

Webprogramozás II.

Göncziné Kapros Katalin – Csanádi Péter

MÉDLAINFORMATIKAI KIADVÁNYOK

Webprogramozás II.

Göncziné Kapros Katalin – Csanádi Péter



Eger, 2013



Korszerű információtechnológiai szakok magyarországi adaptációja

TÁMOP-4.1.2-A/1-11/1-2011-0021

Nemzeti Fejlesztési Ügynökség
www.ujszechenyiterv.gov.hu
06 40 638 638



A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.

Lektorálta:

Nyugat-magyarországi Egyetem Regionális Pedagógiai Szolgáltató és
Kutató Központ

Felelős kiadó: dr. Kis-Tóth Lajos

Készült: az Eszterházy Károly Főiskola nyomdájában, Egerben

Vezető: Kérészy László

Műszaki szerkesztő: Nagy Sándorné

Tartalom

1.	<i>Bevezetés</i>	<i>9</i>
1.1	Célkitűzések, kompetenciák a tantárgy teljesítésének feltételei..	9
1.1.1	Célkitűzés.....	9
1.1.2	Kompetenciák.....	9
1.1.3	A tantárgy teljesítésének feltételei	10
1.2	A kurzus tartalma	10
1.3	Tanulási tanácsok, tudnivalók	11
2.	<i>Elektronikus kereskedelem, online áruházak</i>	<i>13</i>
2.1	Célkitűzések és kompetenciák	13
2.2	Tananyag	13
2.2.1	Az elektronikus kereskedelem kialakulása	14
2.2.2	Elektronikus kereskedelem Magyarországon.....	17
2.2.3	Az elektronikus kereskedelem meghatározása	19
2.2.4	Az elektronikus kereskedelem előnyei, nehézségei	20
2.2.5	Az elektronikus kereskedelem biztonsági kérdései, kockázatok és veszélyforrások.....	23
2.2.6	Online áruházak.....	23
2.3	Összefoglalás, kérdések	32
2.3.1	Összefoglalás	32
2.3.2	Önellenőrző kérdések.....	32
3.	<i>Egy online áruház moduljai, a kivitelezés megtervezése</i>	<i>35</i>
3.1	Célkitűzések és kompetenciák	35
3.2	Tananyag	36
3.2.1	Felület kialakítása	37
3.2.2	Adatbázis megtervezése.....	57
3.2.3	Megjelenítendő adatok szerepe az oldalon	60
3.3	Összefoglalás, kérdések	61
3.3.1	Összefoglalás	61
3.3.2	Önellenőrző kérdések.....	61
4.	<i>Szoftverergonómiai és marketingszempontok.....</i>	<i>63</i>
4.1	Célkitűzések és kompetenciák	63

4.2	Tananyag.....	63
4.2.1	Használhatóság.....	64
4.2.2	A weblap felépítése.....	67
4.2.1	(Hon) Laptervezés.....	68
4.3	Összefoglalás, kérdések	72
4.3.1	Összefoglalás	72
4.3.2	Önellenőrző kérdések.....	73
5.	<i>Az adminisztrációs felület megvalósítása</i>	75
5.1	Célkitűzések és kompetenciák.....	75
5.2	Tananyag.....	75
5.2.1	Tervezés és funkciók létrehozása.....	76
5.2.2	Adminisztrációs lehetőségek.....	77
5.2.3	A felület programozása	79
5.3	Összefoglalás, kérdések	83
5.3.1	Összefoglalás	83
5.3.2	Önellenőrző kérdések.....	83
6.	<i>Felhasználói azonosítás, személyre szabott tartalom</i>	85
6.1	Célkitűzések és kompetenciák.....	85
6.2	Tananyag.....	85
6.2.1	Felhasználói adatok	86
6.2.2	Adatok védelme, biztonság	87
6.2.3	Jogosultságok kialakítása.....	87
6.2.4	Új felhasználó regisztrációja, ki- és bejelentkezés	88
6.3	Összefoglalás, kérdések	122
6.3.1	Összefoglalás	122
6.3.2	Önellenőrző kérdések.....	123
7.	<i>Termékek és kategóriák létrehozása</i>	125
7.1	Célkitűzések és kompetenciák.....	125
7.2	Tananyag.....	125
7.2.1	Termékek tárolása	126
7.2.2	Termékcsoportok, kategóriák módosítása	146
7.2.3	Műveletek a termékekkel.....	154
7.3	Összefoglalás, kérdések	154
7.3.1	Összefoglalás	154
7.3.2	Önellenőrző kérdések.....	155

8.	<i>Termékek rendszerezése, szelekciós funkciók</i>	<i>157</i>
8.1	Célkitűzések és kompetenciák	157
8.2	Tananyag	157
8.2.1	Termékek megjelenítése	158
8.2.2	Termékek keresése	165
8.2.3	Termékek szűrése	168
8.3	Összefoglalás, kérdések	174
8.3.1	Összefoglalás	174
8.3.2	Önellenőrző kérdések.....	174
9.	<i>Elektronikus bevásárlókosár megvalósítása</i>	<i>175</i>
9.1	Célkitűzések és kompetenciák	175
9.2	Tananyag	175
9.2.1	Kosár feladatai.....	176
9.2.2	Kosár megtervezése	176
9.2.3	Termék kosárba helyezése	180
9.3	Összefoglalás, kérdések	185
9.3.1	Összefoglalás	185
9.3.2	Önellenőrző kérdések.....	186
10.	<i>Biztonsági stratégiák</i>	<i>187</i>
10.1	Célkitűzések és kompetenciák	187
10.2	Tananyag	187
10.2.1	Biztonsági fenyegetések	188
10.2.2	Titkosítás alapjai	188
10.2.3	Felhasználói adatok védelme	189
10.2.4	SSL protokoll használata	189
10.2.5	Biztonságos tranzakciók	192
10.3	Összefoglalás, kérdések	193
10.3.1	Összefoglalás	193
10.3.2	Önellenőrző kérdések.....	193
11.	<i>Pénztár és a fizetési rendszer</i>	<i>195</i>
11.1	Célkitűzések és kompetenciák	195
11.2	Tananyag	195
11.2.1	Megjelenő adatok.....	195
11.2.2	Termékek összesítése	198

11.2.3	Fizetés megvalósítása.....	203
11.3	Összefoglalás, kérdések	207
11.3.1	Összefoglalás	207
11.3.2	Önellenőrző kérdések.....	207
12.	Összefoglalás	209
12.1	Tartalmi összefoglalás.....	209
12.2	Zárás	209
13.	Kiegészítések.....	211
13.1	Irodalomjegyzék	211
13.1.1	Hivatkozások.....	211

1. BEVEZETÉS

1.1 CÉLKITÚZÉSEK, KOMPETENCIÁK A TANTÁRGY TELJESÍTÉSÉNEK FELTÉTELEI

1.1.1 Célkitűzés

Könyvünk áttanulmányozása után tisztában lesz az elektronikus kereskedelem fogalmával, működésével, megismeri a benne rejlő lehetőségeket, annak veszélyeit, működésének biztonságos feltételeit. Minden ismeret rendelkezésre áll majd ahhoz, hogy képes legyen egy használható, mind szerkezetében, mind funkciójában átgondolt, jól működtethető saját online áruházat megtervezni és létrehozni.

1.1.2 Kompetenciák

A főbb kompetenciák a következők:

Átfogó ismeretek az elektronikus kereskedelem, on-line áruházak felépítéséről, szerkezetéről, biztonsági kérdéseiről. A PHP programnyelven és MySQL adatbázis használatával történő online áruház kivitelezésének megtervezése, megvalósítása.

- tudás

Releváns ismeretekkel rendelkezik az az elektronikus kereskedelem terén. Birtokában van az online áruházak felépítésének, szerkezetének, biztonsági kérdéseit magába foglaló tudásnak. Rendelkezik program tervezéséhez és megírásához szükséges ismeretekkel. Ismeri az adatbázis rendszerek felépítését, működését. Tud adatbázist programozni, optimalizálni, kezelni. Képes pontos feladatmeghatározás után adatmodellt felállítani. Releváns ismeretekkel rendelkezik a PHP nyelven történő programozást illetően. Képes használni a PHP fejlesztői környezetét, tudja használni a programozási nyelv nyújtotta lehetőségeket. Tudja tesztelni programját, hibát keresni, majd javítani benne, önellenőrzésre képes.

- attitűdök/nézetek

Az e-kereskedelemben, felépítésében, annak veszélyei és lehetőségei ismeretében, a kihívások tudatában legyen képes a felé támasztott követelmények teljesítésére. Nyitott legyen a felmerülő új problémák megoldására, csoportban történő megvalósításában. Ismereteit képes kreatívan felhasználni és

szakszerűen kommunikálni. Az elé táruuló feladatokat felelősségtudattal végezze el, kellő elhivatottsággal és pontossággal. Rálátása legyen az online áruházak működésére, képes legyen megkonstruálni egy jól működtethető áruház menedzselését.

- képességek

Legyen képes a tanultak alapján a tananyag kreatív alkalmazására és az önálló gondolkodásra. Képes legyen a benne foglalt programrészletek felhasználására, átdolgozására, ezek segítségével az adott feladat(ok) kreatív megoldására. Alkalmazni tudja az eddig megszerzett ismereteit, új megközelítésben egyedi megvalósításban képes alkotni. Képes az adott problémát programozói szemszögből megközelíteni és kreatívan megvalósítani. Képes egy használható, mind szerkezetében, mind funkciójában átgondolt, jól működtethető on-line áruház kivitelezésének megtervezésére, létrehozására. A tananyag elsajátítása révén képes on-line áruház létrehozására, a PHP programozási nyelv által nyújtott lehetőségek kihasználására.

1.1.3 A tantárgy teljesítésének feltételei

Az tananyag elsajátítása után ismerni fogja az elektronikus kereskedelem fogalmát, működését, a benne rejlő lehetőségeket, annak veszélyeit, működésének biztonságos feltételeit. Rendelkezik majd azokkal az elméleti és gyakorlati ismeretekkel, amelyek birtokában képes lesz egy használható, mind szerkezetében, mind funkciójában átgondolt, jól működtethető on-line áruház létrehozására.

A tanegység teljesítésének feltételei a következők:

- Az elméleti ismereteket magába foglaló feladatlap eredményes kitöltése.
- Órák elején történő 5-10 perces beszámolók sikeres teljesítése.
- Az órákon elkészített részprogramok otthoni átdolgozása, a félév végére összeálló online bolt elkészítése, bemutatása.

1.2 A KURZUS TARTALMA

1. Bevezetés
2. Elektronikus kereskedelem, online áruházak
3. Egy online áruház moduljai, a kivitelezés megtervezése
4. Szoftverergonómiai és marketing szempontok
5. Az adminisztrációs felület megvalósítása

6. Felhasználói azonosítás, személyre szabott tartalom
7. Termékek és kategóriák létrehozása
8. Termékek rendszerezése, szelekciós funkciók
9. Elektronikus bevásárlókosár megvalósítása
10. Biztonsági stratégiák
11. Pénztár és a fizetési rendszer
12. Összegzés

1.3 TANULÁSI TANÁCSOK, TUDNIVALÓK

A könyv 10 leckéje felöleli mindazokat az ismerteket, amelyekre Önnek a témában szüksége van. A leckék nem csupán száraz tananyagot tartalmaznak, hanem megpróbálnak együtt gondolkodni az olvasóval. Problémákat, kérdéseket vetnek fel, példákat mutatnak be, feladatokat fogalmaznak meg. Minden lecke végén önellenőrző kérdések segítik az olvasót abban, hogy felmérje saját tudását.

2. ELEKTRONIKUS KERESKEDELEM, ONLINE ÁRUHÁZAK

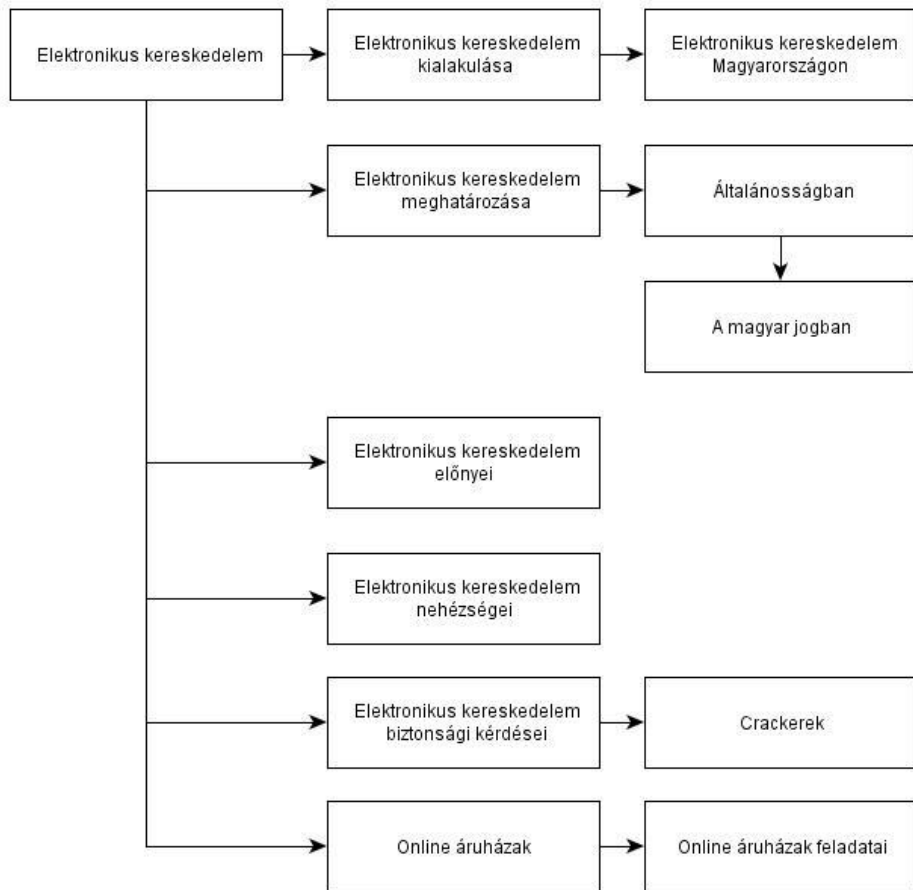
2.1 CÉLKITŰZÉSEK ÉS KOMPETENCIÁK

Tananyagunk első leckéjében megismerkedhet az elektronikus kereskedelem kialakulásával, fogalmával, megismeri a benne rejlő lehetőségeket, előnyeit és nehézségeit, valamint annak veszélyeit.

Az első lecke végére Ön már ismerni fogja az online áruházak működésének és működtetésének biztonságos feltételeit.

2.2 TANANYAG

Fokozatosan több problémát okoz, hogy a termékét eladni kívánó résztvevő számára a piaci szférába való bejutás egyre nehezebb, amennyiben a hagyományos értelemben vett értékesítést kívánja megvalósítani. A technológia fejlődésével lépést tartva és annak előnyeit kihasználva napjaink kereskedelmében egyre fontosabb szerepet kapnak az új, alternatív értékesítési formák. Könnyű és gyors árucserét biztosítanak az internetalapú technológiára támaszkodó direkt értékesítést megvalósító csomagküldő szolgálatok, „licit” vásárok, elektronikus boltok.



1. ábra: A lecke áttekintése

2.2.1 Az elektronikus kereskedelem kialakulása

A számítógépek rohamos fejlődése és elterjedése, valamint a távközlő hálózatokon történő adattovábbítás lehetősége biztosította az üzleti célú felhasználását. Az internet segítségével a kommunikációban megszűntek a földrajzi távolságok, az információk áramlása akadálytalanra vált. Az elektronikus kereskedelem fellendülését az internet és a kereskedelmi tranzakciókat támogató számítástechnikai alkalmazások fejlődése segítette elő. A folyamatosan bővülő technológiai újdonságok kiaknázatlan lehetőségekkel bírnak.

Az elektronikus kereskedelem (továbbiakban e-kereskedelem) kialakulása több mint száz évre nyúlik vissza.

Az első, hálózaton keresztül történő kereskedelmet lebonyolítóknak az 1910. augusztus 18-án összeállt 15 amerikai virágkereskedő tekinthető, akik táviró segítségével valósították meg az áruk és a megrendelések cseréjét. A Florist' Telegraph Delivery virágkereskedési hálózathoz fejlődött ki a mai FTD Inc., amely valószínűleg az első valódi e-kereskedelmi hálózat.¹

1948 júniusa és 1949 májusa között Kelet-Németország a Szovjetunió fennhatósága alá tartozott, amely elvágtatta a közúti, vasúti és vízi összeköttetést Nyugat-Németország és Nyugat-Berlin között. A blokádot a szovjetek azért vezették be, hogy kikényszerítsék akarataik érvényesítését a nyugati hatalmak szektoraiban. A blokádnak súlyos ellátási zavarokhoz vezetett Nyugat-Berlinben, amelynek megoldására a nyugati szövetségesek léghíddal kötötték össze a várost a nyugati országrésszel. Több mint 2 millió tonna élelmiszert és más ellátmányt juttattak el repülőgépek segítségével Nyugat-Berlinbe. De a szállítmányok nyomon követését szinte ellehetetlenítette a nyelvi különbözőségek és a szállítólevelek formai sokszínűsége. A megoldást Edward A. Guilbert, az USA hadseregének szállítmányozási tisztje által kidolgozott szabványos jegyzékrendszer jelentette, amely továbbítható volt telexszel, rádiótávíróval vagy telefonnal. Ezen egységes jegyzék alkalmazásával a továbbított üzenetből mindkét fél ugyanazon információ birtokába került. A rendszer továbbfejlesztésének sikerét bizonyítja, hogy 1968-ra már olyan sok vasúti, légi, közúti és vízi szállító társaság használta az elektronikus jegyzéket, hogy létrehozták a TDCC bizottságot (Transportation Data Coordinating Committee – Szállítványozási Adat Koordinációs Bizottság), hogy koordinálja iparközi szabványok létrehozását. Ebből fejlődött ki 1975-ben a máig használatos első elektronikus adatcsereszabvány, az EDI (Electronic Data Interchange) specifikációja.²

A mai e-kereskedelmi vállalkozások az FTD virágkereskedési hálózat és a berlini léghíd által kifejlesztett elveken alapulnak.

Az első valódi, széles körben elterjedt rendszer, amely digitális üzleti tranzakciókat tett lehetővé, Franciaországban alakult ki 1981-ben. A MINITEL hálózatának működése már a kezdetekben is meglehetősen egyszerű és átlátható volt. Ez egy videotext rendszer volt, amely a telefont kombinálta egy 8 hüvelykes monitorral. A 80-as évek közepén már több mint 3 millió Minitel készüléket alkalmaztak és közel 18 000 szolgáltatás volt elérhető rajta, például jegyvásárlás, utazás- és termékeladás, valamint online bankolás. A Minitel rendszer egé-

¹ Eszes I. – Bányai E. (2002): Online marketing, Műszaki Könyvkiadó, Budapest.

² Eszes I. – Bányai E. (2002): Online marketing, Műszaki Könyvkiadó, Budapest.

szen 2006. december 31-ig létezett, amikor tulajdonosa, a France Telecom végleg leállította.³

A 90-es évek elején az internet még – mondhatni – „gyermekcipőben járt” a mai állapotához képest, hiszen kizárólag szöveges információk megjelenítésére volt alkalmas, nehezen volt kezelhető, széles körben való elfogadtatása a kereskedelmi és a magánszférával szinte lehetetlen volt.

A svájci CERN fizikai kutató központban dolgozó Tim Berners-Lee célja egy olyan „univerzális nyelv” kidolgozása volt, mellyel leküzdhető a szöveges tartalmak kizárólagos hatalma, és egy olyan nyelv kidolgozása, amely minden – a rendszerhez kapcsolódó – számítógépen futtatható. Lee 1990-ben, HTML nyelven íródott „WorldWideWeb” nevű programja meghozta az áttörést. Az „univerzális nyelv” ennek köszönhetően az egérkattintás lett, mely az új programmal (világnézettel) karöltve lehetővé tette, hogy immár nemcsak szöveges tartalmak, hanem ábrák, grafikonok, a későbbiekben pedig képek, sőt multimédiás tartalmak is elérhetővé váltak az interneten keresztül.

Az internet (és a rajta végbemenő e-kereskedelem) kialakulása és elterjedése Tim Berners-Lee munkásságának köszönhető, segítségével az egykor kizárólag szűk réteg számára elérhető rendszer mára a laikusok számára is könnyen megérthetővé és kezelhetővé, így a rajta folyó későbbi kereskedelem bölcsőjévé is válhatott.

Az internet mint új infrastruktúra forradalmasította az emberek közötti kommunikációt, lehetővé téve, hogy emberek milliói cseréljenek információt, legyőzve földrajzi távolságot, ledöntve társadalmi és adminisztratív válaszfalakat.

Az internet és a kereskedelmi tranzakciókat támogató számítástechnikai alkalmazások fejlődése robbanásszerű lendületet adott az elektronikus kereskedelem kibontakozásának. A vállalatok gyorsan felismerték az internetben rejlő üzleti lehetőségeket, és a 90-es évek közepétől kezdődően exponenciálisan nőtt az online üzleti tevékenységet folytató cégek száma. A visszaesések ellenére az e-kereskedelmet folytató cégek jelentős sikereket értek el, mellyel biztosították ezen új kereskedelmi forma fennmaradását.

Napjainkban csak azok a vállalatok maradnak fenn, amelyek jól szervezettek, képesek a marketingkonceptiót szem előtt tartva valamilyen értéket felmutatni, és ezáltal hosszabb távon profitot realizálni. Az ezen feltételeket nem teljesítők kénytelenek lehúzni a virtuális redőnyt. A folyamat, melyben az erő-

³ Eszes I. (2011.): e-Kereskedelem, ESZES Könyvtár

sebb vállalkozások bekebelezik a gyengébbeket, a fúziók és a vállalatfelvásárlások sohasem áll le.

2.2.2 Elektronikus kereskedelem Magyarországon

A Magyarország történelmében adódó viharos időszakok jelentős gátat képeztek a fejlődés útjába. Ennek következményeként az internet és a rajta folytatható tevékenységek csak a 90-es évek elején jelentek meg. Széles körben való elterjedésük viszont dinamikus fejlődést mutat. A legelső magyar .hu végű internetes azonosítót 1991 októberében vették nyilvántartásba. A sztaki.hu, a hazai internet bölcsőjének tekinthető intézmény, az MTA Számítástechnikai és Automatizálási Kutató Intézete nevére lett lefoglalva. 2010-ben a magyar domain nevek száma 4 756 913 darabra gyarapodott. Az internet mindennapos használatának elterjedése fellendülést hozott az e-kereskedelem területén.

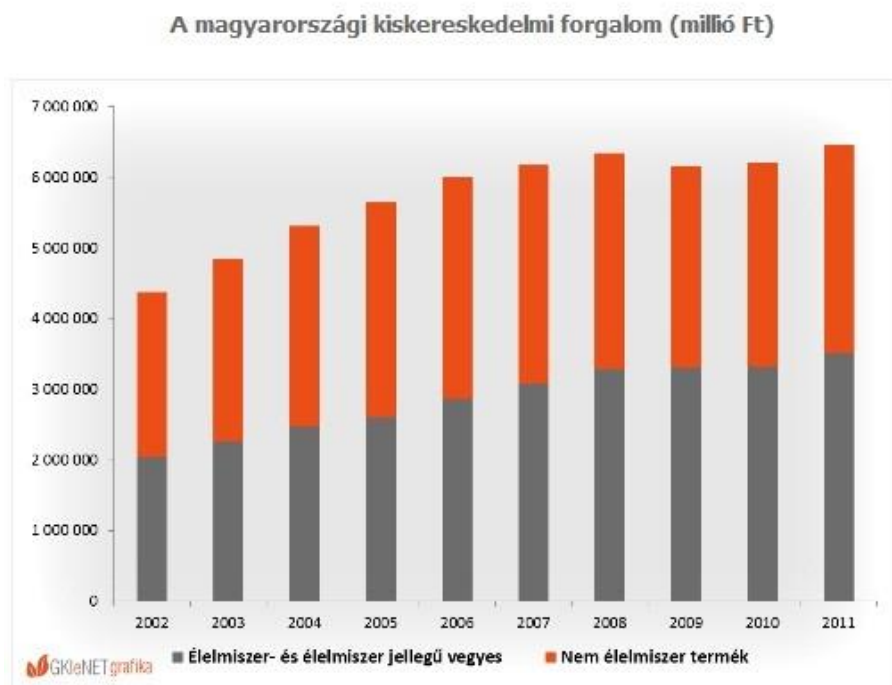
Az NRC-TNS 2009-es Gyorsjelentése felmérése szerint amíg 2005-ben a felnőtt lakosságnak alig egynegyede használta az internetet, mára már a 18–64 évesek fele legalább hetente „online”, és további 8-10% próbálta már ki a világháló nyújtotta előnyöket.

Az internetezők számának növekedésén túl az elmúlt években az online vásárlási kedv is folyamatosan emelkedett; míg 2008-ban az aktív (legalább hetente) netezők 62%-a vásárolt már valaha a világhálón keresztül, a 2009-es felmérés tanúsága szerint mára több mint háromnegyede (78%-a) legalább egyszer élt már az e-shopping lehetőségével. A gyakori internetezőkön belül 40%-ról 46%-ra nőtt azok aránya, akik az elmúlt 12 hónapban legalább egyszer vásároltak a világhálón keresztül. Kisebb-nagyobb rendszerességgel 43 százalékuk rendel termékeket, szolgáltatásokat az internetről.⁴

2011-ben nagy mértékű növekedés volt megfigyelhető az internetes kereskedelemben, amely a gazdaság kevés területén valósulhat meg. Ez az eredmény a GKIE NET – T-Mobile: „Jelentés az internetgazdaságról” című kutatásából derül ki.

A teljes magyarországi kiskereskedelem 2,4%-a már online módon történt. A 2009 és 2010-es a hazai kiskereskedelmi forgalom egésze szempontjából (online, offline) nehéz év után 2011-ben a forintban mért forgalom csekély mértékben, de nőtt, meghaladva a 2008. évi szintet. Ez a fajta növekedés csak az élelmiszer és az élelmiszer jellegű vegyes termékek forgalmára vonatkozik, a nem élelmiszer termékeké jelentősen romlott: még a 2005. évi szintet sem éri el.

⁴ Eszes I. (2011.): e-Kereskedelem, ESZES Könyvtár



Forrás: gkienet.hu

2. ábra: Kiskereskedelmi forgalom Magyarországon⁵

Mára elfogadottá és szinte természetessé vált az a tevékenység, hogy a vásárlás előtti tájékozódás az interneten történik. 2011 végén az elmúlt egy évben online vásárlók száma 1,4 millió fő volt, ami a 14 év feletti internetezők megközelítőleg 35%-a, derül ki a kutatásból. 2011-ben 155 milliárd forintnyi árut vásároltak online módon a fogyasztók, ami a teljes hazai kiskereskedelmi forgalom 2,4%-a. Ha a kereskedők várakozásai teljesülnek, ez a szám tovább nő majd az elkövetkező években – 2012-ben elérheti akár a teljes kiskereskedelmi forgalom 2,9–3,0%-át növekedhet, ami 175-180 milliárd forintos online forgalmat jelentene.

⁵ <http://gkienet.hu/hu/hirek/egyre-tobb-webkosarat-tol-a-magyar/>



3. ábra: Internetes kiskereskedelmi forgalom Magyarországon⁶

Az elektronikus úton történő vásárlás sikerét amennyiben a bizalom kialakult ezen tevékenységgel szemben, a gyors, rugalmas (nincs nyitvatartási idő), kényelmes (bárhonnan elérhető) intézhetőség adja, amely mára nélkülözhetetlen a „rohanás” világában élő emberek számára.

2.2.3 Az elektronikus kereskedelem meghatározása

Az elektronikus kereskedelem a mai fejlett és dinamikusan fejlődő számítógép-hálózatra támaszkodik, elsősorban az internet nyújtotta lehetőségeket kihasználva. Sikerességét és széles körben való elterjedését a hálózat nyújtotta elérhetőség, naprakészség, egyszerű és gyors kapcsolattartás, ügyintézés alapozta meg. Az e-kereskedelem magában foglalja a termelő/szolgáltató vállalatok közötti, vállalat és fogyasztó, és főképp a fogyasztók közötti kapcsolatokat, és elősegíti az áruk, szolgáltatások, információk kereskedelmét. Megvalósítja az elektronikus úton történő beszerzést és értékesítést.

⁶ <http://gkienet.hu/hu/hirek/egyre-tobb-webkosarat-tol-a-magyar/>

Az e-kereskedelem nem különül el más üzleti tevékenységtől, sokkal inkább egy új, egységet alkotó formáról van szó, amely forradalmasítja a hagyományos kereskedelmet, a modern technológia használatával eleget tesz a kor elvárásainak.

Az elektronikus kereskedelem (e-kereskedelem vagy e-commerce) fogalma gyakran keveredik az elektronikus üzletvitel (e-business) fogalmával, ezért lényeges a fogalom tisztázása.

A ma hatályos magyar jogszabályozásban az elektronikus kereskedelmi szolgáltatások, valamint az információs társadalommal összefüggő szolgáltatások egyes kérdéseiről szóló 2001. évi CVIII. törvény 2.§. a) pontja szerint a következő. Elektronikus kereskedelmi szolgáltatás: olyan információs társadalommal összefüggő szolgáltatás, amelynek célja valamely birtokba vehető forgalomképes ingó dolog – ideértve a pénzt és az értékpapírt, valamint a dolog módjára hasznosítható természeti erőket –, szolgáltatás, ingatlan, vagyoni értékű jog üzletszerű értékesítése, beszerzése, cseréje vagy más módon történő igénybevétele.⁷

2.2.4 Az elektronikus kereskedelem előnyei, nehézségei

Az e-kereskedelem legfontosabb előnyei

- A legfőbb előnye, hogy elektronikusan kapcsolja össze megjelenési formájától függően a kereskedőt és a vásárlót (B2C), a vállalkozókat (B2B), illetve a kormányzatot és vállalatokat (B2A), és a bankok kapcsolatával a vásárlás nagy része automatizálható.
- Az e-kereskedelem világában nincs nyitvatartási idő, a nap 24 órájában az év 365 napján hozzáférhetők az információk, lebonyolíthatóak a tranzakciók, ezáltal biztosítva az időtakarékos termékkeresés. Az elérhetőség mind az eladó, mind a vevő számára kényelmes üzletelést biztosít.
- A nagyobb áruválaszték kiterjeszti a lehetőségeket. Több szállító/termelő több terméke közül lehet kiválasztani az egyén számára a legmegfelelőbbet, ami alacsonyabb árakhoz vezethet a jobb összehasonlítási lehetőségek miatt.

⁷ http://net.jogtar.hu/jr/gen/hjegy_doc.cgi?docid=A0100108.TV (2012. július 07.)

- Számos termék csak a webáruházakban érhető el, amely interneten történő fizetés esetén azonnal megkapható. Például szoftverek, digitalizált rajzok, dokumentumok... stb.
- Megszűnnek a földrajzi határok, lehetővé teszi a piacok kiterjesztését és a nemzeti, illetve a nemzetközi piacon való megjelenést, akár eladásról, akár vásárlásról legyen szó. A termékek és szolgáltatások szinte globális elérhetőségét teszi lehetővé.
- A termékek részleteiről gazdagabb információ áll rendelkezésre, mint a hagyományos papír formában, például hanganyagok, termékbemutató videók, klipek, demonstrációs anyagok. Javítani lehet a szolgáltatások színvonalát, a termékegységet, az információk hozzáférhetőségét, a rugalmasságot stb.
- Lehetőség van a készlet rugalmasabb tervezésére, az előre befutó vevői rendelések kapcsán.
- Csökkenti az előállítási, feldolgozási, értékesítési, raktározási költségeket.
- Csökkenti a helyiségbérlés, a reklám és a rendelésfeladás költségeit.
- Lehetővé teszi a papíralapú információfeldolgozás költségeinek csökkentését – olcsóbb az információ létrehozása, feldolgozása, elosztása, tárolása és visszakeresése is.
- Nagyfokú mobilitás és helyrugalmasság. Fizikai helyváltoztatás nélkül lehetővé teszi a munkavégzést (táv munka), a vásárlást, ami csökkenti a környezetterhelést.
- A közvetítők számának csökkentése.
- Csökkenthetők az árak a nagyobb hatékonyság miatt.

Az e-kereskedelem nehézségei

- A rendszer biztonságának és megbízhatóságának kérdései, a szabványok és kommunikációs protokollok hiánya.
- Nagyfokú személytelenség a tranzakcióban résztvevő mindkét fél részéről, a személyes kapcsolatok és interakciók háttérbe szorulnak.
- A globalizáció – előnyei mellett – számos problémát vet fel, megválaszolásra várnak a felmerülő kulturális kérdések, melyik legyen a „standard” kultúra, nem sérti-e ez az országok nemzeti érdekeit vagy nemzeti gazdagságát.

- A vevőkben az online tranzakciók biztonságával szemben támasztott bizalmatlansága a személytelenségből, a személyiségi és jogi problémákból, valamint az elektronikus pénz megfoghatatlansága révén adódik.
- Megoldatlan jogi, etikai szabályozási problémák vannak. Például: mikor lép életbe a szerződés, jogvita esetén mi az eljárás mód, milyen elektronikus dokumentumok fogadhatóak el hiteles okmányként stb. Hasonló megoldatlan probléma az információszolgáltatók felelőssége a közölt információkért. Valamint az elektronikusan terjeszthető termékek esetén felmerülő szerzői jogi szabályozások nem tisztázottak.
- Az előzetes irányított piackutatás hiányában a vevők igényeinek, szokásainak felmérése nélkül a legjobban hangzó márkákat és legtetszetősebb termékeket sem lehet eladni a virtuális térben.
- A termék tényleges fogyasztóját nem lehet, vagy csak nehezen lehet bevonni a termék tervezésébe, kipróbálásába, illetve az ügyfél pozitív vagy negatív visszajelzései is csak körülményesen juthatnak el a gyártóhoz, akinek ez megnehezíti a piaci munkáját.
- A legritkább esetben van arra lehetőség, hogy egy termék vásárlója közvetlen kapcsolatba kerüljön a termék gyártójával.
- Nagyobb webáruházak esetén a tapasztalt szakemberek hiánya. A nem megfelelő képzettséggel rendelkező emberek alkalmazása nagy hátrányt jelent a kereskedelmi piacon.
- Nem megfelelő sáv szélesség lelassíthatja némelyik üzleti folyamatot.
- Fejlesztés hiánya. Fejlesztés nélkül – például újabb szolgáltatások, versenyképes akciók, új termékek, megbízható márkák forgalmazása, a felület látványgazdagsága nélkül – elavult, forgalommal nem rendelkező, összességében nem megbízható online áruház benyomását kelti.
- A működéshez szükséges erőforrások félrebecsülése. Fel kell mérni, hogy milyen szoftver- és hardvereszközök, humán erőforrások szükségesek az optimális működéshez.

A technológia fejlődésével a szolgáltatások és a termékek széleskörű változásának egyenes következménye, hogy az említett szempontok némelyike fokozatosan érvényesül, míg mások teljesen megszűnnek, illetve újabbak lépnek érvénybe.

2.2.5 Az elektronikus kereskedelem biztonsági kérdései, kockázatok és veszélyforrások

A veszélyek az élet minden területének velejárói. Nincs ez másként az üzleti életben sem. A hagyományos vállalkozásokban ismert kockázatok egy része az e-kereskedelmi cégekre nem, vagy csak kis mértékben jelent veszélyt. Az igazi veszélyforrások a következők:⁸

- Crackerek
- A kívánt üzleti eredmény elmaradása
- Számítógépes hardverhibák
- Elektromos, kommunikációs vagy hálózati hibák
- Erős verseny
- Szoftverhibák
- Változó szabályozási környezet és adójogszabályok
- Rendszer-kapacitásbeli korlátok

2.2.6 Online áruházak

Az online áruház egy gyűjtőfogalom, amely tetszőleges internetes értékesítési megoldást takar. Az online áruház lehet egy átlagos internetes bolt – a legelterjedtebb a „kosaras” megoldás –, de lehet szoftver és digitális termékek esetén egy internetes weboldal is, amelyen az értékesítés tetszőleges részfolyamata elvégezhető, legyen az megrendelés, fizetés, vagy akár magának a terméknek a vevőhöz való eljuttatása.

Az online áruházak szinonimájaként használható a webshop, webáruház, az internetes bolt, és online bolt kifejezés is.

Az online áruház célja, hogy megfelelő felületet biztosítson az elektronikus kereskedelemhez, biztosítsa annak feltételeit, egy internetes vásárlási lehetőséget biztosítva a leendő ügyfeleknek.

Lássunk néhány online áruházat.⁹

⁸ Luke Welling, Laura Thomson: PHP és MySQL webfejlesztőknek – Hogyan építsünk webáruházat?, Budapest, 2010 Perfect-Pro Kft.

⁹ www.ebolt.hu; www.grando.hu

eBolt INTERNETES MŰSZAKI ÁRUHÁZ

Forrás: www.ebolt.hu

Bejelentkezés
Kosár
Szerezési Feltételek
Ügyfélszolgálat
Rendeléskövetés
Garancia
Hírfelvi

Létező
Utahány
Akciók
Újdonságok
Akciók raktárról
Leértékelt
Kedvezmények
Kedvenceim

Vedd át rendelésed kedvezményesen RELAY és Inmedio hírlapúletekben!
Az akció időtartama: 2012. 08. 23. – 2012. 09. 23.

Nyitvatartás: H-P 10-18 óráig
Tel.: (06-1-) 88-99-222

2012.09.17 > KERESÉS

Kosár tartalma:
Az Ön kosara üres!
részletek >

SAVÁZOK
VÁSÁRLÓI BOLTA 2012

TERMÉKKATEGÓRIÁK

- > Notebook, netbook, tablet
- > Számítástechnika
- > Irodaszerek, irodatechnika
- > Háztartási gépek
- > Szórakoztató elektronika
- > Szépség-Egészség
- > Telefonok és tartozékok
- > Ház, kert, autó
- > Barkácsológépek, szerszámok

NOTEBOOK, NETBOOK, TABLET 1.362 db termék

Dell Latitude E5520 15.6" notebook
RÉSZLETEK 227.228 Ft kosárba

SZÁMÍTÁSTECHNIKA 13.110 db termék

HKC 9812 18.5" Wide LED monitor
RÉSZLETEK 25.083 Ft kosárba

IRODASZEREK, IRODATECHNIKA 8.504 db termék

Shany MTC-L733 1/3" Sony HAD CCD kültéri IR kamera, 420 sor
RÉSZLETEK 14.859 Ft kosárba

HÁZTARTÁSI GÉPEK 6.950 db termék

Dyras ESI-2200 CS gőzölős vasaló
RÉSZLETEK 4.798 Ft kosárba

SZÓRAKOZTATÓ ELEKTRONIKA 4.765 db termék

SZÉPSÉG-EGÉSZSÉG 1.860 db termék

Bejelentkezés
Új regisztráció

Ma eddig 2 terméket rendeltek az eBoltban!

Miért érdemes az eBoltban vásárolni?

- > Ereketi gyártói garancia minden termékhez
- > Kedvező árak és akciók
- > 1999 óta működő áruház
- > Udvartársi ügyfélszolgálat
- > ISO 9001 tanúsítvány, minőségközpontú kiszolgálás
- > Személyes átvétel az ország több, mint 3000 pontján, részletek >

LEGNEPSZERŰBB TERMÉKEINK

1. Elektromos cigarettatöltő gép 11.641 Ft részletek >
2. Sandisk 16GB Extreme HD Video SDHC memóriakártya 6.990 Ft részletek >

4. ábra: www.ebolt.hu

grando online pláza

Forrás: www.grando.hu

SEGÍTSÉG | REGISZTRÁCIÓ | BEJELENTKEZÉS

KATEGÓRIÁK

KOSÁR (0 DB)

Számítástechnika

- Laptop, netbook, netbook
- Otthon, építkezés, kert
- Háztartási gép, kisgép
- Műszaki cikkek
- Mobiletelefon, vezetékes készülék
- Utazás, szabadidő
- Autó - motor
- Ruházat, divat
- Ékszer, óra
- Játék
- Egészség, szépségápolás
- Sport, fitness
- Videokamerák és fényképezőgépek
- Baba - mama
- Film, zene
- Könyv

TOP termékek

Dekorációs ajtólap juhar, tölgy, éger, bukk, cely 5 000 Ft

Jámin pehelygarnitúra natur színű 25 990 Ft

A legnagyobb online pláza már mobilon is!
Töltsd le okostelefonjára!
Letöltés

Kiemelt ajánlatok | **Újdonságok** | **Népszerű termékek** | **Mai ajánlatok** | **Akciós termékek**

"Beker" csuklópánt 210 Ft

Jámin pehelygarnitúra natur színű, 135x200+68x88+38x48 25 990 Ft

Gyereklülés hátsó csomagtartóra ELIBAS 8 990 Ft

Monster High: Ghoulia Yelps pizsama party baba 2 890 Ft

5. ábra: www.grando.hu

Online áruházak feladatai

Az alábbi kategóriák közül az egyes weboldalak egy vagy több feladatot is ellátnak.¹⁰

- Céges információ megjelenítése online katalógusként
- Termékekre vagy szolgáltatásokra irányuló rendelések felvétele
- Szolgáltatások vagy digitális termékek értékesítése
- Többletérték hozzáadása termékekhez vagy szolgáltatásokhoz
- Költségcsökkentés

Az online áruházak a fenti kategóriák közül akár több kategóriának is megfelelhet. A kategóriák részletes vizsgálatával, megértésével pontosabbá tehetjük boltunkról alkotott elképzeléseinket, hogy vállalatunk sikere megalapozott legyen.

Céges információ megjelenítése online katalógusként

A 1990-es évek elején szinte minden üzleti weboldal pusztán online katalógus vagy értékesítési eszköz szerepét töltötte be.

Mind a mai napig ez az üzleti weboldalak leggyakoribb formája. Akár első internetes próbálkozásként, akár olcsó reklámozási lehetőségként kezelik, sok cég számára van létjogosultsága egy ilyen típusú oldal üzemeltetésének.

A brochureware, azaz a nyomtatott katalógus vagy tájékoztató füzet elektronikus változatának tekinthető honlapok a weboldal formájára alakított névjegykártyáról az átfogó marketinginformációkat kínáló oldalakig terjednek. Az ilyen honlap célja és létének pénzügyi oka, hogy a potenciális ügyfelek kapcsolatba léphessenek a céggel. Közvetlenül nem termelnek bevételt, de hozzájárulhatnak a cég által hagyományos úton szerzett bevételekhez.

Egy ilyen honlap kifejlesztése műszaki szempontból nem túl nagy kihívás. A marketing más területeiről ismert problémákkal kell itt is megküzdeni. Az ilyen oldalak esetében leginkább az alábbi hibákat szokták elkövetni:

- Elmulasztják közzétenni a fontos információkat
- Gyenge minőségű megjelenítéssel állnak elő
- Nem válaszolnak a honlap által generált megkeresésekre
- Hagyják a honlapot megöregedni

¹⁰ Luke Welling, Laura Thomson: PHP és MySQL webfejlesztőknek – Hogyan építsünk webáruházat?, Budapest, 2010 Perfect-Pro Kft.

- Elmulasztják nyomon követni az oldal látogatottságát

Termékekre vagy szolgáltatásokra irányuló rendelések felvétele

A lenyűgöző online megjelenés létrehozása után a következő logikai lépés lehetővé tenni ügyfeleinknek az online rendelést.

Az értékesítők pontosan tudják, mekkora annak fontossága, hogy a vásárlókból azonnali döntést csikarjunk ki. Minél több időt hagyunk nekik, hogy átgondolják a vásárlási döntésüket, annál valószínűbb, hogy más üzletekben is szétnéznek, vagy egyszerűen meggondolják magukat. Ha az ügyfelek szeretnék megszerezni termékünket, akkor saját érdekünk, hogy a lehető leggyorsabbá és legegyszerűbbé tegyük a vásárlási folyamatot. Ha arra kényszerítjük őket, hogy felálljanak számítógépükről, és tárcsázzanak egy telefonszámot, vagy keressék fel üzletünket, akkor megakasztjuk a folyamatot. Ha online hirdetésünk meggyőzte a fogyasztókat, hogy vásároljanak, akkor engedni kell, hogy azonnal, honlapunk bezárása előtt megtehessék azt.

Az online rendelésfelvétel rengeteg cég számára előnyt jelenthet. Minden vállalkozás megrendeléseket szeretne kapni. Az online rendelésfelvétel lehetősége növelheti értékesítéseinket vagy csökkentheti értékesítőink leterheltségét. Az online rendelésfelvételhez szükséges környezet természetesen költségekkel jár: dinamikus honlap kifejlesztése, a fizetési módok megszervezése és az ügyfélszolgálat mind pénzbe kerül.

Az online értékesítés egyik legvonzóbb jellemvonása, hogy ezen költségek nagy része pontosan ugyanannyi lesz ezer és egymillió rendelés esetén is. Az elfogadható költségszinthez éppen ezért megfelelő mennyiségben értékesíthető termékekre vagy szolgáltatásra van szükség. Mielőtt nagyon belelélnénk magunkat az elektronikus kereskedelem gondolatába, próbáljuk végiggondolni, hogy termékeink alkalmasak-e online értékesítésre!

Interneten keresztül leginkább olyan termékeket és szolgáltatásokat vásárolunk, mint a könyvek és a magazinok, a számítógépes szoftverek és hardverek, zene, ruhák, utazás és belépőjegyek (színház, mozi, koncert stb.). Csak azért, mert termékünk nem tartozik ezen kategóriák közé, nem kell elkeserednünk. Ezek a piacok már úgyis telítve vannak jól (és kevésbé jól) ismert márkákkal. Mindazonáltal érdemes lehet végiggondolni, miért pont ezek a legkelendőbb portékák az interneten.

Az ideális e-kereskedelmi termék nem romlandó és egyszerűen szállítható, elég drága ahhoz, hogy elfogadhatóvá tegye a szállítási költségeket, de annyira mégsem drága, hogy a vásárló kényszerítve érezze arra, hogy vásárlás előtt fizikailag is megvizsgálja. A legjobb e-kereskedelmi termékek az árucikkek. Ha valaki

avokádót szeretne venni, minden bizonnyal szeretné kiválasztani (megfogni, megszagolni) az adott darabot. Nem minden avokádó egyforma. Egy könyv, CD vagy szoftver általában tökéletesen megegyezik a többi példánnyal – feltéve persze, hogy ugyanaz a címük. A vásárlók nem igénylik, hogy kezükbe vegyék a beszerezni kívánt darabot.

