SZEGEDI SZAKKÉPZÉSI CENTRUM VASVÁRI PÁL GAZDASÁGI ÉS INFORMATIKAI SZAKGIMNÁZIUMA

Az 54 213 05 számú Szoftverfejlesztő szakképesítés záródolgozata

A TDwebshop és TDmanage alkalmazások bemutatása

Készítette:
TÚRI DÁNIEL

SZEGED

2020

Tartalomjegyzék

Bevezető

Az általam választott záródolgozat témája egy csak gitárokkal foglalkozó webshop készítése, amely a céggel és a webshoppal kapcsolatos hírek megjelenítésére is alkalmas. A webes alkalmazás által használt adatbázis karbantartása egy desktop alkalmazás segítségével lehetséges.

Azért választottam ezt a témát, mert Magyarországon egyetlen egy webshopról van tudomásom, amely csak és kizárólag gitárokkal, azokhoz való erősítőkkel, kiegészítőkkel és alkatrészekkel foglalkozik.

Mivel a hobbim a gitározás, ezért úgy gondolom, van rálátásom arra, hogy milyen termékeket lehet és kell árusítania egy ilyen webshopnak. Továbbá, egy ilyen összetett rendszer (amely a következőkből épül fel: webes alkalmazás, adatbázis, desktop alkalmazás) elkészítése kihívásokkal teli, ezért rengeteg új tudást lehet elsajátítani és némi tapasztalatot is ad. Nem figyelmen kívül hagyható az se, hogy egy ilyen rendszer megvalósítása a szakmában való elhelyezkedéskor jó referencia lehet.

Fejlesztői dokumentáció

1. A téma újbóli, részletesebb kifejtése

A záródolgozatom webes részének témája egy csak és kizárólag gitárokkal, azokhoz való erősítőkkel, kiegészítőkkel és alkatrészekkel foglalkozó webshop elkészítése HTML jelölőnyelv, JavaScript kliensoldali script nyelv és jQuery JavaScript könyvtár, PHP szerveroldali script nyelv, CSS stíluslapok és Bootstrap CSS keretrendszer segítségével.

A weboldal látogatói meg tudják tekinteni a termékeket, de rendelésre csak regisztráció és belépés után van lehetőség.

A záródolgozatom desktop részének témája a fent említett webshop által kezelt adatbázis karbantartása. A desktop alkalmazás C# programozási nyelvben készül el .NET keretrendszer segítségével. Ezt az alkalmazást a cég dolgozói és egy rendszergazda tudja használni, természetesen felhasználónév és jelszó ellenében. Ez fogja meghatározni, hogy dolgozóról, vagy rendszergazdáról van-e szó.

A cég dolgozói meg tudják tekinteni a webshopon leadott rendeléseket, és ez alapján elő tudják készíteni azokat átvételre vagy küldésre. Továbbá képesek új hír írására

(ami a webshop főoldalán jelenik meg) és új termékek felvitelére az adatbázisba, és egy esetleges hiba esetén a felvitt termékeket módosítani tudják. Természetesen hogy elkerüljük az adatbázis integritásának sérülését, bármely termék módosítása esetén a rendelésekben is módosul az adott termék.

A rendszergazdák a desktop alkalmazás segítségével új dolgozói és adminisztrátori hozzáféréseket tudnak kiosztani.

2. Adatbázis

Az asztali alkalmazás és a webshop csak közvetve, az adatbázison keresztül tud egymással kommunikálni.

Az adatbázis mindkét alkalmazásnak biztosít szükséges adatot. Új rekord beszúrását mindkét alkalmazás végrehajthat, (az asztali adatbázis a meglévő termékek frissítését, módosítását is elvégezheti), azonban az adatbázis sémát egyik sem manipulálhatja.

Ebben az esetben egy *relációs adatmodell*[1] elvén működő, MySql adatbázisról beszélhetünk, amely phpMyAdmin szoftver segítségével készült el. Az adatbázisban található összes tábla motorja InnoDB, a karakterkódolás pedig "utf8mb4_hungarian_ci". Ez a karakterkódolás az összes oszlopra is érvényes.

A minél kevesebb redundancia megvalósítása érdekében *idegen kulcs*[2] megkötéseket alkalmaztam.

Az adatbázisomban 3 felhasználó van definiálva, amelyből kettőt jelszó véd.

A *visitor_and_regged_user* a webshophoz létrehozott felhasználó. A webshop csak és kizárólag a SELECT és az INSERT parancsokat adhat az adatbázisnak.

A másik felhasználó az *employee_and_administrator*. Ez az asztali alkalmazáshoz létrehozott felhasználó. Az asztali alkalmazás csak és kizárólag SELECT, INSERT, DELETE és UPDATE utasításokat adhat az adatbázisnak.

A harmadik felhasználó pedig a root. Ez arra az esetre van, ha bármi miatt az adatbázis struktúrájára vonatkozó feladatot kell ellátni. Ez a felhasználó típus teljes

[1] Relációs adatmodell: A halmazelméletre és a matematikai logikára alapozó adatmodell. (Forrás: http://www.info.nytta.hu/temak/abkez/jegyzet_ab2.pdf)
[2] Idegen kulcs: Olyan azonosító, amelynek segítségével egy másik tábla elsődleges kulcsára hivatkozhatunk. (Forrás: http://szh.szltiszk.hu/files/tantargyi/info/info_cd/adatbazis-kezeles_segedlet/alapfogalmak.htm)

hozzáférést élvez az adatbázishoz és bármilyen műveletet elvégezhet rajta. A phpMyAdmin nevű szoftver éri el az adatbázist ezzel a felhasználóval.

2.1. Az adatbázis táblái

- admins**:** A desktop alkalmazás azon felhasználóinak adatait tartalmazza, akik új dolgozót vagy adminisztrátort tudnak hozzáadni. A jelszó SHA256 titkosítás után kerül be az adatbázisba. A struktúra a következő táblázat szerint alakul:

1. táblázat

admins

Oszlopnév	Típus	Null	Alapértelmezett	Extra	Index
id	int(11)	No	None	AUTO_INCREMENT	PRIMARY
username	varchar(64)	No	None	AUTO_INCREMENT	PRIMARY
password	varchar(256)	No	None	-	-

- categories: A termékek kategóriáit tartalmazza. A struktúra a következő táblázat szerint alakul:

2. táblázat

categories

Oszlopnév	Típus	Null	Alapértelmezett	Extra	Index
id	int(11)	No	None	AUTO_INCREMENT	PRIMARY
category_name	varchar(128)	No	None	-	-

- employees: A desktop alkalmazás azon felhasználónak adatait tartalmazza, akik új terméket, új hírt tudnak feltölteni az adatbázisba, látják a rendeléseket és a meglévő

termékeket tudják módosítani. A jelszó SHA256 titkosítás után kerül be az adatbázisba. A struktúra a következő táblázat szerint alakul:

3. táblázat

employees

Oszlop név	Típus	Null	Alapértelmezett	Extra	Index
id	int(11)	No	None	AUTO_INCREMENT	PRIMARY
username	varchar(64)	No	None	-	-
password	varchar(256)	No	None	-	-

- news: A webshop főoldalán, úgy nevezett slideshow-ban megjelenő hírek adatait tartalmazza. A struktúra a következő táblázat szerint alakul:

4. táblázat

news

Oszlop név	Típus	Null	Alapértelmezett	Extra	Index
id	int(11)	No	None	AUTO_INCREMENT	PRIMARY
title	varchar(128)	No	None	-	-
text	varchar(256)	No	None	-	-
image	varchar(512)	No	None	-	-
date	date	No	current_timestamp()	-	-

- orders: A webshopon leadott rendelések adatait tartalmazza. A struktúra a következő táblázat szerint alakul:

5. táblázat

orders

Oszlop név	Típus	Null	Alapértelmezett	Extra	Index
id	int(11)	No	None	AUTO_INCREMENT	PRIMARY
user_id	int(11)	No	None	-	INDEX
address	varchar(128)	No	None	-	-
details	text	No	None	-	-
total_price	int(11)	No	None	-	-
order_date	date	No	current_timestamp()	-	-

- products: A webshopon árusított termékek adatait tartalmazza. A struktúra a következő táblázat szerint alakul:

6. táblázat

products

Oszlop név	Típus	Null	Alapértelmezett	Extra	Index
id	int(11)	No	None	AUTO_INCREMENT	PRIMARY
make	varchar(64)	No	None	-	-
type	varchar(128)	No	None	-	-
color	varchar(64)	No	None	-	-
category	varchar(64)	No	None	-	INDEX
price	int(11)	No	None	-	-
details	text	No	None	-	-
image	varchar(256)	No	None	-	-
lefthanded	tinyint(1)	No	None	-	-

quantity	int(11)	No	None	-	-
active	tinyint(1)	No	1	-	-
upload_date	date	No	current_timestamp()	-	-

- users: A webshop regisztrált felhasználóinak adatait tartalmazza. A webshopon való sikeres regisztráció után a jelszó SHA256 algoritmussal kerül titkosításra (szerver oldalon), majd ezután kerülnek be ide az adatok. A struktúra a következő táblázat szerint alakul:

7. táblázat

users

Oszlop név	Típus	Null	Alapértelmezett	Extra	Index
id	int(11)	No	None	AUTO_INCREMENT	PRIMARY
full_name	varchar(64)	No	None	-	-
email	varchar(64)	No	None	-	-
password	varchar(256)	No	None	-	-

2.2. Kapcsolatok

Ahogyan a *Melléklet 1. ábrája* is mutatja, az adatbázisban csupán 2db kapcsolat létezik.

A *products* tábla *category* oszlopa és a *categories* tábla *category_name* nevű oszlopa között áll fent egy reláció. A 2 tábla között egy-több kapcsolat áll fent, mivel egy kategória tartozhat több termékhez is (több terméknek lehet egyszerre ugyanaz a kategóriája), viszont egy terméknek csak egy kategóriája lehet.

Az *orders* tábla *user_id* oszlopa és a *users* tábla *id* oszlopa között is létezik egy kapcsolat. Ebben az esetben is egy-több kapcsolatról beszélhetünk. Egy felhasználóhoz több megrendelés is tartozhat, de egy rendelés csak egy felhasználóhoz tartozhat.

3. Asztali alkalmazás

Az asztali alkalmazásom, a TDmanage a TDWebshop internetes áruház alapjául szolgáló adatbázist hívatott kezelni.

Az alkalmazás elkészítésekor az MVP (Model View Presenter) tervezési mintát alkalmaztam.

A fejlesztői dokumentáció e részében szeretném felsorolni, hogy milyen fejlesztői környezetet használtam (IDE, nyelv, könyvtárak és keretrendszerek), a számítógép konfigurációját (amelyen az alkalmazás készült) és a funkcionális, valamint minőségi követelményeket, továbbá tisztáznám a szerepköröket. Ezek az információk nagyon fontosak lehetnek azok számára, akiknek egy ilyen fejlesztői dokumentáció elsősorban szól.

Személyes véleményem szerint a legjobb, ha törekszünk az átláthatóságra és a tömörségre. Ennek fényében úgy gondolom, hogy a legcélravezetőbb, ha a fent említetteket felsorolás és vázlat szerűen írjuk le.

Az előzőleg említetteken túl szeretnék említést tenni arról, hogy ezen alkalmazás fő funkciói milyen algoritmusok alapján működnek.

3.1. Fejlesztői környezet leírása

3.1.1. Fejlesztői környezet

- Visual Studio Community 2019 16.4.5

3.1.2. Választott nyelv

- C#

3.1.3. Könyvtárak/keretrendszerek

- .NET Framework 4.8

3.1.4. Futtatási környezet leírása

- OS: Windows 8.1
- CPU: AMD FX8320
- RAM: 8GB
- GPU: GeForce GTX1050
- HDD: 500GB
- Alaplap: AsRock 970 Pro3 R2.0

3.2. Specifikációk és követelmény analízis

3.2.1. Funkcionális követelmények

- Az alkalmazás funkcióit csak bejelentkezés után lehet használni
    - Dolgozói bejelentkezés
    - Adminisztrátori bejelentkezés
- Új termék bevitele az adatbázisba
    - Csak alkalmazotti felhasználónévvel és jelszóval férhetnek hozzá
    - Kötelezően a következő mezők megadásával: kategória, márka, típus, szín, ár, darabszám, leírás, kép.
- Meglévő termékek módosítása (Arra az esetre, ha az új termék bevitelénél a felhasználó hibát vétett és nem megfelelő adatot adott meg, vagy ha az adott terméket nem forgalmazza már a cég a webshopon keresztül, akkor a terméket inaktívra lehet állítani)
    - Csak alkalmazotti felhasználónévvel és jelszóval férhetnek hozzá
    - Találatok szűrése márkára, feltöltés dátumára, és kategóriára
    - Találatok oldalakra tördelése
- Új hír írása
    - Csak alkalmazotti felhasználónévvel és jelszóval férhetnek hozzá
    - Kötelezően a következő mezők megadásával: cím, kép, szöveg
- Rendelések megjelenítése
    - Csak alkalmazotti felhasználónévvel és jelszóval férhetnek hozzá
    - Találatok szűrése megrendelő nevére és rendelés dátumára
    - Találatok oldalakra tördelése
- Új dolgozó hozzáadása
    - Csak adminisztrátori felhasználónévvel és jelszóval férhetnek hozzá
    - Kötelezően felhasználónév és jelszó megadásával
    - A jelszót SHA256 algoritmussal titkosítsa, és utána írja be az adatbázisba
- Új adminisztrátor hozzáadása
    - Csak adminisztrátori felhasználónévvel és jelszóval férhetnek hozzá
    - Kötelezően felhasználónév és jelszó megadásával
    - A jelszót SHA256 algoritmussal titkosítsa, és utána írja be az adatbázisba

3.2.2. Minőségi követelmények

- Maximum 2 másodperces válaszidők

- Jól átlátható menü és grafikus felület
- Reszponzivitás
- Modularitás és egyszerű frissíthetőség
- Könnyű back-up és visszaállítási folyamat

3.3. Szerepkörök és Use-Case diagram

A *Melléklet 2. ábrája* a szerepköröket írja le Use-Case diagram formájában.

3.3.1. Cég alkalmazottja

A cég alkalmazottja új terméket tud bevinni az adatbázisba (amely megjelenik az oldalon), terméket tud módosítani (bármilyen hiba esetén), új hírt tud felvinni az adatbázisba (a weboldal fő oldalán megjelenik), a rendeléseket meg tudja tekinteni különböző célzattal. A rendszergazda oszt ilyen jogosultságot. Ez alatt azt a felhasználót értjük, akit a cég a következő feladatokkal bíz meg:

- új termékek bevitele az adatbázisba
- esetlegesen egy-egy termék módosítása (például emberi hibából fakadó elírás esetén)
- új hír bevitele az adatbázisba
- rendelések megtekintése előkészítés vagy küldés céljából

3.3.2. Adminisztrátor

Az adminisztrátor az a felhasználó, aki az alkalmazotti és adminisztrátori jogosultságot osztja ki meghatározott emberek számára. Minden egyes adminisztrátornak saját jelszava és felhasználóneve van. Eléri a desktop alkalmazás e részére vonatkozó funkciót.

3.4. Algoritmusok és főbb funkciók működése

„Algoritmizálásnak nevezzük azt a folyamatot, amikor egy probléma megoldása érdekében meghatározott lépések sorozatát hajtjuk végre. Ha egy elvégzendő cselekvéssorozatot lépésről lépésre átgondolunk, akkor algoritmust készítünk egy adott cél elérésére.”[3]

Ebben a pontban szeretném röviden részletezni az asztali alkalmazásom főbb funkcióinak működését a mellékletben szereplő folyamatábrák alapján.

[3] Forrás: https://tudasbazis.sulinet.hu/hu/informatika/informatika/informatika-5-evfolyam/problemak-megfogalmazasa-felvetese/az-algoritmus-hetkoznapi-fogalmanak-megismerese

3.4.1. Bejelentkezés

A *Melléklet 3. ábra* a bejelentkezési folyamatot írja le egyszerűsített formában folyamatábrával.

A bejelentkezés gomb megnyomásakor a program az előzőleg megadott felhasználónév és jelszó párost „kiszedi” a szöveg mezökből. Következő lépésben a program a „kiszedett” jelszót SHA265 algoritmussal titkosítja, majd a titkosított jelszót egy változóban eltárolja. Ez után a program vizsgálatot végez, hogy a megadott felhasználónév és titkosított jelszó párosnak van-e megfelelő rekord az adatbázisban. A program az alapján dönti el melyik táblában kell keresnie a felhasználót, hogy melyikre gombra kattintottunk a folyamat legelején (Dolgozói vagy adminisztrátori bejelentkezés). Ha a megadott adatokkal létezik felhasználó, akkor annak függvényében, hogy dolgozóról vagy adminisztrátorról van szó, megtörténik a bejelentkezés és megjelenik a panel, amely lehetővé teszi a munkát. Ha a megadott adatokkal nem létezik felhasználó, akkor a program kivételt dob és a folyamat megszakad.

3.4.2. Új dolgozó és adminisztrátor hozzáadása

A *Melléklet 4. ábra* az új dolgozó és adminisztrátor hozzáadási folyamatot írja le egyszerűsített formában folyamatábrával.

A hozzáadás gomb megnyomásakor a program az előzőleg megadott felhasználónév és jelszó párost „kiszedi” a szöveg mezökből. Következő lépésben a program a megvizsgálja, hogy van-e üres mező. Ha igen, akkor kivételt dob és a folyamat megszakad. Ha nincs üres mező, akkor a „kiszedett” jelszót SHA265 algoritmussal titkosítja, majd a titkosított jelszót egy változóban eltárolja. Ez után a program a megadott felhasználónevet és a titkosított jelszót beírja az adatbázis *admins* vagy *employees* táblájában, attól függően, hogy előzőleg melyik gombra kattintottunk.

