Univerzita Pardubice
Fakulta elektrotechniky a informatiky

Návrh a realizace mobilní aplikace využívající rozšířené reality

Bc. Filip Němeček

Diplomová práce

2019

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2018/2019**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Filip Němeček**
Osobní číslo:	**I17217**
Studijní program:	**N2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Návrh a realizace mobilní aplikace využívající rozšířené reality**
Zadávající katedra:	**Katedra softwarových technologií**

Zásady pro vypracování:

Cílem diplomové práce bude v teoretické části charakterizovat a popsat rozšířenou realitu a analyzovat její využitelnost v mobilních aplikacích. Součástí práce bude i přehled identifikačních a lokačních bezdrátových technologií potřebných pro umísťování digitálních objektů v obrazu reálného světa na displeji mobilního zařízení, identifikace jejich výhod a nevýhod. Práce bude taktéž zahrnovat přehled dostupných knihoven pro vývoj aplikací s rozšířenou realitou. Druhým cílem v praktické části diplomové práce bude navrhnout a realizovat mobilní aplikaci (pro operační systém Android/iOS/Windows Mobile (volitelně)) či multiplatformní mobilní aplikaci v Adobe PhoneGap. Ta bude využívat identifikačních (např. QR kódy) a lokačních (např. GPS v kombinaci s gyroskopem a akcelerometrem) bezdrátových technologií potřebných pro umísťování digitálních objektů v obrazu reálného světa na displeji mobilního zařízení. Mobilní aplikace umožní prostřednictvím rozšířené reality poskytovat uživatelům inovativní uživatelské rozhraní.

Rozsah grafických prací:

Rozsah pracovní zprávy: **50-60**

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

CLAYTON, Craig. Learn iOS 11 Programming with Swift 4: Learn the fundamentals of iOS app development with Swift 4 and Xcode 9. 2. Birmingham, United Kingdom: Packt Publishing Limited, 2018. ISBN 9781788390750
HOFFMAN, Jan. Mastering Swift 4. 4. Birmingham, United Kingdom: Packt Publishing Limited, 2017. ISBN 9781788477802
RAVINDRAN, Arun. Django Design Patterns and Best Practices: Industry-standard web development techniques and solutions using Python. 2. Birmingham, United Kingdom: Packt Publishing Limited, 2018. ISBN 9781788831345

Vedoucí diplomové práce: **Ing. Jiří Kysela, Ph.D.**
Katedra informačních technologií

Datum zadání diplomové práce: **22. října 2018**
Termín odevzdání diplomové práce: **18. května 2019**

Univerzita Pardubice Česká republika

L.S.

Ing. Zdeněk Němec, Ph.D.
děkan

prof. Ing. Antonín Kavička, Ph.D.
vedoucí katedry

V Pardubicích dne 17. listopadu 2018

Prohlašuji:

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 9/2012, bude práce zveřejněna v Univerzitní knihovně a prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 26. 4. 2019

Filip Němeček

PODĚKOVÁNÍ

Rád bych poděkoval svému vedoucímu diplomové práce panu Ing. Jiřímu Kyselovi, Ph.D. za možnost, vypracovat mnou vybrané téma a také za vstřícný přístup při zpracování práce.

ANOTACE

Práce v teoretické části seznamuje s odvětvím augmentované reality. Stručně popisuje aktuální aplikace a možnosti. Dále srovnává možnosti pro umisťování objektů v prostředí augmentované reality a srovnává dostupné knihovny pro vývoj mobilních aplikací využívajících augmentovanou realitu.

Praktická část se zabývá návrhem iOS aplikace, která v prostředí augmentované reality umí rozeznat obrázky a zobrazit k nim virtuální obsah. Zvládá rovněž detekovat zařízení iBeacon a je podpořena webovou administrací pro snadnou správu obsahu.

KLÍČOVÁ SLOVA

Augmentovaná realita, ARKit, mobilní aplikace, iOS, Swift, Django, Python

TITLE

Design and implementation of augmented reality mobile app.

ANNOTATION

Theoretical part introduces augmented reality, and its current possibilities. It also compares available methods of placing virtual objects into the real world and compares most popular augmented reality libraries for mobile applications.

Practical part consists of development of an iOS mobile application which can, in augmented reality setting, recognize images and display virtual content alongside the recognized image. It can also detect iBeacon devices and is powered by web administration for easy content management.

KEYWORDS

Augmented reality, ARKit, mobile application, iOS, Swift, Django, Python

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM ZKRATEK

API	Application Programming Interface
AR	Augmented Reality
ARM	Advanced RISC Machine
GPS	Global positioning system
GmbH	Gesellschaft mit beschränkter Haftung
HTML	HyperText Markup Language
LTE	Long Term Evolution
QR	Quick Response
SDK	Software Development Kit
WWDC	Worldwide Developers Conference
ARC	Automatic Reference Counting
CMS	Content Management System
CSS	Cascading Style Sheets
IDE	Integrated Development Environment
JSON	JavaScript Object Notation
MHD	Městská hromadná doprava
NaN	Not a number
PNG	Portable Network Graphics
UML	Unified Modeling Language
URL	Uniform Resource Locator
USB	Universal Serial Bus
UUID	Universally Unique Identifier
XML	Extensible Markup Language

ÚVOD

Augmentovaná realita bude pravděpodobně jedno z dalších velkých odvětví informatiky následujících let. Ve velkém se o ní zajímají společnosti Apple, Google i Microsoft. Aktuální šéf Apple Tim Cook dokonce již v roce 2017 prohlásil, že augmentovaná realita navždy změní používání technologií [5]. Microsoft vyvíjí profesionální headset HoloLens [6], Google zase knihovnu pro vývoj aplikací s augmentovanou realitou a v minulosti také uvedl chytré brýle Google Glass [7], které jsou aktuálně nabízeny primárně firmám [8].

Cílem práce je v teoretické části stručně seznámit s odvětvím augmentované reality. Ukázat možná praktická využití v mobilních zařízeních již dnes či se letmo ohlédnout do budoucna, jaké změny přinesou dostupné chytré brýle. Teoretická část bude rovněž obsahovat popis dostupných možností, jak do prostředí augmentované reality umístit virtuální objekty. Dále teoretická část přinese představení hlavních knihoven pro vývoj mobilních aplikací s augmentovanou realitou, přičemž do detailu bude probrána hlavně knihovna ARKit od Apple, protože pomocí ní bude vytvořena mobilní aplikace v praktické části. V závěru teoretické části budou ještě popsané knihovny porovnány podle několika parametrů, jako například platformy, cena nebo počet dostupných výukových materiálů.

Hlavním cílem praktické části práce je vytvoření takové mobilní aplikace, využívající augmentovanou realitu, která dokáže i nezasvěcenému uživateli jasně ukázat praktické využití této technologie v běžném životě. U výsledné aplikace tedy nebude problém představit si, jak by mohla, po menších úpravách, fungovat v praxi v různých situacích.

Aplikace bude umět rozeznat konkrétní obrázky či jakýkoliv jiný 2D povrch na základě připravených dat v podobě fotky těchto objektů. Následně po rozpoznání takového obrázku zobrazí v jeho bezprostředním okolí virtuální obsah, se kterým bude možné případně i interagovat a například otevřít webovou stránku nebo spustit aplikaci s navigací na určené místo. Kromě toho aplikace, po příslušné konfiguraci, umožní po rozpoznání obrázku zobrazit dodatečné informace na celý displej zařízení, tedy již bez využití augmentované reality.

Aby aplikace byla skutečně prakticky použitelná, bude podpořena webovou administrační částí. Ta umožní uživateli, bez znalosti jakéhokoliv programovacího jazyka, aby snadno nakonfiguroval, jaké obrázky bude aplikace umět rozeznat a jaký obsah bude následně zobrazen. V neposlední řadě umožní nastavit konkrétní akce virtuálnímu obsahu, jako je třeba již zmíněné otevření webové stránky nebo spuštění navigace. Webová aplikace nabídne moderní, intuitivní design, včetně celé řady náhledů, aby uživatel měl okamžitě přehled, jaký obsah je aktuálně přidán.

Praktické využitelnosti aplikace pomůže rovněž integrace technologie iBeacon, pomocí které může aplikace zjistit, kde se uživatel právě nachází, a podle toho stáhnout konkrétní obrázky k rozpoznání, protože existuje poměrně striktní limit na současný počet rozpoznávaných obrázků. Tato iBeacon zařízení bude samozřejmě možné spravovat ve webové administrační části. Aplikace bude navíc díky speciální kategorii obrázků umět fungovat i bez přidání jediného iBeacon zařízení. Kromě toho praktická část práce obsáhne stručný postup, jak si testovací iBeacon zařízení vytvořit pomocí široce dostupného mikropočítače Raspberry Pi.

1 AUGMENTOVANÁ REALITA

Augmentovaná realita, často označovaná jako rozšířená realita, se dá nejstručněji popsat jako virtuální obraz zobrazený ve skutečném světě. Důležitou podmínkou je, aby byl tento virtuální obraz určitým způsobem „napojený“ na skutečný svět. V jednoduchých případech se může jednat například o zobrazení lokační značky v prostoru podle GPS souřadnic nebo zobrazení virtuálního obsahu po detekování speciální značky (označované anglicky jako „marker“). Pokročilejší aplikace augmentované reality zvládají například detekovat horizontální a vertikální plochy (anglicky „planes“), ty potom mohou sloužit jako báze ke zobrazení komplexních 3D modelů. Tyto modely mohou následně vypadat, jako by na dané ploše byly skutečně položeny nebo pověšeny, v případě vertikální plochy.

Další technikou, která dokáže výrazně přispět ke konečné iluzi, že je virtuální objekt skutečný, je tzv. „light estimation“. Jedná se o techniku, při které zařízení, zobrazující augmentovanou realitu, zároveň snímá světelné informace a vypočítává z nich odhad světla ve scéně. Tento odhad nejčastěji zahrnuje směr a intenzitu a na jeho základě může dojít k velmi jednoduchému stínovaní 3D objektů. Strany orientované ke zdroji světla jsou potom světlejší a objekt do prostředí zapadá mnohem lépe. Programátor v těchto případech typicky nemusí nic dalšího implementovat a stačí pouze odhad světla ve scéně zapnout.

Pro korektní zobrazení virtuálního obsahu a také pro samotné udržení jeho pozice v případě, že se zařízení hýbe, musí daný systém vypočítávat vzdálenosti ve scéně. Tyto informace mohou posloužit také k měření vzdáleností či rozměrů objektů ve skutečném světě. Například operační systém iOS od firmy Apple nabízí od své 12. verze aplikaci Measure [9], jenž umí právě po nasnímání skutečného světa umístit body a zobrazit vzdálenost mezi nimi. Při dobrých světelných podmínkách a korektním použití jsou tyto vzdálenosti přesné na centimetry, jak bylo zjištěno při empirickém testování aplikace Measure na zařízení iPhone 7 a předmětech s rozměry v rozmezí několika centimetrů až několika metrů.

Augmentovaná realita bývá často kombinována s dalšími pokročilými technikami a odvětvími informatiky. GPS již bylo zmíněno výše, používá se často pro zobrazení zajímavých bodů ve městech. Existují také mapové aplikace, které pomocí pozice uživatele a augmentované reality zobrazí šipky na chodníku a tímto navigují [10], byť právě tyto aplikace často naráží na menší přesnost GPS, která není úplně vhodná k umisťování virtuálních objektů do bezprostředního okolí uživatele. Využít lze také počítačové vidění v kombinaci se strojovým učením a připraveným modelem. Tyto aplikace mohou následně rozeznat objekty skutečného světa a například zobrazit stručný popisek, nebo tuto informaci předat uživateli například

zvukově. Může jít o skvělého pomocníka pro nevidomé uživatele, kterým mobilní telefon řekne, jaké předměty se nacházejí v jejich okolí.

V současné době nabízí pokročilejší systémy augmentované reality v podobě headsetů ještě větší možnosti. Nejznámější a nejpokročilejší headset v komerční sféře nabízí, v době psaní práce, Microsoft v podobě headsetu pro augmentovanou realitu HoloLens 2 [6]. Kromě snímání uživatele v prostoru, což je možné i za pomoci mobilního telefonu, dokáže snímat pohyby rukou a drobná gesta prstů. Uživatelé tak mohou snadno například uchopit virtuální objekt a například ho někam posunout. HoloLens 2 je určeno pro firemní použití a Microsoft uvádí mezi možnostmi třeba vzdálenou asistenci pracovníkům, kteří mohou díky headsetu vidět virtuální instrukce či třeba animace, které ukazují, jak komplexní zařízení funguje a jak mají postupovat při jeho diagnostice nebo opravě.

Velmi pokročilou augmentovanou realitu slibuje rovněž společnost Magic Leap [11], která sídlí na Floridě a byla založena už v roce 2010. Pochlubit se může vysokými investicemi, které přesáhly miliardu dolarů a jedním z investorů byla společnost Google [12]. Ovšem produkt pojmenovaný Magic Leap One byl několikrát odložen [13], a světlo světa spatřil nakonec v srpnu roku 2018. Podle recenzí předních technologických webů ovšem nepřinesl tak pokročilou augmentovanou realitu [14, 15], jak jeho tvůrci slibovali.

2 MOŽNOSTI VYUŽITÍ MOBILNÍ AUGMENTOVANÉ REALITY DNES

Díky již poměrně pokročilým možnostem mobilní augmentované reality se objevila řada užitečných aplikací. Švédská IKEA nabízí již poměrně dlouho mobilní aplikaci IKEA Place [16]. Potenciální zákazníci si pomocí aplikace mohou doma „vyzkoušet“ nábytek a následně se rozhodnout, zda daný předmět koupit či nikoliv. Aplikace v místnosti zobrazí 3D model skutečných produktů se skutečnými rozměry a uživatel získá mnohem lepší představu, jak bude daný kousek nábytku zapadat do jeho interiéru.

Obrázek 1 - Ukázka aplikace IKEA Place [16]

Spousta mobilních aplikací je rovněž určena pro výukové účely. Například lze na stole zobrazit kompletní kostru člověka, případně její části včetně orgánů a detailně si pomocí otáčení vše prohlédnout. Podobné aplikace dokáží interaktivně prezentovat také živočichy, solární systém a mnoho dalšího. Fanoušci astronomie mohou využít řadu aplikací zobrazujících souhvězdí a další informace na noční obloze. Zajímavé využití rozpoznávání obrazu

a augmentované reality představuje aplikace PeakFinder pro systémy iOS a Android [17], která má v databázi přes 650 tisíc hor a dokáže k nim kromě názvu zobrazit další užitečné informace.

S trochou nadsázky se dá říci, že pro propagaci mobilní augmentované reality udělala nejvíce hra Pokémon GO od studia Niantic, vydaná v červnu roku 2016 [18]. Využívá primárně GPS pro zobrazení pozice hráče na virtuální mapě, ale také augmentovanou realitu pro zobrazení jednotlivých pokémonů na displeji telefonu, které hráči chytají. Her využívajících augmentovanou realitu vyšlo od dob Pokémon GO spousta, žádné se ale nepodařilo získat takové množství hráčů, jakým se mohlo chlubit Pokémon GO. Podle odborných odhadů si hru stáhlo celkem více než 800 milionů hráčů, přičemž počet aktivních dosahoval v nejsilnějších měsících až ke 150 milionům [19]. Na podobně impresivní čísla dosáhne málokterá hra, ať už využívá mobilní realitu nebo je dostupná pro herní konzole či počítače.

2.1 Chytré brýle jako další posun augmentované reality

Mobilní telefony a tablety mají dnes dostatek výkonu pro zobrazení komplexní augmentované reality a není problém pomocí standardních fotoaparátů snímat scénu. Nevýhodou je ovšem uživatelský komfort, kdy je nutné držet před sebou zařízení a vhodně s ním pohybovat pro zabrání celé scény. V případě tabletů vstupuje do hry ještě nemalá hmotnost zařízení. Například iPad vydaný v roce 2018 váží okolo 470 gramů (v závislosti, jestli se jedná o LTE verzi) [20], takže je nutné tablet držet oběma rukama.

Zkrátka i v případě ještě výkonnějších zařízení s ještě přesnějším snímáním nepůjde, z důvodu výše popsaných omezení, vytvořit augmentované aplikace příjemné pro použití déle než několik minut.

Právě tento problém augmentované reality by měly řešit chytré brýle, které již existují, z důvodu vysoké ceny jsou ale cílené na firemní použití, kde dokáží významně podpořit firemní procesy a produktivitu v případě promyšleného nasazení a cena není ani zdaleka tak limitující, jako v případě zájemců z řad běžných uživatelů.

S rapidně rostoucím výkonem mobilních čipů architektury ARM se dá očekávat [21], že během příštích několika let budou dnešní čipy natolik efektivní, že je zvládne pohánět baterie v chytrých hodinkách nebo právě hypotetických chytrých brýlích.

Očekávané chytré brýle zároveň mohou posloužit jako motivace pro vývojáře, aby s augmentovanou realitou alespoň experimentovali již dnes. Získané zkušenosti a naučené základní koncepty se budou později hodit a tito programátoři budou mít při uvedení brýlí, určených pro masivní adopci, výhodu nad ostatními a snáze budou moci uživatelům přinést

užitečné aplikace. Podobná motivace zřejmě existuje i pro obří firmy jako jsou Apple a Google, které v posledních letech investují velké množství zdrojů do vývoje knihoven ARKit, respektive ARCore, aby nalákaly vývojáře a zároveň začaly budovat knihovny aplikací s augmentovanou realitou pro předpokládané uvedení chytrých brýlí.

Chytré brýle budou moci posloužit jako například všudypřítomná navigace, která nebude vyžadovat, aby uživatel sklopil pohled na mobilní zařízení a k tomu ještě využil minimálně jednu ruku. Snadno si lze představit také obdobu Apple aplikace Measure přímo pro brýle, kdy se například po aktivaci příkazu uživateli zobrazí kóty s rozměry na vybraném kusu nábytku či jiném předmětu. Aplikace pro vaření by snadno mohla vizualizovat požadované kvantity surovin na jednotlivé recepty a podobných příkladů lze určitě vymyslet mnohem více.

3 MOŽNOSTI UMISŤOVÁNÍ VIRTUÁLNÍCH OBJEKTŮ VE SKUTEČNÉM SVĚTĚ

Augmentovaný obsah by měl vytvářet alespoň minimální iluzi, že zapadá do skutečného světa, který uživatel vidí před sebou. V opačném případě se jedná pouze o promítnutí obsahu na displej mobilního zařízení a zobrazení obsahu hledáčku fotoaparátu na pozadí. Existuje celá řada technik, jak virtuální objekty či jednoduše obsah do skutečného světa začlenit a tato kapitola je podrobně popisuje.

