

Pannon Egyetem
Műszaki Informatikai Kar
Matematikai Tanszék
Gazdaságinformatikus BSC

SZAKDOLGOZAT

Antikvár könyvesbolti alkalmazásfejlesztés C# nyelven

Horeczky Tünde

Témavezető: Süle Péter

2022



PANNON EGYETEM

MŰSZAKI INFORMATIKAI KAR

Gazdaságinformatikus BSc szak

Veszprém, 2022. március 23.

SZAKDOLGOZAT TÉMAKIÍRÁS

Horeczky Tünde

Gazdaságinformatikus BSc szakos hallgató részére

Antikvár könyvesbolti alkalmazás fejlesztése C# nyelven


Témavezető: Süle Péter, mesteroktató

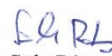
A feladat leírása:

A szakdolgozat célja egy antikvár könyvesbolt eladásait és nyilvántartását kezelő alkalmazás fejlesztése C# nyelven, amely számos funkcióval segíti és könnyíti meg a felhasználók, ezen rendszerrel a bolt eladóinak, pénztárosainak munkáját. A felhasználó képes legyen rögzíteni az üzlet által felvásárolt könyveket egy adatbázisba, illetve azokat törölni, hibás bevétel esetén módosítani a hozzá tartozó adatokat. Az adatok felvitele során a rendszerben kerüljön rögzítésre az adott könyv vásárlási értéke, valamint kerüljön meghatározásra annak eladási ára az előre meghatározott ár és mértékének megfelelően. Az alkalmazás egyik funkciójaként kerüljön beépítésre egy törzsvásárlók rögzítésére, kezelésére szolgáló funkció. Az antikvárium rendszerében tárolt könyvek megvásárlásához hozzon létre egy törzsvásárlói pontrendszert, amely kapcsán gyűjtött pontok a későbbi vásárlás során felhasználhatók legyenek és ezáltal kedvezményes vásárlási lehetőséget biztosítanak a regisztrált ügyfeleknek. A felhasználói elégedettség növelése érdekében kerüljön beépítésre egy kívánságlista funkció is, mely visszajelzést biztosít az antikvárium számára, hogy mely könyvek iránt volna még kereslet, melyeket lenne érdemes beszereznie az üzletnek. Egy-egy ilyen a felhasználó számára keresett új beszerzés beérkezéséről a regisztrált felhasználó kapjon visszajelzést e-mail formájában. A fenti funkciók mellett a felhasználóknak legyen lehetősége kimutatások segítségével megtekinteni az adott időszakokra vonatkozóan a kiadásokat és bevételeket.

Feladatkiírás:

- Ismertesse a szakdolgozatában az alkalmazás fejlesztése alatt használt technológiákat!
- Hasonlítsa össze más, az alkalmazáshoz részben hasonló lehetőségeket!
- Készítse el az alkalmazást, a kiválasztott technológia segítségével!
- Tesztelje a rendszert és értékelje az elért eredményt!
- Számoljon be a felmerült nehézségekről, illetve esetleges további tervekről, amivel a rendszert kibővítené a jövőben!


Dr. Heckl István
egyetemi docens
szakfelelős


Süle Péter
mesteroktató
témavezető

Hallgatói nyilatkozat

Alulírott Horeczky Tünde hallgató kijelentem, hogy a dolgozatot a Pannon Egyetem Matematika Tanszékén készítettem a gazdaságinformatikus végzettség megszerzése érdekében.

Kijelentem, hogy a dolgozatban lévő érdemi rész saját munkám eredménye, az érdemi részen kívül csak a hivatkozott forrásokat (szakirodalom, eszközök stb.) használtam fel.

Tudomásul veszem, hogy a dolgozatban foglalt eredményeket a Pannon Egyetem, valamint a feladatot kiíró szervezeti egység saját céljaira szabadon felhasználhatja.

Dátum: Veszprém, 2022.05.10.



.....
Horeczky Tünde

Témavezetői nyilatkozat

Alulírott Süle Péter témavezető kijelentem, hogy a dolgozatot Horeczky Tünde a Pannon Egyetem Matematika Tanszékén készítette gazdaságinformatikus végzettség megszerzése érdekében.

Kijelentem, hogy a dolgozat védelemre bocsátását engedélyezem.

Dátum: Veszprém, 2022.05.10.



.....
Süle Péter

Köszönetnyilvánítás

Szeretném megköszönni páromnak, családomnak, barátaimnak a támogatást és a biztatás, hogy végig mellettem álltak, valamint segítséget nyújtottak amiben tudtak.

Köszönettel tartozom témavezetőmnek, Süle Péternek, aki szakértelmével, konzultációk során adott tanácsaival végig segített a szakdolgozatom elkészítésében. Illetve, hogy az általa tanított tantárggyal bevezetett a C# programozás világába.

Hálával tartozom tanáraimnak, valamint szak- és hallgató társaimnak, akik segítettek a szak elvégzésében.

Tartalmi összefoglaló

Szakdolgozatom témája egy antikvárium nyilvántartó rendszerének fejlesztése és tesztelése C# nyelven.

A dolgozatban bemutatásra kerülnek az alkalmazás elkészítéséhez használatos technikák. Továbbá ismertetem a program főbb funkcióit lépésről lépésre, az adatbázisismertetésével együttesen. Ezenfelül a tesztelésről ejtek szót, valamint lehetséges tovább fejlesztési lehetőségekről.

Az elkészült alkalmazás segít egy átláthatóbb képet kapni a felhasználónak az üzletben fellelhető könyvekkel kapcsolatosan.

Kulcsszavak: Antikvárium, C#, adatbázis, könyv, tesztelés, fejlesztés

Abstract

The topic of my dissertation is the development and testing of an antique shop registration system on C# language.

The techniques used to create the application are presented in the dissertation. The functions of the program are explained step by step, together with the description of the database. In addition, I will talk about testing and possible improvements.

The completed application will help the user to get a more transparent picture of the books in the shop.

Keywords: Antique shop, C#, database, book, testing, development

Tartalomjegyzék

Jelölésjegyzék	10
1. Bevezetés.....	11
1.1. Választott technológiák	11
1.1.1. A nyelv és a környezet.....	12
1.2. Antikvár könyvek és könyvesboltok	13
1.3. Összehasonlítás.....	14
1.3.1. Assister	14
1.3.2. További alkalmazások	15
2. Felhasználói dokumentáció	17
2.1. Főablak	17
2.2. Eladás.....	18
2.2.1. Könyv megjelenítések	18
2.2.2. Könyv kosárba helyezése	19
2.2.3. Könyv törlése a kosárból	20
2.2.4. Törzsvásárlói pontok jóváírása.....	20
2.2.5. Törzsvásárlói pontok levonása	21
2.2.6. Könyvkedvezmény alkalmazása.....	22
2.2.7. Végösszegkedvezmény alkalmazása	23
2.2.8. Könyvek eladása.....	23
2.2.9. Vonalkódos keresés	24
2.3. Törzsvásárlók.....	25
2.3.1. Törzsvásárlói regisztrálás	26
2.3.2. Törzsvásárló megjelenítése.....	27
2.3.3. Törzsvásárló adatainak módosítása	27
2.3.4. Törzsvásárló törlése	28
2.4. Könyvkezelő	29
2.4.1. Új könyv hozzáadása	29
2.4.2. Könyv keresés	30
2.4.3. Könyv adatainak módosítása	31
2.4.4. Könyv törlése.....	32
2.5. Kívánság lista.....	33
2.5.1. Új kívánság hozzáadása.....	33

2.5.2.	Kívánság megjelenítése	34
2.5.3.	Kívánsághoz tartozó adatok módosítása.....	35
2.5.4.	Kívánság törlése	35
2.6.	Kimutatások.....	36
2.6.1.	Teljes kimutatás adott időszakra.....	37
2.6.2.	Bevételek kimutatása adott időszakra.....	37
2.6.3.	Kiadások kimutatása adott időszakra	38
2.7.	E-mail küldési funkciók.....	38
2.7.1.	Törzsvásárlói regisztráció esetén.....	39
2.7.2.	Kívánság listában szereplő könyv beérkezése alkalmával	39
2.8.	Automatizált funkciók	39
2.8.1.	Előző éves pontok felhasználhatóságával kapcsolatban.....	39
2.8.2.	Törzsvásárlói pontok év elejei átírása	40
2.8.3.	Előző éves pontok törlése	40
2.9.	Menü.....	41
2.9.1.	Fájl.....	41
2.9.2.	Súgó	41
3.	Fejlesztői dokumentáció.....	42
3.1.	Fő ablak	42
3.2.	Vonalkódolvasó	43
3.3.	Törzsvásárlói kód.....	44
3.4.	E-mail küldési funkciók.....	46
3.5.	Automatizált funkciók	48
3.6.	Menü.....	49
3.7.	Formázás.....	50
4.	Az adatbázis	51
4.1.	Adatbázis létrehozása, tervezése	51
4.1.1.	MSSQL.....	52
4.2.	Felépítése	55
5.	Tesztelés.....	57
5.1.	Az adatbázis tesztelése	57
5.2.	A program tesztelése.....	58
6.	A dolgozat írása alatt felmerülő nehézségek.....	60
6.1.	Gmail által kevésbé biztonságosnak vélt bejelentkezési technika.....	60

6.2.	Új Gmail jelszó esetén a rendszerben is frissítést kell végezni	60
6.3.	Adatbázis elérése klónozás esetén	61
7.	Jövőbeli lehetőségek	62
7.1.	Vonalkód.....	62
7.2.	Törzsvásárlói pont levonás	62
7.3.	Könyv kedvezmény	62
7.4.	Áthelyezés más platformra	63
7.5.	Bejelentkezés	63
7.6.	Automatikus e-mail javítása	63
7.7.	Pont törlés, illetve átírás javítása	64
7.8.	Kívánság lista →Törzsvásárlói kód	64
7.9.	Számlázási lehetőség	64
8.	Összefoglalás	66
	Irodalomjegyzék.....	67
	Mellékletek	68
	Ábrajegyzék.....	69
	Táblázatjegyzék	70

Jelölésjegyzék

- CLI: Common Language Infrastructure
- XML: Extensible Markup Language
- HTML: HyperText Markup Language
- SOAP: Simple Object Access Protocol
- CLS: Common Language Specification
- winmd: Windows RT Class Library and Runtime Components

1. Bevezetés

A szakdolgozatom témáját adó program egy antikvár könyvesbolt nyilvántartó rendszere, amely az üzletben fellelhető könyveket, a törzsvásárlókat, kívánságokat tárolja. Emellett a rendszer képes kezelni az eladásokat, valamint kimutatásokat is készíthetünk vele adott időszakokra. A rendszer célja, hogy minél átláthatóbban és könnyebben tudja a felhasználó kezelni a könyveket, illetve az eladásokat.

Egy nyilvántartó rendszer nem csak adatok tárolására szolgál, hanem ezen adatok kezelésére, megjelenítésére is képes, valamint műveleteket tudunk rajta végre hajtani.

A programom fő profiljának tekinthető az adatbázis kezelése a megírt funkciók használatával. Ezekkel a funkciókkal bővíthetjük, módosíthatjuk, törölhetjük az adatbázisban lévő adatokat, illetve lekérdezéseket is végrehajthatunk. Ezek a funkciók mind a könyvek, a törzsvásárlók, kívánságok, bevétel, illetve a kiadások kezelésére is szolgálnak.

A másik fontos része a programnak az eladások kezelése. Az eladások kezeléséből eredően tudunk kimutatásokat készíteni.

A szakdolgozatom témája onnan ered, hogy évek óta olvasok szinte napi szinten. Sokszor találok olyan könyveket az interneten, amelyek leírás alapján érdekelnek, de sajnos nem fellelhetőek már új formában, ezért antikváriumokban kell keresni őket. A könyvek iránti szeretetem hozta magával ezt a témaválasztást. A nyelv választása az egyetemi éveim alatt tanultak alapján történt. Gazdaságinformatikusként a fő nyelvi irány az évek alatt a Java programozási nyelv volt, viszont egy kötelezően választható tárgy elvégzése alatt ismerekedtem meg a C# nyelvvel. Ez a nyelv volt számomra a legegyszerűbben megérthető és átlátható.

A másik ok amiért a szakdolgozatom témája egy nyilvántartó rendszer az az volt, hogy tanulmányaim alatt az informatika terén az adatbázis kezelés volt a kedvenc irányom. Mindig is ezt éreztem magamhoz a legközelebb.

1.1. Választott technológiák

A következő pár oldalon azon technikákat és megoldási módokat ismertetem, amelyeket a rendszer elkészítése során használtam.

1.1.1. A nyelv és a környezet

A C# programozási nyelvvel és a .NET keretrendszerrel egy kötelezően választható tárgy keretein belül találkoztam tanulmányaim alatt. Számomra ez a nyelv és környezet volt a legátláthatóbb és legkönnyebben megérthető, ezért is fogott meg annyira, hogy úgy döntöttem szakdolgozatomat ezek felhasználásának segítségével írom meg.

A C# nyelv a 2002-ben megjelent Visual Studio.NET programcsomag részeként látott napvilágot. A nyelv elődjének a C++ programozást tekinthetjük, hiszen szintaktikájában és szerkezeti felépítésében, ehhez a nyelvhez áll a legközelebb.

Annak a problémának a kiküszöbölésére hozták létre a C# programnyelvet, hogy a C és C++ nyelveken sajnos gyakran hosszabb fejlesztési időre van szüksége, mint más nyelveknek.

Az általam választott nyelv objektumorientált, kényelmes és gyors lehetőséget ad a .NET keretrendszerben történő alkalmazás fejlesztéshez.

Ezen nyelv és keretrendszer alapja a CLI. [1]

1.1.1.1. A nyelv jellemzői

A C# a .NET környezet fő nyelve. Kifejezetten ehhez a keretrendszerhez tervezték meg. A fejlesztőknek a C++ hatékonyságát és a Visual Basic fejlesztés egyszerűségét és gyorsaságát adták össze az eszközben.

A nyelv Neumann-elvű. Rendszer méretű programok megírására is alkalmazható. Több modulból, illetve fájlból álló program. Ezeknek a szerkezete azonos.

A rendszerben a pontosvessző (;) használatával zárunk le egy utasítást. Minden változó létrehozásánál azt deklarálnunk kell. Az elnevezésekben akár ékezetes karaktereket is használhatóak, valamint megkülönböztetésre kerülnek a kis- és nagybetűk is.

Érték és referencia típusú változókat használ. Nincsenek mutatóhasználatok, így a vektorhasználat biztonságosabb. A függvényeket egymásban ágyazva nem lehet definiálni. A függvénytípolimorfizmus a rendszerben engedélyezett. A nyelv delegáltakat és eseményeket használ.

Újfajta operátorok is vannak a rendszerben, amely máshol nem feltételül megtalálható. Például az `is` operátor, amely egy objektumnak ellenőrzi le a típusát (`x is int`), az `as` operátor pedig a bal oldalon található operandust, a jobb oldalon található típusra konvertálja. Ez a hagyományos konverzióval szemben nem generál kivételt!

Privát, illetve statikus konstruktorok használata, kivételkezelések jellemzik. Párhuzamosan végrehajtható szálak definiálhatósága.

Ezeket a jellemzőket éreztem a legfontosabbnak, de ezen felül még rengeteg jellemzővel bír a nyelv. [1]

1.1.1.2. A .NET környezet áttekintése

2002-ben jelent meg a .NET környezet a Microsoft legfrissebb fejlesztőeszközeként. Az eszköz a Visual Studio.NET nevet arról kapta, hogy ezzel akartak utalni a változtatásokra, illetve a hálózati munkák integrálására. Jelenleg a 2022-es frissítés a legújabb verziója a rendszernek. Én személy szerint a 2019-es verziót használtam a fejlesztés során és a rendszer befejezése előtt se álltam át a 2022-es verzióra.

A környezet telepítéséhez először a keretrendszert kell telepítenünk. Ennek legfontosabb erői: A webszabványok használata, mint az XML, HTML, és a SOAP. Univerzális alkalmazási modell használata. A keretrendszer kompatibilitását a CLS definiálja. Típus használatai minden nyelven ugyanazok. Minden osztály rendelkezésre áll. [1]

1.2. Antikvár könyvek és könyvesboltok

Ha azt halljuk, hogy antikvárium, akkor egyből régi megkopott könyveket, újságokat, térképeket, illetve más írott dolgokat árusító bolt vagy üzlet jut eszünkben, pedig egy antikvárium, ennél több. A régi dolgok árusítása mellett irományokat is árusít egy antikvárkönyvtár, csak ezek már nem újak, hanem a legtöbb esetben használtak, de megkímélt állapotuknak köszönhetően értékesíthetőek.