3.4.3. Termék feltöltés

A *Melléklet 5. ábra* a termék feltöltést írja le egyszerűsített formában folyamatábrával.

A feltöltés gombra kattintáskor a program megvizsgálja, hogy minden szükséges adatot megadtunk-e. Ha van üresen hagyott mező, akkor a program kivételt dob és a folyamat megszakad. Ha nincs üresen hagyott mező, akkor a program „kiszedi” a szöveg mezökből az adatokat. Ez után a program megvizsgálja, hogy helyesek-e az adatok (pl. számtípushoz nem adtunk-e meg betűt). Ha van helytelen adat, akkor a program kivételt dob és a folyamat megszakad. Következő lépésként azt vizsgáljuk, hogy a kép létezik-e a megadott mappában. Ha igen, akkor jön a kivételdobás és a folyamat megszakad. Ez után

jön az új rekord létrehozása az adatbázisban a megadott adatokkal és a kép bemásolása a megfelelő mappába.

3.4.4. Termékmódosítás

A *Melléklet 6. és 7. ábra* a termékmódosítást írja le egyszerűsített formában folyamatábrával.

A termékmódosításhoz elengedhetetlen először a termék megjelenítés folyamatát leírni. A termékmódosítás menüpontra kattintva a program feltölt egy DataGridView-t. Először minden esetben beállított szűrés nélkül fognak megjelenni a termékek. Ez után van lehetőségünk szűrni. A szűrés gombra kattintva a programunk „kiszedi" a szöveg mezökből a beírt/beállított adatokat. Természetesen ez esetben az üresen hagyott szöveg mezőket nem veszi figyelembe. Ezek alapján módosítja a program az SQL[4] lekérdezést és tölti fel újra a *DataGridView-t.*[5]

A megfelelő cellára való kattintás után van lehetőségünk átírni az adott értéket. Az adott érték változtatása esetén a program megvizsgálja, hogy az új érték megegyezik-e a jelenlegi, adatbázisban szereplő értékekkel. Ha igen, akkor semmi nem fog történni. Ha nem, akkor a program frissíti az adott rekordhoz való cella értékét és bekerül az adatbázisba. Legvégső lépésként a DataGridView-t újra feltöltjük az adatbázisból, hiszen ha ez a lépés kimarad, nem fogjuk egyből látni a módosított terméket.

3.4.5. Új hír írása

A *Melléklet 8. ábra* az új hír írását írja le egyserűsített formában folyamatábrával.

A feltöltés gombra kattintáskor a program megvizsgálja, hogy minden adatot megadtunk-e. Ha maradt üresen mező, akkor a program kivételt dob és a folyamat megszakad. A következő lépés, hogy megvizsgáljuk a kiválasztott kép létezik-e már a megadott mappában. Ha igen, akkor kivételt dob a program és a folyamat megszakad. Ha a kép nem létszik a megadott mappában, akkor a megadott adatokat feltölti az adatbázisba és a képet bemásolja a szükséges mappába.

3.4.6. Rendelések megtekintése

A *Melléklet 9.ábra* a rendelések megtekintése funkció működését írja le egyszerűsített formában folyamatábrával.

[4] SQL (Structured Query Language): Relációsadatbázis-kezelők lekérdezési nyelve. (Forrás: https://hu.wikipedia.org/wiki/SQL)
[5] DataGridView: Egy kifejezetten adathalmazok és adataik megjelenítésére (pl. adatbázis) alkalmas táblázat.

A termékmódosítás menüpontra kattintáskor a program feltölt egy DataGridView-t. Először minden esetben beállított szűrés nélkül fognak megjelenni a termékek. Ez után van lehetőségünk szűrni. A szűrés gombra kattintva a programunk megvizsgálja a beállított szűréseket és az alapján fogja újra feltölteni a DataGridView-t. Természetesen szűrésnél az üresen hagyott mezőt nem veszi figyelembe. Ezek alapján módosítja a program az SQL lekérdezést és tölti fel újra a DataGridView-t.

3.4.7. Oldalakra tördelés

A *Melléklet 10. ábra* az oldalakra tördelés folyamatát írja le egyszerűsített formában folyamatábrával.

Általánosan kijelenthető, hogy az oldalakra tördelést az SQL lekérdező nyelvben használatos LIMIT és OFFSET kulcsszavakkal valósítottam meg. Az elkészített alkalmazásomban jelenleg a LIMIT értéke fixen 25-re van állítva. A LIMIT értéke megmutatja, hogy az SQL lekérdezés során hány rekordot szeretnénk megjeleníteni. Az oldal navigációs gombjaira való kattintáskor mindig az előzőleg említett OFFSET értékét számoljuk, vagy adunk meg neki konkrét értéket. Az OFFSET értéke gyakorlatilag azt mutatja meg, hogy hányadik sortól szeretnénk megjeleníteni a lekérdezett adatokat.

Mivel ebben az esetben az oldaltördelést nem lehet egyetlen folyamatként leírni, ezért ebben a pontban több, egymástól szorosan függő folyamatot szeretnék röviden részletezni.

Először az összes oldal kiszámítását kell leírnunk. Első lépésként meg kell adnunk a limitet (hány sort szeretnénk oldalanként látni), offsetet (hányadik sortól szeretnénk megjeleníteni az adatbázis rekordjait) és a jelenlegi oldalt. Ezeket változókban tároljuk el. Következő lépésként meg kell számolnunk, hogy pontosan hány rekord található az adatbázis adott táblájában, majd ezt is eltároljuk egy változóban. Az összes oldal kiszámítása a következő képlet alapján történik: *összes oldal=sorok száma/kívánt sorok száma oldalanként*. A kapott értéket minden esetben felfelé kerekítjük! Ha az eredmény 0, akkor a változót, amely tárolja az összes oldal számát, egyenlővé kell tenni 1-el.

A legelső oldal gombra kattintáskor az offsetet 0-ra, a jelenlegi oldalt pedig 1-re állítjuk. Ez után az offset és limit alapján módosítjuk az SQL lekérdezést, amellyel feltöltjük a DataGridView-t, majd újra feltöltjük azt.

Az előző oldal gombra kattintáskor először megvizsgáljuk, hogy a jelenlegi oldal száma nagyobb-e, mint 1. Ha nem, akkor feltételezhető, hogy már az első oldalon vagyunk, ezért ekkor semmi nem fog történni. Ellenkező esetben az offsetből kivonjuk a limit

értékét, a jelenlegi oldal számából pedig kivonunk egyet. Végső lépésként szintén az offset és limit értékek alapján módosítjuk az SQL lekérdezést, amely alapján feltöltjük a DataGridView-t, majd újra fel is töltjük azt.

A következő oldal gombra kattintáskor megvizsgáljuk, hogy a jelenlegi oldal kisebb-e az összes oldal számánál. Ha nem, akkor feltételezhető, hogy az utolsó oldalon vagyunk. Ellenkező esetben a jelenlegi oldal számához hozzáadunk egyet és az offsethez pedig hozzáadjuk a limit értékét. Végső lépésként itt is az offset és limit alapján módosítjuk az SQL lekérdezést és újra feltöltjük a DataGridView-t.

Az utolsó oldal gombra kattintáskor a jelenlegi oldal számát egyenlővé tesszük az összes oldal számával, az offset változó pedig a következő képlet szerint alakul: *offset=(összes oldal száma-1)*limit*. Ez után jön az SQL lekérdezés manipulálása, majd a DataGridView újra feltöltése.

3.5. Tesztesetek

„A tesztelés egy rendszer vagy program kontrollált körülmények melletti futtatása, és az eredmények kiértékelése."[6] Egy szoftver fejlesztése során elengedhetetlen a tesztelés, illetve az általunk megírt kód helyességének ellenőrzése.

„Tesztelésre azért van szükség, hogy a szoftver termékben meglévő hibákat még az üzembe helyezés előtt megtaláljuk, ezzel növeljük a termék minőségét, megbízhatóságát. Abban szinte biztosak lehetünk, hogy a szoftverben van hiba, hiszen azt emberek fejlesztik és az emberek hibáznak."[7]

A záródolgozatomban a legtöbb tesztelést úgynevezett feketedobozos tesztelési módszerrel végeztem, amely azt jelenti, hogy programkódok használatát mellőzve úgy végeztem a legtöbb funkció tesztelését, mintha egy átlag felhasználó próbálná meg használni.

Úgy gondoltam, hogy érdemes lenne hasznosítani az iskolában elsajátított tudásom a teszteléssel kapcsolatban, ezért a bejelentkezés funkcióval kivételt tettem és úgy nevezett unit-tesztet is írtam a helyességének vizsgálatára. A unit-teszt azt jelenti, hogy egy adott paraméterre ismerjük a vizsgált metódus kívánt visszatérési értékét, majd a kapott és kívánt értékeket összehasonlítja nekünk a tesztprogramunk. A legtöbb fejlesztői környezet (köztük természetesen a Visual Studio is) támogatja a unit-tesztek írását és az ilyen teszt projekteket másképp kezeli. Jelen esetben a tesztek futtatása külön funkció segítségével

[6] Forrás: https://hu.wikipedia.org/wiki/Szoftvertesztel%C3%A9s
[7] Forrás: http://aries.ektf.hu/~gkusper/SzoftverTeszteles.pdf

lehetséges. Ha lefutottak a tesztek, akkor a fejlesztői környezet megjeleníti mely tesztek voltak sikeresek és melyek sikertelenek (más szóval: megfelelően működnek-e a tesztelt metódusok, avagy sem).

Négy darab eljárást valósítottam meg a teszt projektben a bejelentkezés funkció vizsgálatára. A négy darab eljárás egyenként meghívja a bejelentkezést megvalósító metódust. Egyszer vizsgálja, hogy nem megfelelő felhasználónév és nem megfelelő jelszó esetén történik-e kivételdobás, utána csak rossz jelszó esetén történik-e, következőre csak rossz felhasználónévvel történik-e, majd utoljára azt vizsgálja, hogy megfelelő adatok esetén nem kapunk-e kivételt. Természetesen az elvárás, hogy az első 3 esetben történjen kivételdobás, az utolsó esetben pedig pont az ellenkezője, hogy ne történjen kivételdobás.

Esetemben az összes unit-teszt eljárásom sikeresen lefutott, tehát feltételezhető, hogy a bejelentkezés funkció megbízhatóan működik.

4. Webshop

A webes alkalmazásom egy internetes áruház, gyakorlatilag maga a TDWebshop. Habár ez számít a rendszer legfőbb elemének, az asztali alkalmazásra szükség van, mivel a webshop csak megjeleníteni tudja a termékeket és új webshop felhasználót beírni az adatbázisba, valamint rendeléseket beírni az adatbázisba.

A fejlesztői dokumentáció e részében szeretném felsorolni, hogy milyen fejlesztői környezetet használtam (IDE, nyelv, könyvtárak és keretrendszerek), a számítógép konfigurációját (amelyen az alkalmazás készült) és a funkcionális, valamint minőségi követelményeket, továbbá tisztáznám a szerepköröket. Ezek az információk nagyon fontosak lehetnek azok számára, akiknek egy ilyen fejlesztői dokumentáció elsősorban szól.

A fejlesztői dokumentáció asztali alkalmazásra vonatkozó részének elején kifejtett véleményem alapján itt is felsorolás és vázlat szerűen fogom leírni a fent említett pontokat.

Az előzőleg említetteken túl szeretnék említést tenni arról, hogy ezen alkalmazás fő funkcióri milyen algoritmusok alapján működnek.

4.1. Fejlesztői környezet leírása

4.1.1. Fejlesztői környezet

- Atom szövegszerkesztő 1.44.0

4.1.2. Választott nyelvek

- PHP 7.4.2
- JavaScript
- CSS
- HTML

4.1.3. Könyvtárak/keretrendszerek

- jQuery 3.4.1
- Bootstrap 4

4.1.4. Futtatási környezet leírása

- OS: Windows 8.1
- Böngésző: Mozilla Firefox 73.0.1
- CPU: AMD FX8320
- RAM: 8GB
- GPU: GeForce GTX1050
- HDD: 500GB
- Alaplap: AsRock 970 Pro3 R2.0

4.2. Specifikációk és követelmény analízis

4.2.1. Funkcionális követelmények

- A főoldalon slideshow, amelyben mindig az oldallal/céggel kapcsolatos legújabb 6 hír jelenik meg
- Regisztrálási lehetőség:
    - Kötelezően vezetéknév, keresztnév, e-mail cím és jelszó megadásával.
    - A jelszót SHA256 algoritmussal titkosítsa és utána írja be az adatbázisba
- Belépési lehetőség:
    - E-mail cím és jelszó megadásával
- Kosár:
    - Csak bejelentkezés után elérhető
    - A kosárba rakott termékek márkájának, típusának, kategóriájának, árának és rendelni kívánt mennyiségének megjelenítése
    - Kosárba rakott termékek egyenkénti törlésének lehetősége

    - Közvetlenül rendelési lehetőség form segítségével, a név mező kitöltése automatikus (korábban regisztrációnál megadott Vezetéknév és Keresztnév alapján)
- A fő menüben csak a termékek fő kategóriái jelenjenek meg. Egy fő kategóriára való kattintás után van lehetőség kiválasztani egy alkategóriát, ami után a termékek jelennek meg.
- Adatbázisban tárolt termékek megjelenítése
    - Oldalakra tördelés
    - Kép, márka, típus, ár, szín, jobbkezes vagy balkezes-e megjelenítése minden terméknél
    - Egy termékre való kattintás után az eddig megjelenített adatokon túl megjeleníti az adott termék leírását és ekkor van lehetőség az adott terméket a kosárba rakni.
    - Ha a termék nem gitár, akkor ne jelenítse meg a jobbkezességet, mivel abban az esetben ez a tulajdonság nem releváns.
    - A termékekre ár szerinti és jobbkezesség szerinti szűrés (természetesen a kiválasztott kategórián felül)

4.2.2. Minőségi követelmények

- Maximum 2 másodperces válaszidők
- Jól átlátható menü
- Reszponzivitás
- Modularitás és egyszerű frissíthetőség
- Könnyű back-up és visszaállítási folyamat

4.3. Szerepkörök és Use-Case diagram

A *Melléklet 2. ábrája* a szerepköröket írja le Use-Case diagram formájában.

4.3.1. Látogató

A látogató az a felhasználó, aki nincs regisztrálva a webshopban. Az adatbázis termékek tábláiból lekérdezéseket hajthat végre, vagyis meg tudja tekinteni a termékeket a weboldalon. A rendelés funkciót elérni nem tudja, de regisztrálásra van lehetősége, ami után regisztrált felhasználóvá válik.

4.3.2. Regisztrált felhasználó

A regisztrált felhasználó az a felhasználó, aki előzőleg sikeresen kitöltötte a regisztrációs űrlapot és adatai bekerültek az adatbázis felhasználókat tároló táblájába. Meg tudja tekinteni a termékeket a weboldalon és eléri a rendelés funkciót is.

4.4. Algoritmusok és főbb funkciók működése

4.4.1. Bejelentkezés

A *Melléklet 11. ábra* a bejelentkezési folyamatot írja le egyszerűsített formában folyamatábrával.

A bejelentkezés gomb megnyomásakor a program az előzőleg megadott felhasználónév és jelszó párost „kiszedi" a szövegmezőkből. Következő lépésben a program a „kiszedett" jelszót SHA256 algoritmussal titkosítja, majd a titkosított jelszót egy változóban eltárolja. Ez után a program vizsgálatot végez, hogy a megadott felhasználónév és titkosított jelszó párosnak van-e megfelelő rekord az adatbázis users táblájában. Ha a megadott adatokkal létezik felhasználó, akkor megtörténik a bejelentkezés, ez esetben ez azt jelenti, hogy egy úgynevezett session változóban (superglobal) eltároljuk a bejelentkezett felhasználóhoz tartozó elsődleges kulcs értékét. Ha a megadott adatokkal nem létezik felhasználó, akkor természetesen nem történik meg a bejelentkezés és visszajelzést adunk a felhasználónak.

4.4.2. Regisztráció

A *Melléklet 12., 13., 14., 15. és 16. ábra* a regisztráció és az adatok validációs folyamatát írja le egyszerűsített formában folyamatábrával.

Ebben az esetben minden adat validációját ketté kell bontani kliens oldali és szerver oldali folyamatokra, mivel a legtöbb böngésző alkalmas kliens oldali forráskódok megjelenítésére és megváltoztatására, ezért kell minden esetben szerver oldalon is validációt végzeni.

A vezetéknév validációnál billentyű felengedésekor az alkalmazás „kiszedi" a megadott adatot a szövegmezőből, majd változóban eltárolja. Ez után megvizsgáljuk, hogy a megadott adat nagyobb-e 2 karakternél. Ha igen, akkor a szövegmező keretének színe zöldre változik, ezzel jelezve, hogy a megadott adat helyes, a visszatérési értéke pedig igaz lesz. Ha nem, akkor a szövegmező keretének színe pirosra változik, ezzel jelezve, hogy a megadott adat helytelen, a visszatérési érték pedig hamis lesz. A szerveroldali hitelesítés nagyon hasonlóan működik. Az egyetlen különbség, hogy itt nincs visszajelzés, a

függvénynek visszatérési értéke lesz. Ha nem megfelelő a megadott adat, akkor hamis, ha megfelelő az adat, akkor igaz lesz a visszatérési érték.

A keresztnév validáció teljes mértékben megegyezik a vezetéknév hitelesítésével mind kliens, mind szerver oldalon.

