

# **SZAKDOLGOZAT**

Szoftverfejlesztő tanfolyam

Budapest  
2021

RUANDER Oktatási Kft.

**„MAXTER WAREHOUSE MANAGEMENT”  
RAKTÁRKEZELŐ SZOFTVER**

Konzulens:

Készítette:

Budapest  
2021

## **Tartalomjegyzék**

# 1. Bevezetés

## 1.1 Témaválasztás

Ezen szakdolgozat megírásának első lépése a témaválasztás volt, mely egyben az egyik legnehezebbnek bizonyult. A képzés elejétől kezdve bíztatva voltunk a választásra. Sok ötletelés után konzulensemre, egyben tanáromra hallgatva világossá vált a választás. Tanácsát megfogadva, egy hozzá közel álló témát választottam. Közelebb nem is állhatna mivel foglalkozásom központi eleme. Így született meg a döntés, hogy egy raktárkezelő szoftvert készítsek.

Munkám alatt nagyon sok időt töltöttem az általunk használt szoftverrel, és megadatott a szakmai tudás, hogy átlássam működését. A terjedelekre (és időre) való tekintettel ez a szakdolgozat csak a legfőbb funkcióit tartalmazza egy hétköznapokban, multinacionálisan forgalmazó cég által használt raktárkezelő szoftvernek.

## 1.2 Feladat

Egy komplex, OOP irányelveket követő grafikus alkalmazás mely képes ellátni egy raktár kezelésénél felmerülő adatbáziskezelést igénylő feladatokat. Az alkalmazás képes nagy mennyiségű részletes termékadat tárolására, feldolgozni a termékek raktárba való érkezését, tárolni a cég partnereinek adatait, rendeléseket rögzíteni és feldolgozni, továbbá számlázásra is lehetőséget biztosít a Billingo rendszerével, JSON objektumokkal kommunikáló Application Programming Interface-én keresztül, mellyel a program a Nemzeti Adó- és Vámhivatal előírásainak megfelel.

Ezen funkciók segítségével a program alkalmas egy termék teljes útját végig követni az érkezéstől az eladásig, egy kompakt alkalmazásban.

## 2. Felhasználói dokumentáció

### 2.1 Bejelentkező felület

The screenshot shows a login form titled 'Login' with the heading 'Maxter Management'. It contains two input fields: 'Felhasználónév:' and 'Jelszó:', both with placeholder text. Below the inputs are two buttons: 'Kilépés' (Logout) and 'Login' (highlighted with a blue border).

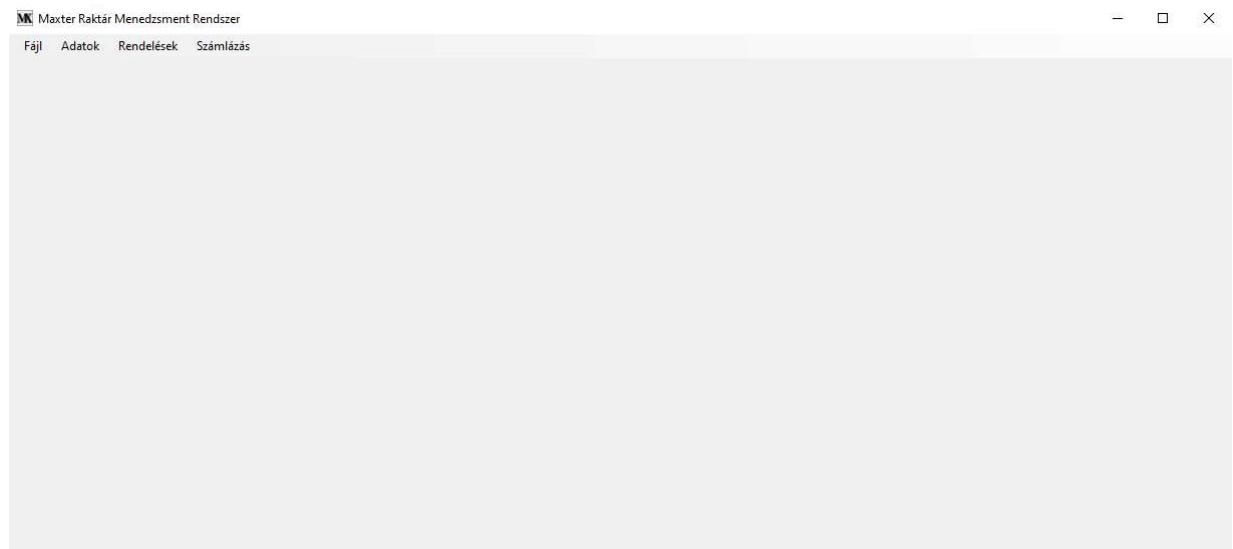
A program elindulása után először a bejelentkező képernyő jelenik meg. A szolgáltatott adatbázis rendelkezik egy alap adminisztrátori hozzáféréssel rendelkező fiókkal:

Felhasználónév: Admin

Jelszó: Admin

Bejelentkezés után erősen javasolt a jelszó megváltoztatása. (2.3.3. pont)

### 2.2 Főképernyő felépítése



A program főképernyőjén található a főmenü menüsor, melyből elérhető a program összes funkciója. A főképernyő felépítése dinamikusan változik a főmenü használata közben.

## **2.2.1 Főmenü**

A főmenü elemei a bejelentkezett felhasználó hozzáférési fokától függnek. Az alap felhasználó teljes hozzáféréssel rendelkezik, mely esetben a választható elemek a következők: Fájl, Adatok, Rendelések, Számlázás.

## **2.3 Fájl menü**

### **2.3.1 Admin beállítások**

Ezen menüponton keresztül lehet új felhasználókat felvinni a rendszerbe, a jelenlegi felhasználók adatait szerkeszteni, vagy a felhasználókat törölni.

### **2.3.2 Billingo API Kapcsolat**

Ezen menüponton keresztül lehet beállítani a Billingo számlázó szolgáltatással való kapcsolatot. A kapcsolat beállítása szükséges, hogy egyenesen a programból lehessen számlázni. A beállításhoz szükséges a Billingo felületén elérhető, API 2.0 publikus és privát kulcs melyet regisztráció és bejelentkezés után a <https://www.billingo.hu/> oldalon lehet elérni.

### **2.3.3 Jelszó változtatása**

Amennyiben az adott felhasználónak hozzáférése van a menüponthoz, lehetősége nyílik saját jelszavának megváltoztatására, mely minimum 6 és legfeljebb 24 karakter lehet a hossza.

### **2.3.4 Nyelv választása**

A program két nyelven használható, magyar és angol. Ezen menü segítségével lehet kiválasztani a kívánt nyelvet.

## 2.3.5 Kilépés

A kilépés menüpont kiválasztásával a program bezárul. Ilyenkor minden nem mentett adat elvész.

## 2.4 Adatok menü

### 2.4.1 Terméktörzs

The screenshot shows the 'Product Catalog' (Terméktörzs) screen of the software. At the top, there is a menu bar with Hungarian labels: Fájl (File), Adatok (Data), Rendelések (Orders), and Számlázás (Invoicing). Below the menu is a toolbar with icons for creating new records, deleting, and saving changes. The main area contains three tables:

- EAN-kódok**: A table showing EAN codes, item numbers, descriptions, quantities, minimum stock levels, and order counts. One row is highlighted in blue.
- Árak**: A table showing prices in HUF for different categories.
- Termék tartalmazó rendelések**: A table showing order details for specific products.

At the bottom of the screen, there are several buttons for managing records and prices, and a 'Bezárás' (Close) button in the bottom right corner.

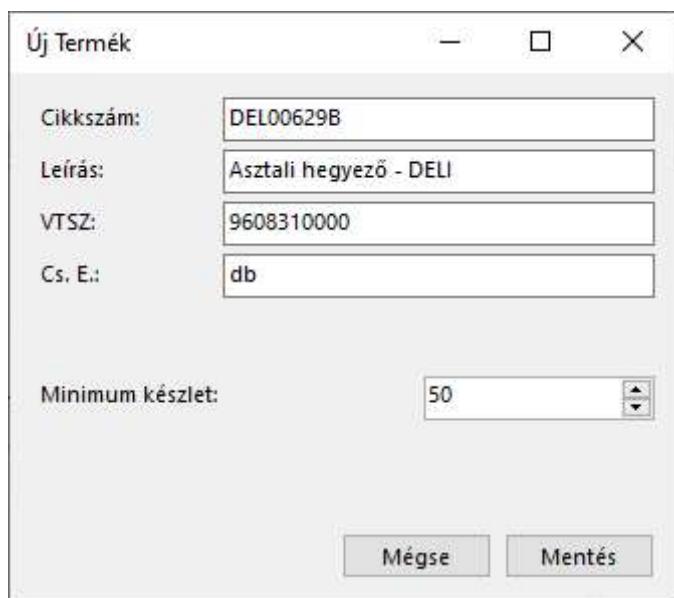
A program legfontosabb komponense a terméktörzs. A felhasználónak lehetősége van termékeket felvenni az adatbázisba, módosítani és törölni azokat. minden terméknek rendelkeznie kell egy egyedi azonosítóval (cikkszám). A termékekhez lehetőség van hozzárendelni a vámtarifaszámukat és megadható egy 16 karakteres leírás/megnevezés, továbbá a termék csomagolási egysége is megjölhető.

Megjelenítéskor csak a termékek listája látható. A bal alul található jelölőnégyzetek segítségével megjeleníthető a termékek vonalkódjai (EAN), árai és a rendelések listája melyeken szerepel a termék.

Ezen felül minden termékhez hozzárendelhető több ár és vonalkód (EAN) is. Az árak egy kategória megjelöléssel rendelkeznek, mely segítségével rendelések létrehozatalánál a partnerek kategóriájuknak megfelelő árak jelennek meg azonnal.

