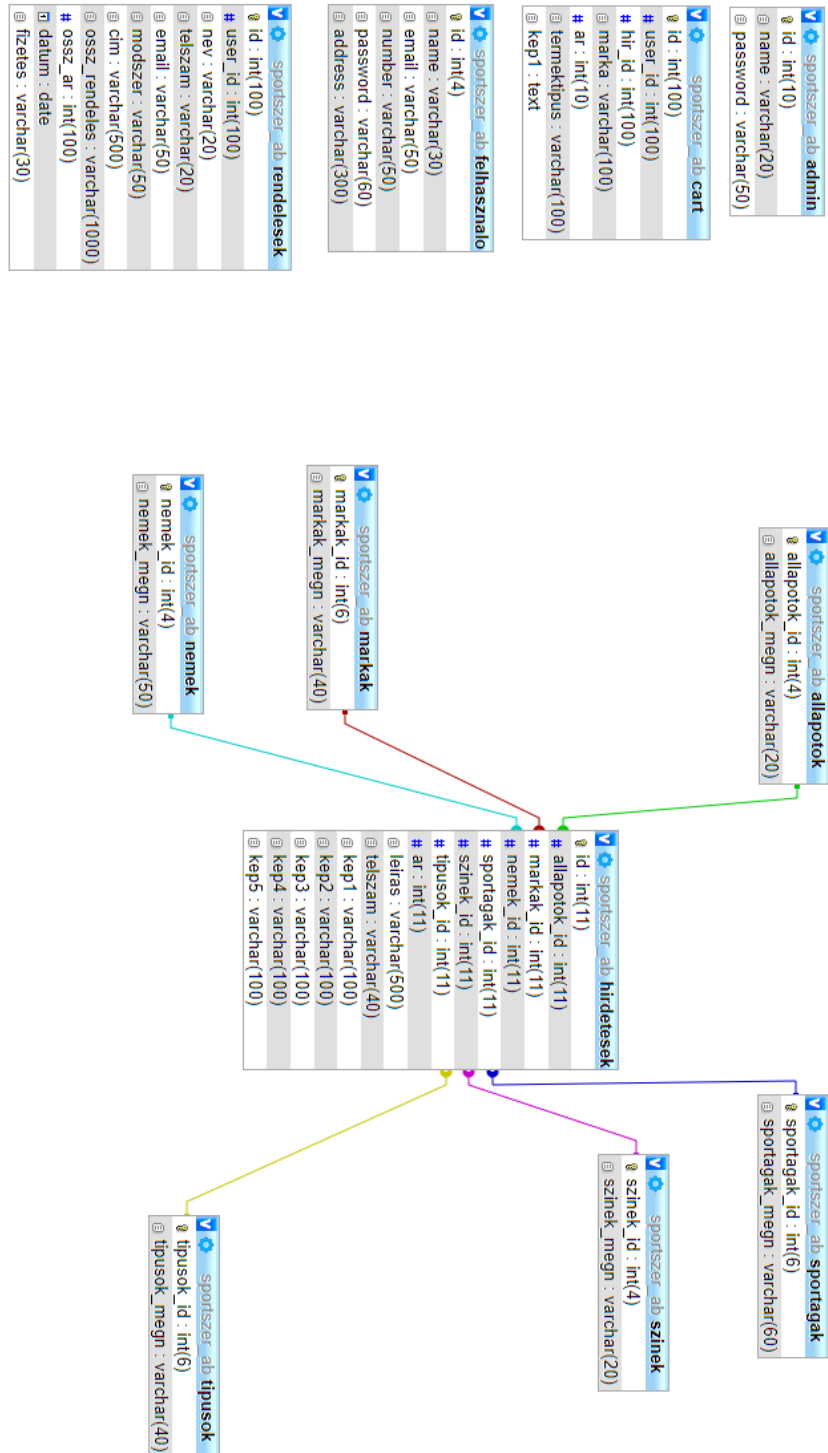
- id: a rendelés azonosítója
- user\_id: a rendelést leadó felhasználó azonosítója
- nev: a rendelést leadó felhasználó neve
- telszam: a rendelést leadó felhasználó telefonszáma
- email: a rendelést leadó felhasználó e-mail címe
- modszer: a rendelést leadó felhasználó fizetésének módja
- cim: a rendelést leadó felhasználó kiszállítási címe
- ossz\_rendeles: az rendelt termékek összegzése
- ossz\_ar: a teljes rendelés összege
- datum: a rendelés leadásának időpontja
- fizetes: a fizetés állapota

### Tábla létrehozása

```
CREATE TABLE IF NOT EXISTS `rendelesek` (  
  `id` int(100) NOT NULL,  
  `user_id` int(100) NOT NULL,  
  `nev` varchar(20) NOT NULL,  
  `telszam` varchar(20) NOT NULL,  
  `email` varchar(50) NOT NULL,  
  `modszer` varchar(50) NOT NULL,  
  `cim` varchar(500) NOT NULL,  
  `ossz_rendeles` varchar(1000) NOT NULL,  
  `ossz_ar` int(100) NOT NULL,  
  `datum` date NOT NULL DEFAULT current_timestamp(),  
  `fizetes` varchar(30) NOT NULL DEFAULT 'fizetés nem  
történt meg'  
  PRIMARY KEY (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungar  
ian_ci;  
COLLATE=utf8_hungarian_ci;) ENGINE=InnoDB DEFAULT  
CHARSET=utf8 COLLATE=utf8_hungarian_ci;
```

## 1.6.4 Adatbázis kapcsolatok

Az utólag hozzáadott táblák javítása a továbbfejlesztés része.



## 1.7 Modulok és kód ismertetése

### 1.7.1 Adatbázis csatlakozás

Az adatbázis csatlakozást a db\_csat.php fájlunkban valósítottam meg, ahol megadjuk a szükséges információkat (host, username, password) a csatlakozáshoz majd ezek által kapcsolódunk a hoston belül az adatbázishoz.

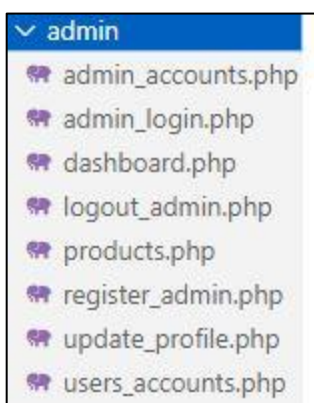


```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "sportszer_ab";

try {
    $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
} catch(PDOException $e) {
    echo "Sikertelen kapcsolódás: " . $e->getMessage();
}

?>
```

### 1.7.2 Admin felület



Úgy éreztük szükség van egy admin felületre is, hogy könnyebben tudjuk kezelni az oldalunkat és a felhasználóinkat.

Az admin felület egy saját css fájlt kapott így a stílus különbözik a fő oldalától.

Az admin felülethez tartozó fájljainkat az admin mappában találjuk egyből a törzskönyvtárból.

A *dashboard.php* adja meg a főképernyőjét az admin felületnek ahol különböző ablakokban érhetjük el a kívánt funkciókat.

Először is adatbázisból lekérdezzük a bejelentkezett admint és azt betöltjük a futó sessionbe.

```

include '../php/db_csat.php';

session_start();

$admin_id = $_SESSION['admin_id'];

if(!isset($admin_id)){
    header('location:admin_login.php');
}

$select_profile = $conn->prepare("SELECT * FROM `admin` WHERE id = ?");
    $select_profile->execute([$admin_id]);
    $fetch_profile = $select_profile->fetch(PDO::FETCH_ASSOC);
?>

```

Ezután a főoldalon létrehoztuk az ablakokat és a megfelelő lekérdezéseket az adatbázisból a kívánt adatok kezeléséhez.

```

<h3>Üdvözlöm</h3>
<p><?= $fetch_profile['name']; ?></p>
<a href="update_profile.php" class="btn">profil szerkesztése</a>
</div>

<div class="box">
    <?php
        $select_products = $conn->prepare("SELECT * FROM `hirdetesek`");
        $select_products->execute();
        $numbers_of_products = $select_products->rowCount();
    ?>
    <h3><?= $numbers_of_products; ?></h3>
    <p>Hirdetések</p>
    <a href="products.php" class="btn">Megnézés</a>
</div>

<div class="box">
    <?php
        $select_users = $conn->prepare("SELECT * FROM `felhasznalo`");
        $select_users->execute();
        $numbers_of_users = $select_users->rowCount();
    ?>
    <h3><?= $numbers_of_users; ?></h3>
    <p>Felhasználói fiókok</p>
    <a href="users_accounts.php" class="btn">Megnézés</a>
</div>

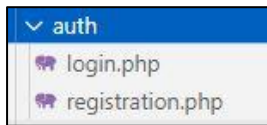
<div class="box">
    <?php
        $select_admins = $conn->prepare("SELECT * FROM `admin`");
        $select_admins->execute();
        $numbers_of_admins = $select_admins->rowCount();
    ?>
    <h3><?= $numbers_of_admins; ?></h3>
    <p>Adminok</p>
    <a href="admin_accounts.php" class="btn">Megnézés</a>
</div>

```



Először egy új php változóba bekérjük az adatokat az adatbázisunkból (pl \$select\_users) majd egy új változóba betöltjük ezeket (pl \$number\_of\_users) és egy rowCount() functionnel kiíratjuk az oldalunkra a meglévő adatok számát az oldalon.