E-mail cím hitelesítésnél billentyű felengedésekor az alkalmazás „kiszedi" a megadott adatot a megfelelő szövegmezőből, majd változóban eltárolja. Ez után megvizsgáljuk, hogy a megadott e-mail cím megegyezik-e, egy előre meghatározott, úgy nevezett reguláris kifejezéssel. Ha megegyezik, a szövegmező keretének színe zöldre változik, a függvény visszatérési értéke pedig igaz lesz. Ha nem egyezik meg, akkor a szövegmező keretének színe pirosra változik, a visszatérési érték pedig hamisra. A szerveroldalon való hitelesítés ebben az esetben is annyiban különbözik, hogy a függvénynek itt is csak visszatérési értéke lesz.

A jelszó hitelesítésnél billentyű felengedésekor az alkalmazás „kiszedi" a megadott adatot a megfelelő szövegmezőből, majd változóban eltárolja. Ez után megvizsgáljuk, hogy a jelszó hossza nagyobb-e 8 karakternél. Ha nem, akkor a visszatérési érték hamis lesz, a szövegmező keretének színe pedig piros. Ha igen, akkor megvizsgáljuk, hogy tartalmaz-e kisbetűt, nagybetűt, számot. Ha nem, akkor a visszatérési érték ez esetben is hamis lesz, a keret színe pedig piros. Ha igen, akkor a visszatérési érték igaz, a szövegmező keret színe pedig zöld. A szerver oldali hitelesítés az előzőekben tárgyalt módon különbözik ebben az esetben is.

A formon a küldés gombra kattintva először megvizsgáljuk van-e üresen hagyott mező. Ha igen, a weboldal erre felhívja a figyelmet az üresen hagyott mezők keretének pirosra változtatásával. Ha nincs, akkor megvizsgálja, hogy az előzőleg lefutott függvények visszatérési értékei igazak-e. Ha igen, akkor átadjuk az összes megadott adatot szerver oldali hitelesítésre. Ha nem, akkor nem történik semmi. Ez után az előzőekben kifejtett módon megtörténik az adatok újra hitelesítése, ezúttal szerver oldalon. Ha az adatok megfelelőek, akkor az alkalmazás beírja az adatbázisba az adatokat. Ha nem, akkor hibát dob.

4.4.3. Termék megjelenítés

A *Melléklet 17. ábra* a termék megjelenítés folyamatát írja le egyszerűsített formában folyamatábrával.

A webshop fő menüjében az egyik fő kategóriára való kattintáskor van lehetőségünk választani alkategóriát. Alkategória választásakor a böngésző url savjába

beírjuk a választott alkategória adatbázisban szereplő elsődleges kulcs értékét, majd ezzel a szűréssel egy SQL lekérdezést futtatunk le a weboldalon, amely megjeleníti a kiválasztott alkategória termékeit. Ez után megvizsgáljuk, hogy a kiválasztott alkategória valamilyen gitár-e. Ha igen, akkor a további szűrésre 2 lehetőség van: ár és jobbkezesség szerint. Ha nem, akkor csak árra adhatunk meg további szűrést. Először a termékek minden esetben csak kategóriaszűréssel jelennek meg. A megjelenítés után van lehetőségünk több szűrést kiválasztani. A szűrés gombra kattintva a megadott szűréseknek megfelelően módosítjuk az SQL lekérdezést (természetesen az üresen hagyott mezőt nem veszi figyelembe), majd újra megjelenítjük a termékeket.

4.4.4. Oldalakra tördelés

A *Melléklet 10. ábra* az oldalakra tördelés folyamatát írja le egyszerűsített formában folyamatábrával.

Általánosan kijelenthető, hogy az oldalakra tördelést az SQL lekérdező nyelvben használatos LIMIT és OFFSET kulcsszavakkal valósítottam meg. Az elkészített alkalmazásomban jelenleg a LIMIT értéke fixen 25-re van állítva. A LIMIT értéke megmutatja, hogy az SQL lekérdezés során hány rekordot szeretnénk megjeleníteni. Az oldal navigációs gombjaira való kattintáskor mindig az előzőleg említett OFFSET értékét számoljuk, vagy adunk meg neki konkrét értéket. Az OFFSET értéke gyakorlatilag azt mutatja meg, hogy hányadik sortól szeretnénk megjeleníteni a lekérdezett adatokat.

Mivel ebben az esetben az oldaltördelést nem lehet egyetlen folyamatként leírni, ezért ebben a pontban több, egymástól szorosan függő folyamatot szeretnék röviden részletezni.

Először az összes oldal kiszámítását kell leírnunk. Első lépésként meg kell adnunk a limitet (hány sort szeretnénk oldalanként látni), offsetet (hányadik sortól szeretnénk megjeleníteni az adatbázis rekordjait) és a jelenlegi oldalt. Ezeket változókban tároljuk el. Következő lépésként meg kell számolnunk, hogy pontosan hány rekord található az adatbázis adott táblájában, majd ezt is eltároljuk egy változóban. Az összes oldal kiszámítása a következő képlet alapján történik: *összes oldal=sorok száma/kívánt sorok száma oldalanként*. A kapott értéket minden esetben felfelé kerekítjük! Ha az eredmény 0, akkor a változót, amely tárolja az összes oldal számát, egyenlővé kell tenni 1-el.

A legelső oldal gombra kattintáskor az offsetet 0-ra, a jelenlegi oldalt pedig 1-re állítjuk. Ez után az offset és limit alapján módosítjuk az SQL lekérdezést, amellyel feltöltjük a DataGridView-t, majd újra feltöltjük azt.

Az előző oldal gombra kattintáskor először megvizsgáljuk, hogy a jelenlegi oldal száma nagyobb-e, mint 1. Ha nem, akkor feltételezhető, hogy már az első oldalon vagyunk, ezért ekkor semmi nem fog történni. Ellenkező esetben az offsetből kivonjuk a limit értékét, a jelenlegi oldal számából pedig kivonunk egyet. Végső lépésként szintén az offset és limit értékek alapján módosítjuk az SQL lekérdezést, amely alapján feltöltjük a DataGridView-t, majd újra fel is töltjük azt.

A következő oldal gombra kattintáskor megvizsgáljuk, hogy a jelenlegi oldal kisebb-e az összes oldal számánál. Ha nem, akkor feltételezhető, hogy az utolsó oldalon vagyunk. Ellenkező esetben a jelenlegi oldal számához hozzáadunk egyet és az offsethez pedig hozzáadjuk a limit értékét. Végső lépésként itt is az offset és limit alapján módosítjuk az SQL lekérdezést és újra feltöltjük a DataGridView-t.

Az utolsó oldal gombra kattintáskor a jelenlegi oldal számát egyenlővé tesszük az összes oldal számával, az offset változó pedig a következő képlet szerint alakul: *offset=(összes oldal száma-1)*limit*. Ez után jön az SQL lekérdezés manipulálása, majd a DataGridView újra feltöltése.

4.4.5. Kosár/rendelés

A *Melléklet 18. ábra, 19. és 20. ábra* a kosár, rendelés és rendelési adatok hitelesítése folyamatot írja le egyszerűsített formában folyamatábrával.

A fő menüben a kosár menüpontra kattintáskor első lépésként ellenőrizzük, hogy be van-e jelentkezve valaki. (Erre azért van szükség, mert habár a kosár menüpont nem elérhető bejelentkezés nélkül, az url sávba bárki be tudja írni a weboldal címe után, hogy „cart.php”.) Ha nincs senki bejelentkezve, akkor visszairányítás történik a fő oldalra. Ha be vagyunk jelentkezve, akkor megvizsgálja, hogy egy úgynevezett poszt szuperglobálban tároltunk-e el termék elsődleges kulcsát. Ha nem, akkor a kiválasztott termék elsődleges kulcsát értékét lementjük egy poszt szuperglobálban. Következő lépésként megvizsgáljuk, hogy a tömb, amelyben a kosár tartalmát tároljuk, deklarálva lett-e. Ha nem, akkor deklaráljuk, ha igen, akkor semmit nem teszünk. Ezek után megvizsgáljuk, hogy érkezett-e kosárhoz hozzáadás vagy kosárból törlés kérés-e. Ha igen, akkor a kérésnek megfelelően a kosár tartalmát tartalmazó tömbből a kérésnek megfelelően elemeket törlünk, vagy hozzáadunk. A következő lépésben a tömb elemeinek számát vizsgáljuk. Ha ez 0, akkor visszajelzésként kiírjuk, hogy a kosár tartalma üres. Ha nagyobb, mint 0, akkor egy táblázatban megjelenítjük a kosárban lévő terméket vagy termékeket.

Ezen az oldalon lehetőség van megrendelni a termékeket a táblázat alatt található formban. Az adatbázisból kikeressük az éppen bejelentkezve lévő felhasználóhoz tartozó nevet, majd egy lekérdezéssel ezt az értéket betöltjük a form név szövegmezőjébe.

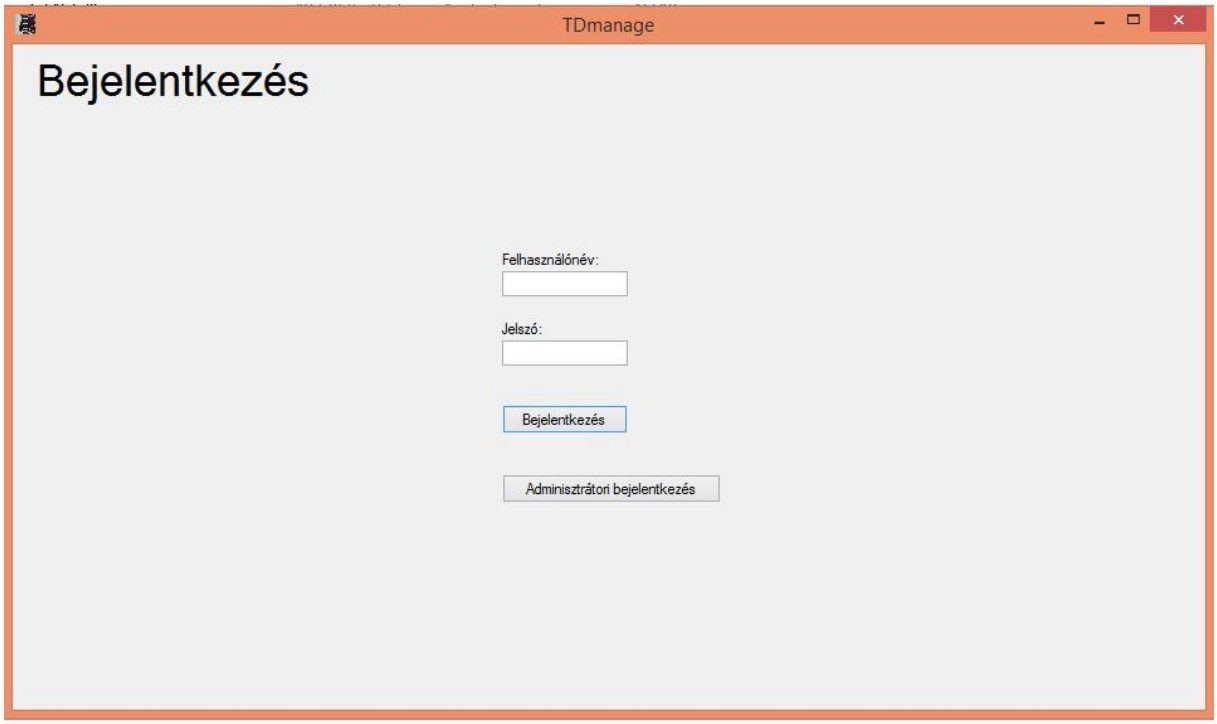
Az irányítószám szövegmezőben történő billentyű felengedésekor vizsgálatot végez az alkalmazás, hogy a beírt szám nagyobb-e, mint 999 és kisebb-e, mint 10000. Ha igen, akkor a függvény visszatérési értéke igaz lesz és a keret színe zöldre változik. Ha nem, akkor a visszatérési érték hamis lesz és a keret színe pirosra változik.

A cím szövegmezőben való billentyű felengedéskor megvizsgáljuk, hogy a megadott érték megegyezik-e egy előre definiált reguláris kifejezéssel. Ha igen, akkor a függvény visszatérési értéke igaz lesz és a megfelelő szövegmező keretének színe zöld lesz. Ha nem, akkor a függvény visszatérési értéke hamis lesz és az előzőekben tárgyaltakkal megegyezően a szövegmező kerete itt is pirosra változik.
A rendelés gombra kattintáskor a weboldal megvizsgálja, hogy van-e üresen hagyott mező. Ha igen, akkor az üresen hagyott mezőkre felhívja a figyelmet. Ha nem, akkor következő lépésként azt vizsgáljuk, hogy az előzőleg lefutott függvények visszatérési értékei igazak-e. Ha igen, akkor átadjuk az adatokat szervernek hitelesítésre.

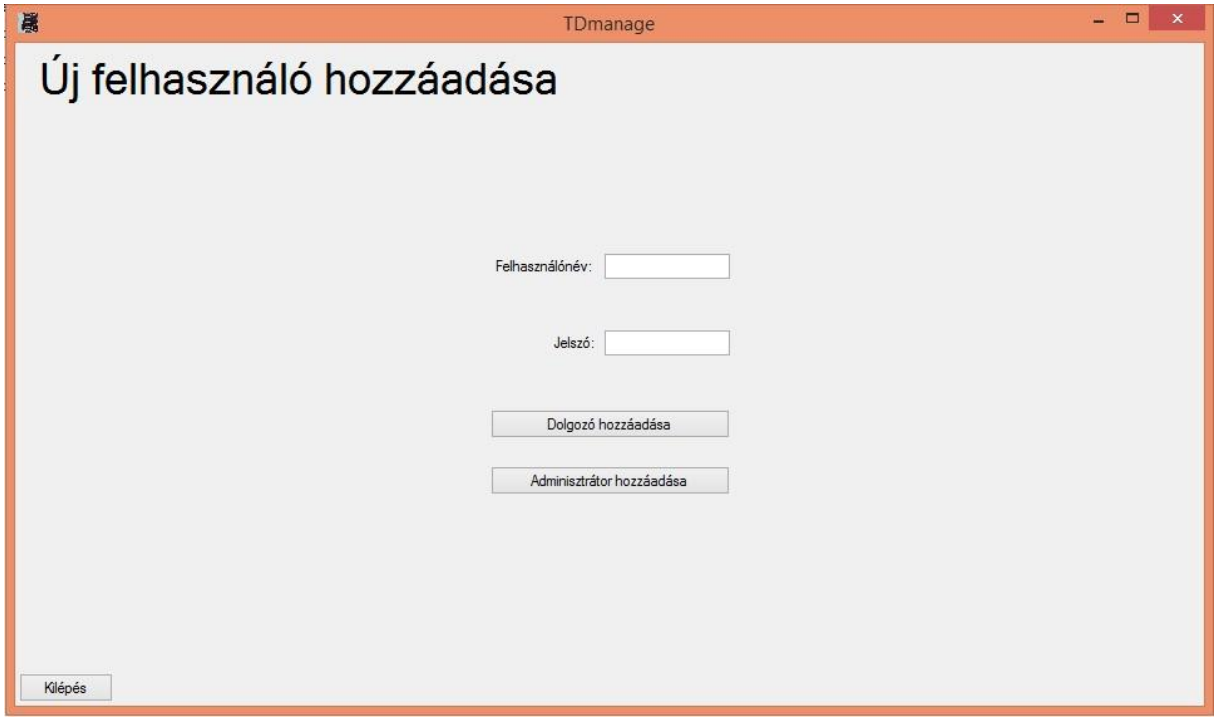
Az adatok vizsgálata a kliensoldalihoz hasonlóan működik, a különbség annyi, hogy a függvényeknek csak visszatérési értékeik lesznek. Következő lépésként egy újabb vizsgálat következik, amely megnézi, hogy az összes függvény visszatérési értéke igaz-e. Ha nem, akkor jelezzük a hibát. Ha pedig a visszatérési értékek igazak, akkor beírjuk az adatbázisba a rendelés minden szükséges adatát (termék neve, termék kategóriája, termékből rendelt darabszám, termék egységára, fizetendő ár, megrendelő neve és címe). Fontos megemlíteni, hogy habár a felhasználó egy minimális hozzáértéssel képes lehet manipulálni a megjelenített kosárban az például az árak értékét, a végén az adatbázisba nem ezek az adatok kerülnek be, hanem a szerveroldalon tömb változóban eltárolt termékek elsődleges kulcsaihoz tartozó ár értéke!

