FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání bakalářské práce

Název:	Věnná města Českých Královen – Prototyp klienta v herním enginu Unreal Engine 4
Student:	Hanna Korniushyna
Vedoucí:	Ing. Jiří Chludil
Studijní program:	Informatika
Obor / specializace:	Webové a softwarové inženýrství, zaměření Počítačová grafika
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	do konce letního semestru 2022/2023

Pokyny pro vypracování

Věnná města českých královen je projekt zabývající se zobrazením historického prostředí ve virtuální a rozšíření realitě.

1. Analyzujte podobné závěrečné práce, které se věnují danému tématu.
2. Analyzujte prostředí a nástroje pro vývoj ve virtuální realitě pod Unreal Engine 4.
3. Analyzujte funkční a nefunkční požadavky zadavatele.
4. Pomocí metod softwarového inženýrství navrhněte zásuvný modul pro generování budov na základě mapových podkladů.
5. Pomocí metod softwarového inženýrství navrhněte klienta pro zobrazení historické scény na základě dat z API.
6. Implementujte prototyp zásuvného modulu a klienta VR.
7. Výsledný prototyp podrobte vhodným testům (uživatelské, akceptační a integrační).

Elektronicky schválil/a Ing. Radek Richtr, Ph.D. dne 12. ledna 2022 v Praze.

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalářská práce

Věnná města českých královen – Prototyp klienta v herním enginu Unreal Engine 4

Hanna Korniushyna

Katedra softwarového inženýrství
Vedoucí práce: Ing. Jiří Chludil

11. května 2022

Poděkování

Chtěla bych poděkovat vedoucímu mé práce Ing. Jiří Chludilovi za odborné vedení, přátelský přístup a přínosné konzultace. Dále děkuji své rodině a přátelům za podporu během celého mého studia.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 2373 odst. 2 zákona č. 89/2012 Sb., občanský zákoník, ve znění pozdějších předpisů, tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou a veškeré jejich dokumentace (dále souhrnně jen „Dílo"), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu) licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 11. května 2022 .....

České vysoké učení technické v Praze
Fakulta informačních technologií

Odkaz na tuto práci

Korniushyna, Hanna. *Věnná města českých královen – Prototyp klienta v herním enginu Unreal Engine 4*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2022.

Abstrakt

Tato bakalářská práce se zabývá analýzou, návrhem a implementací historické scény ve virtuální realitě na základě dat z datového úložiště. Dalším z cílů této práce je modifikace zásuvného modulu pro generování budov v grafickém editoru Blender. Generované budovy poté budou použity pro tvoření kompletní scény ve VR aplikaci.

Aplikace je součástí projektu Věnná města českých královen, proto analýza obsahuje též rozbor podobných závěrečných prací, které jsou věnovány danému projektu a které sloužily jako inspirace pro tuto práci nebo budou použity ve výstupu. Bylo využito privátní API projektu VMČK, které však pro dosažení cíle této práce vyžadovalo několik úprav, proto bylo nutné zahrnout do analýzy popis současného stavu a návrh změn. V závěru analýzy byly sestaveny funkční a nefunkční požadavky pro zásuvný modul a VR klienta.

Návrh je zaměřen na průzkum dvou zásuvných modulů, kde byl vybrán vhodnější z těchto dvou modulů, a byly navrženy modifikace. Návrh VR klienta zahrnuje popis vybraných nastavení a použitých technologií.

Výstup bakalářské práce je implementován na základě navržených požadavků a obsahuje instalační, uživatelskou a programátorskou příručku. Pro uživatelské testování VR klienta byly vytvořeny vhodné podkladové materiály.

Klíčová slova virtuální realita, Věnná města českých královen, historické prostředí, Unreal Engine 4, generování budov, zásuvný modul

Abstract

This bachelor thesis focuses on the analysis, design and implementation of the historical scene in virtual reality based on data from the data warehouse. Another goal of this work is to modify the plug-in for generating buildings in the graphic editor Blender. The generated buildings will then be used to create a complete scene in the VR application.

The application is part of the project Dowry Cities of Czech Queens, so the analysis also includes an analysis of similar final theses, which are related to the project and which served as inspiration for this work or will be used in the output. The private API of the Dowry Cities of Czech Queens project was used, which, however, required several modifications to achieve the goal of this work, so it was necessary to include in the analysis a description of the current state and a proposal for changes. At the end of the analysis, functional and non-functional requirements for the plug-in module and the VR client were defined.

The design is focused on the survey of two plug-ins, where the more suitable of these two modules was selected, and modifications were proposed. The VR client design includes a description of selected settings and used technologies.

The output of the bachelor thesis is implemented on the basis of the defined requirements and contains an installation, user and programming manual. Suitable testing materials were created for user testing of the VR client.

Keywords virtual reality, Dowry Cities of Czech Queens, historical environment, Unreal Engine 4, buildings generation, plug-in

Obsah

Úvod **1**

1 Cíle práce **3**

2 Analýza **5**
2.1 Analýza závěrečných prací, které se věnují tomuto tématu . . . 5
2.1.1 Virtuální realita . . . 6
2.1.1.1 VMČK - Úprava 3D modelu ve virtuální realitě 6
2.1.1.2 Interakce v historickém městě ve virtuální realitě 7
2.1.1.3 Věnná města českých královen – Schvalovací systém pro 3D modely ve virtuální realitě . . . 8
2.1.2 Příprava modelu . . . 9
2.1.2.1 Digitalizace historického oblečení . . . 9
2.1.3 Infrastruktura . . . 10
2.1.3.1 Věnná města českých královen – Backend administrační části . . . 10
2.1.4 Shrnutí analýzy . . . 10
2.2 Analýza současného stavu API a návrh nutných změn . . . 11
2.2.1 Analýza současného stavu . . . 12
2.2.2 Návrh nutných změn . . . 15
2.3 Vývoj virtuální reality . . . 17
2.3.1 Vývoj v Unity 3D . . . 17
2.3.2 Vývoj v Unreal Engine 4 . . . 19
2.3.3 Vybraný software . . . 20
2.4 Blender a vytváření zásuvných modulů . . . 22
2.4.1 Rozšíření aplikace Blender . . . 22
2.4.2 Princip a požadavky . . . 23
2.5 Funkční a nefunkční požadavky . . . 24
2.5.1 Požadavky pro zásuvný modul . . . 24

2.5.2 Požadavky pro VR klienta 25

3 Návrh **27**
3.1 Návrh zásuvného modulu do Blenderu 27
3.1.1 Blender OSM 27
3.1.1.1 Princip fungování 28
3.1.2 Blender GIS 29
3.1.2.1 Princip fungování 29
3.1.3 Vybrané řešení 30
3.1.4 Návrh modifikace vybraného doplňku 32
3.1.4.1 Princip fungování Blender OSM 32
3.1.4.2 Přidání možnosti vybíraní oblasti z API 33
3.1.4.3 Schéma komunikace s API 33
3.1.4.4 Skrytí nepoužité funkcionality 35
3.2 Návrh VR klienta 35
3.2.1 Nastavení projektu 36
3.2.2 Použití VR šablony 36
3.2.3 Návrh uživatelského rozhraní 37
3.2.4 Schéma komunikace s API 40
3.2.5 Zásuvný modul VaRest 40
3.2.6 Zásuvný modul RuntimeMeshLoader 40
3.2.7 Zásuvný modul Web Browser 40
3.3 Závěr návrhu 41

4 Realizace **43**
4.1 Zásuvný modul Blender 43
4.1.1 Instalační příručka 43
4.1.2 Uživatelská příručka 43
4.1.3 Programátorská příručka 45
4.1.3.1 Verzování a komentování 45
4.1.3.2 Uložení dat všech oblasti 45
4.1.3.3 Odstranění historických budov 46
4.1.3.4 Vzniklé problémy 47
4.2 VR klient 48
4.2.1 Instalační příručka 48
4.2.2 Uživatelská příručka 48
4.2.3 Programátorská příručka 51
4.2.3.1 Modifikace RunTimeMeshLoader 51
4.2.3.2 Komunikace s datovým úložištěm 53
4.2.3.3 Převod světových souřadnic 54
4.2.3.4 Uživatelské rozhraní 54
4.2.3.5 Verzování 55

5 Testování **57**

5.1 Dotazníky 57
5.2 Testovací scénář 58
5.2.1 Orientace v prostředí 58
5.2.2 Interakce s uživatelským rozhraním 59

Závěr **61**

Literatura **63**

A Seznam použitých zkratek **67**

B Obsah přiloženého CD **69**

Seznam obrázků

2.1 Mapa věnných měst 5
2.2 Výstup práce Patrika Křepinského 7
2.3 Scéna z prototypu práce Kláry Matouškové 8
2.4 Ukázka aplikace Marka Klofáče 9
2.5 Ukázka výsledku práce Aleny Žižkové 10
2.6 Datový model Daniela Vančury 13
2.7 Datový model Dominika Siváka 14
2.8 Návrh nového datového modelu 16
2.9 Aktuální struktura zásuvného modulu Unity XR 18
2.10 Ukázka VR režimu Unreal Engine 20
2.11 Požadavky pro zásuvný modul 24
2.12 Požadavky pro VR klienta 25

3.1 Prémiová verze Blender OSM 28
3.2 UI Blender OSM 29
3.3 UI Blender GIS 30
3.4 Ukázka fungování zásuvného modulu Blender GIS 31
3.5 Ukázka fungování zásuvného modulu Blender OSM 31
3.6 Flowchart diagram komunikace s API VMČK 34
3.7 Ukázka návrhu uvítacího UI 37
3.8 Ukázka návrhu UI s výběrem oblasti 38
3.9 Ukázka návrhu UI s upozorněním 38
3.10 Ukázka návrhu UI s mapou 39
3.11 Ukázka návrhu UI s informace o budově 39
3.12 Vývojový diagram projektu 41

4.1 Nové uživatelské rozhraní Blender OSM 44
4.2 Generovaná Testovací zóna 1 45
4.3 Ovladači HTC Vive 49
4.4 Uvítací scéna 49

4.5 Ukázka teleportu 50
4.6 Ukázka mapy prostředí 50
4.7 Ukázka informace o budově 51
4.8 Část funkce GenerateModels() 52
4.9 Funkce MakeRequest() 53
4.10 Ukázka použití funkce MakeRequest() 53
4.11 Funkce LonToX() 54

Seznam tabulek

3.1 Popis metod knihovny Requests 35

Úvod

V dnešní době se často setkáváme s pojmem virtuální realita a každým rokem tato technologie získává více zájemců. Pomocí této technologie je vyvíjeno mnoho různých projektů. Většinou jsou určeny pro herní průmysl, ale existuje mnoho příkladů využití virtuální reality v lékařství, ve sportu, konstrukcích a vzdělávání. Jeden z těchto vzdělávacích projektů se nazývá Věnná města českých královen. Projekt je vytvořen ve spolupráci FIT ČVUT (Fakulta informačních technologií Českého vysokého učení technického v Praze), FF UHK (Filozofická fakulta Univerzity v Hradci Králové), Ministerstva kultury a dalších institucí. Hlavním cílem projektu je zobrazeni historického prostředí několika věnných měst a interakce s ním. Součástí projektu je mobilní klient s technologií rozšířené reality, webový portál a aplikace virtuální reality.

Tato práce je věnována části související s virtuální realitou. Práce se navazuje na diplomovou práci Dominika Siváka, který vytvořil backend pro modely historických budov, a je inspirována dalšími pracemi, které jsou věnovány tomuto tématu. Výsledek bude použit hlavně pro širokou veřejnost, která má zájem o historii věnných měst, a také pro historické pracovníky a studenty.

V rámci předmětu softwarový týmový projekt už jsem měla příležitost pracovat s projektem VMČK. Tento projekt byl zaměřen na virtuální realitu a Unreal Engine 4. Projekt a technologie virtuální reality mě zaujaly, a proto jsem se rozhodla pokračovat tímto směrem a použít získané znalosti ve své bakalářské práci.

Kapitola 1

Cíle práce

Hlavním cílem této práce je ve virtuální realitě navrhnout klienta pro zobrazení historické scény na základě dat z API. Součástí tohoto cíle je prozkoumání podobných závěrečných prací a analyzování nástroje pro vývoj ve virtuální realitě. Zaměřím se na herní enginy Unreal Engine 4 a Unity 3D a porovnám jejich možnosti vývoje virtuální reality. Potom budou navrženy funkční a nefunkční požadavky pro výstup této práce.

Vedlejším, ale stejně důležitým cílem je návrh modifikace zásuvného modulu pro generování budov na základě mapových podkladů. Zaměřím se na grafický software Blender 2.83, konkrétně na analýzu různých možností rozšíření funkcionality Blenderu, a porovnám již existující moduly pro generování budov. Z nich bude vybrán vhodnější modul a navrženy nutné modifikace.

V praktické části bude implementován prototyp VR aplikace, který bude využívat modifikovaný zásuvný modul a již existující privátní API. Výstup bude součástí projektu Věnná města českých královen. K VR aplikaci a zásuvnému modulu budou poskytnuty instalační, uživatelská a programátorská příručka.

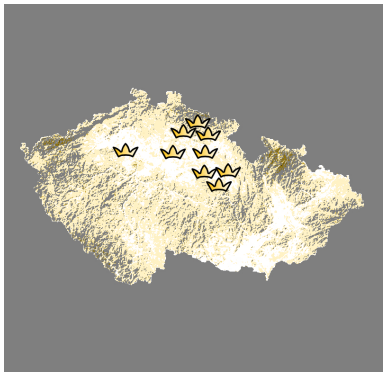
Pro výsledný prototyp aplikace budou provedena uživatelská, akceptační a integrační testování.

Kapitola 2

Analýza

2.1 Analýza závěrečných prací, které se věnují tomuto tématu

Věnná města českých královen jsou města, která byla zdrojem příjmu a důchodu českých královen od začátku 14. století až po 19. století. K těmto městům patří například Hradec Králové, Chrudim, Vysoké Mýto, Polička, Jaroměř, Dvůr Králové nad Labem, Trutnov, Nový Bydžov či Mělník. Tato města jsou znázorněna na obrázku 2.1.



Obrázek 2.1: Mapa věnných měst [1]

Pro prezentaci tohoto výhradně českého dějinného fenoménu vznikl projekt VMČK na Fakultě informačních technologií ČVUT ve spolupráci s Univerzitou v Hradci Králové, Ministerstvem kultury a dalšími institucemi. Hlavními výstupy projektu jsou mobilní aplikace, webový portál a VR aplikace. Ty budou sloužit jako historický průvodce věnnými městy a jejich městskou krajinou včetně 3D rekonstrukce již zaniklých, ale velmi historicky důležitých městských areálů a budov [1].

Projekt Věnná města českých královen je vyvíjen na fakultě už několik let, proto existuje řada bakalářských a diplomových prací, které se věnují tomuto tématu. Tato část práce se zabývá analýzou závěrečných prací, které se týkají projektu VMČK.

Analyzované práce můžeme rozdělit na tři podsekce. První podsekcí jsou závěrečné práce, které se věnují virtuální realitě, jsou to práce Patrika Křepinského, Marka Klofáče a Kláry Matouškové. Tyto práce budou sloužit jako podklady a inspirace pro tuto bakalářskou práci. Do další podsekce patří práce Aleny Žižkové, která připravila materiály, textury a oblečení pro NPC postavy, které budou využity v této práci. Poslední práce popisuje infrastrukturu, která bude upravena a použita v této bakalářské práci. Jde o diplomovou práci Dominika Siváka, jež popisuje a implementuje API pro projekt VMČK.

2.1.1 Virtuální realita

2.1.1.1 VMČK - Úprava 3D modelu ve virtuální realitě

Práce Patrika Křepinského je zaměřena na vytváření rozsáhlých scén ve virtuální realitě, vizualizaci, manipulaci a úpravu 3D modelů. Součástí práce je charakteristika VR brýlí HTC Vive a HTC Vive Pro a analýza technik a manipulace 3D objektů v počítačové grafice. Výstupem je prototyp v herním enginu Unreal Engine 4, kde uživatel může upravovat město a přidávat nové objekty. Když uživatel město opustí nebo vypne aplikaci, veškeré provedené změny se automaticky nahrají na server [2]. Na obrázku 2.2 je znázorněn výstup práce.

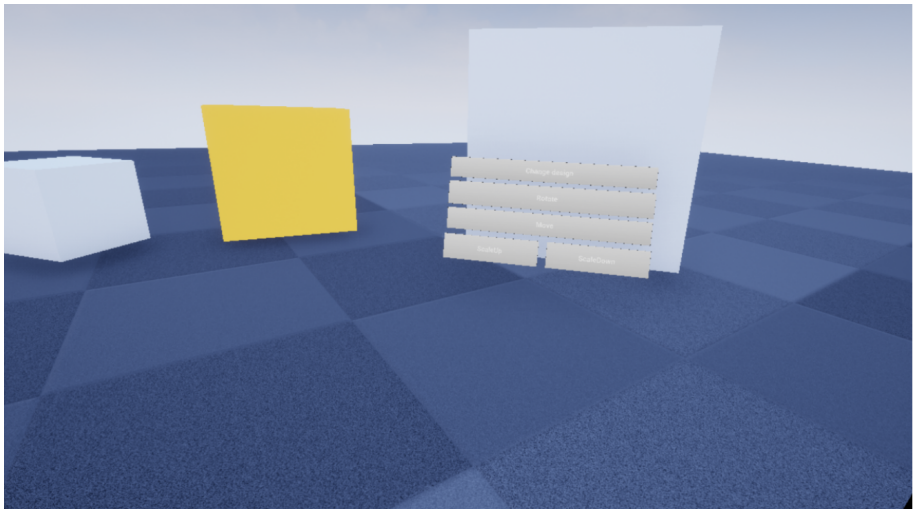
Výhodami práce jsou:

- interakce s objekty,
- upravování města,
- prohlížení města pomocí teleportu,
- uložení změn.

Nevýhody jsou následující:

- nerealistické prostředí, v době vytvoření práce neexistovalo API s vhodnými 3D modely,
- částečná funkcionalita,
- neexistence návodu, jak se aplikace používá,
- špatně navržené rozhraní z hlediska uživatelské přívětivosti,
- skutečnost, že neproběhlo testování na reálných uživatelích.

V závěru lze konstatovat, že řada funkčních požadavků nebyla v rámci bakalářské práce splněna (například různé možnosti prohlížení města, práva uživatele, správa modelu apod.), ale prototyp je součástí velkého projektu, proto na vývoji aplikace dále pracovali další studenti. Na vývoji se podílelo



Obrázek 2.2: Ukázka výstupu práce Patrika Křepinského [2]

několik generací studentů a v současné době je projekt pozastaven, takže ho nemohou využívat ani historici, ani široká veřejnost.

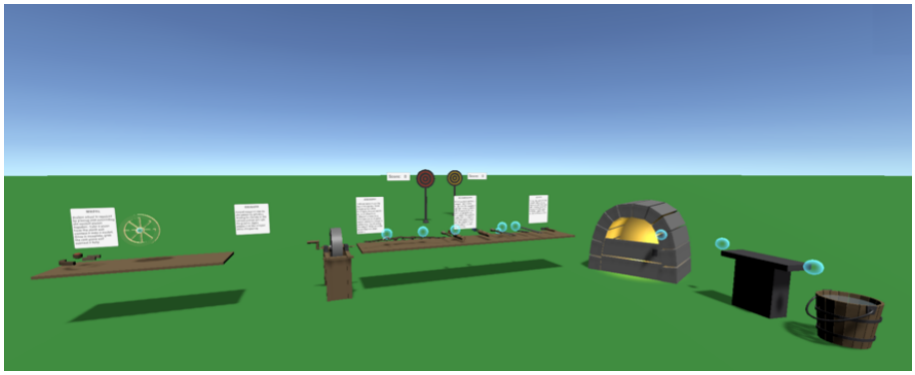
2.1.1.2 Interakce v historickém městě ve virtuální realitě

Druhá analyzovaná práce je bakalářská práce Kláry Matouškové, která se zaměřuje na demonstraci interakce ve virtuální realitě. V práci je analyzována virtuální realita a její zařízení, základní metody interakce a možnosti tvorby VR aplikace v herním enginu Unity 3D, podobné již existující interaktivní aplikaci. Výstupem bakalářské práce je jednoduchá scéna ve virtuální realitě, která obsahuje pět navržených interakcí: lukostřelbu, kování, broušení, opravu kola u vozíku a pochodně. V práci byl použit balíček Unity XR Interaction Toolkit, který umožňuje multiplatformní tvorbu VR a AR aplikace [3]. Na obrázku 2.3 je zobrazena hlavní scéna výsledné práce.

