XXI. reál- és humántudományi Erdélyi Tudományos Diákköri Konferencia (ETDK)
Kolozsvár, 2018. május 24–27.

Legendárium Nagyappó

Kiterjesztett valóságra épülő mobilalkalmazás

Szerzők:

Döngölő Zsolt
Babeş–Bolyai Tudományegyetem, Matematika és Informatika Kar, Informatika szak, II. év
Kovács Kinga
Babeş–Bolyai Tudományegyetem, Matematika és Informatika Kar, Vállalati szoftvertervezés és fejlesztés szak, II. év

Témavezetők:

dr. Simon Károly, egyetemi adjunktus
Babeş–Bolyai Tudományegyetem, Matematika és Informatika Kar
Szécsi Zsolt, szoftverfejlesztő,
Codespring

CODESPRING
OUR CODE YOUR SUCCESS

Kivonat

A dolgozat bemutatja a *Nagyappó* nevű Android-os mobilalkalmazás szerkezetét és működését, valamint annak megvalósítási hátterét.

A projekt célja egy olyan alkalmazás létrehozása, amely képfelismerés alapján képes automatikusan beazonosítani a legendákat a *Székelyföldi Legendárium* című könyvből és azokat „felolvasni". Hogy még érdekesebb, színesebb legyen a felolvasás, az *Augmented Reality* eszközeit felhasználva a legenda felolvasásakor a telefon képernyőjén a kiterjesztett valóság részeként megjelenik a Legendáriumból ismert nagyapó, aki besétál, majd letelepedik a könyv lapjára, és „elmeséli" azt a legendát, amelyre éppen a telefon kamerája irányítva van. Az alkalmazás főként kisgyerekek számára készül, akik ennek segítségével akár már azelőtt megismerhetik Erdély legendáit, mielőtt olvasni tudnának. Ha valaki nem rendelkezik a *Székelyföldi Legendárium* könyvvel, akkor is meghallgathatja az elérhető tartalmakat: a *Keresés* funkcionalitás segítségével lehetőség van a legendák közötti keresésre is.

A fejlesztéshez szükséges médiaanyagokat (animációk, hangoskönyvek, képek) a *Székelyföldi Legendárium* csapata szolgáltatta.

Tartalomjegyzék

Bevezető

A *Székelyföldi Legendárium*[1] egy 2008-ban indult projekt, amely ezidáig 156 legendát fog össze. Térképek, könyvek, hangoskönyvek, kifestősök, színes ceruzák, dokumentumfilm, társas játékok és még sok minden más jelent meg általa, valamint ennek a projektnek a keretein belül indították el az első székelyföldi rajzfilmstúdiót, melynek köszönhetően egy 3D-s rajzfilmsorozat több része is megjelent.

A dolgozat a *Legendárium Nagyappó* alkalmazást mutatja be, amely a *Székelyföldi Legendárium* és a U-Hub[2] csapat közös ötletén alapszik. Mivel sok olyan kisgyerek van, aki rajong a legendákért, azonban még nem tud olvasni, felmerült, hogy jó lenne készíteni számukra egy interaktív Legendárium mesekönyvet, amely „felolvassa" nekik a legendákat tartalmazó könyv oldalait. A kiterjesztett valóságra épülő Android mobilalkalmazás kameráján keresztül nézve a mesekönyvet, látjuk, ahogy a Legendáriumból ismert nagyapó besétál és leül a könyv lapjára, majd „elmeséli" azt a legendát, amelyikre a kamera irányítva van.

Az alkalmazás két nagyobb részre osztható: a képfelismerést és animációk lejátszását megvalósító részre, amely a Vuforia keretrendszer és a Unity fejlesztési platform segítségével van kivitelezve, valamint a mobilalkalmazás felületét és egyéb plusz funkcionalitásokat magába foglaló részre, amely Android szoftverfejlesztői csomag segítségével van létrehozva. A mobilalkalmazás fő funkcionalitása a *„Felolvasás"*, amelyhez szükséges a képfelismerés, az animációk lejátszása és természetesen a legendákat tartalmazó *Székelyföldi legendárium* című könyv. Az alkalmazás egy másik funkcionalitása a *Keresés*, amely által megtekinthetőek az elérhető tartalmak, lehetőség van az ezek közötti keresésre, illetve meghallgathatóak a legendákhoz társított hangoskönyvek. A felhasználó további információkhoz juthat *Az alkalmazásról* menüpont kiválasztása után, ahol elérheti a Székelyföldi Legendárium projekt további termékeit, valamint megtekintheti a fejlesztés során felhasznált technológiák listáját és licenszfeltételeit.

A dolgozat hátralevő része a következőképpen van strukturálva: az 1. fejezet a mobilalkalmazás alapvető funkcionalitásait részletezi, majd ezt követi a 2. fejezet, amely a felhasznált technológiákat mutatja be, érintve a Vuforia keretrendszert, a Unity fejlesztési platformot és az Android platformot. A 3. fejezetben a mobilalkalmazás megvalósítása van részletezve, melyet a 4. fejezetben a fejlesztés során felhasznált eszközök bemutatása követ. Az alkalmazás működését az 5. fejezet taglalja, és végül az utolsó fejezet a következtetéseket, valamint a továbbfejlesztési lehetőségeket foglalja magába.

Az alkalmazás fejlesztése a U-Hub Mentorprogram keretein belül indult 2017 nyarán. A fejlesztés során mentorként segítségünkre volt dr. Simon Károly és Szécsi Zsolt a Codespring részéről, valamint a Székelyföldi Legendárium csapata, akik a médiaanyagok szolgáltatásával

[1] http://www.legendarium.ro/rolunk
[2] https://u-hub.ro/mentorprogram/

hozzájárultak az alkalmazás megvalósításához, valamint segítettek a koncepció kialakításában. Az alkalmazás be volt mutatva a Sepsiszentgyörgyön megszervezett IT és Innovációs konferencián, majd ezt követően az M1 televízió hírcsatornáján, valamint Csíkszeredában a Karácsonyi IT Meetup-on.

1. A Legendárium Nagyappó alkalmazás

1.1. Alapvető funkcionalitások

A **Legendárium Nagyappó mobilalkalmazás** rendelkezik egy felhasználóbarát felülettel, amely a következő funkcionalitásokat biztosítja:

- felismeri a „Székelyföldi Legendárium” könyvben található oldalakat a telefon kamerájának segítségével;
- a felismert oldalhoz társítja a neki megfelelő hangoskönyvet, és lejátszodja azt;
- az illető legenda elindulásakor rávetít egy 3D-s modellt a könyv lapjára, amelyet a telefon kameráján keresztül láthatunk, és úgy körbeforgathatjuk körülötte a kamerát, mintha az valóságos elem lenne az oldalon;
- a megjelenített 3D-s modellt animálja;
- lehetőséget nyújt a legendák meghallgatására a könyv nélkül is, ebben az esetben kereshetünk a legendák között név, illetve azonosító alapján.