Felhasználói dokumentáció

1. Asztali alkalmazás

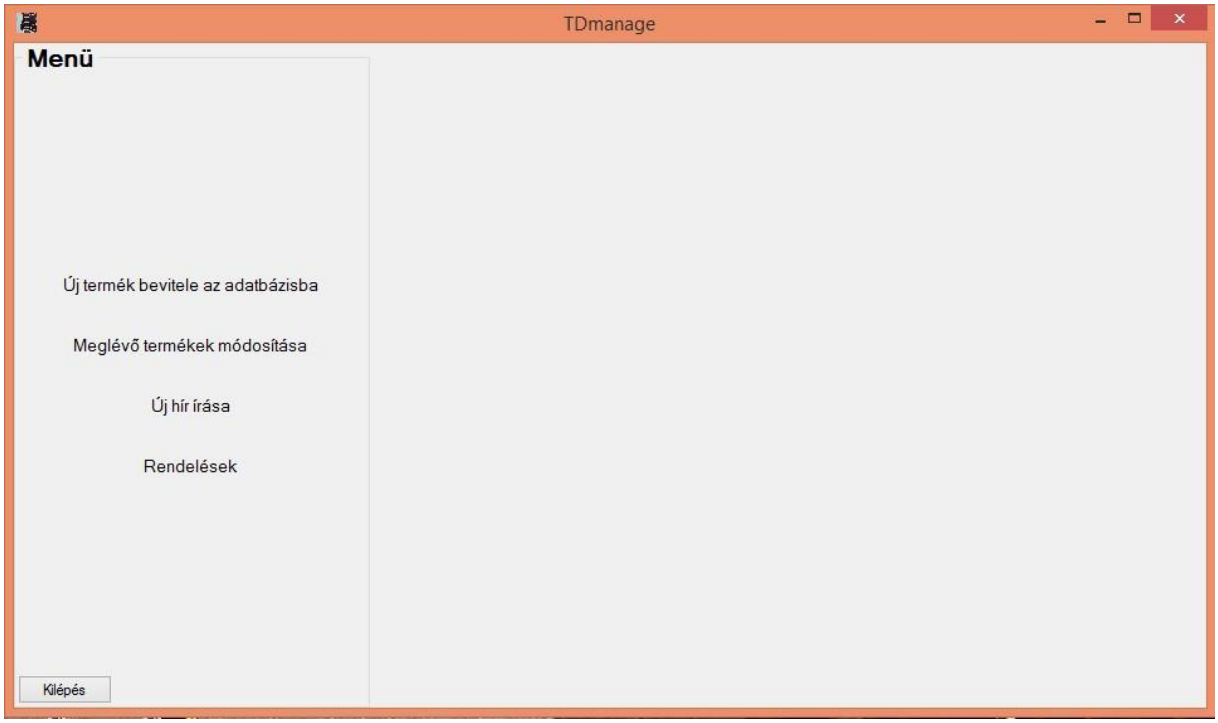


A program elindításakor a *Bejelentkezés panel* fogad minket. Itt tudjuk igazolni magunkat, hogy jogosultak vagyunk a webshop alapjául szolgáló adatbázist kezelni. Egy felhasználónév és jelszó ellenében tudunk belépni. A program különbséget tesz dolgozói és adminisztrátori bejelentkezés között. A kettő között a fent látható képen található 2 gombbal tudunk választani. Természetesen a szerepkörnek megfelelő felhasználónevet és jelszót kell megadni. Ha például dolgozói felhasználónevet és jelszót adtunk meg és kattintunk az *Adminisztrátori bejelentkezés* gombra (vagy fordítva, adminisztrátori felhasználónevet és jelszót adtunk meg kattintunk a *Bejelentkezés* gombra), akkor hibaüzenetet kapunk. Lehetséges hibák: üresen hagyott mező, nem megfelelő adatok.



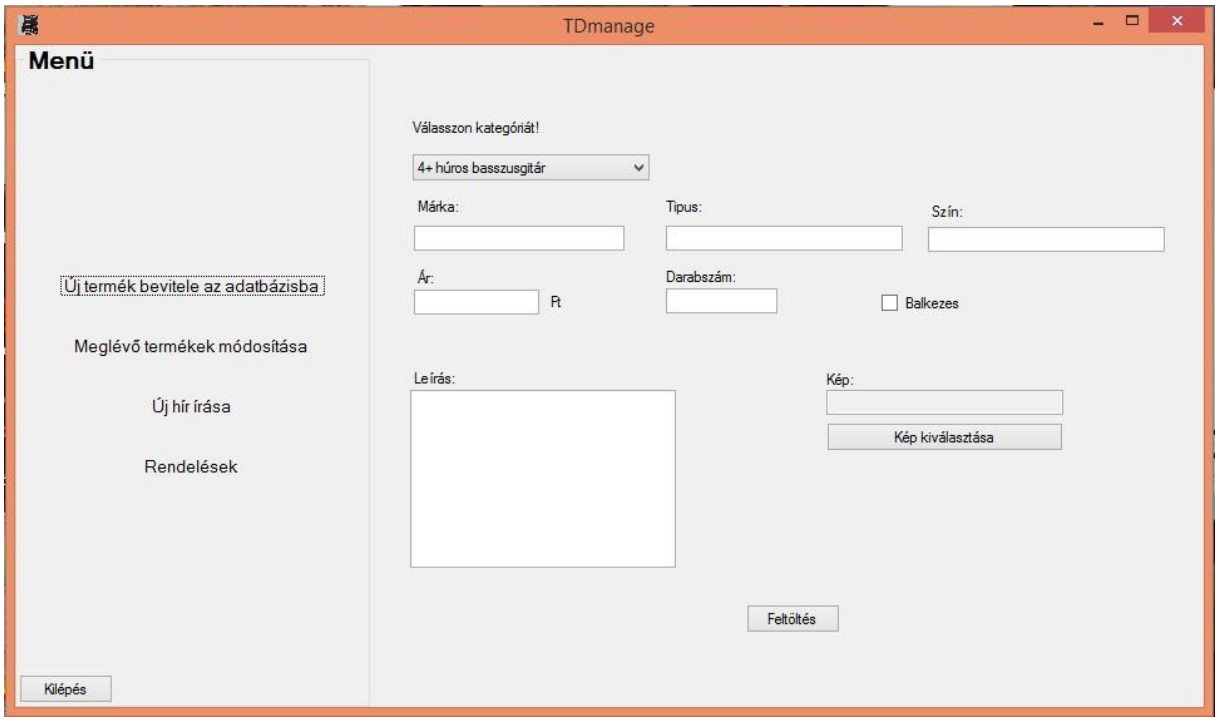
Adminisztrátori bejelentkezés esetén az *Új felhasználó hozzáadása panel* fogad minket.

Az adminisztrátor itt tud új dolgozót vagy adminisztrátort hozzáadni, amely későbbi bejelentkezésnél használható. Hogy melyik szerepkörrel szeretnénk új felhasználót hozzáadni, az fogja eldönteni, hogy a fenti képen látható 2 gomb közül melyikre kattintunk a felhasználónév és jelszó megadása után. Természetesen mindkét adat megadása kötelező. A bal alsó sarokban található a *Kilépés* gomb, amellyel az éppen bejelentkezett adminisztrátor tud kijelentkezni. Lehetséges hibák: üresen hagyott mező.



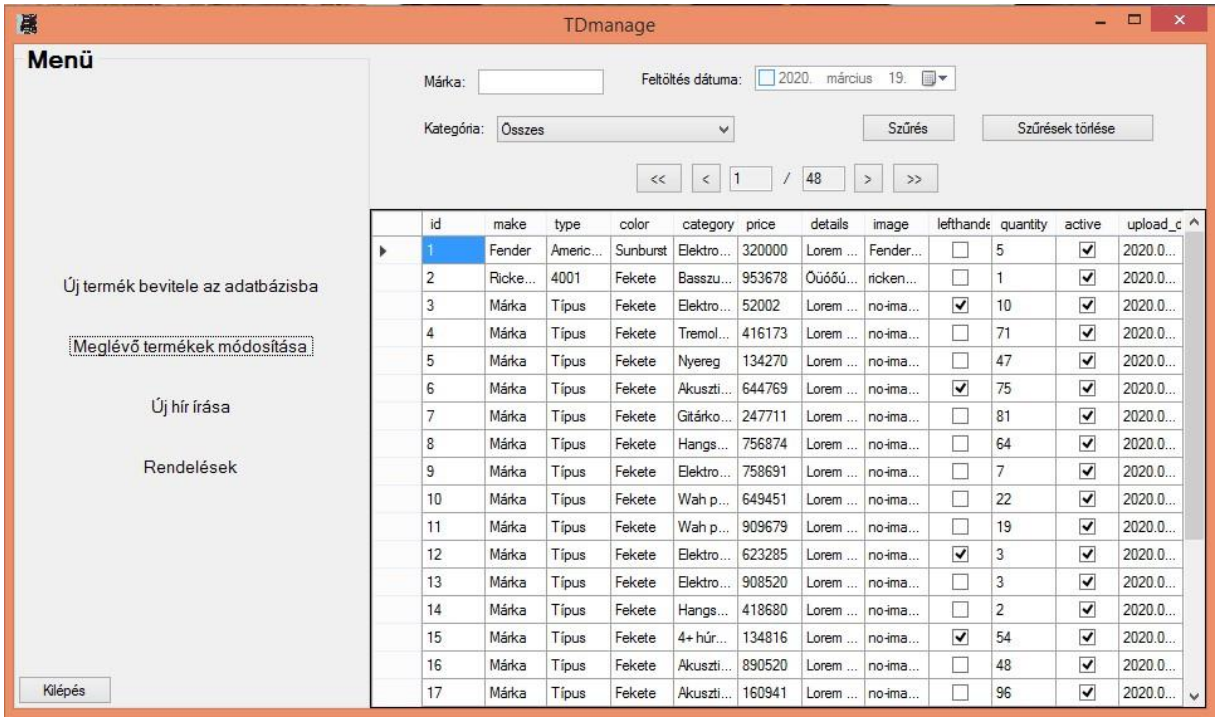
Dolgozói bejelentkezés esetén a *Fő panel* fogad minket.

A baloldalon található a menü. Itt tudjuk kiválasztani, hogy milyen feladatot szeretnénk végrehajtani (Új termék bevitele az adatbázisba, Meglévő termékek módosítása, Új hír írása, Rendelések). Ezek valamelyikére kattintva tudjuk előhozni a hozzátartozó panelt. A menü alján a *Kilépés* gomb található, amellyel az éppen belépett dolgozó tud kilépni.



Új termék bevitele az adatbázisba menüpontra kattintva előjönnek a különböző szöveg mezők, gombok és címkék, amelyek segítségével új terméket tudunk bevinni az adatbázisba.

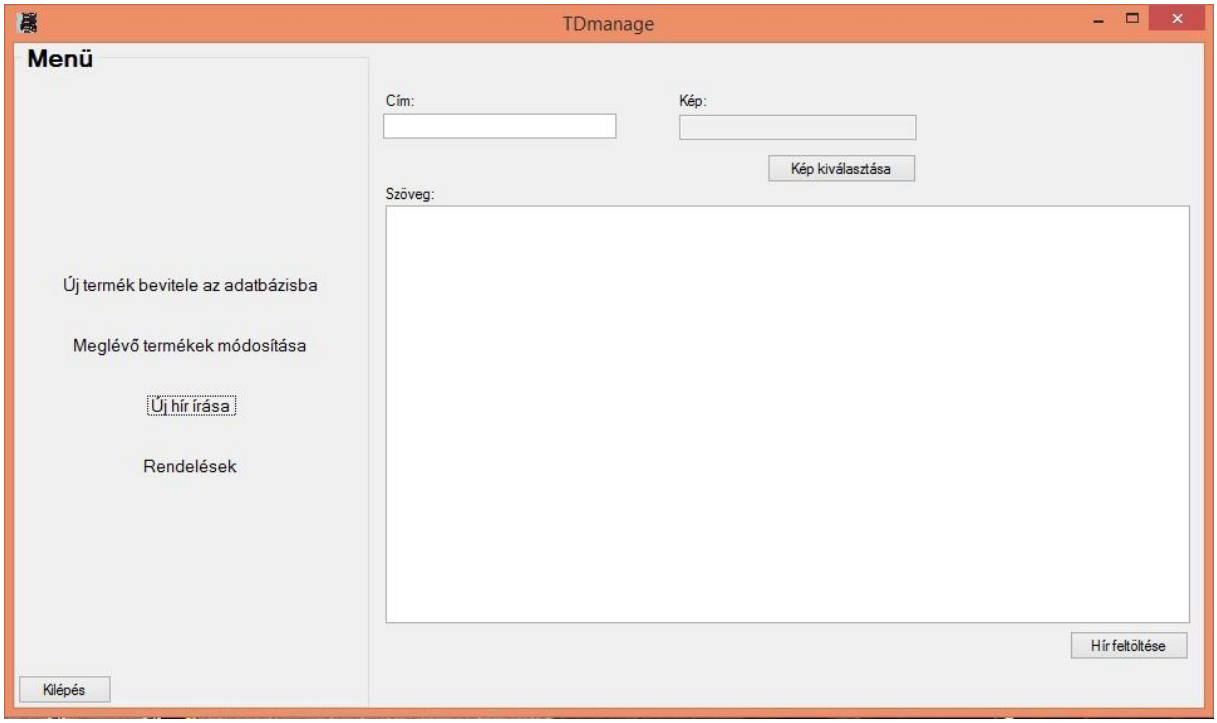
Kép kiválasztása gombra kattintva egy új ablak fog megjelenni, amelyben ki tudjuk választani a termékhez tartozó képet. Az összes mező kitöltése kötelező! Ha megadtuk az új termék minden tulajdonságát a hozzájuk tartozó szövegmezőkbe, akkor a *Feltöltés* gombra kattintáskor történik meg az adatbázisba írás. Lehetséges hibák: üresen hagyott mező, nem megfelelő formátumú adat (pl. árnál vagy darabszámnál betű beírása), a kiválasztott kép már létezik a mappában, ahol a webshop a képeket tárolja.



A *Meglévő termékek módosítása* menüpontra kattintva jelennek meg az adatbázisban lévő termékek, és a találatok szűrésére alkalmas mezők, illetve gombok.

A *Szűrés* gombra kattintva tudjuk aktiválni a megadott szűréseket (márkára, kategóriára és feltöltés dátumára). Az üresen hagyott mezőket a szűrésnél nem vesszük figyelembe. A dátumra való szűrést csak akkor veszi figyelembe a program, ha az be van pipálva. A *Szűrések törlése* gombra kattintva az összes szűrés törlődik és újra szűrés nélkül jelennek meg az adatbázisban lévő termékek.

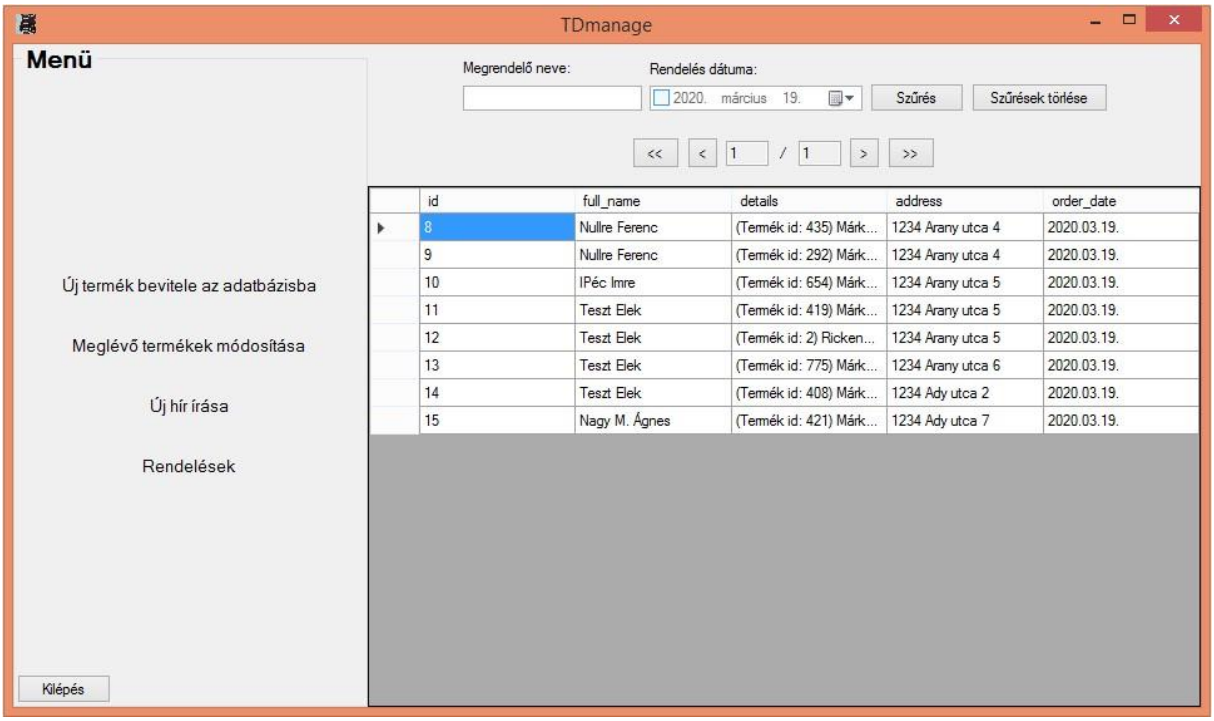
Mivel az adatok oldalakra tördelve jelenik meg (egy oldalon 25 rekord), ezért a szűrések alatt tudunk az oldalak között lépkedni. A négy gomb balról jobbra: Első oldal, Előző oldal, Következő oldal, Utolsó oldal. A gombok között láthatjuk az aktuális oldalt és az összes oldalt.



Termék adatait úgy tudjuk módosítani, hogy az adott cellára duplán kattintunk, majd átírjuk az adatot. Ez után, ha máshol kattintunk (pl. egy üres területen), akkor a módosítás aktiválódik és az adatbázisban is frissül.

Az *Új hír írása* menüpontra kattintva jelennek meg az új hír írására alkalmas szövegmezők.

A *Kép kiválasztása* gombra kattintva egy új ablak fog megjelenni, amelyben a hírhez tartozó képet tudjuk kiválasztani. A *Hír feltöltése* gombra kattintva tudjuk feltölteni a hír az adatbázisba.