Manapság rengetegen azért nem olvasnak, mert mint minden más, így az új könyvek ára is az egekbe szökött már. Egy híresebb író vagy költő köteteit több ezer forintért tudjuk megvásárolni könyvesboltokban. Azok mellett, akik különleges régi könyveket keresnek, amiket esetleg már nem lehet megvásárolni újonnan, mellettük azon személyek, akiknek erre nem telik, illetve nem szeretnének ekkora összegeket kiadni egy könyvért vagy írásos jegyzetért csak azért, hogy egyszer elolvassák azt majd félre rakják, nekik lehet egy jó döntés egy antikváriumba betérni. Hiszen ezeken a helyek sokszor fél áron vagy akár annál olcsóbban is megtudják vásárolni a kívánságuknak megfelelő néha tökéletes állapotú köteteket.

Amellett, hogy az antikváriumokban ilyen könyvekre lelhetünk, akár magánszemélyként el is adhatunk efféle műveket ezeken a helyeken. Általában csak olyan kiadványokat vesznek át ezen üzletek, amelyek értékesíthető állapotban vannak. Nem csak ilyen módon vásárolnak fel árukat az antikváriumok, hanem magán könyvtáraktól, illetve hagyaték tárgyát képző gyűjteményeket is beszereznek.

Az antikváriumnak a könyvek a fő profiljuk, viszont néhány esetben ettől eltérően más régi értékesnek vélt gyűjteményeket, tárgyakat is árusíthatnak. [2]

1.3. Összehasonlítás

1.3.1. Assister

Az Assister egy online elérhető készletezési és értékesítési funkciókkal ellátott logisztikai és motivációs web alapú alkalmazás. Egy modern felülettel és gyors működéssel futó rendszer, amely testre szabható és több funkció kezelésére alkalmas. Ez az alkalmazás nem csak könyvesboltok készletezésére alkalmas, hanem más termékek eladásában is segítséget nyújt. Ilyen például a ruházkodás, a kiegészítők, elektronikai cikkek, játékok és hobbi termékek nyilvántartása, egészségügyi áruk, sport és fitnesz, kozmetikai kiegészítők és még sok más készletezése.

Fő funkciói, az értékesítés és számlázás. Ezen belül a kedvezmények és ügyfelek kezelése. A rendszer képes egyszerre több pénztárat kezelni, ezekről riportot készíteni. A pénztárakban minden pénznem kezelhető, alkalmas hivatalos számlák készítésére, sztornózásra és visszáru folyamatok elvégzésére. Vonalkód olvasó rendszerrel ellátott. Az alkalmazás e részében lehetőség van kedvezmények, promóciók és kuponok alkalmazására. Az ügyfelek szintén nyilvántarthatóak egy ügyféllistában, ahol tárolva vannak az ügyfelek előző vásárlásaik.

A másik fő funkciója ennek az alkalmazásnak a készletkezelése. Ide sorolhatjuk a terméklistázást, bevételezést, készletátadást, kivezetést, leltározást és a raktározást. A termékek listázása értelmezhető, szükség esetén szűrők be állításával szűkíthető akár üzletre, gyártóra, márkára és termékkategóriára. A rendszer használata segíti a leltározás folyamatát, készletkiegyenlítéssel, illetve nyomtatható leltárdokumentációval.

Ezen funkciók mellett a program lehetőséget nyújt szervezetkezelésre és teljesítménymenedzsmentre. Szervezetkezelés a dolgozók adatainak rendszerezésére

szolgál, illetve, ha több üzletről beszélünk abban az esetben az üzletek közötti hálózat kialakítására is alkalmas. Az üzletek közötti dolgozók mozgását is megoldhatjuk vele. A teljesítménymenedzsment lehetőséget nyújt kvóták, jutalékok kezelésére, bónuszrendszerekre. Lehetőség van riportok készítésére, amelyhez számos szűrőfunkció tartozik. Ezek a riportok könnyen átláthatóak. Az alkalmazásban jelentést készíthetünk akár mindenről, például értékesítésről, készletekről, munkaidőről stb. A rendszerben rugalmas Dashboard található. Néhány kattintással azonnali és valós idejű adatokért saját dashboardot hozhatunk létre.

Az alkalmazás tulajdonosinak fő referencia partnere a Magyar Telekom Nyrt., amellyel 2015 óta dolgoznak együtt. Emellett több mint 180 üzlettel állnak kapcsolatban, amelyben 1300-nál több felhasználót elégítenek ki. [3]

1.3.2. További alkalmazások

Az Assister rendszere volt az egyetlen olyan rendszer, amit kutatásaim alatt találva úgy éreztem, hogy legmegfelelőbbképpen össze tudok hasonlítani saját rendszeremmel, mivel ez a rendszer az, ami árucikkek nyilvántartását és értékesítését ellátja. Kereséseim alatt ezen felül személyes értékesítésre alkalmas rendszereket találtam, valamint a külföldről történő rendelő oldalakat. Ezek mellett olyan nyilvántartó rendszerekkel találkoztam, amelyek például jelenlét nyilvántartásra, szabadságoltatásokra, illetve beosztástervezésre alkalmasak.

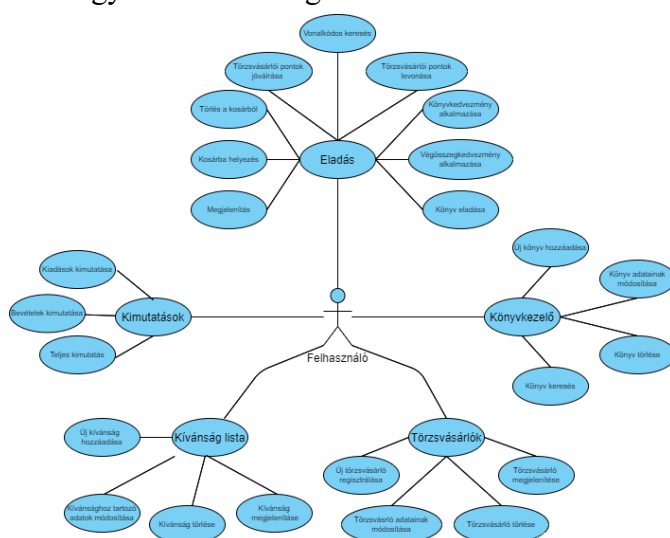
Az egyik ilyen nyilvántartó rendszer a Beosztásom nevezetű alkalmazás, amely egy online munkaidő-beosztáskészítő rendszer. A rendszeralkalmazás jelenléti nyilvántartásra, amely lehetővé teszi a tervezett beosztások és a tényleges jelenléti adatok kezelésére. Szabadságok felvitele, távollétek, illetve túlórák megtekintése is lehetséges benne. Beosztások készítését gyorsan percek alatt lehetővé teszi, mind munkaidőkeretben dolgozó, mind az állandó dolgozók számára is. Mindezen fő funkciók mellett lehetőség van automatikus beosztáskészítésre, munkaerőigény, több munkahely vagy munkakör kezelésére is. [4]

Az interneten rengeteg online antikvárium található, ezek közül van, amely csak online árusítást végez, van, amely pedig személyes árusítást is csinál. Ezekbe a rendszerekbe sajnos túl nagy beelátása egy külső személynek nincsen. Maximum maga a vásárlói felület megismerését tudjuk megtenni.

Az előzőekben áttekintett rendszerek sajnos egyike se egyezik meg teljes mértékben az általam fejlesztett alkalmazással. Ahogy már az előzőekben is említettem leginkább az Assister volt, ami hasonló jellegű szoftver. A fő különbség viszont talán az, hogy míg a saját rendszerem egy kis üzlet nyilvántartására alkalmas és főként ezt is célozza meg, addig az Assisert egy jóval tágabb több üzlet összefogására is alkalmas rendszer, amelyben nem csak az értékesített áruk nyilvántartása történik, hanem a különböző üzletek, valamint azok dolgozóinak a nyomon követése is lehetővé tehető.

2. Felhasználói dokumentáció

A következőkben az alkalmazásban végrehajtható tevékenységek leírás található, a felhasználók részére. Emellett még táblázatokba szedve láthatjuk az egyes funkciók Use Case tábláját, valamint egy Use Case Diagramm.



1. ábra Use Case Diagramm

2.1. Főablak

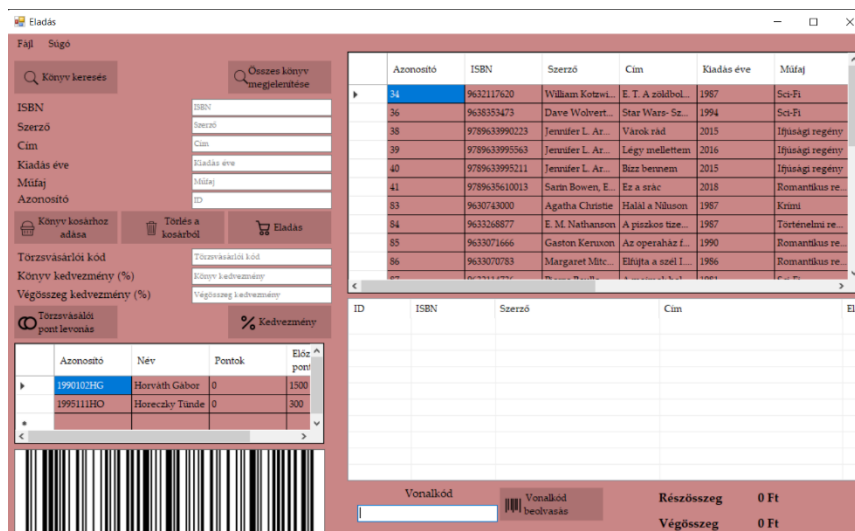
A főablak az első, ami megjelenik az alkalmazás elindításakor. A főablakból főként a többi ablakba tudunk tovább lépni, illetve itt az indításkor megjelenik a könyvek tábla teljes tartalma minden adattal.

ISBN	Szerző	Cím	Kiadás	Műfaj	Kiadó	Oldalak
9632117820	William Kotzwinkle	E. T. A zoldbolygó könyve	1987	Sci-Fi	Kozmosz Könyvek	252
9638353473	Dave Wolverton	Star Wars- Szökevények	1994	Sci-Fi	Valhalla Páholy Kft.	380
9789633990223	Jennifer L. Armentrout	Várok rád	2015	Ifjúsági regény	Könyvmolyképző Kiadó Kft.	491
9789633995563	Jennifer L. Armentrout	Légy mellettem	2016	Ifjúsági regény	Könyvmolyképző Kiadó Kft.	528
9789633995211	Jennifer L. Armentrout	Bizb bennem	2015	Ifjúsági regény	Könyvmolyképző Kiadó Kft.	391
9789635610013	Sarah Bowen, Elle Kennedy	Ez a srác	2018	Romantikus regény	Könyvmolyképző Kiadó Kft.	348
9630743000	Agatha Christie	Halál a Nílison	1987	Krimi	Európa Könyvkiadó	320
9633268877	E. M. Nathanson	A piszokos tizenkettő	1987	Történelmi regény	Zrínyi Katonai Kiadó	595
9633071666	Gaston Kervason	Az operaház fantomja	1990	Romantikus regény	Árkádia	319
9633070783	Margaret Mitchell	Elhúta a szél I.-II.	1986	Romantikus regény	Árkádia	1178
9632114736	Pierre Boulle	A majmok bolygója	1981	Sci-Fi	Kozmosz Könyvek	150
9789632024	Danielle Steel	A sötét oldal	2021	Romantikus regény	Maecenas Kft	320
9789638924438	Ris Attias	Szép bor, jó bor	2020	Gasztronómia	Geclart Bt	182
9789630988902	Carla Bardi	Levesek	2017	Gasztronómia	Kosnuth Kiadó Zrt	130
9789634456230	Mari Salinas	30 edesrecept	2016	Gasztronómia	Nagrasforgo 2005 Kft	64
9789639726437	Krisz Eszter	Paleolit ételek	2012	Gasztronómia	Fu-Therm Kft	175
97896340059168	Stephen King	A ragyogás	2021	Krimi	Európa Könyvkiadó Kft	430
9789634198826	Laurell K Hamilton	Örvén áhíd	2021	Fantasy	Agave Könyvkiadó Kft	608
9789635662746	Anna Todd	Mután	2015	Lektur	Gabo Könyvkiadó és Keresk. Kft	490
9789634061168	Anna Todd	Mután összezsugorult	2015	Lektur	Gabo Könyvkiadó és Keresk. Kft	547
9789634061717	Anna Todd	Mután elbukunk	2015	Lektur	Gabo Könyvkiadó és Keresk. Kft	674
9789634061908	Anna Todd	Mután boldogok leszünk	2016	Lektur	Gabo Könyvkiadó és Keresk. Kft	399

2. ábra Főablak

2.2. Eladás

A rendszer legfontosabb tevékenységei az eladáshoz kapcsolódnak. Ilyen a könyvek kosárba helyezése, onnan való törlésük, a vonalkódos keresés lehetősége, pontjövírás, pont levonás, illetve kedvezmények alkalmazása, és maga az eladás.



3. ábra Eladás ablak

2.2.1. Könyv megjelenítések

Az eladás kezelőben lehetőség van könyv megjelenítésére, akár az azonosító, ISBN szám, szerző, cím, kiadási év vagy pedig műfaj szerint. Emellett az összes könyv megjelenítésére is szolgál egy külön gomb megnyomása.

Könyv keresése esetén a fent felsorolt adatok közül egyet megadva a rendszer megjeleníti azon könyvet, illetve könyveket, amelyek szerepelnek az adatbázisban és egyezést talált a lekérdezés a megadott adattal.

Az összes könyv kimutatásánál nincs szükség semmilyen adat meghatározására. Ez a funkció arra szolgál, hogy egy keresést követően visszaállítsuk könyveket megjelenítő táblát.

1. táblázat Könyv keresés

Név	Könyv keresés
Cél	Keresett könyv megtalálása.
Előfeltétel	Eladás kezelő megnyitása
Sikeres lefutás	Könyv megtalálva
Sikertelen lefutás	Könyv nem található.
Elsődleges aktor	Felhasználó

Kiváltó esemény	A könyv valamely adatának megadása.	
Fő lépések	Lépés	Tevékenység
	1	A könyv valamely adatának megadása. (ISBN, Szerző, Cím, Kiadási év, Műfaj)
	2	Könyv keresés gomb megnyomása
	3	Könyv megjelenik a táblázatban.
Kiegészítések	Lépés	Elágazó tevékenység
	2.1	Összeskönyv megjelenítése gomb megnyomása.
	2.2	Összes könyv megjelenik
	2.3	Funkció futása leáll.
	3.1	Könyv keresés sikertelen.
	3.2	Megadott adat ellenőrzése, javítása.
	3.3.1	Könyv nem szerepel az adatbázisban

2.2.2. Könyv kosárba helyezése

Könyv vásárlás esetén az első lépés a könyv kosárba helyezése.

A könyvet az ISBN száma alapján megkeressük a rendszerbe és kosárba helyezzük. Az első könyv kosárba helyezésekor létre jön egy lista, amely a kosárba helyezett könyvek azonosítóját tartalmazza. Ezen felül egy táblázat segítségével megjelenik a kosártartalma is. Ebben a táblázatban a könyv ISBN száma, szerzője, címe és eladási ár jelenik meg.

Minden könyv kosárba helyezésekor a kosár alatt megjelenik az aktuális könyv ára mint részösszeg, valamint minden alkalommal frissül a teljes fizetendő összeg is a hozzá adott könyv árával növelve. Ha minden könyv bekerült a kosárba, akkor tudunk kedvezményeket adni, pontot levonni, illetve a könyveket eladni.

2. táblázat Könyv kosárba helyezése

Név	Könyv kosárba helyezése	
Cél	Könyvek kosárba kerülése.	
Előfeltétel	Könyv megkeresése az adatbázisban.	
Sikeres lefutás	Könyv bekerült a kosárba.	
Sikertelen lefutás	Könyv nem került be a kosárba.	
Elsődleges aktor	Felhasználó	
Kiváltó esemény	Vásárló kiválasztotta a keresett könyvet/könyveket.	
Fő lépések	Lépés	Tevékenység
	1	Könyv ISBN számának megadása.
	2	Könyv kiválasztása az adatbázisból.
	3	Könyv kosárba helyezése.
	4	Ellenőrzés, hogy a könyv bekerült a kosárba.

Kiegészítések	Lépés	Elágazó tevékenység
	2.1	Könyv nem található az adatbázisban.
	2.2	ISBN szám ellenőrzése/újbolli megadása.
	4.1	Könyv nem került be a kosárba.
	4.2	Folyamat ismételt végre hajtása.

2.2.3. Könyv törlése a kosárból

Ha egy könyv hibásan kerül be a kosárba lehetőségünk van azt törölni onnan, olyan módon, hogy megadjuk a kosár táblában található könyv azonosítót és a törlés gombra kattintva a könyv kikerül a kosárból, illetve a hozzá tartozó azonosító törlődik a kosár azonosítóit tartalmazó listából is.