Kladné stránky práce jsou:

- pravděpodobná interakce s různými objekty,
- jednoduché uživatelské rozhraní,
- srozumitelné ovládání,
- přemístění pomocí teleportu,
- další implementované interakce, které nebyly v návrhu (pouzdra, otevírací dveře, zapálení dalších objektů),
- realistický vzhled materiálů (kov, dřevo apod.).

Nebylo zaznamenáno žádné negativní hodnocení této bakalářské práce a celkově lze ohodnotit práci jako velmi povedenou. V budoucnu tato práce může



Obrázek 2.3: Scéna z prototypu práce Kláry Matouškové [3]

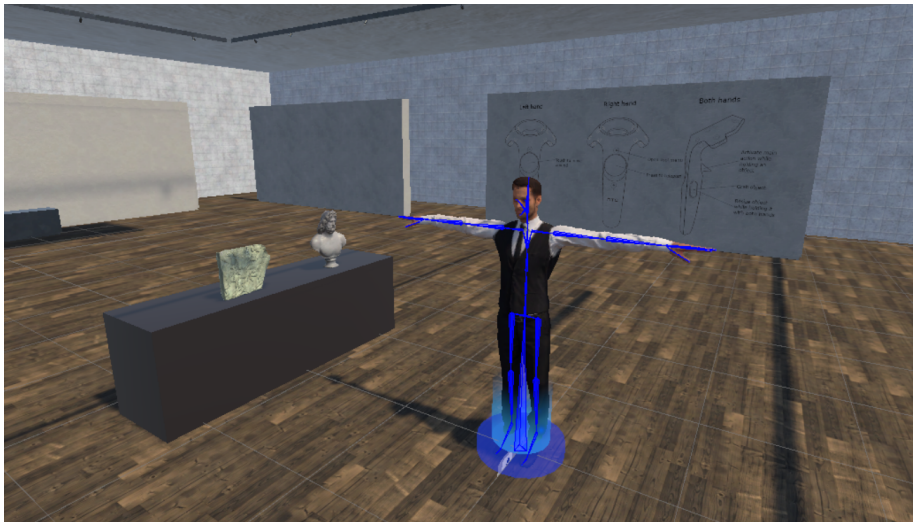
být použita v dalších větších projektech pro VMČK jako jedna z možností interakce. Prototyp používá herní engin Unity 3D, proto nebude použit v této bakalářské práci a sloužil pouze jako inspirace.

2.1.1.3 Věnná města českých královen – Schvalovací systém pro 3D modely ve virtuální realitě

Marek Klofáč se ve své bakalářské práci zabývá vývojem aplikace virtuálního muzea, která slouží jako schvalovací proces 3D modelů. Historik může navrhnout 3D model a poskytnout podklady, modelář poté vytvoří model podle návrhu a následně historik a grafik vyhodnotí hotový model. Tato aplikace umožňuje prozkoumání a schválení modelu ve virtuální realitě. V aplikaci jsou dostupné různé interakce, například škálování modelu nebo možnost zobrazení drátového modelu, což je velmi důležité pro grafiky. Implementace aplikace je vytvořena v herním enginu Unity 3D s použitím vhodných balíčků jako XR Interaction Toolkit pro tvorbu VR aplikace. Součástí práce je analýza backendu projektu VMČK a vývojového prostředí Unity [4]. Na obrázku 2.4 je zobrazeno virtuální prostředí aplikace s avatarem, který reprezentuje uživatele.

Celkově tuto práci můžeme považovat za velmi povedenou, má atraktivní uživatelské rozhraní, splňuje všechny funkční a nefunkční požadavky. K tomu byla testována na fakultě v UsabilityLabu, testování se zúčastnilo devět osob. Několik z nich muselo přerušit testování kvůli zdravotním potížím, důvodem byla nejspíše nezkušenost testerů s VR. Většina hodnotila práci kladně, někteří měli problém s ovládáním uživatelského rozhraní a nalezením tlačítek na ovladači.

Pro tuto bakalářskou práci sloužila práce Marka Klofáče hlavně jako inspirace pro tvorbu uživatelského rozhraní a vizuálního konceptu. Práce také analyzuje současný stav API projektu, stejný úkol má i tato bakalářská práce, takže text posloužil jako studijní materiál pro tvorbu analýzy.



Obrázek 2.4: Ukázka aplikace Marka Klofáče [4]

2.1.2 Příprava modelu

2.1.2.1 Digitalizace historického oblečení

Velmi důležitou částí projektu VMČK je příprava historicky pravděpodobných modelů a budov. Existuje řada závěrečných prací, které jsou věnovány analýze správnosti 3D objektů a návrhu zásuvných modulů pro zlepšení a zrychleni modelování. Jedna z takových prací bude použita i v této bakalářské práci, jde o práci Aleny Žižkové, která je zaměřena na digitalizaci historického oblečení, kterým budou oděny generované postavy.

Výstupem je teoretická část, kde je velmi detailně popsána historie oblečení od renesance do secese a analyzovány nástroje pro úpravu 3D modelů, a praktická část, kde jsou v praxi demonstrovány dvě metody digitalizace historického oblečení: 3D modelování a 3D skenování. Některé části metod digitalizace jsou automatizovány pomocí zásuvných modulů do grafického editoru Blender [5]. Moduly byly podrobeny uživatelskému testování a zaznamenaly velkou spokojenost uživatelů, kteří jsou seznámeni s grafickými nástroji. Na obrázku 2.5 jsou zobrazeny generované postavy, které mají na sobě oblečení vytvořené pomocí Alenou Žižkovou demonstrovaných automatizovaných metod digitalizace historického oblečení.

Tyto postavy byly po odevzdání bakalářské práce animovány a nyní slouží jako NPC postavy pro projekt VMČK. Budou použity ve výstupu této práce pro zlepšení uživatelské zkušenosti a vytvoření pravděpodobnosti. Postavy umí chodit na jednom místě a dělat pohyby rukama, ale zatím se neumějí pohybovat po určitém zadaném vektoru, proto ve VR klientu bude třeba nastavit jejich trajektorie a směr pohybu.

Obrázek 2.5: Generované postavy v oblečení od Aleny Žižkové

2.1.3 Infrastruktura

2.1.3.1 Věnná města českých královen – Backend administrační části

Hlavním cílem diplomové práce Dominika Siváka je tvorba schvalovacího procesu pro backend projektu VMČK. Práce navazovala na jádro, které bylo vytvořeno jako praktická část bakalářské práce Daniela Vančury [6] a diplomové práce Michala Martinka [7]. Jádro používá technologie Docker, Node.js, TypeScript, PostgreSQL apod. Daniel Sivák analyzuje stav jádra při převzetí, navrhuje, realizuje a testuje zlepšení schvalovacího procesu [8]. Detailněji se touto prací bude zabývat další podkapitola.

Výsledkem této diplomové práce je funkční a řádně testované API, které má velmi dobrou logickou strukturu. Tato práce výrazně posunula cely projekt a je dosud využívána v dalších bakalářských a diplomových pracích, které se týkají projektu VMČK. Další část práce, která se bude věnovat API, bude navazovat na tuto diplomovou práci a budou navrženy změny, které jsou potřebné pro návrh a implementaci této bakalářské práce.

2.1.4 Shrnutí analýzy

Na závěr lze říci, že na fakultě FIT je k dispozici celá řada projektů, které se věnují právě projektu VMČK. Každý z nich znamená velký přínos k vývoji projektu a každý pomohl při vytváření této bakalářské práce. Závěrečné práce, které souvisejí s virtuální realitou, posloužily jako inspirace uživatelského roz-

hraní, použitých technologií a pomohly v sepsání všech nutných funkčních a nefunkčních požadavků pro tuto bakalářskou práci. Jejich teoretické částí byly pečlivě nastudovány a použity v analýze a návrhu této bakalářské práce.

Velká část závěrečných prací, které se týkají projektu Věnná města českých královen, zabývá se tvorbou a úpravou modelů pro tento projekt. V této bakalářské práci budou použity NPC postavy v historickém oblečení, což je výsledkem bakalářské práce Aleny Žižkové.

Důležitým přínosem k projektu VMČK jsou závěrečné práce, které jsou zaměřeny na tvorbu backendu projektu. Aktuální API je výsledkem diplomové práce Dominika Siváka, který spojil všechna díla předchůdců dohromady a vytvořil kompletní funkční backend. Po nutných úpravách bude použit i v této bakalářské práci jako privátní API, ze kterého se budou načítat všechny informace a data pro modifikovaný zásuvný modul a VR klienta.

Hlavním úkolem této bakalářské práce je spojení všech vytvořených prací a myšlenek v nich uvedených pro vytvoření kompletního a funkčního VR klienta, který odpovídá všem funkčním a nefunkčním požadavkům celého projektu a který bude moci využívat i široká veřejnost. Tento VR klient též poslouží jako základ finální aplikace projektu VMČK.

2.2 Analýza současného stavu API a návrh nutných změn

API (rozhraní pro programování aplikace) je univerzální rozhraní pro vzájemnou komunikaci aplikací. V praxi obsahuje sbírku tříd, metod, funkcí a protokolů, díky kterým může jedna aplikace komunikovat s jinou, vzájemně synchronizovat uživatelská či systémová data. Ve webovém prostředí se pro komunikaci webového řešení se vzdáleným API používá standardní komunikační protokol HTTP/S, přes který dále obě strany většinou komunikují v platformě nezávislými formáty typu XML, CSV či JSON. Uvedené formáty posílají data ve standardizovaném zápisu a obě dvě strany jsou schopny tyto zápisy číst (parsovat) a vytvářet (generovat) [9]. API zjednodušuje vývoj softwaru, umožňuje rychle a bezpečně předávat data, funkce a urychluje programátorovi práci nad vývojem projektu. Téměř všechny aplikace používají veřejné nebo vlastni API. Potkáváme se s nimi každodenně v našem životě, například když uživatel vyhledává informace v Google [10]. Jeho požadavek se odesílá na server Google a poté, když prohlížeč obdrží odpověď, překládá ji a zobrazuje v prohlížeči uživatele. Pro efektivnější a interaktivnější výsledek bude tato bakalářská práce také používat privátní API vytvořené speciálně pro projekt VMČK.

Tato podkapitola je zaměřena na analýzu jeho struktury a současného stavu, který je výsledkem diplomové práce Dominika Siváka [8]. Ačkoliv práce je na vysoké úrovni a značně posunula projekt, neobsahuje některé struktury a objekty, které budou využity v této bakalářské práci. Proto podkapitola také

zahrnuje návrh změn na API, které byly konzultovány s vedoucím práce Ing. Jiří Chludilem a následně jím provedeny. Navíc návrh nebere v úvahu změny, které jsou nutné pro mobilní AR aplikace, jež je vytvářena paralelně s touto prací.

2.2.1 Analýza současného stavu

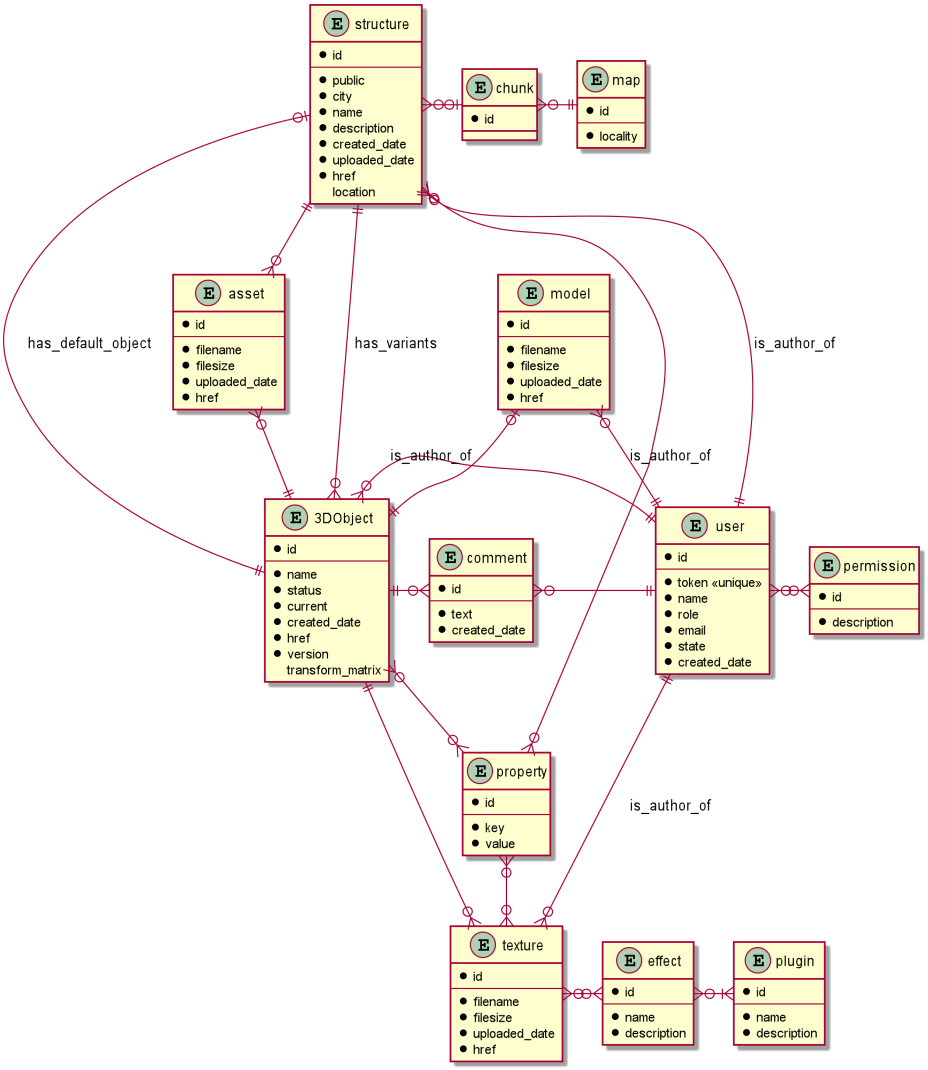
Projekt Věnná města českých královen je na fakultě vytvářen už několik let, proto existuje několik výstupů klientských aplikací a postupně je implementován backend. Práce Dominika Siváka je poslední funkční implementace API, přímo navazuje na bakalářskou práci Daniela Vančury a diplomovou práci Michala Martinka a je zaměřena na tvorbu schvalovacího procesu. Také bylo opraveno a doladěno několik chyb předchozí verze.

Použité technologie se neměnily od začátku implementace, Dominik Sivák uvádí, ze všechny technologie už mají pevné a neměnné místo v projektu, protože je na nich celý projekt postaven a podílel se už na něm nemalý počet lidí. Použité technologie jsou následující: **Docker, Node.js, Express.js, Typescript, TypeORM, JSON Web Token, PostgreSQL, MongoDB a Jest**. V této bakalářské práci nebude každá technologie popisována zvlášť, podrobný popis lze najít v pracích Daniela Vančury [6] a Dominika Siváka [8].

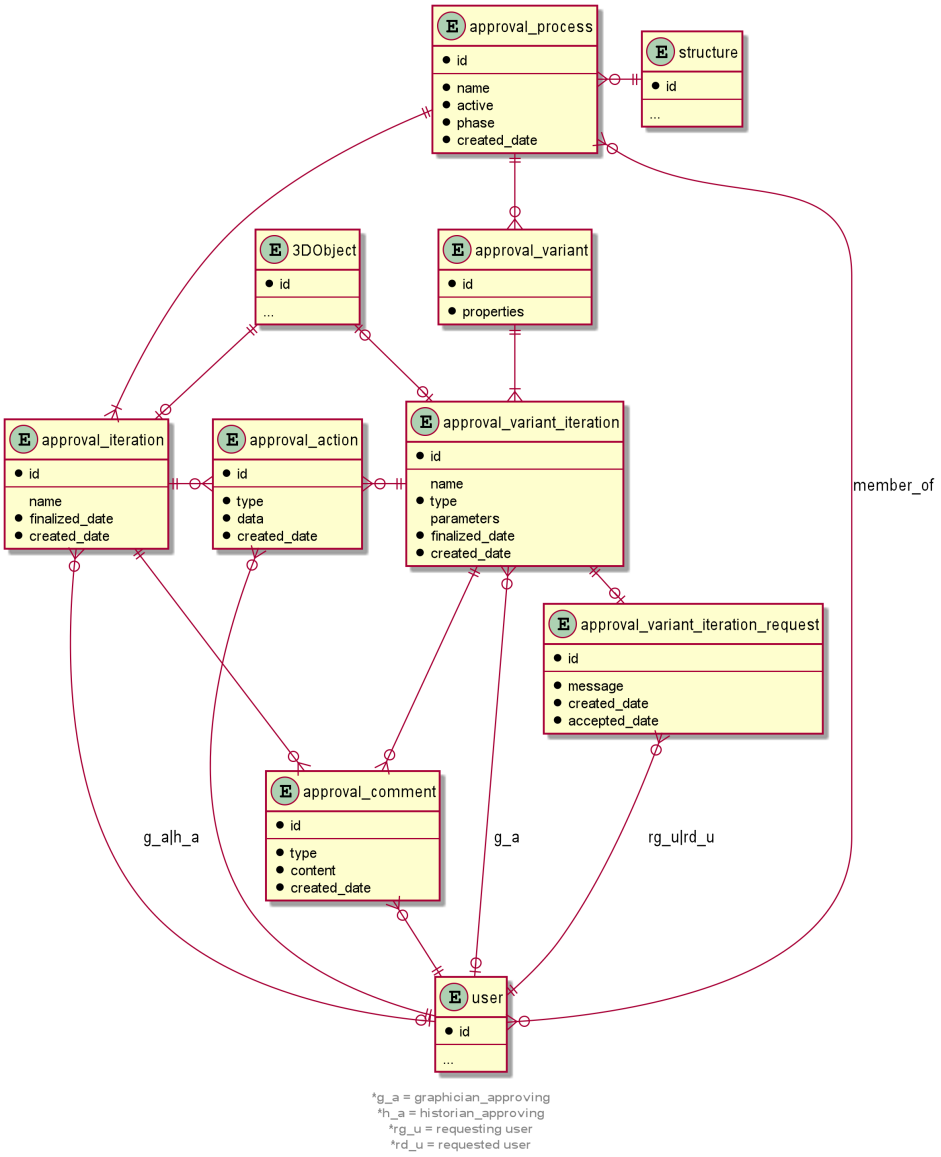
Původně mělo API sloužit jako webová databáze návrhů a hotových modelů různých historických budov. Má několik uživatelských rolí, každá z nich má unikátní funkce. Role **„historik“** vytváří zadání a podklady pro modelování budov a potom schvaluje správnost hotové budovy z historického pohledu. Role **„modelář“** vytváří grafické modely na základě zadání a role **„grafik“** je schvaluje z pohledu grafické správnosti. Role **„administrátor“** má nejvíce práv, může zakládat a spravovat uživatelské účty, nastavovat uživatelské role, mazat struktury a modely, rušit schvalovací proces. Poslední uživatelskou roli je běžný **„uživatel“**, který se může se přihlašovat a odhlašovat, prohlížet modely a měnit nastaveni profilu.

Současný stav odpovídá výsledku diplomové práce Dominika Siváka. Na obrázku 2.6 je znázorněn návrh datového modelu uložení dat od Daniela Vančury, který byl rozšířen a vylepšen Dominikem Sivákem. Byl vytvořen nový datový model schvalovacího procesu (obrázek 2.7), entita 3DObject byla oddělena od uživatele (entita user), dále byla přidána entita pro ukládání historie procesu schválení a statistika pro každý schvalovací proces.