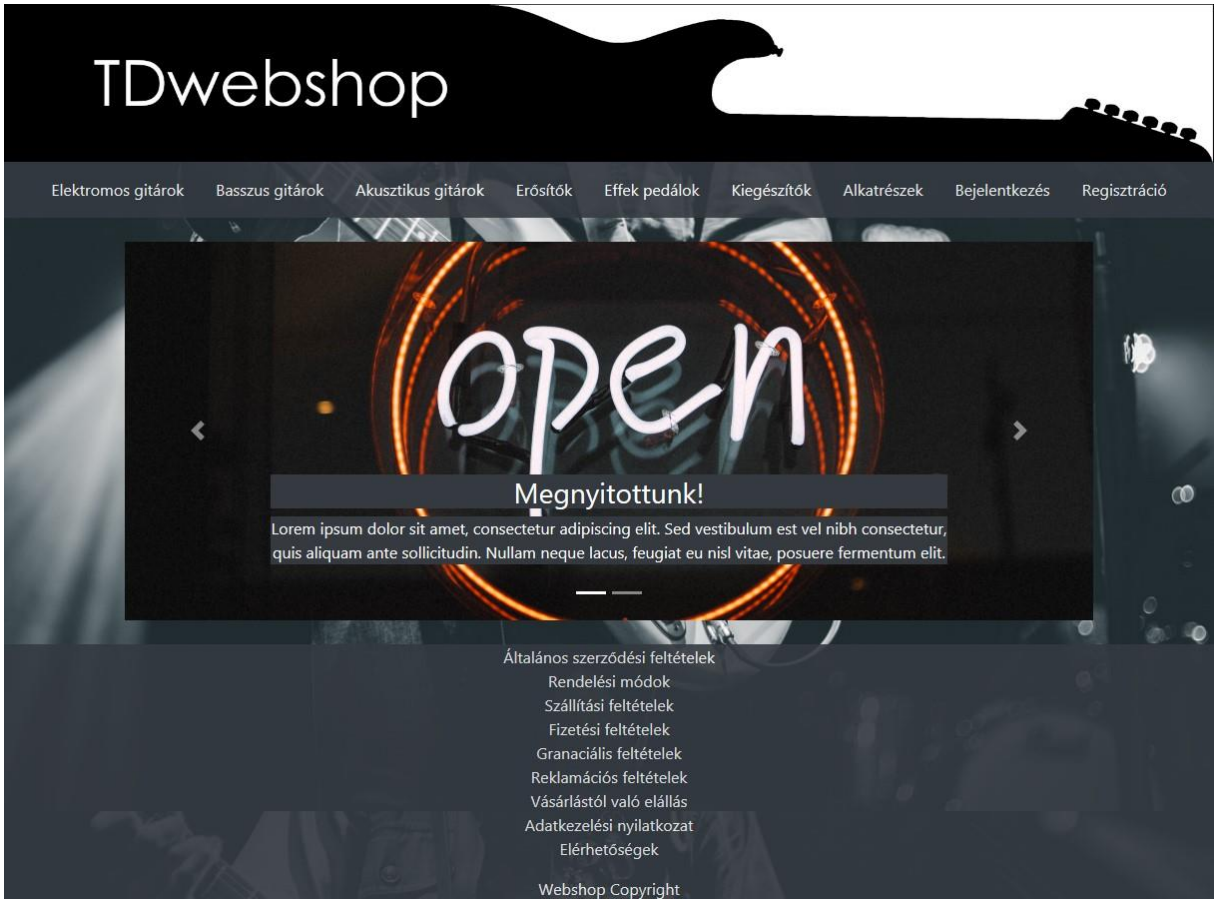


A *Rendelések* menüpontra kattintva tudjuk megtekinteni a webshopon leadott rendeléseket.

Ahogy a fenti képen is látszódik, a megjelenített rendelésekre szűréseket tudunk beállítani, jelen esetben a megrendelő nevére és a rendelés dátumára.

Mivel az adatok oldalakra tördelve jelenik meg (egy oldalon 25 rekord), ezért a szűrések alatt tudunk az oldalak között lépkedni. A négy gomb balról jobbra: Első oldal, Előző oldal, Következő oldal, Utolsó oldal. A gombok között láthatjuk az aktuális oldalt és az összes oldalt.

2. Webshop



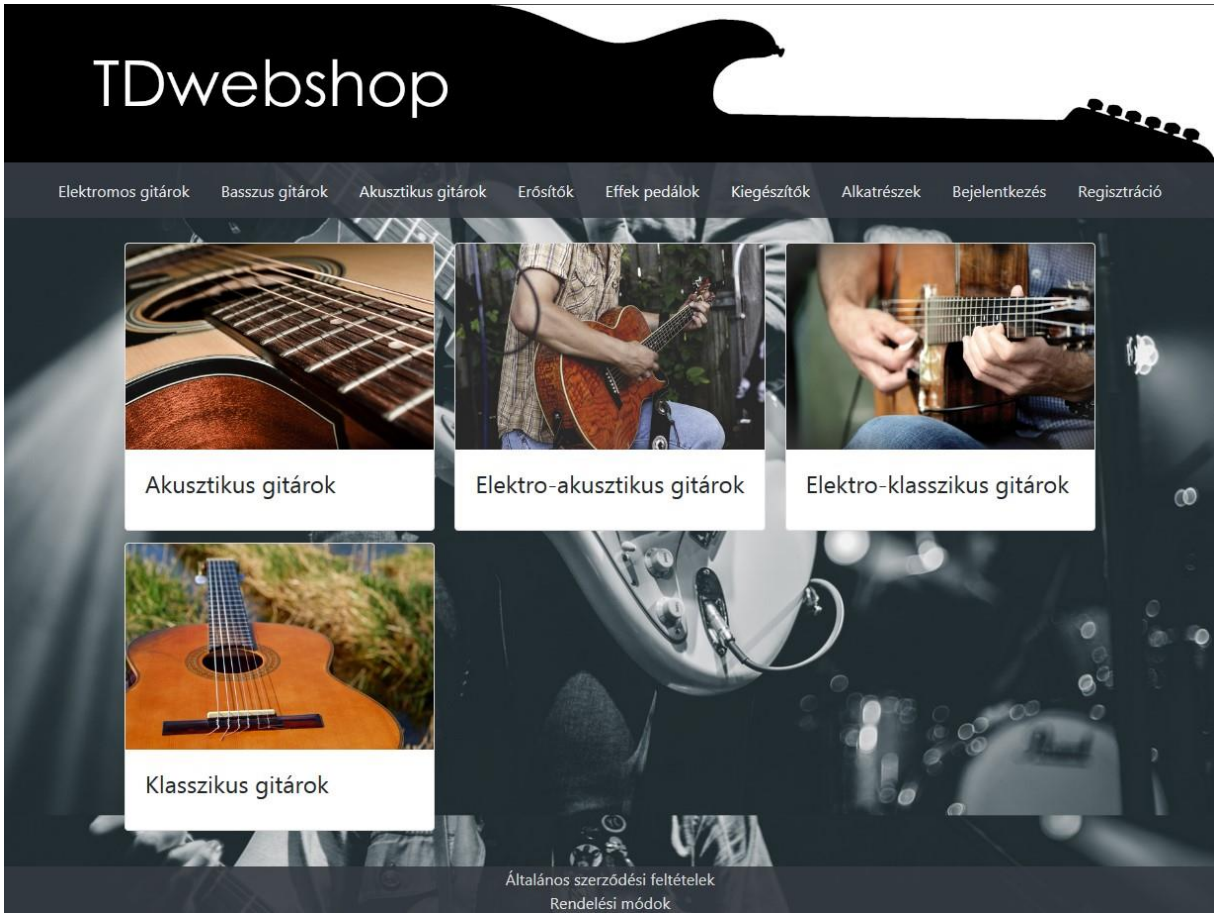
A fenti képen a webshop fő oldala látható.

Az egész webshop megjelenését 4 részre oszthatjuk. Legfelül található a webshop logója. Bárhol vagyunk az oldalon, erre a logóra kattintva mindig a fő oldalara fogunk jutni.

Alatta helyezkedik el a menüsáv. Itt tudjuk kiválasztani termékek fő kategóriáit, továbbá itt van lehetőség bejelentkezni és regisztrálni is. Bejelentkezés után itt érhető el a kosár és a kijelentkezés.

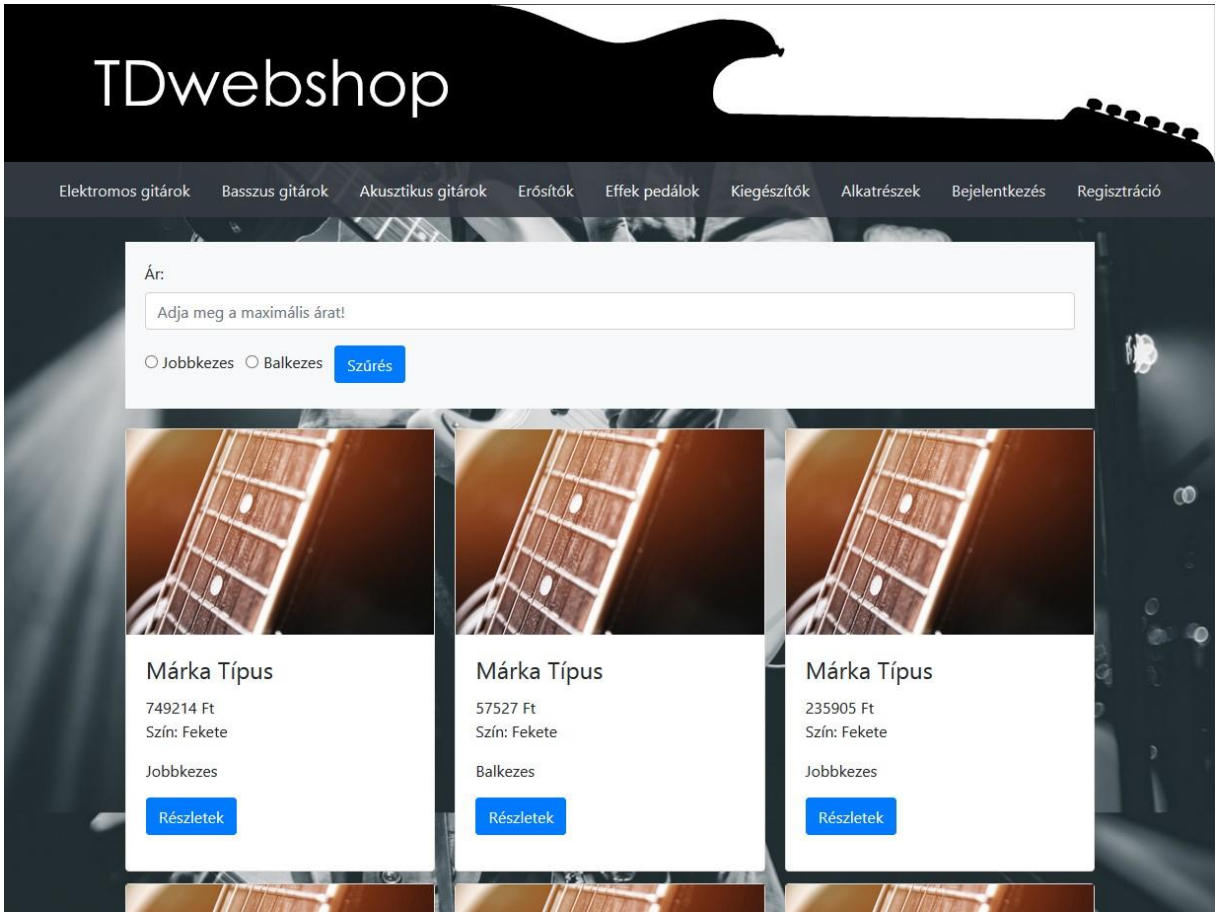
Ez alatt található a legnagyobb rész, amelyben a tartalmak jelennek meg. Ha a fő oldalon vagyunk, akkor itt a hírek jelennek meg egy slideshow-ban.

Ez alatt található a footer. Ebben található minden olyan információnak a linkje, amelyet fontos lehet elolvasni. (Felülről lefelé: Általános szerződési feltételek, Rendelési módok, Szállítási feltételek, Fizetési feltételek, Garanciális feltételek, Reklamációs feltételek, Vásárlástól való elállás, Adatkezelési nyilatkozat, Elérhetőségek.)



A fő menüben valamelyik fő kategóriára kattintva megjelennek az ahhoz tartozó alkategóriák.

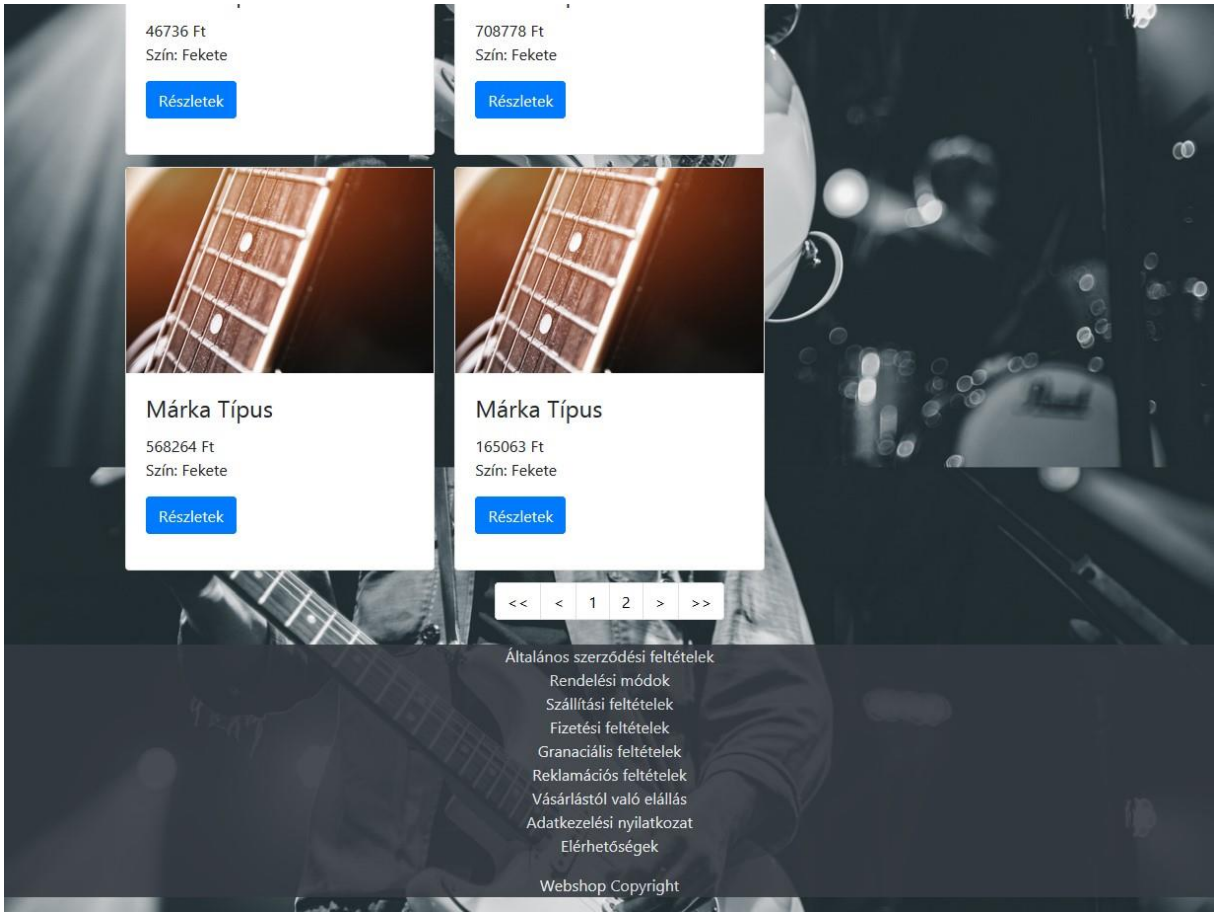
Ebben az esetben az *Akusztikus gitárok* fő kategóriát választottuk ki. A megjelent alkategóriákra rá lehet kattintani, ez után jelennek meg a termékek.



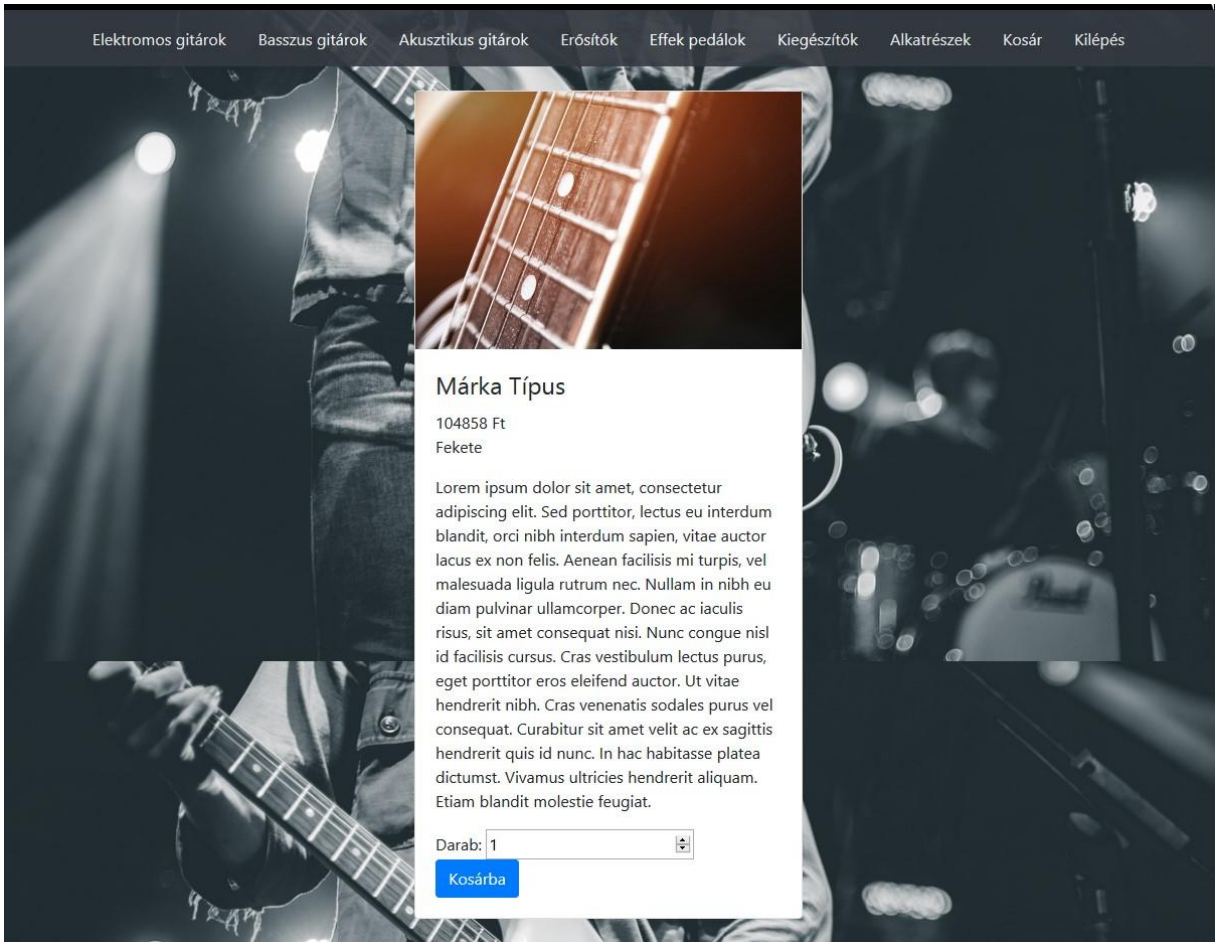
Valamely alkategóriára való kattintás után jelennek meg a termékek, és felettük szűrési lehetőségek. Ha a kiválasztott alkategória nem valamilyen gitár, akkor csak árra tudunk szűrni, mivel ebben az esetben a jobbkezesség nem releváns.