Az e-kereskedelmi termékeknek emellett az internet-felhasználók érdeklődési körébe kell tartozniuk.

Egyes termékek soha nem fognak megjelenni az e-kereskedelemmel foglalkozó kutatásokban, mégis sikeresen értékesíthetők online. Ha termékünk csak egy szűk piaci réteg számára vonzó, az internet ideális módszer lehet a vásárlók elérésére. Ha lakóhelyünkön csak tíz ember gyűjt 1980-as évekbeli játékbrobotokat, egy ezeket forgalmazó honlap akár sikeres is lehet, ha minden más városban is legalább ugyanennyien gyűjtik őket.

Vannak olyan termékek, amelyek online értékesítése nagy valószínűséggel nem fog számunkra sikert hozni. Az olcsó, romlandó árucikkek, például a zöldség és a gyümölcs nem tűnik túl jó választásnak, bár ez nem riasztja el az embereket, hogy próbálkozzanak – többnyire, persze, sikertelenül. Egyes termék kategóriák tökéletesen megfelelőek arra, hogy brochureware honlapot készítsünk hozzájuk, de online rendelésre nem lesznek alkalmasak. Nagy, drága árucikkek tartoznak ide – például az autók és az ingatlanok –, amelyek rengeteg utánajárást igényelnek a vásárlás előtt, túl drágák ahhoz, hogy szemrevételezés nélkül megrendeljék, és leszállítani sem túl egyszerű őket.

Számtalan akadályt kell leküzdenünk a potenciális vásárlók meggyőzéséhez. Ilyenek lehetnek:

- Megválaszolatlan kérdések
- Bizalomhiány
- Nem egyszerű használhatóság
- Kompatibilitási nehézségek

Ha a felhasználókat ezek bármelyike visszatartja, nagy valószínűséggel vásárlás nélkül fognak távozni honlapunkról.

Szolgáltatások vagy digitális termékek értékesítése

Rengeteg terméket és szolgáltatást értékesítenek az interneten, és szállítják ki azokat a vásárlónak futárszolgálattal. Egy kisebb részüket azonnal, online igénybe veheti a vásárló. Ha valamely szolgáltatást vagy árut hálózaton keresztül is lehet továbbítani, akkor azonnal, emberi beavatkozás nélkül megrendelhető, kifizethető és igénybe vehető. Az így értékesített legegyszerűbb szolgálta-

tás az információ. Az információ gyakran teljesen ingyenes vagy hirdetésekkel finanszírozott. Egyes információkhoz előfizetés vagy tranzakciónkénti fizetés alapján lehet hozzáférni.

A digitális árucikkek közé tartoznak egyebek között az e-könyvek és az elektronikus formátumban (például MP3-ban) lévő zene. A képügynökségek által forgalmazott képek is digitalizálhatók és letölthetők. A számítógépes szoftvereknek sem kell szükségszerűen CD-n vagy DVD-n lenniük, közvetlenül is letölthetők. Az így értékesített szolgáltatások közé tartozik például az internet-hozzáférés vagy a webtárhely is.

Az oldalunkon megrendelt termékek fizikai kiszállítása esetén bizonyos előnyökkel és hátrányokkal kell számolnunk. A fizikai formában létező termék kiszállítása pénzbe kerül. A digitális letöltések szinte teljesen ingyenesek. Ez azt jelenti, hogy ha másolható és digitálisan értékesíthető terméket vagy szolgáltatást kínálunk, annak értékesítési költsége 1 és 1000 darab esetén is ugyanaz lesz. Természetesen ez csak bizonyos korlátok között igaz; megfelelő szintű értékesítés és forgalom esetén többet kell például hardverre és sávszélességre fordítani.

A digitális termékek és szolgáltatások könnyen értékesíthetők impulzív vásárlásként. Ha valaki fizikai formában létező árucikket rendel, a szállítás legalább egy napig tart. A letöltéseket ezzel szemben másodpercben, legfeljebb percben mérjük. Ez azt jelenti, hogy a kereskedők az azonnali teljesítés terhével kénytelenek szembesülni. Ha digitálisan teljesítjük a vásárlást, azonnal kell megtennünk azt. Nem tehetjük meg, hogy manuálisan dolgozzuk fel a rendeléseket, vagy napon belül szétterítjük a csúcsidőszaki terhelést. Az azonnali teljesítésű rendszereknél ezért inkább fennáll a csalás veszélye, és nagyobb terhelést jelentenek a számítógépes erőforrásoknak.

A digitális termékek és szolgáltatások kiválóan alkalmasak az e-kereskedelemre, de nyilvánvalóan korlátozott az így értékesíthető termékek és szolgáltatások köre.

Többlemérték hozzáadása termékekhez vagy szolgáltatásokhoz

Az üzleti weboldalak egyes, igen sikeres részei egyáltalán nem árulnak terméket vagy szolgáltatásokat. Az olyan funkciókat, mint a futárcégek – például az UPS (<http://www.ups.com>) vagy a Fedex (<http://www.fedex.com>) – nyomon követési szolgáltatásai, nem közvetlen profitszerzés céljával fejlesztették ki. A szervezet által kínált, meglévő szolgáltatásokhoz adnak többlemértéket. Ha lehetőséget adunk az ügyfeleknek, hogy nyomon kövessék küldeményük útját, vagy megtekintsék banki egyenlegüket, versenyelőnyhöz juttathatjuk vállalkozásunkat.

A támogatást kínáló fórumok is ebbe a kategóriába tartoznak. Komoly üzleti okai vannak annak, hogy miért érdemes a vásárlók számára olyan fórumot működtetni, ahol megtárgyalhatják a cég termékeivel kapcsolatos hibaelhárítási tippeket. A más vásárlók által javasolt megoldásokkal orvosolhatják az ügyfelek problémái, a külföldi vásárlók telefonköltség nélkül kaphatnak támogatást, és a hivatali órákon kívüli időben is választ adhatnak egymás kérdéseire a fórumozók. A támogatás ilyen formája igen alacsony költség mellett növelheti a fogyasztási elégedettségét.

Költségcsökkentés

Az internet használatának gyakori oka a költségek csökkentése. Megtakarítás származhat az információ online megosztásából, a kommunikáció előmozdításából, a szolgáltatások lecseréléséből, illetve a működés központosításából.

Ha jelen pillanatban igen sok embernek adunk tájékoztatást, minden biznnyal gazdaságosabb módon is megtehetnénk ezt egy weboldalon keresztül. Akár árlistákat, termékkatalógusokat, dokumentált eljárásokat, specifikációkat vagy bármilyen más információt adunk át az érdeklődőknek, biztosan olcsóbban jönnénk ki, ha ugyanezt az interneten tennénk elérhetővé, mint a nyomtatott példányok előállításával és kiküldésével. Különösen igaz ez a rendszeresen változó információkra. Az internet a kommunikáció biztosításával pénzt takaríthat meg számunkra. Ez akár azt jelenti, hogy az ajánlatkéréseket gyorsan szétküldhetjük, és rövid időn belül választ kaphatunk, akár azt, hogy az ügyfelek az ügynőket vagy közvetítőket kikerülve közvetlenül a nagykereskedővel vagy a gyártóval léphetnek kapcsolatba – az eredmény ugyanaz lesz. Az árak csökkennek, a nyereség pedig nő.

A pénzbe kerülő szolgáltatások elektronikus változatra cserélése költségcsökkentése eredményezhet. Bátor példája ennek az Egghead.com esete. A cég úgy döntött, hogy bezárja informatikai bolthálózatát, és az e-kereskedelemre összpontosítja figyelmét.

Bár egy komoly e-kereskedelmi oldal létrehozása nyilvánvalóan pénzbe kerül, egy 80 kiskereskedelmi üzletből álló lánc sokkal nagyobb folyamatos költséggel jár. Ám a meglévő szolgáltatások cseréje kockázatot is hordoz magában. Az a minimum, hogy az internetet nem használó ügyfeleket elveszítjük. Az Egghead.com új vállalkozása nem jött be. A cég az 1998-as dotcom lufi idején zárta be üzleteit, majd a lufi kipukkadásakor, 2001-ben csődvédelmet kért.

A központosítás is költségcsökkenése eredményezhet. Ha több telephellyel rendelkezünk, több helyen kell bérleti díjat és rezsit fizetni, alkalmazottakat foglalkoztatni, és mindenhol raktárkészletet kell fenntartani. Egy internetes

vállalkozás elég, ha egyetlen központi telephellyel bír, a világ bármelyik pontjára elérhető lesz.

Az online áruházak jellemzői

Az online áruházakban, a hagyományos kereskedelemben megszokottól eltérően megfoghatatlan termékekkel találkozunk. Az emberek szívesebben vásárolnak olyan termékeket, amik kézzel foghatóak, felpróbálhatóak, megvizsgálhatóak. Bizalmatlanok a csak látottak alapján vásárolni. Ebből kifolyólag hazánkban manapság továbbra is azokat a termékek rendelését részesítik előnyben a vásárlók, ahol az előbb felsoroltak nem lényegesek a vásárlás szempontjából. Ilyen cikkek a DVD-k, könyvek, egyszerűbb háztartási cikkek, valamint az olyan termékek, amelyeket az ember általában szívesen boltban megvenni.

A termék fizetése történhet utánvétellel, készpénzzel, bankkártyával, vagy átutalással. Hazánkban az utánvétellel történő rendelés jellemző, az átutalással és a bankkártyával történő fizetéssel szemben támasztott bizalmatlanságból adódóan. A szállítás történhet postával vagy futárszolgálattal, de lehetőség van az árut személyesen is átvenni, ha az adott webáruház felkínálja ezt a választást.

A bizalom kiépítése érdekében nézzük meg, milyen jellemzőket kell szem előtt tartani a webáruház létrehozása előtt, melyek azok az alappillérek, amelyekre az online áruház alapjait fektetni kell.

- A webáruház arculata

Az igényes, jól strukturált, áttekinthető, részleteiben kidolgozott arculat megbízhatóságot, profizmust sugall. Természetesen a design nem pusztán csak a megjelenítendő színekből, motívumokból és képekből áll, hanem minden egyes elem gondos megtervezéséből. Kiemelt hangsúlyt kell fektetni az oldal szerkezeti felépítésére, grafikai megjelenítésére, a szövegrészek átgondolt elhelyezésére, ergonómiájára. A könnyen navigálható, a vásárló szempontjából szükséges eszközök könnyű elérése nélkülözhetetlen.

A személyesség hiányának csökkentésére számos webshop alkalmaz úgynevezett „kattancs” videókat, mely mintegy virtuális eladó vezet minket körbe az áruházban.

Az átláthatóság mellett elengedhetetlen a könnyen kezelhető adminisztrációs felület.

- Termékekkel szembeni elvárások

A termékkínálatnak a vevői igényeknek megfelelően változtatva, állandóan frissülnie kell, akár csak a hagyományos üzletek tekintetében.

A célirányos termékkeresés érdekében, egy minden lényeges termékjellemzőre kiterjedő keresőt kell a látogatók rendelkezésére bocsátani, világos, logikus, többszintű kategóriarendszerrel ellátott struktúrába kell szervezni.

A termékkínálat megfelelő elhelyezése mellett lényeges, hogy az árucikkekről milyen részlet gazdag termékleírás található. A kimerítő ismertető kiegészítője lehet például hanganyagok, termékbemutató videók, klipek, demonstrációs anyagok. A tendencia azt mutatja, hogy minél bővebb és igényesebb tájékoztatást kínál egy webáruház, annál inkább megbízhatóságot sugall, és komolyabb esélye van annak, hogy a vevők vásárolni is fognak.

Az akciós, kiemelt termékek jól láthatósága, változatossága és gyakori megújulása felkelti az érdeklődést.

A termékekről írható vevői vélemény megnövelheti a termékek forgalmát.

- Vevőszolgálat

Az elérhetőség biztosítja, hogy bármilyen probléma felmerülése esetén van kihez fordulni, azt tükrözi, hogy az üzlet a vevők érdekeit tartja szem előtt és az üzletkötés érdekében hajlandó időt és figyelmet fordítani rá. Az e-mail cím mellett a telefonos elérhetőség nagyban növeli a bizalmat az online áruház felé. A személytelenség érzetének csökkentése érdekében érdemes az ügyfélszolgálat munkatársairól egy-egy fényképet elhelyezni.

- Üzletkötési folyamat

Nagy hangsúlyt kell fektetni a folyamat kidolgozására. A vásárlás folyamatában elengedhetetlen, hogy a vevő minden szükséges adatot pontosan megadjon, ugyanakkor ez a folyamat a számára kényelmes és egyszerű legyen. (Például a termékek keresésének, kiválasztásának, kosárba helyezésének egyértelműnek, egy mozdulattal végrehajthatónak kell lennie.) A megadott adatok védelme, bizalmas tárolása sarkalatos pontja a folyamatnak.

Egy jó webáruház olyan apróbb kényelmi funkciókat is biztosít a megrendelés adatainak megadása során, mint pl. a szállítási címek automatikus kitöltése a korábban megadott szállítási címek alapján (a felhasználó akár több elmentett cím közül is választhat).

Sarkalatos pont a folyamatban a fizetés kérdése. A lehetőséget kell biztosítani a bankkártyás az utánvételes készpénzes esetleg alternatív fizetési módra (PayPal rendszer).

A vásárlónak mindenképp lehetőséget kell adni a szállítási mód megválasztására. Egy jól működő webáruház kezeli a megrendelés összértéke által adható kedvezményeket, vagy a termékek súlya, mérete alapján számított szállítási költséget.

A vásárlás után sem szabad magára hagyni a felhasználókat: kérésére e-mailben vagy sms-ben tájékoztatnunk kell a megrendelés állapotváltozásairól. Főként műszaki cikkek esetében megfontolandó, hogy a garanciális ügyintézés is lehetővé tegyük a webáruház keretein belül.

2.3 ÖSSZEFOGLALÁS, KÉRDÉSEK

2.3.1 Összefoglalás

Az elektronikus kereskedelem a mai fejlett és dinamikusan fejlődő számítógép-hálózatra támaszkodik, elsősorban az internet nyújtotta lehetőségeket kihasználva. Sikerességét és széles körben való elterjedését a hálózat nyújtotta elérhetőség, naprakészség, egyszerű és gyors kapcsolattartás, ügyintézés alapozta meg. Az e-kereskedelem magában foglalja a termelő/szolgáltató vállalatok közötti, vállalat és fogyasztó, és főképp a fogyasztók közötti kapcsolatokat, és elősegíti az áruk, szolgáltatások, információk kereskedelmét. Megvalósítja az elektronikus úton történő beszerzést és értékesítést.

Számos előnye, mint például az időbeli korlátozottságtól (nyitva tartási idő, kapcsolatfelvétel...) való mentessége, földrajzi határok megszűnése, bőséges választék – mint boltok, mint termékek tekintetében – mellett nem felelkezhetünk meg a hátrányairól (nagyfokú személytelenség, vevőkben kialakuló bizonytalanságérzet) és kockázatairól (adatbiztonsági kérdések, crackerek, szoftverhibák) sem.

Mindezek ismerete szükséges ahhoz, hogy egy jól átgondolt és megfelelően működő on-line áruházat tudjunk létrehozni és üzemeltetni. Tisztában kell lennünk a szerkezeti felépítésével, feladataival, jellemzőivel. Ennek a tudásnak a birtokában ki tudnak teljesedni elképzeléseink, és a részletekig kidolgozott működés és működtetés megalapozza sikereinket.

2.3.2 Önellenőrző kérdések

- Mit nevezünk elektronikus kereskedelemnek?
- Határozza meg az elektronikus kereskedelem előnyeit és nehézségeit!
- Milyen kockázatokkal kell számolnunk, milyen biztonsági intézkedéseket kell tennünk a védelem érdekében?
- Mik az online áruház feladatai?

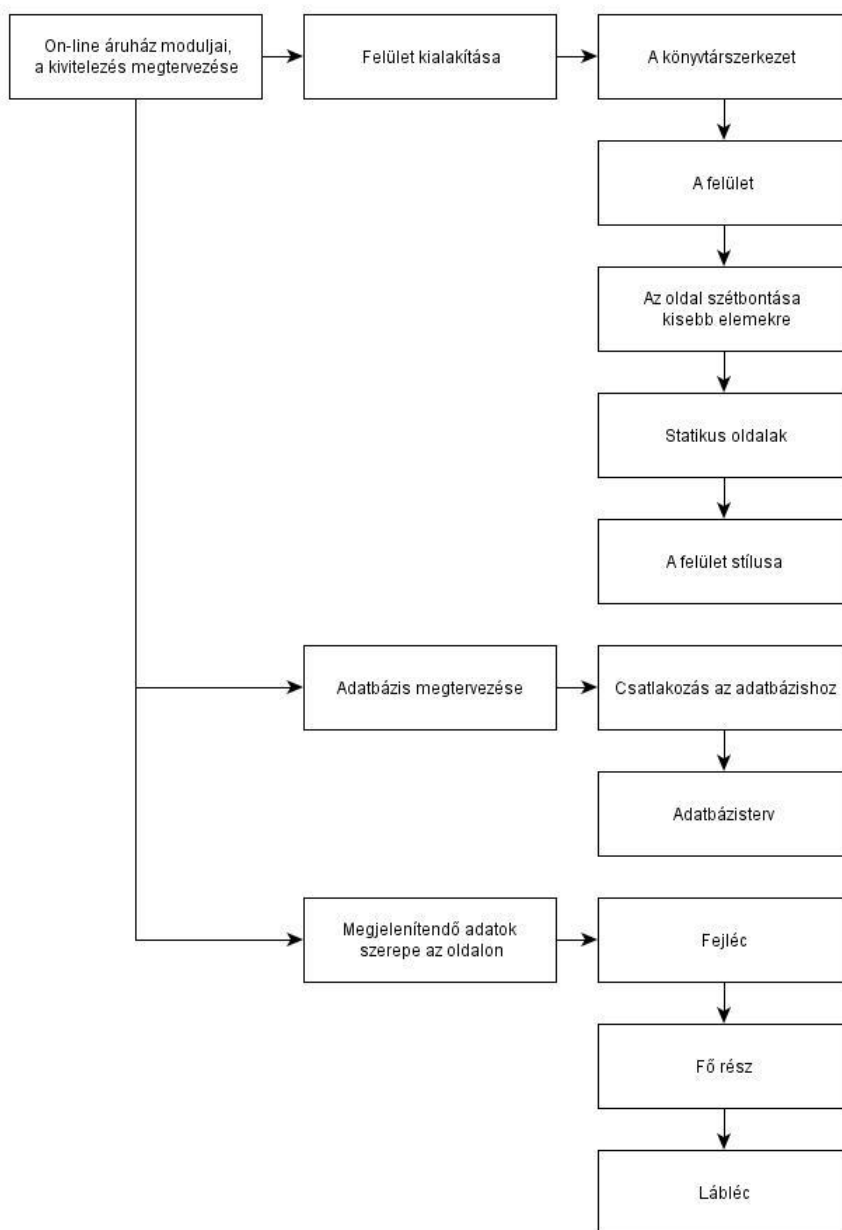
- Fórumot beilleszteni
- Fórum neve: Online bolt típus választás
- (Feladatok/Fórum) Feladat: Beszélje meg a fórumon társaival, hogy kinek mi az elképzelése, konkretizálják, hogy ki milyen típusú (műszaki, ruházati cikkek, élelmiszer...) online boltot szeretne elkészíteni a kurzus végére.
- (Feladatok/Egyetlen állomány feltöltése) Összesítsék az így kapott válaszokat és töltsék fel!

3. EGY ONLINE ÁRUHÁZ MODULJAI, A KIVITELEZÉS MEGTERVEZÉSE

3.1 CÉLKITÚZÉSEK ÉS KOMPETENCIÁK

Ezzel a leckével megkezdjük egy internetes könyváruház kialakítását. Elkészítjük a internetes áruház felületét, amit majd a további leckeken keresztül tovább fejlesztünk. Ezek mellett létrehozuk a webáruház adatbázisát és tábláit. Legvégül átnézzük, hogy melyek azok az adatok, amelyeknek kötelezően meg kell jelennie az oldalon.

3.2 TANANYAG



6. ábra: A lecke áttekintése

3.2.1 Felület kialakítása

A könyvtárszerkezet

A webszerveren a következő könyvtárstruktúrában fogjuk letárolni az oldalt:

- admin
 - pages (adminisztrációs felület oldalai)
- css (stílusok könyvtára)
- images (képek könyvtára)
 - book (könyvek képeinek könyvtára)
- js (javascript állományok)
- pages (a webáruház felületének oldalai)
- config.php (konfigurációs állomány)
- database.php (adatbázis-csatlakozás)
- function.php (függvények, eljárások)
- process.php (műveletek, események feldolgozásai)

A felület

Indulásképpen hozzunk létre egy üres oldalt. Erre megfelel bármilyen szövegszerkesztő vagy akár a Jegyzetömb is. Ajánlatos viszont direkt erre a célra készített programot használnunk, mivel nagyon sok előnyét élvezhetjük. Az alábbi kód például a DreamWeaver¹¹ programban egy új *XHTML 1.0 Transitional* oldal létrehozásánál automatikusan létrejön, ezzel is megkímélve minket a sok kódírástól. Ezeken felül még nagyon sok hasznos lehetőséget találhatunk a programban.

Lássuk a kódot:

```
1<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
  Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
  transitional.dtd">
2<html xmlns="http://www.w3.org/1999/xhtml">
3<head>
4<meta http-equiv="Content-Type"
  content="text/html; charset=utf-8" />
5<title>Untitled Document</title>
```

¹¹ http://hu.wikipedia.org/wiki/Adobe_Dreamweaver

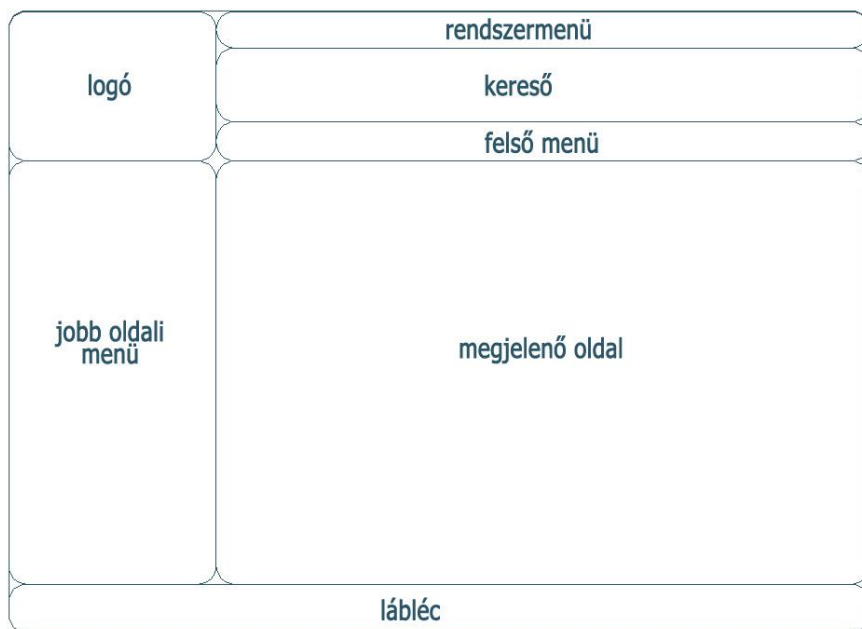
```

6 </head>
7 <body>
8 </body>
9 </html>

```

Az *Untitled Document* név helyett adjuk meg az áruház nevét: *BookStore Webáruház*. Ez a kód jelenleg csak html-kódokat tartalmaz, de a következő fejezetben tovább fogjuk bővíteni PHP-kódokkal, így amikor elmentjük a fájlt, adjuk meg neki az `index.php` nevet. Ez azért is fontos, mivel a webszerver alapértelmezetten az `index.html`, `index.php` állományokat jeleníti meg.

Az oldal 2 hasábos felépítésű lesz, kinézete az alábbi ábrán látható.



7. ábra: Oldal felépítése

Készítsük el az ábrán látható oldalt. Illesszük be a következő kódot a `<body></body>` html-tagek közé.

```

1 <div id="header">
2   <div id="logo">
3   </div>
4   <div id="topBox">
5     <div id="systemMenu">

```

```
6         </div>
7         <div id="searchBox">
8         </div>
9         <div id="topMenu">
10        </div>
11    </div>
12</div>
13<div class="clear"></div>
14<div id="container">
15    <div id="menuColumn">
16    </div>
17    <div id="contentColumn">
18        <div id="contentMenu">
19            <a href=".">Kezdőlap</a>
20        </div>
21        <div id="contentTitle">
22            Hiba
23        </div>
24        <div id="contentBox">
25            <p>
26                A keresett oldal nem található!<br>
27            </p>
28            Ellenőrizze a címet!
29        </div>
30    </div>
31</div>
32<div class="clear"></div>
33    <div id="footer">
34        &copy; 2012 BookStore - All rights
35        reserved
36    </div>
37</div>
```

Mint láthatjuk, nincsen benne táblázat, csak `div` elemek. Minden egyes `div` elem el van látva `id` attribútummal, hogy a stílust tudjuk rá alkalmazni. A 13. és 23. sorban is található egy-egy olyan elem, ami nem látható az oldalon. Ez azért kell, hogy az oldalon lévő igazítást letiltsa a további elemekre.

Az oldal szétbontása kisebb elemekre

Az `index.php` fájlt úgy alakítsuk át, hogy az oldal egyes részeit külső állományokból ágyazzuk be. Készítsük el a következő állományokat úgy, hogy a

megjelölt sorokat lementjük egy-egy állományba, majd használjuk az `include` parancsot a külső állomány beágyazásához.

- `header.php`: 1–13. sor
- `leftMenu.php`: 15–16. sor
- `errorPage.php`: 17–30. sor
- `footer.php`: 32–34. sor

Ez viszont már PHP-s kód, így használjuk a `<? ?>` blokkjeleket. Az `errorPage.php` fájlt fogjuk majd lemásolni mindig, amikor egy új aloldalt fogunk létrehozni. Ez lesz az a fájl, amely akkor fog megjelenni, ha a betölteni kívánt oldal nem található a szerveren. Ezeket az aloldalakat a webszerver gyökérkönyvtárában létrehozott `pages` könyvtárba fogjuk tárolni. Ha ezzel megvagyunk, akkor a következő kódot kell, hogy kapjuk:

```
1 <?
2     include('header.php');
3 ?>
4     <div id="container">
5 <?
6     include('leftMenu.php');
7     include('pages/errorPage.php');
8 ?>
9     </div>
10    <div class="clear"></div>
11 <?
12     include('footer.php');
13 ?>
```

Mivel az egyes paramétereket az oldal a `GET` változóba adja át, ezért figyelniünk kell azt is, hogy megfelelő adatokat kapjunk. Ehhez módosítsuk a 7. sort a következőképpen.

```
1 if (isset($_GET["page"]) && ($_GET["page"] !=
2     ''))
3     if (file_exists('pages/'.$_GET["page"].
4         'Page.php'))
5         include('pages/'.$_GET["page"].'Page.php');
6     else
7         include('pages/errorPage.php');
8 else
9     include('pages/mainPage.php');
```

A fenti kódban az 1. sorban megvizsgáljuk, hogy ki van-e töltve az URL-ben a `page` paraméter. Ha nincs, akkor a 7. sorban lévő kezdőoldal fog betöltődni. A 2. sorban megnézzük, hogy létezik-e az az állomány, amit be akarunk ágyazni.

Ha létezik, akkor a 3. sorban található `include` paranccsal betöltjük a kért oldalt. Amennyiben nincs ilyen fájl, akkor az 5. sorban lévő `errorPage.php` oldalt töltjük be.

A `header.php` fájlban található 3 div, amelyben menüket és a keresést fogjuk megjeleníteni. Ehhez nincs más dolgunk, mint a következő kódokat a megfelelő div belsejébe beilleszteni. A legelső `systemMenu` nevű div-ben a rendszerrel kapcsolatos menüpontokat jelenítjük meg a felhasználó neve mellett.

```
1 <ul>
2 <?
3     if (isset($_SESSION["customerId"]) &&
4         ($_SESSION["customerId"] != '')) {
5 <?>
6     <li>Üdvözöllek, <?=getCustomerName();?>!</li>
7     <li><a href="?page=cart"<?=(($_GET["page"] ==
8         'cart'))?' class="active"': ''?>>Kosár</a></li>
9     <li><a href="?page=logout">Kilépés</a></li>
10 <?
11     } else {
12 <?>
13     <li>Üdvözöllek, Vendég!</li>
14     <li><a href="?page=login"<?=(($_GET["page"] ==
15         'login'))?'
16         class="active"': ''?>>Belépés</a></li>
17     <li><a
18         href="?page=registration"<?=(($_GET["page"] ==
19         'registration'))?'
20         class="active"': ''?>>Regisztráció</a></li>
21 <?
22     }
23 <?>
24 </ul>
```

Itt látható, hogy van egy függvény, amely lekérdezi a belépett felhasználó nevét. Ez a függvény a `function.php`-ben található.

```
1 function getCustomerName($id = '') {
2     if (isset($_SESSION["customerId"]) &&
3         ($_SESSION["customerId"] != '')) {
4         $customerId = ($id !=
5             '')?$id:$_SESSION["customerId"];
```

```

4         $query = mysql_query("SELECT
        customer_name FROM ".$GLOBALS["customerTable"]."
        WHERE customer_id = ".$customerId);
5         if (mysql_num_rows($query) > 0) {
6             $result =
        mysql_fetch_object($query);
7             return $result->customer_name;
8         }
9     }
10    return '';
11 }

```

A 3. sorban, egy rövidített feltételvizsgálattal megvizsgáljuk, hogy lett-e átadva paraméter a függvényének. Ha igen, akkor a megadott azonosítójú felhasználót kérdezzük le, egyébként a jelenleg belépett felhasználó azonosítójával dolgozunk tovább. A 4. sorban megpróbáljuk lekérdezni a felhasználó nevét, majd az 5. sorban megnézzük, hogy a találat nagyobb-e mint 0. Ha igen, akkor a 7. sorban visszaadjuk a megtalált felhasználó nevét, egyébként üres sztringgel térünk vissza.

A másik, `searchBox` nevű `div`-ben a kereséshez szükséges felületet fogjuk megvalósítani.

```

1 <form action="?page=search" method="post">
2   <input type="submit" value="Keresés" />
3   Keresés: <input type="text" name="search"
        id="search" value="" />
4   <select name="field">
5     <option value="title">Címben</option>
6     <option value="subtitle">Alcímben</option>
7     <option value="author">Szerzőben</option>
8     <option value="all">Mindenben</option>
9   </select>
10 </form>

```

A `form`-on belül elhelyezünk egy gombot, egy beviteli mezőt a keresendő szövegnek, és egy lenyíló listát, amelyből kiválaszthatjuk azt, hogy melyik mezőben szeretnénk keresni. Mint látható, a sorrend pont a fordítottja annak ami megjelenik az oldalon. Ennek az az oka, hogy minden egyes elemnek a stílusnál meg van adva a `float: right` jellemző, a jobbra igazítás miatt.

A harmadik `div`-ben, melynek `topMenu` a neve, a webáruházhoz kapcsolatos menüket jelenítjük meg.

```

1 <ul>
2   <li><a href="."<?=(($_GET["page"] == '')?'
   class="active"':'');?>>Kezdőlap</a></li>
3   <li><a href="?page=special"<?=(($_GET["page"]
   == 'special'))?'
   class="active"':'');?>>Akciók</a></li>
4   <li><a href="?page=delivery"<?=(($_GET["page"]
   == 'delivery'))?'
   class="active"':'');?>>Szállítás</a></li>
5   <li><a href="?page=contact"<?=(($_GET["page"]
   == 'contact'))?'
   class="active"':'');?>>Kapcsolat</a></li>
6 </ul>

```

Mint látható, az egyes elemeknél lévő hivatkozásokban, ha az URL-ben található page paraméter megegyezik a kiválasztott menüponttal, akkor alkalmazzuk az adott hivatkozásra az active osztályt.

A leftMenu.php-ben még el kell helyoznünk egy kódot, amely majd az egyes kategóriákat fogja kilistázni. Ehhez nincs más dolgunk, mint az alábbi kódot elhelyezni a menuColumn id-jű div-ben.

```

1 <?
2   generateCategoryList();
3 ?>

```

Ahhoz, hogy a kategóriák is megjelenjenek, a function.php-ben meg kell írunk egy függvényt, amely lekérdezi az összes kategóriát.

```

1 function generateCategoryList() {
2   $query = mysql_query("SELECT * FROM
   ".$GLOBALS["categoryTable"]." ORDER BY
   category_name");
3   if (mysql_num_rows($query) > 0) {
4     print('<ul>');
5     while ($result =
   mysql_fetch_object($query)) {
6       $select = ($result->category_id ==
   $_GET["id"]) && ($_GET["page"] == "bookList")?'
   class="active"':'';
7       print('<li><a
   href="?page=bookList&id='.$result->
   category_id.'".$select.">'.$result->
   category_name.'</a></li>');
8     }
9     print('</ul>');

```

```
10     }  
11 }
```

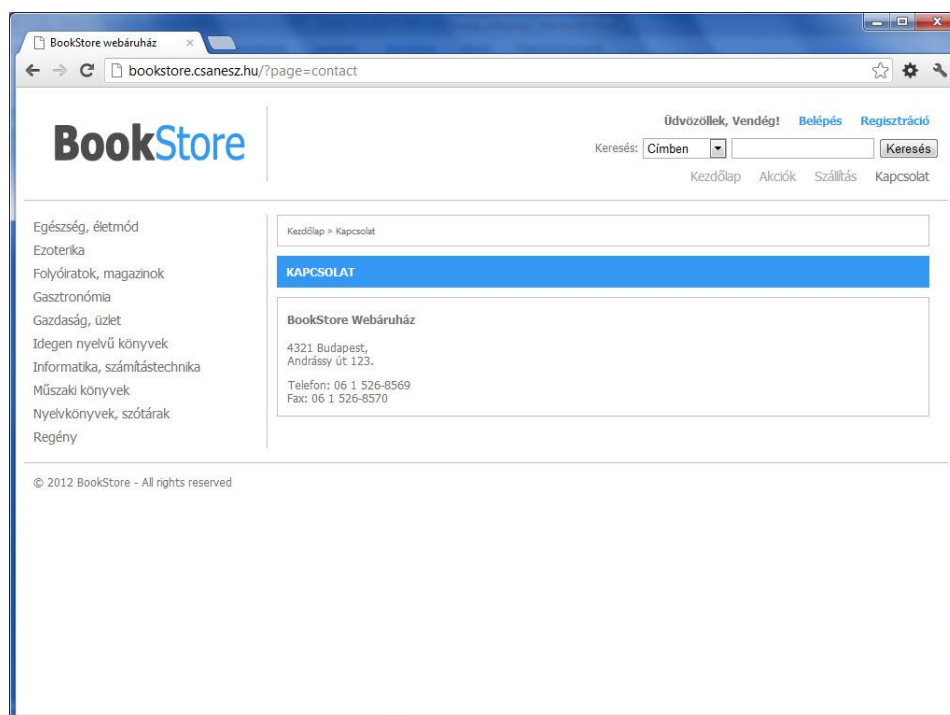
A 2. sorban található a lekérdezés, amelyben név szerint növekvő sorrendben kérdezzük le a kategóriákat. Mint látható, a 3. sorban történik meg annak vizsgálata, hogy van-e kategória feltöltve. Ha nincs, akkor nem jelenítjük meg a listát. A 6. sorban lévő feltételvizsgálat szolgál arra, hogy megjelöljük az éppen kiválasztott kategóriát. Ez ugyanúgy történik, mint a felső részben lévő menüpontoknál. Ha megegyezik a kategória azonosítója az éppen kiválasztott kategória azonosítójával, akkor alkalmazzuk rá az `active` stílus osztályt.

Minden kérés, amelyet az oldalhoz intézünk, az `index.php` állományon keresztül fog végrehajtódni, ezért rögtön a fájl elején indítanunk kell egy munkamenetet – ez fogja tárolni a felhasználó azonosítóját –, és be kell ágyaznunk a szükséges fájlokat. A munkamenetet a `session_start()` utasítással tudjuk elindítani. Amit itt még fontos megjegyezni, hogy előtte nem történhet semmilyen kimenet, ezért is helyeztük el az fájl elején. Egyelőre csak hozzuk létre a fájlokat, és a leckéken előre haladva folyamatosan bővítjük majd az állományok tartalmát.

```
1  <?  
2      session_start();  
3      include('config.php');  
4      include('database.php');  
5      include('function.php');  
6      include('process.php');  
7  ?>
```

Statikus oldalak

A rendszerben van 2 olyan oldal, amelyeknek a tartalma nem változik. Ez a szállítás és kapcsolat oldalak. Ezek a felső menüből érhetőek el.



8. ábra: A kapcsolat oldal

A kapcsolat oldal a pages könyvtárban lévő `contactPage.php` fájlban foglal helyet. Ennek kódja itt található.

```

1      <div id="contentColumn">
2          <div id="contentMenu">
3              <a href=".">Kezdőlap</a> &raquo; Kapcsolat
4          </div>
5          <div id="contentTitle">
6              Kapcsolat
7          </div>
8          <div id="contentBox">
9              <h4>BookStore Webáruház</h4>
10             <p>
11                 4321 Budapest,<br />
12                 Andrássy út 123.
13             </p>
14             <p>

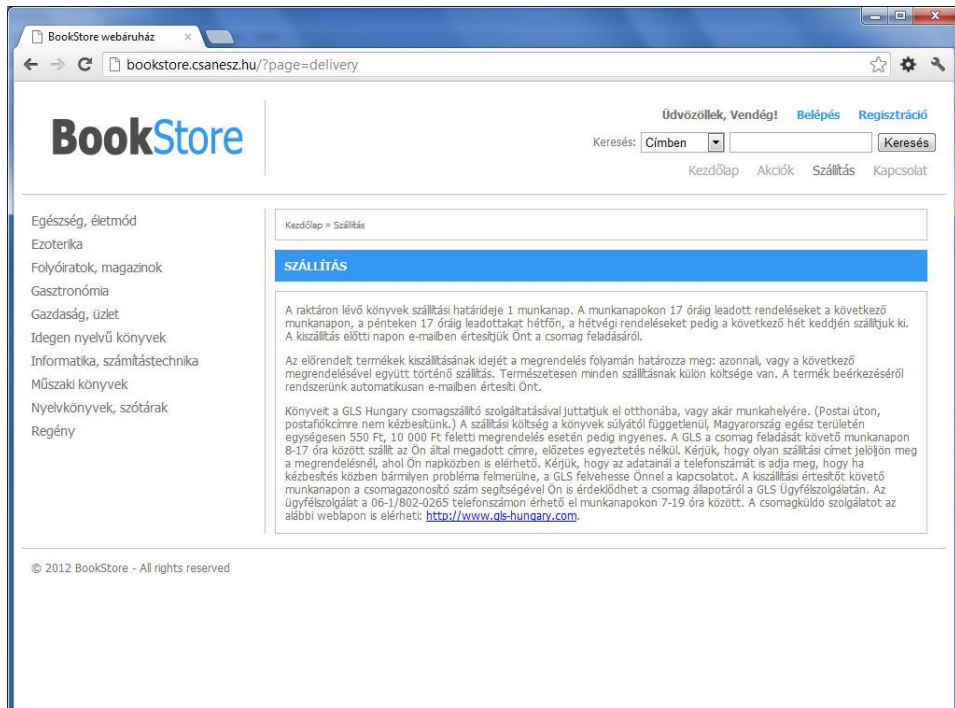
```

```

15         Telefon: 06 1 526-8569<br />
16         Fax: 06 1 526-8570
17     </p>
18 </div>
19 </div>

```

Természetesen ennek az oldalnak a tartalma bármikor átírható, úgy, mint a következő, szállítási adatokat tartalmazó oldal tartalma is.



9. ábra: Szállítás oldal

Ez a `deliveryPage.php` fájlban foglal helyet. Ez a kódjá:

```

1     <div id="contentColumn">
2         <div id="contentMenu">
3             <a href=".">Kezdőlap</a> &raquo; Szállít-
        tás
4         </div>
5         <div id="contentTitle">
6             Szállítás
7         </div>

```

```
8      <div id="contentBox">
9          <p>
10              A raktáron lévő könyvek szállítá-
                si határideje 1 munkanap. A munkanapokon 17 órá-
                ig leadott rendeléseket a következő munkanapon,
                a pénteken 17 óráig leadottakat hétfőn, a hétvé-
                gi rendeléseket pedig a következő hét keddjén
                szállítjuk ki. A kiszállítás előtti napon e-
                mailben értesítjük Önt a csomag feladásáról.
11          </p>
12          <p>
13              Az előrerendelt termékek kiszállí-
                tásának idejét a megrendelés folyamán határozza
                meg: azonnali, vagy a következő megrendelésével
                együtt történő szállítás. Természetesen minden
                szállításnak külön költsége van. A termék beér-
                kezéséről rendszerünk automatikusan e-mailben
                értesíti Önt.
14          </p>
15          <p>
16              Könyveit a GLS Hungary csomag-
                szállító szolgáltatásával juttatjuk el otthoná-
                ba, vagy akár munkahelyére. (Postai úton, posta-
                fiókcímre nem kézbesítünk.) A szállítási költség
                a könyvek súlyától függetlenül, Magyarország
                egész területén egységesen 550 Ft, 10 000 Ft fe-
                letti megrendelés esetén pedig ingyenes.
                A GLS a csomag feladását követő munkanapon 8-17
                óra között szállít az Ön által megadott címre,
                előzetes egyeztetés nélkül. Kérjük, hogy olyan
                szállítási címet jelöljön meg a megrendelésnél,
                ahol Ön napközben is elérhető. Kérjük, hogy az
                adatainál a telefonszámát is adja meg, hogy ha
                kézbesítés közben bármilyen probléma felmerülne,
                a GLS felvehesse Önnel a kapcsolatot. A kiszál-
                lítási értesítőt követő munkanapon a csomagazo-
                nosító szám segítségével Ön érdeklődhet a csomag
                állapotáról a GLS Ügyfélszolgálatán. Az ügyfél-
                szolgálat a 06-1/802-0265 telefonszámon érhető
                el munkanapokon 7-19 óra között. A csomagküldő
                szolgálatot az alábbi weblapon is elérheti: <a
                href="http://www.gls-hungary.com"
                target="_blank">http://www.gls-hungary.com</a>.
17          </p>
```

```
18     </div>
19 </div>
```

A felület stílusa

Ahhoz hogy megkapjuk azt a szerkezetet, amelyet az 1. ábrán látunk, létre kell hoznunk egy stíluslapot. Ezt egy külső állományban fogjuk elkészíteni. Az `index.php`-ben található `</head>` rész elé illesszük be a stíluslap meghívását, ami a webszerver `css` könyvtárában található `style.css` fájlra hivatkozik.

```
1 <link href="css/style.css" type="text/css"
  rel="stylesheet" />
```

A stíluslap tartalma itt található.

```
1 body {
2     padding: 0;
3     margin: 0;
4     font-family: Tahoma, Geneva, sans-serif;
5     font-size: 12px;
6     color: #717070;
7 }
8
9 #header {
10     width: 970px;
11     height: 80px;
12     margin: 0 auto;
13     padding: 20px 10px;
14     border-bottom: 1px solid #c0c0c0;
15 }
16
17 #header #logo {
18     float: left;
19     width: 250px;
20     background-image: url(../images/logo.png);
21     background-position: center center;
22     background-repeat: no-repeat;
23     border-right: 1px solid #c0c0c0;
24     height: 80px;
25 }
26
27 #header #topBox {
28     float: left;
29     width: 719px;
30 }
```

```
31
32 #header #topBox #systemMenu {
33   height: 30px;
34 }
35
36 #header #topBox #systemMenu ul {
37   float: right;
38   padding: 0;
39   margin: 0;
40   list-style-type: none;
41 }
42
43 #header #topBox #systemMenu ul li {
44   float: left;
45   padding: 0 10px;
46   line-height: 30px;
47   font-weight: bold;
48   color: #717070;
49 }
50
51 #header #topBox #systemMenu ul li a.active {
52   color: #717070;
53 }
54
55 #header #topBox #systemMenu ul li a {
56   color: #3598f3;
57   text-decoration: none;
58 }
59
60 #header #topBox #systemMenu ul li a:hover {
61   color: #717070;
62   text-decoration: none;
63 }
64
65 #header #topBox #searchBox {
66   height: 26px;
67   padding-top: 4px;
68   text-align: right;
69   line-height: 20px;
70 }
71
72 #header #topBox #searchBox input {
73   float: right;
74   margin: 0 0 0 5px;
```

```
75 }
76
77 #header #topBox #searchBox select {
78     float: right;
79     margin: 0 0 0 5px;
80     height: 22px;
81 }
82
83 #header #topBox #topMenu {
84     height: 30px;
85 }
86
87 #header #topBox #topMenu ul {
88     float: right;
89     padding: 0;
90     margin: 0;
91     list-style-type: none;
92 }
93
94 #header #topBox #topMenu ul li {
95     float: left;
96     padding: 0 10px;
97     line-height: 30px;
98     font-size: 14px;
99     color: #717070;
100 }
101
102 #header #topBox #topMenu ul li a.active {
103     color: #717070;
104 }
105
106 #header #topBox #topMenu ul li a {
107     color: #a6a6a6;
108     text-decoration: none;
109 }
110
111 #header #topBox #topMenu ul li a:hover {
112     color: #717070;
113     text-decoration: none;
114 }
115
116 #container {
117     width: 990px;
118     margin: auto auto;
```

```
119     overflow: hidden;
120 }
121
122 #container #menuColumn {
123     float: left;
124     width: 240px;
125     padding: 0 10px;
126     margin: 15px 0;
127     margin-bottom: -9999px;
128     padding-bottom: 9999px;
129 }
130
131 #container #menuColumn ul {
132     padding: 0px;
133     margin: 0px;
134     list-style-type: none;
135 }
136
137 #container #menuColumn ul li {
138     font-size: 14px;
139     color: #717070;
140     line-height: 25px;
141 }
142
143 #container #menuColumn ul li a.active {
144     color: #3598f3;
145     font-weight: bold;
146 }
147
148 #container #menuColumn ul li a {
149     color: #717070;
150     text-decoration: none;
151 }
152
153 #container #menuColumn ul li a:hover {
154     color: #3598f3;
155     text-decoration: underline;
156 }
157
158 #container #contentColumn {
159     float: left;
160     width: 699px;
161     padding: 0 10px;
162     margin: 15px 0;
```

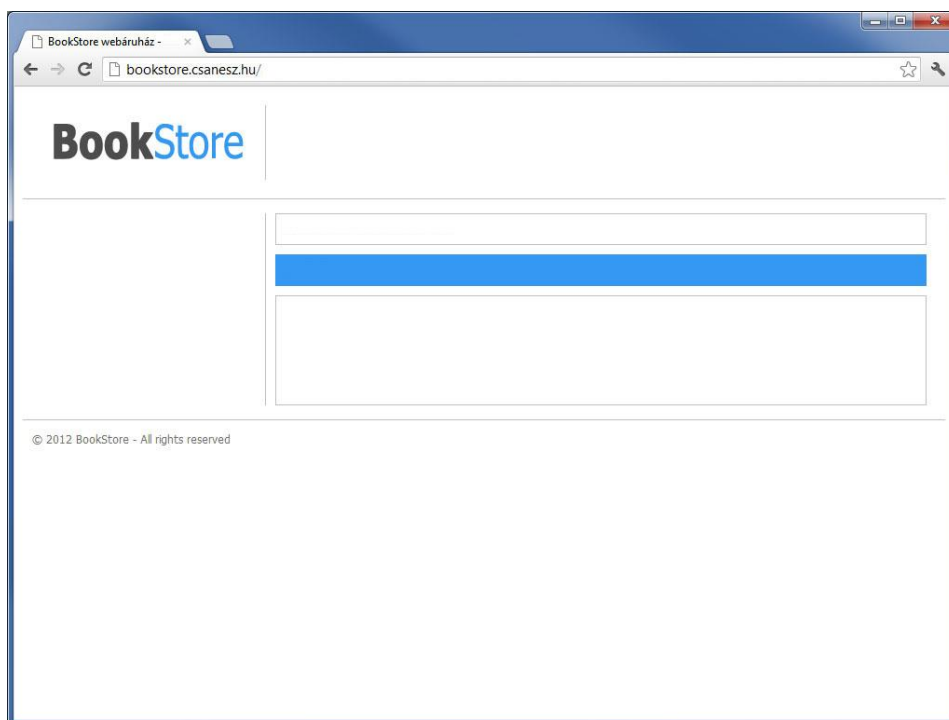
```
163     border-left: 1px solid #c0c0c0;
164     margin-bottom: -9999px;
165     padding-bottom: 9999px;
166 }
167
168 #container #contentColumn #contentMenu {
169     border: 1px solid #c0c0c0;
170     font-size: 10px;
171     color: #717070;
172     padding: 10px;
173     margin: 0 0 10px 0;
174 }
175
176 #container #contentColumn #contentMenu a {
177     color: #717070;
178     text-decoration: none;
179 }
180
181 #container #contentColumn #contentMenu a:hover
182 {
183     color: #3598f3;
184     text-decoration: none;
185 }
186
187 #container #contentColumn #contentTitle {
188     padding: 10px;
189     margin: 0 0 10px 0;
190     background-color: #3598f3;
191     color: #ffffff;
192     font-weight: bold;
193     text-transform: uppercase;
194 }
195
196 #container #contentColumn #contentBox {
197     border: 1px solid #c0c0c0;
198     padding: 0 10px;
199     margin: 10px 0 0 0;
200     color: #717070;
201 }
202
203 #container #contentColumn #contentBox .error {
204     text-align: center;
205     color: #cc0000;
206 }
```

```
206
207 #container #contentColumn #contentBox .message
208 {
209     text-align: center;
209     color: #00cc00;
210 }
211
212 #container #contentColumn #contentBox .require
213 {
214     color: #cc0000;
215 }
216
217 #container #contentColumn #contentBox
218 #cartTable {
219     margin: 10px 0;
219     border-collapse: collapse;
219     font-size: 12px;
220 }
221
222 #container #contentColumn #contentBox
223 #cartTable th {
224     background-color: #d9d9d9;
224     border: 1px solid #c0c0c0;
224     text-align: left;
225 }
226
227
228 #container #contentColumn #contentBox
229 #cartTable td {
230     border: 1px solid #c0c0c0;
231 }
232
233 #container #contentColumn #contentBox table {
234     padding: 10px 0;
235 }
236
237 #container #contentColumn #contentBox table in-
238 put {
239     width: 200px;
240 }
241
242 #container #contentColumn #contentBox table
243 textarea {
244     width: 200px;
245 }
```

```
243
244 #container #contentColumn #contentBox ul {
245     padding: 0;
246     margin: 0;
247     list-style-type: none;
248 }
249
250 #container #contentColumn #contentBox ul li {
251     padding: 10px 0;
252     border-bottom: 1px solid #c0c0c0;
253 }
254
255 #container #contentColumn #contentBox ul li p {
256     padding: 0;
257     margin: 0;
258 }
259
260 #container #contentColumn #contentBox p img {
261     padding: 0 10px 10px 0;
262 }
263
264 #container #contentColumn #contentBox
265 .bookTitle {
266     font-size: 16px;
267     color: #717070;
268     font-style: italic;
269     display: block;
270 }
271
272 #container #contentColumn #contentBox
273 .bookTitle a {
274     color: #717070;
275     text-decoration: none;
276 }
277
278 #container #contentColumn #contentBox
279 .bookTitle a:hover {
280     color: #717070;
281     text-decoration: none;
282 }
283
284 #container #contentColumn #contentBox
285 .bookTitle2 {
286     font-size: 12px;
```

```
283     color: #717070;
284     padding: 0 0 10px 0;
285     display: block;
286 }
287
288 #container #contentColumn #contentBox
289 .bookPrice {
290     color: #3598f3;
291     font-weight: bold;
292     font-size: 14px;
293     padding: 10px 0;
294     display: block;
295 }
296 #container #contentColumn #contentBox
297 .bookPrice del {
298     font-style: italic;
299 }
300 #container #contentColumn #contentBox
301 .bookPrice .special {
302     color: #cc0000;
303 }
304 #footer {
305     width: 970px;
306     height: 40px;
307     padding: 0 10px;
308     margin: 15px auto;
309     line-height: 40px;
310     border-top: 1px solid #c0c0c0;
311     color: #717070;
312 }
313
314 .clear {
315     clear: both;
316 }
```

Megfigyelhetjük, hogy majdnem minden elemnél alkalmazva van a `float` tulajdonág, amivel balra igazítjuk az egyes dobozokat, majd a `width` és `height` értékekkel beállítjuk a méreteit. Itt már be vannak állítva a betűstílusok és a különböző stílusjegyek.