A vonalkódok felvitelénél megadható milyen egységre vonatkozik. Például gyűjtőcsomagolt terméknél megadható a gyűjtőcsomagoláson szereplő vonalkód (24 db) és maga a terméken (1 db) szereplő vonalkód is, így nagyobb mennyiség értékesítése esetén könnyebben nyomon követhetők a termékek.

#### 2.4.1.1 Termék felvitele, módosítása és törlése



Új Termék	
Cikkszám:	DEL00629B
Leírás:	Asztali hegyező - DELI
VTSZ:	9608310000
Cs. E.:	db
Minimum készlet:	50
<b>Mégse</b> <b>Mentés</b>	

A kezelőfelület jobb oldalán található gombokkal kezelhető a termék adatbázis. Új termék felvitekor kötelező megadni egy egyedi azonosítót a terméknek. A többi adat opcionális, később is megadhatók a „Rekord módosítása” gomb használatával. Módosításhoz és törléshez először kikell jelölni a listában egy rekordot.

## 2.4.1.2 EAN felvitele, módosítása és törlése

Lehetőség van egy termékhez tetszőleges számú EAN-t rendelni. Felvitelnél először kikell jelölni a terméket melyhez, hozzákívánja rendelni az EAN-t. Módosításhoz és törléshez először meg kell jeleníteni a termékhez tartozó EAN-okat a jelölőnégyzetek segítségével, majd kiválasztani a módosítani/törölni kívánt rekordot.

## 2.4.1.3 Árak felvitele, módosítása és törlése

Egy termékhez tetszőleges számú kategorizált ár is hozzárendelhető. Először kikell jelölni a terméket. Az árak módosításához és törléséhez először meg kell jeleníteni a termékhez tartozó árakat a jelölőnégyzetek segítségével, majd kiválasztani a módosítani/törölni kívánt rekordot.

## 2.4.2 Partnertörzs

The screenshot shows a software window titled "Maxter Raktár Menedzsment Rendszer". The menu bar includes "Fájl", "Adatok", "Rendelések", and "Számlázás". The main area displays a table of partner information:

Adószám	Partner ID	Név	Számlázási cím	Szállítási cím	Árkategória	E-Mail	Telefonszám	Rendelések száma
12548754	1010	Partner #1	Address #1	Shipping #1	2	email@email.com	+23 45 234 23	1
25468578451	1011	Teszt Cég Kft.	Magyarország 1055 Budapest Válahol utca 8.	Magyarország 1055 Budapest Válahol utca 8.	1	teszt.ceg@mail.hu	+36 70 54 65 875	0
54076542511	1012	Másik Cég Kft.	Magyarország 2200 Pécs József tér 12-16.	Magyarország 2200 Pécs Vámház körút 9.	1	másik@ceg.hu	+36 50 24 57 444	0

Below the table are three buttons: "Partner hozzáadása", "Partner szerkesztése", and "Partner törlése". At the bottom right is a "Bezárás" button.

A partnertörzs menüpont segítségével lehet a cégek partnereinek információit tárolni. Megnyitás után a program egy listában megjeleníti mindegyik partner legfontosabb adatait.

### 2.4.2.1 Partner hozzáadása, módosítása és törlése

A kezelőfelület jobboldalán található gombok segítségével érhető el minden funkció. Egy partner felviteléhez szükséges a partner neve, adószáma és számlázási címe. Ezen felül megadható a partnernek egy szállítási cím (amennyiben különbözik a számlázási címtől, ellenkező esetben a program alapból a számlázási címet veszi szállítási címnek is), telefonszám, e-mail cím és ár kategória.

Partner módosítása esetén először kikell választani a módosítandó partnert. minden adatot meglehet változtatni, kivéve az adószámot. Adószám változás esetén új partnert kell felvinni.

Egy partner törléséhez is először kikell választani a törlendő partnert a listából.

## 2.5 Rendelések

Rendelés ID	Partner ID	Rendelése dátuma	Kérít kiszállítás dátuma	Kiszállítás dátuma	Rendelt termékek	Partner
1	1010	1/14/2020 12:00:00 AM	1/15/2020 12:00:00 AM	1/16/2020 12:00:00 AM	2	Partner #1

A rendelések menüpont segítségével létrehozhatók új rendelések, láthatóak az eddigiek, módosíthatók a még nem szállítottak és törölhetők a nem kívántak. E mellett a készletfeltöltés is ezen a képernyön keresztül történik.

### 2.5.1 Rendelés létrehozása

Cikkszám	Leírás	Mennyiség:	Cs. E.:	Ár
DEL00629B	Asztali hegyező	8	db	1800.0000
DEL00629	Asztali hegyező	10	db	1800.0000

Rendelés létrehozásakor megjelenik egy ablak, melyen megkell adni a rendelést leadó partner azonosítóját és a rendelés teljesítésének határidejét. Ezután megjelenik a rendelés felület.

### **2.5.1.1 Termék hozzáadása a rendeléshez**

Termékeket az első oszlop kitöltésével lehet hozzáadni a cikkszámok segítségével. Miután beütöttünk egy cikkszámot, az „Enter” billentyű megnyomásával a program kigyűjt az összes, a termékhez kapcsolódó adatot. Az ár, amely megjelenik, a partnerhez beállított ár kategória alapján kerül kiválasztásra.

A mennyiség az adott sorban való mennyiség cella kijelölése után tetszőlegesen változtatható. Amennyiben a kívánt érték meghaladja a raktárkészletet, a program erről tájékoztat és korrigálja az értéket.

### **2.5.1.2 Termék törlése a rendelésből**

A rendelésből terméket törölni az „F3” billentyű leütésével lehet.

### **2.5.1.3 Rendelés befejezése**

A rendelés kitöltését a „Kész” gomb megnyomásával lehet befejezni. Ha esetleg változna a rendelés összetétele, később még módosítható.

## **2.5.2 Rendelés szerkesztése**

Miután kijelöltük a szerkeszteni kívánt rendelést, a „Rendelés szerkesztése” gomb megnyomásával megnyílik a rendelés felület, betöltve a rendelés tételeit. Ezután a szerkesztés ugyanúgy zajlik mintha egy új rendelést kezelnénk.

## 2.5.3 Rendelés törlése

Miután kijelöltük a törölni kívánt rendelést, a „Rendelés törlése” gomb megnyomásával törölhetjük azt. Fontos: Csak szállítás előtt álló rendeléseket lehet törölni. Miután meg lett jelölve egy rendelés szállítottként, a rendelés már nem változtatható meg semmilyen módon!

## 2.5.4 Megjelölés szállítottként

Ha egy rendelés elkészült és lelett számlázva, a rendelés elküldésekor/átadásakor használja a „Megjelölés szállítottként” gombot. Ezzel a rendelés véglegesítve lesz, és kap egy dátum jelzést.

## 2.6 Számlázás

Számla ID	Létrehozás dátuma	Partner
744050633	2020-10-23	Partner #1
100191269	2020-10-23	Teszt Elek Kft.
1381321221	2020-09-26	Teszt Elek Kft.
1394004565	2020-09-26	Teszt Elek Kft.
404751010	2020-09-20	Teszt Elek Kft.

Amennyiben a Billingo API Kapcsolat beállításra került, a programban a használati kényelem érdekében lehetőség van számlákat kiállítani a Billingo rendszerén keresztül, a böngésző használata nélkül. A programon belül gyorsan és könnyedén lehet a rendelésekkel

készpénzes vagy átutalásos számlákat kiállítani. A számlázás menüponton látható az elkészített számlák listája, és létrehozhatunk új számlákat.

### **2.6.1 Számla készítése**

A „Számla készítése” gomb megnyomása után megjelenik egy ablak, melyen megkell adni a számlázandó rendelés azonosítóját. Ezután megjelenik a Számla ablak melyen áttekinthetők a partner számlázási adatai és a rendelés értéke. Lehetőség van kiválasztani, hogy a számla készpénzes vagy átutalásos legyen-e. Átutalás esetén megadható a fizetési határidő. Ezen felül még lehet megjegyzést csatolni a számlához. A kész számla az <https://app.billingo.hu/> oldalról letölthető/nyomtatható bejelentkezés után.

## **2.6.2 Számlák letöltése**

A „Számlák letöltése” gombbal lehet a Billingo rendszerből a számlákat letölteni.

## **2.6.3 Számlák megjelenítése**

A „Számlák megjelenítése” gombbal lehet a számlák listáját megjeleníteni a programban.

A megjelenítéshez először le kell a számlákat tölteni.

## **3. Fejlesztői dokumentáció**

### **3.1 Program modellezés**

#### **3.1.1 Modellezési nyelv**

A program modellezése az UML (Universal Modelling Language –Univerzális Modellező Nyelv) nyelven készült. Egész pontosan a 2017-ben adoptált 2.5.1-es változatán.

Az UML, mint ahogy magában az elnevezésében is látszik tökéletesen alkalmas volt a program minden funkciójának, folyamatának megtervezésére és ábrázolására.

#### **3.1.2 Modellező szoftver**

A modellezéshez az oktatás folyamán biztosított Enterprise Architect-et alkalmaztam. Az Enterprise Architect által biztosított lehetőségek messze túlmutatnak a szakdolgozat elkészítéséhez szükséges funkciókon, minden feladat megoldására alkalmat adott.

## **3.2 A szoftver gyakorlati implementálása**

### **3.2.1 Fejlesztői környezet**

Az alkalmazás a tanfolyam alatt használt Visual Studio 2019 Integrált Fejlesztői Környezetben készült, Windows 10 operációs rendszerre tervezve.

