



Univerza v Mariboru

Fakulteta za elektrotehniko,
računalništvo in informatiko

Aleš Spital

Z OBOGATENO RESNIČNOSTJO PODPRT TURISTIČNI VODNIK

Diplomsko delo

Maribor, september 2020



Univerza v Mariboru

Fakulteta za elektrotehniko,
računalništvo in informatiko

Aleš Spital

Z OBOGATENO RESNIČNOSTJO PODPRT TURISTIČNI VODNIK

Diplomsko delo

Maribor, september 2020

Z OBOGATENO RESNIČNOSTJO PODPRT TURISTIČNI VODNIK

Diplomsko delo

Študent:	Aleš Spital
Študijski program:	Visokošolski strokovni študijski program Računalništvo in informacijske tehnologije
Mentor:	doc. dr. Simon Kolmanič
Somentor:	asist. dr. Štefan Kohek

ZAHVALA

Zahvaljujem se mentorju doc. dr.
Simonu Kolmaniču in somentorju asist.
dr. Štefanu Koheku za vso pomoč in
napotke pri izdelavi diplomske naloge.
Zahvaljujem se Neži Videmšek za pomoč
pri lektoriranju in tudi družini za vso
podporo v času študija.

Z obogateno resničnostjo podprt turistični vodnik

Ključne besede: obogatena resničnost, turizem, kulturna dediščina, mobilna aplikacija

UDK: 004.777:004.946(043.2)

Povzetek

Ob naraščajočem širjenju obogatene in navidezne resničnosti na trgu se je dobro vprašati, kako lahko to tehnologijo izkoristimo. Cilj te diplomske naloge je raziskati in razviti mobilno aplikacijo, ki uporablja tehnologijo obogatene resničnosti predvsem na področju turizma. Uporabnikom te aplikacije želimo ponuditi dodaten vir informacij kot način vpogleda v preteklost s prikazovanjem 3D modelov na lokaciji, kjer se nahajajo objekti kulturne dediščine ali znamenitosti. Diplomska naloga obsega pregled podobnih aplikacij, postavitev strežnika, iz katerega mobilna aplikacija črpa podatke, ustvarjanje 3D modela ter postopek izdelave mobilne aplikacije v programu Unity.

Augmented reality based tourist guide

Keywords: augmented reality, tourism, cultural heritage, mobile application

UDC: 004.777:004.946(043.2)

Abstract

With the increasing expansion of augmented and virtual reality on the market, it is good to ask how can we take advantage of this technology. The aim of this diploma thesis is to research and develop a mobile application that uses augmented reality technology primarily in the field of tourism. We want to offer application users an additional source of information as a way of insight into the past by displaying 3D models on the location of cultural heritage sites or landmarks. The diploma thesis includes research of similar applications, setting up a server from which the mobile application draws data, creating a 3D model and the process of creating a mobile application in Unity.

KAZALO VSEBINE

1	UVOD	1
2	KRATKA PRESTAVITEV PODROČJA	3
2.1	Obogatena resničnost	3
2.2	Sorodne aplikacije	4
2.2.1	World Around Me	5
2.2.2	Aplikacija Augment - 3D Augmented Reality	6
2.2.3	Aplikacija Senditur	7
2.2.4	Aplikacija Smartify	8
3	PREDSTAVITEV PROGRAMSKE OPREME IN UPORABLJENIH ORODIJ	9
3.1	Unity	9
3.1.1	ARCore / AR Foundation	10
3.1.2	Runtime OBJ importer	10
3.1.3	AR + GPS Location	10
3.2	Visual Studio	11
3.3	Sublime Text 3	11
3.4	XAMPP	12
4	IZDELAVA SPLETNE APLIKACIJE	13
4.1	Načrtovanje spletne aplikacije	13
4.2	Izdelava podatkovne baze	13
4.3	Izdelava spletnega vmesnika	17
4.3.1	Povezava do podatkovne baze	17
4.3.2	Izpis modelov iz podatkovne baze	17
4.3.3	Vnos novega modela v podatkovno bazo	20
4.3.4	Spreminjanje podatkov modela v podatkovni bazi	20
4.3.5	Izbris modela iz podatkovne baze	21
4.4	Izdelava spletne strani	21
5	IZDELAVA 3D MODELA OBJEKTA	23

5.1	Zasnova 3D modela objekta kulturne dediščine.....	23
5.2	Ustvarjanje 3D modela objekta kulturne dediščine	25
5.3	Priprava 3D modela za prikaz v mobilni aplikaciji.....	28
6	IZDELAVA MOBILNE APLIKACIJE	30
6.1	Načrtovanje mobilne aplikacije	30
6.2	Postopek izdelave mobilne aplikacije	30
6.2.1	Inicializacija projekta.....	30
6.2.2	Branje podatkov iz spletnega vmesnika	32
6.2.3	Nalaganje 3D modelov v aplikacijo	34
6.2.4	Osveževanje podatkov	36
6.2.5	Prikaz podatkov o objektu	37
6.2.6	Prikaz informacij o bližnjih objektih.....	38
6.2.7	Zajemanje slike zaslona	39
6.2.8	Dostop do GPS senzorja in internetne povezave	40
6.3	Objava mobilne aplikacije	41
7	REZULTATI	42
8	ZAKLJUČEK	48
	VIRI IN LITERATURA	49

KAZALO SLIK

Slika 2.1: Ocena naraščanja trga obogatene resničnosti v obdobju od 2019 do 2024 [3]	4
Slika 2.2: Glavni zaslon aplikacije World Around Me [4]	5
Slika 2.3: Delovanje aplikacije Augment - 3D Augmented Reality [5]	6
Slika 2.4: Glavni zaslon aplikacije Senditur [6]	7
Slika 2.5: Prikaz delovanja aplikacije Smartify [7]	8
Slika 3.1: Logotip programa Unity [8]	9
Slika 3.2: Logotip programa Visual Studio [14]	11
Slika 3.3: Logotip programa Sublime Text 3 [15]	12
Slika 4.1: Vklop lokalnega strežnika v programu XAMPP	14
Slika 4.2: Ustvarjanje uporabnika za podatkovno bazo	15
Slika 4.3: Ustvarjanje podatkovne baze	16
Slika 4.4: Povezava do podatkovne baze	17
Slika 4.5: Izpis podatkov modelov do razdalje tisoč metrov ali vseh modelov iz podatkovne baze.....	18
Slika 4.6: Izračun najkrajše razdalje med dvema točkama na površini krogle	19
Slika 4.7: Izpis podatkov enega modela iz podatkovne baze.....	19
Slika 4.8: Vstavljanje modela v podatkovno bazo	20
Slika 4.9: Spreminjanje podatkov modela v podatkovni bazi	21
Slika 4.10: Izbris modela iz podatkovne baze	21
Slika 5.1: Cerkev svetega Mihaela v Družmirju pred in po zrušitvi [18]	23
Slika 5.2: Razglednica cerkve svetega Mihaela v Družmirju [19].....	24
Slika 5.3: Načrt za izdelavo 3D modela po sliki 5.1	24
Slika 5.4: Manipuliranje objekta in uporaba simetrale.....	25
Slika 5.5: Izrez okna v modelu z uporabo logičnih funkcij	26
Slika 5.6: Upravljevec materialov.....	27
Slika 5.7: Pečenje objekta	28
Slika 5.8: Družmirje nekoč in danes [20].....	29
Slika 6.1: Inicializacija glavnih nastavitev aplikacije	31
Slika 6.2: Inicializacija scene	32

Slika 6.3: Pridobivanje in serializacija podatkov iz podatkovne baze	33
Slika 6.4: Razred, ustvarjen po shemi podatkov v podatkovni bazi	34
Slika 6.5: Ustvarjanje objekta in teksturiranje iz datotek na strežniku	35
Slika 6.6: Postavitev objekta v geografsko okolje	36
Slika 6.7: Izsek kode za osveževanje podatkov	37
Slika 6.8: Izsek kode za preverjanje dotika uporabnika	37
Slika 6.9: Izsek kode za prikaz informacij o objektu	38
Slika 6.10: Izsek kode za prikaz vseh bližnjih objektov	39
Slika 6.11: Izsek kode za zajem slike zaslona	40
Slika 6.12: Preverjanje povezljivosti z internetom in GPS	41
Slika 6.13: Nastavitve imena, ikone in verzije mobilne aplikacije	41
Slika 7.1: Obrazec za vstavljanje modela v podatkovno bazo	42
Slika 7.2: Obrazec za spreminjanje podatkov modela v podatkovni bazi.....	43
Slika 7.3: Končani 3D model cerkve svetega Mihaela v Družmirju	43
Slika 7.4: Opozorilno okno v primeru izgube povezave z internetom.....	44
Slika 7.5: Zaslona ob zagonu mobilne aplikacije.....	45
Slika 7.6: Delovanje mobilne aplikacije z modelom cerkve svetega Mihaela	46
Slika 7.7: Prikaz več informacij o objektu	47
Slika 7.8: Prikaz vseh bližnjih objektov do tisoč metrov in njihovo razdaljo	47

UPORABLJENI SIMBOLI IN KRATICE

AR – Augmented reality (obogatena resničnost)

VR – Virtual reality (virtualna resničnost)

GPS – Global Positioning System (globalni sistem pozicioniranja)

JSON – JavaScript Object Notation (objektna notacija JavaScript)

API – Application Programming Interface (aplikacijski programski vmesnik)

HTML - Hypertext Markup Language (jezik za označevanje nadbesedila)

1 UVOD

Že nekaj časa se na trgu pojavljajo različne aplikacije s tehnologijami navidezne ali obogatene resničnosti. A ta tehnologija bo v prihodnosti postala še precej bolj dominantna, saj se proizvajalci raznih elektronskih naprav zelo trudijo, da bi jo čim bolje izkoristili.

Obogatena resničnost omogoča dodajanje digitalnih informacij na osnovi realnega sveta. V nasprotju od navidezne resničnosti nam ta ne prikazuje popolnoma namišljenega ali umetnega okolja. Prikazuje nam sliko realnega sveta preko kamer mobilnih naprav in nanjo doda sloj v obliki zvoka, grafičnih objektov, video posnetkov ali podobno. [1]

Tehnologija na današnjem trgu še vedno ostaja precej neizkoriščena, zato je to dobra priložnost za razvoj novih aplikacij. Dober izkoristek na tem področju ima turizem, saj obiskovalcem določenega kraja omogoča vpogled v stvari, ki trenutno niso dostopne, so bile s časom obnovljene, posodobljene ali porušene. Zato je bil namen te diplomske naloge ustvariti mobilno aplikacijo s tehnologijo obogatene resničnosti, na platformi Android, ki bi uporabnikom na kraju določene kulturne znamenitosti omogočila prikaz modela objekta v stanju pred spremembo. Ustvarili smo tudi podatkovno bazo in spletno aplikacijo za lažje shranjevanje teh modelov z naslovom, opisom in lokacijo v zemljepisnih koordinatah. Tako mobilna aplikacija črpa podatke iz podatkovne baze, pridobiva objekte, ki so v določeni razdalji od naprave in jih tudi prikaže. Uporabnik pa s klikom na modele v aplikaciji lahko pridobi tudi več informacij o sami kulturni dediščini. Aplikacija omogoča tudi zajem slike zaslona, prikaz seznama vseh objektov v okolici, njihovo razdaljo do naprave, samodejno osveževanje objektov na vsakih 800 metrov ter osveževanje ob kliku na gumb.

V naslednjem poglavju smo na kratko predstavili področje naše diplomske naloge, kjer si pogledamo nekaj načinov delovanja in uporabe obogatene resničnosti ter se podrobneje seznanimo in primerjamo našo aplikacijo z nekaj sorodnimi iz trgovine

Google Play. Opisali smo tudi uporabljeno programsko opremo in orodja. Poudarek v tem poglavju je predvsem na programu Unity, v katerem je bila naša aplikacija ustvarjena. Seznanimo se tudi z razvojnim okoljem Visual Studio, urejevalnikom kode Sublime Text 3 in strežniško opremo XAMPP, s katero smo ustvarili podatkovno bazo. Sledi poglavje s postopkom izdelave spletne aplikacije, v katerem ustvarimo spletni obrazec za nalaganje 3D modelov v podatkovno bazo in vmesnike za komunikacijo med mobilno aplikacijo ter podatkovno bazo. Opišemo tudi postopek izdelave 3D modela našega izbranega objekta ter pripravo tega za prikaz v mobilni aplikaciji. Nato sledi poglavje, kjer je podrobneje opisan proces ustvarjanja mobilne aplikacije in postopek objave le-te na mobilno napravo. V predzadnjem poglavju je prikazan končni videz in delovanje naše rešitve. Zatem sledi ovrednotenje rezultatov in sklep.