10. ábra: Az oldal kinézete a stílusokkal

A megjelenő könyveket a listában egy vékony vízszintes vonallal válasszuk el. Ezt is stíusból adjuk meg, úgy, hogy a lista minden egyes elemére alkalmazzuk az alsó szegélyt (`border-bottom`). Az utolsó elemnél a jQuery JavaScript segédkönyvtárat használjuk, amivel könnyen meg tudjuk adni, hogy bizonyos esetekben végrehajtható valami. Ehhez illesszük be a következő sorokat az `index.php` fájl `<link>` kezdetű sora elé.

```
1 <script src="js/jquery.js"
  type="text/javascript"></script>
2 <script type="text/javascript">
3 $(document).ready(function() {
4   $('#contentBox ul li:last-
    child').css('border', '0px solid #c0c0c0');
5 });
6 </script>
```

Az 1. sorban beimportáljuk a jQuery segédkönyvtárat. Fontos, hogy sorrendben ez legyen az első, hogy az utána következő szkriptekben már el tudjuk

érni az általa használható függvényeket. A 3. sorban készítünk egy függvényt: `$(document).ready`, amely akkor fog végrehajtódni, amikor az oldal már betöltődött. Ekkor fog végrehajtódni a 4. sorban lévő kód, amely a `contentBox` id-jű elemen belüli listának egy elemére hivatkozik, ezen belül is az utolsó gyerek (`:last-child`) elemre. Ezen elem stílusában állítjuk be az elem szegélyét (`border`) 0 pixelre, ezzel megvalósítva, hogy a keret az utolsó elemnél már ne jelenjen meg. Ezzel már úgy-ahogy elkészültünk a kinézettel.

3.2.2 Adatbázis megtervezése

Ahhoz, hogy az oldalunkon megjelenjenek az egyes könyvek adatai, létre kell hoznunk egy adatbázist, melynek tábláiban letároljuk a különböző adatokat.

Sokszor lehet találkozni a WAMP¹² és LAMP¹³ rövidítésekkel. Ebből is adódik, hogy az adatbázis szerver kiválasztásánál a MySQL-re esett a választás. A PHP-ben több adatbázis-kezelőhöz is vannak előre megírt függvények, melyek az adott adatbázis-kezelő nevével kezdődik például: `mysql_connect`, `pgsql_dbname`, `oci_fetch_array`.

Csatlakozás az adatbázishoz

A `config.php` fájl tartalmazza az adatbázishoz való csatlakozási adatokat, a karakterkódolást, a táblaneveket. Ezt az állományt még bővíteni fogjuk különböző adatokkal. A kódja itt található.

```
1 <?
2 $dbHost = 'localhost';
3 $dbUser = 'root';
4 $dbPass = '';
5 $dbDatabase = 'malaj';
6 $dbCharset = 'utf8';
7
8 $userTable = '`user`';
9 $customerTable = '`customer`';
10 $categoryTable = '`category`';
11 $bookTable = '`book`';
12 $cartTable = '`cart`';
13 $orderTable = '`order`';
14 $orderListTable = 'orderlist';
15 ?>
```

¹² Windows Apache MySQL PHP - <http://hu.wikipedia.org/wiki/WAMP>

¹³ Linux Apache MySQL PHP - <http://hu.wikipedia.org/wiki/LAMP>

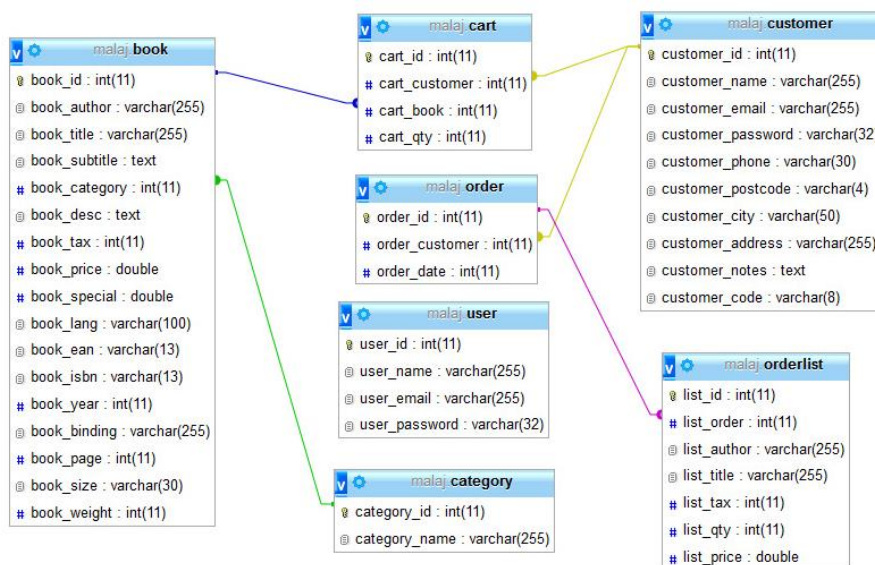
Akik jól megfigyelték a fenti kódot, észrevehettek valami furcsát a táblaneveknél. Két idézőjel között van megadva a tábla neve. Valójában csak az első az idézőjel ('), amit már a szövegeknél is megfigyelhettünk. A másik egy speciális jel, a backtrick (`), amit az `AltGr + 7` billentyűkombinációval érhetünk el. Ennek elengedhetetlen a használata a tábla- és mezőneveknél abban az esetben, ha olyan nevet adunk, ami már egy foglalt szó. Mint például az `order`. Az adatbázis szervertől való csatlakozás a `database.php` fájlban található. Amennyiben nem tudunk csatlakozni az MySQL-szerverhez, vagy nem található az adatbázis, hibajelzést kapunk.

```
1 <?
2     if ($con =
3         mysql_connect($dbHost,$dbUser,$dbPass)
4         if ($db = mysql_select_db($dbDatabase,
5             $con))
6             $ch = mysql_query('SET NAMES
7                 '.$dbCharset);
8         else
9             die('Nem tudom kiválasztani az adatbázis-
10                 zist!');
11     else
12         die('Nem tudok csatlakozni az adatbázis-
13             hoz!');
14 ?>
```

Az 2. sorban látható a `mysql_connect` függvény, amivel csatlakozunk az adatbázishoz. A 3. sorban található `mysql_select_db` utasítással tudjuk kiválasztani az adatbázist, amelyben dolgozni fogunk. A 4. sorban pedig egy kérést küldünk az adatbázis-szervernek, miszerint a lekérdezéseknél a megadott karakterkódolást használja.

Adatbázis-terv

A webáruház működéséhez összesen 7 táblát használunk, amelynek adatbázis-terve lentebb látható.



11. ábra: Az adatbázis terv

A felhasználókat 2 táblában is tároljuk, persze nem párhuzamosan, hanem külön-külön. A `user` táblában lesznek az adminisztrátori felhasználók, a `customer` táblában pedig azokat a felhasználókat rögzítjük, akik a webáruházban regisztrálnak a vásárláshoz. Természetesen a vásárlóknál rögzítjük a táblában a szállítási információkat is. A jelszavak titkosítva vannak letárolva, de erről majd egy későbbi fejezetben lesz szó. A belépéshez nem használunk külön login nevet, hanem a felhasználó e-mail címe lesz felhasználva az azonosításra.

A webruházban található könyveket a `book` táblába tároljuk. A szerzőt, könyvcímet, csoportot, áfatartalmat, akciós árat, nem akciós árat, és a könyv egyéb tulajdonságát ebben a táblában rögzítünk. Az árak tekintetében egy kis furfangosság van a táblában. Két mező szerepel az árak tárolásához, a `book_price` és a `book_special`. Amennyiben egy könyv nem akciós, a `book_price` mezőben található az ára, a másik mező üres. Ha akciós ára van egy könyvnek, akkor az eredeti ára átkerül a `book_special` mezőbe, és a régi ára felülíródik az új árral. Erre azért van szükség, hogy a könyvek listázásakor beállítható rendezésnél az áraknál a megfelelő árakat vegye figyelembe.

Az egyes kategóriához, amelyekbe be lehet sorolni a könyvet, a `category` táblát fogjuk használni. Ebben csak egy elsődleges kulcs és egy név szerepel. Ha szeretnénk többszintű menüstruktúrát létrehozni, akkor célszerű egy harmadik mezőt is bevezetni (pl.: szülő vagy `book_parent`), amelyben

azt tároljuk le, hogy melyik kategóriaazonosító alá van rendelve az adott kategória.

A kosár megvalósítása is adatbázistáblában történik. Igaz, megoldható lenne munkamenettel vagy sütivel is, de mivel ezek a böngésző bezárásával megszűnhetnek, az adatbázis értéke megmarad. Ehhez a `cart` táblát használjuk. Itt valójában azt jegyezzük fel, hogy melyik vevő hány darabot vásárolt az adott könyvből.

Az utolsó két táblában a megrendeléseket fogjuk nyilvántartani. A redundancia kizárására használunk két táblát. Az egyes megrendelt könyveket az `orderlist` táblában fogjuk letárolni. Itt feljegyezzük a könyv szerzőjét, címét, áfatartalmát, darabszámát és egységárát. Azért kell letárolnunk még egyszer ezeket az adatokat, mert ha időközben megváltozik a könyvnek az ára, akkor a megrendelésben már nem a megrendelés kori adatok fognak szerepelni. Az `order` táblában az egyes megrendeléshez szükséges adatokat rögzítjük. Jelen esetben csak a megrendelés dátumát tároljuk le, de van lehetőségünk például a megrendeléshez tartozó megjegyzés mező tárolásához is. Természetesen mind a két táblában található 1-1 mező a két tábla összekapcsolására. Ezzel normálformára hozzuk a táblát, a redundanciák és az anomáliák kisküszöbölésére.

3.2.3 Megjelenítendő adatok szerepe az oldalon

Az internetes oldalakon mindig az információknak van fontos szerepe, ezért nem kerülhető el, hogy a mi weboldalunkon is a megfelelő helyen legyenek az adatok.

Fejléc

A bal felső sarokban elhelyezzük a webáruház logóját. Ezzel egy vonalban helyezkedik el a rendszermenü, alatta a kereső és ez alatt a felső menü. A rendszermenüben fog helyet kapni a kosár, a belépés, regisztráció és egyéb a rendszerrel kapcsolatos menü. A keresőben egy beviteli mező fog helyet kapni, és egy lenyíló lista, amiből kiválaszthatjuk, hogy melyik mezőben szeretnénk keresni. A felső menüben fognak helyet kapni az oldallal kapcsolatos menük, úgy mint szállítási információk, akciós lista, stb.

Fő rész

Ez alatt a baloldalon lesz látható a bal oldali menü, itt jelennek meg a termékkategóriák. E mellett lesz az a rész, ahol az egyes oldalak fognak megjelenni, azaz az oldalnak ez a része fog állandóan változni.

Lábléc

Ez alatt teljes oldalszélességben a lábléc fog helyet kapni a cég nevével, és egyéb információkkal.

3.3 ÖSSZEFOGLALÁS, KÉRDÉSEK

3.3.1 Összefoglalás

Ebben a leckében elértük, hogy a webáruház máris kinézzen valahogy. Elkészítettük a vázát, megalkottuk hozzá a stílust. Ha belegondolunk, akkor már csak a tartalmat kell megírni. Az adatbázis szerkezete is ki lett építve, nincs is más dolgunk, mint a következő leckéken végighaladva a tartalmi elemeket elkészíteni, és a megfelelő adatokkal feltölteni az adatbázist.

3.3.2 Önellenőrző kérdések

- Milyen nagyobb összefüggő részekre lehet bontani egy webáruházat?
- Milyen módszereket lehet alkalmazni az adatbázis legjobb kialakításához?
- Mire figyeljünk oda, ha használjuk a `float: right` CSS-tulajdonságot?

👉 (Feladatok/Egyetlen állomány feltöltése) Feladat: Tervezze meg online áruházát! A tervet és az elkészült programkódot töltsse fel!

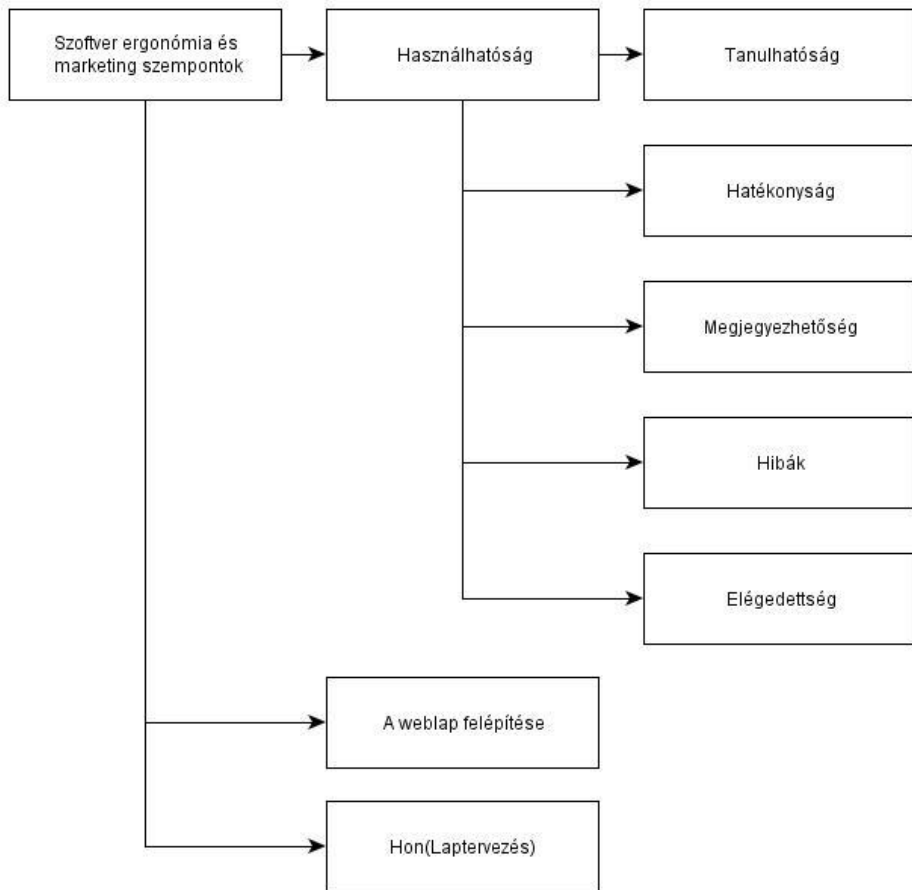
4. SZOFTVERERGONÓMIAI ÉS MARKETINGSZEMPONTOK

4.1 CÉLKITÚZÉSEK ÉS KOMPETENCIÁK

A lecke áttanulmányozása után megismeri a szoftverergonómia fogalmát, célját. Ismeretet szerez, hogy milyen főbb szempontok szerint kell egy weblapot felépíteni, mire kell odafigyelni, és melyek a gyakran előforduló hibák.

4.2 TANANYAG

A szoftverergonómia a számítógépes rendszereket alakítja az ember számára megfelelővé, egyszerűbben, könnyebben használhatóvá, kényelmesen kezelhetővé. Mind funkcionálisan, mind használhatóságában törekszik arra, hogy a felhasználók igényeit a lehető legegyszerűbben kielégítse. Ezáltal nemcsak kedveltebbé válnak az alkalmazások, időt és sok esetben pénzt takarítanak meg gazdaságosabbá téve a működést. Ha röviden szeretnénk összefoglalni, a szoftverergonómia célja a **használhatóság**, hogy az alkalmazások egyszerűek legyenek, és az ember munkáját hatékonyan segítsék.



12. ábra: A lecke áttekintése

4.2.1 Használhatóság

A használhatóság egy olyan minőségi jellemző, amely azt mutatja, hogy milyen könnyen, egyszerűen alkalmazhatjuk az adott felhasználói felületet, rendszert. Valamint magába foglalja azokat a módszereket is, amelyekkel fejleszthetjük a könnyű használatot.

A használhatóság elég összetett tulajdonság, öt komponens határozza meg:

- Tanulhatóság
- Hatékonyság
- Megjegyezhetőség

- Hibák
- Elégedettség

Tanulhatóság

A tanulhatósággal jellemezzük, hogy az adott rendszer működése mennyire könnyen sajátítható el az első alkalommal. A megismerés az észleléssel kezdődik. Az észlelés folyamata hierarchikus: először a nagyobb, színes, figyelemfelkeltő elemeket vesszük észre, majd lépésekben a többit. Nagyon fontos, hogy a szükséges elemek megfelelően ki legyenek emelve, vonzzák a felhasználó figyelmét. Ezután következik az értelmezés, amikor is jelentés kapcsolódik a látottakhoz. Kialakul az ún. célképzet arra vonatkozóan, hogy mit akarunk csinálni. Ez természetesen változhat, s gyakran változik is (a rendelkezésre álló információk számának növekedése következtében). Ha egyszer sikerült valamilyen módon elérni célunkat, legközelebb is úgy fogunk cselekedni – habár lehetséges, hogy nem az volt a legoptimálisabb megoldás.

Hatékonyság

A tanulási fázis után a következő fontos elem, hogy milyen gyorsan, hatékonyan hajtják végre a feladatokat.

A rendszer hatékonyságát az alábbi kérdésekre adott válaszokkal mérhetjük:

- Elérjük-e a kívánt célt, vagyis használható-e arra, amire készült?
- Eredményesség
- Gyorsaság

Megjegyezhetőség

Kulcsfontosságú tényező, hogy a felhasználók milyen könnyen szereznek jártasságot, miután egy ideig nem használták, majd újra visszatérnek a modellhez. A már korábban megszerzett tudását fel tudja-e használni, valamint az itt szerzett tapasztalatok alkalmazhatóak-e más rendszerek esetében is?

Hibák

Hibák alatt az alábbiakat értjük:

- Mennyi hibát vét a felhasználó használat közben?
- Hányféle és milyen súlyos hibákról van szó?
- Az ejtett hibák könnyen javíthatóak-e?

Egy jól megtervezett alkalmazásban – a megfelelő jelzésekkel – nagyon sok lehetséges hibát ki lehet szűrni. A felhasználó segítségére alkalmazhatunk például tooltip-eket vagy infotip-eket. Ezek szükség esetén azonnal elláthatják a szükséges információkkal. Nagy szerepe van a megfelelő elnevezések használatának is, hiszen nincs is annál nehezebb, mint egy olyan mező kitöltése, melynek neve nem teszi egyértelművé az általa várt tartalmat.

Elégedettség

Az elégedettség mutatja meg, hogy – a felhasználó oldaláról nézve – mennyire kényelmes a rendszer használata. Milyen a felhasználói élmény. Egy alkalmazás esetén ez egyszerű és logikus munkavégzést, egy weboldalnál pedig könnyű és gyors információszerzést jelent. Éppen ezért nagy szerepe van a rendszer tervezése és fejlesztése közben a felhasználók bevonásának.

Mint kiderült, a használhatóság nagyon fontos, több minőségi jellemző együttese, a weben az életben maradás elengedhetetlen eszköze. Például, ha egy weblap átláthatatlan, nehézkes eligazodni rajta, nem olvasható, nem derül ki, hogy mit nyújt, akkor az emberek, felhasználók elhagyják.

Az **e-kereskedelem legfőbb szabálya**, ha a felhasználók nem találják a terméket, akkor megvásárolni sem tudják.

A használhatóság fejlesztéséhez a leghasznosabb a felhasználók tesztelése, amely az alábbiak szerint történhet:

- Vegyünk néhány reprezentatív felhasználót (pl. vásárlót)!
- Kérjük meg őket, hogy végezzenek el néhány minta értékű feladatot a modellel!
- Figyeljük meg a felhasználókat, amikor sikeresek, illetve, amikor nehézségeik támadnak az alkalmazási felülettel.

Fontos, hogy külön-külön teszteljük a felhasználókat, és mindent önmaguk próbáljanak megoldani! Ahhoz, hogy azonosítsuk és kiküszöböljük egy tervezés legfőbb gyengéit, általában elég néhány felhasználó tesztelése. Célravezetőbb sok kis tesztelést futtatni, és mindegyik között átdolgozni a tervezetet, mert leginkább ezzel ragadhatjuk meg az előforduló alaphibákat. Az ismételt tervezés, a minél több verzió- és felületképzés vizsgálata az egyik legjobb módja a felhasználói élmény minőségi fejlesztésének is.

A felhasználói tesztek különböznek a fókuszcsoporthoz szóbeli megkérdezéstől, melynek marketingvizsgálatokban van helye. Az interaktív tervezet kiértékeléséhez a használók feladatmegoldó tevékenységét kell megvizsgálnunk. Ha

arra figyelünk, amit az emberek mondanak, akkor az félrevezethet bennünket. Csakis az számít, az a lényeges, amit valójában tesznek.

A tervezési folyamat minden szintjén megtaláljuk a fejlesztés szerepét. A felhasználói tesztek idejében el kell kezdeni, és fokozatosan végig kell vinni a tervezés folyamatai során.

4.2.2 A weblap felépítése

A javítandó hibák és tévedések a weblap különféle szintjein fordulhatnak elő. Nézzük meg melyek lehetnek ezek:

A honlap (homepage), mely a weblap (website) nyitóoldala, az alap kiindulási helyzet („címlapféleség”);

A belső weboldalak, melyekre a nyitólap fő linkjei révén a lapban mélységi-leg továbbkattinthatunk (angolul webpage: termékoldalak, kategóriaoldalak, segítőoldalak stb.);

A rajtuk látható-hallható tartalom (szöveg, kép, animáció, film, hang);

A láthatatlan információs architektúra, vagyis az oldalak menüszerkezetes, belső kereséses, navigációs egybefogása komplett weblappá.

Külön kell szólni a navigációs vagy utasítási eszközökről: a menük, a gördítősávok, a parancsgombok, a szövegbeviteli mezők.

A kevésbé elmélyült felhasználók talán nem mindig tesznek különbséget e szintfogalmak között, mert a (web)lap/honlap/(web)oldal kifejezések pontos értelmével nincsenek tisztában, és – legalábbis magyarul – sokszor szinonimaként kezelik ezeket a szavakat.

A közhiedelemmel ellentétben a weben mindenkit az tölt el jó érzéssel, ha a tartalom minél egyszerűbb, átláthatóbb. Egy szimpla felhasználói felület sohasem lehet unalmas, hiszen a felhasználót az izgalmas tartalommal és a lap mögött álló céggel köti össze! A magasan cizellált, egyedi megoldásokat a felhasználók sokszor észre sem veszik. De ugyanígy a kitalált, bohókás navigációs kifejezéseket is utálják. A felhasználók csakis a céljukat szeretnék elérni. A hipertext eredetileg a könnyű használatra, a technológia és a tartalom tökéletes uralására épült. A felhasználói felületnek teljesen a háttérben kell maradnia, hogy a használók a tennivalókra koncentrálhassanak, a vezető e-kereskedelmi lapok ezt mára jól megtanulták.

El kell hagynunk a fölösleges részeket, hogy az olvasók a fontosabb részekre koncentrálhassanak. Bármely felhasználói felület megtervezésekor javasolható vezérelni, hogy az adott oldal összes elemét próbaképpen külön-külön hagy-

juk el, és ha valamelyikről kiderül, hogy nem feltétlenül kell, rövid úton szabaduljunk meg tőle! Az egyszerűség mindig célravezetőbb.

4.2.1 (Hon) Laptervezés

Az oldaltervezés általában azért áll a figyelem középpontjában, mert a jelenlegi böngészők egyszerre csak egy oldalt tudnak megjeleníteni – maga a teljes webhely egyben soha nem látható a képernyőn. Ergonómiai szempontból azonban az absztraktabb laptervezés – és fő belépőegysége, a nyitóoldal, honlap (homepage) – a nagyobb kihívás, sőt, általában fontosabb is, mint az oldaltervezés.

A honlap (nyitólap) alapelemei: név, logó, navigáció, kereső.

Amint a felhasználók egy weblapra jutnak, rögtön tudni vélik, mire lehet azt használni. Nem vesztegetik arra az idejüket, hogy alaposan átböngésszék a lapot. Ezért a laptervezetnek mindenekelőtt egyszerűnek kell lennie, a lehető legkevesebb zavaró tényezővel, nagyon világos információs felépítéssel és megfelelő navigációs eszköztárral. Mivel a honlap/nyitólap az egész weblap vezérelem, másképpen kell megtervezni, mint a további részeket. Igaz, hogy a nyitólap és a belső oldalak stílusának egyeznie kell, de ugyanígy eltérések is szükségesek.

Például, a honlapon nincs értelme a Nyitólap/Honlap (Home) gombnak, ugyanis felesleges és bosszantó rákattintani egy olyan gombra, amely azonnal visszavisz az éppen aktív oldalra! Jó, ha a honlapon nagyobb a logó, és a cég vagy a lap neve feltűnő helyen látható. Egy nyitólapnak elsőként azt a két felhasználói kérdést kell rögtön megválaszolnia, hogy „Hol vagyok?” és „Mire jó ez a lap?” – mindkettőhöz világos és kellően észrevehető névre van szükség. Aki először látogat oda, ránézésre tudja meg, mit kínál a weblap! A visszatérők pedig a navigációs rendszer kapujaként kezelhetik a honlapot.

A híreknek szentelt rész legyen minél rövidebb, hogy annál több hely jusson a navigációs felületnek – e szabály alól természetesen kivételt képeznek a célzottan hírekre szakosodott lapok. Az ilyen lapokat a felhasználók azért látogatják, hogy megtudják a legfrissebb híreket, pletykákat, információkat. De még a híroldalaknak is oda kell figyelniük arra, hogy néhány felhasználó az aktualitásokról szeretne olvasni, míg mások olyan régi cikkeket keresnek, amelyeket már levettek a lapról.

A legtöbb weblapnak jól működő keresőrendszerre van szüksége, sok felhasználó inkább rákeres valamire, mint hogy hivatkozásokat követve kelljen megtalálnia.

Egy honlapnak (nyitóoldalnak) három fő vonása legyen: a lap központi tartalmait felsoroló lista (ezt nevezzük navigációnak), a legfontosabb hírek és promóciók rövid ismertetése, és egy kereső. A tartalomjegyzéket nem lehet az aktuális ajánlatoknál kisebb helyre zsúfolni.

Milyen széles monitorra tervezzük a webhelyet?

A weblap egészéről elmondható, hogy általában ne tervezzünk semmiféle szabványos szélességre, ha lehet, válasszunk olyan megjelenítést, amely sok különböző méretű monitoron működni fog. Nemcsak azért, mert a felhasználók különböző méretű és felbontású gépekkel rendelkeznek, hanem azért is, mert az ablakokat nem feltétlenül nagyítják fel a monitor teljes méretére. Alkalmazkodó, különböző monitorméretekhez jól illeszkedő, felbontástól független weblapot kell készíteni. Weboldalainkat tervezéskor 1024×768 pixelre optimalizáljuk, viszont rugalmas elrendezést alkalmazzunk, amely 800×600 képponttól 1280×1024-ig bármely felbontáshoz igazodik. Az optimalizálásnak három fő területen kell érvényesülnie, akármilyen felbontást veszünk alapul:

- elsőre látható részek;
- olvashatóság;
- külsín.

A tájékoztató megisméltése a belső oldalakon

Egy honlap legszembetűnőbb eleme a cég vagy lap neve legyen! Ettől még nem kell nagy méretűnek lennie, de mindenképp a bal felső sarokban helyezzük el, ahol könnyű észrevenni (és mivel ez mára normává vált, az ettől eltérő elhelyezés szaktalan és kerülendő). A lapnévnek az összes belső oldalon is szerepelnie kell, hiszen a felhasználó bármely belső oldalra beléphet egy keresőprogramból vagy egy hivatkozáson át. Ugyanakkor a belső oldalak konkrét anyagot tartalmazzanak, ritkábban adjanak általános bemutatkozást, ezt hagyjuk meg a nyitólapnak! A nyitólap mindig egyetlen kattintással elérhető legyen!

Mélylinkelés a belső oldalakra

Sajnos nem kényszeríthetjük a felhasználót arra, hogy feltétlenül a honlapunkon lépjen be a lapra. Az ún. mélylinkelés pontosan abban segíti őt, hogy oda érkezen, ahol fontos információ várja. Egy jó weblap olyan, akár egy ház sok-sok bejáratával: igazi vendégváró hely. Ha csak egyetlen bejáratot hagyunk nyitva, és az összes többi lehetőséget elzárjuk, a felhasználók nem értik, miért is kerültek oda, és hogy ez az oldal miképp szolgálja a céljait. Egy nyitólap soha sem lehet annyira konkrét és ezért ugyanakkora segítség egy probléma megoldásához, mint a belső oldalak.

A navigáció szintjei

A világháló egy navigációs rendszer. A legalapvetőbb felhasználói interakció a kattintás, ezzel az egyetlen technikával hatalmas információs térben mozgunk. A felhasználó kezét irányítani kell, az egyszerű hiperlinkeken túl további navigációs segítséggel is el kell látnunk. A navigációs felületeknek három, a tájékozódást érintő alapvető kérdést kell megválaszolniuk: „Hol vagyok?” „Hol jártam már?”, „Hová mehetek innen?”

A felhasználó éppen aktuális helyzetét két különböző szinten kell bemutatni: a web egészéhez képest, illetve a lap szerkezetéhez képest. Minden oldalon azonosítani kell a lapot, hiszen csak egészében alkothatja a web összetartozó részhalmazát.

Strukturális hibák

Nincs szerkezet

A legszembetűnőbb hiba, amikor a tervezők a lapot egyetlen nagy ingoványnak tekintik az egyes részelemeket összefogó bármiféle szervezőelv nélkül. Általában hírlapok és katalógusra épülő e-kereskedelmi lapok jellemző bűne az átláthatatlanság, természetesen az ilyen webhelyek hamar el is tűnnek a sülylyesztőben.

A keresés és a szerkezet nem integráltak

A felhasználók keresés uralta viselkedésükkel a semmiből esnek be egy-egy oldalra, úgyhogy valószínűleg át kell nézniük az érkezési helyük szomszédságát is, hogy megtalálják pontos céljukat (azaz hogy megtalálják, amit kerestek). Ez persze csak akkor lehet sikeres, ha a lap strukturált.

Az érkezési kategóriaoldalak hiánya

Túlzott polihierarchia

A hálózati világ nagy előnye, hogy ugyanaz az összetevő egyszerre sok helyen megélhet. A felhasználók egy darabig elmerengenek a burjánzó lehetőségek fölött, majd kétségbe esnek attól, hogy rengeteg helyen látják ugyanazt, végül gyorsan elmenekülnek.

A laprészeket vagy önálló mellékleteket szegényesen integráljuk a fő lapba

Nem ajánlatos külön termékmellékleteket, független mikrolapokat készíteni a weben, mert mind „az öröklétnek” készülnek, és idővel megzavarhatják a

fő online stratégiánkat és jelen létünket. Az új termékeket inkább belső laprészekben helyezzük el.

Navigációs hibák

Láthatatlan navigációs opciók

Ha a lapon nem sikerül megmutatni magunkból valamit (ha egyik jellegünk vagy eszközünk nem látszik), az olyan, mintha nem is lenne.

Kontrollálatlan navigációs elemek

Minden, ami mozog és ugrál, elvesz a web használati értékéből.

Inkonzisztens navigáció

A navigáció ne legyen logikai feladvány, a felhasználók azonnal értsék meg! És ne változtassuk oldalról oldalra!

Túl sok navigációs technika

Nagyon sok változat közül válogathatunk, de nem szabad bármit használni.

Egyéni menük

A házilag barkácsolt navigációk, menüopciók, saját címketerminológiák szerencsére eltűnőben vannak.

Linkek

Háromféle hipertext-hivatkozást használhatunk egy oldalon, melyek a következők.

Beágyazott linkek: aláhúzott szövegek, amelyek azt jelzik, hogy a szöveg fő részében tárgyalt témáról további információ található egy másik oldalon.

Strukturális linkek: a lapszerkezet különböző szintjeire, illetve a hierarchiában az oldal „testvéreire” vagy „gyermekeire” mutatnak. Nagyon fontos, hogy ugyanazok a szerkezeti linkek szerepeljenek minden oldalon, hogy a felhasználó tudja, milyen navigációs lehetőségekre számíthat.

Asszociatív linkek: segítik a felhasználót, hogy az aktuális oldalhoz hasonló vagy ahhoz kapcsolódó további oldalakra jusson el.

A legjobb megoldás a linkeket aláhúzott szöveggént megadni a szokásos linkszínekkel, vagyis legyenek a még nem látogatott linkek kékek, a többi meg lila. Mivel az aláhúzás mára összeforrt a kattinthatósággal semmi olyat ne húzunk alá, amire nem lehet rákattintani.

Lineáris és hierarchikus lapfelépítések

Függetlenül attól, hogy melyik navigációs módszer mellett döntünk, alapján véve mindegyiknek ugyanaz a szerepe: vizuálisan megjeleníti a felhasználó adott helyzetét, valamint a továbbhaladási irány lehetőségeit az információs tér szerkezetéhez viszonyítva. Amennyiben maga a szerkezet kidolgozatlan, nincs az a navigációs rendszer, amely segítene rajta. Ha az információs felépítés nem kielégítő, nem lesz megfelelő a használhatóság sem.

A legtöbb lapnak hierarchikus a felépítése, vagyis, minél tovább haladunk, annál részletesebb információhoz jutunk. Vannak olyan lapok, amelyek a táblázatos felépítést használják, ahol az oldalakat bizonyos tulajdonságok vagy paraméterek alapján osztályozzák. Így például az 1996-os olimpia lapja az oldalakat sportág, adatok és helyszín szerint csoportosította, tehát a felhasználók megnézhatték külön a focival foglalkozó oldalakat, vagy az összes olyan oldalt, amely egy bizonyos városban történt eseményekről szólt.

A lineáris szerkezet ezzel szemben olyan webes alkalmazásoknál jó, amelyek lépések sorozatából állnak. A két legfontosabb szabály a lap felépítésével kapcsolatban, hogy létezzen (vagyis előre eltervezett szerkezet mentén felépüljön: tartalomjegyzék, részek kapcsolódása), és hogy a lapot, az ott található információkat és szolgáltatásokat a felhasználó szemszögéből mutassa (a lap szerkezete a szervezeti felépítés helyett a felhasználó nézőpontját tükrözze).

4.3 ÖSSZEFOGLALÁS, KÉRDÉSEK

4.3.1 Összefoglalás

Ebben a leckében választ kaptunk arra, hogy mi is az a szoftver ergonómia, mi a célja, melyek a használhatóság főbb komponensei. Rávilágítottunk, hogyan érdemes a felhasználói élményt vizsgálni, majd a lehető legjobb használhatóság érdekében megtervezni weblapját az ergonómiát szem előtt tartva. Megismerte a laptervezés legfontosabb elemeit és az előforduló hibákat, melyekre érdemes odafigyelni.

4.3.2 Önellenőrző kérdések

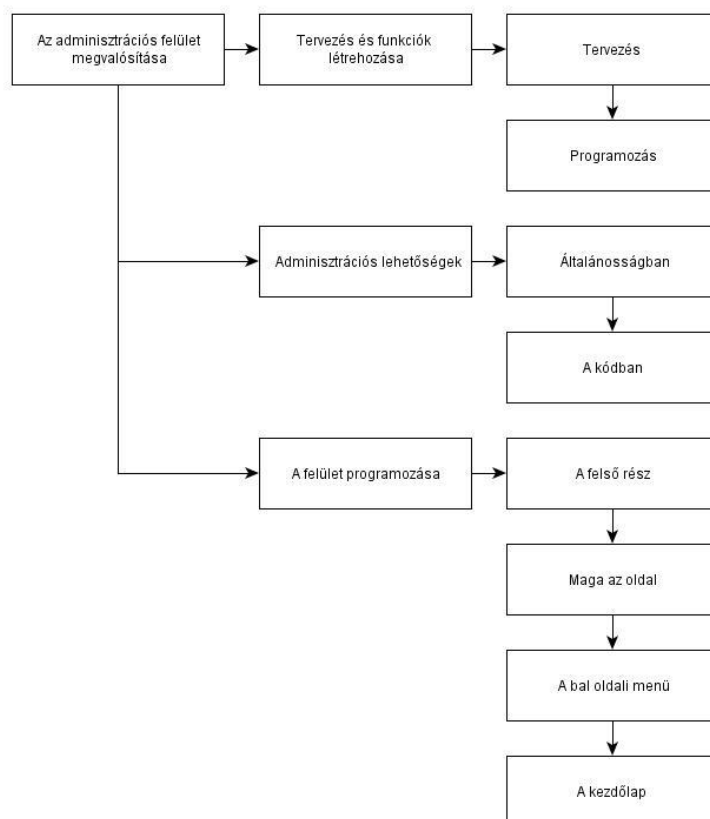
- Mi a szoftver ergonómia fogalma, célja?
- Sorolja fel a használhatóság főbb komponenseit!
- Hogyan történhet a felhasználók tesztelése?
- Mire kell odafigyelni egy laptervezés során?
- Milyen gyakori hibák fordulhatnak elő egy weblapon?

5. AZ ADMINISZTRÁCIÓS FELÜLET MEGVALÓSÍTÁSA

5.1 CÉLKITŰZÉSEK ÉS KOMPETENCIÁK

Az előző leckékben sikerült összeállítani a webáruház szerkezetét és vázát. Most eljött az az idő, hogy létrehozzuk azt a részét az oldalnak azt a részét, ahol a webáruházat lehet irányítani, azaz az adminisztrációs felületet. Itt lehet majd a termékeket, felhasználókat, vevőket, kategóriákat karbantartani.

5.2 TANANYAG



13. ábra: A lecke áttekintése

5.2.1 Tervezés és funkciók létrehozása

Tervezés

Tervezéskor célszerű mindig a legegyszerűbb és átláthatóbb megoldásokra gondolni. Próbáljuk meg szelektálni a kódjainkat.

A stílusoknak készítsünk külön mappát. Itt csak ilyen állományok fognak szerepelni.

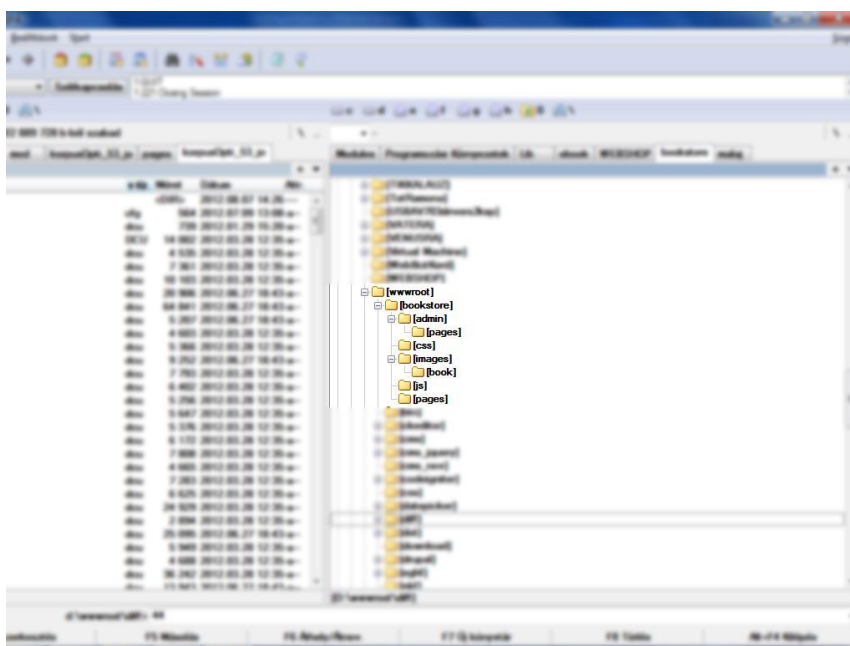
A külső JavaScript-fájloknak is legyen külön könyvtára. Itt fognak helyet foglalni a különböző jQuery-kiegészítések, a képgalériához és egyéb kiegészítő lehetőséghez tartozó JavaScript-állományok.

Magának az adminisztrációs felületnek is célszerű külön könyvtárat létrehozni, és ide csak azokat a fájlokat elhelyezni, amelyek az adminisztrációs részhez tartoznak.

Ha többnyelvű az oldalunk, a nyelveknek is külön létrehozhatunk könyvtárakat, ezzel is elkülönítve egymástól a különböző nyelvű aloldalakat.