### 1.7.3 Authentication



Az *auth* mappában található a *login* illetve a *register* php fájl ami a egy új felhasználó létrehozásáért vagy egy már meglévőbe való beléptetésért felel.

```
if(isset($_SESSION['user_id'])){
    $user_id = $_SESSION['user_id'];
    header('location:../profile.php');
}else{
    $user_id = '';
};
```

Először is ellenőrizzük a *login.php*-ban, hogy van-e a session-ünkben éppen aktív *user\_id*, ha van akkor az oldal átirányít a *profile.php* fájlba ahol a felhasználó elér a megadott adatait és változtathatja a kiszállítási címét.

```
if(isset($_POST['submit'])){
    $email = $_POST['email'];
    $email = filter_var($email, FILTER_SANITIZE_STRING);
    $pass = sha1($_POST['pass']);
    $pass = filter_var($pass, FILTER_SANITIZE_STRING);

    $select_user = $conn->prepare("SELECT * FROM `felhasznalo` WHERE email = ? AND password = ?");
    $select_user->execute([$email, $pass]);
    $row = $select_user->fetch(PDO::FETCH_ASSOC);

    if($select_user->rowCount() > 0){
        $_SESSION['user_id'] = $row['id'];
        header('location:../index.php');
    }else{
        $message[] = 'Helytelen email cím vagy jelszó!';
    }
}

if(isset($message)){
    foreach($message as $message){
        echo '
        <div class="message text-center">
        <span>'.$message.'</span>
        <i class="fas fa-times" onclick="this.parentElement.remove();"></i>
        </div>
        ';
    }
}
```

Miután a felhasználó kitöltötte a bejelentkezéshez megadott formot a *\$\_POST* szuper globális változó lekéri az email címet és a jelszót majd a *filter\_var* function levédi az sql lekérdezést a külső támadások ellen. Ezek után a *\$select\_user* változónkba betöltjük a lekért adatokat az adatbázisból, majd ha talált helyes párosítást akkor a felhasználó id-jét betölti a *user\_id*-be amit a session a továbbiakban használni fog és átirányít az *index.php*

fájltra ami a webshopunk főoldalát tartalmazza, viszont ha nem talál ilyen párosítást küld egy üzenetet a felhasználónak.

A *register.php* fájlban is megtörténik ugyan az az ellenőrzés ami a *login.php*-nál.

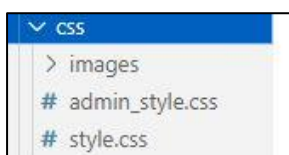
```
if(isset($_POST['submit'])){
    $name = $_POST['name'];
    $name = filter_var($name, FILTER_SANITIZE_STRING);
    $email = $_POST['email'];
    $email = filter_var($email, FILTER_SANITIZE_STRING);
    $number = $_POST['number'];
    $number = filter_var($number, FILTER_SANITIZE_STRING);
    $pass = sha1($_POST['pass']);
    $pass = filter_var($pass, FILTER_SANITIZE_STRING);
    $cpass = sha1($_POST['cpass']);
    $cpass = filter_var($cpass, FILTER_SANITIZE_STRING);

    $select_user = $conn->prepare("SELECT * FROM `felhasznalo` WHERE email = ? OR number = ?");
    $select_user->execute([$email, $number]);
    $row = $select_user->fetch(PDO::FETCH_ASSOC);

    if($select_user->rowCount() > 0){
        $message[] = 'email vagy telefonszám már létezik!';
    }else{
        if($pass != $cpass){
            $message[] = 'A jelszavak nem egyeznek!';
        }else{
            $insert_user = $conn->prepare("INSERT INTO `felhasznalo` (name, email, number, password) VALUES(?,?,?,?)");
            $insert_user->execute([$name, $email, $number, $cpass]);
            $select_user = $conn->prepare("SELECT * FROM `felhasznalo` WHERE email = ? AND password = ?");
            $select_user->execute([$email, $pass]);
            $row = $select_user->fetch(PDO::FETCH_ASSOC);
            if($select_user->rowCount() > 0){
                $_SESSION['user_id'] = $row['id'];
                header('location:../index.php');
            }
        }
    }
}
```

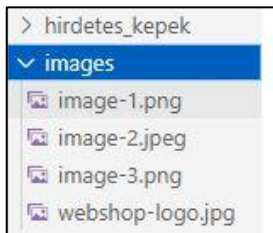
Szintűgy, ha a felhasználónk leadta a regisztrációs formot akkor a *\$\_POST* változónk begyűjti a megadott adatokat. Két jelszó változót adtunk meg (*\$pass* illetve *\$cpass*) a dupla jelszó megadása végett. A *\$select\_user* változó ellenőrzi, hogy új adatokat adnak-e meg, ha nem akkor hiba üzenetet kap a felhasználó. Abban az esetben, ha újak az adatok az 'INSERT INTO' sql paranccsal betöltjük a felhasználó táblába az új adatokat és az akkor kapott id-t hozzá rendeljük egy új user\_id-hez ami a sessionbe futni fog és a felhasználó átkerül a webshop főoldalára.

#### 1.7.4 CSS / Cascading Style Sheets



Ebben a mappában találhatóak a felhasznált ikon képek és a főoldalon felhasznált carosuel képei illetve maga a css stílus fájlok amik az oldalak stílusáért felelősek.

### 1.7.5 Képek



Van két mappánk *hirdetes\_kepek* illetve *images*. Az előbbi mappába kerülnek azok a képek amiket a felhasználók feltöltenek a hirdetésekhez, hogy később az oldal letudja kérni az egyes hirdetések megjelenítéséhez, az utóbbiba pedig a rendelést igazoló email fejléce és ikonjai találhatóak.

### 1.7.6 HTML kód nélküli funkciók

A 'php' nevezetű mappában találhatóak azok a php kódjaink amik nem tartalmazznak semmilyen más nyelvet.

#### 1.7.6.1 Phpmailer

Két darab email küldő kódunk van amit phpmailer segítségével oldottunk meg.

Először is meghívjuk a phpmailer beépített funkcióit,

```
use PHPMailer\PHPMailer\PHPMailer;  
use PHPMailer\PHPMailer\SMTP;  
use PHPMailer\PHPMailer\Exception;
```

```
if(isset($_SESSION['user_id'])){  
    $user_id = $_SESSION['user_id'];  
};  
  
if(isset($_POST['email'])) {  
    $email = filter_var($_POST['email'], FILTER_SANITIZE_STRING);  
};  
if(isset($_POST['name'])) {  
    $name = filter_var($_POST['name'], FILTER_SANITIZE_STRING);  
};  
if(isset($_POST['number'])) {  
    $number = filter_var($_POST['number'], FILTER_SANITIZE_STRING);  
};  
if(isset($_POST['msg'])) {  
    $msg = filter_var($_POST['msg'], FILTER_SANITIZE_STRING);  
};  
  
$mail = new PHPMailer(true);  
  
$mail = new \PHPMailer\PHPMailer\PHPMailer();  
$mail->isSMTP();  
$mail->Host = 'smtp.gmail.com';  
$mail->SMTPAuth = true;  
$mail->Username = 'hasznalt.sportszer@gmail.com';  
$mail->Password = 'oqoowozmhdjifxnj';  
$mail->SMTPSecure = PHPMailer::ENCRYPTION_SMTPS;  
$mail->Port = 465;  
$mail->CharSet = "UTF-8";  
  
$mail->setFrom('hasznalt.sportszer@gmail.com');  
$mail->addAddress('hasznalt.sportszer@gmail.com');
```

majd bekérjük a session-ünkből az éppen aktív felhasználót a user\_id-vel és lekérjük az id-hez rendelt email címet, nevet és telefonszámot illetve az üzenetet amit leírt a form-ba. Ezek után megadjuk az SMTP (Simple Mail Transfer Protocol) hostot ami a mi esetünkben a gmail beépített protokolja. Beállítjuk a bejelentkezési információkat (email cím illetve a generált alkalmazás jelszó) a biztonság érdekében, majd megadjuk a küldő illetve a befogadó email címet ami a segítségkérő üzenet esetében

mindkét cím a webshopunk címe, majd a bekért adatokat betöltjük az email 'body'-jába

és a felhasználót küldés után vissza irányítjuk a *segitse.php* fájlra ami a kontakt oldal kódját tartalmazza.

```
$mail->isHTML(true);  
$mail->Subject = 'Segítség fűlről érkezett üzenet';  
$mail->Body = "Név: $name <br> Telefonszám: $number <br> Email cím: $email <br> Üzenet: $msg";  
  
$mail->send();  
  
header('location:../segitse.php')
```

A rendelést igazoló emailt küldő kódunk eleje ugyan így épül fel azzal a különbséggel, hogy itt nem a webshopunk email címére kell küldenünk az emailt, hanem a felhasználónak aki éppen most rendelt akinek az email címét bekért *\$email* változóval adunk meg. ➡ `$mail->addAddress($email);`

Ezután mivel az emailünk tartalmaz képeket is a feljebb említett *images* mappából be kell kernünk őket hogy majd betudjuk illeszteni az emailünkbe.

