**DEBRECENI SZC BEREGSZÁSZI PÁL SZAKGIMNÁZIUMA**

**ÉS SZAKKÖZÉPISKOLÁJA**



4032 Debrecen, Jerikó u. 17.

OM azonosító: 203033

*:52/ 503-150 Fax: 52/314-204*

*http://www.dszcberegszaszi.hu*

*E-mail:*

*titkarsag@dszcberegszaszi.hu*

**Szakképesítés megnevezése: Szoftverfejlesztő  
OKJ száma: 54 213 05**

**ZÁRÓDOLGOZAT**

**BioQuiz kvíz alkalmazás fejlesztése**

**Készítette**: **Konzulens:**  
MADARÁSZ ANNA Boros Sándor

**Debrecen, 2020**

Tartalomjegyzék

[**1.** **Bevezetés** 2](#_Toc39263984)

[**2.** **Felhasznált technológiák bemutatása** 3](#_Toc39263985)

[2.1 C# 3](#_Toc39263986)

[2.2 MySQL /phpMyAdmin 3](#_Toc39263987)

[**3.** **Rendszerterv** 4](#_Toc39263988)

[3.1 Az adatbázis 4](#_Toc39263989)

[3.2 Program 6](#_Toc39263990)

[**4.** **Összefoglalás** 21](#_Toc39263991)

[**5.** **Irodalomjegyzék** 22](#_Toc39263992)

# **Bevezetés**

Egy esetleges online-oktatás keretében történő számonkérésnél hasznos lehetnek, olyan alkalmazások, melyek nem online weboldalakon elérhetőek, hanem asztali formában.

Záródolgozatomban bemutatom az alapkoncepciót, az elgondolást és végül a megvalósítását az általam készített programnak. Az elején igyekszek bemutatni, az elmúlt években tanult és elsajátított technológiákat elméleti szinten, hogy egy kis rálátást adjak az olvasónak. Aztán magára a munkafolyamatba engedek betekintést, lépésről lépésre, egészen a megvalósult programig. Mindeközben pedig részletesen bemutatva az alkalmazott fejlesztéseket és a technológiák megjelenési formáit a dolgozatomban. A kész programról, illetve a munkafolyamat egyes részleteiről készült fotók is segítik a megértést. Legvégül pedig egy rövid összegzésben, értékelem az elkészült programot, és megosztom az ezzel kapcsolatos gondolataimat, illetve véleményemet.

Záródolgozatomban egy, a tanulmányaim során, a mindennapokon is használható alkalmazást alkottam meg. A felelet-választós tesztek nem állnak messze tőlem. Volt alkalmam már az egyetemi tanulmányai során is ilyen tesztet felhasználóként kitölteni. Jelenlegi programozási tanulmányaim során pedig el tudtam sajátítani azt a tudást, ami ahhoz kell, hogy egy általam készített alkalmazást létre tudjak hozni, olyan formában, ami felhasználóbarát, szórakoztató, játékos, mégis hasznos.

A fentiekből kiderülhet, hogy az alkalmazás, amit készítettem egy több kategóriás kvíz, mely a jelenlegi egyetemi tanulmányaimhoz kapcsolódik. A felhasználó csak egy regisztráció, így egy profil létrehozása, után tud belépni, így van csak esélye a kvíz kitöltésére, ezáltal pedig pontok szerzésére. Az egyes kvízeken elért pontokat egy táblázatban lehet nyomon követni a felhasználó profilján, illetve, ott látható, hogy összesen hány pontot gyűjtött is eddig össze. Az egyes kategóriák, a biológiának több alterületére fókuszál. Egy kategóriában random kérdéseket generál a program, amit előzetesen egy adatbázisban lett rögzítve.

Azért választottama C# programkörnyezetet, mert az elmúlt 2-3 évben, tanulmányaim során ezt használtuk a legtöbbször, illetve ez a mindennapokban használt legfőbb programozási nyelv, amiről úgy gondoltam, hogy szeretnék egy záródolgozat keretében is jobban belemerülni a témába.

Az elkészült program igényelhet egyéb továbbfejlesztést a továbbiakban, illetve némi „csinosítást” de jelenlegi állapotában is hasznos lehet a felhasználó számára. Az elkészítése során megszerzett és elsajátított tudást a jövőben is fogom tudni hasznosítani, akár leendő munkám során.

# **Felhasznált technológiák bemutatása**

## C#

A C++ egyik utódjának tekinthető programozási nyelv a C#, mely 2002 óta érhető el a felhasználóknak. Egyben rendelkezik a C, C++ hatékonyságával, viszonylag egyszerűnek is mondható, illetve elég jól teljesít is gyorsaság terén, ami a Visual Basic egyik fontos jellemzője is. A C# egy objektumorientált programnyelv, mely felhasználóbarát, alkalmazások készítésére alkalmas, emellett általános felhasználású nyelv és típusbiztos. A C# bázisnyelve az új .NET keretrendszernek. Amit még a C# programnyelv jellemzői közé tartozik, hogy Neumann-elvű, professzionális, akár rendszerprogram tervezésére is alkalmas nyelv. Eredetileg platformfüggetlenként jött létre.

A Microsoft .NET Framework csak a Microsoft Windows operációs rendszereknek érhető el. [[1](#_Irodalomjegyzék)] [[2](#_Irodalomjegyzék)]

## Microsoft Visual Studio

A legelterjedtem IDE, vagyis Integrált Fejlesztői Környezet (Integradted Development Environment) a .NET programozáshoz. A Visual Studio rendelkezik az ún. IntelliSense rendszerrel, amely felajánlja automatikusan az elkezdett metódusok/ /osztályok/ változók stb. nevének folytatását. [[2](#_Irodalomjegyzék)]

## MySQL

Az SQL egy a relációs adatbázisok karbantartására, illetve lekérdezések végrehajtásához használható nyelv. A MYSQL egy ingyenesen elérhető és használható adatbázis rendszer, ennek fő feladatköre, a különböző dinamikus oldalak háttér adatbázisának biztosítása.

Legegyszerűbb a XAMPP programcsomag részeként lehet elérni, Windows platformú rendszerek esetében. Ez tartalmazza még az APACHE webszervert, ami alkalmas PHP futtatására is, ha dinamikus weboldalak létrehozására van szükség.

A MYSQL-t a <http://localhost/phpmyadmin> tudjuk szerkeszteni grafikusan, mivel ez egy PHP alapú admin felület, mely beépített szerkesztőfelület. [[3](#_Irodalomjegyzék)]

# **Rendszerterv**

## Az adatbázis

A záródolgozatomhoz, egy C# nyelven íródott, adatbázis alapú, asztali kvíz alkalmazást fejlesztettem. A munkafolyamatot a képzés alatt megtanított és a fentebb leírt technológiákkal felhasználásával készült.

A kész program 3 részből állt össze egy teljes, működő egységgé. Először is a kvíz kérdéseit hoztam létre egy Excel fájlban, melyet .*ods* kiterjesztéssel mentettem, hogy aztán importáljam a *http://localhost/phpmyadmin* oldalon.

4 tábla készült:

* User
* Kérdés
* Válasz
* Kategóriák.

Az Excel fájlon belül a *User* munkalap gyűjti a felhasználóneveket, jelszavakat, illetve a kvízek során összegyűjtött pontszámokat és tárol egyéb adatokat, amiket akár statisztika készítéséhez is fel lehet használni. Az egyes felhasználók rendelkeznek egyedi *id*-val is.

A *Kérdés* munkalap tartalmazza az egyes kategóriák *id-*jához tartozó kérdéseket. A *Válasz* munkalap, az egyes kérdésekhez tartozó, 3 lehetséges választ tartalmazzák. Mindegyik válasz rendelkezik egy egyedi *id-*val, illetve egy-egy hamis, vagy igaz jelzővel, attól függően, hogy a kérdéshez, melyik a helyes válasz. A *Kategória* munkalap tartalmazza, az egyedi *id-*val rendelkező kategóriákat.

A második lépés az volt, hogy a XAMPP telepítését követően a *http://localhost/phpmyadmin* honlapon, egy az alkalmazáshoz illő adatbázist hoztam létre *bioquiz* néven, ahova az .ods kiterjesztésű Excel fájlt importáltam megfelelő beállításokkal*.*

Ahogy az Excel fájlban, itt is 4 táblát hoztam létre:

* *kategoriak*
* *kerdes*
* *user*
* *valasz*

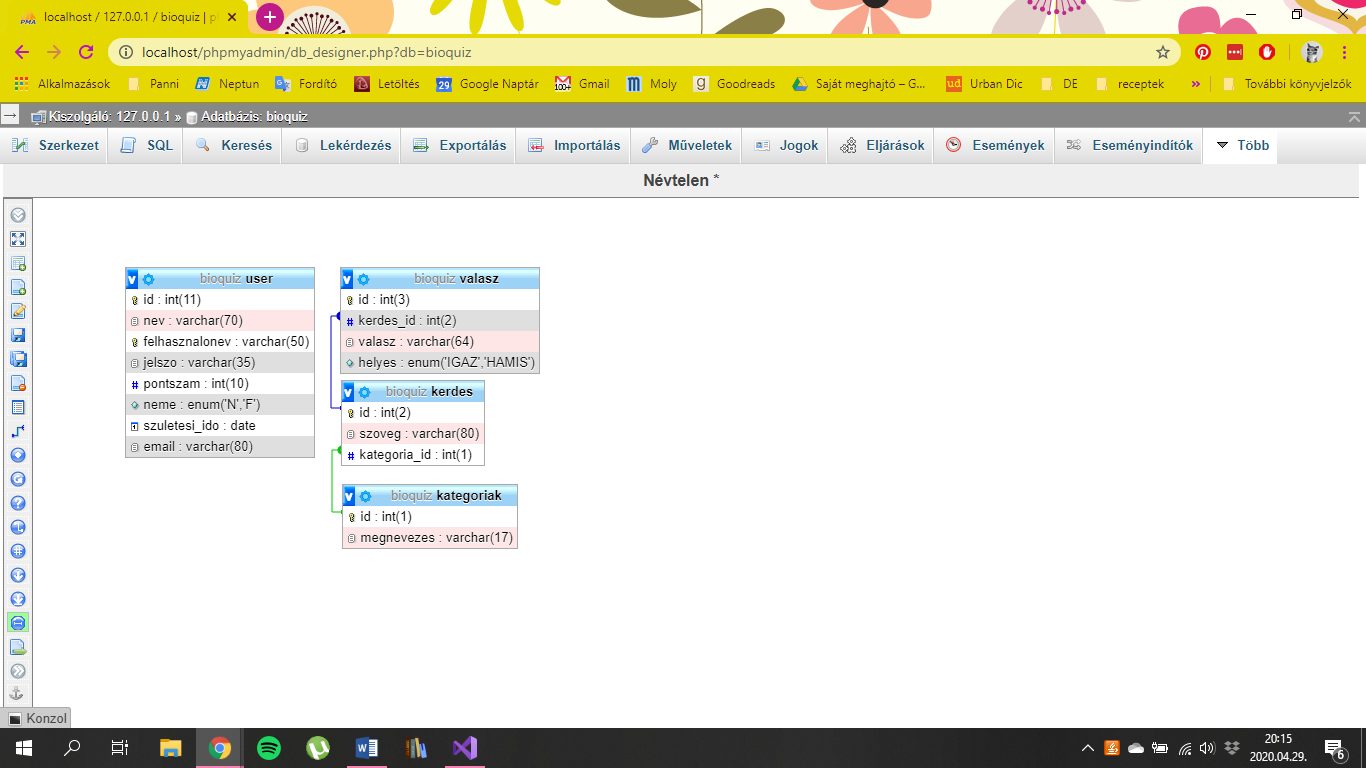
A *kategoriak* tábla beállításánál típusaként *int(1)* lett, mivel a kategóriák száma nem haladja meg a 10-t, ezért nem adtam neki nagyobb értéket, illetve *AUTO-INCREMENT-*t lett beállítva. A *megnevezes* oszlop esetében a típus *varchar(17)*, mivel a kategóriák egyes nevei nem hosszabbak 17 karakternél.

A *kerdes* tábla 3 elemet tartalmaz: az *id*, mely *int(2)* típsú, a *szoveg* *varchar(80)* , és a *kategoria\_id int(1)* típussal. A kapcsolati nézetben beállítottam az idegen kulcs megszorításoknál egy *kerdes\_kat* nevű megszorítási tuljadonságot, mely *ON DELETE NO ACTION, ON UPDATE CASCADE* funkcióval rendelkezik, a hozzá tartozó osztlop a *kategoria\_id,* mely a *kategoria* táblában lévő *id* oszlophoz kapcsolódik.

A *user* táblába gyűjti felhasználok adatait, melyeket egyedi *id-*val lát el. Hozzá kapcsolja a felhasználó adatait: név, felhasználónév, jelszó, pontszám, email. A Jelszó, egy MD5, kiterjesztéssel lett titkosítva. Az adattípusokat tekintve, a *nev*, a *felhasznalonev, a jelszo,* és az *email* mind *varchar*, a hossz pedig mindegyik a sajátjának megfelelő hosszússágú. A *pontszam* *int(10)* típusú.

A *valasz* tábla tartalamazza a feltett kérdésekre az adott lehetséges válaszokat. A szerkezetét tekintve 4 elemből áll: *id,* mely *int(3)*, *kerdes\_id* *int(2), valasz varchar(64), helyes(ENUM(’IGAZ’,’HAMIS’))* típusokkal. Ezek közül kiemelném, az utolsó, *helyes* oszlopot, melynek típusa *ENUM,* melynek értékkészletét én határoztam meg, ebben az esetben, igaz vagy hamis opció lehetésges. Az *id AUTO-INCREMENT* tulajdonsággal rendelkezik. A kapcsolati nézetet tekintve, itt is tettem egy idegen kulcs megszorítást, mely a *kerdes\_valasz* néven fut, *ON DELETE NO ACTION, ON UPDATE CASCADE* tulajdonsággal. A hozzá tartozó oszlop a *kerdes\_id*, ami a *kerdes* tábla, *id* oszlopához tartozik.