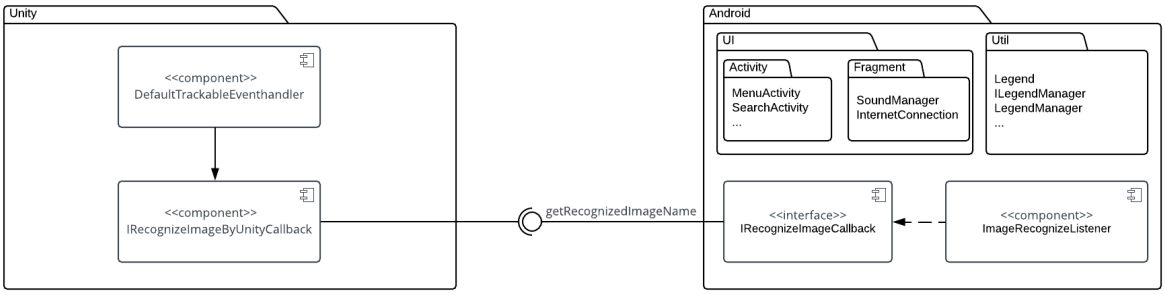
1.2. Architektúra

Az alkalmazást két fő részre bonthatjuk szerkezetileg: a Unity részre, amely a 3D-s modell megjelenítéséért és animálásáért felel, valamint az Android részre, amely a hangoskönyvek lejátszását biztosítja az animáció lejátszásával párhuzamosan. Az alábbi, 1. ábrán látható komponens diagram szemlélteti a Unity és Android projektek közötti kommunikáció megvalósítását. Az Android alkalmazásban létrehozott *IRecognizeImageCallback* interfészt a Unity projektben lévő *IRecognizeImageByUnityCallback* osztály implementálja, amelyet a *DefaultTrackableEventHandler* osztály példányosít, majd átadja neki a felismert kép nevét. Az Android projektben jelen lévő *ImageRecognizeListener* osztály az interfészen keresztül *callback* mechanizmus segítségével lekéri a Unity által lementett képnevet.

A *callback* mechanizmus megvalósítását Unity oldalon az alábbi, 1. kódrészlet szemlélteti, amely a *DefaultTrackableEventHandler C#* szkript egy része.

```
callback.onTrackingFound (mTrackableBehaviour.TrackableName);
imageTarget = callback.getRecognizedImageName();
AndroidJavaObject imageRecognizeListener =
   new AndroidJavaObject("com.legendarium.nagyappo.ImageRecognizeListener");
imageRecognizeListener.Call("onImageRecognize", callback);
```

1. kódrészlet. Példa a felismert kép nevének elküldésére a C# kódból egy Java osztálynak



1. ábra. UML komponens diagram - kommunikáció a Unity és Android projektek között

Ezek után a Java kódban már csak fogadni kell az elküldött információt és feldolgozni azt. Ennek megvalósítása a 2. kódrészletben van szemléltetve.

```
String imageName = callback.getRecognizedImageName();
Log.d("PluginClass", "Returned value: " + imageName);
legendUrl = legendManager.getLegendByImageName(imageName);
```

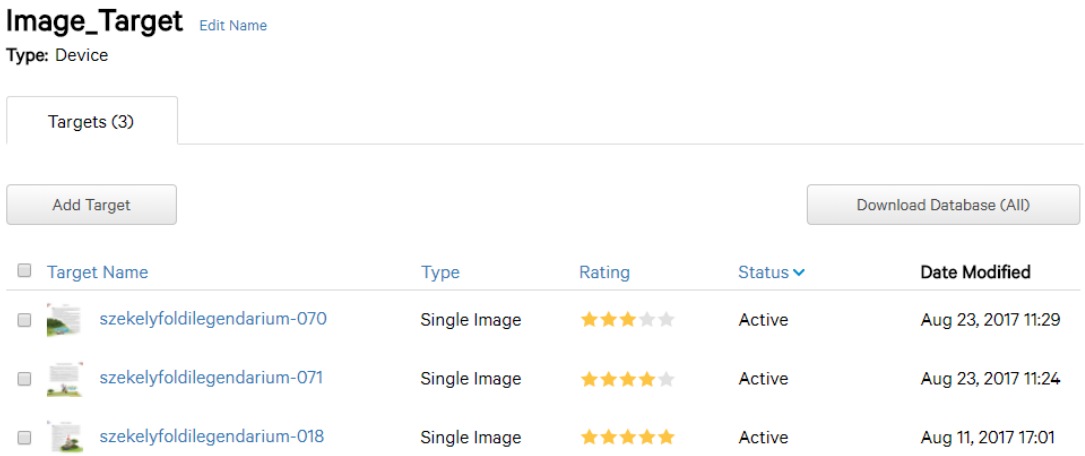
2. kódrészlet. Üzenet feldolgozása és a megfelelő legenda adatainak lekérése json fájlból

Ezek mellett az Android alkalmazás egy külső szerverrel is kommunikál, amely lehetővé teszi a hanganyagok betöltését az internetről. A legendák tartalma hangoskönyv formájában van tárolva a *https://www.indiesound.com* oldalon, ahonnan a megfelelő, lejátszáshoz szükséges példányt az alkalmazás *stream*-eli.

2. Felhasznált technológiák

2.1. Vuforia keretrendszer

A *Vuforia* [11] a legnépszerűbb kiterjesztett valóság (Augmented Reality) platform. Legfőbb előnye, hogy támogatja a népszerű telefonokat és tableteket, amelyek Android vagy iOS operációs rendszerrel rendelkeznek. A platform egyik legfontosabb szerepe az, hogy felismerjen képeket, objektumokat és környezeteket. Lehetőséget ad például épületek, tárgyak felismerésére és a felismert objektumokhoz információk, virtuális tartalmak rendelhetőek. Ez eredményezi a kiterjesztett valóságot: a valós elemek virtuális elemekkel ötvöződnek. A *Nagyappó* alkalmazásban a virtuális elem a nagyapó mesefigurája, a valóságos környezetet a *Székelyföldi Legendárium* könyv oldalai szolgáltatják. A Vuforia rendelkezik egy adatbázissal (2. ábrán látható), amely elérhető a fejlesztők számára. Ide lehetőség van feltölteni képeket, majd ezeket összetömörítve letölteni olyan formátumban, amit *Unity-be* lehet importálni, és 3D-s modellekkel lehet társítani. Ide voltak feltöltve a Legendárium könyv oldalai is, kép formátumban.

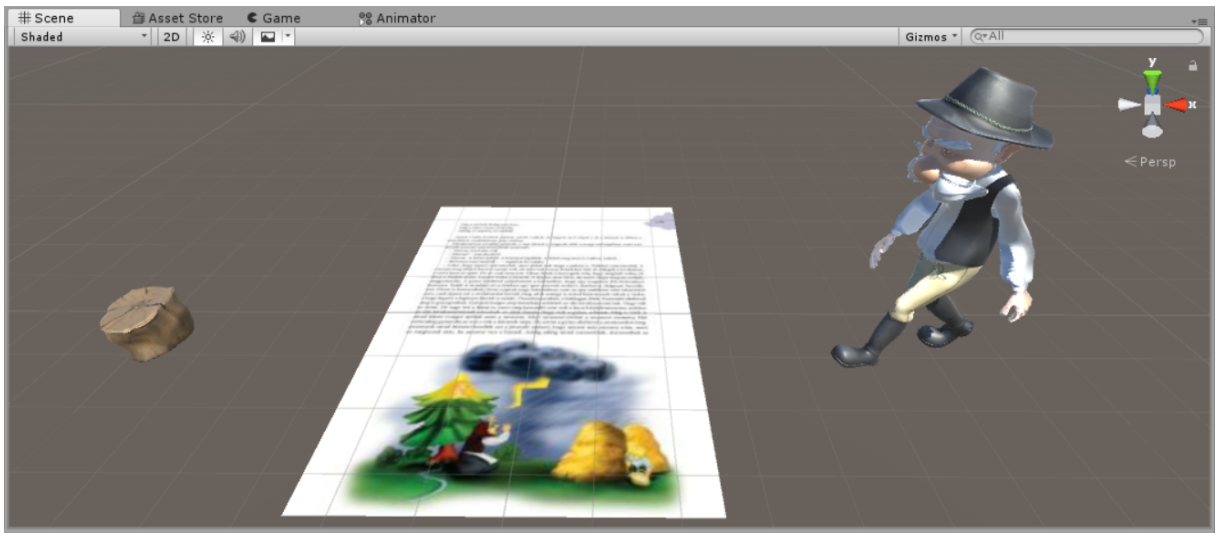


2. ábra. Vuforia adatbázis a *Székelyföldi Legendárium* könyv oldalaival, és a képminőségek értékelésével