2 KRATKA PRESTAVITEV PODROČJA

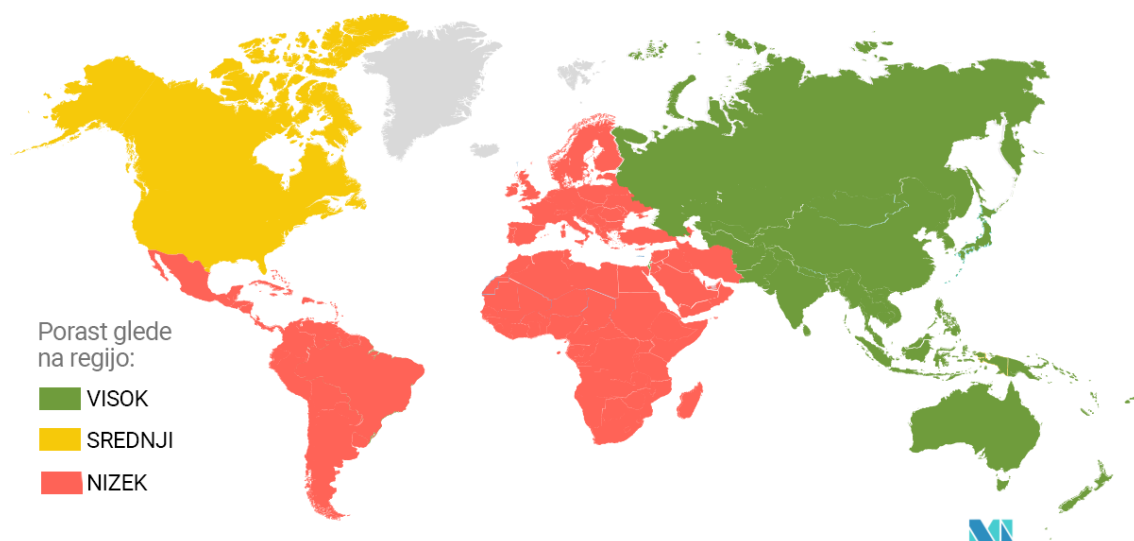
2.1 Obogatena resničnost

Obogatena resničnost je eden izmed večjih trenutnih trendov. Iz prostora, ki ga zazna s pomočjo kamer mobilne naprave, le-ta ustvari digitalno 3D okolje, v katero lahko postavimo digitalno grafiko. To tehnologijo lahko uporabimo na veliko različnih načinov, kot so [22]:

- Napredni navigacijski sistemi, ki uporabljajo obogateno resničnost za prikaz poti skozi pogled na cesto v živo.
- Izdajatelji televizijskih programov prikazujejo črte na igrišču za analizo med nogometnimi igrami.
- Podjetje IKEA ponuja aplikacijo, ki prikazuje, kako bo kos pohištva sestavljen.
- Nevrokirurgi včasih uporabljajo obogateno resničnost za projekcijo 3D možganov za pomoč pri operativnih posegih.
- Na zgodovinskih mestih, kot so Pompeji v Italiji, lahko obogatena resničnost prikazuje starodavne civilizacije nad današnjimi razvalinami.

Sama tehnologija je v razvoju že dlje časa, vendar je bila zaradi slabše zmogljivosti mobilnih naprav ta precej zahtevna za obdelavo v realnem času. Z napredkom ostalih tehnologij, kot je 5G, se bo obogatena resničnost še izboljšala, predvsem na natančnosti, dostopnosti, videzu in hitrosti prikazovanja samih digitalnih informacij. Obogateno resničnost pa je v središče javne pozornosti postavilo podjetje Niantic z mobilno igro Pokémon GO leta 2016, ki je v svojem prvem tednu izdaje zaslužiło dva milijona ameriških dolarjev, vendar ta prikazuje le manjši del zmogljivosti celotnega potenciala obogatene resničnosti. [2]

V letu 2019 je bila ocenjena vrednost tehnologije obogatene resničnosti na trgu 882 milijonov ameriških dolarjev in pričakuje se porast v višini 151,93% v bližnji prihodnosti. Analitiki ocenjujejo visoko razširitev tehnologije predvsem v Aziji in Avstraliji, nekaj manj v severni Ameriki in nižjo drugje po svetu (slika 2.1). [3]



Slika 2.1: Ocena naraščanja trga obogatene resničnosti v obdobju od 2019 do 2024 [3]

Za lažje razumevanje delovanja delimo obogateno resničnost v dve glavni kategoriji [21]:

- Na osnovi oznak (angl. Marker-based AR), kjer se dogodki prožijo ob zaznavi določenih oznak (koda QR, čipi NFC, različne slike). Tukaj ima vodilno vlogo sledenje objektov (angl. tracking), saj moramo ohraniti učinek obogatene resničnosti tudi ob rahlem zamiku oznake ali kamere. Trenutna tehnologija v tej kategoriji je v večini primerov omejena na zaznavo oznak na ravnih površinah.
- Brez oznak (angl. Markerless AR), kjer se za uporabo obogatene resničnosti dogodki prožijo glede na položaj GPS lokacije naprave ali drugih senzorjev. Na ta način deluje tudi naša mobilna aplikacija. V to kategorijo sodijo tudi projekcije, s katerimi projektorji ustvarijo interaktivno uporabniško izkušnjo s svetlobnimi efekti.

2.2 Sorodne aplikacije

V trgovini Google Play lahko že zasledimo aplikacije, ki uporabljajo tehnologijo obogatene resničnosti. Izbrali smo te, ki se osredotočajo na podoben problem v turizmu in jih primerjali z našo rešitvijo tega diplomskega dela.

2.2.1 World Around Me

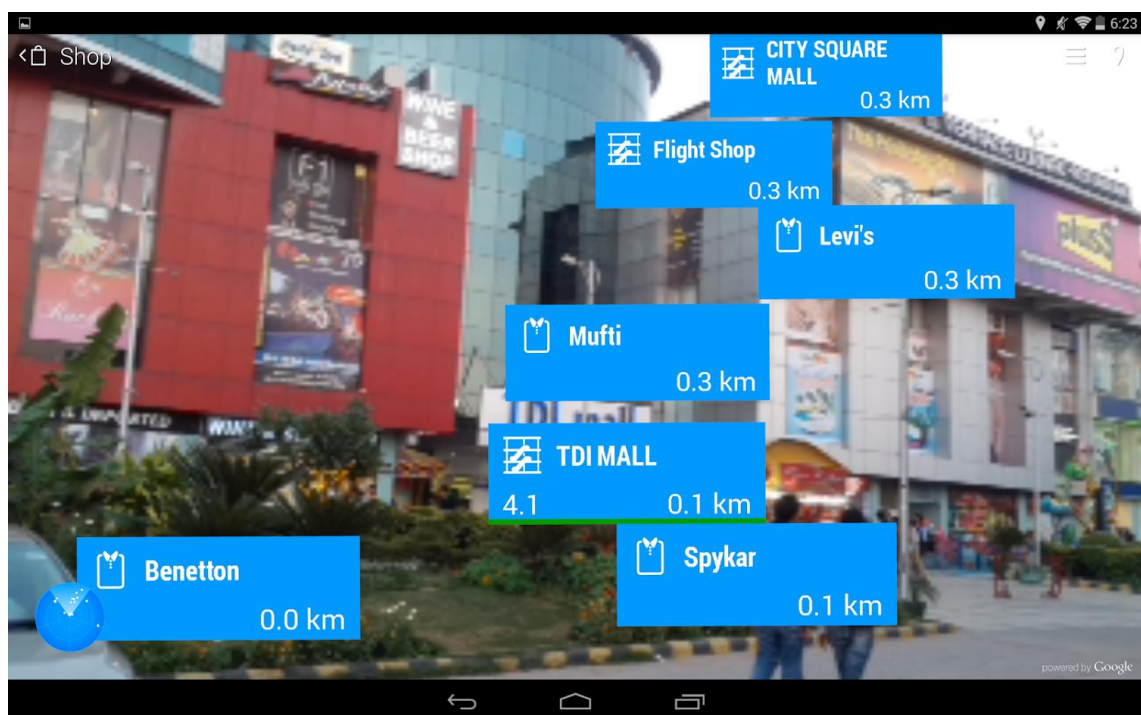
Aplikacija razširja navigacijski zemljevid in omogoča enostaven pogled na točke interesa v bližnji okolici. S pomočjo obogatene resničnosti prikazuje bližnje restavracije, bankomate, muzeje in druge uporabnikom zanimive lokacije, kot prikazuje slika 2.2.

Podobnosti z našo rešitvijo:

- Lokacijsko označuje objekt z obogateno resničnostjo.
- Možnost prikaza več informacij o določenem objektu.

Razlike z našo rešitvijo:

- Ne prikazuje modelov.
- Ne osredotoča se na objekte kulturne dediščine.



Slika 2.2: Glavni zaslon aplikacije World Around Me [4]

2.2.2 Aplikacija Augment - 3D Augmented Reality

Augment - 3D Augmented Reality je splošna aplikacija za obogateno resničnost, ki omogoča prikaz vseh modelov naloženih na spletno stran podjetja Augment. Storitve nalaganja 3D modelov na portal je plačljiva, uporaba same mobilne aplikacije pa je brezplačna in je namenjena predvsem podjetjem, ki želijo obogateno resničnost uporabljati, vendar nimajo pa dovolj virov za izdelavo svoje. Na sliki 2.3 je vidno, kako podjetje ELK uporablja aplikacijo Augment kot pripomoček strankam za vizualizacijo notranje postavitve pohištva v svojih domovih. V tem primeru aplikacija sproži dogodek ob zaznavi specifičnih oznak na arhitektskem načrtu, ki ga strankam ponudi podjetje ELK. Na teh mestih prikazuje 3D modele prehodno naložene na spletno stran podjetja Augment.

Podobnosti z našo rešitvijo:

- Prikaz modelov na določeni lokaciji.

Razlike z našo rešitvijo:

- Ne osredotoča se na objekte kulturne dediščine.
- Sama aplikacija ne ponuja dodatnega vira informacij.
- Zahteva, da uporabniki naredijo modele sami.



Slika 2.3: Delovanje aplikacije Augment - 3D Augmented Reality [5]

2.2.3 Aplikacija Senditur

Ta aplikacija prikazuje sprehajalne poti, cilje in zanimive točke s pomočjo obogatene resničnosti (slika 2.4). Osredotoča se predvsem na območje Španije.

Podobnosti z našo rešitvijo:

- Lokacijsko označuje točko s pomočjo obogatene resničnosti.
- Možnost prikaza več informacij o določeni točki.

Razlike z našo rešitvijo:

- Ne prikazuje modelov.
- Ne osredotoča se na objekte kulturne dediščine.



Slika 2.4: Glavni zaslon aplikacije Senditur [6]

2.2.4 Aplikacija Smartify

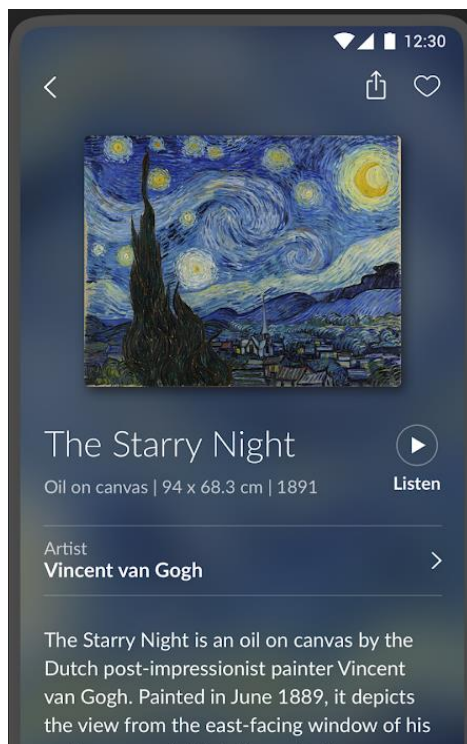
Omogoča podrobnejši vpogled v svet umetnosti in kulture predvsem v galerijah, kjer obiskovalci želijo izvedeti čim več o določenih umetninah, ko vodič ni na voljo. Na sliki 2.5 lahko vidimo prikaz informacij, ki nam jih ponudi aplikacija, ko ta identificira podano umetniško delo.

Podobnosti z našo rešitvijo:

- Možnost ponudbe več informacij o znamenitostih.
- Se delno osredotoča na kulturno dediščino.

Razlike z našo rešitvijo:

- Ne prikazuje modelov.
- Ne označuje lokacije in zahteva, da uporabnik objekt slika.



Slika 2.5: Prikaz delovanja aplikacije Smartify [7]

3 PREDSTAVITEV PROGRAMSKE OPREME IN UPORABLJENIH ORODIJ

3.1 Unity

Je zelo priljubljeno in zmogljivo razvojno okolje video iger za veliko število ciljnih platform. Trenutni logotip je prikazan na sliki 3.1. Grafični pogon Unity je prosto dostopen vsem uporabnikom pod brezplačno licenco, dokler prihodki uporabnika ne presežejo 100 tisoč ameriških dolarjev. Prav tako pa so na voljo tudi plačljive verzije, imenovane Plus, Pro in Enterprise, ki nudijo še dodatne pripomočke ter podporo uporabnikom. Omogoča ustvarjanje 3D in 2D aplikacij ter je orodje, ki ga razvijalci vedno znova posodobljajo in izboljšujejo zmogljivost. Unity je na trgu že od leta 2005, kar pomeni, da ima ogromno število uporabnikov in veliko dobro razvitih knjižic in virov.[9]



Slika 3.1: Logotip programa Unity [8]

Programiranje v pogonu Unity temelji na programskem jeziku C#. Program pa je bil zaradi optimizacije delovanja in same zmogljivosti ustvarjen s programskim jezikom C++. Izvorna koda je dostopna vsem uporabnikom plačljive Pro verzije, saj s tem ponujajo še več dodatnih možnosti za razvijalce.