Ilyen beállításokkal elkészült a *bioquiz* nevű adatbázisom, melynek a kapcsolattáblája lentebb látható (1. ábra):



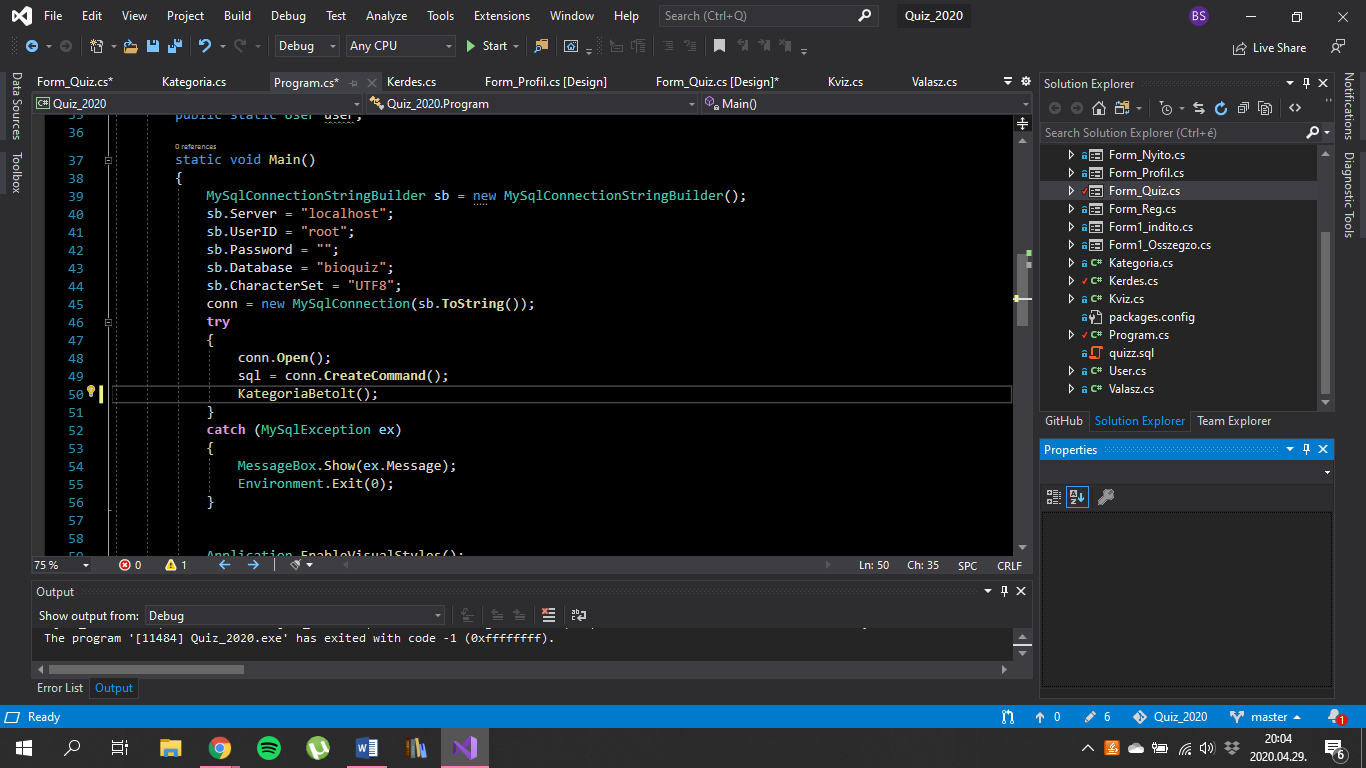
**1. ábra - Kapcsolattábla**

## Program

Ezután következett a Visual Studio telepítését követően, magának az alkalmazás vázának a létrehozása.

Az alkalmazás a BioQuiz nevet kapta, értelemszerűen, mivel egy kvízről van szó, illetve mert a témáját tekinte a biológia kisebb részegységeire fókuszál.

Az adatbázis, összekapcsolása és megnyitása, a programmal a *MySqlConncetionStringBuilder()-* függvénnyel történt, illetve konnektor létrehozásához a *Program.cs*-ben egy osztályt hoztam létre, melynek feladata a konnektor megnyitása, illetve lezárása. Visszatérési értékként magát a kapcsolatot hozza létre (2. ábra).

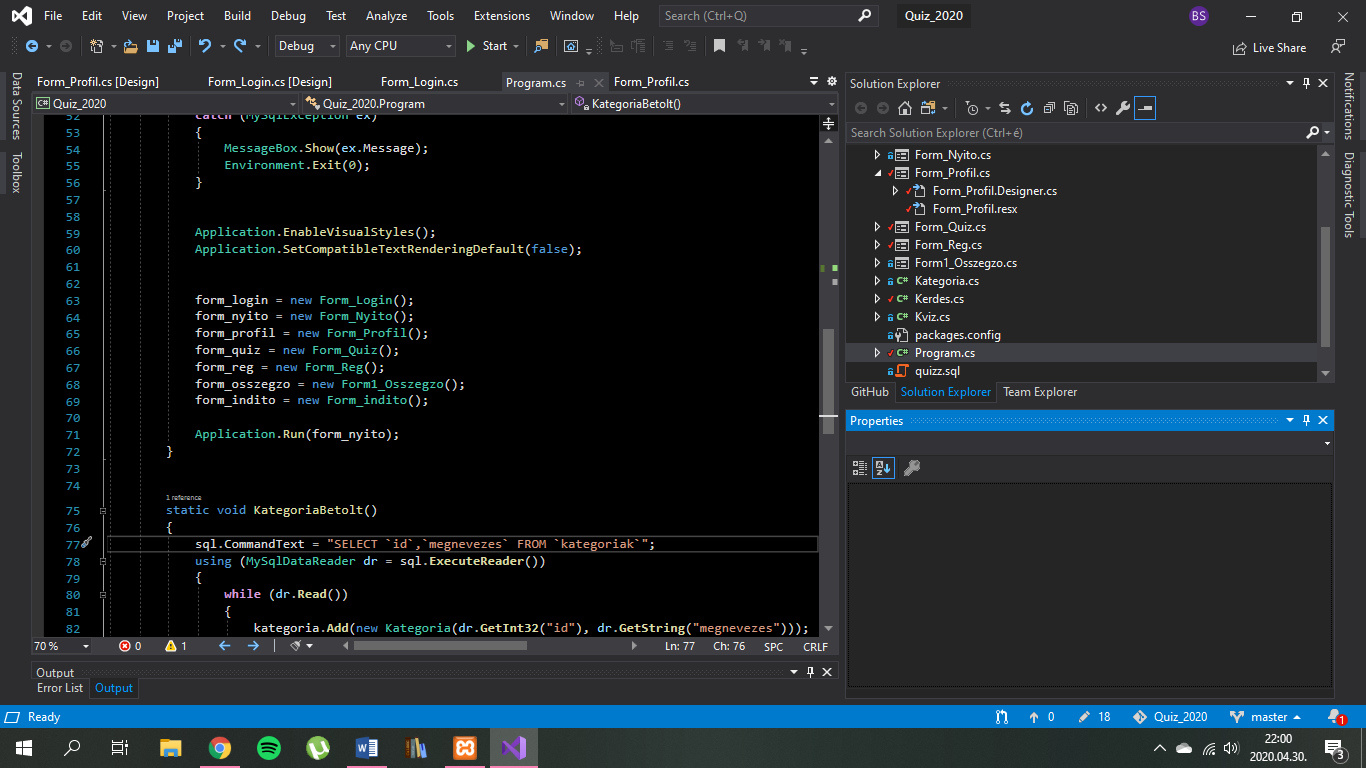


**2. ábra - Adatbázis és a program összekapcsolási programsorai**

Ez a pár program sor, tartalmazza a szerver nevét, mellyel az összekapcsolás létrejött, illetve az adatbázis nevét és karakterkészletét.

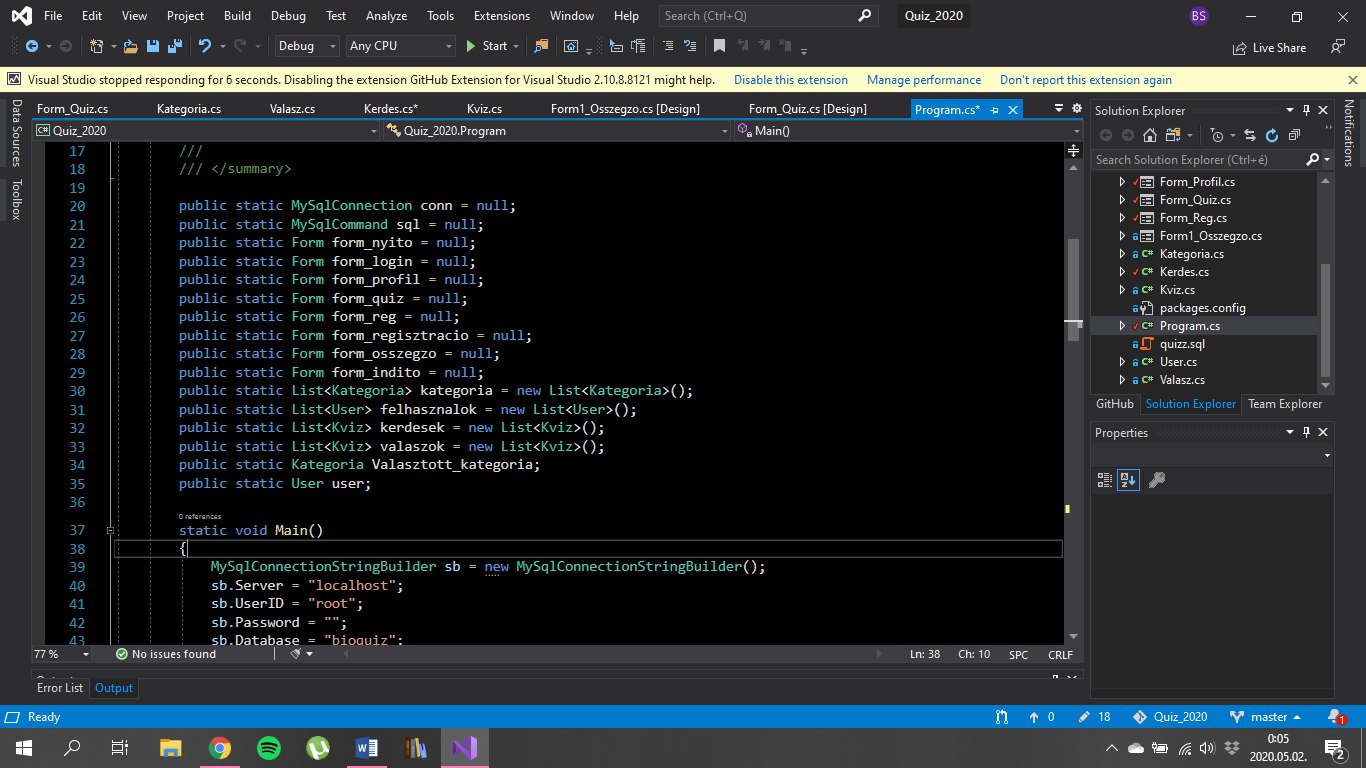
Aztán következett, hogy létrehozzak különböző Form-okat, a program azon részét mutatják, amiket a felhszanáló lát: *Form\_Nyito*, *Form\_Login*, *Form*\_*Regisztracio*, *Form\_Osszegzo*, *Form\_Profil, Form\_Quiz, Form\_indito.*

A fent felsorolt *Form-*okat a következő képen hívtam meg, értékadás után (3. ábra):



**3. ábra - A Form-ok meghívási program sorai**

A Program.cs –ben statikus osztályok deklarálása is szükségszerű volt, mely az alábbi ábrán látható (4. ábra):