3.1 Využití QR kódu nebo jiné značky

QR kódy slouží k zakódování prakticky libovolného typu dat, která jsou následně strojově přečtena. Mohou obsahovat vizitky, webové adresy, prostý text nebo sloužit jako tzv. značka pro aplikace využívající augmentovanou realitu.

Pro tento případ se používají ještě tzv. markery, což jsou speciální obrazce fungující na stejném principu, ale nejsou standardizované. Princip je poměrně jednoduchý. Mobilní zařízení si skrze fotoaparát načte QR kód nebo marker a na jeho polohu umístí virtuální objekty. Fyzické rozměry této značky se využijí pro výpočet rozměrů virtuálního obsahu.

Řešení je to jednoduché a řada prvních aplikací, využívající augmentovanou realitu, na něj spoléhala. Zároveň se jedná o velmi omezené řešení, protože je nutné mít značku stále k dispozici, pokud chceme augmentovanou realitu zobrazit, a navíc nejde o esteticky nic příjemného.

3.2 Využití GPS a gyroskopu

Další možností, jak zobrazit virtuální obsah, je využití polohy. GPS slouží k získání polohy uživatele a gyroskop potom k zjištění orientace zařízení, aby mohla aplikace zjistit, jestli se uživatel přes hledáček fotoaparátu dívá třeba na západ. Následně aplikace může zobrazit například nejbližší bankomaty, obchody či jiná významná místa ve městě.

Tato technika opět patřila mezi jedny z prvních, které se ve světě mobilní augmentované reality objevily. Jednu z prvních populárních aplikací měla společnost Nokia pro své Windows Phone 8 mobilní telefony Lumia (takže například modely Lumia 820 nebo Lumia 920). Aplikace se jmenovala Nokia City Lens a byla vydána koncem roku 2012 [22].

GPS a gyroskop představuje jednoduchou možnost, jak využít augmentovanou realitu společně s geolokací, hlavním limitem je ovšem využití. Tato technika se hodí pouze na

zobrazování značek pro vzdálené objekty. GPS nedisponuje dostatečnou přesností, aby pomocí ní šlo umisťovat virtuální objekty v těsné blízkosti uživatele a zobrazit mu například navigační šipky přímo před ním na chodníku nebo silnici.

3.3 Využití předdefinovaného obrázku nebo objektu

Pokročilejším způsobem, jak využít externí „značky“ pro umístění virtuálního obsahu, je detekce předdefinovaných obrázků nebo dokonce 3D objektů. Právě tato technika je použita v praktické části práce.

S příchodem výkonnějších mobilních zařízení přišla možnost rychlé analýzy obrazu a s ní rozpoznávání obrázků v rámci milisekund. Podobně je možné rozpoznávat předem naskenované 3D objekty. Aplikace mohou díku tomu vytvářet dojem, že „rozumí“ okolnímu světu a umí ho obohatit o kontext a další informace.

Ačkoliv je princip dost podobný umisťování obsahu na základně QR kódu nebo jiné speciální značky, není nutné, aby tato značka narušovala celkový dojem, protože zařízení rozpozná přímo objekt zájmu. Na komplexní stroj v dílně může aplikace zobrazit popisky a animace, jak funguje. Karty fantasy hry mohou uživateli zobrazit působivé animace 3D monster a možností je celá řada.

Nevýhodou je opět nutnost znát obrázky či objekty k rozpoznání předem, což může být limitující. Dalším limitem je počet obrázků nebo objektů, které zvládne aplikace současně rozeznat. Tento limit je možné vyřešit pomocí geolokace, praktická část ukazuje řešení pomocí technologie iBeacon.

3.4 Dynamické umístění objektů na rozpoznané plochy

Často používané a flexibilní řešení fungující na principu obrazové analýzy scény. Mobilní zařízení dokáže rozpoznat horizontální a vertikální plochy, na které lze následně virtuální obsah umístit.

Výpočetně se jedná o výrazně náročnější řešení, než je rozpoznání například QR kódu, proto se také jedná o relativně nový princip v oblasti mobilní augmentované reality. Populární knihovny jako ARKit od Apple nebo ARCore od Google tuto funkcionalitu obsahují a vývojáři ji mohou snadno využít.

Díky všudypřítomným plochám se jedná o pružné řešení pro celou řadu aplikací využívající augmentovanou realitu. Tyto aplikace mohou uživateli rovněž nabídnout, aby s virtuálním

obsahem dále manipuloval. Příkladem může být ilustrace nábytku v místnosti nebo populární hra Pokémon GO, zobrazující fantasy postavičky ve skutečném světě. Jelikož knihovny umí kromě rozpoznání ploch, provést také odhad pozice světelného zdroje, lze výsledné objekty velmi snadno doplnit věrohodným stínováním a přidat také stíny.

Kromě výpočetní náročnosti představuje nevýhodu také nutnost, aby uživatel nejdříve nechal zařízení scénu rozpoznat. Typicky je nutné se zařízením chvíli pohybovat, aby mohlo dojít k detekci ploch. To může zbytečně narušovat celkový uživatelský zážitek, zvlášť v případě, kdy detekce trvá delší dobu. Pro korektní funkčnost je zároveň nutný dostatek světla, v opačném případě nemusí dojít k ukotvení virtuálního obsahu ve skutečném světě a při pohybu s mobilním zařízením dochází k pohybu virtuálního obsahu, což výrazně boří iluzi propojení virtuálního a skutečného světa.

3.5 Načtení předem naskenované scény

S pokročilými technikami v knihovnách pro mobilní augmentovanou realitu přibyla rovněž možnost, uložit data rozpoznané scény do souboru a při dalším spuštění aplikace je načíst, čímž odpadá nutnost opětovně scénu rozpoznávat a celý proces je náležitě zrychlený.

Tuto techniku mohou využít tvůrci mobilních aplikací využívající augmentovanou realitu a nasnímané scény připravit do aplikace pro uživatele k použití. Rovněž si lze představit situaci, kdy si aplikace sama načtené scény stahuje z webové služby a uživateli potom stačí, aby namířil hledáček fotoaparátu na podporovanou scénu a ta je následně rozpoznána. V této scéně mohou být rovnou umístěny virtuální objekty na připravených místech. Lze si představit třeba průmyslové využití, kdy aplikace pro techniky obsahuje nasnímané komplexní stroje a po jejich rozpoznání ze zdrojových dat rovnou zobrazí dodatečné informace technikům.

Výhodou, z uživatelského pohledu, je především mnohem rychlejší „inicializační“ fáze, kdy uživatel nemusí dlouze pohybovat mobilním zařízením, aby došlo k rozpoznání ploch nebo objektů a hned po rozpoznání mohou vidět virtuální obsah. Náročnost na přípravu augmentované reality je tak přesunuta na tvůrce aplikací, kteří musí zajistit kvalitní data nasnímaných scén, aby výsledek fungoval dobře.

4 DOSTUPNÉ KNIHOVNY PRO VÝVOJ MOBILNÍ AUGMENTOVANÉ REALITY

Pro vývoj mobilní augmentované reality existuje již poměrně dostatek různých knihoven. Některé jsou zdarma, pro jiné je třeba zakoupit licenci. Nejvíce prostoru v jejich popisu je věnováno knihovně ARKit od Apple, protože je použita v praktické části práce a zároveň se jedná o nejrozšířenější a nejdostupnější moderní knihovnu pro vývoj mobilních aplikací s augmentovanou realitou, co se týče dostupných zařízení v oběhu, které tyto aplikace mohou spustit.

4.1 ARKit

První verze knihovny ARKit byla představena v červnu 2017 na konferenci WWDC [23], kterou každoročně pořádá společnost Apple, a představuje softwarové novinky pro svoje produkty. Tradičně dochází k odhalení nové verze operačního systému iOS a právě v roce 2017, při představení iOS ve verzi 11, byla světu poprvé ukázána knihovna ARKit jako jeho součást. Od zmiňované konference bylo možné začít vyvíjet ARKit aplikace pomocí dostupných betaverzí a s podzimním vydáním hotové verze iOS 11 se technologie rozšířila mezi miliony uživatelů mobilních telefonů iPhone a tabletů iPad [24].

ARKit se může chlubit velkou podporou iOS zařízení, nejnižší podporovaný model je iPhone 6S vydaný v roce 2015, takže i čtyři roky staré zařízení zvládá aplikace s augmentovanou realitou. Na straně tabletů iPad je nejstarší podporovaný model iPad 5. generace uvedený na trh v roce 2017 a také všechny modely řady iPad Pro [25]. Všechna zařízení od iPhone 6S a výš dostala rovněž aktualizaci na operační systém iOS 12 [26], který je potřeba pro využití verze 2.0 knihovny ARKit.

Podpora relativně starých iOS zařízení, společně s jejich obří popularitou mezi uživateli, znamená, že mají vývojáři aplikací s ARKit potenciálně obří publikum, kterému lze tyto aplikace nabídnout. Pokud se podíváme na prodeje iPhone za poslední roky od uvedení iPhone 6S (tedy nejstaršího podporovaného modelu), povedlo se Apple v roce 2015 prodat zhruba 231 milionů kusů, v roce 2016 potom zhruba 211 milionů a v roce 2017 216 milionů [27]. Koncem roku 2018 přestala firma zveřejňovat prodané kusy a reportuje pouze tržby [28]. Dá se odhadovat, že v roce 2018 se prodeje opět pohybovaly okolo 200 milionů kusů. Tato čísla dohromady znamenají více než 600 milionů prodaných zařízení schopných spustit ARKit aplikace. V konečném počtu nejsou ještě zařazeny tablety iPad od roku 2017, které rovněž ARKit aplikace podporují.

Bez dalšího počítaní se z výše uvedených čísel dá prohlásit, že se vývojáři aplikací, postavených na knihovně ARKit, nemusí obávat malého potenciálního trhu uživatelů, s 500 miliony možných zákazníků mohou „počítat“.

První verze ARKit uměla především přesně snímat skutečný svět za pomoci hlavní kamery iOS zařízení a pomocí akcelerometru a dalších senzorů snímat pohyb uživatele, respektive jeho zařízení. Již první verze zvládala detekovat horizontální plochy, jako jsou třeba stoly, podlahy a další, a následně na ně umístit virtuální objekty. Součástí první verze ARKit byla tzv. technika „light estimation“, zmíněná již v obecné kapitole o augmentované realitě. ARKit tak uměl vypočítat, kde se zhruba nachází zdroj světla, a zároveň jeho intenzitu a na základě toho doplnit 3D objekty o stínování [29].

Pro iPhone X znamenal ARKit, díky přední kameře s technologií TrueDepth, přesné snímání obličeje uživatele, což Apple využil pro 3D emoji nazvané animoji a další vývojáři mohli nabídnout aplikace pro focení s filtry, které různě proměnily obličej uživatelů, ať už na fantasy postavu nebo různá zvířata.

Na nové možnosti nemuseli vývojáři čekat rok, až přijde iOS 12. Apple vydal „přechodnou“ verzi knihovny ARKit s označením 1.5, která přinesla dvě důležitá vylepšení [30]. Verze 1.5 byla součástí aktualizace 11.3 pro systém iOS a umožnila rozpoznávat vertikální plochy, takže bylo možné umisťovat virtuální obsah na zdi, například různé fotky nebo plakáty. Druhou důležitou novinkou byla detekce 2D obrázků, jejichž rozpoznání podle vzoru proběhlo velmi rychle a vývojáři mohli po detekování předem určeného obrázku přidat virtuální obsah. Například šlo rozpoznat plakát chystaného filmu a rovnou uživateli pustit video upoutávku nebo ho přesměrovat na více informací o filmu.

4.2 ARKit 2.0

V současné době nejvyšší dostupná verze knihovny ARKit, která dorazila společně s operačním systémem iOS 12 [26]. Ten je pro novou funkcionalitu vyžadován, a právě na možnostech ARKit 2.0 je postavena praktická část této práce, protože vydatně využívá rozpoznávání obrázků.

Druhá verze ARKit přinesla podporu pro více uživatelů, takže více zařízení se stejnou aplikací může sdílet jednu virtuální scénu, čímž se zajistí, že všichni uživatelé vidí na svých zařízeních stejné objekty umístěné stejně v prostoru. Společně s touto novinkou umí ARKit uložit nasnímanou scénu, včetně přidaných objektů, takže ji lze později snadněji obnovit a uživatel není nucen snímat prostředí znovu a čekat na detekci jednotlivých ploch. Právě toto

je jedna z hlavních bariér augmentovaných aplikací. Po spuštění je tradičně nutné provést základní snímání scény, aby se zařízení zorientovalo, načetlo plochy a až poté je možné zobrazovat obsah či dát kontrolu uživateli.

S ARKit 2.0 přišlo rovněž výrazně vylepšené rozpoznávání 2D obrazu podle předdefinovaných vzorů. Knihovna totiž kromě prvotního rozpoznání umí umístit do prostoru tzv. kotvy, které se následně hýbají společně s rozpoznaným obrázkem, takže umístěný virtuální obsah se pohybuje společně s fyzickým. Stejným stylem je možné rozpoznat prakticky jakýkoliv 3D objekt a opět k němu pomocí kotev umístit virtuální obsah, který se následně pohybuje společně s tímto objektem [31, 32].

V případě 3D objektů je velkou limitací potřeba jejich předchozího naskenování, aby měl ARKit dostupná data pro rozpoznávání. 2D obrázky stačí vyfotit a následně oříznout, čímž je příprava zdrojového obrázku pro pozdější rozpoznání dokončena. Objekty musí v tomto případě vývojář kvalitně naskenovat (Apple nabízí zdarma dostupnou aplikaci, která uživatele, respektive spíše vývojáře celým procesem provede [33]) a následně tyto výsledné soubory nahrát do své aplikace.

Skenování a rozpoznávání 2D obrázků a 3D objektů funguje v základě na velmi podobných principech. Ze zdrojových dat (fotka nebo speciální soubor s naskenovaným objektem) si ARKit načte tzv. „feature points“, což jsou unikátní body rozpoznávaného obsahu a podle nich následně pozná skutečný obraz nebo objekt. V případě velmi podobných objektů nebo obrázků se může stát, že dojde k chybné detekci, protože ARKit podle feature points nerozezná rozdíl. V tomto případě ale mluvíme o skutečně ojedinělých případech, kdy třeba není objekt naskenován kvalitně, a vývojář se snaží pomocí ARKit rozpoznat velmi podobné objekty.

Poslední z velkých novinek ARKit 2.0 je nový formát souborů USDZ, který Apple vyvinulo společně s firmou Pixar [34]. Slouží jako unifikovaný formát pro 3D objekty používané v aplikacích s augmentovanou realitou. V iOS 12 umí řada systémových aplikací zobrazit náhled pro USDZ modely, které si například uživatel z aplikace Mail nebo Notes může snadno zobrazit, buď jako tradiční 3D model nebo je rovnou umístit do prostoru pomocí integrované augmentované reality. Protože takto umí zobrazit USDZ také prohlížeč Safari na iOS, je možné tyto objekty vložit do webových stránek a uživatelé si je mohou v prostoru zobrazit přímo pomocí Safari. Apple na svých webových stránkách nabízí stručnou galerii, kde jsou tyto objekty k vyzkoušení [35]. Stačí ji otevřít v Safari na podporovaném iOS zařízení (iPhone 6S a výš, iPad 2017 a výš) a můžete si je prohlédnout v prostoru.

Tabulka 1 - Vlastnosti ARKit v kostce

Vývojář	Apple
Vydání verze 1.0	Září 2017
Platforma	iOS od verze 11
Jazyk	Swift, Objective-C
Cena	Zdarma

4.3 ARCore

Příběh ARCore od internetového giganta Google vlastně začíná u projektu Tango. Jeho historie se datuje až do roku 2014, kdy Google celý systém poprvé představil [36]. Šlo o ambiciózní technologii, jak přinést pokročilou augmentovanou realitu do mobilních zařízení. Hlavní problém Tango spočíval v nutnosti velmi specializovaného hardware mobilních zařízení, za celou dobu existence se prodávaly pouze dvě zařízení s jeho podporou a sice Lenovo Phab2 Pro a později Asus ZenFone AR. Druhý jmenovaný smartphone se dostal na trhu v průběhu srpna roku 2017 [37], tedy po oznámení konkurenčního ARKit od Apple a Google následně v prosinci oznámil konec projektu Tango s efektivní platností 1. března 2018 [38].

Ačkoliv Tango zmizelo, zkušenosti a vyvinuté technologie daly vznik knihovně ARCore, kterou Google oznámil koncem srpna 2017 [39]. Novinka přinesla jedno zásadní vylepšení oproti Tango. Podobně jako ARKit přestala vyžadovat speciální hardware mobilní zařízení, takže pro použití stačil pouze dostatečně moderní chytrý telefon s operačním systémem Android.

ARCore bylo původně vydáno s podporou zařízení Google Pixel a Samsung Galaxy S8, ale Google poměrně rychle přidával podporu pro další modely a začátkem roku 2018 oznámil, že by ARCore ve verzi 1.0 mělo být dostupné více než 100 milionům uživatelů [40]. Potenciální vývojáři augmentovaných aplikací pro Android tak nově nemuseli doufat, že se rozšíří Tango vyžadující specializovaný hardware, a stačilo sledovat podporovaná existující zařízení. Zajímavý je také fakt, že ARCore podporuje rovněž iOS zařízení, která podporují konkurenční ARKit [41]. Vývojáři mohou využít ARCore k vytvoření aplikace s augmentovanou realitou pro iOS za pomoci jazyka Objective-C. Jedná se ale o málo populární volbu, takže není k dispozici podobný počet výukových materiálů, jako třeba pro ARKit nebo ARCore pro Android zařízení.

Schopnosti první verze ARCore se dost blížily prvotnímu vydání již dříve popsaného ARKit. Knihovna uměla snímat pohyb uživatele (respektive jeho zařízení) v prostoru a zachovat pozici virtuálních objektů společně s detekcí horizontálních ploch pro umístění virtuálního obsahu. Nechybělo ani dříve popsané light estimation pro „pochopení“ světla snímané scény s možností aktivovat stínování pro 3D objekty.

Nové verze ARCore se zájemci dočkali poměrně rychle. Google využil svoji pravidelnou konferenci I/O, kde představuje každoročně nejen softwarové novinky a v roce 2018 odhalil ARCore 1.2 [42, 43]. Při odhalení se společnost rovněž pochlubila, že za pár měsíců, co je na

ARCore dostupné vývojářům, nabízí obchod Google Play Store stovky aplikací nabízející uživatelům augmentovanou realitu [43].

