|  |  |
| --- | --- |
|  | Eötvös Loránd Tudományegyetem  Informatikai kar  Média- és Oktatásinformatikai tanszék |

**Egyedi alkalmazás fejlesztése kvízjátékokhoz**

|  |  |
| --- | --- |
| *Témavezető:*  Dr. Illés Zoltán  habilitált egyetemi docens | *Szerző:*  Ömböli Csaba  programtervező informatikus BSc |

*Budapest, 2022*

Tartalomjegyzék

[1. Témabejelentő 4](#_Toc102841890)

[2. Bevezetés 5](#_Toc102841891)

[2.1. A dolgozat tartalma 5](#_Toc102841892)

[2.2. Mindent vagy semmit 5](#_Toc102841893)

[2.2.1. Az eredeti vetélkedő 5](#_Toc102841894)

[2.2.2. Saját szabályok 5](#_Toc102841895)

[3. Felhasználói dokumentáció 8](#_Toc102841896)

[3.1. Rendszerkövetelmények 8](#_Toc102841897)

[3.2. Telepítés 8](#_Toc102841898)

[3.3. A program részei és használata 9](#_Toc102841899)

[3.3.1. Főmenü 9](#_Toc102841900)

[3.3.2. Játékosok 9](#_Toc102841901)

[3.3.3. Kérdéssorok 11](#_Toc102841902)

[3.3.4. Játék 14](#_Toc102841909)

[4. Fejlesztői dokumentáció 23](#_Toc102841910)

[4.1. Főbb tervezési irányvonalak 23](#_Toc102841911)

[4.2. Architekturális döntések 25](#_Toc102841912)

[4.3. Projektfelépítés 27](#_Toc102841913)

[4.4. Indulás, aktiváció 28](#_Toc102841914)

[4.5. Navigáció 29](#_Toc102841915)

[4.5.1. Shell 30](#_Toc102841916)

[4.5.2. PageService 30](#_Toc102841917)

[4.5.3. NavigationViewService 30](#_Toc102841918)

[4.5.4. NavigationService 31](#_Toc102841919)

[4.6. A játék folyamata 31](#_Toc102841920)

[4.7. Adatok bevitele és szerkesztése 31](#_Toc102841921)

[4.8. Kérdéssorok olvasása fájlból 31](#_Toc102841922)

[4.8.1. Formátumok 31](#_Toc102841923)

[4.8.2. QuestionSerieLoader 33](#_Toc102841924)

[4.9. Az Adatelérési réteg 34](#_Toc102841925)

[4.9.1. Adatbázisséma 34](#_Toc102841926)

[4.9.2. Repository és Unit of work 36](#_Toc102841927)

[4.10. Tesztelési terv 37](#_Toc102841928)

[4.10.1. Unit-tesztek 37](#_Toc102841929)

[4.10.2. End to end tesztek 39](#_Toc102841930)

[5. Összegzés 46](#_Toc102841931)

[6. További fejlesztési lehetőségek 46](#_Toc102841932)

[6.1. Kényelem 46](#_Toc102841933)

[6.2. Funkcionalitás 46](#_Toc102841934)

[6.3. Megjelenés 48](#_Toc102841935)

[7. Hivatkozások 48](#_Toc102841936)

[8. Ábrajegyzék 49](#_Toc102841937)

# Témabejelentő



# Bevezetés

## A dolgozat tartalma

A szakdolgozat keretében elkészített program alapjaiban a témabejelentőhöz illeszkedik, de a tervezési, és implementálási fázis során felmerülő új szempontok, körülmények és nehézségek miatt helyenként minimálisan eltérhet attól.

**Mindent vagy semmit program**

A másik fő komponens, egy olyan alkalmazás, ami segítséget nyújt a Mindent vagy semmit televíziós vetélkedőben megismert játékmenet előkészítésében, zökkenőmentes lebonyolításában, és dokumentálásában. Az alkalmazás belekerült a korábban említett indító-alkalmazásba, de önállóan is használható.

A szakdolgozat két alkalmazást tartalmaz, de mivel mindkettőt ugyanolyan tervezési szempontok mentén, nagyon hasonlóan került megvalósításra, ezért a továbbiakban általánosan mindkettőre vonatkozóan írom a tudnivalókat. Ha valamely szempontnál a két alkalmazás eltér, akkor térek ki külön egyesével rájuk.

## Mindent vagy semmit

### Az eredeti vetélkedő

„A vetélkedő 1997-ben indult az akkor újnak számító TV2 képernyőjén, és minden hétköznap az esti műsorsávban jelentkezett. A játék fődíja egy autó volt, ezen kívül pénz- és tárgynyeremények sokasága talált gazdára minden adásban. Az egész az amerikai Joepardy című kvízműsorra épült. Minden adásban három játékos versenyzett, a nap győztese pedig eldönthette, hogy megtartja-e azt a pénzt, amit nyert és kiszáll, vagy visszatér a következő adásra. Aki egy bizonyos számú játékban részt vett az vihette haza a főnyereményt, az autót.” (Vollai, 2019)

A Vágó István nevével fémjelzett műsorban több játékmódban, általános műveltségi kérdésekre válaszolva juthattak közelebb a nyeremények elnyeréséhez a játékosok.

### Saját szabályok

Középiskolai éveim alatt megismerkedtem egy az eredeti vetélkedő szabályaival szinte teljesen megegyező változattal. Az általam elkészített program ennek a variánsnak a szabályrendszerét követi.

Ahhoz, hogy a játékot ebben a formában lehessen játszani, szükség van egy fizikai nyomógomb rendszerre, aminek segítségével eldönthető, hogy ki adhat választ az adott kérdésre.

**A meglévő rendszer technikai részletei**

A jelenlegi rendszer[[1]](#footnote-1) egy hálózati áramforrásra kötött központi egységből, és az ehhez csatlakoztatott négy darab nyomógombból áll. A központi egység biztosítja a gombok az számára az áramellátást, és kezeli az azok lenyomásából származó input jeleket. Ha egy nyomógombot lenyomunk, a gomb előtt található lámpa világítani kezd, a központi egység pedig (logikai áramkörök segítségével) biztosítja, hogy eztkövetően más gombok lenyomására ne világítson a saját lámpájuk, tehát mindig csak az elsőként lenyomott gombhoz tartozó lámpa világítson. A központi egységen található „nullázó” gomb lenyomásával alaphelyzetbe állíthatjuk a rendszert. (Ha van olyan lámpa, ami épp világít, az kialszik, és rendszer ismét inputra vár.)

**A játék menete**

A játékot az eredeti vetélkedőhöz hasonlóan egy műsorvezető/játékmester vezeti. Neki (általában kinyomtatott papíralapú, vagy Word formátumban) rendelkezésére állnak a játékban felhasználásra kerülő kérdéssorok.

A játékosok számától, és az alkalomtól függően lehet egyéni, vagy csapatos játék. Egyéni játék esetén mindenki egyedül áll ki a gombokhoz és versenyez, míg csapatos játéknál az egész csapat egyszerre használja ugyanazt a nyomógombot válaszadási szándékának jelzéséhez. (Az egyszerűség kedvéért a továbbiakban nem különböztetünk meg egyéni játékost, mivel az felfogható egy egyszemélyes csapatként is.) A csapatokat valamilyen szabály szerint[[2]](#footnote-2) körökre osztjuk. A körökre osztásnak fontossági sorrendben a következő preferenciái szoktak lenni:

1. minden csapat legalább n kört játsszon a döntőt nem számolva, (n > 0, n E N)
2. az egy körben szereplő csapatok száma minél egyenletesebb legyen (pl: 9 csapat, és n = 1 esetén rossz megoldás 2 négyfős és 1 egyfős kör kialakítása. Az optimális megoldás 3 háromfős kör.)
3. minél kevesebbszer játsszon egymás ellen két csapat

A körök kialakítása után a csapatok elfoglalják a helyüket egy-egy nyomógomb mögött, és a játék kezdetét veszi.

Egy kör két szakaszból áll. A csapatok feladata mindkét szakaszban az, hogy minél több pontot szerezzenek, a kérdésekre adott helyes válaszaikkal, fontos azonban, hogy a helytelen válaszokért a megválaszolt kérdés pont értékével megegyező levonás jár. (A csapatok akár negatív összpontszámot is elérhetnek.) Az első szakaszban ún. tematikus kérdéssorok kérdéseire lehet válaszolni. Egy kérdéssor 5 témából áll. Minden téma 6 egyre nehezebb kérdést tartalmaz, sorrendben a következő pont értékben: 1000, 2000, 3000, 4000, 5000, 8000. Mindig az előző kérdésre jó választ adó csapat kérheti a következő kérdést, ha ilyen nincs, akkor a legtöbb ponttal rendelkező csapat kérhet. Pont egyenlőség esetén a játékmester dönti el, hogy ki kérhet. A kiválasztott kérdést a játékmester elmondja, a csapatok pedig nyomógombjaik lenyomásával jelezhetik válaszadási szándékukat. (Bármelyik csapat nyomhat, nem csak az amelyik a kérdést kérte.) Mindig az a csapat adhat választ, amelyiknek világít a lámpája.

A második szakaszban „villámkérdésekre” lehet választ adni. Itt a játékmester sorban egymás után tesz fel véletlenszerű témákból kérdéseket. Ebben a szakaszban az összes kérdés 3000 pontot ér. Mivel itt nincsenek témák, amikből választani lehetne, ezért egy válsz elhangzása után a játékmester automatikusan folytatja a következő kérdéssel. Fontos, hogy míg a tematikus szakaszban a játékvezető megengedőbb, azzal kapcsolatban, hogy mennyi idő telik el a gomb lenyomása, és a válasz elhangzása között, addig a villámkérdéseknél a válasznak azonnal kell jönnie, különben a választ helytelennek ítéli meg.

A tematikus és a villámkérdéses szakasz lejátszására is korlátozott idő áll rendelkezésre. A csapatok által az egyes körökben megszerzett pontszám összeadódik, ez képzi majd az összpontszámot.

Az egyes körök végeztével kezdődik a sorban utánuk következő, egészen addig, amíg az összes kört le ne játszották a résztvevők. Ekkor következik a döntő kör, melynek szabályai megegyezni a korábbiakkal. A döntő résztvevői az első 4 legmagasabb összesített pontszámmal rendelkező csapat, akik a döntőben mind 0 pontról indulnak. A döntő győztese nyeri a játékot.

**A jelenleg használt segédprogram**

A játék során használt kérdéssorok megjelenítéséhez, a csapatok pontszámainak kijelzéséhez, és az idő méréséhez jelenleg egy általam ismeretlen személy által, a 2000-es években készített program van használatban. A programot egy ember kezeli, aki egyrészt figyeli a játékosok által kért kérdést, és mindig azt jeleníti meg a képernyőn, másrészt figyeli hogy a játékmester elfogadta-e a választ, és ez alapján adja meg/vonja le a pontokat a csapatoktól.

Az általam megvalósított Mindent vagy semmit program ennek a jól bevált működését veszi alapul, és ezen program hibáiból tanulva kísérel meg az elődjénél jobb felhasználói élményt nyújtani (A részleteket lásd a későbbi fejezetekben.)

# Felhasználói dokumentáció

## Rendszerkövetelmények

Az alkalmazás futtatásához nem szükségesek átlag feletti hardware paraméterek.

Ajánlott paraméterek:

* Operációs rendszer: Windows 10.0.19041.0 vagy újabb
* Processzor: x64-es architektúrájú processzor
* Memória: 4 GB RAM
* Szabad tárhely: ??? byte
* Videókártya: CPU-ba integrált, vagy jobb

## Telepítés

Az alkalmazás telepítéséhez egy biztonsági tanúsítvány szükséges. Ezt a mappában található .cer kiterjesztésű fáljra duplán kattintva tehető meg. Az így megjelenő ablakban nyomjunk a „Tanúsítvány telepítése…” gombra, majd válasszuk ki a „Helyi számítógép” opciót. A következő képernyőn válasszuk a „Minden tanúsítvány tárolása ebben a tárolóban” lehetőséget, majd a „Tallózás” gombra kattintás után válasszuk a „Megbízható legfelső szintű hitelesítésszolgáltatók” tárolót.

A biztonsági tanúsítvány telepítése után a .msix kiterjesztésű telepítőfájlt elindítva, és az utasításkat követve installálható maga az alkalmazás. A telepített alkalmazás megjelenik a start menüben.

## A program részei és használata

### Főmenü

Az alkalmazást mindig a következőképpen néz ki: A tetjén található egy navigációs sáv, ez alatt pedig az aktuálisan betöltött oldal. A navigációs sáv sosem változtatja a pozícióját, viszont az alatta megjelenített oldalak igen. A sáv célja épp az, hogy jelezze a felhasználónak, hogy jelenleg melyik oldalon jár, illetve hogy az abban megjelenő elemekre kattintva váltani tudjon az oldalak között.