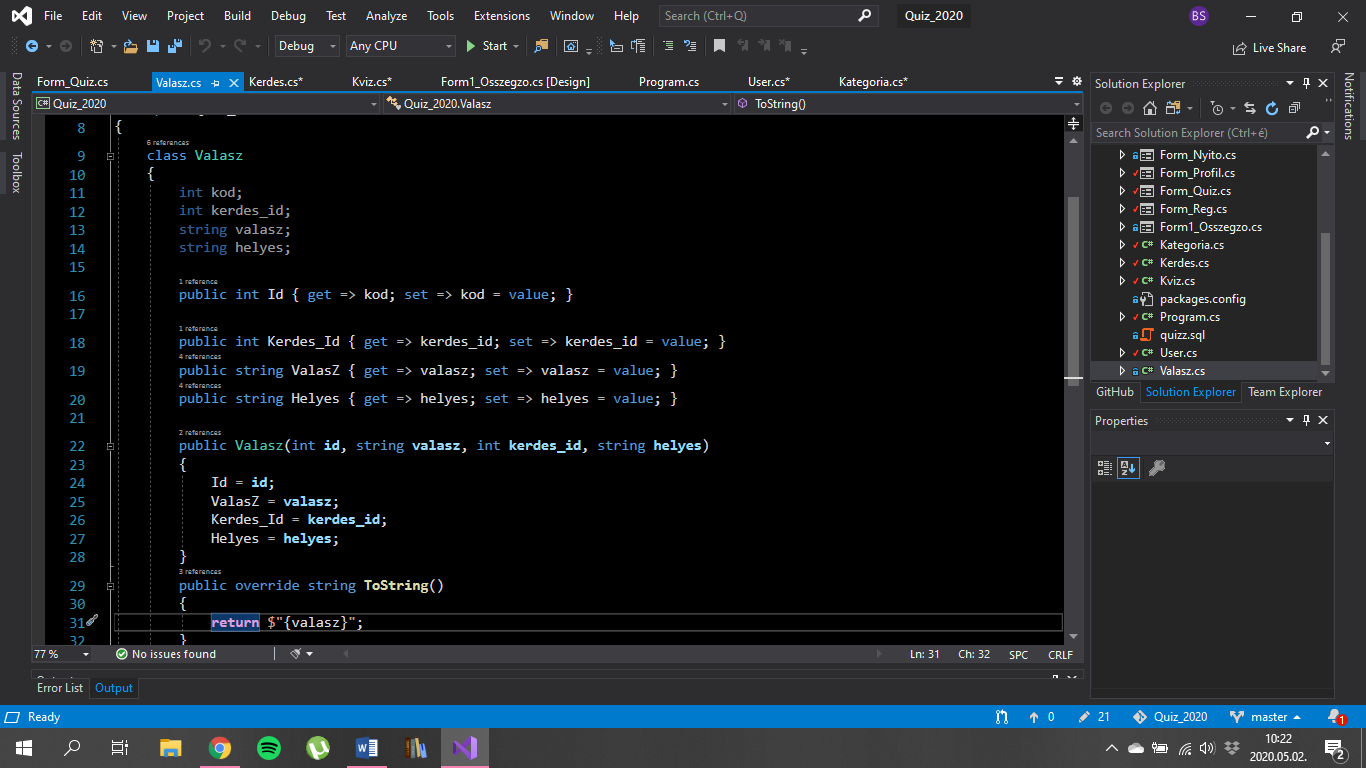


**4. ábra - Statikus osztályok létrehozása**

Mindegyik Form változójának *null* értéket adtam, illetve az egyes osztályoknak (Kategória, User Kvíz) listát hoztam létre. A kvíz listájába egyaránt eltároltam a kérdéseket és a válaszokat. A Kategória osztály statikus változója a *Valasztott\_kategoria,* illetve a User osztályban tárolódnak el a *user* változóban a felhasználó adatai.

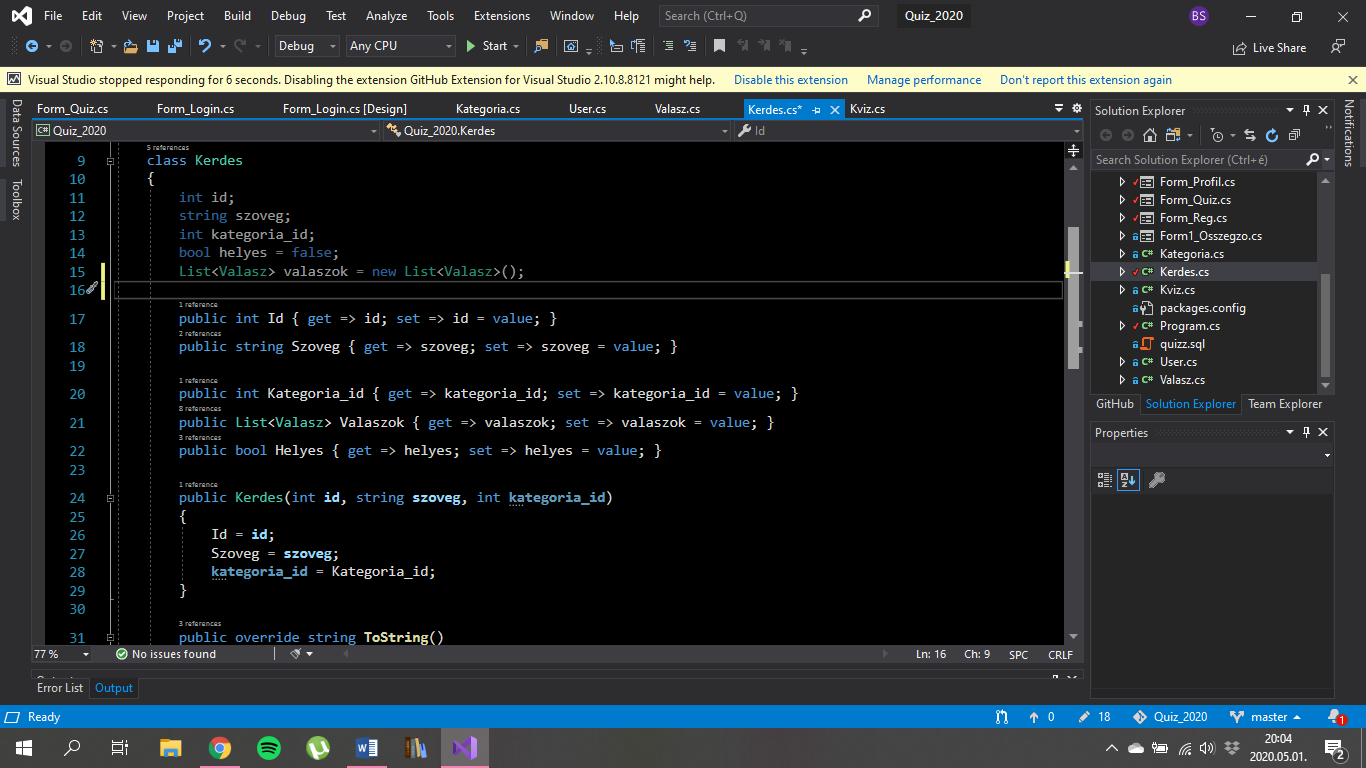
Kitérnék még a létrehozott osztályokra is, melyeknek alapja az adatbázis volt.

A *Válasz* osztályban létrehozott változok, az adatbázisban létrehozott oszlopok neveinek felel meg (5. ábra). Ezeknek a változóknak értéket adtam, és ezután egy publikus *Valasz()* metódus visszatérési értékeinek állítottam be, melyet később többek között a *Kerdes* osztályban is meghívok.



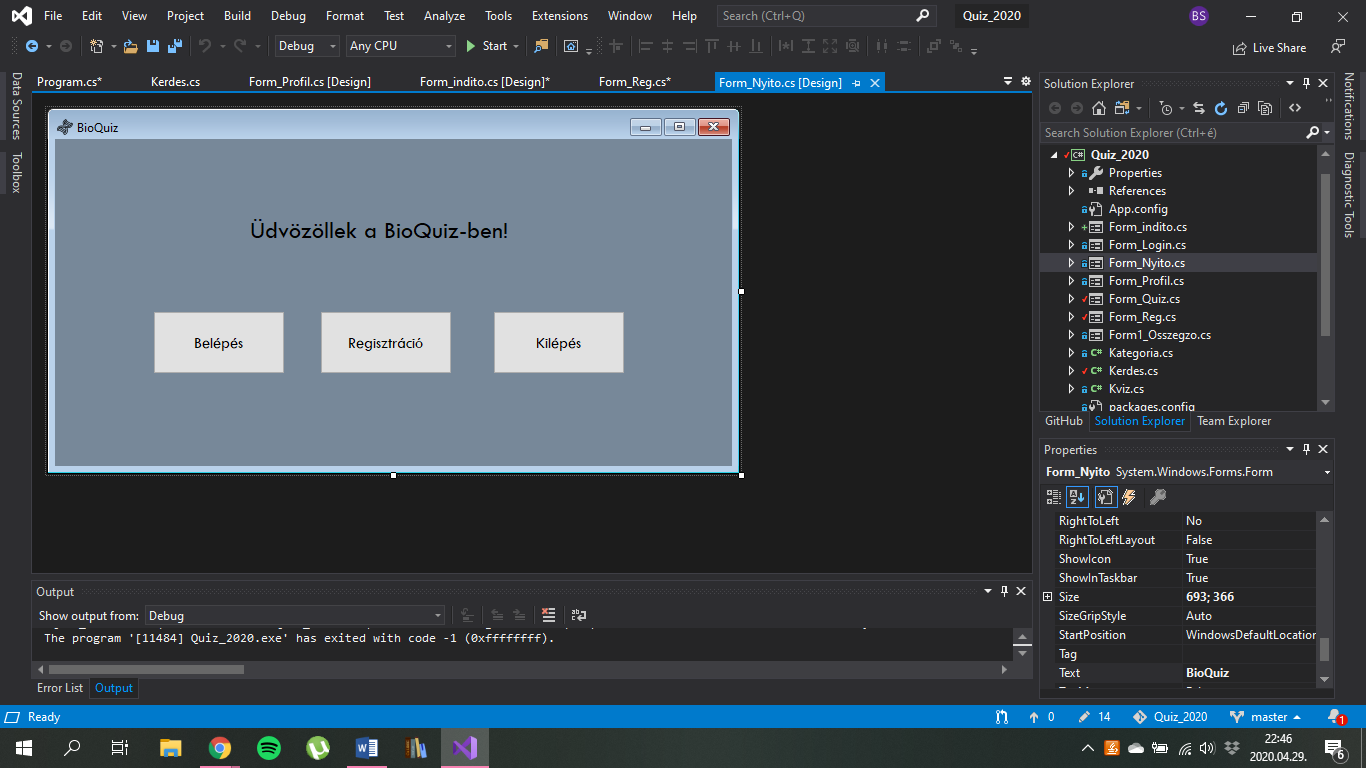
**5. ábra - A válasz osztály visszatérési értékei**

A *Kerdes* osztályba gyűjtöttem össze mind a kérdéseket (6. ábra), melyek adott kategóriákhoz tartoznak, illetve a válaszokat, amelyekből 3 jut egy egyedi azonosítós kérdésre. Az adatbázis, Kérdés táblájának, 3 oszlop azonsoítója mellett, itt deklarálva van még egy *helyes* változó, mely bool adattípussal rendelkezik, a válasz igazság vizsgálata miatt(lásd lentebb). Illetve deklarálva van még egy *Valasz* lista is, mely a Válasz osztály változóit tartalmazza *valaszok* változó alatt.



**6. ábra - Kérdés osztály**

A *Form\_Nyito*, amellyel a programba belépve legelőször találkozik a felhasználó, elindítást követően (7. ábra). Az első ablak egy üdvözlő feliratot és 3 gombot tartalmaz, amelyek egy, az utasításhoz illő *Form-*hoz fogja irányítani a felhasználót.



**7. ábra - A nyitó ablak, amely a felhasználót üdvözli a programba való belépéskor**

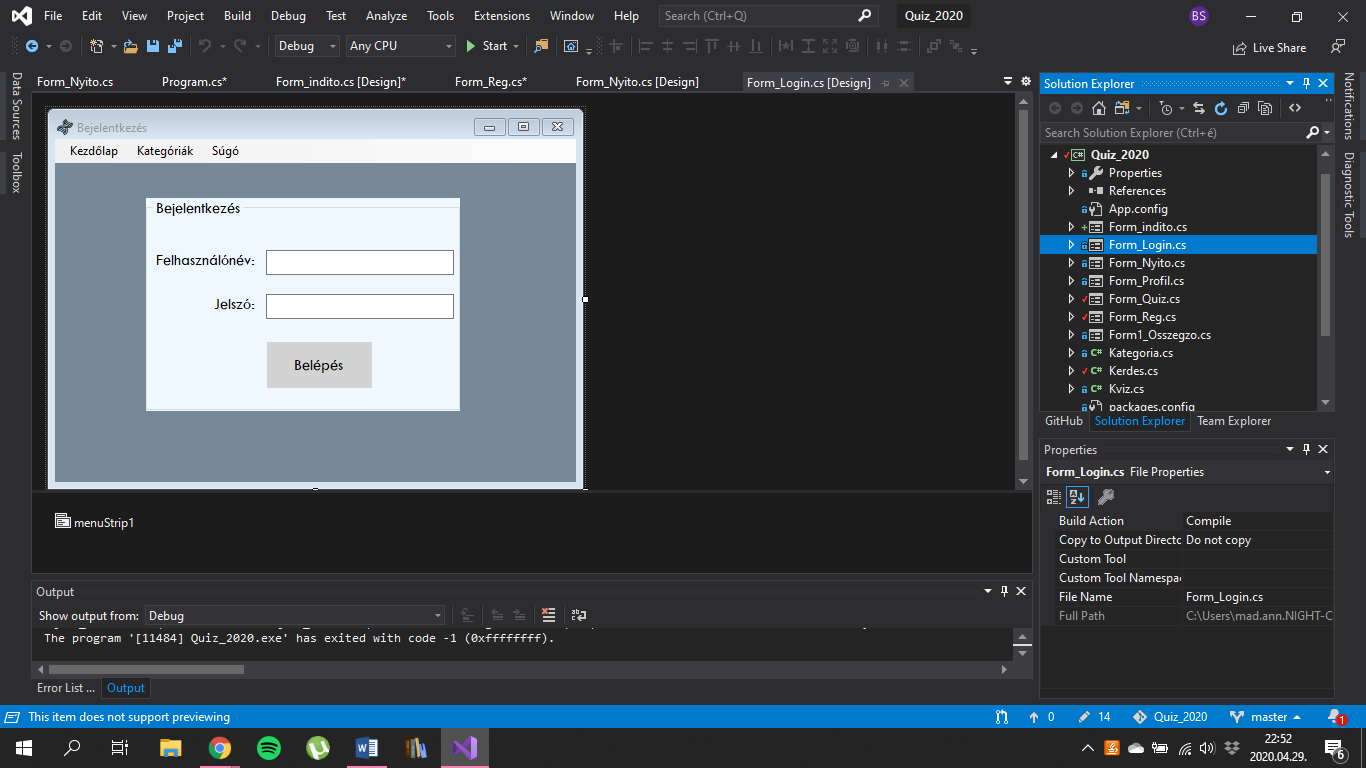
Belépés, ahol a felhasználó, a már előzetesen regisztrációt követően beléphet. Egy Regisztráció gombra kattintva, egy regisztrációs felülethez jut a felhasználó, amit ha kitölt, az adatai tárolódnak az adatbázisban, a regisztrált felhasználónévvel és a hozzá tartozó jelszóval be tud majd lépni.