### **3.2.2 Programnyelv**

Az alkalmazás a tanfolyam során használt C# programnyelven íródott. Személyes, az oktatáson kívül szerzett tapasztalataim alapján egyszerűbbnek bizonyult a programot C# nyelven megírni, mint például C nyelven. Ez köszönhető a C# tisztán objektumorientáltságának és automatikus memória kezelésének.

A program grafikus felhasználói felülete a Windows Forms segítségével készült, mely bár kissé limitált, egy raktárkezelő program megjelenítéséhez tökéletesen elegendő, és otthonosan hat a felhasználó számára a klasszikus, letisztult és professzionális hatású látványvilág.

### 3.3 Adatbázis

A programban az adatbázis kezelés az Entity Framework használatával valósult meg, Database First modellezéssel. Ezzel a megoldással lehetőség nyílt a LINQ to Entities használatára. Előnye a gyorsabb implementálás volt, viszont az Entity Framework használatának elsajátítása miatt igazi időnyereség nem keletkezett.

Kettő adatbázist használ az alkalmazás. Az egyik adatbázis tartalmazza a raktárkezelés szerves részeit képező adatokat, míg a második különálló adatbázis a felhasználók személyes adatait tárolja.

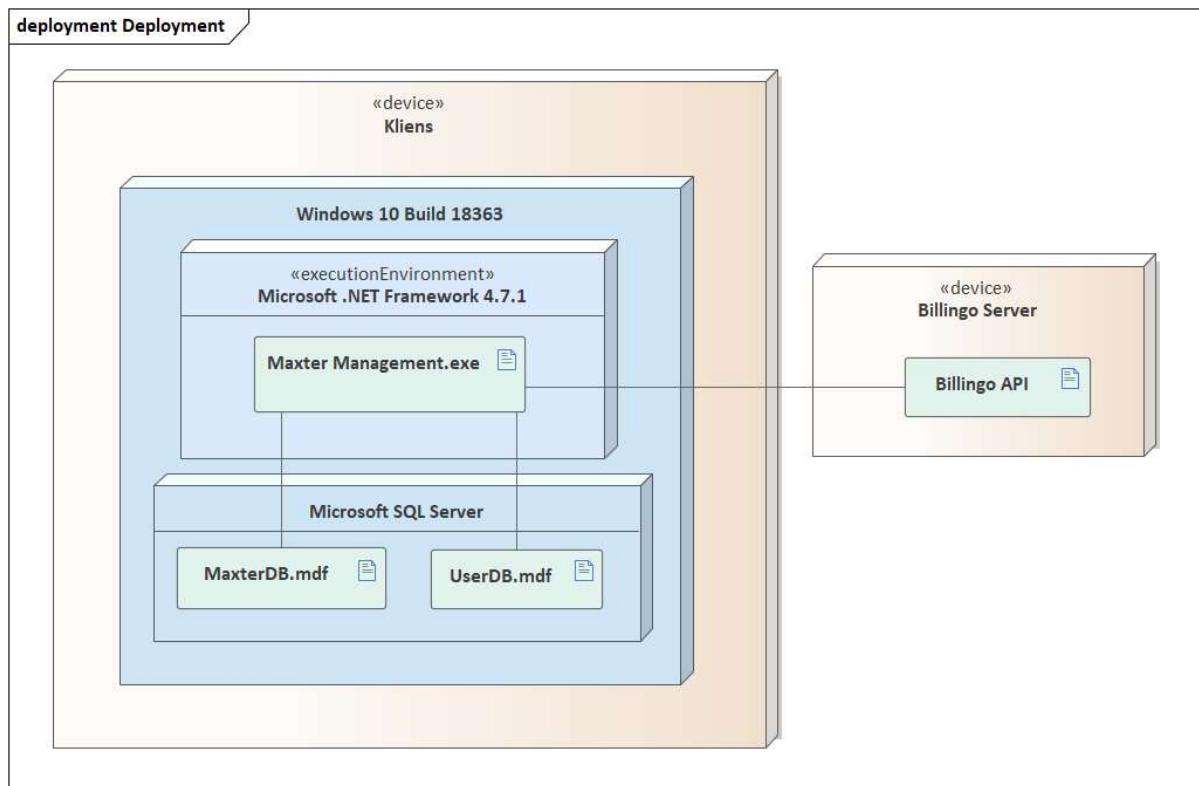
A szakdolgozat elkészítéséhez szerettem volna nem csak a tanfolyamon tanul megoldásokat is alkalmazni, így lehetőségem nyílt valósághűbb tapasztalatok szerzésére. Az oktatás során megtanultuk, a szoftverfejlesztés szerves része ismereteink folyamatos bővítése.

### 3.4 UML Diagrammok

A záródolgozatban több, különböző típusú UML diagram is helyt kapott, melyek szemléltetik az alkalmazás felépítését és működését:

- Deployment diagram
- Class diagram
- Use Case diagram
- Sequence diagram
- Data Model diagram

### 3.4.1 Telepítési diagram (Deployment diagram)

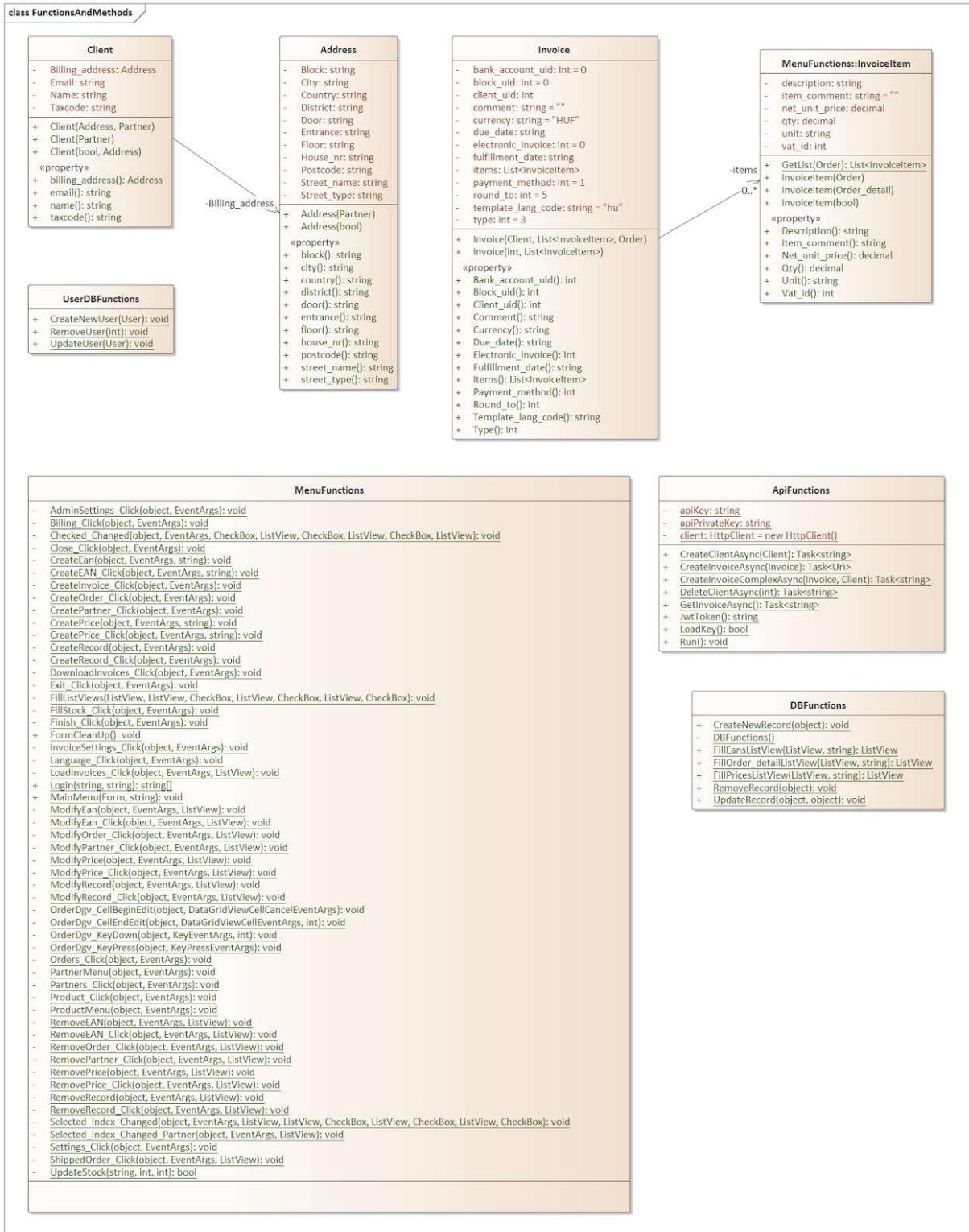


A Deployment diagramon a futtatási környezet, hardver igény, továbbá a kliens és szerver kapcsolata látható. A program lokális adatbázisokkal működik, nem igényel hozzá külön szervert. Futtatáshoz szükséges a .NET Framework keretrendszer 4.7.1-es, vagy újabb verziója.

A számlázáshoz a Billingo API-n keresztül történik a kommunikáció a Billingo szerverével.