2.2. Unity fejlesztési platform

A *Unity* [14] az egyik legnépszerűbb játékfejlesztési platform, amely kiváló minőségű 2D, 3D, VR (Virtual Reality) és AR (Augmented Reality) játékok megalkotását teszi lehetővé, amelyek több platformra kitelepíthetőek, beleértve mobileszközöket, játékkonzolokat, asztali számítógépeket, beágyazott rendszereket stb. [8] A *Unity* erős grafikus motorral rendelkezik és egy teljes funkcionalitású szerkesztővel, a *Unity Editor*-ral, amely egyesíti a vizuális tervezőeszközöket a kódírással. Ez nagyban megkönnyíti a szoftverfejlesztők és grafikusok számára a nagyobb projekteken való együttműködést [13]. A vizuális részhez tartozó grafikus tartalmakat a *Unity*

Editor segítségével tudjuk megalkotni, amelyen belül úgynevezett *scene*-ek hozhatóak létre. Ezen belül hozzáadhatóak vagy létrehozhatóak 2D-s és 3D-s modellek, amelyeket tetszés szerint lehet animálni. A 3. ábrán egy *scene* látható, amely tartalmazza a *Székelyföldi Legendárium* könyv egy oldalát, valamint 3D-s modelleket (bal oldalon egy csutak, jobb oldalon a Legendárium rajzfilmsorozatából ismert nagyapó), amelyekhez animáció is tartozik.



3. ábra. Egy *scene* kinézete Unity Editor-ban

A vizuális részt képező elemekhez *C#* szkripteket lehet hozzárendelni, ahol kódszinten vezérelhető ezeknek a működése (például az animációk vezérlése).

2.3. Android SDK

Az *Android SDK* könyvtárakat és fejlesztői eszközöket biztosít a szoftverfejlesztők számára Android alkalmazások létrehozásához [2]. A készlet mindent tartalmaz, amire szükség van egy alkalmazás fejlesztése során: emulátor, alapvető könyvtárak, debugger, minta forráskód, oktatóanyag az Android operációs rendszerhez, az Android API[3]-ra (alkalmazás programozási interfész) vonatkozó dokumentáció. Az emulátor lehetőséget nyújt tesztelésre emulált virtuális eszközök segítségével. Beállítása könnyen megoldható, futtatása egyszerű, és számos telefontípus elérhető rajta keresztül, illetve ezek felbontása is beállítható. A debugger segítségével a fejlesztőknek hibakeresésre van lehetőségük [6]. Az Android SDK-val együtt használható egyik legnépszerűbb fejlesztői környezet az *Android Studio* [3].

2.4. Butterknife

A Butterknife egy Android platformon alkalmazható könyvtár, amelynek szerepe, hogy kapcsolatot teremtsen az Android nézethez tartozó elemek és az adattagok vagy metódusok között. En-

[3] Application Programming Interface

nek implementálásához annotációkat biztosít, amelyek használatával áttekinthetőbb, érthetőbb és könnyen módosítható, illetve hordozható kódot kapunk eredményül. Az *@OnClick* annotáció segítségével elkerülhető egy új belső névtelen osztály létrehozása, így rövidebbé, átláthatóbbá válik a kód. Továbbá, a *findViewById* metódushívásokat ki lehet küszöbölni, a nézetre vonatkozó adattagokra alkalmazott *@BindView* annotációval. A Butterknife könyvtár külső függőségként van jelen a Legendárium Nagyappó alkalmazásban, a Gradle build konfigurációjában van hozzáadva a projekthez.

```
@OnClick(R.id.search)
    public void search() {
        Intent intent = new Intent(MenuActivity.this, SearchActivity.class);
        startActivity(intent);
    }
```

3. kódrészlet. Név nélküli belső osztály kiküszöbölése az *@OnClick* annotációval

3. A mobilalkalmazás megvalósításának részletei

A mobilalkalmazás megvalósítási folyamatát két nagyobb részre oszthatjuk: a képfelismerés megvalósítása, majd azt követően az animációk lejátszása, illetve az Android mobilalkalmazás fejlesztési folyamata, amely magába foglal más, a kiterjesztett valósághoz nem kapcsolódó funkcionalitásokat is.

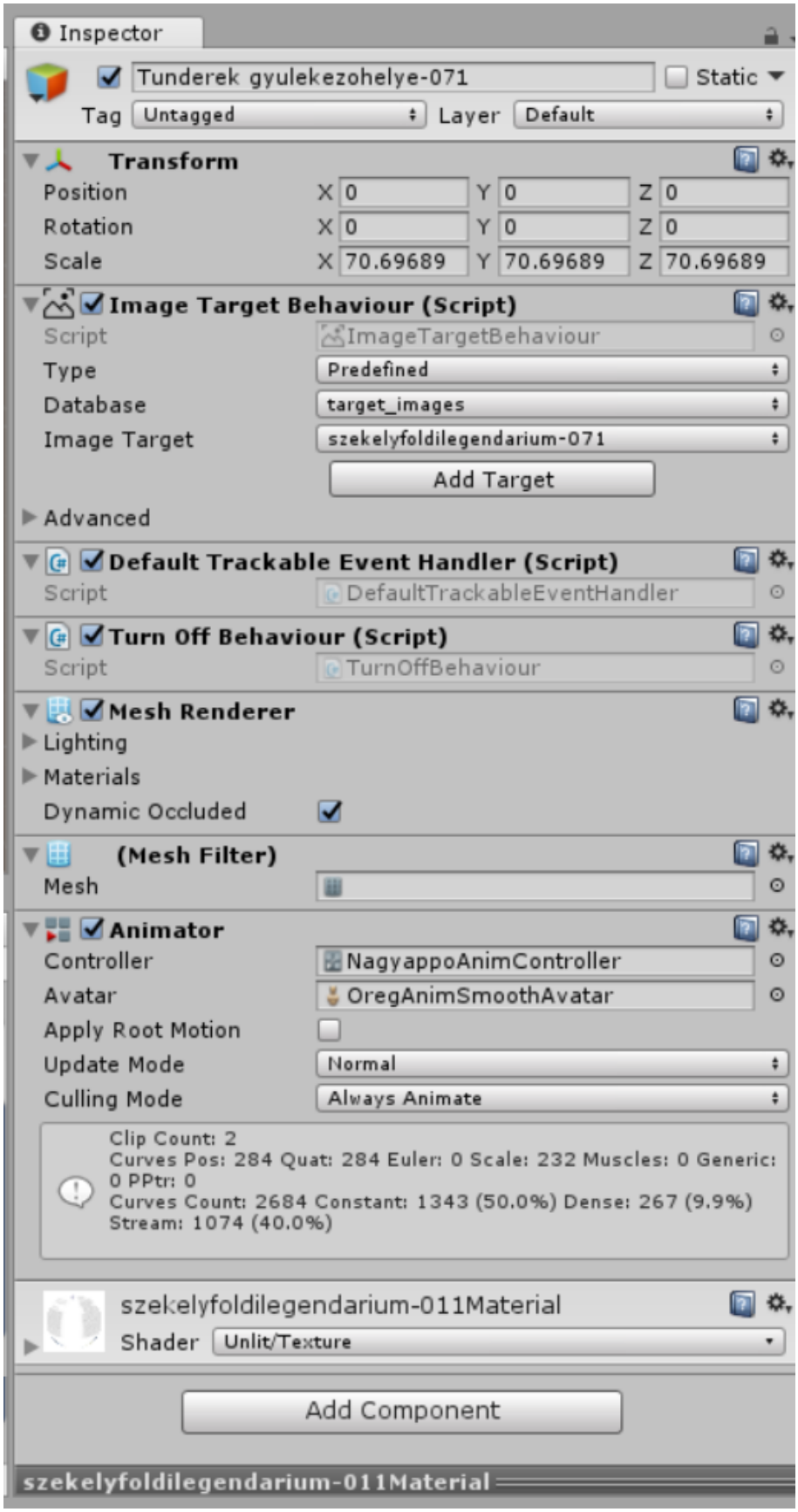
3.1. A képfelismerés és az animációk megvalósítása