Amennyiben regisztráció közben akar a felhasználó kilépni, a program egy figyelmeztető ablakkal adja a tudtára, hogy adati el fognak veszni, amennyiben félbehagyja a regisztrációt.

A Kilépés gomb, pedig magából az alkalmazóból lép ki. Egy felugró ablak, kérdezi meg a felhasználótól, hogy biztosan ki akar-e lépni.

A *Form\_Login* ablakhoz (8. ábra), abban az esetben jut el felhasználó, hogyha a Kezdőlapon a *Belépés* gombra kattintott előzetesen. Regisztrált, és az adatbázisban eltárolt felhasználónév, és az adatbázisban MD5-ös titkosítású jelszavával beléphet.

Amennyiben valamelyik mező üresen marad, akkor egy hibaüzenet fog felugrani, ami figyelmezteti a felhasználót az adott hiányosságra.

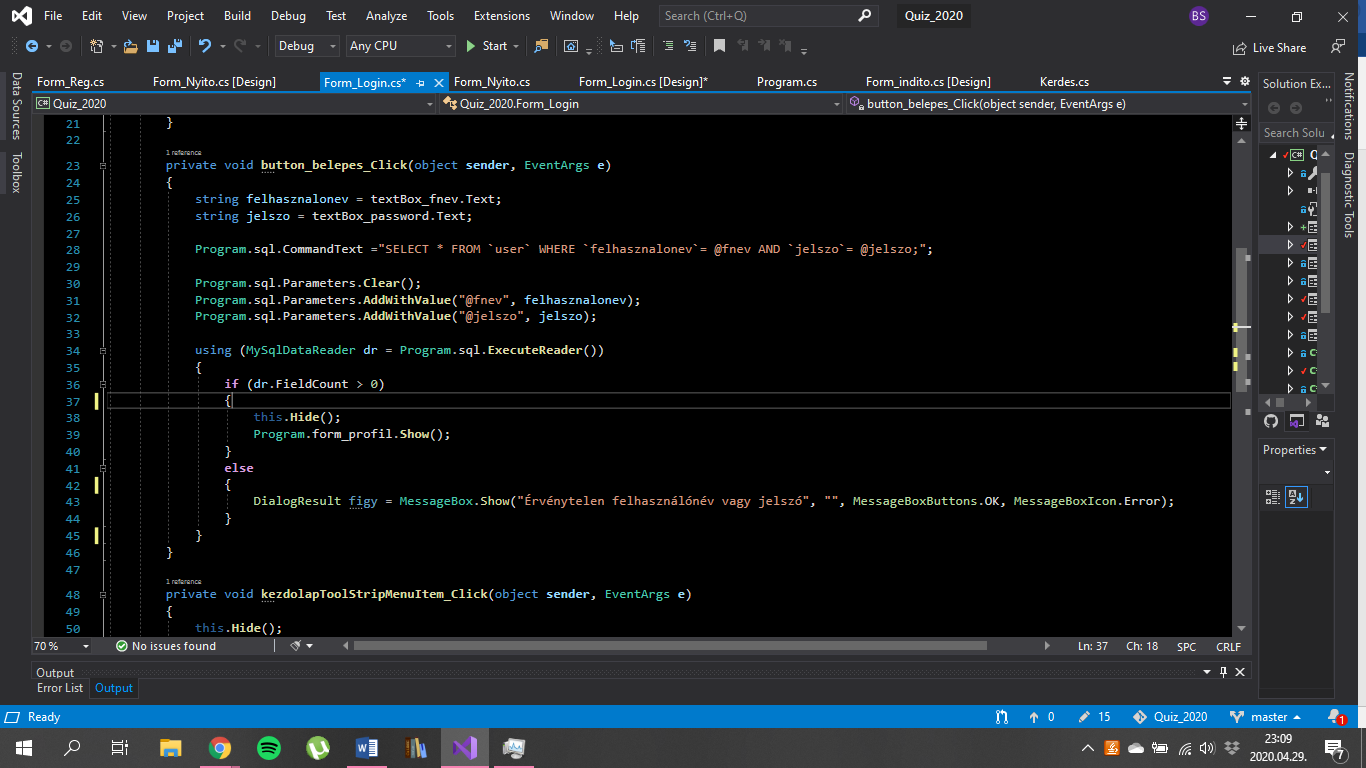


**8. ábra - A bejelentkezés ablak**

Az ablakban található egy menüsor, melynek lenyíló füleivel a felhasználó, visszatérhet a Kezdőlapra, vagy választhat a kvíz kategóriái közül. Utóbbi addig nem lehetséges, amíg a felhasználó be nem jelentkezett, de láthatóak számára a választási lehetőségek.

Amennyiben szüksége lenne segítségre, akkor egy Súgó is a segítségére lehet, ami a legtöbb ablakban megtalálható.

A felhasználónév és a hozzá tartozó regisztrált jelszót megadva, a felhasználót, a program a Profil ablak megnyitásával fogadja.



**9. ábra - A belépés programkódja**

Az 9. ábrán található az a kód, ami a *Textbox-ok*ba bevitt adatokat dolgozza fel. Ez gyakorlatban úgy nézett ki, hogy a deklarált *felhasznalonev* és *jelszo* változónak paramétert adtam, melyeket a program lekérdez az adatbázisból. Ezt az útvonalat is megadtam, az adatbázis és program között.

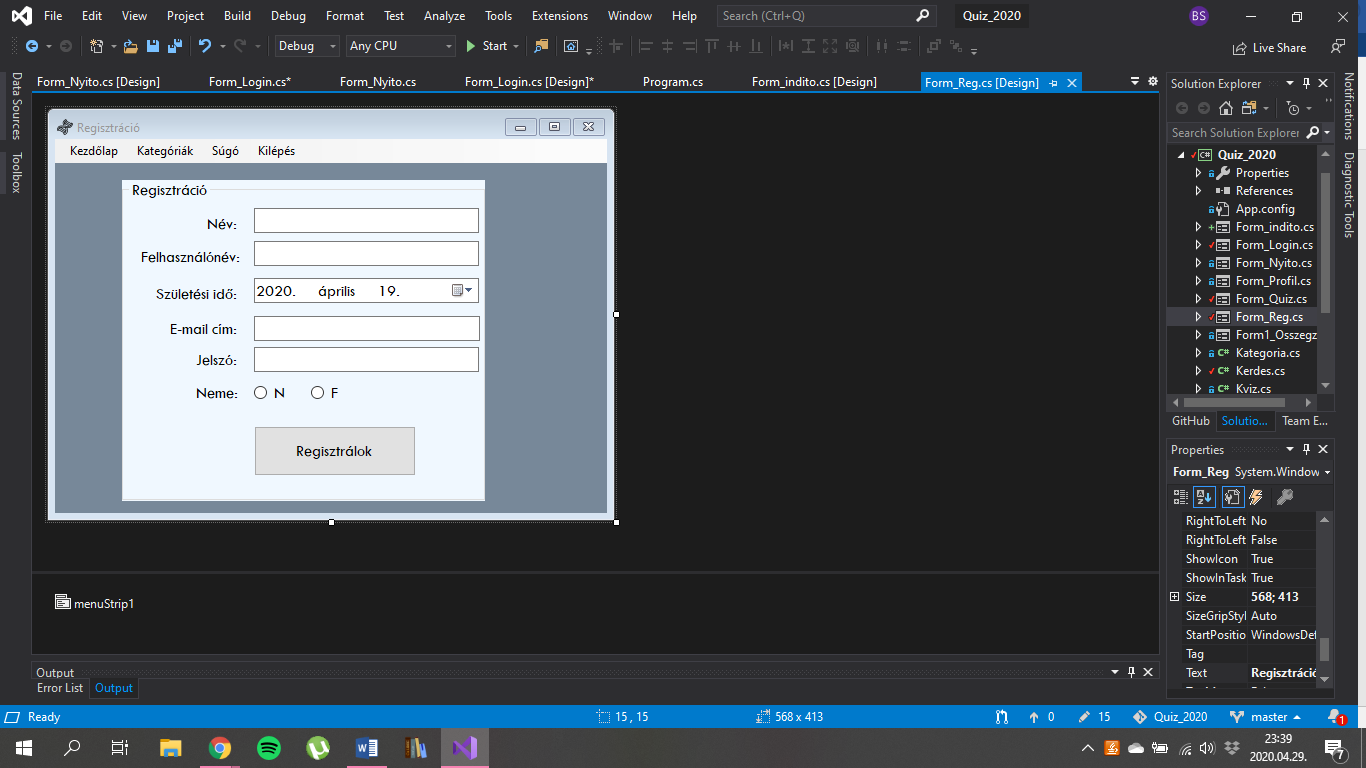
A Regisztrációsablak szintén a Kezdőlapról érhető el. Ez az ablak is kapott egy menüsort, ahol a *Kategóriák* fülre kattintva, nem történik semmi, mivel a nem regisztrált felhasználó számára nem elérhető ez az opció.

A többi gomb ugyanúgy működik, mint a *Belépés* ablaknál. A 10. ábra szemlélteti, hogy hogy is néz ki a regisztrációs űrlap, illetve, hogy milyen adatokat kell megadnia a felhasználónak.

A regisztrációs űrlapon meg kell adni, a felhasználó nevét, egy egyedi felhasználónevet, egy születési időt, ami nem lehet az adott nap dátumánál jövőbbre, és visszamenőleg is 1900.01.01. a legkorábban megadható születési időpont.

Ki kell még tölteni az e-mail-re vonatkozó mezőt, illetve a jelszó mezőt, ahol a karakterek nem láthatóak már a beírásnál sem.

Két radioButton teszi lehetővé, hogy a felhasználó (későbbi statisztika létrehozása miatt), megadhassa a nemét.

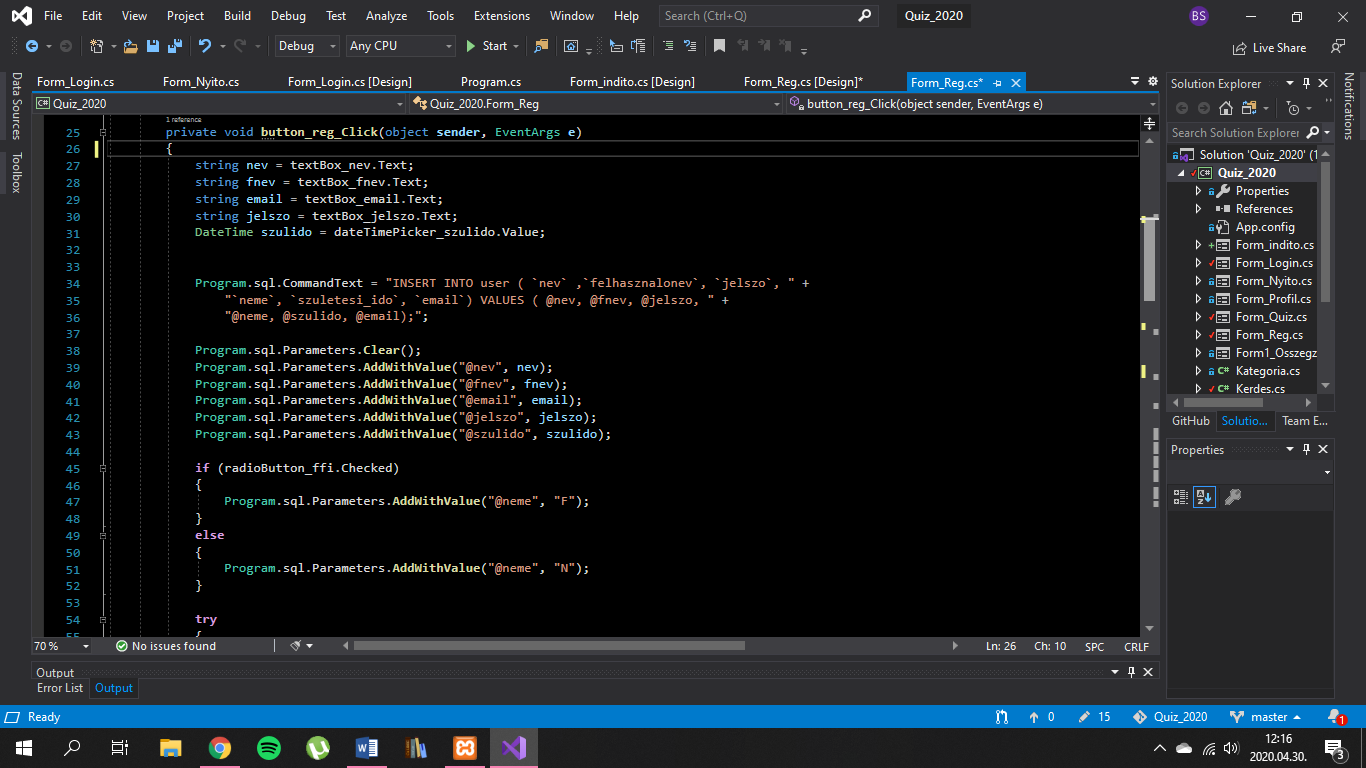


**10. ábra - A regisztrációs ablak**

A felhasználó által megadott adatok, tárolódnak az adatbázis *User* táblájában, és így később sem vesznek el. A megadott jelszót MD5 titkosítás védi. Szintén erről a tábláról történt az útvonal kijelölése, illetve az egyes mezőknek történő értékadás a lentebb látható programkód segítségével (11. ábra).

