Tartalomjegyzék

[1. Bevezetés 2](#_Toc509675719)

[1.1. A program áttekintése 2](#_Toc509675720)

[1.2. Témaválasztás 2](#_Toc509675721)

[2. Felhasználói dokumentáció 2](#_Toc509675722)

[2.1. A program általános leírása 2](#_Toc509675723)

[2.2. Rendszerkövetelmények 3](#_Toc509675724)

[2.3. A program telepítése 3](#_Toc509675725)

[2.4. A program használata 3](#_Toc509675726)

[2.4.1. Felhasználónév 3](#_Toc509675727)

[2.4.2. Játék indítása 3](#_Toc509675728)

[2.4.3. Kategóriaválasztó 3](#_Toc509675729)

[2.4.4. Eredménytábla 3](#_Toc509675730)

[3. Fejlesztői dokumentáció 3](#_Toc509675731)

[3.1. Fejlesztői eszközök 3](#_Toc509675732)

[3.2. Adatbázis 3](#_Toc509675733)

[3.2.1. Adatbázis tárolása 3](#_Toc509675734)

[3.2.2. Adatbázis frissítése 3](#_Toc509675735)

[3.2.3. Adatbázis felépítése 3](#_Toc509675736)

[3.3. A program során használt activity-k és metódusaik 5](#_Toc509675737)

[3.3.1. MainMenu 5](#_Toc509675738)

[3.3.2. CategoriesActivity 7](#_Toc509675739)

[3.3.3. MainActivity 7](#_Toc509675740)

[3.3.4. Scoreboard 7](#_Toc509675741)

[3.3.5. Information 7](#_Toc509675742)

[3.4. Tesztelési dokumentáció 7](#_Toc509675743)

[3.5. Továbbfejlesztési lehetőségek 7](#_Toc509675744)

[4. Forrásjegyzék 7](#_Toc509675745)

[5. Ábrajegyzék 7](#_Toc509675746)

# Bevezetés

## A program áttekintése

A záródolgozatom témája egy Android platformra fejlesztett Quiz alkalmazás, mely több kategóriában tartalmaz kérdés csomagokat, amik közül a játék elején és közben kell választani. A kérdések megválaszolása közben a játék pontokat számol, a jó válasz plusz ponttal, a rossz válasz pontlevonással jár, melyet a telefon rezgéssel jelez.

A főmenüből elérhető a ponttáblázat, melyben a hat legtöbb pontot elért helyezés látható felhasználónévvel együtt.

A játék alkalmazás bármely korosztály számára kikapcsolódást nyújthat, miközben tanulnak is belőle.

## Témaválasztás

A záródolgozat témái közül azért választottam az Android platformra írandó Quiz applikációt, mert véleményem szerint ez a rendszer napjaink legfeltörekvőbb operációs rendszere és még napjainkban is folyamatosan fejlődik és bővül. A szakdolgozat ledása után is szeretném rendszeresen bővíteni az alkalmazást új kérdéscsoportokkal és funkcionalitást illetően is.

Habár a program megírása előtt nem rendelkeztem kiemelkedő ismerettel a rendszert illetően, reményeim szerint a folyamat során piacképes tudásra tesztek szert, mellyel elhelyezkedhetek, mint kezdő Android programozó.

# Felhasználói dokumentáció

## A program általános leírása

A program telepítése és indítás után szükséges megadni egy felhasználónevet, ezen adat megadása nélkül nem is kezdhető el a játék. Amennyiben más felhasználó is használja ugyanezen az eszközön a játékot, lehetőségünk van a főmenüből megváltoztatni ezt az adatot.

Ezután rögtön a főmenübe érkezünk, ahol lehetőségünk van megadni egy felhasználói profilképet, de ez nem kötelező. Alapértelmezetten egy placeholder kép jelenik meg a felhasználó egyéni képe helyén, melyen kattintással lehet választani a felhasználó által kiválasztott galéria alkalmazással.

Főbb menüpontok:

* Új játék indítása: Ezen gombra kattintva a kategóriaválasztó oldalra jutunk, ahol a játék elkezdéséhez egy tetszőleges kategóriával indíthatjuk el a játékot.
* Eredménytábla: A játék során elért legjobb 5 eredményt és a hozzátartozó felhasználónevet nézhetjük meg ezen az oldalon. Amennyiben a felhasználó még nem játszott egy kört sem, nem fog megjelenni semmilyen adat sem.
* Felhasználónév: A játék elején megadott felhasználónevet lehet megváltoztatni ebben a menüpontban, de azzal ellentétben ebből a menüpontból változtatás nélkül ki lehet lépni.
* Kilépés