A szűrés gombra kattintva tudjuk aktiválni a beállított szűréseket. Természetesen csak azt a szűrést veszi figyelembe, amelyet beállítottuk.

Minden egyes terméknél lehetőségünk van a részleteket megtekinteni.

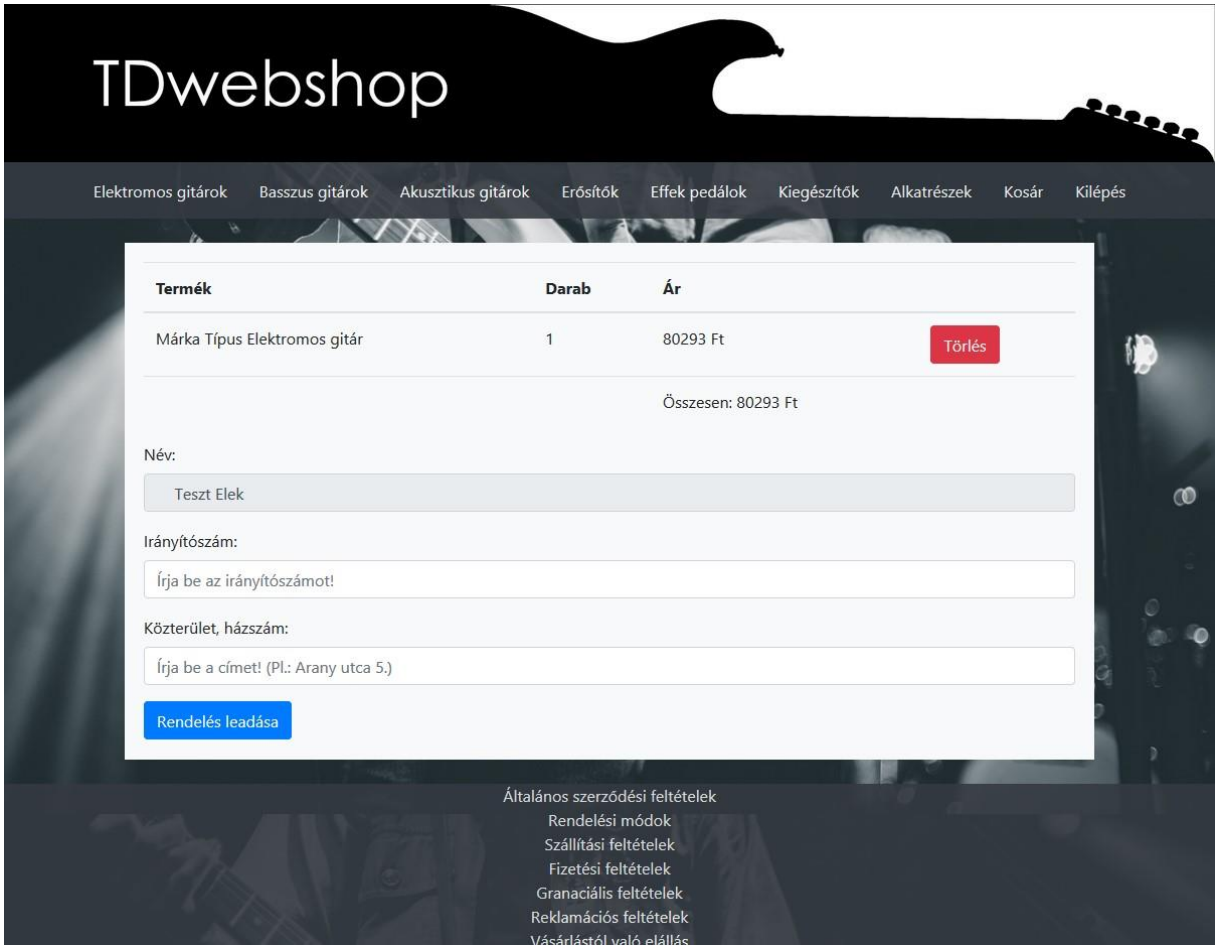


A termékek böngészése közben az oldal aljára görgetésnél találjuk meg az oldalak közötti lépegetési lehetőséget.



Ha egy terméknél rákattintunk a *Részletek* gombra, akkor megjelennek a termék részletei.

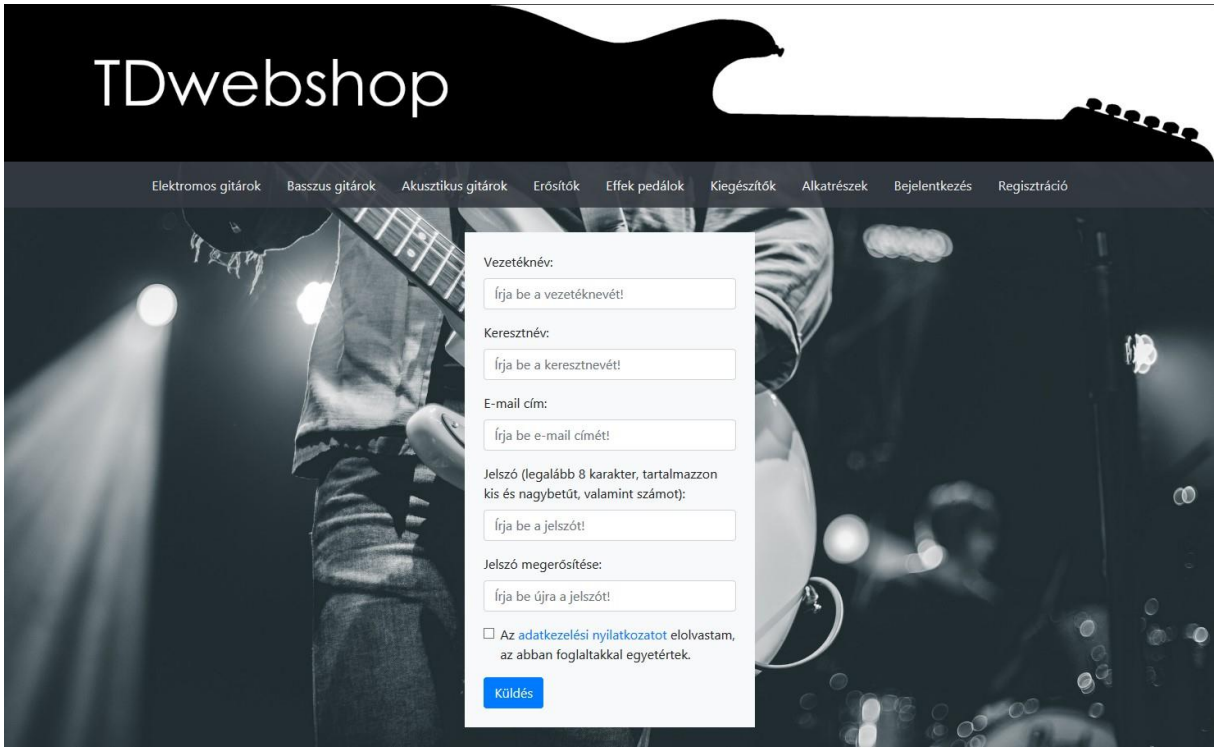
Itt van lehetőségünk az adott terméket hozzáadni a kosárhoz. Ezt úgy tehetjük meg, hogy a *Kosárba* gombra kattintunk. A gomb felett lehetőségünk van kiválasztani, hogy az adott termékből hány darabot szeretnénk rendelni.



Ha be vagyunk jelentkezve, akkor a menüsávban a *Kosár* menüpontra kattintva tudjuk a kosarunkat megjeleníteni.

Ha nem üres a kosarunk, akkor egy táblázatban megjelennek a kosárban lévő termékek, rendelni kívánt darabszámuk, egységáruk és a fizetendő összeg. Lehetőségünk van továbbá termékeket egyenként törölni a kosárból.

A táblázat alatt közvetlenül egy űrlap is megjelenik. Itt tudjuk megrendelni a termékeket. A név mező automatikusan kitöltésre kerül az éppen bejelentkezett felhasználó nevével. Az irányítószám és cím megadása után tudjuk leadni a rendelésünket a *Rendelés leadása* gombra kattintva.

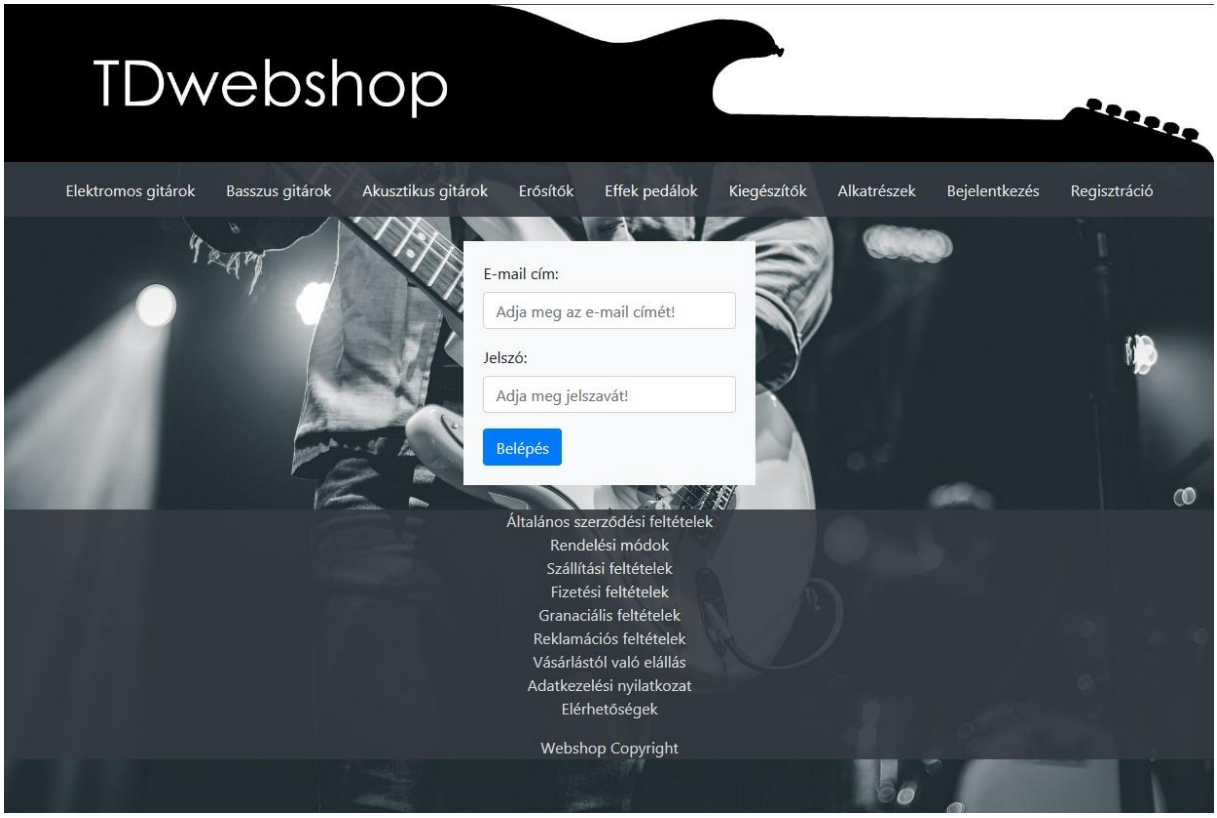


A fent látható kép a regisztrációs űrlapot ábrázolja. Úgy gondolom, ez nem szorul részletes magyarázatra.

Az itt látható összes mező megadása kötelező. Ha valamelyik adatot nem megfelelően adtuk meg vagy üresen hagytuk, arra a weboldal figyelmeztetni fog.

Továbbá lehetőségünk van elolvasni az adatkezelési nyilatkozatot regisztráció előtt.

A *Küldés* gombra való kattintással tudjuk a regisztrációs folyamatot befejezni. Nem megfelelően megadott adat esetén a weboldal figyelmeztet és nem enged regisztrálni.



A fent látható képen a bejelentkezés űrlapot láthatjuk. Úgy gondolom, ez a folyamat sem kíván részletes magyarázatot.

Egy előzőleg regisztrációnál megadott e-mail címmel és jelszóval tudunk belépni. E két adat megadása után a *Belépés* gombra kattintva tudunk belépni. Nem megfelelően megadott adat esetén a weboldal visszajelzést ad és nem enged belépni. Belépés után lesz elérhető a termék kosárba rakása és maga a kosár funkció. Belépve a menüsáv *Kilépés* menüpontjára kattintva tudunk kijelentkezni.

Összegzés

A záródolgozatom megvalósításának végére érve rengeteg új tapasztalatot szereztem és számtalan új dolgot tanultam. A kezdeti terveket, a funkcionális és minőségi követelményeket átnézve rengeteg dolgot sikerült megvalósítanom, azonban tökéletes alkalmazás nem létezik és ez a záródolgozatomra hatványozottan igaz.

Úgy gondolom, a jövőben rengeteg fejlesztést és új dolgot kell megvalósítanom ahhoz, hogy a rendszer akár „élesben” is jól és hatékonyan használható legyen. Nagyon fontos megemlítenem, hogy a képzés befejeztével mindenképp szeretném elvégezni az alább megemlített fejlesztéseket.

A jövőben az asztali alkalmazásomban elvégzendő fejlesztések

A program élesben való működéséhez elengedhetetlen lenne az egyszerűbb és jóval gyorsabb vonalkódos termékbevitel. Jelenlegi állapotban kézzel kell minden termék adatát megadni, amely egy bizonyos számú termék felett nagyon időigényes és nehézkes. Ezen funkció kihagyásának oka legfőképp az eszközhiány és a nem megfelelő tudás szint.

A rendelések részleteit (details oszlop) a jelenlegi állapotban csak akkor tudjuk megtekinteni, ha az egeret a megfelelő cellán tartjuk egy bizonyos ideig, ekkor egy szövegbuborékban jelenik meg a teljes érték. Ennél elegánsabb és jobb lenne, ha a megadott rendelésre kattintva egy külön ablak jelenne meg, amely megjeleníti a rendelés minden egyes részletét és adatát. Továbbá, szükség lehet az itt megjelenő részletek és adatok fizikai, papír alapú formájára. Ez esetben a rendelések adatait ki lehetne nyomtatni, amelyet például a megrendelőnek, az előkészített csomaghoz lehetne mellékelni.

Szerettem még volna látássérültek számára akadálymentesített verziót is készíteni, amely az időhiány miatt nem tudott megvalósulni.

Legutolsó sorban szeretném megemlíteni a szebb dizájnt is. Jelenlegi állapotában a programom nagyon egyhangú, unalmas és idejétmúlt hatást kelt. Habár ez az alkalmazás használhatóságát nem befolyásolja, én úgy gondolom egy szebb és modernebb kinézettel sokat lehet javítani a felhasználói élményen.

A jövőben a webshopon elvégzendő fejlesztések

Legelsőként az online fizetési lehetőséget és a fizetési mód kiválasztását szeretném megemlíteni. Úgy gondolom manapság egy webshop nem lehet versenyképes, ha ezt a funkciót nem tudja. Jelenlegi állapotban csak utánvétes rendelésre van lehetőség. Ezen funkció kihagyásának oka az időhiánynak tudható be.

A termék megjelenítésnél sorba rendezést és több szűrési lehetőséget is szerettem volna megadni. A kihagyás annak köszönhető, hogy ez a jelen esetben egy bizonyos számú szűrési és sorba rendezési lehetőség után nagyon nehézkesen kivitelezhető és nem feltétlenül létfontosságú, -ráadásul találkoztam „élesben” működő webshopokkal, amelyek ennél is kevesebb szűrési lehetőséget biztosítanak-, ezért úgy gondoltam célszerűbb fontosabb funkciók megvalósításával foglalkoznom.

Itt is megoldható az akadálymentesített megjelenés. Ez a funkció is az időhiány miatt került kihagyásra.