A képeket – ha lehet – ne összevissza másoljuk, hanem az ennek létrehozott images könyvtárat vegyük igénybe.

Ha megfogadjuk ezeket a tanácsokat, akkor egy jól átlátható könyvtárstruktúrát tudunk létrehozni.



14. ábra: Könyvtárstruktúra

Programozás

A programozásnál próbáljuk meg törekedni arra, hogy az egyes függvényeket ne többfelé szétszórva, hanem egy közös helyen gyűjtsük össze, egy önálló fájlba. Így ha ezt a fájlt már az oldalunk elején importáljuk, akkor mindenhol elérhetőek lesznek a függvények.

Az adatbázis elérését ugyanígy valósítsuk meg, így nem kell mindenhol csatlakoznunk az adatbázishoz, hanem ebben a fájlban elég megírni a csatlakozáshoz szükséges kódot. Ha létrehozunk egy konfigurációs állományt, amelyben tároljuk az adatbázishoz tartozó adatokat és a táblaneveket, akkor erre tudunk hivatkozni, és egy oldal költöztetése esetén elég itt átírni az adatokat.

5.2.2 Adminisztrációs lehetőségek

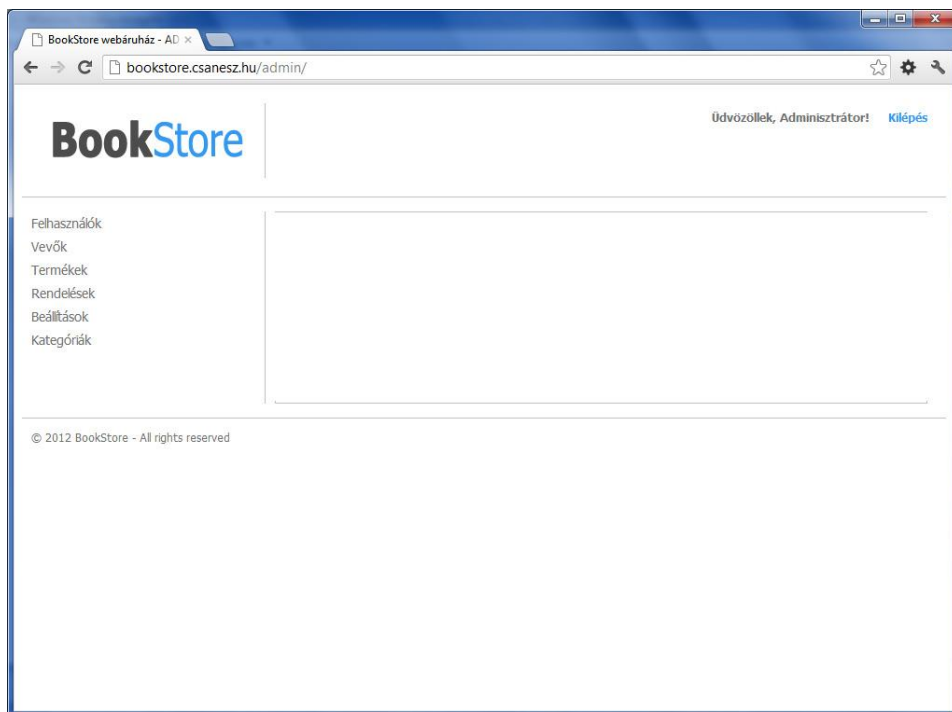
Az adminisztrációs oldal szolgál arra, hogy az oldalon lévő adatokat karbantartsuk.

Általánosságban

Itt van lehetőségünk a vevőtörzs karbantartására. Ez az oldal ugyanúgy működik, mint a vevőknek a regisztrációs oldala, azzal a különbséggel, hogy itt van módunk szerkeszteni és törölni is a vevőket az új vevők hozzáadásán túl.

Ugyanilyen módszerrel tudjuk itt adminisztrálni az adminisztrátorok adatait is, azaz tudunk újat felvenni, régit szerkeszteni és meglévőt törölni.

Nem szabad megfeledkeznünk az egyes kategóriákról sem. Igaz, hogy külső állományokból nem lehet termékkategóriákat feltölteni, de mivel a fejezetek darabszáma nem szokott egyéb esetben sem több száz darabra rúgni, így egyesével van lehetőségünk felvenni őket. Természetesen szerkeszteni és törölni is tudjuk őket.



15. ábra: Az admin felület

Ha már a termékkategóriákat említettük, nem szabad elfelejtenünk a termékeket sem. Sajnos itt sem áll rendelkezésünkre listás feltöltés, de egy újabb fejlesztésnek felírhatjuk magunknak és a későbbiekben megvalósíthatjuk ezt is. A termékeknél fontos odafigyelni, hogy itt már nem csak szöveges adatokat kell feltöltenünk, hanem képeket is. A kisebb helyfoglalás érdekében célszerű a képet már az adott méretre lekicsinyíteni, ezzel is gyorsítva a weboldal megjelenítését. Nem szabad elsiklanunk az egyes rendelések fölött. Mint adminisztrátoroknak jogunk van tudni, hogy ki mikor miből mennyit rendelt. Egy újabb feladat lehet, hogy különböző statisztikát készítsünk az egyes adatok alapján. Először egy listában megjelenítjük a vevőket és a rendelés dátumát, majd kiválasztva egy vevőt a részletes terméklistát kapjuk meg.

A kódban

Ne felejtsük el a kódban a formnak megadni az `enctype` attribútumát, különben a fájlok nem fognak átadódni.

```
1 <form action="" method="post"
  enctype="multipart/form-data">
```

5.2.3 A felület programozása

Az adminisztrációs felület megvalósításához ugyanazt a kinézetet és felépítést használjuk, mint a webáruház oldalán. Ehhez hozzuk létre a szerveren az admin könyvtárat. Ennek a könyvtárnak a felépítése ugyanaz lesz, mint magának a webáruháznak. A legegyszerűbben ezt úgy tudjuk elkészíteni, hogy lemásoljuk a webáruház egyes állományait. Van 2 fájl, amelyek nem változnak, ez a `footer.php` és a `pages` könyvtárban található `errorPage.php`. Ezeket az állományokat értelemszerűen másoljuk is át az `admin` könyvtárba.

A felső rész

A `header.php` fájl tartalma egy kicsit változik az előzőhöz képest. Ennek kódja itt található:

```
1   <div id="header">
2       <div id="logo">
3           &nbsp;
4       </div>
5       <div id="topBox">
6           <div id="systemMenu">
7   <?
8       if (isset($_SESSION["userId"]) &&
9         ($_SESSION["userId"] != '')) {
10      <?>
11          <ul>
12              <li>Üdvözöllek,
13              <?=getUserName();?>!</li>
14              <li><a
15                  href="?page=logout">Kilépés</a></li>
16          </ul>
17      <?>
18          </div>
19      </div>
20  </div>
21  <div class="clear"></div>
```

A 8. sorban látható a felhasználó ellenőrzése a munkamenetben, azaz hogy belépett-e már valaki. Ez úgy történik, hogy a belépéskor a munkamenet `userId` változójában eltároljuk a felhasználó egyedi azonosítóját. Amennyiben már belépett valaki, úgy megjelenítjük a nevét, amit a `getUserName()` függvénnyel kérdezzük le, és megjelenítünk egy *Kilépés* menüpontot, amivel ki tudunk lépni a rendszerből. A gyökérkönyvtárban található `function.php`-ben

helyezzük el az alábbi kódot, ami visszaadja nekünk a bejelentkezett felhasználó nevét.

```
1 function getUsername($id = '') {
2     if (isset($_SESSION["userId"]) &&
3         ($_SESSION["userId"] != '')) {
4         $userId = ($id !=
5             '')?$id:$_SESSION["userId"];
6         $query = mysql_query("SELECT user_name
7             FROM ".$GLOBALS["userTable"]." WHERE user_id =
8             ".$userId);
9         if (mysql_num_rows($query) > 0) {
10             $result = mysql_fetch_object($query);
11             return $result->user_name;
12         }
13     }
14     return 'Ismeretlen';
15 }
```

Láthatjuk, hogy ezt a függvényt paraméterrel és anélkül is meg lehet hívni. Ha paraméter nélkül hívjuk meg, akkor az 1. sorban látható, hogy üres sztringet fog kapni az `id` változó. Mivel ez a függvény azokat a felhasználókat kérdezi le, akik a rendszerrel kapcsolatban végeznek különböző feladatokat, tehát nem a vevőket, így csak akkor adja vissza a felhasználó nevét, ha van bejelentkezett felhasználó egyébként az „*Ismeretlen*” szöveggel tér vissza. A 3. sorban, egy rövidített feltételvizsgálattal, megvizsgáljuk, hogy lett-e átadva paraméter a függvénynek. Ha igen, akkor a megadott azonosítójú felhasználót kérdezzük le, egyébként a jelenleg belépett felhasználó azonosítójával dolgozunk tovább. A 4. sorban megpróbáljuk lekérdezni a felhasználó nevét, majd az 5. sorban megnézzük, hogy a találat nagyobb-e mint 0. Ha igen, akkor a 7. sorban visszaadjuk a megtalált felhasználó nevét.

Maga az oldal

A következő létrehozandó fájl az `index.php`. Ennek felépítése is 90%-ban megegyezik a gyökérben található `index.php`-vel. Másoljuk át az `admin` könyvtárba a fájlt, majd módosítsuk a következő sorokat.

Az első pár sorban található `include` kezdetű soroknál a `process.php` kivételével a fájl neve elé illesszük be a `../` karaktereket, hogy a már meglévő fájlokat importálják be. A `process.php` az `admin` felületen történő kéréseket fogja végrehajtani, tehát ezt létre kell hoznunk az `admin` könyvtárban is.

A `<title>` html-tagek után található `<script>` kezdetű blokkokat töröljük ki, mivel ebben a programkódban nem fogunk alkalmazni JavaScript-et..

A <link kezdetű sor fog következni a <title> tag után, aminek az elérése ../css/style.css-re módosul.

A legutolsó módosítás ebben a fájlban, az include('leftMenu.php') sor után következik. A feltételvizsgálatot ki kell egészíteni, hogy csak akkor teljesüljön a feltétel, ha már belépett valaki.

```
1  if (isset($_GET["page"]) && ($_GET["page"] !=  
    ' ') && isset($_SESSION["userId"]) &&  
    ($_SESSION["userId"] != ' '))
```

Itt látható a munkamenet userId vizsgálata annak megállapítására, hogy belépett-e már valaki.

Ezzel az index.php fájl is elkészült.

A bal oldali menü

A következő fájl, amelyet el kell készítenünk, a bal oldali menüt megvalósító leftMenu.php állomány. A kódja itt látható:

```
1  <div id="menuColumn">  
2  <?  
3  if (isset($_SESSION["userId"]) &&  
    ($_SESSION["userId"] != ' ')) {  
4  ?>  
5      <ul>  
6          <li><a  
            href="?page=users">Felhasználók</a></li>  
7          <li><a  
            href="?page=customers">Vevők</a></li>  
8          <li><a  
            href="?page=products">Termékek</a></li>  
9          <li><a  
            href="?page=orders">Rendelések</a></li>  
10         <li><a  
            href="?page=settings">Beállítások</a></li>  
11         <li><a  
            href="?page=category">Kategóriák</a></li>  
12         </ul>  
13 <?  
14 }  
15 ?>  
16 </div>
```

A 3. sorban található a feltételvizsgálat, ami csak akkor jeleníti meg az egyes menüpontokat, ha már belépett valaki.

A kezdőlap

Létre kell még hoznunk a pages könyvtárban belül a mainPage.php fájlt, ami alapértelmezésben megjelenik. Mivel az adminisztrációs felületnél ez az oldal jelenik meg, ezért itt hozzuk létre a belépési felületet.

```

1      <div id="contentColumn">
2      <div id="contentMenu">
3          <a href="..">Vissza a webáruházhoz</a>
        &raquo; Adminisztráció
4      </div>
5      <div id="contentTitle">
6          Belépés a rendszerbe
7      </div>
8      <div id="contentBox">
9          <div class="error"><?=$error;?></div>
10         <div
11             class="message"><?=$message;?></div>
12             <form action="" method="post">
13                 <table border="0" cellpadding="3"
14                     cellspacing="0" align="center" width="98%">
15                     <tr>
16                         <td align="right" width="50%">
17                             E-mail cím:
18                         </td>
19                         <td align="left">
20                             <input type="text" name="email"
21                             value="" /> <span class="require">*</span>
22                         </td>
23                     </tr>
24                     <tr>
25                         <td align="right">
26                             Jelszó:
27                         </td>
28                         <td align="left">
29                             <input type="password"
30                             name="pass" value="" /> <span
31                             class="require">*</span>
32                         </td>
33                     </tr>
34                     <tr>
35                         <td align="center" colspan="2">
36                             <input type="submit"
37                             name="login" value="Belépés" />
38                         </td>
39                     </tr>
40                 </table>
41             </form>
42         </div>
43     </div>

```

```
33         </tr>
34     </table>
35 </form>
36 </div>
37 </div>
```

Ezzel kialakítottuk az adminisztrációs felület kinézetét, de az még nem működőképes. A további leckeiken végighaladva a könyv végére elkészül a könyv-áruház és a hozzá tartozó adminisztrációs felület is.

5.3 ÖSSZEFOGLALÁS, KÉRDÉSEK

5.3.1 Összefoglalás

Ezzel a leckével elértünk arra a pontra, hogy most már a felületek kialakítása is megtörtént, és belevethetjük magunkat magának az oldal lelkének a leprogramozására. Most jött el az ideje, hogy megírjuk a felhasználók kezelését kiszolgáló részt, a termékek listázásáért és megjelenítéséért szolgáló kódot.

5.3.2 Önellenőrző kérdések

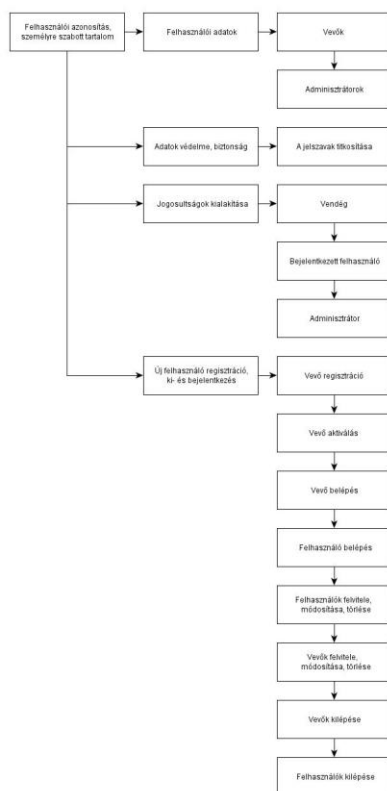
- Miért van szükségünk adminisztrációs felületre?
- Mi értelme van, hogy az oldalt, több kis kódra bontjuk fel?
- Van-e esetleg más megoldás a munkameneten kívül a felhasználók azonosítójának tárolására?

6. FELHASZNÁLÓI AZONOSÍTÁS, SZEMÉLYRE SZABOTT TARTALOM

6.1 CÉLKITÚZÉSEK ÉS KOMPETENCIÁK

A következő fejezetben a felhasználók és vevők adminisztrációját fogjuk áttekinteni. Ha a vevők beregisztrálnak az oldalra, akkor tudnak könyvet rendelni, és a rendeléseket is nyomon lehet követni. A vevők saját magukat is be tudják regisztrálni, viszont az adminisztrátorokat csak adminisztrátor vehet fel a rendszerbe.

6.2 TANANYAG



16. ábra: A lecke áttekintése

6.2.1 Felhasználói adatok

Vevők

Ahhoz, hogy a webáruházban meg tudjunk rendelni egy könyvet, valahogy azonosítanunk kell magunkat. Vevőknél a személyes adatokon kívül a szállítási információkat is el kell tárolnunk a könyv kézbesítéséhez. Első lépésként létre kell majd hozni egy olyan felületet, ahol a felhasználók be tudják majd regisztrálni magukat. Ők a `customer` táblába fognak bekerülni az adatbázisban.

A vevőknél a következő adatokat fogjuk letárolni.

- Név
- E-mail
- Jelszó
- Telefon
- Irányítószám
- Város
- Utca, házszám
- Megjegyzés

Itt csak azok a felhasználók tudnak regisztrálni, akik vásárolni szeretnének.

Adminisztrátorok

Az egyes adminisztrátoroknak is be kell majd tudniuk lépni a rendszerbe, az egyes karbantartási funkciók elvégzésére. Az adminisztrátorok nem regisztrálhatnak be az oldalra, hanem már a létező adminisztrátor tudja az új felhasználókat bejegyezni, amelynél az adatok a `user` táblába kerülnek be, a következő adatokkal:

- Név
- Jelszó
- E-mail

Mindkét esetben az e-mail címmel és jelszóval tudja magát azonosítani a felhasználó. Ez abból a szempontból célszerű, hogy nem kell plusz mezőt létrehozni a felhasználói névnek, és a felhasználónak sem kell plusz egy azonosítót megjegyeznie.

6.2.2 Adatok védelme, biztonság

Az adatok védelmére azért is kell ebben az esetben odafigyelni, mivel minden felhasználó az e-mail címével azonosítja magát, ezért ha valakinek sikerül kiolvasni a táblából az e-mail címeket, akkor már a birtokában is van kellő mennyiségű adatnak, amivel akár egy spamáradatot is megindíthat.

Célszerű az adatbázisnak olyan jelszót adni, amelyben van kis- és nagybetű, számok és valamilyen speciális karakter. Minél bonyolultabb a jelszó, annál nagyobb biztonságban vannak az adataink.

A másik célpont az adathalászok számára a jelszavak megszerzése. Ennél az áruháznál a jelszavak egyirányú titkosítással vannak ellátva. Ez még fűszerezve van egy titkos kulccsal, amely több különböző karakterből álló karakterlánc.

A jelszavak titkosítása

Miután a regisztrációs oldalon megadtuk az adatainkat, köztük a jelszót is, előállítunk egy sztringet. Ennek a sztringnek az első része maga a jelszó, majd ezt követi az e-mail cím, majd végül a titkos kulcs. Miután összeállítottuk az adatokat, kódoljuk md5-ös kódolással. Ezáltal maga a jelszó az e-mail címhez is van kötve, és ha megfigyeljük, akkor ebből a három adatból elég nagy bonyolultságú karakterhalmaz jön össze, amelynek a visszafejtése, nem egyszerű feladat.



Az md5-ös kódolás egyirányú algoritmus, azaz vissza nem fejthető. Hasonló függvény még például, az sha1.

Belépéskor ugyanezt a kódolt szöveget állítjuk elő és így vizsgáljuk meg a jelszót. Ahhoz hogy ezt tudjuk használni, vegyük fel a következő sort a `config.php` állományba.

```
1 $secretKey = '!aNOUJQ784_Lb9ZVH#p_';
```

Ez a sztring bármi lehet, csak arra figyeljünk oda, hogy ha megváltoztatjuk, akkor az addig létrehozott jelszavak már nem lesznek használhatóak!

6.2.3 Jogosultságok kialakítása

Mivel ez az áruház nem egy óriási boltot működtet, így a jogosultsági rendszer sem a legnagyobb. Ha jól megfigyeljük az eddigi adatokat, akkor 3 különböző felhasználót tudunk megkülönböztetni.

Vendég

A vendég a legalacsonyabb szintű felhasználó. Ő a termékeket csak listázhatja, és elolvashatja a részletes adatlapját, de nem rendelhet. Azaz ő az a felhasználó, akinek úgymond semmihez nincs joga.

Bejelentkezett felhasználó

A bejelentkezett felhasználó már tud könyvet rakni a kosárba, majd onnan megrendelni az(oka)t. Ehhez azonban szüksége van egy regisztrációra. Úgy is lehet mondani, hogy ő van a középső szinten. Nem az övé a legkisebb jogosultság, de nem is az övé a legnagyobb.

Adminisztrátor

Az adminisztrátornak van a legmagasabb jogosultsága az egész rendszerben. Neki egy külön felület van létrehozva, ahol az oldalon szereplő adatokat tudja szerkeszteni. Ezzel ellentétben ő nem tartozik azok közé a felhasználók közé, akik könyvet rendelhetnek az oldalon. Az adminisztrátor feladata a webáruház karbantartása.

6.2.4 Új felhasználó regisztrációja, ki- és bejelentkezés**Vevőregisztráció**

A vevők regisztrációjához, hozzunk létre egy új oldalt a pages könyvtárban `registrationPage.php` néven. Ennek tartalma itt található:

```

1      <div id="contentColumn">
2          <div id="contentMenu">
3              <a href=".">Kezdőlap</a> &raquo; Re-
    gisztráció
4          </div>
5          <div id="contentTitle">
6              Regisztráció
7          </div>
8          <div id="contentBox">
9              <div class="error"><?=$error;?></div>
10             <div
    class="message"><?=$message;?></div>
11             <form action="" method="post">
12                 <table border="0" cellpadding="3"
    cellspacing="0" align="center" width="98%">
13                     <tr>
```

```
14         <td align="left" colspan="2">
15             <strong>Személyes ada-
16             tok</strong>
17         </td>
18     </tr>
19     <tr>
20         <td align="right" width="50%">
21             Név:
22         </td>
23         <td align="left">
24             <input type="text" name="name"
25             value="<?=(isset($_POST["name"]))?$_POST["name"]
26             :'';?>" /> <span class="require">*</span>
27         </td>
28     </tr>
29     <tr>
30         <td align="right">
31             E-mail cím:
32         </td>
33         <td align="left">
34             <input type="text" name="email"
35             value="<?=(isset($_POST["email"]))?$_POST["email"]
36             :'';?>" /> <span class="require">*</span>
37         </td>
38     </tr>
39     <tr>
40         <td align="right">
41             Jelszó ismétlés:
42         </td>
43         <td align="left">
44             <input type="password"
45             name="pass2" value="" /> <span
46             class="require">*</span>
47         </td>
48     </tr>
49     <tr>
50         <td align="right">
51             Jelszó:
52         </td>
53         <td align="left">
54             <input type="password"
55             name="pass" value="" /> <span
56             class="require">*</span>
57         </td>
```

```

49         </tr>
50         <tr>
51             <td align="right">
52                 Telefon:
53             </td>
54             <td align="left">
55                 <input type="text" name="phone"
value="<?=(isset($_POST["phone"]))?$_POST["phone
"]:'';?>" />
56             </td>
57         </tr>
58         <tr>
59             <td align="left" colspan="2">
60                 <strong>Szállítási ada-
tok</strong>
61             </td>
62         </tr>
63         <tr>
64             <td align="right">
65                 Irányítószám:
66             </td>
67             <td align="left">
68                 <input type="text"
name="postcode"
value="<?=(isset($_POST["postcode"]))?$_POST["po
stcode"]:'';?>" /> <span
class="require">*</span>
69             </td>
70         </tr>
71         <tr>
72             <td align="right">
73                 Város:
74             </td>
75             <td align="left">
76                 <input type="text" name="city"
value="<?=(isset($_POST["city"]))?$_POST["city"]
:'';?>" /> <span class="require">*</span>
77             </td>
78         </tr>
79         <tr>
80             <td align="right">
81                 Utca, házszám:
82             </td>
83             <td align="left">

```

```
84         <input type="text"
      name="address"
      value="<?=(isset($_POST["address"]))?$_POST["add
      ress"]:'';?>" /> <span class="require">*</span>
85     </td>
86 </tr>
87 <tr>
88     <td align="right" valign="top">
89         Megjegyzés:
90     </td>
91     <td align="left">
92         <textarea
      name="notes"><?=(isset($_POST["notes"]))?$_POST[
      "notes"]:'';?></textarea>
93     </td>
94 </tr>
95 <tr>
96     <td align="center" colspan="2">
97         <input type="submit"
      name="registration" value="Regisztráció" />
98     </td>
99 </tr>
100 </table>
101 </form>
102 </div>
103 </div>
```

A 3. sorban található a `contentMenu` részben az úgynevezett „*breadcrumb*” menü. Ez minden egyes oldal tetején fog látszódni, hogy megkönnyítse a navigációt az oldalon. Egy egyszerű táblázatos formában építjük fel a regisztrációs űrlapot. Az egyes beviteli mezők értékeit a `$_POST` változóból nyerjük, mivel ha hibásan töltjük ki a regisztrációs űrlapot, akkor a már küldött adatokat újra megjelenítjük, így nem kell újból kitöltenünk az egész űrlapot. Itt kérjük be a személyes adatok mellett a szállítási információkat is. A kötelező mezőket egy `require` stílusú `*`-gal jelöljük meg.

17. ábra: Vevő regisztrációja

A felhasználóknak szóló információkat a `form` tag előtti 2 `div` jeleníti meg, attól függően, hogy hibáról, vagy információról van szó.

Ennek az űrlapnak a feldolgozása a `process.php` fájlban fog történni, mivel ennek az importálása az oldal legelején történik, így az egész oldalon elérhető a visszatérési érték.

```

1  if (isset($_POST["registration"]) &&
    ($_POST["registration"] != '')) {
2      if (isset($_POST["name"]) &&
        ($_POST["name"] != '')) && isset($_POST["email"])
        && ($_POST["email"] != '') &&
        isset($_POST["pass"]) && ($_POST["pass"] != '')
        && isset($_POST["pass2"]) && ($_POST["pass2"] !=
        '') && isset($_POST["postcode"]) &&
        ($_POST["postcode"] != '') &&
        isset($_POST["city"]) && ($_POST["city"] != '')
        && isset($_POST["address"]) &&
        ($_POST["address"] != '')) {

```

```
3         if ($_POST["pass"] == $_POST["pass2"]) {
4             $query = mysql_query("SELECT
customer_id FROM ".$GLOBALS["customerTable"]."
WHERE customer_email = '". $_POST["email"]." '");
5             if (mysql_num_rows($query) == 0) {
6                 $code = substr(md5(time()),0,8);
7                 $query = mysql_query("INSERT INTO
".$GLOBALS["customerTable"]."
(customer_name,customer_email,customer_password,
customer_phone,customer_postcode,customer_city,c
ustomer_address,customer_notes,customer_code)
VALUES
('".$_POST["name"]."','".$_POST["email"]."',
'".$_md5($_POST["pass"]." ".$_POST["email"]."
".$GLOBALS["secretKey"])."','".$_POST["phone"]."','
'".$_POST["postcode"]."','".$_POST["city"]."','
'".$_POST["address"]."','".$_POST["notes"]."','
'".$_code.'"");
8                 if ($query) {
9                     $url =
"http://".$_SERVER['HTTP_HOST']."/?page=activate
&email=".$_POST["email"]." &code=".$_code;
10                    $msg = "Kedves
".$_POST["name"]."!\n\n";
11                    $msg .= "A(z)
".$GLOBALS["emailName"]." rendszerében, ezzel az
email címmel regisztráltak.\n\n";
12                    $msg .= "Amennyiben Ön kezde-
ményezte a regisztrációt, kérjük, az alábbi
linke kattintva véglegesítse regisztráció-
ját.\n";
13                    $msg .= $url."\n\n";
14                    $msg .= "Amennyiben nem Ön je-
lezte a regisztrációs igényt, hagyja figyelmen
kívül az e-mailt!\n\n";
15                    $msg .= "Ezt a levelet a
".$GLOBALS["emailName"]." rendszer küldte Önnek.
Kérjük, erre a levélre ne válaszoljon!";
16                    $m = mail($_POST["name"]."
<".$_POST["email"].">",$GLOBALS["emailSubject"],
$msg, "From: ".$GLOBALS["emailName"]."
<".$GLOBALS["emailAddress"].">".$_r\n"."Content-
type: text/plain; charset=UTF-8");
```

```
17         $message = 'Sikeres regisztráció! A(z) "'.$_POST["email"].'" e-mail címre küldtünk egy megerősítő levelet!';
18         unset($_POST);
19     } else {
20         $error = 'Hiba történt a regisztráció közben!';
21     }
22 } else {
23     $error = 'Ezzel az e-mail címmel már regisztráltak a rendszerbe!';
24 }
25 } else {
26     $error = 'A két jelszó nem egyezik meg!';
27 }
28 } else {
29     $error = 'Töltse ki a kötelező mezőket!';
30 }
31 }
```

Az 1. sorban látható a feltételvizsgálat, ami ha teljesül, lefut a regisztrációs kérés. A 2. sorban található az a vizsgálat, ami ellenőrzi, hogy a kötelező mezők ki vannak-e töltve. Ha nincs kitöltve az összes mező, akkor az `error` változóban hibát jelzünk. Ha teljesül a feltétel akkor a 3. sorban azt vizsgáljuk meg, hogy a megadott két jelszó egyezik-e. Amennyiben nem, akkor a már említett `error` változóba hibát írunk, egyébként folytatjuk az ellenőrzéseket. Megvizsgáljuk a 4-5. sorban hogy regisztráltak-e már ilyen e-mail címmel. Ha igen, akkor szintén hibát generálunk, egyébként elkészítünk egy aktiváló kulcsot (6. sor), és rögzítjük az adatokat az adatbázisban (7. sor).

Amennyiben nem sikerült valamilyen okból az SQL-kérés, akkor szintén hibát jelzünk, egyébként elkészítjük az aktiváló levél tartalmát, és a 16. sorban megpróbáljuk elküldeni a levelet a PHP beépített `mail` függvényével. A levélküldéshez szükségünk lesz egy pár változóra, amelyeket szintén a `config.php` fájlban fogunk letárolni. Ezek a következők: az aktiváló levél feladója, címe és tárgya.

```
1 $emailName = "BookStore Webáruház";
2 $emailAddress = "no-reply@bookstore.com";
3 $emailSubject = "Regisztráció megerősítése"
```

Ha a levél elküldése sikeres, akkor a `message` változóban értesítjük a felhasználót az a sikeres regisztrációról és az aktiváló levél érkezéséről. Sikertelen-ség esetén az `error` változóba ismét hiba kerül. A `message` és `error` változók értékét az űrlap előtt található `error` és `message` `div`-ben jelenítjük meg.

Vevőaktiválás

Ez is egy biztonsági funkció. Senki sem örülne annak, ha valaki szórakozásból felíratná az e-mail címe alapján egy csomó webáruházba. Az aktiváló levelet a sikeres regisztrációkor küldjük el a felhasználónak. Amennyiben nem a címzett végezte a regisztrációt, akkor figyelmen kívül hagyja az aktiváló levelet. Az aktiváló levélben található egy link, amelynek segítségével az aktiváló oldalra tudunk eljutni. Ehhez el kell készítenünk egy felületet. Ezt szintén a `pages` könyvtárba fogjuk létrehozni `activatePage.php` néven. Ennek a kódja itt olvasható.

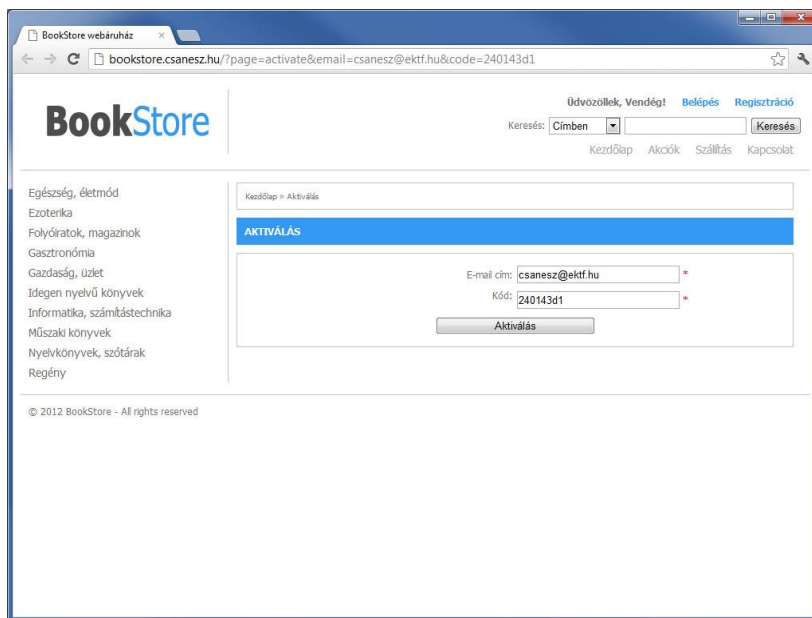
```
1      <div id="contentColumn">
2      <div id="contentMenu">
3          <a href=".">Kezdőlap</a> &raquo; Aktivá-
lás
4      </div>
5      <div id="contentTitle">
6          Aktiválás
7      </div>
8      <div id="contentBox">
9          <div
class="error"><?=$error;?></div>
10         <div
class="message"><?=$message;?></div>
11         <form action="" method="post">
12             <table border="0" cellpadding="3"
cellspacing="0" align="center" width="98%">
13                 <tr>
14                     <td align="right" width="50%">
15                         E-mail cím:
16                     </td>
17                     <td align="left">
18                         <input type="text" name="email"
value="<?=$_REQUEST["email"];?>" /> <span
class="require">*</span>
19                     </td>
20                 </tr>
21                 <tr>
```

```

22         <td align="right" valign="top">
23             Kód:
24         </td>
25         <td align="left">
26             <input type="text" name="code"
27             value="<?=$_REQUEST["code"];?>" /> <span
28             class="require">*</span>
29         </td>
30     </tr>
31     <tr>
32         <td align="center" colspan="2">
33             <input type="submit"
34             name="activation" value="Aktiválás" />
35         </td>
36     </tr>
37 </table>
</form>
</div>
</div>

```

A felépítés hasonló mint a regisztrációnál. Itt is van error és message div, és ez is táblázatos felépítésű.



18. ábra: Vevő aktiválása

Megfigyelhető, hogy a regisztrációs űrlapnál a `$_POST` tömböt használtuk, itt viszont a `$_REQUEST` tömböt használjuk. Ez azért van, mert így nem csak `$_POST` értéket hanem `$_GET` értéket is tudunk fogadni.



A `$_REQUEST` tömb tartalmazza a `$_POST` és `$_GET` értékét is.

Ha az aktiváló levélben kattintunk a linken, akkor az űrlap automatikusan kitöltődik. Ha nem akarunk kattintani a linken, akkor az oldalt behozva az aktiváló kód és e-mail cím ismeretében is ki tudjuk tölteni az oldalt. Az aktiváló kódot feldolgozó függvény szintén a `process.php` fájlban foglal helyet.

```
1  if (isset($_POST["activation"]) &&
    ($_POST["activation"] != '')) {
2      if (isset($_POST["email"]) &&
        ($_POST["email"] != '' && isset($_POST["code"])
        && ($_POST["code"] != '')) {
3          $query = mysql_query("SELECT customer_id
FROM ".$GLOBALS["customerTable"]." WHERE
customer_email = '". $_POST["email"]."' AND
customer_code = '". $_POST["code"]."'");
4          if (mysql_num_rows($query) > 0) {
5              $query = mysql_query("UPDATE
".$GLOBALS["customerTable"]." SET customer_code
= '' WHERE customer_email =
'".$_POST["email"]."'");
6              if ($query) {
7                  $message = 'Sikeres felhasználó
aktiválás!';
8                  unset($_POST);
9                  unset($_REQUEST);
10             } else {
11                 $error = 'Hiba történt az aktivá-
lás közben!';
12             }
13         } else {
14             $error = 'Hibás e-mail cím vagy
kód!';
15         }
16     } else {
17         $error = 'Töltse ki a kötelező mező-
ket!';
18     }
19 }
```

Az 1. sorban lévő feltétel az aktiváló űrlapon található Aktiválás gombra fog teljesülni. A 2. sorban található feltételvizsgálat fogja megvizsgálni, hogy az összes kötelező mező ki lett-e töltve. Ha igen, akkor megvizsgáljuk, hogy van-e az adatbázisban olyan felhasználó, akinek az e-mail címe és aktiváló kódja megegyezik a megadott adatokkal (3. sor). Amennyiben találunk ilyen felhasználót, töröljük az aktiváló kódját, ezzel jelezve, hogy az aktiválás sikeres volt (5. sor). Ezt egy „Sikeres aktiválás” üzenettel jelezzük a felhasználó felé a `message` változón keresztül. Természetesen, ha hiba történik akár az adatok bevitele közben, vagy a feldolgozás közben, akkor az `error` változó tartalma fog értéket kapni.

Vevő belépés

A következő lépés, hogy a már beregisztrált felhasználókat be tudjuk lép-tetni az oldalra. Ehhez hozzuk létre a `pages` könyvtárban a `loginPage.php` fájlt, amely a felső menüben lévő *Belépés* gombra való kattintáskor fog betöl-tődni.

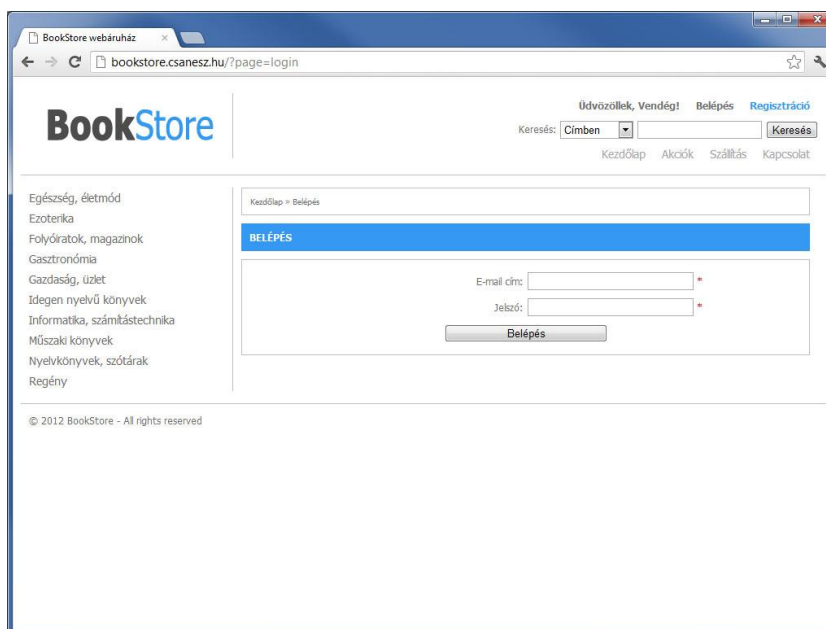
```

1      <div id="contentColumn">
2          <div id="contentMenu">
3              <a href=".">Kezdőlap</a> &raquo; Belépés
4          </div>
5          <div id="contentTitle">
6              Belépés
7          </div>
8          <div id="contentBox">
9              <div class="error"><?=$error;?></div>
10             <div
11 class="message"><?=$message;?></div>
12             <form action="" method="post">
13                 <table border="0" cellpadding="3"
14                 cellspacing="0" align="center" width="98%">
15                     <tr>
16                         <td align="right" width="50%">
17                             E-mail cím:
18                         </td>
19                         <td align="left">
20                             <input type="text" name="email"
21                             value="" /> <span class="require">*</span>
22                         </td>
23                     </tr>
24                     <tr>
25                         <td align="right">
26                             Jelszó:

```

```
24         </td>
25         <td align="left">
26             <input type="password"
27             name="pass" value="" /> <span
28             class="require">*</span>
29         </td>
30     </tr>
31     <tr>
32         <td align="center" colspan="2">
33             <input type="submit"
34             name="login" value="Belépés" />
35         </td>
36     </tr>
37 </table>
</form>
</div>
</div>
```

Mint látható, ez a felület nagyon hasonlít az adminisztrációs felület belépési oldalára. Jóformán ugyanaz az oldal, mindössze annyiban különböznek egymástól, hogy az oldalak tetején lévő menü nem ugyanaz.



19. ábra: Vevő belépés

Ha valaki megpróbál itt belépni, akkor azt valahol fel is kell dolgozni. Ez a `process.php` fájlban fog végrehajtódni. Ehhez illesszük be a következő kódot a fájlba:

```
1  if (isset($_POST["login"]) && ($_POST["login"]
    != '')) {
2      if (isset($_POST["email"]) &&
        ($_POST["email"] != '') && isset($_POST["pass"])
        && ($_POST["pass"] != '')) {
3          $email = $_POST["email"];
4          $password =
            md5($_POST["pass"].$_POST["email"].
              $GLOBALS["secretKey"]);
5          $query = mysql_query("SELECT
            customer_id, customer_code FROM
            ".$customerTable." WHERE customer_email =
            '".$_.$email.'" AND customer_password =
            '".$_.$password.'"');
6          if (mysql_num_rows($query) > 0) {
7              if ($result =
                mysql_fetch_object($query)) {
8                  if ($result->customer_code == '')
9                  {
10                     $message = 'Sikeres belépés!';
11                     $_SESSION["customerId"] =
12                     $result->customer_id;
13                     header('Location: .');
14                 } else {
15                     $error = 'A felhasználó még nem
16                     aktiválta a regisztrációját!';
17                 }
18             }
19             } else {
20                 $error = 'Hibás e-mail cím vagy jel-
21                 szó!';
22             } else {
23                 $error = 'Adja meg az email címét és
24                 jelszavát a belépéshez!';
25             }
26         }
27     }
```

A 2. sorban látható a vizsgálat, hogy ki vannak-e töltve a kötelező mezők. A 4. sorban előállítjuk a jelszót az e-mail címből, jelszóból és a titkos kulcsból, majd a 5. sorban indított lekérdezésben megnézzük, hogy létezik-e ilyen felhasználó. Ha van ilyen felhasználó, akkor egy munkamenet `customerId` vál-

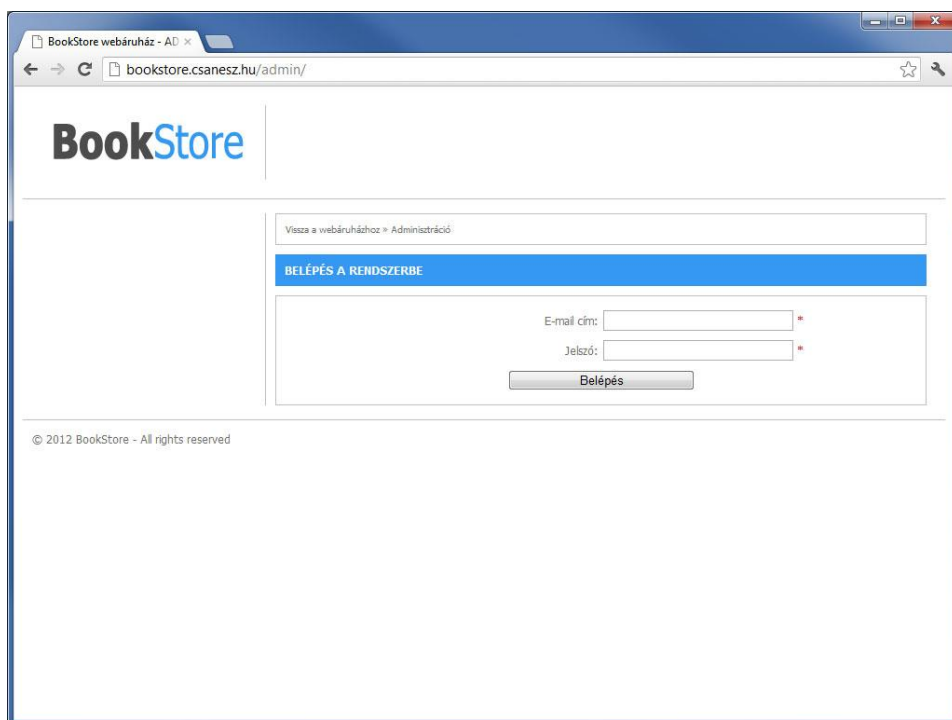
tozójában eltároljuk az azonosítóját, majd a főoldalra irányítjuk át a felhasználót. Hiba esetén az `error` változóban jelezzük a hibát a felhasználónak.

Felhasználó belépés

Az adminisztrációs felület belépési oldalát már a 5.2.3. leckében létrehoztuk, de a feldolgozó eljárást még nem. Ezt az `admin` könyvtáron belüli `process.php` fájlban fogjuk letárolni.

```
1  if (isset($_POST["login"]) && ($_POST["login"]
    != '')) {
2      if (isset($_POST["email"]) &&
        ($_POST["email"] != '' && isset($_POST["pass"])
        && ($_POST["pass"] != '')) {
3          $email = $_POST["email"];
4          $password =
            md5($_POST["pass"].$_POST["email"].
            $GLOBALS["secretKey"]);
5          $query = mysql_query("SELECT user_id
            FROM ".$userTable." WHERE user_email =
            '". $email ."' AND user_password =
            '". $password ."'");
6          if (mysql_num_rows($query) > 0) {
7              if ($result =
                mysql_fetch_object($query)) {
8                  $message = 'Sikeres belépés!';
9                  $_SESSION["userId"] = $result-
                    >user_id;
10                 header('Location: .');
11             }
12         } else
13             $error = 'Hibás e-mail cím vagy jel-
                szó!';
14         } else {
15             $error = 'Adja meg az email címét és
                jelszavát a belépéshez!';
16         }
17     }
```

Az adminisztrációs felületen lehetőségünk van felhasználókat és vevőket is felvinni a rendszerbe.



20. ábra: Felhasználó belépés

Új felhasználók felvitelével több adminisztrátor is lesz a rendszerben, akiknek teljes joguk lesz az oldal karbantartásához.