➡

```
$mail->AddEmbeddedImage('images/image-1.png', 'my-facebook', 'image-1.png');  
$mail->AddEmbeddedImage('images/image-2.jpeg', 'my-logo', 'image-2.jpg');  
$mail->AddEmbeddedImage('images/image-3.png', 'my-github', 'image-3.png');  
  

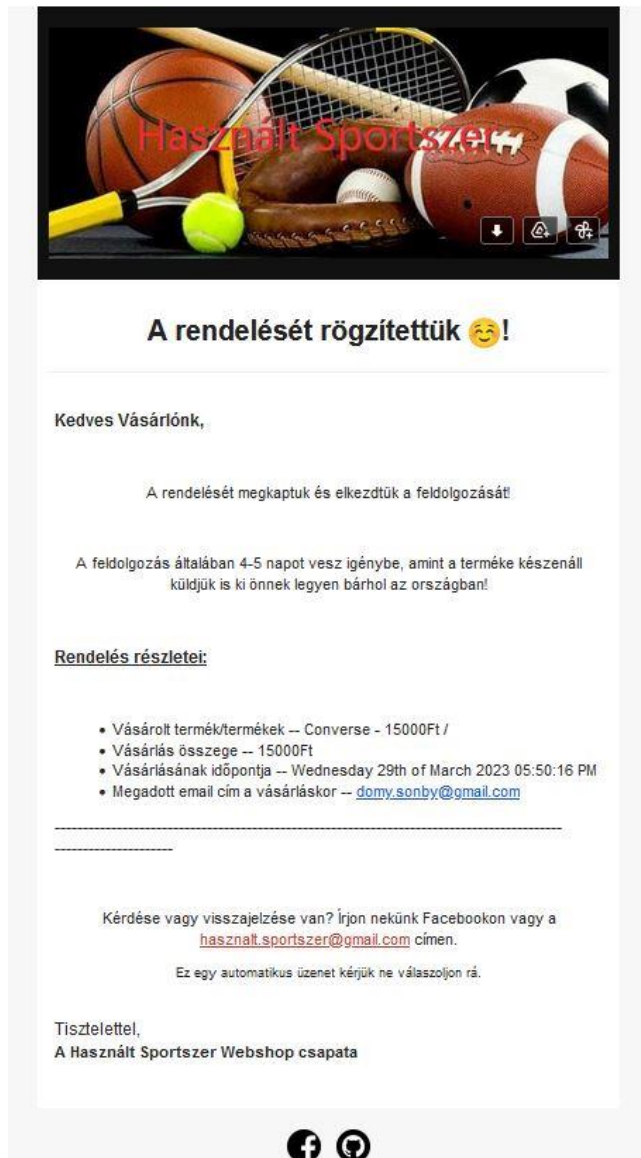
```

Minden képnek adnunk kell egy 'cid'-t ami alapján majd az email HTML kódjában meghívhatjuk. A képen példaként a Facebook ikon raktam aminek a 'my-facebook' 'cid'-t adtam.

Annak érdekében, hogy a kód automatikusan küldje ki ezt az emailt és mindig a megfelelő adatokat küldje ki meg kellett oldanunk, hogy magától kérje be az adatokat és küldje ki ezt a *\$\_POST* változóval értük el az alább látható módon.

```
>Vásárolt termék/termékek -- ' . $_POST['ossz_rendeles'] . '</li>  
>Vásárlás összege -- ' . $_POST['ossz_ar'] . 'Ft</li>  
>Vásárlásának időpontja -- ' . date("l jS \of F Y h:i:s A") . '</li>  
>Megadott email cím a vásárláskor -- ' . $_POST['email'] . '</li>
```

Az email maga pedig így kerül kiküldésre.

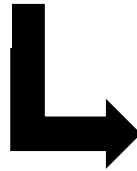


#### 1.7.6.2 Hirdetés megnyitása

A főoldalunkon a hirdetések nem részletesen jelennek meg ezért szükségünk volt egy oldalra amit a főoldalról megnyitva ér el a felhasználó ahol elér minden fontos információt az adott termékről.



Hirdetés előnézet



Megnyitott hirdetés

Mindenek előtt létrehoztunk egy új adatbázis csatlakozást csak ebben az esetben a *mysqli* módszerrel a *PDO* helyett amit eddigiekben használtunk mert szükségünk volt a *mysqli\_query()* functionre, hogy a *hir\_id* által letudjuk kérdezi a megfelelő hirdetés adatait.



```

$host = 'localhost';
$username = 'root';
$password = '';
$dbname = 'sportszer_ab';

$conn = new mysqli($host, $username, $password, $dbname);

if ($conn->connect_error) {
    die("Nem sikerült csatlakozni az adatbázishoz! " . $conn->connect_error);
}
$hir_id = $_GET['hir_id'];

$query = "SELECT * FROM hirdetesek INNER JOIN allapotok ON allapotok.allapotok_id = hirdetesek.allapotok_id";
$result = mysqli_query($conn, $query);
$fetch_product = mysqli_fetch_assoc($result);

$conn->close();

```

Az SQL lekérdezésben a különböző táblák egybe ágyazásait ez a kód teszi lehetővé.



```

"SELECT * FROM hirdetesek INNER JOIN allapotok ON
allapotok.allapotok_id = hirdetesek.allapotok_id INNER JOIN
markak ON markak.markak_id = hirdetesek.markak_id INNER JOIN
tipusok ON tipusok.tipusok_id = hirdetesek.tipusok_id INNER JOIN
nemek ON nemek.nemek_id = hirdetesek.nemek_id INNER JOIN szinek
ON szinek.szinek_id = hirdetesek.szinek_id INNER JOIN sportagak
ON sportagak.sportagak_id = hirdetesek.sportagak_id WHERE id =
$hir_id";

```

Ezek után a HTML kódban a *\$fetch\_product* metódussal iratjuk ki az oldalra a lekért adatokat ezen a módon.

→ `<?= $fetch_product['szinek_megn']; ?>`

### 1.7.6.3 Kosárba helyezés

Először is ismét ellenőrizzük, hogy van-e a sessionben 'user\_id', ha nincs az oldal átirányít a *login.php*-ra, ha van akkor behelyezzük a 'cart' nevű táblába azokat az adatokat amik a megfelelő 'hir\_id'-hez tartoznak ha a felhasználó lenyomta az 'add\_to\_cart' nevű gombot az *index.php*-ban található formban, majd az *\$insert\_cart* nevű változó lefutattja az 'INSERT INTO' SQL parancsot és így a *bevasarlokocsi.php* letudja kérni belőle az adatokat, hogy bekerüljenek a kosárba.

```

session_start();
require_once('db_csat.php');
$user_id = $_SESSION['user_id'];

if(isset($_POST['add_to_cart'])){

    if($user_id == ''){
        header('location:../auth/login.php');
    }else{

        $hir_id = $_POST['hir_id'];
        $hir_id = filter_var($hir_id, FILTER_SANITIZE_STRING);
        $termektipus = $_POST['termektipus'];
        $termektipus = filter_var($termektipus, FILTER_SANITIZE_STRING);
        $marka = $_POST['marka'];
        $marka = filter_var($marka, FILTER_SANITIZE_STRING);
        $ar = $_POST['ar'];
        $ar = filter_var($ar, FILTER_SANITIZE_STRING);
        $kep1 = $_POST['kep1'];
        $kep1 = filter_var($kep1, FILTER_SANITIZE_STRING);

        $insert_cart = $conn->prepare("INSERT INTO `cart` (user_id, hir_id, termektipus, marka, ar, kep1) VALUES ($user_id, $hir_id, $termektipus, $marka, $ar, $kep1)");
        $insert_cart->execute([$user_id, $hir_id, $termektipus, $marka, $ar, $kep1]);
        $message[] = 'Kosárhoz hozzáadva!';

    }

}

if(isset($message)){
    foreach($message as $message){
        echo '
        <div class="message text-center">
        <span>'.$message.'</span>
        <i class="fas fa-times" onclick="this.parentElement.remove();"></i>
        </div>
        ';
    }
}

```

#### 1.7.6.4 Kijelentkezés

A *profil\_kijelentkezés.php* nevű fájlunk a *session\_unset()* nevű functionnel kitöröl minden éppen tárolt adatot a sessionből és leállítja, majd átirányítja a felhasználót a *login.php* nevű fájlra a *header()* functionnel.

```

<?php

include 'db_csat.php';

session_start();
session_unset();
session_destroy();

header('location:../profile.php');

?>

```



### 1.7.7 HTML oldalak/funkciók

Ezeknél az oldalaknál felosztottuk a munkát. Balogh Ádám foglalkozott a HTML illetve a CSS kóddal, Schönberger Dominik pedig a php-val.