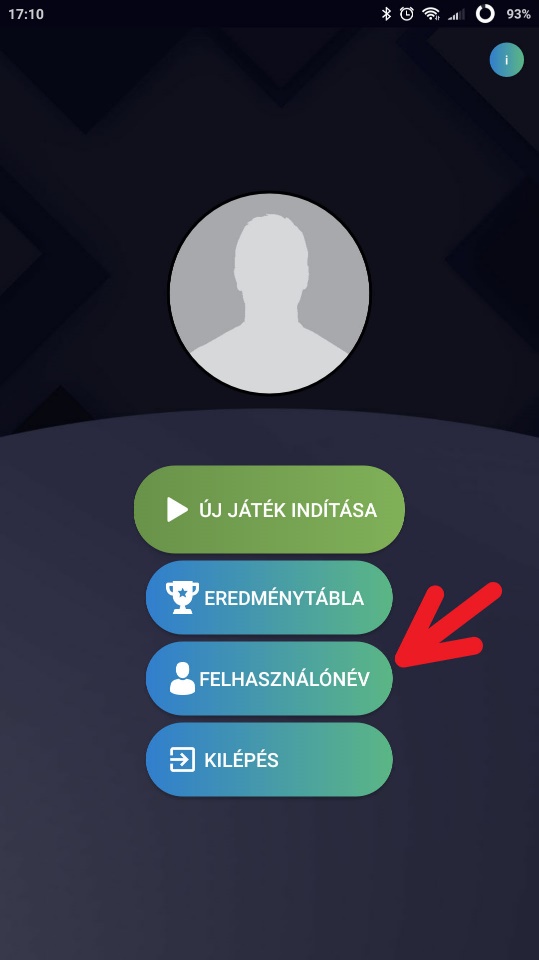
## Rendszerkövetelmények

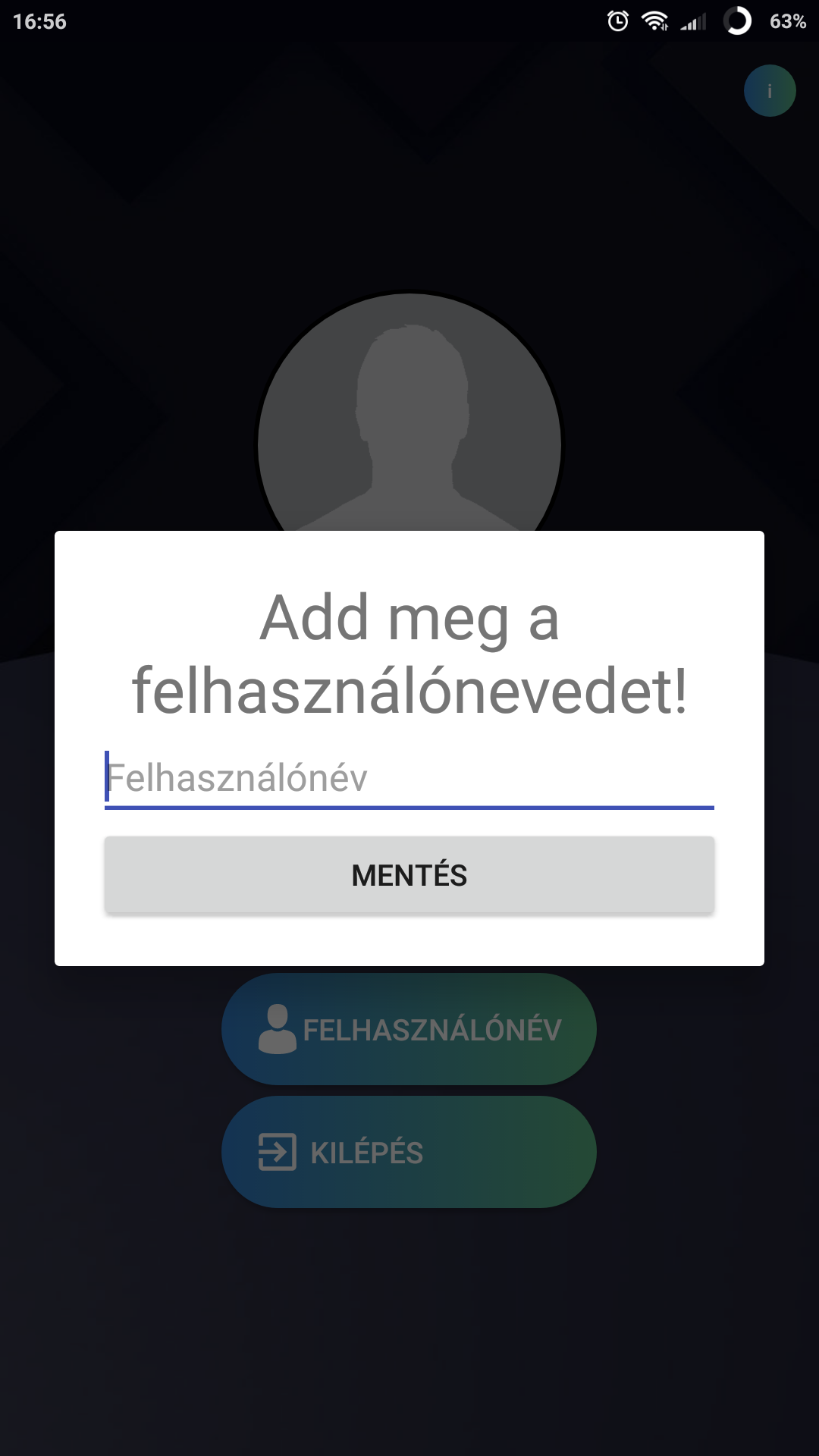
## A program telepítése

## A program használata

### Felhasználónév és profilkép

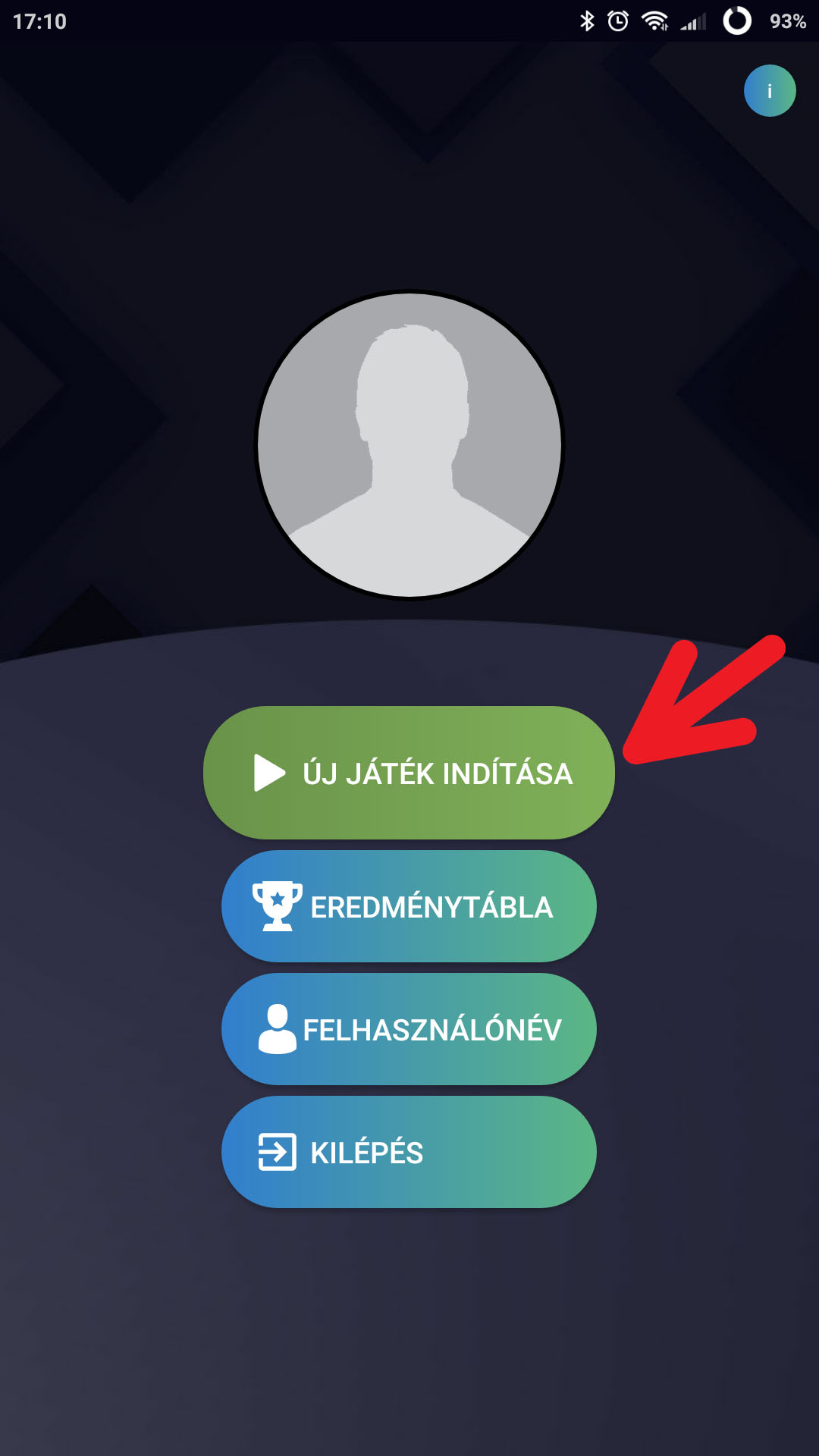
A program első indításakor kötelező megadni egy felhasználónevet, melyet a későbbiekben az eredmények megjelenítésekor fogunk felhasználni. Ezután opcionálisan megadható egy profilkép is, de amennyiben nem választunk semmilyen képet, az alapértelmezett placeholder kép lesz kiválasztva.  
A felhasználónév módosítására a főmenüből a Felhasználónév gombra kattintva van lehetőségünk. A kattintás után megjelenik egy beviteli mező, ugyanaz, ami az első indításkor is megjelent annyi különbséggel, hogy itt már nem kötelező megváltoztatni a felhasználónevet.