Současný stav API umožňuje uživateli kompletní CRUD (Create, Read, Update, Delete) 3D objektu, verzování 3D objektů a ukládání jejich různých podob či filtrování modelů podle autora.



Obrázek 2.6: Datový model z bakalářské práce Daniela Vančury [6]



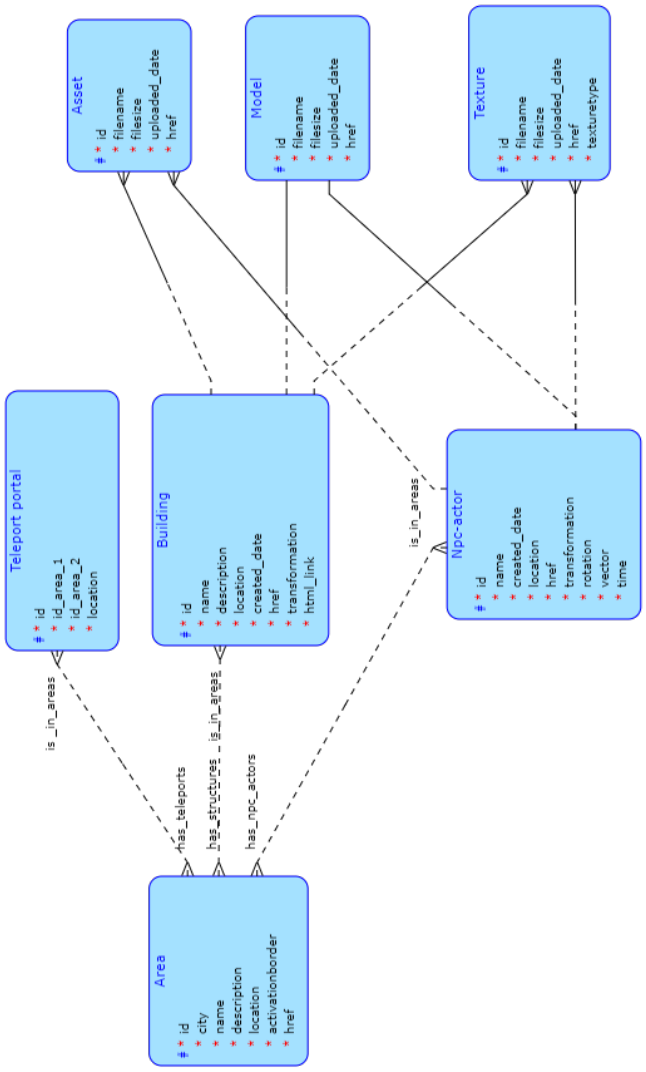
Obrázek 2.7: Datový model schvalovacího procesu z diplomové práce Dominika Siváka [8]

2.2.2 Návrh nutných změn

V současné době se účel API kompletně změnil. Nyní se používá pro tvorbu AR aplikací pro Android, webové stránky, která umožňuje prohlížení 3D modelů, a VR aplikace, která je výsledkem teto práce. Pro implementování VR aplikace s použitím tohoto API je třeba udělat několik změn:

- **Skrytí schvalovacího procesu** Skrytí všech entit, které souvisejí se schvalovacím procesem, a upravení entit, tzn. odstranění nepotřebných atributů, které odpovídají informacím o schvalovacím procesu a o procesu vytváření a upravování modelu.
- **Úprava entit Structure, Asset, 3DObject, Model a Texture** Entita Structure bude přejmenována na Building a bude odpovídat jedné budově a jejím vlastnostem. K vlastnostem patří ID, název, popis, datum poslední modifikace, lokace budovy ve světových souřadnicích, transformace, odkaz na webovou stránku s informacemi o budově a odkazy na entity modelu, textur a assetu, které patří vybrané struktuře. Atributy entity 3DObject budou posunuty do entity Building a sama entita bude odstraněna. V entitách Texture, Model a Asset budou přímo uložena data pro načtení, například v Texture to může být obrázek ve formátu .png a v Model např. model ve formátu .obj nebo .fbx.
- **Entita NPC actor** Vytvoření nové entity odpovídající jedné NPC postavě je velmi podobné entitě Building, má podobné vlastnosti a atributy. Navíc přibudou atributy pro pohyb, tzn. čas, rotace a vektor pro interpolace pohybu, a bude odstraněn atribut s odkazem na webovou stránku.
- **Entita Portal** Nová entita, která odpovídá jednomu portálu pro teleport. Má v sobě unikátní ID, ID oblastí, mezi kterými se teleportuje, a lokaci portálu zadanou světovými souřadnicemi. Předpokládáme, že uživatel se může teleportovat oboustranně.
- **Entita Area** Vytvoření entity, která odpovídá jedné oblasti a nese informace o budovách, portálech a NPC postavách, které se nacházejí uvnitř. Každá oblast musí mít definované souřadnice: minimální a maximální světovou délku a šířku a také bod, kde se objeví uživatel, když se přemístí do vybrané oblasti. Každá oblast bude mít unikátní jméno/ID. Také obsahuje odkazy na všechny objekty (teleporty, budovy a NPC postavy), které jsou v dané oblasti.

Na obrázku 2.8 je zobrazen předpokládaný datový model pro novou verzi API s ohledem na navržené změny. Nezahrnuje skryté entity, které nebudou použity v této bakalářské práci.



Obrázek 2.8: Návrh nového datového modelu

2.3 Vývoj virtuální reality

Virtuální realita je realistická a pohlcující simulace trojrozměrného prostředí vytvořená pomocí interaktivního softwaru a hardwaru a řízená pohybem těla. Používá se v různých oblastech života: hry, turismus, vzdělání, architektura [11]. Posledních několik let se virtuální realita prudce vyvíjí a na trhu se objevuje stále nový software pro tvorbu virtuální reality. Některý je určen výhradně pro vývoj VR, AR a MR aplikací (používá se pojem XR aplikace pro označení všech typů rozšířené reality dohromady) nebo jenom pro jeden typ headsetu. Jiný software je multiplatformní a je určen nejenom pro tvorbu XR aplikací, ale i pro 2D a 3D hry.

Tato kapitola je zaměřena na analýzu softwaru pro tvorbu virtuální reality. Byly vybrány nejznámější multiplatformní herní enginy Unity 3D a Unreal Engine. Budou analyzovány speciálně prostředí a nástroje pro vývoj virtuální reality a poté provedena výběrová řízení na základě požadavku této bakalářské práce.

2.3.1 Vývoj v Unity 3D

Unity 3D je jeden z nejpopulárnějších herních enginů, který umožňuje tvorbu 2D a 3D multiplatformních aplikací. Je velmi populární mezi začátečníky, protože je jednoduchý pro naučení, disponuje velkou komunitou a existuje k němu celá řada materiálů k nastudování. Nabízí skriptovací API v jazyce C#. Umožňuje nastavení osvětlení, animace, fyziky kolize částicových efektů a má mnoho dalších vlastností. Podporuje i tvorbu aplikace virtuální nebo rozšířené reality [12].

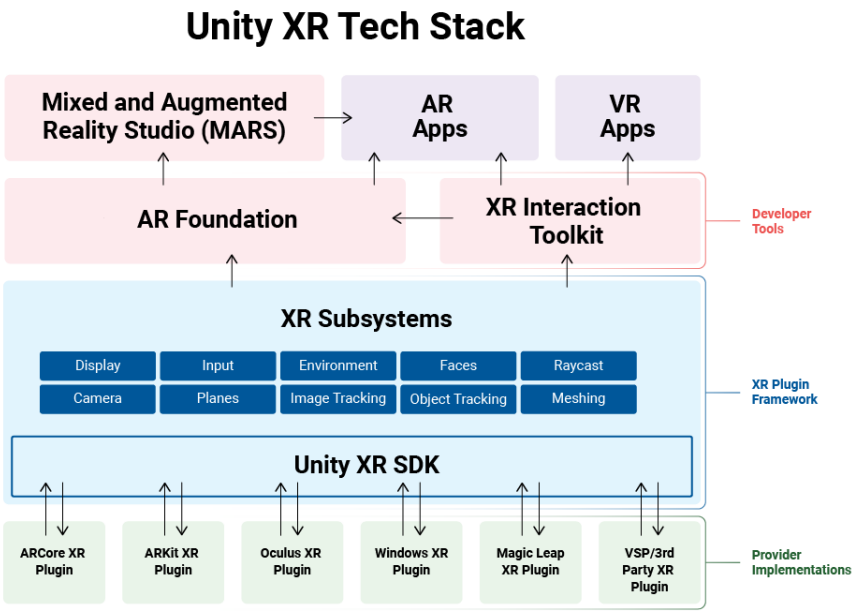
Pro implementaci VR aplikací v Unity 3D je třeba použit jeden z existujících balíčků VR SDK, který podporuje tvorbu XR aplikací, mít nainstalovaný zásuvný modul OpenXR a mít správně připojený jeden z podporovaných headsetů. Dále bude představeno několik populárních VR SDK a popsány jejich základní vlastnosti.

SteamVR Zásuvný modul od společnosti Valve, který propojuje SteamVR a Unity 3D. Pro použití je třeba mít nainstalovaný SteamVR, nainstalovaný SteamVR Unity Plugin a headset, který je kompatibilní se SteamVR. Podle oficiální dokumentace [13] můžeme možnosti tohoto zásuvného modulu rozdělit na pět kategorií: Render Models, SteamVR Input, Skeleton Input, Interaction System a Skeleton Poser. *Render Models* poskytuje základní animovaný model ovladače, který vidí uživatel ve virtuálním prostředí, takže když uživatel zmáčkne tlačítko v reálném životě, uvidí to i ve virtuálním prostředí. *SteamVR Input* abstrahuje části kódu specifické pro zařízení, takže vývojář je nemusí řešit a může se soustředit na programování své akce. *Skeleton Input* a *Skeleton Poser* jsou další metody pro zobrazování ovladačů ve virtuálním prostředí pomocí 3D modelů lidských rukou a snaží se napodobit současnou polohu rukou ve virtuálním prostředí. *Interaction System* je sada skriptů a

hotových elementů, které jsou použity v ukázkovém projektu SteamVR Unity Plugin, tam je například implementovaný teleport, interakce, uživatelské rozhraní atd. Může sloužit jako inspirace pro vytváření vlastního projektu pomoci SteamVR Unity Plugin.

VRTK neboli Virtual Reality Toolkit Open-source sada nástrojů a skriptů pro tvorbu VR v Unity 3D. Neobsahuje žádné nízkoúrovňové komponenty, proto je závislý na jiných balíčcích jako SteamVR, a proto architektura interakce je velmi podobná [14]. VRTK umožňuje základní interakce: teleport, uchopení předmětů, interakce s prvky uživatelského rozhraní Unity 3D, 2D a 3D elementy ovládaní (tlačítka, dveře, páky) a mnoho dalších. Pro použití toolkitu ho stačí stáhnout v Unity Asset Market, naimportovat složku Assets/VRTK do složky vlastního projektu a přidat skript VRTK_SDKManager do GameObject ve scéně [15].

UnityXR Interaction Toolkit Oficiální balíček od Unity umožňující vývoj XR aplikací. Technologický stack se skládá z API, které poskytuje společné funkce všem podporovaným XR platformám a umožňuje poskytovatelům hardwaru a softwaru XR vyvíjet jejich vlastní plug-iny Unity (obrázek 2.9) [16]. Pro použití UnityXR je třeba stáhnout XR Plug-in Management, vybrat vývojovou platformu a též vybrat příslušné zásuvné moduly. UnityXR



Obrázek 2.9: Aktuální struktura zásuvného modulu Unity XR [16]

skládá se ze čtyř základních komponent: Interactors, Interactables, Interaction Manager a Controllers. *Interactors* jsou komponenty, které zpracovávají inter-

akce s objekty ve scéně a většinou jsou připojeny k ovladačům. *Interactables* jsou objekty, se kterými lze interagovat ve scéně. *Interactors* a *Interactables* interagují mezi sebou pomocí *Interaction Manager*, který registruje a odregistruje interakční prvky a může donutit interaktora vybrat interakční prvek. Poslední komponent se nazývá *Controllers* a zajišťuje sledování a manipulaci se vstupy pro fyzické ovladače. Tato komponenta také poskytuje možnost připojit 3D model představující ovladač ve scéně. Baliček UnityXR má několik hotových skriptů, které implementují pohyb pomocí teleportu, uchopení předmětů a plynulý posun [17].

Každý z analyzovaných balíčků má své klady a zápory. Pro různé případy užiti jsou různá kritéria výběru, ale nejuniverzálnějším balíčkem je UnityXR Interaction Toolkit. I když je jedním z nejnovějších, má celou řadu výhod: podporuje většinu zařízení a platforem, umožňuje tvorbu VR, AR a MR aplikací, stále se aktualizuje, má dobrou dokumentaci, je oficiálním balíčkem přímo od Unity, má implementované všechny základní interakce a je nejlepší volbou pro začátečníky, protože je poměrně jednoduchý pro naučení [18].

2.3.2 Vývoj v Unreal Engine 4

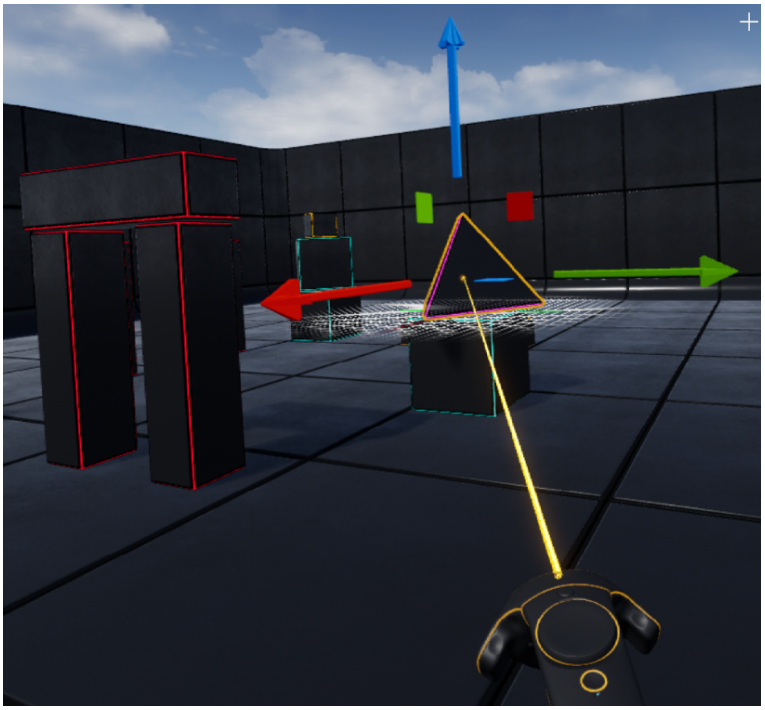
Unreal Engine je populární multiplatformní herní engin od společnosti Epic Games, který se používá nejenom pro vývoj her, ale i pro architektonickou a automobilovou vizualizaci, tvorbu filmového a televizního obsahu, školení a simulace. Zkušenější programátoři používají programátorský jazyk C++ pro tvorbu vlastních skriptů, ale existují i tak zvané Blueprinty, které umožňují vizuální skriptování. Je to systém postavený na konceptu použití rozhraní založeného na uzlech k vytváření herních prvků v Unreal Editoru. Má velmi výkonné nástroje pro tvorbu materiálů a animace, má vestavěný fyzikální engine umožňující částicové efekty a fyziku měkkých těles [19].

Stejně jako Unity 3D používá zásuvný modul OpenXR pro tvorbu VR, AR a MR aplikací. Je automaticky nainstalovaný do Unreal Engine. Pro většinu VR zařízení není třeba mít nainstalované nějaké další balíčky nebo zásuvné moduly. Stačí mít správně nastavený headset (každý headset má své požadavky, detailní instalační příručka je k dispozici na oficiálních stránkách každého headsetu) a nainstalovaný příslušný VR headset driver software, který musí být spuštěn, když se headset používá. Dále stačí vytvořit projekt v Unreal Engine, najít tlačítko *„Play“* a vybrat *VR Preview* jako režim spuštění. Výjimku tvoří mobilní VR headsety jako Samsung Gear nebo Oculus Go, pro něž je třeba projekt zabalit a nasadit do zařízení, aby aplikac bylo možné spustit na headsetu. Také je třeba nainstalovat Android Debug Bridge, NVIDIA CodeWorks for Android a nastavit Android SDK [20]. Mobilní headsety se při tvorbě této bakalářské práce nebudou používat, proto podrobnější informace o nastavení není předmětem této práce.

Unreal Engine má několik ukázkových šablon, kde jsou implementovány základní interakce ve virtuálním prostředí. Jsou implementovány pomocí Blu-

eprintu. Šablona *HMD/Gamepad Teleportation* nabízí pohyb pomocí gamepadu nebo klávesnice a interakci střelby. Šablona *Motion Controller Teleportation* ukazuje implementace pohybu pomocí teleportu a chycení objektu. Tyto šablony lze využít jako výchozí bod pro samostatný projekt.

Unreal Engine má velmi zajímavý režim, který se nazývá *VR Mode* a umožňuje uživateli upravovat vývojový svět přímo ve virtuální realitě. V tomto režimu uživatel může měnit velikost, rotovat, posouvat objekty, teleportovat se ve světě a otáčet s ním. Tento režim má vlastní uživatelské rozhraní s velkým množstvím užitečných nástrojů (obrázek 2.10).



Obrázek 2.10: Ukázka VR režimu Unreal Engine [20]

2.3.3 Vybraný software

Unity 3D a Unreal Engine jsou nejpopulárnější herní enginy pro vývoj her a VR, AR aplikací a je obtížné říci, který z nich je lepší pro vývoj VR aplikace, záleží na užití konkrétní aplikace. Herní enginy byly porovnány podle následujících kritérií:

Proces zprovoznění Pro tvorbu VR aplikace v Unity 3D je nutné stahovat externí balíčky nebo zásuvné moduly, vývojář má velký výběr balíčků, každý má vlastní proces nainstalování a tento proces může byt náročný. Na druhou stranu Unreal Engine už má zprovozněné všechny potřebné zásuvné moduly a stačí zprovoznit headset a vytvořit projekt s VR šablonou.

Cena Cenově jsou Unreal Engine a Unity 3D velmi odlišné, ale každý z nich má licenci zdarma. Pokud je projekt výdělečný, Unity nabízí podnikatelskou licenci, která stojí od 399 do 4 000 USD za rok. V Unreal Engine se platí 5 % výdělku jako licenční poplatky. Tato bakalářská práce není výdělečným projektem, proto jak pro Unity 3D, tak i pro Unreal Engine může být použita licence zdarma [21].

Dokumentace Každý z vybraných herních enginů má rozsáhlou dokumentaci a velkou komunitu. K tomu existuje velké množství vzdělávacích kurzů a videí. Unity 3D jako volně přístupný engine existuje déle než Unreal Engine 4, takže má přirozeně větší komunitu nezávislých vývojářů. Například na fóru Unreal Engine je 12 000 témat o programování v C++ a 4 600 témat o vývoji VR a AR. Na fóru Unity3D je více než 128 000 témat o skriptování a více než 6 100 témat o vývoji VR. To znamená, že pokud se vyskytne jakýkoli problém, existuje velká šance, že najdete někoho, kdo stejný nebo podobný problém již řešil [21].

Technická stránka Unreal Engine používá C++ nebo Blueprints skriptování a Unity 3D používá C#. Pro tvorbu více optimalizovaných a flexibilnějších programů v Unreal Engine nelze použít vizuální skriptování (Blueprints), proto je nutné umět aspoň základy C++. Tento programovací jazyk je těžší pro naučení než C#, proto začátečníci většinou vybírají Unity 3D. Na druhou stranu Unreal Engine je open-source engine, když Unity 3D není. To znamená, že v Unreal Engine jde modifikovat zdrojový kód a v Unity 3D jenom prohlížet.

Grafika Oba enginy umí produkovat vynikající grafiku, nicméně Unreal Engine je považován za lepší. Unreal nabízí velké množství nástrojů, které fungují hned při vytváření projektu a které není třeba upravovat. Největší výhodu v grafice pro Unreal představuje systém osvětlení, díky kterému grafika vypadá velmi realisticky [22].