V zadnjem času se tudi razvijalci programa Unity precej posvečajo navidezni in obogateni resničnosti. To možnost izkorišča veliko priznanih podjetjih, kot sta NASA in Ubisoft. Prav tako pa je s tem programom Niantic ustvaril svojo uspešnico Pokémon GO. [9]

Unity ima visoko podporo, saj razvijalcem omogoča ustvarjanje paketov sredstev (npr. programske kode, modeli, scene, animacije ...) iz svojih projektov in jih delijo ali prodajo ter s tem, namesto da bi ponovno ustvarili podobne vire, drugim zelo olajšajo delo.

3.1.1 ARCore / AR Foundation

ARCore je knjižica podjetja Google za ustvarjanje aplikacij z obogateno resničnostjo. Z uporabo spletnih vmesnikov omogoča, da naprava zazna svoje okolje in se z njim sporazumeva. ARCore uporablja tri glavne zmogljivosti, s katerimi navidezni svet vključuje v realnega preko kamer [10]:

- sledenje gibanju, glede na položaj v okolju,
- zaznavanje velikosti in lokacijo vseh vrst površin,
- ocenjevanje svetlobnih pogojev okolja.

AR Foundation je razširitev te knjižice za uporabo v programu Unity, ki ima poleg vseh glavnih zmogljivosti ARCore vgrajene funkcije tudi za zaznavanje obraza, 3D objektov, slik in še več dodatnih funkcij [11].

3.1.2 Runtime OBJ importer

Je storitev, ki lahko modele v formatu OBJ zgradi med časom delovanja aplikacije. Omogoča branje objektov preko spletne povezave, nalaganja tekstur in materialov iz datoteke MTL [12].

3.1.3 AR + GPS Location

S to storitvijo lahko modele s pomočjo senzorjev GPS postavimo na geografske točke v realnem svetu. Glavne lastnosti storitve [13]:

- Postavitev 3D objekta na geografski položaj glede na širino, dolžino in nadmorsko višino.
- Postavitev 3D tekstovnih oznak na bližnje objekte (restavracija, hotel, bencinska črpalka, ...).
- Premikanje objektov ali postavitve objektov po poteh.
- Prikaz talnih senc objektov v obliki obogatene resničnosti.

- Realnočasovno premikanje lokacije naprave in posodabljanje.

3.2 Visual Studio

Je razvojno okolje za številne vrste aplikacij, od mobilnih do spletnih. Vključuje urejevalnik kod, razhroščevalnik, urejevalnik uporabniškega vmesnika in orodja za oblikovanje podatkovnih baz. Podpira programske jezike, kot so C++, C#, C, Python, HTML, CSS, JavaScript, Visual Basic, .NET, F#, Fossil in M. Slika 3.2 prikazuje trenutni logotip.



Slika 3.2: Logotip programa Visual Studio [14]

Zelo dobra razširjenost, integracija in uporabniški vmesnik programa so nam v pomoč pri hitrejšem in učinkovitejšem razvoju aplikacij. Za uporabo s programom Unity potrebujemo razširitev imenovano Visual Studio za Unity.

3.3 Sublime Text 3

Je urejevalnik kod za veliko programskih in označevalnih jezikov. Slika 3.3 prikazuje trenutni logotip tega programa. Je brezplačen program, ustvarjen s pomočjo uporabnikov za boljšo izkušnjo razvijanja kode. Glavne lastnosti programa so [16]:

- hitro navigacija ter preglednost med datotekami, simboli in vrsticami,
- enostavne bližnjice na tipkovnici,
- podpora za številne platforme,
- visoka prilagodljivost programa uporabnikom,
- razširljivost za veliko število programskih jezikov.



Slika 3.3: Logotip programa Sublime Text 3 [15]

3.4 XAMPP

Strežniška programska oprema XAMPP nam nudi primerno okolje za testiranje aplikacij MySQL, PHP, Apache in Perl na lokalnem računalniku, s čimer si lahko postavimo strežnik in ga tudi nadziramo. S priloženim orodjem phpMyAdmin pa lahko poleg tega postavimo tudi podatkovno bazo MySQL. Ima podporo na operacijskih sistemih Windows, Linux ter OS X. S pomočjo dodatkov lahko na lokalni strežnik postavimo upravljalnik vsebin, kot sta WordPress in Joomla. S pomočjo programa XAMPP lahko spremljamo tudi analitiko, status in varnost strežnika [17].

4 IZDELAVA SPLETNE APLIKACIJE

4.1 Načrtovanje spletne aplikacije

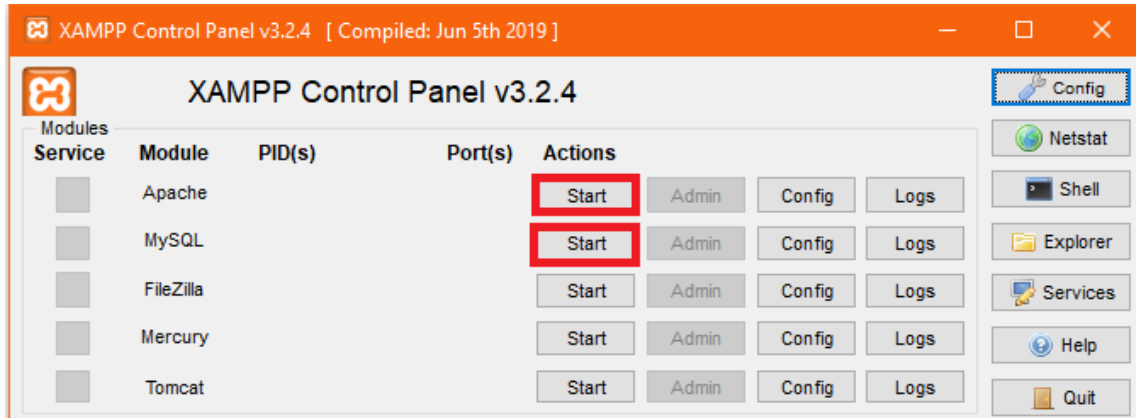
Cilj spletne aplikacije je ustvariti enostavno spletno podatkovno bazo z modeli objektov, ki so nato prikazani preko mobilne aplikacije. S tem smo ustvarili spletne vmesnike, ki te podatke izpišejo v obliki JSON. Za enostavnejše shranjevanje datotek modelov v formatu OBJ in tekstur v formatu PNG, smo te shranjevali z imenom številke glavnega ključa v podatkovni bazi, ki mu je ta model ali tekstura pripadala (npr. »1.obj« in »1.png«). Ustvarili smo tudi spletno stran, na kateri lahko dodajamo, spreminjamo ali brišemo modele oziroma podatke v podatkovni bazi.

4.2 Izdelava podatkovne baze

Tabelo modelov v podatkovni bazi smo gradili z naslednjimi atributi:

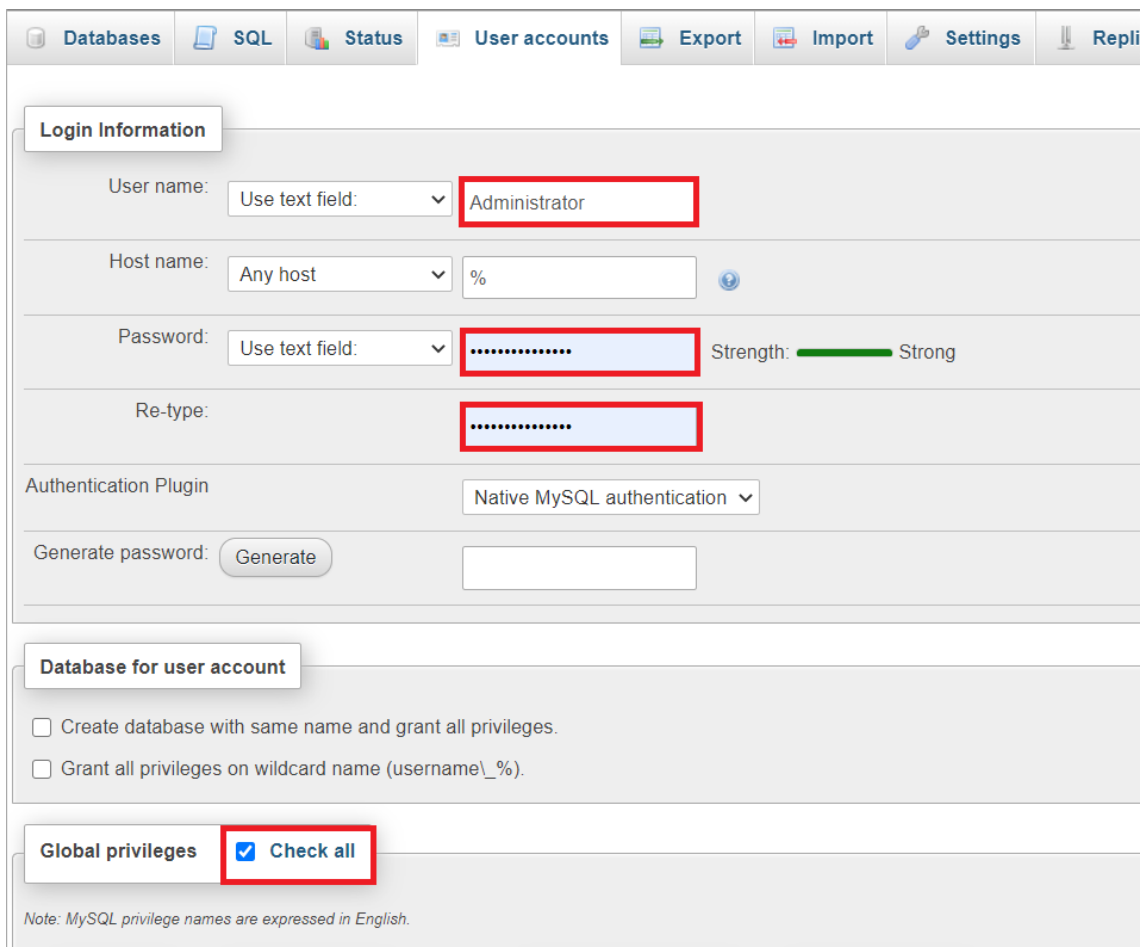
- **Id** - številka glavnega ključa.
- **Lat** - geografska širina, na kateri model je v realnem svetu.
- **Lon** - geografska dolžina, na kateri model je v realnem svetu.
- **Grounded** – določa, ali model je na določeni nadmorski višini ali pa naj ga naprava položi na površino relativno od lokacije naprave.
- **Alt** - nadmorska višina, na kateri model je v realnem svetu in ga uporabimo v primeru, če je »Grounded« enak 0.
- **Scale** – pove, za kolikšno vrednost se pomnoži velikost modela v mobilni aplikaciji.
- **Rotation** – določamo, za koliko stopinj je obrnjen model od severa.
- **Name** - ime objekta.
- **Description** - opis objekta.
- **Url** - povezava do več informacij o objektu.
- **Updated** - informacija o času zadnje spremembe podatkov.

Podatkovno bazo smo vzpostavili na takšen način, da smo znotraj aplikacije XAMPP vklopili modula Apache ter MySQL (slika 4.1) ter s tem tudi naš strežnik. Apache opravlja vsa strežniška dela, kot je prenašanje podatkov med strežnikom ter odjemalci, medtem ko MySQL skrbi za komunikacijo in upravljanje s podatkovno bazo.



Slika 4.1: Vklop lokalnega strežnika v programu XAMPP

Nato smo na spletnem naslovu »<http://localhost/phpmyadmin>« dostopali do orodja za upravljanje s podatkovno bazo, kjer smo pod oknom »User accounts« ustvarili uporabnika in mu nastavili ime (npr. »Administrator«), geslo (npr. »Adm1n_1stator«) ter mu v kategoriji »Global privileges« dodelili globalne pravice, kot prikazuje slika 4.2.



Login Information

User name: Use text field: Administrator

Host name: Any host %

Password: Use text field: Strength: Strong

Re-type:

Authentication Plugin: Native MySQL authentication

Generate password: Generate

Database for user account

☐ Create database with same name and grant all privileges.

☐ Grant all privileges on wildcard name (username_%).

Global privileges ☒ Check all

Note: MySQL privilege names are expressed in English.

Slika 4.2: Ustvarjanje uporabnika za podatkovno bazo

Ustvarili smo podatkovno bazo z imenom »ar_database« ter ji za format kodiranja črk nastavili »utf8_slovenian_ci«.