3. táblázat Könyv törlése a kosárból

Név	Könyv törlése a kosárból	
Cél	Könyv kitörlése a kosárból.	
Előfeltétel	Könyv kosárba helyezése.	
Sikeres lefutás	Könyv törlődik a kosárból.	
Sikertelen lefutás	Könyv bent marad a kosárba.	
Elsődleges aktor	Felhasználó	
Kiváltó esemény	Hibás könyv felvétel a kosárba.	
Fő lépések	Lépés	Tevékenység
	1	Hibásan felvett könyv megkeresése a kosárban.
	2	Könyv azonosítójának megadása.
	3	Könyv törlése a kosárból.
	4	Kosár ellenőrzése.
Kiegészítések	Lépés	Elágazó tevékenység
	2.1	Rossz azonosító megadás.
	2.2	A hibásan törlött könyv újra felvétele a kosárba.
	4.1	Törlés nem történt meg.
	4.2	Törlés folyamatának újra kezdése.

2.2.4. Törzsvásárlói pontok jóváírása

A rendszer lehetőséget nyújt arra, hogy regisztrált törzsvásárlóknak a vásárlásuk után pontokat írjunk jóvá a következők alapján.

A vásárlás végén a törzsvásárlói kód megadása után a végösszegeből a rendszer kiszámolja a pontokat. Minden 100 forint után jár 1 törzsvásárlói pont. Utólagosan nincs lehetőség e pontok jóváírására.

4. táblázat Törzsvásárlói pontok jóváírása

Név	Törzsvásárlói pontok jóváírása	
Cél	Pontok jóváírása.	
Előfeltétel	Regisztrált törzsvásárló.	
Sikeres lefutás	Pontjóváírás.	
Sikertelen lefutás	Pontok nem íródnak jóvá.	
Elsődleges aktor	Felhasználó	
Kiváltó esemény	Könyvvásárlás.	
Fő lépések	Lépés	Tevékenység
	1	Könyvek kosárba helyezése.
	2	Törzsvásárlói kód megadása.
	3	Szükség esetén akciók alkalmazása pontok levonása.
	4	Könyv eladás
	5	Pontjóváírás.
Kiegészítések	Lépés	Elágazó tevékenység
	3.1.1	Könyv kedvezmény mértékének megadása.
	3.1.2	Könyv kedvezmény alkalmazása.
	3.2.1	Végösszeg kedvezmény mértékének megadása.
	3.2.2	Végösszeg kedvezmény alkalmazása.
	3.3.1	Törzsvásárlói pont levonás.

2.2.5. Törzsvásárlói pontok levonása

A törzsvásárlók számára lehetőség van a megszerzett pontokat a vásárlás végén levonattatni.

1 pont 1 forintnak felel meg a levonás esetén. Ebben az esetben is először meg kell adnunk a törzsvásárlói kódot. Ez után a levonás gombra kattintva a rendszer automatikusan megvizsgálja, hogy a vásárlónak milyen és mennyi pontja van. Megvizsgálja első sorban, hogy a pontok értéke nem haladja-e meg a vásárlás összegét. Meghaladás esetén a pontokból csak a megadott összeget vonja le. Emellett azt is vizsgálja, hogy van-e megmaradt előző éves pont, ha igen akkor először ebből vonja le a szükséges mennyiséget, ezt követően pedig az aktuális pontokból von le, ha még szükséges.

Ha pont levonás történik is a vásárlás végén a megmaradt végösszezből a pontjóváírás ugyanúgy megtörténik.

5. táblázat Törzsvásárlói pontok levonása

Név	Törzsvásárlói pontok levonása	
Cél	Törzsvásárlói pontok levonása a végösszezből.	
Előfeltétel	Regisztrált törzsvásárló.	
Sikeres lefutás	Pontlevonás.	
Sikertelen lefutás	Sikertelen pont levonás.	
Elsődleges aktor	Felhasználó	
Kiváltó esemény	Könyvek kosárba helyezése.	
Fő lépések	Lépés	Tevékenység
	1	Könyvek kosárba helyezése.
	2	Törzsvásárlói kód megadása.
	3	Törzsvásárlói pontok levonása.
	4	Levonás sikerességének ellenőrzése.
Kiegészítések	Lépés	Elágazó tevékenység
	4.1	Nem sikerült levonni a pontokat.
	4.2	Tevékenységek a 2-es ponttól való újboni végre hajtása.

2.2.6. Könyvkedvezmény alkalmazása

Az eladás előtt lehetőségünk van adott könyvek összegéből történő kedvezmény levonására.

Ezt a vásárlás folyamata alatt bármikor megtehetjük, olyan módon, hogy megadjuk a kosárban már szereplő könyv azonosítóját, amelyre a kedvezményt alkalmazni szeretnénk. Ezután százalékos formában meg kell adnunk a kedvezmény mértékét. Ezt követően a rendszer kiszámítja a kedvezmény összegét és frissítve kiírja az új végösszeget. A kosárban a kedvezmény utáni ár nem frissül le.

6. táblázat Könyv kedvezmény alkalmazása

Név	Könyv kedvezmény alkalmazása	
Cél	Kedvezmény alkalmazása meghatározott könyvre.	
Előfeltétel	Könyv kosárba helyezése.	
Sikeres lefutás	Könyv kedvezmény levonódott.	
Sikertelen lefutás	Nem történik meg a levonás.	
Elsődleges aktor	Felhasználó	
Kiváltó esemény	Kedvezményezett könyv azonosítójának megadása.	
Fő lépések	Lépés	Tevékenység
	1	Kedvezményezett könyv azonosítójának megadása.
	2	Kedvezmény mértékének meghatározása.
	3	Kedvezmény alkalmazása.

	4	Végösszeg ellenőrzése.
Kiegészítések	Lépés	Elágazó tevékenység
	4.1	A végösszegeből nem vonódott le a kedvezmény.
	4.2	Kedvezmény levonás folyamatának ismétlése.

2.2.7. Végösszegkedvezmény alkalmazása

A vásárlás végén az eladást megelőzően végösszeg kedvezményt is adhatunk. Ezt érdemes minden esetben az eladást követően elvégezni, mikor már minden könyv a kosárban szerepel, mivel a kedvezmény alkalmazása után kosárba kerülő könyvekre a kedvezmény már nem érvényesül.

Ebben az esetben is a könyvkedvezményhez hasonlóan meg kell adnunk a kedvezmény mértékét százalékban, majd gombnyomást követően a rendszer kiszámítja a kedvezményt és azt a végösszegeből levonja. Ebben az esetben is a képernyőn a végösszeg levonást követő értéke jelenik meg.

7. táblázat Végösszeg kedvezmény alkalmazása

Név	Végösszeg kedvezmény alkalmazása	
Cél	Kedvezmény alkalmazása végösszegre.	
Előfeltétel	Minden könyv kosárba helyezése.	
Sikeres lefutás	Végösszeg kedvezmény levonódott.	
Sikertelen lefutás	Nem történik meg a levonás.	
Elsődleges aktor	Felhasználó	
Kiváltó esemény	Kedvezmény mértékének meghatározása.	
Fő lépések	Lépés	Tevékenység
	1	Kedvezmény mértékének meghatározása.
	2	Kedvezmény alkalmazása.
	3	Végösszeg ellenőrzése.
Kiegészítések	Lépés	Elágazó tevékenység
	3.1	A végösszegeből nem vonódott le a kedvezmény.
	3.2	Kedvezmény levonás folyamatának ismétlése.

2.2.8. Könyvek eladása

Az eladási folyamat funkciói közül a leglényegesebb talán maga a végső eladás. Melynek lényege, hogy az adatbázisból kikeresve az adott könyvet, azt onnan kitörölve a könyvet eladjuk, valamint az eladás összeg értéke egy másik táblába kerül az aktuális dátummal együtt, melynek segítségével a későbbiekben kimutatásokat tudunk készíteni.

Ha minden könyv bekerült az kosárba, valamint megtörténtek szükség esetén a levonások és kedvezmények alkalmazásai is, az eladás gomb megnyomásával a rendszer az adott könyveket törli az adatbázisból a könyvek kosárba helyezése során használt lista segítségével, emellett az összeg, amelyet a vásárló fizet, bekerül az eladás táblába. Ezt követően az azonosítókat tartalmazó lista törlődik. Így a következő vásárló esetén egy teljesen üres listával indul a rendszer.

8. táblázat Könyv eladása

Név	Könyvek eladása		
Cél	Kiválasztott könyvek eladása.		
Előfeltétel	Könyvek kosárba helyezése, esetleges akciók alkalmazása, pontok levonása.		
Sikeres lefutás	Könyvek eladása, törlése az adatbázisból, bevételek táblába a végösszeg bejegyzése.		
Sikertelen lefutás	Könyv eladás sikertelen.		
Elsődleges aktor	Felhasználó		
Kiváltó esemény	Minden megvásárolni kívánt könyv a kosárba van.		
Fő lépések	Lépés	Tevékenység	
	1	Minden megvásárolni kívánt könyv a kosárba van. Akciók és pontok levonásra kerültek.	
	2	Könyv eladás.	
	3	Eladás ellenőrzése.	
Kiegészítések	Lépés	Elágazó tevékenység	
	3.1	Sikertelen eladás.	
	3.2	Eladás folyamatának megismétlése.	

2.2.9. Vonalkódos keresés

A rendszer lehetővé teszi, hogy vonalkód segítségével is be tudjuk olvasni az ISBN számot, ezzel felgyorsítva az adott könyv kiválasztását az adatbázisból, így nem kell kézíleg kikeresni a könyvek közül a keresetett, illetve nem kell felesleges időt gépeléssel töltenie a felhasználónak.

Sajnos nem volt lehetőségem vonalkód olvasó beszerzésére, ezért ezt a problémát ideiglenesen olyan módon oldottam meg, hogy egy vonalkódot megjelenítő képet tölthet fel a rendszerbe a felhasználó és az alapján ismeri fel a rendszer ezt a kódot.

Tapasztalatom alapján a mai könyveknél a vonalkód és az ISBN szám megegyezik teljes mértékben, míg a régebbi könyveknél ez sajnos nem teljesen így van. A régi könyveknél a vonalkód részben tartalmazza az ISBN számot, még hozzá úgy, hogy a vonalkód első három karakterét eltávolítva, illetve az utolsó karaktert levágva a kódból

megkapjuk az ISBN szám első 9 karakterét. Így az adatbázisban való keresés részlet kereséssel megy végbe mivel a vonalkódból elő állított új kód csak egy részét adja meg az ISBN számnak.

Ezek beolvasásának segítségével tudunk az adatbázisban egyszerűbben rá keresni egy adott könyvre.

9. táblázat Vonalkódos keresés

Név	Vonalkódos keresés	
Cél	Könyv megtalálása.	
Előfeltétel	Könyvek iránti vásárlási szándék.	
Sikeres lefutás	Könyvet meg lett találva.	
Sikertelen lefutás	Könyv megtalálása sikertelen.	
Elsődleges aktor	Felhasználó	
Kiváltó esemény	Eladás ablak megnyitása.	
Fő lépések	Lépés	Tevékenység
	1	Eladás ablak megnyitása.
	2	Vonalkód olvasás gomb megnyomása.
	3	Kép kiválasztása a felugró ablakban.
	4	Kép betöltése.
	5	Könyv ISBN számának megadása a vonalkód alapján.
Kiegészítések	Lépés	Elágazó tevékenység
	5.1	A beolvasott vonalkódhoz nem tartozik könyv.

2.3. Törzsvásárlók

Az üzletbe betérő vásárlóknak lehetőségük van törzsvásárlói kedvezményekben részesülni, ha a felhasználó beregisztrálja őket a rendszerbe.

A regisztrálást követően pontgyűjtésre van lehetősége a vásárlónak, amelyet később végösszeg belőli levonásra tud felhasználni. A regisztrálás mellett az adatok módosítására, valamint a törzsvásárló törlésére is van lehetőség.

Törzsv kód	Név	Cím	Születési dátum	Nem	Tel. szám	Email cím
199010HG	Horvath Gabor	8500, Pápa	1990. 10. 04.	Férfi	+36-70-346-0031	gary158vip
199511HO	Horeczky Tünde	8415, Nagyesztergár, Radnóti utca 125	1995. 11. 17.	Nő	+36-70-619-3520	horeczky95
*						

4. ábra Törzsvásárlók ablak

2.3.1. Törzsvásárlói regisztrálás

Akár az első vásárlás alkalmával is dönthet úgy a vevő, hogy szeretne részt venni a törzsvásárlói pontgyűjtésben. Ezt jelezve az eladó az erre a célra elkészített felületen a vásárló adatait megadva beregisztrálja törzsvásárlónak.

A regisztráláshoz szükséges adatok a teljes név, a születési dátum, a nem, a lakcím, telefonszám és az e-mail cím. Ezekből az adatokból, 3 adatot kombinálva, egy egyedi törzsvásárlói kódot hoz létre a rendszer, a név, születési dátum és nem kombinációjával adja meg a következő eljárással. Az első 6 karakter a vásárló születési dátumának éve és hónapja. Ezt követi a nem meghatározása olyan módon, hogy ha a személy hölgy/nő akkor a hetedik karaktere a kódznak 1, ha úr/férfi akkor ez a karakter 2-es értéket kapja. Az utolsó két karakter a vásárló vezetéknévének első két betűje nagybetűvé átalakítva.

A regisztráláshoz egy automatikus e-mail küldés is társul, amit a későbbiekben ismertetek.

10. táblázat Törzsvásárlói regisztrálás

Név	Törzsvásárlói regisztrálás	
Cél	Új törzsvásárló regisztrálása.	
Előfeltétel	Igény jelzése.	
Sikeres lefutás	Sikeres regisztrálás.	
Sikertelen lefutás	Sikertelen regisztráció.	
Elsődleges aktor	Felhasználó	
Kiváltó esemény	Törzsvásárlói kezelő felületbe való belépés.	
Fő lépések	Lépés	Tevékenység

	1	Törzsvásárlói kezelő felületbe való belépés.
	2	A vásárló adatainak megadása.
	3	Adatok ellenőrzése.
	4	Regisztrálás.
	5	Regisztráció sikerességének ellenőrzése.
Kiegészítések	Lépés	Elágazó tevékenység
	3.1	Valamely adat hibásan lett megadva.
	3.2	Adat átírása/módosítása.
	5.1	Sikertelen regisztráció.
	5.2	Regisztrációs folyamat ismételt végre hajtása.

2.3.2. Törzsvásárló megjelenítése

Lehetőségünk van arra, hogy a törzsvásárlót az azonosító megadásával megjelenítsük, így könnyedén tudunk adni információt arról, hogy pontokban, hogy is áll, továbbá módosítás esetén gyorsabban tudjuk ellenőrizni az adatokat.

11. táblázat Törzsvásárló megjelenítése

Név	Törzsvásárló megjelenítése	
Cél	Törzsvásárló megjelenítése.	
Előfeltétel	Regisztrált törzsvásárló.	
Sikeres lefutás	Sikeres megjelenítés.	
Sikertelen lefutás	Sikertelen megjelenítése.	
Elsődleges aktor	Felhasználó	
Kiváltó esemény	Törzsvásárlói kezelő felületbe való belépés.	
Fő lépések	Lépés	Tevékenység
	1	Törzsvásárlói kezelő felületbe való belépés.
	2	Törzsvásárlói kód megadása.
	3	Megjelenítés gomb megnyomása.
Kiegészítések	Lépés	Elágazó tevékenység
	3.1	Nem jelenik meg a törzsvásárló.
	3.2	Törzsvásárlói kód ellenőrzése, javítása.
	3.3	Ismételt gomb nyomás.

2.3.3. Törzsvásárló adatainak módosítása

Hibás felvitel vagy adat változás esetén, amelyet a vásárló tud jelezni a felhasználó felé, a felhasználónak lehetősége van a vásárló adatait módosítani.

Ehhez meg kell adni az egyedi azonosító kódot és a megváltoztatni kívánt adatot. Olyan esetben, ha a név változik a rendszerben, az előzetesen létre jött azonosító nem fog megváltozni. Erre a törzsvásárlónak bármikor lehetősége van, nem csak vásárlás esetén.

12. táblázat Törzsvásárló adatainak módosítása

Név	Törzsvásárló adatainak módosítása	
Cél	Hibásan felvitt adatok módosítása.	
Előfeltétel	Vásárló regisztrációja.	
Sikeres lefutás	Adat módosítás végbe megy.	
Sikertelen lefutás	Adat módosítás sikertelen.	
Elsődleges aktor	Felhasználó.	
Kiváltó esemény	Hiba észlelése.	
Fő lépések	Lépés	Tevékenység
	1	Hiba észlelése.
	2	Törzsvásárlói kód megadása.
	3	Hibás adat megadása megfelelően.
	4	Módosítás gomb megnyomása.
	5	Módosítás sikerességének ellenőrzése.
Kiegészítések	Lépés	Elágazó tevékenység
	4.1	Hibás törzsvásárlói kód miatt a rendszer hibát jelez.
	4.2	Törzsvásárlói kód ellenőrzése.
	4.3	Tevékenység ismételt végre hajtása.
	5.1	Módosítás sikertelen.
	5.2	Adatmódosítási folyamat újboni végre hajtása.