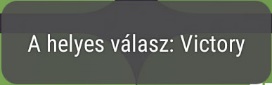


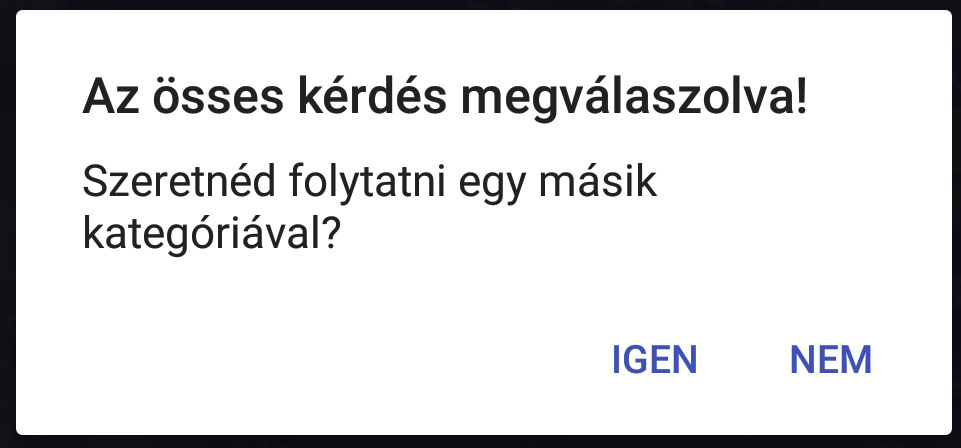
### Játék indítása

A felhasználónév megadása után lehetőségünk van elkezdeni a játékot az Új játék indítása gombra kattintva. Ezután megjelenik a kategóriaválasztó oldala, ahol lehetőségünk van kiválasztani, hogy melyik kategóriából szeretnénk kérdéseket kapni. A kategória kiválasztása után elkezdődik a játék.

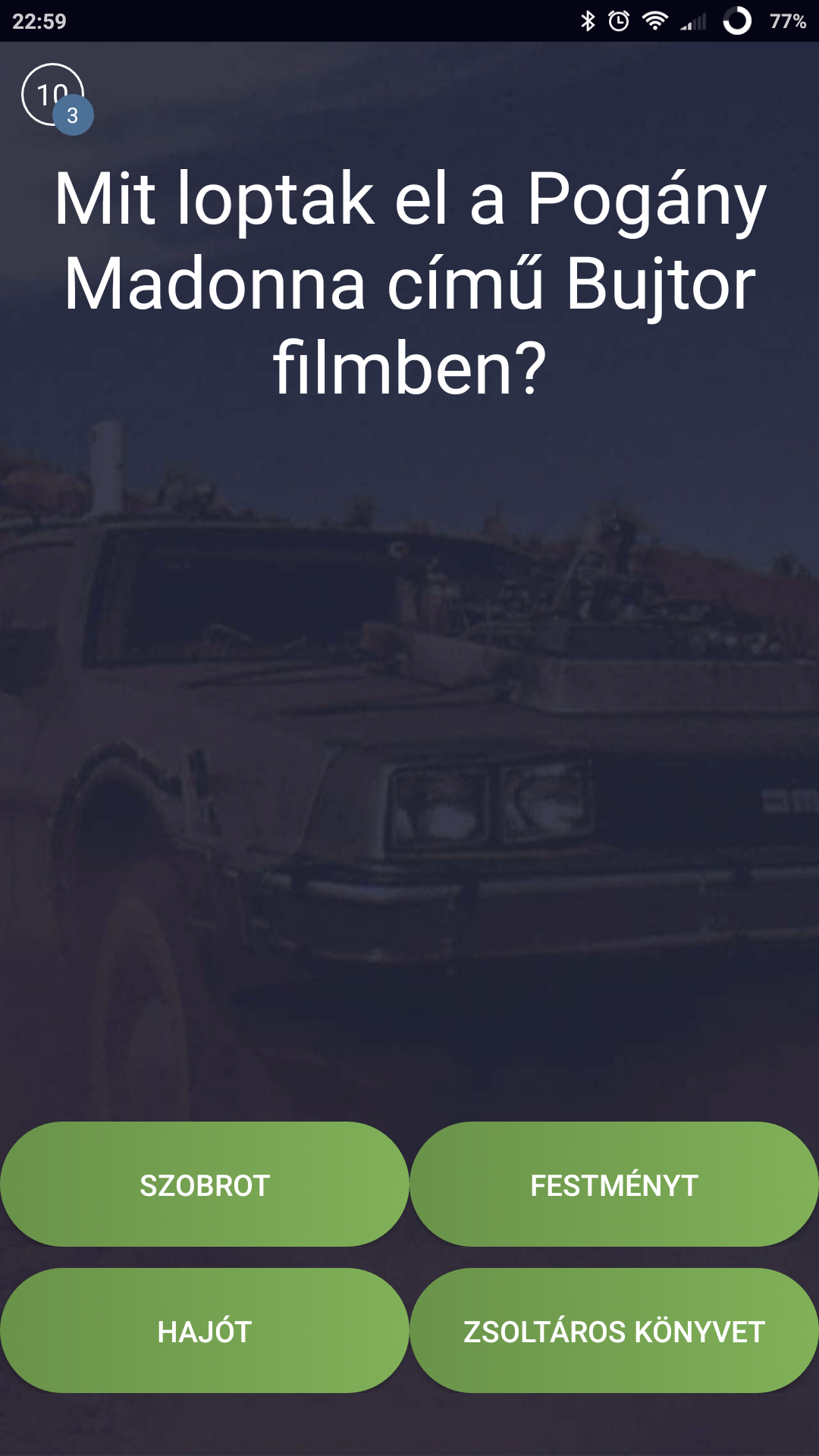


### A játék menete

A kategória kiválasztása után elkezdődik a valódi játék. A képernyő tetején jelenik meg az aktuális kérdés. A képernyő bal felső sarkában látható a jelen kategóriában lévő kérdések száma és előtte a jelenlegi kérdés sorszáma:. A képernyő alján 4 gomb található 4 válasz lehetőséggel. A válasz kiválasztása után azonnal értesülünk a válasz helyességéről. Amennyiben a válasz jó, ennek megfelelő hang játszódik le. Ha az adott válasz nem jó szintén hangjelzéssel értesülünk róla, de ezenkívül a telefon elkezd vibrálni is majd egy felugró ablakban megjelenik a valódi helyes válasz: . A kérdéssorozat végére érve egy felugró ablak tájékoztat arról, hogy nincs több kérdés a kategóriában.

