

**VÁCI SZAKKÉPZÉSI CENTRUM
BORONKAY GYÖRGY MŰSZAKI
SZAKKÖZÉPISKOLÁJA ÉS GIMNÁZIUMA**



SZAKDOLGOZAT

StorageManager, Raktár kezelés

Konzulens: Kemenes Tamás
Wiezl Csaba

Készítette: Fodor Soma, 14.P

Hallgatói nyilatkozat

Alulírott, ezúton kijelentem, hogy a szakdolgozat saját, önálló munkám, és korábban még sehol nem került publikálásra.

Fodor Soma

Konzultációs lap

Vizsgáló neve: Fodor Soma

Szakdolgozat címe: StorageManager

Program nyújtotta szolgáltatások:

- Termék adatok lekérdezése adatbázisból különböző szempontok alapján
- Új adatok bevitele az adatbázisba
- Az adatbázisban már meglévő termékek módosítása
- Termékek törlése az adatbázisból
- Rendelés leadása termékekre
- Minimum készlet kezelése

Sorszám	A konzultáció időpontja	A konzulens aláírása
1.	2015.10.10.	
2.	2015.11.14.	
3	2015.12.05.	
4.	2016.01.11.	
5.	2016.01.25.	
6.	2016.02.08.	

A szakdolgozat beadható:

A szakdolgozatot átvettem:

Vác, 2016.

Vác, 2016.

.....

.....

Konzulens

A szakképzést folytató intézmény
tanfolyamfelelőse

Tartalomjegyzék

Hallgatói nyilatkozat	2
Konzultációs lap	3
Tartalomjegyzék	4
1 Fejlesztői dokumentáció	6
1.1 A program feladata, bemutatása	6
1.2 A feladat választásának indoklása	6
1.3 Feladat specifikáció	6
1.3.1 A feladat, probléma meghatározása	6
1.3.2 A program nyújtotta szolgáltatások	7
1.3.3 Felhasználói szerepkörök	8
1.4 Fejlesztői környezet	10
1.4.1 Definíciója	10
1.4.2 C# 4.0	11
1.4.3 A .Net 4.0	12
1.4.4 MySQL 5.6.17	14
1.5 A fejlesztés során használt programok	14
1.5.1 Microsoft Visual C# 2010 Express	14
1.5.2 WampServer 2.5	14
1.5.3 phpMyAdmin	14
1.5.4 MySQL Connector Net 6.9.8	15
1.6 Adatszerkezet (Adatbázis és Adattáblák)	15
1.6.1 Választott adatbázis szerver (MySQL)	15
1.6.2 Adatbázis létrehozása	15
1.6.3 Normalizás	15
1.6.4 Adattáblák	17
1.6.5 raktár tábla	17
1.6.6 gyarto tábla	18
1.6.7 típus tábla	19
1.6.8 állapot tábla	20
1.6.9 rendeles tábla	21
1.6.10 afakulcsok tábla	22
1.6.11 users tábla	23
1.6.12 usersperm tábla	24
1.6.13 kosar tábla	25
1.6.14 hibak tábla	26
1.6.15 hiba_allapot tábla	27
1.6.16 Táblák kapcsolati rendszerterve	28
1.7 Osztályok	29
1.7.1 MySQL kapcsolatért felelős osztály (MySQLcon.cs)	29
1.7.2 Permission osztály (Permission.cs)	31
1.7.3 Bejelentkezés (Form2.cs)	33
1.7.4 Raktár (Form1)	35
1.7.5 Adduser (Adduser.cs)	40
1.7.6 Beállítások(Beallitasok.cs)	42
1.8 Együttműködési diagramm	44
1.9 Konklúzió	45
2 Felhasználói dokumentáció	46
2.1 Általános ismertető	46

2.2	Telepítés és rendszerkövetelmény	46
2.3	Bejelentkezés	49
2.3.1	Hibaüzenet az induláskor	50
2.4	A program használta (Alap felhasználó)	51
2.5	Program használata (Adatbázis adminisztrátorként)	52
2.6	Program használata (Adatbázis rendszergazda)	53
2.7	Irodalomjegyzék	57
3	Mellékletek.....	58

1 Fejlesztői dokumentáció

1.1 A program feladata, bemutatása

A vizsgamunkának beadott program mely saját ötleten alapul egy számítógépes üzlet raktáron lévő termékeinek nyilvántartására (értem ezt úgy, hogy az üzletben elérhető). A program segítségével a raktáron lévő termékek (elérhető termékek) között hatékonyan és könnyedén lehet keresni több különböző szempont alapján (pl.: gyártó, terméktípus vagy a termék neve alapján). A találatok egy jól áttekinthető listában jelennek meg. Túl nagy találati lista esetén lehetőség van annak szűkítésére egy részletesebb keresésre, melyre a program szintén lehetőséget ad. Ezen felül lehetőség van új áruk, típusok, gyártók, állapotok felvételére, törlésre, módosítására. Valamint minden termék esetén be lehet állítani egy minimum készletet is. Ha ezt eléri a raktáron lévő áru mennyisége, akkor azt jelzi a felhasználónak. Megfelelő jogosultsággal a felhasználó rendelést adhat le a termékekre.

Az adatbázis egy központi szerveren fut és a programot több helyi gépre feltelepítve egyszerre párhuzamosan többben is kereshetnek az adatbázisban vagy módosításokat hajthat végre az adatbázis adminisztrátor.

Ha a felhasználó hibát talál az adatbázisban (elírás, rosszul beállított gyártó, állapot változás stb.) akkor azt jelezheti az adatbázis adminisztrátora felé vagy az ebben illetékes személynek. Ezenfelül ha egy termékkel probléma van, akkor a gyártó támogató oldala felé is küldhet e-mailt.

1.2 A feladat választásának indoklása

Ezen szakdolgozat téma (program) kiválasztása és megírása közben az a cél vezérelt, hogy az elmúlt 1,5 év alatt szerzett programozói tudásom bemutassam, összefoglaljam egy általam választott fejlesztő környezetben és egy általam választott feladat segítségével.

1.3 Feladat specifikáció

1.3.1 A feladat, probléma meghatározása

A feladat egy olyan program, mely a sikeres szakdolgozat eléréséhez támasztott tartalmi és feladat specifikációknak eleget tesz. Bővebben legalább 4-5 adattábla legalább 3NF-re (harmadik normálformára) hozva és egy lehetőleg több felhasználós program, amely

ezt az adatbázist kezeli, lekérdezéseket, kereséseket hajt benne végre és a felhasználó számára jó áttekinthető formában visszaadja a lekérdezések eredményét.

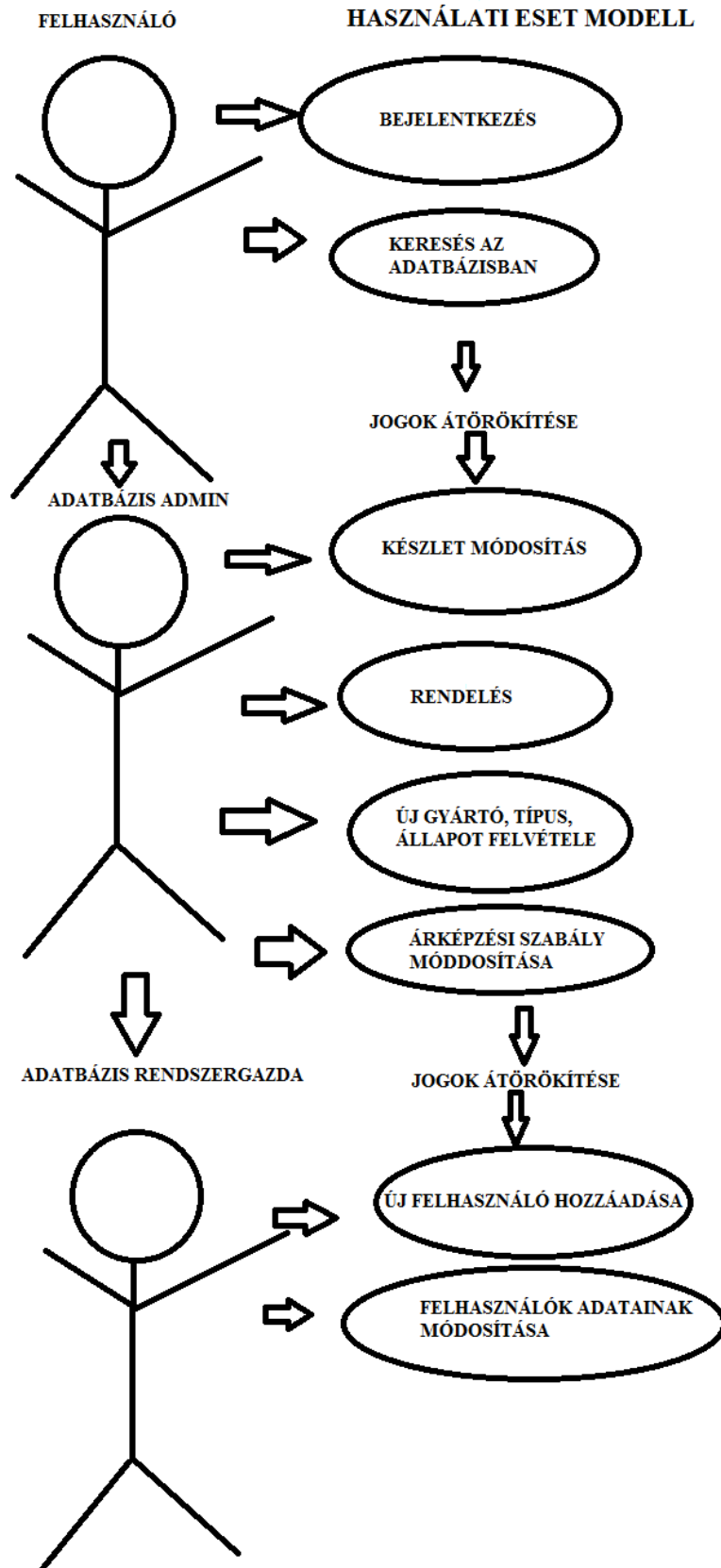
1.3.2 A program nyújtotta szolgáltatások

- Regisztrált felhasználó beléptetése
- Új felhasználó regisztrálása jogokkal (megfelelő jog szükségeltetik a művelet végrehajtásához)
- Felhasználó:
 - jogainak módosítása
 - jelszavának módosítása
 - utolsó bejelentkezési idejének tárolása
 - regisztrálási idejének tárolása
 - törlése
- Keresés:
 - az adatbázisban szereplő termékek között
 - szűkítése
 - termék neve alapján
 - termék típus alapján
 - termék gyártója alapján
 - meghatározott ár intervallumban
- Rendelés leadása az adatbázisban szereplő termékekre vagy új termékekre
- Új termék felvétele a raktárnyilvántartásba
- Raktáron lévő termékek módosítása
- Figyelmeztetés minimum készletre
- Kapcsolat az adatbázis adminisztrátorral, üzletvezetővel a e-mail-en keresztül a program segítségével
- A találati lista háttér színének megváltoztatása
- A program által használt betűtípus és méret megváltoztatása
- Kilépés a programból

1.3.3 Felhasználói szerepkörök

Nem bejelentkezett felhasználó: nincsen semmilyen joga

- Bejelentkezett felhasználó:
- Alap felhasználó:
 - Keresés az adatbázisban
 - Észlelt hiba jelentése
 - Hiba lista megtekintése
- Adatbázis adminisztrátor:
 - Új áru felvétele
 - Adatok módosítása
 - Rendelés szerkesztése és leadása
 - Új gyártó felvétele az adatbázisba
 - Új termék típus felvétele az adatbázisba
 - Új termék állapot felvétele az adatbázisba
 - Termékek törlése
 - Áfa kulcs módosítása
- Adatbázis rendszergazda:
 - Új felhasználók felvétele
 - Felhasználók adatainak és jogainak módosítása
 - Felhasználók törlése
 - Az előző szerepkör összes többi joga
 - Feldolgozza és javítja a hibát



1.4 Fejlesztői környezet

1.4.1 Definíciója

A fejlesztői környezet olyan programozási eszközök, könyvtárak és beállítások csoportja, melyekkel a szoftverfejlesztés során a felhasznált programozási nyelven vagy nyelveken létrehozott forráskódokat futás kész állapotba lehet hozni és azt tesztelni. Ez jelentheti a fordítást, vagy nem önálló programok futtatási környezetbe helyezését. Mindkét esetben a környezet általában tartalmazza a futtatókörnyezetet is a tesztelés miatt.

Sok esetben a felhasznált programozási nyelv rendelkezik egy olyan csomaggal, melyet telepítve a fejlesztői gépen ez a környezet automatikusan létrejön.

Sok (általában grafikus felületre szánt programok írására alkalmas) nyelv rendelkezik saját forráskód szerkesztővel „egybeépített” fordítóval, ezt hívják integrált fejlesztői környezetnek.

A fejlesztői környezet nagyobb méretű, szerteágazóbb programok esetében nem merül ki a fent említett csomagok összességében. A fejlesztés során rengeteg felhasznált programkönyvtár, segédprogram, esetleg a fejlesztési lépések vagy a tesztelés segítésére szolgáló rövid egyedi segédprogramok, illetve egy szerteágazó jegyzékrendszer („mappák” adott szerkezete) alkotja.

A fejlesztői környezet csak a fejlesztést végző személy(ek) által használt számítógép(ek)en szükségesek. A kész program csak a futtatókörnyezetet igényli.