Přidané funkce v ARCore 1.2 zahrnovaly hlavně schopnost rozpoznat vertikální plochy, takže vývojáři mohli využít například stěny k zobrazení virtuálního obsahu. Další velkou novinkou byly tzv. Shared AR Experiences, tedy možnost, aby stejné virtuální objekty vidělo na svém displeji více uživatelů. Tato možnost funguje pomocí cloudové synchronizace a Google pro její demonstraci vydal aplikaci Just a Line, ve které mohou uživatelé v prostoru kreslit a ostatní vidí jejich výtvory. Aplikace je zdarma ke stažení v Google Play Store a v době psaní práce měla počet stažení přes 100 000 [44].

Poslední významnou novinkou představenou společně s ARCore 1.2 bylo API Sceneform pro rychlejší a jednodušší práci s 3D objekty a jejich vykreslování. Sceneform navíc není omezené pouze na augmentované aplikace, ale lze jej využít v prakticky jakémkoliv typu aplikace. Google novinku přestavil jako možnost, jak mohou vývojáři tvořit pohlcující 3D aplikace bez nutnosti znalostí komplikovaných knihoven, jako je například OpenGL.

Google v prvním roce od uvedení ARCore zvládl dodat vývojářům řadu aktualizací. Za další významnou verzi ARCore po 1.2 se dá považovat 1.6 [45]. Mezi novinkami lze najít například realističtější nasvícení Sceneform scén. Předchozí verze Sceneform se blížily při vykreslování teplejším odstínům světla, zatímco od verze 1.6 je výchozí barva neutrální. Google rovněž přidal možnost nahrávání Sceneform scén a pořizování obrázků právě zobrazené scény na žádosti vývojářů. Tvůrci aplikací tak mohli mnohem snadněji nahrát video ukázky své aplikace nebo snímky obrazovky pro zobrazení v obchodu Play Store. Stejně tak mohli uživatelé těchto aplikací snadno sdílet ukázky pomocí obrázků či videích na sociálních sítích. Verze 1.6 s sebou dále přinesla rychlejší a přesnější rozpoznávání ploch.

Společně s vydáním ARCore 1.6 Google oznámil, že je knihovna dostupná na 250 milionech zařízení [45]. Předchozí oficiální číslo bylo 100 milionů, zveřejněné koncem roku 2017. Vývojářům se opět rozrostly řady potenciálních uživatelů jejich augmentovaných aplikací.

V době psaní (březen 2019) je aktuální verze ARCore 1.7, kterou Google uvedl 15. února 2019 [46]. Největší novinkou je podpora detailního snímání obličeje uživatelů pomocí přední kamery. Aplikace tak mohou na obličej uživatelů detailně zobrazit libovolný virtuální obsah a například je proměnit v superhrdiny nebo jiné známé postavy. Google rovněž přidal podporu animací pro Sceneform. Detailní snímání obličeje může být největší novinkou, vývojáři ale pravděpodobně nejvíc ocení přidání ARCore Elements. Jedná se o hotové prvky augmentovaných aplikací, které většina využívá a nově je nemusí vývojáři implementovat sami. Google v oznamovacím blogovém příspěvku vypíchl hlavně rozpoznávání horizontálních

a vertikálních ploch a také manipulaci s 3D objekty pomocí známých gest. Vývojáři tak mohou velmi snadno umožnit uživatelům, aby například virtuální objekt pomocí tažení prstu po displeji přesunuli, otočili nebo ho „pinch“ gestem zvětšili.

Tabulka 2 - Vlastnosti ARCore v kostce

Vývojář	Google
Vydání verze 1.0	Únor 2018
Platforma	Android od verze 7.0*, iOS od verze 11
Jazyk	Java, Kotlin, Objective-C
Cena	Zdarma

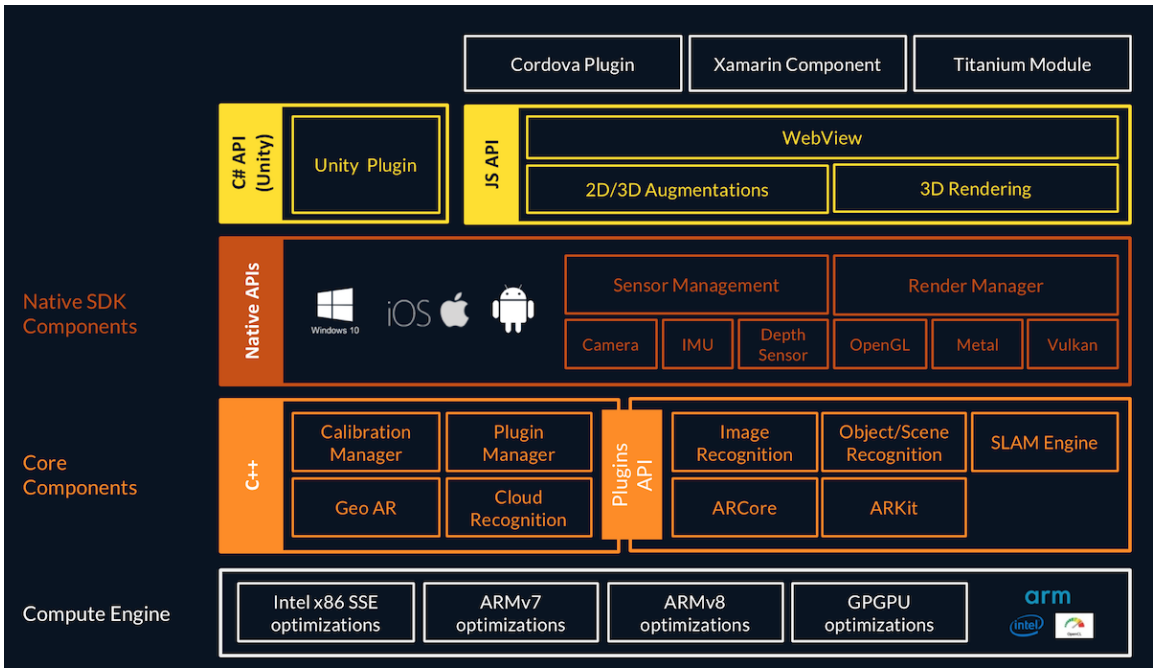
* Některá zařízení vyžadují vyšší verzi systému Android.

4.4 Wikitude

Vývojářské nástroje Wikitude SDK se od dříve popisovaných knihoven ARKit a ARCore liší v mnoha ohledech. Německá firma Wikitude GmbH působí v této oblasti od roku 2008 [47], kdy vydala svoji první aplikaci s augmentovanou realitou. Za více než deset připravila do svého SDK spoustu možností a velkou výhodou Wikitude je podpora širokého množství zařízení. Vývojáři mohou produkt využít k vývoji mobilních aplikací pro platformy iOS a Android, a podporovaný je také operační systém Windows (pro využití u tabletů). Wikitude nabízí rovněž podporu pro chytré brýle, tvůrci specificky zmiňují Epson Moverio a Vuzix, jako produkty, pro které je Wikitude optimalizované [48].

Další výhodou tohoto nástroje je relativní svoboda při vývoji, jelikož si vývojáři mohou zvolit, v jakém jazyku budou aplikace za pomoci Wikitude vytvářet. Pro Android a iOS je Wikitude dostupné jednak jako JavaScript API a také nativní API, což znamená možnost využití jazyk Java, respektive Objective-C pro tyto platformy. Wikitude dále nabízí svoje SDK pro platformy Unity, Cordova, Xamarin a Titanium, které se zaměřují (hlavně v případě tří posledních) na cross-platform mobilní vývoj.

Pozici Wikitude jako všestrannou AR platformu umocňuje ještě podpora platforem React Native, Ionic, Adobe Air a Qt by V-play. Možnost jsou skutečně široké. Tvůrcům je navíc dostupná aplikace 3D Encoder pro systémy Windows a macOS, sloužící k vytvoření souborů wt3 pro 3D modely, které Wikitude využívá.



Obrázek 2 - Architektura Wikitude SDK [49]

Wikitude SDK nabízí vývojářům řadu hotových funkcí, které jsou připraveny k použití. Mezi ně patří rozeznávání konkrétních objektů, místností nebo scén. Dále rozpoznání 2D obrázků s možností sledování jejich pohybu. SDK rovněž obsahuje možnosti, jak tyto rozpoznané obrázky rychle doplnit o multimediální obsah, jako je třeba HTML obsah, obrázky, zvuk a další. Wikitude v sobě zahrnuje dále podporu pro augmentované aplikace pracující s polohou uživatele. Firma na svém webu uvádí navigaci, hledání obchodů či geolokační hry jako možnosti využití [50].

Pokročilou funkcí, kterou se může Wikitude pochlubit, je tzv. Extended Tracking. Umožní rozpoznat určitý objekt a pokračovat ve snímání i poté, co původní objekt není v záběru kamery. Jako příklad užití uvádí tvůrci možnost rozpoznat část komplikovaného stroje a následně zobrazit informace i mimo původní oblast rozpoznání. Poslední z velkých funkcí je Cloud Recognition pro rozpoznání velkého množství obrázků, které jsou uloženy na serverech, a aplikace jim posílá zdrojová (respektive nasnímaná data).

Wikitude SDK je placený produkt, což je pro „hobby“ vývojáře poměrně velká bariéra, jelikož předchozí zmiňované knihovny jsou dostupné zdarma. Základní SDK PRO stojí 1990€ a obsahuje licenci na jednu aplikaci pro iOS, Android a Windows. Obsahuje navíc pouze základní funkce. SDK je možné získat také jako roční předplatné a nejlevnější varianta SDK PRO stojí 2490€ na rok, platí stejné podmínky jako u SDK PRO zakoupeného jednou, výhodou navíc jsou aktualizace zdarma. Také dražší předplatná obsahují licenci pouze pro jednu aplikaci

na každé z platforem a při překročení milionu stažení je nutné přejít na Enterprise plán předplatného, jehož cena je individuální [51].

Tabulka 3 - Vlastnosti Wikitude SDK v kostce

Vývojář	Wikitude
Vydání verze 1.0	Říjen 2008
Platforma	Android od verze 4.4, iOS od verze 9, Windows 10 (od 1803)
Jazyk	Java, Objective-C, JavaScript
Cena	Placené*

* Až na výjimku pro startupy placené, spousta variant viz text výše.

4.5 EasyAR

EasyAR představuje další knihovnu umožňující vytvářet augmentované aplikace na obě hlavní mobilní platformy. EasyAR SDK nabízí nativní API pro platformu iOS ve starším Objective-C a také novém jazyce Swift. Vývojářům Android aplikací je dostupné Java API. Kromě toho obsahuje EasyAR podporu pro několik verzí Unity3D [52].

Ačkoliv pro kompletní funkcionalitu je nutné zakoupit licenci, verze zdarma označená jako EasyAR SDK Basic nabízí poměrně bohaté možnosti pro tvorbu aplikací s augmentovanou realitou. Nechybí rozpoznávání a sledování obrázků s možností sledovat více cílů zároveň, dále možnost rozpoznání ploch, podporu pluginů pro 3D engine nebo nástroje pro využití cloudové služby [52].

Placená verze označená jako Pro přidává možnost rozpoznat a sledovat 3D objekty, možnost nahrávat displej pro snadné vytváření ukázek aplikací s augmentovanou realitou a možnost rozpoznávat a sledovat více typů obsahu zároveň.

EasyAR může být i ve verzi zdarma vhodným nástrojem pro vývoj aplikací s augmentovanou realitou, pokud je jedním z hlavních požadavků vyvinutí aplikace pro obě mobilní platformy. Tedy pro iOS a Android.

Tabulka 4 - Vlastnosti EasyAR v kostce

Vývojář	VisionStar
Vydání verze 1.0	Říjen 2015
Platforma	Android od verze 4.0, iOS od verze 7, Windows od verze 7, macOS
Jazyk	Swift, Objective-C, Java, Kotlin, C, C++
Cena	Zdarma, Pro verze placená (499$)

4.6 Porovnání popsaných AR knihoven pro vývoj mobilních aplikací

Tabulka níže stručně srovnává popsané knihovny pro vývoj aplikací využívajících augmentovanou realitu. Srovnány jsou z programátorského hlediska, to znamená dostupné platformy, na které je možné aplikace s knihovnami vytvářet, dostupné programovací jazyky a v neposlední řadě také cenu a dostupné materiály.

Tabulka 5 - Srovnání vybraných AR knihoven

	ARKit	**ARCore**	**Wikitude**	**EasyAR**
Platforma	iOS	Android, iOS	Android, iOS, Windows, brýle	Android, iOS, Windows, macOS
Programovací jazyk	Swift, Objective-C	Java, Kotlin, Objective-C	Java, Objective-C, JavaScript	Java, Kotlin, Swift, Objective-C, C, C++
Cena	Zdarma	Zdarma	Placené	Zdarma/placené
Dostupné materiály (v tisících)*	227	509	56	15

* Provedeno vyhledávání na Google v anonymním okně s dotazem složeným z názvu knihovny a slova „tutorial“.

5 NÁVRH A REALIZACE MOBILNÍ APLIKACE

Praktická část práce popisuje tvorbu mobilní aplikace podpořenou webovou částí, která slouží pro administraci obsahu. První kapitoly praktické části se věnují motivaci a prvotnímu návrhu fungování mobilní aplikace, později následuje její implementace a popis důležitých částí. Webové části je věnováno méně prostoru, protože se nezabývá přímo problematikou augmentované reality, je však instrumentální pro praktické využití mobilní aplikace.

5.1 Motivace pro výběr tématu

Hlavním důvodem, pro výběr tématu, byl zájem autora o oblast augmentované reality. Jednak se ji okrajově věnoval v bakalářské práci [53], kde pomocí GPS souřadnic zobrazoval pozice zastávek pardubického MHD, a také autora obecně baví sledovat vývoj technologií. O programování aplikací s augmentovanou realitou se autor začal zajímat po představení knihovny ARKit, protože v té době již tvořil aplikace pro systém iOS.

Různých mobilních aplikací s augmentovanou realitou autor zkusil a viděl na videích spousta, většinou dokázaly rychle zaujmout, zároveň ale šlo v mnoha případech pouze o stylové a chytré ukázky, které si nešlo dobře představit pro reálné použití.

Z důvodu zájmu o obor chtěl autor v rámci diplomové práce vytvořit mobilní aplikaci postavenou na rozšířené realitě a zároveň se pokusit ukázat, jak může vypadat praktické využití a jak může augmentovaná realita skutečně přinést uživatelům přidanou hodnotu v porovnání s obdobnými aplikacemi nevyužívající možnosti augmentované reality.

5.2 Prvotní návrh mobilní aplikace

Vůbec první prototyp aplikace uměl načíst jednu vizitku pardubické japonské restaurace King Yo. Konkrétní vizitka byla náhodou po ruce a snadno se dalo vymyslet, jaký augmentovaný obsah přidat. Aplikace v prvních verzích zobrazovala po stranách vizitky fotku exteriéru restaurace, jídla a mapu s vyznačenou pozicí. Tento prototyp se stal základem pro celou práci a stejnou vizitku autor používal v průběhu vývoje a demonstracích.

Prvotní prototyp fungoval dobře, ale bylo třeba vyřešit jedno výrazné omezení. Obrázky pro rozpoznání musely být připravené v zařízení a stejně tak veškerý augmentovaný obsah, který by aplikace měla zobrazovat. Jednalo se o velmi neflexibilní řešení. Další fáze návrhu tedy znamenala přijít se způsobem, jak rozpoznávaný a augmentovaný obsah aplikace spravovat dynamicky, bez zásahu do zdrojového kódu aplikace.

Právě z tohoto důvodu došlo k rozhodnutí vytvořit webovou aplikaci, která by umožnila obsah intuitivně přidávat a zároveň ho poslat do mobilní aplikace pomocí API. Ve výsledku by uživateli stačilo změnit obsah k rozpoznání v administračním rozhraní, bez znalosti jakéhokoliv programovacího jazyka a nutnosti, jakkoliv zasahovat do mobilní aplikace.

V tomto stádiu vývoje zbývalo vyřešit poslední problém limitující praktickou využitelnost aplikace. Apple doporučuje při využití rozpoznávání obrázků pomocí ARKit, aby jich bylo maximálně 25 [54]. U vyššího počtu může docházet k problému s výkonem či špatného rozeznání, pokud bychom měli několik vizuálně velmi podobných obrázků. Pomyslným posledním dílkem skládanky se tak stalo řešení tohoto limitu na počet obrázků k rozpoznání.

Nakonec bylo rozhodnuto využít standardu iBeacon od Apple [55], který slouží k určení pozice uživatele primárně uvnitř budov, protože zde je většinou problém s tradičním určováním polohy pomocí GPS. iBeacon je jednoduchý standard pro Bluetooth zařízení, pomocí kterého lze realizovat navigaci uvnitř budov nebo určit, v jaké části budovy se zhruba uživatel nachází.

Integrace iBeacon znamenala, že limit 25 obrázků pro rozpoznávání už nemusel platit pro celou aplikaci, ale pouze pro jeden iBeacon. Mobilní aplikace by tedy mohla v prvním kroku najít nejbližší iBeacon a podle něj si z API webové služby získat obrázky k rozpoznání přiřazené právě konkrétnímu iBeacon. S tímto řešením prakticky odpadl limit na počet obrázků, který mobilní aplikace zvládne rozpoznat a zobrazit k nim obsah. Stačí vhodně rozmístit iBeacon zařízení a korektně k nim přiřadit obrázky. Například galerie obrazů by mohla pomocí této aplikace zobrazit virtuální obsah pro všechny své exponáty za předpokladu vhodného rozmístění iBeacon vysílačů.

Finální systém tedy umožní ve webovém rozhraní přidat záznamy pro jednotlivé iBeacon zařízení a následně k nim přidat obrázky k rozpoznání. Kromě toho bude existovat kategorie pro „nezařazené“, kde může být uloženo až 25 obrázků pro rozpoznání v případě, že mobilní aplikace nenalezne v okolí žádný iBeacon. Mobilní aplikace nejdříve provede snímání možných iBeacon vysílačů a při nalezení si z API webové služby stáhne obrázky pro ten nejbližší. Kromě toho bude využívat systému cache, aby při každé změně detekovaného iBeacon nemusela stahovat nové obrázky, protože tato data jsou nejnáročnější pro síťový přenos z hlediska objemu.

6 MOBILNÍ APLIKACE

Mobilní aplikace pro systém iOS představuje hlavní část diplomové práce, proto ji bylo věnován odpovídající prostor, zvlášť v porovnání s webovou částí. Aplikace vyžaduje operační systém iOS 12, což je v době psaní aktuální verze systému, protože právě iOS 12 vyžaduje stěžejní knihovna ARKit ve verzi 2.0. Uživatelé iOS zařízení na nové verze přecházejí poměrně rychle a Apple vydal systém iOS 12 dokonce pro iPhone 5S, který se začal prodávat v roce 2013, takže v tomto směru by nemělo jít o velké omezení [56]. Ačkoliv právě ARKit vyžaduje alespoň iPhone 6S, tedy o dvě generace modernější zařízení, než je model 5S.