Po naslednjem postopku smo kreirali tabelo z imenom »objects« po atributih, opisanih v začetku poglavja. Atributu »Id« smo vključili lastnost »A_I«, s čimer določimo, da je ta glavni ključ in se z vsakim vnosom zviša za 1. Atribute »Lat«, »Lon« in »Alt« smo nastavili na tip »Double«, saj smo s tem shranjevali najbolj točno lokacijo modela. »Grounded« smo nastavili na tip »TinyInt« z dolžino 1 in ga vedno definirali z vrednostjo 1, kar v našem primeru pomeni, da se model privzeto nahaja na najbližji površini relativno od naprave. »TinyInt« uporabimo, saj MySQL ne podpira tipa »Boolean« in je to je število, ki ima na voljo samo en zlog (število tipa »Int« ima na voljo štiri zloge) ter je dovolj primerno za shranjevanje manjših vrednosti. Atribut »Alt« smo določili z vrednostjo 0,

saj se je ta upošteval le, če je vrednost atributa »Grounded« bila enaka 0. Atribut »Scale«, smo nastavili na tip »Float« in ga določili z vrednostjo 1, s čimer smo povedali, da je mobilna aplikacija ohranjala velikost modela. Prav tako smo atributu »Rotation« nastavili tip »Float« in ga določili z vrednostjo 0, kar pomeni, da je mobilna aplikacija ohranjala rotacijo modela. Atributoma »Name« in »Url« smo nastavili tip »Varchar« in ju znakovno omejili na vrednosti 50 za »Name« ter 100 za »Url«. »Description« smo nastavili na tip »Text«. Atributoma »Url« in »Description« smo označili okvirček »Null«, saj nista bila vedno obvezna. Atribut »Updated« smo nastavili na tip »Datetime«, za shranjevanje datuma in časa, ter mu določili vrednost »Current_time«, kar pomeni, da je shranjeval trenutni čas ob vnosu podatka. Izpolnjen obrazec je na sliki 4.3.

Name	Type	Length/Values	Default	Collation	Attributes	Null	A_I
<input type="text" value="Id"/> <small>Pick from Central Columns</small>	INT	11	None			<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="text" value="Lat"/> <small>Pick from Central Columns</small>	DOUBLE		None			<input type="checkbox"/>	<input type="checkbox"/>
<input type="text" value="Lon"/> <small>Pick from Central Columns</small>	DOUBLE		None			<input type="checkbox"/>	<input type="checkbox"/>
<input type="text" value="Grounded"/> <small>Pick from Central Columns</small>	TINYINT	1	As defined: 1			<input type="checkbox"/>	<input type="checkbox"/>
<input type="text" value="Alt"/> <small>Pick from Central Columns</small>	DOUBLE		As defined: 0			<input type="checkbox"/>	<input type="checkbox"/>
<input type="text" value="Scale"/> <small>Pick from Central Columns</small>	FLOAT		As defined: 1			<input type="checkbox"/>	<input type="checkbox"/>
<input type="text" value="Rotation"/> <small>Pick from Central Columns</small>	FLOAT		As defined: 0			<input type="checkbox"/>	<input type="checkbox"/>
<input type="text" value="Name"/> <small>Pick from Central Columns</small>	VARCHAR	50	None	utf32_slovenian_		<input type="checkbox"/>	<input type="checkbox"/>
<input type="text" value="Description"/> <small>Pick from Central Columns</small>	TEXT		NULL	utf32_slovenian_		<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input type="text" value="Url"/> <small>Pick from Central Columns</small>	VARCHAR	100	NULL	utf32_slovenian_		<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input type="text" value="Updated"/> <small>Pick from Central Columns</small>	DATETIME		CURRENT_TIME			<input type="checkbox"/>	<input type="checkbox"/>

Slika 4.3: Ustvarjanje podatkovne baze

4.3 Izdelava spletnega vmesnika

V nadaljevanju smo s pomočjo programa Sublime Text 3 ustvarili spletni vmesnik za sporazumevanje podatkovne baze z mobilno aplikacijo. Želeli smo doseči izpis podatkov v obliki formata JSON za en posamezni model (kot parameter smo uporabili glavni ključ) ali za vse modele, ki so bili od lokacije, pridobljene iz parametrov v spletni povezavi do vmesnika, oddaljeni do tisoč metrov. Prav tako smo želeli ustvariti vmesnik za vnos, spreminjanje ali izbris modela iz podatkovne baze.

4.3.1 Povezava do podatkovne baze

Da smo lahko upravljali podatkovno bazo, se je bilo treba s to najprej povezati. V spremenljivke smo shranili ime podatkovne baze in podatke o uporabniku, ki smo jih ustvarili v prejšnjem poglavju, kot prikazuje slika 4.4. S temi smo s pomočjo funkcije »mysqli_connect« vzpostavili povezavo s podatkovno bazo in jo shranili v spremenljivko (npr. »\$conn«), kot prikazuje slika 4.4. Če povezava ni uspešna, uničimo proces in ugotovimo vir napake.

```
<?php
$servername = "localhost";
$username = "Administrator";
$password = "Admin_1strator";
$db_name = "ar_database";

$conn = mysqli_connect($servername, $username, $password,$db_name);
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
?>
```

Slika 4.4: Povezava do podatkovne baze

4.3.2 Izpis modelov iz podatkovne baze

S stavkom »SELECT * FROM objects« smo pridobili vse podatke iz naše podatkovne baze. Te smo v formatu JSON shranjevali v spremenljivko (npr. »\$print«), kot prikazuje slika 4.5.

```

$query = sprintf("SELECT * FROM objects");
$result = $conn->query($query) or die($conn->error.__LINE__);
$print = '[';
if($result->num_rows > 0) {
    while($row = $result->fetch_assoc())
    {
        if(isset($_GET['lat']) && isset($_GET['lon'])){
            if(vincentyGreatCircleDistance($_GET["lat"],
                                            $_GET["lon"],
                                            $row['Lat'],
                                            $row['Lon']) < 1000){
                $print .= '{"Id":'.$row['Id'].',"Lat":'.$row['Lat'].',"Lon":'.$
                $row['Lon'].',"Grounded":'.$row['Grounded'].',"Alt":'.$
                $row['Alt'].',"Scale":'.$row['Scale'].',"Rotation":'.$
                $row['Rotation'].',"Name":'.$row['Name'].',"Description":'.$
                $row['Description'].',"Url":'.$row['Url'].',"Updated":'.$
                $row['Updated'].',"},';
            }
        }
        else{
            $print .= '{"Id":'.$row['Id'].',"Lat":'.$row['Lat'].',"Lon":'.$
            $row['Lon'].',"Grounded":'.$row['Grounded'].',"Alt":'.$
            $row['Alt'].',"Scale":'.$row['Scale'].',"Rotation":'.$
            $row['Rotation'].',"Name":'.$row['Name'].',"Description":'.$
            $row['Description'].',"Url":'.$row['Url'].',"Updated":'.$
            $row['Updated'].',"},';
        }
    }
}
$print = substr($print, 0, -1);
echo $print.']';
exit;

```

Slika 4.5: Izpis podatkov modelov do razdalje tisoč metrov ali vseh modelov iz podatkovne baze

Preverimo, ali sta v podatkih, ki smo jih dobili preko povezave do spletnega vmesnika, spremenljivki »lon« in »lat«, ki ponazarjata določeno zemljepisno lokacijo, nastavljeni na kakšno vrednost. Če je bil ta pogoj izpolnjen, smo s pomočjo funkcije »vincentyGreatCircircleDistance« izračunali oddaljenost po metodi za izračun najkrajše razdalje med dvema točkama na kroglu (slika 4.6) in preverili, ali je ta manjša od tisoč metrov ter tako izpisovali le modele v tem dosegu.

```

function vincentyGreatCircleDistance(
    $latitudeFrom, $longitudeFrom, $latitudeTo,
    $longitudeTo, $earthRadius = 6371000)
{
    // convert from degrees to radians
    $latFrom = deg2rad($latitudeFrom);
    $lonFrom = deg2rad($longitudeFrom);
    $latTo = deg2rad($latitudeTo);
    $lonTo = deg2rad($longitudeTo);

    $lonDelta = $lonTo - $lonFrom;
    $a = pow(cos($latTo) * sin($lonDelta), 2) +
        pow(cos($latFrom) * sin($latTo) - sin($latFrom) *
            cos($latTo) * cos($lonDelta), 2);
    $b = sin($latFrom) * sin($latTo) + cos($latFrom) *
        cos($latTo) * cos($lonDelta);

    $angle = atan2(sqrt($a), $b);
    return $angle * $earthRadius;
}

```

Slika 4.6: Izračun najkrajše razdalje med dvema točkama na površini krogle

Prav tako smo ustvarili spletni vmesnik za izpis podatkov posameznega objekta s stavkom »SELECT * FROM object WHERE Id = \$id«, kjer je spremenljivka »\$id« številka glavnega ključa specifičnega modela v podatkovni bazi (slika 4.7).

```

$id = $_GET["id"];

$query = sprintf("SELECT * FROM objects WHERE Id = ".$id);
$result = $conn->query($query) or die($conn->error.__LINE__);
$print = '';
if($result->num_rows > 0) {
    while($row = $result->fetch_assoc())
    {
        $print .= '{"Id":'.$row['Id'].','."Lat":'."$row['Lat']".
            ','."Lon":'."$row['Lon']".','."Grounded":'."$row['Grounded']".
            ','."Alt":'."$row['Alt']".','."Scale":'."$row['Scale']".
            ','."Rotation":'."$row['Rotation']".','."Name":'."$row['Name']".
            ','."Description":'."$row['Description']".','."Url":'."$row['Url']".
            ','."Updated":'."$row['Updated']".','."}';
    }
}
$print = substr($print, 0, -1);
echo $print;
exit;

```

Slika 4.7: Izpis podatkov enega modela iz podatkovne baze

4.3.3 Vnos novega modela v podatkovno bazo

Podatke, ki smo jih iz obrazca na spletni vmesnik pridobili z metodo »POST«, shranimo v spremenljivke in jih nato s stavkom »INSERT«, ki ga prikazuje slika 4.8, vnesli v podatkovno bazo. Če se je to izvedlo, smo lahko datoteko našega modela v formatu OBJ shranili v posebno datoteko na našem strežniku (npr. »/Models/«) pod imenom številke zadnjega vnesenega glavnega ključa in končnico ».obj«. Zatem smo to ponovili tudi s teksturo v formatu PNG. Ker na tem spletnem vmesniku nič ne izpisujemo, nadaljujemo nazaj na glavno stran »index.php«.

```
$sql = "INSERT INTO objects (Lat,Lon,Grounded,Alt,
Scale,Rotation,Name,Description,Url)
VALUES ('$lat','$lon','$alt_check','$alt',
'$scale','$rotation','$name','$desc','$url')";

if ($conn->query($sql) === TRUE) {
    $last_id = $conn->insert_id;
    //FILE UPLOAD .OBJ
    if (move_uploaded_file($_FILES["obj"]["tmp_name"],
        $target_dir.$last_id.".obj")) {
        //FILE UPLOAD .PNG
        if (move_uploaded_file($_FILES["texture"]["tmp_name"],
            $target_dir.$last_id.".".$textureType)) {
            mysqli_close($conn);
            header("Location: ../index.php");
        } else {
            echo "Sorry, there was an error uploading your texture file.";
        }
    } else {
        echo "Sorry, there was an error uploading your .obj file.";
    }
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}
```

Slika 4.8: Vstavljanje modela v podatkovno bazo

4.3.4 Spreminjanje podatkov modela v podatkovni bazi

Podatke, ki so bili iz obrazca poslani z metodo »POST«, smo shranili v spremenljivke. S pomočjo stavka »UPDATE«, ki ga prikazuje slika 4.9, smo te podatke spremenili in nadaljevali nazaj na glavno stran.

```

$sql = "UPDATE  objects SET Lat = '$lat', Lon = '$lon',
Grounded = '$alt_check', Alt = '$alt', Scale = '$scale',
Rotation = '$rotation', Name = '$name',
Description = '$desc',Url = '$url' WHERE Id=" . $id;

if ($conn->query($sql) === TRUE) {
    header("Location: ../index.php?id=" . $id);
} else {
    echo "Error updating record: " . $conn->error;
}

```

Slika 4.9: Spreminjanje podatkov modela v podatkovni bazi

4.3.5 Izbris modela iz podatkovne baze

Potrebovali smo število glavnega ključa, ki smo ga dobili iz parametrov, pridobljenih preko povezave do vmesnika. Tega uporabimo v stavku »DELETE« (slika 4.10), ki podatek z njegovo vrednostjo v podatkovni bazi izbriše. Če se je to izvedlo, smo lahko iz datoteke na strežniku izbrisali še model in teksturo s funkcijo »unlink«.

```

$id = $_GET["id"];
// sql to delete a record
$sql = "DELETE FROM objects WHERE id=" . $id;

if ($conn->query($sql) === TRUE) {
    $files = glob("../Models/" . $id . ".*");
    foreach ($files as $file) {
        unlink($file);
    }
    echo "Record deleted successfully";
} else {
    echo "Error deleting record: " . $conn->error;
}

```

Slika 4.10: Izbris modela iz podatkovne baze

4.4 Izdelava spletne strani

Izdelali smo tudi spletno stran z elementi za vnos, spreminjanje, možnostjo izbrisa ter pogleda na že vnesene modele v podatkovni bazi.