Cílová platforma Fóra dokládají, že Unreal Engine je lepší pro 3D vývoj a Unity 3D je známý díky dobré podpoře mobilního vývoje, a proto je třeba vybírat herní engine na základě cílové platformy. VR headsety také můžeme rozdělit na mobilní (Google's Cardboard nebo Samsung Gear VR) a počítačové (Oculus nebo HTC Vive). V případě této bakalářské práce budou použity brýle HTC Vive nebo HTC Vive Pro, proto můžeme říci, že tato práce je zaměřena na vývoj aplikace pro počítačové VR headsety.

Dostupnost assetů Pro stahování assetů pro Unity 3D se používá Unity Asset's store, kde je na výběr mnoho modelů zdarma a většinu z nich lze implementovat do VR aplikace. V Unreal Engine existuje Epic Game Marketplace, který nabízí o dost méně modelů a většina z nich je placena. V této bakalářské práci žádné externí assety nebudou použity, protože všechny potřebné modely se nachází na privátním API projektu Věnná města českých královen.

Zkušenost vývojáře Před tvorbou této bakalářské práce jsem už měla pracovní zkušenost s Unreal Engine a tvorbou VR aplikace v tomto herním

enginu. K tomu ovládám jazyk programování C++ a jsem dobře seznámena s vizuálním programováním v Blueprints. Při mé práci na projektu Věnná města českých královen v rámci předmětu softwarový týmový projekt byl také použit Unreal Engine. Co se týče Unity 3D, slyšela jsem o něm, ale nezkoušela a nikdy jsem neprogramovala v C#.

Z analyzovaných bodů lze vidět, ze Unreal Engine a Unity 3D mají své klady a zápory, ale každý z nich je možné použít pro tvorbu XR aplikace. Unreal je lepší pro tvorbu pocitových nebo konzolových aplikací a VR aplikací. Unity 3D má větší podporu mobilních a AR aplikací. Pro tuto bakalářskou práci byl vybrán herní engine **Unreal Engine**, a to hlavně díky velké zkušenosti vývojáře projektu, jednoduchému zprovoznění a použití hotových vestavěných šablon a také cílové platformě projektu, která je silnou stránkou Unreal Enginu.

2.4 Blender a vytváření zásuvných modulů

Blender je multiplatformní bezplatná aplikace, která umožňuje uživatelům vytvářet 3D vizualizace, 3D animace, editovat videa apod. Má malé požadavky na systém na rozdíl od ostatních aplikací pro tvorbu 3D grafiky. Používá rozhraní OpenGL pro poskytnutí konzistentního zážitku napříč vsemi podporovanými platformami a hardwarem [23].

V této kapitole budou popsány možnosti rozšíření aplikace Blender, princip a požadavky pro psaní zásuvných modulů.

2.4.1 Rozšíření aplikace Blender

Aplikace Blender se rozšiřuje pomocí skriptů, používá se Blender/Python API, které umožňuje programátorovi upravovat data (scény, částice atd.), vytvářet nové nástroje, vytvářet nové prvky uživatelského rozhraní, definovat nová nastavení, vytvářet animace atd. Dva nejběžnější způsoby spuštění skriptu jsou použití vestavěného textového editoru nebo zadávání příkazů do konzoly Python, tyto způsoby jsou dostupné přímo z horního panelu (záložka Scripting) [24]. Skripty rozšiřující funkcionalitu lze rozdělit na několik podtypů:

- **Add-ons(doplňky)** – skripty, které rozšiřují funkcionalitu. Mají stejnou funkcionalitu jako jakýkoliv jiný skript, který je spuštěn z textového editoru nebo konzoly. Výhodou použití add-onu je snadné využití a nainstalování dalšími uživateli. Pro přidání a povolení použití add-ons slouží tato cesta – *Edit→Preferences→Add-ons*. Existuje mnoho oficiálních doplňků přímo od vývojáře Blenderu, ale také na webu jsou stovky add-onů napsaných různými lidmi. Tyto doplňky lze stáhnout z webu a nainstalovat do programu pomocí příkazu *Install*, který lze nalézt v *Edit→Preferences→Add-ons*.
- **Modules** – pomocné knihovny pro import do jiných skriptů.

- **Presets(předvolby)** – nastavení nástrojů a klíčových konfigurací aplikace Blender.
- **Startup(spuštěni)** – tyto soubory jsou importovány při spuštění aplikace Blender. Definují většinu uživatelského rozhraní Blenderu a některé dodatečné základní operátory.
- **Custom Scripts(vlastni skripty)** – na rozdíl od doplňků jsou obvykle určeny pro jednorázové provedení prostřednictvím textového editoru.

2.4.2 Princip a požadavky

V této bakalářské práci bude implementován zásuvný modul, který je tvořen jako add-ons, proto v této části bude popsán princip tvoření a požadavky pro add-ony.

Doplňky jakož i další typy skriptů se vytvářejí pomocí programovacího jazyka Python a speciálního Blender/Python API. Proto pro psaní add-onu programátor musí mít aspoň základní znalosti Pythonu, mít představu o modulech a být dobře seznámen s Blenderem.

Pro programování add-onu lze použít jakýkoliv textový editor, který je pak třeba uložit jako Python soubor. Když se kód skriptu skládá z více Python souborů, je třeba zabalit všechny do jednoho .zip souboru. Také lze pro psaní skriptu použít textový editor a konzoly, které se nacházejí přímo v Blenderu, záložka Scripting.

Před začátkem programování zásuvného modulu je třeba rozmyslit, jaký doplněk bude, zda bude například vytvářet geometrii a objekty nebo rozšiřovat funkcionalitu materiálů a textur nebo provádět animace apod.

Také je důležité se zamyslet, jestli budou používány určité matematické a grafické funkce a jestli budou použity externí Python knihovny (jako například SciPy, NumPy). Všechny dostupné knihovny pro Python lze použit i pro psaní add-onů [25].

Hlavním požadavkem pro tvoření doplňků je správně vytvořené nastaveni prostředků a registrace/odregistrace add-onu. Na fragmentu kódu, znázorněném níže, který je převzat z oficiálních stránek aplikace Blender, jsou ukázány všechny prvky pro správné fungování zásuvného modulu. Všimněme si, že tento add-on nedělá nic, co by souviselo s Blenderem, protože nepoužívá Blender/Python API, které je nutné pro komunikaci přímo s aplikací. Za konfiguraci odpovídá *bl_info* – slovník, který odpovídá za metadata add-onu. V tomto případě jsou určeny název a kategorie. Také je zadefinována minimální verze pro spuštění, v příkladu to je verze 2.80. Dále lze nadefinovat popisek, verzi add-onu, lokaci, upozornění atd. Tyto všechny vlastnosti se zobrazí u doplňku v okně *Preferences*. Funkce *register()* se spouští při povolení add-onu. Zde se zapíší všechny funkce, které budou použity pro fungování doplňku. Funkce *unregister()* uvolní všechno, co bylo nastaveno při registraci, bude-li add-on vymazán.

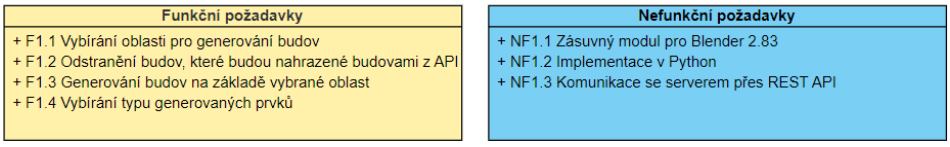
```
bl_info = {
    "name": "Add-on",
    "blender": (2, 80, 0),
    "category": "Object",
}
def register():
    print("HelloWorld")
def unregister():
    print("GoodbyeWorld")
```

2.5 Funkční a nefunkční požadavky

Tato kapitola se zabývá analýzou funkčních a nefunkčních požadavků na zásuvný modul pro generování budov a na VR klienta v Unreal Engine. Požadavky byly konzultovány s vedoucím práce a dále budou použity pro tvorbu návrhu. Protože v čase tvorby požadavků ještě nebylo zcela hotovo API pro tento projekt, je obtížné určit, co všechno bude možné splnit v implementaci.

2.5.1 Požadavky pro zásuvný modul

Tato podkapitola obsahuje model funkčních a nefunkčních požadavků na zásuvný modul pro generování budov. Nebude implementován celý zásuvný modul, ale jen modifikován již existující, rozsah celého modulu mohl být zadán jako samostatná bakalářská práce a není to hlavním cílem této práce, proto do požadavků budou přidány jenom ty body, které jsou nutné pro účely této bakalářské práce bez ohledu na již existující a nepoužité funkcionality základního zásuvného modulu. Na obrázku 2.11 jsou znázorněny požadavky.



Obrázek 2.11: Požadavky pro zásuvný modul

F1.1 Vybírání oblasti pro generování budov

Zjednodušení vybírání oblastí, které jsou uloženy na API. Uživatel může zvolit jednu z nabízených oblastí a nemusí zadávat souřadnice ručně. Oblasti se aktualizují při každém spuštění aplikace.

F1.2 Odstranění budov, které budou nahrazené budovami z API

Při výběru generováni budov podle vybrané oblasti, odstraňují se budovy, které se nachází na místě, kde budou se nacházet historické budovy z API.

F1.3 Generování budov na základě vybrané oblasti

Generování na základě aktuálních mapových podkladů. Budovy se generují na základě dat s OpenStreetMap API. Pokud ve vybrané oblasti leží jenom část budovy, bude vygenerována celá a uživatel ji pak může odstranit podle potřeby.

F1.4 Vybírání typu generovaných prvků

Uživatel může vybrat, zda kromě budov chce také vygenerovat určité doplňky, například vegetaci, cesty, vodní zdroje apod.

NF1.1 Zásuvný modul pro Blender 2.83

Modifikace zásuvného modulu bude implementována pro Blender 2.83. Pro jiné verze funkčnost zásuvného modulu nebude testována.

NF1.2 Implementace v Python

Bude použit vývojový jazyk Python doplněný Blender/Python API a potřebnými knihovnami.

NF1.3 Komunikace se serverem přes REST API

Použití metod REST API, například GET, POST apod. Z API budou převzaty informace o jménu a lokaci oblasti.

2.5.2 Požadavky pro VR klienta

Cílem této podkapitoly je sepsání funkčních a nefunkčních požadavků pro VR klienta – historického průvodce (obrázek 2.12).

Funkční požadavky
+ F2.1 Vybíraní oblasti pro prohlížení
+ F2.2 Prohlížení oblasti
+ F2.3 Změna lokace pomocí teleport portálů
+ F2.4 Zobrazení informace o významných budovách
+ F2.5 Zobrazení mapy prohlíženého prostředí

Nefunkční požadavky
+ NF2.1 Implementace v Unreal Engine 4
+ NF2.2 Implementace pro virtuální realitu
+ NF2.3 Využití technologie Blueprint, případně doplnění C++
+ NF2.4 Komunikace se serverem pomocí zásuvného modulu VaRest
+ NF2.5 Dokumentace

Obrázek 2.12: Požadavky pro VR klienta

F2.1 Vybíraní oblasti pro prohlížení

Při vstupu do aplikace uživatel vybírá oblast, kterou chce prohlížet. Vždy se lze vrátit zpátky a vybrat jinou oblast. Existuje-li oblast na API, ale není v aplikaci, uživatel obdrží varování a může se vrátit zpátky a vybrat jinou.

F2.2 Prohlížení oblasti

Uživatel může prohlížet vybranou oblast pomocí teleportu. Na začátku se objeví v lokaci, která je zadána na API jako počáteční pozice oblasti.

F2.3 Změna lokace pomocí teleport portálů

Jde o teleport portály, kdy při vstupu uživatel mění lokace. Může jít o stejnou nebo jinou oblast. Cílové a výchozí souřadnice teleportu jsou určeny z dat z API.

F2.4 Zobrazení informace o významných budovách

Při zmačknutí speciálního tlačítka na ovládači se zobrazí webová stránka, kde uživatel uvidí informace o nejbližší historické budově.

F2.5 Zobrazení mapy prohlíženého prostředí

Uživatel může otevřít mapu prohlíženého prostředí. Na ní bude zobrazena mapa celé oblasti a označení (tečka), kde se nachází uživatel.

NF2.1 Implementace v Unreal Engine 4

Prototyp aplikace bude implementován v herním enginu Unreal Engine 4, verze 4.26 s využitím zásuvných modulů dostupných na Marketplace Epic Games.

NF2.2 Implementace pro virtuální realitu

VR klient bude implementován pro zařízení virtuální reality, konkrétně pro HTC Vive. Nastavení projektu budou odpovídat nastavením projektu pro VR vývoj.

NF2.3 Využití technologie Blueprint, případně doplnění C++

Pro vývoj prototypu bude využita technologie Blueprint, v případě, že implementace v Blueprint není možná, bude využit programovací jazyk C++.

NF2.4 Komunikace se serverem pomocí zásuvného modulu VaRest

Pro zjednodušení REST API komunikace a možnost komunikace se serverem pomocí Blueprint technologie bude použit zásuvný modul VaRest, který je dostupný zdarma na Marketplace Epic Games.

NF2.5 Dokumentace

Kód aplikace bude okomentován v angličtině, aplikace bude mít instalační, uživatelskou a programátorskou příručku.

Kapitola 3

Návrh

3.1 Návrh zásuvného modulu do Blenderu

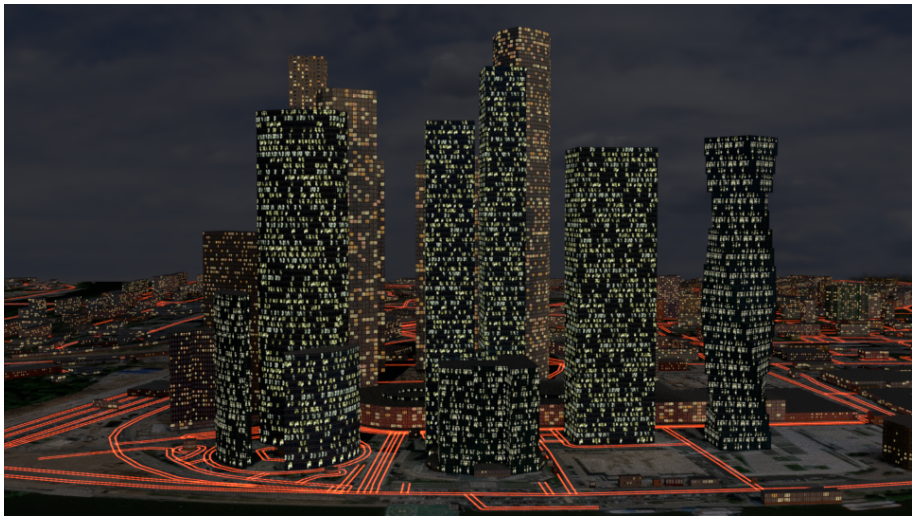
Jelikož návrh a implementaci celého zásuvného modulu pro generování budov s použitím mapových podkladů můžeme považovat za téma pro samostatnou bakalářskou nebo diplomovou práci a v této práci bude tento zásuvný modul použit jenom pro generování budov v rámci hlavního výsledku práce, tzn. VR klienta, bude metodou výběrového řízení vybrán nejlepší již existující add-on, který bude modifikován pro dosazení funkčních a nefunkčních požadavků této práce. Na základě dostupnosti, kladných recenzí a existence licencí, které dovolují modifikovat, byly zvoleny dva doplňky **Blender OSM** a **GIS**.

V této kapitole budou detailněji popsány tyto dva zásuvné moduly a v závěru provedeno výběrové řízení pro zvolení lepší varianty pro použití v této bakalářské práci.

3.1.1 Blender OSM

Blender OSM je open-source doplněk, který umožňuje vygenerování budov a reálných dat terénu do Blenderu. Navíc umí generovat různou vegetaci, vodní objekty, cesty apod. Používá veřejné OpenStreetMap API. OpenStreetMap je mapa světa, která je tvořena různými nezávislými osobami a má otevřenou licenci pro použití.

Existují dvě verze zásuvného modulu: základní a prémiová. Základní verze generuje jednoduché 3D modely budov s použitím základních materiálů a uživatel má na výběr několik typů střech. Prémiová verze kromě základních funkcí navíc umí generovat budovy s texturami a UV mapováním, materiály a textury umějí napodobovat svítící okna při vybraném nočním režimu, existuje možnost využívat vlastní textury pro budovy. Navíc dokáže generovat jednotlivé stromy zvlášť jako 3D objekty a tvořit z nich celý les. Na obrázku 3.1 je ukázka výsledku fungování prémiové verze doplňku se zapnutým nočním režimem a svítícími okny.



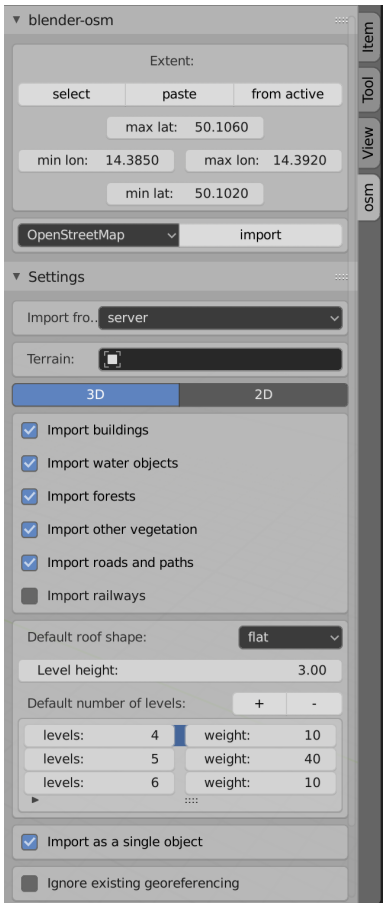
Obrázek 3.1: Ukázka prémiové verze Blender OSM[27]

3.1.1.1 Princip fungování

Princip fungování doplňku Blender OSM bude popsán na základní verzi. Pro nainstalování doplňku je třeba stáhnout .zip soubor z odkazu, který je uveden v oficiální dokumentaci [28]. Následně je třeba přidat ho do add-onu Blenderu (*Edit→Preferences→Add-ons*) a povolit ho. Také je třeba zadat složku, ve které se budou ukládat data z OpenStreetMap. Doplněk lze nalézt v režimu 3D view v sekci Tools (jde otevřít pomocí klávesy N).

Uživatelské rozhraní doplňku je rozděleno na několik částí (obrázek 3.2). První část obsahuje výběr souřadnic pro generování budov, tlačítko *„select“* přemístí uživatele na web s mapou, který slouží pro výběr obdélníkové oblasti pro generování, a pomocí tlačítka *„paste“* uživatel vloží na webu vybrané souřadnice do doplňku. Pokud je alespoň jedna oblast vygenerována, aktivuje se tlačítko *„from active“*, pomoci kterého uživatel může vybrat stejné souřadnice jako u vybrané vygenerované oblasti. Další podsekce je určena pro zvolení dat, ze kterých se budou generovat budovy nebo jiné geografické objekty (například při zvoleni *„terrain“* bude vygenerován jenom terén). Cílům této bakalářské práce vyhovuji data *OpenStreetMap*, která se načítají z příslušného API a nesou v sobě informace o všech objektech v oblasti omezené vybranými souřadnicemi. Poslední část uživatelského rozhraní odpovídá za všechna nastavení generované oblasti. Uživatel může vybrat typ generovaných objektů (2D nebo 3D), podklady pro terén, různá nastaveni generovaných budov (typ střechy, velikost budov atd.), zda chce používat lokální data nebo data ze serveru a co všechno chce vygenerovat. Existuje možnost vybrat, jestli chce mít generované budovy jako jeden objekt, nebo každý zvlášť. Je to velmi užitečná funkce pro malé korekce vygenerované oblasti, například pokud se

některá budova vygenerovala nesprávně nebo jenom její malá část leží v zadané oblasti a uživatel ji chce upravit nebo úplně odstranit.