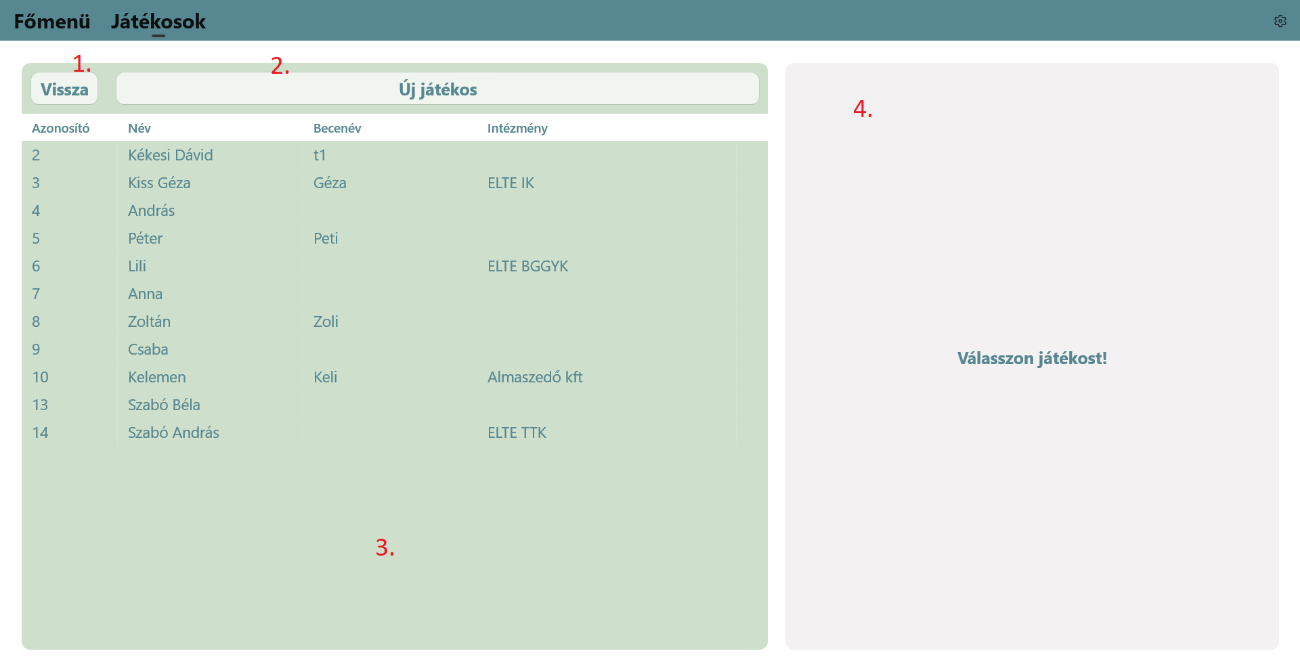
1. ábra, A főmenü

Az alkalmazást elindítva az első oldal amit a felhasználó meglát, az a főmenü (KÉP) Az itt található három gomb lenyomásával indíthatunk új játékot, kezelhetjük a kérdéssorokat, vagy a játékosokat.

### Játékosok

Az alkalmazás saját lehetőséget nyújt játékosok bevitelére, mellyel nagyban megkönnyithetőek az egyes játékok lebonyolítása. Az így bevitt játékosok saját adatbázisban kerülnek tárolásra, de személyes vagy bizalmas jellegű adatokat senkinek sem kell megadnia.

A fűmenüben a „Játékosok” gombra kattintva, vagy ha az oldal már korábban megnyitásra került, akkor a navigációs sávban a „Játékosok” elemre kattintva, megjelenik a játékosok bevitelére, szerkesztésére és törlésére használt oldal



2. ábra, A játékos nyilvántartó oldal

Az oldal részei (az 1. ábra számozásának megfelelően):

1. Ezzel a gombbal térhetünk vissza a főmenübe. Figyelem! Ha a felhasználó nem menti az adatait mielőtt visszalép a főmenübe, az összes nem mentett adat elveszik!
2. Ezzel a gombbal adhatunk hozzá új játékost. Lenyomására az jobb oldali szerkesztő felületen adhatjuk meg az új játékos adatait.
3. Ebben a listában jelenik meg az adatbázisban aktuálisan elérhető összes játékos. Ha valamelyiket kiválasztjuk a jobb oldali szerkesztő felület elérhetővé válik, és szerkeszthetjük a játékos adatait.
4. Egy játékos kiválasztása, vagy új játékos hozzáadása esetén itt válik elérhetővé a szerkesztő felület.

Egy játékost kijelölve, vagy egy újat hozzáadva jobb oldalon megjelenik az adatok szekesztésére szolgáló felület. (2. ábra)



3. ábra, A játékos szerkesztő felület

A szerkesztő felület részei (a 2. ábra számozásának megfelelően):

1. A játékos egyedi azonosítója. Nem szerkeszthető. Új játékos esetén automatikusan -1-et mutat, majd mentés után a rendszer állítja be egy még nem foglalt értékre.
2. Ebben a mezőben lehet szerkeszteni a játékos nevét. Tetszőleges név megadható.
3. Itt adható meg, hogy a játékos milyen becenevet választ magának. A későbbi játékok során alapértelmezetten a becenév lesz kijelezve. Ha egy játékoshoz nem tartozik becenév, a név kerül jelzésre.
4. Ide kell írni a játékos intézményét, cégét, iskoláját, vagy bármit aminek/akinek a színeiben a játékban indul. (Bármilyen karakter megengedett)
5. Ezen gomb lenyomásával menthetőek a változtatások az aktuálisan szerkesztett játékoson
6. Ezen gomb lenyomásával törölhető az aktuálisan szerkesztett játékos a program adatbázisából.

### Kérdéssorok

Az alkalmazásban lehetőség van kérdéssorok eltárolására is, így egy kényelmesebb módot adva arra, hogy minden kérdéssor egy helyen legyen, és könnyen betölthetőek legyenek a játékba.

A kérdéssorokat kezelő oldalra a főmenü „Kérdéssorok” gombjára kattintással juthat el a felhasználó. Ha már meg volt nyitva az oldal, akkor a navigációs sáv „Kérdéssorok” elemére kattintva is ez az oldal kerül betöltésre. (KÉP számokkal)

Az oldal részei:

* Ebben a listában jelennek meg az adatbázisban aktuálisan elérhető kérdéssorok. Egy kérdéssort kiválasztva az megjelenik a jobb oldali szerkesztő felületen
* Ezzel a gombbal hozható létre új kérdéssor. 

4. ábra, A kérdéssorok szerkesztőfelülete

A jobb oldalon fellelhető szerkesztő felület lehetőség nyújt egy teljes kérdéssor bevitelére. Fent a téma címének megadása után lejjebb görgetve vihetőek be az egyes témák adatai. A program csak 5 témából, és témánként hat kérdésből álló kérdéssorok bevitelét támogatja.

A szerkesztő felület részei (a 4. ábra számozásának megfelelően):

1. Ebben a mezőben szerkeszthető az aktuálisan kiválasztott kérdéssor címe. Új kérdéssor létrehozásakor automatikusan az „Új kérdéssor” felirat jelenik meg benne.
2. Itt adható meg az éppen szerkesztés alatt álló téma címe. Témacím választáskor törekedni kell a minél tömörebb megfogalmazásra. Ha egyéb hozzáfűzni valója akad a felhasználónak a témához, akkor azt a leírás mezőbe fejtheti ki.
3. Ide szúrható be az aktuális téma leírás. Itt hosszabban is megfogalmazható, hogy mit is takar a téma címe.
4. Ezen panel segítségével állíthatók be a témát jellemző ismeretkörök
5. A témához tartozó kérdések itt állíthatóak be
   1. Az egyes kérdések szövege itt állítható be
   2. A kérdésre elvárt helyes válasz helye
   3. A kérdés helyes megválaszolásáért járó pont értéke. Érdemes 0-nál nagyobb számot megadni, úgy hogy nagyságrendileg illeszkedjen a többi kérdés pont értékéhez.
6. Erre a gombra kattintva menthető el az aktuálisan szerkesztés alatt álló kérdéssor.
7. Ezzel a gombbal törölhetjük a szerkesztés alatt álló kérdéssort



A program lehetőség nyújt arra, hogy a felhasználó ismeretköröket rendeljen az egyes témákhoz, így is elősegítve az egyes témák illetve kérdéssorok átláthatóságát. A témákhoz rendelt ismeretkörök kérdéssor választáskor hasznos információt szolgáltatnak az adott témáról. Egy témánál a rendelkezésre állók közül bármennyi, és bármely ismeretkör kiválasztható. Az alkalmazásban alapból elérhető 12 téma közül lehet választani:

* Biológia
* Kémia,
* Sport
* Szórakozás
* Életmód
* Művészet
* Mindennapok
* Földrajz
* Történelem
* Irodalom
* Matematika
* Fizika
* Technológia

Az ismeretkör-választó panel részei (az 5. ábra számozásának megfelelően):

1. Ebbe a keresősávba kezdheti el begépelni a felhasználó a keresett ismeretkör nevét, és a program egy legördülő lista formájában automatikusan felajánlja a keresett szöveggel legpontosabban egyező nevű témakört.
2. Itt jelennek meg a fent említett legördülő lista segítségével a témához rendelt ismeretkörök.
3. Minden ismeretkör mellett található egy X gomb, amire kattintva az ismeretkör eltávolításra kerül az adott témából

Kötelező mezők



### Játék

A vetélkedő kapcsán a program megkülönböztet játékot és kört. A kör egy olyan fordulót jelent, amelyben maximum 4 csapat küzd egymás ellen tematikus és/vagy villám kérdésekre válaszolva. A játék több kör összeségét jelenti.

A felhasználónak azt kell megadnia, hogy a csapatok egyenként pontosan hány kört játsszanak[[3]](#footnote-3). Ez nem feltétlenül az összes kör számát jelenti, az a csapatok számától is függ.

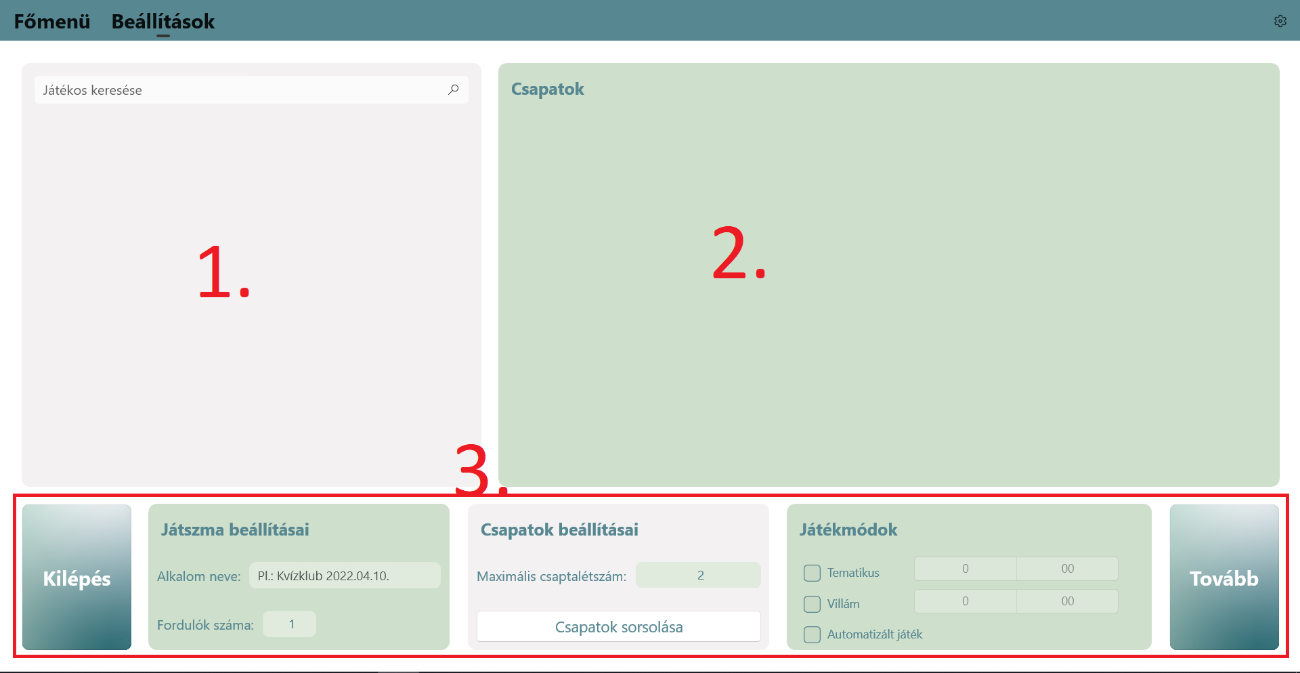
A körökbe több játékmód is választható:

* tematikus játék
* villámkérdéses játék
* automatizált játék

Ha az összes előre kisorsolt kör lejátszásra került, az utolsó kör következik, a döntő. A döntőben azok a csapatok vehetnek részt, akik a körök során megszerzett pontjaik alapján az első 4 helyen végeztek. A döntőben a csapatok 0 pontról indulnak, és ennek győztese lesz a játék végső győztese. A döntő vége után a főmenüből új játék indítható.