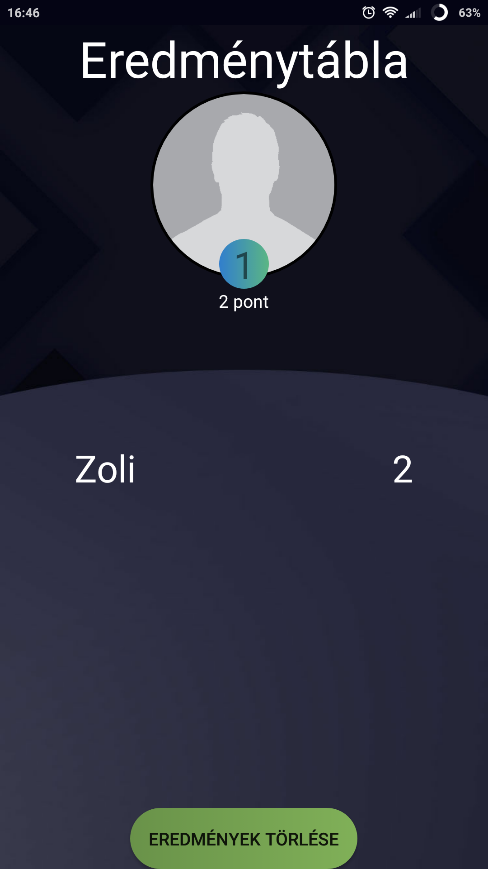


Ebből a felugró ablakból 2 opcióval léphetük tovább. Amennyiben az Igen opciót választjuk ismét a kategóriaválasztó oldalon találjuk magunkat ahonnét egy új kategóriát tudunk választani. A már végigjátszott kategória nem választható ismételten. A nem opciót választva a játék kilép és a fő menübe érkezünk. Az addig szerzett pontok elmentődnek, melyből a 6 legjobb helyezés megtekinthető felhasználónévvel és pontokkal együtt.

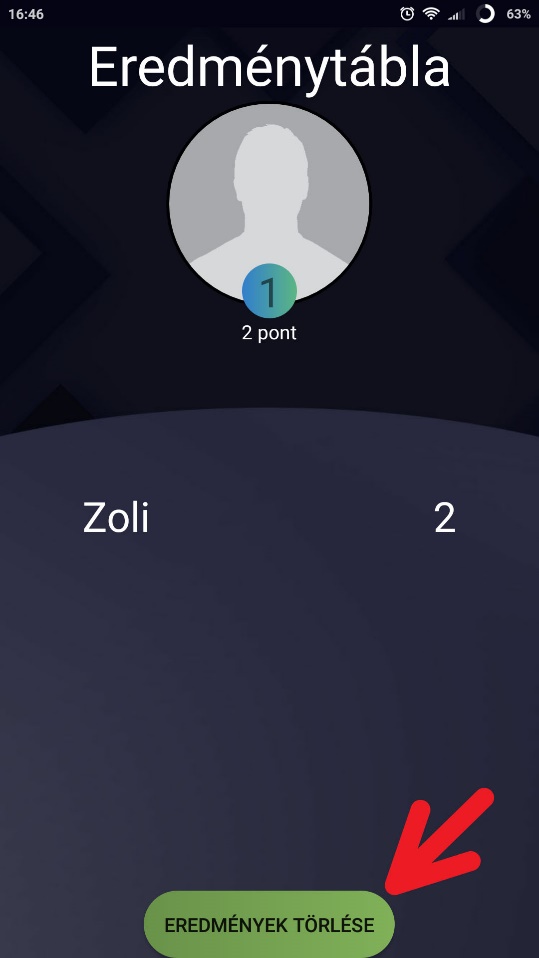


### Eredménytábla és pontozás

A játék során a jó válaszok megadásakor szerezhető pont. Minden egyes jó válasz egy pontot ér. Rossz válasz adásakor a pontszám változatlan marad. Az eredménytáblában az első helyzetnek a választott profilképe jelenik meg az elért pontszámokkal együtt az összes többi eredmény felett. Az első helyezett alatt 5 másik helyezés látható amennyiben már van annyi játszma. Ha még nincs 6 lejátszott játék, akkor üres helyek jelennek csak meg.



Az elért eredmények törölhetők a játékból. Az Eredmények törlése gombra kattintva az összes elért eredmény véglegesen törölve lesz a játékból.



# Fejlesztői dokumentáció

## Fejlesztői eszközök

A program megírásához használt program:

* Android studio 3.0.1

A megíráshoz használt fejlesztői környezet:

* API16 – Android 4.1

Az adatok tárolására használt eszközök:

* Az adatok ideiglenes tárolására Shared Preferences-t használok
* A kérdések és eredmények tárolására 2 táblás Sqlite adatbázis áll rendelkezésre

Az alkalmazásban található képek elkészítéséhez és szerkesztéséhez használt program:

* Adobe Photoshop CC

A teszteléshez használt eszközök:

* Android studio emulátor 7.0 androiddal
* Android 7.0-val rendelkező Xiaomi mi 5S Plus
* Galaxy s2 Android 7.1.2 Lineage os

A képernyőképek elkészítéséhez használt eszköz:

* Xiaomi MI 5s Plus

## Adatbázis

### Adatbázis tárolása

A program egy előre elkészített adatbázist használ, melyből a kérdéseket olvassa be és az elért pontszámokat írja be az előre megadott Felhasználónévvel együtt. Ez az adatbázis a lokális Assets mappában található.

### Adatbázis frissítése

Az adatbázis frissítésére lehetőség van külsőleg. Az alkalmazás DatabaseHelper class-án belül a verziószámot átírva az adatbázis frissítésre kerül. Az adatbázisba lehetőség van az eredmények eltárolása miatt futási időben is adatot illeszteni. A felhasználói eredmények a Scoreboard táblában kerülnek tárolásra.