Obrázek 3.2: Uživatelské rozhraní doplňku Blender OSM

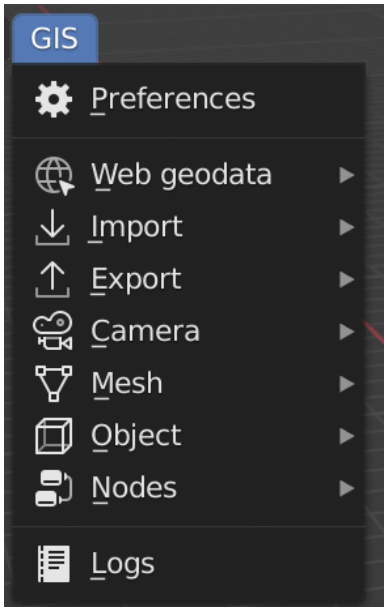
3.1.2 Blender GIS

Blender GIS je doplněk do Blenderu, který umožňuje generování budov na základě mapových podkladů. Má jednu open-source verzi, která je uložena spolu s dokumentací na GitHubu projektu [29]. Má velmi podobnou funkcionalitu a implementaci jako předchozí doplněk, proto rovnou přejdeme k principu fungování.

3.1.2.1 Princip fungování

Pro spuštění add-onu je třeba stáhnout .zip soubor z GitHubu projektu, dále ho přidat do sekce Add-ons v *Edit→Preferences* a povolit ho. Určité parametry lze nastavit jak v sekci *Preferences*, tak i v uživatelském rozhraní doplňku,

které je znázorněno na obrázku 3.3. Záložka GIS se nachází v horním levém panelu. Pro cíle bakalářské práce je třeba jen rozdíl *Web geodata*, všechny ostatní rozdíly jsou pravděpodobně stále ve vývoji, protože nejsou zmiňovány v dokumentaci k doplňku a není úplně jasná jejich funkcionalita. Rozdíl *Web*



Obrázek 3.3: Uživatelské rozhraní doplňku Blender GIS

geodata má na výběr tři příkazy. První, který se jmenuje *Basemap*, vytváří základ terénu na základě vybrané mapy. Mapa se objeví přímo v Blenderu. Uživatel má na výběr z několika známých serverů s mapovými podklady jako Google, Bing apod. Dalším příkazem je *Get OSM*, pomocí něhož se generují budovy. Má na výběr několik typů objektů (budovy, cesty, vegetace atd.), uživatel také může zvolit výšku budov, a zda chce mít budovy spojené jako jeden objekt nebo každou budovu zvlášť. Poslední příkaz, který se jmenuje *Get elevation SRTM*, odpovídá za reliéf.

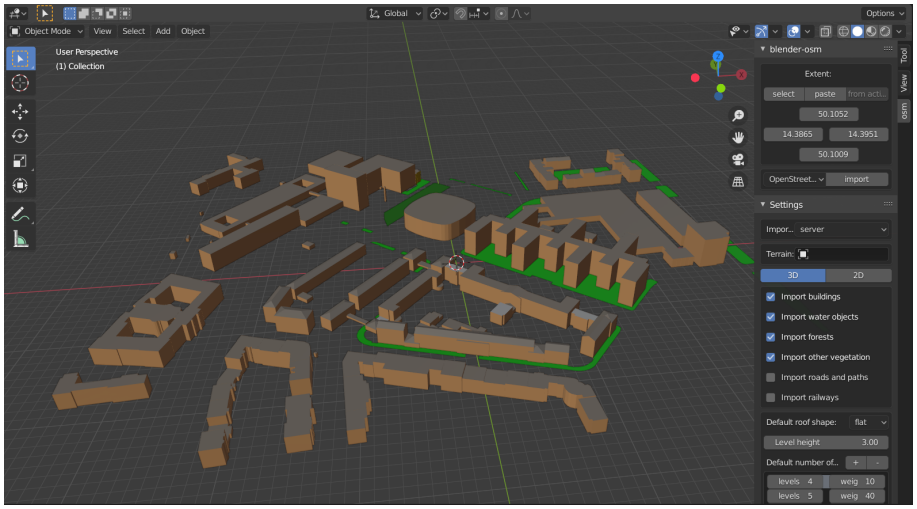
Na závěr lze říci, že tento doplněk je velmi podobný předchozímu, má shodnou funkcionalitu a používá stejné technologie.

3.1.3 Vybrané řešení

Dva výše zmíněné doplňky byly porovnány podle následujících kritérií:

Vizuální vzhled Na obrázcích 3.4 a 3.5 jsou představeny modely kampusu ČVUT generované pomocí popsaných doplňků. Je patrné, že vizuálně lépe vypadá model, který je vytvořen pomoci Blender OSM, protože používá více materiálů, pozice generovaných budov je přesnější než u Blender GIS a nejsou generovány nadbytečné věci.

Obrázek 3.4: Kampus ČVUT generovaný pomocí doplňku Blender GIS



Obrázek 3.5: Kampus ČVUT generovaný pomocí doplňku Blender OSM

Použité technologie Oba zásuvné moduly používají stejné postupy pro generování budov a stejné datové úložiště OpenStreetMap API jako mapové podklady.

Kód add-onu Kód každého doplňku je uložen na GitHubu projektu spolu s dokumentací. Kódy jsou logicky rozděleny do více složek a souborů, mají srozumitelné komentáře v angličtině, kód je dobře čitelný.

Dokumentace Dokumentace Blender OSM je popsána nejenom vývojářským týmem projektu, ale i běžnými uživateli, na webu existuje několik videí s ukázkou fungování doplňku. Každá část uživatelského rozhraní je dobře

a srozumitelně popsána. Na rozdíl od Blender OSM Blender GIS je špatně zdokumentován, má dvě ukázková videa s použitím jenom určitých funkcionalit, některé z funkcí nejsou vůbec srozumitelné a nemají ani žádnou nápovědu.

Podpora Poslední aktualizace Blender GIS byla provedena před 13 měsíci a zdá se, že vývoj doplňku je pozastaven, i když má stále mnoho drobných chyb a vad. Co se týče Blender OSM, stále je aktualizován, má speciální stránku, kde každý běžný uživatel může nahlásit chybu nebo navrhnout určité změny. Vývojáři rychle odpovídají na otázky a snaží se co nejrychleji všechno opravit.

Hodnocení uživatelů Lepší hodnocení od uživatelů má Blender OSM, uživatelé Blender GIS se setkávají s řadou problémů a pro většinu verzí Blenderu se objevují různé problémy a potíže. Při testování Blender GIS na platformě MacOS jsem zaznamenala nestandardní chování add-onu, i když na GitHubu projektu je uvedeno, že pro Blender 2.83 jsou všechny chyby napraveny a všechno by mělo fungovat správně.

Po zvážení všech kladných a negativních stránek lze jednoznačné vyvodit, že **Blender OSM** je lepší volbou pro tuto bakalářskou práci. Aktuální verze je stabilní, funguje pro téměř všechny nové verze Blenderu, má precizně popsanou dokumentaci a podpora rychle odpovídá na jakékoliv otázky, takže při vzniku potíží s modifikací doplňku je možné se na podporu obrátit pro radu.

3.1.4 Návrh modifikace vybraného doplňku

V této podkapitole bude popsán návrh modifikace pro doplněk Blender OSM. K tomu bude rozebrán princip generování budov pomocí OpenStreetMapAPI.

3.1.4.1 Princip fungování Blender OSM

Princip fungování zásuvného modulu je založen na mapových podkladech, které se získávají z OpenStreetMap API. Pokud uživatel zmáčkne tlačítko *„import“*, odesílá se požadavek na server se souřadnicemi zadanými uživatelem. Jako odpověď se vrací XML soubor s elementy OpenStreetMap. Existují tři typy elementů [30]:

- **Nodes (uzel)** – základní prvek v datové struktuře OSM označující konkrétní bod na povrchu Země. Každý uzel se skládá z dvojice souřadnic a svého jednoznačného identifikačního čísla (ID). Uzly se používají k určení průběhu cesty.

- **Ways (cesta)** – prvek definující lomenou čáru. Cesty se používají pro reprezentaci lineárních objektů, jako jsou například řeky nebo silnice. Většinou to je uspořádaný seznam od 2 do 2000 uzlů. Cestami jsou také určovány hranice plošných objektů, jako jsou budovy nebo lesy. V tomto případě cesta začíná i končí ve stejném uzlu.

- **Relations (relace)** – je víceúčelová datová struktura, která popisuje vztahy mezi více prvky (uzly, cestami a např. i dalšími relacemi). Tyto prvky se nazývají členy relace. Každý prvek se může v relaci vyskytovat vícekrát.

Každý z elementů může mít tag, je to pár klíč=hodnota. Tagy patří mezi nejjednodušší datové prvky a jsou základní metodou popisu geografických dat v OpenStreetMap. Mohou například popisovat typ nebo jméno elementu. Dále je představen příklad elementu cesty:

```
<way id="4876027">
  <nd ref="31492372"/>
  <nd ref="31492338"/>
  <nd ref="31492370"/>
  <nd ref="31492371"/>
  <nd ref="31492372"/>
  <tag k="natural" v="water"/>
  <tag k="name" v="Spegeldammen"/>
</way>
```

Je vidět, že se začíná a končí stejným uzlem, takže cesta popisuje hranice určitého objektu. Jeden z tagů naznačuje, že je to vodní objekt, takže je možné předpokládat, že tato cesta označuje hranice jezera.

Algoritmus zásuvného modulu prochází všemi daty a zjišťuje, zda daná cesta popisuje nějaký objekt potřebný pro generování (například budova, cesta, jezero). Pokud ano, vykreslí ji podle souřadnic uzlu, které jsou přiřazeny kontrolované cestě.

3.1.4.2 Přidání možnosti vybíraní oblasti z API

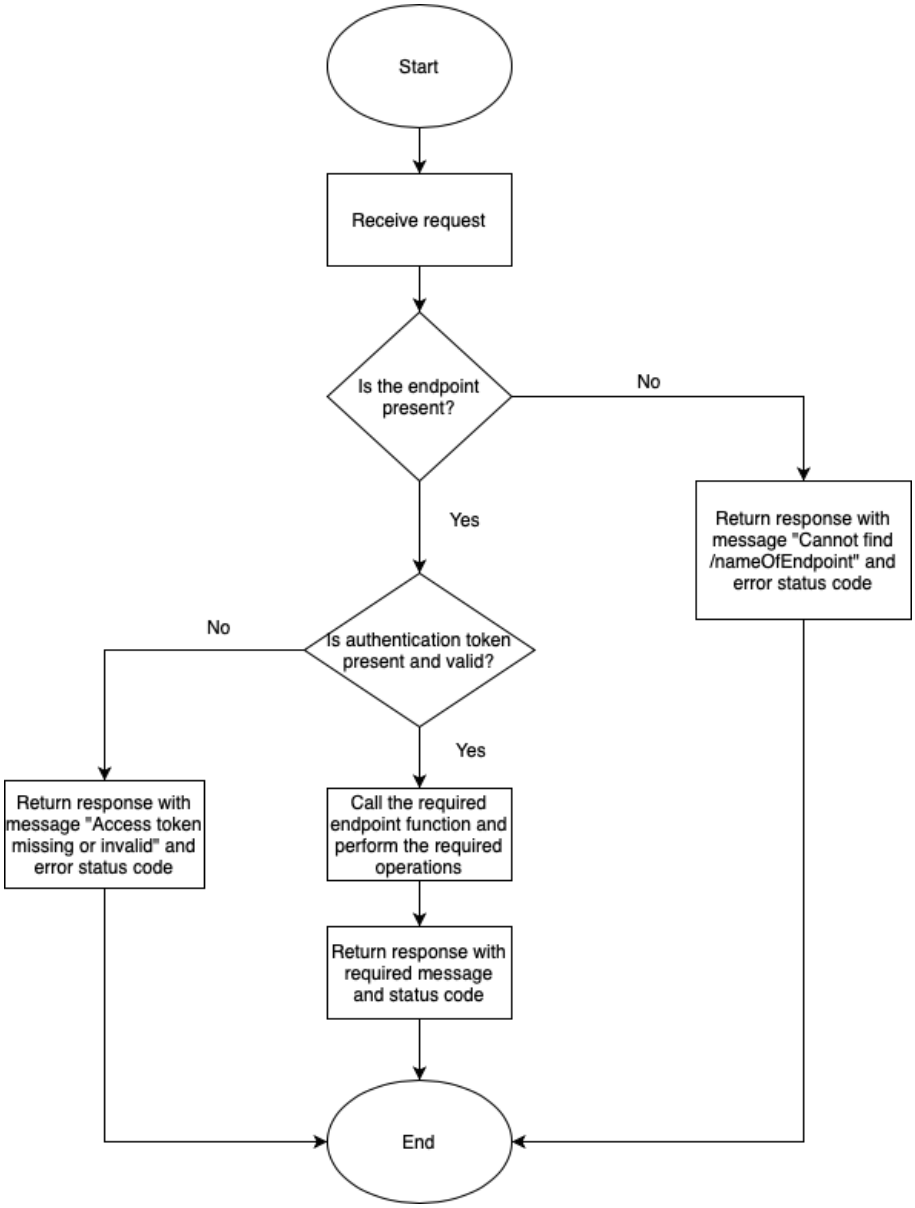
Jelikož API projektu VMČK obsahuje entitu *Areas* s atributem *„activationBorder“*, který nese informace o světových souřadnicích vybrané oblasti, je třeba přidat do doplňku možnost vybírání souřadnic pomocí nabízených oblastí existujících na API, aby je uživatel nemusel přepisovat do doplňku vlastnoručně. Tento element uživatelského rozhraní bude implementován pomocí rozbalovacího seznamu a při každém spuštění Blenderu se data z API budou aktualizovat.

Každá historická budova na API má vlastní souřadnice, kde se nachází a může se stát, že má stejné souřadnice jak generovaná budova, proto je třeba implementovat funkce, která kontroluje tuto kolize a odstraňuje generovanou budovu, na místě které potom bude historická budova.

3.1.4.3 Schéma komunikace s API

Na obrázku 3.6 je znázorněn flowchart diagram REST API komunikace s privátním API projektu VMČK. Stejný způsob komunikace se bude používat i pro VR klienta. REST API je architektonický styl pro aplikační programové rozhraní (API), které používá HTTP požadavky pro přístup a použití dat

[31]. Pro REST komunikaci zásuvného modulu s API bude použita Python



Obrázek 3.6: Flowchart diagram komunikace s API VMČK

knihovna Requests, která umožňuje jednoduše odesílat HTTP požadavky. Požadavek HTTP jako odpověď vrátí objekt se všemi daty (obsah, kódování, stav atd.) Syntaxe odpovídá requests.methodname(params), metody s popisem odpovídajících REST API příkazů jsou znázorněné v tabulce 3.1 [32]. V této bakalářské práci bude většinou použita metoda request pro odesílání zadaných metod a metoda get pro odesílání požadavků pro získání dat z API.

Metoda	Popis
delete(url, args)	Odešle požadavek DELETE (vymaže zadané data) na zadanou adresu URL
get(url, params, args)	Odešle požadavek GET (požaduje reprezentaci dat) na zadanou adresu URL
head(url, args)	Odešle požadavek HEAD (požaduje data jako metoda GET, ale bez těla odpovědi) na zadanou adresu URL
patch(url, data, args)	Odešle požadavek PATCH (používá se pro částečnou úpravu dat) na zadanou adresu URL
post(url, data, json, args)	Odešle požadavek POST (slouží k vytvoření nebo úpravě dat) na zadanou adresu URL
put(url, data, args)	Odešle požadavek PUT (nahradí všechny aktuální reprezentace dat novými z požadavku) na zadanou adresu URL
request(method, url, args)	Odešle požadavek zadané metody na zadanou adresu URL

Tabulka 3.1: Popis metod knihovny Requests

Doplněk také komunikuje s veřejným OpenStreetMap API, tato komunikace byla implementována vývojáři Blender OSM. Také používá REST API komunikaci, většinou její metodu GET. Komunikace je též implementovaná, a to pomocí Python knihovny *Requests*.

3.1.4.4 Skrytí nepoužité funkcionality

Blender OSM nabízí vlastnosti, které nebudou použity v této bakalářské práci, odváděly by pozornost uživatele, a proto budou skryté. Budou odstraněny možnosti vybírání typu střechy a velikosti generovaných budov, importování bude dostupné jenom ze serveru a zůstanou jenom dva typy dat pro generování budov, resp. terénu: OpenStreetMap API nebo terrain. Všechny skryté vlastnosti zůstanou v kódu, ale budou zakomentovány.

3.2 Návrh VR klienta

V této podkapitole bude popsán návrh VR klienta v herním enginu Unreal Engine, verze 4.26. Na začátku budou předvedena vybraná nastavení projektu, která jsou speciální pro tvorbu virtuální reality. Dále bude navrženo uživatelské rozhraní prototypu a popsány použité technologie a zásuvné moduly.

3.2.1 Nastavení projektu

Jelikož vývoj pro VR se značně liší od vývoje pro počítačové nebo mobilní platformy, je třeba provést několik změn v nastavení projektu, aby se výsledek správně zobrazoval a byl optimalizován [20]. Všechna nastavení lze najít v záložce **Project Settings** (nastavení projektu).

Instanced Stereo Ve VR potřebujeme vykreslit dva samostatné pohledy současně – jednou pro každé oko. Ve verzi Unreal Engine 4.11 a starších se tato činnost prováděla dvakrát a kreslily se dva téměř identické pohledy. Instance Stereo rendering toto vylepšuje tak, že vykreslí scénu jen jednou a následně ji upraví pro každé oko, což výrazně zrychlí proces renderingu. Proto je třeba tuto funkcionalitu mít zapnutou (nastavení na True).

Round Robin Occlusion Queries Pro projekty VR nabízí Unreal optimalizovanou metodu odstraňování dynamické okluze nazývanou Round Robin Occlusion, která testuje okluzi pouze pro jedno oko na snímek, nikoli pro obě. To ušetří značné množství času, zejména ve scénách s velkým množstvím objektů, a funkčnost je velmi dobrá, protože pohledy z každého oka jsou téměř totožné. Systém přepíná podle toho, které oko na každém snímku testuje. Je potřeba mít tuto funkcionalitu zapnutou (nastavení na True).

Forward Shading Forward Shading byl přidán do Unreal Engine speciálně pro projekty VR. Při použití Forward Shading je každý geometrický objekt ve scéně stínován při renderování a každé světlo ve scéně je kontrolováno, aby se zjistilo, jak by to ho mohlo ovlivnit. Pokud je ve scéně hodně objektů a hodně světel, může to způsobit řadu komplikací. U většiny VR projektů je vhodné použít dopředné stínování, protože kontroluje každý objekt a světlo zvlášť, takže je možné vypnout nepoužívané objekty.

Multisampling Anti-Aliasing Method Tato metoda vyhlazování je dostupná jen při použití Forward Shading a poskytuje lepší výsledky než například Fast Approximate Anti-Aliasing, který hledá okraje ve scéně a smíchá barvy na těchto okrajích. Je postavena na použití více vzorků vyhlazování na pixel pro barvu a hloubku během hlavního procesu vykreslování.

Ambient Occlusion Static Fraction Okolní okluze je metoda pro vytváření jemných stínů, které se objevují tam, kde se objekty vzájemně dotýkají, ale jejich výpočet je náročný a ve VR mohou vypadat nevhodně, protože se počítají v prostoru obrazovky, a proto je třeba mít ji vypnutou.

Start in VR Je důležité zadat do projektu, aby již začínal ve VR při samotném spuštění.

3.2.2 Použití VR šablony

Jako základ projektu bude použita VR šablona **Motion Controller Teleportation**, která je uložena jako samostatná úroveň se stejným názvem. Šablona už má implementován pohyb pomocí teleportu a též chytání předmětů. Také místo 3D objektů ovladačů se zobrazuji animované ruce. Interakce jsou

přičleněny k tlačítkům různých ovladačů včetně ovladačů k headsetu HTC Vive. Do této bakalářské práce bude převzat jen teleport a budou implementovány další interakce, například stisk tlačítka uživatelského rozhraní nebo navedení paprsku a kliknutí na objekt.