2.3.4. Törzsvásárló törlése

A törzsvásárló törlése egy gyorsan és egyszerűen végre hajtható folyamat. Olyan esetekben használatos ez a funkció, ha valaki szeretne lemondani törzsvásárlói jogáról, vagy egy meghatározott időn belül nem használta törzsvásárlási lehetőségét. Ebben az esetben csak meg kell adnunk az egyedi törzsvásárlói azonosítót és a törzsvásárló törlésére kattintva az törlődik az adatbázisból. Ezt követően az felhasználónak ellenőriznie kell, hogy a törlés végbe ment-e, ha nem akkor újra meg kell ismételnie a szükséges lépéseket.

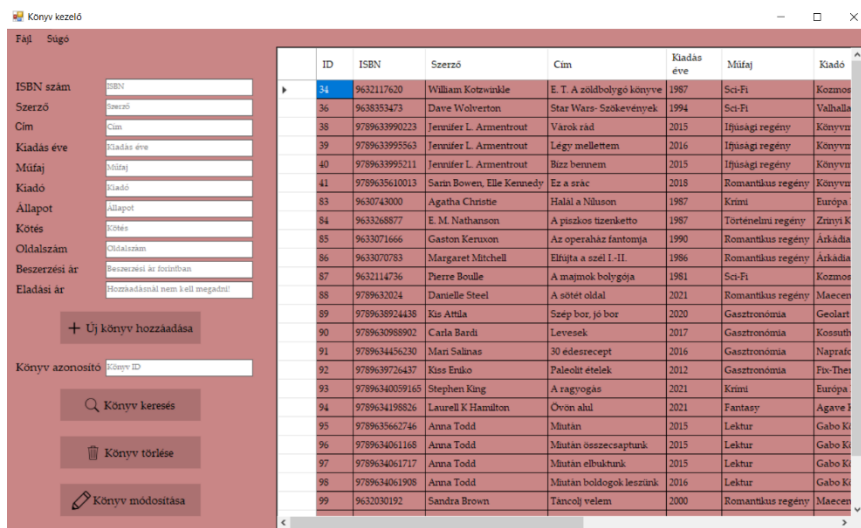
13. táblázat Törzsvásárló törlése

Név	Törzsvásárló törlése	
Cél	Törzsvásárló eltávolítása az adatbázisból.	
Előfeltétel	Vásárló regisztrálása.	
Sikeres lefutás	Törzsvásárló törlése megtörtént.	
Sikertelen lefutás	Törlés folyamata sikertelen.	
Elsődleges aktor	Felhasználó	
Kiváltó esemény	Törzsvásárlói kezelő felület megnyitása.	
Fő lépések	Lépés	Tevékenység

	1	Törzsvásárlói kezelő felület megnyitása.
	2	Törzsvásárló azonosítójának meg keresése.
	3	Törzsvásárlói azonosító megadása.
	4	Törlés.
	5	Törlés sikerességének ellenőrzése.
Kiegészítések	Lépés	Elágazó tevékenység
	5.1	Sikertelen törlés.
	5.2	Törlés ismételt végre hajtása.

2.4. Könyvkezelő

A könyvkezelő felület nevéből is adódóan a könyvek kezelésére szolgál. Itt tudjuk a megvásárolt könyveket felvinni az adatbázisba. A fent lévő könyveket módosítani, illetve törölni szükség esetén.



5. ábra Könyvkezelő ablak

2.4.1. Új könyv hozzáadása

Mikor új könyveket vásárol fel az üzlet eladásra, akkor szükségeltetik a könyveket felvinni az adatbázisba.

Ekkor a felhasználó megadja a könyvhöz tartozó adatokat. Ezek az ISBN szám, a szerző, a cím, a kiadás éve, a műfaj, kiadó, állapot, kötés, oldalszám és a beszerzési ár. Ezek megadása kötelező.

Az alkalmazás automatikusan kiszámítja az eladási árat a rendszerbe meghatározott árrés segítségével. Ez pillanatnyilag 25% növelést jelent. A felvitelt követően az ablakban

található datagridview lefrissül és az újonnan felvett könyvet is megjeleníti, így a felhasználó könnyedén tudja ellenőrizni a felvitel sikerességét.

Ehhez a funkcióhoz tartozik egy automatikus eljárás, amit a lentebb más pontban fejtek ki.

14. táblázat Új könyv hozzáadása

Név	Új könyv hozzáadása	
Cél	Új könyv bekerülése az adatbázisba.	
Előfeltétel	Új könyv felvásárlása.	
Sikeres lefutás	A könyv megjelenik az adatbázisban.	
Sikertelen lefutás	A könyv nem jelenik meg az adatbázisban.	
Elsődleges aktor	Felhasználó	
Kiváltó esemény	Új könyv beérkezése az üzletbe.	
Fő lépések	Lépés	Tevékenység
	1	Könyvek kilistázása.
	2	Az új könyv szükséges adatainak megadása.
	3	Az adatok ellenőrzése.
	4	A könyv hozzáadása az adatbázishoz.
	5	Az adatbázis ellenőrzése, hogy a könyv bekerült-e.
	6	A kívánság lista alapján üzenetküldés, amit a rendszer automatikusan tesz meg.
Kiegészítések	Lépés	Elágazó tevékenység
	3.1	Hibás adatbevitel esetén korrekció.
	5.1	A könyv nem került be az adatbázisba.
	5.2	A könyvet újra felvisszük az adatbázisba.
	6.1	Nincs a kívánság listában a könyv, a rendszer nem küld üzenetet.

2.4.2. Könyv keresés

A felhasználó rá tud keresni egy adott könyvre annak azonosítóját, ISBN számát, szerzőjét, címét, kiadásának évét vagy műfaját megadva.

15. táblázat Könyv keresés

Név	Könyv keresés
Cél	Könyv megtalálása.
Előfeltétel	Könyv hozzá lett adva az adatbázishoz.
Sikeres lefutás	Könyv keresés sikeres.
Sikertelen lefutás	Könyv keresés sikertelen.
Elsődleges aktor	Felhasználó
Kiváltó esemény	Könyv kezelő megnyitása.

Fő lépések	Lépés	Tevékenység
	1	Könyv kezelő megnyitása.
	2	Kereset könyv adatának megadás. (ID, ISBN, Szerző, Cím, Kiadás éve, Műfaj)
	3	Könyv keresés gomb megnyomása.
Kiegészítések	Lépés	Elágazó tevékenység
	3.1	Könyv nem jelenik meg.
	3.2	Megadott adat ellenőrzése.
	3.4	Ismételt keresés.

2.4.3. Könyv adatainak módosítása

Lehetőségünk van arra, hogy az adatbázisba felvitt könyveket módosítsuk szükség esetén.

Hibás felvitel vagy esetleges árváltozás esetén tudjuk ezt a funkciót végbe vinni. A hiba észlelése vagy árváltozás esetén az adatbázisból a datagridviewba megkeresve duplakattintással vagy saját kezűleg beírva a felhasználó megadja a könyv azonosítóját valamint a módosítani kívánt adatot. Ezt követően a módosítás gombra kattintva a módosítás végbe megy.

Ez után a felhasználó ellenőrizheti a lefrissült datagridviewban, hogy a módosítás sikeres volt-e. Ezt megteheti úgy, hogy megadja újfent a könyv azonosítóját, illetve kézzel is kikereshető az adott könyv.

16. táblázat Könyv adatainak módosítása

Név	Könyv adatainak módosítása	
Cél	A könyv hibás adatainak kijavítása.	
Előfeltétel	Könyv bekerülése az adatbázisba.	
Sikeres lefutás	Az adatok módosulnak.	
Sikertelen lefutás	Az adatok nem módosulnak.	
Elsődleges aktor	Felhasználó	
Kiváltó esemény	Az adatbázis ellenőrzésekor észlelt hiba.	
Fő lépések	Lépés	Tevékenység
	1	Hiba észlelése.
	2	Könyv kiválasztása.
	3	Hibás adatok javítása.
	4	Javítások mentése.
	5	Ellenőrzés.
Kiegészítések	Lépés	Elágazó tevékenység
	5.1	Sikertelen volt az adatmódosítás.
	5.2	Adatmódosítási folyamat ismételt végre hajtása.

2.4.4. Könyv törlése

Több esetben is szükség lehet e funkció használatára. Vásárlás, hibás felvitel, illetve leltár esetén, ha olyan könyv szerepel az adatbázisban, ami az üzletben nem felelhető.

Egyik esete mikor vásárlás alkalmával töröljük a könyvet az adatbázisból. Ez az eladás funkcióval van egybekötve és egy egyszerű törlési folyamat megy végbe az eladás gomb megnyomása után annyiszor, ahány könyvet az aktuális értékesítéskor elvisznek. Ez a törlési folyamatot nem közvetlenül a felhasználó végzi el, hanem automatikusan megy végbe az eladás során.

Más esetekben kifejezetten a könyv törlése a cél közvetlenül a felhasználó által. Ilyen eset például, ha hibásan lett hozzáadva az adatbázishoz egy könyv, vagy pedig leltár során, az adatbázisban szereplő könyv nincs készleten az üzletbe. Ebben a helyzetben a felhasználónak meg kell adnia a könyv azonosítóját, ezt követően pedig a törlés gomb megnyomásával a könyv törlődik az adatbázisból. Ha ez megtörtént a felhasználónak ellenőriznie kell, hogy a törlés sikeres volt-e, ezt megteheti úgy, hogy kézzel megadja a törlött könyv azonosítóját. Ha nem történt meg az törlés, akkor újra végre kell hajtania a folyamatot.

17. táblázat Könyv törlése

Név	Könyv törlése	
Cél	Könyv eltávolítása az adatbázisból.	
Előfeltétel	Hibás könyv felvitele.	
Sikeres lefutás	Könyv törlése megtörtént.	
Sikertelen lefutás	Törlés folyamata sikertelen.	
Elsődleges aktor	Felhasználó	
Kiváltó esemény	Könyvkezelő felület megnyitása.	
Fő lépések	Lépés	Tevékenység
	1	Könyvkezelő felület megnyitása.
	2	Könyv meg keresése.
	3	Könyv azonosítójának megadása.
	4	Törlés.
	5	Törlés sikerességének ellenőrzése.
Kiegészítések	Lépés	Elágazó tevékenység
	5.1	Sikertelen törlés.
	5.2	Törlés ismételt végre hajtása.

2.5. Kívánság lista

A kívánság lista egy lehetőség a vásárlók számára, hogy egy általuk keresett, de az üzletben nem fellelhető könyvet „megrendeljenek”. Ebben az esetben a felhasználó

6. ábra Kívánság lista ablak

felviszi a kívánt könyvet az adatbázisba a vásárló adataival együtt. Amikor a könyv megérkezik az üzletbe a megadott adatok alapján az felhasználó valamilyen formában értesíteni tudja a vásárlót.

2.5.1. Új kívánság hozzáadása

Lehetőség van arra, hogy ha egy vásárló egy adott könyvet az üzletben nem talált meg akkor azt a kívánság listához hozzá adva értesítést kérjen arról, ha beérkezik a könyv a boltba.

Ilyenkor a felhasználónak meg kell adni a keresett könyv ISBN számát, szerzőjének nevét és a könyv címét. Ezek mellett a vásárló nevét, címét, e-mail címét, telefonszámát meg kell adni, valamint, ha a vásárló törzsvásárlóként regisztrálva van a rendszerben, akkor a törzsvásárlói kódját is megadhatjuk. Ezt az adatot nem kötelező megadni a többivel szemben. A felhasználó az adatok megadása után az új kívánság hozzá adása gombbal tudja menteni a kívánságot a rendszerbe. Ezt követően ellenőrizni kell, hogy sikeresen végbe ment-e a mentés, ha ez nem történt meg akkor meg kell ismételni a folyamatot.

18. táblázat Új kívánság hozzáadása

Név	Új kívánság hozzáadása	
Cél	Új kívánság regisztrálása.	
Előfeltétel	Igény jelzése.	
Sikeres lefutás	Sikeres regisztrálás.	
Sikertelen lefutás	Sikertelen regisztráció.	
Elsődleges aktor	Felhasználó	
Kiváltó esemény	Kívánság lista kezelő felület megnyitása.	
Fő lépések	Lépés	Tevékenység
	1	Kívánság lista kezelő felület megnyitása.
	2	A kívánt könyv adatainak megadása.
	3	Adatok ellenőrzése.
	4	Felvitel.
	5	Felvétel sikerességének ellenőrzése.
Kiegészítések	Lépés	Elágazó tevékenység
	3.1	Valamely adat hibásan lett megadva.
	3.2	Adat átírása/módosítása.
	5.1	Sikertelen regisztráció.
	5.2	Regisztrációs folyamat ismételt végre hajtása.

2.5.2. Kívánság megjelenítése

Lehetőségünk van arra, hogy egy kívánságot az azonosító megadásával megjelenítsük, így könnyedén tudjuk módosítás esetén ellenőrizni az adatokat.

19. táblázat Kívánság megjelenítése

Név	Kívánság megjelenítése	
Cél	Kívánság megjelenítése.	
Előfeltétel	Regisztrált kívánság.	
Sikeres lefutás	Sikeres megjelenítés.	
Sikertelen lefutás	Sikertelen megjelenítés.	
Elsődleges aktor	Felhasználó	
Kiváltó esemény	Kívánság kezelő felületbe való belépés.	
Fő lépések	Lépés	Tevékenység
	1	Kívánság kezelő felületbe való belépés.
	2	Kívánság azonosító megadása.
	3	Megjelenítés gomb megnyomása.
Kiegészítések	Lépés	Elágazó tevékenység
	3.1	Nem jelenik meg a kívánság.
	3.2	Kívánság azonosító ellenőrzése, javítása.
	3.3	Ismételt gombnyomás.

2.5.3. Kívánsághoz tartozó adatok módosítása

Hibás kívánság felvitel esetén, illetve, ha a vásárlónak változott valamilyen adata akkor azok módosítására is van lehetőség.

Ilyenkor ki kell keresni a kívánság azonosítóját az adatbázisból azt megadva, valamint a módosítani kívánt adatot meghatározva tudja a felhasználó megváltoztatni az adatokat. Ha az adatmódosítás lefolyt, az ablakban található datagridview automatikusan lefrissül, így a felhasználó ellenőrizni tudja, hogy a módosítás sikeres volt-e vagy sem.

20. táblázat Kívánsághoz tartozó adatok módosítása

Név	Kívánsághoz tartozó adatok módosítása	
Cél	Hibásan felvitt adatok módosítása.	
Előfeltétel	Felvitelkor hibás adat megadása.	
Sikeres lefutás	Adat módosítás végbe megy.	
Sikertelen lefutás	Adat módosítás sikertelen.	
Elsődleges aktor	Felhasználó.	
Kiváltó esemény	Hiba észlelése.	
Fő lépések	Lépés	Tevékenység
	1	Hiba észlelése.
	2	Kívánság azonosító megadása.
	3	Hibás adat megadása megfelelően.
	4	Módosítás gomb megnyomása.
	5	Módosítás sikerességének ellenőrzése.
Kiegészítések	Lépés	Elágazó tevékenység
	5.1	Módosítás sikertelen.
	5.2	Adatmódosítási folyamat újboni végre hajtása.

2.5.4. Kívánság törlése

Az alkalmazás lehetővé teszi a felhasználó számára, hogy szükség esetén kívánságot törölni tudjon. Olyan esetekben használható, ha például a vásárló jelzi, hogy már nincsen szüksége a könyvre, mert esetlegesen már máshol megtalálta a keresett könyvet. Vagy olyan esetben törölhetjük a kívánságot, ha feleslegesen lett felvezetve. Valamint akkor törölni kell a kívánságot mikor végbement az adott könyv eladása a kívánságot leadott vásárló által.

Ilyenkor a felhasználónak meg kell keresni-e az adott kívánság azonosítóját és annak megadásával a rendszerből törölni tudja az adott adatokat. Ezt követően ellenőrizni kell, hogy sikeresen végbe ment-e a törlés. Ha nem akkor ismételten végre kell hajtani a folyamatot.

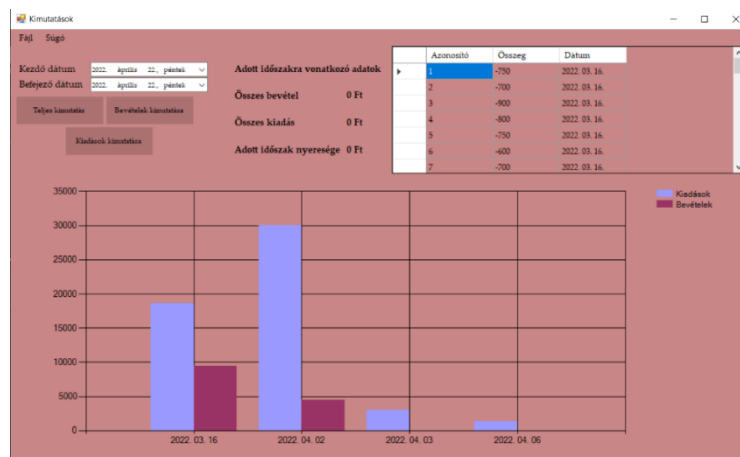
21. táblázat Kívánság törlése

Név	Kívánság törlése	
Cél	Kívánság eltávolítása az adatbázisból.	
Előfeltétel	Felesleges kívánság felvitele.	
Sikeres lefutás	Kívánság törlése megtörtént.	
Sikertelen lefutás	Törlés folyamata sikertelen.	
Elsődleges aktor	Felhasználó	
Kiváltó esemény	Kívánság lista kezelő felület megnyitása.	
Fő lépések	Lépés	Tevékenység
	1	Kívánság lista kezelő felület megnyitása.
	2	Kívánság meg keresése.
	3	Kívánságazonosító megadása.
	4	Törlés.
	5	Törlés sikerességének ellenőrzése.
Kiegészítések	Lépés	Elágazó tevékenység
	5.1	Sikertelen törlés.
	5.2	Törlés ismételt végre hajtása.