**A játék beállításai**

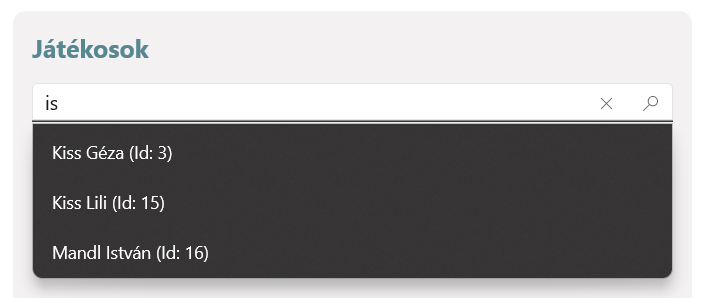
A főmenüben az „Új játék” gombra kattintva az új játék beállításait listázó oldalra navigál az alkalmazás. (6. ábra) Az itt megadott részletek az összes körre[[4]](#footnote-4) vonatkoznak, de egyesével módosíthatók minden kör elindítása előtt.



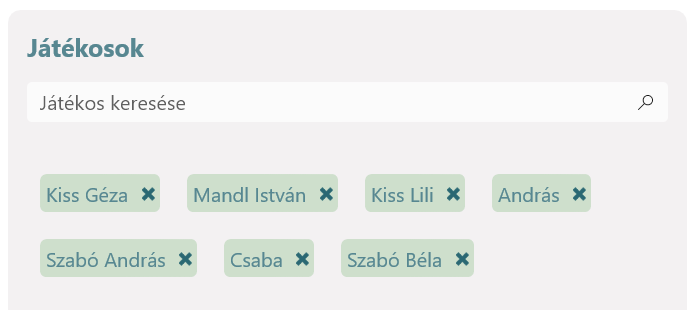
5. ábra, Az új játék beállításai

Az új játék beállításai (az 5. ábra számozásának megfelelően):

1. Játékos hozzáadó felület.

A keresősávba történő gépelés után a program feljaánlja a nevükben legpontosabb egyezést mutató játékosokat egy legördülő lista formájában (7. ábra). Csak azok a játékosok jelennek meg a listában, akiket korábban hozzáadtak az adatbázishoz. A kívánt játékosra kattintva az hozzáadásra kerül a keresősáv alatti területhez. Az újonnan létrehozott játékban az itt szereplő a játékosokból alkothatók csapatok.  
A játékosok neve melletti X-re kattintva eltávolíthatóak a panelről (8. ábra).  


6. ábra, A játékos nevére keresve megjelennek a nevükben legpontosabb egyezést mutató játékosok



7. ábra, A kiválasztott játékosok a keresősáv alatti területen jelennek meg.

1. Csapatok

A játék során csapatok játszanak egymás ellen, melyek a beállítás során itt jelennek meg, és itt módosíthatóak. Alul a csapattagok nevei, felül pedig a csapatnév látható. A csapattagokat drag & drop segítségével át lehet helyezni egyik csapatból a másikba. (10. ábra) Csapatnévnek alapértelmezetten a csapattagok megadott becenevéből (vagy ha nincs becenév megadva, akkor a rendes nevükből) alkot egy csapatnevet, amit a felhasználó tetszőleges névre módosíthat. (9. ábra)



8. ábra, A csapatok panelekbe rendeződve jelennek meg, tetszőleges csapatnév is megadható



9. ábra, A csapattagok drag & drop segítségével áthelyezhetőek

1. Egyéb játékbeállítások

A játék további beállításai (a 11. ábra számozásának megfelelően):

* 1. Erre a gombra kattintva a program visszalép a főmenübe. Figyelem! A kilépéssel az összes módosítás elveszik!
  2. Itt nyílik lehetőség a játék nevének (pl.: verseny aminek alkalmából a vetélkedő működik) beállítására.
  3. Itt adható meg, hogy az egyes csapatok (a döntőt nem számítva) pontosan hány kört játsszanak.
  4. Itt állítható be, hogy maximum hány tagja lehessen egy csapatnak. (Egyéni játék esetén 1-et kell beállítani)
  5. Erre a gomra nyomva a játékosválasztó panelre kiválasztott játékosokból, illetve a maximális csapatlétszámból az alkalmazás véletlenszerűen csapatokat generál, amik a „Csapatok” régióban jelennek meg.
  6. A jelölőnégyzet bepipálásával adhatjuk hozzá a tematikus játékmódot a játékhoz. A mellette szereplő panelen kiválaszthatjuk, hogy mennyi idő álljon rendelkezésre a játékmód lejátszására
  7. A jelölőnégyzet bepipálásával adhatjuk hozzá a villámkérdés játékmódot a játékhoz. A mellette szereplő panelen kiválaszthatjuk, hogy mennyi idő álljon rendelkezésre a játékmód lejátszására
  8. A jelölőnégyzetet bepipálva automatizált játékot indíthatunk (Ez nem használható a villámkérdésekkel együtt.)
  9. Ezt a gombot lenyomva léphetünk tovább a körök beállításaihoz



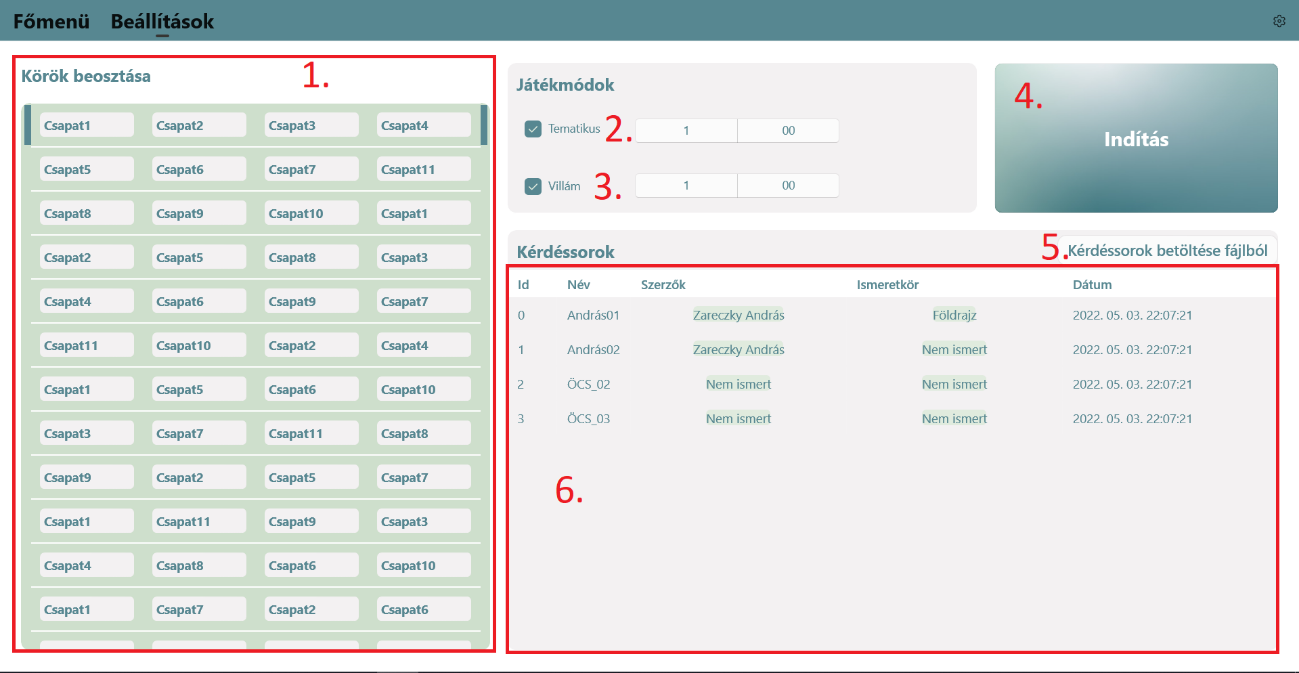
10. ábra, egyéb játékbeállítások

**Kör beállításai, és indítása**

A játék beállításai oldalról tovább lépve a specifikusan körökre vonatkozó beállításokat adhatjuk meg. Ki kell választanunk, hogy mely kör szeretnénk következőnek indítani, és ha tematikus vagy automatizált játékot játszunk, akkor a körben szereplő kérdéssort is ki kell választanunk.

Az oldal részei (a 12. ábra számozásának megfelelően):

1. Itt látható a csapatok egymás elleni sorsolása. Minden sorban az adott körben egymás ellen játszó csapatok láthatóak. A már lejátszott körök szürkén jelennek meg. Az itt kiválasztott kört fogja indítani a program az „Indítás” gombra kattintva.
2. A kiválasztott körre vonatkozó tematikus játékmód beállításai. Teljesen hasonló módon működik, mint a játékra vonatkozó beállítás esetében.
3. A kiválasztott körre vonatkozó villám játékmód beállításai. Teljesen hasonló módon működik, mint a játékra vonatkozó beállítás esetében.
4. Erre a gombra kattintva indíthatjuk el a kiválasztott kört.
5. Ezzel a gombbal tölthetünk be olyan kérdéseket, amelyek nem az adatbázisból származnak. A fájloknak a „Kérdéssorok” mappában kell lenniük, és a két megengedett formátum egyikének megfelelően kell őket fájlon belül tagolni.
6. Itt jelennek meg a rendelkezésre álló kérdéssorok. (Fájlból történő betöltés után a fájlból betöltöttek is.) Tematikus játékmód esetén itt lehet kiválasztani a kiválasztott körhöz a felhasználandó kérdéssort.



11. ábra, A körökre vonatkozó beállítások oldala

Mivel régi típusú kérdéssorokkal nem indítható automatizált játék, ezért amennyibe ilyen játékmóddal kerül indításra a játék, a kérdéssor választó felülethez hozzáadásra kerül egy oszlop, ami jelzi, hogy az adott kérdéssor alkalmas-e a játékra. (13. ábra)



12. ábra, Kérdéssor választó felület automatizált játék indítása után

**Tematikus játékmód**

A tematikus játékmód során a játék menete a Bevezető c. fejezet, Saját szabályok részében leírtaknak megfelelően zajlik.

A kör beállításai oldalon egy kört és hozzá egy kérdéssort kiválasztva a tematikus oldalra navigálhatunk.

Az oldal részei a (14. ábra számozásának megfelelően):

1. Itt jelenik meg a beállításoknál megadott játéknév
2. Ez a mező jelzi a hátralévő időt. Rákattintva elindítható, ismét rákattintva megállítható a visszaszámlálás.
3. Ezen a területen jelenik meg a kiválasztott kérdéssor. A gombrács felső gombjain olvashatóak a témák címei, rájuk kattintva megjelenik a témához tartozó leírás.  
   A gombrács szürke gombjain a kérdések pont értékei olvashatóak. Egy gombra kattintva a kérdés szövege megjelenik az alsó panelen.
4. Ezen a panelen jelenik meg a kiválasztott kérdés szövege
5. Ezen a területen követhetőek nyomon a csapatok által a körben megszerzett pontok. Minden csapathoz egy külön mező tartozik, felül a csapatnévvel, alul a pontszámmal. A programot kezelő személy a + illetve a – gombokra kattintással adhatja meg a pontot a válaszadó csapatnak, illetve vonhatja le azt tőle.
6. Erre a gombra történő kattintással tüntethető el a kérdés szövege, ha nem érkezett rá válasz.
7. Amennyiben a villám játékmód kiválasztásra került, erre a gombra kattintva válthatunk át arra a játékmódra.
8. Ha valamilyen oknál fogva előbb meg szeretnénk szakítani a játékot, erre a gombra nyomva véget vethetünk a körnek az aktuális állásnál.



13. ábra, A tematikus játékmód felülete

**Villámkérdések**

A villám játékmód szintén a bevezetőben leírtaknak megfelelően zajlik. A változás a tematikus oldalhoz képest az, hogy nem lehet kérdést választani, és a + illetve – gombokra történő kattintás fixen 3000, illetve -3000 pontos változást idéz elő.

Mivel ennek a játékmódnak a lényege a fizikai nyomógomb mihamarabbi lenyomása, ezért nem játszható automatizált játékban.

**Automatizált játék**

Az automatizált játék lényege, hogy a játékosok akkor is tudjanak játszani, ha nem áll rendelkezésre a fizikai nyomógomb rendszer. Ebben az esetben egy tematikus kérdéssorral játszhatnak a csapatok. A játék addig tart, amíg el nem fogy az összes kérdés, tehát megszabott időkeret nincs. A csapatok felváltva válaszolhatnak egy általuk kiválasztott kérdésre, de lehetőségük van passzolni is.

**A képen szöveg, képernyőkép, beltéri látható

Automatikusan generált leírás**

14. ábra, Az automatizált játék felülete

Az automatizált játék felületének részei (a 16. ábra számozásának megfelelően):

15. ábra, Az automatizált játék új elemei a tematikushoz képest

1. A korábbi válaszok nyomán szerzett vagy elbukott pontok itt követhetőek nyomon. (Az legutóbbi bejegyzés legalul helyezkedik el.
2. A válaszadó csapat csapatneve. Amelyik csapat neve itt szerepel, az adhat választhatja ki, és az adhat választ a kérdésre.
3. Ebbe a mezőbe kell beírnia a csapatnak a kérdésre adott válaszát.
4. Erre a gombra kattintva véglegesítheti a válaszát a csapat. A megnyomása után megjelenik, hogy a válasz helyes volt vagy nem, és ennek megfelelően alakul a csapat pontszáma is.
5. Egy kérdés kiválasztása után ezen gomb megnyomásával passzolhat a csapat.