A *Székelyföldi Legendárium* könyv tartalmának felismerése a *Vuforia* keretrendszer által történik. Első lépésként a könyv oldalai képek formájában le lettek mentve a Vuforia adatbázisába. Ehhez az adatbázishoz a Vuforia fejlesztői portálon[4] keresztül lehet hozzáférni. Az előfeldolgozást követően az adatbázisból a tartalom a Unity fejlesztési platformnak megfelelő formátumban lett letöltve, melynek következtében létrejött egy *target_images.unitypackage* állomány. (Ezen kívül lehetőség van *Android Studio*, *Xcode* vagy *Visual Studio* fejlesztési platformokkal kompatibilis formátumban is letölteni az adatbázis tartalmát.) A letöltött állomány Unity Editor segítségével volt beimportálva a Unity projektbe, ahonnan már a projekten belül is elérhetővé vált az adatbázis tartalma, vagyis a legendákat tartalmazó könyv oldalai.

A Unity projekt esetében első lépésként egy *Nagyappo* nevű *scene* lett létrehozva, amelyhez hozzáadtuk az *Assets/Prefabs* könyvtárszerkezet alatt található *ARCamera*-t, amely a képfelismeréshez szükséges. Ezután volt létrehozva egy *ImageTarget* panel, amelyet az *Inspector* fül alatt lehet konfigurálni, ahol számos tulajdonság testreszabható. A 4. ábrán láthatóak a *Tunderek gyulekezohelye-071* nevű *ImageTarget* beállításai, valamint a hozzárendelt szkriptek és komponensek. Először a pozíciója, elforgatásának mértéke és a komponens mérete van meghatározva *x*, *y* és *z* tengelyek szerint. Ezután következik a panel hátterének beállítása, amelyhez a *target_images* adatbázisból veszünk egy képet (*szekelyfoldilegendarium-071*). A vezérléséhez különböző *C#* szkriptek vannak hozzárendelve (*DefaultTrackableEventHandler*, *TurnOffBehaviour*). Az animátor kontroller szintén hozzá van adva az *ImageTarget*-hez, amely tartalmaz két animációt is (*„Clip Count: 2”*). Az *Inspector* alján található *Add Component* gombra kattintva tetszés szerint újabb komponensek is hozzáadhatóak.

Az egyik legfontosabb beállítás a panel esetében a tartalmának a meghatározása, amely esetünkben a legendákat tartalmazó könyv egy oldala, ugyanis innen már lehetőség van a beimportált adatbázisból kiválasztani a nekünk szükséges képet. Ahhoz, hogy az adatbázis ténylegesen elérhető legyen, először a *Window* menüpont alatt található *Vuforia Configuration*-t kell kiválasztani, melynek hatására megjelenik egy újabb *Inspector* fül, ahol a *Datasets* lenyíló elem

[4] Vuforia fejlesztői portál - adatbázis: https://developer.vuforia.com/targetmanager/project/deviceTargetListing (az oldal eléréséhez regisztráció és bejelentkezés szükséges)



4. ábra. Unity Editor - *Inspector* fül, ahol számos beállítást meghatározhatunk egy kiválasztott komponens esetében

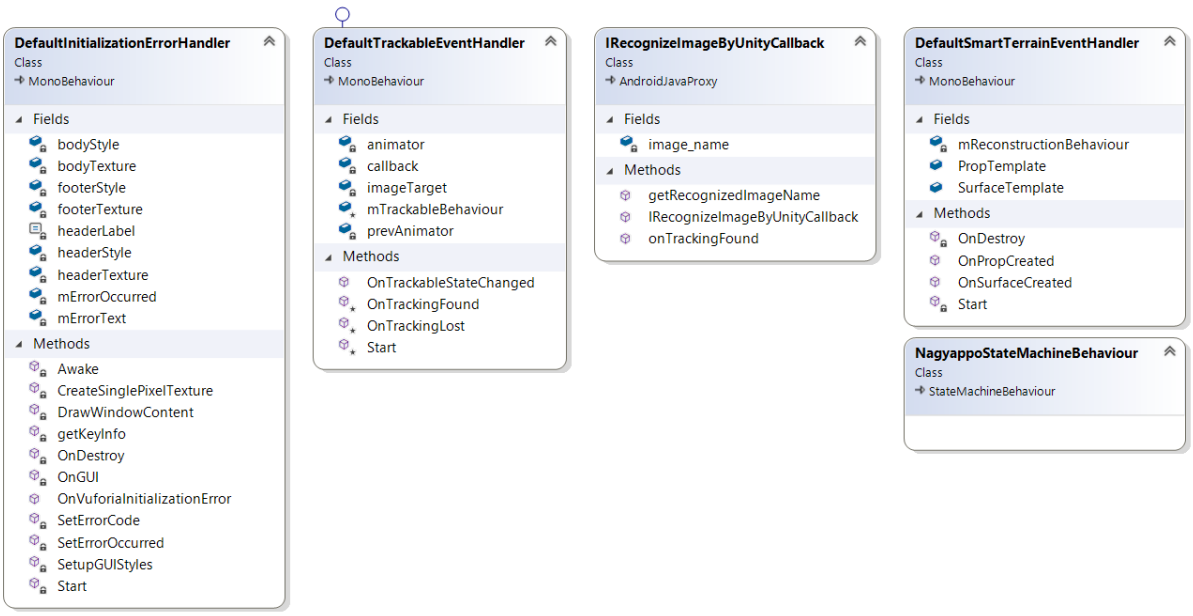
alatt ki lehet jelölni a *Load target_images Database*, illetve az *Activate* jelölőnégyzeteket. Ettől kezdve már elérhetőek és felhasználhatóak lesznek a *Unity* projekten belül a Vuforia adatbázisában lévő legendákkal kapcsolatos képek.

Az *ImageTarget* panelt annyi példányban volt szükséges létrehozni, ahány kép felismerésére volt szükség, ami azt jelenti, hogy a projekten belül a *Székelyföldi Legendárium* köny oldalai egy-egy *ImageTarget*-ként jelennek meg (ez számszerint 27-et jelent, mivel egyelőre 20 legendát ismer fel a rendszer, melyekhez összesen 27 oldal tartozik).

A létrehozott *ImageTarget*-ekhez hozzá lettek rendelve a Legendárium csapata által létrehozott 3D-s modellek (a Legendárium rajzfilmsorozatból ismert mesefigura, azaz nagyapó, illetve egy favágó csutak), valamint az azokhoz tartozó animációk.

Az alkalmazás működését különböző *C#* szkriptek biztosítják, amelyek a projekt létrehozásakor voltak generálva. Az animációk vezérlése a *DefaultTrackableEventHandler.cs* szkripten belül van implementálva, amely hozzá van rendelve az összes *ImageTarget*-hez.

Az alább látható 5. ábra a Unity projekt osztályait ábrázolja.



5. ábra. A Unity projekt osztályai

A *DefaultInitializationErrorHandler*, *DefaultSmartTerrainEventHandler*, *NagyappoStateMachineBehaviour*, illetve *DefaultTrackableEventHandler* osztályokat a Unity generálta, ezek szerves részét képezik a projektnek. Az *IRecognizeImageByUnityCallback* osztály az *IRecognizeImageCallback* interfészt implementálja, amely az Android projekt (lásd utolsó bekezdés) részét képezi, megteremtve így a két projekt közötti kapcsolatot. Az interfész implementációja a 4. kódrészletben látható.