Najprej smo ustvarili polja za vnos vseh potrebnih podatkov v podatkovno bazo, kar vključuje izbiro datoteke modela in texture, ki ju je treba priložiti. Zraven polja za nadmorsko višino smo postavili okvirček, s katerim določamo, ali je naš podatek

»Grounded« enak vrednosti 1 ali 0. Če tega označimo, omogočimo tudi polje za vnos nadmorske višine.

Na spletno stran smo dodali tudi tabelo, ki s pomočjo ustvarjenega spletnega vmesnika prikazuje vse modele, shranjene v podatkovni bazi. Ob kliku na številko glavnega ključa se nam odpre spletni obrazec za spreminjanje podatkov tega modela. Na tem obrazcu ne moremo spreminjati datotek modela ali teksture, temveč samo podatke.

5 IZDELAVA 3D MODELA OBJEKTA

5.1 Zasnova 3D modela objekta kulturne dediščine

Za objekt, ki smo ga modelirali, smo izbrali cerkev svetega Mihaela iz Družmirja. Ta kraj je sedaj zaradi premogovniških del pod vodno gladino Šoštanjskega jezera in spomin na to kulturno dediščino ostaja samo še v slikovnem formatu, zato je bila ta cerkev odličen primer za uporabo v naši diplomski nalogi. Iz slik 5.1 in 5.2, ki sta razstavljeni tudi v Vili Mayer, lahko ustvarimo približni načrt za uporabo pri modeliranju (slika 5.3).



Slika 5.1: Cerkev svetega Mihaela v Družmirju pred in po zrušitvi [18]



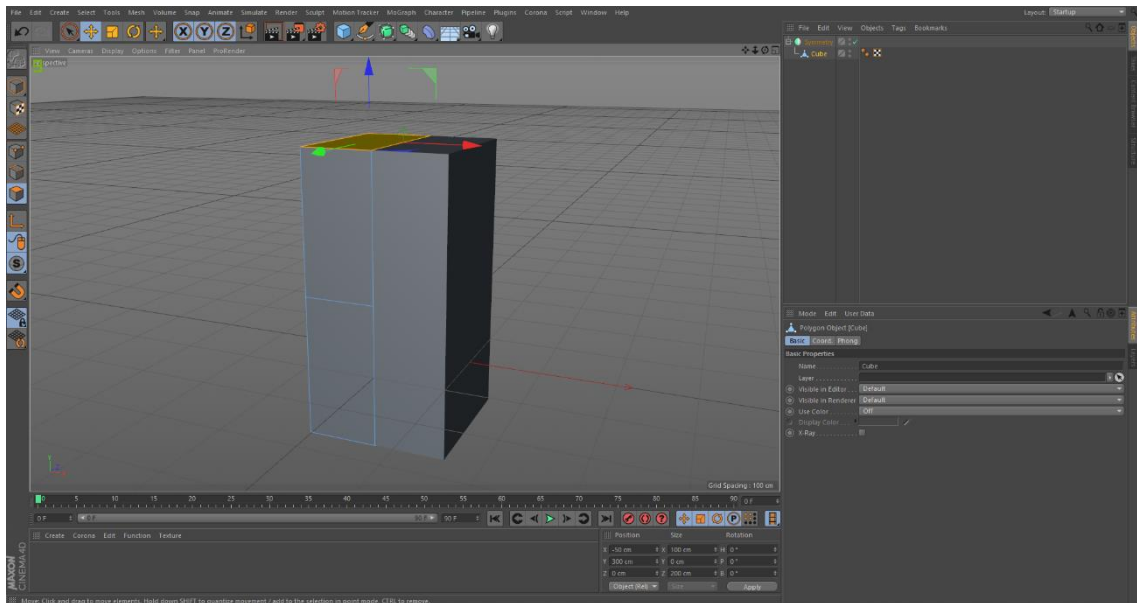
Slika 5.2: Razglednica cerkve svetega Mihaela v Družmirju [19]



Slika 5.3: Načrt za izdelavo 3D modela po sliki 5.1

5.2 Ustvarjanje 3D modela objekta kulturne dediščine

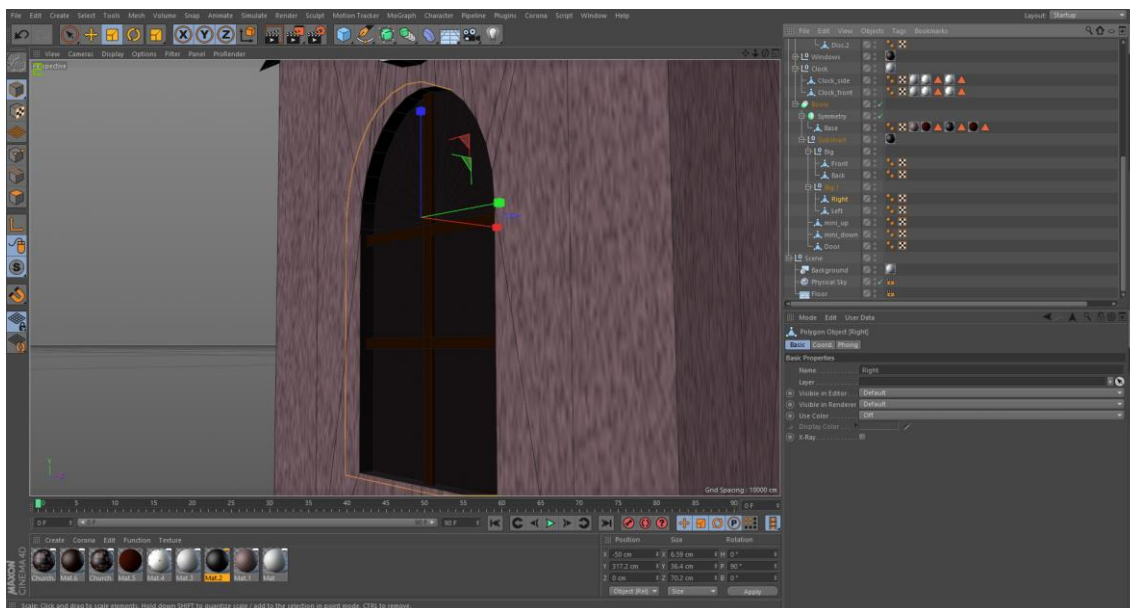
Po načrtu iz prejšnjega dela smo lahko v enem od 3D animacijskih paketov konstruirali 3D model. V sceno smo postavili kocko, omogočili urejanje in začeli graditi z iztiskanjem (angl. extrude) ali premikanjem površin, robov ali oglišč. Ker je naš objekt simetričen, smo uporabili funkcijo simetrale (angl. symmetry), kar omogoča, da urejamo samo eno stran, ki se potem preslika (slika 5.4).



Slika 5.4: Manipuliranje objekta in uporaba simetrale

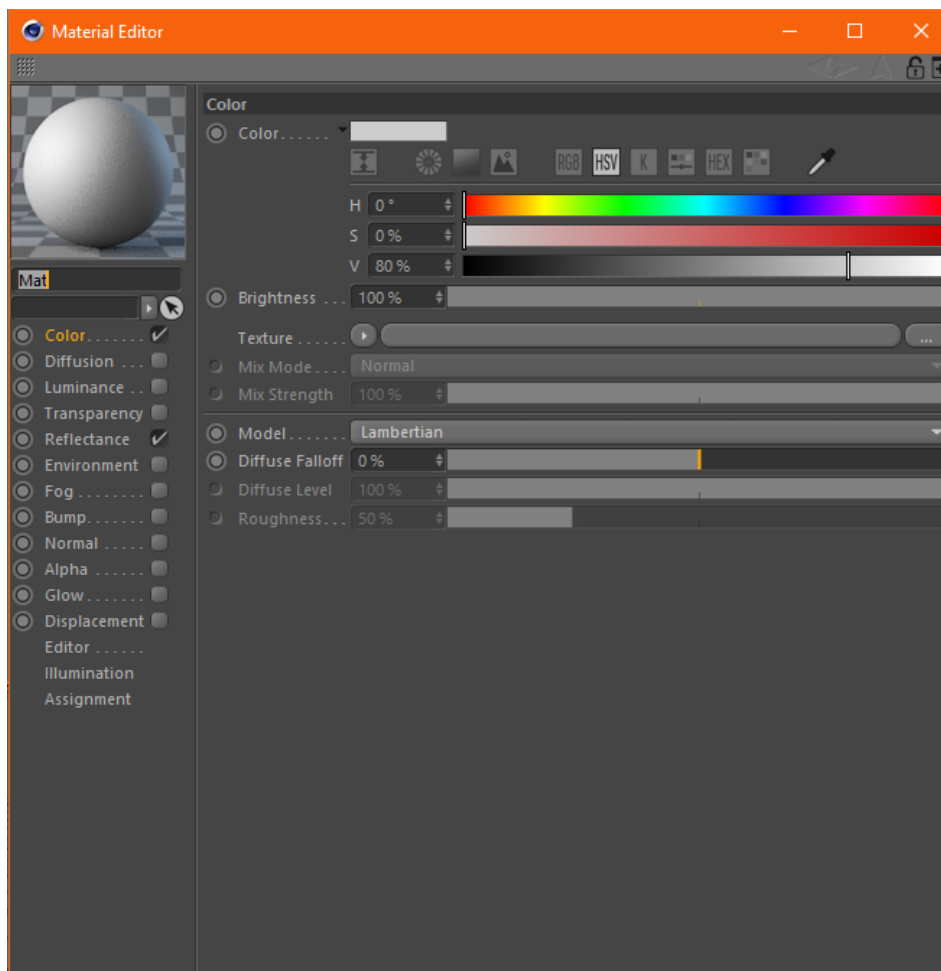
Pri tem smo pazili, da se je na objektu ohranilo čim manjše število trikotnikov, saj je spreminjanje preprostejših modelov lažje, končni model pa je bolj primeren za našo uporabo. Manjše število trikotnikov pomeni hitrejši prenos in nalaganje.

Pri izrezih za okna cerkve smo uporabili logične funkcije (angl. boolean), ki iz poljubne oblike izrežejo ali dodajo drugo obliko. Tako smo iz našega modela cerkve izrezali pol-ovalno obliko in s tem ponazorili okno ter s to funkcijo nanj dodali tudi okvir, kot prikazuje slika 5.5.



Slika 5.5: Izrez okna v modelu z uporabo logičnih funkcij

Ko smo bili z obliko modela zadovoljni, smo temu dodelili še zunanji videz. To storimo z materiali, ki smo jim določili barvo, teksturo, prozornost, odbojnost in ostale značilnosti, ki jih prikazuje slika 5.6.

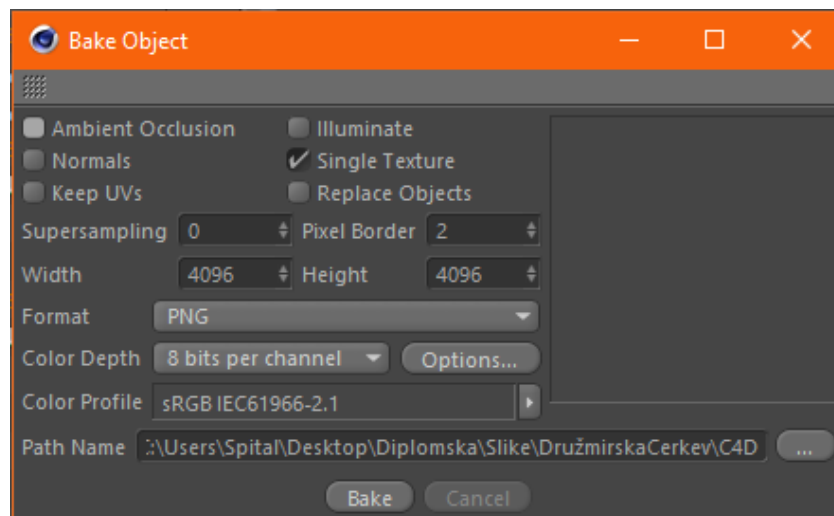


Slika 5.6: Upravljaec materialov

Ta material smo uporabili tako, da smo na modelu označili želeno ploskev in material iz menija enostavno potegnili nanjo. To smo ponavljali, dokler nismo prekrili vseh ploskev z želenimi materiali. Pred izvozom smo morali paziti tudi na smer normale mnogokotnikov. Če je bil vektor normale mnogokotnika obrnjen v enako smer, kot je bil vektor normale našega pogleda v mobilni aplikaciji, tega mnogokotnika zaradi optimizacije nismo prikazovali. S tako imenovano optimizacijo »back-face culling« želimo v grafiki pospešiti upodabljanje s tem, da skrijemo stranice, ki so zakrite oziroma obrnjene stran od kamere.