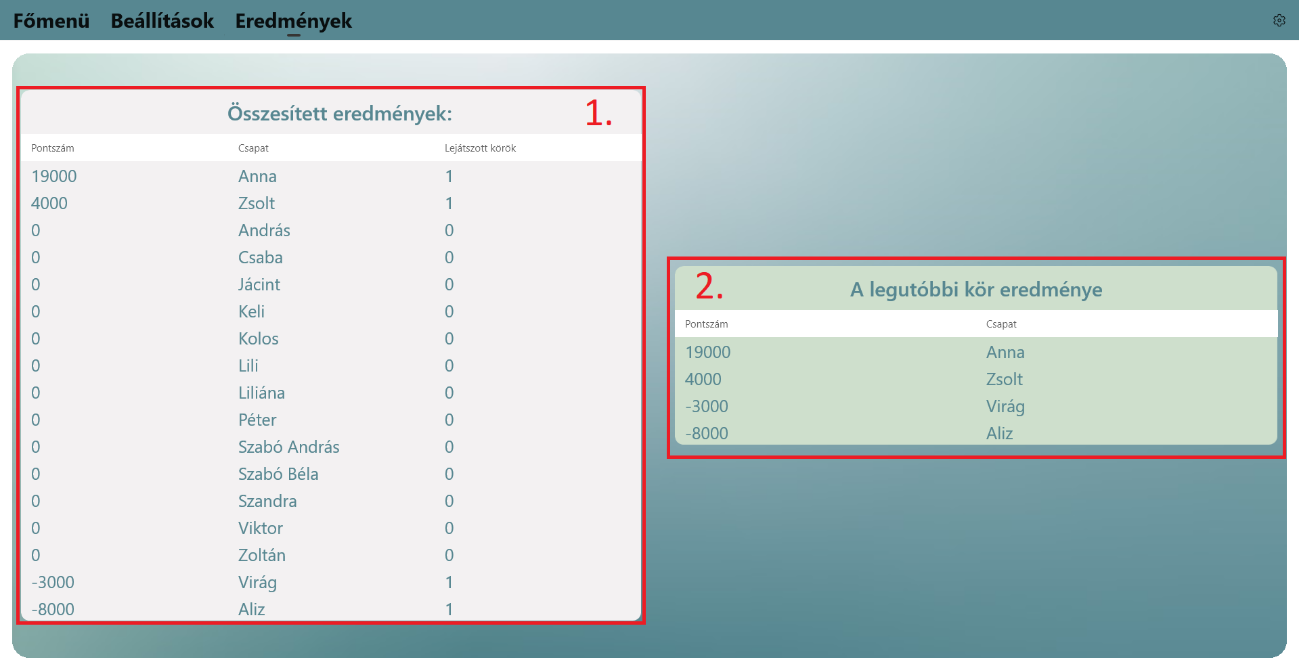
A felület többi része teljesen hasonló módon működik, mint a tematikus felület esetében.

**Eredmények**

A játék során szerzett pontok nyomon követéséhez az egyes körök végén megjelenik az eredményeket jelző oldal. Ez az oldal szolgál tájékoztatással a jelenlegi összesített állásról, az utolsó kör eredményeiről és a döntő lejátszása után a döntő eredményeiről.

Az eredmények oldal részei (a 17. ábra számozásának megfelelően):

1. Ebben a táblázatban szerepel az összes csapat eddigi körökben megszerzett pontjainak összege.
2. Ez a táblázat mutatja a legutóbbi kör eredményét. A döntőben megszerzett pontszámok nem adódnak hozzá az összesített eredménykehez.



# Fejlesztői dokumentáció

## Főbb tervezési irányvonalak

Középiskolás koromtól kezdve, amikor barátaimmal még heti szinten aktívan használtam a kvízes rendszert (kiváltképp a programot), egészen mostanáig megvan bennem a szándék, hogy legyen egy olyan alkalmazás, ami jobban megfelel az eredeti program készítése óta eltelt évek támasztotta igényeknek. Nekem is volt sok ötletem ezzel kapcsolatban, de hogy a tényleges felhasználói igényeket felmérve tudjak nekiállni a tervezésnek, egy ehhez kapcsolódó kérdőív szétküldésével gyűjtöttem visszajelzéseket. Nem érkezett be annyi válasz, amennyire számítottam, de úgy vélem a felmérés, még így is reprezentatív.

Mivel a készítendő alkalmazásnál a funkciót tartottam elsődleges fontosságúnak, ezért az 1. és 2. ábrán látható visszajelzések fontos útmutatásként szolgáltak, hogy milyen mélységben kell „hozzányúlnom”, vagyis módosítanom, esetleg kivennem, az eddig használt funkciókhoz.



16. ábra A régi program funkcionalitására irányuló kérdésre kapott válaszok aránya

A képen szöveg látható

Automatikusan generált leírás

17. ábra A régi program esetleges haszontalan funkcióiról érdeklődő kérdés, és az erre kapott válaszok.

A meglévő funkciók átdolgozására, és újak hozzáadására igen sok javaslat érkezett, amelyek együttes megvalósítása (néhány általam kitalált funkcióval együtt) túlmutat a szakdolgozat keretein, ám többségük a jövőben vélhetően megvalósításra kerül. (Lásd: További fejlesztési lehetőségek c. fejezet.)

A funkció után a megjelenés volt számomra a következő prioritás. Véleményem szerint lehet egy alkalmazás bármennyire fejlett, és funkcionálisan tökéletes, ha nem frissítik a megjelenését, az évenkét változó UI/UX igények és trendek fokozatosan elavulttá teszik a felhasználók szemében. Bár a meglévő program elég egyedi a maga műfajában, de korábbi állításomat a 3. ábrán látható eredmény igazolni látszik.



18. ábra A régi program kinézetét érintő kérdés

Míg az előző két szempont talán felhasználóként a leglényegesebb, addig fejlesztőként a már a témabejelentőben is szereplő bővíthetőség kritikusan fontos. Mivel az alkalmazást a jövőben több területen tervezem továbbfejleszteni, ezért értékesebbnek tartok egy átláthatóan, és koherensen megírtat, egy extra funkciókkal telerakott, és/vagy legelkápráztatóbb UI megoldásokat használó, de kevésbé fejlesztő barát alkalmazásnál.

Összegezve: A fejlesztés során a prioritások sorrendben a következőként alakulnak:

1. Bővíthetőség, átláthatóság
2. Funkcionalitás
3. Kinézet

## Architekturális döntések

Az alkalmazás egy asztali alkalmazás. 2022-ben valóban szokatlan lehet egy dedikáltan asztali alkalmazást készíteni, ám ebben az esetben vannak olyan körülmények, amik ezt indokolttá teszik. A közösségünkben meglévő régi program (aminek az utódjaként ez készült) igénybevételeire, jelentős hányadban olyan környezetben kerül sor, ahol nincs, vagy csak akadozó internetkapcsolat áll rendelkezésre. Ezen okból kizárható a kizárólag online működtetett alkalmazás. A fennmaradó offline platformok közül a PC-s megvalósításon kívül minden más irreleváns lett volna.

A következő fontos döntés, a környezet kiválasztása volt. Mivel Windowson kívül más operációs rendszerre megvalósítani nem lett volna értelme, ezért kézenfekvő választás volt a .NET keretrendszer egy Windows specifikus szegmense.

A választott környezet eleinte az UWP[[5]](#footnote-5) volt, de idő közben a Microsoft kiadta az új MAUI[[6]](#footnote-6) rendszerbe illeszkedő WinUI 3[[7]](#footnote-7) (és ezzel együtt a Windows App SDK[[8]](#footnote-8)) első stabil verzióját. Ez az újszerű platform a korábban lefektetett szempontok közül a funkcionális helyességet nem befolyásolja, de a másik kettőben komoly előrelépés figelhető meg az UWP-hez képest. Általában elmondható, hogy egy újabb rendszer lévén jobban teljesíti az egyszerű bővíthetőséggel és fenntarthatósággal szemben támasztott követelményeinket, ám ezt csak fokozza, hogy a Microsoft megszűntette az UWP platform támogatottságát, a Windows App SDK javára, valamint hogy a Windows 11 operációs rendszer későbbi verzióiban várhatóan nem fognak futni az UWP-ra írt alkalmazások. A WinUI 3 tovább bővíti majd az előd amúgysem szegényes eszköztárát a felhasználói felület megvalósításához.

Ezek alapján az alkalmazás a következők szerint került megvalósításra:

* Windows App SDK-t és WinUI 3-at használ
* Szerkezete a platformhoz illő MVVM architektúrát követ
* A model és view-model részek C# nyelven, a view pedig XAML leíró nyelven kerültek megvalósításra
* Az alkalmazás .NET 5 és C# 9 szabványokat támogat

Az alkalmazás a Visual Studio 2019 fejlesztői környezeteben készült, és a további fejlesztői tevékenység zökkenőmentes folytatásához is ez javasolt. Régebbi és újabb verziók használata az esetlegesen fellépő átmeneti problémák miatt lassíthatja a fejlesztési folyamatot.

A projekt létrehozásakor a Visual Studioba kiegészítőként telepíthető Windows Template Studio-t vettem igénybe, amely néhány alapvető funkcióhoz szükséges sablonnal látta el az újonnan létrehozott alkalmazást. (Mivel a projektet eredetileg UWP projektként hoztam létre és csak később döntöttem a migrálásáról, ezért az akkor készült sablonok sem feltétlenül egyeznek meg egy újonnan generálttal.)

Az említett keretrendszert használva adja magát, hogy MVVM (Model-View-ViewModel) architektúra használata a legkézenfekvőbb. Az alkalmazásban minden megjelentett oldalhoz (Page) egy ViewModel osztály tartozik. Az oldalak fájljai a Page, a ViewModellek fájljai pediga ViewModel posztfixet kapják.

## Projektfelépítés

A forrsákód az egyes elemek logikai felépítésének szétválaszthatósága, és a könnyebb áttekinthetőség érdekében 3 projectbe rendeződik:

**AllOrNothing**

Ez az alkalmazás fő projektje, itt történik a legtöbb lényeges dolog. Ebben található meg a megjelenítéstért felelős réteg, az üzleti logika.

Felhasznált NuGet csomagok:

* AutoMapper (11.0.1)
* CommunityToolkit.Mvvm (7.1.2)
* CommunityToolkit.Mvvm.WinUI.Animations (7.1.2)
* CommunityToolkit.WinUi.Controls (7.1.2)
* H.OxyPlot.WinUI (0.9.19)
* Microsoft.EntityFrameWorkCore.Sqlite (5.0.16)
* Microsoft.Extensions.DependencyInjection (6.0.0-preview.5.21301.5)
* Microsoft.Extensions.Hosting.Abstractions (6.0.0)
* Microsoft.Windows.SDK.BuildTools (10.0.2200)
* Microsoft.WindowsAppSDK (1.0.0)
* Microsoft.Xaml.Behaviors.WinUI.Managed (2.0.8)
* Newtonsoft.Json (13.0.1)
* Serilog.Extensions.Logging.File (3.0.0-dev-00067)

**AllOrNothing.Repository**

Ez az elkalmazás adatelérési rétege. Itt kerül megvalóósításra a Repository és a UnitOfWork programtervezési minta, illetve itt található az EntityFramework működéséhez szükséges fájlok (migrációs fájlok, data-context fájlok stb…), az adatmodellek kivételével. (lsd.: Adatelérési réteg c. fejezet)

Felhasznált NuGet csomagok:

* Microsoft.EntityFrameworkCore.Design (5.0.16)
* Microsoft.EntityFrameworkCore.Sqlite (5.0.16)
* Microsoft.EntityFrameworkCore.Tools (5.0.16)