(forrás: https://hu.wikipedia.org/wiki/Fejlesztői_környezet)

1.4.2 C# 4.0

A nyelv története

A C# programozási nyelv a Microsoft új fejlesztési környezetével, a 2002-ben megjelent Visual Studio.NET programcsomaggal, annak részeként jelent meg. Bár a nyelv hosszú múlttal nem rendelkezik, mindenképpen elődjének tekinthetjük a C++ nyelvet, a nyelv szintaktikáját, szerkezeti felépítését. A C, C++ nyelvekben készült alkalmazások elkészítéséhez gyakran hosszabb fejlesztési időre volt szükség, mint más nyelvekben, például a MS Visual Basic esetén. A C, C++ nyelv komplexitása, a fejlesztések hosszabb időciklusa azt eredményezte, hogy a C, C++ programozók olyan nyelvet keressenek, amelyik jobb produktivitást eredményez, ugyanakkor megtartja a C, C++ hatékonyságát. Erre a problémára az ideális megoldás a C# programozási nyelv. A C# egy modern objektumorientált nyelv, kényelmes és gyors lehetőséget biztosítva ahhoz, hogy .NET keretrendszer alá alkalmazásokat készítsünk, legyen az akár számolás, akár kommunikációs alkalmazás. A C# és a .NET keretrendszer alapja a Common Language Infrastructure(CLI).

A nyelv jellemzői

A C# az új .NET keretrendszer bázisnyelve. Tipikusan ehhez a keretrendszerhez tervezték, nem véletlen, hogy a szabványosítási azonosítójuk is csak egy számmal tér el egymástól. A nyelv teljesen komponens orientált. A fejlesztők számára a C++ hatékonyságát, és a Visual Basic fejlesztés gyorsaságát, egyszerűségét ötvözték ebben az eszközben.

A C# nyelv legfontosabb elemei a következők:

- Professzionális, Neumann-elvű. Nagy programok, akár rendszerprogramok írására alkalmas. • A program több fordítási egységből – modulból – vagy fájlból áll. Minden egyes modulnak vagy fájlnek azonos a szerkezete.
- Egy sorba több utasítás is írható. Az utasítások lezáró jele a pontosvessző (;). Minden változót deklarálni kell. Változók, függvények elnevezésében az ékezetes karakterek használhatóak, a kis- és nagybetűk különbözőek.
- A keretrendszer fordítási parancsa parancssorból is egyszerűen használható. (pl. csc/out:alma.exe alma.cs).
- Minden utasítás helyére összetett utasítás (blokk) írható. Az összetett utasítást a kapcsos zárójelek közé {} írt utasítások definiálják.
- Érték (alaptípusok, enum, struct, value) illetve referencia (class) típusú változók. • Nincs mutatóhasználat; biztonságos a vektorhasználat.

- Boxing, unboxing. Minden típus őse az object, így például egy egész típust (int) csomagolhatunk objektumba (boxing) illetve vissza (unboxing).
- Függvények definíciói nem ágyazhatók egymásba, önmagát meghívhatja (rekurzió). Tehát függvénydefiníció esetén nincs blokkstruktúra. Blokkon belül statikus vagy dinamikus élettartamú változók deklarálhatók. Függvénytípuspolimorfizmus megengedett.
- Érték, referencia (ref) és output (out) függvényparaméterek.
- Kezdő paraméter-értékkadás, változó paraméterszámú függvény deklarálása.
- Delegáltak, események használata.
- Hierarchikus névterekben használt osztályok. Mivel minden osztály, ezért a „program”, a Main függvény public static hozzáférésű. Több osztály is tartalmazhat Main függvényt, de ilyen esetben a fordításkor meg kell mondani, hogy melyik osztálybeli Main függvény legyen az aktuális induló (/main:osztálynév).
- Új operátorok: is operátor egy objektum típusát ellenőrzi (x is Labda), as operátor a bal oldali operandust jobb oldali típussá konvertálja (Labda l = x as Labda;). A hagyományos konverziós operátortól abban különbözik, hogy nem generál kivételt!
- Privát konstruktor (nem akarunk egy példányt se), statikus konstruktor (statikus mezők inicializálása, mindig példány konstruktor előtt hívódik meg, futási időben az osztálybetöltő hívja meg) használata.
- Nincs destruktork, helyette a keretrendszer szemétygyűjtési algoritmus van. Szükség esetén az osztály Dispose metódusa újradefiniálható.
- Egyszeres öröklés, interface-ek használata.
- Operátorok definiálásának lehetősége, property, indexer definiálás.
- Kivételkezelés megvalósítása.
- Párhuzamos végrehajtású szálak definiálhatósága.

(forrás: <http://szatyika.hu/files/Programozas-Csharp-nyelven-Konyv.pdf>)

1.4.3 A .Net 4.0

A Visual Studio 6.0 fejlesztőrendszer átdolgozásaként 2002-ben jelent meg a Microsoft legfrissebb fejlesztőeszköze. Utalva a lényeges változtatásokra, a hálózati munka integrálására, a fejlesztőeszköz a Visual Studio.NET nevet kapta. Jelenleg az eszköz 2003-as frissítése a legutolsó verzió. A könyv szempontjából nincs lényeges különbség a két verzió között, így nem is térünk ki a különbségek bemutatására. Az új eszköz a korábbi fejlesztőrendszerekkel ellentétben nem a hagyományosnak tekinthető ún. Win32 alapú, hanem a .NET környezet alatt futtatható alkalmazásokat készít. Ez azt jelenti, hogy az új eszközzel

készült alkalmazások csak olyan operációs rendszer alatt futtathatók, melyek támogatják a .NET keretrendszert (.NET Framework). A fordító a forráskódot nem natív, hanem egy köztes kódra fordítja le. Ezt a köztes kódot MSIL (Microsoft Intermediate Language) néven szokták említeni.

A Visual Studio.NET telepítése tehát a keretrendszer telepítésével kezdődik. A .NET keretrendszer legfontosabb erői:

- Webszabványokon alapuló (XML, HTML, SOAP).
- Univerzális alkalmazási modellt használ, azaz egy .NET osztály, szolgáltatás tetszőleges .NET kompatibilis nyelvből használható. A .NET kompatibilitást a Common Language Specification (CLS) definiálja.
- Minden nyelvből ugyanazok a típusok használhatók (Common Type System)
- Minden .NET osztály a fejlesztők rendelkezésére áll. A keretrendszer a fejlesztőeszköz számára fordítási idejű szolgáltatásokat végez, míg az így lefordított alkalmazásoknak futási idejű környezetet biztosít (Common Language Runtime). A keretrendszer biztosítja a fentiek mellett az alkalmazások számára a már nem használt objektumok memóriabeli felszabadítását (Garbage Collection).

A keretrendszer osztálykönyvtára az alkalmazások típusa alapján több csoportba osztható:

- ADO.NET, adatbázis alkalmazások új generációs könyvtára.
- ASP.NET, webes alkalmazások (Web Forms) készítésének könyvtára.
- XML Web Service, interneten keresztül elérhető könyvtár.
- Windows alapú felhasználói (Windows Forms, Console Application), alkalmazói könyvtár.

Az osztálykönyvtárak használatához egy fejlesztő nyelvre van szükség. A Visual Studio.NET alapértelmezésben három nyelvet biztosít a fejlesztők számára. A Visual Basic és a Visual C++ nyelveket, mint az előd keretrendszerből megmaradt bázisnyelveket, és egy új nyelvet, amit a .NET keretrendszerhez fejlesztettek ki, a Visual C#-ot. Ma már a Java utódnyelv, a J# is a Visual Studio.NET 2003 fejlesztőrendszer része. Az új C# nyelvet szabványosították, ami a széleskörű elterjedésnek fontos feltétele. A C# nyelv szabványosított dokumentációjához ECMA-334 kód alatt férhetünk hozzá.

A .NET keretrendszer lényege, hogy a közös nyelvi definíciók már szabványosítottak. Ebben a szabványosításban a Microsoft mellett az informatikában érdekelt legnagyobb szervezetek is (HP, Intel, IBM, Sun, Fujitsu, Netscape) részt vettek. Ez ECMA-335-ös azonosítóval, mint a közös nyelvi infrastruktúra vagy eredeti nevén Common Language Infrastructure (CLI) nemzetközi szabványává vált.

1.4.4 MySQL 5.6.17

A MySQL egy többfelhasználós, többszálú, SQL-alapú relációs adatbázis-kezelő szerver. A szoftver eredeti fejlesztője a svéd MySQL AB cég, amely kettős licenccel tette elérhetővé a MySQL-t; választható módon vagy a GPL, vagy egy kereskedelmi licenc érvényes a felhasználásra.

2008 januárjában a Sun felvásárolta 800 millió dollárért a céget. 2010 január 27-én a Sun-t felvásárolta az Oracle Corporation, így a MySQL is Oracle tulajdonba került. A MySQL az egyik legelterjedtebb adatbázis-kezelő, aminek egyik oka lehet, hogy a teljesen nyílt forráskódú LAMP (Linux–Apache–MySQL–PHP) és a WAMP (Windows–Apache–MySQL–PHP) összeállítás részeként költséghatékony és egyszerűen beállítható megoldást ad dinamikus webhelyek szolgáltatására

1.5 A fejlesztés során használt programok

1.5.1 Microsoft Visual C# 2010 Express

A Microsoft Visual C# 2010 Express egy a Microsoft által ingyenesen biztosított vizuális fejlesztői eszköz magán célú használatra a C# fejlesztői környezethez mely leginkább azoknak szól akik a nyelvel még csak ismerkednek, kísérleteznek, tanulják.

1.5.2 WampServer 2.5

A WampServer a három legnépszerűbb és leggyakrabban használt szoftvert tartalmazza, amelyeket dinamikus weboldalak létrehozására szoktak használni.

Név szerint:

- Apache 2.4.9
- MySQL 5.6.17
- PHP 5.5.12

1.5.3 phpMyAdmin

A phpMyAdmin ahogy azt a neve is adja egy PHP nyelven megírt fejlesztői eszköz, mely a MySQL adatbázis fejlesztői környezethez ad vizuális kezelőfelületet és nagyban növeli az adatbázisok áttekinthetőségét és megkönnyíti azok kezelését. A WampServer feltelepítésével együtt feltelepítődik és a localhost/phpmyadmin URL címen keresztül érhető el. Jelenleg az általam használt phpMyAdmin verziója 4.1.14.

1.5.4 MySQL Connector Net 6.9.8

A MySQL Connector egy függvénykönyvtár, amelyet a MySQL fejlesztett ki és több különböző programozási nyelvre tette elérhetővé, köztük a .NET-re épülő C#-hoz is. Segítségével MySQL adatbázis szerver felé küldhetünk lekérdezéseket és kaphatjuk vissza azok eredményeit több különböző módon például a [MySqlDataReader](#)-rel.

1.6 Adatszerkezet (Adatbázis és Adattáblák)

1.6.1 Választott adatbázis szerver (MySQL)

A program adatbázis kezelő szerverének a MySQL-t választottam mivel az általam használt WampServer alapértelmezetten ez kínálja fel, ebben az SQL szerverben van a legtöbb tapasztalatom és MySQL adatbázis szerverhez tartozik egy PHP-ban megírt grafikus kezelő felület, a phpMyAdmin, ami megkönnyíti és felgyorsítja a munkát.

1.6.2 Adatbázis létrehozása

```
CREATE DATABASE alkatresz;
```

1.6.3 Normalizálás

A relációs adatbázisok tervezésének kialakult egy „normalizálás” elnevezésű módszere. Ennek célja, hogy az adatbázis táblái eleget tegyenek néhány biztonságot növelő szabálynak és a lehető legkevesebb redundanciát (felesleges adatot) tartalmazzák. Ez egy többlépcsős folyamat, amelyben az adatbázis különböző „normálformulákba” kerül.

Az első normálformula (1NF)

Kritériumai:

Az adatokat tartalmazó táblázatokban

- a sorok (rekordok) sorrendje tetszőleges
- minden oszlopnak (mezőnek) egyedi neve van
- az egyes oszlopokban (mezőkben) azonos típusú és tulajdonságot leíró értékek vannak
- egy cellában (mezőben) csak egy elemi tulajdonságérték szerepelhet

Az első kritérium feltételezi, hogy van valamilyen kulcskifejezés, ami szerint a rekordok megkülönböztethetők egymástól. A második és a harmadik feltétel az adatok táblázatba szervezésének módját írják le. A két utolsó pedig a rekordok szerkezetére és tartalmára vonatkozik.

A második normálformula (2NF)

Kritériumai:

- legyen az adatbázis első normálformulában (1NF)
- minden nem kulcs mező teljes függőségben álljon a kulcstól

Teljes függőség alatt azt értjük, hogy egy A tulajdonság (mező) teljesen függ egy B tulajdonsághalmaztól (egy vagy több mezőben levő adatoktól), ha ez utóbbi egyértelműen meghatározza az előbbi értékét, de ehhez B-ből már nem hagyható el semelyik összetevő.

Ha a kulcs egyetlen mezőből áll, akkor a 2NF feltételei máris teljesülnek. Ha a kulcs több mezőből áll, akkor ne legyen nem kulcsmező, amely csak a kulcs egy részétől függ.

A harmadik normálformula (3NF)

Kritériumai:

- a reláció második normálformátumban van (2NF)
- nincs az adatbázisban tranzitív függőség

Tranzitív függőség esetén egy relációban egy tulajdonság függ az elsődleges kulcstól, de olyan tulajdonságtól is függ, amely nem része a kulcsnak.