A „*Regisztálok”* gomb megnyomását követően a program elküldi az adatbázisba az adatokat, ami úgy történik, hogy *Program.sql.Commandtext* hívódik meg, aminek segítésgével a szerverre küldöm az adatokat. Itt a textBox-okbólkiveszem az adatokat egyesével, és változokban tárolom el azeket.

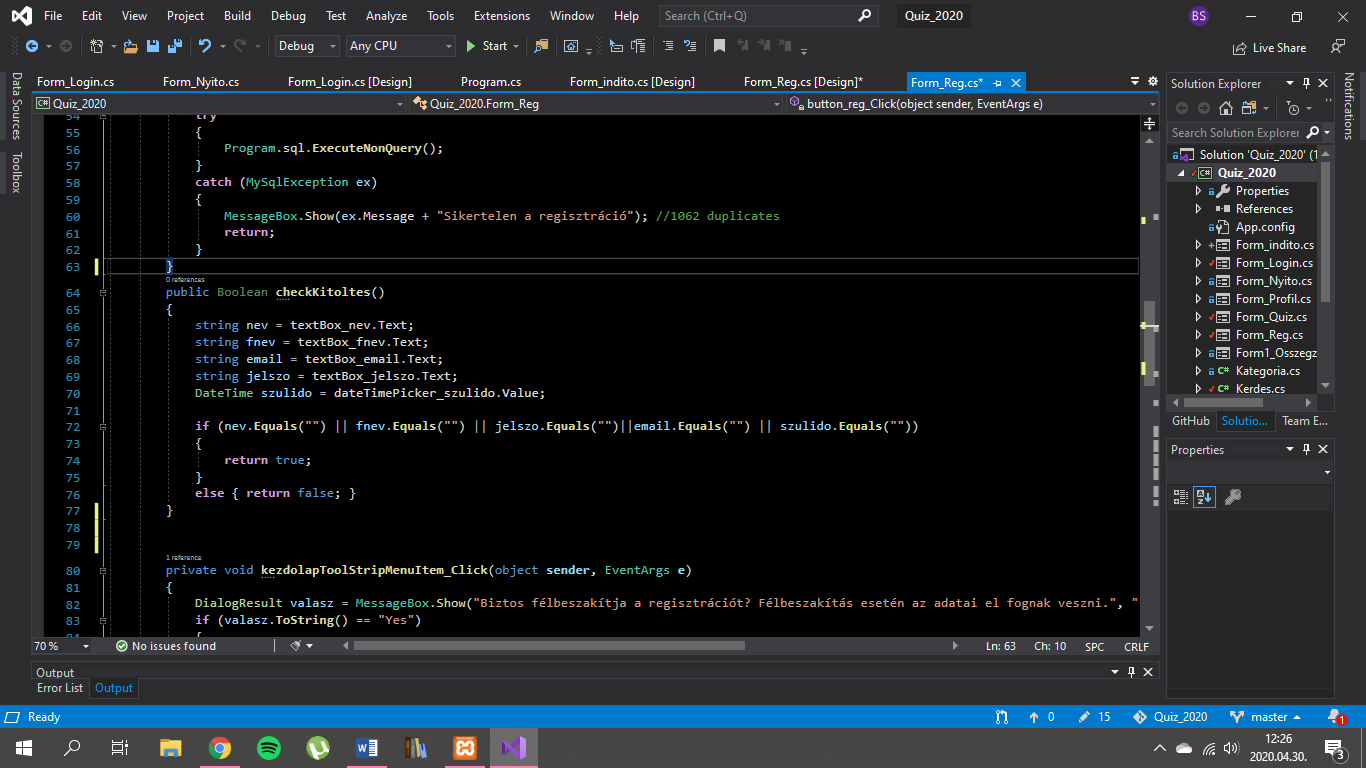
Az egyes mezőknek az *AddWithValue* paranccsal adok értéket, melyek tárolódnak az adatbázisban a felhasználó egyedi azonosítója alatt.



**11. ábra - "Regisztrálok" gomb parancsa**

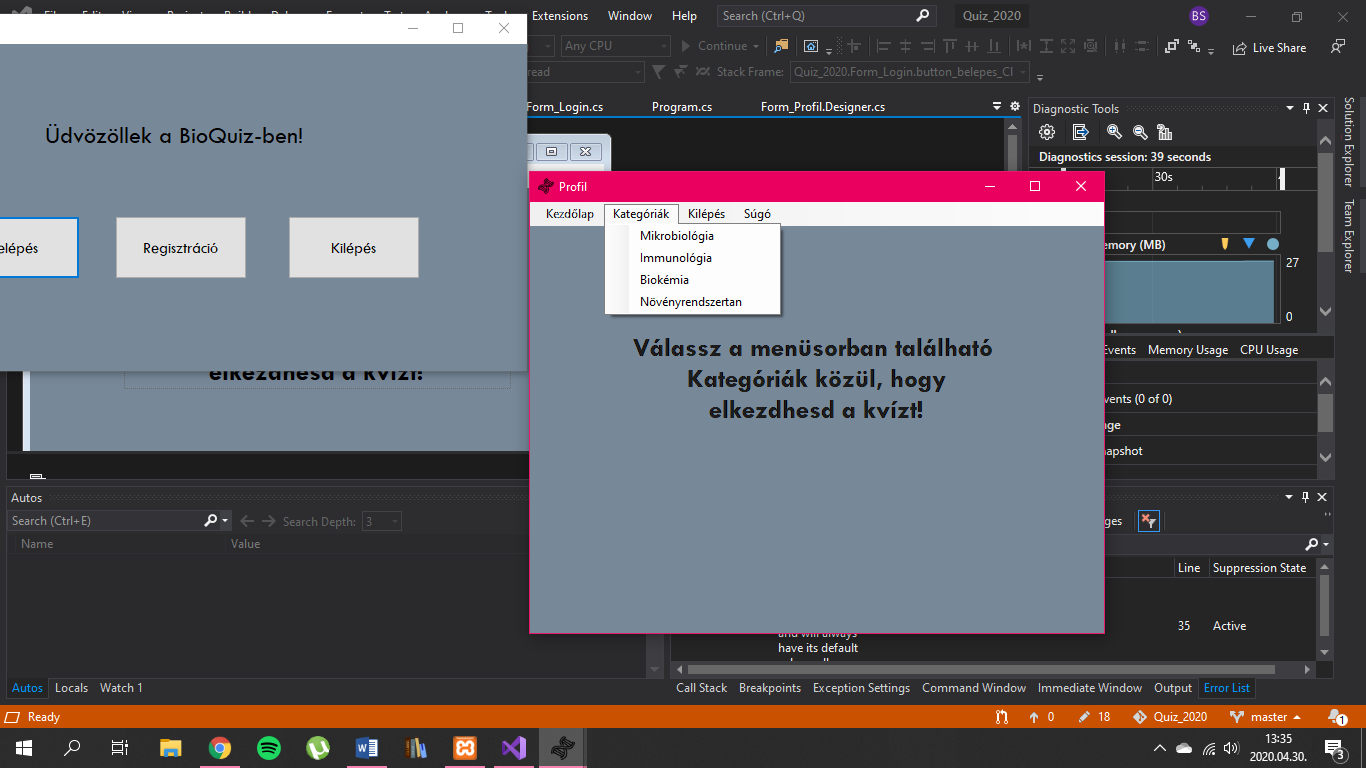
Abban az esetben, ha a felhasználó nem töltött ki egy, vagy több mezőt hibaüzenet jelzi, a javítandó problémát. Ezt egy Boolean() függvénnyel oldottam meg (12. ábra), amely megvizsgálja, hogy van-e olyan mezőm ahova nem történt adatbeírás, illetve ezáltal érték átadás sem. Igaz értékkel tér vissza, ha valamelyik mező üresen maradt és a „Regisztrálok” gombra való kattintás után egy hibaüzenetet dob fel a program.

A program Regisztrációs Form-ja azt is vizsgálja, hogy az adott felhasználónevet, amelyet akkor szeretnének regisztrálni, nem használja-e már más. Amennyiben már valaki regisztrált az adott felhasználó névvel, akkor a program azt jelezni fogja a felhasználónak egy hibaüzenet formájában.



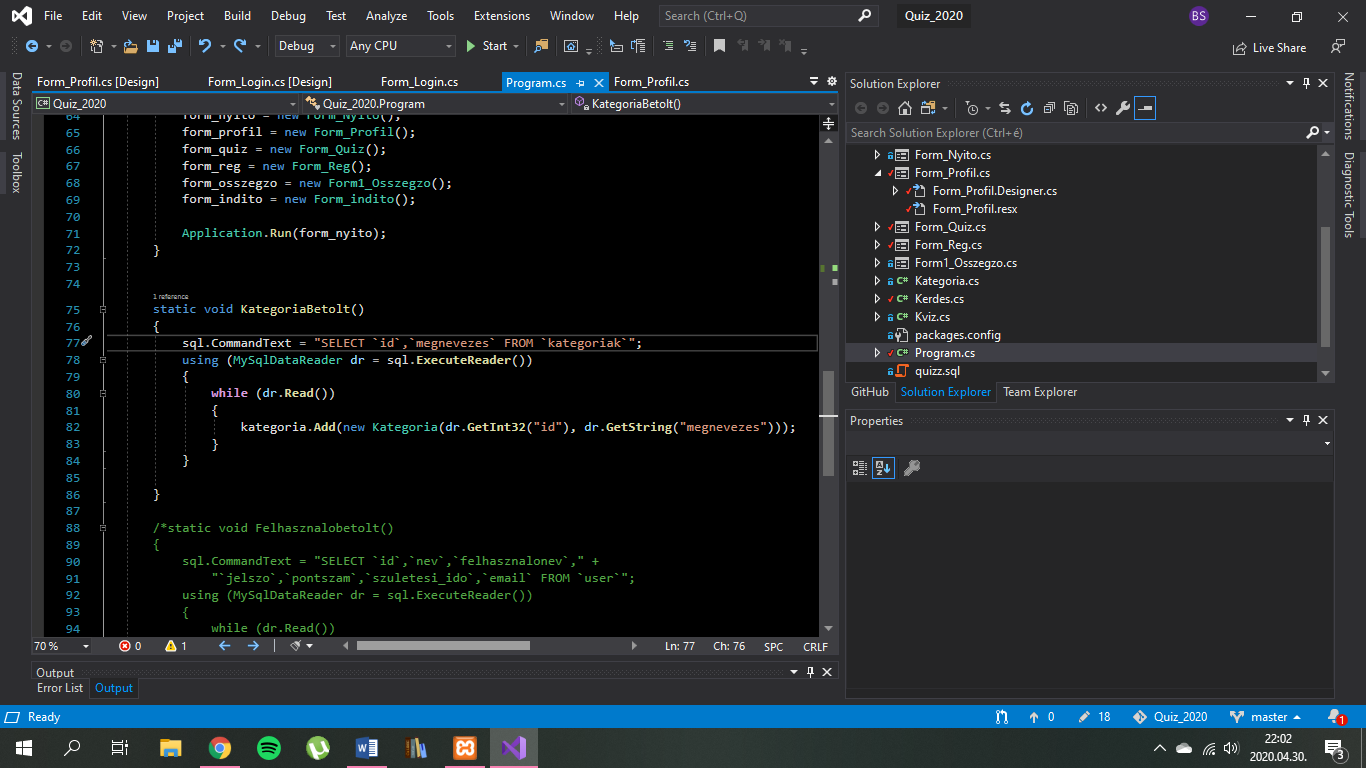
**12. ábra - Annak a vizsgálata, hogy regisztráció során minden mező ki lett-e töltve**

A Profillenyíló menüsorából lehet elérni az összes válaszható kategóriát, amelyre kattintva, a kvíz fog betöltődni. A felhasználó választhatja azt is, hogy visszatér a Kezdőlapra, vagy Kilép a programból (13. ábra).