Felhasználók felvitele, módosítása, törlése

Mint minden egyes művelethez, ehhez is ki kell alakítanunk egy felületet. Ez a felület az admin könyvtárban lévő `usersPage.php` fájlban fog helyet kapni. Lássuk a kódot.

```

1      <div id="contentColumn">
2      <div id="contentMenu">
3          Felhasználók
4      </div>
5      <div id="contentTitle">
6          Felhasználók
7      </div>
8      <div id="contentBox">
9          <div class="error"><?=$error;?></div>
10         <div
            class="message"><?=$message;?></div>

```

```
11 <?
12 if (isset($_POST["func"]) && ($_POST["func"] ==
13 'userAdd')) {
14     <form action="" method="post">
15         <table border="0" cellpadding="3"
16         cellspacing="0" align="center" width="98%">
17             <tr>
18                 <td align="right" width="50%">
19                     Felhasználó neve:
20                 </td>
21                 <td align="left">
22                     <input type="text"
23                     name="userName" value="" /> <span
24                     class="require">*</span>
25                 </td>
26             </tr>
27             <tr>
28                 <td align="right">
29                     Felhasználó e-mail címe:
30                 </td>
31                 <td align="left">
32                     <input type="text"
33                     name="userEmail" value="" /> <span
34                     class="require">*</span>
35                 </td>
36             </tr>
37             <tr>
38                 <td align="right">
39                     Felhasználó jelszava:
40                 </td>
41                 <td align="left">
42                     <input type="password"
43                     name="userPass" value="" /> <span
44                     class="require">*</span>
45                 </td>
46             </tr>
47             <tr>
48                 <td align="center" colspan="2">
49                     <input type="submit"
50                     name="userAdd" value="Hozzáadás" />
51                     <button
52                     onclick="javascript:history.back(); return
53                     false;">Vissza</button>
```

```

44         </td>
45     </tr>
46 </table>
47 </form>
48 <?
49 } elseif (isset($_POST["func"]) &&
50 ($_POST["func"] == 'userMod')) {
51     $query = mysql_query("SELECT * FROM
52     ".$GLOBALS["userTable"]." WHERE user_id =
53     ".$_POST["id"]);
54     if (mysql_num_rows($query) > 0) {
55         $result = mysql_fetch_object($query);
56     ?>
57     <form action="" method="post">
58         <table border="0" cellpadding="3"
59         cellspacing="0" align="center" width="98%">
60             <tr>
61                 <td align="right" width="50%">
62                     Felhasználó neve:
63                 </td>
64                 <td align="left">
65                     <input type="text"
66                     name="userName" value="<?=$result->user_name;?>"
67                     /> <span class="require">*</span>
68                 </td>
69             </tr>
70             <tr>
71                 <td align="right">
72                     Felhasználó e-mail címe:
73                 </td>
74                 <td align="left">
75                     <input type="text"
76                     name="userEmail" value="<?=$result->
77                     user_email;?>" /> <span
78                     class="require">*</span>
79                 </td>
80             </tr>
81             <tr>
82                 <td align="right">
83                     Felhasználó jelszava:
84                 </td>
85                 <td align="left">
86                     <input type="password"
87                     name="userPass" value="" />

```

```
78         </td>
79     </tr>
80     <tr>
81         <td align="center" colspan="2">
82             <input type="hidden" name="id"
value="<?=$result->user_id;?>" />
83             <input type="submit"
name="userMod" value="Módosítás" />
84             <button
onclick="javascript:history.back(); return
false;">Vissza</button>
85         </td>
86     </tr>
87 </table>
88 </form>
89 <?
90     } else {
91         print('Nincs ilyen felhasználó!');
92     }
93 } else {
94     $query = mysql_query("SELECT * FROM
95     ".$GLOBALS["userTable"]." ORDER BY user_name");
96     print('<form action="" method="post"
name="userAdd">');
97     print('<a href="#"
onclick="document.forms[\'userAdd\'].submit();
return false;">Felhasználó hozzáadása</a>');
98     print('<input type="hidden" name="func"
value="userAdd" />');
99     print('</form>');
100     print('<table border="0" cellpadding="3"
cellspacing="0" width="98%" id="cartTable">');
101     print('<tr>');
102     print('<th>Felhasználó neve</th>');
103     print('<th>E-mail címe</th>');
104     print('<th>&nbsp;</th>');
105     print('</tr>');
106     while ($result =
mysql_fetch_object($query)) {
107         print('<tr>');
108         print('<td>'.$result->user_name.'</td>');
109         print('<td>'.$result->user_email.'</td>');
```

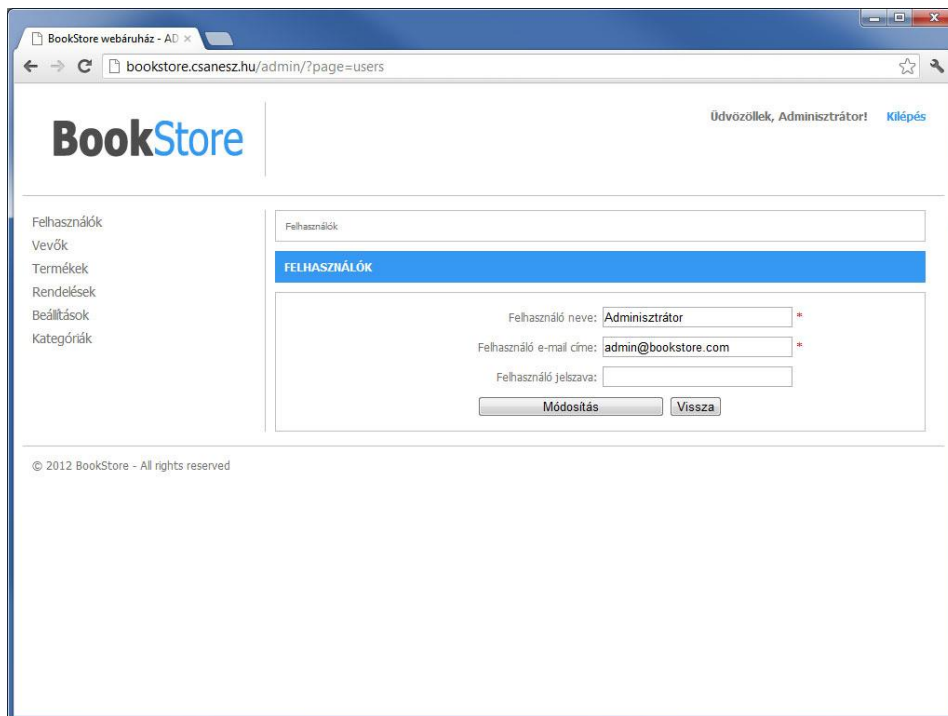
```

109         print('<td align="center">');
110         print('<form action="" method="post"
    name="userMod_'. $result->user_id.' "
    style="display: inline;">');
111         print('<a href="#"
    onclick="document.forms[\'userMod_'. $result-
    >user_id.\'] submit(); return false;"></a> ');
112         print('<input type="hidden"
    name="func" value="userMod" />');
113         print('<input type="hidden"
    name="id" value="'. $result->user_id.' " />');
114         print('</form>');
115         print('<form action="" method="post"
    name="userDel_'. $result->user_id.' "
    style="display: inline;">');
116         print('<a href="#" onclick="if
    (confirm(\'Biztos benne hogy törli?\'))
    document.forms[\'userDel_'. $result-
    >user_id.\'] submit(); return false;"></a>');
117         print('<input type="hidden"
    name="func" value="userDel" />');
118         print('<input type="hidden"
    name="id" value="'. $result->user_id.' " />');
119         print('</form>');
120         print('</td>');
121         print('</tr>');
122     }
123     print('</table>');
124 }
125 ?>
126 </div>
127 </div>

```

A 12. és 49. sorban ellenőrizzük, hogy ki van-e töltve a `$_POST` tömb `func` paramétere. Amennyiben nincs kitöltve, úgy a 93. sortól kezdően lévő kódban kilistázzuk egy táblázatba a rendszerben lévő felhasználókat. Az utolsó oszlopban egy-egy kis ikonnal jelezzük a módosítás és törlés lehetőségét. A törlésnél a JavaScriptből már ismerős `confirm` paranccsal egy megerősítést kérünk a felhasználotól a végleges törléshez. A táblázat fölött található egy

hivatkozás, amivel újabb felhasználókat tudunk hozzáadni a listához. A kötelező mezőket itt is kis piros csillaggal jelöljük meg.



21. ábra: Felhasználó módosítása

Az új felhasználó hozzáadása a 14. sortól a 47. sorig tartó úrlapon zajlik. Itt a *Hozzáadás* gombra kattintva a `process.php`-ben fog lefutni egy kódrészlet, ami felveszi az új felhasználót a táblába, majd a felhasználók listája oldalra irányít vissza minket. A `process.php` fájlba beillesztendő új kódrészlet itt látható.

```
1  if (isset($_POST["userAdd"]) &&
    ($_POST["userAdd"] != '')) &&
    isset($_SESSION["userId"]) &&
    ($_SESSION["userId"] != '')) {
2      if (isset($_POST["userName"]) &&
          ($_POST["userName"] != '')) &&
          isset($_POST["userEmail"]) &&
          ($_POST["userEmail"] != '')) &&
          isset($_POST["userPass"]) && ($_POST["userPass"]
            != '')) {
```

```

3          $query = mysql_query("INSERT INTO
    \".$GLOBALS["userTable"]."
    (user_name,user_email,user_password) VALUES
    ('".$_POST["userName"]."','".$_POST["userEmail"]
    ."', '".$_md5($_POST["userPass"]."".$_POST["userEmail"]
    )."'.
    $GLOBALS["secretKey"])."');
4          if ($query) {
5              $message = 'Sikeres felhasználó hoz-
    záadás';
6          } else {
7              $error = 'Hiba a felhasználó hozzá-
    adása közben!';
8          }
9          } else {
10             $error = 'Töltsön ki minden mezőt!';
11         }
12     }

```

Ha nem töltöttünk ki minden mezőt vagy hiba történt az új felhasználó felvitele közben, akkor hibajelzést kapunk, egyébként visszatérünk a felhasználók listájához, ahol már látható is az újonnan felvitt felhasználó.

Egy felhasználó módosítása is ugyanebben a fájlban fog végrehajtódni. Itt annyit fogunk tenni, hogy a kiválasztott felhasználó azonosítóját megkeressük, majd módosítjuk a hozzá tartozó nevet és e-mail címet, és ha megadtak jelszót, akkor azt lekódoljuk. Majd visszatérünk a listához, ahol már látható lesz a változás. A felhasználókat módosító kódrészlet itt látható.

```

1     if (isset($_POST["userMod"]) &&
    ($_POST["userMod"] != '') && isset($_POST["id"])
    && ($_POST["id"] != '') &&
    isset($_SESSION["userId"]) &&
    ($_SESSION["userId"] != '')) {
2         if (isset($_POST["userName"]) &&
    ($_POST["userName"] != '') &&
    isset($_POST["userEmail"]) &&
    ($_POST["userEmail"] != '')) {
3             $query = mysql_query("UPDATE
    \".$GLOBALS["userTable"]." SET user_name =
    '".$_POST["userName"]."', user_email =
    '".$_POST["userEmail"]." WHERE user_id =
    '".$_POST["id"]."");
4             if ($query) {

```

```
5         if (isset($_POST["userPass"]) &&
($POST["userPass"] != '')) {
6             $query = mysql_query("UPDATE
".$GLOBALS["userTable"]." SET user_password =
'".md5($_POST["userPass"].$_POST["userEmail"].
$GLOBALS["secretKey"])." WHERE user_id =
".$_POST["id"]);
7         }
8         $message = 'Sikeres felhasználó mó-
dosítás';
9     } else {
10        $error = 'Hiba a felhasználó módosí-
tása közben!';
11    }
12    } else {
13        $error = 'Töltsön ki minden mezőt!';
14    }
15 }
```

A 3. sorban található lekérdezés fogja módosítani a felhasználók adatait, a 6. sorban lévő lekérdezés pedig akkor fog végrehajtódni, ha az 5. sorban lévő feltétel teljesül, azaz kitöltik a jelszó mezőt.

A törlésnél vigyáznunk kell, mivel nem logikai törlésről van szó, hanem fizikairól, tehát a törölt adatok nem lesznek visszaállíthatóak. A felhasználók törlésének kódja itt található.

```
1     if (isset($_POST["func"]) && ($_POST["func"] ==
'userDel') && isset($_POST["id"]) &&
($POST["id"] != '')) &&
isset($_SESSION["userId"]) &&
($_SESSION["userId"] != '')) {
2         $query = mysql_query("DELETE FROM
".$GLOBALS["userTable"]." WHERE user_id =
".$_POST["id"]);
3         if ($query) {
4             $message = 'Sikeres felhasználó törlés';
5         } else {
6             $error = 'Hiba a felhasználó törlése
közben!';
7         }
8     }
```

Vevők felvitele, módosítása, törlése

Ahogy a felhasználóknál, úgy a vevőknél is ki kell alakítanunk egy felületet, amely az admin könyvtárban lévő `customersPage.php` fájlban fog helyet kapni. Lássuk a kódot:

```

1      <div id="contentColumn">
2      <div id="contentMenu">
3          Vevők
4      </div>
5      <div id="contentTitle">
6          Vevők
7      </div>
8      <div id="contentBox">
9          <div
10             class="error"><?=$error;?></div>
11             <div
12                 class="message"><?=$message;?></div>
13 <?
14     if (isset($_POST["func"]) && ($_POST["func"] ==
15         'customerAdd')) {
16     ?>
17         <form action="" method="post">
18         <table border="0" cellpadding="3"
19             cellspacing="0" align="center" width="98%">
20             <tr>
21                 <td align="left" colspan="2">
22                     <strong>Személyes adatok</strong>
23                 </td>
24             </tr>
25             <tr>
26                 <td align="right" width="50%">
27                     Név:
28                 </td>
29                 <td align="left">
30                     <input type="text"
31                         name="customerName" value="" /> <span
32                         class="require">*</span>
33                 </td>

```

```
34         <input type="text"
    name="customerEmail" value="" /> <span
    class="require">*</span>
35     </td>
36 </tr>
37 <tr>
38     <td align="right">
39         Jelszó ismétlés:
40     </td>
41     <td align="left">
42         <input type="password"
    name="customerPass2" value="" /> <span
    class="require">*</span>
43     </td>
44 </tr>
45 <tr>
46     <td align="right">
47         Jelszó:
48     </td>
49     <td align="left">
50         <input type="password"
    name="customerPass" value="" /> <span
    class="require">*</span>
51     </td>
52 </tr>
53 <tr>
54     <td align="right">
55         Telefon:
56     </td>
57     <td align="left">
58         <input type="text"
    name="customerPhone" value="" />
59     </td>
60 </tr>
61 <tr>
62     <td align="left" colspan="2">
63         <strong>Szállítási adatok</strong>
64     </td>
65 </tr>
66 <tr>
67     <td align="right">
68         Irányítószám:
69     </td>
70     <td align="left">
```

```
71         <input type="text"
       name="customerPostcode" value="" /> <span
       class="require">*</span>
72     </td>
73 </tr>
74 <tr>
75     <td align="right">
76         Város:
77     </td>
78     <td align="left">
79         <input type="text"
       name="customerCity" value="" /> <span
       class="require">*</span>
80     </td>
81 </tr>
82 <tr>
83     <td align="right">
84         Utca, házszám:
85     </td>
86     <td align="left">
87         <input type="text"
       name="customerAddress" value="" /> <span
       class="require">*</span>
88     </td>
89 </tr>
90 <tr>
91     <td align="right" valign="top">
92         Megjegyzés:
93     </td>
94     <td align="left">
95         <textarea
       name="customerNotes"></textarea>
96     </td>
97 </tr>
98 <tr>
99     <td align="center" colspan="2">
100         <input type="submit"
       name="customerAdd" value="Hozzáadás" />
101         <button
       onclick="javascript:history.back(); return
       false;">Vissza</button>
102     </td>
103 </tr>
104 </table>
```

```
105     </form>
106 <?
107     } elseif (isset($_POST["func"]) &&
108     ($_POST["func"] == 'customerMod')) {
109         $query = mysql_query("SELECT * FROM
110         ".$GLOBALS["customerTable"]." WHERE customer_id
111         = ".$_POST["id"]);
112         if (mysql_num_rows($query) > 0) {
113             $result =
114             mysql_fetch_object($query);
115         ?>
116         <form action="" method="post">
117         <table border="0" cellpadding="3"
118         cellspacing="0" align="center" width="98%">
119         <tr>
120             <td align="left" colspan="2">
121                 <strong>Személyes adatok</strong>
122             </td>
123         </tr>
124         <tr>
125             <td align="right" width="50%">
126                 Név:
127             </td>
128             <td align="left">
129                 <input type="text"
130                 name="customerName" value="<?=$result-
131                 >customer_name;?>" /> <span
132                 class="require">*</span>
133             </td>
134         </tr>
135         <tr>
136             <td align="right">
137                 E-mail cím:
138             </td>
139             <td align="left">
140                 <input type="text"
141                 name="customerEmail" value="<?=$result-
142                 >customer_email;?>" /> <span
143                 class="require">*</span>
144             </td>
145         </tr>
146         <tr>
147             <td align="right">
148                 Jelszó:
```

```
138         </td>
139         <td align="left">
140             <input type="password"
141             name="customerPass" value="" /> <span
142             class="require">*</span>
143         </td>
144     </tr>
145     <tr>
146         <td align="right">
147             Jelszó ismétlés:
148         </td>
149         <td align="left">
150             <input type="password"
151             name="customerPass2" value="" /> <span
152             class="require">*</span>
153         </td>
154     </tr>
155     <tr>
156         <td align="right">
157             Telefon:
158         </td>
159         <td align="left">
160             <input type="text"
161             name="customerPhone" value="<?=$result-
162             >customer_phone;?>" />
163         </td>
164     </tr>
165     <tr>
166         <td align="left" colspan="2">
167             <strong>Szállítási adatok</strong>
168         </td>
169     </tr>
170     <tr>
171         <td align="right">
172             Irányítószám:
173         </td>
174         <td align="left">
175             <input type="text"
176             name="customerPostcode" value="<?=$result-
177             >customer_postcode;?>" /> <span
178             class="require">*</span>
179         </td>
180     </tr>
181     <tr>
```

```
173         <td align="right">
174             Város:
175         </td>
176         <td align="left">
177             <input type="text"
178             name="customerCity" value="<?=$result-
179             >customer_city;?>" /> <span
180             class="require">*</span>
181         </td>
182     </tr>
183     <tr>
184         <td align="right">
185             Utca, házszám:
186         </td>
187         <td align="left">
188             <input type="text"
189             name="customerAddress" value="<?=$result-
190             >customer_address;?>" /> <span
191             class="require">*</span>
192         </td>
193     </tr>
194     <tr>
195         <td align="right" valign="top">
196             Megjegyzés:
197         </td>
198         <td align="left">
199             <textarea
200             name="customerNotes"><?=$result-
201             >customer_notes;?></textarea>
202         </td>
203     </tr>
204     <tr>
205         <td align="center" colspan="2">
206             <input type="hidden" name="id"
207             value="<?=$result->customer_id;?>" />
208             <input type="submit"
209             name="customerMod" value="Módosítás" />
210             <button
211             onclick="javascript:history.back(); return
212             false;">Vissza</button>
213         </td>
214     </tr>
215 </table>
216 </form>
```

```

205 <?
206         } else {
207             print('Nincs ilyen felhasználó!');
208         }
209     } else {
210         $query = mysql_query("SELECT * FROM
211             ".$GLOBALS["customerTable"]." ORDER BY
212             customer_name");
213         print('<form action="" method="post"
214             name="customerAdd">');
215         print('<a href="#"
216             onclick="document.forms[\'customerAdd\'].submit(
217             ); return false;">Vevő hozzáadása</a>');
218         print('<input type="hidden" name="func"
219             value="customerAdd" />');
220         print('</form>');
221         print('<table border="0" cellpadding="3"
222             cellspacing="0" width="98%" id="cartTable">');
223         print('<tr>');
224         print('<th>Vevő neve</th>');
225         print('<th>Címe</th>');
226         print('<th>Elérhetősége</th>');
227         print('<th>Megjegyzés</th>');
228         print('<th>&nbsp;</th>');
229         print('</tr>');
230         while ($result =
231             mysql_fetch_object($query)) {
232             print('<tr>');
233             print('<td>'.$result->
234                 customer_name.'</td>');
235             print('<td>'.$result->
236                 customer_postcode.
237                 ' '.$result->customer_city.'<br />'.$result->
238                 customer_address.'</td>');
239             print('<td>'.$result->
240                 customer_email.'<br />'.$result->
241                 customer_phone.'</td>');
242             print('<td>'.$result->
243                 customer_note.'</td>');
244             print('<td align="center">');
245             print('<form action="" method="post"
246                 name="customerMod_'.$result->customer_id.'"
247                 style="display: inline;">');

```

```
231         print('<a href="#"  
        onclick="document.forms[\'customerMod_\'. $result-  
>customer_id.\'] submit(); return false;"></a> ');  
232         print('<input type="hidden"  
        name="func" value="customerMod" />');  
233         print('<input type="hidden"  
        name="id" value="\'$result->customer_id.\'" />');  
234         print('</form>');  
235         print('<form action="" method="post"  
        name="customerDel_\'. $result->customer_id.\  
        style="display: inline;">');  
236         print('<a href="#" onclick="if  
        (confirm(\'Biztos benne hogy törli?\'))  
        document.forms[\'customerDel_\'. $result-  
>customer_id.\'] submit(); return false;"></a>');  
237         print('<input type="hidden"  
        name="func" value="customerDel" />');  
238         print('<input type="hidden"  
        name="id" value="\'$result->customer_id.\'" />');  
239         print('</form>');  
240         print('</td>');  
241         print('</tr>');  
242     }  
243     print('</table>');  
244 }  
245 ?>  
246     </div>  
247 </div>
```

A 12. és 107. sorban ellenőrizzük, hogy ki van-e töltve a `$_POST` tömb `func` paramétere. Amennyiben nincs kitöltve, úgy a 209. sortól kezdően lévő kódban kilistázzuk egy táblázatba a rendszerben lévő vevőket. Az utolsó oszlopban egy-egy kis ikonnal jelezzük a módosítás és törlés lehetőségét. A törlésnél a JavaScriptből már ismerős `confirm` paranccsal egy megerősítést kérünk a felhasználótól a végleges törléshez. A táblázat fölött található egy hivatkozás, amivel újabb vevőt tudunk hozzáadni a listához. A kötelező mezőket itt is kis piros csillaggal jelöljük meg.

22. ábra: Új vevő felvitele

Az új vevő hozzáadása a 14. sortól a 105. sorig tartó űrlapon zajlik. Itt a **Hozzáadás** gombra kattintva a `process.php`-ben fog lefutni egy kódrészlet, ami felveszi az új vevőt a táblába, majd a vevők listája oldalra irányít vissza minket. A `process.php` fájlba beillesztendő új kódrészlet itt látható.

```

1  if (isset($_POST["customerAdd"]) &&
    ($_POST["customerAdd"] != '')) &&
    isset($_SESSION["userId"]) &&
    ($_SESSION["userId"] != '')) {
2      if (isset($_POST["customerName"]) &&
        ($_POST["customerName"] != '')) &&
        isset($_POST["customerEmail"]) &&
        ($_POST["customerEmail"] != '')) &&
        isset($_POST["customerPass"]) &&
        ($_POST["customerPass"] != '')) &&
        isset($_POST["customerPostcode"]) &&
        ($_POST["customerPostcode"] != '')) &&
        isset($_POST["customerCity"]) &&
        ($_POST["customerCity"] != '')) &&

```

```
isset($_POST["customerAddress"]) &&
($_POST["customerAddress"] != '')) {
3       if ($_POST["customerPass"] ==
$_POST["customerPass2"]) {
4           $query = mysql_query("SELECT
customer_id FROM ".$GLOBALS["customerTable"]."
WHERE customer_email =
'".$_POST["customerEmail"]."');
5           if (mysql_num_rows($query) == 0) {
6               $query = mysql_query("INSERT INTO
".$GLOBALS["customerTable"]."
(customer_name,customer_email,customer_password,
customer_phone,customer_postcode,customer_city,
customer_address,customer_notes,customer_code)
VALUES ('".$_POST["customerName"]."','"
$_POST["customerEmail"]."','" .md5(
$_POST["customerPass"].$_POST["customerEmail"].
$GLOBALS["secretKey"])."','".$_POST["customerPho
ne"]."
'".$_POST["customerPostcode"]."','"
$_POST["customerCity"]."','".$_POST["customerAdd
ress"]."','".$_POST["customerNotes"]."','".$_code
.'"')");
7               if ($query) {
8                   $message = 'Sikeres regisztrá-
ció!';
9                   unset($_POST);
10                  } else {
11                      $error = 'Hiba történt a re-
gisztráció közben!';
12                  }
13                  } else {
14                      $error = 'Ezzel az e-mail címmel
már regisztráltak a rendszerbe!';
15                  }
16                  } else {
17                      $error = 'A két jelszó nem egyezik
meg!';
18                  }
19                  } else {
20                      $error = 'Töltse ki a kötelező mező-
ket!';
21                  }
22      }
```

Ha nem töltöttünk ki minden mezőt, vagy hiba történt az új vevő felvitele közben, akkor hibajelzést kapunk, egyébként visszatérünk a vevők listájához, ahol már látható is az újonnan felvitt vevő.

Egy vevő módosítása is ugyanebben a fájlban fog végrehajtódni. Itt annyit fogunk tenni, hogy a kiválasztott vevő azonosítóját megkeressük, majd módosítjuk a hozzá tartozó adatokat, majd visszatérünk a listához, ahol már látható lesz a változás. Amennyiben a két jelszó nem egyezik meg, úgy az adatok felvitele sikeres lesz, de a jelszót nem tároljuk el. A vevőket módosító kódrészlet itt látható.

```

1  if (isset($_POST["customerMod"]) &&
    ($_POST["customerMod"] != '') &&
    isset($_POST["id"]) && ($_POST["id"] != '') &&
    isset($_SESSION["userId"]) &&
    ($_SESSION["userId"] != '')) {
2      if (isset($_POST["customerName"]) &&
        ($_POST["customerName"] != '') &&
        isset($_POST["customerEmail"]) &&
        ($_POST["customerEmail"] != '') &&
        isset($_POST["customerPostcode"]) &&
        ($_POST["customerPostcode"] != '') &&
        isset($_POST["customerCity"]) &&
        ($_POST["customerCity"] != '') &&
        isset($_POST["customerAddress"]) &&
        ($_POST["customerAddress"] != '')) {
3          $query = mysql_query("UPDATE
            ".$GLOBALS["customerTable"]." SET customer_name
            = '".$_POST["customerName"]."', customer_email =
            '".$_POST["customerEmail"]."', customer_phone =
            '".$_POST["customerPhone"]."', customer_postcode
            = '".$_POST["customerPostcode"]."',
            customer_city = '".$_POST["customerCity"]."',
            customer_address =
            '".$_POST["customerAddress"]."', customer_notes
            = '".$_POST["customerNotes"]."', customer_code =
            '' WHERE customer_id = ".$_POST["id"]);
4          if ($query) {
5              if (isset($_POST["customerPass"]) &&
                ($_POST["customerPass"] != '') &&
                isset($_POST["customerPass2"]) &&
                ($_POST["customerPass2"] != '')) {
6                  if ($_POST["customerPass"] ==
                    $_POST["customerPass2"]) {

```

```
7          $query = mysql_query("UPDATE
      ".$GLOBALS["customerTable"]." SET
      customer_password =
      '".md5($_POST["customerPass"].$_POST["customerEm
      ail"]).$GLOBALS["secretKey"])."' WHERE customer_id =
      "._POST["id"]);
8          } else {
9              $error = 'A két jelszó nem
      egyezik meg!';
10         }
11     }
12     $message = 'Sikeres vevő módosítás';
13 } else {
14     $error = 'Hiba a vevő módosítása
      közben!';
15 }
16 } else {
17     $error = 'Töltsön ki minden mezőt!';
18 }
19 }
```

A 3. sorban található lekérdezés fogja módosítani a vevők adatait, a 7. sorban lévő lekérdezés pedig akkor fog végrehajtódni, ha az 5–6. sorban lévő feltétel teljesül, azaz kitöltik a jelszó mezőket és a két jelszó megegyezik.

A törlésnél vigyáznunk kell, mivel nem logikai törlésről van szó, hanem fizikairól, tehát a törölt adatok nem lesznek visszaállíthatóak. A vevők törlésének kódja itt található.

```
1  if (isset($_POST["func"]) && ($_POST["func"] ==
      'customerDel') && isset($_POST["id"]) &&
      ($_POST["id"] != '')) &&
      isset($_SESSION["userId"]) &&
      ($_SESSION["userId"] != '')) {
2      $query = mysql_query("DELETE FROM
      ".$GLOBALS["customerTable"]." WHERE customer_id
      = "._POST["id"]);
3      if ($query) {
4          $message = 'Sikeres vevő törlés';
5      } else {
6          $error = 'Hiba a vevő törlése közben!';
7      }
8  }
```

Vevők kilépése

Most, hogy már sikerült belépni a rendszerbe, ki is kell majd tudni jelentkezni belőle. Ez a feldolgozó eljárás is a `process.php`-ben fog helyet foglalni. Mint látható, ez az URL-ben lévő `page` paraméter értékét fogja vizsgálni. Mivel a `process.php` fájl az `index.php` fájl legelején található meg, így nem fog hozzá betöltődni oldal, mert a kérést még az oldal betöltés elején feldolgozzuk. Ehhez írjuk be a következő kódot az előzőleg említett fájlba.

```
1  if (isset($_GET["page"]) && ($_GET["page"] ==  
    'logout')) {  
2      $_SESSION["customerId"] = '';  
3      header('Location: .');  
4  }
```

Mint látható, nem hosszú ez a kód, valójában annyit csinál, hogy a munkamenetből törli a `customerId` változót, majd visszairányít a főoldalra. A 3. sorban lévő `header` függvény fog visszacobni minket a főoldalra, és ezzel törölve minden paramétert az URL-ből.

Felhasználók kilépése

Ugyanezt az admin felületen is le kell programoznunk. Ehhez a következő programkódot helyezzük el az admin könyvtárban lévő `process.php` állományban.

```
1  if (isset($_GET["page"]) && ($_GET["page"] ==  
    'logout')) {  
2      $_SESSION["userId"] = '';  
3      header('Location: .');  
4  }
```

Itt ugyanaz történik, mint az előző esetben, annyi különbséggel, hogy itt a `userId` változó tartalmát töröljük a munkamenetből.

6.3 ÖSSZEFOGLALÁS, KÉRDÉSEK

6.3.1 Összefoglalás

Mint azt észrevehettük, a felhasználók adatait letároljuk a regisztrációkor, így nem kell mindig újra és újra bekérni azokat, amikor rendelni szeretnének egy könyvet. A meglévő adatokból statisztikát is tudunk készíteni, például hogy melyik városból rendeltek a legtöbben, vagy honnan regisztráltak a legtöbben stb.

6.3.2 Önellenőrző kérdések

- Melyek azok a mezők, amelyek kötelezőek egy felhasználó azonosítására?
- Milyen lehetőségeket ismerünk a jelszavak biztonságos tárolására?
- Miért van szükség jogosultságok kialakítására?

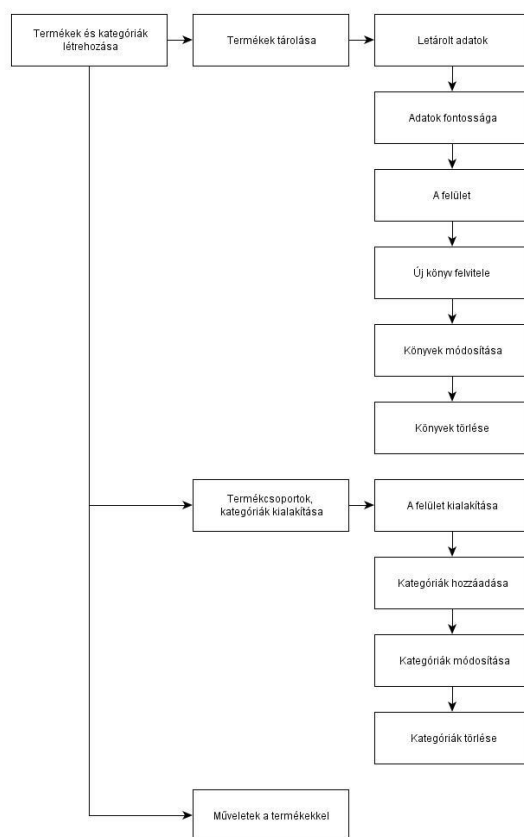
🚩 (Feladatok/Egyetlen állomány feltöltése) Feladat: Töltse fel az eddig elkészült programkódot! (Adminisztrációs felület, felhasználó azonosítás)

7. TERMÉKEK ÉS KATEGÓRIÁK LÉTREHOZÁSA

7.1 CÉLKITŰZÉSEK ÉS KOMPETENCIÁK

A termékek csoportba foglalásához elengedhetetlen, hogy kategóriákat hozzunk létre. Ezáltal csoportosítani tudjuk a könyveket, jobban átlátható lesz a termékinálat. A termékekről letárolt adatok egy részét megjelenítjük egy listás nézetben, és akit jobban érdekel a könyv részletes adatlapja, az megtekintheti a bővebb leírást a könyvről.

7.2 TANANYAG



23. ábra: Lecke áttekintése

7.2.1 Termékek tárolása

Letárolt adatok

A termékek tárolása az adatbázis `book` táblájában történik. Az egyes kategóriák pedig a `category` táblában tárolódnak. A könyvekről a legfontosabb adatokat tároljuk, de ez a lista bővíthető. A letárolt adatok a következők:

- Egyedi azonosító
- Szerző neve
- Könyv címe
- Könyv alcíme
- Kategória azonosítója
- Könyv rövid leírása
- Könyv áfatartalma
- Könyv ára/akciós ára
- Könyv régi ára
- Könyv nyelve
- Könyv EAN száma
- Könyv ISBN száma
- Könyv kiadásának éve
- Könyv kötése
- Könyv oldalszáma
- Könyv mérete
- Könyv súlya

A könyv kategóriáját azért tároljuk másik táblában, hogy a redundanciákat elkerüljük. Gondoljunk csak bele, hogy ha van 3 könyv, amelynek ugyanaz a kategóriája, akkor nem kell három helyen átírnunk a kategóriáját, hanem elég egy helyen, ott ahol hivatkozunk a könyv kategóriájára.

Adatok fontossága

Gondolom, az mindenkinek egyértelmű, hogy a **könyvek kategóriákba** sorolása elengedhetetlen, mivel ha nem lennének kategóriák, akkor az összes könyv egy kupacban lenne, és nehéz lenne a természettudományi és szépirodalmi könyvek között keresni például egy Arany János Összes versei kötetet.

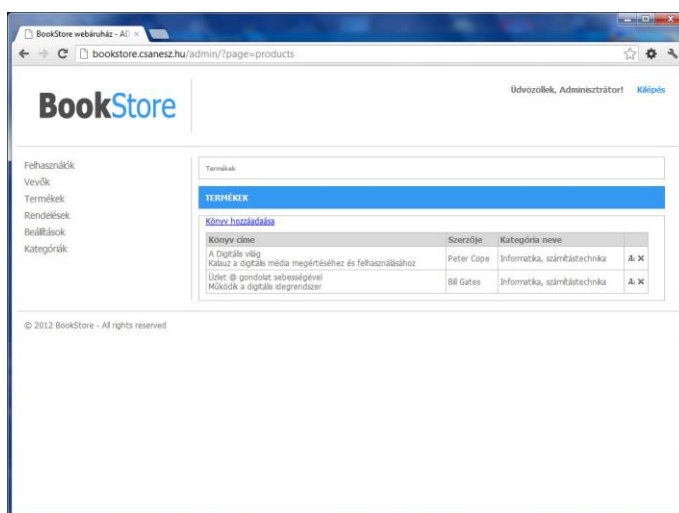
Megfigyelhető, hogy a **könyv ára** két külön mezőben van letárolva. Erre azért van szükség, mivel egy kategória listázásánál megjelenő könyveknél lehet rendezni ár szerint a könyveket. Ha a **könyv régi árát** felülírjuk, akkor már nem tudjuk, hogy mennyi volt a régi ára. Amennyiben az akciós árat egy másik mezőben rögzítjük, akkor a könyvek listájánál ár szerinti rendezéskor nem tudjuk figyelembe venni az akciós árat. Emiatt egy kis furfangosság szerepel az áraknál. Úgy oldjuk meg az árak kérdését, hogy ha egy könyv ára nem akciós, akkor az ár mezőben szerel az ára, az akciós mező pedig üres. Innen tudjuk, hogy nem akciós. Amennyiben akciós a könyv, a régi árat átmozgatjuk az akciós mezőbe, erről már tudni fogjuk, hogy akciós, mivel nem üres a mező, és az új, akciós árat fogjuk beírni az ár mezőbe. Ezáltal már az ár szerinti rendezésnél is helyesen fogja rendezni a rendszer a könyveket.

Az **ISBN** és **EAN** számok is egyedi azonosítóként funkcionálnak a rendszerben. Azért is van erre szükségünk, mert ha a rendszert egy számlázó vagy készletnyilvántartó rendszerrel fogjuk összekötni, akkor a vonalkódolvasó az EAN számot azonnal beolvassa, és értelmezi, ezzel is megkönnyítve a két rendszer közötti kapcsolatot.

A **könyv mérete** és **súlya** akkor lehet hasznos számunkra, hogyha a szállításkor figyelembe kell vennünk ezeket az információkat.

A felület

A termékek egy listás felületen fognak megjelenni, ahol új terméket tudunk felvenni, illetve régiakat tudjuk módosítani vagy törölni.



24. ábra: Termékek listás megjelenése

Az admin/pages könyvtárban létre kell hoznunk egy productsPage.php fájlt, aminek a tartalma a következő lesz.

```

5      <div id="contentColumn">
6          <div id="contentMenu">
7              Termékek
8          </div>
9          <div id="contentTitle">
10             Termékek
11          </div>
12          <div id="contentBox">
13              <div
14                  class="error"><?=$error;?></div>
15              <div
16                  class="message"><?=$message;?></div>
17              <?
18              if (isset($_POST["func"]) && ($_POST["func"] ==
19                  'bookAdd')) {
20              ?>
21                  <form action="" method="post"
22                      enctype="multipart/form-data">
23                      <table border="0" cellpadding="3"
24                          cellspacing="0" align="center" width="98%">
25                          <tr>
26                              <td align="right" width="50%">
27                                  Szerzője:
28                              </td>
29                              <td align="left">
30                                  <input type="text" name="bookAuthor"
31                                  value="" /> <span class="require">*</span>
32                              </td>
33                          </tr>
34                          <tr>
35                              <td align="right">
36                                  Címe:
37                              </td>
38                              <td align="left">
39                                  <input type="text" name="bookTitle"
40                                  value="" /> <span class="require">*</span>
41                              </td>
42                          </tr>
43                          <tr>
44                              <td align="right">
45                                  Alcíme:
46                              </td>

```

```

40         <td align="left">
41             <input type="text"
42             name="bookSubtitle" value="" />
43         </td>
44     </tr>
45     <tr>
46         <td align="right">
47             Kategóriája:
48         </td>
49         <td align="left">
50             <select name="bookCategory">
51                 <option value="0">- Vá-
52                 lasszon -</option>
53                 <?
54                     $query = mysql_query("SELECT *
55                     FROM ".$GLOBALS["categoryTable"]." ORDER BY
56                     category_name");
57                     if (mysql_num_rows($query) >
58                     0) {
59                         while ($result =
60                         mysql_fetch_object($query)) {
61                             print('<option
62                             value="'. $result->category_id.'">'.
63                             $result->category_name.'</option>');
64                         }
65                     }
66                     ?>
67                 </select>
68                 <span class="require">*</span>
69             </td>
70         </tr>
71     <tr>
72         <td align="right" valign="top">
73             Borítója:
74         </td>
75         <td align="left" valign="top">
76             <input type="file" name="bookImg"
77             size="15" /> <span class="require">*</span>
78         </td>
79     </tr>
80 </tr>
81 </table>

```



```
112         <td align="right">
113             EAN száma:
114         </td>
115         <td align="left">
116             <input type="text" name="bookEAN"
117             value="" />
118         </td>
119     </tr>
120     <tr>
121         <td align="right">
122             ISBN száma:
123         </td>
124         <td align="left">
125             <input type="text" name="bookISBN"
126             value="" />
127         </td>
128     </tr>
129     <tr>
130         <td align="right">
131             Kiadási éve:
132         </td>
133         <td align="left">
134             <input type="text" name="bookYear"
135             value="" size="4" maxlength="4" />
136         </td>
137     </tr>
138     <tr>
139         <td align="right">
140             Kötése:
141         </td>
142         <td align="left">
143             <input type="text"
144             name="bookBinding" value="" />
145         </td>
146     </tr>
147     <tr>
148         <td align="right">
149             Oldalszáma:
150         </td>
151         <td align="left">
152             <input type="text" name="bookPage"
153             value="" />
154         </td>
155     </tr>
```

```

151         <tr>
152             <td align="right">
153                 Mérete:
154             </td>
155             <td align="left">
156                 <input type="text" name="bookSize"
157                 value="" />
158             </td>
159         </tr>
160         <tr>
161             <td align="right" valign="top">
162                 Súly:
163             </td>
164             <td align="left">
165                 <input type="text"
166                 name="bookWeight" value="" size="4"
167                 maxlength="4" /> g
168             </td>
169         </tr>
170         <tr>
171             <td align="center" colspan="2">
172                 <input type="submit"
173                 name="bookAdd" value="Hozzáadás" />
174                 <button
175                 onclick="javascript:history.back(); return
176                 false;">Vissza</button>
177             </td>
178         </tr>
179     </table>
180 </form>
181 <?
182     } elseif (isset($_POST["func"]) &&
183     ($_POST["func"] == 'bookMod')) {
184         $query = mysql_query("SELECT * FROM
185         ".$GLOBALS["bookTable"]." WHERE book_id =
186         ".$_POST["id"]);
187         if (mysql_num_rows($query) > 0) {
188             $result =
189             mysql_fetch_object($query);
190             $price = ($result->book_special !=
191             '0')?$result->book_special:$result->book_price;
192             $special = ($result->book_special !=
193             '0')?$result->book_price:"";
194         }
195     }
196 }

```

```

183      <form action="" method="post"
      enctype="multipart/form-data">
184      <table border="0" cellpadding="3"
      cellspacing="0" align="center" width="98%">
185          <tr>
186              <td align="right" width="50%">
187                  Szerzője:
188              </td>
189              <td align="left">
190                  <input type="text"
      name="bookAuthor" value="<?=$result-
      >book_author;?>" /> <span
      class="require">*</span>
191              </td>
192          </tr>
193          <tr>
194              <td align="right">
195                  Címe:
196              </td>
197              <td align="left">
198                  <input type="text" name="bookTitle"
      value="<?=$result->book_title;?>" /> <span
      class="require">*</span>
199              </td>
200          </tr>
201          <tr>
202              <td align="right">
203                  Alcíme:
204              </td>
205              <td align="left">
206                  <input type="text"
      name="bookSubtitle" value="<?=$result-
      >book_subtitle;?>" />
207              </td>
208          </tr>
209          <tr>
210              <td align="right">
211                  Kategóriája:
212              </td>
213              <td align="left">
214                  <input type="text"
      name="bookCategory" value="<?=$result-
      >book_category;?>" />
215                  <select name="bookCategory">

```

```

216                                     <option value="0">- Vá-
lasszon -</option>
217                                     <?
218                                     $query1 =
mysql_query("SELECT * FROM
                ".$GLOBALS["categoryTable"]." ORDER BY
                category_name");
219                                     if
                (mysql_num_rows($query1) > 0) {
220                                     while ($result1 =
mysql_fetch_object($query1)) {
221                                     print('<option
value="'.$result1->category_id.'");
222                                     if ($result-
>book_category == $result1->category_id)
223                                     print('
selected="selected"');
224                                     print('>'.
$result1->category_name.'</option>');
225                                     }
226                                     }
227                                     ?>
228                                     </select>
229                                     <span class="require">*</span>
230                                     </td>
231                                     </tr>
232                                     <tr>
233                                     <td align="right" valign="top">
234                                     Borítója:
235                                     </td>
236                                     <td align="left" valign="top">
237                                     <input type="file" name="bookImg"
/> <span class="require">*</span>
238                                     </td>
239                                     </tr>
240                                     <tr>
241                                     <td align="right" valign="top">
242                                     Leírása:
243                                     </td>
244                                     <td align="left" valign="top">
245                                     <textarea name="bookDesc"
rows="10"><?=$result->book_desc;?></textarea>
<span class="require">*</span>
246                                     </td>

```

```
247         </tr>
248         <tr>
249             <td align="right">
250                 Áfa tartalma:
251             </td>
252             <td align="left">
253                 <input type="text" name="bookTax"
254                 value="<?=$result->book_tax;?>" /> % <span
255                 class="require">*</span>
256             </td>
257         </tr>
258         <tr>
259             <td align="right">
260                 Ára:
261             </td>
262             <td align="left">
263                 <input type="text" name="bookPrice"
264                 value="<?=$price;?>" /> Ft <span
265                 class="require">*</span>
266             </td>
267         </tr>
268         <tr>
269             <td align="right">
270                 Akciós ára:
271             </td>
272             <td align="left">
273                 <input type="text"
274                 name="bookSpecial" value="<?=$special;?>" /> Ft
275             </td>
276         </tr>
277         <tr>
278             <td align="right">
279                 Nyelve:
280             </td>
281             <td align="left">
282                 <input type="text" name="bookLang"
283                 value="<?=$result->book_lang;?>" />
284             </td>
285         </tr>
286         <tr>
287             <td align="right">
288                 EAN száma:
289             </td>
290             <td align="left">
```

```

285         <input type="text" name="bookEAN"
      value="<?=$result->book_ean;?>" />
286     </td>
287 </tr>
288 <tr>
289     <td align="right">
290         ISBN száma:
291     </td>
292     <td align="left">
293         <input type="text" name="bookISBN"
      value="<?=$result->book_isbn;?>" />
294     </td>
295 </tr>
296 <tr>
297     <td align="right">
298         Kiadási éve:
299     </td>
300     <td align="left">
301         <input type="text" name="bookYear"
      value="<?=$result->book_year;?>" />
302     </td>
303 </tr>
304 <tr>
305     <td align="right">
306         Kötése:
307     </td>
308     <td align="left">
309         <input type="text"
      name="bookBinding" value="<?=$result-
      >book_binding;?>" />
310     </td>
311 </tr>
312 <tr>
313     <td align="right">
314         Oldalszáma:
315     </td>
316     <td align="left">
317         <input type="text" name="bookPage"
      value="<?=$result->book_page;?>" /> p
318     </td>
319 </tr>
320 <tr>
321     <td align="right">
322         Mérete:

```

```
323         </td>
324         <td align="left">
325             <input type="text" name="bookSize"
326             value="<?=$result->book_size;?>" />
327         </td>
328     </tr>
329     <tr>
330         <td align="right" valign="top">
331             Súly:
332         </td>
333         <td align="left">
334             <input type="text"
335             name="bookWeight" value="<?=$result-
336             >book_weight;?>" /> g
337         </td>
338     </tr>
339     <tr>
340         <td align="center" colspan="2">
341             <input type="hidden" name="id"
342             value="<?=$result->book_id;?>" />
343             <input type="submit"
344             name="bookMod" value="Módosítás" />
345             <button
346             onclick="javascript:history.back(); return
347             false;">Vissza</button>
348         </td>
349     </tr>
350 </table>
351 </form>
352 <?
353     } else {
354         print('Nincs ilyen könyv!');
355     }
356 } else {
357     $query = mysql_query("SELECT * FROM
358     ".$GLOBALS["bookTable"]." INNER JOIN
359     ".$GLOBALS["categoryTable"]." ON book_category =
360     category_id ORDER BY book_title");
361     print('<form action="" method="post"
362     name="bookAdd">');
363     print('<a href="#"
364     onclick="document.forms[\'bookAdd\'].submit();
365     return false;">Könyv hozzáadása</a>');
366 }
```

```

353         print('<input type="hidden" name="func"
value="bookAdd" />');
354         print('</form>');
355         print('<table border="0" cellpadding="3"
cellspacing="0" width="98%" id="cartTable">');
356         print('<tr>');
357         print('<th>Könyv címe</th>');
358         print('<th>Szerzője</th>');
359         print('<th>Kategória neve</th>');
360         print('<th>&nbsp;  </th>');
361         print('</tr>');
362         while ($result =
mysql_fetch_object($query)) {
363             print('<tr>');
364             print('<td>'.$result->
book_title.'<br />'.$result->
book_subtitle.'</td>');
365             print('<td>'.$result->
book_author.'</td>');
366             print('<td>'.$result->
category_name.'</td>');
367             print('<td align="center">');
368             print('<form action="" method="post"
name="bookMod_'.$result->book_id.'"
style="display: inline;">');
369             print('<a href="#"
onclick="document.forms[\'bookMod_\'
.$result->book_id.\']
.submit(); return false;"></a> ');
370             print('<input type="hidden"
name="func" value="bookMod" />');
371             print('<input type="hidden"
name="id" value="\'
.$result->book_id.'" />');
372             print('</form>');
373             print('<form action="" method="post"
name="bookDel_\'
.$result->book_id.'"
style="display: inline;">');
374             print('<a href="#" onclick="if
(confirm(\'Biztos benne hogy törli?\'))
document.forms[\'bookDel_\'
.$result->book_id.\']
.submit(); return false;"></a>');

```

```

375         print('<input type="hidden"
    name="func" value="bookDel" />');
376         print('<input type="hidden"
    name="id" value="'. $result->book_id.'" />');
377         print('</form>');
378         print('</td>');
379         print('</tr>');
380     }
381     print('</table>');
382 }
383 ?>
384     </div>
385 </div>

```

Ha jól megfigyeljük a kódot, akkor láthatjuk, hogy a felvitel a 12–172. sorig, a módosítás a 172–345. sorig, a listázás pedig a 345. sortól a fájl végéig tart. A hozzáadás és módosítás részben is a kategóriák listáját is legeneráljuk. A módosításnál pedig meg is jelöljük az éppen kiválasztott könyv kategóriáját. Mind a két lehetőségnél tudunk feltölteni borítót a könyvhöz. A rendszer automatikusan átméretezi a megfelelő méretűre a feltöltött képet. A módosításnál a régi képet törli, és az újat feltölti. A kötelező mezők, itt is kis piros csillaggal vannak megjelölve.