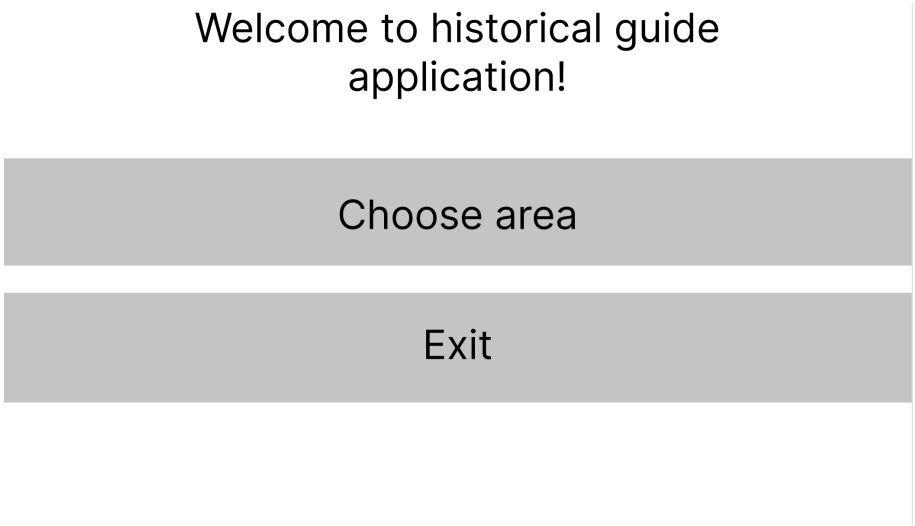
3.2.3 Návrh uživatelského rozhraní

Uživatelské rozhraní bude poměrné jednoduché a bude obsahovat tři widgety. Každý bude popsán zvlášť a návrhy uživatelského rozhraní jsou vytvořeny pomocí on-line servisu Figma [33].

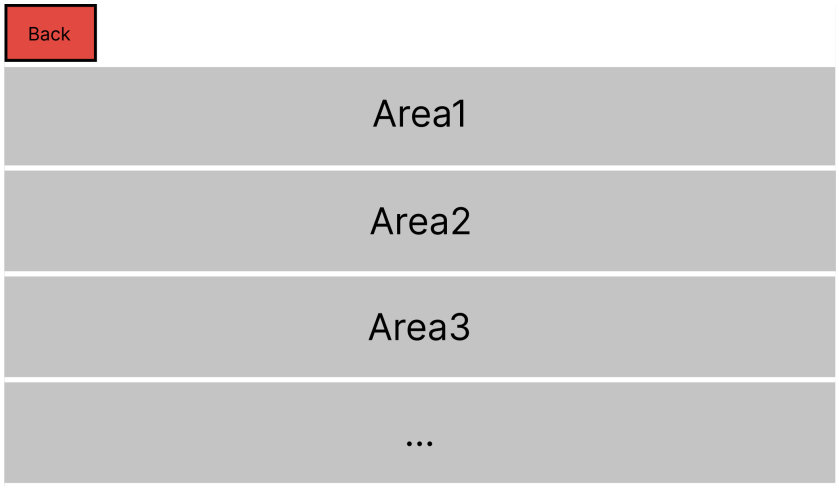
Základní menu Při vstupu do aplikace uživatel uvidí okénko s uvítáním, při stisknutí tlačítka *„Choose Area“* se dostane k výběru oblasti. Při vybírání oblasti, která existuje na API, ale ještě není přidána do aplikace, bude uživatel upozorněn a může se vrátit zpátky a vybrat jinou oblast (obrázky 3.7, 3.8 a 3.9).

Mapy Při prohlížení vybrané oblasti má uživatel k dispozici mapu, která se otevře při stisku tlačítka na pravém ovladači. Uživatel může vidět, kde se nachází na této mapě (obrázek 3.10).

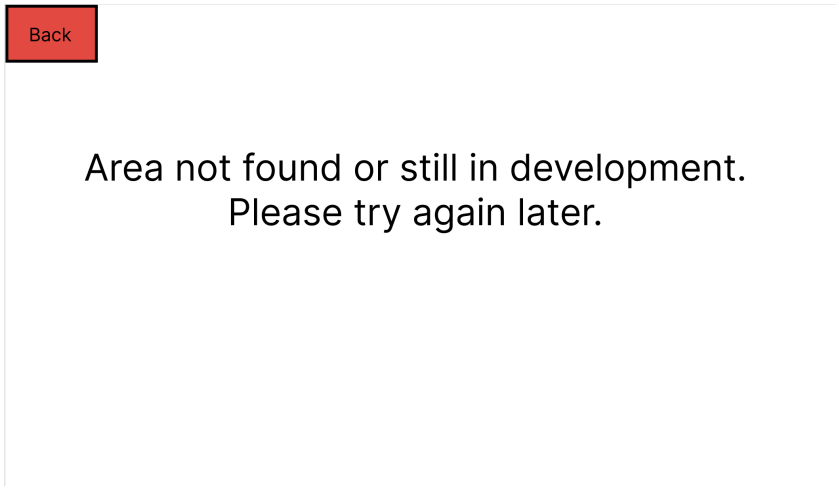
Informace o budově Při zmačknutí tlačítka na levém ovladači se zobrazí uživatelské rozhraní s informacemi o nejbližší budově, toto UI uživatel potom může zavřít (obrázek 3.11).



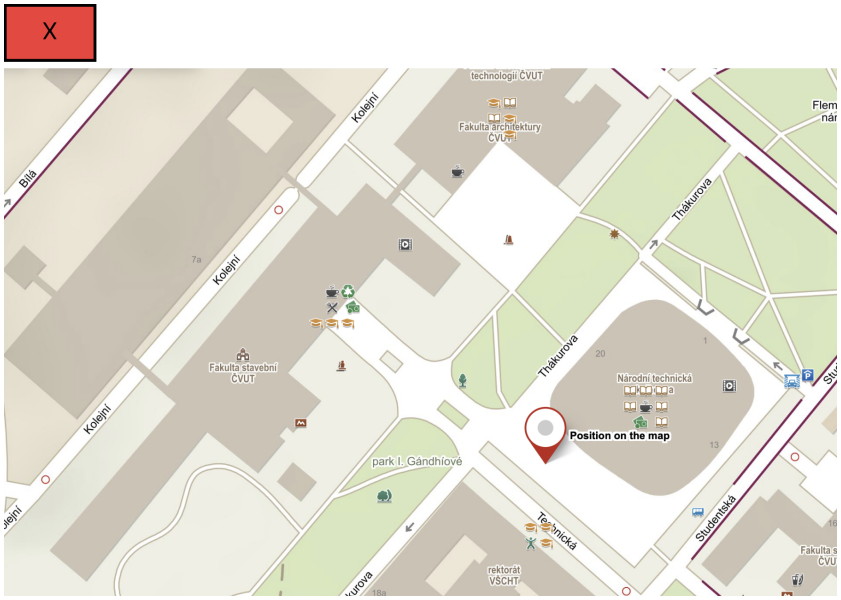
Obrázek 3.7: Ukázka návrhu uvítacího UI



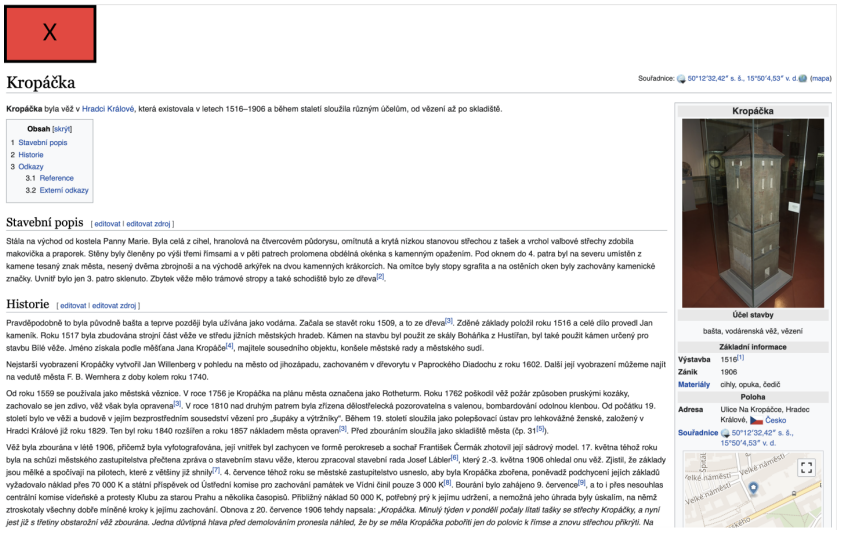
Obrázek 3.8: Ukázka návrhu UI s výběrem oblasti



Obrázek 3.9: Ukázka návrhu UI s upozorněním



Obrázek 3.10: Ukázka návrhu UI s mapou



Obrázek 3.11: Ukázka návrhu UI s informace o budově

3.2.4 Schéma komunikace s API

Prototyp aplikace potřebuje komunikovat s datovým úložištěm, a proto je třeba, aby aplikace disponovala technologií schopnou provádět REST komunikaci stejně jako zásuvný modul pro Blender. Tato aplikace bude používat pouze požadavek GET, protože potřebuje jenom získávat potřebná data z API. Schéma komunikace bude stejné jako výše předvedené na obrázku 3.6. Takový typ komunikace zajišťuje zásuvný modul VaRest.

3.2.5 Zásuvný modul VaRest

Zásuvný modul VaRest je jeden z nejpopulárnějších zásuvných modulů pro REST komunikaci s API pomocí vizuálního skriptování (Blueprints). Je dostupný na Epic Games Marketplace a je zdarma. Jedná se o modul, který poskytuje nástroje k vytváření všech typů požadavků (GET, POST, PUT atd.) a k jejich odesílání na server. Umožňuje vytváření a správu JSON objektů a umí nastavit hlavičku pro autentizaci. Má plnou podporu vlastností JSON: práce s různými typy dat, poli, obsahem binárních dat, obousměrná serializace na typ string. Jako odpověď vrací JSON pole, které lze přeformátovat do stringu [34]. Tento zásuvný modul bude zajišťovat REST komunikaci celého projektu.

3.2.6 Zásuvný modul RuntimeMeshLoader

Pomocí zásuvného modulu RuntimeMeshLoader bude implementováno načítání 3D modelů budov z datového úložiště. Jedná se o open-source zásuvný modul, který je napsán v C++ a používá knihovnu Assimp. Assimp je knihovna pro import a export různých formátů 3D modelů včetně následného zpracování scény pro generování chybějících dat renderování [35]. RuntimeMeshLoader vytváří nové typy blueprintu, ve kterých umožňuje zadání cesty k lokálnímu souboru s 3D modelem, který se pak načte v čase běhu programu. Podporuje téměř všechny existující 3D formáty.

Jedinou nevýhodou tohoto zásuvného modulu je nemožnost importovat 3D soubory, které nejsou uložené v lokálním souboru, ale jako typ string. Jelikož VaRest umí vytvářet jenom JSON objekty, které lze přeformátovat na string, je nutné dotvořit načítání modelů ze stringu. Druhou možností je implementace funkce, která ukládá model ze serveru do určitého lokálního souboru a dále ho načítá pomocí RuntimeMeshLoader.

3.2.7 Zásuvný modul Web Browser

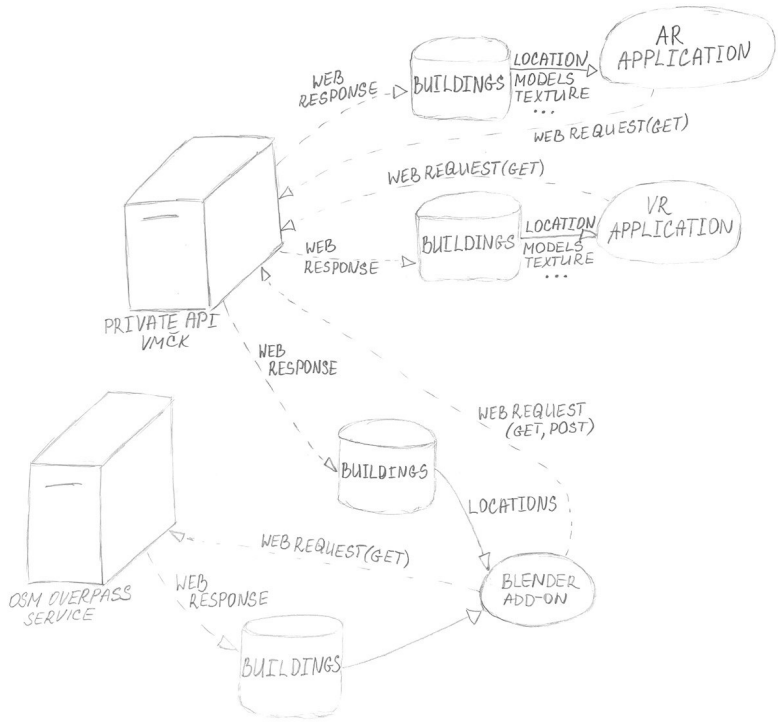
Zásuvný modul Web Browser je oficiálním modulem od Epic Games, který dovoluje vytvářet uživatelské rozhraní založené na webovém prohlížeči nebo HTML, CSS a JavaScriptu. Tento zásuvný modul je automaticky v projektu, pro použití je třeba ho povolit v záložce *Plugins*. Dále je třeba vytvořit nový

Widget Blueprint (blueprint pro uživatelské rozhraní) a přidat do daného elementu. V nastavení Web Browser má vlastnost **Initial URL**, kam je třeba vložit webový URL odkaz nebo lokální odkaz na .html soubor [36].

Tento zásuvný modul bude především použit pro zobrazení informací o vybrané budově. Správný webový odkaz je uložen na privátním API pod atributem *html_link*. Další možné využití tohoto zásuvného modulu je zobrazení mapy světa, která ukazuje přesnou polohu uživatele ve hře.

3.3 Závěr návrhu

V závěru je znázorněn návrh „big picture“ diagramu (vývojový diagram) celého projektu Věnná města českých královen (obrázek 3.12). Na obrázku jsou zobrazeny nejdůležitější prvky systému. Každý z výstupů používá privátní datové úložiště projektu VMČK, doplněk navíc posílá vygenerované budovy zpátky na privátní API (nebude implementováno v této bakalářské práci) a používá veřejné OpenStreetMap API. Tvorbou AR aplikace se momentálně zabývá kolega Tomáš Kiec. V této bakalářské práci se zabývám modifikací doplňku pro Blender a tvorbou VR aplikace.



Obrázek 3.12: Vývojový diagram projektu VMČK

Kapitola 4

Realizace

V této sekci budou popsány instalační, uživatelská a programátorská příručka pro modifikovaný zásuvný modul pro grafickou aplikaci Blender a pro VR klienta v herním enginu Unreal Engine 4. Instalační příručka popisuje postup, jak může uživatel získat a spustit modul nebo aplikaci. Uživatelská příručka je manuál, jak se modul nebo aplikace ovládá a jakou má dostupnou funkcionalitu. Programátorská příručka odpovídá popisu implementace a vybraných zajímavých způsobů řešení problémů.

4.1 Zásuvný modul Blender

4.1.1 Instalační příručka

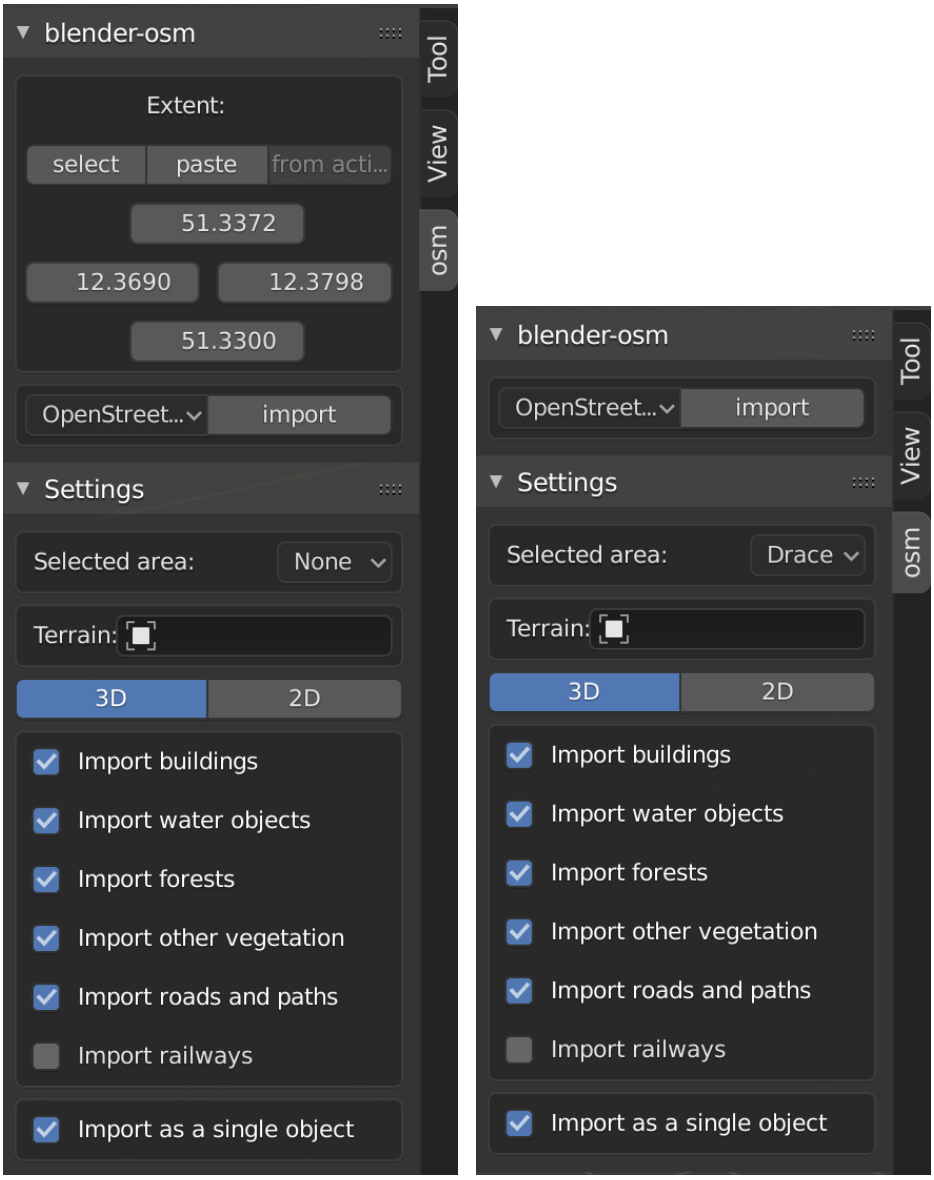
Pro nainstalování zásuvného modulu je třeba mít nainstalovanou grafickou aplikaci Blender, verze 2.83. Soubor pro stažení lze najít na oficiálních stránkách aplikace a je zdarma [37]. Dále je nezbytné stáhnout archivovaný soubor s implementovaným doplňkem. Lze jej nalézt v přiloženém obsahu v adresáři **src/impl/blender/vmckOsm.zip**. Po otevření Blenderu je třeba nainstalovat doplněk, v záložce nahoře *Edit→Preferences* se nachází rozdíl *Add- ons*, do kterého je zapotřebí nainstalovat archivovaný doplněk pomocí příkazu *Instal...*. Po nainstalování doplňku je zapotřebí jej povolit – zaškrtnou políčko vedle názvu doplňku a vybrat adresář, kam se budou stáhnuté soubory z OpenStreetMap API ukládat. Potom by se doplněk měl objevit v režimu 3D view v sekci Tools (jde otevřít pomocí klávesy N). Funkčnost doplňku je testovaná pro operační systémy MacOS a Windows.

4.1.2 Uživatelská příručka

Modifikované uživatelské rozhraní je znázorněno na obrázku 4.1. Na ukázce vlevo je znázorněno uživatelské rozhraní doplňku, kdy žádná generovaná oblast není vybrána. V tomto případě doplněk funguje stejně, jako je popsáno

v kapitole 3.1.1.1. Jedinou výjimkou jsou skryté funkce doplňku, například skryté jsou možnosti vybírání typu střechy nebo možnost generovat ze serveru nebo lokálně.