A konstruktor implementációjában látható az Android projektben létrehozott interfész elérési útvonala (*"com.legendarium.nagyappo.IRecognizeImageCallback"*). Az *onTrackingFound()*

```
using UnityEngine;

public class IRecognizeImageByUnityCallback : AndroidJavaProxy
{
    private string image_name;

    public IRecognizeImageByUnityCallback() :
                    base("com.legendarium.nagyappo.IRecognizeImageCallback")
    {
    }

    public void onTrackingFound(string trackedFileName)
    {
        this.image_name = trackedFileName;
    }

    public string getRecognizedImageName()
    {
        return image_name;
    }
}
```

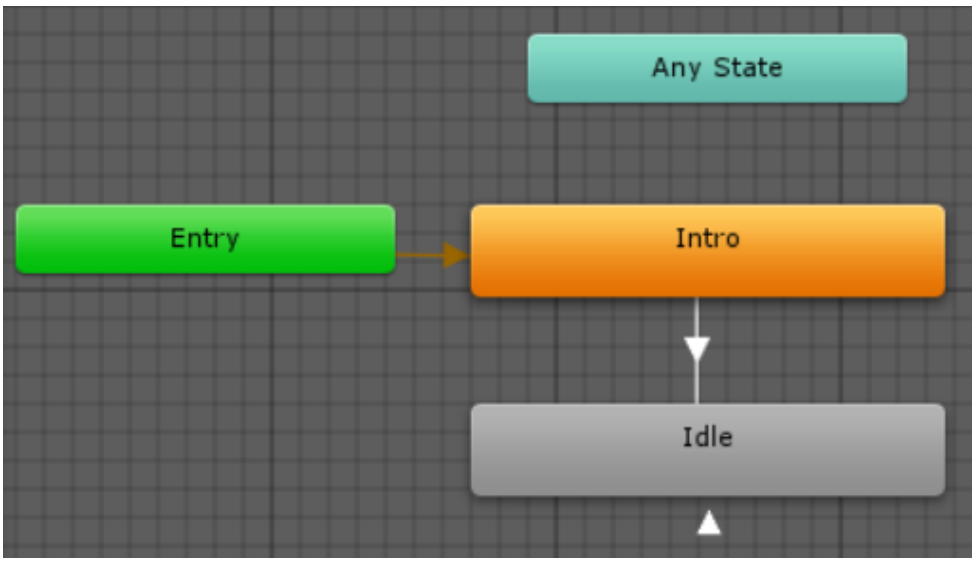
4. kódrészlet. Az *IRecognizeImageByUnityCallback* osztály tartalma, amely implementálja az Android alkalmazásban lévő *IRecognizeImageCallback* interfészt

metódus a *DefaultTrackableEventHandler* osztályban hívódik meg a képfelismerés után, melynek következtében az *image_name* adattag az aktuálisan felismert kép nevét fogja tartalmazni, míg a *getRecognizedImageName()* metódus az Android projektben kerül meghívásra, amely visszatéríti a felismert kép nevét.

A képfelismerés a *DefaultTrackableEventHandler* osztályban van implementálva és itt van megteremtve a konkrét kapcsolat a Unity és Android projektek között, amely egy *callback* mechanizmusra épül (lásd az 1. kódrészletet).

Az animációk vezérlése szintén a képfelismeréshez kapcsolódik, a *DefaultTrackableEventHandler* osztály tartalmazza ennek megvalósítását. Először a *Unity Editor*-ban létre volt hozva egy *NagyappoAnimController.controller* nevű animátor kontroller, amely tartalmazza az animáció klippeket, valamint az azok közötti átmeneteket, tranzíciókat. Az animátor kontroller a 6. ábrán látható módon néz ki.

Egy animátor kontroller első és kötelezően jelenlévő komponense az *Entry* komponens, amely meghatározza, hogy melyik animáció játszódjon le az animátor kontroller elindítása után. Esetünkben az *Intro* nevű animáció klipp kerül először lejátszásra, melynek hatására besétál a kamera fókuszába a nagyapó, illetve „beugrál” egy csutak, amelyen a nagyapó helyet foglal. Ez az animáció játszódik le abban az esetben, amikor a mobileszköz kamerája egy legenda oldalára fókuszál és megtörténik a képfelismerés. A következő animáció klipp az *Idle*, amely magát a



6. ábra. NagyappoAnimController.controller - tartalmazza az animáció klippeket, illetve az azok közötti tranzíciókat

felolvasás folyamatát játsza le. A nagyapó már elfoglalta a csutakon a helyét, és mesélni kezd. Az *Intro* és *Idle* komponensek közötti nyíl a két animáció közötti átmenetet jelképezi. Miután egyszer lejátszódott az első animáció (*Intro*), vagyis besétál a nagyapó, automatikusan, megszakítás nélkül indul el a mesélő nagyapó animációja (*Idle*). Az *Idle* komponens alatt látható önmagába visszamutató nyíl azt jelzi, hogy a komponenshez tartozó animáció lejátszása ismétlődik, vagyis ez az animáció a teljes legenda felolvasása alatt újra és újra lejátszásra kerül. Ez az ismétlődő folyamat akkor fejeződik be, ha kód szinten implementálva van annak leállítása, vagy ha elölről kell kezdeni az animátor kontroller tartalmának lejátszását, mert például közben egy újabb legenda felolvasását szeretné a felhasználó meghallgatni. Az animátor kontroller újraindítása a *Rebind()* metódushívással történik (lásd az 5. kódrészletet).

```
animator = GetComponent<Animator>();

if (!imageTarget.Equals(mTrackableBehaviour.TrackableName))
{
    if (prevAnimator != null)
    {
            prevAnimator.gameObject.SetActive(false);
    }

    animator.gameObject.SetActive(true);
    animator.Rebind();
}

imageTarget = mTrackableBehaviour.TrackableName;
prevAnimator = animator;
```

5. kódrészlet. Az animátor kontroller újraindítása a *Rebind()* metódushívással

Miután sikeresen fel lettek építve *Unity Editor* segítségével a képfelismerés funkcionalitáshoz szükséges komponensek, és lehetőség volt azokat a *C#* szkriptekből vezérelni, az alkalmazás ki lett telepítve egy mobileszközre, ahol a működését lehetett tesztelni.

Végül a meglévő Unity projektből a *Unity Editor File* menüpontja alatt található *Build Settings...* lehetőséget kiválasztva ki lett generálva egy Android alkalmazás, amely a Unity-ben létrehozott funkcionalitást használja fel, valamint további funkcionalitások fejlesztésére ad lehetőséget.

3.2. A mobilalkalmazás megvalósítása

Az Android alkalmazások kódbázisa több részből tevődik össze. Az első egy projektet leíró állomány, amelyet *manifest* állománynak nevezünk és XML alapú. A manifest állományban vannak tárolva a metainformációk, amelyek lehetnek például jogosultságuk, amelyekre szüksége van az alkalmazásnak (pl. internetelérés). A Nagyappó projektben többek között szükség volt arra, hogy az alkalmazás használhassa az eszköz kameráját, valamint internetelérésre, amely a hanganyagok lekéréséhez volt szükséges (lásd a 6. kódrészletben).