Új könyv felvitel

A termékek megjelenítése ebben a fájlban van megírva, viszont az új könyv hozzáadása, a régi könyvek módosítása és a könyvek törlésének a feldolgozása az `admin/process.php` fájlban foglal helyet.

A könyv felvitelének kódja itt látható:

```

1  if (isset($_POST["bookAdd"]) &&
    ($_POST["bookAdd"] != '') &&
    isset($_SESSION["userId"]) &&
    ($_SESSION["userId"] != '')) {
2      if (isset($_FILES["bookImg"]) &&
        ($_FILES["bookImg"]["name"] != '') &&
        isset($_POST["bookAuthor"]) &&
        ($_POST["bookAuthor"] != '') &&
        isset($_POST["bookTitle"]) &&
        ($_POST["bookTitle"] != '') &&
        isset($_POST["bookCategory"]) &&
        ($_POST["bookCategory"] != '0') &&
        isset($_POST["bookDesc"]) && ($_POST["bookDesc"]
        != '') && isset($_POST["bookTax"]) &&

```

```

    ($_POST["bookTax"] != '') &&
    isset($_POST["bookPrice"]) &&
    ($_POST["bookPrice"] != '')) {
3         $price = (isset($_POST["bookSpecial"])
    && ($_POST["bookSpecial"] !=
    '')) ? $_POST["bookSpecial"] : $_POST["bookPrice"];
4         $special = (isset($_POST["bookSpecial"])
    && ($_POST["bookSpecial"] !=
    '')) ? $_POST["bookPrice"] : "0";
5 $query = mysql_query("INSERT INTO
    ".$GLOBALS["bookTable"]."
    (book_id,book_author,book_title,book_subtitle,
    book_category,book_desc,book_tax,book_price,
    book_special,book_lang,book_ean,book_isbn,book_y
    ear,
    book_binding,book_page,book_size,book_weight)
    VALUES
    ('', ''.$_POST["bookAuthor"].'', ''.$_POST["bookTi
    tle"].
    '', ''.$_POST["bookSubtitle"].'', ''.$
    _POST["bookCategory"].'', ''.$_POST["bookDesc"].
    '', ''.$
    _POST["bookTax"].'', ''.$price.'', ''.$
    _special.'', ''.$_POST["bookLang"].'', ''.$
    _POST["bookEAN"].'', ''.$_POST["bookISBN"].'', ''.$
    _POST["bookYear"].'', ''.$_POST["bookBinding"].'
    ', ''.$
    _POST["bookPage"].'', ''.$_POST["bookSize"].'', ''.$
    _POST["bookWeight"].'')");
6
    img_resize($_FILES["bookImg"]["tmp_name"], 80,
    '../images/book/', 'book_'.$mysql_insert_id().
    '_small.jpg');
7
    img_resize($_FILES["bookImg"]["tmp_name"], 200,
    '../images/book/', 'book_'.$mysql_insert_id().
    '_big.jpg');
8         if ($query) {
9             $message = 'Sikeres felvitel!';
10            unset($_POST);
11        } else {

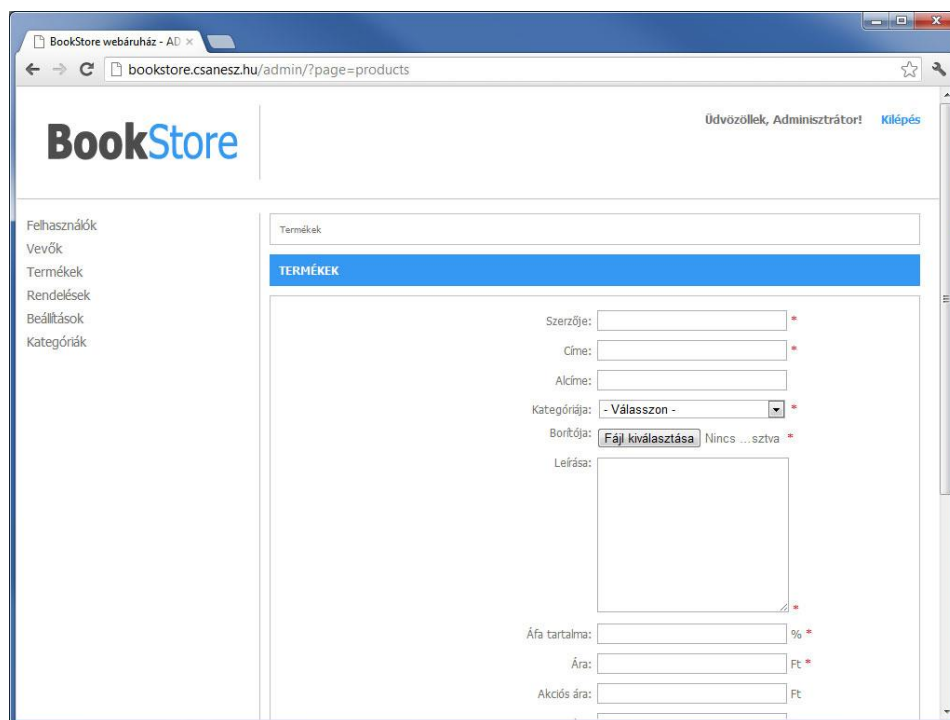
```

```

12         $error = 'Hiba történt a felvitel
közben!';
13     }
14 } else {
15     $error = 'Töltse ki a kötelező mező-
ket!';
16 }
17 }

```

A 2. sorban ellenőrizzük, hogy a kötelező mezőket kitöltötték-e. A 3–4. sorban állítjuk elő sima és akciós árakat. Az 5. sorban megkíséreljük felvenni az adatokat a táblába, majd a 6. és 7. sorban megtörténik a képeknek az átméretezése két különböző méretre.



25. ábra: Új könyv felvitele

A kisebb a listás nézetben, a nagyobb kép a részletes adatlapon fog megjelenni. Az ehhez kapcsolódó függvényt a gyökérkönyvtárban található `function.php` fájlban fogjuk elhelyezni.

```

1 function img_resize($tmpname, $size, $save_dir,
    $save_name, $maxisheight = 0) {

```

```

2      $save_dir      .= ( substr($save_dir,-1) !=
   "/" ) ? "/" : "";
3      $gis           = getimagesize($tmpname);
4      $type          = $gis[2];
5      switch($type) {
6          case "1": $imorig =
imagecreatefromgif($tmpname); break;
7          case "2": $imorig =
imagecreatefromjpeg($tmpname);break;
8          case "3": $imorig =
imagecreatefrompng($tmpname); break;
9          default: $imorig =
imagecreatefromjpeg($tmpname);
10     } // switch
11
12     $x = imagesx($imorig);
13     $y = imagesy($imorig);
14     $woh = (!$maxisheight)? $gis[0] : $gis[1] ;
15
16     if ($woh <= $size) {
17         $aw = $x;
18         $ah = $y;
19     } else {
20         if (!$maxisheight) {
21             $aw = $size;
22             $ah = $size * $y / $x;
23         } else {
24             $aw = $size * $x / $y;
25             $ah = $size;
26         } // if
27     } // if
28
29     $im = imagecreatetruecolor($aw,$ah);
30     if (imagecopyresampled($im,$imorig ,
0,0,0,0,$aw,$ah,$x,$y))
31         if (imagejpeg($im,
$save_dir.$save_name))
32             return true;
33         else
34             return false;
35 }

```

Ennek a függvénynek, több paramétere is van. A első paraméter várja az ideiglenes könyvtárba feltöltött kép nevét, majd második paraméterben meg

kell adni a kép szélességét vagy hosszúságát képpontokban. Ehhez kapcsolódik az utolsó paraméter is, ami azt határozza meg, hogy az előbb megadott szám a maximális képmagasság-e? Itt alapértelmezésben a kép szélessége van figyelembe véve. Ha szeretnénk, hogy a kép magassága ne legyen nagyobb mint a megadott méret, akkor az utolsó paramétert állítsuk 1-re. A 3. és 4. paraméter pedig a kép mentési útvonalát, illetve nevét kéri. A függvény elvileg, képes GIF és PNG formátumú képet is fogadni, de inkább maradjunk a JPG formátumnál, mivel az előző kettő lehet áttetsző is, és ebből érdekes képek alakulhatnak ki, mivel a függvényben még nincsen lekezelve a képek alfa csatornája, ami az áttetszőséget szabályozza.

Könyvek módosítása

A termékek módosítása is az `admin/process.php` fájlban fog helyet kapni.

```
1  if (isset($_POST["bookMod"]) &&
    ($_POST["bookMod"] != '') &&
    isset($_SESSION["userId"]) &&
    ($_SESSION["userId"] != '')) {
2      if (isset($_POST["bookAuthor"]) &&
          ($_POST["bookAuthor"] != '') &&
          isset($_POST["bookTitle"]) &&
          ($_POST["bookTitle"] != '') &&
          isset($_POST["bookCategory"]) &&
          ($_POST["bookCategory"] != '0') &&
          isset($_POST["bookDesc"]) && ($_POST["bookDesc"]
            != '') && isset($_POST["bookTax"]) &&
          ($_POST["bookTax"] != '') &&
          isset($_POST["bookPrice"]) &&
          ($_POST["bookPrice"] != '')) {
3          $price = (isset($_POST["bookSpecial"])
&& ($_POST["bookSpecial"] !=
  '')) ? $_POST["bookSpecial"] : $_POST["bookPrice"];
4          $special = (isset($_POST["bookSpecial"])
&& ($_POST["bookSpecial"] !=
  '')) ? $_POST["bookPrice"] : "0";
5          $query = mysql_query("UPDATE
  ".$GLOBALS["bookTable"]." SET book_author =
  '".$_POST["bookAuthor"]."', book_title =
  '".$_POST["bookTitle"]."', book_subtitle =
  '".$_POST["bookSubtitle"]."', book_category =
  '".$_POST["bookCategory"]."', book_desc =
```

```

        ".$_POST["bookDesc"]."', book_tax =
        ".$_POST["bookTax"]."', book_price =
        ".$price."', book_special = ".$special."',
book_lang = ".$_POST["bookLang"]."', book_ean =
        ".$_POST["bookEAN"]."', book_isbn =
        ".$_POST["bookISBN"]."', book_year =
        ".$_POST["bookYear"]."', book_binding =
        ".$_POST["bookBinding"]."', book_page =
        ".$_POST["bookPage"]."', book_size =
        ".$_POST["bookSize"]."', book_weight =
        ".$_POST["bookWeight"]."' WHERE book_id =
        ".$_POST["id"]);
6         if (isset($_FILES["bookImg"]) &&
        ($_FILES["bookImg"]["name"] != '')) {
7
            unlink('../images/book/book_'.$_POST["id"].
            '_small.jpg');
8
            unlink('../images/book/book_'.$_POST["id"].
            '_big.jpg');
9
            img_resize($_FILES["bookImg"]["tmp_name"],80,
            '../images/book/', 'book_'.$_POST["id"].'_small.j
            pg');
10
            img_resize($_FILES["bookImg"]["tmp_name"],200,
            '../images/book/', 'book_'.$_POST["id"].'_big.jpg
            ');
11
        }
12
        if ($query) {
13
            $message = 'Sikeres módosítás!';
14
            unset($_POST);
15
        } else {
16
            $error = 'Hiba történt a módosítás
közben!';
17
        }
18
    } else {
19
        $error = 'Töltse ki a kötelező mező-
ket!';
20
    }
21
}

```

Itt is a 2. sorban ellenőrizzük a kötelező mezőket, majd a 3. és 4. sorban előállítjuk a sima és akciós árakat.

The screenshot shows the 'BookStore' web application in an admin mode. The browser address bar shows 'bookstore.csanesz.hu/admin/?page=products'. The page title is 'BookStore'. On the right, it says 'Üdvözzök, Adminisztrátor!' and 'Kilépés'. On the left, there is a sidebar with links: 'Felhasználók', 'Vevők', 'Termékek', 'Rendelések', 'Beállítások', and 'Kategóriák'. The main content area is titled 'Termékek' and 'TERMÉKEK'. It displays a form for editing a product. The form fields are: 'Szerzője: Peter Cope', 'Címe: A Digitális világ', 'Alcíme: Kalauz a digitális média megértés', 'Kategóriája: Informatika, számítástechnika', 'Borítója: Fáj kiválasztása', 'Leírása: A több mint 200 színes kép mellett lépésről lépésre ismertetett technikák, egyedi ötletek és kipróbált módszerek segítenek bennünket abban, hogy valóban a digitális világ urai legyünk. A könyv egyszerűsége, a', 'Áfa tartalma: 5%', and 'Ára: 3332 Ft'.

26. ábra: Könyv módosítása

Amennyiben töltenek fel új képet, a régieket töröljük, és feltöltjük az új képeket.

Könyvek törlése

A könyvek törlésének kódja is itt van letárolva. Ennek kódja itt látható:

```

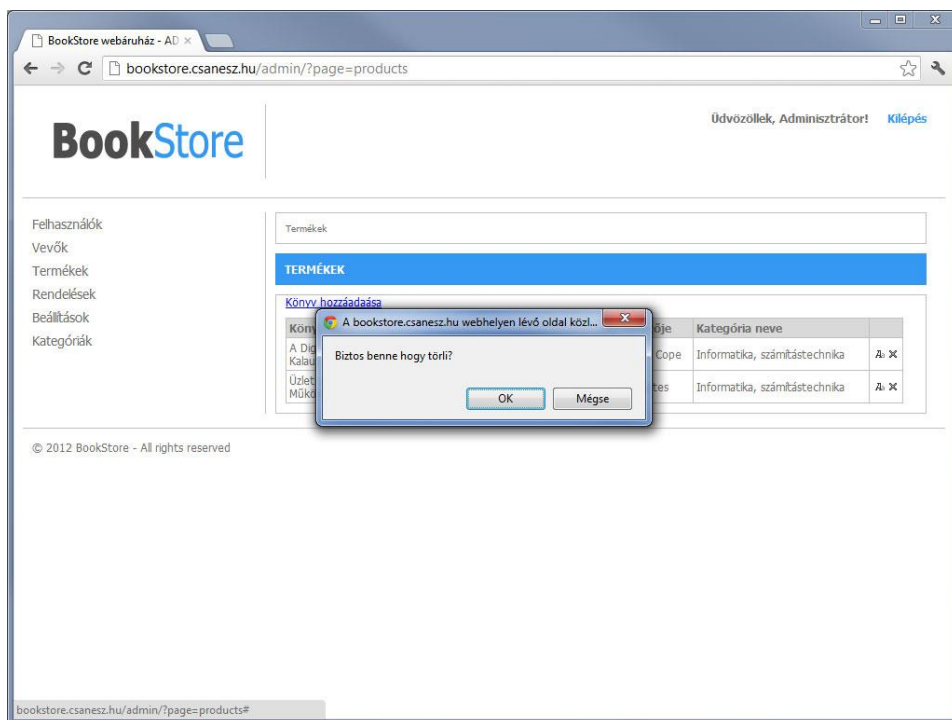
1  if (isset($_POST["func"]) && ($_POST["func"] ==
    'bookDel') && isset($_POST["id"]) &&
    ($_POST["id"] != '')) &&
    isset($_SESSION["userId"]) &&
    ($_SESSION["userId"] != '')) {
2      $query = mysql_query("DELETE FROM
    ".$GLOBALS["bookTable"]." WHERE book_id =
    ".$_POST["id"]);
3      if ($query) {
4          unlink('../images/book/book_'.$_POST["id"].
            '_small.jpg');

```

```

5      unlink('../images/book/book_'.$_POST["id"].
      '_big.jpg');
6      $message = 'Sikeres könyv törlés';
7      } else {
8      $error = 'Hiba a könyv törlése közben!';
9      }
10 }

```



27. ábra: Könyv törlése

Ha töröljük a könyvet, akkor vele együtt eltávolítjuk a képeket is.

7.2.2 Termékcsoportok, kategóriák módosítása

A felület kialakítása

A termékcsoportok táblája az adatbázisban a `category` tábla. Itt tartjuk nyilván az egyes kategóriákat amelyekbe a könyveket soroljuk be. A kategóriák listázását már megírtuk, de ahhoz hogy a kategóriákat módosítani is tudjuk, az adminisztrációs felületen ki kell alakítani hozzá egy felületet. Az

admin/pages könyvtárban hozzuk létre a categoryPage.php fájlt a következő tartalommal.

```
1      <div id="contentColumn">
2          <div id="contentMenu">
3              Termékkategóriák
4          </div>
5          <div id="contentTitle">
6              Termékkategóriák
7          </div>
8          <div id="contentBox">
9              <div class="error"><?=$error;?></div>
10             <div
11                 class="message"><?=$message;?></div>
12 <?
13 if (isset($_POST["func"]) && ($_POST["func"] ==
14     'categoryAdd')) {
15 >?
16     <form action="" method="post">
17         <table border="0" cellpadding="3"
18             cellspacing="0" align="center" width="98%">
19             <tr>
20                 <td align="right" width="50%">
21                     Kategória neve:
22                 </td>
23                 <td align="left">
24                     <input type="text"
25                     name="categoryName" value="" /> <span
26                     class="require">*</span>
27                 </td>
28             </tr>
29             <tr>
30                 <td align="center" colspan="2">
31                     <input type="submit"
32                     name="categoryAdd" value="Hozzáadás" />
33                     <button
34                     onclick="javascript:history.back(); return
35                     false;">Vissza</button>
36                 </td>
37             </tr>
38         </table>
39     </form>
40 <?
41 } elseif (isset($_POST["func"]) &&
42     ($_POST["func"] == 'categoryMod')) {
```

```

34     $query = mysql_query("SELECT * FROM
    ".$GLOBALS["categoryTable"]." WHERE category_id
    = ".$_POST["id"]);
35     if (mysql_num_rows($query) > 0) {
36         $result = mysql_fetch_object($query);
37     }
38     <form action="" method="post">
39         <table border="0" cellpadding="3"
    cellspacing="0" align="center" width="98%">
40             <tr>
41                 <td align="right" width="50%">
42                     Kategória neve:
43                 </td>
44                 <td align="left">
45                     <input type="text"
    name="categoryName" value="<?=$result->
    category_name;?>" /> <span
    class="require">*</span>
46                 </td>
47             </tr>
48             <tr>
49                 <td align="center" colspan="2">
50                     <input type="hidden" name="id"
    value="<?=$result->category_id;?>" />
51                     <input type="submit"
    name="categoryMod" value="Módosítás" />
52                     <button
    onclick="javascript:history.back(); return
    false;">Vissza</button>
53                 </td>
54             </tr>
55         </table>
56     </form>
57 <?
58     } else {
59         print('Nincs ilyen kategória!');
60     }
61 } else {
62     $query = mysql_query("SELECT * FROM
    ".$GLOBALS["categoryTable"]." ORDER BY
    category_name");
63     print('<form action="" method="post"
    name="categoryAdd_'.$result->category_id.'">');

```

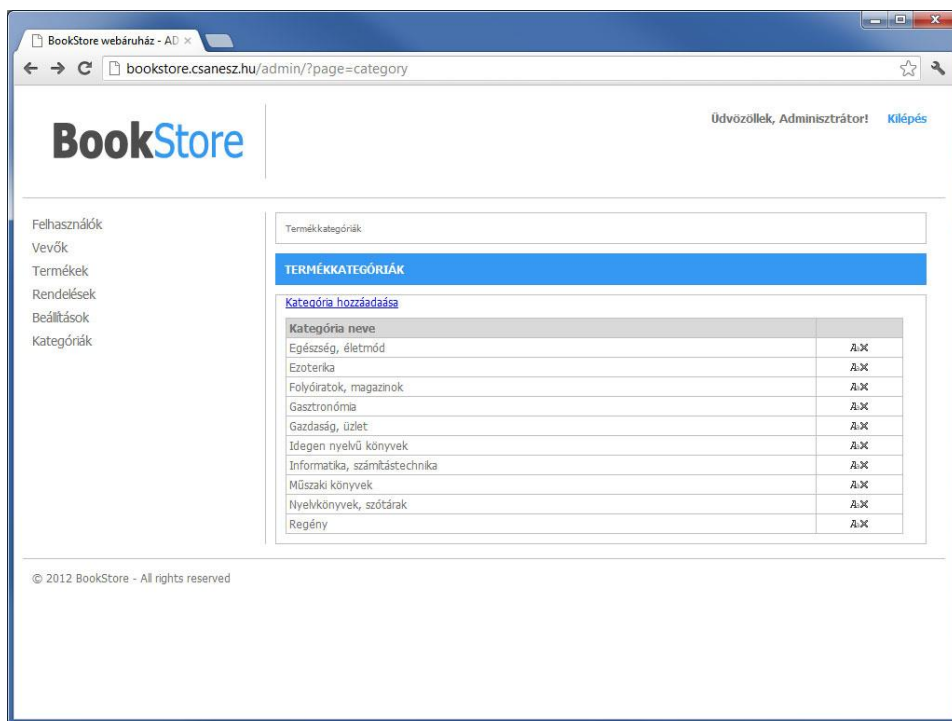
```
64     print('<a href="#"
onclick="document.forms[\'categoryAdd_\'.$result-
>category_id.\'] submit(); return
false;">Kategória hozzáadása</a>');
65     print('<input type="hidden" name="func"
value="categoryAdd" />');
66     print('</form>');
67     print('<table border="0" cellpadding="3"
cellspacing="0" width="98%" id="cartTable">');
68     print('<tr>');
69     print('<th>Kategória neve</th>');
70     print('<th>&nbsp;</th>');
71     print('</tr>');
72     while ($result =
mysql_fetch_object($query)) {
73         print('<tr>');
74         print('<td>'.$result-
>category_name.'</td>');
75         print('<td align="center">');
76         print('<form action="" method="post"
name="categoryMod_\'.$result->category_id.'"
style="display: inline;">');
77         print('<a href="#"
onclick="document.forms[\'categoryMod_\'.$result-
>category_id.\'] submit(); return false;"></a>');
78         print('<input type="hidden" name="func"
value="categoryMod" />');
79         print('<input type="hidden" name="id"
value="'.$result->category_id.'" />');
80         print('</form>');
81         print('<form action="" method="post"
name="categoryDel_\'.$result->category_id.'"
style="display: inline;">');
82         print('<a href="#" onclick="if
(confirm(\'Biztos benne hogy törli?\'))
document.forms[\'categoryDel_\'.$result-
>category_id.\'] submit(); return false;"></a>');
83         print('<input type="hidden" name="func"
value="categoryDel" />');
```

```

84         print('<input type="hidden" name="id"
      value="'. $result->category_id.'" />');
85         print('</form>');
86         print('</td>');
87         print('</tr>');
88     }
89     print('</table>');
90 }
91 ?>
92     </div>
93 </div>

```

A 12. és 33. sorban ellenőrizzük, hogy ki van-e töltve a `$_POST` tömb `func` paramétere. Amennyiben nincs kitöltve, úgy a 62. sortól kezdően lévő kódban kilistázzuk egy táblázatba a feltöltött kategóriákat. Az utolsó oszlopban egy-egy kis ikonnal jelezzük a módosítás és törlés lehetőségét. A töltésnél a JavaScriptből már ismerős `confirm` paranccsal egy megerősítést kérünk a felhasználótól a végleges törléshez.



28. ábra: Termékcsoportok listázása

A táblázat fölött található egy hivatkozás, amivel újabb kategóriát tudunk hozzáadni a listához. A kötelező mezőket itt is kis piros csillaggal jelöljük meg.

Kategóriák hozzáadása

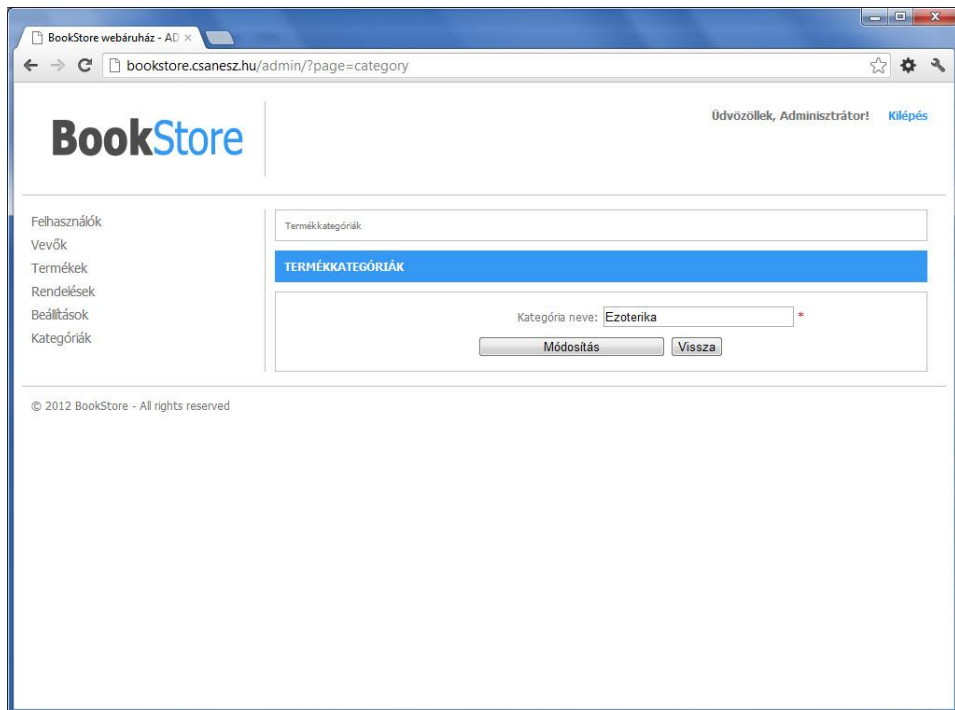
Az új kategória hozzáadása a 14. sortól a 31. sorig tart. Itt a *Hozzáadás* gombra kattintva a `process.php`-ben fog lefutni egy kódrészlet, amely felveszi az új kategóriát a táblába, majd a kategóriák listája oldalra irányít vissza minket. A `process.php` fájlba beillesztendő új kódrészlet itt látható.

```
1  if (isset($_POST["categoryAdd"]) &&
    ($_POST["categoryAdd"] != '') &&
    isset($_SESSION["userId"]) &&
    ($_SESSION["userId"] != '')) {
2      if (isset($_POST["categoryName"]) &&
        ($_POST["categoryName"] != '')) {
3          $query = mysql_query("INSERT INTO
        ".$GLOBALS["categoryTable"]." (category_name)
        VALUES ('".$_POST["categoryName"]."')");
4          if ($query) {
5              $message = 'Sikeres kategória hozzá-
        adás';
6          } else {
7              $error = 'Hiba a kategória hozzáadá-
        sa közben!';
8          }
9          } else {
10             $error = 'Töltsön ki minden mezőt!';
11         }
12     }
```

Ha nem töltöttünk ki minden mezőt vagy hiba történt az új kategória felvitele közben, akkor hibajelzést kapunk, egyébként visszatérünk a kategóriák listájához, ahol már látható is az újonnan felvitt kategória.

Kategóriák módosítása

Egy kategória módosítása is ugyanebben a fájlban fog végrehajtni. Itt annyit fogunk tenni, hogy a kiválasztott kategória azonosítóját megkeressük, majd módosítjuk a hozzá tartozó nevet, és visszatérünk a listához, ahol már látható lesz a változás.



29. ábra: Kategória módosítása

A kategóriát módosító kódrészlet itt látható.

```

1  if (isset($_POST["categoryMod"]) &&
    ($_POST["categoryMod"] != '') &&
    isset($_POST["id"]) && ($_POST["id"] != '') &&
    isset($_SESSION["userId"]) &&
    ($_SESSION["userId"] != '')) {
2      if (isset($_POST["categoryName"]) &&
        ($_POST["categoryName"] != '')) {
3          $query = mysql_query("UPDATE
            ".$GLOBALS["categoryTable"]." SET category_name
            = '".$_POST["categoryName"]." WHERE category_id
            = '".$_POST["id"]."");
4          if ($query) {
5              $message = 'Sikeres kategória módo-
                sítás';
6          } else {
7              $error = 'Hiba a kategória módosítá-
                sa közben!';

```

```

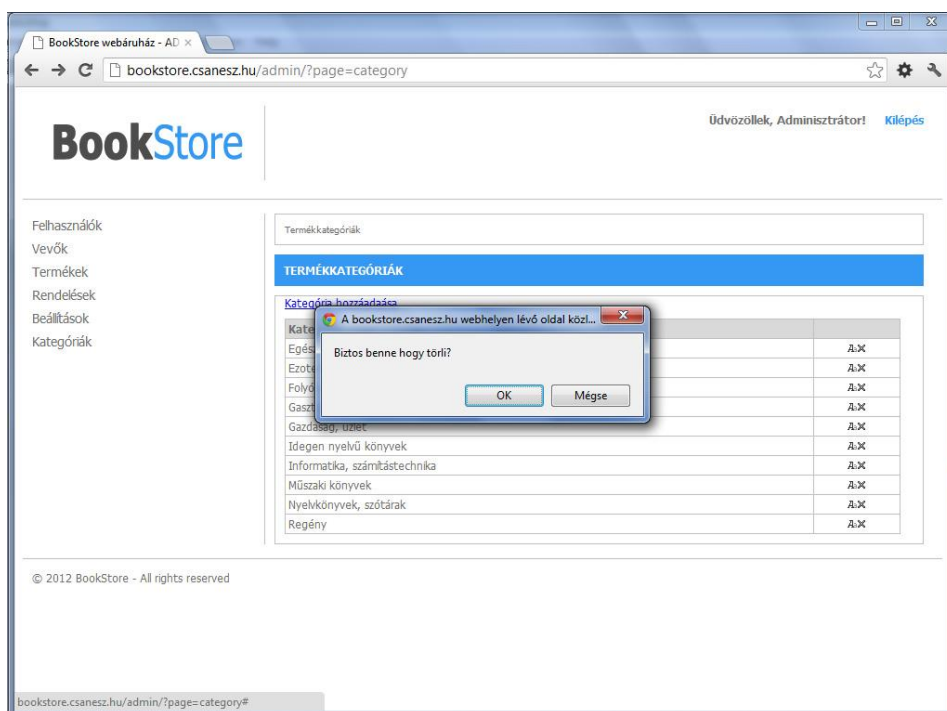
8      }
9      } else {
10         $error = 'Töltsön ki minden mezőt!';
11     }
12 }

```

Látható a 1. és 2. sorban hogy csak akkor fog végrehajtódni a módosítás, ha a kategória azonosítója létezik, a kategória neve nem üres sztring, és be is van lépve valaki.

Kategóriák törlése

A törlésnél vigyáznunk kell, mivel nem logikai törlésről van szó, hanem fizikairól, tehát a törölt adatok nem lesznek visszaállíthatóak.



30. ábra: Kategória törlése

A kategória törlésének kódja itt található.

```

1  if (isset($_POST["func"]) && ($_POST["func"] ==
    'categoryDel') && isset($_POST["id"]) &&
    ($_POST["id"] != '')) &&

```

```
isset($_SESSION["userId"]) &&
($_SESSION["userId"] != '')) {
2     $query = mysql_query("DELETE FROM
    ".$GLOBALS["categoryTable"]." WHERE category_id
    = ".$_POST["id"]);
3     if ($query) {
4         $message = 'Sikeres kategória törlés';
5     } else {
6         $error = 'Hiba a kategória törlése köz-
    ben!';
7     }
8 }
```

Itt ugyanúgy megfigyelhető, hogy kategóriát csak belépett személy tud törölni, és csak akkor, ha létezik a kategória.

7.2.3 Műveletek a termékekkel

Ebben a webáruházban a termékeket lehet majd különböző szempontok alapján rendezni, és kosárba tenni. Ezeket a műveleteket a következő leckékben fogjuk alaposabban átnézni. Nagyobb és komolyabb webáruházakban a termékeknel lehetőségünk van termék összehasonlításra, amivel több különböző termék tulajdonságait láthatjuk egy oldalon. Így akár 3-4 terméket kiválasztva egy lapon tudjuk összehasonlítani ez egyes jellemzőket.

Biztos mindenki hallott már a kedvencekről. Ilyen lehetőséggel is lehet bővíteni az áruházat. Ez a kosár funkcióhoz hasonló, annyi különbséggel, hogy itt nem lehet leadni megrendelést, csak össze lehet gyűjteni azokat a könyveket, amelyek valamiért megtetszettek nekünk.

Készíthetünk akár olyan funkciót is, hogy ajánlani tudjuk e-mailben ismerősünknek valamelyik könyvet.

Látható hogy sokféle funkcióval bővíthető még az áruház. Ez a jegyzet elsősorban az alapvető funkciók bemutatására készült, a továbbiak kidolgozását az Olvasóra bízuk.

7.3 ÖSSZEFOGLALÁS, KÉRDÉSEK

7.3.1 Összefoglalás

A leckében megismerhettük, hogy melyek azok az adatok, amelyek egy könyvnél elengedhetetlenül fontosak a vevő számára. A kategóriák kialakításával pedig a vevők jobban átlátható listát kaptak a megrendelhető könyvekről. Gondoljunk csak bele, mennyivel szebb ez a megoldás, mintha csak úgy ömlesztve kilistáztuk volna az összes könyvet.

7.3.2 Önellenőrző kérdések

- Egy internetes áruházban kettő vagy több termék között milyen műveleteket lehet elvégezni?
- Milyen adatszerkezettel lehet megvalósítani a termékkategóriákat?
- Melyik anomáliára kell odafigyelnünk a termékkategóriák módosításánál?

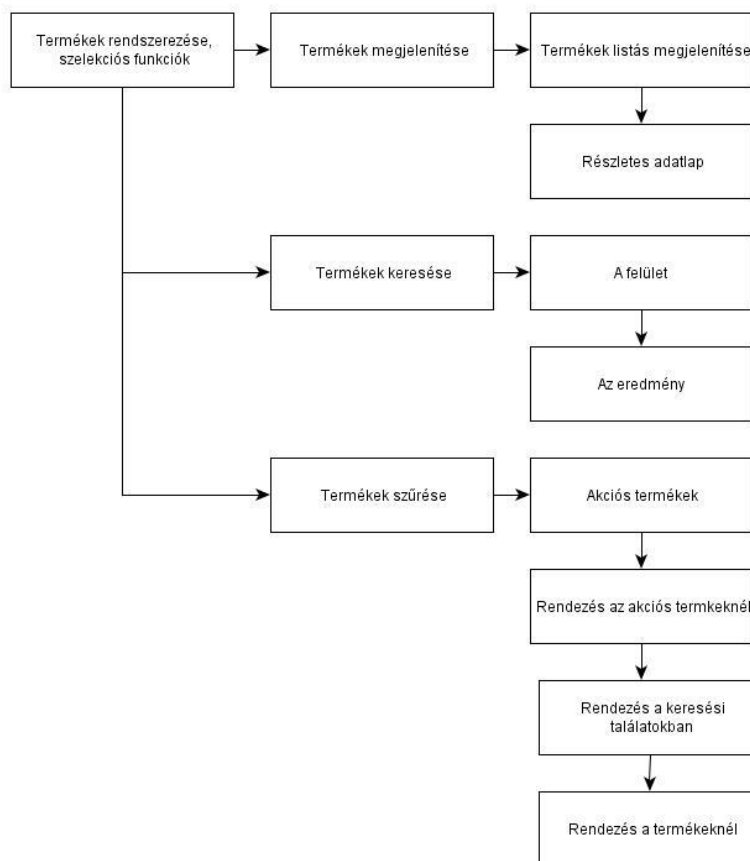
 (Feladatok/Egyetlen állomány feltöltése) Feladat: Töltse fel az eddig elkészült programkódot! (Termékek és kategóriák)

8. TERMÉKEK RENDSZEREZÉSE, SZELEKCIÓS FUNKCIÓK

8.1 CÉLKITÚZÉSEK ÉS KOMPETENCIÁK

Azon kívül, hogy az előző leckében megismerhettük a termékek listázását, nem árt, hogyha valamilyen szempont szerint rendszerezve is vannak ezek az adatok. A most következő leckében a termékeket tudjuk majd rendszerezni különböző adatmezők alapján, és keresni is tudunk majd a termékek között.

8.2 TANANYAG



31. ábra: Lecke áttekintése

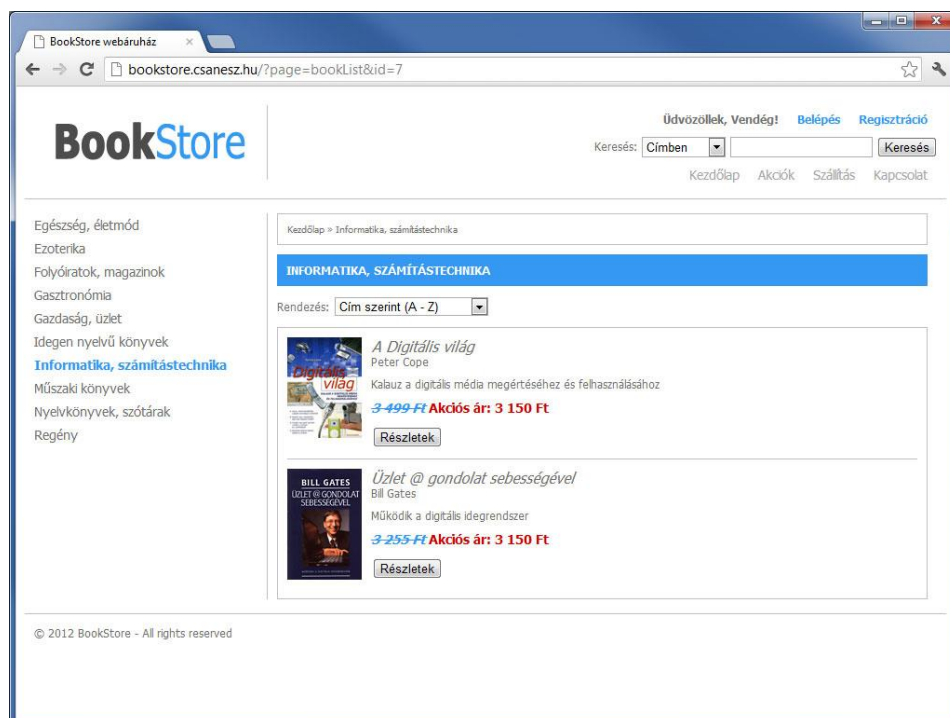
8.2.1 Termékek megjelenítése

Termékek listás megjelenítése

Kezdetben, amikor kiválasztunk egy kategóriát, akkor listás megjelenésben jelennek meg a könyvek. Itt csak a könyv fontosabb paraméterei jelennek meg. A *Részletek* gomb mellett – ha be vagyunk jelentkezve – megjelenik a *Kosárba* gomb is. A listás megjelenés a `bookListPage.php` fájlban található. A kódja itt látható:

```
1      <div id="contentColumn">
2          <div id="contentMenu">
3              <a href=".">Kezdőlap</a> &raquo;
4              <?=getCategoryName($_GET["id"]);?>
5          </div>
6          <div id="contentTitle">
7              <?=getCategoryName($_GET["id"]);?>
8          </div>
9          <div id="contentBox">
10             <?
11             $order = 'book_title|ASC';
12             generateBookList($_GET["id"], $order);
13             <?>
14         </div>
```

A 3. sorban látható függvényhívásnál a függvény a jelenleg kiválasztott kategória nevét adja vissza. Ezt a `funciton.php` fájlban fogjuk megírni.



32. ábra: A termékek listás megjelenítése

A kód itt látható:

```

1 function getCategoryName($id) {
2     $query = mysql_query("SELECT category_name
3     FROM ".$GLOBALS["categoryTable"]." WHERE
4     category_id = ".$id);
5     if (mysql_num_rows($query) > 0) {
6         $result = mysql_fetch_object($query);
7         return $result->category_name;
8     } else
9         return 'Nincs ilyen kategória';
10 }

```

A 2. sorban található lekérdezés kérdezi le a kategóriák közül azt az egyet, amelynek az azonosítója megegyezik a függvény paraméterében megkapott azonosítóval. Ha nincs ilyen kategória, akkor ezzel a szöveggel tér vissza.

A termékek listázása is egy függvényen keresztül történik, amely paraméterben kapja meg a kiválasztott termékkategóriát és a rendezést. Ez a függvény is –mint a többi – a `function.php` fájlban található. A kódja itt olvasható:

```

1  function generateBookList($id,$sort) {
2      $order = str_replace('|',' ', $sort);
3      $query = mysql_query("SELECT * FROM
4      ".$GLOBALS["bookTable"]." WHERE book_category =
5      ".$id." ORDER BY ".$order);
6      if (mysql_num_rows($query) > 0) {
7          print('<ul>');
8          while ($result =
9      mysql_fetch_object($query)) {
10             print('<li>');
11             print('<form action=""
12             method="post">');
13             print('<p>');
14             print('<a
15             href="?page=bookDetails&id=' .
16             $result->book_id.'">book_title.'" border="0" align="left"
20             /></a>');
21             print('<span class="bookTitle"><a
22             href="?page=bookDetails&id=' .
23             $result->book_id.'">'.
24             $result->book_title.'</a></span>');
25             print('<span class="bookTitle2">'.
26             $result->book_author.'</span>');
27             print($result->book_subtitle);
28             print('<span class="bookPrice">');
29             if ($result->book_special) {
30                 print('<del>'.number_format(
31                 $result->book_special * (1 + ($result->book_tax
32                 / 100)),0,' ',' ').' Ft</del> ');
33                 print('<span
34                 class="special">Akciós ár:
35                 '.number_format($result->book_price * (1 + (
36                 $result->book_tax / 100)),0,' ',' ').'
37                 Ft</span>');
38             } else {
39                 print(number_format($result->
40                 book_price * (1 + ($result->book_tax /
41                 100)),0,' ',' ').' Ft');
42             }
43             print('</span>');

```

```

22         print('<button
onclick="window.location.href=\'?page=bookDetail
s
&amp;id=.'.$result->book_id.'\'"; return
false;">Részletek</button>');
23         print('</p>');
24         print('</form>');
25         print('<div class="clear"></div>');
26         print('</li>');
27     }
28     print('</ul>');
29 } else {
30     if (getCategoryName($id) != '')
31         print('<p>Ebben a kategóriában nin-
csenek könyvek!</p>');
32     else
33         print('<p>Nincs ilyen kategó-
ria!</p>');
34 }
35 }

```

Az előző függvény (getCategoryName) ebben a függvényben is szerepet kap (30. sor), mivel figyelmeztetjük a felhasználót, ha olyan termékkategóriát választ ki, amelyben nem szerepelnek könyvek, vagy a kategória nem is létezik.

A 22. sorban található az a hivatkozás, amely a *Részletek* gombra kattintva hajtódik végre.