Na obrázku vpravo je znázorněno nové uživatelské rozhraní, které je dostupné při výběru jedné z nabízených oblastí načítaných z API. Skrytá je možnost ručního nastavení hranic, protože ty jsou nastavovány automaticky na základě dat z API.

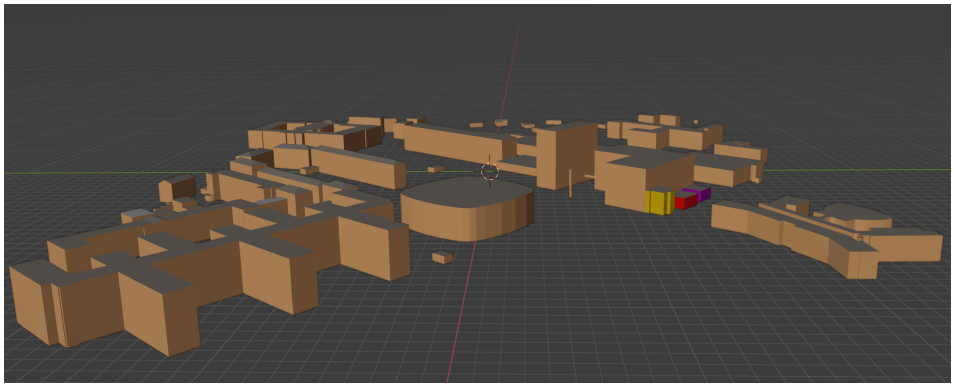


Obrázek 4.1: Nové uživatelské rozhraní Blender OSM

V obou případech si uživatel může vybrat, jestli chce generovat terén

nebo budovy, a co všechno má být vygenerováno (budovy, vodní objekty, lesy apod.), dále jaký bude formát (2D nebo 3D) a jestli vygenerované objekty budou jako jeden objekt nebo každý zvlášť.

Na obrázku 4.2 je ukázána generovaná oblast *Testovací zóna 1*, která je vzata z privátního API projektu Věnná města českých královen. Generovány byly pouze budovy jako jeden objekt. Jelikož nemá žádné historické budovy, které se protínají s generovanými, tak žádná budova nebyla odstraněna.



Obrázek 4.2: Generovaná Testovací zóna 1

4.1.3 Programátorská příručka

4.1.3.1 Verzování a komentování

Doplněk byl verzovaný pomoci Gitu v lokálním podadresáři pro lepší znázornění změn a možnost vrátit se zpět.

Kód je většinou komentovaný vývojáři, kteří původně implementovali doplněk. Přidaný kód je označen komentářem „Added by Hanna Korniushyna“. Komentářem byly opatřeny také části týkající se nepoužité funkcionality.

4.1.3.2 Uložení dat všech oblasti

Při spuštění aplikace se posílá REST API požadavek na získání jmen všech oblastí, na zjištění jejich polohy a struktur (budovy). Dále je znázorněn kód funkce, který tuto informaci získává a vrací items (jméno, které bude dále využito v uživatelském rozhraní, proto má takovou syntaxi), lokace, kam se ukládá název oblasti, dále minimální a maximální délka a šířka oblasti, na základě kterých se budou generovat budovy a struktury (budovy), kde se ukládá název oblasti, název struktury a její lokace.

```
1 def getAreas():
2     headers = {"Authorization": "BearerTokenID"}
3     r = requests.get('http://109.123.202.213:3000/areas',
4     headers = headers).json()
```

```
    location = [("None", 1,2,3,4)]
    items = [("None", "None", "None")]
    structures = [("areaName","name", 1,2)]
    for area in r:
        items.append(
            (area["name"], area["name"], area["name"])
            )
        lons = [
            area["activationBorder"][0]["lon"],
            area["activationBorder"][1]["lon"],
            area["activationBorder"][2]["lon"],
            area["activationBorder"][3]["lon"]
        ]
        lats = [
            area["activationBorder"][0]["lat"],
            area["activationBorder"][1]["lat"],
            area["activationBorder"][2]["lat"],
            area["activationBorder"][3]["lat"]
        ]
        lats.sort(reverse=True)
        lons.sort(reverse=True)
        location.append(
            (area["name"], lats[-1],
            lats[0], lons[-1], lons[0])
        )
        for struct in area["structures"]:
            if(struct["location"] != {}):
                structures.append(
                    (area["name"],
                    struct["name"],
                    struct["location"]["lat"],
                    struct["location"]["lon"])
                )
    return (items, location, structures)
```

Hodnoty lokace oblasti se předávají hodnotám souřadnic generované oblasti pouze tehdy, když v uživatelském rozhraní je vybraná nějaká oblast (není None):

```
if(addon.cities != "None"):
    areas = gui.getAreas()[1]
    for area in areas:
            if(addon.cities == area[0]):
                addon.minLon = area[3]
                addon.minLat = area[1]
                addon.maxLon = area[4]
                addon.maxLat = area[2]
```

4.1.3.3 Odstranění historických budov

Při obdržení odpovědi z OpenStreetMap API ji doplněk parsuje (rozebírá) po částech. Z obdržených odpovědí můžeme zjistit, jestli daná cesta je zamknuta, tzn., že začíná a končí ve stejném uzlu, nebo ne. Jestliže je zamknuta, pak tvoří

nějaký polygon a může označovat například budovu nebo jezero. Pokud je tato cesta polygonem, tak zůstává a bude se podle ní vykreslovat. V ostatních případech se odstraní. Existují i další důvody pro odstranění uzlu, cest a relace, například nevyhovující typ tagu.

V případě výběru nějaké oblasti z privátního API VMČK je nezbytné zkontrolovat všechny její významné budovy, zda se zde nejedná o kolizi s generovanými. Proto každá cesta, která je polygonem, se kontroluje. Ze všech uzlů, které jí naleží, musí být nalezeny minimální a maximální šířka a délka, aby pak byly porovnány se šířkou a délkou historické budovy. Pokud souřadnice budovy z API leží v souřadnicích generované budovy, tak cesta této generované budovy se odstraní a nebude vykreslovaná. Tento algoritmus je zobrazen na následujícím fragmentu kódu:

```
if way.valid:
    skip = False
    #Added by Hanna Korniushyna
    if(self.addon.cities != "None"):
        for structure in self.structures:
            if(self.addon.cities == structure[0]):
                minLon = 1000
                maxLon = 0
                minLat = 1000
                maxLat = 0
                for nodeId in way.nodes:
                    node = self.nodes[nodeId]
                    if node.lat < minLat:
                        minLat = node.lat
                    if node.lon < minLon:
                        minLon = node.lon
                    if node.lat > maxLat:
                        maxLat = node.lat
                    if node.lon > maxLon:
                        maxLon = node.lon
                if (structure[2] > minLat
                    and structure[2] < maxLat)
                    and (structure[3] > minLon
                    and structure[3] < maxLon):
                    skip = True
    #End of added code
```

4.1.3.4 Vzniklé problémy

Při modifikaci kódu doplňku vznikly dvě překážky, kvůli kterých vývoj trval déle:

- **Ladění kódu** Ladění kódu bez nainstalování doplňkového softwaru není možné. Softwary Visual Studio Code a Eclipse nabízí možnosti ladění doplňku Blender, ale jejich nainstalování je obtížné a ne vždy funguje správně. Proto při ladění doplňku Blender OSM nebyl použit žádný software na ladění a pro kontrolu se používalo vypisování v terminálu,

což bylo časově vysoce náročné, protože vždy bylo potřeba nejprve vymazat starý doplněk a následně nainstalovat jeho novou verzi.

- **Hledaní v již existujícím kódu** Hledání souboru pro modifikace bylo rovněž náročným úkolem, protože v celém projektu existuje téměř 50 různých souborů. I když většina z nich je dobře komentovaná, čas od času nebylo zcela jasné, ke kterému elementu patří jaký kód, a za co odpovídají některé z funkcí. Pro pochopení kódu bylo proto třeba jej ladit, což bylo časově velmi náročné.

4.2 VR klient

4.2.1 Instalační příručka

Pro spuštění aplikace je především potřeba mít správně nastavený headset a mít nainstalovanou aplikaci SteamVR. Prototyp zatím funguje jenom pro HTC Vive a HTC Vive Pro headsety.

Pro spuštění ze strany uživatele existuje v přiloženém obsahu archivovaný soubor **build.zip**, který obsahuje spustitelnou formu aplikace. Je třeba soubor rozbalit a spustit .exe soubor.

Pro spuštění ze strany programátora lze najít odkaz na fakultní GitLab, kde se nachází repozitář se zdrojovými kódy, v souboru **src/impl/unreal/link.txt**. Stačí naklonovat GitLab repositář na lokální počítač a spustit soubor **HistoricalGuide.uproject**. Před spuštěním není zapotřebí otevírat SteamVR, ten se otevře automaticky, stačí zkontrolovat, že headset je připojen až po spuštění Unreal Enginu.

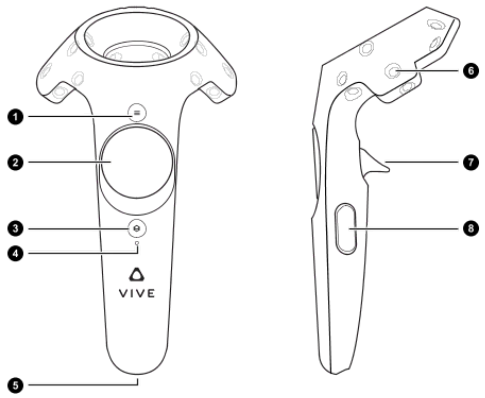
4.2.2 Uživatelská příručka

Prototyp obsahuje několik scén, první scéna je uvítací, každá další odpovídá jedné existující v API oblasti.

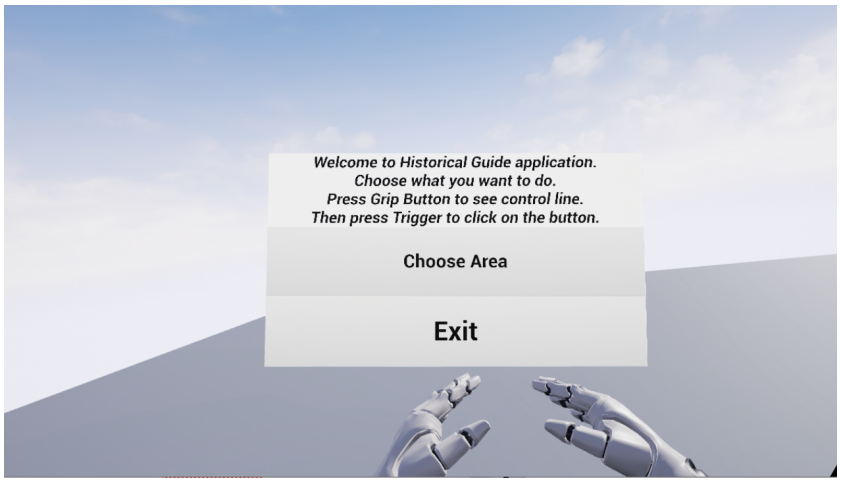
Na obrázku 4.3 jsou schematicky znázorněny ovladače HTC Vive. Pomocí čísel na obrázku bude popsána funkcionalita prototypu.

Před startem aplikace je nutné mít zapnuty oba ovladače. Je třeba zmáčknout tlačítko pod číslem 3 (System button) a zeleně se rozsvítí indikátor pod číslem 4 (Status light). Pokud jsou ovladače zprovozněny při spuštění aplikace, uvidí uživatel uvítací scénu s pohybujícíma se rukama (obrázek 4.4).

Pro interakci s uživatelským rozhraním existuje speciální paprsek, který se zobrazí při zmáčknutí tlačítka číslo 8 (nazývá se Grip button) pravou rukou. Pro zmáčknutí jakéhokoliv tlačítka na uživatelském rozhraní je třeba držet tlačítko číslo 8 a k tomu zmáčknout tlačítko číslo 7, které se nazývá Trigger. Při zmáčknutí tlačítka Choose Area se uživatel přemístí na výběr oblasti. Nabízeny jsou všechny oblasti, ale pokud oblast existuje na API, ale není úplně hotova v aplikaci, obdrží uživatel upozornění a musí vybrat nějakou



Obrázek 4.3: Ovladači HTC Vive [38]

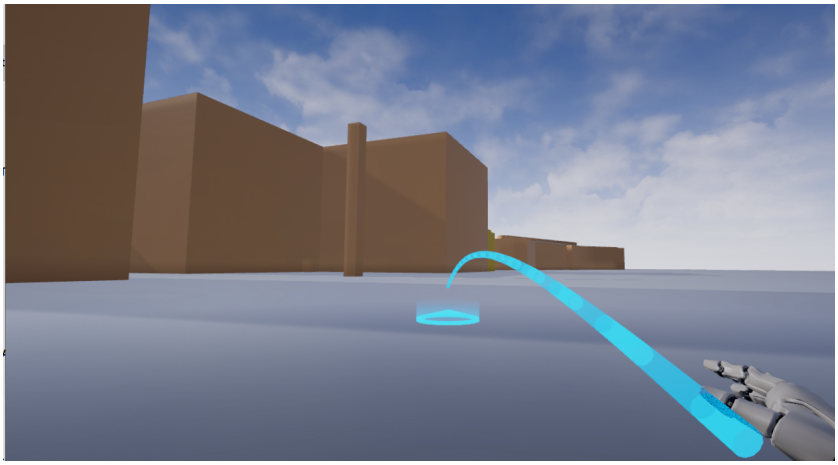


Obrázek 4.4: Uvítací scéna

jinou oblast. Zatím plně funkční a dostupná je jenom jedna oblast (Draček), pro další oblasti se vyvíjí API.

Po výběru oblasti se uživatel přemístí sem a může začít oblast zkoumat. Zatím budovy z API nenesou žádnou historickou hodnotu, jejich lokace není reálná a jsou zde pouze pro kontrolu správné funkčnosti aplikace. Kvůli málo vhodné prezentaci na API zde chybí i textury.

Pohyb v oblasti je řešen pomocí teleportu. Tlačítko číslo 2, které se nazývá Trackpad, odpovídá za teleport. Uživatel musí nejprve zmáčknout toto tlačítko, dále vybrat místo teleportu a směr. Směr se vybírá pomocí pohybu prstu po Trackpadu. Například, když má uživatel prst nahoře, tak směr pohybu je rovně (obrázek 4.5).



Obrázek 4.5: Ukázka teleportu

Kromě teleportace jsou dostupné další interakce:

- Při zmáčknutí tlačítka pod číslem 7 na pravém ovladači se otevře mapa prohlíženého prostředí (obrázek 4.6).
- Při zmáčknutí tlačítka pod číslem 7 na levém ovladači se otevře informace o budově, která se nachází blíž k uživateli (obrázek 4.7).
- Při zmáčknutí tlačítka pod číslem 1 se uživatel vrátí na uvítací scénu a může vybrat jinou oblast.

Obrázek 4.6: Ukázka mapy prostředí

Obrázek 4.7: Ukázka informace o budově

Další ukázky chodu aplikace lze najít v přiloženém obsahu ve složce **previews**.

4.2.3 Programátorská příručka

Většina implementace se nachází v blueprintu **BP_GameModeBase** nebo ve Widget Blueprints (blueprinty, které řeší uživatelské rozhraní aplikace). BP_GameModeBase definuje aplikaci a její pravidla. Pro každou úroveň se obnovuje, což je velmi výhodné pro tuto aplikaci, protože má hodně podobných úrovní s podobnou funkcionalitou, kde každý odpovídá jedné oblasti z API. Globální proměnné se ukládají v **BP_GameInstance**, neaktualizují se při změně úrovně, a proto jsou vhodné pro ukládání proměnných takového typu. Za nastavení, pozice a pohled uživatele odpovídá blueprint **MotionControllerPawn**, k tomu provádí teleportace uživatele, většina funkcionality byla převzata z VR šablony. Blueprint **BP_MotionController** reaguje na všechny interakce uživatele (kromě teleportu) s ovladačem a zpracovává je.

4.2.3.1 Modifikace RunTimeMeshLoader

Pro načítání modelu z API byl použit upravený zásuvný modul RunTimeMeshLoader. Ten byl přidán do složky zásuvných modulů projektu. Bylo třeba přidat možnost načítání modelu ze stringu. Do typu načítaných souborů byl přidán typ FromString:

```
1 UENUM(BlueprintType)
2 enum class EPathType : uint8
3 {
4   Absolute,
5   Relative,
6   /*Added by Hanna Korniushyna*/
7   FromString
```

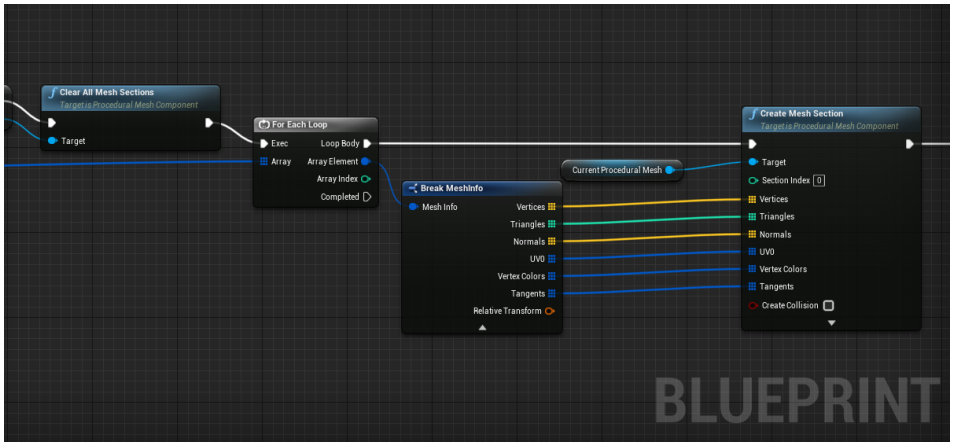
```
  /*end of added code*/
};
```

Dále bylo přidáno načítání ze stringu. Použita byla knihovna Assimp, funkce ReadFileFromMemory(), která přečte daný soubor z buffer paměti a v případě úspěchu vrátí jeho obsah:

```
    /*Added by Hanna Korniushyna*/
  if (type == EPathType::FromString) {
    mScenePtr = mImporter.ReadFileFromMemory(
                    file.c_str(),
                    file.size(),
                    aiProcess_Triangulate |
                    aiProcess_MakeLeftHanded |
                    aiProcess_CalcTangentSpace |
                    aiProcess_GenSmoothNormals |
                    aiProcess_OptimizeMeshes
                );
  }
  /*end of added code*/
```

Po všech nutných úpravách vytvořených zásuvným modulem Blueprint byly funkce použity v projektu. Zásuvný modul podporuje pouze modely formátu .obj nebo jakýkoliv jiný formát, který se ukládá jako soubor stringu, nikoliv jako binární soubor.

Jedné budově odpovídá blueprint **BP_RuntimeMeshImport** typu Actor. Obsahuje Procedural Mesh komponentu, která umožňuje vytvářet 3D objekty trojúhelníků, vrcholů a normál. Pro každou budovu z API je nutné přidat jeden takový blueprint do scény. V současné době jsou zatím čtyři, při přidání dalších není třeba kód měnit. Na obrázku 4.8 je znázorněna část funkce GenerateModel(), která postupně za chodu programu vytváří budovy z API. Zobrazená část odpovídá za vytváření geometrie Procedural Mesh komponenty.