(forrás: <http://szinyeigimibp.hu/moodle/mod/page/view.php?id=316>)

1.6.4 Adattáblák

1.6.5 raktar tábla

raktar	
Raktáron lévő termékek adatait tárolja.	
termek_id PK INDEX, varchar(10)	Termék egyedi azonosítója.
termeknev varchar(256)	Termékneve, megnevezése.
afa_id INDEX int(2)	Kapcsolat az afakulcs táblával
gyarto_id INDEX int(4)	Kapcsolat a gyarto táblával.
tipus_id INDEX int(4)	Kapcsolat a tipus táblával.
mennyiseg int(4)	Hány db van raktáron az adott termékből.
min_keszlet int(3)	Raktáron lévő min. megengedett mennyiség
beszerzesiar int(10)	Beszerezési ár
eladasiar int(10)	eladasiar=beszerzesiar*(1+afakulcsok.haszonkulcs)
allapot_id INDEX int(1)	Milyen állapotú a termék. Pl.: Új, használt...
megjegyzes text	Leírás a termékről.
--raktar tábla létrehozása CREATE TABLE `raktar` (`termek_id` varchar(10) COLLATE utf8_hungarian_ci NOT NULL, `termeknev` varchar(256) COLLATE utf8_hungarian_ci NOT NULL, `afa_id` int(2) NOT NULL, `gyarto_id` int(4) NOT NULL, `tipus_id` int(4) NOT NULL, `mennyiseg` int(4) NOT NULL, `min_keszlet` int(3) NOT NULL, `beszerzesiar` int(10) NOT NULL, `eladasiar` int(10) NOT NULL, `allapot_id` int(1) NOT NULL, `megjegyzes` text COLLATE utf8_hungarian_ci NOT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;	
--A raktar tábla indexei ALTER TABLE `raktar` ADD PRIMARY KEY (`termek_id`), ADD KEY `gyarto_id` (`gyarto_id`), ADD KEY `tipus_id` (`tipus_id`), ADD KEY `allapot_id` (`allapot_id`), ADD KEY `afa_id` (`afa_id`);	
-- Megkötések a `raktar` táblához ALTER TABLE `raktar` ADD CONSTRAINT `raktar_ibfk_1` FOREIGN KEY (`afa_id`) REFERENCES `afakulcsok` (`afa_id`), ADD CONSTRAINT `raktar_ibfk_2` FOREIGN KEY (`gyarto_id`) REFERENCES `gyarto` (`gyarto_id`), ADD CONSTRAINT `raktar_ibfk_3` FOREIGN KEY (`tipus_id`) REFERENCES `tipus` (`tipus_id`), ADD CONSTRAINT `raktar_ibfk_4` FOREIGN KEY (`allapot_id`) REFERENCES `allapot` (`allapot_id`);	

1.6.6 gyarto tábla

gyarto A raktáron lévő termékek gyártóinak neveit tárolja.	
gyarto_id PK INDEX int(4)	Termék gyártójának egyedi azonosítója az adatbázisban.
gyarto varchar(50)	Gyártó megnevezése.
<pre>--gyarto tábla létrehozása CREATE TABLE `gyarto` (`gyarto_id` int(4) NOT NULL, `gyarto` varchar(50) COLLATE utf8_hungarian_ci NOT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8 hungarian ci;</pre>	
<pre>-- A gyarto tábla indexei ALTER TABLE `gyarto` ADD PRIMARY KEY (`gyarto_id`);</pre>	
<pre>-- AUTO_INCREMENT a táblához `gyarto` ALTER TABLE `gyarto` MODIFY `gyarto_id` int(4) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=0;</pre>	

1.6.7 típus tábla

típus A raktáron lévő termékek típusait tárolja.	
tipus_id PK INDEX int(4)	Terméktípusának egyedi azonosítója az adatbázisban.
tipus varchar(50)	Típus megnevezése.
<pre>--tipus tábla létrehozása CREATE TABLE `tipus` (`tipus_id` int(4) NOT NULL DEFAULT '0', `tipus` varchar(50) COLLATE utf8_hungarian_ci NOT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8 hungarian ci;</pre>	
<pre>-- A tipustábla indexei ALTER TABLE `tipus` ADD PRIMARY KEY (`tipus_id`);</pre>	
<pre>-- AUTO_INCREMENT a táblához `tipus` ALTER TABLE `tipus` MODIFY `tipus_id` int(4) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=0;</pre>	

1.6.8 állapot tábla

allapot A raktáron lévő termékek állapotát tárolja.	
allapot_id PK INDEX int(4)	Termék állapotának egyedi azonosítója az adatbázisban.
allapot varchar(20)	Állapot megnevezése.
<pre>--allapot tábla létrehozása CREATE TABLE `allapot` (`allapot_id` int(1) NOT NULL, `allapot` varchar(20) COLLATE utf8_hungarian_ci NOT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8 hungarian ci;</pre>	
<pre>-- Az állapot tábla indexei ALTER TABLE `hiba_allapot` ADD PRIMARY KEY (`hiba_allapot_id`);</pre>	
<pre>-- AUTO_INCREMENT a táblához `allapot` ALTER TABLE `allapot` MODIFY `allapot_id` int(1) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=0;</pre>	

1.6.9 rendeles tábla

rendeles	
Az adatbázis adminisztrátor által összeállított termékeket tárol, melyeket meg kell rendelni.	
rendeles_id PK INDEX. varchar(4)	Rendelés azonosítója.
termeknev varchar(256)	Megrendelendő termék neve.
gyarto_id INDEX int(4)	Gyártó azonosítója. Kapcsolat a gyarto táblával.
tipus_id INDEX int(4)	Típus azonosítója. Kapcsolat a tipus táblával.
mennyiseg int(3)	Megrendelni kívánt mennyiség.
beszerzesiar int(10)	Beszerezési ár.
<pre>--rendeles tábla létrehozása CREATE TABLE `rendeles` (`rendeles_id` varchar(4) COLLATE utf8_hungarian_ci NOT NULL, `termeknev` varchar(256) COLLATE utf8_hungarian_ci DEFAULT NULL, `gyarto_id` int(4) DEFAULT NULL, `tipus_id` int(4) DEFAULT NULL, `mennyiseg` int(3) DEFAULT NULL, `beszerzesiar` int(10) DEFAULT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8 hungarian ci;</pre>	
<pre>-- A rendeles tábla indexei ALTER TABLE `rendeles` ADD PRIMARY KEY (`rendeles_id`), ADD KEY `gyarto_id` (`gyarto_id`), ADD KEY `tipus_id` (`tipus_id`);</pre>	
<pre>-- Megkötések a táblához `rendeles` ALTER TABLE `rendeles` ADD CONSTRAINT `rendeles_ibfk_1` FOREIGN KEY (`tipus_id`) REFERENCES `tipus` (`tipus_id`), ADD CONSTRAINT `rendeles_ibfk_2` FOREIGN KEY (`gyarto_id`) REFERENCES `gyarto` (`gyarto_id`);</pre>	

1.6.10 afakulcsok tábla

afakulcsok	
A raktáron lévő termékek árképzési szabályához tárol adatokat.	
afa_id PK INDEX int(2)	Az afakulcsok tábla egyedi azonosítója.
afa_kulcs double	Az ÁFA mértékét adja meg (pl.: 0,27)
afakod varchar(3)	Az ÁFA szöveges megjelenítéséért felel (pl.:27%)
haszonkulcs double	A eladónak a beszerzési árra ennyi hasznot tesz rá.
<pre>--afakulcsok tábla létrehozása CREATE TABLE `afakulcsok` (`afa_id` int(2) NOT NULL, `afa_kulcs` double NOT NULL, `afa_kod` varchar(3) CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NOT NULL, `haszonkulcs` double NOT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8 hungarian_ci;</pre>	
<pre>-- Az afakulcsok tábla indexei ALTER TABLE `afakulcsok` ADD PRIMARY KEY (`afa_id`),</pre>	
<pre>-- AUTO_INCREMENT a táblához `afakulcsok` ALTER TABLE `afakulcsok` MODIFY `afa_id` int(2) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=0;</pre>	

1.6.11 users tábla

users A felhasználók bejelentkezéséhez szükséges adatokat tárolja .	
user_id PK INDEX int(4)	Felhasználó egyedi azonosítója. Kapcsolat a usersperm táblával.
nev varchar(15)	A felhasználó neve amivel bejelentkezik.
jelszo varchar(150)	A felhasználó jelszava, amivel bejelentkezik. Titkosítva tárolódik.
emlekezteto varchar(30)	A felhasználó jelszó emlékeztetője.
registrated datetime	A felhasználó regisztrálásának dátuma és id-je.
lastlogin datetime	Az felhasználó legutóbbi dátuma és ideje
-- A users tábla létrehozása CREATE TABLE `users` (`user_id` int(4) NOT NULL, `nev` varchar(15) COLLATE utf8_hungarian_ci NOT NULL, `jelszo` varchar(150) COLLATE utf8_hungarian_ci NOT NULL, `emlekezteto` varchar(30) COLLATE utf8_hungarian_ci NOT NULL, `registrated` datetime NOT NULL, `lastlogin` datetime NOT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;	
-- users tábla indexei ALTER TABLE `users` ADD PRIMARY KEY (`user_id`), ADD UNIQUE KEY `nev` (`nev`),	
-- AUTO_INCREMENT a táblához `users` ALTER TABLE `users` MODIFY `user_id` int(4) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=0;	

1.6.12 usersperm tábla

usersperm	
user_id PK INDEX int(4)	Felhasználó egyedi azonosítója.
adduser tinyint(1)	A felhasználó hozzáadhat más felhasználókat vagy törölhet.
modb tinyint(1)	Adatbázis adminisztrátori jog. Termékek módosítása, újak feltöltése, törlés, rendelés leadása.
fullperm tinyint(1)	Teljes hozzáférés az adatbázishoz. Új típus, gyártó, állapot hozzáadása, módosítása, törlése.
<pre>--usersperm tábla létrehozása CREATE TABLE `usersperm` (`user_id` int(4) NOT NULL, `adduser` tinyint(1) NOT NULL, `modb` tinyint(1) NOT NULL, `fullperm` tinyint(1) NOT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;</pre>	
<pre>-- usersperm tábla indexei ALTER TABLE `usersperm` ADD PRIMARY KEY (`user_id`),</pre>	
<pre>-- AUTO_INCREMENT a táblához `usersperm` ALTER TABLE `usersperm` MODIFY `user_id` int(4) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=0;</pre>	
<pre>-- Megkötések a táblához `usersperm` ALTER TABLE `usersperm` ADD CONSTRAINT `usersperm_ibfk_1` FOREIGN KEY (`user_id`) REFERENCES `users` (`user_id`);</pre>	

1.6.13 kosar tábla

kosar	
user_id INDEX int(4)	Felhasználó azonosítója, kihez tartozik a kosár tartalma. Kapcsolat a users táblához.
termek_id INDEX int(4)	a kosárba rakott termék neve
darab int(4)	Hány darabot rakott a kosárba. Ez lehet több is mint amennyi a raktáron van.
<pre>--kosar tábla létrehozása CREATE TABLE `kosar` (`user_id` int(4) NOT NULL, `termek_id` varchar(10) CHARACTER SET utf8 COLLATE utf8_hungarian_ci NOT NULL, `darab` int(4) NOT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;</pre>	
<pre>-- kosar tábla indexei ALTER TABLE `kosar` ADD KEY `user_id` (`user_id`), ADD KEY `termek_id` (`termek_id`);</pre>	
<pre>-- Megkötések a táblához `kosar` ALTER TABLE `kosar` ADD CONSTRAINT `kosar_ibfk_1` FOREIGN KEY (`user_id`) REFERENCES `users` (`user_id`), ADD CONSTRAINT `kosar_ibfk_2` FOREIGN KEY (`termek_id`) REFERENCES `raktar` (`termek_id`);</pre>	

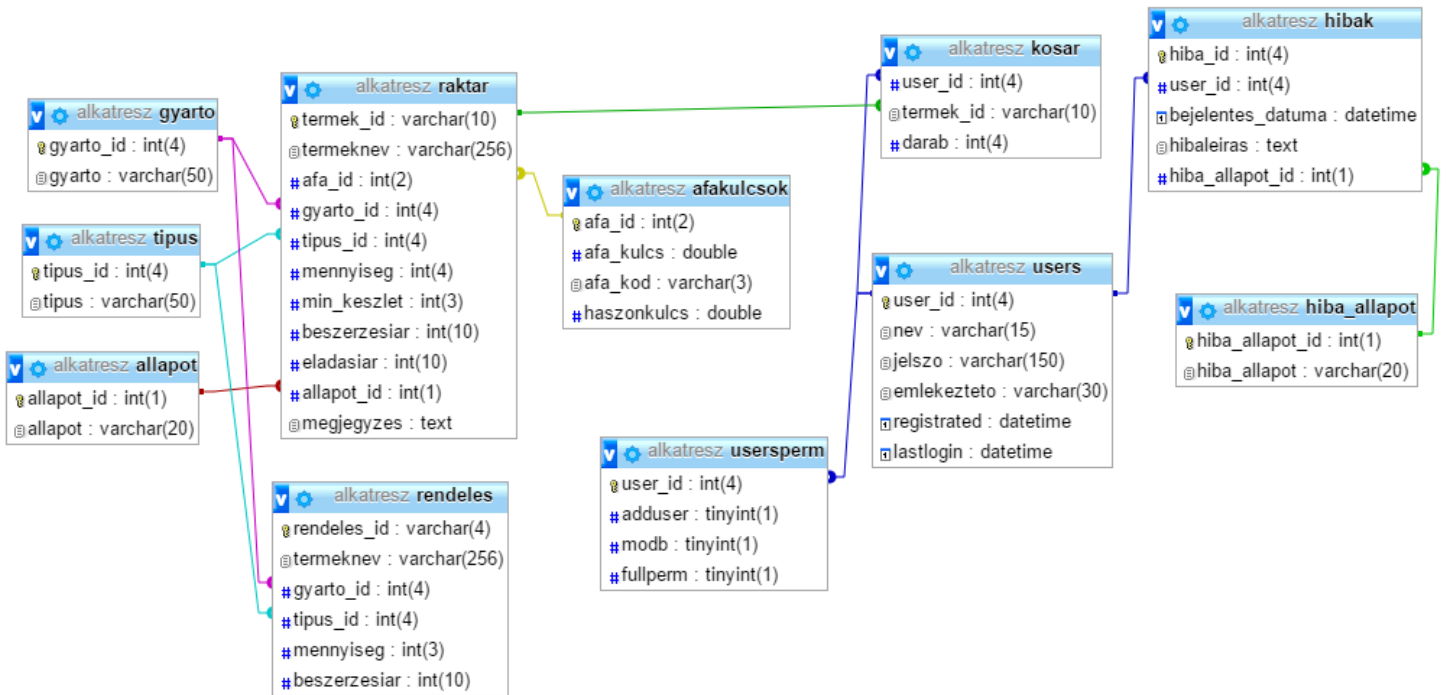
1.6.14 hibak tábla

hibak	
hiba_id PK int(4)	Hiba bejegyzésének egyedi azonosítója.
user_id int(4)	Felhasználó azonosítója, kihez tartozik a kosár tartalma. Kapcsolat a users táblához.
bejelentes_datuma datetime	bejelentés dátuma és ideje.
hibaleiras text	A hiba leírása. Mikor, hol és hogyan jelentkezett a hiba. (pl.: elírás az adatbázisban).
hiba_allapot_id INDEX int(1)	A hiba állapotának azonosítója.
<pre>--hiba tábla létrehozása CREATE TABLE `hibak` (`hiba_id` int(4) NOT NULL, `user_id` int(4) NOT NULL, `bejelentes_datuma` datetime NOT NULL, `hibaleiras` text CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NOT NULL, `hiba_allapot_id` int(1) NOT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;</pre>	
<pre>-- hibak tábla indexei ALTER TABLE `hibak` ADD PRIMARY KEY (`hiba_id`), ADD UNIQUE KEY `hiba_id` (`hiba_id`), ADD KEY `hiba_allapot_id` (`hiba_allapot_id`), ADD KEY `user_id` (`user_id`);</pre>	
<pre>-- Megkötések a táblához `hibak` ALTER TABLE `hibak` ADD CONSTRAINT `hibak_ibfk_1` FOREIGN KEY (`hiba_allapot_id`) REFERENCES `hiba_allapot` (`hiba_allapot_id`), ADD CONSTRAINT `hibak_ibfk_2` FOREIGN KEY (`user_id`) REFERENCES `users` (`user_id`);</pre>	