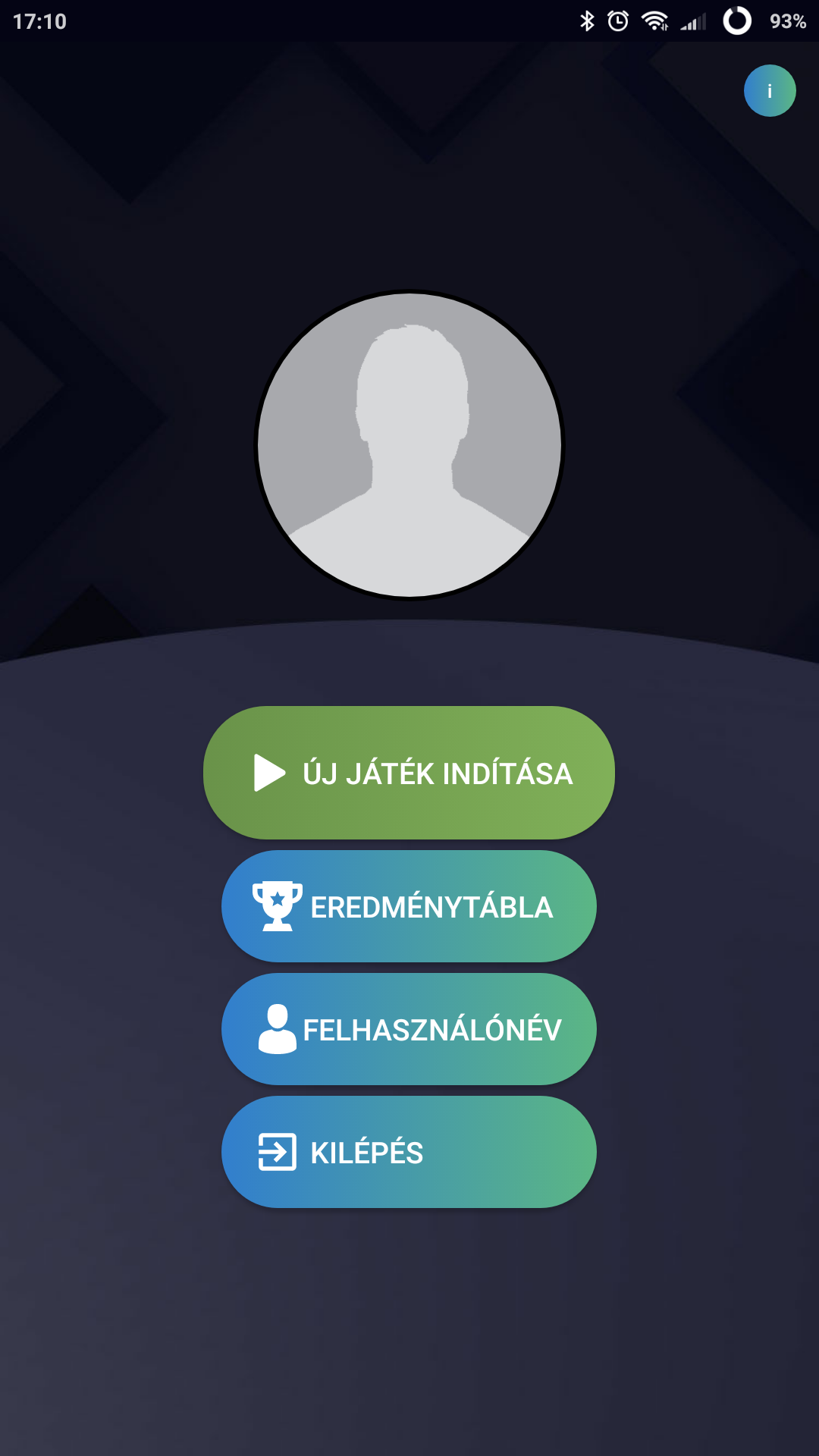
### Adatbázis felépítése

Az adatbázis 2 táblából áll:

* Scoreboard:
  + Jelen adatbázis tárolja el az elért pontszámokat és a hozzá tartozó felhasználónevet is.
  + Mezői a következő értékeket tartalmazzák:
    - ID: Ez a mező auto Increment-tel generálja az id-t az eredményhez
    - Felhasználónév: A játék során megadott felhasználónevet tárolja le, melyet első indításkor kell megadni, de változtatható a Főmenüben lévő „Felhasználónév” opcióval is.
    - Elért pontszám: Ez a kérdések megválaszoláskor összegyűjtött pontszámokat gyűjti össze.
* Quiz:
  + Jelen adatbázist a játék a kérdések és válaszok beolvasására használja
  + Mezői a következő értékeket tartalmazzák:
    - ID: Ez a mező auto Increment-tel generálja az id-t a kérdéséhez
    - Kérdés: Ez a mező tartalmazza a kérdést
    - Válasz1: Ez a kérdéshez tartozó első számú hibás választ tartalmazza
    - Válasz2: Ez a kérdéshez tartozó második számú hibás választ tartalmazza
    - Válasz3: Ez a kérdéshez tartozó harmadik számú hibás választ tartalmazza
    - Válasz4: Ez a kérdéshez tartozó helyes választ tartalmazza
    - Kategória: A játékban lévő kategóriaválasztó ezen mező alapján választja ki az adatbázisból a kategóriának megfelelő kérdéscsoportokat
    - Nehézség: Ezt a mezőt a játék jelenleg nem használja, továbbfejlesztési céllal szerepel benne.

## A program során használt activity-k és metódusaik

### MainMenu



3. ábra - Főmenü

Jelen Activityből lehet tovább irányítani a játék további részeibe. Ebben a layout-ban 6 kattintható felület található, melyből 5 gomb és egy kép.  
Mivel a program a felhasználói játékhoz kapcsolódó pontszámokat, a felhasználónevet és a már végigjátszott kategóriák neveit ideiglenesen egy Shared Preference-ben tárolja, ezért minden alkalommal, amikor ezt az activity-t betölti a felhasználó, ennek a shared preference-nek a tartalma törlésre kerül. Ezzel a módszerrel kiküszöbölhető, ha a játékból nem megfelelő módon lépnek ki, akkor ne tudják ott folytatni ahol megaszakadt.  
A layout első betöltésekor a program bekér egy felhasználónevet, melyet ezentúl egy Shared Preference-ben tárol. Amennyiben szükséges ezt eléri a többi activity. A játék végeztével a játék innét kiolvassa a felhasználónevet és a pontszámmal együtt letárolja egy adatbázisba.

* Új játék indítása: Ez a gomb egy Intent segítségével betölti az activity\_categories layout-ot.
* Eredménytábla: Ez a gomb egy Intent segítségével betölti az activity\_score\_board layout-ot.
* Felhasználónév: Ezen gomb megnyomásával megjelenik egy alert dialog, melyet az első indításánál a felhasználónév megadására használtam. Ehhez létrehoztam egy felhasznalonev.xml nevű layout-ot, melyet egy Alert Dialog-ban hívok meg. Ez a function egy boolean értéket vár meghívásnál, aminek az értékét az Alert Dialog setCancelable tulajdonságának beállítására használok fel. Első indításkor ez a tulajdonság false típusú, de ezen gomb megnyomásakor true a tulajdonsága. Így az első indításnál nem tudnak tovább jutni ennek megadása nélkül, de a gomb megnyomásakor már létezik az érték, így nem kötelező megváltoztatni azt.
* Kilépés: Befejezi az alkalmazás futását, meghívva a finishAffinity metódust.
* i: Ez a gomb egy Intent segítségével betölti az activity\_information layout-ot, melyen információt szerezhetünk a játék pontos működéséről és a pontelosztásról.
* Profil kép: A profilképre kattintva megjelenik a galéria, ahol kiválaszthatunk egy profilképet. Ez a későbbiekben az alkalmazás főmenüjében illetve az eredménytáblában lesz látható. A kép bitmapból át lesz konvertálva base64-re, mely így String-ként SharedPreference-ben lesz letárolva. A későbbiekben, így az adatbázisba is be fog kerülni, ha már elért valamennyi pontot a játék során.