**AllOrNothing.Data**

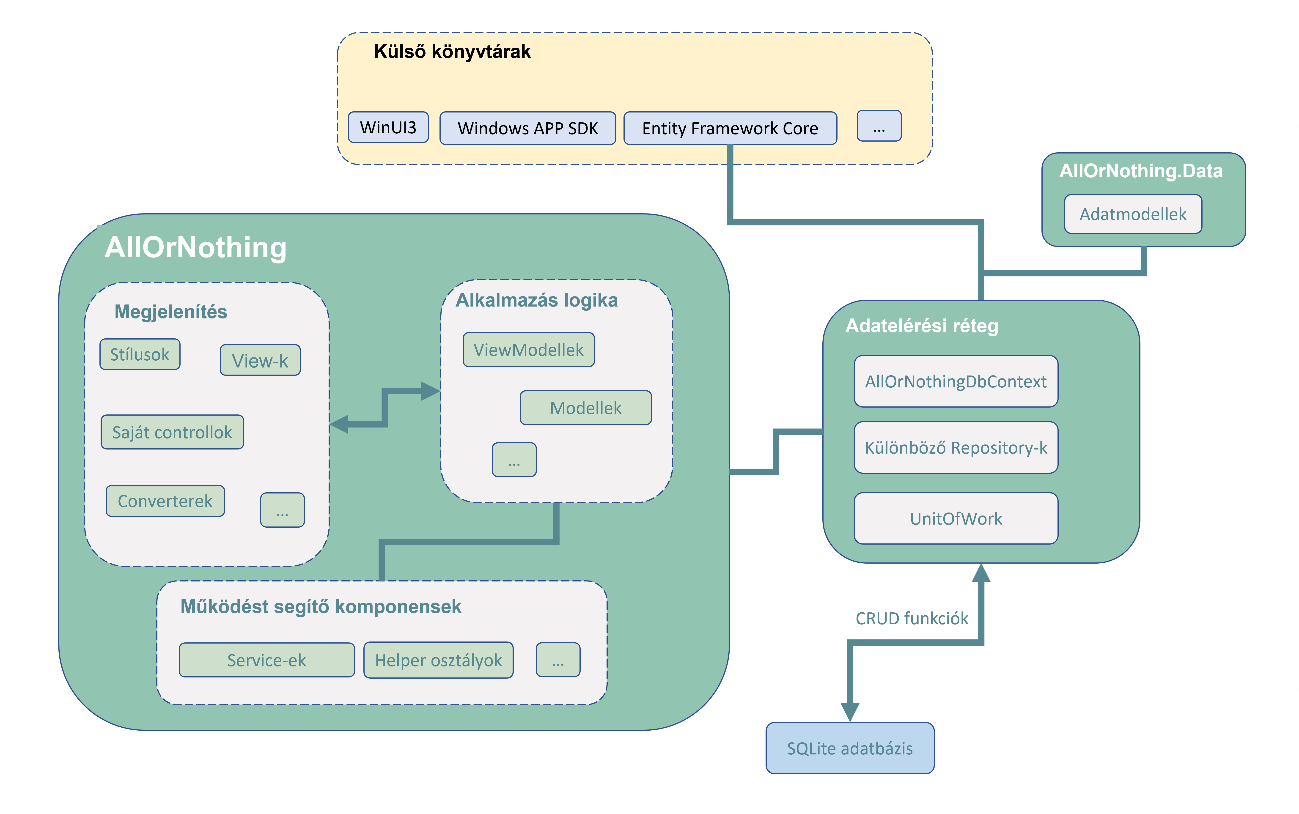
Ebben a projektben kerültek definiálásra az adatok kezeléséhez és tárolásához kapcsolódó adatmodellek.

**AllOrNothingTest**

Ebben a projektben kerültek implementálásra a funkciók megfelelő működését biztosító egység tesztek. (Bővebben lásd: Tesztelési terv c. fejezet)

Felhasznált NuGet csomagok:

* converlet.collector (3.1.0)
* FluentAssertions (6.6.0)
* Microsoft.NET.Test.Sdk (16.11.0)
* xunit (2.4.1)
* xunit.runner.visualstudio (2.4.3)



19. Ábra, Az alkalmazás főbb részeinek vázlatos modellje

## Saját vezérlők

A program bizonyos funkcióinak implementálásához szükség van olyan UI elemekre, amik nem állnak rendelkezésre a keretrendszer alap eszköztárának részeként. Ezen funkciókat célszerűnek tartottam egy-egy új control-ban implementálni, ahelyett, hogy megpróbálom belezsúfolni őket egy-egy ViewModelbe, ezzel is javítva a kód szerkezetét.

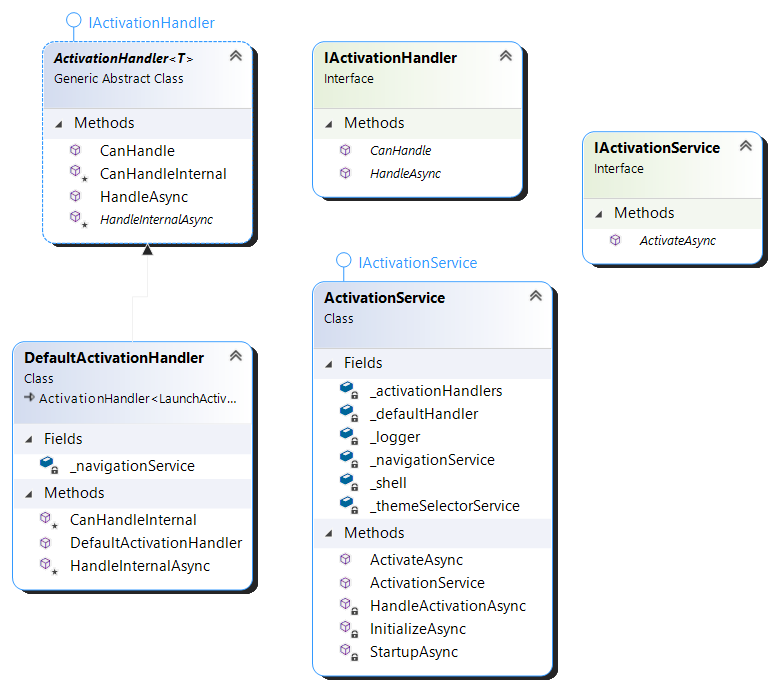
### ValidatedTextBox

Ez a vezérlő egy ellenőrzött bemeneti lehetőséget nyújt, egyenlőre csak számok számára, de későb egyszerűen bővíthető. Az osztály a Microsoft.UI.Xaml.Controls.TextBox osztályból származik.

A VlaidateNumbers property-t igaz értékre állítva garantáltan olyan Text propertyt kapunk, amit intté lehet konvertálni. Ha olyan stringet írunk be, a nem alakítható intté (nem numerikus karakter, vagy túl nagy lenne a szám hogy intté alakítsuk), akkor a mező az ezt megelőző érévnyes értékét veszi fel. Ha ilyen nincs, akkor 0-t.

## Indulás, aktiváció

Az alkalmazás belépési pontja a környzet által generált App.g.cs fájlban található, ahol az App osztály példányosításával át is tértünk a fejlesztői kódra. Az alkalmazás megfelelő indulásáról az ActivationService gondoskodik. Itt történik meg a megjelenítési téma kiválasztása, az adatbázi megfelelő állapotúra hozása. Ide illeszthető be minden olyan logika, amely az indulást követően azonnal szükséges a helyes működéshez.



. Ábra, Az aktivációt végző komponensek UML diagramja

## Navigáció

A program logikailag elkülöníthető funkciói az alkalmazáson belül külön oldalon kerülnek megjelenítésre. A következő oldalak léteznek jelenleg:

* főmenü
* játék beállítások
* Játék
* eredmények
* kérdéssorok-kezelő
* játékos kezelő
* alkalmazás szintű beállítások
* statisztikák (lsd.: További fejlesztési lehetőségek c. fejezet)

Ezen oldalak megjelenítése, és az oldalak közötti váltás, egy Frame, és egy navigációs szolgáltatás segítségével történik.

### Shell

A megjelenítés alapját a ShellPage és a hozzá tartozó ShellViewModel képzi. A SehellPage a Microsoft.UI.Xaml.Controls névtérben megtalálható Page osztályból származik, és funkcióit xaml kódon keresztül nyeri el. Az oldal tartalmát egyetlen, szintén a fenti névtérben megtalálható NavigationView objektum teszi ki. Ennek a MenuItems gyűjteményébe vehetjük fel azokat az elemeket, amiket a menüben szeretnénk látni, a megjelenítendő oldalt pedig a NavigationView Content property-jének értékül adott Frame objektum fogja a képernyőre renderelni. A későbbiekben ha a program oldalt vált (akár a navigációs menüre történő kattintás hatására, akár más okból), akkor mindig a ShellPage belsejében található Frame által megjelenítendő oldalt változtatjuk.

A navigáció másik két meghatározó része a NavigationService és a NavigationViewService.

### PageService

Az oldalak egyszerű betölthetősége érdekében kulcs-érték párokként összekötjük az egyes oldalakhoz tartozó ViewModel-leket, és az oldalakat. A későbbiek során így már elég lesz csak a betöltendő oldal ViewModel-jére hivatkozni, és ebből tudható lesz, hogy melyik oldalt akarjuk betölteni. Fontos, hogy ezen logika miatt viszont egy az egyhez reláció él az oldalak, és a hozzájuk tartozó ViewModellek között. Az előbb leírt folyamat a PageService osztályban történik.

### NavigationViewService

Ez a szolgáltatás a NavigationView által tartalmazott MenuItem-ek felé húzott absztrakciós réteg, ami segíti azok használatát. Referenciaként megkapja a ShellPage-en található NavigationView objektumot, majd a metódusaival ezt manipulálja. Legfontosabb funkciói, az új elem felvétele a navigációs menübe, valamint az egyes menüpontokra történő kattintás kezelése.

Az új elem ellenőrzött hozzáadásához tartozik, hogy ne lehessen többször ugyanazt a navigációs elemet a menühüz adni, valamint, hogy ne legyen a menüben két olyan elem, ami ugyanarra az oldalra vezet. Ez utóbbi ellenőrzéséhez szükséges a privát mezőként tárolt PageService példány.

A NavHelper osztály segítséget nyújt, hogy be tudjuk állítani az egyes menüelemekhez, hogy melyik oldalra vezessenek. A NavigationViewService osztály feliratkozik a NavigationView ItemInvoked eseményére, mely az egyes menü elemekre történő kattintás hatására kerül elküldésre. Az esemény fogadásakor a privát referenciaként tárolt NavigationService példány NavigateTo metódusának meghívásávlakezdjük meg a tényleges navigációt.

### NavigationService

Az osztály paraméterként kapja meg a ShellPage-en szereplő Frame-et, majd ezen objektumot csomagolja be biztonságos metódusokkal, amik hívása biztosan elvárt eredményt ad. Legfontosabb feladata, a Frame NavigateTo metódusának meghívása, mely a saját azonos nevű metódusában történik meg. Mivel a Frame a céloldal típusát kéri, ezen metódus pedig a céloldal ViewModel-jének nevét kapja meg, ezért szükség van a privát PageService objektumra, hogy lekérjük a ViewModel-hez tartozó típust.

## Adatok bevitele és szerkesztése

A programnak lehetőséget nyújt különböző adatok tárolására, és szerkesztésükre (lásd Adatelérési réteg), ezekhez a funkciókhoz pedig külön oldal tartozik a megjelenítési rétegben. Kér ilyen oldal van:

### PlayerAddingPage + ViewModel

Ezen az oldalon lehet a meglévő játékosokat változtatni, és újakat hozzáadni. Az oldal megjelenítésének 2 fő részét a bal oldalon található DataGrid, ahol az elérhető összes játékos megjelenik, és a jobb oldalon található szerksztő felület adja.

Az oldal működése a következő logika mentén működik:

* Az oldalhoz tartozó ViewModel-ben található egy SelectedPlayer property. Ennek az értéke jelenti az aktuáisan kiválasztott játékost.
* A DataGrid SelectedItem property-jét hozzákötjük a SelectedPlayer-hez, tehát amikor a táblázatban más játékost választunk ki, a játékos a ViewModelben is változni fog.
* Az aktuálisan kiválasztott játékos adatai betöltődnek a jobb oldali szerkesztő felületre a megfelelő adatkötések segítségével, és a TwoWay Binding segítsgével az oda bevitt adatok módosítják ViewModel reprezentációját.
* Ha nincs kiválasztva játékos, akkor a szerkesztő felület eltűnik

A fenti logika lehetővé teszi a kiválasztott játékos adatainak szerkesztését, ezt viszont menteni is szükséges. A mentés gomb lenyomásának hatására lefut egy metódus, amely ellenőrzi, hogy kitöltésre került-e minden kötelező mező, illetve egy privát mezőként tárol UnitOfWork példány segítségével elmenti az adatbázisba a változást, majd frissíti a megjelenített játékosok listáját. Ha valamely kötelező mező hiányzik, úgy egy felugró ablak értesíti a felhasználót erről.

„Új játékos” gomb lenyomására a SelectedPlayer új értéket kap, és ugyanúgy szerkeszthetővé válik, mint ha a táblázatban került volna kiválasztásra.

A játékos törléséhez a „Törlés ” gombra kell kattintani, ekkor a játékos IsDeleted adattagja vált át igaz értékre, és később ez alapján már nem kerül listázásra az elérhető játékosok között.

A játkosoknál tárolt adatok között nincs különösebb ellenőrzést igénylő típus, ezért ez itt kihagyható.

### QuestionSeriePage + ViewModel

A kérdéssorok esetében ugyanaz a logika érvénys mint a játékosoknál, és az oldal szerkezeti felépítése is hasonló. Itt is megvan a baloldali DataGrid, benne az elérhető kérdéssorokkal, viszont a jobb oldalon található szerkesztő fellet valamivel összettebb.

Mivel egy kérdéssorhoz több téma, egy témához pedig több kérdés tartozik, ezért a megjelenítést ItemsControl-ok segítségével kellett megoldani. Hasonlóan a játékosok szerkesztéséhez, itt is található egy EditingSerie nevű property, ami az aktuálisan szerkesztett kérdéssort tárolja. Ennek a Topics gyűjteményét rendeljük hozzá az ItemsControl ItemsSource propety-jéhez, és azon belül DataTemplate-ek segítségével jelenítjük meg az adatokat. Szükség lesz egy újabb ItemsControlra a kérdések megjelenítéséhez, az ugyanilyen logika alapján kerül használatba.

Mivel a kérdéssorok tartalmaznak szám típusú mezőt is, ezért szükséges ezek avlidációja. Ezt egy saját control, a ValidatedTextBox oldja meg (lásd. Saját vezérlők).