#### 1.7.7.1 Bevásárlókocsi

A 'bevásárlókocsi' mint funkció elengedhetetlen egy webshopnál így sokat foglalkoztam vele, hogy megfelelően működjön. Először is az első funkció amit létre akartam hozni az az, hogy ha a felhasználó több terméket is a kosárba helyezett, de nem feltétlen akar megválni az összestől ezt megtehesse, ezért két gomb található az oldalon ahol egyenként lehet terméket törölni vagy akár az egészet egyszerre. Ezt úgy értem el, hogy egy termék törlésénél a kosárba helyezésnél kapott id-re szűrtem rá míg a teljes kosárnál pedig az éppen aktív user\_id-re.

```
if(isset($_POST['delete'])){
    $cart_id = $_POST['cart_id'];
    $delete_cart_item = $conn->prepare("DELETE FROM `cart` WHERE id = ?");
    $delete_cart_item->execute([$cart_id]);
    $message[] = 'Termék törölve a kosárból!';
}

if(isset($_POST['delete_all'])){
    $delete_cart_item = $conn->prepare("DELETE FROM `cart` WHERE user_id = ?");
    $delete_cart_item->execute([$user_id]);

    $message[] = 'Kosár kiürítve';
}
```

A kód belsejében csináltam még egy lekérdezést ami lehetővé teszi, hogy az oldalon megtudjuk jeleníteni a pontos adatait a kosár tartalmának.



```
$select_cart = $conn->prepare("SELECT * FROM `cart` WHERE user_id = ?");
$select_cart->execute([$user_id]);
if($select_cart->rowCount() > 0){
    while($fetch_cart = $select_cart->fetch(PDO::FETCH_ASSOC)){
```

#### 1.7.7.2 Fizetés

Mindenek előtt bekérjük a `$_POST`-al az adatokat, hogy feltudjuk töltetni a 'rendelesek' táblánkba és lekérjük a megfelelő kosár tartalmát is a 'user\_id'-vel.

```
$nev = $_POST['nev'];
$nev = filter_var($nev, FILTER_SANITIZE_STRING);
$telszam = $_POST['telszam'];
$telszam = filter_var($telszam, FILTER_SANITIZE_STRING);
$email = $_POST['email'];
$email = filter_var($email, FILTER_SANITIZE_STRING);
$modszerek = $_POST['modszerek'];
$modszerek = filter_var($modszerek, FILTER_SANITIZE_STRING);
$cim = $_POST['cim'];
$cim = filter_var($cim, FILTER_SANITIZE_STRING);
$ossz_rendeles = $_POST['ossz_rendeles'];
$ossz_ar = $_POST['ossz_ar'];

$check_cart = $conn->prepare("SELECT * FROM `cart` WHERE user_id = ?");
$check_cart->execute([$user_id]);
```

Kényelmi szempontból megakartam csinálni, hogy miután a rendelés leadásra került és bekerült a 'rendelesek' táblába a kosár magától kiürüljön ehhez a már a *bevasarlokocsi.php*-ban látható módon 'user\_id' szerint törölök mindent a kosárból az 'INSERT INTO' SQL parancs után.

```
$insert_order = $conn->prepare("INSERT INTO `rendelesek` (user_id, nev, telszam, email, modszerek, cim, ossz_rendeles, ossz_ar)
$insert_order->execute([$user_id, $nev, $telszam, $email, $modszerek, $cim, $ossz_rendeles, $ossz_ar]);

$delete_cart = $conn->prepare("DELETE FROM `cart` WHERE user_id = ?");
$delete_cart->execute([$user_id]);
```

Később létrehoztam két változót amik a `$grand_total` illetve a `$select_cart`. Az előbbi a kosárban található termékek árának az összegét tárolja el, míg az utóbbi bekéri a 'cart' táblából az adatokat amiket majd később a `$fetch_cart` változóval kiiratunk az oldalra.

```
$grand_total = 0;
$cart_items = array();
$select_cart = $conn->prepare("SELECT * FROM `cart` WHERE user_id = ?");
$select_cart->execute([$user_id]);
if($select_cart->rowCount() > 0){
    while($fetch_cart = $select_cart->fetch(PDO::FETCH_ASSOC)){
        $cart_items[] = $fetch_cart['marka'].' - '.$fetch_cart['ar'].' Ft / ' ;
        $total_products = implode($cart_items);
        $grand_total += ($fetch_cart['ar']);
    }
}
```

### 1.7.7.3 Profil műveletek

A felhasználói profil adatait és a kiszállítási címet lehet változtatni, ha esetleg más email címet kezdett volna el használni a felhasználó vagy ha éppen elköltözött.

Először is ellenőrizzük, hogy az új adatok nem egyeznek-e már egy meglévővel a 'felhasznalo' táblában, ha erre nem kerül sor akkor egy 'UPDATE' paranccsal frissítjük a felhasználó adatait.

```
if(!empty($name)){
    $update_name = $conn->prepare("UPDATE `felhasznalo` SET name = ? WHERE id = ?");
    $update_name->execute([$name, $user_id]);
}

if(!empty($email)){
    $select_email = $conn->prepare("SELECT * FROM `felhasznalo` WHERE email = ?");
    $select_email->execute([$email]);
    if($select_email->rowCount() > 0){
        $message[] = 'Ezt az email címet már használják!';
    }else{
        $update_email = $conn->prepare("UPDATE `felhasznalo` SET email = ? WHERE id = ?");
        $update_email->execute([$email, $user_id]);
    }
}

if(!empty($number)){
    $select_number = $conn->prepare("SELECT * FROM `felhasznalo` WHERE number = ?");
    $select_number->execute([$number]);
    if($select_number->rowCount() > 0){
        $message[] = 'Ezt a telefonszámot már használják!';
    }else{
        $update_number = $conn->prepare("UPDATE `felhasznalo` SET number = ? WHERE id = ?");
        $update_number->execute([$number, $user_id]);
    }
}
```

A jelszavak változtatása egy kicsit komplikáltabb folyamat. Először bekérjük a jelszavakt a formból majd ellenőrizzük, hogy a felhasználó helyesen írta-e be a jelszavát mind a két alkalommal, hogy csak utána lehessen jelszót változtatni.

```
$empty_pass = 'da39a3ee5e6b4b0d3255bfef95601890afd80709';
$select_prev_pass = $conn->prepare("SELECT password FROM `felhasznalo` WHERE id = ?");
$select_prev_pass->execute([$user_id]);
$fetch_prev_pass = $select_prev_pass->fetch(PDO::FETCH_ASSOC);
$prev_pass = $fetch_prev_pass['password'];
$old_pass = sha1($_POST['old_pass']);
$old_pass = filter_var($old_pass, FILTER_SANITIZE_STRING);
$new_pass = sha1($_POST['new_pass']);
$new_pass = filter_var($new_pass, FILTER_SANITIZE_STRING);
$confirm_pass = sha1($_POST['confirm_pass']);
$confirm_pass = filter_var($confirm_pass, FILTER_SANITIZE_STRING);
```

```

if($old_pass != $empty_pass){
    if($old_pass != $prev_pass){
        $message[] = 'Helytelen jelszó!';
    }elseif($new_pass != $confirm_pass){
        $message[] = 'Az új jelszavak nem egyeznek!';
    }else{
        if($new_pass != $empty_pass){
            $update_pass = $conn->prepare("UPDATE `felhasznalo` SET password = ? WHERE id = ?");
            $update_pass->execute([$confirm_pass, $user_id]);
            $message[] = 'Sikeresen megváltoztatta a jelszavát!';
        }else{
            $message[] = 'Kérem írja be az új jelszavát!';
        }
    }
}

```

A kiszállítási cím esetén először `$_POST`-al bekérjük az adatokat, majd egy 'UPDATE' paranccsal megváltoztatjuk azt a 'felhasznalo' táblánkban.



```

$address = $_POST['ország'] . ', ' . $_POST['megye'] . ', ' . $_POST['írszám'] . ', ' . $_POST['település'] . ', ' . $_POST['utca'] . ', ' . $_POST['hazszam'];
$address = filter_var($address, FILTER_SANITIZE_STRING);

$update_address = $conn->prepare("UPDATE `felhasznalo` set address = ? WHERE id = ?");
$update_address->execute([$address, $user_id]);

$message[] = 'Cím elmentve!';

```

#### 1.7.7.4 Profil megjelenítés

Mindenekelőtt csináltam egy lekérdezést az adatbázisból ami a `$fetch_profile` változóba tölti az éppen bejelentkezett felhasználót.

```

$select_profile = $conn->prepare("SELECT * FROM `felhasznalo` WHERE id = ?");
$select_profile->execute([$user_id]);
if($select_profile->rowCount() > 0){
    $fetch_profile = $select_profile->fetch(PDO::FETCH_ASSOC);
};

```

Majd ezt a kódot felhasználva kiiradjuk a felhasználó adatokat.

```

<td>Név</td>
<td class="text-dark fw-bold"><?= $fetch_profile['name']; ?></td>

<td>E-mail</td>
<td class="text-dark fw-bold"><?= $fetch_profile['email']; ?></td>

<td>Telefonszám</td>
<td class="text-dark fw-bold"><?= $fetch_profile['number']; ?></td>

<td>Kiszállítási cím</td>
<td class="text-dark fw-bold"><?= $fetch_profile['address']; ?></td>

```



Ez a kód részlet így látható a frontenden.



ADATAIM	
Név	Dominik
E-mail	domy.sonby@gmail.com
Telefonszám	06202300067
Kiszállítási cím	Magyarország, Pest, 2022, Tahitótfalu, Erdész utca 14,

#### 1.7.7.5 Kontakt

A kontakt oldal a *segitseg.php* fájlban található amibe tettem egy formot ahova a felhasználó leírhatja a panaszát vagy a visszajelzését és a *segitseg\_uzenet.php*-t meghívva emailt küld a webshop email címére, hogy tudjunk rá reagálni.

```
<h3 class="text-center h1 fw-bold mb-5 mx-1 mx-md-4 mt-4">Hagyjon nekünk üzenetet!</h3>
<!-- ***** Név ***** -->
<form class="mx-1 mx-md-4" action="php/segitseg_uzenet.php" method="post">
  <div class="d-flex flex-row align-items-center mb-4">
    <div class="form-outline flex-fill mb-0">
      <input type="text" name="name" placeholder="Név" class="form-control" value="<?= $fetch_profile['name']; ?>">
    </div>
  </div>
</form>
```

## 1.8 Tesztelés/Tesztesetek<sup>12</sup>

A tesztelés rendkívül fontos egy program készítése közben, hiszen minden munkában fordulhatnak elő hibák. Ez a programozás esetében még inkább elmondható hiszen ebben az esetben akár egy hiányzó pontos vessző is okozhat nagy hibákat. A mi programunkon vizuális tesztet alkalmaztunk ami annyit jelent, hogy amint lekódoltunk egy adott funkciót vagy oldalt végig nyomkodtuk, hogy biztosak legyünk benne, hogy megfelelően működik mielőtt tovább haladnánk.