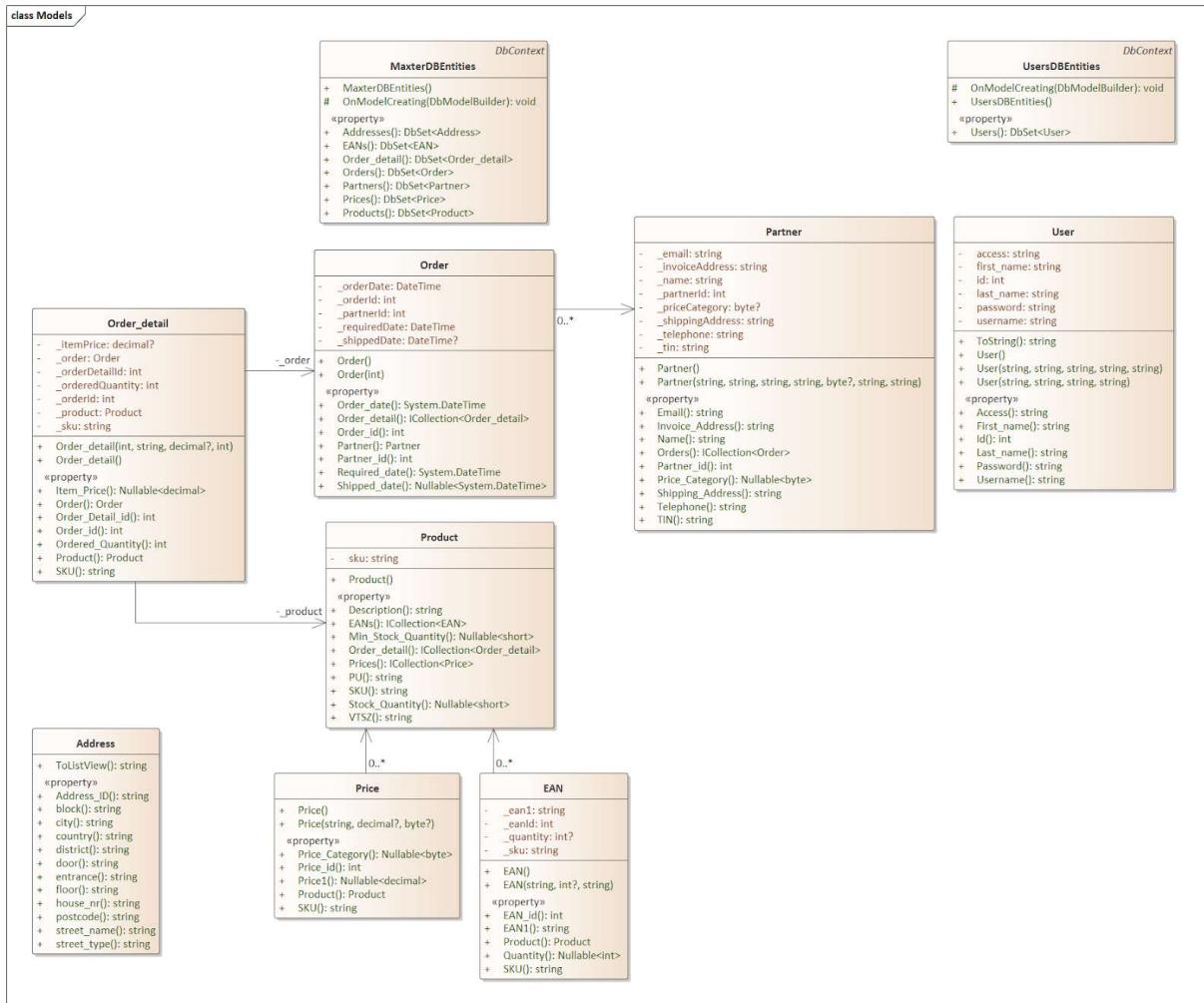
### 3.4.2 Főbb funkciók osztálydiagramja

#### (Class diagram)



### 3.4.3 Entity Framework modellek osztálydiagramja

#### (Class diagram)



Fejlesztés során a már említett Entity Framework Database First megoldását alkalmaztam. A tanfolyamon tanult „valósvilág” modell” alapján indultam el. Figyelembe véve az objektumok valódi tulajdonságait, leegyszerűsítve őket a feladat megoldására elegendőkre.

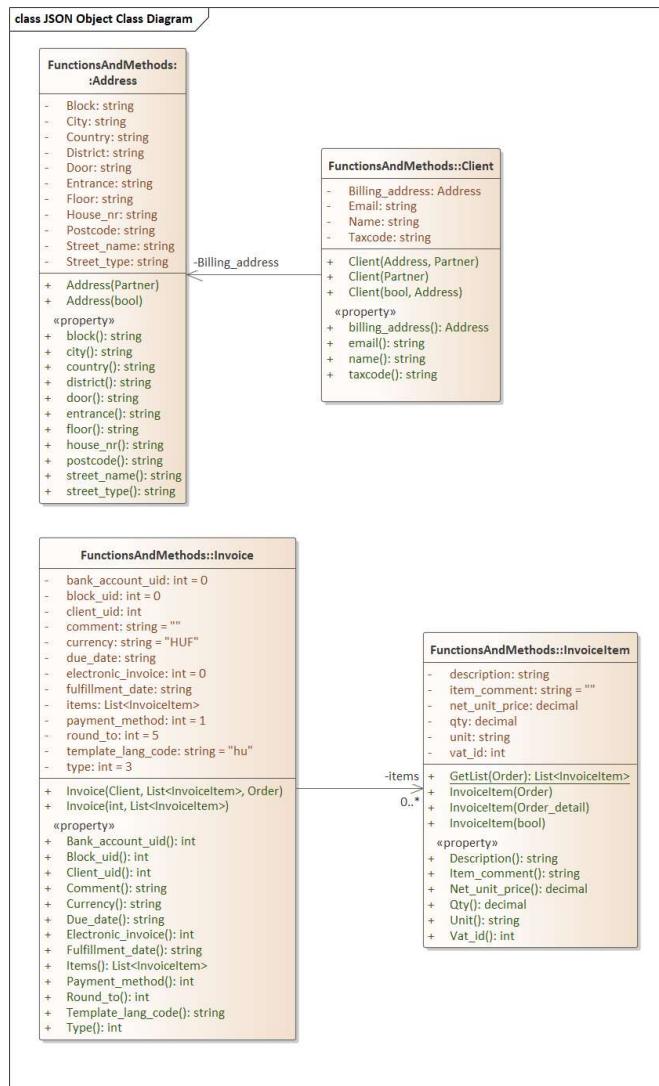
A megmaradt tulajdonoságokból épült fel az adatbázis, melyből az Entity Framework képes volt generálni egy modellt, létrehozva automatikusan az objektumokat. Ezzel a megoldással nincs szükség SQL lekérdezések alkalmazására, egyszerű objektumokként kezelhetők az adatbázis elemei.

Természetesen a generált osztályok korántsem elegendők, csak az objektumok vázát adja meg.

Egyetlen hátránya ennek a megközelítésnek a szerkesztés nehézsége. Amennyiben valamilyen okból kifolyólag, esetleges tervezési hiba miatt változtatni kell az adatbázison az implementálás során, a teljes modellt újra kell generálni mellyel minden változtatás elveszik.

### 3.4.4 JSON Objektumok osztálydiagramja

#### (Class diagram)



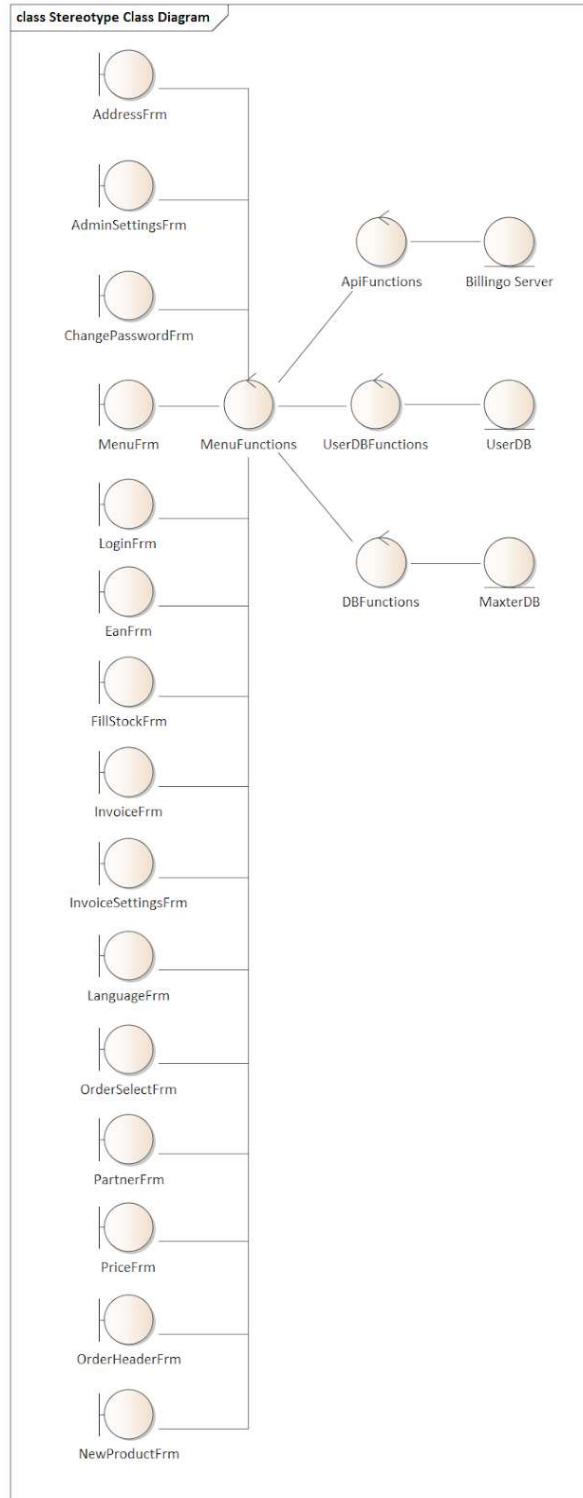
A záródolgozat készítése során megismerkedtem a JSON objektumokkal és az aszinkron funkciókkal is. A Billingo API-val JSON objektumokkal kell kommunikálni, nagyon specifikus megkötésekkel. A diagrammon láthatóak az osztályok melyek erre a célra lettek létrehozva.