6.1 Použité technologie

Aplikace je naprogramována v jazyce Swift, ve verzi 4.2, která byla aktuální v době vývoje. iOS aplikace je rovněž nativně možné programovat ve starším jazyku Objective-C, ten ale není tak přívětivý pro programátora a jedná se o minulost Apple platforem. Proto byl zvolen jazyk Swift, který nabízí všechny funkce moderního programovacího jazyka. Jako IDE byl využit program Xcode. Jedná se prakticky o jedinou možnost, jak vytvářet iOS aplikace. Lze sice sehnat jiná IDE (jmenovitě AppCode od firmy JetBrains [57]), ty se ale opět spoléhají na Xcode pro sestavení aplikace ze zdrojového kódu a komunikaci s iOS zařízením. Xcode rovněž nabízí vývojářům simulátory pro většinu iOS zařízení, konkrétně ale augmentované aplikace nelze takto zkoušet, protože vyžadují fotoaparát a rovněž senzory pohybu zařízení. Pro pomoc při vývoji byla zvolena dvojice populárních knihoven od vývojářské komunity, konkrétně Alamofire a SwiftyJSON. Obě byly nainstalovány pomocí balíčkovacího systému CocoaPods, který je zároveň umí udržovat aktuální. Alamofire nabízí intuitivní a rozsáhlé možnosti pro komunikaci s webovými službami a rovnou tyto činnosti provádí na pozadí, takže není využito hlavní vlákno aplikace, které by ji mohlo, při pomalé odezvě serveru, zablokovat. SwiftyJSON slouží k parsování dat ve formátu JSON na datové typy jazyka Swift. Nabízí velmi granulární možnosti, jak data parsovat, řešit chybějící hodnoty a podobně.

Datový model je popsaný podrobně v části zabývající se webovou administrací, protože zde dochází k jeho správě. Mobilní aplikace tato data pouze čte a ukládá do modelových tříd a struktur.

6.2 Úvodní stránka aplikace

Protože mobilní aplikace společně s webovou částí slouží primárně k demonstraci, jak vytvořit aplikaci s augmentovanou realitou a dynamickým obsahem, je tomu podřízen i design úvodní stránky iOS aplikace. Barevně se aplikace drží stejného schématu, jaký využívá webová část, pro dodání pocitu sjednocení uživateli. Hlavní tlačítko s titulkem „Scan Images“ je nejvyšší a slouží k přesunu na druhou obrazovku, kde je možné rozpoznávat připravené obrázky a dochází ke zobrazování virtuálního obsahu. Následuje tlačítko „Locate Beacons“, které spustí hledání dostupných iBeacon zařízení v okolí. Hned pod ním je umístěna komponenta UILabel zobrazující stav hledání iBeacon zařízení. Jakmile dojde k nalezení zařízení (v případě více výsledků je využito to nejbližší), je tato skutečnost zobrazena uživateli, který vidí název iBeacon zařízení zadaný v administrační části pro lepší orientaci, než kdyby byl zobrazen třeba unikátní identifikační řetězec, který iBeacon zařízení interně používají.

Následující dvě tlačítka jsou umístěna vedle sebe a slouží k explicitnímu stažení obsahu k rozpoznání. První „For beacon“ je aktivní pouze v případě, že se aplikaci povedlo detekovat iBeacon zařízení v okolí a po stisku stáhne obrázky k rozpoznání, které jsou přiřazeny právě tomuto iBeacon zařízení. Druhé „Uncategorized“ stáhne všechny obrázky k rozpoznání, které nejsou ve webové části přiřazené k žádnému iBeacon zařízení a dají se považovat za univerzální. To se velmi dobře hodí k rychlému testování či demonstraci aplikace.

Zbytek úvodní stránky aplikace zabírá komponenta UICollectionView, která po stažení obrázků k rozpoznání zobrazí jejich náhled a název. Slouží hlavně k tomu, aby uživatele ubezpečila, že se podařilo stáhnout nějaké obrázky k rozpoznání a ukáže, co vlastně může uživatel rozpoznat. V opačném případě by musel vše zkoušet metodou pokus omyl.

6.3 Vlastní iBeacon zařízení pomocí Raspberry Pi

Aby bylo možné korektně otestovat funkčnost aplikace vztahující se k detekci iBeacon vysílačů, byl vysílač vytvořen pomocí všestranného mini počítače Raspberry Pi. Konkrétně šlo o model Zero W, který obsahuje zabudované WiFi a Bluetooth moduly, takže je snadnější na použití než tradiční Zero, ke kterému je třeba tyto síťové modely připojit.

Na Raspberry Pi běžel operační systém Raspbian speciálně vytvořený pro tyto počítače, jak už prozrazuje název. Pro samotné vysílání iBeacon signálu je klíčová rozsáhlá knihovna BlueZ, která nabízí všechny důležité Bluetooth vrstvy a protokoly s modulární architekturou.

Po instalaci knihovny (je nutné provést její kompilaci pomocí *sudo make* a *sudo make install* příkazů) je k dispozici nástroj hciconfig v adresáři tools. Právě ten slouží k samotnému vysílání iBeacon signálu.

Pomocí následujících příkazů bylo provedeno jeho zapnutí a prvotní konfigurace:

```
sudo tools/hciconfig hci0 up

sudo tools/hciconfig hci0 leadv 3

sudo tools/hciconfig hci0 noscan
```

Následně je možné provést konfiguraci iBeacon a nastavit unikátní identifikátor UUID, který slouží jako primární identifikátor iBeacon společně s hodnotami major a minor. Hodnota major se používá k přiřazení iBeacon vysílačů do skupin a hodnota minor potom pro jejich identifikaci v rámci skupiny.

Příkaz byl použit následující:

```
sudo tools/hcitool –i hci0 cmd 0x08 0x0008 1E 02 01 1A 1A FF 4C 00
02 15 63 6F 3F 8F 64 91 4B EE 95 F7 D8 CC 64 A8 63 B5 00 00 00 00 C8
```

Samotné UUID je zvýrazněno tučně. Čtyři dvojice nul za tučným UUID identifikátorem slouží k nastavení major a minor hodnot.

Tímto bylo nastavení Raspberry Pi Zero W jako testovacího iBeacon vysílače dokončeno [58]. Pro verifikaci nastavení byla použita macOS aplikace Beacon Scan dostupná zdarma z Mac App Store [59].

6.4 Detekce iBeacon v prostředí iOS

Jak již bylo popsáno, aplikace umožňuje na úvodní obrazovce provést lokaci iBeacon zařízení z dosahu Bluetooth signálu. Protože se jedná o určitý druh zjištění polohy zařízení a iOS je v tomto směru zaměřené silně na soukromí uživatelů, je prvním krokem modifikace souboru Info.plist v projektu.

V tomto souboru je nutné přidat klíč *Privacy - Location When In Use Usage Description* a vyplnit pro něj hodnotu. Jedná se o text, který systém iOS zobrazí uživateli, jakmile aplikace požádá o práva zjišťovat polohu, v tomto případě tedy iBeacon zařízení. Tímto může vývojář snadno vysvětlit, proč daná práva vyžaduje.

Po přidání klíče už je možné v aplikaci zahájit detekci iBeacon zařízení. Celá funkcionalita je zapouzdřena ve třídě BeaconService. Aby bylo možné použít systémovou třídu

CLLocationManager, je nutné použít protokol CLLocationManagerDelegate, čímž se třída BeaconService stává delegátem CLLocationManager a může tak být informovaná o událostech. V tomto případě detekci iBeacon zařízení.

Třída BeaconService si hned po vytvoření stáhne, z API webové aplikace, seznam všech přidaných iBeacon zařízení a v případě prvního spuštění požádá o práva k získávání polohy. Jakmile dojde ke stažení, spustí se jejich lokace pomocí metody *startScanning*. Metoda nejdříve pomocí vlastnosti CLLocationManager.authorizationStatus zkontroluje, jestli uživatel souhlasil a přidělil práva, případně o ně požádá znovu a až po případném udělení začne iBeacon zařízení lokalizovat.

Pro jistotu metoda kontroluje v podmínkách ještě systémové metody *CLLocationManager.isMonitoringAvailable* a *CLLocationManager.isRangingAvailable*, jestli je skutečně možné na daném zařízení lokalizovat iBeacon. Následně je využita metoda *startMonitoring* instance CLLocationManager, která umožňuje současně hledat až 20 specifikovaných iBeacon zařízení, což je pro potřeby aplikace dostatek.

Protože je třída BeaconService delegátem CLLocationManager, dozví se o zjištěných iBeacon zařízeních z implementované metody *func locationManager(_ manager: CLLocationManager,* ***didRangeBeacons*** *beacons: [CLBeacon], in region: CLBeaconRegion).* Metoda následuje konvence dané přímo Apple pro názvy metod používaných pro delegáty, proto ji lze jednoznačně identifikovat až z parametrů. Podobně na iOS prakticky fungují všechny systémové třídy využívající návrhového vzoru delegát.

Po úspěšné detekci iBeacon zařízení stačí použít to nejbližší, pokud jich aplikace nalezla více, a informovat o této skutečnosti uživatele. Ten může následně pomocí tlačítka tyto obrázky stáhnout.

6.5 Komunikace s API webové služby

Úvodní obrazovka aplikace má, kromě detekce iBeacon zařízení, ještě na starost získání dat z API webové služby. Konkrétně obrázky k rozpoznání, virtuální obsah k zobrazení a případně ještě uživatelské akce k tomuto obsahu. Celou funkcionalitu zapouzdřuje třída BackendService.

BackendService obsahuje primárně metody pro stažení potřebných dat z API ve formátu JSON a několik pomocných, které například slouží k připravení správných URL a podobně. S třídou je úzce spjatá pomocná třída ImageCache, která slouží jako velmi jednoduchá cache pro obrázky, aby je nebylo nutné stahovat při každém zapnutí aplikace. ImageCache využívá metoda *loadImage*, která nejdříve zkusí obrázek načíst právě z cache, a pokud se to nepovede

(tedy obrázek ještě stažený a uložený nebyl), dojde ke stažení přímo z webové služby. Metoda *loadImage* je využívána na všech místech, kde dochází ke stahování obrázků, takže cache může fungovat efektivně a v případě budoucích úprav stačí upravit tuto metodu a není nutné provádět další zásahy do zdrojového kódu aplikace.

Pro získání JSON dat z API používá třída BackendService na iOS zavedenou knihovnu Alamofire pro snadnější stahování dat. Knihovna je nainstalovaná pomocí balíčkového systému CocoaPods, stejně jako SwiftyJSON sloužící pro parsování JSON dat, pokud chce programátor pokročilé možnosti a větší kontrolu.

Pro jednodušší získání JSON dat z webového API se mnohdy vyplatí nepoužívat pokročilejší knihovny a místo toho využít dostupné systémové prostředky v čele s protokolem Codable. Struktury nebo třídy, které adoptují tento protokol, mohou být snadno převedeny z JSON dat na instance konkrétního typu pomocí třídy JSONDecoder. V BackendService je tento postup využit pro stahování akcí pro virtuální obsah v metodě *downloadContentActions*. Jako parametr očekává obrázek k rozpoznání, později jsou akce mapované na jednotlivý virtuální obsah, který je pro tento obrázek přidělen. Důležitá část metody běží ve vlákně na pozadí, aby nedošlo k zablokování uživatelského rozhraní. K získání JSON dat je využit jeden z konstruktorů systémové třídy Data (slouží k reprezentaci arbitrárních binárních dat), který jako parametr vyžaduje URL. Následně jsou tato data dekódována na pole struktur ContentAction, které reprezentují interaktivní akce, které může uživatel provést. Struktura ContentAction je podrobně popsána v další kapitole.

7 HLAVNÍ OBRAZOVKA APLIKACE

Hlavní obrazovka iOS aplikace slouží k rozpoznávání obrázků a přes její celou plochu je umístěn náhled hlavního fotoaparátu. K zobrazení náhledu slouží komponenta ARSCNView speciálně vytvořená pro potřeby aplikací s augmentovanou realitou. Automaticky vykresluje v reálném čase obraz z fotoaparátu, stará se o snímání scény a aktualizuje pozici SceneKit kamery. SceneKit je knihovna od Apple pro vývoj 3D her a slouží právě také k zobrazování 3D obsahu v aplikacích využívající augmentovanou realitu. Alternativně je možné použít knihovnu SpriteKit, původně vytvořenou pro snadný vývoj 2D her.

Obrazovku doplňuje samostatná komponenta view controller, vložená jako potomek, která pomocí standardních UIKit komponent zobrazuje status, jako třeba rozpoznaný obrázek nebo varování, že není dostatek světla a dvojici tlačítek. Jedno restartuje rozpoznávání, tedy smaže současný obsah scény a druhé slouží k ukončení rozpoznávání a návrat na úvodní obrazovku aplikace.

O hlavní obrazovku aplikace se stará třída ScannerViewController, která implementuje protokol ARSCNViewDelegate. Díky tomu může přijímat informace o událostech z výše popsané komponenty ARSCNView. Při načtení je v metodě *viewDidLoad*, která je zavolána automaticky okamžitě po načtení všech komponent dané obrazovky, nastaven delegát a zaregistrována instance třídy UITapGestureRecognizer, sloužící k zachytávání dotyku uživatele na displeji mobilního zařízení. Použita je pro interakci s virtuálním obsahem.

Samotné snímání scény a detekce obrázků k rozpoznání je spuštěna v metodě *viewDidAppear*. Ta je zavolána automaticky vždy, když se daná obrazovka objeví a stane se aktivní obrazovkou aplikace. Snímání spouští metoda *resetTracking*, která se používá rovněž pro restart snímání. Kromě toho je zde nastavena globální vlastnost *UIApplication.shared.isIdleTimerDisabled* na *true*, aby po určitém čase nedocházelo k automatickému ztmavení a vypnutí displeje, pokud není registrována interakce uživatele s displejem, což se při využití pouze snímání obrázků může stát.

Metoda *resetTracking* v první řadě načte obrázky k rozpoznání ze třídy BackendService, která zajišťuje komunikaci s API, tedy s webovou službou. Popsána je později. Z těchto obrázků vytvoří instance specializované třídy ARReferenceImage, která popisuje obrázky, jenž může ARKit knihovna rozpoznat. Je třeba nastavit obrazová data, orientaci obrázku a také fyzické rozměry v metrech (stačí šířka, výška je vypočítána). Výsledné obrázky jsou přidané do Setu a pomocí Dictionary mapované na původní instance třídy ImageToRecognize, která o obrázcích k rozpoznání uchovává více informace, především potom virtuální obsah, který je

zobrazen. Pomocí Dictionary lze po rozpoznání obrázku typu ARReferenceImage snadno získat zdrojový ImageToRecognize a z něj virtuální obsah a ten zobrazit.

Metoda poté ještě vytvoří novou konfiguraci typu ARImageTrackingConfiguration, které v dalším kroku předá výše vytvořený set ARReferenceImage objektů k rozpoznání. Všechny možné konfigurace pro AR aplikace jsou popsány níže. Následně je ARSession, zaštiťující hlavní funkce ARKit, spuštěna s vytvořenou konfigurací a parametry *resetTracking* a *removeExistingAnchors,* které se starají právě o restart snímání a odstranění dříve detekovaných kotev. Ty jsou vysvětleny dále.

Nejdůležitější metodou této obrazovky je *func renderer(_ renderer: SCNSceneRenderer,* ***didAdd*** *node: SCNNode, for anchor: ARAnchor)*. Je dostupná, jelikož je třída ScannerViewController delegátem pro ARSCNView. Metoda je automaticky zavolána v případě, že dojde k detekci námi zvolených obrázků z metody *resetTracking*. Následuje rozhodovací krok, který zjistí, jestli má detekovaný obrázek zobrazit obsah na celou obrazovku, nebo zobrazit virtuální obsah v prostoru pomocí augmentované reality. Protože je zobrazení obsahu na celou obrazovku jednodušší, je v podmínce zanořena právě tato situace. Samotné zobrazení obsahu je popsáno v další kapitole, metoda v tomto případě končí.

Pokud má daný obrázek zobrazit vedle sebe augmentovaný obsah, metoda pokračuje dále. Nejdříve je pomocí konstrukce *guard* zjištěno, jestli je daná kotva ARAnchor skutečně specializovaný typ ARImageAnchor pro rozpoznané obrázky. Tento objekt obsahuje pozici obrázku v prostoru a zároveň je dále snímán, což umožňuje, aby se virtuální obsah pohyboval společně s fyzickým zdrojovým obrázkem, který byl rozpoznán podle vzoru.

Metoda dále z této kotvy zjistí konkrétní ARReferenceImage a pomocí dříve vytvořeného Dictionary je získán původní zdrojový ImageToRecognize. Následuje podmínka pro kontrolu, že se skutečně podařilo získat instanci ImageToRecognize z Dictionary, a proměnná díky *if let* konstrukci není dále typu Optional a snadněji se s ní pracuje.

V další části metoda z BackendService získá veškerý virtuální obsah k zobrazení. Opět je využito Dictionary pro rychlé nalezení pomocí id obrázku. Všechen tento obsah je následně postupně zpracován v cyklu.

Cyklus nejdříve vytvoří novou instanci objektu SCNNode, který slouží pro umisťování augmentovaného obsahu do prostoru. O vytváření SCNNode a konfiguraci se stará specializovaná třída NodeFactory, jejíž design je volně inspirován návrhovým vzorem Factory. Fungování NodeFactory je detailně popsáno níže.

Jakmile je nová instance třídy SCNNode vytvořena a nakonfigurována, dojde k nastavení její vlastnosti *opacity* na 0. Tím je zajištěno, že není vidět, protože bude následně animována.

Uživatel získá příjemnější zážitek z používání aplikace, protože obsah se okamžitě neobjeví, ale místo toho je plynule animován efektem, jako by vyjížděl zpod rozpoznaného fyzického obrázku.

7.1.1 Dostupné konfigurace pro ARKit

ARKit nabízí několik konfigurací, se kterými je možné spustit AR session a ovlivnit, jak následné snímání prostoru funguje. Tabulka níže stručně popisuje všechny dostupné typy v ARKit 2.0.