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.CAMERA" />
```

6. kódrészlet. Példa az *AndroidManifest.xml* állományból, amelyben jogosultságot kérünk a kamera használatára valamint az internetelérésre

Továbbá a manifest állományban meg vannak adva az operációs rendszer verziójára vonatkozó követelmények, amelyek a Gradle build során generálódnak, illetve az is, hogy az alkalmazás indításakor melyik nézet jelenjen meg a felhasználó számára. Az Android alkalmazás építőköveit *Activity*-k és *Fragment*-ek alkotják, amelyek egy képernyő nézeteit biztosítják, és amelyek XML layout állományokból konfigurálhatóak. Ennek következtében fontos szerepet játszanak az XML típusú erőforrás állományok, mivel a médiaelemek, stílusok, elrendezések ezekben vannak meghatározva. Az alkalmazásban szereplő képek (menügombok, háttérképek, legenda borítók) több méretben is megtalálhatóak, ami arra nyújt lehetőséget, hogy mind tableten, mind egy kisebb képernyővel rendelkező mobileszközön kényelmesen használható legyen az alkalmazás.

A Nagyappó mobilalkalmazás Java kódjában egy azonosítón keresztül lehet hivatkozni az előbb említett grafikus elemekre. Ezeket a *Butterknife* (2.4 fejezet) által nyújtott annotációk segítségével lehet beinjektálni a projektbe. A legfontosabb művelet, amelyet szükséges volt elvégezni az Android kódból, az a hangoskönyv lejátszása, amely nem gombnyomásra történik, hanem közvetlenül egy kép felismerésének hatására. Az előző fejezetekben említésre került,

hogy a képfelismerés Unity, illetve Vuforia keretrendszer segítségével volt megvalósítva, így a Java kódban a Unity-ben megírt C# szkripttől kellett lekérni a képazonosító információt a felismerés pillanatában. Ennek az információnak az alapján lehet eldönteni, hogy melyik hanganyagot kell lejátszani. Ezek a párosítások egy *Json* típusú fájlban vannak tárolva, és ebből olvasható ki, hogy melyik képhez melyik legenda társul, valamint a legendához tartozó hangoskönyvnek a forrása.

```
{
    "targetImages": [ "szekelyfoldilegendarium-070" ],
    "streamUrl": "https://www.indiesound.com/download.php?track=16443",
    "title": "Medve-tó",
    "coverImage": "m01",
    "subtitle": "m 01"
}
```

7. kódrészlet. Egy legenda a *Json* fájlban

Ahogy a 7. kódrészletben látható, minden legenda öt attribútummal rendelkezik. A *"targetImages"* és a *"streamUrl"* attribútumok a képfelismerésnél töltenek be fontos szerepet, ugyanis Unity oldalról a "targetImages" kap értéket, amelyet társítani lehet a "streamUrl" értékével. A hanganyagok egy külső szerverről vannak *stream*-elve, mivel az összes hangoskönyv beágyazása a projektbe nagyon tárigényes lett volna. A fent látható Json fájl alapján fel van építve egy a legendát reprezentáló Java objektum, amelynek modellje a *Legend* osztályon keresztül van megadva. Ez lényegében egy POJO[5], vagyis csak egy legenda attribútumait és azok *getter* és *setter* metódusait tartalmazza. A stream-elés aszinkron módon van végrehajtva (lásd a 8. kódrészletet), elkerülve az alkalmazás pillanatnyi „lefagyását”, a hangoskönyv betöltése alatt.

```
legendUrl = legendManager.getLegendByImageName(imageName);
UnityPlayerActivity.mediaPlayer.setDataSource(legendUrl);
UnityPlayerActivity.mediaPlayer.prepareAsync();
UnityPlayerActivity.mediaPlayer.start();
```

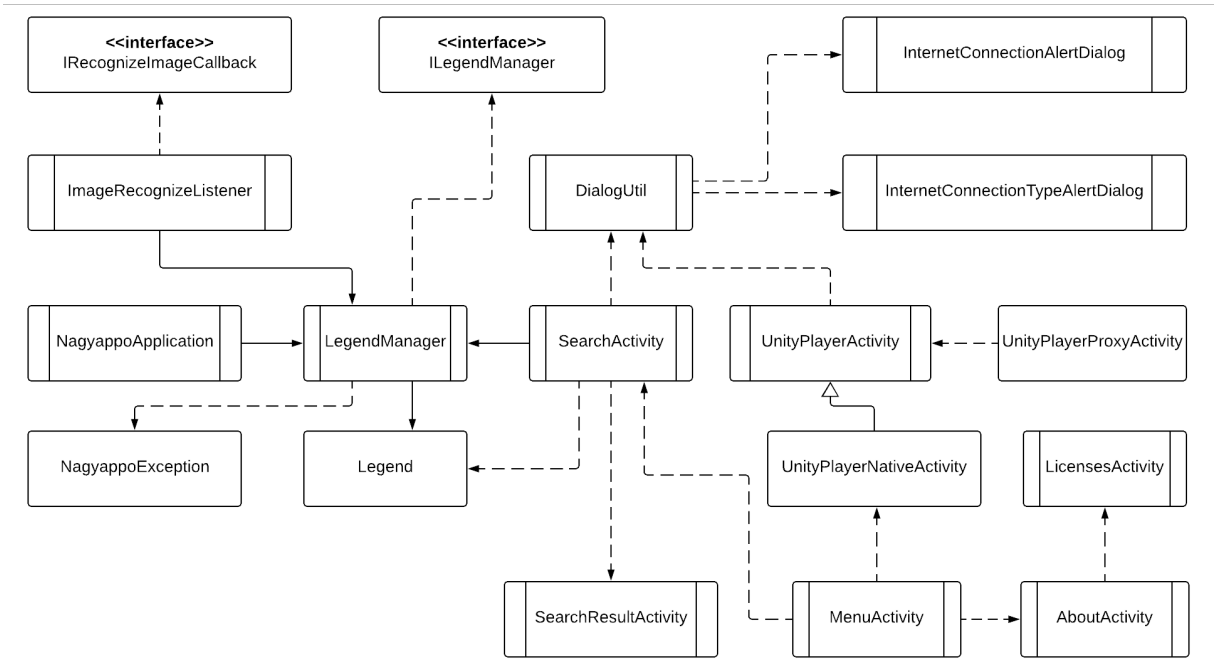
8. kódrészlet. A *LegendManager* osztály metódusának segítségével van lekérve a hanganyag forrása, majd aszinkron módon el van indítva a MediaPlayer

Az alkalmazás Android részében található egy olyan funkcionalitás, amely az elérhető legendákhoz rendelt hangoskönyveket meghallgathatóvá teszi akkor is, ha a felhasználó nem rendelkezik a *Székelyföldi Legendárium* könyvvel. Ehhez biztosítva van egy keresőfelület, amely felhasználja a Json fájlból kiolvasott legendák másik három attribútumát. Ezen a keresőfelületen minden legendához társítva van egy cím, egy azonosító és egy borítókép. A keresés opción belül lehetőség van a cím, illetve az azonosító alapján keresni. A legendák egy *ViewPager*-ben

[5]Plain Old Java Object

vannak megjelenítve, amely a Json fájlból töltődik fel, és könnyű lapozást biztosít a legendák között. Továbbá *DialogBox* segítségével információk vannak megjelenítve a felhasználók számára. Például figyelmeztető ablak ugrik fel, amikor nincs internetkapcsolat vagy a felhasználó mobilhálózaton keresztül van csatlakozva az internetre. Az utóbbi esetben választható az az opció, hogy a felhasználó folytatja az alkalmazás használatát, míg az első esetben a hálózati beállításokhoz lesz átirányítva.

Mindezeket a funkcionalitásokat és műveleteket megvalósító Java osztályok, valamint a közöttük lévő kapcsolatok az alábbi, 7. ábrán látható osztálydiagramon tekinthetőek meg.



7. ábra. Az Android alkalmazást megvalósító osztályok

4. A fejlesztés során felhasznált eszközök

4.1. Android Studio

Az *Android Studio* [3] egy integrált fejlesztői környezet (IDE), amely az *IntelliJ IDEA*-n alapszik és Android platformra történő fejlesztésre alkalmas. Az első stabil verzió 2014-ben jelent meg és teljes mértékben helyettesíteni tudta az *Eclipse Android Development Tool*-ját, valamint újításokat is hozott ehhez képest, így hivatalosan az első számú környezetté vált az Android alkalmazások fejlesztéséhez. Fontosabb funkciók, amelyeket a fejlesztőkörnyezet nyújt: Gradle alapú build rendszer, testreszabható emulátor az eszköz nélküli futtatáshoz, *Instant Run* opció, amely lehetővé teszi, hogy az APK újratelepítése nélkül frissítésre kerüljön az alkalmazás.

4.2. Gradle

Mivel a fent említett fejlesztőkörnyezet *Gradle*-t [10] használ build és függőségmenedzsment rendszerként, minden létrehozott projekt szerkezetében megtalálható egy *build.gradle* állomány, amely tartalmazza a projekthez tartozó függőségeket, illetve olyan információkat, amelyek a build folyamathoz szükségesek. A Gradle esetében egy Groovy alapú projektkonfiguráció (9. kódrészlet) van használva XML helyett.