A Billingo rendszerén jelen időben csak egy, az Ő szerverükön található adatbázisban létező partner felé lehet számlát kiállítani. Annak érdekében, hogy a helyi adatbázis elszeparált legyen, ne függjön egy külső tényezőtől, egy kreatív megoldást kellett találni.

A program először küld egy JSON objektumot, mely a helyi adatbázisban található partner. A szerver válaszában megtalálható az újonnan létrehozott partner azonosítója. Ezután kerül küldésre a számla objektum, tartalmazva a kapott azonosítót. Miután ez is sikerült, az azonosító alapján törlésre kerül a Billingo adatbázisából a partner. Mindez aszinkron történik, hogy ne okozzon fennakadást az alkalmazás működésében.

Elsőre furcsának tűnhet ez a megoldás. A Billingo rendszerén partnerfelvitelként nincs lehetőség megadni az azonosítót, amivel könnyen összelehetne fűzni a két adatbázist. Az egyetlen ténylegesen egyedi azonosító, amivel a partnerek rendelkeznek az az adószámuk, annak kinyerése esetén viszont minden egyes számlázáskor a teljes partnerlistát lekell kérni a szerverről. Egy sok partnerrel rendelkező nagyobb cég esetében ez nem költséghatékony.

### 3.4.5 Osztályok és Form-ök sztereotípusos osztálydiagramja (Class diagram)

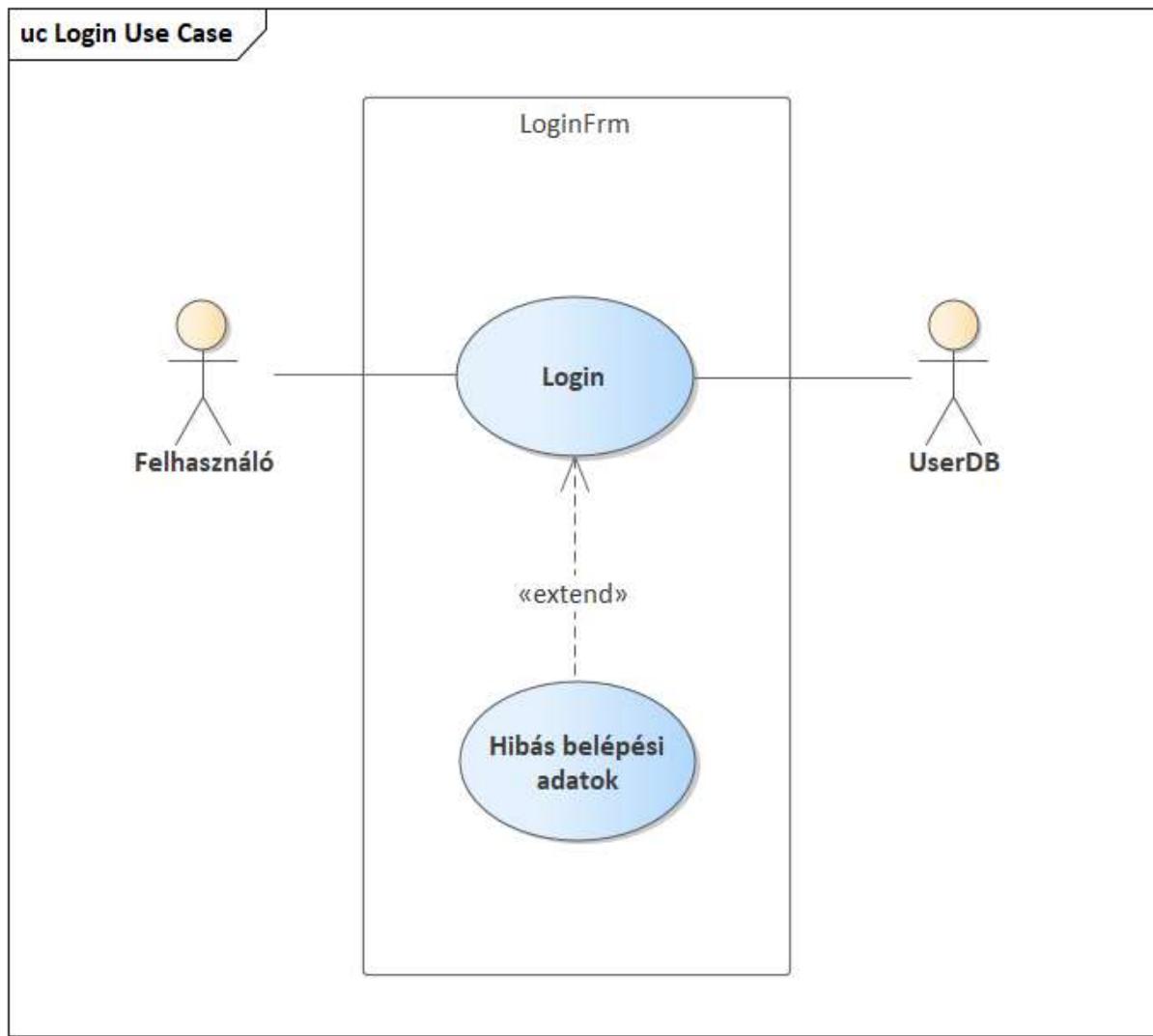


A sztereotípus osztálydiagram jól szemlélteti az alkalmazás felépítését. Letisztultan láthatjuk a kapcsolatot a kezelőfelület, a funkciókat és metódusokat tartalmazó osztályok és az adatbázisok között.

A legtöbb funkció dialógus ablakokon keresztül használható, ez az oka a nagy Form mennyiségnek. A felhasználói felület és a „business logic” a tanultaknak megfelelően elszeparáltak egymástól.

### 3.4.6 Bejelentkezés használati eset diagramja

(Use Case diagram)

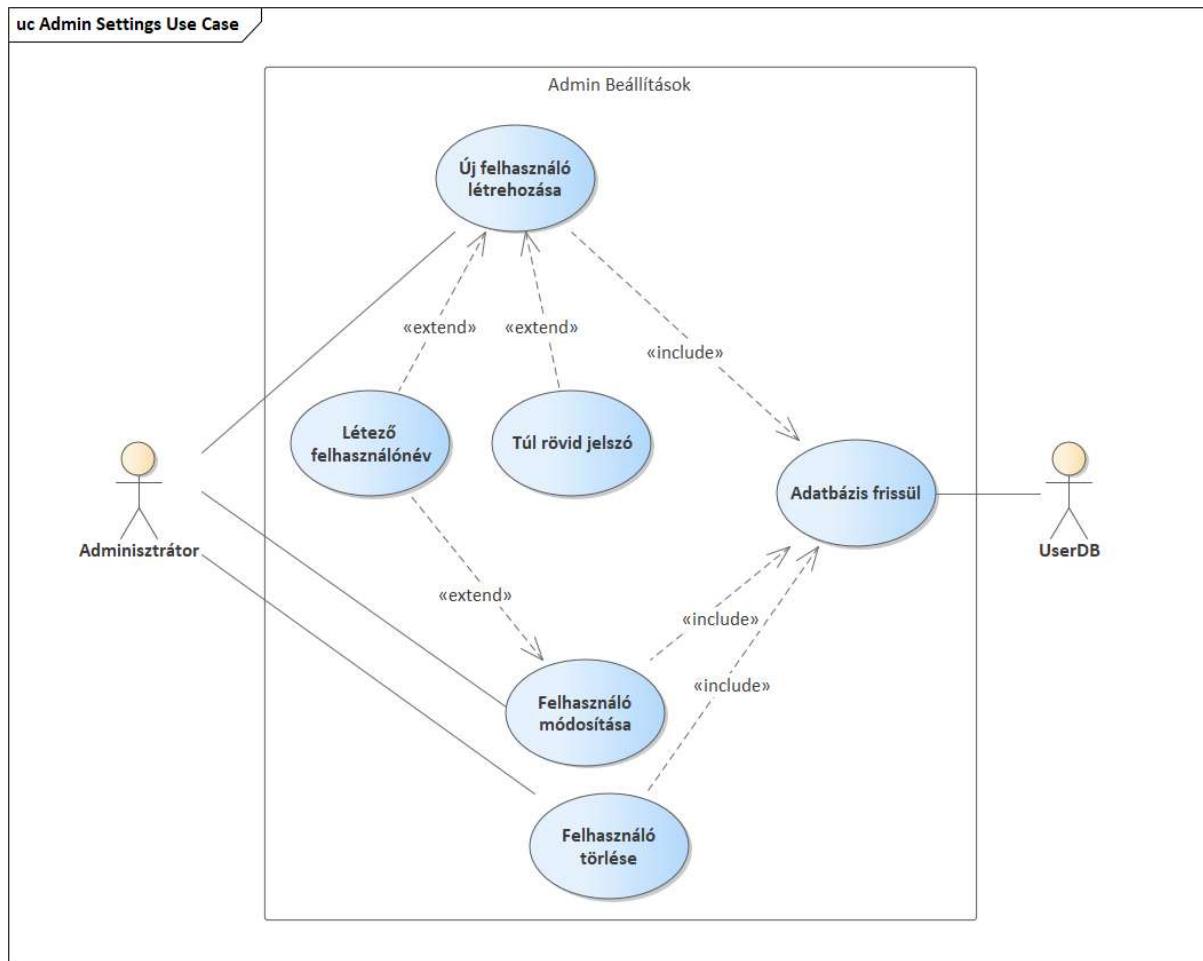


Use Case diagrammok segítségével átláthatóan bemutatható az alkalmazás gyakorlati működése.