1.6.15 hiba_allapot tábla

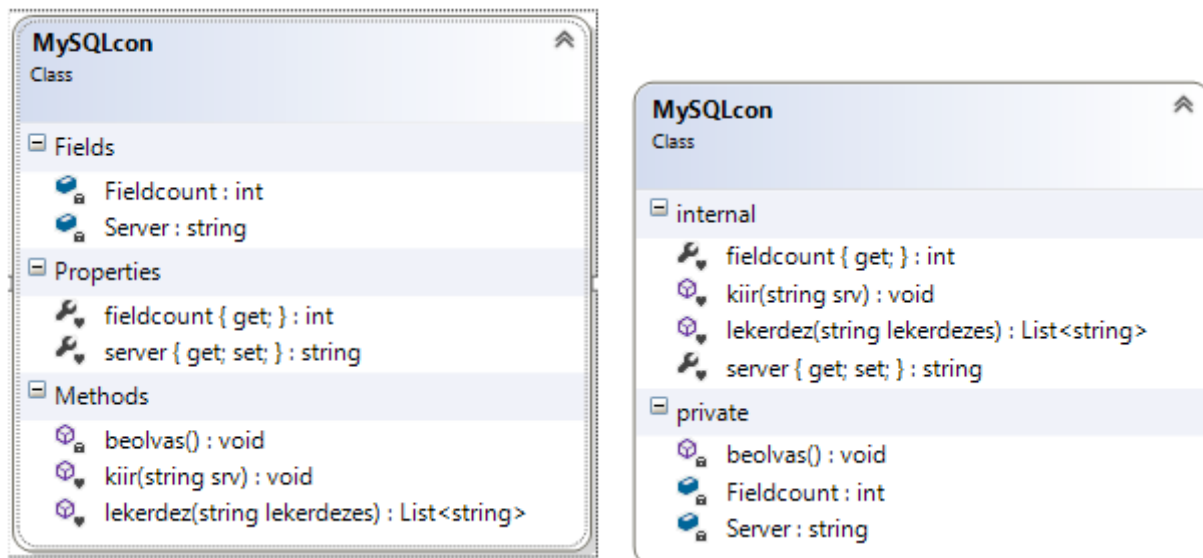
hiba_allapot	
hiba_allapot_id PK INDEX int(1)	A hiba lehetséges állapotainak egyedi azonosítója.
hiba_allapot varchar(20)	A lehetséges hiba állapotok. (pl.: admin látta, dolgozunk rajta, javítva).
<pre>--hiba_allapot tábla létrehozása CREATE TABLE `hiba_allapot` (`hiba_allapot_id` int(1) NOT NULL, `hiba_allapot` varchar(20) CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NOT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8 hungarian ci;</pre>	
<pre>-- hiba_allapot tábla indexei ALTER TABLE `hiba_allapot` ADD PRIMARY KEY (`hiba_allapot_id`), ADD UNIQUE KEY `hiba_allapot_id` (`hiba_allapot_id`);</pre>	

1.6.16 Táblák kapcsolati rendszerterve



1.7 Osztályok

1.7.1 MySQL kapcsolatért felelős osztály (MySQLcon.cs)



Ez egy saját készítésű osztály, amely a MySQL szerveren lévő adatbázishoz való kapcsolódást és lekérdezés futtatását könnyíti meg egy string típusú lista függvény segítségével. Az osztály tartalmazza a `MySql.Data.dll`-t, a `MySqlConnection` függvénykönyvtárát.

Használt névterek (az alapértelmezetten hozzáadottakon túl):

```
using System.Windows.Forms;
using MySql;
using MySql.Data;
using MySql.Data.Common;
using MySql.Data.MySqlClient;
using System.IO;
```

`Fieldcount`: egy egész típusú `private` védettségű változó, amely az oszlopok számát tárolja. Értéke mindig a `MySqlDataReader` osztály példányának a `FieldCount` értékével egyezik meg.

`Fieldcount`: egész típusú property, amely a `Fieldcount` értékét adja vissza. A lekérdez függvény által vissza adott listát e mentén tördeljük sorokra és töltjük fel vele a `DataGridView` sorait egy másik osztályban.

`Server`: szöveg típusú változó. A kapcsolódáshoz szüksége adatokat tárolja.

`server`: ezen a property-n keresztül lehet beállítani a `Server` változó értékét más osztályból.

beolvas: eljárás, amely az srv.dll fájlban eltárolt server adatokat olvassa be és tárolja el a Server változóban. Ha beolvasandó fájl nem létezik akkor megnyitja a Beallitasok formot és a felhasználónak be kell állítania a server adatait. Ha a beolvasás sikeres, de üres volt akkor törli és újra próbálkozik és az előző eset fog megtörténni.

kiir: Eljárás. Az srv.dll fájlba kiírja a server adatait, amelyeket a Beallitas formban adtunk meg. Kívülről is meghívható.

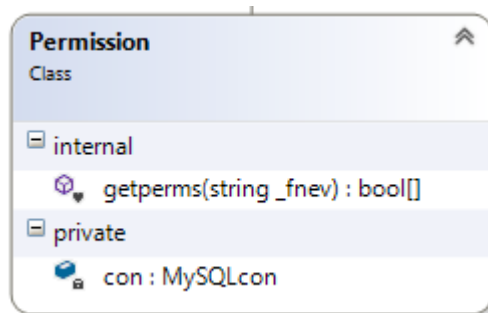
```
internal List<string> lekerdez(string lekerdezes)//lekérdezés futtatása a server felé
{
    //try->catch rész megcsinálni ha kész az összes funkció debugja
    List<string> visszater = new List<string>();
    beolvas();
    try
    {
        string[] db = Server.Split('-');//A beolvasot serveradatok feldaraobolása
és tömben tárolása
        string kapcsolat = "SERVER=" + db[0] + ";" + "DATABASE=" + db[1] + ";" +
"UID=" + db[2] + ";" + "PASSWORD=" + db[3] + ";";
        MySqlConnection con = new MySqlConnection(kapcsolat);
        MySqlCommand parancs = con.CreateCommand();
        MySqlDataReader olvaso;
        parancs.CommandText = lekerdezes;
        con.Open();
        olvaso = parancs.ExecuteReader();
        while (olvaso.Read())
        {
            for (int i = 0; i < olvaso.FieldCount; i++)
            {
                visszater.Add((olvaso.GetValue(i).ToString()));
            }
        }
        Fieldcount = olvaso.FieldCount;//Fieldcount értékének beállítása
        con.Close();
        return visszater;
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
        return visszater;
    }
}
```

```

private void beolvas()//A kapcsolódáshoz szükséges adatok beolvasása
{
    try
    {
        if (File.Exists("srv.dll"))
        {
            StreamReader be = new StreamReader("srv.dll");
            Server = be.ReadLine();
            be.Close();
            if (Server == "")
            {
                File.Delete("srv.dll");
                beolvas();
            }
        }
        else
        {
            Beallitasok s = new Beallitasok();
            s.ShowDialog();
            beolvas();
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}

```

1.7.2 Permission osztály (Permission.cs)



Ez az osztály felel a jogok lekérdezéséért és átadásáért. Egyetlen függvénye a getperms.

Használata és működése:

Paraméterként át kell adni a felhasználó nevet.

A függvény lekérdezi a felhasználó jogait egy string típusú listába. Egy for ciklus áttölti a jogokat egy 3 elemű logikai tömbbe immár logikaivá konvertálva a visszakapott értékeket.

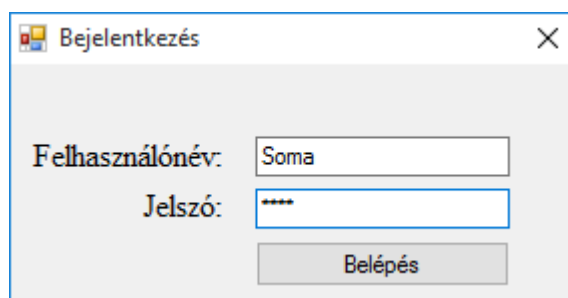
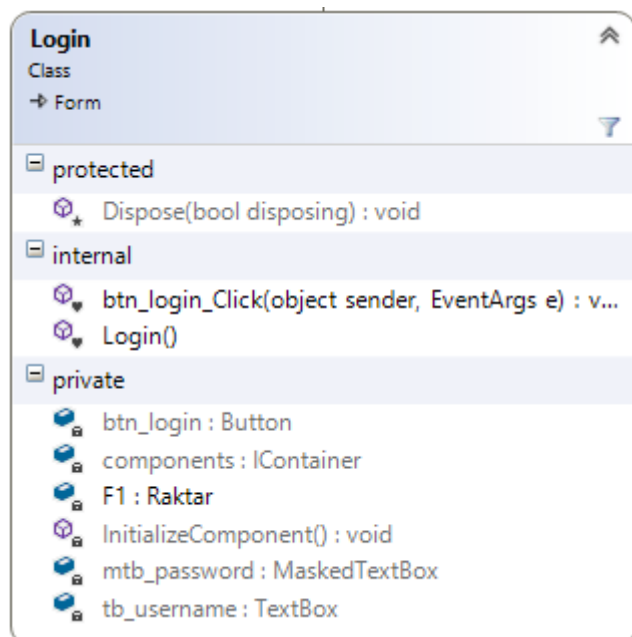
A függvény visszatér a 3 elemű logikai tömbbel. A használandó osztályban példányosítjuk.

Létrehozunk egy 3 elemű logikai tömböt amibe a jogokat fogjuk tölteni. Az új 3 elemű logikai tömböt feltöltjük a Permission osztály getperms függvényével.

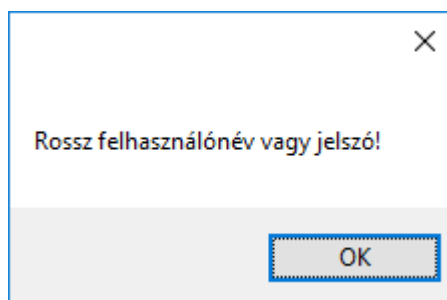
Meghatározzuk, hogy az adott jogokkal mi az amit engedélyezünk és mi az ami tiltva marad.

```
class Permission
{
    MySQLcon con = new MySQLcon();
    internal bool[] getperms(string _fnev)
    {
        List<string> perms = con.lekerdez("SELECT adduser, modb,
fullperm FROM users inner join usersperm ON users.user_id=usersperm.user_id
WHERE nev='" + _fnev + "';");
        bool[] _uperm = new bool[3];
        for (int i = 0; i < perms.Count; i++)
        {
            _uperm[i] = Convert.ToBoolean(perms[i]);
        }
        return _uperm;
    }
}
```


1.7.3 Bejelentkezés (Form2.cs)



A program indulásakor ez az első Form a mi megjelenik. A felhasználó beírja a felhasználó nevét és jelszavát és ha stimmelnek az adatok, akkor tovább engedi a felhasználót a Form1-re, a fő formra. Ha az adatok nem egyeznek, akkor kiírja egy MessageBox-ba, hogy „Rossz felhasználónév vagy jelszó!”. A MessageBox-on csak egy OK gomb van, amire kattintva eltűnik és kitörli a beírt adatokat a tb_username és mtb_password szöveges beviteli mezőkből. Ezután a felhasználó újra próbálkozhat.



Felhasználó adatainak ellenőrzésének folyamata:

- A felhasználó beviszi az adatait a megfelelő mezőkbe.
- A Belépés gombra kattintva a btn_login_Click esemény lefut és ha helyesek az adatok, akkor a felhasználót tovább engedi a programban, ellenkező esetben az előbb leírt folyamat játszódik le.

btn_login_Click:

- MySQLcon osztály példányosítása.
- Lekérdezés futtatása a MySQL szerveren a(z) users táblában a felhasználó által bevitt névvel és jelszóval, majd a visszatért adatot egy listában eltárolja.
- Vizsgálat, hogy a lista első eleme nem üres e. Ha üres akkor hiba üzenet „Rossz felhasználónév vagy jelszó!”, majd az egész kezdődik előlről. Ellenkező esetben a Form1 _felhasznalonev változó értékének beállítása a bejelentkezett felhasználónak nevére.
- Felhasználó utolsó bejelentkezési idejének frissítése.
- Form1 mutatása.
- Form1 engedélyezése.

1.7.4 Raktár (Form1)

	Terméknév	Gyártó	Típus	Állapot	Mennyiség	Beszerzési Ár	Eladási Ár	Megjegyzés
*								

Ez a fő form, a program legfőbb része.