### Tesztelési alapelvek:

1. A tesztelés hibák jelenlétét jelzi: A tesztelés képes felfedni a hibákat, de azt nem, hogy nincs hiba. Ugyanakkor a szoftver minőségét és megbízhatóságát növeli.
2. Nem lehetséges kimerítő teszt: Minden bemeneti kombinációt nem lehet letesztelni (csak egy 10 hosszú karakterláncnak  $256^{10}$  lehetséges értéke van) és nem is érdemes. Általában csak a magas kockázatú és magas prioritású részeket teszteljük.
3. Korai teszt: Érdemes a tesztelést az életciklus minél korábbi szakaszában elkezdni, mert minél hamar találunk meg egy hibát (mondjuk a specifikációban), annál olcsóbb javítani. Ez azt is jelenti, hogy nemcsak programot, hanem dokumentumokat is lehet tesztelni.
4. Hibák csoportosulása: A tesztelésre csak véges időnk van, ezért a tesztelést azokra a modulokra kell koncentrálni, ahol a hibák a legvalószínűbbek, illetve azokra a bemenetekre kell tesztelnünk, amelyre valószínűleg hibás a szoftver (pl. szélsőértékek).
5. A féregirtó paradoxon: Ha az újrateesztelés során (lásd később a regressziós tesztet) mindig ugyanazokat a teszteseteket futtatjuk, akkor egy idő után ezek már nem találnak több hibát (mintha a férgek alkalmazkodnának a teszthez). Ezért a tesztjeinket néha bővíteni kell.
6. A tesztelés függ a körülményektől: Másképp tesztelünk egy atomerőműnek szánt programot és egy beadandót. Másképp tesztelünk, ha a tesztre 10 napunk vagy csak egy éjszakánk van.

---

<sup>12</sup>

[https://www.google.com/url?sa=i&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=0CAMQw7AJahcKEwjAroCtx4P-AhUAAAAAHQAAAAAQAw&url=https%3A%2F%2Faries.ektf.hu%2F~gkusper%2FSzoftverTeszteles.pdf&psig=AOvVaw00b\\_KIZtBsVBJpon2ZoHg4&ust=1680262397248436](https://www.google.com/url?sa=i&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=0CAMQw7AJahcKEwjAroCtx4P-AhUAAAAAHQAAAAAQAw&url=https%3A%2F%2Faries.ektf.hu%2F~gkusper%2FSzoftverTeszteles.pdf&psig=AOvVaw00b_KIZtBsVBJpon2ZoHg4&ust=1680262397248436)

7. A hibátlan rendszer téveszméje: Hiába javítjuk ki a hibákat a szoftverben, azzal nem lesz elégedett a megrendelő, ha nem felel meg az igényeinek. Azaz használhatatlan szoftvert nem érdemes tesztelni.

### ***Tesztelési technikák***

1. Feketedobozos (black-box) vagy specifikáció alapú, amikor a specifikáció alapján készülnek a tesztesetek.
2. Fehéredobozos (white-box) vagy strukturális teszt, amikor a forráskód alapján készülnek a tesztesetek.

Ezen csoportosítások az információknak mi szerint a tesztet lefuttatják.

#### 1.8.1 Tesztesetek

Egy webshop esetében leginkább a profil műveleteknél kaphat a felhasználó hibaüzenetet, így ezt teszteltem leginkább.

Itt látható, ha a felhasználó helytelen adatokat ad meg a bejelentkezés illetve regisztráció során



Emellett leellenőriztem, hogy a profil módosításnál is működik-e a kód.



**Ezt az email címet már használják!**

**Helytelen jelszó!**

[KONTAKT](#) [KOSARAM](#) [REGISZTRÁLÁS](#) [HÍRDETE](#)

## 1.9 Továbbfejlesztési lehetőségek

- Amivel mindenképp foglalkozni szeretnénk, hogy REST API-t használjunk a procedurális felépítés helyett, de mivel ketten maradtunk sajnos erre nem maradt kapacitásunk.
- A webshop hostolása, hogy ne csak localhoston keresztül lehessen elérni.
- A session helyett sütik alkalmazása.
- Befejezni a szűrést a főoldalon.
- Logo készítése.
- Saját hirdetéseim megtekintése, módosítása, törlése.
- Adatbázis tökéletesítése (pl.: kapcsolatok).
- 'Elfelejtett jelszó' funkció.
- 'Emlékezzen rám' funkció a bejelentkezésnél.



## 2 Felhasználói dokumentáció

### 2.1 Alkalmazás rövid ismertetése

Webes alkalmazásunk egy webáruház, melynek célközönsége a sportolók. Oldalunkon sportoló, illetve sportolni vágyó emberek tudnak kedvükre böngészni a különböző sporteszközök között. De nem csak böngészésre, vásárlásra alkalmas a webáruház, hanem saját termékeinket is eladásra tudjuk kínálni. Az elérhető hirdetések megtekintésén kívül, az oldal többi funkciója regisztrációhoz, bejelentkezéshez kötött. Vásárlóink kosárba tudják helyezni a megvásárolni kívánt termékeket, majd a fizetés gombra kattintás után, a fizetési adatok megadásával le tudják adni a rendeléseiket. A sikeres rendelés után mi gondoskodunk az eladó és a vevő kapcsolattartásáról, valamint a termékek kiszállításának teljes koordinációjáról.

### 2.2 Rendszerkövetelmények

Szerencsére egy webshop futtatásához nincs szükségünk NASA szintű szuper perifériához. Mivel a webshopunk reszponzív, működéséhez tökéletesen elég egy olyan számítógép, laptop, tablet vagy telefon amin van internet hozzáférés és található rajta böngésző. Mi a Mozilla Firefoxot vagy a Chrome-ot ajánljuk a webshop használatához.

### 2.3 Alkalmazás telepítése, szükséges beállítások

Webáruházunk egyelőre csak lokális szerveren érhető el, ezért a következő lépések elengedhetetlenek a webáruház eléréséhez

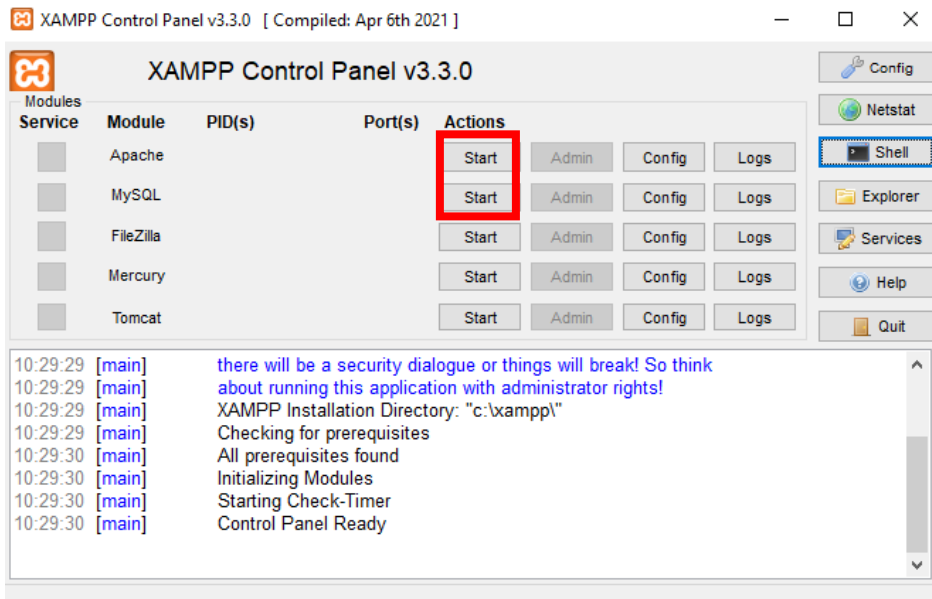
#### 2.3.1 XAMPP telepítése

Válasszuk ki a linkre kattintva az operációs rendszernek megfelelőt, majd kattintson a „Letöltés (64bit)” gombra és várjuk meg a letöltés befejezését. A „C” meghajtóra mentünk. (<https://www.apachefriends.org/hu/download.html>)

Ezután kattintsunk az exe fájlra, és engedélyezzük, hogy módosításokat hajtson végre a számítógépünkön. Innentől már csak a tovább gombokra kell kattintani.

### 2.3.2 XAMPP elindítása

Indítsuk el az Apache és MySQL szervereket a Start gombokra kattintva.



Az elindítás sikeres, hogyha az Apache és MySQL feliratok háttére zöldre változik, illetve a gombok felirata Stop-ra módosul.

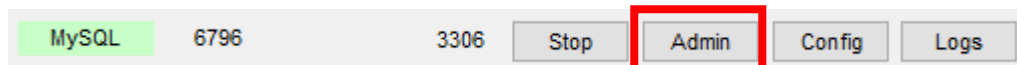
Apache	16060 6240	80, 443	Stop	Admin	Config	Logs
MySQL	6796	3306	Stop	Admin	Config	Logs

### 2.3.3 A fájlok bemásolása a htdocs mappába

Helyezzük az adathordozóról vagy a GitHub-ról a HasznaltSportszer nevű mappát a **XAMPP** ➡ **htdocs** mappájába.