A bizniszszférában alkalmazott szoftvereknél biztonsági okokból nincs lehetőség adminisztrátori hozzáférés nélkül új felhasználót létrehozni. Jelen ábrán látható, hogy a programba csak bejelentkezni lehet indulásnál.

### 3.4.7 Admin beállítások használati eset diagramja

#### (Use Case diagram)



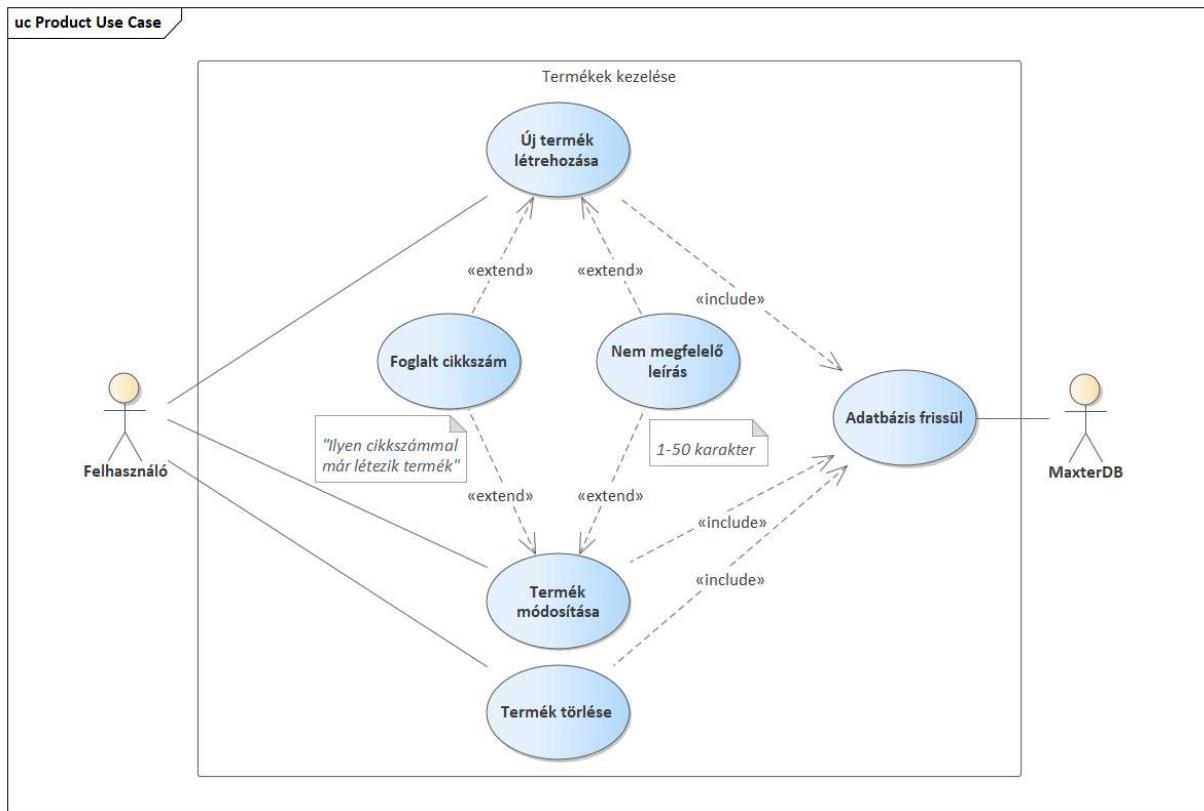
Mint ahogy a diagrammon látható, az Admin beállításokat csak adminisztrátori hozzáféréssel rendelkező személy érheti el.

Továbbá látszik, hogy hiányzik a jelszó módosítás lehetősége. Ez biztonsági megfontolásból fakad. Az adminisztrátor be tud állítani egy alap jelszót a felhasználók létrehozásakor és amennyiben biztonságosnak látja, engedélyezheti a felhasználónak a saját jelszójának megváltoztatását. Így az adminisztrátor nem tudja az egyes felhasználók jelszavát.

Amennyiben valaki kizárta magát a rendszerből, új felhasználót kell számára létrehozni. A rendszer működéséből adódóan nem történik adatvesztés, csak egy kellemetlen beszélgetés az adminisztrátorral.

### 3.4.8 Termékek használati eset diagramja

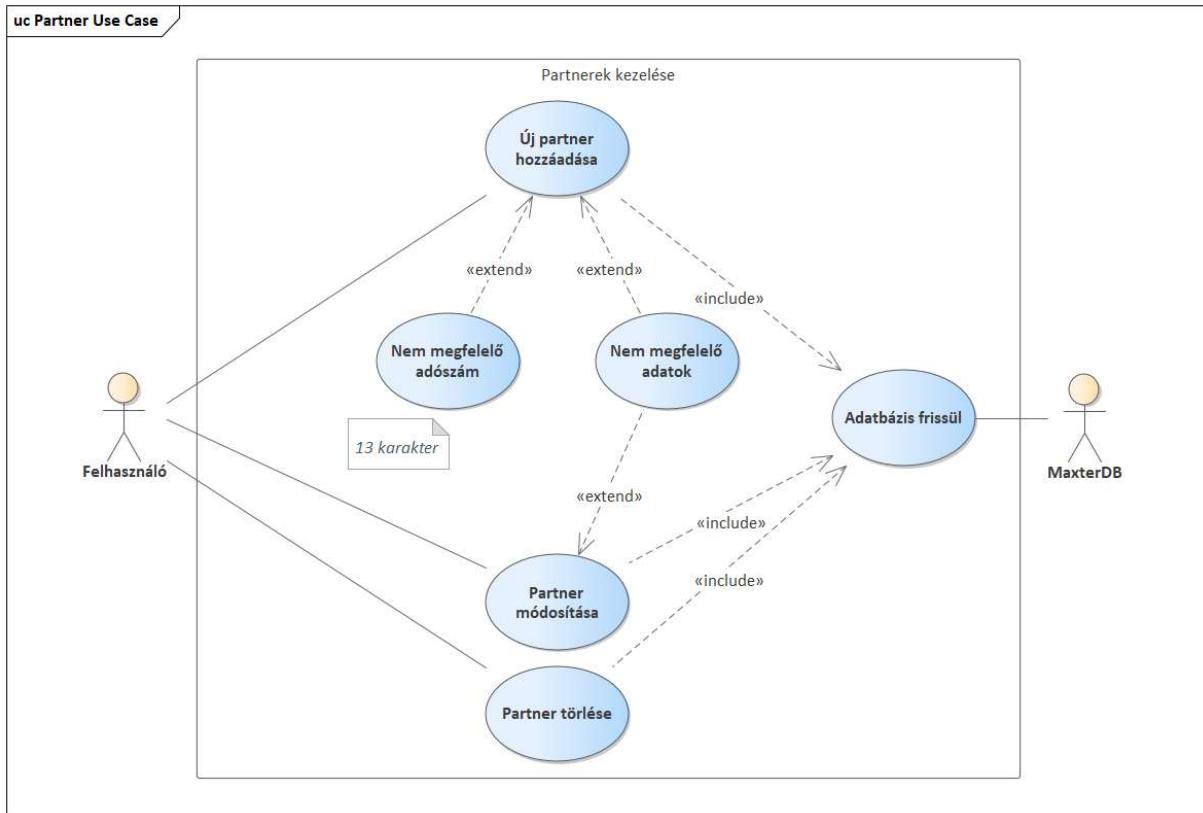
#### (Use Case diagram)



A diagrammon jól látható a termékekkel elvárt műveletek lefutása. Ezek ugyebár a rekord felvitel, rekord módosítás és rekord törlése.