Tabulka 6 - Dostupné ARKit konfigurace [60]

Konfigurace	**Popis**
AROrientationTrackingConfiguration	Základní konfigurace. Snímá pouze orientaci zařízení v prostoru, ne však jeho pohyb
ARWorldTrackingConfiguration	Všestranná konfigurace umožňující snímat orientaci zařízení, pohyb v prostoru, detekovat horizontální/vertikální plochy či obrázky nebo objekty
ARImageTrackingConfiguration	Konfigurace specializovaná na rozpoznání obrázků a následné snímání jejich polohy. Je využita v aplikaci
ARFaceTrackingConfiguration	Konfigurace využívající přední kameru pro precizní snímání obličeje uživatele. Vyžaduje zařízení s TrueDepth kamerou, tedy iPhone X a vyšší
ARObjectScanningConfiguration	Konfigurace pro detailní skenování objektů. Výsledná data mohou být následně využita k rozpoznání těchto objektů pomocí konfigurace AROrientationTrackingConfiguration

7.1.2 Animování virtuálního obsahu

O animování nově vytvořených instancí SCNNode, které zobrazují samotný virtuální obsah, se stará třída NodeAnimator. Je implementována podle návrhového vzoru Singleton, protože dává velký smysl mít pouze jednu instanci. Instance se jmenuje shared, což je doporučený název od Apple, podobně jako current nebo standard, které často používají systémové třídy v případě využití tohoto návrhového vzoru.

Třída NodeAnimator disponuje pouze jednou metodou *animateAugmentedContentDisplay*, která se o samotné animování stará. Jako parametry očekává konkrétní virtuální obsah (jako instanci AugmentedContent) a samozřejmě konkrétní instanci SCNNode, na které jsou animace prováděny. Volitelně je možné specifikovat vlastní délku animace, ve výchozím stavu se používá hodnota 0,3 sekundy. Tato hodnota byla zvolena na základě empirického testování. Animace netrvá zbytečně moc dlouho, a zároveň není moc rychlá na to, aby ji uživatel nestihl zaregistrovat.

Metoda nejdříve pomocí konstrukce *if let* zkontroluje, že má daná instance SCNNode vlastnost geometry nastavenou na instanci SCNPlaneWithTapAction. Právě z této instance jsou následně zjišťovány rozměry virtuálního obsahu, aby bylo možné nastavit původní pozici, ze které bude obsah animován a také na jakou startovní pozici bude konkrétní SCNNode přesunuta po své ose x.

Animace umístění v prostoru (tedy na ose x) a animaci plynulého zobrazení zajišťují instance systémové třídy SCNAction, které nabízí vývojáři snadný způsob, jak definovat znovupoužitelné animační bloky a následně je spouštět na instancích SCNNode.

Metoda tedy připraví jednu instanci SCNAction pro animování vlastnosti opacity (průhlednosti) z 0 na 1, čímž je virtuální obsah plně zobrazen a druhou pro posun po ose x, k čemuž slouží metoda *moveBy* dostupná na SCNAction. Tato druhá instance SCNAction je vytvářena v závislosti, na jaké pozici rozpoznaného obrázku je virtuální obsah zobrazen. Například pokud je virtuální obsah zobrazen na pravé straně rozpoznaného obrázku, je třeba ho před začátkem animace posunou doleva a následně animovat posun vpravo.

Následuje pouze spuštění obou definovaných instancí SCNAction, které obě animace zajistí. Jedná se o velmi příjemný animační framework, nabízející široké možnosti pouze několika řádky zdrojového kódu.

Tabulka níže stručně představuje široké možnosti SCNAction pro manipulaci se SCNNode.

Tabulka 7 - Vybrané metody SCNAction pro animace [61]

Metoda	**Popis**
moveBy	Relativní posunutí vzhledem k aktuální pozici
move(to:	Posunutí na novou pozici z té aktuální
rotateBy	Rotace po osách x, y a z, relativní k aktuální orientaci
rotateTo	Rotace po x, y nebo z bez ohledu na aktuální orientaci
scale(by:	Změna velikosti vzhledem k té aktuální. Například hodnota 2 zvětší daný objekt dvakrát
scale(to:	Změna velikosti bez ohledu na aktuální velikost. Jako škála 1 se bere velikost při prvotním vytvoření
fadeIn	Změna viditelnosti (opacity) na hodnotu 1.0, tedy na plnou viditelnost
fadeOut	Změna viditelnosti na 0.0, obsah tedy kompletně zmizí
fadeOpacity(to:	Změna viditelnosti na vybranou hodnotu
fadeOpacity(by:	Relativní změna viditelnosti na vybranou hodnotu
group	Slouží k seskupení dalších SCNAction, které mohou být následně spuštěny současně
sequence	Opětovné seskupení SCNAction objektů, které jsou ovšem spouštěny v sekvenci. Často kombinované s group
repeat	Slouží k opakování libovolné SCNAction, dostupná rovněž metoda repeatForever pro nekonečné opakování

wait	Slouží k „čekání“ po určitou dobu, často používané společně se sequence
removeFromParentNode	Slouží k odstranění objektu ze scény, často používané jako poslední prvek sequence
reversed	Obrátí průběh SCNAction
run	Umožňuje provést jakýkoliv kód. Opět často jako součást group nebo sequence

Tabulka není vyčerpávající. SCNAction nabízí ještě další metody, často také variace na výše popsané. Všechny jsou popsané v oficiální dokumentaci od Apple [61].

7.1.3 Okluze virtuálního obsahu

Aby výše popsané animace uvěřitelně fungovaly, je třeba využít techniky známé jako okluze. Díky ní je možné virtuální obsah schovat za skutečné předměty a tím umocnit dojem, že opravdu do skutečného světa patří. Použitá animace by ani nemohla bez okluze fungovat, protože by bylo virtuální obsah vidět okamžitě, jak se posouvá přes rozpoznaný obrázek.

Okluze v aplikaci je zajištěna hned na začátku výše popsané metody *func renderer(_ renderer: SCNSceneRenderer,* ***didAdd*** *node: SCNNode, for anchor: ARAnchor).* Jedním z prvních kroků, po rozpoznání obrázku a získání jeho pozice v prostoru, je vytvoření neviditelné instance SCNNode, která právě zprostředkovává okluzi.

O vytvoření se stará metoda *createOcclussionNode* třídy NodeFactory (funkcionalita této třídy je podrobně popsána později). Metoda vytvoří instanci SCNPlane pro zobrazení 2D obsahu s rozměry odpovídajícími rozpoznanému obrázku. Klíčové je vytvoření speciální instance SCNMaterial, která jako svůj obsah používá barvu CIColor.Black (ze systémového frameworku Core Image) a má nastavenou masku *colorBufferWriteMask* na nulovou hodnotu pomocí *SCNColorMask(rawValue: 0)*. Touto technikou je docíleno, aby žádný jiný virtuální obsah nebyl za rozpoznaným obrázkem vidět a dříve popsaná animace může vypadat efektně.

7.2 Zobrazení obsahu na celou obrazovku

Zobrazení definovaného HTML obsahu je provedeno pomocí speciální systémové komponenty SFSafariViewController, která umí načíst URL adresu na samostatné obrazovce.

První podmínkou je, aby třída, která tuto komponentu používá, implementovala protokol SFSafariViewControllerDelegate. Je použit pouze k tomu, aby se bylo možné dozvědět, že uživatel ukončil prohlížení tohoto obsahu a pomocí dostupné systémové metody *dismiss* je komponenta schována, načež aplikace zobrazí předchozí obrazovku.

O zobrazení obsahu na celou obrazovku se stará metoda *showWebsiteInSafari* očekávající jako jediný parametr URL. Webová služba má definovaný konkrétní formát, takže stačí v parametru URL poslat ID rozpoznaného obrázku a stránka automaticky vykreslí veškerý přiřazený HTML obsah a pouze ten. Výsledek tak připomíná spíše připravený interaktivní dokument než tradiční webovou stránku.

Metoda *showWebsiteInSafari* pouze vytvoří novou instanci SFSafariViewController, nastaví aktuální view controller jako delegát této instance a pomocí metody *present* vytvořenou instanci uživateli zobrazí.

Výhoda použití SFSafariViewController nespočívá pouze v jednoduché implementaci pro programátora. Uživatel získá povědomé uživatelské rozhraní systémového prohlížeče Safari, může využít tlačítka pro sdílení a webovou stránku si třeba uložit nebo někomu poslat. Rovněž ji může otevřít přímo v Safari aplikaci.

7.2.1 Další možnosti pro zobrazení obsahu na celou obrazovku

K výše popsanému zobrazení obsahu by šlo využít také metodu *open* na sdílené instanci UIApplication, pomocí které lze otevřít URL (přímo v prohlížeči Safari) či třeba začít hovor. Tyto techniky jsou použity ve třídě ContentAction, která slouží k reprezentaci uživatelských akcí po výběru virtuálního obsahu, a jsou popsány v pozdější kapitole.

Pokud chce programátor větší kontrolu nad zobrazením webového obsahu ve své aplikaci, může využít komplexní komponentu WKWebView z balíčku WebKit, která nabízí robustní možnosti a pro vykreslování webových stránek používá stejný vykreslovací engine jako prohlížeč Safari. V takovém případě je nutné vytvořit vlastní novou komponentu view controller a WKWebView do ní umístit. Široké možnosti komponenty WKWebView namátkově zahrnují možnost nastavit povolené adresy, monitorovat načítání webových stránek, spouštět vlastní JavaScript kód nebo pracovat s cookies.

8 VYTVÁŘENÍ VIRTUÁLNÍHO OBSAHU K ZOBRAZENÍ

Hlavní metoda NodeFactory pojmenovaná *createAugmentedContentNode* očekává na vstupu ARReferenceImage, tedy rozpoznaný obrázek a instanci AugmentedContent obsahující data virtuálního obsahu k zobrazení. Vrací potom instanci SCNNode připravenou k zobrazení. Metoda v prvním kroku vypočítá vhodné rozměry pro virtuální obsah. Pokud má být virtuální obsah zobrazen na levé nebo pravé straně, vezme se jako základní rozměr výška rozpoznaného obrázku a dopočítá se pomocí poměru stran nová šířka. V případě zobrazení virtuálního obsahu na horní nebo dolní pozici se použije šířka rozpoznaného obrázku a dynamicky se dopočítá výška.

Při výpočtu rozměrů je ještě třeba počítat se situací, kdy je virtuální obsah typu mapa, protože neobsahuje obrázek, a tudíž ani poměr stran, ze kterého by šlo přepočítat rozměry. V tomto případě se opět bere buď šířka, nebo výška rozpoznaného obrázku, ale ta se použije také pro výšku, respektive šířku, takže je ve výsledku mapa vykreslena jako čtverec.

Po přepočítání rozměrů následuje využití vestavěné metody *assert*, která slouží ke kontrole libovolného výrazu s návratovou hodnotou typu Bool, pokud je výraz vyhodnocen jako nepravdivý (tedy *false*), dojde k pádu aplikace, ovšem pouze při vývoji. Programátor je tak velmi výrazně upozorněn na chybu. *Assert* zde slouží k ověření přepočítaných rozměrů, aby bylo okamžitě zajištěno, že chybou výpočtu nevyšla 0 nebo NaN.

V dalším kroku metoda vytvoří instanci SCNPlaneWithTapAction s využitím výše vypočítaných rozměrů. Jedná se vytvořeného potomka třídy SCNPlane doplněné o vlastnost *tapAction* typu Optional closure bez parametrů a návratové hodnoty. Díky tomu je možné uložit libovolnou metodu přímo tomuto objektu a následně ji využít při interakci uživatele s těmito objekty.

Následuje samotné nastavení virtuálního obsahu k zobrazení na vytvořené instanci SCNPlaneWithTapAction. U jednoduchého obrázku stačí nastavit tento obrázek nahraný v instanci UIImage jako vlastnost *firstMaterial?.diffuse.contents*. U galerie je využita metoda typu AugmentedContent, protože je nutné evidovat index zobrazeného obrázku, aby bylo možné zobrazit další obrázek v pořadí po interakci uživatele s tímto virtuálním obsahem. Pro zobrazení videa je využita pomocná metoda *createVideoContent*, které je předána vytvořená instance SCNPlaneWithTapAction a také odkaz na video. Ten pochází z API a odkazuje na webovou část, ze které je video později streamováno. Metoda je popsána níže. Rovněž pro vykreslení mapové komponenty MapView je využita pomocná metoda *createMapView*, opět je popsaná později.

V dalším kroku už dochází k vytvoření instance SCNNode, které je v konstruktoru předána dříve vytvořená a upravená instance SCNPlaneWithTapAction. Po vytvoření je ještě modifikována rotace na ose x, aby byl přidaný obsah orientován směrem ke kameře. Pokud se jedná o video obsah, je v dalším kroku ještě provedena rotace o 180°, protože se jedná o SpriteKit obsah a ve výchozím stavu je horizontálně převrácený. Jelikož už se jedná o modifikaci instance SCNNode, není možné tento krok provést v metodě *createVideoNode*, protože ta pracuje s instancí SCNPlaneWithTapAction.

Skoro posledním krokem je nastavení výsledné polohy virtuálního obsahu a dynamické vypočítání a nastavení odsazení virtuálního obsahu od rozpoznaného obrázku pro lepší estetický dojem. Slouží k tomu metoda *setContentPositionWithOffset*. Odsazení je počítáno ze šířky nebo výšky (podle toho, která hodnota je vyšší) a empiricky bylo zjištěno, že příjemného výsledku lze dosáhnout vydělením šířky nebo výšky konstantou 25.

Následně metoda skončí a vrátí instanci SCNNode s potřebnou konfigurací. Nyní může metoda *func renderer(_ renderer: SCNSceneRenderer,* ***didAdd*** *node: SCNNode, for anchor: ARAnchor)* pokračovat a přidat nově vytvořenou SCNNode jako potomek proměnné node, což je v tomto případě automaticky vytvořená instance SCNNode pro rozpoznaný obrázek s korektní pozicí v prostoru. V tomto kroku již dochází k zobrazení virtuálního obsahu na displeji mobilního telefonu.

8.1 Implementace metody createVideoContent

Pro přehrávání videa je využita technologie SpriteKit, která obsahuje třídu SKVideoNode. Pro vytvoření SKVideoNode lze použít instanci třídy AVPlayer a ta je zase vytvořena s parametrem URL odkazující na zobrazované video. Následně stačí pro SKVideoNode vytvořit scénu SKScene, nastavit ji rozměry a pozici, protože SpriteKit používá odlišný souřadnicový systém a následně tuto scénu nastavit jako dříve zmíněnou vlastnost *firstMaterial?.diffuse.contents* instanci SCNPlaneWithTapAction.

Metoda před svým koncem ještě nastaví vlastnost *tapAction* předané instanci SCNPlaneWithTapAction. Obsah této closure je jednoduchý. Využívá totiž konceptu closure capturing a tímto „zachytí“ referenci na instanci AVPlayer pomocí tzv. weak capturing, aby náhodou nedošlo k vytvoření retain cycle, který by bránil korektní dealokaci objektů. Tato closure pouze mění vlastnost *isMuted* instance AVPlayer pomocí metody *toggle* dostupné na typech Bool. Ve výsledku tak může uživatel dotykem na video v prostoru augmentované reality buď zvuk vypnout, nebo znovu zapnout.

Původně bylo v plánu využít vlastnost *tapAction* pro spuštění a případné přerušení přehrávání videa, uživatel by ale v takovém případě byl nucen provést dotyk pokaždé, když by si chtěl pustit video. Jelikož předtím pomocí mobilního zařízení rozpoznal konkrétní obrázek s nastaveným videem, dá se předpokládat, že ho daný obsah zajímá a video chce vidět. Kromě toho aplikace respektuje systémové vypnutí zvuků, v takovém případě zvuk videa není slyšet v žádném případě.

8.2 Implementace metody createMapView

Metoda na vstupu očekává instanci AugmentedContent, ze které zjistí potřebné parametry pro zobrazení mapy a také SCNPlaneWithTapAction, na které je výsledná komponenta MapView vykreslena. V prvním kroku se metoda pokusí zjistit souřadnice pomocí metody *tryGetCoordinates*. Následně je využito služeb systémového API Grand Central Dispatch a pomocí volání metody *DispatchQueue.main.async* je zbytek proveden na hlavním vlákně aplikace. Je to z důvodu omezení iOS, které vyžaduje, aby se všechny činnosti týkající se uživatelského rozhraní prováděly na hlavním vláknu aplikace, kam spadá mimo jiné vytvoření nové komponenty MapView.

Pro MapView je v prvním řadě vytvořen UIView jako kontejner s pevně danými rozměry, které se shodují s rozměry MapView, které je nutné nastavit přes vlastnost frame. Při použití MapView v tradičních aplikacích tento krok není nutný, v augmentované realitě se ale MapView, bez explicitního nastavení rozměrů, nezobrazí. Konfigurace MapView komponenty následně zahrnuje také nastavení vlastnosti *isUserInteractionEnabled* na *false*, což znemožní uživatelské interakce, protože ty jsou nastaveny zvlášť podle hodnot získaných z API. Pokud se v předchozím kroku povedlo získat souřadnice, je na těchto souřadnicích zobrazena anotace a souřadnice se automaticky stávají středem zobrazené mapy. Následně stačí vytvořený kontejner, kterému byla nastavena MapView komponenta jako potomek, nastavit jako vlastnost *firstMaterial?.diffuse.contents* předané instanci SCNPlaneWithTapAction. To stačí, aby byla komponenta mapy vykreslena v prostředí augmentované reality.

8.3 Interakce s virtuálním obsahem

Aplikace podporuje uživatelskou interakci se zobrazeným obsahem. Ten je detailně popsaný v části popisující webovou aplikaci pro administraci obsahu, protože právě tam dochází k jeho přidávání a nastavení.

iOS aplikace v prvním kroku musí detekovat dotyk prstu na displeji mobilního zařízení. K tomu slouží systémová třída UITapGestureRecognizer, která je registrovaná hned po načtení hlavní obrazovky aplikace, a při vytvoření mimo jiné očekává parametr action specifikující metodu, která se má provést po detekci „tap gesta“ na displeji mobilního zařízení. Metoda se jmenuje *tapped* a je dekorována @objc atributem, který je při použití #selector vyžadován. Atribut označuje pro kompilátor, že je tato metoda viditelná pro Objective-C kód, a může být tedy zavolána třídou UITapGestureRecognizer napsanou právě v Objective-C jako celý iOS systém.

Metoda nejdříve získá SCNView z vlastnosti *view* UITapGestureRecognizer a následně pomocí metody location stejné třídy zjistí, kde na displeji došlo k dotyku. Posléze je využita metoda *hitTest* instance SCNView, která zajišťuje zobrazení scény augmentované reality, a pomocí této metody lze zjistit, jestli je na místě dotyku nějaký virtuální obsah. Metoda vrací pole typu SCNHitTestResult. Pokud není prázdné, vezme se první prvek a pomocí vlastnosti node zjistíme objekt SCNNode, kterého se uživatel „dotkl“. Tímto je část detekce u konce a stačí provést danou akci.