2.6. Kimutatások

A kimutatások célja, hogy adott időszakokra a felhasználó vissza tudjon tekinteni, hogy milyen volt a forgalom. Amellett, hogy láthatjuk az egyes kiadásokat és bevételeket, azt is láthatjuk, hogy az adott időszakra a vállalkozás nyereséges vagy veszteséges volt-e.

A kimutatások ablak megnyitásakor az üzlet teljes időszakra vonatkozó teljes kimutatása jelenik meg. Mind a három esetben a felhasználónak meg kell adni egy kezdő és egy befejező dátumot, ami alapján a rendszer kimutatja a nyereségeket és veszteségeket. Minden folyamat többször végre hajtható, mivel mind a táblák mind a diagram folyamatosan lefrissül.



7. ábra Kimutatások

2.6.1. Teljes kimutatás adott időszakra

Teljes kimutatás esetén a kiadások és bevételek együttesen jelennek meg egy táblában összevonva, illetve diagramos formában is.

A rendszer teljes kimutatás futtatásakor, illetve a kimutatások ablakba való belépéskor létrehoz egy 'Cash_Flow' nevezetű táblát, amelybe bele gyűjti a kiadásokat és bevételeket az adott időszakra vonatkozóan, valamint belépéskor az össze adatot. Ezt a táblát hasznával jeleníti meg a rendszer a diagrammot, illetve magát a táblát.

22. táblázat Teljes kimutatás adott időszakra

Név	Teljes kimutatás adott időszakra	
Cél	Adott időszakra való bevételek és kiadások kimutatás.	
Előfeltétel	Bevételek és kiadások tábla nem üres.	
Sikeres lefutás	Teljes kimutatás megjelenik.	
Sikertelen lefutás	Nem jelenik meg kimutatás.	
Elsődleges aktor	Felhasználó.	
Kiváltó esemény	Kimutatás kezelő felület megnyitása.	
Fő lépések	Lépés	Tevékenység
	1	Kimutatás kezelő felület megnyitása.
	2	Kezdő dátum meghatározása.
	3	Befejező dátum meghatározása.
	4	Kimutatás futtatása.
Kiegészítések	Lépés	Elágazó tevékenység
	4.1	Sikertelen kimutatás. Nincs az adott időszakban bevétel és kiadás sem.
	4.2	Dátumok javítása.
	4.3	Futtatás újból

2.6.2. Bevételek kimutatása adott időszakra

Adott időszakon belül megmutatja milyen bevételek történtek az üzletbe.

23. táblázat Bevételek kimutatása adott időszakra

Név	Bevételek kimutatása adott időszakra	
Cél	Adott időszakra való bevételek kimutatás.	
Előfeltétel	Bevételek tábla nem üres.	
Sikeres lefutás	Kimutatás megjelenik.	
Sikertelen lefutás	Nem jelenik meg kimutatás.	
Elsődleges aktor	Felhasználó.	
Kiváltó esemény	Kimutatás kezelő felület megnyitása.	
Fő lépések	Lépés	Tevékenység
	1	Kimutatás kezelő felület megnyitása.

Kiegészítések	2	Kezdő dátum meghatározása.
	3	Befejező dátum meghatározása.
	4	Kimutatás futtatása.
	Lépés	Elágazó tevékenység
	4.1	Sikertelen kimutatás. Nincs az adott időszakban bevétel.
	4.2	Dátumok javítása.
	4.3	Futtatás újból

2.6.3. Kiadások kimutatása adott időszakra

Adott időszakon belül megmutatja milyen kiadások történtek az üzletbe.

24. táblázat Kiadások kimutatása adott időszakra

Név	Kiadások kimutatása adott időszakra	
Cél	Adott időszakra való kiadások kimutatás.	
Előfeltétel	Kiadások tábla nem üres.	
Sikeres lefutás	Kimutatás megjelenik.	
Sikertelen lefutás	Nem jelenik meg kimutatás.	
Elsődleges aktor	Felhasználó.	
Kiváltó esemény	Kimutatás kezelő felület megnyitása.	
Fő lépések	Lépés	Tevékenység
	1	Kimutatás kezelő felület megnyitása.
	2	Kezdő dátum meghatározása.
	3	Befejező dátum meghatározása.
	4	Kimutatás futtatása.
Kiegészítések	Lépés	Elágazó tevékenység
	4.1	Sikertelen kimutatás. Nincs az adott időszakban kiadás.
	4.2	Dátumok javítása.
	4.3	Futtatás újból

2.7. E-mail küldési funkciók

Több e-mailt is legenerál a rendszer a futtatás során. Ebben a pontban azon levél küldési funkciókat ismertetném, amelyek valamilyen eseményhez kapcsolódóan hajtódnak végre a háttérben.

Az e-mail küldési funkciót több esetben is hasznosítottam. Ilyen a törzsvásárlói regisztráció, pontok lejáráásával kapcsolatos üzenet küldés, valamint kívánság teljesüléséről történő e-mail generálás.

2.7.1. Törzsvásárlói regisztráció esetén

Amikor egy új törzsvásárló beregisztrálásra kerül a rendszer a vásárló által megadott e-mail címre küld egy előre meghatározott szöveggel egy üzenetet, amelybe a rendszer által előállított törzsvásárlói kódot is meghatározza. Ezt a funkciót egyszer fut csak le a regisztráció során.

2.7.2. Kívánság listában szereplő könyv beérkezése alkalmával

Abban az esetben mikor egy könyv beérkezik az üzletbe és felvételre kerül az adatbázisba, akkor a rendszer megvizsgálja a Kívánság lista táblát, hogy abban szerepel-e az adott könyv. Ha szerepel a táblában akkor erről egy előre megírt e-mail formájában értesíti a vásárlót.

2.8. Automatizált funkciók

A rendszerben több olyan funkció van, amelyek automatikusan futnak le a rendszer indításakor adott feltételeknek megfelelően. Ezek a funkciók nem kapcsolódnak más funkciók futtatásához.

2.8.1. Előző éves pontok felhasználhatóságával kapcsolatban

Ez a funkció ellentétben a törzsvásárlási regisztráció esetében küldött e-maillal, nem egy másik tevékenység automatikus mellékfunkciója. Ebben az esetben a rendszer meghatározott időszakon belül a rendszer indulásakor automatikusan megvizsgálja a Törzsvásárlók táblában szereplő személyek *Előző éves pontjait*. Ahol ennek értéke az adott időszakon belül nem nulla, annak a vásárlónak egy előre megírt e-mail küld a saját *előző éves pontok* értékével.

A rendszer mostani változatában az adott időszakban történő minden indítás esetén végbe megy ez a funkció. Ennek kiküszöbölését a jövőbeli lehetőségek pontban fejtem ki részletesebben.

2.8.2. Törzsvásárlói pontok év elejei átírása

Az év első napját követően a rendszer az indításkor automatikusan megvizsgálja, hogy a törzsvásárlók táblában szereplő vásárlók pontjai átírásra kerültek-e már. Ha nem akkor azokat frissíti és átírja. Ilyenkor két dolgot vizsgál meg a táblában. Egy részt, hogy a vásárlónak vannak-e *aktuális pontjai*, ha vannak, akkor, pedig azt, hogy vannak-e *előző éves* pontjai. Abban az esetben, ha nincsenek *előző éves pontok*, viszont vannak *aktuálisak*, akkor a rendszer át írja őket. Ha pedig olyan eset áll fent, hogy *aktuális pontok* nincsenek, de *előző évesek* vannak, vagy pedig *aktuális* és *előző éves pontok* is vannak, akkor a rendszer az adott törzsvásárlónál nem frissíti tábla adott sorát. A rendszerben, ez pillanatnyilag az év első 7 napján történik meg indításkor. Ezzel kizárva annak lehetőségét, hogy az üzlet nyitva nem tartása miatt a pontok nem íródnak át.

Egy ezzel a funkcióval kapcsolatos probléma megoldást a jövőbeli lehetőségek között tárgyalom.

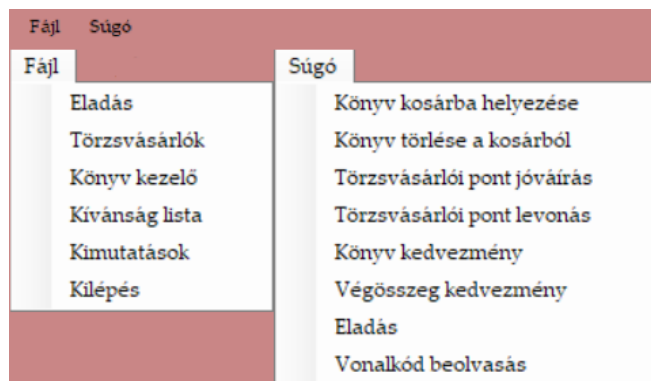
2.8.3. Előző éves pontok törlése

A rendszer képes az *előző éves pontok* törlésére.

Egy meghatározott időszakban a rendszer minden induláskor végig megy a Törzsvásárlók táblán és megvizsgálja, hogy van-e olyan vásárló, akinek van még *előző éves pontja*. Ezen vásárlók *azonosítójának* létrehoz egy listát és összegyűjti benne őket. Mikor a vizsgálattal végzett, a lista alapján lefrissíti a táblázatot és nullázza azon helyeken az *előző éves pontokat*, amely *azonosító* szerepel az erre a célra használt listában. A frissítés lefutását követően a rendszer törli a létrehozott listát.

2.9. Menü

Minden ablakban megtalálható egy menü sáv, amely tartalmaz egy Fájl és egy Súgó fület.



8. ábra Menü sáv

2.9.1. Fájl

A Fájl fülre kattintva megjelennek további fülek, amelyekre kattintva átléphetünk más formokba. Minden formban más és más a Fájl fülben található sorok, mindegyiknél csak a többi form sávja található, illetve egy kilépés sáv, amely segítségével a rendszert leállíthatjuk.

2.9.2. Súgó

A súgó a Fájlhoz hasonlóan további sávokból áll. Minden formnak sajátos a súgója, mivel minden esetben a súgóban található meg a felhasználó az aktuális formhoz tartozó funkciók leírását. Ezzel is próbáltam egy plusz segítséget belevinni a felhasználónak a rendszer használatához.

3. Fejlesztői dokumentáció

Ebben a fejezetben az előzőekben bemutatott funkciók fejlesztői szemmel történő bemutatása a célom, illetve a program kódokkal történő ismertetése. A felhasználói dokumentáció pontjait alapul véve szemléltetem az egyes funkciók kódjait. Nem minden funkciót ismertetek, mivel több olyan is van a különböző részekben, aminek kódjaiban a változó neveken kívül egyezés található, így azok külön történő szemléltetése hasztalan lenne.

Néhány kód részletet, a következő fejezetben az adatbázis bemutatásánál ismertetek.

3.1. Fő ablak

Ahogy már fentebb is említettem a program indulásakor a főablak jelenik meg, ahonnan az ott található gombok segítségével tudunk tovább menni a többi ablakba. Emellett egy DataGridView segítségével láthatjuk az aktuális könyvek listáját is.

A következő kódrészlet egy példa az ablakokba történő átléptetésre, ami ebben az esetben a Törzsvásárlók ablakba visz minket át. Ez a funkció gombnyomás segítségével hatódik végre.

```
private void regular_customer_Click(object sender, EventArgs e)
{
    RegularCustomerForm regCust = new RegularCustomerForm();
    regCust.Show();
    this.Hide();
}
```

A DataGridViewban a megjelenítés mellett az oszlopok automatikus méretezése is történik. A megjelenítés kódja:

```
public void display_data()
{
    connection.Open();
    SqlCommand cmd = connection.CreateCommand();
    cmd.CommandType = CommandType.Text;
    cmd.CommandText = "select [ISBN], [Author] as [Szerző], [Title] as [Cím],
[Release_Year] as [Kiadás], [Genre] as [Műfaj], [Publisher] as [Kiadó], [Pagenumber]
as [Oldal-\nszám], [Condition] as [Állapot]" +
```

```

        ", [Binding] as [Kötés], [Purchase_Price] as [Beszer. ár], [Selling_Price] as
        [Elad. ár] from [Books]";
        cmd.ExecuteNonQuery();
        DataTable dta = new DataTable();
        SqlDataAdapter dataadp = new SqlDataAdapter(cmd);
        dataadp.Fill(dta);
        dataGridView.DataSource = dta;
        connection.Close();
        dataGridView.AutoSizeColumnsMode();
        dataGridView.AutoSizeColumnsMode =
        DataGridViewAutoSizeColumnsMode.AllCells;
    }

```

Ezen kódrészletek mellett más is található még a MainMenu.csben, de ezeken más funkciók ismertetés során mutatom be.

3.2. Vonalkódolvasó

Az alkalmazás fejlesztése során felmerült egy olyan lehetőség, hogy egy vonalkódolvasó beépítésével tegyük kényelmesebbé, gyorsabbá a rendszer használatát. A következőkben az ehhez szükséges bővítmények és technológiák használatát ismertetem.

Mindenekelőtt a vonalkód beépítéséhez szükségem volt egy bővítmény letöltéséhez. Ehhez a Visual Studio NuGet Package Managerében kellett letölteni a ZXing.Net bővítményt. A ZXing.Net egy olyan bővítménykönyvtár, amely támogatja a vonalkódok generálását, valamint dekódolását is.

A következő vonalkódokat támogatja a dekóder: UPC-A, UPC-E, EAN-8, EAN-13, Code 39, Code 93, Code 128, ITF, Codabar, MSI, RSS-14 (és annak minden változata, QR Code, Data Matrix, Aztec és PDF-417.

A kódoló pedig ezeket a formátumokat: UPC-A, EAN-8, EAN-13, Code 39, Code 128, ITF, Codabar, Plessey, MSI, QR Code, PDF-417, Aztec, Data matrix.

A következőkben azon környezeteket sorolom fel, amik támogatják a ZXinget:

.Net 2., 3.5, 4.0, 4.5, 4.6 és a 4.7; winmd; .NET Standard/ .Net Core/ UWP; Portable Class Library; Unity3D; Xamarin.Android, CoreCompat.System.Drawing, ImageSharp, OpenCVSharp, Magick, Kinect V1 és V2 kötések; illetve a COM együttműködést, valamint használható VBA-val. [5]

A rendszerben a ZXing meghívását követően a következő kód részlet megírásával készítettem el a programot.

```

using (OpenFileDialog ofd = new OpenFileDialog() { Filter = "JPG| *.jpg" })

```

```
{
    if(ofd.ShowDialog() == DialogResult.OK){
        pB_barcode.Image = Image.FromFile(ofd.FileName);
        BarcodeReader reader = new BarcodeReader();
        var result = reader.Decode((Bitmap)pB_barcode.Image);
        if(result != null){
            tB_barcode.Text = result.ToString();
        }
    }
}
```

A vonalkódolvasó pillanatnyilag képről való felismerésként funkcionál. A jövőbeli lehetőségek pontnál ezt jobban is kifejtem, hogy hogyan lehetne tovább fejleszteni.

3.3. Törzsvásárlói kód

Az a kód részlet, amely a következőekben bemutatásra kerül, a törzsvásárló kód kialakítását határozza meg.

A törzsvásárlói kód az egyetlen olyan azonosító az adatbázisban, amely nem automatikusan generálódik le, hanem a rendszerben a regisztráláskor megadott adatok alapján generálódik le. A felhasználói dokumentációnál már ismerttettem, hogy milyen adatok felhasználásával jön létre az azonosító.

Előfordulhat olyan eset, mikor két személy adatai annyira hasonlóak, hogy teljes egyezés lép fel a kód generálásakor. Ezt a problémát olyan módon oldottam meg, hogy miután a rendszer legenerálja az azonosítót, megvizsgálja, hogy van-e ilyen egyedi kód már a rendszerbe. Ha nincs, akkor egyszerűen a vásárló beregisztrálásra kerül a rendszerbe. Ellenben, ha van már a generált kóddal megegyező azonosító, akkor a rendszer az egyedi kód első karakterét, ami egy szám, megnöveli az adatbázis törzsvásárlók táblájában található sorok számával. Így kiküszöbölésre került az az eshetőség, hogy egyezés esetén az egyszer már módosított azonosítóhoz hasonló még egyszer szerepeljen a táblában.

