Veszprémi Szakképzési Centrum Ipari Szakgimnáziuma

Szakképzés neve: Szoftverfejlesztő

OKJ száma: 54 213 05SZAKDOLGOZAT

Szakdolgozat címe

Vastag Atila Hanich Kristóf

Témavezető 13. C

Veszprém, 2020

Tartalom

[A program általános specifikációja 3](#_Toc34906813)

[Rendszerkövetelmények 3](#_Toc34906814)

[Hardverkövetelmények 3](#_Toc34906815)

[Szoftverkövetelmények 3](#_Toc34906816)

[A program telepítése 4](#_Toc34906817)

[Fejlesztői Dokumentáció 5](#_Toc34906818)

[Témaválasztás indoklás 5](#_Toc34906819)

[Az alkalmazott fejlesztői eszközök 5](#_Toc34906820)

[Adatbázis modell 7](#_Toc34906821)

[Biztonsági kérdések 7](#_Toc34906822)

# A program általános specifikációja

A program egy lézergravírozással foglalkozó cég rendeléseinek rögzítésére szolgál.  
A program felhasználói felületének kifejlesztésekor felhasználóbarát navigációra törekedtem.

Az első lapon rendelések leadására illetve listázására van lehetőség.  
A rendelés leadás egy ügyfélnévből, a vásárolt árucikkből, a vásárolt termék darabszámából, illetve az elkészítés határidejéből áll.  
A második lapon új terméket vihetünk fel az adatbázisba. A feltöltéshez szükséges adatok közé tartozik a termék neve, rövid leírás, ár, jelenleg hány darab van belőle, illetve képet is csatolnunk kell az adott termékről.  
A harmadik lap vásárlók rögzítésére szolgál legfontosabb adataikkal együtt: név, lakcím, telefonszám.   
A negyedik lap a Raktárkészlet módosítására szolgál.  
Az ötödik lapon a raktár található, ahol meg tudjuk tekinteni, minden termékünk összes adatát.

# Rendszerkövetelmények

A program futtatásához szükséges komponensek a következők:

## Hardverkövetelmények

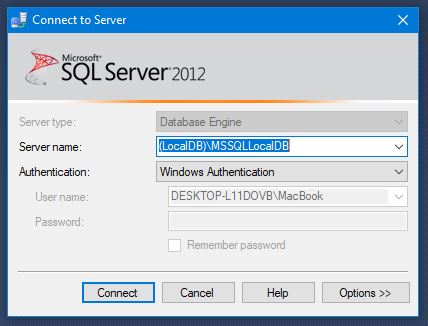
* Minimális:
  + Processzor: 2 x 1.6GHz
  + Memória: 4GB
  + Merevlemez: 2GB szabad hely
* Ajánlott:
  + Processzor: 4 x 1.6GHz
  + Memória: 8GB
  + Merevlemez: 5GB szabad hely

## Szoftverkövetelmények

* Program:
  + .Net keretrendszer 4.7.2-os verzió
  + Windows 7
* Adatbázis:
  + Microsoft® SQL Server® 2012 Express
  + Windows 7

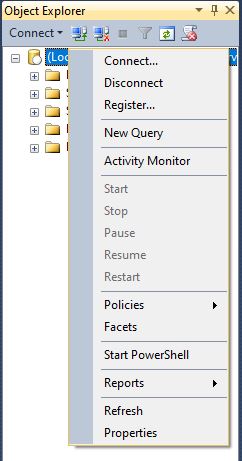
# A program telepítése

Csatlakozás az adatbázishoz



1. Nyissa meg az SSMS-t. A Connect to Server (Kapcsolódás a kiszolgálóhoz) párbeszédpanel jelenik meg.
2. Adja meg a következő információkat:

| **Beállítás** | **Ajánlott érték** | **Leírás** |
| --- | --- | --- |
| **Server type** | Database Engine | Csatlakozás típusa |
| **Server name** | (LocalDB)\MSSQLLocalDB | A teljes kiszolgálónév |
| **Authentication** | Windows hitelesítés | Hitelesítés típusa |

1. Kattintson a Connect gombra. Az Object Explorer ablak nyílik meg.
2. Az adatbázis felvételéhez kattintson jobb egérgombbal a (LocalDB)\MSSQLLocalDB elemre
3. Válassza a New Query opciót.
4. Illessze be ide a program könyvtárban található Adatbázis.sql fájlt.
5. Az adatbázis objektumainak megtekintéséhez bontsa ki az Databases csomópontot, majd bontsa ki a gravirozasDB elemet.
6. 8. Az adatbázis elkészítése sikeres.