Implicitní akce, jako je přepnutí obrázku v galerii nebo vypnutí zvuku u přehrávaného videa, jsou nastavené přímo konkrétní instanci SCNPlaneWithTapAction, která je uložena jako vlastnost *geometry* objektu SCNNode, který je po dotyku zjištěn. Pro volitelné akce, které se přidávají ve webové části, je vytvořena modelová struktura ContentAction.

8.3.1 Struktura ContentAction pro uživatelsky definované akce

Struktura v první řadě využívá protokol Codable a enum CodingKeys, který slouží k mapování mezi Snake case zápisem používaným ve webové verzi a tím také v JSON datech poslaných z API a mezi Camel case, které používá Swift a další silně typové jazyky. Vlastnost *actionType* struktury ContentAction je tak mapovaná na *action_type* z JSON data a při dekódování použije JSONDecoder automaticky definované CodingKeys, aby správně načetl hodnoty do jednotlivých vlastností.

Struktura ContentAction dále obsahuje pomocnou metodu pro získání GPS souřadnic ze String vlastnosti *actionValue*, takže není nutné mít pro každý typ ContentAction také odpovídající vlastnosti a *actionValue* může obsahovat URL adresu k otevření či třeba e-mailovou adresu pro poslání e-mailu.

Hlavní metodou struktury je poté *execute* s volitelným parametrem typu UIViewController, který je nutný pro posílání e-mailů, jelikož to je prováděno pomocí speciálního systémového

view controlleru a ten jen třeba „prezentovat“ z nějaké zdrojové instance view controlleru aplikace.

Metoda obsahuje konstrukci *switch*, jenž řeší všechny hodnoty, kterých může enum ActionType nabývat. Pro připomenutí se jedná o otevření webové stránky, vytočení čísla, poslání e-mailu na konkrétní adresu a spuštění navigace pomocí systémové aplikace Apple Maps na konkrétní GPS souřadnice.

Otevření konkrétní URL adresy je v prostředí iOS a jazyce Swift velmi jednouché. Nejdříve dojde k vytvoření URL adresy pomocí konstrukce *if let* pro případ, že by vlastnost actionValue neobsahovala validní data a následně je tento objekt předán sdílené instanci třídy UIApplication reprezentující spuštěnou aplikaci. Pro otevření URL je využita metoda *open*, po jejímž zavolání dojde ke spuštění prohlížeče Safari a načtení konkrétní URL adresy. Volání je pro jistotu provedeno na hlavním vlákně, kdyby náhodou metoda *execute* byla zavolána z jiného než hlavního vlákna.

Podobně jednoduše, jako otevření URL adresy, funguje vytočení čísla. K číslu je nutné přidat speciální prefix tel:// a následně tuto URL předat opět metodě *open* na sdílené instanci UIApplication. Při vývoji bylo zjištěno, že občas telefonní číslo pocházející z API obsahuje na první pozici netisknutelný Unicode znak, kvůli kterému nefungoval převod na URL. Ještě před vytvářením konkrétní URL tak dochází k odstranění možných netisknutelných znaků pomocí metody *replacingOccurrences* na datovém typu String. Konkrétně metoda hledá znaky \\p{Cf} a nahrazuje je prázdným znakem, čímž je prakticky odstraňuje.

Posílání e-mailu z aplikací je v systému iOS řešeno pomocí speciální komponenty MFMailComposerViewController, kterou je třeba patřičně nakonfigurovat. Pro tyto účely byla vytvořena třída EmailHelper, jenž se o konfiguraci, zobrazení komponenty a její schování stará. EmailHelper je inicializován s parametry view controller, protože je třeba MFMailComposerViewController z konkrétního view controlleru zobrazit a s adresou příjemce. Hned po inicializaci dojde ke konfiguraci MFMailComposerViewController a jeho zobrazení.

V metodě *execute* struktury ContentAction je třeba vytvořenou instanci EmailHelper nastavit jako volitelnou vlastnost na ScannerViewController, aby se držením její reference zabránilo okamžité dealokaci. Pokud by nebyla reference uchována, dojde okamžitě k uvolnění paměti systémem ARC a zobrazený MFMailComposerViewController nepůjde po odeslání emailu nebo zrušení akce skrýt, protože nedojde k zavolání metody jeho delegáta. Jedná se bohužel o krkolomné řešení, bylo ale vyhodnoceno jako nejméně problematické.

Posledním případem ve *switch* konstrukci metody *execute* je spuštění navigace. Tato část se nejdříve pokusí získat GPS souřadnice ze String vlastnosti actionValue pomocí metody *getCoordinatesFromValue*. Pokud se získání GPS souřadnic povede, je vytvořena instance systémové struktury CLLocationCoordinate2D, která je následně použita k vytvoření MKCoordinateRegion. Souřadnice se použije jako střed tohoto regionu a další parametry určují jeho šířku a výšku v metrech. Následně dojde k vytvoření instance MKPlacemark sloužící k vytvoření MKMapItem, který může být konečně použit pro spuštění navigace. Právě instance MKMapItem obsahuje metodu *openInMaps* s možností specifikování parametrů, která spouští samotnou navigaci na zvolenou GPS souřadnici.

8.4 Použití mobilní aplikace

Tato sekce velmi stručně popisuje základní použití mobilní aplikace. Předpokládá rovněž, že byly pomocí webové aplikace korektně přidány iBeacon zařízení a přiřazeny k nim obrázky k rozpoznání včetně virtuálního obsahu k zobrazení. Pokud není využito iBeacon zařízení, předpokládá se, že jsou přidané nějaké obrázky k rozpoznání bez přiřazeného iBeacon zařízení, čímž spadají do kategorie nezařazených a tyto obrázky mají přidaný nějaký virtuální obsah.

Pro použití mobilní aplikace s využitím iBeacon zařízení je nutné po jejím spuštění provést detekci těchto zařízení pomocí tlačítka „Locate beacons“. Stačí chvíli počkat a sledovat informační text pod tlačítkem, který bude, v případě úspěšné detekce iBeacon zařízení, o této skutečnosti informovat a pro kontrolu zobrazí také název daného zařízení, který byl nastaven ve webové části.

Po úspěšné detekci iBeacon zařízení je možné pokračovat stažením obrázků k rozpoznání, k čemuž slouží tlačítko „For beacon“ v sekci označené „Download images“. Na korektní stažení obrázků upozorní sama aplikace. Obrázky budou zobrazeny přímo pod touto sekcí, včetně náhledů, takže má uživatel možnost vidět, jaké obrázky může rozpoznat.

Následně je možné pomocí tlačítka „Scan Images“ pokračovat na hlavní stránku aplikace, kde je zobrazen hledáček fotoaparátu, a okamžitě je možné začít rozpoznávat obrázky. Po detekci obrázku se po jeho stranách zobrazí přidaný virtuální obsah. Dotykem displeje v místě obsahu je možné tento obsah „vybrat“ a provede se daná akce. To může být například zobrazení dalšího obrázku v galerii, které jsou indikovány textem s aktuálním obrázkem a celkovým počtem, například takto: „1/4“.

Pokud uživatel nemá v plánu využít iBeacon zařízení, celý postup je jednodušší. Na úvodní stránce stačí v sekci „Download images“ zvolit tlačítko „Uncategorized“, načež budou z webové služby stáhnuty všechny obrázky, které nejsou přiřazené k žádnému iBeacon zařízení. Opět budou zobrazeny níže pod touto sekcí. Nyní stačí pokračovat jako v případě využití iBeacon zařízení a zvolit tlačítko „Scan Images“, čímž se uživatel dostane na hlavní stránku aplikace a může začít obrázky rozpoznávat.

9 WEBOVÁ ČÁST

Webová část slouží k administraci obsahu, konkrétně tedy ke správě iBeacon zařízení, obrázků k rozpoznání a obsahu, který bude zobrazen v augmentované realitě. Tato část práce popisuje použité technologie, implementaci webové části a její nejdůležitější části.

9.1 Použité technologie

Webová část je napsaná v jazyce Python za pomoci rozšířeného frameworku Django. Django bylo zvoleno z důvodu poměrně rychlého vývoje webových aplikací a předchozích zkušeností autora. Několik webů za pomoci tohoto frameworku autor již vytvořil a úspěšně nasadil na servery. Pro hostování bylo rozhodnuto využít vlastní virtuální server s operačním systémem Ubuntu. Hlavní důvody byly dva. Jednak pro Django existuje málo možností, jak aplikaci hostovat, a vlastní server nabízí mnohem větší flexibilitu a kontrolu nad celým řešením. Vlastní server autor bral také jako možnost, jak prohloubit své znalosti unixových systémů a nasazení webové aplikace od samotného počátku.

Ve výsledku byla na serveru využita poměrně populární databáze PostgreSQL, která je s Django často používaná, což znamená dostupné materiály či vyřešené problémy jiných uživatelů třeba na StackOverflow. Pro samotné rozběhnutí aplikace byl zvolen http server Gunicorn napsaný v jazyce Python. Opět se jedná o časté řešení pro Django aplikace. Poslední komponentou webové architektury je potom server Nginx. Slouží k předávání požadavků Gunicornu a navíc se stará o statické soubory.

Kromě frameworku Django byl pro tvorbu webové aplikace využit rovněž populární CSS framework Boostrap ve čtvrté verzi, který výrazně zrychluje tvorbu stránek webové aplikace. Hlavně díky grid systému pro pozicování prvků, dále vhodným CSS třídám pro obrázky nebo formulářové komponenty. Dostupná je rovněž řada témat, takže je možné jednou změnou CSS souboru proměnit design webové aplikace. Pro autora to znamenalo, že nemusel vymýšlet design a tvořit vlastní CSS styly, což není při občasném použití této technologie snadný úkol. Dále byl využit rozšířený JavaScript framework jQuery pro asynchronní odesílání formulářů a zobrazování modálních oken, aby webová část působila více jako aplikace než tradiční webové stránky.

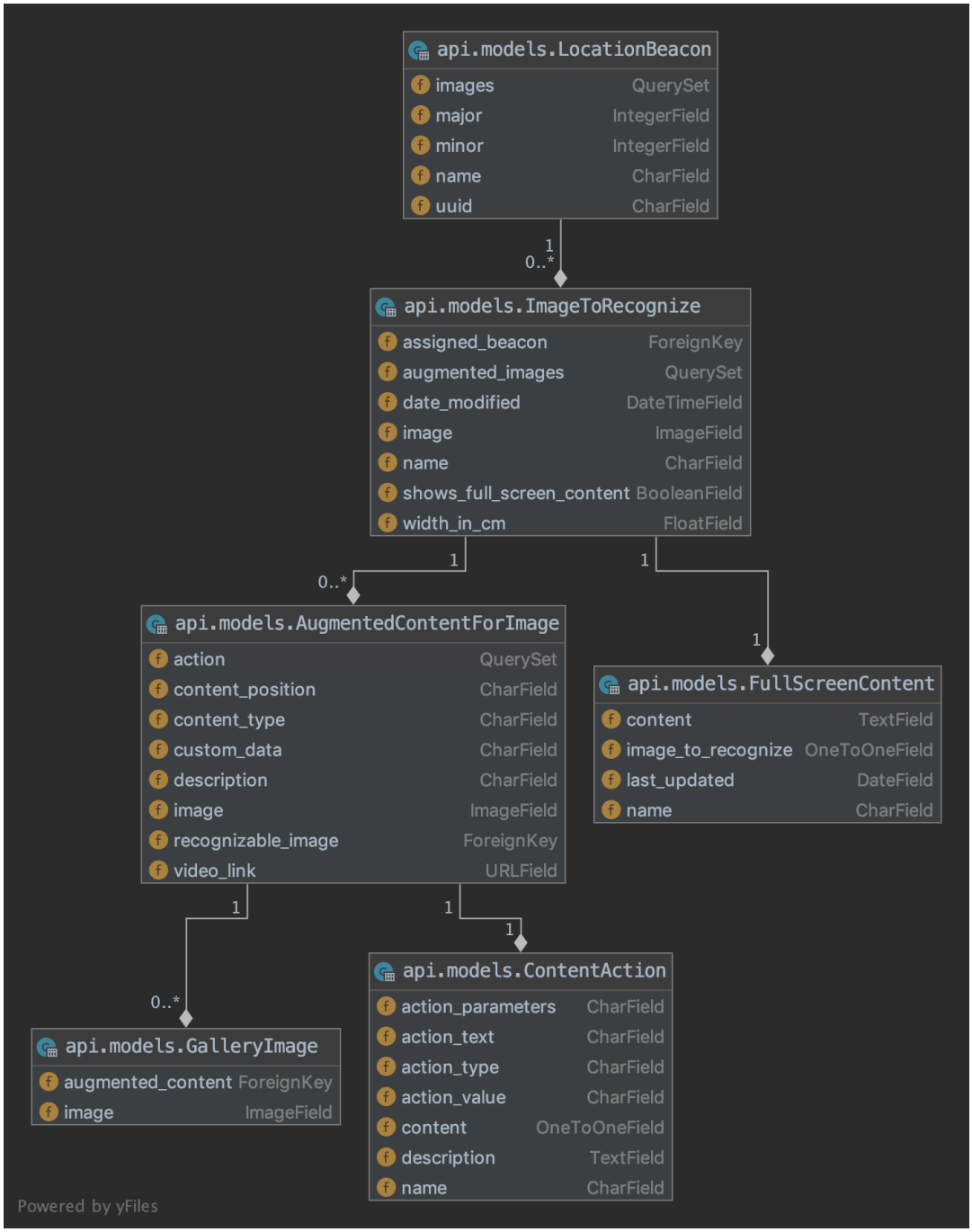
Webovou část byla vyvíjena v IDE PyCharm od firmy JetBrains, která nabízí vývojářské nástroje pro celou řadu jazyků. Minimálně pro Python není k dispozici velké množství IDE a PyCharm byl pro potřeby projektu perfektní. Nabízí podporu frameworku Django, takže například pomůže se snadným vytvořením virtual environment pro Python balíčky, se

spouštěním projektu, verzováním přes Git a mnoho dalšího. PyCharm má autor zaplacený již třetím rokem a může ho vřele doporučit.

9.2 Datový model

Tato část stručně popisuje datový model, včetně jeho UML diagramu. Jeho strukturu úzce kopírují datové třídy a struktury v samotné iOS aplikaci. Datový model webové služby je definován v aplikaci api, v souboru models.py, což je standardem u Django aplikací. Vždy se jedná o definici třídy, která dědí z Django třídy Model.

9.2.1 UML diagram



Obrázek 3 - UML diagram datového modelu

9.2.2 ImageToRecognize

Jedna z klíčových tříd pro fungování celého systému. Slouží k uložení obrázků, které mobilní aplikace umí rozpoznat. Ukládá hlavně samotná obrazová data, název obrázku pro orientaci, délku v centimetrech (aby uživatel nemusel přepočítávat na metry používané interně iOS aplikací), dále také obsahuje přepínač, jestli zobrazuje obsah na celý displej zařízení a volitelný cizí klíč odkazující na LocationBeacon.

9.2.3 LocationBeacon

Třída sloužící k ukládání iBeacon zařízení. Umožňuje uložení názvu pro snadnou uživatelskou orientaci, dále identifikátoru UUID a major & minor verze, což jsou standardní vlastnosti iBeacon a podle nich dochází k rozpoznání těchto zařízení.

9.2.4 AugmentedContentForImage

Další z pilířových tříd pro reprezentaci datového modelu. Slouží k uložení virtuálního obsahu, který je zobrazen po rozpoznání obrázku. Kromě samotných obrazových dat je důležitý popisek pro orientaci při správě, dále také pozice a typ obsahu. Nechybí cizí klíč vázající jednotlivé záznamy na konkrétní ImageToRecognize. Protože může třída reprezentovat odlišné typy obsahu, nabízí nepovinné záznamy pro odkaz na video, a ještě volitelná data, která jsou aktuálně používána pro uložení GPS souřadnic. Do budoucna by mohla ukládat jakýkoliv jiný řetězec, pokud by byla aplikace rozšířena o možnost přidat a následně zobrazit nové typy virtuálního obsahu.

Kromě vlastností má tato třída definovaný pár pomocných metod. Jedna slouží k rychlému zjištění, jestli daný typ podporuje vlastní akce po výběru uživatelem, pro snadnější zjištění této informace v šabloně. Druhá slouží k načtení GPS souřadnic z textového řetězce.

9.2.5 GalleryImage

Velmi jednoduchá modelová třída, která je úzce svázaná s AugmentedContentForImage. Slouží k ukládání obrázků pro případné galerie. Kromě obrazových dat obsahuje pouze cizí klíč, pomocí kterého je navázána právě na konkrétní AugmentedContentForImage. Více pro funkčnost galerie není třeba.

9.2.6 ContentAction

Třída sloužící k reprezentaci nastavitelných akcí po uživatelském výběru zobrazeného virtuálního obsahu. Primárně obsahuje typ konkrétní akce, 1:1 vazbu na konkrétní AugmentedContentForImage, a dále, kromě názvu a volitelného popisku, také další potřebné informace pro provedení akce. To je primárně její hodnota, což může být webová adresa, telefonní číslo nebo GPS souřadnice, v závislosti na vybrané akci.

9.2.7 FullScreenContent

Další z jednoduchých modelových tříd. Slouží pro případ, kdy uživatel vybere, že chce místo virtuálního obsahu, zobrazeného pomocí augmentované reality, zobrazit uživateli stránku s obsahem po rozpoznání obrázku. Obsahuje primárně název, samotný HTML obsah a 1:1 vazbu na ImageToRecognize.

10 STRUKTURA WEBOVÉ APLIKACE

Framework Django webové aplikace dělí na tzv. sites a apps. Site je celý jeden projekt a podle něj se jmenuje nejen hlavní adresář celého projektu, ale také podadresář obsahující základní důležité soubory v jazyce Python. Jmenovitě to je settings.py a urls.py. První jmenovaný obsahuje veškerou důležitou konfiguraci pro samotnou webovou aplikaci. Namátkou jsou zde nastaveny vyžadované Django moduly, aktivují se zde doinstalované balíčky, nastavují adresáře pro šablony či přístupy do databází. Jedná se také o místo s nastavením lokalizace, kořenové adresy pro statické soubory a mnoho dalšího. Druhý soubor urls.py je o poznání jednoduší a slouží k nastavení relativních adres, které webová aplikace umí zpracovat. Typicky mají aplikace svoje vlastní urls.py soubory a ty jsou hromadně přidány do tohoto hlavního, často s prefixem podle názvu aplikace, aby nedocházelo ke kolizím. Webová aplikace se skládá ze dvou Django aplikací pojmenovaných *api* a *cms*.

10.1 API

Aplikace api definuje datové modely, které jsou v Django aplikacích umístěny v souboru models.py. Jedná se prakticky o Python definice tříd, které dědí od speciální třídy Model umístěné v modulu *django.db*. Aplikace se dále primárně stará o odpovědi na příchozí požadavky z mobilní aplikace a ty posílá zpět ve formátu JSON.