A kód először megvizsgálja, hogy minden szükséges adatot megadtunk-e. Hiba esetén erről egy felugró ablakban értesít. Ezt követően létrehozza az adatok alapján az azonosítót, amit utána szintén ellenőriz, szükség esetén módosít. Ezt követően egy INSERT parancs végrehajtásával felviszi a táblába az új törzsvásárlót, valamint elküldi a regisztrálásról szóló e-mailt. Végül pedig a textBoxokat alaphelyzetbe állítva, megváltoztatja azok színét és a bele írt szöveget is.

```
private void new_regcust_Click(object sender, EventArgs e)
```

```

{
    if ( tB_name.Text != "" && tB_name.Text != "Teljes név" && tB_address.Text
!= "" && tB_address.Text != "Lakcím"
        && cB_gender.Text != "" && cB_gender.Text != "Nem" && tB_phone.Text
!= "" && tB_phone.Text != "Telefonszám"
        && tB_email.Text != "" && tB_email.Text != "Email cím")
    {
        string format = "yyyy. MM. dd";
        DateTime born_date = dtP_born_date.Value;
        string born_date_string = born_date.ToString(format);
        string date = null;
        date += born_date_string[0].ToString() + born_date_string[1].ToString() +
born_date_string[2].ToString() + born_date_string[3].ToString() +
born_date_string[6].ToString() + born_date_string[7].ToString();
        string gender_number, gender;
        if (cB_gender.Text == "Nő") {
            gender_number = "1";
            gender = "Nő";
        } else {
            gender_number = "2";
            gender = "Férfi";
        }
        string nick_name = tB_name.Text;
        string name = null;
        name += nick_name[0].ToString() + nick_name[1].ToString().ToUpper();
        string reg_cust = null;
        reg_cust += date.ToString() + gender_number.ToString() + name.ToString();
        connection.Open();
        SqlCommand cmd = connection.CreateCommand();
        SqlDataReader reader = (null);
        cmd.CommandType = CommandType.Text;
        cmd.CommandText = "select Count(Name) as count from
[Regular_Customers] where [Regular_Customer_ID] = '" + reg_cust.ToString() + "'";
        cmd.ExecuteNonQuery();
        reader = cmd.ExecuteReader();
        reader.Read();
        int count = int.Parse(reader["count"].ToString());
        reader.Close();
        if (count == 0)
        {
            cmd.CommandText = "insert into [Regular_Customers]
(Regular_Customer_ID, Name, Address, [Born_Date], Gender, [Phone_Number],
[Email_Address], [Current_Points], [Previous_Year_Points]) " +
            "values ('" + reg_cust.ToString() + "', '" + tB_name.Text + "', '" +
tB_address.Text + "', '" + born_date.ToString(format) + "', '" +
            "'" + gender.ToString() + "', '" + tB_phone.Text + "', '" + tB_email.Text +
            "', '" + 0 + "', '" + 0 + "');"
            cmd.ExecuteNonQuery();
        } else
        {

```

```

        cmd.CommandText = "select Count(Name) as count from
[Regular_Customers]";
        cmd.ExecuteNonQuery();
        reader = cmd.ExecuteReader();
        reader.Read();
        int sum = int.Parse(reader["count"].ToString());
        reader.Close();
        reg_cust = reg_cust.Remove(0, 1);
        int first_char = int.Parse(born_date_string[0].ToString());
        first_char += sum;
        reg_cust = first_char.ToString() + reg_cust;
        cmd.CommandText = "insert into [Regular_Customers]
(Regular_Customer_ID, Name, Address, [Born_Date], Gender, [Phone_Number],
[Email_Address], [Current_Points], [Previous_Year_Points]) " +
        "values ('" + reg_cust.ToString() + "', '" + tB_name.Text + "', '" +
tB_address.Text + "', '" + born_date.ToString(format) + "', '" +
        "'" + gender.ToString() + "', '" + tB_phone.Text + "', '" + tB_email.Text +
        "', '" + 0 + "', '" + 0 + "')";
        cmd.ExecuteNonQuery();
    }

    send_mail(tB_email.Text.ToString(), reg_cust.ToString());
    connection.Close();
    tB_name.ForeColor = Color.Gray;
    tB_name.Text = "Teljes név";
    tB_address.ForeColor = Color.Gray;
    tB_address.Text = "Lakcím";
    cB_gender.ForeColor = Color.Gray;
    cB_gender.Text = "Nem";
    tB_phone.ForeColor = Color.Gray;
    tB_phone.Text = "Telefonszám";
    tB_email.ForeColor = Color.Gray;
    tB_email.Text = "Email cím";
    display_data();
}
else
{
    MessageBox.Show("Hiányosak az adatok!");
}
}

```

3.4. E-mail küldési funkciók

Az e-mailek elküldéséhez a .NET MailMessage osztályát használtam. Ehhez szükség volt olyan adatok megadására, mint a feladó e-mail címe, jelszava, a címzett e-mail címe, port szám, EnableSSL tulajdonság. Ezek mellett a legfontosabb az volt, hogy a kód megírása

előtt beimportáljak két névteret a MailMessage osztály eléréséhez. A két importált névtér a System.Net, illetve a System.Net.Mail.

Ezt követően a következő kódrészletben látható sorok megírása volt a feladatom.

```
NetworkCredential login;
SmtpClient client;
MailMessage msg;
public void send_mail(string e_mail) {
    try
    {
        login = new NetworkCredential("horeczky95@gmail.com", „Louis123!!!“);
        client = new SmtpClient("smtp.gmail.com");
        client.Port = 587;
        client.EnableSsl = true;
        client.Credentials = login;
        msg = new MailMessage { From = new MailAddress("horeczky95@gmail.com")
    };
    msg.To.Add(e_mail);
    msg.Subject = "Keresett könyv beérkezett";
    msg.Body = "Tisztelt Vásárló!<br /><br />" +
        "Az ön által keresett könyv mostantól megtalálható boltunkba.<br /><br />" +
        "Ez egy automatikus email kérem ne válaszoljon rá.<br /><br />Antikvár könyves
    bolt<br /><br />" +
        "Üdvözlettel: Antikvár könyves bolt";
    msg.BodyEncoding = Encoding.UTF8;
    msg.IsBodyHtml = true;
    msg.Priority = MailPriority.Normal;
    msg.DeliveryNotificationOptions = DeliveryNotificationOptions.OnFailure;
    string userstate = "Sending...";
    client.SendAsync(msg, userstate);
    } catch (Exception ex)
    {
        Console.WriteLine(ex.ToString());
        MessageBox.Show("Sikertelen küldés!" + ex.ToString());
    }
}
```

Az EnableSSL tulajdonság általában az SMTP levelezőszerver eléréséhez szükséges SSL használatát határozza meg. Manapság a Gmail különböző biztonsági okokból kifolyólag Hitelesítési hibát jelenít meg. Ezt úgy tudtam kiküszöbölni, hogy a Google fiókomban engedélyeznem kellett a Kevésbé biztonságos alkalmazások hozzáférését a fiókhoz. [6]

A send_mail() metódus megírását követően annak meghívása a megfelelő helyen történt meg. Az ismertetett kód mellett olyan metódus írása is végbe ment, ahol a meghíváskor értékadás is történt az e-mail cím átadását követően. Ilyen volt a

törzsvásárlói regisztráláskor küldendő email, ahol a törzsvásárlói kód átadása történt, illetve az előző éves pontokról küldött e-mail, itt a pontok átadását kellett megoldani.

3.5. Automatizált funkciók

Az automatikusan lefutó funkciók az adatbázist vizsgálják, illetve szükség esetén módosítják. Három esetben írtam meg ilyen funkciót. Az egyik az előző éves pontok vizsgálata után szükség esetén emailt küld. A másik két funkció a törzsvásárlók táblán hajt végre módosítást. A két funkció egy helyen lett deklarálva a szükséges feltételek megadásával. A következő néhány sorban ennek bemutatása látható.

```
public void update_points()
{
    string format = "MM.dd";
    DateTime today = DateTime.Parse(DateTime.UtcNow.ToString(format));
    DateTime update_date = DateTime.Parse( new DateTime(2022, 01,
01).ToString(format));
    DateTime clear_date = DateTime.Parse(new DateTime(2022, 04,
30).ToString(format));
    connection.Open();
    SqlCommand cmd_previous_points = connection.CreateCommand();
    SqlDataReader read = (null);
    cmd_previous_points.CommandText = ("select SUM(Previous_Year_Points) as
sum from Regular_Customers");
    cmd_previous_points.ExecuteNonQuery();
    read = cmd_previous_points.ExecuteReader();
    read.Read();
    float previous_points = float.Parse(read["sum"].ToString());
    read.Close();
    if (previous_points == 0)
    {
        if (today >= update_date && today < clear_date )
        {
            SqlCommand cmd_previous_points_update =
connection.CreateCommand();
            cmd_previous_points_update.CommandType = CommandType.Text;
            cmd_previous_points_update.CommandText = "update
[Regular_Customers] set [Previous_Year_Points] = [Current_Points]";
            cmd_previous_points_update.ExecuteNonQuery();
            SqlCommand cmd_current_points_update = connection.CreateCommand();
            cmd_current_points_update.CommandType = CommandType.Text;
            cmd_current_points_update.CommandText = "update [Regular_Customers]
set [Current_Points] = 0";
            cmd_current_points_update.ExecuteNonQuery();
        }
    } else if (previous_points > 0)
```

```
{
    if (today >= clear_date)
    {
        SqlCommand cmd_previous_points_update =
connection.CreateCommand();
        cmd_previous_points_update.CommandType = CommandType.Text;
        cmd_previous_points_update.CommandText = "update
[Regular_Customers] set [Previous_Year_Points] = 0";
        cmd_previous_points_update.ExecuteNonQuery();

    }
}
connection.Close();
}
```

3.6. Menü

Ahogy az előző fejezetben elmondtam, a menüben egy fájl és egy súgó fület találhatunk.

A fájl menüben történő sávok megnyomásával tudunk tovább lépni más ablakokba. Ezt a következő pár sorban látható kód megírásával tettem lehetővé. A kód az eladás ablakba való átlépést mutatja be.

```
private void saleToolStripMenuItem_Click_1(object sender, EventArgs e)
{
    SaleForm sale = new SaleForm();
    sale.Show();
    this.Hide();
}
```

A súgó menüben található sávokra kattintva felugró ablakokban adtam meg az egyes funkciók használatának menetét. Ezt MessageBoxok segítségével oldottam meg, mivel úgy gondoltam ez a legegyszerűbb formája egy szöveg megjelenítésének a programban. Ezzel nem használtam felesleges helyet el az adott ablakok területéből sem.

A következő kód részlet a könyvek törlését leíró súgó boxot adja vissza.

```
private void book_delete_help_click(object sender, EventArgs e)
{
    MessageBox.Show("Könyv törlése!\n\n" +
        "A könyv azonosítóját megadva a törlés gomb megnyomásával lehet a könyvet törölni.");
}
```

3.7. Formázás

A fent említett kódok és funkciók mellett kódokkal történő formázásokat is készítettem. Ilyenek a textBoxokba való bele kattintás, illetve azokból való kilépés.

A textBoxba való belépés esetén, a benne található szöveg eltűnik és a betű színe feketére vált. Az ezt megoldó kódrészlet a következő:

```
private void tB_ISBN_Enter(object sender, EventArgs e)
{
    if(tB_ISBN.Text == "ISBN")
    {
        tB_ISBN.Text = "";
        tB_ISBN.ForeColor = Color.Black;
    }
}
```

A boxokból való kilépéskor, ha üresen hagytuk őket, a szín szürke értéket kapja meg, valamint egy meghatározott szöveg jelenik meg bennük.

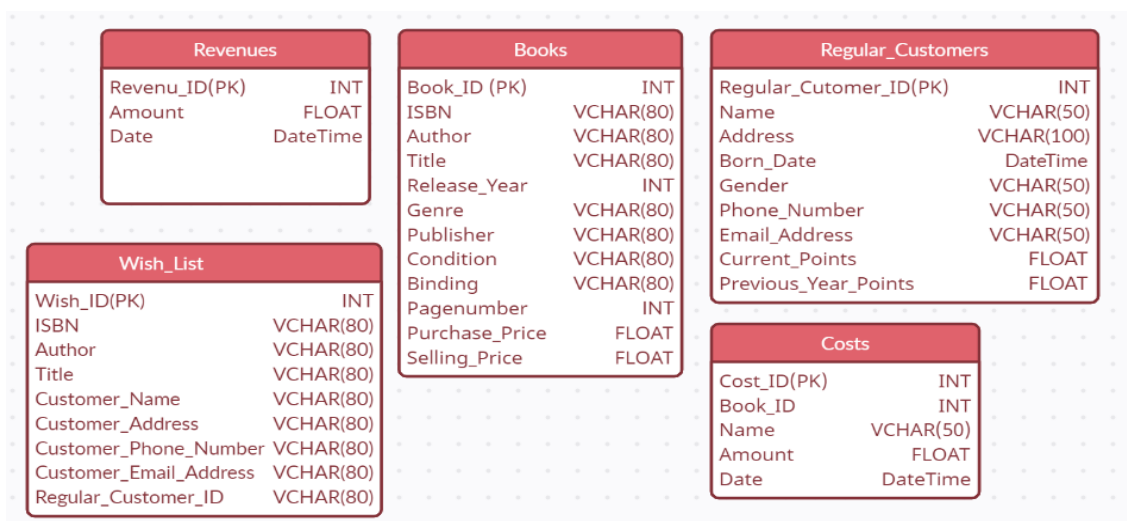
```
private void tB_ISBN_Enter(object sender, EventArgs e)
{
    if(tB_ISBN.Text == "")
    {
        tB_ISBN.Text = " ISBN ";
        tB_ISBN.ForeColor = Color.Gray;
    }
}
```

Az előzőekben felsorolt kódok, illetve funkciók mellett más kód részleteket az adatbázis bemutatása során helyeztem el a szövegben.

4. Az adatbázis

Szakdolgozatom főfunkciói az adatbázis kezeléshez közelálló folyamatok. Szinte minden egyes funkció hasznosítja az adatbázist, azt bővíti, frissíti, illetve tisztítja.

Dolgozatomat az adatbázis létrehozásával, illetve megtervezésével kezdtem. Az elképzelés és megvalósult adatbázis között minimális különbségek adódtak.



9. ábra Az adatbázis felépítése

4.1. Adatbázis létrehozása, tervezése

Először a C# programozás tárgyamon használt „DB Browser for SQLite” alkalmazást használtam az adatbázis elkészítéséhez. Itt megterveztem a táblákat, meghatároztam a táblák oszlopait, illetve azok típusait.

A program megírásának kezdete után döntöttem úgy, hogy adatbázisomat át helyezem egy a Visual Studio-ba beépített Microsoft SQL Server Database Project-be. Ezzel megkönnyítve a program adatbázishoz való hozzáférését, valamint így egy helyen, egy alkalmazáson belül tudtam alakítani szükség esetén a rendszer fő komponenseit. Az MS SQLServer-ről bővebb kifejtést a bevezetésnél lehet olvasni.

4.1.1. MSSQL

Az adatbázis kezeléséhez (létrehozásához, felvitelekhez, törléséhez, illetve módosításához) egy adott szerverre van szükségünk. Ebben az esetben én a .NET által prioritást élvező MS SQL SERVER-t használtam fel az adatbázisom kezelésére. Ezen felül sokféle adatbázis szerveret támogat a .NET környezet, viszont a dolgozatom elkészítése szempontjából is ez a lehetőség volt a legkézenfekvőbb.

A Visual Studio elindítását követően a Server Explorer nézetben új adatbázist tudunk létrehozni, illetve itt tudunk táblákat definiálni, ezt követően feltölteni, már meglévő adatbázisokat megtekinteni, módosítani, törölni akár, vagy pedig tovább bővíteni.

Ha már előzőleg létrehoztunk egy adatbázist a Visual Studio-n kívül, akkor az a Server Explorerben nem lesz látható. A láthatóságához a „DataConnections” feletti egérfültre kattintva az Add Connection-t kell használnunk és a megnyíló panelen azonosíthatjuk be az adatbázist.

Ha elkészítettük az adatbázisunkat, akkor ezt követően minden formban, ahol a funkciókkal használatba igényeljük venni az adatbázisban szereplő valamely táblát, ott definiálnunk kell egy csatlakozást az adatbázishoz. Ezt a következő kód részlettel tehetjük meg.

```
SqlConnection connection = new SqlConnection(@"Data Source =  
(LocalDB)\MSSQLLocalDB;  
AttachDbFilename=D:\SULI\SZAKDOLGOZAT\WINDOWSFORMSAPP1\WINDO  
WSFORMSAPP1\ANTIQUEDB.MDF;Integrated Security = True");
```

Ennek használatához szükségünk van az SQLClient-re is, amit a következő módon definiálhatunk a programban.

```
using System.Data.SqlClient;
```

Ha megvagyunk a kapcsolat definiálásával, akkor a következő legfontosabb részlet a kapcsolat megnyitása, illetve annak lezárása a szükséges helyen. Ezeket a csatlakozásnál megadott változó névvel tudjuk meghívni a következő parancsokkal.

```
connection.Open();
connection.Close();
```

Ezek között kell megadnunk a szükséges utasításokat az adatbázisba való módosítások, felvitelek, illetve törlések során.