ByteArrayOutputStream baos = new ByteArrayOutputStream();  
bitmap.compress(Bitmap.CompressFormat.*JPEG*, 80, baos); //bm is the bitmap object  
byte[] b = baos.toByteArray();  
String encoded = Base64.*encodeToString*(b, Base64.*DEFAULT*);  
  
SharedPreferences sharedPreferences = getSharedPreferences("UserInfo", Context.*MODE\_PRIVATE*);  
SharedPreferences.Editor editor = sharedPreferences.edit();  
editor.putString("image", encoded.toString());  
editor.commit();

A játék indításakor amennyiben már létezik kiválasztott profilkép ezt a base64 típusú string-et visszakonvertálja byte típusúra, melyet beállít profilképnek.

if (!imagestring.isEmpty())  
{  
 byte[] imageAsBytes = Base64.*decode*(imagestring.getBytes(), Base64.*DEFAULT*);  
 profile\_image.setImageBitmap(BitmapFactory.*decodeByteArray*(imageAsBytes, 0, imageAsBytes.length));  
}

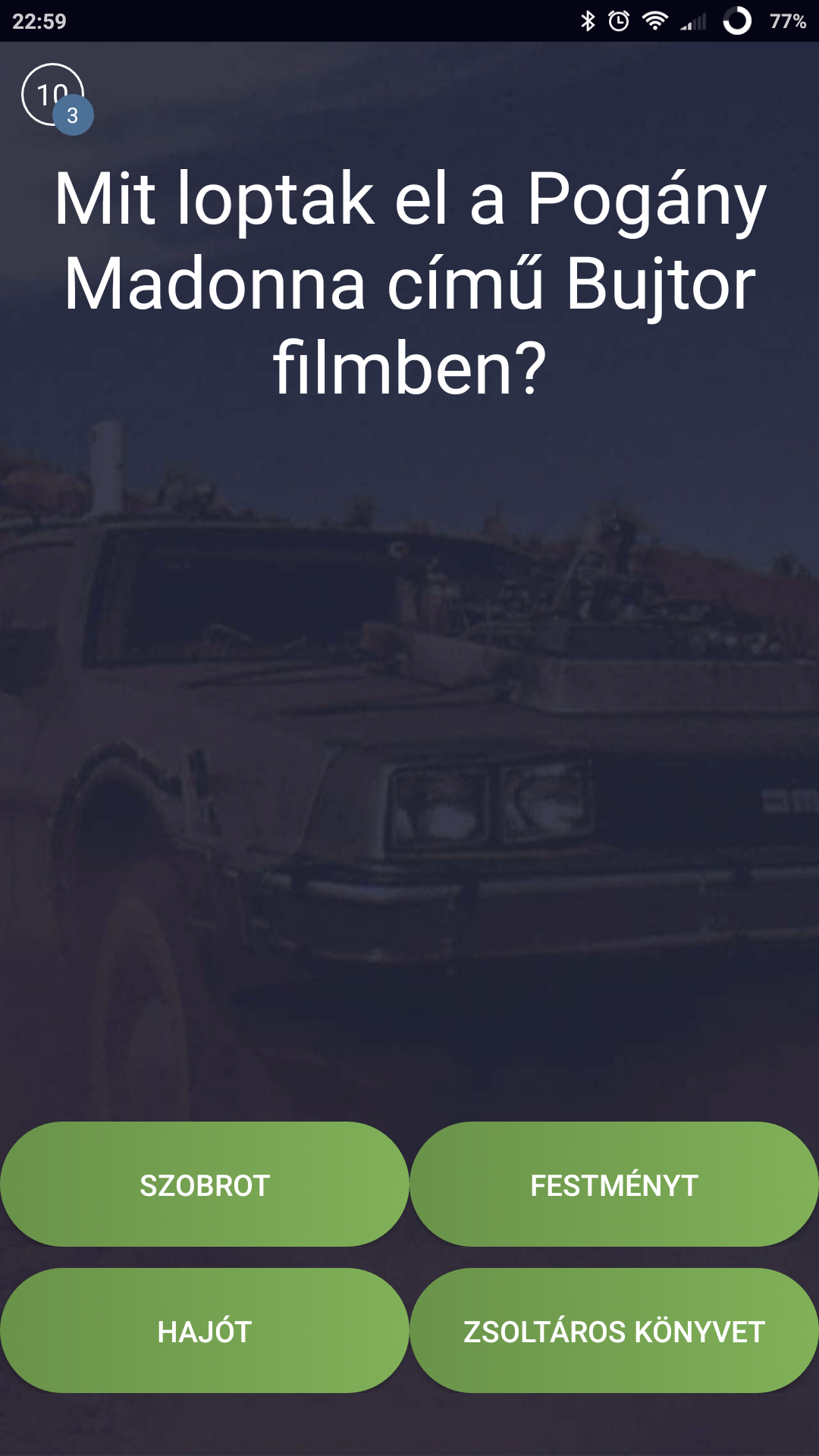
### CategoriesActivity



4. ábra - Kategória választó

Jelen layout-ra a Főmenü „Új játék indítása” gombjára kattintva ugorhatunk, vagy ha az egyik kategória kérdéseinek végére értünk, akkor a felugró Alert dialog box „Igen” gombjára kattintva nyílik meg újra ez a layout, a már végigjátszott kategóriát eltávolítva a kéernyőről.  
Az egyes kategóriákra kattintva Létrejön egy új Shared Preference „Categories” néven. Itt egy „kerdesselect” String-be elment egy sql lekérdezésnek a SELECT utáni részét, melyet a MainActivity-ben hívok meg a kérdések betöltésénél.

### MainActivity



Jelen activity tartalmazza a legtöbb metódust, mivel itt történik maga a játék.

Metódusok:

* init

Ez a metódus tartalmazza a játék során használt Alert Dialog boxokat, melyeket meg kell hívni az adott action-nél.  
Ezenkívül amennyiben már nem ez az első kategória, amelyiket kiválasztotta a játékos, a sharted perefence-ből letárolja egy változóba a már korábban elért pontokat. Az adatbázisba való beillesztéshez szintén változóba tárolja a felhasználó képének a base64-es szöveges változatát, melyet a profilkép megadásokat tároltam le egy shared perference-be és a felhasználónevet.

* onBackPressed

Ez a metódus egy beépített metódus, mellyel megadom, hogy a vissza gomb megnyomásakor az init metódusban deklarált alert\_kilep alert dialog box jelenjen meg. Ezzel a metódussal felülírom az alapértelmezett metódust mellyel az előző CategoriesActivity Intentre ugrana vissza.

* gombok

Ezzel a metódussal frissítem a gombok szövegeit új kérdés behívásakor. Annak érdekében, hogy minden esetben változzon a jó választ tartalmazó gomb, a gombsorrend változó tartalmaz 1-4-ig egy számot, melyet egy random generátor sorsol ki minden új kérdés behívásánál. Egy switch-case metódus kezeli, hogy melyik számot tartalmazza a változó. Mivel 4db gomb van, ezért 4 féle lehetőséget kell lekezelni ebben a metódusban. Mind a 4 case egyéni sorrendet tartalmaz a gombokra.

* helyesvalasz

Jelen metódus a gombok metóduson belül van meghívva, minden esetben akkor, amikor a gombsorrendben kisorsolt szám szerinti helyes gombra kattint a játékos. Ez a metódus először hozzád a pontokhoz egyet, meghívja a disablebuttons metódust, lejátszik egy hangot, mely jelzi a helyes választ majd késleltetve 500 millisecondummal meghívja az új kérdés és a gombok metódust.

A pontok alapértelmezetten 0 értékkel rendelkeznek, de amennyiben már végigjátszott egy kategóriát a játékos akkor egy SharedPreference-ben tárolt értéket kap meg ez a változó és azt növeli eggyel a metódus meghívásakor.

A disablebuttons metódus megakadályozza, hogy az új kérdés betöltéséig kattintható legyen a gomb.

Az 500 millisecundum lejárta után a program meghívja az újkerdes és gombok metódust. Az új kérdéssel generálódik a kategória szerinti következő kérdés, a gombokkal meg aktiválódnak újra az onclick események és kattinthatóvá válnak a gombok.