### 2.3.4 Adatbázis futtatása

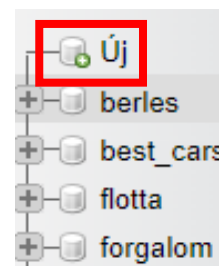
Nyissuk meg a XAMPP-ot, majd kattintsunk a MySQL admin gombjára.



Ekkor az adatbázis kezelőben találhatjuk magunkat (localhost/phpmyadmin), majd az oldal bal oldalán kattintsunk az új adatbázis lehetőségre.

Ezután 2 lehetőségünk van:

1. Felül kiválasztjuk az importálás menüpontot, majd a fájl kiválasztásánál kiválasztjuk a *sportszer\_ab.sql* fájlt, végül az importálás gombra kattintunk.
2. Kimásoljuk *sportszer\_ab.sql* fájl teljes tartalmát, majd felül kiválasztjuk az SQL menüpontot és beillesztjük a kimásolt szöveget, és az indítás gombra kattintunk.



### 2.3.5 Webáruház megnyitása

Ha mindennel megvagyunk, akkor a böngésző keresőjébe írjuk be a következőt: *localhost:8080/HasznaltSportszer*. Ekkor az alkalmazás főoldalán találhatjuk magunkat.

## 2.4 Alkalmazás használata

### 2.4.1 Navigáció, lábjegyzet

#### 2.4.1.1 Navigációs sáv



Az összes oldal tetején található egy navigációs rész az oldal könnyed használhatóságának érdekében. A sávban az éppen aktív oldal gombjának a háttérszíne sötétebb a többitől, illetve nem is tudunk rákattintani. Továbbá a bal oldali *VISSZA* gomb megnyomásával a legutóbb látogatott oldalra tudunk ugrani.

#### 2.4.1.2 Lábjegyzet

Minden oldal alján található egy szürke háttérű lábjegyzet rész, amely a webáruház fejlesztőinek az elérési módját, és az ajándék vásárlásra ösztönzést foglalja magába.

ELÉRHETŐSÉGEK	AJÁNDÉK
Bármilyen segítségre van szüksége, kérdése van, keressen fel minket e-mailben (hasznalt.sportszer@gmail.com) vagy akár Facebook oldalunkon. Használja üzenetküldő rendszerünk a <a href="#">kontakt</a> fül alatt.	Lepje meg családját, barátját vagy önmagát áron aluli kincsekkel. Önnek csak házhöz kell rendelnie a terméket, a szállítást már mi intézzük.

#### 2.4.2 Regisztráció, Bejelentkezés

Webáruhazunkban, hogyha a felhasználó még nem regisztrált vagy csak nincs még belépve, akkor a bejelentkezési és regisztrációs oldalakon kívül csak a főoldalt tudja megnyitni, ha más oldalra szeretne lépni, akkor mindig a beléptetési oldalra fogja dobni a felhasználót az oldal.

##### 2.4.2.1 Regisztráció

Oldalunk teljeskörű használatához regisztráció szükséges. Ezt pár perc alatt kivitelezhetjük is, nincs más teendőnk, mint néhány adatot megadni.

The screenshot shows a registration form with the title 'Regisztráció' in large white letters on a dark background. Below the title, the text 'GYORS, EGYSZERŰ REGISZTRÁCIÓ' is displayed. The form contains several input fields: 'Vezetéknév Keresztnév' (Name), 'E-mail' (Email), 'Telefonszám' (Phone number), 'Jelszó' (Password), and 'Jelszó megerősítés' (Confirm password). Each field has a placeholder text. Below the fields is a 'Regisztrálok' button. At the bottom, there is a link 'MÁR VAN FIÓKJA? LÉPJEN BE'.

A rendszerbe egy e-mail címmel, illetve egy telefonszámmal csak egyszer lehet regisztrálni. A továbbfejlesztés része egy megerősítő e-mail vagy SMS küldése a regisztráció jóváhagyásához.

##### 2.4.2.2 Bejelentkezés

Ha már regisztráltunk, akkor a belépéshez nincs más dolgunk, mint a belépési oldalon megadni a regisztrációkor használt e-mail címet illetve az ehhez tartozó jelszót.

### 2.4.3 Főoldal

Főoldalunkon a navigációs rész alatt található webáruházunk neve. A továbbfejlesztés része lesz, hogy ezt a feliratot egy saját embléma, logó váltsa majd. Ezt követően 3 lapozható kép található, melyek önmaguktól is lapozzák magukat. A rajtuk található adatok természetesen csak kitaláltak, mivel oldalunk még nem üzemel. Ezen képek célja a reklám, illetve, hogy pozitív hatást keltsen a felhasználókban, hogy ezen az oldalon igenis érdemes eladni a termékeket.




Ez alatt a szűrési szekció található. Ez a funkció egyelőre még nem működőképes. Megvalósítása a csoportunk 3. tagja (Hajszter Botond) feladata lett volna és a pótlásra sem maradt már idő.

A screenshot of a web application's filtering section. The title is 'Szűrési lehetőségek' (Filtering options) in a bold, white font. Below the title is a subtitle in a smaller, white font: 'ÁLLÍTSON BE SZŰRŐKET A HATÉKONYABB KERESÉSÉRT!' (Set filters for more efficient searching!). The filtering options are organized into a grid of eight input fields, each with a label above it: 'SPORTÁG' (Sport type), 'MINIMUM Ft-tól:' (Minimum price from Ft), 'MAXIMUM Ft-tól:' (Maximum price from Ft), 'FÉRFI/NŐI' (Male/Female), 'MÁRKA' (Brand), 'SZÍN' (Color), 'ÁLLAPOT' (Condition), and 'TÍPUS' (Type). Each input field contains three asterisks '\*\*\*'. At the bottom of the grid is a large, dark button labeled 'Szűrés' (Filter).


Végül az elérhető hirdetések találhatóak.


### Elérhető Hírdetések



**15000 Ft**  
**Nike**  
CÍPŐ


Vadonatúj cipőm mérethiba miatt eladová vált.


[Bővebben...](#) 



**4000 Ft**  
**Adidas**  
PÓLÓ


Szeretett Real Madrid mezem eldóvá vált. Amióta kaptak a Barcelona-tól azóta sajnos már nem tudom büszkén hordani, ezért eladó

[Bővebben...](#) 



**6000 Ft**  
**Nike**  
KIEGÉSZÍTŐK

Fiam kinőtte kapuskesztyűjét, így eladásra kínálom. Irányár: 6000 Ft

[Bővebben...](#) 

Itt láthatjuk a termék árát, márkáját, típusát és leírását. Ez alatt található 2 gomb, a *Bővebben* gombra kattintva láthatjuk a további képeket a hirdetésről (lapozható képek), illetve a termék részletes adatait.

## Kiválasztott termék



**15000 Ft**

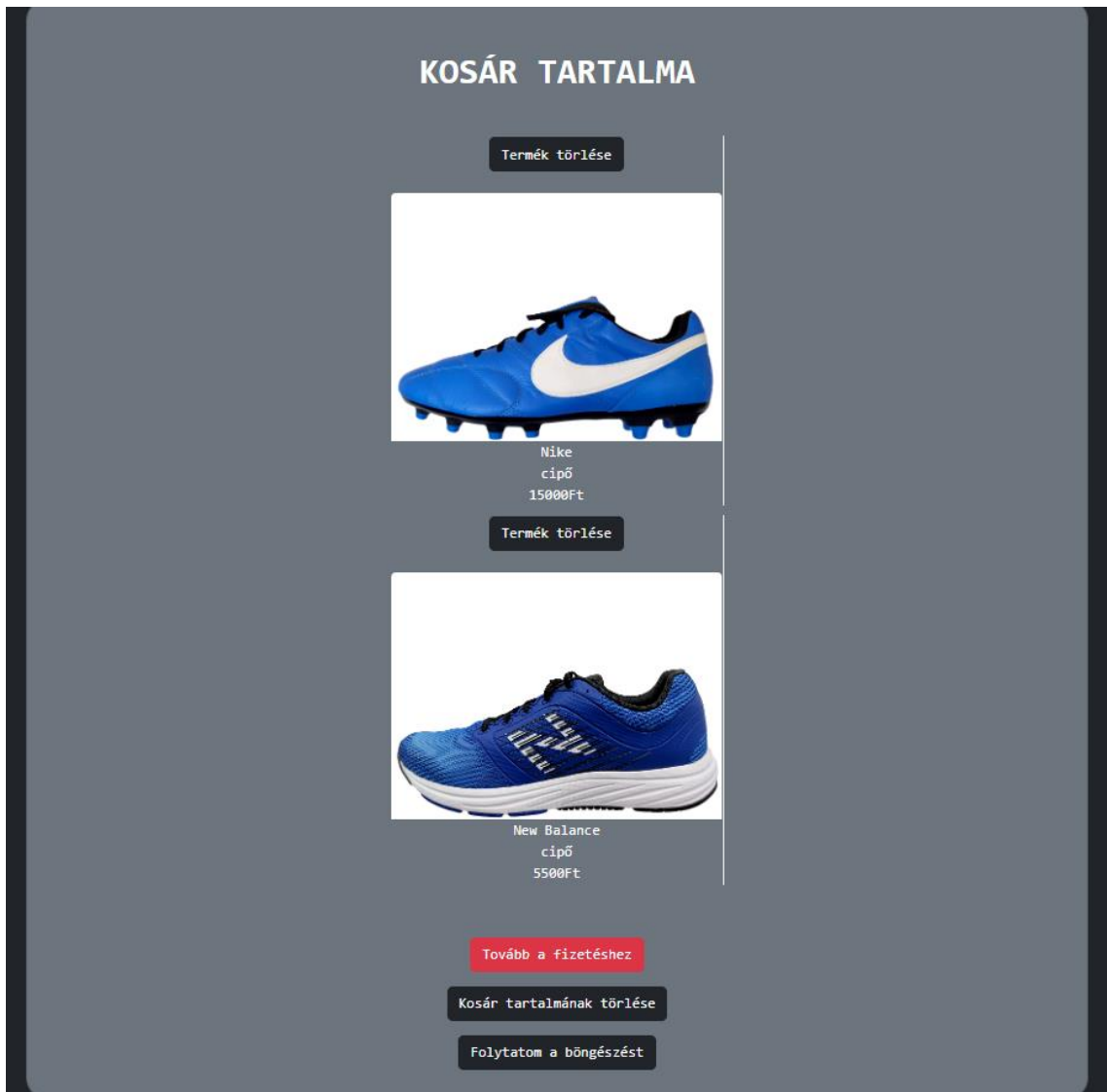
Márka:	Nike
Állapot:	új
Nem:	férfi
Sportág:	labdarúgás
Szín:	kék
Termék típusa:	cipő
<small>Vadonatúj cipőm mérethiba miatt eladová vált.</small>	