Az egyes témák szerkesztő felületén helyet kapott egy panel, amivel a témához tartozó ismeretköröket lehet beállítani. Az ott láható keresősávban keresve megkapjuk a legpontoabb egyezést mutató elemeket egy drop-down-ban, és rájuk kattintva hozzáadodnak a panelhez. Egy témához bármennyi ismeretkör tartozhat, ezek egyediségéről pedig a legördülő lista gondoskodik, oly módon, hogy a már kiválasztott ismeretköröket nem lehet mégegyszer hozzáadni. A témák ismeretköreit a kérdéssorok későbbi megjelenítésekor összeúniózzuk, így mindegyik megjelenik, de mindegyik csak egyszer.

A mentés és a törlés is ugyanolyan logika alapján működik a kérdéssorok esetében is, mint a játékosoknál.

## A játék folyamata

Az alkalmazás a felhasználói dokumentációban leírtak szerint értelmezi a játékot, és a köröket. (A játék egy nagy egysége, amely több körből áll)

A menüből az „Új játék” gombra kattintva a GameSettingsPage-ra jutunk el. aminek kettős feladata van. Az egyik feladat, hogy beállíthassuk rajta azokat a dolgokat, amiket később a játék folyamatában már nem módosíthatunk. Ezek közül a legfontosabbak:

* A játékban résztvevő játékosok

Ezt teljesen hasonló logikával teszi meg, mint ahogy a kérdéssor szerkesztő állítja be az ismeretköröket az egyes témákhoz. Amint hozzáadunk egy játékost az a ViewModelben a Players gyűjteményben kerül tárolásra.

* A játékban résztvevő csapatok

A játékosokból csapatok sorsolahótak a „Csapatok sorsolása” gombra kattintva. Ekkor a játék a „Maximális csapatétszám” mezőben szereplő adat alapján véletlenszerűen sorsol csapatokat, úgy hogy minden csapatban különböző játékosok legyenek. Ezek a csapatok a „Csapatok” panelen kerülnek megjelenítésre ItemsControl segítségével. A kisorsolt csapatok nevei alapból a tagok beceneveiből állnak (ha nincs becenév, akkor a rendes névből), és szerkeszthetők is.

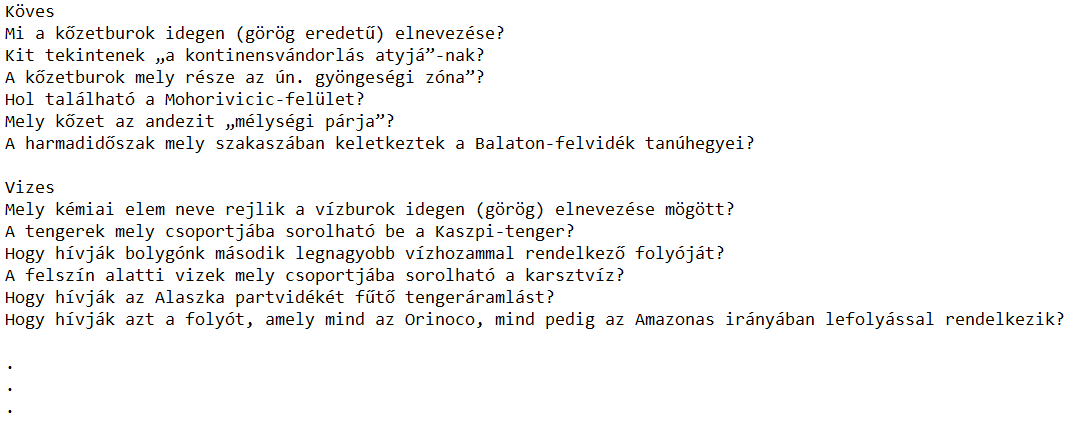
* Szükséges beállítani, hogy egy csapat hány kört játsszon.
* Meg kell adni az alkalom nevét.
* Ki kell választani a játékmódokat.

## Kérdéssorok olvasása fájlból

### Formátumok

Az alkalmazás lehetőséget nyújt kérdéssorok fájlból történő beolvasására is. Az évek során az elkészített kérdéssorok standard formátumává egy egyedi névvel[[9]](#footnote-9) ellátott Word dokumentum vált. Ennek használata közismert, rugalmasan szerkeszthető és egyszerű sima szöveges állománnyá exportálni, ami egy programnak lényegesen kezelhetőbb inputot nyújt.

A bevezetőben említett régi program bemenetül egy szekvenciális .txt kiterjesztésű fáljt várt, amelyben pontosan 5 téma, témánként pontosan 5 kérdés volt megadva. A témák elején a téma címe található, majd utána a kérdések. A témákat 2 \n karakter, a kérdéseket 1 \n karakter választja el ebben a formátumban. (ÁBra)



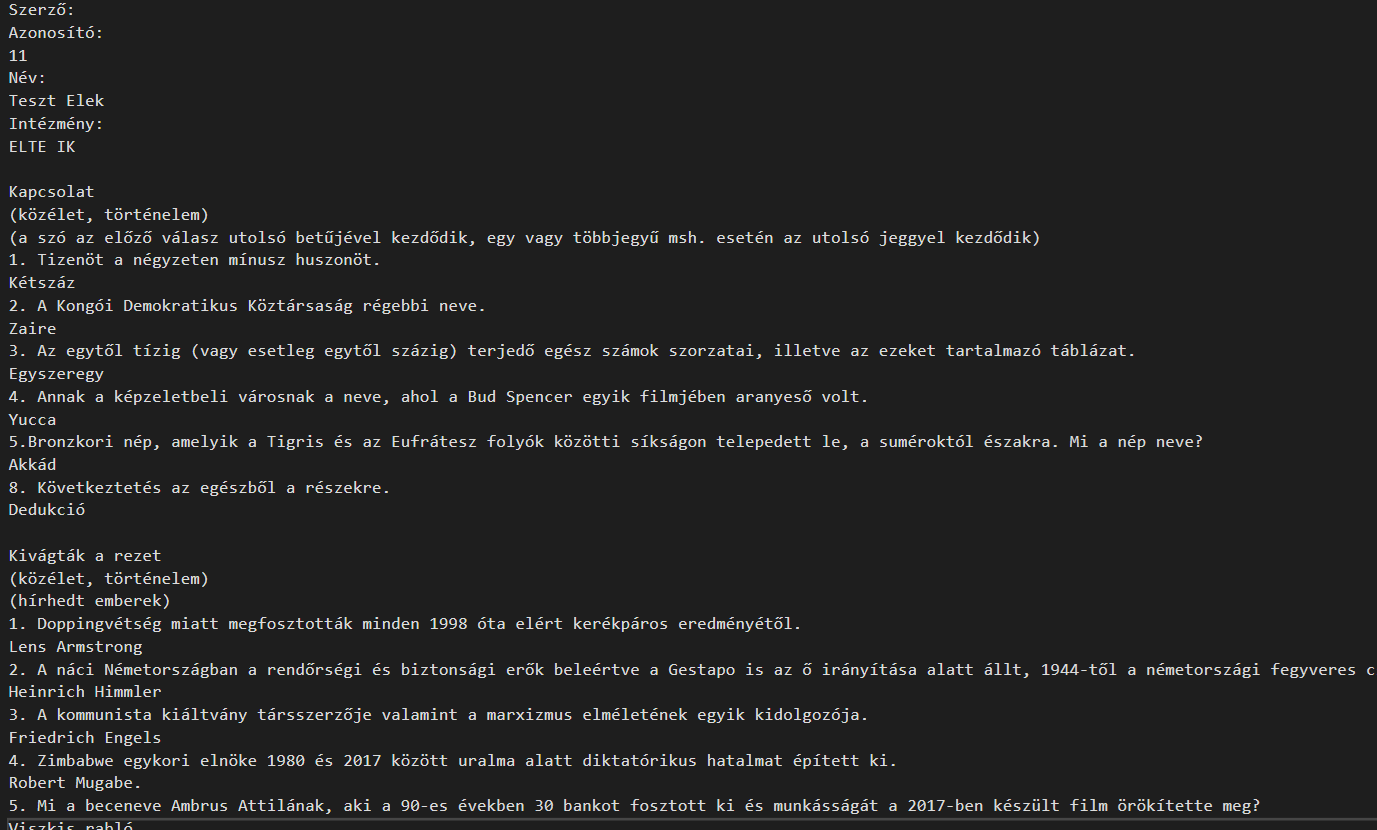
21. ábra, A régi program bemeneti formátimának részlete

Ez azért fontos, mivel ennek az alkalmazásnak a funkciók bővítése mellett visszafele is kompatibilisnek kell lennie, ebben az esetben a kérdéssorok formátumával.

A funkciók bővítésének érdekében az új kérdéssorformátumban szerepel a kérdéssor szerzőjének összes adata, témánként a téma címe, a hozzá kapcsolódó ismeretkörök felsorolása, a téma leírása, a kérdések és az azokhoz tartozó válaszok (lsd.: KérdéssorSablon.docx).

A minta fájlból exportált UTF-8 formátumú .txt állmány a következő struktúrát követi (ábra):

* Az elején \r\n karakterekkel elválasztva találhatóak a szerző adatai.
* Utána \r\n\r\n -nel elválasztva következnek a témák (pontosan 5 darab).
* A témákon belüli egyes sorok mind \r\n karakterekkel vannak elválasztva
* A téma elején annak címe áll
* Ezt követi zárójelek között vesszővel elválasztva a témára jellemző kompetenciák listája
* Ez után a téma lírása jön
* Végül pedig következik a 6 kérdés-válasz pár egymás után



22. ábra, Példa egy helyes szerkezetű kérdéssor fájlra

### QuestionSerieLoader

A programnak támogatnia kell a régi és új típusú kérdéssorok zökkenőmentes betöltését is, ezt a feladatot a QuestionSerieLoader osztály látja el.

Fő metódusai:

* LoadAllSeriesFromFolder  
  Paraméterként megkap egy mappát, majd megkísérli betölteni az ott található összes kérdéssort, majd ezek listájával tér vissza. Ebben a metódusban törtnik meg a beolvasás hibakezelése, az output paraméterként kapott errorMessage változóba táplája be az összes olyan kérdéssort, aminek betöltése során hiba történt, így a felhasználó látja, hogy melyik fájllal volt probléma.
* LoadFromTxt   
  Paraméterként megkapja egy fájl elérési útját, majd megkísérel betölteni egy kérdéssort belőle. Szükséges a teljes fájl beolvasása már ebben a lépésben, hogy el lehessen dönteni, hogy a régi kérdéssorra szabott transzformációs metódus kell végrehajtani, vagy pedig az újra tervezettet. Ha sikerül, akkor a kérdéssort kimeneti paraméterként adja vissza, és logikai igaz értékkel tér vissza, különbe a kérdéssor értéke null, a visszatérési érték pedig logikai hamis lesz.
* ParseOldFormat  
  Egy régi típusú szöveges fájl tartalmát várja stringként, majd ebből próbálja meg létrehozni a kimeneti paraméterként szereplő QuestionSerie objektumot. Itt történik meg ténylegesen a sorok feldarabolása, és az adatok megfelelő formátumúra hozatala. Mivel a régi típusú kérdéssorok kevesebb adatot tartalmaznak az újaknál, ezért a QuestionSerie, és egyéb adattároló objektumok olyan mezőit, amelyeknek megfelelő adatot régi típusú kérdéssor nem tartalmaz a mező típusának megfelelő alapértékre állítja be. Ha hiba történik a transzformáció során a metódus hamis értékkel tér vissza, a kérdéssor értéke pedig null lesz.
* ParseNewFormat  
  Egy új típusú szöveges állomány tartalmát kapja stringként, és ebből készít egy kérdéssort. A ParseNewFormat-hoz hasonlóan ennél is megtörténik az adatok feldolgozása, ám itt segítenek az osztály további metódusai is.

## Az Adatelérési réteg

Az alkalmazás működéséhez szükség van különféle adatok hosszútávú tárolására. Ez egy Entity Framework Core által támogatott SQLite adatbázissal kerül megvalósításra. Az adatelérési réteg logikája az AllOrNothing.Repository, a hozzá kapcsolódó adatmodellek pedig az AllOrNothing.Data projektben találhatóak.

### Adatbázisséma

Adatbázis kialakítása Code-first megközelítéssel történt. Az AllOrNothing.Data projektben található adatmodell osztályok három fő mappába rendeződnek:

**GameHistory**

Ebben mappában találhatóak azok a modellek, amelyek a játékok eredményinek hisztorizálásáért lesznek felelősek a későbbi fejlesztés során. Jelenleg nincsenek használva.