Funkciói:

- Keresés az adatbázisban
- Új áru feltöltése
- Áruk módosítása
- Rendelés összeállítása

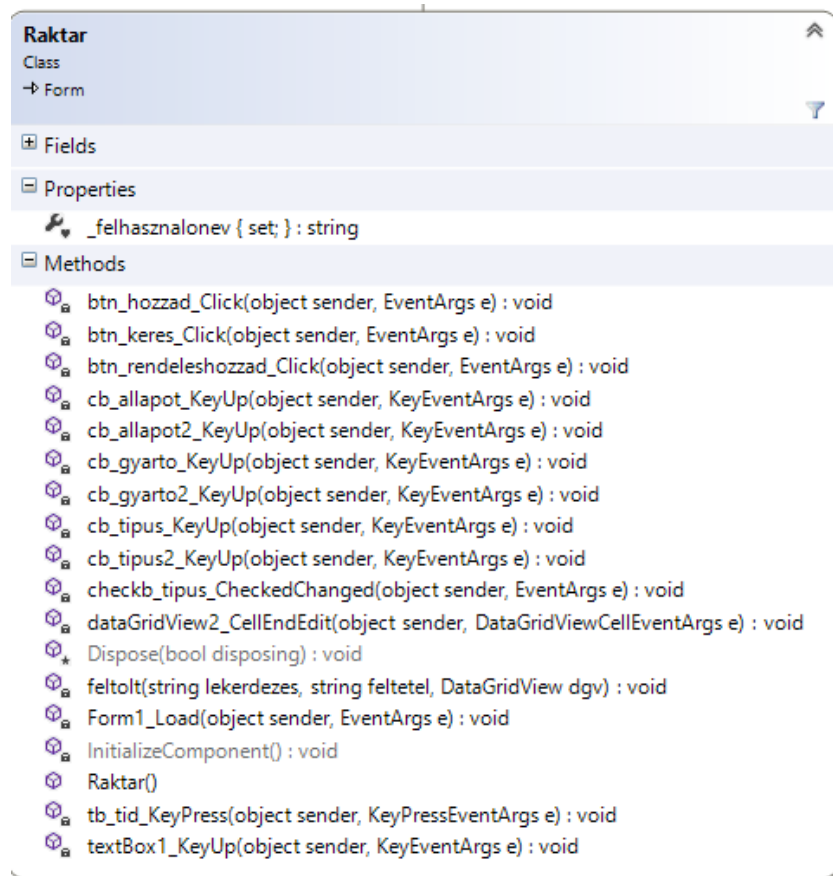
A Form betöltésekor (Form1_Load esemény) példányosítja a Permission osztályt, majd átadja meghívja a getperms függvényt és átadja neki a felhasználó nevet a felhasználonev változó segítségével. A függvény visszatért értékét egy 3 elemű logikai tömbben eltárolja majd a kapott jogoknak megfelelően engedélyezi vagy letiltja a vezérlőket. Továbbá feltölti a Form1-en található összes ComboBox kontrollert a megfelelő adatokkal.

```
Permission p = new Permission();  
bool[] uperms = p.getperms(felhasznalonev);  
menu_felhasznalok.Enabled = uperms[0];  
menu_felhasznalok.Visible = uperms[0];
```

feltölt eljárás:

- paraméterei: a lekérdezés SELECT része, a lekérdezés WHERE része és egy DataGridView példány.
- Ez az eljárás feltölti a paraméterként átadott DataGridView-t a lekérdezés eredményével. A lekérdezés összekapcsoló része (join) bele van építve a függvény, a lekerdezés paraméteréhez fűzi hozzá és ezt adja át a MySQLcon osztály lekerdez

függvényének. A visszatért értékekkel feltölti a DataGridView-t a MySQLcon osztály fieldcount egész típusú változója határozza meg egy sor hosszát.



```
private void feltolt(string lekerdezes, string feltetel, DataGridView dgv)
{
    lekerdezes += " raktar INNER JOIN gyarto ON raktar.gyarto_id=gyarto.gyarto_id INNER
JOIN tipus ON raktar.tipus_id=tipus.tipus_id INNER JOIN allapot ON
raktar.allapot_id=allapot.allapot_id INNER JOIN afakulcsok ON
raktar.afa_id=afakulcsok.afa_id";
    List<string> adatok = connection.lekerdez(lekerdezes + feltetel);
    if (adatok.Count >= 8)
    {
        dgv.RowCount = adatok.Count / connection.fieldcount;
        int s = 0;
        for (int i = 0; i < dgv.RowCount; i++)
        {
            for (int j = 0; j < dgv.ColumnCount; j++)
            {
                dgv[j, i].Value = adatok[s];
                s++;
            }
        }
    }
    else
    {
        MessageBox.Show("Nincs találat a keresett szóra " + tb_termek.Text + ".");
    }
}
```

tabPage_kereses:

- `checkbox_tipus_CheckedChanged`: Aktiválja a részletes keresést engedélyezi a `cb_tipus`, `cb_gyarto`, `cb_allapot`, `num_artol` és a `num_arig` vezérlőket.
- `cb_tipus_KeyUp`: Típusonként lehet keresni.
- `cb_tipus_KeyUp`: Gyártonként lehet keresni.
- `cb_allapot_KeyUp`: Állapotonként lehet keresni.
- `textBox1_keyUp`: Termék neve szerinti keresés. Mind egyszerű, mind részletes kereséshez ki kell tölteni. A mindent helyettesítő karakter a % jel.
- Kosárhoz ad oszlopban bepipált sorokat a `btn_kosarbarak_Click` eseményre hozzáadja kosar adattáblához az adatbázisban.
- A `btn_kosar_Click` eseményre a kosar tábla minden adata betöltődik a `dgvkereses DataGridView`-ba.
- `btn_kosaruritse_Click` esemény törli a kosár tartalmát.

tabPage_keszletmodositas: (modb vagy fullperm jog szükséges)

- `tb_tid`: Új áru felvételekor meg kell adni a termékazonosító számát (pl.: vonal kód).
- `tb_termekek`: Az adatbázisba felvenni kívánt termék neve.
- `checkbox_tipus2`: Részletes keresést tesz lehetővé, ha módosítandó árut keresünk.
- `cb_gyarto2`: Gyártó megadása az adatbázisba felvenni kívánt termékhez.
- `cb_tipus2`: Terméktípus megadása az adatbázisba felvenni kívánt termékhez.
- `cb_allapot2`: Termék állapot megadása az adatbázisba felvenni kívánt termékhez.
- `numkeszlet`: Mennyi van az adott termékből.
- `numminkeszlet`: Minimum készlet megadása.
- `numar`: Az adatbázisba felvenni kívánt termék beszerzési ára.
- `tb_leiras`: Termék leírás megadása, ha üresen hagyjuk akkor a `tb_leiras` szövege „nincs” lesz.
- `btn_hozzad_Click`: Ha minden mezőt kitöltöttünk, akkor feltölti az adatbázisba. Ellenkező esetben figyelmeztet, hogy nem töltöttünk ki minden mezőt.
- `btn_kereses_Click`: A kitöltött mezők adataival keresést hajt végre az adatbázisban és `dgvkeszletmodositas DataGridView`-ban megjeleníti.
- `dataGridView2_CellEndEdit`: `dgvkeszletmodositas DataGridView` bármely írásra engedélyezett oszlopán módosítást hajtunk végre, akkor az a szerkesztő módból kilépve frissíti az adatbázisban.
- Törlés oszlopban a gombra kattintva törli az adott sort az adatbázisból.

tabPage_rendeles: (modb vagy fullperm jog szükséges)

Rendelés adatainak megadása után a btn_listahozad_Click eseményre dgrendelsehez adja az adatokat.

A btn_rendelseelkuldes gombra véglegesíti a rendelést és feltölti az adatbázisba.

Raktar

Class

→ Form

Fields

btn_hozzad : Button

btn_keres : Button

btn_Isitahozad : Button

btn_rendeles_elkuldes : Button

btn_rendeleselkuldes : Button

cb_allapot : ComboBox

cb_allapot2 : ComboBox

cb_gyarto : ComboBox

cb_gyarto2 : ComboBox

cb_gyarto3 : ComboBox

cb_tipus : ComboBox

cb_tipus2 : ComboBox

cb_tipus3 : ComboBox

checkb_tipus : CheckBox

checkb_tipus2 : CheckBox

components : IContainer

connection : MySQLcon

dgvkereses : DataGridView

dgvkeszletmodositas : DataGridView

dgvrendeles : DataGridView

felhasznalok_kezelese_almenu : ToolStripMenuItem

felhasznalonev : string

hatterszin : ToolStripMenuItem

kilepes_almenu : ToolStripMenuItem

kosarhozad : DataGridViewButtonColumn

num_ar3 : NumericUpDown

num_arig : NumericUpDown

num_artol : NumericUpDown

num_mennyiseg3 : NumericUpDown

numar : NumericUpDown

numkeszlet : NumericUpDown

numminkeszlet : NumericUpDown

rendeles_id : DataGridViewTextBoxColumn

tabControl1 : TabControl

tabPage_kereses : TabPage

tabPage_keszletmodositas : TabPage

tabPage_rendeles : TabPage

tb_leiras : TextBox

tb_teremek2 : TextBox

tb_termek : TextBox

tb_termeknev3 : TextBox

tb_tid : TextBox

termek_id : DataGridViewTextBoxColumn

termeknev : DataGridViewTextBoxColumn

termektipus : DataGridViewTextBoxColumn

uj_tipus_allapot_gyarto : ToolStripMenuItem

ujaru_almenu : ToolStripMenuItem

Properties

_felhasznalonev { set; } : string

Methods

btn_hozzad_Click(object sender, EventArgs e) : void

btn_keres_Click(object sender, EventArgs e) : void

btn_rendeles_elkuldes_Click(object sender, EventArgs e) : void

btn_rendeleselkuldes_Click(object sender, EventArgs e) : void

btn_rendeleshozzad_Click(object sender, EventArgs e) : void

cb_allapot_KeyUp(object sender, KeyEventArgs e) : void

cb_allapot2_KeyUp(object sender, KeyEventArgs e) : void

cb_gyarto_KeyUp(object sender, KeyEventArgs e) : void

cb_gyarto2_KeyUp(object sender, KeyEventArgs e) : void

cb_tipus_KeyUp(object sender, KeyEventArgs e) : void

cb_tipus2_KeyUp(object sender, KeyEventArgs e) : void

checkb_tipus_CheckedChanged(object sender, EventArgs e) : void

dataGridView2_CellEndEdit(object sender, DataGridViewCellEventArgs e) : void

dgvkereses_CellClick(object sender, DataGridViewCellEventArgs e) : void

Dispose(bool disposing) : void

feltolt(string lekerdezes, string feltetel, DataGridView dgv) : void

Form1_Load(object sender, EventArgs e) : void

InitializeComponent() : void

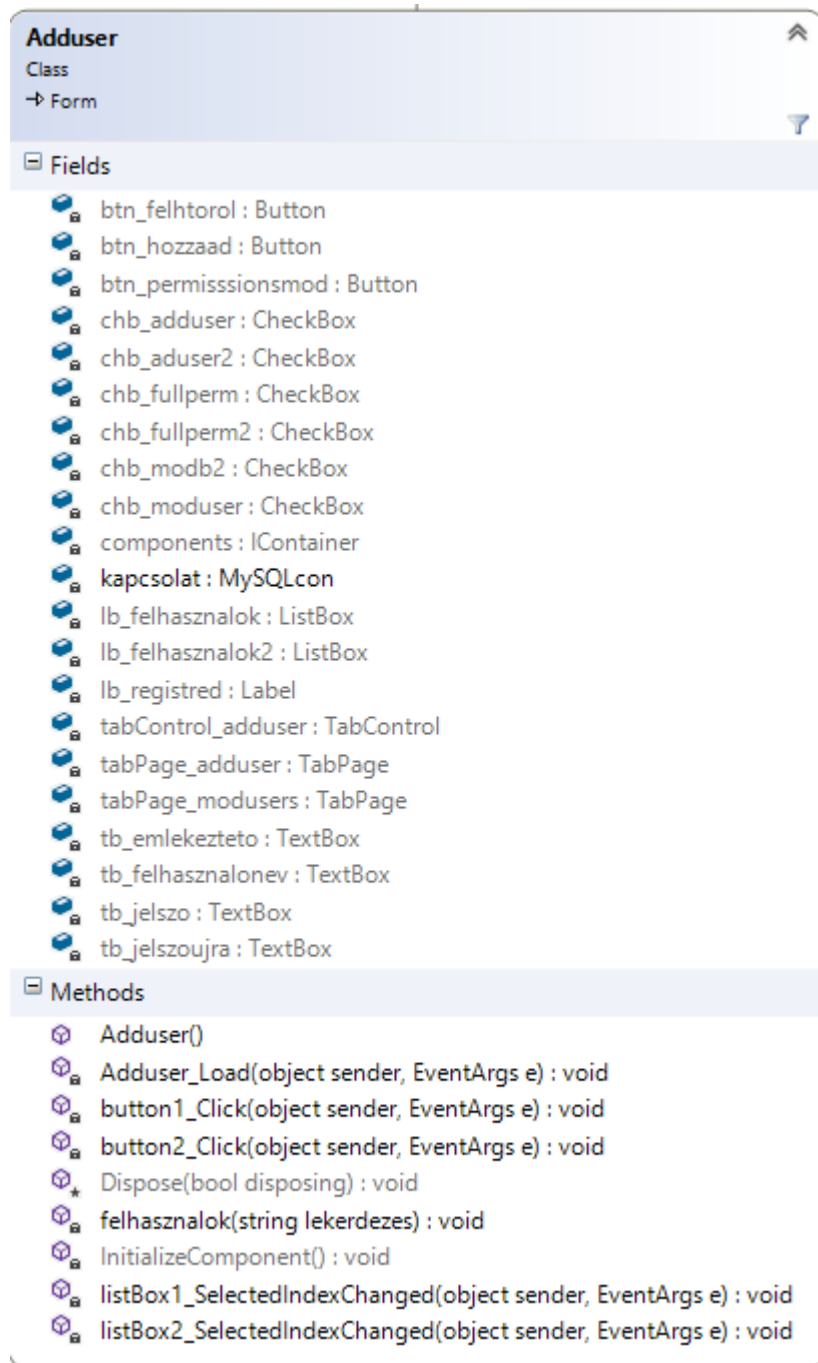
Raktar()

tb_tid_KeyPress(object sender, KeyPressEventArgs e) : void

textBox1_KeyUp(object sender, KeyEventArgs e) : void

uj_tipus_allapot_gyarto_Click(object sender, EventArgs e) : void

1.7.5 Adduser (Adduser.cs)



Ez a Form a felhasználók kezelésében segít. Fullperm vagy adduser joggal kell rendelkeznie a felhasználónak annak érdekében, hogy el tudja érni.