A *kosár* gombra kattintva a kosárba tudjuk helyezni a kiválasztott terméket.

#### 2.4.4 Kosaram

A kosaram oldalon láthatjuk a kosárba helyezett termékünk főbb adatait.



A *Termék törlése* gombra kattintva csak az adott termék kerül törlésre a kosárból. A *Kosár tartalmának törlése* gomb lenyomásával az egész kosarat kiürítjük. A *Folytatom a böngészést* gombbal a főoldalra tudunk visszamenni, hogy folytassuk a nézelődést.

*Tovább a fizetéshez* gomra kattintva egy új oldalra lépünk.:

## 2.4.6 Rendelés leadása

# Rendelés összegzés

## Rendelésem

### 1. TERMÉK

Márka:	Nike
Típus:	cipő
Ár:	15000 Ft

### 2. TERMÉK

Márka:	New Balance
Típus:	cipő
Ár:	5500 Ft

Rendelés összege: 20500Ft

Kosaram megtekintése, módosítása

## Adataim

Név:

Balogh Ádám

Telefonszám:

+36204977683

E-mail:

03adamba@gmail.com

Adatok megváltoztatása

## Kiszállítási cím

Magyarország, Pest megye, 2600, Vác, Akó utca, 13

Kiszállítási cím módosítása

Készpénz

Rendelés leadása

Vásárlás folytatása

Ezen az oldalon először is láthatjuk termékenként a márkát, a fajtát és az árat, illetve az összesített (fizetendő) árat. Alatta található gombbal módosíthatjuk a kiválasztott termékeinket. Ezután a személyes adatainkat láthatjuk, amit még itt is van módunk módosítani. Végül a kiszállítási cím és a fizetési mód megadása után leadhatjuk rendelésünk. A rendelés sikeres leadásáról visszaigazoló e-mailt kapunk a webáruháztól és a következő szöveg jelenik meg az oldal fejlécében.:

Rendelését sikeresen leadta, figyelje az e-mail fiókját!

A vásárlás folytatása gombra kattintva a főoldalra lépünk vissza.



### 2.4.7 Kontakt

A kontakt oldal célja, hogy az ügyfeleinket segíteni tudjuk bármilyen kérdés vagy panasz esetén. Az felhasználó e-mail címéről kapunk egy levelet, amely tartalmazza a személyes adatait, illetve az üzenetét. Ezután felvesszük a kapcsolatot az ügyféllel a megadott e-mail címen vagy telefonszámon.



The image shows a contact form titled "Elérhetőségeink" (Our Contact Information) in a large, bold, white font on a dark background. Below the title, the text "HAGYJON NEKÜNK ÜZENETET!" (Leave us a message!) is displayed in a smaller, bold, white font. The form consists of four input fields, each with a white border and a light gray background, stacked vertically. The first field contains the text "Balogh Ádám", the second contains "+36204977683", and the third contains "03adamba@gmail.com". The fourth field is a larger text area containing the word "Üzenet". Below the text area, there is a small, light gray text prompt "Mi az üzenete számunkra?" (What is your message for us?). At the bottom of the form, there is a dark gray button with the text "Üzenet elküldése" (Send message) in white.

A rendszer a személyes adatokat automatikusan kitölti, de ha az üzenet írója meg szeretne változtatni bármiféle adatot az ügyintézéshez, akkor természetesen megteheti. Ehhez csak egyszerűen kitörli a már beírt adatokat és beírja az általa használni kívánt adatokat.

## 2.4.8 Hirdetés feltöltése

A hirdetés feltöltéséhez az alább látható adatok megadása mind kötelező, képből elég csak egyet feltölteni, de ajánlott a több kép használata. Ahol legördülő menü található, ott az adatokat az adatbázisunkból hívja meg az oldal.

A sikeres feltöltés után hirdetésünk megjelenik a főoldalon.

### HÍRDETÉS FELTÖLTÉSE

...

Adja meg a márkát!

...

Adja meg a sportágat!

...

Adja meg a domináns szint!

...

Adja meg a termék típusát!

Ft

Adja meg az árat!

Rövid leírás

Adja meg a termék leírását!

...

Adja meg a nemet!

...

Adja meg a termék állapotát!

Fájl kiválasztása

Nincs fájl...iválasztva

1. kép

Fájl kiválasztása

Nincs fájl...iválasztva

2. kép

Fájl kiválasztása

Nincs fájl...iválasztva

3. kép

Fájl kiválasztása

Nincs fájl...iválasztva

4. kép

Fájl kiválasztása

Nincs fájl...iválasztva

5. kép

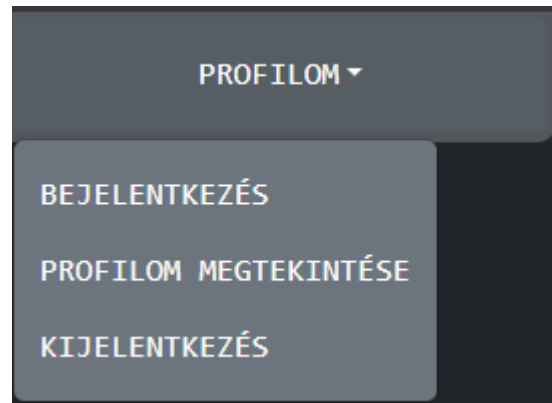
Feltöltés

## 2.4.9 Profilom

A profilom legördülő gombra kattintva az a bejelentkezésen kívül a következő lehetőségek közül választhatunk még.:

### 2.4.9.1 Profilom megtekintése

A profilom megtekintése fül alatt személyes adataink illetve a kiszállítási címünk található.



# Személyes profilom

## ADATAIM

Név	Balogh Ádám
E-mail	03adamba@gmail.com
Telefonszám	+36204977683
Kiszállítási cím	Magyarország, Pest megye, 2600, Vác, Akó utca, 13

Adatok módosításaCím módosítása

A megfelelő gombra kattintva lehetőségünk van megváltoztatni személyes adatainkat, jelszavunkat és a címünket.

### 2.4.9.2 Kijelentkezés

A kijelentkezés gombra kattintva kilépünk profilunkból.