5.3 Priprava 3D modela za prikaz v mobilni aplikaciji

Uporabljene materiale na ustvarjenem modelu je bilo treba pretvoriti v teksturo v slikovni obliki. To smo storili s tako imenovano funkcijo zapeči objekt (angl. bake object), ki je prikazana na sliki 5.7. Pri tem je bilo pomembno izbrati čim manjšo velikost slike za hitrejši prenos preko interneta, vendar dovolj veliko, da se tekstura ni popačila. Prav tako smo pazili, da izvozimo teksturo v format PNG. Z izbiro tega smo zagotovili lažje ustvarjanje aplikacije, saj nam pri tem ni treba skrbeti glede uporabe drugih formatov.



Slika 5.7: Pečenje objekta

Nato smo model izvozili v format OBJ. To datoteko in datoteko texture smo shranili v podatkovno bazo s pomočjo naše spletne aplikacije. Izdelan 3D model je imel 2540 mnogokotnikov, kar je bilo dovolj malo število za uporabo v mobilni aplikaciji. Iz slike 5.8 lahko razberemo nekdanjo lokacijo cerkve svetega Mihaela v Družmirju, ki je $46,376876^\circ$ po geografski širini in $15,056229^\circ$ po geografski dolžini. Zaradi boljše dostopnosti in vidljivosti smo lokacijo zamaknili na bližnji travnik. Prav tako je bilo iz slike možno razviti, da je objekt bil usmerjen -90° od severa. V tem primeru se je objekt nahajal na talni površini relativno od naprave.

Ker pri izvozu iz 3D urejevalnika nismo pazili na merske enote, smo v programu Unity morali uporabiti še faktor skaliranja, ki je v našem primeru 0,025, saj je bila pretvorba enot tega modela iz 3D urejevalnika v Unity v razmerju 400:1. V spletni obrazec smo

dodali še ustrezno ime, opis ter povezavo. Po potrditvi je spletna aplikacija model zapisala v podatkovno bazo.



Slika 5.8: Družmirje nekoč in danes [20]

6 IZDELAVA MOBILNE APLIKACIJE

6.1 Načrtovanje mobilne aplikacije

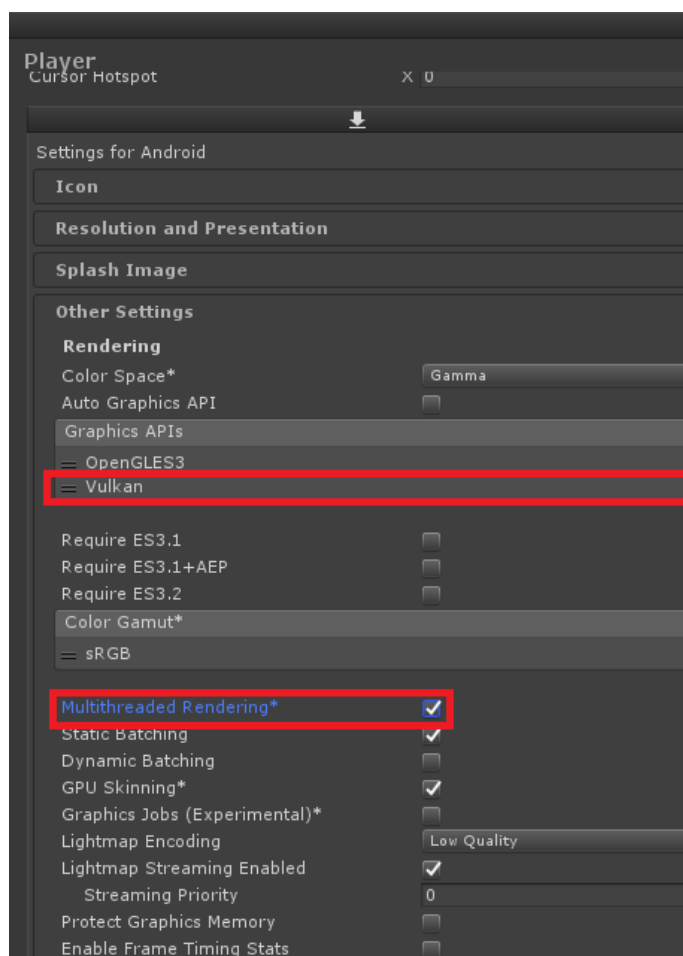
Mobilna aplikacija preko spletnih vmesnikov pridobiva podatke iz podatkovne baze o objektih, ki so v oddaljenosti do tisoč metrov od lokacije naprave. Podatke aplikacija posodablja vsakih 800 metrov od začetne lokacije. Pri tem objektov, ki so izven tega območja, ne prikazuje, saj tako dosežemo boljše delovanje mobilne aplikacije. Ko ta pridobi podatke iz podatkovne baze, aplikacija naloži še pripadajoče 3D modele in nanj naloži še teksturo. Za lažjo koordinacijo smo v mobilni aplikaciji omogočili tudi okno, ki prikazuje bližnje objekte in razdalje do njih. Ko uporabnik pritisne na objekt, se mu prikažejo tudi informacije o njem, hkrati pa tudi gumb, s katerim lahko odpre spletno povezavo do več informacij o tem objektu. V mobilni aplikaciji sta tudi gumba, pri katerem prvi osveži podatke in drugi shrani zajeto sliko brez uporabniškega vmesnika. V aplikaciji je treba omogočiti dovoljenja in dostop do GPS senzorjev ter internetne povezave za uporabo vseh funkcionalnosti. Če uporabnik teh nima, mu aplikacija to sporoči z opozorilnim oknom.

6.2 Postopek izdelave mobilne aplikacije

6.2.1 Inicializacija projekta

Ustvarili smo nov 3D projekt v programu Unity, kjer je najprej bilo treba naložiti vse knjižnice in orodja za naše potrebe. Odprli smo upravitelja paketov (angl. package manager), ki se nahaja v meniju pod kategorijo okno (angl. window) in v njem poiskali in naložili paketa »AR Foundation« in »ARCore XR Plugin«, ki sta omogočila uporabo obogatene resničnosti za platformo Android. Iz okna za nakup sredstev (angl. Asset Store) smo naložili tudi paketa »AR + GPS Location« in »OBJ Importer«. Pred začetkom pisanja kode je bilo treba v nastavitvah projekta (angl. project settings), pod kategorijo igralec (angl. player), za platformo Android iz polja »Graphics API« izbrisati »Vulkan« in

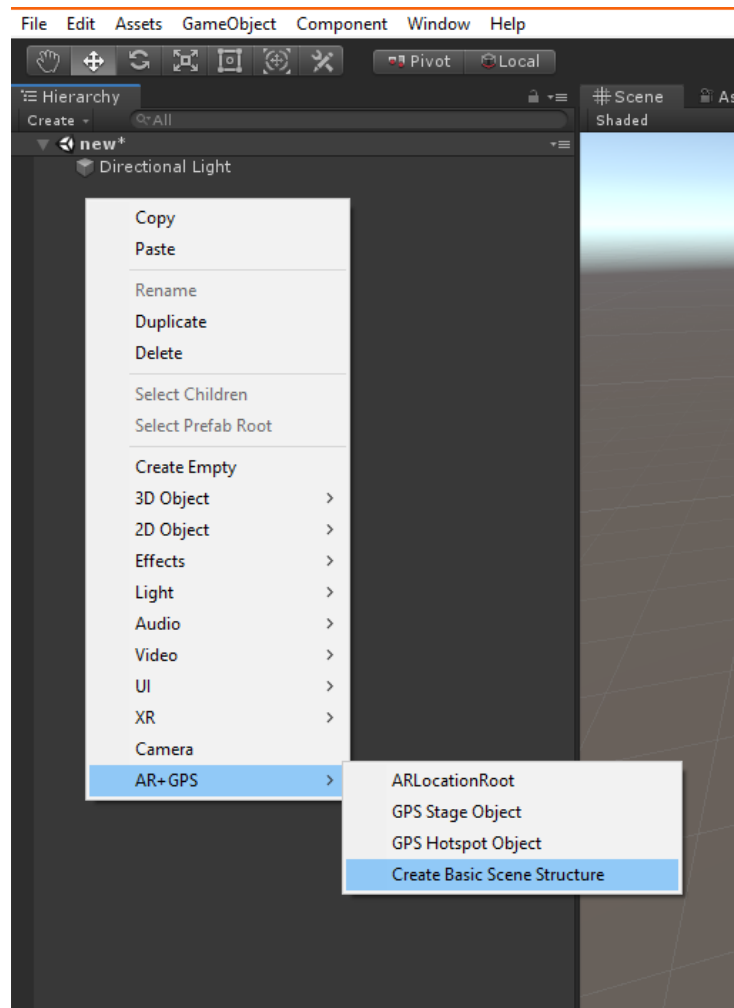
počistiti polje za večnitno upodabljanje (angl. »multithreaded rendering«), kot prikazuje slika 6.1, saj orodja za obogateno resničnost v programu Unity teh funkcij ne podpirajo.



Slika 6.1: Inicializacija glavnih nastavitev aplikacije

Za delovanje orodja »OBJ Importer« je bilo potrebno v kategoriji grafika (angl. graphics) dodati še način senčenja imenovan »Legacy shader/Diffuse«, saj je aplikacija na uporabljeni mobilni napravi le tako lahko prikazovala naše modele.

V hierarhiji scene smo izbrisali trenutno kamero in v sceno dodali vse potrebne elemente s klikom na desni gumb, kjer smo izbrali »AR + GPS« → »Create Basic Scene Structure«, kot prikazuje slika 6.2. Tako smo imeli v projektu vse potrebne elemente za razvoj aplikacije.



Slika 6.2: Inicializacija scene

6.2.2 Branje podatkov iz spletnega vmesnika

Za branje podatkov iz spletnega vmesnika smo ustvarili novo ukazno datoteko v jeziku C# in uporabili funkcijo »WWW«, kot prikazuje slika 6.3. Ko je ta podatke prebrala, je te bilo treba pretvoriti iz formata JSON v razred, ki ima enake vrednosti kot podatkovna baza. Ta razred (slika 6.4) je bilo potrebno označiti kot »[serializable]«, da je bila pretvorba mogoča.

```

IEnumerator GetData()
{
    var www = new WWW(path + "api/get_objects.php?lat="+
        initialLocation.latitude+"&lon="+initialLocation.longitude);
    while (!www.isDone)
    {
        System.Threading.Thread.Sleep(1);
        Debug.Log(www.progress);
    }
    if (string.IsNullOrEmpty(www.error))
    {
        ParseData(www.text);
    }
    else
    {
        Debug.Log(www.error);
    }
    yield return null;
}

void ParseData(string data)
{
    data = "{\"Items\":\"" + data + "\"}";
    objs = JsonHelper.FromJson<Objects>(data);

    foreach(Objects o in objs)
    {
        if(o.Grounded == 1)
        {
            Load(o.Id, o.Scale, o.Rotation, o.Lat, o.Lon, o.Alt, true);
        }
        else
        {
            Load(o.Id, o.Scale, o.Rotation, o.Lat, o.Lon, o.Alt, false);
        }
    }
}

```

Slika 6.3: Pridobivanje in serializacija podatkov iz podatkovne baze

```

[Serializable]
public class Objects
{
    public int Id;
    public double Lat;
    public double Lon;
    public double Alt;
    public int Grounded;
    public float Scale;
    public float Rotation;
    public string Name;
    public string Description;
    public string Url;
    public string Updated;
}

```

Slika 6.4: Razred, ustvarjen po shemi podatkov v podatkovni bazi

6.2.3 Nalaganje 3D modelov v aplikacijo

Za lažjo preglednost smo funkcije za nalaganje 3D modelov napisali v novi ukazni datoteki. V tej smo s pomočjo funkcije »WWW« prav tako pridobili informacije o 3D modelu, ki se nahaja na strežniku v mapi »Models« pod imenom, ki ga določata številka primarnega ključa podatka v podatkovni bazi ter končnica ».obj«. Te informacije o 3D modelu smo posredovali funkciji »Load« iz orodja »OBJLoader«, ki smo ga naložili ob inicializaciji. Zatem smo poskrbeli, da je naš model v aplikaciji pravilno pomanjšan oziroma povečan in rotiran glede na naše podatke. Da zaznamo, kdaj uporabnik s prstom pritisne na ta objekt, za prikaz več informacij, mu je treba dodati tudi mrežasti trkalnik (angl. mesh collider). Spremenili smo tudi ime ustvarjenega objekta, da je ta enak številki primarnega ključa. Nato smo na model naložili še slikovno teksturo, ki jo prav tako preberemo iz strežnika s pomočjo funkcije »WWW«, in jo vstavili v material ustvarjenega objekta. Ta postopek prikazuje slika 6.5.

```

IEnumerator GetObject(int id, string uri, float scale, float rotation)
{
    var www = new WWW(uri + ".obj");
    while (!www.isDone)
    {
        System.Threading.Thread.Sleep(1);
        Debug.Log(www.progress);
    }

    //create stream and load
    var textStream = new MemoryStream(Encoding.UTF8.GetBytes(www.text));
    loadedObject = new OBJLoader().Load(textStream);
    loadedObject.transform.localRotation = Quaternion.Euler(0, rotation, 0);
    loadedObject.transform.localScale = new Vector3(scale, scale, scale);
    loadedObject.transform.GetChild(0).gameObject.AddComponent<MeshCollider>();
    loadedObject.transform.GetChild(0).name = id.ToString();
    loadedObject.name = id.ToString();
    //add textures
    GetTexture(uri);
    yield return null;
}

void GetTexture(string uri)
{
    //find path
    var www = new WWW(uri + ".png");
    while (!www.isDone)
    {
        System.Threading.Thread.Sleep(1);
        Debug.Log(www.progress);
    }

    loadedObject.transform.GetChild(0).
        GetComponent<MeshRenderer>().material.mainTexture = www.texture;
}