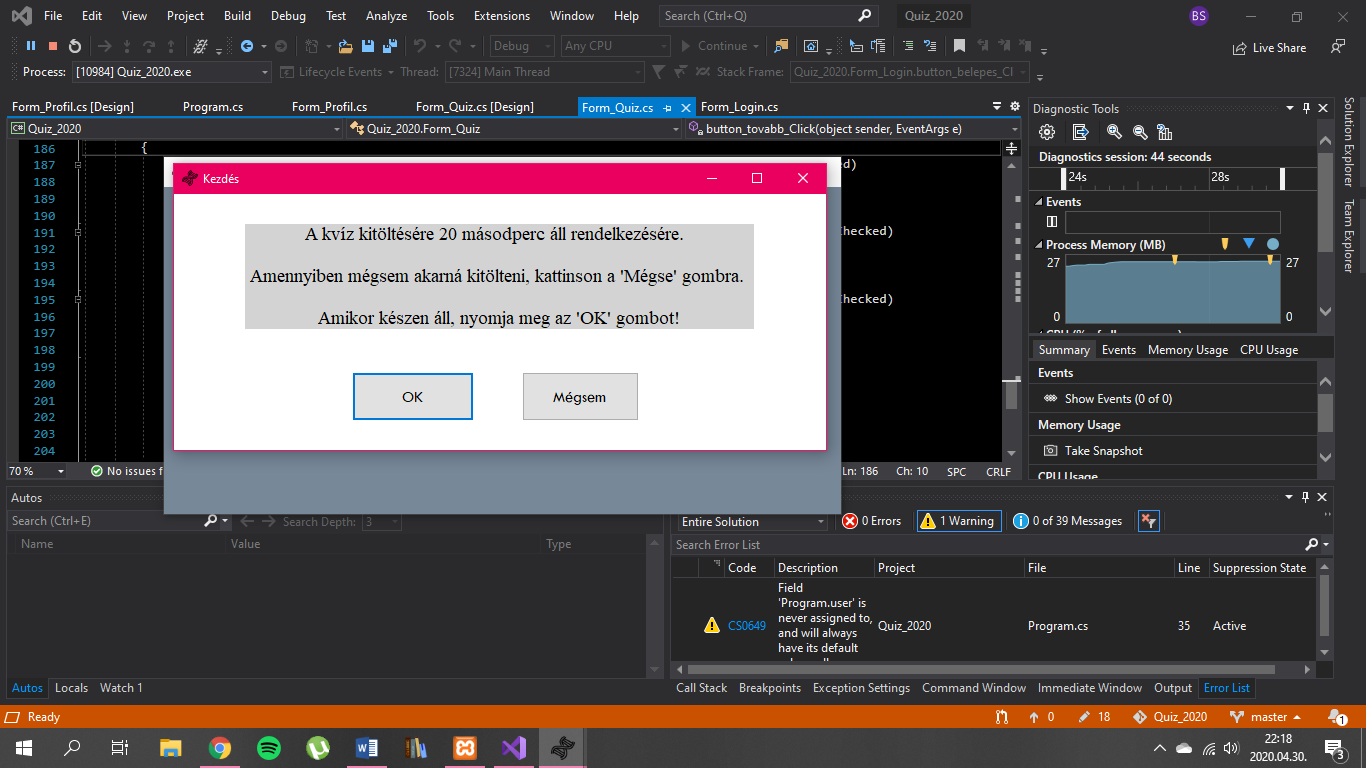
**13. ábra - A profil ablak és a Kategóriák lenyíló menüsora**

A választott kategória, az adatbázisból fogja betölteni a kérdéseket, illetve a hozzá tartozó lehetséges válaszokat. Ahhoz, hogy ez létrejöhessen egy *static void* függvényben adtam meg az elérési útvonalat *CommanText* parancs segítségével, illetve beolvassa egy *dr* változóval és értéket ad az egyes elemeknek, ami ebben az esetben a „megnevezéshez” tartozó „id” (14. ábra).



**14. ábra - Kategóriák függvénye**

A kategória kiválasztása után, felugrik egy ablak (15. ábra), mely figyelmezteti a felhasználót, hogy a kvíz kitöltésére 30 másodperc áll rendelkezésére. Amennyiben az ablak OK, gombjára kattint, felugrik a Kvíz ablak és a játék elkezdődik. A felhasználó választhatja a Mégsem gombot, amely esetben visszakerül a Profil oldalra.

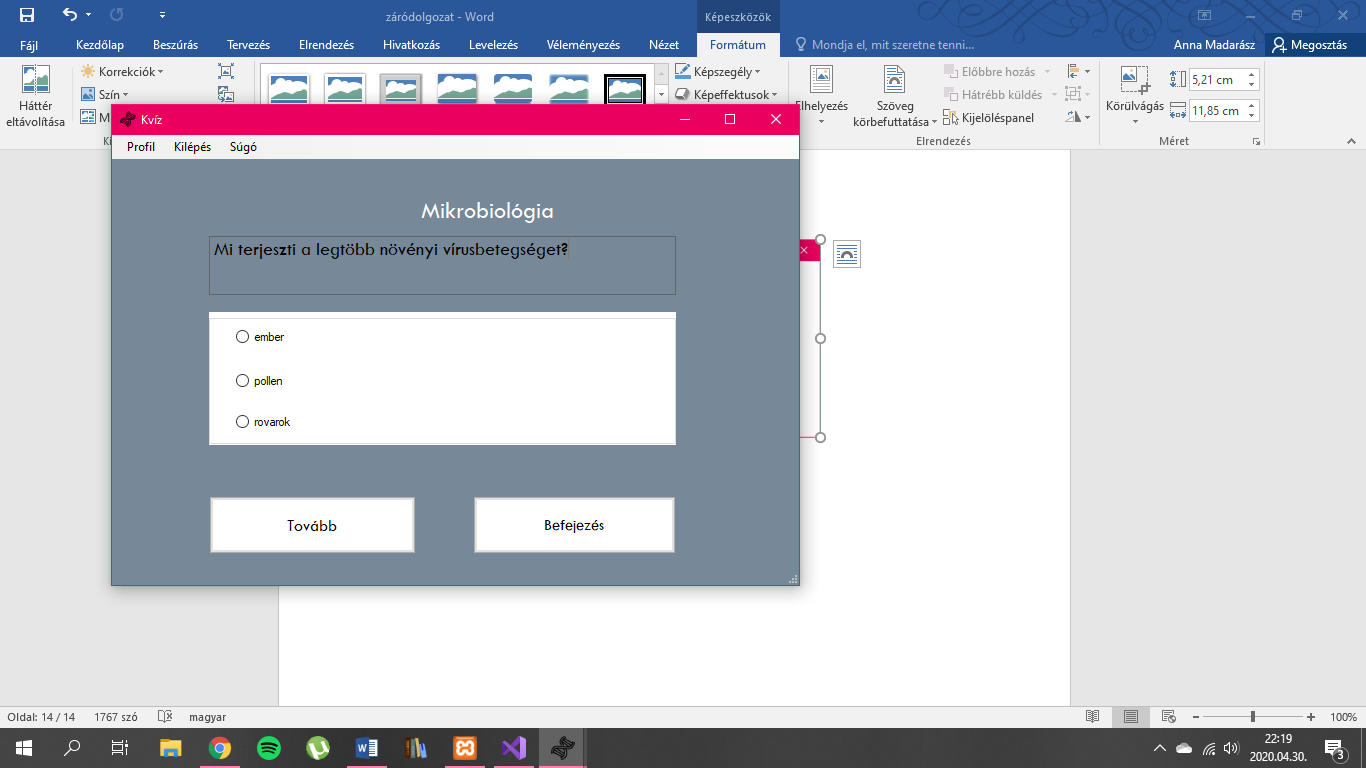


**6. ábra - Kvíz előtti figyelmeztető ablak**

Ezután a *Quiz* ablak felugrásával, a 30 másodperces időkorlát elkezdődik, és random sorrendben, kérdéseket generál a program. A 16. ábrán látható, hogy hogy is néz ki egy kvízkérdés, illetve a kvíz ablaka.

Egy *label\_kat* nevezetű címke a kérdés fölött annak megfelelően jeleníti meg a kategóriát, ahogy a felhasználó kiválasztotta azt.

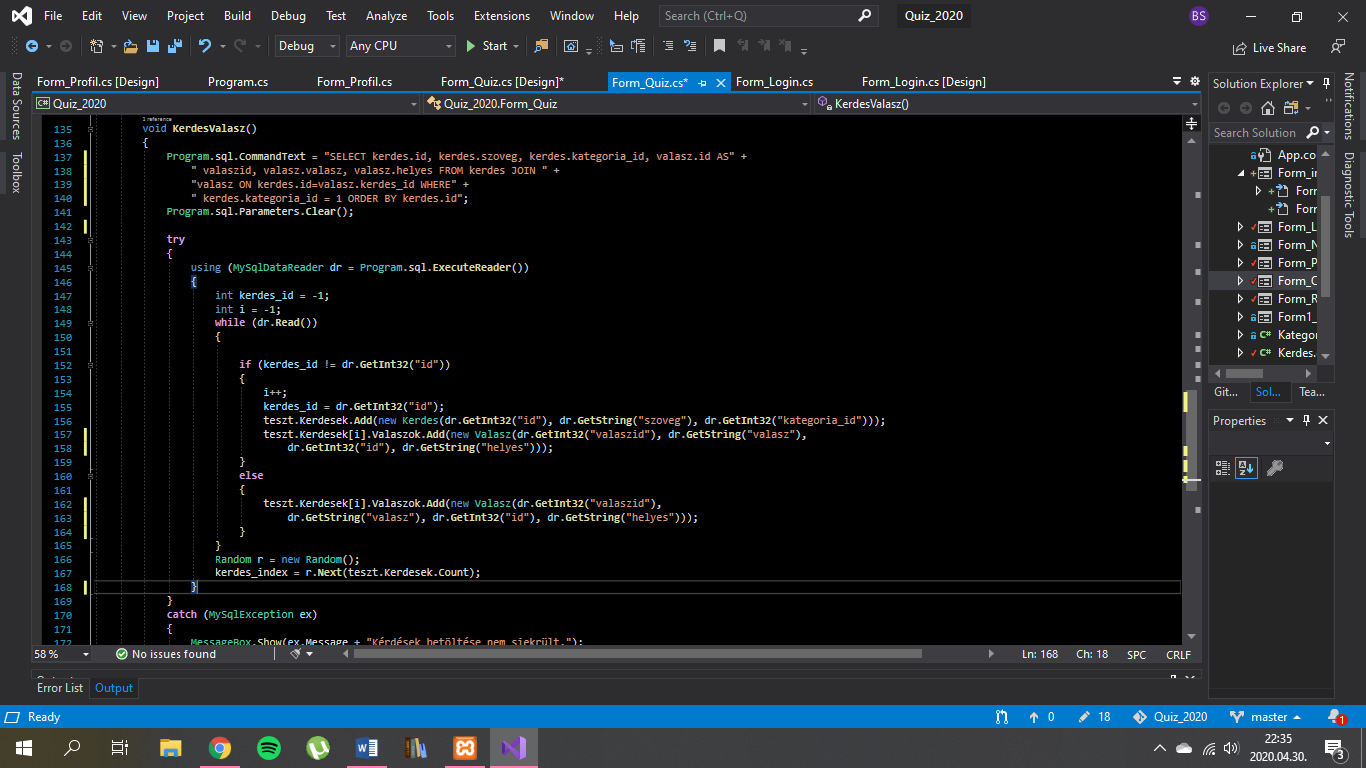
A Befejezés gomb választása esetén, a program egy figyelmeztető ablakot dob fel a felhasználónak. A Tovább gomb részletezése lentebb található.



**7. ábra - A kvíz ablak**

Ahhoz, hogy választott kategóriához tartozó kérdéseket adjon a program a felhasználónak az adatbázisból. Egy eljárást hoztam létre, aminek tartalma a képen látható. Az adatbázis olvasása egy *dr* nevűváltozóvaltörténi. Már előzőleg létrehozott osztályok változóinak segítségével vesszük ki az adatokat az adatbázisból. Mind a kérdések, és a hozzájuk tartozó lehetséges válaszok is bekerülnek a kiolvasásba (17. ábra).

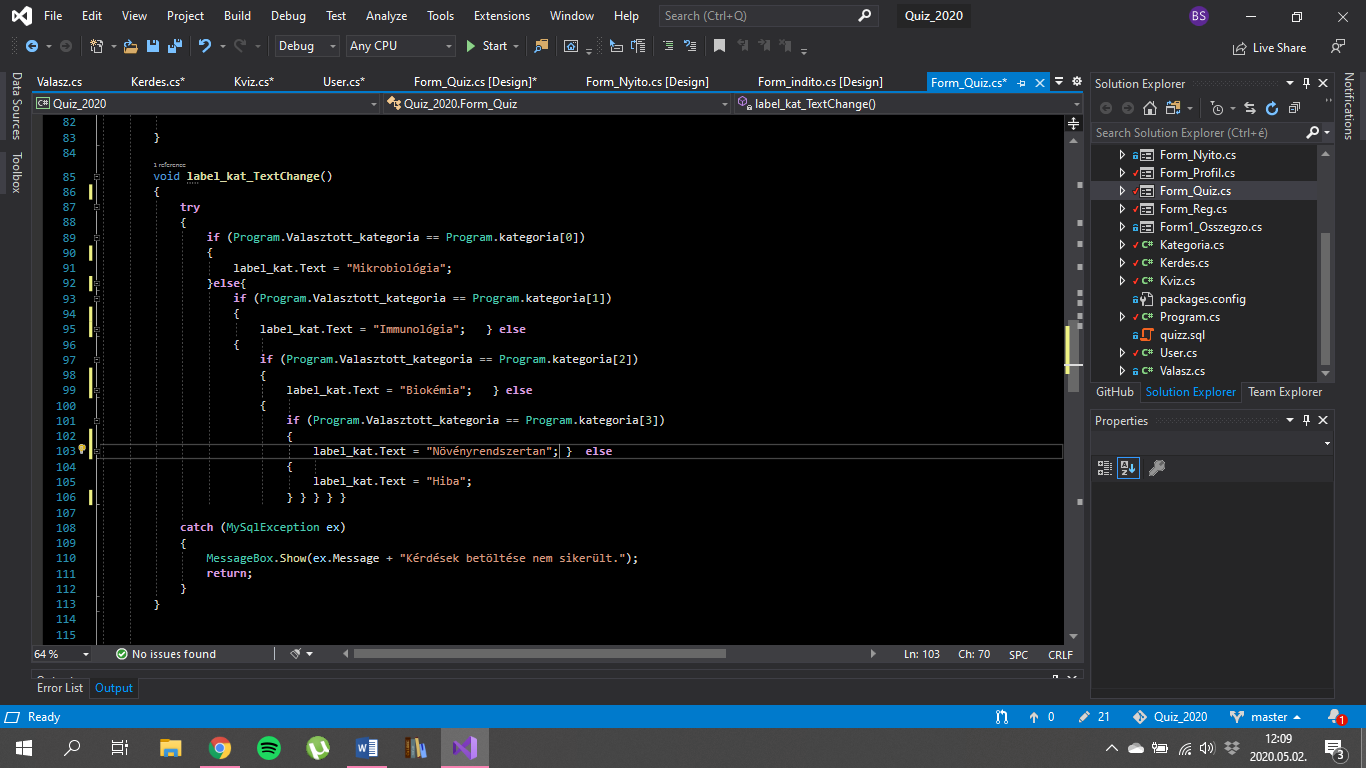
Továbbá feltételvizsgálattal biztosítottam, hogy a kérdések az azonosítójuk révén, ne ismétlődhessek.