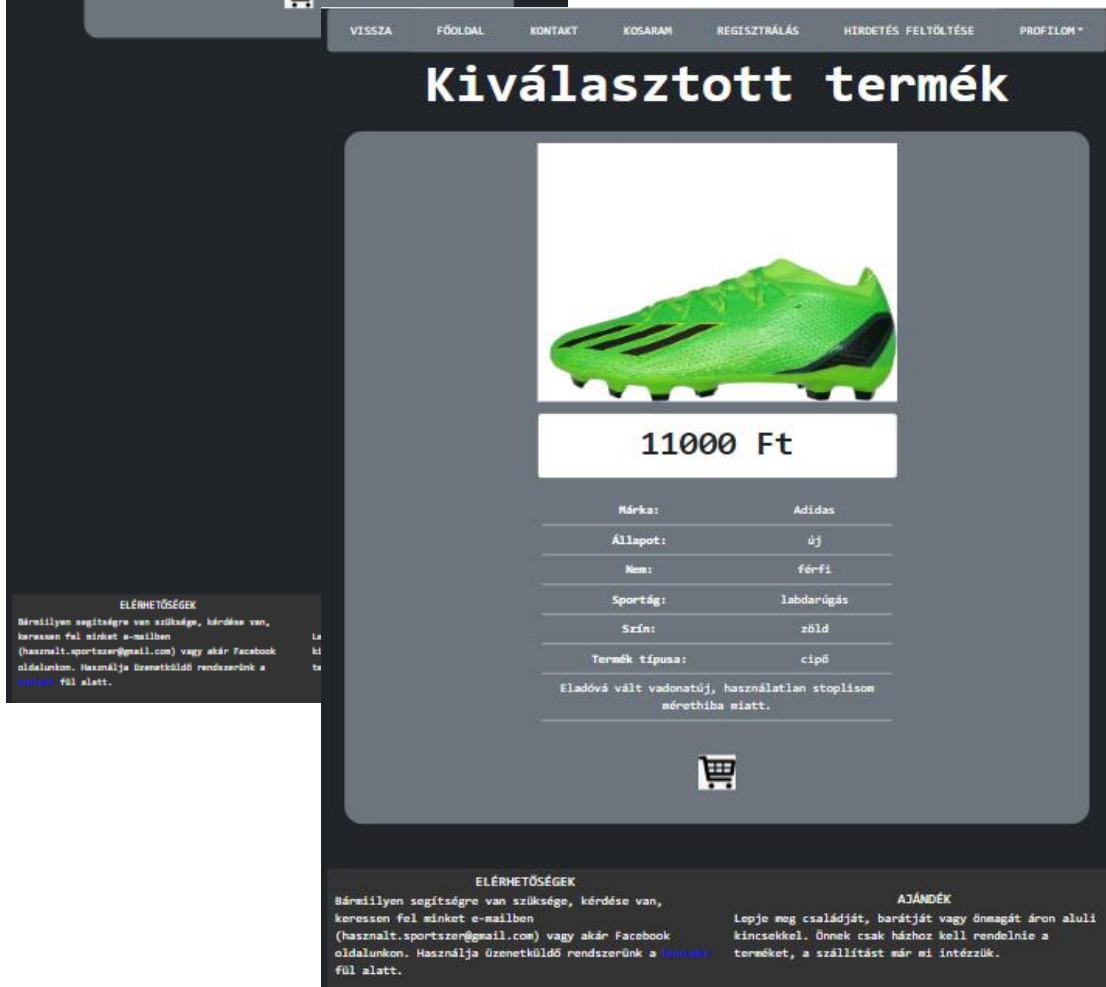
## 2.5 Felhasználható eszközök

### iPhone 12 Pro



A webáruház teljes mértékben reszponzív, tehát akár telefonon, akár tableten is jól használható, melyet a Bootstrap keretrendszerrel valósítottunk meg. Az alábbi képeken egy megnyitott hirdetés oldalát láthatjuk telefon és tablet nézetbe.

### iPad Pro



## 2.6 Admin felület

Ahogy azt már a fejlesztői dokumentációban említettem az admin felület kinézete különbözik a főoldalától. A kezdőképernyőn ablakok találhatók ahol a kívánt feladatainkat tudjuk ellátni.



### 2.6.1 Profil szerkesztés

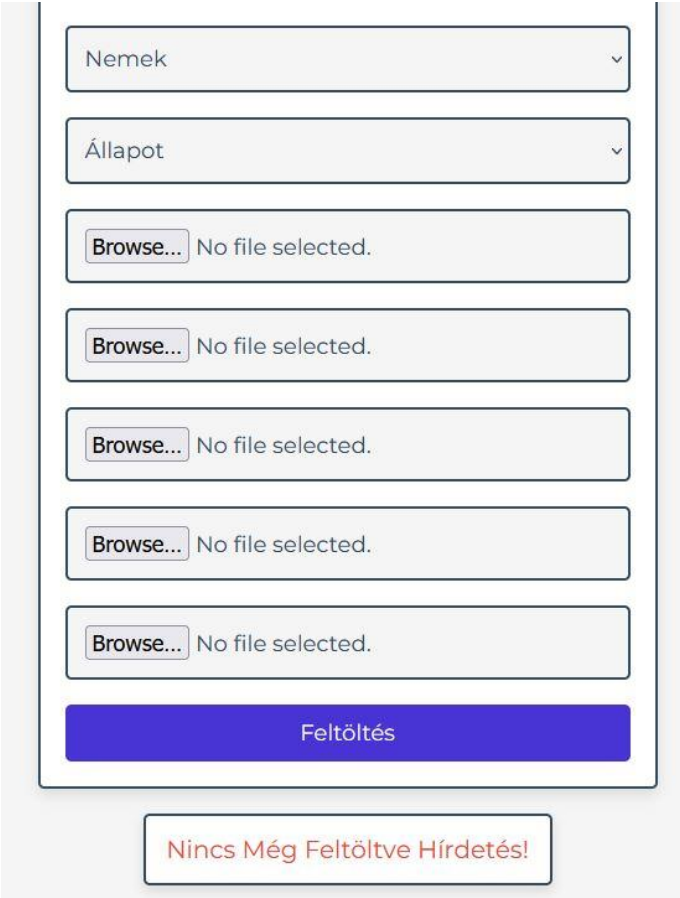
Ezen a fülön az éppen bejelentkezett adminnak változtathatjuk a jelszavát.

The image shows a form titled "Profil Szerkesztése". It contains four text input fields and one submit button:

- Text input field: "írja be a felhasználó nevét".
- Text input field: "írja be a jelszavát".
- Text input field: "írja be az új jelszavát".
- Text input field: "ismételje meg az új jelszavát".
- Submit button: "Frissítés".

### 2.6.2 Hirdetések

A hirdetéseknel adminként tölthetünk fel hirdetést vagy törölhetjük a már meglévőket, ha esetleg a bizonyos termék már elkelt.



The form is a vertical stack of elements. At the top is a dropdown menu labeled 'Nemek' with a downward arrow. Below it is another dropdown menu labeled 'Állapot' with a downward arrow. Then there are five identical rows, each containing a 'Browse...' button and the text 'No file selected.'. Below these rows is a large blue button labeled 'Feltöltés'. At the bottom of the form is a red-bordered box containing the text 'Nincs Még Feltöltve Hírdetés!' in red.

### 2.6.3 Felhasználói fiókok

Ezen a fülön tudjuk kezelni a már regisztrált felhasználóinkat.



The interface has a title 'Felhasználói Fiók' at the top. Below the title is a white box with a black border. Inside this box, the text 'user id : 23' is displayed in blue. Below that, the text 'username : Schönberger Dominik' is displayed in blue. At the bottom of the white box is a large red button labeled 'Törlés' in white.

## 2.6.4 Adminok kezelése

Ezen a fülön tudunk törölni illetve regisztrálni új adminokat, illetve itt is tudjuk frissíteni a már bejelentkezett fiókot.

### Admin Fiókok

Új admin regisztrálása

Regisztrálás

admin id : 5  
felhasználó : sonbi

Törlés

Frissítés

admin id : 7  
felhasználó : admin

Törlés

## 3 Irodalomjegyzék

### 3.1 Internetes források

1. Stack overflow → <https://stackoverflow.com/>
2. W3Schools → <https://www.w3schools.com/>
3. PHP Tutorial → <https://www.phptutorial.net/>
4. GitHub → <https://github.com/>
5. Code Forum → <https://codeforum.org/>
6. Geeks for Geeks → <https://www.geeksforgeeks.org>
7. Javatpoint → <https://www.javatpoint.com>
8. php.net → <https://www.php.net/>
9. mysql.com → <https://www.mysql.com/>

### 3.2 Könyvek

*How To Design Programs* – Robert Bruce Findler, Matthias Felleisen, Shriram Krishnamurthi, Matthew Flatt – MIT Press (2001)

*Web Design with HTML, Css, Javascript and JQuery Set* – Jon Duckett - 2014

*Tiszta Kód* – Robert C. Martin – Kiskapu (2008)

*Learn Javascript Visually* – Ivelin Demirov – 2014

*PHP & MySQL: Server-side Web Development* – Jon Duckett - 2022



## 4 Mellékletek

Alkalmazásunk fontosabb kódjainak elérése githubon:

<https://github.com/SchonbergerDominikBoronkay/HasznaltSportszer>

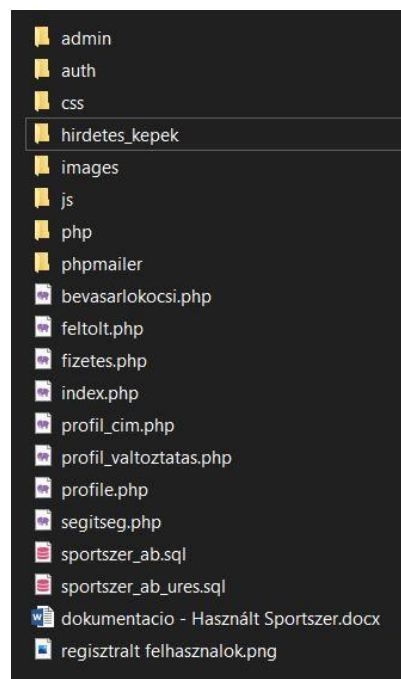
Illetve a teljes állomány elérése GoogleDrive-on:

<https://drive.google.com/drive/u/1/folders/1CoPtOKLU9ZpCrW2RnX8SHeFxAR5LC4HA>

Az alkalmazásunk mappa szerkezete itt látható.



1. Az *admin* mappában találhatóak az admin felülethez szükséges kódjaink,
2. Az *auth* mappában a regisztrációhoz és a bejelentkezéshez szükséges kódok,
3. A *css* mappában a designért felelős fájlok,
4. A *hirdetes\_kepek* illetve az *images* mappában az oldal működéséhez szükséges képek,
5. A *js* mappában a javascript kódok,
6. A *php* mappában pedig azon kódjaink amik nem tartalmaznak mást csak php funkciókat.



Illetve itt található meg a két adatbázis (üres és a teszt adatokkal feltöltött), a dokumentáció és a teszteléshez szükséges bejelentkezési adatok is.