```

Slika 6.5: Ustvarjanje objekta in teksturiranje iz datotek na strežniku

Nato je bilo treba ustvarjen objekt postaviti na geografsko lokacijo. Pozicijo naloženega modela smo najprej postavili na ničelno koordinato (slika 6.6). Ustvarili smo nov lokacijski objekt in vanj vnesli podatke o geografski širini in dolžini ter nadmorski višini in pri tem upoštevali, da se objekt lahko nahaja tudi na talni površini relativno od naprave. Nastavili smo tudi parametre za bolj gladko premike modela v prostoru. S funkcijo »AddPlaceAtComponent« iz orodja »AR + GPS Location«, ki poskrbi za pretvorbo med koordinatami GPS in lokalnimi koordinatami v programu Unity, smo objekt postavili na geografsko lokacijo. Ta postopek prikazuje slika 6.6.

```

IEnumerator PlaceInSpace(double lat, double lon, double alt, bool grounded)
{
    loadedObject.transform.position = Vector3.zero;
    Location loc;
    if (grounded)
    {
        loc = new Location()
        {
            Latitude = lat,
            Longitude = lon,
            Altitude = 0,
            AltitudeMode = AltitudeMode.GroundRelative
        };
    }
    else
    {
        loc = new Location()
        {
            Latitude = lat,
            Longitude = lon,
            Altitude = alt,
            AltitudeMode = AltitudeMode.Absolute
        };
    }
    var opts = new PlaceAtLocation.PlaceAtOptions()
    {
        HideObjectUntilItIsPlaced = true,
        MaxNumberOfLocationUpdates = 2,
        MovementSmoothing = 0.1f,
        UseMovingAverage = false
    };
    PlaceAtLocation.AddPlaceAtComponent(loadedObject, loc, opts, true);
    loadedObjects.Add(loadedObject);
    yield return null;
}

```

Slika 6.6: Postavitev objekta v geografsko okolje

6.2.4 Osveževanje podatkov

V aplikaciji vsako minuto pregledamo, ali je razdalja od naše trenutne večja kot 800 metrov od prvotne lokacije. Če to velja, potem iz scene izbrišemo vse modele in jih ponovno naložimo glede na novo lokacijo. Prvotno shranjeno lokacijo nato nadomestimo z novo (slika 6.7).

```

private void Update()
{
    refresh += Time.deltaTime;

    if (refresh >= 60f)
    {
        if (GetDistance(initialLocation.latitude,
            initialLocation.longitude) > refreshDistance)
        {
            loadObjects.ClearAll();
            StartCoroutine(GetData());
            initialLocation = Input.location.lastData;
        }
        refresh = 0f;
    }
}

```

Slika 6.7: Izsek kode za osveževanje podatkov

6.2.5 Prikaz podatkov o objektu

Pregledovali smo tudi zaznavanja pritiska uporabnika na zaslon. Če je pritisk na lokaciji objekta, ki ga na zaslonu prikazuje aplikacija, potem odpremo okno, v katerem se prikažejo informacije o tem objektu (slika 6.8).

```

if (Input.touchCount > 0 && Input.GetTouch(0).phase == TouchPhase.Ended)
{
    Ray ray = Camera.main.ScreenPointToRay(Input.GetTouch(0).position);
    RaycastHit hit;
    if (Physics.Raycast(ray, out hit, 200))
    {
        if (hit.transform.name != null)
        {
            oi.ShowInfo(int.Parse(hit.transform.name));
        }
    }
}

```

Slika 6.8: Izsek kode za preverjanje dotika uporabnika

Če se v podatkovni bazi v polju »Url« nahaja tudi povezava, v tem oknu prikažemo gumb, ki to povezavo odpre. Okno je mogoče zapreti na pritisk gumba z oznako »X«. Izpis teh podatkov prikazuje slika 6.9.


```

public void ShowInfo(int id)
{
    Objects o = getObjects.GetLoadedObject(id);
    if(o != null)
    {
        url = o.Url;
        if(url == null || url == "")
        {
            moreBtn.SetActive(false);
        }
        else
        {
            moreBtn.SetActive(true);
        }
        title.text = o.Name;
        text.text = o.Description;
        InfoCanvas.SetActive(true);
        MainCanvas.SetActive(false);
    }
}

public void OnClose()
{
    InfoCanvas.SetActive(false);
    MainCanvas.SetActive(true);
}

public void OnMore()
{
    Application.OpenURL(url);
}

```

Slika 6.9: Izsek kode za prikaz informacij o objektu

6.2.6 Prikaz informacij o bližnjih objektih

Uporabniku smo omogočili tudi poizvedbo razdalj do vseh okolišnih modelov, ki smo jih naložili v aplikacijo. Ob pritisku na gumb se nam odpre okno s seznamom, ki prikazuje ime objekta in število, ki ponazarja razdaljo do objekta. Na sliki 6.10 se nahaja izpis seznama in funkcija za pridobitev razdalje do posameznega objekta.

```

void UpdateList()
{
    foreach (Transform child in content.transform)
        Destroy(child.gameObject);

    List<Items> items = new List<Items>();
    foreach (Objects model in obj)
    {
        items.Add(new Items(model.Name, GetDistance(model.Lat, model.Lon)));
    }

    items = items.OrderBy(d => d.distance).ToList();
    foreach (Items i in items)
    {
        if (i.distance < 1000)
        {
            var instance = Instantiate(listPrefab.gameObject) as GameObject;
            instance.GetComponent<Item>().name.text = " " + i.name;
            instance.GetComponent<Item>().distance.text =
                ((int)i.distance).ToString() + "m";
            instance.transform.SetParent(content.transform, false);
        }
    }
}

double GetDistance(double lat, double lon)
{
    return Location.PlaneEllipsoidalFccDistance(
        new Location(Input.location.lastData.latitude,
            Input.location.lastData.longitude),
        new Location(lat, lon));
}

```

Slika 6.10: Izsek kode za prikaz vseh bližnjih objektov

6.2.7 Zajemanje slike zaslona

Ob kliku na gumb se v napravo shrani tudi zajem slike zaslona. Pri tem onemogočimo uporabniški vmesnik in počakamo, da se vsi procesi v tem kadru zaključijo. Zajeto sliko shranimo pod imenom s trenutno časovno oznako. Po tej izvedbi uporabniški vmesnik ponovno omogočimo. Ta postopek prikazuje slika 6.11.

```

public void OnScreenshot()
{
    StartCoroutine(CaptureScreen());
}

private IEnumerator CaptureScreen()
{
    yield return null;
    MainCanvas.SetActive(false);

    yield return new WaitForEndOfFrame();

    string myFileName = "screenshot_" +
        DateTime.Now.ToString("ddMMyyyy_HHmssddMMyyyy") + ".png";

    ScreenCapture.CaptureScreenshot(myFileName);

    MainCanvas.SetActive(true);
}

```

Slika 6.11: Izsek kode za zajem slike zaslona

6.2.8 Dostop do GPS senzorja in internetne povezave

Aplikacija za delovanje ves čas potrebuje povezavo z internetom ter podatke iz senzorjev GPS. Za pridobivanje dovoljenja dostopa poskrbi orodje »AR + GPS Location«, vendar lahko uporabnik med uporabo aplikacije izgubi ali izključi povezavo z internetom oziroma z GPS. V tem primeru se mu v aplikaciji prikaže okno, ki ga na to opozori. Okno se samodejno zapre, ko se povezava ponovno vzpostavi. Slika 6.12 prikazuje izsek kode za preverjanje povezljivosti.

```

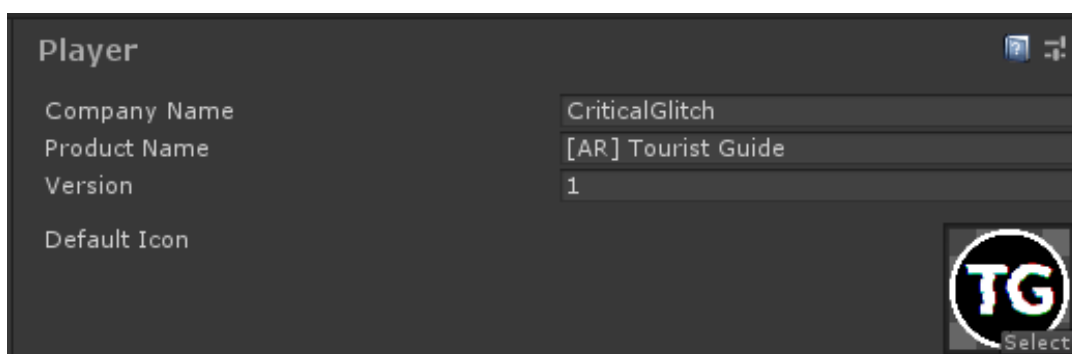
if (Application.internetReachability != NetworkReachability.NotReachable)
{
    if (Input.location.isEnabledByUser)
    {
        alertBox.SetActive(false);
    }
    else
    {
        alertBox.SetActive(true);
        alertTxt.text = "GPS ni vključen!";
    }
}
else
{
    alertBox.SetActive(true);
    alertTxt.text = "Ni povezave z internetom!";
}

```

Slika 6.12: Preverjanje povezljivosti z internetom in GPS

6.3 Objava mobilne aplikacije

Za objavo aplikacije smo se v programu Unity napotili pod nastavitve, ki se nahajajo v »File« → »Build Settings...«, kjer smo v okno dodali vse ustvarjene scene. V oknu za nastavitve igralca (angl. player settings) smo spremenili ikono aplikacije, ji določili ime, želeno ime podjetja in trenutno verzijo izdaje (slika 6.13). V našem primeru se je aplikacija imenovala »[AR] Tourist Guide«. Ob izgradnji projekta se nam je ustvarila datoteka APK, s katero smo aplikacijo naložili na poljubno napravo platforme Android.



Slika 6.13: Nastavitve imena, ikone in verzije mobilne aplikacije

7 REZULTATI

S pomočjo ustvarjenega spletnega obrazca, ki je prikazan na sliki 7.1, smo v podatkovno bazo shranili končni 3D model cerkve svetega Mihaela v Družmirju (slika 7.3). Te podatke smo z obrazcem na sliki 7.2 tudi urejali.

Ime

Ime objekta

Opis

Opis objekta

URL

Povezava

Zemljepisno mesto

Širina

Dolžina

Višina

Faktor skaliranja

1

Rotacija

0

Objekt [.obj]

Choose file No file chosen

Tekstura [.png]

Choose file No file chosen

Dodaj

Baza objektov:

#	Ime	Širina	Dolžina
8	Cerkev svetega Mihaela	46.397387	15.082225
9	Test	46.397448	15.08209

Slika 7.1: Obrazec za vstavljanje modela v podatkovno bazo

Ime

Cerkev svetega Mihaela

Opis

»Močno se je zatreslo, glavni zvonik se je podrl, desna stran cerkve pa je še stala. Dobro je bila zidana,« se rušenja cerkve sv. Mihaela v Družmirju spominja zbiratelj Zvone A. Čebul. Ob 40.