Részletes adatlap

A termékek részletes adatlapja a bookDetailsPage.php fájlban található. Ennek a fájlnak a tartalma itt látható:

```

1     <div id="contentColumn">
2         <div id="contentMenu">
3             <a href=".">Kezdőlap</a> &raquo; <a
href="?page=bookList&amp;id=
<?=getCategoryIdFromBook($_GET["id"]);?>">
<?=getCategoryName(getCategoryIdFromBook($_GET["
id"]);?></a> &raquo;
<?=getBookName($_GET["id"]);?>
4         </div>
5     <div id="contentBox">

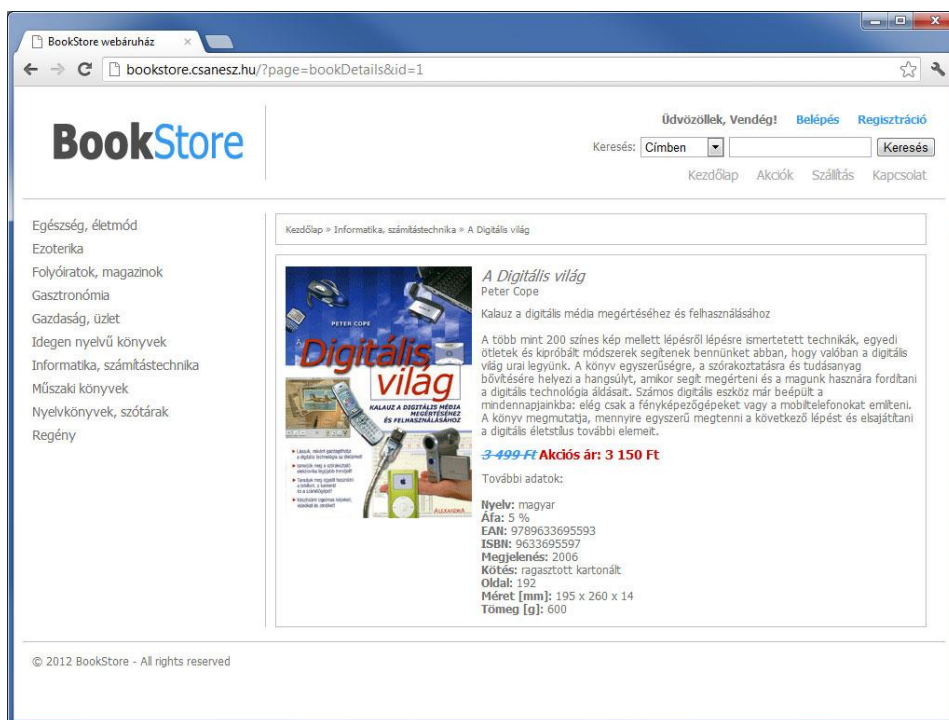
```

```

6         <?
7         generateBookDetails($_GET["id"]);
8         ?>
9     </div>
10 </div>

```

A 3. sorban található egy bonyolultabb hivatkozás, ez valójában egy *Kezdőlap » Kategória neve » Könyv címe* felépítésű menüt ad vissza. A Kezdőlap egyértelmű, a kategória nevét úgy fogjuk megkapni, hogy lekérdezzük az aktuálisan kiválasztott könyv termékkategóriáját, majd a visszakapott kategória azonosítónak lekérdezzük a nevét. Ehhez egy új függvényt kell írunk, ami visszaadja a könyv kategóriáját. Ez a függvény is a `function.php` fájlban fog helyet kapni.



33. ábra: Termékek részletes adatlapja

Lássuk is a programkódot:

```

1 function getCategoryIdFromBook($id) {
2     $query = mysql_query("SELECT category_id
    FROM ".$GLOBALS["categoryTable"]." INNER JOIN

```

```
    ".$GLOBALS["bookTable"]." ON book_category =  
    category_id WHERE book_id = ".$id);  
3     if (mysql_num_rows($query) > 0) {  
4         $result = mysql_fetch_object($query);  
5         return $result->category_id;  
6     } else  
7         return 0;  
8 }
```

A függvény visszatérési értéke jó esetben a kategória azonosítója, rosszabb esetben, 0 értékkel tér vissza. Mivel 0 azonosítójú kategória nem létezik, így ez akár hibafigyelésre is alkalmas.

A könyv nevét is egy függvény fogja nekünk visszaadni, mely szintén ebben a fájlban fog szerepelni. Ehhez bővítsük a fájlt a következő programkóddal:

```
1  function getBookName($id) {  
2      $query = mysql_query("SELECT book_title  
    FROM ".$GLOBALS["bookTable"]." WHERE book_id =  
    ".$id);  
3      if (mysql_num_rows($query) > 0) {  
4          $result = mysql_fetch_object($query);  
5          return $result->book_title;  
6      } else  
7          return "Ismeretlen könyv";  
8  }
```

Mint látatjuk, a legelső programkód 7. sorában a részletes adatlapot is egy függvény fogja legenerálni. Ezt a függvényt is írjuk meg.

```
1  function generateBookDetails($id) {  
2      $query = mysql_query("SELECT * FROM  
    ".$GLOBALS["bookTable"]." WHERE book_id =  
    ".$id);  
3      if (mysql_num_rows($query) > 0) {  
4          $result = mysql_fetch_object($query);  
5          print('<form action="" method="post">');  
6          print('<p>');  
7          print('book_title.'" border="0" align="left"  
    />');  
8          print('<div style="padding-left:  
    210px;">');  
9          print('<span class="bookTitle">'.  
    $result->book_title.'</span>');
```

```

10         print('<span class="bookTitle2">'.
    $result->book_author.'</span>');
11         print($result->book_subtitle.'<br /><br
    />');
12         print($result->book_desc);
13         print('<span class="bookPrice">');
14         if ($result->book_special) {
15             print('<del>'.number_format($result-
    >book_special * (1 + ($result->book_tax /
    100)),0,'',' ').' Ft</del> ');
16             print('<span class="special">Akciós
    ár: '.number_format($result->book_price * (1 + (
    $result->book_tax / 100)),0,'',' ').'
    Ft</span>');
17         } else {
18             print(number_format($result-
    >book_price * (1 + ($result->book_tax /
    100)),0,'',' ').' Ft');
19         }
20         print('</span>');
21         print('További adatok:<br /><br />');
22         print('<strong>Nyelv: </strong>'.
    $result->book_lang.'<br />');
23         print('<strong>Áfa: </strong>'.
    $result->book_tax.' %<br />');
24         print('<strong>EAN: </strong>'.
    $result->book_ean.'<br />');
25         print('<strong>ISBN: </strong>'.
    $result->book_isbn.'<br />');
26         print('<strong>Megjelenés: </strong>'.
    $result->book_year.'<br />');
27         print('<strong>Kötés: </strong>'.
    $result->book_binding.'<br />');
28         print('<strong>Oldal: </strong>'.
    $result->book_page.'<br />');
29         print('<strong>Méret [mm]: </strong>'.
    $result->book_size.'<br />');
30         print('<strong>Tömeg [g]: </strong>'.
    $result->book_weight);
31         print('</div>');
32         print('</p>');
33         print('</form>');
34         print('<div class="clear"></div>');
35     } else {

```

```
36         print('<p>Nincs ilyen kategória!</p>');
37     }
38 }
```

A függvény egy paramétert vár, amelyben megkapja a könyv azonosítóját, majd lekérdezi a hozzá tartozó adatokat. Itt láthatóak a könyv adatai. Az ár formázottan van megjelenítve, azaz szerepel a Ft utótag, és az ezres elválasztó szóköz karakter. Ezt a `number_format` beépített függvénnyel lehet megvalósítani. Ennek rendre a következő paraméterei vannak: szám, tizedes jegyek száma, tizedeselválasztó, ezreselválasztó. A táblában az árak nettó értékben vannak letárolva, így még a kiírás előtt fel kell szorozni azokat az áfatartalommal.

8.2.2 Termékek keresése

A felület

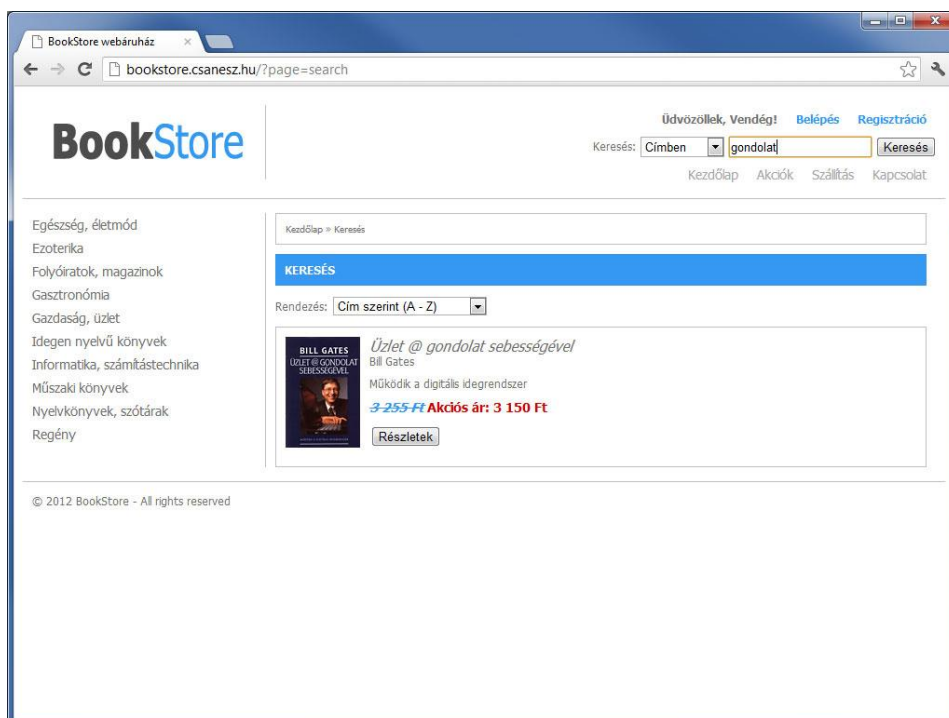
A keresés felületét már nem kell kialakítanunk, mivel a fejléc létrehozásakor ezt már megtettük. A keresési felület kinézete megegyezik a könyvek listás megjelenésével. Ennek kódja a `pages/searchPage.php` fájlban található.

```
1     <div id="contentColumn">
2         <div id="contentMenu">
3             <a href=".">Kezdőlap</a> &raquo; Keresés
4         </div>
5         <div id="contentTitle">
6             Keresés
7         </div>
8         <div id="contentBox">
9     <?
10     $order = 'book_title|ASC';
11     generateSearchBookList($_POST["field"],
12     $_POST["search"], $order);
13 >
14     </div>
```

Az eredmény

A termékek listázása itt is egy függvény kimenete lesz, mégpedig a `generateSearchBookList` függvényé. Ez a függvény három paramétert vár, ebből az első a keresési mező, a második a keresett szöveg részlet, a har-

madik pedig a rendezés. Ez a függvény a gyökérkönyvtárban lévő `function.php`-ben kap helyet.



34. ábra: A keresés eredménye

Nézzük is meg a kódot:

```

1  function
2  generateSearchBookList($field,$search,$sort) {
3      $order = str_replace('|',' ', $sort);
4      switch ($field) {
5          case 'title':
6          case 'subtitle':
7          case 'author': $sql = "book_". $field. "
LIKE '%" . $search. "%'"; break;
8          case 'all': $sql = "((book_title LIKE
 '%" . $search. "%) OR (book_subtitle LIKE
 '%" . $search. "%) OR (book_author LIKE
 '%" . $search. "%))"; break;
9          default: $sql = "book_title LIKE
 '%" . $search. "%'"; break;
10     }

```

```

10     $query = mysql_query("SELECT * FROM
    ".$GLOBALS["bookTable"]." WHERE ".$sql." ORDER
    BY ".$order);
11     if (mysql_num_rows($query) > 0) {
12         print('<ul>');
13         while ($result =
    mysql_fetch_object($query)) {
14             print('<li>');
15             print('<form action=""
    method="post">');
16                 print('<p>');
17                 print('<a
    href="?page=bookDetails&id='.$result->
    book_id.'">book_title.'" border="0" align="left"
    /></a>');
18                 print('<span class="bookTitle"><a
    href="?page=bookDetails&id='.$result->
    book_id.'">'.$result->
    book_title.'</a></span>');
19                 print('<span class="bookTitle2">'.
    $result->book_author.'</span>');
20                 print($result->book_subtitle);
21                 print('<span class="bookPrice">');
22                 if ($result->book_special) {
23                     print('<del>'.number_format(
    $result->book_special * (1 + ($result->book_tax
    / 100)),0,', ' ').' Ft</del> ');
24                     print('<span
    class="special">Akciós ár:
    '.number_format($result->book_price * (1 + (
    $result->book_tax / 100)),0,', ' ').'
    Ft</span>');
25                 } else {
26                     print(number_format($result->
    book_price * (1 + ($result->book_tax /
    100)),0,', ' ').' Ft');
27                 }
28                 print('</span>');
29                 print('<button
    onclick="window.location.href=\'?page=
    bookDetails&id='.$result->book_id.'\'';
    return false;">Részletek</button>');

```

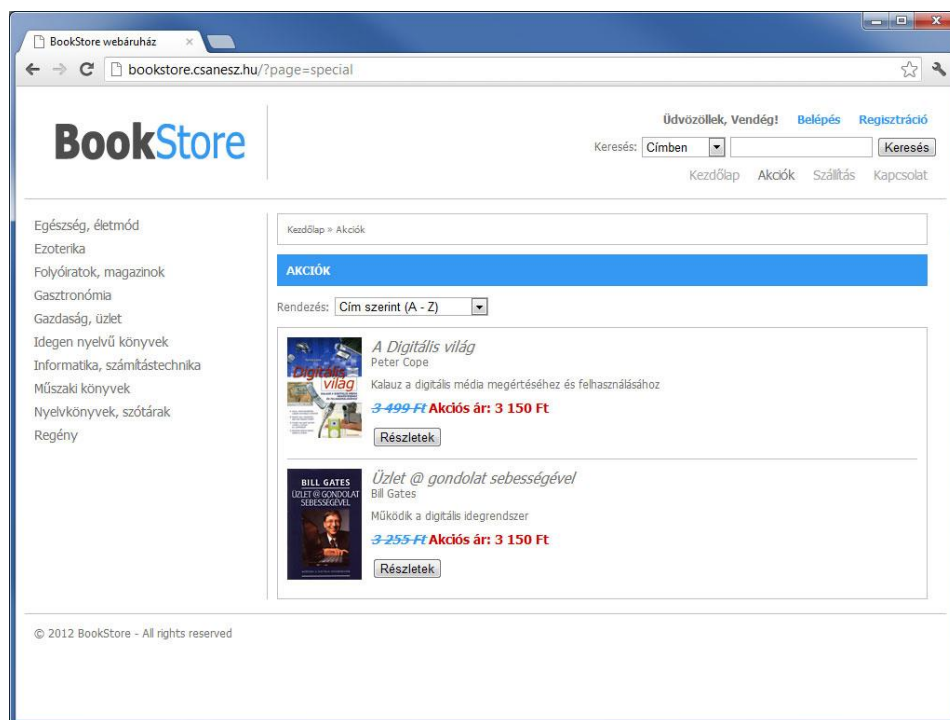
```
30         print('</p>');
31         print('</form>');
32         print('<div class="clear"></div>');
33         print('</li>');
34     }
35     print('</ul>');
36 } else {
37     print('<p>Nincsenek a keresési kritéri-
    umnak megfelelő termékek!</p>');
38 }
39 }
```

Ez a függvény valójában ugyanúgy listázza a termékeket, mintha egy kategóriát választanánk ki, annyi különbséggel, hogy itt nem a kategóriák szerint listáz, hanem a keresett kifejezés alapján. Látható a 2. sorban az `str_replace` függvény, amivel kicseréljük a `|` jelet egy szóközre, így a rendezési feltételt csak be kell szűrni a lekérdezésbe, ami a 10. sorban található. A kódban itt is észre lehet venni, az akciós ár furfangos megjelenítését a 22. sornál. Természetesen a részletes adatlapot is meg tudjuk tekinteni és a kosárba helyezést itt is meg tudjuk oldani.

8.2.3 Termékek szűrése

Akciós termékek

A termékek szűrésével egy bizonyos tulajdonság alapján tudjuk megjeleníteni a könyvek egy részét. Ebben a könyvtárházban ezt a funkciót nem vittük túlzásba. A felső menüben található egy menüpont, melynek neve **Akciók**. Itt csak azok a termékek jelennek meg, amelyek akciós termékek, tehát ezt egyfajta szűrésnek is tekinthetjük.



35. ábra: Akciós termékek

Ez a felület a pages könyvtárban lévő `specialPage.php` fájlban található.

```

1      <div id="contentColumn">
2          <div id="contentMenu">
3              <a href=".">Kezdőlap</a> &raquo; Akciók
4          </div>
5          <div id="contentTitle">
6              Akciók
7          </div>
8          <form action="" method="post">
9              Rendezés:
10             <select name="sorting"
onchange="submit()">
11                 <option
value="book_title|ASC">?=(($_POST["sorting"]=="bo
ok_title|ASC"))?' selected="selected"':'>>Cím
szerint (A - Z)</option>
12                 <option
value="book_title|DESC">?=(($_POST["sorting"]=="b

```

```

ook_title|DESC")?'
selected="selected"':"';?>>Cím szerint (Z -
A)</option>
13         <option
value="book_author|ASC"<?=( $_POST["sorting"]=="b
ook_author|ASC")?'
selected="selected"':"';?>>Szerző szerint (A -
Z)</option>
14         <option
value="book_author|DESC"<?=( $_POST["sorting"]=="
book_author|DESC")?'
selected="selected"':"';?>>Szerző szerint (Z -
A)</option>
15         <option
value="book_price|ASC"<?=( $_POST["sorting"]=="bo
ok_price|ASC")?' selected="selected"':"';?>>Ár
szerint (olcsóbb elől)</option>
16         <option
value="book_price|DESC"<?=( $_POST["sorting"]=="b
ook_price|DESC")?' selected="selected"':"';?>>Ár
szerint (drágább elől)</option>
17         </select>
18     </form>
19     <div id="contentBox">
20 <?
21     $order =
        (isset($_POST["sorting"]))?$_POST["sorting"]:'bo
        ok_title|ASC';
22     generateSpecialBookList($order);
23 ?>
24     </div>
25 </div>

```

A fenti kódból látható, hogy az akciós termékeket is van lehetőségünk különböző feltételek alapján rendezni, mint ahogy azt az előző fejezetben leírtuk. Az is észrevehető, hogy itt is egy külön függvénnyel valósítjuk meg az akciós termékek listázását (22. sor). A rendezés alapesetben a könyv címe szerint, növekvő sorrendben történik (21. sor). A listázásért felelős függvény a `function.php` fájlban található `generateSpecialBookList` néven, melynek kódja itt látható:

```

1     function generateSpecialBookList($sort) {
2         $order = str_replace('|', ' ', $sort);

```

```

3      $query = mysql_query("SELECT * FROM
  ".$GLOBALS["bookTable"]." WHERE book_special <>
  '' ORDER BY ".$order);
4      if (mysql_num_rows($query) > 0) {
5          print('<ul>');
6          while ($result =
mysql_fetch_object($query)) {
7              print('<li>');
8              print('<form action=""
method="post">');
9              print('<p>');
10             print('<a
href="?page=bookDetails&id=' .
$result->book_id.'">book_title.'" border="0" align="left"
/></a>');
11             print('<span class="bookTitle"><a
href="?page=bookDetails&id=' .
$result->book_id.'">' .
$result->book_title.'"</a></span>');
12             print('<span class="bookTitle2">' .
$result->book_author.'"</span>');
13             print($result->book_subtitle);
14             print('<span class="bookPrice">');
15             if ($result->book_special) {
16                 print('<del>' . number_format(
$result->book_special * (1 + ($result->book_tax
/ 100)), 0, ',', ' ') . ' Ft</del> ');
17                 print('<span
class="special">Akciós ár:
' . number_format($result->book_price * (1 + (
$result->book_tax / 100)), 0, ',', ' ') . '
Ft</span>');
18             } else {
19                 print(number_format($result-
>book_price * (1 + ($result->book_tax /
100)), 0, ',', ' ') . ' Ft');
20             }
21             print('<button
onclick="window.location.href=\'?page=bookDetail
s

```

```

&amp;id='.$result->book_id.'\'; return
false;">Részletek</button>');
22         print('</p>');
23         print('</form>');
24         print('<div class="clear"></div>');
25         print('</li>');
26     }
27     print('</ul>');
28 } else {
29     print('<p>Nincsenek akciós termé-
kek!</p>');
30 }
31 }

```

A 2. sorban látható, hogy a kapott paraméterrel vezérelhető a termékek rendezése. A | jel határolja el a mezőnevet a rendezés típusától. Alapértelmezésben a `book_title` mező alapján, azaz a könyv címe szerint rendezi növekvőbe a termékeket. Rendezés sorrendje lehet növekvő (**ASC**) vagy csökkenő (**DESC**). Ezt az `ORDER BY` mezőnév után kell kiadnunk. Alapértelmezésben növekvő a rendezés, így ebben az esetben nem kötelező kiírni.

Rendezés az akciós termékeknél

Ha már szó esett a rendezésről, akkor valósítsuk is meg a termékek oldalán. Első körben az akciós termékeknél alkalmazzuk ezt a lehetőséget. Ehhez módosítsuk a `specialPage.php` fájlt.

Keressük meg a következő sort a kódban:

```
1 <div id="contentBox">
```

Majd illesszük be elé a következő programkódot:

```

1         <form action="" method="post">
2             Rendezés:
3             <select name="sorting"
onchange="submit()">
4                 <option
value="book_title|ASC"<?=( $_POST["sorting"]=="bo
ok_title|ASC")?' selected="selected"':''>>Cím
szerint (A - Z)</option>
5                 <option
value="book_title|DESC"<?=( $_POST["sorting"]=="b
ook_title|DESC")?'
selected="selected"':''>>Cím szerint (Z -
A)</option>

```

```

6         <option
    value="book_author|ASC"<?=( $_POST["sorting"]=="b
    ook_author|ASC")?'
    selected="selected"':'';?>>Szerző szerint (A -
    Z)</option>
7         <option
    value="book_author|DESC"<?=( $_POST["sorting"]=="
    book_author|DESC")?'
    selected="selected"':'';?>>Szerző szerint (Z -
    A)</option>
8         <option
    value="book_price|ASC"<?=( $_POST["sorting"]=="bo
    ok_price|ASC")?' selected="selected"':'';?>>Ár
    szerint (olcsóbb elől)</option>
9         <option
    value="book_price|DESC"<?=( $_POST["sorting"]=="b
    ook_price|DESC")?' selected="selected"':'';?>>Ár
    szerint (drágább elől)</option>
10        </select>
11    </form>

```

Ha megfigyeljük a kódot, akkor további mezőket is felvehetünk a rendezési listába.

Rendezés a keresési találatokban

Ha már elhelyeztük ezt a lehetőséget az akciós termékeknel akkor tegyük be a keresési találatokhoz is.

Módosítsuk a `searchPage.php` fájl tartalmát, az előzőekhez hasonlóan. Keressük meg a következő sort a kódban:

```
1 <div id="contentBox">
```

Majd illesszük be elé az előző programkódot, amelyben a lista található a rendezésekről.

Rendezés a termékeknel

A termékek listáját se felejtjük ki. Itt is szükségünk van rendezni a kategórián belüli könyveket. Ehhez nincs is más dolgunk, mint módosítani a `bookListPage.php` fájlt, úgy, mint ahogy az előző két alkalommal is tettük. Keressük meg a következő sort a kódban:

```
1 <div id="contentBox">
```

Majd illesszük be elé az előző programkódot, amelyben a lista található a rendezésekről.

8.3 ÖSSZEFOGLALÁS, KÉRDÉSEK

8.3.1 Összefoglalás

Ha jól megfigyeltük, a rendezést egyszer kellett csak megírni, és ugyanúgy felhasználható volt több oldalon is. A kereséssel pedig megkönnyítettük a vevők dolgát, mivel nem kell a kategóriákat végignézniük, hanem elég a könyv valamelyik adatára rákeresni, és máris egy szűkebb terméklistát jelenítünk meg, ami könnyebben áttekinthető.

8.3.2 Önellenőrző kérdések

- Egy internetes könyváruháznál mely adatokat jelenítjük meg mindenképpen?
- Keresés megvalósításánál melyik mezőkre célszerű a keresést kiterjeszteni?
- A termékek rendezésénél mely adatokat részesítjük előnyben?

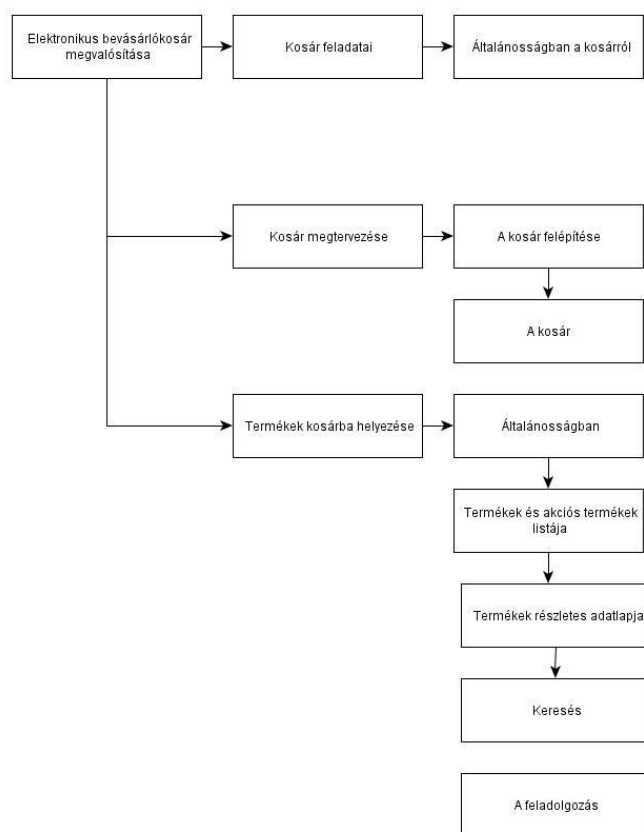
🚩 (Feladatok/Egyetlen állomány feltöltése) Feladat: Töltse fel az eddig elkészült programkódot! (Szelekciós funkciók)

9. ELEKTRONIKUS BEVÁSÁRLÓ-KOSÁR MEGVALÓSÍTÁSA

9.1 CÉLKITŰZÉSEK ÉS KOMPETENCIÁK

Most, hogy már vannak termékeink, tudjuk őket rendezni, keresni is tudunk közöttük, ezért jó lenne, ha valahol össze is tudnánk gyűjteni őket. Erről szól a következő lecke, ahol is egy kosarat fogunk megvalósítani, hasonlóan, mint egy bevásárlóközpontban. A kosárba be tudjuk tenni a könyveket, majd a pénztárnál tudunk fizetni.

9.2 TANANYAG



36. ábra: Lecke áttekintése

9.2.1 Kosár feladatai

Mint minden áruházban, itt is a kosárba gyűjthetjük össze a megvásárolni kívánt könyveket. Természetesen a kosárhoz hozzáadni és kivenni is tudunk könyveket.

Általánosságban a kosárról

A kosár ebben a webáruházban egy adatbázis táblájában fog helyet kapni, mégpedig a `cart` táblában. Lehetne sütivel vagy munkamenettel is megoldani, de így, hogy táblában tároljuk az egyes könyveket, kilépés és belépés után is megmaradnak a könyveink a kosárban. Ezáltal, ha csak szeretnénk összegyűjteni a könyveket, de nem szeretnénk megrendelni, akkor ez tökéletes megoldás erre a célra. Mint tudjuk a sütik és munkamenet-változók az oldal elhagyásakor és a böngésző bezárásakor törlődnek.

Általában a kosárnál a mennyiséget is lehet változtatni. Ez jól jön, ha például nem annyit szeretnénk rendelni az adott termékből, hanem többet vagy kevesebbet. Ekkor nem kell kitörölnünk, és újra felvenni a kosárba a terméket, hanem tudjuk változtatni a mennyiséget az egyes termékeknél.

9.2.2 Kosár megtervezése

A kosár felépítése

A kosár táblájában nyilvántartjuk a következő adatokat:

- Egyedi azonosító
- Vevőazonosító
- Könyvazonosító
- Darabszám

Ezzel a felépítéssel mindig le tudjuk kérdezni, hogy melyik vevő melyik könyvekből hány darabot tett a kosárba. Ha megfigyeljük, akkor egy kosár van létrehozva az összes vevő számára. Ezért is kell, hogy a kosárban azonosítva legyen a vevő. A könyv adatait, pedig egy összetett lekérdezéssel le tudjuk kérdezni.

A kosár

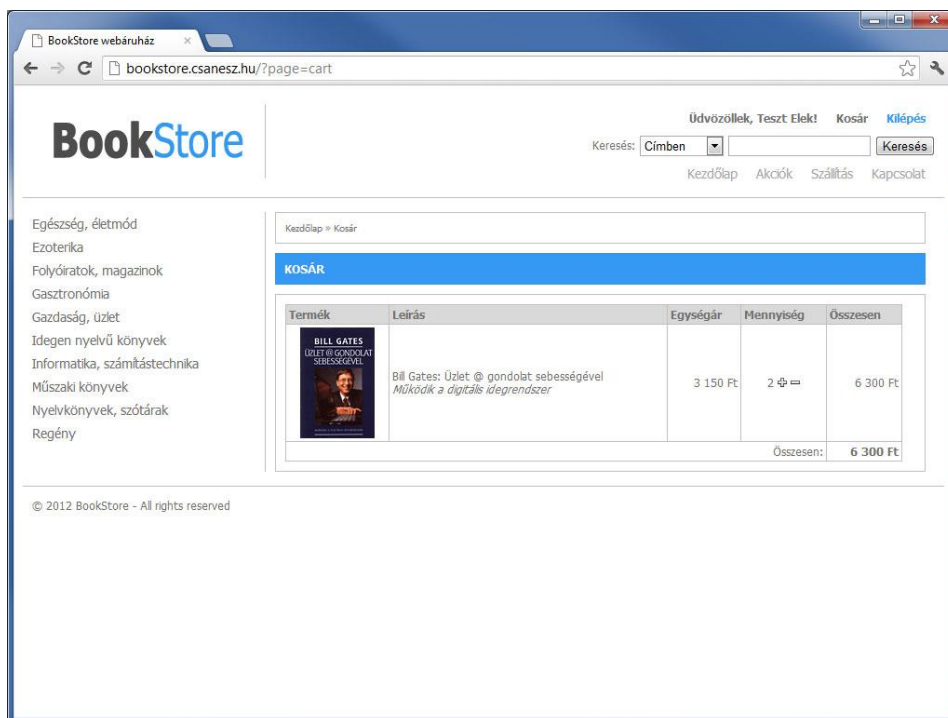
A kosár csak akkor érhető el, ha be van jelentkezve valaki. A felső menüben is csak akkor jelenik meg, ha már bejelentkezett a vevő. Hozzuk létre a `pages` könyvtárban a `cartPage.php` fájlt a következő tartalommal.

```

1      <div id="contentColumn">
2          <div id="contentMenu">
3              <a href=".">Kezdőlap</a> &raquo; Kosár
4          </div>
5          <div id="contentTitle">
6              Kosár
7          </div>
8          <div id="contentBox">
9      <?
10     generateCart();
11     ?>
12         </div>
13     </div>

```

Ezzel a kosár oldalát létre is hoztuk, de még nem működőképes, mivel egy függvény fogja nekünk lekérdezni a kosárban lévő termékeket. Ehhez a `function.php` fájlba kell létrehoznunk egy függvényt, aminek a neve a fenti kód 10. sorában látható.



37. ábra: A kosár

Hozzuk is létre ezt a kódot:

```

1  function generateCart() {
2      if (isset($_SESSION["customerId"]) &&
    ($_SESSION["customerId"] != '')) {
3          $query = mysql_query("SELECT book_id,
    book_author, book_title, book_subtitle,
    book_tax, book_price, book_special, cart_qty
    FROM ".$GLOBALS["cartTable"]." INNER JOIN
    ".$GLOBALS["bookTable"]." ON book_id = cart_book
    WHERE cart_customer =
    '".$_SESSION["customerId"]');
4          if (mysql_num_rows($query) > 0) {
5              print('<table border="0"
    cellpadding="3" cellspacing="0" width="98%"
    id="cartTable">');
6                  print('<tr>');
7                  print('<th>Termék</th>');
8                  print('<th>Leírás</th>');
9                  print('<th>Egységár</th>');
10                 print('<th>Mennyiség</th>');
11                 print('<th>Összesen</th>');
12                 print('</tr>');
13                 while ($result =
    mysql_fetch_object($query)) {
14                     print('<tr>');
15                     print('<td align="center">book_title.'" border="0" /></td>');
16                     print('<td>'.$result->book_author.': '.$result->book_title.'<br
    /><em>'.$result->book_subtitle.'</em></td>');
17                     print('<td
    align="right">'.number_format($result->book_price *
    (1 + ($result->book_tax / 100)),0,', ' ').'
    Ft</td>');
18                     print('<td align="center">');
19                     print($result->cart_qty.' ');
20                     print('<form action="'
    method="post" name="qtyInc_'.$result->book_id.'"
    style="display: inline;">');
21                     print('<a href="#"
    onclick="document.forms[\'qtyInc_'.$result->

```

```

>book_id.'\'].submit(); return false;"></a> ');
22         print('<input type="hidden"
name="qty" value="1" />');
23         print('<input type="hidden"
name="bookId" value="'. $result->book_id.'" />');
24         print('</form>');
25         print('<form action=""
method="post" name="qtyDec_'. $result->book_id.'"
style="display: inline;">');
26         print('<a href="#"
onclick="document.forms[\'qtyDec_'. $result-
>book_id.'\'].submit(); return false;"></a>');
27         print('<input type="hidden"
name="qty" value="-1" />');
28         print('<input type="hidden"
name="bookId" value="'. $result->book_id.'" />');
29         print('</form>');
30         print('</td>');
31         print('<td
align="right">'.number_format($result-
>book_price * (1 + ($result->book_tax / 100)) *
$result->cart_qty,0,',
' ').' Ft</td>');
32         print('</tr>');
33         $total += $result->book_price *
(1 + ($result->book_tax / 100)) * $result-
>cart_qty;
34     }
35     print('<tr>');
36     print('<td colspan="4"
align="right">Összesen: </td>');
37     print('<td
align="right"><strong>'.number_format($total,0,
',
' ').' Ft</strong></td>');
38     print('</tr>');
39     print('</table>');
40     print('<div align="center">');

```

```
41         print('<button  
    onclick="window.location.href=\'?page=order\';  
    return false;">Tovább a pénztárhoz</button> ');  
42         print('</div>');  
43 } else {  
44         print('<p>A kosár üres!</p>');  
45     }  
46 } else {  
47     print('<p>Önnek nincs jogszultsága a  
    lap megtekintéséhez!</p>');  
48 }  
49 }
```

A 2. sor felelős azért, hogy csak az tudja megjeleníteni a kosár tartalmát, aki belépett. A 3. sorban található lekérdezés kérdezi le a kosárból a belépett felhasználó könyveinek az adatait. 7–11. sorban található a táblázat fejléce. Majd egy ciklus fut addig, amíg vannak termékek a kosárban. Rendre megjele-
nítjük a könyv:

- fotóját
- szerzőjét, címét, alcímét
- egységárát formázva, áfával növelve,
- mennyiségét,
- mennyiséggel növelt bruttó egységárát

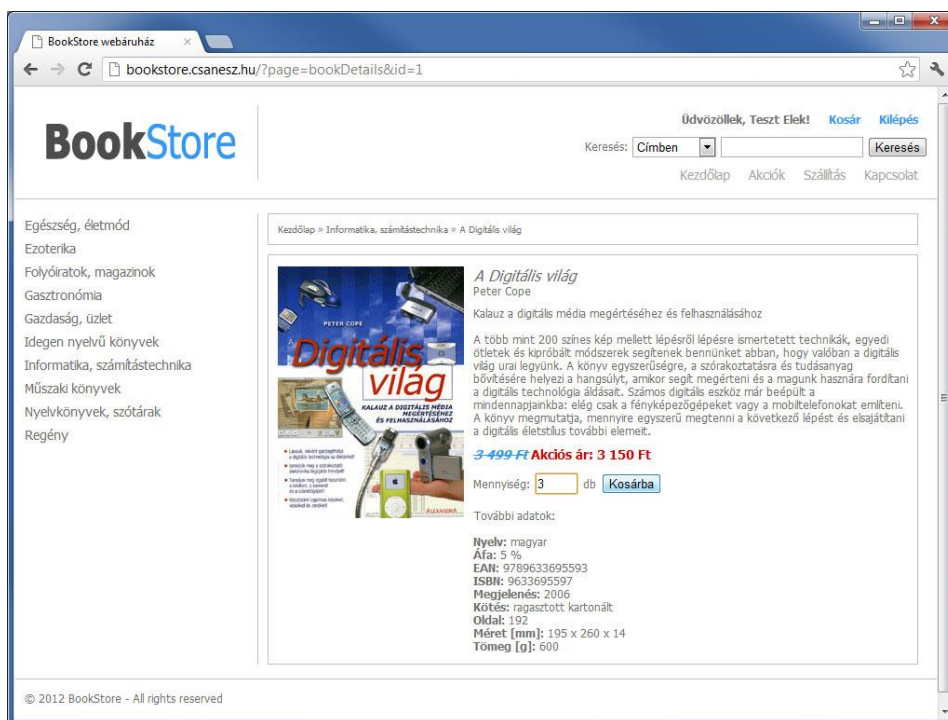
A termékek felsorolásának legvégén látható a végösszeg. (37. sor).

Megfigyelhető a mennyiségnél a két kis ikon, amelyekkel tudjuk a mennyi-
séget növelni és csökkenteni. Ennek a feldolgozása a következő fejezetben lesz
leírva.

9.2.3 Termék kosárba helyezése

Általánosságban

Ahhoz hogy egy terméket a kosárba tudjunk helyezni, regisztrált vevőknek
kell lennünk. Ha nem regisztrálunk, akkor nem tudjuk majd nyomon követni a
rendelést.



38. ábra: Egy termék kosárba helyezése

Miután regisztráltunk az oldalra, be tudunk lépni az e-mail címünkkel, és a jelszavunkkal. A termékek listázásánál, részletes adatlapnál és az akciók oldalon megjelenik a *Kosárba* gomb. Egy darabszám módosítására a kosárban vagy a termékek részletes adatlapján van lehetőségünk. A részletes adatlapon meg tudjuk adni, hogy hány darab könyv kerüljön be a kosárba. A kosár megtekintésekor két nyomógomb segítségével tudjuk a mennyiséget szabályozni. Amennyiben a mennyiséget 0 darabra csökkentjük, az megfelel a termék kosárból való törlésének. A listás megjelenésnél a kosárba tétel 1 darab terméket helyez el a kosárban.

Amennyiben már van ilyen termék, akkor a mennyiségét növeli 1-gyel.

Termékek és akciós termékek listája

A termékeket már az előző leckében kilistáztuk, most bővítsük ki a függvényünket úgy, hogy ha be van jelentkezve valaki, akkor jelenjen meg a *Kosárba* gomb.

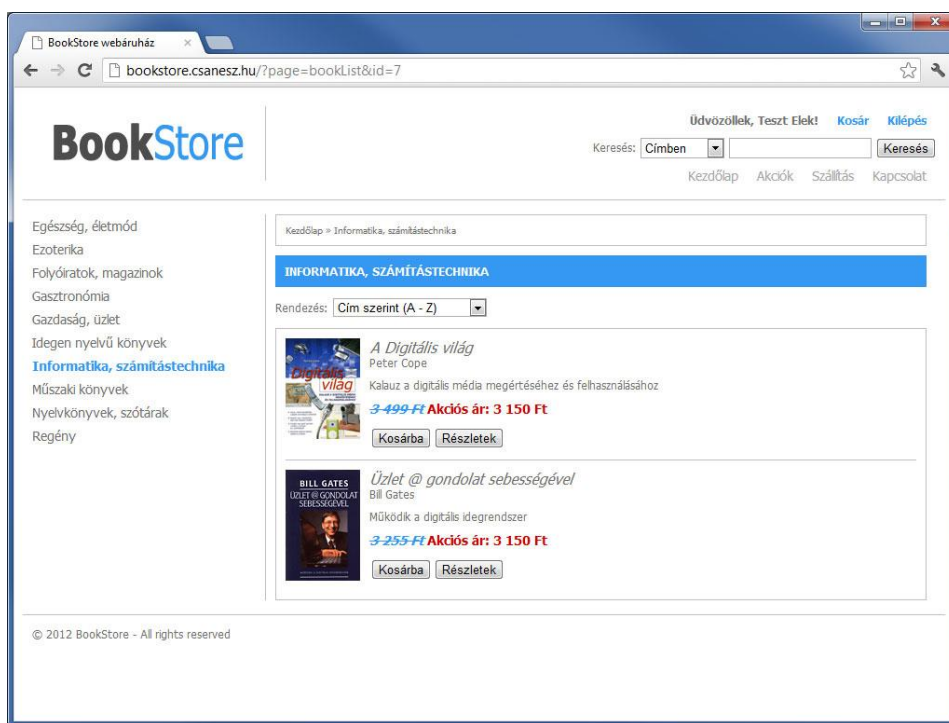
Keressük meg a `function.php` fájlban a `generateBookList` függvényt, a következő sort:

```
1 print('<button onclick="window.location.href=
  \'?page=bookDetails&id='.$result-
  >book_id.'\''; return
  false;">Részletek</button>');
```

Majd illesztjük be a következő kódot az előző kódrészlet elé.

```
1 if (isset($_SESSION["customerId"]) &&
  ($_SESSION["customerId"] != '')) {
2   print('<input type="hidden" name="bookId"
  value="'.$result->book_id.'" />');
3   print('<input type="submit" value="Kosárba"
  />');
4 }
```

Ez a kód megjeleníti a *Kosárba* gombot, és egy rejtett mezőben eltárolja az adott könyv azonosítóját (bookId).



39. ábra: A kosárba gomb

Az akciós termékek oldalán ugyanezt a műveletet kövessük el. Keressük meg a `function.php` fájlban a `generateSpecialBookList` függvényben, a következő sort:

```
1 print('<button onclick="window.location.href=
  \'?page=bookDetails&id='.$result->book_id.'\''; return
  false;">Részletek</button>');
```

Majd illesztjük be a következő kódot, az előző kódrészlet elé.

```
1 if (isset($_SESSION["customerId"]) &&
  ($_SESSION["customerId"] != '')) {
2   print('<input type="hidden" name="bookId"
  value="'.$result->book_id.'" />');
3   print('<input type="submit" value="Kosárba"
  />');
4 }
```

Termékek részletes adatlapja

A részletes adatlapon annyiban különbözik a gomb elhelyezése, hogy itt megadható egy mennyiség is, hogy hány darabot szeretnénk a kosárba helyezni. Mivel az akciós listában és a könyvek listájában is ugyanarra a részletes adatlapra hivatkozunk, így elég egy helyen módosítanunk a függvényt.

Keressük meg a `function.php` fájl `generateBookDetails` függvényben a következő sort.

```
1 print('További adatok:<br /><br />');
```

Majd ez elé illesztjük be a következő kódot:

```
1 if (isset($_SESSION["customerId"]) &&
  ($_SESSION["customerId"] != '')) {
2   print('<input type="hidden" name="bookId"
  value="'.$result->book_id.'" />');
3   print('Mennyiség: <input type="text"
  name="qty" value="" maxlength="3" size="2" /> db
  ');
4   print('<input type="submit" value="Kosárba"
  />');
5   print('<br /><br />');
6 }
```

Látható, hogy egy beviteli mezővel bővült az előzőekhez képest a kód. Amikor majd a termékek részletes adatlapján rákattintunk a gombra, akkor már nemcsak a termék azonosítóját, hanem a mennyiségét is el fogja küldeni a rendszer a feldolgozó részhez.

Keresés

Ha rákeresünk egy termékre, akkor is egy listás nézetet kapunk, amelyben szintén kosárba tehetjük az adott terméket. Ehhez keressük meg a `function.php` fájlban a `generateSearchBookList` függvényben a következő sort.

```
1 print('<button onclick="window.location.href=
  \'?page=bookDetails&amp;id='.$result->
  book_id.'\''; return
  false;">Részletek</button>');
```

Majd ez elé illesztjük be a következő kódot:

```
1 if (isset($_SESSION["customerId"]) &&
  ($_SESSION["customerId"] != '')) {
2   print('<input type="hidden" name="bookId"
  value="'.$result->book_id.'" />');
3   print('Mennyiség: <input type="text"
  name="qty" value="" maxlength="3" size="2" /> db
  ');
4   print('<input type="submit" value="Kosárba"
  />');
5   print('<br /><br />');
6 }
```

A feldolgozás

Ha már megemlítettük a feldolgozást, akkor nézzük meg a kódot, amely a kosárba helyezi a termékeket. A feldolgozó rész – mint minden ilyen kód – a `process.php` fájlban fog helyet kapni. Illesszük be a következő kódot ebbe a fájlba.

```
1 if (isset($_POST["bookId"]) &&
  ($_POST["bookId"] != '')) &&
  isset($_SESSION["customerId"]) &&
  ($_SESSION["customerId"] != '')) {
2   $qty = 1;
3   if (isset($_POST["qty"]) && ($_POST["qty"]
  != '')) $qty = $_POST["qty"];
4   $query = mysql_query("SELECT cart_id,
  cart_qty FROM ".$GLOBALS["cartTable"]." WHERE
  cart_customer = ".$_SESSION["customerId"]." AND
  cart_book = ".$_POST["bookId"]);
5   if (mysql_num_rows($query) > 0) {
6     $result = mysql_fetch_object($query);
```

```
7         if (($result->cart_qty == 1) && ($qty ==  
8         -1))  
9             $query = mysql_query("DELETE FROM  
10            ".$GLOBALS["cartTable"]." WHERE cart_book =  
11            ".$_POST["bookId"]." AND cart_customer =  
12            ".$_SESSION["customerId"]);  
13         else  
14             $query = mysql_query("UPDATE  
15            ".$GLOBALS["cartTable"]." SET cart_qty =  
16            cart_qty + ".$qty." WHERE cart_book =  
17            ".$_POST["bookId"]." AND cart_customer =  
18            ".$_SESSION["customerId"]);  
19         } else {  
20             $query = mysql_query("INSERT INTO  
21            ".$GLOBALS["cartTable"]." (  
22            cart_customer, cart_book, cart_qty) VALUES  
23            ('".$_SESSION["customerId"]."','".$_POST["bookId"].  
24            "','".$_qty."')");  
25         }  
26     }  
27 }
```

Az 1. sorban lévő feltétel felel azért, hogy csak akkor hajtsdjon végre ez a programkód, ha elküldték a könyv azonosítóját. A 2. sorban beállítjuk a mennyiséget 1 darabra. Amennyiben érkezett mennyiség is, akkor ezt felülírjuk. Ez látható a 3. sorban. A 4. sorban lévő feltétel azt vizsgálja meg, hogy a kosárban található-e már ettől a vevőtől ez a fajta könyv. Ha igen, akkor a 6–11. sor hajtodik végre. Itt van még egy vizsgálat, ami azt a célt szolgálja, hogyha a kosárban 1 darab termék van, és csökkentjük a mennyiséget, akkor töröljük az adott terméket a kosárból (8. sor), egyébként frissítjük a mennyiséget (10. sor). A 12. sorban felvesszük a könyvet a kosárba, amennyiben még nem található meg benne.

A termék kosárba helyezése után a felhasználó nem kap semmilyen üzenetet a termék kosárba kerüléséről.