Funkciói:

- Felhasználó hozzáadása
- Felhasználó jogainak beállítása
- Felhasználó törlése
- Felhasználó név módosítása
- Jelszó módosítása

The screenshot shows a Windows-style application window titled 'Felhasználók'. It contains a tabbed interface with 'Felhasználó hozzáadása' and 'Felhasználók kezelése'. The first tab is active, showing input fields for 'Felhasználónév:', 'Jelszó:', 'Jelszó újra:', and 'Emlékeztető:', followed by a 'Hozzáad' button. Below these are three unchecked checkboxes for permissions: 'Felhasználók hozzáadása és kezelése', 'Adatbázis módosítása', and 'Teljes hozzáférés az adatbázishoz'. On the right side, there is a list box containing the text 'A', 'admin', and 'Soma'. At the bottom right of the window is a button labeled 'Kijelölt felhasználó törlése'.

btn_hozzaad: Felhasználó hozzáadása.

btn_felhtotol: A listában kijelölt felhasználó törlése.

Adduser_Load: Már regisztrált felhasználók felhasználónevének betöltése a listBox1 és listBox2-be.

btn_permissionsmod: Felhasználó jogainak módosítása.

listBox1_SelectedIndexChanged: Felhasználó adatainak betöltése a textBoxok-ba;

btn_hozzaad szövegének megváltoztatása „Módosítás”-ra.

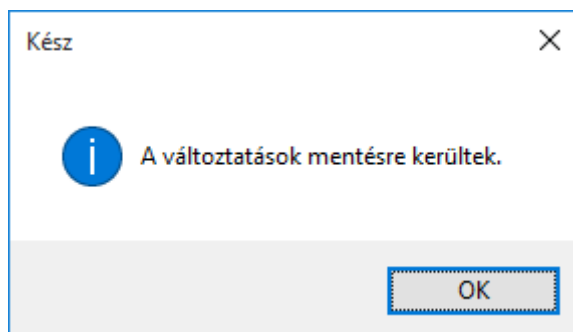
Felhasználó hozzáadása után feljön egy MessageBox és jelzi, hogy a felhasználó sikeresen hozzá lett adva a(z) users táblához és a jogai a(z) usersperm táblához.

1.7.6 Beállítások(Beallitasok.cs)

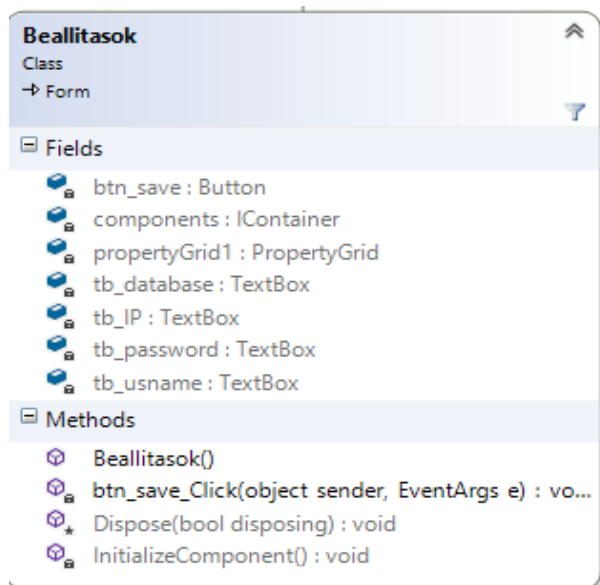


The screenshot shows a Windows-style dialog box titled 'Settings'. It has two tabs: 'Szerver' (Server) and 'Általános' (General). The 'Szerver' tab is selected. Inside the tab, there are four text input fields with labels: 'Szerver IP:', 'Adatbázis:', 'Felhasználónév:', and 'Jelszó:'. Below these fields is a 'Mentés' (Save) button.

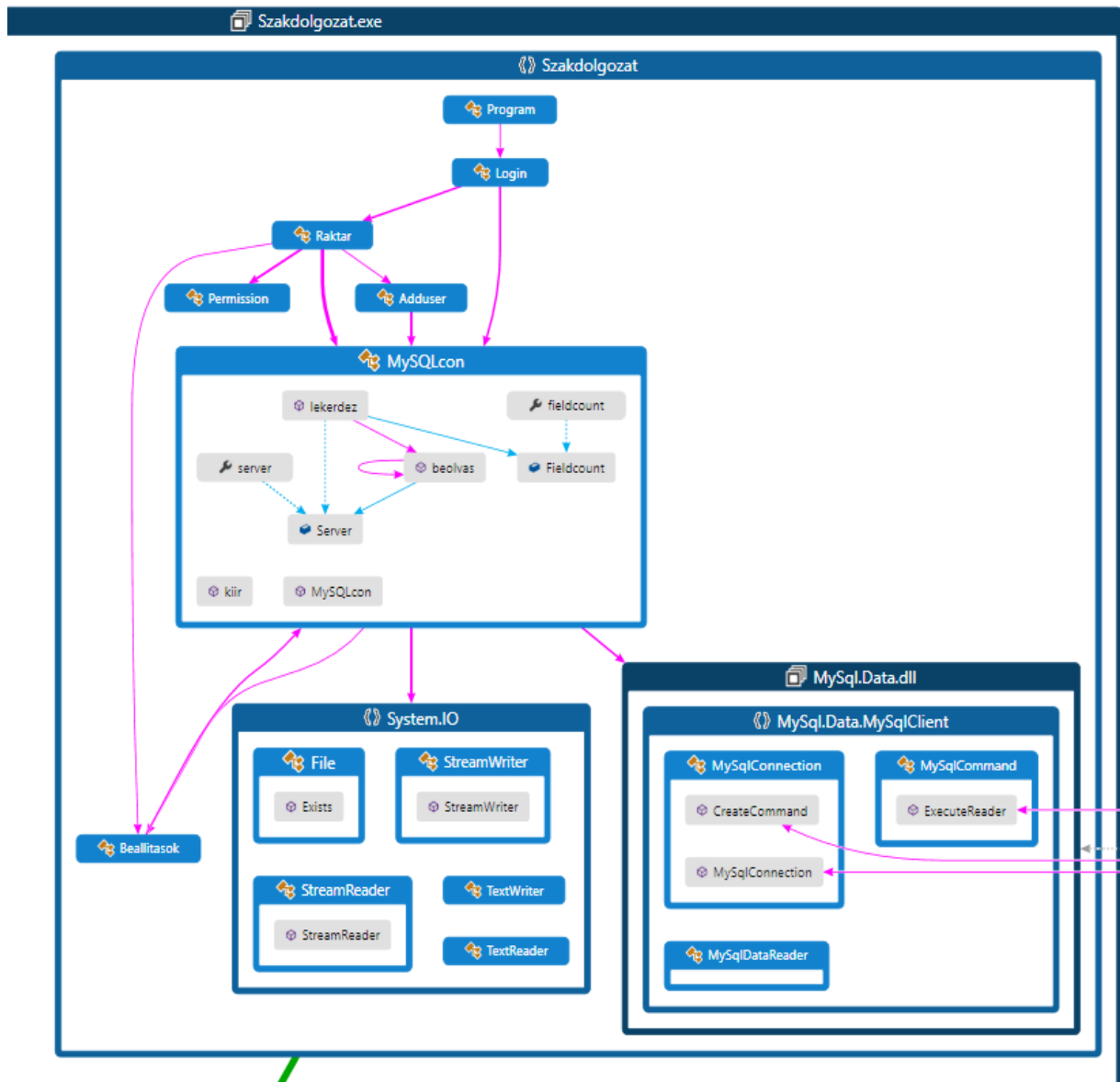
Ezen a formon a MySQL adatbázishoz való csatlakozás adatait tudjuk beállítani. A `btn_save_Click` eseményre meghívja a `MySQLcon` osztályból a `kiir` eljárást, amellyel a szerver adatait kiírja az `srv.dll` fájlba. Ha az esemény sikerrel zajlott le akkor megjelenik egy `MessageBox` és jelzi, hogy a változtatások mentésre kerültek. Ezen Form eléréséhez fullperm jog szükséges.



A tabControl másik fülén a program általános beállításai találhatóak meg, mellyel személyre szabható a teljes program. De csak hozz értő férhet hozzá.



1.8 Együttműködési diagramm



1.9 Konklúzió

Habár a programot még számtalan funkcióval lehetne bővíteni, de én mégis úgy gondolom, hogy összességében a feladat specifikáció azon részének, amely azt tűzi ki célul, hogy egy a valós életnek is megfelelő funkcionalitással és tudással bíró program készítése, mely reprezentálja a körülbelül 1,5 év alatt szerzett tudásomat, eleget tesz.

A program elkészítése közben rengeteg tapasztalatot szereztem, mind programozói gondolkodás mind a valós élet nyújtotta akadályok, feladatok megoldása terén egy ilyen program elkészítése estén, még ha csak részben is. Ugyanakkor az is a tapasztalat része, hogy a program komplexitása miatt nem igazán lehetett látni a végét, még ha nem is volt olyan messze, de kellő idő és energia befektetésével hamar meg lehet lelni a dolgok végét.

2 Felhasználói dokumentáció

2.1 Általános ismertető

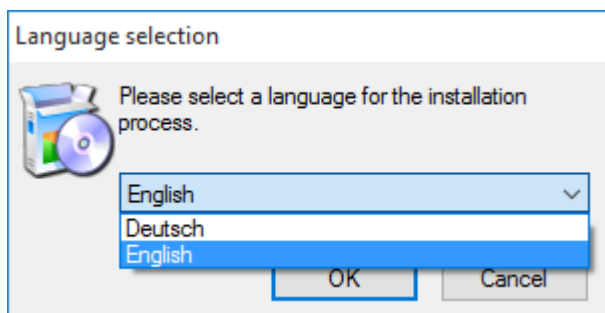
A program egy kisebb számítógépes üzlet vagy üzletlánc árukészletének nyilvántartását könnyíti meg. A program lehetőséget ad arra, hogy a felhasználó gyorsan és könnyedén kereshessen a raktáron lévő termékek között több különböző szempont szerint. Például elég csak a termék nevét bevinni részben, aztán egy Entert nyomni és máris megvan a lehetséges termékek listája, de ha a találati lista túl hosszú lenne vagy csak épp arra kíváncsi a felhasználó, hogy az adott típusú termékből, gyártótól vagy állapotú termékekből mi van raktáron, hány darab vagy mennyi az ára. Új árut kell felvenni? Nem probléma, a program segítségével ezt is gyorsan megoldhatja. Ha pedig nem kell már vagy végleg elfogyott a termék, akkor egyszerűen csak törölheti azt az adatbázisból. Megváltozott a termék ára? Akkor egyszerűen csak írja át a beszerzési árat és a többi változás magától megtörténik majd. Saját árképzési szabályt is könnyedén ki lehet alakítani akár termékenként. A program tudása viszont ennyiben nem merül ki, hiszen van még számos funkciója, ami igen csak hasznos tud lenni, ha raktárkészlet nyilvántartásról van szó.

2.2 Telepítés és rendszerkövetelmény

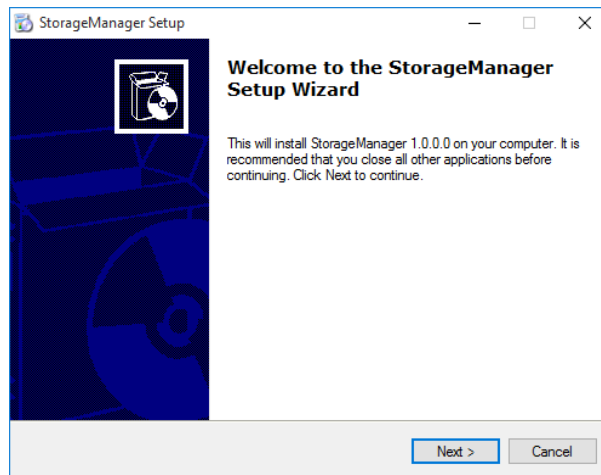
A program futtatásához Windows 7 vagy a Windows újabb verziója kell. 10 MB szabad hely a háttértáron Továbbá telepítve kell legyen a .NET 4.0 vagy a .NET újabb verziója.

A telepítő ikonjára duplán kattintva elindul a telepítő. Majd a telepítő által felajánlott lehetőségek közül válasszuk a nekünk legmegfelelőbbet, ha nem tudja valamelyik lépésről, hogy mi az akkor hagyja meg az alapértelmezett értéket.

1. Lépés: A telepítés nyelvének kiválasztása

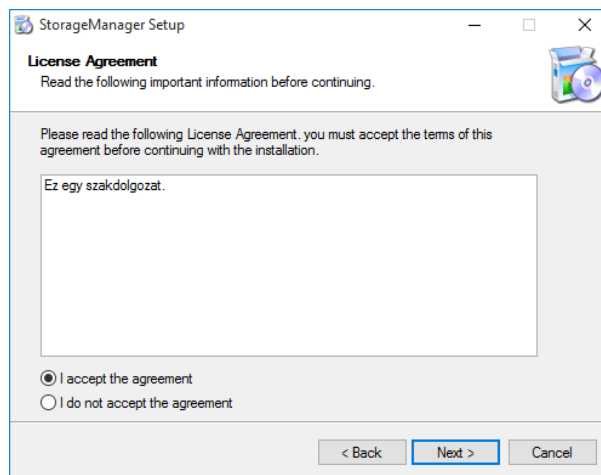


Ezután klikkeljen az OK gombra.