A következő lépés egy SqlCommand objektum létrehozása, amelyet a végrehajtandó SELECT vagy más utasítás adatbázisba való küldésére használunk fel.

```
SqlCommand cmd = connection.CreateCommand();
```

Ezt követően a Command objektum CommandText, illetve CommandType tulajdonságait is be kell állítanunk. A CommandType-val tudjuk a SqlCommand objektum típusát beállítani. A CommandText használatával tulajdonságokat állíthatunk be egy SELECT vagy INSERT, illetve más utasítások esetén.

Ezek meghatározása után az ExecuteNonQuery() metódus használatával tudjuk a parancsot elküldeni az adatbázis szerverre felé.

Az előzőekben említett parancsok végre hajtása után több lehetőségünk is van. Egyrészt egy SELECT utasítás megírásával és egy DataGridView segítségével megtudjuk a programban jeleníteni a következő parancssorok segítségével egy adott tábla tartalmát. Ez a Könyvek táblát mutatja meg jelen esetben.

```
connection.Open();
SqlCommand cmd = connection.CreateCommand();
cmd.CommandType = CommandType.Text;
cmd.CommandText = "select * from [Books]";
cmd.ExecuteNonQuery();
DataTable dta = new DataTable();
SqlDataAdapter dataadp = new SqlDataAdapter(cmd);
dataadp.Fill(dta);
dataGridView_books.DataSource = dta;
connection.Close();
```

Lehetőségünk van adat kinyerésre az adatbázisból. Ehhez az SqlDataReader objektum használata szükséges. Az SqlCommandhoz hasonlóan létre kell hoznunk az objektumot, egy null érték adással. A Command lezárását követően a Read(), tagfüggvénysegítségével tudjuk a DataReaderben lévő sort beolvasni. Az így kapott értéket egy ugyanolyan típusú változó értékének átadva, használhatjuk ezt követően az így megszerzett adatot. Az SqlDataReader-t is a SqlCommandhoz hasonlóan le kell zárni a megfelelő működéshez.

A következő néhány sorba szemléltetném, hogyan is néz ki egy ilyen kód részlet. Ebben az esetben a Törzsvásárlók táblából szerettem volna megkapni az adott törzsvásárlói kódhoz tartozó *Aktuális pontokat*. [7] [8]

```
connection.Open();
SqlCommand cmd_regcst = connection.CreateCommand();
cmd_regcst.CommandType = CommandType.Text;
```

```

SqlDataReader read = (null);
cmd_regcust.CommandText = ("select * from Regular_Customers where
Regular_Customer_ID = '" + tB_regcust_ID.Text + "'");
cmd_regcust.ExecuteNonQuery();
read = cmd_regcust.ExecuteReader();
read.Read();
float current_points = float.Parse(read["Current_Points"].ToString());
read.Close();
connection.Close();

```

Minden esetben pontosan meg kell határoznunk a lekért adat típusát, mert bármilyen egyenlőtlenség is hibás lefutást eredményez.

A következő pár sorban néhány a rendszerben található kód részlettel szeretném szemléltetni, mely utasításokat is használtam a fejlesztés során.

Az első egy INSERT utasítás. Ennek a parancsnak a segítségével vihetünk fel egy adott táblába új értékeket.

```

connection.Open();
SqlCommand cmd_cash = connection.CreateCommand();
cmd_cash.CommandType = CommandType.Text;
cmd_cash.CommandText = "insert into [Cash_Flow] (Amount, Date) select Amount*(-1), Date from Costs";
cmd_cash.ExecuteNonQuery();
cmd_cash.CommandText = "insert into [Cash_Flow] (Amount, Date) select Amount, Date from Revenues";
cmd_cash.ExecuteNonQuery();
connection.Close();

```

A következő egy UPDATE utasítás. Ezzel az utasítással egy tábla mezőit tudjuk módosítani a megadott feltételekkel. A következő sorokban egy részletet láthatunk egy update utasításból.

```

cmd.CommandText = "update [Wish_List] set ISBN = '" + tB_ISBN.Text + "' where
Wish_ID = '" + int.Parse(tB_ID.Text) + "'";

```

Végül pedig egy DELETE utasítást szeretnék mutatni, amellyel egy tábla tartalmát tudjuk törölni vagy pedig meghatározott szűrőkkel csak adott sorokat. Ebben az esetben is csak a kód az ismertetés szempontjából legfontosabb részét mutatom be, mivel az előzőekben már szemléltettem az utasításokhoz szükséges teljes kód részletet.

```

cmd.CommandText = "delete from [Books] where [Book_ID] = '" +
int.Parse(tB_ID.Text) + "'";

```

Ezen felül még a fentebb bemutatott SELECT utasítást használtam. A rendszer futtatásakor létrehozott ideiglenes tábla elkészítéséhez a CREATE TABLE utasítást használtam, majd ennek a táblának a törlésére a DROP TABLE-t.

```

cmd_cash.CommandText = "drop table if exists [Cash_Flow]";
cmd_cash.ExecuteNonQuery();
cmd_cash.CommandText = "create table [Cash_Flow] ([Cash_Flow_ID] INT NOT NULL IDENTITY (1, 1), Amount FLOAT NOT NULL, Date DATETIME NOT NULL, PRIMARY KEY CLUSTERED ([Cash_Flow_ID] ASC))";

```

4.2. Felépítése

Az adatbázist 5 állandó táblázat alkotja. Ezek a rendszerben angol nyelven találhatóak a következőképpen. *'Books'*, *'Costs'*, *'Regular_Customers'*, *'Revenues'*, *'Wish_List'* néven futnak. Emellett egy ideiglenesen létrehozott tábla is szerepel az adatbázisban a teljes kimutatások futtatásakor, aminek neve *Cash_Flow*.

'Books' tábla

Az adatbázisban központi szerepet a *'Books'* tábla kapta. Ez a tábla az antikvár könyvesboltban található könyvek adatait tartalmazza: *'Book_ID'* a tábla azonosító mezője, *'ISBN'* az ISBN számot tartalmazza, az *'Author'* a szerző, *'Title'* a cím, *'Release_Year'* a kiadás éve, *'Genre'* a műfajt jelöli, *'Publisher'* a kiadó neve, *'Condition'* a könyv állapota, *'Binding'* a könyv külső kötése, *'PageNumber'* az oldalszám, *'Purchase_Price'* a beszerzési ár, és *'Selling_Price'* az eladási ár jelölésre szolgálja. Minden adat megadása a *'Selling_Price'* kivételével kötelező a könyv felvétel alkalmával. A *'Books'* tábla elsődleges kulcs a *'Book_ID'*.

'Costs' tábla

A kiadások kezelésére a *'Costs'* tábla szolgál. Ebben a táblában jelenik meg minden kiadás, ami a könyvekhez kapcsolódik. A tábla mezői a következőek: a *'Cost_ID'* az azonosítót, a *'Book_ID'* a bekerült könyv azonosítót, *'Name'* az adott könyv nevét, *'Amount'* a beszerzési árát, *'Date'* pedig a beszerzés napját jelöli. A tábla elsődleges kulcsa a *'Cost_ID'*.

'Regular_Customers' tábla

Ebbe a táblába azon vásárlók adatait tartjuk nyilván, akik beregisztrálásra kerültek a rendszerbe. A tábla tartalma: *'Regular_Customer_ID'* az egyedi azonosítót, *'Name'* a nevét, *'Address'* a címét, *'Born_Date'* születési idejét, *'Gender'* a nemét, *'Phone_Number'* telefonszámát, *'Email_Address'* e-mail címét, *'Current_Points'* aktuális pontjait, és *'Previous_Year_Points'* az előző éves pontjait adja meg a vásárlónak. Elsődleges kulcsa: *'Regular_Customer_ID'*.

'Revenues' tábla

A bevételek táblában azokat az összegeket találjuk, amelyeket adott vásárlások során a vásárlók kifizettek az üzletben. Emellett a dátumot is tartalmazza, amely napon az adott vásárlás történt. A tábla oszlopainak nevei: *'Revenue_ID'*, *'Amount'*, *'Date'*. Elsődleges kulcsa: *'Revenue_ID'*.

'Wish_List' tábla

A *'Wish_List'* tábla szolgál a kívánságok tárolására. Ebbe a táblába kerülnek fel a vásárlók által keresett könyvek, amik épp nem voltak elérhetőek az üzletben. A tábla tartalma: *'Wish_ID'*, *'ISBN'*, *'Author'*, *'Title'*, *'Cutomer_Name'*, *'Cutomer_Address'*, *'Customer_Phone_Number'*, *'Customer_Email_Address'*, *'Regular_Customer_ID'*. A tábla elsődleges kulcsa a *'Wish_ID'*.

Ezen táblák mellett a kimutatások ablak megnyitásakor, illetve adott időszakra való teljes kimutatáskor egy ideiglenes tábla jön létre, amelyben a kiadások és a bevételek együttesen jelennek meg. Ez a tábla a *'Cash_Flow'* elnevezést kapta. A tábla tartalma a következő: *'Cash_Flow_ID'*, *Amount*, *Date*. Az elsődleges kulcs a *'Cash_Flow_ID'*. Ez a kulcs minden insert parancsnál automatikus növekedik eggyel. Az *'Amount'* értékét a hozzáadásnál határozza meg a rendszer. Ha bevételekről van szó, akkor egyszerűen hozzá adja a táblához az értéket, ha kiadásról, akkor negatív számként adja a rendszer hozzá az értéket. A *'Date'* mezőt szintén a két táblából adja hozzá a rendszer, az alapján, hogy milyen kezdő és befejező dátumot adtunk meg a kimutatás előtt.

5. Tesztelés

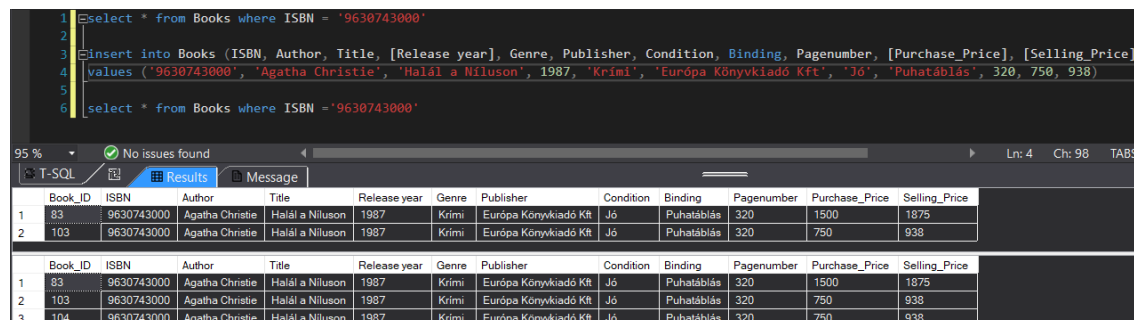
Személy szerint úgy gondolom, hogy egy program elkészítése folyamán az egyik leglényegesebb dolog a tesztelés, hisz ez segíti a fejlesztőt abban, hogy milyen hibák rejlenek esetlegesen a rendszerben, és ezeket szükség esetén javítani lehet. Egy rendszer se hibátlan, szinte lehetetlen tökéletesen hibátlan programot készíteni. Ezért is fontos, hogy egy program folyamatos tesztelés alatt álljon. Akár manuális akár automatizált tesztesetek segítségével, valamint tesztelők bevonásával. Végül, de nem utolsósorban a rendszer végfelhasználója, aki folyamatos tesztelés alatt tartja a rendszert az állandó használattal, és hiba esetén azokat jelezheti a fejlesztőknek, akik ezeket a hibákat kiküszöbölhetik.

Tesztelések során kétféle irányba teszteltem. Először az adatbázist teszteltem az elkészítése után, ezt követően a megírt programot ellenőriztem, továbbá a program teszteléséhez külsős segítséget is kaptam.

5.1. Az adatbázis tesztelése

Az adatbázis elkészítését követően azt folyamatos tesztelés alatt tartottam, adatbázis műveletek megírásával és futtatásával. Insertek, updatek és deletek segítségével vizsgáltam meg minden táblát, hogy megfelelően feltudom-e vinni a szükséges adatokat.

Az első, minden tábla esetében, egy INSERT INTO parancs megadása volt, amivel ellenőriztem, hogy a megadott értékek megfelelően felkerülnek-e az adatbázisba.



```

1 select * from Books where ISBN = '9630743000'
2
3 insert into Books (ISBN, Author, Title, [Release year], Genre, Publisher, Condition, Binding, Pagenumber, [Purchase_Price], [Selling_Price])
4 values ('9630743000', 'Agatha Christie', 'Halál a Niluson', 1987, 'Krimi', 'Európa Könyvkiadó Kft', 'Jó', 'Puhatáblás', 320, 750, 938)
5
6 select * from Books where ISBN = '9630743000'

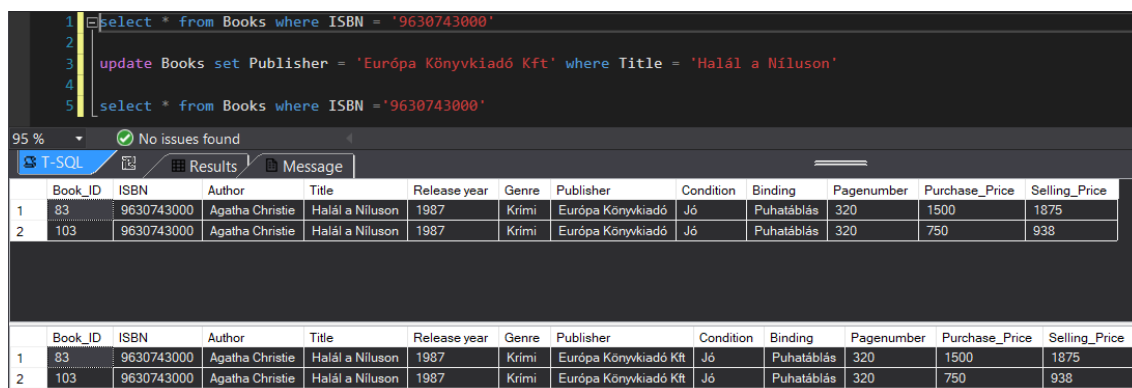
```

Book_ID	ISBN	Author	Title	Release year	Genre	Publisher	Condition	Binding	Pagenumber	Purchase_Price	Selling_Price
83	9630743000	Agatha Christie	Halál a Niluson	1987	Krimi	Európa Könyvkiadó Kft	Jó	Puhatáblás	320	1500	1875
103	9630743000	Agatha Christie	Halál a Niluson	1987	Krimi	Európa Könyvkiadó Kft	Jó	Puhatáblás	320	750	938

10. ábra INSERT parancs

Minden parancsot egy SELECT parancs előzött meg, illetve követett. Ennek segítségével azonnal láthattam, hogy az általam megírt és használt parancs sikeresen futott-e le vagy sem.

A felvitt adatokon az UPDATE parancs segítségével vizsgáltam meg a módosítási lehetőségeket, a megfelelő szűrők meghatározásával.



The screenshot shows a SQL query window with the following commands:

```

1 select * from Books where ISBN = '9630743000'
2
3 update Books set Publisher = 'Európa Könyvkiadó Kft' where Title = 'Halál a Niluson'
4
5 select * from Books where ISBN = '9630743000'

```

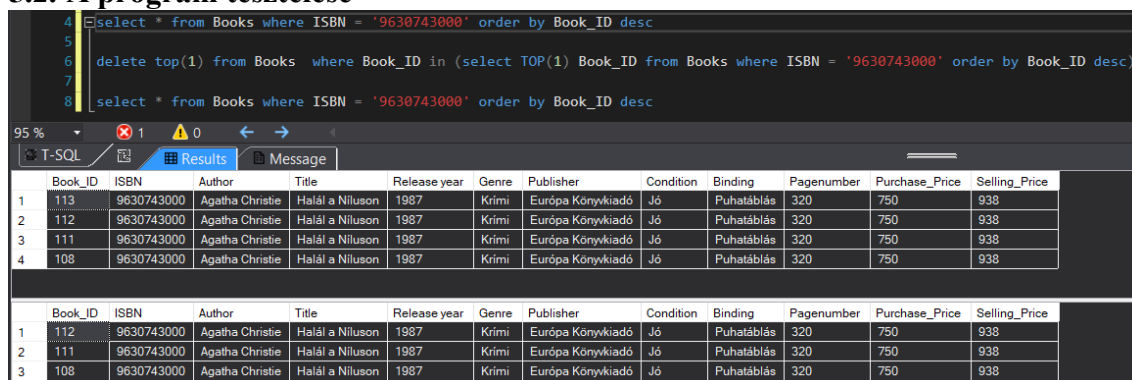
The Results tab displays the following data:

Book_ID	ISBN	Author	Title	Release year	Genre	Publisher	Condition	Binding	Pagenumber	Purchase_Price	Selling_Price
1	83	Agatha Christie	Halál a Niluson	1987	Krimi	Európa Könyvkiadó	Jó	Puhatáblás	320	1500	1875
2	103	Agatha Christie	Halál a Niluson	1987	Krimi	Európa Könyvkiadó	Jó	Puhatáblás	320	750	938

11. ábra UPDATE parancs

Végül pedig a DELETE parancs segítségével próbáltam meg meghatározott feltételek alapján adott sorokat törölni. De végül mindig sikeres lefutást produkált a rendszer a megfelelő szűrések beállításával.