Zpracování požadavků se v Django aplikacích typicky děje v souboru views.py a jinak tomu není ani v případě api aplikace. Pomocí třídy JsonResponse z modulu *django.http* vrací aplikace data ve formátu JSON. Podobně snadno lze pomocí Django využít formát XML. Zvlášť v iOS aplikacích se ale s JSON pracuje lépe.

Kromě odeslání JSON dat řeší aplikace ještě zobrazení obsahu pro obrázky k rozpoznání, které mají nastavený jako typ „Fullscreen content“. K tomu je využita jednoduchá šablona, která kromě tohoto obsahu nic dalšího neobsahuje. Samotné metody ve views.py využívají metodu *render* dostupnou z modulu *django.shortcuts*. Ta slouží k propsání dat předaných jako datový typ Dictionary do připravené šablony.

10.2 CMS

O poznání objemnější aplikace, než dříve popsaná aplikace api, má na starost kompletní administraci webové služby a tím pádem také obsahu, který následně api posílá mobilní iOS aplikaci.

Naprostá většina funkcionality se odehrává v souboru views.py, který definuje jednotlivé stránky webové aplikace a zpracovává odeslané formuláře. Část logiky, kde to dává smysl, je umístěna v jednotlivých šablonách, které najdeme v projektovém adresáři templates ve složce cms.

Protože je veškerá administrační část určena pouze přihlášeným uživatelům, jsou všechny metody ve views.py opatřeny atributem @login_required. V případě, že se na stránku pokouší dostat nepřihlášený uživatel, je automaticky přesměrován na formulář k přihlášení. Pro správu uživatelů je využita dostupná funkcionalita Django frameworku, což platí i pro přihlašovací formulář.

Django aplikace automaticky nabízejí základní administraci, kde je možné kompletně zpravovat všechny databázové záznamy. Stačí nejprve zaregistrovat vybrané modelové třídy v souboru admin.py, který ve výchozím stavu nabízí všechny vytvořené Django aplikace. Protože je tato administrace vytvořena obecně, aby fungovala pro všechny modely, není zrovna intuitivní pro složitější datový model s více vazbami. Z tohoto důvodu byla celá administrační část vytvořena od základu. Díky tomu může webová aplikace nabídnout intuitivní nahrání obrázků k rozpoznání (včetně náhledů) a grafické přidání virtuálního obsahu.

Soubor urls.py aplikace cms obsahuje veškeré definice dostupných relativních adres, které aplikace používá. Použité formuláře jsou definované v souboru forms.py, kde je možné specifikovat, jaké atributy se použijí či další meta konfiguraci. Například formulář FullScreenContentForm, pro přidávání a úpravu obsahu na celou obrazovku, má v tomto souboru definovaný widget SummernoteWidget, což je nutné pro implementaci populárního WYSIWYG editoru Summernote pro tvorbu HTML obsahu pro Django. Pro jeho použití stačí nainstalovat balíček *django-summernote*, přidat ho v souboru settings.py mezi aplikace, přidat v projektovém souboru urls.py záznam pro jeho adresy a potom nastavit jako widget v definici formuláře. Rázem se z obyčejného textového pole stane plnohodnotný editor s formátováním, možností vložit obrázky, video a mnoho dalšího.

V průběhu vývoje byla vyzkoušena řada WYSIWYG editorů, včetně velmi populární volby TinyMCE, nakonec se ale Summernote ukázalo jako nejlepší volba pro Django a tento editor plánuje autor používat i v budoucnu.

Aplikace umožňuje formuláře odesílat na server asynchronně, což pro uživatele znamená práci s modálními dialogy, jejichž potvrzení nevyžaduje načtení celé stránky znovu. Této funkcionality je docíleno pomocí jazyka JavaScript a populární knihovny jQuery. Relevantní soubory jsou umístěny v adresáři *static/js* aplikace cms. Nejobjemnější je image-edit.js, který se stará o stránku umožňující uživateli upravovat obrázky k rozpoznání, respektive přidávat

virtuální obsah, mazat tento obsah či přidat akce po výběru obsahu, pokud se jedná o obrázek nebo mapu. Tento soubor se stará primárně o asynchronní odesílání formulářů, zobrazení obsahu po přidání bez nutnosti nového načtení stránky a o zobrazení náhledového obrázku při přidávání YouTube videa.

11 FUNGOVÁNÍ WEBOVÉ APLIKACE

Webová aplikace vyžaduje pro zobrazení či úpravy obsahu oprávnění, které je zajištěno pomocí uživatelských účtů. Django v sobě obsahuje základní systém pro tvorbu uživatelů a také řešení jejich práv, včetně přihlašovací obrazovky. Pro přístup do webové aplikace byl tedy využit tento systém a pokud není uživatel zrovna přihlášen, je aplikací přesměrován k přihlášení a až poté může pokračovat na jiné stránky webové aplikace.

Django zároveň nabízí vygenerovanou administraci pro všechny datové modely, které programátor vytvoří v souborech models.py a následně zaregistruje v souborech admin.py. Protože se jedná o univerzální a základní řešení pro nejrůznější typy obsahu a datové modely, není pro intuitivní práci příliš vhodné. Právě proto bylo vytvořeno vlastní uživatelské rozhraní pro kompletní zprávu obsahu webové části.

11.1 Úvodní stránka webové aplikace

První stránka dostupná po přihlášení slouží k výpisu všech přidaných iBeacon zařízeních a v tabulce zároveň zobrazuje všechny vyplněné parametry. Kromě toho umožňuje pomocí modálního okna přidat nové iBeacon zařízení či ta stávající upravovat nebo mazat. Zde se také nachází kategorie pro obrázky k rozpoznání, které nejsou přiřazené k žádnému iBeacon zařízení.

Na této úvodní stránce si uživatel vybere, kam chce nový obrázek k rozpoznání mobilní aplikací přidat nebo jaké chce upravovat. Po výběru konkrétního iBeacon zařízení nebo již zmiňované kategorie pro nezařazené se zobrazí pouze ty obrázky k rozpoznání, které spadají pod dané iBeacon zařízení, nebo nejsou zařazené v případě této volby.

11.2 Stránka s výpisem obrázků k rozpoznání

Na této stránce je možné přidávat nové obrázky, které mobilní aplikace rozpozná, a spravovat ty stávající. To znamená hlavně přidávat obsah, který je následně zobrazen pomocí augmentované reality. Také je možné vybrat typ zobrazené obsahu, který nabízí dvě možnosti. Buď je možné zobrazit virtuální obsah v prostoru po stranách rozpoznaného obrázku, nebo zobrazit uživateli na celý displej připravenou webovou stránku právě pro tento obsah. Typy se hodí v závislosti na zvoleném obrázku k rozpoznání.

11.3 Stránka pro přidání obsahu k zobrazení v augmentované realitě

Na této stránce mohou uživatelé intuitivně přidat virtuální obsah k zobrazení vedle skutečných obrázků. Přidat obsah je možné na všechny čtyři strany obrázku a podporován je následující obsah:

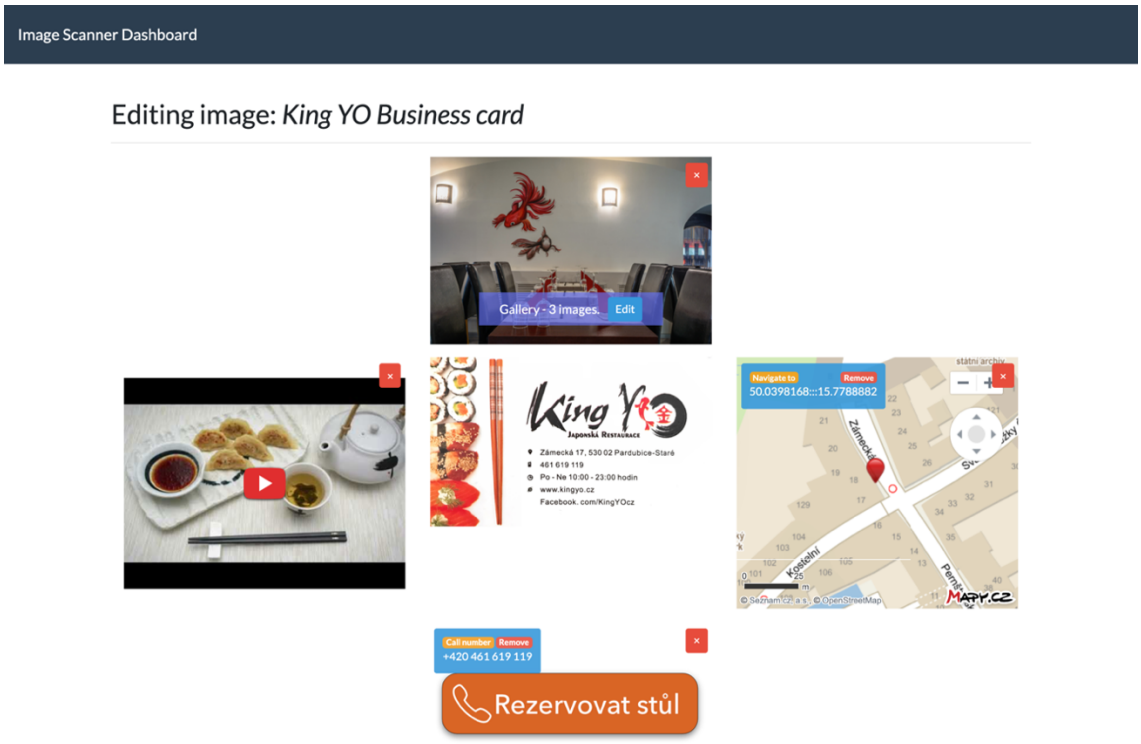
- Obrázek – jakýkoliv obrázek, možnost využít PNG formát s průhledností,
- Galerie obrázků – dotykem lze přepínat mezi jednotlivými obrázky,
- Video – link na YouTube video,
- Mapa se souřadnicí – stačí zadat GPS souřadnice a následně je zobrazen také náhled pomocí Mapy.cz.

Pro výše popsaný virtuální obsah lze podobně snadno definovat ještě akce poté, co uživatel dotykem prstu na displej zařízení tento obsah vybere. Galerie a video mají akce připravené implicitně. V případě galerie slouží dotyk k „listování“ jednotlivými obrázky a u videa potom zase k vypnutí či zapnutí zvuku.

Pro obrázek a mapu lze definovat vlastní akce. Podporované jsou následující:

- Otevření libovolné URL,
- vytočení libovolného čísla,
- poslání emailu na zadanou adresu,
- spuštění navigace na zadanou souřadnici.

Obrázek spojený s vlastními akcemi lze využít mnoha způsoby. Například vytvořit pomyslné tlačítko za pomoci průhlednosti a například u vizitky tak snadno umožnit kontaktovat daný subjekt. U akcí je možné zadat text, který uvidí uživatel mobilní aplikace společně s virtuálním obsahem. Může jít třeba o popisek ve stylu „Tap here to open navigation“.



Obrázek 4 - Stránka pro úpravu obrázku k rozpoznání

Tato stránka obsahuje zdaleka nejvíce funkcionality z celé webové aplikace. Pro intuitivní práci s přidáváním virtuálního obsahu hojně využívá JavaScript a knihovnu jQuery, pomocí které jsou veškeré akce asynchronní, takže není třeba obnovovat stránku po každé drobné úpravě.

Právě zde se také zvlášť hodil grid systém CSS frameworku Bootstrap pro zobrazení obrázku k rozpoznání a po jeho stranách buď přidaného virtuálního obsahu nebo prázdných pozic. Tato část stránky se ve skutečnosti skládá z mřížky 3x3, pouze u prostředního sloupce jsou ale využity všechny tři řádky. Díky frameworku Bootstrap je rovněž tato stránka a další do vysoké míry responzivní. Design byl ovšem navrhován primárně pro počítače nebo tablety z důvodu, že přidávání virtuálního obsahu by nebylo na chytrém telefonu komfortní i z jiných důvodů, především nahrávání obrázků či jejich případné úpravy před nahráním.

11.4 Přidávání virtuálního obsahu

Přidávání obsahu k zobrazení v augmentované realitě je realizováno pomocí „+“ tlačítka zobrazeného uprostřed prázdných pozic po stranách obrázku k rozpoznání. Po kliku je uživateli

nabídnuto dropdown menu s výběrem výše popsaného typu virtuálního obsahu. Po výběru je zobrazen specializovaný modální dialog s formulářem pro konkrétní virtuální obsah. U obrázku je to hlavně samotný obrázek, pro který se okamžitě po vybrání zobrazí náhled, aby si uživatel mohl zkontrolovat, že skutečně vybral chtěný obrázek. Po vyplnění požadovaných polí je virtuální obsah uložen tlačítkem „Save“. Díky asynchronní nátuře je skrze jQuery přidaný obsah okamžitě zobrazen a není třeba načítat znovu celou stránku.

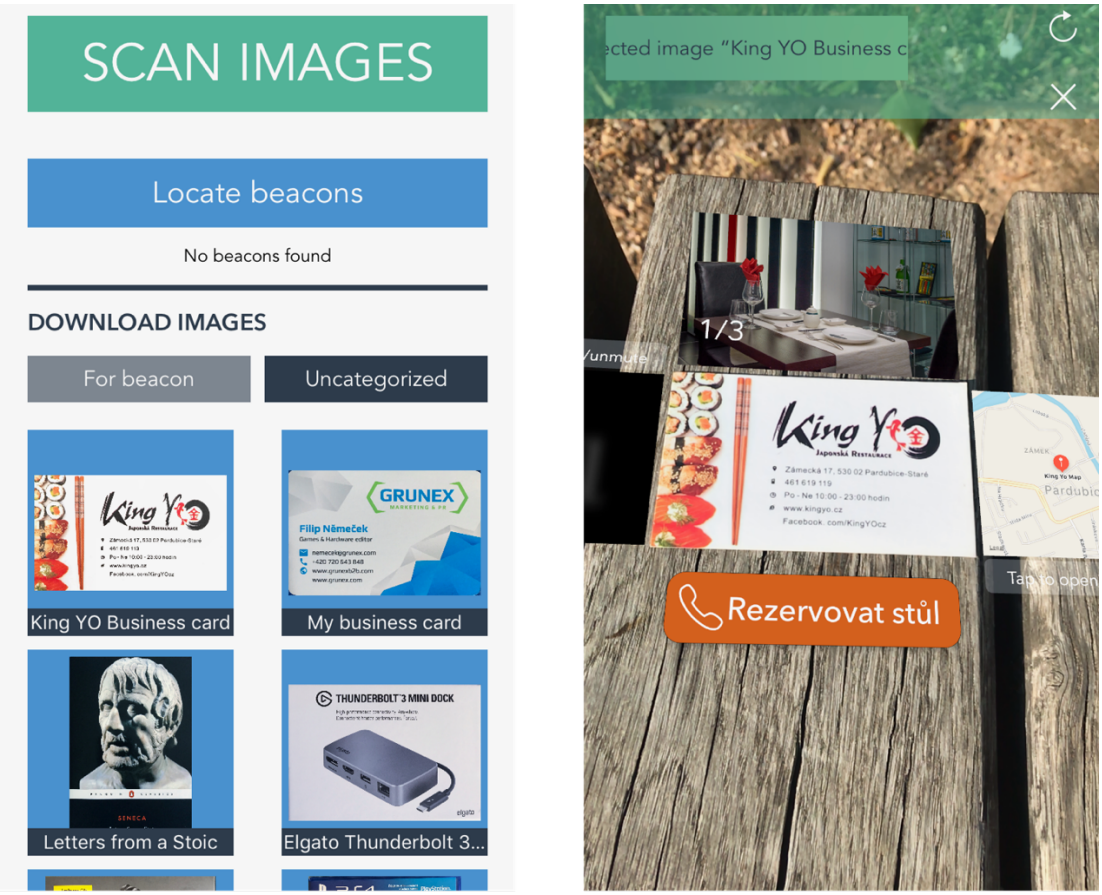
Přidávání akcí k obsahu funguje podobně jako přidání samotného obsahu. U podporovaných typů (obrázek a mapa) je v levém horním rohu zobrazeno tlačítko „+ action“, po jehož výběru se zobrazí modální dialog s možností akci přidat.

Danou akci je možné intuitivně nastavit přesně podle potřeb. Nutné je zadat její jméno pro snadnější orientaci, dále vybrat její typ (ve výchozím stavu je zde nastaveno otevření URL) a její hodnotu, to může být URL k otevření, číslo pro vytočení nebo GPS souřadnice ve speciálním formátu, který je uživateli zobrazen přímo v nápovědě u tohoto pole. Volitelně je možné přidat parametry, popis akce, a ještě text k zobrazení v augmentované realitě společně se samotným obsahem.

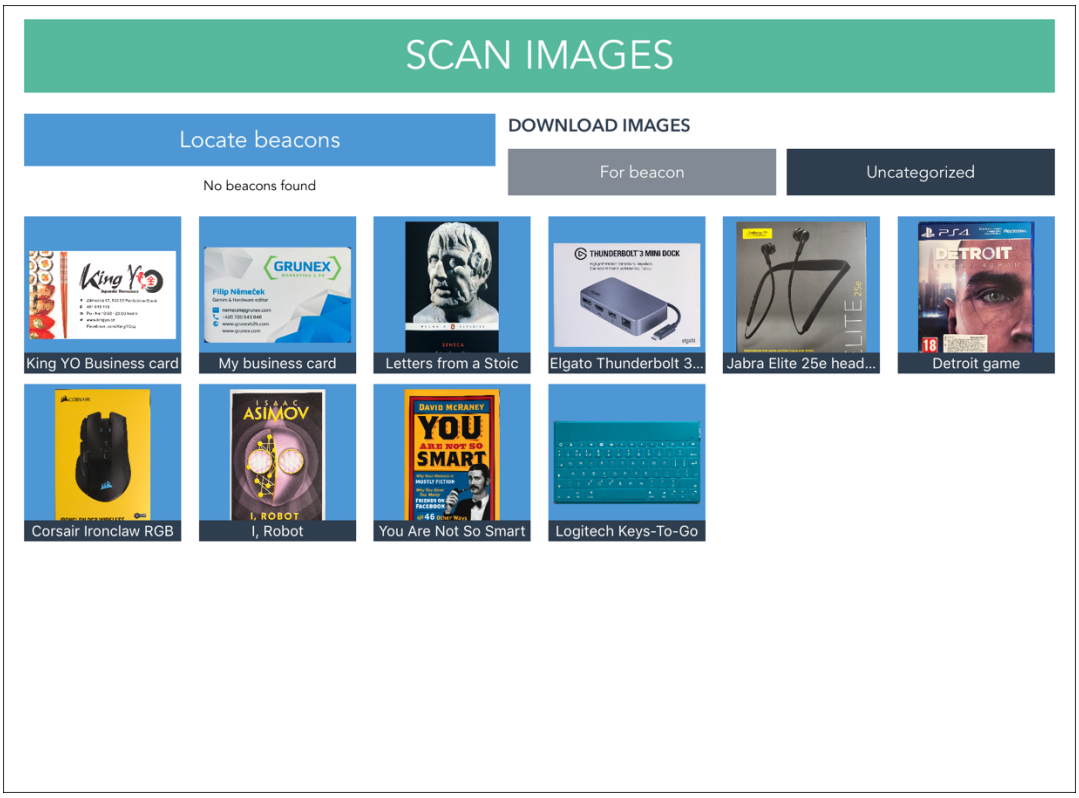
11.5 Stránka pro úpravu galerie

Stránka je úzce spojena s výše popsanou stránkou pro přidávání obsahu k zobrazení v augmentované realitě. Pokud uživatel vybere, že chce přidat galerii obrázků, tak při vytváření nahraje první obrázek a další může přidat právě na této stránce. Stránka obsahuje jednoduchý formulář pro přidání nových obrázků, opět je okamžitě zobrazen náhled, a dále výpis všech přidaných obrázků. Ty lze smazat nebo si je prohlédnout v nové záložce prohlížeče v plném rozlišení.