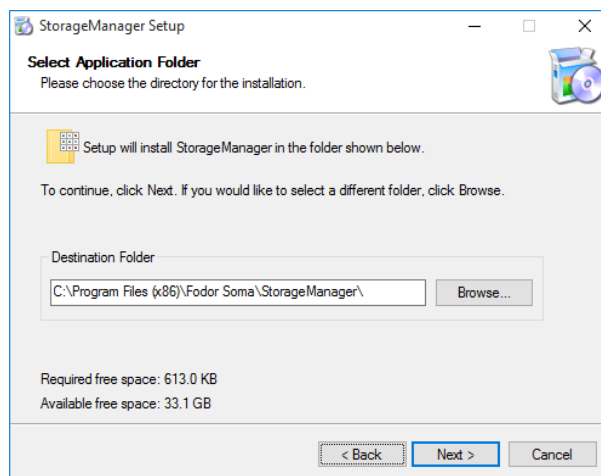
Kattintson a Next gombra.

2. Lépés: Fogadja el a felhasználói feltétteleket.



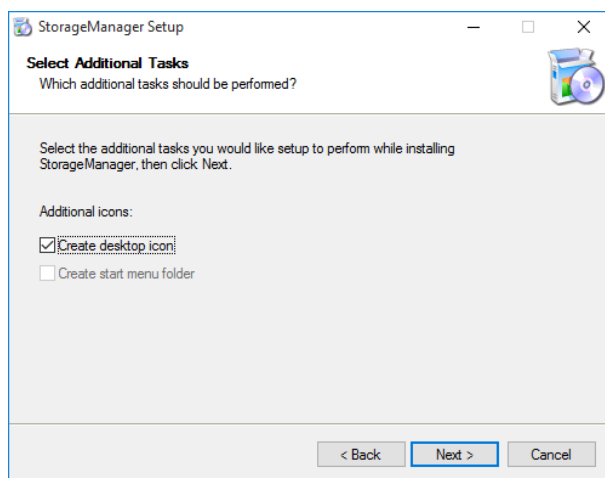
Ezután ismét kattintson a Next gombra.

3. Lépés: Válassza ki a telepítés helyét, a program melyik könyvtárba települjön a gépen.



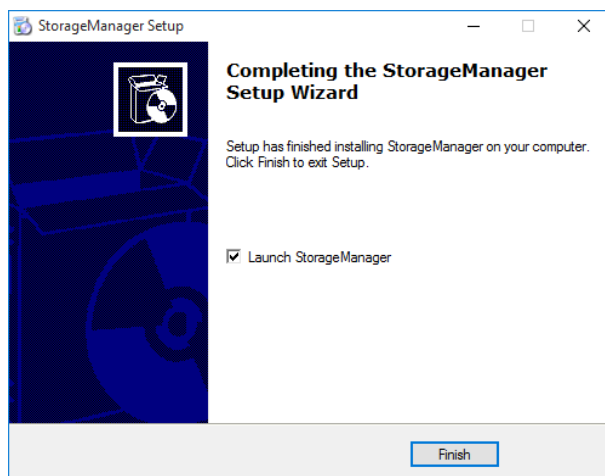
Kattintson a Next gombra, hogy a telepítő tovább lépjen.

4. Lépés: Válassza ki, hogy szeretné e, hogy a telepítő parancsikont hozzon létre az asztalára vagy a Start menüben.



Klikkeljen a Next gombra.

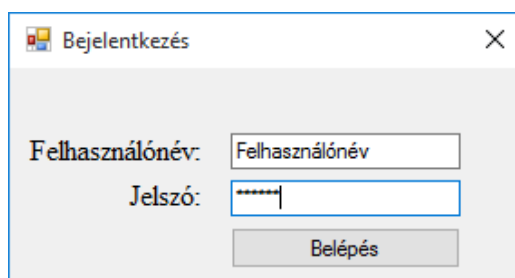
5. Lépés: A telepítés befejezése. Válassza ki, hogy a telepítő elindítsa e a programot a telepítés végén!



Ha eldöntötte, kattintson a Finish gombra, hogy a telepítést befejezze.

2.3 Bejelentkezés

A program megnyitása után az első megjelenő ablak a Bejelentkezés ablak.

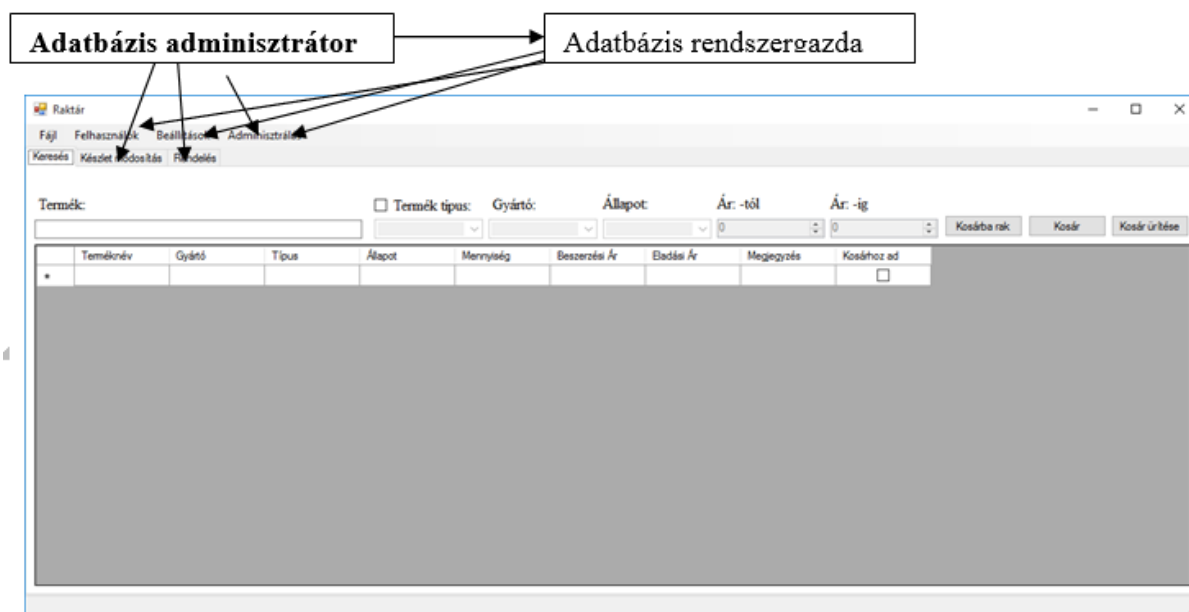


A Bejelentkezés című ablak, amely tartalmazza a felhasználónév és a jelszó beviteli mezőit, valamint a Belépés gombot.

Felhasználónév:

Jelszó:

A mezőket értelem szerűen kitöltve hiteles adatokkal és a Belépés gombra kattintva a program ellenőrzi a felhasználó nevét és jelszavát. Ha helyesek az adatok akkor a felhasználó sikeresen bejelentkezett és a program és funkciói a felhasználó jogainak megfelelően elérhetővé válnak. Ezt a felületet látja egy adatbázis rendszergazda joggal rendelkező felhasználó. Jogok:



A diagram, amely két szerepet mutat: 'Adatbázis adminisztrátor' és 'Adatbázis rendszergazda'. Az 'Adatbázis adminisztrátor' szerephez a 'Raktár' program 'Fájl', 'Felhasználók', 'Beállítások' és 'Adminisztráció' menüi tartoznak. Az 'Adatbázis rendszergazda' szerephez a 'Keresés', 'Készlet módosítás' és 'Függelék' menüi tartoznak.

Adatbázis adminisztrátor

Adatbázis rendszergazda

Raktár

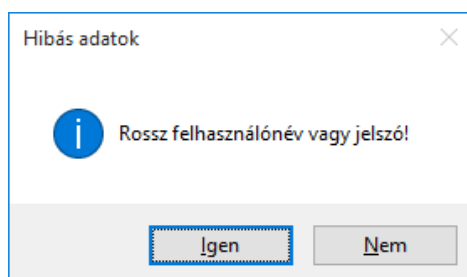
Fájl Felhasználók Beállítások Adminisztráció

Keresés Készlet módosítás Függelék

Termék: ☐ Termék típus: Gyártó: Állapot: Ár: -től: Ár: -ig: Kosárba rak Kosár Kosár ürítése

Terméknév	Gyártó	Típus	Állapot	Mennyiség	Beszerezési Ár	Eladási Ár	Megjegyzés	Kosárhoz ad
*								<input type="checkbox"/>

A többi nem jelölt dologhoz bármelyik felhasználó hozzáférhet. Ha a jelszó, a felhasználónév vagy mind a kettő helytelen akkor ez az ablak jelenik meg. Az OK gombra kattintva az ablak eltűnik majd a bejelentkező ablak ismét elérhetővé válik üres beviteli mezőkkel.



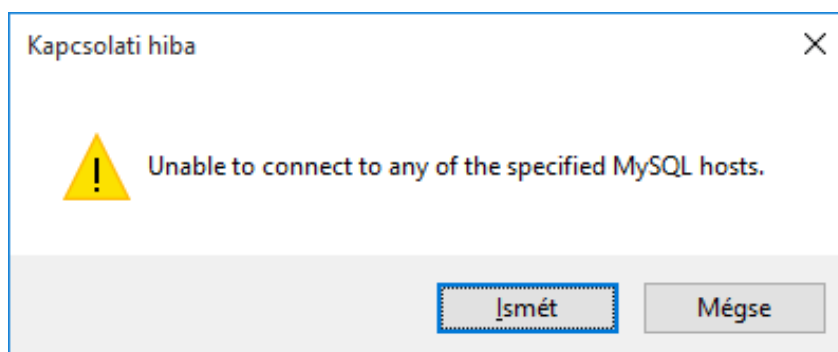
A Hibás adatok című hibajelző ablak, amely közli a rossz felhasználónév vagy jelszó hibaüzenetét, és tartalmazza az Igen és Nem gombokat.

Hibás adatok

i Rossz felhasználónév vagy jelszó!

2.3.1 Hibaüzenet az induláskor

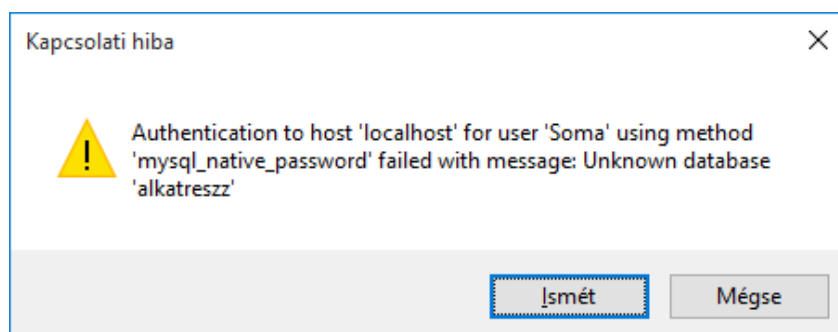
Ha indításkor az alábbi hiba üzenetet kapja akkor az azt jelenti, hogy a program nem tud kapcsolódni a MySQL szerverhez mert nem fut vagy mert rosszul van beállítva a szerver IP címe. Kérjük ellenőrizze internet kapcsolatát, a szerver IP címének beállítását vagy a helyi gépen a MySQL szerver futását és elérhetőségét. Az igen gombra kattintva újra megadhatja az adatbázis eléréséhez szükséges adatokat.



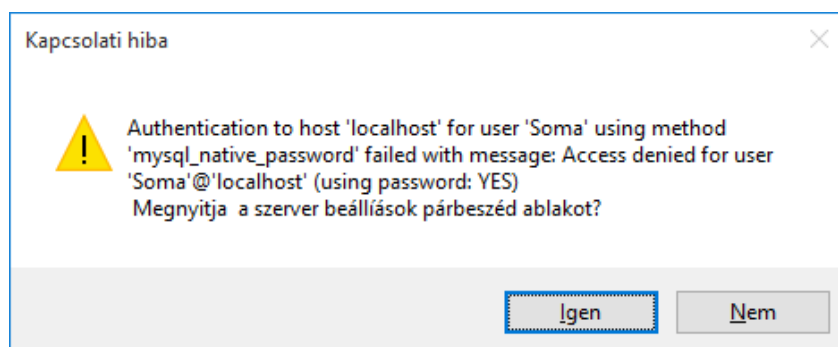
Az Ismét gombra a felhasználó újrapróbálkozhat a bejelentkezéssel.

A mégse gombra kattintva az alkalmazás kilép.

Hibás adatbázis név hibaüzenet. A hiba elhárítása érdekében kérjük helyesbítse az adatbázis nevét. Az igenre kattintva javíthatja.



Hibás adatbázis hozzáférési jelszó vagy felhasználónév hibás. Az igenre kattintva javíthatja.



2.4 A program használta (Alap felhasználó)

Készletfrissítés

Fájl Beállítások

Keresés

Termék: ☐ Termék típus: Gyártó: Állapot: Ár: -tól: Ár: -ig:

	Terméknév	Gyártó	Típus	Állapot	Mennyiség	Beszerzési Ár	Eladási Ár	Megjegyzés	Kosárhoz ad
*									<input type="checkbox"/>

A sikeres bejelentkezés után ez a felhasználói felület jelenik meg. Ezt a felületet bármelyik bejelentkezett felhasználó elérheti. A kereséshez a Termék felirat alatti szöveges mezőbe be kell írni a keresett termék nevét (Tipp: ha a termék nevét csak részben ismerjük, akkor használjuk a % jelet a szó elején és a végén.). Ha a termékről tudunk konkrétabb dolgokat is, mint például típus szerinti besorolása, gyártója vagy ára (megtől meddig intervallumon helyezkedik el), akkor pipáljuk be a „Termék típusa:” szöveg melletti mezőt, ezzel a részletes keresést aktiváljuk.