**8. ábra - A kérdések és a hozzájuk tartozó lehetséges válaszok beolvasásának metódusa.**

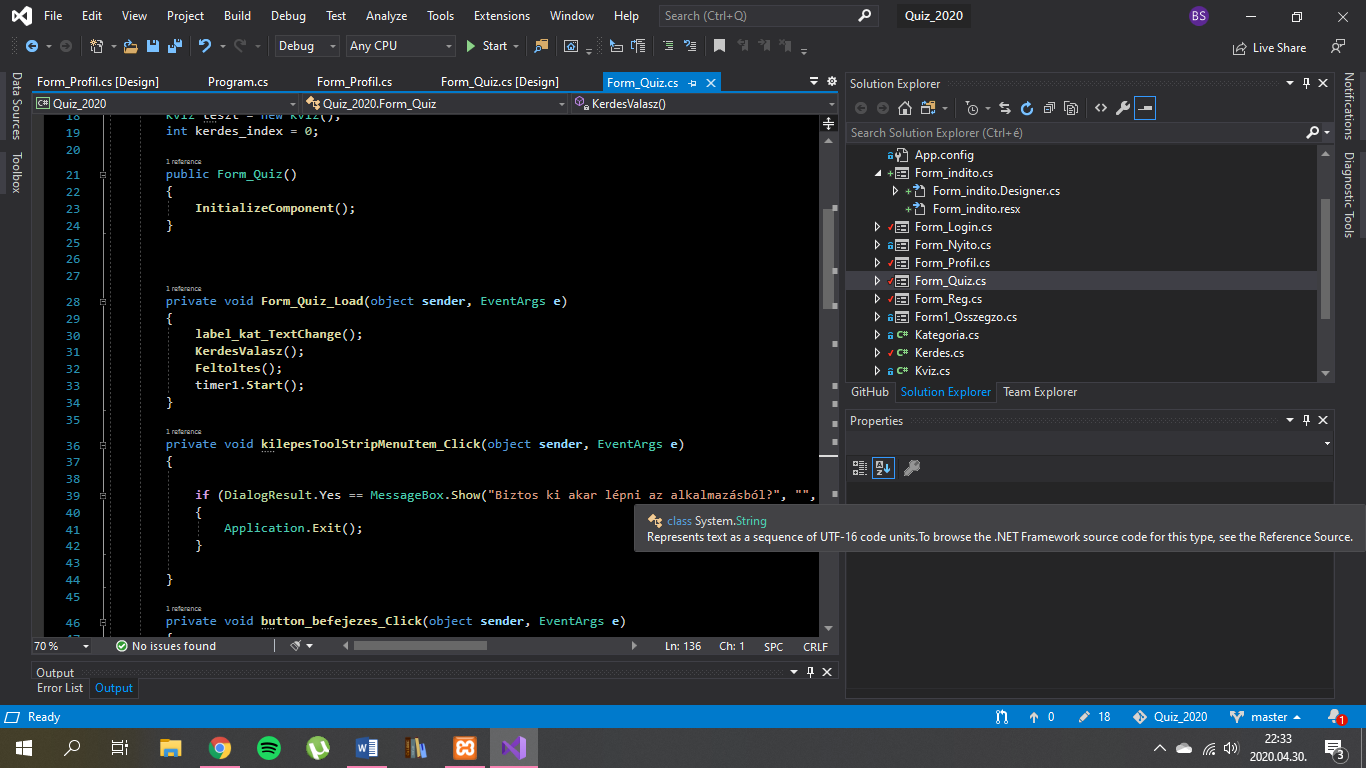
A *label\_kat\_textChange* címke, annak megfelelően automatikusan változik, ahogy a felhasználó kategóriát választott (18. ábra). Ezt *try()…catch()* utasítással futtattam végig, hogy egy hibaüzenet is kiírodjon, ha valami problémát talál a program.

Az egyes kategóriákhoz tartozik egy egyedi azonosító *id* az adatbázisban, aminek segítségével egy feltétel vizsgálattal meg tudtam állapítani, hogy melyik *kategoria\_id*-val egyezik meg a *Valasztott\_kategoria* metódus visszatérési értékével.

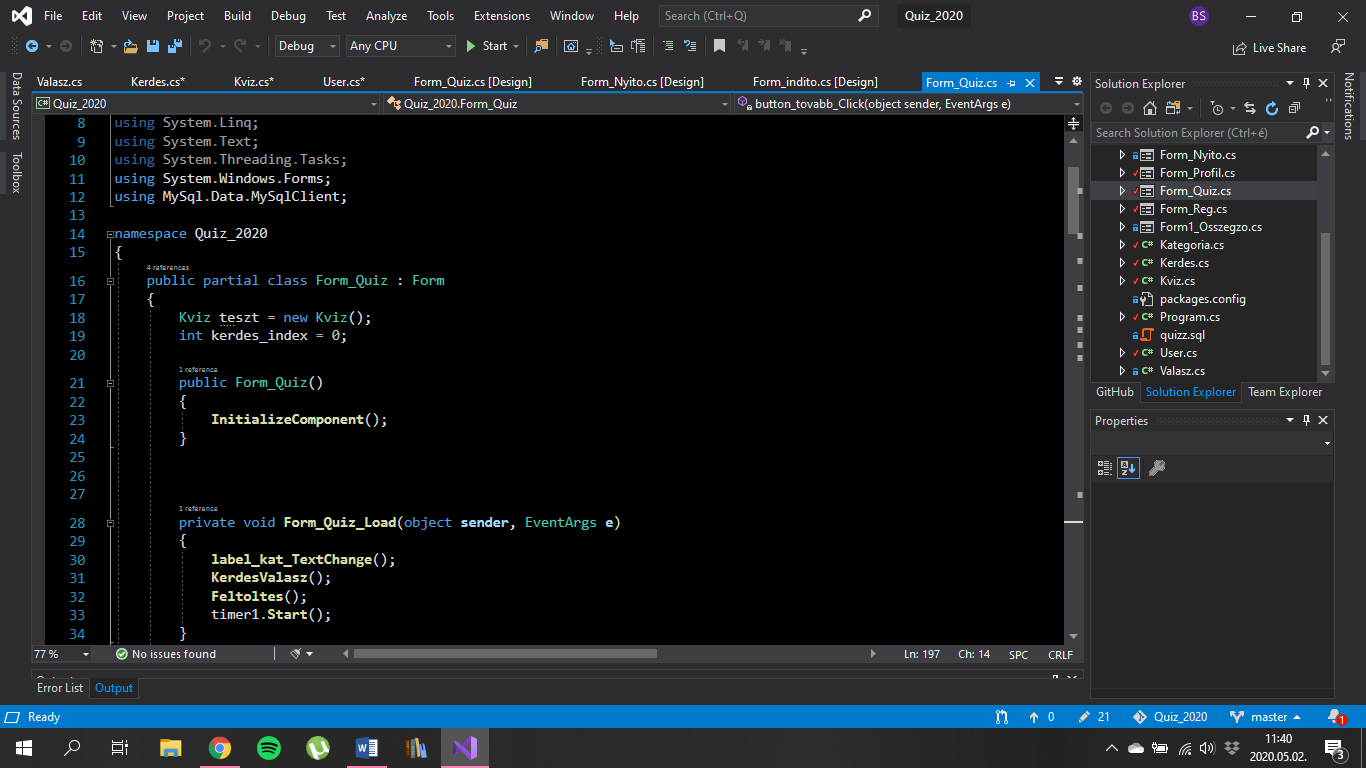


**9. ábra - A választott kategória címkéjéhez tartozó név betöltése, a Kvíz ablakban**

Többek között a fent megírt metódust a *Form\_Quiz\_Load* eljárásban hívtam meg. Ez látható a 19. ábrán. Illetve az időzítő is elindul abban az esetben, ha a maga a Kvíz ablak megjelenik, és a visszaszámlálás megkezdődik, feltöltődnek az egyes szövegdobozok random generált kérdésekkel, melyekhez választási lehetőségeket adtam, illetve megjelenik a választott kategória neve is.