URL

https://www.delo.si/novice/slovenija/jezero-je-spet-tam-kjer-je-ze-bilo.html

Zemljepisno mesto

46.397387

15.082225

0

Faktor skaliranja

0.025

Rotacija

-90

Posodobi

Izbriši

Nazaj

Baza objektov:

#	Ime	Širina	Dolžina
8	Cerkev svetega Mihaela	46.397387	15.082225
9	Test	46.397448	15.08209

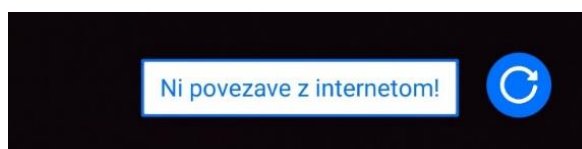
Slika 7.2: Obrazec za spreminjanje podatkov modela v podatkovni bazi



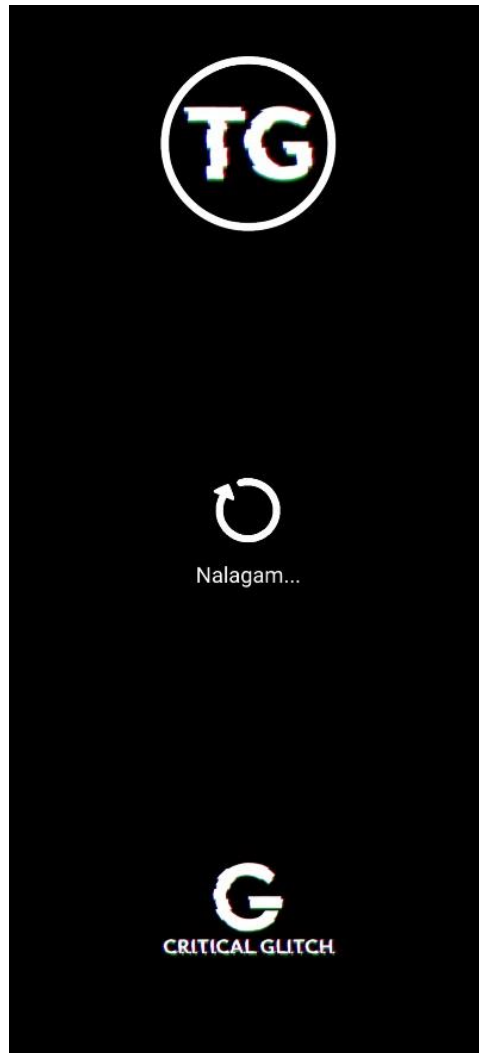
Slika 7.3: Končani 3D model cerkve svetega Mihaela v Družmirju

Za uporabo mobilne aplikacije mora ta biti združljiva z ARCore. To so naprave z Android vsaj 7.0 različico. ARCore lahko za delovanje zahteva določeno strojno opremo, kar je odvisno od same naprave ali ponudnika naprav.

Po odprtju mobilne aplikacije na napravi, se je najprej prikazal zagonski zaslon (slika 7.5), kjer nas je ob prvem zagonu tudi povprašala za dostop do vseh potrebnih pravic (kamera in GPS). Zagonski zaslon nas opozori tudi na to, če ni dostopa do internetne povezave ali pa je bil GPS v napravi izključen. Če je uporabnik izgubil povezljivost med delovanjem mobilne aplikacije, ga na to opozori z oknom, prikazanim na sliki 7.4.



Slika 7.4: Opozorilno okno v primeru izgube povezave z internetom



Slika 7.5: Zaslون ob zagonu mobilne aplikacije

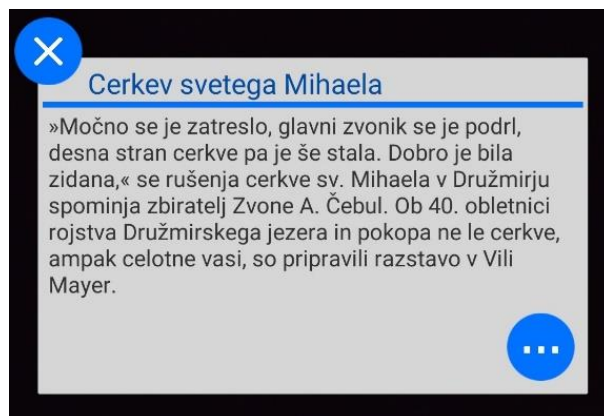
Ko aplikacija dobi dostop, se odpre glavni zaslon aplikacije, kjer so prikazani gumbi za osvežitve objektov, zajem slike zaslona ter prikaz vseh bližnjih objektov. V ozadju smo iz strežnika pridobivali podatke in jih skupaj s 3D modeli nalagali v aplikacijo. 3D model se je v dobrih razmerah postavil na določeno geografsko koordinato z natančnostjo od 2 do 5 metrov, v slabih pa od 10 do 20 metrov. Kar pomeni, da zaradi trenutnih omejitev GPS tega nismo mogli postaviti na točno geografsko koordinato. Hitrost pridobivanja in nalaganja podatkov iz strežnika je bila odvisna od naprave in povezljivosti do interneta. Pri testiranju se je v najboljših razmerah do 10 modelov cerkve svetega Mihaela naložilo v manj kot sekundi, v najslabšem pa je trajalo tudi do nekaj minut. Ker aplikacija ne

podpira večopravilnosti, je med čakanjem pridobivanje podatkov zaslon zamrznil, kar je vplivalo na slabšo uporabniško izkušnjo. Seveda pa je to odvisno tudi od hitrosti procesorja v napravi. Na sliki 7.6 lahko vidimo delovanje mobilne aplikacije, ki prikazuje naš model.



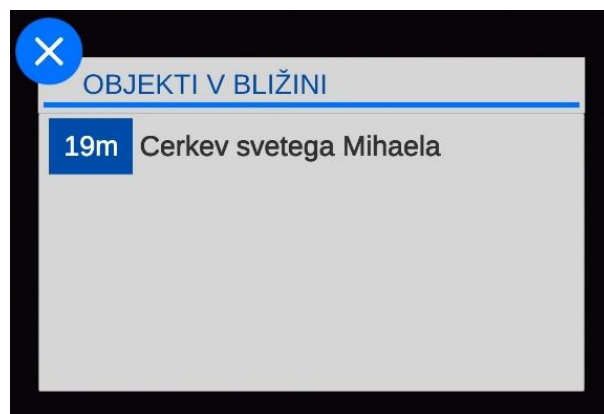
Slika 7.6: Delovanje mobilne aplikacije z modelom cerkve svetega Mihaela

Ob kliku na 3D model se nam je prikazalo ime objekta, njegov opis ter gumb s povezavo do več informacij (slika 7.7). Če smo kliknili na drug model, ki je v bližini, so se informacije v oknu spremenile.



Slika 7.7: Prikaz več informacij o objektu

Zajemali smo tudi sliko zaslona brez uporabniškega vmesnika, vendar se ta ni shranila v galerijo naprave, temveč v datoteko aplikacije. Za shranjevanje v galerijo bi potrebovali še dovoljenje do dostopa datotek. V prikazu vseh bližnjih objektov do tisoč metrov so se nam prikazovala imena in razdalja, kot prikazuje slika 7.8.



Slika 7.8: Prikaz vseh bližnjih objektov do tisoč metrov in njihovo razdaljo

8 ZAKLJUČEK

Menimo, da je cilj diplomske naloge dosežen. Mobilna aplikacija je odličen prikaz uporabe obogatene resničnosti v turizmu in lahko privabi turiste z novim zanimanjem. Dobra lastnost obogatene resničnosti je predvsem to, da je že dostopna skoraj vsem uporabnikom pametnih telefonov. Z nadgradnjami produkta in izboljšanjem samih tehnologij obogatene resničnosti bi s takšno mobilno aplikacijo lahko dosegli dober pripomoček pri obisku znamenitosti.

Možna izboljšava mobilne aplikacije je prikazovanje indikatorja poteka pri prenosu podatkov iz strežnika, tako da bi uporabnik vedel, kdaj se 3D model naloži. Prav tako bi lahko v aplikacijo dodali seznam, iz katerega bi uporabnik sam določil, katere podatke naj aplikacija pridobi, namesto da aplikacija naloži tudi modele, ki uporabnika ne zanimajo. Za boljši vpogled vseh turističnih znamenitosti, ki so na voljo, bi lahko v mobilno aplikacijo vgradili zemljevid, ki bi prikazoval, kje te so.

Za boljšo optimizacijo delovanja bi med samim posodabljanjem podatkov v mobilni aplikaciji lahko ohranili modele, ki so na novi lokaciji še vedno znotraj razdalje do tisoč metrov, namesto da izbrišemo vse in jih nato ponovno naložimo.

Dodali bi lahko tudi funkcije, kjer bi lahko uporabnik premikal model v mobilni aplikaciji in si ga tako bolje ogledal. Izboljšali bi lahko tudi uporabniški vmesnik za boljšo informativnost. Uporabili bi lahko tudi zvočne efekte, ki bi kot turistični vodniki pripovedovali o določenih objektih. Z zamenjavo formata OBJ s formatom FBX 3D modelov pa bi lahko v našo aplikacijo predstavili boljše teksturiranje in tudi animacije, vendar bi s tem ob večji velikosti datoteke tvegali še daljši čas nalaganja. Možnosti za nadgradnjo samega produkta ter tehnologije obogatene resničnosti je še veliko, zato lahko v tem področju vidimo zelo velik potencial tudi v prihodnosti.

VIRI IN LITERATURA

- [1] Merel, T. »The Reality of VR/AR growth«, 12. 1. 2017, dostopno na: <https://techcrunch.com/2017/01/11/the-reality-of-vrar-growth> [Dostopano: 16. 5. 2020]
- [2] Kovach, N. »What is Augmented Reality (AR) and How does it work«, 6. 2. 2018, dostopno na: <https://thinkmobiles.com/blog/what-is-augmented-reality/> [Dostopano: 16. 5. 2020]
- [3] Mordor Intelligence, »AUGMENTED REALITY MARKET SIZE - GROWTH, TRENDS, AND FORECASTS (2020 - 2025)«, dostopno na: <https://www.mordorintelligence.com/industry-reports/augmented-reality-market> [Dostopano: 10. 6. 2020]
- [4] World Around Me, dostopno na: https://lh3.googleusercontent.com/Zm0N5u0dHoNE_dCKyb6gBrWqQRgQK1da_zZSY5vBJ8h5aHNC8nldqvfcWijcxtkEbR24=w1920-h937-rw [Dostopano: 10. 6. 2020]
- [5] Augment - 3D Augmented Reality, dostopno na: <https://images.adsttc.com/media/images/5ca7/5ab9/284d/d1e4/3300/0252/slideshow/Augmented.jpg?1554471601> [Dostopano: 10. 6. 2020]
- [6] Senditur, dostopno na: https://lh3.googleusercontent.com/Yzz5ZWQ_Hti-fw5uwCn5n80xCzIO6NyJhduhOWBOvlqqV3LZ1_3YgiGeNIWvO_sjMv54=w1920-h888-rw [Dostopano: 10. 6. 2020]
- [7] Smartify, dostopno na: <https://lh3.googleusercontent.com/furYYUBYP7bcYVbaD8RwfJNChgDepsF43G4pf1Dkoi0HDDIXmwoPpiqg9tZHo0QGFQ=w1920-h888-rw> [Dostopano: 10. 6. 2020]
- [8] Logotip programa Unity, dostopno na: <https://unity3d.com/files/images/ogimg.jpg> [Dostopano: 10. 6. 2020]
- [9] Petty, J. »What is Unity & What is it Used For?«, dostopno na: <https://conceptartempire.com/what-is-unity/> [Dostopano: 17. 5. 2020]

- [10] Google, »ARCore overview«, 28. 2. 2019, dostopno na:
<https://developers.google.com/ar/discover> [Dostopano: 17. 5. 2020]
- [11] Unity, »About AR Foundation«, dostopno na:
<https://docs.unity3d.com/Packages/com.unity.xr.arfoundation@2.2/manual/index.html> [Dostopano: 17. 5. 2020]
- [12] Runtime OBJ Importer, 17. 8. 2020, dostopno na:
<https://assetstore.unity.com/packages/tools/modeling/runtime-obj-importer-49547> [Dostopano: 17. 8. 2020]
- [13] Unity AR + GPS Location, dostopno na: <https://docs.unity-ar-gps-location.com/>
[Dostopano: 18. 5. 2020]
- [14] Logotip programa Visual Studio, dostopno na: <https://venturebeat.com/wp-content/uploads/2019/11/visual-studio-logo.jpeg?w=1200&strip=all>
[Dostopano 10. 6. 2020]
- [15] Logotip programa Sublime Text 3, dostopno na:
<https://cdn.dribbble.com/users/533705/screenshots/3811091/sublime-icon.png> [Dostopano 10. 6. 2020]
- [16] Wikipedia, Sublime Text, 19. 5. 2020, dostopno na:
https://en.wikipedia.org/wiki/Sublime_Text [Dostopano: 18. 5. 2020]
- [17] WPBlogX, »What is XAMPP? And How to Install XAMPP on Local Computer?«, 10. 16. 2017, dostopno na: <https://www.wpblox.com/what-is-xampp/>
[Dostopano: 18. 5. 2020]
- [18] Kuralt, Š. »Jezero je spet tam, kjer je že bilo«, 6. 10. 2015, dostopno na:
<https://www.delo.si/novice/slovenija/jezero-je-spet-tam-kjer-je-ze-bilo.html>
[Dostopano: 19. 5. 2020]
- [19] Razglednica cerkve svetega Mihaela v Družmirju, dostopno na:
<https://museums.blob.core.windows.net/data/Documents/ARTWORKS/mv/38541/3f15ddc7d5b14d389916141118e3aa8a.jpg> [Dostopano: 21. 5. 2020]
- [20] Družmirje danes in nekoč, dostopno na:
<https://media.rfantasy.si/thumb600/eabdbb36-6c93-457e-bbfd-903959887e44.jpg> [Dostopano: 22. 5. 2020]

- [21] InnovatAR, »Understanding the Types of Augmented Reality«, 5. 6. 2019, dostopno na: <https://innovatar.io/types-augmented-reality/> [Dostopano: 20. 8. 2020]
- [22] The Franklin Institute, »What is Augmented Reality«, 18. 12. 2019, dostopno na: <https://www.fi.edu/what-is-augmented-reality> [Dostopano: 22. 8. 2020]