### 3.4.9 Partnerek használati eset diagramja

#### (Use Case diagram)

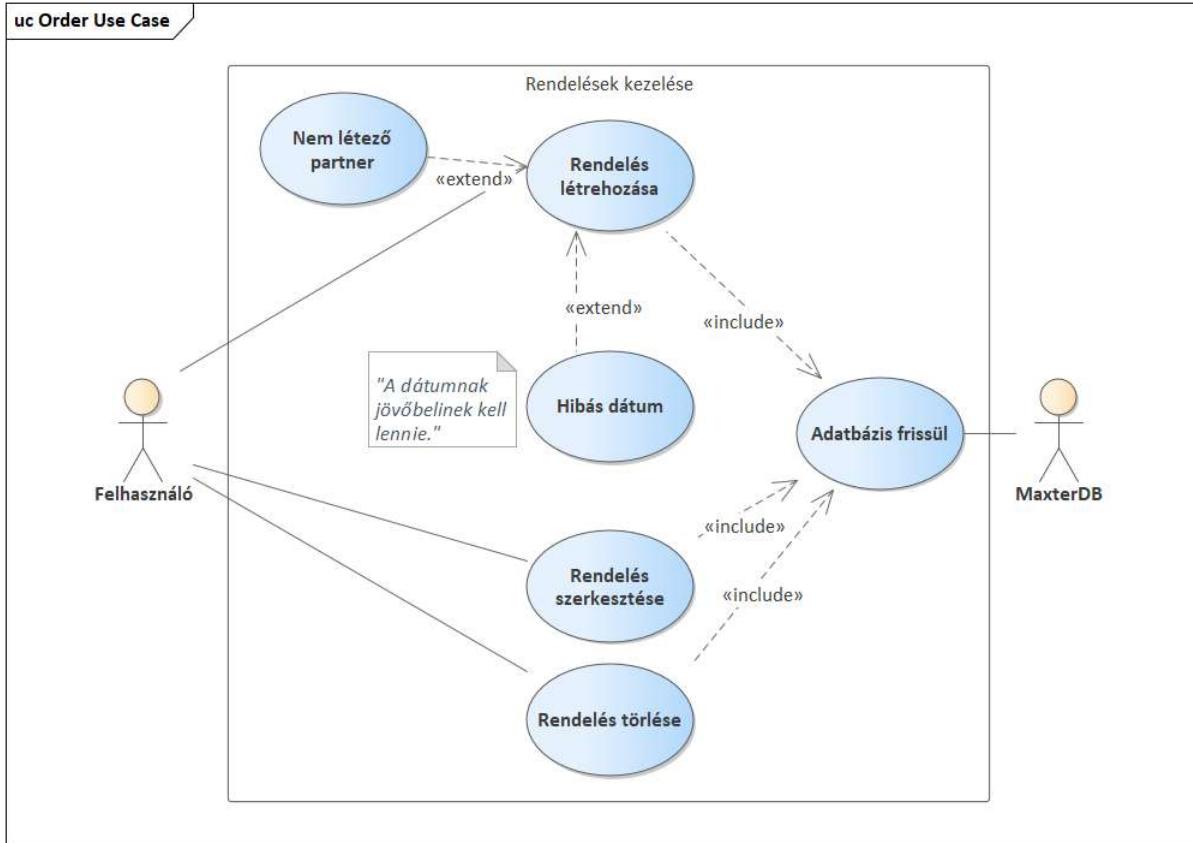


A diagrammon látszik az adószám változtatásának lehetőségének hiánya. Cégek esetén az adószámban csak a helykód változhat. Az egyszerű kezelhetőség érdekében, amennyiben változik a cég adószáma (mely esetben a cím is változik), új partnereként kell újra felvinni a céget.

Ezen kívül látszónak a megszokott új rekord, rekord módosítás és rekord törlése folyamatok működése.

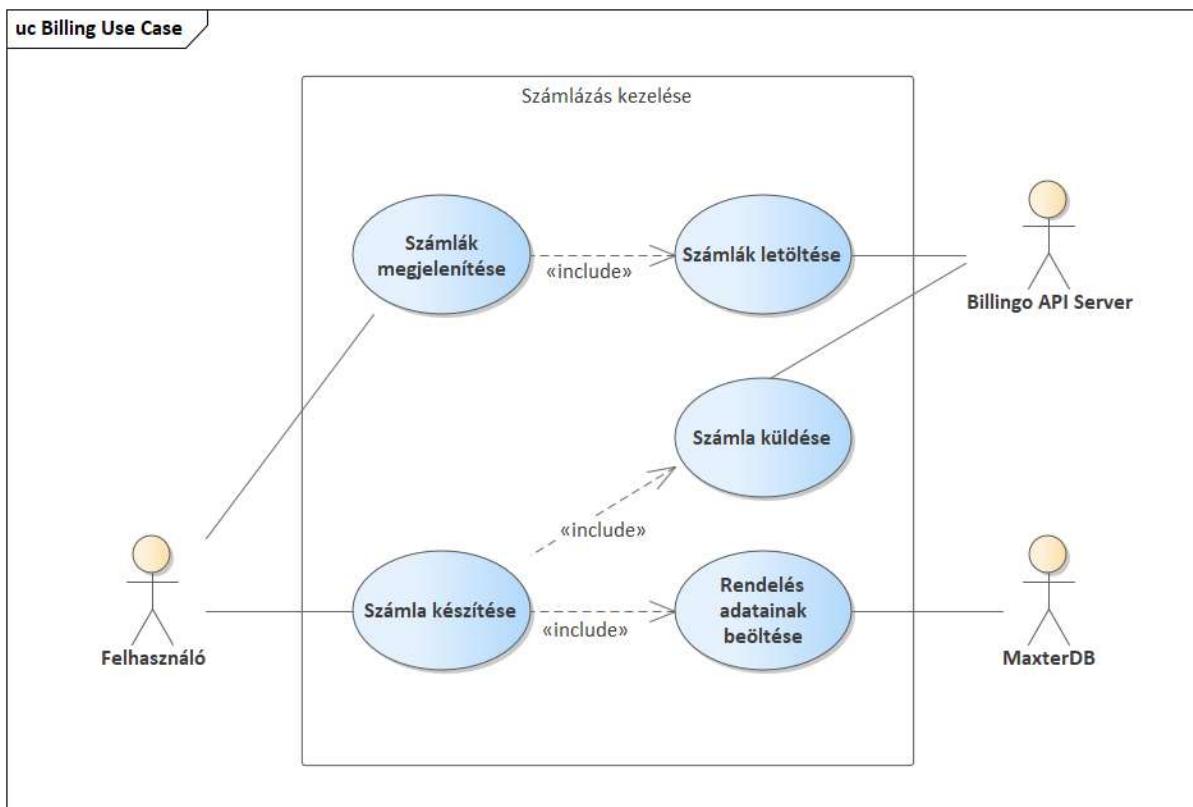
### 3.4.10 Rendelések használati eset diagramja

#### (Use Case diagram)



### 3.4.11 Számlázás használati eset diagramja

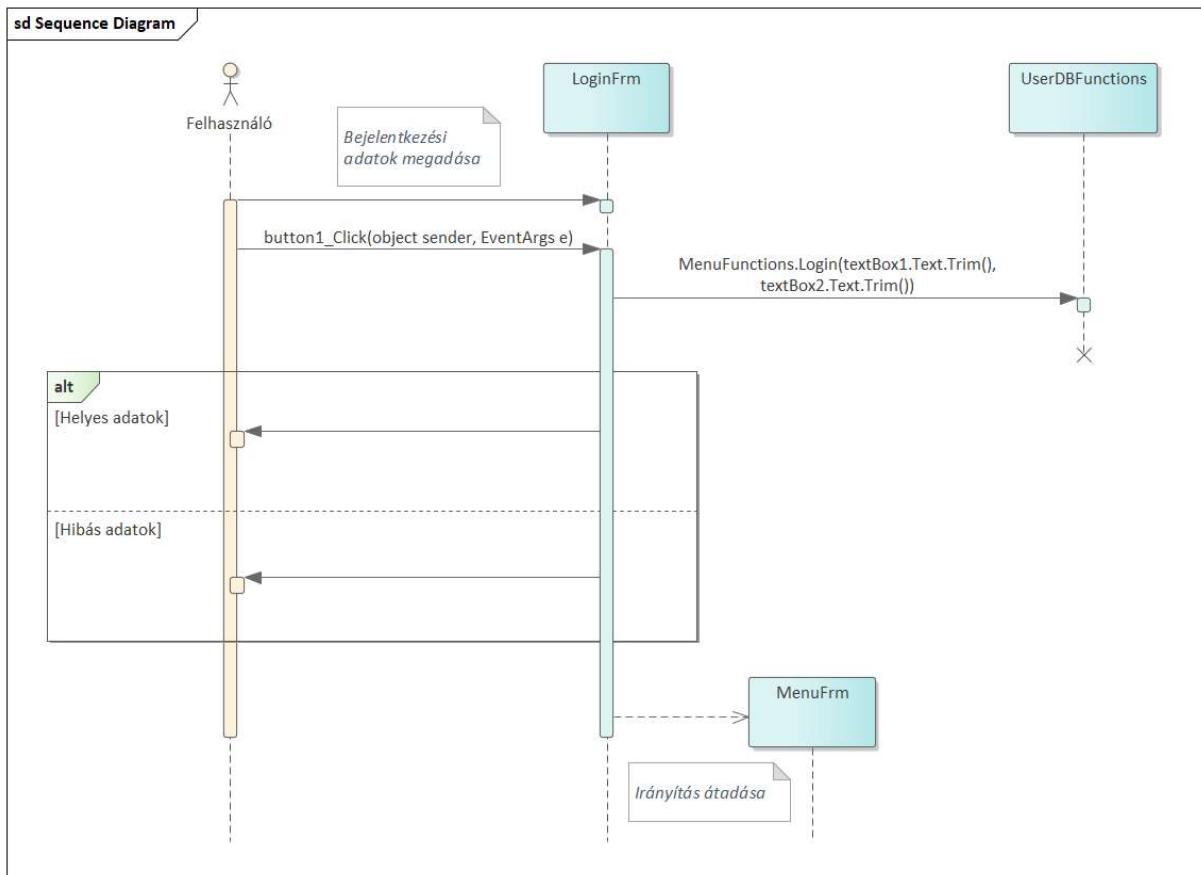
(Use Case diagram)



A fent látható use case diagram egyedisége az Actor-ok számának növekedése. Mivel a számlázás egy külső szolgáltatás segítségével működik, láthatjuk a helyi adatbázis (MaxterDB) mellett a külső szolgáltatás (Billingo API) szerverét is.