# Fejlesztői Dokumentáció

## Témaválasztás indoklás

A témát azért választottam, mivel szabadidőmben lézergravírozással foglalkozom. Ahogy egyre jobban növekedtek a rendeléseink számai, egyre nehezebb volt papíron, illetve telefonon, jegyzetben vezetni a rendeléseket. Szükségünk volt egy programra, amiben rögzíteni és rendszerezni tudjuk a rendeléseket, így nem fordulhat elő az, hogy egy rendelés elkeveredik és csúszunk a határidővel. Később mivel a programban rögzítjük vásárlóinkat és elérhetőségeiket, tudunk nyereményjátékot bevezetni, illetve kedvezményeket is tudunk biztosítani törzsvásárlóinknak.

## Az alkalmazott fejlesztői eszközök

* Microsoft Visual Studio 2019
  + System.Data.SqlClient v4.8.1
  + MaterialDesignColors v1.2.2
  + MaterialDesignThemes v3.0.1
  + NETStandard.Library v2.0.3

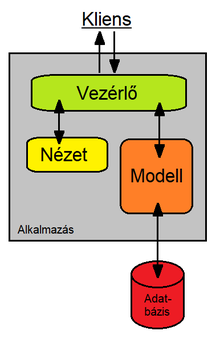
A Microsoft által készített .NET keretrendszer (a .NET Framework) gyors alkalmazásfejlesztést, platformfüggetlenséget és hálózati átlátszóságot támogató szoftverfejlesztői platform. A keretrendszert a korábbi platform, a COM leváltására szánták. Eredetileg a .NET kifejezés nemcsak fejlesztői környezetet jelentett, hanem fejlesztőeszközök, szoftverek, sőt hardvereszközök összességét is. Az évek során a kép kitisztult, így mostanra a .NET alatt a keretrendszert értjük. A .NET Framework eszköztára a szoftverfejlesztés szinte minden aspektusát (kliens-, illetve szerveroldali megoldások, adatbázisok kezelése, játékfejlesztés stb.) lefedi.

A CLI-t úgy tervezték, hogy bármilyen objektumorientált programozási nyelvet támogasson, megosztva egy közös objektum modellt és egy nagy, közös osztálykönyvtárat. A .NET jelenleg több mint 40 programozási nyelvet támogat, melyek többsége ingyenes (a kereskedők fejlesztői környezeteket árulnak). Sok nyelvet jelentősen hangoltak, hogy illeszkedjen a .NET keretrendszerbe. A gyártók ezt kihasználva gyakran egyéb nyelvi funkciókat is módosítottak.

Beépített nyelvek

* C#, a C++-hoz és a Sun Java nyelvéhez hasonló objektumorientált nyelv
* JScript .NET, Microsoft JScript nyelvének fordított verziója
* J#, a Java és a J++ programozók átmeneti nyelve a .NET keretrendszer felé
* Managed C++, a C++ egy változata a .NET platformra
* Visual Basic .NET, a klasszikus Visual Basic egy továbbfejlesztett, objektumorientált, többszálas verziója

A Visual Studio a Microsoft több programozási nyelvet tartalmazó fejlesztőkörnyezete, amely az évek során egyre több új programnyelvvel bővült. Jelenleg a F#, C++, C# és Visual Basic programozási nyelveket, valamint az XML-t támogatja. A csomag része még a MASM (Microsoft Macro Assembler) is, ami részleges assembly támogatást biztosít.

A modell-nézet-vezérlő (MNV) (angolul model-view-controller) a szoftvertervezésben használatos szerkezeti minta. Összetett, sok adatot a felhasználó elé táró számítógépes alkalmazásokban gyakori fejlesztői kívánalom az adathoz (modell) és a felhasználói felülethez (nézet) tartozó dolgok szétválasztása, hogy a felhasználói felület ne befolyásolja az adatkezelést, és az adatok átszervezhetők legyenek a felhasználói felület változtatása nélkül. A modell-nézet-vezérlő ezt úgy éri el, hogy elkülöníti az adatok elérését és az üzleti logikát az adatok megjelenítésétől és a felhasználói interakciótól egy közbülső összetevő, a vezérlő bevezetésével.