12 UKÁZKY HOTOVÉ APLIKACE



Obrázek 5 - Ukázka hotové aplikace



Obrázek 6 - Ukázka iPad verze – hlavní obrazovka

Obrázek 7 - Ukázka iPad verze – rozpoznání obrázku

ZÁVĚR

Cílem teoretické části práce bylo uvedení do problematiky augmentované reality, popis možností umisťování digitálních objektů do prostředí skutečného světa a přehled dostupných knihoven pro vývoj aplikací s augmentovanou realitou. Všechny cíle byly splněny. První kapitola obecně představuje augmentovanou realitu a druhá uvádí možnosti využití dnes. Následující třetí kapitola popisuje dostupné možnosti, jak digitální objekty do skutečného světa umístit, včetně výhod a nevýhod.

Čtvrtá kapitola se věnuje vybraným knihovnám pro vývoj aplikací s augmentovanou realitou. Zvláště podrobně popisuje knihovnu Apple ARKit, která je využita v praktické části práce, dále také konkurenční ARCore od Google. Obě tyto knihovny jsou zdarma, těší se velké podpoře tvůrců a nabízí srovnatelné možnosti pro vývoj. Při výběru tak záleží primárně na tom, na jakou platformu je zamýšlená aplikace cílena. Na závěr této kapitoly bylo ještě provedeno srovnání popsaných knihoven podle několika parametrů. Tímto byl splněn poslední cíl teoretické části.

Cílem praktické části a hlavním cílem práce samotné bylo vytvoření mobilní aplikace využívající identifikační a lokační technologie pro umisťování digitálních objektů do skutečného světa. Tento cíl byl splněn vytvořením aplikace pro systém iOS, která je navíc doplněna webovou administrací pro snadnou správu obsahu. K umisťování digitálních objektů využívá rozpoznávání obrázků, čímž byl splněn požadavek na identifikační technologii. Dále využívá technologii iBeacon pro volitelnou lokaci uživatele, čímž byl splněn požadavek na využití lokační technologie. Výsledek je aplikace umožňující rozpoznání obrázků, zobrazení dodatečného obsahu a možnou interakci s tímto digitálním obsahem, jako je třeba otevření webové stránky nebo spuštění navigace.

Výsledný systém obsahuje webovou část, která nabízí kompletní a intuitivní správu uživatelského obsahu. Uživatelé mohou přidat iBeacon zařízení a následně k nim přiřadit konkrétní obrázky (či jakékoliv jiné 2D plochy) k rozpoznání. V dalším kroku může uživatel přidat virtuální obsah, který bude u těchto obrázků zobrazen, a zvolit akce po vybrání tohoto obsahu. Případně je možné připravit HTML stránku v intuitivním editoru a její obsah následně zobrazovat uživateli po rozpoznání konkrétního obrázku.

Mobilní aplikace komunikuje s webovou službou a umožňuje uživateli detekovat nejbližší iBeacon zařízení a pro něj následně stáhnout obrázky k rozpoznání či využít nezařazené obrázky pro rychlejší postup. Následně aplikace dovolí stažené obrázky rozpoznat a zobrazit k nim definovaný virtuální obsah z webové části projektu.

Protože praktická část slouží jako prototyp kompletního systému pro aplikaci, využívající rozpoznávání obrázků, nabízí spoustu možností k rozšíření a vylepšení. Například by mohla být rozšířena o možnost rozpoznat 3D objekty na základně zdrojových dat a zobrazit virtuální obsah k nim. Systém by mohl rovněž využívat jiné metody pro určení, jaký obsah k rozpoznání bude aktivní. Například QR kódy pro jednotlivé místnosti s objekty k rozpoznání nebo GPS souřadnice. S drobnými úpravami by systém mohl sloužit k překládání důležitých informačních textů, kdy by byl virtuální obsah zobrazen přímo přes ten skutečný. Aplikace by mohla takto poskytnout překlad klidně pro všechny světové jazyky a rovnou využít kontextu mobilního zařízení ke zvolení správného jazyka.

POUŽITÁ LITERATURA

[1] CLAYTON, Craig. *Learn iOS 11 Programming with Swift 4*: Learn the fundamentals of iOS app development with Swift 4 and Xcode 9. 2. Birmingham, United Kingdom: Packt Publishing Limited, 2018. ISBN 9781788390750.

[2] HOFFMAN, Jon. *Mastering Swift 4*. 4. Birmingham, United Kingdom: Packt Publishing Limited, 2017. ISBN 9781788477802.

[3] RAVINDRAN, Arun. *Django Design Patterns and Best Practices:* Industry-standard web development techniques and solutions using Python. 2. Birmingham, United Kingdom: Packt Publishing Limited, 2018. ISBN 9781788831345.

[4] GREENFELD, Daniel Roy a Audrey Roy GREENFELD. *Two Scoops of Django 1.11: Best Practices for the Django Web Framework*. 1. Los Angeles: Two Scoops Press, 2017. ISBN 978-0692915721.

[5] CHMIELEWSKI, Dawn C. Apple's Tim Cook Says Augmented Reality Will "Change Everything". *Deadline* [online]. 2017 [cit. 2019-04-26]. Dostupné z: https://deadline.com/2017/11/apple-tim-cook-augmented-reality-fourth-quarter-2017-earnings-1202200710/.

[6] Microsoft HoloLens: Technologie hybridní reality pro firmy. *Microsoft* [online]. [cit. 2019-04-26]. Dostupné z: https://www.microsoft.com/cs-cz/hololens

[7] Google unveils Project Glass augmented reality eyewear. *BBC News* [online]. 2012 [cit. 2019-04-27]. Dostupné z: https://www.bbc.com/news/technology-17618495

[8] *Glass* [online]. [cit. 2019-04-27]. Dostupné z: https://www.x.company/glass/

[9] Používání aplikace Měření na iPhonu nebo iPadu. *Podpora Apple* [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://support.apple.com/cs-cz/HT208924

[10] WELCOME TO AR CITY: BETA OF AUGMENTED REALITY MAPS AND NAVIGATION. *BlippAR*[online]. 2017 [cit. 2019-04-27]. Dostupné z: https://www.blippar.com/blog/2017/11/06/welcome-ar-city-future-maps-and-navigation

[11] *Magic Leap* [online]. [cit. 2019-04-27]. Dostupné z: https://www.magicleap.com/

[12] PAGE, Holden. A Timeline Of Investor Interest In AR Startup Magic Leap, Which Has Raised $2.3B. *Crunchbase News* [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://news.crunchbase.com/news/a-timeline-of-investor-interest-in-ar-startup-magic-leap-which-has-raised-2-3b/

[13] CHARLTON, Alistair. Magic Leap: What is going on at the secretive $6B AR startup?. *GearBrain*[online]. 2017 [cit. 2019-04-27]. Dostupné z: https://www.gearbrain.com/magic-leap-augmented-reality-startup-2500233423.html

[14] STEIN, Scott. I finally tried Magic Leap, and I have mixed feelings. *CNET* [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://www.cnet.com/news/i-finally-tried-magic-leap-and-i-have-mixed-feelings/

[15] ROBERTSON, Adi. I TRIED MAGIC LEAP AND SAW A FLAWED GLIMPSE OF MIXED REALITY'S AMAZING POTENTIAL. *The Verge* [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://www.theverge.com/2018/8/8/17662040/magic-leap-one-creator-edition-preview-mixed-reality-glasses-launch

[16] IKEA Place. *App Store* [online]. 2017 [cit. 2019-04-27]. Dostupné z: https://itunes.apple.com/us/app/ikea-place/id1279244498

[17] PeakFinder App. *PeakFinder* [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://www.peakfinder.org/mobile/

[18] LESTOC, Costea. Pokémon Go Available for iOS and Android – Pokemon Catching Events. *Neurogadget* [online]. 2016 [cit. 2019-04-27]. Dostupné z: https://neurogadget.net/2016/06/16/pokemon-go-available-for-ios-and-android-pokemon-catching-events/33225

[19] SMITH, Craig. 115 Incredible Pokemon Go Statistics and Facts (2019): By the Numbers. *DMR* [online]. 2019 [cit. 2019-04-27]. Dostupné z: https://expandedramblings.com/index.php/pokemon-go-statistics/

[20] 9,7palcový iPad: Technické specifikace. *Apple* [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://www.apple.com/cz/ipad-9.7/specs/

[21] OWEN, Malcolm. Apple's A12 Bionic comes close to desktop CPU performance in benchmarks. *Apple Insider* [online]. 2018 [cit. 2019-04-27]. Dostupné z:

https://appleinsider.com/articles/18/10/05/apples-a12-bionic-comes-close-to-desktop-cpu-performance-in-benchmarks

[22] LUTZ, Zachary. Nokia reveals new City Lens augmented reality app for Windows Phone 8 lineup. *Engadget* [online]. 2012 [cit. 2019-04-27]. Dostupné z: https://www.engadget.com/2012/09/11/nokia-reveals-new-city-lens-for-windows-phone-8/

[23] Introducing ARKit: Augmented Reality for iOS. *Apple Developer* [online]. 2017 [cit. 2019-04-27]. Dostupné z: https://developer.apple.com/videos/play/wwdc2017/602/

[24] Number of Apple iPhone devices in use in the U.S., China and the rest of the world in 2017* (in millions). *Statista* [online]. 2017 [cit. 2019-04-27]. Dostupné z: https://www.statista.com/statistics/755625/iphones-in-use-in-us-china-and-rest-of-the-world/

[25] HASLAM, Oliver. IOS 11 ARKit Compatibility For Apps: Check If Your Device Is Compatible With Apple's AR Platform. *Redmond Pie* [online]. 2017 [cit. 2019-04-27]. Dostupné z: https://www.redmondpie.com/ios-11-arkit-compatibility-check-if-your-device-is-compatible-with-apples-new-ar-platform/

[26] IOS 12. *Apple* [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://www.apple.com/cz/ios/ios-12/

[27] Unit sales of the Apple iPhone worldwide from 2007 to 2018 (in millions). *Statista* [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://www.statista.com/statistics/276306/global-apple-iphone-sales-since-fiscal-year-2007/

[28] GARUN, Natt. Apple will no longer reveal how many iPhones, iPads, and Macs it sells. *The Verge* [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://www.theverge.com/2018/11/1/18053782/apple-earnings-q4-2018-stop-ipad-iphone-mac-unit-sales-disclosure

[29] DE SIMONE, Sergio. ARKit Sets the Foundations for Augmented Reality on Apple's Platform. *InfoQ* [online]. 2017 [cit. 2019-04-27]. Dostupné z: https://www.infoq.com/news/2017/06/ios-augmented-reality-arkit

[30] DE SIMONE, Sergio. ARKit 1.5 Now Supports Vertical Surface Detection and 2D Image Recognition. *InfoQ* [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://www.infoq.com/news/2018/01/arkit-1.5-available

[31] SAFFER, Stephanie. Apple unveils ARKit 2. *Apple Newsroom* [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://www.apple.com/newsroom/2018/06/apple-unveils-arkit-2/

[32] DEAHL, Dani. Apple's ARKit 2.0 supports multiplayer and introduces a new file format. *The Verge* [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://www.theverge.com/2018/6/4/17418614/arkit-update-multiplayer-announced-apple-wwdc-2018

[33] Scanning and Detecting 3D Objects. *Apple Developer Documentation* [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://developer.apple.com/documentation/arkit/scanning_and_detecting_3d_objects

[34] Introduction to USD. *Pixar* [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://graphics.pixar.com/usd/docs/index.html

[35] AR Quick Look Gallery. *Apple Developer* [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://developer.apple.com/arkit/gallery/

[36] NISTOR, Codrut. Google announces Project Tango experimental Android smartphone. *NotebookCheck* [online]. 2014 [cit. 2019-04-28]. Dostupné z: https://www.notebookcheck.net/Google-announces-Project-Tango-experimental-Android-smartphone.111856.0.html

[37] Asus ZenFone AR is now available for purchase, exclusively from Verizon. *Android Authority*[online]. 2017 [cit. 2019-04-28]. Dostupné z: https://www.androidauthority.com/asus-zenfone-ar-verizon-exclusive-791425/

[38] KASTRENAKES, Jacob. Google's Project Tango is shutting down because ARCore is already here. *The Verge* [online]. 2017 [cit. 2019-04-28]. Dostupné z: https://www.theverge.com/2017/12/15/16782556/project-tango-google-shutting-down-arcore-augmented-reality

[39] AMADEO, Ron. Google's ARCore brings augmented reality to millions of Android devices. *Ars Technica* [online]. 2017 [cit. 2019-04-28]. Dostupné z:

https://arstechnica.com/gadgets/2017/08/googles-arcore-brings-augmented-reality-to-millions-of-android-devices/

[40] SINGH, Sudhanshu. Google ARCore SDK released with support for over 100 million devices. *TechRadar* [online]. 2018 [cit. 2019-04-28]. Dostupné z: https://www.techradar.com/news/google-arcore-sdk-released-with-support-for-over-100-million-devices

[41] Welcome to the ARCore SDK for iOS. *Google Developers* [online]. 2018 [cit. 2019-04-28]. Dostupné z: https://developers.google.com/ar/develop/ios/overview

[42] FEDEWA, Joe. ARCore 1.2 adds collaborative augmented reality experiences and vertical plane detection. *XDA-Developers* [online]. 2018 [cit. 2019-04-28]. Dostupné z: https://www.xda-developers.com/arcore-1-2-cloud-anchors-vertical-plane-detection/

[43] Experience augmented reality together with new updates to ARCore. *The Keyword* [online]. 2018 [cit. 2019-04-28]. Dostupné z: https://www.blog.google/products/arcore/experience-augmented-reality-together-new-updates-arcore/

[44] Just a Line: Draw Anywhere, with AR. *Google Play* [online]. 2018 [cit. 2019-04-28]. Dostupné z: https://play.google.com/store/apps/details?id=com.arexperiments.justaline

[45] SHAH, Ashish. Creating More Realistic AR experiences with updates to ARCore & Sceneform. *Google Developers Blog* [online]. 2018 [cit. 2019-04-28]. Dostupné z: https://developers.googleblog.com/2018/12/arcore-16-brings-improved-lighting.html

[46] PARKER, Evan Hardesty. New UI tools and a richer creative canvas come to ARCore. *Google Developers Blog* [online]. 2019 [cit. 2019-04-28]. Dostupné z: https://developers.googleblog.com/2019/02/new-ui-tools-and-richer-creative-canvas.html

[47] Performance is everything!. *Wikitude* [online]. 2016 [cit. 2019-04-28]. Dostupné z: https://www.wikitude.com/blog-performance-is-everything/

[48] Wikitude Documentation. *Wikitude* [online]. [cit. 2019-04-28]. Dostupné z: https://www.wikitude.com/documentation/

[49] Getting started: Introduction to the Wikitude SDK. *Wikitude* [online]. [cit. 2019-04-28]. Dostupné z: https://www.wikitude.com/external/doc/documentation/latest/iosnative/

[50] Success Stories. *Wikitude* [online]. [cit. 2019-04-28]. Dostupné z: https://www.wikitude.com/showcase/

[51] Wikitude Store. *Wikitude* [online]. [cit. 2019-04-28]. Dostupné z: https://www.wikitude.com/store/

[52] What is EasyAR SDK. *Easy AR SDK* [online]. [cit. 2019-04-28]. Dostupné z: https://www.easyar.com/view/sdk.html

[53] NĚMEČEK, Filip. *Návrh a realizace mobilní aplikace pro MHD využívající geolokaci v operačním systému Windows Phone*. Pardubice, 2017. Bakalářská práce. Univerzita Pardubice. Vedoucí práce Jiří Kysela.

[54] Human Interface Guidelines: Augmented Reality. *Apple* [online]. [cit. 2019-04-28]. Dostupné z: https://developer.apple.com/design/human-interface-guidelines/ios/system-capabilities/augmented-reality/

[55] IBeacon. *Apple Developer* [online]. [cit. 2019-04-28]. Dostupné z: https://developer.apple.com/ibeacon/

[56] ROSSIGNOL, Joe. IOS 12 Adoption Hits 70%, Compared to 59% for iOS 11 Last Year. *MacRumors* [online]. 2018 [cit. 2019-04-28]. Dostupné z: https://www.macrumors.com/2018/12/04/ios-12-reaches-70-percent-adoption/

[57] AppCode: Smart IDE for iOS/macOS development. *JetBrains* [online]. [cit. 2019-04-28]. Dostupné z: https://www.jetbrains.com/objc/

[58] WEGNER, Wade. Create an iBeacon Transmitter with the Raspberry Pi. *WadeWegner*[online]. 2014 [cit. 2019-04-28]. Dostupné z: http://www.wadewegner.com/2014/05/create-an-ibeacon-transmitter-with-the-raspberry-pi/

[59] Beacon Scan. *Mac App Store* [online]. 2018 [cit. 2019-04-28]. Dostupné z: https://itunes.apple.com/us/app/beacon-scan/id995724474?mt=12

[60] ARConfiguration. *Apple Developer Documentation* [online]. [cit. 2019-04-28]. Dostupné z: https://developer.apple.com/documentation/arkit/arconfiguration

[61] SCNAction. *Apple Developer Documentation* [online]. [cit. 2019-04-28]. Dostupné z: https://developer.apple.com/documentation/scenekit/scnaction

[62] UIWebView. *Apple Developer Documentation* [online]. [cit. 2019-04-28]. Dostupné z: https://developer.apple.com/documentation/uikit/uiwebview