### 3.4.12 Bejelentkezés szekvencia diagramja

#### (Sequence diagram)

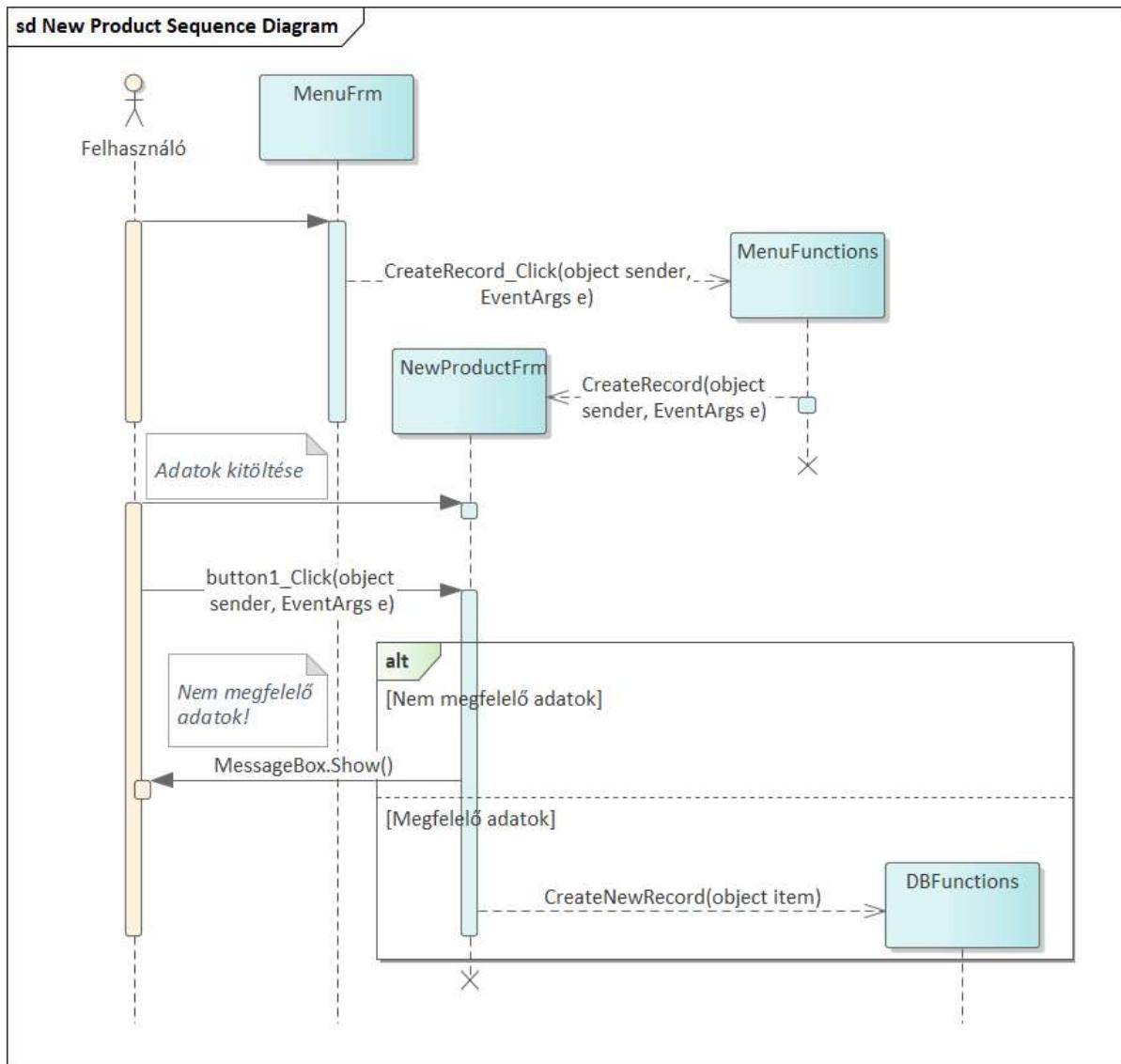


A képen látható szekvencia diagram a bejelentkezés folyamatát mutatja be. Ezen keresztül rálátást szerzünk az objektumok közti kapcsolatokra a futási idő függvényében.

A diagramon láthatjuk milyen tevékenységet folytat a bejelentkezési felületen (LoginFrm), és megmutatja, hogyan reagál a rendszer. Tisztán láthatjuk a kezelőfelület és a „business logic” elszeparálódását a LoginFrm és UserDBFunctions kapcsolatán keresztül.

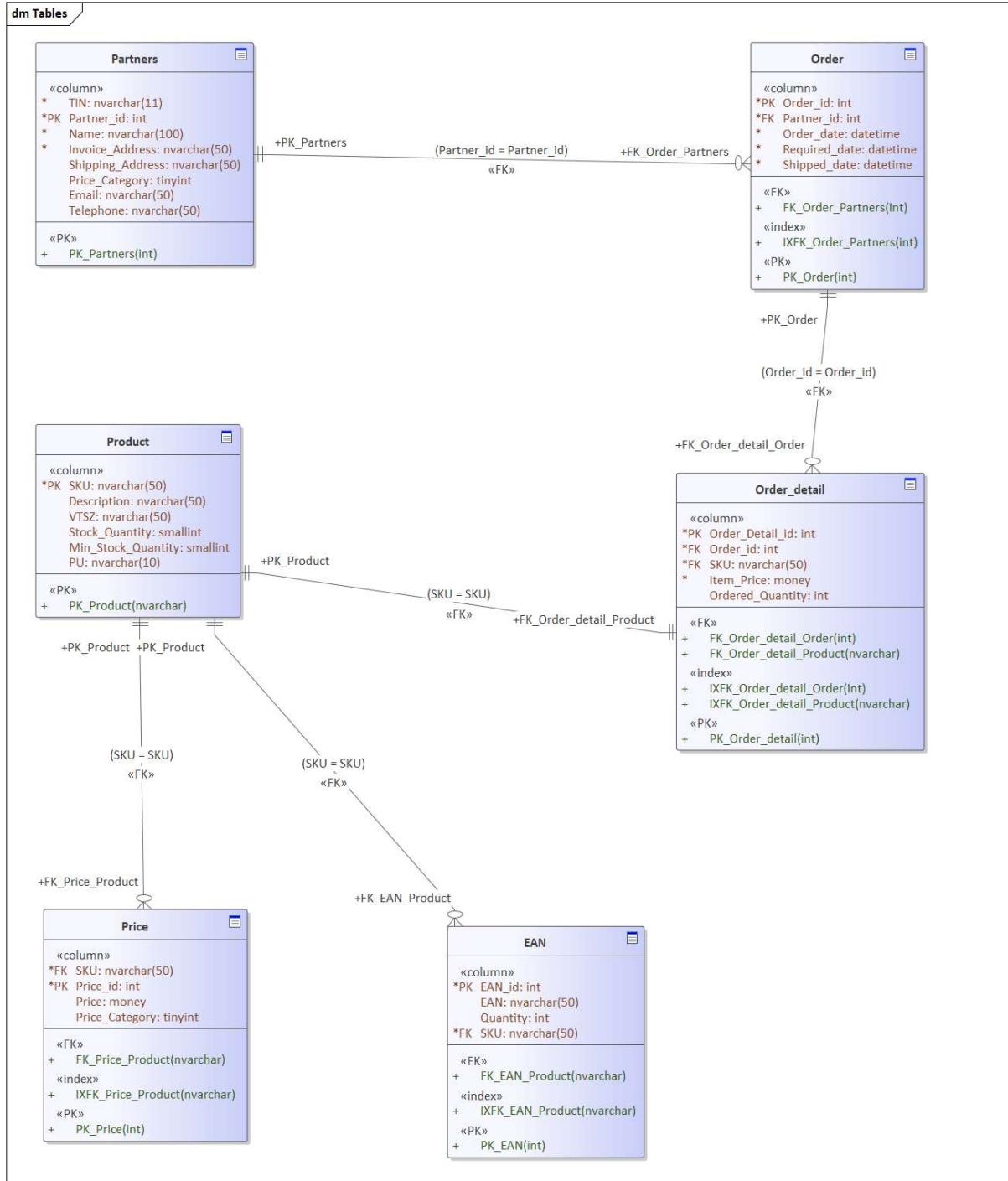
### 3.4.13 Új termék felvitelének szekvencia diagramja

(Sequence diagram)



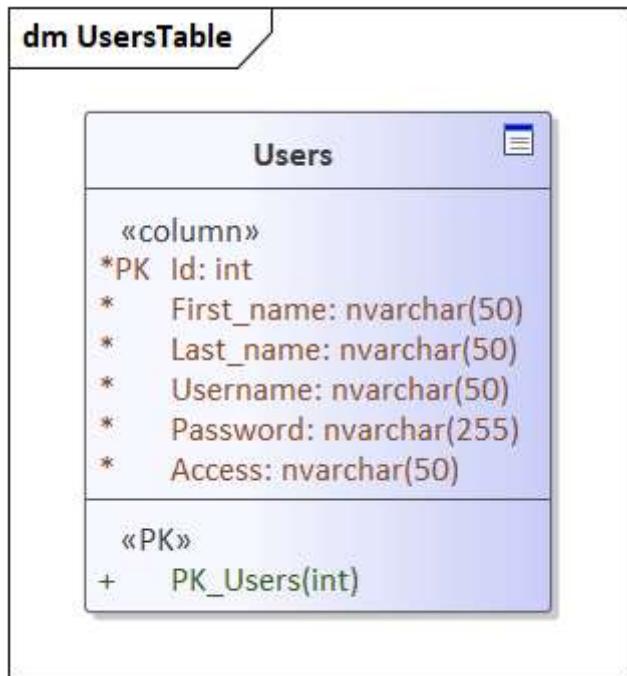
### 3.4.14 Adatmodell diagram a fő adatbázisról

#### (Data Model diagram)



### 3.4.15 Adatmodell diagram a felhasználók adatbázisról

## (Data Model diagram)



A diagrammon látható a felhasználók adatbázisának felépítése. Indoklásom a két adatbázis elválasztására az esetleges migráció fennállása. Lehetőség nyílik az adatbázisok egymástól független kezelésére, és jobban szabályozható az emberek hozzáférése a kényes adatokhoz. Továbbá az esetleges hibák, adatvesztések is elszeparálódnak egymástól.

A biztonság érdekében a jelszó SHA256 kódolással kerül tárolásra, továbbá kódolás előtt kap egy toldalékot, hogy adatlopás esetén ne lehessen kódolt táblázatokkal összevetni az adatokat.

Az Access sor tárolja a hozzáférési jogosultságot. A program karakterenként dolgozza fel és amennyiben a megfelelő karakterek szerepelnek a megfelelő indexen, elérhetővé válnak a felhasználó számára az alkalmazásba épített funkciók.