Gyakori egy alkalmazás több rétegre való felbontása: megjelenítés (felhasználói felület), tartománylogika és adatelérés. Az MNV-ben a megjelenítés tovább bomlik nézetre és vezérlőre. Az MNV sokkal inkább meghatározza egy alkalmazás szerkezetét, mint az egy programtervezési mintára jellemző. Modell Az alkalmazás által kezelt információk tartomány-specifikus ábrázolása. A tartománylogika jelentést ad a puszta adatnak (pl. kiszámolja, hogy a mai nap a felhasználó születésnapja-e, vagy az összeget, adókat és szállítási költségeket egy vásárlói kosár elemeihez). Sok alkalmazás használ állandó tároló eljárásokat (mint mondjuk egy adatbázis) adatok tárolásához. Az MNV nem említi külön az adatelérési réteget, mert ezt beleérti a modellbe. Nézet Megjeleníti a modellt egy megfelelő alakban, mely alkalmas a felhasználói interakcióra, jellemzően egy felhasználói felületi elem képében. Különböző célokra különböző nézetek létezhetnek ugyanahhoz a modellhez. Vezérlő Az eseményeket, jellemzően felhasználói műveleteket dolgozza fel és válaszol rájuk, illetve a modellben történő változásokat is kiválthat. Az MNV gyakran látható webalkalmazásokban, ahol a nézet az aktuális HTML oldal, a vezérlő pedig a kód, ami összegyűjti a dinamikus adatokat és létrehozza a HTML-ben a tartalmat. Végül a modellt a tartalom képviseli, ami általában adatbázisban vagy XML állományokban van tárolva. Habár az MNV-nek sok értelmezése létezik, a vezérlés menete általánosságban a következőképp működik:

* A felhasználó valamilyen hatást gyakorol a felhasználói felületre (pl. megnyom egy gombot).
* A vezérlő átveszi a bejövő eseményt a felhasználói felülettől, gyakran egy bejegyzett eseménykezelő vagy visszahívás útján.
* A vezérlő kapcsolatot teremt a modellel, esetleg frissíti azt a felhasználó tevékenységének megfelelő módon (pl. a vezérlő frissíti a felhasználó kosarát). Az összetett vezérlőket gyakran alakítják ki az utasítás mintának megfelelően, a műveletek egységbezárásáért és a bővítés egyszerűsítéséért.
* A nézet (közvetve) a modell alapján megfelelő felhasználói felületet hoz létre (pl. a nézet hozza létre a kosár tartalmát felsoroló képernyőt). A nézet a modellből nyeri az adatait. A modellnek nincs közvetlen tudomása a nézetről.
* A felhasználói felület újabb eseményre vár, mely az elejéről kezdi a kört.

A modell és a nézet kettéválasztásával az MNV csökkenti a szerkezeti bonyolultságot, és megnöveli a rugalmasságot és a felhasználhatóságot.

* Microsoft SQL Server Management Studio 2012
* dbForge Studio for MySQL

## Adatbázis modell

## Biztonsági kérdések

A program háromrétegű architektúrára épül, ennek köszönhetően biztonságosan kommunikál a felhasználói felület egy szerveren keresztül az adatbázissal.

A háromrétegű architektúra egy szoftvertervezési minta, a szoftverarchitektúrán felül. John J. Donovantól származik. A modell legnagyobb előnye, hogy lehetővé teszi az egyes rétegek egymástól függetlenül történő fejlesztését, sőt, akár teljes cseréjét is, lépést tartva így a folyamatosan változó követelményekkel és az egyre újabb technológiákkal. Ez biztonságosan megtehető, mert egy réteg módosítása nincs hatással a többi réteg működésére. Egymástól független modulokként tartalmazza a felhasználói felületet, az üzleti logikát és az adatbázist a szükséges hozzáférési műveletekkel. Például ha a megjelenítésért felelős szerverre új operációs rendszer kerül, akkor elég csak a megjelenítést vezérlő modulokat frissíteni az alkalmazásunkban, az új körülményeknek megfelelően. Ennek a fejlesztési technikának köszönhetően később mobilalkalmazás is készíthető a már meglévő adatbázishoz és backendhez.