* rosszvalasz

Ez a metódus csak visszajelzést ad a felhasználónak azzal kapcsolatban, hogy rossz választ adott a játékos. Tartalmaz egy vibrator-t, meghívja a disablebuttons metódust, lejátszik egy hangot, egy toast felhasználásával tájékoztatja a felhasználót a jó válaszról, majd meghívom a késleltető metódust, ami 300 millisecundummal késlelteti az ujkerdes és a gombok metódus meghívását.

A vibrálás használatához a manifest file-ban engedélyezni kell a jogosultságot, hogy a telefon rezgő motorját használni tudjam. Ez a sor tudja ezt aktiválni:

<uses-permission android:name="android.permission.VIBRATE" />

* ujkerdes

Ebben a metódusban kérem le az adatbázisból a kérdéseket és megszámolom, hogy mennyi kérdés van a kiválasztott kategóriában, amit letárolok egy lokális változóban és egyben ki is íratom a képernyőre, hogy mennyi kérdés van és jelenleg mennyi kérdést válaszolt már meg a játékos. Minden új kérdésnél meghívódik ez a metódus. A régi kérdést, ami már volt, azt eltárolom egy regikerdes nevű listában, így minden alkalommal megnézem, hogy volt e már a kérdés. Ha szerepel az új kérdés a regikerdes nevű listában, akkor új kérdést választ. Amennyiben a kerdesekszama változó értéke eléri a lekért kérdések számát, a program kiírja, hogy nincs több kérdés és megjelenít egy AlertDialogBox-ot melyből 2 gombbal lehet navigálni. Az egyikkel új kérdés kategóriát lehet választani a másikkal elmenteni a jelenlegi pontokat és kilépni.

* categorieshelper

Ez a metódus kéri le az előző layout-on lementett sql parancs részletet a kérdések lekérdezéséhez az adatbázisból. Ezen kívül le ellenőrzőm, hogy melyik kategóriával egyenlő az sql parancs és beállítom a megfelelő háttérképet az adott kategóriának mivel minden egyes kategória saját háttérképpel rendelkezik.

* databasetolist

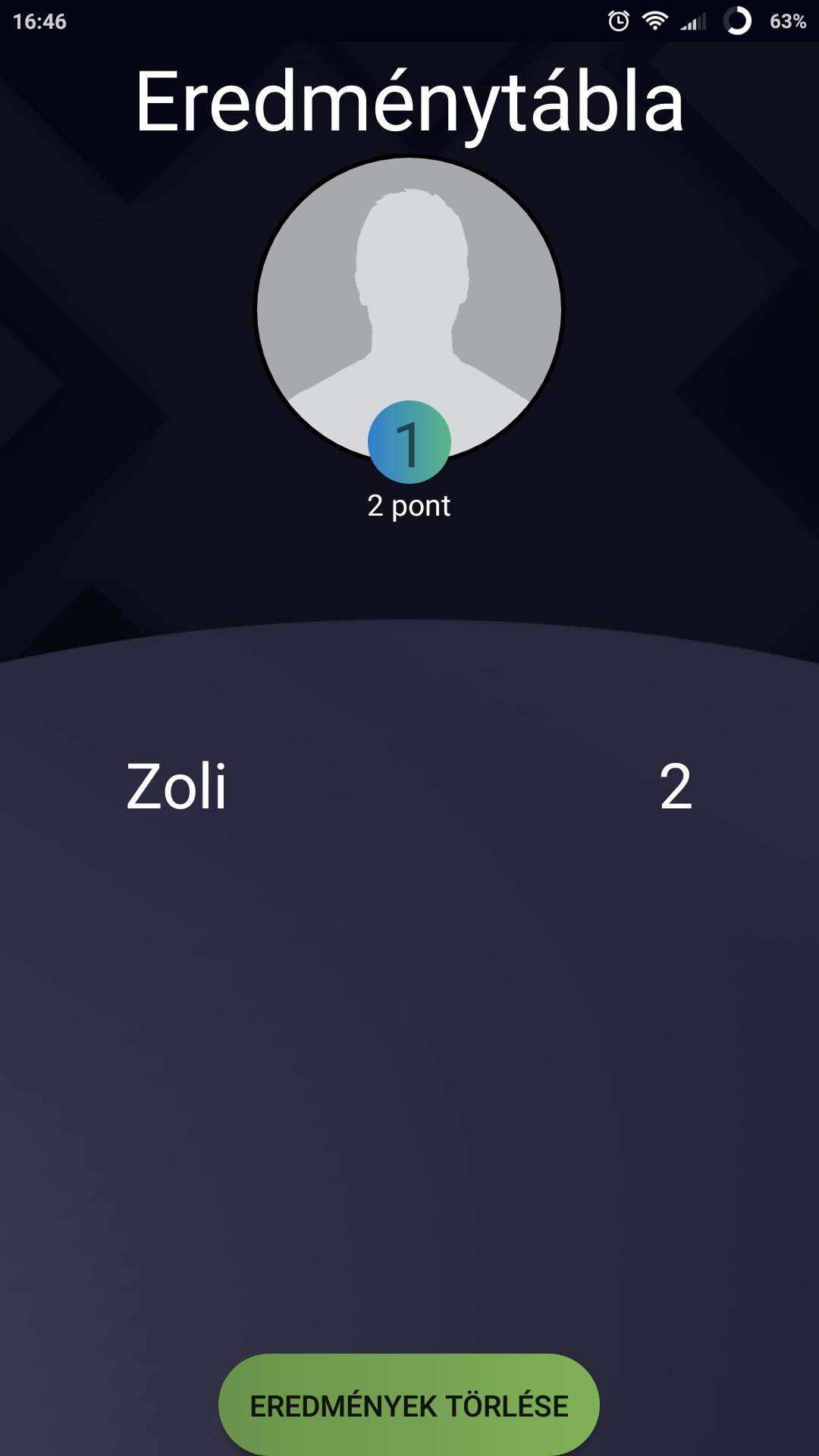
Annak érdekében, hogy ne egy megadott sorrendben jelenjenek meg a kérdések, véletlenszerűen jelenítem meg őket. Ebben a metódusban lekérem a kérdéseket és válaszokat, melyeket listákba mentek. A kérdések betöltésekor ezekből a listákból hívom be az információkat.  
Az adatbázis elemek között Cursor segítségével váltok. A moveToFirst() metódus meghívásával a cursor a legelső elemre ugrik a következőkre meg a moveToNext() metódussal lépek egy for ciklus segítségével. A ciklusba lépés előtt a getCount() metódussal megszámoltam, hogy a ciklusnak hányszor kell lefutnia.

mDb = mDBHelper.getWritableDatabase();  
 this.c = mDb.rawQuery("SELECT " + this.kerdesselect\_sql, null);  
 jelenkerdesek = c.getCount();  
c.moveToFirst();  
for(int i=0; i < this.jelenkerdesek; i++)  
{  
 this.idIndex = c.getColumnIndex("id");  
 this.kerdesIndex = c.getColumnIndex("kerdes");  
 this.valasz1index = c.getColumnIndex("valasz1");  
 this.valasz2index = c.getColumnIndex("valasz2");  
 this.valasz3index = c.getColumnIndex("valasz3");  
 this.valasz4index = c.getColumnIndex("valasz4");  
 kerdes.add(c.getString(kerdesIndex));  
 valasz1.add(c.getString(valasz1index));  
 valasz2.add(c.getString(valasz2index));  
 valasz3.add(c.getString(valasz3index));  
 valasz4.add(c.getString(valasz4index));  
  
 c.moveToNext();  
}

* disablebuttons

Ez a metódus üres onClick eseményeket tartalmaz, melyet akkor hívok meg, ha meg kell szüntetni a gombok kattinthatóságát. Jelen esetben minden jó és rossz válasz adásakor megszűntetem az onclick eseményeket, mivel a hang lejátszásakor szüneteltetem a játék menetét és ha a gombok kattinthatóak maradnának a késleltetést követően azonnal érzékeli a játékos gombnyomását.