**Participants**

A játékosokkal és csapatokkal kapcsolatos modellek találhatóak meg benne.

|  |  |  |
| --- | --- | --- |
| Player | | |
| név | típus | leírás |
| Id | int (kulcs) | Egyedi azonosító |
| Name | string (kötelező) | A játékos neve |
| Institue | string | A játékos intézménye |
| NickName | string | A játékos beceneve |
| IsDeleted | bool | Törléskor a játékost nem törlődik, hanem ez a mezője vált igazra |

|  |  |  |
| --- | --- | --- |
| Team | | |
| név | típus | leírás |
| Id | int (kulcs) | Egyedi azonosító |
| Players | játéksok listája | Ez EF Core több a többhöz kapcsolatának kialakításához |
| TeamName | string (kötelező) | A csapat neve |

**QuestionSerie**

Itt a kérdéssorkkal kapcsolatos adatmodellek vannak. Az egyes kérdésekre vonatkozó adatokat a Question osztály foglalja magában.

|  |  |  |
| --- | --- | --- |
| Question | | |
| név | típus | leírás |
| Id | int (kulcs) | Egyedi azonosító |
| ResourceType | QuestionSerieResourceType | A kérdéshez tartozó erőforrás típusát határozza meg |
| Type | QuestionType | A kérdés típusát határozza meg |
| Resource | byte tömb | A kérdéshez tartozó erőforrás byte tömbként. |
| Text | string | A kérdés szövege |
| Answer | string | A helyes válasz a kérdésre |
| Value | int | A kérdés pont értéke |

A témákhoz tartozó ismeretkörök modelljeként a Competence osztály szolgál:

|  |  |  |
| --- | --- | --- |
| Competence | | |
| név | típus | leírás |
| Id | int (kulcs) | Egyedi azonosító |
| Name | string (kötelező) | Az ismeretkör neve |
| Topics | Topic-ok listája | A több a többhöz kapcsolat kialakításához szükséges |

A témákat a Topic osztály reprezentálja:

|  |  |  |
| --- | --- | --- |
| Topic | | |
| név | típus | leírás |
| Id | int (kulcs) | Egyedi azonosító |
| Name | string (kötelező) | A téma címe |
| Description | string | A téma leírása |
| Questions | Question-ök listája | A több a többhöz kapcsolat kialakításához kell |
| Competences | Competence-ek listája | A több a többhöz kapcsolat kialakításához kell |
| Author | Player | A kérdéssor szerzője |

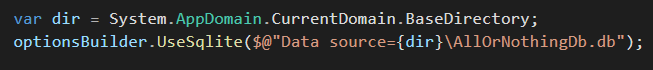
A QuestionSerie osztály az, ami előbb említett modelleket összefogja, és egy kérdéssort alkot ezekből.

|  |  |  |
| --- | --- | --- |
| Topic | | |
| név | típus | leírás |
| Id | int (kulcs) | Egyedi azonosító |
| Name | string | A kérdéssor címe |
| IsDeleted | bool | Törléskor a kérdéssort nem törlődik, hanem ez a mezője vált igazra |
| Topics | Topic-ok listája | A több a többhöz kapcsolat kialakításához kell |
| CreatedOn | DateTime | A létrehozás dátuma |

### Repository és Unit of work

A fenti adatmodellekből az Entity Framework Core segítségével migrációk alkalmazásával kerül kialakításra az adatbázis, az AllOrNothing.Repository projektben.

Az elérést az AllOrNothingDbContext osztály biztosítja, amely az EF Core konvencióinak megfelelően épül fel. A connection string annak érdekében, hogy az adatbázis fájl megfelelő helyre kerüljön a (ÁBRA) szerint alakul.



23. ábra, Az adatbázis connection stringje relatív elérési utat is felhasznál

A Migrations mappában találhatóak az adatséma módosításokat lekövető migrációk, melyek alkalmazásával kényelmesen módosítható az adatbázisséma.

Az adatbázis fölé plusz absztrakciós szintként került megvalósításra egy *repository* tervezési minta, ami esetemben egyszerű CRUD műveleteket valósít meg. A minta implementációja a generikus Repository osztályban található, és ebből leszármazva történik a repositoryk további esetre szabása.

Az egyes

## Tesztelési terv

Az alkalmazás elvárt működése tesztekkel ellenőrizhető. Az alkalmazás jellegéből adódóan egység-tesztek, és end to end tesztek kerültek megvalósításra.

### Unit-tesztek

Az egység tesztek működtetéséhez praktikus és hatékony környezetet nyújt a Visual Studio, ezen teszetek az AllOrNothingTest projektben találhatóak irányultságuknak megfelelő fájlban.

A teszt projekt az xUnit 2.4.1-es verzióját használja a tesztesetek megteremtéséhez. Ezen kívül a kényelmes tesztelés érdekében a Fluent Assertions (6.6.0) NuGet csomag is felhasználásra került.

#### **Fájlból olvasás**

A kérdéssorok különböző formátumainak fájlból történő beolvasása számtalan hiba merülhet fel, és fontos, hogy ezeket megfelelően kezelje a program. Ennek biztosításáhozz a QuestionSerieLoaderTest osztályban a QuestionSerieLoader osztály metódusainak tesztelése talélható.

A tesztesetek a következők:

* Helytelen kiterjesztésű fájl
* Helyes régi formátumú fájlok
* Helytelen régi formátumú fájlok
  + kevesebb téma a fájlban
  + kevesebb kérdés egy témában
  + több téma a fájlban
  + több kérdés egy témában
  + rossz elválasztó karakter a témák és kérdések között
* Helyes új
  + nincs megadva szerzői adat
  + csak a szerző neve van megadva
  + csak a szerző id-je van megadva
  + csak a szerző intézménye van megadva
  + nincs játékos az id-vel, csak id van megadva
  + id és név van megadva, nincs játékos az adatbázisban a megadott id-vel
  + van játékos az id-vel, de nem a megadott néven
  + nincs megadva a téma leírása
  + nincs megadva ismeretkör
  + olyan ismeretkör van megadva, ami nem szerepel a rendszerben
  + egy témánál többször szerepel ugyanaz az ismeretkör
* helytelen új
  + kevesebb téma a fájlban
  + kevesebb kérdés egy témában
  + több téma
  + több kérdés
  + más elválasztó karakter
  + érvénytelen id van megadva (nem szám, túl nagy szám stb)
  + hiányzik a leírás sora
  + hiányzik az ismeretkör sora
  + hiányzik a () a leírásból
  + hiányzik a () az ismeretkörökből
  + nem vesszővel van elválasztva az ismeretkör

#### **Csapatgenerálás**

A játék beállításai során a résztvevő játékosokból a program csapatokat generál. Ezen generálás szabályosságát az AllOrNothingSettingsViewModelTest osztály bizonyos metódusai tesztelik.

A tesztesetek a következők:

* tetszőleges számú játékos, mindenki csak egyszer szerepel csapatban összesen
* tetszőleges számú játékos, a legnagyobb csapat létszáma maximum M[[10]](#footnote-10)
* kevesebb játékos mint M

#### **Körök generálása**

A tesztesetek listája a AllOrNothingSettingsViewModelTest osztályban:

* minden körben különböző csapatok szerepel?
* minden körben megfelelő számú csapat csapat szerepel?
* minden csapat pontosan K körben szerepel
* nincs csapat megadva
* kevés csapat van megadva (?)

### End to end tesztek

|  |  |  |
| --- | --- | --- |
| Player adding page test | | |
| id | Test-case | Expected result |
| 1.1 | Open the application | The application opens the main menu page, with 3 menu points. („Új játék”, „Kérdéssorok”, „Játékosok”) |
| 1.2 | Click ont he menu point „Játékosok” | The player editor page opens and the navigation bar indicates the switch. The existing players are listed on the left side, ont he right side is a text: „Válasszon játékost!” |
| 1.3 | Press the „Új játékos” button | The „Azonosító”, „Név”, „Becenév”, and „Intézmény” fields appears, all of the is empty except „Név”. The „Mentés” and the „Játékos törlése” buttons are visible. |
| 1.4 | Fill the fields „Név” : E2E Test1,  „Becenév”: E2E1, „Intézmény”: Testing | The fields are properly filled |
| 1.5 | Press the „Mentés” button. | A pop-up is shown with the „Sikeres mentés!” message. |
| 1.6 | Press the „Ok” button in the pop-up | The player named „E2E Test1” is present in the list, and there is a text on the right side: „Válasszon játékost!”. |
| 1.7 | Press the „Új játékos” button | The „Azonosító”, „Név”, „Becenév”, and „Intézmény” fields appears, all of the is empty except „Név”. The „Mentés” and the „Játékos törlése” buttons are visible. |
| 1.8 | Fill the fields „Név” : E2E Test2,  „Becenév”: E2E2, „Intézmény”: Testing | The fields are properly filled |
| 1.9 | Press the „Mentés” button. | A pop-up is shown with the „Sikeres mentés!” message. |
| 1.10 | Press the „Ok” button in the pop-up | The player named „E2E Test2” is present in the list, and there is a text on the right side: „Válasszon játékost!”. |
| 1.11 | Select the player named „E2E Test2” | The form is filled with the previously given data. The „Mentés” and „Játékos törlése” buttons are visible. |
| 1.12 | Click on the „Játékos törlése” button | A pop-up message appeares: „Biztosan törli a játékost?” |
| 1.13 | Press the „Mégse” button | The pop-up disappeares, and the player named „E2E Test2” is still avaible in the list. |
| 1.14 | Click on the „Játékos törlése” button | A pop-up message appeares: „Biztosan törli a játékost?” |
| 1.15 | Press the „Igen” button. | The player named „E2E Test2” is deleted from the list, and there is a text on the right side: „Válasszon játékost!”. |
| 1.16 | Repeate the previous steps, to have at least 16 players in the list | At least 16 players are avaible in the list |

|  |  |  |
| --- | --- | --- |
| Questionserie page test | | |
| Id | Test-case | Expected result |
| 2.1 | Open the application | The application opens the main menu page, with 3 menu points. („Új játék”, „Kérdéssorok”, „Játékosok”) |
| 2.2 | Click on the menu point „Kérdéssorok” | The questoin serie editor page opens and the navigation bar indicates the switch. The existing series are listed on the left side, ont he right side is a text: „Válasszon kérdéssort!” |
| 2.3 | Press the „Új kérdéssor” button | Ont he top of the form there is a field called „Kérdéssor címe”, with a valuo of „Új kérdéssor”. The form populates itself with 5 topics, all with the following values: Empty fields: „Téma címe”, „Leírás”. The seletion area named „Ismeretkör” is empty. 6 times the following: „Kérdés” (empty), „Válasz” (empty), „Érték” (0) |
| 2.4 | Change the field „Kérdéssor címe” to „E2E TestSerie1” | The field value is changed |
| 2.5 | Press the button „Mentés” | A pop-up is shown to warn you to fill all the rquiered fields. |
| 2.6 | Press the „Ok” button in the pop-up | The pop-up gets closed, and the form has the same values as before |
| 2.7 | Type some non numeric characters in the field „Érték”, and clikck elsewhere (or get rid of the focus on the field) | The field deletes the non-numeric characters, and set it’s value to 0. |
| 2.8 | Type 1000 to field „Érték” | The value is properly set |
| 2.9 | Type some non-numeric characters in the field „Érték”, and clikck elsewhere (or get rid of the focus on the field) | The field deletes the non-numeric characters, and set it’s value to 1000. |
| 2.10 | Type „t” in the field „Ismeretkör” | A list appears, and all values in it contain „t” |
| 2.11 | Click one of them | The competence gets added tot he area below |
| 2.12 | Press the X nex tot he competence | The competence is now removed from the area |
| 2.13 | Fill all the fields with some values excetp the „Ismeretkör” | Each field has a value, except „Ismeretkör” |
| 2.14 | At the first topic add competence „történelem” to the competence area | „történelem” is in the competence area |
| 2.15 | At the second topic add „történelem” to the competece area | „történelem” is in the competence area |
| 2.16 | At the last topic add „földrajz” to the competence area | „földrajz” is in the competence area |
| 2.17 | Press the „Mentés” button | A pop-up appears with the message „Sikeres mentés” |
| 2.18 | Close the pop-up by pressing „Ok” | The serie named „E2E TestSerie1” is visible in the list. Its „Dátum” field has the value of the current date, its „Ismeretkör” field has exactly the following values: „történelem”, „földrajz”. Ont he right side the „Válasszon kérdéssort” message is visible |
| 2.19 | Click on the previously created serie | The form gets populated with all the previously given values |
| 2.20 | Press the button „Kérdéssor törlése” | A pop-up shows with the message „Biztosan törli a kérdéssort?” |
| 2.21 | Press „Mégse” | The pop-up disappears, and the form’s values are unchanged. |
| 2.22 | Press the button „Kérdéssor törlése”, and after that press the button „Igen” | The same pop-up appears as before, and after that, the serie is removed from the list, and the form displays the message: „Válasszon kérdéssort!” |