9.3 ÖSSZEFOGLALÁS, KÉRDÉSEK

9.3.1 Összefoglalás

Ebben a leckében a kosár funkciót valósítottuk meg, megtanultuk, hogyan lehet leprogramozni, azt, hogy a mennyiséget is tudjuk változtatni. Ez is a vevők dolgát könnyíti meg. Gondoljunk csak bele, milyen nehéz lenne 10 könyvet a kosárba rakni, ha nem lehetne például megadni a mennyiséget a részletes adatlapon, és nem is lehetne változtatni a kosár oldalán.

9.3.2 Önellenőrző kérdések

- Milyen adatoknak kell szerepelnie a kosár oldalon?
- Milyen főbb funkciókat találhatunk meg egy webruház kosár oldalán?
- Mire kell figyelni egy termék kosárba helyezésénél?

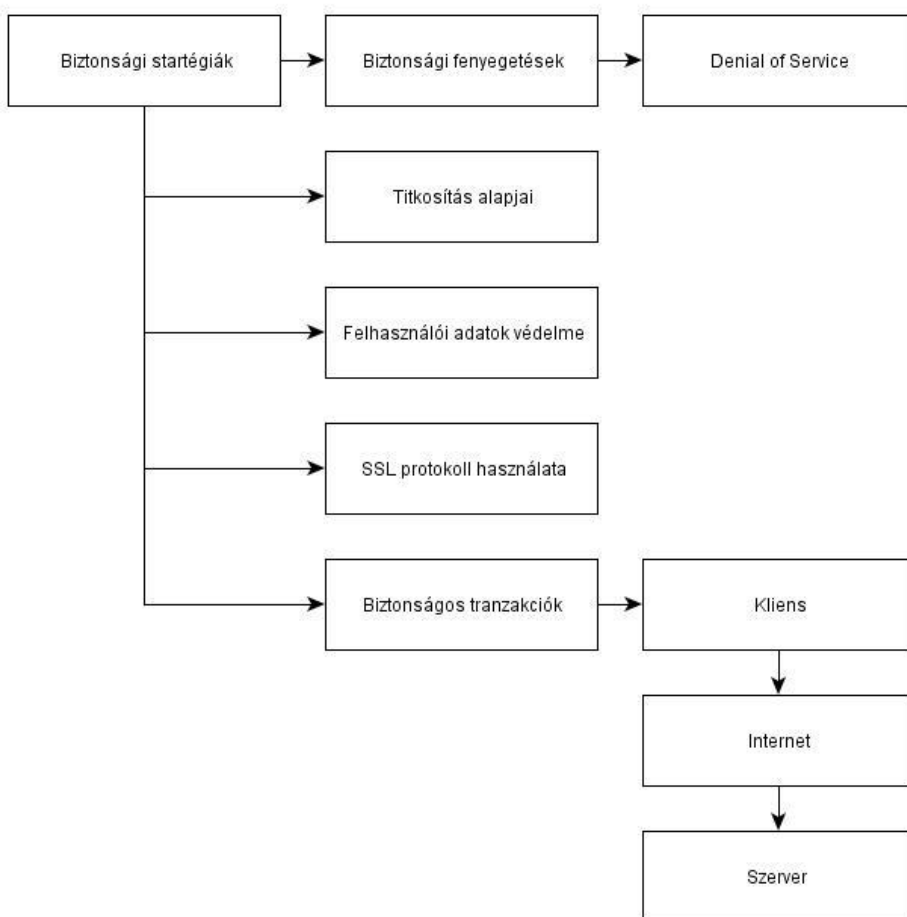
🚩 (Feladatok/Egyetlen állomány feltöltése) Feladat: Töltse fel az eddig elkészült programkódot! (Elektronikus bevásárlókosár)

10. BIZTONSÁGI STRATÉGIÁK

10.1 CÉLKITŰZÉSEK ÉS KOMPETENCIÁK

Ebben a fejezetben a biztonságról lesz szó. Megtudhatjuk, hogyan továbbítsunk biztonságosan adatokat, információkat. Ezzel elérjük azt, hogy a rendszerünk és a felhasználó között teljesen titkosított legyen a kommunikáció.

10.2 TANANYAG



40. ábra: Lecke áttekintése

10.2.1 Biztonsági fenyegetések

Denial of Service

Az egyik legnehezebben kivédhető fenyegetés a szolgáltatásmegtagadással járó támadás, az úgynevezett Denial of Service támadás. Szolgáltatásmegtagadás akkor következik be, ha valaki a cselekedeteivel megnehezíti vagy lehetetlenné teszi a felhasználók számára, hogy hozzáférjenek egy szolgáltatáshoz, vagy késlelteti időkritikus szolgáltatáshoz való hozzáférésüket.

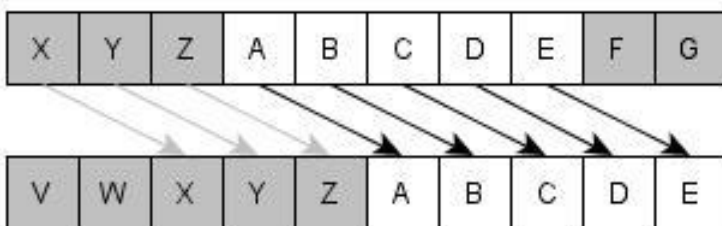
Léteznek olyan oldalak, amelyek jól meghatározható időszakban bonyolítják forgalmuk jelentős részét.

Az ilyen támadások elleni védelem nehézségét egyrészt az okozza, hogy számtalan különböző módon kivitelezhetők. A szóba jöhető módszerek közé tartozik olyan program telepítése a célgépre, amely a rendszer processzoridejének nagy részét lefoglalja.

Automatizált eszközök is léteznek megosztott DoS-támadások indítására. Különösebb tudás nélkül is bárki megteheti, hogy ismert sebezhetőségek után kutatva rengeteg gépet megvizsgál, betör egy nem megfelelően védett gépbe, majd telepíti az eszközt. Automatizált folyamatról lévén szó, a támadó akár gépenként öt másodperc alatt telepítheti az eszközt. Elegendő számú gép beszerzése után mindegyiket utasítja, hogy hálózati forgalommal árásszák el a célt.

10.2.2 Titkosítás alapjai

A titkosítási algoritmus olyan matematikai eljárás, amely az információt látszólag véletlenszerű karaktersorrá alakítja át. A kiinduló adatot egyszerű szövegnek nevezik, bár a folyamat szempontjából nincsen jelentősége annak, hogy mit tartalmaz az információ. A titkosított információt titkosított szövegnek nevezzük, noha legkevésbé szokott szövegnek látszani.



41. ábra: Caesar titkosítás

Az egyszerű és a titkosított szöveg között semmi szemmel látható hasonlóság nincsen. Ez a titkosítási módszer nem visszafordítható. Rengeteg jelszót tárolnak egyirányú titkosítási algoritmussal. A felhasználó által bevitt jelszó ellenőrzéséhez nem szükséges visszafejteni az eltárolt jelszót. Ehelyett a rendszer titkosítja a bejelentkezési kísérletnél megadott jelszót, és összehasonlítja az eredeti titkosított változatával.

Nem az összes, de sok titkosítási folyamat visszafordítható. A fordított folyamatot visszafejtésnek nevezzük.

Számtalan különböző titkosítási algoritmus létezik. Van olyan, például a DES, amelyik titkos kulcsot használ; mások, például az RSA, nyilvános kulcs és külön privát kulcs segítségével működnek.

10.2.3 Felhasználói adatok védelme

Felhasználói adatoknál célszerű a beviteli adatokat szűrni, olyan módon, hogy amikor a beviteli mező tartalmát adatbázisba íránk, előtte egy ellenőrzést hajtunk végre, hogy nincs-e esetleg oda nem illő adat benne. Itt előfordulhatnak SQL-befecskendezések is, amelyek hatására akár az egész adatbázisnak bűcsút mondhatunk.

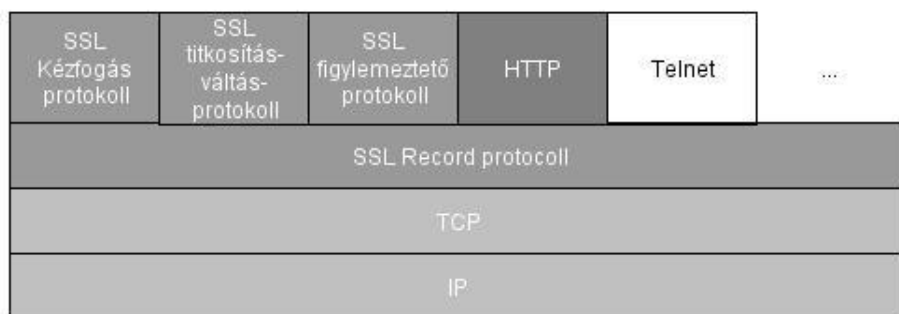
Ezekre a védelmekre a PHP-ban is vannak különböző függvények, melyekkel védeni tudjuk adatainkat az illetéktelen behatolásoktól. Nem szabad elfelejteni, hogy az oldalon szereplő rejtett mezők is átírhatóak. Ha lehetséges, ebben a mezőkben ne tároljuk árakat, illetve ha már tárolunk, a feldolgozás előtt ezt is vizsgáljuk.

A felhasználói adatoknál a jelszavakat sem szabad kifelejtetni. Igaz, hogy erről már egy előző fejezetben is volt szó, de nem elég egyszer megemlíteni.

Kerüljük az olyan felhasználást, amikor a felhasználó által bevitt adat szűrés nélkül átkerül a system vagy exec parancsnak. Ha nem szűrjük meg ezeket az adatokat, akkor akár a szerverünknek is bűcsút mondhatunk.

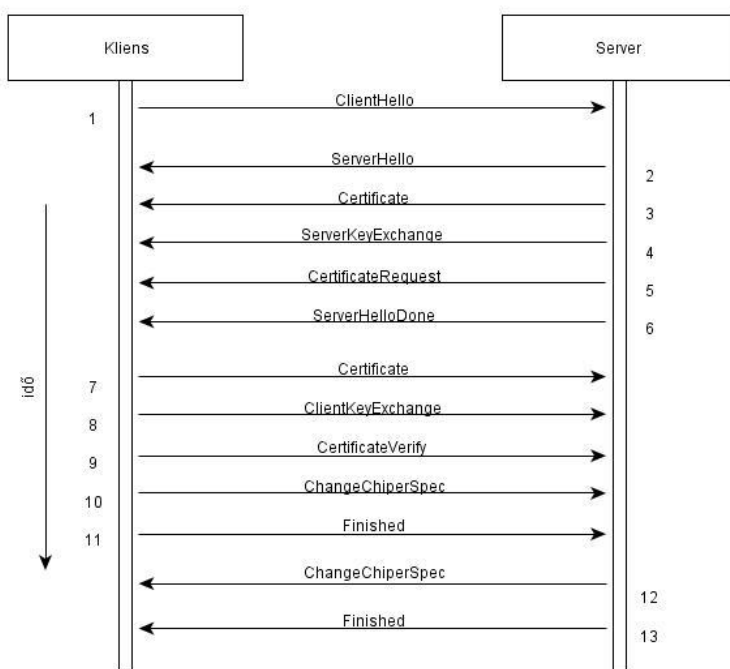
10.2.4 SSL protokoll használata

A Secure Socket Layer, röviddel SSL protokollkészletet a Netscape fejlesztette ki a böngészők közötti kommunikációhoz. Legelterjedtebb verziója az SSL 2-es és 3-as.



42. ábra: Az SSL protokoll elhelyezkedése

A webszerverek támogatják, vagy már alaptól tartalmazzák is a működését.



43. ábra: SSL üzenetcsere

Az ábrán látható az a folyamat, ahogy az SSL üzenetek kicserélődnek a kliens és szerver gép között.

1. A kliens egy **ClientHello** üzenetet küld a szerver számára, néhány plusz kéréssel együtt. Amit a szervertől vár válaszként: a legnagyobb támogatott TLS verziót, egy véletlen számot, egy titkosítási módszert, és egy ajánlott tömörítési eljárást.
2. A szerver egy **ServerHello** üzenettel válaszol a kliensnek, és továbbítja az előbbieken felsorolt kéréseket. A választott protokoll a kliens és a szerver által egyaránt támogatott legújabb verziójú protokoll. Például: Ha a kliens a TLS 1.1-et támogatja, a szerver a TLS 1.2-őt is, akkor az 1.1-es verzió lesz kiválasztva. SSL 3.0 nem választható.
3. A szerver ezután egy **Certificate** üzenetet küld. (A titkosítási módszertől függően ezt a szerver kihagyhatja).
4. A szerver küld egy **ServerKeyExchange** üzenetet, arra az esetre, ha nincs igazolás.
5. Ha a szerver be van konfigurálva a kliens hitelesítéséhez, akkor küld a szerver egy **CertificateRequest** üzenetet.
6. A szerver következik ismét: **ServerHelloDone** üzenet formájában jelzi a kliens felé, hogy a handshake tárgyalási folyamatát lezárja.
7. A kliens elküldi a **Certificate** üzenetet, azaz a tanúsítványt.
8. A kliens **ClientKeyExchange** üzenettel válaszol, ami tartalmazhat egy *PreMasterSecret*-et, vagy egy nyilvános kulcsot, vagy semmit se. Ez a *PreMasterSecret* titkosítva van a szerver nyilvános kulcsa által. A kliens és a szerver a véletlenszámok és a *PreMasterSecret* segítségével előállítja a privát kulcsokat.
9. **CertificateVerify** üzenet nem kerül elküldésre, ha a kliens nem küld visszaigazolást.
10. A kliens egy **ChangeCipherSpec** rekordot küld a szervernek, jelezve felé, hogy „minden amit innentől mondani fogok, titkosítva lesz”.
11. Végül, a kliens küld egy **Finished** üzenetet, amely egy hasht és egy MAC-et tartalmaz az előző üzenetekből.
12. A szerver is küld a kliens részére egy **ChangeCipherSpec** üzenetet.
13. Végül, a szerver is küld egy **Finished** üzenetet.

Az SSL-ben a titkosítás előtt történik a tömörítés, azért, mert a hálózati forgalom nagy része a hálózaton való továbbítás előtt tömöríthető, de a titkosított adatok nem jól tömöríthetők. A tömörítési algoritmusok az adatokon belüli ismétlődéseket próbálnak azonosítani. Általában semmilyen haszonnal nem jár, ha az adatok titkosítása, gyakorlatilag véletlenszerű bitekké alakítása után pró-

báljuk meg tömöríteni azokat. Kellemetlen lenne, ha az SSL mellékhatásaként jelentős mértékben növelné a hálózati forgalmat.

Az SSL viszonylag összetetten működik, a felhasználók és a fejlesztők elől rejtve marad a történetek nagy része, mivel a protokoll külső felületei meglévő protokollokat másolnak.

10.2.5 Biztonságos tranzakciók

Első körben meg kell figyelünk, hogy a rendszerünkben hogyan mozog az információ, és minden lehetséges ponton gondoskodnunk kell a biztonságról. Sajnos az informatikában a mai fejlődő világban nem lehet olyan rendszert létrehozni, amelybe külső támadó valahogy ne tudna bejutni.



44. ábra: Kliens-Internet-Szerver kapcsolat

Ha megvizsgáljuk, hogy miből áll össze egy ilyen rendszer, akkor a következő csoportokat alkothatjuk meg.

Kliens

Ha azt a gépet vizsgáljuk, ahol jelenleg vagyunk, akkor előfordulhat, hogy nem vagyunk teljesen biztonságban, ugyanis ezek a gépek lehetnek kávézóban, iskolában vagy akár a könyvtárban is. Sok felhasználó van, aki kikapcsolja a biztonsági funkciókat, például Java, ActiveX, JavaScript vagy a sütik. Ha készítünk egy rendszert, egyik szempontként utána kell járnunk, hogy ha a felhasználó nem engedélyezi ezeknek a futtatását, akkor miként működik a rendszerünk. Az SSL protokoll automatikusan felépíti a kapcsolatot a kliens böngészője és a szerver számítógépe között. A felhasználók gépei, illetve böngészői között is lehetnek különbségek. Nem is tudjuk módosítani a kliensgépen lévő adatokat. Az adatok ellenőrzésére inkább a szerveroldalon kerüljön sor (PHP), mivel a kliensoldali (JavaScript) ellenőrzés nem biztos, hogy a valóságot tükrözi. Ha adatokat szeretnénk tárolni, erre lehetőség van a kliens gépen is munkamenetben vagy sütiben, illetve szerveroldalon például adatbázisban. Törekedjünk arra, hogy a felhasználó gépén minél kevesebb adatot tároljunk.

Internet

Amikor az interneten elküldünk valamilyen információt, jusson eszünkbe, hogy ez az információ pontról pontra terjed, így az üzenetünket mások láthatják vagy lehallgathatják. Erre az esetre vannak megoldások, például a titkosított adatkapcsolat a szerver és kliens gép között. Ez lehet egy egyszerű titkosítás vagy digitális aláírás is. Két technológia van az adatok titkosítására.

- SSL (Secure Socket Layer)
- HTTPS (Secure Hypertext Transfer Protocol)

Mind a két megoldás titkosított, nem módosítható üzenetküldést kínál számunkra.

Szerver

Webes alkalmazás írásakor figyeljünk oda, hogy a kódunk milyen adatot küldenek a felhasználóknak. Ha a PHP programunk a MySQL adatbázis szerver-től kér le adatokat, akkor ez a TCP/IP protokollon keresztül történik, ami nem titkosított. Ez olyan esetben fordul elő, ha a PHP parancsértelmező és az adatbázis szerver két külön gépen helyezkedik el. Digitális tanúsítvány használatával tudjuk megvédeni magunkat, az illetéktelenektől.

10.3 ÖSSZEFOGLALÁS, KÉRDÉSEK

10.3.1 Összefoglalás

Mint észrevettük, a leckében, nem vagyunk biztonságban. Nagyon sok időt kell fordítani arra, hogy a szervert és a kódunkat biztonságossá tegyük. Ha nagy a forgalma a weboldalnak, és elég népszerű, akkor célszerű erre a célra egy külön embert foglalkoztatni, hogy biztonságban érezzük magunkat.

10.3.2 Önellenőrző kérdések

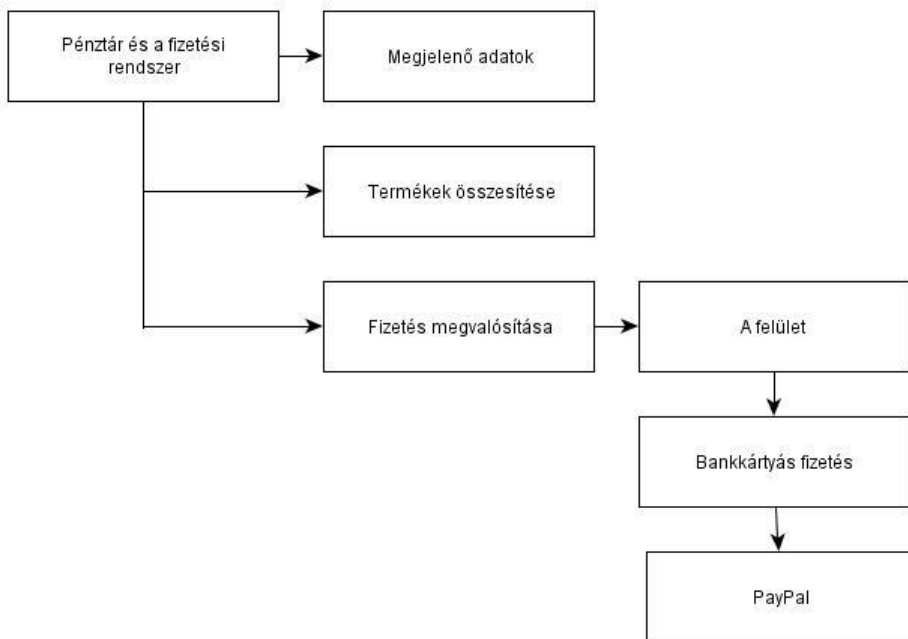
- Milyen fenyegetési formákat ismerünk?
- Miért van szükségünk a felhasználói adatok védelmére?
- Milyen előnyökkel jár az SSL használata?
- Milyen előnyei vannak a tranzakcióknak?

11. PÉNZTÁR ÉS A FIZETÉSI RENDSZER

11.1 CÉLKITŰZÉSEK ÉS KOMPETENCIÁK

Ebben a leckében megnézzük, hogy tudjuk kifizetni a már kiválasztott könyveket. Említést teszünk a különböző fizetési módokról és áttekintjük a különböző állomásokat a fizetés folyamán.

11.2 TANANYAG



45. ábra: Lecke áttekintése

11.2.1 Megjelenő adatok

A kosár oldalon megjelenik a kiválasztott termékek listája és alatta látható lesz egy **Tovább a pénztárhoz** gomb. A gombra kattintáskor megjelenő oldalon a szállítási címet kérjük be a felhasználótól, mivel előfordulhat, hogy nem ugyanoda rendeli a könyvet, ahova azt a regisztrációkor megadta. Itt lehetősége van másik címet megadni.

BookStore webáruház

bookstore.csanesz.hu/?page=order

Üdvözöllek, Teszt Elek! [Kosár](#) [Kilépés](#)

Keresés:

[Kezdőlap](#) [Akciók](#) [Szállítás](#) [Kapcsolat](#)

BookStore

Egészség, életmód
 Ezoterika
 Folyóiratok, magazinok
 Gasztronómia
 Gazdaság, üzlet
 Idegen nyelvű könyvek
 Informatika, számítástechnika
 Műszaki könyvek
 Nyelvkönyvek, szótárak
 Regény

Kezdőlap > Pénztár

PÉNZTÁR

Szállítási adatok

Irányítószám:

Város:

Utca, házszám:

Megjegyzés:

© 2012 BookStore - All rights reserved

46. ábra: Szállítási adatok bekérése

Ez lehet például a munkahelyi címe is. Meg kell jegyeznünk, hogy ez a cím csak a jelenlegi vásárlásnál lesz felhasználva, és a webáruházunk nem fogja eltárolni ezt az információt. Ehhez létre kell hoznunk egy új oldalt a pages könyvtárban `orderPage.php` néven, amelynek a kódját itt látható.

```

1      <div id="contentColumn">
2          <div id="contentMenu">
3              <a href=".">Kezdőlap</a> &raquo; Pénztár
4          </div>
5          <div id="contentTitle">
6              Pénztár
7          </div>
8          <div id="contentBox">
9      <?
10         $query = mysql_query("SELECT * FROM
11             ".$GLOBALS["customerTable"]." WHERE customer_id
12             = ".$SESSION["customerId"]);
11         if (mysql_num_rows($query) > 0) {
12             if (isset($_POST["data"]))
```

```
13         $data = explode('|', $_POST["data"]);
14         $result = mysql_fetch_object($query);
15 ?>
16         <form action="?page=confirm" method="post">
17             <table border="0" cellpadding="3"
18             cellspacing="0" align="center" width="98%">
19                 <tr>
20                     <td align="left" colspan="2">
21                         <strong>Szállítási adatok</strong>
22                     </td>
23                 </tr>
24                 <tr>
25                     <td align="right">
26                         Irányítószám:
27                     </td>
28                     <td align="left">
29                         <input type="text"
30                         name="customerPostcode"
31                         value="<?=(isset($_POST["data"]))?$data[0]:$result->customer_postcode;?>" />
32                     </td>
33                 </tr>
34                 <tr>
35                     <td align="right">
36                         Város:
37                     </td>
38                     <td align="left">
39                         <input type="text"
40                         name="customerCity"
41                         value="<?=(isset($_POST["data"]))?$data[1]:$result->customer_city;?>" />
42                     </td>
43                 </tr>
44                 <tr>
45                     <td align="right">
46                         Utca, házszám:
47                     </td>
48                     <td align="left">
49                         <input type="text"
50                         name="customerAddress"
51                         value="<?=(isset($_POST["data"]))?$data[2]:$result->customer_address;?>" />
52                     </td>
53                 </tr>
```

```

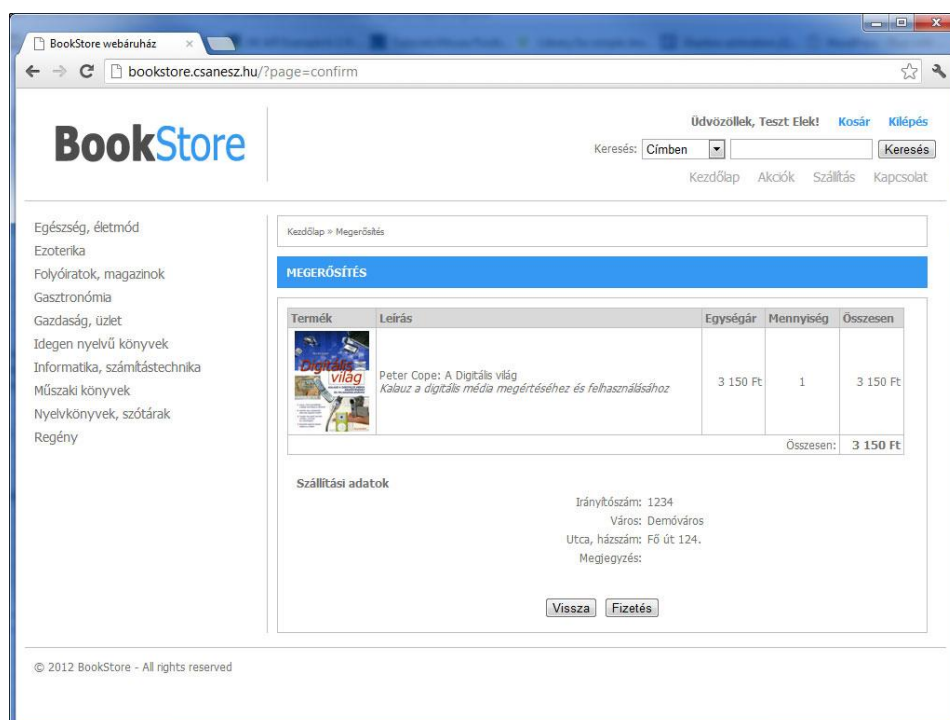
47         <tr>
48             <td align="right" valign="top">
49                 Megjegyzés:
50             </td>
51             <td align="left">
52                 <textarea
53                     name="customerNotes"><?=(isset($_POST["data"]))?
54                     $data[3]:$result->customer_notes;?></textarea>
55             </td>
56             </tr>
57             <tr>
58                 <td align="center" colspan="2">
59                     <button
60                         onclick="submit();">Megrendelés megerősíté-
61                         se</button>
62                     <button
63                         onclick="document.location.href='?page=cart';
64                         return false;">Vissza</button>
65                     </td>
66                 </tr>
67             </table>
68         </form>
69     <?
70     } else {
71         print('Nincs ilyen felhasználó!');
72     }
73 ?>
74 </div>
75 </div>

```

A 12–13. sorban egy posztolt értékkel van dolgunk. Ez a következő oldalról fog érkezni, abban az esetben, ha a felhasználó meggondolja magát. Így a megadott adatokat fogja majd a rendszer a beviteli mezőkbe beírni. Amennyiben nem lett kitöltve, úgy a rendszerben eltárolt értékekkel lesznek feltöltve a mezők. Hasonló felépítésű űrlap szerepelt a vevők regisztrálásánál is.

11.2.2 Termékek összesítése

A termékek összesítése – mondhatni – a fizetés előtti fázis, amikor is az összes adatot egyszerre látjuk, és már csak egy kattintás választja el a felhasználót a fizetéstől.



47. ábra: Összesítő lap

Ehhez is létre kell hoznunk egy új oldalt a pages könyvtárban `confirmPage.php` néven, amelynek a kódja itt látható.

```

1      <div id="contentColumn">
2      <div id="contentMenu">
3          <a href=".">Kezdőlap</a> &raquo; Megerő-
sítés
4      </div>
5      <div id="contentTitle">
6          Megerősítés
7      </div>
8      <div id="contentBox">
9  <?
10     generateConfirmCart();
11  ?>
12     <form action="?page=order" method="post"
id="confirmForm">
13     <table border="0" cellpadding="3"
cellspacing="0" align="center" width="98%">

```

```

14         <tr>
15             <td align="left" colspan="2">
16                 <strong>Szállítási adatok</strong>
17             </td>
18         </tr>
19         <tr>
20             <td align="right">
21                 Irányítószám:
22             </td>
23             <td align="left">
24                 <?=$_POST["customerPostcode"];?>
25             </td>
26         </tr>
27         <tr>
28             <td align="right">
29                 Város:
30             </td>
31             <td align="left">
32                 <?=$_POST["customerCity"];?>
33             </td>
34         </tr>
35         <tr>
36             <td align="right">
37                 Utca, házszám:
38             </td>
39             <td align="left">
40                 <?=$_POST["customerAddress"];?>
41             </td>
42         </tr>
43         <tr>
44             <td align="right" valign="top">
45                 Megjegyzés:
46             </td>
47             <td align="left">
48                 <?=$_POST["customerNotes"];?>
49             </td>
50         </tr>
51         <tr>
52             <td align="center" colspan="2"><br
/><br />
53             <input type="hidden" name="data"
value="<?=$_POST["customerPostcode"]."|".$_POST[
"customerCity"]."|".$_POST["customerAddress"]."|
".$_POST["customerNotes"];?>" />

```

```

54         <button
      onclick="submit();">Vissza</button>
55         <button
      onclick="getElementById('confirmForm').action =
      '?page=pay';
      getElementById('confirmForm').submit();">Fizetés
      </button>
56     </td>
57 </tr>
58 </table>
59 </form>
60 </div>
61 </div>

```

Az 53. sorban található az a mező, amelyik összefűzi az éppen átadott szállítási adatokat arra az esetre, ha a vevő visszalép az előző oldalra. A 10. sorban látható egy függvényhívás, amelynek a programkódja itt olvasható:

```

1  function generateConfirmCart() {
2      if (isset($_SESSION["customerId"]) &&
      ($_SESSION["customerId"] != '')) {
3          $query = mysql_query("SELECT book_id,
      book_author, book_title, book_subtitle,
      book_tax, book_price, book_special, cart_qty
      FROM ".$GLOBALS["cartTable"]." INNER JOIN
      ".$GLOBALS["bookTable"]." ON book_id = cart_book
      WHERE cart_customer =
      '".$_SESSION["customerId"]."");
4          if (mysql_num_rows($query) > 0) {
5              print('<table border="0"
      cellpadding="3" cellspacing="0" width="98%"
      id="cartTable">');
6                  print('<tr>');
7                  print('<th>Termék</th>');
8                  print('<th>Leírás</th>');
9                  print('<th>Egységár</th>');
10                 print('<th>Mennyiség</th>');
11                 print('<th>Összesen</th>');
12                 print('</tr>');
13                 while ($result =
      mysql_fetch_object($query)) {
14                     print('<tr>');
15                     print('<td align="center">book_title.'" border="0" /></td>');
16         print('<td>'.$result-
>book_author.': '.$result->book_title.'<br
/><em>'.$result->book_subtitle.'</em></td>');
17         print('<td
align="right">'.number_format($result-
>book_price * (1 + ($result->book_tax /
100)),0,' ',' ').' Ft</td>');
18         print('<td align="center">');
19         print($result->cart_qty.' ');
20         print('</td>');
21         print('<td
align="right">'.number_format($result-
>book_price * (1 + ($result->book_tax / 100)) *
$result->cart_qty,0,' ',' ').' Ft</td>');
22         print('</tr>');
23         $total += $result->book_price *
(1 + ($result->book_tax / 100)) * $result-
>cart_qty;
24     }
25     print('<tr>');
26     print('<td colspan="4"
align="right">Összesen: </td>');
27     print('<td
align="right"><strong>'.number_format($total,0,'
',' ').' Ft</strong></td>');
28     print('</tr>');
29     print('</table>');
30     } else {
31         print('<p>A kosár üres!</p>');
32     }
33     } else {
34         print('<p>Önnek nincs jogszolgáltatása a
lap megtekintéséhez!</p>');
35     }
36 }

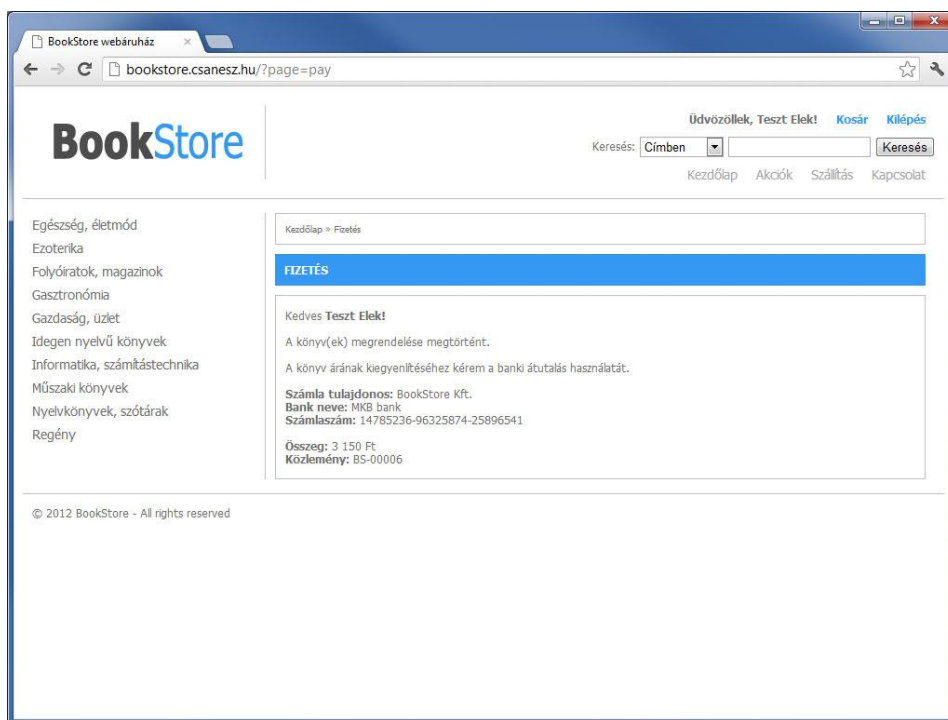
```

A fent látható kód hasonló kimenetet készít, mint a kosárnál látott táblázat, annyi eltéréssel, hogy ebben már nem lehet állítani a mennyiséget az egyes könyveknél. A táblázat alatt látható az előző oldalon megadott szállítási információ, és két gomb. Az egyik gombbal (Vissza) vissza tudunk lépni az előző oldalra, például, ha elértuk a címünket. A másik gombbal (Fizetés) tudunk továbblépni a fizetési oldalra.

11.2.3 Fizetés megvalósítása

A felület

A fizetés megvalósítására többféle megoldás létezik. Mi a legegyszerűbbet választottuk. A mi esetünkben csak elkészítjük a megrendelést, és megadjuk a bankszámlaszámunkat, ahova vevő át tudja utalni a számla végösszegét. Az átutalás beérkezése után pedig feladhatjuk a könyveket.



48. ábra: Fizetés véglegesítése

Ez úgy zajlik le, hogy amikor a vevő kattint a Fizetés gombon, akkor betöltődik egy oldal, amely az aktuális kosár listát rögzíti az adatbázisban, majd törli a kosár tartalmát. Az adatbázisban az `order` táblába egy rekord kerül be, amely tartalmazza a vevő azonosítóját és a vásárlás dátumát. Ezután minden egyes termék bekerül az `orderlist` táblába, amelyben minden adatot letárolunk a könyvről. Erre azért van szükség, mert ha idő közben az egyik könyv már nincs forgalomban, akkor is vissza tudjuk nézni a megrendeléseket.

Ehhez ez oldalhoz is létre kell hozni egy kódot. Ez a `pages` könyvtárban a `payPage.php` néven szerepel.

```

1      <div id="contentColumn">
2          <div id="contentMenu">
3              <a href=".">Kezdőlap</a> &raquo; Fizetés
4          </div>
5          <div id="contentTitle">
6              Fizetés
7          </div>
8          <div id="contentBox">
9      <?
10     if (isset($_SESSION["customerId"]) &&
11         ($_SESSION["customerId"] != '')) {
12         $query = mysql_query("SELECT book_author,
13 book_title, book_subtitle, book_tax, book_price,
14 cart_qty FROM ".$GLOBALS["cartTable"]." INNER
15 JOIN ".$GLOBALS["bookTable"]." ON book_id =
16 cart_book WHERE cart_customer =
17 ".$_SESSION["customerId"]);
18         if (mysql_num_rows($query) > 0) {
19             $ql = mysql_query("INSERT INTO `order`
20 (order_customer,order_date) VALUES
21 (".$_SESSION["customerId"].",".time().")");
22             $lid = mysql_insert_id();
23             while ($result =
24 mysql_fetch_object($query)) {
25                 $q = mysql_query("INSERT INTO orderlist
26 (list_order,list_author,list_title,list_tax,list
27 _qty,list_price) VALUES (". $lid.",'".$_result->
28 book_author."','".$_result->
29 book_title."','".$_result->book_tax."','".$_result->
30 cart_qty."','".$_result->book_price * (1 +
31 ($_result->book_tax / 100)).")");
32                 $total += $_result->book_price * (1 +
33 ($_result->book_tax / 100)) * $_result->cart_qty;
34             }
35             $code = "BS-
36 ".$_str_pad($lid,5,'0',STR_PAD_LEFT);
37             $q = mysql_query("DELETE FROM cart WHERE
38 cart_customer = ".$_SESSION["customerId"]);
39             print('<br /> Kedves
40 <strong>'.getCustomerName().'!</strong><br /><br
41 />');
42 ?>
43 A könyv(ek) megrendelése megtörtént.<br /><br
44 />

```

```
24  A könyv árának kiegyenlítéséhez kérem a banki
    átutalás használatát.<br /><br />
25  <strong>Számla tulajdonos: </strong>BookStore
    Kft.<br />
26  <strong>Bank neve: </strong>MKB bank<br />
27  <strong>Számلاسزám: </strong>14785236-
    96325874-25896541<br /><br />
28  <strong>Összeg: </strong>
    <?=number_format($total,0,'',' ');?> Ft<br />
29  <strong>Közlemény: </strong> <?=$code;?><br
    /><br />
30 <?
31     } else {
32         print('<p>A kosár üres!</p>');
33     }
34 } else {
35     print('<p>Önnek nincs jogosultsága a lap
    megtekintéséhez!</p>');
36 }
37 ?>
38     </div>
39 </div>
```

A 11. sorban lekérdezzük a kosárban lévő könyveket, a 13. sorban pedig felveszi a vevő azonosítóját és a dátumot az `order` táblába. A 15. sorban meggyünk végig a termékeken, és visszük fel az `orderlist` táblába. Közben a `total` változóba számoljuk a teljes összeget.

Bankkártyás fizetés

Ennél a lehetőségnél bekérjük a felhasználótól a személyes adatait, és a bankkártya adatait is. Ezt nem nagyon szeretik használni a vevők, hiszen ki az, aki egy internetes oldalon csak úgy megadja a bankkártyájának adatait. Ehhez a megvalósításhoz azzal a bankkal kell felvennünk a kapcsolatot, akinél a számlaszámunk van létrehozva, mivel ő tud számunkra interfészt biztosítani, amit beépíthetünk az oldalunkba. Hiába, hogy ez a kapcsolat is titkosított protokollon keresztül zajlik, nem szeretik használni.

otpdirekt Internetes fizetés **forrás: qb.hu**

Kérjük, adja meg az alábbi adatokat!

Bolt (szolgáltató) neve	WWW.PATIKASZENTANNA.HU
Fizetendő összeg	1.748
Devizanem	HUF
On-line áruház megjegyzése	Szent Anna Patika - vásárlás

Terhelendő bankkártya adatai

Terhelendő bankkártyaszám

Ha Ön olyan, az OTP Bank által amelynek száma tíz számjegyű, al megadni: 675761. Ezt az Ön bankk összesen 16-jegyű bankkártyaszám

49. ábra: Példa egy bankkártyás fizetés megvalósítására

PayPal

A PayPal-os fizetésnél létre kell hoznunk egy virtuális számlát, amire tölthetünk fel pénzt. Amikor fizetésnél ezt a lehetőséget választjuk, akkor a PayPal oldalán be kell lépünk, és az ott létrehozott számla segítségével tudjuk rendezni az anyagiakat. Ez annyiban jobb, mint a bankkártyás fizetés, hogy itt csak annyi pénzünk van, amit a számlára utalunk, és a bankkártyánkról semmilyen információt nem kell megadni. A rendszerünk átlép a PayPal oldalára, és átküldi a szükséges információkat. Ilyen adatok például az ügyféladatokat, saját adataink és a tranzakció végösszege.

Három Ász Média Kft **forrás: mobilaszh.hu**

Please enter your donation amount and click **Update Total**.

Purpose	Donation amount	Total
Három Ász Média Kft.		
		Total: 0 HUF

Update Total

Donate quickly with PayPal

PayPal **Secure Payments**

PayPal securely processes donations for Három Ász Média Kft. You can complete your payment with just a few clicks.

Why use PayPal?

- It's easy to send money and shop online
- You can donate without sharing your financial information
- Over 50,000 online merchants accept PayPal

Don't have a PayPal account?
Use your credit card or bank account (where available). [Continue](#)

LOG IN TO PAYPAL

Email:

Password:

Log in

50. ábra: PayPal fizetés megvalósítása

11.3 ÖSSZEFOGLALÁS, KÉRDÉSEK

11.3.1 Összefoglalás

A leckében látható volt, hogy egy könyv megrendelésének is külön folyamata van, ugyan úgy, mint egy áruházban. Fontos, hogy minden egyes állomás mindig meglegyen egy vásárlás folyamán. Az előző leckével összehangolva, ajánlott a titkosított adatkapcsolat a fizetés folyamán.

11.3.2 Önellenőrző kérdések

- Milyen módszereket ismerünk a fizetés megvalósításához?
- Mire kell odafigyelni, amikor a fizetés után rögzítjük a megrendelést?
- Milyen adatok szükségesek hogy megjelenjenek a termékekösszesítésénél?

 (Feladatok/Egyetlen állomány feltöltése) Feladat: Töltse fel az elkészített online áruházának programkódját!

12. ÖSSZEFOGLALÁS

12.1 TARTALMI ÖSSZEFOGLALÁS

A könyv elején bepillantottunk az elektronikus kereskedelem világába. Megismerkedtünk ezen dinamikusan fejlődő üzleti tevékenység működésével, előnyeivel, és a velejáró veszélyekkel. A webáruház kialakítása előtt meg kellett terveznünk azt, tisztában kellett lennünk az on-line áruházak szerkezeti felépítésével, feladataival, jellemzőivel.

Ezen ismeretekre alapozva a következő leckéken végighaladva, teljesen a nulláról felépítettünk egy webáruházat a megfelelő adatbázissal, és a hozzá tartozó különböző függvényekkel. Sajnos a könyv véges határai között nem lehetett részletekig menő kialakítást biztosítani. A létrehozás fázisainak lépésről lépésre történő leírását csak több száz oldalon tudnánk megvalósítani. Törekedtünk egy jól működtethető, szerkezetében stabil rendszer létrehozására.

Biztos akadnak az olvasók között olyanok, akik megkérdezik, hogy miért így valósítottunk meg egy-két dolgot. Bizonyára lehetett volna jobb, szebb megoldást is találni az egyes problémákra, de próbáltunk arra törekedni a könyv megírása alatt, hogy a legfontosabb funkciók belekerüljenek a rendszerbe. A termék kategóriáknál is lehetett volna külön fejezetet nyitni arra az esetre, hogy milyen technikákkal lehet megvalósítani egy többszintű kategóriafát. Ehhez már a PHP kódban a rekurziót kell alkalmaznunk, amikor is az egyes kategóriákból egy fa adatszerkezetet építünk fel, és ezt listázzuk ki. Az adatbázisban pedig ez egy újabb mező létrehozását írja elő.

12.2 ZÁRÁS

Számos fejlesztési lehetőség tárul elénk, melyeknek kapuit nyitva hagyjuk az olvasók részére. A könyvünkben megszerzett tudásra támaszkodva ezek megvalósítása nem jelenthet akadályt.

Hasznos funkció az egyes termékeknél az összehasonlítási lehetőség, hogy több termék tulajdonságait is egyszerre megtekintsük. Ha belegondolunk, ehhez nem kell sokat módosítanunk a kódon, mivel egy termék tulajdonságai már így is megjelennek. Csupán annyi a teendőnk, hogy eltávolítjuk az egyes könyveknek az azonosítóját, és nem egy könyv tulajdonságait listázzuk, hanem a már eltávolított könyvekét is.

A fizetési rendszer részleteibe menő kidolgozása kedvezően befolyásolhatja a későbbiekben forgalmunkat. Amire viszont figyelmet kell fordítanunk, hogy

a bankkártyás fizetésekhez a bankkal külön fel kell venni a kapcsolatot, és az Ő kódjukat is be kell építenünk az áruházba. Igaz, ez már komolyabb programozói tudást igényel.

A mai dinamikusan fejlődő világunkban egyre nagyobb szerepet kap az elektronikus üzleti tevékenység. A hagyományos üzletek törekszenek párhuzamosan on-line működést is biztosítani, így fejlesztve vállalkozásukat. A fejlődést tekintve és rohanó világunkra való tekintettel a jövő mérlege az elektronikus kereskedelem felé visz el. Azok a vállalatok maradhatnak csak a piacon, akik fejlődőképeseek, dinamikusan követik annak igényeit és képesen a megújulásra. Könyvünk segítséget nyújt egy jól átgondolt, szerkezetében stabil, működőképessé alap webáruház kialakításában.

13. KIEGÉSZÍTÉSEK

13.1 IRODALOMJEGYZÉK

13.1.1 Hivatkozások

Könyv

luke, Welling – Laura, Thomson: PHP és MySQL webfejlesztőknek: Hogyan építsünk webáruházat. Budapest, Perfact Kiadó, 2010.

Eszes I. – Bányai E.: Online marketing. Budapest, Műszaki Könyvkiadó, 2002.

Elektronikus dokumentumok / források

<http://httpd.apache.org/>

<http://mobilasz.hu/>

<http://www.gb.hu/>

<http://hu.wikipedia.org/>

ESZES I.: E-KERESKEDELEM, ESZES KÖNYVTÁR, 2011.

([HTTP://WWW.ESZES.NET/SAJATPUBLIKACIOK/EKERESKEDELEM_ESZES.PDF](http://WWW.ESZES.NET/SAJATPUBLIKACIOK/EKERESKEDELEM_ESZES.PDF), 2012.JÚNIUS 5.)