Készletfrissítés

Fájl Beállítások

Keresés

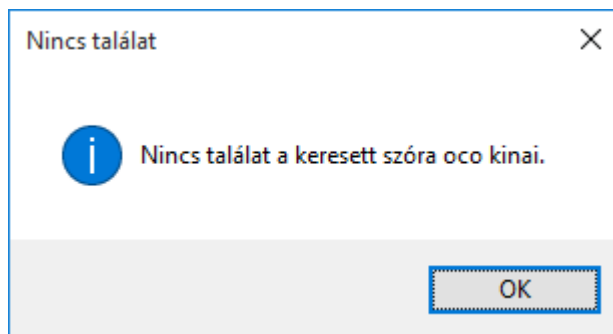
Termék: ☒ Termék típus: Gyártó: Állapot: Ár: -tól: Ár: -ig:

	Terméknév	Gyártó	Típus	Állapot	Mennyiség	Beszerzési Ár	Eladási Ár	Megjegyzés	Kosárhoz ad
▶	D-Link SOHO	D-Link	Hálózati eszköz	ÚJ	10	10000	13335	Wi-Fi router	<input type="checkbox"/>
	D-Link Switch 10...	D-Link	Hálózati eszköz	BONTOTT	5	2000	2667	D-Link 10/100 s...	<input type="checkbox"/>
	D-Link DSP-W215	D-Link	Hálózati eszköz	BONTOTT	2	9000	12001.5	Hálózati okoseszköz	<input type="checkbox"/>
	Linksys PSUS4 P...	Linksys	Hálózati eszköz	HASZNÁLT	1	1500	2000.25	nincs	<input type="checkbox"/>
*	TP-Link TL-WR8...	TP-Link	Hálózati eszköz	HASZNÁLT	4	3000	4000.5	Wi-Fi router	<input type="checkbox"/>

Az aktiválódott legördülő menüből válasszuk ki a keresett terméknek megfelelő sort és nyomjuk le az Entert. Az Enter lenyomására a kiválasztott legördülő menüben kiválasztott értéknek megfelelő típusú, gyártójú vagy állapotú termékeket listázza ki. Az „Ár: -tól” és „Ár: -ig” mezőkbe bevitt értékek a termék árának alsó és felső határát adhatjuk meg, s ezzel is

szűkítve a találati listát. Ha nem akarunk meghatározni a termék minimum árát, akkor hagyjuk alap értéken és csak az „Ár: -ig” mező értékét növeljük meg. A keresést a „Termék:” felirat alatti szöveges mezőbe kattintás után Entert nyomunk vagy az „Ár: -ig” mezőbe kattintás után nyomunk Entert. FIGYLEM: RÉSZLETES KERESÉSKOR MINDEN MEZŐT KI KELL TÖLTENI.

Ha nincs találat a keresett szóra, akkor az alábbi üzenetablak jelenik meg.



Kosár funkció: a találati lista jobb oldalán lévő oszlopban a kosárba rakni kívánt termékek melletti pipálós mezőt kipipáljuk majd, ha minden kosárba rakni kívánt terméket bejelöltünk, akkor a „Kosárba rak” gombra nyomva az betölti a kosárba és ott tárolja addig míg a „Kosár ürítése” gombra nem nyomunk. A kosár tartalma felhasználónként tárolódik, azaz minden felhasználónak saját kosara van és egymás kosarának tartamát nem látják. A „Kosár” gombra nyomva az aktuálisan bejelentkezett felhasználó kosara jelenik meg a találati lista helyén.

2.5 Program használata (Adatbázis adminisztrátorként)

Készletfrissítés

Fájl Felhasználók Beállítások Adminisztrálás

Keresés Készlet módosítás Rendelés

Termék ID: Termék: ☐ Termék típus: Gyártó: Állapot: Készlet: Készlet min: Ár:

Leírás: Hozzáad

Keresés

Töröl

	Termékazonosító	Terméknév	Gyártó	Típus	Állapot	Beszerzési Ár	ÁFA kulcs	Minimum Készlet	Mennyiség	Megjegyzés	Töröl
▶	5	D-Link SOHO	D-Link	Hálózati eszköz	Új	10	0,27	10000	13335	Wi-Fi router	<input type="checkbox"/>
	1	D-Link Switch 10...	D-Link	Hálózati eszköz	BONTOTT	5	0,27	2000	2667	D-Link 10/100 s...	<input type="checkbox"/>
	4	D-Link DSP-W215	D-Link	Hálózati eszköz	BONTOTT	2	0,27	9000	12001,5	Hálózati okosezkö...	<input type="checkbox"/>
	6	Lenovo X220	Linksys	Laptop/Note...	HASZNÁLT	6	0,27	89900	119881,65	8GB RAM, Core i...	<input type="checkbox"/>
	2	Linksys PSUS4 P...	Linksys	Hálózati eszköz	HASZNÁLT	1	0,27	1500	2000,25	nincs	<input type="checkbox"/>
*	3	TP-Link TL-WR8...	TP-Link	Hálózati eszköz	HASZNÁLT	4	0,27	3000	4000,5	Wi-Fi router	<input type="checkbox"/>

Az adatbázis adminisztrátor a „Készlet módosítása” tab Fülre kattintva átlép a készlet módosítása és frissítése felületre, ahol új árút tölthet fel vagy az adatbázisban lévő lévő termékeket módosíthatja.

Új termék hozzáadása: A szöveges bevitelmezők megfelelő kitöltése után az ablak jobb szélé felé lévő „Hozzáad” gombra kattintva adhatja hozzá az új terméket az adatbázishoz. Ha a „Leírás” mező üresen hagyása esetén alapértelmezett értéke „nincs” lesz és ezt a tölti fela az adatbázisba is a leírás helyére.

Termékek módosítása: A „Termék” szöveges beviteli mezőbe beírja keresett termék nevét vagy a „Termék típus” melletti jelölő négyzetet bejelölve a részletes kereséshez. A mezők kitöltése után a „Keresés” gombra kattintva a találati lista megjelenik és a találati lista módosítandó mezőjébe kattintva szerkesztheti a mező értékét. A szerkesztés végeztével a módosított adatok automatikusan feltöltődnek az adatbázisba.

Termék törlése: A találati lista jobb oldali oszlopában a jelölő négyzetet bejelölve az adott termék mellett és a „Töröl” gombra kattintva a kijelölt termékek törlődnek az adatbázisból.

Rendelés leadás: Termék nevének, típusának, gyártójának, mennyiségének és árának megadása után a „Rendelési listához ad” gombra kattintva a rendelési listához adja a terméket. A rendelési listába „termék” mező kitöltése és a „Keresés” gomb lenyomása után a rendelési listában megjelennek a találatok és a találati list jobb szélső törlés oszlopában a „Törlés” oszlopban a gombra kattintva törlődik az adot sor a rendelési listából. A rendelés véglegesítése és elküldése a „Rendelsé véglegesítése” gomb lenyomásával történik.

2.6 Program használata (Adatbázis rendszergazda)

Terméknév	Gyártó	Típus	Állapot	Mennyiség	Beszerzési Ár	Eladási Ár	Megjegyzés	Kosárhoz ad
*								<input type="checkbox"/>

Felhasználók kezelése: A fő ablakban a „Felhasználók” menüre kattintva azon belül a „Felhasználók kezelése” menüre kattintva elő jön a „Felhasználók” ablak.

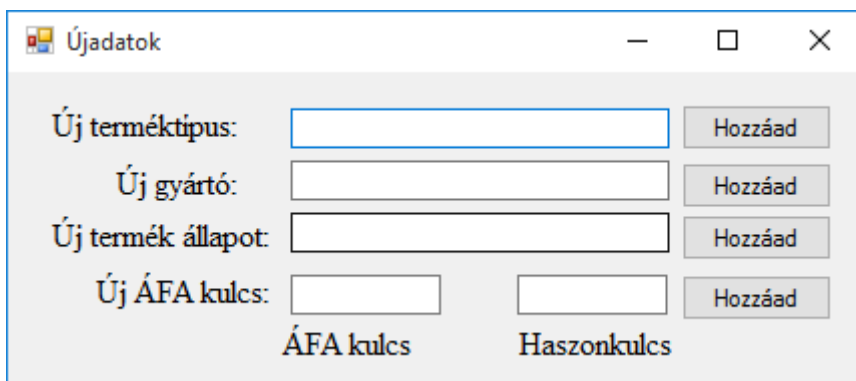
The screenshot shows a window titled 'Felhasználók' with two tabs: 'Felhasználó hozzáadása' and 'Felhasználók kezelése'. The 'Felhasználók kezelése' tab is active. It contains a form for adding a new user with fields for 'Felhasználónév:' (Username), 'Jelszó:' (Password), 'Jelszó újra:' (Password again), and 'Emlékeztető:' (Remember me). Below these fields is a 'Hozzáad' (Add) button. To the right of the form is a list box containing the text 'Soma', 'A', and 'admin'. Below the list box is a 'Kijelölt felhasználó törlése' (Delete selected user) button. At the bottom left, there are three checkboxes: 'Felhasználók hozzáadása és kezelése', 'Adatbázis módosítása', and 'Teljes hozzáférés az adatbázishoz'.

Felhasználó hozzáadása: Mezők értelem szerű kitöltése után és a jogosultságok megadása után hozzáad gombra kattintva a felhasználót hozzáadja felhasználók listájához. Ha a jelszók nem egyeznek, akkor a program egy üzenet ablakban jelzi, ekkor újra be kell írnia az adatokat.

Felhasználó törlése: A listában a felhasználó nevére kattintva a „Kijelölt felhasználó törlése” gombra aktiválódik majd arra kattintva felhasználó törlődik az adatbázisból. Adatok módosítása: A listában a felhasználó nevére kattintva az adatai megjelennek a megfelelő mezőkben és a „Hozzáad” gomb szövege „Módosítás”-ra változik.

The screenshot shows the same 'Felhasználók' window, but the 'Felhasználók kezelése' tab is now active. The 'Hozzáad' button has been replaced by a 'Jogosultság módosítása' (Modify permissions) button. The list box on the right now shows 'Soma', 'A', and 'admin'. Below the list box, there are two labels: 'Utolsó bejelentkezés:' (Last login) and 'Regisztrálás dátuma:' (Registration date). At the bottom left, there are three checkboxes: 'Felhasználók hozzáadása és kezelése', 'Adatbázis módosítása', and 'Teljes hozzáférés az adatbázishoz'.

Felhasználók kezelése fül: A listából a kiválasztani kívánt felhasználó nevére kattintva megjelennek a felhasználó jogai, utolsó bejelentkezés és a regisztrálás dátuma. A jogok beállítása után a jogok módosítása gombra kattintva véglegesíti a beállításokat.



Új adatok

Új terméktípus: Hozzáad

Új gyártó: Hozzáad

Új termék állapot: Hozzáad

Új ÁFA kulcs: Hozzáad

ÁFA kulcs Haszonkulcs

Új mezőket adhatunk a terméktípusok, a gyártók, az állapotok és az ÁFA kulcsok gyűjteményével a mező mellett lévő „Hozzáad” gombra kattintva.



Beállítások

Szerver Általános

Szerver IP:

Adatbázis:

Felhasználónév:

Jelszó:

Mentés

Ezen ablak megnyitása a „Beállítások” menüből érhető el.

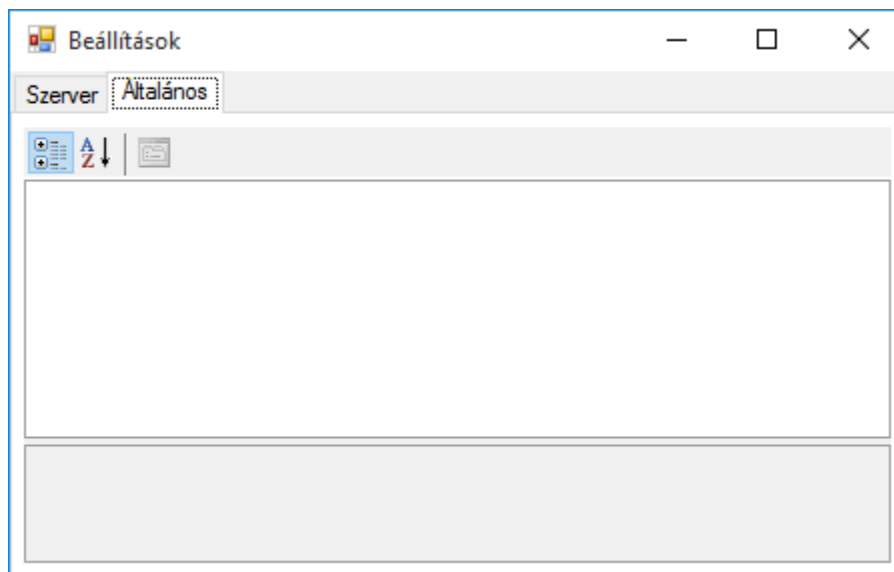
Szerver IP: A MySQL szerver IP címet (Internetes elérési címét lehet megadni).

Adatbázis: A MySQL adatbázis nevét lehet megadni a miben az adatok tárolódnak.

Felhasználónév: A MySQL szerverre való bejelentkezéshez használt felhasználónév.

Jelszó: A MySQL adatbázis felhasználónevéhez tartozó jelszó.

Mentés: Erre a gombra kattintva az adatok eltárolódnak a gépen és további módosításig úgy is maradnak.



Az általános fülön lehet beállítani a program egyes komponenseinek adatait. Példáil mi jelenjen meg a találati listán.

2.7 Irodalomjegyzék

Könyvek

Tanuljuk meg a Visual C# 2008 használatát 24 óra alatt (James Foxall)

Tanuljuk meg a MySQL használatát 24 óra alatt (Julie C. Meloni)

Internet

<https://devportal.hu/download/E-bookok/csharp%20jegyzet/csharp.pdf>

<https://msdn.microsoft.com/hu-hu>

<http://www.mysql.com/>

<http://prog.hu/>

<http://stackoverflow.com/>

3 Mellékletek

Telepítő mappa

Ez a mappa tartalmazza a program telepítéséhez szükséges fájlokat.

Adatbázis mappa

Ez a mappa tartalmazza a teszt adatbázist (teszt.sql fájl), és az üres adatbázist (ures.sql fájl) is.

Források mappa

Ez a mappa tartalmazza a teljes forrásprogramot. Továbbá a wamp, mysqlconnector, .net telepítőket.

Dokumentáció mappa

Ez a mappa tartalmazza a dokumentációt pdf formátumban