Hallgatói nyilatkozat

Alulírott

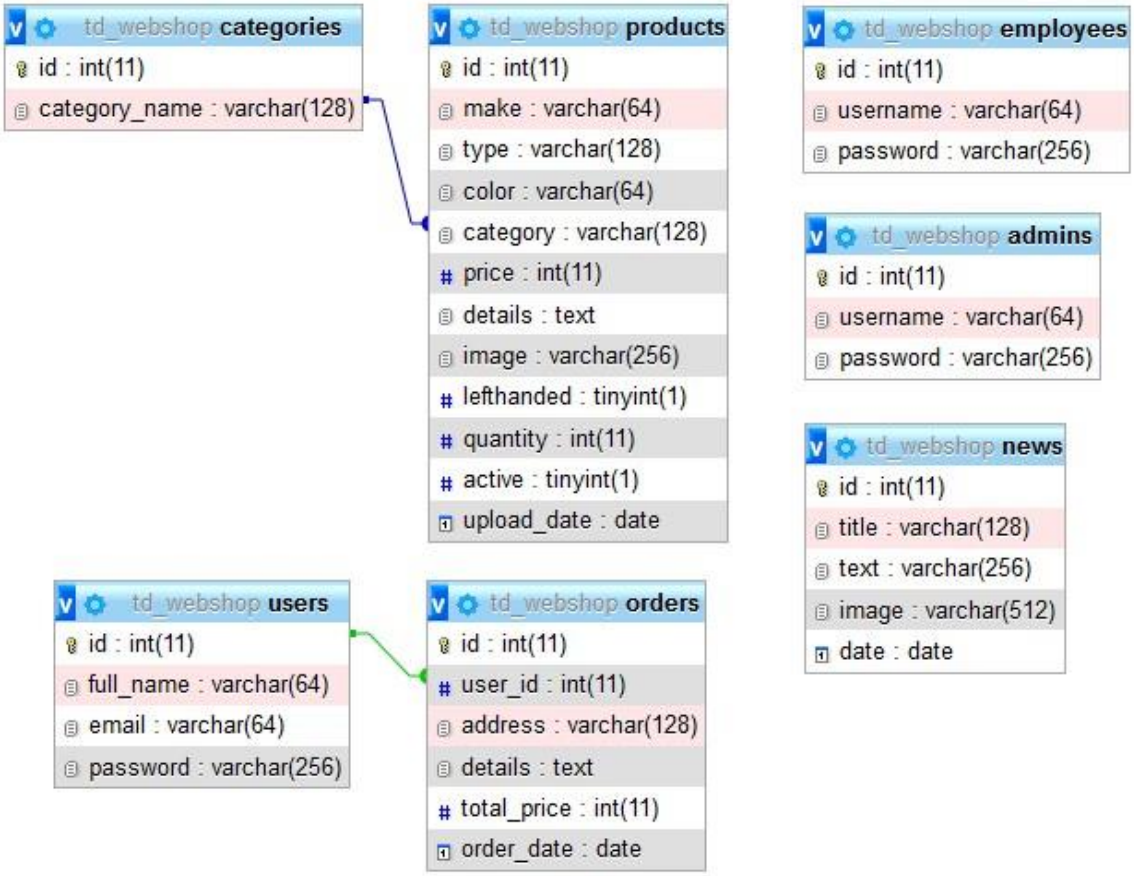
Túri Dániel a Szegedi Gazdasági Szakképző Iskola Vasvári Pál Tagintézménye hallgatója kijelentem, hogy „A TDwebshop és TDmanage alkalmazások bemutatása” című záródolgozat a saját munkám.

Kelt: 2020-04-01

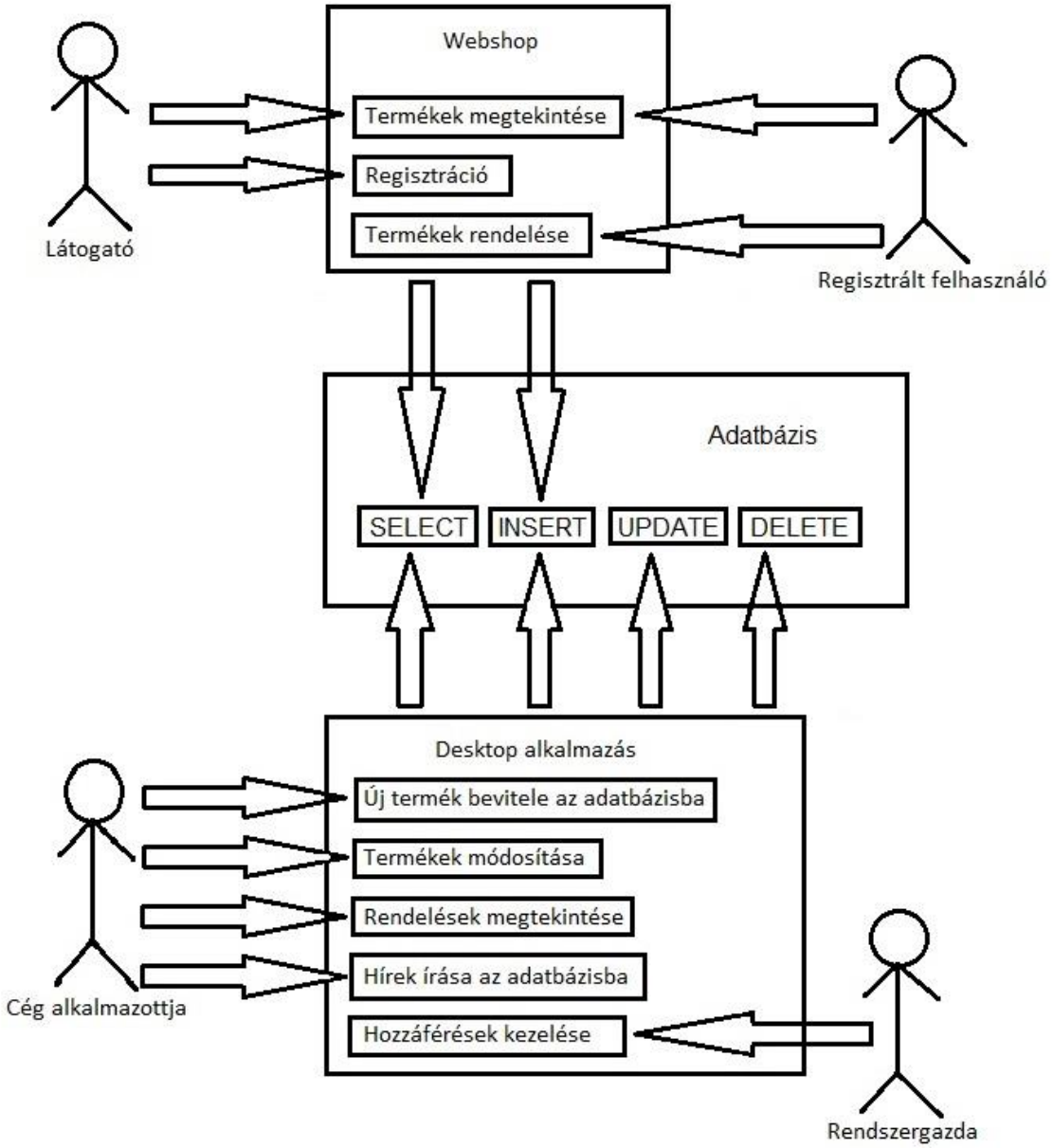
_____

aláírás

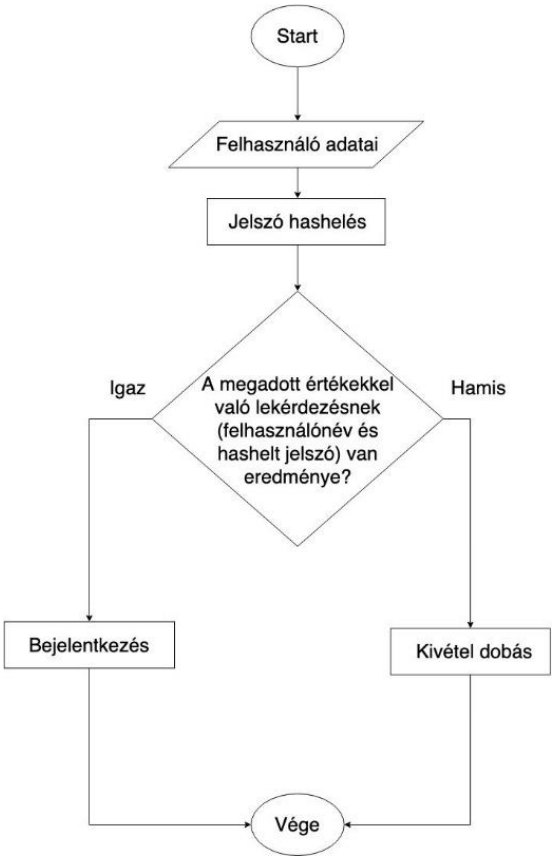
Melléklet



1. ábra: Adatbázis kapcsolatai

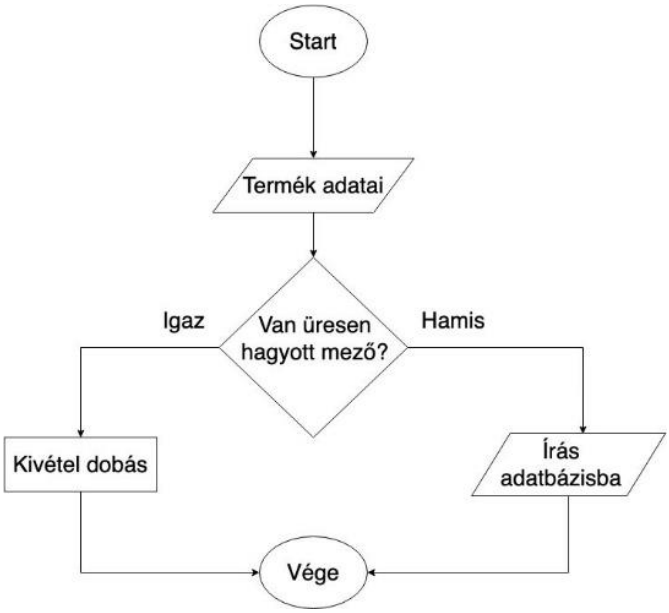


2. ábra: Az egész rendszer use-case diagrammja.

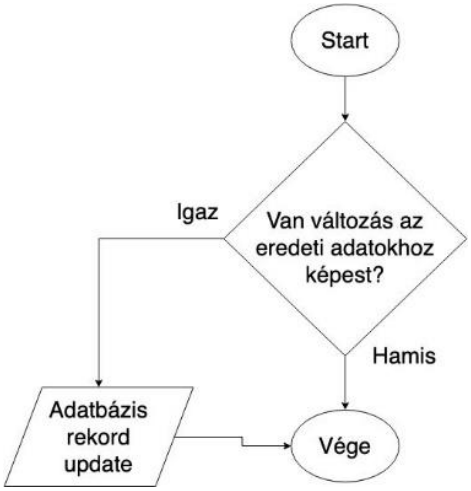


3. ábra: Bejelentkezés (asztali alkalmazás)

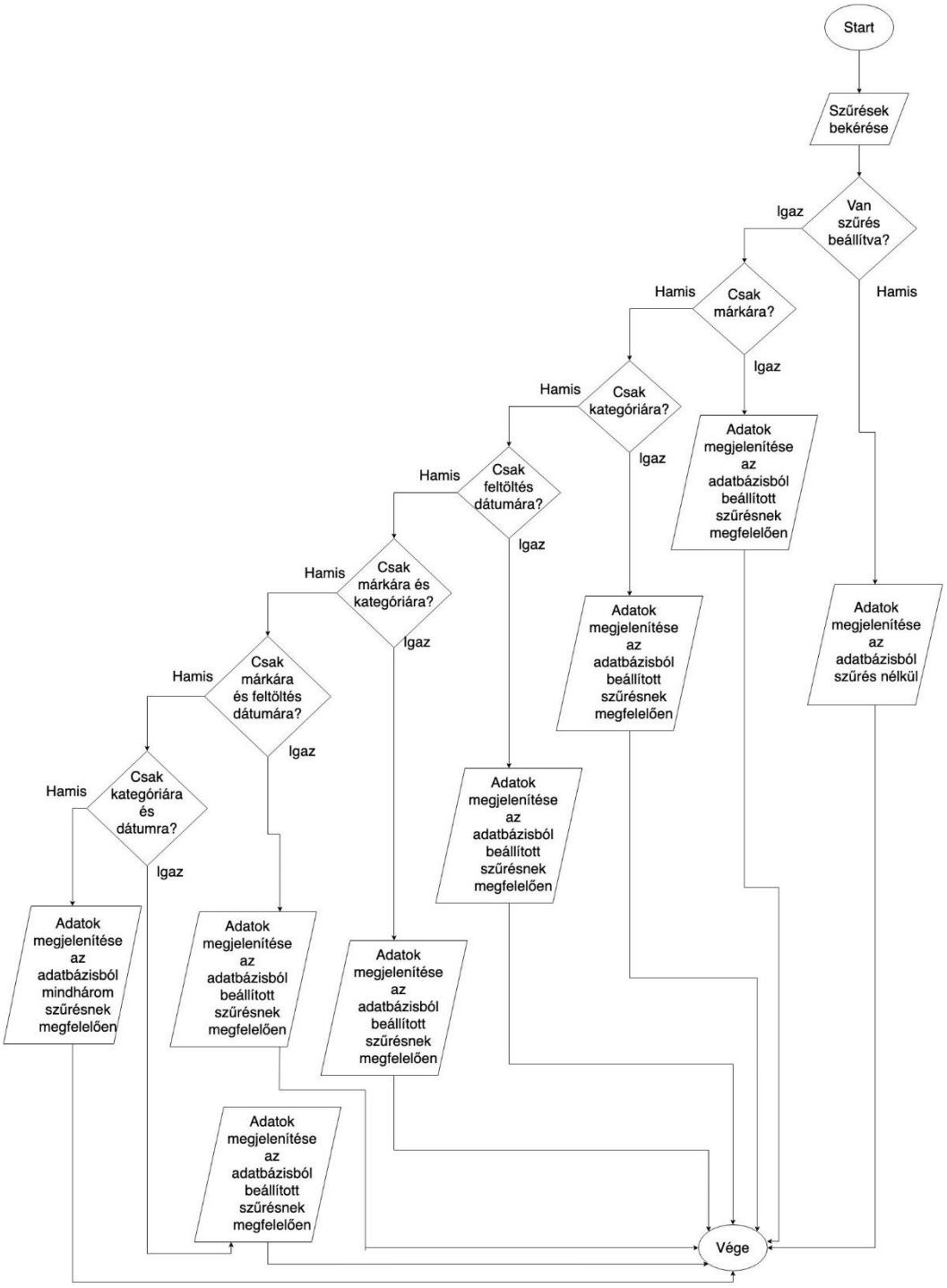
4. ábra: Új dolgozó és adminisztrátor hozzáadása



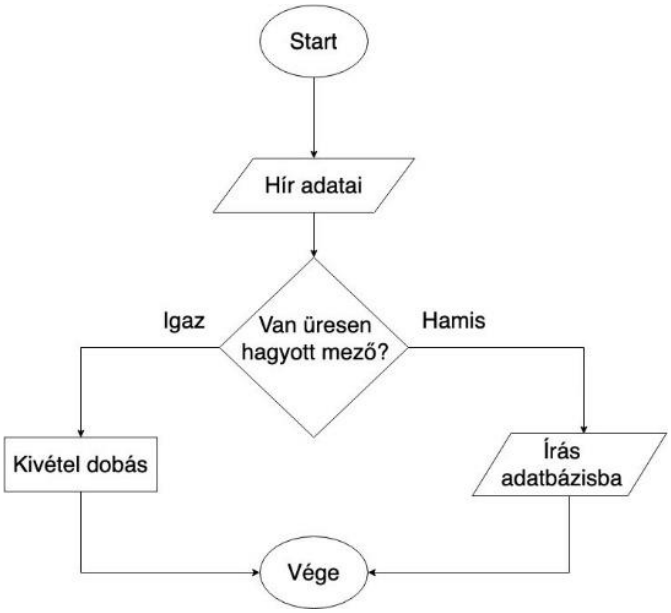
5. ábra: Új termék feltöltése az adatbázisba