### Scoreboard



Ezen az activity-n egy a MainMenu-ben is használt CircleImageView-t használok az első helyezett képének a megjelenítésére, a többi elért eredményt egy táblázatban jelenítem meg, melynek cellái TextView-t tartalmaznak.

Metódusok:

* onCreate

Ebben a metódusban a clearscore id-val rendelkező gomb onClick eseményét határoztam meg. Egy execSQL metódus használatával törli a Scoreboard adatbázis Scoreboard tábla minden elemét, a hozzá tartozó listákat tisztítja ki és CircleImageView kép forrását megváltoztatja a drawable mappában lévő profile\_placeholder-re. Ezután frissíti a táblázatban lévő elemeket a listtotable metódus meghívásával, ami teljesen kiüríti a táblázatot, mivel nincs érték az adatbázisban.

* databasetolist

Ez a metódus a MainActivityben is használt módszert követve az adatbázisból listába menti az eredményeket, melyeket később megjelenít a táblázatban, illetve az első helyezettet a képnél.

* first\_place

Ez a metódus csak akkor fut le, amennyiben a kép mérete nagyobb, mint nulla, vagyis van első helyezett. Az adatbázisban lévő Base64-es formátumú képet visszaalakítja bitmap kép formátumra majd beállítja a CircleImage-be bitmap képet és az azonos azonosítójú pontok\_szama listában lévő pontot a kép alatti TextView-ba állítja be.

* listtotable

Ez a metódus a korábban listába mentett pontokat és a hozzá tartozó neveket állítja be a táblázatban lévő TextView-kba.

### Information

### Általános elemek

* Butterknife

Ennek a használatához a gradle fájl dependencies részét ki kell egészíteni az alábbi 2 sorral:

implementation 'com.jakewharton:butterknife:8.8.1'  
annotationProcessor 'com.jakewharton:butterknife-compiler:8.8.1'

Ennek a használata lehetővé teszi, hogy a layouton lévő elemek változókhoz való hozzárendelése egy sor legyen. Így sokkal átláthatóbb lesz a kód és kevesebbet kell gépelni új activity létrehozásakor. De a gradle fájl kiegészítése még nem elég, mert minden egyes Activity-n amelyiken használni szeretnénk a ButterKnife-ot az onCreate metódusba meg kell hívnunk a Butterknife.bind metódust a setContentView metódus után

setContentView(R.layout.*activity\_main*);  
ButterKnife.*bind*(this);

Amennyiben ez megvan, az elemek deklarálásakor hozzá lehet rendelni a layouton lévő elemeket, mely így néz ki:

@BindView(R.id.*tv\_kerdes*) TextView tv\_kerdes;

A @BindView után a zárójelben megadjuk, hogy hol és milyen néven található az elem, utána a típusát, majd az elem nevét mellyel hivatkozunk majd rá.

* Circleimageview

Ezt az elemet a MainMenu és a Scoreboard Activity-n használom, ebben jelenítem meg a felhasználó által kiválasztott profilképet vagy amennyiben nem választott ki képet akkor a placeholder képet, mely alapértelmezetten jelenik meg.

Használatához szintén a gradle fájl dependencies részét szükséges kiegészíteni egy sorral:

implementation 'de.hdodenhof:circleimageview:2.2.0'

Ezek után ez az elem megtalálható a layout elemek között és hivatkozni is lehet rá a java fájlból ugyanúgy, mint az alapértelmezett eszközkészletre.

## Tesztelési dokumentáció

A fejlesztés során, több eszközön is teszteltem a programot, de régebbi eszközökön 2 vagy kevesebb rammal a program nem tudott megfelelően lefutni, ezt a betöltendő képek méretének és minőségének csökkentésével részben sikerült megoldani, de ez még nem esztétikus mivel az újabb telefonokon rossz minőségben jelentek meg a képek.

A Teszteléshez használt fő eszközök és tulajdonságaik:

* Xiaomi MI 5s Plus
  + Ram: 4gb
  + Processzor: Qualcomm Snapdragon 821
  + Android verzió: MIUI 9 – android 7.0
* Samsung Galaxy s2
  + Ram: 1gb
  + Processzor: ARM Cortex-A9
  + Android verzió: Lineage OS – android 7.1.2
* Samsung Galaxy s5
  + Ram: 2gb
  + Processzor: Snapdragon 801
  + Android verzió: TouchWiz UI – android 6.0.1

A játék létrehozásának elején még az volt az ötletem hogy közvetlenül az adatbázisból hívom meg a kérdéseket és válaszokat. Ám ez a módszer nagyban megnehezítette a véletlenszerű behívását a kérdéseknek. Ezért az összes kérdést, ami a kategóriában szerepel, több lokális listába betöltöm a layout betöltésekor. Ezt követően már a listákat használom az adatok megjelenítésére, így sokkal egyszerűbb a véletlenszerű megjelenítést biztosítani és a duplikátumokat kiszűrni mivel így a lista elemei 0-tól a lista méretéig nyúlik. Az adatbázisból való behívással csak az elemek id-át tudtam volna nézni, de ez nagyon lassította volna a programot is.

A profilkép elmentése több szempontból is nehéznek bizonyult. Először arra kerestem megoldást, hogy milyen módszerrel tudnám a nyers képet az alkalmazás mappájába másolni, de ez feleslegesen foglalta volna a felhasználó tárhelyét. Ezen megoldás helyett az adatbázisban és SharedPreference-ben tárolom le a profilképet. Először átkonvertálom Base64 alapú String-re a képet, melyet így egyszerűen tudok tárolni az adatbázisban szövegként. Amikor használni szeretném ezt a képet, csak vissza kell alakítanom szöveg típusról bitmap képre, amit az ImageView vagy a CircleImageView már egyszerűen meg tud jeleníteni képként.

A játék közben a válaszadáskor hang játszódik le és ezért az új kérdés behívását késleltetem. Ennek következményeként a gombok kattinthatóak maradnak és a késleltetés lejárta után az összes kattintást egyszerre érzékeli. Kiküszöbölésére létrehoztam egy metódust ami üres onClick action-öket hoz létre az összes layouton lévő gombra. Így kattintható marad a gomb, de nem lesz semmi funkciója. A késleltetés lejárta után újra meghívom azt a metódust ami a helyes onClick funkciókat tartalmazza.

## Továbbfejlesztési lehetőségek

# Forrásjegyzék

# Ábrajegyzék