```
dependencies {
        compile 'com.jakewharton:butterknife:8.8.1'
}
```

9. kódrészlet. Példa a *Butterknife* könyvtár beágyazására függőségként a projektbe

4.3. Visual Studio

A *Visual Studio* [12] egy teljes funkcionalitású integrált fejlesztői környezet (IDE) Android, iOS, web, cloud és Windows alkalmazások létrehozásához. Lehetővé teszi a hatékony fejlesztést, könnyen elérhetővé téve olyan funkcionalitásokat, mint például a hívásszerkezet vizsgálata, a kapcsolódó metódusok, bejelentkezések, tesztek állapotai stb. Ezen kívül lehetőséget nyújt a kóddal kapcsolatos problémák beazonosítására és javítására. Visual Studio-n belül többféle programozási nyelvben írhatóak programok (*C#/VB*, *C++*, *JavaScript*, *Python* stb.). A Nagyappó projekt esetében a Unity-s C# szkriptek voltak Visual Studio segítségével fejlesztve.

4.4. Git, GitKraken, GitLab

A *Git* [4] egy osztott verziókövető rendszer. A *GitKraken* [1] a Git-hez tartozó grafikus felhasználói felület, melynek segítségével könnyen elvégezhetőek a verziókövetéssel és a tároló

(*repository*) menedzsmentjével kapcsolatos műveletek (*commit*, *push*, *pull*, új branch létrehozása stb.).

A *GitLab* [7] a központi repository menedzsmentjét valósította meg, illetve a feladatok nyomonkövetésére (*issue tracking*) és folyamatos integrációra (*continuous integration*: CI) volt használva.

4.5. Docker

A *Docker* [9] egy konténer platform, segítségével lehetőség van egy automatizált build rendszer felépítésére, amely minden *push* művelet után automatikusan elindul és végrehajtja a GitLab-ra feltöltött projekt build folyamatát, biztosítva azt, hogy a feltöltött verzió működőképes.

4.6. Fabric

A *Fabric* [5] egy platform mobilalkalmazások fejlesztői és tesztelői számára, amely biztosítja az összes fejlesztési szakaszon (*stage*) belül az alkalmazás teljesítményének és helyességének a monitorizálását. Egyszerűsíti és automatizálja az alkalmazás beta verziójának a kitelepítési folyamatát. A részét képező *Crashlytics* hibajelentő jelentéseket generál a mobileszközökre kitelepített beta verziókban felbukkanó hibákról, amelyekről így a fejlesztők időben értesülhetnek, és javíthatják azokat.

5. A Legendárium Nagyappó alkalmazás működése

Az Android rendszert futtató eszközökre kitelepített *Nagyappó* alkalmazás indítása után elsőként a 8. ábrán látható menü érhető el.

8. ábra. A *Nagyappó* alkalmazás indítása után elsőként látható képernyő

Az alkalmazás központi funkcionalitása a *Felolvasás* gombra kattintva érhető el, amelynek hatására elindul a Unity modul, és bekapcsol az eszköz kamerája. Ha a kamera a *Székelyföldi Legendárium* könyv egyik olyan oldalára van fókuszálva, amelyet a Vuforia keretrendszer képes felismerni az adatbázisból, akkor, miután megtörténik a felismerés, az eszköz képernyőjén, a kiterjesztett valóság részeként, megjelenik a nagyapó, aki besétál, majd letelepedik a könyv lapjára és „elmeséli” a fókuszban lévő legendát. Ha „mesélés” közben fókuszt veszít az eszköz kamerája (például nincs folyamatosan a kiválasztott legendára irányítva a kamera), akkor sem szakad félbe a mesélés, úgy is lehetőség van a legenda végighallgatására, ugyanis a képfelismerésre csak egyszer van szükség minden legenda esetében. Nagyapó viszont csak akkor látható, ha a kamera a felolvasandó legendára van irányítva. Az eszköz körbeforgatható a 3D-s modellként megjelenő nagyapó körül. Abban az esetben, ha egy legenda felolvasása közben a felhasználó az eszköz kameráját egy másik legendára irányítja, újraindul az animáció sorozat, azaz újra besétál a nagyapó, majd leül és elkezdi az aktuális, fókuszban lévő legendát mesélni. A 9. ábra szemlélteti ahogy nagyapó besétál, majd a 10. ábrán mesél.

9. ábra. A képfelismerés után a kiterjesztett valóság részeként látható a besétáló nagyapó

10. ábra. A kiterjesztett valóság részeként megjelenő nagyapó elmeséli a kamera fókuszában lévő legendát (*Hadak útja*)

Az alkalmazás következő funkcionalitása a *Keresés* gombra kattintva érhető el. Az ehhez tartozó felületet a 11. ábra mutatja be.

11. ábra. Keresés a legendák között és azok elindítása, meghallgatása

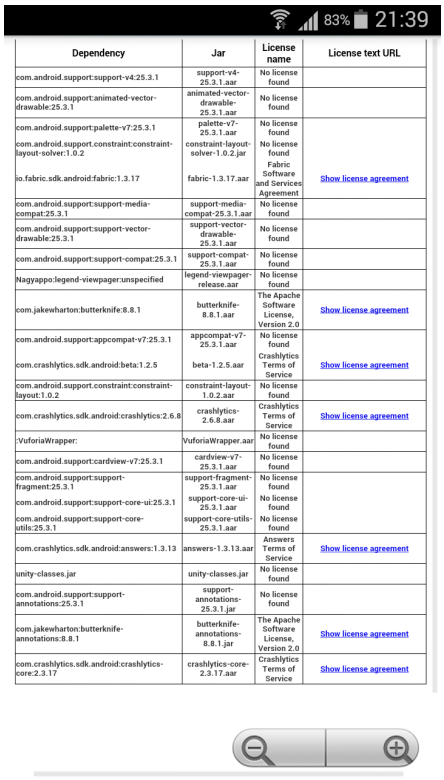
A keresés a legenda címe vagy azonosítója alapján történik, amelyet lehetőség van beírni a 11. ábra felső felén látható kereső mezőbe. Az alkalmazásban az elsőként megjelenő legenda címe a *Medve-tó*, alatta látható a legenda azonosítója (*m 01*), amely a *Székelyföldi Legendárium* könyv lapjainak felső sarkában is fel van tüntetve. A legendák székekre vannak felosztva (Marosszék, Csíkszék, Udvarhelyszék, Gyergyószék stb.). Az azonosító első karaktere a szék kezdőbetűjét tartalmazza, míg az utána következő szám azt mutatja, hogy az adott széken belül hányadik legendaként található meg a könyvben. Az *m 01* kód tehát azt jelenti, hogy a *Medve-tó* legenda a Marosszék „fejezet” első legendájaként jelenik meg. A legenda címe alatt látható gombra kattintva elindítható a legendához tartozó hangoskönyv. Közben lehetőség van szüneteltetni a mesélést, vagy tetszés szerint a legenda felolvasásának közepére/végére lehet ugrani a 11. ábrán látható csúszka segítségével. Ugyanezen a funkcionalitáson belül az alkalmazásból elérhető legendák lapozással végignézhetőek és meghallgathatóak.

Az alkalmazásról gombra kattintva az alábbi, 12. ábrán látható tartalom jelenik meg.

12. ábra. *Az alkalmazásról* felület tartalma

Az *Értékeld a Google Play Store-ban* elem az alkalmazás kiadása után válik majd használhatóvá. A *Licenc* elemet választva, egy lista látható, ahol a fejlesztés során felhasznált technológiák vannak felsorolva (13. ábra).