|  |  |  |
| --- | --- | --- |
| Game test 1 | | |
| id | Test-case | Expected result |
|  | Open the application | The application opens the main menu page, with 3 menu points. („Új játék”, „Kérdéssorok”, „Játékosok”) |
|  | Click on the menu point „Új játék” | The game settings page opens, the navigation bar indicates the page-switch. |
|  | At the searchbox „Játékos keresése” search for the players you created at test-case 1.16. | A list is shown with the previously created players. |
|  | Click all of them one after another | All 16 players gets added to the area below |
|  | In the searchbox search for a player that is added to the area below | The player do not appers in the list |
|  | Click one of the X next to a player | The player is removed from the area |
|  | In the searchbox search for the player you removed in the prev. step | The player is visible in the list |
|  | Fill the field „Alkalom neve” with the value „Test Occasion1” | The vlaue is set properly |
|  | Fill the field „Körök száma” with some non numeric charackters, and click elsewhere | The field resets its value to 1 |
|  | Fill the field „Körök száma” with a nuumber that is less than 1, and click elsewhere | The field resets its value to 1 |
|  | Fill the field „Körök száma” with 3 | The value is set properly |
|  | Fill the field „Maximális csapatlétszám” with some non numeric charackters, and click elsewhere | The field resets its value to 1 |
|  | Fill the field „Maximális csapatlétszám” with a nuumber that is less than 1, and click elsewhere | The field resets its value to 1 |
|  | Fill the field „Maximális csapatlétszám” with 3 | The value is set properly |
|  | Press the „Csapatok sorsolása” button | At the „Csapatok” section many teams appears, all with a maximum of 3 player. |
|  | Change one of the teams team-name field to „TestTeam1” | The team-name is changed |
|  | Check the chekbox „Tematikus” | The checkbox is checked, and the time picker next to it gets enabled. It has a value of 1:00 |
|  | Set 0:00 at the time picker next to „Tematikus” | The value is set to 0:00, and the time picker gets disabled. The checkbox „Tematikus” gets unchecked. |
|  | Check the chekbox „Villám” | The checkbox is checked, and the time picker next to it gets enabled. It has a value of 1:00 |
|  | Set 0:00 at the time picker next to „Villám” | The value is set to 0:00, and the time picker gets disabled. The checkbox „Villám” gets unchecked. |
|  | Press the button „Kilépés” | A pop-up is shown wit the message „Biztosan kilép?” |
|  | Press the „Mégse” button | The pop-up disappears and everything is unchanged |
|  | Press the button „Tovább” | A pop-up is shown wit the message „Legalább egy játékmódot válasszon ki!” |
|  | Press the „Ok” button | The pop-up disappears and everything is unchanged |
|  | Check the chekbox „Tematikus” | The checkbox is checked, and the time picker next to it gets enabled. It has a value of 1:00 |
|  | Press the „Tovább” button | The page changes to the round setup page. There is no change in the navigation bar. Each team is listed 3 times in the section „Körök beosztása”. The first round is selected. |
|  | Select each round one after another from the list. | Each round has the same „Játékmód” state: „Tematikus” is checked and set to 1:00, „Villám” is unchecked |
|  | Set the „Tematikus” timepicker to 0:00 | The selected time is 0:00, and the timepicker gets disabled, and the checkbox unchecked. |
|  | Check the „Villám” checkbox | The checkbox is checked, and the time picker next to it gets enabled. It has a value of 1:00 |
|  | Set the „Villám” timepicker to 0:00 | The selected time is 0:00, and the timepicker gets disabled, and the checkbox unchecked. |
|  | Press the „Tovább” button | A pop-up shows up, with the message: „Válasszon ki egy kérdéssort!”, „Legalább egy játékmódot válasszon ki!” |
|  | Press the „Ok” button | The pop-up disappears, and everything is unchanged |
|  | //TODO: lejátszott kört tesztelni |  |
|  |  |  |
|  |  |  |

Adatelérési rétegről csak a Mindent vagy semmit programnál beszélhetünk, ebben az alfejezetben erről lesz szó.

# Összegzés

# További fejlesztési lehetőségek

A dolgozat tárgyaként elkészült alkalmazás csak egy kezdetleges része az általam elképzelt jövőbeli rendszenek. Az előd összes hasznos funkcióját megvalósítja, és számos új lehetőséggel is bővült, ám annak érdekében, hogy a határidőig egy igazán megbíztható és használható alkalmazás születhessen, sok új, és régóta várt funkció fejlesztését célszerű volt elhalasztani. Néhány példa potenciális fejlesztésekre:

## Kényelem

## Funkcionalitás

**A kérdéssorok jövője**

Azok körében, akik e játékot a bevezetésben leírtak szerint játsszák, megszokás, hogy a kérdéssorokat maguk a játékosok írják. Ez sok szempontból egy jól bevett gyakorlat, hiszen az érdekes témák és kérdések utáni kutatás közben sok új információ jut el az emberhez, folyamatosan művelődik, tanul. Az az álláspont is elfogadható, hogy valaki részéről a kérdések kitalálása, és témába rendezése, monoton és hosszadalmas, ezáltal unalmasnak hat. Egy dolog azonban egészen biztos: Jó kérdéssort írni úgy, hogy az ne a régiek témáinak összeválogatása legyen, illetve, hogy aránylag rövid, jól érthető és mégis érdekes kérdéseket tartalmazzon, ezáltal a játékosoknak és a külső szemlélőknek is izgalmas játékot nyújtson, nem egyszerű, és sok időbe telik. Véleményem szerint ez utóbbi az igazi probléma, mivel gimnazista kortól kezdve idősödve, az egyetem és munka mellett egyre kevesebb idejük van játékosoknak a kérdéssorok írására.

Az én megoldásom erre, illetve arra, hogy a kérdéssor írási kedv fokozatosan csökkent az elmúlt évek során, az egységbontás lenne. A jelenleg használt metodológiában az alapegység a kérdéssor. Elképzelésem szerint, ha kérdéssorok helyett témákat írnánk, az emberek szívesebben állnának neki, tudva, hogy nem egy egész sok témát felülelő produktumot kell megvalósítaniuk, figyelve arra, hogy az egyes témák lehetőleg ne fedjék egymást, hanem csak egy önálló témát.

Ehhez több ponton is szükséges lenne módosítani az alkalmazást. A kérdéssorok kezelésének lehetősége megmaradna benne, de az elsődleges szerkesztési oldal egy webes felületre kerülne. A játékosok fiókokat regisztrálhatnak maguknak az online felületen, és ott tudják rögzíteni, beküldeni az elkészült témáikat, illetve az asztali szerkesztőhöz hasonló módon ismeretkör címkékkel ellátni azokat.[[11]](#footnote-11) Ez praktikus mivel egy webszerveren publikált alkalmazás szinte bárhonnan elérhető, illetve megoldaná azt a problémát is, amit jelenleg a kérdéssorok Word-alapú készítése jelent. Az így készített kérdéssorok utólagos validációra szorulnak, de egy a jövőbeli webes felületen készített téma egyből garantáltan a megfelelő formátumban érkezne az adatbázisba. Az online felületen a megfelelő jogokkal rendelkező felhasználónak lehetősége lenne az összes beküldött kérdéssor megtekintésére, és azokból valamilyen dinamikus felhasználói élményt nyújtó módon (pl.: esztétikusan animált drag and droppal), a témacímkék segítségével kérdéssorok kialakítására. (A kérdéssorok kész témákból történő egyszerű összeállítására az asztali alkalmazásban is lenne lehetőség.)

Az így tárolt adatokat egy olyan online adatbázis tárolná, amihez a megfelelő azonosítók és jogosultságok megléte esetén az asztali alkalmazás példányai is csatlakozhatnának, az ottani kérdéssorok elérése érdekében. Ez egy központosított dinamikusabb rendszert eredményezne, de az asztali alkalmazás lokális offline adatbázisát nem lehet elhagyni, tekintve, hogy gyakran olyan környezetben kerül használatba a program, ahol szegényes az internet kapcsolat. Ezt a kettősséget egyfajta offline szinkronizáció bevezetésével orvosolnám. Ha megfelelő internetkapcsolat áll rendelkezésre, az alkalmazás (részletesebb beállítások alapján) szinkronizálja a saját adatbázisát, az online elérhetővel, így egy esetleges kapcsolatvesztés esetén is rendelkezésre állnak az online rendszerben addig elérhető adatok.

Mivel a kérdéssorok papír alapú verzióira a jövőben vélhetően továbbra is lesz igény, ezért a webes és az asztali alkalmazás is kapna egy funkciót, ami a kívánt formátumban nyomtatja ki azt. (Nyilván megfelelő nyomtató eszköz rendelkezésre állása esetén.)

**A fzikai nyomógomb rendszer**

A bevezetőben megismert fizikai nyomógomb rendszer egyik nagy hátránya, hogy nem támogatja a(z egyébként az eredeti vetélkedőben is szereplő) „rablás” funkciót. A rablás azt jelenti, hogy alap esetben a leggyorsabban gombot nyomó csapat adhat választ, ám ezen csapat rossz válasza esetén a további gombnyomások sorrendjében egyéb csapatok is válaszolhatnak a kérdésre. A jelenlegi rendszer felépítéséből adódóan nem bővíthető, így a jövőben a helyét egy Arduino által vezérelt központi egység venné át. Ez lényegesen javítaná az alkalmazás működési dinamikáját, több szempontból is. Egyrészt a programozható központi egységen egyszerűen beállítható a kívánt rablás funkció, és így a játékélmény is javul, másrészt egy ilyen mikrokontroller rengeteg bővítési lehetőséget rejt magában. Ami ehhez kapcsolódóan egyértelműen cél, az egy vezetéknélküli egy távvezérlő integrálása a rendszerbe, aminek segítségével a játékmester tudja „nullázni” a nyomógombok állapotát, illetve az elhangzott válaszok alapján megítélni, vagy levonni a pontot. A megfelelő ilyen eszközt még nem találtam meg, egy prezentációk irányításához használatos bluetooth-os vezérlő talán alkalmas lehet erre a célra. Ha egy ilyen eszköz sikeresen beépítésre kerülne a rendszerbe, akkor tulajdonképpen kiváltható lenne a nyomógombok „nullázását” végző személy, illetve a vezérlő ügyes konfigurálása esetén még akár a program kezelője is.

## Megjelenés

# Hivatkozások

Vollai, G. (2019. május 15). *Retro TV #1 – Mindent vagy Semmit!* Forrás: ubulvilaga.com: https://ubulvilaga.com/2019/05/15/retro-tv-1-mindent-vagy-semmit/

# Ábrajegyzék

[1. ábra, A főmenü 9](#_Toc102685565)

[2. ábra, A játékos nyilvántartó oldal 10](#_Toc102685566)

[3. ábra, A játékos szerkesztő felület 11](#_Toc102685567)

[4. ábra, A kérdéssorok szerkesztőfelülete 12](#_Toc102685568)

[5. ábra, Az új játék beállításai 15](#_Toc102685569)

[6. ábra, A játékos nevére keresve megjelennek a nevükben legpontosabb egyezést mutató játékosok 16](#_Toc102685570)

[7. ábra, A kiválasztott játékosok a keresősáv alatti területen jelennek meg. 16](#_Toc102685571)

[8. ábra, A csapatok panelekbe rendeződve jelennek meg, tetszőleges csapatnév is megadható 17](#_Toc102685572)

[9. ábra, A csapattagok drag & drop segítségével áthelyezhetőek 17](#_Toc102685573)

[10. ábra, egyéb játékbeállítások 18](#_Toc102685574)

[11. ábra, A körökre vonatkozó beállítások oldala 19](#_Toc102685575)

[12. ábra, Kérdéssor választó felület automatizált játék indítása után 20](#_Toc102685576)

[13. ábra, A tematikus játékmód felülete 21](#_Toc102685577)

[14. ábra, Az automatizált játék felülete 22](#_Toc102685578)

[15. ábra, Az automatizált játék új elemei a tematikushoz képest 22](file:///C:\Szakdolgozat\Szakdolgozat\QuizProgram_LauncherConcept\Documentation\Dokumentáció.docx#_Toc102685579)

[16. ábra A régi program funkcionalitására irányuló kérdésre kapott válaszok aránya 24](#_Toc102685580)

[17. ábra A régi program esetleges haszontalan funkcióiról érdeklődő kérdés, és az erre kapott válaszok. 24](#_Toc102685581)

[18. ábra A régi program kinézetét érintő kérdés 25](#_Toc102685582)

[19. ábra, A régi program bemeneti formátimának részlete 30](#_Toc102685583)

[20. ábra, Példa egy helyes szerkezetű kérdéssor fájlra 31](#_Toc102685584)

1. Ez az általam elvégzett középiskolában található, és nem az én tulajdonomat képezi. [↑](#footnote-ref-1)
2. Ez lehet bármilyen tetszőleges szabály, akár véletlen szám generálással is történhet. [↑](#footnote-ref-2)
3. Ebbe a döntő nem számít bele. [↑](#footnote-ref-3)
4. A döntőt is beleértve [↑](#footnote-ref-4)
5. https://docs.microsoft.com/en-us/windows/uwp/get-started/universal-application-platform-guide [↑](#footnote-ref-5)
6. https://docs.microsoft.com/en-us/dotnet/maui/what-is-maui [↑](#footnote-ref-6)
7. https://docs.microsoft.com/en-us/windows/apps/winui/winui3/ [↑](#footnote-ref-7)
8. Korábban Project Reunion (bővebben: https://docs.microsoft.com/en-us/windows/apps/windows-app-sdk/) [↑](#footnote-ref-8)
9. Általában a szerző nevéből vagy a készítés dátumából álló valamilyen kombináció [↑](#footnote-ref-9)
10. Az egy csapatba osztható játékosok maximális száma [↑](#footnote-ref-10)
11. Az alkalmazás éppen e funkció elősegítése miatt címkézi az ismeretköröket témánként, és nem kérdéssoronként. [↑](#footnote-ref-11)