5.2. A program tesztelése



The screenshot shows a SQL query window with the following commands:

```

4 select * from Books where ISBN = '9630743000' order by Book_ID desc
5
6 delete top(1) from Books where Book_ID in (select TOP(1) Book_ID from Books where ISBN = '9630743000' order by Book_ID desc)
7
8 select * from Books where ISBN = '9630743000' order by Book_ID desc

```

The Results tab displays the following data:

Book_ID	ISBN	Author	Title	Release year	Genre	Publisher	Condition	Binding	Pagenumber	Purchase_Price	Selling_Price
1	113	Agatha Christie	Halál a Niluson	1987	Krimi	Európa Könyvkiadó	Jó	Puhatáblás	320	750	938
2	112	Agatha Christie	Halál a Niluson	1987	Krimi	Európa Könyvkiadó	Jó	Puhatáblás	320	750	938
3	111	Agatha Christie	Halál a Niluson	1987	Krimi	Európa Könyvkiadó	Jó	Puhatáblás	320	750	938
4	108	Agatha Christie	Halál a Niluson	1987	Krimi	Európa Könyvkiadó	Jó	Puhatáblás	320	750	938

12. ábra DELETE parancs

A program tesztelés manuális módon történt a GUI és az egyes funkciók saját kézzel történő tesztesetek megadásával.

A program tesztelése részben kapcsolódott az adatbázis teszteléséhez is, mivel a rendszer legtöbb tevékenysége az adatbázisban hajt végre műveleteket.

A tesztelést az első funkciótól az utolsóig folyamatosan hajtottam végre. Minden funkció az elkészítése alatt és után is több formában is tesztelve lett. Ennek köszönhetően nagyon sokszor időben tudtam hiba kezeléssel is foglalkozni, hogy a rendszer hiba esetén

se leállással jelezze a problémát, hanem az erre a célra kitalált és megírt üzenetekkel jelezze a felhasználónak, hogy valamit hibásan hajtott végre.

A rendszert a befejezést követően saját kezűleg is végigpróbáltam, így tesztelve, hogy minden funkció helyesen működik. Valamint emellett külső segítséget is igénybe vettem. A családom bizonyos tagjait kértem a rendszer manuális tesztelésére, olyan módon, hogy elmondtam a rendszer funkcióinak működését és ezt követően a programot elindítva, hagytam, hogy saját belátásuk alapján próbálják ki a rendszer számukra érdekesebb részeit. Ezen két tesztelés során is volt még kijavítandó hiba a rendszerbe.

6. A dolgozat írása alatt felmerülő nehézségek

A dolgozat írása közben több olyan nehézségbe is ütköztem, ami nem feltétlenül a fejlesztés folyamata, illetve a program hibás megírása miatt vetődött fel. Ezen hibák mellett természetesen rengetegszer ütköztem kisebb elírásokból, rosszul definiált folyamatokból keletkező hibákkal is, de ezeket láttam inkább kifejtendő nehézségeknek.

6.1. Gmail által kevésbé biztonságosnak vélt bejelentkezési technika

Az e-mail küldés elkészítése után lépett fel az az akadály, hogy a Gmail rendszerét használva harmadik félként kevésbé biztonságos hozzáférésnek nyilvánította a Gmail a rendszer által történő bejelentkezést és e-mail küldést. Erről az első e-mail küldési próbálkozásnál a Google egy e-mailben értesített, hogy megakadályozott egy bejelentkezési kísérletet. Az e-mailben leírták, hogy miért veszélyes ez a bejelentkezési mód, illetve, hogy hogyan alkalmazható ez ennek ellenére is. A Google fiókomban be kellett kapcsolnom a Kevésbé biztonságos alkalmazások hozzáférését és ezt követően egyből tudtam használni a programban megírt e-mail küldési funkciókat.

6.2. Új Gmail jelszó esetén a rendszerben is frissítést kell végezni

Előfordult, hogy egy jelszó váltást követően rendszer futtatásakor az esemény lefutott, viszont az e-mailek nem kerültek elküldésre, illetve nem is érkeztek e-mailek a megadott címekre. Ezt a rendszer nem jelezte hibaként, ezért nehezen jöttem rá a hiba okára. Mint kiderült a Google fiókom jelszavának változása okozta a problémát, mivel hibás jelszó volt megadva a rendszernek, a bejelentkezés megghiúsult és az e-mailek emiatt nem lettek elküldve. A hiba megtalálást követően a jelszót mindenhol frissítettem és ezt követően a rendszer az elvárásoknak megfelelően működött.

6.3. Adatbázis elérése klónozás esetén

Több alkalommal ütköztem olyan problémába, hogy az alkalmazásba valamit elállítva azt nem tudtam helyre hozni. Ilyen esetekben a verzió követőnek a GitHub-nak köszönhetően könnyedén tudtam helyre állítani a problémámat egy leklónozás segítségével, viszont ilyen esetekben a rendszer az adatbázis használatok hibába ütközött. Mivel a programban megadott elérési útvonal, ami az adatbázishoz elvezet megváltozott a klónozás következtében és így a rendszer vagy hibásan futott le máshelyről nyerve az adatbázisba található adatokat, vagy hibát jelzett, ha már nem volt elérhető az előző verzióban megadott elérési útvonalon az adatbázis.

Ilyen esetben az elérési útvonal kijavításával a rendszerben jelentkező probléma egyből megoldódott.

7. Jövőbeli lehetőségek

7.1. Vonalkód

Ahogy a fentiekben már többször említettem a rendszerben található vonalkód olvasó mostani kidolgozásával egy képfelismeréssel működő folyamat. Egy valós környezetben ez nem lenne teljesen megfelelő, ezért egy lehetséges tovább fejlesztési lehetőség lenne, hogy a mostani kép betöltési módszert leváltani egy valódi vonalkód olvasóval működő beolvasásra. Ehhez természetesen egy vonalkód olvasó beszerzése az elsődleges tennivaló, hogy lehetséges legyen valódi helyzetben kipróbálni a rendszer működését.

A valódi olvasó elkészítése mellett, egy másik lehetőség a vonalkóddal kapcsolatban, hogy a könyv kezelőbe is beépítésre kerüljön. Ezzel megkönnyítve a keresést az adatbázisban ezen ablak használata során is.

A vonalkód olvasó segítségével egyéb lehetőségeket is el lehetne érni. Ilyen például egy könyv felvitelekor, hogy a vonalkód alapján a rendszer indítana egy internetes keresést, ahol a találatok közül kikeresve a könyvet, a hozzá tartozó adatokat beírva a programba a könyvet gyorsabban feltudnánk vinni.

7.2. Törzsvásárlói pont levonás

Az eladáshoz tartozó törzsvásárlói pont levonását is tovább lehetne fejleszteni. Lehetővé lehetne tenni, hogy a rendszer ne maga döntse el, hogy mennyi pontot is vonjon le a végösszezből, hanem a vásárló döntése alapján levonásra kerüljön az összes pont, vagy csak egy része, például, ha vannak előző éves pontok, akkor csak az kerüljön le. Ezt minden esetben a felhasználó a vásárló döntése alapján tenné meg.

7.3. Könyv kedvezmény

Szintén az eladáshoz kapcsolódó funkció a könyv kedvezmény. Ezt olyan módon lehetne javítani, illetve tovább gondolni, hogy a mostani helyzetben csak a végösszeg értéke frissül le és kerül kiírásra. Ezzel ellentétben egy lehetőség lenne az, hogy a kosár táblában is lefrissüljön az adott könyv eladási ára az kedvezmény alkalmazását követően. Emellett

akár egy plusz mező hozzáadásával jelezhetnénk azt is, hogy a mely könyvre adtunk már kedvezményt, az esetleges duplikált kedvezmények elkerülése érdekében.

7.4. Áthelyezés más platformra

A rendszer jelenleg egy asztali Windows alkalmazásként üzemel. Ezt érdemes lehet áthelyezni Android platformra, abból az elgondolásból, hogy sokan jobban preferálják az Androidos készülékeket. Ez lehetővé tenné, hogy a program ne legyen annyira helyhez kötött, például egy leltározás során elgondolkodtató ez.

Az Android mellett az IOS rendszer kiépítése is egy lehetőség lenne. Mivel a mai világban nagyon kettészakad ebből a szempontból is egy rendszer felhasználóinak köre, hogy az az alkalmazás milyen eszközökön használható. Ezzel a két áthelyezéssel sokkal nagyobb kört lehetne lefedni.

7.5. Bejelentkezés

A fejlesztés elején tervbe volt véve egy bejelentkezési ablak. Ezt az ötletet elvetettem, amiatt az ok miatt, hogy a rendszert úgy építettem fel, hogy egy felhasználó által könnyedén kezelhető legyen a program, így feleslegesnek véltem egy bejelentkezést bele rakni az alkalmazásba.

Ennek lehetőségét viszont nem vettem el teljes mértékben. A tervet érdemes lenne elkészíteni, ha több szerepkört szeretnék meghatározni a rendszerben. Példának okáért, ha azt szeretnénk, hogy a kimutatásokat ne minden felhasználó tekinthesse meg, illetve a törlések és módosítások is egy magasabb szintű felhasználó által történhessen csak meg.

7.6. Automatikus e-mail javítása

Az e-mailek között a pontok lejáráásával kapcsolatos levél küldést lehetne tovább fejleszteni olyan módon, hogy meghatározott időszakban egy e-mail küldés után a rendszer lássa, hogy már elküldésre került az e-mail. Pillanatnyilag a rendszer az meghatározott időszakban minden indulásnál legenerálja és elküldi a figyelmeztető e-maileket a szükséges címekre. Ez nagyon zavaró lehet a törzsvásárlók számára, így az előbbieken leírtak alapján egyszer kapnának e-mailt.

7.7. Pont törlés, illetve átírás javítása

A rendszerben két automatikus funkció is van, ami javításra szorul. Ezek a pontok törlése és átírása. Ezek most adott dátumokon belül minden indításkor lefutnak a szükséges adatok ellenőrzése után.

Ennek javítására a következők lehetnének a megoldás. A törzsvásárlók táblába két plusz oszlop megadása segítene a gyorsabb vizsgálatban. Ezek a táblák boolean típust kapnának és az első dátumkor történő törlés, illetve átírás után ezek értéke megváltozna. Ezt követően a rendszernek mind a két esetben csak a hozzájuk tartozó oszlopban kellene megvizsgálni, hogy előzetesen végbe ment-e már a folyamat és ha nem akkor a rendszer futtatná a szükséges kód részletet.

Ezt követően, hogy a következő évben ezek a funkciók újfent hasznosíthatóak legyenek, a meg határozott időszak után a rendszer automatikusan visszaállítaná két oszlop tartalmát, arra az értékre, amit megvizsgálva a rendszer futtatni tudja a funkciót.

7.8. Kívánság lista →Törzsvásárlói kód

A következő pár sorban egy olyan lehetőség kidolgozását ismertetem, ami meggyorsítaná egy törzsvásárló által keresett könyv kívánság listába való felvitelét.

Mint azt a funkciók leírásánál közöltem, a törzsvásárlói kód megadása is lehetséges, de nem kötelező adat. Egy olyan lehetőség kidolgozása lenne a cél ebben az esetben, hogy ha egy törzsvásárló szeretne leadni kívánságot akkor a törzsvásárlói kód megadása is elegendő legyen az ilyen helyzetekben. A rendszer a kód segítségével az adatbázis törzsvásárlók táblájából a szükséges adatokat áttöltené a felvinni kívánt kívánsághoz, ezzel megkönnyítve a felhasználó dolgát, hogy minél kevesebb adatot keljen a felvitelhez megadnia.

7.9. Számlázási lehetőség

A számlázás egy lehetőség azon vásárlók számára, akiknek az általuk történt vásárlást igazolni kell, illetve arról el kell számolniuk. Ez a lehetőség több dolgot is rejt magában. Egy részt magát a számlázás kiépítését. Másrészt a rendszerhez további táblát lehetne hozzá adni, ami tárolná a már használt adatokat, hogy azokat újabb vásárlás esetén ne

kelljen újra felvinni a rendszerbe, hanem a törzsvásárlók táblához hasonlóan az adatokat kinyerhessük belőle.

8. Összefoglalás

Dolgozatom alkalmazásának elkészítése alatt a célom az volt, hogy egy minél áttekinthetőbb, könnyebben használható és működésre képes rendszert készítek el, tapasztalataim és a szakirodalom felhasználásával. Emellett a rendszer készítése alatt fontosnak láttam a program több módon történő tesztelését is. Mivel nincs olyan fejlesztő, aki ne hibázna, így ez egy fontos része volt a fejlesztésnek, hogy állandó teszteléssel minimálisra küszöböljük ki a hibák előfordulását.

Szakdolgozatom írásakor pedig azt vettem figyelembe, hogy minél jobban át tudjam adni, mire is képes az általam elkészített rendszer, illetve, hogy minél tágabb belátást tudjak adni az alkalmazás működésébe.

Az elkészült alkalmazás képes lett a könyvek, törzsvásárlók és kívánságok adatbázisban történő kezelésére, az eladások kezelésére, valamint kisebb kimutatások elvégzésére.

Irodalomjegyzék

- [1] I. Zoltán, Programozás C# nyelven, JEDLIK OKTATÁSI STÚDIÓ, 2005.
- [2] „Antikvárium,” [Online]. Available: <https://lexiq.hu/antikvarium>. [Hozzáférés dátuma: 28 04 2022].
- [3] „Assister,” [Online]. Available: <https://assister.co/>. [Hozzáférés dátuma: 12 04 2022].
- [4] „Beosztásom,” [Online]. Available: <https://beosztasom.hu/>. [Hozzáférés dátuma: 12 04 2022].
- [5] „GitHub, ZXing.NET,” [Online]. Available: <https://github.com/micjahn/ZXing.Net#donate>. [Hozzáférés dátuma: 03 05 2022].
- [6] „C# Corner,” [Online]. Available: <https://www.c-sharpcorner.com/blogs/send-email-using-gmail-smtp>. [Hozzáférés dátuma: 04 05 2022].
- [7] G. Tibor, „adonet.pdf,” [Online]. Available: <https://www.inf.elte.hu/dstore/document/279/adonet.pdf>. [Hozzáférés dátuma: 03 05 2022].
- [8] J. Price, C# adatbázisprogramozás mesteri szinten, Kiskapu KFT, 2004.

Mellékletek

A program elkészítéséhez és az adatbázis feltöltéséhez használatos weboldalak linkjei:

<https://www.regikonyvek.hu/>

<https://www.antikvarium.hu/>

<https://www.libri.hu/>

<https://moly.hu/>

<https://www.iconfinder.com/>

Ábrajegyzék

1. ábra Use Case Diagramm	17
2. ábra Főablak.....	17
3. ábra Eladás ablak	18
4. ábra Törzsvásárlók ablak	26
5. ábra Könyvkezelő ablak.....	29
6. ábra Kívánság lista ablak	33
7. ábra Kimutatók	36
8. ábra Menü sáv	41
9. ábra Az adatbázis felépítése.....	51
10. ábra INSERT parancs	57
11. ábra UPDATE parancs.....	58
12. ábra DELETE parancs	58

Táblázatjegyzék

1. táblázat Könyv keresés	18
2. táblázat Könyv kosárba helyezése	19
3. táblázat Könyv törlése a kosárból	20
4. táblázat Törzsvásárlói pontok jóváírása.....	21
5. táblázat Törzsvásárlói pontok levonása	22
6. táblázat Könyv kedvezmény alkalmazása	22
7. táblázat Végösszeg kedvezmény alkalmazása	23
8. táblázat Könyv eladása	24
9. táblázat Vonalkódos keresés	25
10. táblázat Törzsvásárlói regisztrálás	26
11. táblázat Törzsvásárló megjelenítése	27
12. táblázat Törzsvásárló adatainak módosítása	28
13. táblázat Törzsvásárló törlése.....	28
14. táblázat Új könyv hozzáadása.....	30
15. táblázat Könyv keresés	30
16. táblázat Könyv adatainak módosítása.....	31
17. táblázat Könyv törlése	32
18. táblázat Új kívánság hozzáadása.....	34
19. táblázat Kívánság megjelenítése	34
20. táblázat Kívánsághoz tartozó adatok módosítása	35
21. táblázat Kívánság törlése	36
22. táblázat Teljes kimutatás adott időszakra	37
23. táblázat Bevételek kimutatása adott időszakra	37
24. táblázat Kiadások kimutatása adott időszakra	38