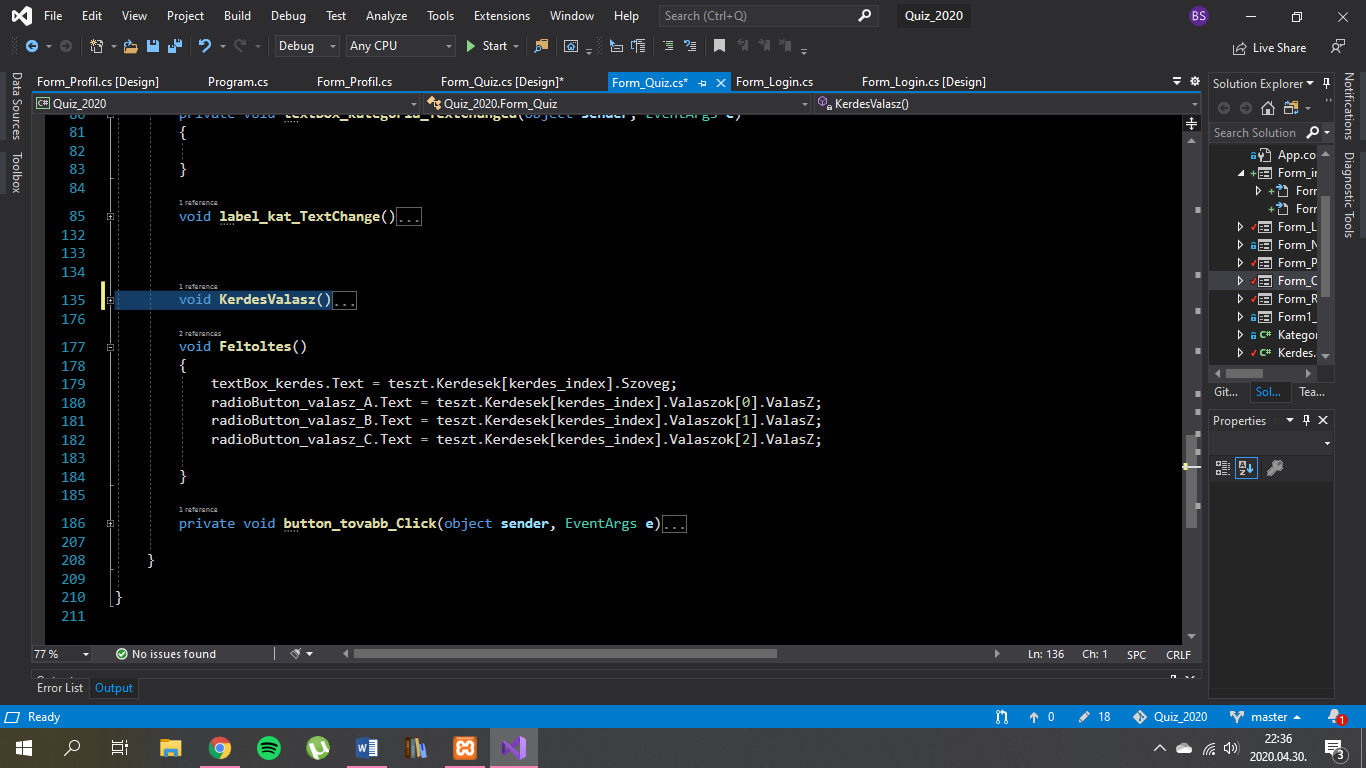


**19. ábra - Függvények meghívása és időzítő elindítása a Kvíz ablak betöltődésénél**



**20. ábra - teszt tömb deklarálása a Kvíz osztály értkeivel**

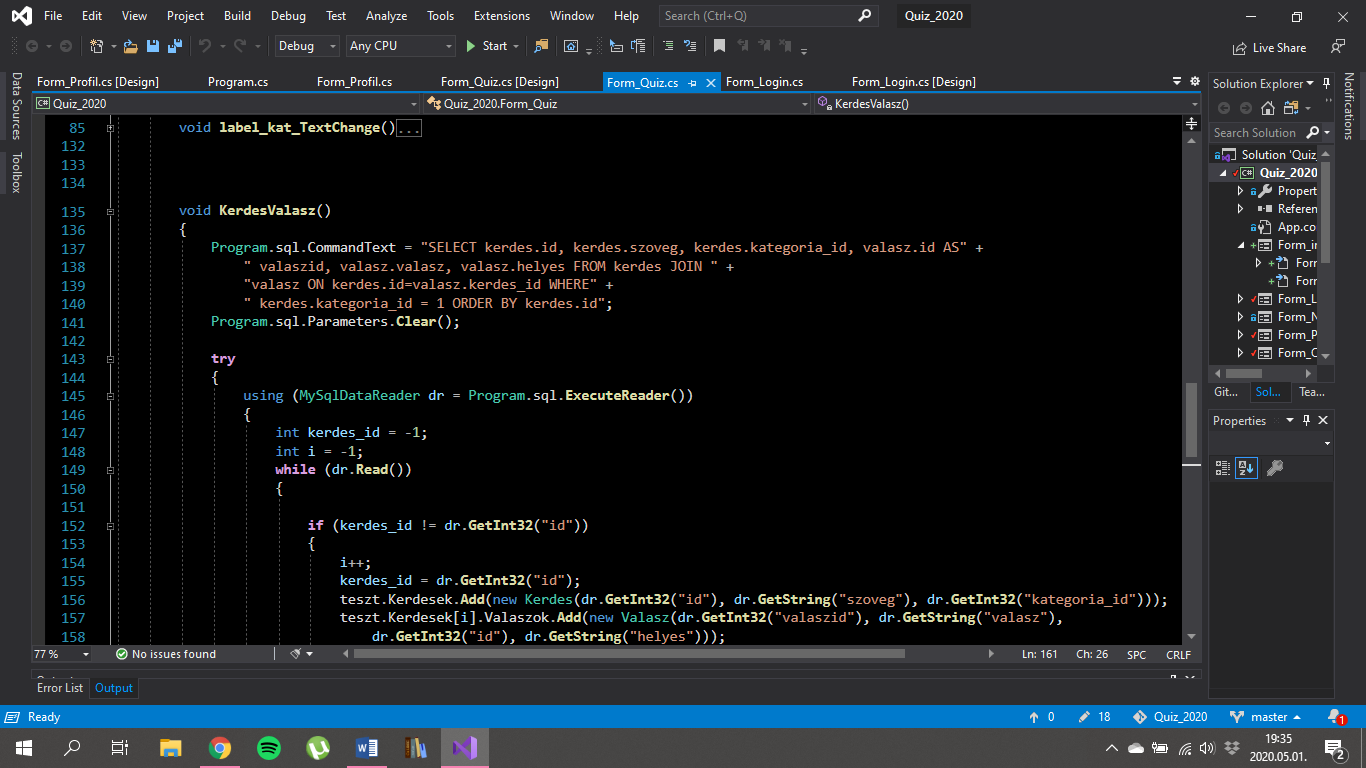
Ezután már csak annyit kellett tennem, hogy a felhasználó számára is láthatóvá tegyem a kérdéseket, ahogy az a 20. ábrán is látható. Ehhez a az ott lévő TextBox-t, és a 3 radioButton-t kellett értékekkel feltölteni. A feltöltés, a Kérdések osztályban létrehozott és definiált *Valasz* listából kerülnek ki,.



**21. ábra - A megjelenítendő kvíz elemek programsorai**

CommandText segítségével történt az adatbázis és a program közötti útvonal létesítése. Azon belül pedig a kérdések, és a válaszoknál megadott *kerdes\_id* –al való összekötés, hogy adott kérdéshez, adott válaszok tartozzanak, melyeket az alább látható (21. ábra) módon hajtottam végre.

*Program.sql.Commandtext* segítségével kérem le a szerverről az adatokat: a kérdéseket, a hozzájuk tartozó válaszokat illetve, hogy azok együttesen melyik kategóriához tartoznak. A kérdések rendezése az azonosítójuk alapján történik (22. ábra).



**22. ábra - Kérdések és lehetséges válaszok összekapcsolása**

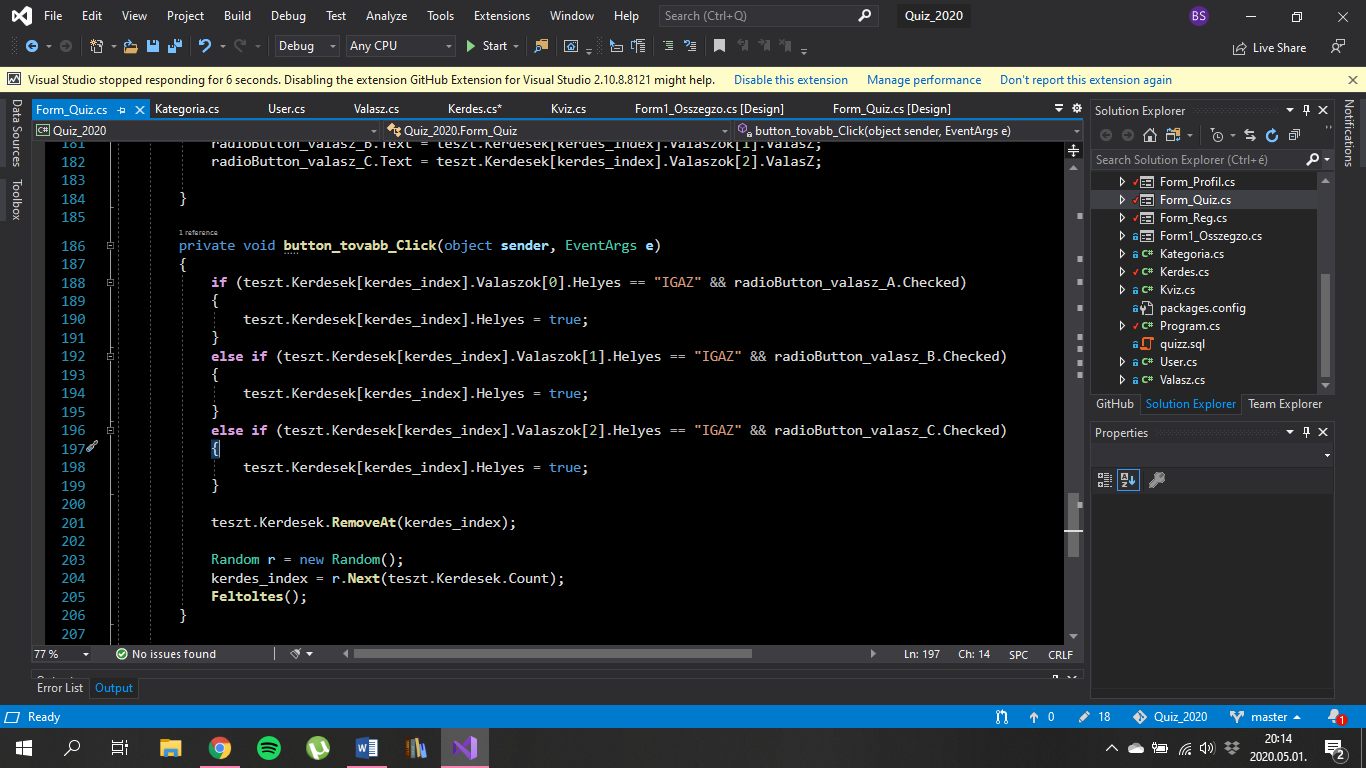
Az eddig még nem derült ki, hogy a feltett kérdésre megjelölt válasz helyes-e vagy sem. Ezt úgy valósítottam meg, hogy egy *Kerdes* osztályt hoztam létre, melyben az adatbázisban található oszlop neveknek megfelelő változókat deklaráltam, illetve egy Valasz nevű listát, melyekben az egyes *valaszok* tároldónak.

Illetve található egy *bool* adatípusú„helyes” nevű változó, melynek vagy igaz, vagy hamis a visszatérési értéke attól függően, hogy az adatbázis ugyanilyen nevezetű oszlopába melyik megjelölés került.

A kvíz ablakban található 2 gomb, a válaszlehetőségeke kívül: Tovább és Kilépés. Amennyiben a felhasználó, ha nem jelöl ki választ, akkor a Tovább gombra kattintva nem történik semmi, így tehát muszáj választania a felkínált lehetőségek közül.

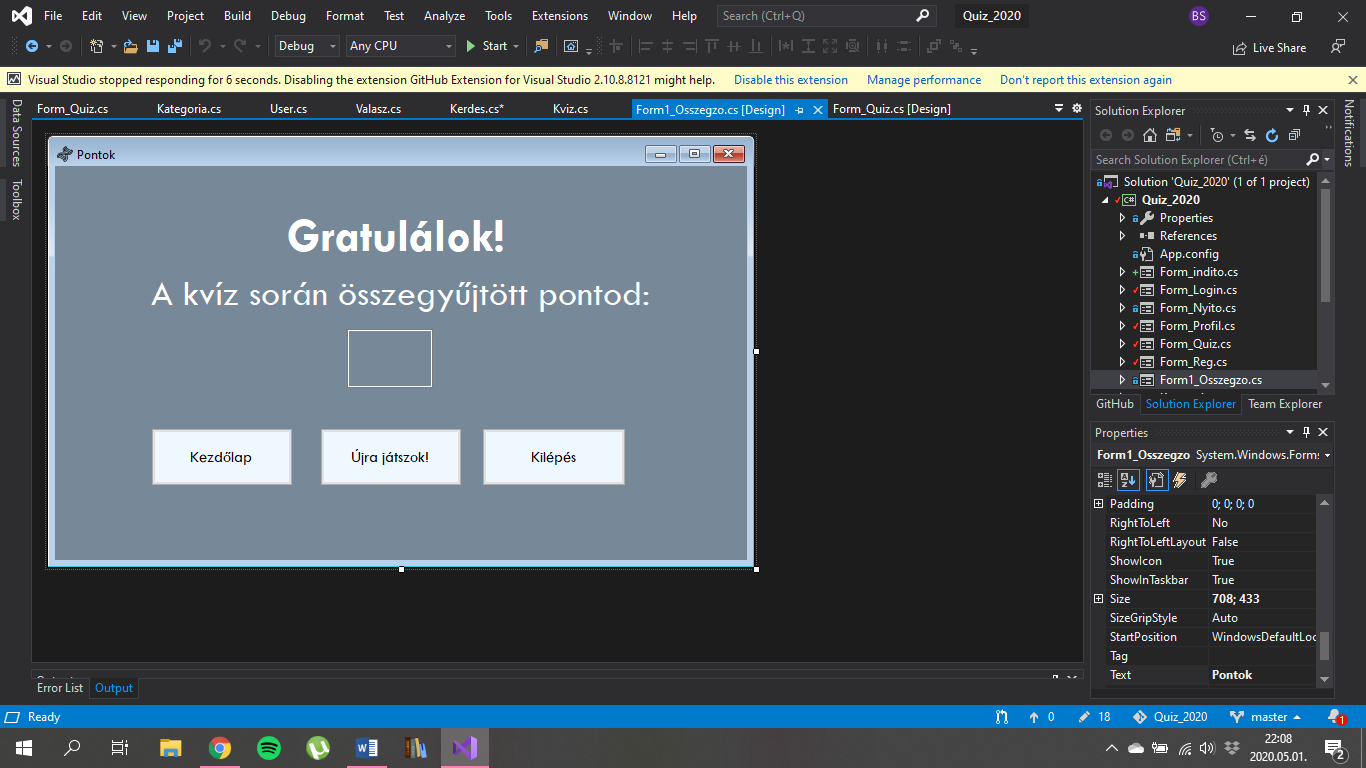
A Tovább gombra kattintás esetén elkezdődik egy feltételvizsgálat, ami azt nézi, hogy a megjelölt radioButton-höz tartozó válasz, helyes volt-e vagy sem. Ez úgy működik, hogy külön minden radioButton-t megvizsgálunk, -ebben az esetben, hogy amit válaszként megjelölt a felhasználó- egyezik-e a válasz listában található helyes változóban eltárolt „IGAZ” értékel. Ha egyezik, akkor a Tovább gomb megnyomása után, a program a felhasználó szerzett pontjaihoz 1 pontot jóváír, amennyiben pedig nem volt helyes a válaszként megjelölt lehetőség, akkor a program, nem növeli a felhasználó összegyűjtött pontszámait (23. ábra).

Minden helyes válasz után a felhasználó 1 pontot kap.



**2310. ábra - Megjelölt válasz helyességének vizsgáltat a Tovább gomb megnyomása esetén**

A *Form\_Osszegzo* feladata, hogy a 30 másodperces időkeret lejárta után automatikusan, kiírja, hogy a felhasználó hány válasza volt helyes és ennek megfelelően, hány pontot sikerült összegyűjtenie, az automatikus Gratuláció címke alatti szövegdobozba. Egy kvíz során megszerezett pontok összege, automatikusan hozzáadódik a profilban tárolt és megjelenített pontszámok alatt (24 .ábra).



**24. ábra - Megszerzett pontok kiírása adott kvíz esetén**

3 gomb is található még található ebben az ablakban. A Kezdőlap gombra kattintva a felhasználót visszaküldi a program a profiljára. Kilépés gomb esetében, a programból lép ki a felhasználó. Az „Újra játszok!” gomb esetében, ugyanaz a kategória marad a kiválasztott, amiben az kvíz történt.

*.*

# **Összefoglalás**

A záródolgozatomban létrehozott alkalmazással az volt a célom, hogy egy önállóan létrehozott programot készítsek a tanulmányaim során tanultak segítségével.

A fejlesztés során voltak dolgok, amiket nem tudtam részletesen prezentálni. A végére, nem minden funkció sikerült úgy, ahogy azt eredetileg elképzeltem, leginkább az időhiány miatt, de a továbbiakban feltétlenül szeretném továbbfejleszteni a programot. Úgy gondolom, hogy ez a projekt fontos tapasztalatnak fog számítani a jövőben, mivel bátrabban fogok tudni belevágni újabb technológiák megtanulásába önállóan is, amit ezen a szakterületen elengedhetetlennek gondolok.

Van néhány elképzelésem még a programot illetően, amivel szeretném, még kiegészíteni az alkalmazást, illetve a kisebb és nagyobb hiba réseket megoldani, komplexebbé tenni a záródolgozatra készített programomat.

Az a véleményem, hogy ez által a munka által alapos betekintést nyerhettem a frontend és a backend világába és, hogy van még hova fejlődni ezen a területen is. A jövőben is az lesz a célom, hogy a fejlesztésben elmélyítsem a tudásomat.

# **Irodalomjegyzék**

[[1](#_C#)] Illés, Z. (2005). *Programozás C# nyelven.* Budapest: Jedlik Oktatási Stúdió.

[2] István, R. (2008). *C#.*

[[3](#_MySQL)] <http://www.sztjg.hu/sql/sql-seged-rg.pdf>