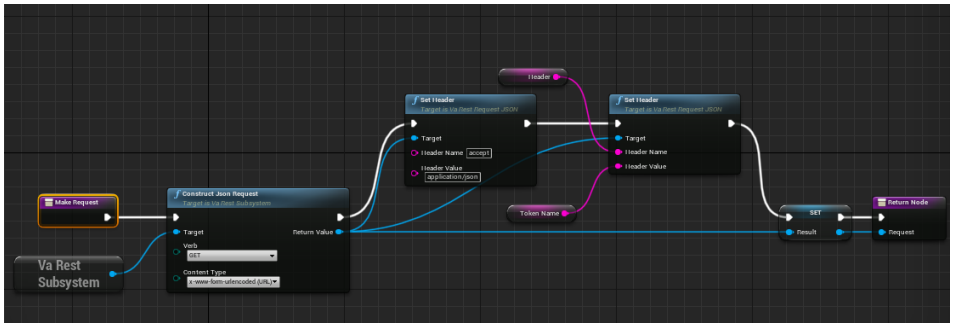


Obrázek 4.8: Část funkce GenerateModels()

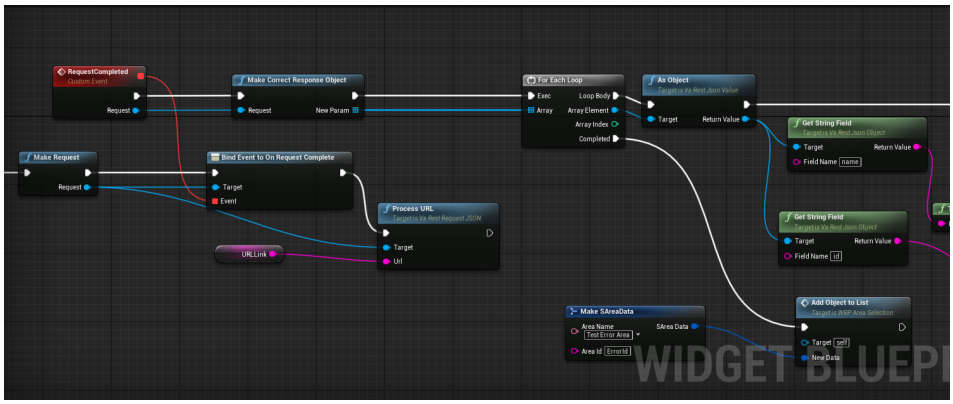
Dále se nastavuje materiál modelu. Používá se dynamický materiál, protože se do budoucna předpokládá, že materiál se bude měnit podle dat z API. Existuje jeden základní materiál, který se jmenuje **M_Base**, obsahuje texturu, která se může měnit za chodu programu. Zatím všechny budovy z API jsou ze stejného šedého materiálu, protože se zjistilo, že textury na API jsou uloženy pro tento projekt nevhodně.

4.2.3.2 Komunikace s datovým úložištěm

Komunikace s datovým úložištěm byly provedeny pomocí zásuvného modulu VaRest v Blueprints. Jako první se vytváří autorizace a request (žádost), zpracovává se zadaný URL. Když je žádost dokončena, získává se odpověď typu reference na objekt JSON. Z odpovědí lze získat hodnoty podle jména atributu. Na obrázku 4.9 je ukázána funkce MakeRequest(), která sestavuje žádost a nastavuje autorizační token. Na obrázku 4.10 je uveden příklad využití funkce MakeRequest() a získávání hodnot a atributu z odpovědí ze serveru.



Obrázek 4.9: Funkce MakeRequest()

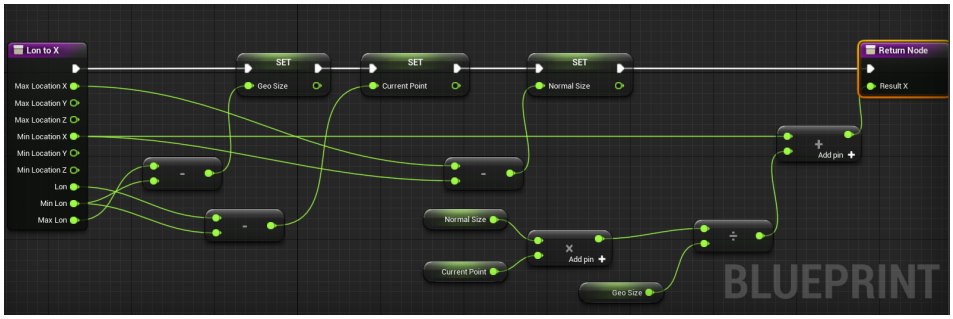


Obrázek 4.10: Ukázka použití funkce MakeRequest()

Tento příklad ukazuje získávání jména a id každé oblasti. Stejný princip komunikace je využit i ve všech dalších API žádostech. Mění se URL žádosti a typy hledaných atributů, které mohou být typu string, number, object apod.

4.2.3.3 Převod světových souřadnic

Startovací pozice hráče a lokace významných budov jsou uvedeny na API pomocí světových souřadnic. Je třeba jejich promítnutí na souřadnice X a Y každé úrovně. Délka odpovídá pozici X a šířka pozici Y. Pro tento převod je nezbytné znát velikost vybrané oblasti, maximální a minimální délku a šířku oblasti a délku a šířku konkrétní budovy, souřadnice, které chceme převést do souřadnic dané úrovně. Většinu hodnot dostaneme z API, postup je podobný, jako je popsáno v předchozí sekci. Velikost vybrané oblasti lze získat z komponent minValue a maxValue, které jsou součástí Floor (statický model podlahy). Na obrázku 4.11 je ukázána funkce LonToX(), která převádí délku na pozici X ve vybrané oblasti, stejná funkce existuje i pro převod šířky na pozici Y a nazývá se LatToY().



Obrázek 4.11: Funkce LonToX()

4.2.3.4 Uživatelské rozhraní

Uživatelské rozhraní není statické, ale je pohyblivé podle pohledu uživatele. Za chodu programu se vytváří **BP_UserInterfaceActor** typu Actor, nastavuje se jeho pozice (vždy záleží na pohledu uživatele) a jeho viditelnost. Obsahuje komponentu Widget, kterou je možné provádět změny během chodu programu. Pro zobrazení nějakého prvku uživatelského rozhraní stačí nastavit viditelnost **BP_UserInterfaceActor** na True a komponentu Widget na vybraný prvek.

Pro interakci uživatele s uživatelským rozhraním je třeba přidat do **BP_MotionController** komponentu WidgetInteraction, která vytvoří interaktivní paprsek a umožňuje uživateli používat prvky uživatelského rozhraní.

4.2.3.5 Verzování

Verzování aplikace bylo prováděno pomoci fakultního GitLabu, přístup ke zdrojovému kódu zajišťuje vedoucí této bakalářské práce. Prototyp se průběžně ukládal v lokálním nebo webovém repositáři GitLabu.

Kapitola 5

Testování

Testování VR aplikace není jednoduchý úkol a pro jeho úspěšné zvládnutí je potřebná důkladná příprava, časově je rovněž velmi náročné. Protože výstupy z této bakalářské práce budou použity i jako základ finální VR aplikace celého projektu Věnná města českých královen, bylo nezbytné, aby před finálním testováním aplikace tato splňovala nejenom požadavky této bakalářské práce, ale i celého projektu. Proto po konzultaci s vedoucím práce bylo rozhodnuto, že testování bude probíhat postupně v několika etapách na akcích, které se budou konat na FIT ČVUT v období konec května až konec červen 2022 (Praxe SPŠ na FIT v 2022, Den dětí na FIT ČVUT a VědaFest). V této bakalářské práci budou představeny pouze testovací scénáře pro budoucí testování.

Testování bude provedeno v UsabilityLabu na FIT ČVUT a bude zaměřeno na ověřování implementovaných funkcionalit. Účastnicí obdrží instrukce k funkcionalitě ovladačů a dva dotazníky, jeden vyplní před testováním a druhý po jeho skončení, a testovací instrukce.

5.1 Dotazníky

Součástí testování jsou dva dotazníky. První se vyplňuje před zahájením testování, druhý po jeho ukončení. Před testováním budou položeny následující otázky:

- Jaké jsou Vaše zkušenosti z aplikacemi pro VR?
- Jaký je Váš věk (do 10 let, 11 až 20 let, 21 až 40 let, nad 40 let)?
- Máte nějakou oční vadu?
- Trpíte často kinetózou?

Tyto otázky slouží především pro získání informací o zkušenostech testujícího s užíváním aplikací a o jeho zdravotním stavu.

Následující otázky budou položeny po ukončení testování:

- Jaký jste měli pocit z použití headsetu?
- Bavilo Vás prohlížení virtuálního prostředí?
- Je podle Vás uživatelské rozhraní intuitivní a je jeho používání snadné?
- Je podle Vás tato aplikace zajímavá z historického pohledu?
- Co se Vám na aplikaci líbilo/nelíbilo?

5.2 Testovací scénář

Účastníci obdrží dva testovací scénáře. První je zaměřen na testování orientace uživatele v prostředí. Druhý je zaměřen na testování interakce s uživatelským rozhraním aplikace.

5.2.1 Orientace v prostředí

Odhadovaný čas
5 min.

Účel testování
Testujeme orientaci v prostředí virtuálního města, zda je uživatel schopen se v něm pohybovat a zda umí nalézt historickou budovu v systému.

Počáteční bod
Uživatel je připraven spustit aplikaci, má nasazený headset.

Koncový bod
Uživatel se přesune do vybrané oblasti.

Instrukce pro testujícího
Zkontrolujte, zda máte headset dobře nasazen, zapněte aplikaci. Přesuňte se do oblasti s věží Kropáčka a poté se přesuňte do oblasti Městská brána pomocí teleport portálu.

Očekávané kroky

1. Uživatel vybere oblast Kropáčka.
2. Uživatel se rozhlédne.
3. Uživatel najde teleport portál pro přesun do oblasti Městská brána.
4. Přesune se na pozici správného teleportu.
5. Je v oblasti Městská brána.

5.2.2 Interakce s uživatelským rozhraním

Odhadovaný čas
5 min.

Účel testování
Testujeme interakce s uživatelským rozhraním, zda všechna tlačítka fungují, zda se všechno zobrazuje správně a zda je rozhraní uživatelsky přívětivé.

Počáteční bod
Uživatel má spuštěnu aplikaci, má správně nasazený headset a nachází se na uvítací scéně.

Koncový bod
Po testování se uživatel vrací na uvítací scénu.

Instrukce pro testujícího
Důkladně nastudujte informace o dostupné funkcionalitě ovladačů. Vyberte jednu z nabízených oblastí, vyzkoušejte všechny možné interakce a vraťte se na uvítací scénu.

Očekávané kroky

1. Uživatel vybere jednu z dostupných oblastí.
2. Pokud je vybraná oblast stále ve vývoji, bude na to upozorněn a vrátí se zpátky.
3. Zkusí otevřít a zavřít informace o budově.
4. Zkusí otevřít a zavřít mapu prostředí.
5. Vrátí se na uvítací scénu.

Závěr

Tato bakalářská práce se zabývala zobrazením historického prostředí ve virtuální realitě pomocí dat z API. Součástí práce je modifikace zásuvného modulu pro generování budov pro grafický editor Blender. Generované budovy byly použity v prototypu VR klienta. Bakalářská práce je součástí projektu Věnná města českých královen.

Byla provedena analýza závěrečných prací, které se věnují stejnému projektu a které byly použity v této bakalářské práci. Prototyp používá privátní datové úložiště projektu Věnná města českých královen, proto dalším úkolem byla analýza současného stavu API a návrh potřebných změn. Dále se práce zabývala analýzou vývoje virtuální reality, porovnávány byly dva populární herní enginy Unity 3D a Unreal Engine a na základě stanovených kritérií byl vybrán ten vhodnější. Jelikož výstupem práce je i modifikovaný zásuvný modul, analýza se zabývala způsoby rozšíření aplikace Blender. Na konci analýzy byly sepsány funkční a nefunkční požadavky, které byly inspirovány předchozími pracemi a které byly konzultovány s vedoucím práce.

Návrh obsahoval porovnání dvou již existujících zásuvných modulů pro generování budov. Byl vybrán doplněk Blender OSM a k němu byly dopracovány nutné modifikace. Dále byl navržen prototyp VR klienta, který obsahoval popis nutných nastavení a použitých zásuvných modulů, a návrh uživatelského rozhraní. Na konci byl představen vývojový diagram celého projektu „Věnná města českých královen“.

Implementace obsahuje modifikovaný zásuvný modul, který obsahuje všechny funkční a nefunkční požadavky, a prototyp VR aplikace tvořený pro headsety HTC Vive a HTC Vive Pro. VR klient zatím nemá implementovány teleport portály a mapu prohlíženého prostředí (chybí správná ukázka místa, kde se uživatel nachází).

Testování zatím nebylo provedeno kvůli vysoké náročnosti přípravy. Tato bakalářská práce obsahuje testovací scénáře a testování bude provedeno až po jejím odevzdání.

Na závěr je možno konstatovat, že téměř všechny cíle této bakalářské práce

se podařilo naplnit a vývoj VR aplikace bude pokračovat i po odevzdání práce.

Literatura

[1] *V*ěnná města českých královen: O projektu, 2022, [cit. 2. dubna 2022]. Dostupné z: https://www.kralovskavennamesta.cz/about.html

[2] Křepinský, Patrik. *Věnná města českých královen - Úprava 3D modelů ve virtuální realitě*. Bakalářská práce, České vysoké učení technické v Praze, Fakulta informačních technologií, 5 2019.

[3] Matoušková, Klára. *Věnná města českých královen - Interakce v historickém městě ve virtuální realitě*. Bakalářská práce, České vysoké učení technické v Praze, Fakulta informačních technologií, 5 2021.

[4] Klofáč, Marek. *Věnná města českých královen - Schvalovací systém pro 3D modelu ve virtuální realitě*. Bakalářská práce, České vysoké učení technické v Praze, Fakulta informačních technologií, 6 2021.

[5] Žižková, Alena. *Dowry Towns of the Queens of Bohemia - Digitization of Historical Clothing*. Bakalářská práce, České vysoké učení technické v Praze, Fakulta informačních technologií, 5 2021.

[6] Vančura, Daniel. *Věnná města českých královen - jádro*. Bakalářská práce, České vysoké učení technické v Praze, Fakulta informačních technologií, 2020.

[7] Bc. Martinek, Michal. *Administrační rozhraní pro projekt Věnná města českých královen*. Diplomová práce, České vysoké učení technické v Praze, Fakulta informačních technologií, 5 2020.

[8] Bc. Sivák, Dominik. *Věnná města českých královen - Backend administrační části*. Diplomová práce, České vysoké učení technické v Praze, Fakulta informačních technologií, 5 2021.

[9] Slovník IT pojmů: Application programming interface, 2022, [cit. 20. dubna 2022]. Dostupné z: https://www.becorp.cz/slovnik/application-programming-interface/

[10] Webový prohlížeč Google, 2022. Dostupné z: https://www.google.cz

[11] Jana, Abhijit; Sharma, Manish; Rao, Mallikarjuna. *HoloLens Blueprints.* [online], 2017. Dostupné z: https://www.packtpub.com/product/hololens-blueprints/9781787281943

[12] Concept Art Empire: What is Unity 3D & What is it Used For?, 2022, [cit. 30. dubna 2022]. Dostupné z: https://conceptartempire.com/what-is-unity/

[13] SteamVR Unity Plugin Documentation, 2022, [cit. 2. května 2022]. Dostupné z: https://valvesoftware.github.io/steamvr_unity_plugin/

[14] Juránek, Vojtěch. *Virtual reality toolkit for the Unity game engine.* Bakalářská práce, Masarykova Univerzita, Fakulta informatiky, 2021.

[15] VRTK: Documentation, 2022, [cit. 3. května 2022]. Dostupné z: https://vrtoolkit.readme.io/docs

[16] Unity Documentation: XR Plug-in Framework, 2022, [cit. 3. května 2022]. Dostupné z: https://docs.unity3d.com/2019.4/Documentation/Manual/XRPluginArchitecture.html

[17] Unity Documentation: XR Interaction Toolkit, 2022, [cit. 3. května 2022]. Dostupné z: https://docs.unity3d.com/Packages/com.unity.xr.interaction.toolkit@2.0/manual/architecture.html

[18] XR Bootcamp: The best 5 VR SDKs for Interactions for Unity & Unreal, 2022, [cit. 3. května 2022]. Dostupné z: https://xrbootcamp.com/the-best-5-vr-sdk-for-interactions/#headline-19-465

[19] Concept Art Empire: What is Unreal Engine?, 2022, [cit. 3. května 2022]. Dostupné z: https://conceptartempire.com/what-is-unreal-engine/

[20] Mack, Kevin; Ruud, Robert. *Unreal Engine 4 Virtual Reality Projects: Build immersive, real-world VR applications using UE4, C++, and Unreal Blueprints.* 1. vydání. Packt Publishing, 2019. ISBN 9781789132878.

[21] Circuit Stream: Unity vs Unreal Engine for XR Development: Which One Is Better? 2022, [cit. 3. května 2022]. Dostupné z: https://circuitstream.com/blog/unity-vs-unreal/

[22] 4Experience: Unity vs Unreal comparison: Choose the best engine for you, 2021, [cit. 4. května 2022]. Dostupné z: https://4experience.co/unity-vs-unreal-comparison-choose-the-best-engine-for-you/

[23] docs.blender.org: Getting Started, 2022, [cit. 16. dubna 2022]. Dostupné z: https://docs.blender.org/manual/en/latest/getting_started/

[24] docs.blender.org: Blender 3.1 Python API Documentation, 2022, [cit. 18. dubna 2022]. Dostupné z: https://docs.blender.org/api/current/

[25] Ing. Jiří Chludil. *3D grafika - Blender: Úvod, metodika, využití jazyka Python* [přednáška]. České vysoké učení technické v Praze, ZS 2020/2021.

[26] docs.blender.org: Add-on Tutorial, 2022, [cit. 18. dubna 2022]. Dostupné z: https://docs.blender.org/manual/en/latest/advanced/scripting/

[27] GitHub: Blender OSM Documentation, 2022, [cit. 19. dubna 2022]. Dostupné z: https://github.com/vvoovv/blender-osm/wiki/Documentation

[28] blender-osm: OpenStreetMap and Terrain for Blender, 2022, [cit. 20. dubna 2022]. Dostupné z: https://prochitecture.gumroad.com/l/blender-osm

[29] github.com: Blender GIS, 2021, [cit. 27. dubna 2022]. Dostupné z: https://github.com/domlysz/BlenderGIS

[30] OSM Wiki, 2022, [cit. 6. května 2022]. Dostupné z: https://wiki.openstreetmap.org/wiki/Cs:Prvky

[31] TechTarget: REST API (RESTful API), 2020, [cit. 5. května 2022]. Dostupné z: https://www.techtarget.com/searchapparchitecture/definition/RESTful-API

[32] W3Schools: Python Requests Module, 2022, [cit. 4. května 2022]. Dostupné z: https://www.w3schools.com/python/module_requests.asp

[33] Figma, 2022, [cit. 6. května 2022]. Dostupné z: https://www.figma.com

[34] Alyamkin, V. VaRest, 2022, [cit. 5. května 2022]. Dostupné z: https://github.com/ufna/VaRest

[35] GitHub: Open Asset Import Library (assimp), 2022, [cit. 6. května 2022]. Dostupné z: https://github.com/assimp/assimp

[36] Web Browser Widget, 2021, [cit. 5. května 2022]. Dostupné z: https://unrealcommunity.wiki/web-browser-widget-13f406

[37] Blender, 2022, [cit. 8. května 2022]. Dostupné z: https://www.blender.org/download/

[38] Vive Support: About the VIVE controllers, 2022, [cit. 9. května 2022]. Dostupné z: https://www.vive.com/us/support/vive/category_howto/about-the-controllers.html

Příloha A

Seznam použitých zkratek

3D Three-dimensional - trojrozměrný

2D Two-dimensional - dvojrozměrný

API Application programming interface - rozhraní pro programování aplikace

AR Augmented reality - rozšířená realita

HTTP Hyper-Text Transfer Protocol

MR Mixed Reality - smíšená realita

NPC Non-player character - nehratelná postava

REST Representational State Transfer - převod reprezentativního stavu

SDK Software development kit - sada pro vývoj softwaru

VMČK Věnná města českých královen

VR Virtual reality - virtuální realita

UE4 Unreal Engine 4

UI User Interface - uživatelské rozhraní

XR Extended reality - rozšířená realita

Příloha **B**

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `build.zip` ..... archiv se spustitelnou formou implementace
- `src`
  - `impl` ..... zdrojové kódy implementace
    - `blender` ..... zdrojový kód modifikovaného zásuvného modulu
    - `unreal` ..... zdrojové kódy prototypu VR klienta
      - `link.txt` ..... odkaz na vzdálený repositář se zdrojovými kódy
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF
- `previews` ..... snímkové ukázky výsledné implementace