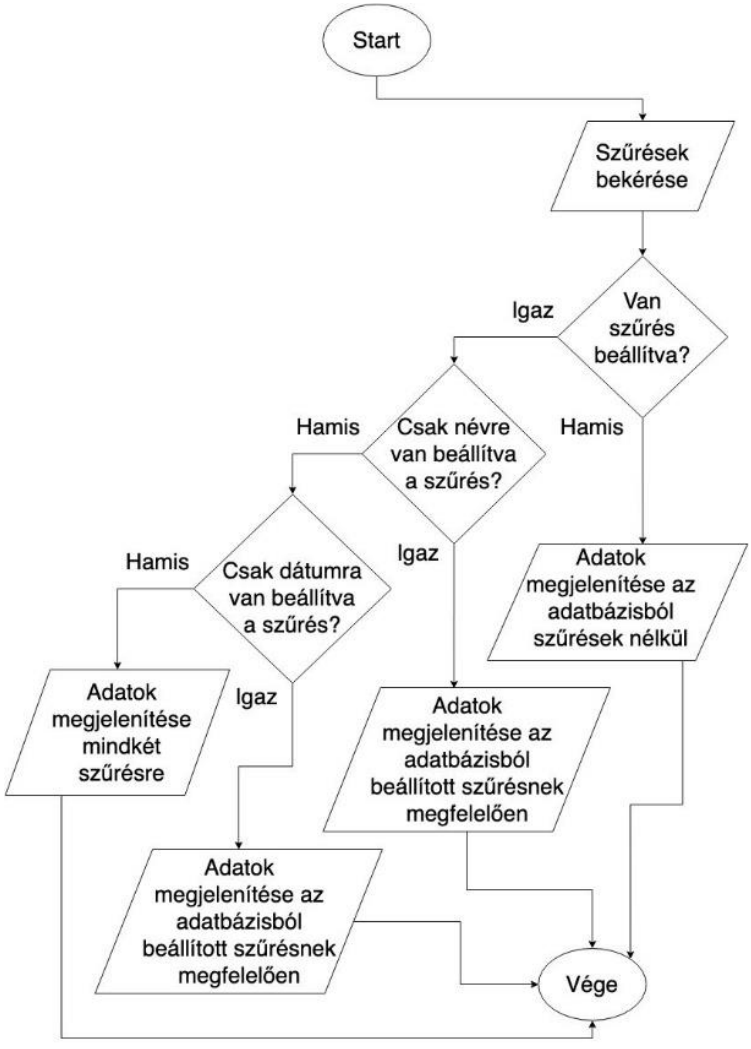
6. ábra: Termék módosítás



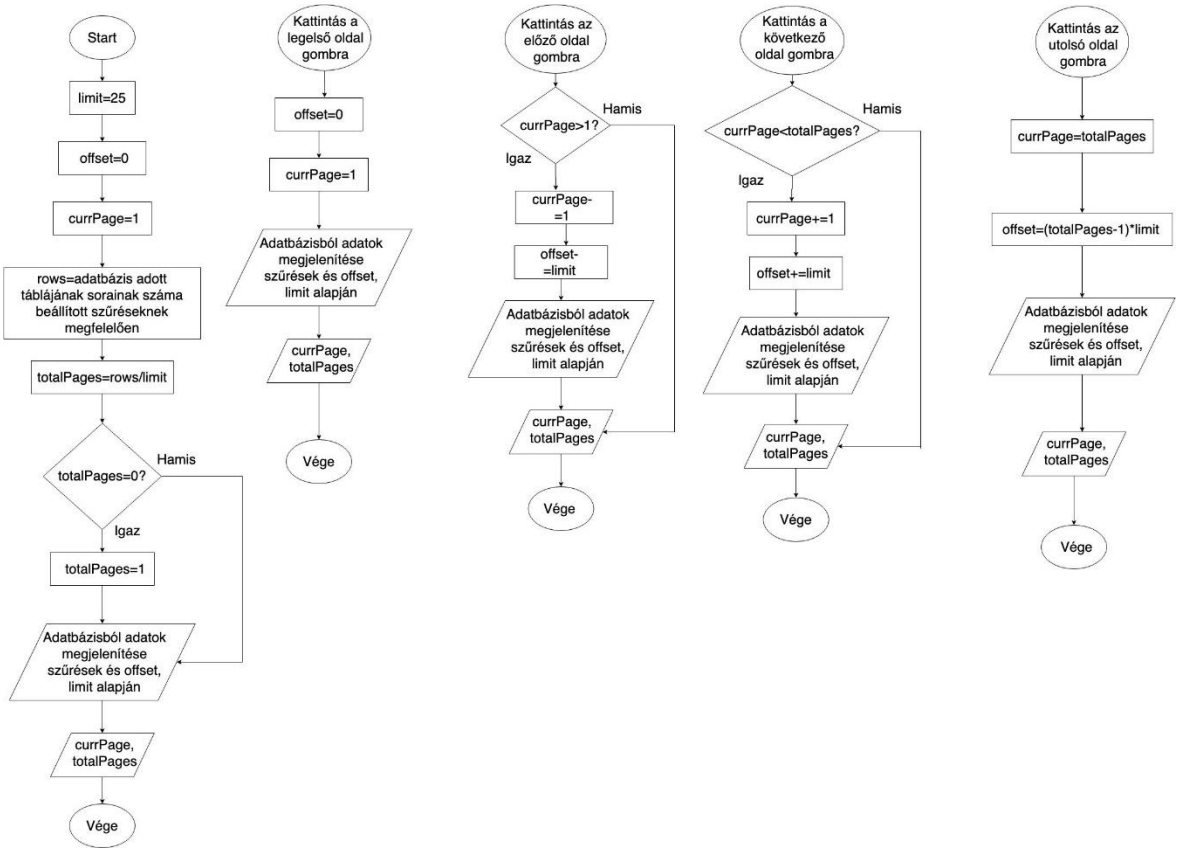
7. ábra: Termékek megjelenítése (asztali alkalmazás)



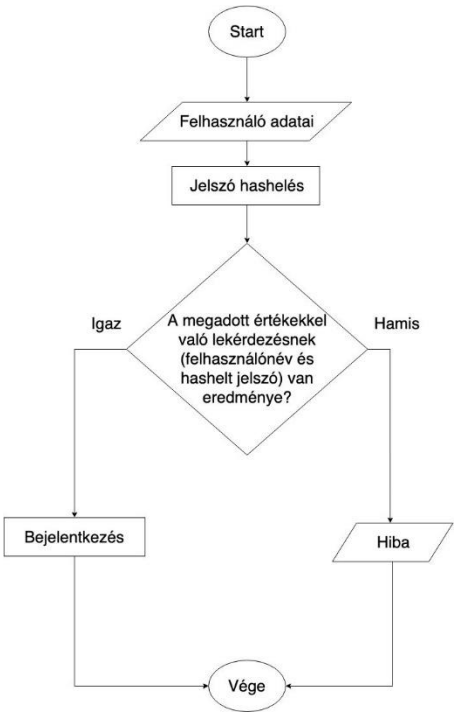
8. ábra: Új hír írása az adatbázisba



9. ábra: Rendelések megtekintése



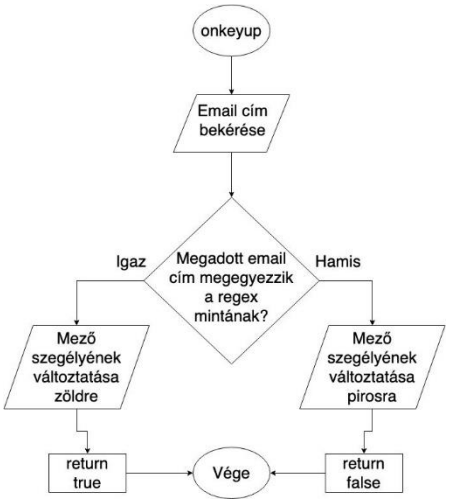
10. ábra: Oldalakra tördelés és oldalak közötti lépés



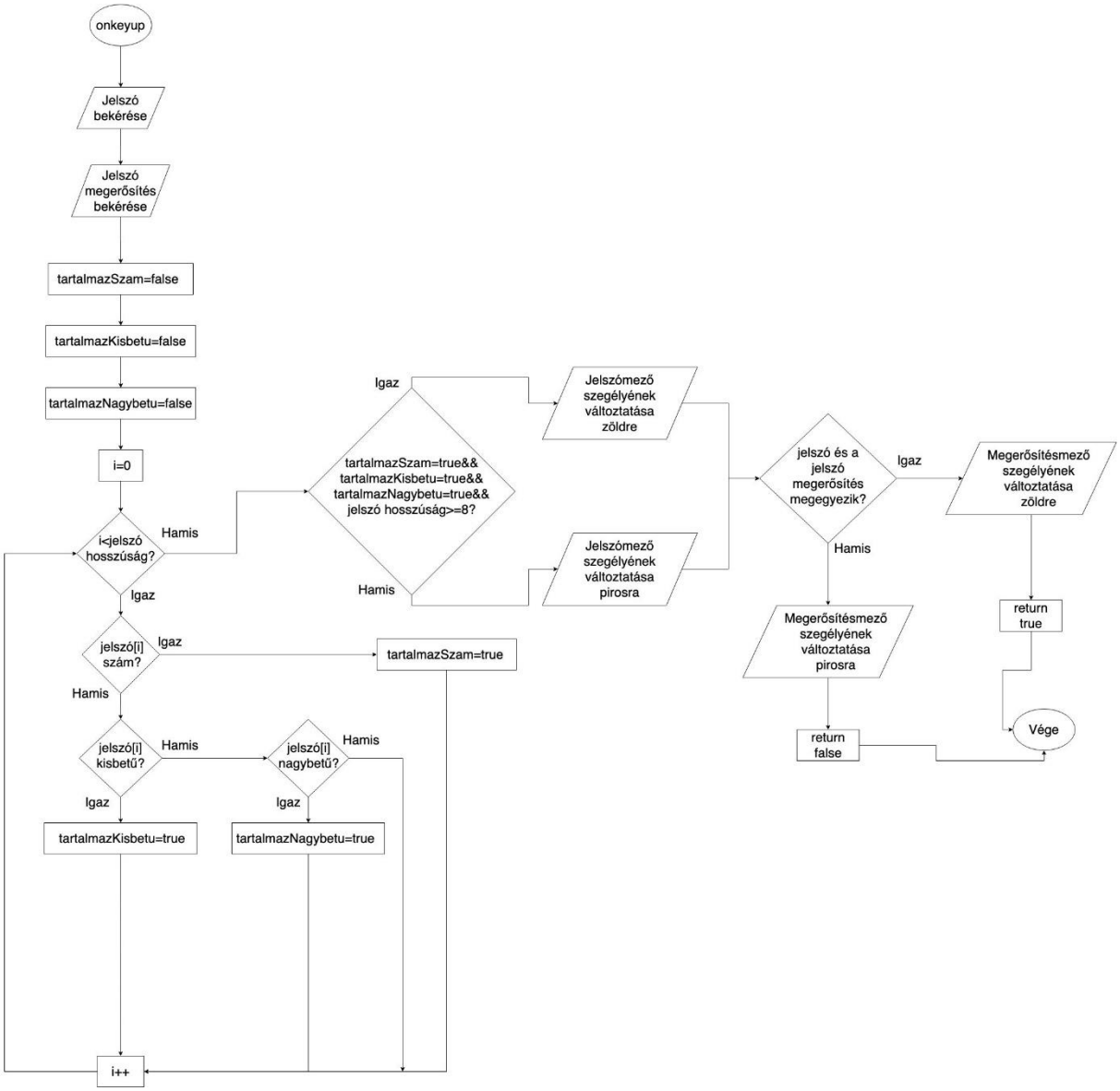
11. ábra: Bejelentkezés (webshop)

12. ábra: Vezetéknév hitelesítése

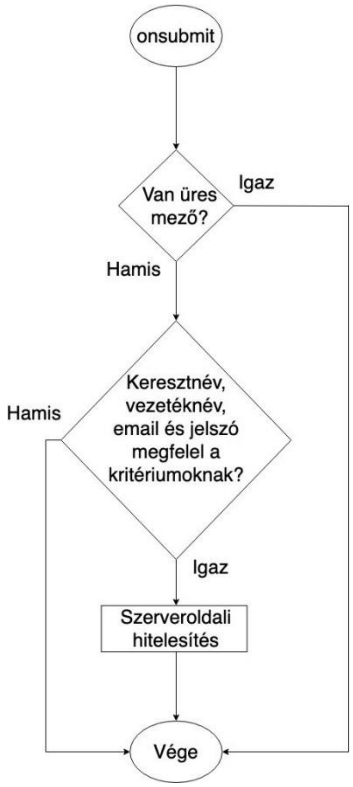
13. ábra: Keresztnév hitelesítése



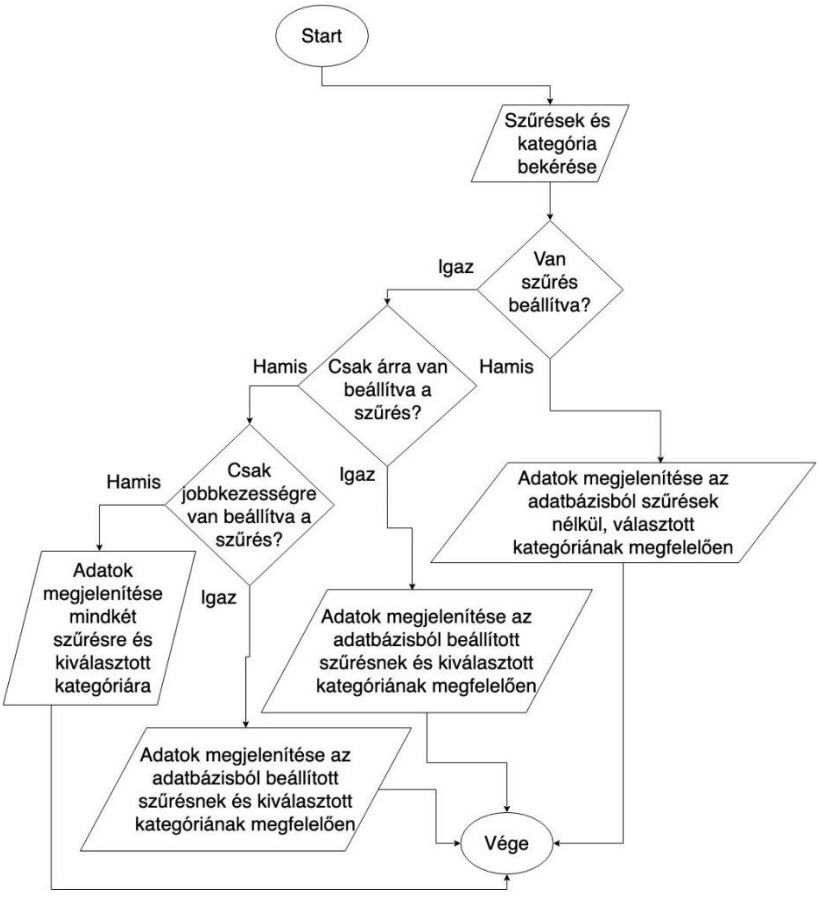
14. ábra: E-mail cím hitelesítése



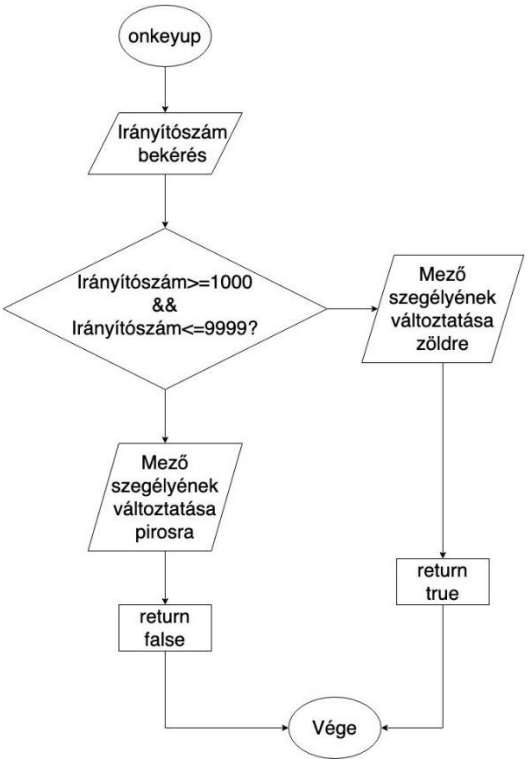
15. ábra: Jelszó hitelesítése



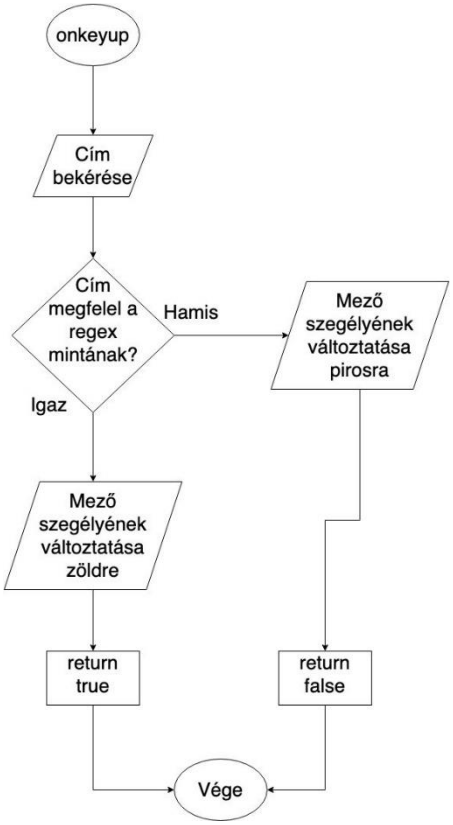
16. ábra: A regisztrációs űrlap hitelesítése



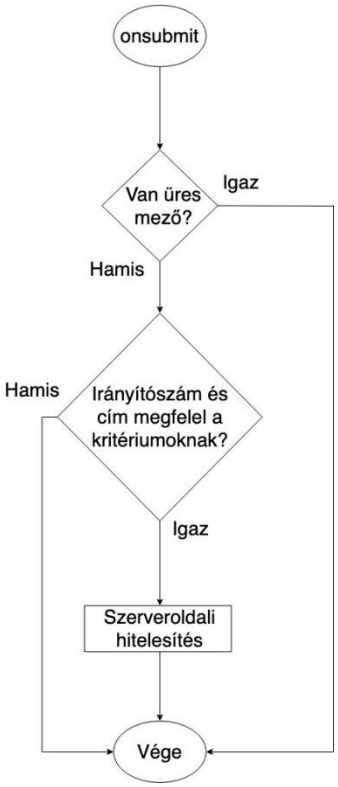
17. ábra: Termékmegjelenítés (webshop)



18. ábra: Irányítószám hitelesítés



19. ábra: Szállítási cím hitelesítés



20. ábra: Kosárhitelesítés