A *Legendárium termékek* elemet választva betöltődik a *Székelyföldi Legendárium* projekt weboldala, ahol a forgalmazott termékek listája tekinthető meg (lásd a 14. ábrát).



Dependency	Jar	License name	License text URL
com.android.support:support-v4:25.3.1	support-v4-25.3.1.aar	No license found	
com.android.support:animated-vector-drawable:25.3.1	animated-vector-drawable-25.3.1.aar	No license found	
com.android.support:palette-v7:25.3.1	palette-v7-25.3.1.aar	No license found	
com.android.support.constraint:constraint-layout-solver:1.0.2	constraint-layout-solver-1.0.2.jar	No license found	
io.fabric.sdk.android:fabric:1.3.17	fabric-1.3.17.aar	Fabric Software and Services Agreement	Show license agreement
com.android.support:support-media-compat:25.3.1	support-media-compat-25.3.1.aar	No license found	
com.android.support:support-vector-drawable:25.3.1	support-vector-drawable-25.3.1.aar	No license found	
com.android.support:support-compat:25.3.1	support-compat-25.3.1.aar	No license found	
Nagyappo:legend-viewpager:unspecified	legend-viewpager-release.aar	No license found	
com.jakewharton:butterknife:8.8.1	butterknife-8.8.1.aar	The Apache Software License, Version 2.0	Show license agreement
com.android.support:appcompat-v7:25.3.1	appcompat-v7-25.3.1.aar	No license found	
com.crashlytics.sdk.android:beta:1.2.5	beta-1.2.5.aar	Crashlytics Terms of Service	Show license agreement
com.android.support.constraint:constraint-layout:1.0.2	constraint-layout-1.0.2.aar	No license found	
com.crashlytics.sdk.android:crashlytics:2.6.8	crashlytics-2.6.8.aar	Crashlytics Terms of Service	Show license agreement
:VuforiaWrapper:	VuforiaWrapper.aar	No license found	
com.android.support:cardview-v7:25.3.1	cardview-v7-25.3.1.aar	No license found	
com.android.support:support-fragment:25.3.1	support-fragment-25.3.1.aar	No license found	
com.android.support:support-core-ui:25.3.1	support-core-ui-25.3.1.aar	No license found	
com.android.support:support-core-utils:25.3.1	support-core-utils-25.3.1.aar	No license found	
com.crashlytics.sdk.android:answers:1.3.13	answers-1.3.13.aar	Answers Terms of Service	Show license agreement
unity-classes.jar	unity-classes.jar	No license found	
com.android.support:support-annotations:25.3.1	support-annotations-25.3.1.jar	No license found	
com.jakewharton:butterknife-annotations:8.8.1	butterknife-annotations-8.8.1.jar	The Apache Software License, Version 2.0	Show license agreement
com.crashlytics.sdk.android:crashlytics-core:2.3.17	crashlytics-core-2.3.17.aar	Crashlytics Terms of Service	Show license agreement

13. ábra. A fejlesztés során felhasznált külső függőségek és könyvtárak listája

14. ábra. A *Székelyföldi Legendárium* projekt weboldala, ahol elérhető a termékek listája

A *Kilépés* gombra kattintva a felhasználónak lehetősége van bezárni az alkalmazást.

Következtetések és továbbfejlesztési lehetőségek

A Legendárium Nagyappó fejlesztése során elkészült az alkalmazásnak egy működő és az elvárásoknak megfelelő prototípusa, amely rendelkezik a legfontosabb funkcionalitásokkal. Az alkalmazás sikeresen be lett mutatva székelyföldi konferenciákon, illetve az M1 televízió egyik műsorában is. A fejlesztés során számos lehetőség felmerült a továbbfejlesztéssel kapcsolatban. Néhány példa:

- saját központi szerver létrehozása a legendák hangoskönyveinek tárolásához;
- *in-app purchase* beágyazása, amely lehetővé tenné, hogy egyes hangoskönyvek megvásárolhatóak legyenek a Székelyföldi Legendáriumtól;
- legenda-specifikus 3D-s elemek megjelenítése: nagyapón kívül más modellek is jelenjenek meg a legendáknál, amelyek az illető legendához kapcsolódnak;
- többféle animáció beágyazása a projektbe, például nagyapó minden legendánál másképpen viselkedhetne.

További fejlesztéseket és javításokat követően a Codespring és a Legendárium csapata hivatalosan is ki szeretné adni és elérhetővé tenni a *Google Play Store*-ban az alkalmazást.

Hivatkozások

[1] Axosoft. *GitKraken*. 2018. URL: https://www.gitkraken.com/ (utolsó elérés dátuma: 2018. ápr. 10.)

[2] Creative Commons. *Android SDK*. 2017. URL: https://stuff.mit.edu/afs/sipb/project/android/docs/sdk/index.html (utolsó elérés dátuma: 2018. ápr. 09.)

[3] Creative Commons. *Android Studio*. 2017. URL: https://developer.android.com/studio/index.html (utolsó elérés dátuma: 2018. ápr. 09.)

[4] Software Freedom Conservancy. *Git*. 2018. URL: https://git-scm.com/ (utolsó elérés dátuma: 2018. ápr. 10.)

[5] Google Developers. *Fabric*. 2018. URL: https://get.fabric.io/ (utolsó elérés dátuma: 2018. ápr. 10.)

[6] FileHippo. *Android SDK*. 2017. URL: https://filehippo.com/download_android_sdk/ (utolsó elérés dátuma: 2018. ápr. 09.)

[7] *GitLab*. 2018. URL: https://gitlab.com/ (utolsó elérés dátuma: 2018. ápr. 10.)

[8] Crunchbase Inc. *Unity Technologies*. 2018. URL: https://www.crunchbase.com/organization/unity-technologies#section-overview (utolsó elérés dátuma: 2018. ápr. 09.)

[9] Docker Inc. *Docker*. 2018. URL: https://www.docker.com/ (utolsó elérés dátuma: 2018. ápr. 10.)

[10] Gradle Inc. *Gradle*. 2018. URL: https://gradle.org/ (utolsó elérés dátuma: 2018. ápr. 10.)

[11] PTC Inc. *Vuforia*. 2018. URL: https://www.vuforia.com/ (utolsó elérés dátuma: 2018. ápr. 10.)

[12] Microsoft. *Visual Studio*. 2018. URL: https://www.visualstudio.com/vs/ (utolsó elérés dátuma: 2018. ápr. 10.)

[13] West Pier Studio. *Unity 3D development*. 2016. URL: http://www.westpierstudio.com/services/unity-3d-development (utolsó elérés dátuma: 2018. ápr. 09.)

[14] Unity Technologies. *Unity*. 2018. URL: https://unity3d.com/ (utolsó elérés dátuma: 2018. ápr. 09.)